Product Documentation

Embarcadero® Rapid SQL™

User Guide

Version XE 2/8.0

1st Edition June, 2011

Embarcadero Technologies, Inc. is a leading provider of award-winning tools for application developers and database professionals so they can design systems right, build them faster and run them better, regardless of their platform or programming language. Ninety of the Fortune 100 and an active community of more than three million users worldwide rely on Embarcadero products to increase productivity, reduce costs, simplify change management and compliance and accelerate innovation. The company's flagship tools include: Embarcadero® Change Manager™, CodeGear™ RAD Studio, DBArtisan®, Delphi®, ER/Studio®, JBuilder® and Rapid SQL®. Founded in 1993, Embarcadero is headquartered in San Francisco, with offices located around the world. Embarcadero is online at www.embarcadero.com.

Contents

WELCOME TO RAPID SQL

Rapid SQL is an integrated cross-platform database development environment that provides a highly-intuitive and well-integrated interface. Its graphical facilities simplify SQL scripting, object management, reverse-engineering, database project management, version control, and schema deployment. Additionally, it provides comprehensive tools for tuning and editing code to ensure high-performance and quality.

Rapid SQL offers support for all major databases including IBM DB2, InterBase/Firebird, Microsoft SQL Server, MySQL, Oracle, and Sybase.

The table below describes the major sections of Help.

Section	Description
Getting Started	Provides an introduction for new users and helps you set up Rapid SQL. Included is an investigation of the major user interface elements, detailed tutorial exercises, and instructions on configuring Rapid SQL features.
Datasource Management	Shows you how to register the datasources that Rapid SQL will work against. This chapter also describes the extensive datasource management facilities.
Database Object Management	Describes how to create new database objects, edit existing objects, and provides details on the operations you can perform against database objects.
SQL Scripting	Describes the development/execution/testing environments offered.
Tools	Tools provide search, script and file execution, scheduling, visual difference, a visual query builder, data import, a DML generation facility, a data editor, and an auto-replace tool for the scripting environment.
Project Management	Rapid SQL provides project support and version control system integration.
Rapid SQL Add-On Tools	License-dependent add-ons include: **Debuggers** - Rapid SQL provides debugging environments for DB2 LUW, SQL Server, Oracle, and Sybase. **Oracle PL/SQL Profiler** - captures metrics of various PL/SQL programmable objects as they are executed **Code Analyst** - helps you identify time-consuming lines of code.

ADDITIONAL PRODUCT RESOURCES

The Embarcadero Web site is an excellent source for product information and access to additional resources:

- Release notes and product documentation

- Information on other Embarcadero application and database development tools

- Embarcadero Development Network

- Product and technical support

- Educational and product expertise material in the form of online demos, white papers, and video presentations

- Access to sales staff and Embarcadero resellers

For access, direct your browser to www.embarcadero.com.

PREREQUISITES AND PRELIMINARY TASKS

The following topics provide information on requirements that must be satisfied and initial setup tasks that must be performed before using Rapid SQL:

- [Technical Requirements](#)

- [Database Support](#)

- [Licensing](#)

TECHNICAL REQUIREMENTS

Rapid SQL is a 32-bit application that runs in a Microsoft Windows environment. Before using Rapid SQL, please verify that your environment meets the following requirements:

- [Hardware requirements](#)

- [Operating system requirements](#)

    NOTE: Users need full registry privileges during the installation and access to the keys under HKEY_CURRENT_USER in the registry after installation.

HARDWARE REQUIREMENTS

Embarcadero Technologies recommends the following minimum hardware requirements:

- 1024 MB of memory

- 512 MB of disk space

OPERATING SYSTEM REQUIREMENTS

Rapid SQL supports the following operating systems:

- Windows 7 (32-bit and 64-bit)

- Windows XP (32-bit and 64-bit)

- Windows Vista (32-bit and 64-bit)

For more detailed information, see the following topics:

- [XP Support](#)

- [Vista and Windows 7 Support](#)

- [32-bit Versus 64-bit Operating System Support and Restrictions](#)

XP SUPPORT

Windows XP has two user security groups:

- Users

- Power Users

Microsoft intentionally does not grant members of the Users Group the authority to install applications or make global changes to the system. Restricted Users are members of the Users Group. Standard users belong to the Power Users Group. Microsoft grants members of the Power Users Group the authority to install programs. You must be a member of the Administrators Group in order to install and use Embarcadero Technologies applications.

Because Restricted Users are members of the Users Group, they cannot install and run Embarcadero Technologies applications.

CAUTION: You must be a member of the Administrators Group in order to install and use Embarcadero Technologies applications.

To open the Group Membership tab where you can determine your group and review the Microsoft security guidelines:

1   On the **Control Panel**, open **User Accounts**.

2   On the **Users** tab, select a user and then click the **Properties** button.

3   Click the **Group Membership** tab.

VISTA AND WINDOWS 7 SUPPORT

Windows Vista UAC and Windows 7 provide two user types:

- Standard user

- Administrator

Rapid SQL can be installed or uninstalled by an administrator or by a standard user using an administrator token. Standard users can run Rapid SQL.

CAUTION: For the purposes of running Rapid SQL, default standard user token privileges should not be modified. Modifying standard user token privileges can result in licensing issues which will prevent Rapid SQL from operating properly.

32-BIT VERSUS 64-BIT OPERATING SYSTEM SUPPORT AND RESTRICTIONS

Rapid SQL is a 32-bit application that runs emulated on 64-bit versions of Windows. The only requirement is that Rapid SQL supports only 32-bit versions of the client software that must be installed for connectivity to each DBMS that Rapid SQL will work against.

DATABASE SUPPORT

The table below describes the database platforms Rapid SQL supports and the server and client requirements:

Platform	Server	Client
IBM DB2 for z/OS	IBM DB2 for z/OS 8 and 9.	When going through a DB2 Connect gateway, DB2 Client for Windows 6.0 or later. When going directly to the mainframe from the client, DB2 Connect Personal Edition v6 or higher on the client machine.
IBM DB2 for Linux, Unix, and Windows	IBM DB2 Universal Database 8.0 - 9.7.	IBM DB2 Client for Windows 7.2 or later. **NOTE**: When using a v8 client, Rapid SQL only supports connecting to a v8 Database on Windows, Linux and Unix.
InterBase/Firebird	InterBase 2007 and InterBase 2009	DataDirect ODBC driver for InterBase and the InterBase Client
	Firebird 2.0	Firebird ODBC Driver
Microsoft SQL Server	Microsoft SQL Server 2000, 2005, and 2008 **NOTE**: For SQL Server 2005 and 2008, only object types like users, logins, roles, config parameters, etc., are supported. We do NOT support Microsoft .NET extensions at this time.	Microsoft SQL Server Client Library
MySQL	MySQL 4.x. (Rapid SQL is tolerant of 5.x but does not support Stored Procedures, Triggers, Views.)	MySQL ODBC driver 3.5.1 and above
Oracle	Oracle 8i, 9i, 10g, and 11g	Oracle SQL*Net Client
Sybase ASE	Sybase ASE 12.5 - 15.0.3	Sybase Open Client

> **NOTE:** Since Rapid SQL is a 32-bit application that runs emulated on 64-bit versions of Windows, only 32-bit versions of the DBMS client software are supported.

IBM DB2 FOR Z/OS STORED PROCEDURE REQUIREMENTS

When working against an IBM DB2 for z/OS data source, Rapid SQL relies on the following stored procedures, provided as an optional installation step in setting up the DB2 subsystem:

- DSNWZP
- DSNUTILS

- ADMIN_COMMAND_DSN

- ADMIN_COMMAND_DB2

Prior to using Rapid SQL against an IBM DB2 for z/OS data source, ensure that these components are installed on the server. See http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.db2.doc.inst/stpadd.htm#stpadd for more information.

LICENSING

Each Embarcadero client application requires one or more licenses in order to run. An Embarcadero product, such as Rapid SQL or ER/Studio, has a baseline license which provides basic feature support for that product. In addition, incremental licenses may be required to support specific DBMS platforms, product add-ons, or other functions.

For more information, see the following topics:

- Licensing Overview

- Licensing your Application

- Selecting a License Cateory During Startup

- Online/Offline Mode and Concurrent License Checkout

LICENSING OVERVIEW

The following topics provide a high-level discussion of key licensing topics and directs you to sources of more detailed information.

- Viewing your License Type and Modules

- Understanding Trial, Workstation, and Networked Licenses

- Rapid SQL License Modules, DBMS Support, and Feature Availability

- Rapid SQL XE License Modules, DBMS Support, and Feature Availability

- Directing Queries Regarding Licenses

VIEWING YOUR LICENSE TYPE AND MODULES

The **About...** dialog, available from the Help menu, displays your license type and each individual license module currently registered.

UNDERSTANDING TRIAL, WORKSTATION, AND NETWORKED LICENSES

Three kinds of licenses are available: Trial, Workstation, and Networked.

Trial licenses	A license for a 14-day, full-featured trial version of the product. The trial license must be registered before you can use the product.
Workstation licensing	A license or set of licenses is tied to a particular workstation. The product can only be used on that workstation.
Networked licensing	Networked licenses are administered and distributed by a central License Server (Embarcadero License Center or Acresso FLEXnet Publisher). There are two types of networked licenses: *Concurrent* and *Networked Named User*. With Concurrent licensing, users on different machines take turns using licenses from a shared pool. With Networked Named User licensing, licenses are pre-assigned to specific users setup on the license server's user list. Those users are guaranteed to have licenses available any time.
	NOTE: Concurrent licenses can be borrowed for use without a network connection. For details, see [Online/Offline Mode and Concurrent License Checkout]().

For a detailed description of licensing options, see [http://www.embarcadero.com/software-licensing-solutions]().

RAPID SQL LICENSE MODULES, DBMS SUPPORT, AND FEATURE AVAILABILITY

Individual license modules correspond to the DBMS platforms you are licensed to use. In addition, each DBMS license module corresponds to a product edition. Feature availability for each license module edition is as follows:

- **Standard** - Provides baseline support including datasource management, object management, SQL editing and execution, and standard tools and utilities.

- **Professional** - Standard edition features plus SQL Debugger, SQL Profiler, and Code Analyst.

RAPID SQL XE LICENSE MODULES, DBMS SUPPORT, AND FEATURE AVAILABILITY

An XE license provides access to all supported DBMS platforms. XE modules and feature availability are as follows:

- **XE Pro** - Provides baseline support including datasource management, object management, SQL editing and execution, and standard tools and utilities, SQL Debugger, SQL Profiler, and Code Analyst.

DIRECTING QUERIES REGARDING LICENSES

Questions regarding license availability, feature availability, and client or server licensing, should be directed as follows:

- If you work in an organization that uses networked licensing, direct any questions to your site's Rapid SQL administrator

- If you are using workstation licensing, direct licensing questions to your Embarcadero Technologies representative

LICENSING YOUR APPLICATION

See the following topics for details on registering your product:

- [Registering a trial or workstation license during installation](#)

- [Registering a workstation license after application startup](#)

- [Registering by phone](#)

- [Registering a networked license](#)

REGISTERING A TRIAL OR WORKSTATION LICENSE DURING INSTALLATION

Shortly after initiating download of a trial version of an Embarcadero product, you should receive an email with a serial number you must register during installation. Similarly, if you purchase an Embarcadero product while no trial version is active, you will receive a serial number that must be registered during installation.

1   Start the installation. An **Embarcadero License Registration** dialog appears.

2   Copy the serial number from the email and paste it in the **Serial Number** field.

3   Enter your Embarcadero Developer Network account credentials in the **Login or Email** and **Password** fields. If you have not previously created an EDN account, or have forgotten your password, click **I need to create ...** or **I've lost my password**.

4   Click **Register**.

Your activation file should be downloaded and installed automatically. If this does not happen, click the **Trouble Connecting? Try Web Registration** link and follow the prompts. If you still have problems, see [Registering by phone](#).

REGISTERING A WORKSTATION LICENSE AFTER APPLICATION STARTUP

The following instructions assume that you have received a workstation license by email and that you currently have a valid trial license. If you did not install a trial version or the trial period has expired, follow the instructions in [Registering a Trial or Workstation License During Installation](#) instead.

1   On the Help menu select **About** and then on the dialog that opens, click **Manage** to open a license manager dialog.

2   On the **Serial** menu, select **Add**.

3   Copy the serial number from the email and paste it in the **Add Serial Number** dialog, then click **OK**.

4 Right-click on the serial number you added, and then select **Register** from the context menu. A registration dialog opens.

   NOTE: The **Registration Code** box shows a machine-specific identifier required with other registration methods.

5 Ensure that the **Register using Online Registration** radio box is selected.

6 Provide Developer Network credentials in the **DN Login name or Email** and **DN Password** boxes. If you have not previously created an EDN account, or have forgotten your password, click **I need to create ...** or **I've lost my password**.

7 Click **Register**.

8 If prompted to restart the application, click **Yes**.

Your activation file should be downloaded and installed automatically. If this does not happen, click the **Trouble Connecting? Try Web Registration** link and follow the prompts. If you still have problems, see Registering by phone.

REGISTERING BY PHONE

If you have problems with either of the above procedures, you can register licenses by phone. You will have to provide Developer Network credentials; the registration code displayed in the Embarcadero License Registration dialog that appears when you start an unlicensed application; the product base license serial number; and the license serial numbers for any additional features you have purchased.

For North America, Latin America and Asia Pacific, call (415) 834-3131 option 2 and then follow the prompts. The hours are Monday to Friday, 6:00 A.M. to 6:00 P.M. Pacific time.

For Europe, Africa and the Middle East, call +44 (0)1628-684 494. The hours are Monday to Friday, 9 A.M. to 5:30 P.M. U.K. time.

Shortly after phoning in, you will receive an email containing an activation file. Then do the following:

1 Save the file to the desktop or a scratch directory such as c:\temp.

2 On the Help menu select **About** and then on the dialog that opens, click **Register**. A registration dialog opens.

3 Select the **I have received an activation file (*.slip or reg*.txt)** radio box.

4 Click the **Browse** button and use the **Select License Activation File** dialog to locate and select the activation file you installed.

5 Click the **Import** button to import the activation file and when complete, click the **Finish** button.

6 If prompted to restart the application, click **Yes**.

REGISTERING A NETWORKED LICENSE

If you work in an organization using Networked licensing, an administrator, department head, or someone providing a similar function will provide you with an activation file.

Once you receive the file, save it to the `license` subfolder of your product's main installation folder (typically `C:\Program Files\Embarcadero\<product><version>\license\`), then restart the application.

No additional steps are necessary.

SELECTING A LICENSE CATEGORY DURING STARTUP

During Rapid SQL startup, if multiple concurrent license categories are available, you are prompted to select a category to use for this Rapid SQL session. Multiple license categories can be set up to provide differing feature access or access to different DBMS versions. Feature and DBMS version access are typically distributed across multiple license categories to optimize use of a site's purchased licenses.

> **NOTE:** This dialog also includes the option to remember your selection on subsequent startups. If you select that option, you can subsequently use the **Select Licenses** button on the **About**... dialog (**Help > About**) to select a different license.

Contact your License Administrator for details on individual license categories or requests for additional feature or DBMS support.

ONLINE/OFFLINE MODE AND CONCURRENT LICENSE CHECKOUT

Concurrent licenses can be used in both online and offline modes. In online mode, you must have a continuous network connection to your License Center. Licenses are checked out when you start Rapid SQL and checked back in when you shut Rapid SQL down.

You can also use a license in offline mode. When you explicitly check out a license for offline use, you can use Rapid SQL without a connection to your License Center for a specified duration. This lets you work while travelling or commuting, work away from your primary work area, or use Rapid SQL when a network connection is unavailable or not required.

> **NOTE:** Contact your site administrator for information on offline license availability, the maximum duration, offline license policy at your site, or any other issues arising from online license usage.

To check out a license for offline use:

1   On the **Help** menu, select **Checkout License**. The **Check Out Licenses For Offline Use** dialog opens.

2   Select the check box associated with each individual license you want to check out.

3   In the **Checkout Duration** box, type the number of hours that you can use the offline license without a network connection to the License Center.

4   Click **OK**.

You can work offline for the specified duration. The duration period begins immediately.

If you subsequently establish a network connection to the License Center before the license duration expires, you can indicate to the License Center that the offline license is no longer required.

To indicate that an offline license is no longer required:

1   On the **Help** menu, select **Checkin License**.

There is no interruption in Rapid SQL usage. The license is not actually checked in until you shut down Rapid SQL.

More Information on Using this Document

The following topics provide additional information that is useful when working with product documentation:

- Accessing third party documentation
- Shorthand for third party product references

Accessing Third Party Documentation

Many Rapid SQL features provide support for functionality available in the supported third party DBMS platforms. In object management for example, properties available when creating or editing objects and actions available against object types, are direct equivalents of clauses, options, or keywords available for use with the third party equivalent.

In such cases, no attempt is made to duplicate detailed documentation by third parties. Casual descriptions are provided, noting the third party equivalent. You can consult the third party documentation for details or clarifications. The following links provide access to online documentation for DBMS platforms supported by Rapid SQL:

- IBM DB2 for Linux, Unix, and Windows documentation
- IBM DB2 for z/OS documentation
- InterBase/Firebird documentation
- Microsoft SQL Server documentation
- MySQL documentation
- Oracle documentation

- [Sybase documentation](#)

SHORTHAND FOR THIRD PARTY PRODUCT REFERENCES

To save space in headings and table columns, the following shorthand is used to represent versions and variations of the DBMS platforms supported by Rapid SQL.

- **DB2 LUW - IBM DB2 for Linux, Unix, and Windows**

- **DB2 z/OS - IBM DB2 for z/OS**

- **ITB/FBD - InterBase/Firebird**

- **SQL SVR - Microsoft SQL Server**

- **MySQL - MySQL**

- **Orcl - Oracle**

- **Syb - Sybase**

GETTING STARTED

The following sets of topics are intended for new users. They provide learning curve material and tell you how to configure Rapid SQL:

Application Basics	Introduces the major user interface elements.
Rapid SQL Tutorial	Provides detailed tutorial exercises that introduce the basic areas of Rapid SQL functionality.
Specifying Rapid SQL application and feature options	Tells you how to configure Rapid SQL.

APPLICATION BASICS

Application Basics is designed to situate you within the application and to provide information about what Rapid SQL offers in the way of design, navigation, and application features. The information presented here is high-level and conceptual.

Application Basics is divided into two sections, the table below describes each section:

Section	Description
Product Design	This section describes the Rapid SQL user interface.
Specifying Rapid SQL application and feature options	This section describes how to customize Rapid SQL's configuration to suit your specific needs.

PRODUCT DESIGN

The Rapid SQL window opens with the Database Explorer on the left, the Workspace on the right, and all toolbars docked at the top of the application. The Output Window is not automatically displayed. Rapid SQL also offers you a number of desktops, or workspaces, that you can toggle among while you work.

DATABASE EXPLORER

Rapid SQL organizes the wealth of information pertaining to your servers through its Database Explorer. The Database Explorer provides a fast and efficient way to access your database objects and scripts. The Database Explorer is a separate window containing a tree object that you can select and expand. The tree object organizes and nests subjects as branches. By expanding or collapsing the tree, you can efficiently browse multiple datasources. The Database Explorer window is dockable so that you can maneuver through the application efficiently.

The Database Explorer includes three tabs:

- Explorer
- Favorites
- Project
- VC Files

EXPLORER TAB

The Explorer Tab provides a visual method for browsing, accessing and manipulating your database objects. The Explorer Tab lets you:

- Connect, disconnect, and browse the objects in the supported datasources on your network.

- Drag objects to the Rapid SQL workspace.

- Create new objects.

> **NOTE:** You can assign datasources to defined categories such as production or development. This lets you color-code and label Datasource Explorer user interface elements associated with the datasource. For details, see <u>Categorizing datasources using color-coding and labelling</u>.

Datasource Node

When you click the datasource node of the Explorer Tab tree, Rapid SQL lists all the databases available for that datasource. When you click a database node, you can view all the database object types available for that datasource. When expand an object type node, Rapid SQL displays available objects. You can collapse any portion of a datasource to concentrate on a particular portion of your database.

For more information, see:

<u>Organizing the Explorer</u>

<u>Creating New Objects from the Explorer</u>

<u>Extracting DDL from the Explorer</u>

<u>Displaying Dependencies from the Explorer Tab</u>

ORGANIZING THE EXPLORER

Rapid SQL contains functionality that lets you configure how objects are organized in the <u>Explorer Tab</u>. The table below explains the available options:

Functionality	Description
Organize by Object Type	Select to display objects by object type for all users in the same list. This display mode cuts performance in databases that contain many objects.
Organize by Owner	Select to display objects by owner. Most efficient if you are working with databases containing a high number of objects.
Show Only My Objects	Select to display the objects you own in the Explorer Tab. Available if you are organizing the Explorer Tab by object type.
Show System Objects	Select to display system objects.
Full Refresh	Select to refresh.
Expand All Groups	Select to expand all groups.
Collapse	Select to collapse all datasources or collapse all groups.
Retain Group View Settings	Select to retain the current state of the Explorer Tab so that the it opens the same way the next time you start Rapid SQL.

> **TIP:** You can also set these options on the Explorer tab of the Options Editor. For details, see <u>Explorer Options</u>.

Organizing the Explorer by Object Owner

1   On the *Explorer* Tab, click the *Explorer* list above the list of datasource groups.

2   Click *Organize by Owner.*

Rapid SQL dynamically reorganizes the display of the Explorer Tab, sorting database objects by object owner.

Organizing the Explorer by Object Type

To organize the Explorer by Object Type, do the following:

1   On the *Explorer* Tab, click the *Explorer* list above the list of datasource groups.

2   Click *Organize by Object Type.*

Rapid SQL dynamically reorganizes the display of the Explorer Tab, sorting database objects by type.

Showing or Hiding System Objects

To show or hide system objects, do the following:

1   On the *Explorer* Tab, click the *Explorer* list above the list of datasource groups.

2   Click *Show System Objects* to show all system objects.

3   Click the *Show System Objects* again to hide all system objects.

CREATING NEW OBJECTS FROM THE EXPLORER

Rapid SQL lets you create new database objects from the [Explorer Tab](#). When you create an object from the Explorer Tab, Rapid SQL does the following:

• Automatically brings up the Object Attachment Facility which associates the new object with an object type.

• Opens a DDL Editor containing a shell script based on the object type you select.

• Automatically ties that object to the current database.

Creating a New Object from the Explorer

To create a new object from the Explorer, do the following:

1   In the *Explorer* Tab, right-click the object type node for the object to create and then click *New* to open the wizard associated with that object.

The Object Type Information dialog box displays. The object type and owner are automatically assigned based on the object type you selected in the previous step.

2   Enter the name of the object.

3   Click *OK*.

Rapid SQL opens an editor containing a template for the object type.

EXTRACTING DDL FROM THE EXPLORER

Rapid SQL's [Explorer Tab](#) provides a quick, convenient method for extracting object DDL. Rapid SQL lets you extract the DDL for multiple or individual database objects. SQL scripts are the text of statements that are used to create database objects. Rapid SQL also lets you run SQL scripts against other databases to recreate the database objects. You can also extract scripts to create a record of how an object was created. Rapid SQL lets you extract the CREATE statements for any object into a DDL Editor. If you are extracting the DDL of an Oracle procedural logic object, an SQL Editor automatically opens instead, saving you the trouble of inserting PL/SQL tags.

Rapid SQL offers two ways to extract DDL from the Explorer Tab:

Using the Shortcut Menu

1   Select one or more objects in the *Explorer* Tab. To select contiguous objects, hold SHIFT and select multiple objects. To select noncontiguous objects, hold ALT and select specific objects.

2   Right-click to display the shortcut menu and then click Extract.

Dragging Objects

1   Select one or more objects in the *Explorer* Tab. To select contiguous objects, hold SHIFT and select multiple objects. To select non-contiguous objects, hold ALT and select specific objects.

2   Drag the script(s) to the desired position in the Rapid SQL workspace.

DISPLAYING DEPENDENCIES FROM THE EXPLORER TAB

Rapid SQL lets you open a result set window, directly from the [Explorer Tab](#), to display dependencies for an object.

Displaying Dependencies from the Explorer Tab

1   On the *Explorer* Tab, double-click the object node.

Rapid SQL displays the list of objects.

2   Right-click the target object and then click Dependencies.

Rapid SQL opens the Dependencies window which lists all dependent objects.

NOTE: Rapid SQL does not retrieve dependencies if objects are created out of order.

FILTERING IN THE EXPLORER

For ease of navigation, Rapid SQL lets you filter the Explorer tree to display only selected nodes and objects. Rapid SQL offers the following filtering options:

- Real-time modification of the tree to show only nodes with names matching a typed character string. For details, see Simple, on-the-fly, character-based tree filtering.

- Permanent object filters, enabled and disabled for individual data sources, that restrict display based on object name and owning schema (and optionally, object type). For details, see Object Name/schema Filtering.

- The ability to restrict object types displayed for each DBMS platform. For details, see Node, or Object Type, Filtering.

Two default filters applying to owned and system objects are available. For details, see Using the Default Filters. Default and custom filters are available in the Explorer tree **Filters** node for each data source. For details, see Filters Node.

SIMPLE, ON-THE-FLY, CHARACTER-BASED TREE FILTERING

The Filter box at the top of the Explorer provides a quick, ad hoc way to restrict the nodes displayed.

To restrict the Explorer tree display to only user-defined nodes whose name contains a particular character string

- Type one or more characters in the Filter box.

The display is updated to show only nodes whose name contains the typed character string and their parent nodes.

NOTE: The Clear button in the Filter box (X) deletes the contents of the Filter box and restores the unfiltered tree display.

OBJECT NAME/SCHEMA FILTERING

Object filters let you restrict Explorer tree display for a data source by name, owning schema, or name/schema combinations. These filters are defined at the DBMS platform level and enabled or disabled on a data source by data source level. Multiple object filters can be created for a DBMS platform, letting you enable combinations of filters for a single data source, ad hoc enabling and disabling of individual filters depending on your requirements, and so on.

Each filter consists of one or more ANDed conditions based on name or schema. For example you could create a two-condition filter that restricts the Explorer tree for a data source to display only objects belonging to the Schema **QA** and whose name starts with the string **Test**.
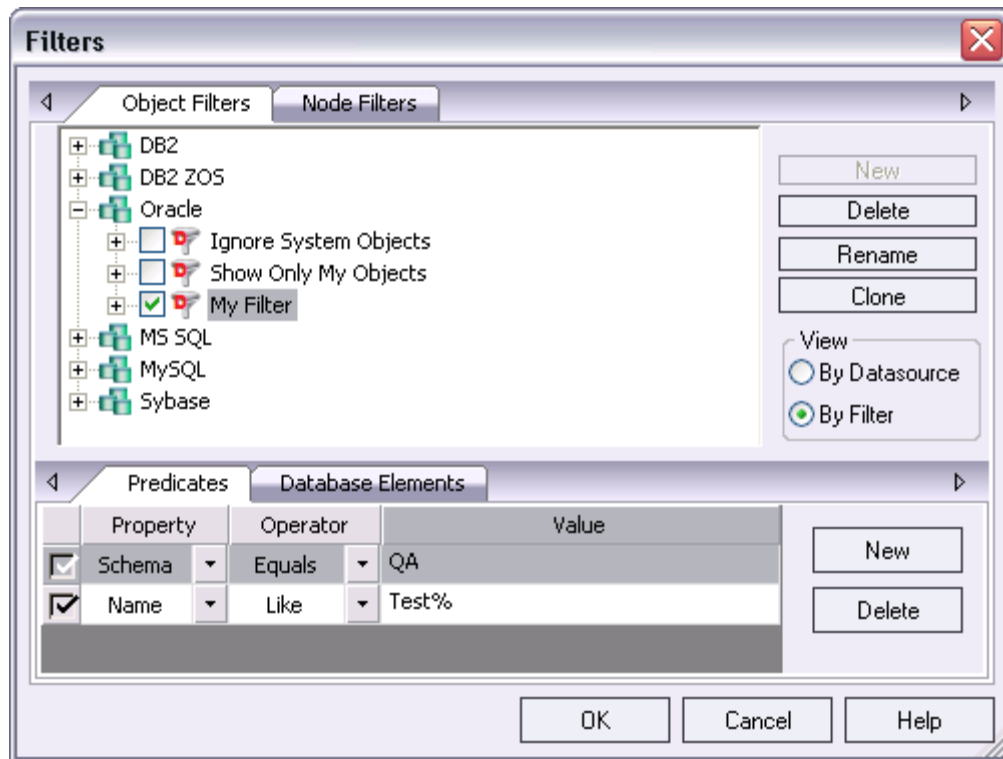
Each filter condition is a simple expression that tests object names for equality (**Equals**, **Not Equals**), inclusion in a specified list (**In, Not In**), or using pattern matching (**Like, Not Like**). Optionally, an object filter can also be used to restrict display of specified object types on a data source.

To create an object filter that can be enabled for all data sources for a DBMS platform

1   Right-click in the left pane of the Datasource Explorer tree and select **Filter** from the context menu. The **Filters** dialog opens.

2   In the **View** area, select **By Filter**.

   NOTE: The current instructions create a filter at the DBMS level. To create a filter for a single data source, select the **By Datasource** view, select a specific data source, and then proceed to .

3   Expand the icon corresponding to the DBMS for which you want to create a filter and ensure that the icon is selected.

4   Click **New** and type a name for the new filter.

5   On the **Predicates** tab, create one or more name-based or schema-based conditions, all of which must be satisfied when this filter is enabled. Objects that do not satisfy the conditions will be filtered from view. Use the following guidelines in constructing conditions:

- The **Like** and **Not Like** operators take pattern-matching **Value**s. Use any pattern matching characters and conventions supported by the DBMS for which you are defining the filter.

- The **In** and **Not In** operators take a comma-delimited list such as **CO,CA,CU**, as a **Value**.

- Click the associated **New** button to add new predicates.

6   Optionally, on the **Database Elements** tab, select the check boxes for all object types to which this filter is to be applied. The filter does not affect the display of any deselected object types.

7   Click **OK**.

By default, when you create an object filter for a DBMS platform, it is enabled for each registered data source of that DBMS type.

To disable or enable an object filter for a data source

1   Right-click in the left pane of the Explorer tree and select **Filter** from the context menu. The **Filters** dialog opens.

2   In the **View** area, select **By Datasource**.

3   Expand the icon corresponding to the target DBMS and then expand the icon corresponding to the specific data source for which you want to for enable or disable a filter.

4   Enable an object filter by selecting the associated check box or disable the object filter by deselecting the check box.

5   Click **OK**.

   NOTE: You can also enable and disable filters using controls found under the **Filters** node for each data source in the Explorer tree.

You can also use the **Filters** dialog to select and edit an object filter, rename or create a clone of an existing object filter, and delete filters.

NODE, OR OBJECT TYPE, FILTERING

Node filtering lets you hide specified Datasource Explorer tree nodes for a DBMS platform. For example, if you do not typically work with credential-related objects such as users or logins, you may wish to filter those out of the Datasource explorer tree.

To change the hidden/displayed status of object types for a DBMS platform

1   Right-click in empty space in the left pane of the Explorer tree and select **Filter** from the context menu. The **Filters** dialog opens.

2   Click the **Node Filters** tab.

3 Expand the icon corresponding to the DBMS for which you want to create a node filter, and continue to expand nodes until all nodes for which you want to change the hidden/displayed status, are visible.

4 Ensure that all nodes corresponding to objects types that are to be displayed, are selected. Similarly, ensure that all nodes corresponding to objects types that are to be hidden, are deselected.

5 Click **OK**.

Hidden/displayed status is retained for the DBMS until you explicitly open the Filters dialog and change the current settings.

Using the Default Filters

The **Filters** node for each registered data source includes two default Explorer tree filters, **Ignore System Objects** and **Show Only My Objects**. These filters can be enabled and disabled for each data source but cannot be edited or deleted.

Filters Node

The Filters node contains default Explorer tree filters and any custom filters you crete for that data source or create across that DBMS type. Controls let you enable or disable the default and custom filters. Options for creating and maintaining cusom filters are provided. For details, see [Filtering in the Explorer](#).

Favorites Tab

Rapid SQL provides the Favorites Tab for designating and accessing favorite scripts. On the Favorite Scripts Tab you can do the following with frequently used SQL scripts:

- View

- Navigate

- Save

- Recall

- Execute

> **TIP:** Sample Favorite Scripts are installed for Microsoft SQL Server, Oracle, and Sybase Adaptive Server.

For more information, see [Favorite Properties Dialog Box](#).

FAVORITE PROPERTIES DIALOG BOX

The table below describes the options and functionality on the Favorite Properties dialog box:

Option	Description
Description	Lets you enter a description.
File Name	Lets you type or browse and locate a file.
Hot Key	Lets you enter a Hot Key. Hot Keys must begin with CTRL or be stand-alone function keys. Rapid SQL automatically places the CTRL command in front of any character.
File Type	Lets you select a file type.

SUBSYSTEM NODE

> **NOTE:** The Subsystem node is available for IBM DB2 LUW for OS/390 and z/OS only.

The Subsystem node displays detailed information about the DB2 subsystem. Subsystem objects include:

- Connections

- DDF

- Parameters

CONNECTIONS

Connections displays the current connections to the subsystem.

> **NOTE:** The information is read-only.

For more information, see Subsystem Node.

DDF

DDF (distributed data facility) displays the information regarding the status and configuration of the DDF, a set of DB2 LUW for OS/390 components through which DB2 LUW for OS/390 communicates with another RDBMS.

> **NOTE:** The information is read-only.

For more information, see Subsystem Node.

PARAMETERS

Parameters displays the DB2 subsystem parameters.

> **NOTE:** The information is read-only.

For more information, see Subsystem Node.

PROJECT TAB

The Project Tab provides a visual method for browsing, accessing, and manipulating your projects. The Project Tab lets you:

- Open, close, modify, and browse projects.

- Drag project items to the Rapid SQL workspace.

- Create new projects.

> **NOTE:** The Project Tab is available after you create or open a project.

For more information on working with projects, see [Project Management](#).

VC FILES TAB

The VC Files Tab displays open version control files. This tab displays files listed in the *.xml file for the version control files. The tab displays a checkbox icon for files that are checked-out.

For more information, see [Version Control](#).

RAPID SQL WINDOWS

The Rapid SQL interface includes several windows to help you develop your program. The windows include:

Topics
- [Describe Window](#)
- [Output Window](#)
- [Browsers](#)
- [Workspaces](#)
- [Preview Dialog Boxes](#)

DESCRIBE WINDOW

Rapid SQL offers a floating Describe window for procedures, tables, views, and functions (Oracle and IBM DB2 LUW for Open Systems only). In the Describe window, you can view columnar information (for tables and views) or input parameter information (for procedures and functions).

Opening the Describe Window

Rapid SQL offers three ways to open the Describe window:

1   In an editor, right-click an object and then click Describe from Cursor.

2   On the *Explorer* Tab, select an object and then click Describe.

3   On the *Explorer* Tab or in an editor, select an object and then press CTRL+D.

Using the Describe Window

In the Describe window:

1   Click the Name list and then click a name to view a list of types of objects in the database.

2   Click the Owner list and then click an owner to view a list of all owners of objects in the database.

3   Click the Type list and then click a type to view columnar information (for tables and views) or input parameter information (for functions and procedures).

OUTPUT WINDOW

Rapid SQL incorporates a separate window to capture all messages returned by the server and to log entries about the progress of operations started by the application.

For more information, see Configuring the Output Window

CONFIGURING THE OUTPUT WINDOW

Rapid SQL lets you display, hide, or dock the Output Window anywhere in the application.

Displaying the Output Window

1   On the *View* menu, click *Output*.

    OR

    On the *Main* toolbar, click Output.

    Rapid SQL displays the Output Window.

Hiding the Output Window

1   On the *View* menu, click *Output*.

    OR

    On the *Main* toolbar, click Output.

    OR

    Right-click the *Output Window* and then click *Hide.*

    Rapid SQL hides the Output Window.

Docking the Output Window

Right-click the *Output Window* and then click *Docking View*.

    Rapid SQL docks the Output Window to the bottom of the application frame.

Undocking the Output Window

Right-click the *Output Window* and then click *Docking View.*

Rapid SQL displays the Output Window as a floating window in the application.

MESSAGES IN THE OUTPUT WINDOW

The [Output Window](#) lets you save, print, copy, and clear server messages.

Saving Server Messages

1   Right-click the *Output Window* and then click *Save.*

Rapid SQL opens the Save As dialog box.

2   Enter the location and name of the file in the *File Name* box.

NOTE: Rapid SQL defaults the file extension to .msg.

3   To save the file, click *OK.*

Printing Server Messages

1   Right-click the *Output Window* and then click *Print.*

Rapid SQL opens the Print dialog box.

NOTE: Rapid SQL prompts you with information on the size of the print job before opening the Print dialog box.

2   Configure your print job.

3   Click *OK* to print the file.

Copying Server Messages

1   Right-click the target *Server Messages* and then click *Copy.*

Rapid SQL copies the selected text to the Microsoft Windows Clipboard.

2   Paste the contents of the clipboard into target applications.

Clearing Server Messages

1   Right-click the *Output Window* and then click *Clear.*

Rapid SQL clears your Server Messages.

BROWSERS

Browsers are a flexible environment where you can examine, extract, and execute database objects and their dependencies. Browsers provide the means to view objects types across multiple database platforms and connections. You can simultaneously view and work with objects from InterBase/Firebird, Oracle, Microsoft SQL Server, Sybase Adaptive Server, and IBM DB2 LUW for Open Systems.

The benefit of using Browsers is the ability to see detailed information about specific object types. You can also print, search, copy, and sort the contents of a Browser window.

Topics

- Browser Toolbar

- Opening Browsers

- Browser Object Types

- Extracting DDL from Browsers

- Displaying Dependencies from Browsers

- Refreshing Browsers

BROWSER TOOLBAR

You can place the floating *Browser* toolbar anywhere on the Rapid SQL workspace.

For more information, see Browsers.

OPENING BROWSERS

Browsers let you view all types of database objects, including the SQL procedures used to build them. Browsers let you:

- Copy database objects

- Modify database objects

- Test database objects

The ability to browse dependencies is especially useful, particularly when modifying SQL code in procedures and triggers. For example, in a situation where a trigger enforces a rule that does not let you update a particular box, you can use the Browser to do the following:

- Browse the triggers to find the offending trigger.

- Extract the DDL for that trigger into one window.

- In another window, drop the offending trigger, make your update to the box, then execute the corrected trigger DDL to replace the trigger in the database.

Opening a Browser Window

Rapid SQL offers two ways to open a Browser:

1 On the *Browse* menu, click the target object type.

   OR

   In the workspace, right-click, click *Browser*, then click the target object.

   Rapid SQL opens a Browser.

For more information, see [Browsers](#).

BROWSER OBJECT TYPES

Rapid SQL's Browsers read the appropriate object types for specific databases. A select statement is issued against the appropriate systems table based on the requested object to bring back a listing of the objects in the database. The table below contains a list of the Browsers available for each database platform:

Object Type	DB2 LUW	DB2 z/OS	InterBase/Firebird	SQL Server	Oracle	Sybase
Aliases	✔	✔		✔		✔
Blob Filters			✔			
Check Constraints	✔	✔	✔	✔	✔	✔
Clusters					✔	
Databases	✔	✔		✔	✔	✔
Database Links					✔	
Defaults				✔		✔
Directories					✔	
Domains			✔			
Encrypted Keys			✔			
Event Monitors	✔					

Object Type	DB2 LUW	DB2 z/OS	InterBase/Firebird	SQL Server	Oracle	Sybase
Exceptions			✔			
Extended Procedures				✔		✔
External Functions			✔			
Foreign Keys	✔	✔	✔	✔	✔	✔
Functions	✔	✔			✔	
Generators			✔			
Groups				✔		✔
Indexes	✔	✔	✔	✔	✔	✔
Libraries					✔	
Packages	✔	✔			✔	
Plans		✔				
PL/SQL Code Profiling					✔	
Primary Keys	✔	✔	✔	✔	✔	✔
Procedures	✔	✔	✔	✔	✔	✔
Profiles					✔	
Roles			✔		✔	
Rules				✔		✔
Rollback Segments					✔	

Object Type	DB2 LUW	DB2 z/OS	InterBase/Firebird	SQL Server	Oracle	Sybase
Segments				✔		
Sequences					✔	
Shadows			✔			
Snapshots					✔	
Snapshot Logs					✔	
Materialized Views					✔	
Materialized View Logs					✔	
Synonyms					✔	
Tables	✔	✔	✔	✔	✔	✔
Tablespaces					✔	
Triggers	✔	✔	✔	✔	✔	✔
Types					✔	
Type Bodies					✔	
Unique Keys	✔	✔	✔	✔	✔	✔
User Datatypes	✔	✔		✔		✔
User Messages				✔		
Users				✔	✔	✔
Views	✔	✔	✔	✔	✔	✔

WORKING WITH BROWSERS

Browsers offer a versatile method of browsing and managing the contents of your databases. To help you maintain and organize your databases, you can:

- Print the contents of a Browser

- Search the contents of a Browser

- Copy the contents of a Browser

- Sort the contents of a Browser

Printing Browsers

1   Open a Browser for the desired object type.

2   On the *File* menu, click *Print to open the Print dialog box.*

3   In the *Name* box, click the list, which contains a list of local and network printers that you can access (if you do not see any listed, then your computer is not configured for any printers).

4   Click the target printer.

5   In the *Print Range* box, click the appropriate option button to indicate print range.

6   In the *Number of copies* text box of the *Copies* box, click the *Up* or *Down* arrow or enter the number of copies.

7   Click *OK*.

    Rapid SQL prints the selection.

Searching Browsers

1   Open a Browser for the desired object type.

2   On the *Edit* menu, click *Find*.

    Rapid SQL opens the Find box.

3   In the *Find What* text box, enter the search string.

4   To make the search case sensitive, select the *Match Case* check box.

5   To specify the direction to search, in the Direction box, click the *Up* or *Down* option button.

6   Click *Find* Next.

    Rapid SQL finds the next occurrence of your search string.

Copying Browsers

1   Open a Browser for the desired object type.

2   Select the objects to copy.

3   On the *Edit* menu, click *Copy.*

4   Place the pointer at the position where you want to paste the objects, and then on the *Edit* menu, click *Paste.*

Sorting Browsers

1   Open or [Create](#) a *Browser* for the desired object type.

2   Double-click the column header for the column of data to sort and Rapid SQL lists the contents of the column in ascending order.

3   Double-click the column header again and Rapid SQL lists the contents of the column in descending order.

For more information, see [Browsers](#).

EXTRACTING DDL FROM BROWSERS

For each database type, Rapid SQL provides an appropriate Browser. The Browsers are mutually exclusive object windows, showing only objects of a given type. If you connect to multiple datasources, you have access to a number of objects that are not available based on the database platform. Rapid SQL includes intelligence to determine the valid object types in the underlying datasource.

Using the Main Menu

1   On the *Browse* menu, click the target object type.

    Rapid SQL opens the appropriate Browser:

2   In the Browser, double-click the target object type to extract the object type DDL into a *DDL Editor.*

Using the Browser Toolbar

1   On the *Browser* toolbar, click Tables.

    Rapid SQL opens the Table Browser:

2   Click the scroll bar arrow to locate the target table.

3   Double-click the target table.

    Rapid SQL extracts the schema DDL into a DDL Editor.

Using the Shortcut Menu

1   Right-click an open area of the workspace, click Browsers, and then click the target object type.

2   In the Browser, double-click the target object type.

    Rapid SQL extracts the schema DDL into a DDL Editor.

For more information, see [Browsers](#).

DISPLAYING DEPENDENCIES FROM BROWSERS

You can display object dependencies for an object from its corresponding object Browser. Rapid SQL displays the dependencies in a separate result set window.

Displaying Dependencies

Rapid SQL offers three ways to display dependencies from Browsers:

1   Open a Browser for an object type.

2   In the Browser, click the target object.

3   On the *Object* menu, click *Dependencies.*

    OR

    On the *Browser* toolbar, click Dependencies.

    OR

    Right-click the target object and then click *Dependencies.*

    Rapid SQL displays dependencies in a separate window.

For more information, see [Browsers](#).

REFRESHING BROWSERS

Rapid SQL lets you refresh and display the results of a Browser operation.

Refreshing the Browser

1   On the *Object* menu, click *Refresh.*

    OR

    On the *Browser* toolbar, click Refresh.

    OR

    Right-click the Browser workspace and then click *Refresh.*

    Rapid SQL refreshes the results of the browser operation.

For more information, see [Browsers](#).

WORKSPACES

Workspaces are a convenient way to maximize your desktop. You can use workspaces to multiply the amount of scripting, script execution, and development resources you have available at any one time. Rapid SQL lets you open and use several workspaces at one time. Using more than one workspace lets you:

- Execute long running scripts in one workspace while working in other workspaces.

- Develop strategies for working on scripts and result sets in one workspace while other scripts reside in one or more of the other workspaces.

Toggling Between Workspaces

Rapid SQL offers two ways to toggle between workspaces:

1   On the *Main* toolbar, click Workspace.

    OR

    Right-click the current workspace and then click the target workspace.

    Rapid SQL brings the target workspace forward.

For more information, see:

Managing Workspaces

MANAGING WORKSPACES

Rapid SQL provides you with three default workspaces. You manage the workspaces in the Workspace dialog box. Using the Workspace dialog box you can:

- Differentiate between workspaces by changing the background color or wallpaper.

- Toggle among workspaces.

- Create, delete, rename, and specify the order of workspaces.

Managing Workspaces

The Workspace dialog box lets you manage all open windows in your workspace.

1   On the *Windows* menu, click Windows.

    Rapid SQL displays the Workspace dialog box. Any open windows in the current workspace display in the list.

The table below describes the options and functionality on the Workspace dialog box:

Option	Description
Activate	Sets the focus onto the window you have selected in the list and closes the Workspace dialog box.
OK	Closes the Workspace dialog box and accepts any changes you have made to the windows in the current workspace.

Option	Description
Save	Saves the contents of the window you have selected in the list. You are prompted to provide a name and location for the file you are saving if you have not done so already.
Close Window	Closes the window you have selected from the list. If you have not saved the contents of the window, you are prompted with a save file alert.
Help	Initiates and displays this Help topic in the Rapid SQL Help.

SET SORT COLUMNS DIALOG BOX

The Set Sort Columns dialog box lets you sort multiple columns, and sort column identification, in the Right Pane of the application.

For more information, see [Completing the Set Columns Dialog Box.](#)

COMPLETING THE SET COLUMNS DIALOG BOX

To complete the Set Columns dialog box, do the following:

1   In the right pane of the application, right-click a column heading and select Sort Multiple Columns.

    Rapid SQL opens the Set Sort Columns dialog box.

2   In Column Name select the column(s) to use to sort the information in the right pane of the application.

3   Click the right arrow to move the column(s) to the Column Name box.

4   Click the up and down arrows to change the order of the sort columns.

For more information, see [Set Sort Column Dialog Box](#).

PREVIEW DIALOG BOXES

Before executing any code, Rapid SQL offers Preview dialog boxes to let you confirm actions before execution. In the Preview dialog boxes, you can:

- Preview the code to execute.

- View the SQL of the code on your database.

- Create a report detailing the affect of executing code on your database.

- Schedule execution of the code.

- Save the code to execute.

- Open your e-mail program with the code to execute as an attachment.

- Print the code to execute.

MENUS

Rapid SQL offers two context-sensitive menus to let you access all the application's features. The Main Menu is always on the top of the application window. The shortcut menu is accessible from almost anywhere in the application. Right-click to view the available shortcut menu. Rapid SQL lets you customize the Tools menu to help you tailor the application to your needs.

Topics

- Main Menu

- Shortcut menus

- Customizing General User Interface Appearance

MAIN MENU

Rapid SQL's features can all be accessed from the Main Menu by clicking the menu name and selecting from the submenu. The menus are context sensitive and change based on the tasks you want to perform. The table below describes the Rapid SQL menus:

Menu Item	Description
File	Create, open, close, print, send, and save script files and result sets. Set application options and defaults.
Datasource	Create, modify, select, connect to, and disconnect from datasources. Access the database search facility.
Project	Available only when a project is open. Configure project management, build projects, and use version control functions.
Browse	Browse any object type a datasource connection.
Logfile	Activate/deactivate, open, set options, and flush the Rapid SQL application log.
View	Arrange the Rapid SQL environment. Display or hide the Database Explorer, toolbars, Output Window, Describe window, activate full-screen mode.
Tools	Choose any of Rapid SQL's tools, such as Database Search and the Visual Diff Utility. Customize and add tools of your own.
Bookmarks	Access and manage bookmarks.
Help	Access HTML Help.
Query	Available only when an Editor is open. Execute and set options for your SQL scripts.
Object	Available only when a browser is open. Execute, view dependencies, extract, and refresh objects in a database.
Edit	Available only when an Editor is open. Edit and manipulate the text in your scripts.
Format	Available only when a Result Window is active. Format the contents of result sets.
Window	Cascade and tile open windows. Toggle among open windows.

For more information, see [Menus](#).

EXPLORER BOOKMARKS
The Bookmarks menu lets you access and manage explorer bookmarks. Explorer bookmarks let you quickly access nodes in the Database Explorer.

Creating Explorer Bookmarks

1 On the Database Explorer, right-click the target node, and then select Add Bookmark.

   Rapid SQL opens the Add Friendly Bookmark Name dialog box.

2 Type the explorer bookmark name.

3 Click OK.

   Rapid SQL displays the explorer bookmark under the Bookmarks menu. Explorer bookmarks are organized by platform.

Editing Explorer Bookmarks

1 On the Main Menu, select Bookmarks.

2 Select Bookmark Manager.

   Rapid SQL opens Bookmark Manager.

3 To rename the explorer bookmark, select the target explorer bookmark, and then click Rename.

   Rapid SQL opens the Edit Bookmark Name dialog box.

4 Type the new explorer bookmark name.

5 Click OK.

6 To delete an explorer bookmark, select the target explorer bookmark, and then click Delete.

   TIP: To add explorer bookmarks without using the Add Friendly Bookmark Name dialog box, select Do not show 'Add Friendly Bookmark Name' dialog option.

SHORTCUT MENUS
Rapid SQL incorporates context-sensitive menus to give you another way to access object functionality. These menus mirror the functionality that you can access from application toolbars or the main menu.

Opening Shortcut Menus

1 Right-click anywhere on the Rapid SQL desktop to open the appropriate shortcut menu.

For more information, see [Menus](#).

TOOLBARS

Rapid SQL toolbars change to reflect the element of the application you are using. The toolbars contain icons that are the fastest way to access commonly used features of Rapid SQL. You can move the toolbars to horizontal or vertical positions anywhere on the screen, and you can toggle them off and on by using the shortcut menu when the pointer is positioned over one of Rapid SQL's toolbars. For more information, see the following topics:

- Available Toolbars

- Using Toolbar Viewing Options

- Moving Toolbars

    NOTE: For related information, see Customizing General User Interface Appearance.

AVAILABLE TOOLBARS

The following list represents Rapid SQL's toolbars:

Datasource Toolbar

Registration Toolbar
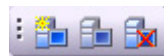
Main Toolbar

Windows Toolbar

Tools Toolbar

Project Toolbar

SQL Edit Toolbar
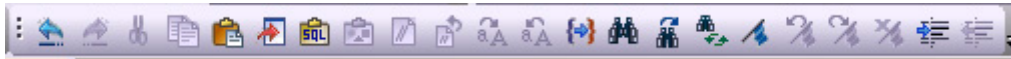
Browsers Toolbar

USING TOOLBAR VIEWING OPTIONS

Rapid SQL offers standard Windows toolbar options such as docking, floating, and positioning toolbars.The only application-specific viewing option is the hiding or display of the individual toolbars.

To hide or display a toolbar

1   On the **View** menu, select **Toolbars** and then select the specific toolbar you want to display or hide.

For information on the toolbars available, see <u>Available Toolbars</u>.

MOVING TOOLBARS

1   Click the pointer at the right edge of any toolbar.

2   Drag the toolbar to the new position.

For more information, see <u>Toolbars</u>.

CUSTOMIZING GENERAL USER INTERFACE APPEARANCE

Rapid SQL lets you choose from a set of general visual application styles, dictate hiding or display of particular items, and select preferences for specific user interface elements.

To customize the general look and feel of Rapid SQL

1   On the **View** menu, select **Toolbars** and then select **Customize**. The **Customize** dialog opens.

2   Use the following table as a guide to understanding and setting options on tabs of the Customize dialog:

Tab	Settings and tasks	
Toolbars	Select the toolbars you want to display in the application. For information on the toolbars available, see <u>Available Toolbars</u>.	
Application Visual Style	Select a visual style such as Microsoft Windows XP or one of the .NET options from the dropdown. Depending on your selection, the following options may or may not be enabled: **Use default WIndows XP colors**, **OneNote style tabs**, **Docking Tab Colors**, **Allow MDI Tab Swapping**, **Enable Smart Docking**, **Enable Tab Menu**, and **3D Rounded Docking Tabs**.	
	Menu animations	Lets you specify a menu animation style of UNFOLD, SLIDE, or FADE.
	Menu Shadows	Displays shadowed 3D effects.
Tools	Lets you define external applications to run from the **Tools** menu of Rapid SQL: the text displayed on the Tools menu command (**Menu contents**), the path and file name of the executable (**Command**), optional **Arguments**, and an optional **Initial DIrectory**. For detailed information on providing arguements, see <u>Specifying an Argument for a Tools Menu Command</u>.	
Keyboard	**Category**	Select a general category for a hot key for the command.
	Commands	Select a hot key command, based on the general category.
	Description	Displays the command description.
	Set Accelerator for	Select application area where you want new hot key to be active.
	Current Keys	Displays current hot key.
	Press New Shortcut Key	Press keyboard key or an unassigned F key.
Options	**Show ScreenTips on toolbars**	Select to display a ScreenTip when you hover your mouse over a button. For example, when you hover your mouse over the New button, Rapid SQL displays the ScreenTip "New."
	Show shortcut keys in ScreenTips	Select to display a shortcut key in the ScreenTip when you hover your mouse over a button. For example, when you hover your mouse over the New button, Rapid SQL displays the ScreenTip "New (CTRL+N)."

3   Click **Apply** at any time to implement any changes you have made and when finished, click **Close**.

SPECIFYING AN ARGUMENT FOR A TOOLS MENU COMMAND

You can specify an argument to be passed to a program for newly added commands by choosing one of Rapid SQL's <u>predefined arguments</u> or entering a command-line argument.

The table below provides scenarios of how to use command-line arguments:

Command	Argument	Description
NOTEPAD.EXE	$$FilePath$$	Starts Microsoft Notepad displaying the contents of the $$FilePath$$ argument.
ISQL.EXE	-U$$CurUserID$$ -P$$CurPwd$$ -S$$CurConString$$ -i$$FilePath$$	Starts ISQL, automatically connects to the current datasource using the current user name and password, and executes the contents of $$FilePath$$.
SQLPLUS.EXE	$$CurUserID$$/ $$CurPwd$$@$$CurConString$$ @$$FilePath$$	Starts SQL*Plus, connects to the current datasource using the current user name and password, and executes the contents of $$FilePath$$.

The table below provides scenarios of how to use Rapid SQL's predefined arguments:

Argument	Description
$$FilePath$$	The complete filename of the current source (defined as drive+path+filename); blank if a non-source window is active.
$$FileDir$$	The directory of the current source (defined as drive+path); blank if a non-source window is active.
$$FileName$$	The filename of the current source (defined as filename); blank if the non-source window is active.
$$FileExt$$	The filename extension of the current source; blank if a non-source window is active.
$$CurLine$$	The current cursor line position within the active window.
$$CurCol$$	The current cursor column position within the active window.
$$CurText$$	The current text (the word under the current cursor position, or the currently selected text, if there is one).
$$CurDir$$	The current working directory (defined as drive+path).
$$CurDatasource$$	The name of the current datasource as defined in Rapid SQL.
$$CurUserID$$	The name of the current datasource user.
$$CurPwd$$	The current datasource password.
$$CurConString$$	The current connection string or server name.

PREDEFINED ARGUMENTS

Rapid SQL provides a number of predefined arguments that you can pass to programs that you have added to the Tools menu. The table below lists the available predefined arguments:

Argument	Description
$$FilePath$$	The complete filename of the current source (defined as drive+path+filename); blank if a non-source window is active.
$$FileDir$$	The directory of the current source (defined as drive+path); blank if a non-source window is active.
$$FileName$$	The filename of the current source (defined as filename); blank if the non-source window is active.
$$FileExt$$	The filename extension of the current source; blank if a non-source window is active.
$$CurLine$$	The current cursor line position within the active window.
$$CurCol$$	The current cursor column position within the active window.
$$CurText$$	The current text (the word under the current cursor position, or the currently selected text, if there is one).
$$CurDir$$	The current working directory (defined as drive+path).
$$CurDatasource$$	The name of the current datasource as defined in Rapid SQL.
$$CurUserID$$	The name of the current datasource user.
$$CurPwd$$	The current datasource password.
$$CurConString$$	The current connection string or server name.

> **NOTE:** Arguments are case-sensitive.

KEYBOARD SHORTCUTS

Rapid SQL provides a number of keyboard shortcuts to help you expedite your tasks. The table below lists the taxes and related shortcuts:

General Editing	Keyboard Command
Delete one character to the left	BACKSPACE
Delete one character to the right	DELETE
Cut selected text to the Clipboard	CTRL+X
Undo the last action	CTRL+Z
Redo the last undo operation	CTRL+Y
Copy text	CTRL+C
Paste the Clipboard contents	CTRL+V

To Extend a Selection	Keyboard Command
One character to the right	SHIFT+RIGHT ARROW
One character to the left	SHIFT+LEFT ARROW
To the end of a word	CTRL+SHIFT+RIGHT ARROW
To the beginning of a word	CTRL+SHIFT+LEFT ARROW
To the end of a line	SHIFT+END
To the beginning of a line	SHIFT+HOME
One line down	SHIFT+DOWN ARROW
One screen up	SHIFT+PAGE UP
To the beginning of a document	CTRL+SHIFT+HOME
To the end of a document	CTRL+SHIFT+END
To include the entire document	CTRL+A

To Move the Insertion Point	Keyboard Command
One character to the left	LEFT ARROW
One character to the right	RIGHT ARROW
One word to the left	CTRL+LEFT ARROW
One word to the right	CTRL+RIGHT ARROW
One line up	UP ARROW
One line down	DOWN ARROW
To the end of a line	END
To the beginning of a line	HOME
One screen up (scrolling)	PAGE UP
One screen down (scrolling)	PAGE DOWN
To the end of a document	CTRL+END
To the beginning of a document	CTRL+HOME

Bookmarks	Keyboard Command
Toggle bookmark on/off	CTRL+F2
Go to next bookmark	F2
Go to previous bookmark	SHIFT+F2

Splitter Windows	Keyboard Command
Go to next pane	F6
Go to previous pane	SHIFT+F6

Debugger Operations	Keyboard Command
Start Debugging	CTRL+F5
Stop Debugging	SHIFT+F5
Step Over	F10
Step Into	F11
Run to Cursor	CTRL+F10
Step Out	SHIFT+F11
Describe from Cursor	CTRL+D
Insert or Remove Breakpoint	F9
Toggle (Enable or Disable) Breakpoint	CTRL+F9
Edit Breakpoint	ALT+F9
Go	F5
Restart	CTRL+SHIFT+F5

Debugger Windows	Keyboard Command
Open or Close Watch Window	ALT+3
Open or Close Variables Window	ALT+4
Open or Close Call Stack Window	ALT+5
Open or Close Dependency Tree Window	ALT+6

Other Windows	Keyboard Command
Go to the Result Tab	CTRL+ALT+R
Go to the Query Tab	CTRL+ALT+Q
Open the Describe window (for highlighted object)	CTRL+D
Toggle between Workspaces	CTRL+W
Toggle between Datasource Explorer and ISQL Window.	CTRL+ALT+E

FULL SCREEN MODE

Rapid SQL has full screen mode capabilities so you can conceal the application framework and use the entire monitor area. Full screen mode hides any other applications running on the computer and uses every available pixel for the application. Main menu functionality is accessible through keyboard commands when you use full screen mode.

Activating Full Screen Mode

Rapid SQL offers two ways to activate full screen mode:

1   On the *View* menu, click *Full Screen.*

    OR

    On the *Main* toolbar, click Full Screen.

    Rapid SQL expands the application to fit the entire monitor area.

    NOTE: The Full Screen mode icon is a stand-alone floating toolbar.

Dismissing Full Screen Mode

1   Click *Full Screen* to expand the application to fit the entire monitor area.

    TIP: If you closed the Full Screen mode toolbar, right-click the top of the Rapid SQL desktop to bring the toolbar back.

2   Click *Full Screen* to restore the application to the default size.

TUTORIAL EXERCISES

The exercises that follow walk you through the Rapid SQL's major functional areas. After completing these exercises, you will have the foundation you need to explore the many features and benefits of Rapid SQL. You'll have learned how to competently manage the major database administration an development tools provided.

This guide is divided into a set of sessions:

- [Session 1: Getting Started](#)
- [Session 2: Productivity Enhancers](#)
- [Session 3: Scripting](#)
- [Session 4: Working with Code Workbench](#)
- [Session 5: Building a Database Project](#)
- [Session 6: Visual Query Builder](#)
- [Session 7: Live Data Editor](#)
- [Session 8: Code Analyst](#)
- [Session 9: SQL Debugging and Profiling](#)

You can use this basic tutorial as a roadmap of product highlights, but also to help you find your own path to explore Rapid SQL.

Once you've started, you can select **Help Topics** from the **Help** menu to find many additional resources that complement and build on many of the activities shown in this tutorial.

SESSION 1: GETTING STARTED

Before anything else, you must perform the following tasks:

- [Starting Rapid SQL](#)
- [Registering Cross-Platform Datasources](#)

STARTING RAPID SQL

How you start Rapid SQL depends on the type of application you are evaluating:

- **InstantOn version** - start the application by double-clicking the file you downloaded.

- **Fully-installed version** - The **Start** menu sequence for Rapid SQL is always in the form **Programs > Embarcadero Rapid SQL** *version identifier* **> Embarcadero Rapid SQL** *version identifier*, where *version identifier* reflects the version you are running.

To get started

1 Run Rapid SQL.

The first time Rapid SQL starts, a dialog opens, prompting you to set up datasources. In addition to letting you manually set up individual datasources, a number of more automated methods are available. If you have installed and used other Embarcadero tools, Rapid SQL can find any active datasources being used by those tools. In addition, Rapid SQL provides a Discover Datasources feature that automatically searches the DBMS configuration files on your system for datasources that are not currently registered. Since other Embarcadero tools let you export datasource definitions to file, you also have the option of importing these definitions.

2 For the purpose of this tutorial, click **Cancel**. You will be registering a datasource manually.

The main Rapid SQL window opens.
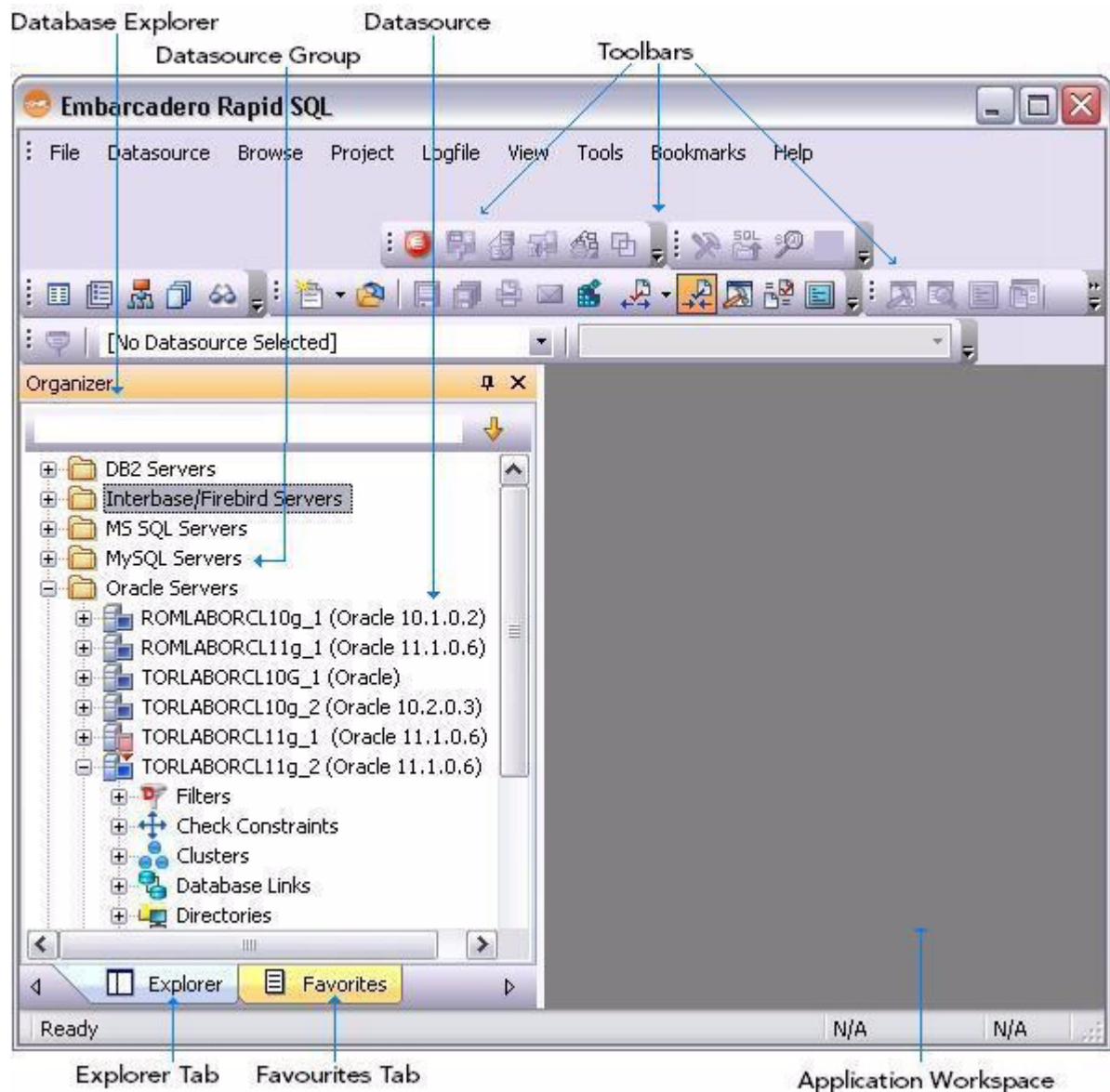
REGISTERING CROSS-PLATFORM DATASOURCES

For now, you will register a datasource manually.

1   On the **Datasource** menu, select **Register Datasource**. A **Datasource Registration** Wizard opens.
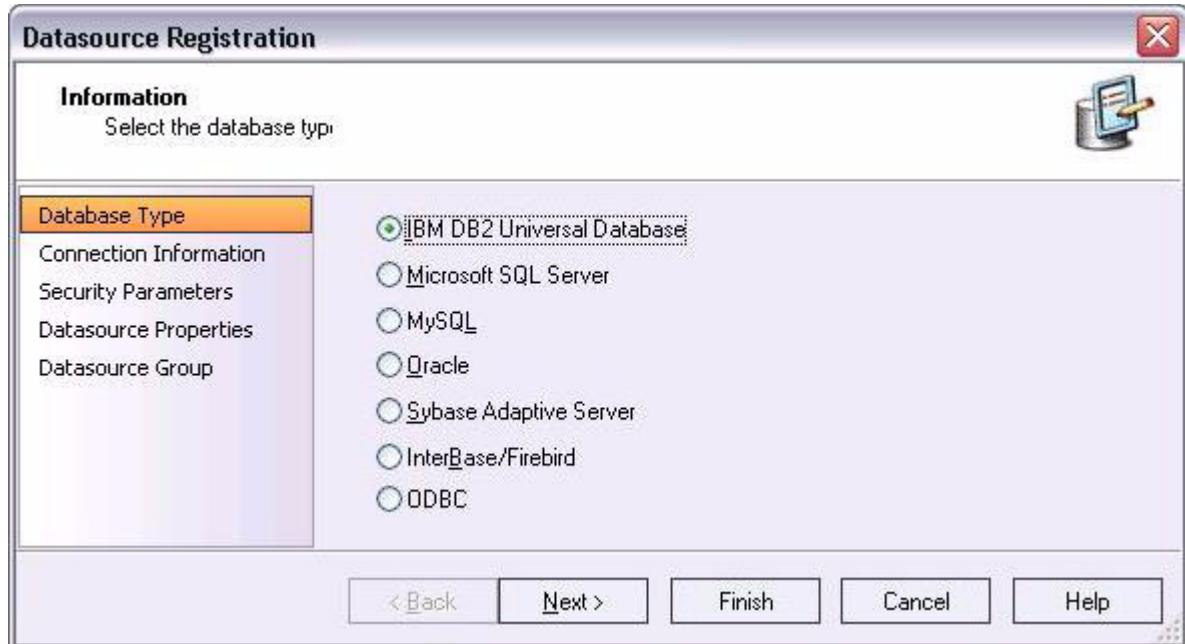
2   Choose **Microsoft SQL Server** as the DBMS type and then click **Next**.

3   Specify the **Host name** of a SQL Server datasource on your network, override the **Datasource Name** with **SAMPLE_DATASOURCE** and then click **Next**.

4   Provide valid credentials in the **User ID** and **Password** boxes, and then select the **Auto-Connect?** checkbox to eliminate having to provide credentials each time you connect to this datasource.

5   In the left-hand pane, select **Datasource Group**, select the **MS SQL Server** folder, and then click **Finish**.

> **NOTE:**   The **Datasource Group** panel also lets you assign a category to a datasource. This provides a means to visually distinguish between different server purposes, development vs. production, for example, in your enterprise. Categorization is a customizable feature.

6   Select **Yes** when prompted to connect to the new datasource.

Rapid SQL offers the same easy-to-use Datasource Registration Wizard for all supported DBMS platform connections. The connection information only needs to be set up one time for each platform and can be saved locally or in a common datasource catalog for use by other Embarcadero products.

By default, Rapid SQL stores datasource definitions in the Windows Registry. There is also a local, file-based option. Embarcadero products supporting these methods can share datasource catalogs on the same machine.

There is also a network-shared storage option.

Rapid SQL also offers the ability to import and export datasource definitions. This lets you share definitions among users and across datasource storage methods.

SESSION 2: PRODUCTIVITY ENHANCERS

This session focuses on some commonly used time-saving features:

- The Database Explorer Tree

- Creating an Object Using the Object Creation Wizard

- Working With an Existing Object Using the Object Editor

- Object Documentation and Reporting

- Working With Code, Files and Data

- Setting Environment Options

- Favorites Tab

- Working with Scripts and Files

- Viewing Data

- Retaining Datasource Explorer View Settings

- Datasource Explorer Bookmarks

- Setting Keyboard Shortcuts and Hotkeys

- Referencing Most Recently Used Datasources

THE DATABASE EXPLORER TREE

Rapid SQL makes it easy and intuitive to navigate between datasources and to drill-down into atomic database objects within the Database Explorer Tree. The Database Explorer Tree displays all registered datasources and serves as the entry-point for much of Rapid SQL's advanced functionality.

1   Click on the Explorer title bar and select **Organize By Object Type**.
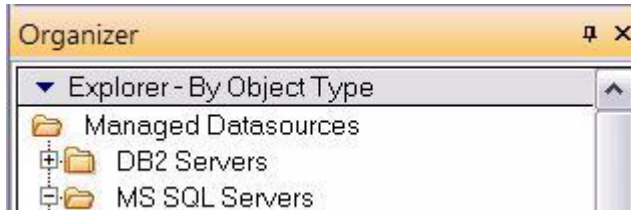
2   Select and expand the **SAMPLE_DATASOURCE > Databases > pubs** node to display the database object sub-nodes.

CREATING AN OBJECT USING THE OBJECT CREATION WIZARD

From within the Database Explorer Tree you can create any database object using simple Object Creation Wizards. The following is an example of how to use the Table Object Creation Wizard. It is similar to the Object Creation Wizards available within Rapid SQL for all database objects and other supported elements.

1   Right-click on the **Tables** node and select **Create**. A **Create Table Wizard** opens.
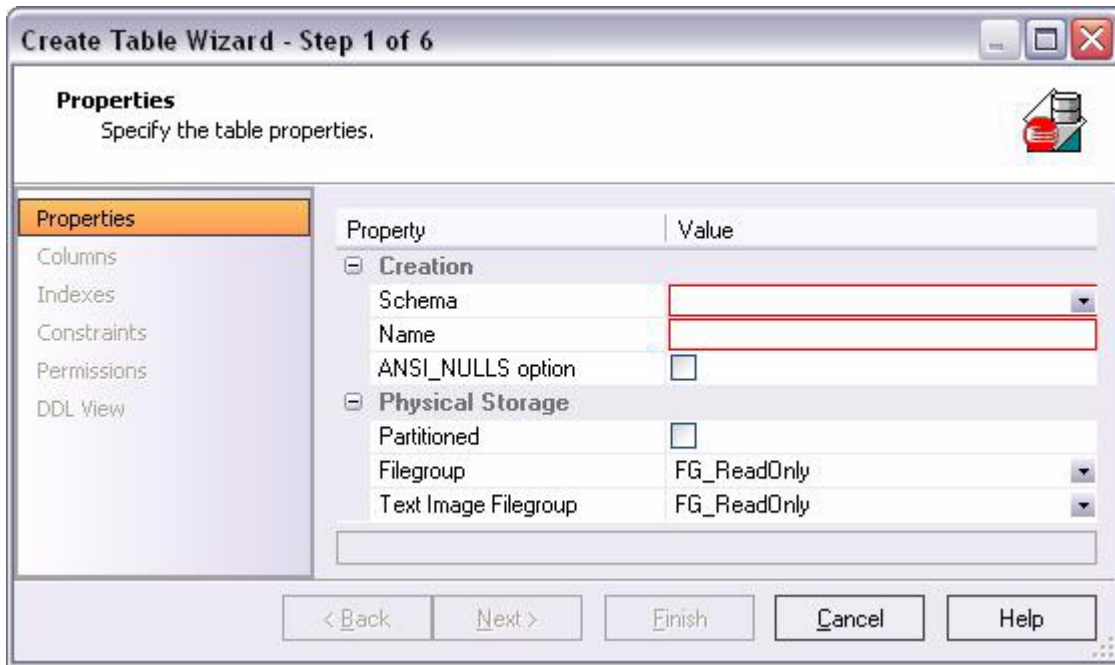
2   Select a **Schema**, provide a **Name** of **SAMPLE_TABLE**. Leave the remaining default settings and click **Next**.

3   Add a column, using a **Name** of **Sample_Column1** and select a **Type** of char. Experiment with the **Add Column** and **Delete** buttons, and with selecting a column and modifying its attributes.

4   Click **Finish**. The **DDL View** panel opens showing the DDL that will be used to create the new table.

5   Deselect the **Launch Object Editor After Execute** and then click **Execute**.

Rapid SQL builds the platform-specific SQL code, syntactically-correct and ready to run the first time. There is no SQL coding required in any of the Rapid SQL Object Creation Wizards.

WORKING WITH AN EXISTING OBJECT USING THE OBJECT EDITOR

While the wizard offered you the option to automatically open an editor on creating the table, you can also manually open an editor.

1. In the Database Explorer, ensure that the **Tables** node is expanded and then right-click on your new table and select **Open**.
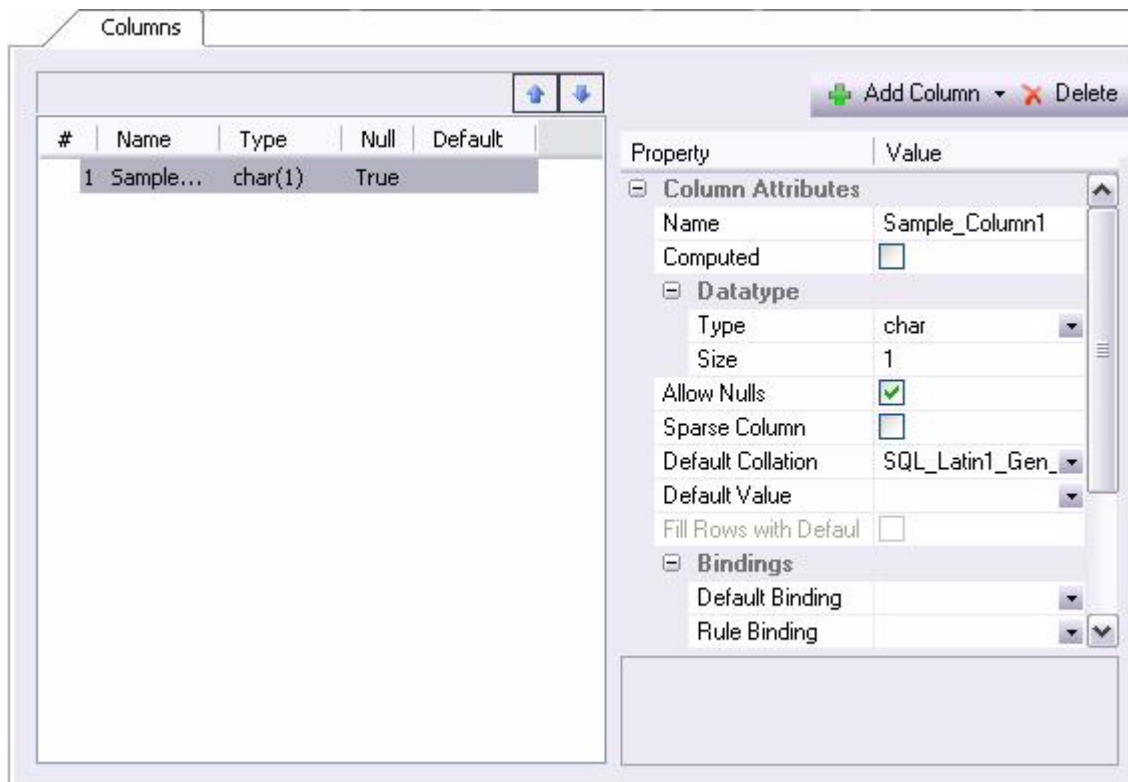
Object Editor features are as follows:

- All Object Editors provide standardized, multi-tabbed windows for each database object type.

- All Object Editors provide fully-functional toolbars for easy object management.

- Rapid SQL has full knowledge of the underlying DBMS system catalog, syntax and alteration rules, so the user can concentrate on what needs to be done, not on how to do it.

- Drop-down boxes allow you to easily move between owners and objects.

- The Rapid SQL Object Editors easily perform operations that would normally require painstaking and error-prone scripting, such as deleting or inserting columns in a table while preserving data, dependencies and permissions. Rapid SQL analyzes the database catalog to determine its structure, and then automatically generates the SQL script required for the extended alteration. For instance, when a full table alteration is required, Rapid SQL automatically unloads and reloads the data, eliminating tedious work.

2. Close the Object Editor window.

OBJECT DOCUMENTATION AND REPORTING

Rapid SQL provides rich, detailed HTML Reporting for all database objects. Building a browser-ready report for any object is only a few mouse-clicks away.

1 Expand the **Tables** node, right-click on any table and select **Report** from the menu. A **Report** dialog opens.

2 Enter a destination **Report Home Page File Name**. This can be a network web server directory\file.

3 Enter a **Report Title** and click **Execute**.

The HTML report will automatically be displayed in the Rapid SQL application workspace. For example:

dbo.authors

Object Type	Table
Datasource	EBTN-08 (SQL Server 08.00.0760)
Login	sa
Database	pubs
Report Date	4/14/2003 16:54:45.847

Columns

Name	Datatype	Null	Default	Default Binding	Rule Binding
au_id	id	No			
au_lname	varchar(40)	No			
au_fname	varchar(20)	No			
phone	char(12)	No	'UNKNOWN'		

The HTML report can be saved to a new file or referenced in the file named above.

> **NOTE:** All HTML reports are browser-ready and suitable for posting directly to the web.

WORKING WITH CODE, FILES AND DATA

Rapid SQL provides many features and powerful development tools for creating and executing SQL code and working with data.

> **NOTE:** For purposes of this Evaluation Guide, we are only covering high-level functionality of the major features and tools within Rapid SQL.

SETTING ENVIRONMENT OPTIONS

The **Options** dialog allows you to set the Rapid SQL development environment to meet your needs.

1   Select **File > Options** from the menu. The **Options** dialog opens.
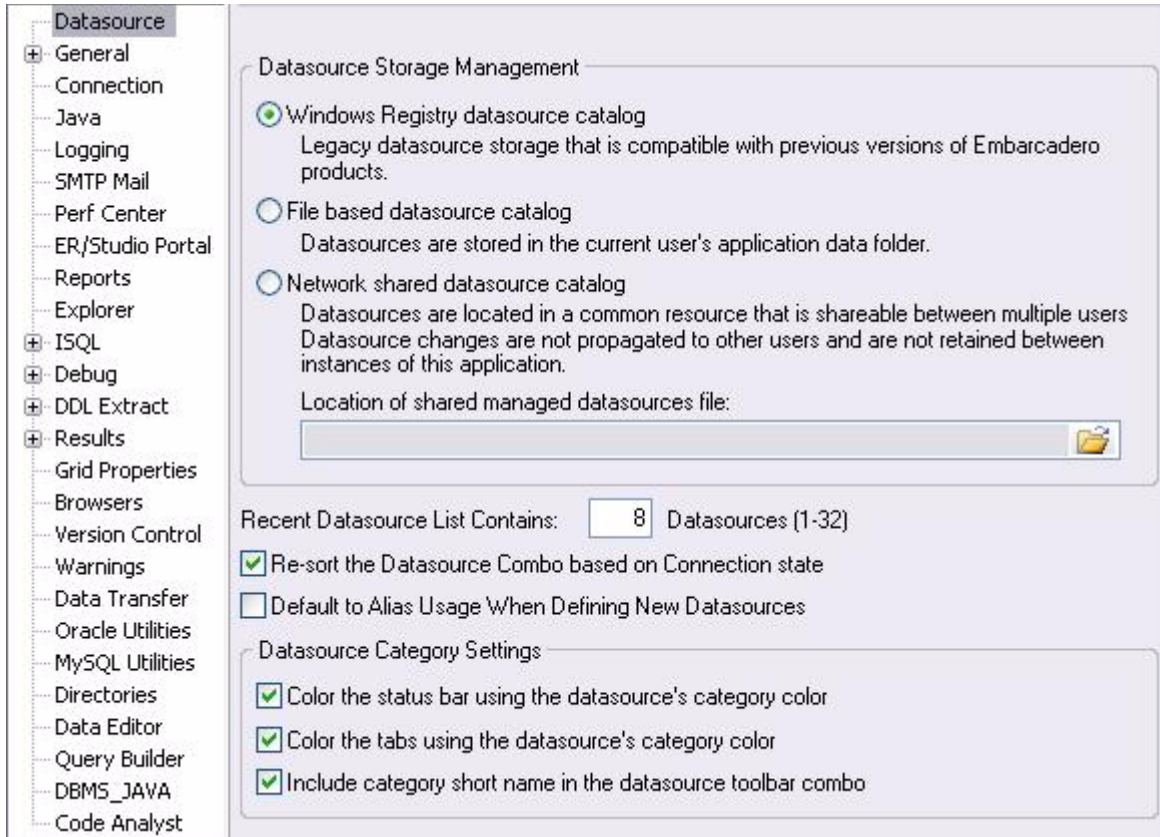
The **Options** dialog has one page per option category. You select an option category in the left-hand pane and can subsequently set options on that page. Options are applied when you click **OK**.

2   Close the **Options** dialog.

FAVORITES TAB

The **Favorites** Tab provides a drag-and-drop library interface of all supported DBMS syntax, SQL syntax, built-in functions, optimizer hints, and SQL-conditional syntax. Additionally, it provides the ability to create custom folders to store commonly-used code for quick and efficient access, as needed.

To open the Favorites Explorer:

1   At the bottom of the Explorer pane, click **Favorites**. The Favorites Explorer appears.

2   Expand the **Microsoft** node and then expand the **Schema** sub-node.

3 Right-click the **Procedures** node and select **Open**. The selected code opens in the SQL Editor window and is ready for execution.

4 Right-click in the editor window and select **Close** from the context menu.

To add a custom folder to the Favorites Explorer

1 Open the Favorites Explorer and right-click the **Favorites** folder.

2 Select **New Folder** from the menu. A new folder is added to the bottom of the tree view.

WORKING WITH SCRIPTS AND FILES

Rapid SQL extends the auto-generation of SQL code by allowing you to run your scripts across multiple databases at the same time. In addition, there is the option to execute the code immediately or schedule it to run later.

SCRIPT EXECUTION FACILITY

1 On the Favorites Explorer, navigate to the **Microsoft SQL Server** node and select the **Schema** sub-node.

2 Right-click the **Procedures** node and select **Script Execution Facility**. The **Script Execution Facility** dialog box appears.

3 Click the **Target** tab and select the datasources to run the script against.

4 Click the **Output** tab and select a type of output format. For the purposes of this example, select **Graphical Output**.

5   If you want to send a notification message indicating that the script has executed, click the **Notify** tab and enter the appropriate information in the fields provided.

6   Click **Execute** if you want Rapid SQL to run the script against the datasources specified on the **Target** tab. If not, simply close the dialog.

    NOTE: Separate script output windows are created for each selected datasource.

FILE EXECUTION FACILITY

Similar to the Script Execution Facility, files containing SQL scripts can be added to the File Execution Facility and executed immediately or scheduled to run later. Other than the origin of the code, all supporting functionality is the same.

1   On the **Tools** menu, select **File Execution Facility**. Rapid SQL opens the **File Execution Facility** dialog box.

2   To locate the file you want to execute, click **Add**. Use the **Add Files** dialog box to locate and select a file.

3   On the **Target** tab, select the datasources to run the script against.

4   On the **Output** tab, select the desired output option. For the purposes of this example, select **Graphical Output**.

    NOTE: To enable the scheduling function for the script, you must select the **File Output** option.

5   If you want to send notification that the script has executed, on the **Notify** tab, complete the target information.

6   Click **Execute** if you want Rapid SQL to run the script against the target datasources. Otherwise, close the dialog without executing.

    NOTE: Separate script output windows are created for each selected datasource.

VIEWING DATA

Rapid SQL provides several options for browsing data. In addition, it gives you the ability to construct even the most complex SQL statements with point-and-click ease.

SELECT * BROWSING

1   On the Database Explorer ensure that the **Explorer** tab is displayed and then expand the **MS SQL Server** node.

2   Expand any database you know has table data, expand the **Tables** node, right-click a table, and then click **SELECT * FROM**.

All columns and rows from the table are displayed in the active workspace.

3   Close the workspace window.

RETAINING DATASOURCE EXPLORER VIEW SETTINGS

1   Click on expandable settings at the top of the Explorer pane.
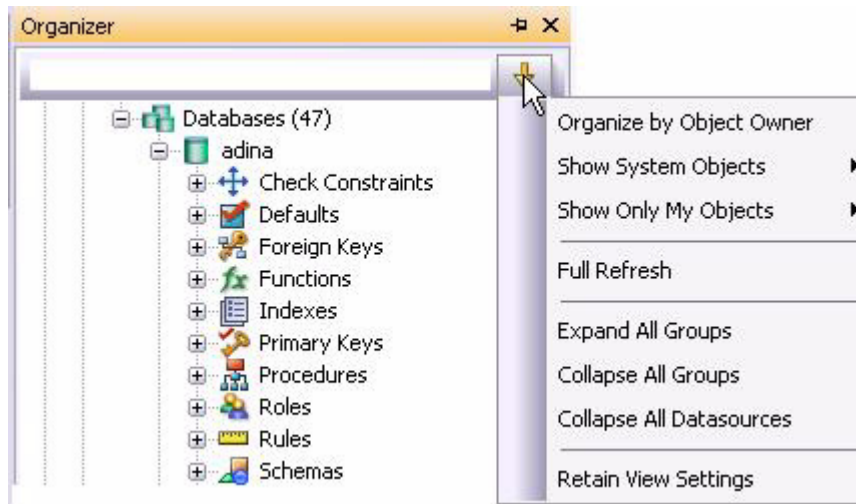
2   Select **Retain View Settings**.

The Explorer will open the next time just as you left it. All connections that were present when you closed Rapid SQL will be re-established.

DATASOURCE EXPLORER BOOKMARKS

Rapid SQL allows you to set bookmarks for frequently visited database objects.

1   Right-click on any node in the Datasource Explorer Tree.
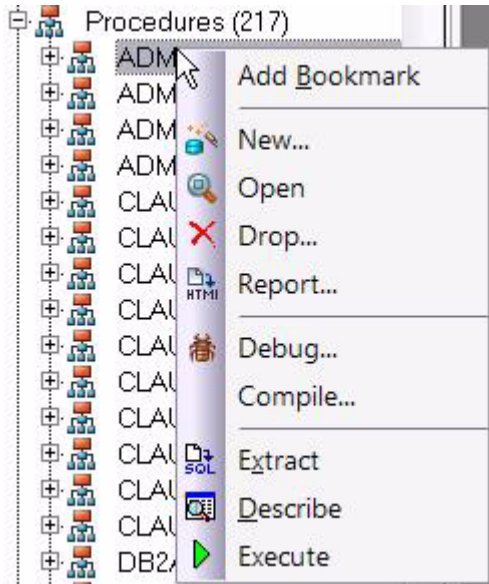
2   Select **Add Bookmark** and use the **Add Friendly Bookmark Name** dialog to optionally provide a new name, and create the bookmark.

After Bookmarks are defined you can use them to easily navigate to commonly used datasource resources using the **Bookmarks** menu.
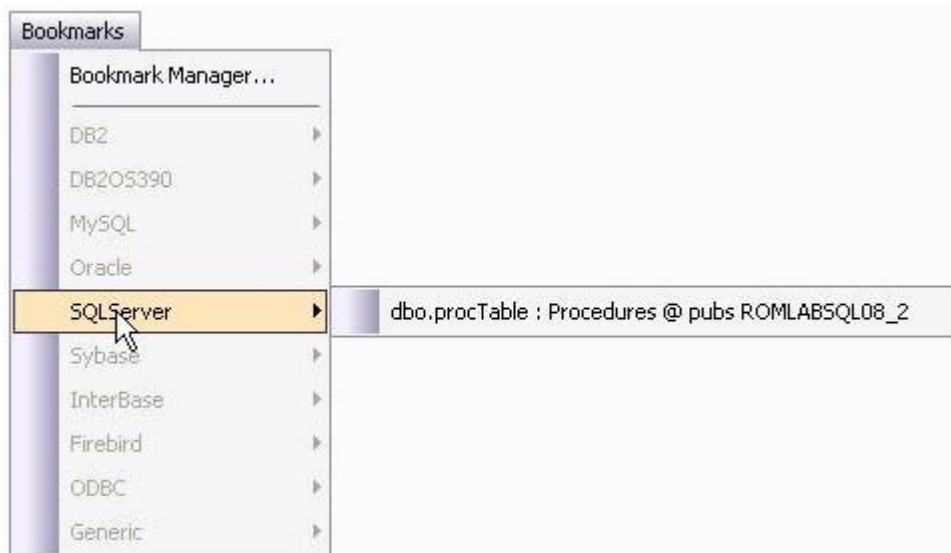
The **Bookmark Manage**r handles maintenance of Bookmarks.

1 Select **Bookmarks > Bookmark Manager**.
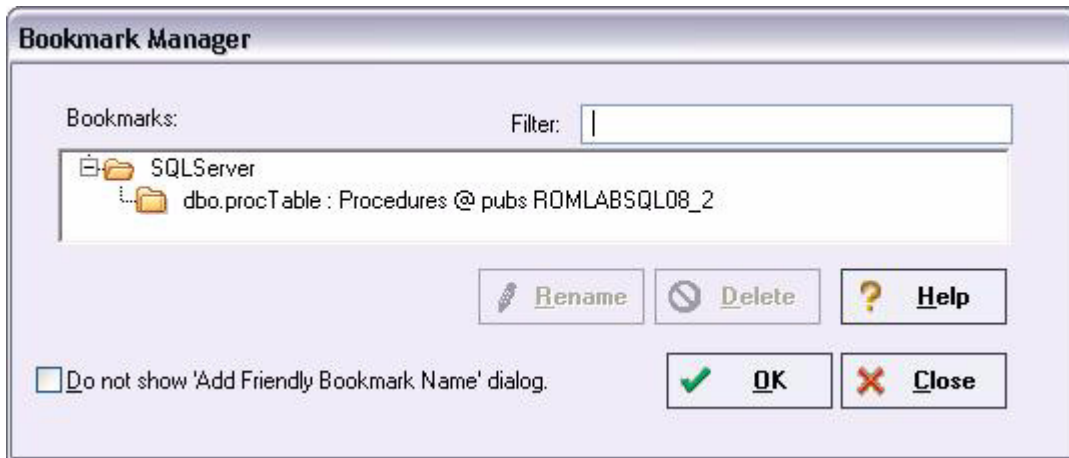
2 Close the **Bookmark Manager** dialog.

SETTING KEYBOARD SHORTCUTS AND HOTKEYS

1 On the **Tools** menu, select **Customize**.

2. In the **Customize** dialog go to the **Keyboard** tab.
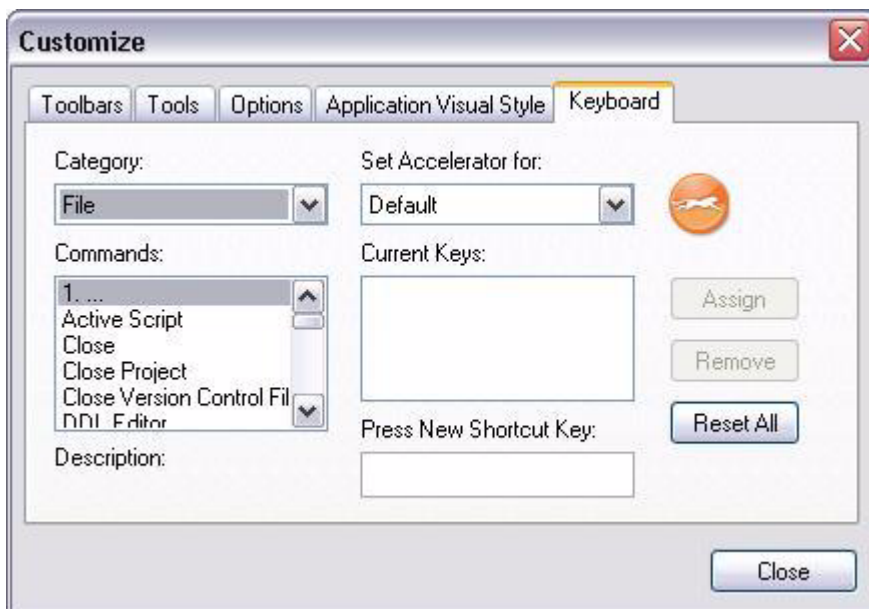
The **Keyboard** tab can be used to set keyboard shortcut hotkeys for all areas of Rapid SQL functionality.

3 Close the **Customize** dialog.

REFERENCING MOST RECENTLY USED DATASOURCES

1   Select **File > Recent Datasources** and then select a datasource.
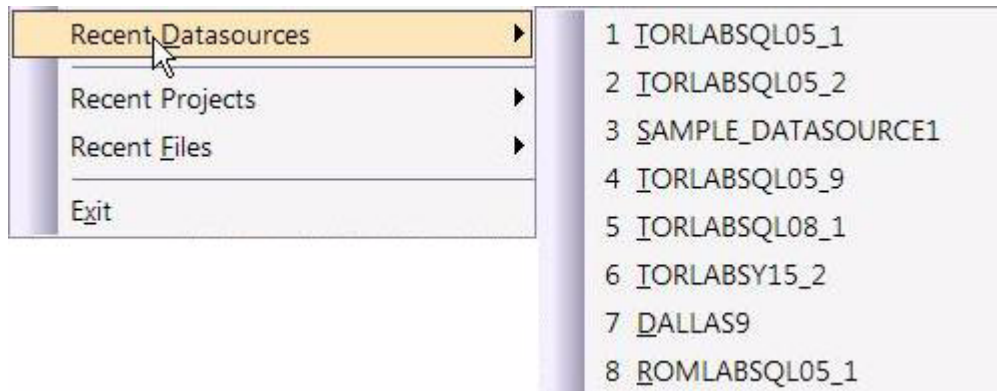
This will automatically place you on the datasource within the Explorer, ready to work with active connection.

SESSION 3: SCRIPTING

This session looks at Rapid SQL's development environment:

- Generating Code

- Right-click feature

- Automated error detection and coding assistance

- Other coding aids

GENERATING CODE

By providing several code generation and assistance options, Rapid SQL makes cross-platform development easy for developers of all experience levels.

> **NOTE:**   The following examples build on the SQL Server 2000 SAMPLE_DATASOURCE registered earlier in this Evaluation Guide. These examples can be applied to any registered datasource for any of the supported platforms.

CODE GENERATION FACILITY

The Code Generation Facility can be used to create complete procedures, functions or packages revolving around views or tables.

1   From the menu, open **Tools > Code Generation Facility**.

2   Select the **SAMPLE_DATASOURCE** datasource and the **pubs** database from the dropdown list boxes.
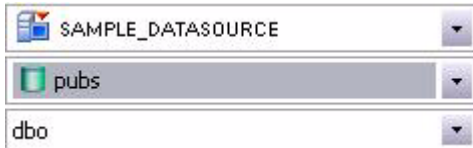
3   Select the **authors** table, **state** as the input column and all columns for output.
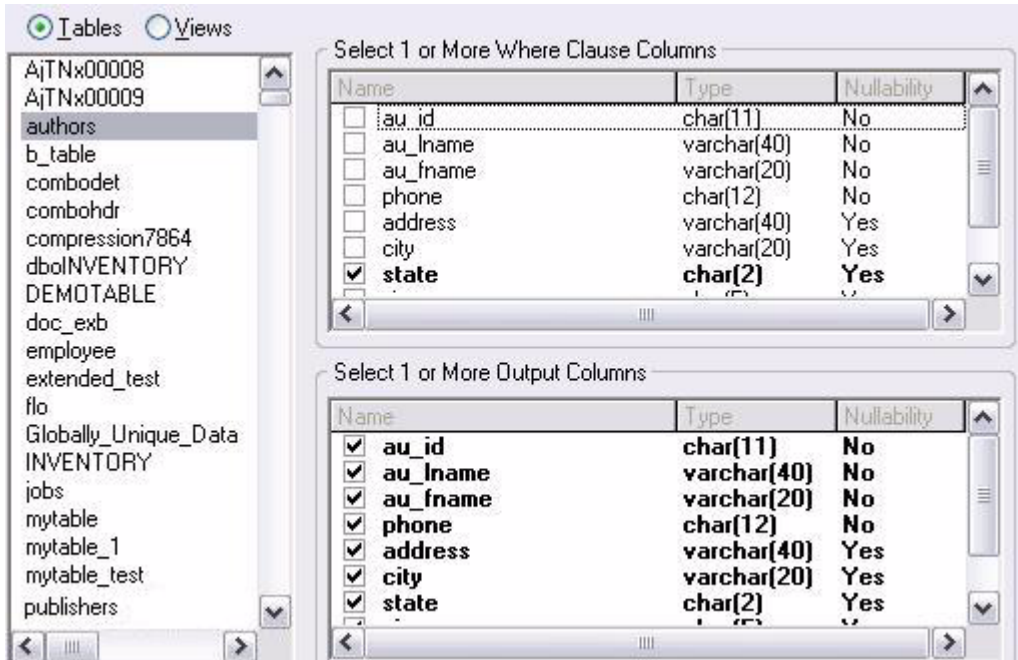
4   Select **Select** as the code option.
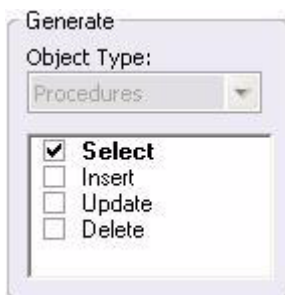
5   Select a file to save the generated script and check **Open**.
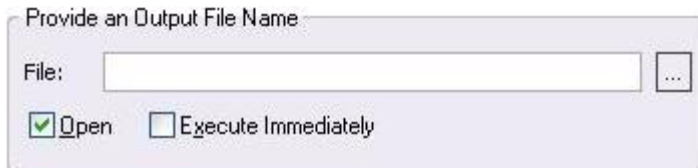
6   Click **OK** and the DDL to create the procedure will be generated and displayed in an editable window, called the DDL Editor. You can edit the name of the new procedure and any of the generated code at this time. Name the new procedure **sample_select_authors**.

7   Click on the Execute or Step Execute button to submit the DDL and create the procedure.

The indicated file will be saved on the selected directory.

> **NOTE:**   No SQL statement coding is required to generate complete stored procedures and packages. If applicable, Rapid SQL allows all generated code to be previewed and edited to fit any development need.

RIGHT-CLICK FEATURE

Similar to the Code Generation Facility, the "right-click" code generation feature can be used to create complete procedures, functions or packages revolving around views or tables.

 1   From the Database Explorer Tree, expand the **SAMPLE_DATASOURCE > pubs > Tables** sub-node.

2. Right-click on the **authors** table.
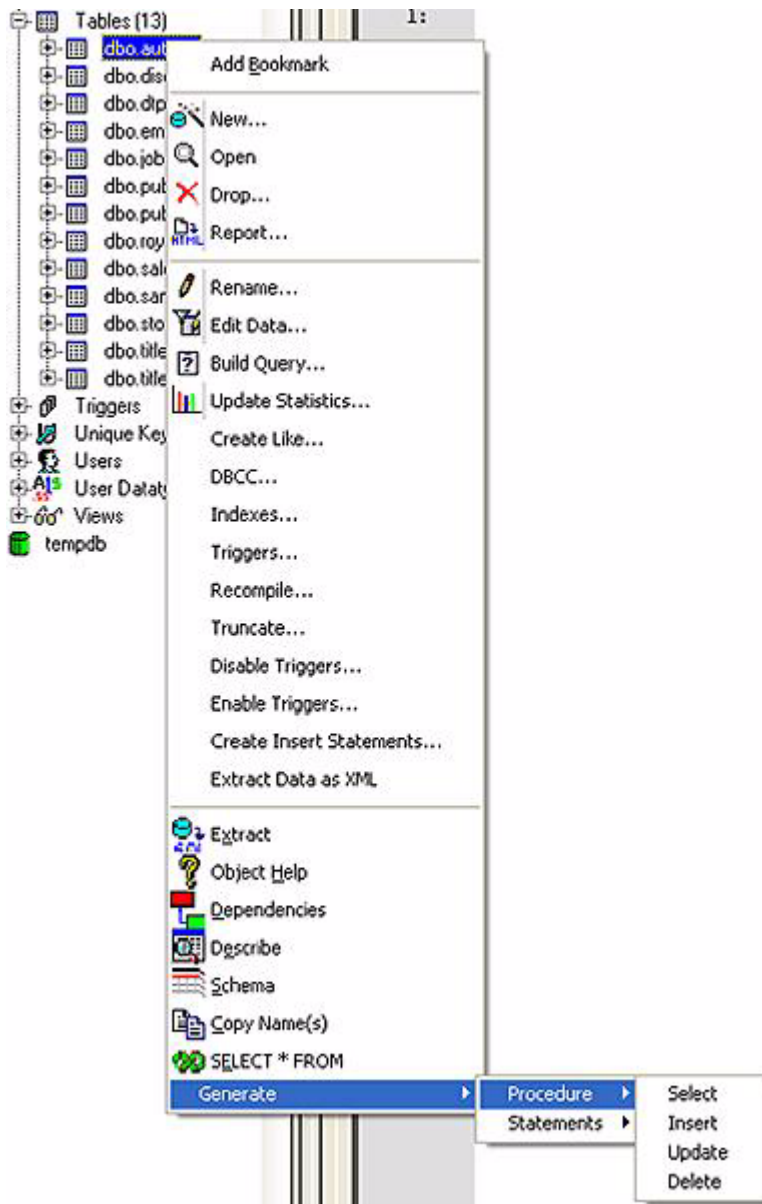
3   Select **Generate > Procedure > Select**.

4   Select **state** as the input column, and leave all output columns selected.

5   Click **OK** and the DDL to create the procedure will be generated and displayed in the DDL Editor. You can edit the name of the new procedure and any of the generated code. Name the new procedure **sample_select_authors2**.

6   Click on the Execute or Step Execute button to submit the DDL and create the procedure.

AUTOMATED ERROR DETECTION AND CODING ASSISTANCE

Rapid SQL provides a range of features that detect or help you avoid errors and save keystrokes in developing your scripts.

To enable these features:

1   On the **File** menu select **Options**. The **Options** dialog opens.

2   In the left-hand pane, expand the **ISQL** node and then select **Code Assist**.

3   On the **Code Assist** panel:

- Ensure that **Enable Code Complete** is selected.

- Ensure that **Severity levels for semantic validation problems** has Warning selected.

- Ensure that **Enable Real-time syntax checking** is selected.

4   Click **OK**.

To see these features in action:

1   On the **File** menu, click **New**, and then **ISQL**.

Rapid SQL opens the SQL Editor window. You can add SQL code via your method of choice (free-form typing, retrieve from a file, paste copied code, etc.).
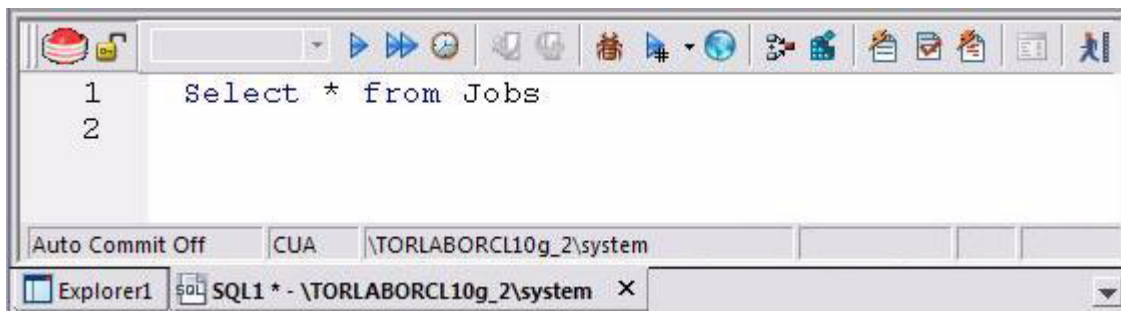
2   Type **SELECT * FROM** and stop typing.

Note the error condition.

Rapid SQL can run a syntax check any time there is an interval of 1.5 seconds between keystrokes. You can also disable automatic syntax checking and only run a check when you manually initiate it. Syntax error annotation persists until you correct the problem.

3   This time type a fragment that includes the name of a nonexistent object, **SELECT * FROM NON.OBJECT**, for example. For now, ignore any popups. The warning condition is a result of on-the-fly semantic validation.

```
1 ⚠   SELECT * FROM NON.OBJECT|
```

Rapid SQL notifies you when a script contains a reference to an object that Rapid SQL cannot resolve.

4   Type **SELECT * FROM** followed by a space and then stop typing. If no popup appears, press CTRL+SPACE. The Code Complete suggestion box lets you select from objects or object name components such as databases or schema. This feature saves keystrokes and minimizes typing errors. See the online Help for full descriptions of these features.

5   Close the current SQL Editor window.

To restore Rapid SQL settings:

1   On the **File** menu select **Options**. The **Options** dialog opens.

2   On the **Code Assist** panel, click the **Restore defaults** button.

3   Click **OK**.

OTHER CODING AIDS

Rapid SQL provides extensive, easy-to-use coding aids for all of the supported DBMS platforms, throughout the application. Aids are provided in the form of ready-to-use code templates and blocks of syntactically correct code.

PASTE SQL

1   From the Database Explorer Tree, expand the S**AMPLE_DATASOURCE > pubs sub-node**.

2   Select **File > New > SQL** to open an SQL Editor window.

3   Select **Edit > Paste SQL Statement** to open the Paste SQL window.
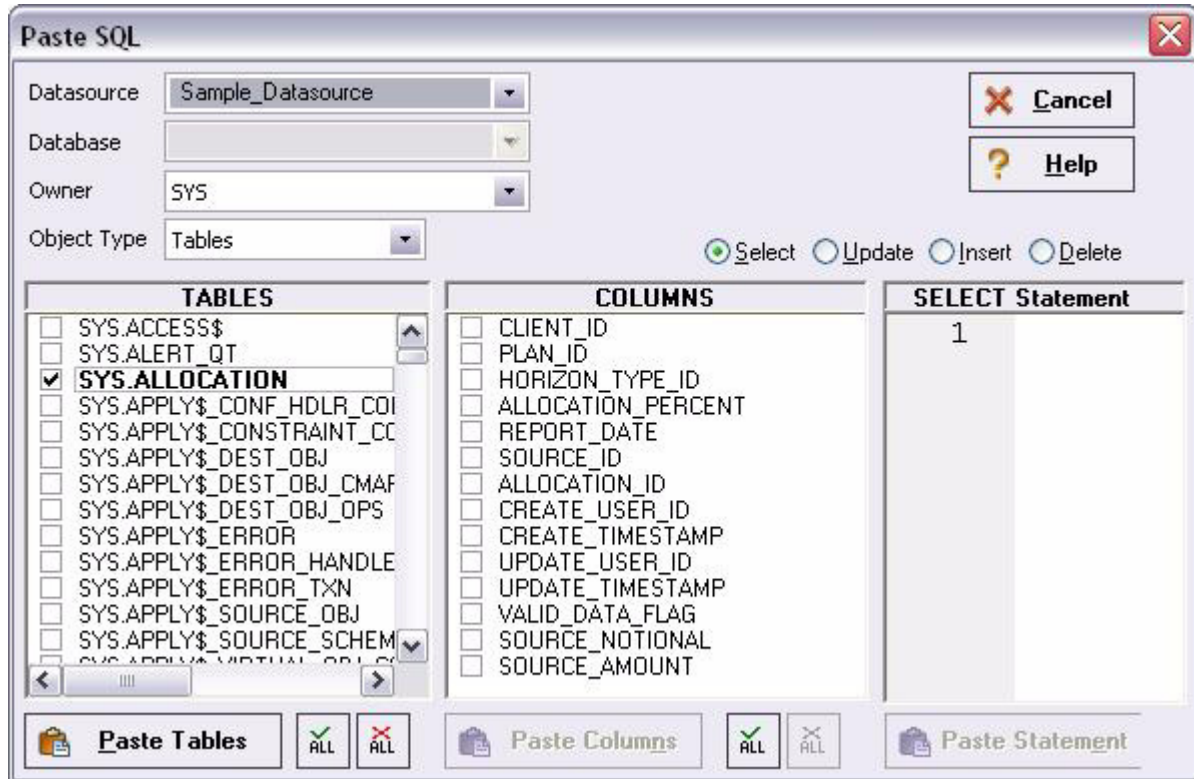
4   In the **Tables** list, select **authors**.

5   Under the **Columns** list, click **ALL**.

6   Click the **Select** radio button.

7   Click **Paste Statement** to copy the generated code to the SQL Editor window.

You can use the statement as is, or modify the code as needed.

PASTE SQL SYNTAX

1   Select **File > New > SQL** to open a fresh SQL Editor window.

2   Select **Edit > Paste SQL Syntax** to open the **SQL Syntax for SQL Server** window.
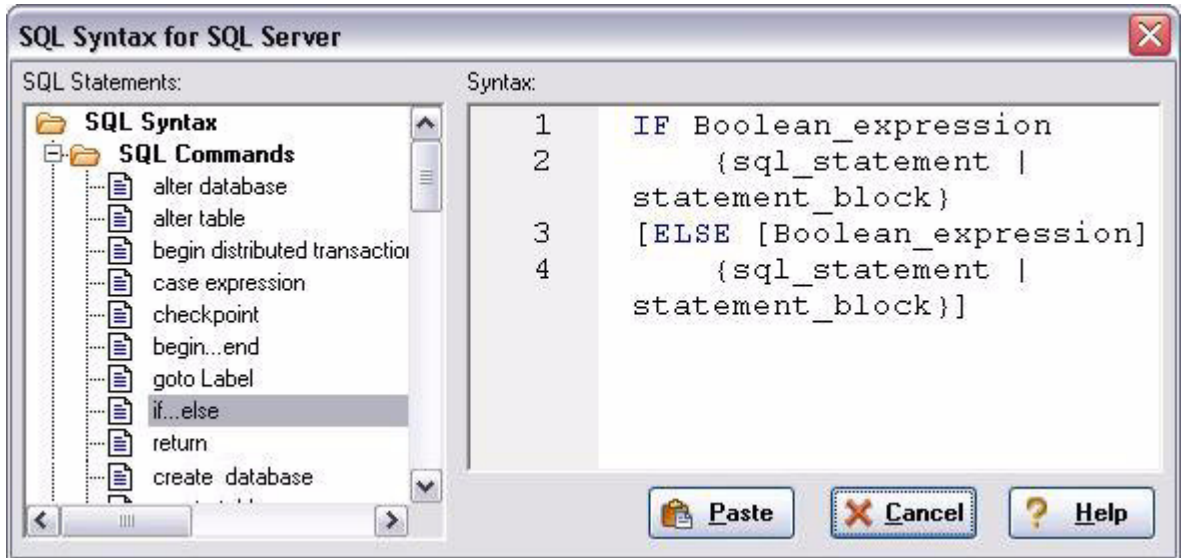
3   Select a template and click **Paste** to copy the code template into the SQL Editor window.

4   Add your own code to complete the needed operation.

SESSION 4: WORKING WITH CODE WORKBENCH

The Code Workbench lets you configure resources for two SQL Editor features:

- Code Templates

- Auto Replace

Code templates are complete code blocks that can be easily added to open windows or scripts with a few keystrokes. When you type CTRL+SPACE, the Code Assist menu opens, letting you select a code template for insertion in the editor window.
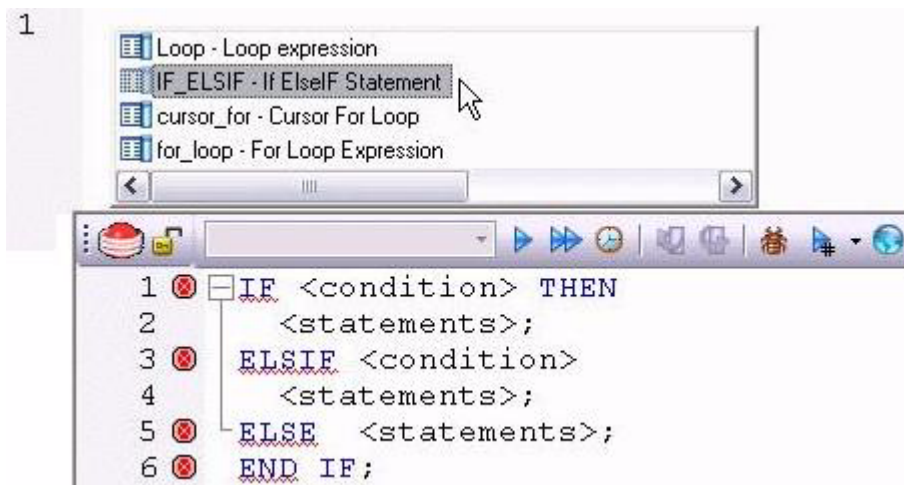
Auto Replace lets you define shortcuts consisting of a few characters that represent longer character strings. Instances of these Auto Replace expressions are automatically replaced by the replacement string on activation events such as typing SPACE, TAB, or RETURN. This feature is useful for creating shortcuts for one-line commands or SQL statement subsets, or even to detect and fix common typographical errors such as teh for the.

For example, consider an Auto Replace definition with an expression of **sel** to represent **Select * From**:

If the associated activation event includes a SPACE, then on typing **sel** followed by pressing SPACE, the following replacement occurs.

Rapid SQL loads a default set of Auto Replace and Code template definitions at startup but you can also add, edit, and delete definitions. In addition, you can save sets of definitions to file and subsequently load specific sets of definitions, allowing you to customize your templates to different platforms or development projects.

To invoke Code Workbench settings:
1 Select **Tools > Code Workbench**.

   The **Settings** tab lets you enable the Auto Replace and Code Template features.

2 Inspect the **Code Templates** and **Auto Replace** tabs.

3 Click **OK**.

SESSION 5: BUILDING A DATABASE PROJECT

CREATING A NEW RAPID SQL PROJECT

Rapid SQL provides an excellent team development environment by allowing you to reverse engineer live database objects into off-line SQL source code files that can then be easily added to a Version Control System (VCS). Rapid SQL's seamless VCS integration offers all version control operations, such as get, check-out, check-in, history and diff. This example will reverse engineer the table objects from the Microsoft SQL Server **pubs** database into a Rapid SQL project and then will add the project to version control under Microsoft Visual Source Safe.

1 Select **File > New > Project** to open the wizard.

2 Enter **sample_project** as the name, and browse and select a directory that contains a VSS database. Enter a description (optional). Select **From Database** and click **OK**.

3 Select **SAMPLE_DATASOURCE** and click **Next**.

4 Select **pubs** and click **Next**.

5 Select **dbo** as the owner. Right-click in the object type selection window and deselect all of the options. Select only **Tables**. Under **Extract Scope** select **Selected Objects Only**. Click **Next**.

6 Select only the **authors**, **discounts**, and **employees** tables.

7 Uncheck all selected **Options** for tables. Click **Next**.

8 Select **Retain**. Click **Next**.

9 Preview the last panel and click **Execute**.
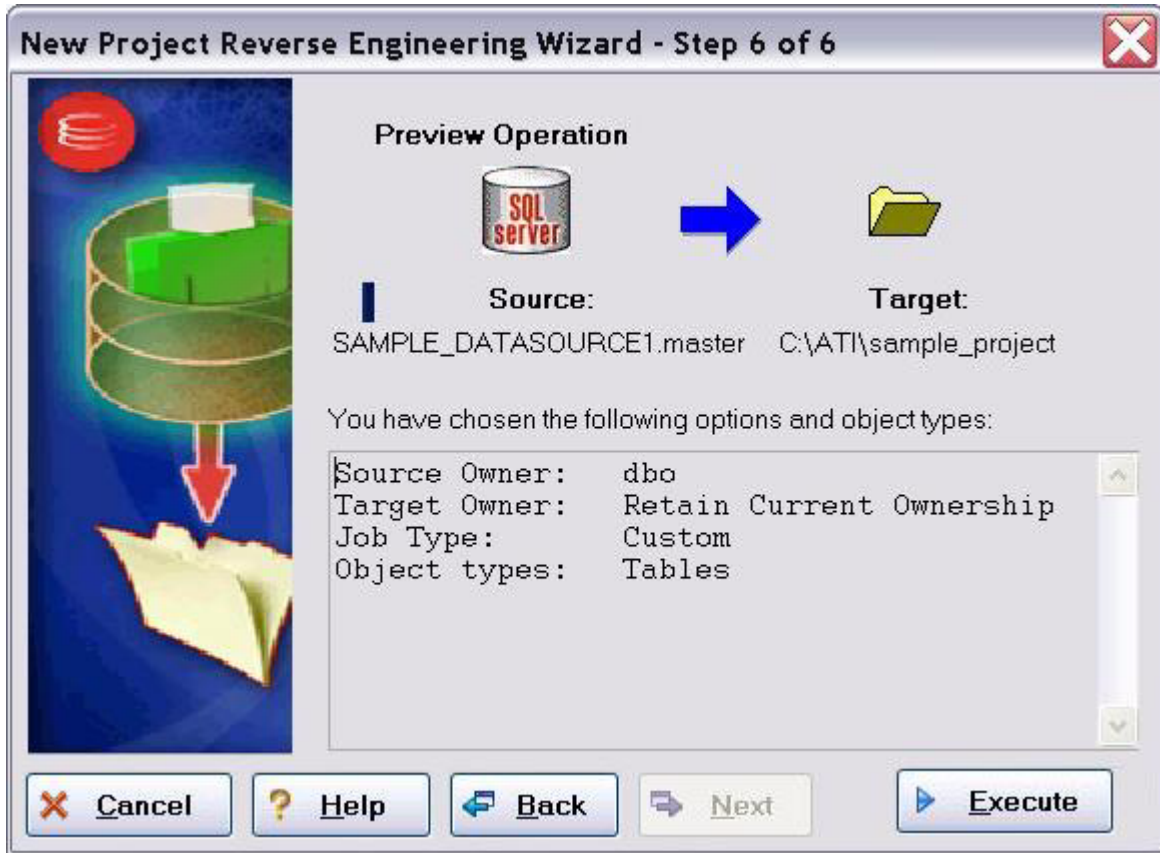
Finished! You have now successfully created a working database project. At this point, the project is available to be added to Version Control. This can be done by clicking Yes when prompted and following the dialog, or by right-clicking on the project within Project Explorer tree. Select Yes for the purpose of this guide.

ADDING A PROJECT TO VERSION CONTROL (SAMPLE - MICROSOFT VISUAL SOURCE SAFE)

When a project is created Rapid SQL will automatically prompt you to add the project files to the selected VCS solution. The following dialog will be displayed:

After providing a user name and password, you can browse to locate and select the project database folder.
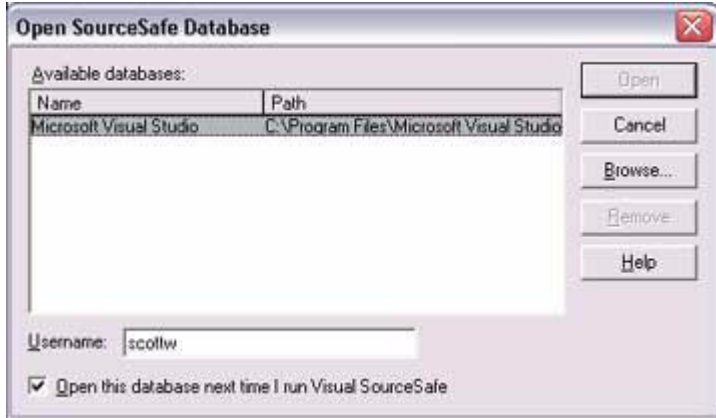
All that remains is to provide a project name.

Finished! The following message indicates that the project was successfully placed under version control.

SESSION 6: VISUAL QUERY BUILDER

Rapid SQL gives you the ability to construct complex SQL statements with point-and-click ease using the Visual Query Builder.

1   From the Database Explorer Tree, right-click on the **authors** table and select **Build Query**.

    The **authors** table will be automatically added to the Query Builder workspace.

2   On the **Tables/Views** tab, right-click on the **titleauthor** table and select **Add**.

Note that the tables are automatically identified as being joined by any columns with the same name and datatype.
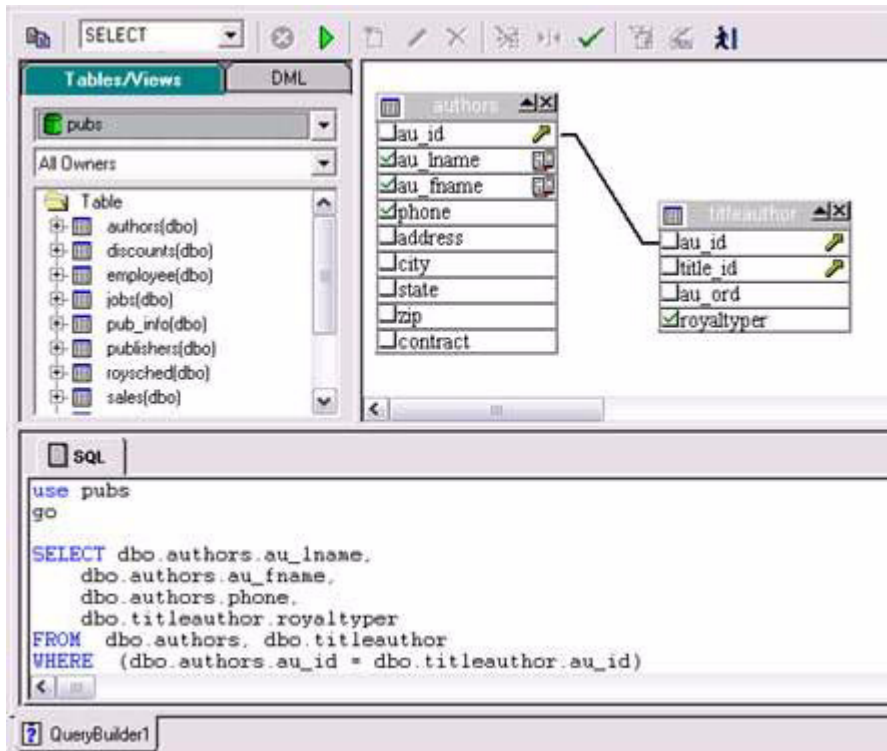
3   Click on the **DML** tab to expose the visual query building clauses and options. You can right-click on any clause to easily add the code to the query.
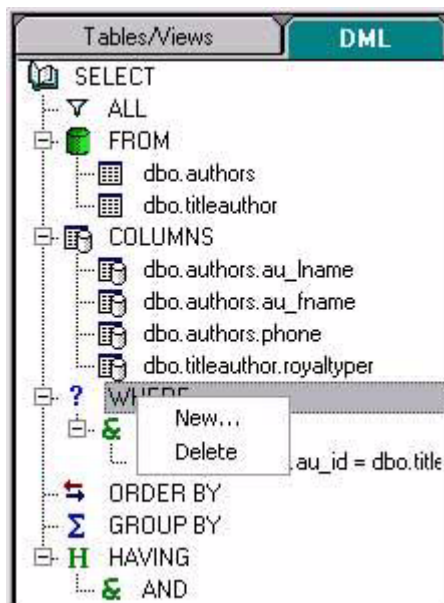
4   Select the **au_lname**, **au_fname**, and **phone** check boxes in the **authors** table.

Note that the lower pane shows the query that is being built.

5   Click the Execute button to execute the query.

▷

The results will display in the lower window.

Before closing the Query Builder, experiment with additional options. Try selecting a different statement type, such as **Insert** or **Update**, from the dropdown at the top of the Query Builder window. Use the different clauses on the **DML** tab.

NOTE: Any visual query builder session can easily be saved to a file for later use.

SESSION 7: LIVE DATA EDITOR

1   From the Database Explorer Tree, right-click on the **authors** table and select **Edit Data**.

2   In the **Data Editor Filter** dialog, click **Add All** to add all columns to the editing session.

At this point, you can add a WHERE clause that will filter for only the desired data. Note that Rapid SQL builds the SQL to retrieve the data to be edited in the **Select Statement** area.

3   Click **OK**. A Data Editor opens.
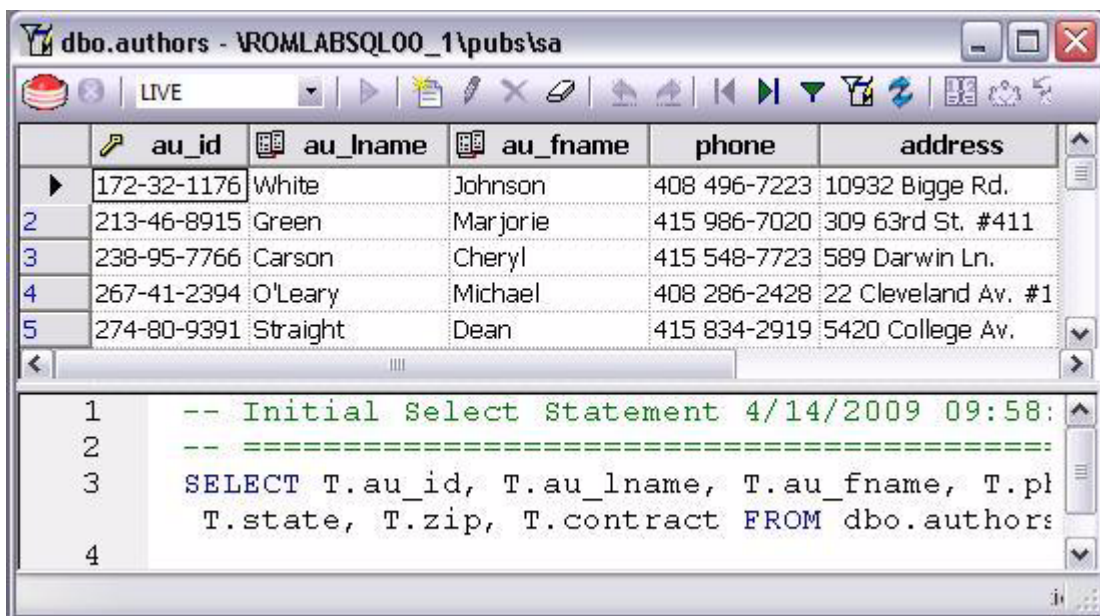
Note the dropdown at the left of the toolbar. The editing window has **LIVE** and **BATCH** modes:

• **LIVE** mode commits your changes each time you move to a new row.

- **BATCH** mode will allow you to move within the window and commit your changes when ready. Changes made in **BATCH** mode can be cancelled by selecting the Reload Data icon.

At any time during the session, you can change the filter parameters by selecting the Filter Data icon.

SESSION 8: CODE ANALYST

The Code Analyst allows you to capture run-time statistics on executable database objects, including stored procedures and functions. Not only can you capture runs for single objects, but you can group more than one object.

To get started

1   Select **Tools > Code Analyst**.

> NOTE: In order for Code Analyst to run, 5 repository tables will be created on the database. Select the database you would like the tables to be installed on and press **OK**. Once the tables are installed, you're ready to start defining a session.

2   On the Code Analyst toolbar, click the **Create New Collection** button.

3   On the **Code Analyst Object Selection** dialog, provide a **Session Name**, locate and selecting the objects to be executed, and click **Next**.
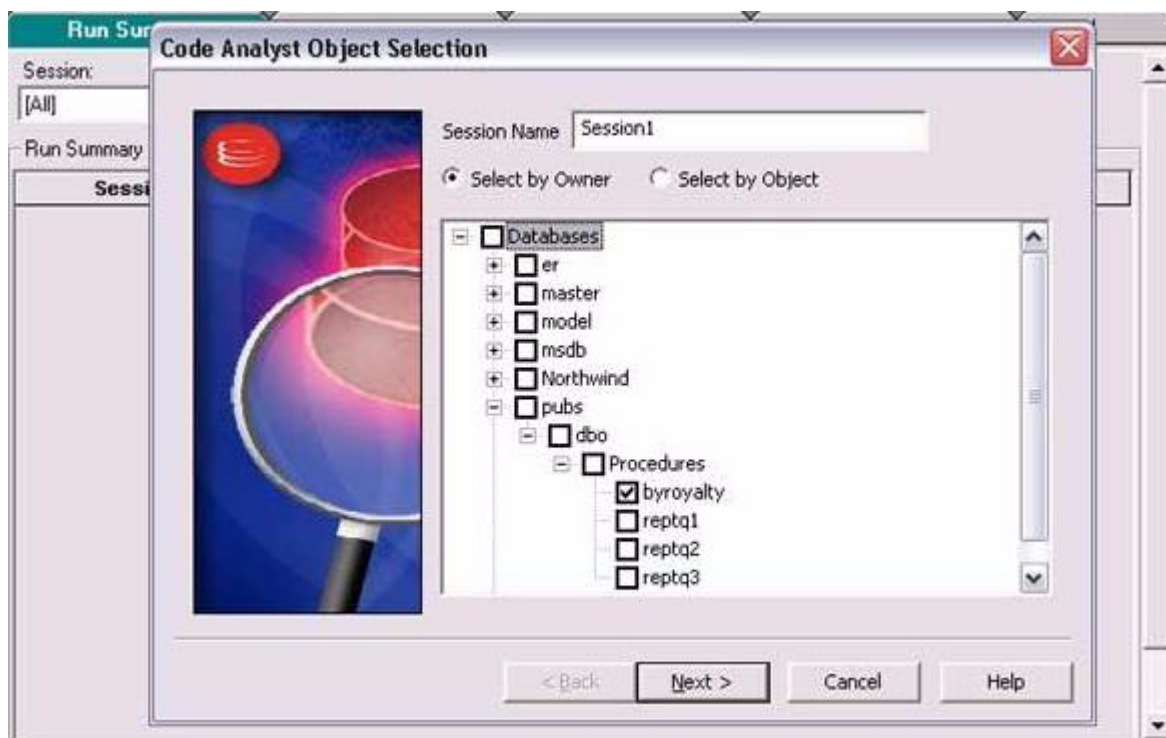
4   Use the **Code Analyst Object Initialization** dialog to initiate providing input parameters as required, change the order of execution, and when ready, click **Finish**.

Once the session has been run, the total time for the run is displayed in the **Run Summary** tab.

5   Select the other tabs to view the tabular and graphical representation of the execution details on your selected objects. For example:

- The **Run Detail** tab shows a breakdown of the different objects that make up the session.

- The **Unit Detail** tab contains the specific time measurements for individual SQL statements.

6   Close the **Code Analyst** window.

SESSION 9: SQL DEBUGGING AND PROFILING

Rapid SQL offers the following facilities that help you test and optimize code:

- SQL Debugging

- SQL Profiling- Oracle Only

SQL DEBUGGING

The SQL Debugger is another database productivity tool that lets you debug SQL Server, Oracle, Sybase or DB2 stored procedures as well as Oracle functions. SQL Debugger simplifies the task of finding coding errors.

1   In the Explorer, expand the **Procedure** or **Function** node.

2   Right-click the object and select **Debug** from the context menu.

3   If the procedure or function takes input parameters, the **Procedure Execution** window prompts you to enter values.
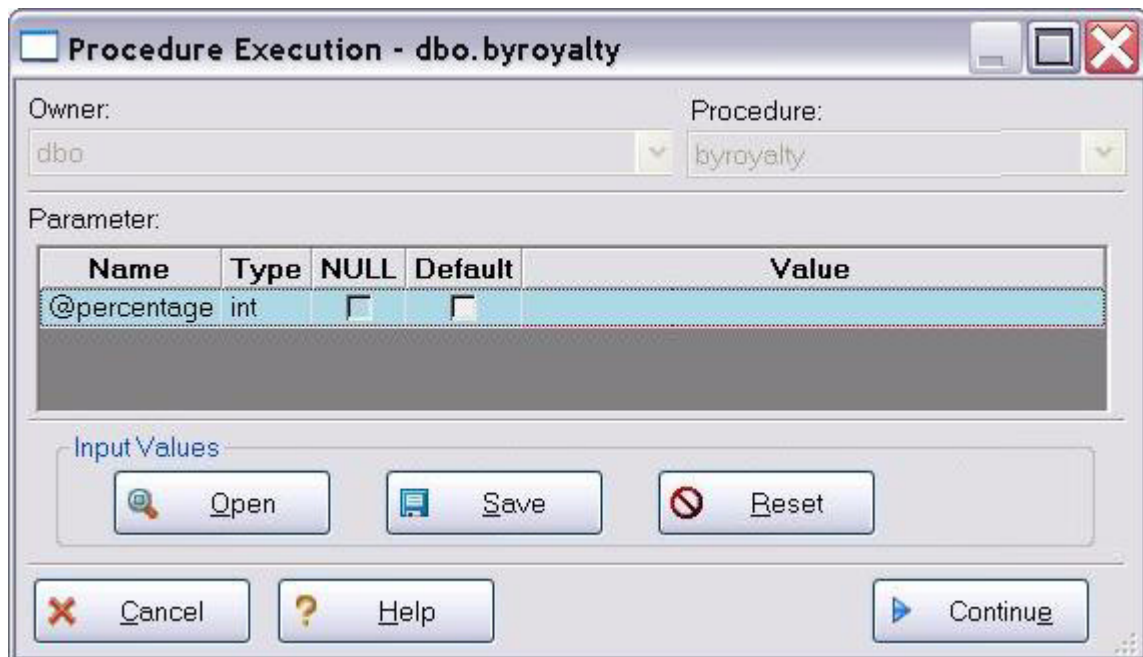
4   Enter input values(s) and press **Continue**.

> **TIP:**   Rapid SQL allows the user to save the input variable values to a file for later use. This is very helpful for procedures/functions with many input variables that need to be run repeatedly.

The application opens the SQL Debugger Interface.
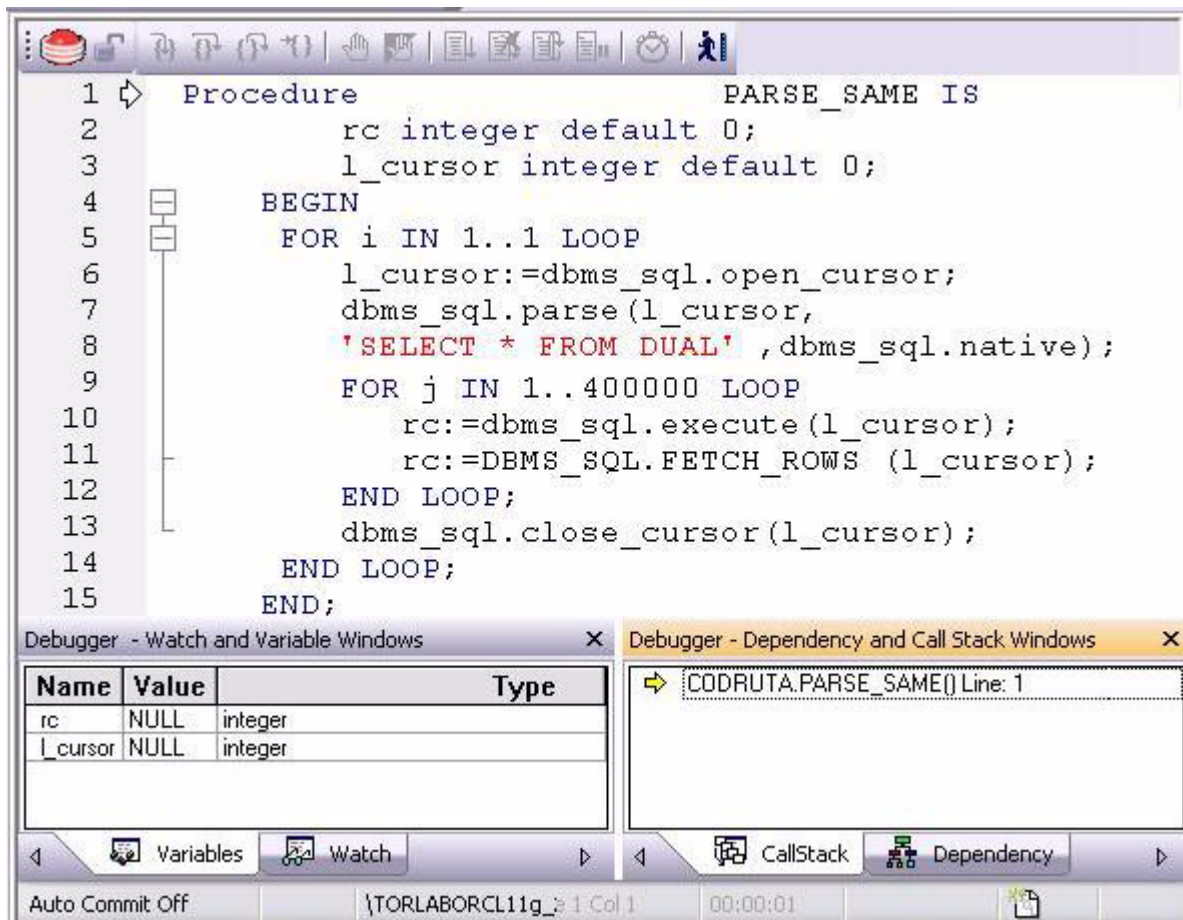
The Debugger features basic execution, line-by-line execution, breakpoint support, and other common debugging features. For details, refer to the relevant online Help topics.

SQL PROFILING- ORACLE ONLY

The SQL Profiler within Rapid SQL provides the ability to capture the metrics of various PL/SQL programmable objects as they are executed in the database. It quickly identifies performance bottlenecks by first calculating the overall runtimes of objects like Oracle packages, and then computing the amount of time each line of PL/SQL code spends executing. Information is presented in an easily viewed, drill-down format.

1 To start a profiling session, use the **Tools** menu option and select **SQL Profiler > Start**.

2 Enter a name for the profiling session or select an existing name from the dropdown. Press **OK**. The Profile session is now active.

3   Execute the programmable object (i.e. Stored Procedure) you wish to capture metrics on.

4   When finished, select **Tools > SQL Profiler > Stop**. The **SQL Profiler – Stop** dialog window prompts you to select an option.
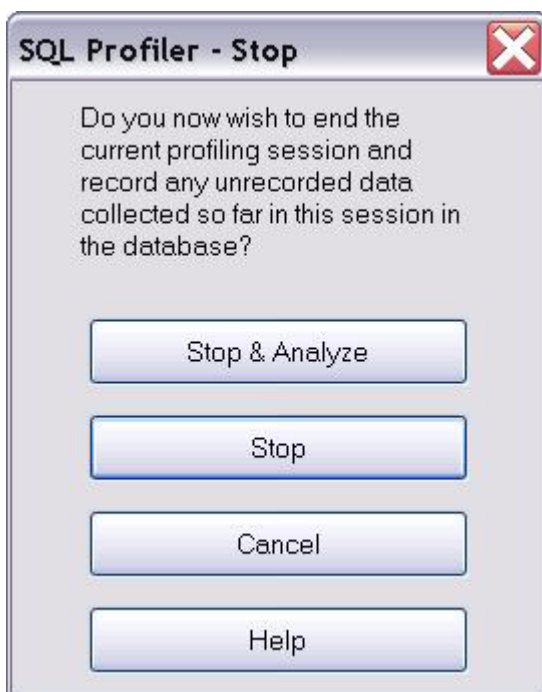
5   Press **Stop**.

6   On the Explorer, expand the **PL/SQL Code Profiling** node.
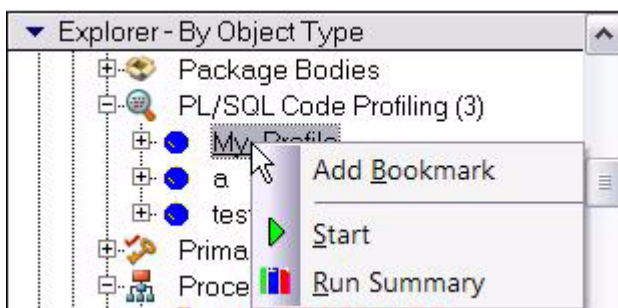
7   Right-click on the profile session and select **Run Summary**. The **Run Summary** window opens.

Comment	Run ID	Run Date	Total Time (ms)
My_Profile	60	01/09/2009 12:00:43 PM	47261
My_Profile	61	01/09/2009 12:01:52 PM	5484

8   Select a session and select **Run Detail** from the right-click menu. The **Run Detail** screen appears allowing you to view the metrics for this run in both a graphical and text format.
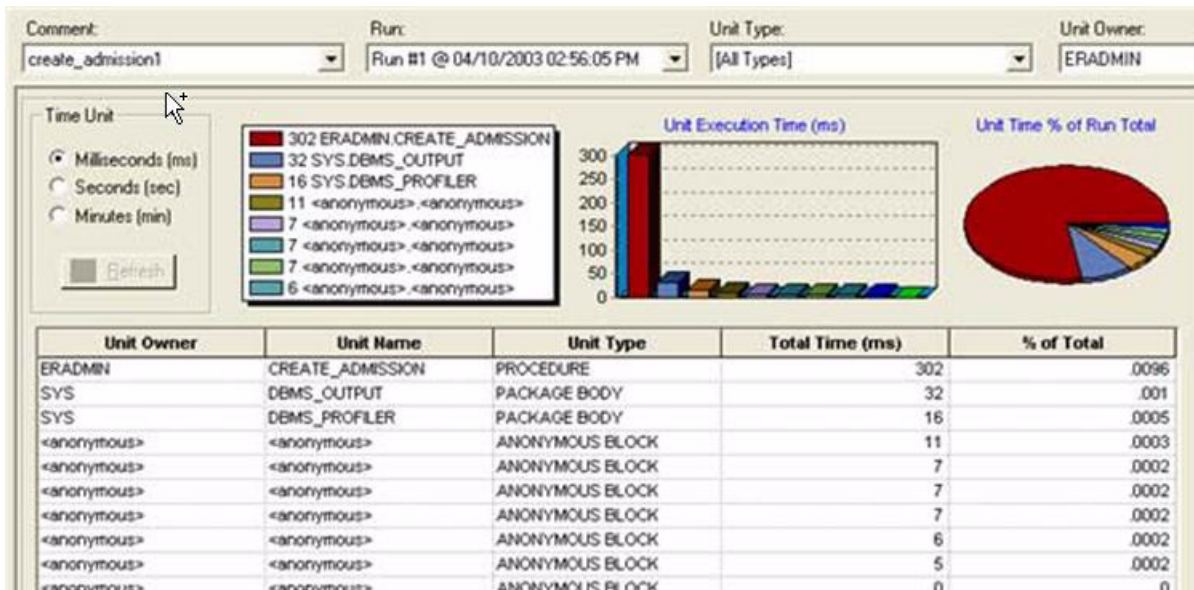
9   To drill down further into the data, highlight a unit and select **Unit Detail** from the right-click menu. Scroll through the Source window to view the times for each statement.

10 To compare 2 cases, select the 2 cases you wish to compare (shift-click to select the second case) from the **Run Summary** screen and select **Compare** from the right-click menu. The **SQL Profiler Run Comparison** screen appears.

See the relevant online Help topics for more information on profiling.

SPECIFYING RAPID SQL APPLICATION AND FEATURE OPTIONS

Rapid SQL lets you customize the application configuration for your specific needs. All Rapid SQL application settings are available in the Options Editor, which is organized in a tabular format based on feature sets.

To specify options for the Rapid SQL application or a particular feature

1   On the *File* menu, click *Options*.

    Rapid SQL opens the Options Editor.

2   Click the tab corresponding to the feature you want to customize:

Browsers Options	Lets you specify appearance of browser windows.
Code Analyst Options	Lets you set dependency profiling level, alarm and threshold, and oracle options for Code Analyst.
Connection Options	Specifies the timeout parameters, packet size for a connection, and ANSI to OEM settings.
Data Editor Options	Specifies settings for Data Editor.
Data Transfer Options	Sets default directories when performing data unload and load and data export and import operations.
Explorer Options	Sets defaults for the organization of objects in the Datasource Explorer.
Datasource Options	Specifies general datasource management options and Datasource Explorer preferences.
DBMS_Java Options	Specifies load Java files and drop Java files.
DDL Extract Options	Specifies whether or not Rapid SQL should include DROP statements when extracting the schema of different database object types.
Debug Options	Sets the duration of your debug initialization and debug session, enable or disable DBMS_OUTPUT, and enable the refresh option.
Directories Options	Sets the default directories for placing the output of different operations such as HTML reports or schema extractions.
General Options	Sets defaults for automatic login, restoring the last session, and other general application options.
Grid Properties (Results window) Options	Dictates the physical appearance of the results window grid.
ISQL Options	Sets defaults for the maximum allowable errors before aborting the execution of a SQL script, executing selected text, and the position of Query and Results tabs.
Java Options	Lets you specify Java Virtual Machine options.
Logging Options	Sets defaults for SQL Logging.
MySQL Utilities Options	Specifies paths to MySQL utilities.
Oracle Utilities Options	Specifies the location of the Oracle Utilities.
Perf Center Options	Specifies Performance Center's integration with Rapid SQL.
Query Builder Options	Specifies global settings for Query Builder.

Results (ISQL) Options	Specifies auto format result sets, sets the display and format of Result Windows, and the mail file type and default fonts.
SMTP Mail Options	Lets you specify defaults for your outgoing mail notifications.
Version Control Options	Lets you select which version control system you want Rapid SQL to use as the underlying version control system
Warnings Options	Activates specific warnings when undesirable actions are attempted against a database.

3   Set feature options on the tab and then click OK.

> **NOTE:** If there is an open document, Rapid SQL opens the **Update Document Statement Properties** dialog box. The Update Document Statement Properties dialog box lets you override changes you made to a current document or documents with new setting you made in the Options Editor.

BROWSERS OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Browsers** tab.

The Browsers Tab of the Options Editor lets you set a number of options for your browser windows, including browser appearance, mail file type options, and default object owner.The table below describes Browser options:

Interface Element	Option	Description	Default
Window	**Show Toolbar**	Toggles the Browser window toolbar. Commands can also be activated from the shortcut menu that displays when you right-click in an open Browser window.	Selected
	Show Status Bar	Toggles the status bar at the bottom of the Browser window. The status bar displays the cell location of the current focus and the number of rows in the Browser window itself.	Selected
	Detailed Listing	Toggles detail columns for a given object type. Each object contains object specific details, such as creation date, segment, and so on. When Detailed Listing is turned off, only the names of objects and the numbers of rows of data for a table are displayed.	Not selected
Appearance	**Text Color**	Sets the text color for all Browsers of that object type. Select the row appropriate to a particular object type, then click a color in the Text column. Remember that text colors appear best against contrasting background colors, such as black on white.	Available
	Background Color	Sets the background color for all Browsers of that object type. Select the row appropriate to a particular object type, then click a color in the Background column. Remember that text colors appear best against contrasting background colors, such as black on white.	Available
Browser File	**Mail File Type**	Specifies what file type you want the browser file saved as when sent as e-mail to another user on a MAPI compliant mail system. The valid types are the proprietary Rapid SQL Results type, Tab delimited, Comma separated, and HTML, which formats the results in a simple HTML table.	Results

Interface Element	Option	Description	Default
Default Owner	**All Owners**	Indicates that the Browser Window should default to displaying database objects in the browsers for all owners in the database.	Not Selected
	Current User	Indicates that the Browser Window should default to displaying database objects owned exclusively by the current user.	Selected
	Specific Owner	Indicates that the Browser Window should default to displaying only the database objects owned by a specific owner. If you select this option, you must provide an owner name in the box.	Not Selected

CODE ANALYST OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can set preferences for Code Analyst. For information on how to use Code Analyst to identify time-consuming lines of code, see Code Analyst.

The table below describes Code Analyst options:

Group	Options and descriptions	
Dependency profiling level	**All Dependencies**	Code Analyst will use all levels of dependencies when profiling the session.
	Dependencies up to level	Lets you specify how many levels of dependencies the Code Analyst will use when profiling the session.
Oracle	**List Package objects**	Code Analyst lists the procedures within a package and functions that are found in the Oracle database.
	Use SQL Profiler	Code Analyst uses Oracle's DBMS_Profiler package to collect time metrics. Code Analyst displays the actual run time on the database, and does not include the time it takes to get to the server.
Alarms and Thresholds	Lets you set an alarm for each type of object that is collected. For each supported **Object Type** (procedure, function, package, trigger, type), you can provide the following settings: **Alarm %** - lets you specify the percentage of total run time a line of code takes when an alarm appears. If the object took more than specified percent of total time, Code Analyst alerts the user by changing the color of the text **Ignore 100%** - lets you ignore lines of code that take all of the total run time	
Show Code Analyst Confirmation Dialog	When you create or execute a session, Code Analyst displays a message that the Code Analyst will run longer than the actual code. You can also select the "Please do not show me this dialog again" option in the dialog box.	

CONNECTION OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can configure a number of server parameters for interacting with your datasources on the Connection tab.

> **NOTE:** Setting Connection Options are available for Microsoft SQL Server and Sybase ASE.

The table below describes the options and functionality on the Connection Tab:

Option	Description	Default
Login Timeout	Specifies the number of seconds that the application should wait for a response to a connection request from server. If server does not respond within the specified period, the application aborts the connection and returns an error message.	30
Query Timeout	Specifies the number of seconds that the application should wait for a response to a query from the server. If the server does not respond within the specified period, the application terminates its query process and returns an error.	0
Packet Size	Specifies the network packet size to be used when communicating with the server, in bytes.	512
Max Connections (ctlib)	The maximum number of connections allowed by the ctlib setting on the client.	0
Client Character Set	Character set of client computer.	Local character set
Host Name	Name of the client computer.	Local name
Use Quoted Identifiers	If you plan to use delimited identifiers, select this option.	Not selected

DATA EDITOR OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Data Editor** tab.

> **NOTE:** Option parameters set in the Options Editor override options set within Query Builder and Data Editor.

The table below describes the options and functionality on the **Data Editor** Tab:

Interface Element	Option	Description	Default
Default Execution Mode	**Live Mode** lets you execute changes one row at a time. **Batch Mode** lets you make unlimited changes prior to execution. Batch Mode offers three sub-options: **Ignore errors-continue processing**, **Prompt on Error**, and **Stop after error(s)** (number of errors allowed before stopping execution)		Live Mode
Data Editor File	**Mail File Type**	Sets the default mail output style as Results, Tab Delimited, Comma Separated, or HTML.	Results

Interface Element	Option	Description	Default
	Include column titles when saving	Includes column titles when saving.	Not selected
Grid Font	Customizes font style, and size for the Data Editor and the Results Grid.		Available
Printer Font	Sets font style, and size for printing output.		Available
Auto Format (Best Fit)	Fits formatting to match your desktop.		Selected
Begin and End Transaction Statements	Adds a beginning and ending transaction on each statement.		Selected
Default Date/ Time Format	Displays the current date/time format and lets you customize the date/time display.		Results
	Use Calendar Control as default	If selected, Rapid SQL uses the Calendar Control window.	Not selected
	2 digit year system setting warning	If selected, Rapid SQL sends a warning when you use a two digit year system setting.	Selected
Confirmation Dialog Options	Enabling **Show Delete Confirmation Dialog**, **Show Update Confirmation Dialog**, or **Show Update LOB Confirmation Dialog** lets you display a confirmation dialog box when you use a delete command, update a row, or update a LOB, respectively.		Selected

DATA TRANSFER OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Data Transfer** tab.

You can configure Rapid SQL to use default directories when performing data unload and load and data export and import operations. Setting a default directory saves time because it describes a single reference point for loading or unloading, exporting and importing table or view data files.

The table below describes the options and functionality on the **Data Transfer** Tab:

Option	Description
Data Unload	Specifies the name and location of the default directory in the Data Unload box.
Data Load	Specifies the name and location of the default directory in the Data Load box.
Oracle Export (for Oracle)	Specifies the name and location of the default directory in the Oracle Export box.
Oracle Import (for Oracle)	Specifies the name and location of the default directory in the Oracle Import box.

DATASOURCE OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Datasource** tab. The table below describes the options and functionality on the **Datasource** Tab:

Option	Description	Default
Datasource Storage Management	Rapid SQL offers both registry-based and file-based datasource catalog management. Prior to changing methods, see [Understanding the Datasource Catalog](#).	Windows Registry
	The following options are offered:	
	Windows Registry datasource catalog - The datasource catalog is stored and managed locally, in the Windows registry. This user can add and delete datasources and edit the registration details of existing resources. Any changes are retained across startups of Rapid SQL.	
	File based datasource catalog - The datasource catalog is stored locally and is file-based (*%APPDATA%***Embarcadero\Data Sources** in a default installation). This user can add and delete datasources and edit the registration details of existing resources. Any changes are retained across startups of Rapid SQL.	
	Network shared datasource catalog - The datasource catalog is file-based, and obtained from the location specified in the **Location of shared datasources file** control. Changes such as new datasources, deleted datasources or edited datasource registrations are lost as soon as this user shuts down Rapid SQL.	
	Changing methods does not delete datasource definitions stored using the current method. When you change methods, Rapid SQL does not automatically convert existing catalog definitions to the new method	
Recent Datasource List Contains	Lets you specify number of datasources to display when you select **Recent Datasources** from the **File** menu.	8
Default to Alias Usage When Defining New Datasources	For DBMS that support alias usage, defaults the **Connection Information** tab to alias usage selection.	Unselected
Datasource Category Settings	Lets you select the optional user interface elements that are to be color-coded or labelled for datasources that have been assigned a category:	N/A
	Color the status bar using the datasource's category color	
	Color the tabs using the datasource's category color	
	Include category short name in the datasource toolbar combo	
	For an introduction to datasource categorization and related tasks, see [Categorizing datasources using color-coding and labelling](#).	

DBMS_JAVA OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Java** tab. The table below describes the options and functionality on the **DBMS_Java** tab:

> **NOTE:** Options differ by platform.

Interface Element	Option	Description	Default
Load Java files	**Use the DBMS_JAVA package**	Lets you schedule your SQL.	Selected
	Use batch file (oracle_home/bin/ loadJava.bat)	Uses batch file (oracle_home/bin/ loadJava.bat)	Not selected
	Default Encoding option	Leave blank to use the default.	Default
Drop Java files	**Use the DBMS_JAVA package**	Lets you schedule your SQL.	Selected
	Use batch file (oracle_home/bin/ dropJava.bat)	Uses batch file (oracle_home/bin/ dropJava.bat)	Not selected

DDL EXTRACT OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **DDL Extract** tab.The DDL Extraction Tab of the Options Editor lets you specify common and DBMS-specific DDL Extract options.

Common options

The table below describes the options and functionality on the **DDL Extract** Tab, main view, of the Options Editor:

Option	Description
Extract to multiple windows	Select to extract the schema for each object into separate DDL windows.
	NOTE: This option only works when you extract DDL for multiple objects.
Extract in dependency order	This is the default.
	When you select a number of objects as part of an extraction job, this option ensures objects will be extracted in the proper dependency order.
	If the option is not selected, errors may result when you execute the script.
	It's also true, however, that loading the dependencies can add significant overhead when you are extracting numerous objects.
Script Use Statement	This option is for Sybase. Optimizes extraction through generating Use statements in the generated script.

Option	Description
Grouping Extracted DDL	There are two option as to how the DDL is generated for an extract operation: **Group the Drop and Create DDL for each object together** - Objects are extracted one at a time. In the DDL generated by the extract, the Drop statement for a selected or referenced object is followed by the Create statement for that object. **Group DDL statements by type - all Drop statements, then all Create statements** - In the DDL generated by the extract, Drop statements for all selected or referenced objects are grouped to precede Create statements for those objects. This method ensures that all dependencies are respected. For more information on DROP options and dependencies, see DBMS-specific DDL Extract Options.

DBMS-specific DDL Extract Options

The DBMS-specific views of the DDL Extract tab let you specify:

- The object types to include DROP statements for

- The default dependent object types for each object type included in extraction/migration operations

You can choose to include DROP statements before you perform ad hoc DDL extractions. You can use this feature to modify and to re-compile database objects. To recompile a database object, drop it before recreating it. This option drops any existing objects of the same name before recreating the object. The data in the existing table is not saved when you specify a DROP statement for extracted DDL.

CAUTION: Because dropping an object is a destructive action, you should carefully consider including drop statements before activating this option.

You can also specify dependent object types included in extraction/migration operations. Each DBMS-specific view of the DDL Extract tab lets you specify the object dependencies for each object type. Dependent object types selected on that view are by default selected when you run an extraction/migration operation and can be overridden using those wizards. For more information see Extract.

DEBUG OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Debug** tab. The Debug Tab of the Options Editor lets you set the duration of your debug initialization and debug session, enable or disable DBMS_OUTPUT, and enable the refresh option.

The table below describes the options and functionality on the **Debug** Tab:

Tab	Option	Description	Default
General	**Dependency Tree Option**	Lets you specify pre-fetch levels.	Pre-Fetch All Dependencies

Tab	Option	Description	Default
Profiler	**Profiler Time Unit**	Lets you select a unit of milliseconds, seconds or minutes.	Milliseconds
	Save Profiler Reports	Lets you save profiler reports and type or browse for the report path.	Not Selected
Oracle	**Initialization Timeout (seconds)**	Specifies the point at which the application stops trying to initialize the debugger. If it cannot initialize the debugger in the specified time, it displays message in the Debug Output window.	60
	Debug Session Timeout (seconds)	Specifies the point at which the application terminates your debug session due to idle time.	7200
	Enable DBMS Output	Enables the Oracle built-in package, DBMS_OUTPUT, letting you send messages from stored procedures, packages, and triggers.	Selected
	Refresh Dependencies for each run	Refreshes the dependencies each time you run the debugger.	Not selected
	Compile with Debug Option	When debugging a procedure against an Oracle datasource, Rapid SQL performs a compilation before debugging. During a compilation, Oracle invalidates referencing objects. For example, if procedures A2 and A3 both reference procedure A1, debugging procedure A1 in Rapid SQL, will result in procedures A2 and A3 being marked as invalidated. The **Compile with Debug Options** settings let you control compilation of dependent objects while debugging.	Compile dependent options
DB2	**Debug Session Timeout (seconds)**	Specifies the point at which the application terminates your debug session due to idle time.	300
	Compile with Debug Option before Debug Session	Lets you specify options.	Prompt Always

DIRECTORIES OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Directory** tab. Rapid SQL lets you configure Directory Options to use default directories when performing certain operations. You can set the default directories for:

- Wizard operations

- Report generation

- Schema extraction

- HTML templates for customizing reports

The table below describes the options and functionality on the **Directories** Tab:

Option	Description	Default
Wizard Definitions	Specifies the name and location of the default directory for saving wizard operations. This option applies to wizard operations in which you have the option to save the definition file, such as data unload/load or schema migration operations.	**C:\Documents and Settings**\user name**\Application Data\Embarcadero\RapidSQL\DefFiles**
HTML Reports	Specifies the name and location of the default directory for the output from generated HTML reports.	**C:\Documents and Settings**\user namer**\Application Data\Embarcadero\RapidSQL\Report**
Schema Extraction	Specifies the name and location of the default directory for placing the output from schema extraction operations.	**C:\Documents and Settings**\user name**\Application Data\Embarcadero\RapidSQL\Extract**
HTML Template	Specifies the name and location of the default directory where Rapid SQL can find the HTML template on which to base HTML reports. This feature lets you customize your HTML reports.	**C:\Program Files\Embarcadero\RSQL**<version identifier>**\HtmlTpl**
User SQL Scripts	Specifies the name and location of the default directory for SQL Scripts.	**C:\Documents and Settings**\user name**\Application Data\Embarcadero\RapidSQL\UserSQLScripts**
Job Config Files	Specifies the location of any ETSQLX job configuration files you have set on your local machine.	**C:\Documents and Settings\davidt\My Documents\Embarcadero\RapidSQL\Directories\ETSQLXJobCf**g
#include Files	Specifies the name and location of the directory searched for files specified by a #include directive in the ISQL editor, Procedure Object Editor, or Package Body Object Editor if there are no paths specified on the Datasource Properties tab of the Datasource Registration Editor. For more information, see [Registering or editing datasources](#).	**C:\Documents and Settings\davidt\My Documents\Embarcadero\RapidSQL\Directories\IncludeFiles**

EXPLORER OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Explorer** tab. The Explorer Tab of the Options Editor lets you configure how objects are organized in the Datasource Explorer.

The table below describes the options and functionality on the **Explorer** Tab:

Option	Description	Default
Explorer organization	Lets you select a default grouping option of **Organize by Object Owner** (groups objects, by object type, for each user**)** or **Organize by Object Type** (groups objects by object type for all users in the same list). Grouping by object owner is most efficient if you are working with databases containing a high number of objects. You can change the organization manually during a Rapid SQL session.For details, see Explorer Tab.	Owner
Show System Objects and **Show Only My Objects** (available if you are organizing the Datasource Explorer by object type)	Lets you select a default Datasource Explorer tree display options for system objects and objects that you own. Available options in the two groups are: **Never** - the relevant objects are not displayed by default. **Always** - the relevant objects are always displayed by default. **Use datasource filter** - The relevant default datasource filter is enabled.For more information, see Filters Node. You can change the **Show System Objects** and **Show Only My Objects** settings manually during a Rapid SQL session.For details, see Using the Default Filters.	N/A
Options	Lets you enable or disable the following settings: **Refresh after Object Commands** - Refreshes the Explorer automatically after an object has been modified or created. **Retain View Setting on Close** - Select to retain the current state of the Datasource Explorer so that it opens the same way the next time you start Rapid SQL.	N/A

GENERAL OPTIONS

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **General** tab. The General tab of the Options Editor lets you specify general application options.

The table below describes the options and functionality on the **General** Tab:

Tab	Option	Description	Default
Main tab	**Confirm on Exit**	Instructs Rapid SQL to issue a message confirming the operation before exiting the application.	Selected
	Max Editors in a Single Open Operation	Specifies the maximum number of editors allowable from a single Open operation.	5
	Max Entries in Output Window	Specifies the maximum number of messages that can appear in the Output Window before the contents are flushed. This option conserves memory resources. You can clear the Output window or raise the maximum number allowed at any time.	1500

The General tab also offers the following DBMS-specific aub-tabs:

Oracle sub-tab

Option	Description	Default
Data Dictionary View Usage	Lets you synchronize Rapid SQL use of the Oracle Data Dictionary DBA and ALL/USER views with the privileges associated with the role used in creating and using the datasource. **Role-based Views** - Rapid SQL will use views based on the user's role. That is, users with DBA privileges will use DBA views. **DBA Views** - Rapid SQL always uses DBA views. **All Views** - Rapid SQL always uses ALL views. **Use Datasource Configuration for Views** - lets you set up Data Dictionary usage on a view-by-view basis when editing or registering a datasource. For more information, see <u>Registering or editing datasources</u>.	
Preserve Case in Object Identifiers	Preserves case of the database object.	Not selected

Sybase and SQL Server sub-tabs

Option	Description	Default
Rename action style	Lets you specify whether user-defined objects or datatypes are renamed using an extended ALTER or using the `sp_rename` procedure.	extended alter

InterBase/Firebird sub-tab

Option	Description	Default
Preserve Case in Object Identifiers	Preserves case of the database object.	Not selected

GRID PROPERTIES (RESULTS WINDOW) OPTIONS

After opening the Options editor (see <u>Specifying Rapid SQL application and feature options</u>), you can make changes to the Grid Properties tab.

The Grid Properties tab of the Option editor lets you set preferences for the layout and appearance of the grid in an Results window. The table below describes the options and functionality on the **Grid Properties** tab:

Interface Element	Options and descriptions		Default
Titles and Gridlines	**3D-Buttons**	Enables or disables a 3-D appearance on row and column headings in the grid.	Set
	Horizontal Gridlines	Enables or disables ruling between rows of the grid.	Set
	Vertical Gridlines	Enables or disables ruling between columns of the grid.	Set
	Mark Headings	Enables or disables highlighted row and column headings.	Set

Interface Element	Options and descriptions	Default
Preview	Displays a preview of the settings currently selected in the TItles and Gridlines group.	

ISQL OPTIONS

The ISQL options are available on the following tabs:

- [ISQL Options - Main Tab](#)

- [ISQL Options - Oracle Tab](#)

- [ISQL Options - DB2 Tab](#)

- [ISQL Options - Sybase Tab](#)

- [ISQL Options - SQL Server Tab](#)

- [ISQL Options - MySQL Tab](#)

- [ISQL Options - DB2 OS390 Tab](#)

- [ISQL Options - InterBase Tab](#)

- [ISQL Options - Editor Tab](#)

- [ISQL Options - Auto format Tab](#)

- [ISQL Options - Code Assist Tab](#)

ISQL OPTIONS - MAIN TAB

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the main tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Max Errors Before Aborting Execution	Sets the maximum number of errors allowed before aborting the execution of a script. A zero indicates that the option is inactive and that you recognize no limit to the number of errors allowed. Rapid SQL refers to this value when step executing SQL scripts.
Execute Selected Text	Executes a portion of a selected SQL script.
Check Syntax When Executing	For DB2, required to execute DB2 call statements in the ISQL Window.
Automatically lock connection	When disabled, a prompt to commit or rollback the transaction is displayed when you close the ISQL editor window. Enabling this options disables the prompts and locks the connection automatically.
Prompt to lock database connection	Locks the database connection on execution.
Tabs	Sets the appearance of the ISQL Window tabs to either the top or bottom of the ISQL Window.

File Association	Specifies whether the application should open an unknown file type automatically into an ISQL Window or prompt you with a message that Rapid SQL does not recognize the file type.
Query Plan Layout	Sets the default orientation of a graphical query plan. For more information, see <u>Query options</u>.

ISQL OPTIONS - ORACLE TAB

After opening the Options editor (see <u>Specifying Rapid SQL application and feature options</u>), you can make changes to the **Oracle** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Enable DBMS Output	Lets you specify **Buffer** size. 0 is the default.
Auto-Commit changes	Applies auto commit status changes to all open windows.
View xmltype as clob	When enabled, xmltype columns are displayed as CLOBs in the Results grid and the Data editor. Without this option selected, SELECT statements that qualify xmltype columns produce an **OciTypeBinder** conversion error. With the option selected, the SELECT submitted by Rapid SQL is modified to include a getclobval() method call.
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Default Query Plan	Lets you select the default display, tree-based or graphical, when you generate a query plan. For more information, see <u>Using the Query Plan Facility</u>.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options. You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see <u>Query Options</u>.

ISQL OPTIONS - DB2 TAB

After opening the Options editor (see <u>Specifying Rapid SQL application and feature options</u>), you can make changes to the **DB2** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Set Isolation Level	Sets the default for the value of the **Isolation Level** option on the Query Options dialog. For details, see <u>Query Options</u>.
Auto-Commit changes	Applies auto commit status changes to all open windows.
Create Explain plan tables if required	If set to TRUE, Explain Plan tables are created, as necessary. If set to FALSE and you don't manually create tables, Explain Plan operations will fail.

Option	Description
Create explain plan tables on the SYSTOOLS schema	If set to TRUE, Explain Plan tables are created on the SYSTOOLS schema. If the tables already exist in the user's default schema, Rapid SQL continues to use those tables. Refer to DB2 documentation for a listing of Explain Plan tables that must be deleted in order to use the SYSTOOLS option. If set to FALSE, Explain Plan tables are created under the user's default schema.
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options.
	You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see Query Options.

ISQL OPTIONS - SYBASE TAB

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Sybase** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Auto-Commit changes	Applies auto commit status changes to all open windows.
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options.
	You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see Query Options.

ISQL OPTIONS - SQL SERVER TAB

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **SQL Server** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Enable Set Query Options	Sets the default value of the **Send Set Options** setting on the Query Options dialog. For details, see [Query Options](#). Changing this value does not affect ISQL sessions currently open.
Auto-Commit changes	Applies auto commit status changes to all open windows.
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options. You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see [Query Options](#).

ISQL OPTIONS - MYSQL TAB

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **MySQL** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options. You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see [Query Options](#).

ISQL OPTIONS - DB2 OS390 TAB

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **DB2 OS390** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options.
	You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see Query Options.

ISQL OPTIONS - INTERBASE TAB

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Interbase** tab of the **ISQL** pane. Use the following table as a guide to modifying settings on this tab:

Option	Description
Retain connection in pool after use	This option determines what happens to a connection after it is unlocked by the ISQL window. If selected, the connection will be returned to the application connection pool. If unselected, the unlocked connection will be disconnected.
Load Query Options	If this option is not enabled, Rapid SQL loads a set of default query options that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces Rapid SQL to load a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override Rapid SQL's default query options.
	You can also enable this option when saving a query options file. For information on other load and save options, specific query options offered, manually updating query options, and conditions for which query options are sent to the server, see Query Options.

ISQL OPTIONS - EDITOR TAB

After opening the Options editor (see Specifying Rapid SQL application and feature options), you can make changes to the **Editor** tab of the **ISQL** pane. The table below describes the options and functionality on the **Editor** Tab:

Group	Option	Description	Default
Window	**Show Toolbar** and **Show Status Bar**	Enables/disables these ISQL Window user interface elements.	Selected
	Maximize on new or open	Indicates that Rapid SQL should maximize the ISQL Editor. If you already have an active MDI Window that is maximized, the default behavior is to maximize a new child window. To deactivate this option, ensure you do not have any active MDI Windows, such as the Explorer.	Selected
	File Tracking	Indicates that the ISQL Editor should use the File Tracking Facility to monitor the status of a file. If a file has been modified and saved outside the application, the application loads the most current version of the file into the IISQL Editor based on the options set for Auto-Reload File (see above.)	Selected
	Auto-Reload File	If File Tracking is enabled, indicates that the application should automatically reload a file that has been externally modified without prompting you. If you turn this option off, Rapid SQL prompts you before reloading your file if external changes have been saved.	Not selected
	Auto-Save File	Indicates that files in the ISQL Editor should automatically be saved at the indicated time interval. Backup files are saved to **C:\Documents and Settings**username**\Application Data\Embarcadero\RapidSQL\Backup**. Backup files use a naming convention of the form **~ET**xxxx**.tmp**., See the registry under **HKEY_CURRENT_USER\Software\Embarcadero\RapidSQL**versio n**\Backup** for a listing of backup file names and the path of the file to which they correspond.	Selected (5 minutes)
Command History	**Save File Before Overwriting**	Specifies the action you want the application to take when selecting a command from the Command History box. You have the option to be reminded to save a file before overwriting (Ask First), to automatically save a file before overwriting (Always), or to automatically overwrite the file with the command (Never).	Ask First
	Save Most Recent	Specifies the number of commands you want to save in the Command History list in the top of the ISQL Window toolbar. The maximum value is 99.	15
Printing	Lets you select common printer options.		
Appearance	**Enable Syntax Highlighting**	Sets syntax highlighting on so that all keywords and comments are colored for easier reading and debugging.	Selected
	Show Line Numbers	Places line numbers in the left column of an ISQL Window.	Selected
	Enable Outlining	Enables and disables outlining.	Selected
	Enable Text Wrapping	Enables and disables a typical text wrap feature.	Selected

Group	Option	Description	Default
	Editor Font	Sets the font face, style, and size displayed in the editor.	Available
	Syntax Colors	Sets syntax coloring for keywords, comments, quotes, and default text for various file types and scripts from the Syntax Coloring dialog box.	Available
Formatting	**Auto Indent**	Sets automatic indentation for each carriage return, new line in your SQL script.	Selected
	Expand Tabs and **Tab Size**	Sets tabs as the specified number of spaces in result sets.	Selected (4)
Clipboard Line Endings	**None**	When copying text from the editor, no substitution is made for end-of-line characters. This provides the fastest cut and paste operation.	Selected
	Line Endings	When copying text from the editor, end-of-line characters are replaced with platform-specific equivalents (Windows - CR LF, UNIX - LF, Macintosh - CR). This lets you copy text into environments such as Notepad, that require platform-specific end-of-line characters. This option avoids unnecessary characters and incorrect display, but cut-and-paste operations take more time to complete.	

ISQL OPTIONS - AUTO FORMAT TAB

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Auto Format** tab. of the **ISQL** pane.

The **Auto Format** tab lets you specify the style and spacing of SQL statements in an ISQL window when you choose to auto format SQL in the ISQL editor.

On opening, the Preview area shows an SQL statement formatted according to the current Auto Format settings. Click **Edit** to open the **Auto Format Options** dialog. Use the following table as a guide to understanding and modifying the settings in this dialog:

Setting	Description
Keywords	Lets you select a character case (UPPERCASE, LOWERCASE, INITIALCAPS, or NOCHANGE) formatting treatment for SQL keywords.
Right Margin	Specifies the maximum number of characters per line.
New line before keyword	If enabled, a new line is forced before every SQL keyword. If disabled, lines are only forced for statement type and clause type keywords.
BEGIN..END block	If enabled, empty BEGIN..END blocks occupy a single line.
Stack lists	If enabled, a new line and indenting is forced for each item in comma-separated list such as argument lists.
List indent size	If **Stack lists** is enabled, this control lets you specify the number of spaces each list item will be indented from the current offset.
Parenthesis indent size	Lets you specify the number of indenting spaces for forced lines following an open parenthesis.

Setting	Description
Conditions format style	Specifies how conditions in clauses are formatted: CONDITIONS_WRAPPED - lines are not forced before conditions. CONDITIONS_STACKED_WITH_LEADING_OPERATORS - a new line is forced for each condition in a clause, with logical operators, if present for a line, displayed at the start of the line. CONDITIONS_STACKED_WITH_TRAILING_OPERATORS - a new line is forced for each condition in a clause, with logical operators, if present for a line, displayed at the end of the line.
Conditions stack threshold	For **Conditions format style** selections that specify stacking, this value specifies the minimum number of conditions that must appear in a clause before conditions are stacked.
THEN statements	If enabled, simple THEN clauses are kept to a single line.

ISQL OPTIONS - CODE ASSIST TAB

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **ISQL - Code Assist** tab. This tab lets you activate and select options for three coding assistance and error-detection features available from the ISQL editor:

- **Code Complete** - lets you insert or replace object names, selected from suggestion lists, as you edit a script. For more information, see [Code Complete](#).

- **Semantic validation** - detects object name references to objects not present in the datasource index. For more information, see [Semantic Validation](#).

- **Real-time syntax checking** - on-the-fly syntax checking is performed as you type. For more information, see [Syntax Checking](#).

The **ISQL - Code Assist** tab has the following settings:

Group/Option		Description
Code Complete	**Enable Code Complete**	Lets you enable or disable the Code Complete feature. When disabled, Code Complete cannot be invoked automatically or manually. Because of the overhead in maintaining internal resources used in implementing this feature, a slight performance increase can result from disabling the feature.
	Enable auto-activation and **Auto-activation delay**	When auto-activation is disabled, the Code Complete feature must be invoked manually. When auto-activation is enabled, the Code Complete feature is invoked automatically each time the interval between keystrokes exceeds the specified **Auto-activation delay**.
	Insert single proposals automatically	Specifies that if a Code Complete suggestion list would contain only a single suggestion, that suggestion is inserted automatically.
	Fully qualify completions automatically	Specifies that Code Complete results are returned fully qualified, rather than the minimum required to identify the object.

Group/Option		Description
Code Validation	**Enable Real-time syntax checking**	If enabled, syntax-checking is performed as you type. If disabled, syntax checks must be initiated manually.
	Severity levels for semantic validation problems	Lets you select the severity level (ERROR, WARNING, or IGNORE) associated with detected semantic errors. The WARNING option provides a contrast to syntax errors, which are always flagged with a severity level of ERROR.The IGNORE setting disables the feature. For more information, see [Semantic Validation](#).
	Perform syntax and semantic validation on files smaller than	Places an upper limit on the size of files for which automatic syntax checking and semantic validation are active.
Code Assist Resources	**Refresh**	Each time you open an ISQL editor window, Rapid SQL obtains schema information for the object types supported for the Code Complete and Semantic Validation features. That information is updated with any modifications that you make and persists until the window is closed. The **Refresh** button ensures that the information is current in case other users or applications are modifying the schema while the current SQL editor session window is open.
Restore defaults	Restores all settings on the tab to the original defaults.	

JAVA OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Java** tab.

Rapid SQL requires a JDBC connection. You have the ability, with this panel of the Options Editor, to set or change options that apply to the Java virtual machine (JVM) that is running on the client. The table below describes the options and functionality of the **Java** tab:

Option	Description	Default
JVM Options:		
Initial Heap Size	Set the size, in MB, for the repository where live and dead objects comingle with free memory. If memory runs out, JVM executions stop so the garbage can be collected and expunged. Each platform responds differently, so trial and error can help you maximize performance.	64 MB
Maximum Heap Size	Set the upper limit for your heap size.	64 MB
Maximum Perm Gen	Set the upper limit for the permanent generation size.	Version-specific
Additional Options	Add options here ONLY in consultation with Embarcadero Technical Support.	N/A

LOGGING OPTIONS

After opening the Options editor (see <u>Specifying Rapid SQL application and feature options</u>), you can make changes to the **Logging** tab. The Logging tab lets you set defaults that specify the behavior and placement of SQL Logging and Output Logging.

The table below describes the options and functionality on the **Logging** Tab:

Option	Description	Default
Log all SQL Statements to a File	Indicates that the application should log all of the SQL that it executes to a file. SQL logging provides an audit trail for Rapid SQL. You can examine this log to determine the SQL executed to complete a task.	Not selected
Logfile Path	If you choose to log the SQL generated by the application, specify the drive, directory, and file name.	None
Max File Size	Specifies the maximum size for the logfile. When the logfile reaches this threshold, it automatically starts deleting lines in the logfile (starting with the oldest statements) to remain within the specified size limit.	1024 KB
Truncate	Empties the entire contents of the logfile.	Not available
Log all Output Messages to a File	Indicates that the application should log all server messages sent to the Output window. This type of logging lets you monitor only messages issued by the server versus all SQL logged by the application. You can examine this log to determine all messages the server issued.	Not selected
Logfile Path	If you choose to log the server messages generated in the Output window, specify the drive, directory, and file name.	None
Max File Size	Specifies the maximum size for the output logfile. When the output logfile reaches this threshold, it automatically starts deleting lines in the file (starting with the oldest statements) to remain within the specified size limit.	1024 KB
Truncate	Empties the entire contents of the output logfile.	Not available

MYSQL UTILITIES OPTIONS

After opening the Options editor (see <u>Specifying Rapid SQL application and feature options</u>), you can make changes to the **MySQL Utilities** tab. Rapid SQL integrates with MySQL utilities. For Rapid SQL to access these utilities, you need to specify their location in the MySQL Utilities Tab of the Options Editor. You can use the MySQL Dump and Import Utilities. The table below describes the options and functionality on the **MySQL Utilities** Tab

Option	Description
mysqldump	Specifies a path for the MySQL dump utility, mysqldump.exe. By default, MySQL installs this utility in the C:\mysql\bin directory.
mysqlimport	Specifies a path for the MySQL import utility, mysqlimport.exe. By default, MySQL installs this utility in the C:\mysql\bin directory.

ORACLE UTILITIES OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Oracle Utilities** tab.

Rapid SQL integrates with multiple Oracle utilities. For Rapid SQL to access these utilities, you need to specify their location in the Oracle Utilities Tab of the Options Editor. You can use the Oracle Export, Import Utilities, and SQL * Loader. The table below describes the options and functionality on the **Oracle Utilities** Tab:

Option	Description
Export	Specifies a path for the Oracle Export utility. By default, Oracle installs to C:\Orant\Bin directory.
Import	Specifies a path for the Oracle Import utility. By default, Oracle installs to C:\Orant\Bin directory.
SQL*Loader	Specifies a path for the SQL * Loader utility. By default, Oracle installs to C:\Orant\Bin directory.
Default Bind Size	Specifies the bind size. Default is set to 70KB.

PERF CENTER OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Performance Center** tab. Rapid SQL lets you customize Performance Center's integration with Rapid SQL. The table below describes the options and functionality on the Perf Center Tab:

Option	Description	Default
Computer Name or IP Address	Specify where the Performance Center Server is installed.	localhost
Port Number	Specify the port for the Apache server or IIS web server.	80
Test	Verifies the settings.	
Connect to the server using	Specify if you want Rapid SQL to open the **Web Client** or the Performance Center file (**PerfCntr.exe**) within Rapid SQL.	Web Client

QUERY BUILDER OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the Query Builder tab. Option parameters set on the Options Editor elicit a prompt if there are different options set on an open individual session. Global options override properties set within individual Query Builder sessions.

The table below describes the options and functionality on the **Query Builder** Tab:

Interface Element	Option	Description	Default
Code Generation	**Generate Use Database statement**	Adds a line of SQL code indicating which database or instance is used in the statement.	Selected
	Generate owner names	Adds a line of SQL code showing the table owner name as part of the query.	Selected
	Include Row Count limits	Includes the output row limit set in the Execution settings.	Not selected
	Generate SQL/92 if supported by DBMS	SQL/92 is a standard for relational database management systems.	Not selected
Execution	To set the maximum number of rows in your result set, type the number in the dialog box. This lessens congestion of server processes when queries execute by setting row count limits.		1000 rows
General	**Show Column Data types in Query Diagram**	Reveals the data type in each column for tables in the SQL Diagram pane.	Not selected
	Confirm on Item delete	Opens a Confirm Delete dialog box when an item is deleted.	Selected
	Auto Populate Views	Checks syntax every time an execute statement, refresh or copy statement begins.	Not Selected
	Auto Format	Automatically sets style and spacing of display.	Selected
Auto Join	**Run Automatically**	Automatically detects names and data types, and create joins for multiple tables.	Selected
	Require Indexes	Joins only indexed columns. Requires an indexed column for joins.	Selected
	Require same data type	Automatically joins columns with the same data type.	Selected
Syntax Checker	**Automatic Syntax Check**	Automatically checks SELECT and CREATE VIEW statements for errors.	Selected
	Warn on non index join	Returns a warning when it detects a join against a non-indexed column, or a column not participating in a primary key	Not selected
Display	Lets you sets the style, size, and color of **Column Font** and **Title Font**. Also lets you set the background **Table Color** for the SQL Diagram Pane.		Available

REPORTS OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Reports** tab. The **Reports** tab lets you:

- Append a date and timestamp to report file names. This prevents existing reports from being overwritten.

- Save reports in date-specific folders.

- Save reports in PDF format as opposed to HTML.

• Let you include a provided image and title to reports.

RESULTS (ISQL) OPTIONS

After opening the Options editor (see <u>Specifying Rapid SQL application and feature options</u>), you can make changes to the **Results** tab. The table below describes the options and functionality on the **Results** Tab:

Interface Element	Options and descriptions	
Results Option pane	**Result Window**	**Single Window** - Displays all results in one tabbed result window. Multiple result sets together in the window. Single Window and Multiple Windows options are mutually exclusive.
		Multiple Windows - Displays multiple result sets one result set per window.
		Attached to Editor - Sets results as tabbed windows attached to the ISQL window. Used in conjunction with Single Window option or Multiple Window option. Attached and Unattached options are mutually exclusive.
		Unattached - Sets results appear separate from the ISQL Window. Used in conjunction with Single Window option or Multiple Windows option.
		Reuse Window - Sets new result sets to overwrite any current result sets in an open Result Window. Only valid for Single and Attached to Editor combination.
	Results File	**Mail File Type** - Selects the file type to use when sending result sets via a MAPI-compliant mail package. Valid formats include the proprietary Results type, Tab delimited, Comma separated, and HTML.
		Schedule File Type - Selects the schedule file type. Valid formats include Tab delimited, Comma separated, and HTML.
		Include column titles when saving - Includes column titles when saving a result set. If this option is turned off, column titles Rapid SQL does not save result sets.
	Result Set Options	**Default Rowcount** - Lets you limit the number of rows returned to the result window of the ISQL window (default 0).
		Sybase and SQL Server: Text Size (bytes) - Lets you specify the text size (default 8192).
		Oracle:LONG Size (bytes) - Lets you specify the LONG size (default 8192).
		LOB Preview: Text Size (bytes) - Specifies the length of the preview of LOB column data (default 4096).
	Sort Type	Lets you select a result sorting: **Alphameric** or **Lexicographic**.
Format pane	**Column Formatting**	**Auto Format (Best Fit)** - Sets column widths automatically to accommodate the longest piece of data in a column. Large queries depend on the longest row formatting, so activating this option can affect performance.
		Use pre-defined column - Lets you select column type and character length
	Enable Date/ Time Format	Lets you select the date/time format.

Interface Element	Options and descriptions	
	Format	**Standard Grid** - Displays all result sets in a standard grid format. Rapid SQL displays result sets in grid format in IISQL Editors that are opened after you have selected this option. Rapid SQL does not display IISQL Editors that are already open.
		HTML - Displays all result sets as HTML tables. Rapid SQL displays result sets in HTML format in ISQL Editors that are opened after you have selected this option. Rapid SQL does not display ISQL Editors that are already open.
		ASCII Text - Displays all result sets as ASCII Text. Rapid SQL displays result sets in ASCII Text format in ISQL Editors that are opened after you have selected this option. Rapid SQL does not display ISQL Editors that are already open.
		Grid Font and **Printer Font** buttons - Open a Font dialog box, letting you select the font, style, and size for the result sets grid or printed result sets.
		Enable Locale - If selected, numbers are formatted (thousands separator, decimal separator) as locale-specific strings. If unselected, numbers are formatted according to the **C** locale.

SMTP MAIL OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **SMTP Mail** tab. The SMTP Mail Tab of the Options Editor lets you specify outgoing notification e-mail message options.

The table below describes the options and functionality on the **SMTP Mail** tab:

Option	Description	Default
Send messages through SMTP	Enables SMTP messaging and makes the other controls on this tab available.	Unselected
Name	Name that appear as the e-mail sender.	Name you specified during installation.
E-mail Address	Address to send e-mails from Rapid SQL.	E-mail address you specified during installation.
Authentication	Lets you specify authentication options.	None
User Name	User name for authentication.	Not available
Password	Password for authentication.	Not available
Host Name	SMTP server for outgoing messages. For Microsoft Outlook, select Tools, and then Accounts. On the Mail Tab, select target account, and then click Properties. On Servers Tab, copy the Outgoing Mail(SMPT) and paste.	Host Name you specified during installation.
Port Number	Port number you connect to on your outgoing SMTP server.	25
Test	Opens an SMTP Configuration Test e-mail addressed to your e-mail address. Click Send Mail to send the e-mail.	Available
Bind to	Your IP address the message is bound to.	ANY_IP_ADDRESS

Option	Description	Default
Encoding	E-mail encoding.	Western Europe (ISO)
Send messages Mime encoded	Messages encoded using Multipurpose Internet Mail Extensions (MIME) support enriched content and attachments.	Selected
Send all messages as HTML	Messages include text formatting.	Selected
Auto Connect to the Internet	Rapid SQL connects to internet at launch.	Selected

VERSION CONTROL OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Version Control** tab. The **Version Control** Tab of the Options Editor lets you select which version control system you want Rapid SQL to use as the underlying version control system.The table below describes Version Control options:

Item	Description	Default
Specify Version Control System	Lists the Version Control systems that integrate with Rapid SQL. Select the option button for the product you want Rapid SQL to use.	None
Optional Fields	If you select Microsoft Visual SourceSafe, Rapid SQL uses your default SourceSafe database unless you override this setting by entering the path for the SourceSafe file associated with the desired database. To use a SourceSafe database other than the default, provide a user name, password, and the location of a scrsafe.ini file.	Optional

WARNINGS OPTIONS

After opening the Options editor (see [Specifying Rapid SQL application and feature options](#)), you can make changes to the **Warnings** tab. Rapid SQL lets you issue warning messages to users whenever improper operations are attempted in a database. Warning messages differ by platform.

The tables below describes the options of the **Warnings** Tab:

Option	Description	Platform
Create an index on the same tablespace as the associated table	Issues a warning message whenever a user does not create an index on a different tablespace than the table. This makes it possible for the server to read the index and table data in parallel.	DB2
Create an index on the same tablespace as the associated table	Issues a warning message whenever a user does not create an index on a different tablespace than the table. This makes it possible for the server to read the index and table data in parallel.	Oracle
Create an object in the SYSTEM tablespace	Issues a warning message whenever a user tries to create or place an object on the SYSTEM tablespace.	Oracle

Option	Description	Platform
Create a user with default or temp tablespace as the SYSTEM tablespace	Issues a warning message when a user is created with a default or temp tablespace on the SYSTEM tablespace.	Oracle
Create an object in the master database	Issues a warning message when an object is created in the master database.	Microsoft SQL Server and Sybase ASE
Create a table or index on the default segment	Issues a warning message when a table or index is created on the default segment.	Microsoft SQL Server and Sybase ASE
Create a non-clustered index on same segment as the associated table	Issues a warning message when a non-clustered index is created on the same segment as the associated table.	Microsoft SQL Server and Sybase ASE

DATASOURCE MANAGEMENT

After installing Rapid SQL, you must set up [datasources](#) to establish reusable connections to your database servers. A datasource is a database connection profile that is similar to the connection information that you have stored in your SQL.INI or WIN.INI files. Rapid SQL stores information about the datasource specification in the system registry and provides a visual interface for maintaining it.

The Datasource Registration Wizard guides you through the required steps to establish a connection to your server and makes the process of setting up datasources, easier. For details, see [Registering or editing datasources](#).

This section describes the function of datasources, the process of establishing datasource connections and managing your datasources and datasource groups.

DATASOURCES

A datasource is a database connection profile. A datasource includes:

- Name
- Connection String
- Default User ID
- Optional Password Specification

All database operations are performed through datasources. You can create a datasource profile for each database instance (or database server) in your enterprise, and you can create multiple datasource profiles with different logins for a single database. The table below describes the data items stored for each datasource:

Data Item	Description
Name	A unique, user-defined name for the datasource.
Connection String	For Oracle: the SQL*Net connect string, for Sybase ASE: the database server name.
Default User	Default user ID to use when logging in to the datasource. Can be null.
Default Password	Default password to use when logging in to the datasource. This is encrypted. Can be null.
Auto-Connect Flag	If Yes, then automatically login using default user and password. If No, open Login dialog box.
Default Database	SYBASE ASE ONLY: Database to automatically use after logging in.

Available Functionality

Rapid SQL offers the following functionality for the Datasource Explorer:

- [Change Group](#)

- [Connect](#)

- [Disconnect](#)

- [Drop](#)

- [Registering or editing datasources](#)

- [New LUW Datasource](#)

- [Register Datasource](#)

- [Unregister Datasource](#)

Related Topics

- [Selecting Datasources](#)

- [Viewing Datasource Properties](#)

UNDERSTANDING THE DATASOURCE CATALOG

A Datasource Catalog is a collection of defined datasources. On startup, Rapid SQL loads a Datasource Catalog using one of three methods, specified using the Options editor:

- **Windows Registry datasource catalog** - The datasource catalog is stored and managed locally, in the Windows registry. This user can add and delete datasources and edit the registration details of existing resources. Any changes are retained across startups of Rapid SQL.

  This method is compatible with other Embarcadero products using registry-based datasource definition storage. If more than one of those products is installed on the same machine, they can share the same Datasource Catalog.

- **File based datasource catalog** - The datasource catalog is stored locally and is file-based. Each datasource definition included in the catalog is stored in a single file (`.dsd` file suffix) in the *%APPDATA%***Embarcadero\Data Sources** folder, in a default installation. This user can add and delete datasources and edit the registration details of existing resources. Any changes are retained across startups of Rapid SQL.

  This method is compatible with other Embarcadero products using file-based datasource definition storage. If more than one of those products is installed on the same machine with Rapid SQL, all can share the same Datasource Catalog.

- **Network shared datasource catalog** - The datasource catalog is file-based, and obtained from a user-specified location such as a network share. At each startup, Rapid SQL imports definitions from the network shared file (`.etsds` file suffix).

  Changes such as new datasources, deleted datasources or edited datasource registrations are lost as soon as this user shuts down Rapid SQL. This method allows centralized maintenance of the Datasource Catalog and sharing of a single catalog among multiple users across a network.

  NOTE: The definition files used in both the **File based datasource catalog** and **Network shared datasource catalog** methods do not store user ID and password credentials. Credentials, links to definition files in the current user's catalog, and folder structure are stored in the *%APPDATA%***Embarcadero\Data Sources\metadata\dsuri.xml** file.

For information on choosing a method and specifying the network location of a Shared Datasource Management definition file, see Datasource Options.

Regardless of the catalog storage method, the Datasource Catalog can be built manually by explicitly registering datasources or using automated methods such as the Discover Datasources feature. For details, see Automatically Discovering Datasources and Registering or editing datasources. Keep in mind that changes made with **Network shared datasource catalog** in effect, are lost on shutdown of Rapid SQL.

In order to set up a **Network shared datasource catalog** scheme, you can build a catalog first using the **File based datasource catalog** or **Windows Registry datasource catalog** method. You can subsequently use the **Manage Datasources** facility to export your **Network shared datasource catalog** file to the location that others will use to load the file. For details, see Importing and Exporting Datasource Definitions.

More generally, the **Manage Datasources** facility's import/export functions can be used to exchange datasource definitions between users employing the different storage methods. Even more simply, these functions save time in allowing datasource definitions to be defined once and then shared with multiple users.

AUTOMATICALLY DISCOVERING DATASOURCES

The first time you run Rapid SQL a dialog box displays, giving you the option to Auto-Discover all configured datasources. If you click Yes, the Rapid SQL Auto-Discover feature searches the DBMS configuration files on your computer and automatically discovers all the datasources that you are licensed for. For example, if you have a cross-platform license, Discover Datasources finds all unregistered datasources. If you have an Oracle only license, Discover Datasources finds all unregistered Oracle datasources.

> **NOTE:** Microsoft SQL Server datasources are registered through a Windows NT system call to your network. Provide login information (user name and password) the first time you <u>connect</u> to a datasource.

> **NOTE:** IBM DB2 LUW for Linux, Unix, and Windows databases use ODBC/CLI or DB2 (attach) to connect. Therefore, you need the proper ODBC/CLI Connection established in order for the auto-discover feature to find your IBM DB2 LUW for Linux, Unix, and Windows databases, including registering the DB2 datasource to ODBC as a system datasource. Although your datasources are auto-discovered, provide login information (user name and password) the first time you connect to a datasource.

In addition to Auto-Discovering your database servers, the application creates Datasource Groups based on RDBMS type. Each registered datasource is placed in its respective Datasource Group. For example, all Microsoft SQL Server datasources are added to the Microsoft SQL Server Group. Each registered datasource is placed in its respective Datasource Group.

For more information on how to configure your datasources, see <u>Working with Datasources.</u>

CHANGING DATASOURCE GROUPS

Rapid SQL lets you change datasource groups by:

- Dragging the datasource between groups.
- Invoking the Change Group dialog box.

Dragging and Dropping Between Groups

1   On the Datasource Explorer, left-click the datasource group you want to move, drag it over the folder for the new group, and release the pointer.

Using the Change Group Dialog Box

Disconnect your datasource before changing groups.

1   On the Datasource Explorer, right-click the datasource you want to move, and then click *Change Group*.

    Rapid SQL opens the Change Group dialog box.

2   In the *Select Group tree*, click new group.

3   Click OK.

Rapid SQL changes groups.

For more information, see <u>Datasource Groups.</u>

CONNECT

Rapid SQL lets you set datasources to <u>automatically connect</u> each time you open the application.The first time you start the application, Rapid SQL prompts you to register your datasources. During this process, you can select the Auto Connect check box, which automatically connects all registered datasource each subsequent time you open the application.

If you did not check the Auto Connect box, or if you clicked No when prompted to connect to a database after registering, you must connect manually, each time you want to access that datasource, using the <u>Datasource Login dialog box</u>. If you later want to automatically connect your datasources, edit the datasource accordingly. See <u>Registering or editing datasources on page 130</u>.

The table below describes the options and functionality on the Datasource Login dialog box:

Option	Description
Login ID	Lets you type the Login ID.
Password	Lets you type the password.
Auto Connect	Select to automatically connect to the datasource in the future, select the check box.

For more information, see <u>Completing the Datasource Login Dialog Box</u>.

COMPLETING THE DATASOURCE LOGIN DIALOG BOX

To complete the Datasource Login dialog box, do the following:

1   On the *Datasource* menu, click Connect.

    OR

    On the *Datasource* tool bar, click Connect.

    OR

    On the *Datasource Explorer* tool bar, click Connect.

    OR

    On the *Datasource Explorer*, right-click the datasource, and then click Connect.

    OR

    On the *Datasource Explorer*, double-click the datasource.

    Rapid SQL opens the Datasource Login dialog box.

2   In Login ID type the Login ID.

3   In Password type the password.

4   Select Auto Connect to automatically connect to the datasource in the future.

5   Click Connect.

    Rapid SQL opens the Datasource Landing Page in the right pane of the application.

For more information, see [Working with Datasources](#).

DISCONNECT

When you disconnect from a server, the application immediately breaks the connection between any open ISQL Windows, the servers, and databases. Although your ISQL Windows are still visible, the connections are no longer valid. If you attempt to execute a script, Rapid SQL attempts to reconnect to a registered datasource, if available.

For more information, see [Completing the Disconnect Dialog Box](#).

COMPLETING THE DISCONNECT DIALOG BOX

To complete the Datasource Login dialog box, do the following:

1 On the *Datasource* menu, click Disconnect.

    OR

    On the *Datasource* tool bar, click Disconnect.

    OR

    On the *Datasource Explorer* tool bar, click Disconnect.

    In the right pane of the *Datasource Explorer* window, right-click the datasource, and then click Disconnect.

    Rapid SQL opens a dialog box asking if you want to commit all pending transactions for that connection or to rollback all before disconnecting. You cannot disconnect if there is an uncommitted transaction.

2 Click Yes.

    Rapid SQL confirms you want to disconnect and closes the dialog box.

DISCOVER DATASOURCE

Rapid SQL discovers datasources residing on your system that are not currently registered datasources through a Windows NT system call to your network. The [Discover Datasource dialog box](#) includes a list, which includes the name of the server or instance and the type of DBMS of all unregistered datasources found on your network or local machine. Once discovered, you have the option to register datasources.

Completing the Discover Datasources Dialog Box

1 On the *Datasource* menu, click *Discover Datasource*.

    Rapid SQL opens the Discover Datasources dialog box.

2 Select the *check box* next to the datasource you want to register.

3 Click Select All to select all the datasources on the list.

4 Click Register.

    `Rapid SQL registers` the datasource or datasources selected.

5 Click OK.

    Rapid SQL closes the Rapid SQL message.

For more information, see:

[Datasources](#)

MANAGING DATASOURCES

The Manage Datasources dialog box lets you manage datasources throughout your enterprise from a single vantage point. It provides concise, relevant information on your datasources in a simple grid format. It provides centralized access to common datasource operations such as adding, modifying, deleting, discovering, importing, and exporting datasources.

To open the Manage Datasources dialog box:

1   On the *Datasource* menu, click *Manage Datasources.*

Rapid SQL opens the Manage Datasources dialog box.
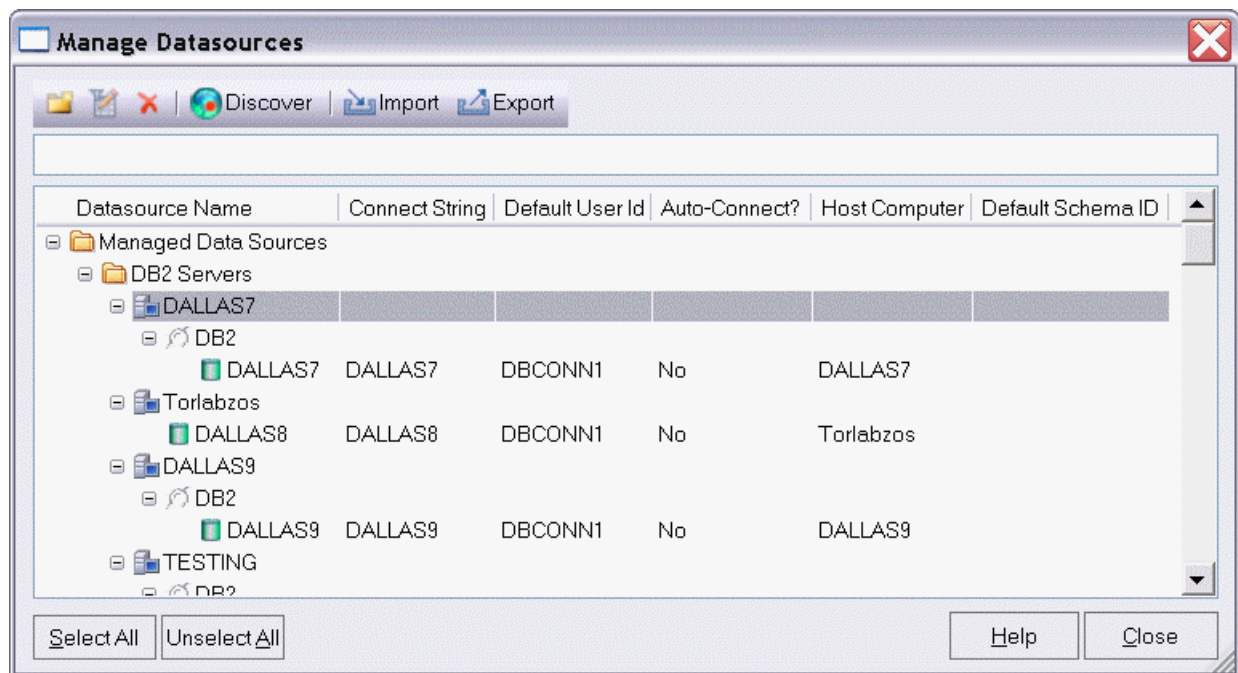
The grid format lets you access and view datasource specifications. datasources are grouped according to the default, DBMS-based folder structure and then within any custom datasource groups you have set up. The table below describes information available for each datasource entry:

Column	Description
Datasource Name	Uses an explorer-like interface to display all registered datasources and their groups. You can navigate this column in the same manner as the datasource explorer, by clicking on nodes to expand or collapse your view of the datasources.
Connect String	Displays the full connection string for the datasource.
Default User Id	Displays the Default User ID for the datasource.

Column	Description
Auto-Connect?	Indicates whether the Auto Connect feature is turned on or off.
Host Computer	Displays the name of the Host Computer if one has been configured.
Default Schema	Displays view default schemas for your DB2 datasources.

In addition, you can take the following actions:

- Click the New Datasource button to register a datasource. For more information, see [Registering or editing datasources](#).

- Select a datasource and click the Edit Datasource button to edit that datasource definition. For more information, see [Registering or editing datasources](#).

- Select a datasource and click the Remove Datasource button to unregister that datasource definition. For more information, see [Unregistering Datasource](#).

- Click the **Discover** button to locate all configured datasources on your network and automate the process of registering them. For more information, see [Automatically Discovering Datasources](#).

- Import or export datasource definitions. For details, see [Importing and Exporting Datasource Definitions](#).

IMPORTING AND EXPORTING DATASOURCE DEFINITIONS

On startup, Rapid SQL loads a Datasource Catalog, a collection of datasource definitions that the user can work against. How the catalog is stored and whether changes to the catalog are temporary or permanent, depends on the datasource storage method used. For an introduction, see [Understanding the Datasource Catalog](#).

Regardless of the storage method used, Rapid SQL lets you add datasources to the currently loaded catalog by importing datasource definitions from the following sources:

- Windows Registry - if you previously used the **Windows Registry datasource catalog** storage method but are now using one of the file-based methods

- **File based datasource catalog** storage method files (`.dsd` file suffix)

- **Network shared datasource catalog** storage method files (`.etsds` file suffix)

Similarly, you can export the currently loaded Datasource Catalog definitions to **File based datasource catalog** or **Network shared datasource catalog** storage method files.

The ability to export and import datasource definitions provides a number of practical advantages. For example:

- Simple sharing among users - if you have a large number of datasources to register, you can have one user walk through the process of registering each datasource and then export those definitions. Other users can then import the resulting datasource definition files.

- Sharing definitions among users employing different datasource storage methods - datasource definitions or entire catalogs created using the **Windows Registry datasource catalog** or **File based datasource catalog** methods can be exported for use by users employing any of the three methods.

To export one or more datasource definitions:

1   On the *Datasource* menu, click *Manage Datasources*.

    Rapid SQL opens the **Manage Datasources** dialog box.

2   Select a single datasource, multiple datasources, or a default or custom datasource group folder.

    NOTE: Datasource group folder information is not exported. Datasources are exported as individual files with no higher level grouping.

3   Click **Export**. An **Export Data Sources Folder** dialog opens.

4   Take one of the following actions:

    - To export datasource definitions to individual files that can be imported on a datasource-by-datasource basis, select **Export Data Sources to Individual Data Source Files** and use the associated control to locate and select a target folder.

    - To export datasource definitions to a single file for use by installations employing the Shared Datasource Managed catalog storage method, select **Export Data Sources To a Shared Data Source Management File** and use the associated control to locate and select a target folder and provide a file name. For information on how to designate a user as a Shared Datasource Managed catalog storage method, see [Datasource Options](#).

5   Click **OK**.

    NOTE: User ID and password credentials are not exported to the datasource definition file.

After export, other users can then import the datasource definitions.

To add one or more datasource definitions to a catalog:

1   On the *Datasource* menu, click *Manage Datasources*.

    Rapid SQL opens the **Manage Datasources** dialog box.

2   Select the datasource group folder where the files or files are to be imported.

3   Click **Import**. An **Import Data Sources** dialog opens.

4   Take one of the following actions:

- If you previous used the Windows Registry catalog storage method but are currently using the Locally Managed or Shared Datasource Managed method, select **Import Previously Registered Embarcadero Data Sources From Registry** to load any datasources registered when you were using the Windows Registry catalog method.

- To import datasource definitions from individual datasource definition files (`.dsd` file suffix), select **Import Previously Registered Embarcadero Data Sources From File(s)**, click **Add**, and use the **Select datasource definitions to import** dialog to locate and select the files to be added.

- To import datasource definitions from a single Shared Datasource Managed storage method file (`.etsds` file suffix) file, select **Import Data Sources From a Shared Datasource Management File** and use the associated control to locate and select the target file.

5   Click **OK**.

After importing a datasource definition file, you can either edit the datasource registration to provide security parameters or provide credentials when connecting to the datasource.

MANAGING DATASOURCE PROPERTIES

The [Datasource Properties dialog box](#) displays the name, type, version, status and mode of the datasource. The Datasource Properties box also lets you view the middleware or connectivity software that is being used to establish a particular datasource connection. You can use this information to troubleshoot connectivity problems, determining vital information such as the server version, connectivity library used, and library version and date.

Completing the Datasource Properties Dialog Box

1   On the Datasource Explorer, click a datasource with an established connection.

2   On the *Datasource* menu, click *Properties*.

Rapid SQL opens the Datasource Properties dialog box.

For more information, see:

[Datasources](#)

[Working with Datasources](#)

REGISTERING OR EDITING DATASOURCES

Each database instance must be registered. Whether you are registering a new data source, registering a new data source based on an existing data source definition, or editing preexisting connection information, you use the Datasource Registration dialog box.

To register a new data source

- On the Explorer, right-click **Managed Datasources**, and select **Register Datasource** from the context menu.

To register a new data source based on an existing definition

- In the left pane of the explorer, select the existing data source and then from the **Datasource** menu, select **Register Like**.

To edit registration information

- On the **Explorer**, right-click a data source, and select **Edit Datasource Registration** from context menu.

The table below describe the fields you will encounter when registering a data source or editing the connectivity information:

Panel/Tab	Option	Description
Database Type	Select the radio box corresponding to the DBMS type for the data source. If there is currently a DBMS node selected in the Explorer, that database type is automatically selected.	
Connection Information	For detailed descriptions of the options on this tab, see <u>Data Source connection information on page 133</u>.	
Advanced (only available for InterBase/ Firebird datasources after clicking the **Advanced** button on the **Connection Information** tab)	**JDBC Driver**	You should see the JDBC Driver that's appropriate to the platform. If you select **...**, the JDBC Driver Editor opens letting you browse to a new or possibly undetected JDBC driver. For details, see <u>JDBC Driver Editor</u>.
	Connection URL	The connection URL cannot be edited--it's for reference only.
	URL Check	Indicates whether the connection URL is valid or not.

Panel/Tab	Option	Description
Security Parameters	**User ID**	The User ID that Rapid SQL will use to connect to the data source.
	Password	The password associated with the User ID.
	Connect As: (Oracle)	When relevant, choose the appropriate user/administrator level.
	Domain:	For MS SQL only. Identify the domain if the user has restricted access.
	Auto connect?	Spares the user from reentering the password every time you connect.
	Connect using Windows Authentication or **Connect using OS Authentication** (IBM DB2 for Windows, Unix, and Linux, MySQL, Oracle, SQL Server)	Login to the server is verified using Windows/OS authentication
	Connect using Kerberos Authentication (Sybase)	If this option is selected, login to the server is verified using Kerberos authentication. **NOTE**: This is only available of the **Use alias information in the SQL.INI file/Sybase Server** option on the **Connection Information** tab is selected. For details, see [Data Source connection information](#).
	Test Connection	If the connection fails, read the error message and backtrack as necessary.
Datasource Properties	**#Include Search Directories**	Enter one or more paths on this datasource, which will be searched for files in conjunction with use of the **#include** directive in the ISQL editor, Procedure Object Editor, or Package Body Object Editor. Separate multiple paths using semicolons. For example: `c:\myscripts;c:\Program Files\Scripts` For more information on use of the #incliude directiive, see [Preprocessing #define and #include Directives](#). Note that if there are no entries specified here, the directory specified on the **Directories** tab of the Options editor will be searched. For more informtion, see [Directories Options](#).
	Database device default path	The database device default path is used when creating a new database device.

Panel/Tab	Option	Description
Oracle View Config (Oracle only)	Lets you manually configure Rapid SQL access to Oracle Data Dictionary DBA or ALL/USER views to conform to the view privileges associated with a specific role.	
	NOTE: In order to enable manual configuration of Oracle Data Dictionary views for Oracle datasources and make this panel available, you must select **Use Datasource Configuration for Views** in the Options Editor (**File > Options > General > Oracle**). For more information, see [General Options]().	
	Only DBA views with an ALL/USER equivalent are offered on this panel. You can use the following methods to select DBA or ALL/USER type views on a view-by-view basis:	
	Manual selection using the **Use DBA View** check boxes	Selecting the **Use DBA View** check box associated with an individual view makes the DBA group version of that view available to Rapid SQL. Deselecting the **Use DBA View** check box associated with an individual view makes the ALL/USER group version of that view available to Rapid SQL.
	Reset to All	Deselects all **Use DBA View** check boxes, making only ALL views accessible to Rapid SQL.
	Reset to DBA	Selects all **Use DBA View** check boxes, making DBA views accessible to Rapid SQL.
	Resolve Automatically (only available when editing a connected datasource)	Queries the datasource and selects the **Use DBA View** check box for each DBA view to which the current user has access.
	Changes take effect as soon as you click the **Finish** button.	
Datasource Group	**Select the datasource group folder**	Select the folder that is to contain this datasource.
	Select a category	If you want to use category-based user interface item color-coding and labelling for this datasource, this dropdown lets you select an existing category to assign this datasource to that category. You can also select **Customize** to manage your categories. For details, see [Customizing Datasource Categories]().
		For an introduction to datasource categorization and related tasks, see [Categorizing datasources using color-coding and labelling]().

After clicking **Finish**, you are prompted to connect to the newly registered data source.

   NOTE: On initial connection to a Sybase Cluster edition data source, you are offered the choice to connect to the entire cluster. If you select **NO**, your connection scope is an instance. This choice sets the default connection scope for this data source. You can subsequently change the connection scope for the data source by editing the data source.

   NOTE: After you successfully connect to a data source, you can see the name of the host and the connection string information displayed at the bottom of the Rapid SQL window. For Oracle and Sybase ASE, should a question arise, you can see if perhaps you set up the data source NOT to use the alias file. You can also use these cues to remind yourself that if you changed the alias file, that change will not be reflected here unless you very specifically made a change to the defaults in the Options Editor-JAVA tab.

DATA SOURCE CONNECTION INFORMATION

The following table lists the **Connection Information** tab options, DBMS-by-DBMS, available when registering or editing a data source.

DBMS	Option	Description
IBM DB2	**Server**	The name of the server.
	Schema ID	This is an optional field. If you want to include the current schema as part of the data source connection properties, you can set the implicit schema for unqualified object references to a schema different from the user's login id.
	Function Path	This is an optional field. If you choose, you can enter the appropriate function path name so the search scope of functions will not be limited to the IBM DB2 library.
	Datasource Name:	This field is automatically populated with the server name, but you can rename it to whatever you want.
Interbase/Firebird	**Host name**	The name of the machine where the database resides.
	Port	Optionally, provide a port number.
	Database file path	The full path to the database file, a .gdb file for Interbase or a .fdb file for Firebird.
	Datasource Name	Optionally, override any value in this field with a value to be displayed in the Rapid SQL Datasource Explorer tree.
ODBC	**Connect String**	A valid ODBC connect string.
	Datasource Name	The name that will be displayed in the Rapid SQL Datasource Explorer tree.
SQL Server	**Use Network Library Configuration**	When you select this, you can connect to a Net-Library that is listening on the server that is configured to either the Named Pipes or TCP/IP protocols.
	Alias	This box is enabled only when you opt to use the Network Library Configuration
	Protocol: TCP/IP or Named Pipes	You need to select one or the other to register a SQL Server database.
	Host	Enabled only when you have not selected an Alias from the Network Library.
	Port (optional)/ Pipe Name	Depending on your section, you can optionally indicate the port for TCP/IP or the pipe name depending on your means of connection.
	Default DB (optional)	Optionally give the default name for the database.
	Datasource Name	The field is automatically populated with the host name, but you can rename to whatever you want.
MySQL	**Server**	Enter the name of the host, for example doctest01.
	Port (optional)	The default port is 3306. You do not need to enter this information.
	Default Database	You must enter the name of the default database.
	Datasource Name	This field is automatically populated with the server name, but you can rename it to whatever you want.

DBMS	Option	Description
Oracle	**Use TNS Names Alias/Oracle Alias**	If this option is checked, look for the Alias you want to register from the drop-down list. The remaining fields will automatically populate. If it is unchecked, you must manually enter the requisite information.
	Host	Manually enter the name of the host machine.
	Port	The default is 1521, but you can change it to wherever the Oracle listener is set up.
	SID/Service Name / Type: SERVICE_NAME or SID	Enter the SID or Service Name to correspond with the option you select.
	Instance Name	The specific name used to identify the Oracle instance (the SGA and the Oracle processes).
	Datasource Name	The field is automatically populated with the host name, but you can change it.
Sybase	**Use alias information in the SQL.INI file/Sybase Server**	If this option is selected, choose a name from the drop-down list in the Sybase Server field.
	Use SSL encryption	If this option is selected, the JDBC connection will be established using SSL encryption
	Host	If you selected the Alias option, this field automatically populates. Otherwise, you need to manually enter the name of the host.
	Port	There is no default for this optional field.
	Default DB (optional)	Optionally give the default name for the database.
	Datasource	This field automatically populates with the server name. You can change it to whatever you want.
	Connection scope (Cluster Edition only)	Lets you specify a connection to the entire cluster or to a single instance.

For context information, see [Registering or editing datasources](#).

JDBC DRIVER EDITOR

The following table lists the controls available on the JDBC Driver Editor dialog. You use this dialog when defining or editing a data source.

Required Information	Description
Driver Type	The database platform.
Driver List	You see what platform-specific drivers are available to you and which driver (highlighted) is currently associated with the server you are connecting to.
Name	The name of the driver.
Class	The name of the class the driver uses for creating connections to the server.

Required Information	Description
Version	The driver's version.
Driver supports the use of native aliases	Rapid SQL cannot detect whether the driver you are using can support native aliases. You need to know by checking the driver's documentation.
Required jars	Rapid SQL detects the required jars and lists the path to them in the little window. You can select or add new jar files as required by your driver.

CUSTOMIZING DATASOURCE CATEGORIES

Rapid SQL lets you customize your datasource category scheme. A datasource category has the following configurable components:

- **Category name** - The name displayed as a selection in the **Select a category** dropdown

- **Short Name** - The abbreviation shown in the Datasource toolbar dropdown when a categorized datasource is selected.

- **Color** - The color used to denote a categorized datasource in Datasource Explorer tree icons and window tabs.

  NOTE: Display of specific user items used to denote categories is configurable. For more information, see Datasource Options.

To customize your datasource categories

1   On the **Datasource** menu select **Register Datasource** and then select the **Datasource Group** panel.

2   From the **Select a category** dropdown select **Customize**. The **Manage Datasource Categories** dialogs opens.

3   Take one of the following actions:

- Create a new category by clicking **New** and selecting or providing a name, short name and color combination.

- Edit an existing category by selecting the category, clicking **Edit** and modifying the name and color combination.

  NOTE: The short name for a category cannot be edited.

- Delete an existing category by selecting the category, clicking **Delete** and verifying the deletion at the prompt.

SELECTING DATASOURCES

The Rapid SQL Select Datasource dialog box lets you select a datasource and connect to it.

Completing the Select Datasource Dialog Box

1　On the *Datasource* menu, click *Select* to open the *Select Datasource* dialog box.

2　Click the *Datasource* list box, and then click the target datasource.

3　Click Connect to connect to the datasource.

4　Click the *Database* list, and then click the target database.

5　Click OK to close the *Select Datasource* dialog box.

For more information, see:

[Datasources](#)

[Working with Datasources](#)

UNREGISTERING DATASOURCE

Rapid SQL lets you unregister datasources when you no longer need them.

> **TIP:**　Removing a datasource from Rapid SQL does not delete the physical database. It simply removes the datasource definition, and connection information, from the Rapid SQL catalog.

1　On the *Datasource* menu, click Unregister.

　　OR

　　On the *Registration* tool bar, click Unregister.

　　OR

　　On the *Datasource Explorer*, right-click the datasource, and then click Unregister Datasource.

　　Rapid SQL opens a dialog box.

2　Click Yes.

　　`Rapid SQL confirms` you want to unregister the datasource.

> **NOTE:**　The datasource manager is shared across Embarcadero's database management products. When you remove a datasource in any of Embarcadero's database management tools the datasource is removed across all relevant products.

For more information, see:

[Datasources](#)

[Working with Datasources](#)

DATASOURCE GROUPS

Rapid SQL lets you define datasource groups to organize the datasources in your enterprise. Datasource Groups behave as folders in the Windows Explorer, allowing you to group related datasources together. If you manage or monitor many Microsoft SQL Servers, datasource groups are a great mechanism for alleviating desktop clutter.

Anywhere that datasources are presented in a hierarchical tree format, datasource group folders expand to display one or more contained datasources. Upon installation of the first Embarcadero database management product, an initial datasource group is called Managed Datasources. You can rename this group.

Available Functionality

Rapid SQL offers the following functionality for Datasource Groups:

- Removing a Datasource Group

- New Datasource Group

- Rename Datasource Group

REMOVING A DATASOURCE GROUP

Rapid SQL lets you remove database groups that you no longer need to access, or that have become obsolete. Keep the following in mind when removing datasource groups:

- If a datasource group contains datasources or other datasource groups, those items are deleted when you delete the containining group.

- The default datasource groups correspond to the DBMS platforms supported by Rapid SQL. If you delete a default group, it will be recreated if you subsequently register a datasource for the associated platform.

To remove a datasource group

1   Right-click the datasource, and then click Remove. Rapid SQL prompts you to verify.

2   Click *Yes*.

For more information, see Datasource Groups

NEW DATASOURCE GROUP

Rapid SQL lets you define datasource groups to organize the datasources in your enterprise. Datasource Groups behave as folders in the Windows Explorer, allowing you to group related datasources together. If you manage or monitor many Microsoft SQL Servers, datasource groups are a great mechanism for alleviating desktop clutter.

Anywhere that datasources are presented in a hierarchical tree format, datasource group folders expand to display one or more contained datasources. Upon installation of the first Embarcadero database management product, an initial datasource group is called Managed Datasources. You can rename this group.

Completing the New Datasource Group Dialog Box

1　On the *Datasource Explorer*, right-click the datasource group folder, and then click New.

　　Rapid SQL opens the New Datasource Group dialog box.

2　In the *Datasource Group Name* box, type the new name of the datasource group.

3　Click OK.

　　Rapid SQL closes the New Datasource Group dialog box.

RENAME DATASOURCE GROUP

Rapid SQL lets you:

- Rename a datasource group.
- Change the members of a datasource group.
- Change the group to which a datasource belongs.

Completing the Rename Datasource Group

1　On the *Datasource Explorer*, right-click the datasource group folder, and then click *Rename*.

　　Rapid SQL opens the Rename Datasource Group.

2　In the *Datasource Group Name* box, type the new name of the datasource group.

3　Click OK.

　　Rapid SQL closes the Rename Datasource Group dialog box.

4   On the *Datasource* menu, click Shutdown.On the *Datasource* menu, click Configure.

Option	Description
Normal	Shuts down the server in an orderly fashion. disables logins, and waits for currently executing Transact-SQL statements and stored procedures to finish.
Immediate	Shuts down the server immediately. does not perform checkpoints in every database. The server terminates all user processes and rolls back any active transactions.

OR

On the Utilities tool bar, click Configure to open the Server Configuration dialog box.

OR

On the Datasource Explorer tool bar, click Configure to open the Server Configuration dialog box.

Option	Description
Edit Button	Click to modify the target parameter. Opens the Edit Configuration dialog box.
New Value	Lets you type the value for the parameter.

Option	Description
New Value	Lets you type the value for the parameter.

On the Datasource menu, click Session Recording.

OR

On the *Main* tool bar, click Session Recording.On the *Datasources* menu, click Session

Option	Description
Session Name	Lets you type the name of the session.
Session File	Lets you type the location and session file name or click the browse button. uses *.ses file extension for session files.

Recording.

OR

On the *Main* tool bar, click Session Recording. On the *File* menu, click Open.

OR

On the *Main* tool bar, click Open. On the *Datasource* menu, click Find Object.

OR

On the Main tool bar, click Find.

OR

On the Datasource Explorer, right-click a database, and then click Find Object.

Changing an ODBC Datasource to a Native Datasource

Option	Description
Object to Find	Lets you type the target string of text. You can also click the list to choose from a list of up to ten previous search strings.
Type of Object	Lets you select the target database object type.
Object Owner	Lets you select the target database object owner.
Search Direction	Lets you select a search direction: From Beginning Down Up
Case-Sensitive Search	Select to perform the search with the same capitalization as the search string.
Find Entire String Only	Select to perform the search using the entire search string, not partial strings.
Search My Objects Only	Select to perform the search only on your database objects.

CAUTION: The section below refers only to Microsoft SQL Server connectivity.

Microsoft SQL Server uses ODBC to connect to Microsoft SQL Servers. Rapid SQL requires native connectivity. To connect through Rapid SQL, register your Microsoft SQL Server(s) using native connectivity in the Microsoft SQL Server Client Utility.

To change your ODBC servers to native connectivity, do the following:

1   Open the Microsoft SQL Server Client Network Utility dialog box, CLICONFG.exe.

2   On the General Tab, click Add.

Microsoft SQL Server opens the Add Network Library Configuration dialog box.

The table below describes the options and functionality on the Add Network Library Configuration dialog box:

Option	Description
Server alias	In the box, type the unique name of the server.
Network libraries	In the box, click the appropriate option button to specify the network library that connects to the server.
Computer name	In the box, type the name of the target computer.

Option	Description
Port number	In the box, type the port number of the target computer.

3   Click OK.

Rapid SQL returns to the Client Network Utility dialog box.

4   In the Client Network Utility dialog box, click Apply.

Rapid SQL adds the server.

5   Open Rapid SQL and the Datasource Registration Wizard.

For more information, see Datasources.

CATEGORIZING DATASOURCES USING COLOR-CODING AND LABELLING

Rapid SQL provides a means to visually distinguish between datasources used for different purposes in your organization. You can assign a category, such as development or production, when creating or editing a datasource. This color-codes or labels particular user interface elements associated with that datasource.

When you assign a category to a datasource, the Datasource Explorer icons for the datasource shows a distinctive color-coded category indicator. Optionally, you can customize your categorization scheme to include the following user interface elements:

- A short name label included in the Datasource toolbar combo/dropdown when a categorized datasource is selected

- Color-coded tabs on windows associated with actions against a categorized datasource, editor or ISQL windows for example

- Color-coding of the status bar at the bottom of the Rapid SQL window
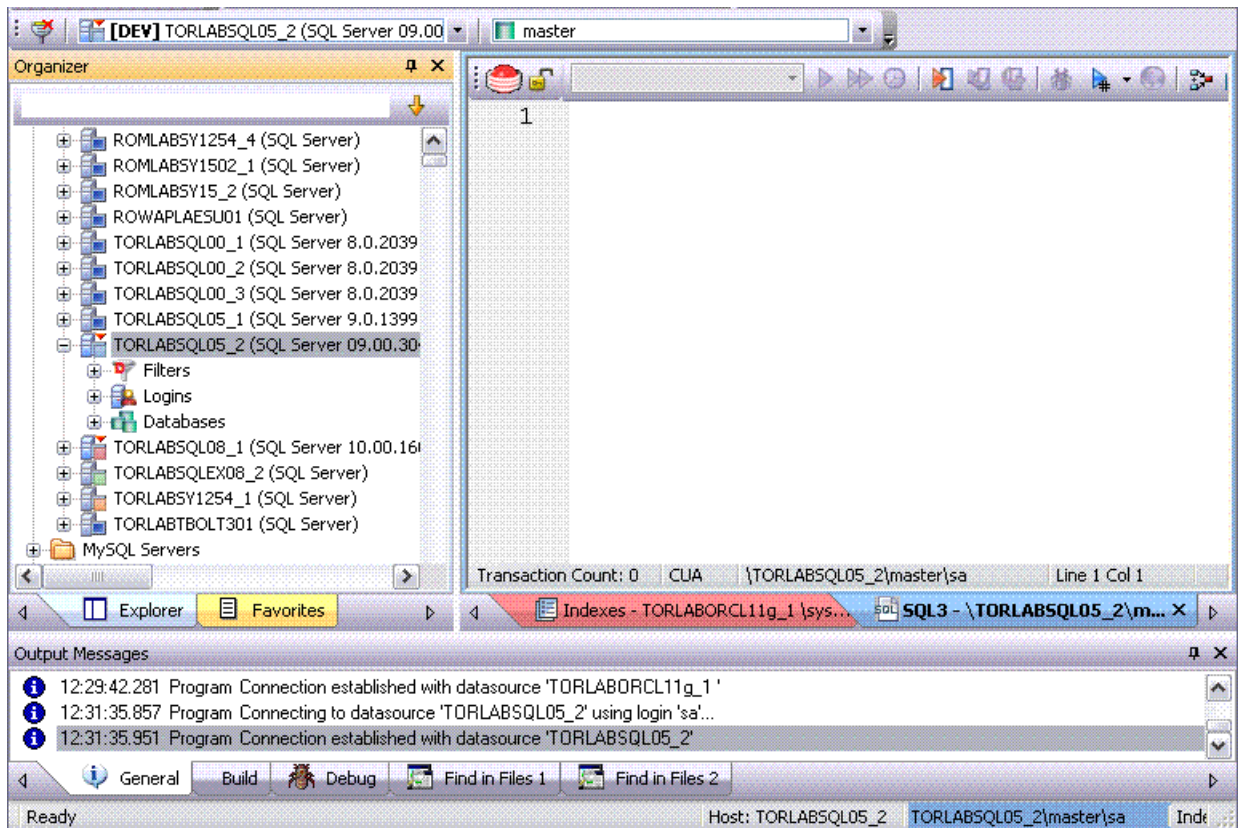
The default categories [and short name labels] and the associated colors are as follows:

Default Category	Short Name Label	Color
Production	**PROD**	
Development	**DEV**	
QA	**QA**	
Test	**TEST**	

In addition to the defaults, you can create custom category/color code/label category combinations.

To work with color-coded datasource resources, you must be familiar with the following tasks:

- Enabling the specific user interface items that can be color-coded or labelled. For details, see Datasource Options.

- Customizing your category scheme. For details, see Customizing Datasource Categories.

- Assigning a category to a datasource when creating or editing it. For details, see Datasource Registration.

ACTIVATING/DEACTIVATING ROLES IN THE CURRENT SESSION

Rapid SQL lets you activate or deactivate roles for the current session.

> **NOTE:** This functionality is available for Oracle and Sybase only.

The **Role Activation** action builds and submits one or more SET ROLE commands.

To enable or disable one or more roles in the current Rapid SQL session

1   Right-click on a connected Sybase or Oracle datasource node and select **Role Activation** from the context menu. The **Role Activation** dialog opens.

2   Use the following table as a guide to understanding and modifying settings in the wizard:

Panel	Settings and tasks
Role Activation	The list contains entries for each role that has been granted to the current login. For information on granting/assigning roles to a login or user, see Users Editor (Oracle) and Logins Editor (Sybase). To activate a role, select the **Active** check box associated with the role. If the role is configured to require a password, provide it in the **Password** box. To deactivate a role, deselect the Active check box associated with the role.
Preview	Preview the DDL generated for the operation and when ready, click Execute. For information on the other options, see Preview and Scheduling.

DATABASE OBJECT MANAGEMENT

For each supported DBMS, Rapid SQL provides extensive object management facilities. AImed at both Database Administrators and Developers, Rapid SQL provides support for the most commonly used object types. The following sets of topics describe the major areas of support:

Supported Objects	Describes the database objects and related elements supported by Rapid SQL. In addition to a listing of all supported objects by DBMS platform, a support summary is provided for each object type.
Creating objects	Tells you how to create new objects using graphical wizards.
Modifying objects using editors	Tells you how to edit existing objects.
Object actions	Provides details on the actions and operations that Rapid SQL lets you perform against database objects.

SUPPORTED OBJECTS

Rapid SQL lets you manage database objects across different database platforms. The table below indicates the objects that Rapid SQL supports by platform:

Objects	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	ORCL	SQL SVR	SYB
Aliases	✓	✓				✓	✓
Blob Filters			✓				
Check Constraints	✓	✓	✓		✓	X	✓
Clusters					✓		
Database Links					✓		✓
Database Triggers						✓	
Databases		✓			✓	✓	✓
Defaults		✓					
Defaults						✓	✓
Directories					✓		
Domains			✓				
Encryption Keys			✓				
Exceptions			✓				
Extended Procedures							✓
External Functions			✓				
Foreign Keys	✓	✓	✓	✓	✓	✓	✓
Full-text Catalogs						✓	
Full-text Indexes						✓	
Functions	✓	✓		✓	✓	✓	✓
Generators			✓				
Groups	✓				✓	✓	✓
Indexes	✓	✓	✓	✓	✓	✓	✓
Instance	✓				✓		
Java Classes					✓		
Java Resources					✓		
Java Sources					✓		
Job Queues					✓		
Libraries					✓		
Logins						✓	✓
Materialized Query Tables	✓						
Materialized Views					✓		

Objects	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	ORCL	SQL SVR	SYB
Materialized View Logs					✓		
Outlines					✓		
Package Bodies					✓		
Packages	✓	✓			✓		
Partition Functions						✓	
Partition Schemes						✓	
Plans		✓					
Primary Keys	✓	✓	✓	✓	✓	✓	✓
Procedures	✓	✓	✓		✓	✓	✓
Profiles					✓		
Recycle Bin					✓		
Roles			✓		✓	✓	✓
Rules						✓	✓
Segments						✓	✓
Sequences	✓	✓			✓		
Shadows			✓				
Structured Types	✓	✓					
Synonyms		✓			✓	✓	
Tables	✓	✓	✓	✓	✓	✓	✓
Tablespaces	✓	✓			✓		
Triggers	✓	✓	✓		✓	✓	✓
Type Bodies					✓		
Types					✓		
Unique Keys	✓	✓	✓	✓	✓	✓	✓
User Datatypes	✓	✓				✓	✓
User Messages	✓					✓	✓
Users	✓	✓			✓	✓	✓
Views	✓	✓	✓		✓	✓	✓

ALIASES

Aliases let you assume the permissions of another database user without creating a separate user identity. You can use an alias when a user requires only temporary access to a database. You can also use an alias to mask a user's identity.

Platform availability

DB2 LUW	DB2 z/OS	InterBase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓					✓

Creating and editing

- Alias Wizard (DB2 LUW) and Aliases Editor (IBM DB2 LUW)

- Alias Wizard (DB2 Z/OS) and Aliases Editor (IBM DB2 Z/OS)

- Alias Wizard (Sybase) and Aliases Editor (Sybase)

Object actions/operations supported

Drop (DB2 LUW, DB2 z/OS, Sybase)

Extract (DB2 LUW, DB2 z/OS, Sybase)

Report (DB2 LUW, DB2 z/OS, Sybase)

Transfer Ownership (DB2 LUW)

BLOB FILTERS

Rapid SQL provides support for blob filters, user-written programs that convert data stored in Blob columns from one subtype to another.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See Blob Filters Wizard (InterBase/Firebird) and Blob Filters Editor (InterBase/Firebird).

Object actions/operations supported

Drop   Extract   Rename   Report

CHECK CONSTRAINTS

Check constraints are data values that are acceptable in a column. They are logical expressions that verify column values meet defined acceptance criteria.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓	✓		✓	✓	✓

Creating and editing

See Add or Modify Check Constraint.

Object actions/operations supported

Change Status (check constraints) (Oracle, SQL Server)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Hide Text (Sybase)

Rename (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

CLUSTERS

Clusters provide an optional method of storing table data. A cluster comprises of a group of tables that share the same data blocks, and which are grouped together because they share common columns and are often used together. The related columns of tables stored in a cluster are known as the cluster key.

There are two types of clusters:

- Index
- Hash

Index clusters store the cluster data together and index the cluster key, which should make them faster at retrieving a range of data rows.

Hash clusters apply hashing functions to the cluster key to determine the physical location of a data row, which should make them faster at retrieving specific data rows.

> **NOTE:** To place a table on a cluster, 6 include the ON CLUSTER syntax within the CREATE TABLE statement. Placing a table on a cluster precludes you from placing it on a tablespace or defining the associated storage parameters.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

- Cluster Wizard (Oracle) and Clusters Editor (Oracle).

Object actions/operations supported

Allocate Extent	Analyze	Deallocate Unused Space	Drop	Extract
Report	Truncate			

DATABASE LINKS

Database links are named schema objects that describe a path from one database to another. Database links are implicitly used when a reference is made to a global object name in a distributed database. To use a database link, either it is public or you own it.

> **NOTE:** Oracle syntax does not let you alter an existing database link. To change its definition, drop and re-create it.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

- Database Link Wizard (Oracle) and Database Links Editor (Oracle)

Object actions/operations supported

[Drop](#)   [Extract](#)   [Rename](#)   [Report](#)

DATABASE TRIGGERS

A database trigger implements a database-scope DDL trigger. These triggers fire in response to events associated with DDL statements.

> **NOTE:** Before working with database triggers, consult Microsoft SQL Server documentation for a general understanding of DDL triggers. For more information, see [Accessing Third Party Documentation](#).

Platform availability

DB2 LUW	DB2 z/OS	InterBase/Firebird	MySQL	Oracle	SQL Server	Sybase
					2005 ^	

Creating and editing

- [Database Triggers Wizard (SQL Server)](#) and [Database Triggers Editor (SQL Server)](#)

Object actions/operations supported

[Change Status](#)   [Drop](#)   [Extract](#)   [Report](#)

DATABASES

Databases are a collection of tables, or a collection of index spaces and tablespaces. The goals of a database system are straightforward but challenging. In general, a database aims to manage large amounts of data in a multi-user environment. It should achieve high performance while letting many users access the same information concurrently without compromising data integrity. A database also must protect against unauthorized access and provide reliable solutions for failure recovery.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓		✓		✓	✓

Creating and editing

- Database Wizard (IBM DB2 LUW) and Databases Editor (IBM DB2 LUW)

- Database Wizard (DB2 Z/OS) and Databases Editor (IBM DB2 Z/OS)

- Database Wizard (SQL Server) and Databases Editor (SQL Server)

- Database Wizard (Sybase) and Databases Editor (Sybase)

Object actions/operations supported

Attach Database (SQL Server)

Checkpoint (SQL Server, Sybase)

Coalesce (Oracle)

Copy Schema (DB2 LUW)

DBCC (SQL Server, Sybase)

Detach Database (SQL Server)

Drop (DB2 LUW, DB2 z/OS, MySQL, SQL Server, Sybase)

Extract (DB2 z/OS, MySQL, SQL Server, Sybase)

Hide Text (Sybase)

Move Log (Sybase)

Quiesce (Database) DB2 LUW)

Rename (SQL Server, Sybase)

Report (DB2 z/OS, MySQL, SQL Server, Sybase)

Set Online/Offline (SQL Server, Sybase)

Shrink (SQL Server)

Start Database (DB2 z/OS)

Stop Database (DB2 z/OS)

Unquiesce (DB2 LUW)

Update Statistics (SQL Server)

DBMS-specific information

- [IBM DB2 for Linux, Unix, and Windows Instances](#)

- [Microsoft SQL Server Databases](#)

- [IBM DB2 for OS/390 and z/OS Instances](#)

- [Sybase ASE Databases](#)

IBM DB2 for Linux, Unix, and Windows Instances

Databases are a collection of tables, or a collection of index spaces and tablespaces. The goals of a database system are straightforward but challenging. In general, a database aims to manage large amounts of data in a multi-user environment. It should achieve high performance while letting many users access the same information concurrently without compromising data integrity. A database also must protect against unauthorized access and provide reliable solutions for failure recovery.

Microsoft SQL Server Databases

Databases are a collection of tables, or a collection of index spaces and tablespaces. The goals of a database system are straightforward but challenging. In general, a database aims to manage large amounts of data in a multi-user environment. It should achieve high performance while letting many users access the same information concurrently without compromising data integrity. A database also must protect against unauthorized access and provide reliable solutions for failure recovery.

> **NOTE:** Microsoft SQL Server recommends that you do not create any user objects, such as tables, views, stored procedures, or triggers, in the master database. The master database includes the system tables that store the system information used by SQL Server, such as configuration option settings.

IBM DB2 for OS/390 and z/OS Instances

Databases are a collection of tables, or a collection of index spaces and tablespaces. The goals of a database system are straightforward but challenging. In general, a database aims to manage large amounts of data in a multi-user environment. It should achieve high performance while letting many users access the same information concurrently without compromising data integrity. A database also must protect against unauthorized access and provide reliable solutions for failure recovery.

Sybase ASE Databases

Databases are a collection of tables, or a collection of index spaces and tablespaces. The goals of a database system are straightforward but challenging. In general, a database aims to manage large amounts of data in a multi-user environment. It should achieve high performance while letting many users access the same information concurrently without compromising data integrity. A database also must protect against unauthorized access and provide reliable solutions for failure recovery.

Defaults

Defaults promote data integrity by supplying a default value to a table column if the user does not explicitly provide one. They are reusable objects that you can bind to table columns or user datatypes.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	✓

Creating and editing

- Default Wizard (SQL Server) and Defaults Editor (SQL Server)

- Default Wizard (Sybase) and Defaults Editor (Sybase)

Object actions/operations supported

Drop (SQL Server, Sybase)

Extract (SQL Server, Sybase)

Hide Text (Sybase)

Rename (SQL Server, Sybase)

Report (SQL Server, Sybase)

Directories

Directories create an alias to an external operating system directory to your database files, which can be used for storing large binary object files. When you create a directory, provide the full path name to the outside operating system where the BFILE's are stored. This object lets you store large files, such as video, outside of the database. The directory object lets you provide a simple alias to the full path name of an outside server's file system, which you can then use to point to the files when creating procedural logic objects. This saves the developer from having to type the full path name when coding.

To create a Directory object, you need CREATE ANY DIRECTORY system privileges. You also create or have in place a corresponding operating system directory to store the file. This directory must have the correct read permissions for Oracle processes.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing
- [Directory Wizard (Oracle)](#) and [Directories Editor (Oracle)](#)

Object actions/operations supported

[Drop](#) [Extract](#) [Report](#)

DOMAINS

Rapid SQL provides support for domains, letting you work with these global column/datatype-based objects.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See [Domains Wizard (InterBase/Firebird)](#) and [Domains Editor (InterBase/Firebird)](#).

Object actions/operations supported

[Drop](#) [Extract](#) [Rename](#) [Report](#)

ENCRYPTION KEYS

Rapid SQL provides support for encryption keys, used in encrypting and decrypting tables and columns.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See [Encryption Keys wizard (InterBase/Firebird)](#) and [Encryption Keys editor (InterBase/Firebird)](#).

Object actions/operations supported

[Drop](#)   [Extract](#)   [Rename](#)   [Report](#)

EXCEPTIONS

Rapid SQL provides support for exceptions, letting you work with these named error messages.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See [Exceptions Wizard (InterBase/Firebird)](#) and [Exceptions Editor (InterBase/Firebird)](#).

Object actions/operations supported

[Drop](#)   [Extract](#)   [Rename](#)   [Report](#)

EXTENDED PROCEDURES

Extended Procedures are dynamic link libraries that can be used to load and execute application routines written in other programming languages, such as C or Visual Basic. Extended Procedures function and appear in the same manner as normal stored procedures in that you can pass parameters to them and obtain results.

> **NOTE:** Extended Procedures can only be accessed on the Master database.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
						✓

Creating and editing

- Extended Procedure Wizard (Sybase) and Extended Procedures Editor (Sybase)

Object actions/operations supported

Drop	Describe	Execute	Extract	Hide Text
Rename	Report			

EXTERNAL FUNCTIONS

Rapid SQL provides support for external functions, letting you work with declarations for existing functions.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See External Functions Wizard (InterBase/Firebird) and External Functions editor (InterBase/Firebird).

Object actions/operations supported

Rename

FOREIGN KEYS

Foreign keys enforce referential integrity between tables by verifying the existence of foreign key values in the parent table before letting you insert or update foreign key values in the child table.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓	✓	✓	✓	✓	✓

Creating and editing

- Foreign Key Wizard (IBM DB2 LUW) and Foreign Keys Editor (IBM DB2 LUW)

- Foreign Key Wizard (DB2 Z/OS)) and Foreign Keys Editor (IBM DB2 Z/OS)

- Foreign Keys wizard (MySQL) and Foreign Keys editor (MySQL)

- Foreign Key Wizard (Oracle) and Foreign Keys Editor (Oracle)

- Foreign Key Wizard (SQL Server) and Foreign Keys Editor (SQL Server)

- Foreign Key Wizard (Sybase) and Foreign Keys Editor (Sybase)

Object actions/operations supported

Change Status (check constraints) (Oracle, SQL Server)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Rename (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

DBMS platform-specific notes

MySQL	MySQL recognizes two types of tables, MyISAM and InnoDB. MyISAM tables access data records using an index, where InnoDB tables allow transactions and foreign keys. Therefore, the discussion of foreign keys is limited to your InnoDB tables.

FULL-TEXT CATALOGS

A full-text catalog is a set of operating system files that store full-text indexes. Full-text catalogs, along with full-text indexes and the object actions supported for these object types, provide full text search support in Rapid SQL.

NOTE: Before working with full-text catalogs, consult Microsoft SQL Server documentation for a general understanding of full-text searching. For more information, see Accessing Third Party Documentation.

Platform availability

DB2 LUW	DB2 z/OS	InterBase/Firebird	MySQL	Oracle	SQL Server	Sybase
					2005 ^	

Creating and editing

- Full-text Catalogs Wizard (SQL Server) and Full-text Catalogs Editor (SQL Server)

Object actions/operations supported

Drop   Extract   Rebuild (Full-text Catalogs)   Reorganize (SQL Server Full-text   Report
                                                                                Catalogs)

Related information

- Full-text Indexes

FULL-TEXT INDEXES

A full-text index stores information used by the Full-Text Engine to compile full-text queries that can quickly search a table for particular words or word combinations. A full-text index stores information about key words and their location within one or more columns of a table.

Full-text indexes, along with full-text catalogs and the object actions supported for these object types, provide full text search support in Rapid SQL.

> **NOTE:** Before working with full-text indexes, consult Microsoft SQL Server documentation for a general understanding of full-text searching. For more information, see Accessing Third Party Documentation.

Platform availability

DB2 LUW	DB2 z/OS	InterBase/Firebird	MySQL	Oracle	SQL Server	Sybase
					2005 ^	

Creating and editing

- Full-text Indexes Wizard (SQL Server) and Full-text Indexes Editor (SQL Server)

Object actions/operations supported

Change Status (Full-text Indexes)   Drop   Extract   Population status   Report

Related information
- Full-text Catalogs

FUNCTIONS

Functions are subroutines that you define. Functions are useful for reusable application logic. You can use functions to determine the best methods for controlling access and manipulation of the underlying data contained in an object.

The table below describes the types of user-defined functions that Rapid SQL lets you create:

Function	Description
Column or External Table Function	You can write in a host programming language, such as C. This function can act on a table and returns a table value rather than a scalar value.
External Scalar Function	You can write in a language other than SQL, such as C++ or Java and returns a scalar value to the program. This type of function is referenced by the CREATE FUNCTION statement and can be used to perform computations on data contained in the database but cannot directly reference the data.
OLEDB Function	Accesses OLE DB data in user-defined OLE DB external tables.
Sourced Function	Inherits the semantics of another function and can be an operator.
Template Function	Partial functions that do not contain any executable code. Mainly used in a federated database to map the template function to a data source function -Oracle, SQL Server, Sybase, etc. A function mapping needs to be created in conjunction with the template function.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓		✓	✓	✓	✓

Creating and editing
- Function Wizard (IBM DB2 LUW) and Functions Editor (IBM DB2 LUW)

- Function Wizard (DB2 Z/OS) and Functions Editor (IBM DB2 Z/OS)

- Functions wizard (MySQL) and Functions editor (MySQL)

- Function Wizard (Oracle) and Functions Editor (Oracle)

- Function Wizard (SQL Server) and Functions Editor (SQL Server)

- [Functions Wizard (Sybase)](#) and [Functions Editor (Sybase)](#)

Object actions/operations supported

[Compile](#) (Oracle)

[Create Synonym](#) (Oracle)

[Describe](#) (DB2 LUW, DB2 z/OS, Oracle)

[Drop](#) (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

[Execute](#) (Oracle)

[Extract](#) (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

[Hide Text](#) (Sybase)

[Rename](#) (SQL Server, Sybase)

[Report](#) (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

[Transfer Ownership](#) (DB2 LUW)

Important Notes

With respect to functions and stored procedures, it can be necessary to have statements executed before and after creation of the procedure or function. This can be useful for example, if you need to create or drop temporary tables used by the function or procedure. Rapid SQL supports the use of two tag pairs, ETStart and ETEnd, that let you embed statements in the first comment block of a stored procedure or function. The following shows the expected syntax:

```
create procedure dbo.procname(@a numeric) as
/*
<ETStart>SQL Statement</ETStart>
<ETEnd>SQL Statement</ETEnd>
*/
begin
...
```

GENERATORS

Rapid SQL provides support for generators, letting you work with these objects used to generate sequential numbers in a column.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See [Generators Wizard (InterBase/Firebird)](#) and [Generators editor (InterBase/Firebird)](#).

Object actions/operations supported

[Drop](#)   [Extract](#)   [Rename](#)   [Report](#)

GROUPS

Groups are a defined collection of database users. The primary use of groups is to consolidate the management of permissions. By matching together similar users into groups, you can greatly reduce the number of commands required to set permissions.

Every user automatically belongs to the public group. To assign a user to another group, add the user to that group. Then the user belongs to that group and public.

> **NOTE:** A user can only belong to one group at a time other than public.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
						✓

Creating and editing

- [Group Wizard (Sybase)](#) and [Groups Editor (Sybase)](#)

Object actions/operations supported

[Drop](#)   [Extract](#)   [Report](#)

INDEXES

Indexes are optional structures associated with tables. You can create indexes specifically to speed SQL statement execution on a table. When properly used, Indexes are the primary means of reducing disk I/O. Indexes are logically and physically independent of the data in the associated table. Unique Indexes guarantee that no two rows of a table have duplicate values in the columns that define the index.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓	✓	✓	✓	✓	✓

Creating and editing

- Index Wizard (DB2 LUW) and Indexes Editor (IBM DB2 LUW)

- Index Wizard (DB2 Z/OS) and Indexes Editor (IBM DB2 Z/OS)

- Indexes, Primary Keys, or Unique Keys wizard (MySQL) and Indexes, Primary Keys, and Unique Keys editors (MySQL)

- Index Wizard (Oracle) and Indexes Editor (Oracle)

- Index Wizard (SQL Server) and Indexes Editor (SQL Server)

- Index Wizard (Sybase) and Indexes Editor (Sybase)

Object actions/operations supported

Allocate Extent (Oracle)

Analyze (Oracle)

Deallocate Unused Space (Oracle)

Disable Index (ITB/FBD, SQL Server)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

DBCC (SQL Server, Sybase)

Estimate Size (Oracle, Sybase)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Place (Sybase)

Rebuild Index (ITB/FBD, Oracle, SQL Server)

Rename (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Reorganize /Reorg (DB2 LUW, SQL Server, Sybase)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Set Statistics (ITB/FBD)

Shrink (Oracle)

Transfer Ownership (DB2 LUW)

Update Statistics (DB2 LUW, SQL Server, Sybase)

DBMS-specific information

- [IBM DB2 for Linux, Unix, and Windows Indexes](#)

- [IBM DB2 for OS/390 and z/OS Indexes](#)

- [Microsoft SQL Server Indexes](#)

- [Oracle Indexes](#)

- [Sybase ASE Indexes](#)

IBM DB2 FOR LINUX, UNIX, AND WINDOWS INDEXES

IBM DB2 for Linux, Unix, and Windows offers two types of indexes:

- Unique

- Non-Unique

Unique Indexes guarantee that no two rows of a table have duplicate values in the columns that define the index.

MICROSOFT SQL SERVER INDEXES

Microsoft SQL Server offers two types of indexes: clustered and non-clustered. Clustered indexes physically sort table data to match their logical order. Non-clustered indexes only order the table data logically. In a database, an index lets you speed queries by setting pointers that allow you to retrieve table data without scanning the entire table. An index can be unique or non-unique.

Microsoft SQL Server creates indexes as B-Trees, which are a series of pointers mapping index pages to their underlying data pages. As tables and, therefore, indexes grow, the number of levels in the B-Tree increases. The B-Tree of a clustered index is shorter than that of a non-clustered index because the leaf level of a clustered index is the data page.

A sound indexing strategy is critical to overall system performance. One pitfall to avoid is placing many indexes on a table without regard for their cumulative cost. Remember that indexes improve read but slow write performance because Microsoft SQL Server must update more information in the system catalog. Consequently, extra indexes can actually slow overall performance if data modification occurs frequently on the table. To determine the efficacy of indexes, you should tune your queries using SHOWPLAN and IO STATISTICS and analyze the selectivity of indexes using DBCC SHOW_STATISTICS.

Rapid SQL separates system indexes from user-defined indexes in the Datasource Explorer to ensure that system indexes are not accidentally altered or dropped.

ORACLE INDEXES

Oracle offers two types of indexes. The table below describes these indexes:

Index	Description
Table	A table index is defined on an individual table.
Cluster	A cluster index is defined on a set of tables physically stored together in a cluster. In an Oracle database, both table and cluster indexes use a B-tree structure.

The indexing strategy, particularly with large, active tables, is critical to overall system performance. The optimal definition and number of indexes for a given table is determined by the mix of access paths to that table performing insert, update, delete and select operations. For example, adding or changing an index can speed up your selects but slow your inserts, updates and deletes. Careful tuning and testing helps you achieve the best overall performance.

> **TIP:** Indexes generally improve read operations in a database, but you should not place too many indexes on some tables. Since Oracle must maintain each index along with its referenced table, placing too many indexes on a table that is the object of much insert, update, and delete activity, can actually degrade performance.

Even when an index exists on a table, the way a SQL statement is coded can actually disallow the use of the index. To prevent this from happening, follow these rules of thumb:

- Try not to use SQL statements that include the NOT IN, NOT LIKE, <>, IS NULL operators because they typically suppress the use of indexes.

- When referencing concatenated indexes with queries, be sure the leading column in the index is used. If it isn't, the index won't be used at all.

- Avoid using functions in WHERE predicates.

If you must use functions, and you are using Oracle8i, investigate the use of function-based indexes.

INDEX PARTITIONS

Index partitions are similar to table partitions. There are three types of partitioned indexes that Oracle supports:

1 Local prefixed

2 Local nonprefixed

3 Global prefixed

> **NOTE:** An index cannot be partitioned if it is a cluster index or if the index is defined on a clustered table.

Local prefixed and nonprefixed indexes

A local partitioned index has keys that refer to rows in a single table partition. A local partitioned index is automatically partitioned to mirror the underlying table. The number of partitions or subpartitions and the partition bounds for the partitioned index correspond with the partitions on the table. Oracle maintains this correspondence. If the table partitions are altered, the index partitions are altered accordingly.

A local partitioned index is prefixed if it is partitioned on the same column as the underlying table. The local partitioned index is nonprefixed if it is partitioned on a different column.

Global prefixed indexes

A global partitioned index can refer to rows in more than one table partition or subpartition. Global partitioned indexes are more difficult to manage than local partitioned indexes because any change in the underlying table partition affects all partitions in a global index. As a result, there is increased partition maintenance.

> **NOTE:** A global index can only be range partitioned but it can be defined on any kind of partitioned table.

IBM DB2 FOR OS/390 AND z/OS INDEXES

IBM DB2 for OS/390 and z/OS offers two types of indexes:

- Unique

- Non-Unique

Unique Indexes guarantee that no two rows of a table have duplicate values in the columns that define the index.

Non-Unique indexes let table rows have duplicate values in the columns that define the indexes.

SYBASE ASE INDEXES

Sybase ASE offers two types of indexes: clustered and non-clustered. Clustered indexes physically sort table data to match their logical order. Non-clustered indexes only order the table data logically. In a database, an index lets you speed queries by setting pointers that let you retrieve table data without scanning the entire table. An index can be unique or non-unique.

Sybase ASE creates indexes as B-Trees, which are a series of pointers mapping index pages to their underlying data pages. As tables and, therefore, indexes grow, the number of levels in the B-Tree increases. The B-Tree of a clustered index is shorter than that of a non-clustered index because the leaf level of a clustered index is the data page.

A sound indexing strategy is critical to overall system performance. One pitfall to avoid is placing many indexes on a table without regard for their cumulative cost. Remember that indexes improve read but slow write performance because Sybase ASE must update more information in the system catalog. Consequently, extra indexes can actually slow overall performance if data modification occurs frequently on the table. To determine the efficacy of indexes, you should tune your queries using SHOWPLAN and IO STATISTICS and analyze the selectivity of indexes using DBCC SHOW_STATISTICS.

Rapid SQL separates system indexes from user-defined indexes in the Datasource Explorer to ensure that system indexes are not accidentally altered or dropped.

INSTANCE

> **NOTE:** This object is support for IBM DB2 and Oracle.

Rapid SQL places Instance as the first level of information under the Datasource node in the Datasource Explorer. Instance includes:

- DB Manager Configuration

- Data sources

Available Functionality

Rapid SQL offers the following functionality for this object:

Database Wizard (IBM DB2 LUW)	Quiesce (Database)	Unquiesce		

JAVA CLASSES

The Java Classes contain compiled Java code. Java Classes are made up of a group of data items, with associated functions that perform operations. The data items are called fields or variables; the functions are referred to as methods.

> **TIP:** Oracle is shipped with a JVM (Java Virtual Machine). The JVM provided by Oracle sits atop the Oracle RDBMS and interacts directly with the RDBMS instead of the operating system.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Object actions/operations supported

Compile	Create Synonym	Drop	Load Java	Report

JAVA RESOURCES

> **NOTE:** This object is supported by Oracle only.

The Java Resources node of the *Explorer* tab offers support for browsing Java resources.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Available Functionality

- Drop

JAVA SOURCES

Java Sources contain the uncompiled Java source code.

> **TIP:** Oracle is shipped with a JVM (Java Virtual Machine). The JVM provided by Oracle sits atop the Oracle RDBMS and interacts directly with the RDBMS instead of the operating system.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Object actions/operations supported

Compile	Drop	Extract	Load Java	Rename
Report				

JOB QUEUES

Job Queues are built-in mechanisms that let you schedule a variety of SQL-based or command-line driven tasks.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

- Job Queue Wizard (Oracle) and Job Queue Editor (Oracle)

Object actions/operations supported

Disable Job	Drop	Enable Job (Job Queue)	Extract	Report
Run Job				

LIBRARIES

Libraries are an object type introduced in Oracle8 that represent a call to an operating system shared library. After the call is made, libraries can be used by SQL or PL/SQL to link to external procedures or functions. Libraries are only to be used on operating systems that support shared libraries and dynamic linking. Libraries serve as pointers or aliases to physical operating system shared library files and do not have existence as a physical object on their own, rather they rely on the physical existence of the files in the external operating system library to which they refer. To access the function or procedures stored in the library, you need execute privileges at the operating system level where the shared library resides.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

- Library Wizard (Oracle) and Libraries Editor (Oracle)

Object actions/operations supported

Drop   Extract   Rename   Report

LOGINS

Logins let you access your account. Your login account controls access to the server and all of the databases within it. Only the System Administrator or System Security Officer can create logins. Once you can log into a server, you need additional privileges to access user databases. Specifically, each database owner adds the login as a user or alias to the database.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	✓

Creating and editing

- Login Wizard (SQL Server) and Logins Editor (SQL Server)

- Login Wizard (Sybase) and Logins Editor (Sybase)

Object actions/operations supported

Add/Modify Login Trigger (Sybase)

Bind To Temporary Database (Sybase)

Change Password (SQL Server)

Create Like (SQL Server)

Drop (SQL Server, Sybase)

Drop Login Trigger (Sybase)

Extract (SQL Server, Sybase)

Report (SQL Server, Sybase)

Unbind From Temporary Database (Sybase)

For more information, see the following topics:

- Microsoft SQL Server Logins

- Sybase ASE Logins

MICROSOFT SQL SERVER LOGINS

Logins let you access your account. Your login account controls access to the server and all of the databases within it. Only the System Administrator or System Security Officer can create logins. Once you can log into a server, you need additional privileges to access user databases. Specifically, each database owner adds the login as a user or alias to the database.

SYBASE ASE LOGINS

Logins let you access your account. Your login account controls access to the server and all of the databases within it. Only the System Administrator or System Security Officer can create logins. Once you can log into a server, you need additional privileges to access user databases. Specifically, each database owner adds the login as a user or alias to the database.

MATERIALIZED QUERY TABLES

NOTE: This object is supported by IBM DB2 for Linux, Unix, and Windows version 8.

A materialized query table is a table whose definition is based on the result of a query. The materialized query table typically contains pre-computed results based on the data existing in the table or tables that its definition is based on. If the SQL compiler determines that a query will run more efficiently against a materialized query table than the base table or tables, the query quickly executes against the materialized query table.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓						

Creating and editing

- Materialized Query Table Wizard (DB2 LUW) and Materialized Query Tables Editor (IBM DB2 LUW)

Object actions/operations supported

Build Query	Create Alias	Create Like	Create View	Describe
Drop	Drop Materialized Query Table	Extract	Lock	Refresh Table
Reorganize /Reorg	Report	Schema	Select * From	Set Integrity
Transfer Ownership	Update Statistics			

MATERIALIZED VIEWS

> **NOTE:** This object is supported by Oracle only.

Materialized views are used to dynamically copy data between distributed databases. There are two types of materialized views:

- Complex

- Simple

Complex materialized views copy part of a master table or data from more than one master table. Simple materialized views directly copy a single table. You cannot directly update the underlying data contained in materialized views.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				8.1 ^		

Creating and editing
- Materialized View Wizard (Oracle) and Materialized Views Editor (Oracle)

Object actions/operations supported

Compile	Create Synonym	Drop	Extract	Rename
Report	Shrink			

MATERIALIZED VIEW LOGS

Materialized View logs are tables that maintain a history of modifications to the master table, and they are used to refresh simple materialized views. When you create a materialized view log, Oracle automatically creates a log table to track data changes in the master table and a log trigger to maintain the data in the log table.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				8.1 ^		

Creating and editing

- [Materialized View Log Wizard (Oracle)](#) and [Materialized View Logs Editor (Oracle)](#)

Object actions/operations supported

[Drop](#) [Extract](#) [Report](#) [Shrink](#)

OUTLINES

Outlines are a set of results for the execution plan generation of a particular SQL statement. When you create an outline, plan stability examines the optimization results using the same data used to generate the execution plan. That is, Oracle uses the input to the execution plan to generate an outline, and not the execution plan itself.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

- [Outline Wizard (Oracle)](#) and [Outlines Editor (Oracle)](#)

Object actions/operations supported

[Change Category](#)	[Drop](#)	[Drop By Category](#)	[Drop Unused](#)	[Disable Keys](#)
[Extract](#)	[Reassign By Category](#)	[Rebuild Outlines](#)	[Rename](#)	[Report](#)

PACKAGES

NOTE: This object is supported by IBM DB2 for Linux, Unix, and Windows, IBM DB2 for OS/390 and z/OS, and Oracle only.

Packages contain all the information needed to process SQL statements from a single source file. You can use packages to process and call batches of SQL. Depending on the platform, packages can include:

- Procedures

- Functions

- Types

- Variables

- Constants

- Exceptions

- Cursors

- Subprograms

Packages offer a number of important advantages over using standalone procedures and functions, including the ability to:

- Modify package objects without recompiling dependent database objects.

- Declare global variables and cursors that can be shared within the package.

- Grant privileges more efficiently.

- Load multiple package objects into memory at once.

Packages usually have two parts: a header or specification and a body, although sometimes the body is unnecessary. The package header declares the members of the package while the body details the logic underlying each of the package components.

> **NOTE:** Rapid SQL splits out package headers and package bodies in the Datasource Explorer; however, you create both a package header and body from the packages node.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓			✓		

Creating and editing
- Packages Editor (IBM DB2 LUW)

- Packages Editor (IBM DB2 Z/OS)

- Package Wizard (Oracle) and Packages Editor (Oracle)

Object actions/operations supported
Bind Package (DB2 z/OS)

Compile (Oracle)

Create Synonym (Oracle)

Drop (DB2 LUW, Oracle)

Extract (Oracle)

Flush Cache (DB2 LUW)

Free (Packages) (DB2 z/OS)

Rebind Packages (DB2 LUW, DB2 z/OS)

Report (DB2 LUW, DB2 z/OS, Oracle)

PACKAGE BODIES

Package Bodies implement the package specification in that the package body includes the definition of every cursor and subprogram declared in the package specification.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

While Package Bodies are listed as a separate object in the Datasource Explorer, they are created on the Packages Editor in conjunction with Packages. For more information, see Package Bodies Editor (Oracle).

Object actions/operations supported

Compile	Create Synonym	Drop	Extract	Report

PARTITION FUNCTIONS

Partition functions are functions that maps the rows of a table or index into partitions based on the values of a specified column. Partition functions are referenced in partition scheme definitions in partitioning indexes or tables.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	

Creating and editing

See the following topics:

- Partition Functions Wizard (SQL Server) and Partition Functions Editor (SQL Server)

Available Functionality

Rapid SQL offers the following functionality for this object type:

Drop	Extract	Report

For related information, see the following topics:

- Partition Schemes and Partition Scheme Wizard (SQL Server)

- Index Wizard (SQL Server)

- Table Wizard (SQL Server)

PARTITION SCHEMES

Partition schemes map the partitions of a partitioned table or index to filegroups. The number and domain of the partitions for a partitioned table or index are specified in a partition function definition. Partition scheme definitions are referenced when partitioning indexes or tables.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	

Creating and editing

See the following topics:

- Partition Scheme Wizard (SQL Server) and Partition Schemes Editor (SQL Server)

Available Functionality

Rapid SQL offers the following functionality for this object type:

Drop	Extract	Next Used Filegroup	Report

For related information, see the following topics:

- [Partition Functions](#) and [Partition Functions (SQL Server) - Properties](#)

- [Index Wizard (SQL Server)](#)

- [Table Wizard (SQL Server)](#)

PLANS

A Plan is an executable application created in the bind process. It can include one or more packages or debris.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
	✓					

Creating and editing

- [Plan Wizard (DB2 Z/OS)](#) and [Plans Editor (IBM DB2 Z/OS)](#)

Object actions/operations supported

[Drop](#)   [Free Plan](#)   [Rebind Plans](#)   [Report](#)

PRIMARY KEYS

> **NOTE:**   This object is supported by all platforms.

Primary Keys are a set of table columns that can uniquely identify every row of a table.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓		✓	✓	✓	✓

Creating and editing

- [Primary Key Wizard (DB2 LUW)](#) and [Primary Keys Editor (IBM DB2 LUW)](#)

- [Primary Key Wizard (DB2 Z/OS)](#) and [Primary Keys Editor (IBM DB2 Z/OS)](#)

- [Indexes, Primary Keys, or Unique Keys wizard (MySQL)](#) and [Indexes, Primary Keys, and Unique Keys editors (MySQL)](#)

- [Primary Key Wizard (Oracle)](#) and [Primary Keys Editor (Oracle)](#)

- [Primary Key Wizard (SQL Server)](#) and [Primary Keys Editor (SQL Server)](#)

- [Primary Key Wizard (Sybase)](#) and [Primary Keys Editor (Sybase)](#)

Object actions/operations supported

[Allocate Extent](#) (Oracle)

[Analyze](#) (Oracle)

[Change Status (check constraints)](#) (Oracle)

[Deallocate Unused Space](#) (Oracle)

[Disable Index](#) (SQL Server)

[Drop](#) (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

[Extract](#) (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

[Rebuild Index](#) (Oracle, SQL Server)

[Rename](#) (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

[Reorganize /Reorg](#) (SQL Server)

[Report](#) (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

[Transfer Ownership](#) (DB2 LUW)

PROCEDURES

Procedures are a reusable block of PL/SQL, stored in the database, that applications can call. Procedures streamline code development, debugging, and maintenance by being reusable. Procedures enhance database security by letting you write procedures granting users execution privileges to tables rather than letting them access tables directly.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓			✓	✓	✓

Creating and editing

- [Procedure Wizard (DB2 LUW)](#) and [Procedures Editor (IBM DB2 LUW)](#)

- [Procedure Wizard (DB2 Z/OS)](#) and [Procedures Editor (IBM DB2 Z/OS)](#)

- [Procedure Wizard (Oracle)](#) and [Procedures Editor (Oracle)](#)

- [Procedure Wizard (SQL Server)](#) and [Procedures Editor (SQL Server)](#)

- [Procedure Wizard (Sybase)](#) and [Procedures Editor (Sybase)](#)

Object actions/operations supported

[Build](#) (DB2 z/OS)

[Compile](#) (DB2 LUW, Oracle)

[Create Synonym](#) (Oracle)

[Describe](#) (DB2 LUW, DB2 z/OS, Oracle, SQL Server, Sybase)

[Drop](#) (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

[Execute](#) (ITB/FBD, Oracle, Sybase)

[Extract](#) (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

[Hide Text](#) (Sybase)

[Rename](#) (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

[Report](#) (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

[Transfer Ownership](#) (DB2 LUW)

For more specific information, see the following topics:

- [DB2 z/OS Procedures](#)
- [Sybase Procedures](#)
- [Executing Statements Before and After Procedure or Function Creation](#)

DB2 z/OS PROCEDURES

Only IBM DB2 for OS/390 and z/OS SQL stored procedures created by Rapid SQL, or IBM's Stored Procedure Builder can be retrieved by Rapid SQL.

SYBASE PROCEDURES

Procedures perform procedural logic in your Sybase ASE applications. They are batches of SQL statements that are compiled and stored in the system catalog. Procedures execute faster than embedded SQL statements because they are pre-compiled and have execution plans for use by the optimizer. When you create a procedure, Sybase ASE builds a query tree, which it stores in a system table. When you execute a procedure for the first time, Sybase ASE loads it from the system table, compiles, and optimizes it. Sybase ASE places the resulting query plan in the procedure cache where it remains on a most recently used basis. In addition to better performance, procedures yield other benefits, including easier code maintenance, additional security and reduced network traffic.

> **NOTE:**   Rapid SQL also supports extended procedures. For details, see [Extended Procedures](#).

EXECUTING STATEMENTS BEFORE AND AFTER PROCEDURE OR FUNCTION CREATION

With respect to functions and stored procedures, it can be necessary to have statements executed before and after creation of the procedure or function. This can be useful for example, if you need to create or drop temporary tables used by the function or procedure. Rapid SQL supports the use of two tag pairs, ETStart and ETEnd, that let you embed statements in the first comment block of a stored procedure or function. The following shows the expected syntax:

```
create procedure dbo.procname(@a numeric) as
/*
<ETStart>SQL Statement</ETStart>
<ETEnd>SQL Statement</ETEnd>
*/
begin
...
```

PROFILES

Profiles are a mechanism for allocating system and database resources to users. Profiles let you specify limits on:

- Number of sessions

- CPU time

- Connect time

- Idle time

- Logical reads and space in the SGA's shared pool

You can assign profiles to one or more users. The database's default profile and all of its resource limits are assigned to users without a specific profile assignment.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing
- Profile Wizard (Oracle) and Profiles Editor (Oracle)

Object actions/operations supported

[Drop](#)   [Extract](#)   [Report](#)

RECYCLE BIN

Rapid SQL provides basic support for Oracle's Recycle Bin feature. When enabled, dropped tables, indexes, and associated objects are stored in the Recycle Bin until explicitly purged.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

The Recycle Bin cannot be created or edited.

Object actions/operations supported

On the Database Explorer, expanding the **Recycle Bin** node has the same effect as issuing a SHOW RECYCLEBIN statement. The contents of the Recycle Bin are displayed.

Related actions include:

[Enable Recycle Bin](#)	[Flashback Recycle Bin Entry](#)	[Purge Recycle Bin](#)	[Purge Recycle Bin Entry](#)

> **NOTE:** **Select * From** functionality is supported as a right-click option for individual tables in the Recycle Bin. For details on this feature, see [Select * From](#).

ROLES

Roles are sets of user privileges you associate with access to objects within a database. Roles streamline the process of granting permissions. You can use roles to grant sets of permissions and privileges to users and groups.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓	✓	

Creating and editing
- [Role Wizard (SQL Server)](#) and [Roles Editor (SQL Server)](#)

- Role Wizard (Oracle) and Roles Editor (Oracle)

Object actions/operations supported

Drop (ITB/FBD, Oracle, SQL Server, Sybase)

Extract (ITB/FBD, Oracle, SQL Server, Sybase)

Report (ITB/FBD, Oracle, SQL Server, Sybase)

For related information

- Microsoft SQL Server Roles

- Oracle Roles

-

MICROSOFT SQL SERVER ROLES

Roles let you manage permissions for users to perform a set of activities based on job functions. Roles are a mechanism for streamlining the assignment of object and system privileges to multiple users. Instead of granting sets of privileges to individual users, you can create a role, grant system and object privileges to that role, and then grant that role to all the users who should share the same privileges. You can grant one or more roles to the same user. Therefore, if you need to change the permissions for a certain role you can do so, and not have to grant or revoke the permissions for each individual user.

ORACLE ROLES

Roles are a mechanism for streamlining the assignment of object and system privileges to multiple users. Instead of granting sets of privileges to individual users, you can create a role, grant system and object privileges to that role, then simply grant that role to all the users who should share the same privileges. You can grant one or more roles to the same user.

DBMS-specific functionality

- For information on activating and deactivating roles for the current login in the current session, see Activating/Deactivating Roles in the Current Session.

RULES

Rules promote data integrity by allowing you to validate the values supplied to a table column. They are reusable objects that you can bind to table columns or user datatypes. For example, you can create a rule, bind it to a column in a table and have it specify acceptable values that can be inserted into that column.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	✓

Creating and editing
- Rule Wizard (SQL Server) and Rules Editor (SQL Server)

- Rule Wizard (Sybase) and Rules Editor (Sybase)

Object actions/operations supported

Drop (SQL Server, Sybase)

Extract (SQL Server, Sybase)

Hide Text (Sybase)

Rename (SQL Server, Sybase)

Report (SQL Server, Sybase)

SCHEMA

A Schema is a container of objects that can be owned by any user.

> **NOTE:** The Schema nodes available under DB2 LUW, DB2 z/OS, and Oracle datasources do not own a specific set of objects. They are simply an organizational container for all database objects for a datasource or database.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	

Creating and editing
- Schema Wizard (SQL Server) and Schemas Editor (SQL Server)

Object actions/operations supported

Drop   Extract   Report

SEGMENTS

> **NOTE:** This object is supported by Sybase only.

Segments are a mechanism for placing tables and indexes on specific logical partitions. You create segments on one or more fragments of a database. You can map segments to specific database fragments, which in turn reside on specific hard disks; and, mapping segments lets you increase i/o throughput by placing intensively used tables and indexes on different physical devices. You can allocate tables and indexes to segments by including placement statements at the end of CREATE TABLE or CREATE INDEX statements.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
						✓

Creating and editing
- Segment Wizard (Sybase) and Segments Editor (Sybase)

Object actions/operations supported

Drop   Extract   Report

SEQUENCES

Sequences are programmable database objects that generate a definable sequence of values. Once defined, a sequence can be made available to many users. A sequence can be accessed and incremented by multiple users with no waiting. A sequence can be used to automatically generate primary key values for tables. When you create a sequence, you can define its initial value, increment interval and maximum value.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓				✓		

Creating and editing
- Sequence Wizard (DB2 LUW) and Sequences Editor (IBM DB2 LUW)
- Sequence Wizard (Oracle) and Sequences Editor (Oracle)

Object actions/operations supported

Create Alias (DB2 LUW)

Create Synonym (Oracle)

Drop (DB2 LUW, Oracle)

Extract (DB2 LUW, Oracle)

Rename (DB2 LUW, Oracle)

Report (DB2 LUW, Oracle)

Restart (DB2 LUW)

Transfer Ownership (DB2 LUW)

-

SHADOWS

Rapid SQL provides support for shadows, letting you create one or more in-sync copies of the database stored on secondary storage devices.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
		✓				

Creating and editing

See Shadows wizard (InterBase/Firebird) and Shadows editor (InterBase/Firebird).

Object actions/operations supported

Drop	Extract	Rename	Report

STRUCTURED TYPES

Structured types define an abstract data type or object composed of a collection of similar types of data. For example, create an structured type that defines a full address rather than the pieces of an address, such as city, state and postal code. An structured type stores the pieces of an address in a single type, storing them in the same location and allowing the full address to be accessed and manipulated as single unit rather than multiple units.

Structured types are useful for ensuring uniformity and consistency as they are defined as single encapsulated entity that can be reused in other structured types and objects. They also offer flexibility by allowing for the creation of objects that represent real-world situations which is limited in relational objects.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓						

Creating and editing
- [Structured Type Wizard (DB2 LUW)](#) and [Structured Types Editor (IBM DB2 LUW)](#)

Object actions/operations supported

[Drop](#)   [Extract](#)   [Report](#)   [Transfer Ownership](#)

SYNONYMS

The function and usage of synonyms differs by DBMS platform

DB2 z/OS	An alternate name for an existing table, view, or alias.
Oracle	An alternative name for a table, view, sequence, procedure, function, package, materialized view, java class, type, or another synonym.
SQL Server	A single-part name used to reference multi-part names in SQL statements. While synonyms can be created for functions, procedures, tables, and views, consult Microsoft SQL Server documentation for restrictions on function and procedure support. For more information, see [Accessing Third Party Documentation](#).

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
	✓			✓	✓	

Creating and editing
- [Synonym Wizard (DB2 Z/OS)](#) and [Synonyms Editor (IBM DB2 Z/OS)](#)
- [Synonym Wizard (Oracle)](#) and [Synonyms Editor (Oracle)](#)
- [Synonym Wizard (SQL Server)](#) and [Synonyms Editor (SQL Server)](#)

Object actions/operations supported

[Create Synonym](#) (Oracle)

[Describe](#) (Oracle)

[Drop](#) (DB2 z/OS, Oracle, SQL Server)

[Extract](#) (DB2 z/OS, Oracle, SQL Server)

Rename (Oracle, SQL Server)

Report (DB2 z/OS, Oracle, SQL Server)

TABLES

Tables are a the basic unit of data storage. Tables store all the data accessible to users in rows and columns. Each column has a name, datatype and other associated properties. After you define a table, users can insert valid data into the table, which you can later query, update and delete.

> **NOTE:** Rapid SQL separates system tables from user-defined tables in the Datasource Explorer to ensure that system tables are not accidentally altered or dropped.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓		✓	✓	✓	✓

Creating and editing

- Table Wizard (DB2 LUW) and Tables Editor (IBM DB2 LUW)

- Table Wizard (DB2 Z/OS) and Tables Editor (IBM DB2 Z/OS)

- Tables wizard (MySQL) and Tables editor (MySQL)

- Table Wizard (Oracle) and Tables Editor (Oracle)

- Table Wizard (SQL Server) and Tables Editor (SQL Server)

- Table Wizard (Sybase) and Tables Editor (Sybase)

Object actions/operations supported

Allocate Extent (Oracle)

Analyze (Oracle)

Analyze Tables (MySQL)

Build Query (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Change Access Status (Oracle)

Check Tables (MySQL)

Checksum Tables (MySQL)

Convert Tables (MySQL)

Create Alias (DB2 LUW, DB2 z/OS)

Create Clone (DB2 z/OS)

Create Insert Statements (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Create Like (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Create Synonym (Oracle)

Create View (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

DBCC (SQL Server, Sybase)

Deallocate Unused Space (Oracle)

Delete Statistics (Sybase)

Describe (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Disable Keys (MySQL)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Drop Clone (DB2 z/OS)

Drop Unused Columns (Oracle)

Edit Data (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Enable Keys (MySQL)

Disable/Enable Triggers (SQL Server, Sybase)

Estimate Size (Oracle, Sybase)

Exchange Data With Clone (DB2 z/OS)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Extract Data as XML (Oracle, SQL Server, Sybase)

Flashback Table (Oracle)

Flush Tables (MySQL)

Hide Text (Sybase)

Import Data From File (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Lock (DB2 LUW, DB2 z/OS)

Optimize Tables (MySQL)

Place (Sybase)

Rebuild Table (MySQL)

Recompile (SQL Server, Sybase)

Rename (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Reorganize /Reorg (DB2 LUW, Oracle, Sybase)

Repair Tables (MySQL)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Schema (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Select * From (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Set Integrity (DB2 LUW)

Shrink (Oracle)

Transfer Ownership (DB2 LUW)

Truncate (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Update Statistics (DB2 LUW, SQL Server, Sybase)

DBMS platform-specific notes

MySQL	MySQL servers can store tables in multiple formats, including MyISAM and InnoDB. MyISAM tables (ISAM is the acronym for indexed sequential access method) are used most often for read operations. The read operation is very fast, but you cannot include any referential integrity, such as a foreign key. Also, MyISAM tables only issue table-level locks. InnoDB tables, on the other hand, do permit transactions and foreign key constraints. InnoDB tables also lock data at the row level, which is appropriate for high transaction tables. Additional table types available are MERGE, MEMORY, FEDERATED, and ARCHIVE among others.
	For a complete discussion of table types, go to the MySQL documentation of table and engine types. For more information, see Accessing Third Party Documentation.

TABLESPACES

Tablespaces are storage structures that act as partitions for the database. You can create a tablespace to store table data and other objects related to table performance such as indexes or large object data. Tablespaces are used to manage large complex databases. Once you have created a tablespace, you can place objects on it.

> **TIP:** Create separate tablespaces for your tables and indexes and put each tablespace on a different drive or file system. Segmenting tables and their corresponding indexes in this manner helps eliminate I/O contention at the server level.

> **NOTE:** IBM DB2 for Linux, Unix, and Windows lets you assign a location for table or index data directly to physical storage devices. Each tablespace can also be broken down into a collection of containers which are the actual physical storage files or devices. You can then spread the data or database objects across multiple file systems, thereby giving you the necessary space for objects that require it.

Once you have created a tablespace, you can place individual tables and indexes on it. Because tablespaces map to physical drives, you can improve i/o performance by placing tables and their indexes on physically separated table spaces.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓				✓		

Creating and editing

- Tablespace Wizard (DB2 LUW) and Tablespaces Editor (IBM DB2 LUW)

- Tablespace Wizard (Oracle) and Tablespaces Editor (Oracle)

Object actions/operations supported

Change Status (Oracle)

Coalesce (Oracle)

Create Alias (DB2 z/OS)

Drop (DB2 LUW, DB2 z/OS, Oracle)

Extract (DB2 LUW, DB2 z/OS, Oracle)

Rename (DB2 LUW, Oracle)

Reorganize /Reorg (DB2 z/OS)

Report (DB2 LUW, DB2 z/OS, Oracle)

Set Default (Oracle)

Set UNDO (Oracle)

Switch Online (DB2 LUW)

Transfer Ownership (DB2 LUW)

TRIGGERS

Triggers are a special type of procedure that automatically fire when defined data modification operations (insert, update, or delete) occur on a target table or view. Trigger support in Rapid SQL differs by DBMS type, but in general offers options such as:

- Standard INSERT/DELETE/UPDATE event support

- BEFORE/AFTER timing control

- Row/statement trigger type control

- INSTEAD OF triggers

- Enabled/disabled status options

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓			✓	✓	✓

Creating and editing

- Trigger Wizard (DB2 LUW) and Triggers Editor (IBM DB2 LUW)

- Trigger Wizard (DB2 Z/OS) and Triggers Editor (IBM DB2 Z/OS)

- Trigger Wizard (Oracle) and Triggers Editor (Oracle)

- Trigger Wizard (SQL Server) and Triggers Editor (SQL Server)

- Trigger Wizard (Sybase) and Triggers Editor (Sybase)

Object actions/operations supported

Change Status (ITB/FBD, SQL Server, Sybase)

Compile (Oracle)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Hide Text (Sybase)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

TYPES

Types define an abstract data type or object composed of a collection of similar types of data. For example, create an object type that defines a full address rather than the pieces of an address, such as city, state and postal code. An object type stores the pieces of an address in a single type, storing them in the same location and allowing the full address to be accessed and manipulated as single unit rather than multiple units.

Object types are useful for ensuring uniformity and consistency as they are defined as single encapsulated entity that can be reused in other object types and objects. They also offer flexibility by allowing for the creation of objects that represent real-world situations which is limited in relational objects.

You can choose to create a type that is incomplete, complete, a VARRAY, or a nested table or any combination of the above. An incomplete type specifies no attributes and can be used for circular references such as person - female. It lets the type be referenced before it is complete. The VARRAY type can be used to store small sets of related data. For example, if you have ten offices (each one with a different description) at a particular division in your company, you could create a VARRAY of 10 to hold the details of these offices. The values for a VARRAY type must be fixed and known and small values as they are stored in RAW format. A nested table type can be used when data is repeated for the same entity an unknown number of times and storage is a concern.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing
- Object Type Wizard (Oracle) and Types Editor (Oracle)

Object actions/operations supported

Compile   Drop   Extract   Report

TYPE BODIES

Type Bodies implement object type specification by containing the definition of every cursor and subprogram declared in the object type specification. While Type Bodies are listed as a separate object in the Datasource Explorer, they are created on the Types editor in conjunction with Types.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
				✓		

Creating and editing

While listed as separate objects in the Datasource Explorer, type bodies are created and edited on the **Body** tab of the Types Editor. For details, see Types Editor (Oracle).

Object actions/operations supported

Compile   Drop   Extract   Report

UNIQUE KEYS

Unique keys can enforce logical keys that are not chosen as the primary key. They enforce uniqueness for specified columns in a table.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓		✓	✓	✓	✓

Creating and editing

- Unique Key Wizard (DB2 LUW) and Unique Keys Editor (IBM DB2 LUW)

- Unique Key Wizard (DB2 Z/OS) and Unique Keys Editor (IBM DB2 Z/OS)

- Indexes, Primary Keys, or Unique Keys wizard (MySQL) and Indexes, Primary Keys, and Unique Keys editors (MySQL)

- Unique Key Wizard (Oracle) and Unique Keys Editor (Oracle)

- Unique Key Wizard (SQL Server) and Unique Keys Editor (SQL Server)

- Unique Key Wizard (Sybase) and Unique Keys Editor (Sybase)

Object actions/operations supported

Allocate Extent (Oracle)

Analyze (Oracle)

Change Status (Oracle)

Deallocate Unused Space (Oracle)

Disable Index (SQL Server)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Rebuild Index (Oracle, SQL Server)

Rename (DB2 LUW, DB2 z/OS, Oracle, SQL Server, Sybase)

Rename (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Reorganize /Reorg (SQL Server)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, MySQL, Oracle, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

USER DATATYPES

User-defined datatypes promote domain consistency by streamlining the definition of commonly used table columns in a database. You can build a customized datatype from system datatypes and bind defaults and rules to it to enhance integrity. When you reference the user datatype in a column, the column assumes all of the properties of the user datatype.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓				✓	✓

Creating and editing

- User Datatype (DB2 LUW) and User Datatypes Editor (IBM DB2 LUW)

- User Datatype Wizard (DB2 Z/OS) and User Datatypes Editor (IBM DB2 Z/OS)

- User Datatype Wizard (SQL Server) and User Datatypes Editor (SQL Server)

- User Datatype Wizard (Sybase) and User Datatypes Editor (Sybase)

Object actions/operations supported

Drop (DB2 LUW, DB2 z/OS, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, SQL Server, Sybase)

Rename (SQL Server, Sybase)

Report (DB2 LUW, DB2 z/OS, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

USER MESSAGES

User Messages lets you catalog error messages that your database applications can re-use. Microsoft SQL Server stores your error messages in a system table, sysmessages. To return error messages from stored procedures and triggers, you need to call a system stored procedure and pass an error number as a parameter.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
					✓	✓

> **NOTE:** On SQL Server, User Messages can only be accessed from the master database for a specific server.

Creating and editing
- User Message Wizard (SQL Server) and User Messages Editor (SQL Server)
- User Message Wizard (Sybase) and User Messages Editor (Sybase)

Object actions/operations supported

Drop (SQL Server, Sybase)

Extract (SQL Server, Sybase)

Report (SQL Server, Sybase)

USERS

A user is an individual with access to the DBMS.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓		✓	✓	✓	✓

Creating and editing
- User Wizard (DB2 LUW) and Users Editor (IBM DB2 LUW)
- User Wizard (DB2 Z/OS) and Users Editor (IBM DB2 Z/OS)
- Users wizard (MySQL) and Users editor (MySQL)
- User Wizard (Oracle) and Users Editor (Oracle)
- User Wizard (SQL Server) and Users Editor (SQL Server)
- User Wizard (Sybase) and Users Editor (Sybase)

Object actions/operations supported

Analyze (Oracle)

Change Password (DB2 z/OS, MySQL, Oracle)

Compile (Oracle)

Create Like (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Drop (DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Hide Text (Sybase)

Report (DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

Related information

- Microsoft SQL Server Users

- MySQL Users

- Oracle Users

- Sybase ASE Users

MICROSOFT SQL SERVER USERS

Microsoft SQL Server controls access at the database level by requiring the System Administrator or Database Owner to add a login as a database user or alias. After you create a database user, you can implement further security by the granting or revoking the privileges for that user on specific database objects. To consolidate the process of granting or revoking permissions to many users, the database owner can assign users to groups.

MYSQL USERS

Unlike adding new users to other platforms, MySQL doesn't rely solely on a userid/password combination. The MySQL server must not only validate the user and password, but also the host from which the connection is being requested before the new user is admitted to the server's inner sanctum. The Rapid SQL User Wizard and User Editor enable these validations.

ORACLE USERS

To access an Oracle database, you need a user account authenticated with a password. A user account is what Oracle uses to permit access by the user. You can assign the following optional properties to the user:

- Default tablespace

- Temporary tablespace

- Quotas for allocating space in tablespaces

• Profile containing resource limits

SYBASE ASE USERS

Sybase ASE controls access at the database level by requiring the System Administrator or Database Owner to add a login as a database user or alias. After you create a database user, you can implement further security by granting or revoking the privileges for that user on specific database objects. To consolidate the process of granting or revoking permissions to many users, the database owner can assign users to groups.

VIEWS

Views are SQL queries stored in the system catalog that customize the display of data contained in one or more tables. Views behave like tables because you can query views and perform data manipulation operations on them. However, views do not actually store any data. Instead, they depend on data contained in their base tables. Views let you:

• View a customized selection of data from one or more tables. As a result, you can display data more cogently to different sets of users, even though the underlying data is the same.

• Restricting access to a defined set of rows and columns.

Platform availability

DB2 LUW	DB2 z/OS	Interbase/Firebird	MySQL	Oracle	SQL Server	Sybase
✓	✓			✓	✓	✓

Creating and editing
• View Wizard (DB2 LUW) and Views Editor (IBM DB2 LUW)

• View Wizard (DB2 Z/OS) and Views Editor (IBM DB2 Z/OS)

• View Wizard (Oracle) and Views Editor (Oracle)

• View Wizard (SQL Server) and Views Editor (SQL Server)

• View Wizard (Sybase) and Views Editor (Sybase)

Object actions/operations supported
Build Query (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Compile (Oracle)

Create Alias (DB2 LUW, DB2 z/OS)

Create Synonym (DB2 z/OS, Oracle)

Describe (DB2 LUW, Oracle, SQL Server, Sybase)

Drop (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Extract (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Generate Package/Procedure/Statement (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Hide Text (Sybase)

Rename (Oracle, SQL Server, Sybase)

Report (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Schema (DB2 LUW, DB2 z/OS, Oracle, SQL Server, Sybase)

Select * From (DB2 LUW, DB2 z/OS, ITB/FBD, Oracle, SQL Server, Sybase)

Transfer Ownership (DB2 LUW)

Update Statistics (DB2 LUW)

CREATING OBJECTS

Rapid SQL offers easy-to-use wizards and dialog boxes for creating objects. For an introduction to object creation wizards, see Overview and Common Usage of Object Wizards.

Platform-by-platform discussions of wizards are discussed as follows:

- IBM DB2 for Linux, Unix, and Windows Object Wizards

- IBM DB2 for z/OS Object Wizards

- InterBase/Firebird Object Wizards

- Microsoft SQL Server Object Wizards

- MySQL object wizards

- Oracle Object Wizards

- Sybase ASE Object Wizards

OVERVIEW AND COMMON USAGE OF OBJECT WIZARDS

An object wizard lets you create new database and server objects on a datasource. For example, a Tables wizard lets you create columns, set up permissions to work with that table, provide details on physical storage for the table, and so on. Each pane on an object wizard lets you perform a logical task or collection of logical tasks for that object type. For example:
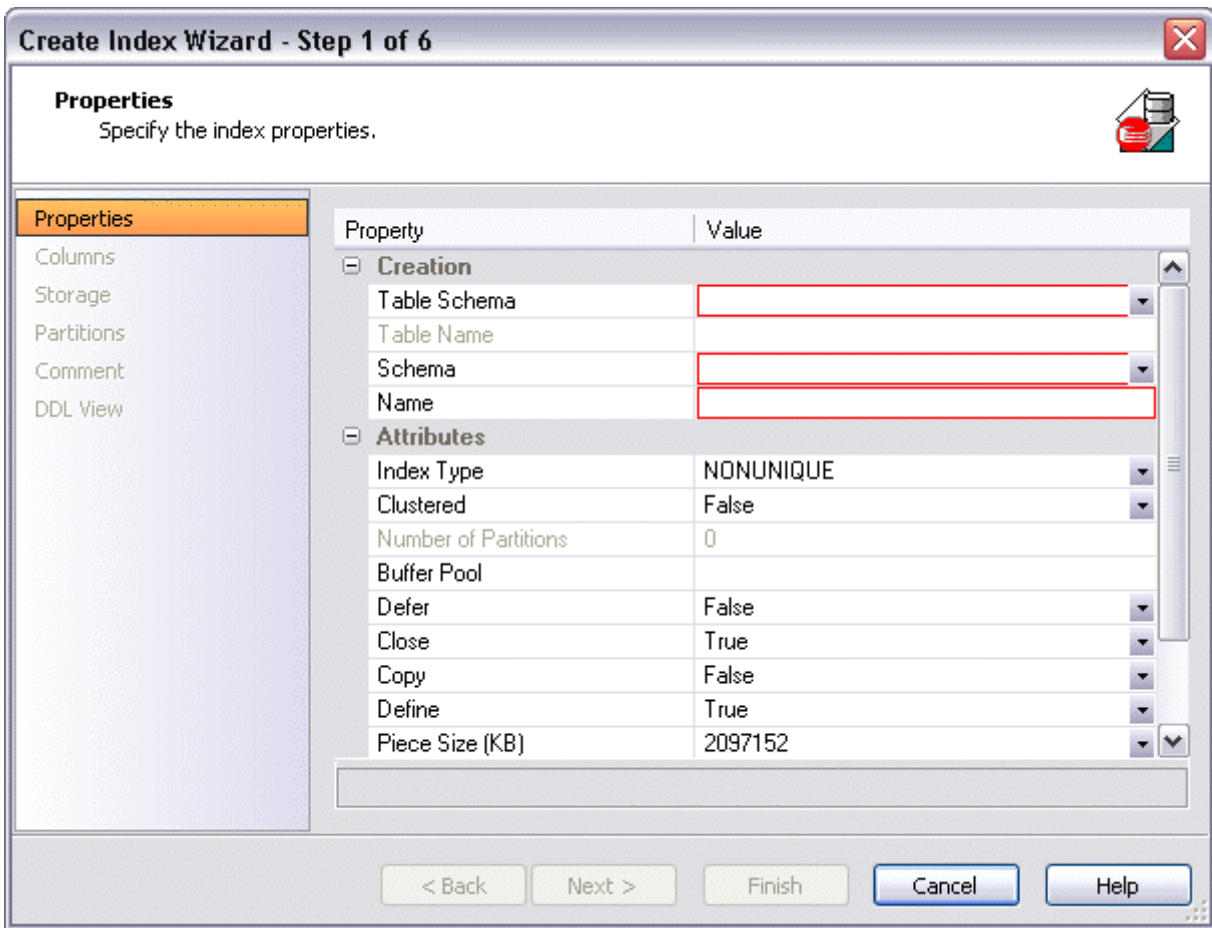
In order to work with object wizards, you must know how to:

- [Open an Object Wizard](#) for a server or database object.

- [Navigate and Set Properties in an Object Wizard](#)

- [Create the New Object](#).

OPEN AN OBJECT WIZARD

Object wizards are accessed from the Datasource Explorer pane of the Rapid SQL main window.

To open an Object wizard for a server or database object

1   Connect to the datasource where you want to create a new resource. For more information, see [Datasource Management](#).

2   On the Datasource Explorer, expand the target datasource.

3   Continue to expand folders under the datasource until the type of object you want to create is visible.

4 On the Datasource Explorer, right-click on the specific type of object that you want to create and select **Create** from the context menu.
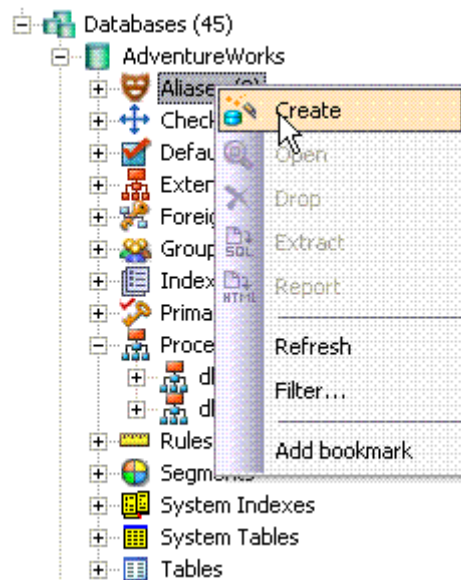
Rapid SQL opens the object wizard for that object type.

NAVIGATE AND SET PROPERTIES IN AN OBJECT WIZARD

When you invoke an object wizard, it opens on the first pane of the wizard, typically a **Properties** pane. As you select options or provide details, you use the navigation buttons at the bottom of the window and the pane controls at the left of the window to navigate through the wizard.

- Use the **Next** button to move to the next pane of the wizard.

  NOTE: In some cases, the **Next** button is not activated until required information is provided. Similarly, some panes of a wizard do not become accessible until activated by choice of an option or property on a previous pane. For example, a **Partitions** tab in a table wizard may only become available if the **Clustered** and **Number of Partitions** options on a prior tab are set accordingly.

- Use the **Back** button to move to the previous pane of the wizard.

- Use the pane controls at the left of the window to move to a specific pane of the wizard.

ADD A COMMENT TO AN OBJECT

The object wizards for certain object types feature a **Comment** tab that lets you add an explanatory note to specific object definitions. Comments are stored in the REMARKS column of the object's system-catalog.

To add a comment to an object:

1 [Open an Object Wizard](#) on an object type that permits comments. See the topics for specific Object editors later in this chapter for information on whether that object type supports comments.

2 Click the **Comment** panel when enabled.
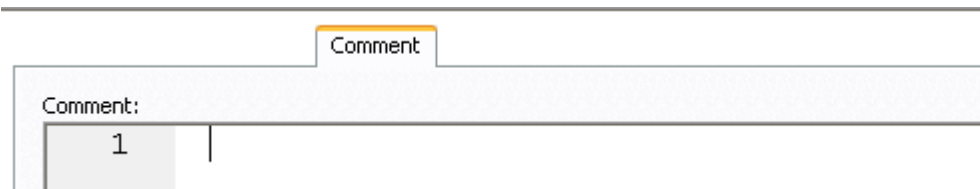
3 In the **Comment** area, type an explanatory note of up to 254 characters long.

SET PERMISSIONS FOR AN OBJECT

When you open an Object wizard to create an object with associated privileges, the **Permissions** (or **Privileges** or **Object Permissions** or **System Permissions**) panel for that editor displays the relevant privileges and lets you make changes accordingly:

To set permissions for an object:

1 [Open an Object Wizard](#) on an object type with associated permissions or privileges. See the topics for specific Object wizards later in this chapter for information on whether that object type supports permissions/privileges.

2 Click the **Permissions** (**Privileges**, **Object Permissions** or **System Permissions**) panel when enabled.

3 For each specific permission to be granted to an entity such as a user, login, or group, select the cell corresponding to the entity and specific permission, and click the **Grant** button. To revoke a privilege, select a cell showing a Granted permission and click **Revoke**.

CREATE THE NEW OBJECT

The final pane or tab of a wizard or editor is most commonly designated **DDL View** or **Definition**.

- A **DDL View** tab/panel lets you view the DDL generated by your selections on the other tabs/panels of the wizard or editor.

- A **Definition** tab/panel is only available for objects such as triggers and procedures, for which you must provide the additional SQL statements that provide the actions taken for that object.

In most cases you arrive at **DDL View** or **Definition** by navigating through panes or tabs, choosing options as you proceed. Alternatively, if you have provided all required information for that object type, you can click **Finish** or **Execute** to create the object.

IBM DB2 FOR LINUX, UNIX, AND WINDOWS OBJECT WIZARDS

Rapid SQL lets you create DB2 LUW objects using the following wizards:

- Alias Wizard (DB2 LUW)

- Database Wizard (IBM DB2 LUW)

- Foreign Key Wizard (IBM DB2 LUW)

- Function Wizard (IBM DB2 LUW)

- Index Wizard (DB2 LUW)

- Materialized Query Table Wizard (DB2 LUW)

- Nodegroup Wizard (DB2 LUW)

- Primary Key Wizard (DB2 LUW)

- Procedure Wizard (DB2 LUW)

- Schema Wizard (DB2 LUW)

- Sequence Wizard (DB2 LUW)

- Structured Type Wizard (DB2 LUW)

- Table Wizard (DB2 LUW)

- Tablespace Wizard (DB2 LUW)

- Trigger Wizard (DB2 LUW)

- Unique Key Wizard (DB2 LUW)

- User Datatype (DB2 LUW)

- User Wizard (DB2 LUW)

- View Wizard (DB2 LUW)

In addition, see Create Synonym.

ALIAS WIZARD (DB2 LUW)

An alias offers you security and convenience so that you can refer to an object without revealing who owns it or what database it belongs to. You can create aliases for tables, views, and even other aliases. The Alias Wizard lets you create an alias without knowing the underlying commands. As you complete the Alias Wizard process, Rapid SQL constructs the necessary CREATE ALIAS statement based on the information that you supply.

To create a new alias using a wizard:

1 Open an Object Wizard for an alias.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details see Aliases (DB2 LUW) - Properties.

- **Comment** panel - for details, see Add a Comment to an Object.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

Aliases (DB2 LUW) - Properties

When creating or editing an alias, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the schema that is to own the alias.
Name	Provide a name for the alias
Target Owner	Select the owner of the object to which you are creating an alias.
Target Type	Select the type of object to which you want to create an alias.
Target Name	Select the specific object to which you are creating an alias.

DATABASE WIZARD (IBM DB2 LUW)

The Database Wizard lets you create a database (a structured collection of data that can be updated or queried) without knowing the underlying commands. Databases can be simple, that is one file with many records and the same fields, or much more complicated with multiple files with different fields.

To open the IBM DB2 for Linux, Unix, and Windows Database Wizard

On the *Registration* tool bar, click New UDB Database.

OR

On the *Datasource Explorer*, right-click an instance node or the Datasources node, and then click New UDB Database.

The tables below describe the fields you may encounter as you complete the wizard.

Naming the New Database

Required Information	Description
What is the name of the database?	Each database requires a unique name. The database name should be between 1 - 8 characters and must begin with a letter A-Z. It's wise to avoid the symbols @, #, and $ if the database will exist in a communications environment or be summoned from an area where those characters do not exist in the language (and hence, on a keyboard).
What is the datasource name for the new database?	Each datasource should have a unique name. This is the name that will appear on the Explorer.
What is the alias name of the database.	OPTIONAL: A database alias is the local synonym given to the database you are creating, and must be unique. If you don't assign an alias, the database name becomes the default alias. Note that a DB2 client can have connections to two different databases with the same name if those databases live on different servers and each database has its own alias.
What is the comment used for the database?	OPTIONAL: Lets you type a comment up to 30 characters. Any comment you enter can be changed later.

Drive Path and Parameters

Required Information	Description
On what drive/path will the database reside?	OPTIONAL: Leave blank if you want Rapid SQL to create the database using the DFTBPATH (default database path configuration) parameter.
What default tablespace parameters should be used?	OPTIONAL Extent size: This is the number of pages of table data that will be written before data is written to the next container. Number of segments: The default is 0. The number you designate specifies the number of extended memory segments available for the database to use. You should only set this number if you have a large amount of real memory.
What global default parameters should the database use?	Territory: The territory where the database was created. This is purely informational, for example, en_US and is related to region-specific support. Codeset: IBM-1252 or UTF-8 are the options for language support. Collating Sequence: Compatibility, Identity, System are your choices for comparing character data. Note: The collation sequence can't be changed after the database has been created.
Finish	Opens the **Preview SQL** dialog box.

Add Catalog/Temporary/User Containers

There are three distinct wizard panels, one for each container type, that enable you to customize your database. Click Next or Finish depending on how you want to proceed.

Required Information	Description
What container(s) will be used to store the catalog/temporary tables?	Use System Managed Space: If you select this option, the operating system's file system manager allocates and manages the space where the catalog tables are stored. This is the default.
	Use Database Managed Space: If you select this option, the database manager controls the storage space for catalog tables.
	Add or Edit - For details, see Add/Edit Container for Tablespace.
What optional default storage parameters should be used.	Optionally identify values for extent size, pre-fetch size, transfer rate, and overhead.
Finish	Opens the **Preview SQ**L dialog box.

ADD/EDIT CONTAINER FOR TABLESPACE

The table below describes the options and functionality on this dialog box:

Required Information	Description
Container Parameters	These parameters are enabled for modification only if you have opted to use database managed space. When enabled, you can either indicate file size parameters or specify device information.
Directory	Identify the directory where the catalog table container will be stored.
File Name	Identify a name for the file where the catalog table container will be stored.
Device Information	Enable either raw partition and type a value, or raw drive, and select a value.

FOREIGN KEY WIZARD (IBM DB2 LUW)

Foreign keys are unique values that refer to specific columns of other tables. Thus, a foreign key links two tables together. Embarcadero Rapid SQL's Foreign Key Wizard makes it easy for you to create a relational link between two tables, thereby speeding queries and giving you faster access to data.The Foreign Key Wizard lets you create a foreign key without knowing the underlying commands.

To create a new foreign key using a wizard:

1   <u>Open an Object Wizard</u> for a foreign key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see <u>Foreign Keys (DB2 LUW) - Properties</u>

- **Column Mapping** panel - for details, see <u>Foreign Keys (DB2 LUW) - Column Mapping</u>

- **Comment** panel - for details, see <u>Add a Comment to an Object</u>.

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

Foreign Keys (DB2 LUW) - Properties

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema	The schema owning the table where the foreign key is being created.
Table Name	This is the table where the foreign key link originates--the child table.
Name	Lets you select a constraint name.
	System Generated Name - DB2 automatically generates a name.
	User Specified Constraint Name - You type the name.
Enabled	TRUE enables the Foreign Key while FALSE disables the Foreign Key.
Delete Rule	Select an action:
	NO ACTION - ensures that referenced values cannot be updated or deleted if to do so would violate referential integrity.
	CASCADE permits a referenced row in a child table to be deleted/updated if it is deleted/updated in the parent table. A row in the child table is SET NULL when rows in the parent table are deleted/updated.
	RESTRICT prevents a command from executing if changes to referential data prompts an error.
Update Rule	Select an action.
	NO ACTION - ensures that referenced values cannot be updated or deleted if to do so would violate referential integrity.
	RESTRICT - prevents a command from executing if changes to referential data prompts an error

Foreign Keys (DB2 LUW) - Column Mapping

1   Under **Referenced Table**, choose the **Owner** and then the **Name** of the referenced, or parent, table.

2   Under the **Main Table**, select check boxes corresponding to the columns that are to reference columns in the referenced table.

FUNCTION WIZARD (IBM DB2 LUW)

To create a relationship between one set of values and another, Rapid SQL offers the Function Wizard. You can develop reusable subroutines so you can control, access, and manipulate the data that underlies an object. As you complete the Function Wizard process, Rapid SQL constructs the necessary CREATE FUNCTION statement based on the information that you supply. The Function Wizard lets you create a function without knowing the underlying commands.

> **NOTE:** To create a user-defined function, you need CREATE ANY privileges or IMPLICIT_SCHEMA authority on the database if the schema does not already exist.

To create a new function using a wizard:

1   Open an Object Wizard for a function.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Functions (DB2 LUW) - Properties.

- **Advanced** panel - for details, see Functions (DB2 LUW) - Advanced.

- **Source** panel (only available for a **Function Type** of SOURCED) - for details, see Functions (DB2 LUW) - Source.

- **Parameters** panel (only available for a **Function Type** of SQL, SOURCED, or TEMPLATE) - for details, see Functions (DB2 LUW) - Parameters.

- **Return Scalar** panel - for details, see Functions (DB2 LUW) - Return Scalar.

- **Return Columns** panel (only available when you choose a **Function Type** of EXTERNAL TABLE) - for details, see Functions (DB2 LUW) - Return Columns.

- **Body** panel - for details, see Functions (DB2 LUW) - Body.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FUNCTIONS **(DB2 LUW)** - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Schema, **Name**, and **Specific Name**	Let you select the owner of the function, provide a name for the function, and provide the Specific name to be used by some SQL statements and DB2 commands for this function.

Setting	Description
Function Type	Select the type of function:
	External Scalar - written in a programming language and returns a scalar value.
	External Table - written in a programming language and returns a complete table.
	OLEDB - accesses OLE DB data in user-defined OLE DB external tables.
	Sourced - another function is invoked to implement the function you are creating.
	SQL Language - written in SQL and returns a table, scalar value, or single row.
	Template: - this is a partial function and can only be invoked from a federated datasource.
	For more information, see [About function types](#).
Language	If you chose a **Function Type** of EXTERNAL SCALAR or EXTERNAL TABLE, specify a language of C, JAVA, or OLE. For more information, see [About function types](#).
Return Type	For a **Function Type** of SQL, select ROW, TABLE, or SCALAR. For other **Function Type** choices, this indicates the default return type for that choice.
External Name	Provide the External Name of the function.
SQL Access Level	Indicates whether the function can execute SQL statements.
	CONTAINS SQL: Statements that don't read or modify SQL can be executed.
	NO SQL: No SQL statements can be executed.
	READS SQL: Statements that cannot modify SQL can be executed.

Functions (DB2 LUW) - Advanced

This tab is only available after clicking the **Advanced** button on the Function wizard's **Properties** panel. It lets you work with the **Threadsafe**, **Fenced**, **Scratchpad**, **Scratchpad Length**, **Allow Parallel**, **Final Call**, **Parameter Style**, **Inherit Special Registers**, **DBINFO**, **Deterministic**, **External Action**, **Called on Null Input**, and **Parameter CCSID** properties.

Functions (DB2 LUW) - Source

> **NOTE:** This panel is only available for a **Function Type** of SOURCED.

Select the **Schema**, **Name**, and **Specific Name** of the source function. Function **Parameters** and **Return Type** for the selected function are displayed. For more information, see [About function types](#).

Functions (DB2 LUW) - Parameters

> **NOTE:** This panel is only available for a **Function Type** of SQL, SOURCED, or TEMPLATE.

For each parameter for this function, use the New button to add a new parameter, provide a name for the parameter, and in the **Attributes** area, select a **Type**, and if appropriate, the **Precision**, **Scale**, and **Size** options.

Functions (DB2 LUW) - Return Scalar

Under **Return Datatype**, select a Type and depending on your choice, provide or select **Precision**, **Scale**, **Size**, and **As Locator** options.

To make use of a CAST FROM clause, under **Cast Datatype** set **Enabled** to True, select a **Type**, and if appropriate, the **Scale**, **Size**, and **As Locator** options

Functions (DB2 LUW) - Return Columns

> **NOTE:** This panel is only available when you choose a **Function Type** of EXTERNAL TABLE.

For each column returned by this function, use the New button to add a new parameter, provide a name for the parameter, and in the **Attributes** area, select a **Type**, and if appropriate, the **Precision**, **Scale**, **Size**, and **As Locator** options.

Functions (DB2 LUW) - Body

Enter the return statement for the function.

ABOUT FUNCTION TYPES

External Scalar/Table/OLE DB Function

External scalar user-defined functions are implemented in an external programming language. The functions are executed on the server and can read SQL data but cannot make changes to the data. These functions are often used to extend the set of built-in functions for DB2, perform logic inside a SQL query that SQL can't perform on its own, and, encapsulate a scalar query that is often used as a subquery in SQL statements, for example, if given an ingredient, search a table for a recipe that uses that ingredient.

When specifying the **Specific Name** for one of these function types:

- If you are using C language, specify the full library path and the function name, otherwise IBM DB2 Database Manager assumes the function is under the IBM DB2 library.

- If you are using Java script, specify the Class ID and the function name, otherwise IBM DB2 Database Manager assumes the function is under the IBM DB2 library.

- If you are using OLE language, specify the full library path and the function name, otherwise IBM DB2 Database Manager assumes the function is under the IBM DB2 library.

Sourced Functions

When you create a sourced function, the new function you are creating will be implemented by a preexisting (source) function that's known to the database manager. The source function can be a built-in function or a previously created user-defined scalar function.

When you select the appropriate schema, you are really specifying an implicit schema privilege. In other words, you're selecting a schema/function that belongs to a user with DBAdmin privileges. If you want to use a built-in function, you must specify the function's specific name.

SQL Language Function

The function you are creating is written in SQL. A table, scalar value, or single row is returned.

Template Function

A template function is better thought of as a template for a function. It's a partial function that has no executable code. You create a function template for mapping it to a datasource function. When the mapping is created, you can specify the function template be used in queries submitted to the federated server. When the query is processed, the federated server invokes datasource function where the template is mapped and returns the appropriate values.

INDEX WIZARD (DB2 LUW)

Comparable to an index in a book, an index gives you speedy access to particular records in a table. The Index Wizard lets you create an index without knowing the underlying commands.

To create a new index using a wizard:

1   Open an Object Wizard for an index.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Indexes (DB2 LUW) - Properties.

   • **Columns** and **Include Columns** panels - for details, see Indexes (DB2 LUW) - Columns and Include Columns.

   • **Comment** panel - for details, see Add a Comment to an Object.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

INDEXES (DB2 LUW) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table in which the index is being created.
Schema and **Name**	Choose the owner and name of the index being created.
Index Type	Index enforces uniqueness on the values of the table's index key.
Clustered	Specifies that the index is the clustering index of the table. The cluster factor of a clustering index is maintained or improved dynamically as data is inserted into the associated table, by attempting to insert new rows physically close to the rows for which the key values of this index are in the same range.

Setting	Description
Percent Free	Lets you type or select the percentage of each index page to leave as free space when building the index, from 0 to 99.

INDEXES (DB2 LUW) - COLUMNS AND INCLUDE COLUMNS

Index columns can be segregated into unique key columns (**Columns** pane) and Include columns that are to be part of the index but do not form part of the unique key.

The steps in completing the panes for the two column types are identical.

- From the **Column** dropdown, select a column for the index and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

MATERIALIZED QUERY TABLE WIZARD (DB2 LUW)

A materialized query table (MQT) is a table based on the result of a query. An MQT contains information that is summarized from other tables and can save time when processing dynamic SQL queries. The Materialized Query Table Wizard lets you create a table without knowing the underlying commands.

To Open the Materialized Query Wizard

1   On the Explorer, find the datasource where you want to add the new materialized query table.

2   Expand the Schema branch, right-click Materialized Query Tables, and select New.

The table that follows describes the fields you may encounter as you complete the Materialized Query Table Wizard.

> NOTE: These options are only available if the tablespace you selected is a database managed tablespace.

Required Information	Description
Who owns the table?	Choose the owner of the table you're creating from the drop-down list.
What is the name of the table?	Type the name of the materialized query table you are creating.
Select a tablespace on which to place the table:	OPTIONAL: No selection is the default. But you can select a tablespace that belongs to the new table's owner if you want.
Specify separate tablespaces for index and long data	OPTIONAL: Lets you separate indexes or long data from the table. Indexes Long data

Required Information	Description
Specify the query on which this table is based	Write the query you want to use to create the table. Note: Every select list element must have a name (use the AS clause for expressions)
Add the columns belonging to this table Add the columns belonging to the partition key	Click **Add**, **Insert** or **Edit** to add or modify table columns. Click **Drop** to delete a selected column.
Do you want the table replicated across database partitions?	The default is no, but check the box if you do want to replicate the table. Replicated materialized query tables can help you improve query performance by enabling collocation between tables. Replication is especially helpful when you have joins between large fact tables and small dimension tables. It's best if tables that are to be replicated are infrequently updated.
Definition Only	Lets you select definition options, Include Column Defaults and Include Identity Column Attributes. When you select the Definition Only option, the new table is treated as if it was a base table rather than a MQT. After you complete the wizard, Rapid SQL opens the Tables Editor.
Refreshable	Lets you select refresh options: Immediate: The table you are creating will be refreshed automatically when changes are made to the base table(s). Deferred: Static SQL will not be optimized. Changes to the base table(s) will not be reflected. Query Optimization: Enabled: Queries will be routed to the MQT. Disabled: This is the default. Maintained by: System: This is the default User After you complete the wizard, Rapid SQL opens the Material Query Tables Editor.
Would you like extra information regarding SQL changes to this table to be written to the log?	If you opted to replicate the table, you must make a selection here. Yes means you want to receive error messages issued by the DB2 replication programs. Include Longvar Columns means you want to receive error messages for these columns if long varchar data is a part of the table you're replicating. No
Would you like to skip logging changes made to this table by Insert... and Alter Table operations in the same unit of work in which this table is created?	Initially Not Logged: This is an option that can reduce logging and increase performance, but also means that you cannot recover the table when rolling forward. Logged Initially. This is the default.
What type of table lock would you like when it is being accessed?	Row: This is the default. During a table or index scan, DB2 locks each row that is scanned before determining whether that row is relevant to the query. Table: During a table scan, DB2 locks the table so no data can be added or altered while the query is executed.

Required Information	Description
What percent of free space to leave for load and reorganization?	-1 is the default.
Do you want data to append to the end of the table?	Yes No: This is the default
Do you want the access plan to this table to be based on existing statistics and optimization level?	Volatile: A volatile table's contents can vary from empty to huge at run time and can render collected statistics inaccurate. Not Volatile: This is the default.
Enter a comment	Optional

NODEGROUP WIZARD (DB2 LUW)

A nodegroup is a named subset of one or more database partitions. The Node Group Wizard lets you create a node group without knowing the underlying commands. When you create a nodegroup, the wizard simply asks you to name the nodegroup and select the partitions to include in the nodegroup.

To Open the Nodegroup Wizard

1 On the Explorer, find the datasource where you want to add the new Nodegroup.

2 Expand the Storage branch, right-click Nodegroups, and select New.

PRIMARY KEY WIZARD (DB2 LUW)

A primary key is a column or group of columns that you can use to identify or access one or more specific rows in a table. A primary key is 'constrained' in that no values can be null and no two values are equal. You can only create one primary key for any table. The Create Primary Key Constraint dialog box lets you create a primary key without knowing the underlying commands.

When you create a primary key, specify the table owner and the table on which you want to place the primary key constraint. You name the constraint and select the column(s) you want to include.

To create a new primary key using a wizard:

1 [Open an Object Wizard](#) for a primary key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Primary Keys (DB2 LUW) - Properties](#).

- **Columns** panel - for details, see [Primary Keys (DB2 LUW) - Columns](#).

- **Comment** panel - for details, see [Add a Comment to an Object](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

PRIMARY KEYS (DB2 LUW) - PROPERTIES

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table in which the primary key is being created.
Name	Choose the owner and name of the primary key being created.

PRIMARY KEYS (DB2 LUW) - COLUMNS

From the **Column** dropdown, select a column for the primary key and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

PROCEDURE WIZARD (DB2 LUW)

Procedures are a reusable block of PL/SQL, stored in the database, that applications can call. Procedures streamline code development, debugging, and maintenance by being reusable. Procedures enhance database security by letting you write procedures granting users execution privileges to tables rather than letting them access tables directly.

To create a new procedure using a wizard:

1   <u>Open an Object Wizard</u> for a procedure.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see <u>Procedures (DB2 LUW) - Properties</u>.

- **Advanced** panel - for details, see <u>Procedures (DB2 LUW) - Advanced</u>.

- **Parameters** panel - for details, see <u>Procedures (DB2 LUW) - Parameters</u>.

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

PROCEDURES (DB2 LUW) - PROPERTIES

When creating or editing a procedure, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the owner for the procedure.
Name	Provide the name of the function.
Specific Name	Optionally, provide the unique name of the procedure.
Language	Select among C, JAVA, COBOL, OLE, or SQL. The database manager will call the procedure accordingly.
External Name	Provide the external name of the procedure.
SQL Access Level	Select an option:
	MODIFIES SQL DATA - the procedure can support any SQL statement except those that cannot be supported in procedures.
	CONTAINS SQL DATA - only SQL statements that neither modify nor read SQL data can be executed in the procedure.
	READS SQL DATA - some SQL statements that don't modify SQL data can be included in the procedure.

PROCEDURES (DB2 LUW) - ADVANCED

When creating or editing a procedure, this tab/panel lets you work with the following settings:

Setting	Description
Results Sets	Indicate the estimated upper bound of returned result sets. 0 is the default.
External Action	Select the External Action option.
New Save Point	Lets you specify a NEW SAVEPOINT LEVEL clause for the procedure.
Threadsafe	Specify whether the procedure is safe to run within the same process as other routines.

Setting	Description
Fenced	If you select yes, you are saying you do not want the procedure to run in the manager operating system environment. This means the database management system will protect its internal resources from the procedure. This option can affect the procedure's operation.
	To run a procedure as not fenced, or a No selection, you must have SYSADMIN or DBADMIN privileges because of the potential to compromise data if the procedure has not been adequately tested.
Parameter Style	Lets you select an option: DB2DARI, DB2GENERAL, DB2SQL, GENERAL, GENERAL WITH NULLS, JAVA, and SQL.
	DB2GENERAL is for Java Language only.
	DB2SQL is for C, COBOL, or OLE Language only.
	GENERAL is for C Language only.
	GENERAL WITH NULLS is for C or COBOL Language only.
	JAVA is for Java Language only.
	SQL is for C, COBOL, or OLE Language only.
Program Type	MAIN: valid for C or COBOL Language and Parameter Style GENERAL, GENERAL WITH NULLS, SQL, or DB2SQL only. In this case, parameters will be passed as an argument counter or argument vector.
	SUBROUTINE: the procedure expects the parameters to be passed as separate arguments.
Inherit Special Registers	Lets you specify this optional clause dictating that the procedure will inherit initial values from the environment of the invoking statement.
DBINFO	Specific information contains such information as the database name, application ID, database code page, and so on.
Deterministic	Enabling this feature specifies the procedure will always return the same result for given argument values. Disabling it means there are state values that affect the results and so the same result will not always be returned when you use identical inputs to invoke the procedure.
Parameter CCSID	Select an encoding scheme of ASCII, UNICODE, or NONE for character or graphic string parameters.

PROCEDURES (DB2 LUW) - PARAMETERS

For each parameter for this function, use the New button to add a new parameter, provide a name for the parameter, and in the **Attributes** area, select a **Type**, specify a **Parameter Mode** of INPUT, OUTPUT, or INPUT_OUTPUT, and if appropriate, the **Precision**, **Scale**, and **Size** options.

SEQUENCE WIZARD (DB2 LUW)

A sequence allows the automatic generation of values, something well-suited to the generation of unique key values. Sequences are not tied to particular table columns. The Sequence Wizard lets you create a sequence without knowing the underlying commands. As you complete the Sequence Wizard, Rapid SQL constructs the necessary CREATE SEQUENCE statement from the information that you have supplied. When finished, you can instruct Rapid SQL to compile the sequence on the target Rapid SQL database or to write a script file containing the CREATE SEQUENCE statement.

The Sequence Wizard lets you:

- Specify the name and owner of the sequence.

- Set both the value of the sequence, and an interval and ranges for incrementing it.

- Cache the sequence, cycle the sequence when it reaches its minimum or maximum values, and guarantee that Rapid SQL generates sequence numbers in the order of request.

    NOTE: To create a sequence, it must belong to your schema or you need CREATE SEQUENCE privileges.

To Open the Sequence Wizard

1   On the Explorer, find the datasource where you want to add the new Sequence.

2   Expand the Schema branch, right-click Sequences, and select New.

The table that follows describes what you may encounter as you complete the Sequence Wizard:

Required Information	Description
Who owns the sequence?	You decide.
What is the sequence name?	Your choice.
What numeric datatype should the Sequence use?	Choose among BIGINT (big integer), decimal (choose width as well), integer, small integer.
What is the first sequence number to be generated?	Starting with 1 is the default.
What is the interval between sequence numbers?	Increment by 1 is the default.
What is the sequence's minimum value?	Choose none or set a value
What is the sequence's maximum value?	Choose none or set a value
Should DB2 preallocate sequence numbers and cache them for faster access?	Preallocating and storing values in the cache reduces synchronous I/O to the log when values are generated for the sequence. If Yes, give number of values No
Should the sequence continue to generate values after reaching either its maximum or minimum value?	Lets you make the sequence cycle and continue to generate numbers. Yes or No
Should the sequence numbers be generated in the order of request?	Select to generate sequence numbers in the order of request. The ORDER option is useful when you are using the sequence number as a timestamp. Yes or No

SCHEMA WIZARD (DB2 LUW)

The Schema Wizard lets you create the structure of a database system including database objects.

To Open the Schema Wizard

1   On the Explorer, find the datasource where you want to add the new Schema.

2   Right-click Schema and select New.

All you need to do when the single-panel wizard opens is to give a unique name to the schema you're creating.

STRUCTURED TYPE WIZARD (DB2 LUW)

The Structured Type wizard lets you create the object type specification and define attributes for a structured type. You then use the Structured Type editor to define the methods and body for the type.

To Open the Type Wizard

1   On the Explorer, find the datasource where you want to add the new Structured Type.

2   Expand the Schema branch, right-click Structured Type, and select Create.

3   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Structured Types (DB2 LUW) - Properties.

   • **Attributes** panel - for details, see Structured Types (DB2 LUW) - Attributes.

   • **DDL View** panel - for details, see Create the New Object.

4   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

On completion, the Structured Type editor opens, letting you provide method and body specifications for the structured type. For details, see Structured Types Editor (IBM DB2 LUW).

STRUCTURED TYPES (DB2 LUW) - PROPERTIES

When creating or editing a structured type, this tab/panel lets you work with the following settings:

Setting	Description
Attributes	**Instantiable** - if disabled, no constructor function can be generated, nor can a non-instantiable type be used as the type of a table or view.
	Final Type - enabled, indicates that the structured type cannot be used as a supertype.
	With Function Access - If enabled, all methods of the type you are creating, and those you will create, can be accessed using functional notation. Some applications prefer functional notation over method invocation.
	Without Comparisons - If enabled, indicates that there are no comparison functions supported for instances of the structured type.
	Inline Length - Specifies the maximum size of an instance of a structured type in a column. If the size of a structured type instance is less than the defined maximum, the data is stored inline with the other values in the row. If the size of the structured type is larger than the defined maximum, the structured type data is stored outside of the table (like LOBs).
Supertype	**Supertype Schema** and **Supertype Name** - let you designate this structured type as a subtype by providing details of the owning supertype.
Reference	**Cast (Source as Ref) With** and **Cast (Ref as Source) With** - Lets you name the cast function, although one will be created with the default name of the structured type you are creating. The cast function "casts" a value between the reference type and the representation type in both directions.
	Reference Using, **Size**, **Precision**, and **Scale** - Define the built-in data type used as the underlying data type for the structured type you are creating and all its subtypes.

STRUCTURED TYPES (DB2 LUW) - ATTRIBUTES

For each attribute to be added to the structured type, click the New button, and provide a name for the attribute.

With an attribute selected, you can modify the **Datatype**. Depending on the datatype you choose you can provide the following datatype qualifiers:

- **Width** (decimal only)

- **Scale** (decimal only)

- **Size** (blob, character, clob, dbclob, graphic, varchar, vargraphic only)

- **Allow log** (blob, clob, dbclob only)

- **Allow compact** (blob, clob, dbclob only)

- **For bit data** (character, long varchar, varchar, only)

TABLE WIZARD (DB2 LUW)

The Table Wizard lets you create a table without knowing the underlying commands.

To create a new table using a wizard:

1 [Open an Object Wizard](#) for an table.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [Tables (DB2 LUW) - Properties](#).

   • **Columns** panel - for details, see [Tables (DB2 LUW) - Columns](#).

   • **Partition** panel - for details, see [Tables (DB2 LUW) - Partition](#).

   • **Tablespaces** panel - for details, see [Tables (DB2 LUW) - Tablespaces](#).

   • **Dimension** panel - for details, see [Tables (DB2 LUW) - Dimensions](#).

   • **Distribution Key Columns** panel - for details, see [Tables (DB2 LUW) - Distribution Key Columns](#).

   • **Indexes** panel - for details, see [Tables (DB2 LUW) - Indexes](#).

   • **Constraints** panel - for details, see [Tables (DB2 LUW) - Constraints](#).

   • **Comment** panel - for details, see [Add a Comment to an Object](#).

   • **Permissions** panel - [Set Permissions for an Object](#)

   • **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TABLES (DB2 LUW) - PROPERTIES
When creating or editing a table, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the schema that is to own the table.
Name	Provide a name for the table.
Percent Free	
Lock Size	The table-level lock prevents concurrently operating applications from using or changing a table. When scanning a table for a query, a row-level lock locks the row when it is being assessed.
Append Data	Enable or Disable appending data to the end of the table.
Volatile	Enable this feature if a table contents may fluctuate from empty to very large. The access plan will not depend on the existing statistics for that table.
Compress	Enable or disable value compression
Row Compression	Enable or disable row compression.
Security Policy	Lets you add a security policy to a table.
RestrictDrop	Corresponds to the DB2 Restrict on Drop attribute.
Log Index Build	Enables this level of logging when creating, recreating, or reorganizing an index.

Setting	Description
CCSID	Specify ASCII or UNICODE or leave unspecified. If specified, this is the encoding scheme for string data. If unspecified, CCSID default encoding is used.
Tablespace, Index Tablespace, and Long Data Tablespace	Select a tablespace, an index tablespace, and a tablespace for Long or LOB table columns.
Do not initially log	If enabled, all changes to the table will be flushed out at commit time. This also means that if a statement fails, the unit of work will rollback. If you are concerned about recoverability, disable this feature.
Data Capture	Specify additional information logged by selecting DATACAPTURE NONE, DATA CAPTURE CHANGES, or DATA CAPTURE CHANGES INCLUDE LONGVAR.

TABLES (DB2 LUW) - COLUMNS

For each column in the table, click the New button to create a column and provide a name for the column. Then, in the **Column Attributes** area, provide details for the column.

Use the Delete button to drop a selected column.

TABLES (DB2 LUW) - PARTITION

Under **Partition Columns**, for each partition column, click the New button and then choose a column name from the dropdown. To add a **Data Partition**, click the New button to open a dialog that lets you add a partition.

TABLES (DB2 LUW) - TABLESPACES

For each Data Tablespace or Long Tablespace, click the New button and then choose a tablespace from the dropdown. To specify an Index Tablespace, select a tablespace from the dropdown.

TABLES (DB2 LUW) - DIMENSIONS

For each column that is to make up a dimension, click the New button to open a dialog that lets you add a column.

TABLES (DB2 LUW) - DISTRIBUTION KEY COLUMNS

For each column that is to make up the distribution key, click the New button and then select a column from the dropdown

TABLES (DB2 LUW) - INDEXES

Click **Add** to open the Index wizard. For more information, see Index Wizard (DB2 LUW).

TABLES (DB2 LUW) - CONSTRAINTS

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- [Primary Key Wizard (DB2 LUW)](#)

- [Unique Key Wizard (DB2 Z/OS)](#)

- [Foreign Key Wizard (DB2 Z/OS))](#)

TABLESPACE WIZARD (DB2 LUW)

Tablespaces establish connections between the physical storage devices of your database system and the logical containers or tables being use to store data. In essence, a tablespace is a storage structure that can hold tables, indexes, large objects, and long data. The Tablespace Wizard lets you create a tablespace without knowing the underlying commands.

To create a new tablespace using a wizard:

1 [Open an Object Wizard](#) for a tablespace.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Tablespaces (DB2 LUW) - Properties](#).

- **Container** panel - for details, see [Tablespaces (DB2 LUW) - Container](#).

- **Comment** panel - for details, see [Add a Comment to an Object](#).

- **Permissions** panel - [Set Permissions for an Object](#)

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TABLESPACES (DB2 LUW) - PROPERTIES

When creating or editing a tablespace, this tab/panel lets you work with the following settings:

Setting	Description
Tablespace properties	**Type** - Select REGULAR, LARGE, TEMPORARY, or USER TEMPORARY.
	Use Automatic Storage and **Managed By** let you specify whether storage is managed automatically, by the database, or by the system.
	Database Partition Group - lets you select a database partition group.
	Buffer Pool - lets you select a buffer pool.
	Drop Recovery - For REGULAR type tablespaces, lets you enable/disable drop recovery.
Performance properties	This group lets you specify or select the **Page SIze**, **Extent Size**, **Prefetch Automatic**, **Prefetch Size**, **Overhead**, **Transfer Rate**, and **File System Caching** properties.

Setting	Description
Automatic Storage properties	This group lets you specify or select the **AutoResize**, **Initial Size**, **Increase Size**, **Max Size Unlimited**, and **Max Size** attributes.

TABLESPACES (DB2 LUW) - CONTAINER

For each container in the tablespace, in the **Container Properties** area, provide the following container properties: **Database Partitions**, **Type** (FILE or DEVICE), **Name**, and **Size**, and then click the **New** button.

Use the **Delete** button to drop a selected container.

TRIGGER WIZARD (DB2 LUW)

A trigger defines a set of actions that take place in conjunction with, or are triggered, for a particular base table, with an insert, update, or delete statement. Triggers are dandy ways to validate input data, read from other tables for cross-referencing purposes, and other similar purposes. The Trigger Wizard lets you create a trigger without requiring you to know any of the underlying commands.

To create a new trigger using a wizard:

1   Open an Object Wizard for a trigger.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Triggers (DB2 LUW) - Properties.

   • **Column Selection** panel - for details, see Triggers (DB2 LUW) - Column Selection.

   • **Definition** panel - for details, see Triggers (DB2 LUW) - Definition.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

TRIGGERS (DB2 LUW) - PROPERTIES

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table for which the trigger is being created.
Schema and **Name**	Choose the owner and name of the trigger being created.

Setting	Description
Trigger Timing	BEFORE: These triggers serve as extensions to the constraint subsystem and are most often used to validate input data, generate values for newly inserted rows, and read from other tables for cross-reference purposes. Note: Before triggers must be created as a For Each Row.
	AFTER: Such a trigger is run after the integrity constraint validations; they can be used to modify operations in the database or be used for activities beyond the database, like supporting an alert notification.
Trigger Events	An INSERT trigger must be associated with an INSERT statement. For example, if a data load operation doesn't include an INSERT statement, the trigger will not be invoked.
	An UPDATE trigger can be associated with specific columns of the base table and will only be activated if those columns are updated.
Trigger Type	**STATEMENT**: (only fires once).
	ROW (fires for each affected row): The trigger runs as many times as there are rows in the affected section. If the set of affected rows is empty, the trigger doesn't run.
Old Table Alias	Type the name of a temporary table of rows as they exist before they're updated or deleted.
New Table Alias	Type a name for a temporary table of rows as they exist after they're inserted or updated.
Old Row Alias	Type a name for the rows as they are before they've been deleted or updated.
New Row Alias	Type a name for the rows as they are after they've been inserted or updated.

TRIGGERS (DB2 LUW) - COLUMN SELECTION

If you chose UPDATE as the **Trigger Event**, select the columns, select the check box beside each column that is to fire the trigger.

TRIGGERS (DB2 LUW) - DEFINITION

Complete the CREATE TRIGGER outline provided by typing or pasting the body of the trigger. Finally, use the **Execute** button to [Create the New Object](#).

UNIQUE KEY WIZARD (DB2 LUW)

A unique key constraint is a key for which no two of its values can be equal and no values can be null. A table can have a number of unique constraints, but it cannot have more than one unique constraint on the same set of columns. If you are creating a unique key constraint on a table that already exists (as opposed to creating a unique key at the time the table is first generated), a unique index must already exist on the columns of the unique key you want to constrain. If no unique index exists, the Index Wizard will open as you complete the Create Unique Key Constraint dialog box.

To create a new unique key using a wizard:

1   <u>Open an Object Wizard</u> for a unique key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see <u>Unique Keys (DB2 LUW) - Properties</u>.

   • **Columns** panel - for details, see <u>Unique Keys (DB2 LUW) - Columns</u>.

   • **Comment** panel - for details, see <u>Add a Comment to an Object</u>.

   • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

UNIQUE KEYS (DB2 LUW) - PROPERTIES

When creating or editing a unique key, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table in which the unique key is being created.
Name	Provide a name for the unique key being created.

UNIQUE KEYS (DB2 LUW) - COLUMNS

From the **Column** dropdown, select a column for the primary key and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

USER DATATYPE (DB2 LUW)

User defined datatypes allow column definitions to be consistent across a database. They let you associate frequently used datatype information to be associated with a specified function and take advantage of performance features available to built-in datatypes including indexing and parallel database queries.

To create a new user datatype using a wizard:

1   <u>Open an Object Wizard</u> for a user datatype.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see <u>User Datatype (DB2 LUW) - Properties</u>.

   • **Comment** panel - for details, see <u>Add a Comment to an Object</u>.

   • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USER DATATYPE (DB2 LUW) - PROPERTIES

When creating or editing a user datatype, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the user datatype.
Datatype	Provide a name for the datatype.
Type	Select the base datatype.
Size	Provide the size of the datatype.
Allow Bit Data	The option is only available for certain datatypes. A check means you want to store the data in a bit format.

USER WIZARD (DB2 LUW)

Users have authorization to use a database and its objects, and the User Wizard gives you an easy way to add new ones.

To create a new user using a wizard:

1   Open an Object Wizard for a user

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Users (DB2 LUW) - Properties.

   • **Object Permissions** and **System Permissions** panels - Set Permissions for an Object

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USERS (DB2 LUW) - PROPERTIES

When creating or editing a user datatype, this tab/panel lets you provide the user **Name**.

VIEW WIZARD (DB2 LUW)

A view gives you an alternative way to look at data in one or more tables. You can customize how you see a table's contents.

To create a new view using a wizard:

1   <u>Open an Object Wizard</u> for a view.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see <u>Views (DB2 LUW) - Properties</u>.

- **Definition** panel - for details, see <u>Views (DB2 LUW) - Definition</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

VIEWS (DB2 LUW) - PROPERTIES

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the owner of the view. The owner of the view must have SELECT privileges for the tables in the CREATE view statement or DBADM authority on the database that contains the table.
Name	Provide a name for the view.
Check Type	**CHECK_NONE** - No search conditions must be satisfied for insert or update operations.
	CHECK_LOCAL - Update and insert operations on view must satisfy the search conditions of the view and underlying views that are defined with a check option. Furthermore, every updatable view that is directly or indirectly defined on view inherits those search conditions (the search conditions of view and all underlying views of that are defined with a check option) as a constraint on insert or update operations.
	CHECK_CASCADED - Update and insert operations on the view must satisfy the search conditions of view and all underlying views, regardless of whether the underlying views were defined with a check option. Furthermore, every updatable view that is directly or indirectly defined on view inherits those search conditions (the search conditions of view and all underlying views) as a constraint on insert or update operations.

Views (DB2 LUW) - Definition

Complete the CREATE VIEW statement by typing or pasting in the relevant query. Finally, use the **Execute button** to <u>Create the New Object</u>.

IBM DB2 FOR Z/OS OBJECT WIZARDS

Rapid SQL lets you create DB2 OS390 objects using the following wizards:

- <u>Alias Wizard (DB2 Z/OS)</u>

- <u>Database Wizard (DB2 Z/OS)</u>

- <u>Foreign Key Wizard (DB2 Z/OS))</u>

- <u>Function Wizard (DB2 Z/OS)</u>

- Index Wizard (DB2 Z/OS)

- Plan Wizard (DB2 Z/OS)

- Primary Key Wizard (DB2 Z/OS)

- Procedure Wizard (DB2 Z/OS)

- Stogroup Wizard (DB2 Z/OS)

- Synonym Wizard (DB2 Z/OS)

- Table Wizard (DB2 Z/OS)

- Tablespace Wizard (DB2 Z/OS)

- Trigger Wizard (DB2 Z/OS)

- Unique Key Wizard (DB2 Z/OS)

- User Datatype Wizard (DB2 Z/OS)

- User Wizard (DB2 Z/OS)

- View Wizard (DB2 Z/OS)

In addition, see Create Synonym.

ALIAS WIZARD (DB2 Z/OS)

An alias offers you security and convenience so that you can refer to an object without revealing who owns it or what database it belongs to. You can create aliases for tables or views. The Alias Wizard lets you create an alias without knowing the underlying commands. As you complete the Alias Wizard process, Rapid SQL constructs the necessary CREATE ALIAS statement based on the information that you supply. To create an alias, you must have CREATE ALIAS privileges or sysadmin or sysctrl authority.

To create a new alias using a wizard:

1   Open an Object Wizard for an alias.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Aliases (DB2 z/OS) - Properties.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

ALIASES (DB2 Z/OS) - PROPERTIES

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the schema that is to own the alias.
Name	Provide a name for the alias.
Target Owner	Select the owner of the object to which you are creating an alias.
Target Type	Select the type of object (TABLE, VIEW) to which you are creating an alias.
Target Name	Select the specific object to which you are creating an alias.

DATABASE WIZARD (DB2 Z/OS)

The Database Wizard lets you create a database (a structured collection of data that can be updated or queried) without knowing the underlying commands. Databases can be simple, that is one file with many records and the same fields, or much more complicated with multiple files with different fields.

To create a new database using a wizard:

1   Open an Object Wizard for a database.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Databases (DB2 z/OS) - Properties.

   • **Permissions** panel - for details, see Set Permissions for an Object

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

DATABASES (DB2 Z/OS) - PROPERTIES

When creating or editing a database, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a unique name for the database.
Type	Workfile: This option in only available if the server is configured in IBM DB2 for OS/390 and z/OS to allow sharing. For more information, contact your System administrator.
	Temp: This option indicates the database is only for declared temporary tables. A temp database cannot be shared.
Group Member	Specifies the member for which this database is being created. Use this only in a shared environment.
Tablespace Buffer Pool	Select the default buffer pool to be used for tablespaces created within the database.

Setting	Description
Index Buffer Pool	Select the default buffer pool name to be used for indexes created within the database.
Storage Group	Select the default storage group to support the DASD space requirements for tablespaces and indexes within the database.
Encoding Scheme	Select an encoding schema of DEFAULT, ASCII, EBCDIC, or UNICODE. NOTE: To change the encoding scheme for a database after it has been created to use a different coded character set identifier (CCSID) that supports the Euro symbol, all data must be unloaded and reloaded. For more information regarding the encoding scheme, contact your System administrator.

FOREIGN KEY WIZARD (DB2 Z/OS))

Foreign keys are unique values that refer to specific columns of other tables. Thus, a foreign key links two tables together. Embarcadero Rapid SQL's Foreign Key Wizard makes it easy for you to create a relational link between two tables, thereby speeding queries and giving you faster access to data.The Foreign Key Wizard lets you create a foreign key without knowing the underlying commands.

To create a new foreign key using a wizard:

1 Open an Object Wizard for a foreign key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Foreign Keys (DB2 z/OS) - Properties.

- **Column Mapping** panel - for details, see Foreign Keys (DB2 z/OS) - Column Mapping.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FOREIGN KEYS (DB2 Z/OS) - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema	Select the owner of the referencing, or child, table.
Table Name	Select the name of the referencing, or child, table.
Name	If you do not want to use the system-generated name, provide a new one.
Delete Rule	Select the action to be taken (NO ACTION, RESTRICT, CASCADE, or SET NULL) when a row of the referenced, or parent, table is deleted.

Under **Referenced Table**, choose the **Owner** and then the **Name** of the referenced, or parent, table.

Under the **Main Table**, select check boxes corresponding to the columns that are to reference columns in the referenced table. Then, under **Referenced Table**, select the corresponding column check boxes.

FUNCTION WIZARD (DB2 Z/OS)

To create a relationship between one set of values and another, Rapid SQL offers the Function Wizard. You can develop reusable subroutines so you can control, access, and manipulate the data that underlies an object. As you complete the Function Wizard process, Rapid SQL constructs the necessary CREATE FUNCTION statement based on the information that you supply.

> **NOTE:** To create a user-defined function, you need CREATE ANY privileges or IMPLICIT_SCHEMA authority on the database if the schema does not already exist.

To create a new function using a wizard:

1 Open an Object Wizard for a function.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, seeFunctions (DB2 z/OS) - Properties.

- **Source** panel - (Only available for a Function Type of SOURCED.) for details, see Functions (DB2 z/OS) - Source.

- **Parameters** panel - for details, see Functions (DB2 z/OS) - Parameters.

- **Return Scalar** panel - for details, see Functions (DB2 z/OS) - Return Scalar.

- **Return Columns** panel - (Only available when you choose a Function Type of EXTERNAL TABLE.) for details, see Functions (DB2 z/OS) - Return Columns.

- **Body** panel - for details, see **Functions (DB2 z/OS) - Body**.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

When creating or editing a function, this tab/panel lets you work with the following settings:

Setting	Description
Schema, **Name**, and **Specific Name**	Select the owner of the function, provide a name for the function, and provide the Specific name to be used by some SQL statements and DB2 commands for this function.

Setting	Description
Function Type	External Scalar: This allows you to extend the function by adding your own or another party's definition for the function.
	External Table: Use this to create a function that is written in ASSEMBLE, C, COBOL, or PLI to return a table after it is deployed.
	Sourced: Here you are creating a function that is based on an existing scalar or table function with an application server.
	SQL: This type of function returns a single value when the function is invoked if the SQL statement that defines it is valid.
Language	If you chose a Function Type of EXTERNAL SCALAR or EXTERNAL TABLE, specify a language of ASSEMBLE, C, COBOL, or PLI.
Return Type	Identifies the return type of the function.
External Name	Provide the External Name of the function.
SQL Access Level	Indicates whether the function can execute SQL statements.
	CONTAINS SQL: Statements that don't read or modify SQL can be executed.
	NO SQL: No SQL statements can be executed.
	READS SQL: Statements that cannot modify SQL can be executed.
WLM Environment	Specify a Workload Management Environment (Required if Language is JAVA/COMPJAVA/ REXX, the Procedure contains a LOB parameter, Security is set to 'USER' or 'DEFINER', or program type is 'SUB').
WLM For Nested	Self-explanatory

FUNCTIONS (DB2 Z/OS) - SOURCE
> **NOTE:**   Only available for a **Function Type** of SOURCED.

Select the **Schema**, **Name**, and **Specific Name** of the source function.

FUNCTIONS (DB2 Z/OS) - PARAMETERS

For each parameter for this function, use the New button to add a new parameter, provide a name for the parameter, and in the **Attributes** area, select a **Type**, and if appropriate, the **Precision**, **Scale**, **Size**, and **As Locator** options.

FUNCTIONS (DB2 Z/OS) - RETURN SCALAR

Under **Return Datatype**, select a Type and depending on your choice, provide or select **Precision**, **Scale**, **Size**, and **As Locator** options.

To make use of a CAST FROM clause, under **Cast Datatype** set **Enabled** to True, select a **Type**, and if appropriate, the **Scale**, **Size**, and **As Locator** options

FUNCTIONS (DB2 Z/OS) - RETURN COLUMNS
> **NOTE:**   Only available when you choose a **Function Type** of EXTERNAL TABLE.

For each column returned by this function, use the New button to add a new parameter, provide a name for the parameter, and in the **Attributes** area, select a **Type**, and if appropriate, the **Precision**, **Scale**, **Size**, and **As Locator** options.

Functions (DB2 z/OS) - Body

Enter the return statement for the function.

INDEX WIZARD (DB2 Z/OS)

Like the index in a book, a database index makes it easier for you to access information quickly. An index lists the location of rows, sorted by the contents of one or more columns.

To create a new index using a wizard:

1 Open an Object Wizard for an index.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Indexes (DB2 z/OS) - Properties

   • **Columns** panel - for details, see Indexes (DB2 z/OS) - Columns

   • **Indexes** panel - for details, see Indexes (DB2 z/OS) - Storage

   • **Partitions** panel - for details, see Indexes (DB2 z/OS) - Partitions

   • **Comment** panel - for details, see Add a Comment to an Object.

   • **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

INDEXES (DB2 Z/OS) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table in which the index is being created.
Schema and **Name**	Choose the owner and name of the index being created.
Index Type	Unique: Prevents the selected table from having two or more rows with the same value of the index key. The uniqueness is enforced at the end of the SQL statement update. Also, null values are treated like any other values, so a column cannot contain more than one null. If you later elect to partition the index, the columns specified for the unique key must include all columns for the partitioning key.
	Non-Unique (default)

Setting	Description
Clustered	Enable or disable clustering. Unless you specifically select the CLUSTER option when you create an index, the first index you create on a table will be bestowed with that distinction. Each table can only have one clustering index at a time.
	The clustering index tells DB2 to insert rows in the table in the order of the clustering key values. Inserted rows will be stored contiguously in sequence when possible. Columns frequently searched with a range of values BETWEEN, less than, greater than, and LIKE, or GROUP BY, etc., are good candidates for a clustering index.
Number of Partitions	If you enabled Clustering, specify the number of partitions.
Buffer Pool	Provide the buffer pool in which this index should reside
Defer, **Close**, **Copy**, and **Define**	Enable or disable these DB2 options as required.
Piece Size	The maximum addressability of the dataset for a non-partitioned index.
Compress	Enabling this setting specifies a COMPRESS=YES parameter when creating the table, resulting in a compressed index. This feature is available if the size of the specified **Buffer Pool** is 8k, 16k, or 32k.
Index on expression	Setting the check box enables the Index on Expression feature. The key expression is then provided on the **Columns** tab/panel. For details, see Indexes (DB2 z/OS) - Columns.
Padded	Setting this check box specifies that the table is to be created with the PADDED keyword.
	This feature must be enabled if you are using index key randomization. The RANDOM setting is applied on a column by column basis on the **Columns** tab/panel.

INDEXES (DB2 Z/OS) - COLUMNS

The actions you take on this tab/panel depend on whether you set the Index on Expression check box on the Properties panel. For more information, see Indexes (DB2 z/OS) - Properties.

- If Index on Expression is disabled:

  From the **Column** dropdown, select a column for the index and specify a **Sort** option.

  NOTE: If you want to use the RANDOM sort option, you must first set the **Padded** check box on the **Properties** tab/panel.

- If Index on Expression is enabled:

  Enter a valid key-expression in the **Expression** field and choose a **Sort** order.

To add more columns/key-expressions, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

INDEXES (DB2 Z/OS) - STORAGE

This tab/panel lets you perform the following tasks:

- Select a dataset management scheme

- Provide associated attribute values

To select a data set management scheme:

1   Click the **Edit** button. The **Data Set Management** dialog opens.

2   Set one of the following data set management options:

   • **DB2 will define and manage the data sets on a volume of the default storage group of the database**

   • **DB2 will define and manage the data sets on a volume of the specified storage group** - Select a storage group (a storage group is made up of disk device volumes): Each data set will be defined on a volume listed in the storage group you select.

   Minimum primary space allocation: 12 kilobytes is the default.

   Minimum secondary space allocation: 12 kilobytes is the default.

   NOTE: NOTE: If the primary and (118 x) secondary space allocations combine to be greater than 2 gigabytes, more than one data set may eventually be used.

   Erase data sets when index dropped? If choose this option, DB2 will overwrite all data with zeros before they are deleted as a security measure.

   • **User will manage the data sets on a specified VCAT catalog-name** - Enter or select the VCAT. Do not select this option for an index on a declared temporary table.

3   Click **OK**.

In addition to the attributes specific to your data set management choice, this tab/panel also offers the following settings:

Free Page	One free page exists for every *x* pages. The *x* specifies how often to leave a page of free space when index entries are created from executing a DB2 utility or creating an index for a table with pre-existing rows. (0-255)
Percent Free	The percentage of free space you want to leave in every page when entries are added to an existing index. The default is 10%.
GBP Cache	This option is available only in a data-sharing environment.
	ALL: As pages are read, all of them will be cached in the group buffer pool.
	CHANGED: Updated pages are cached to the group buffer pool.
	NONE: No pages will be cached.

INDEXES (DB2 Z/OS) - PARTITIONS

Displays the default settings for the number of partitions you specified on the **Properties** pane. Select a partition and click the **Edit** button to modify details for that partition.

PLAN WIZARD (DB2 Z/OS)

A plan, also known as the application plan, is a control structure that is used to process the SQL statements DB2 encounters when it is executing those SQL statements. The Plan Wizard, really, the Bind Plan Wizard, creates the structure that is used in the bind process--the process by which the output from the SQL precompiler is converted into usable form. Some authorization checking is necessary.

To Open the Bind Plan Wizard

1   On the Explorer, find the database where you want to add the new bind plan.

2   On the Plan branch, right-click and select Create.

The Bind Plan Wizard lets you set plan parameters, add packages, and set bind properties. The table below describes the options and functionality on the Bind Plan wizard:

Panel	Settings and tasks	
1	Plan Name	Lets you select the plan name.
	Qualifier	OPTIONAL: Lets you select a qualifier, the plan creator.
	Action	OPTIONAL: Lets you select an action.
	Sql Rules	OPTIONAL: Determines whether you can execute a type 2 CONNECT statement to an existing SQL connection, according to DB2 rules. Lets you select DB2 or STD.
	Cache Size	OPTIONAL: Lets you select or type the cachesize in bytes, the authorization cache acquired in the EDM pool for the plan. At run time, the authorization cache stores user IDs authorized to run. Consulting the cache can avoid a catalog lookup for checking authorization to run the plan.
	Plan Owner	OPTIONAL: Determines the authorization ID of the owner of the plan.
	Current Server	OPTIONAL: Determines the location to connect to before running the plan.
	Resource Acquire	OPTIONAL: Use - Acquires table space locks only when first used by a bound application program. Allocate - Acquires all table space locks when the plan is allocated. The value has no effect on dynamic SQL statements, which always use ACQUIRE(USE).
	Disconnect	OPTIONAL: Determines which remote connections to destroy during commit operations. The option applies to any application process that uses the plan and has remote connections of any type. Regardless of the value of this option, a commit operation destroys all connections in the release pending state. Explicit - Destroy only connections in the release pending state. This value allows you maximum flexibility for controlling remote connections. Automatic - Destroy all remote connections. Conditional - Destroy all remote connections unless an open cursor defined as WITH HOLD is associated with the connection.
2	Lets you select the **Member Name**, **PDS Name** (partitioned data set) and click **Add** to enter each member and PDS name.	
3	Lets you select the **Location** to connect to, the **Collection** (location of the DBMS where the plan binds and where the description of the plan resides.) and a **Package**.	

Panel	Settings and tasks	
4	Isolation	Determines how far to isolate an application from the effects of other running applications.
	Keep Dynamic	Specifies that DB2 keeps dynamic SQL statements after commit points. The application does not need to prepare an SQL statement after every commit point. DB2 keeps the dynamic SQL statement until the application process ends, a rollback operation occurs or the application executes an explicit PREPARE statement with the same statement identifier.
		If the prepared statement cache is active, DB2 keeps a copy of the prepared statement in the cache. If the prepared statement cache is not active, DB2 keeps only the SQL statement string past a commit point. DB2 then implicitly prepares the SQL statement if the application executes an OPEN, EXECUTE, or DESCRIBE operation for that statement.
	Current Data	Determines whether to require data currency for read-only and ambiguous cursors when the isolation level of cursor stability is in effect. It also determines whether block fetching can be used for distributed, ambiguous cursors.
	Degree	Determines whether to attempt to run a query using parallel processing to maximize performance. Lets you select an option.
	Dynamic Rules	Determines what values apply at run time for the following dynamic SQL attributes:
		The authorization ID that is used to check authorization
		The qualifier that is used for unqualified objects
		The source for application programming options that DB2 uses to parse and semantically verify dynamic SQL statements
		Whether dynamic SQL statements can include GRANT, REVOKE, ALTER, CREATE, DROP, and RENAME statements
	Release	Determines when to release resources that a program uses. Options are at each **Commit** point or **Deallocate** when the program terminates.
	Validate	Determines whether to recheck, at run time, errors found during bind. The option has no effect if all objects and needed privileges exist.
		Bind - If not all objects or needed privileges exist at bind time, the wizard displays an error messages, and does not bind the package.
		Run - If not all objects or privileges exist at bind time, the process issues warning messages, but the bind succeeds. DB2 checks existence and authorization again at run time for SQL statements that failed those checks during bind. The checks use the authorization ID of the plan owner.
5	Explain	Obtains information about how SQL statements in the member list of the plan, are to execute, and then inserts that information into the table owner.PLAN_TABLE, where owner is the authorization ID of the owner of the plan or package. This option does not obtain information for statements that access remote objects.
	Reopt(VARS)	Re-determines the access path at run time.
	Prepare	Prepares dynamic SQL statements that refer to remote objects.
	ImmedWrite	Immediate writes will be done for updates made to group buffer pool dependent pagesets or partitions.
	Opthint	Query optimization hints are used for static SQL.
	Encoding	Lets you select type of language for the package.
	Path	Lets you select a path that DB2 uses to resolve unqualified user-defined distinct types, functions, and stored procedure names (in CALL statements).
	Flag	Lets you select all message types or a specified subset to display: informational, warning, error, and completion messages.

Panel	Settings and tasks
6	**Enable** or **Disable** system connection types that can use the plan or package, select a **System** or **Cname** option.

PRIMARY KEY WIZARD (DB2 Z/OS)

A primary key is a unique key that is part of a table's definition. There can be only one primary key for each table, and the columns contained in a primary key cannot have null values. A primary key constraint forbids duplicate values in one or more columns. A table with a primary key will be considered the parent table to a table with a foreign key, which becomes the dependent table.

> **NOTE:** A nullable column cannot be a part of a primary key.

To create a new primary key using a wizard:

1 Open an Object Wizard for a primary key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Primary Keys (DB2 z/OS) - Properties.

- **Columns** panel - for details, see Primary Keys (DB2 z/OS) - Columns.

- **Storage** panel - for details, see Primary Keys - Storage - Edit button (manage datasets) and Primary Keys (DB2 z/OS) - Storage - Attributes.

- **Partitions** panel - for details, see Primary Keys (DB2 z/OS) - Partitions.

- **Comment** panel - for details, see Add a Comment to an Object.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

PRIMARY KEYS (DB2 Z/OS) - PROPERTIES

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table in which the index is being created.
Schema and **Name**	Choose the owner and name of the index being created.
Clustered	Enable or disable clustering.
Number of Partitions	If you enabled Clustering, specify the number of partitions.
Buffer Pool	Provide the buffer pool in which this index should reside

Setting	Description
Defer, **Close**, **Copy**, and **Define**	Enable or disable these DB2 options as required.
Piece Size	The maximum addressability of the dataset for a non-partitioned index.

PRIMARY KEYS (DB2 z/OS) - COLUMNS

From the **Column** dropdown, select a column for the primary key and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

PRIMARY KEYS - STORAGE - EDIT BUTTON (MANAGE DATASETS)

> **NOTE:** Availability differs according to the dataset management options you chose

Choose a data set management option:

DB2 will define and manage the data sets on a volume of the default storage group of the database

DB2 will define and manage the data sets on a volume of the specified storage group

Select a storage group (a storage group is made up of disk device volumes): Each data set will be defined on a volume listed in the storage group you select.

Minimum primary space allocation: 12 kilobytes is the default.

Minimum secondary space allocation: 12 kilobytes is the default.

> **NOTE:** If the primary and (118 x) secondary space allocations combine to be greater than 2 gigabytes, more than one data set may eventually be used.

Erase data sets when index dropped? If you choose this option, DB2 will overwrite all data with zeros before they are deleted as a security measure.

User will manage the data sets on a specified VCAT catalog-name Enter or select the VCAT. Do not select this option for an index on a declared temporary table.

PRIMARY KEYS (DB2 z/OS) - STORAGE - ATTRIBUTES

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Storage Group, **Primary Space Allocation**, **Secondary Space Allocation**, **Erase**, and **VCAT catalog**	The ability to set these options depends on the dataset management options you chose.

Setting	Description
Free Page	One free page exists for every *x* pages. The *x* specifies how often to leave a page of free space when index entries are created from executing a DB2 utility or creating an index for a table with pre-existing rows. (0-255)
Percent Free	The percentage of free space you want to leave in every page when entries are added to an existing index. The default is 10%.
GBP Cache	This option is available only in a data-sharing environment.
	ALL: As pages are read, all of them will be cached in the group buffer pool.
	CHANGED: Updated pages are cached to the group buffer pool.
	NONE: No pages will be cached.

PRIMARY KEYS (DB2 z/OS) - PARTITIONS

Displays the default settings for the number of partitions you specified on the **Properties** pane. Select a partition and click the **Edit** button to modify details for that partition.

PROCEDURE WIZARD (DB2 Z/OS)

Procedures are a reusable block of PL/SQL, stored in the database, that applications can call. Procedures streamline code development, debugging, and maintenance by being reusable. Procedures enhance database security by letting you write procedures granting users execution privileges to tables rather than letting them access tables directly.

To create a new procedure using a wizard:

1   [Open an Object Wizard](#) for a procedure.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Procedures (DB2 z/OS) - Properties](#).

- **Parameters** panel - for details, see [Procedures (DB2 z/OS) - Parameters](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

Important Notes

- If you are creating a SQL routine procedure, you must have the latest DB2 fixpack installed on your OS/390 Client. If you do not have the latest fixpack installed, the SQL routine procedure creation will fail.

PROCEDURES (DB2 Z/OS) - PROPERTIES

When creating or editing a procedure, this tab/panel lets you work with the following settings:

Setting	Description
Schema, **Name**, and **Specific Name**	Select the schema that is to own the procedure, provide a name for the procedure, and provide the Specific Name for the procedure.
Language	The database manager will call the procedure accordingly assuming the program is designed to run in the server's environment.
	Assemble: a stored procedure written in Assembler
	C: a stored procedure written in C or C++
	COBOL: a stored procedure written in COBOL
	CompJAVA: CompJAVA is no longer supported. Stored procedures should alternatively be written in JAVA.
	JAVA
	PLI: A stored procedure written in PL/I.
	REXX (Restructured Extended Executor Language) - Don't use this language when SQL parameter style is in effect. To specify REXX, the general parameter style or general with nulls.
	SQL
SQL Access Level	MODIFIES SQL DATA (Default): The procedure can support any SQL statement except those that cannot be supported in procedures.
	CONTAINS SQL DATA: Only SQL statements that neither modify nor read SQL data can be executed in the procedure.
	READS SQL DATA Some SQL statements that don't modify SQL data can be included in the procedure
	NO SQL: Only SQL statements with a data access classification of NO SQL can be executed. Don't select this option for a JAVA procedure that uses a .jar.
WLM Environment	Specify a Workload Management Environment (Required if Language is JAVA/COMPJAVA/REXX, the Procedure contains a LOB parameter, or Security is set to 'USER' or 'DEFINER').
WLM For Nested	Self-explanatory

PROCEDURES (DB2 Z/OS) - PARAMETERS

For each parameter for this function, use the New button to add a new parameter, provide a name for the parameter, and in the **Attributes** area, select a **Type**, specify a **Parameter Mode** of INPUT, OUTPUT, or INPUT_OUTPUT, and if appropriate, the **Precision**, **Scale**, and **Size** options.

STOGROUP WIZARD (DB2 Z/OS)

Stogroups are storage groups. You create them on the current server. Storage from the named sets of volumes you create can be allocated at a later date for tablespaces or index spaces.

To Open the Stogroup Wizard

1   On the Explorer, find the datasource where you want to add the new Storage Group.

2   Expand the Storage branch, right-click Stogroup, and select New.

The table below describes the fields you will encounter as you complete the Stogroup Wizard.

Required Information	Description
What is the name of the Stogroup?	Enter a name for the storage group.
VCAT	This is the integrated catalog facility catalog, or volume catalog. Name the catalog or choose one from the drop-down list if it is available.
Select the volumes in the Stogroup	Specify a set of volumes that may exist on the system but may not be in use by other storage groups.
Select All	Selects all listed volumes.
Unselect All	Unselects all listed volumes.
Add	Opens the Add Volume Dialog Box.
Remove	Deletes all selected volumes from the list.

ADD VOLUME DIALOG BOX
The table below describes the options and functionality on the Add Volume dialog box:

Required Information	Description
Enter one or more volumes to add to the stogroup	Type the names of the volumes (separated by spaces) to add to the stogroup.
Check	Click to see if any additional information is available about the volumes you typed. Opens the Volumes Info Dialog Box.
Or select volumes for the list	Lets you select volumes.

VOLUMES INFO DIALOG BOX
The table below describes the options and functionality on the Volume Info dialog box:

Required Information	Description
Volumes	Lets you select volumes.
OK	Click to add volumes to Stogroup Wizard.

SYNONYM WIZARD (DB2 Z/OS)

A synonym is an alternative name you can create for a table or view. Using a synonym can make it easier for you to remember that table or view instead of having to think about the possibly cumbersome formal name (for example, a show dog may have a formal name that incorporates all his ancestors, but will answer to the name Spot around the house).

To create a new synonym using a wizard:

1   Open an Object Wizard for a synonym.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Synonyms (DB2 z/OS) - Properties.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

SYNONYMS **(DB2 z/OS)** - PROPERTIES

When creating or editing a synonym, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the schema that is to own the synonym.
Name	Provide a name for the synonym.
Referenced Object Owner	Select the owner of the object to which you are creating a synonym.
Referenced Object Type	Select the type of object (TABLE, VIEW) to which you are creating a synonym.
Referenced Object Name	Select the specific object to which you are creating a synonym.

TABLE WIZARD (DB2 Z/OS)

All data in a database is stored in a tabular format, that is a collection of rows and columns. Tables, therefore are fundamental to whatever database you are administering.

To create a new table using a wizard:

1 <u>Open an Object Wizard</u> for a table.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see <u>Tables (DB2 z/OS) - Properties</u>.

- **Columns** panel - for details, see <u>Tables (DB2 z/OS) - Columns</u>.

- **Indexes** panel - for details, see <u>Tables (DB2 z/OS) - Indexes</u>.

- **Constraints** panel - for details, see <u>Tables (DB2 z/OS) - Constraints</u>.

- **Comment** panel - for details, see <u>Add a Comment to an Object</u>.

- **Permissions** panel - <u>Set Permissions for an Object</u>

- **DDL View** panel - for details, see <u>Create the New Object</u>.

The table that follows describes the fields you may encounter as you complete the Table Wizard.

TABLES (DB2 Z/OS) - PROPERTIES
Select the **Schema**, provide a **Name**, and provide or select other table properties.

TABLES (DB2 Z/OS) - COLUMN**s**
For each column in the table, click the **Add Column** button to create a column, provide a **Name** for the column and provide or select the remaining column attributes.

Use the **Delete** button to drop a selected column.

Use the arrow buttons to reposition the columns.

Tables (DB2 z/OS) - Indexes
Click Add to open the <u>Index Wizard (DB2 Z/OS)</u>.

TABLES (DB2 Z/OS) - CONSTRAINTS
Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- <u>Foreign Key Wizard (DB2 Z/OS))</u>

- <u>Primary Key Wizard (DB2 Z/OS)</u>

- <u>Unique Key Wizard (DB2 Z/OS)</u>

- <u>Create Synonym</u>

TABLESPACE WIZARD (DB2 Z/OS)

Tablespaces establish connections between the physical storage devices of your database system and the logical containers or tables being used to store data. In essence, a tablespace is a storage structure that can hold tables, indexes, large objects, and long data.

> **NOTE:** To change the encoding scheme for a database after it is created to utilize a different coded character set identifier (CCSID) that supports the Euro symbol, all data must be unloaded and reloaded. For more information regarding the encoding scheme, contact your System administrator.

To Open the Tablespace Wizard

1 On the Explorer, find the datasource where you want to add the new Tablespace.

2 Expand the Storage branch, right-click Tablespaces, and select New.

3 Use the following table as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Tablespaces (DB2 z/OS) - Properties](#).

- **Permissions** panel - [Set Permissions for an Object](#)

- **DDL View** panel - for details, see [Create the New Object](#).

4 Finally, use the **Execute** button to create the object.

TABLESPACES **(DB2 z/OS)** - PROPERTIES

When creating or editing a tablespace, this tab/panel lets you work with the following settings:

Group	Settings	Description
Creation	**Name** and **Database**	Provide a name and select an assocaited database.
	Database type	Lets you select a REGULAR or WORKFILE database type.
	Creation type	Lets you select among PARTITIONED, SEGMENTED, RANGE-PARTITIONED UNIVERSAL, PARTITION-BY-GROWTH UNIVERSAL, and LOB options.
		For detailed information on these types and other required settings, see http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp. Search on a term such as **universal tablespaces**.
Tablespace management	**Management type**	Lets you select how you want the tablespace managed: DEFAULT STORAGE GROUP, STORAGE GROUP, or VCAT CATALOG-NAME.
	Storage group, Minimum primary space allocation, Minimum secondary space allocation, and Erase rule	If you selected a **Management type** of STORAGE GROUP, select a storage group, specify minimum space allocations, and specify whether data sets are to be erased when the tablespace is dropped.
	VCatName	If you selected a **Management type** of VCAT catalog-name, select a catalog name.

Group	Settings	Description
Bufferpool	**Buffer Pool**	Select the buffer pool where the tablespace is to reside.
Partitions and size	**Number of partitions**	If you selected a **Creation Type** of PARTITIONED or RANGE-PARTITIONED UNIVERSAL, specify the number of partitions.
	Partition size (DSSIZE)	If you did not select a **Creation Type** of SEGMENTED, select a partition size.
	Segment size	If you selected a **Creation Type** of SEGMENTED, PARTITION-BY GROWTH, or RANGE-PARTITIONED UNIVERSAL, select the number of pages that are to be assigned to each segment of the tablespace.
	Max rows per page	Specify the number of rows (1-255) that can be placed on each data page.
Space Management	**Free space portion of each page (%)**	If you did not select a **Creation Type** of LOB, specify the percentage of each page to be left as free space when a tablespace is reloaded or reorganized.
	Free page frequency	If you did not select a **Creation Type** of LOB, specify how often to leave a free page when a tablespace is reloaded or reorganized.
Other parameters	**GBPCACHE**	Select a group buffer pool cache scheme of CHANGED, ALL, SYSTEM, or NONE.
	Compress	Enable this setting to allow compression.
	Track modified pages	Enable this setting to track modified pages in the space map pages.
	Encoding scheme	Select an encoding scheme of EBCDIC, UNICODE, ASCII, or NONE.
	Log	Enabling this check box specifies that changes to a LOB column are to be written to the log.
	Define	Enabling this check box specifies that tablespace data sets are defined when the tablespace is created. Otherwise, data sets are not created until data is inserted.
	Member Cluster	Enable this setting to manage space for inserts on a member-by-member basis.
	Close rule	Enable this setting to specify CLOSE YES, dictating the data set close priority.
	Lock Size	Specify a lock size of ANY, TABLESPACE, PAGE, or ROW.
	Maximum locks	Specify the number of locks that are allowed before escalating.

TRIGGER WIZARD (DB2 Z/OS)

All data in a database is stored in a tabular format, that is a collection of rows and columns. Tables, therefore are fundamental to whatever database you are administering.

To create a new trigger using a wizard:

1 [Open an Object Wizard](#) for a trigger.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Triggers (DB2 z/OS) - Properties](#).

- **Column Selection** panel - for details, see [Triggers (DB2 z/OS) - Column Selection](#).

- **Definition** panel - for details, see [Triggers (DB2 z/OS) - Definition](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TRIGGERS (DB2 z/OS) - PROPERTIES

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table for which the trigger is being created.
Schema and **Name**	Choose the owner and name of the trigger being created.
Trigger Timing	BEFORE: These triggers serve as extensions to the constraint subsystem and are most often used to validate input data, generate values for newly inserted rows, and read from other tables for cross-reference purposes. Note: Before triggers must be created as a For Each Row.
	AFTER: Such a trigger is run after the integrity constraint validations; they can be used to modify operations in the database or be used for activities beyond the database, like supporting an alert notification.
Trigger Events	An INSERT trigger must be associated with an INSERT statement. For example, if a data load operation doesn't include an INSERT statement, the trigger will not be invoked.
	An UPDATE trigger can be associated with specific columns of the base table and will only be activated if those columns are updated.
Trigger Type	**STATEMENT**: (only fires once).
	ROW (fires for each affected row): The trigger runs as many times as there are rows in the affected section. If the set of affected rows is empty, the trigger doesn't run.
Old Table Alias	Type the name of a temporary table of rows as they exist before they're updated or deleted.
New Table Alias	Type a name for a temporary table of rows as they exist after they're inserted or updated.
Old Row Alias	Type a name for the rows as they are before they've been deleted or updated.
New Row Alias	Type a name for the rows as they are after they've been inserted or updated.

TRIGGERS (DB2 z/OS) - COLUMN SELECTION

If you chose UPDATE as the **Trigger Event**, select the columns, select the check box beside each column that is to fire the trigger.

TRIGGERS (DB2 Z/OS) - DEFINITION

Complete the CREATE TRIGGER outline provided by typing or pasting the body of the trigger. Finally, use the **Execute** button to Create the New Object.

UNIQUE KEY WIZARD (DB2 Z/OS)

A unique key constraint is a key for which no two of its values can be equal and no values can be null. A table can have a number of unique constraints, but it cannot have more than one unique constraint on the same set of columns. If you are creating a unique key constraint on a table that already exists (as opposed to creating a unique key at the time the table is first generated), a unique index must already exist on the columns of the unique key you want to constrain. If no unique index exists, the Index Wizard will open as you complete the Create Unique Key Constraint dialog box.

To create a new unique key using a wizard:

1 Open an Object Wizard for a unique key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Unique Keys (DB2 z/OS) - Properties.

- **Columns** panel - for details, see Unique Keys (DB2 z/OS) - Columns.

- **Storage** panel - for details, see Unique Keys (DB2 z/OS) - Storage - **Edit** button (manage datasets) and Unique Keys (DB2 z/OS) - Storage - Attributes.

- **Partitions** panel - for details, see Unique Keys (DB2 z/OS) - Partitions.

- **Comment** panel - for details, see Add a Comment to an Object.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

UNIQUE KEYS (DB2 Z/OS) - PROPERTIES

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Choose the owner and name of the table in which the unique key is being created.
Schema and **Name**	Choose the owner and name of the index being created.
Clustered	Enable or disable clustering.
Number of Partitions	If you enabled Clustering, specify the number of partitions.
Buffer Pool	Provide the buffer pool in which this unique key should reside

Setting	Description
Defer, **Close**, **Copy**, and **Define**	Enable or disable these DB2 options as required.
Piece Size	The maximum addressability of the dataset for a non-partitioned index.

UNIQUE KEYS (DB2 Z/OS) - COLUMNS

From the **Column** dropdown, select a column for the unique key and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

UNIQUE KEYS (DB2 Z/OS) - STORAGE - **Edit** BUTTON (MANAGE DATASETS)
> **NOTE:**   Availability differs according to the dataset management options you chose

Choose a data set management option:

DB2 will define and manage the data sets on a volume of the default storage group of the database

DB2 will define and manage the data sets on a volume of the specified storage group

Select a storage group (a storage group is made up of disk device volumes): Each data set will be defined on a volume listed in the storage group you select.

Minimum primary space allocation: 12 kilobytes is the default.

Minimum secondary space allocation: 12 kilobytes is the default.

> **NOTE:**   If the primary and (118 x) secondary space allocations combine to be greater than 2 gigabytes, more than one data set may eventually be used.

Erase data sets when index dropped? If you choose this option, DB2 will overwrite all data with zeros before they are deleted as a security measure.

User will manage the data sets on a specified VCAT catalog-name Enter or select the VCAT. Do not select this option for an index on a declared temporary table.

UNIQUE KEYS (DB2 Z/OS) - STORAGE - ATTRIBUTES
> **NOTE:**   Availability differs according to the dataset management options you chose

When creating or editing a unique key, this tab/panel lets you work with the following settings:

Setting	Description
Storage Group, **Primary Space Allocation**, **Secondary Space Allocation**, **Erase**, and **VCAT catalog**	The ability to set these options depends on the dataset management options you chose.
Free Page	One free page exists for every *x* pages. The *x* specifies how often to leave a page of free space when index entries are created from executing a DB2 utility or creating an index for a table with pre-existing rows. (0-255)
Percent Free	The percentage of free space you want to leave in every page when entries are added to an existing index. The default is 10%.
GBP Cache	This option is available only in a data-sharing environment. ALL: As pages are read, all of them will be cached in the group buffer pool. CHANGED: Updated pages are cached to the group buffer pool. NONE: No pages will be cached.

UNIQUE KEYS (DB2 Z/OS) - PARTITIONS
Displays the default settings for the number of partitions you specified on the **Properties** pane. Select a partition and click the **Edit** button to modify details for that partition.

USER DATATYPE WIZARD (DB2 Z/OS)
A datatype is a named set of valid values that can be manipulated by a set of operations. There are intrinsic datatypes, which are predefined and always available, and derived datatypes. A derived datatype is a user-defined datatype, which can include both intrinsic and previously derived datatypes. The User Datatype Wizard lets you create a derived datatype without knowing the underlying commands.

To create a new user datatype using a wizard:
1 [Open an Object Wizard](#) for a user datatype.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [User Datatypes (DB2 z/OS) - Properties](#).

   • **Comment** panel - for details, see [Add a Comment to an Object](#).

   • **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

USER DATATYPES (DB2 Z/OS) - PROPERTIES

When creating or editing a user datatype, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the user datatype.
Datatype	Provide a name for the datatype.
Type	Select the base datatype.
Size	Provide the size of the datatype.
For Data	Select the MIXED, SBCS, or BIT option for the datatype.
CCSID	Select the NONE, ASCII, EBCDIC, or UNICODE option for the datatype.

USER WIZARD (DB2 Z/OS)

Users have authorization to use a database and its objects, and the User Wizard gives you an easy way to add new ones.

To create a new user using a wizard:

1 [Open an Object Wizard](#) for a user

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [Users (DB2 z/OS) - Properties](#).

   • **Object Permissions** and **System Permissions** panels - [Set Permissions for an Object](#)

   • **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

USERS (DB2 Z/OS) - PROPERTIES

When creating or editing a user datatype, this tab/panel lets you provide the user **Name**.

VIEW WIZARD (DB2 Z/OS)

A view gives you a new way of looking at data in a results table. Views behave like tables because you can query views and perform data manipulation operations on them. However, views do not actually store any data. Instead, they depend on data contained in their base tables. Columns added to the base table(s) after the view is created are not included in the result set. Views are thus handy tools for controlling access to a table. You can allow someone to see portions of data without allowing that user to see the table in its entirety. For example, you can create a view that will permit a user to see employee names in a table without allowing access to the Social Security numbers of that same table.

The wizard itself is a single panel. After you complete the wizard, the View Editor opens so you can complete the definition of the view, choose the columns to show in the view, the dependencies, and access privileges to the view.

To create a new view using a wizard:

1   Open an Object Wizard for a view.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

 • **Properties** panel - for details, see Views (DB2 z/OS) - Properties.

 • **Definition** panel - for details, see Views (DB2 z/OS) - Definition.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

VIEWS (DB2 Z/OS) - PROPERTIES

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the view. The owner of the view must have SELECT privileges for the tables in the CREATE view statement or DBADM authority on the database that contains the table.
Name	Provide a name for the view.
Check Type	**CHECK_NONE** - No search conditions must be satisfied for insert or update operations. CHECK_LOCAL - Update and insert operations on view must satisfy the search conditions of the view and underlying views that are defined with a check option. Furthermore, every updatable view that is directly or indirectly defined on view inherits those search conditions (the search conditions of view and all underlying views of that are defined with a check option) as a constraint on insert or update operations. CHECK CASCADED - Update and insert operations on the view must satisfy the search conditions of view and all underlying views, regardless of whether the underlying views were defined with a check option. Furthermore, every updatable view that is directly or indirectly defined on view inherits those search conditions (the search conditions of view and all underlying views) as a constraint on insert or update operations.

Views (DB2 z/OS) - Definition
Complete the CREATE VIEW statement by typing or pasting in the relevant query.   Finally, use the **Schedule** or **Execute** buttons to [Create the New Object](#).

INTERBASE/FIREBIRD OBJECT WIZARDS

Rapid SQL provides wizards that let you create all supported InterBase/Firebird objects. For details, see the following topics:

- [Blob Filters Wizard (InterBase/Firebird)](#)
- [Domains Wizard (InterBase/Firebird)](#)
- [Encryption Keys wizard (InterBase/Firebird)](#)
- [Exceptions Wizard (InterBase/Firebird)](#)
- [External Functions Wizard (InterBase/Firebird)](#)
- [Foreign Key Wizard (IBM DB2 LUW)](#)
- [Generators Wizard (InterBase/Firebird)](#)
- [Indexes Wizard (InterBase/Firebird)](#)
- [Primary Keys Wizard (InterBase/Firebird)](#)
- [Procedures Wizard (InterBase/Firebird)](#)
- [Roles Wizard (InterBase/Firebird)](#)
- [Shadows wizard (InterBase/Firebird)](#)
- [Tables Wizard (InterBase/Firebird)](#)
- [Triggers Wizard (InterBase/Firebird)](#)
- [Unique Keys Wizard (InterBase/Firebird)](#)
- [Users wizard (InterBase/Firebird)](#)
- [Views Wizard (InterBase/Firebird)](#)

BLOB FILTERS WIZARD (INTERBASE/FIREBIRD)

The Blob Filter wizard lets you create and submit a DECLARE FILTER statement. Properties let you provide input and output subtypes, an entry point, and the module name.

To create a new blob filter using a wizard:

1   Open an Object Wizard for a blob filter.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Blob Filters (InterBase/Firebird) - Properties.

- **Comment** panel - for details, see Add a Comment to an Object.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

BLOB FILTERS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a Blob Filter, this tab/panel lets you work with the following settings:

Setting	Description
Name	A name that must be unique among filters on the database.
Input subtype	Lets you provide an INPUT_TYPE argument that specifies the Blob subtype from which data is to be converted.
Output subtype	Lets you provide an OUTPUT_TYPE argument that specifies the Blob subtype into which data is to be converted
Entrypoint	Lets you provide an ENTRY_POINT argument. Provide a quoted string specifying the name of the Blob filter as stored in a linked library.
Module name	Lets you provide a MODULE_NAME argument. Provide a quoted file specification identifying the object module in which the filter is stored.

DOMAINS WIZARD (INTERBASE/FIREBIRD)

The Domain wizard lets you provide datatype details and other basic clause/option values, in creating a domain.

To create a new domain using a wizard:

1   Open an Object Wizard for a domain.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Domains (InterBase/Firebird) - Properties.

- **Comment** panel - for details, see Add a Comment to an Object.

- **DDL View** panel - for details, see Create the New Object.

DOMAINS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a domain, this tab/panel lets you work with the following settings:

Group	Settings and descriptions	
Creation	Lets you provide a **Datatype** name.	
Base Datatype	Lets you select an InterBase/Firebird-supported **Type**. The following additional settings are available, depending on the type you choose:	
	Allow Nulls	Disabled, the domain is created with a NOT NULL clause.
	Size, **Precision**, and **Scale**	Provide InterBase/Firebird-standard datatype options.
	Array	Clicking the associated search button lets you select the upper and lower bounds for an array dimension.
Character Options	If you choose a character-based **Type**, the Character Set and Collation settings lets you select CHARACTER SET and COLLATION clause values used when creating the domain.	
Lob Options	If you choose a BLOB **Type**, you can specify the expected **LOB SEgment Length**, in bytes.	
Domain Options	**Default**	Specify a valid InterBase/Firebird DEFAULT argument value, to indicate the default column value that is entered when no other entry is made.
	Check	Specify a condition or requirement to be tested on data values at the time data is entered by applying a check constraint to a column.

ENCRYPTION KEYS WIZARD (INTERBASE/FIREBIRD)

The Encryption Key wizard lets you create keys used to encrypt and decrypt databases or table columns. It lets you specify the credentials, basic algorithm details, and other encryption options.

> **NOTE:** Before working with encryption keys, you should be familiar with InterBase/Firebird requirements regarding passwords and user permissions. See the InterBase documentation at http://docs.embarcadero.com/products/interbase/.

To create a new encryption key using a wizard:

1 Open an Object Wizard for an encryption key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Encryption Keys (InterBase/Firebird) - Properties.

- **DDL View** panel - for details, see Create the New Object.

Encryption Keys (InterBase/Firebird) - Properties

When creating or editing an encryption key, this tab/panel lets you work with the following settings:

Property	Descriptions
Name	When creating an encryption key, provide a unique name. After creating the encryption key, the name cannot be edited.
Algorithm	Options represent choices of the Advanced Encryption Standard or Data Encryption Standard algorithm and a key length.
Pad random	Enabled, the encryption key is created with a PAD RANDOM option, causing equal values to have different ciphertext. Disabled, the encryption key is created with a PAD NULL option, specifying that random padding should not be performed.
Init Vector random	Enabled, the encryption key is created with an INIT_VECTOR RANDOM option, enabling Cipher Block Chaining (CBC) encryption, forcing equal values to take different ciphertext. Disabled, the encryption key is created with a INIT_VECTOR NULL option, enabling Electronic Cookbook (ECB).
Password	Optionally, provide a user password.
Is Default	When enabled, the encryption key is created with an AS DEFAULT option, making this key the database default when no explicit key is specified for database or column encryption.

EXCEPTIONS WIZARD (INTERBASE/FIREBIRD)

The Exception wizard lets you provide the text of an exception and the name by which it is raised from a stored procedure.

To create a new exception using a wizard:

1 [Open an Object Wizard](#) for an exception.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Exceptions (InterBase/Firebird) - Properties](#).

- **Comment** panel - for details, see [Add a Comment to an Object](#).

- **DDL View** panel - for details, see [Create the New Object](#).

EXCEPTIONS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing an exception, this tab/panel lets you work with the following settings:

Group	Settings and descriptions
Creation	Lets you provide the **Name** by which the exception is referenced.
General	Lets you provide the **Text** of the exception.

EXTERNAL FUNCTIONS WIZARD (INTERBASE/FIREBIRD)

The External Functions wizard lets you DECLARE an existing, user-defined function.

To create a new external function using a wizard:

1 [Open an Object Wizard](#) for an external function.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [External Functions (InterBase/Firebird) - Properties](#).

   • **Parameters** panel - for details, see [External Functions (InterBase/Firebird) - Parameters](#).

   • **Return Type** panel - for details, see [External Functions (InterBase/Firebird) - Return Type](#).

   • **Comment** panel - for details, see [Add a Comment to an Object](#).

   • **DDL View** panel - for details, see [Create the New Object](#).

EXTERNAL FUNCTIONS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a generator, this tab/panel lets you work with the following settings:

Group	Settings and descriptions	
Creation	Lets you specify the **Name** of the function.	
General	**EntryPoint**	Type a quoted identifier as the ENTRY_POINT clause value for the declaration.
	ModuleName	Type a quoted identifier as the MODULE_NAME clause value for the declaration.

EXTERNAL FUNCTIONS (INTERBASE/FIREBIRD) - PARAMETERS

For each parameter to be added to this function:

1 Use the New button to add a new parameter and type a name for the parameter in the space provided.

2 With the parameter selected, in the **Attributes** area select a **Type**, and if appropriate, provide or select **Precision**, **Scale**, and **Size** options.

Use the Delete button to drop a selected parameter and use the arrow keys to reorder the parameter list.

EXTERNAL FUNCTIONS (INTERBASE/FIREBIRD) - RETURN TYPE

When creating an external function, this tab/panel lets you specify the datatype of the value returned by the external function. Select a **Type**, and if appropriate, provide or select **Precision**, **Scale**, and **Size** options.

You cannot modify the return type when editing an external function.

For each parameter to be added to this function:

1 Use the New button to add a new parameter and type a name for the parameter in the space provided.

2 With the parameter selected, in the **Attributes** area select a **Type**, and if appropriate, provide or select **Precision**, **Scale**, and **Size** options.

Use the Delete button to drop a selected parameter and use the arrow keys to reorder the parameter list.

FOREIGN KEYS WIZARD (INTERBASE/FIREBIRD)

The Foreign Key Wizard lets you create a foreign key constraint, mapping the relevant columns and specifying delete and update integrity checks.

To create a new foreign key using a wizard:

1 Open an Object Wizard for a foreign key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

  • **Properties** panel - for details, see Foreign keys (InterBase/Firebird) - Properties

  • **Column Mapping** panel - for details, see Foreign Keys (InterBase/Firebird) - Column Mapping

  • **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FOREIGN KEYS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner	The owner of the table where the foreign key is being created.
Table Name	The table where the foreign key link originates--the child table.
Name	Lets you provide a constraint name.
Delete Rule and **Update Rule**	Let you select ON UPDATE and ON DELETE argument values of NONE, CASCADE, SET DEFAULT, SET NULL, NO ACTION, or RESTRICT.

1 Under **Referenced Table**, choose the **Owner** and then the **Name** of the referenced, or parent, table.

2 Under the **Main Table**, select check boxes corresponding to the columns that are to reference columns in the referenced table. Then, under **Referenced Table**, select the corresponding column check boxes.

GENERATORS WIZARD (INTERBASE/FIREBIRD)

The Generator wizard lets you create a simple, named generator and set its initial value.

To create a new generator using a wizard:

1 [Open an Object Wizard](#) for a generator.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Generators (InterBase/Firebird) - Properties](#).

- **DDL View** panel - for details, see [Create the New Object](#).

GENERATORS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a generator, this tab/panel lets you work with the following settings:

Group	Settings and descriptions
Creation	Lets you provide the **Name** for the generator.
Current Value	Lets you provide a SET GENERATOR value, setting the value of the generator.

INDEXES WIZARD (INTERBASE/FIREBIRD)

The Index Wizard lets you create a basic InterBase/Firebird index, specifying the sort order and ACTIVE/INACTIVE status for the index, and indicating whether duplicate values are allowed

To create a new index using a wizard:

1 [Open an Object Wizard](#) for an index.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Indexes (InterBase/Firebird) - Properties](#).

- **Columns** panel - for details, see [Indexes (InterBase/Firebird) - Columns](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

INDEXES (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Name	The name of the index.
Table Name	The owning table.
Unique	When enabled, the index is created with the UNIQUE keyword, disallowing duplicate values.
Enabled	When enabled, the index is created or altered with an ACTIVE status. When disabled, the index is created or altered with an INACTIVE status.
Descending	Specifies the sort order. When enabled, the index is created with the DESCENDING keyword. When disabled, the index is created with the ASCENDING keyword.

INDEXES (INTERBASE/FIREBIRD) - COLUMNS

When adding or editing an index, this tab/panel lets you manage the columns that make up the index.

To add a column to the index:

Click the New button and select a column from the **Column** dropdown.

To delete a column from the index:

Select the column and click the Remove button.

To change the position of a column in the list:

Select the column and use the arrow keys to reorder the column list.

PRIMARY KEYS WIZARD (INTERBASE/FIREBIRD)

The Primary Key Wizard lets you add a basic InterBase/Firebird primary key to a table.

To create a new primary key using a wizard:

1 [Open an Object Wizard](#) for a primary key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

  - **Properties** panel - [Primary Keys (InterBase/Firebird) - Properties](#).

  - **Columns** panel - [Primary Keys (InterBase/Firebird) - Columns](#).

  - **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

PRIMARY KEYS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Name	The name of the primary key.
Table Name	The owning table.

PRIMARY KEYS (INTERBASE/FIREBIRD) - COLUMNS

When adding or editing a primary key, this tab/panel lets you manage the columns that make up the primary key.

  NOTE: Only columns defined with the **Allow Nulls** property disabled can be added to a primary key.

To add a column to the primary key:

Click the New button and select a column from the **Column** dropdown.

To delete a column from the primary key:

Select the column and click the Remove button.

To change the position of a column in the list:

Select the column and use the arrow keys to reorder the column list.

PROCEDURES WIZARD (INTERBASE/FIREBIRD)

The Procedures Wizard lets you create a procedure definition that includes parameter specifications, the body of the procedure, and execution permissions.

To create a new procedure using a wizard:

1   <u>Open an Object Wizard</u> for a procedure.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - <u>Procedures (InterBase/Firebird) - Properties</u>.

- **Parameters** panel - <u>Procedures (InterBase/Firebird) - Parameters</u>.

- **Body** panel - <u>Procedures (InterBase/Firebird) - Body</u>.

- **Permissions** panel - <u>Set Permissions for an Object</u>

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

PROCEDURES (INTERBASE/FIREBIRD) - PROPERTIES
When creating a procedure, this panel/tab lets you provide a **Name** for the procedure. After creation, you cannot change its name.

PROCEDURES (INTERBASE/FIREBIRD) - PARAMETERS
This tab lets you add and modify the parameters for a procedure.

To add a parameter:

1   Click the New button and type a name for the parameter in the space provided.

2   In the **Attributes** area, select a **Type** and if appropriate, provide **Precision**, **Scale**, and **Size** values.

3   Select a **Parameter Mode** of INPUT or OUTPUT.

To edit a parameter:

- Select the parameter and modify the values in the **Attributes** area.

To delete a parameter:

- Select the parameter and click the Delete button.

To change the ordering position of a parameter:

- Select the parameter and use the arrow buttons to move the parameter up or down in the list.

PROCEDURES (INTERBASE/FIREBIRD) - BODY

This tab provides a simple editor that lets you add or modify the body of the procedure.

ROLES WIZARD (INTERBASE/FIREBIRD)

The Roles Wizard lets you create a basic InterBase/Firebird role.

To create a new role using a wizard:

1   Open an Object Wizard for a role.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - Roles (InterBase/Firebird) - Properties.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

The Roles editor opens, letting you immediately assign users to the new roles and set up a set of permissions for the role. For more information, see Roles editor (InterBase/Firebird).

ROLES (INTERBASE/FIREBIRD) - PROPERTIES

When creating a role, this tab/panel lets you provide a **Name** for the role. When editing a role, this tab/panel displays the **Name** and **Authorization Owner** for the role.

SHADOWS WIZARD (INTERBASE/FIREBIRD)

The Shadows Wizard lets you create a shadow, consisting of one or more in-sync copies of the database stored on secondary storage devices.

To create a new shadow using a wizard:

1   Open an Object Wizard for a shadow.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - Shadows (InterBase/Firebird) - Properties.

   • **Storage** panel - Shadows (InterBase/Firebird) - Storage.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

SHADOWS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a shadow, this tab/panel lets you work with the following settings:

Setting	Description
Shadow Set	Lets you specify a set number, designating a shadow set to which files specified on the **Storage** panel belong. Once the shadow set is created, this value cannot be edited.
Conditional	Lets you include the CONDITIONAL argument with the CREATE SHADOW statement, when creating a new shadow. This allows shadowing to continue if the primary shadow becomes unavailable or if the shadow replaces the database due to disk failure
Behavior	Lets you include AUTO or MANUAL arguments with the CREATE SHADOW statement, specifying default access behavior when no shadow is available. AUTO specifies that all attachments and accesses succeed, all references to the shadow are deleted, and the shadow file is detached. MANUAL specifies that database attachments and accesses fail until a shadow becomes available or until all shadow references are removed from the database.

SHADOWS (INTERBASE/FIREBIRD) - STORAGE

When creating or editing a shadow, this tab/panel lets you manage a primary and one or more secondary files making up the shadow set. Note the following when managing shadow files:

- When creating a shadow, a primary file entry is automatically created when you select this tab. You must provide primary file property values before proceeding.

- A secondary file cannot be created if there is no primary file entry.

To add a primary or secondary file to the shadow:

1   Click the New button.

2   Use the following table as a guide to providing values in the **Property/Value** list for the file entry.

Setting	Description
Physical File Name	Provide the path and file name for the shadow file, which must reside on the local file system.
Starting Page	Starting page number of a secondary shadow file.
Size	The size, in pages.

To delete a primary or secondary file entry:

Select the file entry in the primary and secondary file list and click the Delete button.

TABLES WIZARD (INTERBASE/FIREBIRD)

The Tables wizard lets you create a table definition, providing column descriptions, indexes, permissions, and constraints for the table.

To create a new table using a wizard:

1   <u>Open an Object Wizard</u> for a table.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

  - **Properties** panel - for details, see <u>Tables (InterBase/Firebird) - Properties</u>.

  - **Columns** panel - for details, see <u>Tables (InterBase/Firebird) - Columns</u>.

  - **Indexes** panel - for details, see <u>Tables (InterBase/Firebird) - Indexes</u>.

  - **Constraints** panel - for details, see <u>Tables (InterBase/Firebird) - Constraints</u>.

  - **Comment** panel - for details, see <u>Add a Comment to an Object</u>.

  - **Permissions** panel - <u>Set Permissions for an Object</u>

  - **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

TABLES (INTERBASE/FIREBIRD) - PROPERTIES

When creating a table, this tab/panel lets you provide a **Name** for the table. When editing a table, the **Name** is for display only.

TABLES (INTERBASE/FIREBIRD) - COLUMNS

When creating or editing a table, this tab/panel lets you manage the columns for the table:

To add a column to the table:

1   Use the **Add Column** dropdown to add a new column to the bottom of the column list or to insert a new column above the currently selected column in the column list.

2   Provide a **Name** for the column.

3   Either select a **Type** for the column or select **Computed** and provide a valid InterBase/Firebird **Computed Expression**.

4   Use the following table as a guide to providing additional property values, noting that availability of a property differs by data type and other property selections.

Property	Description
Array Dimensions	Lets you type a valid InterBase/Firebird array_dim specifier or use the **...** button to open a dialog that lets you build the specifier.
Size	Lets you provide a size for character datatypes.
Allow Nulls	Disabled, submits a NOT NULL argument.
Default Character Set	Submits a CHARACTER SET argument value.
Default Collation	Submits a COLLATE argument value.

Property	Description
Default Value	Submits a literal COLLATE argument value.

To edit a column:

- Select the column from the list and edit property values as per the descriptions above.

To drop a column:

- Select the column from the list and click the **Delete** button.

To change the position of a column in the list:

- Select the column from the list and use the arrow buttons to change its position.

TABLES (INTERBASE/FIREBIRD) - INDEXES

When creating or editing tables, this tab/panel lets you manage indexes for the table:

- Click **Add** to open a wizard that lets you add a new index to the table. For information on using the wizard, see Indexes Wizard (InterBase/Firebird).

- Select an index and click **Edit** to open an editor on the index. For information on using the editor, see Indexes editor (InterBase/Firebird).

- Select an index and click **Drop** to drop an index from the table.

TABLES (INTERBASE/FIREBIRD) - CONSTRAINTS

When creating or editing tables, this tab/panel lets you manage foreign key, primary key, unique key and check constraints for the table:

- Select a constraint type and click **Add** to open a wizard that lets you add a new constraint of that type to the table.

- Select an existing constraint and click **Edit** to open a wizard that lets you add modify that constraint.

  NOTE: The wizards used to add and modify constraints offer functionality identical to the editors and wizards used when creating those object types. For details, see the following topics:

  Foreign Key Wizard (IBM DB2 LUW) and Foreign Keys editor (InterBase/Firebird)

  Primary Keys Wizard (InterBase/Firebird) and Primary Keys editor (InterBase/Firebird)

  Unique Keys Wizard (InterBase/Firebird) and Unique Keys editor (InterBase/Firebird)

  Add or Modify Check Constraint

- Select a constraint and click **Drop** to drop a constraint from the table.

TRIGGERS WIZARD (INTERBASE/FIREBIRD)

The Triggers wizard lets you create a basic InterBase trigger, providing a trigger body and specifying key properties such as the associated event type and the trigger timing.

To create a new trigger using a wizard:

1   Open an Object Wizard for a trigger.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Triggers (InterBase/Firebird) - Properties.

- **Body** panel - for details, see Triggers (InterBase/Firebird) - Body.

- **Comment** panel - for details, see Add a Comment to an Object.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

TRIGGERS (INTERBASE/FIREBIRD) - PROPERTIES

When creating a trigger, this tab/panel lets you provide values for the following trigger properties:

Property	Description
Parent Type	Select TABLE or VIEW as the Parent type.
Parent Name	Select the table or view that causes the trigger to fire when the specified operation occurs.
Name	Provide a name for the trigger.
Trigger Timing	Select whether the trigger fires BEFOR or AFTER the associated operation.
Trigger Events	Specifies the table operation (DELETE, INSERT, or UPDATE) that causes the trigger to fire.
Enabled	Enabled, this options submits an ACTIVE argument. Otherwise, an INACTIVE argument is submitted.
Position	Specify a POSITION number (0 - 32,767) to control firing order for triggers before or after the same action.

When editing a trigger, the values cannot be modified.

TRIGGERS (INTERBASE/FIREBIRD) - BODY

When creating or editing a trigger, this tab/panel previews the DDL reflecting your property values and lets you provide or modify the trigger body.

UNIQUE KEYS WIZARD (INTERBASE/FIREBIRD)

The Unique Key Wizard lets you add a basic InterBase/Firebird unique key to a table.

To create a new unique key using a wizard:

1   Open an Object Wizard for a unique key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - Unique Keys (InterBase/Firebird) - Properties.

- **Columns** panel - Unique Keys (InterBase/Firebird) - Columns.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

UNIQUE KEYS (INTERBASE/FIREBIRD) - PROPERTIES

When creating a unique key, this tab/panel lets you provide a **Name** for the unique key and select the owning **Table Name**. These values cannot be changed when editing the unique key.

UNIQUE KEYS (INTERBASE/FIREBIRD) - COLUMNS

When adding or editing a unique key, this tab/panel lets you manage the columns that make up the unique key.

> **NOTE:**   Only columns defined with the **Allow Nulls** property disabled can be added to a unique key.

To add a column to the unique key:

Click the New button and select a column from the **Column** dropdown.

To delete a column from the unique key:

Select the column and click the Remove button.

To change the position of a column in the list:

Select the column and use the arrow keys to reorder the column list.

USERS WIZARD (INTERBASE/FIREBIRD)

The Users wizard lets you create a basic InterBase/Firebird user, providing basic properties, assigning roles, and granting relevant permissions.

> **NOTE:** The Users node is only available for databases with the Embedded User Authentication option enabled. Before using Rapid SQL against InterBase/Firebird users, you should be familiar with this feature and with InterBase/Firebird permissions with regard to this object type. For access to InterBase documentation, see http://docs.embarcadero.com/products/interbase/.

To create a new user:

1 Open an Object Wizard for a user.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Users (InterBase/Firebird) - Properties.

   • **Comment** panel - for details, see Add a Comment to an Object.

   • **Roles** panel - for details, see Users (InterBase/Firebird) - Roles.

   • **Object Permissions** panel - Set Permissions for an Object

   • **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

The User editor opens. For more information, see Users editor (InterBase/Firebird).

USERS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a user, this tab/panel lets you provide values for the following properties:

Property	Description
Name	The login name for this user.
Password	The password for this user.
Group	Lets you specify a group name for the user. If unspecified, the user is created with the NO GROUP NAME option.
Default Role	Lets you select a default role for the user. If unspecified, the user is created with the NO DEFAULT ROLE option. Additional roles can be assigned for this user on the Roles tab/panel. For details, see Users (InterBase/Firebird) - Roles.
First Name, **Middle Name**, and **Last Name**	Let you provide the user's name, as opposed to the login name.
Active	This control is not enabled when creating a user and by default the user is created with the ACTIVE option. When editing a user, selecting this check box allows the user to log into the database. When unselected, the user is prevented from logging in to the database.

Property	Description
System User Name	Lets you specify a systemuser name for the user. If unspecified, the user is created with the NO SYSTEM USER NAME option.
Description	Lets you provide a short description for this user
GID	Lets you specify a numeric group ID for this user.
UID	Lets you specify a numeric user ID for this user.

USERS (INTERBASE/FIREBIRD) - ROLES

When creating or editing a user, this tab/panel lets you assign the valid, non-default roles that a user can specify when logging in to the database.

> **NOTE:** A default login role can be assigned for this user on the Properties tab/panel. For details, see Users (InterBase/Firebird) - Roles.

To assign roles to the user:

- Select the associated check box.

VIEWS WIZARD (INTERBASE/FIREBIRD)

The Views wizard lets you create a basic InterBase/Firebird view

To create a new view using a wizard:

1 Open an Object Wizard for a view.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Views (InterBase/Firebird) - Properties.

- **Comment** panel - for details, see Add a Comment to an Object.

- **Definition** panel - for details, see Views (InterBase/Firebird) - Definition.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

VIEWS (INTERBASE/FIREBIRD) - PROPERTIES

When creating or editing a view, this tab/panel lets you provide values for the following properties:

Property	Description
Name	Provide a name for the view.

Property	Description
Check Option	When enabled, a WITH CHECK OPTION argument is included with the DDL generated to create the view. This prevents INSERT or UPDATE operations on an updatable view if that operation violates the search condition specified in the associated WHERE clause.

When editing a view, these values cannot be modified.

VIEWS (INTERBASE/FIREBIRD) - DEFINITION
When creating or editing a view, this tab/panel previews the DDL reflecting your property values and lets you provide or edit the relevant query.

MICROSOFT SQL SERVER OBJECT WIZARDS

Rapid SQL lets you create SQL Server objects using the following wizards:

- Database Wizard (SQL Server)

- Database Triggers Wizard (SQL Server)

- Default Wizard (SQL Server)

- Extended Procedure Wizard (SQL Server)

- Foreign Key Wizard (SQL Server)

- Full-text Catalogs Wizard (SQL Server)

- Full-text Indexes Wizard (SQL Server)

- Function Wizard (SQL Server)

- Index Wizard (SQL Server)

- Login Wizard (SQL Server)

- Partition Functions Wizard (SQL Server)

- Partition Scheme Wizard (SQL Server)

- Primary Key Wizard (SQL Server)

- Procedure Wizard (SQL Server)

- Role Wizard (SQL Server)

- Rule Wizard (SQL Server)

- Schema Wizard (SQL Server)

- Synonym Wizard (SQL Server)

- Table Wizard (SQL Server)

- [Trigger Wizard (SQL Server)](#)

- [Unique Key Wizard (SQL Server)](#)

- [User Message Wizard (SQL Server)](#)

- [User Wizard (SQL Server)](#)

- [User Datatype Wizard (SQL Server)](#)

- [View Wizard (SQL Server)](#)

DATABASE WIZARD (SQL SERVER)

The Database Wizard presents you with a different set of options based on your server version to create the database accurately on each platform.

> **TIP:** Microsoft SQL Server recommends that you do not create any user objects, such as tables, views, stored procedures, or triggers, in the master database. The master database includes the system tables that store the system information used by SQL Server, such as configuration option settings.

To create a new database using a wizard:

1 [Open an Object Wizard](#) for a database.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Options** panel - for details, see [Databases (SQL Server) - Options](#).

- **Placement** panel - for details, see [Databases (SQL Server) - Placement](#).

- **Transaction Log** panel - for details, see [Databases (SQL Server) - Transaction Log](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

DATABASES (SQL SERVER) - OPTIONS

When creating or editing a database, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the database.
Attach existing OS files	To create a database from an existing set of operating system files, there must be a <filespec> entry for the first PRIMARY file. The PRIMARY filegroup contains all the database system tables. Primary files have a .mdf extension.
Compatible Level	Select a version compatibility level.

Setting	Description
Properties group	Select the following settings: **ANSI null default**, **ANSI nulls**, **ANSI padding**, **ANSI warnings**, **auto create statistics**, **auto update statistics**, **autoclose**, **autoshrink**, **concat null yields null**, **cursor close on commit**, **arithabort**, **db chaining**, **dbo use only**, **default to local cursor**, **merge publish**, **numeric roundabout**, **offline**, **published**, **quoted identifier**, **read only**, **recursive triggers**, **select into/bulkcopy/pllsort**, **single user**, **subscribed**, **torn page detection**, and **trunc log on chkpt**.

DATABASES (SQL SERVER) - PLACEMENT

Indicate the file where you want the database to live. For example, a new Books database could include author and title filegroups.

By default, when you open the Wizard and click the **Placement** tab, a filegroup definition, using the name you provided for the database and default settings, is displayed. For each filegroup to be added, click the New button, provide a **Device File Name** for the filegroup, and use the **File Group Properties** and **Device File Properties** groups to provide the attributes of the filegroup.

Use the Delete button to delete a selected filegroup.

DATABASES (SQL SERVER) - TRANSACTION LOG

The transaction log file is a required file for each database. This file holds the log information to recover the database. There can be multiple log files for a database, but there has to be at least one. Traditionally the logfile extension has been .ldf.

By default, when you open the Wizard and click the **Transaction Log** tab, a transaction log file definition, using the name derived from the name you provided for the database and with default settings, is displayed. For each file to be added, click the New button, provide a **Device File Name**, and use the **Log Device Properties** group to provide the attributes of the file.

Use the Delete button to delete a selected file.

> **NOTE:** As you complete the wizard, be aware that the Primary file contains startup information for the database and is also used to store data. The transaction log files hold the information used to recover the database.

DATABASE TRIGGERS WIZARD (SQL SERVER)

This wizard builds and submits a CREATE TRIGGER... ON DATABASE statement, letting you create a DDL trigger with database scope, that fires in response to selected DDL events. The wizard builds basic syntax only, letting you provide the actions and conditions SQL making up the main body of the trigger.

> **NOTE:** This functionality is available for SQL Server 2005 and up.

> **NOTE:** Before working with database triggers, consult Microsoft SQL Server documentation for general information on DDL triggers as well as specifics such as FOR/AFTER argument details, valid actions and conditions, and supported events. For more information, see Accessing Third Party Documentation.

To create a new database trigger using a wizard:

1 [Open an Object Wizard](#) for a database trigger.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [Database Triggers (SQL Server) - Properties](#).

   • **Trigger Events** panel - for details, see [Database Triggers (SQL Server) - Trigger Events](#).

   • **Definition** panel - for details, see [Create the New Object](#).

3 Finally, use the Execute button to create the object. For more information, see [Create the New Object](#).

   > **NOTE:**   In order to fire in response to specified events, a database trigger must be enabled. For information on enabling and disabling a database trigger, see [Database Triggers Editor (SQL Server)](#) and [Change Status (SQL Server DDL triggers)](#).

DATABASE TRIGGERS (SQL SERVER) - PROPERTIES

When creating or editing a database trigger, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the DDL trigger.
Trigger Timing	Use this control to specify a FOR/AFTER argument option. The specific events for the argument are provided on the **Trigger Events** tab/panel.
Encrypted	Select this check box to add a WITH ENCRYPTION argument to the generated DDL. This prevents the trigger from being published as part of a SQL Server replication.

DATABASE TRIGGERS (SQL SERVER) - TRIGGER EVENTS

When creating or editing a database trigger, this tab/panel lets you specify the FOR/AFTER argument appearing in the generated CREATE TRIGGER... ON DATABASE syntax. For each event that is to fire this trigger, select the associated check box.

DEFAULT WIZARD (SQL SERVER)

When bound to a column or user-defined object, a default ensures that a specific value will be inserted into the column where the object will be bound if no explicit value is given.

The Default Wizard lets you name the default and specify its value.

To create a new default using a wizard:

1   <u>Open an Object Wizard</u> for a default.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - <u>Defaults (SQL Server) - Properties</u>.

- **Dependencies** panel - <u>Defaults (SQL Server) - Dependencies</u>.

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

DEFAULTS (SQL SERVER) - PROPERTIES

When creating or editing a default, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the schema that is to own the default.
Name	Provide a name for the default.
Value	Provide the value of the default.

DEFAULTS (SQL SERVER) - DEPENDENCIES

From the **Type** dropdown, choose Column or Datatype, and if you chose **Column**, choose a Table from the **Table** dropdown. The list on the left is populated with candidate columns or datatypes. To move a candidate from the list on the left to the dependencies column on the right, select the candidate and click **Add**. Remove columns or datatypes from the dependencies list on the right by selecting the column or datatype and clicking **Remove**.

EXTENDED PROCEDURE WIZARD (SQL SERVER)

Extended Procedures are dynamic link libraries that can be used to load and execute application routines written in other programming languages, such as C or Visual Basic. Extended Procedures function and appear in the same manner as SQL Server stored procedures in that you can pass parameters to them and obtain results.

Extended stored procedures provide a method for calling procedural language functions from within the Adaptive Server.

> **NOTE:**   Extended procedures can only be created in the Master database.

To create a new Extended procedure using a wizard:

1 [Open an Object Wizard](#) for an extended procedure.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - [Extended Procedures (SQL Server) - Properties](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

EXTENDED PROCEDURES (SQL SERVER) - PROPERTIES

When creating or editing an extended procedure, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the extended procedure.
Name	Provide a name for the extended procedure.
Library	Provide the name of the DLL containing the extended procedure.

FOREIGN KEY WIZARD (SQL SERVER)

Foreign keys are unique values that refer to specific columns of other tables. Thus, a foreign key links two tables together. Embarcadero Rapid SQL's Foreign Key Wizard makes it easy for you to create a relational link between two tables, thereby speeding queries and giving you faster access to data. The column in the initial table, the parent table, is called the primary key. The corresponding column in the (child) table that references the primary key, is the foreign key. Foreign keys can also refer to columns within the same table.

To create a new Foreign Key using a wizard:

1 [Open an Object Wizard](#) for a foreign key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - [Foreign Keys (SQL Server) - Properties](#).

- **Column Mapping** panel - [Foreign Keys (SQL Server) - Column Mapping](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

FOREIGN KEYS (SQL SERVER) - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner and **Table Name**	Select the owner and name of the table for which the foreign key is being created.
Name	Provide a name for the foreign key.
Enabled	Enables or disables the foreign key.
Not For Replication	Replication copies and distributes data and database objects from one database to another and then synchronizes information between databases for consistency.
Delete Rule	If you choose the CASCADE option, all rows containing data involved with the foreign key will be deleted after a delete operation.
Update Rule	If you choose the CASCADE option, all rows containing data involved with the foreign key will be deleted after an update operation.

FOREIGN KEYS (SQL SERVER) - COLUMN MAPPING

Under **Referenced Table**, choose the **Owner** and then the **Name** of the referenced, or parent, table.

Under the **Main Table**, select check boxes corresponding to the columns that are to reference columns in the referenced table. Then, under **Referenced Table**, select the corresponding column check boxes.

FULL-TEXT CATALOGS WIZARD (SQL SERVER)

This wizard builds and submits a CREATE FULLTEXT CATALOG statement. This lets you add a full-text catalog to a database, specifying accent sensitivity and an owner, and letting you define the catalog as the default full-text catalog. Against SQL Server 2005 datasources, you can also provide filegroup and root directory details.

Along with full-text indexes, full-text catalogs facilitate full-text searching. For related information, see [Full-text Indexes Wizard (SQL Server)](#).

> **NOTE:** This functionality is available as of SQL Server 2005.

> **NOTE:** Full-text catalogs cannot be created in the master catalog.

> **NOTE:** Before working with full-text catalogs, consult Microsoft SQL Server documentation for general information on full-text searching and detailed information on topics such as restrictions on where these objects can be created, required permissions, virtual object and filegroup considerations, and the relationship with full-text indexes and the default collation. For more information, see [Accessing Third Party Documentation](#).

To create a new full-text catalog using a wizard:

1   Open an Object Wizard for a full-text catalog.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Full-text Catalogs (SQL Server) - Properties.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FULL-TEXT CATALOGS (SQL SERVER) - PROPERTIES

When creating or editing a full-text catalog, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the catalog.
Filegroup (SQL Server 2005 only)	Selecting a filegroup adds an explicit ON FILEGROUP clause, specifying the selected filegroup to which the catalog will belong. If not specified, the new catalog will be part of the default filegroup used for all full-text catalogs.
Path (SQL Server 2005 only)	Specifying a path adds an explicit IN PATH clause, specifying the root directory for the catalog. If no path is specified, the new catalog will be located in the default directory specified during setup.
Accent Sensitive	Lets you specify a WITH ACCENT_SENSITIVITY = ON/OFF clause, reflecting the selected/deselected status of the check box.
Default	Select this check box to add an AS DEFAULT argument, specifying that this catalog is the default full-text catalog. If an existing full-text catalog is currently specified as the default, this catalog becomes the new default.
Authorization Owner	Select a user or role to add an AUTHORIZATION argument that sets the owner of the full-text catalog.

FULL-TEXT INDEXES WIZARD (SQL SERVER)

This wizard lets you build and submit a CREATE FULLTEXT INDEX statement, adding a full-text index for a table or indexed view. Along with full-text catalogs, full-text indexes facilitate full-text searching.

NOTE: This functionality is available as of SQL Server 2005.

NOTE: Full-text indexes cannot be created in the master catalog.

NOTE: Before working with full-text indexes, consult Microsoft SQL Server documentation for general information on full-text searching and detailed information on topics such as required permissions, column specification details, recommended datatype, and XML considerations. For more information, see Accessing Third Party Documentation.

To create a new full-text index using a wizard:

1   <u>Open an Object Wizard</u> for a full-text index.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see <u>Full-text Indexes (SQL Server) - Properties</u>.

- **Columns** panel - for details, see <u>Full-text Indexes (SQL Server) - Columns</u>.

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

For related information, see <u>Full-text Catalogs Wizard (SQL Server)</u>.

FULL-TEXT INDEXES (SQL SERVER) - PROPERTIES
When creating or editing a full-text index, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type	Select table or indexed view.
Parent Schema	The parent's owning schema.
Parent Name	Select the name of the table or indexed view on which the index is to be created.
Key Index	Select the unique key index on the table or view that will be used as the value for the KEY INDEX argument.
Full-text Catalog Name	Select an existing full-text catalog. For information on creating full-text catalogs, see <u>Full-text Catalogs Wizard (SQL Server)</u>. If you do not select a catalog, SQL Server will assign the index to the default full-text catalog, generating an error if no default exists.
Filegroup Name (SQL Server 2008 ^)	Select the filegroup that the index is to be created on. Specifying no filegroup results in SQL Server default behavior, placing the index in the same filegroup as the base table or view for a non-partitioned table or in the PRIMARY filegroup for a partitioned table.
Change Tracking	Select a CHANGE_TRACKING argument value of MANUAL, AUTO, OFF, or OFF, NO POPULATION to specify whether changes to columns covered by the full-text index will be propagated to the full-text index.
Stoplist (SQL Server 2008 ^)	Select a STOPLIST argument value of OFF to specify that no stoplist be associated with the index or SYSTEM to specify that the default full-text system STOPLIST be used for this index.

FULL-TEXT INDEXES (SQL SERVER) - COLUMNS
When creating or editing a full-text index, this tab/panel lets you specify the columns that make up the index:

- For each column to include in the full-text index, select the associated **Indexed** check box.

- Optionally, select a language from the associated **Language** column.

- For varbinary(max) and image columns, provide a TYPE COLUMN argument filename extension.

FUNCTION WIZARD (SQL SERVER)

Functions are subroutines that you define so you can reuse code without having to reinvent the wheel each time. You can use functions to determine the best methods for controlling access and manipulation of the underlying data contained in an object. A function returns a value, unlike a stored procedure, which does not.

- To create a user-defined function, you need CREATE ANY privileges or IMPLICIT_SCHEMA authority on the database if the schema does not already exist.

To create a new function using a wizard:

1 [Open an Object Wizard](#) for a function.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Functions (SQL Server) - Properties](#).
- **Definition** panel - for details, see [Functions (SQL Server) - Definition](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

FUNCTIONS (SQL SERVER) - PROPERTIES

When creating or editing a function, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the function.
Name	Provide a name for the function.
Schema Binding	Choose whether the function is bound to database objects that it references.
Encryption	Choose whether SQL Server encrypts table columns that contain the text of the CREATE FUNCTION statement.

FUNCTIONS (SQL SERVER) - DEFINITION

Complete the CREATE FUNCTION outline provided by typing or pasting the body of the function. Finally, use the **Schedule** or **Execute** buttons to [Create the New Object](#).

INDEX WIZARD (SQL SERVER)

Like an index in a book, a table index helps you get at the data you want without having to read through the whole table. Indexes can exist on single column or on multiple columns. Indexes appear in the form of B-trees. And, as for books, you can have multiple indexes for a single table. You can also create indexes for a view.

To create a new index using a wizard:

1   Open an Object Wizard for an extended procedure.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - Indexes (SQL Server) - Properties.

   • **Columns** panel - Indexes (SQL Server) - Columns.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

INDEXES (SQL SERVER) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type	Select TABLE or VIEW.
Parent Owner	Select the owner of the table or view.
Parent Name	Select the specific table or view containing the columns you want to index.
Name	Provide a name for the index.
Build Online	Enabling this feature specifies that the ONLINE=ON clause is used when creating this object and can subsequently be used when rebuilding or dropping this object.
Max degree of parallelism	Lets you specify a MAXDOP index operation value, limiting the number of processors used in parallel plan execution.
Index Type	Select UNIQUE or NONUNIQUE. An index is unique when no two rows are permitted to have the same index value. (Note: A clustered index on a view must be unique.) If an INSERT or UPDATE statement creates a duplicate value, the operation may fail.
Clustered	A clustered index is one in which the physical order of rows matches the order of indexed rows. A table or view can only have one clustered index at a time.
	In a nonclustered index, the physical order of rows is independent of the indexed order of rows.
	For an indexed view, you can only create a nonclustered index if there is already a clustered index extant.
	With this check box enabled, subsequent rebuild and drop operations offer an online option.

Setting	Description
Ignore Duplicate Key	This option controls what happens when an attempt is made to insert a duplicate key value into a column that is part of a unique clustered index.
	If the option is selected and an INSERT statement that creates a duplicate key is executed, SQL Server issues a warning and ignores the duplicate row.
	If not selected, SQL Server issues an error message and rolls back the entire INSERT statement.
Statistics Recompute	Enabling this feature means queries involving the table run at the optimal level as distribution statistics are updated automatically when the index is created. If you disable this option, you can compromise query performance.
Partitioned	When selected, an ON clause is added to the CREATE TABLE statement, letting you specify a **Partition Scheme**, partitioning this table.
Partition Scheme	This property is only available if the **Partitioned** check box is selected.
	Select the partition scheme that specifies the filegroup mapping for this table. For information on creating partition schemes, see [Partition Scheme Wizard (SQL Server)](#).
Partition Column	This property is only available if the **Partitioned** check box is selected.
	Select the column that the index will be partitioned against.
Filegroup	This property is only available if the **Partitioned** check box is not selected.
	Select an existing, named filegroup to have the index stored on the specified filegroup. Select PRIMARY to have the index stored on the default database filegroup.
Pad Index	Enable or disable padding of index pages.
Sort in TempDB	Select to store the intermediate index sort results in tempdb. This option may reduce the time needed to create an index if tempdb is on a different set of disks than the user database, but it increases the amount of disk space used to create an index. In addition to the space required in the user database to create the index, tempdb must have about the same amount of additional space to hold the intermediate sort results.
Allow Row Locks and **Allow Page Locks**	Lets you enable locking granularity at the page and row level (Default=TRUE)
	NOTE: You cannot reorganize an index (primary key, or unique key) that has an **Allow Page Locks** property set to FALSE. For information on reorganizing indexes, see [Reorganize (SQL Server indexes, primary keys, and unique keys)](#).

INDEXES (SQL SERVER) - COLUMNS

From the **Column** dropdown, select a column for the index and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

LOGIN WIZARD (SQL SERVER)

The Login wizard lets you build and submit a CREATE LOGIN statement, adding a new SQL Server, Windows, certificate-based, or asymmetric key-based login. The Login wizard also lets you:

- Add the login to fixed server roles

- Create user accounts for the login on selected databases

To create a new login using a wizard:

1 <u>Open an Object Wizard</u> for a login.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see <u>Logins (SQL Server) - Properties</u>.

- **Server Roles** panel - for details, see <u>Logins (SQL Server) - Server Roles</u>.

- **Users** panel - for details, see <u>Logins (SQL Server) - Users</u>.

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3 Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

LOGINS (SQL SERVER) - PROPERTIES

When creating or editing a login, this tab/panel lets you work with the following properties:

Setting	Description
Name	Specifies a name for this login.
Default Database	Lets you specify a DEFAULT_DATABASE argument, specifying a default database for this login.
Default Language	Lets you specify a DEFAULT_LANGUAGE argument, specifying a default language for this login.
Account Type	Lets you select one of the following login types:
	STANDARD - (SQL Server logins) if you select this type, you must also provide **Password** and **Check Policy** values.
	NTGROUP or NTUSER - (Windows logins) if you select one of these types, you must also provide a **Domain** value.
	CERTIFICATE - (certificate-mapped logins) if you select this type, you must also provide a **Certificate** value.
	ASYMMETRIC KEY - (asymmetric key-mapped logins) if you select this type, you must also provide an **Asymmetric Key** value.
Password	This is only available for an **Account Type** of STANDARD. It lets you provide a password for the login.
Check Policy	This is only available for an **Account Type** of STANDARD. If selected, login policies defined on the server apply to this login.
Check Expiration	This is only available if **Check Policy** is selected. If selected, password expiration policies are applied to this login.
Must Change	This is only available if **Check Expiration** is selected. If selected, the user will be prompted to change the password on the first login.
Domain	This is only available for an **Account Type** of NTUSER or NTGROUP. Lets you specify the domain name value that will be specified with the loginName argument ([<domainName>\<loginName>]).

Setting	Description
Certificate	This is only available for an **Account Type** of CERTIFICATE. It lets you specify the name of a certificate that is to be associated with this login. It lets you specify the name of a certificate that is to be associated with this login.
Asymmetric Key	This is only available for an **Account Type** of ASYMMETRIC KEY. It lets you specify the name of an asymmetric key that is to be associated with this login.

LOGINS (SQL SERVER) - SERVER ROLES

When creating or editing a login, this tab/panel builds sp_addsrvrolemember procedure calls that will be submitted along with the CREATE LOGIN statement issued by this wizard/editor. This lets you add the login as a member of one or more fixed server roles by selecting the check boxes associated with those roles.

For information on editing fixed server roles, see [Roles Editor (SQL Server)](#).

LOGINS (SQL SERVER) - USERS

When creating or editing a role, this tab/panel lets you build CREATE USER... FOR LOGIN statements that will be submitted with the CREATE LOGIN statement issued with this wizard/editor. Maintain database user accounts for a login as follows:

- Add a user account for this login to a database by selecting a database from the **Databases where the Login does NOT have a User Account** and clicking the **Add** button. This opens the SQL Server User wizard, letting you define a new user on that database. For more information, see [User Wizard (SQL Server)](#).

- Remove a user account for this login from a database by selecting a database from the **Databases where the Login HAS a User Account** and clicking the **Remove** button.

PARTITION FUNCTIONS WIZARD (SQL SERVER)

The Partition Functions Wizard lets you build and submit a CREATE PARTITION FUNCTION statement, specifying a range, input parameter type, and boundary values. Once created, the partition function can be referenced by the partition scheme used to create a partitioned table or index.

NOTE: Before creating a partition function, consult Microsoft SQL Server documentation for detailed information on topics such as type restrictions and boundary value usage. For more information, see [Accessing Third Party Documentation](#).

To create a new partition function using a wizard:

1   <u>Open an Object Wizard</u> for a partition function.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties panel** - for details, see <u>Partition Functions (SQL Server) - Properties</u>.

   • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

For related information, see the following topics:

   • <u>Partition Scheme Wizard (SQL Server)</u>

   • <u>Index Wizard (SQL Server)</u>

   • <u>Table Wizard (SQL Server)</u>

PARTITION FUNCTIONS (SQL SERVER) - PROPERTIES

When creating or editing a partition function, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the partition function.
Input Parameter Type	Select the data type of the column used for partitioning.
Range	Select to which side of each functional value interval (LEFT, RIGHT) the functional value belongs, when interval values are sorted in ascending order from left to right.
Function Values	Lets you specify the boundary values to include as FOR VALUES arguments to the CREATE PARTITION FUNCTION statement. Each boundary value is a constant expression that can reference variables. The set of values provided define the partition points for an index or table partitioned using this function. Use the New, Delete, Up, and Down buttons to create and maintain the value list.

PARTITION SCHEME WIZARD (SQL SERVER)

The Partition Scheme Wizard builds and submits a CREATE PARTITION SCHEME statement, letting you specify a partition function and a file group mapping. Once created, the partition scheme can be referenced when creating a partitioned table or index.

   NOTE: Before creating a partition scheme, consult Microsoft SQL Server documentation for detailed information on topics such as the relationship between the partition function and filegroups specified. For more information, see <u>Accessing Third Party Documentation</u>.

To create a new partition wizard using a wizard:

1   Open an Object Wizard for a partition scheme.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Partition Schemes (SQL Server) - Properties.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

For related information, see the following topics:

- Partition Functions Wizard (SQL Server)

- Index Wizard (SQL Server)

- Table Wizard (SQL Server)

PARTITION SCHEMES (SQL SERVER) - PROPERTIES

When creating or editing a partition scheme, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the partition scheme.
Partition Function	Select the partition function that specifies the range, input parameter type, and boundary values for this partition scheme.
Filegroup Name	Lets you specify the names of the filegroups to store the partitions specified by the selected **Partition Function**.
	Consult Microsoft SQL Server documentation for details on the relation ship between filegroups and partitions. For more information, see Accessing Third Party Documentation.
	Use the New, Delete, Up, and Down buttons to create and maintain the value list.

PRIMARY KEY WIZARD (SQL SERVER)

Primary key constraints make sure that no duplicate values or NULLS are entered in the columns you specify. You can use primary key constraints to enforce uniqueness and referential integrity. A table can only have a single primary key constraint.

The dialog box lets you specify the owner and table on which you want to place the primary key constraint.

To create a new primary key using a wizard:

1 <u>Open an Object Wizard</u> for a primary key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - <u>Primary Keys (SQL Server) - Properties</u>.

- **Columns** panel - <u>Primary Keys (SQL Server) - Properties</u>.

- **DDL View** panel - for details, see <u>Create the New Object</u>.

3 Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

PRIMARY KEYS (SQL SERVER) - PROPERTIES

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner and **Table Name**	Choose the owner and name of the table in which the primary key is being created.
Name	Provide a name of the primary key being created.
Build Online	Enabling this feature specifies that the ONLINE=ON clause is used when creating this object and can subsequently be used when rebuilding or dropping this object.
Max degree of parallelism	Lets you specify a MAXDOP index operation value, limiting the number of processors used in parallel plan execution.
Clustered	Enable or disable clustering. With this check box enabled, subsequent rebuild and drop operations offer an online option.
File Group	If you do not specify a filegroup, Microsoft SQL Server creates the index in the default filegroup.
Fill Factor	This specifies how full each index page that's storing data should be. The fill factor is a percentage value between 0 and 100.

Primary Keys (SQL Server) - Columns

From the **Column** dropdown, select a column for the primary key and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

PROCEDURE WIZARD (SQL SERVER)

Procedures are a reusable block of PL/SQL, stored in the database, that applications can call. Procedures streamline code development, debugging, and maintenance by being reusable. Procedures enhance database security by letting you write procedures granting users execution privileges to tables rather than letting them access tables directly.

The Procedure Wizard lets you:

- Name the procedure and specify its body.

- Specify any execution options and you can encrypt the stored procedure text in syscomments.

To create a new procedure using a wizard:

1. [Open an Object Wizard](#) for a procedure.

2. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   - **Properties** panel - for details, see [Procedures (SQL Server) - Properties](#).

   - **Definition** panel - for details, see [Procedures (SQL Server) - Definition](#).

3. Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

PROCEDURES (SQL SERVER) - PROPERTIES

When creating or editing a procedures, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the procedure.
Name	Provide a name for the procedure
Procedure Number	Optionally, provide a procedure number. By using a number, you can group procedures of the same name together. This also enables you to drop them using only one DROP PROCEDURE statement. So, the procedures bill;1, bill;2, bill;3, etc. will be dropped simultaneously when the time comes.
Replication	This option creates a procedure that is used as a stored procedure filter and is executed only during replication.
Recompile	The plan for this procedure will not be cached and the procedure is recompiled when it is run. This option is appropriate when you're using atypical or temporary values and you don't want to override the execution plan cached in memory.
Encryption	If you select this option, SQL Server will encrypt the syscomments table entry containing the text of the CREATE PROCEDURE statement. It keeps the procedure from being published as part of replication.

PROCEDURES (SQL SERVER) - DEFINITION

Complete the CREATE PROCEDURE outline provided by typing or pasting the body of the procedure. Finally, use the **Execute** button to [Create the New Object](#).

ROLE WIZARD (SQL SERVER)

Roles are sets of user privileges you associate with access to objects within a database. Roles streamline the process of granting permissions. You can use roles to grant sets of permissions and privileges to users and groups. Roles can help you comply with Sarbanes Oxley regulations by limiting which users can have access to what privileges, for example a Human Resources Role versus an Accounting Role.

To create a new role using a wizard:

1   Open a creation wizard for a role. For details, see Open an Object Wizard.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Roles (SQL Server) - Properties.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object.

After you click **Finish** and the role has been created, the Roles Editor open. You can then assign object and system privileges to the role and determine which users can take part in the role. For more information, see Roles Editor (SQL Server).

ROLES (SQL SERVER) - PROPERTIES

When creating or editing a role, this tab/panel lets you work with the following settings:

Setting	Description
Name	The name used to reference the role.
RoleType	The value can be NONE, APPLICATION, or STANDARD.
Authorization Owner	Only required with a **Role Type** of STANDARD.
Password	Only required with a **Role Type** of APPLICATION.

RULE WIZARD (SQL SERVER)

Rules promote data integrity by allowing you to validate the values supplied to a table column. They are reusable objects that you can bind to table columns or user datatypes. Check constraints are similar to rules, and are in fact the preferred way of restricting data. A column or user-defined data type can have only one rule bound to it, but a column can have both a rule and one or more check constraints associated with it. Not that a rule cannot apply to data already existing in the database at the time you're creating the rule and can't be bound to a system-created data type. If you create a new rule when one already exists, the new rule will override the previous one.

To create a new rule using a wizard:

1   <u>Open an Object Wizard</u> for a rule.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

  • **Properties** panel - <u>Rules (SQL Server) - Properties</u>.

  • **Dependencies** panel - <u>Rules (SQL Server) - Dependencies</u>.

  • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

RULES (SQL SERVER) - PROPERTIES
When creating or editing a rule, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the Owner of the rule.
Name	Provide a name for the rule.
Restriction	Type the condition. The rule restriction is the condition that defines the rule and can be any expression valid in a WHERE clause and can include such elements as arithmetic operators, relational operators, and predicates (for example, IN, LIKE, BETWEEN).

RULES (SQL SERVER) - DEPENDENCIES
From the **Type** dropdown, choose Column or Datatype, and if you chose **Column**, choose a Table from the **Table** dropdown. The list on the left is populated with candidate columns or datatypes. To move a candidate from the list on the left to the dependencies column on the right, select the candidate and click **Add**. Remove columns or datatypes from the dependencies list on the right by selecting the column or datatype and clicking **Remove**.

SCHEMA WIZARD (SQL SERVER)
Rapid SQL lets you create a schema on Microsoft SQL Server.

To create a new schema using a wizard:

1   <u>Open an Object Wizard</u> for a schema.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

  • **Properties** panel - <u>Schema (SQL Server) - Properties</u>.

  • **Permissions** panel - <u>Schema (SQL Server) - Permissions</u>.

  • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

SCHEMA (SQL SERVER) - PROPERTIES

When creating or editing a schema, this tab/panel lets you provide a **Name** and select the **Owner** of the schema.

SCHEMA (SQL SERVER) - PERMISSIONS

For each specific permission to be granted, select the cell corresponding to the name and specific permission, and click the **Grant** button. To revoke a privilege, select a cell showing a Granted permission and click **Revoke**.

SYNONYM WIZARD (SQL SERVER)

This wizard builds and submits a CREATE SYNONYM statement, letting you build a one-part name that can be used in SQL statements instead of a fully-qualified, multi-part name.

To create a new synonym using a wizard:

1   Open an Object Wizard for a wizard.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Synonyms (SQL Server) - Properties.

   • **Permissions** panel - Set Permissions for an Object

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

SYNONYMS (SQL SERVER) - PROPERTIES

When creating a synonym, this tab/panel lets you provide a synonym name and fully qualify the object name that it references. When editing a synonym, this tab/panel lets you view the synonym definition.

> **NOTE:**   Before creating or editing a synonym, consult Microsoft SQL Server documentation for information on topics such as support for specific function and procedure types, and three-part and four-part name restrictions. For assistance, see Accessing Third Party Documentation.

The following table describes the available settings:

Setting	Description
Schema and **Name**	Use these setting to select an owner and type the one-part name that will be used to reference
Server, **Database**, **Referenced Object Type**, **Referenced Object Owner**, and **Referenced Object Name**	Use these setting to select the specific object to which the synonym is to refer. Optionally, you can use the server, database, and owner settings to construct a partially-qualified to fully-qualified name for the object. The user interface will allow you to construct all valid, multi-part name variations (for example: *server*.*database*.*owner*.object, *database*.*owner*.*object*, *owner*.*object*, *server*...*object*).

TABLE WIZARD (SQL SERVER)

A table is a column-based arrangement of data in which the content of one column has a bearing on the other column(s). So, for example, a table might have a column for authors, another column for the books each author has written, and a third for the number of copies each title by a given author has sold. The data moves across the columns in rows.

You must have CREATE TABLE permissions to generate a new table.

To create a new table using a wizard:

1 Open an Object Wizard for a table.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Dependencies** panel - Tables (SQL Server) - Properties.

   • **Columns** panel - Tables (SQL Server) - Columns.

   • **Indexes** panel - Tables (SQL Server) - Indexes.

   • **Constraints** panel - Tables (SQL Server) - Constraints.

   • **Permissions** panel - Set Permissions for an Object.

   • **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

TABLES (SQL SERVER) - PROPERTIES

When creating or editing a table, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Select the owner of the table.
Name	Provide a name for the table

Setting	Description
ANSI_NULLS option	By setting this option, you are setting ANSI_NULLS, ANSI_PADDING, ANSI_WARNINGS, and QUOTED_IDENTIFIER to on so the table can be used in an Indexed View.
Partitioned	When selected, an ON clause is added to the CREATE TABLE statement, letting you specify a **Partition Scheme**, partitioning this index.
Partition Scheme	This property is only available if the **Partitioned** check box is selected.
	Select the partition scheme that specifies the filegroup mapping for this index. For information on creating partition schemes, see [Partition Scheme Wizard (SQL Server)]().
Filegroup	This property is only available if the **Partitioned** check box is not selected.
	Select an existing, named filegroup to have the table stored on the specified filegroup. Select PRIMARY to have the table stored on the default database filegroup.
Text Image Filegroup	Select an existing, named filegroup to have text, ntext, image, xml, varchar(max), nvarchar(max), varbinary(max), and CLR user-defined type columns stored on the specified filegroup. Select PRIMARY to have columns of those types stored on the default database filegroup.

TABLES (SQL SERVER) - COLUMNS

Use the **Add Column** button to add the columns for the table. After providing a **Name** for a new column, you can modify column properties in the **Property/Value** list. Available properties depend on the datatype you choose as well as on the property values you select:

Computed and **Computed Expression** - Let you define a column as a computed column and provide the computed column expression.

Type - Lets you select a datatype (depending on the type, additional properties such as **Size**, **Width**, and **Scale** may be available).

Identity Column - Select this check box to define the column as an identity column.

Allow Nulls - Select this check box to allow nulls in this column.

Default Collation - available for text/character datatypes, lets you specify a default collation.

Default Value - Lets you type a constant value or select a function returning a constant value to serve as the default for the column

Default Binding and **Rule Binding** - Let you bind a rule or default to a column.

Is Sparse - Available to columns that allow NULL values, optimizes storage of the column for null values. This property does apply to the following data types: text, ntext, image, timestamp, user-defined data type, geometry, or geography. Columns with default values, default or rule bindings, cannot be defined as sparse. Computed columns cannot be defined as Sparse, but the columns in the computed expression can be Sparse columns.

Optionally, you can select a column and modify its values or select a column and **Delete** it.

NOTE: For SQL Server 2000, if you create a table with a Column datatype = text., you can set the storage and image values on the Storage Tab of the Tables Editor Storage Tab. When you have a text datatype, the Storage Tab displays a Text In Row box where you can specify the maximum size to be stored.

NOTE: Because the smalldatetime datatype stores dates and time with less precision than the datetime datatype, before outputting you use the CAST or CONVERT functions to convert any boxes with the smalldatetime datatype to either VARCHAR or datetime datatypes. For more information, see SQL Server Books Online, Transact-SQL Reference.

TABLES (SQL SERVER) - INDEXES
Click Add to open the Index Wizard (SQL Server).

TABLES (SQL SERVER) - CONSTRAINTS
Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- Primary Key Wizard (SQL Server)
- Unique Key Wizard (SQL Server)
- Foreign Key Wizard (SQL Server)
- Create Synonym

TRIGGER WIZARD (SQL SERVER)
Triggers are a special type of procedure that automatically fire when defined data modification operations (insert, update, or delete) occur on a target table or view. Triggers fire after an insert, update or delete, but belong to the same transaction as the data modification operation. Triggers can be implemented to enforce business rules or referential data integrity.

Important Notes

- For more information on the syntax for Trigger bodies, consult the Microsoft SQL Server Transact-SQL Documentation.

To create a new trigger using a wizard:

1   [Open an Object Wizard](#) for a trigger.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [Triggers (SQL Server) - Properties](#).

   • **Definition** panel - for details, see [Triggers (SQL Server) - Definition](#).

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

> **NOTE:**   You can use the Trigger Editor opens to create dependencies or alter the trigger statement.

TRIGGERS (SQL SERVER) - PROPERTIES

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type	Select the TABLE or VIEW on which the trigger is to be created.
Parent Schema	Select the owner of the table or view on which the trigger is to be created.
Parent Name	Select the specific table or view in which the trigger is to be created.
Name	Provide a name for the trigger.
Trigger Timing	INSTEAD OF: This is the only option for a View trigger. An INSTEAD OF trigger fires in place of the triggering statement and will not make changes to the data unless the conditions of the INSTEAD OF statement are met first. So, your UPDATE execution statement is replaced by an INSTEAD OF UPDATE statement as a way to enforce particular business rules you establish.
	AFTER: An AFTER trigger fires following the successful completion of the triggering action. So, for example, the trigger would fire after an UPDATE statement has executed and after constraints have been checked and verified.
Fire On Insert	An INSERT trigger must be associated with an INSERT statement. For example, if a data load operation doesn't include an INSERT statement, the trigger won't be invoked.
Fire On Update	An UPDATE trigger can be associated with specific columns of the base table and will only be activated if those columns are updated.
Fire On Delete	A **DELETE** trigger is associated with a DELETE operation.
Encrypted	If you choose to encrypt the trigger, the trigger can't be published as part of SQL Server replication.

4   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TRIGGERS (SQL SERVER) - DEFINITION

Complete the CREATE TRIGGER outline provided by typing or pasting the body of the trigger. Finally, use the **Schedule** or **Execute** buttons to [Create the New Object](#).

UNIQUE KEY WIZARD (SQL SERVER)

Unique keys can enforce logical keys that are not chosen as the primary key. In other words, you can use a unique key to ensure no duplicate values are entered in specific columns that are not a part of the primary key. Although you can only attach one primary key to a table, you can attach multiple unique keys. Also, you can use unique keys on columns that allow null values.

To create a new unique key using a wizard:

1   [Open an Object Wizard](#) for a unique key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - [Unique Keys (SQL Server) - Properties](#).

   • **Columns** panel - [Unique Keys (SQL Server) - Columns](#).

   • **DDL View** panel - for details, see [Create the New Object](#).

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

UNIQUE KEYS (SQL SERVER) - PROPERTIES

When creating or editing a unique key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner and **Table Name**	Choose the owner and name of the table in which the unique key is being created.
Name	Provide a name of the unique key being created.
Build Online	Enabling this feature specifies that the ONLINE=ON clause is used when creating this object and can subsequently be used when rebuilding or dropping this object.
Max degree of parallelism	Lets you specify a MAXDOP index operation value, limiting the number of processors used in parallel plan execution.
Clustered	Enable or disable clustering. With this check box enabled, subsequent rebuild and drop operations offer an online option.
File Group	If you do not specify a filegroup, Microsoft SQL Server creates the unique key in the default filegroup.
Fill Factor	This specifies how full each index page that's storing data should be. The fill factor is a percentage value between 0 and 100.

UNIQUE KEYS (SQL SERVER) - COLUMNS

From the **Column** dropdown, select a column for the primary key and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

USER MESSAGE WIZARD (SQL SERVER)

The User Message wizard lets you create multiple-language versions of user messages, such as errors and warnings, associated with a single message number. Key properties include the severity level and whether the message is automatically written to the NT Event Log.

> **NOTE:** The user messages node only displays under the master database.

To create a new user message using a wizard:

1 Open a creation wizard for a user message. For details, see Open an Object Wizard.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see User Messages (SQL Server) - Properties.

- **Information** panel - for details, see User Messages (SQL Server) - Information.

- **Object Permissions** and **System Permissions** panels - Set Permissions for an Object

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object.

USER MESSAGES (SQL SERVER) - PROPERTIES

When creating or editing a user message, this tab/panel lets you work with the following settings:

Setting	Description
Message Number	Lets you specify a message number. The number must be greater than 50,000. The wizard automatically loads a value one higher than the currently highest used message number, but you can override the value.
Severity	Lets you select a predefined severity level between 001 and 025. The levels and their meanings are indicated in the User Message editor.
Write to NT Event Log	Lets you specify that the message is always written to the Windows NT Event Log.

USER MESSAGES (SQL SERVER) - INFORMATION

Lets you create the different language versions of the text for the message. The first version must be created in us_english. Click the **Add new text for the user message** button, and in the dialog box that opens, select a **Language** of us_english and provide the **Message Text**.

You can subsequently use the same process to create each different language version of the us_english message.

> **NOTE:** You cannot create two versions of a message for the same language.

This tab/panel also lets you edit and delete messages.

USER WIZARD (SQL SERVER)

The User Wizard lets you create a user who will then have access to the database where you are registering him or her. You can also identify the appropriate user group and the system privileges you want to assign to the new user.

To create a new user using a wizard:

1   Open an Object Wizard for a user

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - Users (SQL Server) - Properties.

- **Roles** panel - Users (SQL Server) - Roles.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USERS (SQL SERVER) - PROPERTIES

When creating or editing a user, this tab/panel lets you work with the following settings:

Setting	Description
Login Name	Select the server login associated with this user.
Name	Provide the user name.

USERS (SQL SERVER) - ROLES

For each role to be assigned to the user, select the check box beside that role.

USER DATATYPE WIZARD (SQL SERVER)

User datatypes promote domain consistency by streamlining the definition of commonly used table columns in a database. You can build a customized datatype from system datatypes and bind defaults and rules to it to enhance integrity. When you reference the user datatype in a column, the column assumes all of the properties of the user datatype.

To create a new user datatype using a wizard:

1   Open an Object Wizard for a user datatype.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - User Datatypes (SQL Server) - Properties.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

USER DATATYPES (SQL SERVER) - PROPERTIES

When creating or editing a user datatype, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the user datatype.
Datatype	Provide a name for the datatype.
Type	Select the base datatype.
Size	Provide the size of the datatype.
Allow Nulls	Null has no explicitly assigned value. Null is not equivalent to zero or blank. A value of null is not considered to be greater than, less than, or equivalent to any other value, including another value of null.
Default Binding	Defaults promote data integrity by supplying a default value to a column if the user does not explicitly provide one. They are reusable objects that you can bind to user datatypes.
Rule Binding	Rules promote data integrity by allowing you to validate the values supplied to a column. They are reusable objects that you can bind to user datatypes.

VIEW WIZARD (SQL SERVER)

Views are SQL queries stored in the system catalog that customize the display of data contained in one or more tables. Views behave like tables because you can query views and perform data manipulation operations on them. However, views do not actually store any data. Instead, they depend on data contained in their base tables.

To create a new view using a wizard:

1   [Open an Object Wizard](#) for a view.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [Views (SQL Server) - Properties](#).

   • **Definition** panel - for details, see [Views (SQL Server) - Definition](#).

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

VIEWS (SQL SERVER) - PROPERTIES

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the view. The owner of the view must have SELECT privileges for the tables in the CREATE view statement or DBADM authority on the database that contains the table.
Name	Provide a name for the view.
Encryption	If you select this option, the view will not be published with SQL Server replication.
Schema Binding	When you specify this option, the base table or tables cannot be modified in a way that would affect the view definition. The view definition itself must first be modified or dropped to remove dependencies on the table that is to be modified
Check Condition	When a row is modified through a view, this option makes sure the data remains visible through the view after the modification is committed.
Owner	Select the owner of the view. The owner of the view must have SELECT privileges for the tables in the CREATE view statement or DBADM authority on the database that contains the table.

VIEWS (SQL SERVER) - DEFINITION

Complete the CREATE VIEW statement by typing or pasting in the relevant query.

MYSQL OBJECT WIZARDS

Rapid SQL lets you create MySQL objects using the following wizards:

- Database wizard (MySQL)
- Foreign Keys wizard (MySQL)
- Functions wizard (MySQL)
- Indexes, Primary Keys, or Unique Keys wizard (MySQL)
- Tables wizard (MySQL)
- Users wizard (MySQL)

DATABASE WIZARD (MYSQL)

The MySQL Database Wizard lets you build and execute a basic CREATE DATABASE statement and grant privileges on the database.

To create a new database using a wizard:

1   Open an Object Wizard for a database.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Databases - Properties

- **Privileges** panel - Set Permissions for an Object

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

DATABASES - PROPERTIES

When creating or editing database, this tab/panel lets you work with the following settings:

Setting	Description
Database Name	The database name you choose can be up to 64 characters long. Feel free to use all alphabetic characters or mix in some special or numeric characters. But a name cannot be exclusively special or numeric characters.
Default Character Set	This is the character set for a language or alphabet. If you do not make a selection from the drop-down list (click the arrow to see the choices), the database will use the server's default character set.
Collation	The collation encodes the rules governing character used for a language (like Greek) or an alphabet. The database will use the server's default collation unless you specify otherwise.

> **NOTE:**   It's possible to create databases with different character sets and collations on the same MySQL server.

FOREIGN KEYS WIZARD (MYSQL)

The MySQL Foreign Keys Wizard lets you build and execute an ALTER TABLE statement with an ADD CONSTRAINT option implementing a foreign key.

The wizard makes it easy for you to create a relational link between two tables, thereby speeding queries and giving you faster access to data. By using the Create Foreign Key Wizard you obviate the need for remembering the code underlying the creation process.

To create a new foreign key using a wizard:

1   Open an Object Wizard for a foreign key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

    • **Properties** panel - for details, see Foreign Keys - Properties

    • **Column Mapping** panel - for details, see Foreign Keys - Column Mapping

    • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FOREIGN KEYS - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Group	Setting	Description
Constraint Name	**System Generated** and **User Specified**	The constraint name must be unique. You can either rely on MySQL to assign a unique name or you can specify one.
Constraint State	**On Update** and **On Delete**	Lets you specify ON UPDATE and ON DELETE values of NO ACTION, CASCADE, RESTRICT, or SET NULL.

FOREIGN KEYS - COLUMN MAPPING

1   Under **Referenced Table**, choose the **Database** and then the referenced, or parent, **Table**.

2   Under the **Main Table**, select the referencing **Table**.

3   Under the **Main Table**, select the check box corresponding to the referring column and then under **Referenced Table**, select the check box corresponding to the referenced column.

FUNCTIONS WIZARD (MYSQL)

The MySQL Functions Wizard lets you build and submit a CREATE FUNCTION declaration, specifying a return type and the owning library.

> **NOTE:**   Functions must be written in C or C++, your operating system must support dynamic loading, and you must have compiled mysqld dynamically (not statically).

To create a new function using a wizard:

1 Open an Object Wizard for a function.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Functions - Properties.

   • **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FUNCTIONS - PROPERTIES

When creating or editing a function, this tab/panel lets you work with the following settings:

Required Information	Description
Function Name	In 64 characters or less, you can name your function.
Return Value	STRING, REAL (also known as Double), INTEGER
Shared Object Library	Identify the file that holds the library where your functions are stored.
Aggregate	Check the box if the function you are creating will collapse a large amount of data into a single output, or aggregate the data.

INDEXES, PRIMARY KEYS, OR UNIQUE KEYS WIZARD (MYSQL)

The Indexes Wizard, Primary Keys Wizard, and Unique Keys wizard offer the same steps in creating these MySQL objects.

To create a new index, primary key, or unique key using a wizard:

1 Open an Object Wizard for an index, primary key, or unique key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Indexes, Primary Keys, or Unique Keys - Properties

   • **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

INDEXES, PRIMARY KEYS, OR UNIQUE KEYS - PROPERTIES

When creating or editing an index, primary key, or unique key, this tab/panel lets you work with the following settings:

Setting	Description
Table Name	Select the table on which you want to create the object.
Index Name	Provide a name of up to 64 alphanumeric characters.
Constraint Type	Primary: Each record of a table is identified as uniquely and relates to a foreign key in another table.
	Unique: Ensures that there is no duplication between values in columns where you place this constraint.
	Full Text: Enables the search for several words in arbitrary order in a table.
	Spatial: Allows you to find records that are defined by location, or geometry types.
Index Storage Type	Hash: Used for equality comparisons. Only whole keys can be used to search for a row.
	BTree: Tree data structure that keeps data sorted. BTrees grow from the bottom up as elements are inserted.
	RTree: Tree data structure used for spatial indexes and access to multidimensional information.
Specify Columns in Index	Columns are listed by name and datatype and whether or not they are nullable. As you check selections, the sort order is identified.

TABLES WIZARD (MYSQL)

The MySQL Table Wizard lets you create a basic table definition.

To create a new table using a wizard:

1 [Open an Object Wizard](#) for a table.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see [Tables - Properties](#)

   • **Columns** panel - for details, see [Tables - Columns](#)

   • **Indexes** panel - for details, see [Tables - Indexes](#)

   • **Foreign Keys** panel - for details, see [Tables - Foreign Keys](#)

   • **MERGE tables** panel - for details, see [Tables - MERGE Tables](#)

   • **Privileges** panel - for details, see [Set Permissions for an Object](#)

   • **DDL View** panel - for details, see [Create the New Object](#)

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TABLES - PROPERTIES

When creating or editing a table, this tab/panel lets you work with the following properties:

> **NOTE:** Prior to working with MySQL table properties, you should have a detailed understanding of MySQL table creation options. For access to MySQL documentation, see [Accessing Third Party Documentation](#).

Setting	Description
Table Name	Provide a unique name of up to 64 characters.
Storage Type	Lets you select a storage engine value of MyISAM, InnoDB, BerkeleyDB, ISAM, MRG_MyISAM, HEAP, or MEMORY.
Row Format	Lets you select a row format of DEFAULT (returns the default value if there is one), FIXED (each row is stored with a fixed number of bytes), DYNAMIC (data records have variable length), or COMPRESSED (each record is compressed separately).
Create via SELECT	When you create a table using a SELECT command, the individual columns take their data types from the SELECT command, and don't have to be declared explicitly. Attributes, however, are not carried over.
Unique Key Violations	When you try to insert or update a row that causes a violation, you can choose to IGNORE the violation and continue processing the next row. Or, you can choose REPLACE, which is equivalent to an insert statement if a violation occurs. The row will be replaced with the new data.
Default Character Set	This is the character set for a language or alphabet.
Default Collation	The collation encodes the rules governing character used for a language (like Greek) or an alphabet.
Auto-increment	You can specify the initial value used for the auto-increment sequence. This is possible only for MyISAM tables and for InnoDB tables built in MySQL versions 5.0.3 or greater. For InnoDB tables built in versions prior to 5.0.3, you can manually insert a dummy row with a value one less than the desired value after creating the dummy table; you then delete the dummy row.
Comment	Enter a descriptive comment.
Min Rows and **Max Rows**	Note that if you set the value for either parameter at 0, MySQL removes the setting. MySQL will take care of the row settings in this case.
Average Row Length	The average length (compressed or uncompressed) of table rows in the table space.
Pack Keys	This setting is only enabled for a **Storage Type** of **MyISAM**. It lets you specify a PACK KEYS option value of 0 (keys packing disabled), 1 (smaller indexes), or DEFAULT (pacl long columns only).
Check Sum	This setting is only enabled for a **Storage Type** of **MyISAM**. When selected the CREATE TABLE statement is issued with a CHECKSUM = 1 option. When deselected, the CREATE TABLE statement is issued with a CHECKSUM = 0 option.
Delay Key Write	This setting is only enabled for a **Storage Type** of **MyISAM**. When selected, the CREATE TABLE statement is issued with a DELAY_KEY_WRITE = 1 option. When deselected, the CREATE TABLE statement is issued with a DELAY_KEY_WRITE = 0 option.

TABLES - COLUMNS

When creating or editing a table, this tab/panel lets you modify the table's column setup, as follows:

- Use the **Add Column** button to add the columns to the table. After providing a **Name** for a new column, you can select a **Type** and modify other **Datatype** properties corresponding to the type you selected. When selected, the **Allow Nulls** property corresponds to a CREATE TABLE with a NULL attribute, while unselected it corresponds to a NOT NULL attribute. When available the **Default Value** property corresponds to a DEFAULT attribute.

- Modify a column definition by selecting the column from the column list and modifying values under **Column Attributes**.

- Delete a column definition by selecting the column from the column list and clicking **Delete**.

- Use the arrow buttons to change the position of a selected column.

TABLES - INDEXES

When creating or editing a table, this tab/panel lets you build an associated PRIMARY KEY, UNIQUE KEY, FULL TEXT, or SPATIAL clause that will be included with the CREATE TABLE statement.

- Use the Insert a New Index button to add an index. In the **Index** field, type a name for the index, select a **Constraint Type** of PRIMARY KEY, UNIQUE KEY, FULL TEXT, or SPATIAL, and select an **Index Type** of BTREE, HASH, or RTREE as appropriate to the constraint type. Under **Specify Columns in Index**, select the column or columns that are to make up the index.

- Modify an index definition by selecting the index from the list and modifying the **Index** name, **Constraint Type**, **Index Type** or the columns making up the index.

- Delete an index by selecting the index from the list and clicking the Remove button.

TABLES - FOREIGN KEYS

When creating or editing a table, this tab/panel lets you work with foreign keys for a table:

> **NOTE:** This tab/panel is only available for a table with a **Storage Type** property value of INNODB.

- Use the Insert a New Foreign Key button to add a foreign key. Provide a **Foreign Key Name**, select the referenced database and table from the **Ref. Database** and **Ref. Table** dropdowns, and select **On Delete** and **On Update** actions of NONE, CASCADE, DEFAULT, SET NULL, NO ACTION, or RESTRICT. Under **Main Table**, select the foreign key column and then under **Referenced Table**, select the column that the foreign key is to reference.

- Modify a foreign key definition by selecting the foreign key from the list and changing the name, referenced database or table, delete or update actions, or the column specifications.

- Delete a foreign key by selecting the key from the list and clicking the Remove the selected foreign key button.

TABLES - MERGE TABLES

When creating or editing a table, this tab/panel lets you work with a collection of identical MyISAM tables that are to be used as a single table.

> NOTE: This tab/panel is only available for a table with a **Storage Type** property value of MRG_MyISAM.

- Use the New button to add a MyISAM table to the collection, selecting a table from the associated dropdown.

- Use the Delete button to remove a selected MyISAM table from the collection.

- Use the arrow buttons to change the ordering position of a selected table.

USERS WIZARD (MYSQL)

The MySQL Users Wizard lets you provide basic identification, grant privileges, and specify valid host information for a user definition.

To create a new user using a wizard:

1   Open an Object Wizard for a user.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **User Information** panel - for details, see Users - User Information

- **User Hosts** panel - for details, see Users - User Hosts

- **System Privileges** and **Object Privileges** panels - for details, see Set Permissions for an Object

- **DDL View** panel - for details, see Create the New Object

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USERS - USER INFORMATION

When creating or editing a user definition, this tab/panel lets you work with the following settings:

Setting	Description
User Name	The name of the new user. At this time, the name cannot exceed 16 characters, although the field accepts 34. Rapid SQL can't enforce the discrepancy, so we urge you to show a little restraint. This is the name that is displayed in the User branch of the Explorer tree.
	If you leave this field blank and complete the process, you create an <anonymous> user. Only one anonymous user per datasource is allowed.

Setting	Description
Full Name	OPTIONAL. At this time, the name cannot exceed 16 characters, although the field accepts 34. If you entered a nickname or an alias in the user name field, you can display the user's true identity here.
Description	OPTIONAL. Write a brief description of the user if you want.
Email	OPTIONAL. The new user's email address.
Contact Information	OPTIONAL. The new user's title, address, phone number or whatever contact information you want to include.
User Icon	OPTIONAL. If you want to identify your new user by color or pattern, you can use the icon editor to make a distinguishing mark. You can assign groups of users the same icon or assign each user his or her own icon. This tool helps you distinguish users at a glance. Note: If you don't want to use the head that's preloaded, simply click the red X and start designing your own with the other tools.

USERS - USER HOSTS

When creating or editing a user definition, this tab/panel lets you specify the valid hosts from which the user can connect to the server.

- Add a new host to the list by clicking the New button. Use the associated dropdown to select a valid host-name value (such as localhost or %), overwriting the text as necessary to provide a specific quoted identifier. Provide a password in the **Password** and **Confirm** fields and click **Apply**.

    NOTE: Click **Apply To All** to assign the current password for each host currently in the list.

- Remove a selected host from the list by clicking the Delete button.

ORACLE OBJECT WIZARDS

Rapid SQL lets you create Oracle objects using the following wizards:

- Cluster Wizard (Oracle)
- Database Link Wizard (Oracle)
- Directory Wizard (Oracle)
- Foreign Key Wizard (Oracle)
- Function Wizard (Oracle)
- Index Wizard (Oracle)
- Library Wizard (Oracle)
- Materialized View Wizard (Oracle)
- Materialized View Log Wizard (Oracle)

- [Outline Wizard (Oracle)](#)

- [Package Wizard (Oracle)](#)

- [Primary Key Wizard (Oracle)](#)

- [Procedure Wizard (Oracle)](#)

- [Profile Wizard (Oracle)](#)

- [Role Wizard (Oracle)](#)

- [Rollback Segment Wizard (Oracle)](#)

- [Sequence Wizard (Oracle)](#)

- [Synonym Wizard (Oracle)](#)

- [Table Wizard (Oracle)](#)

- [Tablespace Wizard (Oracle)](#)

- [Trigger Wizard (Oracle)](#)

- [Object Type Wizard (Oracle)](#)

- [Unique Key Wizard (Oracle)](#)

- [User Wizard (Oracle)](#)

- [View Wizard (Oracle)](#)

CLUSTER WIZARD (ORACLE)

The Cluster Wizard lets you create a cluster. A cluster is a schema object that has one or more tables that all have one or more common columns. Rows of one or more tables that share the same value in these common columns are stored together in the database. The related columns of tables stored in a cluster are known as the cluster key.

Important Notes
- To create a cluster, you need the CREATE CLUSTER or CREATE ANY CLUSTER system privilege.

1   On the Explorer, find the schema where you want to add the new cluster.

2   On the Cluster branch, right-click and select New.

    OR

1   On the main toolbar, click Datasource and scroll to Objects

2   Click Clusters and then click New from the toolbar.

The tables below describe the fields you may encounter as you complete the wizard.

Required Information	Description
Who owns the cluster?	Pick an owner
What is the name of the cluster?	Type a unique name
On which tablespace do you want to create the cluster?	Self-explanatory
Add columns that are in this cluster These are the columns that are common between the tables you are "clustering"	**Add** or **Edit** Button - For more information, see <u>Adding or modifying a cluster column</u>. **Drop** Button - Drops the column.
What is the size of this cluster?	This is the estimated number of bytes/KB/MB required by an average cluster key and its associated rows. Do not exceed the size of a data block.
What is the cluster type?	Index: Rows having the same cluster key value are stored together. Each separate cluster key is stored only once in each data block. An indexed cluster is helpful if your clustered tables might grow unpredictably. Hash: Rows with the same hash key value are stored together. This is helpful if the tables are static.
If this is a hash cluster, what is the number of hash keys?	Type the number of hash keys. Oracle will round the value up to the nearest prime number.
If this is a hash cluster, what is the hash function?	Oracle uses a hash function to generate a distribution of numeric values, called hash values, which are based on specific cluster key values. The key of a hash cluster, like the key of an index cluster, can be a single column or composite key (multiple column key). To find or store a row in a hash cluster, Oracle applies the hash function to the row's cluster key value. The resulting hash value corresponds to a data block in the cluster, which Oracle then reads or writes on behalf of the issued statement. Default is the Oracle internal hash function, otherwise specify the hash expression you want to use.
How many transaction entries are allowed for each data block in the cluster?	Each transaction that updates a data block requires a transaction entry. Initial (1-255): The initial parameter ensures that a minimum number of concurrent transactions can update a data block, avoiding the overhead of allocating a transaction entry dynamically. Maximum (1-255): The maximum parameter limits concurrency on a data block.
What is the percent of space reserved for future updates?	Percent Free (0-99): This sets the percentage of a data block to be reserved for possible row updates that are included in the block. The value you set is the percent kept free.

Required Information	Description
What is the minimum percentage of used space that Oracle maintains for each data block?	The storage parameter lets you tune performance by minimizing the occurrence of row migration and chaining caused by update operations that extend the length of rows stored on the data block. Percent Used (1-99)
How large are the cluster's extents?	The unit of space allocated to an object whenever the object needs more space. Initial Extent - The initial space extent (in bytes) allocated to the object. Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
Specify the number of free lists	Free lists are lists of data blocks that have space available for inserting rows. Identifying multiple free lists can reduce contention for free lists when concurrent inserts take place and potentially improve the performance of the cluster. Free Lists: The default and minimum value is 1; this option should be set higher if multiple processes access the same data block.
Specify the number of free list groups (specify only if you are using the parallel server option)	This is the number of groups of free lists for the database objects being created.
Define a default buffer pool for this cluster	Default - Select to retain the default. Keep - Select to retain the object in memory to avoid I/O conflicts. ORACLE 8i ONLY: Recycle - Select to rid data blocks from memory as soon as they are no longer in use.
Oracle's parallel query option	The parallel server query option lets you process queries using many query server processes running against multiple CPUs, which provides substantial performance gains such as reducing the query completion time.
Choosing Cache	Cache: This keeps the data block in memory by placing it at the most recently used end. This option is useful for small lookup tables. No Cache

ADDING OR MODIFYING A CLUSTER COLUMN

The Add or Modify Cluster Column dialog box lets you manage cluster columns. You can open the dialog box in the Oracle Cluster Wizard. For details, see Cluster Wizard (Oracle).

The table below describes the options and functionality on the Add or Modify Cluster Column dialog box:

Option	Description
Column Name	Lets you type the column name.
Datatype	Lets you select the datatype for the cluster. If you select CHAR, RAW or VARCHAR2, in the Width box, type the width value. If you select NUMBER, in the Width box, type the width value and in the Scale box, type the scale value.

Completing the Add or Modify Cluster Column Dialog Box

To complete this dialog box, do the following:

1   In the *Add Cluster Column* dialog box, in the *Column Name* box, type the column name.

2   Click the *Datatype* list, click the datatype for the cluster.

      • If you clicked *CHAR*, *RAW* or *VARCHAR2*, in the *Width* box, type the width value.

      • If you clicked *NUMBER*, in the *Width* box, type the width value and in the *Scale* box, type the scale value.

3   Click the Add button.

4   To continue adding columns to the cluster, repeat steps 1-3.

5   When you finish adding columns, click Close.

      Rapid SQL closes the Add Cluster Column dialog box.

For more information, see <u>Adding or modifying a cluster column</u>.

DATABASE LINK WIZARD (ORACLE)

A database link specifies a communication path from one database to another. If you're creating a link to a remote database, a database session is established in the remote database on behalf of the local application request. By creating either a public or private database link, you can determine which schema on the remote database the link will establish connections to by creating fixed, current, and connected database links. By creating a link you can reuse connectivity instructions each time you connect to the remote database.

To Open the Database Link Wizard

1   On the Explorer, find the schema where you want to add the new database link.

2   On the Database Links branch, right-click and select New.

      OR

1   On the main toolbar, click Datasource and scroll to Objects

2   Click Database Links and then click New from the toolbar.

The table that follows describes the fields you will encounter as you complete the wizard.

   NOTE: To create a public database link, you need CREATE PUBLIC DATABASE LINK privileges.

Required Information	Description
What is the name of the database link?	Create a unique name

Required Information	Description
Should the database link be made public?	A public link is a link on a local database that's accessible to all users on that database.
	If you keep the database link private by selecting No, a link is created in a specific schema of the local database and only the owner of the link can use it to access database objects in the corresponding remote database.
What is the name of the remote user?	Self-explanatory
What is the remote user's password?	Create a password for the remote user.
What is the connection string?	Self-explanatory.

DIRECTORY WIZARD (ORACLE)

A directory object specifies an alias for a directory on the server file system where external binary file LOBs and external table data are located. The wizard completes a CREATE DIRECTORY statement from the information you supply. The Directory Wizard prompts you to name the directory and provide the full-qualified directory path.

To Open the Directory Wizard

1   On the Explorer, find the Datasource where you want to create a directory and expand the Storage node.

2   Right-click Directories and select New.

The table that follows describes the fields you will encounter as you complete the wizard.

Required Information	Description
What is the name of the directory?	Type a meaningful name for the directory.
What is the directory path?	Type the full path name of the outside operating system directory that you want to alias in the directory (for example, /Video/Library/G_Rated). NOTE: Oracle trusts that directory you're specifying exists. The onus is on you to make sure it's valid and in the correct format (as required by your operating system).

After you have created the Directory, you can give other users Read or Write privileges by opening the new Directory on the Directories node, and making changes at the Privileges tab.

FOREIGN KEY WIZARD (ORACLE)

A foreign key value in one table (child table) refers to a primary key value in another table (parent table). For example, the Author Name column in a publisher's database may be the primary key for a table of addresses that includes the Author Name column. If an author isn't included in the parent table, you can't add the address to the dependent address table. So foreign keys enforce referential integrity between tables by verifying the existence of foreign key values in the parent table before letting you insert or update foreign key values in the child table. In other words, a foreign key is an integrity constraint that requires each value in one table's column to match a value in a related table's data.

To create a new foreign key using a wizard:

1   Open an Object Wizard for a foreign key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - Foreign Keys (Oracle) - Properties.

- **Column Mapping** panel - Foreign Keys (Oracle) - Column Mapping.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FOREIGN KEYS (ORACLE) - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner	The owner of the table where the foreign key is being created.
Table Name	This is the table where the foreign key link originates--the child table.
Name	Lets you select a constraint name. System Generated Name - DB2 automatically generates a name. User Specified Constraint Name - You type the name.
Enabled	Enable or disable the foreign key. Enabled ensures that all data modifications to a given table (or tables) satisfy the conditions of the constraints. When disabled, the constraint is temporarily not operational.
Delete Rule	Select an action: **NO ACTION** - ensures that referenced values cannot be updated or deleted if to do so would violate referential integrity. **CASCADE** permits a referenced row in a child table to be deleted/updated if it is deleted/updated in the parent table. A row in the child table is SET NULL when rows in the parent table are deleted/updated.

FOREIGN KEYS (ORACLE) - COLUMN MAPPING

Under **Referenced Table**, choose the **Owner** and then the **Name** of the referenced, or parent, table.

Under the **Main Table**, select check boxes corresponding to the columns that are to reference columns in the referenced table. Then, under **Referenced Table**, select the corresponding column check boxes.

FUNCTION WIZARD (ORACLE)

Functions are subroutines that you define and are useful for reusing application logic. You can use functions to determine the best methods for controlling access and manipulation of the underlying data contained in an object. A function returns a value, unlike a procedure, which does not.

To create a new function using a wizard:

1   Open an Object Wizard for a function.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - Functions (Oracle) - Properties.

   • **Definition** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

FUNCTIONS (ORACLE) - PROPERTIES

When creating or editing a function, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the function.
Name	Provide a name for the function.

FUNCTIONS (ORACLE) - DEFINITION

Complete the CREATE FUNCTION outline provided by typing or pasting the body of the function. Finally, use the **Execute** button to Create the New Object.

INDEX WIZARD (ORACLE)

Indexes are optional structures associated with tables. You can create indexes specifically to speed SQL statement execution on a table. When properly used, indexes are the primary means of reducing disk I/O. Indexes are logically and physically independent of the data in the associated table. Unique Indexes guarantee that no two rows of a table have duplicate values in the columns that define the index.

> **NOTE:** The Index Wizard varies slightly in content based on the version of Oracle to which you are connected.

> **NOTE:** To create indexes in your own schema, you need INDEX privileges on the target table. To create indexes in other schema, you need CREATE ANY INDEX privileges.

> **NOTE:** For Oracle 8i or later, you can place a unique key constraint on an Index-Organized table.

> **TIP:** Index-organized tables take up less storage space and quickly access table rows. Index-organized tables stores rows in primary key order reducing the amount of storage space needed.

> **TIP:** An advantage of using index-organized tables is that the tables use less memory because key columns are not duplicated in the table and index. Rapid SQL stores the remaining non-key columns in the index structure.

To create a new index using a wizard:

1   Open an Object Wizard for an index.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - Indexes (Oracle) - Properties.

   • **Columns** panel - Indexes (Oracle) - Columns.

   • **Storage** panel - Indexes (Oracle) - Storage.

   • **Partition** panel - Indexes (Oracle) - Partition.

   • **Definition** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

INDEXES (ORACLE) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner and **Table Name**	Choose the owner and name of the table in which the index is being created.
Owner and **Name**	Choose the owner and provide the name of the index being created.

Setting	Description
Index Type	NONUNIQUE - In a non-unique index, the ROWID is treated as part of the key. Oracle treats a constraint as deferrable.
	UNIQUE - Select if the index is a unique constraint.The values in the indexed columns must be distinct.
	BITMAP - Widely used in data warehousing environments. The environments typically have large amounts of data and ad hoc queries, but a low level of concurrent DML transactions.
No Sort	Enable this feature if the rows in the table already stored in ascending order. This increases the speed of the index creation process. Oracle does not sort the rows.
Logging	Enabling logs this operation to the redo file.
Reverse	Enabling this feature stores the bytes of the index block in reverse order and excludes the ROWID. The ROWID is a globally unique identifier for a row in a database. It is created at the time the row is inserted into a table, and destroyed when it is removed from a table.
Function-Based	Permits the results of known queries to be returned much more quickly. When you select this option, you are asked for the expression that governs the function-based index you are creating.
Invisible (Oracle 11g)	Adds an INVISIBLE keyword to the DDL generated to create or edit this index. The optimizer ignores an invisible index unless the OPTIMIZER_USE_INVISIBLE_INDEXES is set to TRUE at the session or system level.
No Parallel Execution	The parallel server query option lets you process queries, using many query server processes, running against multiple CPUs. This option provides substantial performance gains such as reduction of the query completion time.
	After creation, ALTER INDEX for NOPARALLEL execution - when you use multiple query servers and you select this option, the parallel query option remains in place, but parallel processing will be removed. If, for example, multiple users on numerous nodes are modifying the same small set of data, the cost of synchronization from the parallel processing may have an unnecessarily large drag on throughput.
Parallel Degree	The value you select indicates the number of query server processes that should be used in the operation.
Parallel Instances	The value you select indicates how you want the parallel query partitioned between the Parallel Servers.

INDEXES (ORACLE) - COLUMNS

From the **Column** dropdown, select a column for the index and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

INDEXES (ORACLE) - STORAGE

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Data Block Storage group	Select the DEFAULT **Tablespace** only if you are creating a local partitioned index and want the partitions in the same tablespace as the partitions in the underlying table. (Each partition of a local index is associated with one partition of the table. Oracle can then keep the index partitions in synch with table partitions.)
	A transaction entry is needed for each INSERT, UPDATE, DELETE, etc. statement that accesses one or more rows in the block. Transaction entries in many operating systems require approximately 23 bytes.
	Percent Free identifies how much space you want to allocate for new rows or updates to existing rows.
	Initial Transactions ensures that a minimum number of concurrent transactions can update an index block, avoiding the overhead of allocating a transaction entry dynamically.
	Maximum Transactions limits concurrency on an index block.
Extents group	An extent is the unit of space allocated to an object whenever the object needs more space.
	Initial Extent - The initial space extent (in bytes) allocated to the object.
	Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
	Percentage Increase - Lets you type the percentage.
	NOTE: You should be careful when setting Percent Increase because it magnifies how an object grows and, therefore, can materially affect available free space in a tablespace.
	Minimum Extents - For a dictionary managed tablespace, this is the total number of extents to be allocated when the index is first created. For a locally managed tablespace, this is simply the initial amount of space allocated.
	Maximum Extents - For a dictionary managed tablespace, this is the total number of extents that can ever be allocated to the index. In a locally managed tablespace, the database will automatically manage the extents.
Freelists group	**Free lists** let you manage the allocation of data blocks when concurrent processes are issued against the index. You can potentially improve the performance of the index by identifying multiple free lists, which can reduce contention for free lists when concurrent inserts take place.
	The default and minimum value is 1. You should increase this number if multiple processes access the same data block.
	Free List Groups is the number of groups of free lists.
	NOTE: This option is only applicable for the parallel server option.
Buffer Pool	DEFAULT - Choose this if you want to use the default bufferpool.
	KEEP - Use this to retain the object in memory to avoid I/O conflicts. This type of bufferpool stores frequently referenced data blocks in a separate cache.
	RECYCLE - Select this option to save cache space by ridding data blocks from memory as soon as they are no longer in use.

INDEXES (ORACLE) - PARTITION

Clicking **Create Partition** opens a wizard that lets you create a partition.

JOB QUEUE WIZARD (ORACLE)

Job Queues are built-in mechanisms that let you schedule a variety of SQL-based or command-line driven tasks.

The Job Queue Wizard lets you:

- Specify the PL/SQL code that will run in the job.

- Specify when the job will run, if it will run again, and if it should be enabled to run.

To Open the Job Queue Wizard

1   On the Explorer, find the datasource where you want to create an Oracle job and expand the Instance node.

2   Right-click the Oracle Job Queue node, and select New.

The table that follows describes the fields you may encounter using this wizard:

Required Information	Description
Enter the PL/SQL code you would like submitted	Type PL/SQL code or retrieve a previously saved PL/SQL script.
No Parse/Parse	Parsing can be expensive resource-wise, so think carefully before you choose the parse option. For a parse call, Oracle checks the syntactic and semantic validity of the SQL statement, whether the process has the correct permissions, and allocates a private SQL area for the statement. Oracle will also check to see whether the statement exists in a shared library cache.
When would you like for the job to begin execution?	Self-explanatory.
Would you like to have the job run on an ongoing basis?	Self-explanatory.
Would you like to have the job submitted as disabled?	Self-explanatory.

LIBRARY WIZARD (ORACLE)

Libraries are an object type introduced in Oracle8 that represent a call to an operating system shared library cache. After the call is made, libraries can be used by SQL or PL/SQL to link to external procedures or functions. Libraries are only to be used on operating systems that support shared libraries and dynamic linking. Libraries serve as pointers or aliases to physical operating system shared library files and do not exist as a physical object; rather they rely on the physical existence of the files in the external operating system library to which they refer. To access the function or procedures stored in the library, you need execute privileges at the operating system level where the shared library resides.

- To create a library in your own schema, you need CREATE ANY LIBRARY privileges. To use the functions or procedures stored in the library, you need object EXECUTE privileges on the library.

To Open the Library Wizard

1 On the Explorer, find the datasource where you want to create a Library and expand the Schema node.

2 Right-click the Libraries node, and select New.

The table that follows describes the fields you may encounter using this wizard:

Required Information	Description
Who owns the library?	Self-explanatory.
What is the name of the library?	Self-explanatory.
What is the file specification?	Lets you type the file name and location. You must type the complete location (for example, D:\Embarcadero\ETLIB21D.DLL).

MATERIALIZED VIEW WIZARD (ORACLE)

A materialized view gives you indirect access to table data by storing a query's results in a separate schema object. Essentially, a materialized view is a database object that contains the results of a query.

The Materialized View Wizard lets you:

- Specify the materialized view owner and to name the materialized view.

- Specify the materialized view's refresh configuration.

- Place the materialized view on a tablespace and specify the query that should be used to populate the materialized view.

- Specify how Oracle should allocate data blocks to store the materialized view.

- Specify how Oracle should manage the growth of the materialized view.

- Specify if Oracle updates the materialized view, register an existing table, and specify how to populate a materialized view.

- Specify if the data for the materialized view is cached, if you want the updates logged, and to specify a number of threads for a parallel operation.

- Specify rollback segments, and enable query rewrites.

To Open the Materialized View Wizard

1 On the Explorer, find the datasource where you want to create a Materialized View and expand the Schema node.

2 Right-click the Materialized Views node, and select New.

The table that follows describes the fields you may encounter as you complete the wizard:

Required Information	Description
Who owns the materialized view?	Self-explanatory
What is the name of the materialized view?	Self-explanatory.
How should the materialized view be refreshed?	Fast - Using the information logged on the materialized view logs or a partition maintenance operation, the refresh applies incremental changes. See Fast Refresh Requirements for more information.
	Complete - This refresh recalculates the materialized view's defining query.
	Force - Applies fast refresh when feasible, otherwise uses a complete refresh.
	Never - Materialized view will not be refreshed with the optional refresh mechanisms.
Choose a refresh mechanism	On Demand - This option requires that all refreshes be manually executed.
	On Commit - Select to refresh the materialized view whenever Oracle processes a transaction. Only select this option for materialized views on single table aggregates and materialized views containing joins.
	Automatically - Select to refresh the materialized view automatically. In the On this date: boxes select a time and date, and then select a refresh amount and a unit of time.
Where do you want to place the materialized view?	Select the tablespace where you want the materialized view placed.
What is the materialized view query?	Type the SQL query to be used to populate and to refresh the materialized view.
Select a refresh method	Primary Key - A primary key's values uniquely identify the rows in a table. Changes are propagated according to row changes as identified by the primary key value of the row. Only one primary key can be defined for each table.The primary key of the master table is the basis for this refresh option, which is the default option.
	ROWID - A globally unique identifier for a row in a database based on the physical row identifiers. A RowID is created at the time the row is inserted into a table, and destroyed when it is removed from a table. ROWID materialized views cannot contain distinct or aggregate functions or GROUP BY subqueries, joins and set operations.
How many transaction entries are allowed for each datablock in the materialized view?	A transaction is a logical unit of work that contains one or more SQL statements. Each transaction that updates a data block requires a transaction entry.
	Initial (1-255) - Ensures that a minimum number of concurrent transactions can update a data block, avoiding the overhead of allocating a transaction entry dynamically.
	Maximum (1-255) - Limits concurrency on a data block.
What is the percent of space reserved for future updates?	Percent Free (0 99) - This sets the percentage of a data block to be reserved for possible row updates that are included in the block. The value you set is the percent kept free.

Required Information	Description
What is the minimum percentage of used space that Oracle maintains for each datablock?	Percent Used (0-99) - Set the amount of space to be used for each datablock. NOTE: The sum of percent free and the percent used cannot exceed 100.
How large are the materialized views extents?	The unit of space allocated to an object whenever the object needs more space. An extent is a specific number of contiguous data blocks set aside for storing a specific type of information. Initial Extent (KB) - The initial space extent (in bytes) allocated to the object. Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
How many extents should be allocated to the materialized view?	Minimum Extents - The appropriate minimum extents value for the object. Maximum Extents - The appropriate maximum extents value for the object.
What is the growth rate for sizing additional materialized views?	Percent Increase - Magnifies how an object grows and can materially affect available free space in a tablespace. Select a value in the corresponding box.
Can the materialized view be updated?	Yes/No
Do you want to register a prebuilt table to the view?	Yes/No. This option is particularly useful for registering large materialized views in a data warehousing environment.
Should the materialized view be immediately filled?	Yes/No: Select Yes if you want the materialized view populated immediately or during the next refresh operation.
Should data for the materialized view be cached?	Yes/No: Select if you want Oracle to put data you access frequently at the most recently used end of the list in the buffer cache when a full table scan is performed. This option is useful for small lookup tables.
Do you want updates to be logged?	Yes/No.
Do you want to specify the number of threads used in a parallel operation?	Parallel processes means that multiple processes work simultaneously to run a single statement. This can cut the amount of time it takes to get a response. Specify the degree of parallelism if you so desire. If you leave the default at 1, the operation will not be "parallelized."
Would you like to specify rollback segments to be used for the materialized view refresh?	A rollback segment temporarily stores old data that has changed in a SQL statement transaction until it is committed. The "before" image of the database, as it were. Local Rollback Segment - Default indicates that Oracle will select the rollback segment to use on the local machine. Master Rollback Segment - Specify the remote rollback segment used at the remote master site for the individual materialized view.
Is the materialized view eligible for query rewrite?	Select to enable the materialized view for query rewrite. Only enable query rewrite if expressions in the statement are repeatable.

Required Information	Description
Do you want to partition this materialized view?	Yes/No Partitioning methods available are: Range: Data is mapped to partitions based on ranges of column values. This is the default. Composite: Based on the range method of partitioning, you can create subpartitions within each partition. Hash: Data is distributed evenly over a specified number of partitions. Data need not fit into a logical range. List: You control explicitly how rows map to partitions. List partitions allow you to group and organize unrelated sets of data.
Do you want to enable Row Movement?	Yes/No Enabling row movement allows you to specify whether Oracle can move a table row when you are compressing a table or performing an update on partitioned data.
Select the partitioning columns	Self-explanatory
Select the subpartitioning method	Self-explanatory
Select the subpartitioning columns	Self-explanatory
Hash Partitioning methods	None Partition Definition: Specify number of partitions and (optionally) tablespaces Specify individual partitions by name and (optionally) tablespaces
Create list/ordered list of partitions	Self-explanatory The **Add Partition** dialog box may open.
Specify number of subpartions	Self-explanatory Click Add, Insert, or Edit to open a dialog box that lets you work with subpartition properties.
Select the default tablespaces to contain the subpartitions (optional)	Self-explanatory

Fast Refresh Requirements

	When the Materialized View has:		
	Only Joins	Joins and Aggregates	Aggregate on a Single Table
Detail tables only	X	X	X
Single table only			X
Table Appears only once in the FROM list	X	X	X

	When the Materialized View has:		
No non-repeating expressions like SYSDATE and ROWNUM	X	X	X
No references to RAW or LONG RAW	X	X	X
No GROUP BY	X		
Rowids of all the detail tables must appear in the SELECT list of the query	X		
Expressions are allowed in the GROUP BY and SELECT clauses provided they are the same		X	X
Aggregates allowed but cannot be nested		X	X
AVG with COUNT		X	X
SUM with COUNT			X
	Only Joins	Joins and Aggregates	Aggregate on a Single Table
VARIANCE with COUNT and SUM		X	X
STDDEV with COUNT and SUM		X	X
WHERE clause includes join predicates which can be ANDed bit not ORed.	X	X	
No WHERE clause			X
No HAVING or CONNECT BY	X	X	X
No subqueries, inline views, or set functions like UNION or MINUS	X	X	X
COUNT(*) must be present			X
No MIN and MAX allowed			X
If outer joins, then unique constraints must exist on the join columns of the inner join table	X		
Materialized View logs must exist and contain all columns referenced in the materialized view and have been created with the LOG NEW VALUES clause			X
Materialized View Logs must exist with rowids of all the detail tables	X		
Non-aggregate expression in SELECT and GROUP BY must be straight columns			X
DML to detail table	X		X

	When the Materialized View has:		
Direct path data load	X	X	X
ON COMMIT	X		X
ON DEMAND	X	X	X

MATERIALIZED VIEW LOG WIZARD (ORACLE)

Materialized view logs are tables that maintain a history of modifications to the master table, and they are used to refresh simple materialized views. When you create a materialized view log, Oracle automatically creates a log table to track data changes in the master table and a log trigger to maintain the data in the log table. A log can refresh the materialized view incrementally and is therefore often less time-consuming than a complete refresh.

The Materialized View Log Wizard lets you:

- Specify the materialized view log owner and master table.

- Select refresh types and select column filters.

- Specify how Oracle should allocate data blocks to store the materialized view log.

- Specify how Oracle should manage the growth of the materialized view.

- Specify if you want the data for the materialized view log cached, if you want updates logged, and to enable parallel query.

- Specify if you want the log to hold new values.

To Open the Materialized View Log Wizard

1 On the Explorer, find the datasource where you want to create a Materialized View Log and expand the Schema node.

2 Right-click the Materialized View Log node, and select New.

The table that follows describes the fields you may encounter as you complete the wizard:

Required Information	Description
Who owns the materialized view log's master table?	Self-explanatory
Which table will serve as the materialized view log's master table?	Self-explanatory
On which tablespace do you want to place the log?	Self-explanatory

Required Information	Description
Which refresh types would you like to use?	Primary Key - The log records changes to the master table based on the primary key of affected rows. A primary key's values uniquely identify the rows in a table. Changes are propagated according to row changes as identified by the primary key value of the row. Only one primary key can be defined for each table.The primary key of the master table is the basis for this refresh option, which is the default option.
	ROWID - The log records changes to the master table based on the RowID of the affected rows. A ROWID is a globally unique identifier for a row in a database based on the physical row identifiers. A ROWID is created at the time the row is inserted into a table, and destroyed when it is removed from a table. ROWID materialized views cannot contain distinct or aggregate functions or GROUP BY subqueries, joins and set operations.
Optional: Select any filter column(s) to be recorded in the materialized view log.	A filter column is a column whose values you want to be recorded in the materialized view log for any rows that are changed. You can specify only one primary key, one ROWID, and one filter column list per materialized view log.
How many transaction entries are allowed for each data block in the materialized view log?	A transaction is a logical unit of work that contains one or more SQL statements. Each transaction that updates a data block requires a transaction entry.
	Initial (1-255) - Ensures that a minimum number of concurrent transactions can update a data block, avoiding the overhead of allocating a transaction entry dynamically.
	Maximum (1-255) - Limits concurrency on a data block.
What is the percent of space reserved for future updates?	Percent Free (0 99) - This sets the percentage of a data block to be reserved for possible row updates that are included in the block. The value you set is the percent kept free.
What is the minimum percentage of used space that Oracle maintains for each data block?	Percent Used (0-99) - Set the amount of space to be used for each datablock.
	NOTE: The sum of percent free and the percent used cannot exceed 100.
How large are the materialized view log's extents?	Initial Extent (KB) -The default is the value specified for the tablespace where the materialized view log resides.
Should the data for a materialized view log be cached?	Yes/No - Select Yes if you want Oracle to put data you access frequently at the most recently used end of the least recently used list in the buffer cache when a full table scan is performed. This option is useful for small lookup tables.
	No indicates that your most frequently accessed data blocks are put at the least recently used end of the least recently used list of the buffer cache.
Do you want updates to be logged?	Yes/No
Do you want to enable parallel query for the log?	Degree. The integer is number of parallel threads used in the parallel operation.
	The Parallel server query option lets you process queries using many query server processes running against multiple CPUs. This option provides substantial performance gains such as reduction of the query completion time.
Should the log hold new values?	Yes/No - Yes indicates both old and new values should be saved in the materialized view log.
	No disables recording of new values in the log. This is the default.

OUTLINE WIZARD (ORACLE)

Outlines are a set of results for the execution plan generation of a particular SQL statement. When you create an outline, plan stability examines the optimization results using the same data used to generate the execution plan. That is, Oracle uses the input to the execution plan to generate an outline, and not the execution plan itself.

> **NOTE:** To create an outline, you must have CREATE ANY OUTLINE system privileges.

To create a new outline using a wizard:

1   Open an Object Wizard for an outline.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   - **Properties** panel - Outlines (Oracle) - Properties.

   - **Definition** panel - Outlines (Oracle) - Definition.

   - **DDL** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

OUTLINES (ORACLE) - PROPERTIES

When creating or editing an outline, this tab/panel lets you work with the following settings:

Setting	Description
Stored Outline Name	Provide a unique name for the outline.
Category	The category is a name you want to use to group stored outlines. (You can type over the word Default that appears automatically.)

OUTLINES (ORACLE) - DEFINITION

Type the SQL statement you want to store as an outline.

> **NOTE:** The only SQL statements possible with stored outlines are SELECT, DELETE, UPDATE, INSERT…SELECT, and CREATE TABLE…AS SELECT.

PACKAGE WIZARD (ORACLE)

A package is a collection of related program objects stored together in the database. A package specification or package header, declares variables, constants, etc., that are visible outside the package's immediate scope. The package body defines objects declared by the specification but are not visible to applications outside the package. Packages contain all the information needed to process SQL statements from a single source file. You can use packages to process and call batches of SQL.

To Open the Package Wizard

1 On the Explorer, find the datasource where you want to create a package and expand the Schema node.

2 Right-click the Packages node, and select New.

You're asked to name an owner for the package and give it a name. When you click Finish, the Packages editor opens to the header tab where you indicate any package specifications and create the package body. For more information, see Package Bodies Editor (Oracle).

PRIMARY KEY WIZARD (ORACLE)

Primary key (constraint)s are a set of table columns that can uniquely identify every row of a table. No fields that are a part of the primary key can have null values, and each table can have only one primary key.

To create a new primary key using a wizard:

1 Open an Object Wizard for a primary key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - Primary Keys (Oracle) - Properties.

- **Columns** panel - Primary Keys (Oracle) - Columns.

- **Storage** panel - Primary Keys (Oracle) - Storage.

- **Partition** panel - Primary Keys (Oracle) - Partition.

- **DDL** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

PRIMARY KEYS (ORACLE) - PROPERTIES

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner and **Table Name**	Choose the owner and name of the table in which the primary key is being created.
Name	Provide a name for the primary key being created.
No Sort	Enable this feature if the rows in the table already stored in ascending order. This increases the speed of the index creation process. Oracle does not sort the rows.
Logging	Enabling logs this operation to the redo file.
Reverse	Enabling this feature stores the bytes of the index block in reverse order and excludes the ROWID. The ROWID is a globally unique identifier for a row in a database. It is created at the time the row is inserted into a table, and destroyed when it is removed from a table.

Setting	Description
Validate	Enabling this option indicates that existing data is checked against the constraint when the primary key is enabled. Leaving it disabled indicates that only new data is to be checked against the constraint.
Deferrable	Dictates whether constraint checking can be deferred until the end of a transaction.
Deferred	This option is enabled only if you enabled the **Deferrable** option. Select IMMEDIATE to have the constraint checked at the end of every DDL statement. Select DEFERRED to have the constraint checked only at the end of a transaction.
Enabled	Enables or disables the primary key.

PRIMARY KEYS (ORACLE) - COLUMNS

From the **Column** dropdown, select a column for the index and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

PRIMARY KEYS (ORACLE) - STORAGE

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Data Block Storage group	Select the DEFAULT **Tablespace** only if you are creating a local partitioned index and want the partitions in the same tablespace as the partitions in the underlying table. (Each partition of a local index is associated with one partition of the table. Oracle can then keep the index partitions in synch with table partitions.)
	A transaction entry is needed for each INSERT, UPDATE, DELETE, etc. statement that accesses one or more rows in the block. Transaction entries in many operating systems require approximately 23 bytes.
	Percent Free identifies how much space you want to allocate for new rows or updates to existing rows.
	Initial Transactions ensures that a minimum number of concurrent transactions can update an index block, avoiding the overhead of allocating a transaction entry dynamically.
	Maximum Transactions limits concurrency on an index block.
Extents group	An extent is the unit of space allocated to an object whenever the object needs more space.
	Initial Extent - The initial space extent (in bytes) allocated to the object.
	Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
	Percentage Increase - Lets you type the percentage.
	NOTE: You should be careful when setting Percent Increase because it magnifies how an object grows and, therefore, can materially affect available free space in a tablespace.
	Minimum Extents - For a dictionary managed tablespace, this is the total number of extents to be allocated when the index is first created. For a locally managed tablespace, this is simply the initial amount of space allocated.
	Maximum Extents - For a dictionary managed tablespace, this is the total number of extents that can ever be allocated to the index. In a locally managed tablespace, the database will automatically manage the extents.

Setting	Description
Freelists group	**Free lists** let you manage the allocation of data blocks when concurrent processes are issued against the index. You can potentially improve the performance of the index by identifying multiple free lists, which can reduce contention for free lists when concurrent inserts take place.
	The default and minimum value is 1. You should increase this number if multiple processes access the same data block.
	Free List Groups is the number of groups of free lists.
	NOTE: This option is only applicable for the parallel server option.
Buffer Pool	DEFAULT - Choose this if you want to use the default bufferpool.
	KEEP - Use this to retain the object in memory to avoid I/O conflicts. This type of bufferpool stores frequently referenced data blocks in a separate cache.
	RECYCLE - Select this option to save cache space by ridding data blocks from memory as soon as they are no longer in use.

PRIMARY KEYS (ORACLE) - PARTITION

Clicking **Create Partitio**n opens a wizard that lets you create a partition.

PROCEDURE WIZARD (ORACLE)

Procedures are a reusable block of PL/SQL, stored in the database, that applications can call. Procedures streamline code development, debugging, and maintenance by being reusable. Procedures enhance database security by letting you write procedures granting users execution privileges to tables rather than letting them access tables directly. Procedures do not return values, unlike functions.

> **NOTE:** To create a procedure in your own schema, you need CREATE PROCEDURE privileges. To create a procedure in someone else's schema, you need CREATE ANY PROCEDURE privileges.

To create a new procedure using a wizard:

1 Open an Object Wizard for a procedure.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - Procedures (Oracle) - Properties.

- **Definition** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

PROCEDURES (ORACLE) - PROPERTIES

When creating or editing a procedure, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the procedure.
Name	Provide a name for the procedure

PROCEDURES (ORACLE) - DEFINITION

Complete the CREATE PROCEDURE outline provided by typing or pasting the body of the procedure. Finally, use the **Execute** button to [Create the New Object](#).

PROFILE WIZARD (ORACLE)

Profiles are a mechanism for allocating system and database resources to users. In essence, a profile is a set of limits with a name attached to them. If the profile is active, Oracle will limit use and resources to the limits defined by the profile.

The Profile Wizard lets you:

- Name the profile.

- Set composite limit.

- Set session limits for SGA shared pool.

- Set limits on total connection time per session and Idle time per session.

- Set limits on concurrent sessions per user, CPU time per session, and data blocks read per session.

- Set limits on CPU time per call, and number of data blocks read for a call to process an SQL statement.

- Set the number of failed login attempts, and the days an account locks.

    NOTE: To create a profile, you need the CREATE PROFILE system privilege.

The Default option is subject to the limit specified for that particular resource. The default profile initially permits unlimited resources. Limits to the default profile can be made using an Alter statement.

The Unlimited option allows the user with that profile to take use unlimited amounts of that resource.

The Other options take on the resource limits that you indicate.

To Open the Profile Wizard

1   On the Explorer, find the datasource where you want to create a procedure and expand the Security node.

2   Right-click the Profile node, and select New.

The table that follows describes the fields you may encounter as you complete the wizard.

Required Information	Description
What is the name of the profile?	Write a name that's 30 characters or less
What is the composite limit on resources per session?	You can set a single composite limit for all resource limits in a profile in addition to setting specific resource limits for a profile. Explicit and composite limits can peaceably coexist. The limit that is reached first stops the session's activity. Service units are a weighted sum of CPU per session, connect time, logical reads per session, and private SGA. Default/Unlimited/Other (service units)
What is the limit on the amount of private space a session can allocate in the shared pool of the SGA?	This limit only applies if you are using Shared Server architecture. Default/Unlimited/Other (KB)
What is the limit on the total connection time per session?	The total elapsed time limit for a session Default/Unlimited/Other (Minutes)
What is the limit on idle time per session?	Permitted periods of continuous inactive time during a session, expressed in minutes. Long-running queries and other operations are not subject to this limit. Default/Unlimited/Other (Minutes)
What is the limit on concurrent sessions per user?	Default/Unlimited/Other
What is the limit on CPU time per session?	Default/Unlimited/Other (Hundredths of a Second)
What is the limit on data blocks read per session?	Default/Unlimited/Other
What is the limit on CPU time per call?	The limit on a parse, execute, or fetch. Default/Unlimited/Other (Hundredths of a Second)
What is the limit on the number of data blocks read for a call to process a SQL statement?	This is the number of logical reads per call. Default/Unlimited/Other
How many failed login attempts will be allowed before an account is locked?	Default/Unlimited/Other
How long will an account be locked after the specified number of failed login attempts?	Default/Unlimited/Other (Days)
What is the lifetime of the password?	Default/Unlimited/Other (Days)
How many days must pass before a password can be reused?	Default/Unlimited/Other (Days)

Required Information	Description
How many password changes are required before the current password can be reused?	Default/Unlimited/Other
What is the grace period allowed for a password to be changed without expiring?	Default/Unlimited/Other (Days)
What is the name of the password complexity verification routine?	Default Null - Specifies no password verification is performed. Function Name

ROLE WIZARD (ORACLE)

As you complete the Role Wizard, Rapid SQL constructs the necessary CREATE ROLE statement from the information that you have supplied. The Role Wizard lets you specify a name for the role and whether or not the role should be identified.

> **NOTE:** To create a role, you need the CREATE ROLE system privilege.

To Open the Role Wizard

1 On the Explorer, find the datasource where you want to create a role and expand the Security node.

2 Right-click the Roles node, and select New.

The table that follows describes the fields you may encounter as you complete the wizard.

Required Information	Description
What is the name of the role?	Self-explanatory.
How should the role be identified?	Not identified - Selecting this means the role you are creating will be enabled immediately. Identified: If this is your choice, you're indicating one of the following authorization methods will be followed: Globally - Select to indicate that Oracle permits access to the user by obtaining user name and password information from the security domain central authority. Externally - Select to indicate that Oracle should verify the database user name against an existing operating system user name. Password - Select to indicate that Oracle should identify the role with the password you provide. In the Password box, type the password for the user.

For information on activating and deactivating roles for the current login in the current session, see Activating/Deactivating Roles in the Current Session.

ROLLBACK SEGMENT WIZARD (ORACLE)

Rollback segments manage all transactions in your Oracle databases. A transaction is a read, modify, and write cycle for an Oracle database. A rollback entry is made for all transactions unless a particular clause is specified. So, a rollback segment is a transaction log that consists of a pre-update image value and transaction status, among other things. The rollback segments maintain read consistency among concurrent users in a database and if the transaction fails for any reason, the old image is taken from the rollback segment. By maintaining a history of data changes, rollback segments can rollback uncommitted transactions so that data is rolled back to the prior state. SYS owns all rollback segments no matter who created them and are not accessible to users, just Oracle.

Oracle, it should be mentioned, strongly recommends that you use automatic undo management to simplify managing databases. Tuning rollback segments is a manual process that has largely been deprecated by Oracle, but it is supported for backward compatibility reasons.

The Rollback Segment Wizard lets you:

- Name the rollback segment and to place it online or off-line.

- Place the rollback segment on a tablespace.

- Specify the initial next and optimal extent size as well a the minimum and maximum number of extents that should be allocated to the rollback segment.

   NOTE: This wizard is not available if auto-UNDO management is enabled.

   TIP: Make sure enough rollback segments exist on a database to handle the imposed workload. One rule of thumb is to create one rollback segment for every four concurrent users.

To Open the Rollback Segment Wizard

1   On the Explorer, find the datasource where you want to create a rollback segment and expand the Storage node.

2   Right-click the Rollback Segments node, and select New.

The table that follows describes the fields you may encounter as you complete the wizard.

Required Information	Description
What is the name of the rollback segment?	Self-explanatory.
Should this rollback segment be made public?	Yes - A public rollback segment can be brought online by any instance in a parallel server. Public rollback segments form a pool of rollback segments that can be used by any instance that needs one.
	No - This is the default. A private rollback segment can only be acquired by the instance specifying the segment in its initialization file.

Required Information	Description
Do you want to place the rollback segment to be online following its creation?	Online - To be useful, a rollback segment must be online. Offline - You may want to take rollback segments offline if you want to take a tablespace offline and it contains rollback segments that you want to keep from being used.
On which tablespace do you want to place this rollback segment?	Self-explanatory. Oracle suggests that you create one or more tablespaces specifically to hold all rollback segments. This way, the data contained in the rollback segments is held apart from other data types.
What extent sizes do you want to assign to this rollback segment?	Initial size (KB) Next size (KB) Optimal size (KB) Null/Default
What are the minimum and maximum number of extents to allocate to the rollback segment?	Minimum Maximum

SEQUENCE WIZARD (ORACLE)

Sequences are programmable database objects that provide numbers in sequence for input to a table. A sequence can be used to automatically generate primary key values for tables.Once defined, a sequence can be made available to many users. When you create a sequence, you can define its initial value, increment interval, and maximum value.

The Sequence Wizard lets you:

- Specify the name and owner of the sequence.

- Set both the value of the sequence, and an interval and ranges for incrementing it.

- Cache the sequence, cycle the sequence when it reaches its minimum or maximum values, and guarantee that Oracle generates sequence numbers in the order of request.

  NOTE: To create a sequence, it must belong to your schema or you need CREATE SEQUENCE privilege.

To Open the Sequence Wizard

1 On the Explorer, find the datasource where you want to create a rollback segment and expand the Schema node.

2 Right-click the Sequences node, and select New.

The table that follows describes the fields you may encounter as you complete the wizard.

Required Information	Description
Who owns the sequence?	Self-explanatory.
What is the sequence name?	Self-explanatory.

Required Information	Description
What is the first sequence number to be generated?	Start with: Pick an integer.
What is the interval between sequence numbers?	Increment by: Positive numbers will generate ascending values, and a negative number will generate descending numbers
What is the sequence's minimum value?	The default is 1 for ascending sequences; This integer value can have 28 or fewer digits. None Minimum value - Identify how low the sequence can go.
What is the sequence's maximum value?	For descending values, the default is 1. Lets you specify the maximum value the sequence can generate. This integer value can have 28 or fewer digits. None Maximum value - Indicate the highest sequence value that will be allowed.
Should Oracle preallocate sequence numbers and cache them for faster access?	This is the number of sequence values you want to specify in the SGA buffers. This speeds access, but cached numbers are erased when the database is shut down. The default value is 20. Yes Number of values No
Should the sequence continue to generate values after reaching either its maximum or minimum value?	If you say no, the sequences will automatically recycle to the minimum value when you've hit the maximum for ascending sequences and vice versa for descending sequence values. Yes No
Should the sequence numbers be generated in the order of request?	This may be required when sequences are required for timestamping. But generally, because sequences are naturally ordered, this is only necessary for if you use Oracle RAC clusters for parallel mode. Yes No

SNAPSHOT AND SNAPSHOT LOG WIZARDS (ORACLE)

Oracle has replaced the snapshot functionality with materialized views. Refer to the Materialized View Wizard (Oracle) and Materialized View Log Wizard (Oracle).

Synonym Wizard (Oracle)

Synonyms are alternate names for database objects to be used as a reference by users or applications. A synonym offers you security and convenience so that you can refer to an object without revealing who owns it or what database it belongs to. Synonyms Depending on the platform, you can define synonyms on tables, views, sequences, procedures, functions, packages, and materialized views. If an underlying object needs to be renamed or moved, it's easy enough to redefine the synonym without having to modify any applications based on the synonym.

> **NOTE:** To create a private synonym, you need CREATE SYNONYM privileges. To create a public synonym, you need CREATE PUBLIC SYNONYM privileges.

To create a new synonym using a wizard:

1 [Open an Object Wizard](#) for a synonym.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - [Synonyms (Oracle) - Properties](#).

- **Definition** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

Synonyms (Oracle) - Properties

When creating or editing a synonym, this tab/panel lets you work with the following settings:

Setting	Description
Owner and **Name**	Choose the owner and provide a name for the synonym being created.
Referenced Object Owner	Select the owner of the object to which the synonym is to refer.
Referenced Object Type	Select the type of the object to which the synonym is to refer.
Referenced Object Name	Select the specific object to which the synonym is to refer.
Database Link	If the object resides on a remote database, select a database link.

Table Wizard (Oracle)

Tables are the most basic data storage units for Oracle. As you might expect, data is stored in rows and columns. By completing the Table Wizard, Rapid SQL constructs the necessary CREATE TABLE statement from the information that you supply. The Table Wizard varies slightly in content based on the version of Oracle on the target datasource. But in all cases, you name columns and determine column width or precision and scale depending on the column's data type. A row collects the column information that corresponds to a single record.

You can set rules for the columns to live by, and these are called integrity constraints. For example, if you select NOT NULL, that column will have to have a value in each row.

Also, before beginning, consider what kind of table you want to create as the wizard will ask you to choose:

Heap organized table	This is a basic table where data is stored as an unordered collection, i.e., heap.
Index-organized table	A B-tree index structure stores data, sorted by primary key. Nonkey column values are stored too.
Partitioned table	Data is broken down into smaller, more manageable pieces called partitions or subpartitions. Each partition can be managed individually and operate independent of the other partitions.
Clustered table	This is a table that, once created, is part of a cluster. A cluster is a group of tables that share some data blocks and columns and are often used together. To create a cluster, use the Cluster Wizard (Oracle).

> **NOTE:** The table wizard panels differ depending on what options you select.

> **NOTE:** To simplify the process of creating a table, the Table Wizard focuses on creating the basic table definition with a primary key constraint. After you create the basic table definition you can add unique and foreign keys to the table on the Constraints Tab of the Tables Editor.

To create a new table using a wizard:

1 Open an Object Wizard for a table.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Tables (Oracle) - Properties.

- **Columns** panel - for details, see Tables (Oracle) - Columns.

- **Indexes** panel (not available with a **Row Organization** of **EXTERNAL**) - for details, see Tables (Oracle) - Columns.

- **Constraints** panel (not available with a **Row Organization** of EXTERNAL) - for details, see Tables (Oracle) - Constraints.

- **Storage** panel (not available with a **Row Organization** of EXTERNAL) - for details, see Tables (Oracle) - Storage.

- **IOT Properties** panel (not available with a **Row Organization** of EXTERNAL) - for details, see Tables (Oracle) - IOT Properties.

- **Partition** (not available with a **Row Organization** of **EXTERNAL**) - for details, see Tables (Oracle) - Partition.

- **Comment** panel - for details, see Add a Comment to an Object.

- **Permissions** panel - for details, see Set Permissions for an Object.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TABLES (ORACLE) - PROPERTIES
When creating or editing a table, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the table.
Name	Provide a name for the table
Cache	Enabling this feature keeps a block in memory by placing it at the most recently used end. This option is useful for small lookup tables.
Row Movement	Enabling this option permits the migration of a row to a new partition if its key is updated.
Parallel Degree	A value indicating the number of query server processes that should be used in the operation.
Parallel Instances	A value indicating how you want the parallel query partitioned between the Parallel Servers.
Physical group	Choose a Row Organization of INDEX, HEAP, or **EXTERNAL**. If you chose INDEX or HEAP, enable or disable **Logging**. If you chose EXTERNAL, provide an **External Type**, **Default Directory**, **Access Parameters**, **Reject Limit**, and **Location**.
Logging	Redo logs minimize the loss of data in the event that an uncontrolled shutdown happens.

TABLES (ORACLE) - COLUMNS
When creating or editing a table, this tab/panel lets you manage the columns for the table.

To add a column to the table

1   Click **Add Column**, provide a **Name** he column in the **Property/Value** list, and press TAB or ENTER.

    The column is added to the columns list on the left, with default attribute values.

2   Proceed to edit the column attributes.

To edit column attributes

3   Select the column in the columns list on the left. The controls in the Property/Values list are updated with values of the selected column.

4   Use the following table as a guide to providing additional property values, noting that availability of a property differs by data type and other property selections.

Property or group	Description
Virtual	Selecting this check box defines the column as an Oracle virtual column, with a value calculated from a column expression. Virtual columns do not use any disk space as there is o data to store, and INSERT/UPDATE operations are not supported. Consult Oracle documentation before working with virtual columns to familiarize yourself with topics such as column expressions and virtual column-specific details such as automatic type conversion. For more information, see [Accessing Third Party Documentation](#). After specifying a column as virtual, use the **Default Value** box to provide the calculation.
Datatype	This group lets you select a valid **Type** for the column. Depending on your selection, additional properties such as **Scale**, **Size**, **Width**, and **Unused** may be available.
Allow Nulls	Select this check box to allow nulls in this column.
Encryption	Use controls in this group to have the column encrypted using transparent data encryption. Select the **Encrypted** check box to enable transparent data encryption for the column. Type a **Password** value used to build an IDENTIFIED BY clause. The column key will be derived from the value you provide. Select the **Salted** check box to add a SALT option, appending a random 'salt' string, to the clear text of the column before encrypting. Leaving the check box unselected adds a NO SALT option. From the **Encryption Algorithm** dropdown, select an algorithm (AES256, NONE, DES168, AES192, AES128, or DES156) that will be used to build a USING clause.
Default Value	If you selected the **Virtual** check box, type a valid Oracle column expression that will calculate the value of the column. Otherwise, either select a pseudocolumn (CURRENT_TIMESTAMP, USER, SYSDATE, or UID) from the dropdown or type an expression resulting in a value that matches the selected **Datatype**.
Comment	Lets you add a comment to the column.
LOB Storage	This group is available for bfile, blob, clob, and nclob types. Settings include **Segment Name**, **Configuration** properties (**Tablespace**, **Chunk**, **Percent Version**, **Enable Storage In Row**, **Cache**, and **Logging**), and **Storage** properties (**Initial Extent**, **Next Extent**, **Percent Increase**, **Minimum Extents**, **Maximum Extents**, **Free Lists**, and **Free List Groups**).

To delete a column

1   Select a column from the column list on the left.

2   Click **Delete** to remove the column from the table.

To change a column's ordering position

1   Select a column from the column list on the left.

2   Use the arrow buttons to move the column up or down.

TABLES (ORACLE) - INDEXES
> **NOTE:** This tab/panel is not available with a **Row Organization** of **EXTERNAL**.

Click **Add** to open the [Index Wizard (Oracle)](#).

TABLES (ORACLE) - CONSTRAINTS
> **NOTE:** This tab/panel is not available with a **Row Organization** of **EXTERNAL**.

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- [Primary Key Wizard (DB2 LUW)](#)

- [Unique Key Wizard (DB2 LUW)](#)

- [Foreign Key Wizard (IBM DB2 LUW)](#)

TABLES (ORACLE) - STORAGE
> **NOTE:** This tab/panel is not available with a **Row Organization** of **EXTERNAL**.

When creating or editing a table, this tab/panel has the following settings:

Settings	Description
Data Block Storage group	Select the DEFAULT **Tablespace** only if you want the partitions in the same tablespace as the partitions in the underlying table.
	Percent Free identifies how much space you want to allocate for new rows or updates to existing rows.
	Initial Transactions ensures that a minimum number of concurrent transactions can update a primary key block, avoiding the overhead of allocating a transaction entry dynamically.
	Maximum Transactions limits concurrency on a primary key block.
Extents group	An extent is the unit of space allocated to an object whenever the object needs more space.
	Initial Extent - The initial space extent (in bytes) allocated to the object.
	Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
	Percentage Increase - Lets you type the percentage.
	NOTE: You should be careful when setting Percent Increase because it magnifies how an object grows and, therefore, can materially affect available free space in a tablespace.
	Minimum Extents - For a dictionary managed tablespace, this is the total number of extents to be allocated when the index is first created. For a locally managed tablespace, this is simply the initial amount of space allocated.
	Maximum Extents - For a dictionary managed tablespace, this is the total number of extents that can ever be allocated to the index. In a locally managed tablespace, the database will automatically manage the extents.

Settings	Description
Freelists group	**Free lists** let you manage the allocation of data blocks when concurrent processes are issued against the primary key. You can potentially improve the performance of the primary key by identifying multiple free lists, which can reduce contention for free lists when concurrent inserts take place.
	The default and minimum value is 1. You should increase this number if multiple processes access the same data block.
	Free List Groups is the number of groups of free lists.
	NOTE: This option is only applicable for the parallel server option.
Buffer Pool	DEFAULT - Choose this if you want to use the default bufferpool.
	KEEP - Use this to retain the object in memory to avoid I/O conflicts. This type of bufferpool stores frequently referenced data blocks in a separate cache.
	RECYCLE - Select this option to save cache space by ridding data blocks from memory as soon as they are no longer in use.

TABLES (ORACLE) - IOT PROPERTIES
> **NOTE:** This tab/panel is not available with a **Row Organization** of **EXTERNAL**.

Provide compression and space details for an index-organized table.

TABLES (ORACLE) - PARTITION
> **NOTE:** This tab/panel is not available with a **Row Organization** of **EXTERNAL**.

Prior to working with partitions, you should be familiar with the material in [Rapid SQL and Oracle Partitioning](#).

Click **Create Partition** to [Partition a table](#).

RAPID SQL AND ORACLE PARTITIONING

Partitioning your tables lets you get around the problem of supporting large tables. Partitioning lets you break large tables into smaller pieces, which are called partitions. Partitions make the data in your table easier to manage and analyze. Your SQL statements can access the partitions rather than the entire table. Partitions are most useful in data warehouse applications, which store large amounts of data.

The table below describes the types of partitions in Oracle:

Partition Type	Description
Range	Use range partitioning to map rows to partitions based on ranges of column values. This type of partitioning is useful when dealing with data that has logical ranges into which it can be distributed; for example, months of the year. Performance is best when the data evenly distributes across the range. If partitioning by range causes partitions to vary dramatically in size because of unequal distribution, you may want to consider one of the other methods of partitioning.

Partition Type	Description
Hash	Use hash partitioning if your data does not easily lend itself to range partitioning, but you would like to partition for performance and manageability reasons. Hash partitioning provides a method of evenly distributing data across a specified number of partitions. Rows are mapped into partitions based on a hash value of the partitioning key. Creating and using hash partitions gives you a highly tunable method of data placement, because you can influence availability and performance by spreading these evenly sized partitions across I/O devices (striping).
Composite	In Oracle 8i, Oracle introduced both hash and composite partitioning. Hash partitions partition the table according to a hash function. Composite partitions use both range and hash types, first partitioning the data by a range of values, and then further dividing the partitions into subpartitions by way of a hash function. This option is not available for index-organized tables.
List	Use list partitioning when you require explicit control over how rows map to partitions. You can specify a list of discrete values for the partitioning column in the description for each partition. This is different from range partitioning, where a range of values is associated with a partition, and from hash partitioning, where the user has no control of the row to partition mapping.

PARTITION A TABLE

The Add partition wizard lets you set up partitions for a table. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

Step	Settings and tasks
Properties	Select a **Partition Type** and optionally, a Subpartition Type.
Columns	For each column, click the New button and select a name from the Column dropdown. Use the Delete button to drop a selected column.
Subpartition Columns (only available with a **Partition Type** of **RANGE** and **Subpartition Type** of **HASH** or **LIST**)	For each column, click the New button and select a name from the Column dropdown. Use the Delete button to drop a selected column.
Subpartitions (only available with a **Subpartition Type** of **HASH**)	Specify a **Default number of partitions**. For each partition, click the New button and then select a tablespace from the dropdown.
Range Definitions (only available with a **Partition Type** of **RANGE**)	Click the New button to [Add a partition definition].
Partition Definition (only available with a **Partition Type** of **HASH**)	To specify a partition method other than **None**, take one of the following actions: (1) Select the **Number Of Partitions** radio box, specify the **Number Of Partitions**, and for each partition, click the New button and choose a tablespace from the dropdown, or (2) select the **By Partition Name** radio box and for each partition, click the New button provide a name and then choose a tablespace from the dropdown.

Step	Settings and tasks
List Definitions (only available with a **Partition Type** of **List**)	Click New to <u>Add a partition definition</u>.

ADD A PARTITION DEFINITION

Use the following topics as a guide in completing the settings in this wizard:

Step	Settings and tasks	
Partition Definition	**Name**	Provide a name.
	Tablespace	Select a tablespace from the dropdown
	Logging	Enable or disable logging.
Subpartitions	To specify a partition method other than **None**, select the **By Subpartition Name** radio box and for each partition, click the New button to open a dialog that lets you provide subpartition values.	
	NOTE: When you split a range-list partition, you cannot specify the new partitions' subpartition information.	
Storage	Provide or select Data Block Storage, Extents, Freelists, and Buffer Pool values.	

TABLESPACE WIZARD (ORACLE)

Tablespaces are logical storage structures that act as partitions for the database. Each tablespace consists of one or more datafiles which are the physical structures holding the data. You can create a tablespace to store table data and other objects related to table performance such as indexes or large object data. Tablespaces are used to manage large complex databases. Once you have created a tablespace, you can place objects on it.

The Tablespace Wizard lets you:

- Name the tablespace, and specify space management.

- Specify what types of objects are stored on the tablespace, and place the tablespace online or offline.

- Add the datafiles that comprise the tablespace and specify the parameters for the datafiles.

- Specify how Oracle should manage the growth of the tablespace.

Important Notes

- For auto-UNDO management to be in effect, set init.ora parameter to undo_management. When set to MANUAL (the default), it disables auto-UNDO management. When to set AUTO, auto-UNDO management is enabled.

- To determine if the undo_management parameter is set to AUTO, use the following query:

  SELECT VALUE

  FROM   SYS.V_$PARAMETER

  WHERE NAME = 'undo_management'

  NOTE: This parameter cannot be set dynamically via the ALTER SYSTEM or ALTER SESSION.

For users using a version earlier than Oracle 8i and locally managed tablespaces, there are manual methods you can use to assist in the fight against tablespace fragmentation. They include:

- Setting PCTINCREASE to zero for all tablespaces and objects to promote same-sized extents.

- Specifying equal-sized allotments for your INITIAL and NEXT object storage parameters.

- Grouping objects with like growth and storage needs together in their own tablespaces.

One of the best ways to avoid fragmentation in a tablespace is to pre-allocate the space that your objects will use. If possible, plan for one to two years' growth for each object and allocate your space accordingly. Having initial empty objects will not affect table scan times as Oracle only scans up to the high-water mark (the last used block) in a table.

Of all your tablespaces, you want to avoid fragmentation problems in your SYSTEM tablespace the most as this is the major hotbed tablespace for Oracle activities. The easiest way to avoid this is to not allow any user (even the default DBA ID's SYS and SYSTEM) to have access to it. There are three ways to do this:

- Ensure no user has a DEFAULT or TEMPORARY tablespace assignment of SYSTEM.

- Ensure no user has a quota set for SYSTEM.

- Ensure no user has been granted the UNLIMITED TABLESPACE privilege.

To create a new tablespace using a wizard:

1 [Open an Object Wizard](#) for a tablespace.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - see [Tablespaces (Oracle) - Properties](#).

- **Datafiles** panel - see [Tablespaces (Oracle) - Datafiles](#).

- **DDL** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TABLESPACES (ORACLE) - PROPERTIES

When creating or editing a tablespace, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the tablespace.
Big File	Selecting this check box results in a CREATE BIGFILE TABLESPACE statement being generated. This results in a large, single file tablespace.
	Leaving this check box unselected results in a tablespace containing multiple smaller datafiles.
Type	Leaving the default PERMANENT selection creates a permanent tablespace, intended to contain schema objects you want to keep on an ongoing basis. The objects are stored in datafiles.
	The other two options let you generate a CREATE TABLESPACE statement that specifies TEMPORARY or UNDO keywords.
	An Undo tablespace is a kind of permanent tablespace that holds undo data if your database is operating in automatic undo mode. Oracle recommends the automatic undo mode as the wiser choice than using rollback segments to undo.
	In a Temporary tablespace, shema objects will last only as long as your session continues.
Tablespace Group	This option is only available with a **Type** value of TEMPORARY.
	If there are currently entries in the SYS.DBA_TABLESPACE_GROUPS table, you can either select an existing group or type the name for a new tablespace group. If there are currently no entries in the SYS.DBA_TABLESPACE_GROUPS table, you can type the name of a new tablespace group to be created.
	When editing a tablespace, clearing the entry from the **Tablespace Group** box deletes the <tablespace_name, group_name> entry from the SYS.DBA_TABLESPACE_GROUPS table.
Retention Guarantee	This option is only available with a **Type** value of UNDO.
	Selecting this check box specifies that unexpired data should be preserved for all Undo segments of the tablespace, regardless of whether this may force failure of ongoing operations that require space in those segments.
	Leaving this check box unselected specifies default RETENTION NOGUARANTE behavior. Space currently being used by unexpired undo data in undo segments can be consumed as necessary by ongoing transactions.
Status	This option is only available with a **Type** value of PERMANENT.
	Use this control to specify an ONLINE/OFFLINE clause in the generated DDL. The ONLINE selection makes the tablespace available immediately after creation. The OFFLINE selection results in a table is unavailable immediately after creation.
Logging	This option is only available with a **Type** value of PERMANENT.
	Use this control to specify a LOGGING/NOLOGGING clause in the generated DDL. This sets the default logging attributes of all tables, indexes, materialized views, materialized view logs, and partitions within this tablespace.
Force Logging	This option is only available with a **Type** value of PERMANENT.
	Selecting this check box specifies that a FORCE LOGGING clause be included in the generated DDL.
	Changes to all objects in the tablespace (temporary segments excepted) are logged, overriding any NOLOGGING setting for individual objects.

Setting	Description
Compression Type	This option is only available with a **Type** value of PERMANENT.
	This lets you select default compression type of ALL OPERATIONS (corresponds to COMPRESS FOR ALL OPERATIONS in the DDL), COMPRESS_DIRECT_LOAD (corresponds to COMPRESS FOR DIRECT_LOAD OPERATIONS in the DDL) or no compression.
Flashback	This option is only available with a **Type** value of PERMANENT.
	Use this control to specify a FLASHBACK ON/FLASHBACK OFF clause in the generated DDL. Select this check box to put the tablespace in FLASHBACK mode. In FLASHBACK mode, Oracle saves Flashback log data for the tablespace and the tablespace can participate in a FLASHBACK DATABASE operation.
	If you leave this check box unselected, Oracle does not save Flashback log data for the tablespace. Prior to FLASHBACK DATABASE operations, you must either take datafiles in the tablespace off line or drop them OR take the entire tablespace offline. The database does not drop existing Flashback logs.
Encrypted and **Algorithm**	This option is only available with a **Type** value of PERMANENT.
	Selecting the **Encryption** check box enables the **Algorithm** control. Together, they let you add an ENCRYPTION USING clause to the generated DDL. The **Algorithm** lets you select from AES 128, AES192, AES256, DES56, DES168, or NONE encryption algorithms.
Locally Managed, **Uniform Allocation**, **Size**, **Automatic Segment Space Management**, and **Minimum Extent Size**	These option are only available with a **Type** value of PERMANENT or UNDO.
	These controls let you work with the extent-related clauses that will be included in the generated DDL.
	Selecting the **Locally Managed** check box lets you generate an EXTENT MANAGEMENT LOCAL clause. Subsequently leaving the **Uniform Allocation** check box deselected adds an AUTOALLOCATE option. Alternatively, selecting the **Uniform Allocation** check box enables the **Size** check box, letting you add a UNIFORM SIZE option. Select the **Automatic Segment Space Management** check box to generate a SEGMENT SPACE MANAGEMENT AUTO or leave it deselected to generate a SEGMENT SPACE MANAGEMENT MANUAL clause.
	Leaving the **Locally Managed** check box deselected generates an EXTENT MANAGEMENT DICTIONARY clause. This also enables the **Minimum Extent Size** control, letting you select a MINIMUM EXTENT clause value.
Use Default Block Size and **Block Size**	Disabling the **Use Default Block Size** check box enables the **Block Size** control, generating a BLOCKSIZE clause to specify a nonstandard block size for the tablespace.

TABLESPACES (ORACLE) - DATAFILES

When creating or editing a tablespace, this tab/panel lets you build DATAFILE or TEMPFILE clauses that will be submitted with the CREATE TABLESPACE statement creating or editing this tablespace. This lets you manage the files that make up the tablespace.

To add a datafile

1   Click **Add**, either select an existing file name from **File Name** dropdown or type the name of a new file, and then press TAB or ENTER.

    The file is added to the datafiles list on the left, with default attribute values.

2   Proceed to edit the datafile attributes.

To edit datafile attributes

3   Select the file from the list on the left. The controls in the Property/Values list are updated with values of the selected file.

4   Use the following table as a guide to modifying property values.

Properties	Description
Size	Lets you specify the SIZE option for the DATAFILE/TEMPFILE clause, to specify the size of the file
Reuse Existing File	Lets you specify the REUSE option for the DATAFILE/TEMPFILE clause, to specify that an existing file can be reused.
Auto Extend and **Growth Amount**	Use these options to select an AUTOEXTEND option for the DATAFILE/TEMPFILE clause. Leaving the **Auto Extend** check box unselected disables the **Growth Amount** disabled and specifies an AUTOEXTEND OFF option in the generated DDL. Selecting the **Auto Extend** check box lets you specify an AUTOEXTEND ON NEXT clause. Selecting **Auto Extend** enables the **Growth Amount** control, letting you specify the size of the next increment of disk space automatically allocated when more extents are required.
Unlimited and **Value**	These options are only available with **Auto Extend** selected. They let you specify a MAXSIZE clause. Selecting the **Unlimited** check box lets you specify a MAXSIZE UNLIMITED clause, placing no limit on the size of the file. Leaving the **Unlimited** check box unselected enables the **Value** disabled. This specifies a MAXSIZE clause with a size provide by the **Value** control.

To delete a datafile

1   Select a datafile from the column list on the left.

2   Click **Delete** to remove the datafile from the tablespace.

TRIGGER WIZARD (ORACLE)

A trigger is a special type of procedure that automatically fires when defined data modification operations (insert, update, or delete) occur on a designated table or view. Triggers fire after an insert, update or delete, but belong to the same transaction as the data modification operation. A stored procedure, on the other hand, is activated explicitly by the user or application (or trigger). Triggers should only be used for centralized, global operations that should be initiated by the triggering statement and should not be based on whether a particular user or database application issues the statement.

Important Notes

• To create triggers in your own schema, you need CREATE TRIGGER privileges. To create triggers in other schemas, you need CREATE ANY TRIGGER privileges.

To create a new trigger using a wizard:

1 [Open an Object Wizard](#) for a trigger.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   - **Properties** panel - for details, see [Triggers (Oracle) - Properties](#).

   - **Column Selection** panel - for details, see [Triggers (Oracle) - Column Selection](#).

   - **Action** panel - for details, see [Triggers (Oracle) - Action](#).

   - **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

TRIGGERS (ORACLE) - PROPERTIES

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type, **Parent Owner**, and **Parent Name**	Select the type (TABLE or VIEW), owner and specific object for which the trigger is being created.
Name	Provide a name for the trigger being created.
Enabled	Enable or disable the trigger.
Trigger Timing	BEFORE: These triggers serve as extensions to the constraint subsystem and are most often used to validate input data, generate values for newly inserted rows, and read from other tables for cross-reference purposes. Note: Before triggers must be created as a For Each Row. AFTER: Such a trigger is run after the integrity constraint validations; they can be used to modify operations in the database or be used for activities beyond the database, like supporting an alert notification.
Fire On Insert, **Fire On Update**, and **Fire On Delete**	Enable the events that fire the trigger. An INSERT trigger must be associated with an INSERT statement. For example, if a data load operation doesn't include an INSERT statement, the trigger won't be invoked. An UPDATE trigger can be associated with specific columns of the base table and will only be activated if those columns are updated. A **DELETE** trigger fires automatically after items in the table are deleted.
Trigger Type	**STATEMENT**: (only fires once). **ROW** (fires for each affected row): The trigger runs as many times as there are rows in the affected section. If the set of affected rows is empty, the trigger doesn't run.
Old Table Alias	Type the name of a temporary table of rows as they exist before they're updated or deleted.
New Table Alias	Type a name for a temporary table of rows as they exist after they're inserted or updated.
When Clause	Type a WHEN clause or open a WHEN CLAUSE wizard to further qualify the trigger behavior.

TRIGGERS (ORACLE) - COLUMN SELECTION

If you chose UPDATE as the **Trigger Event**, select the columns, select the check box beside each column that is to fire the trigger.

TRIGGERS (ORACLE) - ACTION

Provide the body of trigger. If the SQL statement you have in mind for a trigger is longer than 60 lines of code, you would be better off creating a stored procedure.

INSTEAD OF statements can only be used by view triggers. BEFORE and AFTER options cannot be used for view triggers.

OBJECT TYPE WIZARD (ORACLE)

Types define an abstract data type or object composed of a collection of similar types of data. For example, you can create an object type that defines a full address rather than the pieces of an address, such as city, state and postal code. An object type stores the pieces of an address in a single type, storing them in the same location, and allowing the full address to be accessed and manipulated as single unit rather than multiple units.

To Open the Object Type Wizard

 1 On the Explorer, find the datasource where you want to create a Type and expand the Schema node.

 2 Right-click the Type node, and select New.

The single page wizard asks that you pick an owner for the type and name the type. The fun begins after you make your choices and click Finish. The Object Editor, where you can finalize the object type's creation, opens.

The Object Type Editor's tabs are:

Header: Here you flesh out the specification for the type you're creating including any methods (a subprogram), attributes, and any parameters or return types for a function.

Body: For every specification, the object type body defines the code for the method. If the object type header declares only attributes, not a method, the body is unnecessary.

You need to create the object type before you can see the following tabs. After you've added what you need to the Header and Body pages of the Editor, click Create on the toolbar.

Information: Read-only page that indicates the vital statistics for the object type.

Dependencies: A tree structure displays the objects that depend on the type you have created.

Privileges: Displays individual, group, and role permissions associated with the object type.

UNIQUE KEY WIZARD (ORACLE)

A unique key constraint requires that every value in a column or set of columns be unique. Thus, no two rows of a table have duplicate values in the column or set of columns you identified. So, for example, you can use a unique key constraint to make sure you haven't duplicated a social security number in a list of employees.

To create a new unique key using a wizard:

1 Open an Object Wizard for a unique key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Unique Keys (Oracle) - Properties.

   • **Columns** panel - for details, see Unique Keys (Oracle) - Columns.

   • **Storage** panel - for details, see Unique Keys (Oracle) - Storage.

   • **Partition** panel - for details, see Unique Keys (Oracle) - Partition.

   • **DDL** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

UNIQUE KEYS (ORACLE) - PROPERTIES

When creating or editing a unique key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner and **Table Name**	Choose the owner and name of the table in which the unique key is being created.
Name	Provide a name for the unique key being created.
No Sort	Enable this feature if the rows in the table already stored in ascending order. This increases the speed of the index creation process. Oracle does not sort the rows.
Logging	Enabling logs this operation to the redo file.
Reverse	Enabling this feature stores the bytes of the index block in reverse order and excludes the ROWID. The ROWID is a globally unique identifier for a row in a database. It is created at the time the row is inserted into a table, and destroyed when it is removed from a table.
Validate	Enabling this option indicates that existing data is checked against the constraint when the primary key is enabled. Leaving it disabled indicates that only new data is to be checked against the constraint.
Deferrable	Dictates whether constraint checking can be deferred until the end of a transaction.
Deferred	This option is enabled only if you enabled the **Deferrable** option. Select IMMEDIATE to have the constraint checked at the end of every DDL statement. Select DEFERRED to have the constraint checked only at the end of a transaction.
Enabled	Enables or disables the primary key.

UNIQUE KEYS (ORACLE) - COLUMNS

From the **Column** dropdown, select a column for the index and specify a **Sort** option. To add more columns, click the New button and then follow the steps in the last instruction. Use the Delete button to drop columns.

UNIQUE KEYS (ORACLE) - STORAGE

When creating or editing a primary key, this tab/panel lets you work with the following settings:

Setting	Description
Data Block Storage group	Select the DEFAULT **Tablespace** only if you are creating a local partitioned index and want the partitions in the same tablespace as the partitions in the underlying table. (Each partition of a local index is associated with one partition of the table. Oracle can then keep the index partitions in synch with table partitions.) A transaction entry is needed for each INSERT, UPDATE, DELETE, etc. statement that accesses one or more rows in the block. Transaction entries in many operating systems require approximately 23 bytes. **Percent Free** identifies how much space you want to allocate for new rows or updates to existing rows. **Initial Transactions** ensures that a minimum number of concurrent transactions can update an index block, avoiding the overhead of allocating a transaction entry dynamically. **Maximum Transactions** limits concurrency on an index block.
Extents group	An extent is the unit of space allocated to an object whenever the object needs more space. **Initial Extent** - The initial space extent (in bytes) allocated to the object. **Next Extent** - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required. **Percentage Increase** - Lets you type the percentage. NOTE: You should be careful when setting Percent Increase because it magnifies how an object grows and, therefore, can materially affect available free space in a tablespace. **Minimum Extents** - For a dictionary managed tablespace, this is the total number of extents to be allocated when the index is first created. For a locally managed tablespace, this is simply the initial amount of space allocated. **Maximum Extents** - For a dictionary managed tablespace, this is the total number of extents that can ever be allocated to the index. In a locally managed tablespace, the database will automatically manage the extents.
Freelists group	**Free lists** let you manage the allocation of data blocks when concurrent processes are issued against the index. You can potentially improve the performance of the index by identifying multiple free lists, which can reduce contention for free lists when concurrent inserts take place. The default and minimum value is 1. You should increase this number if multiple processes access the same data block. **Free List Groups** is the number of groups of free lists. NOTE: This option is only applicable for the parallel server option.
Buffer Pool	DEFAULT - Choose this if you want to use the default bufferpool. KEEP - Use this to retain the object in memory to avoid I/O conflicts. This type of bufferpool stores frequently referenced data blocks in a separate cache. RECYCLE - Select this option to save cache space by ridding data blocks from memory as soon as they are no longer in use.

UNIQUE KEYS (ORACLE) - PARTITION

Clicking Create Partition opens a wizard that lets you create a partition.

USER WIZARD (ORACLE)

Whoever you add as a user will have access to the Oracle database. You also can set up the means by which the database recognizes the user.

Important Notes

- To create a user, you need the CREATE USER system privilege.

To create a new user using a wizard:

1 Open an Object Wizard for a user.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Users (Oracle) - Properties.

- **Roles** panel - for details, see Users (Oracle) - Roles.

- **Quotas** panel - for details, see Users (Oracle) - Quotas.

- **DDL** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USERS (ORACLE) - PROPERTIES

When creating or editing a user, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the user.
Default Tablespace and **Temporary Tablespace**	Select the default tablespace for objects the user creates and the tablespace to be used for the user's temporary segments.
	If you do not specify a default tablespace when creating a user, the assigned tablespace will be that set using the Set Default function. For details, see Set Default.
Profile	The profile you choose affects the amount of database resources available to the user.
Identified By	REQUIRED_YES - A user identified by password is a local user.
	REQUIRED_EXTERNAL - This is a user who is validated by an external service like an operating system or third-party service. The login authentication is handled by that external entity.
	REQUIRED_GLOBAL - The user you're creating is a global user who is authenticated by the enterprise directory service.
Password	If you specified an **Identified By** value of REQUIRED_YES, provide a password for the user.

Setting	Description
External Name	If you specified an **Identified By** value of REQUIRED_EXTERNAL, provide an external name for the user.
Account Locked	When enabled, the account cannot be altered by anyone but the creator. It also means that after a specified number of failed attempts to access the database, the database will remain closed to the user for a period of time.
Password Expired	Marks the password as expired, forcing the user to change the password before being allowed to connect to the database.

USERS (ORACLE) - ROLES

For each role to be assigned to the user, select the check box beside that role.

USERS (ORACLE) - QUOTAS

To assign a tablespace quota for a user, select a tablespace, select the **Other** radio button, and provide the value in the **Quota** box.

VIEW WIZARD (ORACLE)

Views are SQL queries stored in the system catalog that customize the display of data contained in one or more tables. Views behave like tables because you can query views and perform data manipulation operations on them. However, views do not actually store any data. Instead, they depend on data contained in their base tables. You can use views to help enforce corporate security policies by creating a view that limits information a user can see.

Important Notes

- To create a view in your own schema, you need CREATE VIEW privileges. To create a view in someone else's schema, you need CREATE ANY VIEW privileges.

To create a new view using a wizard:

1 [Open an Object Wizard](#) for a view.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Views (Oracle) - Properties](#).

- **Definition** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

VIEWS (ORACLE) - PROPERTIES

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the view. The owner of the view must have SELECT privileges for the tables in the CREATE view statement or DBADM authority on the database that contains the table.
Name	Provide a name for the view.

VIEWS (ORACLE) - DEFINITION

This is where you build the query that will display the data you're interested in seeing. The template is CREATE VIEW <name> AS
SELECT: Identify the columns you want to show in the view
FROM: Identify the table(s) you want to draw the data from
WHERE: Write the equivalence you want to view.

You can use the Query Builder to help you write the appropriate SQL statement. For details, see Query Builder.

SYBASE ASE OBJECT WIZARDS

Rapid SQL lets you create Sybase objects using the following wizards:

- Alias Wizard (Sybase)
- Database Wizard (Sybase)
- Default Wizard (Sybase)
- Extended Procedure Wizard (Sybase)
- Foreign Key Wizard (Sybase)
- Functions Wizard (Sybase)
- Group Wizard (Sybase)
- Index Wizard (Sybase)
- Login Wizard (Sybase)
- Primary Key Wizard (Sybase)
- Procedure Wizard (Sybase)
- Rule Wizard (Sybase)
- Segment Wizard (Sybase)
- Table Wizard (Sybase)

- Trigger Wizard (Sybase)

- Unique Key Wizard (Sybase)

- User Datatype Wizard (Sybase)

- User Wizard (Sybase)

- View Wizard (Sybase)

ALIAS WIZARD (SYBASE)

The Alias Wizard lets you map a login to an existing user in the database. You can set up aliases so that multiple users log in to the same account and therefore have the same privileges. You can also set up an alias based on individual log ins and give those users access to the same alias with the advantage that you can track their activity within the database.

To create a new alias using a wizard:

1 Open an Object Wizard for an alias.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   - **Properties** panel - for details, see Aliases (Sybase) - Properties.

   - **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

ALIASES (SYBASE) - PROPERTIES

When creating or editing an alias, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a name for the alias being created.
User	Select the user to which the alias refers. The user has to have a valid account on SQL Server but cannot be a user in the current database.

DATABASE WIZARD (SYBASE)

The Database Wizard lets you create a database (a structured collection of data that can be updated or queried) without knowing the underlying commands. Databases can be simple, that is one file with many records and the same fields, or much more complicated with multiple files with different fields.

To create a new database using a wizard:

1 [Open an Object Wizard](#) for a database.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Databases (Sybase) - Properties](#).

- **Placement** panel - for details, see [Databases (Sybase) - Placement](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

DATABASES (SYBASE) - PROPERTIES

When creating or editing a database, this tab/panel lets you work with the following settings:

Setting	Description
Database Name	Provide a name for the database.
For Load	This option speeds loading by eliminating the step for pre-initializing panels. The for load option is appropriate when creating a database for recovering from media failure or moving a database from one machine to another.
With Override	Enable this option to specify whether logs and data are to be kept on the same logical device
Type	Optionally, use the TEMP option to designate this as a temporary database.
Options group	Enable or disable the following Sybase database options: **abort tran on log full**, **allow nulls by default**, **auto identity**, **dbo use only**, **ddl in tran**, **identity in nonunique index**, **no chkpt on recovery**, **no free space acctg**, **read only**, **select into/bulkcopy/pllsort**, **single user**, **trunc log on chkpt**, and **unique auto_identity index**.

DATABASES (SYBASE) - PLACEMENT

In the **Fragment Properties** area provide or select values for the default fragment: **Device Name**, **Size** for the fragment, and a **Device Type** value of data only, log only, or data and log.

If necessary use the New button to add a new fragment and repeat the steps above to provide details for that fragment. Use the Delete button to drop a fragment.

4

-

-

1

DEFAULT WIZARD (SYBASE)

Here you create a default for table column or user-defined datatype in the event that no value is available when the data is inserted. The default value you specify will be inserted only in the current database. You can then bind the default to a specific column or user-datatype.

To create a new default using a wizard:

1   Open an Object Wizard for a default.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Defaults (Sybase) - Properties.

- **Dependencies** panel - for details, see Defaults (Sybase) - Dependencies.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

Defaults (Sybase) - Properties

When creating or editing a default, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the schema that is to own the default.
Name	Provide a name for the default.
Value	Provide the value of the default.

DEFAULTS (SYBASE) - DEPENDENCIES

From the **Type** dropdown, choose Column or Datatype, and if you chose **Column**, choose a Table from the **Table** dropdown. The list on the left is populated with candidate columns or datatypes. To move a candidate from the list on the left to the dependencies column on the right, select the candidate and click **Add**. Remove columns or datatypes from the dependencies list on the right by selecting the column or datatype and clicking **Remove**.

EXTENDED PROCEDURE WIZARD (SYBASE)

Extended stored procedures provide a method for calling external procedural language functions from within the Adaptive Server. A procedural language is a language capable of calling a C language function or manipulating C language datatypes.

To create a new extended procedure using a wizard:

1   Open an Object Wizard for an procedure.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Extended Procedures (Sybase) - Properties.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

EXTENDED PROCEDURES (SYBASE) - PROPERTIES

When creating or editing an extended procedure, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the extended procedure.
Name	Provide the name of the procedure.
Library Name	Provide the name of the library containing the procedure.

FOREIGN KEY WIZARD (SYBASE)

Foreign keys are used to relate information from one table to another. Foreign keys are unique values that refer to specific columns of other tables. Thus, a foreign key links two tables together. Embarcadero Rapid SQL's Foreign Key Wizard makes it easy for you to create a relational link between two tables, thereby speeding queries and giving you faster access to data. The column in the initial table, the parent table, is called the primary key. The corresponding column in the (child) table that references the primary key, is the foreign key. Foreign keys can also refer to columns within the same table.

To create a new foreign key using a wizard:

1   <u>Open an Object Wizard</u> for a foreign key.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see <u>Foreign Keys (Sybase) - Properties</u>.

   • **Column Mapping** panel - for details, see <u>Foreign Keys (Sybase) - Column Mapping</u>.

   • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

FOREIGN KEYS (SYBASE) - PROPERTIES

When creating or editing a foreign key, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner	Select the owner of the referring, or child, table
Table Name	Select the name of the referring, or child, table.
Match Full	Specify a referential integrity option.

FOREIGN KEYS (SYBASE) - COLUMN MAPPING

Under **Referenced Table**, choose the **Owner** and then the **Name** of the referenced, or parent, table.

Under the **Main Table**, select check boxes corresponding to the columns that are to reference columns in the referenced table. Then, under **Referenced Table**, select the corresponding column check boxes.

FUNCTIONS WIZARD (SYBASE)

The Functions Wizard creates and submits a CREATE FUNCTION command. This lets you create a user-defined function, a saved Transact-SQL routine that returns a specified value. The wizard collects basic identifying information and generates basic CREATE FUNCTION statement syntax, letting you provide the body of the function.

   NOTE: This functionality is supported for Sybase ASE 12.5.

To create a new function using a wizard:

1 [Open an Object Wizard](#) for a function.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties panel** - for details, see [Functions (SQL Server) - Properties](#).

- **Definition panel** - for details, see [Functions (SQL Server) - Definition](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

Functions (Sybase) - Properties
When creating or editing a function, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the function.
Name	Provide a name for the function.

Functions (Sybase) - Definition
Complete the CREATE FUNCTION outline provided by typing or pasting the body of the function. Finally, use the Schedule or Execute buttons to [Create the New Object](#).

GROUP WIZARD (SYBASE)
A group is a collection of privileges that the DBA assigns to designated users.

To create a new group using a wizard:

1 [Open an Object Wizard](#) for a group.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Groups (Sybase) - Properties](#).

- **Object Permissions** and **System Permissions** panels - see [Set Permissions for an Object](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

After you click Finish and Execute, use the Group Editor to assign users to the new group. For details, see [Groups Editor (Sybase)](#).

When creating a group, this panel lets you provide a name for the group.

INDEX WIZARD (SYBASE)

Like an index in a book, a table index helps you get at the data you want without having to read through the whole table. Indexes can exist on single column or on multiple columns. Indexes appear in the form of B-trees. And, as for books, you can have multiple indexes for a single table.

To create a new index using a wizard:

1   Open an Object Wizard for an index.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Indexes (Sybase) - Properties.

   • **Columns** panel - for details, see Indexes (Sybase) - Columns.

   • **Partition** panel - for details, see Indexes (Sybase) - Partition.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

INDEXES (SYBASE) - PROPERTIES

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner, **Table Name**, and **Name**	Select or provide identifying information.
Attributes	Provide or select values for the following Sybase options: **Index Type**, **Clustered**, **Ignore Duplicate Key**, **Ignore Duplicate Rows**, and **Maximum Rows Per Page** properties.
Storage	Provide or select values for the following Sybase options: **Reserve Page Gap**, **Segment Name**, **Fill Factor**, **Prefetch Strategy**, and **MRU Replacement Strategy**.

INDEXES (SYBASE) - COLUMNS

Select a column from the **Columns** dropdown and specify a **Sort** option.

Use the New button to add more columns to the index. Use the Delete button to remove selected columns from the index.

INDEXES (SYBASE) - PARTITION
Click **Add** to open the [Index Partition wizard (Sybase)](#).

INDEX PARTITION WIZARD (SYBASE)
The Sybase Index Partition wizard can be opened from the following editors and wizards:

- [Index Wizard (Sybase)](#)

- [Indexes Editor (Sybase)](#)

Use the following topics as a guide to understanding and setting the options on this wizard:

Step	Settings and tasks	
Properties	Select a **Locality**.	
Partition Definitions	Click the Define a new partition button to open a dialog that lets you provide a name, select a segment, click the New button to add values, and then **Add** the partition definition.	

When finished, click the **Finish** button.

LOGIN WIZARD (SYBASE)

The Login wizard lets you build and submit an sp_addlogin call, adding a new login with a basic set of specified database, language, and password properties. An additional sp_locklogin call can be issued to lock the new login. The Login wizard also lets you:

- Grant roles to the login

- Create user accounts on databases for the role, optionally with an alias or by changing ownership

To create a new login using a wizard:

1 [Open an Object Wizard](#) for a login.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see [Logins (Sybase) - Properties](#).

- **Roles** panel - for details, see [Logins (Sybase) - Roles](#).

- **Users** panel - for details, see [Logins (Sybase) - Users](#).

- **DDL View** panel - for details, see [Create the New Object](#).

3 Finally, use the Execute button to create the object. For more information, see [Create the New Object](#).

> **NOTE:** Rapid SQL also provides support for login triggers, configured for individual logins. For details, [Logins Editor (Sybase)](#).

LOGINS (SYBASE) - PROPERTIES

When creating or editing a login, this tab/panel lets you work with the following settings:

Group	Setting	Description
Creation	**Name**	The user login name.
	Full Name	The full name of the user, an optional but useful identifier.
General	**Default Database**	The database where the user starts each session.
	Default language	The language in which prompts and messages are displayed.
	Locked	If selected, the sp_addlogin call is immediately followed by an sp_locklogin call, locking the new login and effectively disabling it.
	NOTE: After creating a login, this group also displays an assigned Login trigger. For details, see [Logins Editor (Sybase)](#).	
Authentication	**Password**	The initial password for the new user.
	Password expiration	The duration, in days, before the password expires.
	Minimum password length	The minimum number of characters a password can contain.
	Maximum login attempts	The maximum number of login attempt failures before the login is suspended.
	Authentication mechanism	Lets you select ASE, LDAP, PAM, or ANY authentication.

LOGINS (SYBASE) - ROLES

When creating or editing a login, this tab/panel lets you grant roles to the login by selecting the the check boxes assocaited with those roles.

For information on activating and deactivating roles for the current login in the current session, see [Activating/Deactivating Roles in the Current Session](#).

LOGINS (SYBASE) - USERS

When creating or editing a login, this tab/panel lets you maintain database user accounts for a login as follows:

- Add a user account for this login to a database by selecting a database from the **Databases where the Login does NOT have a User Account** and clicking the **Add** button. This opens the **Add User** dialog letting you provide additional details. For more information, see [Adding a login account to a database](#).

- Remove a user account for this login from a database by selecting a database from the **Databases where the Login HAS a User Account** and clicking the **Remove** button.

When adding a user account for a login to a database, the Add user dialog lets you provide details on the method used (sp_adduser, sp_addalias, or sp_changedbowner). Use the following table as a guide to setting options in this dialog:

Setting or Group	Description
User Type	Lets you select the method by which the user account will be added to the database: USER (sp_adduser), ALIAS (sp_addalias) or DBO (sp_changedbowner).
Creation	Login Name - displays the user name from the Master database. If you selected a User Type of USER, the following settings are available: User Name - is a new name for the user in the target database. Group Name - lets you select a group in the target database to which the user will be added.
Alias	LoginUserName - If you selected a User Type of ALIAS, this control lets you select the database user name to alias the login name to.
dbo	TransferAliasesAndPermissions - If you selected a User Type of DBO, this control lets you transfer aliases and their permissions to the new database owner.

For more information, see Login Wizard (Sybase) and Logins Editor (Sybase).

PRIMARY KEY WIZARD (SYBASE)

Primary key constraints make sure that no duplicate values or NULLS are entered in the columns you specify. You can use primary key constraints to enforce uniqueness and referential integrity. A table can only have a single primary key constraint.

To create a new primary key using a wizard:

1 Open an Object Wizard for a primary key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Primary Keys (Sybase) - Properties.

- **Columns** panel - for details, see Primary Keys (Sybase) - Columns.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

PRIMARY KEYS (SYBASE) - PROPERTIES

Provide or select the following:

- The **Table Owner** and **Table Name** of the table for which to create the index.

- A **Name** for the Index.

- Whether the index is **Clustered**.

- The **Maximum Rows Per Page**.

- A **Reserve Page Gap**.

- A **Segment Name**.

- A **Fill Factor** value.

PRIMARY KEYS (SYBASE) - COLUMNS

Use the **Column** dropdown to choose a column name and choose a **Sort** option for that column. Use the New button to add an additional column to the index or use the Drop button to delete selected columns.

PROCEDURE WIZARD (SYBASE)

Procedures are a reusable block of PL/SQL, stored in the database, that applications can call. Procedures streamline code development, debugging, and maintenance by being reusable. Procedures enhance database security by letting you write procedures granting users execution privileges to tables rather than letting them access tables directly.

To create a new procedure using a wizard:

1   [Open an Object Wizard](#) for a procedure.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   - **Properties** panel - for details, see [Procedures (Sybase) - Properties](#).

   - **Definition** panel - for details, see [Procedures (Sybase) - Definition](#).

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

PROCEDURES (SYBASE) - PROPERTIES

When creating or editing a procedure, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the procedure.
Name	Provide the name of the procedure.
Procedure Number	Provide a procedure number if you want to group identically-named procedures.
Recompile	Optionally, enable this feature to prevent Sybase from saving a plan for the procedure.

Procedures (Sybase) - Definition
Complete the provided CREATE PROCEDURE statement by typing or pasting the body of the procedure.

RULE WIZARD (SYBASE)

Rules promote data integrity by allowing you to validate the values supplied to a table column. They are reusable objects that you can bind to table columns or user datatypes. Check constraints are similar to rules, and are in fact the preferred way of restricting data. A column or user-defined data type can have only one rule bound to it, but a column can have both a rule and one or more check constraints associated with it. Not that a rule cannot apply to data already existing in the database at the time you're creating the rule and can't be bound to a system-created data type. If you create a new rule when one already exists, the new rule will override the previous one.

To create a new rule using a wizard:

1   Open an Object Wizard for a rule.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see Rules (Sybase) - Properties.

   • **Dependencies** panel - for details, see Rules (Sybase) - Dependencies.

   • **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

RULES (SYBASE) - PROPERTIES

When creating or editing a rule, this tab/panel lets you work with the following settings:

Setting	Description
Owner and **Name**	Select an **Owner** and provide a **Name** for the rule.
Restriction	Type the condition. The rule restriction is the condition that defines the rule and can be any expression valid in a WHERE clause and can include such elements as arithmetic operators, relational operators, and predicates (for example, IN, LIKE, BETWEEN).
Type	Choose STANDARD_RULE, AND_ACCESS_RULE, or OR_ACCESS_RULE.

RULES (SYBASE) - DEPENDENCIES

From the **Type** dropdown, choose Column or Datatype, and if you chose **Column**, choose a Table from the **Table** dropdown. The list on the left is populated with candidate columns or datatypes. To move a candidate from the list on the left to the dependencies column on the right, select the candidate and click **Add**. Remove columns or datatypes from the dependencies list on the right by selecting the column or datatype and clicking **Remove**.

SEGMENT WIZARD (SYBASE)

Segments allow you to control the placement of objects on database storage devices. A segment is a subset of the database devise on which a specific database is stored. Each database can contain as many as 32 segments. Each database includes system, logsegment, and default segments. Others can be added using the Segment Wizard. By judiciously placing large tables and nonclustered indexes on segments on different devices or segments of specific sizes, you can improve I/O throughput or control space usage.

To Open the Segment Wizard

1   On the Explorer expand the database node where you want to add the Segment.

2   Right-click the Segments node and select New.

The table that follows describes the fields you may encounter as you complete the wizard.

Required Information	Description
What is the name of the segment?	Self-explanatory
On which devices do you wish to place the segment?	Self-explanatory

When you click **Execute**, Rapid SQL opens the **Segments Editor**. Here you can extend the segment and see the objects stored in the segment.

TABLE WIZARD (SYBASE)

A table is a column-based arrangement of data in which the content of one column has a bearing on the other column(s). So, for example, a table might have a column for authors, another column for the books each author has written, and a third for the number of copies each title by a given author has sold. The data moves across the columns in rows.

> NOTE: You must have CREATE TABLE permissions to generate a new table.

To create a new table using a wizard:

1   Open an Object Wizard for a table.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Tables (Sybase) - Properties.

- **Columns** panel - for details, see Tables (Sybase) - Columns.

- **Indexes** panel - for details, see Tables (Sybase) - Indexes.

- **Constraints** panel - for details, see Tables (Sybase) - Constraints.

- **Partitions** panel - for details, see Tables (Sybase) - Partition.

- **Permissions** panel - for details, see Tables (Sybase) - Permissions.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

TABLES (SYBASE) - PROPERTIES

When creating or editing a table, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the table.
Name	Provide a name for the table
Segment Name	Specify the segment on which you want to place the table.
Maximum Rows Per Page	Specifying a number allows you to override the default. The default, 0, creates indexes with full pages and nonclustered indexes with full leaf pages. This number can be changed at any time.
Reserve Page Gap	
Identity Gap	Specify the size of the identity gap for the table.
MRU Replacement Strategy	When enabled, new pages are read into the least recent end of the page chain. When pages reach the most recent end of the chain, the pages are flushed.
Prefetch Strategy	Enabling this feature allows you to fetch as many as eight 2K data pages simultaneously instead of one at a time (the default).
Lock Scheme	Select a locking scheme of ALLPAGES, DATAPAGES, or DATAROWS.
Expected Row Size	If you specified a **Lock Scheme** of DATAROWS or DATAPAGES, provide an expected row size

TABLES (SYBASE) - COLUMNS

For each column in the table, click the **Add Column** button to create a column, provide a **Name** for the column and provide or select the remaining column attributes.

Use the **Delete** button to drop a selected column.

Use the arrow buttons to reposition the columns.

> **NOTE:** Because the smalldatetime datatype stores dates and time with less precision than the datetime datatype, before outputting you use the CAST or CONVERT functions to convert any boxes with the smalldatetime datatype to either VARCHAR or datetime datatypes.

TABLES (SYBASE) - INDEXES

Click **Add** to open the [Index Wizard (Sybase)](#).

TABLES (SYBASE) - CONSTRAINTS

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- [Primary Key Wizard (Sybase)](#)

- [Unique Key Wizard (Sybase)](#)

- [Foreign Key Wizard (Sybase)](#)

- [Create Synonym](#)

TABLES (SYBASE) - PARTITION

Click **Add** to open the [Table Partition wizard for Sybase ASE](#).

TABLES (SYBASE) - PERMISSIONS

Set up the user permissions for this table.

TABLE PARTITION WIZARD FOR SYBASE ASE

The Sybase Table Partition wizard can be opened from the following editors and wizards:

- [Table Wizard (Sybase)](#)

- [Tables Editor (Sybase)](#)

Use the following topics as a guide to understanding and setting the options on this wizard:

Step	Settings and tasks	
Properties	Select a **Partition Type** of ROUNDROBIN, RANGE, HASH, or LIST	
Columns (not available with a **Partition Type** of ROUNDROBIN)	For each column, click the Insert a new column button to create a column, provide a **Name** for the column and provide or select the remaining column attributes. Use the **Delete** button to drop a selected column. Use the arrow buttons to reposition the columns.	

Step	Settings and tasks	
Partition Definition (only available with a **Partition Type** of HASH or ROUNDROBIN)	Take one of the following actions: (1) Select the **Number Of Partitions** radio box, specify the **Number Of Partitions**, and for each partition, click the New button and choose a tablespace from the dropdown, or (2) select the **By Partition Name** radio box and for each partition, click the New button provide a name and then choose a tablespace from the dropdown.	
Range Definitions (only available with a **Partition Type** of RANGE)	Click the Define a new partition button to open a dialog that lets you provide a name, select a segment and **Add** the partition definition.	
List Definitions (only available with a **Partition Type** of LIST)	Click the Define a new partition button to open a dialog that lets you provide a name, select a segment, click the New button to add values, and then **Add** the partition definition.	

When finished, click the **Finish** button.

TRIGGER WIZARD (SYBASE)

The Triggers Wizard lets you build and submit a CREATE TRIGGER command, selecting a parent object type of table or view, and selecting the triggering events for the trigger. Depending on whether you are creating a trigger for a view or a table, the Wizard generates customizable CREATE TRIGGER syntax as follows:

- Tables - The wizard generates a FOR INSERT, UPDATE, DELETE trigger code template (all supported Sybase versions)

- Views - The wizard generates an INSTEAD OF trigger code template (Sybase ASE 15.0.2)

In both cases, you provide the SQL statements that are to make up the body of the FOR or INSTEAD OF trigger.

To create a new trigger using a wizard:

1   [Open an Object Wizard](#) for a trigger.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties panel** - for details, see [Triggers (Sybase) - Properties](#).

- **Definition panel** - for details, see [Triggers (Sybase) - Definition](#).

3   Finally, use the **Execute** button to create the object. For more information, see [Create the New Object](#).

Triggers (Sybase) - Properties

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type, Parent Owner, and Parent Name	Provide or select details on the target table or view.
Owner and Name	Select the owner and provide a name for the trigger being created.
Fire On Insert, Fire On Update, and Fire On Delete	Enable the events that fire the trigger.
	An INSERT trigger must be associated with an INSERT statement. For example, if a data load operation doesn't include an INSERT statement, the trigger won't be invoked.
	An UPDATE trigger can be associated with specific columns of the base table and will only be activated if those columns are updated.
	A DELETE trigger fires automatically after items in the table are deleted.

NOTE: For triggers with a table parent type, the editor lets you enable and disable the trigger, after creation. For details, see Triggers Editor (Sybase).

TRIGGERS (Sybase) - DEFINITION

Complete the CREATE TRIGGER statement by typing or pasting in content.

- See the Microsoft SQL Server Transact-SQL documentation for information on the syntax for Trigger bodies. For more information, see Accessing Third Party Documentation.

UNIQUE KEY WIZARD (SYBASE)

Unique keys can enforce logical keys that are not chosen as the primary key. In other words, you can use a unique key to ensure no duplicate values are entered in specific columns that are not a part of the primary key. Although you can only attach one primary key to a table, you can attach multiple unique keys. Also, you can use unique keys on columns that allow null values.

To create a new unique key using a wizard:

1 Open an Object Wizard for a unique key.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Unique Keys (Sybase) - Properties.

- **Columns** panel - for details, see Unique Keys (Sybase) - Columns.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

Important Notes

- If you are creating a non-clustered index constraint, you should place it on a separate segment from the target table.

UNIQUE KEYS (SYBASE) - PROPERTIES

Provide or select the following:

- The **Table Owner** and **Table Name** of the table for which to create the index.

- A **Name** for the Index.

- Whether the index is **Clustered**.

- The **Maximum Rows Per Page**.

- A **Reserve Page Gap**.

- A **Segment Name**.

- A **Fill Factor** value.

UNIQUE KEYS (SYBASE) - COLUMNS

Use the **Column** dropdown to choose a column name and choose a **Sort** option for that column. Use the New button to add an additional column to the index or use the Drop button to delete selected columns.

USER DATATYPE WIZARD (SYBASE)

User datatypes promote domain consistency by streamlining the definition of commonly used table columns in a database. You can build a customized datatype from system datatypes and bind defaults and rules to it to enhance integrity. When you reference the user datatype in a column, the column assumes all of the properties of the user datatype.

To create a new user datatype using a wizard:

1   Open an Object Wizard for a user datatype.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see User Datatypes (Sybase) - Properties.

- **DDL View** panel - for details, see Create the New Object.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USER DATATYPES (SYBASE) - PROPERTIES

When creating or editing a user datatype, this tab/panel lets you work with the following settings:

Setting	Description
Datatype	Provide a name for the datatype.
Type	Select the base datatype.
Size	Provide the size of the datatype.
Allow Nulls	Null has no explicitly assigned value. Null is not equivalent to zero or blank. A value of null is not considered to be greater than, less than, or equivalent to any other value, including another value of null.
Identity	This allows you to specify whether or not you want to limit access to the new datatype based on the user's privileges.
Default Binding	Defaults promote data integrity by supplying a default value to a column if the user does not explicitly provide one. They are reusable objects that you can bind to user datatypes.
Rule Binding	Rules promote data integrity by allowing you to validate the values supplied to a column. They are reusable objects that you can bind to user datatypes.

USER MESSAGE WIZARD (SYBASE)

A user message let's you write the error message users will see when a user-defined event transpires. The User Message Wizard lets you multiple language versions of a user message, specifying the message numbers, specifying the text for each language version, and binding the message to one or more integrity constraints.

To create a new user message using a wizard:

1 Open an Object Wizard for a user message.

2 Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see User Messages (Sybase) - Properties.

- **Information** panel - for details, see User Messages (Sybase) - Information.

- **Bindings** panel - for details, see User Messages (Sybase) - Bindings.

- **DDL View** panel - for details, see Create the New Object.

3 Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

USER MESSAGES (SYBASE) - PROPERTIES

When creating or editing a user message, this tab/panel lets you provide the message number parameter for the sp_addmessage system procedure calls that will create each language-based version of this user message.

To specify the message number for this user message

In the **Message Number** box, type a unique message number (20000 or greater).

USER MESSAGES (SYBASE) - INFORMATION

For each language in which the user message is to be available, the User Message Wizard/Editor builds a sp_addmessage system procedure call. This tab/panel manages the language and message text parameters for each sp_addmessage call.

To add a language version of the user message

1   Click the Add new text for the user message button. The **Create User Message Text** dialog opens.

2   From the **Language** dropdown, select the language for the message.

3   In the **Message Text** box, type the text of the message.

4   Click **OK**.

To edit an existing language version of the message

1   Select the version from the **Language**/**Message** list.

2   Click the Modify user message text button. The **Edit User Message Text** dialog opens.

3   In the **Message Text** box, replace the text of the message.

4   Click **OK**.

To delete an existing language version of the message

1   Select the version from the **Language**/**Message** list.

2   Click the Remove user message text button and verify when prompted.

USER MESSAGES (SYBASE) - BINDINGS

When creating or editing a user message, this tab/panel manages one or more sp_bindmsg system procedure calls that will be issued with the sp_addmessage calls that create or edit a user message. This lets you bind the user message to, or unbind the user message from foreign keys or check constraints.

To bind a user message to an integrity constraint

1   From the **Constraint Type** dropdown, select the type of integrity constraint (**Check Constraints** or **Foreign Keys**) to which you are binding the user message.

2   Select one or more check constraints or foreign keys from the list on the left and click **Bind**.

To unbind a user message from a currently bound integrity constraint

1   Select one or more check constraints or foreign keys from the list on the right and click **Unbind**.

USER WIZARD (SYBASE)

The User Wizard lets you create a user who will then have access to the database where you are registering him or her. You can also identify the appropriate user group and the system privileges you want to assign to the new user.

To create a new user using a wizard:

1   <u>Open an Object Wizard</u> for a user

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   • **Properties** panel - for details, see <u>Users (Sybase) - Properties</u>.

   • **DDL View** panel - for details, see <u>Create the New Object</u>.

3   Finally, use the **Execute** button to create the object. For more information, see <u>Create the New Object</u>.

You can use the User Editor to assign System and Object privileges as well as create an alias.

USERS (SYBASE) - PROPERTIES

When creating or editing a user, this tab/panel lets you work with the following settings:

Setting	Description
Login Name	Choose a login ID from the drop-down list.
Name	Provide the user name.
Group Name	Specify a group for the user.

VIEW WIZARD (SYBASE)

Views are SQL queries stored in the system catalog that customize the display of data contained in one or more tables. Views behave like tables because you can query views and perform data manipulation operations on them. However, views do not actually store any data. Instead, they depend on data contained in their base tables. Views can only be

To create a new view using a wizard:

1   Open an Object Wizard for a view.

2   Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

- **Properties** panel - for details, see Views (Sybase) - Properties.

- **Definition** panel - for details, see Views (Sybase) - Definition.

3   Finally, use the **Execute** button to create the object. For more information, see Create the New Object.

You can use the User Editor to assign System and Object privileges as well as create an alias.

VIEWS (SYBASE) - PROPERTIES

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Description
Owner	Select the owner of the view.
Name	Provide a name for the view.
Check Type	When enabled, when a row is modified through a view, this option makes sure the data remains visible through the view after the modification is committed.

VIEWS (SYBASE) - DEFINITION

Complete the CREATE VIEW statement by typing or pasting in the relevant query. Finally, use the **Execute** button to Create the New Object.

MODIFYING OBJECTS USING EDITORS

An Object editor is a tabbed dialog box that stores information about existing server and database types and objects, and lets you modify those items. For an introduction to object editors, see <u>Overview and common usage of object editors</u>.

> **NOTE:** Availability of editors for specific object type varies from DBMS to DBMS. Similarly, functionality offered for an editor for an object type common to two or more DBMS, will differ from DBMS to DBMS.

See the following topics for information the object editors available for each supported platform:

- <u>IBM DB2 for Linux, Unix, and Windows Object Editors</u>

- <u>IBM DB2 for z/OS Object Editors</u>

- <u>InterBase/Firebird Object Editors</u>

- <u>Microsoft SQL Server Object Editors</u>

- <u>MySQL editors</u>

- <u>Oracle Object Editors</u>

- <u>Sybase ASE Object Editors</u>

OVERVIEW AND COMMON USAGE OF OBJECT EDITORS

An Object editor lets you view and modify settings and properties of existing object types and servers on a datasource. It also lets you add new resources and provides access to related, datasource management facilities. For example, the Tables editor lets you add or insert, edit, or drop columns, work with permissions to work with that table, access information on physical storage and the distribution of data and indexes across table spaces, and so on.

Each tab on the Object editor lets you perform a logical task or collection of logical tasks for that object type. The Object editor toolbar has a set of commands common to all object types and also includes a **Command** menu with commands specific to the object type you are currently working with. For example:
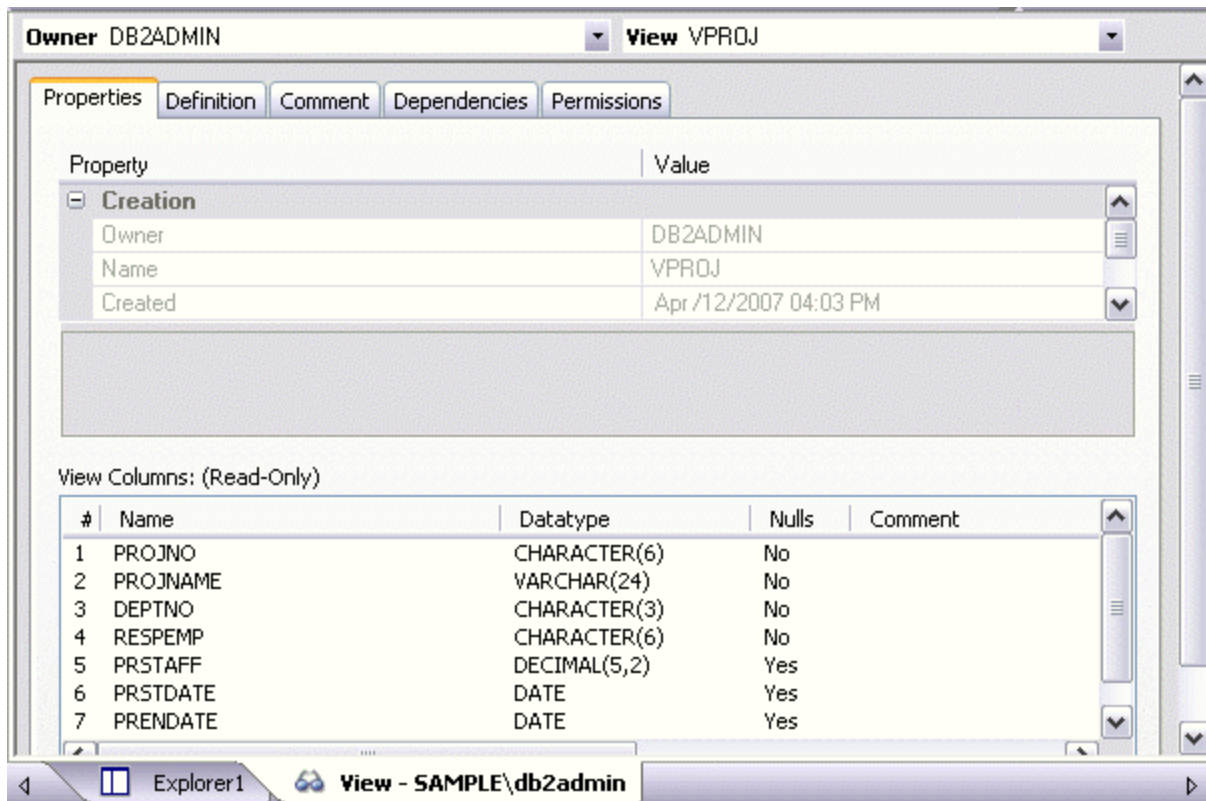
In order to work with object editors, you must be familiar with the following tasks:

- Opening an Object Editor on a server or database object.

- Viewing and Modifying Object Properties using functionality common to all object editors as well as object-specific functionality available to specific object editors.

- Previewing and Submitting Object Editor Changes to effect your changes to the datasource.

OPENING AN OBJECT EDITOR

Object editors are accessed from the Datasource Explorer pane of the Rapid SQL main window.

To open an Object editor on a specific database object

1 Connect to the datasource that contains the object you want to modify. For more information, see Datasource Management.

2 On the Datasource Explorer, expand the target datasource.

3 Continue to expand folders until objects of the target object type are displayed.

4 Right-click the object to be edited and select **Open** from the context menu.
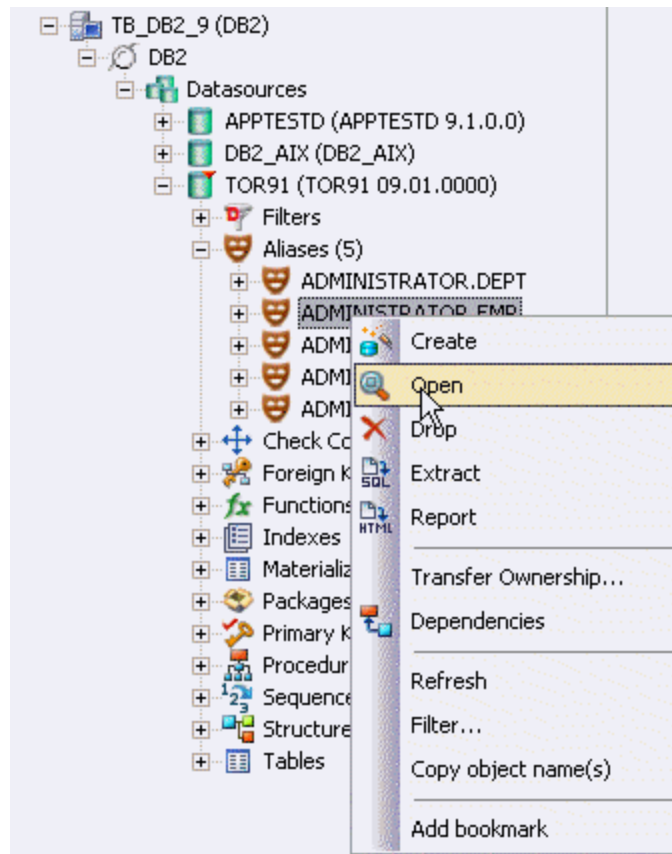
Rapid SQL opens an editor for that object type on the object you selected.

For information on editing objects, see Viewing and Modifying Object Properties.

VIEWING AND MODIFYING OBJECT PROPERTIES

There are two categories of tasks you can perform using an Object editor:

- Work with Tasks and Settings of Specific Object Types

- Work with Tasks and Settings Common to Object Editors

WORK WITH TASKS AND SETTINGS OF SPECIFIC OBJECT TYPES

Many of the tasks you perform using an Object editor are specific to the object type you are working with. For example, the Triggers editor lets you work directly with SQL code for the trigger while the Tables editor lets you work with columns of the table. Object-specific tasks are documented in the topics for specific Object editors provided later in this chapter.

WORK WITH TASKS AND SETTINGS COMMON TO OBJECT EDITORS

The following tasks can be performed against many of the supported Object editors:

- [Using the object editor toolbar](#)

- [Opening Another Object in the Same Object Editor Explorer](#)

- [Viewing the SQL/DDL for an Object](#)

- [Working with Privileges and Permissions](#)

- [Working with Object Dependencies](#)

- [Adding a Comment to an object](#)

USING THE OBJECT EDITOR TOOLBAR

The Object editor toolbar appears above the tabs of the Object editor.

It provides the following functions:

- **Create** - Launches a Rapid SQL object creation wizard. For more information, see [Creating objects](#).

- **Alter** - Enabled when a property of the current database has been modified. Opens a dialog that lets you preview the SQL code that will effect the modification and then submit the change. For more information, see [Preview](#).

- **Drop** -Lets you drop one or more database objects and remove their definition from the system catalog. For more information, see [Drop](#).

- **Extract** - Rapid SQL lets you extract data from one database to another database and extract the statements required to create objects into an Interactive SQL window. For more information, see [Extract](#).

- **Report** - Lets you generate detailed or summary reports on database objects. For details, see.[Report](#).

- **Command Menu** - Provides menu commands specific to the object being viewed or modified. For a listing of commands available, see the topic for the specific Object editor, later in this chapter.

- **Refresh** - Refreshes or clears Object editor contents.

- **Close** - Closes the Object editor and if appropriate, prompts you to verify any changes.

Commands on the Object editor toolbar are disabled if they do not apply to the object type being viewed or modified.

OPENING ANOTHER OBJECT IN THE SAME OBJECT EDITOR EXPLORER

When an object editor is open, the area between the object editor toolbar and the tabs has one or more dropdown lists. For example:
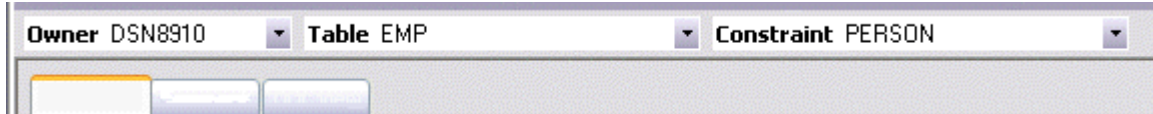
These lists allow you to qualify any object of the same type as the object currently displayed in the Object editor, and display information for that object in the current Object editor. The number of, and specific dropdown lists, differ according to the type of Object editor but there are always sufficient controls to uniquely identify another object of the same type.

To display the information for another object of the same type

- Use the dropdown lists to specify a different object.

VIEWING THE SQL/DDL FOR AN OBJECT

The object editors for most object types feature a **DDL** tab that lets you view an object's underlying SQL code.

To view the underlying DDL/SQL for a database object

1   Open an editor on an object type for which DDL code can be displayed. For details, see [Opening an Object Editor](#).

    See the topics for specific Object editors later in this chapter for information on whether that object type supports display of underlying SQL code.

2   Click the **DDL** tab.

WORKING WITH PRIVILEGES AND PERMISSIONS

When you open an Object editor on an object with associated privileges, the **Permissions** (or **Object Permissions** or **System Permissions**) tab for that editor displays the relevant privileges and lets you make changes accordingly:

For database objects such as tables or procedures, to which permissions or privileges are granted, a **Permissions** tab lets you manage permissions or privileges to that object from all grantees, such as users or groups, on that datasource. The tab has a navigation area, a command area, and a display area.
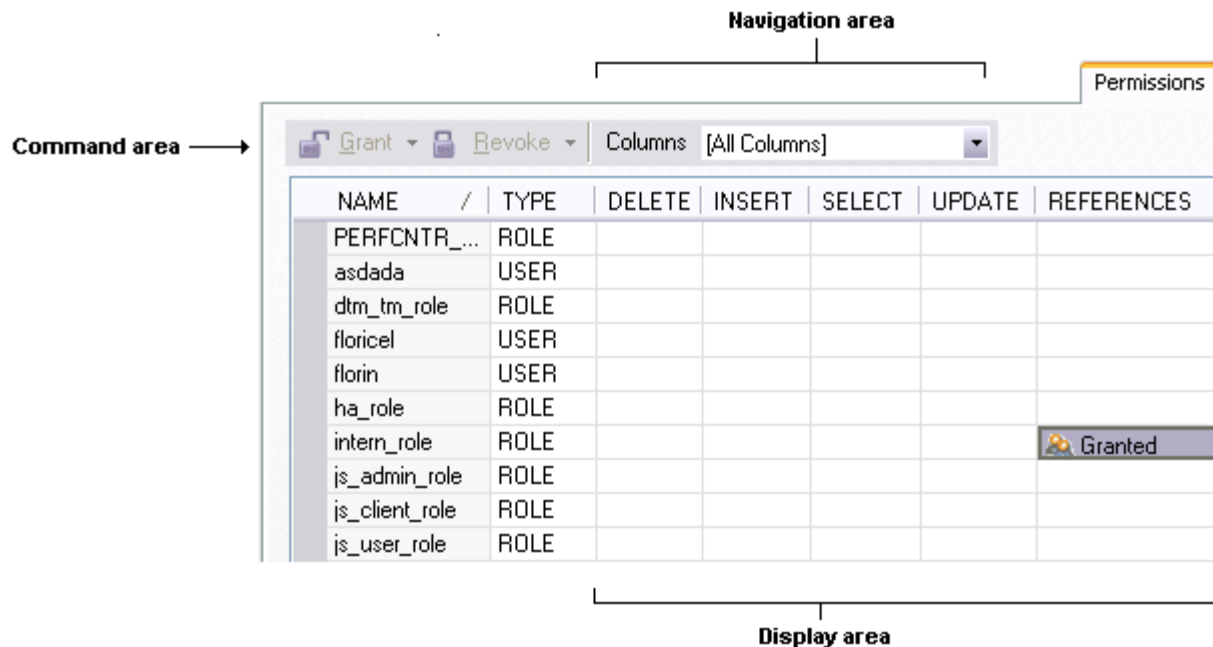
The Navigation area lets you change the content of the Display area to view more specific privilege details. For example, when viewing privileges for a database object such as a table, the dropdown lists in the navigation area, if present, let you drill down to populate the display area with privileges for a lower level component, such as a particular column. Similarly, when viewing privileges for a grantee such as a user, the **Object Permissions** tab's Navigation area lets you populate the display area with that user's permissions on specific object types, such as tables or procedures.
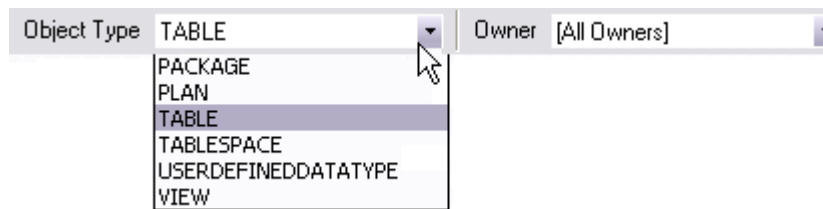
The Display area shows privilege details for an object or grantee. When viewing privileges for a grantee such as a user or role, the display area shows privileges for that recipient on objects currently qualified by the current selection in the Navigation area. When viewing privileges for an object such as a table, the display area shows the privileges for all grantees. Each cell in the Display area corresponds to a specific permissions and a cell representing a granted permission shows a distinctive icon.

The Command area lets you initiate granting or revocation of permissions.

To view or modify privileges using an Object editor

1   Open an editor on a database object with associated privileges or permissions. For details, see Opening an Object Editor.

2   Click the **Permissions** (or **Privileges** or **Object Privileges** or **System Privileges**) tab.

3   Use the dropdown lists in the navigation area, if present, to populate the Display area with details for more specific or different permissions.

4   In the Display area, select a cell corresponding to a specific permission.

5   Use the Command area controls to grant or revoke permissions.

6   Submit your changes. For details, see Previewing and Submitting Object Editor Changes.

WORKING WITH OBJECT DEPENDENCIES

For objects such as views and procedures, whose definition references other objects, such as tables, Rapid SQL lets you view all dependencies. The Object editors for referencing or referenced objects have a **Dependencies** tab that shows all dependencies for an object.
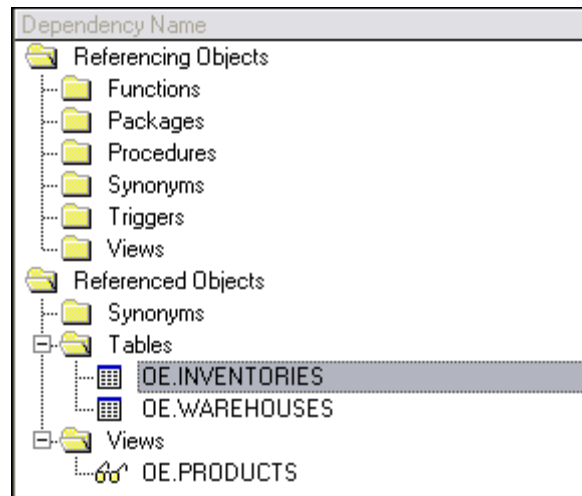
In addition to letting you view dependencies, Object editors let you drop a referencing or referenced object, or open an Object editor on that object.

To manage dependences for an object

1   Open an editor on a database object with referencing or referenced objects. For details, see Opening an Object Editor.

    See the object editor descriptions in subsequent topics to verify that an Object editor supports display of dependencies.

2   Click the **Dependencies** tab.

    The **Dependency Name** area lists all referenced or referencing objects. Objects are grouped according to their object type.

3   Optionally, select a referenced or referencing object in the right pane and either:

   • Use the **Edit** button to open an Object editor on that object

   • Use the **Drop** button to initiate dropping that object.

ADDING A COMMENT TO AN OBJECT

The object editors for certain object types feature a **Comment** tab that lets you add an explanatory note to specific object definitions. Comments are stored in the REMARKS column of the object's system-catalog.

To add a comment to an object

1   Open an editor on an object type that permits comments. For details, see Opening an Object Editor.

2   Click the **Comment** tab.
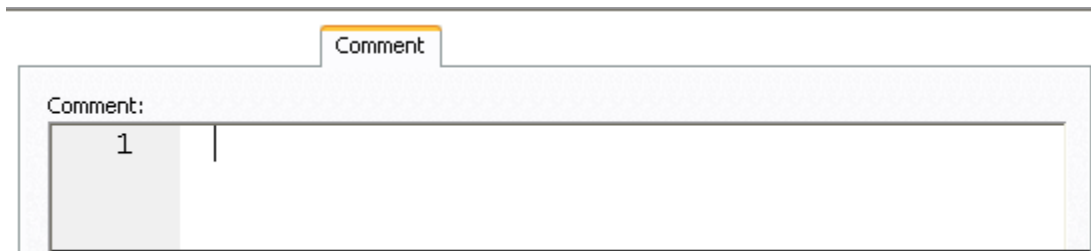
3   In the **Comment** area, type an explanatory note of up to 254 characters long.

   NOTE: Right-clicking in the Comment area opens a context menu that lets you perform edit text operations such as search, selection, copying and pasting, as well as coding-specific operations such as enabling/disabling line numbers and indenting lines.

4   Submit your changes. For details, see Previewing and Submitting Object Editor Changes.

PREVIEWING AND SUBMITTING OBJECT EDITOR CHANGES

After you use an Object editor to modify the settings or properties of a database object, you can preview the SQL that will be executed to effect those changes on the datasource. You can then submit the SQL for execution on the server.

To preview and submit object editor changes to a database object

1   Click the **Alter** button on the Object editor toolbar. For details, see Overview and common usage of object editors.

   A **Preview: Alter** dialog opens.
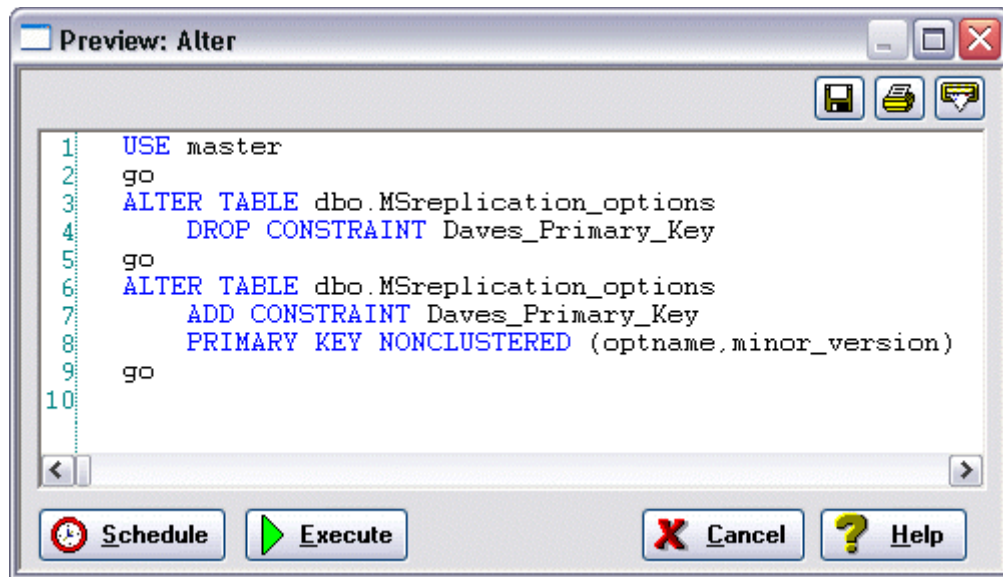
2 Use one of the following options to submit and execute your changes:

- **Execute** button - executes the changes immediately.

- **Schedule** button - opens a dialog that lets you schedule the change. For more information, see Scheduling.

IBM DB2 FOR LINUX, UNIX, AND WINDOWS OBJECT EDITORS

Rapid SQL includes an Object Editor for all supported IBM DB2 or Linux, Unix, and Windows objects. To see an Editor for a specific object, click the corresponding link below:

- Aliases Editor (IBM DB2 LUW)

- Check Constraints Editor (IBM DB2 LUW)

- Databases Editor (IBM DB2 LUW)

- Foreign Keys Editor (IBM DB2 LUW)

- Functions Editor (IBM DB2 LUW)

- Indexes Editor (IBM DB2 LUW)

- Materialized Query Tables Editor (IBM DB2 LUW)

- Packages Editor (IBM DB2 LUW)

- Primary Keys Editor (IBM DB2 LUW)

- Procedures Editor (IBM DB2 LUW)

- Sequences Editor (IBM DB2 LUW)

- [Structured Types Editor (IBM DB2 LUW)](#)

- [Tables Editor (IBM DB2 LUW)](#)

- [Tablespaces Editor (IBM DB2 LUW)](#)

- [Triggers Editor (IBM DB2 LUW)](#)

- [Unique Keys Editor (IBM DB2 LUW)](#)

- [User Datatypes Editor (IBM DB2 LUW)](#)

- [Users Editor (IBM DB2 LUW)](#)

- [Views Editor (IBM DB2 LUW)](#)

ALIASES EDITOR (IBM DB2 LUW)

The Aliases Editor lets you view basic alias properties, change the comment associated with an alias, and view the DDL that will be issued to alter the alias definition.

To edit an alias

1   Open an editor on the alias. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to the owning **Schema**, this tab lets you view the **Target Owner**, **Target Type**, and **Target Name**, of the referenced object.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

CHECK CONSTRAINTS EDITOR (IBM DB2 LUW)

The Check Constraints Editor lets you modify and enable/disable check constraints.

To edit a check constraint:

1   Open an editor on the check constraint. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Use the **Enabled** control to enable/disable the check constraint. You can also edit the condition in the **Check Condition** box. The **Table Column** button opens a dialog that lets you select and paste column names into the check condition expression.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

DATABASES EDITOR (IBM DB2 LUW)

The Databases Editor lets you manage database tablespace placement details and modify configuration parameters for the database.

To edit a database

1   Open an editor on the database. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Placement	Select a tablespace and click the **Edit** button to open an object editor on that tablespace. For more information, see <u>Tablespaces Editor (IBM DB2 LUW)</u>.
Options	Lets you modify the DB2 Database Manager configuration file parameters for the database. Select a parameter and click the **Edit** button to open a dialog that lets you change the current value. **NOTE:** To set database options for all future databases, set the database options on the model database.
DDL	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

FOREIGN KEYS EDITOR (IBM DB2 LUW)

The Foreign Keys Editor lets you manage column mapping for a foreign key, modify update and delete rule actions, and specify a NOT ENFORCED value.

To edit a foreign key

1   Open an editor on the foreign key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mapping	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes.
	If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you modify the **Delete Rule** (NO ACTION, RESTRICT, CASCADE, SET NULL) and **Update Rule** (NO ACTION, RESTRICT) actions. This tab also lets you deselect the **Enforced** check box to specify a NOT ENFORCED option.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

FUNCTIONS EDITOR (IBM DB2 LUW)

The Functions Editor lets you manage the body, inputs and outputs, and properties for a function.

To edit a function

1   Open an editor on the function. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Displays the **Language**, **Return Type**, **External Name**, **SQL Access Level**, **Origin**, **Threadsafe**, **Fenced**, **Scratchpad**, **Scratchpad Length**, **Allow Parallel**, **Final Call**, **Parameter Style**, **Inherit Special Registers**, **DBInfo**, **Deterministic**, **External Action**, **Called on Null Input**, and **Parameter CCSID** properties.	
Parameters	Lets you manage function parameters. On opening, this tab shows the existing parameters. Optionally you can:	
		Select a parameter from the list, modify the **Type** and if appropriate, the **Precision**, **Size**, and **Scale** of the parameter.
		Click the New button, provide a name for the new parameter and modify its attributes.
		Select a parameter and click the Drop button to delete the parameter.
		Select a parameter and use the arrow buttons to reorder the parameter list.

Tab	Settings and tasks
Return Scalar (only available for functions created with a **Function Type** of EXTERNAL SCALAR, SOURCED or TEMPLATE and for SQL **Function Type** functions with an explicit **Return Type** of SCALAR)	Lets you change the **Type** of a return scalar and depending on the type specified, modify the **Precision**, **Scale**, and **Size** of the return scalar.
Return Columns (only available for functions created with a **Function Type** of OLEDB and EXTERNAL TABLE and for SQL **Function Type** functions with an explicit **Return Type** of TABLE or ROW)	Lets you manage return columns for a function. On opening, this tab shows the existing columns. Optionally you can:
	Select a column from the list, modify the **Type** and if appropriate, the **Precision**, **Size**, and **Scale** of the parameter. You can also specify that the return column is to be treated **As Locator**.
	Click the New button, provide a name for the new column and modify its attributes.
	Select a parameter and click the Drop button to delete the parameter.
	Select a parameter and use the arrow buttons to reorder the parameter list.
Body	Lets you edit the body of the function.
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

INDEXES EDITOR (IBM DB2 LUW)

The Index Editor lets you manage columns and properties, and view space details for an index.

NOTE: IBM DB2 for Windows, Unix, and Linux, lets you segregate *Include columns*; columns that are to be part of the index but not part of the unique key.

To edit an index

1   Open an editor on the index. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks		
Columns	Lets you manage columns that make up the index. On opening, this tab shows the existing columns. Optionally you can:		
		Change the **Sort** order of a column.	
		Click the New button and select a column name from the dropdown, to add a column to the index.	
		Select a column and click the Drop button to delete the column from the index.	
		Select a column and use the arrow buttons to reorder the columns in the index.	
Properties	Lets you work with properties in the following categories:		
	Attributes	Lets you view the **Defined By** property. Lets you set the **Index Type** (UNIQUE or NONUNIQUE), specify the index as **Clustered**, and **Allow Reverse Scans**.	
	Storage	Lets you set **Percent Free** and **Minimum Percent Used** settings.	
Include Columns	Lets you Include columns that are to be part of the index but not part of the unique key. On opening, the tab shows a listing of the Include columns currently defined as part of this index.		
	Optionally, take one of the following actions:		
		Click the New button to add a new column to the index.	
		Select an existing column and click the Delete button to delete that column.	
Space	Lets you view the following property groups:		
	Attributes	**Make Unique**, **System Required**, **Total Keys**, **Page Fetch Pairs**, and **Distinct Keys**	
	Statistics	**Index Level**, **Cluster Ratio**, **Cluster Factor**, **Leaf Pages**, **Sequential Pages**, and **Density**	
	Cards	**First Key, First 2 Keys, First3 Keys, First 4 Keys,** and **Full Keys**	
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.		
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.		
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.		

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

MATERIALIZED QUERY TABLES EDITOR (IBM DB2 LUW)

The Materialized Query Tables Editor lets you manage the columns, base query and options for a materialized query table, and work with storage and performance settings.

To edit a materialized query table

1   Open an editor on the materialized query table. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Displays the currently defined columns for the materialized Query Table. Optionally, you can **Add**, **Insert**, or **Edit** columns for the materialized query table.	
Definition	**Base Query**	Lets you view the query that this materialized query table is based on.
	Materialized Query Table Options	Lets you specify an immediate or deferred **Refresh**, and explicitly set a Materialized Query Table as **Enabled** or **Disabled** or go with the **Defaul**t. This tab also displays whether the Materialized Query Table is maintained by the system or by a user.
Performance	**Tablespace Placement**	Displays **Table Tablespace**, **Index Tablespace**, and **Long Tablespace** settings for the materialized query table.
	Log Options	Lets you select logging options: **Logged initially** **Data Capture** - Lets you select none or change to **Include Longvar Columns**. NOTE: If you select the **Data Capture** option, the table name / column name cannot be longer than 18 bytes.
	Options	Let you set the **Percent Free**, **Locksize**, **Append**, and **Volatile** settings for the materialized query table.
	Partition Columns	Use the **Add** and **Delete** buttons to manage the partition columns for the materialized query table.
Space	**Page Information**	Lets you view **Row Count** and **Num. of Overflow Rows** values.
	Row Information	Lets you view **Num. of Pages with Rows**, **Num. of Pages**, **Percent Free** values.
Comment	For details on using this tab, see [Adding a Comment to an object](#).	
Permissions	For details on using this tab, see [Working with Privileges and Permissions](#).	
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

PACKAGES EDITOR (IBM DB2 LUW)

The Packages Editor lets you view contents and settings for a package.

To edit a package

1   Open an editor on the package. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you view the following settings: **Binder**, **Definer**, **Default Schema**, **Degree**, **Function Path**, **Language Level**, **SQL Math Warn**, **SQL Warn**, **Buffered Inset**, **Status**, **Code Page**, **Total Sections**, **Multi-node Bound**, **Intra-partition**, **Query Optimization**, **Cursor Blocking**, **Isolation Level**, **Date Time Format**, **Last Bind Time**, **Explicit Bind Time**, **Explain Level**, **Explain Snapshot**, and **Explain Mode**.
Statements	Shows the contents of a package statement. Optionally, you can select a statement and click **Explain** to copy the statement to an ISQL Editor window. For more information, see <u>ISQL Editor</u>.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PRIMARY KEYS EDITOR (IBM DB2 LUW)

The Primary Keys Editor lets you manage columns and properties for a primary key, and view space usage/allocation details.

To edit a primary key

1   Open an editor on the primary key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Displays the columns that make up the primary key. Optionally, you can:	
		Select a column and use the Delete key to remove the column from the primary key.
		Select a column and use the arrow keys to reorder the columns in the primary key.
Properties	Displays the owning **Table Schema** and **Table Name**, the **Name** of the primary key, a **Defined By** property, and **Percent Free** and **Minimum Percent Used** properties.	

Tab	Settings and tasks	
Space	Lets you view the following property groups:	
	Attributes	**Make Unique**, **System Required**, **Total Keys**, **Page Fetch Pairs**, and **Distinct Keys**
	Statistics	**Index Level**, **Cluster Ratio**, **Cluster Factor**, **Leaf Pages**, **Sequential Pages**, and **Density**
	Cards	**First Key, First 2 Keys, First3 Keys, First 4 Keys,** and **Full Keys**
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PROCEDURES EDITOR (IBM DB2 LUW)

The Procedures Editor lets you manage the body and parameters for a procedure.

To edit a procedure

1   Open an editor on the procedure. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you modify the following settings: **SQL Access Level** (MODIFIES SQL DATA, CONTAINS SQL, READS SQL DATA, NO SQL), number of **Result Sets**, **External Action**, New Save Point, Inherit Special Registers, **Deterministic**, and **Parameter CCSID**.	
	Lets you view the following settings: **Language** (C, COBOL, JAVA, OLE, SQL CLR), **External Name**, Threadsafe, **Fenced**, **Parameter Type** (GENERAL, DB2, GENERAL WITH NULLS, SQL), **Program Type** (MAIN, SUB and not enabled for a **Parameter Style** of SQL), and **DBInfo**.	
Parameters	Lets you manage procedure parameters. On opening, this tab shows the existing parameters. Optionally you can:	
		Select a parameter from the list, modify the **Type**, and if appropriate, the **Precision**, **Size**, and **Scale** of the parameter. You can also set the **Parameter Mode** to INPUT, OUTPUT, or INPUT_OUTPUT.
		Click the New button, provide a name for the new parameter and modify its attributes.
		Select a parameter and click the Drop button to delete the parameter.
		Select a parameter and use the arrow buttons to reorder the parameter list.
	NOTE: You cannot use host variables in the CALL statement for the name of the procedure.	
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
Body	Lets you modify the body of the procedure.	
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.	
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.	

Tab	Settings and tasks
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

SEQUENCES EDITOR (IBM DB2 LUW)

The Sequences Editor lets you manage cycle numbers, increments, the datatype, and other options for a sequence.

To edit a sequence

1 Open an editor on the sequence. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Definition	Lets you work with settings in the following categories:	
	Parameters	**Increment By** - Lets you specify the interval between sequence numbers. This integer value can be any positive or negative integer, but it cannot be 0. This value can have 28 or fewer digits. The absolute of this value must be less than the difference of MAXVALUE and MINVALUE. If this value is negative, then the sequence descends. If the increment is positive, then the sequence ascends. If you omit this clause, the interval defaults to 1.
		Minimum Value - Lets you specify the minimum value of the sequence. This integer value can have 28 or fewer digits.
		Maximum Value - Lets you specify the maximum value the sequence can generate. This integer value can have 28 or fewer digits.
	Next Sequence Numbers	Lets you work with sequence cycle numbers.
	Sequence Datatype	Lets you specify the datatype and width for the sequence.
	Options	Lets you specify **Cache Size**, **Cycle When Reach Max/Min**, and **Generate Numbers in Order** (useful when you are using the sequence number as a timestamp) values.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

STRUCTURED TYPES EDITOR (IBM DB2 LUW)

The Structured Types Editor lets you manage the attributes, methods and body for a structured type.

To edit a structured type

1   Open an editor on the structured type. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Displays the initially-defined values for the following properties: **Instantiable**, **Final Type**, **With Function Access**, **Without Comparisons**, **Inline Length**, **Supertype Schema**, **Supertype Name**, **Cast (Source as Ref) With**, **Cast (Ref as Source) With**, **Reference Using**, **Size**, **Precision**, and **Scale**. For details on these properties, see Structured Type Wizard (DB2 LUW).	
Attributes	Displays the currently defined attributes for the structured type. Optionally, take one of the following actions:	
		Select an attribute and change the **Datatype** in the Property/Value list. Depending on the datatype you choose you can also provide additional datatype options. For details on option availability, see Structured Type Wizard (DB2 LUW).
		Add an attribute by clicking the **New** button and typing a name for the attribute.
		Select an attribute and click **Drop** to delete the attribute.
Methods	This tab lets you initiate creation of method specifications and prepopulate associated method bodies. On opening, this tab lists all method specifications associated with the structured type and for each method, includes name and language details. Optionally, take one of the following actions:	
		Click **Add** to begin the process of adding a new method specification. Similarly, select a method and click **Edit** to modify the method specification. For more information, see Adding or editing structured type methods.
		Select a method specification and click **Drop** to delete that method specification from the structured type.
	The **Create Body** and associated controls are available for selected methods for which no method body has yet been defined. They lets you prepopulate the method body definition:	
	External name - lets you provide a 'string" or SQL identifier for a method specified with a **Language** of C, JAVA, or OLE.	
	Transform group - lets you specify the transform group used when invoking the method. This setting is available for methods specified with a **Language** of C, JAVA, or OLE.	
	As identifier - This setting is available for methods specified with a **Language** of C. When this check box is checked, the method body is created with the external name as provided. Otherwise, the external named provided will appear within quotes in the CREATE METHOD statement on the **Body** tab.	
	Inherit isolation level with lock request - specifies whether the INHERIT clause is specified as INHERIT ISOLATION LEVEL WITHOUT LOCK REQUEST or INHERIT ISOLATION LEVEL WITH LOCK REQUEST. This setting is available for methods specified with a Language of SQL.	
	After specifying options, click **Create Body** to work with the CREATE METHOD statement you generated on the editor's **Body** tab.	

Tab	Settings and tasks
Body	Lets you view and modify the CREATE METHOD statement generated with your choices on the **Methods** tab.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

ADDING OR EDITING STRUCTURED TYPE METHODS

The **Add Method** and **Edit Method** wizards let you work with the methods of a structured type. They let you specify basic properties, parameter and return value details for methods.

To add or edit a structured type method:

1   Open an editor on a structured type. For details, see <u>Opening an Object Editor</u>.

2   Navigate to the **Methods** panel, and then open a wizard using one of the following techniques:

   • Click **Add** to create a new method

   • Select an existing method and click **Edit**.

3   Use the following table as a guide to understanding and modifying the settings on the tabs of this wizard:

Tab		Settings and tasks
Properties		When adding a method this pane lets you specify a **Name**, **Specific Name**, **Language** (OLE, SQL, JAVA, C), and **SQL Access Level** (CONTAINS SQL, READS SQL DATA, NO SQL). When editing a method, these properties are for display only.
Advanced (available after clicking the **Advanced** button on the **Properties** pane)		Lets you select or enable the following ADD METHOD options: **Deterministic**, **External Action**, **Called on NULL Input**, **DBINFO**, **Fenced**, **Allow Parallel**, **Scratchpad**, **Scratchpad Length**, **Parameter Style** (SQL, DB2GENERAL), and **Final Call**.
Parameters		Lets you work with input parameters:
		Add a parameter by clicking the New button and typing a name for the parameter.
		Select a parameter and change the **Datatype** in the Property/Value list. Depending on the datatype you choose you can also provide **Precision**, **Size**, and **Scale** options.
		Delete a selected parameter by clicking the DELETE button.
		Order the parameters using the arrow buttons.
Return		Lets you provide details of the return value of the method:
	Return Datatype	Lets you specify the base **Type** of the return value. Depending on the datatype you choose you can also provide **Precision**, **Size**, and **Scale** options.
	Cast Datatype	This group lets you optionally use the CAST FROM form of the ADD METHOD RETURN clause. It lets you have the method return a different datatype, cast from the datatype specified in the **Return Datatype** group. This feature must be **Enabled** and provide the same type options as the **Return Datatype** group.

Tab		Settings and tasks
		An **As Locator** option specifies that the method return a LOB locator instead of the actual value. The option is only available for LOB and LOB-based datatypes. **As Locator** applies to a LOB type in the **Cast Datatype** group, if specified. Otherwise it applies to a LOB type in the **Return Datatype** group

4   When ready click **Finish**.

TABLES EDITOR (IBM DB2 LUW)

The Tables Editor lets you manage basic properties, columns, dimension columns, distribution key columns, partitions, tablespaces, and constraints for a table.

> **TIP:** Before modifying a table, familiarize yourself with the material in Altering Tables for IBM DB2 LUW for Linux, Unix, and Windows.

To edit a table

1   Open an editor on the table. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab		Settings and tasks
Columns		Displays the currently defined columns in the table. Optionally, you can:
		Select a column, and in the **Property/Value** list modify property values for that column.
		Click **Add Column**, provide a name for the new column, and set property values for the column.
		Select a column and click **Delete** to remove the column from the table.
Properties		Displays the **Name, Created**, **Last RunStats**, **Invalidate Time**, and **Defined By** properties. Lets you set **Percent Free**, **Lock Size**, **Append Data**, **Volatile**, **Compress**, **Row Compression**, **Security Policy**, **RestrictDrop**, **LogIndexBuild**, **CCSID**, **Do not initially log**, and **Data Capture** properties.
Status		Lets you view the following statistics: **Total Number of Rows**, **Number of Overflow Rows**, **Number of Pages With Rows**, **Number of Page**s, **Table Status**, **Row Organization**, **RowTypeSchema**, **RowTypeName**, **AccessMode**, **ActiveBlocks**, **AvgCompressedRowSize**, **AvgRowCompressionRatio**, **CodePage**, **PercentOfPagesSaved**, **LastRegeneratedTime**, and **ProtectionGranularity**.
Partition		This tab provides details on the partition columns and data partitions for the table. Optionally you can:
		Use the New or Delete buttons to create or drop a partition column or data partition.
		Use the Edit button to edit a data partition.
		Use the **Commands** menu to attach or detach a data partition.
Tablespaces	**Data Tablespace**, **Long Tablespace**, and **Index Tablespace**	Lets you view the **Database partition Group**, **Managed By**, **Page Size**, and **Extent Size** properties. Lets you choose the **Name** of a tablespace.

Tab	Settings and tasks
Dimensions	Lets you group columns to form a dimension:
	Click the New button to add a new column to the dimension for the table.
	Select a column and click the Edit button to modify the dimension column properties.
	Select a column and click the Delete button to drop a column from the dimension.
Distribution Key Columns	Lets you group one or more columns to form a distribution key:
	Click the New button to add a new column to the dimension for the table.
	Select a column and click the Edit button to modify the dimension column properties.
	Select a column and click the Delete button to drop a column from the dimension.
Indexes	Lets you manage indexes for a table:
	Click **Add** to open a dialog that lets you add a new index to the table.
	Select an index and click **Edit** to open a dialog that lets you edit index properties.
	Select an index and click **Drop** to open a dialog that lets you remove the index from the table.
Constraints	Lets you manage primary key, unique key, foreign key, and check constraints for a table. Constraints are grouped by type, under folders:
	Select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type.
	Select a constraint and click **Edit** to open a dialog that lets you modify the constraint details.
	Select a constraint and click **Drop** to remove the constraint.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
Permissions	For details on using this tab, see [Working with Privileges and Permissions](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

ALTERING TABLES FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

The ALTER TABLE command of Transact-SQL is limited to adding NULL columns to the end of a table and to adding or dropping constraints. Unfortunately, this scenario does not address many requirements of administrators and developers who need to add, delete or modify columns more broadly:

- Add columns anywhere in a table, not just the end

- Add columns that do not permit a NULL value.

- Change the NULL/NOT NULL status of table columns

- Change column datatypes to other compatible types

- Change the length of datatypes

- Delete a column

Due to the limitations of the ALTER TABLE command, the only way to make broader modifications is to write SQL scripts that step through all desired changes. To perform an enhanced table alter, Rapid SQL constructs an SQL script that completes the following steps:

1 Renames the existing table so that the original and its data remain intact

2 Builds a CREATE TABLE statement with the new table definition, including declared defaults, primary key and check constraints

3 Builds an INSERT statement to copy data from the original, renamed table to the new one

4 Builds foreign keys on the new table

5 Reapplies any privileges granted on the table

6 Rebuilds all dependencies on the new table, including indexes, triggers, procedures, packages, functions and views. When rebuilding procedures, functions, packages and views, Rapid SQL also rebuilds any permissions on them.

TABLESPACES EDITOR (IBM DB2 LUW)

The Tablespaces Editor lets you manage containers, basic properties, performance, space usage, and associated objects for a tablespace.

To edit a tablespace

1 Open an editor on the tablespace. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with settings in the following categories:	
	Tablespace	Lets you view the **Type** (REGULAR, LARGE, TEMPORARY, or USER TEMPORARY), **Use Automatic Storage**, **Managed By**, and **Database Partition Group** properties. Lets you modify the **Buffer Pool** and **Drop Recovery** properties.
	Performance	Lets you view the **Page SIze** and **Extent Size** properties. Lets you specify or select the **Prefetch Automatic**, **Overhead**, **Transfer Rate**, and **File System Caching** properties.
	Automatic Storage	Lets you work with the **AutoResize** (enables following size settings), **Initial Size**, **Increase Size**, **Max Size Unlimited**, and **Max Size** (enabled if **Max Size Unlimited** is not set) attributes. The **Initial Size** and **Max Size** values can be provided as kB, mB, or gB values.

Tab	Settings and tasks	
Container	This option is only available for database managed tablespaces. This tab lets you add or delete containers of a tablespace.	
		Select an existing container and in the Property/Value list, use the **Size** controls to RESIZE, EXTEND, or REDUCE the container.
		Add a container by clicking **New**, and then in the Property/Value list, provide **Name**, number of **Database Partitions**, **Type** (DEVICE, FILE) and original **Size**.
		Click **Delete** to remove a container.
Performance	Lets you manage settings in the following categories:	
	Page Setting	Lets you view **Page SIze** and **Extent Size** settings and modify the **Prefetch Size** setting.
	I/O Setting	Let you modify the **Overhead** and **Transfer Rate** settings.
	Dropped Table Settings	Lets you view the **Recovery Status**.
	Defaults	Lets you view the **Nodegroup** and specify a **Bufferpool**.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
Space	Lets you view the table usage and the distribution of space for a tablespace. Specific statistics include **Free Pages**, **Used Pages**, **Reserved Pages**, and **Total Pages**.	
Objects	Lets you manage database objects associated with the tablespace. Objects are organized in a tree structure with folders containing the objects. Optionally, take one of the following actions:	
		Select an object and click **Edit** to open an object editor on the selected object.
		Select an object and click **Drop** to initiate dropping the selected object.
Privileges	For details on using this tab, see <u>Previewing and Submitting Object Editor Changes</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

TRIGGERS EDITOR (IBM DB2 LUW)

The Triggers Editor lets you view properties for a trigger.

To edit a trigger

1   Open an editor on the trigger. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Trigger Timing**, **Trigger Events**, **Trigger Type**, **Object Status**, **Define**r, and **Function Path** properties.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.

Tab	Settings and tasks
Definition	Lets you view and modify the CREATE TRIGGER DDL that will implement your changes. To modify a trigger, Rapid SQL must drop then create the trigger.

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

UNIQUE KEYS EDITOR (IBM DB2 LUW)

The Unique Keys Editor lets you manage columns and basic properties for a unique key, as well as view space details.

To edit a unique key

1 Open an editor on the unique key. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the unique key. On opening, this tab shows the existing columns. For each column, the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
	Click the New button and select a column name from the dropdown, to add a column to the index.	
	Select a column and click the Drop button to delete the column from the index.	
Properties	Displays the owning **Table Schema** and **Table Name**, the Name of the primary key, a **Defined By** property, and **Percent Free** and **Minimum Percent Used** properties.	
Space	Lets you view settings in the following categories:	
	Attributes	Lets you view **Make Unique**, **System Required**, **Total Keys**, **Distinct Keys,** and **Page Fetch Pairs** settings.
	Statistics	Lets you view **Index Level**, **Cluster Ration**, **Cluster Factor**, **Leaf Pages**, **Sequential Pages**, and **Density** settings.
	Cards	Lets you view **First Key**, **First 2 Keys**, **First 3 Keys**, **First 4 Keys** and **Full Keys** settings.
Comment	For details on using this tab, see [Adding a Comment to an object](#).	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

USER DATATYPES EDITOR (IBM DB2 LUW)

The User Datatypes Editor lets you manage the basic properties of a user datatype.

To edit a user datatype

1   Open an editor on the user datatype. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the user **Datatype** name. Lets you select a new **Owner** or **Type**, and depending on the type specified, offers additional type-specific properties such as **Size**. Also lets you modify the **Allow Bit Data** property, if appropriate.
Comment	For details on using this tab, see Adding a Comment to an object.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

USERS EDITOR (IBM DB2 LUW)

The Users editor lets you manage object dependencies and permissions for a user.

To edit a user

1   Open an editor on the user. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Displays the user **Name**.	
Objects	Lets you manage database objects associated with the user. Objects are organized in a tree structure with folders containing the objects. Optionally, take one of the following actions:	
		Select an object and click **Edit** to open an object editor on the selected object.
		Select an object and click **Drop** to initiate dropping the selected object.
Object Permissions and **System Permissions**	For details on using these tabs, see Working with Privileges and Permissions.	
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.	

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

VIEWS EDITOR (IBM DB2 LUW)

The Views Editor lets you manage the columns as well as view and modify properties for a view.

To edit a view

1   Open an editor on the view. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays all columns for the view. Details for each column include the **Column Name**, the **Datatype** (and if applicable, with the precision in parentheses), and whether or not **Nulls** are allowed for that column. You can also set the **EnableQueryOptimization** property for the view.
Definition	Lets you view and modify the CREATE VIEW DDL that will implement any changes you make in this editor. To modify a view, Rapid SQL must drop then create the view.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
Permissions	For details on using this tab, see [Working with Privileges and Permissions](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

IBM DB2 FOR Z/OS OBJECT EDITORS

Rapid SQL includes an Object Editor for all supported IBM DB2 for z/OS objects:

- [Aliases Editor (IBM DB2 Z/OS)](#)

- [Check Constraints Editor (IBM DB2 Z/OS)](#)

- [Databases Editor (IBM DB2 Z/OS)](#)

- [Foreign Keys Editor (IBM DB2 Z/OS)](#)

- [Functions Editor (IBM DB2 Z/OS)](#)

- [Indexes Editor (IBM DB2 Z/OS)](#)

- [Packages Editor (IBM DB2 Z/OS)](#)

- [Plans Editor (IBM DB2 Z/OS)](#)

- [Primary Keys Editor (IBM DB2 Z/OS)](#)

- [Procedures Editor (IBM DB2 Z/OS)](#)

- [Stogroups Editor (IBM DB2 Z/OS)](#)

- [Synonyms Editor (IBM DB2 Z/OS)](#)

- [Tables Editor (IBM DB2 Z/OS)](#)

- [Tablespaces Editor (IBM DB2 Z/OS)](#)

- [Triggers Editor (IBM DB2 Z/OS)](#)

- [Unique Keys Editor (IBM DB2 Z/OS)](#)

- [User Datatypes Editor (IBM DB2 Z/OS)](#)

- [Users Editor (IBM DB2 Z/OS)](#)

- [Views Editor (IBM DB2 Z/OS)](#)

ALIASES EDITOR (IBM DB2 Z/OS)

The Aliases Editor lets you view details of an alias definition.

To edit an alias

1 Open an editor on the alias. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to the owning **Schema** and **Name** of the alias, this tab lets you view the **Target Owner**, **Target Type**, and **Target Name**, of the referenced object.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

CHECK CONSTRAINTS EDITOR (IBM DB2 Z/OS)

The Check Constraints Editor lets you view definition details and edit a check condition expression.

To edit a check constraint

1 Open an editor on the check constraint. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you view basic identification information on the check constraint: **Table Schema** and **Table Name**, the **Name** of the constraint and the date it was **Created**. You can also edit the condition in the **Check Condition** box. The **Table Columns** button acts a time saver in editing the condition. It opens a dialog that lets you select and paste column names into the check condition expression.

Tab	Settings and tasks
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see **Previewing and Submitting Object Editor Changes**.

DATABASES EDITOR (IBM DB2 Z/OS)

The Databases Editor lets you manage basic properties, permissions, and object dependencies for a database.

To edit a database

1 Open an editor on the database. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	The **Database Creation** group lets you view the **Name** and **Group Member** properties and set the **Type** property. The **Database Attributes** group lets you view the **Encoding Scheme**, **Create Date**, and **Last Altered** properties and lets you set the **Tablespace Buffer Pool**, **Index Buffer Pool**, **Storage Group**, and **CCSID** properties.
Permissions	For details on using this tab, see Working with Privileges and Permissions.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see **Previewing and Submitting Object Editor Changes**.

FOREIGN KEYS EDITOR (IBM DB2 Z/OS)

The Foreign Keys Editor lets you manage column mapping and specify a delete rule for a foreign key.

To edit a foreign key

1   Open an editor on the foreign key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mapping	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes. If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you specify a **Delete Rule** (CASCADE, NO ACTION, RESTRICT, SET NULL).
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

FUNCTIONS EDITOR (IBM DB2 Z/OS)

The Functions Editor lets you view and modify properties for a function, manage its inputs and outputs, and modify the code in the body of the function.

To edit a function

1   Open an editor on the function. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with properties in the following categories:	
	Identification	Lets you view the **Create Timestamp**, **Schema**, **Routine ID**, and **Origin** properties. Lets you set the **External Name** and **Collection ID** properties.
	Run Time	Lets you view the **Result Sets** property. Lets you set the **WLM Environment**, **WLM For Nested**, **ASU Time**, and **Run Options** properties.
	Structure	Lets you view the **Language**, **Parameter Style**, **Number of LOB Columns**, **Number of Parameters**, and **Allow Parallel** properties. Lets you set the **Program Type**, **Security Type**, **SQL Access Level**, **Inherit Special Registers**, **Fenced**, **DBINFO**, **Deterministic**, **Called On Null Input**, **External Action**, **Stay Resident**, **Final Call**, **Scratchpad**, and **Scratchpad Length** properties.
	Run Estimates	Lets you view the **Initial I/Os**, **I/Os Per Invocation**, **Initial Instructions**, and **Instructions Per Invocation** properties.
	Java Structure	Lets you view the **Java Class**, **Jar ID**, **Package ID**, **Method ID**, **Jar Schema**, and **Java Signature** properties.

Tab	Settings and tasks		
Parameters, **Return Columns**, and **Return Scalar**	On opening, these tabs displays the current function parameters, return columns or return scalar. Optionally, take one of the following actions:		
		Select a parameter, return column, or return scalar in the **Attributes** area, modify values, as permissible. Attributes differ by the tab you chose but typically include items such as **Type**, **Precision**, **Scale**, **As Locator**, and **Size**.
		Click the New button to provide datatype and size details for a new parameter, return column, or return scalar.
		Select a parameter, return column, or return scalar and click **Drop** to delete that parameter.
Body	Lets you modify the code in the text area of the tab.		
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.		
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.		
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.		

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

INDEXES EDITOR (IBM DB2 Z/OS)

The Indexes Editor lets you manage basic, storage, and space properties for an index, as well as work with its columns and partitions.

To edit an index

1 Open an editor on the index. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage the columns or key-expressions that make up the index. On opening, this tab shows the existing columns/key-expressions.	
	NOTE: Key-expression functionality is only enabled if the **Index on Expression** setting is enabled on the **Properties** tab.	
	Optionally you can:	
		Change the **Sort** order of a column or key-expression.
		If **Index on Expression** is disabled, click the New button and select a column name from the dropdown to add a column to the index.
		If **Index on Expression** is enabled, click the New button and type a valid key-expression in the **Expression** field, to add an expression to the index.
		Select a column or key-expression and click the Drop button to delete the column from the index.

Tab	Settings and tasks		
Properties	Lets you work with the **Buffer Pool**, **Piece Size**, **Close**, **Copy**, **Compress**, **Index on Expression**, and **Padded** properties. For more information on these properties, see [Indexes (DB2 z/OS) - Properties](#).		
Storage	Lets you select a dataset management scheme and provide associated attribute values. For details, see [Indexes (DB2 z/OS) - Storage](#).		
Partitions	Lets you work with partitions for the index. For each data partition, the listing shows the storage group, VCAT Catalog, primary and secondary space allocations, and if appropriate, the free space percentage, GBP Cache, limit key value, and erase on delete settings. Optionally, you can select a partition and click the Edit button top open an editor that lets you edit that partition.		
Space	Lets you view the following property groups:		
	Attributes	**Make Unique**, **System Required**, **Total Keys**, **Page Fetch Pairs**, and **Distinct Keys**	
	Statistics	**Index Level**, **Cluster Ratio**, **Cluster Factor**, **Leaf Pages**, **Sequential Pages**, **Density**, **DASD Storage**, and **Data Blocks/Key**.	
	Cards	**First Key** and **Full Keys**	
Comment	For details on using this tab, see [Adding a Comment to an object](#).		
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).		

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

PACKAGES EDITOR (IBM DB2 Z/OS)

The Packages Editor lets you manage properties and bind parameters for a package, as well as view environment information, package contents and associated plans.

To edit a package

1   Open an editor on the package. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you work with the following properties: **Average Size**, **Bind Time**, **Consistency Token**, **Dec31**, **Decimal Point**, **Group Member**, **Katakana Charset**, **Language**, **Mixed Character Set**, **Operative**, **Package Size**, **Package Source**, **PDS Name**, **Precompile Timestamp**, **Release Bound**, **String Delimiter**, **SysEntries**, **Timestamp**, and **Valid**.
Bind Parameters	Lets you set the **Schema Path**, **Creator**, **Qualifier**, **Explain**, **Validate**, **Degree**, **Dynamic Rules**, **Lock Isolation**, **Encoding**, **Page Writes**, **Resource Release**, **CurrentData**, **DB Protocol**, **Reoptvar**, **Keep Dynamic**, **Defer Prepare**, **SQL Error**, and **Optimization Hint** properties.

Tab	Settings and tasks
Plan/Packlists	Displays the plans contained in the package if the package was bound individually, or as part of a complete collection ID. Optionally, you can: Select a plan and click **Edit** to open the Plans editor on the plan. For more information, see Plans Editor (IBM DB2 Z/OS). Select a plan and click **Rebind** to initiate rebinding the plan. For more information, see Rebind Plans. Select a plan and click **Free** to initiate deleting the plan. For more information, see Drop.
Statements	Shows the contents of any package statement on the datasource. Optionally, select a statement and click **Explain** to copy the statement to an ISQL Editor window. NOTE: For more information, see ISQL Editor.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Privileges	For details on using this tab, see Working with Privileges and Permissions.
Environments	Displays run-time environments information for a package. Use the arrow buttons to move environments between the **Enabled Environments** and **Disabled Environments** lists. To modify a connection, select the environment and click Edit Connections. For more information, see Connection Editor.
Command	Displays the command that originally built the package.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

CONNECTION EDITOR

The Connection Editor lets you modify plan and package connections. It is opened from the Packages editor. For more information, see Packages Editor (IBM DB2 Z/OS).

The table below describes the options and functionality of the Connection Editor:

Option	Description
Connections	Displays the connections for the plan or package.
Add	Click to add the connection.

PLANS EDITOR (IBM DB2 Z/OS)

The Plans Editor lets you manage plan properties, view DBRM information, manage packages and contents, and manage run-time environments for a plan.

To edit a plan

1 Open an editor on the plan. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks		
Definition	Lets you work with the following attribute groups:		
	Properties		Lets you view the **Timestamp**, **Group Member**, **Plan Size**, **Average Size**, **Valid, Operative**, **Pkg. List Entries**, **SysEntries**, and **Release Bound** properties. Lets you set the **Owner**, **Qualifier**, and **Current Server** properties.
	Bind Parameters		Lets you set the **Schema Path**, **Explain**, **Validate**, **Degree**, **Dynamic Rules**, **Lock Isolation**, **Resource Acquire**, **Resource Release**, **Disconnect**, **Current Data**, **DB Protocol**, **Reoptvar**, **SQL Rules**, **Keep Dynamic**, **Defer Prepare**, **Encoding**, **Page Writes**, **Flag**, **Optimization Hint**, and **Cache Size** properties.
DBRMs	Lists details for each DBRM associated with a plan. Optionally, select a DBRM from the list and click **Edit** to open the DBRM editor on that DBRM.		
Packages	Displays details for each package associated with a plan. Optionally, you can:		
			Select a package and click **Edit** to open the Package editor on that package. For details, see Packages Editor (IBM DB2 Z/OS).
			Select a package and click **Rebind** to open the Rebind Packages dialog.
			Select a package and click **Free** to open the Free (Packages) dialog.
DBRM/Packages	Displays the entire contents of the plan, DBRMs and packages, in a single display. The first column contains either a 'D' for DBRM or 'P' for packages.		
	Optionally, take one of the following actions:		
			Select a package list entry and click **Edit** to open the Package editor on that package. For details, see Packages Editor (Oracle).
			Select a package and click **Rebind** to open the Rebind Packages dialog.
			Select a package and click **Free** to open the Free (Packages) dialog.
Dependencies	For details on using this tab, see Working with Object Dependencies.		
Privileges	For details on using this tab, see Working with Privileges and Permissions.		
Environments	Displays run-time environments information for a plan. Optionally, you can:		
			Use the arrow buttons to move environments between the **Enabled Environments** and **Disabled Environments** lists.
			Select an environment and click **Edit Connections** to open the Connection editor.
Command	Displays the command that originally built the plan.		

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

PRIMARY KEYS EDITOR (IBM DB2 Z/OS)

The Primary Keys Editor lets you manage primary key columns as well as work with storage, space, and partitions for a primary key.

> **TIP:** The refresh button lets you refresh or clear the editor's contents, and log SQL.

To edit a primary key

1 Open an editor on the primary key. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Column	Lets you manage columns that make up the primary key. On opening, this tab shows the existing columns. For each column, the listing shows the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
		Change the **Sort** order of a column.
		Click the New button and select a column name from the dropdown, to add a column to the primary key.
		Select a column and click the Drop button to delete the column from the primary key.
		Select a column and use the arrow buttons to reorder the columns in the primary key.
Properties	Lets you set **Buffer Pool**, **Close**, **Copy** and **Piece Size** properties.	
S**torage**	Lets you view **Storage Group** and **VCAT Catalog** properties. Lets you set **Primary Space Allocation**, **Secondary Space Allocation**, **Erase**, **Free Page**, **Percent Free,** and **GBP Cache** properties.	
Partitions	Lets you work with partitions for the primary key.	
Space	Lets you view values in the following property groups:	
	Attributes	**Make Unique**, **System Required**, **Total Keys**, **Page Fetch Pairs**, and **Distinct Keys**.
	Statistics	**Index Level**, **Cluster Ratio**, **Cluster Factor**, **Leaf Pages**, **Sequential Pages**, and **Density**.
	Cards	**First Key** and **Full Keys**.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PROCEDURES EDITOR (IBM DB2 Z/OS)

The Procedures Editor lets you manage the properties and input/output parameters of a procedure.

To edit a procedure

1   Open an editor on the procedure. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with properties in the following categories:	
	Identification	Lets you view the **Schema** and **Routine ID** properties. Lets you set the **External Name** and **Collection ID** properties.
	Run Time	Lets you set the **Result Sets**, **WLM Environment**, **WLM For Nested**, **ASU Time**, and **Run Options** properties.
	Structure	Lets you view the **Language**, **Number of LOB Columns**, **Number of Parameters**, and **DBINFO** properties. Lets you set the **Program Type**, **Security Type**, **SQL Access Level**, **Parameter Style**, **Inherit Special Registers**, **Fenced**, **Commit on Return, Deterministic**, and **Stay Resident** properties.
	Run Estimates	Lets you view the **Initial I/Os**, **I/Os Per Invocation**, **Initial Instructions**, and **Instructions Per Invocation** properties.
	Java Structure	Lets you view the **Java Class**, **Jar ID**, **Package ID**, **Method ID**, **Jar Schema**, and **Java Signature** properties.
Parameters	Displays a listing of the existing input/output parameters for the procedure. For the selected parameter, the **Datatype** list shows details for that parameter, including the **Type** and **Parameter Mode** (INPUT, OUTPUT, INPUT_OUTPUT). Depending on the type other parameters such as **Size**, **Precision**, or **Scale** may be available for viewing or modification.	
	Optionally you can:	
		Select a parameter and in the **Datatype** list, modify details for that parameter.
		Click the Add button to add a new parameter, provide a name for the new parameter, and edit the parameter values in the **Datatype** list.
		Click the Delete button to drop a selected parameter.
		Use the arrow buttons to change the order location of a selected parameter.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
Body	Lets you modify the SQL code for procedures on the current datasource	
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

STOGROUPS EDITOR (IBM DB2 Z/OS)

The Stogroups Editor lets you view and modify volumes, manage function privileges, and view DDL for a stogroup.

To edit a stogroup

1   Open an editor on the stogroup. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Volume Devices	Shows details on volumes associated with the stogroup. Optionally you can:
	Click **Add** to open a dialog that lets you add volumes.
	Select a volume and click **Remove** to delete the volume.
Privileges	For details on using this tab, see [Working with Privileges and Permissions](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

SYNONYMS EDITOR (IBM DB2 Z/OS)

The Synonyms Editor lets you view base object information and manage database object dependencies for a synonym.

To edit a synonym

1   Open an editor on the synonym. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the owning **Schema**, the **Name** of the synonym, and the synonym's **Referenced Object Owner**, **Referenced Object Type**, and the **Referenced Object Name**.
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

TABLES EDITOR (IBM DB2 Z/OS)

The Tables Editor lets you: manage columns, basic properties, partitions, indexes, and constraints for a table.

> **NOTE:**   Before editing tables, refer to the material in Altering Tables for IBM DB2 z/OS.

To edit a table

1   Open an editor on the table. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab		Settings and tasks
Columns		Displays the currently defined columns in the table. For each column, the Property/Value list displays the **Name** and **Type** for the column. In addition, depending on the **Type** selected, the list also displays **Size**, **Scale**, **Identity Column**, **Allow Nulls**, **Default Value**, **Comment**, and **For Data** property values, as appropriate.
		The **Statistics** group displays a **Number of Distinct Values in the Column** value.
		Optionally, you can:
		Select a column, and in the **Property/Value** list modify property values for that column.
		Click **Add Column**, provide a name for the new column, and set property values for the column.
		Select a column and click **Delete** to remove the column from the table.
Properties		Lets you work with settings in the following categories:
	Table	Lets you view **EditProcedure**, **Table Type**, **Table Status**, **Check Flag**, **DBID**, and **OBID** properties. Lets you set **Volatile**, **Audit**, **RestrictDrop**, **Label**, and **ValidProc** properties.
	Tablespace Placement	Lets you select a **Tablespace**.
	Log Options	Lets you select a **Data Capture** option of DATA CAPTURE NONE or DATA CAPTURE CHANGES.
	Statistics	Lets you view the following statistics: **Last Runstats**, **Total number of rows**, **Average Row Length**, **Number of Pages**, **Percent Compressed Rows**, **Max Record Length**, and **DASD storage**.
Partitions		Displays existing partition columns and data partitions. Optionally you can add, edit, or delete partition columns and data partitions.
Indexes		Displays the list of indexes for the table. Optionally, take one of the following actions:
		Click **Add** to open a dialog that lets you add a new index to the table.
		Select an index and click **Edit** to open a dialog that lets you edit index properties.
		Select an index and click **Drop** to open a dialog that lets you remove the index from the table.

Tab	Settings and tasks
Constraints	Displays constraints in a tree structure. The tree contains folders which contain all constraints associated with the target table. The objects are organized in folders based on the type of constraint.
	Optionally take one of the following actions:
	Select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type.
	Select a constraint and click **Edit** to open a dialog that lets you modify the constraint details.
	Select a constraint and click **Drop** to remove the constraint.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

ALTERING TABLES FOR IBM DB2 Z/OS

The ALTER TABLE command of Transact-SQL is limited to adding NULL columns to the end of a table and to adding or dropping constraints. Unfortunately, this scenario does not address many requirements of administrators and developers who need to add, delete or modify columns more broadly:

- Add columns anywhere in a table, not just the end

- Add columns that do not permit a NULL value.

- Change the NULL/NOT NULL status of table columns

- Change column datatypes to other compatible types

- Change the length of datatypes

- Delete a column

Due to the limitations of the ALTER TABLE command, the only way to make broader modifications is to write SQL scripts that step through all desired changes. To perform an enhanced table alter, Rapid SQL constructs an SQL script that completes the following steps:

1   Renames the existing table so that the original and its data remain intact

2   Builds a CREATE TABLE statement with the new table definition, including declared defaults, primary key and check constraints

3   Builds an INSERT statement to copy data from the original, renamed table to the new one

4   Builds foreign keys on the new table

5   Reapplies any privileges granted on the table

6   Rebuilds all dependencies on the new table, including indexes, triggers, procedures, packages, functions and views. When rebuilding procedures, functions, packages and views, Rapid SQL also rebuilds any permissions on them

TABLESPACES EDITOR (IBM DB2 Z/OS)

The Tablespaces Editor lets you work with the basic properties and partitions for a tablespace as well as view space details, status, and objects stored on the tablespace.

To edit a tablespace

1   Open an editor on the tablespace. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with properties in the following categories:	
	Bufferpool	**Buffer Pool**
	Partitions and size	**Number of partitions**, Partition size (DSSIZE), Segment size, and Max rows per page
	Other parameters	**GBPCACHE**, **Compress**, **Track modified pages**, **Encoding scheme**, **Log**, **Define**, **Member Cluster**, **Close rule**, **Lock Size**, and **Maximum locks**
	For more information on these properties, see Tablespaces (DB2 z/OS) - Properties.	
Partitions	Displays a list of partitions for the tablespace. Details for each partition include the storage group, VCAT, primary and secondary space allocations, the free space portion of each page and free page frequency, the group buffer cache scheme, and whether modifications are tracked.	
	Optionally, you can:	
		Select a partition and click **Edit** to open a dialog that lets you modify properties for that partition.
		Select a partition and click **Clone** to open a dialog that lets you apply the attributes of the selected partition to another partition.
Status	Lets you display CLAIMERS, LOCKS, LPL, USE, or WEPR status details	
Space	Lets you view space usage and allocation details for the tablespace.	
Objects	Displays the objects stored on the tablespace. Objects are organized in a tree structure with folders containing the objects.	
	Optionally, you can:	
		Select an object and click **Edit** to open an object editor on that object.
		Select an object and click **Drop** to initiate dropping that object.
Permissions	For details on using this tab, see Working with Privileges and Permissions.	
DDL	For details on using this tab, see Viewing the SQL/DDL for an Object.	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

TRIGGERS EDITOR (IBM DB2 Z/OS)

The Triggers Editor lets you modify the CREATE TRIGGER statement and manage properties for a trigger.

To edit a trigger

1   Open an editor on the trigger. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Trigger Timing**, **Trigger Events**, **Trigger Type**, **Object Status**, **Definer**, and **Function Path** properties.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
Definition	Lets you modify the CREATE TRIGGER body for a trigger. To modify a trigger, Rapid SQL must drop then create the trigger.

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

UNIQUE KEYS EDITOR (IBM DB2 Z/OS)

The Unique Keys Editor lets you manage columns, basic properties, and partitions for a unique key, as well as view storage details.

To edit a unique key

1   Open an editor on the unique key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs
    of this editor:

Tab	Settings and tasks	
Column	Lets you manage columns that make up the primary key. On opening, this tab shows the existing columns. For each column, the listing shows the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
	Change the **Sort** order of a column.	
	Click the New button and select a column name from the dropdown, to add a column to the primary key.	
	Select a column and click the Drop button to delete the column from the primary key.	
	Select a column and use the arrow buttons to reorder the columns in the primary key.	
Properties	Lets you set the **Buffer Pool**, **Piece Size**, **Close**, and **Copy** properties.	
Storage	Lets you view the **Storage Group** and **VCAT Catalog** properties. Lets you set the **Primary Space Allocation**, **Secondary Space Allocation**, **Erase**, **Free Page**, **Percent Free**, and **GBP Cache** properties.	
Partition	Lets you work with partitions for the index.	
Space	Lets you view values in the following property groups:	
	Attributes	**Make Unique**, **System Required**, **Total Keys**, **Page Fetch Pairs**, and **Distinct Keys**
	Statistics	**Index Level**, **Cluster Ratio**, **Cluster Factor**, **Leaf Pages**, **Sequential Pages**, and **Density**
	Cards	**First Key** and **Full Keys**
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting
    Object Editor Changes</u>.

USER DATATYPES EDITOR (IBM DB2 Z/OS)

The User Datatypes Editor lets you manage basic properties of a user datatype.

To edit a user datatype

1   Open an editor on the user datatype. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with properties in the following categories:	
	Base Datatype	Lets you set **Type** and **Size** settings.
	Character Options	Lets you set the **For Data** and **CCSID** properties.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.	
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

USERS EDITOR (IBM DB2 Z/OS)

The Users Editor lets you manage permissions for a user and the objects owned by that user.

To edit a user

1   Open an editor on the user. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Displays the user **Name**.	
Objects	Lets you manage database objects associated with the user. Objects are organized in a tree structure with folders containing the objects. Optionally, take one of the following actions:	
		Select an object and click **Edit** to open an object editor on the selected object.
		Select an object and click **Drop** to initiate dropping the selected object.
Object Permissions and **System Permissions**	For details on using these tabs, see <u>Working with Privileges and Permissions</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

VIEWS EDITOR (IBM DB2 Z/OS)

The Views Editor lets you view columns for a view and work with the dependencies and permissions for the view.

To edit a view

1   Open an editor on the view. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays all columns for the view. Details for each column include the **Column Name**, the **Datatype** (and if applicable, with the precision in parentheses), and whether or not **Nulls** are allowed for that column.
Definition	Lets you view and modify the CREATE VIEW DDL that will implement any changes you make in this editor. To modify a view, Rapid SQL must drop then create the view.
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

INTERBASE/FIREBIRD OBJECT EDITORS

Rapid SQL includes an Object Editor for all supported InterBase/Firebird objects. To see an Editor for a specific object, click the corresponding link below:

- Blob Filters Editor (InterBase/Firebird)

- Domains Editor (InterBase/Firebird)

- Encryption Keys editor (InterBase/Firebird)

- Exceptions Editor (InterBase/Firebird)

- External Functions editor (InterBase/Firebird)

- Foreign Keys editor (InterBase/Firebird)

- Generators editor (InterBase/Firebird)

- Indexes editor (InterBase/Firebird)

- Primary Keys editor (InterBase/Firebird)

- Procedures editor (InterBase/Firebird)

- Roles editor (InterBase/Firebird)

- Shadows editor (InterBase/Firebird)

- Tables editor (InterBase/Firebird)

- Triggers editor (InterBase/Firebird)

- Unique Keys editor (InterBase/Firebird)

- Users editor (InterBase/Firebird)

- Views editor (InterBase/Firebird)

BLOB FILTERS EDITOR (INTERBASE/FIREBIRD)

The Blob Filters editor lets you modify the input and output types, entry point, and the module name of a blob filter declaration.

To edit a blob filter

1   Open an editor on the blob filter. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying the **Name** property, this tab also lets you modify the **Input subtype**, **Output subtype**, **Entrypoint**, and **Module name** properties. For details on these properties, see Blob Filters Editor (InterBase/Firebird).
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

DOMAINS EDITOR (INTERBASE/FIREBIRD)

The Domains editor lets you modify datatype details and other basic clause/option values of a domain.

To edit a domain

1 Open an editor on the domain. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying the **Datatype** name, this tab lets you work with the **Type**, **Allow Nulls**, **Size**, Precision, **Scale**, **Array**, Character Set, Collation, **LOB Segment Length**, **Default**, and **Check** settings. For details on these settings and their availability, see Domains (InterBase/Firebird) - Properties.
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

ENCRYPTION KEYS EDITOR (INTERBASE/FIREBIRD)

The Encryption Keys editor lets you modify the encryption algorithm and key length options for a key, as well as the padding and Cipher Block Chaining versus Electronic Cookbook details.

To edit an encryption key

1 Open an editor on the encryption key. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying the **Name** of the encryption key, this tab lets you work with the **Algorithm**, **Pad random**, **Init Vector random**, and **IsDefault** settings. This tab also lets you modify the associated **Password**. For details on these settings, see Encryption Keys (InterBase/Firebird) - Properties.
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

EXCEPTIONS EDITOR (INTERBASE/FIREBIRD)

The Exceptions Editor lets you modify the text of the message associated with an exception.

To edit an exception

1   Open an editor on the exception. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying the **Name** of the exception, this tab lets you work with the **Text** setting. For details on these settings, see Exceptions (InterBase/Firebird) - Properties.
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

EXTERNAL FUNCTIONS EDITOR (INTERBASE/FIREBIRD)

The External Functions editor lets you modify basic properties, input parameters, and return datatype of an external function.

To edit an external function

1   Open an editor on the external function. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	In addition to displaying the **Name** of the external function, this tab lets you work with the **EntryPoint** and **Module Name** settings. For details on these settings, see External Functions (InterBase/Firebird) - Properties.	
Parameters	Displays the input parameters to the external function. Optionally, you can:	
		Use the New button to add a new parameter and type a name for the parameter in the space provided. With the parameter selected, in the **Attributes** area select a **Type**, and if appropriate, provide or select **Precision**, **Scale**, and **Size** options.
		Modify a parameter by selecting it, and in the **Attributes** area editing the **Type**, **Precision**, **Scale**, and **Size** options.
		Use the Delete button to drop a selected parameter and use the arrow keys to reorder the parameter list.
		Use the arrow keys to change the list ordering of a selected parameter.

Tab	Settings and tasks
Return Type	Displays datatype details of the value returned by the external function.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

FOREIGN KEYS EDITOR (INTERBASE/FIREBIRD)

The Foreign Keys Editor lets you manage column mapping for a foreign key as well as modify the update and delete rule actions.

To edit a foreign key

1 Open an editor on the foreign key. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mapping	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes.
	If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you modify the **Delete rule** and **Update Rule** actions. For details on these properties, see <u>Foreign keys (InterBase/Firebird) - Properties</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

GENERATORS EDITOR (INTERBASE/FIREBIRD)

The Generators Editor lets you view the name and work with the object dependencies for a generator.

To edit a generator

1   Open an editor on the generator. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Name** of the generator and modify the **Current Value**. For details on these properties, see Generators (InterBase/Firebird) - Properties.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

INDEXES EDITOR (INTERBASE/FIREBIRD)

The Indexes Editor lets you modify the columns and basic properties for an index.

To edit an index

1   Open an editor on the index. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the index. On opening, this tab shows the existing columns. For each column, the listing displays the datatype and whether the table definition permits nulls. Optionally you can:	
		Click the New button and select a column name from the dropdown, to add a column to the index.
		Select a column and click the Drop button to delete the column from the index.
		Select a column and use the arrow buttons to reorder the columns in the index.
Properties	In addition to displaying the **Name** and associated **Table Name** for the index, this tab lets you modify the **Unique**, **Enabled**, and **Descending** properties. For details on these properties, see Indexes (InterBase/Firebird) - Properties.	
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.	

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

PRIMARY KEYS EDITOR (INTERBASE/FIREBIRD)

The Primary Keys Editor lets you modify the columns that make up a primary key.

To edit a primary key

1   Open an editor on the primary key. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the primary key. On opening, this tab shows the existing columns. For each column, the listing displays the datatype and whether the table definition permits nulls. Note that only columns defined with the **Allow Nulls** property disabled can be added to a primary key. Optionally you can:	
		Click the New button and select a column name from the dropdown, to add a column to the index.
		Select a column and click the Drop button to delete the column from the index.
		Select a column and use the arrow buttons to reorder the columns in the index.
Properties	Displays the **Name** of the index and the associated **Table Name**.	
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.	

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

PROCEDURES EDITOR (INTERBASE/FIREBIRD)

The Procedures Editor lets you modify the body and parameters for a procedure.

To edit a procedure

1   Open an editor on the procedure. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the **Name** of the procedure.

Tab	Settings and tasks
Parameters	Lets you manage procedure parameters. On opening, this tab shows the existing parameters. Optionally you can:
	Select a parameter from the list, modify the **Type**, and if appropriate, the **Precision**, **Size**, and **Scale** of the parameter. You can also set the **Parameter Mode** to INPUT or OUTPUT.
	Click the New button, provide a name for the new parameter and modify its attributes.
	Select a parameter and click the Delete button to delete the parameter.
	Select a parameter and use the arrow buttons to reorder the parameter list.
Comment	For details on using this tab, see Adding a Comment to an object.
Body	Lets you modify the body of the procedure.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see **Previewing and Submitting Object Editor Changes**.

ROLES EDITOR (INTERBASE/FIREBIRD)

The Roles Editor lets you manage the users assigned to a role and the permissions associated with the role.

To edit a role

1   Open an editor on the role. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the **Name** and **Authorization Owner** for the role.
Users	Let you manage users for the role. Use the **Join Role** and **Leave Role** buttons to move users between the **Users In Role** and **Users Not In Role** lists.
Object Permissions	For details on using these tabs, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see **Previewing and Submitting Object Editor Changes**.

SHADOWS EDITOR (INTERBASE/FIREBIRD)

The Shadows Editor lets you modify CONDITIONAL, AUTO, and MANUAL argument usage and the file specification for a shadow.

To edit a shadow

1   Open an editor on the shadow. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the **Shadow Set** number and lets you modify the **Conditional** and **Behavior** properties. For details on these properties, see <u>Shadows (InterBase/Firebird) - Properties</u>.
Storage	Use the **Delete** button to drop a selected file. Modify an existing file by selecting the file in the primary and secondary files list, and modify the **Physical File Name**, **Starting Page** (for secondary files), and **Size** properties. Add a new file by clicking **New**, and providing file specification values in the **Property/Value** list.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

TABLES EDITOR (INTERBASE/FIREBIRD)

The Tables Editor lets you manage basic properties, columns, indexes, and constraints for a table.

To edit a table

1   Open an editor on the table. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Columns	Displays the currently defined columns in the table. Optionally, you can:
	Select a column, and in the **Property/Value** list modify property values for that column. For details on the properties, see <u>Tables (InterBase/Firebird) - Columns</u>.
	Use the **Add Column** dropdown to add a new column to the bottom of the column list or to insert a new column above the currently selected column in the column list. Provide a **Name** for the column, and then set property values for the column.
	Select a column and click **Delete** to remove the column from the table.
Properties	Displays the **Name** of the table and the date and time **Created**.

Tab	Settings and tasks
Indexes	Lets you manage indexes for a table:
	Click **Add** to open a dialog that lets you add a new index to the table. For more information, see [Indexes Wizard (InterBase/Firebird)](#).
	Select an index and click **Edit** to open a dialog that lets you edit index properties. For more information, see [Indexes editor (InterBase/Firebird)](#).
	Select an index and click **Drop** to open a dialog that lets you remove the index from the table.
Constraints	Lets you manage primary key, unique key, foreign key, and check constraints for a table. Constraints are grouped by type, under folders:
	Select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type.
	Select a constraint and click **Edit** to open a dialog that lets you modify the constraint details.
	Select a constraint and click **Drop** to remove the constraint.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
Permissions	For details on using this tab, see [Working with Privileges and Permissions](#).
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

TRIGGERS EDITOR (INTERBASE/FIREBIRD)

The Triggers Editor lets you modify the basic properties, trigger body, and dependencies for a trigger.

To edit a trigger

1   Open an editor on the trigger. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the **Parent Type**, **Parent Name**, and **Name** properties for the trigger, and lets you modify the **Trigger Timing**, **Trigger Events**, **Position**, and **Enabled** properties. For details on these properties, see [Triggers (InterBase/Firebird) - Properties](#).
Body	Lets you work with the trigger body.
Comment	For details on using this tab, see [Adding a Comment to an object](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

UNIQUE KEYS EDITOR (INTERBASE/FIREBIRD)

The Unique Keys editor lets you modify the columns that make up a unique key.

To edit a unique key

1 Open an editor on the unique key. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the unique key. On opening, this tab shows the existing columns. For each column, the listing includes the datatype and whether the table definition permits nulls.	
	Note that only columns defined with the **Allow Nulls** property disabled can be added to a unique key.	
	Optionally you can:	
		Click the New button and select a column name from the dropdown, to add a column to the unique key.
		Select a column and click the Drop button to delete the column from the unique key.
		Select a column and use the arrow buttons to reorder the columns in the unique key.
Properties	Displays the **Name** of the unique key and the associated **Table Name**.	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

USERS EDITOR (INTERBASE/FIREBIRD)

The Users Editor lets you modify basic properties, assigned roles, and object permissions for a user.

To edit a user

1   Open an editor on the user. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying the login **Name** for the user, this tab lets you work with the **Password**, **Group**, **Default Role**, **First Name**, **Middle Name**, **Last Name**, **Active**, **System User Name**, **Description**, **GID**, and **UID** properties. For details on these properties, see <u>Users (InterBase/Firebird) - Properties</u>.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
Roles	Lets you assign the valid, non-default roles that a user can specify when logging in to the database. A default login role can be assigned for this user on the Properties tab/panel. For details, see <u>Users (InterBase/Firebird) - Roles</u>. To assign roles to the user, select the associated check box.
Object Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

VIEWS EDITOR (INTERBASE/FIREBIRD)

The Views editor lets you work with the CREATE VIEW DDL for a view, grant and revoke associated permissions, and work with the view's dependencies.

To edit a view

1   Open an editor on the view. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the **Name** of the view and a listing of the associated columns. Details for each column include the column **Name**, the **Datatype**, whether or not **Nulls** are allowed for that column, and an associated **Comment**.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
Definition	Lets you view and modify the CREATE VIEW DDL for the view.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

MICROSOFT SQL SERVER OBJECT EDITORS

Rapid SQL includes an Object Editor for all supported Microsoft SQL Server objects. To see an Editor for a specific object, click the corresponding link below:

- Check Constraints Editor (SQL Server)

- Databases Editor (SQL Server)

- Database Triggers Editor (SQL Server)

- Defaults Editor (SQL Server)

- Extended Procedures Editor (SQL Server)

- Foreign Keys Editor (SQL Server)

- Full-text Catalogs Editor (SQL Server)

- Full-text Indexes Editor (SQL Server)

- Functions Editor (SQL Server)

- Indexes Editor (SQL Server)

- Logins Editor (SQL Server)

- Partition Functions Editor (SQL Server)

- Partition Schemes Editor (SQL Server)

- Primary Keys Editor (SQL Server)

- Procedures Editor (SQL Server)

- Rules Editor (SQL Server)

- Schemas Editor (SQL Server)

- Synonyms Editor (SQL Server)

- Tables Editor (SQL Server)

- Triggers Editor (SQL Server)

- Unique Keys Editor (SQL Server)

- Users Editor (SQL Server)

- User Datatypes Editor (SQL Server)

- User Messages Editor (SQL Server)

- Views Editor (SQL Server)

CHECK CONSTRAINTS EDITOR (SQL SERVER)

The Check Constraints Editor lets yo view and modify check constraints properties and edit the check constraint expression.

To edit a check constraint

1   Open an editor on the check constraint. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Use the **Enabled** check box to enable or disable the check constraint. Use the **Not For Replication** check box to enable or disable the check constraint for replication.
	You can also edit the condition in the **Check Condition** box. The **Table Column** button opens a dialog that lets you select and paste column names into the check condition expression.
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

DATABASES EDITOR (SQL SERVER)

The Databases Editor lets you manage basic properties, log and data files for a database.

To edit a database

1 Open an editor on the database. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks		
Options	Lets you work with options in the following categories:		
	Creation Properties	Lets you choose values for the **Compatible Level** and **Owner** settings.	
		NOTE: The Compatible level option sets certain database behaviors to be compatible with the specified earlier version of Microsoft® SQL Server. The compatibility level affects the behaviors in the specified database, not the entire server.To set database options for all future databases, set the database options on the model database.	
		NOTE: When changing the database owner (dbo) select the check box to transfer the existing aliases of users who could act as the old dbo (including their permissions) to the new dbo.	
	Properties	Lets you set the following properties: **ANSI null default**, **ANSI nulls**, **ANSI padding**, **ANSI warnings**, **arithabort**, **auto create statistics**, **auto update statistics**, **auto close**, **auto shrink**, **concat null yields nul**, **cursor close on commit**, **db chaining**, **dbo use only**, **default to local cursor**, **merge publish**, **numeric roundabout**, **offline**, **published**, **quoted identifier**, **read only**, **recursive triggers**, **select into/bulkcopy/pllsort**, **single user**, **subscribed**, **torn page detection**, and **trunc log on chkpt**.	
Placement	Displays the currently defined data files for the database. Optionally, you can:		
		Select a file from the list on the left and in the **Device File Properties** group, modify the **Size**, **File Growth Rate**, **Max Size**, and **Unlimited Max Size** settings.	
		Click the **New** button, provide a **Device File Name** for the new file, and use the other settings in the **Device File Properties** group to provide additional details for the file.	
		Select a file from the list on the left and click **Delete** to remove the file.	
Transaction Log	Displays the currently defined transaction logs for the database. Optionally, you can:		
		Select a transaction log from the list on the left and in the **Log Device Properties** group, modify the **Size**, **File Growth Rate**, **Max Size**, and **Unlimited Max Size** settings.	
		Click the **New** button, provide a **Device File Name** for the new transaction log, and use the other settings in the **Log Device Properties** group to provide additional details for the transaction log.	
		Select a transaction log from the list on the left and click **Delete** to remove the transaction log.	
Space	Lets you view pie charts showing the data space usage and the transaction log (if available) space usage for the database.		
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.		

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

DATABASE TRIGGERS EDITOR (SQL SERVER)

The Database Triggers Editor lets you enable or disable a database trigger and modify the body of the generated CREATE TRIGGER statement.

> **NOTE:** This functionality is available for SQL Server 2005 and up.

To edit a database trigger:

1 Open an editor on the database trigger. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Select the **Enabled** check box to enable the database trigger or deselect it to disable the trigger. In addition to the name and creation date details, this tab also lets you view the **Trigger Timing** and **Encrypted** property settings you chose when creating the trigger. For details on these properties, see Database Triggers (SQL Server) - Properties.
Definition	Lets you modify the CREATE TRIGGER statement.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Working with Object Dependencies.
Trigger Events	Lets you view the DDL events you selected to fire this trigger.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

DEFAULTS EDITOR (SQL SERVER)

The Defaults Editor lets you change the owner and value of a default.

To edit a default

1 Open an editor on the default. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you modify the **Value** of the Default and the owning **Schema**.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

EXTENDED PROCEDURES EDITOR (SQL SERVER)

The Procedures Editor lets you modify the library name for an extended procedure. It also lets you work with dependencies and permissions.

> **NOTE:** Extended Procedures are only available on the master database.

To edit an extended procedure

1   Open an editor on the extended procedure. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you provide a dynamic-link library (DLL) **Library Name** for the extended procedure.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

FOREIGN KEYS EDITOR (SQL SERVER)

The Foreign Keys Editor lets you manage the columns and basic properties of a foreign key.

To edit a foreign key

1   Open an editor on the foreign key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mapping	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes.
	If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you specify **Enabled** and **Not For Replication** properties. This tab also lets you select a **Delete rule** and an **Update rule** (NONE, SET NULL, SET DEFAULT, CASCADE).
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

FULL-TEXT CATALOGS EDITOR (SQL SERVER)

The Full-text Catalogs Editor lets you work with the basic definition and dependencies of a full-text catalog.

To edit a full-text catalog:

1   Open an editor on the full-text catalog. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Parent Type**, **Parent Schema**, **Parent Name**, **Key Index**, and **Full-text Catalog Name** properties for the partition scheme. It also lets you modify the **Filegroup Name** (SQL Server 2008 ^), **Change Tracking**, and **Stoplist** (SQL Server 2008 ^) properties. For more information on these properties, see Full-text Catalogs (SQL Server) - Properties.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

FULL-TEXT INDEXES EDITOR (SQL SERVER)

The Full-text Indexes Editor lets you work with the basic definition and dependencies of a full-text index.

To edit a full-text index:

1   Open an editor on the full-text index. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Name**, property and modify the **Accent Sensitive**, **Default**, and **Authorization Owner** properties. For more information on these properties, see Full-text Indexes (SQL Server) - Properties.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

FUNCTIONS EDITOR (SQL SERVER)

The Functions Editor lets you view and modify function definitions and dependencies.

To edit a function

1 Open an editor on the function. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you modify the owning **Schema** of the function.
Definition	Lets you view and modify the dynamic-link library (DLL) or view the data definition language (DDL).
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

INDEXES EDITOR (SQL SERVER)

The Indexes Editor lets you manage columns and basic properties of an index and view index statistics.

To edit an index

1 Open an editor on the index. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the index. On opening, this tab shows the existing columns. For each column, the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
		Change the **Sort** order of a column.
		Click the New button and select a column name from the dropdown, to add a column to the index.
		Select a column and click the Drop button to delete the column from the index.
		Select a column and use the arrow buttons to reorder the columns in the index.

Tab	Settings and tasks	
Properties	Lets you work with settings in the following categories:	
	Creation	In addition to displaying identifying information, you can modify the **Build Online** and **Max degree of parallelism** properties. For details on these properties, see <u>Index Wizard (SQL Server)</u>.
	Attributes	Lets you set **Index Type**, **Clustered**, **Ignore Duplicate Key** (for **Index Type** of UNIQUE), **Statistics Recompute**, **Allow Row Locks**, and **Allow Page Locks** properties. For details on these properties, see <u>Index Wizard (SQL Server)</u>.
		NOTE: You cannot reorganize an index (primary key, or unique key) that has an **Allow Page Locks** property set to FALSE. For information on reorganizing indexes, see <u>Reorganize (SQL Server indexes, primary keys, and unique keys)</u>.
	Storage	Lets you set **Partitioned**, **Partition Scheme**, **Parttition Column, Fill Factor**, **File Group**, **Pad Index**, and **Sort in Tempdb** settings.
Statistics	Lets you view statistics in the following categories:	
	Page Statistics	**Data Pages**, **Reserved Pages**, **Used Pages**, and **Total Pages Modified**.
	Row Statistics	**Maximum Row Size**, **Minimum Row Size**, **Max Size of Non-Leaf Index Row**, and **Total Rows Modified**.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

LOGINS EDITOR (SQL SERVER)

The Logins Editor lets you manage basic properties for a login, associated users and roles, and accounting.

To edit a login:

1 Open an editor on the login. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you view or modify the following properties set when creating the login: **Name**, **Default Database**, **Default Language**, **Password**, **Check Policy**, **Check Expiration**, **Must Change**, **Certificate**, and **Asymmetric Key**. For detailed information on these settings, see <u>Login Wizard (SQL Server)</u>. In addition, when editing an existing login, you have access to the following settings:	
	Currently Logged In	Displays whether the user is currently logged in.
	Enabled	Enabled, an ALTER LOGIN... ENABLE is submitted following the CREATE LOGIN statement issued by the editor. Disabled, an ALTER LOGIN... DISABLE is submitted
Server Roles	Lets you add the login as a member of one or more fixed server roles. For more information, see <u>Logins (SQL Server) - Server Roles</u>.	
Users	Lets you add and remove user accounts for this login to specified databases. For more detailed information, see <u>Logins (SQL Server) - Users</u>.	
DDL	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PARTITION FUNCTIONS EDITOR (SQL SERVER)

The Partition Functions Editor lets you view the basic definition and work with dependencies of a partition function.

To edit a partition function:

1 Open an editor on the partition function. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Name**, **Input Parameter Type**, **Range**, and **Function Values** properties for the partition function. For more information on these properties, see <u>Partition Functions (SQL Server) - Properties</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PARTITION SCHEMES EDITOR (SQL SERVER)

The Partition Schemes Editor lets you view the basic definition and work with dependencies of a partition scheme.

To edit a partition scheme:

1   Open an editor on the partition scheme. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Name**, **Partition Function**, and **Filegroup Name** properties for the partition scheme. For more information on these properties, see [Partition Schemes (SQL Server) - Properties](#).
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

PRIMARY KEYS EDITOR (SQL SERVER)

The Primary Keys Editor lets you manage the columns and basic properties for a primary key and lets you view page and row statistics.

To edit a primary key

1   Open an editor on the primary key. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the primary key. On opening, this tab shows the existing columns. For each column, the listing shows the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
		Change the **Sort** order of a column.
		Click the New button and select a column name from the dropdown, to add a column to the primary key.
		Select a column and click the Drop button to delete the column from the primary key.
		Select a column and use the arrow buttons to reorder the columns in the primary key.

Tab	Settings and tasks		
Properties	Lets you work with settings in the following categories:		
	Creation	In addition to displaying identifying information, you can modify the **Build Online** and **Max degree of parallelism** properties. For details on these properties, see [Primary Key Wizard (SQL Server)](#).	
	Attributes	**Clustered** - Indicates whether the target primary key is clustered.	
		Ignore Duplicate Key - Indicates whether the target primary key ignores duplicate key values. If you select his option, the transaction that generated the duplicate key values can continue.	
		Statistics Recompute - Indicates that index statistics are automatically recomputed as the index is updated. Microsoft does not recommend this.	
	Storage	**File Group** - Lets you select a file group.	
		Fill Factor - Lets you specify the fill factor that specifies how full each index page can be. If no fill factor is specified, Microsoft SQL Server uses the database's default fill factor.	
		Pad Index - If you specified a Fill factor of more than 0 percent, and you selected the option to create a unique index, you can specify to use the same percentage you specified in Fill Factor as the space to leave open on each interior node. By default, Microsoft SQL Server sets a 2 row index size.	
Statistics	Lets you view statistics in the following categories:		
	Page Statistics	**Data Pages**, **Reserved Pages**, **Used Pages**, and **Total Pages Modified**.	
	Row Statistics	**Maximum Row Size**, **Minimum Row Size**, **Max Size of Non-Leaf Index Row**, and **Total Rows Modified**.	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).		

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

PROCEDURES EDITOR (SQL SERVER)

The Procedures Editor lets you manage properties, the definition, dependencies, and permissions for a procedure.

To edit a procedure

1   Open an editor on the procedure. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you change the owning **Schema**.
Definition	Lets you view the SQL code for the procedure.
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).
Permissions	For details on using this tab, see [Working with Privileges and Permissions](#).

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

ROLES EDITOR (SQL SERVER)

The Roles Editor lets you manage the users for a role.

To edit a role

1   Open an editor on the role. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the **Name**, and either a **Role Type** of STANDARD and an **Authorization Owner** or a **Role Type** of APPLICATION and a **Password**.
Users	Let you manage users for the role. A user becomes associated with an application role after running the target application. Use the **Join Rule** and **Leave Role** buttons to move users between the **Users In Role** and **Users Not In Role** lists.
Object Privileges and **System Privileges**	For details on using these tabs, see <u>Working with Privileges and Permissions</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

RULES EDITOR (SQL SERVER)

The Rules Editor lets you view or modify basic properties of a rule.

To edit a rule

1   Open an editor on the rule. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you specify whether the rule is **Enabled** and view the **Fire On Insert**, **Fire On Update**, **Fire On Delete**, and **Encrypted** settings.
Definition	Lets you view and modify the SQL code that will implement any changes you make using this editor.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

SCHEMAS EDITOR (SQL SERVER)

The Schemas Editor lets you change the owner of a group and manage the objects contained in the schema.

To edit a user

1   Open an editor on the user. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you modify the **Owner** of the schema.	
Permissions	For details on using this tab, see [Working with Privileges and Permissions](#).	
Objects	Displays the specific objects owned by the schema, grouped under object type folders. Optionally, you can:	
		Select an object and click **Edit** to open an object editor on that object.
		Select an object and click **Drop** to initiate dropping the object.
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

SYNONYMS EDITOR (SQL SERVER)

The Synonyms Editor lets you view the basic definition and work with dependencies of a partition scheme.

To edit a synonym:

1   Open an editor on the synonym. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Schema**, **Name**, **Server**, **Database**, **Referenced Object Type**, **Referenced Object Owner**, and **Referenced Object Name** properties for the synonym. For more information on these properties, see [Synonyms (SQL Server) - Properties](#).
	This tab also includes a **CreateDate** property showing the data and time the synonym was created.

Tab	Settings and tasks
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3 When finished, you can submit your changes. For details, see <u>**Previewing and Submitting Object Editor Changes**</u>.

TABLES EDITOR (SQL SERVER)

The Tables Editor lets you manage columns, basic properties, indexes, and constraints for a table and view space usage details.

To edit a table

1 Open an editor on the table. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Displays the currently defined columns in the table. Optionally, you can:	
		Select a column, and in the **Property/Value** list modify property values for that column.
		Click **Add Column**, provide a name for the new column, and set property values for the column.
		Select a column and click **Delete** to remove the column from the table.
Properties	Lets you work with settings in the following categories:	
	Physical Storage	Lets you view the **Partitioned** and **Partition Scheme** properties. Lets you specify a **Filegroup** and **Text Image Filegroup**. This is only available for Microsoft SQL Server 7.0 and later.
	Full-Text Index	Lets you view whether Full-Text Indexing is installed and active.
		The full-text index feature provides support for sophisticated word searches in character string data. A full-text index stores information about significant words and their location within a given column. This information is used to quickly complete full-text queries that search for rows with particular words or combinations of words.
		This feature is available for Microsoft SQL Server 8.0 or later.
	Text In Row	Lets you enable Text In Row Data and specify a limit.

Tab	Settings and tasks		
Indexes	Displays any indexes currently defined for the table. Optionally, you can:		
		Click **Add** to open an Index editor, letting you create a new index for this table. For more information, see <u>Index Wizard (SQL Server)</u>.	
		Select an index and click **Edit** to open an wizard that lets you modify that index. For more information, see <u>Indexes Editor (SQL Server)</u>.	
		Select an index and click **Drop** to open a dialog that lets you confirm that you want to drop the index. When dropping an index with the Clustered option enabled, the confirmation dialog includes an **Online** option that lets you specify an online drop (ONLINE-ON clause).	
Constraints	Lets you manage constraints for the table. Constraints are grouped by type, under folders. Optionally take one of the following actions:		
		Select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type.	
		Select a constraint and click **Edit** to open a dialog that lets you modify the constraint details.	
		Select a constraint and click **Drop** to open a dialog that lets you confirm that you want to drop the index. When dropping a clustered unique key or clustered primary key, the confirmation dialog includes an **Online** option that lets you specify an online drop (ONLINE-ON clause).	
Space	Lets you view pie charts showing the space usage for the table. Optionally you can double-click a slice in the pie chart for detailed statistics.		
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.		
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.		
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.		

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

TRIGGERS EDITOR (SQL SERVER)

The Triggers Editor lets you enable and disable a trigger, view and modify the CREATE TRIGGER statement that will implement changes, and manage dependencies.

To edit a trigger

1   Open an editor on the trigger. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you set the **Enabled** property. It also lets you view the **Trigger Timing** (AFTER, INSTEAD OF), **Fire On Insert**, **Fire On Update**, **Fire On Delete**, and **Encrypted** properties.

Tab	Settings and tasks
Definition	Lets you modify the CREATE TRIGGER body for a trigger. To modify a trigger, edit the text of the trigger body in the Trigger Text area. Rapid SQL must drop then create the trigger.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.

3   When finished, you can submit your changes. For details, see <u>**Previewing and Submitting Object Editor Changes**</u>.

UNIQUE KEYS EDITOR (SQL SERVER)

The Unique Keys Editor lets you manage columns and properties for a unique key and view associated statistics.

To edit a unique key

1   Open an editor on the unique key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the unique key. On opening, this tab shows the existing columns. For each column, the listing displays the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
		Change the **Sort** order of a column.
		Click the New button and select a column name from the dropdown, to add a column to the index.
		Select a column and click the Drop button to delete the column from the index.
		Select a column and use the arrow buttons to reorder the columns in the index.

Tab	Settings and tasks	
Properties	Lets you work with settings in the following categories:	
	Creation	In addition to displaying identifying information, you can modify the **Build Online** and **Max degree of parallelism** properties. For details on these properties, see [Unique Key Wizard (SQL Server)]().
	Attributes	**Clustered** - Indicates whether the target index is clustered.
		Ignore Duplicate Key - Indicates whether the target primary key ignores duplicate key values. If you select his option, the transaction that generated the duplicate key values can continue.
		Statistics Recompute - Indicates that index statistics are automatically recomputed as the index is updated. Microsoft does not recommend this.
	Storage	**File Group** - Lets you specify the filegroup on which to place the index. This is for Microsoft SQL Server 7.0 or later.
		Fill Factor - Lets you specify a percentage that indicates how full Microsoft SQL Server should make the leaf level of each index page during index creation. When an index page fills up, Microsoft SQL Server must take time to split the index page to make room for new rows, which is quite expensive. For update-intensive tables, a properly chosen Fill factor value yields better update performance than an improper Fill factor value.
		Pad Index - If you specified a Fill factor of more than 0 percent, and you selected the option to create a unique index, you can specify to use the same percentage you specified in Fill Factor as the space to leave open on each interior node. By default, Microsoft SQL Server sets a 2 row index size.
Statistics	Displays statistics in the following categories:	
	Page Statistics	**Data Pages**, **Pages Reserved**, **Used Pages**, and **Total Pages Modified**.
	Row Statistics	**Maximum Row Size**, **Minimum Row Size**, **Max Size of Non-Leaf Index Row**, and **Total Rows Modified Since Last**.
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object]().	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes]().

USERS EDITOR (SQL SERVER)

The Users Editor lets you manage properties, assign roles, and manage object bindings for a user.

To edit a user

1   Open an editor on the user. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Without Login**, **Login Name**, **Name**, **User Type**, **Certificate**, **Asymmetric Key**, and **Default Schema** properties.
Roles	Lets you assign roles to a user by selecting the check boxes associated with the roles to be assigned.
Objects	Lets you manage database objects (Defaults, Indexes, Procedures, Rules, Tables, Triggers, User Datatypes, Views) associated with the user. Objects are organized in a tree structure with folders containing the objects. Optionally, take one of the following actions:
	Select an object and click **Edit** to open an object editor on the selected object. For information on each editor, use the links in Microsoft SQL Server Object Editors.
	Select an object and click **Drop** to initiate dropping the selected object.
Object Permissions and **System Permissions**	For details on using these tabs, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

USER DATATYPES EDITOR (SQL SERVER)

The User Datatypes Editor lets you modify the base datatype, rule and default bindings, and referencing or referenced objects for a user datatype.

To edit a user datatype

1   Open an editor on the user datatype. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	In addition to letting you change the **Schema**, this tab lets you work with settings in the following categories:	
	Base Datatype	Lets you modify the **Type** and **Allow Nulls** properties. Depending on the **Type** selected, **Precision**, **Scale**, or **Size** properties are available as appropriate.
	Bindings	**Default Binding** - Lets you select a default. For information on creating defaults, see Default Wizard (SQL Server).
		Rule Binding - Lets you select a rule. For information on creating rules, see Rule Wizard (SQL Server).

Tab	Settings and tasks	
Usage	Lets you manage database objects referencing or referenced by a user datatype. Objects are grouped within object type folders. Optionally, you can:	
	Select an object and click **Edit** to open an object editor on the selected object.	
	Select an object and click **Drop** to initiate dropping the selected object.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

USER MESSAGES EDITOR (SQL SERVER)

NOTE: User Messages are only on the master database.

The User Messages Editor lets you view or modify basic properties of a user message. It also lets you add, edit, or delete individual language versions of the message.

To edit a user message

1   Open an editor on the user message. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with the following properties:	
	Message Number	Displays the message number chosen when the user message was created.
	Severity	Lets you modify the SQL Server error severity level (001-025).
	Write to NT Event Log	Lets you specify that the message is always written to the Windows NT Event Log.
Information	Displays the currently-defined language versions for this message number. Optionally you can:	
	Create a new language version for this message number. Click the **Add new text for the user message** button, and in the dialog box that opens, select a **Language** and provide the **Message Text**.	
	Select a language version of the message and click the **Modify user message text** button. This opens a dialog that lets you modify the **Language** or **Message Text**.	
	Select a language version of the message and click the **Remove user message text** button.	
	NOTE: You cannot create two versions of a message for the same language.	
	NOTE: FOr a given message number there should always be a least one, us_english version of the message.	

Tab	Settings and tasks
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

VIEWS EDITOR (SQL SERVER)

The Views Editor lets you view columns and change the schema of a view.

To edit a view

1   Open an editor on the view. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays a list of the columns that make up the view. Details include the column **Name**, its **Datatype**, and whether the definition allows **Null**s. This tab also lets you change the **Schema** owning the view.
Definition	Lets you view the SQL for the View.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

MYSQL EDITORS

Rapid SQL includes an Object Editor for all supported MySQL objects. To see an Editor for a specific object, click the corresponding link below:

- <u>Foreign Keys editor (MySQL)</u>

- <u>Functions editor (MySQL)</u>

- <u>Indexes, Primary Keys, and Unique Keys editors (MySQL)</u>

- <u>Tables editor (MySQL)</u>

- <u>Users editor (MySQL)</u>

FOREIGN KEYS EDITOR (MYSQL)

The Foreign Keys Editor lets you manage column mapping, delete, and update actions for a foreign key.

To edit a foreign key:

1   Open an editor on the foreign key. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mapping	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes.
	If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you modify the **On Delete** and **On Update** (NO ACTION, RESTRICT, CASCADE, SET NULL) reference options.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

FUNCTIONS EDITOR (MYSQL)

The MySQL Functions editor lets you view properties of a basic function declaration.

To edit a function:

1   Open an editor on the function. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Function Name**, **Return Type**, and **Shared Object Library** properties. For details on these properties, see Functions - Properties.
	Under **Specify Columns** in Index, you can change the selection of columns that make up the index, primary key, or unique key.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

INDEXES, PRIMARY KEYS, AND UNIQUE KEYS EDITORS (MYSQL)

These editors lets you manage columns and view properties of an index, primary key, or unique key.

To edit an index:

1 Open an editor on the index. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the **Table Name**, **Index Name**, **Constraint Type**, and **Index Type** properties. For details on these properties, see [Indexes, Primary Keys, or Unique Keys - Properties](#). Under **Specify Columns** in Index, you can change the selection of columns that make up the index, primary key, or unique key.
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

TABLES EDITOR (MYSQL)

The MySQL Tables editor lets you modify basic properties and view usage statistics for a MySQL table.

To edit a table:

1 Open an editor on the index. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Columns	Displays the currently defined columns in the table. Optionally, you can:
	Select a column, and in the **Property**/**Value** list modify property values for that column.
	Click **Add Column**, provide a name for the new column, and set property values for the column.
	Select a column and click **Delete** to remove the column from the table.
Properties	Lets you work with the following settings: **Table Name**, **Storage Type**, **Row Forma**t, **Default Character Set**, **Default Collation**, **Auto Increment**, **Comment**, **Min Rows**, **Max Rows**, **Average Row Length**, **Pack Keys**, **Checksum**, and **Delay Key Write**. For details on these properties, see [Tables - Properties](#).
Indexes	Lets you add, delete, or modify table indexes. For details on available options, see [Tables - Indexes](#).

Tab	Settings and tasks
Foreign Keys	For INNODB tables, this tab lets you add, delete, or modify table foreign keys. For details on available options, see <u>Tables - Foreign Keys</u>.
MERGE tables	For MRG_MyISAM tables, this tab lets you work with the collection of identical MyISAM tables that are to be used as a single table. For more information, see <u>Tables - MERGE Tables</u>.
Privileges	For details on using this tab, see <u>Working with Privileges and Permissions</u>.
Space	Lets you view graphical and numerical table usage statistics. Values include **Data Length**, **Index Length**, **Data Free**, **Max Data Length**, **Rows**, and **Average Row Length**.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

USERS EDITOR (MYSQL)

The MySQL Functions editor lets you work with basic properties, host details, and privileges for a user.

To edit a user:

1   Open an editor on the user. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
User Information	Lets you view the **User Name** property and lets you modify the **Full Name**, **Description**, **Email**, and **Contact Information** properties. For details on these properties, see <u>Users wizard (MySQL)</u>.
User Hosts	Add a new host to the list by clicking the New button. Use the associated dropdown to select a valid host-name value (such as localhost or %), overwriting the text as necessary to provide a specific quoted identifier. Provide a password in the **Password** and **Confirm** fields and click **Apply**. Remove a selected host from the list by clicking the Delete button.
Object Privileges and **System Privileges**	For details on using these tabs, see <u>Working with Privileges and Permissions</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

ORACLE OBJECT EDITORS

Rapid SQL includes an Object Editor for all supported Oracle objects. To see an Editor for a specific object, click the corresponding link below.

> **NOTE:** If an objects has dependent objects, such as tables, triggers, procedures and views, you can view and access their dependent objects in the editor.

- Check Constraints Editor (Oracle)

- Clusters Editor (Oracle)

- Database Links Editor (Oracle)

- Directories Editor (Oracle)

- Foreign Keys Editor (Oracle)

- Functions Editor (Oracle)

- Indexes Editor (Oracle)

- Job Queue Editor (Oracle)

- Libraries Editor (Oracle)

- Materialized Views Editor (Oracle)

- Materialized View Logs Editor (Oracle)

- Outlines Editor (Oracle)

- Package Bodies Editor (Oracle)

- Packages Editor (Oracle)

- Primary Keys Editor (Oracle)

- Procedures Editor (Oracle)

- Profiles Editor (Oracle)

- Redo Log Groups Editor (Oracle)

- Roles Editor (Oracle)

- Rollback Segments Editor (Oracle)

- Sequences Editor (Oracle)

- Synonyms Editor (Oracle)

- Tables Editor (Oracle)

- Tablespaces Editor (Oracle)

- Triggers Editor (Oracle)

- [Type Bodies Editor (Oracle)](#)

- [Types Editor (Oracle)](#)

- [Unique Keys Editor (Oracle)](#)

- [Users Editor (Oracle)](#)

- [Views Editor (Oracle)](#)

CHECK CONSTRAINTS EDITOR (ORACLE)

The Check Constraints Editor lets you modify and enable/disable check constraints.

To edit a check constraint

1   Open an editor on the check constraint. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Use the **Enabled** control to enable/disable the check constraint.
	You can also edit the condition in the **Check Condition** box. The **Table Columns** button acts a time saver in editing the condition. It opens a dialog that lets you select and paste column names into the check condition expression.
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

CLUSTERS EDITOR (ORACLE)

The Clusters Editor lets you manage cluster column and table details, view and modify storage and space parameters, and manage performance settings for a cluster.

To edit a cluster

1 Open an editor on the cluster. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Use the controls in the **General Properties** group to specify a hash or index **Cluster Type** and to specify a **Key Size** for the cluster.	
	If you specify a hash cluster type, use the controls in the **Hash Specifications** group to specify the number of hash keys and to specify use of either the default hash function or a specified expression used as the hash function.	
Storage	Lets you manage storage for a cluster:	
	Data Block Storage	Each transaction that updates a data block requires a transaction entry.
		Percent Free - minimum percentage of free space in a block
		Percent Used - minimum percentage of used space in a block.
		Initial transactions - The initial parameter ensures that a minimum number of concurrent transactions can update a data block, avoiding the overhead of allocating a transaction entry dynamically.
		Maximum transactions - The maximum parameter limits concurrency on a data block.
	Extents	The unit of space allocated to an object whenever the object needs more space.
		Initial Extent - The initial space extent (in bytes) allocated to the object.
		Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
		Minimum Extents - The appropriate minimum extents value for the object.
		Maximum Extents - The appropriate maximum extents value for the object.
		Percent Increase - Magnifies how an object grows and, can materially affect available free space in a tablespace. Select a value in the corresponding box.
	Optionally, you can modify the **Percent Free**, **Percent Used**, or **Max Transactions** values.	
Performance	Lets you modify the following performance settings:	
	Parallel Query Option group	Lets you specify **Degrees** and **Instances** settings for processing queries using many query server processes running against multiple CPUs, which provides substantial performance gains such as reduction of the query completion time.
	Cache group	The **Cache** setting keeps the blocks in memory by placing it at the most recently used end. This option is useful for small lookup tables.
Space	Lets you view the following usage and the space distribution details:	
	Space Utilization group	Displays the percent of space reserved for future updates.
	FreeLists group	Displays the allocation of data blocks when concurrent processes are issued against the cluster. Identifying multiple freelists can reduce contention for freelists when concurrent inserts take place and potentially improve the performance of the cluster.
	Extents group	The unit of space allocated to an object whenever the object needs more space.

Tab	Settings and tasks
Tables	Lets you view details for each cluster column. Details for each column include the key column in the cluster, the clustered table name, and the key column in the table.
DDL	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

DATABASE LINKS EDITOR (ORACLE)

The Database Links Editor lets you view connection string information for a database link.

To edit a database link

1   Open an editor on the database link. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Displays connection string information for a database link. Details include the **Public** setting, the **User** name and **Password** associated with the database link, and a **Connect String**.
DDL	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

DIRECTORIES EDITOR (ORACLE)

The Directories Editor lets you change the path for a directory and modify associated privileges.

To edit a directory object

1   Open an editor on the directory. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you view or modify the SQL code for the directory.
Privileges	For details on using this tab, see Working with Privileges and Permissions.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

FOREIGN KEYS EDITOR (ORACLE)

The Foreign Keys Editor lets you enable a foreign key, manage its column mappings, and specify a delete rule.

To edit a foreign key

1 Open an editor on the foreign key. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mapping	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes.
	If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you set the foreign key as **Enabled** and specify a **Delete Rule** of CASCADE or NO ACTION.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

FUNCTIONS EDITOR (ORACLE)

The Functions Editor lets you modify basic properties and modify the CREATE/REPLACE DDL for a function.

To edit a function

1 Open an editor on the function. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you view properties in the following categories:	
	Properties	Lets you view **Status** and **Last Modified** details for the function
	Size Information	Lets you view **Source SIze**, **Parsed Size**, **Code Size**, and **Error Size** for the function
Definition	Lets you modify the CREATE OR REPLACE FUNCTION DDL for the function.	
Dependencies	For details on using this tab, see Working with Object Dependencies.	
Permissions	For details on using this tab, see Working with Privileges and Permissions.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

INDEXES EDITOR (ORACLE)

The Indexes Editor lets you manage columns, basic properties, storage, space, and partitioning for an index.

To edit an index

1   Open an editor on the index. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab		Settings and tasks
Columns	Displays a listing of the columns making up the index. Optionally, you can:	
		Use the **Sort** column check box to specify a sort option.
		Click the New button to add a new column to the index.
		Select an existing column and click the Delete button to delete that column from the index.
Properties	Lets you work with settings in the following categories:	
	Attributes	Lets you view **IsValid** and **Function-Based** properties. Lets you set **Index Type** (UNIQUE, NONUNIQUE, BITMAP), **No Sort**, **Logging**, **Reverse**, and **Invisible** properties.
	Parallel Query Option	Lets you view the **No Parallel Execution** property. Lets you set the **Parallel Degree** and **Parallel Instances** values.
Storage	Lets you work with settings in the following categories:	
	Data Block Storage	Lets you choose the **Tablespace**, and specify **Percent Free**, **Initial Transactions**, and **Max Transactions values**.
	Extents	Lets you specify **Initial Extent**, **Next Extent**, **Minimum Extents**, **Maximum Extents**, and **Percent Increase** values.
	Freelists	Lets you specify **Freelists** and **Freelist Groups** values.
	Buffer Pool	Lets you specify a buffer pool.
Space	Lets you view settings in the following categories:	
	Space Utilization	Lets you view the **Size** and number of **Blocks** settings.
	Statistics	Lets you view **Index Level**, **Distinct Keys**, **Cluster Factor**, **Leaf Blocks**, **Leaf Blks/Key** and **Data Blks/Key** properties.

Tab		Settings and tasks
Partition		If the index is not currently partitioned, you can click the **Convert To Partitioned** button to partition the index. For more information, see Partitioning Oracle indexes, primary keys, and unique keys. If the index is currently partitioned, this tab displays the following partition details:
	Properties	Lets you view the **Locality** (Global/Local), **Alignment** (Prefixed/Non-Prefixed), **Partition Type** (RANGE or HASH), and **Subpartition type** properties.
		Click the Edit Partition button to edit partition details. or more information, see Partitioning Oracle indexes, primary keys, and unique keys.
		Click the Drop Partition button to revert to an unpartitioned index.
		Use the following **Partition Commands** menu items: Allocate Extent Analyze Coalesce Deallocate Unused Space **Mark Unusable** - opens a dialog that lets you select one or more partitions/subpartitions to be marked as unusable. **Rebuild** - opens a dialog that lets you select one or more unusable subpartitions to be rebuilt. **Split** - opens a dialog that lets you divide a single partition into two partitions. You can split partitions if a single partition is causing maintenance problems because it is too large. **NOTES:** If you are preparing to drop or rebuild an index, mark local indexes as unusable. If you want to make unusable indexes valid or to recover space and improve performance, rebuild the unusable indexes. You cannot split a local index partition defined on a hash or composite table. Make sure that you specify an upper bound for the column that is lower than the upper bound for that column in the original partition.
	Columns	Displays partitioning columns.
	Partition Definitions	Displays the list of partition definitions. For each partition definition, listing shows the partition definition **Value**, the associated **Tablespace**, and whether the index is **Usable** or has been marked as Unusable by Oracle. Use the **Partition Commands** menu **Mark Unusable** or **Rebuild** commands to change the current **Usable** value.
DDL View		For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

JOB QUEUE EDITOR (ORACLE)

The Job Queue lets you modify the job's definition (code) and manage the schedule and status for a job.

To edit a job queue

1   Open an editor on the job queue. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you modify the statement, function call, or procedure call in the **Job Definition** area.
Properties	Use the settings in the **Schedule** group to specify that the job is to run once only or is to run according to a schedule. If it is to run on a schedule, specify the interval in days or using a custom date expression.
	Use the settings in the **Status** group to **Enable** or **Disable** the job.
DDL	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

LIBRARIES EDITOR (ORACLE)

The Libraries Editor lets you view and modify library definitions and manage dependencies and privileges for the library.

To edit a library

1   Open an editor on the library. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	The **Definition** tab of the Libraries Editor lets you modify the file name and path for a library and view the current status and whether the library is dynamic.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Privileges	For details on using this tab, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

MATERIALIZED VIEWS EDITOR (ORACLE)

The Materialized Views Editor lets you view and modify materialized view information and partitions.

To edit a materialized view

1 Open an editor on the materialized view. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Information	Lets you work with properties in the following categories:	
	Master	Lets you work with the **Table Name** and **Master View** properties.
	Last Refresh	Lets you work with the **Last Date** and **Errors** properties.
	Refresh Configuration	Lets you work with the **Type**, **Refresh Method**, and **Mode** properties.
	Rollback Usage	Lets you work with the Local and Master properties.
	Options	Lets you work with the **Updatable** and **Enable Query Rewrite** properties.
Storage	Lets you work with properties in the following categories:	
	Placement	Lets you work with **Tablespace** and **Cluster** values.
	Data Block Storage	Lets you work with **Percent Free**, **Percent Used**, **Initial transactions**, and **Maximum transactions** values.
	Extents	Lets you work with **Initial Extent**, **Next Extent, Minimum Extents**, **Maximum Extents**, and **Percent Increase** values.
Performance	Lets you work with settings in the following categories:	
	Parallel Query Option	The Parallel server query option lets you process queries using many query server processes running against multiple CPUs. This option provides substantial performance gains such as reduction of the query completion time.
		Degrees - Lets you type a value indicating the number of query server processes that should be used in the operation.
		Instances - Lets you type a value indicating how you want the parallel query partitioned between the Parallel Servers.
	Logging	Select **Logging** to create a log for all Materialized View updates.
	Cache	Select **Cache** if you want Oracle to put data you access frequently at the most recently used end of the list in the buffer cache when a full table scan is performed. This option is useful for small lookup tables.
Query	Displays the associated query.	
Partitions	**Partitioning Method**	Displays the partitioning method, including Range-Hash Composite or Range-List Composite.
		Hash partitions partition the table according to a hash function. Composite partitions use both range and hash types, first partitioning the data by a range of values, and then further dividing the partitions into subpartitions by way of a hash function.
		List partitioning lets you control how rows map to partitions.
		You can specify a list of discrete values for the partitioning column in the description for each partition.
	Row Movement	If its key is updated, migrates the row to a new partition.

Tab	Settings and tasks	
	Partitioning Columns	Displays partitioning columns.
	Subpartitioning Columns	Displays subpartitioning columns.
	Partitions	Click Add or Edit to open the Partition dialog box. Click Drop to drop a partition.
	Subpartition Template	If the partitioning type is Range-Hash Composite, displays a list of subpartitions in the subpartition template. Click Add, Insert, or Edit to open the Subpartition dialog box. Click Drop to drop a subpartition.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.	
Privileges	For details on using this tab, see <u>Working with Privileges and Permissions</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3　When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

MATERIALIZED VIEW LOGS EDITOR (ORACLE)

The Materialized View Logs Editor lets you view replication log table details, and manage space, storage, and performance details for a materialized view log.

To edit a materialized view log

1　Open an editor on the materialized view log. For details, see <u>Opening an Object Editor</u>.

2　Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Information	Displays information on the log table used for replication.	
Storage	Lets you work with settings in the following categories:	
	Data Block Storage	Lets you view the associated **Tablespace** and **Initial Transactions** value, and modify the **Percent Free**, **Percent Used**, and **Max Transactions** values.
	Extents	Lets you view **Initial Extent**, **Next Extent**, **Minimum Extents**, **Maximum Extents**, and **Percent Increase** values.
	Column Filtering	Lets you select the filter columns to be recorded in the materialized view log. You can specify only one primary key, one ROWID and one filter column list per materialized view log. The ROWID is a globally unique identifier for a row in a database. It is created at the time the row is inserted into a table, and destroyed when it is removed from a table.

Tab	Settings and tasks	
Performance	Lets you work with settings in the following categories:	
	Parallel Query Option	The Parallel server query option lets you process queries using many query server processes running against multiple CPUs. This option provides substantial performance gains such as reduction of the query completion time. **Degrees** - Lets you type a value indicating the number of query server processes that should be used in the operation. **Instances** - Lets you type a value indicating how you want the parallel query partitioned between the Parallel Servers.
	Logging	Select **Logging** to create a log for all Materialized View updates.
	Cache	Select **Cache** if you want Oracle to put data you access frequently at the most recently used end of the list in the buffer cache when a full table scan is performed. This option is useful for small lookup tables.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

OUTLINES EDITOR (ORACLE)

The Outlines Editor lets you view information on an outline, and modify its category and associated SQL statement.

> **NOTE:** The only SQL statements possible with stored outlines are SELECT, DELETE, UPDATE, INSERT…SELECT, and CREATE TABLE…AS SELECT.

To edit an outline

1 Open an editor on the outline. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying basic creation and identification properties, this tab lets you select a new **Category** for the outline.
Definition	Lets you view and modify the SQL Statement associated with the outline.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PACKAGE BODIES EDITOR (ORACLE)

While listed as separate objects in the Datasource Explorer, package bodies are created and edited on the **Body** tab of the Packages Editor. For details, see Packages Editor (Oracle).

PACKAGES EDITOR (ORACLE)

The Packages Editor lets you view and modify header and body specifications of a package.

To edit package

1   Open an editor on the package. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Header	Lets you modify the package header specifications.	
Body	Lets you modify the package body specifications.	
Information	Lets you work with status and size properties in the following categories:	
	Header and **Body**	Lets you view **Status**, **Created**, and **Last Modified Source SIze**, **Parsed Size**, **Code Size**, and **Error Size** details for the header and body of the package.
Dependencies	For details on using this tab, see Working with Object Dependencies.	
Permissions	For details on using this tab, see Working with Privileges and Permissions.	

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

PRIMARY KEYS EDITOR (ORACLE)

The Primary Keys Editor lets you manage columns, basic properties, storage and space, and partitions for a primary key.

To edit primary key

1   Open an editor on the primary key. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab		Settings and tasks
Columns		Displays a listing of the columns making up the primary key. Optionally, you can:
		Click the New button to add a new column to the index.
		Select an existing column and click the Delete button to delete that column from the index.

Tab		Settings and tasks
Properties		Lets you work with settings in the following categories:
	Enforcing Index	Lets you view **User Defined**, **Index Owner**, and **Index Name** properties.
	Attributes	Lets you work with **No Sort** (only available if **Enabled** is set), **Logging** (YES, NO, or NONE), **Reverse** (disabled if **No Sort** is enabled), **Validate**, **Deferrable**, **Deferred** (IMMEDIATE or DEFERRED and only enabled if **Deferrable** is enabled), **Enabled**, **Cascade** (disabled if **Enabled** is set), **Rely** and **Update Date** properties.
Storage		Lets you work with settings in the following categories:
	Data Block Storage	Lets you specify **Tablespace**, **Percent Free**, **Initial Transactions**, and **Max Transactions** values. NOTE: You should never place primary keys on the SYSTEM tablespace.
	Extents	Displays **Initial Extent**, **Next Extent**, **Percent Increase**, **Minimum Extents** and **Maximum Extents** values.
	Freelists	Lets you specify **Freelists** and **Freelist Groups** values.
	Buffer Pool	Lets you specify a buffer pool.
Space		Lets you work with settings in the following categories:
	Space Utilization	Lets you view **Size** and **Blocks** properties.
	Statistics	Lets you view **Index Level**, **Distinct Keys**, **Cluster Factor**, **Leaf Blocks**, **Leaf Blks/Key**, and **Data Blks/Key** properties.
Partition		If the primary key is currently partitioned, this tab displays the following partition details:
	Properties	Lets you view the **Locality** (Global/Local), **Alignment** (Prefixed/Non-Prefixed), **Partition Type** (RANGE or HASH), and **Subpartition type** properties.
		Click the Edit Partition button to edit partition details. or more information, see [Partitioning Oracle indexes, primary keys, and unique keys](#).
		Click the Drop Partition button to revert to an unpartitioned primary key.
		Use the following **Partition Commands** menu items: [Allocate Extent](#) [Analyze](#) [Coalesce](#) [Deallocate Unused Space](#) **Mark Unusable** - opens a dialog that lets you select one or more partitions to be marked as unusable. **Rebuild** - opens a dialog that lets you select a partition to be rebuilt. **Split** - opens a dialog that lets you divide a single partition into two partitions. You can split partitions if a single partition is causing maintenance problems because it is too large.
	Columns	Displays partitioning columns.
	Partition Definitions	Displays details for each partition.
		If the primary key is not currently partitioned, you can click the **Convert To Partitioned** button to partition the primary key. For more information, see [Partitioning Oracle indexes, primary keys, and unique keys](#).

Tab	Settings and tasks
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PROCEDURES EDITOR (ORACLE)

The Procedures Editor lets you view and modify the SQL code and properties of a procedure.

To edit a procedure

1   Open an editor on the procedure. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with properties in the following categories:	
	Properties	Lets you view **Status**, and **Last Modified** properties.
	Size Information	Lets you view **Source Size**, **Parsed Size**, **Code Size**, and **Error Size**
Definition	Lets you modify the SQL code for a procedure.	
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.	
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PROFILES EDITOR (ORACLE)

The Profiles Editor lets you manage limits and manage user assignments for the profile.

To edit a profile

1 Open an editor on the profile. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Resources	Lets you work with settings in the following categories:	
	General Limits	Lets you specify **Composite Limit** and **Private SGA** settings.
	Session Limits	Lets you specify the limit on the amount of private space a session can allocate in the shared pool of the SGA. Specific settings are **Sessions Per User**, **CPU Per Session**, **Logical Reads**.
	Time Limits	Lets you specify the limit on total connection time per session. Specific settings are **Connect Time** and **Idle Time**.
	Call Limits	Lets you specify the CPU time limit for a call (a parse, execute, or fetch), expressed in hundredths of seconds. Specific settings are **CPU Per Call** and **Logical Reads**.
	Login Limits	Lets you specify the number of **Failed Login Attempts** on the user account before the account is locked and the **Account Lock Time**.
	Password Limits	Lets you specify the **Lifetime**, **Reuse Time**, **Reuse Max**, **Grace Period** and a **Verify Function** for passwords.
Users	Use the **Assign** button to open a dialog that lets you assign a user to this profile or select a user from the list, and click the **Unassign** button to open a dialog prompting you to confirm that the user is to be unassigned.	
DDL	For details on using this tab, see Viewing the SQL/DDL for an Object.	

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

REDO LOG GROUPS EDITOR (ORACLE)

The Redo Log Groups Editor lets you manage the members in a redo log group.

To edit a redo log group

1 Open an editor on the redo log group. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Redo Log Members	Lets you add new members to the redo log group, edit existing members, and delete members from the redo log group.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

ROLES EDITOR (ORACLE)

The Roles Editor lets you manage authentication and grant/revoke profiles for users and logins.

To edit a role

1   Open an editor on the role. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Authentication	Lets you manage role identity. When creating a role, you must establish certain rules governing its use. You can specify whether or not a role must be identified when used. If you require role identification, you can authenticate the user externally through the operating system, or with a specific password. If you specified that the role requires identification, provide a **Password** and specify whether the role is to be authenticated **Globally** or **Externally**.
User/Roles	Displays permissions for this role to logins or users. Click **Grant** to open a dialog that lets you grant this role to a login or another role. Select a role or login and click **Revoke** to revoke the role or login.
Object Privileges and **System Privileges**	For details on using this tab, see [Working with Privileges and Permissions](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

For information on activating and deactivating roles for the current login in the current session, see [Activating/Deactivating Roles in the Current Session](#).

ROLLBACK SEGMENTS EDITOR (ORACLE)

The Rollback Segments Editor lets you view rollback segment status, manage rollback segment storage, and view activity levels.

To edit a rollback segment

1   Open an editor on the rollback segment. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Status	Lets you enable and disable a rollback segment and displays status details for the rollback segment. The tab displays whether the rollback segment is online or offline and provides the associated **Tablespace**, **Size**, and **No. of Extents**.	
Storage	Lets you work with settings in the following categories:	
	Extents	The unit of space allocated to an object whenever the object needs more space.
		Initial Extent - The initial space extent (in bytes) allocated to the object.
		Next Extent - The next extent (in bytes) that the object will attempt to allocate when more space for the object is required.
		Optimal Size - optimal extent size
		Minimum Extents - The appropriate minimum extents value for the object.
		Maximum Extents - The appropriate maximum extents value for the object.
	Extent Detail	Displays extent details.
Activity	Lets you work with settings in the following categories:	
	Activity Levels	Displays **Active Transactions**, **Writes**, **Gets** and **Waits** values.
	Dynamic Sizing	Displays **High Watermark**, **Extends**, **Shrink**s, and **Wraps** values.
DDL	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

SEQUENCES EDITOR (ORACLE)

The Sequences Editor lets you manage parameters for a sequence, manage database objects dependent on the sequence, and manage privileges for the sequence.

To edit a sequence

1   Open an editor on the sequence. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Definition	Lets you work with settings in the following categories:	
	Parameters	Lets you specify increment, minimum value and maximum value settings.
	Current/Next Sequence Numbers	Lets you work with sequence cycle numbers.
	Options	Lets you specify **Cache Size**, **Cycle When Reach Max/Min**, and **Generate Numbers in Order** (useful when you are using the sequence number as a timestamp) values.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.	
Privileges	For details on using this tab, see <u>Working with Privileges and Permissions</u>.	
DDL	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

SYNONYMS EDITOR (ORACLE)

The Synonyms Editor lets you view base object information and manage database objects dependent on a synonym.

To edit a synonym

1   Open an editor on the synonym. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays the type, owner, name, and other details of the object referenced by the synonym.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

TABLES EDITOR (ORACLE)

The Tables Editor lets you manage columns, constraints, storage and space, and partitions for a table.

To edit a table

1   Open an editor on the table. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks		
Columns	Displays the currently defined columns in the table. For any selected column, the Property/Value list provides additional detail on that column. Available properties depend on the datatype you choose as well as on the property values you select:		
	Virtual - indicates whether this column is defined as an Oracle virtual column. If selected, the value is provided by the calculation in the **Default Value** box.		
	Datatype properties- Lets you select a **type** and depending on your selection, additional properties such as **Scale**, **Width**, and **Unused** may be available.		
	Allow Nulls - Select this check box to allow nulls in this column.		
	Encryption properties include **Password**, **Salted**, and **EncryptionAlgorithm**.		
	Default Value - If **Virtual** is selected, this box lets you specify a valid Oracle column expression. Otherwise, it can contain a pseudocolumn (CURRENT_TIMESTAMP, USER, SYSDATE, or UID) or an expression.		
	lets you choose among CURRENT_TIMESTAMP, USER, SYSDATE, and UID.		
	Comment lets you add a comment to the column.		
	LOB Storage settings are available for bfile, blob, clob, and nclob types. They include **Segment Name**, **Configuration** properties (**Tablespace**, **Chunk**, **Percent Version**, **Enable Storage In Row**, **Cache**, and **Logging**), and **Storage** properties (**Initial Extent**, **Next Extent**, **Percent Increase**, **Minimum Extents**, **Maximum Extents**, **Free Lists**, and Free List Groups).		
	Optionally, you can:		
		Select a column, and in the **Property**/**Value** list modify property values for that column.	
		Click **Add Column**, provide a name for the new column, and set property values for the column.	
		Select a column and click **Delete** to remove the column from the table.	
Properties	Lets you work with settings in the following categories:		
	Table	Lets you work with the following properties, corresponding to CREATE TABLE clauses: **Cache**, **Row Movement**, **Parallel Degree**, and **Parallel Instances**.	
	Physical	Lets you view the **Row Organization** property. Lets you set the **Logging** and **Table Compression** properties.	

Tab	Settings and tasks	
Indexes	Lets you manage indexes for a table. On opening, the list of current indexes for the table is displayed. Optionally, take one of the following actions:	
		Click **Add** to open a dialog that lets you add a new index to the table.
		Select an index and click **Edit** to open a dialog that lets you edit index properties.
		Select an index and click **Drop** to open a dialog that lets you remove the index from the table.
Constraints	Lets you manage constraints for the table. Constraints are grouped by type, under folders. Optionally take one of the following actions:	
		Select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type.
		Select a constraint and click **Edit** to open a dialog that lets you modify the constraint details.
		Select a constraint and click **Drop** to remove the constraint.
Storage	Lets you work with settings in the following categories:	
	Data Block Storage	Lets you choose the **Tablespace Name**, and specify **Percent Free**, **Initial Transactions**, and **Max Transactions** values.
	Extents	Lets you view **Initial Extent**, **Next Extent**, **Minimum Extents**, **Maximum Extents**, and **Percent Increase** values.
	Freelists	Lets you specify **Freelists** and **Freelist Groups** values.
	Bufferpool	Lets you specify a **Buffer Pool**.
IOT Properties	Lets you work with settings in the following categories:	
	Ungrouped	Lets you specify an overflow segment.
	Percent Threshold	Lets you specify the percentage of space reserved for an index-organized table.
	Key Compression	Lets you enable or disable compression and provide a compression value.
Space	Lets you work with settings in the following categories:	
	Space Utilization	Lets you view **Size** and number of **Blocks**.
	Row Information	Lets you view the **Number of Rows**, **Average Row Length**, and **Chain Rows** properties.
	Extents	Let you view the **Number of Extents** and **Maximum Extents** values.
LOB columns	Lets you work with settings in the following categories:	
	LOB Column Segment and **LOB Column Index**	Lets you view the **Segment Name**, **No. of Extents**, **Max Extents**, **Size**, and **Blocks** properties.
	Segment Extents and **Index Extents**	Lets you view the **Extent ID**, **File ID**, **Block ID** and number of **Blocks** properties for segment and index extents.
Partition	Lets you work with table partitions	

Tab	Settings and tasks
Comment	For details on using this tab, see Adding a Comment to an object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

TABLESPACES EDITOR (ORACLE)

The Tablespaces Editor lets you manage datafiles, space, storage, quotas, and objects for a tablespace.

To edit a tablespace

1   Open an editor on the tablespace. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Datafiles	The tab lists details for each datafile on the tablespace. Optionally you can add or delete files or edit file attributes. For detailed information on using this tab, see Tablespaces (Oracle) - Datafiles.
Properties	This tab lets you view the **Name**, **Big File**, **Type**, **Encrypted**, **Locally Managed**, **Minimum Extent Size**, **Uniform Allocation**, **Use Default Block Size**, **Block Size**, **Automatic Segment Space Management**, **Initial Extent**, **Next Extent**, **Minimum Extents**, **Maximum Extents**, and **Percent Increase** properties. This tab lets you modify the **Status**, **Logging**, **Force Logging**, and **Compression Type** properties. For more information on these properties, see Tablespaces (Oracle) - Properties.
Extent Details	Provides details (**Name**, **Type**, **File ID**, **Extent ID**, **Block ID**, **Blocks**, and **Bytes**) for each extent in the tablespace.
Space	Displays free blocks versus used blocks statistics for the tablespace, including a pie chart representation.

Tab	Settings and tasks
Map	This tab presents a graphical, segment map of the tablespace. The map is color-coded to show usage of each segment (FREE, SELECTED, OTHER, TEMPORARY, ROLLBACK, CACHE, TABLE, INDEX, TABLE PARTITION, NESTED TABLE, INDEX PARTITION, INDEX SUBPARTITION, TEXT, CLUSTER, LOB, LOBSEGMENT, LOBINDEX, or LOB PARTITION). Optionally you can:
	Click **Legend** to view the color legend of the map.
	Hover the mouse over a segment to display its **Segment ID**, **Block ID**, **Size** (kb), **Size** (in blocks), **Tablespace File ID**, and **Segment Type** details.
	Click **Display** to open a window that aids in navigating the graphical segment map. The **Overview** window displays a smaller version of the segment map. The selected area can be dragged up and down the window, displaying the selected area in the larger, main map. The selected area can also be resized, to display a larger or smaller area in the main map.
	Select an INDEX or TABLE segment on the map and click **Reorganize** to initiate a **Rebuild Indexes** or **Reorganize** (respectively) action. For details, see [Rebuild Index](#) and [Reorganize / Reorg](#).
	The **Object Demographics** area provides a tabular representation of the tablespace map. It provides details on a segment-by-segment basis.
	For background information, see [About Oracle Tablespace Storage](#).
Objects	Displays the objects currently stored on the tablespace, grouped under object type folders. Optionally you can:
	Specify a logging option using the **Log Changes When Scheme/Data is Modified?** radio set.
	Select an object under one of the object folders and click **Edit** to open an object editor on that object.
Quotas	Oracle limits the amount of space that can be allocated for storage of a user's objects within the specified tablespace to the amount of the quota. Users with privileges to create certain types of objects can create those objects in the specified tablespace. The **Quotas** tab of the Tablespace editor lets you manage user space quotas for tablespaces on the current datasource. Optionally, you can:
	Click **Add** or select a user and click **Edit** to assign a user unlimited or a specific space usage quota on the tablespace. For details, see [Adding or Editing User Tablespace Quotas](#).
	Select an existing user and click **Drop** to delete the quota for that user
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

ADDING OR EDITING USER TABLESPACE QUOTAS

When you assign a quota:

- Users with privileges to create certain types of objects can create those objects in the specified tablespace.

- Oracle limits the amount of space that can be allocated for storage of a user's objects within the specified tablespace to the amount of the quota.

  NOTE: This functionality is available for Oracle only.

User tablespace quotas are added and modified from the Tablespaces editor. For more information, see [Tablespaces Editor (Oracle)](#).

The table below describes the options and functionality on the **Add User Quota...** or **Edit User Quota...** dialog boxes:

Option	Description
User selection list (Add only)	Lets you select one or more users to assign a quota.
Quota	Lets you set a quota for the selected user or users. You can select an unlimited, or a specified size. Unlimited - Lets you place an unlimited quota on the tablespace. Other - Lets you place a specified quota in KB or MB on the tablespace.

ABOUT ORACLE TABLESPACE STORAGE

The **Storage** tab of the Tablespace Editor lets you view storage details for tablespaces on the current datasource.

> **TIP:**   Always create tablespaces for user data and never place user tables and indexes in the SYSTEM tablespace. Placing user objects in the SYSTEM tablespace can degrade performance and introduce space-related headaches to the database.

Oracle8i or later supports locally managed tablespaces, which can all but eliminate the problem of tablespace fragmentation. It totally does away with the storage parameters of MINEXTENTS, MAXEXENTS, PCTINCREASE, and NEXT. With locally managed tablespaces you either specify the initial extent size and let Oracle automatically size all other extents, or specify a uniform extent size for everything.

For users using a version earlier than Oracle 8i and locally managed tablespaces, there are manual methods can employ to assist in the fight against tablespace fragmentation. They include:

- Setting PCTINCREASE to zero for all tablespaces and objects to promote same-sized extents.

- Specifying equal-sized allotments for your INITIAL and NEXT object storage parameters.

- Grouping objects with like growth and storage needs together in their own tablespaces.

> **TIP:**   One of the best ways to avoid fragmentation in a tablespace is to pre-allocate the space that your objects will use. If possible, plan for one to two years' growth for each object and allocate your space accordingly. Having initial empty objects will not affect table scan times as Oracle only scans up to the high-water mark (the last used block) in a table.

Of all your tablespaces, you want to avoid fragmentation problems in your SYSTEM tablespace the most as this is the major hotbed tablespace for Oracle activities. The easiest way to avoid this is to not allow any user (even the default DBA ID's SYS and SYSTEM) to have access to it. There are three ways to do this:

- Ensure no user has a DEFAULT or TEMPORARY tablespace assignment of SYSTEM.

- Ensure no user has a quota set for SYSTEM.

Ensure no user has been granted the UNLIMITED TABLESPACE privilege.

TRIGGERS EDITOR (ORACLE)

The Triggers Editor lets you modify actions, events, and other details for a trigger.

To edit a trigger

1   Open an editor on the trigger. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	In addition to displaying basic identification and creation properties, this tab lets you work with settings in the following categories:	
	Attributes	Lets you specify whether the trigger is **Enabled**, select the **Trigger Timing** (BEFORE, AFTER), and the **Trigger Type** (ROW, STATEMENT).
	Correlation Names	Lets you provide an **Old Table Alias** and a **New Table Alias** as well as construct a **When Clause**.
	Status	Lets you view **Object Status**, **Create Date**, **Last Modified**, and **Base Object Type** properties.
	Size Information	Lets you view **Source Size**, **Parsed Size**, **Code Size**, and **Error Size** properties.
Events	Lets you select the INSERT, UPDATE, or DELETE event for the trigger.	
Column Selection	Lets you select the associated columns.	
Action	Lets you modify the trigger action PL/SQL block for any trigger on the datasource. You modify the body of the trigger in the Trigger Action (PL/SQL Block) area.	
Dependencies	For details on using this tab, see [Working with Object Dependencies](#).	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

TYPE BODIES EDITOR (ORACLE)

The Type Bodies Editor contains the code for the methods that implement an object type. To create or replace a type body in one of your schema, you must have the CREATE TYPE or CREATE ANY TYPE system privilege. To replace a type in another user's schema, you must have the DROP ANY TYPE system privilege.

While listed as separate objects in the Datasource Explorer, type bodies are created and edited on the **Body** tab of the Types Editor. For details, see Types Editor (Oracle).

TYPES EDITOR (ORACLE)

The Types Editor lets you manage header text and body text for a type.

To edit a type

1   Open an editor on the type. For details, see Opening an Object Editor.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Header	Lets you manage the type header text.
Body	Lets you create and modify type body text for the type. For information on creating type bodies, see Object Type Wizard (Oracle).
Information	Displays the header and body information for a type.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Privileges	For details on using this tab, see Working with Privileges and Permissions.

3   When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

UNIQUE KEYS EDITOR (ORACLE)

The Unique Keys Editor lets you manage columns, basic properties, storage and space, and partitions for a unique key.

To edit a unique key:

1   Open an editor on the unique key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Displays a listing of the columns making up the index. Details include the **Column** name, the **Datatype**, and whether the definition allows **Nulls**. Optionally, you can:	
		Click the New button to add a new column to the index.
		Select an existing column and click the Delete button to delete that column from the index.
Properties	In addition to displaying basic identification information, this tablets you work with settings in the following categories:	
	Enforcing Index	Lets you view **User-Defined**, **Index Owner**, and **Index Name** settings.
	Attributes	Lets you set **No Sort**, **Logging**, **Reverse**, **Validate**, **Deferrable**, **Deferred**, **Enabled**, **Cascade**, **Rely**, and **Update Date** properties.
Storage	Lets you work with settings in the following categories:	
	Data Block Storage	Lets you choose the **Tablespace Name**, and specify **Percent Free**, **Initial Transactions**, and **Max Transactions values**.
	Extents	Lets you specify **Initial Extent**, **Next Extent**, **Minimum Extents**, **Maximum Extents**, and **Percent Increase** values.
	Freelists	Free lists let you manage the allocation of data blocks when concurrent processes are issued against the cluster. Identifying multiple free lists can reduce contention for free lists when concurrent inserts take place and potentially improve the performance of the cluster. This tab lets you specify **Freelists** and **Freelist Groups** values.
	Buffer Pool	Lets you select a **Buffer Pool**.
Space	Lets you work with settings in the following categories:	
	Space Utilization	Displays the **Size** and number of **Blocks** properties.
	Statistics	Lets you view **Index Level**, **Distinct Keys**, **Cluster Factor**, **Leaf Blocks**, **Leaf Blks/Key** and **Data Blks/Key** settings.
	Extents	The unit of space allocated to an object whenever the object needs more space. This tab lets you view **Number of Extents** and **Maximum Extents** properties.

Tab	Settings and tasks	
Partition	If the unique key is currently partitioned, this tab displays the following partition details:	
	Properties	Lets you view the **Locality** (Global/Local), **Alignment** (Prefixed/Non-Prefixed), **Partition Type** (RANGE or HASH), and **Subpartition type** properties.
		Click the Edit Partition button to edit partition details. For more information, see [Partitioning Oracle indexes, primary keys, and unique keys](#).
		Click the Drop Partition button to revert to an unpartitioned unique key.
		Use the following **Partition Commands** menu items: [Allocate Extent](#) [Analyze](#) [Coalesce](#) [Deallocate Unused Space](#) **Mark Unusable** **Rebuild** **Split**
	Columns	Displays partitioning columns.
	Partition Definitions	Displays details for each partition.
	If the unique key is not currently partitioned, you can click the **Convert To Partitioned** button to partition the unique key. For more information, see [Partitioning Oracle indexes, primary keys, and unique keys](#).	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

USERS EDITOR (ORACLE)

The Users Editor lets you manage basic properties, roles, tablespace quotas, and associated objects for a user.

To edit a user

1   Open an editor on the user. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with settings in the following categories:	
	Defaults	Lets you select a **Default Tablespace** and Temporary **Tablespace** as well as select the **Profile** used by this user. For more on creating profiles, see <u>Profile Wizard (Oracle)</u>.
	Account	**Identified By** - Lets you select among REQUIRED YES, REQUIRED EXTERNAL, and REQUIRED_GLOBAL.
		Password - (available with **Identified By** value of REQUIRED_YES only) lets you specify a password.
		External Name - (available with **Identified By** value of REQUIRED_GLOBAL only) lets you specify the name of the user in the enterprise directory service.
		Account Locked - lets you lock this user account.
		Password Expired - (available with **Identified By** value of REQUIRED_YES only) lets you mark the password as expired, forcing the user to change their password before being allowed to connect.
Role	Lets you select the roles that are to be assigned to this user.	
Quota	Lets you set tablespace quotas for the user. Set an unlimited quota by selecting a tablespace and selecting the **Unlimited** radio button. Set a specific quota by selecting a tablespace, selecting the **Other** radio button, and providing a **Quota** value in kilobytes or megabytes.	
Objects	Lets you manage database objects associated with a user. Objects are grouped within object type folders. Optionally, you can:	
		Select an object and click **Edit** to open an object editor on the selected object.
		Select an object and click **Drop** to initiate dropping the selected object.
Object Privileges and **System Privileges**	For details on using these tabs, see <u>Working with Privileges and Permissions</u>.	
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

VIEWS EDITOR (ORACLE)

The Views Editor lets you work with view columns and manage the dependencies and permissions for the view.

To edit a view

1   Open an editor on the view. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays a listing of the columns making up the view.
Definition	Lets you view and modify the DDL that will implement any changes made with the editor.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PARTITIONING ORACLE INDEXES, PRIMARY KEYS, AND UNIQUE KEYS

The **Add Partition** wizard lets you partition an Oracle index, primary key, or unique key.

To partition an index, primary key, or unique key

1   Open an editor on the index, primary key, or unique key. For details, see <u>Opening an Object Editor</u>.

2   On the **Partition** tab, click **Convert to Partitioned**.

> NOTE: If the object is already partitioned, the **Convert to Partitioned** button is not present.

3   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	In addition to displaying basic identification and ownership properties, this pane lets you work with the following settings: **Locality** - GLOBAL or LOCAL. **Alignmen**t - displays whether this partitioned object is prefixed or non-prefixed. **Partition Type** - RANGE or HASH. **Subpartition Type** - displays the subpartition type.
Columns	For each column to be added, click New to add a column, select a **Column** name, and specify a **Sort** order.
Range Definitions (only available for a **Partition Type** of RANGE)	Lets you create an ordered list of partitions. For each partition, click New to open the **Add Partition Definition** wizard. On the **Partition Definition** pane, provide a **Name**, select a **Tablespace**, and enable or disable **Logging**. On the **Storage** pane, provide **Data Block Storage**, **Extents**, and **Freelists** property values. Click **Add**. The **Range Definitions** pane also lets you edit and remove partitions.

Tab	Settings and tasks
Hash Definitions (only available for a **Partition Type** of HASH)	Lets you specify a Partition method: **None** **Number of Partitions** - Specify a **Number of Partitions**, and then for each partition, click the New button and select a tablespace. **By Partition Name** - For each partition, click New, specify a **Name** and select a **Tablespace**.

4   When ready, click **Finish**.

SYBASE ASE OBJECT EDITORS

Rapid SQL includes an Object Editor for all supported Sybase ASE objects. To see an Editor for a specific object, click the corresponding link below:

> NOTE: If an objects has dependent objects, such as tables, triggers, procedures and views, you can view and access their dependent objects in the editor.

- Aliases Editor (Sybase)

- Check Constraints Editor (Sybase)

- Databases Editor (Sybase)

- Defaults Editor (Sybase)

- Extended Procedures Editor (Sybase)

- Foreign Keys Editor (Sybase)

- Functions Editor (Sybase)

- Groups Editor (Sybase)

- Indexes Editor (Sybase)

- Logins Editor (Sybase)

- Primary Keys Editor (Sybase)

- Procedures Editor (Sybase)

- Rules Editor (Sybase)

- Segments Editor (Sybase)

- Tables Editor (Sybase)

- Triggers Editor (Sybase)

- Unique Keys Editor (Sybase)

- User Datatypes Editor (Sybase)

- Users Editor (Sybase)

- Users Editor (Sybase)

ALIASES EDITOR (SYBASE)

The Aliases Editor lets you view and modify an alias definition.

To edit an alias

1 Open an editor on the alias. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you change the user referenced by the alias.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

CHECK CONSTRAINTS EDITOR (SYBASE)

The Check Constraints Editor lets you manage a check constraint definition and modify the condition expression.

To edit a check constraint

1 Open an editor on the check constraint. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you edit the condition in the **Check Condition** box. The **Table Columns** button acts a time saver in editing the condition. It opens a dialog that lets you select and paste column names into the check condition expression. NOTE: If the **Check Condition** box shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see Hide Text.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

DATABASES EDITOR (SYBASE)

The Databases Editor lets you manage properties, placement, and binds for a database as well as view associated space statistics.

To edit a database

1 Open an editor on the database. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with the following categories:	
	Creation Properties	Has the **Database Name**, **Owner**, **Transfer aliases and permissions**, and **Type** properties.
	Options	Lets you work with the **abort tran on log full**, **allow nulls by default**, **auto identity**, **dbo use only**, **ddl in tran**, **identity in nonunique index**, **no chkpt on recovery**, **no free space acctg**, **read only**, **select into/bulkcopy/pllsor**t, **single user**, and **trunc log on chkpt** properties.
Placement	Displays device fragments. Optionally you can:	
		Select a fragment to view the **Device Name**, **Size**, and **Device Type** (DATA AND LOG or DATA ONLY) for the fragment.
		Click the New button to open a dialog that lets you provide the data device and log device details for a new database fragment.
		NOTE: We strongly recommend that you place the transaction log on a separate device from all other database fragments.
Space	Lets you view pie charts showing the data space usage and the transaction log (if available) space usage for the database.	
Bindings	This tab is only available for databases created with a **Type** of TEMP. It displays all applications or logins bound to the database. Optionally, you can:	
		Bind another application or login to the database by selecting the APPLICATION or LOGIN folder as appropriate, clicking **Add**, and in the dialog that opens select a login or provide an application name.
		Remove bindings by selecting the specific application or login and clicking **Drop**.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.	

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

DEFAULTS EDITOR (SYBASE)

The Defaults Editor lets you modify the value and manage bindings for a default.

To edit a default

1 Open an editor on the default. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you change the **Value** of the default. NOTE: If the **Value** box shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see Hide Text.
Dependencies	For details on using this tab, see Working with Object Dependencies.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.

3 When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

EXTENDED PROCEDURES EDITOR (SYBASE)

The Procedures Editor lets you view and modify an extended procedure definition.

To edit an extended procedure

1 Open an editor on the extended procedure. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you specify the **Library Name** for the extended procedure. NOTE: If the **Library Name** box shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see Hide Text.
DDL View	For details on using this tab, see Viewing the SQL/DDL for an Object.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Permissions	For details on using this tab, see Working with Privileges and Permissions.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting</u> <u>Object Editor Changes</u>.

FOREIGN KEYS EDITOR (SYBASE)

The Foreign Keys Editor lets you modify the column mapping and specify properties of a foreign key.

To edit a foreign key

1   Open an editor on the foreign key. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Column Mappings	The existing column mapping for the foreign key is represented by selected columns in the **Main Table** and **Referenced Table** lists. Additional candidates in the **Referenced Table** list are indicated by enabled column check boxes.
	If necessary, use the **Table** dropdown in the **Referenced Table** group to choose a new table for this foreign key. Select or deselect columns in the **Main Table** list and **Referenced Table** list to form the referential constraint between the two tables.
Properties	Lets you specify a **Match Full** value of TRUE or FALSE.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting</u> <u>Object Editor Changes</u>.

FUNCTIONS EDITOR (SYBASE)

The Functions Editor lets you view and modify the body of a function, as well as work with permissions and dependencies.

To edit a function:

1   Open an editor on the function. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you view and modify the CREATE FUNCTION statement, including the function body, for the function.
	NOTE: If the **Definition** tab for the function shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see <u>Hide Text</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.

Tab	Settings and tasks
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

GROUPS EDITOR (SYBASE)

The Groups Editor lets you manage users in a group and the group's associated privileges.

> **TIP:**   The refresh button lets you refresh or clear the editor's contents, and log SQL.

To edit a group

1   Open an editor on the group. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Users	Lets you add users to, or remove users from the group. To move a user from one of the **Users Not In Group** or **Users In Group** lists to the other list, select the user in the list and click the **Join Group** or **Leave Group** button.
Object Privileges	For details on using this tab, see <u>Working with Privileges and Permissions</u>.
System Privileges	For details on using this tab, see <u>Working with Privileges and Permissions</u>.
DDL	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

INDEXES EDITOR (SYBASE)

The Indexes Editor lets you manage columns, properties, and partitions, and view statistics for an index.

To edit an index

1 Open an editor on the index. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks		
Columns	Lets you manage columns that make up the index. On opening, this tab shows the existing columns. For each column, the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:		
			Change the **Sort** order of a column.
			Click the New button and select a column name from the dropdown, to add a column to the index.
			Select a column and click the Drop button to delete the column from the index.
			Select a column and use the arrow buttons to reorder the columns in the index.
Properties	Lets you work with settings in the following categories:		
	Attributes		Has the **Index Type** (UNIQUE or NONUNIQUE), **Clustered**, **Ignore Duplicate Key** (available with **Index Type** of UNIQUE), **Ignore Duplicate Rows** (available with **Clustered** enabled), **Allow Duplicate Rows** (available with **Clustered** enabled), and **Maximum Rows Per Page** properties.
	Storage		Lets you set **Reserve Page Gap**, **Segment Name**, **Fill Factor**, **Prefetch Strategy**, **MRU Replacement Strategy** settings
Partition	Lets you work with settings in the following categories:		
	Properties		Lets you view the **Locality** (Global/Local), **Alignment** (Prefixed/Non-Prefixed), **Partition Type** (including Range-Hash Composite or Range-List Composite), and **Subpartition type** properties.
	Columns		Displays partitioning columns.
	Partition Definitions		Displays details for each partition.
Statistics	Lets you view statistics in the following categories:		
	Page Statistics		**Data Pages**, **Reserved Pages**, **Used Pages**, **OAM Page Ratio**, and **Index Page Ratio**.
	Row Statistics		**Maximum Row Size**, **Minimum Row Size**, **Max Size of Non-Leaf Index Row**, and **Maximum Rows Per Page**.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.		

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

LOGINS EDITOR (SYBASE)

The Logins Editor lets you manage basic properties for a login, associated users and roles, and accounting.

To edit a login:

1   Open an editor on the login. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with the following properties set when creating the login: Name, FullName, Default database, Default language, Locked, Password, Password expiration, Minimum password length, Maximum Login attempts, and Authentication Mechanism. For detailed information on these settings, see <u>Login Wizard (Sybase)</u>. In addition, when editing an existing login, you have access to the following settings:	
	Is currently logged in	Displays whether the user is currently logged in.
	Login trigger	Displays the login trigger currently configured for this login. This is the stored procedure automatically executed when the user logs on. For information on configuring login triggers for a login, see <u>Add/Modify Login Trigger</u> and <u>Drop Login Trigger</u>.
Users	Lets you maintain database user accounts for a login. For more detailed information, see <u>Logins (Sybase) - Users</u>.	
Roles	Lets you maintain the roles granted to this login. For more detailed information, see <u>Logins (Sybase) - Users</u>. For information on activating and deactivating roles for the current login in the current session, see <u>Activating/Deactivating Roles in the Current Session</u>.	
Accounting	The Accounting tab of the Logins Editor lets you manage chargeback accounting statistics for every login on the current server. Chargeback accounting statistics are CPU and I/O usage statistics that Sybase ASE accumulates for every login. To start a new accounting period, the system administrator must clear all previous login statistics. Optionally, you can click the Clear Statistics button to start a new accounting interval. The Clear Statistics dialog lets you clear statistics immediately or use scheduling options.	
DDL	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PRIMARY KEYS EDITOR (SYBASE)

The Primary Keys Editor lets you manage the columns that make up the primary key and its basic properties, as well as view statistics for the primary key.

To edit a primary key

1 Open an editor on the primary key. For details, see <u>Opening an Object Editor</u>.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the primary key. On opening, this tab shows the existing columns. For each column, the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
		Change the **Sort** order of a column.
		Click the New button and select a column name from the dropdown, to add a column to the primary key.
		Select a column and click the Drop button to delete the column from the primary key.
		Select a column and use the arrow buttons to reorder the columns in the primary key.
Properties	Lets you work with settings in the following categories:	
	Attributes	Lets you set the **Clustered** and **Maximum Rows Per Page** properties.
	Storage	Lets you set **Reserve Page Gap**, **Segment** (DEFAULT, LOGSEGMENT or SYSTEM), and **Fill Factor** properties.
Statistics	Lets you view statistics in the following categories:	
	Page Statistics	**Data Pages**, **Reserved Pages**, **Used Pages**, **OAM Page Ratio**, and **Index Page Ratio**.
	Row Statistics	**Maximum Row Size**, **Minimum Row Size**, **Max Size of Non-Leaf Index Row**, and **Maximum Rows Per Page**.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.	

3 When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PROCEDURES EDITOR (SYBASE)

The Procedures Editor lets you view and modify procedure definitions and modify associated privileges and dependencies.

To edit a procedure

1   Open an editor on the procedure. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Definition	Lets you modify the SQL code for a procedure. NOTE: If the **Definition** tab for the procedure shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see <u>Hide Text</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

RULES EDITOR (SYBASE)

The Rules Editor lets you modify basic properties of a rule.

To edit a rule

1   Open an editor on the rule. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you specify a **Restriction** and select a **Type** (STANDARD_RULE, AND_ACCESS_RULE, OR_ACCESS_RULE). NOTE: If the **Restriction** box shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see <u>Hide Text</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

SEGMENTS EDITOR (SYBASE)

The Segments Editor lets you manage segment location, associated objects, segment space, and segment thresholds.

To edit a segment

1 Open an editor on the segment. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Location	Lets you **Drop** or **Extend** selected segments.
Objects	Lets you manage database objects associated with the segment. Objects are organized in a tree structure. You can open an object editor on a table, index, or constraint by double-clicking that object
Space	Lets you view **Segment Usage**, **Distribution of Segment Space**, **AutoGrowth Properties**, and **Object Space Usage** for the segment.
	Shows the **Hysteresis** value for the segment and for each threshold for the segment, the number of **Free Pages** and an associated **Procedure Name**. Manage thresholds as follows:
	Click the **Add** button to specify the number of free pages and procedure name for a new segment threshold
	Select an existing threshold and click the **Edit** button to modify the threshold's procedure name value
	Select an existing threshold and click the **Drop** button to delete that threshold
Threshold	Shows the **Hysteresis** value for the segment and for each threshold for the segment, the number of **Free Pages** and an associated **Procedure Name**. Manage thresholds as follows:
	Click the **Add** button to specify the number of free pages and procedure name for a new segment threshold
	Select an existing threshold and click the **Edit** button to modify the threshold's procedure name value
	Select an existing threshold and click the **Drop** button to delete that threshold
DDL	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

TABLES EDITOR (SYBASE)

The Tables Editor lets you manage columns, basic properties, constraints, storage and space, and partitions of a table.

To edit a table

1 Open an editor on the table. For details, see Opening an Object Editor.

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks		
Columns	Displays the currently defined columns in the table. For any selected column, the Property/Value list provides additional detail on that column. Available properties depend on the datatype you choose as well as on the property values you select: **Computed** and **Computed Expression** - Let you define a column as a computed column and provide the computed column expression. **Type** - Lets you select a datatype (depending on the type, additional properties such as **Size**, **Width**, and **Scale** may be available). **Identity Column** - Select this check box to define the column as an identity column. **Allow Nulls** - Select this check box to allow nulls in this column. **Default Collation** - available for text/character datatypes, lets you specify a default collation. **Default Value** - Lets you type a constant value or select a function returning a constant value to serve as the default for the column **Default Binding** and **Rule Binding** - Let you bind a rule or default to a column. Optionally, you can:		
		Select a column, and in the **Property/Value** list modify property values for that column.	
		Click **Add Column**, provide a name for the new column, and set property values for the column.	
		Select a column and click **Delete** to remove the column from the table.	
Properties	Lets you work with properties in the following categories:		
	Physical Storage	Lets you set the **Segment Name** (DEFAULT, SYSTEM), **Maximum Rows Per Page**, **Reserve Page Gap**, and **Identity Gap** properties.	
	Cache Strategy	Lets you set **MRU Replacement Strategy** and **Prefetch Strategy** properties.	
	Row Locking Strategy	Lets you view the **Expected Row Size** property and set the **Lock Scheme** (ALLPAGES, DATAPAGES, ROWPAGES) property.	
Indexes	Lets you manage indexes for a table. On opening, the list of current indexes for the table is displayed. Optionally, take one of the following actions:		
		Click **Add** to open a dialog that lets you add a new index to the table.	
		Select an index and click **Edit** to open a dialog that lets you edit index properties.	
		Select an index and click **Drop** to open a dialog that lets you remove the index from the table.	

Tab	Settings and tasks		
Constraints	Lets you manage constraints for the table. Constraints are grouped by type, under folders. Optionally take one of the following actions:		
		Select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type.
		Select a constraint and click **Edit** to open a dialog that lets you modify the constraint details.
		Select a constraint and click **Drop** to remove the constraint.
Space	Lets you view the table usage and the distribution of table space for a table. Optionally, double-click a slice in the pie chart for detailed statistics.		
Partitions	Displays partition details for the table including the **Partition Type** (ROUNDROBIN, HASH, LIST, RANGE), partitioning **Columns** (name and datatype) and **Partition Definitions** (Name, values, segments). You can also add or modify partition definitions. For more information, see <u>Partitioning Sybase Tables</u>.		
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.		
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.		
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.		

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

PARTITIONING SYBASE TABLES

The **Create Partition** and **Edit Partition** wizards let you work with table partitions.

To create or edit table partitions

1   Open an editor on a table. For details, see <u>Opening an Object Editor</u>.

2   On the **Partition** tab, click the **Create Partition** or **Edit Partition** button.

3   Use the following table as a guide to understanding and modifying the settings on the tabs of this wizard:

Tab	Settings and tasks
Properties	In addition to displaying basic identification and ownership properties, this pane lets you select a **Partition Type** (ROUNDROBIN, HASH, LIST, RANGE).
Columns (not available for a **Partition Type** of ROUNDROBIN)	For each column to be added, click New to add a column and select a **Column** name.
Range Definitions (only available for a **Partition Type** of RANGE)	Lets you create an ordered list of partitions. For each partition, click New to open the **Add Partition Definition** wizard. Provide a **Name**, select a **Segment**, (DEFAULT, SYSTEM, LOGSEGMENT) and click **Add**. The **Range Definitions** pane also lets you edit and remove partitions.

Tab	Settings and tasks
Partitions Definitions (only available for a **Partition Type** of HASH)	Lets you specify a Partition method: **None** **Number of Partitions** - Specify a **Number of Partitions**, and then for each partition, click the New button and select a segment. **By Partition Name** - For each partition, click New, specify a **Name** and select a **Segment**.
List Definitions (only available for a **Partition Type** of LIST)	Lets you create a list of partition definitions. For each partition, click New to open the **Add Partition Definition** wizard. Provide a **Name**, select a **Segment**, (DEFAULT, SYSTEM, LOGSEGMENT) and click **Add**. The **List Definitions** pane also lets you edit and remove partitions.

4   When ready, click **Finish**.

TRIGGERS EDITOR (SYBASE)

The Triggers Editor lets you modify the trigger body of a trigger and change its enabled/disabled status.

To edit a trigger:

1   Open an editor on the trigger. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the basic identifying properties of the trigger and its parent view or table: **Parent Type**, P**arent Owner**, **Parent Name**, **Owner**, **Name**, **Create Date**, **Fire On Insert**, **Fire On Update**, and **Fire On Delete**. For details on these properties, see <u>Triggers (Sybase) - Properties</u>. This panel also lets you specify whether the trigger is Enabled.
Definition	Lets you modify the CREATE TRIGGER body for a trigger. To modify a trigger, Rapid SQL must drop then create the trigger. NOTE: If the **Definition** tab for the trigger shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see <u>Hide Text</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>

UNIQUE KEYS EDITOR (SYBASE)

The Unique Keys Editor lets you manage the columns and basic properties for a unique key as well as view page and row statistics.

To edit a unique key

1 Open an editor on the unique key. For details, see [Opening an Object Editor](#).

2 Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Columns	Lets you manage columns that make up the unique key. On opening, this tab shows the existing columns. For each column, the datatype (and if applicable the precision in brackets) and whether the table definition permits nulls in the target table column. Optionally you can:	
		Change the **Sort** order of a column.
		Click the New button and select a column name from the dropdown, to add a column to the index.
		Select a column and click the Drop button to delete the column from the index.
		Select a column and use the arrow buttons to reorder the columns in the index.
Properties	Lets you work with properties in the following categories:	
	Attributes	Lets you set **Clustered** and **Maximum Rows Per Page** properties.
	Storage	Lets you set the **Reserve Page Gap**, and **Segment** (DEFAULT, LOGSEGMENT, SYSTEM) properties.
Statistics	Lets you view statistics in the following categories:	
	Page Statistics	**Data Pages**, **Reserved Pages**, **Used Pages**, **OAM Page Ratio**, and **Index Page Ratio**.
	Row Statistics	**Maximum Row Size**, **Minimum Row Size**, **Max Size of Non-Leaf Index Row**, and **Maximum Rows Per Page**.
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3 When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

USER DATATYPES EDITOR (SYBASE)

The User Datatypes Editor lets you view and modify user datatype definitions and view objects associated with a user datatype.

To edit a user datatype

1   Open an editor on the user datatype. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks	
Properties	Lets you work with properties in the following categories:	
	Base datatype	Lets you set **Type**, **Size**, **Allow Nulls**, and **Identity** settings.
	Bindings	Lets you set **Default Binding** and **Rule Binding** properties.
Usage	Displays all objects referring to or referenced by a user datatype.	
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).	

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

USER MESSAGES EDITOR (SYBASE)

The User Messages Editor lets you manage user messages and their respective object bindings.

> **NOTE:**   On Sybase, user messages are only available under the **Master** database.

To edit a user message

1   Open an editor on the user message. For details, see [Opening an Object Editor](#).

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Information	Displays the message text for each language version of the message and offers the options to add a new language version, drop a language version, or modify a language version. For detailed instructions, see [User Messages (Sybase) - Information](#).
Bindings	Displays any constraints to which the user message is bound. This tab also lets you create new bindings and delete current bindings. For details, see [User Messages (Sybase) - Bindings](#).
DDL View	For details on using this tab, see [Viewing the SQL/DDL for an Object](#).

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

USERS EDITOR (SYBASE)

The Users Editor lets you view and modify user definitions and manage associated logins and objects.

To edit a user

1   Open an editor on the user. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Lets you view the user **Name** and **Login Name** for the user. Lets you specify a **Group Name** for the user.
Logins	Lets you assign logins to a user by selecting associated check boxes.
Objects	Lets you manage database objects associated with the user. Objects are organized in a tree structure with folders containing the objects. Optionally, take one of the following actions:
	Select an object and click **Edit** to open an object editor on the selected object.
	Select an object and click **Drop** to initiate dropping the selected object.
Object Permissions and **System Permissions**	For details on using these tab, see <u>Working with Privileges and Permissions</u>.
Comment	For details on using this tab, see <u>Adding a Comment to an object</u>.
DDL View	For details on using this tab, see <u>Viewing the SQL/DDL for an Object</u>.

3   When finished, you can submit your changes. For details, see <u>Previewing and Submitting Object Editor Changes</u>.

VIEWS EDITOR (SYBASE)

The Views Editor lets you view the columns for the view and work directly with the SQL to create the view.

To edit a view

1   Open an editor on the view. For details, see <u>Opening an Object Editor</u>.

2   Use the following table as a guide to understanding and modifying the settings on the tabs of this editor:

Tab	Settings and tasks
Properties	Displays all columns for the view. Details for each column include the **Column Name**, the **Datatype** (and if applicable, with the precision in parentheses), and whether or not **Nulls** are allowed for that column.
Definition	Displays the SQL (CREATE VIEW) for the view. NOTE: If the **Definition** tab for the view shows a **Source text for this compiled object is hidden** message, source text for this compiled object has been hidden. For more information, see <u>Hide Text</u>.
Dependencies	For details on using this tab, see <u>Working with Object Dependencies</u>.
Permissions	For details on using this tab, see <u>Working with Privileges and Permissions</u>.

3   When finished, you can submit your changes. For details, see [Previewing and Submitting Object Editor Changes](#).

OBJECT ACTIONS

In addition to letting you create and perform basic editing on various object types, Rapid SQL offers a number of actions that can be applied to objects. While object action support differs by DBMS platform, they are most commonly implementations of:

- ALTER *OBJECT TYPE* clauses/options not typically available from object editors, such as REBUILD INDEX and RENAME

- Other SQL commands such as TRUNCATE, REORG, and LOCK

- System procedures and other available utilities

Most object actions are implemented as multi-panel wizards that guide you through the action. For an introduction to invoking object actions and working through the panels, see Overview of object actions/operations execution.

For a listing of available actions and access to detailed instructions, see Available object actions by DBMS.

OVERVIEW OF OBJECT ACTIONS/OPERATIONS EXECUTION

Most commonly, initiating an object operation opens a wizard with panels that guide you through initiating the action. For details, see the following topics

- Initiating an object operation

- Using object operation wizards

INITIATING AN OBJECT OPERATION

Object operations are most commonly available as right-click context menu selections on the Database Explorer, available with one or more objects selected.
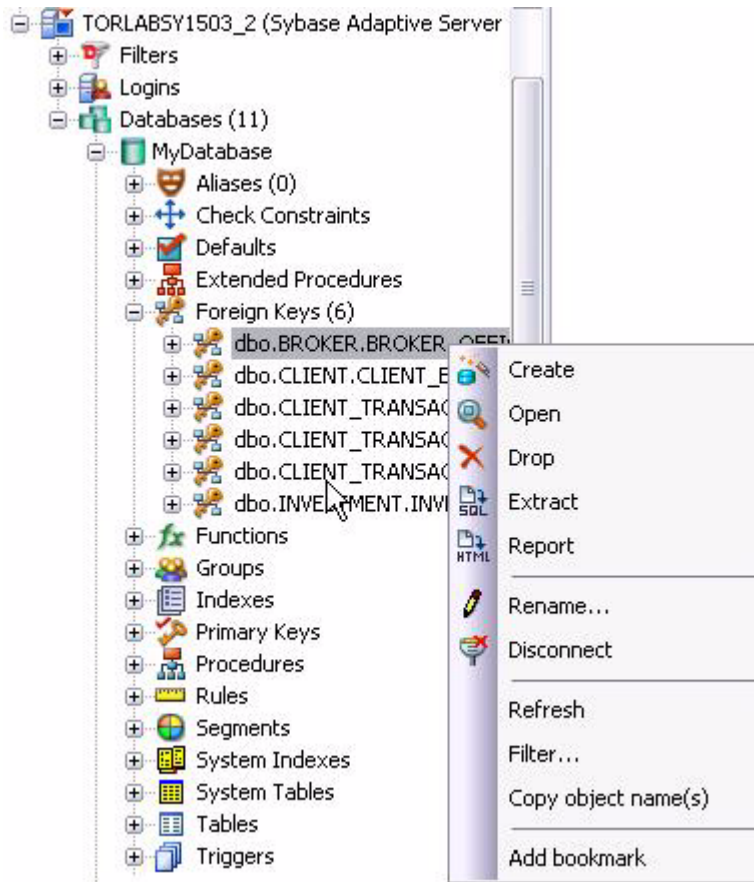
NOTE: To verify availability of an object action for a specific object type and DBMS platform, see the topic covering that object type in Supported Objects.

To initiate an object operation

1   Connect to the datasource that contains the object you want to perform the operation against. For more information, see Connect.

2   On the Database Explorer, expand nodes until the target object type is visible and then expand the target object type. See the following topics for more information:

- Database Explorer - for general information on working with the Explorer.

- Supported Objects, specific object type topic - for information on navigating Datasource Explorer nodes to the target object type.

Rapid SQL displays objects of that type in the Database Explorer.

3 Take one of the following actions:

- To initiate an operation against a single object, right-click the object and select the operation from the context menu.

- To initiate an operation against a multiple objects, use CTRL-clicking to select multiple objects, right-click one of the selected objects and then select an operation from the context menu.

    NOTE: Not all object operations can be applied to multiple objects.

    Rapid SQL opens the wizard for the selected object operation.

For information on using the wizard panels to provide required information and execute an operation, see Using object operation wizards.

USING OBJECT OPERATION WIZARDS

A typical object operation wizard is a multi-panel window that walks you through the steps necessary to execute the operation. The wizard typically opens on a first panel named either **Action Options** or using a name specific to that object action. For example, the first panels for the **Rename** and **Drop** operations:
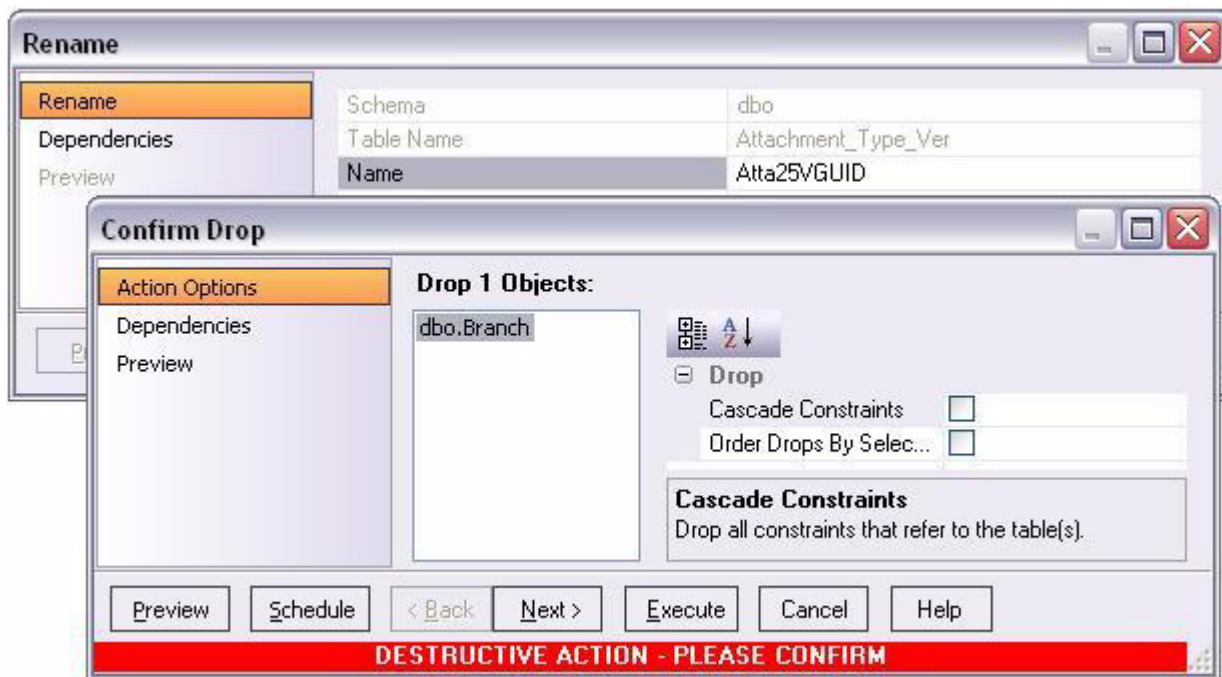
The wizard's **Next** and **Back** buttons let you navigate through the wizard panels. While the remaining panels of the wizard differ by action type, the following are the most common:

- Dependencies

- Preview

After providing information and inspecting settings on the panels, there are two execution options:

- Use the **Execute** (or **Finish**) button to execute the operation immediately

- Use the **Schedule** button to schedule the operation to execute the operation at a later time or to be executed on a regular basis. For information, see Scheduling.
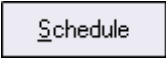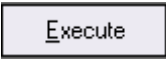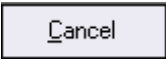
DEPENDENCIES

For those object action wizards that include a **Dependencies** panel, it lists the objects potentially impacted by this object action. Within the main categories of **Referencing Objects** and **Referenced Objects**, objects are grouped within object type folders.

CAUTION: A **Dependencies** panel, if present, is provided for information only. Rapid SQL does not attempt to resolve the current action in referenced or referring objects.

PREVIEW

The Preview panel lets you inspect the SQL generated by wizards or editors when you create, edit, or perform other actions against a database object. Depending on the action being taken, this panel may contain the following controls:

Button	Description
	Lets you save the SQL.
	Lets you print the SQL.
	Lets you send the SQL via e-mail.
	Opens an SQL Editor on the SQL contained in the Preview dialog. For more information, see ISQL Editor.
Schedule	Opens a dialog that lets you select job scheduling options. Once you have selected your options, a third-party job scheduler opens. For more information, see Scheduling.
Execute	Executes the SQL.
Cancel	Closes the Preview dialog without executing the SQL and returns you to the wizard, editor, or dialog box from which you invoked the Preview dialog. Since you cannot edit the SQL in the Preview dialog, this lets you make changes before once again previewing and then executing the SQL.

AVAILABLE OBJECT ACTIONS BY DBMS

The table below lists the object actions provided by Rapid SQL

	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	Oracle	SQL SVR	Sybase
Add or Modify Check Constraint	✓	✓	✓		✓	✓	✓
Add/Modify Login Trigger							✓
Allocate Extent					✓	✓	
Analyze					✓	✓	
Analyze Tables				✓			
Attach Database						✓	
Bind Package		✓					
Bind To Temporary Database							✓
Build		✓					
Build Query	✓	✓	✓	✓	✓	✓	✓
Change Access Status					✓		
Change Category					✓		
Change Password					✓	✓	
Change Status			✓		✓	✓	✓
Check Tables				✓			
Checkpoint						✓	✓
Checksum Tables				✓			
Coalesce					✓		
Compile					✓		
Convert Tables				✓			
Copy Object Names	✓	✓			✓	✓	✓
Copy Schema	✓						
Create Alias	✓	✓					
Create Clone		✓					
Create Insert Statements	✓	✓	✓	✓	✓	✓	✓
Create Like	✓	✓	✓	✓	✓	✓	✓
Create Synonym	✓	✓			✓		
Create View	✓	✓	✓		✓	✓	✓
DBCC						✓	✓
Deallocate Unused Space					✓		
Delete Statistics							✓
Describe	✓						
Detach Database						✓	

	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	Oracle	SQL SVR	Sybase
Disable Index			✓			✓	
Disable Job					✓		
Disable Keys				✓			
Disable/Enable Triggers						✓	✓
Drop	✓	✓	✓	✓	✓	✓	✓
Drop By Category					✓		
Drop Clone		✓					
Drop Login Trigger							✓
Drop Materialized Query Table	✓						
Drop Unused					✓		
Drop Unused Columns					✓		
Edit Data	✓	✓	✓	✓	✓	✓	✓
Enable Job (Job Queue)					✓		
Enable Keys				✓			
Enable Recycle Bin					✓		
Estimate Size							✓
Exchange Data With Clone		✓					
Execute			✓		✓		✓
Extract	✓	✓	✓	✓	✓	✓	✓
Extract Data as XML					✓	✓	
Flashback Table					✓		
Flashback Recycle Bin Entry					✓		
Flush Cache	✓						
Flush Tables				✓			
Free (Packages)		✓					
Free Plan		✓					
Generate Package/ Procedure/Statement	✓	✓	✓		✓	✓	✓
Hide Text							✓
Import Data From File	✓	✓	✓	✓	✓	✓	✓
Load Java					✓		
Lock	✓	✓					
Move Log							✓
Next Used Filegroup						✓	
Optimize Tables					✓		
Optimize Tables				✓			

	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	Oracle	SQL SVR	Sybase
Place						✓	✓
Population status						✓	
Purge Recycle Bin					✓		
Purge Recycle Bin Entry					✓		
Quiesce	✓						
Reassign By Category					✓		
Rebind Packages	✓	✓					
Rebind Plans	✓	✓					
Rebuild (Full-text Catalogs)						✓	
Rebuild Index		✓	✓		✓	✓	
Rebuild Outlines					✓		
Rebuild Table				✓			
Recompile						✓	✓
Refresh Table	✓						
Rename	✓	✓	✓	✓	✓	✓	✓
Reorganize /Reorg	✓	✓			✓	✓	✓
Repair Tables				✓			
Report	✓	✓	✓	✓	✓	✓	✓
Restart	✓						
Run Job					✓		
Schema	✓	✓	✓	✓	✓	✓	✓
Select * From	✓	✓	✓	✓	✓	✓	✓
Set Default					✓		
Set Integrity	✓	✓					
Set Online/Offline							✓
Set Statistics			✓				
Set UNDO					✓		
Shrink					✓	✓	
Start Database		✓					
Stop Database		✓					
Switch Online	✓						
Transfer Ownership	✓						
Truncate	✓	✓		✓	✓	✓	✓
Unbind From Temporary Database							✓
Update Statistics	✓					✓	✓

ADD OR MODIFY CHECK CONSTRAINT

NOTE: This functionality is available for DB2 LUW, DB2 z/OS, Interbase/Firebird, Oracle, SQL Server, and Sybase.

To open the Check Constraint Editor or Check Constraint Wizard

1   Take one of the following actions:

   • On the Explorer, right-click the **Check Constraints** node and select **Create**.

   • Select the **Check Constraints** node and in the right hand side of the Explorer, right-click a check constraint and select **Edit**.

   • Expand the **Check Constraints** node to display existing check constraints, right-click a check constraint and select **Edit**.

   • On the **Constraints** tab of a Table Wizard or Table Editor, select the **Check Constraints** folder and click **Add**.

2   Use the following table as a guide to creating or editing the constraint:

Step	Settings and tasks	
Definition	**Table Owner** or **Table Schema** (DB2 LUW, DB2 z/OS, SQL Server, Oracle, Sybase)	Select the owner of the table for which the check constraint is to be created.
	Table or **Table Name** (DB2 LUW, DB2 z/OS, Interbase/Firebird, SQL Server, Oracle, Sybase)	Lets you select where you want to place the check constraint.
	Name (DB2 LUW, DB2 z/OS, Interbase/Firebird, SQL Server, Oracle, Sybase)	Lets you type the name of the constraint, which can be up to 30 characters long and must be unique across the entire database.
	Enabled or **Enforced** (DB2 LUW, DB2 z/OS, SQL Server, Oracle)	Enables or disables the check constraint.
	Not for Replication (SQL Server)	Enabling this feature ensures that the constraint is not enforced if the table is replicated for another database.
	Check Condition (DB2 LUW, DB2 z/OS, Interbase/Firebird, SQL Server, Oracle, Sybase)	Type the check constraint condition. As a convenience, use the **Table Columns** button to paste in column names as part of the condition.
Comment (DB2 LUW)	Lets you type a comment.	
DDL View	Preview the DDL generated by your choices. Finally, use the **Execute** button to create the check constraint.	

ADD/MODIFY LOGIN TRIGGER

This action lets you build and submit an sp_modifylogin call that specifies a login script option. The target login trigger can be any stored procedure located on the user's default database, and the procedure will run each time the user successfully logs in. The action can be used to:

- Add a login trigger to a login that currently has no login trigger configured

- Change the specified stored procedure for a login that already has a login trigger configured

> **NOTE:** This functionality is available for Sybase.

To specify the stored procedure used as a login trigger for a login:

1 Expand the **Logins** node. Rapid SQL displays all logins for the datasource.

2 Right-click a login and select **Login Trigger Actions > Add/Modify Login Trigger** from the context menu. Rapid SQL opens the **Add/Modify Login Trigger** dialog box.

3 Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	From the Login Procedure dropdown, choose the default database stored procedure that is to be configured as the login trigger for this login.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see Dependencies.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.

After executing this action, the specified stored procedure name is available in the Login editor. For details, see Logins Editor (Sybase).

For related information, see Drop Login Trigger.

ALLOCATE EXTENT

> **NOTE:** This functionality is available for Oracle only.

The Allocate Extent dialog box lets you explicitly allocate extents for clusters, tables, and indexes in Oracle. Though Oracle dynamically allocates extents when additional space is required, explicit allocation of additional extents can be useful if you know that an object grows.

Explicit allocation of additional extents can be particularly helpful when using Oracle Parallel Server. When using Oracle Parallel Server and allocating additional extents, you can allocate an extent explicitly to a specific instance in order to minimize contention for free space among multiple instances.

Important Notes

For composite-partitioned tables, you can allocate extents to subpartitions as well as partitions.

To allocate an extent for a cluster, table, or index

1   Initiate an **Allocate Extent** action against a table, cluster, index, primary key, or unique key. For more information see <u>Initiating an object operation</u>.

2   Use the following table as a guide to setting the options on this dialog:

Option	Describe
Extent Size	Specify an extent size in KBs or MBs.
Datafile	Lets you select the new datafile. You can choose a specific datafile from which to take space for the added extent. If you choose (Default), Oracle takes the space from any accessible datafile in the tablespace containing the table, index, or cluster.
Instance	Lets you specify a freelist from which to draw the extent. If you are using Oracle Parallel Server, you can assign the new extent to a free list group associated with a specific instance. The number you enter in the Instance text box should be the number of the freelist group that you wish to use, rather than the number of the specific instance. If you are using Oracle Parallel Server and you omit this parameter, Oracle allocates the extent, but the extent is drawn from the master freelist by default. Only use this parameter for Oracle Parallel Server. NOTE: The number you enter in the Instance field should be the number of the free list group that you wish to use, rather than the number of the specific instance.

3   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

ANALYZE

This action lets you work with statistics for tables, indexes (primary keys and unique keys), and clusters. When using this action, Rapid SQL lets you build and submit the following ANALYZE statement variations or DBMS_STATS package procedure calls:

Rapid SQL options	ANALYZE statement variation or DBMS_STATS call generated
Compute Statistics	ANALYZE INDEX/CLUSTER/TABLE... COMPUTE STATISTICS
Gather Statistics	DBMS_STATS.GATHER_INDEX_STATS DBMS_STATS.GATHER_TABLE_STATS,
Delete Statistics	ANALYZE INDEX/CLUSTER/TABLE... DELETE STATISTICS DBMS_STATS.DELETE_INDEX_STATS DBMS_STATS.DELETE_TABLE_STATS
Estimate Statistics	ANALYZE INDEX/TABLE/CLUSTER... ESTIMATE STATISTICS
Validate Structure	ANALYZE INDEX/TABLE/CLUSTER... VALIDATE STRUCTURE
Import Statistics	DBMS_STATS.IMPORT_INDEX_STATS DBMS_STATS.IMPORT_TABLE_STATS
List Chained Row Into Table	ANALYZE TABLE/CLUSTER... LIST CHAINED ROWS INTO

> **NOTE:** Before using this object action, consult Oracle documentation for information on the ANALYZE statement and relevant DBMS_STATS package procedures. Be familiar with restrictions, required permissions, partition options, parameter values, and requirements. For more information, see <u>Accessing Third Party Documentation</u>.

To obtain or modify statistics for a database object:

1   Initiate an **Analyze** action against one or more clusters, indexes, primary keys, unique keys, or tables. For more information see <u>Initiating an object operation</u>.

2   Use the following table as a guide to understanding and modifying settings in the **Analyze** wizard:

Step	Settings and tasks
Action Options	Take the following steps:
	If you are not working against a cluster, either select the **Use dbms_stats_package** check box to generate one of the DBMS_STATS procedure calls or deselect the **Use dbms_stats_package** check box to generate an ANALYZE statement variation.
	From the **Type** dropdown, select one of the Rapid SQL options from the table above.
	Additional options are offered depending on the **Type** value you choose. They correspond to ANALYZE statement clauses or DBMS_STATS procedure parameters. See Oracle documentation for details.
Preview	Displays the DDL that will execute the object action. For details, see <u>Preview</u>.

3   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

ANALYZE TABLES

Rapid SQL lets you issue an ANALYZE TABLE statement, analyzing and storing the key distribution for a table.

> **NOTE:**   This functionality is available for MySQL only.

To analyze a table:

1   On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2   Select the **Tables** node.

Rapid SQL displays the tables in the right pane of the Explorer window.

3   In the right pane of the application, select one or more tables.

4   From the Command menu select Analyze Tables.

Rapid SQL opens the Analyze Tables dialog box.

5   Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Lets you review the tables you selected.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see <u>Dependencies</u>.
Preview	Displays the DDL that will execute the object action.

6   Click **Execute**. For information on the other options, see Preview and Scheduling.

ATTACH DATABASE

The **Attach Database** object action lets you specify a set of previously detached data and transaction files to be used to attach a database to a Microsoft SQL Server instance. This makes the database available in exactly the same state it was in when it was detached

> **NOTE:**   This functionality is available for Microsoft SQL Server 2000 or later only.

To attach a database

1   Initiate an **Attach Database** action against a database. For more information see Initiating an object operation.

Rapid SQL opens the **Attach Database File(s)** dialog box.

2   In the **Database Name to be Attached** box, type the database name.

3   In the grid, do one of the following:

- Select the target database file(s).

- To add database file(s), click **Add** and then enter the name of the MDF (master data file) of the database to attach.

Rapid SQL automatically adds the appropriate *.ldf file.

4   To drop database file(s), click Drop and then select the target file(s).

5   Click **Execute**. For information on the other options, see Preview and Scheduling.

For related information, see Detach Database

BIND PACKAGE

The Bind Package Wizard lets you set package parameters, add environments, and set package properties.

> **NOTE:**   This functionality is available for IBM DB2 for z/OS only.

To package parameters, add environments, and set package properties:

1   In the explorer, under the **Schema** node, expand an owning schema, right-click **Packages**, and select **Bind** from the context menu.

Rapid SQL opens the **Bind Package** wizard.

2 Use the following table as a guide to understanding and setting options on this wizard:

Step	Settings and tasks	
1	**Location**	Lets you select the name of the location to connect to.
	Collection	Lets you select the location of the DBMS where the package binds and where the description of the package resides.
	Would you like to Copy an Existing Package from Another Collection	
	New Package (available if you select **No** for the **Would you like**... control)	Lets you select a Package Name and PDS Name.
	Copy Package (available if you select **Yes** for the **Would you like**... control)	Lets you qualify with the following: **Collection** - the collection to copy from. **Options** - COMMAND or COMPOSITE. **Package** - Lets you select a package. **Version** - Lets you select a version of the package.
2	Lets you select a package **Owner**, a **Qualifier** (the package creator), an **Action** of ADD or REPLACE, and the **Version** of the package.	

Step	Settings and tasks	
3	**Isolation**	Determines how far to isolate an application from the effects of other running applications.
	Keep Dynamic	Specifies that DB2 keeps dynamic SQL statements after commit points. The application does not need to prepare an SQL statement after every commit point. DB2 keeps the dynamic SQL statement until the application process ends, a rollback operation occurs or the application executes an explicit PREPARE statement with the same statement identifier.
		If the prepared statement cache is active, DB2 keeps a copy of the prepared statement in the cache. If the prepared statement cache is not active, DB2 keeps only the SQL statement string past a commit point. DB2 then implicitly prepares the SQL statement if the application executes an OPEN, EXECUTE, or DESCRIBE operation for that statement.
	Current Data	Determines whether to require data currency for read-only and ambiguous cursors when the isolation level of cursor stability is in effect. It also determines whether block fetching can be used for distributed, ambiguous cursors.
	Degree	Determines whether to attempt to run a query using parallel processing to maximize performance. Lets you select an option.
	DB Protocol	Specifies which protocol to use when connecting to a remote site that is identified by a three-part name statement.
	Dynamic Rules	Determines what values apply at run time for the following dynamic SQL attributes:
		The authorization ID that is used to check authorization
		The qualifier that is used for unqualified objects
		The source for application programming options that DB2 uses to parse and semantically verify dynamic SQL statements
		Whether dynamic SQL statements can include GRANT, REVOKE, ALTER, CREATE, DROP, and RENAME statements
	Release	**Commit** - Releases resources at each commit point.
		Deallocate - Releases resources only when the program terminates.
	Validate	Determines whether to recheck, at run time, errors found during bind. The option has no effect if all objects and needed privileges exist.
		Bind - If not all objects or needed privileges exist at bind time, the wizard displays an error messages, and does not bind the package.
		Run - If not all objects or privileges exist at bind time, the process issues warning messages, but the bind succeeds. DB2 checks existence and authorization again at run time for SQL statements that failed those checks during bind. The checks use the authorization ID of the package owner.

Step	Settings and tasks	
4	**Explain**	Obtains information about how SQL statements in the package are to execute, and then inserts that information into the table owner.PLAN_TABLE, where owner is the authorization ID of the owner of the plan or package. This option does not obtain information for statements that access remote objects.
	Optimization Hint	Query optimization hints are used for static SQL.
	Reoptvars	Re-determines the access path at run time.
	Encoding	Lets you select type of language for the package.
	Defer Prepare	Optionally, prepares dynamic SQL statements that refer to remote objects. Lets you choose from DEFER(PREPARE) and NODEFER(PREPARE).
	Degree	1 or ANY.
	Page Writes	NO, PH1, or YES.
	Flag	Lets you select flag for messages to display: all (default), completion, error or warning, and informational.
5	Use the arrow buttons to move system connections between the **Disabled Environments** and **Enabled Environments** lists.	

3   When ready, click **Finish**.

BIND TO TEMPORARY DATABASE

This action builds and submits a **sp_tempdb bind** system procedure call. It lets you bind a login to a temporary database.

> **NOTE:**   This functionality is available for Sybase only.

To bind a login to a temporary database

1   Initiate a **Bind To Temporary Database** action against one or more logins. For more information see <u>Initiating an object operation</u>.

2   Use the following table as a guide to understanding and modifying settings in the **Bind To Temporary Database** wizard:

Step	Settings and tasks
Action Options	Select the target temporary database from the **Bind To Temporary Database** dropdown.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see <u>Dependencies</u>.
Preview	Displays the DDL that will execute the object action. For details, see <u>Preview</u>.

3   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

BUILD

This action lets you build a procedure, specifying a **Build utility function** (ALTER_REBUILD, REBUILD, REBIND, or DESTROY), a **Build owner**, **Precomplie options**, **Compile options**, **Prelink options**, **Link options**, and **Bind options**.

> **NOTE:** This functionality is available for DB2 z/OS only.

To build a procedure

1   Initiate a **Build** action against a procedure. For more information see <u>Initiating an object operation</u>.

2   Type or select the build options in the **Procedure Build Options** dialog:

3   Click **Build**.

BUILD QUERY

Right-clicking a table and selecting **Build Query** opens a tool that lets you create, structure, and manipulate queries. For details, see <u>Query Builder</u>.

CHANGE ACCESS STATUS

This action builds and submits a basic ALTER TABLE... READ ONLY/READ WRITE statement. This lets you toggle the table between READ ONLY and READ/WRITE modes.

> **NOTE:** This functionality is available as of Oracle 11g.

> **NOTE:** The Browser listing for an Oracle table has a **Read-Only** field that displays the current status for a table. For more information, see <u>Browsers</u>.

To change between READ ONLY and READ WRITE modes for a table

1   Initiate a **Change Access Status** action against one or more tables. For more information, see <u>Initiating an object operation</u>.

2   Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action options	Select the **Read-only** check box to generate a ALTER TABLE... READ ONLY statement, putting the table into read-only mode. Deselect the **Read-only** check box to generate a ALTER TABLE... READ WRITE statement, putting the table into read-write mode.
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

CHANGE CATEGORY

Rapid SQL lets you change the category of a target Stored Outline.

> **NOTE:** This functionality is available for Oracle only.

To change the category of a stored outline

1 Initiate a **Change Category** action against a stored outline. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying settings in the **Change Category** wizard:

Step	Settings and tasks
Action options	From the Category dropdown, choose a new, target category.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Displays the DDL that will execute the object action. For details, see [Preview](#).

3 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

For related information, see the following topics:

- [Drop By Category](#)

- [Reassign By Category](#)

CHANGE PASSWORD

> **NOTE:** This functionality is available for Microsoft SQL Server, MySQL, Oracle, and Sybase ASE only.

The Change Password dialog box lets you change user or login passwords, which you should do on a regular basis to guard against security leaks.

To change the password for a user or login

1 Take one of the following actions:

- For Sybase or SQL Server, expand the Logins node.

- For MySQL and Oracle, expand the Users node.

2 Right-click the user or login, and then select **Change Password** from the context menu.

Rapid SQL opens the **Change Password** dialog box.

3 For Oracle and SQL Server datasources, in the **Old Password** box, type the old password. Providing the old password is optional on SQL Server datasources.

4 In the **New Password** box, type the new password.

5   In the **Confirm Password** box, type the new password.

6   Click **Execute**. For information on the other options, see Preview and Scheduling.

CHANGE STATUS

You can change the status of a number of relevant object types. For details, see the following topics:

- Change Status (InterBase Triggers)
- Change Status (SQL Server and Sybase triggers)
- Change Status (SQL Server DDL triggers)
- Change Status (Full-text Indexes)
- Change Status (check constraints)
- Change Status (tablespaces)

CHANGE STATUS (INTERBASE TRIGGERS)

This action lets you build an ALTER TRIIGGER statement with an ACTIVE or INACTIVE argument, and issue the statement against one or more triggers.

> **NOTE:**   This functionality is available for InterBase/Firebird.

To change the status of a trigger:

1   On the *Datasource Explorer*, expand the Triggers node.

2   Select one or more triggers, right-click the selected triggers, and then select Change Status from the context menu.

Rapid SQL opens the **Change Status** dialog.

3   Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action options	With **Enabled** selected, the ALTER TRIGGER statement is created with an ACTIVE argument. Deselected, the ALTER TRIGGER statement is created with an INACTIVE argument.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.

4   Click **Execute**. For information on the other options, see Preview and Scheduling.

CHANGE STATUS (SQL SERVER AND SYBASE TRIGGERS)

The **Change Status** object action lets you build an issue an ALTER TABLE statement that specifies a DISABLE TRIGGER or ENABLE TRIGGER option. This lets you enable or disable any triggers.

> **NOTE:** This functionality is available for Microsoft SQL Server and Sybase ASE only.

Loading a database from a previous dump causes any triggers defined in the database to fire. To speed the time required to load a database you should disable triggers.

> **NOTE:** Disabling triggers can lead to problems with maintaining referential integrity and business rules

To enable or disable a trigger:

1   Initiate a **Change Status** action against one or more triggers. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Change Status** wizard:

Step	Settings and tasks
Action Options	Select the **Enabled** check box to enable the trigger. Deselect it to disable the trigger.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

CHANGE STATUS (SQL SERVER DDL TRIGGERS)

The **Change Status** object action lets you build and issue a DISABLE TRIGGER... ON DATABASE or ENABLE TRIGGER... ON DATABASE statement. This lets you enable or disable one or more database triggers.

> **NOTE:** This functionality is available for SQL Server 2005 and up.

> **NOTE:** The Browser listing for a database trigger indicates its current status. For more information, see Browsers.

To enable or disable a trigger:

1   Initiate a **Change Status** action against one or more triggers. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Change Status** wizard:

Step	Settings and tasks
Action Options	Select the **Enabled** check box to enable the trigger. Deselect it to disable the trigger.

Step	Settings and tasks
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

CHANGE STATUS (FULL-TEXT INDEXES)

The action builds and submits an ALTER FULLTEXT INDEX statement with an ENABLE or DISABLE argument, letting you enable or disable the full-text index. Full-text searching is unavailable on tables or views with a disabled full-text index.

>   **NOTE:**   This functionality is available as of SQL Server 2005.

To enable or disable a full-text index:

1   Initiate a **Change Status** action against one or more full-text indexes. For more information, see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action Options	Select the **Enabled** check box to enable the full-text index. Deselect the **Enabled** check box to disable the full-text index. Note that on opening the dialog, the **Enabled** check box state reflects whether the index is currently enabled or disabled.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

CHANGE STATUS (CHECK CONSTRAINTS)

Rapid SQL lets you change the enabled/disabled status of check constraints, foreign key constraints, primary key constraints, and unique key constraints.

For detailed instructions, see:

- Setting constraint status for Microsoft SQL Server objects

- Setting Constraint Status for Oracle Objects

SETTING CONSTRAINT STATUS FOR MICROSOFT SQL SERVER OBJECTS

The Set Constraint Status dialog box lets you specify the ability of a group of constraints to be replicated, and (for Microsoft SQL Server version 7 or later) enable or disable check constraints, foreign key constraints, primary key constraints, and unique key constraints.

Completing the Set Constraint(s) Status Dialog Box

To complete this dialog box, do the following:

1 Initiate a **Change Status** action against one or more check constraints. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying the settings on the **Change Status** wizard:

Step		Functionality
Action Options	**Enabled**	Deselect to temporarily override listed check constraints. Useful when you need to execute special processes that would ordinarily incur constraint-related errors.
	Not for Replication	When you duplicate the table schema and data of a source database containing constraints marked "Not for Replication", these objects are not carried over to the duplicate of the schema.
Dependencies		View the dependencies on the tablespace.
Preview		Preview the DDL generated from your choices.

3 Use the **Execute** or **Schedule** button to complete the operation.

SETTING CONSTRAINT STATUS FOR ORACLE OBJECTS

The Set Constraint(s) Status dialog box lets you change the status of check constraints, foreign key constraints, primary key constraints, and unique key constraints. Rapid SQL lets you enable or disable selected constraints and, in the case of primary key and unique key constraints, lets you enable with or without validation and disable with or without the changes cascading.

When enabled, the rule defined by the constraint is enforced on the data values in the columns on which the constraint is placed. When disabled, the constraint rule is not enforced but the constraint continues to be stored in the data dictionary.

Temporarily disabling constraints can improve performance when you are loading large amounts of data or when you are making massive changes to a table. Disabling constraints also can be useful if you are importing or exporting one table at a time.

> **NOTE:** Primary keys for index-organized tables cannot be disabled.

> **NOTE:** You cannot drop a unique or primary key constraint that is part of a referential integrity constraint without also dropping the foreign key. To drop the referenced key and the foreign key together, select the Cascade check box in the Set Constraint(s) Status dialog box.

The table below describes the options and functionality on the Set Constraint(s) Status dialog box.

> **NOTE:** The options differ by object.

Option	Description
Enable	Enabling the constraint and not selecting the Validate check box automatically uses Oracle ENABLE NOVALIDATE clause which enables a constraint so that it does not validate the existing data. A table using constraints in enable novalidate mode can contain invalid data but you cannot add new invalid data to that table.The enable novalidate mode is useful as an intermediate state or when you do not want the constraint to check for possible exceptions (e.g., after a data warehouse load).
Validate	Enabling the constraint and selecting the Validate check box causes Oracle to validate all existing data in the key columns of the table with the constraint. If an exception exists, Oracle returns an error and the constraint remains disabled.
Cascade	Selecting the Cascade check box when disabling a primary key or foreign key constraint instructs Oracle to simultaneously disable any constraints that depend on the primary or unique key. Selecting the Delete Cascade check box instructs Oracle to delete data in the child table (on which the foreign key is defined) if the referenced data is the parent table is deleted.

Completing the Set Constraint(s) Status Dialog Box

To complete this dialog box, do the following:

1 On the *Datasource Explorer*, select the target object node.

    Rapid SQL displays the target objects in the right pane of the Explorer window.

2 In the right pane of the application, select one or more check constraints.

3 From the Command menu select Change Status.

    Rapid SQL opens the **Change Status** dialog box.

4 Use the table above to select dialog box options.

5 Use the **Execute** or **Schedule** button to complete the operation.

CHANGE STATUS (TABLESPACES)

You can change the online, offline, or read only status of a tablespace to control access to its segments. In addition, when setting a tablespace offline, you can choose between NORMAL, TEMPORARY, or IMMEDIATE modes of taking the tablespace offline.

> **NOTE:** This functionality is available for Oracle only.

To change the status of a tablespace

1 Initiate a **Change Status** action against one or more tablespaces. For more information see Initiating an object operation.

2 Use the following table as a guide to understanding and modifying settings in the **Change Status** wizard:

Step	Settings and tasks
Action options	From the **Change Status** dropdown, select ONLINE, OFFLINE, or READ ONLY. If you select OFFLINE, from the **OfflineMode** dropdown, select NORMAL, TEMPORARY, or IMMEDIATE.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.. For information on the other options, see Preview and Scheduling.

CHECK TABLES

Rapid SQL lets you issue a CHECK TABLE statement, checking one or more tables for errors. This operation returns a standard CHECK TABLE result set. Each returned message consists of **Table** (name), **Op** (with a value of CHECK), **Msg_type** (STATUS, ERROR, INFO, WARNING), and **Msg_text** components.

> **NOTE:** This functionality is available for MySQL only.

To check a table for errors:

1 On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2 Select the **Tables** node.

Rapid SQL displays the tables in the right pane of the Explorer window.

3 In the right pane of the application, select one or more tables.

4 From the Command menu select Check Tables.

Rapid SQL opens the **Check Tables** dialog box.

5 Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks	
Action Options	**Changed**	Only check tables that have changed since the last check.
	Extended	Perform a full key lookup for all keys in each row.
	Fast	Check only tables that have not been properly closed.
	Medium	Scan rows to verify that deleted links are valid. Calculate a key checksum for the rows and verify using a calculated checksum for keys.
	Quick	Perform a full key lookup for all keys in each row.
Preview	Displays the DDL that will execute the object action.	

6   Click **Execute**. For information on the other options, see Preview and Scheduling.

The results set opens on the **Results** tab of an ISQL editor window. For more information, see Results Editor.

CHECKPOINT

Running a Checkpoint lets you force all dirty pages for the current database to be written to disk. A dirty page is any data or log page which, after being read into the buffer cache, is modified but not yet written to disk. The Checkpoint statement saves time in a subsequent recovery by creating a point at which all modifications to data and log pages are guaranteed to have been written to disk. If the current database is in log-truncate mode, CHECKPOINT also truncates the inactive portion of the log.

> **NOTE:**   This functionality is available for Microsoft SQL Server, and Sybase ASE only.

Important Notes

The default permission for executing a checkpoint is the db_owner fixed database role.

To run a Checkpoint against one or more databases

1   Initiate a **Checkpoint** action against one or more databases. For more information see Initiating an object operation.

2   Use the following table as a guide to working through the panels of the **Checkpoint** dialog box

Step	Description
Action options	Displays the names of the database(s) you chose.
Preview	Displays the DDL generated to execute the Checkpoint operation.

3   Use one of the **Schedule** or **Execute** buttons to execute the Checkpoint.

CHECKSUM TABLES

Rapid SQL lets you issue a CHECKSUM TABLE statement, returning a live checksum.

> **NOTE:**   This functionality is available for MySQL only.

To check a table for errors:

1   On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2   Select the **Tables** node.

Rapid SQL displays the tables in the right pane of the Explorer window.

3   In the right pane of the application, select one or more tables.

4   From the Command menu select Checksum Tables.

Rapid SQL opens the **Checksum Tables** dialog box.

5   Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks	
Action Options	**Extended**	Specifies that the whole table is to be read, row by row.
	Quick	Specifies that the live checksum is to be reported. For information on enabling a live checksum for a table, see [Tables wizard (MySQL)]().
Preview	Displays the DDL that will execute the object action.	

6   Click **Execute**. For information on the other options, see [Preview]() and [Scheduling]().

The results set opens on the **Results** tab of an ISQL editor window. For more information, see [Results Editor]().

COALESCE

You can maximize the size of free space chunks in tablespaces to avoid the situation in which an object cannot acquire enough contiguous free space to accommodate its next extent size. Towards this goal, look for opportunities to coalesce adjacent blocks of free space into a single, larger block.

Oracle automatically coalesces adjacent free space chunks with a background process. However, it still supports the commands for coalescing free space manually. Depending on the size of the tablespace, coalescing its free space can take a long time. So determine when to perform this operation. It you coalesce immediately, Rapid SQL locks the tablespace.

Important Notes

• You cannot coalesce on an UNDO tablespace.

To coalesce a tablespace

1   Initiate a **Coalesce** action against one or more tablespaces. For more information see [Initiating an object operation]().

2   Use the following table as a guide to understanding and modifying settings in the **Coalesce** wizard:

Step	Settings and tasks
Action options	Verify display of the tablespaces to be coalesced.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.. For information on the other options, see [Preview]() and [Scheduling]().

COMPILE

Rapid SQL lets you recompile specific objects by issuing the proper ALTER statement. The explicit recompilation of invalid objects eliminates the need for implicit run-time recompilation which, in turn, can cause run-time compilation errors and performance overhead. Recompile objects after you make changes to that object or dependent objects.

The following table lists the object types by DBMS for which the Compile operation is available:

DB2 LUW						Procedures			
Oracle	Functions	Java Sources	Materialized Views	Packages	Package Bodies	Procedures	Schema	Types and Type Bodies	Views

> **NOTE:** The objects you can compile and the steps in compiling them differ according to the type of data source you are working against. Before proceeding, make sure you are familiar with the material in [Compiling Oracle Objects](#).

To compile an object

1 Initiate a **Compile** action against one or more supported objects (see the table above). For more information see [Initiating an object operation](#).

Rapid SQL opens the **Compile** wizard. Options offered on the **Action Options** tab differ by object type. For example, no options are provided when compiling Java Sources, while a full range of dependent object-based and debug options are provided when compiling procedures.

2   Use the following table as a guide to working through the panels of the dialog box

Step	Settings and tasks		Availability
Action options	**Compile dependents**	If enabled, this option compiles statements for all objects referenced by the object being compiled. For example, if you compile a function that references a specific procedure and you select to compile the dependent objects, an ALTER COMPILE statement is created for that referenced procedure. If disabled, only the current object is compiled and the object's dependencies are ignored. This is the default setting.	DB2 LUW Procedures Oracle Functions, Packages, Procedures, Types, and Views
	Compile only invalid dependents	If you enabled the **Compile dependents** option, enabling this option compiles only invalid dependent objects - Creates ALTER COMPILE statements for only those objects that are currently invalid.	DB2 LUW Procedures Oracle Functions, Packages, Procedures, Types, and Views
	Compile system dependents	If you enabled the **Compile dependents** option, enabling this option compiles dependent system objects - Compiles all of the referenced objects with the debug option.	DB2 LUW Procedures Oracle Functions, Packages, Procedures, Types, and Views
	Compile with debug	Enabling this option instructs the Oracle PL/SQL compiler to generate and store the code for use in debugging sessions.	DB2 LUW Procedures Oracle Functions, Packages, Package Bodies, Procedures, Types, Type Bodies, and Views
	Keep specific name	When enabled, compilation will keep the current name.	DB2 LUW Procedures
Dependencies	Lets you review any dependencies before you proceed.		
Preview	Displays the DDL generated to execute the Checkpoint operation.		

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

COMPILING ORACLE OBJECTS

To recompile an object it must belong to your schema or you need ALTER ANY privileges on that object. You must also have appropriate privileges for any associated objects. Prior to compiling objects of a particular type, see the relevant topic below:

- Notes on Compiling Oracle Functions

- Notes on Compiling Oracle Java Sources

- Notes on Compiling Oracle Materialized Views

- Notes on Compiling Oracle Packages and Package Bodies

- Notes on Compiling Oracle Procedures

- [Notes on Compiling Oracle Types and Type Bodies](#)

- [Notes on Compiling Oracle Views](#)

NOTES ON COMPILING ORACLE FUNCTIONS

Rapid SQL lets you recompile a function. Oracle first recompiles any invalid objects on which the function depends. In addition, it marks any objects that depend on the function as invalid.

To recompile a function that is part of a package, compile the package itself. Rapid SQL uses the ALTER FUNCTION statement to compile a stand-alone function. However, you should not use the ALTER FUNCTION statement to individually recompile a function that is part of a package.

For more information, see [Compile.](#)

NOTES ON COMPILING ORACLE JAVA SOURCES

Oracle lets you compile a Java source. Oracle resolves references to other Java sources.

For more information, see [Compile.](#)

NOTES ON COMPILING ORACLE MATERIALIZED VIEWS

Rapid SQL lets you compile materialized views. If a materialized view fails to revalidate after you recompile, that materialized view cannot be fast refreshed ON DEMAND or used for query rewrite.

For more information, see [Compile.](#)

NOTES ON COMPILING ORACLE PACKAGES AND PACKAGE BODIES

Rapid SQL lets you recompile a package, and recompiles all package objects together.

Recompiling a package in Rapid SQL compiles both the package specification and the package body by issuing two consecutive ALTER statements. However, Rapid SQL shows only the ALTER statement for the specification in the Preview: Confirm Compile dialog box. You can recompile only the package body by explicitly compiling the package body itself.

When recompiling the entire package, Oracle recompiles the package even if it is invalid. However, if there are compilation errors, the package remains invalid and Oracle invalidates all dependent objects.

Recompiling only a package body does not invalidate objects that depend upon the package specification regardless of whether or not the package body has compilation errors.

For more information, see [Compile.](#)

NOTES ON COMPILING ORACLE PROCEDURES

Rapid SQL lets you compile a procedure that is part of a package, by compiling the package itself. Rapid SQL uses the ALTER PROCEDURE statement to compile a stand-alone procedure. However, you should not use the ALTER PROCEDURE statement to individually recompile a procedure that is part of a package.

For more information, see [Compile.](#)

NOTES ON COMPILING ORACLE TYPES AND TYPE BODIES

Rapid SQL lets you recompile a type. Rapid SQL recompiles both the type specification and the type body.

For more information, see [Compile.](#)

NOTES ON COMPILING ORACLE VIEWS

Rapid SQL lets you recompile a view when you have altered an object referenced by that view.

When you recompile a view, Oracle invalidates all dependant objects.

For more information, see [Compile.](#)

CONVERT TABLES

This action lets you build and submit an ALTER TABLE statement, specifying an ENGINE= option, specifying any available storage engine.

> **NOTE:** This functionality is available for MySQL only.

> **NOTE:** For information on how to quickly convert a table to use the INNODB storage engine, see [Rebuild Table](#).

To change the storage engine for a table:

1 On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2 Select the **Tables** node.

   Rapid SQL displays the tables in the right pane of the Explorer window.

3 In the right pane of the application, select one or more tables.

4 Right-click the table or tables and select Convert Tables from the context menu.

   Rapid SQL opens the Convert Tables dialog box.

5 Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Use the **Convert To** dropdown to select an available storage engine.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Displays the DDL that will execute the object action.

6 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

COPY OBJECT NAMES

NOTE: This functionality is available for Microsoft SQL Server and Oracle only.

The Copy Object Names functionality lets you copy and then paste object name(s) into other applications.

To copy an object name to be pasted into another application

1 On the left pane of the *Datasource Explorer*, select the node corresponding to the object type whose name you want to copy.

Rapid SQL displays objects of that type in the right pane of the Explorer window.

2 Right-click on the specific database object and select **Copy**.

COPY SCHEMA

For IBM DB2 for Windows, Unix, and Linux (version 9), the Copy Schema function lets you quickly make copies of a database schema and its corresponding objects.

To copy a schema

1 In the data sources pane, right-click a data source and select **Copy Schema** from the context menu.

2 Use the following table as a guide to understanding and choosing options the options in the Copying Schema dialog:

Pane	Options
Action Options	Lets you specify **Copying Schema**, **Target Schema**, specify a **Copy Mode** of COPY, COPYNO, or DDL, specify a **Log filename** (COPY and COPYNO modes only), select an **Object Owner**, and **Error table schema**, provide an **Error table name**, and specify a **Drop error table after action execution** options.
Tablespace mapping	Lets you specify source and target tablespace options.
Preview	Displays the DDL generated to execute the operation.

3 **Execute** or **Schedule** the action.

CREATE ALIAS

The **Create Alias** object action lets you create an alias to a selected object. The following table shows the object types for which this action is available:

	Materialized Query Tables	Sequences	Tables	Views
DB2 LUW	X	X	X	X
DB2 z/OS			X	X

To create an alias to one of the supported object types:

1   Initiate a **Create Alias** action against a supported object (see the table above). For more information see [Initiating an object operation](#).

Rapid SQL opens the Alias Editor. For information on using the Alias editor, see the topic corresponding to the DBMS you are working against:

- [Alias Wizard (DB2 LUW)](#)

- [Alias Wizard (DB2 Z/OS)](#)

CREATE CLONE

Rapid SQL provides support for the ADD CLONE option functionality, letting you create a clone on an existing base table. This creates a table with a cloned definition but with no data.

> **NOTE:**   This functionality is available as of IBM DB2 for z/OS version 9. The base table must reside on a universal tablespace. For more information, see [Tablespace Wizard (DB2 Z/OS)](#).

To create a clone for one or more tables:

1   Initiate a **Create Clone** action against one or more tables. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings in the **Create Clone** wizard:

Step	Settings and tasks
Action Options	Lets you provide a **Clone Name**.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Displays the DDL that will execute the object action.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).
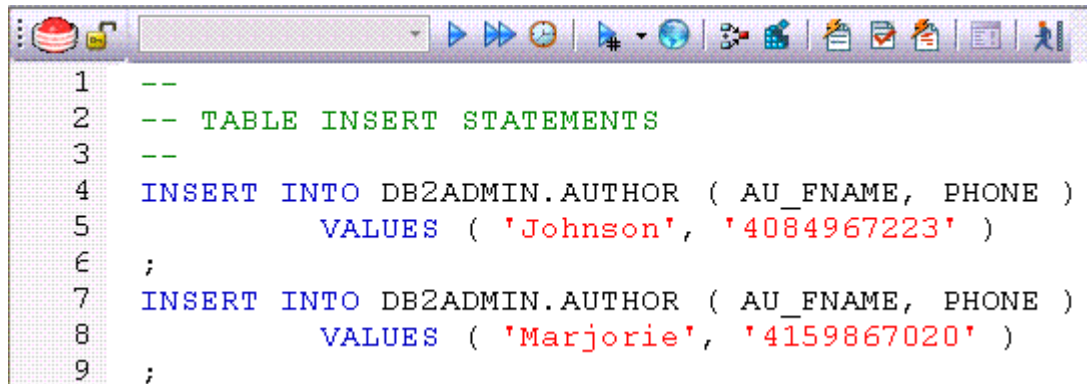
The clone for a table is not displayed in the Datasource Explorer listing of tables for a data source. To determine whether a table has a clone, see the **Dependencies** tab/panel on a Table editor or on the wizard for object actions such as Drop or Rename.

After creating a clone for a table, you can exchange data between the base table and the clone. For related information, see the following topics:

- [Exchange Data With Clone](#)

- [Drop Clone](#)

CREATE INSERT STATEMENTS

The Create Insert Statements dialog box lets you automatically create a set of Insert Statements for selected columns and records and based on the data in a table.

> **NOTE:** This functionality is available for all DBMS platforms.

To create Insert Statements based on existing table data

1 In the left pane of the application, select the Tables node.

   Rapid SQL displays the Tables in the right pane of the Explorer window.

2 In the right pane of the application, right-click a table, and then select Create Insert Statements.

   Rapid SQL opens the **Create Insert Statements** dialog box.

3 In Columns, select columns to be included in the INSERT statement.

4 Optionally, in the Where box, type a WHERE clause.

5 Enable or disable **Include owner information in insert statements**.

6 Enable or disable **Set row count**, which lets you specify a row count, the number of rows in a table that were affected by the Insert statement executed against the table, or a view based on the table.

7 Click OK.

An ISQL window opens with a set of Insert statements corresponding to the criteria you provided and the table data. For more information, see ISQL Windows.

CREATE LIKE

The Create Like Editor lets you create a new object based on an existing object. The following table shows availability of the **Create Like** function for particular object types by DBMS:

DB2 LUW	DB2 Z/OS	ITB/FBD	MySQL	SQL Server	Oracle	Sybase
Materialized Query Tables Tables	Tables	Tables	Tables	Logins Tables Users	Tables	Logins Tables Users

To create an object based on another object

1   Initiate a **Create Like** action against a table. For more information see Initiating an object operation.

Rapid SQL opens a Create Like wizard.

2   Provide a name for the new object.

3   Modify settings on each panel of the wizard as required.

For particular object types, tasks and settings on the panels of the Create Like wizard are similar to those for the object editors for that object type. For information, see:

- Materialized Query Tables Editor (IBM DB2 LUW)
- Tables Editor (IBM DB2 LUW)
- Tables Editor (IBM DB2 Z/OS)
- Tables editor (InterBase/Firebird)
- Tables editor (MySQL)
- Logins Editor (SQL Server)
- Tables Editor (SQL Server)
- Users Editor (SQL Server)
- Tables Editor (Oracle)
- Logins Editor (Sybase)
- Tables Editor (Sybase)
- Users Editor (Sybase)

4   Click **Execute** to create the new object.

CREATE OR EDIT JAVA SOURCE

> **NOTE:**   This functionality is available for Oracle only.

The Java Editor lets you enter Java code. The table below describes the Java Editor toolbar options:

Option	Description
Lock/Unlock Connection	Click to lock or unlock connection.
Create	Click to open the Create Options dialog box, which lets you select the options for creating the java source.
Errors	Click to split the workspace in half, displaying the error messages in the lower half of the workspace.

When you add a script in a new file, you not only choose a name for the file, but choose among create options. Finally, when you complete your script, you can Preview, Schedule or immediately Execute it.

CREATE SYNONYM

Rapid SQL lets you perform an ad hoc Create Synonym operation against a specific object displayed in the Datasource Explorer. For background information, see Synonyms.

> **NOTE:** This function is available for Oracle.

You can perform this operation against the following object types:

Functions	Materialized Views	Packages	Package Bodies
Procedures	Sequences	Tables	Views

To create a synonym for an object

1   Expand the node corresponding to one of the object types supported for this operation, listed in the table above.

    The right-hand pane is populated with all objects of that type.

2   Right-click the target object and select Create Synonym from the context menu.

3   A **Create Synonym** wizard opens.

For information on using the wizard, see Synonym Wizard (Oracle).

CREATE VIEW

Right-clicking one or more tables in the right pane of the explorer and selecting **Create View** opens a wizard that lets you create a new view. For more information, see the following topics:

- View Wizard (DB2 LUW)

- View Wizard (DB2 Z/OS)

- [Views Wizard (InterBase/Firebird)](#)

- [View Wizard (SQL Server)](#)

- [View Wizard (Oracle)](#)

- [View Wizard (Sybase)](#)

DBCC

Rapid SQL's DBCC (Database Consistency Check) function box lets you:

- Specify single or multiple tables or indexes for validation.

- Perform database-wide validations.

- Perform object-level validations of databases.

    NOTE: This functionality is available for Microsoft SQL Server and Sybase ASE only.

To perform a Database Consistency Check

1   On the left pane of the *Datasource Explorer*, expand the **Databases**, **Tables**, or **Indexes** node.

    Rapid SQL displays objects of that type in the right pane of the Explorer window.

2   In the right pane of the Explorer window, select the specific object.

3   Initiate the DBCC operation using the method appropriate to the DBMS and object type:

- If you are checking consistency of a Sybase or SQL Server database or a SQL Server table, right-click the object and select **DBCC** from the context menu, as appropriate.

- If you are checking consistency of a Sybase table, right-click the table and select either **DBCC > Check Allocation**, **DBCC > Check Table**, **DBCC > Check Text**, or **DBCC > Rebuild Index** from the context menu, as appropriate.

- If you are checking consistency of a Sybase index, right-click the index and select either **DBCC > Check Allocation**, or **DBCC > Check Index** from the context menu, as appropriate.

- If you are checking consistency of a SQL Server index, right-click the table and select either **DBCC > Check Fragmentation**, **DBCC > Update Usage**, **DBCC > Check Index**, **DBCC > Show Statistics**, or **DBCC > Rebuild Indexes** from the context menu, as appropriate.

    Rapid SQL opens a dialog that lets you specify DBCC options.

4   See the following topics as a guide to understanding and setting options on the dialog:

- [DBCC for Microsoft SQL Server Databases](#)
- [DBCC for Microsoft SQL Server Tables](#)
- [DBCC for Microsoft SQL Server Indexes](#)
- [DBCC for Sybase ASE Databases](#)
- [DBCC for Sybase ASE Tables](#)
- [DBCC for Sybase ASE Indexes](#)

5   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

DBCC FOR MICROSOFT SQL SERVER

The DBCC (Database Consistency Check) dialog box lets you specify single or multiple tables or indexes for validation in Microsoft SQL Server. Use this dialog box to perform table-level or index-level validations of databases which are too large to undergo database-level DBCC operations in a time-efficient manner.

The DBCC dialog box includes the following elements:

- A window displaying the target database objects
- A drop-down list of DBCC Operations
- Buttons for previewing the operation's SQL code, scheduling the operation, and executing the operation

For more information, see:

[DBCC for Microsoft SQL Server Databases](#)

[DBCC for Microsoft SQL Server Tables](#)

[DBCC for Microsoft SQL Server Indexes](#)

DBCC FOR MICROSOFT SQL SERVER DATABASES

The DBCC dialog box for databases lets you perform database-wide validations. You should validate your databases as part of regular database maintenance to guard against corruption and failure. Microsoft SQL Server offers a set of DBCC commands to validate the integrity of your databases. Generally, you should perform these DBCC commands prior to dumping your databases to ensure that you are capturing clean backups of your databases.

The fundamental difference between the DBCC dialog box for databases, tables and indexes is the content of the DBCC Operation drop-down list.

The table below describes the options and functionality on the DBCC dialog box.

DBCC Operation	Description
Check Allocation	Executes a DBCC CHECKALLOC command. Makes sure that all data and index panels are correctly allocated and used. It reports on the amount of space allocated and used in the database. When checking allocation, you have the option to skip non-clustered indexes by selecting the Skip non-clustered indexes check box.
Check Catalog	Executes a DBCC CHECKCATALOG command. Checks for consistency in and between system tables.
Check Database	Executes a DBCC CHECKDB command. Verifies that all tables and indexes are properly linked, that indexes are in proper sorted order, that all pointers are consistent, that the data on each panel is reasonable, and that panel offsets are reasonable. When checking a database, you have the option to skip non-clustered indexes by selecting the Skip non-clustered indexes check box.
Check FileGroup	Executes a DBCC CHECKFILEGROUP command. Verifies that all tables and indexes for the specified filegroup are properly linked, that indexes are in proper sorted order, that all pointers are consistent, that the data on each panel is reasonable, and that panel offsets are reasonable. When checking filegroups, you have the option to skip non-clustered indexes by selecting the Skip non-clustered indexes check box.
Show Oldest Transaction	Executes a DBCC OPENTRAN command. Displays information on the oldest active transaction and the oldest distributed and non distributed replicated transactions, if any, within the specified database.
Update Usage	Executes a DBCC UPDATEUSAGE command. Reports and corrects the rows, used, reserved, and dpanels columns of the sysindexes table for any clustered indexes on objects of the type U (user-defined table) or S (system table).

For more information, see [DBCC.](#)

DBCC FOR MICROSOFT SQL SERVER TABLES

The DBCC dialog box for tables lets you perform table-level validations of databases. The fundamental difference between the DBCC dialog box for tables and indexes is the content of the DBCC Operation drop-down list.

The table below describes the options and functionality on the DBCC dialog box.

Option	Description
Check Current Identity Value	Checks the current identity value for the target objects, correcting values if needed depending on parameter specifications. Identity columns created with a NOT FOR REPLICATION clause in either the CREATE TABLE or ALTER TABLE statement are not corrected by this operation.
Check Fragmentation	Displays the target table's data and index fragmentation information, determining whether the table is heavily fragmented. When a table is heavily fragmented, you can reduce fragmentation and improve read performance by dropping and recreating a clustered index (without using the SORTED_DATA option). Doing so reorganizes the data, resulting in full data pages. To adjust the level of fullness, use the Rebuild Index operation's FILLFACTOR option. When INSERT, UPDATE, and DELETE statements fragment tables, they usually do so with unequal distribution across the entire database so that each page varies in fullness over time, forcing additional page reads for queries that scan part or all of a table.

Option	Description
Check Table	Checks the linkages and sizes of text, ntext and image pages for selected tables. For the data, index, text, ntext, and image pages of the target tables, this operation also checks that index and data pages are correctly linked, indexes are in their proper sorted order, pointers are consistent, the data on each page is reasonable, and the page offsets are reasonable. DBCC CHECKTABLE requires a shared lock on all tables and indexes in the database for the duration of the operation. However, DBCC CHECKTABLE does not check the allocations of pages in the specified table (for this, use DBCC CHECKALLOC). To perform DBCC CHECKTABLE on every table in the database, use DBCC CHECKDB.
Check Text/Image Allocation	Checks the allocation of text, ntext, or image columns for a table. In later versions of Microsoft SQL, use DBCC CHECKTABLE to check the integrity of the data, index, text, ntext, and image pages for the target table.
Pin Table	Pins target tables in memory so that they are not flushed when Microsoft SQL Server needs space to read in new pages. DBCC PINTABLE is best used for keeping small, frequently referenced tables in memory. Pinning a large table can consume a large portion of the buffer cache, leaving inadequate memory to service other tables in the system. A pinned table that is larger than the buffer cache itself can fill the entire cache, necessitating a shut down of the system by a sysadmin user, who must then restart Microsoft SQL Server and unpin the table. Pinning too many small tables can result in a similar problem. This option is not available for Microsoft SQL Server 2005.
Rebuild Index	Dynamically rebuilds one, multiple, or all indexes for a table in the target database, allowing indexes which enforce either primary key or unique constraints to be rebuilt without need for dropping and recreating. This operation is not supported for use on system tables.
Unpin Table	Marks target tables as unpinned, rendering their pages flushable from the buffer cache if space is needed to read in a new page from disk. This option is not available for Microsoft SQL Server 2005.
Update Usage	Reports and corrects inaccuracies in the sysindexes table (which can result in incorrect space usage reports by the sp_spaceused system stored procedure) and corrects the rows, used, reserved, and dpages columns of the sysindexes table for tables and clustered indexes. If there are no inaccuracies in sysindexes, DBCC UPDATEUSAGE returns no data. Use this operation to synchronize space-usage counters. Executing this operation on large tables or databases can require some time, so it should typically be used only when you suspect incorrect values returned by sp_spaceused.

For more information, see DBCC.

DBCC FOR MICROSOFT SQL SERVER INDEXES
The DBCC dialog box for indexes lets you perform index-level validations of databases. The fundamental difference between the DBCC dialog box for tables and indexes is the content of the DBCC Operation drop-down list.

The table below describes the options and functionality on the DBCC dialog box.

Option	Description
Check Fragmentation	Displays the target index's table data and fragmentation information, determining whether the table is heavily fragmented. For more information, see tip below.
	When a table is heavily fragmented, you can reduce fragmentation and improve read performance by dropping and recreating a clustered index (without using the SORTED_DATA option). Doing so reorganizes the data, resulting in full data pages. To adjust the level of fullness, use the Rebuild Index operation's FILLFACTOR option. When INSERT, UPDATE, and DELETE statements fragment tables, they usually do so with unequal distribution across the entire database so that each page varies in fullness over time, forcing additional page reads for queries that scan part or all of a table.
Check Index	Checks the linkages and sizes of text and image pages for selected indexes. DBCC CHECKTABLE requires a shared lock on all tables and indexes in the database for the duration of the operation.
Rebuild Index	Dynamically rebuilds all target indexes, allowing those which enforce either primary key or unique constraints to be rebuilt without need for dropping and recreating. This operation is not supported for use on system tables.
Show Statistics	Displays the current distribution statistics for the target indexes. The results returned indicate the selectivity of each target index (a lower density equals a higher selectivity) and provide the basis for determining the usefulness of target indexes to the optimizer.
Update Usage	Reports and corrects inaccuracies in the sysindexes table (which can result in incorrect space usage reports by the sp_spaceused system stored procedure) and corrects the rows, used, reserved, and dpages columns of the sysindexes table for tables and clustered indexes. If there are no inaccuracies in sysindexes, DBCC UPDATEUSAGE returns no data. Use this operation to synchronize space-usage counters. Executing this operation on large tables or databases can require some time, so it should typically be used only when you suspect incorrect values returned by sp_spaceused. Additional Options: Update Index Option, and DBCC General Option.

For more information, see DBCC.

DBCC FOR SYBASE ASE

The DBCC (Database Consistency Check) dialog box lets you specify single or multiple databases, tables or indexes for validation in Sybase ASE. Use this dialog box to perform table-level or index-level validations of databases which are too large to undergo database-level DBCC operations in a time-efficient manner.

The DBCC dialog box includes the following elements:

- A window displaying the target database objects

- A drop-down list of DBCC Operations

- Buttons for previewing the operation's SQL code, scheduling the operation, and executing the operation

For more information, see:

DBCC for Sybase ASE Databases

DBCC for Sybase ASE Tables

[DBCC for Sybase ASE Indexes](#)

DBCC FOR SYBASE ASE DATABASES

The DBCC dialog box for databases lets you perform database-wide validations. The fundamental difference between the DBCC dialog box for databases, tables and indexes is the content of the DBCC Operation drop-down list.

The table below describes the options and functionality on the DBCC dialog box.

Option	Description
Check Allocation	Checks the allocation and use of all pages in the target database.
Check Catalog	Checks for consistency in and between system tables in the target database.
Check Database	Checks the allocation and structural integrity of all the objects in the target database.
Check Storage	Checks the target database for allocation, OAM page entries, page consistency, text valued columns, allocation of text valued columns, and text column chains. The results of this operation are stored in the dbccdb database.
Database Repair	Drops a damaged database.

For more information, see [DBCC.](#)

DBCC FOR SYBASE ASE TABLES

The DBCC dialog box for tables lets you perform table-level validations of databases. The fundamental difference between the DBCC dialog box for tables and indexes is the content of the DBCC Operation drop-down list.

The table below describes the options and functionality on the DBCC dialog box.

Option	Description
Check Allocation	Checks the database to see that every page is correctly allocated, and that no allocated page is unused. Use TABLEALLOC frequently (daily) to check page linkages in the Adaptive Server before performing a database dump to ensure the integrity of the dumped data.
Check Table	Checks the linkages and sizes of text, ntext and image pages for selected tables. For the data, index, text, ntext, and image pages of the target tables, this operation also checks that index and data pages are correctly linked, indexes are in their proper sorted order, pointers are consistent, the data on each page is reasonable, and the page offsets are reasonable. DBCC CHECKTABLE requires a shared lock on all tables and indexes in the database for the duration of the operation. However, DBCC CHECKTABLE does not check the allocations of pages in the specified table (for this, use DBCC CHECKALLOC). To perform DBCC CHECKTABLE on every table in the database, use DBCC CHECKDB.
Check Text	Upgrades text values after you have changed an Adaptive Server's character set to a multibyte character set.
Rebuild Index	Dynamically rebuilds one, multiple, or all indexes for a table in the target database, allowing indexes which enforce either primary key or unique constraints to be rebuilt without need for dropping and recreating. This operation is not supported for use on system tables.

DBCC Operation Options

Rapid SQL offers additional options for selected operations which you can specify to further customize a database consistency check. The table below describes each option:

Option	Description
Error Option	Click Fix Error to instruct Rapid SQL to fix any allocation errors it finds. You must put your database in single-user mode to fix errors, so specify this option during times of low usage.
Job Scope	Select Optimize to produce a report based on the allocation pages listed in the object allocation map (OAM) pages for the table. It does not report and cannot fix unreferenced extents on allocation pages that are not listed in the OAM pages. The optimized option is the default. Select Full to perform the equivalent of a table-level CHECKALLOC, reporting all types of allocation errors. Select Fast to produce an exception report of pages that are referenced but not allocated in the extent. Fast does not produce an allocation report.
Update Index Option	Click this check box to skip non-clustered indexes when updating index options.

For more information, see DBCC.

DBCC FOR SYBASE ASE INDEXES

The DBCC dialog box for indexes lets you perform index-level validations of databases. Unlike the DBCC dialog box for tables, this DBCC dialog box offers only one option on the DBCC Operation drop-down list: Check Allocation. This option checks the specified database to see that all pages are correctly allocated and that no allocated page is unused.

The table below describes the options and functionality on the DBCC dialog box.

Option	Description
DBCC Option	Checks the specified database to see that all pages are correctly allocated and that no page that is allocated is not used.
Error Option	Rapid SQL to fixes any allocation errors it finds. You must put your database in single-user mode to fix errors, so specify this option during times of low usage
Job Scope	Produces a report based on the allocation pages listed in the object allocation map (OAM) pages for the table. It does not report and cannot fix unreferenced extents on allocation pages that are not listed in the OAM pages. The optimized option is the default. A full job is the equivalent to a table-level CHECKALLOC, reporting all types of allocation errors. A fast job does not produce an allocation report, but produces an exception report of pages that are referenced but not allocated in the extent.

For more information, see DBCC.

DEALLOCATE UNUSED SPACE

> **NOTE:** This functionality is available for Oracle only.

The Deallocate Unused Space dialog box lets you deallocate space from clusters, indexes, and tables. You can also deallocate unused space from table partitions and subpartitions. When you find that allocated space is not being used, you can free that space for use by other objects by explicitly deallocating space. Oracle releases the freed space to the user quota for the tablespace in which the deallocation occurs.

Oracle deallocates unused space from the end of the object toward the high water mark. In other words, Oracle frees space starting with the space that would have been used last. If an extent is completely contained in the space to be deallocated, then the whole extent is freed. If an extent is only partially contained in the space to be deallocated, then Oracle shrinks that extent to the size of the used space up to the high water mark, and frees the unused space in the extent.

If you are deallocating unused space from and index and the index is range-partitioned or hash-partitioned, Oracle deallocates unused space from each partition in the index. If an index is a local index on a composite-partitioned table, Oracle deallocates unused space from each of the subpartitions in the index.

> **TIP:** You can verify that the deallocated space is freed by going to the Space Tab in the appropriate Rapid SQL editor.

To deallocate unused space from an index or table:

1   On the *Datasource Explorer*, select the Clusters, Indexes, Primary Keys, Tables, or Unique Keys node.

    Rapid SQL displays the objects of the selected type in the right pane of the Explorer window.

2   In the right pane of the application, select one or more objects, and from the **Command** menu select Deallocate Unused Space.

    Rapid SQL opens the **Deallocate Unused Space** dialog box.

3   Use the following table as a guide to understanding and specifying options on this dialog:

Option	Description
Specify the number of bytes above the high-water mark that the objects will have after deallocation. If no value is specified, all unused space will be freed.	If you do not specify an amount of unused space and the high water mark is above the size of INITIAL and MINEXTENTS, then all of the unused space is freed. If the high water mark is less than the size of INITIAL or MINEXTENTS, then all unused space above MINEXTENTS is freed.
	If you specify an amount of unused space and the remaining number of extents is smaller than MINEXTENTS, then the MINEXTENTS value changes to reflect the new number. If the initial extent becomes smaller as a result of the deallocation, the INITIAL value changes to reflect the new size of the initial extent.

4   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

DELETE STATISTICS

Lets you create and issue a DELETE STATISTICS statement, allowing you to delete statistics for a table or for specified columns of a table.

> **NOTE:**   This functionality is available for Sybase only.

To delete statistics for a table

1   Initiate a **Delete Statistics** action against one or more tables. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings in the **Delete Statistics** wizard:

Step	Settings and tasks
Action options	Displays the names of the tables you selected and if appropriate, lets you select a specific **Partition Name**.
Columns (Sybase 15 only)	**All columns** - lets you delete statistics for all columns in the table. **Specific columns** - select this option and then select check boxes associated with each specific column for which statistics are to be deleted.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For more information, see [Dependencies](#).
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

DESCRIBE

RIght-clicking a table, view, procedure or function in the right pane of the explorer and selecting **Describe** opens a window that lets you view columnar information (for tables and views) or input parameter information (for procedures and functions). For details, see [Describe Window](#).

DETACH DATABASE

The **Detach Database** object action removes the database from the server but leaves the database intact within the data and transaction log files that compose the database. These data and transaction log files can then be used to attach the database to any instance of Microsoft SQL Server, including the server from which the database was detached. This makes the database available in exactly the same state it was in when it was detached.

> **NOTE:**   This functionality is available for Microsoft SQL Server 2000 or later only.

To detach data and transaction files for a database

1   On the *Datasource Explorer*, select the Databases node.

    Rapid SQL displays the Databases in the right pane of the Explorer window.

2   In the right pane of the application, right-click a database, and then select Detach Database.

    Rapid SQL opens the **Detach Database(s)** dialog box.

3   To skip the UPDATE STATISTICS operation when detaching the database, select the *Skip Checks check* box for the target database(s).

> **TIP:**   This option is useful for databases that are to be moved to read-only media.

4   Click **Execute**. For information on the other options, see Preview and Scheduling.

For related information, see Attach Database.

DISABLE INDEX

The **Disable Index(es)** action lets you mark an index as disabled or inactive and unavailable for use.

> **NOTE:**   This functionality is available for InterBase/Firebird and Microsoft SQL Server 2005 and above.

> **NOTE:**   On SQL Server this action can also be applied to primary keys and unique keys.

To disable an index

1   Initiate a **Disable Index(es)** action against one or more indexes. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	Displays the name of the index, primary key, or unique key being disabled.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

DISABLE JOB

Rapid SQL lets you enable or disable any job queue. Job Queues are built-in mechanisms that let you schedule a variety of SQL-based or command-line driven tasks.

> **NOTE:** This functionality is available for Oracle only.

To disable a job queue

1   On the *Datasource Explorer*, select the Job Queues node.

Rapid SQL displays the job queues in the right pane of the Explorer window.

2   In the right pane of the application, right-click the target job queue, and then select Disable Job.

Rapid SQL opens the **Disable** dialog box.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

For related information, see Enable Job (Job Queue) and Run Job.

DISABLE KEYS

Rapid SQL lets you issue an ALTER TABLE statement that specifies the DISABLE KEYS option. This action tells MySQL stop updating nonunique indexes.

> **NOTE:** This functionality is available for MySQL only.

To stop updates to nonunique indexes:

1   On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2   Select the **Tables** node.

Rapid SQL displays the tables in the right pane of the Explorer window.

3   In the right pane of the application, select one or more tables.

4   From the Command menu select Disable Keys.

Rapid SQL opens the **Disable Keys** dialog box.

5   Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Lets you review the tables you selected.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action.

6   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

> **NOTE:**   You can subsequently recreate missing indexes. For details, see <u>Enable Keys</u>.

DISABLE/ENABLE TRIGGERS

These actions let you build and submit ALTER TABLE statements that specify ENABLE TRIGGER or DISABLE TRIGGER options.

> **NOTE:**   This functionality is available for Microsoft SQL Server and Sybase ASE only.

To enable or disable triggers for a table

1   On the Database Explorer, select the **Tables** node. Rapid SQL displays the tables in the Database Explorer.

2   On the Database Explorer, right-click one or more selected tables and select **Enable Triggers** or **Disable Triggers**.

Rapid SQL opens the **Enable Triggers** or **Disable Triggers** dialog box, as appropriate.

3   Click **Execute**.

DROP

> **NOTE:**   This functionality is available for all DBMS platforms.

The Confirm Drop dialog box lets you drop one or more database objects and remove their definition from the system catalog.

To drop an object

1   On the *Datasource Explorer*, select the node corresponding to the object type to be dropped.

Rapid SQL displays the target objects in the right pane of the Explorer window.

2   In the right pane of the application, right-click one or more selected objects, and then select **Drop** from the context menu.

Rapid SQL opens the **Confirm Drop** dialog box on the **Action Options** panel.

3   On the **Action Options** panel, select any options specific to the object type or configuration of the object you are dropping.

4   Navigate to the **Dependencies** pane to view database objects that are dependent on the object you are dropping.

> **NOTE:**   For Microsoft SQL Server 2005, when dropping Logins, you also have the option to delete corresponding user objects.

> **NOTE:**   For Microsoft SQL Server, when dropping Databases, you also have the option to delete the backup and restore history for the database.

5   Navigate to the **Preview** pane to verify that the correct DDL was created, and if necessary navigate backward and modify your choices.

6   Click **Execute** to drop the object.

DROP BY CATEGORY

This action lets you build and submit one or more calls to the **outln_pkg.drop_by_cat** subprogram, letting you drop outlines belonging to specified categories.

> **NOTE:**   This functionality is available for Oracle only.

To drop one or more stored outlines of a particular category

1   Initiate a **Drop By Category** action against one or more stored outlines. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in this **Drop By Category** wizard:

Step	Settings and tasks
Action Options	Select the categories to drop.
Preview	Displays the DDL that will execute the object action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

For related information, see the following topics:

- Change Category

- Reassign By Category

DROP CLONE

If you have created a clone from a base table, you can drop that clone. To determine whether a table has a clone, see the **Dependencies** tab/panel on a Table editor or on the wizard for object actions such as Drop or Rename.

> **NOTE:**   This functionality is available as of IBM DB2 for z/OS version 9.

To drop the clones of one or more tables:

1   Initiate a **Drop Clone** action against one or more tables. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings in the **Confirm Drop Clone** wizard:

Step	Settings and tasks
Action Options	Lets you review the action you initiated.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Displays the DDL that will execute the object action.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

For related information, see the following topics:

*   [Create Clone](#)

*   [Exchange Data With Clone](#)

DROP LOGIN TRIGGER

The Drop Login Trigger action lets you build and submit an sp_modifylogin call that specifies a NULL login script option. This removes the configured login trigger for the login, the default database stored procedure that executes each time the user successfully logs in.

> **NOTE:**   This functionality is available for Sybase.

> **NOTE:**   If a login has a login trigger configured, the specified stored procedure name is visible in the Login editor. For details, see [Logins Editor (Sybase)](#).

To remove the currently configured login trigger for a login:

1   Expand the **Logins** node. Rapid SQL displays all logins for the datasource.

2   Right-click a login and select **Login Trigger Actions > Drop Login Trigger** from the context menu. Rapid SQL opens the **Drop Login Trigger** dialog box.

3   Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	Displays the login you selected to run the action against.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see [Dependencies](#).
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.

For related information, see [Add/Modify Login Trigger](#).

DROP MATERIALIZED QUERY TABLE

The **Drop Materialized Query** action lets you convert a materialized query table to a regular table.

> **NOTE:** This functionality is available for DB2 LUW.

To convert a materialized query table to a regular table

1 On the *Datasource Explorer*, expand the Materialized Query Tables node.

Rapid SQL displays the materialized query tables in the right pane of the Explorer window.

2 In the right pane of the application, select one or more materialized query tables.

3 From the Command menu select Drop Materialized Query Table.

Rapid SQL opens the **Confirm MQT Conversion to Regular** dialog box.

4 Click **Execute**. For information on the other options, see Preview and Scheduling.

DROP UNUSED

This action lets you build and submit one or more calls to the **outln_pkg.drop_unused** subprogram, letting you drop outlines that have not been used since being created.

> **NOTE:** This functionality is available for Oracle only.

To drop one or more stored outlines that have not been used since being created

1 On the *Datasource Explorer*, expand the Stored Outlines node.

Rapid SQL displays the stored outlines in the right pane of the Explorer window.

2 In the right pane of the application, right-click one or more selected stored outlines and select Drop Unused from the context menu.

Rapid SQL opens the **Drop Unused** dialog box.

3 Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Preview	Displays the DDL that will execute the object action.

4 Click **Execute**. For information on the other options, see Preview and Scheduling.

DROP UNUSED COLUMNS

This action lets you build and submit an ALTER TABLE... DROP UNUSED COLUMNS statement, letting you remove columns currently marked as unused.

> **NOTE:** This functionality is available for Oracle.

To drop the clones of one or more tables:

1   Initiate a **Drop Unused Columns** action against one or more tables. For more information see <u>Initiating an object operation</u>.

2   Use the following table as a guide to understanding and modifying settings in the **Drop Unused Columns** wizard:

Step	Settings and tasks
Action Options	Displays the tables you selected
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see <u>Dependencies</u>.
Preview	Displays the DDL that will execute the object action. For details, see <u>Preview</u>/.

3   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

EDIT DATA

Right-clicking a table and selecting Edit Data lets you edit table data. For details, see <u>Data Editor</u>.

ENABLE/DISABLE STORED OUTLINE AUTO-CREATION/AUTO-USE

Rapid SQL provides two operations that can be performed against one or more stored outlines, that dictate whether execution plans are automatically generated or used:

Enable/disable stored outline auto-creation	Lets you specify whether Oracle to create a stored outline for every query submitted to the server.
Enable/disable stored outline auto-use	Lets you specify whether Oracle uses stored outlines to generate execution plans.

> **NOTE:** This functionality is available for Oracle only.

To specify whether stored outlines are automatically created and used

1   On the *Datasource Explorer*, expand the Stored Outlines node.

Rapid SQL displays objects of the selected type in the right pane of the Explorer window.

2   Select one or more stored outlines, on the *Datasource Explorer* tool bar click Command, and then select the item corresponding to the feature you want to enable or disable:

   • **Enable/disable stored outline auto-use**

   • **Enable/disable stored outline auto-creation**

3   Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks	
Action Options	**Automatically create stored outlines** or **Automatically use stored outlines**	Lets you enable or disable the feature assocaited with your selection.
	Category	Lets you select an outline category. This option is only available if you enabled the feature.
	No Override	If selected, outlines will not override. This option is only available if you enabled the feature.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.	
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.	

4   When ready, click **Execute**.

ENABLE JOB (JOB QUEUE)

Rapid SQL lets you enable or disable any job queue. Job Queues are built-in mechanisms that let you schedule a variety of SQL-based or command-line driven tasks.

> **NOTE:**   This functionality is available for Oracle only.

To enable a job queue

1   On the *Datasource Explorer*, select the Job Queues node.

Rapid SQL displays the job queues in the right pane of the Explorer window.

2   In the right pane of the application, right-click the target job queue, and then select Enable Job.

Rapid SQL opens the **Enable** dialog box.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

For related information, see Disable Job and Run Job.

ENABLE KEYS

Rapid SQL lets you issue an ALTER TABLE statement that specifies the ENABLE KEYS option. This action tells MySQL to recreate missing indexes.

To recreate missing indexes:

1   On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2   Select the **Tables** node.

Rapid SQL displays the tables in the right pane of the Explorer window.

3   In the right pane of the application, select one or more tables.

4   From the Command menu select Enable Keys.

Rapid SQL opens the **Enable Keys** dialog box.

5   Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Lets you review the tables you selected.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action.

6   Click **Execute**. For information on the other options, see Preview and Scheduling.

   NOTE: You can also instruct MySQL to stop updating nonunique indexes. For details, see
   Disable Keys.

ENABLE RECYCLE BIN

This action builds and submits an ALTER SYSTEM SET "_recyclebin"=TRUE or ALTER SYSTEM SET "_recyclebin"=FALSE statement, letting you enable or disable the Oracle Recycle Bin.

This action lets you enable or disable the Oracle Recycle Bin, building and submitting one of the following version-specific statement variations:

   • **Oracle 10g R1** - ALTER SYSTEM SET "_recyclebin" TRUE / FALSE

   • **Oracle 10g R2** - ALTER SYSTEM SET RECYCLEBIN ON / OFF SCOPE=BOTH

   • **Oracle 11g** - ALTER SYSTEM SET RECYCLEBIN ON / OFF DEFERRED SCOPE=BOTH (note
     that in this case, the change takes effect after reconnecting to the datasource)

When the Recycle Bin is enabled, dropped objects are stored in the Recycle Bin until explicitly purged.

   NOTE: This functionality is available for Oracle 10g.

To enable or disable the Recycle Bin

1 Right-click an Oracle datasource node, and select **Enable Recycle Bin** from the context menu.

2 Use the following table as a guide to understanding and modifying settings in this wizard:

Panel	Settings and tasks
Action Options	Select the **Enable** check box to enable Recycle Bin functionality. Deselect the **Enable** check box to disable the feature.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

> **NOTE:** In general, you can determine the current enabled/disabled state of the Recycle Bin by opening the **Enable Recycle Bin** dialog. If the **Enable** check box is selected, the Recycle Bin is currently Note however, that the **Enable Recycle Bin** action reads the current state of the Recycle Bin by querying the v$parameter object. On some Oracle 10g R1 servers, the "_recyclebin" parameter does not appear listed in v$parameter – it is hidden. As a result, for this specific case, the value of the check box when opening the **Enable Recycle Bin** dialog will always be TRUE, regardless of the actual value.

3 Click Execute. For information on the other options, see Preview and Scheduling.

For related information, see:

- Recycle Bin (for a support summary including topics such as displaying Recycle Bin contents)

- Flashback Recycle Bin Entry

- Purge Recycle Bin Entry

- Purge Recycle Bin

ERROR

The Error message displays a warning. After reading the warning, click Close to continue.

ESTIMATE SIZE

The Estimate Size dialog box for tables and indexes lets you estimate how large a table or index will become given a row growth projection. The results let you proactively plan your space-related object needs.

> **NOTE:** This functionality is available for Oracle and Sybase ASE only.

To estimate size for a table or index:

1 On the *Datasource Explorer*, expand a Tables or Indexes node.

   Rapid SQL displays the tables or indexes in the right pane of the Explorer window.

2 In the right pane of the application, select one or more tables or indexes.

3   From the Command menu select Estimate Size.

Rapid SQL opens the **Estimate Size** dialog box.

4   Use the following topics as a guide to estimating size and using the other options offered on this dialog:

- [Estimating Oracle index or table sizes](#)

- [Estimating Sybase index or table sizes](#)

ESTIMATING ORACLE INDEX OR TABLE SIZES

The **Estimate Size** dialog remains open until you manually close it. This lets you run several estimations, varying the method and various metrics. Each estimation run is a three-step process:

1   Select a size estimation method:

- **Use current row data from current statistics** - The table or index must have been previously analyzed for statistics to be obtained. Note that the **Analyze Index(es)** and **Analyze Table(s)** buttons can be used to initiate this action. For more details, see [Analyze](#).

- **Use table/index definition source** - The maximum row length for the index or table will be calculated. Note that this usually produces much higher space estimates.

- **Manually enter estimation metrics** - Manually enter in data for space estimates if the objects have no data or if the index definition source method is not desired.

2   Depending on the method you chose, provide additional metrics:

- **Percent Free**

- **Row Size**

- **Number of rows**

3   Click the **Estimate Size** button and view the estimate in the **Estimated Index Size** box.

Other options include:

- Use the **Report** button to generate an HTML report of results

- Use the **Save As** button to save a result grid file

- Use the **Owner** and **Name** controls to select another table or index for size estimates.

- Use the **Add Index** or **Add Table** button to add another table or index to your size estimates. Similarly, use the **Remove Index** or **Remove Table** button to remove a table or index from the dialog.

ESTIMATING SYBASE INDEX OR TABLE SIZES

The **Estimate Size** dialog remains open until you manually close it. This lets you run several estimations, varying the available metrics. Each estimation run is a two-step process:

1 Provide the metrics for your estimate:

- **Fill Factor** (indexes only) - Specify a percentage of how full each index page can become.

- **Number of rows** - Specify the number of rows for your size estimate.

> **NOTE:** Optionally you can use the **Update Statistics** button to open a dialog that lets you generate new statistics for the index or table. For details, see Update Statistics.

2 Click the **Estimate Size** button and view the estimate in the **Estimated Index Size** box.

Other options include:

- Use the **Report** button to generate an HTML report of results

- Use the **Save As** button to save a result grid file

- Use the **Database**, **Owner**, **Table Name**, and **Index Name** controls to select another table or index for size estimates.

- Use the **Add Index** or **Add Table** button to add another table or index to your size estimates. Similarly, use the **Remove Index** or **Remove Table** button to remove a table or index from the dialog.

EXCHANGE DATA WITH CLONE

This action lets you issue an EXCHANGE statement with the DATA BETWEEN TABLE *table1* AND *table2* syntax, letting you swap data between a base table and its clone. To determine whether a table has a clone, see the **Dependencies** tab/panel on a Table editor or on the wizard for object actions such as Drop or Rename.

> **NOTE:** This functionality is available as of IBM DB2 for z/OS version 9.

To exchange data between one or more tables and their clones:

1 On the *Datasource Explorer*, expand the Tables node.

Rapid SQL displays the tables in the right pane of the Explorer window.

2 In the right pane of the application, select one or more tables.

3 From the Command menu select Clone Actions > Exchange Data With Clone.

Rapid SQL opens the **Exchange Data With Clone** dialog box.

4 Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Lets you review the action you initiated.

Step	Settings and tasks
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Displays the DDL that will execute the object action.

5   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

For related information, see the following topics:

- [Create Clone](#)

- [Drop Clone](#)

EXECUTE

The Execution dialog box lets you execute extended procedures, functions, and procedures.

Platform and object type availability

	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	Oracle	SQL Server	Sybase
Extended procedures							✓
Functions					✓		
Procedures	✓	✓	✓		✓	✓	✓

To compile an extended procedure, function, or procedure

1   In the left pane of the Datasource Explorer, select a valid object node. Valid object nodes for this object operation are shown in the table above.

2   In the right pane, right click the target object and select **Execute** from the context menu.

> **NOTE:**   If the extended procedure, function, or procedure takes input parameters, Rapid SQL opens an execution dialog. For more information, see [Working With Execution Parameters](#).

The Results editor opens, showing either results of a successfully executed extended procedure, function, or procedure, or showing error messages if execution was not successful. For details, see [Results Editor](#).

WORKING WITH EXECUTION PARAMETERS

If you initiate an **Execute** action against an extended procedure, function, or procedure that takes input parameters, Rapid SQL opens a dialog that lets you provide parameter values and otherwise manage parameters.

The Procedure Execution dialog box lets you:

- Save input parameters as *.prm files to preserve specific input parameter configurations.

- Open *.prm files to save the effort of reentering specific input parameters.

- Reset parameters to their default setting.

The table below describes the options and functionality of the Procedure Execution dialog box:

Option	Description
Parameter	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message displays in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open	Click to open an Open dialog box, from which you can open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset	Click to reset the parameters in the Parameter window to their default values.
Default	Select to gather default information from the data dictionary. ORACLE ONLY: You can not specify non-default parameters after specifying a default parameter.
Continue	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

For more general information on the **Execute** operation, see [Execute](#).

EXTRACT

Rapid SQL lets you extract the statements required to create existing objects into an Interactive SQL window.

> **NOTE:**   This functionality is available for all DBMS platforms.

To extract the DDL for an object

1   On the *Datasource Explorer*, expand the node corresponding to the object type you want to extract.

    Rapid SQL displays objects of that type in the right pane of the Explorer window.

2   In the right pane of the application, select one or more objects.

3   On the Datasource Explorer tool bar, click *Extract*.

Rapid SQL opens the DDL Editor with the DDL to create the selected objects. For more information, see [DDL Editors](#).

EXTRACT DATA AS XML

This function allows you to take SQL data, extract it, and make it available as XML data. The XML Editor Filter allows you to pick and choose among columns and designate rows and then creates the For XML statement that enables the operation. The resulting XML document is created and presented in an active XML Editor. At this point the document can be saved in XML format.

> **NOTE:** This functionality is available for Oracle 9i, SQL Server 8.0, and Sybase 12.5.1 or later.

> **CAUTION:** To use the **Extract Data as XML** feature in Sybase, you must first purchase and install an XML Services license from Sybase.

To Open the XML Editor Filter

1 On the Datasource Explorer, expand the target data source.

2 Expand the Tables node.

3 In the right pane of the Explorer window, right-click any table listed, and then select Extract Data as XML.

   Rapid SQL opens an **XML Editor Filter** dialog.

4 In the **Columns** area, select the check boxes corresponding to the columns with data you want to extract.

5 Optionally, in the **Where** box, type a WHERE clause to qualify the records that will be extracted.

> **NOTE:** As you select options, inspect the **Select** box to verify the query you are creating.

6 Use the table below as a guide to understanding and specifying the optional DBMS-specific settings in this dialog:

DBMS	Setting or group	Description
Sybase		Refer to Sybase online documentation at http://infocenter.sybase.com. Search the documentation corresponding to your Sybase version/product for XML mapping information.
Oracle	**Row Set Tag**	Identify the XML element name you want to use to replace the Row Set tag.
	Row Tag	Identify the XML element names you want to use to replace the Row tag.
	Max Rows.	The maximum number of rows to fetch

DBMS	Setting or group	Description
SQL Server	XML Mode	**AUTO** mode returns query results as nested XML elements. For any table in the From clause with a column in the Select clause, it is represented as an XML element.
		When you select **RAW** mode, each row in the query result set is transformed into an XML element with the generic row identifier. The XML attribute name will be the same as the column name for non-null columns.
	XML Options	**XML DATA** specifies that an XML data schema will be returned.
		ELEMENTS specifies that columns will be returned as subelements-- otherwise they are mapped to XML attributes. This is an option only if AUTO is selected.
		When **BINARY BASE64** is selected, any binary data is encoded in base-64 format. You must specify this option in RAW mode. It is the default for AUTO.

7 When ready, click **OK**. Rapid SQL opens an XML representation of the extracted data in an XML editor.

FLASHBACK RECYCLE BIN ENTRY

This action lets you build and submit a FLASHBACK TABLE... TO BEFORE DROP statement against a table in the Recycle Bin. This recovers the table and restores it to its state immediately before the drop.

> **NOTE:**   This functionality is available for Oracle 10g.

> **NOTE:**   The Datasource Explorer entry for a Recycle Bin entry has a **Can Undrop** field that indicates whether a **Flashback Recycle Bin Entry** action can be performed against the item

To recover a table currently in the Recycle Bin

1 Expand the **Recycle Bin** node to display its contents.

2 Right-click one or more selected tables in the Recycle Bin and select **Flashback Recycle Bin Entry** from the context menu. The **Flashback Recycle Bin Entry** dialog opens.

3 Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	Displays the table entries you selected to restore.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see <u>Dependencies</u>.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.

4 Click Execute. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

For related information, see:

- [Recycle Bin](#) (for a support summary including topics such as displaying Recycle Bin contents)

- [Enable Recycle Bin](#)

- [Purge Recycle Bin Entry](#)

- [Purge Recycle Bin](#)

FLASHBACK TABLE

This action builds and submits a FLASHBACK TABLE statement, letting you restore a table to a state at some specified time in the past.

> **NOTE:** This functionality is available for Oracle 10g and above. Available options differ by the version of Oracle that Rapid SQL is working against.

> **NOTE:** Before using this object action, consult Oracle documentation for information on restrictions, required permissions, and details on System Change Numbers and Restore Points. For more information, see [Accessing Third Party Documentation](#).

To restore a table to some previous state

1   Initiate a **Flashback Table** action against a table. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings on the dialog:

Panel	Settings and tasks	
Action Options	**Type**	Lets you select a specific FLASHBACK TABLE clause:
		BEFORE DROP - If the table is currently in the Recycle Bin, the table will be restored to its state immediately before being dropped.
		SCN - restores the table to a time corresponding to a specified System Change Number.
		TIMESTAMP - restores the table to a time corresponding to a specified timestamp value.
		RESTORE POINT - (10g R2 ^) restores the table to a time corresponding to a specified name associated with an SCN.
	SCN	If you selected a **Type** value of SCN, type a System Change Number.
	Timestamp	If you selected a **Type** value of TIMESTAMP, use this control to provide a timestamp.
	Restore Point (10g R2 ^)	If you selected a **Type** value of RESTORE POINT, select a restore point.
	Enable Trigger	If you selected a **Type** value of SCN, TIMESTAMP, or RESTORE POINT, selecting this check box overrides the default behavior, ensuring that triggers remain enabled during the flashback process.
	Rename To	If you selected a **Type** value of BEFORE DROP, specify a new name for the table being retrieved from the Recycle Bin.

Panel	Settings and tasks
Dependencies	For details on using this tab, see <u>Dependencies</u>.
Preview	Displays the DDL that will execute the object action. For details, see <u>Preview</u>.

3   Click Execute. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

For related information, see <u>Flashback Recycle Bin Entry</u>.

FLUSH CACHE

Rapid SQL lets you clear all dynamic SQL in the cache for a package and forces IBM DB2 for Linux, Unix, and Windows to recompile the SQL the next time it is called.

> **NOTE:**   This functionality is available for IBM DB2 for Linux, Unix, and Windows 8.1 only.

To flush the cache for a package

1   On the Datasource Explorer, expand the target data source.

2   Expand the Packages node.

3   In the right pane of the Explorer window, right-click any package listed, and then select Flush Cache from the context menu.

Rapid SQL opens the **Flush Package Cache** dialog box.

4   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

FLUSH TABLES

This action lets you build and submit a FLUSH statement, specifying a TABLE or TABLES option. This flushes the specified tables and removes all query results from the query cache.

> **NOTE:**   This functionality is available for MySQL only.

To flush one or more tables:

1   On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2   Select the **Tables** node.

Rapid SQL displays the tables in the right pane of the Explorer window.

3   In the right pane of the application, select one or more tables.

4   Right-click the table or tables and select Flush Tables from the context menu.

Rapid SQL opens the Flush Tables dialog box.

5   Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Displays the tables selected for this flush operation.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action.

6   Click **Execute**. For information on the other options, see Preview and Scheduling.

FREE (PACKAGES)

Rapid SQL lets you delete one or more packages.

> **NOTE:**   This functionality is available for IBM DB2 for z/OS only.

To delete packages:

1   Initiate a **Free Packages** action against one or more packages. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Confirm Free** wizard:

Step	Settings and tasks
Action Options	Displays the names of the packages you chose to free. This panel also lets you select a **Order Drop By Selection** setting, which dictates whether drop statements are generated in the order objects are selected.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

FREE PLAN

This action lets you drop plans associated with packages.

To drop a plan associated with a package

• Click the **Free** button on the **Plans/Packlists** tab of the Packages editor. For more information, see Packages Editor (IBM DB2 Z/OS).

For more information on dropping database objects, see Drop.

GENERATE PACKAGE/PROCEDURE/STATEMENT

> **NOTE:** This functionality is available for DB2 LUW, DB2 z/OS, InterBase/Firebird, Oracle, SQL Server, and Sybase.

Rapid SQL lets you generate simple packages, procedures, and statements for selected tables and lets you generate a simple select statement for views. Because the packages and procedures generated by Rapid SQL are rudimentary, they are intended merely as a starting point and should be modified to reflect your specific needs. Rapid SQL opens the generated statements in the ISQL Window.

When Rapid SQL creates a package from a table, it generates a series of procedures designed to emulate the typical variety of procedures in a package. Rapid SQL lets you choose the IN and OUT columns and generates the procedures based on your selections. Rapid SQL creates procedures and statements from tables in the same way.

Important Notes

None

For more information, see:

- [Generate Packages, Procedures, and Statements](#)

- [Generate Select Statement](#)

GENERATE PACKAGES, PROCEDURES, AND STATEMENTS

Rapid SQL opens this dialog box when you want to generate code for an Insert, Update, or Delete statement. This dialog box lets you specify the columns you want to include in the generation of an Insert, Update, or Delete statement or procedure.

GENERATE SELECT STATEMENT

Rapid SQL opens this dialog box when you want to generate code for a Select statement. This dialog box includes two panes: one for specifying Input Columns, one for specifying Output Columns.

> **NOTE:** This dialog box mirrors the functionality of Rapid SQL's Embarcadero Code Generator dialog box, which is accessible from the application's Tools menu and Tools toolbar.

The **Select 1 or More Columns** dialog box lets you specify the IN and OUT columns for a package or procedure, or the columns to select for a Select statement and any associated WHERE clause. Rapid SQL uses the Input Columns to generate the WHERE clause.

HIDE TEXT

This action lets you build and submit sp_hidetext calls against one or more objects. This allows you to hide the source text for particular object types:

> **NOTE:** This functionality is supported for Sybase only.

> **CAUTION:** The **Hide Text** action is irreversible. Once hidden, source text cannot be made viewable or otherwise retrievable again.

> **NOTE:** For details on the specific elements hidden, consult Sybase documentation. See Accessing Third Party Documentation.

Within Rapid SQL, when viewing an object after hiding text, hidden elements occupying entire tabs or individual property boxes, are annotated with a **Source text for this compiled object is hidden** message.
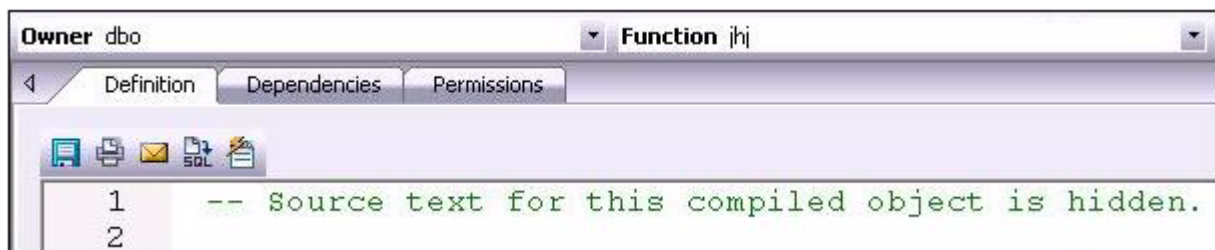
Hide Text can be applied both to individual objects and to objects containing or owning other objects. Supported object types for which Hide Text can be applied to individual objects, and the resulting Rapid SQL-specific hidden elements in object editors are as follows:

Check constraints	Check Condition box, Definition tab.
Defaults	Value box, Properties tab.
Extended procedures	Library name box, Properties tab.
Functions	Definition tab.
Procedures	Definition tab.
Rules	Restriction box, Properties tab.
Triggers	Definition tab.
Views	Definition tab.

Object-containing or object-referencing object types to which Hide Text can be applied are as follows:

Databases	Hides the source text of all compiled objects in the selected database.
Tables	Hides the source text of all check constraints, defaults, and triggers defined on the selected table.
Users	Hides the source text of all compiled objects owned by the selected user.

CAUTION: Because of the potentially broad scope of hiding text for users and databases, you should exercise extreme care when applying Hide Text to those object types.

To permanently hide the source text for one or more objects:

1   In the left pane of the *Datasource Explorer*, expand nodes until the node corresponding to one of the target object types is visible, and select that node. Objects of that type are shown in the right pane.

2   In the right pane, right-click one or more selected objects, and then select **Hide Text** from the context menu. The **Confirm Hide Text** dialog opens.

3   Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	Displays the name of the objects you selected.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Preview the DDL generated for the operation and when ready, use the Schedule or Execute button to perform this action.

4   Click Execute. For information on the other options, see [Preview](#) and [Scheduling](#).

Browser views for supported objects show a **Hidden** property that indicates whether **Hide Text** has been applied to the object. **Y** indicates that source text for this compiled object has been irreversibly hidden.

IMPORT DATA FROM FILE

RIght-clicking a table in the right pane of the explorer and selecting **Import Data From File** opens a dialog that lets you import table data from a file. For details, see [Import Data](#).

LOAD JAVA

Before you can call Java stored procedures, you must load them into the Oracle database and publish them to SQL.

> **NOTE:**   This functionality is available for Oracle 8i or later.

The Java Load Wizard lets you:

- Select the owner of the Java object and the files to load.

- Select options for the loading of the files.

- Select Resolver options.

For more information, see [Java Load Wizard - Panel 1](#).

JAVA LOAD WIZARD - PANEL 1

The table below describes the options of the first panel of the Java Load Wizard.

Option	Description
Who owns the Java Object?	Lets you select the owner of the Java object.
Select files to be loaded	Select a file, and then click Add.

JAVA LOAD WIZARD - PANEL 2

The table below describes the options of the second panel of the Java Load Wizard.

Option	Description
When do you want the Java files to be resolved?	Lets you specify when the source file is loaded as a source schema object, the source file is compiled, class schema objects are created for each class defined in the compiled .java file, and the compiled code is stored in the class schema objects.
Select the Encoding Options	Lets you specify the encoding of the .java file.
Grant Access to the following users	Lets you select one or more users.

JAVA LOAD WIZARD - PANEL 3

The table below describes the options of the third panel of the Java Load Wizard.

Option	Description
Other Load Options	OPTIONAL: Lets you select options.
Add Resolver Options	Lets you specify the objects to search within the schemas defined.
	Add - Click to open the Select a Resolver Option dialog box to add a new resolver option in the list.
	Edit - Click to open the Resolver Edit dialog box to modify a resolver option.
	Remove - Select one or more resolver option and click to delete.

LOCK

The Lock Table dialog box lets you lock tables to prevent other users from reading or updating the table data. Locking a table saves the time of locking each row of the table that needs to be updated. Lock mode options preventing other users from viewing or modifying table data, or allowing other users to view but not modify table data. Rapid SQL releases locks at the end of a transaction.

> NOTE: This functionality is available for IBM DB2 for Linux, Unix, and Windows only.

To lock a table:

1   On the *Datasource Explorer*, select the Tables node.

    Rapid SQL displays the tables in the right pane of the Explorer window.

2   In the right pane of the application, right-click one or more selected tables, and then select Lock.

    Rapid SQL opens the **Lock Table** dialog box.

3   Select a Lock Mode option:

    • Share - Lets other users view but not modify the table data.

    • Exclusive - Prevents other users from viewing or modifying the table data.

4   Click **Execute**. For information on the other options, see Preview and Scheduling.

MOVE LOG

Rapid SQL lets you move a transaction log from one device to another.

> **NOTE:**   This functionality is available for Sybase ASE only.

To move a database transaction log to another device:

1   On the Datasource Explorer, select the Databases node.

    Rapid SQL displays the Databases in the right pane of the Explorer window.

2   In the right pane of the application, right-click one or more selected databases, and then select **Move Log** from the context menu.

    Rapid SQL opens the **Move Log** dialog box.

3   From the **New Device** dropdown, select the defined device to which you want to move the transaction log. For information on creating database devices, see Database Device Wizard (Sybase).

NEXT USED FILEGROUP

This action builds and submits an ALTER PARTITION SCHEME... NEXT USED statement, letting you alter the designated NEXT USED filegroup for a partition scheme.

> **NOTE:**   This functionality is available as of SQL Server 2005.

To change a partition scheme's next used filegroup:

1   Initiate a **Next Used Filegroup** action against a partition scheme. For more information, see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Next Used Filegroup	Select the new filegroup from the **Filegroup Name** dropdown.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

OPTIMIZE TABLES

Rapid SQL lets you issue an OPTIMIZE TABLE statement, reclaiming unused space and defragmenting the data file.

> **NOTE:**   This functionality is available for MySQL only.

To optimize table storage:

1   On the *Datasource Explorer*, expand nodes until the **Tables** node is visible.

2   Select the **Tables** node.

    Rapid SQL displays the tables in the right pane of the Explorer window.

3   In the right pane of the application, select one or more tables.

4   From the Command menu select Optimize Tables.

    Rapid SQL opens the **Optimize Tables** dialog box.

5   Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Lets you review the tables you selected.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Displays the DDL that will execute the object action.

6   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

PLACE

The Placement dialog box lets you place tables and indexes on different segments. From a performance standpoint it is not recommended to have a table and its supporting indexes on the same device or disk segment. It is also good to have more frequently accessed indexes and tables grouped together on higher speed devices, if possible.

> **NOTE:**   Place functionality is available for Sybase ASE.

To place an index or table on a segment:

1   Initiate a **Place** action against one or more tables or indexes. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Place** wizard:

Step	Settings and tasks
Action options	Use the **Segment** dropdown to specify the segment on which you can place objects, the default, logsegment or system.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

POPULATION STATUS

This action builds and submits an ALTER FULLTEXT INDEX statement with a START {FULL|INCREMENTAL|UPDATE} POPULATION or STOP | PAUSE | RESUME POPULATION argument, letting you start, stop, pause or resume population of the full-text index.

> **NOTE:**   This functionality is available as of SQL Server 2005.

> **NOTE:**   Before using this action, consult Microsoft SQL Server documentation for detailed information on topics such as timestamp column restrictions on the INCREMENTAL option and the interaction of these options with the background update index and auto change tracking features. For more information, see Accessing Third Party Documentation.

To control population of a full-text index:

1   Initiate a **Population Status** action against one or more full-text indexes. For more information, see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action Options	Select an option from the **Action** dropdown:
	START FULL - every row of the table is retrieved for full-text indexing, even rows that have already been indexed.
	START INCREMENTAL - only the rows modified since the last population are retrieved. This option only works properly against tables with a timestamp column.
	START UPDATE - only insertions, updates, or deletions processed since the last change-tracking index update. Change-tracking population must be enabled on the table, but the background update index or the auto change tracking should be disabled.
	STOP - stops a paused population or population in progress.
	PAUSE - pauses a FULL population in progress
	RESUME - resumes a paused FULL population.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

PURGE RECYCLE BIN

This action builds and submits a PURGE RECYCLEBIN, PURGE DBA_RECYCLEBIN, PURGE TABLESPACE *tablespacename*, or PURGE TABLESPACE *tablespacename* USER *username* statement. This lets you select one of the following purge Recycle Bin options:

- Purge all objects belonging to the current user

- Purge all objects belonging to all users

- Purge all objects residing on a specified tablespace

- Purge all objects residing on a specified tablespace and belonging to a specified user

   NOTE: This functionality is available for Oracle 10g.

To purge objects in the Recycle Bin tables and associated objects

1   Right-click the **Recycle Bin** node and select **Purge Recycle Bin** from the context menu.

2   Use the following table as a guide to understanding and modifying settings on the **Confirm Purge Recycle Bin** wizard:

Panel	Settings and tasks	
Action Options	**Purge**	Select a PURGE option type: RECYCLEBIN purges all objects belonging to the current user. DBA_RECYCLEBIN purges objects system wide. TABLESPACE purges objects residing on a specific tablespace.
	Tablespace	If you selected a **Purge** value of TABLESPACE, select the tablespace for which all residing objects will be purged. Optionally, use the **User** control to further qualify the objects that will be purged.
	User	If you selected a **Purge** value of TABLESPACE, you can use this control to restrict purging to only those tablespace-residing objects belonging to a particular user.
Preview	Displays the DDL that will execute the object action. For details, see Preview.	

3   Click Execute. For information on the other options, see Preview and Scheduling.

For related information, see:

- Recycle Bin (for a support summary including topics such as displaying Recycle Bin contents)

- Enable Recycle Bin

- Flashback Recycle Bin Entry

- Purge Recycle Bin Entry

PURGE RECYCLE BIN ENTRY

This action builds and submits a PURGE INDEX or PURGE TABLE statement, letting you permanently delete tables or indexes from the Oracle Recycle Bin. Once purged, the tables or indexes and any associated objects that were contained in the Recycle Bin are no longer retrievable.

> **NOTE:** This functionality is available for Oracle 10g.

> **NOTE:** The Datasource Explorer entry for a Recycle Bin entry has a **Can Purge** field that indicates whether a **Purge Recycle Bin Entry** action can be performed against the item

To permanently purge selected tables or indexes currently in the Recycle Bin

1   Expand the **Recycle Bin** node to display its contents.

2   Right-click one or more selected tables or indexes in the Recycle Bin and select **Purge Recycle Bin Entry** from the context menu. The **Confirm Purge Recycle Bin Entry** dialog opens.

3   Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	Displays the table and index entries you selected to purge.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see Dependencies.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.

4   Click Execute. For information on the other options, see Preview and Scheduling.

> **NOTE:**   When dropping tables on a datasource with the Recycle Bin feature enabled, the Rapid SQL **Drop** action has an option that lets you immediately purge the table.

For related information, see:

• Recycle Bin (for a support summary including topics such as displaying Recycle Bin contents)

• Enable Recycle Bin

• Flashback Recycle Bin Entry

• Purge Recycle Bin

QUIESCE

Rapid SQL lets you quiesce at the instance or database level. For details, see the following topics:

• Quiesce (Database)

• Quiesce (Instance)

QUIESCE (DATABASE)

This action lets you restrict user access to a database.

> **NOTE:**   This functionality is available for IBM DB2 for Windows/Unix 8.1 only. Rapid SQL does not support Quiesce Database for an IBM DB2 8 server from a IBM DB2 for Windows/Unix 7 client or for an IBM DB2 for Windows/Unix 7 server from an IBM DB2 for Windows/Unix 7 client.

To quiesce an database

1 On the Explorer, right-click a datasource node and select **Command > Quiesce** from the context menu. Provide login credentials if prompted.

2 Use the following table as a guide to understanding and modifying settings on the **Quiesce Database** dialog:

Setting	Description
Connections	Lets you select an **Immediate** quiesce or **Defer.**

3 Click **Execute**.

For related information, see Unquiesce.

QUIESCE (INSTANCE)

This action lets you restrict user access to an instance.

> **NOTE:** This functionality is available for DB2 LUW 8.1 only.

To quiesce an instance

1 On the Explorer, right-click the Instance node.

    ⌀

2 Select **Command > Quiesce** from the context menu. Provide login credentials if prompted.

3 Use the following table as a guide to understanding and modifying settings on the **Quiesce Instance** dialog:

Setting	Description
For user	Select this radio button to restrict a user and then type the user name in the associated box.
For group	Select this radio button to restrict a group and then type the group name in the associated box.
Connections	Lets you select an **Immediate** quiesce or **Defer.**

4 Click **Execute**.

For related information, see Unquiesce.

REASSIGN BY CATEGORY

Rapid SQL lets you reassign the category of stored outlines in Oracle.

> **NOTE:** This functionality is available for Oracle only.

Outlines are a set of results for the execution plan generation of a particular SQL statement. When you create an outline, plan stability examines the optimization results using the same data used to generate the execution plan. That is, Oracle uses the input to the execution plan to generate an outline, and not the execution plan itself.

To reassign a stored outline to a different category

1   Initiate a **Reassign By Category** action against one or more stored outlines. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Reassign By Category** wizard:

Step	Settings and tasks
Action options	Displays the outlines you selected. Select a new category for the outline or outlines from the **Category** dropdown.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

For related information, see the following topics:

- Change Category

- Drop By Category

REBIND PACKAGES

The Rebind Package dialog box lets you update the best access path for SQL statements when the contents of a package changes.

> **NOTE:**   This functionality is available for IBM DB2 for Linux, Unix, and Windows and IBM DB2 for z/OS only.

> **TIP:**   If the physical storage of a package is changed or dropped, rebinding updates the path of the SQL statements.

To rebind one or more packages:

1   On the Datasource Explorer, select the Packages node.

    Rapid SQL displays the Packages in the right pane of the Explorer window.

2   In the right pane of the application, select one or more packages, right-click the selected packages, and select Rebind from the context menu.

    Rapid SQL opens the **Rebinding Package** dialog box.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

REBIND PLANS

This action lets you update the best access path for SQL statements when the contents of a plan change.

> **NOTE:** This functionality is available for IBM DB2 for z/OS only.

> **TIP:** If the physical storage of a plan is changed or dropped, rebinding updates the path of the SQL statements.

To rebind a plan associated with a package

- Click the **Rebind** button on the **Plans/Packlists** tab of the Packages editor. For more information, see Packages Editor (IBM DB2 Z/OS).

REBUILD (FULL-TEXT CATALOGS)

This action builds and submits a basic ALTER FULLTEXT CATALOG *catalogname* REBUILD statement, with no additional options or arguments. This lets you rebuild the entire catalog by deleting the existing catalog and building a new one.

> **NOTE:** This functionality is available as of SQL Server 2005.

To rebuild a full-text catalog:

1 Initiate a **Rebuild** action against one or more full-text catalogs. For more information, see Initiating an object operation.

2 Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action options	Displays the catalogs you selected.
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3 Click **Execute**. For information on the other options, see Preview and Scheduling.

REBUILD INDEX

See the following topics for DBMS-specific instructions on rebuilding indexes:

- Rebuild Index (Oracle)

- Rebuild Index (SQL Server)

- Rebuild Indexes (InterBase/Firebird)

REBUILD INDEX (ORACLE)

NOTE: This functionality is available for Oracle only.

The Rebuild Indexes dialog box lets you rebuild an index that has become fragmented. Rebuilding an index is a good alternative to coalescing an index because you can move the index to a different tablespace and change both tablespace and storage parameters while eliminating fragmentation. However, rebuilding an index has a higher cost than coalescing an index. These same qualities also make rebuilding an index a viable alternative to dropping an index then re-creating it.

As a rule of thumb, check indexes for rebuilds when their level (or tree depth) reaches four or greater, or many deleted leaf rows are found. The Rebuild Indexes dialog box can also be used to easily move an index from one tablespace to another.

Important Notes

- If you are rebuilding a function-based index, the index is enabled when the rebuild is finished.

- You cannot rebuild a partitioned index. You must rebuild each partition or subpartition individually.

The table below describes the options and functionality on the Rebuild dialog box.

Option	Description
New Tablespace	Defaults to the tablespace which currently includes the index. To change the tablespace containing the index, choose a new tablespace from the list.
Logging	Recoverable - The creation of the index logs in the redo log file.
	Non-Recoverable - The creation of the index is not logged in the redo log file.
Use Parallel Processes	Performs processes for the sequential execution of a SQL statement in parallel using multiple parallel processes. One process, known as the parallel execution coordinator, dispatches the execution of a statement to several parallel execution servers and coordinates the results from all of the server processes to send the results back to the user.
	NOTE: Only available for Oracle with the Parallel Server option.
	NOPARALLEL execution - Select this if you are concerned that the cost of synchronizing parallel processes will impede the throughput of data.
Order	Reverse - Instructs Oracle to store the bytes of the index block in reverse order and to exclude the ROWID when rebuilding the index.
	No Reverse - Instructs Oracle to store the bytes of the index block in normal order when rebuilding the index.

To rebuild an Oracle index

1   On the *Datasource Explorer*, select the Indexes node.

Rapid SQL displays the Indexes in the right pane of the Explorer window.

2   In the right pane of the application, right-click one or more selected indexes, and then select Rebuild from the context menu.

Rapid SQL opens the Rebuild Indexes dialog box.

3   To move the index to a new tablespace, click the *New Tablespace* list and then click the new tablespace.

4   In the *Logging* box, click:

- The *Recoverable* option button to make the operation log in the redo file.

- The *Non-Recoverable* option button if you do not want the operation logged in the redo file.

5   If you are using Parallel Server, select the *Parallel Server* check box and:

- Type a value indicating the number of query server processes that should be used in the operation in the *Degree* box.

- Type a value indicating how you want the parallel query partitioned between the Parallel Servers in the *Instances* box.

6   In the *Order* box:

- Click the *Reverse* option button to rebuild the index to store the bytes of the index block in reverse order.

- Click the *No Reverse* option button to rebuild the index to store the bytes of the index block in order.

   NOTE: This option is only available for Oracle8.

7   Click **Execute**. For information on the other options, see [Preview](#), and [Scheduling](#).

REBUILD INDEX (SQL SERVER)

The **Rebuild Indexes** dialog box lets you rebuild an entire index, primary key, or unique key or a single partition of those objects. Depending on your choice, a number of REBUILD WITH clause options are available.

   NOTE: This functionality is available for Microsoft SQL Server 2005 and above.

To rebuild an index

1 Initiate a **Rebuild Index** action against one or more indexes, primary keys, or unique keys. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying settings in the **Rebuild Index** wizard:

Step	Settings and tasks
Action options	To rebuild a single partition, specify a **Partition number** and optionally, provide **Sort in tempdb** and **MaxDOP** property values used to create the REBUILD WITH clause.
	To rebuild the entire index, primary key, or unique key, DO NOT provide a **Partition Number**, and optionally, provide **Pad Index**, **Sort in tempdb**, **Ignore Duplicate Key**, **Statistics no recompute**, **Online**, **Allow Row Locks**, **Allow Page Locks**, **MaxDOP**, and **Fill Factor** property values used to create the REBUILD WITH clause.
	For more information on these properties, see [Index Wizard (SQL Server)](#).
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3 Click **Execute**. For information on the other options, see [Preview](#), and [Scheduling](#).

REBUILD INDEXES (INTERBASE/FIREBIRD)

This action rebuilds indexes by issuing ALTER INDEX/INACTIVE followed by ALTER INDEX/ACTIVE statements.

To rebuild an index

1 On the *Datasource Explorer*, expand nodes until the **Indexes** node is displayed.

2 Expand the indexes node and select one or more indexes.

3 Right-click the selected indexes and select **Rebuild Indexes** from the context menu. The **Rebuild Index** dialog opens.

4 Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	Displays the indexes you chose to rebuild.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

5 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

REBUILD OUTLINES

Rapid SQL lets you rebuild one or more stored outlines in a single operation.

> **NOTE:** This functionality is available for Oracle only.

To rebuild a stored outline:

1   Initiate a **Rebuild Outlines** action against one or more stored outlines. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Rebuild Outlines** dialog:

Step	Settings and tasks
Action options	Displays the stored outlines you selected.
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

REBUILD TABLE

This action builds and submits an ALTER TABLE statement with an ENGINE=InnoDB option.

> **NOTE:** This functionality is available for MySQL only.

To rebuild a table:

1   Initiate a **Rebuild Table** action against one or more tables. For more information, see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Rebuild Table** dialog:

Step	Settings and tasks
Action options	Displays the tables you selected.
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

RECOMPILE

Rapid SQL lets you recompile one or more tables. Recompilation causes each procedure and trigger that uses the target table to be recompiled the next time it runs.

> **NOTE:** This functionality is available for Microsoft SQL Server and Sybase ASE only.

The queries used by procedures and triggers are optimized only once, when they are compiled. As you add indexes or make other changes to your database that affect its statistics, your compiled procedures and triggers may lose efficiency. By recompiling the procedures and triggers that act on a table, you can optimize the queries for maximum efficiency.

To recompile one or more tables:

1   On the *Datasource Explorer*, select the Tables node.

    Rapid SQL displays the tables in the right pane of the Explorer window.

2   In the right pane of the application, right-click one or more selected tables, and then select Recompile from the context menu.

    Rapid SQL opens the **Recompile Tables** dialog box.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

REFRESH TABLE

The Refresh Table Dialog box lets you reload materialized query tables that have been defined with refresh options.

> **NOTE:** This functionality is available for IBM DB2 for Linux, Unix, and Windows 8.1 and later.

To reload a materialized query table

1   Initiate a **Refresh Table** action against one or more materialized query tables. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings on the **Refresh Materialized Query Table** dialog:

Pane	Description
Action Options	Select an **Online Option** of ALLOW NO ACCESS, ALLOW READ ACCESS or ALLOW WRITE ACCESS. Enable or disable **Query Optimization**. Select an INCREMENTAL or NOT INCREMENTAL **Refresh Option.**
Preview	Preview the DDL generated by the options you chose.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

Important Notes

For procedures on restoring a damaged master database, consult the Commands Reference Manual.

RENAME

This action lets you rename an object. In general, all referenced or referring objects are updated to reflect the new name.

The following table outlines object type support for the operation for all supported DBMS:

	DB2 LUW	DB2 z/OS	ITB/FBD	MySQL	Oracle	SQL SVR	Sybase
Blob filters			✓				
Check constraints	✓	✓	✓		✓	✓	✓
Databases						✓	✓
Defaults							✓
Domains			✓				
Exceptions			✓				
Extended procedures							✓
Foreign keys	✓	✓	✓		✓		✓
Generators			✓				
Indexes	✓		✓		✓		✓
Materialized Views					✓		
Primary keys	✓		✓		✓		✓
Procedures			✓		✓		✓
Roles			✓				
Rules							✓
Sequences	✓				✓		
Shadows			✓				
Stored outlines					✓		
Synonyms					✓		
Tables	✓	✓		✓	✓		✓
Tablespaces	✓				✓		
Unique keys	✓		✓		✓		✓
Views			✓		✓		

The following table notes the exceptions and provides prerequisite tasks to be performed before renaming an object.

DBMS	Notes and restrictions on renaming
Microsoft SQL Server	Microsoft SQL Server lets you rename a database if you own it. Before renaming a database, set it to single-user mode.
	Microsoft SQL Server will not rename a table if it is referenced within the body of other objects that call it, such as tables, triggers or views. As a result, renaming a table can result in broken dependencies with other objects. Also, Microsoft SQL Server does not let you rename System Tables.
IBM DB2 for OS/390 and z/OS	Rapid SQL lets you rename a primary key if the underlying table has only one owner.
	The rename operation does not rename the table if it is referenced within the body of other objects, such as tables, triggers or views, that call it. As a result, renaming a table can result in broken dependencies with other objects.
Sybase ASE	Before renaming a database, set it to single-user mode.
	System indexes can not be renamed.
	The rename operation does not rename the stored procedure if it is referenced within the body of other objects, such as another stored procedure, that call it. As a result, renaming a stored procedure can result in broken dependencies with other objects.
	The rename operation does not rename the table if it is referenced within the body of other objects, such as tables, triggers or views, that call it. As a result, renaming a table can result in broken dependencies with other objects.
	The rename operation does not rename the view if it is referenced within the body of other objects, such as stored procedures, triggers or other views, that call it. As a result, renaming a view can result in broken dependencies with other objects.

To rename an object

1 Initiate a **Rename** action against a supported object (see the table above). For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying settings in the **Rename** dialog:

Step	Settings and tasks	
Rename	**Name**	Provide the new name for the object.
Dependencies	Review the referring and referred objects for which naming will be automatically resolved when you execute the renaming operation.	
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.	

3 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

REORGANIZE /REORG

Rapid SQL offer Reorganize or Reorg options against the following DBMS platform/object types:

- [Reorganizing IBM DB2 for Linux, Unix, and Windows Objects](#)

- [Reorganizing Oracle Objects](#)

- [Reorganize (SQL Server indexes, primary keys, and unique keys)](#)

- [Reorganize (SQL Server Full-text Catalogs)](#)

- [Reorganize (Sybase Indexes and Tables)](#)

- [Reorg Index (DB2 z/OS)](#)

- [Reorg (DB2 z/OS Tablespaces)](#)

REORGANIZING IBM DB2 FOR LINUX, UNIX, AND WINDOWS OBJECTS

Reorganize Dialog Box (One Table)

The table below describes the options and functionality on the Reorganize dialog box.

Step	Description
Action Options	• Choose a **Reorg Type** of table or all indexes of the table. • Optionally, choose an **Index Schema** and specify an **Index name**. • Enable or disable **Inplace Reorg** and if you enable, choose an **Inplace reorg mode** of Start, Stop, Pause, or Resume. • Select an **Access mode** to control read and write access. • Select an **Index reorg mode** to clean up empty pages, delete after cleaning up empty pages, or convert to a type 2 index. • If you did not enable **Inplace Reorg**, select a **Tablespace**. • Select a **Long Tablespace**. • Enable or disable the following options: **Index scan**, **Reorg long field and LOB data**, **Reset Dictionary**, **No truncate table**, and **Reorganize all partitions**.
Partitions	If you did not enable **Reorganize all partitions**, select the partitions to reorganize.
Preview	Preview the DDL code generated from your choices.

Reorganize Dialog Box (Multiple Tables)

The table below describes the options and functionality on the Reorganize dialog box.

Option	Description
Temporary Tablespace	Associates a temporary tablespace with the table's tablespace. You can select another tablespace from the list.

REORGANIZE (SQL SERVER INDEXES, PRIMARY KEYS, AND UNIQUE KEYS)

The **Reorganize Indexes** dialog box lets you reorganize an entire index, primary key, or unique key or a single partition of that object. It also lets you specify a LOB_COMPACTION option.

> **NOTE:** This functionality is available for Microsoft SQL Server 2005 and above.

> **NOTE:** You cannot reorganize an index (primary key, or unique key) that has an **Allow Page Locks** property set to FALSE. For information on setting index properties, see <u>Indexes Editor (SQL Server)</u>.

To reorganize an index

1 On the *Datasource Explorer*, expand nodes until Microsoft SQL Server index nodes, primary keys, or unique keys are displayed.

2 Right-click an index, primary key, or unique key and select **Reorganize** from the context menu. The **Reorganize Index** wizard opens.

3 Use the following table as a guide to understanding and modifying settings in the wizard:

Step	Settings and tasks
Action options	To reorganize a single partition, specify a **Partition number** and optionally, specify **Lob Compaction** used to create the REORGANIZE WITH clause.
	To reorganize the entire index, primary key, or unique key, DO NOT provide a **Partition Number**, and optionally, specify **Lob Compaction** used to create the REORGANIZE WITH clause.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

4 Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

REORGANIZING ORACLE OBJECTS

The Reorganize dialog box lets you reduce query processing time against tables.

To reorganize an Oracle table

1   Initiate a **Reorganize** action against a table. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying settings in the **Reoganize** dialog:

Panel	Group	Setting and description
Action Options	Reorganize method	**Online** - If selected, reorganization is carried out online, using DBMS_REDEFINITION procedures. DBMS_REDEFINITION is Oracle's online table reorganization package and is available for Oracle 9i. During an online reorganization, the table can still be accessed using DML statements like SELECT and UPDATE. If this check box is unselected, reorganization is carried out using a series of simple ALTER TABLE statements. This method is intended for offline usage.
	Tablespace	If you want to move the table(s) to a new tablespace, select the new tablespace from this list.
	Data Block Storage	Type ALTER TABLE property values for the **Percent Free** (PCTFREE property), **Percent Used** (PCTUSED property), **Initial Transactions** (INITTRANS property), and **Max Transactions** (MAXTRANS property).
	Extents	Type an ALTER TABLE MOVE STORAGE property value for the **Initial Extent** (INITIAL) property). Type ALTER TABLE STORAGE property values for the **Next Extent** (NEXT property), **Percent Increase** (PCTINCREASE property), **Minimum Extents** (NEXT property), and **Maximum Extents** (MAXEXTENTS property).
	Freelists	Use the **Free Lists** and **BufferPool** (DEFAULT, KEEP, RECYCLE) settings to provide values for the FREELISTS and BUFFER_POOL components of a STORAGE clause. Use the **Free List Groups** setting to provide a value for the FREELIST GROUPS component of a STORAGE clause
	Parallel Query	Use the **Parallel Degree** setting to provide a value for the DEGREE component of a PARALLEL clause. This specifies the number of servers used in processing operations.
	Physical	Logging - When disabled, a NOLOGGING clause is added to the ALTER TABLE statement.
Dependencies	For details on using this tab, see [Dependencies](#).	
Preview	For details on using this tab, see [Preview](#).	

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

REORGANIZE (SQL SERVER FULL-TEXT CATALOGS)

This action builds and submits a basic ALTER FULLTEXT CATALOG *catalogname* REORGANIZE statement, with no additional options or arguments. This lets you merge smaller indexes into a larger, master index.

> **NOTE:**   This functionality is available as of SQL Server 2005.

To reorganize a full-text catalog:

1 Initiate a **Reorganize** action against one or more full-text catalogs. For more information, see Initiating an object operation.

2 Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action Options	Displays the catalogs you selected.
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

3 Click **Execute**. For information on the other options, see Preview and Scheduling.

REORGANIZE (SYBASE INDEXES AND TABLES)
The Reorganize dialog box lets you reduce query processing time against tables. This functionality is available for both tables and indexes.

For more information, see

• Reorganizing Sybase ASE Tables

• Reorganize Sybase ASE Indexes

REORGANIZING SYBASE ASE TABLES
This action lets you build and submit one of four REORG subcommands: FORWARDED_ROWS, RECLAIM_SPACE, COMPACT, or REBUILD. REORG optimizes the use of table space and improves performance.

The Reorganize Table dialog box lets you reduce the query processing time against a table by reorganizing the table to ensure that space is properly allocated to it. For lengthy reorganization processes, this dialog box also lets you execute a process in increments, lets you resume an incomplete process, and lets you specify the duration of each increment. For more information, see Incremental Reorganizations.

> **TIP:** Frequent update activity on a table can cause data rows to migrate and to chain over multiple data pages. Chained or forwarded rows can degrade performance because more physical reads are required to access a row of data. Consequently, you should monitor chained rows regularly to spot performance bottlenecks before they become severe. In addition, altering physical storage parameters can lead to fragmentation of space on your data pages, which also results in reduced performance levels.

You should consider reorganizing a table if you are experiencing slow performance due to:

• A large number of chained or forwarded rows on your data pages

• A large amount of fragmentation in your data pages

> **NOTE:** You can reorganize tables in Sybase ASE versions 12 and 12.5.

The table below describes the options and functionality on the Reorganize Table dialog box:

Option	Description
Compact	Lets you reclaim space and undo row forwarding.
	Minimizes interference with other activities by using multiple small transactions of brief duration. Each transaction is limited to eight pages of reorg processing. These three commands also provide resume and time options that allow you to set a time limit on how long a reorg runs and to resume a reorg from the point at which the previous reorg stopped, making it possible to use a series of partial reorganizations at off-peak times to reorg a large table.
	For information on result options, see Incremental Reorganizations.
Reclaim Space	Lets you reclaim unused space resulting from deletions and row-shortening updates on a page.
	Minimizes interference with other activities by using multiple small transactions of brief duration. Each transaction is limited to eight pages of reorg processing. These three commands also provide resume and time options that allow you to set a time limit on how long a reorg runs and to resume a reorg from the point at which the previous reorg stopped, making it possible to use a series of partial reorganizations at off-peak times to reorg a large table.
	For information on result options, see Incremental Reorganizations.
Rebuild	Lets you undo row forwarding and reclaim unused page space. It also rewrites all rows to comply with the target table's clustered index, writes rows to data pages to comply with space management setting changes (via sp_chgattribute), and drops and re-creates all the target table's (or tables') indexes.
	Reorg rebuild holds an exclusive table lock for its entire duration. On a large table this can be a significant amount of time. However, reorg rebuild accomplishes everything that dropping and re-creating a clustered index does and takes less time. In addition, reorg rebuild rebuilds the table using all of the table's current space management settings. Dropping and re-creating an index does not use the space management setting for reservepagegap. In most cases, reorg rebuild requires additional disk space equal to the size of the table it is rebuilding and its indexes.
Undo Row Forwarding	Lets you undo row forwarding, a process that occurs when an update increases a row's length in a data-only-locked table such that the row is too large to fit on its original page.
Options	Start at the point where a previous reorg left off - Select to resume a previously initiated but incomplete partial reorganization. Then specify the duration for which you want the resumed reorganization to continue before stopping again. This box is disabled for the rebuild command.

INCREMENTAL REORGANIZATIONS

If target tables are too long to reorganize in one session, Rapid SQL lets you reorganize them in increments over multiple sessions by specifying a maximum duration for each session. After Rapid SQL reorganizes tables for the specified duration, the operation stops until you resume it again from the Options box of the ReOrganize Table dialog box. The Options box lets you specify to resume a previously initiated but incomplete partial reorganization. It also lets you specify the duration for which you want a resumed reorganization to continue before stopping again. The Option box is disabled for the rebuild command.

NOTE: The duration you specify refers to elapsed time, not CPU time

In the option box, if you select the check box without specifying a duration, Rapid SQL executes the reorg at the point where the previous reorg stopped and continues to the end of the target tables. If you clear the check box and specify a duration, the reorg starts at the beginning of the target tables and continues for the specified number of minutes. If you select the check box and specify a duration, Rapid SQL runs the reorg from the point where it last left off, and continues for the specified number of minutes.

NOTE: If you reorganize a table using one command (Compact, Reclaim Space, or Undo Forwarding) for a specified duration, you cannot resume the process from its resume point using a different command. For example, you cannot compact a table for an hour, and then reclaim space on the remainder of the table. A resumed reorganization process must utilize the same command from start to finish. Selecting a different command begins a new reorganization process.

CAUTION: While this option lets you reorganize a large table in multiple manageable pieces, any updates to the table between reorganization runs can cause pages to be skipped or processed more than once.

REORGANIZE SYBASE ASE INDEXES

The Reorganize Index dialog box lets you reduce the query processing time against a table by running a reorg rebuild command on the target index.

This operation:

- Undoes row forwarding and reclaim unused page space

- Rewrites all rows in the table to comply with the table's clustered index

- Writes rows to data pages to comply with space management setting changes (via sp_chgattribute)

- Drops and re-creates the table's indexes

Reorg rebuild holds an exclusive table lock for its entire duration. On a large table this can be a significant amount of time. However, reorg rebuild accomplishes everything that dropping and re-creating a clustered index does and takes less time. In addition, reorg rebuild rebuilds the table using all of the table's current space management settings. Dropping and re-creating an index does not use the space management setting for reservepagegap. In most cases, reorg rebuild requires additional disk space equal to the size of the table it is rebuilding and its indexes.

REORG INDEX (DB2 Z/OS)

This action lets you build and submit a REORG INDEX utility call.

NOTE: Before using this action consult IBM documentation for details on REORG INDEX utility options. For online access to DB2 documentation, see Accessing Third Party Documentation.

To reorganize an index

1 Initiate a **Set Default** action against a tablespace. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying settings in this wizard. Note that some settings are only available based on other selections.

Setting	Description
Select Indexes	On opening, the list displays the index you chose to reorganize.
	Optionally, you can add more indexes to the list that are to be reorganized by clicking **Add**. Adding indexes using the **Index Selector** dialog is a two-step process. Use the **Database Like**, **Index Creator**, **Index Like**, and **Match Case** controls to provide a search criteria pattern and then click **Query** to display qualified indexes in the **Index Selector** list. Then select one or more indexes from the list, click **Add**, and close the dialog.
	To remove a selected index from the list to be reorganized, select the index in the list and click **Delete**.
Do you want the utility to be reusable	Select this check box to make the utility restartable.
Share Level	Lets you select share level that allows users to view but not modify the table data: REFERENCE, CHANGE, or None.
Deadline	Lets you specify a deadline for the switch phase of reorganization to start. If Rapid SQL estimates that the switch phase will not start by the deadline, Rapid SQL terminates reorganization.
	None - Specifies that there is no deadline for the read-only iteration of log processing.
	Timestamp - Specifies the deadline for the switch phase to start processing.
	labeled-duration-expression - Click the ... button to open a dialog that lets you build a custom CURRENT_TIMESTAMP or CURRENT_DATE value.
Drain Specification	Lets you specify that DB2 drains the write claim class after the delay. The number of log records, and thus the estimated time, for a future iteration of log processing will be 0.
Fast Switch	Keeps the data set name unchanged and updates the catalog to reference the newly reorganized data set.
Maxro (only available with a **Share Level** of CHANGE)	Lets you DEFER or specify the maximum amount of time for the last iteration (read-only access iteration) of log processing.
Drain (only available with a **Share Level** of CHANGE)	Lets you specify drain behavior (WRITERS or ALL) at the end of the log phase after the MAXRO threshold is passed and when the last log iteration is to be applied:
Long Log (only available with a **Share Level** of CHANGE)	Lets you provide a LONGLOG value of CONTINUE, DRAIN, or TERM
Delay (only available with a **Share Level** of CHANGE)	Lets you specify the minimum interval between the time REORG sends the LONGLOG message to the console and the time REORG performs the action specified by the **Long Log** setting.
Timeout (only available with a **Share Level** of CHANGE)	Lets you specify a timeout option of ABEND or TERM.

Setting	Description
Leaf dist limit	Lets you specify the value that is to be compared to the LEAFDIST value for the specified partitions of the index in SYSIBM.SYSINDEXPART. If any LEAFDIST value exceeds the specified **Leaf dist limit**, REORG is performed or, if you specify REPORTONLY, recommended.
Report	If selected, REORG is only recommended.
Unload	Specify a continue or terminate value of CONTINUE, ONLY, or PAUSE.
Do you want to specify the stats option	Specifies that statistics are to be collected and either reported or stored in the DB2 catalog.
Do you want to output message to SYSPRINT	Specifies that a set of messages is to be generated to report the collected statistics and output to SYSPRINT.
Do you want to specify the correlation stats option	Lets you specify a correlation-stats-spec for statistics reporting.
Do you want to force aggregation or rollup processing to be done event though some parts do not contain data	Specifies whether statistics are to be aggregated or rolled up when RUNSTATS is executed.
Work DDN	Lets you specify a DD name or a TEMPLATE name from a previous TEMPLATE control statement.
Would you like to preformat	Lets you specify that remaining pages are to be preformatted up to the high-allocated RBA in the index space.

3 When ready, click **Finish**.

REORG (DB2 z/OS TABLESPACES)

> **NOTE:** This functionality is available for IBM DB2 for OS/390 and z/OS only.

The Reorganize Tablespace Wizard reorganizes a tablespace to improve access performance and reclaim fragmented space. The wizard can also reorganize a single partition or range of partitions of a partitioned table space. You can specify the degree of access to the data during reorganization, and optionally collect inline statistics.

Completing the Reorganize Tablespace Wizard

To complete the Reorganize Tablespace Wizard, do the following:

1 On the *Datasource Explorer*, select the tablespaces node.

   Rapid SQL displays the tablespaces in the right pane of the Explorer window.

2 On the *Datasource Explorer* tool bar, click Command, and then select Reorganize.

   Rapid SQL opens the Reorganize Tablespace Wizard.

3 Complete the wizard panels, and then click Finish.

   Rapid SQL opens the <u>Preview</u> dialog box.

REORGANIZE TABLESPACE WIZARD - PANEL 1

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Do you want the utility to be restartable?	Select to make the utility restartable.
Database	Displays the list of databases in the system. You can select the tablespace database or click the filter button to limit the list of databases to a string pattern.
Tablespace	Displays the list of tablespaces.
Partition	Lets you specify a partition or a range of partitions.
Reuse	Select to reuse this tablespace.

REORGANIZE TABLESPACE WIZARD - PANEL 2

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Log	List of the sequence of events after you execute the wizard.
Sort Data	Sorts columns alphanumerically in single result sets.
No Sysrec	Select Yes for no sysrec.
Sortkeys	Select Yes for sortkeys. Specifies that index keys are to be sorted and built in parallel during the SORTBLD phase to improve performance.

REORGANIZE TABLESPACE WIZARD - PANEL 3

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Share level	Lets you select share level that allows users to view but not modify the table data: None, Reference, or Change.
Deadline	Lets you specify a deadline for the switch phase of reorganization to start. If Rapid SQL estimates that the switch phase will not start by the deadline, Rapid SQL terminates reorganization. None - Specifies that there is no deadline for the read-only iteration of log processing. Timestamp - Specifies the deadline for the switch phase to start processing. Click the button to open the Deadline Expression Builder dialog box.
Drain Specification	Lets you specify that DB2 drains the write claim class after the delay. The number of log records, and thus the estimated time, for a future iteration of log processing will be 0.
Fast Switch	Keeps the data set name unchanged and updates the catalog to reference the newly reorganized data set.

Deadline Expression Builder

The table below describes the options and functionality on the Deadline Expression Builder dialog box:

Option	Description
Current Date	Lets you select today as the basis of the deadline, click + or -, type the number of years, months, days, hours, minutes, seconds and microseconds.
Current Timestamp	Lets you select the current timestamp as the basis of the deadline, click + or -, type the number of years, months, days, hours, minutes, seconds and microseconds.

REORGANIZE TABLESPACE WIZARD - PANEL 4

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Local Site Primary	Lets you specify the Local Site Primary.
Local Site Backup	Lets you specify the Local Site Backup.
Recovery Site Primary	Lets you specify the Recovery Site Primary.
Recovery Site Backup	Lets you specify the Recovery Site Backup.

REORGANIZE TABLESPACE WIZARD - PANEL 5

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Off Pos Limit	The specified value is compared to the result of the calculation (NEAROFFPOSF + FAROFFPOSF) × 100 / CARDF for the specified partitions in SYSIBM.SYSINDEXPART for the explicit clustering indexes for every table in the specified table space. Alternatively, the values in SYSINDEXPART are checked for a single non-partitioned table space, or for each partition if you specified an entire partitioned table space as the target object. If at least one calculated value exceeds the OFFPOSLIMIT value, REORG is performed or recommended. NOTE: This option is valid for non-LOB table spaces only.
Ind Ref Limit	The specified value is compared to the result of the calculation (NEARINDREF + FARINDREF) × 100 / CARDF for the specified partitions in SYSIBM.SYSTABLEPART for the specified table space. Alternatively, the values in SYSTABLEPART are checked for a single non-partitioned table space, or for each partition if you specified an entire partitioned table space as the target object. If at least one calculated value exceeds the INDREFLIMIT value, REORG is performed or recommended. NOTE: This option is valid for non-LOB table spaces only.
Report Only	The reorganization is only recommended, not performed.

Option	Description
Unload	Specifies whether the utility job is to continue processing or end after the data is unloaded.
	Continue - Specifies that, after the data has been unloaded, the utility continues processing.
	Pause - Specifies that after the data has been unloaded, processing ends.
	Only - Specifies that after the data has been unloaded, the utility job ends and the status in SYSIBM.SYSUTIL corresponding to this utility ID is removed.
	External - Specifies that after the data has been unloaded, the utility job is to end and the status in SYSIBM.SYSUTIL corresponding to this utility ID is removed.

REORGANIZE TABLESPACE WIZARD - PANEL 5

> **NOTE:** This panel is displays only if the share level is Change.

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Table Owner	Displays the schema names in the system.
Table	Displays all tables for the selected schema.
Maxrow	Lets you specify the maximum amount of time for the last iteration of log processing. During that iteration, applications have read-only access. Type an integer value or select DEFER. DEFER Specifies that the iterations of log processing with read/write access can continue indefinitely.
Drain	Lets you select Writers or All. Specifies that Rapid SQL drains the write claim class after the delay. The number of log records, and thus the estimated time, for a future iteration of log processing will be 0.
Long Log	Lets you specify the action that Rapid SQL performs (after sending the LONGLOG message to the console) if the number of log records that are processed during the next iteration is not sufficiently lower than the number of log records that were processed during the previous iterations.
	Continue - Specifies that Rapid SQL continues performing reorganization.
	Term - Specifies that Rapid SQL terminates reorganization after the delay.
	Drain - Specifies that Rapid SQL drains the write claim class after the delay. The number of log records, and thus the estimated time, for a future iteration of log processing will be 0.
Delay	Lets you type a lower bound for the interval between the time when REORG sends the LONGLOG message to the console and the time when REORG performs the action specified by the LONGLOG parameter.
Timeout	Lets you select timeout option:
	Abend - Abnormal end of task
	Term - Termination

REORGANIZE TABLESPACE WIZARD - PANEL 6

> **NOTE:** This panel is displays only if the share level is Reference or Change.

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Local Site Primary Copy	Lets you specify the Local Site Primary Copy.
Local Site Backup Copy	Lets you specify the Local Site Backup Copy.
Recovery Site Primary Copy	Lets you specify the Recovery Site Primary Copy.
Recovery Site Backup Copy	Lets you specify the Recovery Site Backup Copy.

REORGANIZE TABLESPACE WIZARD - PANEL 7

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Off Pos Limit	The specified value is compared to the result of the calculation (NEAROFFPOSF + FAROFFPOSF) × 100 / CARDF for the specified partitions in SYSIBM.SYSINDEXPART for the explicit clustering indexes for every table in the specified table space. Alternatively, the values in SYSINDEXPART are checked for a single non-partitioned table space, or for each partition if you specified an entire partitioned table space as the target object. If at least one calculated value exceeds the OFFPOSLIMIT value, REORG is performed or recommended. NOTE: This option is valid for non-LOB table spaces only.
Ind Ref Limit	The specified value is compared to the result of the calculation (NEARINDREF + FARINDREF) × 100 / CARDF for the specified partitions in SYSIBM.SYSTABLEPART for the specified table space. Alternatively, the values in SYSTABLEPART are checked for a single non-partitioned table space, or for each partition if you specified an entire partitioned table space as the target object. If at least one calculated value exceeds the INDREFLIMIT value, REORG is performed or recommended. NOTE: This option is valid for non-LOB table spaces only.
Report Only	The reorganization is only recommended, not performed.
Unload	Specifies whether the utility job is to continue processing or end after the data is unloaded. Continue - Specifies that, after the data has been unloaded, the utility continues processing. Pause - Specifies that after the data has been unloaded, processing ends. Only - Specifies that after the data has been unloaded, the utility job ends and the status in SYSIBM.SYSUTIL corresponding to this utility ID is removed. External - Specifies that after the data has been unloaded, the utility job is to end and the status in SYSIBM.SYSUTIL corresponding to this utility ID is removed.

REORGANIZE TABLESPACE WIZARD - PANEL 6

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Do you want to keep the dictionary?	A collection of language-related linguistic information used during text analysis, indexing, retrieval, and highlighting of documents in a particular language. NOTE: Available only if Continue or Pause option was specified for Unload on the previous panel.
Do you want to specify statistics?	Lets you include statistics for the tablespace in the SQL statement. Enabled only if Continue or Pause option was specified for Unload on the previous panel.
Do you want to output message to SYSPRINT?	Lets you specify to output message to SYSPRINT.
Do you want to force aggregation or rollup processing to be done even though some parts do not contain data?	Lets you specify to process with forced aggregation, gathered into a mass, or rollup.

REORGANIZE TABLESPACE WIZARD - STATISTICS TABLES

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Select Tables	Lets you select tables (and columns) to collect statistics for. The sample column is editable and you can type an integer value into this column.
Add	Opens the Table Selector Dialog Box.
Set Properties	Lets you customize the selection of columns in the selected tables.

REORGANIZE TABLESPACE WIZARD - STATISTICS INDEXES

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Select Index	Lets you specify the indexes and the correlation values for collecting the statistics. The sample column is editable and you can type an integer value into this column.
Add	Opens the Index Selector Dialog Box.
Set Properties	Lets you specify the correlation information for each index.

REORGANIZE TABLESPACE WIZARD - PANEL 7

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Punch DDN	Lets you select punch DDN from the list. Contains generated LOAD statements that let you reload the discarded data. Required if Unload External option is chosen on Panel 5.
Discard DDN	Lets you select discard DDN from the list.
Unload DDN	Lets you select DD name of the unload data set. Required unless NOSYSREC on Panel 2 or Share level CHANGE Panel 3 is selected.
Work DDN	Lets you select the DD statement for a temporary data set used for intermediate output.

REORGANIZE TABLESPACE WIZARD - PANEL 7

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Discard	Lets you select discard DDN from the list.
No Pad	Rapid SQL does not add a character to fill empty space.
Add	Opens the Table Selector Dialog Box.
Set Condition	Lets you customize the selection of columns in the selected tables.

REORGANIZE TABLESPACE WIZARD - PANEL 8

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard

Option	Description
Would you like to specify the device type?	Lets you specify the device type.
Do you want to specify the sort number?	Lets you type the number of datasets to be dynamically allocated by the sort program. The range of the text field is 0 to 2147483647.
Preformat	Lets you specify that the remaining pages are preformatted up to the high allocated RBA in the table space and index spaces associated with the table specified in table-name. The preformatting occurs after the data has been loaded and the indexes are built.

REORGANIZE TABLESPACE WIZARD - PANEL 9

The table below describes the options and functionality on this panel of the Reorganize Tablespace Wizard:

Option	Description
Punch DDN	Lets you select punch DDN from the list. Contains generated LOAD statements that let you reload the discarded data. Required if Unload External option is chosen on Panel 5.
Discard DDN	Lets you select discard DDN from the list.
Unload DDN	Lets you select DD name of the unload data set. Required unless NOSYSREC on Panel 2 or Share level CHANGE Panel 3 is selected.
Work DDN	Lets you select the DD statement for a temporary data set used for intermediate output.

Condition Dialog Box

The Condition Dialog Box lets you type free form condition text.

> **NOTE:** Rapid SQL does not test the correctness of the condition text.

REPAIR TABLES

This action builds and submits a REPAIR TABLE statement, letting you repair a corrupted table. This operation returns a standard REPAIR TABLE result set. Each returned message consists of **Table** (name), **Op** (with a value of REPAIR), **Msg_type** (STATUS, ERROR, INFO, WARNING), and **Msg_text** components.

> **NOTE:** This functionality is available for MySQL only.

To repair a possibly corrupted table:

1 On the Datasource Explorer, expand nodes until the **Tables** node is visible.

2 Expand the **Tables** node.

3 Right-click one or more selected tables and select **Repair Tables** from the context menu.

Rapid SQL opens the Repair Tables dialog box.

4 Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks	
Action Options	**Quick**	Only the index tree is fixed.
	Extended	Creates the index row by row instead of creating one index at a time with sorting.
	Use FRM file	Recreates the `.MYI` file header using information in the `.frm` file.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.	

Step	Settings and tasks
Preview	Displays the DDL that will execute the object action.

5   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

The results set opens on the **Results** tab of an ISQL editor window. For more information, see [Results Editor](#).

REPORT

Rapid SQL lets you generate a detailed report of one or more objects of a given type for a single database. For each object, the report contains information similar to that available in the object editor for that object type. The report opens in Rapid SQL's built-in HTML browser and is also written to an .htm file.

> **NOTE:**   This functionality is available for DB2 LUW, DB2 z/OS, Oracle, SQL Server, and Sybase.

To generate a Detail Report on one or more objects

1   On the Datasource Explorer, select the target object node.

Rapid SQL displays the target objects in the right pane of the Explorer window.

2   In the right pane of the application, right-click one or more selected objects, and then select **Report** from the context menu.

Rapid SQL opens the Report dialog box.

3   In **Report Home Page File Name**, type the report name or click **Browse** to locate the report.

4   Use the **Append date and time stamp to report file name** and **Save report in date specific folders** controls to ensure that the disk files are not overwritten each time you generate reports against the specified file name.

5   In **Report Title**, type the title that is to appear in the report.

6   If you selected multiple objects, use the **Report Options** controls to specify a **Single HTML File for All Objects** or **Separate HTML Files for Each Object**.

7   Click **Execute**.

Rapid SQL opens the report in the built-in browser and writes a copy to disk.

RESTART

The Restart Sequence dialog box lets you restart a sequence, starting at the value specified in the sequence definition or starting at a specified value.

> **NOTE:**   This functionality is available for IBM DB2 for Linux, Unix, and Windows only.

To rebuild a stored outline:

1   On the *Datasource Explorer*, select the Sequences node.

    Rapid SQL displays the sequences in the right pane of the Explorer window.

2   In the right pane of the application, right-click a sequence and select Restart from the context menu.

    Rapid SQL opens the **Restart Sequence** dialog box.

3   Select one of the following **Restart Sequence at** options:

    • **Originating Value** use the value specified in the object definition

    • **This value** lets you specify a custom start value

4   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

RUN JOB

Rapid SQL lets you immediately run any defined job queue, regardless of its current schedule. Job Queues are built-in mechanisms that let you schedule a variety of SQL-based or command-line driven tasks.

> **NOTE:**   This functionality is available for Oracle only.

To run a job

1   On the *Datasource Explorer*, select the Job Queues node.

    Rapid SQL displays the job queues in the right pane of the Explorer window.

2   In the right pane of the application, right-click the target job queue, and then select Run Job.

    Rapid SQL opens the **Run** dialog box.

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

For related information, see [Enable Job (Job Queue)](#) and [Disable Job](#).

SCHEMA

The Schema dialog box lets you view key properties of the columns in a table or view.

> **NOTE:**   This functionality is available for all DBMS platforms.

To view a column summary for a table or view

1   On the *Datasource Explorer*, select the target object node.

    Rapid SQL displays the target objects in the right pane of the Explorer window.

2   In the right pane of the application, right-click the target object, and then select Schema from the context menu.

Rapid SQL opens Schema results in a Results Editor. For information on functionality offered, see Results Editor.

SELECT * FROM

This action lets you retrieve all data contained in tables or views. Results are displayed in a row and column grid.

> **NOTE:**   This functionality is available for all DBMS platforms.

To select all data in one or more tables or views

1   Initiate a **Select * From** action against one or more tables or views. For more information see Initiating an object operation.

For each selected table or view, Rapid SQL opens a two-tabbed window:

- The **Results** tab displays the data selected from the table, in a grid similar to a spreadsheet. Right-click and toolbar options offer many of the functions provided in the Results editor. For more information, see Results Editor.

- The **Query** tab displays the full query generated for the operation. Right-click and toolbar options offer many of the functions provided in the ISQL editor. For more information, see ISQL Editor.

SET DEFAULT

This function lets you set a tablespace as the default tablespace. Users created without a specified default tablespace will be assigned this default tablespace. If no default tablespace is set, users created without a specified tablespace will have their default tablespace set to SYSTEM.

> **NOTE:**   This functionality is available for Oracle 10g and above.

To set a tablespace as the default

1   Initiate a **Set Default** action against a tablespace. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Set Default** wizard:

Step	Settings and tasks
Action options	Verify that the panel displays the tablespace to be set as default.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation.

Step	Settings and tasks
Preview	Preview the DDL generated for the operation and when ready, use the **Schedule** or **Execute** button to perform this action.

SET INTEGRITY

Rapid SQL provides an implementation of the DB2 SET INTEGRITY statement, letting you perform tasks such as taking tables into and out of set integrity pending state, placing tables into full access state, and pruning the contents of one or more staging tables.

> **NOTE:** This functionality is available for IBM DB2 LUW only.

> **NOTE:** This operation should not be performed without an in-depth understanding of the SET INTEGRITY statement. For online access to DB2 documentation, see Accessing Third Party Documentation.

To Open the Set Integrity Wizard

1 On the Explorer, expand a DB2 database, and click the Table node.

2 Select the appropriate table(s), right-click, and choose Set Integrity from the menu. You can select more than one table by clicking CTRL + the tables you want.

   OR

   Select the appropriate table, click Command on the toolbar, and then choose Set Integrity from the drop-down menu.

The table below lists all fields you can see in the Set Integrity Wizard. Depending on the options you choose, you may not see them all.

Required Field	Description
Tables	
Tables	The list of one or more tables you selected appear.
Integrity Option	

Required Field	Description
Integrity Option	**OFF**--When selected, tables have their foreign key and check constraints, and column generation disabled and so are put in a pending state. This also means materialized query or staging tables are not immediately refreshed and are put in a pending state.
	TO DATALINK RECONCILE PENDING--When selected, DATALINK integrity is disabled and tables are put in a check pending no access state. Dependent and descendant tables are not affected.
	IMMEDIATE CHECKED--This turns on a table's integrity checking turned on. Any checking that was deferred is carried out.
	FULL ACCESS--Tables become fully accessible as soon as the SET INTEGRITY statement executes.
	PRUNE--This is appropriate only for staging tables. The content of the staging table is pruned and set to an inconsistent state. If any table in the table-name list is not a staging table, an error is returned.
	UNCHECKED--Allows you turn on some or all integrity checking but the table will not be checked for integrity violations. This can affect data integrity.
Table Readability/Cascade/Descendent Types	
Specifies the readability of the table while it is in check pending state:	**NO ACCESS**--The table(s) are put in a check pending no access state so read/write access to the table is prohibited.
	READ ACCESS--The table(s) are put in a check pending read state. This allows read access to the non-appended portions of any tables.
Specifies whether to be immediately cascaded to all descendents	**CASCADE IMMEDIATE**--The check pending state for foreign key constraints is immediately extended to descendant foreign key constraints or to materialized query or staging tables.
	CASCADE DEFERRED--Only the selected tables are put in the check pending state. Descendant foreign key, materialized query, or staging tables remain unchanged.
Descendent Types	**Materialized Query Tables**--When selected, the check pending state is immediately cascaded to only descendant materialized query tables.
	Foreign Key Tables--When selected, the check pending state is cascaded immediately only to descendant foreign key tables.
	Staging Tables--When selected, the check pending state is cascaded immediately only to descendant staging tables.
Check Appended Portion?	
Do you want to check on the appended portion (if any) of the table?	**Default/Yes/No**
	Force Generated--If you do not specify this generated column current values will be compared to the computed value of the expression as if an equality check constraint existed. If this is specified, generated columns are computed only for the appended portion.
	Prune--Possible only for staging tables. When you check this, the contents of the staging table are pruned and the staging table is set to an inconsistent state.
	Full Access--When selected, tables will become accessible after the SET INTEGRITY statement executes.
Specify Exception Tables	
List of Base Tables	Any row that is in violation of a foreign key or check constraint is copied to the exception table you select.
Integrity Options: IMMEDIATE UNCHECKED	

Required Field	Description
IMMEDIATE UNCHECKED options	**Foreign Key**--These constraints are turned on when the table is removed from check pending status.
	Check--Check constraints are turned on when the table is removed from check pending status.
	Datalink Reconcile Pending--DATALINK integrity constraints are turned on when the table is removed from check pending status.
	Materialized Query--Immediate refreshing is turned on for a materialized query table when it is removed from a check pending state.
	Generated Column--When the table is removed from check pending status, generated columns are turned on.
	Staging--Immediate propagation is turned on for a staging table.
Do you want tables to become fully accessible after the SET INTEGRITY statement executes?	**Yes/No**

3   When ready, click the **Finish** button to preview and submit the generated DDL. For more information, see [Preview](#).

SET ONLINE/OFFLINE

The Set Database(s) Online/Offline dialog box lets you disable your databases to prevent access, and enable your databases to grant access through the Datasource menu.

> **NOTE:**   This functionality is available for Microsoft SQL Server and Sybase ASE only.

Important Notes

For Sybase, Rapid SQL only lets you set databases online.

To set one or more databases online or offline

1   Initiate a **Set Online/Offline** action against one or more databases. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to working through the panels of the **Set Online/Offline** dialog box

Step	Settings and tasks	
Action options	**Set offline** (SQL Server only)	Set this to TRUE to set the database offline or set it to True to set the database online.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).	
Preview	Displays the DDL that will execute the object action. For details, see [Preview](#).	

3   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

SET STATISTICS

This action creates and submits a SET STATISTICS statement, letting you recompute the selectivity of an index.

> **NOTE:** The action is available for Interbase/Firebird.

To recompute the selectivity of an index:

1 On the *Datasource Explorer*, expand nodes until the **Indexes** node is visible, and then expand the **Indexes** node.

2 Select one or more indexes.

3 RIght click the selected indexes and select Set Statistics from the context menu.

   Rapid SQL opens the **Set Statistics** dialog box.

4 Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Displays the indexes you selected.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Displays the DDL that will execute the object action.

5 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

SET UNDO

Set UNDO Tablespace dialog box lets you dynamically set an UNDO tablespace if the tablespace is running in AUTO UNDO mode.

> **NOTE:** This functionality is available for Oracle 9 or later.

To dynamically set an UNDO tablespace

1 Initiate a **Set Undo** action against a tablespace. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying the settings on the **Set Undo Tablespace** wizard:

Step	Functionality
Action Options	Displays the tablespace you selected to be set as an Undo tablespace.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).
Preview	Preview the DDL generated from your choices.

3 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

SHRINK

For details on using Rapid SQL Shrink operations to save or reclaim space, see the following topics:

- Shrinking SQL Server databases

- Shrinking Oracle rollback segments

- Shrinking Oracle tables or indexes

SHRINKING SQL SERVER DATABASES

Rapid SQL lets you reclaim space from a database that is too large.

To shrink a SQL Server database

1 Initiate a **Shrink** action against one or more databases. For more information see Initiating an object operation.

2 Use the following table as a guide to understanding and modifying the settings on the **Shrink** wizard:

Step	Description	
Action Options	Displays the databases you selected and lets you work with the following settings:	
	Move data pages to beginning of file before shrink	Select to move data pages to the beginning of the file before the shrink.
	Release All Unused Space	Deselect to set the target free space to retain, and then in the Target Free Space to Retain (percent) box, type the new value of free space to retain.
		The new size for the database must be at least as large as the Minimum Allowable Size displayed in the Current File Size box.
	Target free space percent after shrink	Lets you specify the target free space percent after the shrink.
Preview	Preview the DDL generated from your choices.	

3 Click **Execute**. For information on the other options, see Preview and Scheduling.

SHRINKING ORACLE ROLLBACK SEGMENTS

Rapid SQL lets you shrink the size of rollback segments. The proper sizing of rollback segments is critical to their overall performance. Performance degrades whenever a rollback segment must extend, wrap or shrink in response to transaction loads. Ideally, you want to make the extents of rollback segments as small as possible while still ensuring that each transaction can fit into a single extent.

After an abnormally large transaction load, you might consider shrinking a rollback segment to eliminate unnecessary space. Oracle lets you shrink a rollback segment manually by a specific amount or back to its Optimal Size.

Important Notes

For Oracle 9 or later, Shrink is not available if auto-UNDO management is enabled.

To shrink the size of one or more rollback segments

1 Initiate a **Shrink** action against one or more rollback segments. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying the settings on the **Shrink** wizard:

Step	Description	
Action Options	Displays the rollback segments you selected and lets you work with the following settings:	
	Specify the size...	If you do not provide a specific number the Rollback Segment uses the OPTIMAL value specified in the Storage clause. If an OPTIMAL value is not specified, the size defaults to the MINEXTENTS value of the Storage clause.
Preview	Preview the DDL generated from your choices.	

3 Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

SHRINKING ORACLE TABLES OR INDEXES

Rapid SQL lets you shrink the size of tables or indexes.

Important Notes

Shrink is only available for tables in which the **Row Movement** property is enabled. For information on setting the **Row Movement** when creating or editing a table, see [Table Wizard (Oracle)](#) and [Tables Editor (Oracle)](#).

To shrink the size of one or more indexes or tables

1 Initiate a **Shrink** action against one or more indexes or tables. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying the settings on the **Shrink** wizard:

Step	Description	
Action Options	Displays the objects you selected and lets you work with the following settings:	
	Compact	Enable **Compact** to restrict shrinking to defragmenting the segment space and compact rows.
	Cascade	Enable **Cascade** to simultaneously shrink all dependent objects.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).	
Preview	Preview the DDL generated from your choices.	

3   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

START DATABASE

The Start Database dialog box lets you start a database:

- When the database that has been stopped with a **Stop Database** dialog box. For more information, see <u>Stop Database</u>.

- After a tablespace, partition, or index has been placed in group buffer pool RECOVER pending status (GRECP) or if pages have been put on the logical page list (LPL) for that object.

  NOTE: This functionality is available for IBM DB2 for OS/390 and z/OS only.

Depending on the specified options, the database can be made available for read-only processing, read-write processing, or utility-only processing. In a data sharing environment, the command can be issued from any DB2 on the group that has access to the database.

To start a database

1   On the *Datasource Explorer*, expand the Databases node.

Rapid SQL displays objects of the selected type in the right pane of the Explorer window.

2   In the right pane, right-click the target object and select Start Database from the context menu.

Rapid SQL opens the **Start Database** dialog.

3   Use the following table as a guide to understanding and settings options on this dialog:

Option	Functionality
Database Grid	The Partition column is editable and can include one or more unique numeric values separated by commas. Initially the Partition column is blank. No validation is made for the correctness of partition numbers so make sure that the partitions exist.
Access	Lets you select access options of **Read/Write**, **Read only**, **Utility** only or **Force**.

4   Click **Execute**. For information on the other options, see <u>Preview</u> and <u>Scheduling</u>.

STOP DATABASE

The Start Database dialog box lets you stop a database, optionally letting you allow running applications access until their next COMMIT.

  NOTE: This functionality is available for IBM DB2 for z/OS only.

To stop a database

1   On the *Datasource Explorer*, expand the Databases node.

    Rapid SQL displays objects of the selected type in the right pane of the Explorer window.

2   In the right pane, right-click the target object and select Stop Database from the context menu.

    Rapid SQL opens the **Stop Database** dialog.

3   Use the following table as a guide to understanding and settings options on this dialog:

Option	Functionality
Database Grid	The Partition column is editable and can include one or more unique numeric values separated by commas. Initially the Partition column is blank. No validation is made for the correctness of partition numbers so make sure that the partitions exist.
Stop at COMMIT	Enabling this option sends the STOP DATABASE statement.with an AT(COMMIT) clause, allowing running applications to continue access until their next commit.

4   Click **Execute**. For information on the other options, see Preview and Scheduling.

SWITCH ONLINE

> **NOTE:**   The Switch Online functionality is available for IBM DB2 for Linux, Unix, and Windows only.

The Switch Online dialog box lets you access a tablespace by switching it online after the parent container(s) have been recovered or restored.

To switch a tablespace online:

1   On the *Datasource Explorer*, select the **Tablespaces** node.

    Rapid SQL displays the target objects in the right pane of the Explorer window.

2   In the right pane, right-click one or more selected tablespaces and then select Switch Online from the context menu.

    Rapid SQL opens the Switch Online dialog box.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

TRANSFER OWNERSHIP

Rapid SQL lets an object owner or a user with SECADM authority transfer ownership of an object to another user. The new user is automatically granted the same privileges as the former owner. Ownership can be transferred on an object-by-object basis or you can transfer all objects currently owned by an individual user.

> NOTE: This functionality is available for IBM DB2 for Linux, Unix, and Windows version 9.1 and higher.

The following table lists the object types that support ownership transfer:

Aliases	Check Constraints	Event Monitors	Foreign Keys
Functions	Indexes	Primary Keys	Procedures
Materialized Query Tables	Schema	Sequences	Structured Types
Tables	Tablespaces	Triggers	Unique Keys
User Datatypes	Views		

To transfer ownership of an object

1 Initiate a **Transfer Ownership** action against one or more supported objects (see table above). For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying settings in the **Transfer Ownership** wizard:

Step	Settings and tasks
Action options	If you initiated the ownership transfer against a user, select the specific objects belonging to that user that are to be assigned to a new user.
	Use the **New Owner** dropdown to select a defined user as the new owner of the selected object or objects.
Preview	Preview the DDL generated for the operation and when ready, click **Execute**.. For information on the other options, see [Preview](#) and [Scheduling](#).

TRUNCATE

Rapid SQL lets you quickly delete the rows of one or more tables or clusters. Truncating a table is a faster alternative to deleting all of its rows.

> CAUTION: If you truncate a table, Rapid SQL deletes all the rows. These rows are not logged as individual drops and cannot be recovered from a transaction log or other type of log.

> NOTE: This functionality is available for DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, and Sybase.

To truncate a table or cluster:

1   On the *Datasource Explorer*, select the target object node.

    Rapid SQL displays the target objects in the right pane of the Explorer window.

2   In the right pane of the application, right-click the target object or multiple, selected objects, and then select Truncate or Truncate Table from the context menu.

    Rapid SQL opens the Truncate wizard/dialog box.

3   For help with DBMS-specific settings and additional information, see the following topics:

    • [Notes on truncating Oracle objects](#)

    • [Notes on truncating Sybase ASE objects](#)

    • [Notes on truncating IBM DB2 z/OS objects](#)

4   Click **Execute**. For information on the other options, see [Preview](#) and [Scheduling](#).

NOTES ON TRUNCATING ORACLE OBJECTS

You can truncate any table or cluster in their schema or, if you have the DROP ANY TABLE system privilege, you can truncate any table in any schema.

Observe the following when truncating tables or clusters:

• Before truncating a cluster or table containing a parent key, disable all referencing foreign keys existing in different tables.

• Truncating a cluster or table does not generate any rollback information and commits immediately.

• Oracle alters the storage parameter NEXT to the size of the last extent deleted from the segment.

• Oracle automatically deletes all data in the table's indexes and any materialized view direct-load INSERT information associated with a truncated table.

• If the table is not empty, all associated nonpartitioned indexes and all partitions of associated global partitioned indexes are marked unusable.

• You cannot truncate a hash cluster nor can you truncate individual tables in a hash cluster or an index cluster.

When you truncate a table or cluster, you can specify whether space currently allocated for the table is returned to the containing tablespace or if it is returned to the system. The table below describes the Truncate dialog options available when truncating an Oracle table or cluster:

Option	Description
Drop Storage	Select if you want the freed extents returned to the system where they can be used by other objects.
Reuse Storage	Select if you want the space to remain allocated to the table or cluster you have just truncated.

For more information, see [Truncate](#).

NOTES ON TRUNCATING SYBASE ASE OBJECTS

Truncate against a Sybase table can be applied to the entire table or to a single partition.

> **NOTE:** You cannot truncate a table referenced by a foreign key constraint. Instead, use a DELETE statement without a WHERE clause.

> **TIP:** When you truncate a table, Sybase ASE removes all rows from the target table, but retains the table structure (its indexes, columns, constraints, etc.). The counter used by an identity for new rows is reset to the seed for the column. To retain the identity counter, use a DELETE statement instead of TRUNCATE. To remove the target table definition and its data, use a DROP TABLE statement.

The following table shows the settings on the Truncate Table dialog:

Option	Description
Partition Name (only available with a single table selected)	Selecting a partition name from the dropdown restricts the truncate operation to a single partition.

For more information, see [Truncate](#).

NOTES ON TRUNCATING IBM DB2 Z/OS OBJECTS

When truncating an IBM DB2 z/OS table, the **Action Options** tab offers the following settings:

Setting	Description
Reuse Storage	Corresponds to the DROP STORAGE/REUSE STORAGE clause of a TRUNCATE TABLE statement. This setting specifies whether storage currently allocated to the table is reused or dropped.
Restrict When Delete Triggers	Corresponds to the RESTRICT WHEN DELETE TRIGGERS/IGNORE DELETE TRIGGERS clause of a TRUNCATE TABLE statement. When enabled, an error is returned if triggers are defined for the table. When disabled, triggers defined for the table are not activated by the Truncate operation.
Immediate	Corresponds to the IMMEDIATE clause of a TRUNCATE TABLE statement. If enabled, the truncate operation is executed immediately and cannot be undone. If disabled, a Rollback can undo the Truncate operation.

For more information, see [Truncate](#).

UNBIND FROM TEMPORARY DATABASE

This action builds and submits a **sp_tempdb unbind** system procedure call. It lets you unbind a login from a temporary database.

> **NOTE:** This functionality is available for Sybase only.

To unbind a login from a temporary database

1   Initiate a **Unbind From Temporary Database** action against one or more logins. For more information see Initiating an object operation.

2   Use the following table as a guide to understanding and modifying settings in the **Unbind From Temporary Database** wizard:

Step	Settings and tasks
Action Options	Displays the logins you selected for unbinding.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3   Click **Execute**. For information on the other options, see Preview and Scheduling.

UNQUIESCE

The Unquiesce operation lets you restore user access to a single database or all databases of an instance previously quiesced.

> **NOTE:**   This functionality is available for IBM DB2 for Windows/Unix 8.1 only. Rapid SQL does not support Unquiesce Database for supported servers from IBM DB2 for Windows/Unix 7 or earlier clients.

To restore user access to a database:

1   Take one of the following actions, based on whether you are unquiescing a single database or all databases on an instance:

   • Right-click a datasource node and select **Command > Unquiesce** from the context menu.

   • Right-click the Instance node, select **Command > Unquiesce** from the context menu, and provide credentials if prompted.

   Rapid SQL opens the **Unquiesce Database** or **Unquiesce Database** dialog box.

2   Click **Execute**.

For related information, see Quiesce.

UPDATE STATISTICS

The Update Statistics dialog box lets you update the statistics for an active table or index. As indexes grow and shrink in response to data modification, the accuracy of their statistics can deteriorate.

The following topics provide details on updating statistics by supported DBMS and object type:

   • Updating statistics for tables or indexes (IBM DB2 for Linux, Unix, and Windows)

- [Updating statistics for views (IBM DB2 for Windows, Unix, and Linux)](#)

- [Updating statistics for databases, indexes, or tables (Microsoft SQL Server)](#)

- [Updating statistics for indexes or tables (Sybase)](#)

UPDATING STATISTICS FOR TABLES OR INDEXES (IBM DB2 FOR LINUX, UNIX, AND WINDOWS)

To update statistics for an index or table

1   On the *Datasource Explorer*, expand datasource nodes until the **Tables** or **Indexes** node is visible and then select the Indexes or Tables node.

    Rapid SQL displays the objects of that type in the right pane of the Explorer window.

2   In the right pane of the Explorer window, select one or more indexes or tables.

3   On the *Datasource Explorer* tool bar, click Command, and then select Update Statistics.

    Rapid SQL opens the **Update Statistics** dialog box.

4   Use the following table as a guide to understanding and modifying the settings in this dialog.

Tab	Options	Description
Table Options	**Update table statistics**	Updates table statistics.
	Distribution Options	**Do not collect column statistics** - Column statistics provide information that the optimizer uses to choose the best access plans for queries.
		Collect column statistics on key columns only - Collects column statistics on columns that make up all the indexes defined on the table.
		Collect column statistics on all columns - Collects column statistics for all columns.
		Column statistics provide information that the optimizer uses to choose the best access plans for queries.
		Frequency - Lets you specify the maximum number of frequency values to collect, between 1 and 32767.
		Quantiles - Lets you specify the maximum number of distribution quantile values to collect, between 1 and 32767.
	Column Options	**Do not collect distribution statistics** - Does not collect basic statistics or distribution statistics on the columns.
		Collect distribution statistics on key columns only - Collects both basic statistics and distribution statistics on key columns only.
		Collect distribution statistics on all columns - Collects both basic statistics and distribution statistics on all columns.
		For efficiency both of RUNSTATS and subsequent query-plan analysis, you might collect distribution statistics on only the table columns that queries use in WHERE, GROUP BY, and similar clauses. You might also collect cardinality statistics on combined groups of columns. The optimizer uses such information to detect column correlation when it estimates selectivity for queries that reference the columns in the group.
		Exclude XML columns - Lets you collect statistics on non-XML columns only. XML columns are not included in statistics collection.
Index Options	**Update index statistics**	Lets you enable and disable statistics updates for indexes and controls the following settings:
	Collect extended index statistics	Collects extended index statistics, the CLUSTERFACTOR and PAGE_FETCH_PAIRS statistics that are gathered for relatively large indexes.
	Collect sample statistics	Rapid SQL uses a CPU sampling technique when compiling the extended index statistics. If the option is not specified, every entry in the index is examined to compute the extended index statistics.
Access Options	**Allow read only access during collection**	Allows read only access while Rapid SQL updates the statistics.
	Allow read/ write access during collection	Allows read and write access while Rapid SQL updates the statistics.

5   Click **Execute**. For information on the **Preview** option, see [Preview](#).

UPDATING STATISTICS FOR VIEWS (IBM DB2 FOR WINDOWS, UNIX, AND LINUX)
For IBM DB2 for Windows, Unix, and Linux (version 9), Rapid SQL lets you update the statistics for a view.

To update statistics for a view

1   Initiate an **Update Statistics** action against one or more views. For more information see [Initiating an object operation](#).

2   Use the following table as a guide to choosing options:

Pane	Options
Action Options	Lets you work with the following settings:
	Column Statistics- lets you enable or disable collection of Column Statistics update,
	Distribution Statistics, **Frequency**, and **Quantiles** - let you enable and disable collection of distribution statistics and provide num_freqvalues and num_quantiles RUNSTATS options
	Allow Write Access - lets you enable or disable write access during statistics collection.
Dependencies	Lets you view referencing or referenced objects.
Preview	Lets you view the DDL generated for the operation.

3   **Schedule** or **Execute** the statistics update.

UPDATING STATISTICS FOR DATABASES, INDEXES, OR TABLES (MICROSOFT SQL SERVER)
You can update statistics so that Microsoft SQL Server performs the most efficient query possible. This feature updates statistical information on your database so that the query processor can determine the optimal strategy for evaluating a query. These statistics record the key values used for distribution in an database.

You can use the Update Statistics dialog box if there is significant change in the key values in the database, if a large amount of data in an indexed column has been added, changed, or removed, or if a table has been truncated causing significant changes in the distribution of key values.

The Update Statistics dialog box lets you specify tables and indexes for validation. This dialog box offers different update options depending on your version of Microsoft SQL Server.

> **TIP:**   Avoid updating statistics on your target tables during busy access periods. Microsoft SQL Server locks remote tables and indexes while reading data for update statistics.

For Microsoft SQL Server version 7 or later, the Update Statistics dialog box lets you specify a full or a percentage of a full scan to be used for updating table or index statistics. It also lets you enable or disable future automatic recomputations of statistics. These recomputations are made at Microsoft SQL Server's discretion. When updating statistics for tables, this dialog box also lets you specify the type of statistics you require.

To update statistics for a database, index, or table

1 Initiate an **Update Statistics** action against one or more databases, indexes, or tables. For more information see [Initiating an object operation](#).

2 Use the following table as a guide to understanding and modifying the settings in the **Update Statistics** dialog:

Step	Settings and tasks	
Action Options	**Scan Range** (tables and indexes only)	Full Scan - Select when you want index statistics on every available row.
		Sample Scan - Select when database size prohibits a full scan and you can afford to rely on statistics extrapolated from a sample of all available rows.
	Sample Count (tables and indexes only)	If you specified a **Scan Range** of **Sample Scan**, provide a count.
	Sample Unit (tables and indexes only)	If you specified a **Scan Range** of **Sample Scan**, specify either **%** or **Rows**.
	Statistics Type (tables only)	Index - Select if you only require statistics on the target tables' indexed columns.
		Columns - Select if you require statistics on the target tables in their entirety.
		All existing statistics - Select if you require statistics on the whole database.
	Statistics Recompute (tables and indexes only)	Select if you want Microsoft SQL Server to recompute and update the statistics for the index as part of its normal internal maintenance.
		Deselect if you want the scheduling of future recomputations to be solely your responsibility.
Dependencies (tables and indexes only)	Lets you view referencing or referenced objects.	
Preview	Lets you view the DDL generated for the operation.	

3 Use the **Execute** or **Schedule** button to perform the operation.

UPDATING STATISTICS FOR INDEXES OR TABLES (SYBASE)

The Update Statistics dialog box lets you specify tables and indexes for validation. This dialog box offers different update options depending on your version of Sybase ASE.

> **TIP:** Avoid updating statistics on your target tables during busy access periods. Sybase ASE locks remote tables and indexes while reading data for update statistics.

To update statistics for a database, index, or table

1   Initiate an **Update Statistics** action against one or more tables or indexes. For more
    information see [Initiating an object operation](#).

2   Use the following table as a guide to understanding and modifying the settings in the
    Update Statistics dialog:

Step	Settings and tasks	
Action Options	**Index** (tables only)	Enabling this option updates statistics for indexes of the table.
	Table (tables only)	Enabling this option updates table-specific statistics. Column statistics stored in sysstatistiics are not affected.
	Partition Name (available when a single index or table is selected)	The name of the partition to be updated.
	Step Values	Lets you specify the number of histogram steps.
	Consumers	Specifies the number of consumer processes to be used for a sort when a list of columns is provided and parallel query processing is enabled.
	Sampling Percent	Specifies the percentage of the column to be randomly sampled in order to gather statistics.
Dependencies	Lists referring and referenced objects potentially impacted by the change. For details, see [Dependencies](#).	
Preview	Lets you view the DDL generated for the operation.	

3   Use the **Execute** or **Schedule** button to perform the operation.

SQL SCRIPTING

Rapid SQL incorporates a powerful SQL scripting environment, the ISQL Editor. The ISQL Editor lets you write, debug, test and deploy solid SQL code for your database applications. The scripting environment lets you:

- Open multiple interactive script windows.

- Execute multiple scripts on the same desktop.

- Capture multiple result sets on the same desktop.

Rapid SQL's scripting environment is comprised of different windows:

- ISQL Editor

- DDL Editor

- Results Editor

These windows are context sensitive to the type of script you are opening or extracting. For example, if you extract the schema for a table, Rapid SQL opens a DDL Window containing the script. If you execute a script, a result window displays containing the results of your execution statement.

> **TIP:**   Since you must drop a database object before you can recreate it, you can set the DDL Editor to automatically include DROP statements for specified objects.

> **TIP:**   You can have multiple ISQL windows open at the same time, with each running separate queries.

ISQL EDITOR

The ISQL Editor includes the ISQL Window and DDL Editor.

The ISQL Window lets you:

- Insert files and open files.

- Rename and configure query tabs.

- Find and replace with regular expressions.

- Mail your script files.

  TIP: To enlarge or reduce (zoom) the display font size, press Ctrl-= or Ctrl-- (plus or minus on the numeric pad), or hold the Ctrl key while scrolling the mouse wheel. To return to the default size, press Ctrl-/ (on the numeric pad).

  TIP: To toggle to the next SQL window, press CTRL +T.

  TIP: The row limit option lets you display only the first 'n' rows from any submitted query.

  TIP: For Oracle, Rapid SQL displays REF CURSOR contents in the ISQL Window and Results Tab.

  TIP: For IBM DB2 LUW for Linux, Unix, and Windows and Oracle, you can access the Code Completion functionality with the CTRL+K shortcut.

Related Topics

- Toolbar Options

- Opening ISQL Windows

- Opening DDL Editors

- Opening Script Files

- Inserting Files into an ISQL Window

- Splitter Windows

- Find and Replace in an ISQL Window

- Regular Expressions

- Navigating in an ISQL Window

- Scheduling

- Sending SQL Scripts

- Renaming and Closing Query Window Tabs

- Printing a Script

- Saving and Closing Scripts

- SQL Preprocessor

- Automated Error Detection and Coding Aid Features
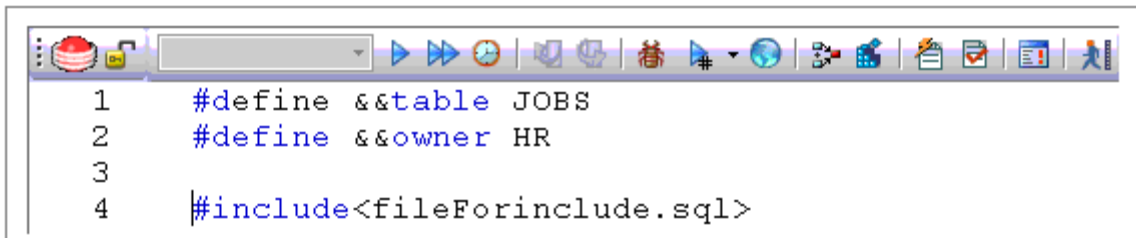
PREPROCESSING #DEFINE AND #INCLUDE DIRECTIVES

Rapid SQL provides SQL preprocessing similar to functionality provided by C language compiler directives. The ISQL Editor supports a simplified version of the following directives.

- **#include** provides a means to include the contents of a file in a script at the location of the directive

- **#define** provides a simple, global search and replace function within a script

The following figure illustrates the result of basic preprocessing of a script, nesting of **#define** directives/references, and the notations required by the ISQL editor. The original script includes two **#define** directives and a **#include** reference to a one-line file named **fileForInclude.sql**. The referenced file includes two identifiers to be replaced with **#define** processing.

Script content before SQL preprocessing
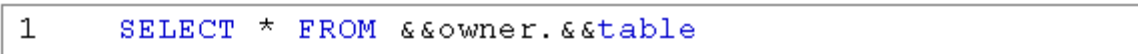
```
1       #define &&table JOBS
2       #define &&owner HR
3
4      |#include<fileForinclude.sql>
```

Contents of fileForinclude.sql
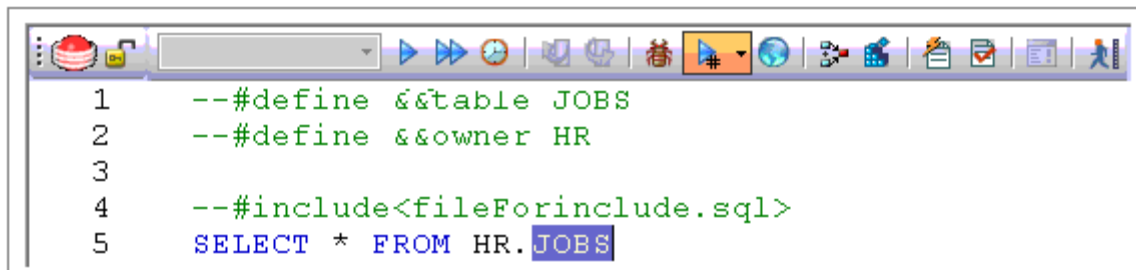
```
1       SELECT * FROM &&owner.&&table
```

Script content produced by SQL preprocessing

```
1       --#define &&table JOBS
2       --#define &&owner HR
3
4       --#include<fileForinclude.sql>
5       SELECT * FROM HR.JOBS
```

The key steps in working with SQL preprocessing are:

- **Preparing the ISQL editor for #include and #define preprocessing** - Prior to using SQL preprocessing features, you should set the paths that will be searched in processing **#include** directives. For details, see <u>Setting Up Rapid SQL to Preprocess #include Directives</u>.

- **Using #define and #include directives in scripts** - While the supported directives approximate typical C language #define and #include functionality, there are differences in functionality and required syntax. For example, identifiers in the **#define** directive and in all instances to be replaced, must be prefixed with two ampersand characters (`#define &&PI 3.14159`). For detailed information, see <u>#include Functionality and Syntax</u> and <u>#define Functionality and Syntax</u>.

- **Preprocessing and executing scripts containing #define and #include directives** - The ISQL Editor offers two preprocessing options. You can have a script preprocessed without being executed, opening the processed script in a new editor window with all **#define** and **#include** substitutions made. This lets you view the processed script before execution or continue working with the processed SQL. Alternatively, you can have the script preprocessed and executed in a single step. For details, see Preprocessing and Executing Scripts Containing #define and #include Directives.

SETTING UP RAPID SQL TO PREPROCESS #INCLUDE DIRECTIVES

In ISQL editor processing of a #include directive, Rapid SQL searches the following locations, in the following order, for the specified file:

1   The location specified on the **Datasource Properties** tab of the Datasource Registration Wizard/Editor. For details, see Registering or editing datasources.

2   The location specified on the Directories tab of the Options editor. For details, see Directories Options.

> **NOTE:**   For detailed information on setting Rapid SQL options, see Specifying Rapid SQL application and feature options.

Before using SQL preprocessing, ensure that server side or server side and local search paths for include files are specified.

#INCLUDE FUNCTIONALITY AND SYNTAX

Rapid SQL support for the **#include** directive provides a means to include the contents of a file in a script at the location of the directive. For example, if a script contains the following:

```
#include mydeclarations.sql
```

then on preprocessing of the script, there are two effects:

- The line containing the **#include** directive is commented out before the script is sent to the database

- The text in the file mydeclarations.sql is placed in the script following the commented out line with the **#include** directive.

The **#include** directive is supported for simple file names only. Supported syntax of the **#include** directive for use in the ISQL editor, Procedure Object Editor, or Package Body Object Editor, is as follows:

#include <_filename.ext_**>**
where:

- *filename.ext* is a simple filename and extension

> **NOTE:** For those familiar with C compiler functionality, angle bracket and quoted forms are supported only indirectly. While **#include <***filename***.***ext***>** and **#include "***filename***.***ext***"** forms are valid, they are functionally equivalent to the **#include** *filename***.***ext*. Using the angle bracket or quoted forms has no effect on locations searched for the target file.

Searches are performed in the locations specified in the setup for this feature. For details, see [Setting Up Rapid SQL to Preprocess #include Directives](#).

Error processing is as follows:

- If the preprocessor fails to include the specified file, it displays an error message noting the reason for the failure (such as the file does not exist, insufficient permissions on the file, or file too large). Preprocessing or execution of the script cannot continue until the error is corrected.

- If the file is found in the first search location specified in [Setting Up Rapid SQL to Preprocess #include Directives](#) but cannot be opened (permission denied for example), no attempt will be made to locate the file in the second specified search location.

#DEFINE FUNCTIONALITY AND SYNTAX

The **#define** directive provides a simple, global search and replace function within a script. For example, if a script contains the following:

```
#define &&PI 3.14159
```

then on execution of the script, there are two effects:

- All instances of **&&PI** in the script would be replaced by **3.14159** on execution of the script

- The line containing the **#define** directive is commented out before the script is sent to the database

The **#define** directive is supported for symbolic constants only. Supported syntax of the #define directive for use in the ISQL editor, Procedure Object Editor, or Package Body Object Editor, is as follows:

#define &&*Identifier Replacement_text*
where:

- *Identifier* is any character string appearing in the script

- *Replacement_text* is the string that will replace all instances of the string specified by the *Identifier* argument. Valid values are strings, numbers or combinations consisting of the digits **0**-**9**, characters **a**-**z**, characters **A**-**Z**, and the underscore character.

> **NOTE:** In addition to the actual **#define** directive appearing in a script, the ampersand notation is also required in all references that are to be replaced. References that are nor prefixed with ampersand characters are not processed.

Nested **#define** directives are also supported. For example if a script contains the following:

#define &&myTable Clients

```
#define &&embtClients Embarcadero
#define &&tempTable New&&myTable
#define &&embtTempTable &&embtClients&&myTable

Select * from &&tempTable;
Select * from &&embtTempTable
```

then after preprocessing, the contents of the script would be as follows:

```
Select * from  NewClients;
Select * from EmbarcaderoClients
```

PREPROCESSING AND EXECUTING SCRIPTS CONTAINING #DEFINE AND #INCLUDE DIRECTIVES

The ISQL editor offer two preprocess/execute modes:

- Rapid SQL can preprocess the script, opening the script in a new tab, with all **#define** and **#include** substitutions made. This lets you view your preprocessed script or continue with edits after preprocessing, before executing it in the new tab.

- Rapid SQL can preprocess and execute the script in a single step.

Preprocessing or preprocessing/executing the script consists of selecting a mode and the executing the script. Preprocessing mode is controlled by the preprocessing dropdown on the ISQL editor toolbar.

To preprocess a script and have the preprocessed script opened in a new tab:

1 From the Preprocess dropdown, select **Pre-Process Only**. The Preprocess dropdown icon takes on a distinctive appearance to indicate **Pre-Process Only** mode.

2 On the ISQL toolbar, click the Execute button.

The script with all #define and #include replacements made, opens in a new ISQL editor.

NOTE: For details on error processing and specific handling of directives, see #include Functionality and Syntax and #define Functionality and Syntax.

To preprocess and execute a script in a single step:

1   From the Preprocess dropdown, select **Pre-Process and Execute**.

2   On the ISQL toolbar, click the Execute button.

    The script executes. While the script sent to the server for execution includes all **#define** and **#include** substitutions, the new **Query** tab contains the original, unprocessed script.

For related information, see the following topics:

- See Executing Scripts for other script execution options

- See Preprocessing #define and #include Directives for an introduction to SQL preprocessing in the ISQL editor

TOOLBAR OPTIONS

The table below describes the options of the ISQL Editor toolbar:

Option	Description
Lock	Lets you lock an ISQL Window to a specific datasource connection:
	Locked (manual session termination) - the ISQL window will open a session and keep it open while the window remains open, unless the DBMS terminates the session due to a timeout, or the session is closed. The ISQL Window does not respond to datasource change events in other windows, and only uses the current connection for subsequent execution.
	Unlocked (automatic session terminations) - the ISQL window will execute each statement in a new session with no session state maintained between them.
	When explicitly unlocking an ISQL window, the connection can be returned to the connection pool or disconnected. For details, see the DBMS specific information under ISQL Options.
Execute	Executes the script.
Step Execute	Initiates step execution of the script. For more information, see Step Execution Facility.
Schedule	Opens a dilaog that lets you schedule an SQL job. For more information, see Scheduling.
SQL Begin Transaction	Toggles the Begin Transaction status on and off.
SQL Rollback	Performs a rollback on the current transaction.
SQL Commit	Commits the current transaction.
Debug	Opens the Embarcadero SQL Debugger.
Preprocess	Lets you set a preprocessing state for execution of the script.
	If you select **Pre-process Only**, subsequent use of the Execute button will result in the script being preprocessed without beng executed. The script opens n a new editor window with all #define and #include substitutions made.
	If you select **Pre-process and Execute**, all #include and #define directives are carried out and the script is executed. No new editor window opens.
	For more information, see Preprocessing #define and #include Directives.

Option	Description
Script Execution Facility	Provides a shotcut to opening the Script Execution Facility. For more information, see [Executing Scripts Against Multiple Datasources](#).
Query Plan	Activates and deactivates Query Plan mode. For more information, see [Using the Query Plan Facility](#).
Query Options	Provides a shortcut to opening the Query Options dialog. For more information, see [Query Options](#).
Format	Transforms spaghetti-style written SQL code into an easier read format.
Syntax Check	Initiates a syntax check. This is only necessary if SQL parsing is disabled and autmatic syntax checking is therefore disabled. For more information, see [Syntax Checking](#).
Errors	Closes the Errors tab.

ISQL WINDOWS

A script is a collection of SQL statements used to perform a task. Scripts, stored as files, form the foundation of most application projects. Definitions of tables, indexes, stored procedures, as well as reports and batch jobs, are usually stored in script files and shared by groups of developers using version control software. You can manage SQL scripts in SQL windows. Rapid SQL lets you open multiple SQL windows in one or more workspaces.

Opening ISQL Windows

To open the ISQL Window, do the following:

1   On the *File* menu, click New ISQL.

OR

On the Main tool bar, click New ISQL.

Rapid SQL opens an SQL window in your current workspace.

For more information, see

[ISQL Window Status Bar](#)

[ISQL Editor](#).

ISQL WINDOW STATUS BAR

The ISQL window Status bar lets you view:

- Auto commit status (Oracle) - Automatically commits SQL statements as soon as the statements are run.

- Begin Transaction ON/OFF (SQL Server and Sybase)

> **TIP:** For Microsoft SQL Server and Sybase, to set Begin Transaction status to "Yes", on the ISQL Window toolbar, click the SQL Begin Tran button.

> **TIP:** For Oracle, you can apply auto commit status changes to all open ISQL windows. You can modify the Oracle Auto Commit status and ISQL tab of the Options Editor. For details, see ISQL Options.

AUTOMATED ERROR DETECTION AND CODING AID FEATURES

Rapid SQL analyzes code as you add content to an ISQL editor session and offers the following automated features:
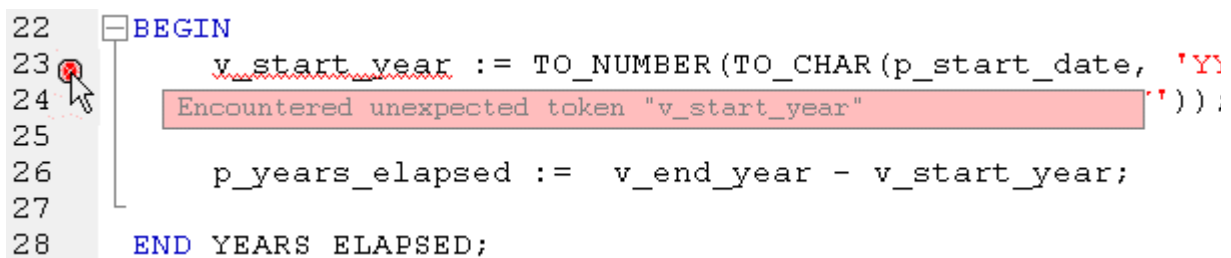
- Syntax Checking automatically flags syntactical errors in your scripts

- Semantic Validation detects object name references to objects not present in the datasource index.

- Code Complete lets you insert or replace object names, selected from suggestion lists, as you edit a script.

SYNTAX CHECKING

The ISQL editor can perform on-the-fly or manually-initiated syntax checking. Rapid SQL checks the syntax of the current contents of an ISQL Editor window against the SQL dialect native to the datasource to which the ISQL editor session is connected. Syntax checking can be performed regardless of whether an ISQL editor session is locked to a datasource.

For each syntax error detected, Rapid SQL annotates your script and offers assistance as follows:

- Line numbers for lines containing syntax errors are flagged with an error icon.

- Hovering the mouse over an error icon displays a tooltip with an error message

- The specific error in the line of code is underlined in red


```
22    BEGIN
23          v_start_year := TO_NUMBER(TO_CHAR(p_start_date, 'Y'
24    Encountered unexpected token "v_start_year"          ')).
25
26          p_years_elapsed :=  v_end_year - v_start_year;
27
28    END YEARS_ELAPSED;
```

> **NOTE:** A line-by-line listing of individual syntax errors is also available in the ISQL editor's Error pane. For more information, see About the Error pane.

Rapid SQL can be configured to perform syntax checks automatically or to require a syntax check to be initiated manually:

- **Automatic** - Automatic syntax checking is enabled from the Options Editor. For details, see ISQL Options - Code Assist Tab.

  If the **Enable Real-time syntax checking** check box is selected, a syntax check is performed whenever there is an interval of 1.5 seconds or more between key strokes. Syntax error annotations persist until the error is corrected and the next automatic syntax check is executed.

- **Manual** - If automatic syntax checking is disabled, a syntax check is only performed when you click the Syntax button on the ISQL editor toolbar:

  When automatic syntax checking is disabled, syntax error annotations persist until the error is corrected and you explicitly run another syntax check.

SEMANTIC VALIDATION

Semantic validation is an on-the-fly ISQL editor feature that verifies that object names are correctly specified. It ensures that the names of supported object types (columns, tables, synonyms, and views) present in a script match those for the connected datasource. Minimizing errors associated with typographical errors or references to obsolete object definitions, Rapid SQL analyzes the names of supported object types as you type, and raises an error condition when it detects a name not present on the datasource.
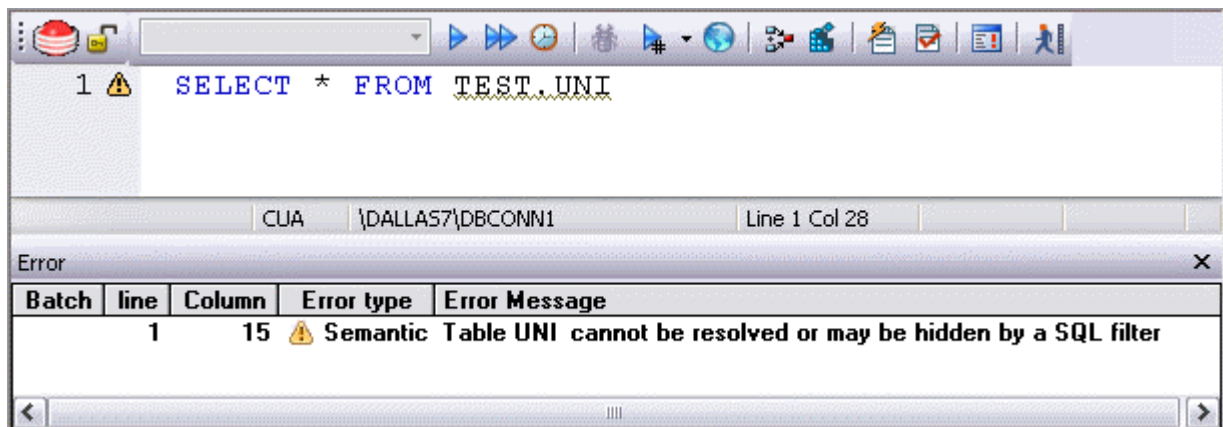
A semantic error can indicate one of the following conditions:

- The object name as specified in the script is not present on the database

- The referenced object is hidden by a SQL filter

- The schema information that Rapid SQL obtains to implement this feature is outdated. This can happen if other users or applications modify the schema during your ISQL editor session. For information on refreshing this information, see ISQL Options - Code Assist Tab.

When making use of this feature, always consider the following points

- For SQL Server and Sybase datasources, statements that contain temporary table names (that is, tables whose name begins with the # character) are ignored by the semantic validation feature.

- Semantic errors in a statement are not displayed for a statement that currently has outstanding syntax errors. For more information, see Syntax Checking.

- For SQL Server and Sybase datasources, system table names (that is tables whose name begins with the **#** character) are ignored by the semantic validation feature.

- While semantic errors can be flagged with an ERROR severity level, they can be more easily distinguished from syntax errors by setting a severity level of WARNING. Similarly, if you do not want semantic errors flagged, you can set the severity level to IGNORE. For details, see ISQL Options - Code Assist Tab.

- Ensure that the ISQL Editor window is connected to a datasource. Rapid SQL requires a datasource name in order to resolve an object name.

  In the case of SQL Server and Sybase, you must also select a database. The exception to this requirement is scripts that have a USE statement, specifying a database, appearing before any object name references.

  NOTE: A line-by-line listing of individual semantic errors is also available in the ISQL editor's Error pane. For more information, see About the Error pane.

Code Complete

Code Complete lets you quickly and accurately write DML and call\execute statements by providing a fast and intuitive lookup for database objects. Code Complete lets you select from a suggestion box that lists objects appropriate at the cursor location within a statement. Code Complete offers intelligent suggestions in providing simple object names or in constructing fully-qualified names.
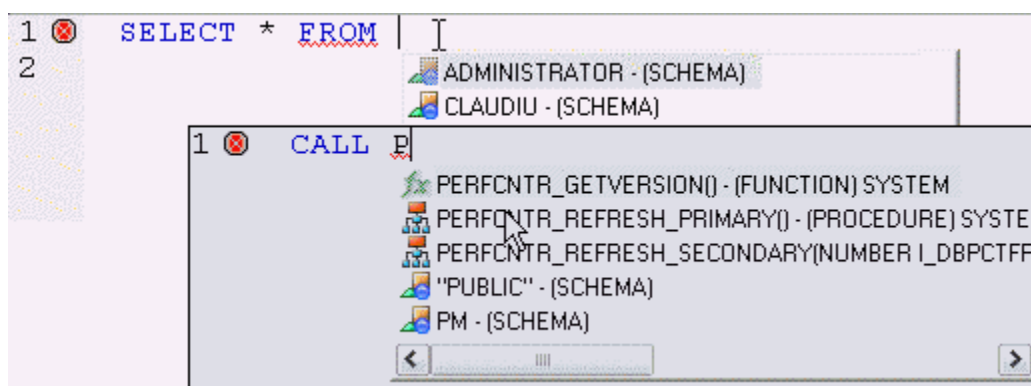
This feature becomes available at all points in INSERT, UPDATE, DELETE, SELECT, CALL, and EXEC statements, including all legal clauses, where a reference to the name of one of the following object types is valid:

- Columns

- Tables

- Views

- Functions

- Procedures

- Packages

- Synonyms

When the Code Complete feature is enabled, it can be invoked in two ways:

- Automatically, if auto-activation is enabled. A Code Complete suggestion box is offered when you start typing an object name in a relevant location and then stop typing for an interval that exceeds the specified auto-activation delay.

- Manually, if auto-activation is disabled or if the specified auto-activation delay is sufficiently large, by pressing CONTROL+SPACE. or typing a period (.).

    NOTE: By default, Code Complete is enabled and configured for automatic invocation. For information on enabling or disabling this feature and configuring auto-activation and other Code Complete preferences, see <u>ISQL Options - Code Assist Tab</u>.

Code Complete offers context sensitive suggestions based on the cursor location within the statement when Code Complete is invoked. For example:

- **When invoked without any preceding text** - Code Complete will supply basic suggestions appropriate to that clause.

    For example, suggestions for the table name in a FROM clause would help a user construct a qualified table or view name. Against a SQL Server datasource for example, this would include schemas and databases.
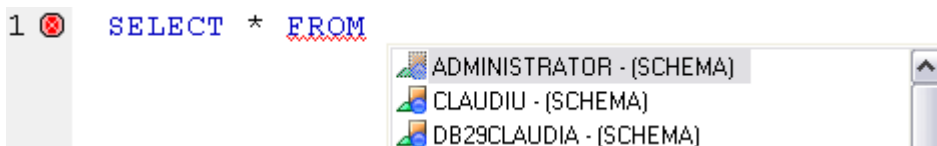
    Suggestions for the column name in a WHERE clause in the same statement would contain the schemas, databases and tables\views referenced in the FROM clause as well as a listing of the applicable columns based on the referenced tables\views.
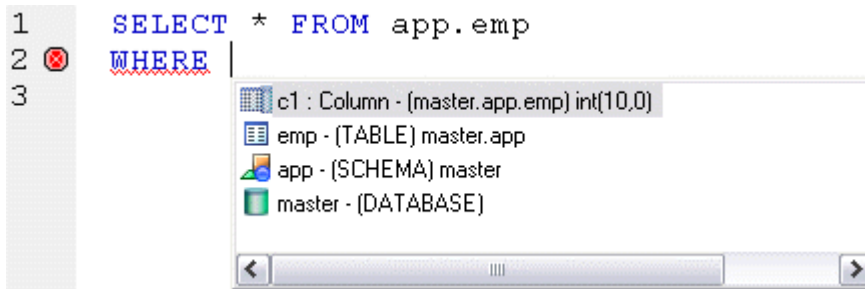
- **After the delimiter within an object name** - Code Complete offers a list that lets you select the next element to build a fully qualified object name.
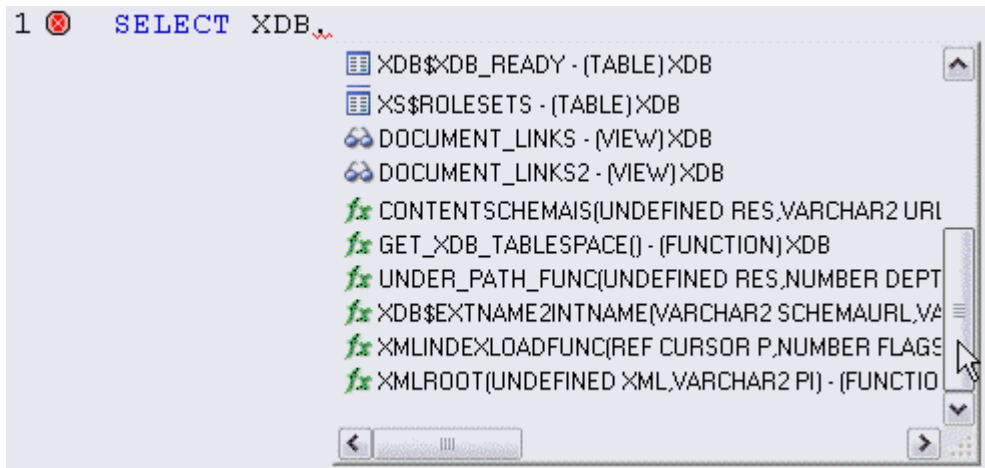
- **After typing one or more characters of an object name** - Code Complete offers a listing of the names of all object name elements, relevant to the cursor location within the object name, that start with the typed characters.
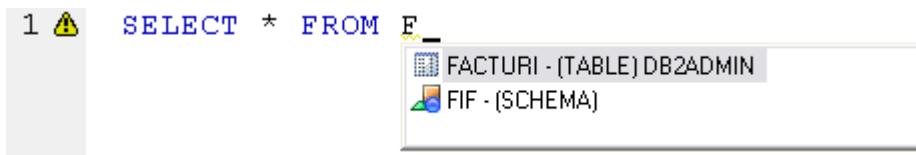
- **At any point within a partially-specified or complete object name** - Code Complete offers all selections appropriate to the location of the cursor.

```
1       SELECT * FROM ADMIN.ADDRESS
```


```
1       SELECT * FROM ADMIN.ADDRESS
```


```
1 ⚠    SELECT * FROM dbo
2
3
```
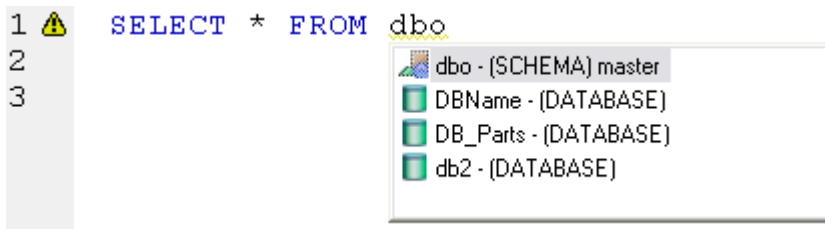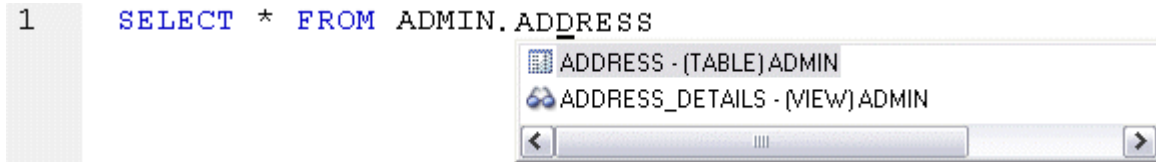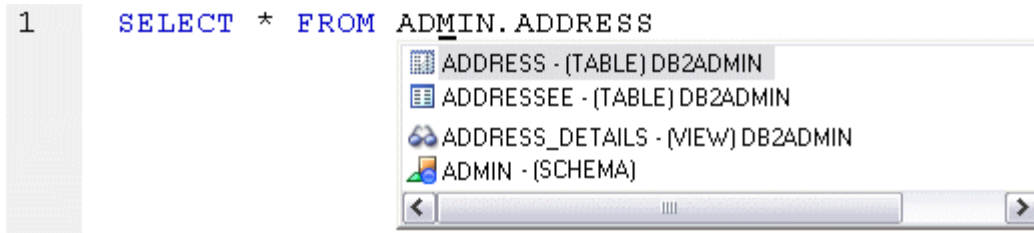

In addition, you can type additional characters while the suggestion list is available to further filter the items appearing in the list.

To insert or replace a suggestion:

1   Double-click an object name or object name element in the list. Alternatively, you can use the arrow keys and press ENTER.

To dismiss the suggestion list without making a selection:

1   Click anywhere outside the suggestion list or press ESC.

> **NOTE:**   The schema information that Rapid SQL obtains to implement this feature can be outdated by the actions of other users or applications during an ISQL editor session. For information on refreshing this information, see ISQL Options - Code Assist Tab.

USING CODE TEMPLATES

Code templates are complete code blocks that can be easily added to open windows or scripts with a few keystrokes. Templates let you define standard or common code units such as loops, comment blocks, query structures, and so on, add them to your code and edit them accordingly.

When you activate Code Assist by typing CTRL+SPACE, the Code Assist menu opens, offering any defined code templates for the currently connected DBMS platform.
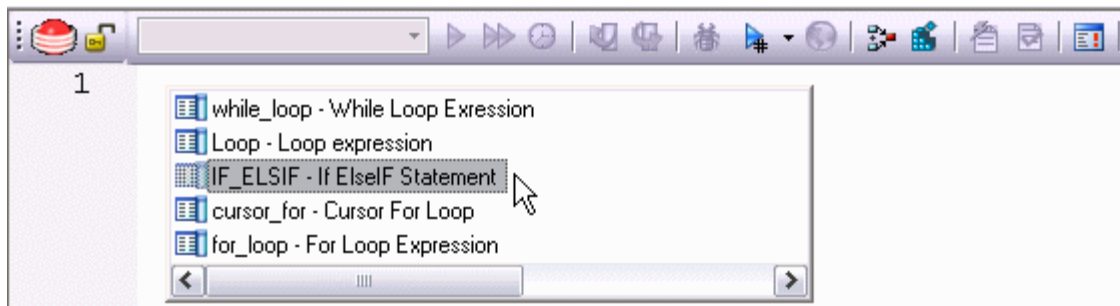
Selecting the short name for the code template and pressing RETURN adds the template at the cursor position.
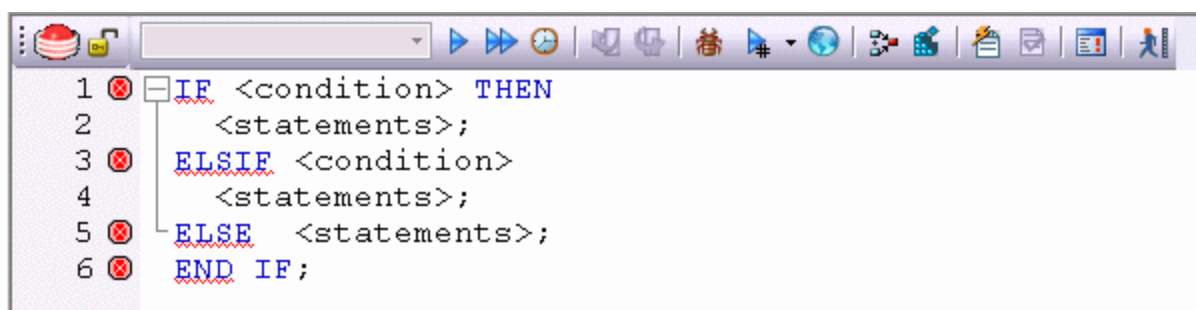
Rapid SQL loads a default set of code templates at startup but you can also add and delete templates from the currently available set. In addition, you can save sets of templates to file and subsequently load specific template sets, allowing you to customize your templates to different platforms or development projects.

Before making use of this feature, you should be familiar with the following tasks:

- Enabling the code templates feature. For details, see Enabling the Code Templates and Auto Replace Features.

- Maintaining the list of available templates. For details, see Maintaining Code Template Definitions and Loading and Saving Custom Code Workbench Settings

USING AUTO REPLACE EXPRESSIONS

Rapid SQL lets you define shortcuts consisting of a few characters that represent longer character strings. Instances of these Auto Replace expressions are automatically replaced by the replacement string on activation events such as typing SPACE, TAB, or RETURN. This feature is useful for creating shortcuts for one-line commands or SQL statement subsets, or even to detect and fix common typographical errors such as **teh** for **the**.

For example, consider an Auto Replace definition with an expression of **sel** to represent **Select * From**:

If the associated activation event includes a SPACE, then on typing **sel** followed by typing SPACE, the following replacement occurs.
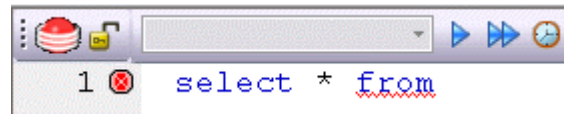
Rapid SQL loads a default set of Auto Replace definitions at startup but you can also add, edit, and delete Auto Replace definitions. In addition, you can save sets of definitions to file and subsequently load specific Auto Replace definitions, allowing you to customize your templates to different platforms or development projects.

Before making use of this feature, you should be familiar with the following tasks:

- Enabling the Auto Replace feature. For details, see Enabling the Code Templates and Auto Replace Features.

- Creating and maintaining the list of available Auto Replace definitions. For details, see Maintaining Auto Replace Expressions and Loading and Saving Custom Code Workbench Settings.

DDL EDITORS

> **NOTE:** This functionality is available for all platforms.

Rapid SQL lets you open a new DDL Editor when you want to create a script that is tied to an object type in your database. The DDL Editor opens containing a template script for the selected object type. Because the DDL Editor is directly tied to a database, database warning messages can be issued. For example, if you have a create table script which includes a DROP TABLE statement, the DDL Editor warns you about the existence of this statement and that you could lose existing table data.

Important Notes

None

The table below describes the options and functionality on the Create New Database Object dialog box.

Option	Description
Object Type	Lets you select the object type to which you want to attach the script.
Owner	Lets you type the name of the object owner for the object. The name of the owner connected to the current datasource is used as the default.
Object Name	Lets you type the name of the object type.

Rapid SQL opens a DDL Editor containing a template for the object type you selected.

For more information, see Opening DDL Editors.

OPENING DDL EDITORS

To open DDL Editors, do the following.

1   On the File menu, click *New* DDL Editor.

    OR

    On the Main tool bar, click the Down arrow on New, and then click DDL Editor.

2   In Object Type select the object type to which you want to attach the script.

3   In Owner type the name of the object owner for the object. The name of the owner connected to the current datasource is used as the default.

4   In Object Name type the name of the object type.

5   Click OK.

    Rapid SQL pastes the DDL into the ISQL Editor.

For more information, see ISQL Editor.

OPEN FILES

> **NOTE:**   This functionality is available for all platforms.

The Open Files dialog box lets you open existing files.

Important Notes

None

The table below describes the options and functionality on the Open Files dialog box.

Option	Description
File Name	Lets you type a script file name. You can open multiple script files simultaneously by selecting multiple file names in the file list.
Files of Type	Lets you control the files displayed in the file list by changing the selection. Rapid SQL displays multiple default file extensions.
Open Button	Click to open one or more files into the current workspace.

Completing the Open Files Dialog Box

To complete the Open Files dialog box, do the following:

1   On the File menu, click Open.

    OR

    On the Main tool bar, click Open.

2   Select a script file.

    OR

3   In File Name, type a script file name.

4   In Files of type, select types of files to display.

5   Click Open to open one or more files into the current workspace.

    Rapid SQL pastes the script into the ISQL Editor.

For more information, see ISQL Editor.

WHAT TYPE OF FILE

> **NOTE:**   This functionality is available for all platforms.

The What Type of File Is dialog box lets you select options for unknown file types.

Important Notes

None

The table below describes the options and functionality on the What Type of File Is dialog box:

Option	Description
The file is a general purpose SQL script	Select if the unknown file is a SQL script.
The file contains the DDL to create a database object of the file contains Oracle Anonymous PL/SQL.	Lets you select object type, type the owner, and object name.
Always open unknown files into a SQL window without prompting.	Select to hide What Type of File Is Dialog Box for future unknown file types.

For more information, see Completing the What Type of File Is Dialog Box.

COMPLETING THE WHAT TYPE OF FILE IS DIALOG BOX

To complete the What Type of File Is dialog box, do the following:

1   On the *File* Menu, click *Open*.

    Rapid SQL opens the Open File(s) dialog box.

2   In the *Open File(s)* dialog box, select the target script, and then click *Open*.

    Rapid SQL opens the What type of file dialog box.

3   Select options, and then click OK.

    Rapid SQL opens the target script in an SQL Editor.

For more information, see ISQL Editor.

INSERT FILE INTO CURRENT FILE

> **NOTE:**   This functionality is available for all platforms.

The ISQL Editor facilitates the reuse of SQL scripts by letting you insert an existing file into another script.

The table below describes the options and functionality on the Insert File into Current File dialog box.

Option	Description
File Name	Lets you type a script file name. You can open multiple script files simultaneously by selecting multiple file names in the file list.

Option	Description
Files of Type	Lets you control the files displayed in the file list by changing the selection. Rapid SQL displays multiple default file extensions.
Open Button	Click to insert one or more files into the current workspace.

Important Notes

None

For more information, see [Completing the Insert File into Current File Dialog Box](#).

COMPLETING THE INSERT FILE INTO CURRENT FILE DIALOG BOX

To complete the Insert File into Current File dialog box, do the following:

1   On the Edit menu, click Edit Insert File.

    OR

    On the *Edit* tool bar, click Edit Insert File.

    Rapid SQL opens the Insert File into Current File dialog box.

2   Select a script file.

    OR

3   In File Name, type a script file name.

4   In Files of type, select types of files to display.

5   Click Open to insert one or more files into the current workspace.

    Rapid SQL inserts the file.

For more information, see [ISQL Editor](#).

SPLITTING WINDOWS

You can split a SQL window into four different screens so that you can view different sections of a document simultaneously. You have the option to split the ISQL Window horizontally, vertically or into quadrants.

Splitting the ISQL Window Horizontally

To split the ISQL Window horizontally, do the following:

1   Point to the split box at the top of the vertical scroll bar on the right side of the SQL window.

2   When the pointer changes, drag it to the desired position.

Splitting the ISQL Window Vertically

To split the ISQL Window vertically, do the following:

1   Point to the split box at the top of the horizontal scroll bar on the bottom left side of the SQL window.

2   When the pointer changes, drag it to the desired position.

Removing Splits from an ISQL Window

Double-click the appropriate split bar to return the SQL window to its normal state.

For more information, see ISQL Editor.

FIND

NOTE: This functionality is available for all platforms.

The Find dialog box lets you search text in your SQL scripts.

Important Notes

None

The table below describes the options and functionality on the Find dialog box.

Option	Description
Find What	Lets you type your search string.
Match whole word only	Select to search for only the complete word.
Match Case	Select to make the search case-sensitive.
Regular expression	Select if you are looking for a regular text expression.
Wrap around search	Lets you search from the end of the script and back to the insertion point.
Direction	Lets you specify the direction you want to search. Click the Up or Down option button.
Find Next Button	Click to find the next occurrence of your search string.
Mark All Button	Click to place a small blue dot next to every line number in the script which meets the required search string criteria.

For more information, see Completing the Find Dialog Box.

COMPLETING THE FIND DIALOG BOX

To complete the Find dialog box, do the following

1 On the Edit menu, click Find.

      OR

      On the *Edit* tool bar, click Find.

2 In Find What, type your search string.

3 Select Match whole word only to search for only the complete word.

4 Select Match Case to make the search case-sensitive.

5 Select Regular expression to search for a regular text expression.

6 Select Wrap around search to search from the end of the script and back to the insertion point.

7 In Direction, click Up or Down.

8 Click Find Next to find the next occurrence of your search string.

9 Click Mark All to place a small blue dot next to every line number in the script which meets the required search string criteria.

      The ISQL Editor highlights the object name if it matches the search criteria.

For more information, see <u>Find</u>.

REPLACE

NOTE: This functionality is available for all platforms.

The Replace dialog box lets you search and replace text in your SQL scripts.

The table below describes the options and functionality on the Replace dialog box.

Option	Description
Find What	Lets you type your search string.
Replace With	Lets you type the replacement text.
Match Case	Select to make the search case-sensitive.
Regular expression	Select if you are looking for a <u>regular text expression</u>.
Wrap around search	Lets you search from the end of the script and back to the insertion point.
Direction	Lets you specify the direction you want to search. Click the Up or Down option button.

Option	Description
Find Next Button	Click to find the next occurrence of your search string.
Replace Button	Click replace the current selection.
Replace All Button	Click to automatically find and replace all occurrences of your search string within the current window.

For more information, see [Completing the Replace Dialog Box](#).

COMPLETING THE REPLACE DIALOG BOX

To complete the Replace dialog box, do the following:

1   On the Edit menu, click Replace.

    OR

    On the Edit tool bar, click Replace.

2   In Find What, type your search string.

3   In Replace With, type the replacement text.

4   Select Match whole word only to search for only the complete word.

5   Select Match Case to make the search case-sensitive.

6   Select Regular expression to search for a regular text expression.

7   Select Wrap around search to search from the end of the script and back to the insertion point.

8   In Direction, click Up or Down.

9   Click Find Next to find the next occurrence of your search string.

10 Click Replace to replace the current selection.

11 Click Replace All to automatically find and replace all occurrences of your search string within the current window.

For more information, see [ISQL Editor](#).

REGULAR EXPRESSIONS

Regular Expressions are offered as an optional search criteria in the SQL windows search facility. Regular Expressions serve as powerful notation for describing string matching patterns. Special characters are used to denote certain match criteria on which the ISQL Editor should conduct its search. The table below describes the special characters and their meanings:

Character	Meaning	Example
^	Circumflex - Constrains the search to the start of a line	^Rap -- Match lines beginning with Rap ^Emb -- Match lines beginning with Emb
$	Dollar - A dollar as the last character of the string constrains matches to the end of lines.	if$--Match lines ending with if ^end$-- Match lines consisting of the single word end
.	Period - A period anywhere in the string matches any single character.	T..l -- Matches Tool, Till, Tail etc. H.w -- Matches Huw, How, Haw etc. ^Sin.ers -- Matches lines beginning with Sinders, Sinners etc.
*	Asterisk - An expression followed by an asterisk matches zero or more occurrences of that expression.	to* -- Matches t, to, too etc. 00* -- matches 0, 00, 000, 0000 etc.
+	Plus - An expression followed by a plus sign matches one or more occurrences of that expression.	to+ -- Matches to, too etc. 10+ -- Matches 10, 100, 1000, 10000 etc. /(:d+/) -- Matches (0), (12464), (12) etc.
?	Question mark - An expression followed by a question mark optionally matches that expression.	for? -- Matches f and for 10? -- Matches 1 and 10
()	Brackets - Brackets can be used to group characters together prior to using a * + or?.	Rap(id)? -- Matches Rap and Rapid B(an)*a -- Matches Ba, Bana and Banana
[ ]	Square brackets - A string enclosed in square brackets matches any character in that string, but no others. If the first character of the string is a circumflex, the expression matches any character except the characters in the string. A range of characters can be specified by two characters separated by a -. These should be given in ASCII order (A-Z, a-z, 0-9 etc.).	{[0-9]} -- Matches {0}, {4}, {5} etc. /([0-9]+/) -- Matches (100), (342), (4), (23456) etc. H[uo]w -- Matches Huw and How Gre[^py] -- Matches Green, Great etc. but not Grep, Grey etc. [z-a] -- Matchesnothing ^[A-Z] -- Match lines beginning with an upper-case letter

Character	Meaning	Example
\	Backslash - A backslash quotes any character. This allows a search for a character that is usually a regular expression specifier.	\$ -- Matches a dollar sign $ \+ -- Matches a +

For more information, see ISQL Editor.

GOTO

> **NOTE:** This functionality is available for all platforms.

The Goto dialog box lets you move to a specific line or column in your script.

Important Notes

None

The table below describes the options on the Goto dialog box:

Option	Description
Line Number	Lets you type or select the target line number.

Completing the Goto Dialog Box

To complete the Goto dialog box, do the following:

1   On the Edit menu, click *Goto*.

    Rapid SQL opens the Go To dialog box.

2   In Line Number, type or select the target line number.

3   Click OK.

    Rapid SQL moves the cursor to the target line.

For more information, see ISQL Editor.

SENDING SQL SCRIPTS

If you have MAPI-compliant E-mail software installed on your computer, then you can send SQL scripts to other users.

To send a SQL script, do the following:

1   On the File menu, click Send.

    OR

    On the Main tool bar, click Send.

    Rapid SQL opens your E-mail application.

2   In the *Address* box, type the name of the addressee(s) and any other options.

    NOTE: The ISQL Editor automatically attaches a file containing your SQL script to the e-mail message.

3   Click *Send.*

    Rapid SQL sends the result set to the specified addressee(s).

For more information, see [ISQL Editor](#).

RENAMING AND CLOSING QUERY WINDOW TABS

SQL windows are tabbed windows that can be closed or renamed.

To rename a Query Tab, you need an open SQL window that includes an executed script. For information on how to execute scripts, see:

- [Executing Scripts](#)

- [Script Execution Facility](#)

Renaming a Query Window Tab

To rename a Query Window Tab, do the following:

1   Right-click the Query Tab on the *SQL window*, and then click *Rename.*

    Rapid SQL opens the Rename Tab dialog box.

    NOTE: The Query Tab can be located on the top or bottom of the SQL window. You can set the location of the tab when configuring Datasource options.

2   In the *New Name* box, type the name of the new Query window.

3   Click *OK.*

    Rapid SQL changes the name and closes the Rename Tab dialog box. The new name of the tab displays directly on the Query Window Tab at the top of the window.

Closing a Query Window Tab

To close a Query Window Tab, do the following:

1   At the top of the *ISQL window,* right-click the Query Tab, and then click Close or Close All.

Rapid SQL closes the Query.

For more information, see [ISQL Editor](#).

PRINT

> **NOTE:**   This functionality is available for all platforms.

The Print dialog box lets you can specify a range of panels to print, or print the contents of a script window to a file.

Important Notes

None

The table below describes the options and functionality on the Print dialog box.

Tab	Option	Description
Scope	Printer	Displays default printer.
	Print Range	Lets you select the appropriate print range.
	Number of Copies	Lets you click the Up or Down arrow or type the number of copies you want.
Page Setup	Header	Lets you type header type to display at the top of the page.
	Footer	Lets you type header type to display at the bottom of the page.
	Header/Footer not within Margins	Select to position header and footer outside the margins.
	Margins	Lets you specify margins in either inches or centimeters.
Options	Chromacoding	Lets you select Use Color if you have a color printer. Lets you select Use Font if script includes italics and bold fonts.
	Line Numbering	Lets you specify the interval between numbered lines.
	Other Options	Lets you select other options.
Documents	Document Box	Lets you select documents to print.
	Clear	Click to clear list.
	Invert	Click to switch printing order.

Tab	Option	Description
Configurations	New Configuration Name	Lets you type a new configuration which saves you current settings, and then click Create.
	Delete	Lets you delete an existing configuration.
	Load	Lets you load an existing configuration.
	Update	Lets you update an existing configuration.

The table below describes the buttons on the Print dialog box:

Button	Description
Save Settings	Lets you save settings.
Font	Lets you open the Font dialog box and select fonts.
Setup	Lets you open the Print Setup dialog box and select printer options.
Preview	Lets you open the Print Preview dialog box that lets you preview the document before you print it.
Print	Lets you print the document.

For more information, see Completing the Print Dialog Box.

COMPLETING THE PRINT DIALOG BOX

To complete the Print dialog box, do the following:

1 On the File menu, click Print.

    OR

    On the Main tool bar, click Print.

2 On the tabs, select options.

3 Click Print.

For more information, see ISQL Editor.

SAVING AND CLOSING SCRIPTS

Untitled scripts are named SQL1...SQLn by default, and all scripts are saved with the SQL extension. You can save your files with any valid name.

For more information, see:

Saving a Script

Closing a Script

About the Error pane

ISQL Editor

SAVING A SCRIPT

To complete the Save dialog box, do the following:

1 On the File menu, click Save.

   OR

   On the Main tool bar, click Save.

   Rapid SQL opens the Save As dialog box.

2 If this is a new file, in the File Name box, type the name of the file.

3 If this is an existing file and you want to use save as, on the Main menu, click *Save As*, and in the File Name box, type the name of the file.

4 Click Save.

   Rapid SQL closes the Save As dialog box.

For more information, see Saving and Closing Scripts.

SAVING MODIFIED DOCUMENTS

The table below describes the options and functionality on the Save Modified Documents dialog box:

Option	Description
Save All	Click to save documents in all open ISQL windows.
Save Selected	Click to save selected documents.
Invert	Click to clear selection.
Save None	Click to not save documents and close the application.

Important Notes

None

For more information, see Saving and Closing Scripts.

CLOSING A SCRIPT

To close a script, do the following:

1 On the on the Main menu, click Close.

    OR

    On the *ISQL window* tool bar, click Close.

    OR

    In the upper right corner of the window, double-click the *System* menu icon.

    `Rapid SQL starts closing` the script.

2 If you have not saved your script, Rapid SQL prompts you to save the file. Click *Yes* to save and *No* to close without saving.

For more information, see [Saving and Closing Scripts](#).

ABOUT THE ERROR PANE

You can view errors currently present in an ISQL editor window in a separate pane. The error pane provides a dynamic, line-by-line listing of the following error types:

- **Semantic** - errors in specified object names. Semantic problems persist in the Error pane until they are corrected. For more information, see [Semantic Validation](#).

- **Syntactic** - syntax problems associated with the SQL dialect of the data source to which the ISQL Editor session is connected. Syntax problems persist in the Error pane until they are corrected. For more information, see [Syntax Checking](#).

- **Execution** - errors detected the last time the script was executed in the current ISQL editor window. Execution errors persist in the Error pane until they are refreshed by a subsequent execution.

A single menu command toggles the error pane open and closed.

To open or close the error pane:

1 On the Query menu, select *Show Errors*.

For more information, see [ISQL Editor](#).

EDITING SCRIPTS

The ISQL Window incorporates a powerful editor that lets you edit your SQL scripts. The ISQL Window includes editing features such as:

- Uppercase or lowercase character conversion.

- Commenting and uncommenting blocks of text.

- Selecting text.

- Inserting PL/SQL tags.

- Setting and navigating with bookmarks.

The ISQL Editor provides [Paste SQL Syntax](#) and [Paste SQL Statements](#) utilities from which you can paste syntax for SQL commands and functions directly into an ISQL Window.

Editing a Script

To edit a script, do the following:

1   In the *SQL* window, type your changes or additions.

2   When you are finished with your changes, on the *ISQL* window tool bar, click [Execute](#) to compile the script.

For more information, see [ISQL Editor](#).

PASTE SQL SYNTAX

> **NOTE:**   This functionality is available for all platforms.

The Paste SQL Syntax facility lets you paste SQL syntax without having to refer to documentation to find syntax for SQL commands. You can also paste the SQL directly into an ISQL window.The Paste SQL Syntax facility that includes SQL syntax for:

- Commands

- Functions

- XML

- Other object syntax

Important Notes

None

The table below describes the options and functionality on the SQL Syntax dialog box.

Option	Description
SQL Statements	Lets you select the target command, functions, or optimizer hint.
Syntax	Displays the syntax.
Paste Button	Click to paste the SQL statement into your ISQL Window.

> **NOTE:** You must change the placeholders (e.g., expression) in the statements to reflect the specific objects in your database. For assistance with basic SQL statements, such as Select, Insert, Update, or Delete, use the Paste SQL Statement facility instead.

The table below describes options for each RDBMS platform:

Oracle	Sybase ASE	Microsoft
SQL Commands	SQL Commands	SQL Commands
Number Functions	Aggregate Functions	Aggregate Functions
Character Functions	Datatype Conversion Functions	Datatype Conversion Functions
Date Functions	Date Functions	Date Functions
Conversion Functions	Mathematical Functions	Mathematical Functions
Group Functions	String Functions	String Functions
Other Functions (User, NVL, etc.)	System Functions	System Functions
Optimizer Hints	System Diagnostics	Text/Image Functions

For more information, see [Completing the SQL Syntax Dialog Box](#).

COMPLETING THE SQL SYNTAX DIALOG BOX

To complete the SQL Syntax dialog box, do the following:

1 Place your insertion point in an open *SQL* window.

2 On the Edit menu, click Paste SQL Syntax.

    OR

    On the *ISQL Window* tool bar, click Paste SQL Syntax.

3 In SQL Statements, select the target command, functions, or optimizer hint.

4 To paste the SQL statement into your ISQL Window, click Paste.

For more information, see [Editing Scripts](#).

PASTE SQL STATEMENTS

NOTE: The functionality is available for all platforms.

The Paste SQL Statement facility lets you create Select, Insert, Update, and Delete SQL statements. The Paste SQL Statement window displays all available database object names from which you can choose to construct SQL statements. These statements are dependent on the object type you choose.

Important Notes

None

The table below describes the options and functionality on the Paste SQL dialog box.

Option	Description
Datasource	Lets you select the target datasource.
Database	Lets you select the target database.
Owner	Lets you select the owner.
Object Type	Lets you select the target object type.
Select	Click if you want a SELECT SQL statement.
Update	Click if you want an UPDATE SQL statement.
Insert	Click if you want an INSERT SQL statement.
Delete	Click if you want a DELETE SQL statement.
Object Type Box	Rapid SQL displays a column of objects, given your selections of datasource, owner, and object type. Lets you select the check box next to the target object type, or click All to select all. Paste - Click to paste this object into your Editor window.
Middle Box	Rapid SQL displays attributes appropriate to the type of object you selected. For example, if you select Tables, Rapid SQL fills the second column with the columns of the table you select in the first column. If you select Procedures, Rapid SQL fills the second column with available parameters for the procedure you select in the first column. Lets you click the target object type properties or click All to select all. For example, if you selected a Table, then select one or more columns in the middle box; or, if you selected a Procedure, then select parameters in the middle box. Paste - Click to paste the object properties into your Editor window, under the Object Type box.

Option	Description
Right Box	Displays the SQL statement.
	Paste - Click when you are satisfied with the entire SQL statement. Rapid SQL pastes the SQL statement into your Editor window.

For more information, see [Completing the Paste SQL Dialog Box](#).

COMPLETING THE PASTE SQL DIALOG BOX

To complete the Paste dialog box, do the following

1. Place your insertion point in an open *Editor* window.

2. On the Edit menu, click Paste SQL.

   OR

   On the *Editor* window tool bar, click Paste SQL.

3. In Datasource, select the target datasource.

4. In Database, select the target database.

5. In Owner, select the owner.

6. In Object Type, select the target object type.

7. Click Select if you want a SELECT SQL statement.

8. Click Update if you want an UPDATE SQL statement.

9. Click Insert if you want an INSERT SQL statement.

10. Click Delete if you want a DELETE SQL statement.

11. In the Object Type Box, select the check box next to the target object type, or click All to select all.

12. Click Paste to paste this object into your Editor window.

13. In the Middle Box, click the target object type properties or click All to select all.

14. Click Paste to paste the object properties into your Editor window, under the Object Type box.

    Rapid SQL the SQL statement in the right box.

15. In the Right Box, click Paste when you are satisfied with the entire SQL statement.

    Rapid SQL pastes the SQL statement into your Editor window.

For more information, see [Editing Scripts](#).

SELECTING TEXT

The ISQL Editor lets you select a single word, a single line, or a block of text.

Selecting a Single Word

To select a single word, do the following:

1   In the ISQL *Editor* window, position the pointer in the word and double-click.

Selecting a Line of Text

Rapid SQL offers two ways to select a line of text:

1   In the ISQL *Editor* window, click the line number listed on the left side of the window.

    OR

2   Position the pointer at the beginning of the line of text, hold down the SHIFT key, and then click the end of the line of text.

Selecting a Block of Text

To select a block of text, do the following:

1   In the ISQL *Editor* window, drag until the block of text is selected.

For more information, see [Editing Scripts](#).

MOVING AND COPYING TEXT IN AN ISQL WINDOW

You can move or copy information anywhere in an Editor window or into a different Editor window by dragging it.

Moving Text

To move text, do the following:

1   In the ISQL *Editor* window, select the text you want to move.

2   Drag to the target location.

Copying Text

To copy text, do the following:

1   In the ISQL *Editor* window, select the text you want to copy.

2   On the *Edit* menu, click *Copy*.

3 Drag to the target location.

> **NOTE:** If you are dragging between different Editor windows, arrange the Editor windows so that the source and destination windows are open and visible. You must be able to see both the original and target locations.

For more information, see [Editing Scripts](#).

COMMENTING AND UNCOMMENTING SCRIPTS

For most developers and development teams, documenting code is a fundamental part of the coding process. Besides contributing explanatory material for others who need to analyze or maintain your code later, the ability to comment and uncomment code can be very useful for bypassing statements in a procedure during compilation.

Commenting Code

To commenting out code, do the following:

1 On the Edit menu, click Comment Out.

OR

Select a line of code, and on the Edit tool bar, click Comment Out.

Rapid SQL comments code.

Uncommenting Code

To uncomment code, do the following:

1 On the Edit menu, click *Undo* Comment Out.

OR

Select a line of code, and on the Edit tool bar, click Undo Comment Out.

Rapid SQL uncomments code.

For more information, see [Editing Scripts](#).

CHANGING CASE

When writing SQL scripts, you can change the letter casing of certain statements or lines of code. You can change case from lowercase to uppercase, or from uppercase to lowercase, using the case functions.

Changing Case

To change case, do the following:

1 Select one or more letters in your script.

2 On the Edit menu, click Upper Case.

   OR

   On the *Edit* menu, click *Lower Case.*

   OR

   On the *Edit* tool bar, click Upper Case.

   OR

   On the Edit tool bar, click Lower Case.

   Rapid SQL changes the case.

For more information, see Editing Scripts.

CUTTING, COPYING AND PASTING TEXT AND COLUMNS IN AN ISQL WINDOW

The Editor window incorporates Cut, Copy and Paste text, and whole columns functions. You can move the selected text or columns to and from the Windows clipboard.

For more information, see:

Copying and Pasting Text

Cutting and Pasting Text

Copying and Pasting a Column

Cutting and Pasting a Column

COPYING AND PASTING TEXT

To copy and paste text, do the following:

1 In the *Editor* window, select the target text.

2 On the Edit menu, click Copy.

   OR

   On the *Editor* tool bar, click Copy.

   Rapid SQL copies the text.

3   On the Edit menu, click Paste.

   OR

   On the *Editor* tool bar, click Paste.

   Rapid SQL pastes the text.

For more information, see [Cutting, Copying and Pasting Text and Columns in an ISQL Window](#).

CUTTING AND PASTING TEXT

To cut and paste text, do the following:

1   In the *Editor* window, select the target text.

2   On the Edit menu, click Cut.

   OR

   On the *Editor* tool bar, click Cut.

   Rapid SQL cuts the text.

3   On the Edit menu, click Paste.

   OR

   On the *Editor* tool bar, click Paste.

   Rapid SQL pastes the text.

For more information, see [Cutting, Copying and Pasting Text and Columns in an ISQL Window](#).

COPYING AND PASTING A COLUMN

To copy and paste a column, do the following:

1   In the *Editor* window, position the pointer in front of the column of the target text.

2   Press *ALT* and simultaneously drag the pointer over the target column.

3   On the *Edit* menu, click Copy.

   OR

   On the *Editor* tool bar, click Copy.

   OR

   Right-click the text, and then click *Copy*.

   Rapid SQL copies the column.

4   In the *Editor* window, position the pointer where you want to paste the column.

5   On the *Edit* menu, click Paste.

OR

On the *Editor* tool bar, click Paste.

OR

Right-click the text, and then click *Paste*.

Rapid SQL pastes the column.

For more information, see [Cutting, Copying and Pasting Text and Columns in an ISQL Window](#).

CUTTING AND PASTING A COLUMN

To cut and paste a column, do the following:

1   In the *Editor* window, position the pointer in front of the column of the target text.

2   Press *ALT* and simultaneously drag the pointer over the target.

3   On the *Edit* menu, click Cut.

OR

On the *Editor* tool bar, click Cut.

OR

Right-click the text, and then click *Cut*.

Rapid SQL cuts the column.

4   In the *Editor* window, position the pointer where you want to paste the column.

5   On the *Edit* menu, click Paste.

OR

On the *Editor* tool bar, click Paste.

OR

Right-click the text, and then click *Paste*.

Rapid SQL pastes the column.

For more information, see [Cutting, Copying and Pasting Text and Columns in an ISQL Window](#).

SETTING BOOKMARKS

Bookmarks are useful tools for navigating throughout an Editor window. You can jump back and forth between bookmarks easily, and there is no practical limit to the number of bookmarks you can set.

The table below describes the options for setting bookmarks:

Feature	Description
Bookmarks:	Bookmarks are valuable navigation aids for jumping from one portion of a script to another. You can add bookmarks in important areas of your scripts, then jump back and forth between bookmarks.

Setting a Bookmark

To set a bookmark, do the following:

1   In the Editor window, position the pointer in front of the line you want to bookmark.

2   On the Editor window tool bar, click Bookmark.

    Rapid SQL inserts a blue dot in the gutter next to the line you have book marked.

For more information, see:

Moving Between Bookmarks

Clearing Bookmarks

MOVING BETWEEN BOOKMARKS

You use the Next Bookmark and the Previous Bookmark buttons to move back and forth between bookmarks.

Goto the Next Bookmark

To goto the next bookmark, do the following:

1   In the Editor window, position the pointer where you have set bookmarks, and then click Bookmark.

    Rapid SQL jumps to the next bookmark.

Goto the Previous Bookmark

To goto the previous bookmark, do the following:

1   In the Editor window, position the pointer where you have set bookmarks, and then click Goto Bookmark.

    Rapid SQL jumps to the previous bookmark.

For more information, see Setting Bookmarks.

CLEARING BOOKMARKS

To clear bookmarks, do the following:

1   On the Editor tool bar, click Clear Bookmark.

    Rapid SQL clears all bookmarks.

    NOTE: This does not clear any error indicators (red dots) that might be showing in the gutter.

For more information, see Setting Bookmarks.

EXECUTING SCRIPTS

The ISQL Editor lets you execute all or part of your SQL scripts. Unless you are executing large scripts that have multiple statements within them, or you need to view optimizer statistics, you execute most of your SQL scripts without options.

NOTE: For Oracle Client 8.0.5, if you execute a multi-line SQL statement with an error in the syntax, the cursor does not jump to the line of the error.

The ISQL Editor incorporates multiple features to refine and ease script execution. The table below describes these features:

Feature	Description
Script Execution Facility	The Script Execution Facility lets you execute scripts in parallel against multiple datasources. The facility also lets you schedule a job to perform the script execution at the appointed time, which saves development time and ensures accuracy and integrity across your databases.
Step Execution Facility	The Step Execution Facility processes batches from one delimiter to the next.
Query Plan	The Query Plan Facility provides a graphical display that lets you identify the execution path that your SQL follows. Rapid SQL's Query Plan window now displays data for the estimated costs, number of rows, and bytes returned by each plan step.
Query Options Dialog Box	MICROSOFT SQL SERVER and SYBASE ASE ONLY: The Query Options dialog box lets you customize what you see when you execute your query.

Executing a Script

To execute a script, do the following:

1   On the Editor window tool bar, click Execute.

    OR

    In the Editor window, right-click, and then click Execute.

    Rapid SQL starts executing the script.

Executing Partial Scripts

To execute a partial script, select the portion of the script to be executed and follow the steps outlined above. This feature is helpful while debugging large scripts and stored procedures.

SCRIPT EXECUTION FACILITY

Rapid SQL has a Script Execution Facility that lets you run parallel queries against multiple datasources. This facility is also a stand-alone utility that is available from the utilities tool bar. If used as a stand-alone utility, you can directly type your script or copy and paste a script in an additional tab. Once you have selected the desired options, Rapid SQL establishes multiple threads and database connections to run the script simultaneously against the selected target datasources. Additionally, ANSI SQL scripts can run against multiple datasources from different DBMS vendors.

The Script Execution Facility runs with two output modes:

• Graphical

• File

Graphical output mode opens multiple result windows, one for each target datasource.

File output mode sends all output to a specified file or files. This feature allows for the execution of scripts against a large number of target datasources in a single operation and enables scheduling capabilities.

Once the scripts have finished executing, you have the option to send a notification message indicating that the script execution has completed via e-mail or Net Send. This message includes attachments of all the output files generated.

For more information, see [Executing Scripts Against Multiple Datasources](#).

EXECUTING SCRIPTS AGAINST MULTIPLE DATASOURCES

To execute scripts against multiple datasources, do the following:

1   On the *Utilities* menu, click Script Execution Facility.

    OR

    On the Utilities tool bar, click Script Execution Facility.

    OR

    On the Editor window tool bar, click Script Execution Facility.

The table below describes the options and functionality on the Script Execution Facility dialog box.

> **NOTE:**   This option is not available for a DDL Editor or PL/ISQL Editor. To execute a script from a DDL or a PL/ISQL Editor, use the stand-alone Script Execution Facility.

2   If you selected File output:

Option	Description
Script	Lets you type, copy and paste, or cut and paste a script.
Target	Only Connected Datasources - Select to see only your currently connected datasources.
	All DBMS Types - Select to see all DBMS types.
	In the Datasource grid, select the check boxes next to the datasources against which you want to run your query, or click Select All to select all datasources.
	Database - Lets you type the name of the target database.
Output	Specify how you want the output to be handled
	Graphical Output - Select for graphical output.
	File Output - Select for file output.
	In the Directory box, type the full path and directory name in which you want to place the output file, or click Browse File icon to locate the directory.
	Click the File Type list to specify the file type you want to apply to the output file.
	Include column titles when saving - Select to include column titles in your saved file.
	Open files with registered applications - Select to open the files with registered application.
Notify	Job Description - Lets you type a job description to supply a subject in your e-mail message.
	E-mail addresses - Lets you type the e-mail address(es) separated by semi-colons.
	Net Send User Names - Lets you type the Net Send user name(s) separated by semi-colons.

Option	Description
Schedule Button	This option is available when you select the File Output Option button and have the Microsoft Task Scheduler installed. For more information regarding scheduling, see Scheduling.

For more information, see Script Execution Facility.

COMPILATION ERROR TAB

Rapid SQL displays any errors encountered when a package, function or procedure was last compiled by a user in a separate tab of those object editors. The Errors Tab provides the line number, column position and error message for each compilation error.

For more information, see Executing Scripts.

COMMAND HISTORY

The ISQL Editor is equipped with a command history box, which lets you keep a history of previously used SQL commands. The Command History is a list that is available in the Results window. Command history lets you go back and run previously used commands from the list rather than opening or inserting a script.

Executing from Command History

To execute from Command History, do the following:

1   In the Results window, click the Command History list, and then click the command you want to execute.

    The Query window is automatically populated with the selected command.

2   To execute the command, click Execute.

For more information, see Executing Scripts.

CANCELING A QUERY

The ISQL Editor lets you cancel a query while the rows are still being returned.

Canceling a Query

To cancel a query, do the following:

1   On the Editor window tool bar, click Cancel.

    NOTE: This button is only enabled after a script has begun executing.

For more information, see Executing Scripts.

STEP EXECUTION FACILITY

Step execution of scripts is an invaluable method to debug your scripts. The Step Execution facility lets you step through each batch in your script. While some batches can include many lines of code, some batches can consist of one line. Rapid SQL parses the statements and moves from batch to batch during the step execution process, each step controlled by you clicking the step execution button.

The table below describes how Rapid SQL provides a number of useful tools for working with step execution of your scripts:

Feature	Description	
ISQL Window Gutter	The ISQL Window Gutter is a vertical gray display bar located to the left of the ISQL window. It includes location indicators such as line numbers, error indicators, and bookmarks. The gutter is a quick visual cue to your current position in the script.	
Script Line Numbers	Line numbers are included to let you navigate large scripts. Error messages in the output window indicate the line number where the error occurred.	
Automatic Error Flagging	When using the Step Execution mode, Rapid SQL flags errors with a red dot in the ISQL window gutter. The errors are flagged sequentially as they are stepped into.	
Point and Click Error Navigation	Rapid SQL displays errors in the output window at the bottom of the screen and selects the errors as they occur. You can click each error and Rapid SQL takes you directly to the line where that error occurred.	
Step		Click the Step button to step into the next batch of code.
Step Back		Click the Step Back icon to step back to the most previous batch of code.
Step Over		Click the Step Over icon to jump over a batch to the next batch.
Run to Cursor		Click the Run to Cursor icon to execute all of the code between the beginning of the script to wherever you have inserted the pointer.
Cancel Step Execution		Click the Cancel Step Execution icon to change to regular execution mode.

For more information, see Using the Step Execution Facility.

USING THE STEP EXECUTION FACILITY

Rapid SQL offers three ways to use the step execution facility:

1   Open a script.

2   On the *Query* menu, click *Step-Execute*.

    OR

    On the Editor window tool bar, click Step-Execute.

    OR

    Right-click, and then click, Step-Execute.

    Rapid SQL starts the procedure and displays errors at the bottom of the Editor window.

    NOTE: Rapid SQL indicates each executing line with a yellow arrow in the gutter of the Editor window. The gutter is that gray area between the line numbers on the left and the code window. As you step through your batches, Rapid SQL moves the arrow to indicate your current location.

3   To continue stepping through the script, on the Editor window tool bar, click Step, which displays in the Editor window after you have started the Step Execute procedure.

4   To step back, on the Editor window tool bar, click Step Back, which displays in the Editor window after you have started the Step Execute procedure.

5   To step over a batch and execute the next batch, on the Editor window tool bar, click Step Over, which displays in the Editor window after you have started the Step Execute procedure.

6   To stop *Step Execution* mode, on the Editor window tool bar, click Stop Step Execute, which displays in the Editor window after you have started the Step Execute procedure.

7   To *Run to Cursor*, on the Editor window tool bar, click Run to Cursor, which is available when the pointer is in the Editor window.

For more information, see [Executing Scripts](#).

USING THE QUERY PLAN FACILITY

Each RDBMS platform lets you view the execution path that your SQL follows. For details, see the following topics:

- [Viewing a tree-based Query Plan (IBM DB2 for LUW, SQL Server, Sybase)](#)

- [Viewing a graphical Query Plan (Oracle)](#)

VIEWING A TREE-BASED QUERY PLAN (IBM DB2 FOR LUW, SQL SERVER, SYBASE)

For IBM DB2 for Windows, Unix, and Linux, Microsoft SQL Server, and Sybase ASE, Rapid SQL lets you view a tree-based representation of a query plan. The Query Plan toolbar button is a toggle. Set it to enable the Show Plan mode.

Rapid SQL's Query Plan window displays data for the estimated costs, number of rows, and bytes returned by each plan step.

> **NOTE:** For IBM DB2 for Linux, Unix, and Windows, Rapid SQL includes a tree view of statements and associated costs.

To view a tree-based representation of a query plan

1 Open a script.

2 On the *Query* menu, click Query Plan.

   Rapid SQL starts the Show Plan mode.

3 To generate the Show Plan in a separate result window, click Execute.

For more information, see [Query Options](#).

VIEWING A GRAPHICAL QUERY PLAN (ORACLE)

For Oracle, Rapid SQL lets you view a graphical representation of a query plan. You can toggle the view between the graph-based view and a tree-based view, view details for each step, and work with a number of viewing options.

The Query Plan toolbar button is a toggle. Set it to enable the Show Plan mode.

To view a graphical representation of a query plan

1 Open a script.

2 On the *Query* menu, click Query Plan.

   Rapid SQL starts the Show Plan mode.

3 To generate the Show Plan in a separate result window, click Execute.

Options when working with the Query Plan facility include:

- Hovering the mouse over an execution step node to display detailed cost details for that step

- Right-clicking and choosing **Find Node** or **Find Next Node** to search large plans for nodes whose label contains a specified text string

- Right-clicking and choosing **Zoom In** or **Zoom Out**

- Right-clicking and choosing an **Orientation** sub-menu command to change the orientation of the view

- Right-clicking and choosing **Overview Window** to open a small window showing the entire plan

- Right-clicking and choosing **Save to File** to open a dialog that lets you save the graphical plan as a graphics file

- Clicking the Query Plan button to toggle between the graphical view and a tree-based view

- For multiple plans created rom the same script/ISQL window, using the dropdown at the top of the plan to change the plan displayed

QUERY OPTIONS

Rapid SQL offers a set of options corresponding to DBMS-specific query options. They let you customize the execution environment for an ISQL session with query-handling directives in areas such as performance, logging/reporting, and error handling. The following tables lists the query options available for each DBMS platform:

Minimally, query options are sent to the server on an ISQL session's first execution. Whether Rapid SQL needs to be sent to the server on subsequent executions depends on whether the session is locked or unlocked and whether you have changed the query option settings.

If you take no action, Rapid SQL will use a set of default query option settings to send to the server, one set per DBMS platform. You can, however save a set of query option settings for each DBMS, and have that set of settings used as the default for ISQL sessions against that platform. As well, during any ISQL session, you can modify the query option settings for the current session.

For more information on understanding and working with query options, see the following topics:

- [Setting query options](#)

- [Query option summary by DBMS](#)

- [Detailed query option descriptions](#)

- [When query options are sent to the server](#)

SETTING QUERY OPTIONS

The query options you select apply only to the current ISQL window instance. In addition, non-default query option settings you specify are not saved. You can however, save a set of query option settings to a file and subsequently open that file in other ISQL editor sections to have those settings applied to that session.

To specify query options:

1 With the ISQL Editor open, select Query Options from the **Query** menu.

   Once open, you have the following options:

   • Set each option manually using the associated check box. For detailed descriptions of query options, see <u>Query option summary by DBMS</u>.

   • Use the **Save** button to open a dialog that lets you save the current option query settings to an XML file. The dialog also has a **Load these query options as default** option that lets you use the settings in the saved file as the default query option settings for the current platform. Exercising this option sets the **Load Query Options** feature on the DBMS-specific **ISQL** tab of the Options editor to specify the file you saved. For more information, see <u>ISQL Options</u>.

   • Use the **Load** button to open a dialog that lets you locate and open a previously-saved XML file containing query option settings.

   • Use the **Reset** button to restore the Rapid SQL query option defaults.

2 Click **OK** to set query options for the current ISQL window session.

   NOTE: For information on conditions for which query options are sent to the server, see <u>When query options are sent to the server</u>.

QUERY OPTION SUMMARY BY DBMS
The following lists the specific query options offered for each platform:

IBM DB2 for Linux, Unix, and Windows	<u>Batch Delimiter</u>	<u>Check syntax when executing</u>	<u>Create Explain plan tables if required</u>	<u>Create explain plan tables on the SYSTOOLS schema</u>
	<u>Isolation Level</u>	<u>Max Errors Before Aborting</u>	<u>Row Count</u>	<u>Run Script with batch execution</u>
IBM DB2 for z/OS	<u>Batch Delimiter</u>	<u>Check syntax when executing</u>	<u>Max Errors Before Aborting</u>	<u>Row Count</u>
	<u>Run Script with batch execution</u>			
Interbase/Firebird	<u>Batch Delimiter</u>			
Microsoft SQL Server	<u>Abort On Overflow</u>	<u>Batch Delimiter</u>	<u>Force Plan</u>	<u>Ignore Overflow</u>
	<u>Isolation Level</u>	<u>Max Errors Before Aborting</u>	<u>No Count</u>	<u>No Exec</u>
	<u>Parse Only</u>	<u>Row Count</u>	<u>Run Script with batch execution</u>	<u>SET ... Options</u>
	<u>Show Plan</u>	<u>Statistics I/O</u>	<u>Statistics Time</u>	<u>Text Size</u>

MySQL	Batch Delimiter	Big Tables	Client Character	Insert ID
	Interactive Timeout	Last Insert ID	Max Errors Before Aborting	Query Cache Type
	Row Count	SET Other Variables	SQL Auto IS NULL	SQL Big Selects
	SQL Big Tables	SQL Buffer Result	SQL Log Bin	SQL Log Off
	SQL Log Update	SQL Low Priority Updates	SQL Max Join Size	SQL Quote Show Create
	SQL Safe Updates	SQL Select	SQL Warnings	Transaction Isolation
	Unique Checks			
Oracle	Batch Delimiter	Check syntax when executing	Execution Information	I/O Activity
	Index Activity	LONG Size Bytes	Max Errors Before Aborting	Network Activity
	Parse Activity	Row Count	Run Script with batch execution	Sort Activity
	Table Activity			
Sybase	Abort On Overflow	Abort On Truncation	ANSI NULL	Batch Delimiter
	Chained	Force Plan	Ignore Overflow	Index Selection
	Isolation Level	Join Selection	Last Insert ID	LONG Size Bytes
	Max Errors Before Aborting	Network Activity	No Count	No Exec
	Output to Execution Window	Output to Server Error Log	Parse Activity	Parse Only
	Prefetch	Row Count	Run Script with batch execution	Set Quoted Identifier
	Show Plan	Statistics I/O	Statistics Subduer Cache	Statistics Time

DETAILED QUERY OPTION DESCRIPTIONS

The following table, provides an alphabetical listing of options, option descriptions, and the platform to which the option applies:

Option/Variable	Description	Platform
Abort On Overflow	If set to TRUE, queries will be aborted on encountering a value greater than the specified size.	SQL Server, Sybase
Abort On Truncation	Specifies behavior following a loss of scale by an exact numeric datatype during an implicit datatype conversion. When set to TRUE, a statement that causes the error is aborted but other statements in the transaction or batch continue to be processed. When set to FALSE, Rapid SQL truncates the query results and continues processing.	Sybase
ANSI NULL	When set to TRUE, controls results of logical operations with NULL values.	Sybase

Option/Variable	Description	Platform
Batch Delimiter	The batch separator must be a viewable character and not a space, new line, or tab. The defaults have been selected to ensure compatibility with the features of Rapid SQL and the respective platform, but can be customized. Note: A custom delimiter works only from within an ISQL window and can't be used for extraction operations. Oracle: "/ " (forward slash) DB2: ";" (semicolon) Sybase: "go" SQL Server: "go" MySQL: ";" (semicolon)	DB2 LUW, DB2 z/OS, InterBase/Firebird, MySQL, Oracle, SQL Server, Sybase
Big Tables	When set to TRUE, allows big result sets by saving all temporary sets to file. This can slow queries.	MySQL
Chained	Invokes a begin transaction before the following statements: delete, insert, open, fetch, select, and update. You must still explicitly close the transaction with a commit.	Sybase
Check syntax when executing	TRUE/FALSE	DB2 LUW, DB2 z/OS, Oracle
Client Character	Default character set.	MySQL
Create Explain plan tables if required	If set to TRUE, Explain Plan tables are created, as necessary. If set to FALSE and you don't manually create tables, Explain Plan operations will fail.	DB2 LUW
Create explain plan tables on the SYSTOOLS schema	If set to TRUE, Explain Plan tables are created on the SYSTOOLS schema. If the tables already exist in the user's default schema, Rapid SQL continues to use those tables. Refer to DB2 documentation for a listing of Explain Plan tables that must be deleted in order to use the SYSTOOLS option. If set to FALSE, Explain Plan tables are created under the user's default schema.	DB2 LUW
Execution Information	True/False	Oracle
Force Plan	When set to TRUE, processes a join in the same order as tables appear in the FROM clause of a SELECT statement only.	SQL Server, Sybase
I/O Activity	True/False	Oracle
Ignore Overflow	When set to TRUE, Rapid SQL will ignore any overflow caused by a resulting value that is larger than a column's specified size.	SQL Server, Sybase
Index Activity	True/False	Oracle
Index Selection	Valuable when tuning query performance.	Sybase
Insert ID	Choose a value to be used a following INSERT or ALTER TABLE statement when you supply an AUTO_INCREMENT value.	MySQL
Interactive Timeout	28800 default	MySQL
Isolation Level	Lets you set DB2 Isolation Levels of UNCOMMITED READ, RESET, CURSOR STABILITY, REPEATABLE READ, or READ STABILITY.	DB2 LUW

Option/Variable	Description	Platform
Isolation Level	Read Committed: Microsoft SQL Server default transaction isolation level. Specifies that shared locks are held while data is read to avoid dirty reads. You can change the data before the end of the transaction, resulting in nonrepeatable reads or phantom data. Read Uncommitted: The lowest level of transaction isolation. Transactions are isolated to ensure that physically corrupt data is not read. Applies dirty read, or isolation level 0 locking, which ensures that no shared locks are issued and no exclusive locks are honored. If set, it is possible to read uncommitted or dirty data values in the data can be changed and rows can appear or disappear in the data set before the end of the transaction. Repeatable Read: Places locks on all data used in a query, preventing other users from updating the data. Other users can insert new phantom rows into the data and are included in later reads in the current transaction. Concurrency is lower than Read Committed. Use this option only when necessary. Serializable: The highest level of transaction isolation. Transactions are completely isolated from one another. Places a range lock on the data set, preventing other users from updating or inserting rows into the data set until the transaction is complete. Concurrency is lower than Repeatable Read. Use this option only when necessary.	SQL Server
Isolation Level	0 1: Sybase default isolation level. Prevents dirty reads. 2: Prevents dirty and non-repeatable reads. 3: Prevents dirty and non-repeatable reads and phantoms. This level is equivalent to performing all selects with holdlock.	Sybase
Join Selection	Valuable when tuning query performance.	Sybase
Last Insert ID	Set the value to be stored in the binary log when you use LAST_INSERT_ID() in a statement that updates a table.	MySQL
LONG Size Bytes	8,192 is the default	Oracle
Max Errors Before Aborting	Select the maximum number of errors encountered before Rapid SQL aborts a script. Setting this value to zero disables the feature.	DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase
Network Activity	True/False	Oracle
No Count	Terminates the message indicating the number of rows affected by a Transact-SQL statement from being returned as part of the results.	SQL Server, Sybase
No Exec	When set to TRUE, compiles each query without executing it.	SQL Server, Sybase
Output to Execution Window	TRUE/FALSE	Sybase

Option/Variable	Description	Platform
Output to Server Error Log	TRUE/FALSE	Sybase
Parse Activity	TRUE/FALSE	Oracle
Parse Only	When set to TRUE, checks the syntax of each Transact-SQL statement and returns any error messages without compiling or executing the statement. When TRUE, makes Microsoft SQL Server only parse the statement. When FALSE, makes Microsoft SQL Server compile and execute the statement. Do not use Parse Only in a stored procedure or a trigger.	SQL Server, Sybase
Prefetch	When set to TRUE, enables large I/Os to the data cache. When set to FALSE, disables large I/Os to the data cache.	Sybase
Query Cache Type	The query is cached for ON or DEMAND.	MySQL
Row Count	When set to TRUE, Rapid SQL terminates the query after returning the specified number of rows.	DB2 LUW, DB2 z/OS, MySQL, Oracle, SQL Server, Sybase
Run Script with batch execution	TRUE/FALSE	DB2 LUW, DB2 z/OS, Oracle, SQL Server, Sybase
SET ... Options	The **Send Set Options** setting dictates whether the remaining ANSI SQL Options in this category are sent to the server. The default for this option is set using the **Enable SET query options** setting on the ISQL tab of the Options editor. For details, see ISQL Options. If the **Send Set Options setting** is enabled, the remaining settings in this category let you specify the specific ANSI SQL options that are sent to the server: **ansi_defaults**, **ansi_nulls**, **ansi_null_dflt_on**, **QUOTED IDENTIFIER**, **cursor_close_on_commit**, **ANSI_PADDING**, **ANSI WARNINGS**, and **IMPLICIT_TRANSACTIONS**. The initial default values are hard-coded, not obtained from server settings.	SQL Server
SET Other Variables	Variables to be set at runtime.	MySQL
Set Quoted Identifier	TRUE/FALSE	Sybase
Show Plan	When set to TRUE, reports data retrieval methods chosen by the Microsoft SQL Server query optimizer.	SQL Server, Sybase
Sort Activity	TRUE/FALSE	Oracle
SQL Auto IS NULL	When set to TRUE, enables you to find the last inserted row for a table.	MySQL
SQL Big Selects	When set to TRUE, SELECT statements likely to take a very long time to execute will be aborted (i.e., where the number of rows examined exceeds the max join size)	MySQL
SQL Big Tables	TRUE/FALSE	MySQL
SQL Buffer Result	When set to TRUE, forces results from SELECT statements into temporary tables.	MySQL

Option/Variable	Description	Platform
SQL Log Bin	When set to TRUE, allows logging to the binary log.	MySQL
SQL Log Off	When set to TRUE, no logging is done to the general query log.	MySQL
SQL Log Update	When set to TRUE, allows logging to the binary log.	MySQL
SQL Low Priority Updates	When set to TRUE, gives table modifying operations lower priority than SELECT operations	MySQL
SQL Max Join Size	When set to TRUE, you can catch SELECT statements where keys are not used properly and that would probably take a long time. Set it if your users tend to perform joins that lack a WHERE clause, that take a long time, or that return millions of rows.	MySQL
SQL Quote Show Create	When set to TRUE, table and column names will be quoted.	MySQL
SQL Safe Updates	When set to TRUE, the query aborts UPDATE or DELETE statements that do not use a key in the WHERE clause or a LIMIT clause. This makes it possible to catch UPDATE or DELETE statements where keys are not used properly and that would probably change or delete a large number of rows	MySQL
SQL Select	The maximum number of records that should be returned from SELECT statements.	MySQL
SQL Warnings	Defines whether or not single row insert statements generate an information string in the event of a warning.	MySQL
Statistics I/O	Lets you display information regarding the amount of disk activity generated by Transact-SQL statements.	SQL Server, Sybase
Statistics Subduer Cache	Displays the number of cache hits, misses, and the number of rows in the subquery cache for each subquery.	Sybase
Statistics Time	Displays the number of milliseconds required to parse, compile, and execute each statement.	SQL Server, Sybase
Table Activity	True/False	Oracle
Table Count	Sets the number of tables that Sybase ASE considers at one time while optimizing a join.	Sybase
Text Size	8,192 is the default	SQL Server, Sybase
Transaction Isolation	Repeatable Read is the default. Read Committed, Read Uncommitted, and Serializable are the other options. Refer to MS SQL Query Options dialog box for an explanation.	MySQL
Unique Checks	Performs uniqueness checks for secondary indexes of MyISAM tables.	MySQL

WHEN QUERY OPTIONS ARE SENT TO THE SERVER

Rapid SQL optimizes sending query options to the server for locked connections. Since an unlocked connection results in a new connection for each execution, query options must be sent with each execution. The following table outlines the specific cases:

Session unlocked	When you open an ISQL window, if you not lock the connection, all query options are sent to the server each time you execute a script, immediately before executing the actual script. As long as the connection remains unlocked, all query options are sent to the server at each execution.
	Similarly if you unlock a currently locked session, all query options are sent to the server at each subsequent execution.
	For information on locking sessions, see <u>Toolbar Options</u>.
Session locked	When you open an ISQL and lock the connection, all query options are sent to the server on the first execution. On subsequent executions, if you do not modify query option settings, no query options are sent to the server on subsequent executions.
	If you open **Query Options** dialog and modify options, on the next execution only the modified settings are sent to the server. For more information, see <u>Setting query options</u>.
	These rules apply as long as the connection remains locked.

RESULT OPTIONS

The Result Options dialog box lets you set the SQL Results Window options. The table below describes Results options:

> **NOTE:** The options you select only apply to the current window. To make options permanent, set the options in the Options Editor.

Interface Element	Option	Description	Default
Result Window	Single Window	Displays all results in one tabbed result window. Multiple result sets are appended together in the window. Single Window and Multiple Windows options are mutually exclusive.	Selected
	Multiple Windows	Displays multiple result sets one result set per window. Single Window and Multiple Windows options are mutually exclusive.	Not selected
Placement	Attached to Editor	Used in conjunction with Single Window option or Multiple Window option. Indicates that results appear as tabbed windows attached to the ISQL Window. Attached to Editor and Unattached options are mutually exclusively.	Selected

Interface Element	Option	Description	Default
	Unattached	Used in conjunction with Single Window option or Multiple Windows option. Indicates that results appear in windows separate from the ISQL Window. Attached to Editor and Unattached options are mutually exclusive.	Not Selected
Format	Standard Grid	Displays all result sets in a standard grid format. Result sets are only displayed in grid format in SQL Editors that are opened after you have selected this option. It does not apply to SQL Editors that are already open.	Selected
	HTML	Displays all result sets as HTML tables. Result sets are only displayed in HTML format in SQL Editors that are opened after you have selected this option. It does not apply to SQL Editors that are already open.	Not selected
	ASCII Text	Displays all result sets as ASCII Text. Result sets are only displayed in ASCII Text format in SQL Editors that are opened after you have selected this option. It does not apply to SQL Editors that are already open.	Not selected

USING THE TOKEN MATCHING CAPABILITY

When you are working with a large script with multiple levels of embedded steps, compare left and right parentheses, curly braces, square brackets and BEGIN/END pairs to make sure that you have delimited your code properly.

The Token Matching capability of Rapid SQL helps you achieve clean code.

Finding Matching Tokens

To find matching tokens, do the following:

1   Position the insertion pointer just to the left of the first token in a token pair you want to locate.

2   Click Match Token.

   Rapid SQL jumps the pointer to the next available token.

For more information, see [Query Plan Facility](#).

RESULTS EDITOR

The results of your SQL queries are displayed in the Results Tab of each Editor Window, which captures result sets in a virtual data grid that accommodates large result sets. The data grid offers many of the features of a basic Windows spreadsheet, giving you a great deal of flexibility in editing the worksheet and formatting its contents.

> **TIP:** For Oracle, Rapid SQL displays REF CURSOR contents in the ISQL Window and Results Tab.

You have many choices for navigating and viewing your SQL query results. The Results window includes functionality to set result window options, find and replace, export data to other products such as Microsoft Excel, and mail your results files.

For more information, see [Configuring Result Set Windows](#).

CONFIGURING RESULT SET WINDOWS

Result set windows can be configured in a variety of ways. You can configure your result set windows to present result sets in multiple or single panels, attached or detached from the corresponding ISQL window. These options can be set globally to save you the hassle of setting options for each result window. Additionally, Result windows can be torn off and dropped onto an open area of the workspace to create separate windows. These are known as Tear Off Tab Windows.

You can set the Result Window to display result sets in any of the following modes:

- [Single result sets in one attached tab window.](#)

- [Multiple result sets in one attached tab window.](#)

- [Single result sets in one separate unattached tab windows.](#)

- [Multiple result sets in one separate unattached tab windows.](#)

- Tear Off Tab windows.

VIEWING MULTIPLE RESULT SETS IN SEPARATE UNATTACHED WINDOWS

To set the option to have multiple result sets displayed in multiple unattached windows, do the following:

1   On the *File* menu, click Options.

    OR

    On the Main tool bar, click Options.

    Rapid SQL opens the Options Editor.

2   In the Options Editor, click the list, and then click Results or click the Results Tab.

3   In the Results Window grid, click the Multiple Windows option.

4   In the Results Window grid, click the Unattached option.

5   Click OK.

> Rapid SQL saves the settings and closes the Options Editor.

For more information, see [Configuring Result Set Windows](#).

VIEWING MULTIPLE RESULT SETS IN ONE ATTACHED TAB WINDOW

To set the option to have multiple result sets displayed in one attached tab window, do the following:

1   On the *File* menu, click Options.

> OR

> On the Main tool bar, click Options.

> Rapid SQL opens the Options Editor.

2   In the Options Editor, click the list, and then click Results or click the Results Tab.

3   In the Results Window grid, click the Multiple Windows option.

4   In the Results Window grid, click the Attached to Editor option.

5   Click OK.

> Rapid SQL saves the settings and closes the Options Editor.

For more information, see [Configuring Result Set Windows](#).

VIEWING SINGLE RESULT SETS IN ONE SEPARATE UNATTACHED WINDOW

To set the option to have a single result set displayed in one unattached window, do the following:

1   On the *File* menu, click Options.

> OR

> On the Main tool bar, click Options.

> Rapid SQL opens the Options Editor.

2   In the Options Editor, click the list, and then click Results or click the Results Tab.

3   In the Results Window grid, click the Single Window option.

4   In the Results Window grid, click the Unattached option.

5   Click OK.

Rapid SQL saves the settings and closes the Options Editor.

For more information, see [Configuring Result Set Windows](#).

VIEWING SINGLE RESULT SETS IN ONE ATTACHED TAB WINDOW

To set the option have all SQL query results displayed in a single attached tab window, do the following:

1   On the *File* menu, click Options.

OR

On the Main tool bar, click Options.

Rapid SQL opens the Options Editor.

2   In the Options Editor, click the list, and then click Results or click the Results Tab.

3   In the Results Window grid, click the Multiple Windows option.

4   In the Results Window grid, click the Attached to Editor option.

5   To re-use the same result set window for subsequent result sets, select the Reuse window check box. This option is only valid for single, attached result windows.

6   Click OK.

Rapid SQL saves the settings and closes the Options Editor.

For more information, see [Configuring Result Set Windows](#).

EXPORTING DATA TO OTHER PRODUCTS

You can export data from a result set to traditional spreadsheet products, such as Microsoft Excel. You can copy part or all of a result set and paste it into your traditional spreadsheet product by way of the Microsoft Windows Clipboard function. You can also save your result sets as tab delimited files, comma separated files or as HTML tables. HTML tables can be opened in your default Internet browser. Tab delimited files and CSV files can be opened in any spreadsheet or word processing program.

> **NOTE:**   Rapid SQL supports pipe delimited ('|") files when you save result sets.

For more information, see [Results Editor](#).

SETTING RESULT WINDOWS TO READ ONLY MODE

To set your result windows to read only mode to keep anyone from accidentally editing or altering a result set, do the following:

1   Select a *Result* window that you want to make read only.

2   On the Edit menu, click *Read Only*.

Rapid SQL places a check-mark next to read only and sets the command.

> **NOTE:**   The Read Only command is a toggle. When it is set, the formatting buttons on the Edit menu are not available.

For more information, see [Results Editor](#).

RESULT WINDOW STATUS BAR

The Result Window Status Bar displays information about each Result window at the bottom of each window. You can display or hide the status bar by setting the Status Bar preference. This preference applies to all result windows.

For more information, see [Results Editor](#).

MAILING RESULT SETS

If you have MAPI-compliant electronic mail software installed on your computer, then you can mail result sets to other users.

Mailing a Result Set

To mail a result set, do the following:

1   Open the Message dialog box:

2   On the *File* menu, click Send.

OR

On the File tool bar, click Send.

Rapid SQL opens the open Message dialog box.

3   In the *Address* box, type the name of the addressee(s) and any other options.

The ISQL Editor automatically attaches a file containing your result set to the mail message.

4   Click *Send*.

Rapid SQL sends the result set to the specified addressee(s).

For more information, see [Results Editor](#).

CLOSING AND RENAMING RESULT WINDOW TABS

Rapid SQL lets you close or rename tabbed Result set windows.

Renaming a Result Window Tab

To rename a Result Window Tab, do the following:

1   Place your pointer over the Results Tab.

2   Right-click the Results Tab, and then click *Rename*.

    Rapid SQL opens the Rename Tab Editor.

3   In the *New Name* box, type the new name.

4   Click OK.

    Rapid SQL changes the name and closes the Rename Tab Editor.

Closing a Result Window Tab

To close a Result Window Tab, do the following:

1   On the *Result Window Tab* tool bar, click Close.

    OR

    Right-click the Results Tab, and then click Close.

    `Rapid SQL closes the Result Window Tab.`

For more information, see [Results Editor](#).

SAVING AND CLOSING RESULT SETS

You can save your result sets using the standard Save and Save As functions. You can save multiple copies of the same result set and modify each copy to specific formatting requirements.

Saving Results

To save results, do the following:

1   On the *File* menu, click Save.

    OR

    On the Main tool bar, click Save.

    Rapid SQL opens the Save Results dialog box.

2   In the File name box, type the name of the result set.

3   In Save as type, select the file type.

>   **TIP:**   You can now save data in Excel 2000 or later .xls, user-specified delimited .txt, and
>   XML formats.

4   To include column titles, select Include column titles when saving.

5   For delimited results, in User Specified Delimiter type the delimiter.

6   Click Save.

Rapid SQL saves the file and closes the Save As dialog box.

Closing a Result Set

To close a result set, do the following:

1   On the *Main* menu, click Close.

OR

On the *Result* tool bar, click Close.

OR

In the upper-right corner of the window, double-click the System menu icon.

Rapid SQL closes the Result Set.

2   If you have not saved your result set, Rapid SQL prompts you to save the file. Click *Yes* to
save and *No* to close without saving.

For more information, see [Results Editor](#).

EDITING RESULT SETS

The Results Editor provides many ways to edit and customize your result windows. The Data Grid
offers a host of features for manipulating, sorting and formatting data.

Topics

- [Cutting, Copying, and Pasting Cell Contents](#)

- [Cutting, Copying, and Pasting Rows](#)

- [Cutting, Copying, and Pasting Columns](#)

- [Adding and Inserting Rows](#)

- [Adding and Inserting Columns](#)

- [Deleting Rows and Column](#)

- [Resizing Rows and Columns](#)

• [Sorting Data](#)

CUTTING, COPYING, AND PASTING CELL CONTENTS

The Result window supports standard cut, copy and paste functionality.

Cutting Cell Contents

To cut cell contents, do the following:

1 In the Results window, double click or tab to the target cell. A double line bounds the selected cell. You can also select text using standard text selection techniques.

2 On the *Edit* menu, click Cut.

    OR

    On the *Result* tool bar, click Cut.

    Rapid SQL cuts the cell.

Copying Cell Contents

To copy cell contents, do the following:

1 In the Results window, double click or tab to the target cell. A double line bounds the selected cell. You can also select text using standard text selection techniques.

2 On the *Edit* menu, click Copy.

    OR

    On the *Result* tool bar, click Copy.

    Rapid SQL copies the cell.

Pasting Cell Contents

To paste cell contents, do the following:

1 In the *Results* window, double click or tab to the target cell. A double line bounds the selected cell.

2 On the *Edit* menu, click Paste.

    OR

    On the *Result* tool bar, click Paste.

    Rapid SQL pastes the cell.

For more information, see [Editing Result Sets](#).

CUTTING, COPYING, AND PASTING ROWS

You can perform standard cut, copy, and paste functions on rows, just as you can on individual cells.

Cutting Rows

To cut a row, do the following:

1 In the Results window, click the numbered row heading on the left side of the row.

2 On the *Edit* menu, click Cut.

   OR

   On the *Result* tool bar, click Cut.

   Rapid SQL cuts the row.

Copying Rows

To copy rows, do the following:

1 In the Results window, double click or tab to the target row. A double line bounds the selected row. You can also select text using standard text selection techniques.

2 On the *Edit* menu, click Copy.

   OR

   On the *Result* tool bar, click Copy.

   Rapid SQL copies the row.

Pasting Rows

To paste rows, do the following:

1 In the Results window, double click or tab to the target row. A double line bounds the selected row.

2 On the *Edit* menu, click Paste.

   OR

   On the *Result* tool bar, click Paste.

   Rapid SQL pastes the row.

For more information, see [Editing Result Sets](#).

CUTTING, COPYING, AND PASTING COLUMNS

You can perform standard cut, copy, and paste functions on columns, just as you can on rows.

Cutting Columns

To cut columns, do the following:

1   In the Results window, click the column heading above the first row.

2   On the *Edit* menu, click Cut.

   OR

   On the *Result* tool bar, click Cut.

   Rapid SQL cuts the column.

Copying Columns

To copy columns, do the following:

1   In the Results window, click the column heading.

2   On the *Edit* menu, click Copy.

   OR

   On the *Result* tool bar, click Copy.

   Rapid SQL copies the column.

Pasting Columns

To paste columns, do the following:

1   In the Results window, click the column heading above the first row to select the target column.

2   On the *Edit* menu, click Paste.

   OR

   On the *Result* tool bar, click Paste.

   Rapid SQL pastes the column.

For more information, see [Editing Result Sets](#).

ADDING AND INSERTING ROWS

You can add or insert rows to expand or rearrange your result sets.

Adding a Row

To add a row, do the following:

1   To add a row as the last row of the result set, position the pointer inside the result set.

2   On the Edit menu, click *Add Row*.

3 To add a row inside the result set, click the numbered row heading where you want to add a row.

4 On the Edit menu, click *Add Row*.

Inserting a Row

To insert a row, do the following:

1 To insert a row as the last row of the result set, position the pointer inside the result set.

2 On the Edit menu, click Insert Row.

3 To insert a row inside the result set, click the numbered row heading where you want to insert a row.

4 On the Edit menu, click Insert Row.

For more information, see Editing Result Sets.

ADDING AND INSERTING COLUMNS

You can add or insert columns to expand or rearrange your result sets.

Adding a Column

To add a column, do the following:

1 Position the pointer inside the result set.

2 Select Edit, Add Column from the main menu. The new column is added as the last column of the result set.

Inserting a Column

To insert a column, do the following:

1 Select the column where you want to insert a column.

2 Select Edit, Insert Column from the main menu. The new column is inserted to the left of the column that you selected.

For more information, see Editing Result Sets.

DELETING ROWS AND COLUMNS

You can delete entire rows and columns to edit your result sets.

Deleting a Row

To delete a row, do the following:

1   Select the target row of data to delete.

2   On the *Edit* menu, click Delete Row.

    OR

    Right-click the row, and then click Delete Row.

    Rapid SQL deletes the row.

Deleting a Column

To delete a column, do the following:

1   Select the target column of data to delete.

2   On the *Edit* menu, click Delete Column.

    OR

    Right-click the column, and then click Delete Column.

    Rapid SQL deletes the column.

For more information, see [Editing Result Sets](#).

RESIZING ROWS AND COLUMNS

Resizing rows and columns can aid readability of the spreadsheet contents or condense space for editing and formatting purposes.

Resizing Rows to Their Default Height

To resize rows to their default height, do the following:

1   Select one or more rows by clicking on the numbered row headings to the left of the rows.

2   Right-click the selected rows, and then click *Resize Rows.*

Resizing Rows to a New Height

To resize rows to a new height, do the following:

1   Select one or more rows by clicking the numbered row headings to the left of the rows.

2   Change the pointer to a solid horizontal bar with arrows on top and bottom by moving it to one of the upper or lower borders of the row heading.

3   Click and grab the row border and drag the pointer to enlarge or shrink the height of the row.

Resizing Columns to Their Default Widths

To resize columns to their default widths, do the following:

1   Select one or more columns by clicking the column headings.

2   Right-click the selected columns, and then click *Resize Columns.*

Resizing Columns to a New Width

To resize columns to a new width, do the following:

1   Select one or more columns by clicking the column headings.

2   Change the pointer to a solid horizontal bar with arrows on top and bottom by moving it to one of the upper or lower borders of the column heading.

3   Click and grab the column border and drag the pointer to enlarge or shrink the height of the column.

For more information, see [Editing Result Sets](#).

SORTING DATA

To order and organize data in a coherent manner, you can sort columns alphanumerically in single result sets.

Sorting Data

To sort data, do the following:

1   In the data grid, select the column(s) you want to sort.

2   Double click the column header(s) to sort the data in the column in ascending, alphanumeric order. To sort the column in descending order, double click the column header again.

> **NOTE:**   This option is not valid for multiple result sets.

For more information, see [Editing Result Sets](#).

FORMATTING RESULT SETS

The ISQL Editor provides flexibility in formatting result sets, for analysis and reporting, from within a Result Window. Result sets can be formatted to best suit your purpose, whether it be sending via e-mail, printing, or exporting a file to other programs, such as Microsoft Excel. Some of these features change only the screen display of the results, while others allow you to format for printing.

Topics

- <u>Setting Alignment Properties</u>

- <u>Setting Border Properties</u>

- <u>Setting Result Set Display Properties</u>

- <u>Setting Fonts</u>

- <u>Setting Colors</u>

SETTING ALIGNMENT PROPERTIES IN RESULT WINDOWS

You can use the format menu or the shortcut menu to alter result set alignment properties. Selecting the Alignment command displays the Format Styles tabbed dialog box:

Setting Alignment Properties

To set alignment properties, do the following:

1 On the *Format* menu, click Alignment.

   OR

   Right-click the Result data grid, and then click Alignment.

   Rapid SQL opens the Format Styles dialog box.

2 Click the Align Tab.

3 To change *Horizontal* properties, click the appropriate *option* button in the *Horizontal* box.

4 To enable *Wrap Text*, *Allow Enter* and/or *Auto Size,* select the appropriate check boxes.

5 To disable *Wrap Text*, *Allow Enter* and/or *Auto Size,* select the appropriate check boxes and deselect these options.

The table below describes alignment properties:

Property	Option	Description
Horizontal	Standard	Aligns data in cells based on their default datatype alignment properties. For example, numbers are right aligned, whereas text is left aligned.
	Left Aligned	Forces all data to be left aligned, regardless of datatype
	Center	Forces centering of all data, regardless of datatype
	Right Aligned	Forces all data to be right aligned, regardless of datatype
Vertical	Standard	Aligns data in cells based on their default datatype alignment properties. For example, numbers are right aligned, whereas text is left aligned.

Property	Option	Description
	Top	Forces all data to be aligned to the top of each cell
	Center	Forces all data to be aligned in the center of each cell
	Bottom	Forces all data to be aligned at the bottom of each cell
Wrap Text		Specifies that data exceeding the length of the cell should wrap to additional lines in the same cell
Allow Enter		Specifies that the contents of cells can be edited or not
Auto Size		Specifies whether or not rows should size automatically to accommodate the longest data item in a column

For more information, see [Formatting Result Sets](#).

FORMAT BORDER STYLES

You can use the format menu or the shortcut menu to alter border properties. Selecting the Border command displays the Format Styles tabbed dialog box.

Completing the Format Styles Dialog Box

To complete the Format Styles dialog box, do the following:

1 On the *Format* menu, click Borders.

   OR

   Right-click the Result data grid, and then click Borders.

   Rapid SQL opens the Format Styles dialog box.

2 Click the Borders Tab.

3 On the *Border* box, you can indicate whether or not a border should appear on the top, bottom, right, left, or for a range of cells by clicking the corresponding boxes.

4 To set a range of cells apart by setting a particular border style around that range, select the range in the result set before opening the *Format Styles* dialog box. To select the *Range* property, click the range box.

5 In the *Type* box, you can select the type of line you want to border the cell or cells by clicking the corresponding boxes.

6 To select a color, click the *Color* list, and then click the border color.

7 Click OK.

   Rapid SQL saves your changes and closes the Format Styles dialog box.

For more information, see [Formatting Result Sets](#).

DISPLAY SETTINGS

You can use the format menu or the shortcut menu to alter display properties. Selecting the Properties command displays the Display Setting dialog box.

The Display Settings dialog box lets you set a number of format properties for the result set window:

- How titles and grid lines are displayed.

- How lines in the grid are displayed.

- How the result set window background is displayed.

- How current cells, rows and columns are indicated.

Completing the Display Settings Dialog Box

To complete the Display Settings dialog box, do the following:

1   On the *Format* menu, click Properties.

    OR

    Right-click the Result data grid, and then click Properties.

    Rapid SQL opens the Display Settings dialog box.

2   In the *Titles and Gridlines* box, select the target display properties check boxes. The change is displayed in the right-pane *Preview* window of the *Properties* dialog box.

3   *In the Color* box, click the property you want to change, and then click the new color. Changes are propagated automatically when you change them.

4   To alter the user properties, in the *User Properties* list, click the attribute you want applied to the current cell.

5   To save your changes to the current configuration, select the *Save* settings to profile check box.

6   Click OK.

    Rapid SQL saves your changes and closes the Properties dialog box.

For more information, see <u>Formatting Result Sets</u>.

FORMAT FONT STYLES

You can use the format menu or the shortcut menu to alter font properties. Selecting the Font command displays the Format Styles tabbed dialog box.

Completing the Format Styles Dialog Box

To complete the Format Styles dialog box, do the following:

1   On the *Format* menu, click Font.

>   OR

>   Right-click the Result data grid, and then click Font.

>   `Rapid SQL opens` the Format Styles dialog box.

2   Click the Font Tab.

3   In the *Font Combo* box, type or click the font you want to use.

4   In the *Outline Combo* box, type or click the outline you want to use.

5   In the *Size Combo* box, type or click the size you want to use.

6   To make a line cross through the length of the text, in the *Effects* box, select the *Strikeout* check box.

7   To underline the text, in the *Effects* box, select the *Underline* check box.

8   To change the text color, click the *Text Color* list, and then click the new color.

>   The Preview box displays the sample text of your selections

For more information, see [Formatting Result Sets](#).

FORMAT COLOR STYLES

You can use the format menu or the shortcut menu to alter color properties. Selecting the Color command displays the Format Styles tabbed dialog box.

The table below describes the options and functionality on the Format Styles dialog box:

Option	Description
Interior	To change the result set pattern, in the box, click the list, and then click a new pattern.
	To set the foreground color of the result set, in the box, click a foreground group color button.
	To set the background color of the result set, in the box, click a background group color button.

Option	Description
3D-Effect	To change the 3D-Effect, in the box, click the appropriate option button:
	The Normal option button provides a standard flat appearance.
	The Raised option button provides a raised 3D appearance in the selected cells.
	The Inset option button provides a depressed appearance in the selected cells. The default is typically set to Normal.

Completing the Format Styles Dialog Box

To complete the Format Styles dialog box, do the following:

1   On the *Format* menu, click Color.

    OR

    Right-click the Result data grid, and then click Color.

    Rapid SQL opens the Format Styles dialog box.

For more information, see [Formatting Result Sets](#).

NOTES ON XML TYPES AND UNICODE DISPLAY IN THE RESULTS EDITOR

When viewing data in the Results grid, keep the following in mind:

- XML data types are supported for IBM DB2 for Windows, Unix, and Linux, Microsoft SQL Server, and Oracle. In the Results grid, XML data types are displayed as LOB content.

- Support for display of Uincode characters is provided as follows:

  - IBM DB2 for Windows, Unix, and Linux V8 and V9: **character**, **clob**, **varchar**, and **longvarchar** types

  - SQL Server 2000: for **nchar**, **nvarchar**, and **ntext** types

  - SQL Server 2005: **nchar**, **nvarchar**, **ntext**, and **nvarchar(max)** types

  - Oracle 8i, 9i, and 10g: **NCHAR**, **NVARCHAR2** and **NCLOB** for non-Unicode UTF8 Character Set Instances and **NCHAR**, **NVARCHAR2**, **CHAR**, **VARCHAR2**, **LONG**, **NCLOB** and **CLOB** for Unicode UTF8 Character Set Instances

  - Sybase 12.5 and 15.2: **UNICHAR**,**UNIVARCHAR** and **UNITEXT** for non-Unicode UTF8 Character Set Instances and **UNICHAR**, **UNIVARCHAR**, **UNITEXT**, **NCHAR**, **NVARCHAR**, **CHAR**, **VARCHAR** and **TEXT** for Unicode UTF8 Character Set Instances

PERMISSIONS MANAGEMENT

Rapid SQL includes permissions management capabilities that include:

- [Explicit Permissions](#)
- [Cascading Permissions](#)
- [Using Roles to Grant Permissions and Privileges](#)
- [Using Roles to Revoke Permissions and Privileges](#)
- [Grant Privilege(s) To](#)
- [Revoke Privilege(s) From](#)
- Deny Privileges From

EXPLICIT PERMISSIONS

Explicit permissions are permissions specifically granted by one user to another. Granting a permission is an integral part of database security and management. Rapid SQL tracks explicit permission changes across your enterprise. Multiple explicit permissions can be consolidated in roles. Different groups and roles can share permissions for an object. Rapid SQL lets you grant permissions with the Roles Editor. Embarcadero lets you edit permissions on the Privileges tabs of the object editors.

> **TIP:** The Admin option is similar to the grantable option for privileges. If the user has the admin option, they can grant that role to other people.

> **TIP:** In most cases, you will want to make all roles granted, default roles. If you do not make a role default for a user, they will have to enable it with a SQL command. To avoid this complication, assign roles as default unless business rules specify otherwise.

> **TIP:** The Grantable option gives the user the ability to grant that privilege to other users. Use the Grantable option SPARINGLY.

For more information, see [Permissions Management](#).

CASCADING PERMISSIONS

Cascading permissions are the path of privileges granted from one user to another user, group, or role. Using cascading permissions is a way to manage multiple sets of permissions and privileges for groups of users. When you drop a user with a revoke cascade command, all permissions and privileges granted by the dropped user are also revoked. Embarcadero lets you edit and set grant permission parameters with the Permissions Editor.

Once a user with grant permission privileges is dropped with cascade, reinstate permissions for all affected users.

For more information, see [Permissions Management](#).

USING ROLES TO GRANT PERMISSIONS AND PRIVILEGES

Roles are sets of user privileges you associate with access to objects within a database. Roles streamline the process of granting permissions. You can use roles to grant sets of permissions and privileges to users and groups. Rapid SQL lets you grant permissions to roles with the Roles Editor.

For more information, see [Permissions Management](#).

USING ROLES TO REVOKE PERMISSIONS AND PRIVILEGES

Roles can be effective in revoking permissions and privileges from users. Rather than individually revoke permissions from users, you can revoke groups of permissions from multiple users simultaneously using roles. Rapid SQL lets you revoke permissions with the Roles Editor.

Rapid SQL lets you identify existing users and their database permissions, and to detect and manage changes to user permissions by rolling change back or forward. Rapid SQL lets you manage database permissions in a cross-platform database environment. and gives you the ability to compare objects/permissions and migrate them to other instances. Using the compare functionality for permissions management, Rapid SQL gives you:

- Enhanced permissions management across the enterprise (Oracle, Sybase ASE, Microsoft SQL Server, and IBM DB2 LUW for Linux, Unix, and Windows).

- Faster detection of changed user rights.

- Ability to detect database accounts (users/logins) that are invalid.

- Rollback of invalid permissions in a single click.

- Archives of users, groups, roles and their permissions.

For more information, see [Permissions Management](#).

GRANT PRIVILEGE(S) TO

The Grant Privilege(s) To dialog box lets you select or clear the check boxes corresponding to the target privileges.

Permissions and privileges allow or limit access to system database administration and database objects. To manage databases, it is important to monitor who can access the enterprise structure and change the underlying schema. It is also important to monitor who can grant and revoke permissions and privileges in the enterprise. Rapid SQL lets you manage permissions and privileges in the Grant Privilege(s) To dialog box.

On the Privileges Tab of the editor, you can see whether a privilege was inherited from a role or group, or if it was granted explicitly by a user.Rapid SQL also shows if the privilege is grantable (granted with the GRANT OPTION.) The table below describes the icons:

NOTE: The available privileges depend on the target DBMS and object.

The table below describes the options and functionality on the Grant Privilege(s) From Dialog Box:

Option	Description
Privilege	Lets you select or clear the check boxes corresponding to the target privileges.
Grantable	Select No to prevent cascading the privileges to other users.

The table below describes the object permissions:

Object	Permission(s)
Index	CONTROL
Packages	BIND, CONTROL, EXECUTE
Schema	ALTERIN, CREATIN, DROPIN (w/GRANT OPTION)
Tables	ALTER, CONTROL, DELETE, INDEX, INSERT, REFERENCES (& on column), SELECT, UPDATE (& on column) (w/GRANT OPTION)
Tablespaces	USE (w/GRANT OPTION)
Views	CONTROL, DELETE, INSERT, SELECT, UPDATE (& on column) (w/GRANT OPTION)

For more information, see:

[Permissions Management](#)

REVOKE PRIVILEGE(S) FROM

The table below describes the options and functionality on the Revoke Privilege(s) From dialog box:

Option	Description
Privilege	Lets you select or clear the check boxes corresponding to the target privileges.
Cascade	Select No to prevent cascading the revocation privileges to other users.

For more information, see:

[Permissions Management](#)

DENY PRIVILEGES FROM

> **NOTE:** Deny Privileges is available for Microsoft SQL Server only.

The Deny Privileges From dialog box lets you deny a privilege from a security account in the current database.

The table below describes the options and functionality on the Deny Privilege(s) From dialog box:

Option	Description
Privilege	Lets you select or clear the check boxes corresponding to the target privileges.
Cascade	Select No to prevent cascading the deny privileges to other users.

> **TIP:** On the editor, the Deny privilege can be Revoked, just as a Grant permission can be revoked.

For more information, see:

[Permissions Management](#)

PROJECT MANAGEMENT

Rapid SQL offers the following project managment-oriented functions:

- Project support

- Version Control

PROJECT SUPPORT

Rapid SQL database project management facilities help you organize, alter, and keep track of changes to database objects or SQL scripts. The project management facilities act as a repository to maintain all source code for a database project. Rapid SQL also incorporates version control functions and build management facilities to help you manage and build projects. Once a project has been created, Rapid SQL lets you:

- Review a file's history.

- Return to earlier versions of a file.

- Develop concurrently.

A project generally includes SQL script files that you can maintain and create in unison with your database administration and development cycle. Projects can also contain subfolders (for easy categorization) as well as other projects (subprojects.) You can create projects manually by inserting existing SQL script files. You can also create projects automatically by reverse-engineering a database schema or an existing version control project.

When administering or developing SQL database schemas, SQL source code files are the foundation of effective change management for database objects. The DDL commands used to create database objects on SQL database servers are often in a constant state of flux as new columns and constraints are added to table schemas or stored procedure logic is changed. The ability to track and store these changes to files directly from Rapid SQL alleviates any disruption in the development environment.

Lengthy script files containing the DDL to compile stored procedures and triggers on a database server such as Oracle or Sybase Adaptive Server go through constant revisions similar to C program files and word processing documents. Rapid SQL's Project Management facilities let you effectively monitor revisions for your database servers.

For more information, see:

Create a New Project

Working With Projects

CREATE A NEW PROJECT

Creating a project can be a very time-consuming task. Rapid SQL eases this task by providing step-by-step wizards to guide you through the process. You have a variety of options to choose from when you create a new project. Rapid SQL lets you create new projects:

- From a database.

- From existing files.

- From a version control project.

- Without initialization.

The New Project dialog box lets you select how you want to create a new project. Using the New Project dialog box, you can name, specify a file location, and provide a description of the project.

NEW PROJECT DIALOG BOX

The New Project dialog box lets you select how you want to create a new project. Using the New Project dialog box, you can name, specify a file location, and provide a description of the project.

Important Notes

None

The table below describes the options and functionality of the New Project dialog box:

Option	Description
Name	Lets you enter a name for the project.
Location	Lets you type or browse and locate a directory for the project.
	NOTE: This is set as the working directory for your project.
Description	OPTIONAL: Lets you enter a description of the project.
Initialize New Project	Lets you select how you want to create a new project.
	From Database - Lets you reverse engineer an existing database.
	From Existing Files - Lets you create a project from scratch, or use an existing project.
	From Version Control - Lets you reverse-engineer a project from existing version control system projects. This is helpful for users who have already created database projects in their version control systems.
	Do Not Initialize - Lets you create an empty project where you can later add files and/or database objects.

For more information, see Completing the New Project Dialog Box.

COMPLETING THE NEW PROJECT DIALOG BOX

To complete the New Project Dialog Box, do the following:

1   On the File menu, select New, and then Project.

    Rapid SQL opens the New Project Dialog Box.

2   In Name, enter a name for the project.

3   In Location, type or browse and locate a directory for the project.

4   OPTIONAL: In Description, lets you enter a description of the project.

5   In Initialize New Project, select how you want to create a new project.

6   Click OK.

CREATE A NEW PROJECT FROM A DATABASE

Rapid SQL offers a context-sensitive wizard that reverse-engineers all or part of an existing database. When extracting database schema, Rapid SQL captures the complete definition of the following objects:

- Tables

- Views

- Stored Procedures

- Functions

Rapid SQL then creates a project containing files based on the relevant object types. Rapid SQL creates a separate file for each database object.

Reverse engineering is a powerful tool for analyzing, controlling, and documenting existing database objects. You can use the extracted SQL source code for archival and reference purposes.

Important Notes

- You must create a New Project from a Database in order to access the New Project Reverse Engineering Wizard.

- Rapid SQL automatically discovers and sets the project script file build order by referencing the system catalog to determine dependencies.

- In the New Project dialog box, you must select the From Database option.

For more information, see Creating a New Project from a Database.

CREATING A NEW PROJECT FROM A DATABASE

To create a new project, do the following:

1   On the *File* menu, click New, and then click Project.

    OR

    On the *Main* toolbar, click the drop-down arrow to the right of the New button.

    OR

    On the Project toolbar, click the New Project.

    OR

    In the workspace, right-click *New, and then click Project.*

    Rapid SQL opens the New Project dialog box.

2   Complete the New Project dialog box and select the *From Database option*.

3   Click *OK*.

    Rapid SQL opens the New Project Reverse Engineering Wizard.

4   Complete the New Project Reverse Engineering Wizard.

COMPLETING THE NEW PROJECT REVERSE ENGINEERING WIZARD

   NOTE: You must create a New Project from a Database in order to access the New Project Reverse Engineering Wizard.

   NOTE: Rapid SQL automatically discovers and sets the project script file build order by referencing the system catalog to determine dependencies.

To complete the New Project Reverse Engineering Wizard, do the following:

1   Complete the New Project dialog box, and then click OK.

    NOTE: You must select the From Database option.

    Rapid SQL opens the New Project Reverse Engineering Wizard.

2   Complete the wizard.

3   To make changes, click *Back* to the appropriate panels of the Wizard and make changes.

4   When you are satisfied with the definition, click *Execute*.

    Rapid SQL starts creating project files from database.

For more information, see New Project Reverse Engineering Wizard - Panel.

NEW PROJECT REVERSE ENGINEERING WIZARD - PANEL 1

The first panel of the *New Project Reverse Engineering Wizard* lets you specify the datasource that you want to reverse engineer into a project.

The table below describes the options and functionality of the first panel of the New Project Reverse Engineering Wizard:

Option	Description
Select a Datasource	Lets you select a datasource to reverse engineer. Select a datasource.

For more information, see [Completing the New Project Reverse Engineering Wizard](#).

NEW PROJECT REVERSE ENGINEERING WIZARD - PANEL 2

The second panel of the New Project Reverse Engineering Wizard lets you specify the target database on the server.

The table below describes the options and functionality of the second panel of the New Project Reverse Engineering Wizard:

Option	Description
Select Database to reverse-engineer	Lets you select a database to reverse engineer. Select a database.

For more information, see [Completing the New Project Reverse Engineering Wizard](#).

NEW PROJECT REVERSE ENGINEERING WIZARD - PANEL 2

The third panel of the New Project Reverse Engineering Wizard lets you specify the database object owners, the database object types, and general script types of the objects you want to extract.

The table below describes the options and functionality of the third panel of the New Project Reverse Engineering Wizard:

Option	Description
Select Database Object Types	Owner list - Lets you select the owner of the objects you want to extract.
	Objects - Lets you select the objects types you want to extract.
	TIP: To select or deselect all object types right-click to Select or Unselect all.
Extract Scope	Lets you select the objects to extract from the database.
	All Objects - Lets you extract all objects for each object type that you select. Opens [Panel 4 for All Objects](#).
	Selected Objects Only - Lets you specify the specific objects that you want to extract. Opens [Panel 4 for Selected Objects](#).
	NOTE: Rapid SQL continues to a different Panel 4 depending on which option you choose for the Extract Scope.

Option	Description
Include Indexes with Table DDL	If selected, includes indexes when extracting table DDL.
Include FKs with Table DDL	If selected, includes foreign keys when extracting table DDL.
Include Drop Statement	If selected, includes drop statements when extracting database SQL.
Include Object Privileges	If selected, includes object privileges when extracting database SQL.

For more information, see [Completing the New Project Reverse Engineering Wizard](#).

New Project Reverse Engineering Wizard - Panel 3 for All Objects

This panel of the New Project Reverse Engineering Wizard lets you specify the ownership.

The table below describes the options and functionality of the fourth panel of the New Project Reverse Engineering Wizard for All Objects:

Option	Description
Retain	Select to include the existing object owners' names in the CREATE statements.
Exclude	Select to exclude the object owners' names in the CREATE statements.
Transfer	Select to include the designated owner's name in the CREATE statements. Select a user.

For more information, see [Completing the New Project Reverse Engineering Wizard](#).

NEW PROJECT REVERSE ENGINEERING WIZARD - PANEL 4 FOR SELECTED OBJECTS

This panel of the New Project Reverse Engineering Wizard only applies if you have chosen to generate a customized script. You can select specify objects and set script options for each object type.

The table below describes the options and functionality of the fourth panel of the New Project Reverse Engineering Wizard for selected objects.

> NOTE: You must select options and statements for each object type you select.

Option	Description
Object Type	Lets you select an object type.
Object list	Lists the owner and names of objects for the selected type in the database.
Options for Objects	Lets you select an option for the object type. NOTE: Options vary depending on the object type selected.

Option	Description
Statements	Lets you select a script option for the particular object type. NOTE: Statements vary depending on the option and object type selected.

For more information, see [Completing the New Project Reverse Engineering Wizard](#).

NEW PROJECT REVERSE ENGINEERING WIZARD - PANEL 4

This panel of the New Project Reverse Engineering Wizard lets you review the Reverse Engineering definition to verify its accuracy.

> **NOTE:** If you have specified to integrate Rapid SQL with an underlying version control system during installation, Rapid SQL asks to add the new files to version control. Click Yes to add the files.

For more information, see [Completing the New Project Reverse Engineering Wizard](#).

CREATE A NEW PROJECT FROM EXISTING FILES

Rapid SQL lets you create a new project from:

- Scratch.

- Existing files.

To create a new project from an existing file, do the following:

1 On the *File* menu, click New, and then click Project.

   OR

   On the *Main* toolbar, click the drop-down arrow to the right of the New button.

   OR

   On the Project toolbar, click the New Project.

   OR

   In the workspace, right-click *New, and then click Project.*

   Rapid SQL opens the New Project dialog box.

2 Complete the [New Project dialog box](#) and select the From Existing Files option.

3 Click *OK.*

   Rapid SQL opens the Add File(s) to Project dialog box.

4 In the *Add File(s) to Project* dialog box, type the files that you want to add in the *File* name field. Use the following to help locate the files:

- In Directories, browse by clicking a directory.

- In the *List* files of type list, select a file type.

- In the Drives list, select a drive.

5 Click Add to add a file or click Add All to add all the files.

6 When you finish, click *OK*.

Rapid SQL creates the project.

NOTE: If you have specified to integrate Rapid SQL with an underlying version control system during installation, Rapid SQL asks to add the new files to version control. Click Yes to add the files.

CREATE A NEW PROJECT FROM A VERSION CONTROL PROJECT

Rapid SQL lets you reverse-engineer a project from existing version control system projects. This is helpful if you have already created database projects in a version control system.

NOTE: The Intersolv PVCS API does not support the creation of a project from a source code control project.

Rapid SQL also lets you create a project from a version control project that contains sub-directories, while including files from those sub-directories in the Rapid SQL project.

To create a new project from a Version Control project, do the following:

1 On the *File* menu, click New, and then click Project.

OR

On the *Main* toolbar, click the drop-down arrow to the right of the New button.

OR

On the Project toolbar, click the New Project.

OR

In the workspace, right-click *New, and then click Project.*

Rapid SQL opens the New Project dialog box.

2 Complete the [New Project dialog box](#) and select *From Version Control Project.*

3 Click *OK*.

Rapid SQL opens the Choose project from (Version Control name) dialog box.

4 Select the project or files you want to include, and then click *OK*.

Rapid SQL opens the Files to be included in (project path) dialog box.

5   Select the project of files you want to include, and then click OK.

Rapid SQL creates the project.

CREATE A NEW PROJECT WITHOUT INITIALIZATION

Rapid SQL gives you the option to not initialize a project. Rapid SQL creates a project tab with the name you specify in the New Project dialog box. This tab functions as a shell to:

- Add files.

- Add database objects.

To create a new project without initialization, do the following:

1   On the *File* menu, click New, and then click Project.

OR

On the *Main* toolbar, click the drop-down arrow to the right of the New button.

OR

On the Project toolbar, click the New Project.

OR

In the workspace, right-click *New, and then click Project.*

Rapid SQL opens the New Project dialog box.

2   Complete the New Project dialog box and select the *Do Not Initialize* option.

3   Click *OK.*

Rapid SQL creates the Project Tab.

WORKING WITH PROJECTS

A project is similar to a file system. Both are a collection of files that you create and maintain. Because projects are hierarchical, you can place a subproject under another project, and a subproject under a subproject. Once you have created a new project, Rapid SQL provides many functions to help you maintain and modify the project.

Related Topics

New Project Dialog Box

Opening an Existing Project

Opening a Recent Project

Closing a Project

[Build Project](#)

[Set Build Order](#)

[Add Database Object File(s) to Project Wizard](#)

[Execute Project Files](#)

[Add File(s) to a Project](#)

[Open a File from a Project](#)

[Subprojects](#)

[Project Properties](#)

[Confirm Delete Dialog Box](#)

OPENING AN EXISTING PROJECT

To open an existing project, do the following:

1   On the *File* menu click *Open Project.*

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

    NOTE: Project files are designated with a *.epj extension.

3   Click Open.

    Rapid SQL opens the Project Tab containing the project.

For more information, see [Working with Projects.](#)

OPENING A RECENT PROJECT

To open a recent project, do the following:

1   On the *File* menu select *Recent Project,* and then select a project.

    Rapid SQL opens the Project Tab containing the project.

For more information, see [Working with Projects](#).

CLOSING A PROJECT

To close a project, do the following:

1   On the *File* menu, click *Close* Project.

2   On the *Project Tab*, right-click and then click *Close* Project.

    Rapid SQL closes the project.

For more information, see <u>Working with Projects</u>.

BUILD PROJECT

The Build Project dialog box lets you:

- Generate a project build script and display it in a SQL window.

- Execute the project build immediately on build target.

- Schedule a project build.

Important Notes

None

The table below describes the Build Project dialog box:

Option	Description
Datasource	Displays the target datasource.
Database	Displays the target database.
Build subprojects	Select to include subprojects.
Generate a Project Build Script and Display it in a SQL window.	Select to generate the project build script and have Rapid SQL display it in a SQL window.
Execute Project Build Immediately on Build Target	Select to execute build immediately.
Schedule Project Build for a Later Time	Select to schedule the build and then specify optional options.

For more information, see <u>Completing the Build Project Dialog Box</u>.

COMPLETING THE BUILD PROJECT DIALOG BOX

To complete the Build Project dialog box, do the following:

1   On the *File* menu click *Open Project*.

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

   NOTE: Project files are designated with a *.epj extension.

3   Click Open.

   Rapid SQL opens the Project Tab containing the project.

1   On the Project Tab, right-click the target project, and then select Build.

   Rapid SQL opens the Build Project dialog box.

2   Select options.

3   Click OK.

   Rapid SQL builds the project according to your specifications.

SET BUILD ORDER

You can specify the order in which you want Rapid SQL to build your project files in the Set Build Order dialog box. If you created your project manually or from a version control project, you must specify a build order, otherwise the files are built in the order that they appear in the tree of the Project Tab.

Setting Build Order

1   On the *File* menu click *Open Project*.

   Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

   NOTE: Project files are designated with a *.epj extension.

3   Click Open.

   Rapid SQL opens the Project Tab containing the project.

4   On the *Project* menu, click Build Order.

   OR

   On the *Project* toolbar, click Build Order.

   OR

   On the Project Tab, right-click a project file, and then click Build.

   Rapid SQL opens the Set Build Order dialog box.

5   Click the files you want to move and then click the *Up* and *Down* to change the order to build the files.

6   When you finish specifying the order, select *OK*.

Rapid SQL sets the build order.

> **NOTE:**   The next time you build the project, Rapid SQL uses this build order.

ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD

You can add database objects to an existing project using a simple wizard that can reverse-engineer an entire database or any portion of it. This lets you keep your project in sync with databases where objects are constantly being created and updated.

For more information, see:

Working with Projects

Completing the Add Database Object File(s) to Project Wizard

COMPLETING THE ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD

To add database objects to a project, do the following:

1   On the *File* menu click *Open Project*.

Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

> **NOTE:**   Project files are designated with a *.epj extension.

3   Click Open.

Rapid SQL opens the Project Tab containing the project.

4   On the *Project* menu, click Add Database Objects.

OR

On the *Project* toolbar click Add Database Objects.

OR

On the Project Tab, right-click and then click Add Database Objects.

Rapid SQL opens the first panel of the Add Database Object File(s) to Project Wizard.

For more information, see Add Database Objects to a Project Wizard.

ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD - PANEL 1

The first panel of the Add Database Object File(s) to Project Wizard lets you specify the datasource that you want to reverse engineer into a project.

The table below describes the options and functionality of the first panel of the Add Database Object File(s) to Project:

Option	Description
Select a Datasource	Lets you select a datasource to reverse engineer.

For more information, see [Completing the Add Database Object File(s) to Project Wizard](#).

ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD - PANEL 2

The second panel of the Add Database Object File(s) to Project Wizard lets you specify the target database on the server.

The table below describes the options and functionality of the second panel of the Add Database Object File(s) to Project:

Option	Description
Select Database to reverse-engineer	Lets you select a database to reverse engineer.

For more information, see [Completing the Add Database Object File(s) to Project Wizard](#).

ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD - PANEL 3

The third panel of the Add Database Object File(s) to Project Wizard lets you specify the database object owners, the database object types, and general script types of the objects you want to extract.

The table below describes the options and functionality of the third panel of the Add Database Object File(s) to Project:

Option	Description
Select Database Object Types	Owner list - Lets you select the owner of the objects you want to extract.
	Objects - Lets you select the objects types you want to extract.
	TIP: To select or deselect all object types, in the Object list, right-click to Select or Unselect all.
Extract Scope	Lets you select the objects to extract from the database.
	All Objects - Lets you extract all objects for each object type that you select. Opens [Panel 4 for All Objects](#).
	Selected Objects Only - Lets you specify the specific objects that you want to extract. Opens [Panel 4 for Selected Objects](#).
	NOTE: Rapid SQL opens a different Panel 4 depending on which option you choose for the Extract Scope.
Include Indexes with Table DDL	Includes indexes when extracting table DDL.
Include FKs with Table DDL	Includes foreign keys when extracting table DDL.
Include Drop Statement	Includes drop statements when extracting database SQL.

Option	Description
Include Object Privileges	Includes object privileges when extracting database SQL.

For more information, see [Completing the Add Database Object File(s) to Project Wizard](#).

ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD - PANEL 4 FOR SELECTED OBJECTS

This panel of the Add Database Object File(s) to Project Wizard only applies if you have chosen to generate a customized script. You can select specify objects and set script options for each object type.

The table below describes the options and functionality of the fourth panel of the Add Database Object File(s) to Project for selected objects.

> NOTE: You must select options and statements for each object type you select.

Option	Description
Object Type list	Lets you select an object type.
Object list	Lists the owner and names of objects for the selected type in the database.
Options for Object list	Lets you select an option for the object type. NOTE: Options vary depending on the object type selected.
Statement list	Lets you select a script option for the particular object type. NOTE: Statements vary depending on the option and object type selected.

For more information, see [Completing the Add Database Object File(s) to Project Wizard](#).

Add Database Object File(s) to Project Wizard - Panel 4 for All Objects

The fourth panel of the Add Database Object File(s) to Project Wizard lets you specify the ownership.

The table below describes the options and functionality of the fourth panel of the Add Database Object File(s) to Project for All Objects:

Option	Description
Retain	Select to include the existing object owners' names in the CREATE statements.
Exclude	Select to exclude the object owners' names in the CREATE statements.
Transfer	Select to include the designated owner's name in the CREATE statements. Select a user.

For more information, see [Completing the Add Database Object File(s) to Project Wizard](#).

ADD DATABASE OBJECT FILES(S) TO PROJECT WIZARD - PANEL 5
This panel of the Add Database Object File(s) to Project Wizard lets you retain, exclude, or transfer object ownership.

The table below describes the options and functionality on this panel:

Option	Description
Retain	Select to retain object ownership.
Exclude	Select to exclude object ownership.
Transfer ownership to following user:	Select to transfer ownership, and then select the target user from the list.

For more information, see Completing the Add Database Object File(s) to Project Wizard.

ADD DATABASE OBJECT FILE(S) TO PROJECT WIZARD - PANEL 6
This panel of the Add Database Object File(s) to Project Wizard lets you review the Reverse Engineering definition to verify its accuracy.

NOTE: If you have specified to integrate Rapid SQL with an underlying version control system during installation, Rapid SQL asks to add the new files to version control. Click Yes to add the files.

For more information, see Completing the Add Database Object File(s) to Project Wizard.

EXECUTE PROJECT FILES

You can directly execute project script files from the Project Tab using the File Execution Facility. You can also execute multiple scripts in parallel against different datasources. If a file has been placed under version control, you need to perform a Check Out operation to execute the file. Otherwise, the file opens in read-only mode.

For more information, see:

Working with Projects

Executing Project Files

EXECUTING PROJECT FILES
To execute project files, do the following:

1 On the *Project Tab*, right-click the files you want to execute, and then click *File Execution Facility.*

    Rapid SQL opens the Script Execution Facility dialog box.

For more information, see Execute Project Files.

ADD FILE(S) TO A PROJECT

You can add external files to your project. This lets you manipulate and expand your project as needed.

The Add File(s) to Project dialog box lets you add files to a project.

The table below describes the options and functionality of the Add Files to Project dialog box:

Option	Description
File Name	Lets you type the files that you want to add.
Directories	Lets you browse by select a directory.
List files of type	Lets you select a file type.
Drives	Lets you select a drive.
Network	Open the Map Network Drive dialog box.
Add	Lets you add a file.
Add All	Lets you add all the files.

For more information, see:

Working with Projects

Completing the Add Files to Project Dialog Box

COMPLETING THE ADD FILE(S) TO PROJECT DIALOG BOX

To add files to a project from, do the following:

1   On the *Project* menu, click Add Files.

    OR

    On the Project Tab, right-click and then click Add Files.

    Rapid SQL opens the Add Files to Project dialog box.

2   In *File name*, type the file name(s) or select the file(s) that you want to add to the project.

3   In Directories browse for the project. that contains the file(s) that you want to add to the project.

4   In the *List files of type* select the type of files you want to add to the project.

5   In Drives select a drive.

6   Click Network to open the Map Network Drive dialog box.

7   Click *Add* to add a file or click *Add All* to add all of the files.

8   Click *OK*.

    Rapid SQL adds the files to the project.

OPEN A FILE FROM A PROJECT

You can open a file directly from the Project Tab using a number of different methods.

For more information, see:

[Working with Projects](#)

[Opening a File from a Project](#)

OPENING A FILE FROM A PROJECT

To open a file from a project, do the following:

1 On the *Project Tab*, right-click the file(s) you want to open, and then click *Open*.

   OR

   On the *Project Tab*, double-click the file(s).

   Rapid SQL opens the file(s) in a new SQL window.

For more information, see [Open a File from a Project](#).

SUBPROJECTS

Subprojects are projects within projects. you can use them to help categorize your source code files. On the Project Tab of the Database Explorer Rapid SQL lets you:

- [Create subprojects](#).
- [Delete subprojects](#).
- [Rename subprojects](#).
- [Sort subprojects](#).

For more information, see [Working with Projects](#).

CREATING A NEW SUBPROJECT

Rapid SQL offers three ways to create a new subproject:

1 On the *File* menu click *Open Project*.

   Rapid SQL opens the Open Project dialog box.

2 In *File name*, type the name and location of the project or use browse to locate the project.

   NOTE: Project files are designated with a *.epj extension.

3 Click Open.

   Rapid SQL opens the Project Tab containing the project.

4   On the *Project* menu, click New SubProject.

    OR

    On the *Project* toolbar, click New SubProject.

    OR

    On the Project Tab, right-click the project or subproject, and then click New SubProject.

5   Type the name of the new subproject and then press *Enter.*

For more information, see [Subprojects](#).

DELETING A SUBPROJECT

To delete a project, do the following:

1   On the *File* menu click *Open Project.*

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

    NOTE: Project files are designated with a *.epj extension.

3   Click Open.

    Rapid SQL opens the Project Tab containing the project.

4   On the Project Tab, right-click a subproject, and then click Delete.

    OR

    On the Project Tab, select a subproject, and then click Delete.

    Rapid SQL opens the Confirm Delete dialog box.

5   In the *Confirm Delete* dialog box, select the *Delete* local copy to delete the local copy of the subproject, and then click *OK.*

    Rapid SQL deletes the subproject.

For more information, see [Subprojects](#).

CONFIRM DELETE DIALOG BOX

The table below describes the options and functionality on the Confirm Delete Dialog Box:

Option	Description
Delete local copy	Deletes the local copy of the subproject.

For more information, see [Deleting a Subproject](#).

RENAMING A SUBPROJECT

To rename a subproject, do the following:

1   On the *File* menu click *Open Project.*

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

    NOTE: Project files are designated with a *.epj extension.

3   Click Open.

    Rapid SQL opens the Project Tab containing the project.

4   On the *Project Tab*, right-click a subproject, and then click *Rename.*

5   Type the name of the new subproject and then press *Enter.*

    Rapid SQL renames the subproject.

For more information, see Subprojects.

SORTING SUBPROJECTS

So sort a subproject, do the following:

1   On the *File* menu click *Open Project.*

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

    NOTE: Project files are designated with a *.epj extension.

3   Click Open.

    Rapid SQL opens the Project Tab containing the project.

4   On the *Project Tab*, right-click the directory containing the subprojects you want to sort and then click *Sort.*

    Rapid SQL sorts the subproject(s).

For more information, see Subprojects.

PROJECT PROPERTIES

Rapid SQL lets you view properties of Projects, Subprojects and individual files in Projects or Subprojects. The Project Properties, Subproject Properties, and File Properties dialog boxes display information about the Projects, Subprojects, and files.

For more information, see Working with Projects.

VIEWING PROJECT PROPERTIES

To view project properties, do the following:

1   On the *File* menu click *Open Project.*

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

    NOTE: Project files are designated with a *.epj extension.

3   Click Open.

    Rapid SQL opens the Project Tab containing the project.

4   On the *Project* menu, click Project Properties.

    OR

    On the Project Tab, right-click a project and then click Properties.

    Rapid SQL opens the Project File Properties dialog box.

5   Complete the Project File Properties dialog box.

6   Click OK.

For more information, see:

Project Properties

Project File Properties Dialog Box

PROJECT FILE PROPERTIES DIALOG BOX

The table below describes the options and functionality of the Project Properties dialog box:

Option	Description
Name	Displays the name of the project.
Full Specification	Displays the project location.
Description	Displays the project description. OPTIONAL: Lets you type or edit the project description.
Associated Datasource	Lets you select a datasource for the project.
Associated Database	Lets you type the name of the database.

For more information, see Viewing Project Properties.

VIEWING SUBPROJECT PROPERTIES
To view subproject properties, do the following:

1   On the *File* menu click *Open Project*.

    Rapid SQL opens the Open Project dialog box.

2   In *File name*, type the name and location of the project or use browse to locate the project.

    NOTE: Project files are designated with a *.epj extension.

3   Click Open.

    Rapid SQL opens the Project Tab containing the project.

4   On the Project Tab, select a subproject.

5   Right-click the subproject, and then click Properties.

    Rapid SQL opens the Subproject Properties dialog box.

6   Complete the [Subproject Properties dialog box](#).

7   Click OK.

For more information, see:

[Project Properties](#)

[Subproject Properties Dialog Box](#)

SUBPROJECT PROPERTIES DIALOG BOX
The table below describes the options and functionality of the Subproject Properties dialog box:

Option	Description
Name	Displays the name of the subproject. You can enter or edit the subproject name.
Full Specification	Displays the subproject location.
Status	Displays the subproject status.

For more information, see [Viewing Subproject Properties](#).

VIEWING FILE PROPERTIES
To view file properties, do the following:

1   Click File and then Open Project.

    Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

Rapid SQL opens the project on the Projects Tab.

4   On the Project Tab, select a file.

5   Right-click the file, and then click Properties.

Rapid SQL opens the File Properties dialog box.

6   Complete the [File Properties dialog box](#).

7   Click OK.

For more information, see:

[Project Properties](#)

[File Properties Dialog Box](#)

FILE PROPERTIES DIALOG BOX
The table below describes the options and functionality of the File Properties dialog box:

Option	Description
Name	Displays the file name.
Full Specification	Displays the file location.
Description	Displays the file description. OPTIONAL: Type or edit the file description.
Include In Build	Sets the file to be included in the build.
Last Modified	Displays the date and time of the last modification.
Size	Displays the file size.
Status	Displays the file status.
Object Type	Displays the file object type. Select an object type if the file is unspecified.

Version Control

Version control archives files and tracks changes to files over time. With an integrated version control in Rapid SQL, you can easily track changes to database objects.

Version control addresses the following issues:

- Team Development By controlling access to a file so that only one person at a time can modify, it prevents accidental replacement or loss of another user's changes.

- Version Tracking By archiving and tracking versions of source code files, you can to retrieve them if necessary, thereby effectively creating files so that source code can be reused.

- Safety By adding database object scripts and files, it creates backups in case of loss, thereby ensuring a recovered version of source code.

When you create a project, Rapid SQL lets you place the project immediately into version control. You can also add projects and files to version control later.

To use integrated version control in Rapid SQL, you must have the version control client software installed on the same computer as Rapid SQL. You must also select the appropriate version control system during installation or after installation on the [Version Control Tab of the Option Editor](#).

Rapid SQL offers version control integration for the following version control systems:

- Rational ClearCase 5.0 and 6.0
- [Merant/Intersolv PVCS Version Manager version 6.0](#)
- [Microsoft Visual SourceSafe 5.0 and 6.0](#)
- [MKS Source Integrity version 7.3c](#)

For more information, see:

[Version Control Integration](#)

[Version Control Configuration](#)

[Using Version Control](#)

Version Control Integration

Rapid SQL offers version control integration so you can take advantage of the following version control systems:

- Rational ClearCase 5.0 and 6.0
- [Merant Version Manager version 6.0 or later](#)
- [Microsoft Visual SourceSafe 5.0 and 6.0](#)
- [MKS Source Integrity version 7.3c](#)

For more information, see:

[Integrating with Merant Version Manager](#)

[Integrating with Microsoft Visual Source Safe](#)

[Integrating with MKS Source Integrity](#)

INTEGRATING WITH MERANT VERSION MANAGER

Rapid SQL works with Merant Version Manager version 6.0. or later.

To integrate Merant Version Manager with Rapid SQL, do the following:

1   Install Merant Version Manager (formerly PVCS) 8.0.

2   In the setup option for the client, select IDE Client.

    This installs the correct DLLs and registry entries, and "Merant Version Manager" appears in the Version Control Tab of the Options Editor.

    NOTE: To use Rapid SQL with an existing Merant project, you must import the individual files into a native Rapid SQL project. Merant does not provide the third party API support for projects that it does for basic version control operations on archived files.

For more information, see [Version Control Integration](#).

INTEGRATING WITH MICROSOFT VISUAL SOURCESAFE

Rapid SQL works with Microsoft Visual SourceSafe versions 5.0. and 6.0. Rapid SQL uses your default Visual SourceSafe database unless you override the setting on the Version Control Tab of the Option Editor. This file is always called srcsafe.ini.

Troubleshooting

If you are having trouble configuring Rapid SQL to use Visual SourceSafe, check the system registry to determine that the COM automation portion of Visual SourceSafe has been properly installed. To check the registry, do the following:

1   From the Windows *Start* button, click *Run* and then type REGEDIT to view the system registry.

    Windows opens the Registry Editor.

2   Click the *HKEY_CLASSES_ROOT* directory.

3   Click the *SourceSafe* key.

4   Ensure that the Key contains the subkeys *CLSID* and *CurVer*.

5   If either of these keys are missing, reinstall Visual SourceSafe.

For more information, see [Version Control Integration](#).

INTEGRATING WITH MKS SOURCE INTEGRITY

Rapid SQL works with MKS Source Integrity version 7.3c.

1   After installing MKS Source Integrity, install the MKS Source Integrity Extensions.

For more information, see [Version Control Integration](#).

VERSION CONTROL CONFIGURATION

Rapid SQL lets you configure version control. You can add or remove entire projects or specific files to and from version control. The Version Control Tab of the Option Editor lets you configure version controls to your specific needs.

To use version control functions, you must have a version control system up and running on your system. You can integrate Rapid SQL with:

- Rational ClearCase 5.0 and 6.0
- [Merant Version Manager version 6.0 or later](#)
- [Microsoft Visual SourceSafe 5.0 and 6.0](#)
- [MKS Source Integrity version 7.3c](#)

Configuring Version Control During Installation

1   If you have version control installed on your machine, during the Rapid SQL installation, select the option for your version control system.

> NOTE: If you select None, Rapid SQL does not support any version control functions in Rapid SQL until you select a version control system from the application. See below.

Configuring Version Control After Installation

1   On the *File* menu, click *Options*.

Rapid SQL opens the Options Editor.

2   Select *Version Control* to open the *Version Control* Tab.

3   Select your version control system.

4   Click *OK*.

WORKING WITH PROJECTS IN VERSION CONTROL

Projects can exist independently of version control. You can add an entire project to version control at any point. When you decide to place a project under version control, Rapid SQL creates a project on the underlying version control system that has been specified.

Once a project or file has been added to version control, the features of your version control system are available directly from the Project Tab within Rapid SQL. Any changes you make to a project or file from within Rapid SQL are simultaneously changed in your version control system.

The table below describes the options and functionality on the Add to Version Control dialog box:

Option	Description
Files to be added	Lets you select files to add to Version Control.
Comment	OPTIONAL: Lets you add a comment.
Check out immediately	Select to add file and keep it checked-out.
Store only latest	Select to add the latest version.
Remove local copy	Select to add file and remove the local copy.

For more information, see:

[Completing the Add to Version Control Dialog Box](#)

COMPLETING THE ADD TO VERSION CONTROL DIALOG BOX

To add a project to version, do the following:

1   Click File and then Open Project.

    Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

    Rapid SQL opens the project on the Projects Tab.

4   On the *Project* menu, click Version Control.

    OR

    On the Project Tab, right-click a project file.

    Rapid SQL opens the Add to Version Control dialog box.

5   Click Add to Version Control.

6   Select options.

7   Click *OK* to add a project, and then click *Create.*

Rapid SQL adds the project to version control.

> **NOTE:**   Rapid SQL dims the project file icon to indicate that the project has been placed under version control.

For information on opening projects, see <u>Working with Projects</u>.

WORKING WITH FILES IN VERSION CONTROL

Rapid SQL lets you view and work with files stored in various projects in a version control system without including the files in a Rapid SQL project. The Add Version Control Files dialog box lets you create a list of files in a version control system, select the version control project, and then add it and its files to the <u>VC Files Tab</u>.

The table below describes the options and functionality on the Add Version Control Files dialog box:

Option	Description
Project	Lets you type the version control project path.
Browse	Click to open Choose project from (version control name) dialog box.
List Files	Click to view the project's files in the Version control project files tree.
File Types	Click to list the project's available files, filtered by file type.
Version control project files	List the project available files. NOTE: If the VC Files Tab is already open and contains files from the project previously selected in the Add Version Control Files dialog box, only files not already in the VC Files Tab will be listed.
Add	Click to add selected file(s) to the Files being added to version control files list box.
Add All	Click to add all file(s) to the Files being added to version control files list box.
Files being added to version control files list	Displays the files that will appear on the VC Tab.
Remove	Click to remove selected file from the Files being added to version control files list box.
Check Out	Select to automatically check out the file from version control on the VC Files Tab.
Get Latest Version	Select to automatically get latest version of the file from version control on the VC Files Tab.

For more information, see:

Completing the Add Version Control Files Dialog Box

COMPLETING THE ADD VERSION CONTROL FILES DIALOG BOX
To add files to version control, do the following:

1   Click File and then Open Version Control Files List.

    Rapid SQL opens the version control system login dialog box.

2   Type login information.

    Rapid SQL opens the Add Version Control Files dialog box.

3   In Project, type the project name, or click Browse to open the Choose project from (version control name) dialog box.

4   Select the project and click OK.

    Rapid SQL displays the files in the Version control project files box.

5   Click File Types to list the project's available files, filtered by file type.

6   In List files of type select the type of files to list.

7   In the Version control project files box, select the target files and click Add or Add All to add the files to the version control files list.

8   In Options, select Check Out or Get Latest Version.

9   Click OK.

    ```
    Rapid SQL opens the Check Out File or Get File dialog box.
    ```

10 In Comment, type a comment. If you select multiple files, the comment will apply to all the files.

11   In To, type the directory to place the file(s).

12   For SourceSafe, for advanced options, click Advanced.

13   Click OK.

    Rapid SQL adds the files in the Files being added to version control files list box to the [VC Files Tab](). If you selected the Check Out option, Rapid SQL opens the [Check Out (Files)]() dialog box. If you selected the Get Latest Version option, Rapid SQL opens the [Get from Version Control]() dialog box.

USING VERSION CONTROL

Once you add a project or file to version control, most functions found in the underlying version control system are available directly from the Rapid SQL interface, including the [VC Files Tab](). Basic version control procedures include:

- [Get Latest Version](#)

- [Check Out](#)

- [Check In](#)

- [Undo Check Out](#)

- [Open](#)

- [Show History](#)

- [Show Difference](#)

- [Version Control Properties](#)

> **NOTE:** Your Rapid SQL version control functionality depends on your underlying version control system. For more information on version control procedures, consult the documentation included with your version control system.

VC FILES TAB

The VC Files Tab displays version control files listed in the *.evc (Embarcadero version control file.) The tab displays file icons indicating their current status, for example if they are checked out to the user logged in to the source control system. The files can be opened from this list, as well as operated on to manipulate their version control properties. For example, a file can be checked out or checked into the system from the VC File Tab.

The table below describes the VC Files Tab icons:

Icon	Description
File	File is not checked out by anyone.
File with single red check mark	File is checked out non-exclusively and only by the user logged into source control.
File with a single black check mark	File is checked out non-exclusively only by a one user who is not the user logged into source control.
File with a single red check mark and a red border	File is checked out exclusively and only by the user logged in to source control.
File with a single black check mark and a red border	File is checked out exclusively and only by a user who is not the user logged in to source control.
File with two red check marks	File is checked out by multiple users, including the user logged in to source control.
File with two black check marks	File is checked out by multiple users, not including the user logged in to source control.

For more information, see:

[VC Files Tab Available Functionality](#)

[Opening the VC Files Tab](#)

[Closing the VC Files Tab](#)

VC FILES TAB AVAILABLE FUNCTIONALITY
Once the files are on the list in the VC Files Tab, you can use the various version control functionalities.

At the VC Files level, Rapid SQL lets you:

- [Add Files](#)
- [Sort](#)
- [Get Latest Version](#)
- [Check Out](#)
- [Check In](#)
- [Undo Checkout](#)
- [Remove from Version Control](#)
- [Expand All](#)
- [Collapse All](#)
- [Refresh](#)
- [Close Files List](#)

At the Project level, Rapid SQL lets you:

- [Add Files](#)
- [Delete](#)
- [Sort](#)
- [Get Latest Version](#)
- [Check Out](#)
- [Check In](#)
- [Undo Checkout](#)
- [Remove from Version Control](#)
- [Expand All](#)
- [Collapse All](#)
- [Refresh](#)

- Close Files List

At the Directory (file type) level, Rapid SQL lets you:

- Add Files
- Delete
- Sort
- Get Latest Version
- Check Out
- Check In
- Undo Checkout
- Remove from Version Control
- Expand All
- Collapse All
- Refresh
- Close Files List

At the File level, Rapid SQL lets you:

- Open
- Delete
- Get Latest Version
- Check Out
- Check In
- Undo Checkout
- Show History
- Show Differences
- Remove from Version Control
- Version Control Properties
- Refresh
- Close Files List

For more information, see:

Opening the VC Files Tab

Closing the VC Files Tab

[Working with Files in Version Control](#)

OPENING THE VC FILES TAB

1   Select File, Open Version Control File List.

    If you have files on your version control list, Rapid SQL opens the VC Files Tab. If you do not have files on your version control list, Rapid SQL opend the Add Version Control Files dialog box.

For more information, see:

[Working with Files in Version Control](#)

[Closing the VC Files Tab](#)

CLOSING THE VC FILES TAB

1   Select File, Close Version Control File List.

    Rapid SQL closes the VC Files Tab.

For more information, see:

[Working with Files in Version Control](#)

[Opening the VC Files Tab](#)

VERSION CONTROL FUNCTIONALITY - OPEN

The Open functionality opens the selected file(s) with the application registered for the type(s) of the selected file(s).

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL FUNCTIONALITY - DELETE

The Delete functionality deletes the local copy of the selected item from the tree. To remove a file from version control, see [Remove From Version Control](#).

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL FUNCTIONALITY - SORT

The Sort functionality sorts the tree items alphabetically.

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL FUNCTIONALITY - GET LATEST VERSION

The Get Latest Version functionality lets you access the latest version of a file for viewing only. The Get functionality creates a local copy of the most current version of a project file in your working folder. The file is read-only, so any modifications cannot be saved.

Before working with a project, you should perform a Get on the entire project to ensure that you are working with the latest copy of the project. You should also perform a project-level, recursive Get at intervals to ensure that you have the latest version of files, in the event that they have been altered by others working on the same project.

The following table describes the option and functionality on the Get from Version Control dialog box:

Option	Description
Files to Get	Lets you select file(s) to get latest version of.
Advanced	Click to open the Advanced Get Options dialog box.

> **TIP:** You can specify the file directory in the Version Control option of the Options Editor. For details, see Version Control Options.

Getting Latest Version of a Project

To get the latest version of project, do the following:

1   Click File and then Open Project.

    Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

    Rapid SQL opens the project on the Projects Tab.

4   Click the project or target files.

5   On the *Project* menu, click Version Control.

    OR

    On the *Project* toolbar, click Version Control.

    OR

    On the Project Tab, right-click a project or file.

    Rapid SQL opens the Get From Version Control dialog box.

6   Click *Get Latest Version*.

    Rapid SQL opens the Get From Version Control dialog box.

7   In the *Files to Get* box, click the project or files.

8   For advanced options, click Advanced.

9   Click *OK*.

Rapid SQL writes the most current version of the file to your working directory.

Getting Latest Version of a File

To get the latest version of a file, do the following:

1   On the VC Tab, right-click the target file(s) and select Get Latest Version.

Rapid SQL opens the Get From Version Control dialog box.

2   Click *Get Latest Version*.

Rapid SQL opens the Get From Version Control dialog box.

3   In the *Files to Get* box, select the file(s).

4   For advanced options, click Advanced.

5   Click *OK*.

Rapid SQL writes the most current version of the file to the VC Tab.

For more information, see <u>Using Version Control</u> .

VERSION CONTROL FUNCTIONALITY - CHECK OUT

The Check Out functionality retrieves a copy of one or more selected files and creates a writable working file copy in the working directory. You must perform a Check Out to edit any file that has been placed under version control.

You can check out a single file, multiple files at once or an entire project. Rapid SQL displays a red check mark over the file icon to indicate that the file has been checked out and is writable. This does not prevent other users from performing a Get or a Check Out on the file unless you are using the exclusive Check Out feature found in MKS Source Integrity and Merant/Intersolv PVCS.

The following table describes the options and functionality on the Check Out File(s) dialog box:

Option	Description
Files to be Checked Out	Displays list of files that are eligible for check out. A black mark indicates that another user has the file(s) checked out.
Advanced	Click to open the Advanced Check Out Options dialog box.

TIP: You can specify the file directory in the Version Control Working Directory option of the Options Editor. For details, see <u>Version Control Options</u>.

Checking Out a Project

To check-out a project, do the following:

1   Click File and then Open Project.

     Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

     Rapid SQL opens the project on the Projects Tab.

4   Select the target project.

5   On the *Project* menu, click Version Control.

     OR

     On the *Project* toolbar, click Check Out.

     OR

     On the Project Tab, right-click a project.

6   Select *Check Out*.

     Rapid SQL opens the Check Out dialog box.

7   In the *Files to Be Checked Out* box select the project.

8   For advanced options, click Advanced.

9   Click *OK*.

     Rapid SQL checks out the project or files from version control and writes the most current version of the file to your working directory.

Check Out a File

To check out a file, do the following:

1   On the VC Tab, right-click the target file(s) and select Check Out.

     Rapid SQL opens the Check Out File(s) dialog box.

2   In the *Check Out File(s) dialog* box select the files.

3   For advanced options, click Advanced.

4   Click *OK*.

     Rapid SQL checks out the file(s) from version control and writes the most current version of the file to the VC Tab.

For more information, see [Using Version Control](#).

VERSION CONTROL FUNCTIONALITY - CHECK IN

After editing your files, you must Check In the revised file in order save the changes you made to the file in a project. The Check In functionality stores the new version of the updated file in the current project. The Check In functionality is only available if you have Checked Out a file. You have the option to Check In an entire project or individual files.

> **TIP:** You can specify the file directory in the Version Control Working Directory option of the Options Editor. For details, see Version Control Options.

The table below describes the options and functionality on the Check In dialog box:

Option	Description
Files to be checked in	Lets you specify the file(s) to check in.
Keep checked out	Adds latest version(s) of file(s) to source control but keeps the file(s) checked out.
Comment	OPTIONAL: Lets you type an optional comment.

Checking In a Project

To check-in a project, do the following:

1   Click File and then Open Project.

    Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

    Rapid SQL opens the project on the Projects Tab.

4   On the *Project* menu, click Version Control, and then Check In.

    OR

    On the *Version Control* toolbar, click Check In.

    OR

    On the Project Tab, right-click a project or file, and then Check In.

    Rapid SQL opens the Check In dialog box.

5   In the *Files to Be Checked In* box click the project or files.

6   To update the version control copy but keep the project or files checked out so that you can continue working, select the *Keep Checked Out* check box.

7   To remove the file from the working directory and from the *.xml file, and from the VC Files Tab, select Remove Local Copy.

8   OPTIONAL: In the *Comment* text box type a description of the changes.

9 Click *OK*.

Rapid SQL saves the modified project into version control.

Check In a File

To check in a file, do the following:

1 On the VC Tab, right-click the target file(s) and select Check In.

Rapid SQL opens the Check In File(s) dialog box.

2 In the *Check In File(s) dialog* box select the files.

3 To update the version control copy but keep the files checked out so that you can continue working, select the *Keep Checked Out* check box.

4 To remove the file from the working directory and from the *.evc file, and from the VC Files Tab, select Remove Local Copy.

5 OPTIONAL: In the *Comment* text box type a description of the changes.

6 Click *OK*.

Rapid SQL saves the modified file(s) into version control.

For more information, see [Using Version Control](#).

VERSION CONTROL FUNCTIONALITY - UNDO CHECK OUT

If you decide that you do not want to save any revisions you have made to a checked out file, you can undo the procedure that releases the lock placed on the project or file. However, you do not have the option of deleting the local copy of the file.

> **TIP:** You can specify the file directory in the Version Control Working Directory option of the Options Editor. For details, see [Version Control Options](#).

The table below describes the options and functionality on the Undo Check Out dialog box.

Option	Description
Cancel the checkout for the following files	Lets you specify the files to undo checkout.
Advanced	Opens the Undo Check Out Advanced Options dialog box that lets you leave, replace, or apply the default action to your local copy.

Undoing Checkout for a Project

To undo a checkout, do the following:

1   Click File and then Open Project.

    Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

    Rapid SQL opens the project on the Projects Tab.

4   Click the project.

5   On the *Project* menu, click Version Control.

    OR

    On the *Project* toolbar, click Undo Check Out.

    OR

    On the Project Tab, right-click a project or file.

6   Click *Undo Check Out.*

    Rapid SQL opens the Undo Check Out dialog box.

7   In the *Cancel the check out for the following files* box of the *Undo Check Out* dialog box, click the project.

8   Click *OK.*

    Rapid SQL undoes the check out the project from version control.

    NOTE: If you Undo Check Out, you lose any changes you have made to the local copy of your project.

Undoing Checkout for a File

To undo checkout for a file, do the following:

1   On the VC Tab, right-click the target file(s) and select Undo Checkout.

    Rapid SQL opens the Undo Checkout dialog box.

2   In the *Cancel the check out for the following files* box of the *Undo Check Out* dialog box, click the project.

3   Click *OK.*

    Rapid SQL undoes the check out the file from version control.

    NOTE: If you Undo Check Out, you lose any changes you have made to the local copy of your file(s).

For more information, see [Using Version Control](#).

VERSION CONTROL FUNCTIONALITY - SHOW HISTORY

The Show History functionality lets you view the history of version control files.

Showing History for a Project

To show history, do the following:

1 Click File and then Open Project.

   Rapid SQL opens the Open Project dialog box.

2 Type the project name or select the project.

3 Click Open.

   Rapid SQL opens the project on the Projects Tab.

4 Click the target file.

5 On the *Project* menu, click Version Control, and then select Show History.

   OR

   On the Project Tab, right-click the a file, and then select Show History.

   Rapid SQL opens the History Options dialog box.

   NOTE: The History dialog box depends on your version control system.

Showing History for a File

To show history, do the following:

1 On the VC Tab, right-click the target file(s) and select Show History.

   Rapid SQL opens the History Options dialog box.

   NOTE: The History dialog box depends on your version control system.

For more information, see [Using Version Control](#).

VERSION CONTROL FUNCTIONALITY - SHOW DIFFERENCES

The Show Differences functionality lets you view any differences between the current files in your working folder and the master files in the version control database. You cannot make changes to the files from this dialog box because it is used for display purposes only.

Viewing Project Differences

To view project differences, do the following:

1   Click File and then Open Project.

     Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

     Rapid SQL opens the project on the Projects Tab.

4   Click the target file.

5   On the *Project* menu, click Version Control.

     OR

     On the Project Tab, right-click a file.

6   Click *Show Differences.*

     Rapid SQL opens the Differences dialog box.

     If the file in your working directory is the same as the one in the project, a message tells you they are identical. If there are differences, the Differences dialog box from your version control system opens and displays the two versions of the file side-by-side, highlighting any differences.

7   In the *Differences* dialog box, you can maneuver through the files by using the Up and Down arrows.

8   To set *Diff Options*, click the *Options* button.

     NOTE: The Differences dialog box and Options depend on your version control system.

Showing History for a File

To show history, do the following:

1   On the VC Tab, right-click the target file(s) and select Show History.

     Rapid SQL opens the Difference Options dialog box.

2   In the *Difference Options* dialog box, you can maneuver through the files by using the Up and Down arrows.

     NOTE: The Difference Options dialog box and options depend on your version control system.

For more information, see [Using Version Control](#).

VERSION CONTROL FUNCTIONALITY - REMOVE FROM VERSION CONTROL

The Remove from Version Control functionality lets you remove entire projects or files from version control.

> **NOTE:** This functionality does not destroy the file permanently from the source control system or remove the local copies of the file(s).

Removing a Project from Version Control

To remove a project from Version Control, do the following:

1   Click File and then Open Project.

Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

Rapid SQL opens the project on the Projects Tab.

4   On the *Project* menu, click Version Control, and then click Remove from Version Control.

OR

On the Project Tab, right-click a project file, and then click Remove from Version Control.

Rapid SQL opens the Remove File(s) dialog box.

5   To permanently destroy the project select the check box, click *OK*.

Rapid SQL removes the project from version control. The project remains active in Rapid SQL.

Removing a File from Version Control

The table below describes the options and functionality of the Remove Files dialog box:

Option	Description
Files to be removed	Lets you specify the files to be removed from version control. NOTE: Does not delete the file(s) in version control.

To remove a file from Version Control, do the following:

1   On the VC Tab, right-click the target file(s) and select Remove from Version Control.

Rapid SQL opens the Remove File(s) dialog box.

2   Specify the file(s).

3   Click *OK*.

Rapid SQL removes the file(s) from version control. The project remains active in Rapid SQL.

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL PROPERTIES

The Version Control File Properties dialog box displays general information and check out status, links, and paths.

Viewing Version Control Properties for a Project

To view the version control properties, do the following:

1   Click File and then Open Project.

    Rapid SQL opens the Open Project dialog box.

2   Type the project name or select the project.

3   Click Open.

    Rapid SQL opens the project on the Projects Tab.

4   Click the target file.

5   On the *Project* menu, click Version Control, and then Version Control Properties.

    OR

    On the Project Tab, right-click the a file, and then select Version Control Properties.

    Rapid SQL opens the Version Control Properties dialog box.

6   Review properties.

7   Click Close.

Viewing Version Control Properties for a File

To view the version control properties, do the following:

1   On the VC Tab, right-click the target file(s) and select Version Control Properties.

    Rapid SQL opens the Version Control Properties dialog box.

2   Review properties.

3   Click Close.

For more information, see [Using Version Control](#).

VERSION CONTROL FUNCTIONALITY - EXPAND ALL

The Expand All functionality expands all tree items under the selected item(s).

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL FUNCTIONALITY - COLLAPSE ALL

The Collapse All functionality collapses all tree items under a selected item(s).

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL FUNCTIONALITY - REFRESH

The Refresh functionality obtains the current version control status for the file(s).

For more information, see [Working with Files in Version Control](#).

VERSION CONTROL FUNCTIONALITY - CLOSE FILES LIST

The Close Files List functionality closes the list of files.

For more information, see [Working with Files in Version Control](#).

TOOLS

Rapid SQL incorporates a number of powerful tools to help you create, edit and manage your development environment. You can use Tools to:

- Conduct extensive database searches across multiple databases.

- Execute scripts or files across multiple databases.

- Schedule tasks.

- Identify differences in files or objects.

- Graphically build complex queries.

- Auto-generate complete procedures and packages.

- Code and test basic macros.

- Administer your ODBC data sources.

Tools is divided into sections. The table below describes each section:

Section	Description
Find in Files	This section describes the Find in Files dialog box that lets you find a phrase or character in your files.
Database Search	This section describes the powerful database search utility that helps you to find instances of a string across multiple databases.
Script Execution Facility	This section describes the Rapid SQL Script Execution Facility, a stand-alone utility that establishes multiple threads and database connections letting you simultaneously execute SQL statements against multiple Oracle, Sybase Adaptive Server, Microsoft SQL Server, and IBM DB2 LUW for Open Systems datasources.
File Execution Facility	This section describes the Rapid SQL File Execution Facility, a stand-alone utility that establishes multiple threads and database connections that lets you execute parallel queries and ANSI SQL files against multiple, cross-platform datasources.
Scheduling	The Rapid SQL scheduling programs and utilities let you schedule and execute jobs on local datasources anytime.
Visual Difference	Rapid SQL lets you compare two files or database objects. Using the Visual Difference Utility, you can easily synchronize and analyze database objects or files across multiple database platforms.
Query Builder	This section describes Query Builder, a tool that lets you construct, structure, and manipulate up to five different types of queries simultaneously.

Section	Description
Data Editor	This section describes the Data Editor to edit your tables in real-time. The Data Editor supports all editable datatypes and is an alternative way to add, edit, or delete data from your tables.
Code Generation Facility	This section describes the Code Generation Facility that offers a quick way to generate DML statements for tables and views.
Import Data	This section describes the Import Data Wizard that lets you create insert statements based on external files including Excel spreadsheets and text files.
Embarcadero Products	The Tools menu lists all installed Embarcadero Technologies products. This lets you toggle to or start another Embarcadero product.
Code Workbench	This section describes the Code Workbench that lets you enable table column look-up and population in the ISQL window, define auto replacement expressions that can be used to quickly insert commonly used blocks of SQL syntax or commands in any open window and to import and export Code Workbench specific settings for client sharing purposes.

FIND IN FILES

The Find in Files dialog box lets you find a phrase or character in your files.

Completing the Find in Files Dialog Box

1   On the *Tools* menu, click *Find in Files.*

    OR

    On the *Tools* toolbar, click Find in Files.

    Rapid SQL opens the Find in Files dialog box.

The table below describes the options and functionality on the Find in Files dialog box.:

Option	Description
Find what	Specifies the character(s) or phrase you want to find. Use the browse arrow button next to the textbox to choose options from a pop-up list.
In files/file types	Specifies the files in which to search for the character(s) or phrase. Either enter the filename(s) in the drop-down box, or click the arrow to choose a file type.
In folder	Specifies the directory where the file(s) is located. Click the browse button to view your Windows Explorer.
Match whole word only	Specifies the application to find only the entire phrase.
Match case	Specifies the application to find only the specified phrase in the case you have entered.
Regular Expression	Tells the application whether the specified character(s) is a regular expression.
Look in subfolders	Specifies the application to search the file(s) any folders located within the specified folder.
Output to Pane 2	Specifies the application to display the results in another window.

NOTE: You can also use the Find feature to locate a phrase or character in an ISQL window.

DATABASE SEARCH

The powerful [database search utility](#) helps you to find instances of a string across multiple databases.

Starting the Database Search Wizard

1 On the *Tools* menu, click *Database Search.*

    OR

    On the *Tools* toolbar, click Database Search.

    Rapid SQL opens the [first panel of the Database Search Wizard](#).

DATABASE SEARCH WIZARD - PANEL 1

The first panel of the Database Search Wizard lets you specify the owner.

The table below describes the options and functionality on the first panel of the Database Search Wizard:

Option	Description
Select the owner(s) whose objects you would like to search	Lets you expand the nodes, select the target owner, and then click the right arrow button to include the target owner.

1   Click Next.

Rapid SQL opens the next panel of the wizard.

DATABASE SEARCH WIZARD - PANEL 2

The second panel of the Database Search Wizard lets you specify the search criteria.

The table below describes the options and functionality on the second panel of the Database Search Wizard:

Option	Description
Search Database For	Lets you enter the search string.
	Strings can also be searched for using DB2-standard wildcards. And for multiple string searches, separate each string with a vertical bar (for example, DEPARTMENTS|wage_cap|status). When searching for strings that already contain vertical bars, enclose each string in double quotation marks.
Match Case	Select Yes tot make the search case sensitive.
	NOTE: IBM DB2 LUW for OS/390 and Microsoft SQL Server 7.0 or before searches are always case insensitive.
Search DDL of these Objects	In the grid, select the target object check boxes.
	NOTE: Event Monitors are available for IBM DB2 LUW for Linux, Unix, and Windows only.

1   Click Execute to start the operation.

Rapid SQL displays a progress dialog box while the search runs. When the search completes, Rapid SQL opens the Database Search Window.

DATABASE SEARCH RESULTS

Rapid SQL displays Database Search operation results in a Database Search Window, listing all of the objects containing the search string in the left pane. You can browse instances of the search string by selecting different objects in the tree. The DDL of the objects displays in the right pane and the search string is highlighted.

The table below describes the buttons on the Database Search Window toolbar:

Button	Description
Search	Opens the first panel of the <u>Database Search Wizard</u>.
Criteria	Opens the Search Criteria dialog box.
Open	Opens the editor for the target object.
Extract	Lets you extract the target object.
Print	Lets you print the target object SQL.

SCRIPT EXECUTION FACILITY

Rapid SQL's Script Execution Facility is a stand-alone utility that establishes multiple threads and database connections letting you simultaneously execute SQL statements against multiple Oracle, Sybase Adaptive Server, Microsoft SQL Server, and IBM DB2 LUW for Linux, Unix, and Windows datasources. After completing a scheduled job, Rapid SQL generates a results report that lists errors, verifies script execution, and details the output of the job. The Script Execution Facility also works in conjunction with Rapid SQL's scheduling facilities, letting you schedule script execution jobs. When used in conjunction with a scheduler, Rapid SQL can automatically send the results report to any e-mail or network recipients. The Script Execution Facility is a tabbed dialog box where you set the parameters and options for the script execution. In the Script Execution Facility dialog box you can:

- Type or paste the target SQL script.

- Specify the datasources against which to execute the script.

- Specify the output mode for the results report.

- Open one of the Rapid SQL scheduling programs to schedule the script execution.

- Specify execution notification e-mail and Net Send addresses.

For more information, see <u>Completing the Script/File Execution Facility</u>.

FILE EXECUTION FACILITY

Rapid SQL's File Execution Facility is a stand-alone utility that establishes multiple threads and database connections that lets you execute parallel queries and ANSI SQL files against multiple, cross-platform datasources. The Script Execution Facility also works in conjunction with Rapid SQL's scheduling facilities, letting you schedule script execution jobs. After completing a scheduled job, Rapid SQL generates a results report that lists errors, verifies execution, and details the output of the job. When used in conjunction with a scheduler, Rapid SQL can automatically send the results report to any e-mail or network recipients. The File Execution Facility is a tabbed dialog box where you set the parameters and options for the file execution. In the File Execution Facility dialog box you can:

- Specify the target files or ANSI SQL scripts.

- Specify the datasources against which to execute the files.

- Specify the output mode for the results report.

- Open one of the Rapid SQL scheduling programs to schedule the file execution.

- Specify execution notification e-mail and Net Send addresses.

For more information, see Completing the Script/File Execution Facility.

COMPLETING THE SCRIPT/FILE EXECUTION FACILITY

Rapid SQL lets you run parallel queries against multiple datasources with the File Execution Facility.

1 On the *Tools* menu, click *Script Execution Facility or File Execution Facility.*

   OR

   On the *Tools* toolbar, click *Script Execution Facility or File Execution Facility.*

   Rapid SQL opens the Script or File Execution Facility dialog box.

2 Complete the Script Tab (Script Execution Facility)

3 Complete the Files Tab (File Execution Facility)

4 Complete the Target Tab.

5 Complete the Output Tab.

6 Complete the Notify Tab.

For more information, see:

File Execution Facility

Script Execution Facility

SCRIPT EXECUTION FACILITY - SCRIPT TAB

The table below describes the options and functionality on the Script Tab of the File/Script Execution Facility:

Option	Description
Script box	Lets you type or paste a script.

For more information, see:

File Execution Facility

Script Execution Facility

Completing the Script/File Execution Facility

FILE EXECUTION FACILITY - FILES TAB

The table below describes the options and functionality on the Files Tab of the File Execution Facility:

Option	Description
Show Full File Paths	Select to display the full path. Deselect to display only the file name.
File Name	Displays the file names.
Add	Click to open the Select Files dialog box.
Remove	Click to remove the selected file.
View	Opens the View File dialog box.
Up	Click to move the selected file up in the list.
Down	Click to move the selected file down in the list.

For more information, see:

File Execution Facility

Script Execution Facility

Completing the Script/File Execution Facility

FILE/SCRIPT EXECUTION FACILITY - TARGET TAB

The table below describes the options and functionality on the Target Tab of the File/Script Execution Facility:

Option	Description
Select the Target Datasource(s) to Execute the Script Against	Only Connected Datasources - Displays only datasources that are currently connected in the Datasource grid.
	All DBMS Types - Displays all DBMS types in the Datasource grid.
Datasource grid	Displays the target datasource(s) to execute the script/file against. Select a datasource name. If the datasource has multiple databases, type in a database in the Database box.

For more information, see:

File Execution Facility

Script Execution Facility

Completing the Script/File Execution Facility

FILE/SCRIPT EXECUTION FACILITY - OUTPUT TAB

The table below describes the options and functionality on the Output Tab of the File/Script Execution Facility:

Option	Description
Graphical Output	If selected, specifies a graphical output.
File Output	If selected, specifies a file output.
	Directory - Type or browse to enter the full path and directory name in which you want to place the output file.
	File Type - Specifies a file type.
	Include column titles when saving - If selected, lets you save column titles.
	Open files with registered applications - If selected, opens files with registered applications.

For more information, see:

File Execution Facility

Script Execution Facility

Completing the Script/File Execution Facility

FILE/SCRIPT EXECUTION FACILITY - NOTIFY TAB

The table below describes the options and functionality on the Notify Tab of the File/Script Execution Facility:

Option	Description
Job Description	Lets you enter a job description. This description will be the subject of the notification E-mail.
E-mail address	Lets you enter E-mail addresses. Separate each E-mail address with a semicolon (;).
Net Send User Names	Lets you enter net send user names. Separate each name with a semicolon (;).

For more information, see:

File Execution Facility

Script Execution Facility

Completing the Script/File Execution Facility

SCHEDULING

The Rapid SQL scheduling programs and utilities let database administrators schedule and execute jobs on local datasources 24-hours-a-day, 7-days-a-week.

Rapid SQL offers the following programs and utilities to let you schedule routine tasks and jobs:

- Microsoft Task Scheduler

- ETSQLX Command Line Utility

MICROSOFT TASK SCHEDULER

Rapid SQL lets you use the Microsoft Task Scheduler to schedule jobs. The Microsoft Task Scheduler is included with various Microsoft applications. If you do not have this program on your system, the first time you attempt to schedule a job, Rapid SQL provides you with a link to the Microsoft Web site where you can download the Microsoft Task Scheduler at no cost.

> **NOTE:** Rapid SQL's ETSQLX command line utility runs a scheduled job even if Rapid SQL is not running. For more information, see ETSQLX Command Line Utility.

To open the Microsoft Job Scheduler from within Rapid SQL

1   On the **Tools** menu, click **Scheduler**.

2   Use the associated online Help for assistance with using this Microsoft utility.

ETSQLX COMMAND LINE UTILITY

Rapid SQL's ETSQLX command line utility, is a multi threaded, cross-platform, SQL scripting engine. You can use ETSQLX in conjunction with the Microsoft Task Scheduler to schedule and automate routine jobs. ETSQLX creates batch files (with the extension.cfg) containing commands to execute automated and scheduled jobs. ETSQLX creates a directory, CFG, in which it stores the.cfg files. You can run.cfg files directly from the command line.

> **NOTE:** ETSQLX supports.csv,.tab,.htm, and.html formats for result reports attachments.

VISUAL DIFFERENCE

Rapid SQL lets you compare two files or database objects. Using the Visual Difference dialog box, you can easily synchronize and analyze database objects or files across multiple database platforms. The files are displayed side by side in the Visual Difference dialog box. The Visual Difference Utility highlights any differences between two files. Viewing differences between objects and files helps you negotiate between the different phases of development as well as providing a visual aid to rapidly changing and evolving production environments.

> **NOTE:** Because contents of the Visual Difference dialog box are read-only, you will not be able to modify your files or objects directly from this dialog box.

The Visual Difference dialog box is composed of two panes; the left pane displays your source object or file and the right pane shows your target object or file. The Visual Difference dialog box also contains its own toolbar which lets you:

- Search

- Navigate differences

- Set options

- Print

Opening the Visual Difference Dialog Box

1   On the *Tools* menu, click *Visual Diff.*

    OR

    On the *Tools* toolbar, click Visual Diff.

    Rapid SQL opens the Visual Difference dialog box.

For more information, see:

Comparing Files

Comparing Database Objects

Navigating in the Visual Difference Dialog Box

Printing a Pane of the Visual Difference Dialog Box

Searching in the Visual Difference Dialog Box

Setting Options in the Visual Difference Dialog Box

COMPARING FILES

You can compare two files side-by-side in the Visual Difference dialog box. The file you want to compare is called the Source. The file you want to compare the first file to is the Target.

Comparing Items

1   On the *Tools* menu, click *Visual Diff.*

    OR

    On the *Tools* toolbar, click Visual Diff.

    Rapid SQL opens the Visual Difference dialog box.

2   On the *Visual Difference* toolbar, click the *Source* icon or click the Down arrow next to the *Source* icon and then click *File.*

    Rapid SQL opens the Select the 1st File to Compare dialog box.

3   Click the file that you want to be the Source file.

4   On the *Visual Difference* toolbar, click the Target icon or click the Down arrow next to the Target icon and then click *File.*

    Rapid SQL opens the Select the 2nd File to Compare dialog box.

    NOTE: The Visual Difference Utility highlights all differences between the two files.

For more information, see [Visual Difference Utility](#).

COMPARING DATABASE OBJECTS

The schema of database objects is automatically extracted so you can view the underlying differences between object and perform a side-by-side comparison in the Visual Difference Dialog.

Comparing Database Objects

1   On the *Tools* menu, click *Visual Diff.*

    OR

    On the *Tools* toolbar, click Visual Diff.

    Rapid SQL opens the Visual Difference dialog box.

2   On the *Visual Difference* toolbar, click the Down arrow next to the *Source* icon and then click *Database Object.*

    Rapid SQL opens the Select the 1st Database Object to Compare dialog box.

3   Click the datasource and then click *OK* to connect.

4   Navigate through the datasource tree and double-click the database object that you want to be the *Source.*

5   On the *Visual Difference* toolbar, click the Down arrow next to the *Target* icon and then click *Database Object.*

Rapid SQL opens the Select the 2nd Database Object to Compare dialog box.

> **NOTE:**   The Visual Difference Utility highlights all differences between the two database objects.

For more information, see <u>Visual Difference Utility</u>.

NAVIGATING IN THE VISUAL DIFFERENCE DIALOG BOX

You can navigate through the Visual Difference dialog box using the up and down arrow buttons. You can move back and forth between highlighted differences in your compared files or database objects.

Going To the Next Difference

1   From the *Visual Difference* dialog box, click down arrow to go to the next difference.

Going To the Previous Difference

1   From the *Visual Difference* dialog box, click up arrow to go to the next difference.

For more information, see <u>Visual Difference Utility</u>.

PRINTING A PANE OF THE VISUAL DIFFERENCE DIALOG BOX

You can print each pane of the Visual Difference dialog box.

Printing a Pane of the Visual Difference Dialog Box

1   Position your cursor inside the pane you want to print.

2   Click the *Print* icon on the *Visual Difference* toolbar.

Rapid SQL opens the Print Setup dialog box.

3   Click *OK* to print the pane.

> **NOTE:**   You can only print one pane of the Visual Difference dialog box at a time.

For more information, see <u>Visual Difference Utility</u>.

SEARCHING IN THE VISUAL DIFFERENCE DIALOG BOX

The Visual Difference dialog box lets you search for text in your files or database objects.

Searching for Text

1 Place your cursor inside the pane you want to search.

2 Click the *Find* icon on the *Visual Difference* toolbar.

   Rapid SQL opens the Find dialog box.

3 In the *Find What* box, enter the search string.

4 To match whole words only, select the *Match whole word only* check box.

5 To make the search case sensitive, select the *Match case* check box.

6 Click *Find Next* to find the next occurrence of your search string. You can also click the *Find Next* icon on the *Visual Difference* toolbar to search for the next occurrence at a later time.

For more information, see [Visual Difference Utility](#).

SETTING OPTIONS IN THE VISUAL DIFFERENCE DIALOG BOX

The Visual Difference dialog box lets you set display and comparison options to help you customize the dialog box to view differences in a comprehensive manner.

Setting Options

1 Click the *Options* icon on the *Visual Difference* toolbar.

   Rapid SQL opens the Visual Diff Options dialog box.

The following table describes the check box options:

Option	Description	Default
Display Line Numbers	Indicates that line numbers should appear in the Visual Diff dialog box.	Off
Display Hidden Characters	Indicates that hidden characters (nonprintable) should be displayed.	Off
Ignore White Space	Indicates that White Space (such as spaces, carriage returns, line feeds, and tabs) should be ignored. If this option is set on, text will be considered equivalent regardless of white space, otherwise the text will be shown as being different.	On
Ignore Hidden Characters	Indicates that hidden characters (nonprintable) should be excluded.	Off
Ignore Case	Indicates that case should not be a differentiating factor.	On

2 Click *OK*.

   Rapid SQL accepts the options.

For more information, see [Visual Difference Utility](#).

QUERY BUILDER

Query Builder is a database productivity tool that lets you construct, structure, and manipulate up to five different types of queries simultaneously. It includes a separate graphical interface that opens within your current workspace. You can run Query Builder against all Embarcadero Technologies supported database platforms.

Query Builder displays the interconnections of your queries as you work. The powerful visual components of Query Builder let you see your query grow and change to fit your needs. Query Builder eases the task of drawing data from tables by automatically creating correct SQL code as you build a statement. You can use Query Builder to create and execute SELECT statements for tables and views. You can also test queries, and easily adjust your information, before you save. Query Builder does not rely on knowledge of the underlying SQL code.

You can save and reopen queries in Query Builder. Query Builder automatically checks for changes in your tables or columns between the time you save the query and the time you reopen it.

The table below describes the types of queries available in Query Builder:

Query Type	Description
SELECT	Create, manipulate and execute SELECT Statements for tables and views.
INSERT	Create and manipulate INSERT Statements for tables.
UPDATE	Create and manipulate UPDATE Statements for tables.
DELETE	Create and manipulate DELETE Statements for tables.
CREATE VIEW	Create and manipulate CREATE VIEW Statements for tables and views.

> **NOTE:** You can execute SELECT statements directly from Query Builder. INSERT, UPDATE, DELETE, and CREATE VIEW statements must be moved to an ISQL Editor for execution.

For more information, see:

[Query Builder Design](#)

[Using Query Builder](#)

QUERY BUILDER DESIGN

Query Builder lets you build DML statements using an intuitive, graphical interface. It offers you a powerful and flexible way to quickly create complex statements without sacrificing time manipulating SQL code. Query Builder lets you add tables or columns, create joins, and change statements within the graphic display without leaving Rapid SQL. It also lets you have multiple sessions working at the same time.

Query Builder includes many different features to assist you in building and manipulating your query:

- Query Builder Statement Properties

- Workspace Windows

- Query Builder Explorer

- Tool Bar

- SQL Diagram Pane

- SQL Statement Pane

For more information, see Build Query.

WORKSPACE WINDOWS

The Workspace Windows provide a comprehensive view of your data structure and query. The table below describes the Workspace Windows:

Pane	Description
Query Builder Explorer Window	Includes two tabs that display selected object details: Tables/Views DML
SQL Diagram Pane	Displays tables or views included in the current query.
SQL Statement Pane	Displays the SQL code, and when appropriate, a Results Tab.

For more information, see Build Query.

QUERY BUILDER EXPLORER WINDOW

The Query Builder Explorer is a separate tree that exposes all the tables and views in your target database. It also displays your current query structure. The Query Builder Explorer includes two tabs that display information about the selected objects:

- Tables/Views

- DML

Tables/Views Tab

The Tables/View Tab displays information about the selected tables or views. You can use the drop-down lists to change your table or view, and when appropriate, the owner. The table below describes each drop-down list on the Tables/Views Tab:

List	Description
First	Displays all databases for a target Microsoft SQL Server or Sybase ASE.
Second	Displays all valid owners.

> **NOTE:** To change your current database, select the new database in the Explorer, and then open another Query Builder session. Query Builder prompts you to save the current session prior to opening a new session.

DML Tab

The DML Tab displays all the basic elements of a query statement in the SQL Statement Tree. You can access any element of the current statement display and perform SQL editing from the SQL Statement Tree.

For more information, see:

Creating a Clause Using the SQL Statement Tree

Build Query

SQL DIAGRAM PANE

The SQL Diagram Pane displays tables, views, and joins included in the current query. You can manipulate elements of your query, using the mouse functionality, in the SQL Diagram Pane. From the SQL Diagram Pane you can:

- Add and Remove tables and views.

- Create and delete joins.

- Add and Subtract columns.

All changes in the SQL diagram reflect in correct SQL code in the SQL Statement Pane.

For more information, see Build Query.

SQL STATEMENT PANE

The SQL Statement Pane displays the current query SQL code. When you run a query, Query Builder displays results to your query in the SQL Statement Pane. The SQL Statement Pane is divided into two tabs:

- SQL Tab

- [Results Tab](#)

SQL Tab

The SQL Tab displays the query in progress. It displays each element of your query as you build it, and updates as you do edits such as selecting or deselecting columns, adding clauses, and creating joins. Rapid SQL lets you open the current statement directly into an ISQL editor or copy it to the clipboard for later use.

Results Tab

The Results Tab displays the results of your executed query in the Results grid. To edit data, use the [Data Editor](#) application from Query Builder. When you begin building a new query, the tab title changes to Old Results until you execute the new query.

For more information, see [Build Query](#).

QUERY BUILDER TOOL BAR

The Query Builder tool bar lets you access commonly used features.

The table below describes Query Builder tool bar functionality:

Name	Function
Copy	Copies the current SQL statement to the clipboard.
Statement Box	Displays the type of statement currently on display in the main workspace window.
Stop Execution	Stops an executing query.
Execute	Executes the current SELECT or CREATE VIEW statement. If the button is not available, the statement is not executable.
New	Adjusts to the target node in the Query Builder Explorer window.
Edit	Displays, on the DML Tab, the ORDER BY or GROUP BY dialog boxes when target node is selected.
Delete	Deletes the target object.
Auto Layout	Resets the main workspace to the auto layout mode.
Auto Join	Finds and joins, automatically, like items by name.
Statement Check	Checks query syntax.
Edit Data	Opens Data Editor.
Close	Closes the current query.

> **NOTE:** Query Builder adjusts tool availability to match the current query functionality.

For more information, see [Build Query](#).

TABLES AND VIEWS SHORTCUT MENUS

Query Builder includes a shortcut menu that lets you manipulate a table or view. The table below describes the table shortcut options:

Option	Description
Delete	Removes the table from the SQL Diagram Pane, and the SQL Statement.
Title Font	Specifies the table title font for this diagram.
Column Font	Specifies the column font for this diagram.
Background Color	Specifies the table background color for this diagram.
Select Star	Selects every column in the table.
Select None	Deselects every column in the table.
Bring to Front	Moves the table to the top layer of the diagram.
Properties	Opens the Table Properties dialog box.

> **NOTE:** Your selection applies to all selected tables and views.

For more information, see Build Query.

TABLES AND VIEWS KEYBOARD COMMANDS

Query Builder provides a number of keyboard shortcuts that let you quickly construct queries. The table below describes the keyboard commands:

Keyboard Command	Location	Description
ESCAPE	SQL Diagram Pane	Breaks off a join.
F5	Query Builder	Refreshes screen and runs Schema Change Detection. In a CREATE VIEW, this key adds the new view to the Table Tree Pane.
CTRL A	SQL Diagram Pane	Selects all tables and joins in the current diagram.
F1	Query builder and application	Obtains context sensitive Help.

For more information, see Build Query.

QUERY BUILDER DIALOG BOXES

Query Builder includes a number of dialog boxes to assist you in building and customizing your query.

Dialog Box	Description
Statement Properties	Specifies general properties in an individual Query Builder session.
Table Properties	Specifies column selection and alias names for a table or view.
Column Properties	Specifies column functionality within SELECT and CREATE VIEW statements.

For more information, see Build Query.

STATEMENT PROPERTIES

The Statement Properties dialog box lets you customize properties in an individual Query Builder session. For example, you can set options to limit the number of rows returned in a query to save execution time, or turn off the auto join function to manually control all joins for an individual query. The table below describes the options and functionality of the Statement Properties dialog box.

Interface Element	Option	Description	Default
Code Generation	Generate Use Database statement	Adds a line of SQL code indicating which database or instance is used in the statement.	Selected
	Generate owner names	Adds a line of SQL code showing the table owner name as part of the query.	Selected
	Include Row Count limits	Includes the output row limit set in the Execution settings.	Selected
Execution	Max Row Count in Results Set	Sets row count limits to build and check a query without congesting server processes when a query executes.	1000 rows
General	Show Column Data types in Query Diagram	Lets Query Builder reveal the data type in each column for tables in the SQL Diagram Pane.	Not selected
	Confirm on Item delete	Lets Query Builder open a Confirm Delete dialog box when an item is deleted. NOTE: Clearing this function can result in unexpected changes to your query diagram and statement.	Selected
	Auto populate views	Lets Query Builder automatically populate views.	Not selected
Auto Join	Require Indexes	Joins indexed columns automatically, and requires indexed columns for joins.	Selected
	Require same data type	Automatically joins columns with the same data type.	Selected
Syntax Checker	Automatic Syntax Check	Lets Query Builder check syntax every time an execute statement, refresh or copy statement begins.	Selected
	Run Automatically	Lets Query Builder automatically detect like names and data types and create joins for multiple tables.	Selected
Display	Columns Font	Lets you set the font, font style, size, and color of column fonts.	Available
	Title Font	Lets you set the font, font style, size, and color of table/view title fonts.	Available
	Table Color	Lets you set the background color of your tables in the SQL Diagram Pane.	Available

NOTE: If you set options while Query Builder is running, Rapid SQL displays a warning indicating that you are about to change options or properties.

For more information, see [Completing the Statement Properties Dialog Box](#).

COMPLETING THE STATEMENT PROPERTIES DIALOG BOX

To complete the Statement Properties dialog box, do the following:

1   On the Query Builder menu, click Statement Properties.

    OR

    In the SQL Diagram Pane, right-click, and then click Statement Properties.

2   Set options.

3   Click OK.

    Query Builder saves the options.

For more information, see [Build Query]().

TABLE PROPERTIES

The Tables Properties dialog box lets you set parameters for tables or views in your SQL Diagram. The table below describes the options and functionality on the Table Properties dialog box.

Option	Description
Table Alias	Creates an alias name for your table.
Show Datatypes	Shows or hides the datatype for every column in the target table.
Displayed Columns	Displays columns visible in the SQL Diagram.
Hidden Columns	Displays columns hidden in the SQL Diagram.
Hide All	Moves all non selected columns in the table to the Hidden Columns window.
Display All	Moves all columns in the table to the Displayed Columns window.
Right Arrow	Moves a target file from Displayed Columns to Hidden Columns.
Left Arrow	Moves a target file from Hidden Columns to Displayed Columns.

For more information, see [Completing the Table Properties Dialog Box]().

COMPLETING THE TABLE PROPERTIES DIALOG BOX

To complete the Table Properties dialog box, do the following:

1   Double click the target table or view title bar.

    OR

    Right-click target table or view, and then click Properties.

2   If you only want to hide or display columns in your table, click the arrow button on the table title bar.

3   You can also edit view properties from the Table Properties dialog box.

4   Click OK.

Query Builder saves the changes.

For more information, see Build Query.

COLUMN PROPERTIES

The Column Properties dialog box lets you set properties for individual columns in your SELECT or CREATE VIEW statements. You can set aggregate functions and create an alias for an individual column.

The Column Properties dialog box is not available for INSERT, UPDATE or DELETE statements.

The table below describes the options and functions Columns Properties dialog box:

Interface Element	Description
Tables/Views	Displays all tables and views in the SQL Diagram Pane.
Aggregate	Specifies aggregate options for the target column.
	AVG - An average is taken for a column with an int or numeric datatype.
	COUNT - Returns the number of rows which contain data for the target column.
	MAX - Returns the highest number in a row in the column.
	MIN - Returns the lowest number in a row in the column.
	SUM - Returns the sum of the target column in all rows which contain data. This function is only operable on int or numeric datatypes.
Alias	Displays the alias name for the target column. Lets you type the name of the alias.
	NOTE: Query Builder displays the results of an aggregate column without a column name unless you create an alias for that column.
Available Columns	Displays all available columns in the target table or view.
Selected Columns	Displays all selected columns in the target table or view. To create an aggregate function or alias for a different column, select target column, select an aggregate function, and then type the name of the alias.
Select All	Moves all columns in the Available Columns box to the Selected Columns box.
Clear All	Moves all columns in the Selected Columns box to the Available Columns box.

Interface Element	Description
Right Arrow	Moves target column in the Available Columns box to the Selected Columns box.
Left Arrow	Moves target column in the Selected Columns box to the Available Columns box.
Select List Statement	Displays the current query.

Completing the Column Properties Dialog Box

To complete the Column Properties dialog box, do the following:

1   On the SQL Statement Tree, double-click target column.

2   Select options.

3   Click OK.

For more information, see [Build Query](#).

JOIN PROPERTIES

Query Builder lets you edit joins with the Join editor. You can edit join parameters in a SELECT, UPDATE, DELETE, and CREATE VIEW Statement.

The table below describes the options and functionality on the Join dialog box.

Option	Description
From Table Column	The primary column in the join.
To Table Column	The secondary column in the join.
Select the join relation operator	Click the target join operator. If it is not equals, the operator displays on the join in the SQL Diagram Pane.
Join Type: Inner	Click to make the join an inner join. Aggregates are only available for inner joins.
Join Type: Left Outer	Click to make the join a left outer join.
Join Type: Right Outer	Click to make the join a right outer join.

> **NOTE:** For IBM DB2 LUW for Linux, Unix, and Windows servers, there is an additional join object in the SQL Statement Tree. The Join On node displays join relations between columns in IBM DB2 LUW for Linux, Unix, and Windows tables and views.

Completing the Join Dialog Box

To complete the Join dialog box, do the following:

1   In the *SQL Diagram Pane*, right-click the target join, and then click Properties.

    OR

    In the *SQL Diagram Pane*, double-click the target join.

    OR

    On the *SQL Statement Tree*, expand the *Where* and *And* nodes, and then double-click the target join.

2   Select options.

3   Click OK.

For more information, see Build Query.

USING QUERY BUILDER

Query Builder provides a visual display of your queries as you construct them. You can run Query Builder against any registered datasource in Rapid SQL. Query Builder lets you build five separate types of queries simultaneously:

- SELECT
- INSERT
- UPDATE
- DELETE
- CREATE VIEW

You can execute a SELECT statement from Query Builder. To execute an INSERT, UPDATE, DELETE, and CREATE VIEW statement, copy them to an ISQL Editor. You can also copy the statements to the clipboard for later use in the ISQL Editor. Query Builder also lets you save a statement at any time so that you can open them later for editing or execution.

Rapid SQL lets you open Query Builder with multiple tables or views with the same or different owners. If you open tables or views with different owners, Query Builder displays "All Owners" in the Owner drop-down list. You can start multiple Query Builder sessions from Rapid SQL. You can use different tables and views for each query. You can also toggle back and forth among each of the queries.

You can save and reopen queries in Query Builder. Query Builder automatically checks for changes in your database or instance between the time you save the query and the time you reopen it with the Schema Change detection component.

Query Builder is integrated with Data Editor so you can edit data in real time and then continue to build your query with the new information embedded in the query.

USING QUERY BUILDER

To use Query Builder, do the following:

- Select an instance or database

- Select a statement.

- Select a table(s) or view(s).

- Select a column or columns.

> **NOTE:** You can start Query Builder directly from a table or view which automatically selects the instance or database which contains that table or view.

1 On the Tools menu, click Query Builder.

    OR

    On the Datasource Explorer, expand the Database or Schema node, click Tables, and then on the Command Menu, click Build Query.

    OR

    On the Tools tool bar, click Build Query.

    OR

    On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

    OR

    On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

    Rapid SQL opens Query Builder.

SELECTING A DATABASE

To create an SQL statement, first select an instance or database.

> **NOTE:** You can start Query Builder directly from a table or view which automatically selects the database which contains that table or view.

If you are working with Microsoft SQL Server or Sybase ASE, Query Builder provides two drop-down lists. The first drop-down list displays all available databases for the target server. The second drop-down list displays owners.

If you are working with Oracle or IBM DB2 LUW for Linux, Unix, and Windows the first drop-down list is unavailable.

> **NOTE:** You can start Query Builder directly from a table or view which automatically selects the database which contains that table or view.

1 Start Query Builder.

2 In the database drop-down list, click the target instance or database.

3   In the owners drop-down list, select the appropriate owner.

Query Builder is ready for Statement selection.

4   To select different instances or databases while Query Builder is running, on the Tables/ Views Tab, in the database drop-down list, click the target instance or database.

Rapid SQL clears the current query and displays a warning prompt.

5   To save the current query, click Yes.

Rapid SQL opens the Save As dialog box.

6   To continue without saving, click No.

Rapid SQL clears the SQL Diagram Pane and SQL Statement Pane.

For more information on saving queries in Query Builder, see <u>Saving and Reopening Queries</u>.

SELECTING A STATEMENT

Query Builder lets you build <u>SELECT</u>, <u>INSERT</u>, <u>UPDATE</u>, <u>DELETE</u>, and <u>CREATE VIEW</u> queries simultaneously.

To select a statement, do the following:

1   On the Query Builder tool bar, click the statement drop-down list, and then click the target statement type.

OR

In the *SQL Diagram Pane*, right-click, and then click the target statement type.

For more information, see <u>Using Query Builder</u>.

SELECTING TABLES AND VIEWS

To build a query, open one or more tables or views in the <u>SQL Diagram Pane</u>. You can use different tables or views for each type of query.

TIP: For multiple tables: Press SHIFT+click for adjacent tables or CTRL+click for nonadjacent tables. You can also drag the bounding line with your pointer to select multiple tables.

To select a Table or View, do the following:

1   In the *Tables/Views* Tab, drag the target table or view to the Diagram Pane.

    OR

    In the *Tables/Views* Tab, click target table or view and then, on the Query Builder tool bar, click Add.

    OR

    In the *Tables/Views* Tab, right-click target table or view, and then click Add.

    Query Builder displays the target table(s) and view(s) in the SQL Diagram Pane.

For more information, see [Using Query Builder](#).

SELECTING COLUMNS

You must select at least one column to build a query. Query Builder displays columns in each table in the SQL Diagram window. By default, Query Builder exposes every column in a table. You can select the columns you want to use for your query. Query Builder orders them, in your statement, in the select order.

Query Builder lets you select columns in the:

- [SQL Diagram Pane](#).

- [SQL Statement Tree](#).

You can select an individual column or all columns. Query Builder orders them, in your statement, in the select order. You can reorder columns after you set them in your diagram or statement.

For more information, see:

[Selecting Columns in the SQL Diagram Pane](#)

[Selecting Columns in the SQL Statement Tree](#)

[Selecting All Columns](#)

SELECTING COLUMNS IN THE SQL DIAGRAM PANE

To select a column in the SQL Diagram Pane, do the following:

1   Select the check box to the left of the target column name.

For more information, see [Using Query Builder](#).

SELECTING COLUMNS IN THE SQL STATEMENT TREE

Query Builder lets you select and set individual properties using the [Selected Column Properties Dialog Box](#).

For more information, see [Using Query Builder](#).

SELECTING ALL COLUMNS

Query Builder uses columns in statements based on the order of selection. When you select all columns, Query Builder displays the columns in the order they appear in the table.

> **NOTE:** Query Builder lets you select all columns in single or multiple tables.

To select all columns, do the following:

1   On the *Query Builder* menu, click Select Star.

  OR

  On the SQL Statement Tree, or in the SQL Diagram, right-click target table, or any of a group of selected tables, and then click Select Star.

For more information, see Using Query Builder.

BUILDING A QUERY

Query Builder lets you build five different queries, which you can run separately or simultaneously, depending on your needs. The table below describes these queries:

Query Type	Description
SELECT	Lets you create, manipulate and execute SELECT Statements for tables and views.
INSERT	Lets you create and manipulate INSERT Statements for tables.
UPDATE	Lets you create and manipulate UPDATE Statements for tables.
DELETE	Lets you create and manipulate DELETE Statements for tables.
CREATE VIEW	Lets you create and manipulate CREATE VIEW Statements for tables and views.

To build a Query, do the following:

- Select an instance or database.

- Select a Statement.

- Select your table(s) or view(s).

- Select your columns.

> **NOTE:** You can start Query Builder directly from a table or view which automatically selects the database which contains that table or view.

Query Builder lets you build queries that include both tables and views in the SQL Diagram Pane for SELECT and CREATE VIEW statements. For the INSERT, UPDATE, and DELETE statements, use one or the other object, but you cannot use both.

Once you make your selections, you can edit, restructure, and streamline your query. Query Builder offers many options for streamlining your queries.

For more information, see:

[Working with Tables and Views in the SQL Diagram Pane](#)

[Working with Columns in the SQL Diagram Pane](#)

[Joins](#)

[Creating a Clause using the SQL Statement Tree](#)

[Moving Tables and Columns in the SQL Statement Tree](#)

[Subqueries](#)

[Aliases](#)

BUILDING A SELECT STATEMENT

Query Builder lets you construct and execute simple-to-complex SELECT statements using data from any table or view. You can also create and edit [joins](#) for SELECT statements. Query Builder can check your query and warn you if there are syntax errors with the [Syntax Checker](#).

To build a SELECT statement, do the following:

1   On the Tools menu, click Query Builder.

    OR

    On the Datasource Explorer, expand the Database or Schema node, click Tables, and then on the Command menu, click Build Query.

    OR

    On the Tools tool bar, click Query Builder.

    OR

    On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

    OR

    On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

    Rapid SQL opens Query Builder.

2   In the statement drop-down list, click SELECT.

3   In the *Table Tree* Pane, select target table(s) or view(s) and move them to the *SQL Diagram Pane*.

4   In the target table or view, click target column(s), or click *Select Star* to select every column.

5   To check syntax, click Check.

6   To copy the statement, click Copy.

7   To execute the statement, click Execute.

Copying a SELECT Statement from the SQL Statement Pane

To copy any part of a statement from the SQL Statement Pane, do the following:

1 Open Query Builder, then begin a new SELECT statement.

   OR

   Open an existing SELECT statement.

2 In the SQL Statement Pane, select all, or the target portion of the statement.

3 On the Query Builder tool bar, click Copy.

   OR

   In the SQL Statement Pane, right-click, and then click Copy.

   Query Builder makes the target statement portion available on the clipboard.

For more information, see Building a Query.

BUILDING AN INSERT STATEMENT

Query Builder lets you construct and execute simple-to-complex INSERT statements using data from any table. To execute an INSERT statement, copy it to an ISQL Editor. You can also copy the statement to the clipboard for later use in the ISQL Editor. Query Builder also lets you save your statement at any time so that you can open it later for editing or execution.

Building an INSERT Statement

To build an INSERT Statement, do the following:

1 On the Tools menu, click Query Builder.

   OR

   On the Datasource Explorer, expand the Database or Schema node, click Tables, and then on the Command menu, click Build Query.

   OR

   On the Tools tool bar, click Query Builder.

   OR

   On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

   OR

   On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

   ```
   Rapid SQL opens Query Builder.
   ```

2 In the statement drop-down list, click INSERT.

3 In the *Table Tree* Pane, select target table, and move it to the *SQL Diagram Pane*.

4 In the target table, click target column(s).

Copying an INSERT Statement from the SQL Statement Pane

To copy any part of a statement from the SQL Statement Pane, do the following:

1   Open Query Builder, then begin a new INSERT statement.

    OR

    Open an existing INSERT statement.

2   In the SQL Statement Pane, select all, or the target portion of the statement.

3   On the Query Builder tool bar, click Copy.

    OR

    In the SQL Statement Pane, right-click, and then click Copy.

    Query Builder makes the target statement portion available on the clipboard.

For more information, see Building a Query.

BUILDING AN UPDATE STATEMENT

Query Builder lets you construct and execute simple-to-complex UPDATE statement using data from any table. To execute an UPDATE statement, copy it to an ISQL Editor. You can also copy the statement to the clipboard for later use in the ISQL Editor. Query Builder also lets you save your statement at any time so that you can open it later for editing or execution.

Building an UPDATE Statement

To build an UPDATE statement, do the following:

1   On the Tools menu, click Query Builder.

    OR

    On the Datasource Explorer, expand the Database or Schema node, click Tables, and then on the Command menu, click Build Query.

    OR

    On the Tools tool bar, click Query Builder.

    OR

    On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

    OR

    On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

    Rapid SQL opens Query Builder.

2   In the statement drop-down list, click *UPDATE*.

3   In the *Table Tree* Pane, select target table and move it to the SQL Diagram Pane.

4   In the target table, click target column(s).

Copying an UPDATE Statement from the SQL Statement Pane

To copy any part of a statement from the SQL Statement Pane, do the following:

1   Open Query Builder, then begin a new UPDATE statement.

OR

Open an existing UPDATE statement.

2   In the SQL Statement Pane, select all, or the target portion of the statement.

3   On the Query Builder tool bar, click Copy.

OR

In the SQL Statement Pane, right-click, and then click Copy.

Query Builder makes the target statement portion available on the clipboard.

For more information, see Building a Query.

BUILDING A DELETE STATEMENT

Query Builder lets you construct DELETE statements using data from any table. Query Builder displays a Confirmation Option Message box when you create a DELETE statement. You can set the Statement Properties dialog box to display or hide this message when creating a DELETE statement.

To execute a DELETE statement, copy it to an ISQL Editor. You can also copy the statement to the clipboard for later use in the ISQL Editor. Query Builder also lets you save your statement at any time so that you can open it later for editing or execution.

Building a DELETE Statement

To build a DELETE statement, do the following:

1   On the Tools menu, click Query Builder.

OR

On the Datasource Explorer, expand the Database or Schema node, click Tables, and then on the Command menu, click Build Query.

OR

On the Tools tool bar, click Query Builder.

OR

On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

OR

On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

Rapid SQL opens Query Builder.

2   In the statement drop-down list, click *DELETE*.

3   In the *Table Tree* Pane, select target table, and move it to the *SQL Diagram Pane*.

Copying a DELETE Statement from the SQL Statement Pane

To copy any part of a statement from the SQL Statement Pane, do the following:

1   Open Query Builder, then begin a new DELETE statement.

OR

Open an existing DELETE statement.

2   In the SQL Statement Pane, select all, or the target portion of the statement.

3   On the Query Builder tool bar, click Copy.

OR

In the SQL Statement Pane, right-click, and then click Copy.

Query Builder makes the target statement portion available on the clipboard.

For more information, see Building a Query.

BUILDING A CREATE VIEW STATEMENT

Query Builder lets you construct and execute simple-to-complex CREATE VIEW statements using data from any table or view. You can also copy the statement to the clipboard for later use in the ISQL Editor. Query Builder also lets you save your statement at any time so that you can open it later for editing or execution.

To build a CREATE VIEW statement, do the following:

1   On the Tools menu, click Query Builder.

OR

On the Datasource Explorer, expand the Database or Schema node, click Tables, and on the Command menu, click Build Query.

OR

On the Tools tool bar, click Query Builder.

OR

On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

OR

On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

`Rapid SQL opens Query Builder.`

2   In the statement drop-down list, click CREATE VIEW.

3   In the *Table Tree* Pane, select target table or view and move it to the *SQL Diagram Pane*.

NOTE: Query Builder supports multiple tables and views in a CREATE VIEW statement.

4   In the target table or view, click the target column(s).

5   To check syntax, click Check.

6   To copy the statement, click Copy.

7   To execute the CREATE VIEW Statement, click the SQL Statement Pane, and then press any key.

     Query Builder opens the Edit SQL dialog box.

8   Click OK.

CAUTION: If you have used this method previously, and you selected the Please do not show me this dialog again check box, on the Edit SQL dialog box, Query Builder does not display the Edit SQL dialog box. It pastes your statement directly to the ISQL Editor.

     Rapid SQL opens the ISQL Editor.

9   In the *ISQL Editor*, on the line, CREATE VIEW NAME AS, replace the word NAME with a name for your view.

10   On the tool bar, click Execute.

     Rapid SQL executes the CREATE VIEW query.

11   To close the Editor, click Close.

     Rapid SQL opens the ISQL Editor save message.

12   Click No.

     Rapid SQL returns to Query Builder.

13   To add the view to the table tree, on the *Query Builder* menu, click Refresh.

     OR

     Press F5.

     Query Builder adds the view to the Table Tree Pane.

Copying a CREATE VIEW Statement from the SQL Statement Pane

To copy any part of a statement from the SQL Statement Pane, do the following:

1   Open Query Builder, then begin a new CREATE VIEW statement.

     OR

     Open an existing CREATE VIEW statement.

2   In the SQL Statement Pane, select all, or the target portion of the statement.

3   On the Query Builder tool bar, click Copy.

OR

In the SQL Statement Pane, right-click, and then click Copy.

Query Builder makes the target statement portion available on the clipboard.

For more information, see Building a Query.

WORKING WITH TABLES AND VIEWS IN THE SQL DIAGRAM PANE

Query Builder lets you organize your tables and views in the SQL Diagram Pane. You can also customize appearance, change visual aspects, and adjust layout while continuing to manufacture a query. You can resize or customize a selected table and view, or move them to the front or back of the diagram. The key symbol indicates a column that is indexed or participates in a primary key.

- Selecting and Deselecting Tables and Views

- Moving Tables and Views

- Moving Additional Tables and Views to the SQL Diagram Pane

- Deleting Tables and Views

Query Builder can automatically dictate a layout in the SQL Diagram Pane using the Auto Layout button.

SELECTING AND DESELECTING TABLES AND VIEWS

You can select tables and views in the SQL Diagram Pane. You can make changes to more than one table or view simultaneously by selecting multiple tables or views.

To select and deselect Tables and Views, do the following:

1   To select a table, click the table title bar.

2   To select more than one table, drag the pointer to enclose all target tables with the bounding line.

Query Builder selects all target tables; none have handles.

3   To select all tables, in the SQL Diagram, right-click, and then click *Select All*.

4   Click the SQL Diagram workspace to deselect all tables.

For more information, see Working with Tables and Views.

MOVING TABLES AND VIEWS

Query Builder lets you move tables and views in the SQL Diagram Pane. It also moves selections and joins with the tables and views.

To move Tables and Views, do the following:

1   To move a table or view, drag the title bar to the target location.

>   **NOTE:**   If you select more than one table or view, Query Builder moves all selected tables and
>   views and any joins with the pointer.

For more information, see <u>Working with Tables and Views</u>.

MOVING ADDITIONAL TABLES AND VIEWS TO THE SQL DIAGRAM PANE
Query Builder sets tables and views in your statement in the order that you move them to the
SQL Diagram Pane. Tables and views moved into the Diagram Pane appear first in your
statement, including all joins connecting that table. To change the order of tables, move them
back into the Table Tree and re-select them in the order in which you would like to join them.

Moving Additional Tables or Views

To move additional tables or views, do the following:

1   Click the target table or view and drag it to the Diagram Pane.

For multiple tables or views: Use SHIFT+click for adjacent tables or views or use CTRL+click
for nonadjacent tables and views.

OR

Click the target table or view, and then on the Query Builder tool bar, click Add.

OR

Right-click the target table or view, and then click *Add*.

For multiple tables or views: Use SHIFT+click for adjacent tables or views or use CTRL+click
for non-adjacent tables and views.

>   **NOTE:**   Moving a table or view to the SQL Diagram Pane is not available while a query is
>   executing.

For more information, see <u>Working with Tables and Views</u>.

DELETING A TABLE OR VIEW

To delete tables from the SQL Diagram Pane, do the following:

1 Right-click the target table or view, and then click Delete.

   OR

   In the SQL Diagram, click target table or view, and then on the Query Builder tool bar, click Delete.

   OR

   In the SQL Diagram, right-click the target table or view, and then click Delete.

   Query Builder deletes the table from the SQL Diagram, SQL Statement, and SQL Statement Tree.

For more information, see <u>Working with Tables and Views</u>.

WORKING WITH COLUMNS IN THE SQL DIAGRAM PANE

You can customize queries by selecting and deselecting columns in the SQL Diagram Pane. You can customize columns using the <u>Selected Column Properties dialog box</u>.

SELECTING AND DESELECTING COLUMNS

You can select and deselect columns in the SQL Diagram. Query Builder lets you select and deselect individual columns or all columns. Your results reflect the order of selection. You can change the order of columns after you set them in your diagram or statement.

> **TIP:** You can also select, re-order and deselect columns in the <u>SQL Statement Tree</u>.

Selecting Individual Columns

To select individual columns, do the following:

1 To select a column, in the SQL Diagram, select the check box to the left of the target column name.

Deselecting Individual Columns

To deselect individual columns, do the following:

1 To deselect a column, in the SQL Diagram, select the check box to the left of the target column name.

> **NOTE:** When you clear the columns, Query Builder deletes the columns and any sub clauses from the SQL Statement Pane and SQL Statement Tree.

Selecting All Columns

To select all columns, do the following:

1 On the *Query Builder* menu, click Select Star.

OR

On the SQL Statement Tree, or in the SQL Diagram, right-click target table, or any of a group of selected tables, and then click Select Star.

NOTE: Query Builder uses columns in statements based on the order of selection. When you select all columns, Query Builder displays the columns as they appear in the table.

Deselecting All Columns

To deselect all columns, do the following:

1 On the *Query Builder* menu, click *Select None*.

OR

On the SQL Statement Tree, or in the SQL Diagram, right-click target table, or any of a group of selected tables, and then click Select None.

Query Builder adds or removes selected columns from the SQL Statement Tree and the SQL Statement Pane.

SELECTING ALL OR DISTINCT COLUMNS

Selecting ALL or DISTINCT columns is a way to filter data in your query. Selecting ALL columns means all rows displays results in the grid regardless of duplication in non-primary key columns. The DISTINCT column function is a query process that limits duplicate data in non-primary key columns to rows with the first iteration of any identical data. For example, if there are two identical addresses for different last names, and the column with a primary key does not participate in the query, only the row with the first instance of the address displays in the results of the query.

To select ALL or DISTINCT columns, do the following:

1 In the Statement Tree pane, right-click the ALL or DISTINCT node, click Properties, and then select the ALL or DISTINCT check box.

OR

In the Statement Tree pane, double click the ALL or DISTINCT node. Query Builder toggles to the opposite function.

NOTE: You can change between ALL or DISTINCT at any time prior to executing or copying a query.

JOINS

Joins let you distill the information in your database to a usable form. Query Builder lets you create, manipulate, and edit work with joins without requiring knowledge of the underlying SQL code. Query Builder lets you create any type of join for SELECT and CREATE VIEW Statements. You can create self joins for UPDATE or DELETE Statements. You cannot create joins for INSERT Statements.

Query Builder includes four types of joins. The table below describes joins and their availability in Query Builder:

Join	Statement Availability	Description
Inner Join	SELECT, CREATE VIEW, DELETE, UPDATE	Returns data from the joined tables that match the query's join criteria and set a relation between tables or views. Inner joins return results where the join condition is true.
Left Outer Join	SELECT, CREATE VIEW	Returns all data from the primary table and data from the joined tables that match the query's join criteria and set a join relation operator from a column in a primary table or view to a column in a secondary table or view.
Right Outer Join	SELECT, CREATE VIEW	Returns all data from the primary table and data from the joined tables that match the query's join criteria and set a join relation operator from a column in a secondary table or view to a column in a primary table or view.
Self Join	SELECT, CREATE VIEW	Set a relation between columns in the same table.

In the Query Builder SQL Diagram Pane, you can create, edit, and delete joins. You can edit joins in the Join dialog box.

Joins are the way you can filter data in relational databases. Query Builder lets you change the types of joins between tables, views and columns. It is important that you have some knowledge of the data in your tables, and the datatypes for each column. This information helps you frame a better query, and filter your data for maximum effect.

For more information, see Joins.

INNER JOINS

Inner joins are the most common types of joins for SELECT statements. An inner join returns information from two tables where the relation between two target columns is true for both columns.

The join operand determines the relation results, for example, if the join operand is equals, then identical data, in two columns, is the only result. If the join operand is not equals, Query Builder only returns data that is different between two columns.

For example, if you have an inner join matching territory numbers between the table dbo.Managers and dbo.Clients, running the query returns all Managers and Clients with matching territory numbers:

Query Builder displays the following results from this query with an inner join

> **NOTE:** Query Builder displays results of columns in the order of selection. You can reorder columns by deselecting and selecting in the SQL Diagram Pane, the Selected Columns Properties dialog box, or the SQL Statement Tree.

For more information, see Joins.

LEFT OUTER JOINS

Left outer joins bring back a different data set than inner joins. Left outer joins retrieve all the data in columns selected from the primary table, and only matching data from the joined or secondary table.

For example, in the same pair of tables, a left inner join from dbo.Managers to dbo.Clients, where the columns Current Territory and Territory are joined, displays different results.

> **NOTE:** There is one additional manager who does not have a client, but because a left outer join includes all data from selected columns in the primary table, the last entry in the illustration is displayed.

For more information, see Joins.

RIGHT OUTER JOINS

Right outer joins return opposite results from a left outer join. In a right outer join, you are asking for all the information in the secondary table's column, and the join operator's matching information from the primary table.

For example, in the same set of data we used in the left outer join example, a right outer join returns all clients from dbo.Client, and only managers who match territory numbers, in the joined column.

> **NOTE:** The managers are the same as the first, inner join, but a right outer join returns the additional clients without matching managers.

For more information, see Joins.

SELF JOINS

A self join is a join within a single table. Query Builder lets you return specific information from a single table using a self join.

For example, in our example table, there is a column for the number of clients and another column with the goal client total for a territory.

A self join can ascertain which managers are reaching their quota. Notice that the join relation operator in the example is greater than or equal to, which shows managers exceeding quota as well.

For more information, see Joins.

ADDING AND DELETING A JOIN IN THE SQL DIAGRAM PANE

Query Builder lets you add and delete joins. This method adds a WHERE clause in your query. You can join different tables and or views in a SELECT or CREATE VIEW statement.

Adding a Join

To add a Join, do the following:

1   In the SQL Diagram Pane, drag the target column to the second column.

Query Builder displays both a line joining the two columns in the SQL Diagram Pane and the corresponding SQL code in the SQL Statement Pane.

Removing a Join

Query Builder lets you remove joins from your query. Query Builder automatically deletes joins from the query in the SQL Statement Pane, when you remove them from the SQL Diagram Pane.

To remove a join, do the following:

1   Click the target join, and then on the *Query Builder* tool bar, click Delete.

OR

Right-click the target join, and then click Delete.

Query Builder deletes the Join.

For more information, see Joins.

EDITING JOINS

Query Builder lets you edit joins with the Join editor. You can edit join parameters in a SELECT, UPDATE, DELETE, and CREATE VIEW Statement.

The table below describes the options in the Join dialog box:

Option	Description
From Table Column	The primary column in the join.
To Table Column	The secondary column in the join.
Select the join relation operator	Click the target join operator. If it is not equals, the operator displays on the join in the SQL Diagram Pane.
Join Type: Inner	Click to make the join an inner join. Aggregates are only available for inner joins.

Option	Description
Join Type: Left Outer	Click to make the join a left outer join.
Join Type: Right Outer	Click to make the join a right outer join.

Completing the Join Dialog Box

1   In the *SQL Diagram Pane*, right-click the target join, and then click Properties.

    OR

    In the *SQL Diagram Pane*, double-click the target join.

    OR

    On the *SQL Statement Tree*, expand the *Where* and *And* nodes, and then double-click the target join.

    Query Builder opens the Join dialog box.

    NOTE: For IBM DB2 LUW for Linux, Unix, and Windows servers, there is an additional join object in the SQL Statement Tree. The Join On node displays join relations between columns in IBM DB2 LUW for Linux, Unix, and Windows tables and views.

Changing a Join Color

Query Builder lets you change the color at a join in the SQL Diagram Pane. Complex statements using many tables and multiple joins can be easier to view if joins have different colors.

To change the color of a join, do the following:

1   Right-click the target join, and then click Color.

    Query Builder opens the Color dialog box.

2   In the Basic colors grid, click a target color

    OR

    Click Define Custom Colors, then create a custom color.

    NOTE: Query Builder lets you save custom colors for the current color. Click Add to Custom Color to have the option of using that color for your queries.

3   Click OK.

For more information, see Joins.

AUTO LAYOUT

The Auto Layout function displays tables and views in the SQL Diagram Pane. It makes the best use of the available area in the SQL Diagram Pane by placing your tables and views in the most efficient manner. If the automatic join function is on, Query Builder displays all joins between columns in your diagram. Query Builder lets you run the automatic layout function any time you have tables or views in the SQL Diagram Pane.

Using Auto Layout

To use Auto Layout, do the following:

1   On the Query Builder menu, click Auto Layout.

OR

On the Query Builder tool bar, click Auto Layout.

OR

In the SQL Diagram Pane, right-click, and then click Auto Layout.

Query Builder organizes your tables in the SQL Diagram Pane.

AUTO JOINS

Query Builder includes an automatic join function that displays joins between selected tables and views in the SQL Diagram Pane. The Auto Join function seeks columns with the same name and data type. You can set global automatic join parameters in the Rapid SQL Options Editor. You can use the Statement Properties Editor to set local join parameters for the current Query Builder session without changing the global parameters.

Using Auto Join

To use Auto Join, do the following:

1   On the Query Builder menu, click Auto Join.

OR

On the Query Builder tool bar, click Auto Join.

OR

In the SQL Diagram Pane, right-click, and then click Auto Join.

Query Builder joins columns in the SQL Diagram Pane.

CREATING A CLAUSE USING THE SQL STATEMENT TREE

Query Builder lets you build more detailed WHERE, ORDER BY, GROUP BY, and HAVING clauses using the SQL Statement Tree. Query Builder lets you add clauses to SELECT, UPDATE, DELETE, and CREATE VIEW statements.

> **NOTE:** Query Builder does not support clauses for INSERT statements.

The table below describes these clauses:

Clause	Description
WHERE	Limits rows in the query.
ORDER BY	Orders the results of the query to a target column.
GROUP BY	Groups target columns in the query.
HAVING	Filters out groups of data.

CREATING A WHERE CLAUSE

Query Builder lets you create a WHERE clause from the SQL Statement Tree which automatically displays in your query.

> **NOTE:** Any additional WHERE clauses are displayed as HAVING clauses.

The table below describes the options and functionality on the Where dialog box.

Option	Description
Operand (Left)	Lets you click the target column for the first part of your WHERE clause.
	NOTE: Query Builder lists every column in all tables in the SQL Diagram in the Operand lists.
Operator	Lets you select the target operator.
Operand (Right)	Lets you click the target column for the second part of your WHERE clause.
	Query Builder automatically writes the query language in the Statement option box.

> **NOTE:** Query Builder does not display clause phrases created from the SQL Statement Tree in the SQL Diagram Pane.

Creating a WHERE Clause

To Create a WHERE clause, do the following:

1   Click the WHERE node, and then on the Query Builder tool bar, click New.

OR

Right-click the WHERE node, and then click New.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

DELETING A WHERE CLAUSE
To delete a WHERE clause, do the following:

1   Expand the AND node, and then on the Query Builder tool bar, click Delete.

    OR

    Expand the AND node, right-click target column and then click Delete.

    Query Builder deletes the target clause and removes it from the SQL Statement Pane.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

CREATING AN AND CLAUSE IN A WHERE CLAUSE
Query Builder lets you add an AND clause from the SQL Statement Tree which automatically displays in your query.

The table below describes the options and functionality on the Where dialog box.

Option	Description
Operand (Left)	Lets you click the target column for the first part of your WHERE clause.
Operator	Lets you select the target operator.
Operand (Right)	Lets you click the target column for the second part of your WHERE clause.
	Query Builder automatically writes the query language in the Statement option box.
New Button	Click to clear your selections but remain in the Where dialog box.
	Query Builder adds another AND clause to your query.

To open the Where dialog box, do the following:

1   Click the AND node, and then on the Query Builder tool bar, click New.

    OR

    Expand the WHERE node, right-click the AND node, and then click New.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

DELETING AN AND CLAUSE

To delete an AND clause, do the following:

1 Expand the AND node, click target column, and then on the Query Builder tool bar, click Delete.

   OR

   Expand the AND node, click target column, and then on the keyboard press DELETE.

   OR

   Expand the AND node, right-click the target column, and then click Delete.

   Query Builder deletes the target clause and removes it from the SQL Statement Pane.

For more information, see Creating a Clause using the SQL Statement Tree.

INSERTING AN AND OR OR CLAUSE

Query Builder lets you insert an AND or an OR WHERE clause from the SQL Statement Tree which automatically displays in your query. Query Builder lets you insert AND or OR clauses at any appropriate point in the SQL Statement Tree.

The table below describes the options and functionality on the Where dialog box.

Option	Description
Operand (Left)	Lets you click the target column for the first part of your WHERE clause.
Operator	Lets you select the target operator.
Operand (Right)	Lets you click the target column for the second part of your WHERE clause.
	Query Builder automatically writes the query language in the Statement option box.
New Button	Click to clear your selections but remain in the Where dialog box.
	Query Builder adds another AND clause to your query.

To insert an AND or OR Clause, do the following:

1 On the SQL Statement Tree, expand the WHERE node, right-click the target AND node, then click Insert, and then click And or Or.

For more information, see Creating a Clause using the SQL Statement Tree.

DELETING AN OR CLAUSE

To delete an OR clause, do the following:

1   Expand the OR node, and then on the Query Builder tool bar, click Delete.

    OR

    Expand the OR node, right-click the target column and then click Delete.

    Query Builder deletes the target clause and removes it from the SQL Statement Pane.

For more information, see Creating a Clause using the SQL Statement Tree.

CREATING AN ORDER BY CLAUSE

Query Builder lets you create an ORDER BY clause from the SQL Statement Tree which automatically displays in your query.

The table below describes the Order By Columns dialog box.

Option	Description
Available Columns	Select target column(s) and click the right arrow. Query Builder moves target column from the Available Columns list to the Order By Columns list.
	NOTE: Query Builder sorts query results based on the order that columns are placed in the ORDER BY clause.
Order	Lets you select the target sort order.
	ASC - Ascending
	DESC - Descending
	Query Builder displays the SQL language in the Order By Statement box.

To open the Order By Columns dialog box, do the following:

1   On the SQL Statement Tree, click the ORDER BY node, and then on the Query Builder tool bar, click Properties.

    OR

    On the SQL Statement Tree, right-click the ORDER BY node, and then click Properties.

For more information, see Creating a Clause using the SQL Statement Tree.

CHANGING THE SORT ORDER IN AN ORDER BY CLAUSE

To quickly change the sort order of a column in a query, do the following:

1 On the SQL Statement Tree, expand the ORDER BY node, and then double-click the target column.

    OR

    On the SQL Statement Tree, expand the ORDER BY node, then right-click the target column, and then click Properties.

    Query Builder opens the Order dialog box.

2 Click the target sort order, and then click OK.

    Query Builder appends the Order By clause for target column with the appropriate sort order in the SQL Statement Pane.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

DELETING AN ORDER BY CLAUSE

To delete an ORDER BY clause, do the following:

1 Expand the ORDER BY node, and then on the Query Builder tool bar, click Delete.

    OR

    Expand the ORDER BY node, right-click the target column, and then click Delete.

    Query Builder deletes the target clause and removes it from the SQL Statement Pane.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

CREATING A GROUP BY CLAUSE

The table below describes the options and functionality on the Group By Columns dialog box.

Option	Description
Selected Columns	Select target column(s) and click the right arrow. Or click the Select All button. Query Builder moves target column from the Selected Columns list to the Group By Columns list.
	NOTE: Query Builder sorts query results based on the order that columns are placed in the ORDER BY clause.
Clear All Button	Click to move target column from the Group By Columns list to the Selected Columns list.
	Query Builder displays the SQL language in the Group By Statement window.

Creating a GROUP BY Clause

To create a GROUP BY clause from the SQL Statement Tree which automatically displays in your query, do the following:

1 On the SQL Statement Tree, double-click the GROUP BY node.

   OR

   On the SQL Statement Tree, right-click the GROUP BY node, and then click New.

   Query Builder adds all the selected columns in your table(s) to the GROUP BY node in the SQL Statement Tree, and to the appropriate location in the SQL Statement Pane.

2 On the GROUP BY node, double-click any column.

   OR

   On the GROUP BY node, click any column, then on the Query Builder menu, click New.

   OR

   On the GROUP BY node, right-click any column, then click Properties.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

DELETING A GROUP BY CLAUSE

To delete a GROUP BY clause, do the following:

1 On the SQL Statement Treeexpand the GROUP BY node, and then on the Query Builder tool bar, click Delete.

   OR

   On the SQL Statement TreeExpand the GROUP BY node, right-click the target column, and then click Delete.

   Query Builder deletes the target clause and removes it from the SQL Statement Pane.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

CREATING A HAVING CLAUSE

A HAVING clause is a type of WHERE clause. It filters additional information from your tables. Query Builder lets you create a HAVING clause from the SQL Statement Tree which automatically displays in your query. Query Builder lists every column in all tables in the SQL Diagram in the Operand lists. Query Builder displays the datatype of a column in the operand boxes.

The table below describes the options and functionality on the Having dialog box.

Option	Description
Operand (Left)	Lets you click the target column for the first part of your HAVING clause.
Operator	Lets you select the target operator.
Operand (Right)	Lets you click the target column for the second part of your HAVING clause.
	Query Builder automatically writes the query language in the Statement option box.
New Button	Click to clear your selections but remain in the Having dialog box.
	Query Builder adds another AND clause to your query.

> **NOTE:** Query Builder does not display clause phrases created from the SQL Statement Tree in the SQL Diagram Pane.

To create a HAVING clause, do the following:

1 On the SQL Statement Tree, expand the *HAVING* node, and then expand the *And* node. If there is not a join listed on the *And* node, double-click *And*. If there is a join listed, use the shortcut option below.

   OR

   On the SQL Statement Tree, right-click the *HAVING* node, and then click New.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

DELETING A HAVING CLAUSE

To delete a HAVING clause, do the following:

1 On the SQL Statement Treeexpand the HAVING node, and then on the Query Builder tool bar, click Delete.

   OR

   On the SQL Statement Tree expand the HAVING node, right-click the target column, and then click Delete.

   Query Builder deletes the target clause and removes it from the SQL Statement Pane.

For more information, see [Creating a Clause using the SQL Statement Tree](#).

CHANGING TABLES AND COLUMNS LOCATION IN THE SQL STATEMENT TREE

Query Builder lets you move tables and columns on the SQL Statement Tree by dragging them to new locations. You can move columns from the AND and OR nodes to an AND or OR node on the WHERE and HAVING clause nodes. Query Builder changes the query in the SQL Statement Pane to match each move. Query Builder moves tables or columns you are dragging below target table or column.

To move a table or column in the SQL Statement Tree, do the following:

1   Expand target node, then drag the target table or column to a new location.

    Query Builder makes the appropriate change in the query in the SQL Statement Pane.

    NOTE: Query Builder lets you select multiple tables or columns.

2   To move a table or column to the bottom of a node, drag it to the target node.

    Query Builder displays the target table or column at the bottom of target node.

For more information, see Creating a Clause using the SQL Statement Tree.

SUBQUERIES

Query Builder lets you build subqueries for SELECT and CREATE VIEW statements in the WHERE or HAVING clause. The table below describes the options available for a subquery in Query Builder:

Operand	Location	Description
EXISTS	Left operand	Specifies data that exists in a column.
NOT EXISTS	Left operand	Specifies data that does not exist in a column.
ANY	Right operand	Specifies data satisfying the operator parameters.
ALL	Right operand	Specifies data satisfying the operator parameters.
SELECT	Right operand	Specifies data satisfying the operator parameters.

The table below describes the options and functionality on the Where or Having dialog boxes.

Option	Description
Operand (Left)	Lets you click the target column for the first part of your clause.
Operator	Lets you select the target operator.
Operand (Right)	Lets you click the target column for the second part of your clause. Query Builder displays the working subquery in the Statement window.
Subquery	Paste or type the SUBQUERY statement.

To use the WHERE and HAVING dialog boxes to create subqueries, do the following:

1 On the SQL Statement Tree, expand the *Where* or *Having* node, and then expand the *And* node. If there is not a join listed on the *And* node, double-click And. If there is a join listed, use the shortcut option below.

   OR

   On the SQL Statement Tree, right-click the *Where* or *Having* node, and then click New.

For more information, see Creating a Clause using the SQL Statement Tree.

SYNTAX CHECKER

The Syntax Checker scans SQL statements for errors. You can check your syntax at any time while you are fashioning a query, or a Procedure or Function. Query Builder can automatically run a syntax check to validate your query when you are executing or copying a statement.

> **NOTE:** Query Builder lets you continue with your query even if there are errors detected in the syntax.

Using the Syntax Checker

The table below describes the possible syntax errors the Query Builder Syntax Checker tool displays, in order:

Error	Description
Does the query contain duplicate aliases?	Query Builder returns an error message when it detects duplicate aliases.
If the query has a HAVING clause, is there a GROUP BY clause?	Query Builder returns an error message when it detects a HAVING clause without a GROUP BY clause.
If there are aggregates, or a GROUP BY clause, are all columns in one or the other?	Query Builder returns an error message when it detects an aggregate, or a GROUP BY clause without all columns in one or the other.
Are there joins against non-indexed columns, or columns not participating in a primary key?	Query Builder returns a warning when it detects a join against a non-indexed column, or a column not participating in a primary key.
Are there joins between different datatypes?	Query Builder returns a warning when it detects a join between different datatypes.
Are there cross-products in the query?	Query Builder returns a warning when it detects a cross-product in the query.

SAVING AND REOPENING QUERIES

You can save and reopen queries in Query Builder. Saving a query saves the SQL Diagram, SQL Statement, and Query Builder Explorer view. Query Builder automatically checks for changes in your database or instance between the time you save the query and the time you reopen it. Query Builder prompts you to save any time you try to close Query Builder, or any time you attempt quitting Rapid SQL.

Query Builder runs Schema Change detection any time you set a query to execute, refresh the data, or open a saved query.

Rapid SQL lets you open multiple saved queries simultaneously.

Saving Queries

To save a query using standard Save and Save As functions, do the following:

1   On the File menu, click Save or Save As.

    OR

    On the Main tool bar, click Save As.

    Rapid SQL opens the Save As dialog box.

2   In the File name box, type the name of the query.

    NOTE: By default, the product appends the.qbl extension to Query Builder files. If there is more than one Query Builder session in progress when you save, the file is further appended with an integer, for example.qbl2.

    TIP: Rapid SQL lets you save data in text (*.txt) and XML (*.xml) file formats.

3   Click OK.

    `Rapid SQL saves the file and closes the Save As dialog` box.

Reopening Queries

You can open a query using standard Open functions. Query Builder displays the Query Builder diagram, statement and Query Builder Explorer Pane and it checks the instance or database for schema changes.

The Query Builder Schema Change Detection component checks for:

- Renamed or dropped tables referenced in the query. Renamed tables that have been renamed are considered dropped.

- Renamed or dropped columns referenced in the query. Renamed columns are considered dropped and inserted.

- <u>Columns</u> added or reordered in tables referenced in the query.

If Query Builder detects a change, it opens the Schema Change Detected dialog box. The dialog box displays details of changes to your schema.

Query Builder opens an ISQL Editor with the last saved versions of the SQL statement.

USING DATA EDITOR WITH QUERY BUILDER

SELECT statements. Rapid SQL lets you open multiple Data Editor sessions so that you can continue to change your data until you find the best match for your query.

CAUTION: Data Editor is a real-time editor. Changes in your data using Data Editor are permanent.

Opening the Data Editor from Query Builder

To open the Data Editor from Query Builder, do the following:

1   On the Tools menu, click Query Builder.

OR

On the Datasource Explorer, expand the Database or Schema node, click Tables, and then on the Command menu, click Build Query.

OR

On the Tools tool bar, click Query Builder.

OR

On the Explorer, expand Tables, right-click the target table(s), and then click Build Query.

OR

On the Explorer, expand Views, right-click the target view(s), and then click Build Query.

Rapid SQL opens Query Builder.

2   Select a [database](#) or instance.

3   Select a [table](#).

4   Select a [column](#), or columns.

5   On the Query Builder menu, click Edit Data.

OR

On the Query Builder tool bar, click Edit Data.

Rapid SQL opens Data Editor.

For more information on using the Data Editor, see [Using Data Editor](#).

DATA EDITOR

The Edit Data function opens the Data Editor. You can use the Data Editor to edit your tables in real-time. The Data Editor supports all editable datatypes and is an alternative way to add, edit, or delete data from your tables.

> NOTE: You can use Data Editor within Query Builder to edit data in tables while you create SELECT statements. You can open multiple Data Editor sessions so that you can continue to change your data until you find the best match query.

The Data Editor includes a Data Editor Filter that lets you select the columns in your table that you want to edit. You must select at least one column to use the Data Editor. The Data Editor Filter is not available for the Query Builder.

For more information, see:

Data Editor Design

Using Data Editor

DATA EDITOR DESIGN

The Data Editor includes the following components:

- Edit Window
- ISQL Window
- Tool Bar
- Data Editor Filter
- Date/Time Format Builder
- Using Data Editor

DATA EDITOR EDIT WINDOW

Data Editor displays all the information in the target table in the Data Editor Edit Window. You can edit data directly in this window.

For more information, see:

ISQL Window

Tool Bar

Data Editor Filter

Date/Time Format Builder

Using Data Editor

DATA EDITOR ISQL WINDOW

The Data Editor ISQL Window displays the active SQL statement, which uses the data from the target table.

When appropriate, Data Editor displays a History Tab. The History Tab displays all SQL Statements created in the current session. If there is an error, Data Editor displays an Error Tab. The Error Tab details any errors in data entry encountered during execution.

For more information, see:

Edit Window

Tool Bar

Data Editor Filter

Date/Time Format Builder

Using Data Editor

DATA EDITOR TOOL BAR

The Data Editor tool bar lets you access commonly used features.

The table below describes the function of each Data Editor tool.

Description	Function
Stop Button.	Stops loading data to the Data Editor. Data Editor displays rows up to the stopping point.
List of options for the target table.	Displays the editing mode for the target table.
Execute SQL button	Executes the current SQL statement for the target table.
Insert Record button	Inserts new record for the target table. New records display at the end of the table. For related information, see Default Value Handling.
Save Current Row button	Saves data in the current selected row. Data Editor prompts to save when you attempt to leave a row in Live mode. For related information, see Default Value Handling.
Remove Data button	Removes data in target row. Data Editor displays an optional prompt.
Clear SQL Text button	Clears SQL text from the SQL Statement Pane.
Undo button	Undoes the most recent operation.
Redo button	Redoes the most recent operation.
First Record button	Moves to the first record in the target table.
Last Record button	Moves to the final record in the target table.
Filter Data button	Filters table using the target cell as the filter parameter.
Refresh button	Reloads data for target table

Description	Function
Calendar button	Sets correct format for target date/time cell. Enables the Calendar window.
Date/Time Format Builder button	Opens the [Date/Time Format Builder dialog box](#).
Date/Time Format Undo button.	Undoes the last date/time format display.
Date/Time Format Redo button.	Redoes the last date/time format display.
Close button	Closes and exits Data Editor.

For more information, see:

[Edit Window](#)

[ISQL Window](#)

[Data Editor Filter](#)

[Date/Time Format Builder](#)

[Using Data Editor](#)

DATA EDITOR FILTER

The Data Editor Filter displays the columns of a target table and the corresponding SELECT SQL Statement. You can select columns from the filter for selective data editing.

For more information, see:

[Edit Window](#)

[ISQL Window](#)

[Tool Bar](#)

[Date/Time Format Builder](#)

[Using Data Editor](#)

NOTES ON XML TYPES AND UNICODE DISPLAY IN THE DATA EDITOR

When working with data in the Data editor, keep the following in mind:

- XML data types are supported for IBM DB2 for Windows, Unix, and Linux, Microsoft SQL Server, and Oracle. In the Data editor, XML data types are displayed and entered as LOB content.

- Support for display of Uincode characters is provided as follows:

  - IBM DB2 for Windows, Unix, and Linux V8 and V9: **character**, **clob**, **varchar**, and **longvarchar** types

  - SQL Server 2000: for **nchar**, **nvarchar**, and **ntext** types

  - SQL Server 2005: **nchar**, **nvarchar**, **ntext**, and **nvarchar(max)** types.

  - Oracle 8i, 9i, and 10g: **NCHAR**, **NVARCHAR2** and **NCLOB** for non-Unicode UTF8 Character Set Instances and **NCHAR**, **NVARCHAR2**, **CHAR**, **VARCHAR2**, **LONG**, **NCLOB** and **CLOB** for Unicode UTF8 Character Set Instances

  - Sybase 12.5 and 15.2: **UNICHAR**,**UNIVARCHAR** and **UNITEXT** for non-Unicode UTF8 Character Set Instances and **UNICHAR**, **UNIVARCHAR**, **UNITEXT**, **NCHAR**, **NVARCHAR**, **CHAR**, **VARCHAR** and **TEXT** for Unicode UTF8 Character Set Instances

USING DATA EDITOR

Data Editor lets you edit data in your tables with any editable datatype without leaving the parent application. Data Editor lets you use your new data immediately.

CAUTION: Data Editor is a real-time editor. Changes in your data using Data Editor are permanent.

The table below describes the functions and options of the Data Editor:

Option	Description
Live	Edits data one row at a time. You must execute when you leave the row.
Batch	Edits data in multiple rows before executing.

NOTE: You can also use the Data Editor to edit date and time functions in a table.

NOTE: If you make an incorrect edit in a cell, Data Editor displays the error on the Error Tab of the ISQL Editor. Data Editor does not incorporate this error(s) in data into the table. Data Editor saves any changes in data prior to the error message.

CAUTION: Data Editor is a real-time editor. Changes in your data using Data Editor are permanent.

For more information, see:

Edit Window

ISQL Window

Tool Bar

Data Editor Filter

Date/Time Format Builder

Default Value Handling

EDITING DATE AND TIME FUNCTIONS

The Data Editor lets you edit date and time functions in a table. Data Editor uses a calendar tool to guarantee accurate input for date and time data. You can also change the display of date and time using the Date/Time Format Builder.

For more information, see:

Calendar Button

Date/Time Format Builder

DATE/TIME FORMAT BUILDER

The Date/Time Format Builder lets you customize your date/time display. The Data Editor uses this format to display your dates and times. You control how the Data Editor displays the dates and time by using predefined formats, or by customizing a format to fit your needs.

The Data Editor uses the default date/time format of your Operating System. If you do not make any global changes in Rapid SQL, the Date/Time Format Builder displays dates and times using the default formats of your operating system.If you make changes to dates and times in the Data Editor, Rapid SQL commits the changes in the format used by the database.

> **NOTE:** The changes you make using the Date/Time Format Builder do not affect the way your database stores dates and times.

EDITING THE DATE/TIME DISPLAY

You can edit the date/time display on a global, table, or column level. The table below describes the different ways you can edit your date/time format display:

Option	Description	Access
Global	Lets you make global changes to the Data Editor date display.	Options Editor
Grid	Lets you make changes to the date display of the entire Data Editor grid for that session only.	Data Editor grid
Column	Lets you make changes to the date display of a single column in the Data Editor for that session only.	Data Editor column

> **NOTE:** Date/Time formats changed on a table or column level are valid for that session only.

Editing Date/Time Globally

You can use the Options Editor to make global changes to your date/time display in the [Data Editor]. When you change the date/time format, using the Options Editor, the Data Editor displays all dates and times in the global manner. To change the date/time display for a particular session, see [Editing Grid Date/Time Format] or [Editing Column Date/Time Format].

To edit the date and time globally, do the following:

1   On the *File* menu, click *Options*.

    OR

    On the *Main* tool bar, click Options.

    Rapid SQL opens the Options Editor.

2   On the Options Editor, click the Data Editor Tab.

3   On the Data Editor Tab, click ...

    Rapid SQL opens the Date/Time Format Builder dialog box.

4   On the Date/Time Format Builder dialog box, click the Date/Time Format list, and then click the target predefined date/time format.

5   To customize the date/time format to your specifications, click Customize.

    Rapid SQL opens the Date/Time Format Builder dialog box.

6   On the Date/Time Format Builder dialog box, select the appropriate Date/Time Format Options:

Option	Description
Date/Time Format	Displays the predefined Date/Time format.
Day Format	Lets you choose the day display.
Separator	Lets you choose the display separator between the day, month, and year.
Month Format	Lets you choose the month display.
Year Format	Lets you choose the year display.
Date Order	Lets you choose the date order display.
Hour Format	Lets you choose the hour display.
Minute	Lets you choose the minute display.
Sec Format	Lets you choose the second display.
AM/PM	Lets you choose the AM/PM display.
Date/Time Order	Lets you choose the date/time order display.
Format Display	Displays the current format.
Sample	Displays a sample of the current format.

7   When you have finished selecting the Date/Time format options, click OK.

Rapid SQL accepts the date/time format changes and closes the Date/Time Format Builder dialog box.

8   On the Options Editor, select the appropriate Default Date/Time Format options:

Option	Description
Use Calendar Control as default	If selected, Rapid SQL uses the Calendar Control window.
Two-digit year system setting warning	If selected, Rapid SQL sends a warning when you use a two-digit year system setting.

9   Click OK.

Rapid SQL accepts the Default Date/Time Format changes and closes the Options Editor.

> NOTE: To use a different format for a particular session, change the date/time at the session level.

Editing Grid Date/Time

You can change the date/time display for a particular session when working in the Data Editor. The Data Editor does not maintain the format changes once you close your session. To make this display permanent, use the Editing Global Date/Time Format.

To edit the grid date and time, do the following:

1   On the Datasource Explorer, select the target table.

2   Right-click the table, and then click Edit Data.

Rapid SQL opens the Data Editor.

3   On the Data Editor tool bar, click Date/Time Format Builder.

Rapid SQL opens the Date/Time Format Builder.

4   On the Date/Time Format Builder, click the Date/Time Format list, and then click the target predefined date/time format.

5   To customize the date/time format to your specifications, click Customize.

Rapid SQL opens the Date/Time Format Builder dialog box.

6   On the Date/Time Format Builder dialog box, select the appropriate Date/Time Format Options:

Option	Description
Date/Time Format	Displays the predefined Date/Time format.
Day Format	Lets you choose the day display.
Separator	Lets you choose the display separator between the day, month, and year.

Option	Description
Month Format	Lets you choose the month display.
Year Format	Lets you choose the year display.
Date Order	Lets you choose the date order display.
Hour Format	Lets you choose the hour display.
Minute	Lets you choose the minute display.
Sec Format	Lets you choose the second display.
AM/PM	Lets you choose the AM/PM display.
Date/Time Order	Lets you choose the date/time order display.
Format Display	Displays the current format.
Sample	Displays a sample of the current format.

7   When you have finished selecting the Date/Time format options, click OK.

Rapid SQL accepts the date/time format changes and closes the Date/Time Format Builder dialog box.

- To undo changes, on the Data Editor tool bar, click Undo Change.

- To redo changes, on the Data Editor tool bar, click Redo Change.

   NOTE: Date/Time formats changed on a table level are valid for that session only.

Editing Column Date/Time

You can change the date/time display for a particular column when working in the Data Editor. The Data Editor does not maintain the format changes once you close your session. To change the format for the entire grid, see Editing Grid Date/Time Format. To make this display permanent, Editing Global Date/Time Format.

To edit the column date and time, do the following:

1   On the Datasource Explorer, select the target table.

2   Right-click the table, and click Edit Data.

Rapid SQL opens the Data Editor.

3   On the Data Editor, click the column header to select the column.

4   Right-click the column and click Format.

Rapid SQL opens the Date/Time Format Builder.

5   On the Date/Time Format Builder dialog box, click the Date/Time Format list, and then click the target predefined date/time format.

6   To customize the date/time format to your specifications, click Customize.

Rapid SQL opens the Date/Time Format Builder dialog box.

7   On the Date/Time Format Builder dialog box, select the appropriate Date/Time Format Options:

Option	Description
Date/Time Format	Displays the predefined Date/Time format.
Day Format	Lets you choose the day display.
Separator	Lets you choose the display separator between the day, month, and year.
Month Format	Lets you choose the month display.
Year Format	Lets you choose the year display.
Date Order	Lets you choose the date order display.
Hour Format	Lets you choose the hour display.
Minute	Lets you choose the minute display.
Sec Format	Lets you choose the second display.
AM/PM	Lets you choose the AM/PM display.
Date/Time Order	Lets you choose the date/time order display.
Format Display	Displays the current format.
Sample	Displays a sample of the current format.

8   When you have finished selecting the Date/Time format options, click OK.

Rapid SQL accepts the date/time format changes and closes the Date/Time Format Builder dialog box.

- To undo changes, on the Data Editor tool bar, click Undo Format.

- To redo changes, on the Data Editor tool bar, click Redo Format.

NOTE: Date/Time formats changed on a column level are valid for that session only.

DEFAULT VALUE HANDLING

When adding new records, when the transaction committing a row occurs, if no value has been entered for a column defined as having a default value, the default value is used for that column and validation is not required. The default value is not visible in the Data Editor grid until you commit the record and use the Reload Data button to refresh the grid.

In the case of tables with all columns defined as having default values, for example, you can add multiple records by repeatedly clicking the Insert New Record and Save Current Row buttons.

Page Setup

The table below describes the options and functionality on the Page Setup dialog box:

Option	Functionality
Margins	Lets you select the size of the left, right, top, and bottom margins.
Titles and Gridlines	Lets you select options.
Preview	Displays how the table will appear when printed.
Page Order	Lets you specify when to print columns and rows.
Center on Page	Lets you select how table floats on the page.

CODE GENERATION FACILITY

The Rapid SQL Code Generation Facility offers a quick way to generate DML statements for tables and views. The Code Generation Facility provides an efficient way to establish and enforce coding standards within the application by generating code with standardized formatting. The Code Generation Facility also lets you:

- Create packages and procedures for Oracle.

- Create procedures for IBM DB2 LUW, Microsoft SQL Server, and Sybase ASE.

    NOTE: The Code Generation Facility can generate procedures for IBM DB2 LUW that are based on tables but not views.

The table below describes the options and functionality on the Embarcadero Code Generator:

Option	Description
Datasource List	To change the default datasource, click the list and then click a new datasource.
	NOTE: If you are not connected to the target datasource, click the Connect button to establish a connection.
	The current datasource, and database and datasource user name where applicable, are preselected when you open Code Generation Facility.
Database List	Lets you click the list and then click a target database.
Owner List	Lets you click the list and then click the new owner.
Tables Option Button	Click this option to see a list of the tables in the scroll box. In the scroll box, click the target table to be used as a code generation base.
	NOTE: The list of objects corresponds to the selected datasource owner schema. If there are no tables in the target schema, the list will be empty.

Option	Description
Views Option Button	Click this option to see a list of the views in the scroll box. In the scroll box, click the target view to be used as a code generation base. NOTE: The list of objects corresponds to the selected datasource owner schema. If there are no views in the target schema, the list will be empty.
Select 1 or More Where Clause	In the grid, select the check boxes that correspond to the target where clauses. NOTE: Columns of primary keys are preselected.
Select 1 or More Output Columns	In the grid, select the check boxes that correspond to the target output columns. NOTE: All existing columns are preselected.
Provide and Output File Name	In the box, enter the file name in the File box or click the Browse button to locate the output file.
Open	Select to open the file after the procedure runs.
Execute Immediately	Select to execute the file immediately.
Generate	To generate an SQL statement for tables, click the Object Type list and then click the object type. NOTE: If you are generating an SQL statement for Views, you only have the Select option.
Grant Execute to	If you are a DBA or have DBA privileges, select the Users, Roles, and Groups check boxes to grant execute privileges.

Using the Code Generation Facility

1   On the *Tools* menu, click *Code Generation Facility.*

   OR

   On the *Tools* toolbar, click *Code Generation Facility.*

   Rapid SQL opens the Embarcadero Code Generator dialog box.

IMPORT DATA

It is often necessary to import data into database tables from an external source. Developers commonly need to bring sample test data into a database to assist with Use Case scenarios. These scenarios may simulate particular data retrievals where segments of the data are preferred over performing a full data load. Business Analysts often acquire spreadsheets from outside sources. It is helpful to them to load this data into tables to perform more in-depth queries and pull meaningful data to make informed decisions.

To leverage the power of Microsoft Excel, database users may prefer to pull data from the database and load it into spreadsheet. Once using manipulations like Average, Aggregate, Row Count, or simple additions or deletions have been used on the data, the need remains to import this massaged data back into the database. Additionally, once the data is placed back in the database, recurring reporting and documentation are easily accomplished.

Rapid SQL's Import Data tool eliminates the time-consuming, manual process of working with data. The Import Data Wizard lets you pull data from a text file or a Microsoft Excel spreadsheet.

Important Notes

None

Completing the Import Data Wizard

1   On the *Tools* menu, click *Import Data.*

    Rapid SQL opens the [first panel](#) of the Import Data Wizard.

2   On the first panel of the Wizard, specify the location of the file and enter the catalogue, schema (owner), and table into which the data will be imported.

3   Click Next.

4   On the second panel of the Wizard, select the character delimited type for the columns in the data file – Tab, Semi colon, Comma, Space, or Other (Custom – Tilde, Ampersand etc.)

5   You can select the option that the first row of the data contains field names.

6   Click Next.

7   On the third panel of the Wizard, you can preview and confirm the data format. You can also use the custom mapping to match columns in the table.

8   If you need to make any changes, click Back to scroll back to the appropriate panels of the wizard to make your corrections.

9   Click Finish.

    Rapid SQL generates the Insert statements that can be saved as a *.sql file for re-use across several datasources (versus simply loading the data directly into the database).

IMPORT DATA WIZARD - PANEL 1

The table below describes the options and functionality on this panel of the Import Data Wizard:

Option	Description
Specify the file to be used in this data load operation.	Lets you type the file path or browse for the file.

Option	Description
Which table do you want to load data into?	Lets you select the catalog, schema, and the table to create the insert statements.

For more information, see Import Data.

IMPORT DATA WIZARD - PANEL 2 FOR TEXT FILES

The table below describes the options and functionality on this panel of the Import Data Wizard:

Option	Description
What character delimited the columns in the data file?	Lets you select the character delimited type for the columns in the data file: Tab, Semicolon, Comma, Space, or Other (Custom – Tilde, Ampersand etc.)
First Row Contains Field Names	Select for the first row of the data to contain field names.

For more information, see Import Data.

IMPORT DATA WIZARD - PANEL 3 FOR TEXT FILES

This panel of the Import Data Wizard lets you assign column names to fields. Column names in red indicate an invalid mandatory column names. Blue columns are already used.

For more information, see Import Data.

IMPORT DATA WIZARD - PANEL 2 FOR EXCEL FILES

The table below describes the options and functionality on this panel of the Import Data Wizard:

Option	Description
The spreadsheet you have selected contains more than one worksheet. Which worksheet contains the data you wish to import?	Lets you select the worksheet to import.
Start Cell	Lets you type the first cell of data to import.
End Cell	Lets you type the last cell of data to import.
First Row Contains Field Names	Select to assign column names to fields in the grid.

For more information, see Import Data.

EMBARCADERO PRODUCTS

The Tools menu lists all installed Embarcadero Technologies products. This lets you toggle to or start another Embarcadero product.

To open the Performance Center web client, do the following:

CODE WORKBENCH

The Code Workbench is a multi-tabbed dialog that lets you enable and configure the following, time-saving ISQL editor features:

- **Code templates** - common, multi-line blocks of code you can add to a script with a couple of keystrokes and modify to the purpose of the script. For information on using this feature, see Using Code Templates.

- **Auto replace** - commonly used SQL syntax or commands that can be added to a script using a shortcut, replacement expression. For information on using this feature, see Using Auto Replace Expressions.

The Code Workbench lets you modify, add, and delete Code Template and Auto Replace resources. In addition, you can save your Code Workbench settings for the purpose of user-sharing or customization. The following topics provide detailed instructions and information on setting up these features:

- Enabling the Code Templates and Auto Replace Features

- Maintaining Code Template Definitions

- Maintaining Auto Replace Expressions

- Loading and Saving Custom Code Workbench Settings

ENABLING THE CODE TEMPLATES AND AUTO REPLACE FEATURES

In order to use the Code Templates or Auto Replace features, they must be enabled.

To enable the Code Templates or Auto Replace features

1 From the **Tools** menu, select **Code Workbench**. The **Code Workbench** dialog opens.

2 Use the following table as a guide to using the controls on the **Settings** panel:

Group	Settings and descriptions	
Auto Replace	**Enable Auto Replace**	Select this checkbox to use this feature in the ISQL editor.
	Replace substrings	If unselected, only blank-delimited occurrences of auto replace expressions are replaced when an activation event occurs. If selected, all occurrences of auto replace expressions are replaced.
		For example, consider an Auto Replace expression of **BEG** that is to be replaced with **Begin**. With **Replace substrings** enabled, **BEGBEG** would be replaced with **BeginBegin** when an activation event occurs. With **Replace substrings** disabled, no replacement would occur.
Code Templates	**Enable Code Templates**	Select this check box to use this feature in the ISQL editor.

3 Either click **OK** to dismiss the dialog or use the other tabs to configure the code templates and auto replace features.

For information on configuring the Code Template and Auto Replace features, see [Maintaining Code Template Definitions](#) and [Maintaining Auto Replace Expressions](#).

See also:

• [Using Code Templates](#)

• [Using Auto Replace Expressions](#)

MAINTAINING CODE TEMPLATE DEFINITIONS

Rapid SQL loads a default set of code templates automatically on startup. The Code Workbench lets you add templates to the currently loaded templates, edit any existing templates, or delete templates.

To modify the currently loaded code templates

1 From the **Tools** menu, select **Code Workbench**.

2 Select the **Code Templates** tab.

3   Use the following table as a guide to performing the template maintenance tasks offered on this tab:

Task	Steps
To add a new template	Click **Add** to open the **Edit Code Template expression** dialog and provide the following information:
	Shortcut - used to identify the template when the Code Assist menu is open.
	Description - displayed in the **Code Templates** tab listing for this template.
	Platform - Select a specific DBMS platform or select ALL. The template will be available in ISQL editor windows with connections to the specified platform or all platforms, accordingly.
	Template - Type the code block that will be added at the cursor when this template is selected.
To edit a template	Select a template from the list, click **Edit** to open the **Edit Code Template expression** dialog, and edit the template values as per the descriptions above.
To delete a template	Select a template from the list, click **Delete** and when prompted to confirm, click **Yes**.

> **NOTE:**   You can also save the currently loaded code templates and other Code Workbench settings to an XML file for subsequent loading. This lets you share settings among users and tailor Code Template and Auto Replace settings to particular environments or projects. For details, see Loading and Saving Custom Code Workbench Settings.

MAINTAINING AUTO REPLACE EXPRESSIONS

Rapid SQL is packaged with a set of default Auto Replace definitions, stored in the registry and loaded automatically on startup. The Code Workbench lets you add new definitions, edit any existing definitions, or delete definitions from the currently loaded set.

To modify the currently loaded expressions

1   From the **Tools** menu, select **Code Workbench**.

2   Select the **Auto Replace** tab.

3   Use the following table as a guide to performing the template maintenance tasks offered on this tab:

Task	Steps
To add a new expression	Click **Add** to open the **Edit Auto Replace Expression** dialog and provide the following information:
	Expression - the shortcut that will be replaced.
	Activation - Type one or more individual keys that are to trigger replacement in the ISQL editor. Valid activation keystrokes are SPACE, TAB, and RETURN. As you type, the following representations are shown in the **Activation** box: **\s** represents SPACE, **\t** represents TAB, and **\n** represents RETURN (newline).
	Replace With - Type the command or SQL syntax that will be replaced when one of the activation events occurs.
To edit an expression	Select an expression from the list, click **Edit** to open the **Edit Auto Replace Expression** dialog, and edit the expression values as per the descriptions above.
To create an expression based on an existing expression	Select an expression from the list, click **Clone** to open the **Edit Auto Replace Expression** dialog, prepopulated with values of the existing expression. Edit the expression values as per the descriptions above.
To delete an expression	Select an expression from the list, click **Delete** and when prompted to confirm, click **Yes**.

> **NOTE:** You can also save the currently loaded Auto Replace expressions and other Code Workbench settings to an XML file and subsequently load those settings. This lets you share settings among users and tailor Auto Replace and Code Template settings to particular environments or projects. For details, see <u>Loading and Saving Custom Code Workbench Settings</u>.

LOADING AND SAVING CUSTOM CODE WORKBENCH SETTINGS

At each startup, Rapid SQL loads a default set of code templates and auto replace definitions stored in the registry. The enabled/disabled status of the two features is retained on shutdown.

After changing the enabled/disabled status of the Code Templates or Auto Replace features, or modifying the associated templates or replacement expressions, you can save the status and currently loaded resources to an XML file. The ability to save, load, and restore settings means you can:

- Share Code Workbench settings among users

- Create custom Code Workbench settings tailored to DBMS platforms or coding projects

To make use of Code Workbench fie options

1 From the **Tools** menu, select **Code Workbench**.

2 Use the following table as a guide to performing the Code Workbench file option tasks:

Task	Steps
To save the current Code Workbench settings to an XML file:	Click **Export Settings** and use the **Save As** dialog to select a location and provide a name for the exported file.
To replace the currently loaded Code Workbench settings with a set of saved settings:	Click **Import Settings** and use the **Open** dialog to locate and select the settings file to be loaded.
To restore the default Code Workbench settings from the registry:	Click **Restore Settings**.

See also:

- [Maintaining Code Template Definitions](#)

- [Maintaining Auto Replace Expressions](#)

CODE ANALYST

The Code Analyst is a tool to identify time-consuming lines of code. Code Analyst lets you:

- Perform detailed response time analysis on the execution of procedures and foreign keys.

- Benchmark the execution of one or more procedures or functions to determine exactly what code objects and lines of code are taking the longest to run.

- Save response time metrics and perform intelligent compares against current execution times so you can determine deviations from previous acceptable response times.

>   **TIP:**   You can set Code Analyst options in the [Code Analyst Options](#).

>   **NOTE:**   Availability of this feature depends on your Rapid SQL licensing. For more information, see [Licensing](#).

Important Notes

- For DB2, before profiling with Code Analyst, [Compile](#) all procedures with the debugging option selected.

- For Oracle, when using the Oracle Debugger, [Compile](#) all procedures with the debugging option selected before profiling with Code Analyst.

Common Tasks

[Creating a Code Analyst Session](#)

CODE ANALYST DBMS NOTES

Code Analyst is available for:

- Microsoft SQL Server 7 or later

- Oracle 7 or later

- IBM DB2 LUW 8

- Sybase ASE 12.0.0.3 or later

Rapid SQL utilizes debugger technology to capture the data for each line of executed code. For Oracle, you can use the debugger or using Oracle's supplied DBMS_Profiler package.

> **TIP:** For Oracle, you can specify to use the debugger or the DBMS_Profiler package on the Code Analyst Options.

The Code Analyst will step through each line of code, stopping to record data for those lines of code onto which a breakpoint can be issued. Some debuggers cannot capture time metrics for all lines of a stored procedure or function.

Procedures and functions that contain looping constructs will require more time to run. The additional amount of time needed to run is proportional to the number of iterations in the loop.

For more information, see:

[IBM DB2 LUW 8i Data Captured by Code Analyst](#)

[Microsoft SQL Server Data Captured by Code Analyst](#)

[Oracle Data Captured by Code Analyst](#)

[Sybase ASE Data Captured by Code Analyst](#)

IBM DB2 LUW 8i Data Captured by Code Analyst

Code Analyst utilizes the IBM Debugger when capturing time data.

The debugger is verified to run on IBM DB2 LUW version 7.2 and up. There is a known issue running version 7.2 with Fixpack 9.

DB2 has documented limitations on lines of code can be profiled.

The following are SQL statements that are NOT valid break point lines:

BEGIN

BEGIN

BEGIN NOT ATOMIC

BEGIN ATOMIC

CLOSE CURSOR

DECLARE cursor WITH RETURN FOR <sql statement>

DECLARE , var without default

DECLARE CONDITION (CONDITION) FOR SQLSTATE (VALUE) "..."

DECLARE CONTINUE HANDLER

DECLARE CURSOR

DECLARE EXIT HANDLER

DECLARE RESULT_SET_LOCATOR [VARYING]

DECLARE SQLSTATE

DECLARE SQLCODE (unless there is a default)

DECLARE UNDO HANDLER (unless they are entered)

DO

ELSE

END

END CASE

END IF

END FOR

END REPEAT

END WHILE

ITERATE

LEAVE

LOOP

OPEN CURSOR

REPEAT (as a keyword alone)

RESIGNAL

SIGNAL

THEN

labels, e.g. P1:

 NOTE: Code containing these statements will not have times associated with them.

For more information, see [Code Analyst DBMS Notes](#).

MICROSOFT SQL SERVER DATA CAPTURED BY CODE ANALYST

In order to execute a Code Analyst session against a Microsoft SQL Server database, the SQL Server debugger must be installed and functioning properly. Please refer to [Embarcadero SQL Debugger for Microsoft SQL Server](#) for details concerning set up.

Related Information

[Code Analyst DBMS Notes](#)

ORACLE DATA CAPTURED BY CODE ANALYST

When using the PL/SQL Profiler, Oracle has documented an issue regarding extremely large times being returned by the profiler. The times are sometimes hundred times larger than the actual run time of the stored procedure or function. Oracle documents that this is a vendor/os problem rather than an Oracle problem, because the RDTSC instruction is reporting wrong time stamp counter. They indicate that they have seen this problem on some INTEL Pentium processors.

Related Information

[Code Analyst DBMS Notes](#)

SYBASE ASE DATA CAPTURED BY CODE ANALYST

Sybase has documented a problem with their debugger API. The problem involves reporting the wrong line number through the debugger. Because of this bug, Code Analyst may report back data for blank lines or lines that contain comments. Sybase has fixed this problem release 12.5.2 of the database. All procedures affected must be dropped and recreated in order to correct the problem.

Related Information

[Code Analyst DBMS Notes](#)

CODE ANALYST REQUIREMENTS

Debuggers

The Code Analyst uses fully configured Embarcadero SQL Debuggers to profile, and therefore availability of this feature depends on your Rapid SQL licensing. For more information, see the following topics:

- Licensing

- Rapid SQL Add-On Tools

Using Code Analyst with the Oracle Profiler

Oracle users have the option of either using the Oracle Debugger or the Oracle Profiler with Code Analyst to capture statistics. To use Code Analyst with the profiler option, Oracle's profiler package must be installed. The install is user specific, so it must be installed by each user wishing to use Code Analyst. To install the package, users can invoke the Rapid SQL PL/SQL Profiler.

> **TIP:** You can set profiler options in the Code Analyst Options. You can specify that Code Analyst display the actual run time on the database, and does not include the time it takes to get to the server.

Privileges

For Oracle, SYS privileges are required to install the Code Analyst tables. If you do not have SYS privileges, ask your Server Administrator to log into the Oracle datasource as SYSDBA, and then open Code Analyst to install the tables.

During install, the following privileges are set for the Code Analyst tables.

- DB2 – Permissions are granted to the Public Group

- Oracle – Permissions are granted to the Public group.

- Microsoft SQL Server – Permissions are granted to the Public role.

- Sybase – Permissions are granted to the Public group.

All users can use the Code Analyst but each user will only see their own run ids. Users need to belong to the public group.

> **TIP:** You can check this/modify privileges in the Users Editor.

INSTALLING CODE ANALYST

To install Code Analyst, do the following:

1   On the Tools menu, select *Code Analyst*.

2   In Select the database you would like to install the tables on, select a database.

3 For IBM DB2 LUW for Open Systems, in select the tablespace you would like to install the tables on, select a tablespace.

4 For IBM DB2 LUW for Open Systems, in select the schema you would like to install the tables on, select a schema. The default is EMBTCA schema.

5 In Select the filegroup you would like to install the tables to, select a filegroup.

Code Analyst installs the following repository tables in the repository:

- EMBT_CODE_ANA_RUNS - Holds all the code analyst sessions created by users.

- EMBT_CODE_ANA_UNITS - Holds all the objects to be run for all.

- EMBT_CODE_ANA_PARAMS - Contains all the parameters for the objects that were run.

- EMBT_CODE_ANA_DATA - Contains the run data and is used to populate all the charts and statistics.

- EMBT_CODE_ANA_VERSION - Contains the version number of code analyst.

Code Analyst opens to the Run Summary tab.

6 Create a session using the [Creating a Code Analyst Session](#).

UNINSTALLING CODE ANALYST

The Uninstall functionality lets you uninstall Code Analyst from the server.

> **NOTE:** To uninstall a repository table, you need create table and grant privileges. Generally, you need sysadmin privileges.

1 On the Tools menu, select *Code Analyst*.

2 Select a session or object, and then select Uninstall.

Code Analyst removes the repository tables in the repository:

- EMBT_CODE_ANA_RUNS - Holds all the code analyst sessions created by users.

- EMBT_CODE_ANA_UNITS - Holds all the objects to be run for all.

- EMBT_CODE_ANA_PARAMS - Contains all the parameters for the objects that were run.

- EMBT_CODE_ANA_DATA - Contains the run data and is used to populate all the charts and statistics.

- EMBT_CODE_ANA_VERSION - Contains the version number of code analyst.

CODE ANALYST PRODUCT DESIGN

Code Analyst performs detailed response time analysis. Code Analyst steps through each line of code and profiles those lines of code that the debugger or profiler can capture time metrics for.

> NOTE: Some debuggers do not capture time metrics for all lines of a procedure or function. For more information, see [Code Analyst DBMS Notes](#).

After capturing the time metrics, Code Analyst displays the data in an easy-to-read format on the tabs.

The Code Analyst is comprised of the following tabs:

Tab	Option	Description
Run Summary	Session	Lets you select the run session(s).
	Session	Displays the name of the session as created by the user.
	Run ID	Displays the run ID for the run(s). This number is system generated.
	Run Date	Displays the time and date of the session.
	Total Profile Time (ms)	Displays the total time taken for the profiled code to execute. This time is limited to the lines of code that are profiled. Overhead is not included in this calculation.
	Total Analysis Time	Displays the total time taken for the session to complete, including all overhead time needed to analyze the procedure or function.
	Scheduler	Displays the scheduler used to schedule the session. This information displays until the scheduled job has been run.
Run Detail	Session	Lets you select the object execution session.
	Run	Lets you select the object execution.
	Unit Type	Lets you select the object type for the object execution.
	Unit Owner	Lets you select the object owner for the object execution.
	Unit Database	Lets you select the object database for the object execution.
	Time Unit	Lets you specify the time unit for the Unit Execution graph.
	Unit Owner	Displays the owner of the procedure or function.
	Unit Name	Displays the name of the procedure or function.
	Unit Type	Displays the types of captured objects, including Anonymous Block, Function, Package Body, and Procedure for Oracle databases. Also displays SQL Statement and Procedure for the other platforms.
	Unit Database	Displays the database on which the object is stored.
	Total Profiled Time	Displays the total time taken for the profiled code to execute.
	% of Profiled Time	Displays the percentage of the Total Profiled Time for the run that this unit accounts for.
Comparison	Base Run	Lets you select the earlier object execution.

Tab	Option	Description
	New Run	Lets you select the later object execution.
	Unit Owner	Displays the owner of the procedure or function.
	Unit Name	Displays the name of the procedure or function.
	Unit Type	Displays the types of captured objects, including Anonymous Block, Function, Package Body, and Procedure for Oracle databases. Also displays SQL Statement and Procedure for the other platforms.
	Unit Database	Displays the object database for the object execution.
	Time Diff	Displays the time difference in milliseconds between the base run and the new run.
	New Profiled Time	Displays the profiled time of the new run.
	Base Profiled Time	Displays the profiled time of the base run.
Unit Summary	Unit Owner	Lets you select the object owner for the session(s).
	Unit Name	Lets you select the object name for the session(s).
	Unit Database	Lets you select the object database for the session(s).
	Number of Top Runs	Lets you specify the number of top object executions to display in the Top 5 Runs graph and select the unit of time for the Unit Time graph.
	Session	Displays the name of the session as created by the user.
	Run ID	Displays the unique id for the Run. This number is system generated.
	Run Date	Displays the time and date of the session.
	Total Analysis Time	Displays the total time taken for the session to complete, including all overhead time needed to analyze the procedure or function.
	Total Profiled Time	Displays the total time taken for the profiled code to execute.
	Unit Profiled Time	Displays the unit time for the session(s).
	% of Profiled Time	Displays the percentage of the Total Profiled Time for the run that this unit accounts for.
	% of Run Time	Displays the percentage of object execution time for the session(s).
Unit Detail	Session	Lets you select the session.
	Run	Lets you select the object execution.
	Unit Name	Lets you select the object name for the session.
	Number of Top Lines	Lets you specify the number of top lines in the Top 5 Lines Execution Time graph and select the total time units.
	Percentage Calculation	Lets you specify total object execution time or object run time.
	Calls	Displays the number of times the line was executed.
	Total Time	Displays the total time the line was executed.
	% of Total Profiled	Displays the percentage of the Total Profiled Time that this line of code was responsible for.

Tab	Option	Description
	Avg Time	Displays the average profiled time for this line.
	Min Time	Displays the minimum recorded time for execution of this line.
	Max Time	Displays the maximum recorded time for execution of this line.
	Dependency	Displays the UNIT_NUMBER of the dependency object that was called by that line. Lets you right-click and quickly go to that UNIT_NUMBER to see its Unit detail information.
	Source	Displays the object's SQL source code.

Common Tasks

Creating a Code Analyst Session

Identifying and Fixing Bottlenecks Using Code Analyst

Comparing Code Analyst Sessions

Cloning a Code Analyst Session

Deleting a Code Analyst Session

Stopping a Code Analyst Session Execution

Executing a Code Analyst Session

Scheduling a Code Analyst Session

Unscheduling a Code Analyst Session

Refreshing a Code Analyst Session

Saving Results in Code Analyst

Printing Results in Code Analyst

Viewing Run Details in Code Analyst

Viewing Unit Summary Information in Code Analyst

Viewing Unit Details in Code Analyst

Setting View Options for the Unit Detail Tab in Code Analyst

Extracting SQL Text in Code Analyst

Executing SQL in Code Analyst

CODE ANALYST TUTORIAL

The following tutorial guides you through the process of using the Code Analyst.

Creating a Code Analyst Session

1   On the Tools menu, select Code Analyst.

    Initially, Code Analyst installs the repository tables. For more information, see <u>Installing Code Analyst</u>. Then Rapid SQL opens the Code Analyst to the Run Summary tab.

2   On the Code Analyst Tools toolbar, click the Create New Collection button.

    Rapid SQL opens the first panel of the Code Analyst Wizard.

3   Select the individual object or group of objects to analyze. In this example, an individual stored procedure (CREATE_ADMISSION2) is selected.

    TIP: Code Analyst does not let you select objects that do not have stored procedures.

4   Click Next.

    If the object(s) selected to be analyzed requires parameters, the second panel of the wizard prompts you to enter the parameters.

5   Double-click the object to set the parameters.

6   For IBM DB2 LUW for Open Systems and Oracle, the Compile button opens the Confirm Compile dialog box that lets you compile the objects to ensure that the Code Analyst can capture the time metrics.

7   Click Finish.

    Code Analyst displays a message that the Code Analyst will run longer than the actual code. Then Code Analyst analyzes the objects, using the Embarcadero SQL Debugger to profile and then opens the Run Detail tab.

    TIP: You can select the "Please do not show me this dialog again" option in the dialog box or set the option on the <u>Code Analyst Options</u>.

Identifying and Fixing Bottlenecks Using Code Analyst

The Run Detail tab displays the total time for the objects being analyzed. The tab information may be enough to identify the potential bottleneck.

1   To view more detailed information, double-click the Unit Name.

    Code Analyst opens to the Unit Detail tab.

The Unit Detail Tab displays the object code and other information related to the individual lines of code. You can determine which line of code is taking too long and why. The Unit Detail Tab is where you troubleshoot, and then resolve the problem in the Modifying objects using editors.

2   In Rapid SQL, open the object editor, and then modify the code on the Definition tab.

3   Click Alter.

4   In Code Analyst, on the Unit Detail Tab, click Execute.

Comparing Code Analyst Sessions

1 Click the Comparison tab.

Code Analyst has a Comparison facility to allow quick compares of two object executions, showing the base time and the new time, as well as the time differences. The Comparison tab lets you compare which of the two procedures or functions ran faster.

2 Examine the Time Diff which indicates improvement to code.

3 If necessary, continue to modify the code on the Definition tab of the object editor, and then press Alter.

4 In Code Analyst, on the Unit Detail Tab, click Execute.

5 Examine the Time Diff until the bottleneck is solved.

Common Tasks

Creating a Code Analyst Session

Identifying and Fixing Bottlenecks Using Code Analyst

Comparing Code Analyst Sessions

Cloning a Code Analyst Session

USING THE CODE ANALYST

When working with database code stored in database objects, it is sometimes difficult to pinpoint bottlenecks within the code. When situations like this arise, Code Analyst can assist in identifying the trouble spots. Code Analyst can be used to analyze one object or a group of objects. You select one or multiple objects, execute them, view the results, and save those results for later viewing or comparing.

Common Tasks

[Refreshing a Code Analyst Session](#)

[Saving Results in Code Analyst](#)

[Printing Results in Code Analyst](#)

[Viewing Run Details in Code Analyst](#)

[Viewing Unit Summary Information in Code Analyst](#)

[Viewing Unit Details in Code Analyst](#)

[Setting View Options for the Unit Detail Tab in Code Analyst](#)

[Extracting SQL Text in Code Analyst](#)

[Executing SQL in Code Analyst](#)

CREATING A CODE ANALYST SESSION

The Code Analyst Wizard creates a new Code Analysis session that creates data for the Code Analyst tabs:

1   On the Tools menu, select Code Analyst.

2   On the Code Analyst Tools toolbar, click Create New Collection.

    Rapid SQL opens the first panel of the Code Analyst Wizard.

3   Select the individual object or group of objects to analyze.

4   Click Next.

    If the object(s) selected to be analyzed requires parameters, the second panel of the wizard prompts you to enter the parameters.

5   Double-click the object to set the parameters.

6   For IBM DB2 LUW for Open Systems and Oracle, the Compile button opens the Confirm Compile dialog box that lets you compile the objects to ensure that the Code Analyst can capture the time metrics.

7   Click Finish.

    Code Analyst displays a message that the Code Analyst will run longer than the actual code. Then Code Analyst analyzes the objects, using the Embarcadero SQL Debugger to profile and then opens the Run Detail tab.

> **TIP:**   You can also select the "Please do not show me this dialog again" option in the dialog box or set the option on the [Code Analyst Options](#).

For more information, see [Code Analyst Wizard](#).

CODE ANALYST WIZARD

The table below describes the options and functionality of the Code Analyst Wizard:

Option	Description
Session Name	Lets you type a name.
Select by Owner	Code Analyst Wizard queries the database to get the list of procedures and functions and lets you select objects to retrieve Codes for. Select to display available objects by owner, and then select the database(s).
Select by Object	Code Analyst Wizard queries the database to get the list of procedures and functions and lets you select objects to retrieve Codes for. Select to display available objects by object, and then select the object(s).
Object Name	Double-click each object to specify the input parameters. Specify which object executes first by clicking the Up and Down buttons.
Compile	IBM DB2 LUW FOR OPEN SYSTEMS AND ORACLE ONLY: Opens the Confirm Compile dialog box that lets you compile the objects to ensure that the Code Analyst can capture the time metrics.
Schedule	Opens the Select Scheduler dialog box or opens scheduling application.
Finish	Code Analyst analyzes the code.

For more information, see: <u>Creating a Code Analyst Session</u>.

IDENTIFYING AND FIXING BOTTLENECKS USING CODE ANALYST

The Unit Detail Tab displays the object code and other information related to the individual lines of code. You can identify time-consuming lines of code in the Unit Detail Tab. The Unit Detail Tab is where you troubleshoot, and then resolve the problem in the <u>Modifying objects using editors</u>.

1   On the Tools menu, select Code Analyst.

2   Click the Unit Detail Tab.

    Percent of Run Time displays the percentage of object execution time for the session(s).

3   Identify an object that contains time-consuming code.

4   In Rapid SQL, open the object editor, and then modify the code on the Definition tab.

5   Click Alter.

6   In Code Analyst, on the Unit Detail Tab, click Execute.

7   Click the Comparison tab.

The Comparison Tab lets you compare times of the objects in two different object executions to determine which run was more efficient. The Comparison Tab displays the base time and the new time, as well as the time differences. The Comparison tab lets you compare which of the two procedures or functions ran faster.

8   Examine the Time Diff which indicates improvement to code.

9   If necessary, continue to modify the code on the Definition tab of the object editor, and then press Alter.

10   Create new Code Analyst sessions and examine the Time Diff until the bottleneck is solved.

For more information, see:

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

COMPARING CODE ANALYST SESSIONS

The Comparison Tab lets you compare times of the objects in two different object executions to determine which run was more efficient. The Comparison Tab displays the base time and the new time, as well as the time differences. The Comparison tab lets you compare which of the two procedures or functions ran faster.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, right-click the sessions, and then select Compare.

3   Examine the Time Diff which indicates improvement to code.

For more information, see:

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

CLONING A CODE ANALYST SESSION

The Clone Collection functionality lets you clone an existing Code Analyst session using the Code Analyst Wizard. Clone lets you reset the parameters or the order of the objects in the session without creating a new session.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to clone.

3   On the Code Analyst Tools toolbar, click Clone Collection.

   Rapid SQL opens the [first panel of the Code Analyst Wizard](#).

For more information, see:

Code Analyst Product Design

Using the Code Analyst

DELETING A CODE ANALYST SESSION

The Delete Collection functionality lets you delete the selected Code Analyst session.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to delete.

3   On the Code Analyst Tools toolbar, click Delete Collection.

    Code Analyst deletes the session.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

STOPPING A CODE ANALYST SESSION EXECUTION

The Stop Execution kills the execution of the selected collection.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to kill.

3   On the Code Analyst Tools toolbar, click Stop Execution.

    Code Analyst kills the execution.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

EXECUTING A CODE ANALYST SESSION

The Execute Collection functionality extracts the SQL text and then executes the code.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to execute.

3   On the Code Analyst Tools toolbar, click Execute Collection.

    Code analyst extracts and executes the SQL.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

SCHEDULING A CODE ANALYST SESSION

The Schedule Session functionality lets you schedule the session for a future run.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to schedule.

3   On the Code Analyst Tools toolbar, click Schedule Session.

    Code Analyst opens the default scheduler.

For more information, see:

Scheduling

Code Analyst Product Design

Using the Code Analyst

UNSCHEDULING A CODE ANALYST SESSION

The Delete Session functionality lets you remove the session from a schedule.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to unschedule.

3   On the Code Analyst Tools toolbar, click Delete Session.

For more information, see:

Scheduling

Code Analyst Product Design

Using the Code Analyst

REFRESHING A CODE ANALYST SESSION

The Refresh Data functionality refreshes the data.

1   On the Tools menu, select Code Analyst.

2   On the tab, on the Code Analyst Tools toolbar, click Refresh Data.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

SAVING RESULTS IN CODE ANALYST

The Save functionality lets you save results for later viewing or comparing.

1   On the Tools menu, select Code Analyst.

2   On the tab, right-click the session or unit, and then select Save.

    Code Analyst opens the Save Results dialog box.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

PRINTING RESULTS IN CODE ANALYST

The Print functionality lets you print results for later viewing or comparing.

1   On the Tools menu, select Code Analyst.

2   On the tab, right-click the session or unit, and then select Print.

    Code Analyst opens the Print Results dialog box.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

VIEWING RUN DETAILS IN CODE ANALYST

The Run Detail tab displays the total time for the objects being analyzed. The tab information may be enough to identify the potential bottleneck.

To open the Run Details Tab in Code Analyst, do the following:

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, right-click the session, and then select Run Detail.

    OR

3   On the Unit Summary tab, right-click the session, and then select Run Detail.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

VIEWING UNIT SUMMARY INFORMATION IN CODE ANALYST

The Unit Summary Tab in Code Analyst displays the individual runs for a session.

To open the Unit Summary Tab in Code Analyst, do the following:

1   On the Tools menu, select Code Analyst.

2   On the Comparison tab, right-click the session, and then select Unit Summary.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

VIEWING UNIT DETAILS IN CODE ANALYST

The Unit Detail Tab displays the object code and other information related to the individual lines of code. You can identify time-consuming lines of code in the Unit Detail Tab. The Unit Detail Tab is where you troubleshoot, and then resolve the problem in the Modifying objects using editors.

To open the Unit Details tab in Code Analyst, do the following:

1   On the Tools menu, select Code Analyst.

2   On the Unit Summary tab, right-click the session, and then select Unit Detail.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

SETTING VIEW OPTIONS FOR THE UNIT DETAIL TAB IN CODE ANALYST

The table below describes the options on the shortcut menu for the Unit Details Tab in Code Analyst:

Option	Description
Dependency Details	Displays dependency details.
Show Only Hit Lines	Displays only those lines with time metrics.
Show Only Missed Lines	Displays only those lines without time metrics.
Show All Lines	Resets the view to show all lines.

Option	Description
Advanced View	Displays the default view.
Normal View	Displays a limited number of data columns.

EXTRACTING SQL TEXT IN CODE ANALYST

The Extract SQL Text functionality extracts SQL text to an ISQL window.

1   On the Tools menu, select Code Analyst.

2   On the Unit Detail tab, right-click the session, and then select Extract SQL Text.

   Code analyst extracts the SQL text to an ISQL window.

For more information, see:

Extract

Code Analyst Product Design

Using the Code Analyst

EXECUTING SQL IN CODE ANALYST

The Execute SQL functionality extracts SQL text to an ISQL window, and then executes the code.

1   On the Tools menu, select Code Analyst.

2   On the Unit Detail tab, right-click the session, and then select Execute SQL.

   Code analyst extracts the SQL text to an ISQL window and executes the code.

For more information, see:

Execute

Code Analyst Product Design

Using the Code Analyst

RAPID SQL ADD-ON TOOLS

Rapid SQL includes the following add-on tools:

[Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows](#)

[Embarcadero SQL Debugger for Microsoft SQL Server](#)

[Embarcadero SQL Debugger for Oracle](#)

[Embarcadero SQL Debugger for Sybase ASE](#)

[Rapid SQL PL/SQL Profiler](#)

[Code Analyst](#)

EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows lets you locate and fix bugs in procedures and triggers for IBM DB2 LUW for Linux, Unix, and Windows version 7.2 or later. Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows lets you debug triggers by debugging the procedures that call them.

> **NOTE:** Availability of this feature depends on your Rapid SQL licensing. For more information, see [Licensing](#).

> **TIP:** For Rapid SQL, [Code Analyst](#) is a tool to identify time-consuming lines of code. Code Analyst lets you perform detailed response time analysis on the execution of Procedures and Functions.

The table below describes the sections of this chapter:

Section	Description
[Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Features](#)	This section describes how Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows lets you identify problems within your code.
[Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Interface](#)	This section describes the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows graphical interface that includes an editor window and four debug view windows.
[Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality](#)	This section describes the functionality on the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows.

Section	Description
Using the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows	This section describes how to run a debug session.

EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS FEATURES

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows lets you identify problems within your code. The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows lets you:

- Interactively step through the flow of script execution.

- Examine the value of variables.

- Solve logical problems with your script design.

> NOTE: The Debugger is available on the Rapid SQL main menu, the Procedures window, the DDL Editor and ISQL windows.

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows offers fundamental debugging features and options to fine tune debugging. The table below describes these features:

Debugging Feature	Description
Step Into	Lets you execute each instruction step-by-step and step inside a stored object.
Step Out	Lets you stop stepping through the current object and execute the remainder of the script. This option is only active when the pointer indicates a child dependent instruction.
Step Over	Lets you execute the current instruction without stepping into any child dependents.
Breakpoints	Lets you specify positions in a program where the debugger stops execution.

To set specific Debugger values on Rapid SQL's Options Editor, see Debugger Options.

EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS REQUIREMENTS

Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows requires proper configuration of the server and client.

For more information, see:

[Prerequisites for Adding and Compiling Procedures](#)

[Configuring the IBM DB2 LUW for Linux, Unix, and Windows Server for Procedures](#)

[Prerequisites for Debugging Procedures](#)

PREREQUISITES FOR ADDING AND COMPILING STORED PROCEDURES
The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows requires the following products and components.

Client
- IBM DB2 LUW for Linux, Unix, and Windows 7.2 or later

- DB2 Application Development Client

- DB2 Administration Client

- Communications Protocols

- Stored Procedure Builder

- Applications Development Interfaces

- System Bind Files

- DB2 Connect Server Support

- Documentation

- Base DB2 LUW for Windows/Unix Support

- Administration and Configuration Tools

Server
- IBM DB2 LUW for Linux, Unix, and Windows 7.2 or later

- DB2 Enterprise Edition

- Communications Protocols

- Stored Procedure Builder

- Applications Development Interfaces

- System Bind Files

- DB2 Connect Server Support

- Documentation

- Base DB2 LUW for Windows/Unix Support

- Administration and Configuration Tools

- Microsoft Visual Studio, Visual C++

   NOTE: The server must have a local directory structure and file C:\program files\sqllib\function\routine\sr_cpath.bat. This file is installed with IBM DB2 LUW 7.2 and includes the C compiler options needed to compile the procedure on the server. If it is not found, install the IBM DB2 LUW 7.2 Administration and Configuration Tools option on the server.

CONFIGURING THE IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS SERVER FOR PROCEDURES

Rapid SQL lets you create procedures on the targeted server using Rapid SQL.

To create or run any procedure, set up the configuration environment and enable the C compiler options on the server.

To configure your server, do the following:

1   Open a DB2 Command Window, and then type:

```
DB2set DB2_SQLROUTINE_COMPILER_PATH="C:\program
files\sqllib\function\routine\sr_cpath.bat"
```

   DB2 sets the DB2_SQLROUTINE_COMPILER_PATH DB2 registry variable to call the required initialization script for the C compiler on the server.

To enable the C compiler options on your server:

1   Open the file C:\program files\sqllib\function\routine\sr_cpath.bat.

2   Remove the REM (remarks) prefix on the lines that match the version of Visual Studio that is installed on the server. VCV6 = version 6.0 and VCV5 = version 5.0.

   NOTE: Only remove the REM prefix on the lines that apply to your installation of Visual Studio

3   Restart the DB2 services on the server.

PREREQUISITES FOR DEBUGGING PROCEDURES

To enable debugging on the server, do the following:

1   Open a DB2 Command window and type:

```
Db2set DB2ROUTINE_DEBUG=ON
```
   NOTE: Client must have a licensed or evaluation copy of the Embarcadero LUW SQL Debugger.

EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS OPTIONS

You can specify debugger options from the Debug Tab of Rapid SQL's Options editor. The Debug Tab of the Options Editor lets you set the duration of your debug initialization and debug session, enable DBMS output, and refresh dependencies.

Setting Debugger Options

To set debugger options, do the following:

1   On the *File* menu, click *Options.*

OR

On the *Main* toolbar, click Options.

Rapid SQL opens the Options Editor.

2   Specify debugger options. The table below describes the options available:

Option	Description	Default
Initialization Timeout (seconds)	Specifies the number of seconds Rapid SQL tries to initialize the debugger. If it cannot initialize the debugger in the specified time, a message displays in the Debug Output window.	60
Debug Session Timeout (seconds)	Specifies, in seconds, the length of your debug session.	7200
Enable DBMS Output	Toggles the print output. Enable this option if you use dbms_output.put_line calls in your procedures and you want these lines displayed.	Selected
Refresh Dependencies for each run	Refreshes dependencies for each run. This potentially time-consuming process is useful if the target procedure has rapidly varying dependencies that can require updating during the debugging process.	Cleared

3   Click Close.

Rapid SQL closes the Options Editor.

For more information, see [Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Features](#).

EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS INTERFACE

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows includes an editor window and four debug view windows. When you open a *debug session*, Rapid SQL extracts the code for the object into a DDL Editor and opens four debug view windows at the bottom of the screen. The four debug view windows are optional, dockable windows designed to let you debug your script.

TIP: All Embarcadero debuggers display Performance Metrics that let you measure the execution time of each statement in the debug session.

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows's includes five windows:

1   [DDL Editor window](#)

2   [Watch window](#)

3   [Variables window](#)

4   [Call Stack window](#)

5   [Dependency Tree window](#)

Working with T-SQL Debugger Windows

Rapid SQL lets you resize, move, dock and float the following windows:

- [Watch window](#)

- [Variables window](#)

- [Call Stack window](#)

- [Dependency Tree window](#)

1   To resize the target window, click its frame and drag it.

Rapid SQL resizes the window.

2   To move and dock the target window, click its grab bar and drag it.

Rapid SQL moves the window to its new location and docks it with surrounding windows.

3   To float the target window, press Shift, then click its grab bar and drag it.

Rapid SQL frames the window in its own floating frame and moves the window to its new location.

DDL EDITOR WINDOW FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

The DDL Editor displays your code in read-only format. When you start debugging, the SQL Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows extracts your code into a DDL Editor. The DDL Editor uses the default Rapid SQL syntax coloring.

For more information, see [Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Interface](#).

WATCH WINDOW FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR

LINUX, UNIX, AND WINDOWS

The Watch window displays the watch variables for the database object you are debugging. The Watch window also lets you specify variables you want to evaluate or modify while debugging your program.

For example, to check what happens when a variable (x) has a value of 100, you can double-click the variable in the DDL Editor, drag it into the Watch Window, and change the value to 100. When you execute the script, the Debugger uses the value x =100. This window is only visible when the T-SQL Debugger is active.

> **NOTE:** Until you step at least once into a script, variables are not defined. Therefore, step at least once before dragging or typing a local variable in the Watch Window.

> **NOTE:** When you exit a debug session and reenter it, the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows retains any watch variables or breakpoints you have set.

Opening and Closing the Watch Window

To open and close the Watch Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Watch.*

    OR

    Press ALT+3.

Setting a Watch Variable

To set a Watch Variable, do the following:

1   In the *DDL Editor,* double-click the target variable and drag it to the *Watch window*.

2   In the *Watch window,* change the value of the variable.

3   On the *DDL Editor*, click Debug.

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows executes the script using the new variable.

Removing a Watch Variable

To remove a Watch variable, do the following:

1   In the *Watch window*, click the target variable and press *DELETE*.

For more information, see [Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Interface](#).

VARIABLES WINDOW FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR

LINUX, UNIX, AND WINDOWS

The Variables window displays the local variables and their current values during script execution.

> **NOTE:** You cannot edit the variables in the Variables window.

If the DDL Editor displays an external database object, and that object is a dependent of the object you are debugging, then the Variables Window automatically refreshes and displays the variables for that particular object. The Variables Window is only visible when the Debugger is active.

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows also lets you monitor your variables while debugging.

Opening and Closing the Variables Window

To open and close the Variables Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Variable.*

    OR

    Press ALT+4.

Monitoring Variables

To monitor the values of your variables while debugging, do the following:

1   In the SQL Editor, hold the pointer over the target variable.

    Rapid SQL opens a ScreenTip displaying the current value of that variable.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Interface.

CALL STACK WINDOW FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

The Call Stack window displays the stack of currently active calls. The Call Stack Window is only visible when the Debugger is active.

Opening and Closing the Call Stack Window

To open and close the Call Stack Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Call Stack.*

    OR

    Press ALT+5.

Using the Call Stack Window

To display a line of code that references the call in the DDL Editor, do the following:

1 In the Call Stack window, double-click the target line.

   In the DDL Editor, Rapid SQL displays a green arrow on the line of the referenced call.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Interface.

DEPENDENCY TREE WINDOW FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

The Dependency Tree window displays any external database objects the script accesses. Rapid SQL displays these database objects in a hierarchical tree, with the child objects as database objects accessed by the parent objects. You can use this window to display the code for a dependent database object in the *DDL Editor* window. This window is only visible when the Debugger is active.

Opening and Closing the Dependency Tree Window

To open and close the Dependency Tree Window, do the following:

1 On the *Debug Menu, on the Debug Views sub-menu, select or clear Dependencies.*

   OR

   Press ALT+6.

Displaying Dependencies

To display the code for a dependent database object in the DDL Editor window, do the following:

1 In the Dependency Tree window, double-click the target object.

   Rapid SQL displays the SQL of the target object in the DDL Editor window.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Interface.

EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS FUNCTIONALITY

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows offers the following functionality:

- Input Parameters
- Step Into

- [Step Out](#)

- [Step Over](#)

- [Run to Cursor](#)

- [Insert or Remove a Breakpoint](#)

- [Toggle Breakpoint](#)

- [Go](#)

- [Stop](#)

- [Restart](#)

- [Break](#)

- [Close](#)

To use these functionalities, first [open a debugging session](#).

INPUT PARAMETERS FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

Input parameters are set when you first create an object. If the object you want to debug requires input parameters, Rapid SQL opens a Procedure Execution dialog box and prompts you for the input parameters when you [open a debugging session](#).

The Procedure Execution dialog box also lets you:

- Save input parameters as *.prm files to preserve specific input parameter configurations.

- Open *.prm files to save the effort of reentering specific input parameters.

- Reset parameters to their default setting.

The table below describes the options and functionality on Procedure Execution dialog box:

The following table describes the options available in this dialog box:

Dialog box component	Description
Owner drop-down list	Displays the current procedure's owner
Procedure drop-down list	Displays the name of the current procedure.
Parameter window	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message appears in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open button	Click to open an Open dialog box, from which you can open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.

Dialog box component	Description
Save button	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset button	Click to reset the parameters in the Parameter window to their default values.
Execute button	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

Option	Description
Owner	Displays the current procedure's owner.
Procedure	Displays the name of the current procedure.
Parameter	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message displays in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open	Click to open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset	Click to reset the parameters in the Parameter window to their default values.
Continue	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

NOTE: You cannot debug a script that requires input parameters until you provide input parameters.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

STEP INTO FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session, Step Into lets you execute the current instruction. If the current instruction makes a call to a stored SQL object, the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows steps inside the nested child object.

To use the Step Into facility, do the following:

1   On the *Debug* menu, click *Step Into.*

    OR

    On the *DDL Editor* toolbar, click Step Into.

    OR

    In the DDL Editor window, right-click, and then click *Step Into.*

    OR

    Press F11.

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows moves the arrow to execute the current instruction.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

STEP OUT FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session, Step Out lets you execute the remainder of the dependent child object and resumes line-by-line, step-debugging in the parent object.

> **NOTE:**   Step Out is only active when the pointer indicates a child dependent instruction.

To use the Step Out facility, do the following:

1   On the *Debug* menu, click *Step Out.*

    OR

    On the *DDL Editor* toolbar, click *Step Out.*

    OR

    In the DDL Editor window, right-click, and then click *Step Out.*

    OR

    Press SHIFT+F11.

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows stops stepping through the current object and executes the remainder of the script.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

STEP OVER FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX,

UNIX, AND WINDOWS

After you open a debugging session, Step Over lets you execute the current instruction without stepping into a nested child object if the instruction makes a call to a dependent object.

To use the Step Over, do the following:

1   On the *Debug* menu, click *Step Over.*

    OR

    On the *DDL Editor* toolbar, click *Step Over.*

    OR

    In the DDL Editor window, right-click, and then click *Step Over.*

    OR

    Press F10.

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows executes the current instruction.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

RUN TO CURSOR FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session, Run to Cursor lets you execute all instructions between the yellow arrow and the cursor.

To use the Run to Cursor facility, do the following:

1   *Scroll down from the yellow arrow to the target line.*

2   Click the target line.

    Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows places the cursor on the target line.

3   On the *Debug* menu, click *Run to Cursor.*

    OR

    On the *DDL Editor* toolbar, click *Run to Cursor.*

    OR

    In the DDL Editor window, right-click, and then click *Run to Cursor.*

    OR

    Press CTRL+F10.

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows executes all instructions between the pointer and the cursor.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

INSERT OR REMOVE BREAKPOINT FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

A breakpoint is a position in a program where a debugger stops execution. When you start debugging, Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows opens the script in a DDL Editor. A yellow arrow pointer indicates which line the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows executes next.

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows executes all lines of code between the yellow arrow and the first breakpoint. If no breakpoints are present, Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows debugs the entire script.

While debugging you can set one or more breakpoints in the currently executing object or in any object in the program call stack. You can toggle, temporarily disable or enable breakpoints without having to add or remove breakpoints.

Rapid SQL's Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows displays each enabled breakpoint as a red dot in the left margin of the DDL Editor Window, and each disabled breakpoint as a red circle.

Rapid SQL stores all breakpoints you set so that when you debug the same script on separate occasions, you can reuse the same breakpoints. After you open a debugging session, you can insert a breakpoint on the line where your cursor is located, and you can remove a breakpoint on the line where your cursor is located.

> **NOTE:** Script execution stops at the first breakpoint.

To insert and remove breakpoints, do the following:

1 In the DDL Editor window, click the target line of SQL.

2 On the *Debug* menu, click *Breakpoint.*

> OR
>
> On the *DDL Editor* toolbar, click Breakpoint.
>
> OR
>
> In the *DDL Editor* window, right-click, and then click *Breakpoint.*
>
> OR
>
> Press F9.
>
> The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows inserts a new breakpoint or removes an existing breakpoint on the target line of code.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

TOGGLE BREAKPOINT FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session and insert a breakpoint, Toggle Breakpoint lets you enable or disable that breakpoint. Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows displays each enabled breakpoint as a red dot in the left margin of the DDL Editor Window, and each disabled breakpoint as a red circle.

You can toggle any breakpoint in the DDL Editor window. When you exit a debugging session and reenter it, the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows retains any breakpoints you set.

To use the Toggle Breakpoint facility, do the following:

1   In the DDL Editor window, click the line of the target breakpoint.

2   On the *Debug* menu, click *Enable/Disable Breakpoint.*

   OR

   On the *DDL Editor* toolbar, click *Enable/Disable Breakpoit.*

   OR

   In the *DDL Editor* window, right-click, and then click *Enable/Disable Breakpoint.*

   OR

   Press CTRL+F9.

   The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows toggles the breakpoint indicated by the pointer.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

GO FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session, Go lets you execute all instructions stopping only when when it encounters a breakpoint or when the program is complete.

To use the Go facility, do the following:

1   On the *Debug* menu, click *Go.*

OR

On the *DDL Editor* toolbar, click Go.

OR

In the *DDL Editor* window, right-click, and then click *Go.*

OR

Press F5.

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows executes all instructions.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

Stop for Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows

After you open a debugging session, Stop lets you halt the script execution and terminate the session.

To use the Stop facility, do the following:

1   On the *Debug* menu, click *Stop Debugging.*

OR

On the *DDL Editor* toolbar, click *Stop Debugging.*

OR

In the *DDL Editor* window, right-click, and then click *Stop Debugging.*

OR

Press SHIFT+F5.

The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows stops the script execution and terminates the session.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

Restart for Embarcadero SQL Debugger for IBM DB2 LUW for Linux,

UNIX, AND WINDOWS

After you open a debugging session, Restart lets you terminate the current debug session and open a new one. When the new session opens, Rapid SQL prompts you for new input parameters.

To use the Restart facility, do the following:

1   On the *Debug* menu, click *Restart.*

    OR

    On the *DDL Editor* toolbar, click Restart.

    OR

    In the *DDL Editor* window, right-click, and then click *Restart.*

    OR

    Press CTRL+SHIFT+F5.

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows restarts the debug session.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

BREAK FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session, Break lets you pause the debug session.

To use the Break facility, do the following:

1   On the *Debug* menu, click *Break.*

    OR

    On the *DDL Editor* toolbar, click Break.

    OR

    In the *DDL Editor* window, right-click, and then click *Break.*

    The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows suspends the debug session.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

CLOSE FOR EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX,

AND WINDOWS

After you open a debugging session, Close lets you close the DDL Editor and the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows.

To use the Close facility, do the following:

1 On the *DDL Editor* toolbar, click Close.

   OR

   In the upper right corner of the window, click Close.

   OR

   In the *DDL Editor* window, right-click, and then click *Close.*

   The Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows closes the debug session.

For more information, see Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows Functionality.

USING THE EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

This section offers a general overview of how to use Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows's full range of debugging functionality. After you open a debugging session for any procedure or trigger, you can begin debugging.

OPENING A DEBUGGING SESSION IN IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

When you open a debugging session, Rapid SQL opens the five windows of the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows interface. If the target script requires input parameters, Rapid SQL opens a Procedure Execution dialog box and prompts you for the necessary input parameters before displaying the target code in the SQL Editor window. When Rapid SQL displays the target script in the SQL Editor window, you can begin debugging.

> **NOTE:** Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows only lets you debug the SQL script of procedures or triggers.

To debug triggers by debugging the procedures that call them, do the following:

1 On the Explorer Tab, click the node of the target trigger or procedure. The node expands and displays the Code object.

2 Double-click Code. The DDL Editor opens and displays the code of the target object.

3 On the Debug menu, click Start Debugging.

   OR

   On the SQL Editor toolbar, click Debug.

   OR

   In the *DDL Editor* window, right-click, and then click *Debug.*

   OR

   Press CTRL+F5.

4 On the toolbar, click Debug.

   OR

   In the *DDL Editor* window, right-click, and then click *Debug.*

   If the script requests input parameters, Rapid SQL opens a Procedure Execution dialog box. If the script does not require input parameters, Rapid SQL displays the script in the DDL Editor window for you to begin debugging.

   > **NOTE:** You cannot use the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows until it has fully initialized.

5 In the Procedure Execution dialog box, specify the appropriate parameters, and then click *Continue.*

   Rapid SQL displays the script in the DDL Editor window for you to begin debugging.

DEBUGGING AN SQL SCRIPT WITH EMBARCADERO SQL DEBUGGER FOR IBM DB2 LUW FOR LINUX, UNIX, AND WINDOWS

After you open a debugging session and enter any required input parameters, you can begin working with your script in the Embarcadero SQL Debugger for IBM DB2 LUW for Linux, Unix, and Windows.

Debugging an SQL Script

To debug a SQL Script, do the following:

1 On the *Debug* menu, click one of the T-SQL Debugger options (Step Into, Step Over, and so forth) or click *Go.*

   OR

   On the *DDL Editor* toolbar, click one of the T-SQL Debugger options (Step Into, Step Over, and so on) or click Go.

   > **NOTE:** You can monitor the progress of your debug session in the Variables window.

2   On the *Debug* menu, click *Breakpoint*.

OR

On the *DDL Editor* toolbar, click Breakpoint.

OR

Press F9.

> **NOTE:**   When you set a breakpoint, the Call Stack window shows what was called before the breakpoint.

> **NOTE:**   You can use the <u>Run to Cursor</u> option to test the lines of code between a breakpoint and your cursor (indicated by the yellow arrow in the DDL Editor).

3   To check your variables:

1) In the **DDL Editor**, click a variable in your script and drag it to the **Watch** window.

2) In the **Watch** window, change the value of the watch variable, and then click **Go** to run your script and see the results of the new value.

4   To check a record in stored objects:

1) Drag the record to the **Watch** window.

2) In the **Watch** window, change the value of the record, then click **Go** to run your script and see the results of the new value.

5   To check the dependencies:

1) In the **Dependency Tree** *window* double-click the target dependent object to extract the code into a new **DDL Editor**.

2) **Step through** the script while monitoring the <u>Dependency Tree window.</u>

6   When you finish debugging the script, click Close.

Rapid SQL closes the T-SQL Debugger DDL Editor.

> **NOTE:**   When you exit a debug session and reenter it, the Embarcadero SQL Debugger for MSSQL retains any watch variables or breakpoints you have set.

EMBARCADERO SQL DEBUGGER FOR MICROSOFT SQL SERVER

Embarcadero SQL Debugger for Microsoft is a programming tool that helps you locate and fix bugs in Microsoft SQL Server procedures and triggers for Microsoft SQL Server version 7.0 or later.

> **NOTE:**   Availability of this feature depends on your Rapid SQL licensing. For more information, see <u>Licensing</u>.

Objects

You can use Embarcadero SQL Debugger for Microsoft to debug the following objects:

- Procedures

- Triggers

You can only debug triggers by debugging the procedures that call them.

> **TIP:** The 'Code Analyst is a tool to identify time-consuming lines of code. Code Analyst lets you perform detailed response time analysis on the execution of Procedures and Functions.

The table below describes the sections of this chapter:

Section	Description
Embarcadero SQL Debugger for Microsoft Features	This section describes how Embarcadero SQL Debugger for Microsoft helps you identify problems within your code.
Embarcadero SQL Debugger for Microsoft Interface	This section describes the Embarcadero SQL Debugger for Microsoft graphical interface that includes an editor window and four debug view windows.
Embarcadero SQL Debugger for Microsoft Functionality	This section describes the way in which Embarcadero SQL Debugger for Microsoft functions.
Using the Embarcadero SQL Debugger for Microsoft	This section describes how to run a debug session.

EMBARCADERO SQL DEBUGGER FOR MICROSOFT FEATURES

The Embarcadero SQL Debugger for Microsoft helps you identify problems within your code. The Embarcadero SQL Debugger for Microsoft lets you:

- Interactively step through the flow of script execution.

- Examine the value of variables.

- Solve logical problems with your script design.

The Embarcadero SQL Debugger for Microsoft offers fundamental debugging features and several options to help fine tune debugging, as listed in the table below:

Debugging Feature	Description
Step Into	Lets you execute each instruction step-by-step and step inside a stored object.
Step Out	Lets you stop stepping through the current object and execute the remainder of the script. This option is only active when the pointer indicates a child dependent instruction.
Step Over	Lets you execute the current instruction without stepping into any child dependents.

Debugging Feature	Description
Breakpoints	A position in a program where the debugger stops execution.

To set specific Debugger values on Rapid SQL's Options Editor, see Debugger Options.

EMBARCADERO SQL DEBUGGER FOR MICROSOFT REQUIREMENTS

To use the Embarcadero SQL Debugger for Microsoft, you must properly configure the server and client. For more information, see:

Server Requirements

Client Requirements

SERVER REQUIREMENTS

To use the Embarcadero SQL Debugger for Microsoft you must be running Windows 2000 or Windows NT 4.0 or later, and your Microsoft SQL Server version must be 7.0 or later.

Setting Up the Server

There are three parts to setting up the server:

- Installing the Microsoft SQL Debugger Interface subcomponent.

- Configuring the Service.

- Configuring DCOM on the server

Enabling SQL Debugger for Microsoft on SQL Server SP3

SQL Debugging is disabled by default in SQL Server SP3 and greater. Please refer to Microsoft Support for information regarding enabling the SQL Debugger for Microsoft on SQL Server SP3.

INSTALLING THE MICROSOFT SQL DEBUGGER INTERFACE SUBCOMPONENT

The Microsoft server must have the Development Tools, Debugger Interface subcomponent of Microsoft SQL Server 7.0. To determine if the Debugger Interface subcomponent is installed, locate the following files in the \Program Files\Common Files\Microsoft Shared\SQL Debugging directory:

- SQLDBREG.exe

- SQLDBG.dll

If these files are not in the \Program Files\Common Files\Microsoft Shared\SQL Debugging directory, you must install them before running the Embarcadero SQL Debugger for Microsoft.

Installing the Microsoft SQL Debugger Interface on the Server

To install the Debugger Interface subcomponent on the server after the initial installation, do the following:

1   Start Microsoft Visual Studio, Enterprise Edition Setup.

    OR

    Start Microsoft SQL Server 7.0 Setup.

2   Select Custom Install.

    Microsoft SQL Server opens the Select Components dialog box:

3   In the Components box, select the Development Tools check box.

4   In the Sub-components box, select the Debugger Interface check box.

5   Click Next.

    Microsoft SQL Server proceeds through the Microsoft SQL Server wizard to install the components.

CONFIGURING THE SERVICE

Configuring the service is an operating-system-dependent operation. See the instructions below for your server operating system.

- [WIndows 2000](#)

- [Windows NT 4.0](#)

Windows 2000

1   On the Windows taskbar, click the Start button, click Settings, and then click Control Panel.

2   Double-click Administrative Tools, and then click Services.

    Windows opens the Services explorer.

3   In the right pane of the Services explorer, right click MSSQLServer, and then click Properties.

    Windows opens the Net Logon Properties dialog box.

4   Click the Logon Tab.

5   Select the This Account option button.

6   In the This Account box, type (or browse to locate) the logon user account (including domain name, if necessary) of the person using the Embarcadero SQL Debugger for Microsoft.

    NOTE: This person must have admin permissions on the server.

7   In the Password and Confirm Password boxes, type the password.

8   Click Apply.

9   Click the General Tab.

10   Click Start.

     Windows starts the server and applies the changes.

Important Notes about Microsoft SQL Server 2000 Service Pack 3 (SP3)

By default, after you install Microsoft SQL Server 2000 Service Pack 3 (SP3), you cannot use the Embarcadero SQL Debugger for Microsoft. You may receive the following error message:

> "Server: Msg 514, Level 16, State 1, Procedure sp_sdidebug, Line 1 [Microsoft][ODBC SQL Server Driver][SQL Server]Unable to communicate with debugger on [SQL Server Name] (Error = 0x80070005). Debugging disabled for connection 53."

Microsoft made this change for security reasons. To enable debugging, a member of the sysadmins server role, such as sa, must explicitly enable debugging by running the following code:

```
 Exec sp_sdidebug 'legacy_on'
```

You must repeat this procedure whenever you restart the server.

Windows NT 4.0

1   On the Windows taskbar, click the Start button, select Settings and then click Control Panel.

2   Double-click Services.

     Windows opens the Services dialog box.

3   In the Service list, select MSSQLServer and then click Startup.

4   In the Log On As: box, select the This Account option button.

5   In the This Account box, type the logon user account (including domain name, if necessary) of the person using the Embarcadero SQL Debugger for Microsoft.

     NOTE: This person must have admin permissions on the server.

6   In the Password and Confirm Password boxes, type the password.

7   Click OK.

     Windows returns to the Services dialog box.

8   Click Start.

     Windows starts the server and applies the changes.

CONFIGURING DCOM ON THE SERVER

To configure DCOM, do the following:

1   After the server restarts, on the Windows taskbar, click the Start button, and then click Run.

2   In the Open box, type dcomcnfg.exe.

3   Click OK.

    Windows opens the Distributed COM Configuration Properties dialog box.

4   Click the Default Security Tab.

5   In the Default Access Permissions box, click Edit Default.

    Windows opens the Registry Value Permissions dialog box.

6   Click Add.

    Windows opens the Add Users and Groups dialog box.

7   In the Names box, select SYSTEM, and then click Add.

8   Click the Type of Access list and then click Allow Access.

9   To let any user use the Embarcadero SQL Debugger for Microsoft, you must grant them remote access on the server. To grant remote access, you must configure their DCOM permissions on the server. In the Names box, click the target users and then click Add.

    NOTE: You can add individual users or groups.

10   Click the Type of Access list and then click Allow Access.

11   Click OK.

12   Restart the server to apply the changes.

CLIENT REQUIREMENTS

There are three categories of client requirements for the Embarcadero SQL Debugger for Microsoft:

- Operating System

- Microsoft SQL Server 7.0, Client Connectivity Component

- Microsoft SQL Server 7.0, Development Tools - Debugger Interface Subcomponent

Operating System

The client must be running one of the following operating systems:

- Microsoft Windows 95

- Microsoft Windows 98

- Microsoft Windows NT 4.0 or later

Important Notes about Microsoft SQL Server 2000 Service Pack 3 (SP3)

By default, after you install Microsoft SQL Server 2000 Service Pack 3 (SP3), you cannot use the Embarcadero SQL Debugger for Microsoft. You may receive the following error message:

> "Server: Msg 514, Level 16, State 1, Procedure sp_sdidebug, Line 1 [Microsoft][ODBC SQL Server Driver][SQL Server]Unable to communicate with debugger on [SQL Server Name] (Error = 0x80070005). Debugging disabled for connection 53."

Microsoft made this change for security reasons. To enable debugging, a member of the sysadmins server role, such as sa, must explicitly enable debugging by running the following code:

```
 Exec sp_sdidebug 'legacy_on'
```

You must repeat this procedure whenever you restart the server.

Client Connectivity

The client must have the Client Connectivity component of Microsoft SQL Server 7.0 or later.

Microsoft Debugger Interface

The client must have the Development Tools, Debugger Interface subcomponent of Microsoft SQL Server 7.0 or later. To determine if the Debugger Interface subcomponent is installed, locate the following files in the \Program Files\Common Files\Microsoft Shared\SQL Debugging directory:

- SQLDBREG.exe

- SQLDBG.dll

If these files are not in the \Program Files\Common Files\Microsoft Shared\SQL Debugging directory, you must install them before running the Embarcadero SQL Debugger for Microsoft.

Installing the Microsoft SQL Debugger Interface on the Client

To install the Debugger Interface subcomponent on the client:

1   Start the Microsoft SQL Server Setup program.

2   Select Custom Install.

    Microsoft SQL Server opens the Select Components dialog box.

3   In the Components box, select the Development Tools check box.

4   In the Sub-Components box, select the Debugger Interface check box.

5   Click Next.

Microsoft SQL Server proceeds through the Microsoft SQL Server Wizard to install the components.

EMBARCADERO SQL DEBUGGER FOR MICROSOFT OPTIONS

You can specify T-SQL Debugger options from the Debug Tab of Rapid SQL's Options editor. The Debug Tab of the Options Editor lets you set the duration of your debug initialization and debug session, enable DBMS output, and refresh dependencies.

Setting Debugger Options

To set debugger options, do the following:

1   On the *File* menu, click *Options.*

OR

On the *Main* toolbar, click Options.

Rapid SQL opens the Options Editor.

2   Specify debugger options. The table below describes the options available:

Option	Description	Default
Initialization Timeout (seconds)	Specifies the number of seconds Rapid SQL tries to initialize the debugger. If it cannot initialize the debugger in the specified time, a message displays in the Debug Output window.	60
Debug Session Timeout (seconds)	Specifies, in seconds, the length of your debug session.	7200
Enable DBMS Output	Toggles the print output. Enable this option if you use dbms_output.put_line calls in your procedures and you want these lines displayed.	Selected
Refresh Dependencies for each run	Refreshes dependencies for each run. This potentially time-consuming process is useful if the target procedure has rapidly varying dependencies that can require updating during the debugging process.	Cleared

3   Click Close.

Rapid SQL closes the Options Editor.

EMBARCADERO SQL DEBUGGER FOR MICROSOFT INTERFACE

The Embarcadero SQL Debugger for Microsoft has a graphical interface that includes an editor window and four debug view windows. When you open a *debug session*, Rapid SQL extracts the code for the object into a DDL Editor and opens four debug view windows at the bottom of the screen. The four debug view windows are optional, dockable windows designed to help you debug your script.

> **TIP:** All Embarcadero debuggers display Performance Metrics that let you measure the execution time of each statement in the debug session.

Embarcadero SQL Debugger for Microsoft's five windows are:

1 [DDL Editor window](#)

2 [Watch window](#)

3 [Variables window](#)

4 [Call Stack window](#)

5 [Dependency Tree window](#)

Working with T-SQL Debugger Windows

Rapid SQL lets you resize, move, dock and float the following T-SQL Debugger windows:

- Watch
- Variables
- Call Stack
- Dependency Tree

To work with the above windows, do the following:

1 To resize the target window, click its frame and drag it

   Rapid SQL resizes the window.

2 To move and dock the target window, click its grab bar and drag it.

   Rapid SQL moves the window to its new location and docks it with surrounding windows.

3 To float the target window, press Shift, then click its grab bar and drag it.

   Rapid SQL frames the window in its own floating frame and moves the window to its new location.

DDL EDITOR WINDOW

The Embarcadero SQL Debugger for Microsoft provides a DDL Editor that displays your code in read-only format. When you start debugging, the SQL Embarcadero SQL Debugger for Microsoft extracts your code into a DDL Editor. The DDL Editor uses the default Rapid SQL syntax coloring.

WATCH WINDOW

The Rapid SQL Embarcadero SQL Debugger for Microsoft provides a watch window that displays the watch variables for the database object you are debugging and lets you specify variables you want to evaluate or modify while debugging your program. For example, to check what happens when a variable (x) has a value of 100, you can double-click the variable in the DDL Editor, drag it into the Watch Window, and change the value to 100. When you execute the script, the Debugger uses the value x =100. This window is only visible when the T-SQL Debugger is active.

> **NOTE:** Until you step at least once into a script, variables are not defined. Therefore, you must step at least once before dragging or typing a local variable in the Watch Window.

> **NOTE:** When you exit a debug session and reenter it, the Embarcadero SQL Debugger for Microsoft retains any watch variables or breakpoints you have set.

Opening and Closing the Watch Window

To open and close the Watch Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Watch.*

    OR

    Press ALT+3.

Setting a Watch Variable

To set a Watch Variable, do the following:

1   In the *DDL Editor,* double-click the target variable and drag it to the *Watch window.*

    > **NOTE:** Microsoft SQL Server requires that local variables begin with @. You must drag the @ to the Watch Window.

2   In the *Watch window,* change the value of the variable.

3   On the *DDL Editor,* click Go.

    The Embarcadero SQL Debugger for Microsoft executes the script using the new variable.

Removing a Watch Variable

To remove a Watch Variable, do the following:

1   In the *Watch window*, click the target variable and press *DELETE*.

VARIABLES WINDOW

The Embarcadero SQL Debugger for Microsoft provides a variables window that displays the local variables and their current values during script execution. You cannot edit the variables in the Variables window. If the DDL Editor displays an external database object, and that object is a dependent of the object you are debugging, then the Variables Window automatically refreshes and displays the variables for that particular object. The Variables Window is only visible when the Debugger is active.

The Embarcadero SQL Debugger for Microsoft also lets you monitor your variables while debugging.

Opening and Closing the Variables Window

To open and close the Variables Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Variable.*

    OR

    Press ALT+4.

Monitoring Variables

To monitor the values of your variables while debugging, do the following:

1   In the SQL Editor, hold the pointer over the target variable.

    Rapid SQL opens a ScreenTip displaying the current value of that variable.

CALL STACK WINDOW

The Embarcadero SQL Debugger for Microsoft provides a call stack window that displays the stack of currently active calls. The Call Stack Window is only visible when the Debugger is active.

Opening and Closing the Call Stack Window

To open and close the Call Stack Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Call Stack.*

    OR

    Press ALT+5.

Using the Call Stack Window

To display a line of code that references the call in the DDL Editor, do the following:

1   In the Call Stack window, double-click the target line.

In the DDL Editor, Rapid SQL displays a green arrow on the line of the referenced call.

DEPENDENCY TREE WINDOW

The Embarcadero SQL Debugger for Microsoft provides a dependency tree window that displays any external database objects the script accesses. Rapid SQL displays these database objects in a hierarchical tree, with the child objects as database objects accessed by the parent objects. You can use this window to display the code for a dependent database object in the *DDL Editor* window. This window is only visible when the Debugger is active.

Opening and Closing the Dependency Tree Window

To open and close the Dependency Tree Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Dependencies.*

OR

Press ALT+6.

Displaying Dependencies

To display the code for a dependent database object in the DDL Editor window, do the following:

1   In the Dependency Tree window, double-click the target object.

Rapid SQL displays the SQL of the target object in the DDL Editor window.

EMBARCADERO SQL DEBUGGER FOR MICROSOFT FUNCTIONALITY

The Embarcadero SQL Debugger for Microsoft offers the following functionality:

- Input Parameters
- Step Into
- Step Out
- Step Over
- Run to Cursor
- Insert or Remove a Breakpoint
- Toggle Breakpoint

- [Go](#)

- [Stop](#)

- [Restart](#)

- [Break](#)

- [Close](#)

To make use of the above functionality, you must first [open a debugging session](#).

INPUT PARAMETERS

Input parameters are set when you first create an object. If the object you want to debug requires input parameters, Rapid SQL opens a Procedure or Trigger Execution dialog box and prompts you for the input parameters when you [open a debugging session](#).

This dialog box also lets you:

- Save input parameters as *.prm files to preserve specific input parameter configurations.

- Open *.prm files to save the effort of reentering specific input parameters.

- Reset parameters to their default setting.

The table below describes the options and functionality on Procedure Execution dialog box:

The following table describes the options available in this dialog box:

Dialog box component	Description
Owner drop-down list	Displays the current procedure's owner
Procedure drop-down list	Displays the name of the current procedure.
Parameter window	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message appears in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open button	Click to open an Open dialog box, from which you can open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save button	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset button	Click to reset the parameters in the Parameter window to their default values.
Execute button	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

Option	Description
Owner	Displays the current procedure's owner.
Procedure	Displays the name of the current procedure.
Parameter	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message displays in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open	Click to open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset	Click to reset the parameters in the Parameter window to their default values.
Continue	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

NOTE: You cannot debug a script that requires input parameters until you provide input parameters.

STEP INTO

After you [open a debugging session](#), Step Into lets you execute the current instruction. If the current instruction makes a call to a stored SQL object, the Embarcadero SQL Debugger for Microsoft steps inside the nested child object.

Step Into

To use the Step Into facility, do the following:

1   On the *Debug* menu, click *Step Into.*

OR

On the *DDL Editor* toolbar, click Step Into.

OR

In the DDL Editor window, right-click and then click *Step Into.*

OR

Press F11.

The Embarcadero SQL Debugger for Microsoft moves the arrow to execute the current instruction.

STEP OUT

After you open a debugging session, Step Out lets you execute the remainder of the dependent child object and resumes line-by-line, step-debugging in the parent object.

> **NOTE:** Step Out is only active when the pointer indicates a child dependent instruction.

Step Out

To use the Step Out facility, do the following:

1   On the *Debug* menu, click *Step Out.*

    OR

    On the *DDL Editor* toolbar, click Step Out.

    OR

    In the DDL Editor window, right-click and then click *Step Out.*

    OR

    Press SHIFT+F11.

    The Embarcadero SQL Debugger for Microsoft stops stepping through the current object and executes the remainder of the script.

STEP OVER

After you open a debugging session, Step Over lets you execute the current instruction without stepping into a nested child object if the instruction makes a call to a dependent object.

Step Over

To use the Step Over facility, do the following:

1   On the *Debug* menu, click *Step Over.*

    OR

    On the *DDL Editor* toolbar, click Step Over.

    OR

    In the DDL Editor window, right-click and then click *Step Over.*

    OR

    Press F10.

    The Embarcadero SQL Debugger for Microsoft executes the current instruction.

RUN TO CURSOR

After you [open a debugging session](#), Run to Cursor lets you execute all instructions between the yellow arrow and the cursor.

Run to Cursor

To use the Run to Cursor facility, do the following:

1   *Scroll down from the yellow arrow to the target line.*

2   Click the target line.

   Embarcadero SQL Debugger for Microsoft places the cursor on the target line.

3   On the *Debug* menu, click *Run to Cursor.*

   OR

   On the *DDL Editor* toolbar, click Run to Cursor.

   OR

   In the DDL Editor window, right-click and then click *Run to Cursor.*

   OR

   Press CTRL+F10.

   The Embarcadero SQL Debugger for Microsoft executes all instructions between the pointer and the cursor.

INSERT OR REMOVE BREAKPOINT

A breakpoint is a position in a program where a debugger stops execution. When you start debugging, Embarcadero SQL Debugger for Microsoft opens the script in a DDL Editor. A yellow arrow pointer indicates which line the Embarcadero SQL Debugger for Microsoft executes next. The Embarcadero SQL Debugger for Microsoft executes all lines of code between the yellow arrow and the first breakpoint. If no breakpoints are present, Embarcadero SQL Debugger for Microsoft debugs the entire script.

While debugging you can set one or more breakpoints in the currently executing object or in any object in the program call stack. Breakpoints can be [toggled](#), temporarily disabled, or enabled, without having to add or remove them. Rapid SQL Embarcadero SQL Debugger for Microsoft displays each enabled breakpoint as a red dot in the left margin of the DDL Editor Window, and each disabled breakpoint as a red circle.

Rapid SQL stores all breakpoints you set so that when you debug the same script on separate occasions, you can reuse the same breakpoints. After you [open a debugging session](#), Insert Break lets you insert a breakpoint on the line where your cursor it located, and Remove Break lets you remove a breakpoint on the line where your cursor is located.

   NOTE: Script execution stops at the first breakpoint.

Inserting or Removing a Breakpoint

To insert of remove a breakpoint, do the following:

1   In the DDL Editor window, click the target line of SQL.

2   On the *Debug* menu, click *Breakpoint.*

    OR

    On the *DDL Editor* toolbar, click Breakpoint.

    OR

    In the *DDL Editor* window, right-click and then click *Breakpoint.*

    OR

    Press F9.

    The Embarcadero SQL Debugger for Microsoft inserts a new breakpoint or removes an existing breakpoint on the target line of code.

TOGGLE BREAKPOINT

After you open a debugging session and insert a breakpoint, Toggle Breakpoint lets you enable or disable that breakpoint. Embarcadero SQL Debugger for Microsoft displays each enabled breakpoint as a red dot in the left margin of the DDL Editor Window, and each disabled breakpoint as a red circle. You can toggle any breakpoint in the DDL Editor window. When you exit a debugging session and reenter it, the Embarcadero SQL Debugger for Microsoft retains any breakpoints you set.

Toggling a Breakpoint

To toggle a breakpoint, do the following:

1   In the DDL Editor window, click the line of the target breakpoint.

2   On the *Debug* menu, click *Enable/Disable Breakpoint.*

    OR

    On the *DDL Editor* toolbar, click *Enable/Disable Breakpoint*.

    OR

    In the *DDL Editor* window, right-click and then click *Enable/Disable Breakpoint.*

    OR

    Press CTRL+F9.

    The Embarcadero SQL Debugger for Microsoft toggles the breakpoint indicated by the pointer.

GO

After you open a debugging session, Go lets you execute all instructions stopping only when when it encounters a breakpoint or when the program is complete.

Go

To use the Go facility, do the following:

1   On the *Debug* menu, click *Go.*

    OR

    On the *DDL Editor* toolbar, click Go.

    OR

    In the *DDL Editor* window, right-click and then click *Go.*

    OR

    Press F5.

    The Embarcadero SQL Debugger for Microsoft executes all instructions.

STOP

After you open a debugging session, Stop lets you halt the script execution and terminate the session.

Stop

To stop the debugger, do the following:

1   On the *Debug* menu, click *Stop Debugging.*

    OR

    On the *DDL Editor* toolbar, click *Stop Debugging.*

    OR

    In the *DDL Editor* window, right-click and then click *Stop Debugging.*

    OR

    Press SHIFT+F5.

    The Embarcadero SQL Debugger for Microsoft stops the script execution and terminates the session.

RESTART

After you [open a debugging session](#), Restart lets you terminate the current debug session and open a new one. When the new session opens, Rapid SQL prompts you for new input parameters.

Restart

To restart the debugger, do the following:

1   On the *Debug* menu, click *Restart.*

    OR

    On the *DDL Editor* toolbar, click *Restart.*

    OR

    In the *DDL Editor* window, right-click and then click *Restart.*

    OR

    Press CTRL+SHIFT+F5.

    The Embarcadero SQL Debugger for Microsoft restarts the debug session.

BREAK

After you [open a debugging session](#), Break lets you pause the debug session.

Break

To pause the debugger, do the following:

1   On the *Debug* menu, click *Break.*

    OR

    On the *DDL Editor* toolbar, click Break.

    OR

    In the *DDL Editor* window, right-click and then click *Break.*

    The Embarcadero SQL Debugger for Microsoft suspends the debug session.

CLOSE

After you [open a debugging session](#), Close lets you close the DDL Editor and the Embarcadero SQL Debugger for Microsoft.

Close

1   On the *DDL Editor* toolbar, click Close.

   OR

   In the upper right corner of the window, click Close.

   OR

   In the *DDL Editor* window, right-click and then click *Close.*

   The Embarcadero SQL Debugger for Microsoft closes the debug session.

USING THE EMBARCADERO SQL DEBUGGER FOR MICROSOFT

This section offers a general overview of how to use Embarcadero SQL Debugger for Microsoft's full range of debugging [functionality](). After you [open a debugging session]() for any procedure or trigger, you can [begin debugging]().

For more detailed information, see [Debugging a Sample Script]().

OPENING A DEBUGGING SESSION

When you open a debugging session, Rapid SQL opens the [five windows]() of the Embarcadero SQL Debugger for Microsoft interface. If the target script requires input parameters, Rapid SQL opens a Procedure Execution dialog box and prompts you for the necessary input parameters before displaying the target code in the SQL Editor window. When Rapid SQL displays the target script in the SQL Editor window, you can [begin debugging]().

> **NOTE:**   Rapid SQL Embarcadero SQL Debugger for Microsoft only lets you debug the SQL script of procedures or triggers.

Opening a Debugging Session

To debug a trigger or procedure, do the following:

1   On the *Explorer* Tab, click the node of the target procedure.

   Rapid SQL opens the node and displays two items: Code and Privileges.

2   Under the target object node, double-click Code.

   Rapid SQL opens a DDL Editor *displaying the code of the target object.*

3   On the Debug menu, click Start Debugging.

OR

On the SQL Editor toolbar, click Debug.

OR

In the *DDL Editor* window, right-click and then click *Debug.*

OR

Press CTRL+F5.

If the script requests input parameters, Rapid SQL opens a Procedure Execution dialog box. If the script does not require input parameters, Rapid SQL displays the script in the DDL Editor window for you to begin debugging.

   NOTE: You cannot use the Embarcadero SQL Debugger for Microsoft until it has fully initialized.

4   In the Procedure Execution dialog box, specify the appropriate parameters, and then click *Continue*.

Rapid SQL displays the script in the DDL Editor window for you to begin debugging.

DEBUGGING AN SQL SCRIPT

After you open a debugging session and enter any required input parameters, you can begin working with your script in the Embarcadero SQL Debugger for Microsoft.

Debugging an SQL Script

To debug a SQL script, do the following:

1   On the *Debug* menu, click one of the Embarcadero SQL Debugger for MSSQL Server options (Step Into, Step Over, and so forth) or click *Go*.

OR

On the *DDL Editor* toolbar, click one of the Embarcadero SQL Debugger for MSSQL Server options (Step Into, Step Over, and so on) or click Go.

   NOTE: You can monitor the progress of your debug session in the Variables window.

2   On the *Debug* menu, click *Breakpoint*.

OR

On the *DDL Editor* toolbar, click Breakpoint.

OR

Press F9.

> **NOTE:**   When you set a breakpoint, the Call Stack window shows what was called before the breakpoint.

> **NOTE:**   You can use the <u>Run to Cursor</u> option to test the lines of code between a breakpoint and your cursor (indicated by the yellow arrow in the DDL Editor).

3   To check your variables, do the following:

> 1) In the **DDL Editor**, click a variable in your script and drag it to the **Watch** window.

> 2) In the **Watch** window, change the value of the watch variable and then click **Go** to run your script and see the results of the new value.

4   To check a record in stored objects, do the following:

> 1) Drag the record to the **Watch** window.

> 2) In the **Watch** window, change the value of the record, then click **Go** to run your script and see the results of the new value.

5   To check the dependencies, do the following:

> 1) In the **Dependency Tree** *window* double-click the target dependent object to extract the code into a new **DDL Editor**.

> 2) **Step through** the script while monitoring the <u>Dependency Tree window.</u>

6   When you finish debugging the script, click Close.

Rapid SQL closes the T-SQL Debugger DDL Editor.

> **NOTE:**   When you exit a debug session and reenter it, the Embarcadero SQL Debugger for MSSQL retains any watch variables or breakpoints you have set.

DEBUGGING A SAMPLE SCRIPT

This walk-through demonstrates basic debugging functionality. During the course of this walk-through you debug two procedures using the Embarcadero SQL Debugger for Microsoft.

This section is divided into the following seven sections, each designed to familiarize you with basic debugging features and functionality:

- <u>Getting Started</u>

- <u>Testing a Procedure</u>

- <u>Starting the Debugging Session</u>

- [Breakpoints](#)

- [Step Into](#)

- [Step Out](#)

- [Correcting the Script](#)

GETTING STARTED

This part of Debugging the Sample Script explains how to create the following two procedures to be used for debugging:

- check_modulo

- calculate_sum_with_overflow_bug

  NOTE: The procedure calculate_sum_with_overflow_bug intentionally includes a bug which prevents it from executing successfully. You use the Embarcadero SQL Debugger for Microsoft to identify this bug.

Overview

The Getting Started section guides you through:

- [Creating procedure 1.](#)

- [Creating procedure 2.](#)

- [Confirming the creation of the procedures.](#)

CREATING PROCEDURE 1

Procedure 1, check_modulo, calculates the modulo of any two user-specified numbers. The user passes the numbers into the procedure as input parameters. The procedure returns the result as an output parameter. If the modulo equals zero, procedure execution returns the output "YES". If the modulo is not zero, procedure execution returns the output "NO". This procedure is nested in the second procedure, calculate_sum_with_overflow_bug.

To create this procedure, you must open Rapid SQL, connect to a MSSQL datasource, open a new SQL editor and, in the SQL editor, type or copy and paste the following code:

```
CREATE PROCEDURE username.check_modulo
@p_dividend_in  INT,
@p_divisor_in  INT,
@result VARCHAR(3)OUTPUT
AS
IF @p_dividend_in % @p_divisor_in = 0
    SELECT @result = 'YES'
ELSE
SELECT @result = 'NO'
go
```

  NOTE: For the purposes of this walk-through, this procedure was created under the user name Spence. Before executing the DDL above, substitute your user name for the word "username".

Creating Procedure 1

To create Procedure 1, do the following:

1   Start Rapid SQL.

2   Connect to a Microsoft SQL Server 7.0 datasource.

3   On the Datasource menu, click the database node and then click the target database.

> NOTE: For this walk-through, we recommend that you select a non-production database.

4   On the Main toolbar, click New.

OR

On the File menu, click New, and then click SQL.

OR

Press CTRL+N.

Rapid SQL opens an SQL Editor in the current workspace.

5   In the SQL Editor, type the DDL for procedure check_modulo.

> NOTE: You must substitute your user name once in the DDL for this procedure.

6   On the SQL Editor toolbar, click Execute.

Rapid SQL executes the script and creates Procedure 1, then opens the SQL Editor Results Tab with the results of the script execution. If you were not able to create the procedure, check the error messages to determine the problem.

CREATING PROCEDURE 2

Procedure 2, calculate_sum_with_overflow_bug, requires two user-specified numbers as input parameters. Upon execution, the procedure calculates the sum of the all numbers divisible by five between the two user-specified numbers. This procedure calls sample procedure 1 (check_modulo) to calculate the modulo of the user-specified numbers.

> NOTE: The procedure calculate_sum_with_overflow_bug intentionally includes a bug which prevents it from executing successfully. You use the Embarcadero SQL Debugger for Microsoft to identify this bug.

CAUTION: When inputting parameters, you must enter the smaller number in the @p_num1_in int box.

To create this procedure, you must open Rapid SQL, connect to a MSSQL datasource, open a new SQL editor and, in the SQL editor, type or copy and paste the following code:

```
CREATE PROCEDURE username.calculate_sum_with_overflow_bug
@p_num1_in  INT,
@p_num2_in  INT,
@result TINYINT OUTPUT
/*INT-Integer (whole number) data from -2^31 (-2,147,483,648)
  through 2^31 - 1 (2,147,483,647).
  TINYINT-Integer data from 0 through 255.*/
AS
DECLARE @temp INT
DECLARE @temp_1 INT
DECLARE @v_divisor INT
DECLARE @v_condition VARCHAR(3)

SET @temp = @p_num1_in
SET @temp_1 = 0
SET @v_divisor = 5
SET @v_condition = 'NO'

WHILE 1=1
BEGIN

    SELECT @temp = @temp + 1 /*Increase temp starting from p_num1*/

   IF @temp = @p_num2_in /*Check if we reached p_num2*/
       /*If yes, leave the LOOP*/
       BREAK

/*Call Procedure 2 to check if number is divisable by 5*/
EXEC username.check_modulo @temp,@v_divisor,@result=@v_condition output

  IF @v_condition = 'YES'
      SELECT @temp_1 = @temp_1 + @temp

END /*WHILE LOOP*/

SELECT @result = @temp_1

RETURN
go
```

> **NOTE:** For the purposes of this walk-through, this procedure was created under the user name Spence. Before executing the DDL above, substitute your user name for the word "username".

Creating Procedure 2

To create Procedure 2, do the following:

1 Start Rapid SQL.

2 Connect to a Microsoft SQL Server 7.0 datasource.

3 On the Datasource menu, click the database node and then click the target database.

> **NOTE:** For this walk-through, we recommend that you select a non-production database.

4   On the Main toolbar, click New.

OR

On the File menu, click New, and then click SQL.

OR

Press CTRL+N.

Rapid SQL opens an SQL Editor in the current workspace.

5   In the SQL Editor, type the DDL for procedure calculate_sum_with_overflow_bug.

NOTE: You must substitute your user name twice in the DDL for this procedure.

6   On the SQL Editor toolbar, click Execute.

Rapid SQL executes the script and creates Procedure 2, then opens the SQL Editor Results Tab with the results of the script execution. If you were not able to create the procedure, check the error messages to determine the problem.

CONFIRMING THE CREATION OF THE PROCEDURES

After you create Procedure 1 and Procedure 2, you can confirm their creation in Rapid SQL's Database Explorer.

Confirming the Creation of the Procedures

To confirm creation of the procedures, do the following:

1   On the Explorer Tab, click the Explorer list, and then click Organize By Owner.

The Explorer Tab refreshes with the new display configuration.

2   On the Explorer Tab, double-click the Databases node, and then double-click the target database node.

Rapid SQL displays the list of object owners.

3   Double-click your user name to display a list of your objects.

4   Double-click Procedures to display a list of procedures and confirm the creation of check_modulo and calculate_sum_with_overflow_bug.

You are now ready to begin testing a procedure.

TESTING A PROCEDURE

After you confirm the creation of the procedures, you must execute the procedure calculate_sum_with_overflow_bug (which includes a bug) to view its error message. This procedure requires two integer input parameters: @p_num1_in int and @p_num2_in int. For all integers between these two integers, this procedure identifies those divisible by 5, and then returns their sum.

CAUTION: When inputting parameters, you must enter the smaller number in the @p_num1_in int box.

Testing a Procedure

To test a procedure, do the following:

1   On the Explorer Tab, right-click calculate_sum_with_overflow_bug, and then click Execute.

   Rapid SQL opens the Procedure Execution window.

2   In the Value column of the @p_num1_in row, type 1.

3   In the Value column of the @p_num2_in row, type 11.

4   Click Execute.

   Rapid SQL compiles the procedure and opens a Results Tab, displaying the sum 15. There are two numbers between 1 and 11 that are divisible by 5: 5, and 10. The sum of these two numbers is 15.

5   On the Explorer Tab, right-click calculate_sum_with_overflow_bug, and then click Execute.

   Rapid SQL again opens the Procedure Execution window.

6   In the Value column of the @p_num1_in row, type 100.

7   In the Value column of the @p_num2_in row, type 121.

8   On the Procedure Execution window toolbar, click Execute.

   Rapid SQL returns an error stating "Arithmetic overflow occurred". You are now ready to Start the Debugging Session.

STARTING THE DEBUGGING SESSION

After you test the procedure, you must open the procedure in Embarcadero SQL Debugger for Microsoft and enter input parameters before debugging. To start a session, do the following:

1   On the Explorer Tab, right-click the procedure, calculate_sum_with_overflow_bug, and then click Debug to start the debug session.

   Rapid SQL extracts the DDL for the procedure into a DDL Editor and opens the Procedure Execution dialog box.

2   In the Value column of the @p_num1_in row, type 100.

3   In the Value column of the @p_num2_in row, type 121.

4   Click Continue.

Rapid SQL closes the dialog box and opens the Embarcadero SQL Debugger interface, which includes the following five windows:

- DDL Editor

- Watch Window

- Variables Window

- Call Stack Window

- Dependency Tree Window

You are now ready to insert breakpoints.

BREAKPOINTS

After you start the debugging session, you must insert a breakpoint into the code of the procedure calculate_sum_with_overflow_bug. Then you must run to the breakpoint. After you run to the breakpoint, Embarcadero SQL Debugger displays a yellow arrow on the red breakpoint icon and populates the Variables Window with values for the following variables:

Variable	Value
@temp	Current number
@p_num2_in	Second input parameter
@p_num1_in	First input parameter
@temp_1	Sum of the numbers, between the input parameters, divisible by 5
@result	Condition of the output parameter
@v_condition	Output parameter
@v_divisor	Divisor

Breakpoints

To insert a breakpoint, do the following:

1   In the DDL Editor, scroll to and click the following line:

```
EXEC username.check_modulo @temp,@v_divisor,@result=@v_condition output
```

NOTE: This line is located near the end of the procedure's code.

2　On the *Debug* menu, click *Breakpoint.*

OR

On the *DDL Editor* toolbar, click Breakpoint.

OR

In the *DDL Editor* window, right-click and then click *Breakpoint.*

OR

Press F9.

Rapid SQL inserts a breakpoint (indicated by dot) next to the number of the target line.

3　On the *Debug* menu, click *Go.*

OR

On the *DDL Editor* toolbar, click Go.

OR

In the *DDL Editor* window, right-click and then click *Go.*

OR

Press F5.

Rapid SQL Embarcadero SQL Debugger for Microsoft displays the value of the variables before the breakpoint in the Variables Window.

You are now ready to [Step Into](#) the code.

STEP INTO

After setting the [breakpoint](#), you must step into the dependent procedure, check_modulo.To step into the dependent procedure, do the following:

1　On the *Debug* menu, click *Step Into.*

OR

On the *DDL Editor* toolbar, click Step Into.

OR

In the DDL Editor window, right-click and then click *Step Into.*

OR

Press F11.

Rapid SQL extracts the DDL for the dependent, nested procedure into the DDL Editor.

2 Step Into again.

Rapid SQL executes the next part of the code and displays the values for the variables in the Variables Window.

The Call Stack Window displays calls to the procedures.

You are now ready to <u>Step Out</u> of the code.

STEP OUT

After you <u>Step Into</u> the modulo_check (nested procedure) code, you must step back out and return to the calculate_sum_with_overflow_bug (outside procedure) code. To step back out and return, do the following:

1 On the *Debug* menu, click *Step Out.*

   OR

   On the *DDL Editor* toolbar, click Step Out.

   OR

   In the DDL Editor window, right-click and then click *Step Out.*

   OR

   Press SHIFT+F11.

   Rapid SQL opens the DDL Editor containing the code for calculate_sum_with_overflow_bug.

2 On the *Debug* menu, click *Go.*

   OR

   On the *DDL Editor* toolbar, click Go.

   OR

   In the *DDL Editor* window, right-click and then click *Go.*

   OR

   Press F5.

   When the value of the variable, @temp is equal to the value of the variable, @p_num2_in, the WHILE LOOP is complete and the Embarcadero SQL Debugger for Microsoft continues to the next executable statement in the code.

3 While monitoring the value of the variables in the Variables Window, continue to click Go to cycle through the WHILE LOOP.

   After executing the SELECT and RETURN statements, Rapid SQL closes the Debugger and opens a DDL Editor to the Results Tab.

   Now you are ready to <u>correct the script</u>.

CORRECTING THE SCRIPT

When you finished Stepping Out of the nested code and encounter the error, you must do the following to fully fix the bug:

- Locate the source of the error
- Scroll to the line in the script displaying the error
- Analyze the code
- Correct the error
- Compile the corrected script

When you first executed the procedure, Rapid SQL displayed the error message "Arithmetic overflow error for data type tinyint, value = 450". According to *Microsoft SQL Server Books Online:* "This error occurs when an attempt is made to convert a float or real data type value into a data type that cannot store the result. This error prevents the operation from being completed."

The data type used in this procedure (TINYINT) stores values from 0 to 255. The sum of the four numbers between 100 and 121 that are divisible by 5 (105, 110, 115, and 120) is 450. But because the TINYINT variable @result can only accept a maximum value of 255, Rapid SQL returns the error message and the procedure fails.

Correcting the Script

To correct the script, do the following:

1 On the Explorer Tab, right-click calculate_sum_with_overflow_bug, and then click Extract.

   Rapid SQL extracts the DDL for the procedure into a DDL Editor.

2 On the Edit toolbar, click Find.

   Rapid SQL opens the Find dialog box.

3 In the Find What box, type TINYINT.

4 Click Find Next.

   Rapid SQL selects the first occurrence of TINYINT.

5 Change the data type for @result from TINYINT to INT.

6 On the DDL Editor toolbar, click Execute to execute the modified script.

   Rapid SQL executes the script and opens the Results Tab.

7 On the Explorer Tab, right-click calculate_sum_with_overflow_bug, and then click Execute.

   Rapid SQL opens the Procedure Execution dialog box.

8 In the Value column of the @p_num1_in row, type 100.

9 In the Value column of the @p_num2_in row, type 121.

10 Click Execute.

Rapid SQL executes the procedure with the new data type and opens the Results Tab, returning the value 450. You successfully corrected the script and debugged the procedure.

EMBARCADERO SQL DEBUGGER FOR ORACLE

Embarcadero SQL Debugger for Oracle is a programming tool that lets you debug functions, procedures and triggers for Oracle versions 7.3.3 or later.

Objects

Using Embarcadero SQL Debugger for Oracle, you can debug the following objects:

- Functions

- Procedures

- Triggers

You can only debug triggers by debugging the functions or procedures that call them. You cannot debug packages, but you can debug the functions and procedures within packages.

> **NOTE:**   You cannot debug any objects contained in the Exclusion List.

> **NOTE:**   Availability of this feature depends on your Rapid SQL licensing. For more information, see Licensing.

> **TIP:**   The Code Analyst is a tool to identify time-consuming lines of code. Code Analyst lets you perform detailed response time analysis on the execution of Procedures and Functions.

The table below describes the sections of this chapter:

Section	Description
Features	This section describes how the Embarcadero SQL Debugger for Oracle helps you identify problems within your code.
Interface	This section describes the Embarcadero SQL Debugger for Oracle graphical interface that includes an editor window and four debug view windows.
Functionality	This section describes the functions of the Embarcadero SQL Debugger for Oracle.
Using Embarcadero Debugger for Oracle	This section describes how to run a debug session.

DEBUGGING FEATURES

Embarcadero SQL Debugger for Oracle is designed to help identify problems within your code. Embarcadero SQL Debugger for Oracle lets you:

- Interactively step through the flow of script execution.

- Examine the value of variables.

- Solve logical problems with your script design.

Embarcadero SQL Debugger for Oracle offers fundamental debugging features, an Oracle Exclusion List and several options to help fine tune debugging, as listed in the table below:

Debugging Feature	Description
Step Into	Lets you execute each instruction step-by-step and step inside a stored object if the object is not on the Exclusion List.
Step Out	Lets you stop stepping through the current object and execute the remainder of the script. This option is only active when the pointer indicates a child-dependent instruction.
Step Over	Lets you execute the current instruction without stepping into any child dependents.
Breakpoints	A position in a program where the debugger stops execution.

To set specific Debugger values on the Options Editor, see Debugger Options.

EXCLUSION LIST

Upon installation, Rapid SQL sets up an Exclusion List on your computer which includes packages that the application cannot debug. The Exclusion List is located in the Rapid SQL directory, at the default installation location C:\Program Files\Embarcadero\Nov2001Shared\deborcex.etd. You can add or remove packages from this file by editing the Exclusion List.

Editing the Exclusion List

To Edit the Exclusion List, do the following:

1 Open the Exclusion List, *deborcex.etd,* in a text editor, such as Notepad or WordPad.

2 To add a package, enter the name of the package at the end of the list. Use the following format: OWNER.OBJECT_NAME.

> **NOTE:** There must be a carriage return after each item on the list.

3   To remove a package from the Exclusion List, delete the package from the list.

> **NOTE:**   Embarcadero SQL Debugger for Oracle does debug a package procedure listed on the Exclusion List.

4   Save the changes to *deborcex.etd.*

EMBARCADERO SQL DEBUGGER FOR ORACLE OPTIONS

You can specify PL/SQL Debugger options from the Debug Tab of Rapid SQL's Options editor. The Debug Tab of the Options Editor lets you set the duration of your debug initialization and debug session, enable DBMS output, and refresh dependencies.

Setting Debugger Options

To set debugger options, do the following:

1   On the *File* menu, click *Options.*

> OR

On the *Main* toolbar, click Options.

Rapid SQL opens the Options Editor.

2   On the Debug Tab, specify debugger options. The table below describes the options available:

Option	Description	Default
Initialization Timeout (seconds)	Specifies the number of seconds Rapid SQL tries to initialize the debugger. If it cannot initialize the debugger in the specified time, a message displays in the Debug Output window.	60
Debug Session Timeout (seconds)	Specifies, in seconds, the length of your debug session.	7200
Enable DBMS Output	Toggles the print output. Enable this option if you use dbms_output.put_line calls in your procedures and you want these lines displayed.	Selected
Refresh Dependencies for each run	Refreshes dependencies for each run. This potentially time-consuming process is useful if the target procedure has rapidly varying dependencies that can require updating during the debugging process.	Cleared

3   Click Close.

Rapid SQL closes the Options Editor.

EMBARCADERO SQL DEBUGGER FOR ORACLE INTERFACE

Embarcadero SQL Debugger for Oracle has a graphical interface that includes an editor window and four debug view windows. When you open a debug session, Rapid SQL extracts the code for the object into an SQL Editor and opens four debug view windows at the bottom of the screen.

> **TIP:** All Embarcadero debuggers display Performance Metrics that let you measure the execution time of each statement in the debug session.

The four debug view windows are optional, dockable, floatable windows designed to help debug your script. Embarcadero SQL Debugger for Oracle's five windows are:

- SQL Editor window
- Watch window
- Variables window
- Call Stack window
- Dependency Tree window

Working with Embarcadero SQL Debugger Windows

Rapid SQL lets you resize, move, dock and float the following Debugger windows:

- Watch
- Variables
- Call Stack
- Dependency Tree

To work with the above windows, do the following:

1   To resize the target window, click its frame and drag it

    Rapid SQL resizes the window.

2   To move and dock the target window, click its grab bar and drag it.

    Rapid SQL moves the window to its new location and docks it with surrounding windows.

3   To float the target window, press Shift, then click its grab bar and drag it.

    Rapid SQL frames the window in its own floating frame and moves the window to its new location.

SQL EDITOR WINDOW

Embarcadero SQL Debugger for Oracle provides an SQL Editor window that displays your code in Read-Only format. When you start debugging, Embarcadero SQL Debugger for Oracle extracts your code into an SQL Editor window, making it editable. The SQL Editor uses the default Rapid SQL syntax coloring.

> **NOTE:** Rapid SQL displays LOB datatypes, and REF CURSOR variables, in the Results Tab.

WATCH WINDOW

Embarcadero SQL Debugger for Oracle provides a Watch window that displays the watch variables for the database object you are debugging and lets you specify variables you want to evaluate or modify while debugging your program. For example, to check what happens when a variable (x) has a value of 100, you can double-click that variable in the SQL Editor, drag it into the Watch window, and change the value to 100. When you execute the script, the debugger uses the value x=100. This window is only visible when the PL/SQL Debugger is active.

> **NOTE:** You can type a fully qualified record variable into the Watch window.

> **NOTE:** When you exit a debug session and reenter it, the Embarcadero SQL Debugger for Oracle retains any watch variables or breakpoints you have set.

Opening and Closing the Watch Window

To open and close the Watch Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Watch.*

    OR

    Press ALT+3.

Setting a Watch Variable

To set a Watch Variable, do the following:

1   In the *SQL Editor,* double-click the target variable and drag it to the *Watch window.*

2   In the *Watch window,* change the value of the variable.

3   On the *SQL Editor* toolbar, click Go.

    Rapid SQL executes the script using the new value of the variable.

Removing a Watch Variable

1   In the *Watch window*, click the target variable and press *DELETE*.

VARIABLES WINDOW

Embarcadero SQL Debugger for Oracle provides a Variables window that displays the local variables and their current values during script execution. You cannot edit variables in the Variables window. This window is only visible when the Debugger is active. If the SQL Editor displays an external database object, and that object is a dependent of the object you are debugging, then the Variables window automatically refreshes and displays the variables for that particular object.

The Embarcadero SQL Debugger for Oracle also lets you monitor the value of your variables while debugging.

Opening and Closing the Variables Window

To open and close the Variables Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Variable.*

OR

Press ALT+4.

Monitoring Variables

To monitor the values of your variables while debugging, do the following:

1   In the SQL Editor, hold the pointer over the target variable.

Rapid SQL opens a ScreenTip displaying the current value of that variable.

CALL STACK WINDOW

Embarcadero SQL Debugger for Oracle provides a Call Stack window that displays the stack of currently active function calls. The Call Stack window is only visible when the PL/SQL Debugger is active.

Opening and Closing the Call Stack Window

To open and close the Call Stack Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Call Stack.*

OR

Press ALT+5.

Using the Call Stack Window

To display a line of code that references the call in the *SQL Editor*, do the following:

1 In the *Call Stack window*, double-click the target line.

In the SQL Editor, Rapid SQL displays a green arrow on the line of the referenced call.

DEPENDENCY TREE WINDOW

Embarcadero SQL Debugger for Oracle provides a Dependency Tree window that displays any external database objects the script accesses. Rapid SQL displays these database objects in a hierarchical tree, with the child objects as database objects accessed by the parent objects. You can use this window to display the code for a dependent database object in the *SQL Editor* window. This window is only visible when the Debugger is active.

Opening and Closing the Dependency Tree Window

To open and close the Dependency Tree Window, do the following:

1 On the *Debug Menu, on the Debug Views sub-menu, select or clear Dependencies.*

OR

Press ALT+6.

Displaying Dependencies

To display the code for a dependent database object in the *SQL Editor* window, do the following:

1 In the *Dependency Tree* window, double-click the target object.

Rapid SQL displays the SQL of the target object in the SQL Editor window.

EMBARCADERO SQL DEBUGGER FOR ORACLE FUNCTIONALITY

Embarcadero SQL Debugger for Oracle offers you the following functionality:

- Input Parameters
- Step Into
- Step Out
- Step Over
- Run to Cursor
- Insert or Remove Breakpoint
- Toggle Breakpoint

- [Go](#)

- [Stop](#)

- [Restart](#)

- [Break](#).

- [Close](#)

To make use of the above functionality, you must first [open a debugging session](#).

INPUT PARAMETERS

Input parameters are set when you first create an object. If the object you want to debug requires input parameters, Rapid SQL opens a Function, Procedure, or Trigger Execution dialog box and prompts you for the input parameters when you [open a debugging session](#).

This dialog box also lets you:

- Save input parameters as *.prm files to preserve specific input parameter configurations.

- Open *.prm files to save the effort of reentering specific input parameters.

- Reset parameters to their default setting.

The table below describes the options and functionality on Procedure Execution dialog box:

The following table describes the options available in this dialog box:

Dialog box component	Description
Owner drop-down list	Displays the current procedure's owner
Procedure drop-down list	Displays the name of the current procedure.
Parameter window	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message appears in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open button	Click to open an Open dialog box, from which you can open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save button	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset button	Click to reset the parameters in the Parameter window to their default values.
Execute button	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

Option	Description
Owner	Displays the current procedure's owner.
Procedure	Displays the name of the current procedure.
Parameter	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message displays in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open	Click to open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset	Click to reset the parameters in the Parameter window to their default values.
Continue	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

> **NOTE:** You cannot debug a script that requires input parameters until you provide input parameters.

STEP INTO

After you <u>open a debugging session</u>, *Step Into* lets you execute the current instruction. If the current instruction makes a call to a stored Oracle object, Embarcadero SQL Debugger for Oracle steps inside the nested child object.

> **NOTE:** Oracle 7.3 has problems running the debugger on an object with cursors.

Step Into

To use the Step Into facility, do the following:

1 On the *Debug* menu, click *Step Into.*

   OR

   On the *SQL Editor* toolbar, click Step Into.

   OR

   In the *SQL Editor* window, right-click and then click *Step Into.*

   OR

   Press F11.

   Embarcadero SQL Debugger for Oracle moves the arrow to execute the current instruction.

STEP OUT

After you open a debugging session, Step Out lets you execute the remainder of the dependent child object and resumes line-by-line, step-debugging in the parent object.

> **NOTE:** Step Out is only active when the pointer indicates a child-dependent instruction.

> **NOTE:** Oracle 7.3 has problems running the debugger on an object with cursors.

Step Out

To use the Step Out facility, do the following:

1 On the *Debug* menu, click *Step Out.*

   OR

   On the *SQL Editor* toolbar, click Step Out.

   OR

   In the *SQL Editor* window, right-click and then click *Step Out.*

   OR

   Press SHIFT+F11.

   Embarcadero SQL Debugger for Oracle stops stepping through the current object and executes the remainder of the script.

STEP OVER

After you open a debugging session, Step Over lets you execute the current instruction without stepping into a nested child object if the instruction makes a call to a dependent object.

> **NOTE:** Oracle 7.3 has problems running the debugger on an object with cursors.

Step Over

To use the Step Over facility, do the following:

1 On the *Debug* menu, click *Step Over.*

 OR

 On the *SQL Editor* toolbar, click Step Over.

 OR

 In the *SQL Editor* window, right-click and then click *Step Over.*

 OR

 Press F10.

 Embarcadero SQL Debugger for Oracle executes the current instruction.

RUN TO CURSOR

After you open a debugging session, Run to Cursor lets you execute all instructions between the yellow arrow and your cursor.

Run to Cursor

To use the Run to Cursor facility, do the following:

1 Scroll down from the yellow arrow to the target line.

2 Click the target line.

 Embarcadero SQL Debugger for Oracle places the cursor on the target line.

3 On the *Debug* menu, click *Run to Cursor.*

 OR

 On the *SQL Editor* toolbar, click Run to Cursor.

 OR

 In the *SQL Editor* window, right-click and then click *Run to Cursor.*

 OR

 Press CTRL+F10.

 Embarcadero SQL Debugger for Oracle executes all instructions between the pointer and your cursor.

INSERT OR REMOVE A BREAKPOINT

A breakpoint is a position in a program where a debugger stops execution. When you start debugging, Embarcadero SQL Debugger for Oracle opens the script in an SQL Editor window. A yellow pointer indicates which line the Debugger executes next. Embarcadero SQL Debugger for Oracle executes all lines of code between the pointer and the first breakpoint. If no breakpoints are present, Embarcadero SQL Debugger for Oracle debugs the entire script.

While debugging, you can set one or more breakpoints in the currently executing object or in any object in the program call stack. Breakpoints can be toggled, temporarily disabled, or enabled, without having to add or remove them. Embarcadero SQL Debugger for Oracle displays each enabled breakpoint as a red dot in the left margin of the SQL Editor Window, and each disabled breakpoint as a red circle.

Rapid SQL stores all breakpoints you set, so that when you debug the same script on separate occasions, you can reuse the same breakpoints.

After you open a debugging session, you can insert a breakpoint on the line where your cursor is located, and you can remove a breakpoint on the line where your cursor is located.

> **NOTE:** Script execution stops at the first breakpoint.

Inserting or Removing a Breakpoint

To insert and remove a breakpoint, do the following:

1   In the SQL Editor window, click the target line of SQL.

2   On the *Debug* menu, click *Breakpoint.*

> OR
>
> On the *SQL Editor* toolbar, click Breakpoint.
>
> OR
>
> In the *SQL Editor* window, right-click and then click *Breakpoint.*
>
> OR
>
> Press F9.
>
> Embarcadero SQL Debugger for Oracle inserts a new breakpoint or removes an existing breakpoint on the target line of code.

TOGGLE BREAKPOINT

After you open a debugging session and insert a breakpoint, Toggle Breakpoint lets you enable or disable that breakpoint. Embarcadero SQL Debugger for Oracle displays each enabled breakpoint as a red dot in the left margin of the SQL Editor Window, and each disabled breakpoint as a red circle. You can toggle any breakpoint in the SQL Editor window. When you exit a debugging session and reenter it, the Embarcadero SQL Debugger for Oracle retains any breakpoints you set.

Toggling a Breakpoint

To toggle a breakpoint, do the following:

1 In the SQL Editor window, click the line of the target breakpoint.

2 On the *Debug* menu, click *Enable/Disable Breakpoint.*

   OR

   On the *SQL Editor* toolbar, click *Enable/Disable Breakpoint.*

   OR

   In the *SQL Editor* window, right-click and then click *Enable/Disable Breakpoint.*

   OR

   Press CTRL+F9.

   Embarcadero SQL Debugger for Oracle toggles the breakpoint indicated by the pointer.

Go

After you [open a debugging session](#), Go lets you execute all instructions, stopping only when it encounters a breakpoint or when the program is complete.

Go

Go use the Go facility, do the following:

1 On the *Debug* menu, click *Go.*

   OR

   On the *SQL Editor* toolbar, click Go.

   OR

   In the *SQL Editor* window, right-click and then click *Go.*

   OR

   Press F5.

   Embarcadero SQL Debugger for Oracle executes all instructions.

Stop

After you [open a debugging session](#), Stop lets you halt the script execution and terminate the session.

Stop

To stop the debugger, do the following:

1 On the *Debug* menu, click *Stop Debugging.*

  OR

  On the *SQL Editor* toolbar, click *Stop Debugging.*

  OR

  In the *SQL Editor* window, right-click and then click *Stop Debugging.*

  OR

  Press SHIFT+F5.

  Embarcadero SQL Debugger for Oracle stops the script execution and terminates the session.

RESTART

After you open a debugging session, Restart lets you terminate the current debug session and open a new one. When the new session opens, Rapid SQL prompts you for new input parameters.

Restart

To restart the debugger, do the following:

1 On the *Debug* menu, click *Restart.*

  OR

  On the *SQL Editor* toolbar, click Restart.

  OR

  In the *SQL Editor* window, right-click and then click *Restart.*

  OR

  Press CTRL+SHIFT+F5.

  Embarcadero SQL Debugger for Oracle restarts the debug session.

BREAK

After you open a debugging session, Break lets you pause the debug session.

Break

To pause the session, do the following:

1   On the *Debug* menu, click *Break.*

> OR

> On the *SQL Editor* toolbar, click Break.

> OR

> In the *SQL Editor* window, right-click and then click *Break.*

> Embarcadero SQL Debugger for Oracle suspends the debug session.

CLOSE

After you [open a debugging session](#), Close lets you close the SQL Editor and the Embarcadero SQL Debugger for Oracle.

Close

To close the SQL Editor and the debugger, do the following:

1   On the *SQL Editor* toolbar, click Close.

> OR

> In the upper right corner of the window, click Close.

> OR

> In the *SQL Editor* window, right-click and then click *Close.*

> Embarcadero SQL Debugger for Oracle closes the debug session.

USING THE EMBARCADERO SQL DEBUGGER FOR ORACLE

This section offers a general overview of how to use Embarcadero SQL Debugger for Oracle's full range of debugging [functionality](#). After you [open a debugging session](#) for any Oracle procedure or function, you can [begin debugging](#).

For more detailed information, see [Debugging a Sample Script](#).

OPENING A DEBUGGING SESSION

When you open a debugging session, Rapid SQL opens the five windows of the Embarcadero SQL Debugger for Oracle interface. If the target script requires input parameters, Rapid SQL opens a Procedure or Function Execution dialog box and prompts you for the necessary input parameters before displaying the target code in the SQL Editor window. When Rapid SQL displays the target script in the SQL Editor window, you can begin debugging.

> **NOTE:** Embarcadero SQL Debugger for Oracle only lets you debug the SQL script of functions, triggers and procedures.

Opening a Debugging Session

To debug a function, trigger or procedure, do the following:

1 On the *Explorer* Tab, click the node of the target function, trigger, or procedure.

   Rapid SQL opens the node and displays two items: Code and Privileges.

2 Under the target object node, double-click Code.

   Rapid SQL opens an SQL *editor displaying the code of the target object.*

3 On the Debug menu, click Start Debugging.

   OR

   On the SQL Editor toolbar, click Debug.

   OR

   In the *SQL Editor* window, right-click and then click *Debug.*

   OR

   Press CTRL+F5.

   If the script requests input parameters, Rapid SQL opens the Procedure or Function Execution dialog box. If the script does not require input parameters, Rapid SQL displays the script in the SQL Editor window for you to begin debugging.

   > **NOTE:** You cannot use the Embarcadero SQL Debugger for Oracle until it has fully initialized.

4 In the Procedure or Function Execution dialog box, specify the appropriate parameters, and then click Continue.

   Rapid SQL displays the script in the SQL Editor window for you to begin debugging.

   > **NOTE:** If the script requires Oracle types (tables, records, or Booleans) as input parameters, the PL/SQL Debugger generates an anonymous block.

DEBUGGING AN SQL SCRIPT

After you open a debugging session and enter any required input parameters, you can begin working with your script in the Debugger.

Debugging an SQL Script

To debug an SQL script, do the following:

1 On the *Debug* menu, click one of the PL/SQL Debugger options (Step Into, Step Over, and so forth) or click *Go*.

   OR

   On the *SQL Editor* toolbar, click one of the PL/SQL Debugger options (Step Into, Step Over, and so on) or click Go.

   NOTE: You can monitor the progress of your debug session in the Variables window.

2 On the *Debug* menu, click *Breakpoint*.

   OR

   On the *SQL Editor* toolbar, click Breakpoint.

   OR

   Press F9.

   NOTE: When you set a breakpoint, the Call Stack window shows what was called before the breakpoint.

   NOTE: You can use the Run to Cursor option to test the lines of code between a breakpoint and your cursor (indicated by the yellow arrow in the SQL Editor).

3 To check your variables, do the following:

   1) In the **SQL Editor**, click a variable in your script and drag it to the **Watch** window.

   2) In the **Watch** window, change the value of the watch variable and then click **Go** to run your script and see the results of the new value.

4 To check record in stored objects, do the following:

   1) Drag the record to the **Watch** window.

   2) In the **Watch** window, change the value of the record, then click **Go** to run your script and see the results of the new value.

5 To check the dependencies, do the following:

   1) In the **Dependency Tree** *window* double-click the target dependent object to extract the code into a new **SQL Editor**.

   2) **Step through** the script while monitoring the Dependency Tree window.

6 When you finish debugging the script, click Close.

   Rapid SQL closes the PL/SQL Debugger SQL Editor.

   NOTE: When you exit a debug session and reenter it, the Embarcadero SQL Debugger for Oracle retains any watch variables or breakpoints you have set.

DEBUGGING A SAMPLE SCRIPT

The Rapid SQL installation includes a sample script intended to walk you through basic debugging functionality. The sample script creates a package that includes functions and procedures that you debug.

> **NOTE:** To create the sample package, you must have CREATE privileges.

Overview

Debugging a Sample Script is divided into three sections that will familiarize you with basic debugging features and functionality. These sections are:

- [Getting Started,](#) which guides you through creating the package you will use in Debugging Sample Script 1 and Debugging Sample Script 2.

- [Debugging Sample Script 1,](#) which guides you through debugging functionality and demonstrates the Embarcadero SQL Debugger for Oracle interface features.

- [Debugging Sample Script 2,](#) which guides you through debugging functionality and error correction.

> **NOTE:** For the purposes of this walk-through we have created this package under the user name DEMO_SPENCE.

GETTING STARTED

The Rapid SQL installation includes a sample script that you execute to create a package containing functions and procedures. These functions and procedures demonstrate basic debugging features available in the Embarcadero SQL Debugger for Oracle

> **NOTE:** To create the sample package, you must have CREATE privileges.

If you create the package included with the Rapid SQL installation, you can delete it and its objects from your system when you finish working with them. The objects to delete are as follows:

- The package COUNT_TIME_INTERVAL

- The package function WEEKEND_DAYS_( )

- The package function WORKING_DAYS_( )

- The package function YEARS_ELAPSED_BETWEEN_( )

- The procedure YEARS_ELAPSED

- The procedure YEARS_ELAPSED_Y2K

Overview

The Getting Started section guides you through:

- Opening the sample debug script.

- Executing the sample debug script.

- Changing the *Explorer* Tab display.

- Confirming the creation of the package, including its functions and procedures.

Getting Started

1   Start Rapid SQL.

2   On the *File* menu, click *Open*. Rapid SQL opens the Open File(s) dialog box.

> **NOTE:**   The sample file used in this example, DEBUGGER_DEMO.sql, is stored in the configured user SQL scripts folder. For details, see <u>Directories Options</u>.

3   In the *Open File(s)* dialog box, navigate to the user SQL scripts folder and open the *DEBUGGER_DEMO.sql* file.

    Rapid SQL opens the What type of file dialog box.

4   On the What type of file dialog box, click The file includes the DDL to create a database object, and then click OK.

    Rapid SQL opens the target script in an SQL Editor.

5   On the *SQL Editor* toolbar, click Execute to execute the script and create the package.

    Rapid SQL executes the target script and opens the SQL Editor Results Tab, displaying the results of the script execution. If you were not able to create the package, check the error messages to determine the problem.

6   On the *Explorer* Tab list, click *Organize by Owner*.

    Rapid SQL displays a list of owners in the Database Explorer.

7   On the *Explorer* Tab, double-click your owner name.

    Rapid SQL displays a list of your schema objects.

8   Under your owner node, double-click the *Packages* node.

    Rapid SQL displays *COUNT_TIME_INTERVAL*, confirming the package's creation. You are now ready to begin debugging <u>Sample Script 1</u> and <u>Sample Script 2.</u>

DEBUGGING SAMPLE SCRIPT 1

Sample Script 1 demonstrates Embarcadero SQL Debugger for Oracle's basic features and functionality with the function WORKING_DAYS( ), which counts the number of business days between two dates.

Overview

Debugging Sample Script 1 is divided into five parts:

- [Starting the Debug Session](#)

- [Entering Input Parameters](#)

- [Inserting Breakpoints](#)

- [Stepping Into](#)

- [Viewing Debug Session Results](#)

SAMPLE SCRIPT 1 - STARTING THE DEBUG SESSION

After you [open and execute](#) DEBUGGER_DEMO.sql, you can begin debugging Sample Script 1. To begin debugging the function WORKING_DAYS( ), you must start a debug session.

Starting the Debug Session

To start the debug session, do the following:

1 On the *Explorer* Tab, under the *Packages* node, double-click the *COUNT_TIME_INTERVAL* node.

   Rapid SQL opens the COUNT_TIME_INTERVAL node and displays the following items:

2 Under the COUNT_TIME_INTERVAL node, double-click *Functions*.

   Rapid SQL opens the Functions node and displays the following items:

3 Under the Functions node, right-click *WORKING_DAYS ( )*, and then click *Debug* to start the debug session.

   Rapid SQL opens the Function Execution dialog box with the current date in the boxes. You are now ready to begin working with [input parameters](#).

SAMPLE SCRIPT 1 - ENTERING INPUT PARAMETERS

After you [start a debugging session](#), you can enter input parameters. You cannot debug a script that requires input parameters until you input those parameters in the Function Execution dialog box.

Input Parameters

To input parameters, do the following:

4 At the end of the *P_START_DATE DATE* row, click the drop-down arrow.

   Rapid SQL opens a calendar.

5 On the calendar, click left arrow to set the month to *November 1999.*

6   Click *1*.

Rapid SQL displays 11/01/1999 in the Value column of P_START_DATE.

7   Click the *P_END_DATE DATE* box, and then click the drop-down arrow.

Rapid SQL opens a new calendar.

8   On the calendar, click left arrow to set the month to *November 1999.*

9   Click *8*.

Rapid SQL displays 11/08/1999 in the Value column of P_END_DATE.

10   Click Continue.

Rapid SQL closes the *Function Execution* dialog box, and then opens the following five Embarcadero SQL Debugger for Oracle interface windows:

- SQL Editor, which displays the SQL code for the function.
- Watch window.
- Variables window.
- Call Stack window.
- Dependency Tree window, which displays the dependent objects.

You are now ready to begin inserting breakpoints.

SAMPLE SCRIPT 1- INSERTING BREAKPOINTS

After you input parameters in the Input Parameters dialog box, you can begin inserting breakpoints. In this example, the breakpoints must be inserted in the extracted dependent object code. After you extract this code, you must locate the target breakpoint lines by searching for the text DBMS_OUTPUT.

Breakpoints

To insert breakpoints, do the following:

1   In the *Dependency Tree window*, double-click the *COUNT_TIME_INTERVAL* package body.

Rapid SQL displays the SQL code for the package body in the SQL Editor window.

2   On the *Edit* toolbar, click Find.

Rapid SQL opens the *Find* dialog box.

3   On the Find dialog box, in the *Find What* box, type *DBMS_OUTPUT.*

4   Click *Find Next*.

In the SQL Editor, Rapid SQL highlights the first occurrence of DBMS_OUTPUT, on line 22.

5   On the *SQL Editor* toolbar, click Breakpoint.

Rapid SQL inserts a breakpoint next to the target line number.

6   On the *Find dialog box*, click Find Next.

Rapid SQL highlights the next occurrence of *DBMS_OUTPUT.*

7   Click *Find Next* a third time.

Rapid SQL highlights the next occurrence of *DBMS_OUTPUT,* on line 35.

8   On the Find dialog box, click Cancel.

Rapid SQL closes the Find dialog box.

9   On the *SQL Editor* toolbar, click Breakpoint to insert a second breakpoint.

You should now have breakpoints set at lines 22 and 35. You are now ready to begin [stepping into](#) the code.

SAMPLE SCRIPT 1- STEPPING INTO
After you [insert breakpoints](#), you can step into the function code.

Step Into
To Step Into the code, do the following:

1   On the *SQL Editor* toolbar, click Go.

Embarcadero SQL Debugger for Oracle begins debugging and runs to the first breakpoint, placing the yellow arrow on line 22.

2   On the *SQL Editor* toolbar, click Step Into .

Embarcadero SQL Debugger for Oracle moves the yellow arrow to the next line of the code.

3   Click Step Into again to enter the LOOP block.

Embarcadero SQL Debugger for Oracle displays the value of the variables in the Variables window.

4   Click Step Into again to start moving through the LOOP block.

In the Variables window, Embarcadero SQL Debugger for Oracle updates the value of variable v_currdate from 01-NOV-1999 to 02-NOV-1999.

5   Click Step Into two more times.

In the Variables window, Embarcadero SQL Debugger for Oracle updates the value of v_theday from NULL to Tuesday.

> **NOTE:**   If you continued stepping through the LOOP block, the Embarcadero SQL Debugger for Oracle would continue to update v_currdate and v_theday until v_currdate is greater than p_end_date.

6   On the *SQL Editor* toolbar, click Go.

Embarcadero SQL Debugger for Oracle runs to the next breakpoint.

7   On the *SQL Editor* toolbar, click Go once more.

Rapid SQL PL/SQL concludes the debug session and displays the [Debug Session Results box](#).

SAMPLE SCRIPT 1 - VIEWING DEBUG SESSION RESULTS

After [Stepping Into](#) and running to the end of the code, Embarcadero SQL Debugger for Oracle displays a Debug Session Results box containing the following information:

• Variable output

• DBMS_OUTPUT

> **NOTE:**   In this example, the Embarcadero SQL Debugger for Oracle displays a Debug Session Results box because the sample program includes DBMS_OUTPUT.

Debug Session Results

1   Click *OK*.

Rapid SQL closes the *Debug Session Results* box and terminates your debug session.

DEBUGGING SAMPLE SCRIPT 2

Sample Script 2 demonstrates Embarcadero SQL Debugger for Oracle's functionality when used on a function containing a bug which prevents it from executing successfully. The buggy function, WEEKEND_DAYS( ), requires input parameters and counts the number of weekend days between two dates. In this section you must use Embarcadero SQL Debugger for Oracle to identify the bug, and then correct the script so that it can execute successfully.

Overview

Debugging Sample Script 2 is divided into six parts:

• [Executing the Function](#)

• [Starting the Debug Session](#)

- Entering Input Parameters

- Inserting Breakpoints

- Stepping Into

- Correcting the Function

SAMPLE SCRIPT 2 - EXECUTING THE FUNCTION

After you open and execute DEBUGGER_DEMO.sql, you can begin debugging Sample Script 2. To begin debugging the function WEEKEND_DAYS ( ), you must first execute the function to discover the type of error it returns when it fails to execute.

Executing the Function

To execute the function, do the following:

1 On the *Explorer* Tab, under the *Packages* node, double-click the *COUNT_TIME_INTERVAL* node.

   Rapid SQL opens the COUNT_TIME_INTERVAL node and displays the following items:

2 Double-click the *Functions* node.

   Rapid SQL opens the Functions node and displays the following items:

3 Click *WEEKEND_DAYS ( )*, then right-click it and click *Execute*.

   Rapid SQL opens the Function Execution dialog box.

4 In the Value column of the *P_START_DATE* row, type *11/01/1999*.

5 In the Value column of the *P_END_DATE* row, type *11/30/1999*.

6 Click Execute.

   Rapid SQL attempts to execute the function but returns an error indicating that the character string buffer is too small.

   You are now ready to start the debug session.

SAMPLE SCRIPT 2 - STARTING THE DEBUG SESSION

After you unsuccessfully execute the function WEEKEND_DAYS( ) and Rapid SQL displays the nature of its execution error, you can start a debugging session to determine the actual cause of the error.

Starting the Debugging Session

To start the debugging session, do the following:

1   On the *Explorer* Tab, under the *COUNT_TIME_INTERVAL node, under the Functions* node, right-click *WEEKEND_DAYS ( )*, and then click *Debug* to start the debug session.

    Rapid SQL opens the Function Execution dialog box.

    You are now ready to begin <u>entering input parameters</u>.

SAMPLE SCRIPT 2 - ENTERING INPUT PARAMETERS

After you <u>start the debug session</u>, you can enter input parameters in the Function Execution dialog box.

Entering Input Parameters

To enter the input parameters, do the following:

1   At the end of the *P_START_DATE* row, click the drop-down arrow.

    Rapid SQL opens a calendar.

2   On the calendar, click left arrow to set the month to *November 1999.*

3   Click *1.*

    Rapid SQL displays 11/01/1999 in the Value column of the P_START_DATE row.

4   At the end of the *P_END_DATE* row, click the drop-down arrow.

    Rapid SQL opens a new calendar.

5   On the calendar, click left arrow to set the month to *November 1999.*

6   Click *30.*

    Rapid SQL displays 11/08/1999 in the Value column of the P_END_DATE row.

7   Click Continue.

    Rapid SQL closes the *Function Execution* dialog box, and then opens the following five Embarcadero SQL Debugger for Oracle interface windows:

    • <u>SQL Editor</u>, which displays the SQL code for the function.

    • <u>Watch window.</u>

    • <u>Variables window.</u>

    • <u>Call Stack window.</u>

    • <u>Dependency Tree window</u>, which displays the dependent objects.

    You are now ready to begin <u>inserting breakpoints</u>.

SAMPLE SCRIPT 2- INSERTING BREAKPOINTS

After you enter input parameters, you can begin inserting breakpoints. In this example, the breakpoints must be inserted in the extracted dependent object code. After you extract this code, you must locate the target breakpoint lines by searching for a particular line of code.

Breakpoints

To insert breakpoints, do the following:

1   In the *Dependency Tree window*, double-click the *COUNT_TIME_INTERVAL* package body.

    Rapid SQL displays the SQL code for the package body in the SQL Editor window.

2   On the *Edit* toolbar, click Find.

    Rapid SQL opens the *Find* dialog box.

3   On the Find dialog box, in the *Find What* box, type *Function weekend_days, and then click Find Next.*

    Embarcadero SQL Debugger for Oracle highlights the first occurrence of Function weekend_days.

4   On the Find dialog box, click Cancel.

    Rapid SQL closes the Find dialog box.

5   Click line 60, the first line of executable code:

6   On the *SQL Editor* toolbar, click Breakpoint.

    Rapid SQL inserts a breakpoint next to the line number.

7   Click Go to start debugging and run to the breakpoint.

    Embarcadero SQL Debugger for Oracle places the yellow arrow on line 60 and populates the Variables window with the first set of variables in the function code.

    Embarcadero SQL Debugger for Oracle also populates the Call Stack window with everything called before the breakpoint.

    You are now ready to begin stepping into the function.

SAMPLE SCRIPT 2- STEPPING INTO

After you set and run to the breakpoint, you can step into the function to locate the cause of the error. To locate the cause of the error, you must monitor the Variables window. As you step through the code, the Variables window updates with the value of the variables.

Step Into

To step into the function, do the following:

1 On the *SQL Editor* toolbar, click Step Into.

   Rapid SQL moves the yellow arrow to the next line of the code, line 64.

2 On the *SQL Editor* toolbar, click Step Into.

   Embarcadero SQL Debugger for Oracle's Variables window updates the value of v_currdate to 02-NOV-1999.

3 On the *SQL Editor* toolbar, click Step Into.

   Rapid SQL moves the yellow arrow to the next line of the code, line 66.

4 On the *SQL Editor* toolbar, click Step Into.

   Rapid SQL moves the yellow arrow to the next line of the code, line 67, and, in the Variables window, updates the value of v_theday to Tuesday.

5 On the *SQL Editor* toolbar, click Step Into.

   Rapid SQL moves the yellow arrow back to line 64 to repeat the loop.

6 On the *SQL Editor* toolbar, click Step Into.

   Embarcadero SQL Debugger for Oracle's Variables window updates the value of v_currdate to 03-NOV-1999.

7 On the *SQL Editor* toolbar, click Step Into.

   Rapid SQL moves the yellow arrow to the next line of the code, line 66.

8 On the *SQL Editor* toolbar, click Step Into.

   The Embarcadero SQL Debugger for Oracle locates the error. The application terminates the debug session, returns an error indicating that the numeric or value character string buffer is too small, extracts the COUNT_TIME_INTERVAL package code into an SQL Editor, and returns an error indicating the line on which the code failed. You are now ready to [correct the script](#).

SAMPLE SCRIPT 2 - CORRECTING THE SCRIPT

After you [step through the SQL code and locate the error](#), you can correct the bug in Sample Script 2. When Embarcadero SQL Debugger for Oracle locates an error, it extracts the target package body into an SQL Editor. To correct this script, you must do the following:

- Scroll to the incorrect line in the script

- Analyze the code

- Correct the error

- Execute the corrected SQL script

- Execute the WEEKEND_DAYS ( ) function

The code in Sample Script 2 fails on line 66, returning an error when the variable v_theday increments from the value Tuesday to the value Wednesday. The cause of this error is found in the declarations section of the function script, where the width of the VARCHAR2 variable v_theday is set to 8. Because "Wednesday" includes nine characters, the value of the variable v_theday fails when it attempts to place a nine-character value in an eight-character variable. To correct this error, you must increase the width of the variable v_theday to accommodate nine characters.

Correcting the Script

To correct the script, do the following:

1   On the *Explorer* Tab, under the *Packages* node, under the *COUNT_TIME_INTERVAL. node*, right-click *Package Body*, and then click *Extract.*

    Rapid SQL extracts the package body into an SQL Editor.

2   In the *SQL Editor*, scroll to line 57, the line defining the variable *v_theday.*

3   On line 57, change the value of the width from *8* to *9.*

4   On the *SQL Editor* toolbar, click Execute to execute the script.

    Rapid SQL successfully executes the script.

5   On the *Explorer* Tab, under the *COUNT_TIME_INTERVAL package* node, under the *Functions* node, click *WEEKEND_DAYS ( ).*

6   Right-click *WEEKEND_DAYS ( )*, and then click Execute.

    Rapid SQL opens the Function Execution dialog box.

7   In the Value column of the *P_START_DATE* row, type *11/01/1999.*

8   In the Value column of the *P_END_DATE* row, type *11/30/1999.*

9   Click Execute.

    Rapid SQL successfully executes the corrected function.

EMBARCADERO SQL DEBUGGER FOR SYBASE ASE

Embarcadero SQL Debugger for Sybase is a programming tool that lets you debug Sybase objects in the following Sybase versions:

- 12

- 12.0 (special version)

- 12.5

Objects

Using Embarcadero SQL Debugger for Sybase, you can debug the following objects:

- Procedures
- Triggers

You can only debug triggers by debugging the procedures that call them.

> **NOTE:** Availability of this feature depends on your Rapid SQL licensing. For more information, see Licensing.

> **TIP:** The Code Analyst is a tool to identify time-consuming lines of code. Code Analyst lets you perform detailed response time analysis on the execution of Procedures and Functions.

The table below describes the sections of this chapter:

Section	Description
Features	This section describes how Embarcadero SQL Debugger for Sybase helps you identify problems within your code.
Interface	This section describes Embarcadero SQL Debugger for Sybase's graphical interface, which includes an editor window and four debug view windows.
Functionality	This section describes the way in which Embarcadero SQL Debugger for Sybase functions.
Using Embarcadero SQL Debugger for Sybase	This section describes how to run a debug session.

EMBARCADERO SQL DEBUGGER FOR SYBASE FEATURES

Embarcadero SQL Debugger for Sybase is designed to help identify problems within your code. Embarcadero SQL Debugger for Sybase lets you:

- Interactively step through the flow of script execution.
- Examine the value of variables.
- Solve logical problems with your script design.

Embarcadero SQL Debugger for Sybase offers fundamental debugging features and several options to help fine tune debugging, as listed in the table below:

Debugging Feature	Description
Step Into	Lets you execute each instruction step-by-step and step inside a stored object.

Debugging Feature	Description
Step Out	Lets you stop stepping through the current object and execute the remainder of the script. This option is only active when the pointer indicates a child-dependent instruction.
Step Over	Lets you execute the current instruction without stepping into any child dependents.
Breakpoints	A position in a program where the debugger stops execution.

To set specific Debugger values on Rapid SQL's Options Editor, see Debugger Options.

EMBARCADERO SQL DEBUGGER FOR SYBASE OPTIONS

You can specify Debugger options from the Debug Tab of Rapid SQL's Options editor. The Debug Tab of the Options Editor lets you set the duration of your debug initialization and debug session, enable DBMS output, and refresh dependencies.

Setting Debugger Options

To set debugger options, do the following:

1   On the *File* menu, click *Options.*

    OR

    On the *Main* toolbar, click Options.

    Rapid SQL opens the Options Editor.

2   On the Debug Tab, specify debugger options. The table below describes the options available:

Option	Description	Default
Initialization Timeout (seconds)	Specifies the number of seconds Rapid SQL tries to initialize the debugger. If it cannot initialize the debugger in the specified time, a message displays in the Debug Output window.	60
Debug Session Timeout (seconds)	Specifies, in seconds, the length of your debug session.	7200
Enable DBMS Output	Toggles the print output. Enable this option if you use dbms_output.put_line calls in your procedures and you want these lines displayed.	Selected
Refresh Dependencies for each run	Refreshes dependencies for each run. This potentially time-consuming process is useful if the target procedure has rapidly varying dependencies that can require updating during the debugging process.	Cleared

3   Click Close.

Rapid SQL closes the Options Editor.

EMBARCADERO SQL DEBUGGER FOR SYBASE INTERFACE

Embarcadero SQL Debugger for Sybase has a graphical interface that includes an editor window and four debug view windows. When you open a debug session, Rapid SQL extracts the code for the object into a DDL Editor and opens four debug view windows at the bottom of the screen.

> **TIP:**   All Embarcadero debuggers display Performance Metrics that let you measure the execution time of each statement in the debug session.

The four debug view windows are optional, dockable windows designed to help debug your script. Embarcadero SQL Debugger for Sybase's five windows are:

- DDL Editor window

- Watch window

- Variables window

- Call Stack window

- Dependency Tree window

Working with Debugger Windows

Rapid SQL lets you resize, move, dock and float the following Debugger windows:

- Watch

- Variables

- Call Stack

- Dependency Tree

To work with the above windows, do the following:

1   To resize the target window, click its frame and drag it.

Rapid SQL resizes the window.

2   To move and dock the target window, click its grab bar and drag it.

Rapid SQL moves the window to its new location and docks it with surrounding windows.

3   To float the target window, press Shift, then click its grab bar and drag it.

Rapid SQL frames the window in its own floating frame and moves the window to its new location.

DDL Editor Window

Embarcadero SQL Debugger for Sybase provides a DDL Editor window that displays your code in read-only format. When you start debugging, Embarcadero SQL Debugger for Sybase extracts your code into a DDL Editor window, making it editable. The DDL Editor uses the default Rapid SQL syntax coloring.

Watch Window

Embarcadero SQL Debugger for Sybase provides a Watch window that displays the watch variables for the database object you are debugging and lets you specify variables you want to evaluate or modify while debugging your program. For example, to check what happens when a variable (x) has a value of 100, you can double-click that variable in the DDL Editor and drag it into the Watch window. In the Watch window, change the value to 100. When you execute the script, the debugger uses the value x=100. This window is only visible when Embarcadero SQL Debugger for Sybase is active.

> **NOTE:** You can type a fully qualified record variable into the Watch window.

> **NOTE:** When you exit a debug session and reenter it, Embarcadero SQL Debugger for Sybase retains any watch variables or breakpoints you have set.

Opening and Closing the Watch Window

To open and close the Watch Window, do the following:

1 On the *Debug Menu, on the Debug Views sub-menu, select or clear Watch.*

   OR

   Press ALT+3.

Setting a Watch Variable

To set a Watch Variable, do the following:

1 In the *DDL Editor,* double-click the target variable and drag it to the *Watch window.*

2 In the *Watch window, in the Value column,* change the value of the variable.

3 On the *DDL Editor* toolbar, click Go to execute the script using the new value of the variable.

   Rapid SQL executes the script using the new value of the variable.

Removing a Watch Variable

To remove a Watch Variable, do the following:

1   In the *Watch window*, delete the variable.

VARIABLES WINDOW

Embarcadero SQL Debugger for Sybase provides a Variables window that displays the local variables and their current values during script execution. You cannot edit variables in the Variables window. This window is only visible when the Debugger is active. If the DDL Editor displays an external database object, and that object is a dependent of the object you are debugging, then the Variables window automatically refreshes and displays the variables for that external object.

Embarcadero SQL Debugger for Sybase also lets you monitor the value of your variables while debugging.

Opening and Closing the Variables Window

To open and close the Variables Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Variable.*

OR

Press ALT+4.

Monitoring Variables

To monitor the values of your variables while debugging, do the following:

1   In the DDL Editor, hold the pointer over the target variable.

Rapid SQL opens a ScreenTip displaying the current value of that variable.

CALL STACK WINDOW

Embarcadero SQL Debugger for Sybase provides a Call Stack window that displays the stack of currently active calls. The Call Stack window is only visible when the Debugger is active.

Opening and Closing the Call Stack Window

To open and close the Call Stack Window, do the following:

1   On the *Debug Menu, on the Debug Views sub-menu, select or clear Call Stack.*

OR

Press ALT+5.

Using the Call Stack Window

To display a line of code that references the call in the *DDL Editor*, do the following:

1 In the *Call Stack window*, double-click the target line.

   In the DDL Editor, Rapid SQL displays a green arrow on the line of the the referenced call.

DEPENDENCY TREE WINDOW

Embarcadero SQL Debugger for Sybase provides a Dependency Tree window that displays any external database objects accessed by the script. Rapid SQL displays these database objects in a hierarchical tree, with the child objects as database objects accessed by the parent objects. You can use this window to display the code for a dependent database object in the *DDL Editor* window. This window is only visible when the Debugger is active.

Opening and Closing the Dependency Tree Window

To open and close the Dependency Tree Window, do the following:

1 On the *Debug Menu, on the Debug Views sub-menu, select or clear Dependencies.*

   OR

   Press ALT+6.

Displaying Dependencies

To display the code for a dependent database object in the *DDL Editor* window, do the following:

1 In the *Dependency Tree* window, double-click the target object.

   Rapid SQL displays the SQL of the target object in the DDL Editor window.

EMBARCADERO SQL DEBUGGER FOR SYBASE FUNCTIONALITY

Embarcadero SQL Debugger for Sybase offers you the following functionality:

- Input parameters
- Step Into
- Step Out
- Step Over
- Run to Cursor
- Insert or Remove Breakpoint
- Toggle Breakpoint

- [Go](#)

- [Stop](#)

- [Restart](#)

- [Break](#)

- [Close](#)

To make use of the above functionality, you must first [open a debugging session](#).

INPUT PARAMETERS

Input parameters are set when you first create an object. If the object you want to debug requires input parameters, Rapid SQL opens a Procedure Execution dialog box and prompts you for the input parameters when you [open a debugging session](#).

This dialog box also lets you:

- Save input parameters as *.prm files to preserve specific input parameter configurations.

- Open *.prm files to save the effort of reentering specific input parameters.

- Reset parameters to their default setting.

The table below describes the options and functionality on Procedure Execution dialog box:

The following table describes the options available in this dialog box:

Dialog box component	Description
Owner drop-down list	Displays the current procedure's owner
Procedure drop-down list	Displays the name of the current procedure.
Parameter window	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message appears in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open button	Click to open an Open dialog box, from which you can open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save button	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset button	Click to reset the parameters in the Parameter window to their default values.
Execute button	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

Option	Description
Owner	Displays the current procedure's owner.
Procedure	Displays the name of the current procedure.
Parameter	Specify the required input parameters in this window. If input parameters are not required for the execution of the target procedure, a message displays in this window, stating that the procedure "has no input parameters. Press execute to run it."
Open	Click to open an existing *.prm file. The saved parameters immediately populate the dialog box upon opening.
Save	Click to save the values of your input parameters as a *.prm file. You can reopen a saved *.prm file from this dialog box at any time.
Reset	Click to reset the parameters in the Parameter window to their default values.
Continue	Click to execute the procedure once you have entered values for all required parameters in the Parameter window.

> **NOTE:** You cannot debug a script that requires input parameters until you provide input parameters.

STEP INTO

After you [open a debugging session](), *Step Into* lets you execute the current instruction. If the current instruction makes a call to a stored Sybase object, Embarcadero SQL Debugger for Sybase steps inside the nested child object.

Step Into

To use the Step Into facility, do the following:

1 On the *Debug* menu, click *Step Into*.

   OR

   On the *DDL Editor* toolbar, click Step Into.

   OR

   In the *DDL Editor* window, right-click and then click *Step Into*.

   OR

   Press F11.

   Embarcadero SQL Debugger for Sybase moves the arrow to execute the current instruction.

STEP OUT

After you <u>open a debugging session</u>, Step Out lets you execute the remainder of the dependent child object and resumes line-by-line, step-debugging in the parent object.

> **NOTE:** Step Out is only active when the pointer indicates a child-dependent instruction.

Step Out

To use the Step Out facility, do the following:

1 On the *Debug* menu, click *Step Out.*

   OR

   On the *DDL Editor* toolbar, click Step Out.

   OR

   In the *DDL Editor* window, right-click and then click *Step Out.*

   OR

   Press SHIFT+F11.

   Embarcadero SQL Debugger for Sybase stops stepping through the current object and executes the remainder of the script.

STEP OVER

After you <u>open a debugging session</u>, Step Over lets you execute the current instruction without stepping into a nested child object if the instruction makes a call to a dependent object.

Step Over

To use the Step Over facility, do the following:

1 On the *Debug* menu, click *Step Over.*

   OR

   On the *DDL Editor* toolbar, click Step Over.

   OR

   In the *DDL Editor* window, right-click and then click *Step Over.*

   OR

   Press F10.

   Embarcadero SQL Debugger for Sybase executes the current instruction.

RUN TO CURSOR

After you open a debugging session, Run to Cursor lets you execute all instructions between the yellow arrow and your cursor.

Run to Cursor

To use the Run to Cursor facility, do the following:

1   *Scroll down from the yellow arrow to the target line.*

2   Click the target line.

Embarcadero SQL Debugger for Sybase places the cursor on the target line.

3   On the *Debug* menu, click *Run to Cursor.*

OR

On the *DDL Editor* toolbar, click Run to Cursor.

OR

In the *DDL Editor* window, right-click and then click *Run to Cursor.*

OR

Press CTRL+F10.

Embarcadero SQL Debugger for Sybase executes all instructions between the yellow arrow and your cursor.

INSERT OR REMOVE A BREAKPOINT

A breakpoint is a position in a program where a debugger stops execution. When you start debugging, Embarcadero SQL Debugger for Sybase opens the script in a DDL Editor window. A yellow pointer indicates which line the Debugger executes next. Embarcadero SQL Debugger for Sybase executes all lines of code between the pointer and the first breakpoint. If no breakpoints are present, Embarcadero SQL Debugger for Sybase debugs the entire script.

While debugging, you can set one or more breakpoints in the currently executing object or in any object in the program call stack. Breakpoints can be toggled (temporarily disabled or enabled) without having to remove or reinsert them. Embarcadero SQL Debugger for Sybase displays each enabled breakpoint as a red dot in the left margin of the DDL Editor Window, and each disabled breakpoint as a red circle.

Rapid SQL stores all breakpoints you set so that when you debug the same script on separate occasions, you can reuse the same breakpoints each time.

After you open a debugging session, you can insert a breakpoint on the line where your cursor it located, and you can remove a breakpoint on the line where your cursor is located.

> **NOTE:**   Script execution stops at the first breakpoint.

Inserting or Removing a Breakpoint

To insert or remove a breakpoint, do the following:

1   In the DDL Editor window, click the target line of SQL.

2   On the *Debug* menu, click *Breakpoint.*

OR

On the *DDL Editor* toolbar, click Breakpoint.

OR

In the *DDL Editor* window, right-click and then click *Breakpoint.*

OR

Press F9.

Embarcadero SQL Debugger for Sybase inserts a new breakpoint or removes an existing breakpoint on the target line of code.

TOGGLE BREAKPOINT

After you [open a debugging session](#) and [insert a breakpoint](#), Toggle Breakpoint lets you enable or disable that breakpoint. Embarcadero SQL Debugger for Sybase displays each enabled breakpoint as a red dot in the left margin of the DDL Editor Window, and each disabled breakpoint as a red circle. You can toggle any breakpoint in the DDL Editor window. When you exit a debugging session and reenter it, the Embarcadero SQL Debugger for Sybase retains any breakpoints you set.

Toggling a Breakpoint

To toggle a breakpoint, do the following:

1   In the DDL Editor window, click the line of the target breakpoint.

2   On the *Debug* menu, click *Enable/Disable Breakpoint.*

OR

On the *DDL Editor* toolbar, click Enable/Disable Breakpoint.

OR

In the *DDL Editor* window, right-click and then click *Enable/Disable Breakpoint.*

OR

Press CTRL+F9.

Embarcadero SQL Debugger for Sybase toggles the breakpoint indicated by the pointer.

Go

After you <u>open a debugging session</u>, Go lets you execute all instructions, stopping only when it encounters a breakpoint or when the program is complete.

Go

To use the Go facility, do the following:

1   On the *Debug* menu, click *Go.*

> OR

> On the *DDL Editor* toolbar, click Go.

> OR

> In the *DDL Editor* window, right-click and then click *Go.*

> OR

> Press F5.

> Embarcadero SQL Debugger for Sybase executes all instructions.

Stop

After you <u>open a debugging session</u>, Stop lets you halt the script execution and terminate the session.

Stop

To stop the debugger, do the following:

1   On the *Debug* menu, click *Stop Debugging.*

> OR

> On the DDL Editor toolbar, click Stop Debugging.

> OR

> In the *DDL Editor* window, right-click and then click *Stop Debugging.*

> OR

> Press SHIFT+F5.

> Embarcadero SQL Debugger for Sybase stops the script execution and terminates the session.

RESTART

After you open a debugging session, Restart lets you terminate the current debug session and open a new one. When the new session opens, Rapid SQL prompts you for new input parameters.

Restart

To restart the debugger, do the following:

1   On the *Debug* menu, click *Restart.*

OR

On the *DDL Editor* toolbar, click Restart.

OR

In the *DDL Editor* window, right-click and then click *Restart.*

OR

Press CTRL+SHIFT+F5.

Embarcadero SQL Debugger for Sybase restarts the debug session.

BREAK

After you open a debugging session, Break lets you pause the debug session.

Break

To pause the debugger, do the following:

1   On the *Debug* menu, click *Break.*

OR

On the *DDL Editor* toolbar, click Break.

OR

In the *DDL Editor* window, right-click and then click *Break.*

Embarcadero SQL Debugger for Sybase suspends the debug session.

CLOSE

After you open a debugging session, Close lets you close the DDL Editor and Embarcadero SQL Debugger for Sybase.

Close

To close the DDL Editor and debugger, do the following:

1   On the *DDL Editor* toolbar, click Close.

> OR

> In the upper right corner of the window, click Close.

> OR

> In the *DDL Editor* window, right-click and then click *Close.*

> Embarcadero SQL Debugger for Sybase closes the debug session.

USING EMBARCADERO SQL DEBUGGER FOR SYBASE

This section offers a general overview of how to use Embarcadero SQL Debugger for Sybase's full range of debugging <u>functionality</u>. After you <u>open a debugging session</u> for any Sybase procedure or trigger, you can <u>begin debugging</u>.

OPENING A DEBUGGING SESSION

When you open a debugging session, Rapid SQL opens the <u>five windows</u> of the Embarcadero SQL Debugger for Sybase interface. If the target script requires input parameters, Rapid SQL opens a Procedure Execution dialog box and prompts you for the necessary input parameters before displaying the target code in the DDL Editor window. When Rapid SQL displays the target script in the DDL Editor window, you can <u>begin debugging</u>.

> **NOTE:**   Embarcadero SQL Debugger for Sybase only lets you debug the SQL script of triggers and procedures.

Opening a Debugging Session

To debug a trigger or procedure, do the following:

1   On the *Explorer* Tab, click the node of the target procedure.

> Rapid SQL opens the node and displays two items: Code and Privileges.

2   Under the target object node, double-click Code.

> Rapid SQL opens an DDL Editor *displaying the code of the target object.*

3   On the Debug menu, click Start Debugging.

OR

On the DDL Editor toolbar, click Debug.

OR

In the *DDL Editor* window, right-click and then click *Debug.*

OR

Press CTRL+F5.

If the script requests input parameters, Rapid SQL opens a Procedure Execution dialog box. If the script does not require input parameters, Rapid SQL displays the script in the DDL Editor window for you to <u>begin debugging</u>.

> **NOTE:**   You cannot use Embarcadero SQL Debugger for Sybase until it has fully initialized.

4   In the Procedure Execution dialog box, type the parameters, and then click *OK*.

Rapid SQL displays the script in the DDL Editor window for you to <u>begin debugging</u>.

> **NOTE:**   If the script requires Sybase types (tables, records, or Booleans) as input parameters, Embarcadero SQL Debugger for Sybase generates an anonymous block.

DEBUGGING AN SQL SCRIPT

After you <u>open a debugging session</u> and enter any required input parameters, you can begin working with your script in Embarcadero SQL Debugger for Sybase.

Debugging an SQL Script

To debug a SQL script, do the following:

1   On the *Debug* menu, click an Embarcadero SQL Debugger for Sybase option (<u>Step Into</u>, <u>Step Over</u>, and so forth) or click *Go*.

OR

On the *DDL Editor* toolbar, click an Embarcadero SQL Debugger for Sybase option (<u>Step Into</u>, <u>Step Over</u>, and so on) or click Go.

> **NOTE:**   You can monitor the progress of your debug session in the Variables window.

2   On the *Debug* menu, click *Breakpoint.*

OR

On the *DDL Editor* toolbar, click Breakpoint.

> **NOTE:**   When you set a breakpoint, the Call Stack window shows what was called before the breakpoint.

> **NOTE:**   You can use the Run to Cursor option to test the lines of code between a breakpoint and your cursor (indicated by the yellow arrow in the DDL Editor).

3   To check your variables, do the following:

   1) In the **DDL Editor**, click a variable in your script and drag it to the **Watch** window.

   2) In the **Watch** window, change the value of the watch variable and then click **Go** to run your script and see the results of the new value.

4   To check record in stored objects, do the following:

   1) Drag the record to the **Watch** window.

   2) In the **Watch** window, change the value of the record, then click **Go** to run your script and see the results of the new value.

5   To check the dependencies, do the following:

   1) In the **Dependency Tree** *window* double-click the target dependent object to extract the code into a new **DDL Editor**.

   2) **Step through** the script while monitoring the **Dependency Tree** window.

6   When you finish debugging the script, click Close.

Rapid SQL closes an Embarcadero SQL Debugger for Sybase's DDL Editor.

> **NOTE:**   When you exit a debug session and reenter it, Embarcadero SQL Debugger for Sybase retains any watch variables or breakpoints you have set.

RAPID SQL PL/SQL PROFILER

The Rapid SQL PL/SQL Profiler module lets you capture metrics of various PL/SQL programmable objects as they are executed in the database. Developers can use data collected in profile sessions to improve performance of PL/SQL code execution. Rapid SQL PL/SQL Profiler collects and stores data in database tables and helps identify and isolate performance problems, and provide code coverage information. The Rapid SQL PL/SQL Profiler lets you:

- Graphically browse PL/SQL profiling data within the Explorer Tab

- View profiling data in the right pane of the application, which is populated as you navigate the Explorer Tab

- Start and stop PL/SQL profiling sessions with a single click

- Graphically analyze time spent in each programmable object (unit)

- Graphically analyze time spent in each source code line of a unit

The table below describes the sections of this chapter:

Section	Description
Setting Up the Profiler	This section describes the process of setting up Rapid SQL PL/SQL Profiler.
Profiler Functionality	This section describes the functionary of Rapid SQL PL/SQL Profiler.
Using the Profiler	This section describes how to run a profile session.

> **NOTE:** Availability of this feature depends on your Rapid SQL licensing. For more information, see Licensing.

For more information, see:

- Setting Up Rapid SQL PL/SQL Profiler
- Rapid SQL PL/SQL Profiler Explorer
- Rapid SQL PL/SQL Profiler Functionality
- Using Rapid SQL PL/SQL Profiler

SETTING UP RAPID SQL PL/SQL PROFILER

The profiling tables must be on the server before using the Rapid SQL PL/SQL Profiler. The first time you open the PL/SQL Profiler, Rapid SQL checks the server for the profiling tables. If the profiling tables are not on the server, Rapid SQL automatically installs profiling tables on the server.

RAPID SQL PL/SQL PROFILER EXPLORER

The Rapid SQL PL/SQL Profiler displays profiling data in the right pane of the application, which is populated as you navigate the Explorer Tab.

The table below describes the nodes of the Rapid SQL PL/SQL Profiler Explorer and the corresponding information in the right pane of the application:

Node	Right pane information
PL/SQL Code Profiling	Contain all Comment, Run ID and Run Date\time data that is current stored in the Profiling tables.
Label\Comment level	Contains all Run ID and Run Date\time data for the specific Label\Comment.

Node	Right pane information
Run level	Contains all Unit, Unit Name, Unit Type, Run Date\time data for the specific Run ID.

RAPID SQL PL/SQL PROFILER FUNCTIONALITY

Rapid SQL PL/SQL Profiler offers you the following functionality:

- Start

- Flush

- Run Summary

- Run Detail

- Unit Summary

- Unit Detail

- Clear Profile Table

- Stop

START

Rapid SQL PL/SQL Profiler lets you begin a new profiling session or open a previous profiling session with the Start command.

Starting a New Profile Session

To start a new Profiler session, do the following:

1   On the *Tools* menu, click *PL/SQL Profiler* and then click *Start.*

OR

On the *PL/SQL Profiler* toolbar, click Execute.

OR

On the *Explorer* Tab, right-click the *PL/SQL Code Profiling* node, and then click *Start.*

Rapid SQL opens the PL/SQL Profiler - Start dialog box.

2   In the *Profile Label* box, type the name of the new profile.

   NOTE: Each user can own one or more Profiles.

3   Click *OK* to begin profiling.

Starting an Existing Profile Session

To start an existing Profiler session, do the following:

1   On the Explorer Tab, expand the PL/SQL Code Profiling node.

    Rapid SQL displays the list of existing Profiles.

2   On the *Tools* menu, click PL/SQL Profiler, and then click Start.

    OR

    On the *PL/SQL Profiler* toolbar, click Execute.

    OR

    On the Explorer Tab, right-click the PL/SQL Profiler node, and then click Start.

    OR

    On the Explorer Tab, right-click the target *Profile,* and then click *Start*.

    Rapid SQL opens the PL/SQL Profiler - Start dialog box.

3   Click the *Profile Label* list and then click the existing profile.

4   Click *OK* to begin profiling.

FLUSH

Rapid SQL PL/SQL Profiler lets you move the data from the dynamic tables into Analysis tables with the Flush command.

Flushing a Profile

To flush a Profile, do the following:

1   On the *Tools* menu, click *PL/SQL Profiler* and then click *Flush.*

    OR

    On the *PL/SQL Profiler* toolbar click Flush.

    OR

    On the *Explorer Tab*, right-click the *PL/SQL Code Profiling* node, and then click *Flush.*

    Rapid SQL opens the PL/SQL Profiler - Flush dialog box.

2   In the *PL/SQL Profiler - Flush* dialog box:

    • Click *Flush* to delete the data in a running profile.

    • Click *Flush & Analyze* to open the [PL/SQL Profiler Run Detail](#) window.

    • Click *Cancel* to abort the flush and continue the profiling session.

    NOTE: You can only Flush a running Profile.

RUN SUMMARY

The Rapid SQL PL/SQL Profiler Run Summary window lets you view the following information for each of your profiles:

- Run ID
- Run Date
- Total Time

Opening the Run Summary Window

To open the Run Summary Window, do the following:

1   On the Explorer Tab, expand the PL/SQL Code Profiling node.

     Rapid SQL displays the list of existing Profiles.

2   On the *Tools* menu, click PL/SQL Profiler, and then click Run Summary.

On the *PL/SQL Profiler* toolbar click Run Summary.

On the Explorer Tab, right-click the PL/SQL Profiler node, and then click Run Summary.

On the Explorer Tab, right-click the target *Profile,* and then click *Run Summary.*

3

     Rapid SQL opens the PL/SQL Profiler - Run Summary window.

4   Click the *Profile Label* list, and then click the target profile.

RUN DETAIL

The Rapid SQL PL/SQL Profiler Run Detail window lets you view the following information for each of your profiles:

- Run Number
- Run Date
- Run Time

The Rapid SQL PL/SQL Profiler lets you view the information for all runs or you can view profile information based on the unit type or unit owner.

The Rapid SQL PL/SQL Profiler Run Detail window lets you view results in milliseconds, seconds and minutes. The Run Detail window also contains graphical displays of the profiling data that you can go to the specific unit within the summary portion of the window.

Opening the Run Detail Window

To open the Run Detail Window, do the following:

1   On the Explorer Tab, expand the PL/SQL Code Profiling node.

    Rapid SQL displays the list of existing Profiles.

2   On the *Tools* menu, click *PL/SQL Profiler* and then click *Run Detail*.

    OR

    On the *PL/SQL Profiler* toolbar click Run Detail.

    OR

    On the *Explorer* Tab, right-click the *PL/SQL Code Profiling* node and then click *Run Detail*.

    OR

    Right-click the target run and then click *Run Detail.*

    OR

    In a *PL/SQL Profiler -* <u>Run Summary</u> window right-click and then click *Detail.*

    OR

    In a *PL/SQL Profiler -* <u>Unit Summary</u> window right-click and then click *Detail.*

    Rapid SQL opens the PL/SQL Profiler - Run Detail window.

3   In the *PL/SQL Profiler - Run Detail* window:

    • Click the *Label* list box and then click the target profile.

    • Click the *Run* list and then click the target run.

    • Click the *Unit Type* list and then click the target unit type(s).

    • Click the *Unit Owner* list and then click the target unit owner(s) to populate the table.

UNIT SUMMARY

The Rapid SQL PL/SQL Profiler Unit Summary window lets you view the following information for each of your profiles:

• Run ID

• Run Date

• Run Time

• Unit Time

• Percentage of Run Time

The Rapid SQL PL/SQL Profiler Unit Summary window lets you view results in milliseconds, seconds and minutes. The Unit Summary window also displays graphs of execution statistics for the top N runs and associated units. You can use the graphical displays to go to the specific run within summary portion of the window.

Opening the Unit Summary Window

To open the Unit Summary Window, do the following:

1   On the *Tools* menu, click *PL/SQL Profiler,* and then click *Unit Summary.*

    OR

    On the *PL/SQL Profiler* toolbar click Unit Summary.

    OR

    On the *Explorer* Tab, right-click the *PL/SQL Code Profiling* node, and then click *Unit Summary.*

    Rapid SQL opens the PL/SQL Profiler - Unit Summary window.

2   In the *PL/SQL Profiler - Unit Summary* window:

    • Click the *Unit Owner* list and then click the target unit owner.

    • Click the *Unit Name* list and then click the target unit name to populate the table.

CLEAR PROFILE TABLE

Rapid SQL PL/SQL Profiler lets you delete data from the user's profile tables with the command Clear Profile Table.

Clearing a Profile Table

To clear a profile table, do the following:

1   On the *Tools* menu, click PL/SQL Profiler, and then click Clear Profile Table.

    OR

    On the *PL/SQL Profiler* toolbar, click Clear Profile Table.

    OR

    On the *Explorer* Tab, right-click the *PL/SQL Code Profiling* node, and then click *Clear Profile Table.*

    Rapid SQL clears the profile table.

2   In the dialog box, if you are sure that you want to clear out the profiler tables, click *Yes.*

UNIT DETAIL

The Rapid SQL PL/SQL Profiler Unit Detail window lets you view the following information for each of your profiles:

- Average Time
- Source
- PL/SQL Script

The Rapid SQL PL/SQL Profiler Unit Detail window lets you view results in milliseconds, seconds and minutes. The Unit Detail window also provides two calculation options for viewing unit execution time as a percentage of total execution time (total run vs unit run). The Rapid SQL PL/SQL Profiler Unit Detail window also displays graphs of execution statistics for the top N run. You can use the graphical displays to go to the specific line within source code portion of the window.The graphical display portion of the window contains options for viewing advanced statistics.

The [Advanced View](#) of the Rapid SQL PL/SQL Profiler Unit Detail window, lets you view the following information for each of your profiles:

- Hit Lines
- Missed Lines
- Line Number
- Calls
- Total Time
- Percentage of the Total Time
- Average Time
- Minimum Time
- Maximum Time

Opening the Unit Detail Window

To open the Unit Detail Window, do the following:

1 On the Explorer Tab, expand the PL/SQL Code Profiling node.

   Rapid SQL displays the list of existing Profiles.

2   On the *Tools* menu, click *PL/SQL Profiler* and then click *Unit Detail.*

   OR

   On the *PL/SQL Profiler* toolbar click Unit Detail.

   OR

   On the *Explorer* Tab, right-click the *PL/SQL Code Profiling* node and then click *Unit Detail.*

   OR

   Right-click the target run and then click *Unit Detail.*

   OR

   In a *PL/SQL Profiler -* [Run Summary] window right-click and then click Unit *Detail.*

   OR

   In a *PL/SQL Profiler -* [Unit Summary] window right-click and then click Unit *Detail.*

   Rapid SQL opens the PL/SQL Profiler - Unit Detail window.

3   In the *PL/SQL Profiler - Unit Detail* window, do any of the following:

   • Click the *Label* list and then click the target profile.

   • Click the *Run* list and then click the target run.

   • Click the *Unit* list and then click the target unit to populate the table.

   • Right-click and then click *Show Only Hit Lines* to populate the table with the *Average Time* and *Source* for hit lines.

   • Right-click and then click *Show Only Missed Lines* to populate the table with the *Average Time* and *Source* for missed lines.

Opening the Unit Detail Window Advanced View

To open the Unit Detail Window Advanced View, do the following:

1   In the Unit Detail table, right-click and then click *Advanced View* to populate the table with advanced view information.

STOP

Rapid SQL PL/SQL Profiler command Stop pauses the data gathering operation. Stop & Analyze populates the summary tables so you can view the Unit Detail and Run Summary windows

Stopping a Profiling Session

To stop a Profiling session, do the following:

1 On the *Tools* menu, click PL/SQL Profiler and then click Stop.

   OR

   On the *PL/SQL Profiler* toolbar click Stop.

   OR

   On the *Explorer* Tab, right-click the *PL/SQL Code Profiling* node, and then click *Stop*.

   Rapid SQL opens the PL/SQL Profiler - Stop dialog box.

2 In the *PL/SQL Profiler - Stop* dialog box:

   • Click *Stop* to stop the profiling session.

   • Click *Stop & Analyze* to open the [PL/SQL Profiler Run Detail](#) window.

   • Click *Cancel* to continue the profiling session.

USING RAPID SQL PL/SQL PROFILER

The steps below provide a high level overview of running a profiling session:

   • Starting the Session

   • Executing the Sample Script

   • Stopping and Analyzing the Session

For more information, see [Sample Profiling Session](#).

Using the Rapid SQL PL/SQL Profiler

   NOTE: The first execution of a PL/SQL unit can take more time to execute because the code is loading into memory; subsequent runs take less time.

To use the Profiler, do the following:

1 On the *Tools* menu, click *PL/SQL Profiler* and then click *Start*.

   OR

   On the *PL/SQL Profiler* toolbar, click Start.

   Rapid SQL opens the PL/SQL Profiler - Start dialog box.

2 In the *Profile Label* box, enter the name of the new profile.

   NOTE: Each user can own one or more Profiles.

3   Click *OK*.

Rapid SQL begins profiling.

4   In the *Explorer* Tab, *execute* one of the following PL/SQL database objects:

- Procedure

- Function

- Package Procedure

- Package Function

Profiler displays profiling data in the right pane of the application.

5   On the *Tools* menu, click *PL/SQL Profiler* and then click Stop.

Rapid SQL opens the PL/SQL Profiler - Stop dialog box.

6   In the *PL/SQL Profiler - Stop* dialog box, do any of the following:

- Click *Stop* to stop the profiling session.

- Click *Stop & Analyze* to open the PL/SQL Profiler Run Detail window.

- Click *Cancel* to continue the profiling session.

7   If you clicked *Stop & Analyze* do the following:

- Click the *Label* list and then click the target profile.

- Click the *Run* list and then click the target run.

- Click the *Unit Type* list and then click the target unit type(s).

- Click the *Unit Owner* list and then click the target unit owner(s) to populate the table.

8   Use the *Tools* menu to open any of the following PL/SQL Profiler windows:

- PL/SQL Profiler Run Summary

- PL/SQL Profiler Unit Summary

- PL/SQL Profiler Unit Detail

SAMPLE PROFILING SESSION

The Rapid SQL installation includes two scripts for the sample profiling session:

- PROFILER_BUILD_DEMO.SQL

- PROFILER_DEMO.SQL

The PROFILER_BUILD_DEMO.SQL creates the objects that you will profile in the walk-through, and the PROFILER_DEMO.SQL is what you will profile during the walk-through.

> **NOTE:**   To create the objects in the PROFILER_BUILD_DEMO.SQL script, you must have CREATE privileges.

The sample script demonstrates the following features of the Rapid SQL PL/SQL Profiler:

- Unit Detail

- Run Detail

- Show Only Hit Lines

- Advanced View

During the installation, Rapid SQL places the scripts in the configured, default user SQL scripts folder. For details, see [Directories Options](#).

Overview

Sample Profiling Session is divided into six parts:

- Getting Started

- Starting the Session

- Executing the Sample Script

- Stopping the Session

- Rerunning & Reexecuting the Session

- Stopping & Analyzing

SAMPLE PROFILING SESSION - GETTING STARTED

In this step of Sample Profiling Session, you create the objects that you will profile in the walk-through.

Overview

The Getting Started section guides you through:

- Opening PROFILER_BUILD_DEMO.SQL

- Changing the Explorer tab Display

- Confirming the Creation of the Package

Getting Started

1   Start Rapid SQL.

2   On the *File* menu, click *Open*. Rapid SQL opens the Open Files dialog box.

> **NOTE:**   The sample file used in this example, PROFILER_BUILD_DEMO.SQL, is stored in the configured user SQL scripts folder. For details, see [Directories Options](#).

3   In the *Open File(s)* dialog box, navigate to the user SQL scripts folder and open the PROFILER_BUILD_DEMO.SQL file.

Rapid SQL opens the PROFILER_BUILD_DEMO.SQL script in an SQL Editor window

4   On the *SQL Editor* window, click Execute.

Rapid SQL executes the script and creates the package.

5   On the *Explorer* Tab list, click *Organize by Owner.*

6   On the *Explorer* Tab, click the node of your owner name.

Rapid SQL displays your schema objects.

7   Double-click the *Packages* node to display *PF_COUNT_TIME_INTERVAL* and confirm its creation.

   NOTE: If you were not able to create the package, check the error messages to determine the problem.

SAMPLE PROFILING SESSION - STARTING THE SESSION
In this step of Sample Profiling Session, you start the profiling session.

Sample Profiling Session - Starting the Session

To start the session, do the following:

1   On the *File* Menu, click *Open* to open the *Open Files* dialog box.

2   In the *Open Files* dialog box, type the path to the UsrScrpt directory, press *ENTER* and then double-click *PROFILER_DEMO.SQL* to open the script in an SQL Editor.

Rapid SQL opens the PROFILER_DEMO.SQL script in an SQL Editor.

3   On the *Tools* menu, click *PL/SQL Profiler* and then click *Start.*

Rapid SQL opens the PL/SQL Profiler - Start dialog box.

4   In the *Profile Label* list, type *DemoProfile.*

5   Click *OK* to begin the profiling session.

Rapid SQL begins the profiling session.

If this is the first time you start the Rapid SQL PL/SQL Profiler, Rapid SQL displays a message that user profiling tables need to be installed.

6   Click Yes.

Rapid SQL opens SQL*Plus to create the tables. You will need to start the profiling session again (see Step 3).

SAMPLE PROFILING SESSION - EXECUTING THE SAMPLE SCRIPT
In this step of Sample Profiling Session, you execute the DEMO script.

Sample Profiling Session - Executing the Sample Script

To execute the sample script, do the following:

1   On the *SQL Editor* window toolbar, click Execute.

    Rapid SQL executes the script and opens a Results Tab.

SAMPLE PROFILING SESSION - STOPPING THE SESSION
In this step of Sample Profiling Session, you stop the profiling run.

Sample Profiling Session - Stopping the Session

To stop the session, do the following:

1   On the *Tools* menu, click *PL/SQL Profiler* and then click Stop.

    Rapid SQL opens the PL/SQL Profiler - Stop dialog box.

2   Click *Stop.*

SAMPLE PROFILING SESSION - RERUNNING & REEXECUTING THE SESSION
In this step of Sample Profiling Session, you run the same profile session and execute the DEMO
script again.

Sample Profiling Session - Rerunning & Reexecuting the Session

To rerun and reexecute the session, do the following:

1   In the *SQL Editor*, click the Query Tab.

2   On the *Tools* menu, click *PL/SQL Profiler* and then click *Start*.

    Rapid SQL opens the PL/SQL Profiler - Start dialog box again.

3   Click the down arrow on the *Profile Comment* list and then click *DemoProfile.*

4   Click *OK* to begin the profiling session.

5   On the *SQL Editor* toolbar, click Execute.

    Rapid SQL executes the script again and opens the Results Tab.

SAMPLE PROFILING SESSION - STOPPING & ANALYZING
In this step of Sample Profiling Session, you stop profiling and analyze the runs.

Sample Profiling Session - Stopping & Analyzing

To stop and analyze, do the following:

1   On the *Tools* menu, click *PL/SQL Profiler* and then click *Stop.*

Rapid SQL opens the PL/SQL Profiler - Stop dialog box again.

2   Click *Stop & Analyze.*

Rapid SQL opens the PL/SQL Profiler - Run Detail window.

3   Click the *Run* list and then click *Run#x.*

> NOTE: Rapid SQL assigns a number to each profiling session. These numbers increase incrementally each time you run a profiling session. x= the number that was assigned to your first run.

Rapid SQL PL/SQL Profiler populates the grid with information on the procedure, package body, and package specification.

> NOTE: For the purposes of this walk-though we have created this package under the account SCOTT.

4   Click the *Run* list again and then click the *Run#* for your second run.

Notice this time there is no information on the package specification. It was created in the first run.

5   Right-click and then click *Detail.*

Rapid SQL PL/SQL Profiler opens the PL/SQL Profiler - Unit Detail window and populates the grid with the average time to execute each unit and the source code. Notice the time to execute SELECT object_name, in the example is 126 ms.

6   In the PL/SQL Profiler - Unit Detail window, click the *Run* list and then click *Run#x* for your first run.

7   Click the *Unit* list and then click *user name.PF_COUNT_SYSTEM_OBJECTS.*

Notice the time to execute SELECT object_name is considerably greater: in the example it is 24476 ms.

8   Right-click and then click *Show Only Hit Lines.*

THE RAPID SQL PL/SQL PROFILER SHOWS ONLY THE LINES OF CODE THAT EXECUTED.

9   Right-click and then click <u>Advanced View</u>.

The Rapid SQL PL/SQL Profiler opens the Advanced View window.

10   Continue clicking the *Run* and *Unit* lists to compare the performance of each run and each session.

This concludes the Sample Profiling Session. You can now delete the objects created during the Sample Profiling Session. They are:

- Check Constraints, PLSQL_PROFILER_UNITS, PLSQL_PROFILER_DATA

- Foreign Keys, PLSQL_PROFILER_UNITS, PLSQL_PROFILER_DATA

- Package, PF_COUNT_TIME_INTERVAL

- Package functions, WEEKEND_DAYS_( ), WORKING_DAYS_( ), YEAR

- S_ELAPSED_BETWEEN_ ()

- PL/SQL code Profiles, DemoProfile

- Primary Keys, PLSQL_PROFILER_RUNS, PLSQL_PROFILER_UNITS, PLSQL_PROFILER_DATA

- Procedure, PF_COUNT_SYSTEM_OBJECTS

- Sequence, PLSQL_PROFILER_RUNNUMBER

- Tables, PLSQL_PROFILER_RUNS, PLSQL_PROFILER_UNITS, PLSQL_PROFILER_DATA

CODE ANALYST

The Code Analyst is a tool to identify time-consuming lines of code. Code Analyst lets you:

- Perform detailed response time analysis on the execution of [Procedures](#) and [External Functions](#).

- Benchmark the execution of one or more procedures or functions to determine exactly what code objects and lines of code are taking the longest to run.

- Save response time metrics and perform intelligent compares against current execution times so you can determine deviations from previous acceptable response times.

> **TIP:** You can set Code Analyst options in the [Code Analyst Options](#).

> **NOTE:** Availability of this feature depends on your Rapid SQL licensing. For more information, see [Licensing](#).

Important Notes

- For DB2, before profiling with Code Analyst, [Compile](#) all procedures with the debugging option selected.

- For Oracle, when using the Oracle Debugger, [Compile](#) all procedures with the debugging option selected before profiling with Code Analyst.

Common Tasks

[Creating a Code Analyst Session](#)

[Identifying and Fixing Bottlenecks Using Code Analyst](#)

[Comparing Code Analyst Sessions](#)

[Cloning a Code Analyst Session](#)

[Deleting a Code Analyst Session](#)

[Stopping a Code Analyst Session Execution](#)

[Executing a Code Analyst Session](#)

[Scheduling a Code Analyst Session](#)

[Unscheduling a Code Analyst Session](#)

[Refreshing a Code Analyst Session](#)

[Saving Results in Code Analyst](#)

[Printing Results in Code Analyst](#)

[Viewing Run Details in Code Analyst](#)

[Viewing Unit Summary Information in Code Analyst](#)

[Viewing Unit Details in Code Analyst](#)

[Setting View Options for the Unit Detail Tab in Code Analyst](#)

[Extracting SQL Text in Code Analyst](#)

[Executing SQL in Code Analyst](#)

CODE ANALYST DBMS NOTES

Code Analyst is available for:

- Microsoft SQL Server 7 or later

- Oracle 7 or later

- IBM DB2 LUW 8

- Sybase ASE 12.0.0.3 or later

Rapid SQL utilizes debugger technology to capture the data for each line of executed code. For Oracle, you can use the debugger or using Oracle's supplied DBMS_Profiler package.

> **TIP:** For Oracle, you can specify to use the debugger or the DBMS_Profiler package on the [Code Analyst Options](#).

The Code Analyst will step through each line of code, stopping to record data for those lines of code onto which a breakpoint can be issued. Some debuggers cannot capture time metrics for all lines of a stored procedure or function.

Procedures and functions that contain looping constructs will require more time to run. The additional amount of time needed to run is proportional to the number of iterations in the loop.

For more information, see:

[IBM DB2 LUW 8i Data Captured by Code Analyst](#)

[Microsoft SQL Server Data Captured by Code Analyst](#)

[Oracle Data Captured by Code Analyst](#)

[Sybase ASE Data Captured by Code Analyst](#)

IBM DB2 LUW 8I DATA CAPTURED BY CODE ANALYST

Code Analyst utilizes the IBM Debugger when capturing time data.

The debugger is verified to run on IBM DB2 LUW version 7.2 and up. There is a known issue running version 7.2 with Fixpack 9.

DB2 has documented limitations on lines of code can be profiled.

The following are SQL statements that are NOT valid break point lines:

BEGIN

BEGIN

BEGIN NOT ATOMIC

BEGIN ATOMIC

CLOSE CURSOR

DECLARE cursor WITH RETURN FOR <sql statement>

DECLARE , var without default

DECLARE CONDITION (CONDITION) FOR SQLSTATE (VALUE) "..."

DECLARE CONTINUE HANDLER

DECLARE CURSOR

DECLARE EXIT HANDLER

DECLARE RESULT_SET_LOCATOR [VARYING]

DECLARE SQLSTATE

DECLARE SQLCODE (unless there is a default)

DECLARE UNDO HANDLER (unless they are entered)

DO

ELSE

END

END CASE

END IF

END FOR

END REPEAT

END WHILE

ITERATE

LEAVE

LOOP

OPEN CURSOR

REPEAT (as a keyword alone)

RESIGNAL

SIGNAL

THEN

labels, e.g. P1:

NOTE: Code containing these statements will not have times associated with them.

For more information, see <u>Code Analyst DBMS Notes</u>.

MICROSOFT SQL SERVER DATA CAPTURED BY CODE ANALYST

In order to execute a Code Analyst session against a Microsoft SQL Server database, the SQL Server debugger must be installed and functioning properly. Please refer to <u>Embarcadero SQL Debugger for Microsoft SQL Server</u> for details concerning set up.

Related Information

<u>Code Analyst DBMS Notes</u>

ORACLE DATA CAPTURED BY CODE ANALYST

When using the PL/SQL Profiler, Oracle has documented an issue regarding extremely large times being returned by the profiler. The times are sometimes hundred times larger than the actual run time of the stored procedure or function. Oracle documents that this is a vendor/os problem rather than an Oracle problem, because the RDTSC instruction is reporting wrong time stamp counter. They indicate that they have seen this problem on some INTEL Pentium processors.

Related Information

<u>Code Analyst DBMS Notes</u>

SYBASE ASE DATA CAPTURED BY CODE ANALYST

Sybase has documented a problem with their debugger API. The problem involves reporting the wrong line number through the debugger. Because of this bug, Code Analyst may report back data for blank lines or lines that contain comments. Sybase has fixed this problem release 12.5.2 of the database. All procedures affected must be dropped and recreated in order to correct the problem.

Related Information

<u>Code Analyst DBMS Notes</u>

CODE ANALYST REQUIREMENTS

Debuggers

The Code Analyst uses fully configured Embarcadero SQL Debuggers to profile, and therefore availability of this feature depends on your Rapid SQL licensing. For more information, see the following topics:

- Licensing

- Rapid SQL Add-On Tools

Using Code Analyst with the Oracle Profiler

Oracle users have the option of either using the Oracle Debugger or the Oracle Profiler with Code Analyst to capture statistics. To use Code Analyst with the profiler option, Oracle's profiler package must be installed. The install is user specific, so it must be installed by each user wishing to use Code Analyst. To install the package, users can invoke the Rapid SQL PL/SQL Profiler.

> **TIP:** You can set profiler options in the Code Analyst Options. You can specify that Code Analyst display the actual run time on the database, and does not include the time it takes to get to the server.

Privileges

For Oracle, SYS privileges are required to install the Code Analyst tables. If you do not have SYS privileges, ask your Server Administrator to log into the Oracle datasource as SYSDBA, and then open Code Analyst to install the tables.

During install, the following privileges are set for the Code Analyst tables.

- DB2 – Permissions are granted to the Public Group

- Oracle – Permissions are granted to the Public group.

- Microsoft SQL Server – Permissions are granted to the Public role.

- Sybase – Permissions are granted to the Public group.

All users can use the Code Analyst but each user will only see their own run ids. Users need to belong to the public group.

> **TIP:** You can check this/modify privileges in the Users Editor.

INSTALLING CODE ANALYST

To install Code Analyst, do the following:

1 On the Tools menu, select *Code Analyst*.

2 In Select the database you would like to install the tables on, select a database.

3   For IBM DB2 LUW for Open Systems, in select the tablespace you would like to install the tables on, select a tablespace.

4   For IBM DB2 LUW for Open Systems, in select the schema you would like to install the tables on, select a schema. The default is EMBTCA schema.

5   In Select the filegroup you would like to install the tables to, select a filegroup.

Code Analyst installs the following repository tables in the repository:

- EMBT_CODE_ANA_RUNS - Holds all the code analyst sessions created by users.
- EMBT_CODE_ANA_UNITS - Holds all the objects to be run for all.
- EMBT_CODE_ANA_PARAMS - Contains all the parameters for the objects that were run.
- EMBT_CODE_ANA_DATA - Contains the run data and is used to populate all the charts and statistics.
- EMBT_CODE_ANA_VERSION - Contains the version number of code analyst.

Code Analyst opens to the Run Summary tab.

6   Create a session using the [Creating a Code Analyst Session](#).

UNINSTALLING CODE ANALYST

The Uninstall functionality lets you uninstall Code Analyst from the server.

> NOTE: To uninstall a repository table, you need create table and grant privileges. Generally, you need sysadmin privileges.

1   On the Tools menu, select *Code Analyst*.

2   Select a session or object, and then select Uninstall.

Code Analyst removes the repository tables in the repository:

- EMBT_CODE_ANA_RUNS - Holds all the code analyst sessions created by users.
- EMBT_CODE_ANA_UNITS - Holds all the objects to be run for all.
- EMBT_CODE_ANA_PARAMS - Contains all the parameters for the objects that were run.
- EMBT_CODE_ANA_DATA - Contains the run data and is used to populate all the charts and statistics.
- EMBT_CODE_ANA_VERSION - Contains the version number of code analyst.

CODE ANALYST PRODUCT DESIGN

Code Analyst performs detailed response time analysis. Code Analyst steps through each line of code and profiles those lines of code that the debugger or profiler can capture time metrics for.

> **NOTE:** Some debuggers do not capture time metrics for all lines of a procedure or function. For more information, see Code Analyst DBMS Notes.

After capturing the time metrics, Code Analyst displays the data in an easy-to-read format on the tabs.

The Code Analyst is comprised of the following tabs:

Tab	Option	Description
Run Summary	Session	Lets you select the run session(s).
	Session	Displays the name of the session as created by the user.
	Run ID	Displays the run ID for the run(s). This number is system generated.
	Run Date	Displays the time and date of the session.
	Total Profile Time (ms)	Displays the total time taken for the profiled code to execute. This time is limited to the lines of code that are profiled. Overhead is not included in this calculation.
	Total Analysis Time	Displays the total time taken for the session to complete, including all overhead time needed to analyze the procedure or function.
	Scheduler	Displays the scheduler used to schedule the session. This information displays until the scheduled job has been run.
Run Detail	Session	Lets you select the object execution session.
	Run	Lets you select the object execution.
	Unit Type	Lets you select the object type for the object execution.
	Unit Owner	Lets you select the object owner for the object execution.
	Unit Database	Lets you select the object database for the object execution.
	Time Unit	Lets you specify the time unit for the Unit Execution graph.
	Unit Owner	Displays the owner of the procedure or function.
	Unit Name	Displays the name of the procedure or function.
	Unit Type	Displays the types of captured objects, including Anonymous Block, Function, Package Body, and Procedure for Oracle databases. Also displays SQL Statement and Procedure for the other platforms.
	Unit Database	Displays the database on which the object is stored.
	Total Profiled Time	Displays the total time taken for the profiled code to execute.
	% of Profiled Time	Displays the percentage of the Total Profiled Time for the run that this unit accounts for.
Comparison	Base Run	Lets you select the earlier object execution.

Tab	Option	Description
	New Run	Lets you select the later object execution.
	Unit Owner	Displays the owner of the procedure or function.
	Unit Name	Displays the name of the procedure or function.
	Unit Type	Displays the types of captured objects, including Anonymous Block, Function, Package Body, and Procedure for Oracle databases. Also displays SQL Statement and Procedure for the other platforms.
	Unit Database	Displays the object database for the object execution.
	Time Diff	Displays the time difference in milliseconds between the base run and the new run.
	New Profiled Time	Displays the profiled time of the new run.
	Base Profiled Time	Displays the profiled time of the base run.
Unit Summary	Unit Owner	Lets you select the object owner for the session(s).
	Unit Name	Lets you select the object name for the session(s).
	Unit Database	Lets you select the object database for the session(s).
	Number of Top Runs	Lets you specify the number of top object executions to display in the Top 5 Runs graph and select the unit of time for the Unit Time graph.
	Session	Displays the name of the session as created by the user.
	Run ID	Displays the unique id for the Run. This number is system generated.
	Run Date	Displays the time and date of the session.
	Total Analysis Time	Displays the total time taken for the session to complete, including all overhead time needed to analyze the procedure or function.
	Total Profiled Time	Displays the total time taken for the profiled code to execute.
	Unit Profiled Time	Displays the unit time for the session(s).
	% of Profiled Time	Displays the percentage of the Total Profiled Time for the run that this unit accounts for.
	% of Run Time	Displays the percentage of object execution time for the session(s).
Unit Detail	Session	Lets you select the session.
	Run	Lets you select the object execution.
	Unit Name	Lets you select the object name for the session.
	Number of Top Lines	Lets you specify the number of top lines in the Top 5 Lines Execution Time graph and select the total time units.
	Percentage Calculation	Lets you specify total object execution time or object run time.
	Calls	Displays the number of times the line was executed.
	Total Time	Displays the total time the line was executed.
	% of Total Profiled	Displays the percentage of the Total Profiled Time that this line of code was responsible for.

Tab	Option	Description
	Avg Time	Displays the average profiled time for this line.
	Min Time	Displays the minimum recorded time for execution of this line.
	Max Time	Displays the maximum recorded time for execution of this line.
	Dependency	Displays the UNIT_NUMBER of the dependency object that was called by that line. Lets you right-click and quickly go to that UNIT_NUMBER to see its Unit detail information.
	Source	Displays the object's SQL source code.

Common Tasks

[Creating a Code Analyst Session](#)

[Identifying and Fixing Bottlenecks Using Code Analyst](#)

[Comparing Code Analyst Sessions](#)

[Cloning a Code Analyst Session](#)

[Deleting a Code Analyst Session](#)

[Stopping a Code Analyst Session Execution](#)

[Executing a Code Analyst Session](#)

[Scheduling a Code Analyst Session](#)

[Unscheduling a Code Analyst Session](#)

[Refreshing a Code Analyst Session](#)

[Saving Results in Code Analyst](#)

[Printing Results in Code Analyst](#)

[Viewing Run Details in Code Analyst](#)

[Viewing Unit Summary Information in Code Analyst](#)

[Viewing Unit Details in Code Analyst](#)

[Setting View Options for the Unit Detail Tab in Code Analyst](#)

[Extracting SQL Text in Code Analyst](#)

[Executing SQL in Code Analyst](#)

CODE ANALYST TUTORIAL

The following tutorial guides you through the process of using the Code Analyst.

Creating a Code Analyst Session

1   On the Tools menu, select Code Analyst.

    Initially, Code Analyst installs the repository tables. For more information, see <u>Installing Code Analyst</u>. Then Rapid SQL opens the Code Analyst to the Run Summary tab.

2   On the Code Analyst Tools toolbar, click the Create New Collection button.

    Rapid SQL opens the first panel of the Code Analyst Wizard.

3   Select the individual object or group of objects to analyze. In this example, an individual stored procedure (CREATE_ADMISSION2) is selected.

> **TIP:**   Code Analyst does not let you select objects that do not have stored procedures.

4   Click Next.

    If the object(s) selected to be analyzed requires parameters, the second panel of the wizard prompts you to enter the parameters.

5   Double-click the object to set the parameters.

6   For IBM DB2 LUW for Open Systems and Oracle, the Compile button opens the Confirm Compile dialog box that lets you compile the objects to ensure that the Code Analyst can capture the time metrics.

7   Click Finish.

    Code Analyst displays a message that the Code Analyst will run longer than the actual code. Then Code Analyst analyzes the objects, using the Embarcadero SQL Debugger to profile and then opens the Run Detail tab.

> **TIP:**   You can select the "Please do not show me this dialog again" option in the dialog box or set the option on the <u>Code Analyst Options</u>.

Identifying and Fixing Bottlenecks Using Code Analyst

The Run Detail tab displays the total time for the objects being analyzed. The tab information may be enough to identify the potential bottleneck.

1   To view more detailed information, double-click the Unit Name.

Code Analyst opens to the Unit Detail tab.

The Unit Detail Tab displays the object code and other information related to the individual lines of code. You can determine which line of code is taking too long and why. The Unit Detail Tab is where you troubleshoot, and then resolve the problem in the Modifying objects using editors.

2   In Rapid SQL, open the object editor, and then modify the code on the Definition tab.

3   Click Alter.

4   In Code Analyst, on the Unit Detail Tab, click Execute.

Comparing Code Analyst Sessions

1 Click the Comparison tab.

Code Analyst has a Comparison facility to allow quick compares of two object executions, showing the base time and the new time, as well as the time differences. The Comparison tab lets you compare which of the two procedures or functions ran faster.

2 Examine the Time Diff which indicates improvement to code.

3 If necessary, continue to modify the code on the Definition tab of the object editor, and then press Alter.

4 In Code Analyst, on the Unit Detail Tab, click Execute.

5 Examine the Time Diff until the bottleneck is solved.

Common Tasks

Creating a Code Analyst Session

Identifying and Fixing Bottlenecks Using Code Analyst

Comparing Code Analyst Sessions

Cloning a Code Analyst Session

USING THE CODE ANALYST

When working with database code stored in database objects, it is sometimes difficult to pinpoint bottlenecks within the code. When situations like this arise, Code Analyst can assist in identifying the trouble spots. Code Analyst can be used to analyze one object or a group of objects. You select one or multiple objects, execute them, view the results, and save those results for later viewing or comparing.

Common Tasks

Refreshing a Code Analyst Session

Saving Results in Code Analyst

Printing Results in Code Analyst

Viewing Run Details in Code Analyst

Viewing Unit Summary Information in Code Analyst

Viewing Unit Details in Code Analyst

Setting View Options for the Unit Detail Tab in Code Analyst

Extracting SQL Text in Code Analyst

Executing SQL in Code Analyst

CREATING A CODE ANALYST SESSION

The Code Analyst Wizard creates a new Code Analysis session that creates data for the Code Analyst tabs:

1   On the Tools menu, select Code Analyst.

2   On the Code Analyst Tools toolbar, click Create New Collection.

    Rapid SQL opens the first panel of the Code Analyst Wizard.

3   Select the individual object or group of objects to analyze.

4   Click Next.

    If the object(s) selected to be analyzed requires parameters, the second panel of the wizard prompts you to enter the parameters.

5   Double-click the object to set the parameters.

6   For IBM DB2 LUW for Open Systems and Oracle, the Compile button opens the Confirm Compile dialog box that lets you compile the objects to ensure that the Code Analyst can capture the time metrics.

7   Click Finish.

    Code Analyst displays a message that the Code Analyst will run longer than the actual code. Then Code Analyst analyzes the objects, using the Embarcadero SQL Debugger to profile and then opens the Run Detail tab.

> **TIP:** You can also select the "Please do not show me this dialog again" option in the dialog box or set the option on the Code Analyst Options.

For more information, see Code Analyst Wizard.

CODE ANALYST WIZARD

The table below describes the options and functionality of the Code Analyst Wizard:

Option	Description
Session Name	Lets you type a name.
Select by Owner	Code Analyst Wizard queries the database to get the list of procedures and functions and lets you select objects to retrieve Codes for. Select to display available objects by owner, and then select the database(s).
Select by Object	Code Analyst Wizard queries the database to get the list of procedures and functions and lets you select objects to retrieve Codes for. Select to display available objects by object, and then select the object(s).
Object Name	Double-click each object to specify the input parameters. Specify which object executes first by clicking the Up and Down buttons.
Compile	IBM DB2 LUW FOR OPEN SYSTEMS AND ORACLE ONLY: Opens the Confirm Compile dialog box that lets you compile the objects to ensure that the Code Analyst can capture the time metrics.
Schedule	Opens the Select Scheduler dialog box or opens scheduling application.
Finish	Code Analyst analyzes the code.

For more information, see: [Creating a Code Analyst Session](#).

IDENTIFYING AND FIXING BOTTLENECKS USING CODE ANALYST

The Unit Detail Tab displays the object code and other information related to the individual lines of code. You can identify time-consuming lines of code in the Unit Detail Tab. The Unit Detail Tab is where you troubleshoot, and then resolve the problem in the [Modifying objects using editors](#).

1   On the Tools menu, select Code Analyst.

2   Click the Unit Detail Tab.

Percent of Run Time displays the percentage of object execution time for the session(s).

3   Identify an object that contains time-consuming code.

4   In Rapid SQL, open the object editor, and then modify the code on the Definition tab.

5   Click Alter.

6   In Code Analyst, on the Unit Detail Tab, click Execute.

7   Click the Comparison tab.

The Comparison Tab lets you compare times of the objects in two different object executions to determine which run was more efficient. The Comparison Tab displays the base time and the new time, as well as the time differences. The Comparison tab lets you compare which of the two procedures or functions ran faster.

8   Examine the Time Diff which indicates improvement to code.

9   If necessary, continue to modify the code on the Definition tab of the object editor, and then press Alter.

10   Create new Code Analyst sessions and examine the Time Diff until the bottleneck is solved.

For more information, see:

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

COMPARING CODE ANALYST SESSIONS

The Comparison Tab lets you compare times of the objects in two different object executions to determine which run was more efficient. The Comparison Tab displays the base time and the new time, as well as the time differences. The Comparison tab lets you compare which of the two procedures or functions ran faster.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, right-click the sessions, and then select Compare.

3   Examine the Time Diff which indicates improvement to code.

For more information, see:

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

CLONING A CODE ANALYST SESSION

The Clone Collection functionality lets you clone an existing Code Analyst session using the Code Analyst Wizard. Clone lets you reset the parameters or the order of the objects in the session without creating a new session.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to clone.

3   On the Code Analyst Tools toolbar, click Clone Collection.

    Rapid SQL opens the [first panel of the Code Analyst Wizard](#).

For more information, see:

Code Analyst Product Design

Using the Code Analyst

DELETING A CODE ANALYST SESSION

The Delete Collection functionality lets you delete the selected Code Analyst session.

1 On the Tools menu, select Code Analyst.

2 On the Run Summary tab, select the session to delete.

3 On the Code Analyst Tools toolbar, click Delete Collection.

   Code Analyst deletes the session.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

STOPPING A CODE ANALYST SESSION EXECUTION

The Stop Execution kills the execution of the selected collection.

1 On the Tools menu, select Code Analyst.

2 On the Run Summary tab, select the session to kill.

3 On the Code Analyst Tools toolbar, click Stop Execution.

   Code Analyst kills the execution.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

EXECUTING A CODE ANALYST SESSION

The Execute Collection functionality extracts the SQL text and then executes the code.

1 On the Tools menu, select Code Analyst.

2 On the Run Summary tab, select the session to execute.

3 On the Code Analyst Tools toolbar, click Execute Collection.

   Code analyst extracts and executes the SQL.

For more information, see:

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

SCHEDULING A CODE ANALYST SESSION

The Schedule Session functionality lets you schedule the session for a future run.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to schedule.

3   On the Code Analyst Tools toolbar, click Schedule Session.

    Code Analyst opens the default scheduler.

For more information, see:

[Scheduling](#)

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

UNSCHEDULING A CODE ANALYST SESSION

The Delete Session functionality lets you remove the session from a schedule.

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, select the session to unschedule.

3   On the Code Analyst Tools toolbar, click Delete Session.

For more information, see:

[Scheduling](#)

[Code Analyst Product Design](#)

[Using the Code Analyst](#)

REFRESHING A CODE ANALYST SESSION

The Refresh Data functionality refreshes the data.

1   On the Tools menu, select Code Analyst.

2   On the tab, on the Code Analyst Tools toolbar, click Refresh Data.

For more information, see:

[Code Analyst Product Design](#)

SAVING RESULTS IN CODE ANALYST

The Save functionality lets you save results for later viewing or comparing.

1   On the Tools menu, select Code Analyst.

2   On the tab, right-click the session or unit, and then select Save.

Code Analyst opens the Save Results dialog box.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

PRINTING RESULTS IN CODE ANALYST

The Print functionality lets you print results for later viewing or comparing.

1   On the Tools menu, select Code Analyst.

2   On the tab, right-click the session or unit, and then select Print.

Code Analyst opens the Print Results dialog box.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

VIEWING RUN DETAILS IN CODE ANALYST

The Run Detail tab displays the total time for the objects being analyzed. The tab information may be enough to identify the potential bottleneck.

To open the Run Details Tab in Code Analyst, do the following:

1   On the Tools menu, select Code Analyst.

2   On the Run Summary tab, right-click the session, and then select Run Detail.

OR

3   On the Unit Summary tab, right-click the session, and then select Run Detail.

For more information, see:

Code Analyst Product Design

VIEWING UNIT SUMMARY INFORMATION IN CODE ANALYST

The Unit Summary Tab in Code Analyst displays the individual runs for a session.

To open the Unit Summary Tab in Code Analyst, do the following:

1   On the Tools menu, select Code Analyst.

2   On the Comparison tab, right-click the session, and then select Unit Summary.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

VIEWING UNIT DETAILS IN CODE ANALYST

The Unit Detail Tab displays the object code and other information related to the individual lines of code. You can identify time-consuming lines of code in the Unit Detail Tab. The Unit Detail Tab is where you troubleshoot, and then resolve the problem in the Modifying objects using editors.

To open the Unit Details tab in Code Analyst, do the following:

1   On the Tools menu, select Code Analyst.

2   On the Unit Summary tab, right-click the session, and then select Unit Detail.

For more information, see:

Code Analyst Product Design

Using the Code Analyst

SETTING VIEW OPTIONS FOR THE UNIT DETAIL TAB IN CODE ANALYST

The table below describes the options on the shortcut menu for the Unit Details Tab in Code Analyst:

Option	Description
Dependency Details	Displays dependency details.
Show Only Hit Lines	Displays only those lines with time metrics.
Show Only Missed Lines	Displays only those lines without time metrics.
Show All Lines	Resets the view to show all lines.

Option	Description
Advanced View	Displays the default view.
Normal View	Displays a limited number of data columns.

EXTRACTING SQL TEXT IN CODE ANALYST

The Extract SQL Text functionality extracts SQL text to an ISQL window.

1   On the Tools menu, select Code Analyst.

2   On the Unit Detail tab, right-click the session, and then select Extract SQL Text.

    Code analyst extracts the SQL text to an ISQL window.

For more information, see:

Extract

Code Analyst Product Design

Using the Code Analyst

EXECUTING SQL IN CODE ANALYST

The Execute SQL functionality extracts SQL text to an ISQL window, and then executes the code.

1   On the Tools menu, select Code Analyst.

2   On the Unit Detail tab, right-click the session, and then select Execute SQL.

    Code analyst extracts the SQL text to an ISQL window and executes the code.

For more information, see:

Execute

Code Analyst Product Design

Using the Code Analyst

Index

deallocating unused space 548

Symbols

.mdf/.ndf/.ldf files 437
#define 627
#include 627
    default file location 100

Numerics

3D Rounded Docking Tabs. 49

A

abort tran on log full 490
ACCENT_SENSITIVITY 281
account, user 486
accounting, logins 495
accounts/accounting
    setting lock time 473
activating roles 143
active
    transactions, rollback segments 475
Add Cluster Column 314
Add Database Object Files(s) to Project Wizard
    Panel 1 715
    Panel 2 716
    Panel 3 716
    Panel 4 for All Objects 717
    Panel 4 for Selected Objects 717
    Panel 5 718
    Panel 6 718
add/modify login trigger object action 514
Additional resources 11
AES... algortihms 343
After 94
AFTER trigger timing 450
aggregate, MySQL functions 306
alarms/thresholds
    code analyst 93
algorithms
    encryption, Oracle columns 343
aliases
    cration shortcut 536
    creating, DB2 LUW 206
    creating, Sybase 359
    editor, DB@ LUW 390
    editor, DB2 Z/OS 408
    editor, Sybase 489
    see also objects, database
    support for 148
Alignment index partition setting 465, 471, 485, 494
ALL views 102
allocate extent object action 515
Allow Bit Data setting 406
Allow Duplicate Rows index setting 494

allow nulls
    domains 258
    Oracle columns 343
Allow Nulls setting
    tables 418
allow nulls setting
    by default, databases 490
    by default, databsaes 360
    user datatype 502
allow parallel option 400
Allow Parallel setting 211, 392
Allow Reverse Scans index setting 394
ALLPAGES 499
ALTER SYSTEM SET "_recyclebin"= 558
ALTER_REBUILD 522
alter, rename style 102
Analyze Code 631
analyze tables object action 517
ANSI null default 276, 438
ANSI nulls 276, 438
ANSI padding 276, 438
ANSI warnings 276, 438
Append Data setting 401
Append, materialized query tables 395
applications
    bind to database 490
arithabort 438
array dimension, domains 258
AS DEFAULT 281
as locator 393
As Locator function setting 411
As Locator option 401
Asc Sort, indexes 464
Ascending option
    indexes, InterBase 263
asymmetric keys
    SQL Server logins 286
AT(COMMIT) clause 613
attach database object action 518
Audit table setting 418
authorization
    owner, full-text catalogs 281
auto close 276, 438
Auto Commit Status 632
auto create statistics 276, 438
auto format options, ISQL 109
auto format, DDL extract options 97
auto identity 360, 490
Auto Replace feature
    enabling/configuring 816
auto shrink 276, 438
auto update statistics 276, 438
auto-increment, MySQL tables 308
auto-reload
    files, ISQL Editor 108