Product Documentation

Embarcadero® Rapid SQL™

User Guide

Version 2016 (16.0.1)

February, 2016

CONTENTS

Welcome to Rapid SQL

Welcome to Rapid SQL, the database administration and development solution that lets you manage IBM DB2 for Linux, Unix, and Windows, IBM DB2 for z/OS and OS/390, Interbase/Firebird, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, Sybase IQ, and Sybase ASE databases.

The list below describes the major sections of this document.

- 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, instructions on configuring Rapid SQL features, and additional information on using this document and third-party documentation.
- Datasource and Server Management - Shows you how to register the datasources that Rapid SQL will work against.This chapter also describes the extensive datasource management facilities.
- Team Server 2016 Support - Shows you how to make use of datasource definitions and ER/Studio model data in a Team Server 2016 Repository.
- 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.
- Coding Environments/Editors - 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 data editor, and an auto-replace tool for the scripting environment.
- Project Management - Rapid SQL provides project support and version control system integration.
- Add-on tools - License-dependent add-ons include:
- **Debuggers** - Rapid SQL provides debugging environments for DB2 LUW, SQL Server, Oracle, and Sybase ASE.
- **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.

Note: For information on access to third party documentation and descriptions of conventions used in this document, see More Information on Using this Document.

Additional Product Information

The Embarcadero Web site provides access to information and resources useful through the life of your product. This includes access to:

- Technical Support
- Case studies, video presentations, webinars and white papers
- Product communities through the Embarcadero Developer Network
- Free trials of related products

See www.embarcadero.com/support.

You can find general information about system requirements, licensing information, and details on DBMS product support in Prerequisites and Preliminary Tasks.

Prerequisites and Preliminary Tasks

You can find general information about system requirements, licensing information, and details on DBMS product support in:

- Introducing Rapid SQL
- Technical Requirements
- DBMS Support and Connectivity Options
- Installing and Licensing Rapid SQL
- Quick Start Setup
- More Information on Using this Document

For more information, you can also see the **Read Me** and **Quick Start** at http://docs.embarcadero.com/products/rapid_sql/.

Introducing Rapid SQL

Rapid SQL is an integrated development environment that enables developers to create, edit, version, tune, and deploy server-side objects residing on Microsoft SQL Server, Oracle, Sybase Adaptive Server, InterBase/Firebird, IBM DB2 for Linux, Unix, and Windows, and IBM DB2 for z/OS databases. Its unified database development environment provides extensive graphical facilities that simplify SQL scripting, object management, reverse engineering, database project management, version control, and schema deployment. With Rapid SQL, programmers can develop and maintain high-quality, high-performance client/server and web-based applications in less time, and with greater accuracy.

About Rapid SQL

Your new application provides tools that can be used by a number of functions within an organization using DBMS from multiple vendors, in testing, development, or production environments.

Datasource management tools: Datasources must be registered. Datasources can be registered manually or semi-automatically. Storage can be registry-based or file-based, and a network storage option facilitates shared use of datasource catalogs.

Object management tools: Your application supports a wide range of database objects and related elements for each DBMS. You can create new objects, edit existing objects, and use a range of object operations that support common, general or DBMS-specific actions.

Coding Environments/Editors: Fully-featured environments dedicated to SQL, DDL and Active Script development are provided. Execution options are available as appropriate, and environment-specific related features, such as rollback/commit and query plan options, are provided. Related coding aids include on-the-fly semantic and syntactic validation, text substitution shortcuts, and visual query building tools. Related execution tools include script and file execution facilities.

Project Management, Version Control, and Script Library: 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.

Other Time-savers and Productivity Tools: Working within the Rapid SQL environment, you have access to database and file search facilities, use visual difference to compare files or database objects, and access the Microsoft Windows task scheduler.

License-specific Add-ons: Depending on the licenses you purchased, you have access to the following tools:

- The Rapid SQL Code Analyst helps you identify time-consuming lines of code. It lets you perform detailed response time analysis, benchmark the execution of one or more procedures or functions, save response time metrics, and perform intelligent comparisons against current execution times.

- Debuggers, available for IBM DB2 for Linux, Unix, and Windows, Oracle, Sybase, and SQL Server, let you test functions and procedures. A Profiler is available for Oracle datasources.

Technical Requirements

Before using Rapid SQL, please verify that your environment meets the requirements listed below:

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

Browser Requirements

Rapid SQL requires Microsoft Internet Explorer 11 or later.

Hardware Requirements

Embarcadero Technologies recommends the following minimum hardware requirements:

- Pentium 4-level processor
- 1 GB of memory
- 1 GB of disk space
- 1024 x 768 display

Operating System Requirements

Rapid SQL supports the following operating systems:

- Windows XP SP" (32-bit and 64-bit)
- Windows Vista (32-bit and 64-bit)
- Windows 7 (32-bit and 64-bit)
- Windows 8 (32-bit and 64-bit)

XP Support Notes

Windows XP has two user security groups: Users and 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 to install and use Embarcadero Technologies applications. Because Restricted Users are members of the Users Group, they cannot install and run Embarcadero Technologies applications.

Use the **Group Membership** tab to determine your group and review the Microsoft security guidelines. On the **Control Panel**, open **User Accounts**. On the **Users** tab, select a user and then click the **Properties** button. Click the **Group Membership** tab.

Vista and Windows 7 Support Notes

Windows Vista UAC and Windows 7 provide two user types: Standard users and Administrators. 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. For the purpose of running Rapid SQL, default standard user token privileges should not be modified. Modifying standard user token privileges can result in licensing issues that will prevent Rapid SQL from operating correctly.

32-bit versus 64-bit Application Considerations and Restrictions

If you install the 64-bit version of Rapid SQL, and you are using custom drivers, you must be using 64-bit versions of those drivers when using the 64-bit version of Rapid SQL. Similarly, 32-bit versions of custom drivers must be used with the 32-bit Rapid SQL installation.

For version control integration, if you install the 64-bit version of Rapid SQL, you can work with either a 32-bit or 64-bit MSSCCI provider. The feature is controlled from the Options Editor's **Version Control** tab (File > Options > General > Version Control).

DBMS Support and Connectivity Options

Dedicated Support Connectivity Options

Rapid SQL provides dedicated connectivity to a specific version range of IBM DB2 for Linux, Unix, and Windows, IBM DB2 for z/OS,
InterBase/Firebird,
Microsoft SQL Server, MySQL, PostgreSQL, Oracle, Sybase, and Teradata databases. The following connectivity options are provided:

- Native Embarcadero drivers: Rapid SQL is packaged with a set of native drivers, each requiring a DBMS-specific client to be installed.
- InterBase/Firebird: InterBase requires both client and driver software to be installed. Firebird requires driver software.
- JDBC drivers: Rapid SQL can connect to a datasource more directly using one of the packaged, third-party JDBC drivers. No additional connectivity components need to be installed. One or more third-party drivers, tested against Rapid SQL, are installed with Rapid SQL.

The following table provides a summary of resources/requirements for connectivity to dedicated DBMS platforms. For each platform, it lists supported versions, the client software that must be installed if using native Embarcadero clients, and the third-party, Type 4 JDBC drivers packaged with Rapid SQL.

DBMS Platform	Supported Versions	Client Required for use with native Embarcadero drivers	Packaged JDBC Driver	Source & License
Apache Hive (Technical Preview)*	0.13.1	Simba Hive ODBC Driver (v1.4.13.1013) Hortonworks Hive ODBC Driver (v1.4.08.1008) Cloudera ODBC Driver for Apache Hive (v2.05.10.1003)	Apache Software Foundation - Apache Hive JDBC v0.13.1	license
Firebird	Firebird 2.0	Firebird ODBC Driver	Jaybird JDBC Driver	LGPL, source & license
IBM DB2 for z/OS	v8, v9, and v10	DB2 UDB Client for Windows 8.0 or later	IBM Data Server Driver for JDBC	-
IBM DB2 for LUW	Versions 9.0 - 10.0	IBM DB2 LUW Client for Windows 8.0 or later	IBM Data Server Driver for JDBC	-
InterBase	InterBase 2007 InterBase 2009 InterBase XE3	InterBase ODBC Driver	InterBase JDBC Driver	-
Microsoft SQL Server	2005 (All editions including 2005 Express Edition for 32-bit x86 and 64-bit Itanium & x86-64) 2008 2012 2014	Microsoft SQL Server Client Library	jTDS Type 4 JDBC Driver for Microsoft SQL Server Microsoft SQL Server JDBC Driver	LGPL source, license
MySQL	4.x	MySQL Connector/ODBC Driver 5.2.x Driver - MySQL Connector/ODBC driver 3.51.x Driver	MySQL Connector/J JDBC Driver	GPL, source & license
Oracle	Oracle 9i, 10g, 11g, and 12c	Oracle SQL*Net Client	Oracle JDBC Thin Driver	-
PostgreSQL	9.3 minimum Specifically-supported PostgreSQL-based database products include Greenplum, Pivotal HAWQ, and BigSQL.	PostgreSQL ODBC Driver (latest version recommended)	PostgreSQL JDBC Driver	BSD, license

Sybase ASE	Sybase **15.7** - Sybase **16**	**Sybase Open Client**	**jTDS Type 4 JDBC Driver for Microsoft SQL Server** **Sybase jConnect JDBC Driver**	LGPL, source, license
Sybase IQ	**12.7, 15.4, 16**	**SQL Anywhere ODBC drivers for Sybase IQ 12.7** **Sybase IQ 32-bit ODBC drivers**	**Sybase jConnect JDBC Driver for Sybase IQ**	-
Teradata (Technical Preview)*	**13.10, 14.0, 14.10,** and **15.0**	**Teradata ODBC driver for Windows** (version 15 recommended)	**Teradata JDBC Driver**	-

*** About Technical Previews** - Technical previews are intended to introduce a new DBMS platform only. The available functionality is typically minimal. For example, command Line startup, standard datasource registration and connection features, limited Datasource Navigator tree functionality, a minimal set of object actions, as well as SQL Editor execution and related, common SQL Editor functionality such as Query Options, Paste SQL Syntax, and Paste SQL Statement. Access may not be covered under SE or trial licenses. Contact Oz.Basarir@embarcadero.com to obtain a separate license for a platform designated as a Technical Preview.

Embarcadero Team Server 2016 Support

Rapid SQL can make use of datasource definitions stored on an Embarcadero Team Server 2016.

Generic JDBC/ODBC Connectivity

Generic JDBC/ODBC connectivity to non-dedicated DBMS systems or non-database datasources is also provided. Rapid SQL can connect to a datasource using a customer-provided, third-party JDBC version 4.0 or ODBC version 3.0 driver. Minimal Rapid SQL functionality is provided, including a basic Explorer tree and SQL querying.

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 IBM DB2 for z/OS documentation for more information.

Specifically-tested JDBC/ODBC Connectivity Products: Apache Hive/Hadoop

Rapid SQL has been successfully tested against Apache Hive/Hadoop datasources using the Hortonworks ODBC driver. Similarly, Cloudera Impala datasources have been tested using the Cloudera JDBC driver. In both cases, SQL querying and a Datasource Navigator tree are available.

DBMS Versions no Longer Supported

As of this release, Rapid SQL is no longer being tested against Sybase ASE versions before 15.7.

Installing and Licensing Rapid SQL

Before you can register any Embarcadero Technologies application, you must meet the minimum technical requirements. If you meet all the minimum technical requirements, you can install Rapid SQL. After installation, you must license the application.

The following topics walk you through this process:

- Technical Requirements
- DBMS Support and Connectivity Options
- Installing Rapid SQL
- Licensing Your Product

Installing Rapid SQL

To install the application successfully, you must complete each panel of the installation wizard. 64-bit and 32-bit versions of Rapid SQL are available with the trial version, which can be downloaded from http://www.embarcadero.com/products.

The Rapid SQL installation wizard installs all corresponding files on your machine. The installer offers common options such as license agreement, file and folder selections, and shortcut options. An application-specific option lets you associate file suffixes of elements such as SQL scripts with Rapid SQL.

> **NOTE:** Before installing Rapid SQL from Embarcadero ToolCloud, see http://docs.embarcadero.com/products/xe/ for additional requirements or revised instructions.

The panels presented by the wizard vary depending on whether you are upgrading or performing an initial installation.

> **Caution:** If you have not registered a license when you start a Rapid SQL installation, you will be prompted to register a license. For more information, see Licensing Your Product.

After the installation is complete, we recommend that you reboot your machine before using Rapid SQL.

Licensing Your Product

Each Embarcadero client application requires one or more licenses to run. An Embarcadero product, such as Rapid SQL, has a baseline license that provides basic feature support for that product. Also, 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 Category 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 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. Also, 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** (Rapid SQL): 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 module and feature availability are as follows:

- **XE Pro** (Rapid SQL): 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:

- o Registering a Trial or Workstation License during Installation
- o Registering a Workstation License After Application Startup
- o Registering By Phone
- o Registering a Networked License

Registering a Trial or Workstation License during Installation

Shortly after initiating the 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, and 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, 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
- 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 through 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 through 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 performing 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 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 the 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 on startup and checked back in on shutdown.

You can also use a license in offline mode. When you explicitly check out a license for offline use, you can use the license 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 the license 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:

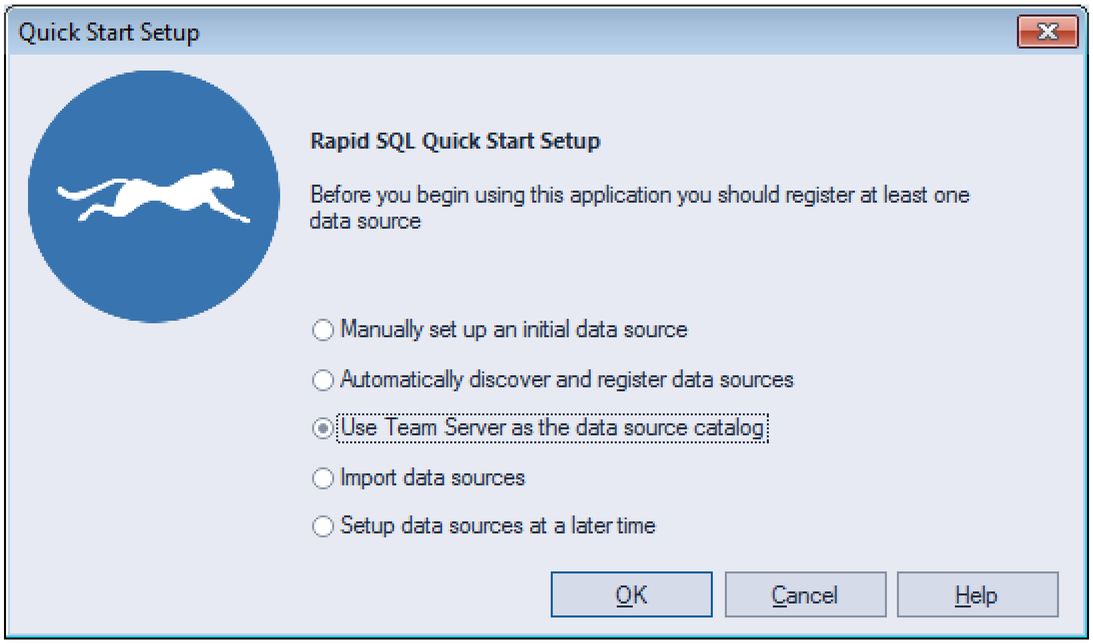
1. On the **Help** menu, select **Checkin License**.

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

Quick Start Setup

After Installing and Licensing Rapid SQL, the first time you launch the product, a dialog is displayed for registering data sources.

Datasource Registration


Quick Start Setup

Rapid SQL Quick Start Setup

Before you begin using this application you should register at least one data source

- ○ Manually set up an initial data source
- ○ Automatically discover and register data sources
- ● Use Team Server as the data source catalog
- ○ Import data sources
- ○ Setup data sources at a later time

This dialog includes the options described in the table below:

Option	Description
Manually set up an initial data source	Registering Cross-Platform Datasources to Rapid SQL
Automatically discover and register data sources	The Automatically Discovering Datasources features is launched.
Use Team Server as the data source catalog	Data sources in the configured Team Server will be registered in Rapid SQL.
Import data sources	The Import Data Sources wizard is launched.
Setup data sources at a later time	Rapid SQL opens without registered data sources.

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 supported DBMS platforms:

- 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
- PostgreSQL documentation
- Sybase ASE and Sybase IQ documentation

Shorthand for Third Party Product References

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

- DB2 LUW
- DB2 z/OS
- ITB/FBD *
- MySQL
- ORCL
- PSTGRS *
- SQL SVR
- SYB ASE
- SYB IQ

For specific DBMS version support for Rapid SQL, see the **Read Me** at http://docs.embarcadero.com/products/rapid_sql/.

DB2 LUW

DB2 LUWis used as shorthand for IBM DB2 for Linux, Unix, and Windows.

DB2 z/OS

DB2 z/OS is used as shorthand for IBM DB2 for z/OS.

ITB/FBD *

ITB/FBD is used as shorthand for InterBase/Firebird.

InterBase/Firebird is supported on Rapid SQL only

MySQL

MySQL is used as shorthand for the MySQL RDBMS.

ORCL

ORCL is used as shorthand for Oracle.

PSTGRS *

PSTGRS is used as shorthand for PostgreSQL.

Specifically-supported PostgreSQL-based database products include Greenplum, Pivotal HAWQ, and BigSQL.

SQL SVR

SQL SVR is used as shorthand for Microsoft SQL Server.

SYB ASE

SYB ASE is used as shorthand for Sybase Adaptive Server.

SYB IQ

SYB IQ is used as shorthand for Sybase IQ.

What's New in Rapid SQL 2016

PostgreSQL Object Management

Object Wizards/Editors

A Wizard and an Editor for **Types** have been added for PostgreSQL: Types Wizard and Types Editor.

Missing Indexes

The Missing Indexes feature analyze the system looking for **Missing Indexes** which, once added, would improve the performance.

Rapid SQL proposes you some **Indexes** that can be created created after analyzing the system. You can create these **Indexes** by launching the Indexes Wizard (SQL Server) and directly double-clicking the selected **Missing Index**.

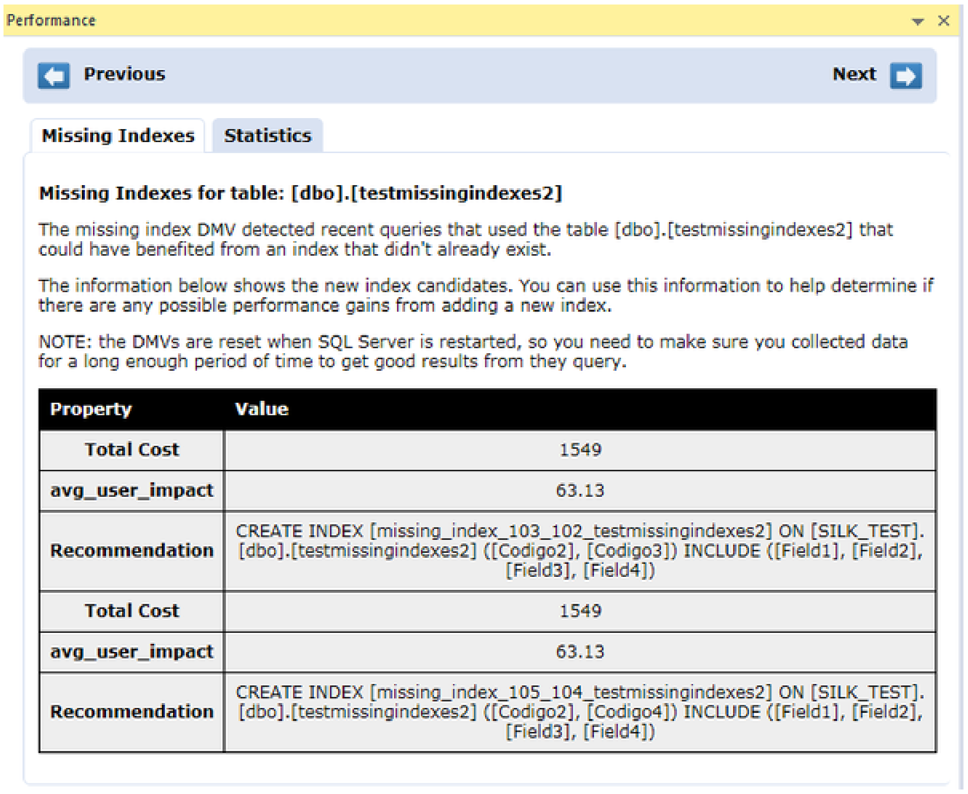
Description about how to use and access **Missing Indexes** is in Missing Indexes (SQL Server).

Note: Missing Indexes is only available for **SQL Server**.

ISQL Performance

SQL Editor includes new functionalities for SQL Server. In this new release, SQL editor provides information about the SQL statements while you are writing in the SQL Editor. This information is included in a new window as well as listed in the Error Pane. Information like **Missing Indexes**, **Statistics**, **Overlapping Statistics** and **Recommendations** are displayed in this new window. You can find more information in ISQL Performance

Performance window



Performance

Previous | Next

Missing Indexes | Statistics

Missing Indexes for table: [dbo].[testmissingindexes2]

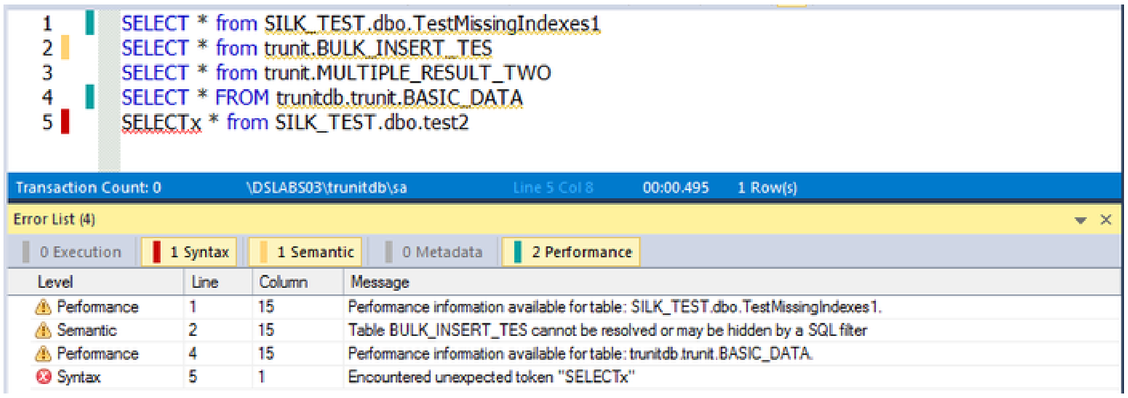
The missing index DMV detected recent queries that used the table [dbo].[testmissingindexes2] that could have benefited from an index that didn't already exist.

The information below shows the new index candidates. You can use this information to help determine if there are any possible performance gains from adding a new index.

NOTE: the DMVs are reset when SQL Server is restarted, so you need to make sure you collected data for a long enough period of time to get good results from they query.

Property	Value
Total Cost	1549
avg_user_impact	63.13
Recommendation	CREATE INDEX [missing_index_103_102_testmissingindexes2] ON [SILK_TEST].[dbo].[testmissingindexes2] ([Codigo2], [Codigo3]) INCLUDE ([Field1], [Field2], [Field3], [Field4])
Total Cost	1549
avg_user_impact	63.13
Recommendation	CREATE INDEX [missing_index_105_104_testmissingindexes2] ON [SILK_TEST].[dbo].[testmissingindexes2] ([Codigo2], [Codigo4]) INCLUDE ([Field1], [Field2], [Field3], [Field4])

ISQL Performance listed in Error Pane



```
SELECT * from SILK_TEST.dbo.TestMissingIndexes1
SELECT * from trunit.BULK_INSERT_TES
SELECT * from trunit.MULTIPLE_RESULT_TWO
SELECT * FROM trunitdb.trunit.BASIC_DATA
SELECTx * from SILK_TEST.dbo.test2
```

Transaction Count: 0 \DSLABS03\trunitdb\sa Line 5 Col 8 00:00.495 1 Row(s)

Error List (4)

0 Execution | 1 Syntax | 1 Semantic | 0 Metadata | 2 Performance

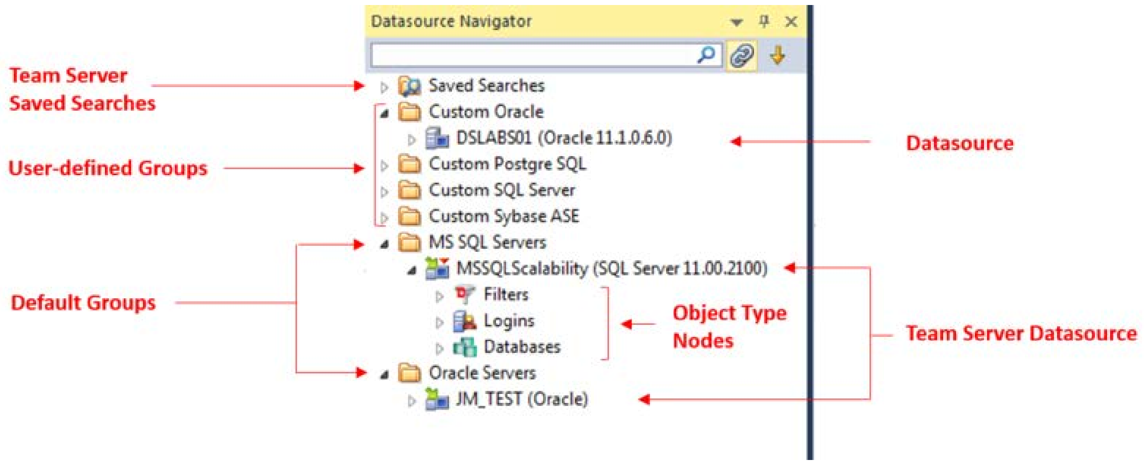
Level	Line	Column	Message
Performance	1	15	Performance information available for table: SILK_TEST.dbo.TestMissingIndexes1.
Semantic	2	15	Table BULK_INSERT_TES cannot be resolved or may be hidden by a SQL filter
Performance	4	15	Performance information available for table: trunitdb.trunit.BASIC_DATA.
Syntax	5	1	Encountered unexpected token "SELECTx"

Note:ISQL Performance only available for **SQL Server**.

Team Server 2016 Support

Integration of Team Server 2016 in Rapid SQL has been improved. The old **TeamServer Navigator** has been fully integrated into the **Datasource Navigator** as well as all the features only accessible from the TeamServer Navigator. For more information, see Using the Datasource Navigator and Basic Viewing Options in the Navigator. Notice that when logged in Team Server 2016, a node is added:

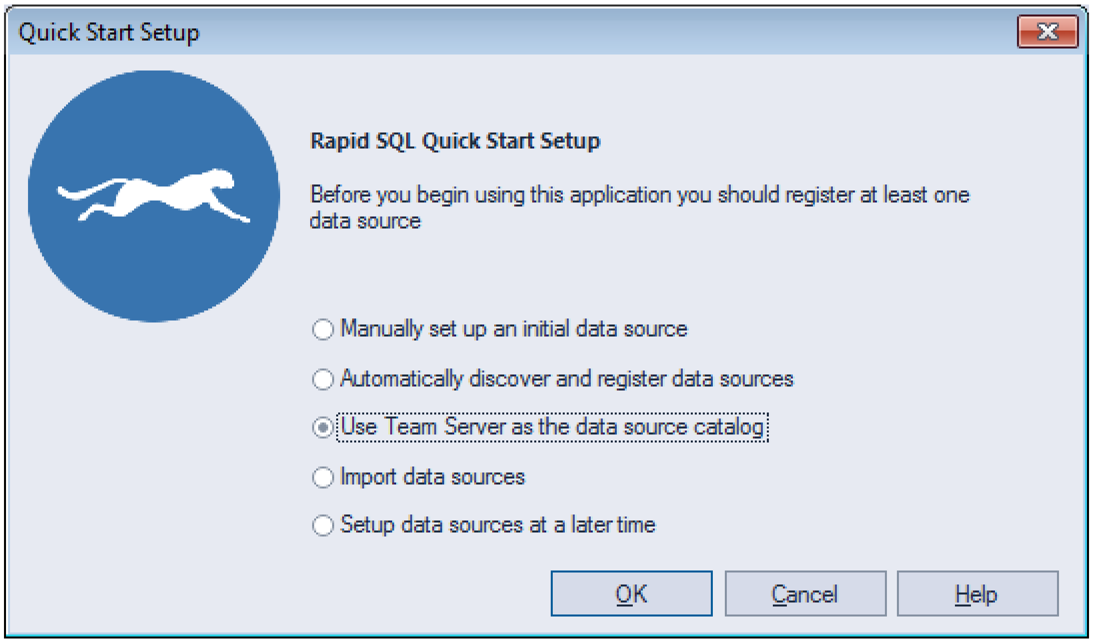
- o Saved Searches



Team Server 2016 as the Data Source Catalog

Rapid SQL 2016 adds the option of registering datasources from the Team Server 2016 Repository. To allow that, new options have been added to Quick Start Setup, and in Tools > Options > Datasources.

In Quick Start Setup, the import of datasources using Team Server 2016 as a catalog has been added.

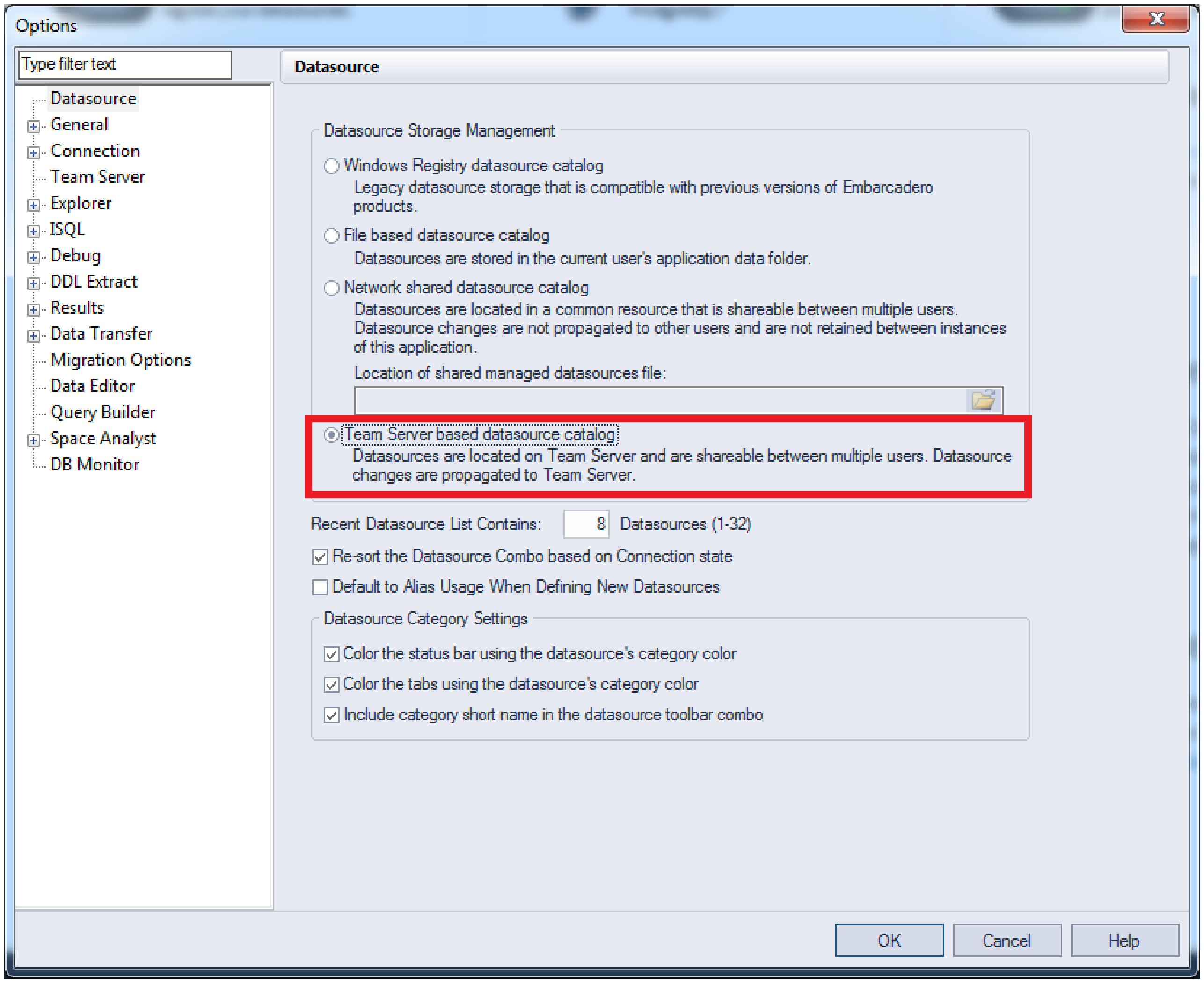

Quick Start Setup

Rapid SQL Quick Start Setup

Before you begin using this application you should register at least one data source

- ◯ Manually set up an initial data source
- ◯ Automatically discover and register data sources
- ◉ Use Team Server as the data source catalog
- ◯ Import data sources
- ◯ Setup data sources at a later time

In Tools > Options > Datasources the possibility of selecting Team Server 2016 as the source of the catalog has also been added. For more information, see Working with Team Server 2016 as Datasource Catalog.



Datasource

Datasource Storage Management

( ) Windows Registry datasource catalog
Legacy datasource storage that is compatible with previous versions of Embarcadero products.

( ) File based datasource catalog
Datasources are stored in the current user's application data folder.

( ) Network shared datasource catalog
Datasources are located in a common resource that is shareable between multiple users. Datasource changes are not propagated to other users and are not retained between instances of this application.
Location of shared managed datasources file:

(●) Team Server based datasource catalog
Datasources are located on Team Server and are shareable between multiple users. Datasource changes are propagated to Team Server.

Recent Datasource List Contains: 8 Datasources (1-32)

☑ Re-sort the Datasource Combo based on Connection state

☐ Default to Alias Usage When Defining New Datasources

Datasource Category Settings

☑ 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

Topics

- Release Notes for Rapid SQL 2016 Patch
- What Was New in Past Releases

Release Notes for Rapid SQL 2016 Patch

Important Advisory Notes

- If you want to run Rapid SQL on Microsoft Windows 8.1, verify that third-party DBMS native drivers install properly. Consider using a JDBC connection.
- Performance IQ has been removed.

Known Issues

- The current version of Rapid SQL does not support the MySQL ODBC client that shipped with previous versions of Rapid SQL. If Rapid SQL crashes after connecting to a MySQL data source:

1. Close Rapid SQL.
2. Uninstall the older MySQL ODBC client.
3. Restart Rapid SQL and try connecting to a MySQL datasource. You are prompted to install the MySQL client drivers.
4. Follow the installation instructions to install the new driver.

- Note that when debugging a procedure against an Oracle data source, 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 invalidated.
- For servers running on Sybase version 12.5.3a, the CREATE ENCRYPTION KEY and DECRYPT permissions do not display properly--they'll be shown as null. See Sybase's document CR392544 ("sp_helpprotect does not show privileges related to encryption keys."). This has been fixed in ASE version 12.5.3a/ESD#1 (now available as a download) and ASE version 12.5.4.
- When you are using the Debugger:
- The Debugger windows (Variables, Watch, CallStack, Dependency) cannot be undocked. This behavior is deliberate and is true only for the Debugger windows. The Output window can be undocked.
- When running Windows XP SP2, SQL Server 2000 SP4 is required on both the client and the server.
- When running the Debugger on SQL Server 2005 CLIENTS:

1. Verify that the file ssdebugps.dll is registered on the client machine. This file is REQUIRED for debugging and Microsoft only installs it on the server machine and not as a part of a client-only install.
2. Follow the instructions under the "Configure DCOM on the computer that is running Visual Studio .NET" heading from Microsoft to reconfigure the DCOM settings on your client machine: http://support.microsoft.com/default.aspx?kbid=833977.
3. On the Default Properties tab of the My Computer dialog box, make sure the Default Authentication Level is set to None and that the Default Impersonation Level is set to Impersonate.

- When using the Embarcadero Sybase debugger against a Sybase database with multiple temporary databases (supported in Sybase ASE 12.5.0.3 and above), either the user or the debugger application must be bound to a specific temporary database.
- To bind the Embarcadero Sybase Debugger:

sp_tempdb "bind" , "ap", "Sybase Debugger", "DB", "<tempdb name>"

- To bind the user:

sp_tempdb "bind", "lg", "<login id>", "DB", "<tempdb name>"

- Rapid SQL is currently unable to create table names, column names, and check constraint names longer than 18 characters on an OS390 mainframe database.
- On SQL Server 2005, the Browser component is disabled by default. With this service disabled, in Rapid SQL you cannot reference SQL Server instances by name, only by port number. If you want the ability to reference SQL Server instances by name, you must enable this service on the server.

Bug Fixes

Bugs Fixed in DBArtisan 2016 Patch (16.0.1)

Issue #	Description	DBMS
RAP-6050	[JDBC][Procedures] Smallint being treated as bit datatype	SQL Server
RAP-6048	[Data Source Navigator] Occassional crashes when expanding databases node from the datasource navigator	SQL Server
RAP-6027	[Crash] Sybase IQ customer crashes	Sybase IQ
DBA-33166	Case Number 00455434: Invalid column name 'systemwide password expiration'	Sybase ASE

DBA-33159	[Enhancement] Change fillfactor default value to prevent space consumption and performance issues on index rebuild	SQL Server
DBA-33154	[BCBSVT] [SQL Editor] Application crashes during SQL Editor session	DB2 for LUW
DBA-33152	DBA-33150 [Enhancement] Improve the exception handling in case of driver blowing up	Oracle
DBA-33151	DBA-33150 [ISQL] Nomura crashes	Oracle
DBA-33150	Find a solution to the random crashes experienced by customers	All
DBA-33141	[DBMS Object Management] Crash and inoperative state with views	DB2 for LUW
DBA-33108	Oracle 11g and higher already has a PLAN_TABLE installed	Oracle
DBA-33095	Provide support for SQL*PLUS script commands	Oracle
DBA-33076	[Analysts] Unable to run an analysis report	SQL Server
DBA-33068	[Enhancement][Functions]Some objects do leave out the schema name in the CREATE	SQL Server
DBA-33033	Customer case 00443195: Schedule of Capacity Analyst does not work on machines with Windows in Spanish	SQL Server
DBA-32999	[DBMS Object Management] Database option "allow incremental dumps" is not shown for Sybase 15.7 SP130	Sybase ASE
DBA-32987	[Edit Data] Copy-Paste from Edit Data grid returns weird characters	SQL Server
DBA-32967	[Schema extraction/publication] Can't execute schema extraction and schema publication with some DBMS	SQL Server
DBA-32960	In Edit Data, the DELETE of a row is not actually performed in the table	SQL Server
DBA-32959	[Enhancement]Migrate Database object is missleading	Sybase ASE
DBA-32945	[DBMS Object Management] Can't exchange a table partition	Oracle
DBA-32943	Unable to kill sessions from the Lock tab in Database monitoring	Sybase ASE

DBA-32935	Analyst update error on Oracle datasources ORA-01031: insufficient privileges	Oracle
DBA-32713	[Utilities] No records are migrated with migrated feature	Sybase ASE
DBA-32698	[ISQL] Crashes in Isql when running multiple scripts	All
DBA-31366	[DBMS Actions] Tables based on materialized views do not get extracted with the appropiate unique index	Oracle

Bugs Fixed in Rapid SQL 2016 (16.0.0)

Issue #	Description	DBMS
RAP-5970	Customer case 00430700 - Rapid SQL generates a corrupt xlsx file when importing results to Excel	Oracle
RAP-5865	[Tools] Ability to save a .csv file from results where all the fields are enclosed in double quotes.	Oracle
RAP-5742	508 compliance for Rapid SQL: JAWS incorrectly reads the drop-down options that have seperators	

What Was New in Past Releases

- What's New in Rapid SQL XE6 Patch
- What's New in Rapid SQL XE6
- What's New in Rapid SQL XE5
- What's New in Rapid SQL XE4
- What's New in Rapid SQL XE3.5

What's New in Rapid SQL XE6 Patch

New features for this release fall into the following categories:

PostgreSQL Object Management

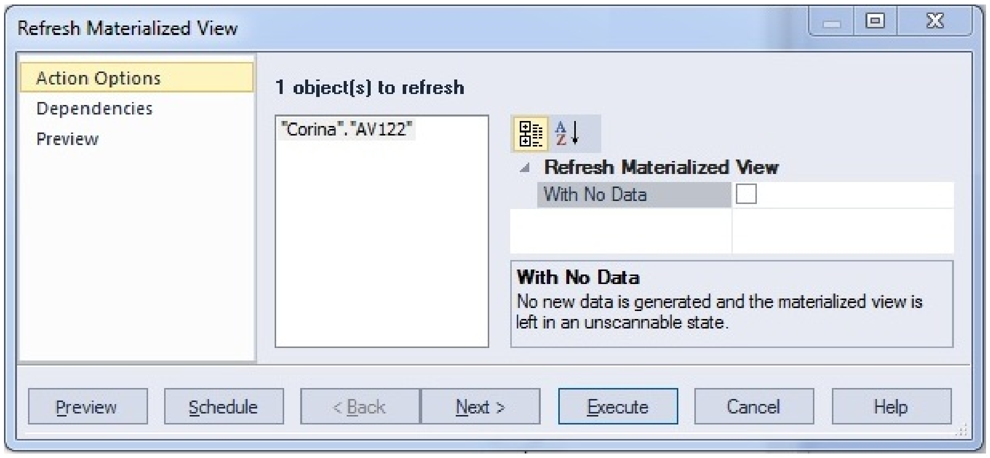
The following topics describe PostgreSQL object management updates for this release.

Object Wizards/Editors

You can now create and edit objects of the following types: check constraints, domains, exclusion constraints, foreign keys, functions, indexes, primary keys, roles, rules, schemas, tables, tablespaces, triggers, unique keys, and views.

Refresh Materialized View Object Action

A new **Refresh Materialized View** action is available against views created using the CREATE MATERIALIZED VIEW statement. It lets you build and submit a REFRESH MATERIALIZED VIEW statement. Simple default behavior replaces the contents of a materialized view. If **With No Data** is selected, no new data is generated and the materialized view will be in an unscannable state.



1 object(s) to refresh

"Corina"."AV122"

Refresh Materialized View

With No Data ☐

With No Data
No new data is generated and the materialized view is left in an unscannable state.

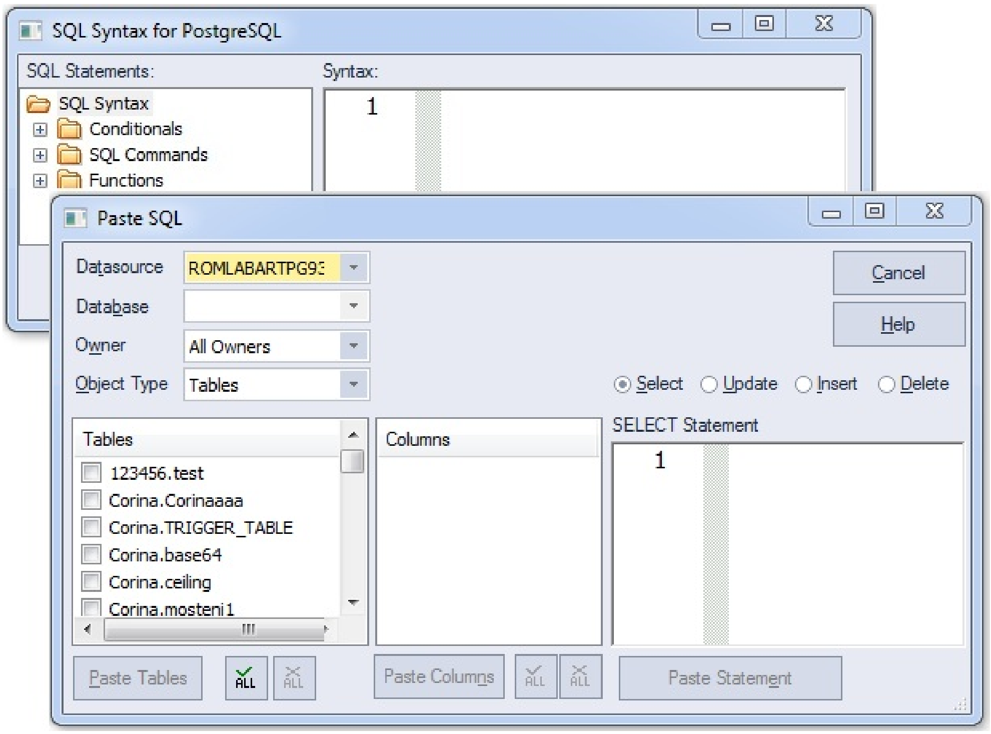
PostgreSQL Tools Menu Updates

The following Tools menu features are now available against PostgreSQL datasources:

- **Database Search** lets you search for objects whose DDL contains a specified character string, across multiple databases.
- **Query Builder** lets you construct, structure, and manipulate queries using a graphical interface.
- **Data Editor** lets you edit table data in real time.

PostgreSQL Updates to the SQL Editor

The Paste SQL Syntax and Paste SQL Statement features are now available against PostgreSQL datasources.



Welcome Window

The **Welcome Window** helps customers to get information about product features, news and upcoming events. This window also includes direct links to useful Rapid SQL features like **New Project**, **Open Project** or links to **Recent Datasources**.

User Interface Changes

- o For the Project Management feature, subproject nodes have been renamed to project subfolder nodes.
- o The Facourites feature has been renamed to Script Library.

Datasource Registration Updates

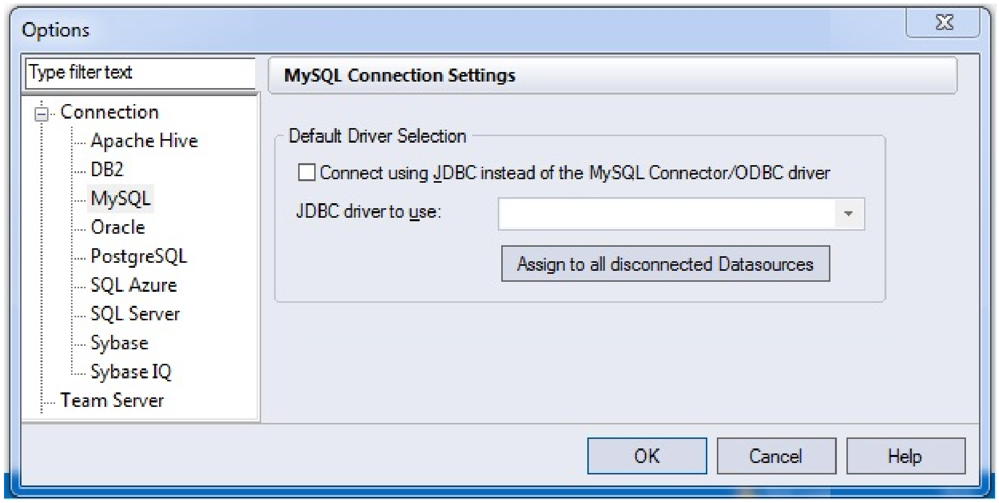
For connectivity purposes, Rapid SQL is packaged with a native driver as well as a set of one or more JDBC drivers for each DBMS platform. In previous releases, the native driver for each platform was the default connection option when registering a datasource.

This release introduces two enhancements:

- o You can now designate one of the JDBC drivers as the default connectivity option for a DBMS platform.

- You can universally change the definition for all existing, currently unconnected datasources for a platform to use either the native driver or a JDBC driver as the connection option.

In the Options Editor, the **Connections** page now has a tab corresponding to each supported DBMS platform. The **Default Driver Selection** group of settings provides the new functions.


Options

Type filter text

- Connection
  - Apache Hive
  - DB2
  - MySQL
  - Oracle
  - PostgreSQL
  - SQL Azure
  - SQL Server
  - Sybase
  - Sybase IQ
- Team Server

MySQL Connection Settings

Default Driver Selection

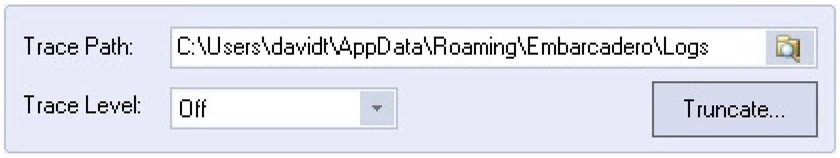
☐ Connect using JDBC instead of the MySQL Connector/ODBC driver

JDBC driver to use:

For DBMS platforms with a native driver and one or more JDBC drivers packaged, the **Connect using...** and **JDBC driver to use** controls enable selection of a JDBC driver as the default connection option and let you select a JDBC driver. The **Assign to all disconnected Datasources** control assigns the selected JDBC driver as the connection option for all disconnected datasources for the current platform, if the **Connect using...** control is selected. Otherwise it assigns the native driver as the connection option for all disconnected datasources for the current DBMS platform

Trace File Logging Options

As a new diagnostic tool, you can now generate a trace file consisting of the sequence of application event messages. The Options Editor's **Logging** page now lets you activate and deactivate, specify a location for, and select a severity level of messages logged.


Trace Path: C:\Users\davidt\AppData\Roaming\Embarcadero\Logs

Trace Level: Off

In descending order of severity (and ascending order of total messages logged), **Trace Level** options are **Fatal Errors**, **Errors**, **Warning**, **Info**, **Debug**, **Trace**, and **Max**.

Version Control Updates

Rapid SQL now supports any source control product that uses the MSSCCI plug-in interface.

Note: For the 64-bit version of Rapid SQL, a 32-bir version of the MSSCCI provider can be used for version control access. tThe feature is controlled fro the Options Editor's **Version Control** tab (File > Options > General > Version Control).

DBMS Platform Updates

The following topics describe DBMS platform and version updates for this release.

- Sybase IQ 16 - Functionality available for previous Sybase IQ releases is now available against Sybase IQ 16 datasources.
- Apache Hive - This release presents a technical preview of Apache Hive support. As with other technical previews, minimal functionality is provided. Features include datasource registration and connection, a minimal Navigator tree, and a minimal set of object management features.

Object Browser Updates

Improved numeric column filtering in **Object Browser**. Users can now filter numeric columns by using operators (>,=,<,<=,>=). When no operator is used, >= is assumed.

ISQL Windows Backup

For Windows Vista and higher, the auto save feature of Rapid SQL is now connected to the Restart Manager. Also, unlike the previous version (o Windows XP) all modified ISQL windows will have a backup saved at a regular interval, even those ISQL windows that do not have a file name associated with them. In the event of a crash, those ISQL windows will automatically open up on the next restart. If the Restart Manager is able to detect the crash, it will attempt to automatically restart the application for you. On Windows XP, this feature will function as it did in past versions.

Connectivity Improvement

Connectivity has been improved to have a central location in the options page to set the driver for all datasources of a certain type to use. An individual datasource can override this setting, but by default all datasources will obey the global setting. No longer will customers have to change each individual datasource if they want to use a particular driver for their system.

Java Update

Rapid SQL has updated to Java 1.7.

Improved Multi-Monitor Support

Users can now detach any MDI tab from the main application window. They can group these detached tabs togetther or keep them separate.

Topics

- Release Notes for Rapid SQL XE6 Patch
- What Was New in Past Releases

What's New in Rapid SQL XE6

As of this release, support is provided for Postgres datasources. The following topics provide a summary of the new functionality.

Version Support and Connectivity Options

Postrgres versions 9.3 and higher are supported. Connectivity is available with the PostgreSQL ODBC Driver (version 9.02 minimum) or with the pre-packaged PostgreSQL JDBC Driver.

Command Line Startup Against Postgres Datasources

As with other platforms, command line startup against Postgres datasources is available.

In Rapid SQL, you can use the following syntax options:

rsql.exe -D*datasource*-U*username* [-P*password*]

rsql.exe -R*connectionstring*-D*datasource*-U*username* [-P*password*]

The *connectionstring* can take the following form:

postgresql://*HOST*

postgresql://*HOST*/*DB*

postgresql://*HOST*:*PORT*

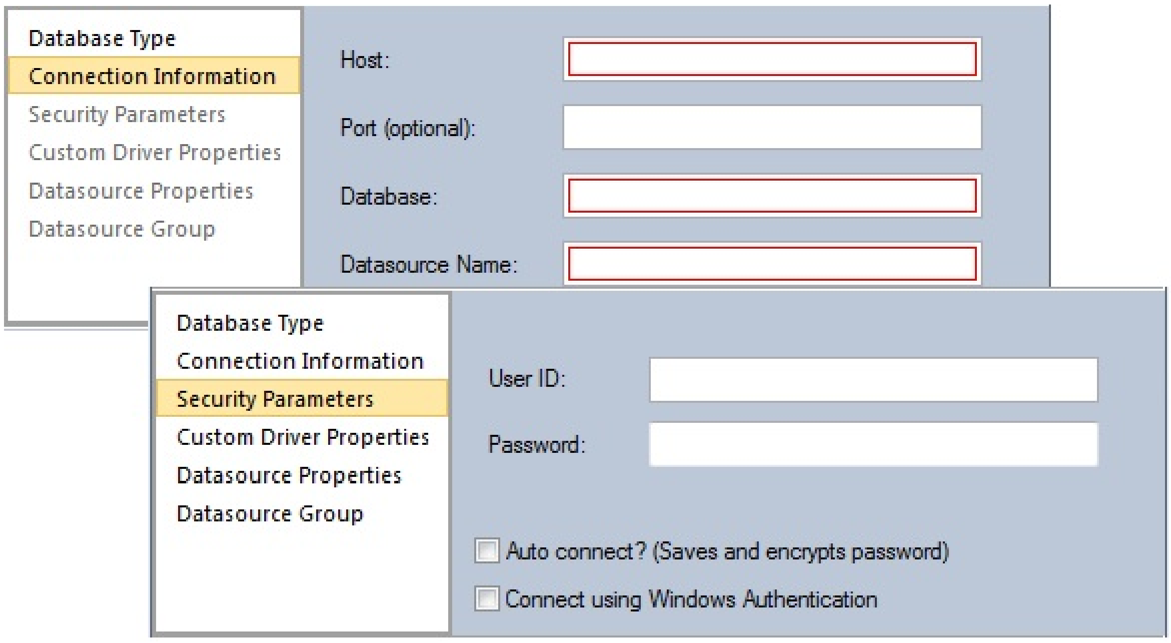
postgresql://*HOST*:*PORT*/*DB*

NOTE:Using **postresqlp** instead of **postgreql** will make the datasource created permanent.

If the Teradata ODBC driver is installed, the connection will be established using that driver. Otherwise, the Teradata JDBC driver will be used.

Standard Datasource Registration and Connectivity

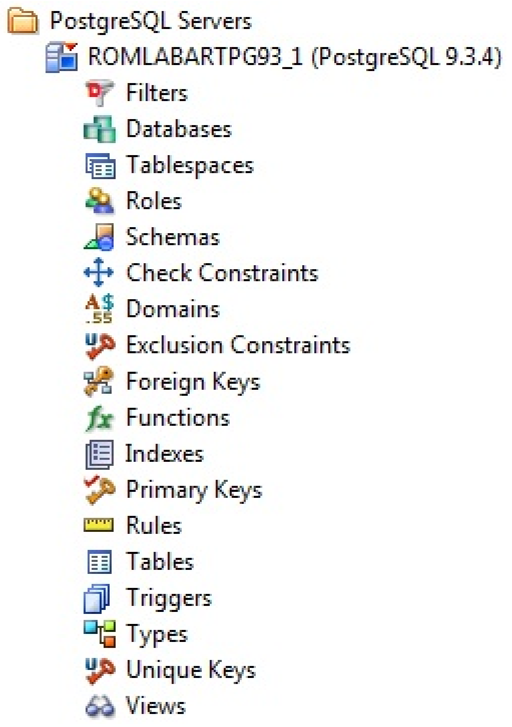
Currently, you can manually register a Postgres datasource, identifying a host and database name, and optionally providing a port number. Basic User ID and Password credentials can be provided.



Similarly, a login dialog, prompting for a user name and password, lets you connect to a Teradata datasource.

Datasource Explorer/Navigator Object Node Availability

In Rapid SQL, object nodes for the new platform are as follows:



- PostgreSQL Servers
  - ROMLABARTPG93_1 (PostgreSQL 9.3.4)
    - Filters
    - Databases
    - Tablespaces
    - Roles
    - Schemas
    - Check Constraints
    - Domains
    - Exclusion Constraints
    - Foreign Keys
    - Functions
    - Indexes
    - Primary Keys
    - Rules
    - Tables
    - Triggers
    - Types
    - Unique Keys
    - Views

Object actions **Drop** and **Extract** are available for each supported object type. In addition, some object type-specific actions, such as **Select * From**, **Rename**, **Change Schema**, and **Change Owner** are available.

SQL Editor

Execution against Postrges sources is available. Also available are related, common SQL Editor tools such as **Query Options**, **Paste SQL Syntax**, and **Paste SQL Statement**.

What's New in Rapid SQL XE5

New features for this release fall into the following categories:

Team Server 2016

Functionality previously provided by Embarcadero Connect, is now provided by Team Server 2016. Like Connect, Team Server 2016 is a data governance repository that stores an inventory of datasource definitions and ER/Studio data models.

In addition to functionality provided by Connect, the new product also offers the following features:

- An advanced search mechanism lets you locate datasource definitions by a variety of criteria, including DBMS platform, version, and host name.You can also search on operational characteristics such as space statistics and ping time.
- Users can monitor the activity stream associated with their Team Server 2016 account or the activity stream for a datasource definition. Streams include user posts, Likes, and Follows.

64-bit Installation

Rapid SQL and DBArtisan now run as native, 64-bit applications. Separate installers let you choose between 32-bit and 64-bit applications.

64-bit versions of DBMS platform drivers are installed when you select that option. If you are using custom drivers, as of this release, you must be using 64-bit versions of those custom drivers when using the 64-bit version of Rapid SQL or DBArtisan.

Windows 8 Support

Rapid SQL and DBArtisan are now fully supported on the Windows 8 operating system.

Teradata

As of this release, Rapid SQL and DBArtisan provide support for Teradata versions 13.10, 14, and 14.10. The following topics provide a summary of the new functionality.

Command line startup against Teradata datasources

As with other platforms, command line startup is available.

Rapid SQL offers the following syntax options:

```
rsql.exe -D datasource -U username -P password
```

```
rsql.exe -R ‘connectionstring’ -D datasource -U username -P password
```

The connectionstring can take the following form:

```
teradata://HOST
```

```
teradata://HOST/DB
```

```
teradata://HOST:PORT
```

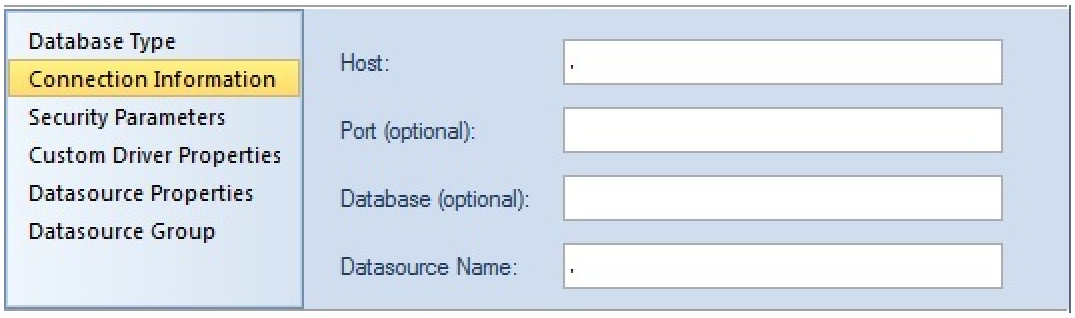
```
teradata://HOST:PORT/DB
```

Using `teradatap` will make the datasource created permanent.

If the Teradata ODBC driver is installed, the connection will be established using that driver. Otherwise, the Teradata JDBC driver will be used.

Standard Datasource Registration and Connectivity

Currently, you can manually register a Teradata datasource, identifying a host and optionally providing port and database name information.

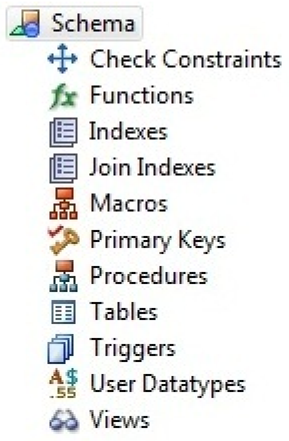


Host:	.
Port (optional):	
Database (optional):	
Datasource Name:	.

Similarly, a login dialog, prompting for a user name and password, lets you connect to a Teradata datasource.

Datasource Explorer/Navigator Node Availability

In Rapid SQL, object nodes for the new platform are as follows:



Object actions such as **Object Properties**, **Execute**, **Edit Data**, **Build Query**, **Select * From**, and **Schema** are available, as appropriate to the object type.

SQL Editor

Execution against Teradata sources is available. Also available are related, common SQL Editor tools such as **Query Options**, **Paste SQL Syntax**, and **Paste SQL Statement**.

Other New DBMS Version Support

As of this release, Rapid SQL and DBArtisan provide full support for the following DBMS platform versions:

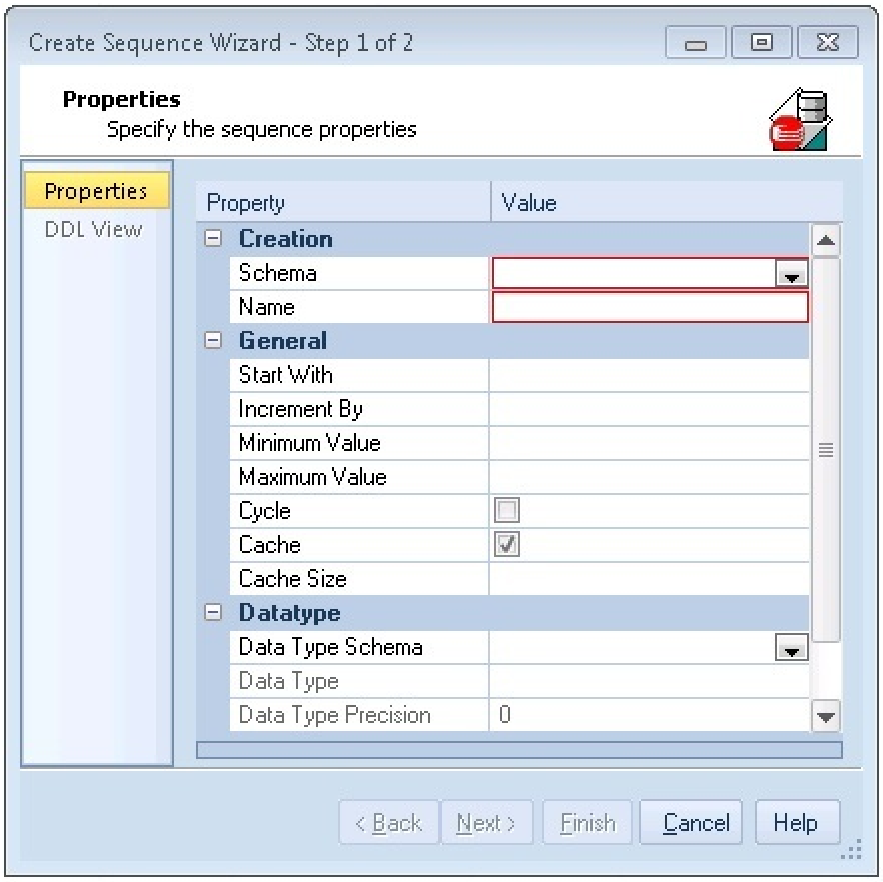
- IBM DB2 LUW version 10
- IBM DB2 z/OS version 10
- Oracle 12c
- Sybase ASE 15.7.

SQL Server Updates

The following topics describe SQL Server updates for the current release.

Sequence Object Type

The sequence object type, introduced with SQL Server version 12 is now supported. A creation wizard, editor, and a basic set of object actions are provided.

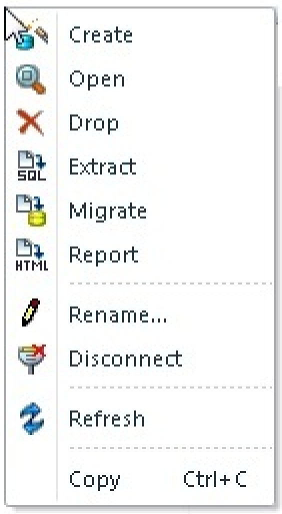


Create Sequence Wizard - Step 1 of 2

Properties

Specify the sequence properties

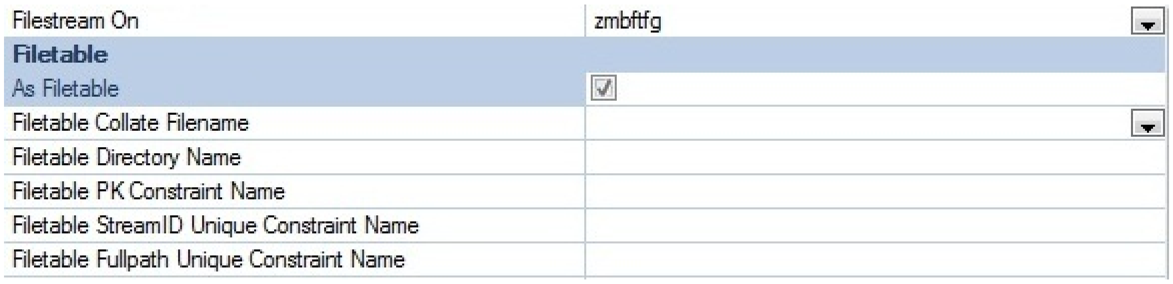
Property	Value
Creation	
Schema	
Name	
General	
Start With	
Increment By	
Minimum Value	
Maximum Value	
Cycle	☐
Cache	✓
Cache Size	
Datatype	
Data Type Schema	
Data Type	
Data Type Precision	0



FileTable and FileStream updates to tables

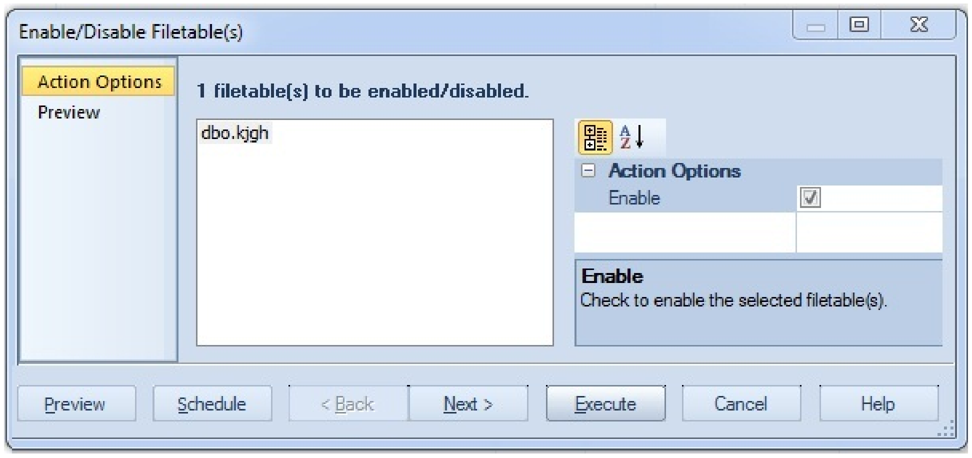
When creating or editing SQL Server tables, a new set of properties is available. They let you:

- Specify a filegroup for the table by providing a FILESTREAM_ON argument value
- Create the table as a FileTable, with a full set of FileTable arguments.



Filestream On	zmbftfg
Filetable	
As Filetable	✓
Filetable Collate Filename	
Filetable Directory Name	
Filetable PK Constraint Name	
Filetable StreamID Unique Constraint Name	
Filetable Fullpath Unique Constraint Name	

A new **Enable/Disable Filetable(s)** action lets you build and submit an ALTER TABLE statement, specifying either an ENABLE FILETABLE_NAMESPACE or DISABLE FILETABLE_NAMESPACE argument. This enables or disables system-defined constraints on a FileTable.


Enable/Disable Filetable(s)

Action Options
Preview

1 filetable(s) to be enabled/disabled.

dbo.kjgh

Action Options
Enable ✓

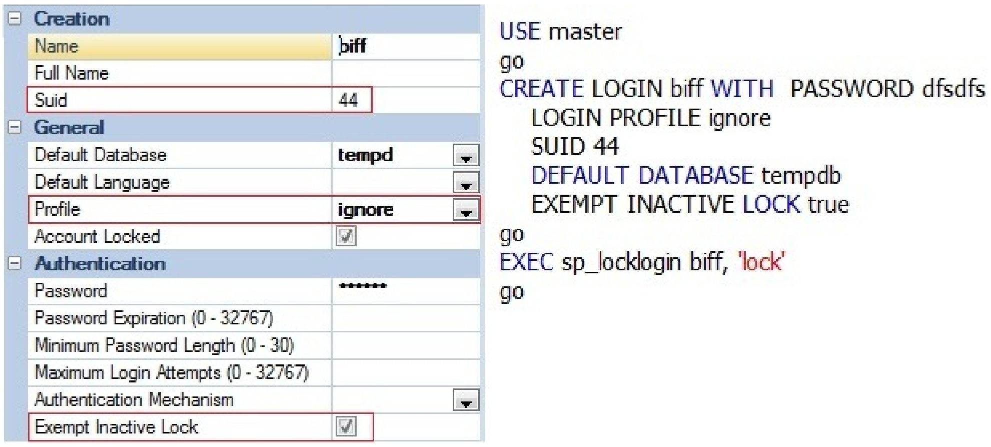
Enable
Check to enable the selected filetable(s).

DB2 LUW Updates

The **Update Statistics** object action has been updated to use the ADMIN_CMD system procedure. This eliminates the constraint of having target datasources registered by alias and using the native client. **Update Statistics** can now be used against all datasources, regardless of how they were registered.

Sybase ASE Updates

Against version 15.7 datasources, the Login Wizard/Editor now generates DDL that uses CREATE/ALTER LOGIN syntax. New properties let you work with SUID, LOGIN PROFILE, and EXEMPT INACTIVE LOCK parameters.



Creation	
Name	biff
Full Name	
Suid	44
General	
Default Database	tempd
Default Language	
Profile	ignore
Account Locked	✓
Authentication	
Password	******
Password Expiration (0 - 32767)	
Minimum Password Length (0 - 30)	
Maximum Login Attempts (0 - 32767)	
Authentication Mechanism	
Exempt Inactive Lock	✓

```
USE master
go
CREATE LOGIN biff WITH  PASSWORD dfsdfs
    LOGIN PROFILE ignore
    SUID 44
    DEFAULT DATABASE tempdb
    EXEMPT INACTIVE LOCK true
go
EXEC sp_locklogin biff, 'lock'
go
```

The **Change Login Password**, **Add/Modify Login Trigger**, and **Drop Login Trigger** actions have also been updated to generate DDL that uses ALTER LOGIN syntax.

Oracle Updates

If the Oracle SQL*Net Client is not installed, Rapid SQL and DBArtisan will attempt to connect to an Oracle datasource using the Oracle Database Instant Client.

Query Builder

As of this release, when you add the same table to the diagram pane multiple times, a new alias is automatically assigned to each instance. Adding a table multiple times can be useful when selecting additional data from the table, based on different join criteria.

What's New in Rapid SQL XE4

New features for this release fall into the following categories:

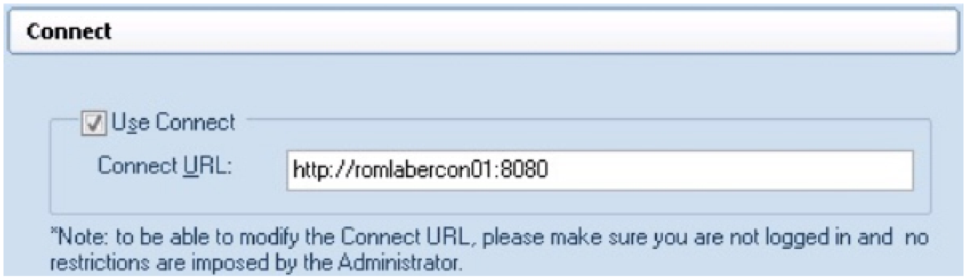
- Data Governance
- Other Datasource Registration Updates
- User Interface and Navigator/Explorer Updates
- SQL Server Updates
- Oracle Updates

Data Governance

Rapid SQL now integrates with Embarcadero Connect, Embarcadero's data governance repository. This lets Rapid SQL participate in governance schemes aimed at control and centralization of data-related resources, providing controlled access to those resources, and distributing relevant information on those resources.

Enabling and Connecting to an Embarcadero Connect Server

The Options Editor's **Connect** tab lets you configure Embarcadero Connect server usage:

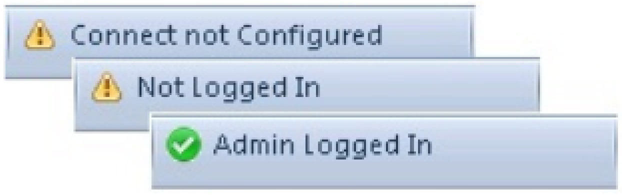

Connect

[✓] Use Connect

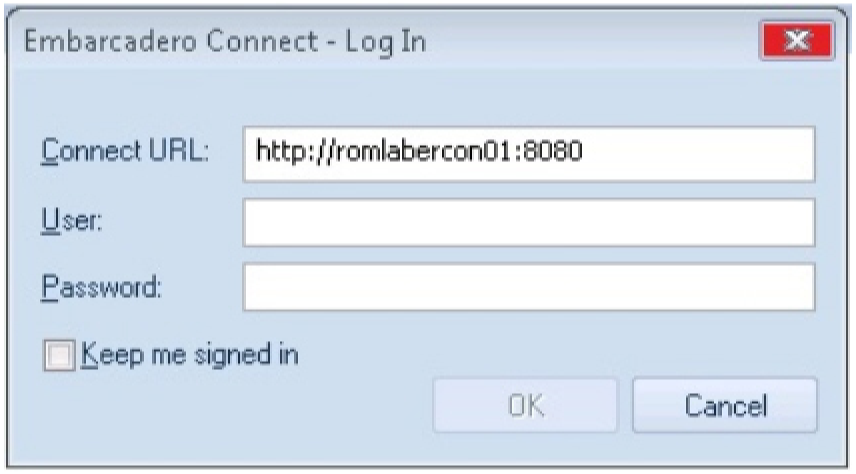
Connect URL: http://romlabercon01:8080

*Note: to be able to modify the Connect URL, please make sure you are not logged in and no restrictions are imposed by the Administrator.

Use Connect enables connections to an Embarcadero Connect server while **Connect URL** lets you provide an address. A control on the main window's status bar provides current configuration/status information on the Connect server.



Double-clicking the control initiates an action appropriate to the current status. For example if Connect server usage is enabled and properly configured, but Rapid SQL is not currently logged in, then double-clicking the status control opens a login dialog.



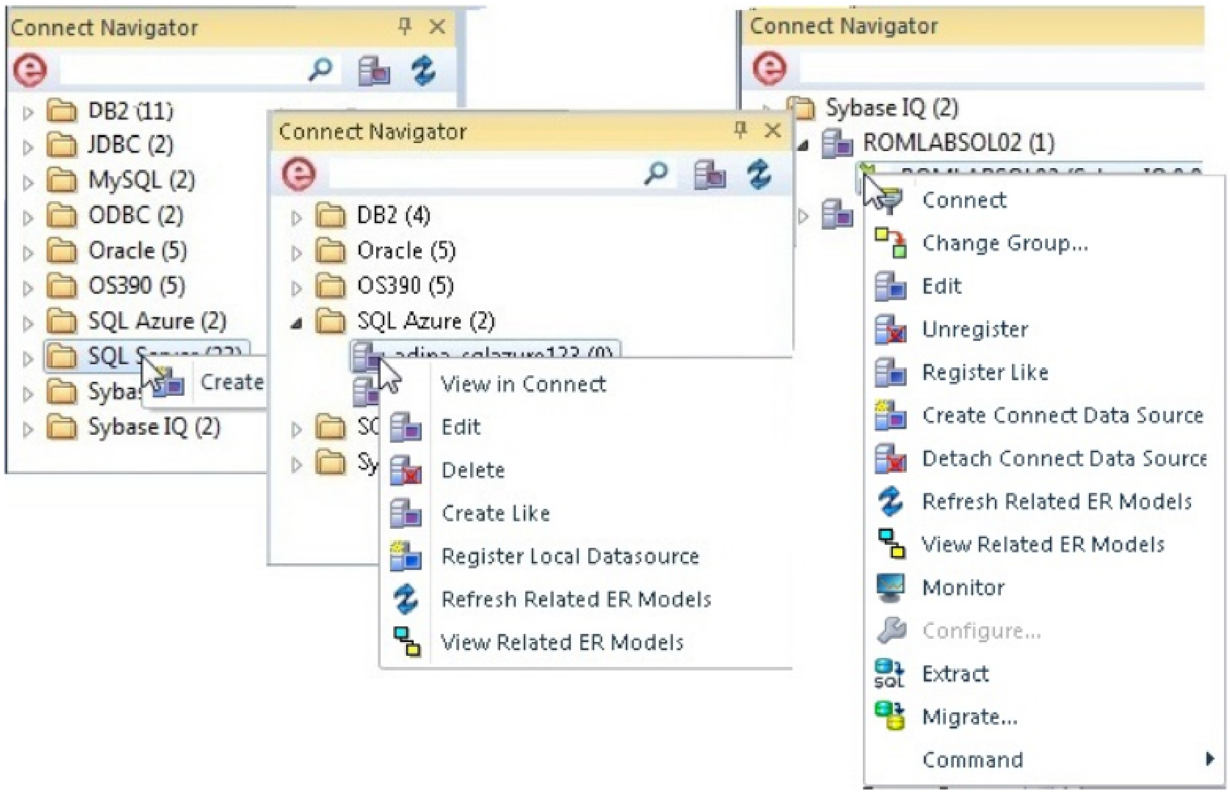
http://romlabercon01:8080

Three new HKLM/HKCU registry settings (ConnectURL, UseConnect, and AllowKeepMeSignedIn) let you lock down those features.

Embarcadero Connect and Datasource Management

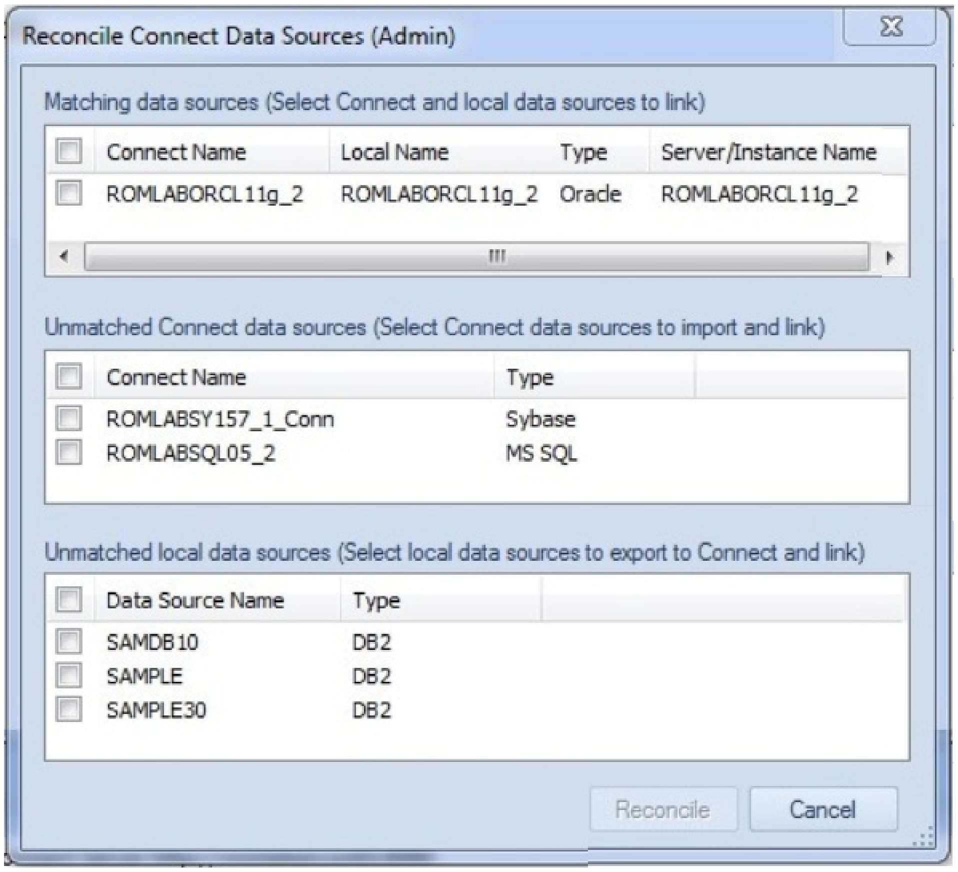
A new Connect Navigator, similar to the existing Navigator/Explorer, lets you work with datasource definitions residing on an Embarcadero Connect server. The Navigator hierarchy shows

- Datasource groups
- Datasource definitions residing on the Embarcadero Connect repository
- Locally datasources that were registered using definitions in the Embarcadero Connect repository



In order to make use of a Connect datasource, you must register it as a local datasource. This creates a local copy of the datasource definition, accessible in the existing Navigator/Explorer. Key connectivity information, the **Connection Information** tab settings (excluding **Datasource Name**) and a custom JDBC driver if supplied, is not editable in the resulting local datasource definition. **Security Parameters** tab information, such as **User ID** and **Password**) is not copied to the local datasource definition. By default, when you register a Connect server datasource as a local datasource, the local definition is "attached" to the Connect version. The local definition's key connectivity information is synchronized with the Connect server definition on each attempt to connect to the datasource. An attached datasource has a distinctive icon:

You can subsequently detach an attached datasource definition, leaving that definition available as a local datasource. The datasource will no longer be synchronized with the corresponding definition on the Connect server. While the attach and detach operations can be used against individual datasource definition, the Connect Navigator also offers a **Reconcile Datasources** operation that lets you manage multiple Connect datasources simultaneously.



Matching data sources (Select Connect and local data sources to link)

	Connect Name	Local Name	Type	Server/Instance Name
☐	ROMLABORCL11g_2	ROMLABORCL11g_2	Oracle	ROMLABORCL11g_2

Unmatched Connect data sources (Select Connect data sources to import and link)

	Connect Name	Type
☐	ROMLABSY157_1_Conn	Sybase
☐	ROMLABSQL05_2	MS SQL

Unmatched local data sources (Select local data sources to export to Connect and link)

	Data Source Name	Type
☐	SAMDB10	DB2
☐	SAMPLE	DB2
☐	SAMPLE30	DB2

The three lists offer the following options:

- **Matching data sources**: lists Embarcadero Connect server datasource definitions and local datasource definitions with identical connection information details but that are currently not attached. Selecting an entry from this list attaches the local datasource to the Connect datasource, ensuring that any changes to the Connect server definition will be copied to the local datasource definition on the next attempt to connect to the datasource
- **Unmatched Connect data sources**: lists Embarcadero Connect server datasource definitions for which there are no local datasource definitions with matching connection information. Selecting an entry from this list creates a matching local datasource definition and attaches the local definition to the matching Connect server datasource definition.
- **Unmatched local data sources**: lists local datasource definitions for which there are no Embarcadero Connect server datasource definitions with matching connection information. Selecting an entry from this list creates a matching Connect server datasource definition and attaches the local definition to the Connect server datasource definition.

In addition to local Rapid SQL use of Connect datasources, the Connect Navigator lets you create, edit, and delete datasource definitions on the Connect server.

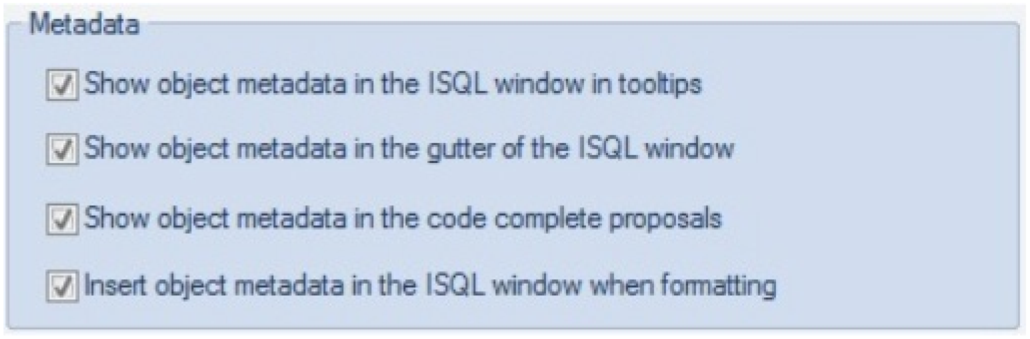
Embarcadero Connect Model Metadata and the SQL Editor

Rapid SQL integrates with Embarcadero Connect, a product that allows easy distribution of data governance repository metadata to the enterprise. Specifically, if a database or schema has associated model data in Connect, model data such as alerts and attribute/entity descriptions can be displayed/included in the SQL Editor in the following ways:

- Attribute/Entity descriptions can be included in Code Assist proposals
- On explicitly formatting a script, metadata for tables and views referenced in SELECT, CREATE PROCEDURE, CREATE FUNCTION, CREATE TRIGGER, CREATE VIEW statements are added in a single comment block above the referencing statement
- When hovering the mouse over a table column name, table name or view name, a tooltip showing the entity/attribute description is displayed
- Warning icons are displayed in the window gutter beside lines referencing objects whose metadata includes alerts
- On execution of a script referencing objects whose metadata includes alerts, the error pane opens with an entry for each referenced object.

To use model metadata in the SQL Editor

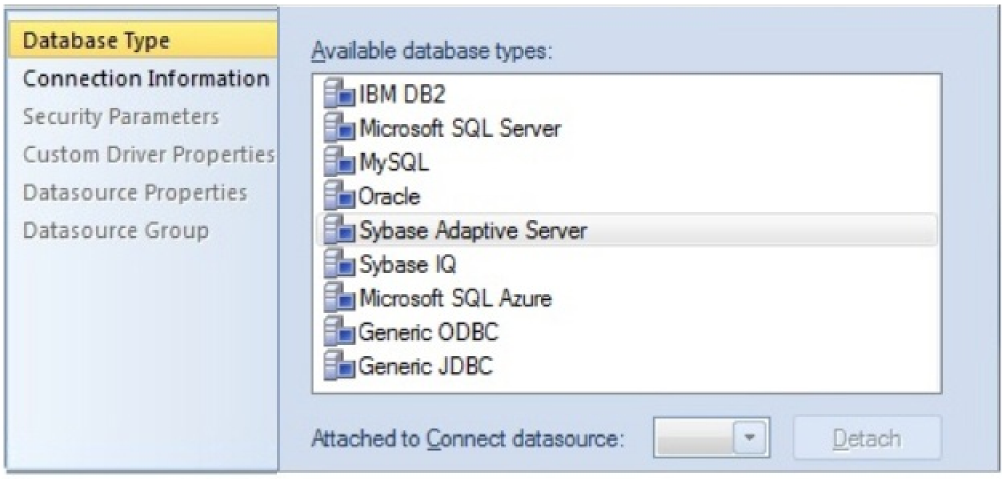
1. Ensure that Connect repository usage is enabled and that you are logged in to the Connect Server.
2. Configure the **Metadata** features on the **ISQL > Code Assist** page of the Options editor.

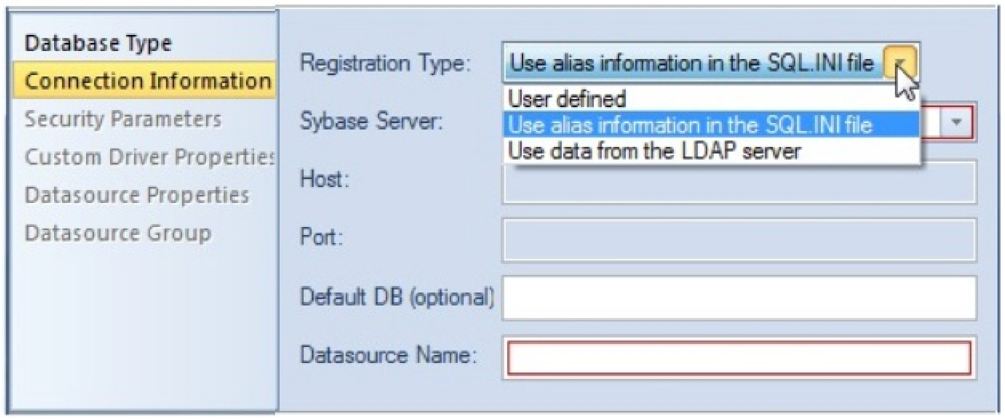

Metadata

- ✓ Show object metadata in the ISQL window in tooltips
- ✓ Show object metadata in the gutter of the ISQL window
- ✓ Show object metadata in the code complete proposals
- ✓ Insert object metadata in the ISQL window when formatting

Other Datasource Registration Updates

The Datasource Registration Wizard has been simplified so that all connection information is, as in earlier releases, provided on a single page.





Registration Type: Use alias information in the SQL.INI file

Additional improvements include:

- o When using the Sybase JConnect driver, you now provide the sql.ini alias name as opposed to the host and port.
- o When using the Oracle Thin JDBC driver, Rapid SQL now lets you provide a tnsnames.orai alias name as opposed to host, port, and SID/service details.

In both cases, if the information in the alias file changes, those changes do not have to be made manually in Rapid SQL.

User Interface and Navigator/Explorer Updates

Bookmarks Enhancements

The number of Navigator/Explorer Bookmarks you can create has been increased from 64 to 256. In addition, Rapid SQL will no longer allow you to create multiple bookmarks referencing the same Navigator/Explorer node.

SQL Server Updates

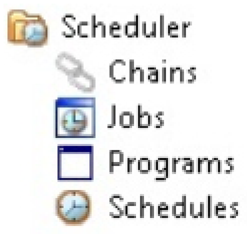
SQL Server 12 SQL Features

This release introduces dedicated SQL Editor support for SQL Server 12 Transact SQL. Rapid SQL features such as SQL execution, debugging, formatting/editing, and automated error detection and coding assistance, now work seamlessly against SQL Server version 12. Specific updates have been made to accommodate reserved words/keywords, buillt-in functions, and updates to the ALTER, CREATE, DROP and other statements.

Oracle Updates

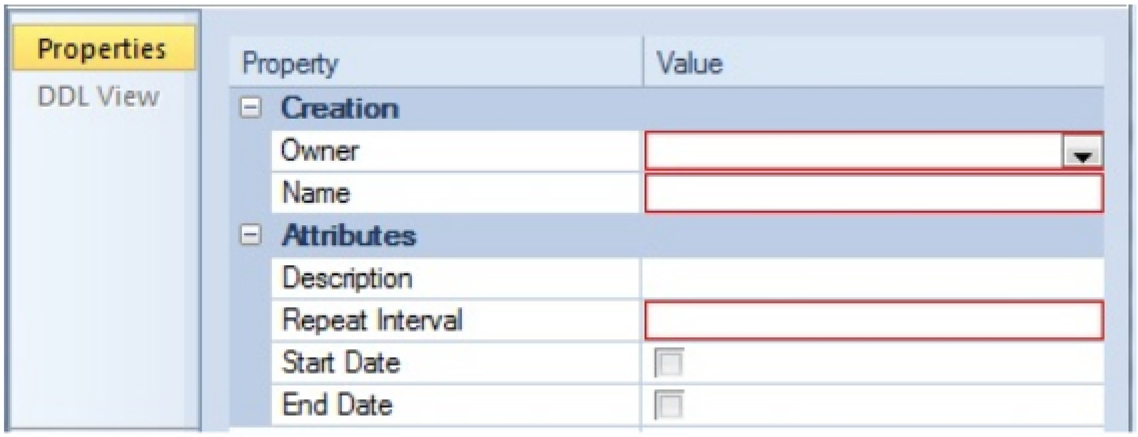
DBMS_SCHEDULER Support

In previous releases, Rapid SQL supported the DBMS_JOB package through its Oracle Job Queue mechanism. As of this release, Rapid SQL provides support for the DBMS_SCHEDULER package. Support includes all DBMS_SCHEDULER action types, Oracle's calendaring syntax, as well as commonly used attributes. DBMS_SCHEDULER support is implemented in Rapid SQL with four new object types, available in the Datasource Navigator/Explorer under a new **Scheduler** node:



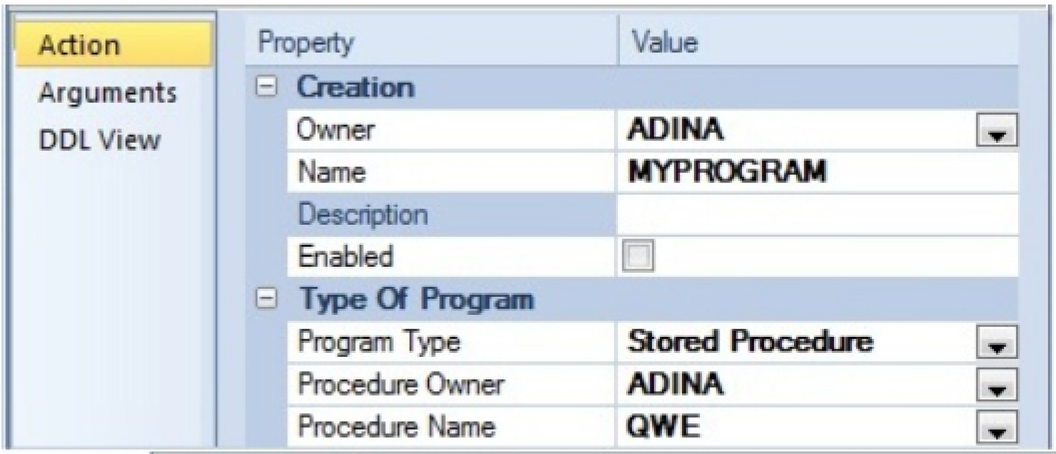
Three new wizards/editors are available:

- **Schedules Wizard**: Builds and submits a call to DBMS_SCHEDULER.CREATE_SCHEDULE call.

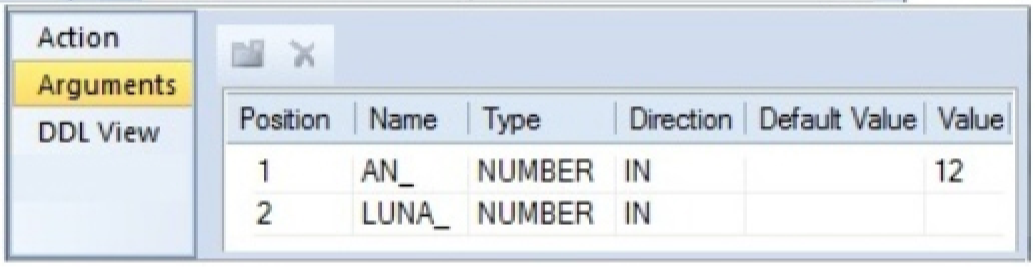


Property	Value
Creation	
Owner	
Name	
Attributes	
Description	
Repeat Interval	
Start Date	☐
End Date	☐

- **Programs Wizard**: Builds and submits a call to DBMS_SCHEDULER.CREATE_PROGRAM with optional DBMS_SCHEDULER.DEFINE_PROGRAM_ARGUMENT calls, letting you create a program.

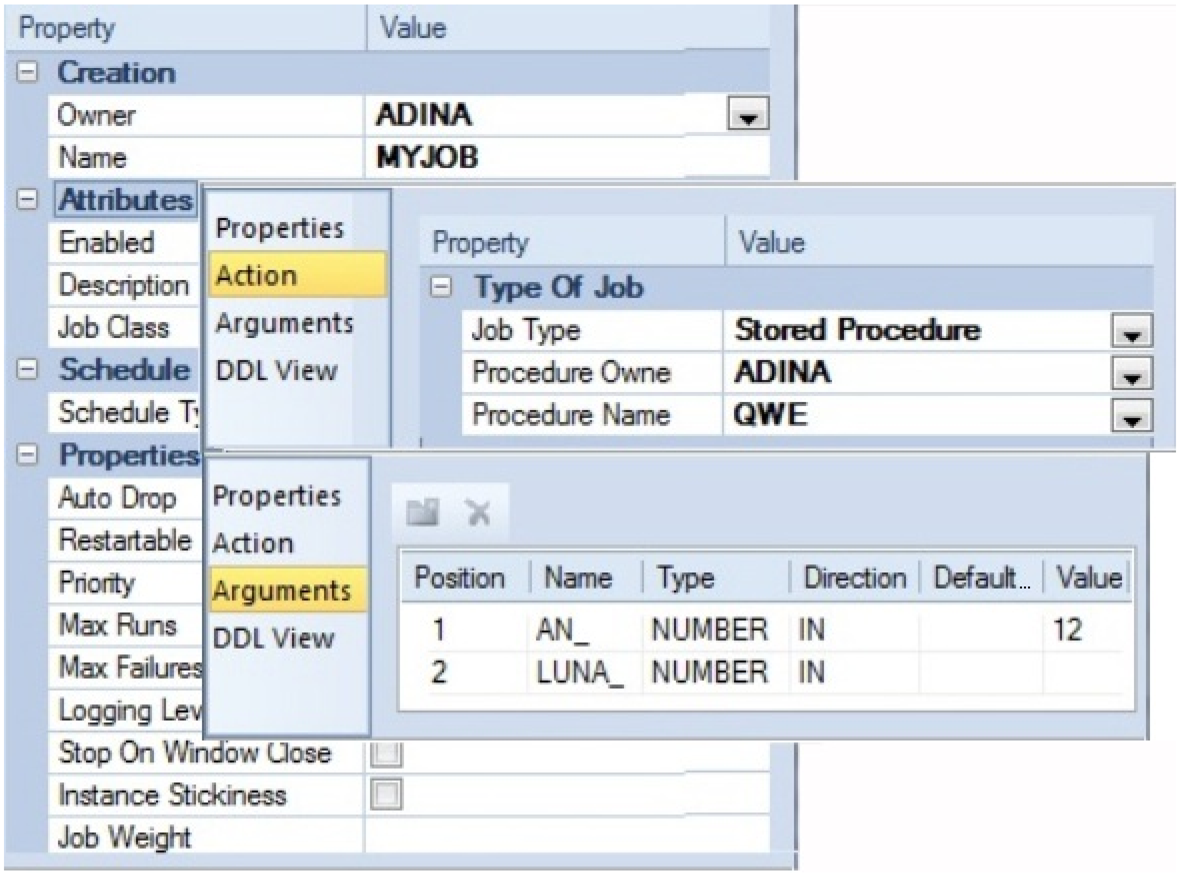


Property	Value
Creation	
Owner	ADINA
Name	MYPROGRAM
Description	
Enabled	☐
Type Of Program	
Program Type	Stored Procedure
Procedure Owner	ADINA
Procedure Name	QWE



Position	Name	Type	Direction	Default Value	Value
1	AN_	NUMBER	IN		12
2	LUNA_	NUMBER	IN		

- Jobs Wizard: Builds and submits a call to DBMS_SCHEDULER.CREATE_JOB with optional DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE, DBMS_SCHEDULER.SET_ATTRIBUTE, and DBMS_SCHEDULER.ENABLE calls.



Property	Value
Creation	
Owner	ADINA
Name	MYJOB
Attributes	
Enabled	
Description	
Job Class	
Schedule	
Schedule Ty	
Properties	
Auto Drop	
Restartable	
Priority	
Max Runs	
Max Failures	
Logging Lev	
Stop On Window Close	☐
Instance Stickiness	☐
Job Weight	

Property	Value
Type Of Job	
Job Type	Stored Procedure
Procedure Owne	ADINA
Procedure Name	QWE

Position	Name	Type	Direction	Default...	Value
1	AN_	NUMBER	IN		12
2	LUNA_	NUMBER	IN		

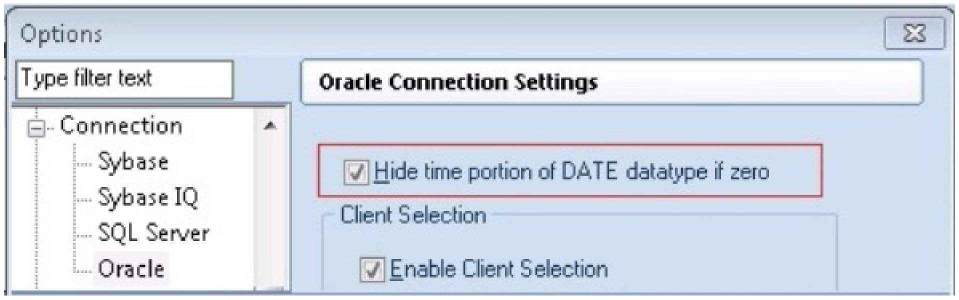
Against jobs, three DBMS_SCHEDULER-specific object actions are available:

- o **Enable/Disable**: Corresponds to DBMS_SCHEDULER.DISABLE or DBMS_SCHEDULER.ENABLE calls.
- o **Run Job**: Corresponds to a DBMS_SCHEDULER.RUN_JOB call.
- o **Stop Job**: Corresponds to a DBMS_SCHEDULER.STOP_JOB call.

The **Enable/Disable** action is also available against programs.

Suppression of time component in DATE datatype results

In the Results window of the ISQL Editor, display of Oracle DATE datatype results can now be configured to suppress display of time components (DD:MM:SS) if the time component is all zeroes. The Options Editor (File > Options > Connection > Oracle) now has a **Hide time portion of DATE datatype if zero** control that controls this feature.



☑ Hide time portion of DATE datatype if zero

Client Selection

☑ Enable Client Selection

What's New in Rapid SQL XE3.5

New features for this release fall into the following categories:

Feature Lockdown

Administrators can now restrict certain features using the registry. Registry location **HKLM\Software\Embarcadero\Rapid SQL\Admin** has the following entries:

RestrictedPlatforms (REG_SZ)	A comma-separated list (**db2=0,mssqlserver=1**, for example) dictating DBMS platform support on the machine. Valid names for the name=value pairs are **db2**, **os390**, **mssqlserver**, **mysql**, **oracle**, **sybase**, **interbase**, **mssqlazure**, **odbc**, **jdbc**, and **sybaseiq**. Any non-zero value for a **name=value** pair prevents new datasources for that platform from being registered.
DisableAutoConnect (REG_DWORD)	When set to any non-zero value, users cannot use the auto-connect feature when registering or logging in to a datasource.
SharedDatasourceCatalogPath (REG_SZ)	Providing a path to the network shared datasource catalog forces Rapid SQL to use that datasource catalog storage option. Registry-based and file-based catalog storage will be unavailable on this machine.

Command Line Startup

This release introduces command line startup of Rapid SQL with switches letting you specify a datasource and credentials: '*dbart.exe -Ddatasource*-U*username*-P*password***rsql.exe** -D*datasource*-U*username*-P*password* This is primarily aimed at identity management systems working in conjunction with Rapid SQL. Such systems can be configured to start Rapid SQL and connect to a specific datasource using credentials that are invisible to the user. Other applications that hide credentials, force connection to a specific datasource, or otherwise automate startup of Rapid SQL can make use of this feature.

Connectivity Updates

The following topics describe connectivity updates for this release.

DBMS Version Support updates

Rapid SQL now provides support for Microsoft SQL Server 2012. All functionality supported by Rapid SQL for previous SQL Server versions is supported for the 2012 version. As of this release, Rapid SQL is no longer tested against the following DBMS versions: Microsoft SQL Server 2000 IBM DB2 for Linux, Unix, and Windows version 8.x

Discover Datasources Feature Updated to Detect TOAD Datasources

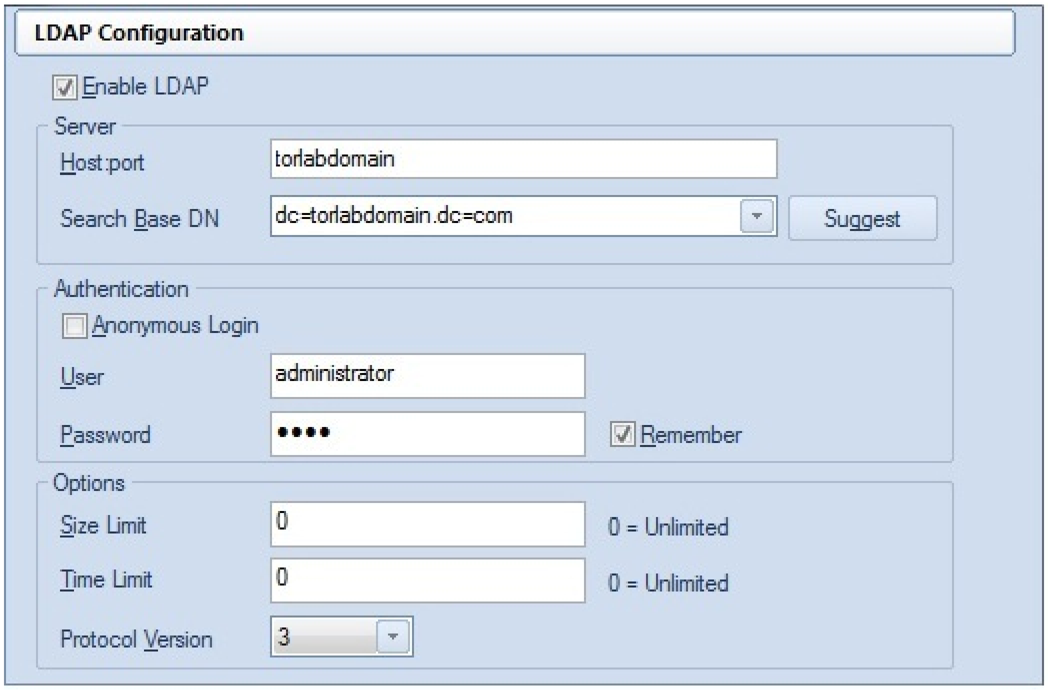
You can now import definitions from the following Quest Software products:

- Toad for MySQL
- Toad for Oracle
- Toad for SQL Server
- Toad for Sybase (ASE and IQ)

The Discover Datasources feature will automatically search registration files for datasource definitions and offer you the option to register any datasources found.

LDAP connectivity option

Rapid SQL now supports LDAP connections to DB2, Oracle, and Sybase ASE datasources. This lets Rapid SQL users of those DBMS products take advantage of the centralized access details and authentication provided by an LDAP Server. The Options Editor now lets you enable LDAP lookup, identify an LDAP Server, provide a distinguished name, and provide associated user login details and options.



LDAP Configuration

☑ Enable LDAP

Server

Host:port	torlabdomain
Search Base DN	dc=torlabdomain.dc=com

Authentication

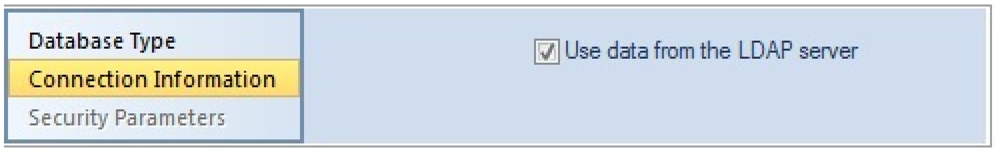
☐ Anonymous Login

User	administrator
Password	•••• ☑ Remember

Options

Size Limit	0	0 = Unlimited
Time Limit	0	0 = Unlimited
Protocol Version	3	

When registering a datasource against one of the supported DBMS platforms, a new **Connection Information** tab control lets you designate the registration as being LDAP-based.

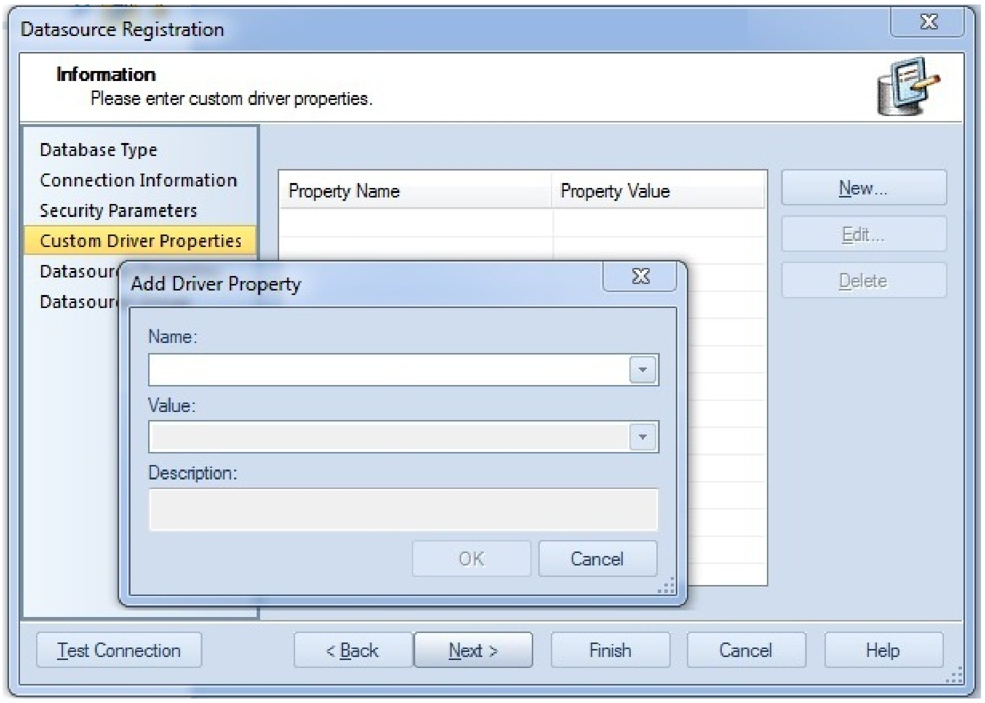


☑ Use data from the LDAP server

You can then select from the servers defined on the specified LDAP Server with availability according to the credentials provided on the Options Editor's **LDAP Configuration** tab. Similarly, with LDAP Server access configured in Rapid SQL, the Discover Datasources feature can detect datasources defined on that server.

Driver Connection String Properties/Keywords Availability With Native Drivers

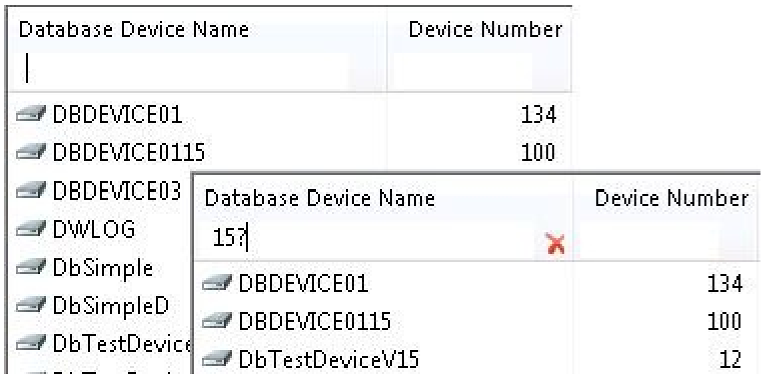
In previous releases, datasource registration allowed basic connection string generation, consisting of server/database identification, security details, and credentials. Other property or keyword specifications could only be provided if you were using a generic JDBC connection. A new **Custom Driver Properties** panel lets you provide custom driver property specifications when creating or editing a datasource definition that uses one of the Rapid SQL native drivers.



For each platform, all connection string parameters/driver properties supported for that driver can be specified. Generic JDBC and ODBC Support Rapid SQL now supports generic JDBC/ODBC connectivity to non-dedicated DBMS systems or non-database datasources. Rapid SQL can connect to a datasource using a customer-provided, third-party JDBC version 3.0 or ODBC version 3.0 driver. Minimal Rapid SQL functionality is provided, including a basic Explorer tree and SQL querying.

Explorer Updates

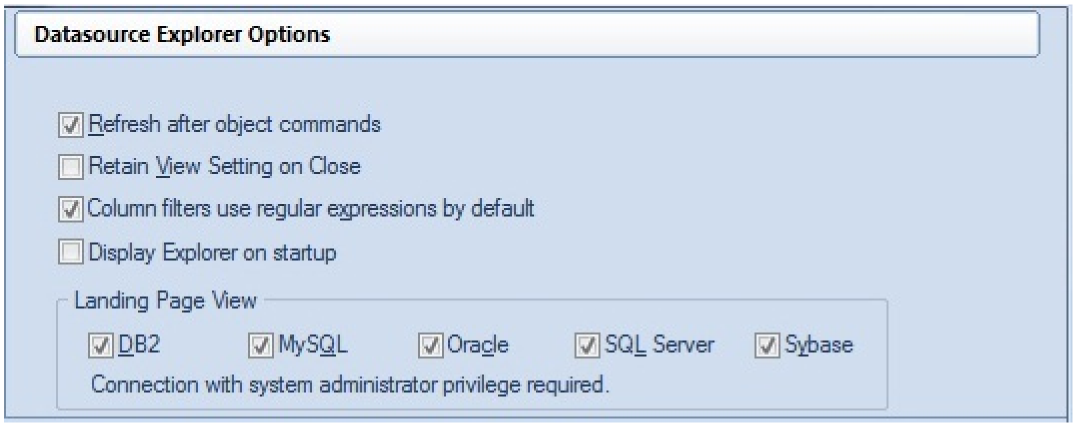
Column Filtering Using Regular Expressions When database objects are displayed in the right-hand pane of the Explorer, column filtering now lets you qualify displayed objects using regular expressions.



Database Device Name	Device Number
DBDEVICE01	134
DBDEVICE0115	100
DBDEVICE03	
DWLOG	
DbSimple	
DbSimpleD	
DbTestDevic	

Database Device Name	Device Number
15?	
DBDEVICE01	134
DBDEVICE0115	100
DbTestDeviceV15	12

This feature must be enabled. The Options Editor’s Explorer pane has a Column filters use regular expressions by default control that enables regular expression usage when Rapid SQL starts or a new Explorer is manually opened.



Datasource Explorer Options

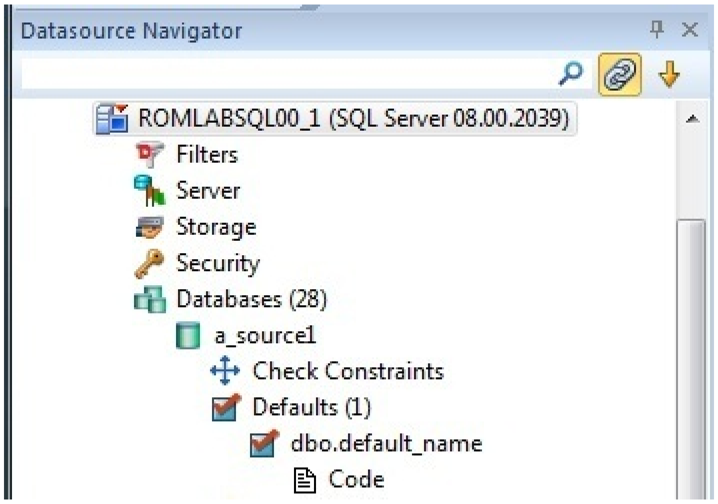
- [✓] Refresh after object commands
- [ ] Retain View Setting on Close
- [✓] Column filters use regular expressions by default
- [ ] Display Explorer on startup

Landing Page View

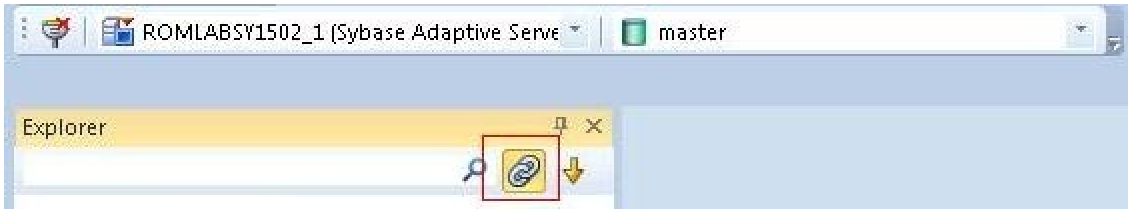
[✓] DB2 [✓] MySQL [✓] Oracle [✓] SQL Server [✓] Sybase

Connection with system administrator privilege required.

In addition, a REGEX On/REGEX Off button in the status bar lets you toggle the feature on and off for a particular window. The Datasource Navigator An alternative to the Datasource Explorer is now available.The Datasource Navigator is a single-pane window that, like the Explorer, lets you navigate and drill down into datasources, as well as initiate actions against database objects. Without the second pane (the object type-based landing page used by the Explorer), the Navigator occupies less space.



The Navigator also provides greater control over the active datasource. Disconnecting the Navigator tree from the Select Datasource combo You can suppress linking of the Datasource Navigator tree to the Select Datasource combo box. A clickable Link button appears at the top of the Navigator pane:



How the currently selected datasource changes depends on the state of the Link button

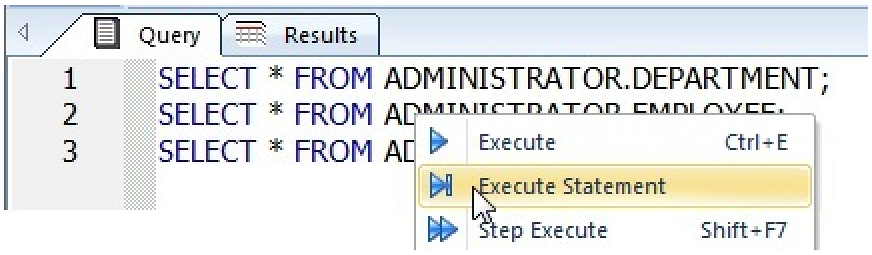
The Explorer is linked to the **Select Datasource** combo	You can explicitly change the active datasource by selecting a datasource from the **Select Datasource** dropdown. The active datasource changes automatically as datasource-associated windows, such as object editors, become active in the Workspace or as you select nodes in the Explorer. As a side effect, the associated datasource node in the Explorer tree is selected.
The Explorer is NOT linked to the **Select Datasource** combo	When linking is inactive, you can only explicitly change the active datasource when an SQL Editor window is open. The active datasource still changes automatically as datasource-associated windows, such as object editors, become active in the Workspace. And when linking is inactive, the change in the selected datasource has no effect on the selected node in the Explorer tree.

Locked/unlocked Login and User Icons Depending on the DBMS platform, the icons associated with logins and users differ according to the account's locked or unlocked state, when viewed in the Explorer

Object/Account Type	DBMS Platform	Icon State
Login	SQL Server or Sybase ASE	Locked
		Unlocked
User	Oracle or Sybase IQ	Locked
		Unlocked

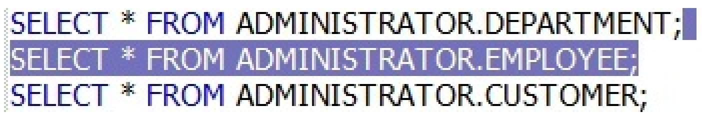
SQL Editor Updates

The following topics describe updated support for the Rapid SQL SQL Editor environment. Executing the statement at the cursor You can now execute the statement where the cursor is placed. Rapid SQL parses SQL and DDL statements according to the position of valid statement delimiters. When you select the new Execute Statement option, Rapid SQL executes the current statement.



```
SELECT * FROM ADMINISTRATOR.DEPARTMENT;
SELECT * FROM AD
SELECT * FROM AD
```

If the statement returns data, a Results tab opens. As a visual aid, Rapid SQL highlights the entire statement that was executed.

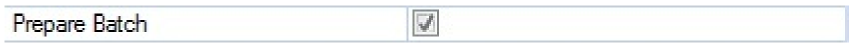


```
SELECT * FROM ADMINISTRATOR.DEPARTMENT;
SELECT * FROM ADMINISTRATOR.EMPLOYEE;
SELECT * FROM ADMINISTRATOR.CUSTOMER;
```

Bind Variable Parameterization in Prepared Statements As a step in improving performance, Rapid SQL now lets you provide bind variable values when executing scripts containing prepared statements. Bind variables, or parameterized literals, can help optimize explain/execution plan usage by minimizing the requirement to "hard parse" queries or statements that differ by one or more literal values. In scripts, Rapid SQL recognizes named or unnamed variables that use the following notation:

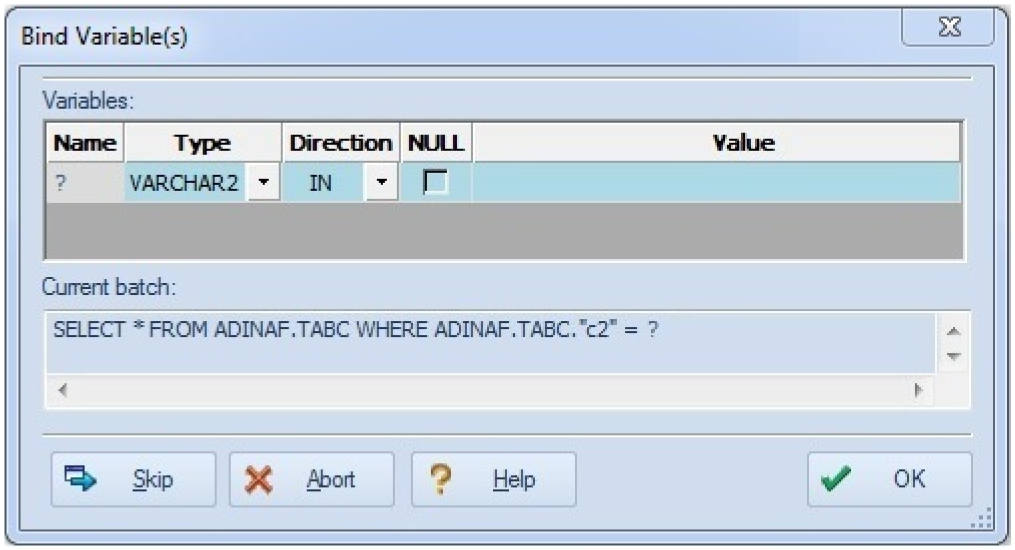
DBMS Platform/Driver Type	Notation
Oracle with native driver	<name> or :<number>
Oracle with JDBC driver	?
All other platforms/drivers	?

For example: SELECT * FROM MYTABLE WHERE MYCOLUMN = ? CALL MYPROC (:a, :b, :c) Support for bind variables is enabled on an editor window-by-editor window basis. When setting Query Options, each platform offers a new Prepare Batch setting.



Prepare Batch	✓

For an enabled SQL Editor window, when using execution options, whenever a statement containing bind variable notation is encountered, a Bind Variables dialog opens.



Bind Variable(s)

Variables:

Name	Type	Direction	NULL	Value
?	VARCHAR2	IN	☐	

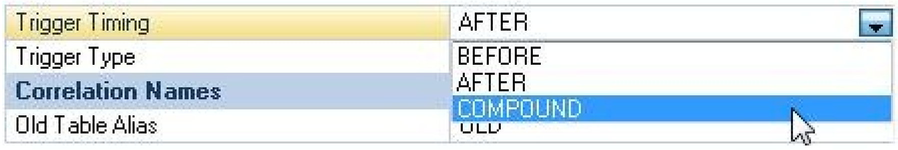
Current batch:

```
SELECT * FROM ADINAF.TABC WHERE ADINAF.TABC."c2" = ?
```

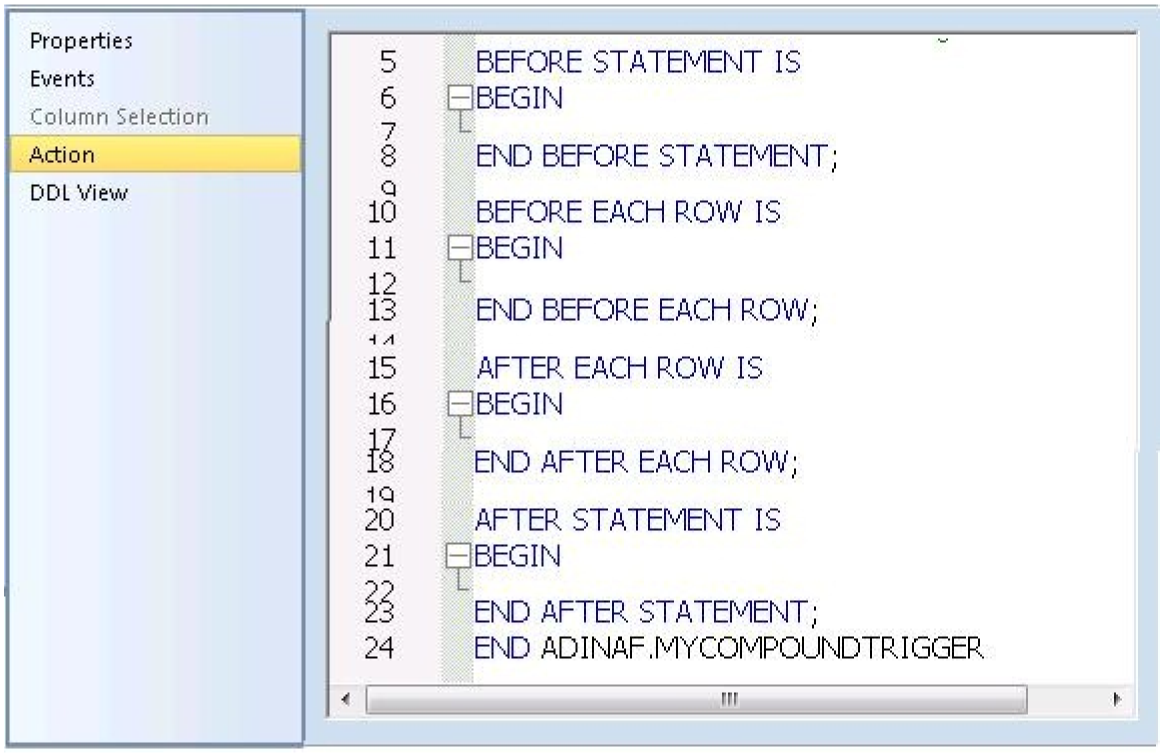
For each bind variable in the current statement, the dialog lets you specify: A type An IN, OUT, or INOUT designation for elements such as function or procedure parameters or arguments, variables used in INTO clauses of an INSERT statement, and variables used in RETURNING clauses of an Update statement. A value You can then execute, skip the statement, or abort execution for the currently running script.

Oracle Feature Support

The following topics describe updated support for Oracle features. Compound Trigger Support The Oracle Trigger Wizard's Trigger Timing property now offers a COMPOUND option, letting you create compound triggers.

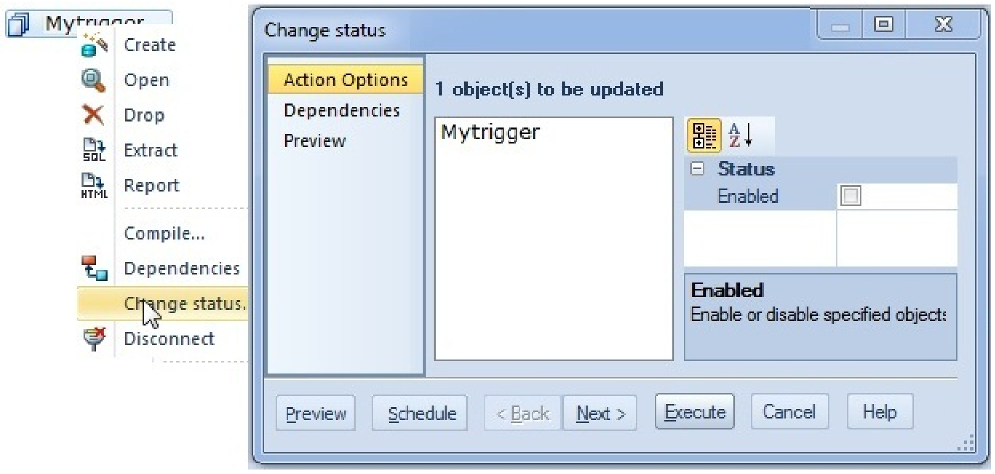


As an aid, when you select this option, the Action tab is automatically populated with a compound trigger block template.



```
BEFORE STATEMENT IS
BEGIN

END BEFORE STATEMENT;

BEFORE EACH ROW IS
BEGIN

END BEFORE EACH ROW;

AFTER EACH ROW IS
BEGIN

END AFTER EACH ROW;

AFTER STATEMENT IS
BEGIN

END AFTER STATEMENT;
END ADINAF.MYCOMPOUNDTRIGGER
```

The CREATE OR REPLACE TRIGGER... DDL generated for the trigger built using these options includes the compound trigger-specific FOR clause. Enable/Disable Trigger Support A new Change Status object action generates ALTER TRIGGER... DISABLE/ENABLE DDL, letting you enable or disable one or more Oracle triggers.



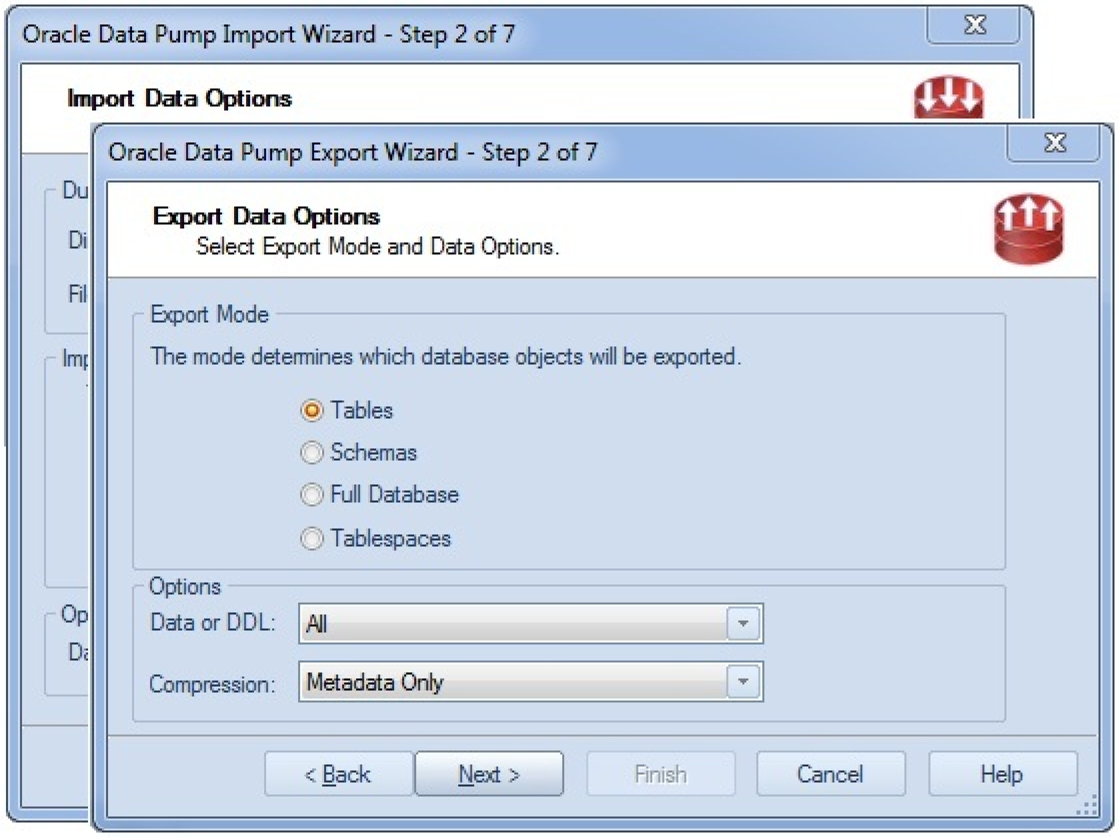
1 object(s) to be updated

Mytrigger

Status
Enabled ☐

Enabled
Enable or disable specified objects

Data Pump Import/Export support You can now make calls to the expdp.exe and impdp.exe utilities from Rapid SQL.


Oracle Data Pump Import Wizard - Step 2 of 7
Import Data Options
Oracle Data Pump Export Wizard - Step 2 of 7
< Back
Next >
Finish
Cancel
Help

Export Data Options
Select Export Mode and Data Options.

Export Mode
The mode determines which database objects will be exported.

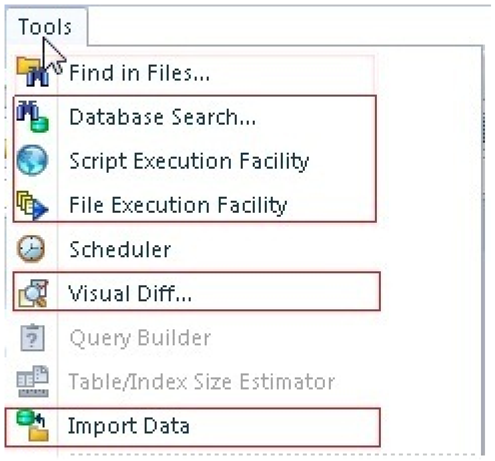
[✓ Tables] Schemas Full Database Tablespaces

Options
Data or DDL: All
Compression: Metadata Only

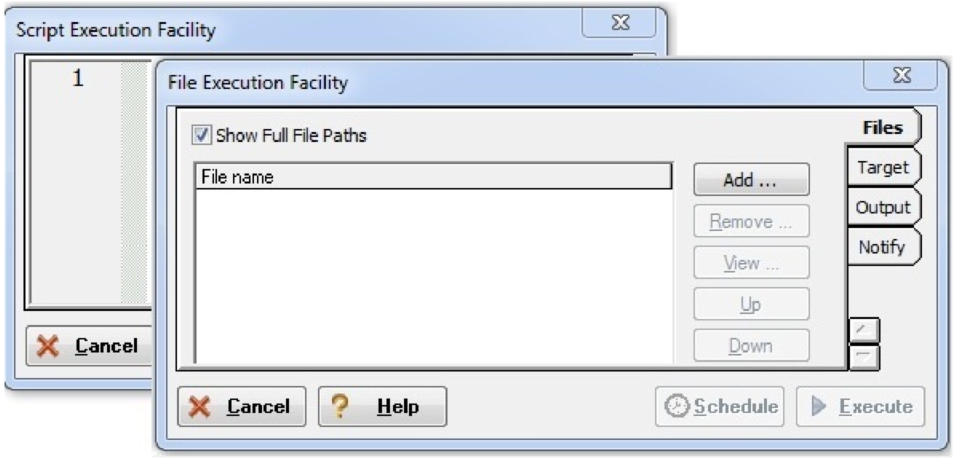
The wizard lets you specify a full set of parameters. In addition, you can execute the utilities from the wizard or using a command line call specifying an automatically-generated parameters file.

Sybase IQ Support

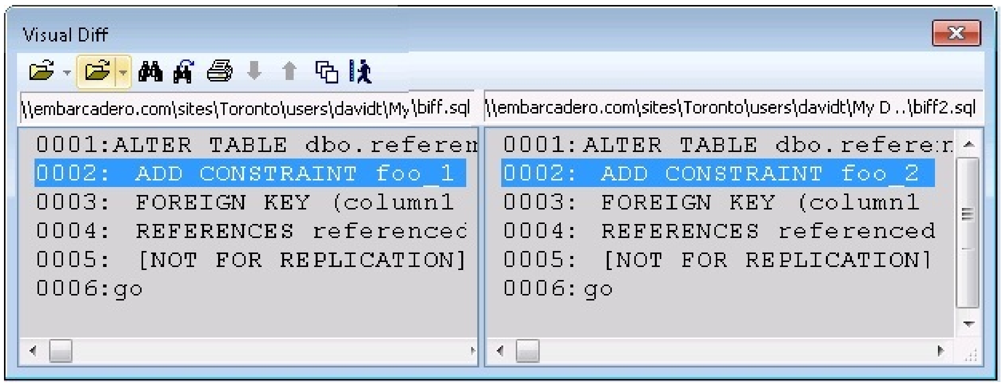
The following topics describe updated support for Sybase IQ features. Schema Extraction/Migration The Schema Extraction and Schema Migration Wizards are now available against Sybase IQ datasources. All features common to extraction/migration on other platforms are available for the new Sybase IQ Wizards. Additional Tools Menu Support Some Tools menu commands, formerly only available on other DBMS platforms, are now available against Sybase IQ datasources.



The Script Execution Facility and File Execution Facility commands are now available for the Sybase IQ platform. Functionality is the same as for other supported DBMS platforms.



You can now use the Database Search command to search for database objects containing a specified text string. Similarly, you can use the Visual Diff... command to compare files.



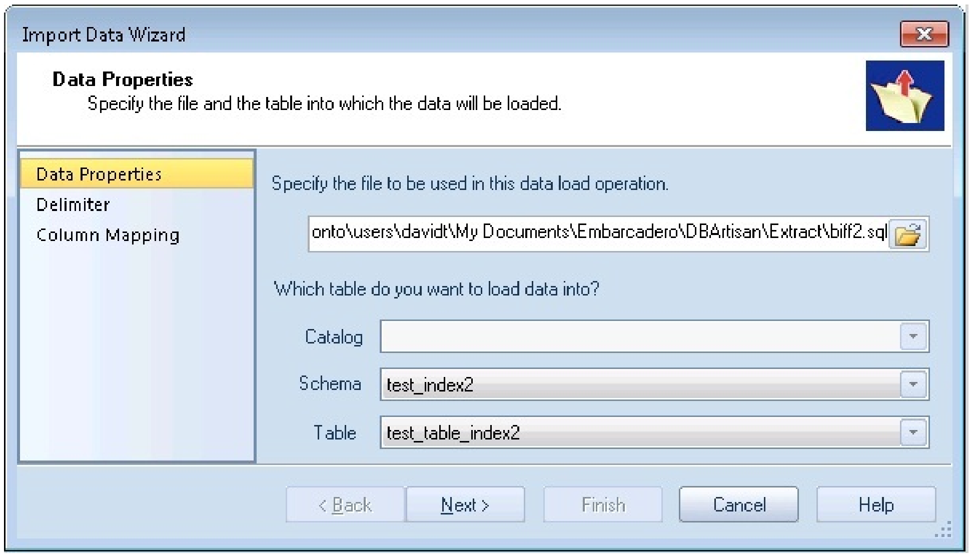
\\embarcadero.com\sites\Toronto\users\davidt\My\biff.sql

```
0001:ALTER TABLE dbo.referen
0002:  ADD CONSTRAINT foo_1
0003:  FOREIGN KEY (column1
0004:  REFERENCES referencec
0005:  [NOT FOR REPLICATION]
0006:go
```

\\embarcadero.com\sites\Toronto\users\davidt\My D..\biff2.sql

```
0001:ALTER TABLE dbo.referen
0002:  ADD CONSTRAINT foo_2
0003:  FOREIGN KEY (column1
0004:  REFERENCES referenced
0005:  [NOT FOR REPLICATION]
0006:go
```

Lastly, the Import Data command lets you load tables from commonly-delimited files.



Data Properties
Specify the file and the table into which the data will be loaded.

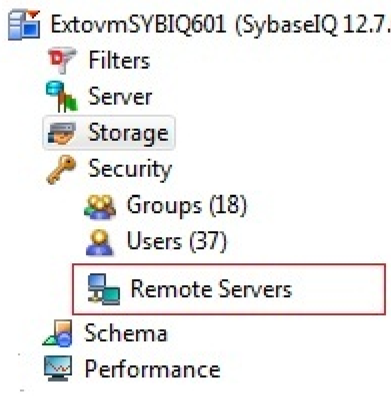
Specify the file to be used in this data load operation.

onto\users\davidt\My Documents\Embarcadero\DBArtisan\Extract\biff2.sql

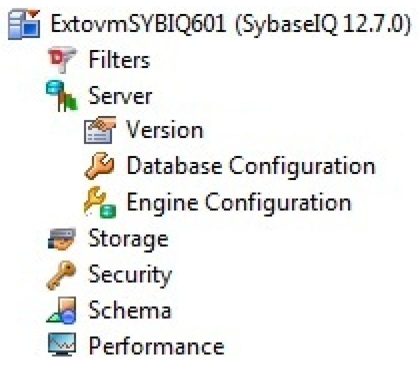
Which table do you want to load data into?

Catalog	
Schema	test_index2
Table	test_table_index2

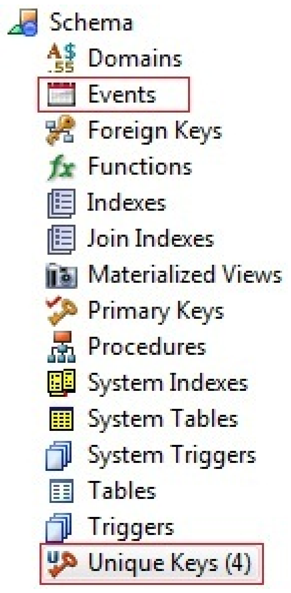
Remote Server Object Types Remote Server objects are now available under the Security node for Sybase IQ datasources.



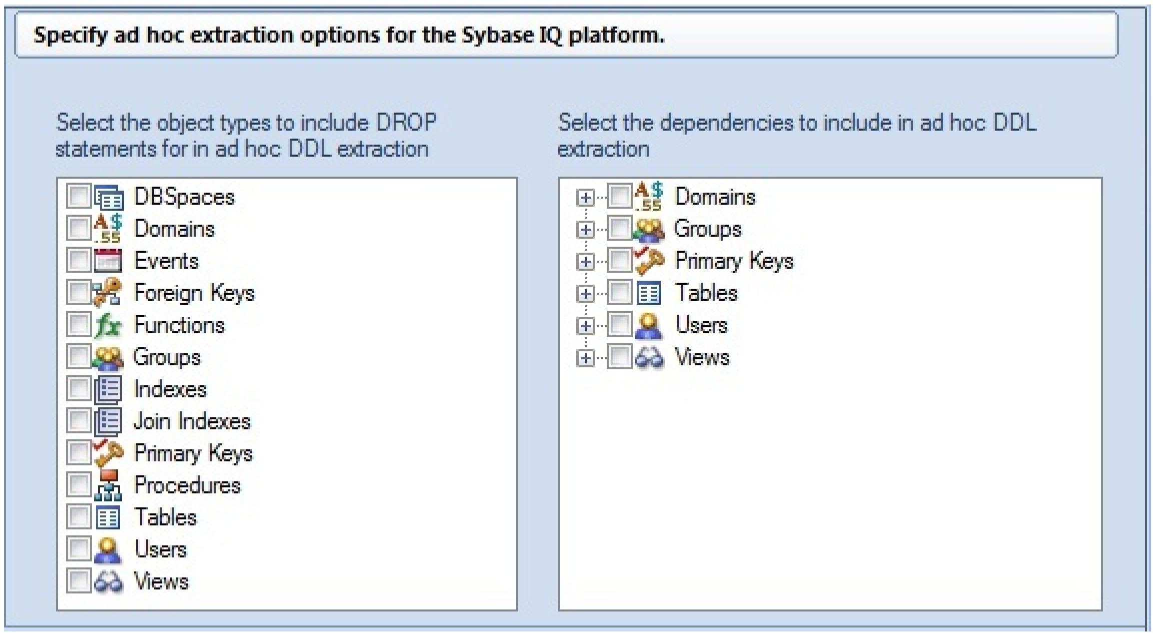
As with other Sybase IQ object types, Drop and Extract object actions are available. Server Node Parameters The Datasource Explorer now offers Version, Database Configuration, and Engine Configuration nodes for the Sybase IQ DBMS platform.



The Server subnodes provide access to server, database, and connection properties as well as configuration/initialization parameters. Events and Unique Keys Object Types The Datasource Explorer now offers Sybase IQ Events and Unique Keys object types.



A basic set of Drop, Extract, and Migrate actions are available for Unique Key objects. Those actions, as well as an Object Properties action, are supported against Events. A basic set of Drop and Extract actions are available for Unique Key objects. Those actions, as well as an Object Properties action, are supported against Events. DDL Extract Options You now have more control over Extract operations. The Options Editor now includes a DDL Extract > Sybase IQ panel.



Specify ad hoc extraction options for the Sybase IQ platform.

Select the object types to include DROP statements for in ad hoc DDL extraction

- [ ] DBSpaces
- [ ] Domains
- [ ] Events
- [ ] Foreign Keys
- [ ] Functions
- [ ] Groups
- [ ] Indexes
- [ ] Join Indexes
- [ ] Primary Keys
- [ ] Procedures
- [ ] Tables
- [ ] Users
- [ ] Views

Select the dependencies to include in ad hoc DDL extraction

- [ ] Domains
- [ ] Groups
- [ ] Primary Keys
- [ ] Tables
- [ ] Users
- [ ] Views

As for other platforms, this panel lets you specify: The object types to include DROP statements for The default dependent object types for each object type included in extraction/migration operations

SQL Server Feature Support

The following topics describe updated support for SQL Server features. User Permissions Updates When creating or editing users, the System Permissions tab now offers permissions introduced since SQL Server 2005. Newly available system permissions for users include:

ALTER ANY APPLICATION ROLE	ALTER ANY ASSEMBLY	ALTER ANY ASYMMETRIC KEY
ALTER ANY CERTIFICATE	ALTER ANY CONTRACT	ALTER ANY DATABASE AUDIT
ALTER ANY DATABASE DDL TRIGGER	ALTER ANY DATABASE EVENT NOTIFICATION	ALTER ANY DATASPACE
ALTER ANY FULLTEXT CATALOG	ALTER ANY MESSAGE TYPE	ALTER ANY REMOTE SERVICE BINDING
ALTER ANY ROLE	ALTER ANY ROUTE	ALTER ANY SCHEMA
ALTER ANY SERVICE	ALTER ANY SYMMETRIC KEY	ALTER ANY USER
AUTHENTICATE	CHECKPOINT	CONNECT
CONNECT REPLICATION	CREATE AGGREGATE	CREATE ASSEMBLY
CREATE CONTRACT	CREATE DATABASE DDL NOTIFICATION	CREATE FULLTEXT CATALOG
CREATE MESSAGE TYPE	CREATE QUEUE	CREATE REMOTE SERVICE BINDING
CREATE ROLE	CREATE ROUTE	CREATE SCHEMA
CREATE SERVICE	CREATE TYPE	CREATE XML SCHEMA COLLECTION
DELETE	EXECUTE	INSERT
REFERENCES	SELECT	SHOWPLAN
SUBSCRIBE QUERY NOTIFICATIONS	TAKE OWNERSHIP	UPDATE
VIEW DATABASE STATE		

Similarly, the User Wizard/Editor’s Object Permissions tab lets you provide Grant/Revoke/Deny permissions on the following object types, previously unavailable with this functionality in Rapid SQL

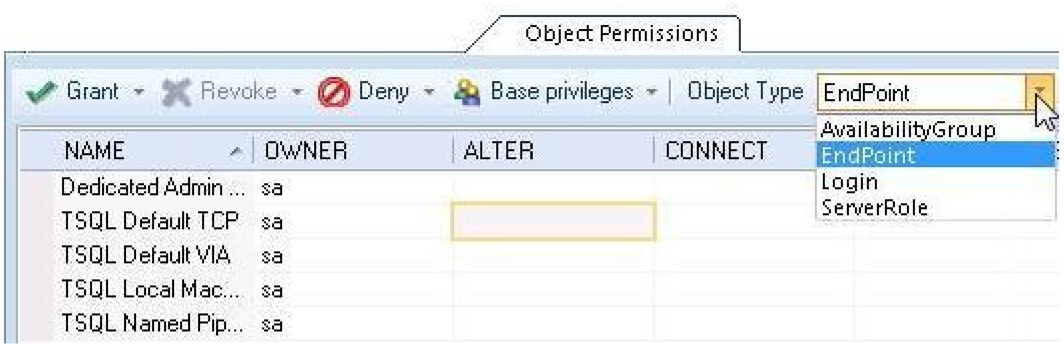
ApplicationRole	Assembly	Contract	FullTextCatalog
FullTextStopList	MessageType	Queue	RemoteServiceBinding
Role	Route	SearchPropertyList	Sequence
Service	User	UserDefinedDatatype	XMLSchemaCollection

Login Permission Updates After creating a login, the SQL Server Login Editor now lets you assign permissions. While specific permissions differ by DBMS version, the Login

Editor’s System Permissions tab now lets you assign the following permissions, previously unavailable with Rapid SQL

ADMINISTER BULK OPERATIONS	ALTER ANY AVAILABILITY GROUP	ALTER ANY CONNECTION
ALTER ANY CREDENTIAL	ALTER ANY DATABASE	ALTER ANY ENDPOINT
ALTER ANY EVENT NOTIFICATION	ALTER ANY EVENT SESSION	ALTER ANY LINKED SERVER
ALTER ANY LOGIN	ALTER ANY SERVER AUDIT	ALTER ANY SERVER ROLE
ALTER RESOURCES	ALTER SERVER STATE	ALTER SETTINGS
ALTER TRACE	AUTHENTICATE SERVER	CONNECT SQL
CONTROL SERVER	CREATE ANY DATABASE	CREATE AVAILABILITY GROUP
CREATE DDL EVENT NOTIFICATION	CREATE ENDPOINT	CREATE SERVER ROLE
CREATE TRACE EVENT NOTIFICATION	EXTERNAL ACCESS ASSEMBLY	SHUTDOWN
UNSAFE ASSEMBLY	VIEW ANY DATABASE	VIEW ANY DEFINITION
VIEW SERVER STATE		

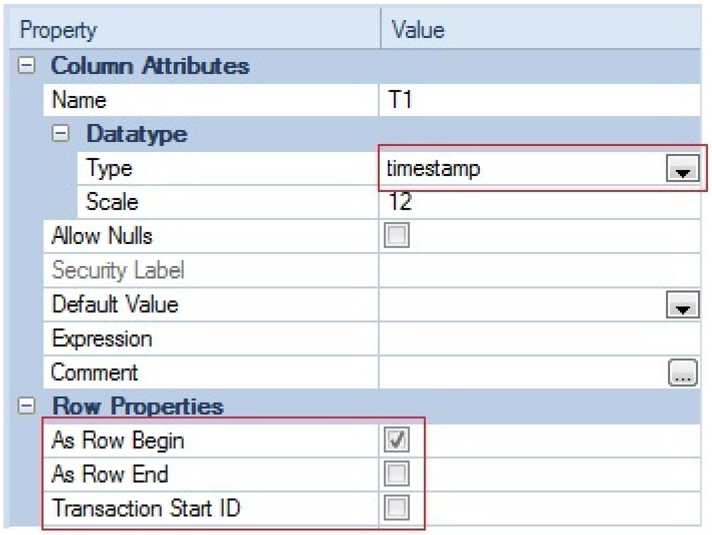
Similarly, the new Object Permissions tab lets you assign Grant/Revoke/Deny permissions to EndPoints, Availability Groups, Logins, and Server Roles.



NAME	OWNER	ALTER	CONNECT
Dedicated Admin ...	sa		
TSQL Default TCP	sa		
TSQL Default VIA	sa		
TSQL Local Mac...	sa		
TSQL Named Pip...	sa		

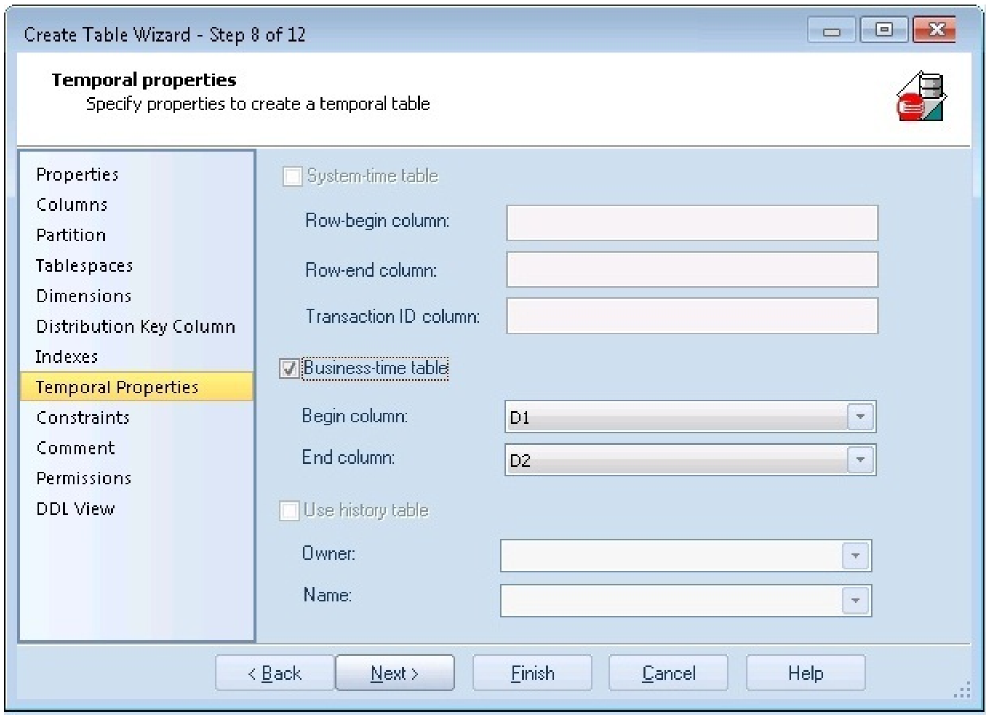
DB2 LUW and DB2 z/OS Feature Support

Temporal Tables (version 10) The Tables Wizard now lets you include attributes that indicate when data in a row is current and when transactions affected the data. Tables can be designated as system-period or application-period tables. The Columns panel now includes settings that let you properly specify row-begin, row-end, and transaction start ID columns. The new Row Properties settings are available for columns assigned a Type of TIMESTAMP.



Property	Value
Column Attributes	
Name	T1
Datatype	
Type	timestamp
Scale	12
Allow Nulls	☐
Security Label	
Default Value	
Expression	
Comment	
Row Properties	
As Row Begin	✓
As Row End	☐
Transaction Start ID	☐

A new Temporal Properties panel lets you: Designate a table as a system-time table or business-time table Select the columns used to define the relevant period Provide additional temporal tables implementation details



Create Table Wizard - Step 8 of 12

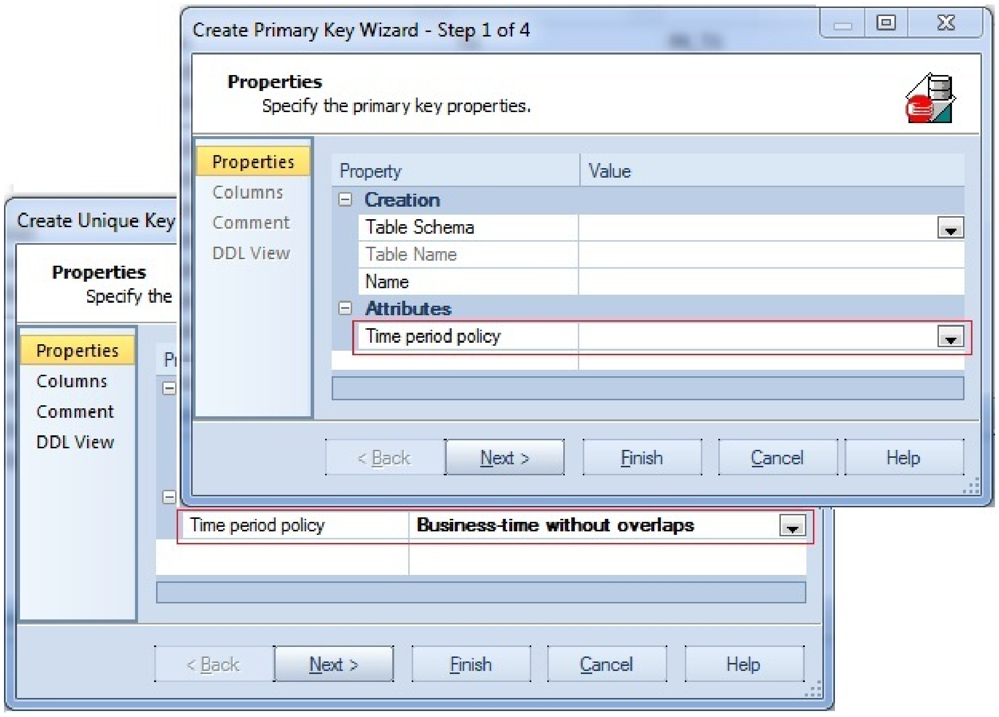
Temporal properties
Specify properties to create a temporal table

Properties
Columns
Partition
Tablespaces
Dimensions
Distribution Key Column
Indexes
Temporal Properties
Constraints
Comment
Permissions
DDL View

☐ System-time table
Row-begin column:
Row-end column:
Transaction ID column:
☑ Business-time table
Begin column: D1
End column: D2
☐ Use history table
Owner:
Name:

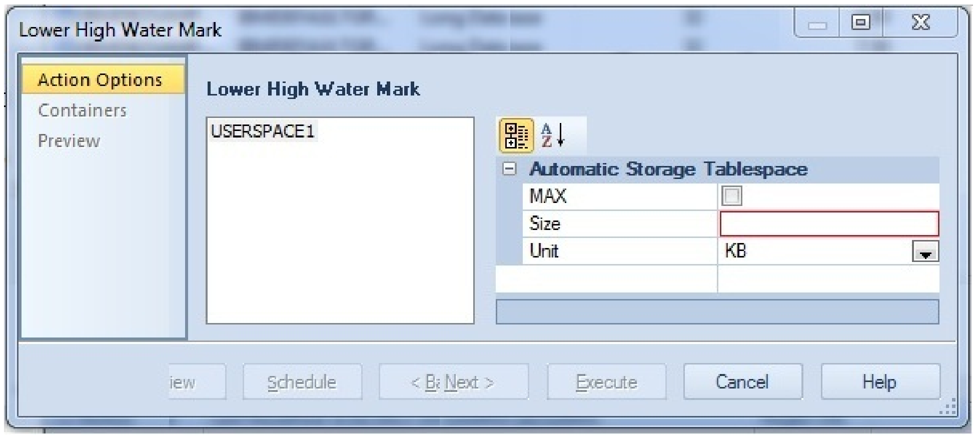
< Back | Next > | Finish | Cancel | Help

In addition, the Primary Key Wizard and Unique Key Wizard have been updated to facilitate temporal tables. A new Time period policy property lets you create a constraint on a business-time table that can enforce uniqueness of the data for any point in business time.

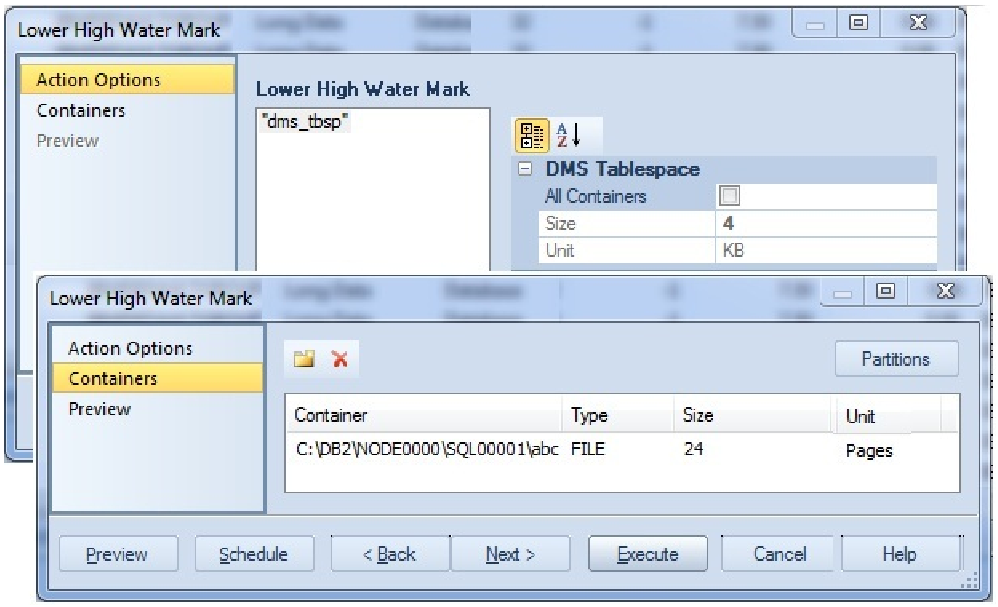


DB2 LUW Feature Support

The following topics describe updated support for IBM DB2 for Linux, Unix, and Windows features. Command Line Processor Export Support Against 9.7 datasources, an Export utility is now available. It lets you export data to one of several external file formats. The CLP Export Wizard lets you: Select an output file format (IXF, DEL, or WSF) Specify a set of modifiers relevant to each format Provide LOB and XML handling details Select a SELECT/XQUERY, default traversal, or user-specified traversal export The Wizard generates two db2.exe calls: one to connect to the database and one to issue the export directive. Lowering Tablespace High Water Mark A new Lower High Water Mark operation lets you reduce the size of existing containers. Against automatically-managed tablespaces, this action lets you build and submit an ALTER TABLESPACE... REDUCE statement, specifying that existing containers be reduced by a specified size.



Against database-managed tablespaces, the Lower High Water Mark operation builds and submits an ALTER TABLESPACE... LOWER HIGH WATER MARK statement followed by an ALTER TABLESPACE... REDUCE statement, explicitly lowering the high water mark before reducing the size of the tablespace.

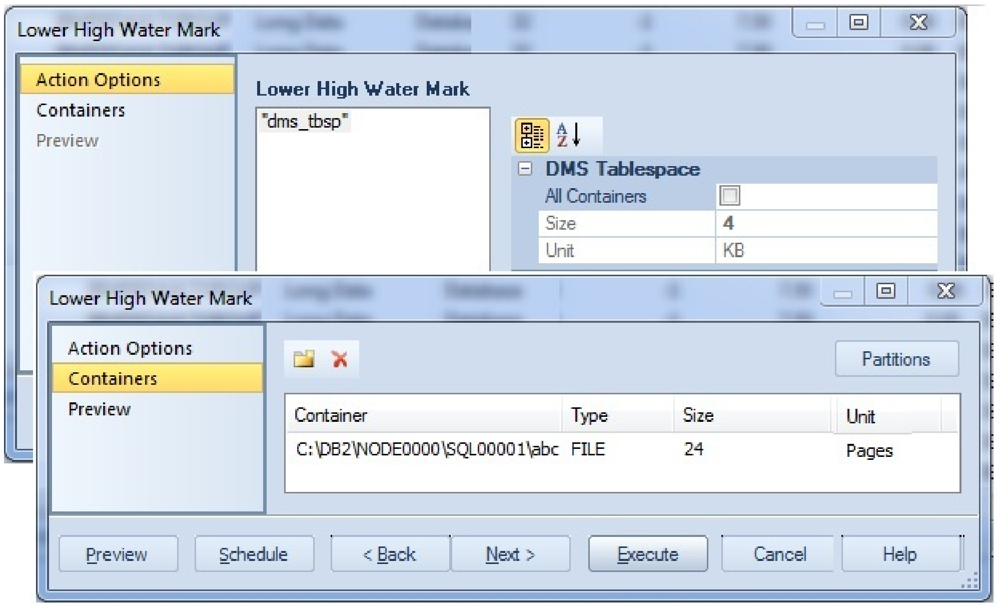


"dms_tbsp"

DMS Tablespace	
All Containers	☐
Size	4
Unit	KB

Container	Type	Size	Unit
C:\DB2\NODE0000\SQL00001\abc	FILE	24	Pages

Moving Table Data A new Move Table object action builds and submits a SYSPROC.ADMIN_MOVE_TABLE call, letting you move data in an active table into a new, identically-named table, leaving the data online and available for use. This action offers two methods: You can create the table manually and use the action to move data to your newly created table. You can have the action create the table, specifying tablespaces used, and have the data moved as part of the operation. Storage Path and Container-based Tablespace Actions Two new tablespace actions are now available.


"dms_tbsp"

DMS Tablespace	
All Containers	☐
Size	4
Unit	KB

Container	Type	Size	Unit
C:\DB2\NODE0000\SQL00001\abc	FILE	24	Pages

Rebalance lets you create containers on recently added storage paths and drop containers from storage paths that are in DROP PENDING state. Drop Automatic Storage Paths removes one or more storage paths with an optional REBALANCE operation. Index Compression The Index Wizard and Index Editor now include a Compress property. This controls whether a COMPRESS clause is added to the generated CREATE/ALTER INDEX statement, enabling or disabling index compression.

Getting Started

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

- Prerequisites and Preliminary Tasks: Provides information on installation and licensing as well as general information on using this document.
- Starting Rapid SQL: Describes the various ways to start the application.
- Rapid SQL Application Basics: Provides detailed tutorial exercises that introduce the basic areas of Rapid SQL functionality.
- Rapid SQL Tutorial exercises: Tutorial exercises that show the major functional areas of Rapid SQL.
- Specifying Application Preferences and Feature Options: Tells you how to configure Rapid SQL.

Starting Rapid SQL

There are three ways to start the application:

- Starting an Installed version of Rapid SQL from the Start Menu
- Starting an Installed version of Rapid SQL from the Command Line

Starting an Installed version of Rapid SQL from the Start Menu

The **Start** menu sequences for Rapid SQL startup are always in the form

- Programs > Embarcadero Rapid SQL*version identifier*> Embarcadero Rapid SQL*version identifier*

where *version identifier* reflects the version you are running.

Starting an Installed version of Rapid SQL from the Command Line

Command line startup is primarily aimed at identity management systems working in conjunction with Rapid SQL. Such systems can be configured to start Rapid SQL and connect to a specific datasource using credentials that are invisible to the user. Other applications that hide credentials, force connection to a specific datasource, administer usage audits, or otherwise automate startup of Rapid SQL can make use of this feature.

Two forms of command line startup are available for each product. Syntax and parameters are as follows:

Rapid SQL

- rsql.exe -D*datasource*-U*username*-P*password*
- rsql.exe -R*connectionstring*-D*datasource*-U*username* [-P*password*]

For example

- rsql.exe -D torlabsy01-Umyusername-P mypassword
- dbart.exe -R "oracle://(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=TORLABORCL10g_2)(PORT=1521))(CONNECT_DATA=(SID=ORASID)))" -D torlabora10g_2 -U sys -P extreme

Note: For a listing of valid connection string values, see -R Switch Values by Platform.

Keep the following points in mind when using command line startup:

- Credentials remain valid until the user disconnects from the datasource.
- When you try disconnect, a confirmation message box includes a warning message that the connection has external credentials that will be lost if you choose to proceed with disconnect.
- Once disconnected, a user must connect to the previously disconnected datasource using specified, valid credentials or rely on Auti-connect. For more information, see Manually Registering or Editing Datasources.
- Command line credentials override the default credentials provided if a datasource is registered using the Auto-connect setting.

Regarding the switch specifications:

- The password switch/value pair is optional and the **P** switch can be provided without a value.
- The switches are case-sensitive while the datasource is case-insensitive;

- The order of these switches doesn't matter;
- Switches can be specified with the minus sign or with a slash and any combinations are allowed. For example, DBArt -D ROMLABSQL08 /U sa /P.
- Switches can be specified with the minus sign or with a slash and any combinations are allowed. For example, RSQL.-D ROMLABSQL08 /U sa /P.
- If you specify a switch multiple times, it's last valid occurrence from left to right is used. We auto-connect a single datasource, no matter the number of times a -D switch is specified;
- The -p switch (with lower case) prints a file;
- If -R is specified then one of the following will occur:
- If -D is also specified, it will check to see if that datasource already exists. If it does exist, it will check to see if its connection information matches the provided -R parameter. If it matches, startup behavior will be the same as if -R was not specified. If it doesn't match, a modified form of the datasource name is generated using the -D-provided name to ensure uniqueness and give it the connection information specified by -R. If the datasource doesn't exist, a new datasource is generated using the name provided by -D and the remaining connection information provided by -R.
- If -D is not specified, the -R-provided datasource is checked for a match against existing datasources. If a match is found, ithat datasource is connected. If no match is found, a new datasource is generated using the -R-provided information to create a unique name.

Note: If connecting to an existing datasource with auto-connect enabled, if -U and -P values were provided, then auto-connect will be disabled and the provided -U and -P values are used.

-R Switch Values by Platform

The following table lists the valid connection strings that can be used with the -R switch for a command line startup.

Platform	Connect String	Notes
DB2 LUW and DB2 z/OS	db2://<HOST>/<DB> db2://<HOST>:<PORT>/<DB> db2:@<ALIAS> where ALIAS is the name from the DB2 client registration	If the DB2 ODBC/CLI driver is installed on the system, that driver will be used for DB2 connections. Otherwise the IBM Data Server Driver for JDBC will be used. Using "db2p" will make the datasource created permanent
MySQL	mysql://<HOST> mysql://<HOST>/<DB> mysql://<HOST>:<PORT> mysql://<HOST>:<PORT>/<DB>	If the MySQL ODBC driver is installed on the system, that driver will be used to make the connection. Otherwise, the MySQL Connector/J JDBC driver will be used. Using "mysqlp" will make the datasource created permanent

ODBC	odbc://<CONNECT_STRING> where connect string can be the a System DSN name on it's own or the full ODBC connection string containing at least one of the following tags: DSN=, FILEDSN= or DRIVER=	Using "odbcp" will make the datasource created permanent.
ORCL	oracle://<CONNECT_DESCRIPTOR> oracle:@<ALIAS> where CONNECT_DESCRIPTOR is in the same form as in the TNSNames.ora file. i.e. (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=<HOST>)(PORT=<PORT>))(CONNECT_DATA=(SID=<SID>))) where ALIAS is the name from the TNSNAMES.ora file	If OCI is installed on the system, that driver will be used. Otherwise, the Oracle Instant Client will be used. Using "oraclep" will make the datasource created permanent.

PSTGRS *	postgresql://<HOST> postgresql://<HOST>/<DB> postgresql://<HOST>:<PORT> postgresql://<HOST>:<PORT>/<DB>	If <HOST> is a IPv6 address, it must be enclosed in square brackets. If the PostgresSQL ODBC driver is installed on the system, that driver will be used., Otherwise, the PostgreSQL JDBC driver. will be used Using "postgresqlp" will make the datasource created permanent
SQL SVR	sqlserver://<HOST> sqlserver://<HOST>/<DB> sqlserver://<HOST>:<PORT> sqlserver://<HOST>:<PORT>/<DB> sqlserver://<HOST>\<INSTANCE> sqlserver://<HOST>\<INSTANCE>/<DB> sqlserver://<HOST>\<INSTANCE>:<PORT> sqlserver://<HOST>\<INSTANCE>:<PORT>/<DB> sqlserver:@<ALIAS> sqlserver:@<ALIAS>/<DB> where ALIAS is the name from the Network Library Configuration	The latest Microsoft ODBC driver for SQL Server installed on the system is used to make the connection. Using "sqlserverp" will make the datasource created permanent.

SQL SVRAzure	sqlserverazure://<HOST> sqlserverazure://<HOST>/<DB>	The latest Microsoft ODBC driver for SQL Server installed on the system is used to make the connection. Using "sqlserverazurep" will make the datasource created permanent
SYBASE	sybase://<HOST> sybase://<HOST>/<DB> sybase://<HOST>:<PORT> sybase://<HOST>:<PORT>/<DB> sybase:@<ALIAS> sybase:@<ALIAS>/<DB> where ALIAS is the name from the SQL.ini file	If CTLIb is installed on the system that driver will be used. Otherwise the Sybase jConnect JDBC driver will be used. Using "sybasep" will make the datasource created permanent

SYB IQ	sybaseiq://<HOST> sybaseiq://<HOST>/<DB> sybaseiq://<HOST>:<PORT> sybaseiq://<HOST>:<PORT>/<DB>	If the Sybase IQ ODBC driver is installed on the system it will use that, otherwise it will use the Sybase jConnect JDBC driver. Using "sybaseiqp" will make the datasource created permanent

The following table lists the valid connection strings that can be used with the **-R** switch for a command line startup.

Platform	Connect String	Notes
DB2 LUW and DB2 z/OS	db2://<HOST>/<DB> db2://<HOST>:<PORT>/<DB> db2:@<ALIAS> where ALIAS is the name from the DB2 client registration	If the DB2 ODBC/CLI driver is installed on the system, that driver will be used for DB2 connections. Otherwise the IBM Data Server Driver for JDBC will be used. Using "db2p" will make the datasource created permanent

ITB/FBD *	interbase://<HOST>/<DB> interbase://<HOST>:<PORT>/<DB> firebirdsql://<HOST>/<DB> firebirdsql://<HOST>:<PORT>/<DB>	If the InterBase ODBC driver is installed on the system, that driver will be used for an InterBase connection. Otherwise, the InterBase JDBC driver will be used. Using "interbasep" will make the datasource created permanent. If the Firebird ODBC driver is installed on the system, that driver will be used for Firebird connections. Otherwise, the Jaybird JDBC driver will be used. Using "firebirdsqlp" will make the datasource created permanent

MySQL	mysql://<HOST> mysql://<HOST>/<DB> mysql://<HOST>:<PORT> mysql://<HOST>:<PORT>/<DB>	If the MySQL ODBC driver is installed on the system, that driver will be used to make the connection. Otherwise, the MySQL Connector/J JDBC driver will be used. Using "mysqlp" will make the datasource created permanent
ODBC	odbc://<CONNECT_STRING> where connect string can be the a System DSN name on it's own or the full ODBC connection string containing at least one of the following tags: DSN=, FILEDSN= or DRIVER=	Using "odbcp" will make the datasource created permanent.
ORCL	oracle://<CONNECT_DESCRIPTOR> oracle:@<ALIAS> where CONNECT_DESCRIPTOR is in the same form as in the TNSNames.ora file. i.e. (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=<HOST>)(PORT=<PORT>))(CONNECT_DATA=(SID=<SID>))) where ALIAS is the name from the TNSNAMES.ora file	If OCI is installed on the system, that driver will be used. Otherwise, the Oracle Instant Client will be used. Using "oraclep" will make the datasource created permanent

PSTGRS *	postgresql://<HOST> postgresql://<HOST>/<DB> postgresql://<HOST>:<PORT> postgresql://<HOST>:<PORT>/<DB>	If <HOST> is a IPv6 address, it must be enclosed in square brackets. If the PostgresSQL ODBC driver is installed on the system, that driver will be used., Otherwise, the PostgreSQL JDBC driver. will be used Using "postgresqlp" will make the datasource created permanent
SQL SVR	sqlserver://<HOST> sqlserver://<HOST>/<DB> sqlserver://<HOST>:<PORT> sqlserver://<HOST>:<PORT>/<DB> sqlserver://<HOST>\<INSTANCE> sqlserver://<HOST>\<INSTANCE>/<DB> sqlserver://<HOST>\<INSTANCE>:<PORT> sqlserver://<HOST>\<INSTANCE>:<PORT>/<DB> sqlserver:@<ALIAS> sqlserver:@<ALIAS>/<DB> where ALIAS is the name from the Network Library Configuration	The latest Microsoft ODBC driver for SQL Server installed on the system is used to make the connection. Using "sqlserverp" will make the datasource created permanent

SQL SVRAzure	sqlserverazure://<HOST> sqlserverazure://<HOST>/<DB>	The latest Microsoft ODBC driver for SQL Server installed on the system is used to make the connection. Using "sqlserverazurep" will make the datasource created permanent
SYBASE	sybase://<HOST> sybase://<HOST>/<DB> sybase://<HOST>:<PORT> sybase://<HOST>:<PORT>/<DB> sybase:@<ALIAS> sybase:@<ALIAS>/<DB> where ALIAS is the name from the SQL.ini file	If CTLIb is installed on the system that driver will be used. Otherwise the Sybase jConnect JDBC driver will be used. Using "sybasep" will make the datasource created permanent

SYBIQ	sybaseiq://<HOST> sybaseiq://<HOST>/<DB> sybaseiq://<HOST>:<PORT> sybaseiq://<HOST>:<PORT>/<DB>	If the Sybase IQ ODBC driver is installed on the system it will use that, otherwise it will use the Sybase jConnect JDBC driver. Using "sybaseiqp" will make the datasource created permanent

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
Starting Rapid SQL	Describes the various ways to start the application.
Rapid SQL Product Design	This section describes the Rapid SQL user interface.
Specifying Application Preferences and Feature Options	This section describes how to customize Rapid SQL's configuration to suit your specific needs.

Rapid SQL Product Design

The Rapid SQL window opens with the Datasource Navigator 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.

These desktops and workspaces are:

- The Rapid SQL Datasource Navigator
- Rapid SQL Windows
- Rapid SQL Menus
- Rapid SQL Toolbars
- Customizing the Rapid SQL General User Interface Appearance
- Rapid SQL Keyboard Shortcuts
- Rapid SQL Full Screen Mode

The Rapid SQL Datasource Navigator

Rapid SQL organizes the wealth of information pertaining to your servers through its Datasource Navigator. The Datasource Navigator provides a fast and efficient way to access your database objects and scripts. The Datasource Navigator 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 Navigator window is dockable so that you can maneuver through the application efficiently.

For details on the Datasource Navigator and datasource management, in general, see the following topics:

- Datasource and Server Management
- Using the Datasource Navigator

Tip: You can also set these options on the Explorer tab of the Options Editor. For details, see Explorer Options - Main Tab.

Rapid SQL Windows

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

Topics

The Rapid SQL 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 Navigator Tab, select an object and then click Describe.
3. On the Navigator 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).

The Rapid SQL 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

1. 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

1. 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
- Column filtering in the Browser

- 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.

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. OpenOpening Browsers 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 Opening BrowsersBrowser 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. OpenOpening Browsers 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.

Column filtering in the Browser

Two methods of filtering are available when database objects are displayed in the Browser. Both make use of the text boxes appearing below column names in the display.

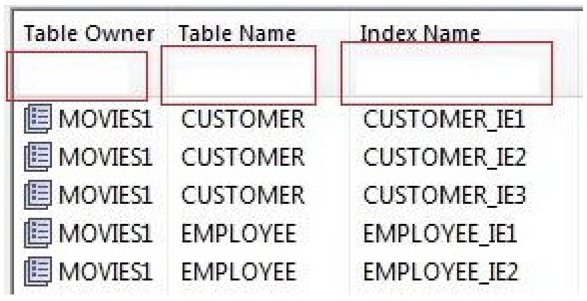


Table Owner	Table Name	Index Name
MOVIES1	CUSTOMER	CUSTOMER_IE1
MOVIES1	CUSTOMER	CUSTOMER_IE2
MOVIES1	CUSTOMER	CUSTOMER_IE3
MOVIES1	EMPLOYEE	EMPLOYEE_IE1
MOVIES1	EMPLOYEE	EMPLOYEE_IE2

To show only rows for which a given column value contains one or more contiguous characters

1. Type those characters into the text box below that column name.

To make use of regular expressions in column filtering

1. Ensure that regular expression usage is enabled. For details, see Explorer Options - Main Tab.

2. Type a valid regular expression search criteria into the text box below that column name. For more information, see Regular Expressions Support.

Note: The **RegEx ON/RegEx OFF** button lets you enable or disable regular expression usage on-the-fly for a given Navigator window.

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.
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.

Rapid SQL 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 the Rapid SQL 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 Navigator, 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.

Explorer Bookmarks

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

Creating Explorer Bookmarks

1. On the Database Navigator, 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.

Rapid SQL 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:

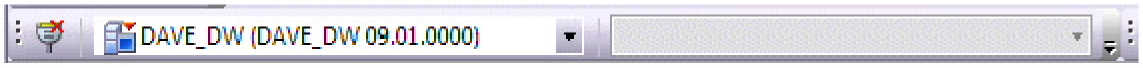
- Available Toolbars
- Using Toolbar Viewing Options
- Moving Toolbars

Note: For related information, see Customizing the Rapid SQL General User Interface Appearance.

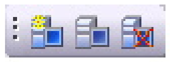
Available Toolbars

The following list represents Rapid SQL's toolbars:

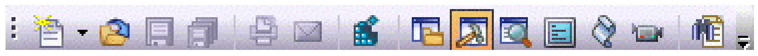
Datasource Toolbar



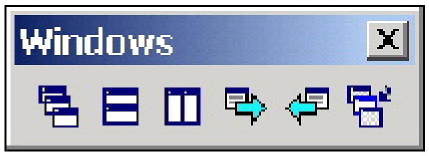
Registration Toolbar



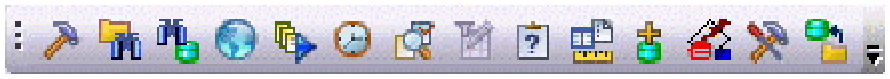
Main Toolbar



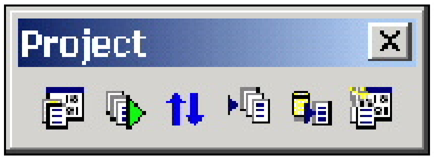
Windows Toolbar



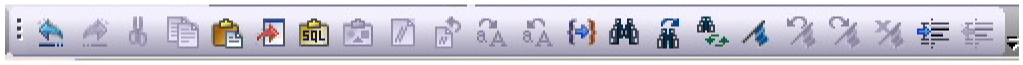
Tools Toolbar



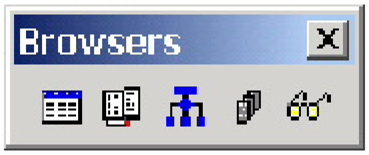
Project Toolbar



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 Available Toolbars.

Moving Toolbars

1. Click the pointer at the right edge of any toolbar.
2. Drag the toolbar to the new position.

Customizing the Rapid SQL 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. Click **Apply** at any time to implement any changes you have made and when finished, click **Close**. Use the following table as a guide for 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 [Available Toolbars]().	
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 [Specifying an Argument for a Tools Menu Command]().	
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).”

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 predefined arguments 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.

Rapid SQL 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

Rapid SQL 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 tool bar, 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.

Rapid SQL 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 and development tools provided.

This guide is divided into a set of sessions:

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

Note: The tutorial exercises make use of the sample SQL Server database, **AdventureWorks**. Since this is not part of a typical install, you can obtain the files from installation CDs or online sources to install and attach that database. Alternatively, you can perform the tasks targeted for the **AdventureWorks** database against another available database. No destructive actions are initiated in the exercises.

Session 1: Getting Started

Before anything else, you must perform the following tasks:

- Starting the Rapid SQL Application
- Registering Cross-Platform Datasources to Rapid SQL

Starting the Rapid SQL Application

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*> Rapid SQL*version identifier*, where *version identifier* reflects the version you are running.

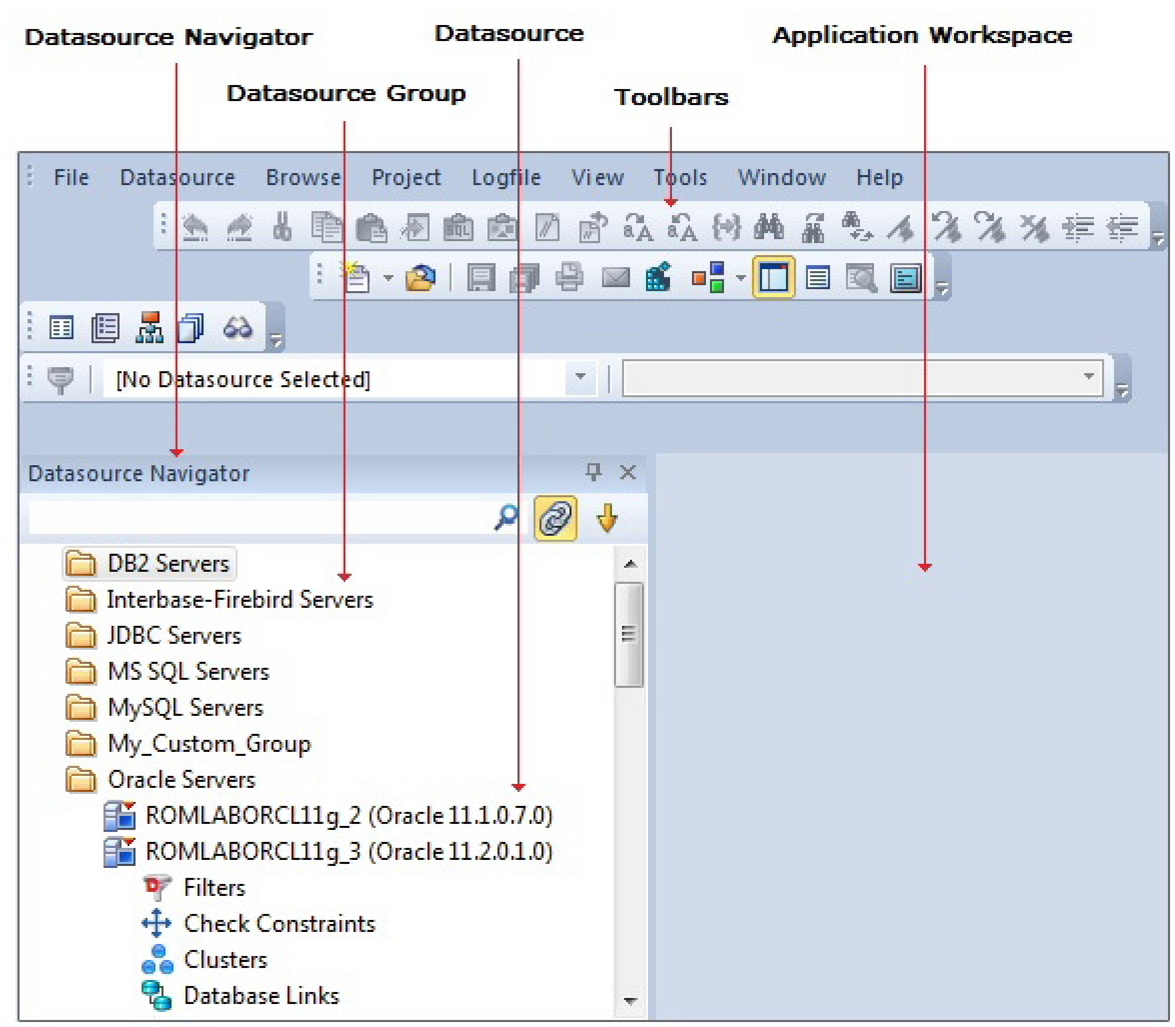
To get started

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. Also, 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.

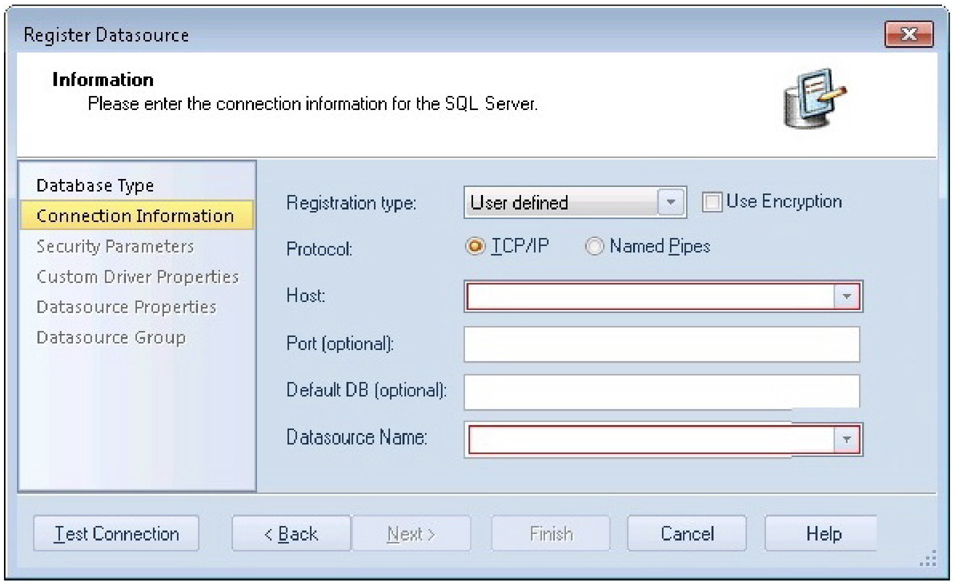


Proceed to Registering Cross-Platform Datasources to Rapid SQL.

Registering Cross-Platform Datasources to Rapid SQL

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**. The next panel opens.


Register Datasource

Information
Please enter the connection information for the SQL Server.

Registration type: User defined
Use Encryption
Protocol: TCP/IP [✓] Named Pipes
Host:
Port (optional):
Default DB (optional):
Datasource Name:

3. Ensure that a **Registration type** of **User defined** is selected, specify the **Host** name of an 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. Lastly, datasource definitions can be stored centrally on an Embarcadero Team Server 2016 for use by Rapid SQL users.

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

Proceed to Session 2: Productivity Enhancers.

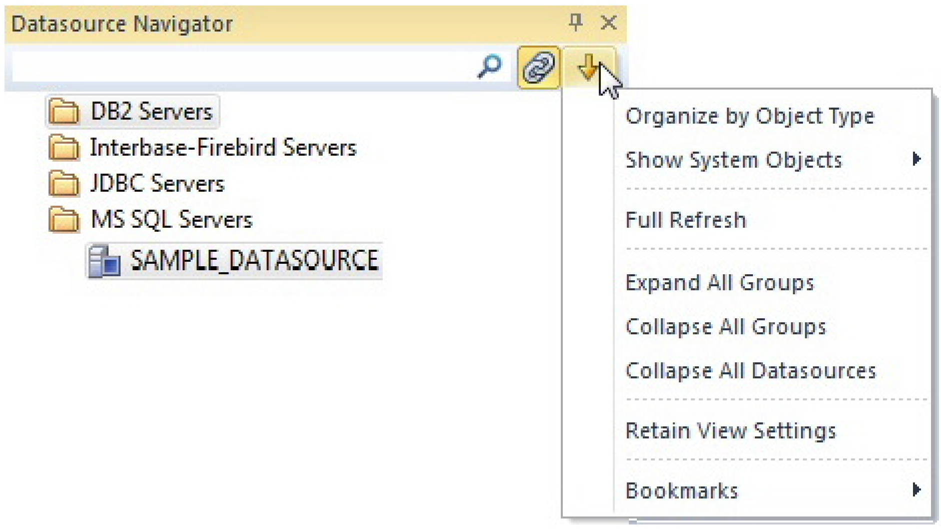
Session 2: Productivity Enhancers

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

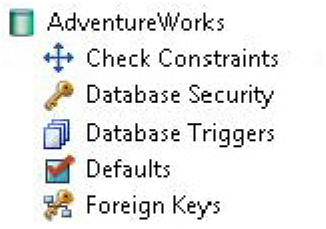
The Datasource Navigator Tree

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

1. Click on the Navigator's dropdown and ensure that **Organize By Object Type** is selected.



2. Select and expand the SAMPLE_DATASOURCE > Databases > AdventureWorks node to display the database object sub-nodes.

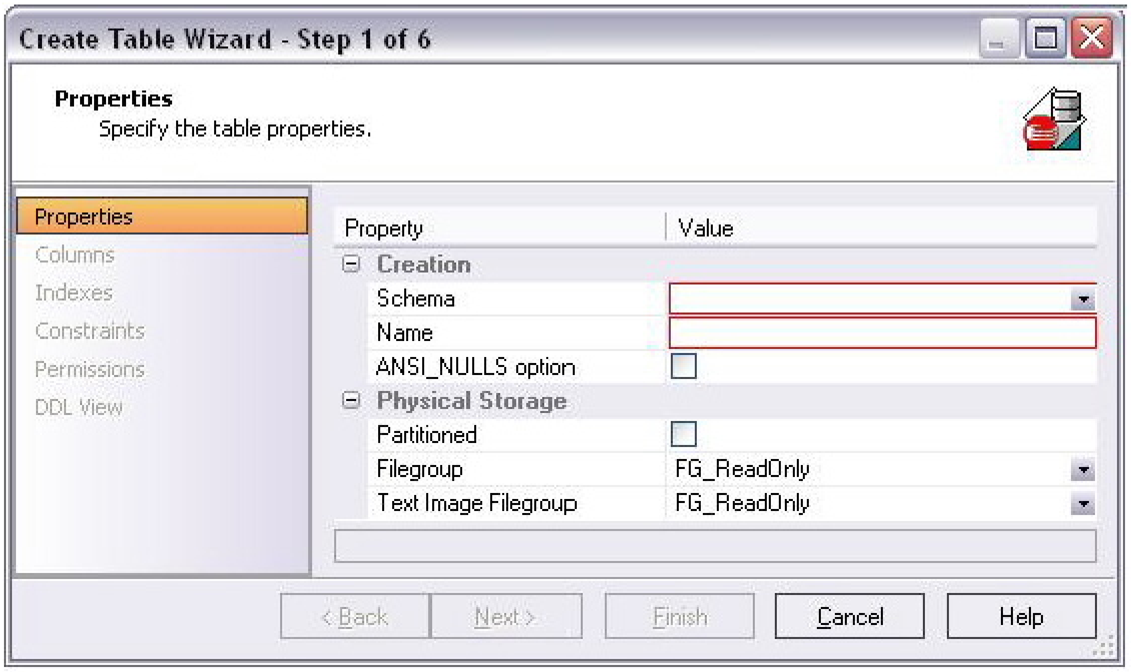


Proceed to Creating an Object Using the Object Creation Wizard.

Creating an Object Using the Object Creation Wizard

From within the Navigator 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.



Create Table Wizard - Step 1 of 6

Properties
Specify the table properties.

Property	Value
Creation	
Schema	
Name	
ANSI_NULLS option	☐
Physical Storage	
Partitioned	☐
Filegroup	FG_ReadOnly
Text Image Filegroup	FG_ReadOnly

2. Select a **Schema** and 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**.

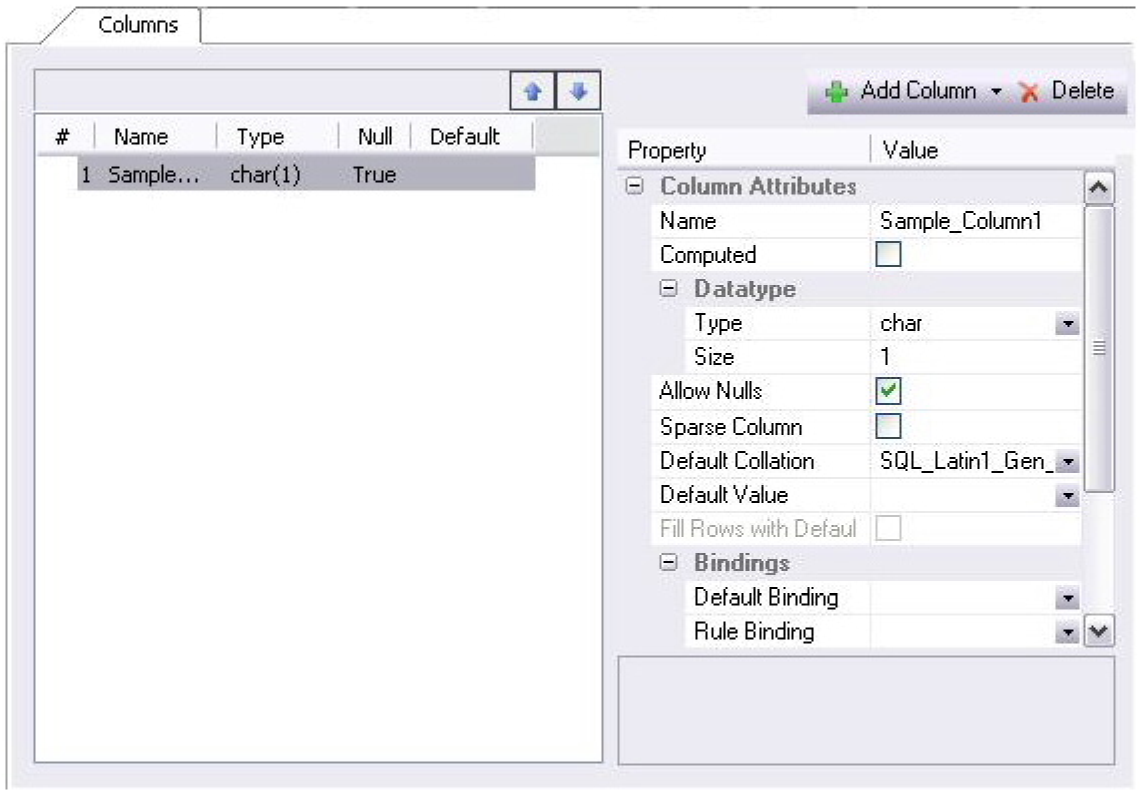
Rapid SQL builds the platform-specific SQL code, syntactically-correct and ready to run the first time. There is no SQL coding required in any of the Rapid SQL Object Creation Wizards.

Proceed to Working With an Existing Object Using the Object Editor.

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 Navigator, ensure that the **Tables** node is expanded and then right-click on your new table and select **Open**.



#	Name	Type	Null	Default
1	Sample...	char(1)	True	

Property	Value
Column Attributes	
Name	Sample_Column1
Computed	☐
Datatype	
Type	char
Size	1
Allow Nulls	✓
Sparse Column	☐
Default Collation	SQL_Latin1_Gen_
Default Value	
Fill Rows with Defaul	☐
Bindings	
Default Binding	
Rule Binding	

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.

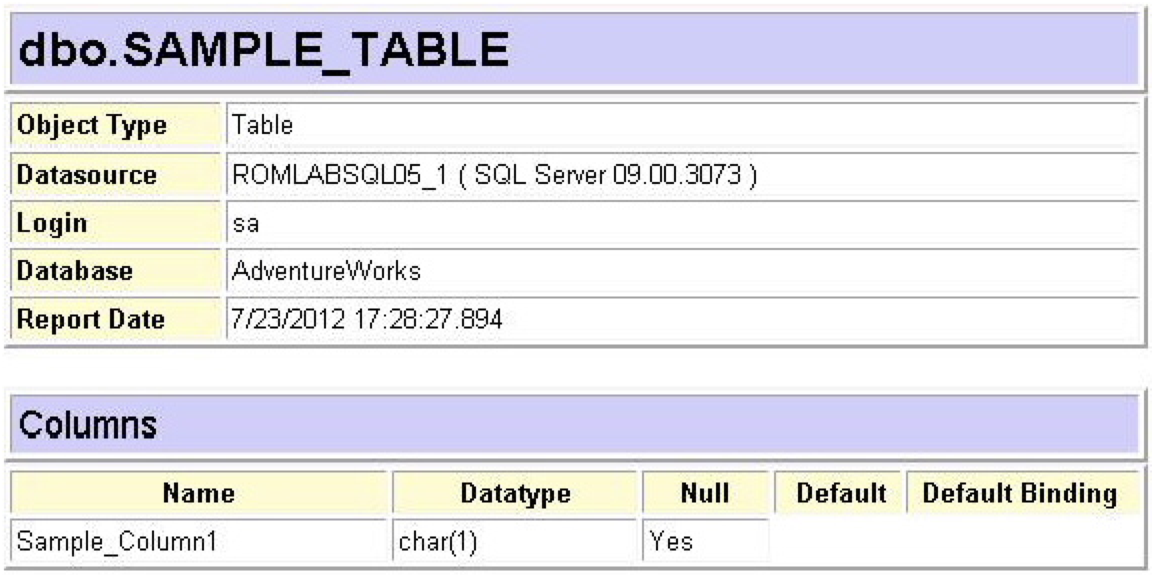
Proceed to Object Documentation and Reporting.

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

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



dbo.SAMPLE_TABLE

Object Type	Table
Datasource	ROMLABSQL05_1 (SQL Server 09.00.3073)
Login	sa
Database	AdventureWorks
Report Date	7/23/2012 17:28:27.894

Columns

Name	Datatype	Null	Default	Default Binding
Sample_Column1	char(1)	Yes		

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.

Proceed to Working With Code, Files and Data.

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.

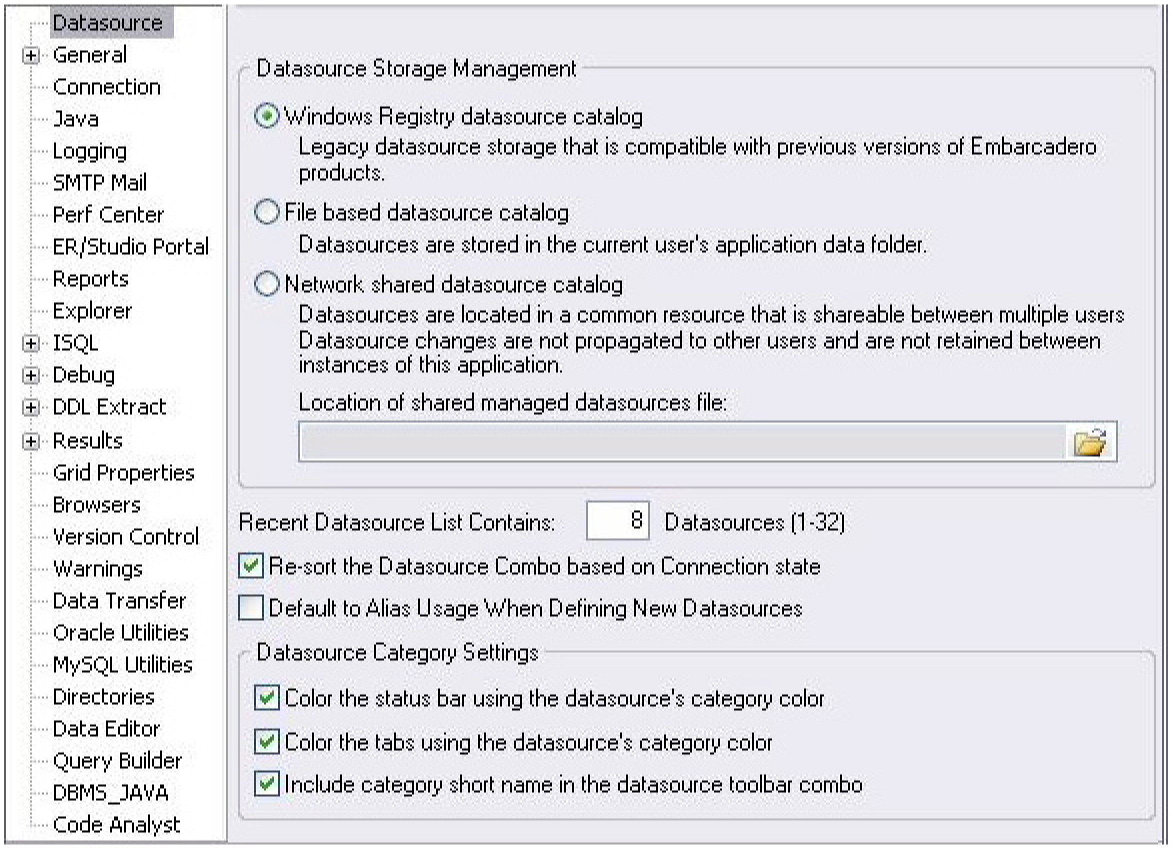
Note: For purposes of this exercise, we are only covering the high-level functionality of the major features and tools within Rapid SQL.

Proceed to Setting Environment Options.

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.



Datasource Storage Management

[✓ Windows Registry datasource catalog]
Legacy datasource storage that is compatible with previous versions of Embarcadero products.

File based datasource catalog
Datasources are stored in the current user's application data folder.

Network shared datasource catalog
Datasources are located in a common resource that is shareable between multiple users Datasource changes are not propagated to other users and are not retained between instances of this application.

Location of shared managed datasources file:

Recent Datasource List Contains: 8 Datasources (1-32)

✓ Re-sort the Datasource Combo based on Connection state

Default to Alias Usage When Defining New Datasources

Datasource Category Settings

✓ 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

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

2. Close the **Options** dialog.

Proceed to Script Library.

Script Library

The Script Library 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 Script Library:

1. Select View > Script Library. The **Script Library** window opens.
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 Script Library

1. Right-click the **Script Library** folder.
2. Select **New Folder** from the context menu. A new folder is added to the **Script Library** folder.

To close the Script Library window:

Select View > Script Library.

Proceed to Working with Scripts and Files.

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.

File Execution Facility

Files containing SQL scripts can be added to the File Execution Facility and executed immediately or scheduled to run later.

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.

Script Execution Facility

The **Tools** menu also offers a Script Execution Facility. Similar to the File Execution facility, it lets you type or paste the script to be executed.

Proceed to Viewing Data.

Viewing Data

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

Select * Browsing

1. On the Datasource Navigator, expand the **MS SQL Servers** node.
2. Expand any database you know has table data, expand the **Tables** node, right-click a table, and then click **SELECT * FROM**.

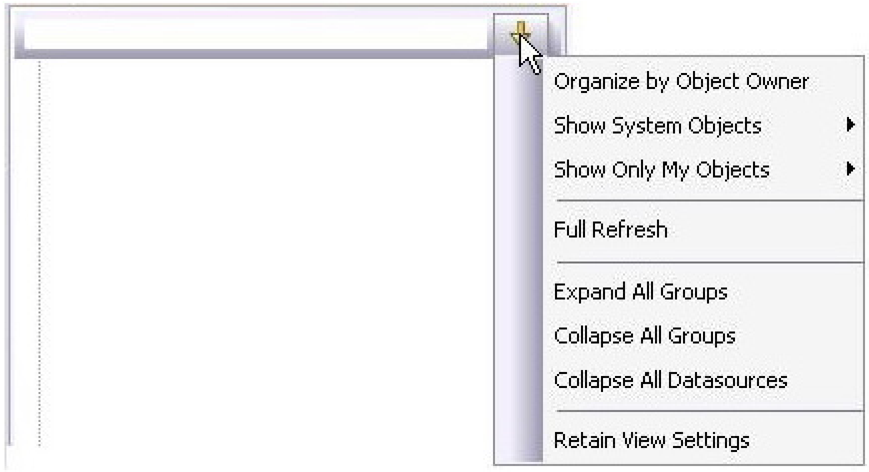
All columns and rows from the table are displayed in the active workspace.

3. Close the workspace window.

Proceed to Retaining Datasource Navigator View Settings.

Retaining Datasource Navigator View Settings

1. Click on the dropdown at the top of the Datasource Navigator.



2. Select **Retain View Settings**.

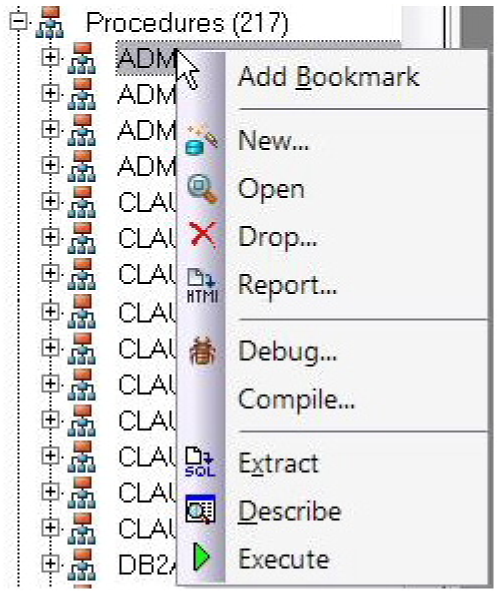
Next time, the Navigator will open just as you left it. All connections that were present when you closed Rapid SQL will be re-established.

Proceed to Datasource Navigator Bookmarks.

Datasource Navigator Bookmarks

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

1. Right-click on any node in the Datasource Navigator.

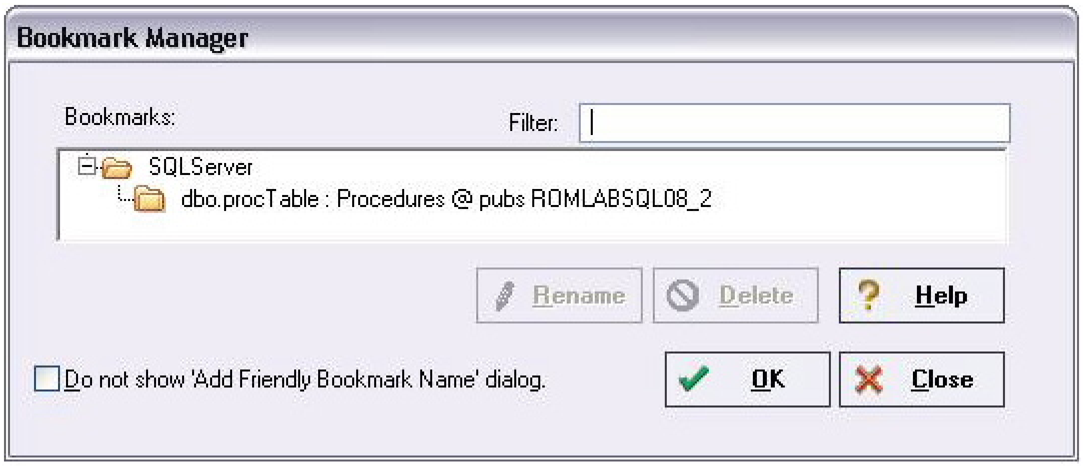


2. Select **Add Bookmark** and use the **Add Friendly Bookmark Name** dialog to optionally provide a new name, and create the bookmark.

After Bookmarks are defined, you can use them to easily navigate to commonly used datasource resources. In this case, you would select the dropdown at the top of the Datasource Navigator, selecting Bookmarks > SQLServer and then selecting the bookmark you just created.

The **Bookmark Manage**r handles maintenance of Bookmarks.

3. Select the dropdown at the top of the Datasource Navigator and then select Bookmarks>Bookmark Manager.

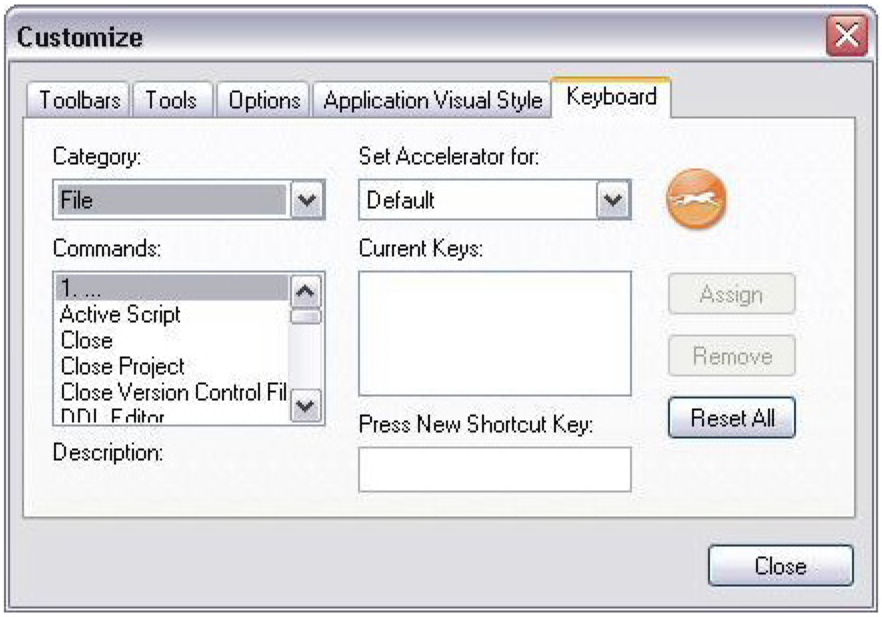

Bookmark Manager
Bookmarks:
Filter:
SQLServer
dbo.procTable : Procedures @ pubs ROMLABSQL08_2
Rename
Delete
Help
Do not show 'Add Friendly Bookmark Name' dialog.
OK
Close

4. Close the **Bookmark Manager** dialog.

Proceed to Setting Keyboard Shortcuts and Hotkeys.

Setting Keyboard Shortcuts and Hotkeys

1. On the **Tools** menu, select **Customize**.
2. In the **Customize** dialog, go to the **Keyboard** tab.



Category: File

Set Accelerator for: Default

Commands:
1. ...
Active Script
Close
Close Project
Close Version Control Fil
DDL Editor

Current Keys:

Press New Shortcut Key:

Description:

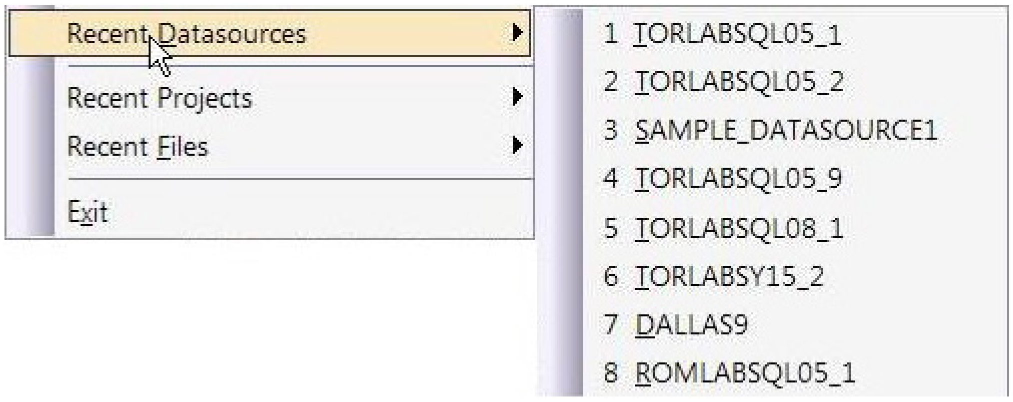
The **Keyboard** tab can be used to set keyboard shortcut hotkeys for all areas of Rapid SQL functionality.

3. Close the **Customize** dialog.

Proceed to Referencing Most Recently Used Datasources.

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 Navigator, ready to work with active connection.

Proceed to Session 3: Scripting.

Session 3: Scripting

This session looks at Rapid SQL's development environment:

- o Generating Code
- o Right-click feature
- o Automated error detection and coding assistance
- o Other coding aids

Generating Code

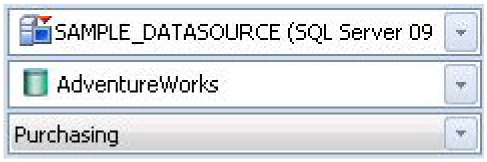
By providing several code generation and assistance options, Rapid SQL makes cross-platform development easy for developers of all experience levels.

Note: The following examples build on the 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, the **AdventureWorks** database, and the **Purchasing** schema from the dropdown list boxes.

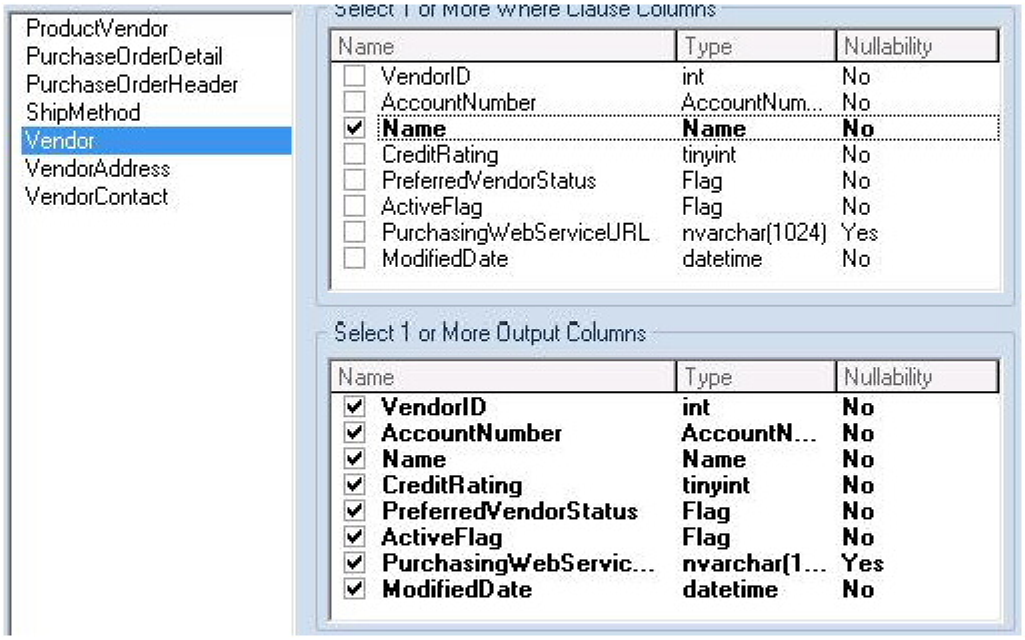


SAMPLE_DATASOURCE (SQL Server 09

AdventureWorks

Purchasing

3. Select the **Vendor** table, **Name** as the input column and all columns for output.



ProductVendor
PurchaseOrderDetail
PurchaseOrderHeader
ShipMethod
Vendor
VendorAddress
VendorContact

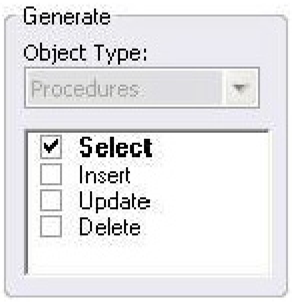
Select 1 or More Where Clause Columns

Name	Type	Nullability
☐ VendorID	int	No
☐ AccountNumber	AccountNum...	No
✓ **Name**	**Name**	**No**
☐ CreditRating	tinyint	No
☐ PreferredVendorStatus	Flag	No
☐ ActiveFlag	Flag	No
☐ PurchasingWebServiceURL	nvarchar(1024)	Yes
☐ ModifiedDate	datetime	No

Select 1 or More Output Columns

Name	Type	Nullability
✓ **VendorID**	**int**	**No**
✓ **AccountNumber**	**AccountN...**	**No**
✓ **Name**	**Name**	**No**
✓ **CreditRating**	**tinyint**	**No**
✓ **PreferredVendorStatus**	**Flag**	**No**
✓ **ActiveFlag**	**Flag**	**No**
✓ **PurchasingWebServic...**	**nvarchar(1...**	**Yes**
✓ **ModifiedDate**	**datetime**	**No**

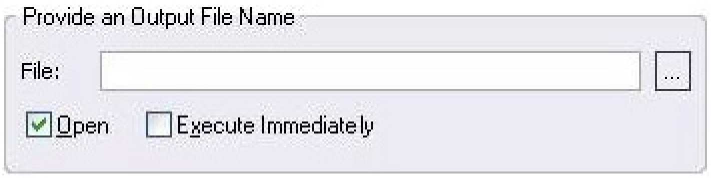
4. Select **Select** as the code option.



Generate
Object Type: Procedures

✓ **Select**
☐ Insert
☐ Update
☐ Delete

5. Select a file to save the generated script and check **Open**.



Provide an Output File Name
File: ...
✓ Open ☐ Execute Immediately

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_vendors**.

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

The indicated file will be saved on the selected directory.

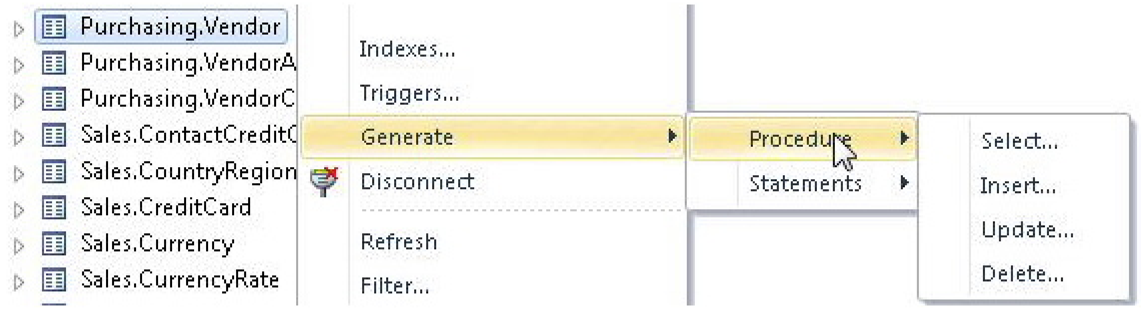
Note: No SQL statement coding is required to generate complete stored procedures and packages. If applicable, Rapid SQL allows all generated code to be previewed and edited to fit any development need.

Proceed to Right-click feature.

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 Datasource Navigator tree, expand the SAMPLE_DATASOURCE > AdventureWorks > Tables sub-node.
2. Right-click on the **Vendors** table.



3. Select **Generate** > **Procedure** > **Select**.
4. Select **Name** 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_vendors2**.
6. Click on the **Execute** or **Step Execute** button to submit the DDL and create the procedure.

Proceed to Automated error detection and coding assistance.

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.

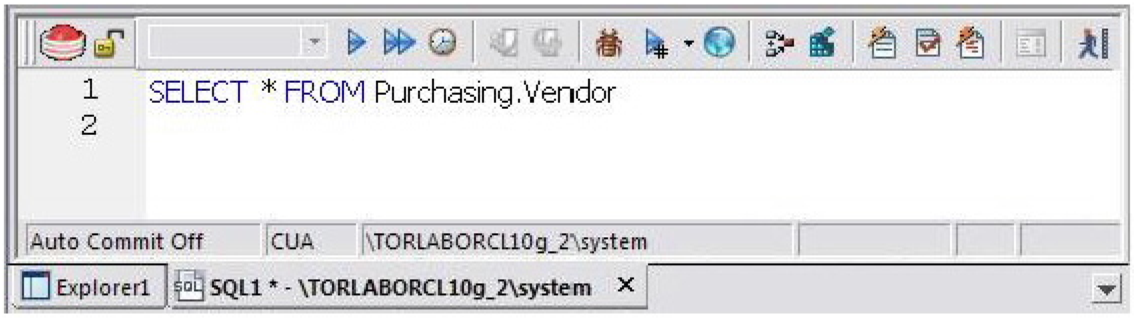
To enable these features:

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 **SQL**.

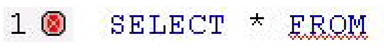
Rapid SQL opens the SQL Editor window. You can add SQL code via your method of choice (free-form typing, retrieve from a file, paste copied code, etc.).



```
SELECT * FROM Purchasing.Vendor
```

2. Type **SELECT * FROM** and stop typing.

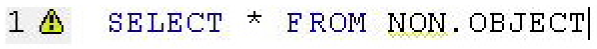
Note the error condition.



```
SELECT * FROM
```

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.



```
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**.

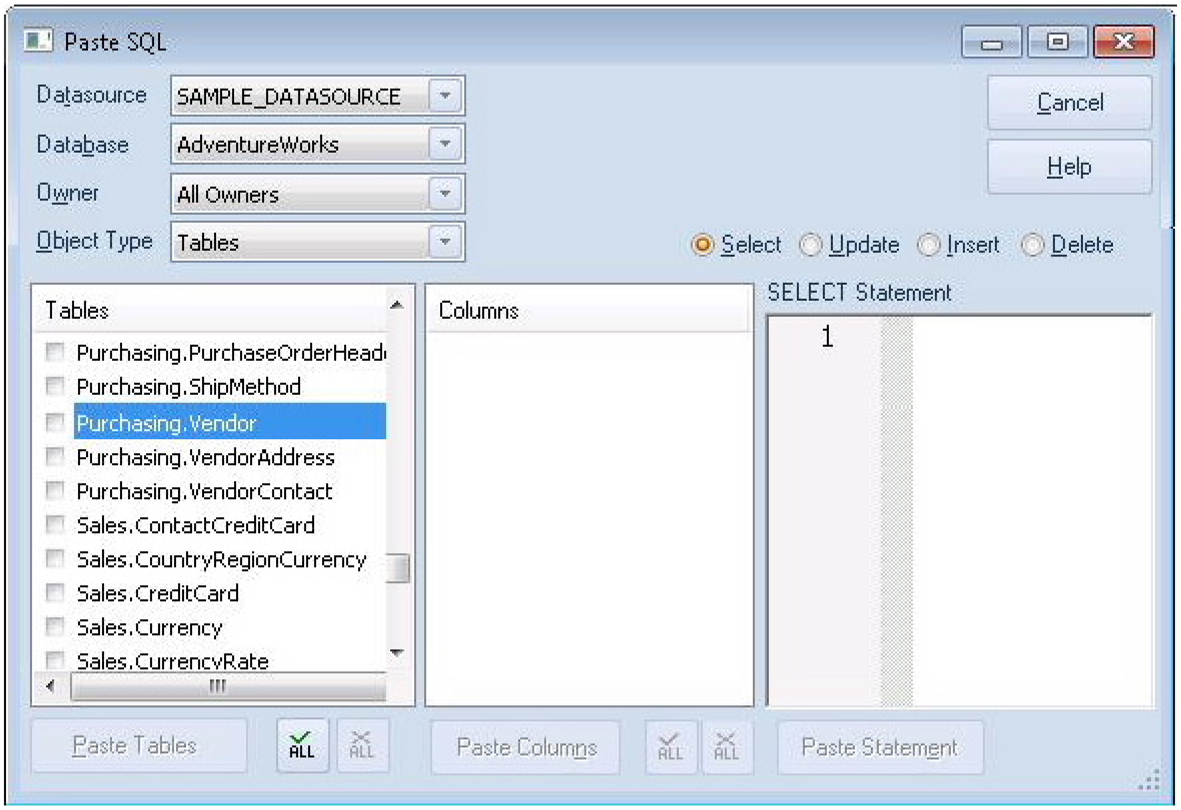
Proceed to Other coding aids.

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 Datasource Navigator tree, expand the SAMPLE_DATASOURCE > AdventureWorks 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. On the **Paste SQL** dialog, select **Sample_Datasource** from the **Datasource** dropdown, **AdventureWorks** from **Database**, **All Owners** from **Owner**, and an **Object Type** of **Tables**.


Paste SQL

Datasource: SAMPLE_DATASOURCE
Database: AdventureWorks
Owner: All Owners
Object Type: Tables

[✓ Select] Update Insert Delete

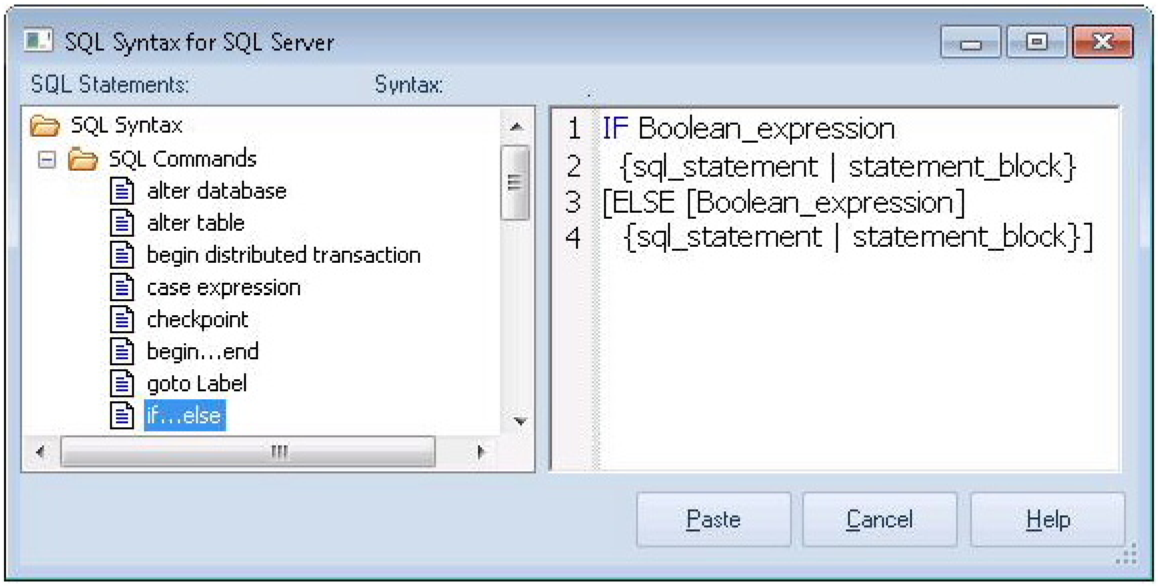
Tables
- Purchasing.PurchaseOrderHead
- Purchasing.ShipMethod
- Purchasing.Vendor
- Purchasing.VendorAddress
- Purchasing.VendorContact
- Sales.ContactCreditCard
- Sales.CountryRegionCurrency
- Sales.CreditCard
- Sales.Currency
- Sales.CurrencyRate

5. In the **Tables** list, select **Purchasing.Vendor**.
6. Under the **Columns** list, click **ALL**.
7. Click the **Select** radio button.
8. 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.



```
IF Boolean_expression
  {sql_statement | statement_block}
[ELSE [Boolean_expression]
   {sql_statement | statement_block}]
```

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.

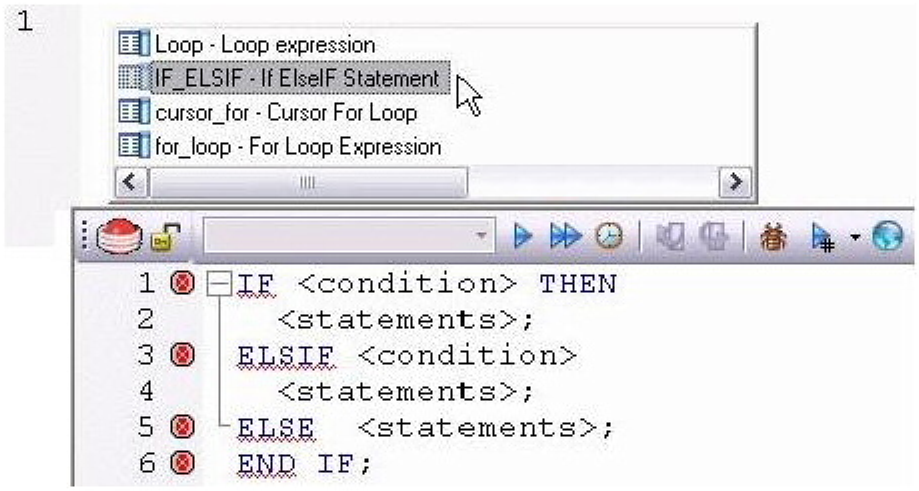
Proceed to Session 4: Working with Code Workbench.

Session 4: Working with Code Workbench

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

- Code Templates
- Auto Replace

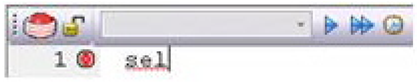
Code templates are complete code blocks that can be easily added to open windows or scripts with a few keystrokes. When you type CTRL+SPACE, the Code Assist menu opens, letting you select a code template for insertion in the editor window.


Loop - Loop expression
IF_ELSIF - If ElseIF Statement
cursor_for - Cursor For Loop
for_loop - For Loop Expression

```
IF <condition> THEN
  <statements>;
ELSIF <condition>
  <statements>;
ELSE  <statements>;
END IF;
```

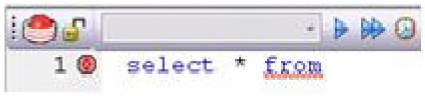
Auto Replace lets you define shortcuts consisting of a few characters that represent longer character strings. Instances of these Auto Replace expressions are automatically replaced by the replacement string on activation events such as typing SPACE, TAB, or RETURN. This feature is useful for creating 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**:



```
sel
```

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

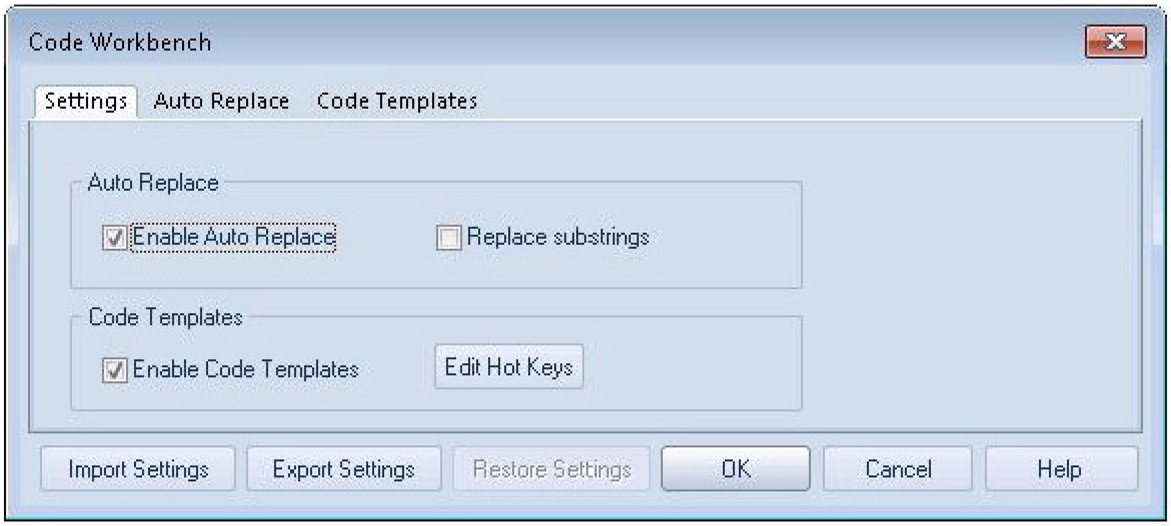


```
select * from
```

Rapid SQL loads a default set of Auto Replace and Code template definitions at startup, but you can also add, edit, and delete definitions. Also, 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.



Auto Replace

[✓] Enable Auto Replace
[ ] Replace substrings

Code Templates

[✓] Enable Code Templates

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**.

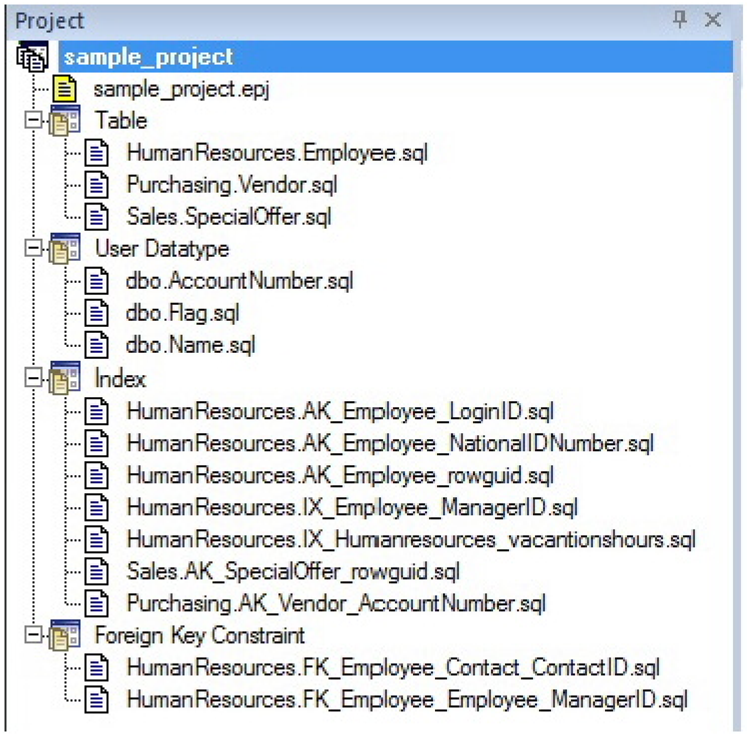
Proceed to Session 5: Building a Database Project.

Session 5: Building a Database 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. You can subsequently perform tasks such as distribute the files or add them to a Version Control System (VCS). This example will reverse engineer the table objects from the Microsoft SQL Server **AdventureWorks** database into a Rapid SQL project

1. Select File > New > Project to open the wizard.
2. Enter **sample_project** as the name, and accept the default target directory. Select **From Database** and click **OK**.
3. On the **Connection** page, select **SAMPLE_DATASOURCE** and click **Next**.
4. On the **Catalogs** page, select **AdventureWorks** and click **Next**.
5. Click **Schemas** to open the **Schema Selection** dialog. Select **Human Resources**, **Purchasing**, and **Sales**, and then click **OK**. The **Object Selection** page opens.
6. Under **Object Types**, select **Tables**. Then, under **Objects**, expand the **Tables** node and select the **Purchasing.Vendor**, **Sales.SpecialOffer**, and **HumanResources.Employee** tables. Click **Next**.
7. On the **Options** page, leave the default selections and click **Finish**.
8. When the **Execute** page shows that reverse engineering is complete, click **Continue**.

The **Project** tab displays the results of the reverse engineering.


Project

sample_project
- sample_project.epj
- Table
  - HumanResources.Employee.sql
  - Purchasing.Vendor.sql
  - Sales.SpecialOffer.sql
- User Datatype
  - dbo.AccountNumber.sql
  - dbo.Flag.sql
  - dbo.Name.sql
- Index
  - HumanResources.AK_Employee_LoginID.sql
  - HumanResources.AK_Employee_NationalIDNumber.sql
  - HumanResources.AK_Employee_rowguid.sql
  - HumanResources.IX_Employee_ManagerID.sql
  - HumanResources.IX_Humanresources_vacantionshours.sql
  - Sales.AK_SpecialOffer_rowguid.sql
  - Purchasing.AK_Vendor_AccountNumber.sql
- Foreign Key Constraint
  - HumanResources.FK_Employee_Contact_ContactID.sql
  - HumanResources.FK_Employee_Employee_ManagerID.sql

Before closing the **Project** tab, investigate the individual project elements. Also, check the **File** and **Project** menus for project and version control options.

Proceed to Session 6: Visual Query Builder.

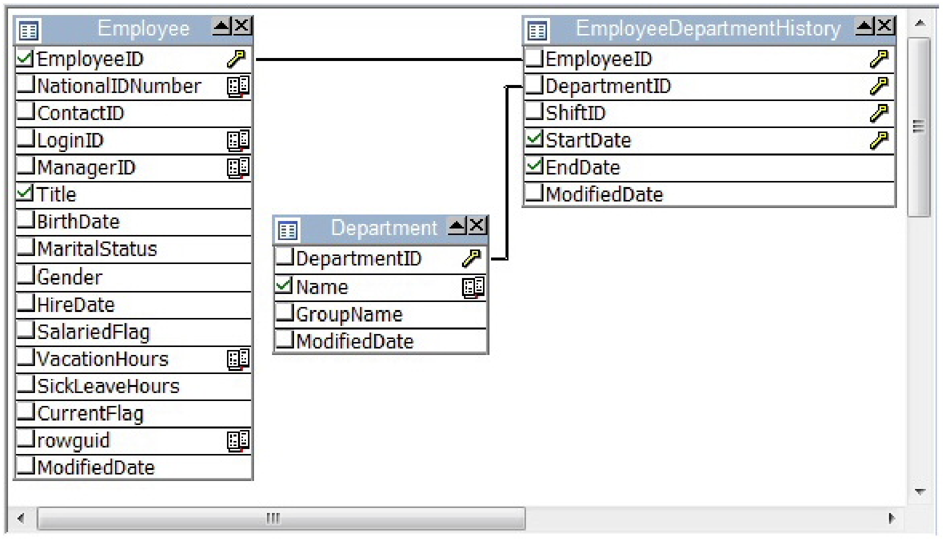
Session 6: Visual Query Builder

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

1. From the Navigator tree, right-click on the **HumanResources.Employee** table and select **Build Query**.

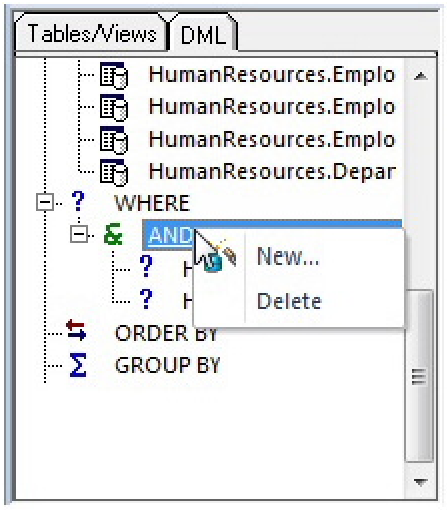
The table is automatically added to the Query Builder workspace.

2. On the **Tables/Views** tab, right-click on the **Department** table and select **Add**.
3. Similarly, on the **Tables/Views** tab, right-click on the **EmployeeDepartmentHistory** table and select **Add**.
4. Rearrange the contents of the window to look like the following graphic.
5. Select the columns indicated in the graphic.



Note that the tables are automatically identified as being joined by any columns with the same name and datatype. Also, note that the query is being built in the lower pane.

6. 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.



7. Experiment with adding, deleting, and modifying clauses.

8. 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.

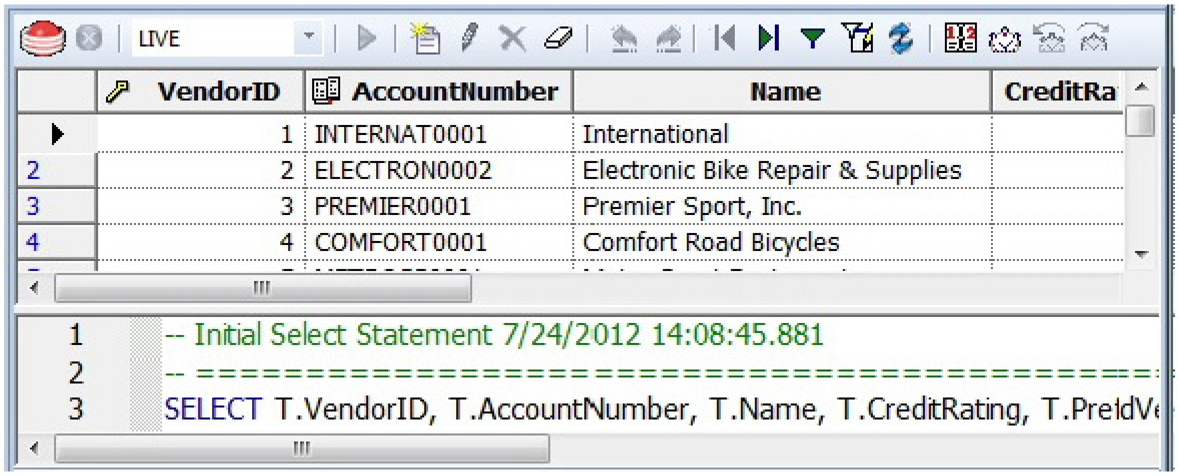
Proceed to Session 7: Live Data Editor.

Session 7: Live Data Editor

1. From the Navigator, right-click on the **Purchasing.Vendor** 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.



	VendorID	AccountNumber	Name	CreditRa
▶	1	INTERNAT0001	International	
2	2	ELECTRON0002	Electronic Bike Repair & Supplies	
3	3	PREMIER0001	Premier Sport, Inc.	
4	4	COMFORT0001	Comfort Road Bicycles	

```
-- Initial Select Statement 7/24/2012 14:08:45.881
-- ==========================================
SELECT T.VendorID, T.AccountNumber, T.Name, T.CreditRating, T.PreIdVe
```

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 canceled by selecting the **Reload Data** icon.

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

Proceed to Session 8: Code Analyst.

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 statistics for single objects, but you can group more than one object.

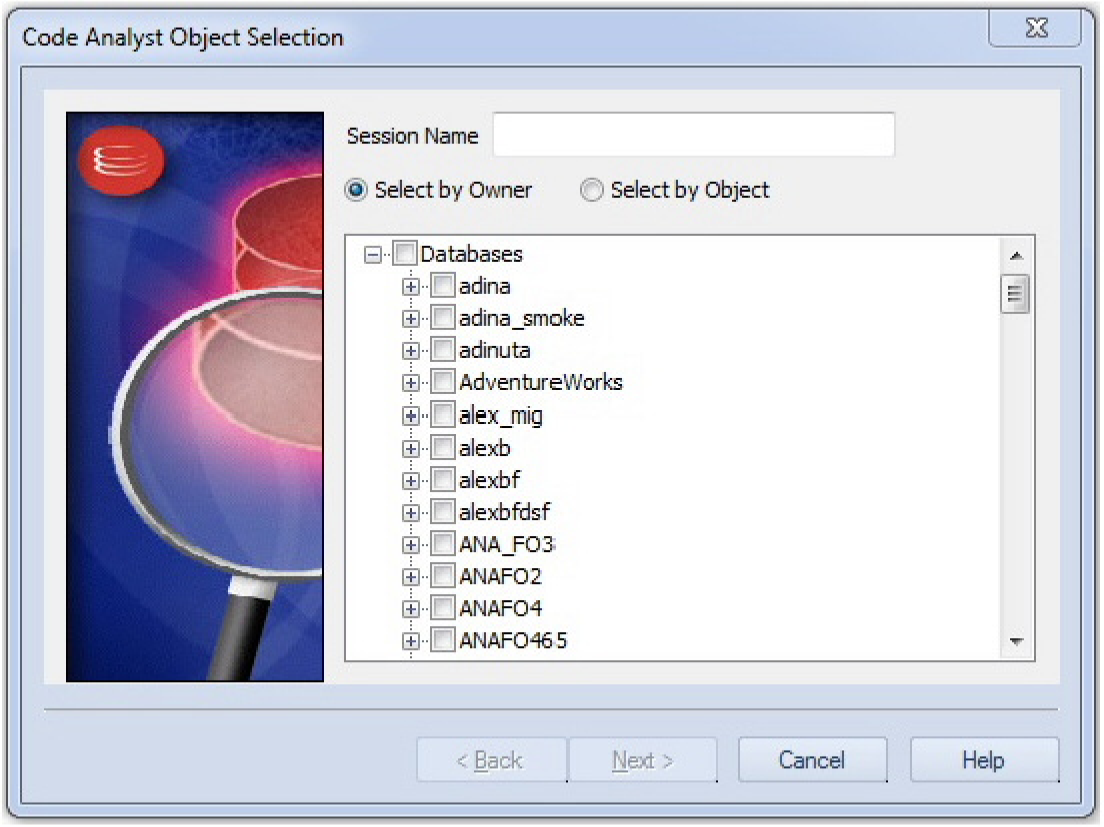
To get started

1. Select Tools > Code Analyst.

Note: For Code Analyst to run, 5 repository tables will be created in the database. Select the database you would like the tables to be installed on and press **OK**. Once the tables are installed, you are ready to start defining a session.

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

2. On the **Code Analyst Object Selection** dialog, provide a **Session Name**, locate and select the objects to be executed, and click **Next**.



Session Name

[✓ Select by Owner] Select by Object

- Databases
  - adina
  - adina_smoke
  - adinuta
  - AdventureWorks
  - alex_mig
  - alexb
  - alexbf
  - alexbfdsf
  - ANA_FO3
  - ANAFO2
  - ANAFO4
  - ANAFO465

3. 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 execution is displayed in the **Run Summary** tab.

4. 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.
5. Close the **Code Analyst** window.

Proceed to Session 9: SQL Debugging and Profiling.

Session 9: SQL Debugging and Profiling

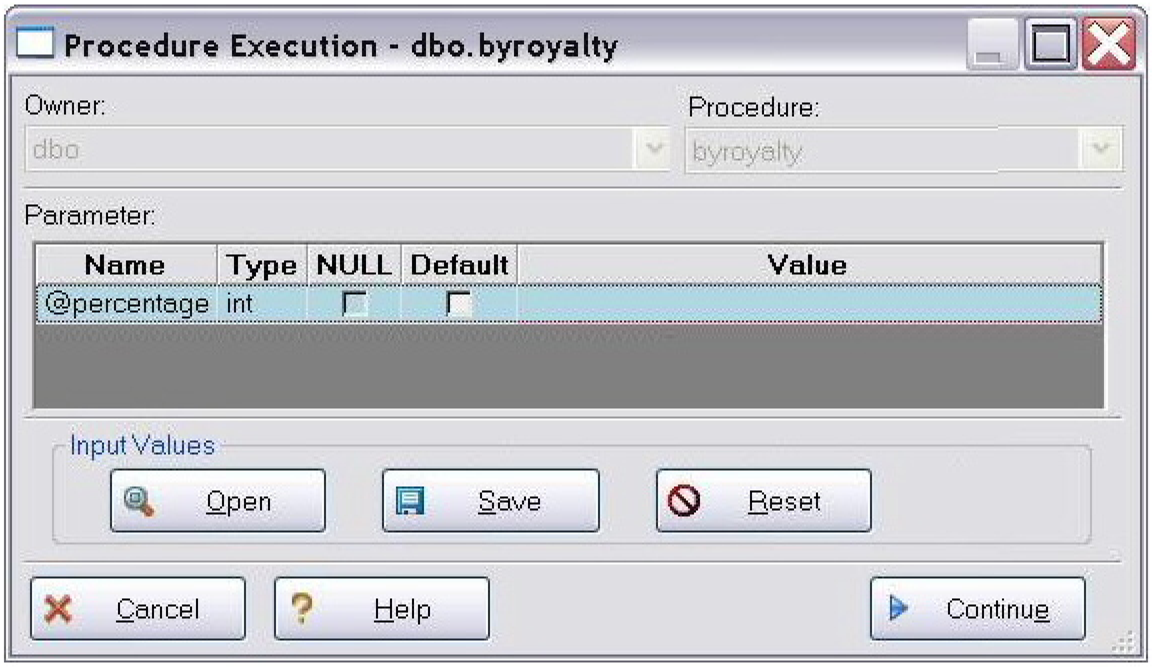
Rapid SQL offers the following facilities that help you test and optimize code:

- SQL Debugging
- SQL Profiling- Oracle Only

SQL Debugging

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

1. In the Navigator, 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.



Procedure Execution - dbo.byroyalty

Owner: dbo

Procedure: byroyalty

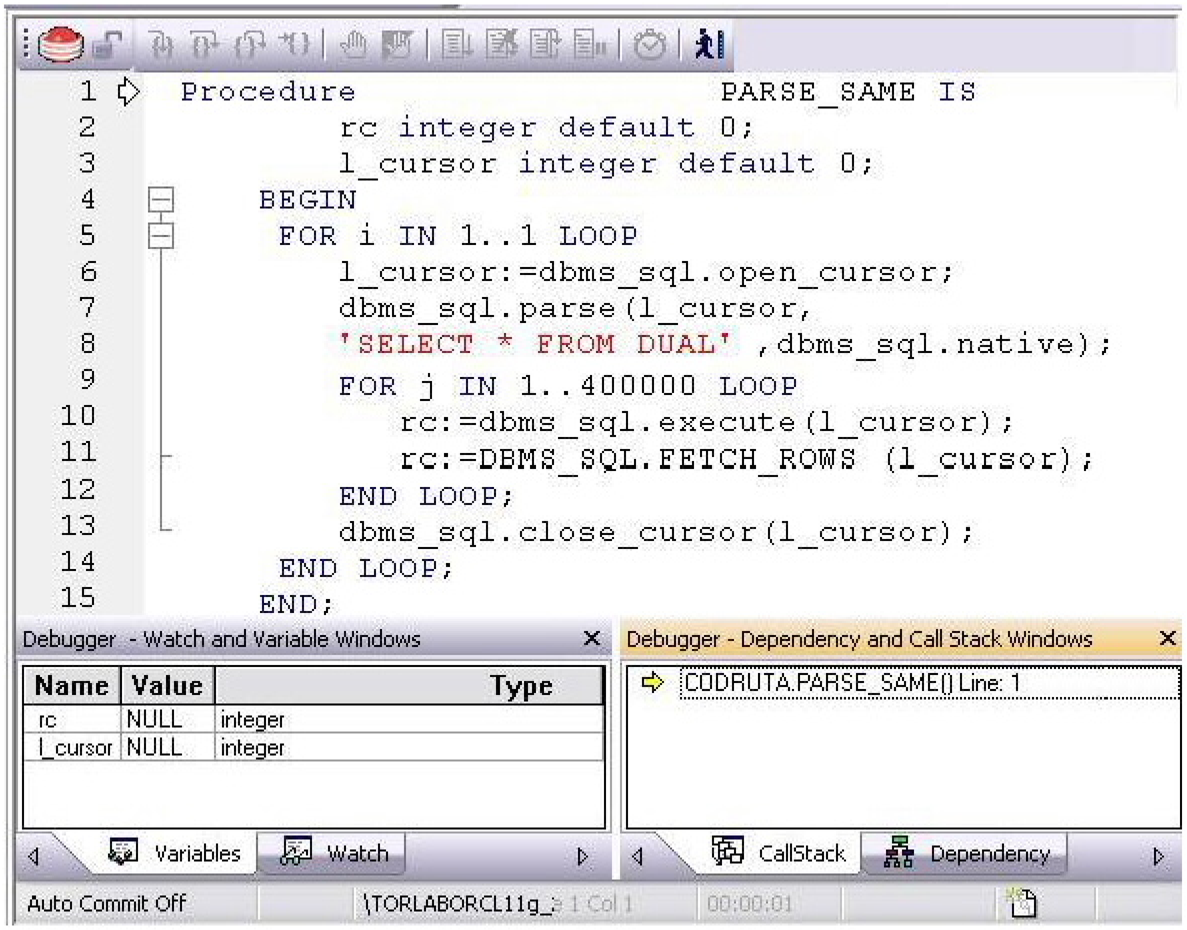
Parameter:

Name	Type	NULL	Default	Value
@percentage	int	☐	☐	

4. Enter input values 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.



```
Procedure                    PARSE_SAME IS
        rc integer default 0;
        l_cursor integer default 0;
    BEGIN
     FOR i IN 1..1 LOOP
        l_cursor:=dbms_sql.open_cursor;
        dbms_sql.parse(l_cursor,
        'SELECT * FROM DUAL' ,dbms_sql.native);
        FOR j IN 1..400000 LOOP
           rc:=dbms_sql.execute(l_cursor);
           rc:=DBMS_SQL.FETCH_ROWS (l_cursor);
        END LOOP;
        dbms_sql.close_cursor(l_cursor);
     END LOOP;
    END;
```

Debugger - Watch and Variable Windows

Name	Value	Type
rc	NULL	integer
l_cursor	NULL	integer

Debugger - Dependency and Call Stack Windows

CODRUTA.PARSE_SAME() Line: 1

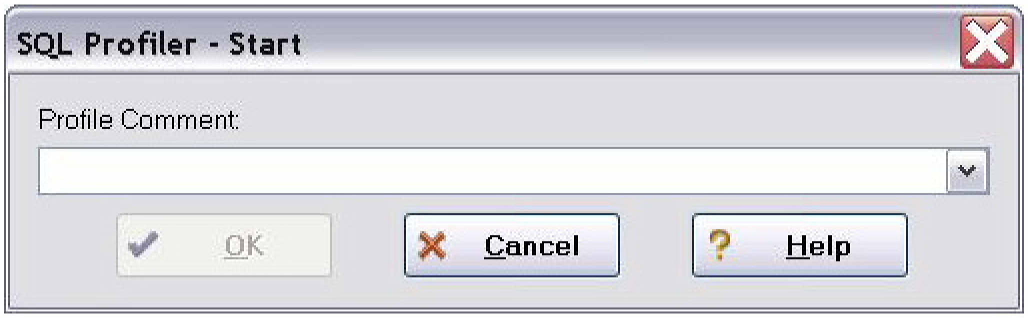
The Debugger features basic execution, line-by-line execution, breakpoint support, and other common debugging features. For details, refer to the relevant online Help topics.

Proceed to SQL Profiling- Oracle Only.

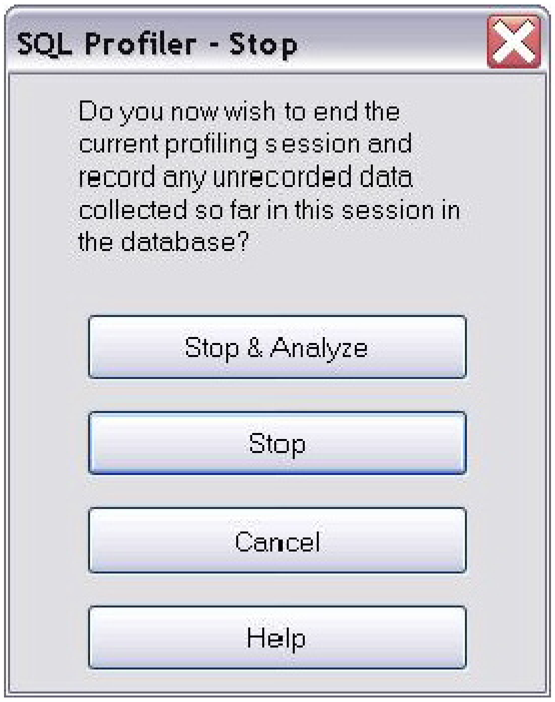
SQL Profiling- Oracle Only

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.

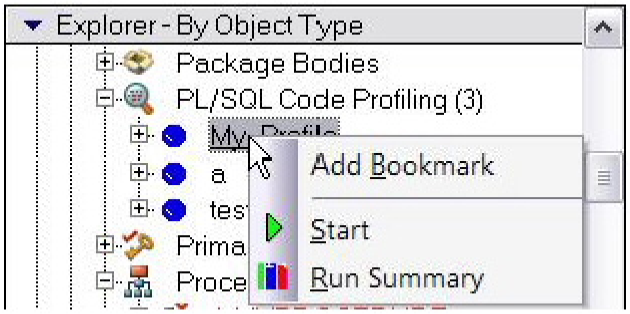


3. Execute the programmable object (i.e. Stored Procedure) you wish to capture metrics on.
4. When finished, select Tools > SQL Profiler > Stop. The **SQL Profiler – Stop** dialog window prompts you to select an option.

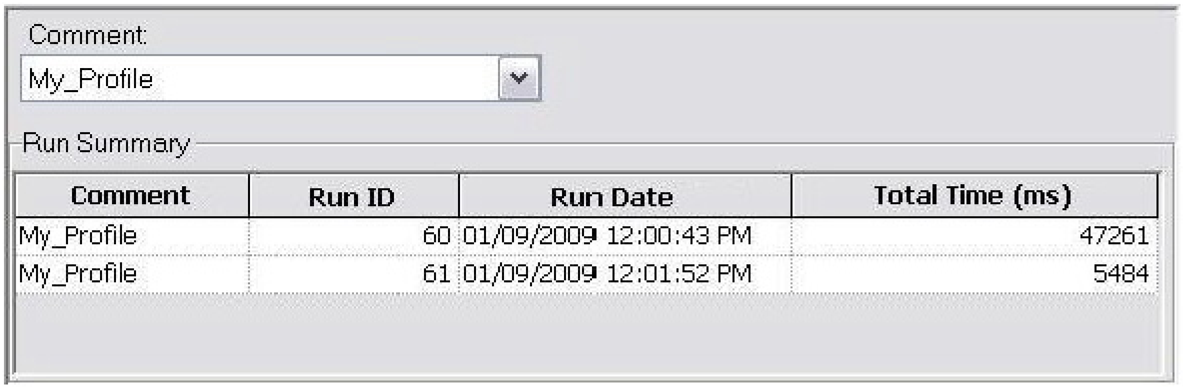


Do you now wish to end the current profiling session and record any unrecorded data collected so far in this session in the database?

5. Press **Stop**.
6. On the Navigator, expand the **PL/SQL Code Profiling** node.



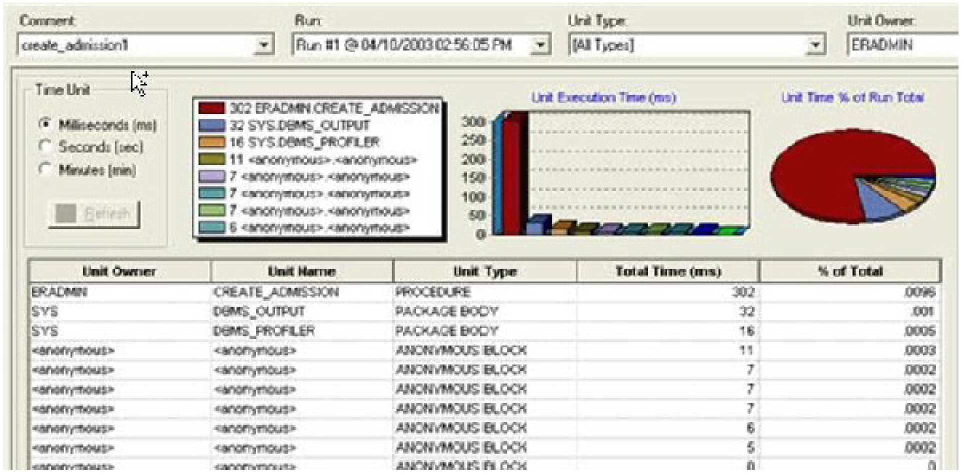
7. Right-click on the profile session and select **Run Summary**. The **Run Summary** window opens.


Comment: My_Profile

Run Summary

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 execution in both a graphical and text format.


Comment: create_admission1
Run: Run #1 @ 04/10/2003 02:56:05 PM
Unit Type: [All Types]
Unit Owner: ERADMIN

Time Unit: (●) Milliseconds (ms) ( ) Seconds (sec) ( ) Minutes (min)

Refresh
302 ERADMIN.CREATE_ADMISSION
32 SYS.DBMS_OUTPUT
16 SYS.DBMS_PROFILER
11 <anonymous>.<anonymous>
7 <anonymous>.<anonymous>
7 <anonymous>.<anonymous>
7 <anonymous>.<anonymous>
6 <anonymous>.<anonymous>
Unit Execution Time (ms)
300
250
200
150
100
50
0
Unit Time % of Run Total

Unit Owner	Unit Name	Unit Type	Total Time (ms)	% of Total
ERADMIN	CREATE_ADMISSION	PROCEDURE	302	.0096
SYS	DBMS_OUTPUT	PACKAGE BODY	32	.001
SYS	DBMS_PROFILER	PACKAGE BODY	16	.0005
<anonymous>	<anonymous>	ANONYMOUS BLOCK	11	.0003
<anonymous>	<anonymous>	ANONYMOUS BLOCK	7	.0002
<anonymous>	<anonymous>	ANONYMOUS BLOCK	7	.0002
<anonymous>	<anonymous>	ANONYMOUS BLOCK	7	.0002
<anonymous>	<anonymous>	ANONYMOUS BLOCK	6	.0002
<anonymous>	<anonymous>	ANONYMOUS BLOCK	5	.0002
<anonymous>	<anonymous>	ANONYMOUS BLOCK	0	0

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 Application Preferences and Feature Options

The following topics show you how to customize the application configuration for your specific needs:

- Configuring Feature Options
- Locking Down Features Using the Registry

Configuring Feature Options

All 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, select **Options**.

The Options Editor opens.

2. Click the tab corresponding to the feature you want to customize. Set feature options on the tab and then click **OK**.

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.
Datasource Options	Specifies general datasource management options and Datasource Navigator preferences.
DBMS Java Options	Specifies load Java files and drop Java files.
DB2 Utilities Options	Sets default directories for executables used by DB2 LUW.
DDL Extract Options	Specifies whether or not DROP statements should be included 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.
Explorer Options	Sets defaults for the organization of objects in the Datasource Navigator.
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.
LDAP Configuration Options	Sets preferences for LDAP Server usage.
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.
Query Builder Options	Specifies global settings for Query Builder.
Reports Options	Specifies global settings for reports.
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.
Team Server 2016 Options	Lets you integrate with the Embarcadero Team Server 2016 data governance matadata repository.
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.

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

Connection Options

The **Connection** tab of the Options Editor lets you specify common connectivity options and provides access to DBMS-specific options.

Note: For Rapid SQL system requirements and details on DBMS product support, see the **Read Me** at http://docs.embarcadero.com/products/rapid_sql/.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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
Host Name	Name of the client computer.	Local name

See the following topics for DBMS platform-specific options:

- Connection Options - DB2
- Connection Options - MySQL
- Connection Options - Firebird
- Connection Options - InterBase
- Connection Options - PostgreSQL
- Connection Options - Sybase
- Connection Option - Sybase IQ
- Connection Option - SQL Server
- Connection Options - Oracle

Connection Options - DB2

The Connection > DB2 tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. NOTE: For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the DB2 CLI driver	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the DB2 CLI driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - Firebird

The Connection > Firebird tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. NOTE: For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the Firebird ODBC Driver	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the Firebird ODBC Driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - InterBase

The Connection > InterBase tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. NOTE: For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the InterBase ODBC Driver	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the InterBase ODBC Driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - MySQL

The Connection > MyQSL tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. **NOTE:** For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the MySQL Connector/ODBC driver	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the MySQL Connector/ODBC driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - PostgreSQL

The Connection > PostgreSQL tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. NOTE: For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the PostgreSQL ODBC driver	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the PostgreSQL ODBC driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - Sybase

The Connection > Sybase tab of the Options Editor lets you select an installed client other than the active/default client version for Embarcadero driver connections. It also lets you override default client/server settings.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
	Use Quoted Identifiers	If you plan to use delimited identifiers, select this option.

Sybase Open Client Selection	This group of controls apply only to use of the Embarcadero drivers with native Sybase clients.	
	Enable Client Selection	Selecting this check box lets you override the default/active Sybase client specified by the SYBASE, SYBASE_OCS, and/or SYBROOT environment variables. This control enables and disables the **Client Home Directory** and **Client OCS Directory** controls.
	Client Home Directory	Browse to the root directory of the client to be used.
	Client OCS Directory	Select the **OCS-** directory corresponding to the client version that you want to use.
Connection Settings	**SQL INI File**	Browse to locate and select the specific, SQL.INI-formatted file containing server-configuration information. The file does not have to be named **SQL.INI**.
	Packet Size, **Max Connections**, and **Client Character Set**	These settings let you create a set of default parameters used in setting up connections to Sybase ASE datasources. These settings can be overridden when registering a Sybase ASE datasource. For details, see Providing Sybase ASE Connection Information When Registering Datasources.
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. **NOTE:** For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the Sybase Open Client	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the Sybase Open Client** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - Sybase IQ

The Connection > Sybase IQ tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Setting	Description	
	Use Quoted Identifiers	If you plan to use delimited identifiers, select this option.
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. **NOTE:** For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the Sybase IQ ODBC Driver	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the Sybase IQ ODBC Driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - SQL Server

The Connection > SQL Server tab of the Options Editor lets you specify DBMS-specific connection options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Setting	Description	
SQL Server Native Client Selection	These controls let you specify a default native driver. NOTE: For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes.	
	Enable Client Selection	Where multiple native drivers are supported, this check box enables selection of a native client to use as the default option.
	Clients	Select the native client to be used as default.
	Use Quoted Identifiers	If you plan to use delimited identifiers, select this option.
Connection Settings	**Packet Size**	This control lets you specify a default packet size parameter used with SQL Server connections. This setting can be overridden when registering a SQL Server datasource. For details, see Providing SQL Server Connection Information When Registering Datasources.
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. NOTE: For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the SQL Server Native Client	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the DB2 CLI driver** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Connection Options - Oracle

The Connection > Oracle tab of the Options Editor lets you select an installed client other than the active/default client version for Embarcadero driver connections. It also lets you override default client/server settings.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For more general connectivity options, see **Connection Options**.

Group	Setting	Description
Hide time portion of DATE datatype if zero		If selected, the time portion (HH:MM:SS) of DATE datatype results are suppressed in the ISQL Editor's Results window, if the time portion is all zeroes.
OCI Client Selection	This group of controls apply only to use of the Embarcadero drivers with native Oracle clients. If no client is specified using these controls, the following sequence is used in determining the client to use for Oracle connections. - The driver specified in the ORACLE_HOME environment variable - The first driver found in the PATH environment variable - The bundled Oracle Instant Client driver (v. 11)	
	Enable Client Selection	Selecting this check box lets you override the default/active client specified by Oracle environment variables. This control enables and disables the **Oracle Home** control.
	Oracle Home	Select the directory of the Oracle client you wish to use.

TNSNAMES.ORA	**Directory Path of Oracle configuration files**	Browse to locate and select the directory containing the **TNSNAMES.ORA** file to use. Ensure that no file name is included with the path. If no file is specified using this control, the following sequence is used in determining the **TNSNAMES.ORA** file to use for Oracle connections. -The directory specified in the TNS_ADMIN environment variable - The directory specified in the TNS_ADMIN key at the top level in HKLM\SOFTWARE\ORACLE - The directory specified in the TNS_ADMIN key in the client subkey (HOME or KEY_ prefixed) corresponding to the value of the ORACLE_HOME (if there is a particular client specified by the user, its TNS_ADMIN subkey will be used if present) - The directory ORACLE_HOME\NETWORK\ADMIN (if there is a particular client specified by the user, ORACLE_HOME is the client's full path) - The directory ORACLE_HOME\net80\ADMIN (if there is a particular client specified by the user, ORACLE_HOME is the client's full path) - All directories in PATH environment variable
Default Driver Selection	By default, when you register a datasource, the native driver is automatically selected as the default connection option. **NOTE:** For details on native driver and JDBC driver support, see the **ReadMe** file selecting View > Release Notes. The controls in this group let you specify one of the installed or packaged DBMS platform-specific JDBC drivers as the default connection option.	
	Connect using JDBC instead of the OCI client	For platforms for which a native driver is supported, this check box enables selection of a JDBC driver as the default connection option. For platforms for which no native driver is available, this check box is disabled.
	JDBC driver to use	If multiple JDBC drivers are available for this platform, this control lets you select the specific JDBC driver to use as the default.
	Assign to all disconnected Datasources	This control lets you assign either the native driver or the currently selected JDBC driver as the connection option for all datasources defined on the relevant DBMS platform that are currently not connected. If the **Connect using JDBC instead of the OCI client** setting is selected, all disconnected datasources are assigned the driver selected from the **JDBC driver to use** list as the connection option. Otherwise, the native driver is assigned to all disconnected datasources on this DBMS platform.

Browsers Options

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.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on Browser features, see Browsers.

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 Results type, Tab delimited, Comma separated, and HTML, which formats the results in a simple HTML table.	Results
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

Code Analyst is a utility that helps you identify time-consuming lines of code. The **Code Analyst** tab of the Options Editor lets you specify Code Analyst operational parameters.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using Code Analyst, 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.	

Data Editor Options

The **Data Editor** tab of the Options Editor lets you provide preferences that dictate behavior of the table data editor.

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

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the Data Editor, see 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
	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, the Calendar Control window is used.	Not selected
	2 digit year system setting warning	If selected, a warning is issued 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 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 Application Preferences 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

The **Datasource** tab of the Options Editor lets you provide datasource categorization and storage preferences.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Option	Description	Default
Datasource Storage Management	Registry-based and file-based datasource catalog management is available. Prior to changing methods, see Specifying a Datasource Catalog Storage Option. The following options are offered: o **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. o **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. o **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 on shutdown. Changing methods does not delete datasource definitions stored using the current method. When you change methods, existing catalog definitions are not automatically converted to the new method. o **Team Server based datasource catalog:** The datasource catalog is located on Team Server 2016 and is shareable between multiple users. Datasources changes are propagated to Team Server 2016. **Note:** Team Server 2016 URL must be configured in Team Server 2016 Options.	Windows Registry
Recent Datasource List Contains	Lets you specify number of datasources to display when you select **Recent Datasources** from the **File** menu.	8
Re-sort the Datasource Combo based	Re-sorts the lists of datasources in the combo. It locates the connected datasources first.	Selected

on Connection state		
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: o **Color the status bar using the datasource's category color** o **Color the tabs using the datasource's category color** o **Include category short name in the datasource toolbar combo** For an introduction to datasource categorization, see Categorizing Datasources using Color-coding and Labeling.	All selected

Note: For information on using Embarcadero Team Server 2016 repository-based datasource definitions, see Team Server 2016 Support.

DBMS Java Options

The **DBMS_JAVA** tab of the Options Editor lets you choose between use of the DBMS_JAVA package and the loadJava.bat batch file when loading or dropping Java classes.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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

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

For related information, see the following object actions:

- Drop Java
- Load Java

DB2 Utilities Options

Some functionality against DB2 LUW datasources makes use of native DB2 utilities.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

The table below describes the options and functionality of the **DB2 Utilities** tab:

Option	Description
db2cmd.exe	Use this control to specify the location of the directory containing the **db2cmd.exe** utility.
db2.exe	Use this control to specify the location of the directory containing the **db2.exe** utility.

DDL Extract Options

The **DDL Extract** tab of the Options Editor lets you specify results-handling preferences for use of the **Extract** object action.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the **Extract** object action, see Extract.

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.
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:

- o The object types to include DROP statements for
- o 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

The **Debug tab** of the Options Editor lets you specify operational preferences for the Debugger.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the Debugger, see Embarcadero SQL Debugger.

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
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, a compilation is performed before debugging. During a compilation, Oracle invalidates referencing objects. For example, if procedures A2 and A3 both reference procedure A1, debugging procedure A1 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

The **Debug tab** of the Options Editor lets you configure default directories when performing certain operations. You can set the default directories for:

- Wizard operations
- Report generation
- Schema extraction
- HTML templates for customizing reports

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

The table below describes the options and functionality on the **Directories** tab. The %APPDATA%, %MYDOCS%, and %PROGFILES% placeholders represent the location of the Application Data, My Documents, and Program Files folders relevant to your operating system :

Directories Options

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.	*%APPDATA%*\Embarcadero \RapidSQL\DefFiles
HTML Reports	Specifies the name and location of the default directory for the output from generated HTML reports. For information on generating reports, see Report.	*%APPDATA%*\Embarcadero \RapidSQL\Report
Schema Extraction	Specifies the name and location of the default directory for placing the output from schema extraction operations.	*%MYDOCS%*\Embarcadero\ RapidSQL\Extract
HTML Template	Specifies the name and location of the default directory where the HTML template on which to base HTML reports can be found. This feature lets you customize your HTML reports.	*%PROGFILES%*\Embarcader o\RSQL*<version identifier>*\HtmlTpl
User SQL Scripts	Specifies the name and location of the default directory for SQL Scripts.	*%MYDOCS%*\Embarcadero\ RapidSQL\UserSQLScripts
Job Config Files	Specifies the location of any ETSQLX job configuration files you have set on your local machine. For more information, see ETSQLX Command Line Utility.	*%MYDOCS%*\Embarcadero\ RapidSQL\Directories\ETS QLXJobCfg
#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 Datasources. For more information on compiler directives, see SQL Preprocessing: #define and #include.	*%MYDOCS%*\Embarcadero\ RapidSQL\Directories\Inclu deFiles

Explorer Options

The Explorer options are available on the following Options Editor tabs:

- Explorer Options - Main Tab
- Explorer Options - Organization Tab
- Explorer Options - Limits Tab
- Explorer Options - Browsers Tab

Explorer Options - Main Tab

The Options Editor's **Explorer** tab controls the appearance and behavior of the Datasource Navigator.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For related information, see Using the Datasource Navigator.

Use the following table as a guide to modifying settings on this tab

Setting	Description
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 on the next startup.
Column filters use regular expressions by default	Lets you use regular expressions when defining criteria for display of database objects in the right-hand pane of the Explorer.
Display Explorer on startup	Activates the Datasource Explorer on startup. For more information, see Using the Datasource Navigator.
Landing Page View	Enables landing page views on a DBMS-by-DBMS basis. Landing pages display informative metadata that gives you a quick overview of your server/database.

Explorer Options - Organization Tab

The Options Editor's Explorer > Organization tab controls the appearance and behavior of the Datasource Navigator.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For related information, see Using the Datasource Navigator.

Use the following table as a guide to modifying settings on this tab

Setting	Description
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 session. For details, see Basic Viewing Options in the Navigator.
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 Filtering in the Navigator. You can change the **Show System Objects** and **Show Only My Objects** settings manually during a session. For details, see Basic Viewing Options in the Navigator.

Explorer Options - Limits Tab

The Options Editor's **Explorer > Limits** tab controls the appearance and behavior of the Datasource Navigator.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For related information, see Using the Datasource Navigator.

While the DBMS is queried as necessary when you navigate the node hierarchy, it also caches nodes for the purpose of quickly repopulating if you move away from and then back to a node. For example if you open a **Tables** node and then an **Indexes** node and then again select the **Tables** node, the cached **Tables** sub-nodes are displayed.

The following setings are available

Setting	Description
Cache Expiration Time	The duration a node is cached. Selecting a node after this expiration interval results in a fresh population.
Maximum number of children to display per tree node	Lets you place an upper limit of the number of nodes to display per node.

Explorer Options - Browsers Tab

The Options Editor's Explorer > Browsers tab controls the appearance and behavior of the Database Navigator

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Use the following table as a guide to modifying settings on this tab

Interface Element	Setting	Description
Default Owner	All Owners	Indicates that the Browser Window should default to displaying database objects in the browsers for all owners in the database.
	Current User	Indicates that the Browser Window should default to displaying database objects owned exclusively by the current user.
	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.
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.	
Browser colors	Lets you specify foreground and background colors for object types displayed in the Browser	

General Options

The **General** tab of the Options Editor lets you specify general application options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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

Tab	Option	Description	Default
Main tab	**Confirm on Exit**	Ensures that a prompt is issued 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 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** - Views based on the user's role will be used. That is, users with DBA privileges will use DBA views. **DBA Views** - DBA views are always used. **All Views** - ALL views are always used. **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 Registering Datasources.	
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 (Rapid SQL)

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

Grid Properties (Results window) Options

The **Grid Properties** tab of the Option editor lets you set preferences for the layout and appearance of the grid in an Results window.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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
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 Options Editor tabs:

- ISQL Options - Main Tab
- ISQL Options - Oracle Tab
- ISQL Options - DB2 Tab
- ISQL Options - Sybase Tab
- ISQL Options - SQL Server Tab
- ISQL Options - Sybase IQ Tab
- ISQL Options - MySQL Tab
- ISQL Options - DB2 OS390 Tab
- ISQL Options - InterBase Tab (Rapid SQL)
- ISQL Options - Editor Tab
- ISQL Options - Auto format Tab
- ISQL Options - Code Assist Tab

ISQL Options - Main Tab

The Options Editor's **ISQL** tab and its sub-tabs control the appearance and behavior of the SQL Editor.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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. This value is used 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 the file type is not recognized.
Query Plan Layout	Sets the default orientation of a graphical query plan. For more information, see Setting up the Execution Environment with Query options.

ISQL Options - Oracle Tab

The Options Editor's ISQL > Oracle tab controls the appearance and behavior of the SQL Editor when executing against an Oracle environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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 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 Viewing a Tree-based or Graphical Query Plan (Oracle).
Load Query Options	If this option is not enabled, a set of default query options is loaded that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override 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 Setting up the Execution Environment with Query options.

ISQL Options - DB2 Tab

The Options Editor's ISQL > DB2 tab controls the appearance and behavior of the SQL Editor when executing against aDB2 LUW environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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 Setting up the Execution Environment with Query options.
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.
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, those tables continue to be used. 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, a set of default query options is loaded that customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - Sybase Tab

The Options Editor's ISQL > Sybase tab controls the appearance and behavior of the SQL Editor when executing against a Sybase ASE environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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, a set of default query options is loaded. That set of query options customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - Sybase IQ Tab

The Options Editor's ISQL > Sybase IQ tab controls the appearance and behavior of the SQL Editor when executing against a Sybase IQ environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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 Setting up the Execution Environment with 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, a set of default query options is loaded. That set of options customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - SQL Server Tab

The Options Editor's ISQL > SQL Server tab controls the appearance and behavior of the SQL Editor when executing against a SQL Server environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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 Setting up the Execution Environment with 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, a set of default query options is loaded. That set of options customize your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - MySQL Tab

The Options Editor's ISQL > MySQL tab controls the appearance and behavior of the SQL Editor when executing against a MySQL environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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, a set of default query options is loaded. That set of options customizes your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - DB2 OS390 Tab

The Options Editor's ISQL > DB2 OS390 tab controls the appearance and behavior of the SQL Editor when executing against a DB2 z/OS environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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, a set of default query options is loaded. That set of options customizes your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - InterBase Tab (Rapid SQL)

The Options Editor's ISQL > InterBase tab controls the appearance and behavior of the SQL Editor when executing against an InterBase/Firebird environment.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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, a set of default query optionsis loaded. That set of options customizes your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - Teradata Tab

The Options Editor's ISQL > Teradata tab controls the appearance and behavior of the SQL Editor when executing against a Teradata environment.

Note: Teradata datasources support is intended as a technical preview. Minimal functionality is provided. For information on technical previews, see the **Read Me** at http://docs.embarcadero.com.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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, a set of default query options is loaded. That set of options customizes your execution environment, and periodically sends those settings to the server. Enabling this option and specifying an XML file, forces load of a previously saved query options file each time an ISQL windows opens against this DBMS platform. This lets you override the 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 Setting up the Execution Environment with Query options.

ISQL Options - Editor Tab

The Options Editor's ISQL > Editor tab controls the appearance and behavior of the SQL Editor window.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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 the ISQL Editor window should be maximized on opening. 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.	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, you are prompted 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: In Rapid SQL, `%APPDATA%\RapidSQL\Backup\`. Backup files use a naming convention of the form ~ET*xxxx*.tmp.. For a listing of backup file names and the path of the file to which they correspond, see the registry key: In Rapid SQL, `HKEY_CURRENT_USER\Software\Embarcadero\RapidSQL\`*version*`\Backup`	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
	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.	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, you can make changes to the **Auto Format** tab. of the **ISQL** pane.

The Options Editor's ISQL > 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.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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.
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

Code assist options are available on the following tabs:

- ISQL Options - Code Assist - Main Tab
- ISQL Options - Code Assist - Advanced Tab

ISQL Options - Code Assist - Main Tab

The Options Editor's ISQL > Code Assist tab lets you activate and select options for 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.
- **Hyperlinks**: lets you invoke object actions against objects referenced in a script. For more information, see Hyperlink Object Actions.
- **Object metadata display:** lets you view Connect server metadata for referenced objects. For details, see Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor.

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.
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.

DB Action Menu	**Enable DBAction menu in the editor**	Enables the Hyperlinks feature.
	Show DB Actions in submenu	If unselected, Hyperlink menu actions are available as first level options in the context menu. If selected, Hyperlink menu actions are available as a **DB Actions** submenu.
	Key used in conjunction with right-mouse	Lets you select the key (CTRL, SHIFT, or CTRL+SHIFT) used in conjunction with right-clicking, that activates the object action menu against a supported object type reference in a script. The default keystroke combination is CTRL+right-click.
Metadata	Enables display/use of metadata for referenced objects, obtained from a Connect server repository, in the following user interface elements: **tooltips**, **window gutter**, and **code complete proposals**. You can also have metadata added as comments when formatting code. For detailed information on setting up and using this feature, see Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor.	
Performance	Enables/disables the ISQL Performance feature. For more information see ISQL Performance.	
Cache lifespan in minutes	Allows you to select the update timing for the ISQL Performance feature. ISQL performance updates the cache information with the sample rate selected. For more information, see ISQL Performance.	
Restore defaults	Restores all settings on the tab to the original defaults.	

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

ISQL Options - Code Assist - Advanced Tab

Coding assistance and error detection features such as Hyperlink object actions, Code Complete, and Semantic Validation rely on:

- Parsing of SQL entered in the SQL Editor into keywords, object name references, data, and so on
- A database object index generated for a connected datasource. On the first connection to a datasource, an index of specified object types is generated, with an imposed, configurable, upper limit on the number of objects contained in the index. On each subsequent connection, the index is updated to reflect any additions or changes on the datasource.

Note: The object index is stored in a persistent database with a database path of **%TEMP%**. With that in mind, the index may be deleted by the system administrator or automated administration-related processes.

When coding assistance and error detection features are enabled, the index is queried for the names of objects referenced in your scripts queries. This information is then used for tasks such as recognizing object types, detecting misspelled object names, and populating suggestion lists.

The Options Editor's ISQL > Code Assist > Advanced tab lets you force parsing, if required, and customize how the index is generated. Since there is performance overhead associated with generation and usage of the index, you can control the properties and usage of the index.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

The table below describes the options and functionality on the ISQL > Code Assist > Advanced Tab of the Options Editor:

Option	Description
Scan entire file for context	This control is enabled only if the ISQL > Code Assist tab's **Enable Real-time syntax checking** check box is deselected and the **Severity levels for semantic validation problem**s is set to IGNORE. When this control is enabled, SQL Editor content continues to be parsed. That is, if you have disabled real-time syntax checking and effectively disabled semantic validation, you can still use the Hyperlinks feature.
Limit scanning to files smaller than	Lets you place an upper limit on the size of files parsed for Code Assist features.
Object types to be indexed	Lets you select specific object types (aliases, synonyms, functions, packages, procedures, tables, or views) that will be included in the object index. While this can be used to reduce the index size and therefore improve performance, Hyperlinks, Semantic Validation, and Code Complete functionality associated with object types not included in the index will not be available.
Recycle cache space when index size exceeds	This imposes a soft ceiling on the number of objects stored in the index. After updating the index on connection to a datasource, the total number of objects in the index is compared to this value. If the value is exceeded, the oldest definitions are deleted.
The maximum number of results to return when querying the object index	Specifies the number of results when querying against the cache databases, to improve performance when working with large databases. This limit is applied per object type.
Refresh	Each time you open an ISQL editor window, schema information for the object types supported is obtained 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.

Java Options

Rapid SQL requires JDBC connections. The Options Editor's **Java** tab lets you set or change options that apply to the Java virtual machine (JVM) that is running on the client.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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

LDAP Configuration Options

When using the Discover Datasources feature, you can have an LDAP Server searched for datasources. Similarly, when manually registering a datasource, you can select from server names obtained from an LDAP Server. The **LDAP Configuration** tab of the Options Editor lets you provide the information required to establish a session with the LDAP Server and obtain the required datasource information.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

The table below describes the options and functionality on the **LDAP Configuration** tab:

Group	Option	Description
	Enable LDAP	Enables LDAP lookup. This check box must be selected in order for the Discover Datasources feature to search an LDAP Servers or for server names obtained from an LDAP server to be available when manually registering datasources.
Server	**Host Port**	The FQDN or IP address of the LDAP server.
	Search Base DN and Suggest	Lets you provide the distinguished name to be used as the search base. The **Suggest** button is only enabled after you select an authentication type and provide details, as required. It queries the server for possible DNs and populates the combo box with the results.
Authentication	**Anonymous Login**	Selecting this check box directs Rapid SQL to send LDAP requests without performing a bind. This also disables the other **Authentication** controls.
	User, **Password**, and Remember	Let you provide the credentials used when sending LDAP requests.
Options	**Size Limit** and Time Limit	Lets you specify the maximum number of entries returnable and the maximum time, in seconds, to wait for the search to complete.
	Protocol Version	The LDAP protocol version.

Topics

- Manually Registering or Editing Datasources
- Automatically Discovering Datasources

Logging Options

After opening the Options editor (see Specifying Application Preferences and Feature Options), you can make changes to the **Logging** tab. The Logging tab lets you set defaults that specify the behavior and placement of SQL, output and trace file logging.

SQL logging provides an audit trail. You can examine this log to determine the SQL executed to complete a task. The table below describes the SQL Logging options and functionality on the **Logging** tab:

Option	Description
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. You can examine this log to determine the SQL executed to complete a task.
Logfile Path	If you choose to log the SQL generated by the application, specify the drive, directory, and file name.
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.
Truncate	Empties the entire contents of the logfile.

Output 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. The table below describes the output logging options and functionality on the **Logging** tab:

Option	Description
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.
Logfile Path	If you choose to log the server messages generated in the Output window, specify the drive, directory, and file name.
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.
Truncate	Empties the entire contents of the output logfile.

Tracing is typically disabled (**Trace Level** set to **Off**) unless you are trying to diagnose a problem. When enabled, the sequence of trace messages generated by the application, severity specified, is written to a **.log** file. THe file name includes a date stamp.

A trace file is created on each run, one per day. If multiple instances of the same application are running, each instance will also have its own trace file for that day. The trace file will be written to until the OS denies writing to it due to the file being too big or not enough disk space. Unless the **Trace Level** is set to the higher levels such as **Trace** or **Max**, there won't be much in this file normally, so space should not be a problem. Up to 10 days of trace files are kept.

If tracing is turned on in the application (any **Trace Level** other than **Off** is set) the trace file will be locked until the application exits or the **Trace Level** is set to **Off** . The table below describes the Trace File options:

Option	Description	
Trace Path	Fully specify the directory where trace files are to be stored.	
Trace Level	In descending order of severity (and ascending order of total messages logged), **Trace Level** options are **Fatal Errors**, **Errors**, **Warning**, **Info**, **Debug**, **Trace**, and **Max**.	
Truncate	Empties the entire contents of the current trace file.	Not available

MySQL Utilities Options

Rapid SQL integrates with MySQL utilities. For access to 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.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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

To make use of features that integrate with Oracle utilities, you need to specify their location on the **Oracle Utilities** tab of the Options Editor. You can use the Oracle Export, Import Utilities, and SQL * Loader.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

The table below describes the options and functionality on the **Oracle Utilities** tab:

Option	Description
Data Pump Export	Specifies a path for the Oracle **expdp.exe** utility.
Data Pump Import	Specifies a path for the Oracle **impdp.exe** utility.
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.

Query Builder Options

The Options Editor's **Query Builder Options** tab lets you specify operational and execution preferences for the Query Builder. 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.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the Query Builder, see Query Builder.

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. 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 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

The Options Editor's **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.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on generating reports, see Report.

Results (ISQL) Options

The Options Editor's **Results** tab lets you specify operational and appearance details for the SQL Editor.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

Note: For information on using the SQL Editor, see Coding Environments/Editors.

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, result sets are not saved.
	Result Set Options	**Default Rowcount** - Lets you limit the number of rows returned to the result window of the ISQL window (default 0). **Sybase ASE 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.
	Format	Standard Grid - Displays all result sets in a standard grid format. Result sets are displayed in grid format in SQL Editors that are opened after you have selected this option. This has no effect on ISQL Editors that are already open. HTML - Displays all result sets as HTML tables. Result sets are displaedin HTML format in ISQL Editors that are opened after you have selected this option. This has no effect on SQL Editors that are already open. ASCII Text - Displays all result sets as ASCII Text. Result sets are displayed in ASCII Text format in ISQL Editors that are opened after you have selected this option. There is no effect on SQL Editors that are already open. Grid Font and Printer Font buttons - Open a Font dialog, 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

The Options Editor's **SMTP Mail Options** tab lets you specify outgoing notification e-mail message options.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature 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.	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
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	An internet connection is established at launch.	Selected

Team Server 2016 Options

The **Team Server** tab of the Options Editor integrates Rapid SQL with Embarcadero Team Server 2016, a data governance metadata repository.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

The table below describes the options and functionality on the **Team Server** tab:

Option	Description
Use Team Server	Enables use of the Embarcadero Team Server 2016 repository.
Team Server URL	The login URL of the Team Server 2016 repository.
Moving datasources	Action performed when moving a datasource. **Note:** This option is available only if you are using Team Server based datasource catalog.

Moving Datasources

When you moved a datasource through drag-and-drop into a group or from one group to another, three options are available:

- **Prompt always:** When moving a datasource, a window is prompted asking if you want to move the datasource or move it leaving a copy in the old group.
- **Move Always:** When moving a datasource, no copy will be left in the old group.
- **Don't remove from old group Always:** When moving a datasource, a copy will be left in the old group.

Note: These controls can be set or disabled using registry entries. For details, see Locking Down Features Using the Registry.

For more information on repository-based resources, see Team Server 2016 Support.

Version Control Options

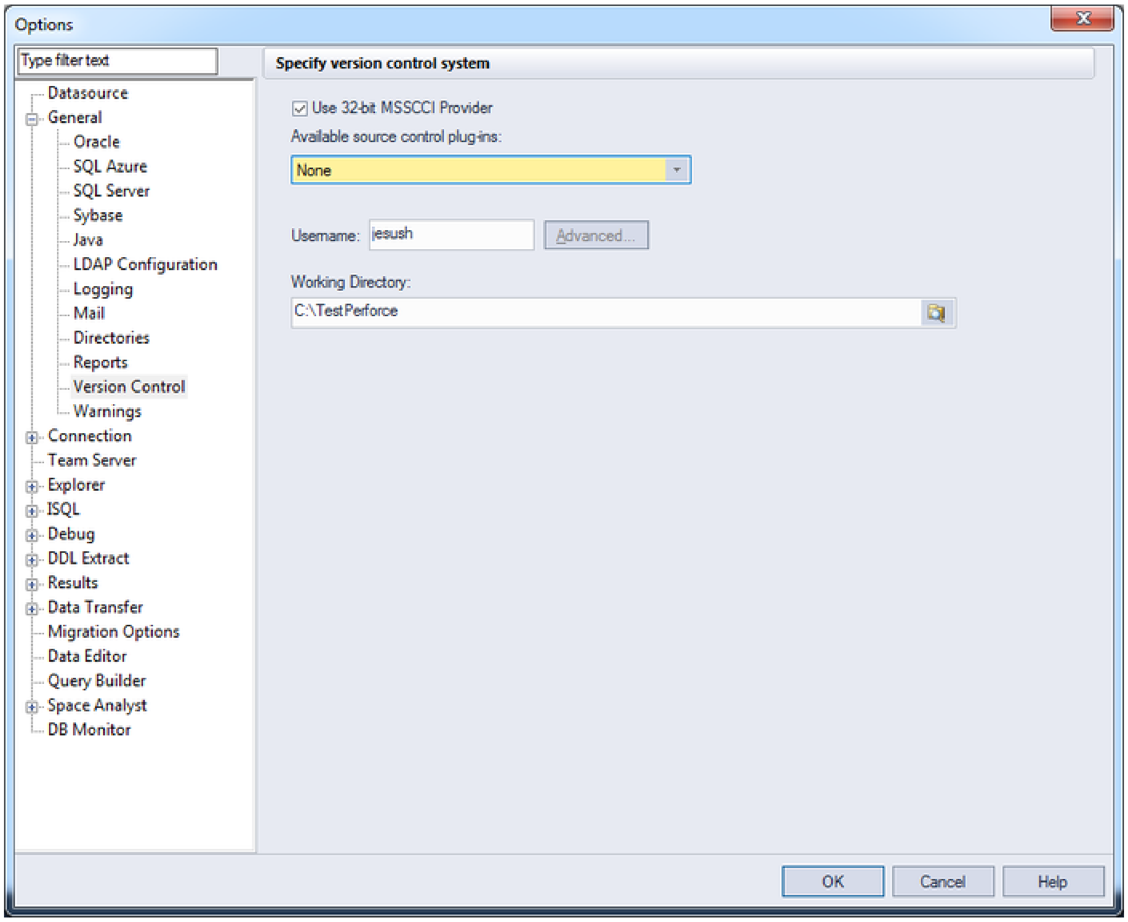
The Options Editor's **Version Control tab** lets you select which version control system you want to use as the underlying version control system.

For configuring the Version Control functionality:

1. Open the **Options Editor** Clicking **File** menu and selecting **Options**.
2. Select **General** and **Version Control** to open the **Version Control Tab**.
3. Select the Version Control System in the **Available source control plug-ins** drop-down list.

Note: If you are using a 64 bit version of Rapid SQL with a 32 bit MSSCCI version control provider, the **Use 32-bit MSSCCI Provider** checkbox must be checked.

4. Select the **Username** for the Version Control System.
5. Specify the local **Working Directory**
6. Click **Ok**



Specify version control system

☑ Use 32-bit MSSCCI Provider

Available source control plug-ins: None

Username: jesush

Working Directory: C:\TestPerforce

Note: After Adding a project or file to the version control, the Version Control System launches its Configuration Wizard to start to work with it and add that project or file to the version control.

Topics

- Version Control

Warnings Options

You can have warning messages issued to users whenever improper operations are attempted in a database. Warning messages differ by platform. The Options Editor's **Warnings** tab lets you select the DBMS-specific events that generate warnings.

Note: For information on opening the Options Editor, see Specifying Application Preferences and Feature Options.

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
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

Locking Down Features Using the Registry

Administrators can restrict certain features on individual machines through registry settings. Those settings are stored in the following location:

- In Rapid SQL, `HKLM\Software\Embarcadero\Rapid SQL\`*optional version identifier*`\Admin`

The following general datasource and connection settings are available:

Name (and Type)	Description and Supported Values
RestrictedPlatforms (REG_SZ)	A comma-separated list (**db2=0,mssqlserver=1**, for example) dictating DBMS platform support on the machine. Valid names for the *name=value* pairs are **db2**, **os390**, **mssqlserver**, **mysql**, **oracle**, **sybase**, **interbase,** **mssqlazure**, **odbc**, **jdbc**, and **sybaseiq**. Any non-zero value for a **name=value** pair results in that platform not being unavailable on that machine. A zero value (the default) results in the platform being available on that machine.
DisableAutoConnect (REG_DWORD)	When set to any non-zero value, users cannot use the auto-connect feature when registering or logging in to a datasource.
SharedDatasourceCatalogPath (REG_SZ)	Providing a path to the network shared datasource catalog forces Rapid SQL to use that datasource catalog storage option. Registry-based and file-based catalog storage will be unavailable on this machine.

The following settings control configuration options on the **Team Server** page of the Options Editor and the **Embarcadero Team Server Log In** dialog. For details, see Team Server 2016 Options and Setting up and Connecting to Team Server 2016.

Name (and Type)	Description and Supported Values
ConnectURL (REG_SZ)	The full Embarcadero Team Server 2016 URL that the user must use if Team Server 2016 is enabled. If a value is specified, the **Team Server URL** control on the **Team Server** page of the Options Editor is set to read-only.
UseConnect (REG_DWORD)	If a value of 1 (TRUE) is specified, use of Embarcadero Team Server 2016 is enabled. If a value of 0 (FALSE) is specified, use of Team Server 2016 is disabled. If either value is specified, the **Use Team Server** control on the **Team Server** page of the Options Editor is disabled.
AllowKeepMeSignedIn (REG_DWORD)	If a value of 1 (TRUE) is specified, the **Keep me signed in** control on the **Embarcadero Team Server - Log In** dialog is is set to an unchecked state and the control is disabled.

UseConnect and **AllowKeepMeSignedIn** is type REG_DWORD (legal values for both are 0 for FALSE and 1 for TRUE).

Administrators not wanting to use these settings from the HKLM root key, can use the same options in HKCU root key:

- In Rapid SQL, `HKCU\Software\Embarcadero\Rapid SQL\`*optional version identifier*`\Admin`

If the option is specified in both locations, HKLM takes precedence over HKCU.

Datasource and Server Management

After installation, you must set up datasource definitions to establish reusable connections to your database servers. The following topics describe the function of datasources, the process of establishing datasource connections, and the related options provided:

Registering Datasources	Provides instructions on how to set up datasource definitions, manually and automatically.
Connecting to Datasources	Provides instructions on how to how to connect to datasources, manually and automatically.
Trouble-shooting Client Setup	Provides advice on solving common connectivity problems.
Using the Datasource Navigator	Describes the user interface mechanism that lets you browse among datasources, drill down to the database object level, and initiate datasource or object actions.
Using the Manage Datasources Utility	Describes a utility offering shortcuts to datasource options.

Registering Datasources

A datasource is the connection profile that lets you have access to a database, and, therefore, to your data. So, when you register a datasource, you're telling the application how to get at the database(s) you want to operate on. In fact, all database activities are performed through datasources, so you need a datasource profile for each database instance (or database server) in your enterprise. Each connection profile consists of basics such as unique name, DBMS-specific connection parameters, and user credentials.

Collectively, datasource definitions are stored in a datasource catalog. The catalog can be registry-based, local file-based, or network shared file-based, with each method addressing particular advantages. For details, see Specifying a Datasource Catalog Storage Option.

You register datasources using the following techniques:

Manual registration	You can register datasources individually. For details, see Manually Registering or Editing Datasources.
Automatic discovery	You can have search computers or your network for datasources. For details, see Automatically Discovering Datasources.
Import/Export	You can export and import sets of datasource registrations. For details, see Importing and Exporting Datasource Definitions.
Embarcadero Team Server 2016	Integrating with Team Server 2016 lets you work with datasource definitions on a data governance metadata repository. For details, see Team Server 2016 Support.

As an aid in maintaining your datasource registrations, you can quickly view basic registration details of a datasource. For details, see Viewing Datasource Properties.

Lastly, when a datasource registration becomes obsolete, you can unregister the datasource. For details, see Unregistering Datasources.

Specifying a Datasource Catalog Storage Option

A Datasource Catalog is a collection of defined datasources. On startup, a Datasource Catalog is loaded 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.

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.

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, the definitions are imported 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.

Note: On individual computers, administrators can force use of the network shared option. For details, see Locking Down Features Using the Registry.

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 Registering Datasources.

Keep in mind that changes made with **Network shared datasource catalog** in effect, are lost on shutdown.

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.

Manually Registering or Editing Datasources

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

To register a new datasource or edit an existing datasource

1. Use one of the following actions:

    - To register a new datasource, from the **Datasource** menu, select **Register Datasource**.
    - To register a new datasource based on an existing definition, select the existing datasource in the Navigator and then from the **Datasource** menu, select **Register Like**.
    - To edit registration information, on the Navigator, right-click a datasource and select **Edit**.

2. Use the following topics as a guide to providing the information required to register or edit the datasource:

    - Selecting Database Type/Connectivity Options When Registering Datasources
    - Providing Connection Information When Registering Datasources
    - Providing Security Parameters When Registering Datasources
    - Providing Custom Driver Properties When Registering Datasources
    - Specifying Datasource Properties When Registering Datasources
    - Specifying Oracle Views Configuration When Registering Datasources
    - Specifying Group and Category Details When Registering Datasources

3. When ready, click **Finish** and respond to the prompt to connect immediately.

Note: The **Test Connection** button can be used when creating or editing a datasource to verify that a connection can be established. Keep in mind that the information currently provided in the wizard is used to test the connection and therefore connection information, credentials, and custom JDBC driver properties if appropriate, must be valid.

Selecting Database Type/Connectivity Options When Registering Datasources

The **Datasource Registration** Wizard's **Database Type** panel lets you specify the basic connectivity method for the datasource you are creating.

Prior to registering datasources, you should be familiar with the connectivity options available for each DBMS platform.

- For Rapid SQL system requirements and details on DBMS product support, see the **Read Me** at http://docs.embarcadero.com/products/rapid_sql/.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

Note: On individual computers, administrators can restrict the available DBMS platforms available using registry settings. For details, see Locking Down Features Using the Registry.

To use Embarcadero's default drivers requiring dedicated client software installed on the client machine

1. From the **Available database types** list, select one of the following:
   - **IBM DB2 Universal Database**
   - **Microsoft SQL Server**
   - **MySQL**
   - **Oracle**
   - **PostgreSQL Servers**
   - **Sybase Adaptive Server**
   - **Sybase IQ**
   - **Teradata**
   - **InterBase/Firebird (Rapid SQL only)**

If there is currently a DBMS node selected in the Navigator, that database type is automatically selected.

Note: If you are connected to an Embarcadero Team Server 2016 installation, you can select an entry of the selected database type from the **Related Team Server Data Source** list. This prepopulates the wizard with definitions/settings from the Connect server for that datasource and sets up the new datasource definition to be attached to the Team Server 2016 datasource. When an item is selected from the **Related Team Server Data Source** list, clicking **Unrelate** removes the Team Server 2016 settings/definitions from the current definition and cancels attachment to the Team Server 2016 datasource. When no item is selected from the **Related Team Server Data Source** list, clicking **Relate** uses the (alphabetically) first matching Team Server 2016 datasource of the selected database type to populate/attach, and if there are no matching definitions, opens a dialog to let you select a definition from Team Server 2016. For more information on Team Server 2016 datasources, see Team Server 2016 Support.

2. Click **Next**.

To specify a custom JDBC driver

1. From the **Available database types** list, select **Generic JDBC**.
2. Click **Next**.

To specify a generic ODBC driver or to use an ODBC connection

1. From the **Available database types** list, select **Generic JDBC**.
2. Click **Next**.

Providing Connection Information When Registering Datasources

Connection information is provided on a DBMS platform basis. The following topics describe platform-by-platform descriptions of the required and optional connection information:

- Providing DB2 Connection Information When Registering Datasources
- Providing InterBase/Firebird Connection Information When Registering Datasources
- Providing MySQL Connection Information When Registering Datasources
- Providing Oracle Connection Information When Registering Datasources
- Providing PostgreSQL Connection Information When Registering Datasources
- Providing SQL Server Connection Information When Registering Datasources
- Providing Sybase ASE Connection Information When Registering Datasources
- Providing Sybase IQ Connection Information When Registering Datasources
- Providing Teradata Connection Information When Registering Datasources
- Providing Connection Information When Registering a Generic JDBC Datasource
- Providing Connection Information When Registering a Generic ODBC Datasource

Providing DB2 Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** panel lets you provide details required to establish a connection to a DB2 datasource. Available options depend on the **Registration Type** value selected.

The following table describes these settings:

Registration Type	Option/Description
Simple	Host/Instance - The network name of the machine hosting the datasource. Port - This defaults to 50000 but can be overwritten. Database - The name of the database this user should connect to. **Datasource Name** - This is the name of the datasource as it will appear in the Navigator.
Use alias from IBM client	**Server** - Select from the names of servers defined in the IBM client. **Datasource Name** - This is the name of the datasource as it will appear in the Navigator.
Use Data from the LDAP Server	This option is only enabled if LDAP is enabled and configured on the Options Editor's **LDAP Configuration** tab. For details, see LDAP Configuration Options. **Server** - Select from the names of servers obtained from the LDAP server Host/Instance - The network name of the machine hosting the datasource. Port - This defaults to 50000 but can be overwritten. Database - The name of the database this user should connect to. **Datasource Name** - The datasource as it will appear in the Navigator.

Keep the following in mind when registering a DB2 datasource:

- The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.
- Consult DB2 documentation for detailed information on general LDAP support, supported LDAP servers, and client configuration before trying to register LDAP-specified datasources.. For assistance, see Accessing Third Party Documentation.
- For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing InterBase/Firebird Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to an InterBase or Firebird datasource. The following table describes these setting:

Option	Description
Host name	The name of the machine where the database resides.
Port	Optionally, provide a port number.
Character Set	The default character set used for this connection.
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 Navigator tree.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing MySQL Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a MySQL datasource. The following table describes these setting:

Option	Description
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.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing Oracle Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to an Oracle datasource. Available options depend on the **Registration Type** value selected.

The following table describes these setting:

Registration Type	Options/Description
Simple	**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** -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.
Use Connection Descriptor	**Connect Descriptor** - provide a TNS Connect String, a connect identifier that defines the parameters that need the Oracle Net Service to connect to a database service. **Datasource Name** - The field is automatically populated with the host name, but you can change it.
Use TNS Names alias	You can specify a non-default TNSNAMES.ORA file using the Options editor. For details, see Connection Options - Oracle. Oracle Alias - The dropdown is populated with server names from the TNSNAMES.ORA file. Select an Oracle alias. **Connect Descriptor** - A read-only field that displays the TNS Connect String obtained from the alias definition from the TNSNAMES.ORA file. **Datasource Name** - The field is automatically populated with the host name, but you can change it.
Use Data from the LDAP Server ()	This option is only enabled if LDAP is enabled and properly configured on the Options Editor's **LDAP Configuration** tab. For details, see LDAP Configuration Options. Oracle Alias - The dropdown is populated with the names of servers obtained from the LDAP server specified on the Options Editor's **LDAP Configuration** tab. **Connect Descriptor** - A read-only field that displays the TNS Connect String obtained from the alias definition from the LDAP server. **Datasource Name** - The field is automatically populated with the host name, but you can change it.

Keep the following in mind when registering an Oracle datasource:

- You can register an Oracle ASM instance directly using the Datasource Registration Wizard. No Datasource Navigator tree is available when connecting to the ASM instance. The ISQL Editor is available with this type of connection.
- Consult Oracle documentation for detailed information on general LDAP support, supported LDAP servers, and client configuration before trying to register LDAP-specified datasources.. For assistance, see Accessing Third Party Documentation.
- For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing PostgreSQL Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a PostgreSQL datasource. The following table describes these settings:

Option	Description
Host	The host name for the PostgreSQL Server.
Port	The port that the PostgreSQL server is listening on.
Database	The name for the database to connect to.
Datasource Name	This field automatically populates with the host name. You can change it to whatever you want.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing SQL Server Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a SQL Server datasource. The following table describes these setting:

Note: If you are registering a named instance on a server, you are restricted to use of the **Use Network Library Configuration/Alias** method below. Prior to registering the datasource, use the SQL Server Configuration Manager to create an alias to the instance, making it available as an **Alias** selection on the **Connection Information** panel.

Option	Description
Simple	**Protocol** (**TCP/IP** or **Named Pipes**) - You need to select one or the other to register a SQL Server database. **Host** - the host name. **Port** - 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** - 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.
Use Network Library Configuration	**Use Encryption** - enables multiprotocol encryption. **Alias** - Select an alias **Protocol** (**TCP/IP** or **Named Pipes**) - You need to select one or the other to register a SQL Server database. **Host** - Displays the host name. **Port** - 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** - 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.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing Sybase ASE Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a Sybase ASE datasource. The following table describes these settings:

Registration Type	Option/Description
Simple	**Use SSL Encryption** - If this option is selected, the JDBC connection will be established using SSL encryption **Host** - Manually enter the name of the host. **Port** - There is no default for this optional field. **Default DB** - Optionally give the default name for the database. **Datasource Name** - This field automatically populates with the server name. You can change it to whatever you want.
Use alias information from the SQL.INI FILE	**Sybase Server** - Select a server from the dropdown **Host** - Automatically populated when you select an alias. **Port** - There is no default for this optional field. **Default DB** - Optionally give the default name for the database. **Datasource Name** - This field automatically populates with the server name. You can change it to whatever you want.
Use data from the LDAP server	**Use SSL Encryption** - The connection will be established using SSL encryption. **Sybase Server** - Select a server from the dropdown **Host** - Automatically populated when you select an alias. **Port** - There is no default for this optional field. **Default DB** - Optionally give the default name for the database. **Datasource Name** - This field automatically populates with the server name. You can change it to whatever you want.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing Sybase IQ Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a Sybase IQ datasource. The following table describes these settings:

Option	Description
Host	The host name for the Sybase IQ Server.
Port	The port that Sybase IQ is listening on.
Database	The name for the database to connect to.
Datasource Name	This field automatically populates with the host name. You can change it to whatever you want.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing Teradata Connection Information When Registering Datasources

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a Teradata datasource. The following table describes these settings:

Option	Description
Host	The host name for the Teradata Server.
Port	The port that the Teradata server is listening on.
Database	The name for the database to connect to.
Datasource Name	This field automatically populates with the host name. You can change it to whatever you want.

Note: The Options Editor lets you indicate preferences, select options, and provide details not available with the Datasource Registration Wizard. For details, see Connection Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing Connection Information When Registering a Generic JDBC Datasource

The **Datasource Registration** Wizard's **Connection Information** lets you provide basic information details required to establish a connection to a generic JDBC datasource.

Note: For requirements and functionality restrictions for generic JDBC connections, see Generic JDBC/ODBC Connectivity.

The following table describes the settings on this panel:

Option	Description
JDBC driver to use	
Connection URL	If you are defining a datasource using a custom JDBC driver, this field is automatically populated with the value you entered in the **Template Connect URL for new data source** field. For more information. see Providing details for a new custom JDBC driver. Type a valid connection URL or edit the URL template to provide a valid connection URL.
Datasource Name	This is the name that will appear in the Datasource Navigator.

Note: In addition to basic connection properties, you can also specify provider-specific JDBC properties. For details, see Providing Custom Driver Properties When Registering Datasources.

Take one of the following actions:

1. To specify a JDBC driver that you have not used before, click **Manage** to register a new driver. For more information, see Providing details for a new custom JDBC driver.
2. To use a custom JDBC driver you have already registered, select that driver from the **JDBC driver to use** dropdown.

Providing details for a new custom JDBC driver

When registering a datasource, if you select a generic JDBC connection, you must specify the installed JDBC driver to be used. If you have not used that driver, you can register that driver while defining a datasource.

Note: Before setting up a custom driver, consult the documentation for that driver to obtain details such as the driver class and connection URL for that driver type.

To specify a new custom JDBC driver

1. Open a datasource registration dialog. For details, see Manually Registering or Editing Datasources.
2. From the **Available database types** list, select **Generic JDBC**.
3. On the **Connection Information** page, click **Manage**. The **JDBC Driver Editor** opens.
4. In the **Name** field, type a name or description for the new driver.
5. Click the **Add** button associated with the **Driver Archives** list and use the **Open** dialog to locate and select the library or archive for the driver.
6. In the **Class** field, type the driver class.

7. In the **Template Connect URL for new data source** field, provide a connection URL template. There are no constraints on the value that you enter. When providing details on the **Datasource Registration** Wizard's **Connection Information** panel, the template will automatically be entered in the **Connection URL** field, letting you provide actual values for any placeholders you provide in the URL template entered here.

Providing Connection Information When Registering a Generic ODBC Datasource

The **Datasource Registration** Wizard's **Connection Information** lets you provide details required to establish a connection to a generic ODBC datasource.

Note: For requirements and functionality restrictions for generic ODBC connections, see Generic JDBC/ODBC Connectivity.

The following table describes the settings on this panel:

Use Connect String	Option/Description
Unselected	**Datasource Name** - Provide the name that will appear in the Datasource Navigator for this datasource. **DSN** - Provide the DSN for the datasource.
Selected	**Datasource Name** - Provide the name that will appear in the Datasource Navigator for this datasource. **Connect String** - Provide the full connection string for the datasource.

Note: Consult the documentation for the ODBC driver and the datasource you are connecting to for details on DSN and connect string requirements.

Providing Security Parameters When Registering Datasources

The **Datasource Registration** Wizard's **Security Parameters** panel lets you provide user credentials and specify other DBMS-specific security settings. The following table describes these settings and their DBMS availability:

Option	Description
User ID	The User ID that Rapid SQL will use to connect to the datasource.
Password	The password associated with the User ID.
Connect As (Oracle)	When relevant, choose the appropriate user/administrator level: NORMAL, SYSDBA, or SYSOPER.
Domain (SQL Server)	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. On individual computers, auto connect can be disabled by administrators. For details, see Locking Down Features Using the Registry.
Omit User ID and Password (Generic JDBC and Generic ODBC only)	Selected, no user ID or password are passed when connecting to the datasource.
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 ASE)	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 Editor Options.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Providing Custom Driver Properties When Registering Datasources

The **Datasource Registration** Wizard's **Custom Driver Properties** panel lets you specify optional, provider-specific connection string keywords or properties beyond those offered on other panels of the wizard.

Note: Consult driver/client documentation for details on supported driver properties. For a listing of supported DBMS drivers and clients, see the **ReadMe.htm** file for this release (View > Release Notes).

To add a driver property for this datasource connection

1. Click **New**. An **Add Driver Property dialog** opens.
2. Select a property name from the **Name** dropdown or type the name of the property. The dropdown is automatically populated with the most commonly-used properties for a platform driver. However, all properties (name/value pairs) supported for that driver can be specified.
3. Type the property value in the **Value** field.
4. Optionally, provide a **Description**.
5. Click **OK**.

The dialog closes and the new property's name and value are displayed in the **Property Name**/**Property Value** list.

Use the **Edit** button to initiate editing of a selected property name/value pair. Similarly, use the **Delete** button to remove a selected property name/value pair from the list.

Specifying Datasource Properties When Registering Datasources

The **Datasource Registration** Wizard's **Datasource Properties** panel lets you specify properties used in conjunction with application features. The following table describes these settings and their DBMS availability:

Setting	Description
#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 #include directive, see SQL Preprocessing: #define and #include. 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 information, see Directories Options.
Database device default path (Sybase ASE)	The database device default path is used when creating a new database device.
View in Team Server	If there is an attached datasource on a connected Embarcadero Team Server 2016 installation, this control opens a browser on the Team Server 2016 with the definition displayed.
View Related ER Models	Opens a dialog displaying models related to the current datasource and letting you create and delete relationships.
Edit Related Data Source	Opens a Registration Editor that will let you modify the associated Team Server 2016 datasource definition.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Specifying Oracle Views Configuration When Registering Datasources

The **Datasource Registration** Wizard's **Oracle View Config** panel lets you manually configure 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:

Option	Description
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. Deselecting the **Use DBA View** check box associated with an individual view makes the ALL/USER group version of that view available.
Reset to All	Deselects all **Use DBA View** check boxes, making only ALL views accessible.
Reset to DBA	Selects all **Use DBA View** check boxes, making DBA views accessible.
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.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Specifying Group and Category Details When Registering Datasources

The **Datasource Registration** Wizard's **Datasource Group** panel lets you specify properties that help you organize your datasources. The following table describes these settings and their DBMS availability:

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 labeling 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 Labeling.

For more general information on registering datasources, see Manually Registering or Editing Datasources.

Automatically Discovering Datasources

You can automate registration of datasources by importing datasource definitions from the following locations:

- DBMS configuration files/registry entries
- LDAP Servers
- TOAD product datasource definitions (specifically from TOAD for Oracle, TOAD for Microsoft SQL Server, TOAD for MySQL, TOAD for Sybase (both ASE and IQ))
- An Embarcadero Team Server 2016 repository. For detailed information, see Team Server 2016 Support

For example, if you have a multiple DBMS platform license, the Discover Datasources feature finds all unregistered datasources that you are licensed for. If you have an Oracle only license, Discover Datasources finds all unregistered Oracle datasources.

Use the information in Setting up to Automatically Discover Datasources to ensure that you are ready to use the Doscover Datasources feature.

There are two relevant scenarios for this feature.

For details, see the following topics:

- Discovering Datasources on First Application Startup
- Manually Discovering Datasources

In addition to Auto-Discovering your database servers, datasources are automatically placed within a DBMS-specific database group folder in the Navigator. For more information, see Categorizing Datasources using Color-coding and Labeling.

For related information, see Connecting to Datasources.

Setting up to Automatically Discover Datasources

Some search targets used by the Discover Datasources feature rely on customer-provided information. The Options Editor lets you provide the following:

- The names and locations of configuration files such as the Oracle TNSNAMES.ORA file and the Sybase ASE SQL.INI file. For details, see Connection Options.
- Identification and authentication details of an LDAP Server that can publish datasources. For details, see LDAP Configuration Options.
- Enabling and configuration of an Embarcadero Team Server 2016 connection. For details, see Team Server 2016 Support. In addition, you must be logged in to the Embarcadero Team Server 2016. For details, see Setting up and Connecting to Team Server 2016.

In addition, there are environment details that can impact this feature. For details, see Trouble-shooting Client Setup.

Discovering Datasources on First Application Startup

On first startup, a dialog box displays, giving you the option to Auto-Discover all configured datasources. If you click Yes, the Auto-Discover feature searches the DBMS configuration files on your computer and automatically discovers all the datasources that you are licensed for.

Note: Microsoft SQL Server datasources are registered through a Windows system call to your network. Provide login information (user name and password) the first time you connect to a datasource. For more information, see Disconnecting from a Datasource.

Note: IBM DB2 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 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.

Manually Discovering Datasources

You can discover datasources residing on your system that are not currently registered datasources. The Discover Datasources operation can retrieve a list of discovered datasources, which includes the name of the server or instance, the type of DBMS, and source (DBMS configuration files/registry entries, LDAP Servers, TOAD datasource definitions, Embarcadero Team Server 2016) 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 Datasources**.

The **Discover Datasources** dialog box opens and when the search is complete, displays all discovered datasources. The source (**Local alias**, **Network**, **LDAP**, or **Toad**) and status of each is displayed.

2. Select the check box next to any datasources you want to register.
3. Click **Register**.

Importing and Exporting Datasource Definitions

A datasource catalog is loaded on startup. It is 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 Specifying a Datasource Catalog Storage Option.

Note: On individual computers, administrators can restrict the available DBMS platforms using registry settings. For details, see Locking Down Features Using the Registry.

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

- **Windows Registr**y - 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 a **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, select **Manage Datasources**.

The **Manage Datasources** dialog box opens.

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**.

The **Manage Datasources** dialog box opens.

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 previously 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.

Viewing Datasource Properties

You can view the name, type, version, status and mode of a connected datasource. The **Datasource Properties** dialog 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.

Note: For information on establishing a datasource connection, see Connecting to Datasources.

To view key properties of a connected datasource

1. On the Datasource Navigator, select a datasource with an established connection. For more information, see Using the Datasource Navigator.
2. On the **Datasource** menu, select **Properties.**

The **Datasource Properties** dialog box opens.

Unregistering Datasources

You can unregister datasources when you no longer need them.

Note: Unregistering a datasource does not delete the physical database. It simply removes the datasource definition, and connection information, from the datasource catalog.

1. In the Datasource Navigator, right-click the target datasource and select **Unregister**. For more information, see Using the Datasource Navigator.
2. When prompted to confirm, click **Yes**.

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.

Topics

- Registering Datasources

Connecting to Datasources

There are two general categories of methods for connecting to a datasource.

Automatic	You can set datasources to automatically connect each time you open the application. You can do this when registering a datasource or as an option when explicitly connecting to a datasource. For details, see Registering Datasources and Manually Registering or Editing Datasources
Explicit	If you do not use the automatic connection option for a datasource, you must explicitly connect to a datasource before you can use that datasource.

The following topics provide instruction on explicit connection and disconnection options:

- Connecting to a Datsource From the Navigator
- Connecting to Datasources from an Alphabetical Listing
- Disconnecting from a Datasource

Topics

- Registering Datasources

Connecting to a Datasource From the Navigator

The first time you start the application, you are prompted 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. For details, see Moving Datasources Between Groups.

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.

If you later want to automatically connect your datasources, you can edit the datasource to make that change.

Tip: To configure your datasource to login automatically, refer to Registering Datasources.

To manually connect to a datasource

1. In the Datasource Navigator, right-click a datasource and select **Connect** from the context menu. For details on the Navigator, see Using the Datasource Navigator.

A **Login to...** dialog box opens.

2. Type a **Login ID** and **Password**.
3. Optionally, and only if available, provide **Login As**, **Default Schema ID**, and **Default Function Path** values.
4. Select **Auto Connect** to automatically connect to the datasource in the future.
5. Click **OK**.

When a successful connection is made, datasource display expands to show the top level of nodes available on that datasource.

Connecting with Team Server 2016 Credentials

When connecting to a datasource from Team Server 2016 you can connect using the Team Server 2016 credentials. For that you must create the credentials in the Team Server 2016 application before create Login Credentials.

Connecting to Datasources from an Alphabetical Listing

The **Select Datasource** dialog lets you sequentially connect to a set of datasources chosen from a list.

To connect to one or more datasources

1. On the **Datasource** menu, select **Select**. The **Select Datasource** dialog opens.
2. For each datasource that you want to connect to, select a datasource from the **Datasource** list, and then click **Connect** to connect to the selected datasource.
3. If the last datasource you connect to supports multiple databases, you can optionally select a database from the **Database** list, and then click **OK** to have that database selected.

The datasource display expends to show the top level of nodes available on the datasources you connected to. For more information, see Using the Datasource Navigator.

Disconnecting from a Datasource

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, an attempt to reconnect to a registered datasource, if available, is made.

To disconnect from a datasource

1. In the Datasource Navigator, right-click a datasource and select **Disconnect**. For information on the Navigator, see Using the Datasource Navigator.

Note: If necessary, a dialog box opens, 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.

Trouble-shooting Client Setup

If you are using native Embarcadero drivers to connect to a datasource, improperly configured client software can lead to the following symptoms:

- Failure to Auto-Discover accessible datasources.
- Inability to properly register a datasource
- Failure to manually connect to a registered datasource
- Error messages indicating missing client software files

Note: For details on the various ways to register and connect to datasources, see Datasource and Server Management.

Regardless of the DBMS platform, your first step in trouble-shooting client connectivity is to ensure that your client is specified in the **PATH** environment variable.

For DBMS-specific help:

- Trouble-shooting Sybase Connectivity
- Trouble-shooting Oracle Connectivity
- Trouble-shooting DB2 LUW and DB2 z/OS Connectivity
- Trouble-shooting SQL Server Connectivity

Trouble-shooting Sybase Connectivity

A common cause of client connectivity issues against a Sybase DBMS is the client on the workstation not being compatible with Rapid SQL or the Sybase environment on your workstation not being configured properly.

Verify that your Sybase environment variables have been set properly on your workstation. If you make any changes, restart Rapid SQL

Ensure you at least have the following variables (Environment Variables accessible through the Control Panel):

`SCROOT`

`SYBASE`

`SYBASE_OCS`

`SYBROOT`

`LIB` - ensure that the Sybase LIB directory is listed here

`INCLUDE` - ensure that the Sybase **INCLUDE** directory is listed here

Verify that the **PATH** value has only ONE SET of Sybase directories listed. You can modify this from the same Environment Variable window above in the **PATH** value:

```
%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\system32\WBEM;C:\PROGRA
~1\Serena\vm\win32\bin;C:\PROGRA~1\Serena\vm\common\bin\win32;C:\oracle
\ora92\bin;C:\oracle\ora92\jre\1.4.2\bin\client;c:\sybase\OCS-
15_0\lib3p;c:\sybase\OCS-15_0\dll;c:\sybase\OCS-
15_0\bin;c:\sybase\DataAccess\ADONET\dll;c:\sybase\DataAccess\OLEDB\dll
;c:\sybase\DataAccess\ODBC\dll;c:\sybase\Shared\Sybase Central
4.3;c:\sybase\ua\bin;C:\Program Files\Internet
Explorer;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\Pro
gram
```

In order to use the Auto-Discover feature, a valid SQL.ini file must be located in the following location:

`C:\sybase\`*`version_identifier_directory`*`\ini`

Where the *version_identifier_directory* is the name of the release-specific directory associated with this purpose, such as OCS-15_0 or OCS-12_5.

Trouble-shooting Oracle Connectivity

An Oracle client must be installed on the workstation. It must be defined/specified in the following locations:

- The ORACLE_HOME in the Windows Registry
- The Environment PATH value

In addition, in order for the Auto-Discover feature to work, there must be a valid Oracle tnsnames.ora and sqlnet.ora file in the Oracle client installed. They must be installed in the `%Oracle_home\Network\Admin` directory. As well, the sqlnet.ora file must have the NAMES.DIRECTORY_PATH value of TNSNAMES.

A typical sqlnet.ora file:

```
# sqlnet.ora Network Configuration File:

C:\oracle\product\10.2.0\client_2\network\admin\sqlnet.ora

# Generated by Oracle configuration tools.


# This file is actually generated by netca. But if customers choose to

# install "Software Only", this file wont exist and without the native

# authentication, they will not be able to connect to the database on
NT.


SQLNET.AUTHENTICATION_SERVICES= (NTS)


SQLNET.INBOUND_CONNECT_TIMEOUT = 30

SQLNET.SEND_TIMEOUT = 30
SQLNET.RECV_TIMEOUT = 30

TRACE_FILENO_SERVER=3

LOG_FILE_SERVER=svr.log

TRACE_FILE_CLIENT=clientsqlnet.trc


NAMES.DIRECTORY_PATH= (TNSNAMES)
```

Trouble-shooting DB2 LUW and DB2 z/OS Connectivity

The client must be defined in the workstation PATH value.

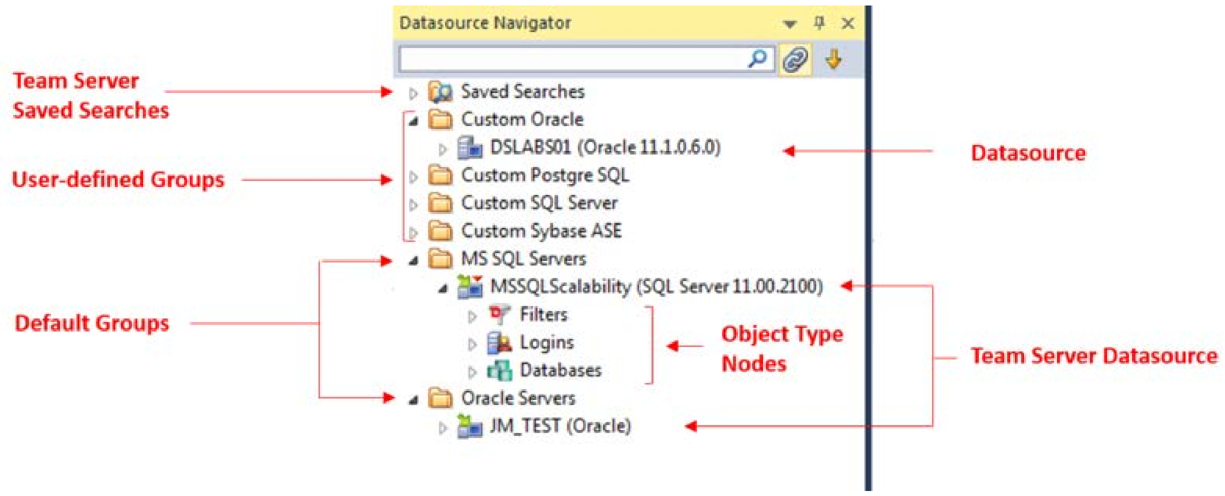
In order to Auto-Discover or manually register DB2 datasources, they must be configured in the DB2 Configuration Assistant. If DB2 datasources are not being detected, have your DBA provide a Configuration Assistant profile.

Trouble-shooting SQL Server Connectivity

In general, SQL Server datasources rarely go undetected. Each database server sends out a ping (via a service) to the network and Rapid SQL is usually able to find this. The SQL Server client must be defined in the workstation PATH value.

Using the Datasource Navigator

The Datasource Navigator provides a visual method for browsing, accessing, and manipulating objects. The Datasource Navigator contains a hierarchy consisting of groups, datasources, object type, and object nodes. Higher level nodes in the hierarchy have an associated Expand/Collapse icon, letting you drill down through the hierarchy to expose lower level nodes and objects.



- Groups organize the available data sources. A default set of groups corresponds to the supported DBMS platforms. You can also create custom groups to organize your data sources. For more information, see Categorizing Datasources using Color-coding and Labeling.
- Inside these groups are the data sources on your network that you have registered. You must register data sources in order to work against them. For more information, see Registering Datasources and Connecting to Datasources.
- With the exceptions of the **Filters** node, nodes below the datasource level correspond to the object types supported on the DBMS platform. For more information on the **Filters** node, see Filtering in the Navigator.
- At the lowest level, are the individual database objects on the data source. For a listing of object types by DBMS platform and detailed support summaries for each type, see Supported Objects.

For information on opening and closing the Navigator, Hiding and Displaying the Datasource Navigator

Bookmarks provide another means to quickly navigate to specific Navigator object nodes. For details, see Working with Bookmarks.

You can also initiate object type-specific actions and open object editors from the Navigator. For details, see Initiating Object Actions in the Navigator.

A number of viewing and organization options are available. They let you optimize the Navigator for your own purposes. These include:

- Basic viewing options that let you toggle between type-based or owner-based display, one-step expansion and collapse of hierarchy elements, hiding or display of system objects, saving of view option settings.
- Creating custom datasource groups to provide a deeper level of organization.
- Categorizing datasources using color-coding and labeling to quickly distinguish between datasources used for different purposes in your organization.
- Filtering the Navigator tree to display only selected nodes and objects. You can filter by name, schema or object type,

Connected to Team Server 2016

When Rapid SQL is connected to Team Server 2016 some other features are added to the **Datasource Navigator** and also, the Datasource Navigator displays datasources on Team Server 2016 and lets you work with relationships to local datasources.

Information about the basics of Team Server 2016 datasource support is in Working with Team Server 2016 Datasources.

To connect to **Team Server 2016**, ensure that Team Server 2016 usage is enabled and that you are logged in. For details, see Setting up and Connecting to Team Server 2016.

When Rapid SQL is connected to Team Server 2016, a node is added:

- **Saved Searches:** This node contains search definitions that you created using the advanced search feature. For details, see Searching Team Server 2016 for Datasources.

When Rapid SQL is connected to Team Server 2016, you can:

- Create a local datasource from a definition on the repository: register a local datasource as you would do for any static datasource, but use the **Relate Team Server Source** control to attach the local definition to a specified Team Server 2016 datasource definition. For more information, see Manually Registering or Editing Datasources.

Datasources attached to a Team Server 2016 definition are identified by a green arrow in the upper left part.

- Detach a currently attached local datasource definition so that it is no longer syncronized with the Team Server 2016 definition: right-click the attached Datasource Navigator and select **Unrelate Team Server Data Source** and select **Yes** when prompted to verify.
- Refresh/View Related ER Models: Right-click an attached datasource and select either **Refresh Related ER Models** or **View Related ER Models**.

- Work with multiple local and Team Server 2016 datasources. For details, see Reconciling Local and Team Server 2016 Datasources.

Topics

- Customizing the Navigator Displays.
- Connected/Selected Datasource options
- Filtering columns in Database Object Displays
- Working with Bookmarks
- Working with Supplementary Organizer Tabs

Hiding and Displaying the Datasource Navigator

Display of the Datasource Navigator can be toggled on and off.

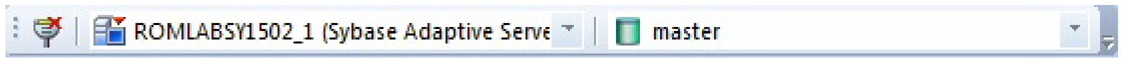
To Toggle Display of the Navigator On and Off

- Select View > Datasource Navigator.

For context information, see Using the Datasource Navigator.

Connected/Selected Datasource options

While you can connect to multiple datasources, at any one time only one connected datasource is the currently selected datasource. The currently selected datasource is displayed in the **Datasource** toolbar. For multiple-database datasources such as SQL Server and Sybase, a currently selected database is also displayed.



Certain features, to varying degrees, depend on the currently selected datasource or selected datasource/database combination. On a fairly trivial level, availability of some features or the initial state of a feature on being invoked is dependent on the selected datasource:

- In Rapid SQL, **Browse** menu object type availability is dependent on DBMS platform of the currently selected datasource.
- Some features, Query Builder for example, have user interface controls for selecting a datasource. On opening, these user interface items default to the currently selected datasource.

More strongly dependent on the selected datasource is the SQL Editor. A SQL Editor window not locked to a datasource, will direct all transactions to the currently selected datasource.

Note: For information on how to lock and unlock a SQL Editor window, see Locking a SQL Editor Window to a Datasource

When the Datasource Navigator is active, a SQL Editor opened against a particular datasource will continue to direct transactions to that datasource, until you explicitly change the selected datasource. Whether a SQL Editor or any other window or other user interface element is active, the method of explicitly changing the selected datasource is the same.

To Explicitly Make a Datasource the Selected Datasource

1. Select the datasource from the **Select Datasource** dropdown on the Datasource toolbar. Optionally, on a multi-database DBMS platform, select a database using the **Select From Available Databases** dropdown.

With a Datasource Navigator active, while an unlocked SQL Editor window is open, the selected datasource can be changed explicitly or automatically in response to Navigator navigation or related user interface actions.

This can result in the following issues:

- Active session-specific features such as execution directives and query options being sent to the wrong datasource

- o Features such as Semantic Validation returning false errors

How the selected datasource changes depends on the state of the **Link** button appearing at the top of the Navigator pane:

Link Button State	Description
The Navigator is linked to the **Select Datasource** combo	You can explicitly change the active datasource by selecting a datasource from the **Select Datasource** dropdown. The active datasource changes automatically as datasource-associated windows such as object editors become active in the Workspace or as you select nodes in the Navigator. As a side effect, the associated datasource node in the Navigator tree is selected.
The Navigator is NOT linked to the **Select Datasource** combo	When linking is inactive, you can only explicitly change the active datasource when an SQL Editor window is open. The active datasource still changes automatically as datasource-associated windows such as object editors become active in the Workspace. And when linking is inactive, the change in the selected datasource has no effect on the selected node in the Navigator tree.

To toggle between the two link states

1. Click the **Link** button.

Filtering columns in Database Object Displays

In the Database Navigator, name filtering is available with the Browser feature.

Two methods of filtering are available in the object displays. Both make use of the text boxes appearing below column names in the display.

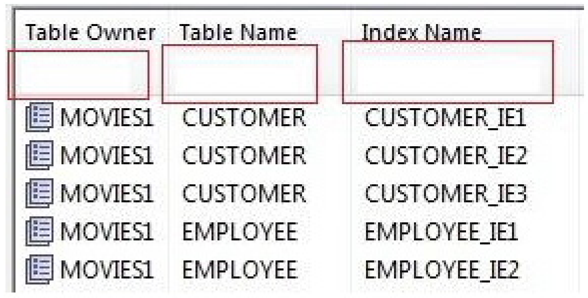


Table Owner	Table Name	Index Name
MOVIES1	CUSTOMER	CUSTOMER_IE1
MOVIES1	CUSTOMER	CUSTOMER_IE2
MOVIES1	CUSTOMER	CUSTOMER_IE3
MOVIES1	EMPLOYEE	EMPLOYEE_IE1
MOVIES1	EMPLOYEE	EMPLOYEE_IE2

To show only rows for which a given column value contains one or more contiguous characters

1. Type those characters into the text box below that column name.

To make use of regular expressions in column filtering

1. Ensure that regular expression usage is enabled. For details, see Explorer Options.
2. Type a valid regular expression search criteria into the text box below that column name. For more information, see Regular Expressions Support.

Note: The **RegEx ON/RegEx OFF** button in the lower right lets you enable or disable regular expression usage on-the-fly.

Working with Bookmarks

Bookmarks provides easy datasource navigation. You can right-click any node of the Navigator tree and add a bookmark. After a bookmark is created, you can simply select it from the **Bookmarks** menu, and be instantly positioned and connected to the resource in the Datasource Navigator tree.

Up to 256 bookmarks can be created.

To create a bookmark

1. On the Navigator, right-click the target node, and then select **Add Bookmark**. The **Add Friendly Bookmark Name** dialog opens.
2. Type the bookmark name.
3. Click **OK**.

Note: You cannot create multiple bookmarks referencing the same node.

To navigate to a bookmarked node

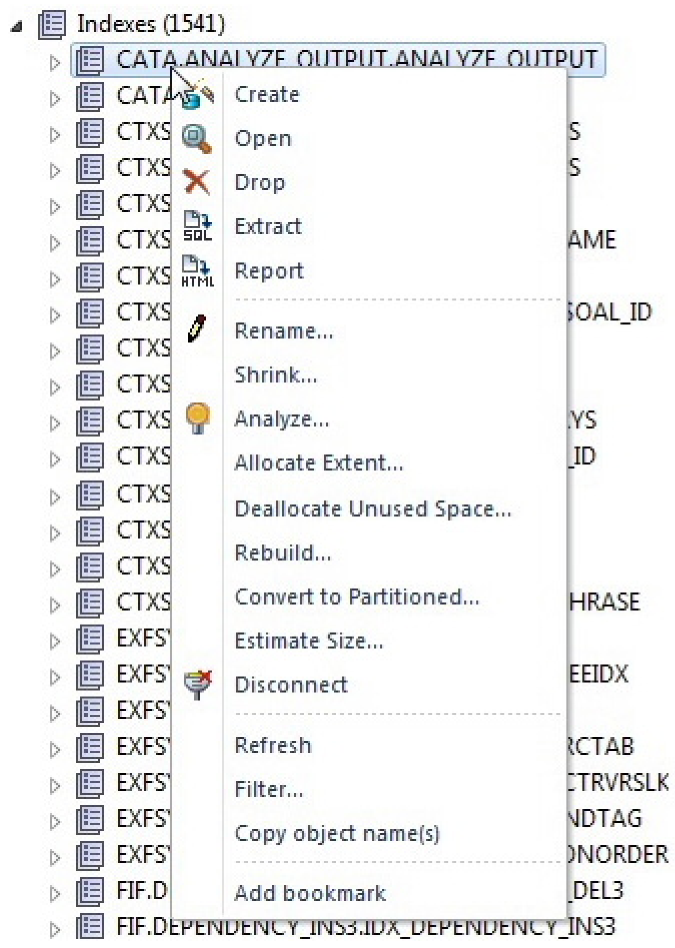
1. Select the Options menu (designated with a down arrow, above the Navigator tree), select **Bookmarks**, select a DBMS platform, and then select the bookmarked node.

Editing Bookmarks

1. Select the Options menu (designated with a down arrow, above the Navigator tree), and then select Bookmarks > Bookmark Manager. the Bookmark Manager dialog opens.
    - Rename a bookmark by selecting it and then clicking **Rename**. Then use the **Edit Bookmark Name** dialog to provide a new name.
    - Delete a bookmark by selecting it and then clicking **Delete**.

Initiating Object Actions in the Navigator

In the Datasource Navigator, right-click options let you initiate object actions against object types or against specific objects.



Right-click Object Actions in the Datasource Navigator

The right-click context menu for each object type lets you open creation wizards or editors or initiate object type specific actions.

For context information, see Hiding and Displaying the Datasource Navigator.

Topics

- Creating objects
- Modifying objects using editors

Working with Supplementary Organizer Tabs

In addition to the commonly used **Navigator** tab, the Organizer window offers the following tabs:

- **Project** - when working with the Project Management feature, this tab provides a visual method for browsing, accessing, and manipulating your projects. For more information on working with projects, see Project Management.
- **VC Files** - when working with the Version Control feature, this tab provides access to version control files. For more information on working with projects, see Version Control.
- **Favorites** - lets you designate and access favorite scripts. This tab lets you view, navigate, save, recall, and execute scripts. Sample Favorite Scripts are installed for Microsoft SQL Server, Oracle, and Sybase Adaptive Server. For more information, see Using the Script Library.

Using the Script Library

The Script Library 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 or execution, as needed.

To open the Script Library:

1. Select View > Script Library. The **Script Library** tab opens.

See the following topics for more information:

- Basic File and Folder Operations in the Script Library
- Execution Options in the Script Library
- Viewing and Setting File Properties in the Script Library

Basic File and Folder Operations in the Script Library

By default, the Script library contains a set of folders corresponding to supported DBMS platforms. These high level folders are preloaded with platform-specific subfolders, **System Diagnostics** and **Schema** for example, as well as files corresponding to DDL for object types supported on that platform.

The following basic options are provided.

- **Creating a subfolder**: right-click on a folder, select **New Folder**, and immediately provide a non-default name for the new folder.
- **Sorting the files in a folder to alphabetical order**: right-click on a file or folder and select **Sort**.
- **Renaming a file or folder**: right-click on a file or folder, select **Rename**, and provide a new name for the file or folder.
- **Deleting a file or folder**: right-click on a file or folder, select **Delete**, and select **Yes** when prompted to confirm.
- **Adding a SQL, JScript, or VBScript file to a folder**: right-click on a folder, select **New Item**, and use the controls on the **Script Library Item Properties** dialog to specify the file type, locate and select the file, and provide a hot key keystroke sequence to open the file.

Execution Options in the Script Library

The following execution are offered:

- **Opening a SQL, JScript, or VBScript file in the relevant editor**: right-click the file and select **Open**.

- **Executing a file**: right-click the file and select **Execute**, and then respond to any prompts regarding datasource or file type.
- **Opening a file in the Script Execution Facility**: right-click the file and select **Script Execution Facility**. For more information, see Script/File Execution Facilities.

Viewing and Setting File Properties in the Script Library

The following options are available:

- **Viewing the file/folder information, file type, and hot key sequence for an item**: right-click the file and select **Properties**.

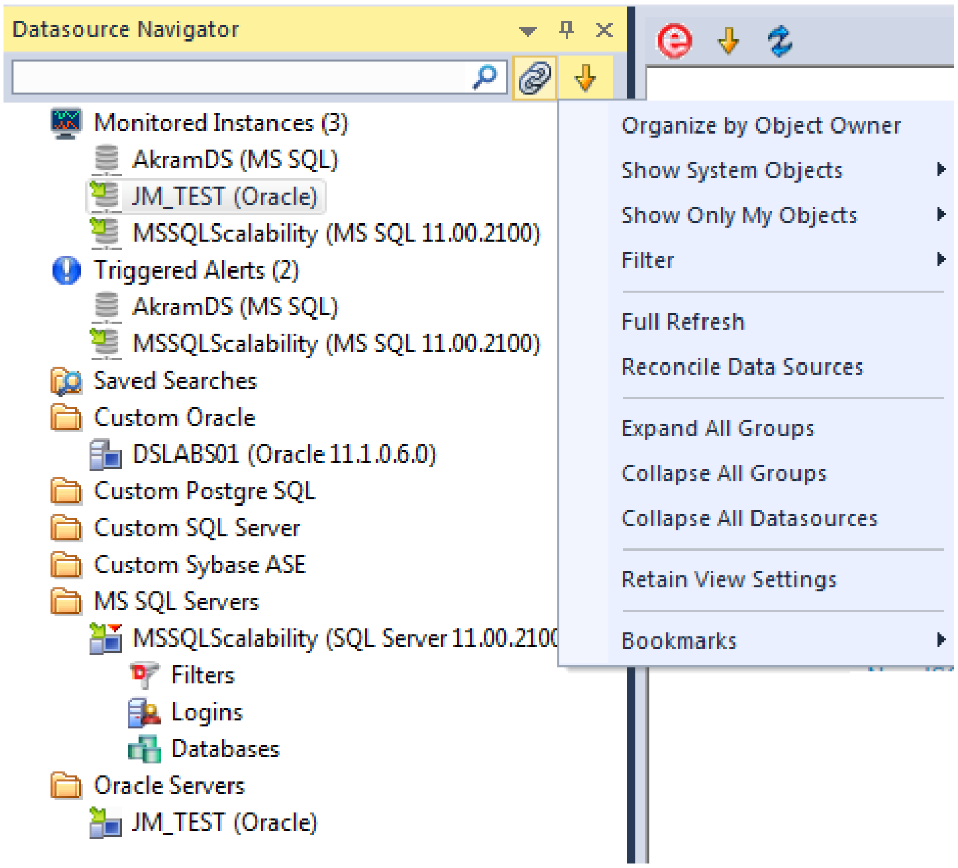
Customizing the Navigator Display

Rapid SQL offers flexibility in the elements displayed and in how elements are displayed in the Datasource Navigator.

- Basic Viewing Options in the Navigator
- Categorizing Datasources using Color-coding and Labeling
- Filtering in the Navigator
- Working with Datasource Groups

Basic Viewing Options in the Navigator

The dropdown menu, available above the Datasource Navigator tree, provides basic organization and display options.



To change basic organization and viewing options

1. From the Options dropdown menu, select an option. Use the following table as a guide to your selection.

Option	Description
Organize by Object Owner or **Organize by Object Type**	Toggling these options has the following effect: **Owner** - the Navigator displays objects for each user, categorized by object type. This display mode is most efficient if you are working with databases containing a high number of objects. **Type** - the Navigator displays all objects, categorized by object type, for all users in the same list. This display mode speeds or slows performance in databases that contain many objects.
Show System Objects and **Show Only My Objects**	Lets you control display of system objects and objects that you own: NEVER, ALWAYS, or USE DATASOURCE FILTER. For information on filter options, see Simple, on-the-fly, Character-based Tree Filtering.
Filter	Lets you control the datasources you want to show when connected to Team Server 2016. The filter lets you **Show Only My Data Sources**. If you do not use Team Server 2016 as datasource catalog, the option **Show Only Team Server Data Sources** is available, otherwise, the option is not enabled.
Full Refresh	Select to refresh.
Reconcile Data Sources	The **Reconcile Team Server Data Sources** dialog lets you work simultaneously with multiple datasource definitions. For more information see Reconciling Local and Team Server 2016 Datasources.
Expand All Groups	Select to expand all groups.
Collapse All Groups	Select to collapse all groups.
Collapse All Datasources	Select to collapse all datasources.
Retain View Settings	Select to retain the current state of the Navigator so that it opens the same way the next time you start Rapid SQL.
Bookmarks	Provides access to bookmark resources.

Tip: You can also set these options on the Explorer Options tab of the Options editor.

Working with Datasource Groups

Datasource groups help you to organize the datasources in your enterprise. Datasource Groups behave as folders in the Windows Explorer, allowing you to group related datasources together. Anywhere that datasources are presented in a hierarchical tree format, datasource group folders expand to display one or more contained datasources. By default, the Datasource Navigator has one group per supported DBMS platform.

You can create custom groups or subgroups. This level of organization is useful if you want to group large numbers of registered datasources or have another need to organize datasources. For basic database group maintenance tasks, see the following topics:

- Removing, Adding, and Renaming Datasource Groups.
- Changing the Level of a Datasource Group in the Hierarchy.

When you manually register a datasource, you have the option of changing the default group to which the datasource will automatically be assigned. Subsequently, you can move a datasource from one folder to another. For details, see Moving Datasources Between Groups.

Moving Datasources Between Groups

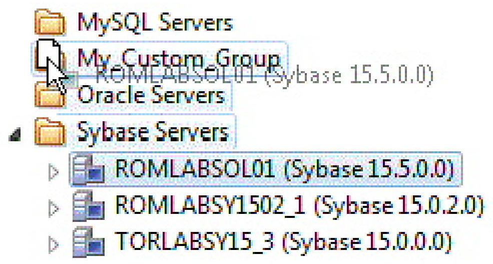
There are two ways to change datasource groups:

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

Note: You should always disconnect from a datasource before changing datasource groups.

To move a datasource between groups by dragging and dropping

1. If necessary, expand groups until both the datasource and the target datasource group are visible.
2. Drag the datasource over the target folder and drop it there.


MySQL Servers
My Custom Group
ROMLABSOL01 (Sybase 15.5.0.0)
Oracle Servers
Sybase Servers
ROMLABSOL01 (Sybase 15.5.0.0)
ROMLABSY1502_1 (Sybase 15.0.2.0)
TORLABSY15_3 (Sybase 15.0.0.0)

To move a datasource using the Change Datasource Group dialog box

1. If necessary, expand groups until the datasource you want to move is visible.

2. Right-click the datasource you want to move, and select **Change Group**.

The **Change Datasource Group** dialog box opens.

3. Use the S**elect...** tree to locate and select the target group.
4. Click **OK**.

For more general information on datasource groups and associated tasks, see Categorizing Datasources using Color-coding and Labeling.

Removing, Adding, and Renaming Datasource Groups

A set of datasource group maintenance tasks are available. You can remove database groups that you no longer need to access, or that have otherwise 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 containing group. You should move any contained datasources or groups before preceding. For details on moving datasources, see Moving Datasources Between Groups.
- The default datasource groups correspond to the DBMS platforms supported. 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. If necessary, expand datasource groups in the Navigator until the datasource you want to remove is visible.
2. Right-click the datasource group, and select **Remove**. You are prompted to verify.
3. Click **Yes**.

Similarly, you can define datasource groups to organize the datasources in your enterprise.

To create a new datasource group

1. If creating a folder within an existing folder, expand datasource groups in the Navigator until the target group is visible.
2. On the Navigator tree take one of the following actions:
    - To create a new folder at the top level, right-click in empty space and select **New** from the context menu.
    - To create a folder within an existing folder, right-click the target folder and select **New** from the context menu.

The **New Datasource Group** dialog opens.

3. In the **Datasource Group Name** box, type the name of the new datasource group.
4. Click **OK**.

Lastly, you can change the name of an existing datasource group.

To rename a datasource group

1. If necessary, expand datasource groups in the Navigator until the datasource you want to rename is visible.
2. On the Navigator, right-click the datasource group folder, and select **Rename**. The **Rename Datasource Group** dialog opens.
3. In the **Datasource Group Name** box, type the new name of the datasource group.
4. Click **OK**.

For related information, see the following topics:

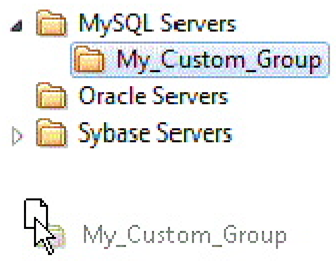
- See Changing the Level of a Datasource Group in the Hierarchy for information on changing hierarchy levels.
- See Categorizing Datasources using Color-coding and Labeling for more general information on working with datasource groups.

Changing the Level of a Datasource Group in the Hierarchy

You can reorganize the datasource group tree by moving individual groups up and down the hierarchy.

To move a datasource group to the top level of the hierarchy

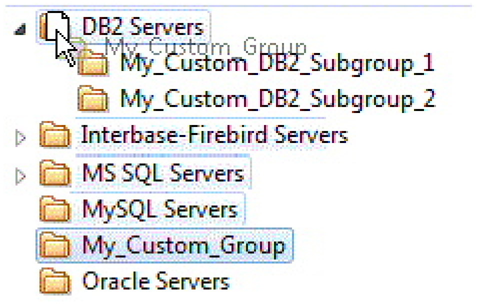
1. If necessary, expand groups until the datasource group you want to move is visible.
2. Drag the target group to open space in the Navigator pane.



To make a datasource group a subgroup within another group

1. If necessary, expand groups until the datasource group you want to move and the target group are visible.

2. Drag the target group and drop it on the datasource group that is to be the new parent.



For related information, see the following topics:

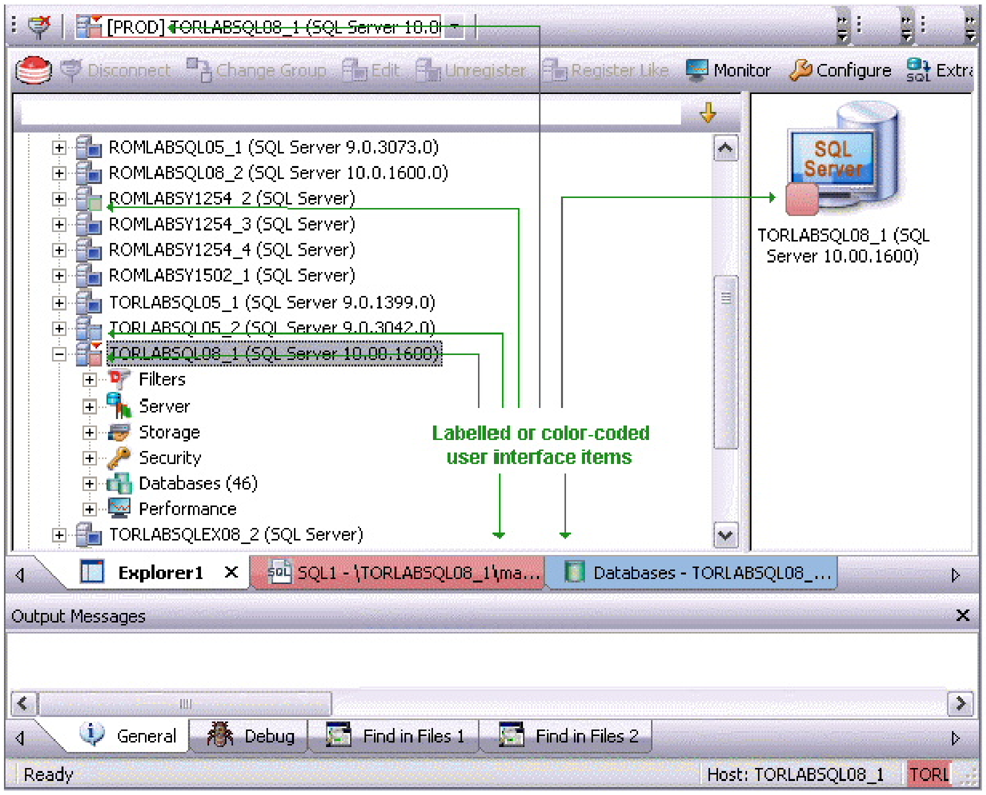
- See Removing, Adding, and Renaming Datasource Groups for information on basic datasource tree tasks.
- See Categorizing Datasources using Color-coding and Labeling for more general information on working with datasource groups.

Categorizing Datasources using Color-coding and Labeling

You can set up 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 Navigator icons for the datasource show 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 main window


[PROD] TORLABSQL08_1 (SQL Server 10.0
Disconnect
Change Group
Edit
Unregister
Register Like
Monitor
Configure
Extra
ROMLABSQL05_1 (SQL Server 9.0.3073.0)
ROMLABSQL08_2 (SQL Server 10.0.1600.0)
ROMLABSY1254_2 (SQL Server)
ROMLABSY1254_3 (SQL Server)
ROMLABSY1254_4 (SQL Server)
ROMLABSY1502_1 (SQL Server)
TORLABSQL05_1 (SQL Server 9.0.1399.0)
TORLABSQL05_2 (SQL Server 9.0.3042.0)
TORLABSQL08_1 (SQL Server 10.00.1600)
Filters
Server
Storage
Security
Databases (46)
Performance
TORLABSQLEX08_2 (SQL Server)
SQL Server
TORLABSQL08_1 (SQL Server 10.00.1600)
Labelled or color-coded user interface items
Explorer1
SQL1 - \TORLABSQL08_1\ma...
Databases - TORLABSQL08_...
Output Messages
General
Debug
Find in Files 1
Find in Files 2
Ready
Host: TORLABSQL08_1
TORL

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 Manually Registering or Editing Datasources.

Customizing Datasource Categories

You can 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 the Navigator 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 **Full 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.

Filtering in the Navigator

For ease of navigation, you can filter the Navigator tree to display only selected nodes and objects. The following filtering options are available:

- 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 datasources, 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.

Simple, on-the-fly, Character-based Tree Filtering

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

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

- Type one or more characters in the Filter box.

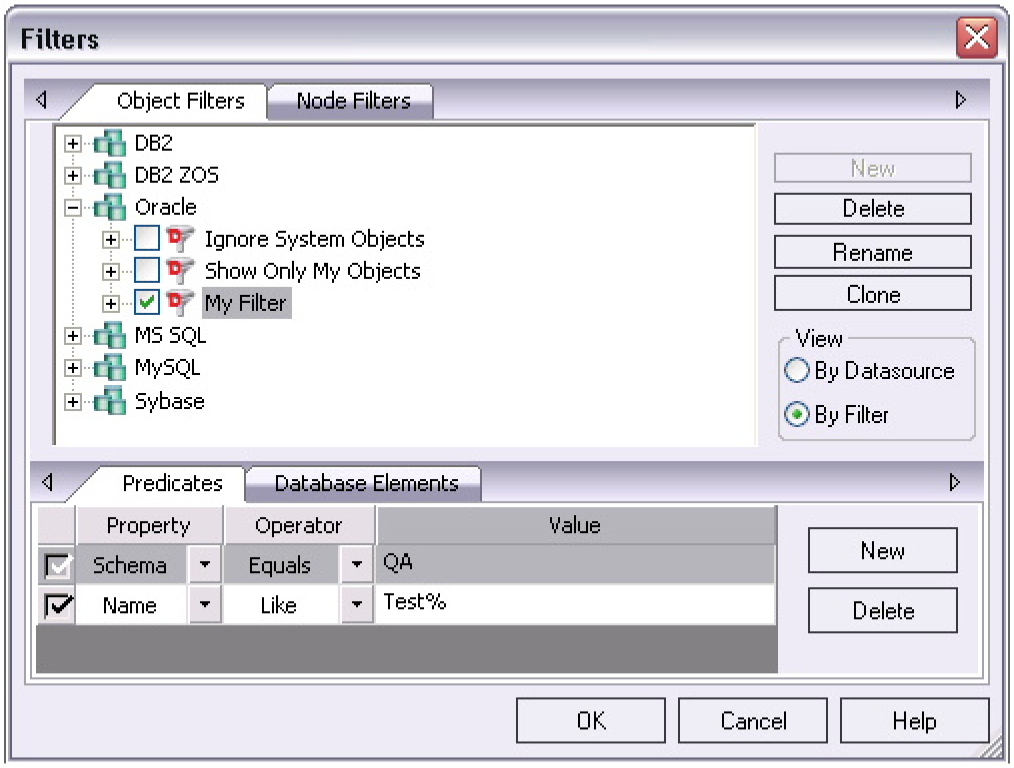
The display is updated to show only nodes whose name contains the typed character string and their parent nodes.

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 the Navigator tree display for a datasource by name, owning schema, or name/schema combinations. These filters are defined at the DBMS platform level and are enabled or disabled on a datasource by datasource level. Multiple object filters can be created for a DBMS platform, letting you enable combinations of filters for a single datasource, 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 Navigator tree for a datasource to display only objects belonging to the Schema **QA** and whose name starts with the string **Test**.



View: By Datasource [✓ By Filter]

	Property	Operator	Value
✓	Schema	Equals	QA
✓	Name	Like	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 datasource.

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

1. Right-click in the Navigator tree and select **Filter** from the context menu.

The **Filters** dialog opens. The **Object Filters** tab opens by default, letting you create an object filter definition. The **Node Filters** tab, on the other hand, lets you perform real-time filtering of object types for all datasources on a DBMS platform. For details, see Node, or Object Type, Filtering.

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 datasource, select the **By Datasource** view, select a specific datasource, and then proceed to **step 4**.

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. Wildcard characters and conventions supported on the target DBMS are supported.
- 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 datasource of that DBMS type.

To disable or enable an object filter for a datasource

1. Right-click in the Navigator 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 datasource 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 datasource in the Navigator 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 Navigator tree nodes for a DBMS platform. For example, if you do not typically work with credential-related objects such as users, groups, profiles or logins, you may wish to filter those out of the Navigator tree. You could either disable display of the individual object types or if practical, disable display of the **Security** node for the DBMS.

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

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

The **NodeFilters** tab opens by default, letting perform real-time filtering of object types for all datasources on a DBMS platform. The **Object Filters** tab, on the other hand, lets you create permanent, name-based or schema-based filters. For details, see Object Name/schema Filtering.

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 datasource includes two default Navigator tree filters, **Ignore System Objects** and **Show Only My Objects**. These filters can be enabled and disabled for each datasource but cannot be edited or deleted.

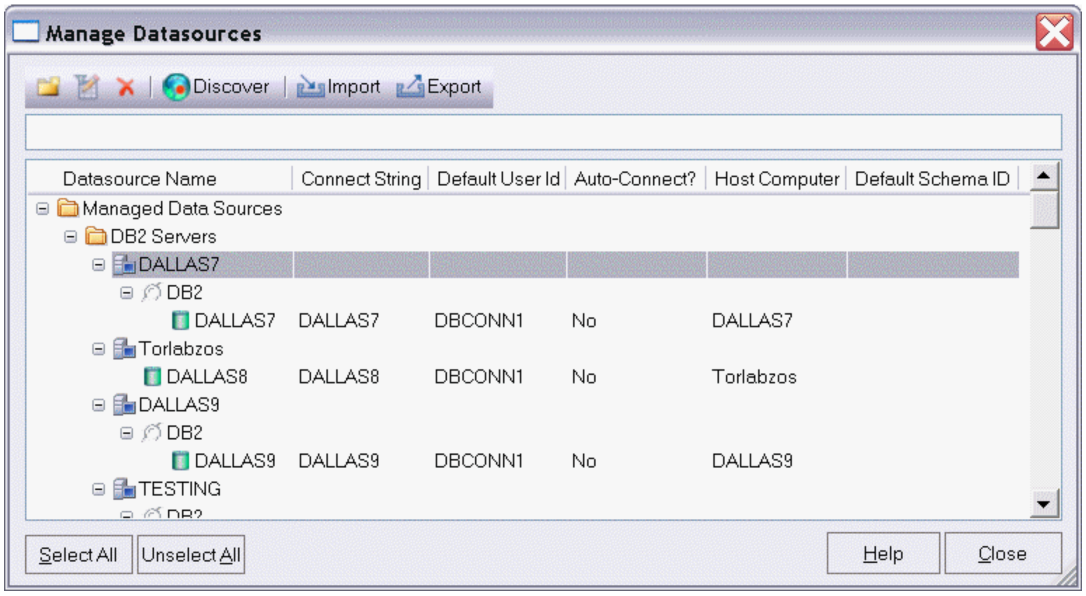
Using the Manage Datasources Utility

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 ManageDatasources.

The Manage Datasources dialog box opens.



Datasource Name	Connect String	Default User Id	Auto-Connect?	Host Computer	Default Schema ID
Managed Data Sources					
DB2 Servers					
DALLAS7					
DB2					
DALLAS7	DALLAS7	DBCONN1	No	DALLAS7	
Torlabzos					
DALLAS8	DALLAS8	DBCONN1	No	Torlabzos	
DALLAS9					
DB2					
DALLAS9	DALLAS9	DBCONN1	No	DALLAS9	
TESTING					
DB2					

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.
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 Datasources.
- Select a datasource and click the Edit Datasource button to edit that datasource definition.
- Select a datasource and click the Remove Datasource button to unregister that datasource definition. For more information, see Categorizing Datasources using Color-coding and Labeling.
- Click the **Discover** button to locate all configured datasources on your network and automate the process of registering them. For more information, see Moving Datasources Between Groups.
- Import or export datasource definitions. For details, see Importing and Exporting Datasource Definitions.

Team Server 2016 Support

Team Server 2016 is a data governance repository that can be used as an aid in managing enterprise data. It centralizes data resources, aids in resource-allocation planning, and fosters collaboration.

As an inventory, Team Server 2016 stores the following resources:

- Datasource definitions
- ER/Studio data models
- Glossaries and definitions
- User details.

Team Server 2016 is also a communications hub and information center. Activity streams provide the following:

- Add, modify, and delete resource details are logged.
- Users can post comments on resources or to other users.

Common social media conventions such as *Like*s and *Follow*s are supported.

Rapid SQL integrates with Team Server 2016. For information on preliminary steps, see Setting up and Connecting to Team Server 2016. Once configured and connected, the following options are provided:

Datasources	Datasource definitions on Team Server 2016 can be viewed in the Datasource Navigator. Local datasources created from the Team Server 2016 definitions can be permanently synchronized with the definitions on Team Server 2016. A set of options and actions let you manage your local and Team Server 2016 datasource definitions and their relationships. An advanced search mechanism is also provided. For details, see Working with Team Server 2016 Datasources.
Data models	If a database or schema of a synchronized datasource has associated model data in Team Server 2016, model data such as alerts and attribute/entity descriptions can be displayed in the SQL Editor as Code Assist proposals, as comments automatically added to scripts, as tooltips, and as the basis for warnings due to alerts. For details, see Viewing Embacadero Team Server 2016 Model Metadata in the SQL Editor.
Activity Streams	Users can monitor their own activity stream or the activity stream for a datasource definition on the Team Server 2016. For details, see Working with Team Server 2016 Streams.

Team Server 2016 as Datasource Catalog

Rapid SQL lets you work with **Team Server 2016** as the datasource catalog. For details, see Working with Team Server 2016 as Datasource Catalog.

Working from Team Server 2016

In addition, Rapid SQL lets you open a browser interface that lets you work directly with Team Server 2016. For details, see Working Directly with Team Server 2016.

Topics

Setting up and Connecting to Team Server 2016

Before you can make use of resources on Team Server 2016, you must login to the server. See your Team Server 2016 administrator for a login URL, credentials, and permissions setup.

1. Ensure that Team Server 2016 usage is enabled. For details, see Team Server 2016 Options.
2. Select Team > Login... and then use the **Embarcadero Team Server - Log In** dialog to provide credentials and complete the login.

Note: Use the **Keep me signed in** check box to have logins persist across startups and shutdowns of Rapid SQL. The **Keep me signed** control can be set and disabled using a registry entry. For details, see Locking Down Features Using the Registry.

After logging in, you have access to resources on the Team Server 2016 in the Datasource Navigator.

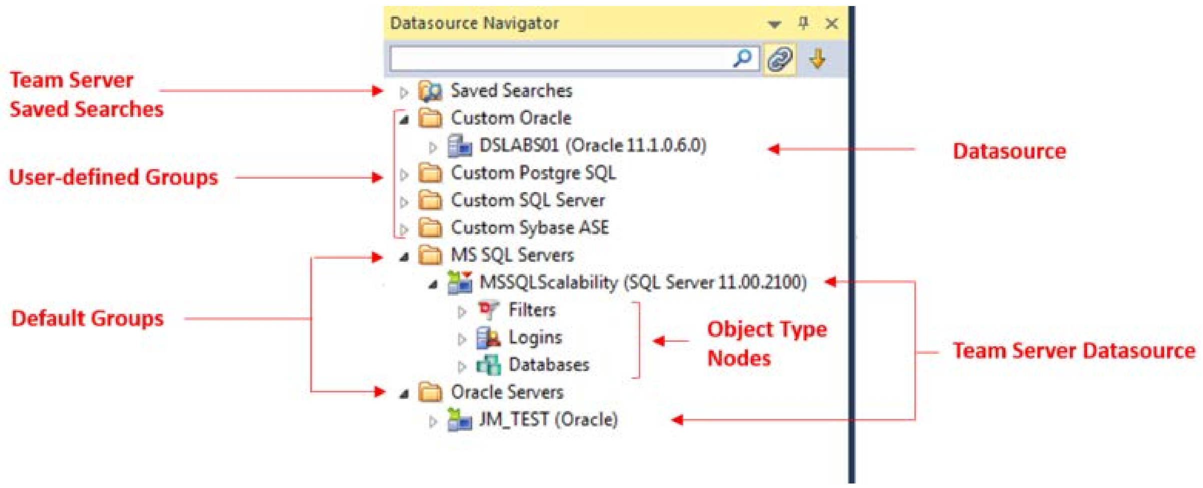
Topics

- Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor
- Working with Team Server 2016 Streams
- Working Directly with Team Server 2016

Working with Team Server 2016 Datasources

With Rapid SQL properly configured, and subsequent login to Team Server 2016, the Datasource Navigator the available datasource definitions on the Team Server 2016. A datasource definition on Team Server 2016 can be registered as a local datasource.

Registering a Team Server 2016 datasource as a local datasource creates a local copy of the datasource definition, accessible in the existing Datasource Navigator.



Note: For access to Team Server 2016 documentation, see docs.embarcadero.com.

By default, when you register a Team Server 2016 datasource as a local datasource, the local definition is “attached” to the Team Server 2016 version. The local definition’s key connectivity information is synchronized with the Team Server 2016 definition on each attempt to connect to the datasource.

An attached datasource has a distinctive icon:

You can subsequently detach a local datasource definition from a Team Server 2016 definition. Settings for the detached, local datasource definition are those that were active on the Team Server 2016 definition when the datasource was detached. When no longer attached to a Team Server 2016 datasource definition, a detached datasource is functionally no different than any locally registered (manually registered or automatically discovered) datasource.

Datasource Navigator lets you perform tasks such as attaching local datasources and managing datasource definitions residing on Team Server 2016.

Basic steps in getting started with Team Server 2016 datasources are as follows:

1. **Enable and configure Team Server 2016 usage** - For details, see Team Server 2016 Options.

2. **Log in to Team Server 2016** - For details, on how to establish a session with Team Server 2016, see Setting up and Connecting to Team Server 2016.

3. **Register Team Server 2016 datasources definitions as local datasources** - See Using the Datasource Navigator for information on how to use a simple Navigator interface to manage your local/Team Server 2016 datasources.

Note: Key connectivity information, the **Connection Information** tab settings (excluding **Datasource Name**) and a custom JDBC driver if supplied, is not editable in the resulting local datasource definition. **Security Parameters** tab information, such as **User ID** and **Password**) is not copied to the local datasource definition.

4. **Connect to a Team Server 2016 datasource that you have registered locally** - See Connecting to Datasources.

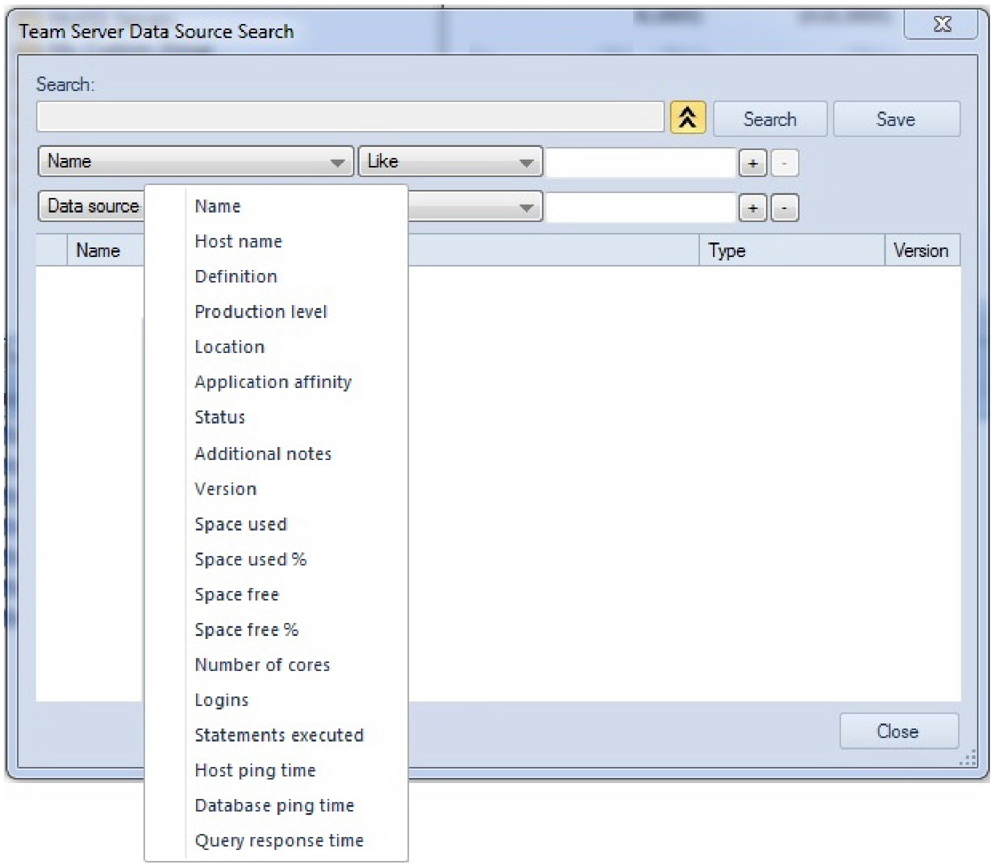
Topics

- Searching Team Server 2016 for Datasources
- Reconciling Local and Team Server 2016 Datasources
- Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor.

Searching Team Server 2016 for Datasources

While the Datasource Navigator provides a useful browsing mechanism, you can also perform an advanced search of datasources on Team Server 2016. The **Team Server Data Source Search** dialog lets you search for Team Server 2016 datasource definitions by name or by one or more ANDed conditions:

Name	Datasource Type	Host Name
Definition	Production Level	Location
Application Affinity	Status	Additional Notes
Version, Space Used	Space Used %	Space Free
Space Free %	Number of Cores	Logins, Statements Executed
Host Ping Time	Database Ping Time	Query Response Time



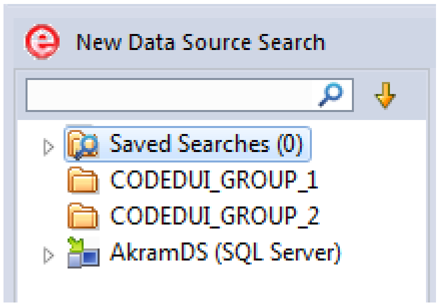
To initiate a name-based or criteria-based advanced search

1. Ensure that Team Server 2016 usage is enabled and that you are logged in. For details, see Setting up and Connecting to Team Server 2016.

2. Select Datasource > Team Server Data Source Search....

OR

3. Click the **New Data Source Search** button, located in the top of the **Datasource Navigator**. The **Saved Searches** node must be selected.



OR

4. Right-click the **Saved Searches** node, located in the top of the **Datasource Navigator** and selecting **New Data Source Search...**.

The **Team Server Data Source Search** dialog opens.

To search by criteria

1. Use the expand/collapse control to expose the condition controls.

This deactivates the **Search** box.

2. Use the criteria, operator, and value controls to create a condition.
3. To add conditions, use the + control and repeat the previous step.
4. When ready, click **Search**.

For more general information on Team Server 2016 integration, see Team Server 2016 Support.

Reconciling Local and Team Server 2016 Datasources

The **Reconcile Team Server Data Sources** dialog lets you work simultaneously with multiple datasource definitions.

To reconcile multiple datasource definitions

1. Ensure that Team Server 2016 usage is enabled and that you are logged in. For details, see Setting up and Connecting to Team Server 2016.

2. On the Datasource Navigator dropdown, select **Reconcile Data Sources**. The **Reconcile Team Server Data Sources** dialog opens.

3. Reconcile your Team Server 2016 and local datasources using the three list boxes as follows:

    - **Matching data sources** - lists Team Server 2016 datasource definitions and local datasource definitions with identical connection information details but that are currently not attached. Selecting an entry from this list attaches the local datasource to the Team Server 2016 datasource, ensuring that any changes to the Team Server 2016 definition will be copied to the local datasource definition on the next attempt to connect to the datasource

Note: For more information on local, attached, and matching datasource definitions, see Working with Team Server 2016 Datasources.

    - **Unmatched Team Server data sources** - lists Team Server 2016 datasource definitions for which there are no local datasource definitions with matching connection information. Selecting an entry from this list creates a matching local datasource definition and attaches the local definition to the matching Team Server 2016 datasource definition.

    - **Unmatched local data sources** - lists local datasource definitions for which there are no Team Server 2016 datasource definitions with matching connection information. Selecting an entry from this list creates a matching Team Server 2016 datasource definition and attaches the local definition to the Team Server 2016 datasource definition.

4. Click **Reconcile**.

Working with Team Server 2016 Streams

Embarcadero Team Server 2016 activity streams include user Likes, user follows, and alerts associated with users or datasources. You can open a window to view your notification.

Note: Embarcadero Team Server 2016 is a licensed feature. For access to Team Server 2016 features and to obtain credentials, contact your Team Server 2016 administrator. For access to Team Server 2016 documentation, see http://docs.embarcadero.com/.

To view the activity stream for a Team Server 2016 resource

1. Ensure that Team Server 2016 usage is enabled and configured. For details, see Team Server 2016 Options.
2. Select **View > Team Server > Streams**. The **Team Server 2016 Streams** window opens.
3. Use the controls in the **Team Server 2016 Streams** window to display the activity stream for the current user or a Team Server 2016 datasource definition.

For more general information on Team Server 2016 integration, see Team Server 2016 Support.

Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor

If a database or schema has associated model data in Team Server 2016, model data such as alerts and attribute/entity descriptions can be displayed/included in the SQL Editor in the following ways:

- Attribute/Entity descriptions can be included in Code Assist proposals
- On explicitly formatting a script, metadata for tables and views referenced in SELECT, CREATE PROCEDURE, CREATE FUNCTION, CREATE TRIGGER, CREATE VIEW statements are added in a single comment block above the referencing statement
- When hovering the mouse over a table column name, table name or view name, a tooltip showing the entity/attribute description is displayed
- Warning icons are displayed in the window gutter beside lines referencing objects whose metadata includes alerts
- On execution of a script referencing objects whose metadata includes alerts, the error pane opens with an entry for each referenced object.

Note: Metadata for view columns is not available.

To use model metadata in the SQL Editor

1. Ensure that Team Server 2016 repository usage is enabled and that you are logged in to the Team Server 2016. For details, see Team Server 2016 Options and Setting up and Connecting to Team Server 2016.
2. Configure the **Metadata** features in the Options editor. For details, see ISQL Options - Code Assist Tab.

For related information, see Team Server 2016 Support.

Working Directly with Team Server 2016

You can open a browser interface on Team Server 2016 to access its full range of functionality. This includes increased datasource and data model functionality, access to glossaries and terms, reporting, and access to other users.

To open a browser interface on Team Server 2016

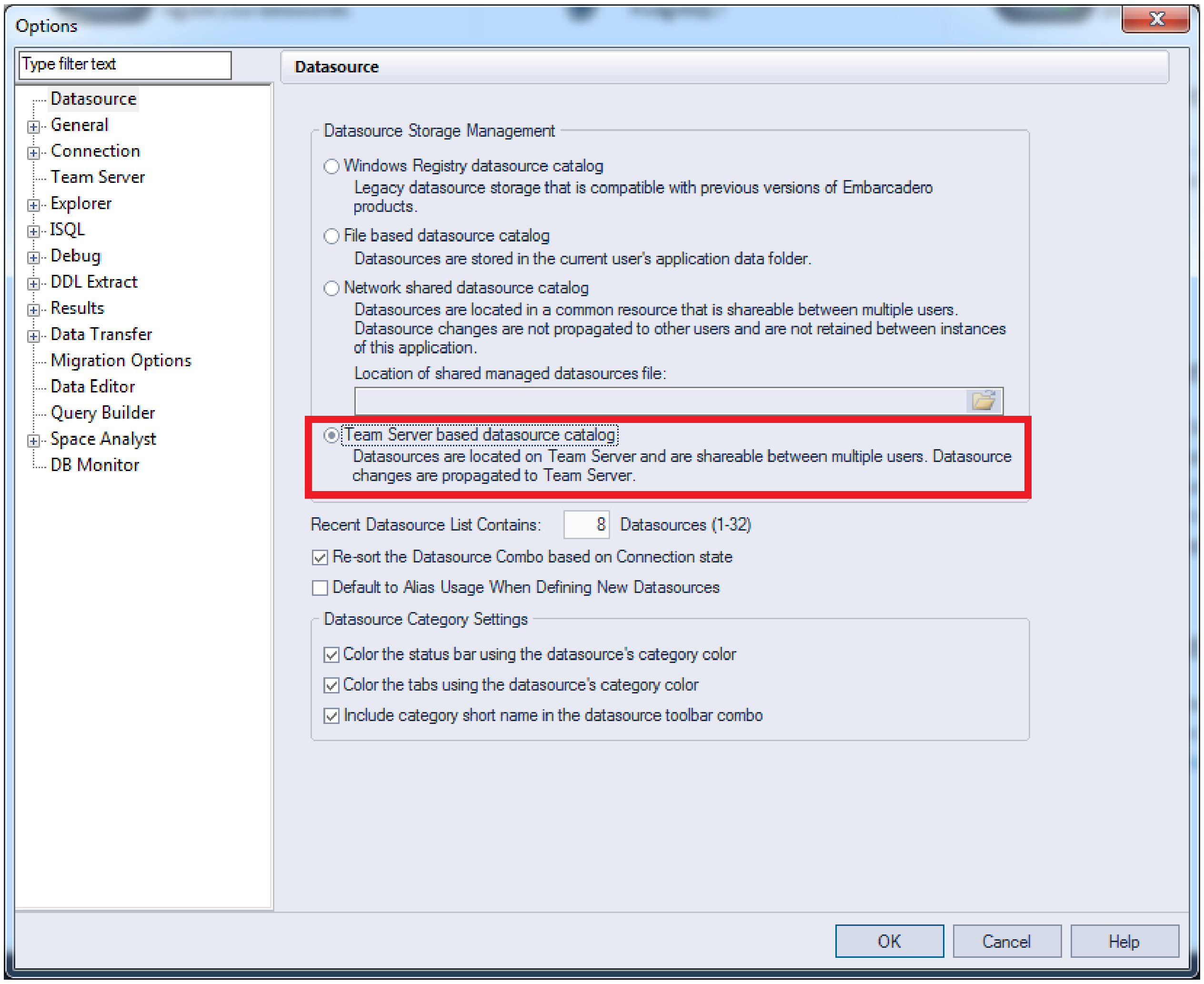
1. Ensure that access to Team Server 2016 has been enabled. For details, see Team Server 2016 Support.
2. Select Tools > Team Server > View in Web browser.

For access to the full set of Team Server 2016 documentation, see http://docs.embarcadero.com/.

Working with Team Server 2016 as Datasource Catalog

Team Server 2016 can be used as a datasource Catalog. When configuring **Team Server 2016** as datasource catalog, all the dbms from the configured Team Server 2016 are shown in the Datasource Navigator.

In Tools > Options > Datasources you can select Team Server 2016 as the source of the catalog. For more information, see Datasource Options.



Datasource

Datasource Storage Management

( ) Windows Registry datasource catalog
Legacy datasource storage that is compatible with previous versions of Embarcadero products.

( ) File based datasource catalog
Datasources are stored in the current user's application data folder.

( ) Network shared datasource catalog
Datasources are located in a common resource that is shareable between multiple users. Datasource changes are not propagated to other users and are not retained between instances of this application.
Location of shared managed datasources file:

(●) Team Server based datasource catalog
Datasources are located on Team Server and are shareable between multiple users. Datasource changes are propagated to Team Server.

Recent Datasource List Contains: 8 Datasources (1-32)

☑ Re-sort the Datasource Combo based on Connection state
☐ Default to Alias Usage When Defining New Datasources

Datasource Category Settings

☑ 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

Database Object Management

Extensive object management facilities are available for each supported DBMS platform. The following sets of topics describe the major areas of support:

Supported Objects	Describes the database objects and related elements supported. 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 you can perform against database objects.

Supported Objects

Database object management features are available across different supported database platforms. The table below indicates the supported object types by platform:

	DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
Aliases	✓	✓						✓	
Asymmetric Keys							✓		
Blob Filters			✓						
Certificates							✓		
Chains					✓				
Check Constraints	✓	✓	✓		✓	✓	✓	✓	
Clusters					✓				
Database Links					✓				
Database Triggers							✓		
Databases		✓			✓		✓	✓	
Defaults							✓	✓	
Directories					✓				
Domains			✓			✓			✓

Encryption Keys			✓						
Events									✓
Exceptions			✓						
Exclusion Constraints						✓			
Extended Procedures								✓	
External Functions			✓						
Foreign Keys	✓	✓	✓	✓	✓	✓	✓	✓	✓
Full-text Catalogs							✓		
Full-text Indexes							✓		
Functions	✓	✓		✓	✓	✓	✓	✓	✓
Generators (Rapid SQL)			✓						
Groups	✓				✓		✓	✓	✓
Indexes	✓	✓	✓	✓	✓	✓	✓	✓	✓
Instance	✓				✓				
Java Classes					✓				
Java Resources					✓				
Java Sources					✓				
Jobs (Oracle)					✓				
Job Queues					✓				
Join Indexes									✓
Libraries					✓				
Logins							✓	✓	
Materialized Query Tables	✓								
Materialized Views					✓				
Materialized View Logs					✓				

Outlines					✓				
Package Bodies					✓				
Packages	✓	✓			✓				
Partition Functions							✓		
Partition Schemes							✓		
Plans		✓							
Primary Keys	✓	✓	✓	✓	✓	✓	✓	✓	✓
Procedures	✓	✓	✓		✓		✓	✓	✓
Profiles					✓				
Programs					✓				
Recycle Bin					✓			✓	
Roles			✓		✓	✓	✓	✓	
Rules						✓	✓	✓	
Schedules					✓				
Schema						✓	✓		
Segments							✓	✓	
Sequences	✓	✓			✓				
Shadows			✓						
Structured Types	✓	✓							
Symmetric Keys							✓		
Synonyms		✓			✓		✓		
System Indexes							✓	✓	✓
System Tables							✓	✓	✓
System Triggers									✓
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.

Creating and editing

See the following topics:

- Aliases Wizard (DB2 LUW) and Aliases Editor (IBM DB2 LUW)
- Aliases Wizard (DB2 Z/OS) and Aliases Editor (IBM DB2 Z/OS)
- Aliases Wizard (Sybase ASE) and Aliases Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	SYB ASE
Drop	✓	✓	✓
Extract	✓	✓	✓
Report	✓	✓	✓
Transfer Ownership	✓		

Asymmetric Keys

Asymmetric keys, in addition to symmetric keys and certificates, provide support for Public Key Encryption. Asymmetric keys consist of a private key and corresponding public key and are used to secure symmetric keys.

Creating and editing

- Asymmetric Keys Wizard (SQL Server) and Asymmetric Keys Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR
Drop	✓
Extract	✓
Report	✓

Related information

- Certificates
- Symmetric Keys

Blob Filters

Rapid SQL provides support for blob filters, user-written programs that convert data stored in Blob columns from one subtype to another.

Creating and editing

See Blob Filters Wizard (ITB/FBD) and Blob Filters Editor (InterBase/Firebird).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ITB/FBD *
Drop	✓
Extract	✓
Rename	✓
Report	✓

Certificates

Certificates, in addition to symmetric and asymmetric keys, provide support for Public Key Encryption. Certificates are digitally signed security objects containing a public key and optionally, a private key.

Creating and editing

- Certificate Wizard (SQL Server) and Certificates Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR
Add Private Key	✓
Backup Certificate	✓
Drop	✓
Extract	✓
Report	✓

Related information

- Asymmetric Keys
- Symmetric Keys

Chains

The following summarizes support for named series of tasks

Platform availability

DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
				✓				

Creating and editing

No support is provided for creation or editing this object type.

DBMS platform availability and object actions/operations supported

No specific object actions are available for this object type/

Related Information

See details for the following, related object types:

- Jobs (Oracle)
- Programs
- Schedules

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.

Creating and editing

See Add or Modify Check Constraint.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	ORCL	PSTGRS *	SQL SVR	SYB ASE
Change Status (check constraints)				✓		✓	
Drop	✓	✓	✓	✓	✓	✓	✓
Extract	✓	✓	✓	✓	✓	✓	✓
Hide Text							✓
Rename	✓	✓	✓	✓	✓	✓	✓
Report	✓	✓	✓	✓		✓	✓
Transfer Ownership	✓						

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.

Creating and editing

- Clusters Wizard (Oracle) and Clusters Editor (Oracle).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

- Database Links Wizard (Oracle) and Database Links Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

- Database Triggers Wizard (SQL Server) and Database Triggers Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR
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.

Creating and editing

- Databases Wizard (DB2 LUW) and Databases Editor (IBM DB2 LUW)
- Databases Wizard (DB2 Z/OS) and Databases Editor (IBM DB2 Z/OS)
- Databases Wizard (SQL Server) and Databases Editor (SQL Server)
- Databases Wizard (Sybase ASE) and Databases Editor (Sybase ASE)

Note: Objects of this type cannot be created or edited against PostgreSQL datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	PSTGRS *	SQL SVR	SYB ASE

Attach Database				✓	
Checkpoint				✓	✓
Copy Schema	✓				
DBCC				✓	✓
Detach Database				✓	
Drop	✓	✓	✓	✓	✓
Drop Automatic Storage Path(s)	✓				
Extract	✓	✓	✓	✓	✓
Hide Text					✓
Move Log					✓
Quiesce (Database)	✓				
Rename				✓	✓
Report		✓		✓	✓
Resynchronize					✓
Set Online/Offline				✓	✓
Shrink				✓	
Start Database		✓			
Stop Database		✓			
Unquiesce	✓				
Update Statistics				✓	

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.

Creating and editing

- Defaults Wizard (SQL Server) and Defaults Editor (SQL Server)
- Defaults Wizard (Sybase ASE) and Defaults Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR	SYB ASE
Drop	✓	✓
Extract	✓	✓
Hide Text		✓
Rename	✓	✓
Report	✓	✓

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.

Creating and editing

- Directories Wizard (Oracle) and Directories Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Drop	✓
Extract	✓
Report	✓

Domains

Support for domains is provided, letting you work with these type definitions.

Creating and editing

Domains Wizard (ITB/FBD) and Domains Editor (InterBase/Firebird).

Domains Wizard (PostgreSQL) and Domains Editor (PostgreSQL).

Note: No support is provided for creation or editing of this object type against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions..

	ITB/FBD *	PSTGRS *	SYB IQ
Change Owner		✓	
Change Schema		✓	
Drop	✓	✓	✓
Extract	✓	✓	✓
Object Properties			✓
Rename	✓	✓	✓
Report			✓

Encryption Keys

Rapid SQL provides support for encryption keys, used in encrypting and decrypting tables and columns.

Creating and editing

See Encryption Keys wizard (ITB/FBD) and Encryption Keys editor (InterBase/Firebird).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ITB/FBD *
Drop	✓
Extract	✓
Rename	✓
Report	✓

Events

These state changes events can be monitored or used as trigger conditions.

Creating and editing

No support is provided for creation or editing of this object type against a Sybase IQ datasource.

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SYB IQ
Drop	✓
Extract	✓
Object Properties	✓

Exceptions

Rapid SQL provides support for exceptions, letting you work with these named error messages.

Creating and editing

See Exceptions Wizard (ITB/FBD) and Exceptions Editor (InterBase/Firebird).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ITB/FBD *
Drop	✓
Extract	✓
Rename	✓
Report	✓

Exclusion Constraints

An Exclusion Constraint guarantees that if any two rows are compared on the specified columns or expressions using the specified operators, not all of comparisons will return TRUE.

Creating and editing

Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL) and Exclusion Constraints, Primary Keys, or Unique Keys Editors (PostgreSQL).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	PSTGRS *
Drop	✓
Extract	✓
Rename	✓

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.

Creating and editing

- Extended Procedures Wizard (Sybase ASE) and Extended Procedures Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SYB ASE
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.

Creating and editing

See External Functions Wizard (ITB/FBD) and External Functions editor (InterBase/Firebird).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ITB/FBD *
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.

Creating and editing

- Foreign Keys Wizard (DB2 LUW) and Foreign Keys Editor (IBM DB2 LUW)
- Foreign Keys Wizard (DB2 Z/OS)) and Foreign Keys Editor (IBM DB2 Z/OS)
- Foreign Keys Wizard (ITB/FBD) and Foreign Keys editor (InterBase/Firebird)
- Foreign Keys wizard (MySQL) and Foreign Keys editor (MySQL)
- Foreign Keys Wizard (Oracle) and Foreign Keys Editor (Oracle)
- Foreign Keys Wizard (PostgreSQL) and Foreign Keys Editor (PostgreSQL)
- Foreign Keys Wizard (SQL Server) and Foreign Keys Editor (SQL Server)
- Foreign Keys Wizard (Sybase ASE) and Foreign Keys Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
Change Status (check constraints)				✓		✓		
Drop	✓	✓		✓	✓	✓	✓	✓
Extract	✓	✓		✓	✓	✓	✓	✓
Rename	✓	✓		✓	✓	✓	✓	
Report	✓	✓		✓		✓	✓	
Transfer Ownership	✓							

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.

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.

Creating and editing

- Full-text Catalogs Wizard (SQL Server) and Full-text Catalogs Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR
Drop	✓
Extract	✓
Rebuild (Full-text Catalogs)	✓
Reorganize (SQL Server Full-text Catalogs)	✓
Report	✓

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.

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.

Creating and editing

- Full-text Indexes Wizard (SQL Server) and Full-text Indexes Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR
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 you can 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 ASE, etc. A function mapping needs to be created in conjunction with the template function.

Creating and editing

- Functions Wizard (DB2 LUW) and Functions Editor (IBM DB2 LUW)
- Functions Wizard (DB2 Z/OS) and Functions Editor (IBM DB2 Z/OS)
- Functions wizard (MySQL) and Functions editor (MySQL)
- Functions Wizard (Oracle) and Functions Editor (Oracle)
- Functions Wizard (PostgreSQL) and Functions Editor (PostgreSQL)
- Functions Wizard (SQL Server) and Functions Editor (SQL Server)
- Functions Wizard (Sybase ASE) and Functions Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
Change Owner					✓			
Change Owner					✓			
Compile				✓				
Create Synonym				✓				
Describe	✓	✓		✓				
Drop	✓	✓	✓	✓	✓	✓	✓	✓
Execute				✓				
Extract	✓	✓	✓	✓	✓	✓	✓	✓
Hide Text							✓	
Object Properties								✓
Rename					✓	✓	✓	
Report	✓	✓	✓	✓		✓	✓	
Transfer Ownership	✓							

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. Two tag pairs, ETStart and ETEnd, 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)

Rapid SQL provides support for generators, letting you work with these objects used to generate sequential numbers in a column.

Creating and editing

See Generators Wizard (ITB/FBD) and Generators editor (InterBase/Firebird).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ITB/FBD *
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.

Creating and editing

- Groups Wizard (Sybase ASE) and Groups Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	ORCL	SYB ASE	SYB IQ
Drop			✓	✓
Extract	✓	✓	✓	✓
Report	✓	✓	✓	

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

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.

Creating and editing

- Indexes Wizard (DB2 LUW) and Indexes Editor (IBM DB2 LUW)
- Indexes Wizard (DB2 Z/OS) and Indexes Editor (IBM DB2 Z/OS)
- Indexes Wizard (ITB/FBD) and Indexes editor (InterBase/Firebird)
- Indexes, Primary Keys, or Unique Keys wizard (MySQL) and Indexes, Primary Keys, and Unique Keys editors (MySQL)
- Indexes Wizard (Oracle) and Indexes Editor (Oracle)
- Indexes Wizard (PostgreSQL) and Indexes Editor (PostgreSQL)
- Indexes Wizard (SQL Server) and Indexes Editor (SQL Server)
- Indexes Wizard (Sybase ASE) and Indexes Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ	SYB IQ
Allocate Extent					✓					
Analyze					✓					
Deallocate Unused Space					✓					
Disable Index			✓				✓			
Drop	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DBCC							✓	✓		
Estimate Size					✓			✓		
Extract	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Place								✓		
Rebuild Index		✓	✓		✓			✓		
Rename	✓	✓	✓		✓	✓	✓	✓		
Reorganize /Reorg	✓	✓					✓	✓		
Report	✓	✓	✓	✓	✓		✓	✓		
Shrink					✓					
Set Statistics			✓							
Transfer Ownership	✓									
Update Statistics	✓						✓	✓		

Missing Indexes

Missing indexes is a feature which helps you to improve the performance of your system. The Missing indexes functionality proposes you to create new indexes for improving the system in case they are needed.

Availability of **Missing Indexes** for different DBMS:

- Missing Indexes (SQL Server)

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.

System indexes and user-defined indexes are handled separately in the Datasource Navigator 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, 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.

System indexes and user-defined indexes are handled separately in the Datasource Navigator to ensure that system indexes are not accidentally altered or dropped.

Missing Indexes (SQL Server)

Missing indexes is a feature which helps you to improve the performance of your system. The Missing indexes functionality proposes you to create new indexes for improving the system in case they are needed.

To access Missing Indexes:

From Databases Node

Accessing Missing Indexes

1. Open the **Databases** node of the tree.
2. Open the database of the tree where you want to check if there are **Missing Indexes**.
3. Select the **Tables** node in the tree.
4. Open the **Table Editor** where you want to check if there are "Missing Indexes". See Opening an Object Editor.
5. Select the **Missing Indexes** panel if enabled.

You can also access the **Missing Indexes** with the dedicated node under the **Database** node.

Adding Indexes

From the **Missing Index** panel you can create an **Index** with the information needed. For that:

1. Double click in the **Missing Index** you want to create.

OR

2. Right-click and **Create...**

It opens the Indexes Wizard (SQL Server) filled with the information from the selected **Missing Index**.

Instance

Note: This object is supported for IBM DB2 and Oracle.

Instance is placed as the first level of information under the Datasource node in the Datasource Navigator. Instance includes:

- o DB Manager Configuration
- o Data sources

Available Functionality

The following functionality is available for this object type:

Databases Wizard (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.

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Compile	✓
Create Synonym	✓
Drop Java	✓
Load Java	✓
Report	✓

Java Resources

Note: This object is supported by Oracle only.

The Java Resources node offers support for browsing Java resources.

Available Functionality

	ORCL
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.

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Compile	✓
Drop	✓
Extract	✓
Load Java	✓
Rename	✓
Report	✓

Jobs (Oracle)

The following summarizes support for this object type, making up a combination of a schedule a program, and arguments required by the program.

Creating and editing

- Jobs wizard (Oracle) and Jobs Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Drop	✓
Enable/Disable (Oracle Jobs)	✓
Extract	✓
Run Job (Oracle Jobs)	✓
Stop Job	✓

Related Information

See details for the following object types

- Chains
- Programs
- Schedules

Job Queues

Job Queues are built-in mechanisms that let you schedule a variety of SQL-based or command-line driven tasks.

Creating and editing

- Oracle Job Queue Wizard (Oracle) and Job Queue Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Disable Job	✓
Drop	✓
Enable Job (Job Queue)	✓
Extract	✓
Report	✓
Run Job (job queues)	✓

Join Indexes

Limited support is offered for these multi-table indexes.

Creating and editing

No support is offered for creation or editing of objects this type.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	SYB IQ
Drop	
Extract	✓

Libraries

Libraries are an object type 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.

Creating and editing

- Libraries Wizard (Oracle) and Libraries Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

- Logins Wizard (SQL Server) and Logins Editor (SQL Server)
- Logins Wizard (Sybase ASE) and Logins Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR	SYB ASE
Add/Modify Login Trigger		✓
Bind To Temporary Database		✓
Change Password	✓	
Create Like	✓	
Drop	✓	✓
Drop Login Trigger		✓
Extract	✓	✓
Report	✓	✓
Unbind From Temporary Database		✓

Navigator Details

Depending on the DBMS platform, the icon associated with a login differs according to the login's locked or unlocked state, when viewed in the Navigator.

DBMS Platform	Icon	Login Accont State
SQL SVR or SYB ASE	🔒	Locked
	👤	Unlocked

DBMS-specific notes

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

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.

Creating and editing

- Materialized Query Tables Wizard (DB2 LUW) and Materialized Query Tables Editor (IBM DB2 LUW)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

DB2 LUW

Materialized Query Tables

Compile	✓
Build Query	✓
Create Alias	✓
Create Like	✓
Create View	✓
Describe	✓
Drop	✓
Drop Materialized Query Table	✓
Extract	✓
Lock	✓
Refresh Table	✓
Reorganize /Reorg	✓
Report	✓
Schema (Object Action)	✓
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:

- o Complex
- o 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.

Creating and editing

- o Materialized Views Wizard (Oracle) and Materialized Views Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

- Materialized View Logs Wizard (Oracle) and Materialized View Logs Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

- Outlines Wizard (Oracle) and Outlines Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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: The Datasource Navigator splits out package headers and package bodies; however, you create both a package header and body from the packages node.

Creating and editing

- Packages Editor (IBM DB2 LUW)
- Packages Editor (IBM DB2 Z/OS)
- Packages Wizard (Oracle) and Packages Editor (Oracle)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ORCL
Bind Package		✓	
Compile			✓
Create Synonym			✓
Drop	✓		✓
Extract			✓
Flush Cache	✓		
Free (Packages)		✓	
Rebind Packages	✓	✓	
Report	✓	✓	✓

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.

Creating and editing

While Package Bodies are listed as a separate object in the Datasource Navigator, they are created on the Packages Editor in conjunction with Packages. For more information, see Package Bodies Editor (Oracle).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

See the following topics:

- Partition Functions Wizard (SQL Server) and Partition Functions Editor (SQL Server)

Available Functionality

	SQL SVR
Drop	✓
Extract	✓
Report	✓

For related information, see the following topics:

- Partition Schemes and Partition Schemes Wizard (SQL Server)
- Indexes Wizard (SQL Server)
- Tables 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.

Creating and editing

See the following topics:

- Partition Schemes Wizard (SQL Server) and Partition Schemes Editor (SQL Server)

Available Functionality

	SQL SVR
Drop	✓
Extract	✓
Next Used Filegroup	✓
Report	✓

Topics

- Partition Functions
- Partition Functions (SQL Server) - Properties
- Indexes Wizard (SQL Server)
- Tables Wizard (SQL Server)

Plans

A Plan is an executable application created in the bind process. It can include one or more packages or debris.

Creating and editing

- Plans Wizard (DB2 Z/OS) and Plans Editor (IBM DB2 Z/OS)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 z/OS
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.

Creating and editing

- Primary Keys Wizard (DB2 LUW) and Primary Keys Editor (IBM DB2 LUW)
- Primary Keys Wizard (DB2 Z/OS) and Primary Keys Editor (IBM DB2 Z/OS)
- Primary Keys Wizard (ITB/FBD) and Primary Keys editor (InterBase/Firebird)
- Indexes, Primary Keys, or Unique Keys wizard (MySQL) and Indexes, Primary Keys, and Unique Keys editors (MySQL)
- Primary Keys Wizard (Oracle) and Primary Keys Editor (Oracle)
- Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL) and Exclusion Constraints, Primary Keys, or Unique Keys Editors (PostgreSQL)
- Primary Keys Wizard (SQL Server) and Primary Keys Editor (SQL Server)
- Primary Keys Wizard (Sybase ASE) and Primary Keys Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ

Allocate Extent					✓				
Analyze					✓				
Change Status (check constraints)					✓				
Deallocate Unused Space					✓				
Disable Index							✓		
Drop	✓	✓	✓	✓	✓	✓	✓	✓	✓
Extract	✓	✓	✓	✓	✓	✓	✓	✓	✓
Rebuild Index					✓		✓		
Rename	✓	✓	✓	✓	✓	✓	✓	✓	
Reorganize /Reorg							✓		
Report	✓	✓	✓	✓	✓		✓	✓	✓

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.

Creating and editing

- Procedures Wizard (DB2 LUW) and Procedures Editor (IBM DB2 LUW)
- Procedures Wizard (DB2 Z/OS) and Procedures Editor (IBM DB2 Z/OS)
- Procedures Wizard (ITB/FBD) and Procedures editor (InterBase/Firebird)
- Procedures Wizard (Oracle) and Procedures Editor (Oracle)
- Procedures Wizard (SQL Server) and Procedures Editor (SQL Server)
- Procedures Wizard (Sybase ASE) and Procedures Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	ORCL	SQL SVR	SYB ASE	SYB IQ

Build		✓					
Compile	✓			✓			
Create Synonym				✓			
Describe	✓	✓		✓	✓	✓	
Drop	✓	✓	✓	✓	✓	✓	✓
Execute			✓	✓		✓	
Extract	✓	✓	✓	✓	✓	✓	✓
Hide Text						✓	
Object Properties							✓
Rename	✓	✓	✓	✓	✓	✓	
Report	✓	✓	✓	✓	✓	✓	
Transfer Ownership	✓						

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 ASE 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: Extended procedures are also supported. 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. Two tag pairs, ETStart and ETEnd, 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.

Creating and editing

- Profiles Wizard (Oracle) and Profiles Editor (Oracle)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Drop	✓
Extract	
Report	✓

Programs

The following summarizes support for this object type, making up a collection of metadata about a particular task.

Creating and editing

- Programs wizard (Oracle) and Programs Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Drop	✓
Enable/Disable (Oracle Jobs)	✓
Extract	✓

Related Information

See details for the following object types

- Chains
- Jobs (Oracle)
- Schedules

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.

Creating and editing

The Recycle Bin cannot be created or edited.

DBMS platform availability and object actions/operations supported

On the Datasource Navigator, selecting 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:

	ORCL
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.

Creating and editing

- Roles Wizard (ITB/FBD) and Roles editor (InterBase/Firebird)
- Roles Wizard (SQL Server) and Roles Editor (SQL Server)
- Roles Wizard (Oracle) and Roles Editor (Oracle)
- Roles Wizard (PostgreSQL) and Roles Editor (PostgreSQL)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL	ITB/FBD *	PSTGRS *	SQL SVR	SYB ASE
Create Like			✓		
Drop	✓	✓	✓	✓	✓
Extract	✓	✓	✓	✓	✓
Rename			✓		
Report	✓	✓		✓	✓
Role Activation	✓				✓

Topics

- 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 Role Activation.

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.

Creating and editing

- Rules Wizard (PostgreSQL) and Rules Editor (PostgreSQL)
- Rules Wizard (SQL Server) and Rules Editor (SQL Server)
- Rules Wizard (Sybase ASE) and Rules Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	PSTGRS *	SQL SVR	SYB ASE
Create Like	✓		
Drop	✓	✓	✓
Extract	✓	✓	✓
Hide Text			✓
Rename		✓	✓
Rename	✓		
Report		✓	✓

Schedules

The following summarizes support for the object type that implements a schedule defining when a job should be run or when a window should open.

Creating and editing

- Schedules Wizard (Oracle) and Schedules Editor (Oracle)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
Drop	✓
Extract	✓

Topics

- Chains
- Jobs (Oracle)
- Programs

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.

Creating and editing

- Schemas Wizard (PostgreSQL) and Schemas Editor (PostgreSQL)
- Schema Wizard (SQL Server) and Schemas Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	PSTGRS *	SQL SVR
Change Owner	✓	
Drop	✓	✓
Extract	✓	✓
Rename	✓	
Report		✓

Segments

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.

Creating and editing

- Segment Wizard (Sybase ASE) and Segments Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	SYB ASE
Drop	✓
Extract	✓
Report	✓

Sequences

Sequences are programmable database objects that generate a definable sequence of values.

Creating and editing

- Sequences Wizard (DB2 LUW) and Sequences Editor (IBM DB2 LUW)
- Sequences Wizard (Oracle) and Sequences Editor (Oracle)
- Sequences Wizard (SQL Server) and Sequences Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	ORCL	SQL SVR
Create Alias	✓		
Create Synonym		✓	
Drop	✓	✓	✓
Extract	✓	✓	✓
Rename	✓	✓	✓
Report	✓	✓	✓
Restart Sequence(s)			✓
Run Job (job queues)	✓		
Transfer Ownership	✓		

Shadows

Rapid SQL provides support for shadows, letting you create one or more in-sync copies of the database stored on secondary storage devices.

Creating and editing

See Shadows wizard (ITB/FBD) and Shadows editor (InterBase/Firebird).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ITB/FBD *
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.

Creating and editing

- Structured Types Wizard (DB2 LUW) and Structured Types Editor (IBM DB2 LUW)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW
Drop	✓
Extract	✓
Report	✓
Transfer Ownership	✓

Symmetric Keys

Symmetric keys, in addition to asymmetric keys and certificates, provide support for Public Key Encryption. Symmetric keys consist of a private key and corresponding public key and are used to secure symmetric keys.

Creating and editing

- Symmetric Keys Wizard (SQL Server) and Symmetric Key Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR
Drop	✓
Extract	✓
Report	✓

Related information

- Certificates
- Asymmetric Keys

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.

Creating and editing

- Synonyms Wizard (DB2 Z/OS) and Synonyms Editor (IBM DB2 Z/OS)
- Synonyms Wizard (Oracle) and Synonyms Editor (Oracle)
- Synonyms Wizard (SQL Server) and Synonyms Editor (SQL Server)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 z/OS	ORCL	SQL SVR
Create Synonym		✓	
Describe		✓	
Drop	✓	✓	✓
Extract	✓	✓	✓
Rename		✓	✓
Report	✓	✓	✓

System Indexes

On some DBMS platforms, system indexes are treated separately from standard indexes.

Platform availability

DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
						✓	✓	✓

Creating, Editing, and Object Actions Supported

Against SQL Server and Sybase ASE datasources, functionality offered for system indexes is identical to that offered for standard indexes. For details, see Indexes.

Against Sybase IQ datasources, no support is offered for creation or editing against this object type. The followng object actions are available:

- Drop
- Extract

System Tables

On some DBMS platforms, system tables are listed separately from standard tables.

Platform availability

DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
						✓	✓	✓

Creating, Editing, and Object Actions Supported

Against SQL Server and Sybase ASE datasources, functionality offered for system indexes is identical to that offered for standard indexes. For details, see Indexes.

Against Sybase IQ datasources, no support is offered for creation or editing against this object type. The followng object actions are available:

- Drop
- Extract
- Object Properties
- Select * From

System Triggers

Against Sybase IQ datasources, system triggers are listed separately from standard triggers.

Platform availability

DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
								✓

Creating, Editing, and Object Actions Supported

No support is offered for creation, editing, or object actions against this object type.

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: System tables are treated separately from user-defined tables in the Navigator to ensure that system tables are not accidentally altered or dropped.

Creating and editing

- Tables Wizard (DB2 LUW) and Tables Editor (IBM DB2 LUW)
- Tables Wizard (DB2 Z/OS) and Tables Editor (IBM DB2 Z/OS)
- Tables Wizard (ITB/FBD) and Tables editor (InterBase/Firebird)
- Tables wizard (MySQL) and Tables editor (MySQL)
- Tables Wizard (Oracle) and Tables Editor (Oracle)
- Tables Wizard (PostgreSQL) and Tables Editor (PostgreSQL)
- Tables Wizard (SQL Server) and Tables Editor (SQL Server)
- Tables Wizard (Sybase ASE) and Tables Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
Allocate Extent					✓				
Analyze					✓				
Analyze Tables				✓					
Build Query	✓	✓	✓	✓	✓	✓	✓	✓	
Change Access Status					✓				
Change Owner						✓			
Change Schema						✓			
Check Tables				✓					
Checksum Tables				✓					
Convert Tables				✓					
Create Alias	✓	✓							
Create Clone		✓							
Create Insert Statements	✓	✓	✓	✓	✓	✓	✓	✓	
Create Like	✓	✓	✓	✓	✓	✓	✓	✓	
Create Synonym					✓				
Create View	✓	✓	✓		✓		✓	✓	
DBCC							✓	✓	
Deallocate Unused Space					✓				
Delete Statistics								✓	
Describe	✓	✓		✓	✓		✓	✓	
Disable Keys				✓					
Disable/Enable Triggers							✓	✓	
Drop	✓	✓	✓		✓	✓	✓	✓	✓

Drop Clone		✓							
Drop Unused					✓				
Edit Data	✓	✓	✓	✓	✓	✓	✓	✓	
Enable Keys				✓					
Enable/Disable Filetable(s)							✓		
Estimate Size					✓			✓	
Exchange Data With Clone		✓							
Extract	✓	✓	✓		✓	✓	✓	✓	✓
Extract Data as XML					✓		✓	✓	
Flashback Table					✓				
Flush Tables				✓					
Hide Text								✓	
Import Data From File	✓	✓	✓	✓	✓		✓	✓	
Lock	✓	✓							
Move Table	✓								
Object Properties									✓
Optimize Tables				✓					
Place								✓	
Rebuild Table				✓					
Recompile							✓	✓	
Rename	✓	✓		✓	✓	✓	✓	✓	
Reorganize /Reorg	✓				✓			✓	
Repair Tables				✓					
Report	✓	✓	✓		✓		✓	✓	
Schema (Object Action)	✓	✓	✓	✓	✓		✓	✓	

Select * From	✓	✓	✓	✓	✓	✓	✓	✓	✓
Set Integrity	✓								
Shrink					✓				
Transfer Ownership	✓								
Truncate	✓	✓		✓	✓		✓	✓	
Update Statistics	✓						✓	✓	

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.

Creating and editing

- Tablespaces Wizard (DB2 LUW) and Tablespaces Editor (IBM DB2 LUW)
- Tablespaces Wizard (Oracle) and Tablespaces Editor (Oracle)
- Tablespaces Wizard (PostgreSQL) and Tablespaces Editor (PostgreSQL)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

DB2 LUW DB2 z/OS ORCL PSTGRS *

Change Owner				✓
Change Status			✓	
Coalesce			✓	
Drop	✓	✓	✓	✓
Extract	✓	✓	✓	✓
Lower High Water Mark	✓			
Rebalance	✓			
Rebuild Index		✓		
Rename	✓		✓	✓
Reorganize /Reorg		✓		
Report	✓	✓	✓	
Set Default			✓	
Set UNDO			✓	
Start Database		✓		
Stop Database		✓		
Switch Online	✓			
Transfer Ownership	✓			

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 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
- Compound triggers
- Enabled/disabled status options

Creating and editing

- Triggers Wizard (DB2 LUW) and Triggers Editor (IBM DB2 LUW)
- Triggers Wizard (DB2 Z/OS) and Triggers Editor (IBM DB2 Z/OS)
- Triggers Wizard (ITB/FBD) and Triggers editor (InterBase/Firebird)
- Triggers Wizard (Oracle) and Triggers Editor (Oracle)
- Triggers Wizard (PostgreSQL) and Triggers Editor (PostgreSQL)
- Triggers Wizard (SQL Server) and Triggers Editor (SQL Server)
- Triggers Wizard (Sybase ASE) and Triggers Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	ORCL	PSTGRS *	SQL SVR	SYB ASE

Triggers

Change Status			✓			✓	✓
Compile				✓			
Drop	✓	✓	✓	✓	✓	✓	✓
Extract	✓	✓	✓	✓	✓	✓	✓
Hide Text							✓
Report							
Rename	✓	✓	✓	✓	✓	✓	✓
Transfer Ownership	✓						

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.

Creating and editing

- Object Types Wizard (Oracle) and Types Editor (Oracle)

Note: Objects of this type cannot be created or edited against PostgreSQL datasources.

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL	PSTGRS *
Change Owner		✓
Change Schema		✓
Compile	✓	
Drop	✓	✓
Extract	✓	✓
Rename		✓
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 Navigator, they are created on the Types editor in conjunction with Types.

Creating and editing

While listed as separate objects in the Datasource Navigator, type bodies are created and edited on the **Body** tab of the Types Editor. For details, see Types Editor (Oracle).

DBMS platform availability and object actions/operations supported

The table below lists the platforms on which this object type is available. It also lists the object actions available against objects of this type For an introduction to object actions and details on usage of specific actions, see Object actions.

	ORCL
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.

Creating and editing

- Unique Keys Wizard (DB2 LUW) and Unique Keys Editor (IBM DB2 LUW)
- Unique Keys Wizard (DB2 Z/OS) and Unique Keys Editor (IBM DB2 Z/OS)
- Unique Keys Wizard (ITB/FBD) and Unique Keys editor (InterBase/Firebird)
- Indexes, Primary Keys, or Unique Keys wizard (MySQL) and Indexes, Primary Keys, and Unique Keys editors (MySQL)
- Unique Keys Wizard (Oracle) and Unique Keys Editor (Oracle)
- Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL) and Exclusion Constraints, Primary Keys, or Unique Keys Editors (PostgreSQL)
- Unique Keys Wizard (SQL Server) and Unique Keys Editor (SQL Server)
- Unique Keys Wizard (Sybase ASE) and Unique Keys Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ

Unique Keys

Allocate Extent					✓				
Analyze					✓				
Change Status					✓				
Deallocate Unused Space					✓				
Disable Index							✓		
Drop	✓	✓	✓	✓	✓	✓	✓	✓	✓
Extract	✓	✓	✓	✓	✓	✓	✓	✓	✓
Rebuild Index					✓		✓		
Rename	✓	✓	✓		✓	✓	✓	✓	
Reorganize /Reorg							✓		
Report	✓	✓	✓	✓	✓		✓	✓	
Transfer Ownership	✓								

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.

Creating and editing

- User Datatypes (DB2 LUW) and User Datatypes Editor (IBM DB2 LUW)
- User Datatypes Wizard (DB2 Z/OS) and User Datatypes Editor (IBM DB2 Z/OS)
- User Datatypes Wizard (SQL Server) and User Datatypes Editor (SQL Server)
- User Datatypes Wizard (Sybase ASE) and User Datatypes Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	SQL SVR	SYB ASE
Drop	✓	✓	✓	✓
Extract	✓	✓	✓	✓
Rename			✓	✓
Report	✓	✓	✓	✓
Transfer Ownership	✓			

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.

Creating and editing

- User Messages Wizard (SQL Server) and User Messages Editor (SQL Server)
- User Messages Wizard (Sybase ASE) and User Messages Editor (Sybase ASE)

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	SQL SVR	SYB ASE
Drop	✓	✓
Extract	✓	✓
Report	✓	✓

Users

A user is an individual with access to the DBMS.

Creating and editing

- Users Wizard (DB2 LUW) and Users Editor (IBM DB2 LUW)
- Users Wizard (DB2 Z/OS) and Users Editor (IBM DB2 Z/OS)
- Users wizard (ITB/FBD) and Users editor (InterBase/Firebird)
- Users wizard (MySQL) and Users editor (MySQL)
- Users Wizard (Oracle) and Users Editor (Oracle)
- Users Wizard (SQL Server) and Users Editor (SQL Server)
- Users Wizard (Sybase ASE) and Users Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	SQL SVR	SYB ASE	SYB ASE
Analyze					✓			
Change Password		✓		✓	✓			
Compile					✓			
Create Like	✓	✓		✓	✓	✓	✓	
Drop	✓	✓		✓	✓	✓	✓	✓
Extract	✓	✓		✓	✓	✓	✓	✓
Hide Text							✓	
Report	✓	✓		✓	✓	✓	✓	
Transfer Ownership	✓							

Navigator Details

Depending on the DBMS platform, the icon associated with a user differs according to the account's locked or unlocked state, when viewed in the Navigator.

DBMS Platform	Icon	User Accont State
ORCL or SYB IQ	🔒	Locked
	👤	Unlocked

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 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.

Creating and editing

- Views Wizard (DB2 LUW) and Views Editor (IBM DB2 LUW)
- Views Wizard (DB2 Z/OS) and Views Editor (IBM DB2 Z/OS)
- Views Wizard (ITB/FBD) and Views editor (InterBase/Firebird)
- Views Wizard (Oracle) and Views Editor (Oracle)
- Views Wizard (PostgreSQL) and Views Editor (PostgreSQL)
- Views Wizard (SQL Server) and Views Editor (SQL Server)
- Views Wizard (Sybase ASE) and Views Editor (Sybase ASE)

Note: Creation and editing of objects of this type is not supported against Sybase IQ datasources.

DBMS platform availability and object actions/operations supported

The following table lists object actions available for this object type. For an introduction to object actions and details on usage of specific actions, see Object actions.

	DB2 LUW	DB2 z/OS	ITB/FBD *	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ	SYB IQ

Views

Build Query	✓	✓	✓	✓		✓	✓		✓
Change Owner					✓				
Change Schema					✓				
Compile				✓					
Create Alias	✓	✓							
Create Synonym		✓		✓					
Describe	✓			✓	✓	✓	✓		
Drop	✓	✓	✓	✓	✓	✓	✓	✓	✓
Extract	✓	✓	✓	✓	✓	✓	✓	✓	✓
Generate Package/Procedure/Statement	✓	✓	✓	✓		✓	✓		
Hide Text							✓		
Object Properties								✓	✓
Refresh Materialized View					✓				
Rename				✓	✓	✓	✓		✓
Report	✓	✓	✓	✓		✓	✓		
Schema (Object Action)	✓	✓	✓	✓		✓	✓		✓
Select * From	✓	✓	✓	✓	✓	✓	✓	✓	✓
Transfer Ownership	✓								
Update Statistics	✓								

Creating objects

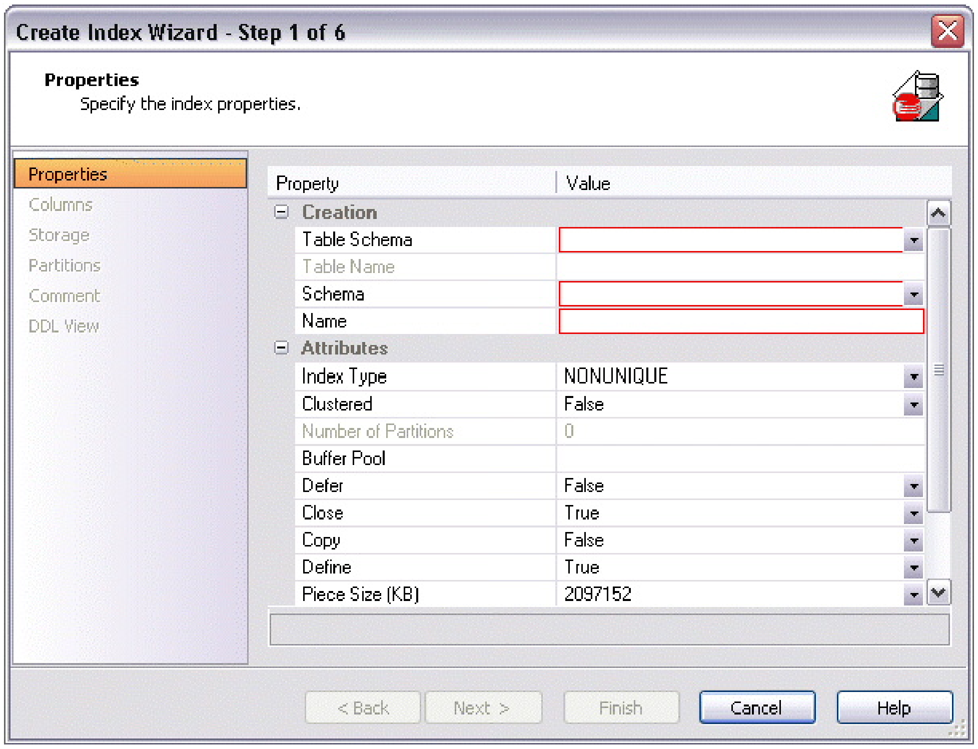
A set of easy-to-use wizards are available for creating objects. For an introduction to object creation wizards, see Overview and Common Usage of Object Wizards.

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
- PostgreSQL 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:



Property	Value
Creation	
Table Schema	
Table Name	
Schema	
Name	
Attributes	
Index Type	NONUNIQUE
Clustered	False
Number of Partitions	0
Buffer Pool	
Defer	False
Close	True
Copy	False
Define	True
Piece Size (KB)	2097152

Topics

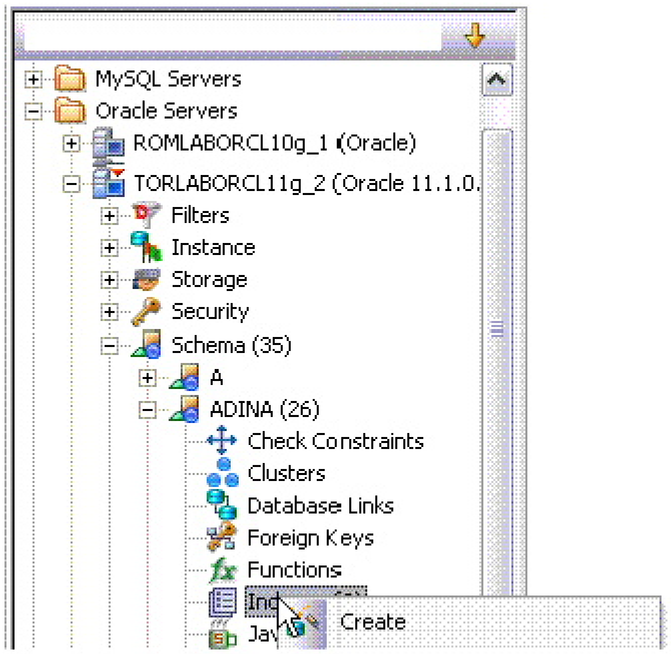
- Opening an Object Wizard
- Navigating and Setting Properties in an Object Wizard
- Adding a Comment to an Object
- Setting Permissions or Privileges for an Object
- Previewing the DDL Generated to Create the New Object.

Opening an Object Wizard

Object wizards are accessed from the Datasource Navigator pane of the 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 and Server Management.
2. On the Datasource Navigator, 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 Navigator, right-click on the specific type of object that you want to create and select **Create** from the context menu.



The object wizard for that object type opens.

Navigating and Setting 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.

Adding a Comment to an Object

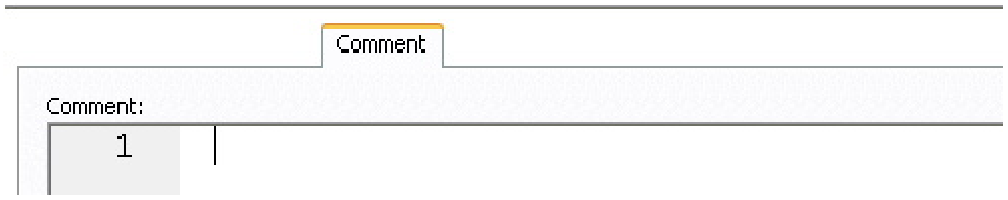
The object wizards for certain object types feature a **Comment** tab that lets you add an explanatory note to specific object definitions. Comments are stored in the REMARKS column of the object's system-catalog.

To add a comment to an object:

1. Open an object wizard on an object type that permits comments. For details, see Opening an Object Wizard.

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.

Setting Permissions or Privileges 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. For details, see Opening an Object Wizard.

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**.

Previewing the DDL Generated to Create the New Object

The final pane or tab of a wizard or editor is most commonly designated **DDL View** or **Definition**.

- o A **DDL View** tab/panel lets you view the DDL generated by your selections on the other tabs/panels of the wizard or editor.
- o 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

You create DB2 LUW objects using the following wizards:

- Aliases Wizard (DB2 LUW)
- Databases Wizard (DB2 LUW)
- Foreign Keys Wizard (DB2 LUW)
- Functions Wizard (DB2 LUW)
- Indexes Wizard (DB2 LUW)
- Materialized Query Tables Wizard (DB2 LUW)
- Nodegroups Wizard (DB2 LUW)
- Primary Keys Wizard (DB2 LUW)
- Procedures Wizard (DB2 LUW)
- Schemas Wizard (DB2 LUW)
- Sequences Wizard (DB2 LUW)
- Structured Types Wizard (DB2 LUW)
- Tables Wizard (DB2 LUW)
- Tablespaces Wizard (DB2 LUW)
- Triggers Wizard (DB2 LUW)
- Unique Keys Wizard (DB2 LUW)
- User Datatypes (DB2 LUW)
- Users Wizard (DB2 LUW)
- Views Wizard (DB2 LUW)

In addition, see Create Synonym.

Aliases 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, a CREATE ALIAS statement is generated based on the information that you supply.

To create a new alias using a wizard:

1. Open a creation wizard for an alias. For details, see Opening 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 Aliases (DB2 LUW) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

For context information on the wizard and editor that include this tab/panel, see **Aliases Wizard (DB2 LUW)** and Aliases Editor (IBM DB2 LUW).

Databases Wizard (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 Navigator, 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 Navigator.
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 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.

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, seeAdd/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 SQL** dialog.

Add/Edit Container for Tablespace

The table below describes the options and functionality on this dialog:

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 Keys Wizard (DB2 LUW)

Foreign keys are unique values that refer to specific columns of other tables. Thus, a foreign key links two tables together. The 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 a creation wizard for a foreign key. For details, see Opening 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 Foreign Keys (DB2 LUW) - Properties
   - **Column Mapping** panel - for details, see Foreign Keys (DB2 LUW) - Column Mapping
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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.

Functions Wizard (DB2 LUW)

The Functions Wizard lets you create a relationship between one set of values and another. 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, a CREATE FUNCTION statement is generated 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 a creation wizard for a function. For details, see Opening 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 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 Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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:

- o 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.
- o 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.

- o 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.

Indexes 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 a creation wizard for an index. For details, see Opening 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 Indexes (DB2 LUW) - Properties.
   - **Columns** and **Include Columns** panels - for details, see Indexes (DB2 LUW) - Columns and Include Columns.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Indexes (DB2 LUW) - Properties

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type, **Parent Schema** and **Parent Name**	Choose the type (TABLE or MATERIALIZED QUERY TABLE) owning schema, and name of the parent object 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.
Allow Reverse Scans	If selected, a ALLOW REVERSE SCAN clause is included with the CREATE/ALTER INDEX statement, specifying that the index supports both forward and reverse scans.
Compress (9.7^)	YES - Selecting this option adds a COMPRESS YES clause, enabling index compression. NO - Selecting this option adds a COMPRESS NO clause, disabling index compression. DEFAULT - Selecting this option specifies default compression handling to that of the parent object.
Percent Free	Lets you provide a PCTFREE value, specifying the percentage of each index page to leave as free space when building the index, from 0 to 99.
Minimum Percent Used	Lets you provide a MINPCTUSED value, specifyinf whether index leaf pages are merged online, and the minimum percentage threshold of space used on index leaf pages.

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 Tables 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 Navigator, 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
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, the Tables Editor opens.
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, the Material Query Tables Editor opens.
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.
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

Nodegroups 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 Navigator, find the datasource where you want to add the new Nodegroup.
2. Expand the **Storage** branch, right-click **Nodegroups**, and select **New**.

Primary Keys 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 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 a creation wizard for a primary key. For details, see Opening 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 Primary Keys (DB2 LUW) - Properties.
   - **Columns** panel - for details, see Primary Keys (DB2 LUW) - Columns.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object. For more information, see Previewing the DDL Generated to 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.
Time period policy	This setting is for use with tables set up as business-time tables. For more information, see Tables (DB2 LUW v10) - Temporal Properties. When selected, a BUSINESS_TIME WITHOUT OVERLAPS option is specified against the column or columns defining the constraint, in the generated DDL. This ensures that values for the constraint columns are unique with respect to the time for the BUSINESS_TIME period. When you select this option, columns specified as business-time begin period and end period are not available as candidates on the **Columns** tab/panel.

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.

Procedures 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. Open a creation wizard for a procedure. For details, see Opening 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 Procedures (DB2 LUW) - Properties.
    - **Advanced** panel - for details, see Procedures (DB2 LUW) - Advanced.
    - **Parameters** panel - for details, see Procedures (DB2 LUW) - Parameters.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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.
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.

Sequences 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, a CREATE SEQUENCE statement is generated from the information that you have supplied. When finished, you can compile the sequence on the target 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 sequence numbers are generated 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 Navigator, 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

Schemas 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 Navigator, 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 Types 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 Navigator, 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 Previewing the DDL Generated to Create the New Object.
4. Finally, use the **Execute** button to create the 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)

Tables 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 a creation wizard for a table. For details, see Opening 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 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.
   - **Temporal Properties** panel - for details, see Tables (DB2 LUW v10) - Temporal Properties.
   - **Constraints** panel - for details, see Tables (DB2 LUW) - Constraints.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **Permissions** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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.

Row Properties (V10) settings are available when you select a **Type** of TIMESTAMP. These settings are used in conjunction with settings on the **Temporal Properties** tab in setting up system and application temporal tables. For more information, see Tables (DB2 LUW v10) - Temporal Properties.

Specific settings are:

- **As Row Begin** - adds a GENERATED ALWAYS AS ROW BEGIN option to the column specification, used to track when a row is current.
- **As Row End** - adds a GENERATED ALWAYS AS ROW END option to the column specification.
- **Transaction Start ID** - adds a GENERATED ALWAYS AS TRANSACTION START ID option to the column specification indicating that this column captures the start times for transactions that affect rows.

A column can have only one of the three settings selected.

When you select a **Row Properties** setting, the **Scale** value and if appropriate, the **Allow Nulls**, **Default Value**, and **Expression** settings are automatically adjusted for compatibility with row generation. Similarly, if one of the three **Row Properties** settings is selected for a column, and any of the **Scale**, **Allow Nulls**, **Default Value**, or **Expression** settings are modified, then the selected **Row Properties** setting is cleared.

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 Indexes Wizard (DB2 LUW).

Tables (DB2 LUW v10) - Temporal Properties

This tab/panel lets you specify temporal properties to a table, designating it as a table that records the period of time when a row is valid.

Note: Before using this object action, consult DB2 LUW documentation for details on setup and use of the temporal tables feature. For more information, see Accessing Third Party Documentation.

This tab/panel has the following controls:

- **System-time table** - selecting this option creates a table with the PERIOD SYSTEM_TIME clause, designating it as a system-period temporal table. This option is available if the table has three columns of type TIMESTAMP, and the following prerequisites have been met:
- At least one TIMESTAMP-typed column has the **As Row Begin** property selected
- At least one TIMESTAMP-typed column has the **As Row End** property selected
- At least one TIMESTAMP-typed column has the **Transaction Start ID** property selected

For details on column definitions, see Tables (DB2 LUW) - Columns.

The **Row-begin column**, **Row-end column**, and **Transaction Start ID column** controls are automatically populated with the columns having the corresponding property.

When you select this option, in the generated DDL, the **Row-begin column** and **Row-end column** values are used as the PERIOD SYSTEM_TIME column parameters, defining the system period.

- **Business-time table** - selecting this option creates a table with the PERIOD BUSINESS_TIME clause, designating it as an application-period temporal table. This option is available if the table has two columns of type DATE or TIMESTAMP, for which no **Row Properties** have been assigned.

The **Begin column** and **End Column** controls are automatically populated with the valid candidate columns from the table's column list and the first distinct column pair is automatically chosen for convenience.

- **Use history table** - this set of controls is only active when **Business-time table** is selected. It lets you associate a history table. When you select **Use history table**, the Owner and Name controls are activated, letting you identify the history table to associate with this table.

Note that a constraint created on a business-time table can enforce the uniqueness of the data for any point in business time via the **Time period policy** property that has been added in the constraint wizards. For details, see the following topics:

- Primary Keys (DB2 LUW) - Properties
- Unique Keys (DB2 LUW) - Properties

Tables (DB2 LUW) - Constraints

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- Primary Keys Wizard (DB2 LUW)
- Unique Keys Wizard (DB2 Z/OS)
- Foreign Keys Wizard (DB2 Z/OS))

Tablespaces 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 a creation wizard for a tablespace. For details, see Opening 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 Tablespaces (DB2 LUW) - Properties.
   - **Container** panel - for details, see Tablespaces (DB2 LUW) - Container.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **Permissions** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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.

Triggers 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 a creation wizard for a trigger. For details, see Opening 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 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.

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.
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.

Unique Keys 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.

To create a new unique key using a wizard:

1. Open a creation wizard for a unique key. For details, see Opening 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 Unique Keys (DB2 LUW) - Properties.
   - **Columns** panel - for details, see Unique Keys (DB2 LUW) - Columns.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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.
Time period policy	This setting is for use with tables set up as business-time tables. For more information, see Tables (DB2 LUW v10) - Temporal Properties. When selected, a BUSINESS_TIME WITHOUT OVERLAPS option is specified against the column or columns defining the constraint, in the generated DDL. This ensures that values for the constraint columns are unique with respect to the time for the BUSINESS_TIME period. When you select this option, columns specified as business-time begin period and end period are not available as candidates on the **Columns** tab/panel.

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 Datatypes (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. Open a creation wizard for a user datatype. For details, see Opening 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 Datatypes (DB2 LUW) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

User Datatypes (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.

Users 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 a creation wizard for a user. For details, see Opening 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 Users (DB2 LUW) - Properties.
   - **Object Permissions** and **System Permissions** panels - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Users (DB2 LUW) - Properties

When creating or editing a user datatype, this tab/panel lets you provide the user **Name**.

Views 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. Open a creation wizard for a view. For details, see Opening 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 Views (DB2 LUW) - Properties.
   - **Definition** panel - for details, see Views (DB2 LUW) - Definition.
3. Finally, use the **Execute** button to create the object.

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.

IBM DB2 for z/OS Object Wizards

You create DB2 OS390 objects using the following wizards:

- Aliases Wizard (DB2 Z/OS)
- Databases Wizard (DB2 Z/OS)
- Foreign Keys Wizard (DB2 Z/OS))
- Functions Wizard (DB2 Z/OS)
- Indexes Wizard (DB2 Z/OS)
- Plans Wizard (DB2 Z/OS)
- Primary Keys Wizard (DB2 Z/OS)
- Procedures Wizard (DB2 Z/OS)
- Stogroups Wizard (DB2 Z/OS)
- Synonyms Wizard (DB2 Z/OS)
- Tables Wizard (DB2 Z/OS)
- Tablespaces Wizard (DB2 Z/OS)
- Triggers Wizard (DB2 Z/OS)
- Unique Keys Wizard (DB2 Z/OS)
- User Datatypes Wizard (DB2 Z/OS)
- Users Wizard (DB2 Z/OS)
- Views Wizard (DB2 Z/OS)

In addition, see Create Synonym.

Aliases 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, a CREATE ALIAS statement based on the information that you supply is generated. 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 a creation wizard for an alias. For details, see Opening 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 Aliases (DB2 z/OS) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Databases 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 a creation wizard for a database. For details, see Opening 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 Databases (DB2 z/OS) - Properties.
   - **Permissions** panel - for details, see Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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 Keys 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. The 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 a creation wizard for a foreign key. For details, see Opening 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 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 Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Foreign Keys (DB2 z/OS) - 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 (DB2 Z/OS)

The Functions Wizrd lets you create a relationship between one set of values and another. 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, a CREATE FUNCTION statement is generated 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 a creation wizard for a function. For details, see Opening 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, 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 Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Functions (DB2 z/OS) - Properties

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.
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.

Indexes 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 a creation wizard for an index. For details, see Opening 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 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 Adding a Comment to an Object.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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)
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 keyrandomization. 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: 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.

Plans 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 Navigator, 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**.	
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.

6 **Enable** or **Disable** system connection types that can use the plan or package, select a **System** or **Cname** option.

Primary Keys 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 a creation wizard for a primary key. For details, see Opening 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 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 Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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
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.
Time period policy	This setting is for use with tables set up as business-time tables. For more information, see Tables (DB2 z/OS v10) - Temporal Properties. When selected, a BUSINESS_TIME WITHOUT OVERLAPS option is specified against the column or columns defining the constraint, in the generated DDL. This ensures that values for the constraint columns are unique with respect to the time for the BUSINESS_TIME period. When you select this option, columns specified as business-time begin period and end period are not available as candidates on the **Columns** tab/panel.

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.
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.

Procedures 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 a creation wizard for a procedure. For details, see Opening 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 Procedures (DB2 z/OS) - Properties.
   - **Parameters** panel - for details, see Procedures (DB2 z/OS) - Parameters.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Important Notes

o 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.

Stogroups 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 Navigator, 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:

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:

Required Information	Description
Volumes	Lets you select volumes.
OK	Click to add volumes to Stogroup Wizard.

Synonyms 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 a creation wizard for a synonym. For details, see Opening 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 Synonyms (DB2 z/OS) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Tables 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. Open a creation wizard for a table. For details, see Opening 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 Tables (DB2 z/OS) - Properties.
   - **Columns** panel - for details, see Tables (DB2 z/OS) - Columns.
   - **Indexes** panel - for details, see Tables (DB2 z/OS) - Indexes.
   - **Temporal Properties** panel - for details, see Tables (DB2 z/OS v10) - Temporal Properties.
   - **Constraints** panel - for details, see Tables (DB2 z/OS) - Constraints.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **Permissions** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.

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) - 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.

Row Properties (V10) settings are available when you select a **Type** of TIMESTAMP. These settings are used in conjunction with settings on the **Temporal Properties** tab in setting up system and application temporal tables. For more information, see Tables (DB2 z/OS v10) - Temporal Properties.

Specific settings are:

- **As Row Begin** - adds a GENERATED ALWAYS AS ROW BEGIN option to the column specification, used to track when a row is current.
- **As Row End** - adds a GENERATED ALWAYS AS ROW END option to the column specification.
- **Transaction Start ID** - adds a GENERATED ALWAYS AS TRANSACTION START ID option to the column specification indicating that this column captures the start times for transactions that affect rows.

A column can have only one of the three settings selected.

When you select a **Row Properties** setting, the **Scale** value and if appropriate, the **Allow Nulls**, **Default Value**, and **Expression** settings are automatically adjusted for compatibility with row generation. Similarly, if one of the three **Row Properties** settings is selected for a column, and any of the **Scale**, **Allow Nulls**, **Default Value**, or **Expression** settings are modified, then the selected **Row Properties** setting is cleared.

Tables (DB2 z/OS) - Indexes

Click **Add** to open the Indexes Wizard (DB2 Z/OS).

Tables (DB2 z/OS v10) - Temporal Properties

This tab/panel lets you specify temporal properties to a table, designating it as a table that records the period of time when a row is valid.

Note: Before using this object action, consult DB2 LUW documentation for details on setup and use of the temporal tables feature. For more information, see Accessing Third Party Documentation.

This tab/panel has the following controls:

- **System-time table** - selecting this option creates a table with the PERIOD SYSTEM_TIME clause, designating it as a system-period temporal table. This option is available if the table has three columns of type TIMESTAMP, and the following prerequisites have been met:
- At least one TIMESTAMP-typed column has the **As Row Begin** property selected
- At least one TIMESTAMP-typed column has the **As Row End** property selected
- At least one TIMESTAMP-typed column has the **Transaction Start ID** property selected

For details on column definitions, see Tables (DB2 z/OS) - Columns.

The **Row-begin column**, **Row-end column**, and **Transaction Start ID column** controls are automatically populated with the columns having the corresponding property.

When you select this option, in the generated DDL, the **Row-begin column** and **Row-end column** values are used as the PERIOD SYSTEM_TIME column parameters, defining the system period.

- **Business-time table** - selecting this option creates a table with the PERIOD BUSINESS_TIME clause, designating it as an application-period temporal table. This option is available if the table has two columns of type DATE or TIMESTAMP, for which no **Row Properties** have been assigned.

The **Begin column** and **End Column** controls are automatically populated with the valid candidate columns from the table's column list and the first distinct column pair is automatically chosen for convenience.

- **Use history table** - this set of controls is only active when **Business-time table** is selected. It lets you associate a history table. When you select **Use history table**, the Owner and Name controls are activated, letting you identify the history table to associate with this table.

Note that a constraint created on a business-time table can enforce the uniqueness of the data for any point in business time via the **Time period policy** property that has been added in the constraint wizards. For details, see the following topics:

- Primary Keys (DB2 z/OS) - Properties
- Unique Keys (DB2 z/OS) - Properties

Tables (DB2 z/OS) - Constraints

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- Foreign Keys Wizard (DB2 Z/OS))
- Primary Keys Wizard (DB2 Z/OS)
- Unique Keys Wizard (DB2 Z/OS)
- Create Synonym

Tablespaces 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 Navigator, 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 - Setting Permissions or Privileges for an Object
    - **DDL View** panel - for details, see Previewing the DDL Generated to 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. Refer to DB2 z/OS product documentation for detailed information on these types and other required settings. For more information, see Accessing Third Party Documentation.

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.
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.

Triggers 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 a creation wizard for a trigger. For details, see Opening 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 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.

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.

Unique Keys 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.

To create a new unique key using a wizard:

1. Open a creation wizard for a unique key. For details, see Opening 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 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 Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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
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.
Time period policy	This setting is for use with tables set up as business-time tables. For more information, see Tables (DB2 z/OS v10) - Temporal Properties. When selected, a BUSINESS_TIME WITHOUT OVERLAPS option is specified against the column or columns defining the constraint, in the generated DDL. This ensures that values for the constraint columns are unique with respect to the time for the BUSINESS_TIME period. When you select this option, columns specified as business-time begin period and end period are not available as candidates on the **Columns** tab/panel.

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 Datatypes 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 a creation wizard for a user datatype. For details, see Opening 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 Datatypes (DB2 z/OS) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Users 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 a creation wizard for a user. For details, see Opening 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 Users (DB2 z/OS) - Properties.
    - **Object Permissions** and **System Permissions** panels - Setting Permissions or Privileges for an Object
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Users (DB2 z/OS) - Properties

When creating or editing a user datatype, this tab/panel lets you provide the user **Name**.

Views 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 a creation wizard for a view. For details, see Opening 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 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.

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.

InterBase/Firebird Object Wizards

A set of wizards let you create all supported InterBase/Firebird objects. For details, see the following topics:

- Blob Filters Wizard (ITB/FBD)
- Domains Wizard (ITB/FBD)
- Encryption Keys wizard (ITB/FBD)
- Exceptions Wizard (ITB/FBD)
- External Functions Wizard (ITB/FBD)
- Foreign Keys Wizard (ITB/FBD)
- Generators Wizard (ITB/FBD)
- Indexes Wizard (ITB/FBD)
- Primary Keys Wizard (ITB/FBD)
- Procedures Wizard (ITB/FBD)
- Roles Wizard (ITB/FBD)
- Shadows wizard (ITB/FBD)
- Tables Wizard (ITB/FBD)
- Triggers Wizard (ITB/FBD)
- Unique Keys Wizard (ITB/FBD)
- Users wizard (ITB/FBD)
- Views Wizard (ITB/FBD)

Blob Filters Wizard (ITB/FBD)

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 creation wizard for a blob filter. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Blob Filters (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for a domain. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Domains (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for an encryption key. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Encryption Keys (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for an exception. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Exceptions (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for an external function. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Parameters** panel - for details, see External Functions (ITB/FBD) - Parameters.
   - **Return Type** panel - for details, see External Functions (ITB/FBD) - Return Type.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

External Functions (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for a foreign key. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties
   - **Column Mapping** panel - for details, see Foreign Keys (ITB/FBD) - Column Mapping
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Foreign keys (ITB/FBD) - 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.

Foreign Keys (ITB/FBD) - 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. Then, under **Referenced Table**, select the corresponding column check boxes.

Generators Wizard (ITB/FBD)

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 creation wizard for a generator. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Generators (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for an index. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Columns** panel - for details, see Indexes (ITB/FBD) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Indexes (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for a primary key. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Columns** panel - Primary Keys (ITB/FBD) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Primary Keys (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD)

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. Open an object creation wizard for a procedure. For details. see Opening an Object Wizard on page406.
2. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:
    - **Properties** panel - Procedures (ITB/FBD) - Properties.
    - **Parameters** panel - Procedures (ITB/FBD) - Parameters.
    - **Body** panel - Procedures (ITB/FBD) - Body.
    - **Permissions** panel - Setting Permissions or Privileges for an Object
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Procedures (ITB/FBD) - Properties

When creating a procedure, this panel/tab lets you provide a **Name** for the procedure. After creation, you cannot change its name.

Procedures (ITB/FBD) - 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 (ITB/FBD) - Body

This tab provides a simple editor that lets you add or modify the body of the procedure.

Roles Wizard (ITB/FBD)

The Roles Wizard lets you create a basic InterBase/Firebird role.

To create a new role using a wizard:

1. Open an object creation wizard for a role. For details. see Opening an Object Wizard on page406.
2. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:
   - **Properties** panel - Roles (ITB/FBD) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object. For more information, see Previewing the DDL Generated to 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 (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for a shadow. For details. see Opening an Object Wizard on page406.
2. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:
    - **Properties** panel - Shadows (ITB/FBD) - Properties.
    - **Storage** panel - Shadows (ITB/FBD) - Storage.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Shadows (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD)

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. Open an object creation wizard for a table. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Columns** panel - for details, see Tables (ITB/FBD) - Columns.
   - **Indexes** panel - for details, see Tables (ITB/FBD) - Indexes.
   - **Constraints** panel - for details, see Tables (ITB/FBD) - Constraints.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **Permissions** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Tables (ITB/FBD) - 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 (ITB/FBD) - 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.
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 (ITB/FBD) - 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 (ITB/FBD).
- 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 (ITB/FBD) - 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.

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 Keys Wizard (DB2 LUW) and Foreign Keys editor (InterBase/Firebird)
- Primary Keys Wizard (ITB/FBD) and Primary Keys editor (InterBase/Firebird)
- Unique Keys Wizard (ITB/FBD) 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 (ITB/FBD)

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 creation wizard for a trigger. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Body** panel - for details, see Triggers (ITB/FBD) - Body.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Triggers (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD)

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 creation wizard for a unique key. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Columns** panel - Unique Keys (ITB/FBD) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Unique Keys (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD)

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 working with 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 creation wizard for a user. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **Roles** panel - for details, see Users (ITB/FBD) - Roles.
   - **Object Permissions** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

The User editor opens. For more information, see Users editor (InterBase/Firebird).

Users (ITB/FBD) - 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 (ITB/FBD) - 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.
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 (ITB/FBD) - 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 (ITB/FBD) - Roles.

To assign roles to the user:

Select the associated check box.

Views Wizard (ITB/FBD)

The Views wizard lets you create a basic InterBase/Firebird view

To create a new view using a wizard:

1. Open an object creation wizard for a view. For details. see Opening an Object Wizard on page406.
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 (ITB/FBD) - Properties.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **Definition** panel - for details, see Views (ITB/FBD) - Definition.
3. Finally, use the **Execute** button to create the object.

Views (ITB/FBD) - 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.
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 (ITB/FBD) - 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

You create SQL Server objects using the following wizards:

- Asymmetric Keys Wizard (SQL Server)
- Certificate Wizard (SQL Server)
- Databases Wizard (SQL Server)
- Database Triggers Wizard (SQL Server)
- Defaults Wizard (SQL Server)
- Extended Procedures Wizard (SQL Server)
- Foreign Keys Wizard (SQL Server)
- Full-text Catalogs Wizard (SQL Server)
- Full-text Indexes Wizard (SQL Server)
- Functions Wizard (SQL Server)
- Indexes Wizard (SQL Server)
- Logins Wizard (SQL Server)
- Partition Functions Wizard (SQL Server)
- Partition Schemes Wizard (SQL Server)
- Primary Keys Wizard (SQL Server)
- Procedures Wizard (SQL Server)
- Roles Wizard (SQL Server)
- Rules Wizard (SQL Server)
- Schema Wizard (SQL Server)
- Sequences Wizard (SQL Server)
- Symmetric Keys Wizard (SQL Server)
- Synonyms Wizard (SQL Server)
- Tables Wizard (SQL Server)
- Triggers Wizard (SQL Server)
- Unique Keys Wizard (SQL Server)
- User Messages Wizard (SQL Server)

- o Users Wizard (SQL Server)
- o User Datatypes Wizard (SQL Server)
- o Views Wizard (SQL Server)

Asymmetric Keys Wizard (SQL Server)

This wizard lets you build and submit a CREATE ASYMMETRIC KEY statement with options corresponding to the contained private and public keys.

To create a new asymmetric key using a wizard:

1. Open a creation wizard for an asymmetric key. For details, see Opening 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 - Asymmetric Keys (SQL Server) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Asymmetric Keys (SQL Server) - Properties

When creating or editing an asymmetric key, this tab/panel lets you work with the following settings:

Setting	Availability and description	
Authorization Owner	The name of the user that will own the certificate.	
Name	The name for the certificate.	
Use Existing Keys	Selected, a WITH ALGORITHM option variation of the CREATE ASYMMETRIC KEY statement is generated. Deselected, a FROM FILE, FROM EXECUTABLE FILE, FROM ASSEMBLY, or FROM PROVIDER option variation of the CREATE ASYMMETRIC KEY statement is generated.	
Provider Name	Available if **Use Existing Keys** is not selected.	Lets you provide a FROM PROVIDER argument value, specifying an Extensible Key Management provider and name.
Key Algorithm	Available if **Use Existing Keys** is not selected.	Lets you select a WITH ALGORITHM value of RSA_512, RSA_1024, or RSA_2048, specifying the key length.
Key Name In Provider	Available if **Use Existing Keys** is not selected, after **Provider Name** is entered.	Lets you provide a PROVIDER_KEY_NAME value, specifying the key name from the external provider.
Creation Disposition		Lets you select a CREATION_DISPOSITION value of OPEN_EXISTING or CREATE_NEW, specifying whether the asymmetric key is mapped to an existing EKM key or a new key is created on the EKM device.
File Executable File Assembly Provider Name	Available if **Use Existing Keys** is selected.	Let you provide FROM FILE, FROM EXECUTABLE FILE, FROM ASSEMBLY, or FROM PROVIDER option values, respectively.
Encryption Password	Available if **Use Existing Keys** is not selected and after **Subject** is provided.	Lets you provide an ENCRYPTION BY PASSWORD = clause value, specifying the password used to encrypt the private key.

Certificate Wizard (SQL Server)

This wizard lets you build and submit a CREATE CERTIFICATE statement, adding a certificate to a database.

To create a new certificate using a wizard:

1. Open a creation wizard for a certificate. For details, see Opening 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 - Certificates (SQL Server) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Certificates (SQL Server) - Properties

When creating or editing a certificate, this tab/panel lets you work with the following settings:

Setting	Availability and description	
Owner	The name of the user that will own the certificate.	
Name	The name for the certificate.	
Use Existing Keys	Lets you choose between EXISTING KEYS and GENERATE NEW KEYS options.	
Active For Begin Dialog	Lets you select ON/OFF values for a ACTIVE FOR BEGIN_DIALOG = option.	
Subject	Available if **Use Existing Keys** is not selected.	Lets you provide a WITH SUBJECT = clause value, referring to a field in the certificate metadata.
StartDate and **ExpiryDate**	Available if **Use Existing Keys** is not selected and after **Subject** is provided.	Lets you provide START_DATE = and EXPIRY_DATE = values, defining the period that the certificate is valid
Assembly ExecutableFile File	Available if **Use Existing Keys** is selected.	These controls let you select among FROM ASSEMBLY =, FROM FILE =, FROM EXECUTABLE FILE = clauses and provide the relevant signed assembly, path to a DER-encoded certificate file, or path to a private key file.
PrivateKeyFile	Available if **Use Existing Keys** is selected.	Lets you provide a WITH PRIVATE KEY (FILE =... clause value, specifying that the certificate's private key is loaded.
Encryption Password	Available if **Use Existing Keys** is not selected and after **Subject** is provided.	Lets you provide an ENCRYPTION BY PASSWORD = clause value, specifying the password used to encrypt the private key.
DecryptionPassword	Available if **Use Existing Keys** is not selected. Also available if **Use Existing Keys** is selected and after an **Assembly** or **PrivateKeyFile** value is provided.	Lets you provide a DECRYPTION BY PASSWORD = clause value, specifying the password used to decrypt a private key retrieved from file.

Databases 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 a creation wizard for a database. For details, see Opening an Object Wizard.
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 Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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: 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 a creation wizard for a database trigger. For details, see Opening 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 Database Triggers (SQL Server) - Properties.
   - **Trigger Events** panel - for details, see Database Triggers (SQL Server) - Trigger Events.
   - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Defaults 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. Open a creation wizard for a default. For details, see Opening 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 - Defaults (SQL Server) - Properties.
   - **Dependencies** panel - Defaults (SQL Server) - Dependencies.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Defaults (SQL Server) - Properties

When creating or editing a default, this tab/panel lets you work with the following settings:

Setting	Description
Schema	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 Procedures 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 a creation wizard for an extended procedure. For details, see Opening 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 - Extended Procedures (SQL Server) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 Keys Wizard (SQL Server)

Foreign keys are unique values that refer to specific columns of other tables. Thus, a foreign key links two tables together. The 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 a creation wizard for a foreign key. For details, see Opening 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 - Foreign Keys (SQL Server) - Properties.
   - **Column Mapping** panel - Foreign Keys (SQL Server) - Column Mapping.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
With NOCHECK	Specifies whether the data in the table is validated against a newly added or re-enabled 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. 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: 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 a creation wizard for afull-text catalog. For details, see Opening 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 Full-text Catalogs (SQL Server) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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	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	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: 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. Open a creation wizard for a full-text index. For details, see Opening 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 Full-text Indexes (SQL Server) - Properties.
    - **Columns** panel - for details, see Full-text Indexes (SQL Server) - Columns.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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 Full-text Catalogs Wizard (SQL Server). 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.

Functions 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 a creation wizard for a function. For details, see Opening 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 Functions (SQL Server) - Properties.
   - **Definition** panel - for details, see Functions (SQL Server) - Definition.
3. Finally, use the **Execute** button to create the 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.

Indexes 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 a creation wizard for an index. For details, see Opening 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 - Indexes (SQL Server) - Properties.
   - **Columns** panel - Indexes (SQL Server) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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 Schemes 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.
Fill Factor	Specifies the percentage that the Database Engine should make the leaf level of each index page when the index is created or rebuilt.
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.

Logins 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. Open a creation wizard for alogin. For details, see Opening 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 Logins (SQL Server) - Properties.
   - **Server Roles** panel - for details, see Logins (SQL Server) - Server Roles.
   - **Users** panel - for details, see Logins (SQL Server) - Users.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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>]).
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 (or common user-defined roles when selecting multiple databases) 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 Users 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. Open a creation wizard for a partition function. For details, see Opening 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 Partition Functions (SQL Server) - Properties.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

For related information, see the following topics:

- Partition Schemes Wizard (SQL Server)
- Indexes Wizard (SQL Server)
- Tables Wizard (SQL Server)

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 Schemes 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 Accessing Third Party Documentation.

To create a new partition wizard using a wizard:

1. Open a creation wizard for a partition scheme. For details, see Opening 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 Partition Schemes (SQL Server) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

For related information, see the following topics:

- Partition Functions Wizard (SQL Server)
- Indexes Wizard (SQL Server)
- Tables 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 Keys 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. Open a creation wizard for a primary key. For details, see Opening 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 - Primary Keys (SQL Server) - Properties.
   - **Columns** panel - Primary Keys (SQL Server) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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.

Procedures 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 a creation wizard for a procedure. For details, see Opening 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 Procedures (SQL Server) - Properties.
   - **Definition** panel - for details, see Procedures (SQL Server) - Definition.
3. Finally, use the **Execute** button to create the 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.

Roles 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 Opening 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 Previewing the DDL Generated to 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.

Rules 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. Open a creation wizard for a rule. For details, see Opening 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 - Rules (SQL Server) - Properties.
   - **Dependencies** panel - Rules (SQL Server) - Dependencies.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

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)

This wizard lets you create a schema on Microsoft SQL Server.

To create a new schema using a wizard:

1. Open a creation wizard for a schema. For details, see Opening 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 - Schema (SQL Server) - Properties.
   - **Permissions** panel - Setting Permissions or Privileges for an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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**.

Sequences Wizard (SQL Server)

This wizard builds and submits a CREATE SEQUENCE statement.

To create a new full-text catalog using a wizard:

1. Open a creation wizard for a sequence. For details, see Opening 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 Full-text Catalogs (SQL Server) - Properties.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Sequences (SQL Server) - Properties

When creating or editing a sequence, this tab/panel lets you work with the following settings:

Setting	Description
Schema and **Name**	Lets you provide identification for the sequence.
Start With	Lets you provide a START WITH argument value, specifying the first value to be returned by the sequence.
Increment By	Lets you provide an INCREMENT BY argument value, specifying the value by which to increment (or decrement if negative) the sequence object for each call to the NEXT VALUE FOR function.
Minimum Value and **Maximum Value**	Lets you provide MINVALUE and MAXVALUE argument values, specifying the bounds for this sequence.
Cycle	If selected, a CYCLE argument is generated, specifying that sequence values should restart from the minimum value (or maximum for descending sequences) or throw an exception when the minimum/maximum value is exceeded. If unselected, the default NO CYCLE is assumed.
Cache and Cache Size	If **Cache** is unselected, a NO CACHE argument is generated. If **Cache** is selected, a CACHE argument is generated. A CACHE argument value can be provided using the Cache Size control, and if no Cache Size value is provided, SQL Server will generate a value.
Data Type Schema, **Data Type Name**, and **Data Type Precision**	These controls let you build an AS argument, specifying datatype details for the new sequence. If no values are selected/provided, SQL Server will, by default, generate a sequence object of type BIGINT. After selecting a **Data Type Schema**, the **Data Type Name** dropdown is populated with user datatypes created from a base datatype of BIGINT, TINYINT, SMALLINT, INT, BIGINT, DECIMAL, or NUMERIC. **Data Type Precision** is a read-only field displaying the precision of the selected user datatype. **NOTE:** You can use system types if you select a **Data Type Schema** of SYS.

After creation, the Sequences Editor lets you view the current value of the sequence and whether the maximum value has been exceeded. For details. see Sequences Editor (SQL Server).

Symmetric Keys Wizard (SQL Server)

This wizard lets you build and submit a CREATE SYMMETRIC KEY statement.

To create a new symmetric key using a wizard

1. Open a creation wizard for a symmetric key. For details, see Opening 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 - Symmetric Keys (SQL Server) - Properties.
   - **Encryption Mechanisms** panel - Symmetric Keys (SQL Server) - Encryption Mechanisms.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Symmetric Keys (SQL Server) - Properties

When creating or editing an asymmetric key, this tab/panel lets you work with the following settings:

Setting	Description
Authorization Owner	The name of the user that will own the symmetric key.
Name	The name for the symmetric key
Provider Name	Lets you provide a FROM PROVIDER argument value, specifying an Extensible Key Management provider and name.
Key Algorithm	Lets you select a WITH ALGORITHM value of DES, AES_192, RC2, RC4_128, DESX, AES_128, TRIPLE_DES, or TRIPLE_DES_3KEY, specifying the encryption algorithm.
Pass Phrase	Lets you provide a KEY_SOURCE argument value, specifying a pass phrase from which to derive the key.
Identity Phrase	Lets you provide an IDENTITY_VALUE argument value, specifying an identity phrase from which to generate a GUID that tags data encrypted with a temporary key.
Key Name In Provider	Lets you provide a PROVIDER_KEY_NAME value, specifying the key name from the external provider.
Creation Disposition	Lets you select a CREATION_DISPOSITION value of OPEN_EXISTING or CREATE_NEW, specifying whether the symmetric key is mapped to an existing EKM key or a new key is created on the EKM device.

For context information such as opening the wizard or editor for this object type, see Synonyms Wizard (SQL Server).

Symmetric Keys (SQL Server) - Encryption Mechanisms

When creating or editing a symmetric key, this tab/panels lets you provide an ENCRYPTION BY... argument, and indicate the specific asymmetric key, symmetric keys, or certificate used to encrypt the key being created or the password from which to derive a TRIPLE_DES key.

The **Used Encryption Mechanisms** lists the symmetric key, asymmetric key, certificate, or password options currently included in the ENCRYPTION BY argument. Selecting ASYMMETRIC KEYS, CERTIFICATES, PASSWORD, or SYMMETRIC KEYS from the **Encryption Type** dropdown displays the list of available elements of that type in the **Existing Encryption Mechanisms** list.

To specify encrypting mechanisms for a symmetric key

1. From the **Encryption Type** dropdown, select an option among ASYMMETRIC KEYS, CERTIFICATES, PASSWORD, or SYMMETRIC KEYS.
2. Take one of the following actions:

- Select CERTIFICATE or ASYMMETRIC KEY from the **Encryption Type** dropdown to display elements of that type in the **Existing Encryption Mechanisms** list. Select a specific certificate or asymmetric key and click **Add** to move that element to the **Used Encryption Mechanisms** list.
- Select SYMMETRIC KEY from the **Encryption Type** dropdown to display elements of that type in the **Existing Encryption Mechanisms** list. Select a specific symmetric key and click **Add** to open the **Open Key** dialog. For more information on opening and decrypting the key, see Opening a Symmetric Key.
- Select PASSWORD from the **Encryption Type** dropdown and click **Add** to open the **Add Password** dialog. That dialog lets you add an ENCRYPTION BY PASSWORD argument. Use the **Password** control to provide the ENCRYPTION BY PASSWORD value. Click **OK** to add the password to the **Used Encryption Mechanisms** list.

3. Repeat step1 and step2 to add more encrypting mechanisms for this symmetric key.

To remove an encryption mechanism from an asymmetric key

1. Select an item from the **Used Encryption Mechanisms** list.
2. Click **Remove**.

For context information such as opening the wizard or editor for this object type, see Synonyms Wizard (SQL Server).

For information on creating asymmetric keys or certificates to be used with the encryption method for this key, see Asymmetric Keys Wizard (SQL Server) and Certificate Wizard (SQL Server).

Opening a Symmetric Key

When specifying a symmetric keys as the encryption mechanism, you must decrypt the symmetric key and make it available for use. The **Open Key** dialog lets you build an OPEN SYMMETRIC KEY that will precede the CREATE SYMMETRIC KEY statement submitted to create or edit a symmetric key.

Use the **Certificate** control to provide a DECRYPTION BY CERTIFICATE value and the **Password** control to provide a WITH PASSWORD value. Click **Open Key** to add the symmetric key to the **Used Encryption Mechanisms** list.

Note: This functionality is available for Microsoft SQL Server only.

To open a symmetric key

1. Open a creation wizard or editor for a symmetric key. For details, see Opening an Object Wizard and Opening an Object Editor.
2. Select the **Encryption Mechanisms** tab.

3. Select **Symmetric Keys** from the **Encryption Type** dropdown, select a symmetric key from the **Existing Encryptions Mechanisms** list, and then click **Add**.

The **Open Key** dialog opens.

4. Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks	
Action options	Lets you work with the following settings:	
	Decryption Mechanism and **Certificate/ Symmetric Key/Asymmetric Key/Password**	These controls let you build the DECRYPTION BY option, specifying the decryption mechanism and the specific certificate, asymmetric key, symmetric key or password to be used.
	Password	Lets you provide the BY PASSWORD or WITH PASSWORD values for a **Decryption Mechanism** of CERTIFICATE, ASYMMETRIC KEY, or PASSWORD.
Preview	Displays the DDL that will execute the object action. For details, see Preview.	

5. Click **Open Key** to dismiss the dialog and return to the Symmetric Key wizard or editor.

Synonyms 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 a creation wizard for a synonym. For details, see Opening 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 Synonyms (SQL Server) - Properties.
   - **Permissions** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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*).

Tables 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.

Note: Before working with tables, consult Microsoft SQL Server documentation for general information on filegroups as well as specifics on CREATE TABLE... FILESTREAM and FILETABLE arguments. For more information, see Accessing Third Party Documentation.

To create a new table using a wizard:

1. Open a creation wizard for a table. For details, see Opening an Object Wizard.
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 - Setting Permissions or Privileges for an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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
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 Schemes 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.
Filestream On (Version 2008^)	This lets you specify the filegroup for FileStream data, specifying a FILESTREAM_ON argument to the generated DDL. This control is only available if a filegroup that can contain filestreams is available in the database. For more on filegroups, see Databases Wizard (SQL Server).
As Filetable and other **Filetable...** properties (Version 2012^)	**As Filetable** lets you add an AS FILETABLE argument to the generated DDL, specifying that the table is created as a FileTable. The **Columns**, **Indexes**, and **Constraints** tabs/panels are disabled when this control is selected. **Filetable Collate Filename** lets you provide a FILETABLE_COLLATE_FILENAME argument value, specifying the name of the collation used. If no value is provided, the default collation for the database is used. **Filetable Directory Name** lets you provide a FILETABLE_DIRECTORY argument value, specifying a FileTable directory name. If no value is provided, the name of the filetable is used. **Filetable PK Constraint Name** lets you provide a FILETABLE_PRIMARY_KEY_CONSTRAINT_NAME argument value, specifying the name used for the primary key constraint automatically created on the FileTable. If this value is not provided, a system-generated name is used. **Filetable StreamID Unique Constraint Name** lets you provide a FILETABLE_STREAMID_UNIQUE_CONSTRAINT_NAME argument value, specifying the name used for the unique key constraint automatically created on the FileTable. If this value is not provided, a system-generated name is used. **Filetable Fullpath Unique Coonstraint Name** lets you provide a FILETABLE_FULLPATH_UNIQUE_CONSTRAINT_NAME argument value, specifying the name used for the unique constraint automatically created on the parent_path_locator and name columns in the FileTable. If this value is not provided, a system-generated name is used.

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: 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.

Note: This tab/panel is not available if **As Filetable** is selected. For details, see Tables (SQL Server) - Properties.

Tables (SQL Server) - Indexes

Click **Add** to open the Index Wizard. For details, see Indexes Wizard (SQL Server).

Note: This tab/panel is not available if **As Filetable** is selected. For details, see Tables (SQL Server) - Properties.

Tables (SQL Server) - Constraints

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- Primary Keys Wizard (SQL Server)
- Unique Keys Wizard (SQL Server)
- Foreign Keys Wizard (SQL Server)
- Create Synonym

Note: This tab/panel is not available if **As Filetable** is selected. For details, see Tables (SQL Server) - Properties.

Triggers 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 a creation wizard for a trigger. For details, see Opening 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 Triggers (SQL Server) - Properties.
   - **Definition** panel - for details, see Triggers (SQL Server) - Definition.
3. Finally, use the **Execute** button to create the 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.

Triggers (SQL Server) - Definition

Complete the CREATE TRIGGER outline provided by typing or pasting the body of the trigger.

Unique Keys 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 a creation wizard for a unique key. For details, see Opening 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 - Unique Keys (SQL Server) - Properties.
   - **Columns** panel - Unique Keys (SQL Server) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 Messages 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 Opening 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 - Setting Permissions or Privileges for an Object
    - **DDL View** panel - for details, see Previewing the DDL Generated to 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.

Users 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 a creation wizard for a user. For details, see Opening 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 - Users (SQL Server) - Properties.
   - **Roles** panel - Users (SQL Server) - Roles.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 Datatypes 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 a creation wizard for a user datatype. For details, see Opening 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 - User Datatypes (SQL Server) - Properties.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Views 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 a creation wizard for a view. For details, see Opening 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 Views (SQL Server) - Properties.
    - **Definition** panel - for details, see Views (SQL Server) - Definition.
3. Finally, use the **Execute** button to create the 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

You create MySQL objects using the following wizards:

- Databases wizard (MySQL)
- Foreign Keys wizard (MySQL)
- Functions wizard (MySQL)
- Indexes, Primary Keys, or Unique Keys wizard (MySQL)
- Tables wizard (MySQL)
- Users wizard (MySQL)

Databases 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 a creation wizard for a database. For details, see Opening 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 Databases - Properties
   - **Privileges** panel - Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 a creation wizard for a foreign key. For details, see Opening 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 Foreign Keys - Properties
   - **Column Mapping** panel - for details, see Foreign Keys - Column Mapping
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 a creation wizard for a function. For details, see Opening 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 Functions - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 a creation wizard for an index, primary key, or unique key. For details, see Opening 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 Indexes, Primary Keys, or Unique Keys - Properties
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object. For more information, see Previewing the DDL Generated to 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 a creation wizard for a table. For details, see Opening 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 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 Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object
3. Finally, use the **Execute** button to create the 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 a creation wizard for a user. For details, see Opening an Object Wizard.
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 Setting Permissions or Privileges for an Object
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object
3. Finally, use the **Execute** button to create the 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. The discrepancy cannot be enforced, so we urge you to show a little restraint. This is the name that is displayed in the User branch of the Navigator tree. If you leave this field blank and complete the process, you create an <anonymous> user. Only one anonymous user per datasource is allowed.
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

You create Oracle objects using the following wizards:

- Clusters Wizard (Oracle)
- Database Links Wizard (Oracle)
- Directories Wizard (Oracle)
- Foreign Keys Wizard (Oracle)
- Functions Wizard (Oracle)
- Indexes Wizard (Oracle)
- Oracle Job Queue Wizard (Oracle)
- Jobs wizard (Oracle)
- Libraries Wizard (Oracle)
- Materialized Views Wizard (Oracle)
- Materialized View Logs Wizard (Oracle)
- Outlines Wizard (Oracle)
- Packages Wizard (Oracle)
- Primary Keys Wizard (Oracle)
- Procedures Wizard (Oracle)
- Profiles Wizard (Oracle)
- Programs wizard (Oracle)
- Roles Wizard (Oracle)
- Rollback Segments Wizard (Oracle)
- Schedules Wizard (Oracle)
- Sequences Wizard (Oracle)
- Snapshots and Snapshot Logs Wizards (Oracle)
- Synonyms Wizard (Oracle)
- Tables Wizard (Oracle)
- Tablespaces Wizard (Oracle)
- Triggers Wizard (Oracle)
- Object Types Wizard (Oracle)

- Unique Keys Wizard (Oracle)
- Users Wizard (Oracle)
- Views Wizard (Oracle)

Clusters 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.

To Open the Cluster Wizard

1. On the Navigator, 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 Adding or modifying a cluster column. **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.

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'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 lets you manage cluster columns. You can open the dialog in the Oracle Cluster Wizard.

The table below describes the options and functionality on the Add or Modify Cluster Column dialog

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, do the following:

1. In the **Add Cluster Column** dialog, in the **ColumnName** 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**.

The **Add Cluster Column** dialog closes.

For more information, see Adding or modifying a cluster column.

Database Links 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 Navigator, 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
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.

Directories 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 Directories Wizard

1. On the Navigator, 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 Keys 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 a creation wizard for a foreign key. For details, see Opening 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 - Foreign Keys (Oracle) - Properties.
   - **Column Mapping** panel - Foreign Keys (Oracle) - Column Mapping.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Functions 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 a creation wizard for a function. For details, see Opening 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 - Functions (Oracle) - Properties.
    - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Indexes 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: 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. The remaining non-key columns are stored in the index structure.

To create a new index using a wizard:

1. Open a creation wizard for an index. For details, see Opening 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 - Indexes (Oracle) - Properties.
   - **Columns** panel - Indexes (Oracle) - Columns.
   - **Storage** panel - Indexes (Oracle) - Storage.
   - **Partition** panel - Indexes (Oracle) - Partition.
   - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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.

Oracle Job Queue Wizard (Oracle)

This wizard lets you build and submit a DBMS_JOB.SUBMIT call, creating a once-only or scheduled job, optionally submitted as broken.

To create a new job queue using a wizard:

1. Open a creation wizard for a job queue. For details, see Opening 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 Job Queues (Oracle) - Properties.
   - **Definition** panel - for details, see Job Queues (Oracle) - Definition.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object. For more information, see Previewing the DDL Generated to Create the New Object.

Job Queues (Oracle) - Properties

When creating or editing a job, this tab/panels lets you work with the following settings:

Option	Availability and Description
No Parse	Specifies the value of the NO_PARSE flag. If deselected, the associated procedure is parsed on submission. If selected, the associated procedure is parsed the first time the job is executed.
Submit as disabled	If selected, the DBMS_JOB.SUBMIT call generated to create the job, is followed by a DBMS_JOB.BROKEN call. Marking the job as broken, this disables the job and in order to execute the job, you must use the **Run Job** object action. For details, see Run Job (job queues).
Next Date	Use this control to open a calendar that lets you select a date for the job to run with a default time of midnight. You can then modify the time value.
Run Again	
Every **Time Unit**	Only available if **Run Again** is selected and **Use Custom Expression** is not selected..
Use Custom Expression and **Custom Expression**	Only available if **Run Again** is selected. **Custom Expression** becomes available when **Use Custom Expression** is selected.

Job Queues (Oracle) - Definition

When creating or editing a job, the tab/panel lets you provide the DBMS_JOB.SUBMIT call's WHAT parameter, specifying the name of the PL/SQL procedure to run.

Jobs wizard (Oracle)

This wizard builds and submits a call to DBMS_SCHEDULER.CREATE_JOB with optional DBMS_SCHEDULER.SET_JOB_ARGUMENT_VALUE, DBMS_SCHEDULER.SET_ATTRIBUTE, and DBMS_SCHEDULER.ENABLE calls. This lets you create a fully-specified job.

Note: This functionality is available as of Oracle 10g.

Note: Before working with jobs, consult Oracle documentation for general information on DBMS_SCHEDULER. For more information, see Accessing Third Party Documentation.

To create a job using a wizard

1. In Rapid SQL, on a connected Oracle datasource, expand the **Scheduler** node, right-click the **Jobs** node and select **Create** from the context menu.

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 Jobs (Oracle) - Properties.
   - **Action** panel - for details, see Jobs (Oracle) - Action.
   - **Arguments** panel - for details, see Jobs (Oracle) - Arguments.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.

3. Finally, use the **Finish** button to create the object.

Jobs (Oracle) - Properties

When creating or editing a job, this tab/panel lets you work with the following settings:

Group	Settings and Description
Creation	Lets you provide an **Owner** and **Name** for the job.
Attributes	Lets you provide **Enabled**, **Description** (COMMENTS), and **Job Class** (JOB_CLASS) attributes.
Schedule	Lets you select a **Schedule Type** of IMMEDIATE, ONCE, REPEATING, or specified SCHEDULE. Depending on your selection, the following settings may be available: **Repeat Interval** - corresponds to a REPEAT_INTERVAL attribute. Calendaring syntax, as described in Oracle DBMS_SCHEDULER documentation, can be typed free-form, and will be passed directly to the server. For information on third-party documentation access, see Accessing Third Party Documentation. **Start Date** and **End Date** - correspond to START_DATE and END_DATE, attributes. **Schedule Owner** and **Schedule Name** - correspond to SCHEDULE_NAME attribute components.
Properties	The **Auto Drop** setting corresponds to a CREATE_JOB AUTO_DROP attribute. Other settings result in one or more DBMS_SCHEDULER.SET_ATTRIBUTE calls being generated with RESTARTABLE, JOB_PRIORITY, MAX_RUNS, MAX_FAILURES, FAILED RUNS, STOP_ON_WINDOW_CLOSE, INSTANCE_STICKINESS, or JOB_WEIGHT attributes specified, as appropriate.

Jobs (Oracle) - Action

When creating or editing a job, this tab/panel lets you specify a JOB_ACTION attribute for the DBMS_SCHEDULER.CREATE_JOB call, specifying the job action for the job.

To specify a job action

1. From **Job Type**, select one of PLSQL BLOCK, STORED PROCEDURE, EXECUTEABLE, or CHAIN.
2. Based on your **Job Type** selection, provide additional details as follows:
   - PLSQL BLOCK - Type or paste the SQL to be executed.
   - STORED PROCEDURE - Provide the **Procedure Owner** and **Procedure Name**.
   - EXECUTEABLE - Use the browse button to locate the executable.
   - PROGRAM - Provide the **Program Owner** and **Program Name**.
   - CHAIN - Provide the **Chain Owner** and **Chain Name**.

Jobs (Oracle) - Arguments

When creating or editing a job, if you selected an **Action** tab choice of PROGRAM, EXECUTABLE, or STORED PROCEDURE, this tab/panel lets you manage parameters to be passed.

Note: If you provide arguments with an **Action** tab choice of PROGRAM, the provided arguments override those specified in the program definition. For details, see Programs wizard (Oracle).

Libraries Wizard (Oracle)

Libraries are an object type 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 Navigator, 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 Views 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 Navigator, 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.

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.

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 may open.
Specify number of subpartions	Self-explanatory Click **Add**, **Insert**, or **Edit** to open a dialog 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
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
Direct path data load	X	X	X
ON COMMIT	X		X
ON DEMAND	X	X	X

Materialized View Logs 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 Logs Wizard

1. On the Navigator, 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
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.

Outlines 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 a creation wizard for an outline. For details, see Opening 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 - Outlines (Oracle) - Properties.
   - **Definition** panel - Outlines (Oracle) - Definition.
   - **DDL** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Packages 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 Navigator, 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 Keys 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 a creation wizard for a primary key. For details, see Opening 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 - 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 Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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 cheeked 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.
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.

Procedures 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 a creation wizard for a procedure. For details, see Opening 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 - Procedures (Oracle) - Properties.
   - **Definition** panel - for details, see Procedures (Oracle) - Definition.
3. Finally, use the **Execute** button to create the 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.

Profiles 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 Navigator, 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)
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

Programs wizard (Oracle)

This wizard builds and submits a call to DBMS_SCHEDULER.CREATE_PROGRAM with optional DBMS_SCHEDULER.DEFINE_PROGRAM_ARGUMENT calls, letting you create a program.

Note: This functionality is available as of Oracle 10g.

Note: Before working with jobs, consult Oracle documentation for general information on DBMS_SCHEDULER. For more information, see Accessing Third Party Documentation.

To create a program using a wizard

1. In Rapid SQL, on a connected Oracle datasource, expand the **Scheduler** node, right-click the **Programs** node and select **Create** from the context menu.

2. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

   o **Action** panel - for details, see Programs (Oracle) - Action.

   o **Arguments** panel - for details, see Programs (Oracle) - Arguments.

   o **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.

3. Finally, use the **Finish** button to create the object.

Programs (Oracle) - Action

When creating or editing a job, this tab/panel lets you work with the following settings:

Group	Settings and Description
Creation	Lets you provide an **Owner** and **Name** for the job as well as ENABLED and COMMENTS (**Description**) attribute values
Type of Program	The **Type of Program** setting lets you choose among EXECUTABLE, STORED PROCEDURE, or PLSQL Block. Depending on your choice, take one of the following actions: PLSQL BLOCK - Type or paste the SQL to be executed. STORED PROCEDURE - Provide the **Procedure Owner** and **Procedure Name**. EXECUTEABLE - Use the browse button to locate the executable.

Programs (Oracle) - Arguments

When creating or editing a program, if you selected an Action tab choice of EXECUTABLE or STORED PROCEDURE, this tab/panel lets you manage parameters to be passed.

Roles Wizard (Oracle)

The Role Wizard 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 Navigator, 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 Role Activation.

Rollback Segments 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 Navigator, 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.
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

Schedules Wizard (Oracle)

This wizard builds and submits a call to DBMS_SCHEDULER.CREATE_SCHEDULE call, letting you create a schedule.

Note: This functionality is available as of Oracle 10g.

Note: Before working with jobs, consult Oracle documentation for general information on DBMS_SCHEDULER. For more information, see Accessing Third Party Documentation.

To create a schedule using a wizard

1. In Rapid SQL, on a connected Oracle datasource, expand the **Scheduler** node, right-click the **Schedules** node and select **Create** from the context menu.

2. Use the following topics as a guide to setting properties and performing tasks as you pass through the wizard panels:

    o **Properties** panel - for details, see Schedules (Oracle) - Properties.

    o **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.

3. Finally, use the **Finish** button to create the object.

Schedules (Oracle) - Properties

When creating or editing a job, this tab/panel lets you work with the following settings:

Group	Settings and Description
Creation	Lets you provide an **Owner** and **Name** for the schedule.
Attributes	**Description** - corresponds to a COMMENT attribute. **Repeat Interval** - corresponds to a REPEAT_INTERVAL attribute. Calendaring syntax, as described in Oracle DBMS_SCHEDULER documentation, can be typed free-form, and will be passed directly to the server. For information on third-party documentation access, see Accessing Third Party Documentation. **Start Date** and **End Date** - correspond to START_DATE and END_DATE, attributes.

Sequences 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 Navigator, 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.
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

Snapshots and Snapshot Logs Wizards (Oracle)

Oracle has replaced the snapshot functionality with materialized views. Refer to the Materialized Views Wizard (Oracle) and Materialized View Logs Wizard (Oracle).

Synonyms 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 a creation wizard for a synonym. For details, see Opening 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 - Synonyms (Oracle) - Properties.
   - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Tables Wizard (Oracle)

Tables are the most basic data storage units for Oracle. As you might expect, data is stored in rows and columns. The Table Wizard 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 Clusters 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 a creation wizard for a table. For details, see Opening 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 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 Adding a Comment to an Object.
- **Permissions** panel - for details, see Setting Permissions or Privileges for an Object.
- **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.

3. Finally, use the **Execute** button to create the 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. For details, see Indexes 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 Keys Wizard (DB2 LUW)
- Unique Keys Wizard (DB2 LUW)
- Foreign Keys Wizard (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.
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 Oracle Partitioning.

Click **Create Partition** to Partition a table.

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.
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	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.
List Definitions (only available with a **Partition Type** of **List**)	Click **New** to Add a partition definition.

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.	

Tablespaces 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.

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 a creation wizard for a tablespace. For details, see Opening 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 - see Tablespaces (Oracle) - Properties.
   - **Datafiles** panel - see Tablespaces (Oracle) - Datafiles.
   - **DDL** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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.

Triggers Wizard (Oracle)

This wizard lets you build and submit a CREATE OR REPLACE TRIGGER statement, creating a trigger to be associated with a table, view, schema, or database. BEFORE, AFTER, and COMPOUND triggers are supported for a variety of DDL, DML, and Database manipulation events.

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 a creation wizard for a trigger. For details, see Opening 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 Triggers (Oracle) - Properties.
   - **Events** panel - for details, see Triggers (Oracle) - Events.
   - **Column Selection** panel - for details, see Triggers (Oracle) - Column Selection.
   - **Action** panel - for details, see Triggers (Oracle) - Action.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Triggers (Oracle) - Properties

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Availability and description	
Parent Type	Select the type (TABLE, DATABASE, SCHEMA, or VIEW) of element that this trigger is to be associated with.	
Parent Schema, and Parent Name	If available and appropriate, select the specific object for which the trigger is being created.	
Schema and Name	Select a schema and provide a name for the trigger being created.	
Enabled	Deselecting this check box adds a DISABLE clause to the CREATE OR REPLACE TRIGGER statement, disabling 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. **COMPOUND** (11.1.0.7 ^): These triggers fire at more than one timing point.	
Trigger Type	Not available with a **Parent Type** of **Database** or **Schema** and not available with a **Trigger Timing** value of **COMPOUND**	Selecting **ROW** adds a FOR EACH ROW clause to the CREATE OR REPLACE TRIGGER statement, making this a row trigger that fires for each affected row. Selecting **STATEMENT** designates this as a statement trigger that only fires once.
Old Table Alias	Not available with a **Parent Type** of **Database** or **Schema**	Type the name of a temporary table of rows as they exist before they are updated or deleted.
New Table Alias		Type a name for a temporary table of rows as they exist after they are inserted or updated.
When Clause		Type a WHEN clause or open a WHEN CLAUSE wizard to further qualify the trigger behavior.

Triggers (Oracle) - Events

When creating or editing a trigger, this tab/panel lets you provide BEFORE, AFTER, or FOR clause options.

If you selected a **Parent Type** of **Table** or **View**, select one or more of the INSERT, UPDATE, or DELETE check boxes to specify the events that are to fire the trigger.

If you selected a **Parent Type** of **Database** or **Schema**, select one or more of the ALTER, CREATE, ANALYZE, ASSOCIATE_STATISTICS, AUDIT, COMMENT, DISASSOCIATE_STATISTICS, DROP, GRANT, NOAUDIT, RENAME, REVOKE, TRUNCATE, LOGON, SERVERERROR, or SUSPEND check boxes to specify the events that are to fire the trigger.

Triggers (Oracle) - Column Selection

If you chose UPDATE as the **Trigger Event**, this tab/panel lets you provide FOR UPDATE OF, BEFORE UPDATE OF, or AFTER UPDATE OF clause options. Select the check box beside each column that is to fire the trigger.

Triggers (Oracle) - Action

If you selected a **Trigger Timing** value of BEFORE or AFTER, provide the complete body of trigger. Keep the following in mind:

- 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.

If you selected a **Trigger Timing** value of COMPOUND, a compound trigger block template is generated. Customize the generated code provide a valid trigger body.

Object Types 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 Navigator, 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 Keys 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 a creation wizard for a unique key. For details, see Opening 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 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 Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

Users 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 a creation wizard for a user. For details, see Opening 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 Users (Oracle) - Properties.
   - **Roles** panel - for details, see Users (Oracle) - Roles.
   - **Quotas** panel - for details, see Users (Oracle) - Quotas.
   - **DDL** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.
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.

Views 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 a creation wizard for a view. For details, see Opening 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 Views (Oracle) - Properties.
   - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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.

PostgreSQL Object Wizards

You create PostgreSQL objects using the following wizards:

- Check Constraints Wizard (PostgreSQL)
- Domains Wizard (PostgreSQL)
- Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL)
- Foreign Keys Wizard (PostgreSQL)
- Functions Wizard (PostgreSQL)
- Indexes Wizard (PostgreSQL)
- Roles Wizard (PostgreSQL)
- Rules Wizard (PostgreSQL)
- Schemas Wizard (PostgreSQL)
- Tables Wizard (PostgreSQL)
- Tablespaces Wizard (PostgreSQL)
- Triggers Wizard (PostgreSQL)
- Types Wizard (PostgreSQL)
- Views Wizard (PostgreSQL)

Check Constraints Wizard (PostgreSQL)

This wizard lets you build and submit an ALTER TABLE... ADD CONSTRAINT... CHECK... statement, adding a constraint to an existing table.

To create a new check constraint using a wizard:

1. Open a creation wizard for a check constraint. For details, see Opening 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 - Check Constraints (PostgreSQL) - Properties.
   - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Check Constraints (PostgreSQL) - Properties

When creating or editing a check constraint, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Identify the table for which the constraint is being created.
Name	Lets you type the name of the constraint.
No Inherit	If selected, a NO INHERIT parameter is added to the generated ALTER TABLE statement, letting you create a non-inherited check constraint.
Is Validated	If not selected, a NOT VALID parameter is added to the generated ALTER TABLE statement, letting you ensure that initial check to verify that all rows satisfy the constraint is not executed.
Check Condition	Type the check constraint condition. As a convenience, use the **Table Columns** button to paste in column names as part of the condition.

Domains Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE DOMAIN... AS... statement, adding a new, datatype-based domain.

Note: Before working with this object type, consult PostgreSQL documentation for details on topics such as NULL and default values. For more information, see Accessing Third Party Documentation.

To create a new domain using a wizard:

1. Open a creation wizard for a domain. For details, see Opening 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 - Domains (PostgreSQL) - Properties.
   - **Constraints** panel - Domains (PostgreSQL) - Constraints.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Domains (PostgreSQL) - Properties

When creating or editing a domain, this tab/panel lets you work with the following settings:

Setting	Description
Owner and **Schema**	Lets you select the schema and owner of the new domain.
Datatype	Lets you provide a name for the new domain.
Type	Lets you select the underlying datatype of the new domain.
Allow Nulls	If unselected, a NOT NULL parameter is included with the generated DDL, generally assuring that the domain cannot take a NULL value.
Default	Lets you enter a value or expression that defines the default value for columns of this domain type.
Collation Schema and **Collation Name**	Let you provide the value of a COLLATE parameter, specifying an optional collation for the domain.

Domains (PostgreSQL) - Constraints

When creating or editing domains, this tab/panel lets you manage foreign key, primary key, unique key, exclusion constraints, and check constraints for the domain:

Click **Add** to open the Check Constraints Wizard. Use the wizard to create a new constraint for the domain. See Check Constraints Wizard (PostgreSQL)

Select an existing constraint and click **Edit** to open a wizard that lets you add modify that constraint. See Check Constraints Editor (PostgreSQL)

Select a constraint and click **Drop** to drop a constraint from the table.

Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL)

The wizards for creating these object types let you build and submit an ALTER TABLE... ADD CONSTRAINT... statements specific to the object type.

Note: Before working with these object types, consult PostgreSQL documentation. For more information, see Accessing Third Party Documentation.

To create a new exclusion constraint, primary key, or unique key using a wizard

1. Open a creation wizard for the target object type. For details, see Opening 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 - Exclusion Constraints, Primary Keys and Unique Keys (PostgreSQL) - Properties.
    - **Constraints** panel - Exclusion Constraints, Primary Keys, and Unique Keys (PostgreSQL) - Columns.
    - **Comment** panel - for details, see Adding a Comment to an Object.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Exclusion Constraints, Primary Keys and Unique Keys (PostgreSQL) - Properties

When creating or editing an exclusion constraint, primary key, or unique key, this tab/panel lets you work with the following settings:

Setting	Availability and Description
Table Schema and **Table Name**	Select the schema and name of the parent table.
Name	Provide a name for the object being created.
Index Name	Primary Keys and Unique Keys only. Lets you provide the name of an existing index that will be used as the base for the primary or unique key.
Deferrable	If selected, a DEFERRABLE parameter is included in the generated DDL, specifying that checking of constraints can be postponed until the end of the transaction.
Deferred	If IMMEDIATE is not selected, an INITIALLY DEFERRED parameter is added to the generated DDL, ensuring that the constraint is checked at the end of the transaction.
Predicate	Exclusion Constraints only. Lets you provide an expression used in a WHERE clause that defines the constraint on a subset of the table.
Access Name	Exclusion Constraints only. Lets you select an index method (**btree**, **gist**, **hash**, or **spgist**) for the USING option of the EXCLUDE clause.
Tablespace	Lets you optionally select the tablespace on which the new or altered object is to be created. If not specified, the default tablespace will be used.
Fill Factor	Lets you provide a FILLFACTOR value, specifying the percentage of tables that can be packed.
Buffering	Excusion constraints only. Only available with an **Access Name** of **gist**. Lets you select a BUFFERING value (**AUTO**, **ON**, or **OFF**).

Exclusion Constraints, Primary Keys, and Unique Keys (PostgreSQL) - Columns

When creating or editing an exclusion constraint, this tab/panel lets you set up the operators on the columns or expressions that define the exclusion constraint.

Foreign Keys Wizard (PostgreSQL)

This wizard lets you build and submit an ALTER TABLE... ADD CONSTRAINT .. statement, adding a new, foreign key to an existing table.

To create a new foreign key using a wizard:

1. Open a creation wizard for a foreign Key. For details, see Opening 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 - Foreign Keys (PostgreSQL) - Properties.
   - **Column Mapping** panel - Foreign Keys (PostgreSQL) - Column Mapping.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Foreign Keys (PostgreSQL) - Properties

When creating or editing a domain, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Let you select the table in which the foreign key is being defined.
Name	Lets you provide the ADD CONSTRAINT parameter value, naming the new foreign key.
Validate	If not selected, a NOT VALID parameter is included in the generated DDL, specifying that the initial check to verify that all rows satisfy the constraint is not executed.
Deferrable	If selected, a DEFERRABLE parameter is included in the generated DDL, specifying that checking of constraints can be postponed until the end of the transaction
Deferred	If IMMEDIATE is not selected, an INITIALLY DEFERRED parameter is added to the generated DDL, ensuring that the constraint is checked at the end of the transaction.
Delete Rule and **Update Rule**	Choices correspond to ON DELETE... and ON UPDATE parameter variations added to the generated DDL: **CACHE**, **SET DEFAULT** , **SET NULL** , and **RESTRICT**.
Match Full	If selected, a MATCH FULL parameter is added to the generated DDL, ensuring that no column of a multicolumn foreign key can be null unless all foreign key columns are null.

Foreign Keys (PostgreSQL) - Column Mapping

When creating or editing a foreign key, this tab/panel lets you specify the REFERENCES and FOREIGN KEY parameters of the generated ALTER TABLE... DDL.

1. Under **Main Table**, select the check boxes corresponding to the columns in the current table that are to use values in the referenced table.
2. Under **Referenced Table**, choose the **Owner** and then the **Table** name of the referenced table and then select the check boxes corresponding to the columns in the referenced table whose values are to be used in the main table.

Functions Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE OR REPLACE FUNCTION statement, creating a new function or replacing an existing function.

Note: Before working with this object action, consult PostreSQL documentation for details on CREATE FUNCTION parameter values. For more information, see Accessing Third Party Documentation.

To create a new function using a wizard:

1. Open a creation wizard for a function. For details, see Opening 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 - Functions (PostgreSQL) - Properties.
   - **Parameters** panel - Functions (PostgreSQL) - Parameters.
   - **Body** panel - Functions (PostgreSQL) - Body.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Functions (PostgreSQL) - Properties

When creating or editing a domain, this tab/panel lets you work with the following settings:

Setting	Description
Schema and **Name**	Let you identify the owning schema and provide a name for the function.
Return Type	Lets you provide a RETURNS clause value, specifying a valid type (base, composite, domain type, or a reference to the type of a table column) for the value returned by the function.
Language	Lets you provide a LANGUAGE clause value, specifying a valid language implementation.
Object File and **Link Symbol**	Let you specify the AS clause appropriate to dynamically loadable C language functions for which the function name in the C language source code is identical to the name of the SQL function. **Object File** is the name of the file containing the dynamically loadable object. **Link Symbol** is the function name in the C language source code.
Is Window	If selected, specifies that this C language function is a window function.
Leakproof	If selected, a LEAKPROOF clause is added to the generated DDL, indicating that the function provides no information about its arguments other than by its return value.
Execution Cost	Lets you provide a COST clause value, estimating the execution cost in CPU operator cost units.
Result Rows	Lets you provide a ROWS clause value, estimating the number of rows the function is expected to return.
Behavior	Lets you select an IMMUTABLE, STABLE, or VOLATILE clause to the generated DDL, indicating the functions behavior with respect to database lookups and ability to make changes to the database.
Strict	Lets you add a STRICT clause to the generated DDL, specifying whether non-NULL values can be returned if any provided input parameters are NULL.
Is Security Definer	Lets you add a SECURITY DEFINED clause to the generated DDL, specifying that the function is to execute with the same privileges as the user who created it.

Functions (PostgreSQL) - Parameters

When creating or editing a function, this tab/panel lets you build and maintain a list of input/output parameters for the function. For each input or output parameter for this function, use the New button to add a new parameter and then provide a name for the parameter.

To edit a parameter in the attributes list, first select the parameter in the list. Then, in the **Attributes** area, select a **Datatype**, select a **Parameter Mode** (INPUT, OUTPUT, INPUT_OUTPUT, or VARIADIC), and if you chose INPUT, you can optionally provide a **Default** value.

To delete a parameter from the list, select the parameter and click the Delete button.

To reposition a parameter in the list, select the parameter and use the arrow buttons to move the parameter up or down in the list.

Functions (PostgreSQL) - Body

When creating or editing a function, this tab/panel lets you build and maintain a valid AS clause value that defines the function. You can use an internal function name, the path to an object file, a SQL query, or text in a procedural language.

Indexes Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE INDEX or CREATE UNIQUE INDEX statement, creating a new index on a table.

To create a new index using a wizard:

1. Open a creation wizard for an index. For details, see Opening 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 - Indexes (PostgreSQL) - Properties.
    - **Columns** panel - Indexes (PostgreSQL) - Columns.
    - **Comment** panel - for details, see Adding a Comment to an Object.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Indexes (PostgreSQL) - Properties

When creating or editing a domain, this tab/panel lets you work with the following settings:

Setting	Availability and Description
Parent Type	Lets you specify whether the index is being defined for a table or view.
Table Schema and Table Name	Select the schema and name of the parent table.
Name	Provide a name for the index being created.
Unique	If selected, CREATE UNIQUE INDEX DDL is generated, enforcing uniqueness. Otherwise simple CREATE INDEX DDL i generated.
Concurrently	If selected, s CONCURRENTLY parameter is added to the generated DDL, specifying that the index is built without taking any locks that would prevent concurrent inserts, updates, or deletes on the target table.
Clustered	If selected, an ALTER TABLE... CLUSTER ON... statement is generated following the DDL to generate the index, specifying this index as the default for subsequent cluster operations.
Tablespace	Lets you optionally select the tablespace on which the index is to be created. If not specified, the default tablespace will be used.
Index Method	Lets you specify a USING option value of **btree**, **hash**, **gist**, **gin**, or **spgist**.
Fill Factor	Lets you add a FILLFACTER= option value, specifying the percentage that the index method will try to fill index pages.
Fast Update	Lets you add a FASTUPDATE= ON option, enabling fast update.
Buffering	Available with an **Index Method** of **gist**. Determines whether the buffering build technique is set to ON, OFF, or AUTO.
IndexPredicate	Lets you provide a WHERE condition that will be used as a constraint expression for a partial index.

Indexes (PostgreSQL) - Columns

From the Column dropdown, select a column for the index and specify **Sort**, **Collation**, and **Operator Class** 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.

Roles Wizard (PostgreSQL)

This wizard lets you build and submit a simple CREATE ROLE statement, defining a new database role.

To create a new role using a wizard:

1. Open a creation wizard for a role. For details, see Opening 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 - Roles (PostgreSQL) - Properties.
   - **Membership** panel - Roles (PostgreSQL) - Membership.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Roles (PostgreSQL) - Properties

When creating or editing a role, this tab/panel lets you work with the following settings:

Setting	Availability and Description
Name	Lets you provide the name of the new role.
Password and **Encrypted Password**	These items control how the password is created for the new role. The **Password** control lets you provide a password for the new role. If **Encrypted Password** is selected, an ENCRYPTED PASSWORD... parameter is generated. Otherwise, an UNENCRYPTED PASSWORD... parameter is generated.
Account Expires	Selecting this control enables a calendar widget that lets you select a value for a VALID UNTIL parameter in the generated DDL.
Connection Limit	Lets you provide the value for a CONNECTION LIMIT parameter, setting a limit on the number of concurrent connections the new role can make.
Login	This control determines whether a role is allowed to log in. If **Login** is selected, a LOGIN parameter is included in the DDL. Otherwise, a NOLOGIN parameter is included in the DDL.
Superuser	This control determines whether a role can override all access restrictions within the database. If **Superuser** is selected, a SUPERUSER parameter is included in the DDL. Otherwise, a NOSUPERUSER parameter is included in the DDL.
Create Database and **Create Roles**	These controls determine whether a role can create databases or roles, respectively. When selected, CREATEDB or CREATEROLE parameters are added to the generated DDL. Otherwise, NOCREATEDB or NOCREATEDB parameter are included in the DDL.
Update Catalog	Only available if **Superuser** is selected. By default, this option is selected, dictating that this role can update system catalogs directly. If deslected, an UPDATE pg_catalog.pg_authid SET rolcatupdate=false WHERE rolname=... statement is generated and appended to the CREATE ROLE statement, explicitly disabling catalog update for the new role.
Replication	This setting determine whether a role can initiate streaming replication or can control backup mode. If **Replication** is selected, a REPLICATION parameter is included in the DDL. Otherwise, a NOREPLICATION parameter is included in the DDL.

Roles (PostgreSQL) - Membership

When creating or editing a role, this tab/panel lets you append one or more GRANT statements to the generated CREATE ROLE statement, granting membership in the role to other roles. For each GRANT statement to be generated, select a role from the **Groups** list and click **Join** to assign membership in the selected role to the role being created.

Rules Wizard (PostgreSQL)

This wizard lets you build and submit a simple CREATE OR REPLACE RULE statement, defining a new database rule.

To create a new rule using a wizard:

1. Open a creation wizard for a rule. For details, see Opening 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 - Rules (PostgreSQL) - Properties.
   - **Definition** panel - Rules (PostgreSQL) - Definition.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Rules (PostgreSQL) - Properties

When creating or editing a rule, this tab/panel lets you work with the following settings:

Setting	Availability and Description
Name	Lets you provide the name of the new role.
Parent Type	Select the type (TABLE or VIEW) of element to which this rule is to be applied.
Parent Schema and **Parent Name**	Select the schema and name of the parent object.
Connection Limit	Lets you provide the value for a CONNECTION LIMIT parameter, setting a limit on the number of concurrent connections the new role can make.
Name	Provide a name for the rule being created.
IsInstead	If unselected, a simple DO clause will be generated for the rule. If selected, a DO INSTEAD clause will be generated. The DO or DO INSTEAD clause command or commands are provided on the **Definition** tab/panel.
Event	Lets you select an AS ON event of SELECT, INSERT, UPDATE, or DELETE.

Rules (PostgreSQL) - Definition

When creating or editing a rule, this tab/panel lets you provide the DO or DO INSTEAD clause value, specifying the additional or alternative commands for this rule.

Schemas Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE SCHEMA... AUTHORIZATION... statement, creating a new schema.

To create a new schema using a wizard:

1. Open a creation wizard for a schema. For details, see Opening 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 Schemas (PostgreSQL) - Properties.
   - **Permissions** panel - for details, see Setting Permissions or Privileges for an Object
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Schemas (PostgreSQL) - Properties

When creating or editing a check constraint, this tab/panel lets you work with the following settings:

Setting	Description
Name	Lets you type the name of the schema.
Owner	Lets you select the name of the user who will own the schema.

Tables Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE TABLE statement, creating a new table.

To create a new table using a wizard:

1. Open a creation wizard for a table. For details, see Opening 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 Tables (PostgreSQL) - Properties.
   - **Ancestors** panel - for details, see Tables (PostgreSQL) - Ancestors.
   - **Columns** panel - for details, see Tables (PostgreSQL) - Columns.
   - **Constraints** panel - for details, see Tables (PostgreSQL) - Constraints.
   - **Permissions** panel - for details, see Setting Permissions or Privileges for an Object
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Tables (PostgreSQL) - Properties

When creating or editing a table, this tab/panel lets you work with the following settings:

Setting	Description
Owner, **Schema**, and **Name**	Lets you provide ownership details and provide a name for the table.
Tablespace	Lets you select a CREATE TABLE... TABLESPACE parameter value, specifying the tablespace in which the new table will be created.
With OIDS	If selected, a WITH (OIDS=TRUE) parameter is added to the generated DDL, specifying that rows of the table will have object identifiers. Otherwise, a WITH (OIDS=FALSE) parameter is added to the generated DDL.
Persistence	Lets you change from the default form of CREATE TABLE statement by selecting either **TEMPORARY** (CREATE TEMPORARY TABLE) or **UNLOGGED** (CREATE UNLOGGED TABLE).
Of Type	Lets you select the value for an OF clause, designating this as a typed table. Any columns defined using the **Ancestors** or **Columns** tab are overwritten when a type is selected.
On Commit	Only available with a **Persistence** selection of **TEMPORARY**. Lets you change the ON COMMIT... parameter DDL generated between **DROP** (ON COMMIT DROP), **PRESERVE ROWS** (ON COMMIT PRESERVE ROWS), and **DELETE ROWS** (ON COMMIT DELETE ROWS).
Fill Factor	Lets you provide a FILLFACTOR parameter value, an integer expressing a percentage between 10 and 100, dictating the upper limit to which INSERTs can pack table pages.
Table Autovacuum... settings	Lets you enable or disable the autovacuum daemon as well as specify values for parameters used in autovacuum calculations (Threshold, Scale Factor, Analyze Threshold, Analyze Threshold, Cost Delay, Cost Limit, Freeze Min Age, Freeze Max Age, Freeze Table Age, Multixact Freeze Min Age, Multixact Freeze Max Age, and Multixact Freeze Table Age.
Toast Table Autovacuum... settings	Lets you enable or disable the autovacuum daemon for the table's secondary toast table, as well as specify values for parameters used in autovacuum calculations (Threshold, Scale Factor, Cost Delay, Cost Limit, Freeze Min Age, Freeze Max Age, Freeze Table Age, Multixact Freeze Min Age, Multixact Freeze Max Age, and Multixact Freeze Table Age.

Tables (PostgreSQL) - Ancestors

This tab/panel lets you add an INHERITS clause to the generated DDL, building a list of tables from which this table inherits columns. For each table to be added to the list, use the New button to enable a dropdown that lets you select a table. Use the Delete button to remove a selected table from the list.

Tables (PostgreSQL) - 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 (**Add Column**) to the bottom of the column list or to insert a new column (**Insert Column**) above the currently selected column in the column list.
2. Provide a Name for the column.
3. 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:

Setting	Description
Type	Lets you select the datatype for the column.
Allow Nulls	If not selected, a NOT NULL argument is added to the column specification.
Default	Lets you provide a default value for the column.
Comment	Lets you append a COMMENT ON COLUMN statement to the generated table creation DDL.
Statistics Target	Lets you append an ALTER TABLE... ALTER COLUMN... SET STATISTICS statement to the generated table creation DDL, setting the per-column statistics-gathering target for ANALYZE operations.
Storage Mode	Lets you append an ALTER TABLE... ALTER COLUMN... SET STORAGE statement to the generated table creation DDL, with a storage specifier of EXTERNAL, EXTENDED, or MAIN.
NDistinct Values and **NDistinct Values Inherited**	Let you append ALTER TABLE... ALTER COLUMN... SET (n_distinct or n_distinct_inherited) statements to the generated table creation DDL. These override the number-of-distinct-values estimates made by ANALYZE operations for the table or its inherited children.

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 reorder the column list:

- Move a selected column up or down the list using the Move Item Up or Move Item Down buttons.

Tables (PostgreSQL) - Constraints

When creating or editing tables, this tab/panel lets you manage foreign key, primary key, unique key, exclusion constraints, 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. 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:
  - Check Constraints Wizard (PostgreSQL) and Check Constraints Editor (PostgreSQL)
  - Foreign Keys Wizard (PostgreSQL) and Foreign Keys Editor (PostgreSQL)
  - Roles Wizard (PostgreSQL) and Roles Editor (PostgreSQL)
  - Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL) and Exclusion Constraints, Primary Keys, or Unique Keys Editors (PostgreSQL)
- Select a constraint and click **Drop** to drop a constraint from the table.

Tablespaces Wizard (PostgreSQL)

This wizard lets you build and submit a simple CREATE TABLESPACE statement, registering a new, cluster-wide tablespace.

To create a new tablespace using a wizard:

1. Open a creation wizard for a tablespace. For details, see Opening 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 - Check Constraints (PostgreSQL) - Properties.
   - **Definition** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Tablespaces (PostgreSQL) - Properties

When creating or editing a tablespace, this tab/panel lets you work with the following settings:

Setting	Description
Table Schema and **Table Name**	Identify the table for which the constraint is being created.
Name	Lets you type the name of the tablespace.
Owner	Lets you select the name of the user who will own the tablespace.
Location	Lets you provide a LOCATION parameter clause, identifying the directory used for the tablespace.
random_page_code and **seq_page_cost**	When either of these settings are assigned values, the generated CREATE TABLESPACE DDL is followed by an ALTER TABLESPACE statement that lets you override the RANDOM_PAGE_COST and SEQ_PAGE_COST configuration parameters.

Triggers Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE TRIGGER or CREATE CONSTRAINT TRIGGER statement, creating a trigger to be associated with a table or view. Optionally, an ALTER TABLE statement can be generated to control the firing of the trigger.

To create a new trigger using a wizard:

1. Open a creation wizard for a trigger. For details, see Opening 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 - Triggers (PostgreSQL) - Properties.
   - **Column Selection** panel - Triggers (PostgreSQL) - Column Selection.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Triggers (PostgreSQL) - Properties

When creating or editing a trigger, this tab/panel lets you work with the following settings:

Setting	Description
Parent Type	Select the type (TABLE or VIEW) of element that this trigger is to be associated with.
Parent Schema and **Parent Name**	Select the schema and name of the parent object.
Name	Provide a name for the trigger being created.
Constraint trigger, **Referenced Table Schema**, and **Referenced Table Schema**	Selecting the **Constraint Trigger** control specifies that a CREATE CONSTRAINT TRIGGER statement is generated instead of a CREATE TRIGGER statement. Optionally, you can use the **Referenced Table Schema**, and **Referenced Table Schema** controls to specify a FOR clause to indicate the referenced table.
Trigger Type	This control lets you specify a FOR clause, specifying whether the trigger is called once per row modified or once per operation. ROW specifies FOR EACH ROW while STATEMENT (not available if **Constraint Trigger** is selected) specifies FOR EACH STATEMENT.
Trigger Timing	Lets you select whether the trigger fires BEFORE or AFTER the operation.
Trigger Event List	Lets you build a list of events (INSERT, DELETE, UPDATE, and in the case of FOR EACH STATEMENT triggers, TRUNCATE) that cause the trigger to fire.
Deferrable and **Deferred**	These controls are only available if **Constraint Trigger** is selected. Selecting **Deferrable** generates a DEFERRABLE INITIALLY IMMEDIATE clause. Subsequently selecting **Deferred** changes the clause to DEFERRABLE INITIALLY DEFERRED.
Function Schema, **Function Name**, and **Function Arguments**	These controls let you specify details of the function to be called when the trigger fires.
Fire Mode	Optionally, lets you include an ALTER TABLE statement specifying an action of DISABLE TRIGGER, ENABLE ALWAYS TRIGGER. or ENABLE REPLICA TRIGGER. Selecting ORIGIN LOCAL results in no ALTER TABLE statement being appended to the main CREATE TRIGGER/CREATE CONSTRAINT TRIGGER statement.

Triggers (PostgreSQL) - Column Selection

If you chose UPDATE as the **Trigger Event List** option, this tab/panel lets you provide FOR UPDATE OF, BEFORE UPDATE OF, or AFTER UPDATE OF clause options. Select the check box beside each column that is to fire the trigger.

Types Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE TYPE, creating a new data type to be used in the current database. Optionally, an ALTER TABLE statement can be generated to modify a specific type.

To create a new type using a wizard:

1. Open a creation wizard for a type. For details, see Opening 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.
   - **Range Kind** panel.
   - **Base Kind** panel.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Finish** button to create the object.

Types (PostgreSQL) - Properties

When creating or editing a type, this tab/panel lets you work with the following settings:

Setting	Description
Schema	Schema where the new data type is created.
Name	Provide a name for the data type being created.
Kind	Indicates the kind of data of the data type being created. Options are PSEUDO, BASE, RANGE, ENUM and COMPOSITE.

Types (PostgreSQL) - Range Kind

Setting	Description
SubtypeSchema	Schema where the subtype is registered.
SubtypeName	Name of the element type that the range type will represent ranges of.
SubtypeOpClassFunctionSchema	Schema where the subtype operator class is registered.
SubtypeOpClassFunctionName	Name of a b-tree operator class for the subtype.
SubtypeCollationSchema	Schema where the collation is registered.
SubtypeCollationName	Name of an existing collation to be associated with a range type.
SubtypeCanonicalFunctionSchema	Schema where the canonicalization function is registered.
SubtypeCanonicalFunctionName	Name of the canonicalization function for the range type.
SubtypeDiffFunctionSchema	Schema where the difference function is registered.
SubtypeDiffFunctionName	Name of the difference function for the subtype.

Only SubtypeName is required, the other parameters are optional.

Types (PostgreSQL) - Base kind

Setting	Description
InputFunctionSchema	Schema where the input function is registered.
InputFunctionName	Name of a function that converts data from the type's external textual form to its internal form.
OutputFunctionSchema	Schema where the output function is registered.
OutputFunctionName	Name of a function that converts data from the internal form of the type to its external textual form.
RecvFunctionSchema	Schema where the receive function is registered.
RecvFunctionName	Name of a function that converts data from the external binary form of the type to its internal form.
SendFunctionSchema	Schema where the Send function is registered.
SendFunctionName	Name of a function that converts data from the internal form of the type to its external binary form.
ModInFunctionSchema	Schema where the type modifier input function is registered.
ModInFunctionName	Name of a function that converts an array of modifier for the types into internal form.
ModOutFunctionSchema	Schema where the type modifier output function is registered.
ModOutFunctionName	Name of a function that converts the internal form of the type's modifier to an external textual form.
AnalyzeFunctionSchema	Schema where the Analyze function is registered.
AnalyzeFunctionName	Name of a function that performs statistical analysis for the data type.
InternalLength	Numeric constant that specifies the length in bytes of the new type's internal representation. The default assumption is that is is variable-length.
Alignment	The storage alignment requirement of the data type. If specified, it must be char, int2, int4, or double; the default is int4.
PassByValue	Indicates whether the type can be passed by value or not. By default, is set to false.
Storage	The storage strategy for the data type. If specified, it must be plain, external, extended, or main; the default is plain.
Category	The category code (a single ASCII character) for this type. The default is 'U' for "user-defined type". Other standard category codes can be found in Table 45-45. You may also choose other ASCII characters in order to create custom categories.

IsPreferred	True if this type is a preferred type within its type category, else false. The default is false. Be very careful about creating a new preferred type within an existing type category, as this could cause surprising changes in behavior.
DefaultValue	The default value for the data type. If this is omitted, the default is null.
Delimiter	The delimiter character to be used between values in arrays made of this type.
IsCollatable	Indicates whether the type can use collation information or not. By default is set to false.
ElementSchema	Schema where the element is registered.
ElementName	The type being created is an array; this specifies the type of the array elements.

Only InputFunctionName and OutputFunctionName are required, the other parameters are optional.

Views Wizard (PostgreSQL)

This wizard lets you build and submit a CREATE VIEW or CREATE MATERIALIZED VIEW statement.

To create a new view using a wizard:

1. Open a creation wizard for a view. For details, see Opening 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 - Viewss (PostgreSQL) - Properties.
   - **Definition** panel - Views (PostgreSQL) - Definition.
   - **Comment** panel - for details, see Adding a Comment to an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Views (PostgreSQL) - Properties

When creating or editing a view, this tab/panel lets you work with the following settings:

Setting	Availability and Description
Schema	Lets you select the owner of the new view.
Name	Lets you provide the name of the new view.
Is Materialized	When selected, a CREATE MATERIALIZED VIEW statement is generated. When unselected, a CREATE VIEW statement is generated.
Security Barrier	This setting is only available with **Is Materialized** unselected. If selected, a WITH (security_barrier=true) clause is added to the generated DDL, specifying that the view is to provide row-level security.If unselected, a WITH (security_barrier=false) clause is added..
Tablespace	This setting is only available with **Is Materialized** selected. It lets you add a TABLESPACE clause to the generated DDL, specifying the tablespace where the materialized view is to be created.
Fill Factor	This setting is only available with **Is Materialized** selected. It lets you add a WITH (fillfacter= ...) clause to the generated DDL, specifying a an integer expressing a percentage between 10 and 100, dictating the upper limit to which INSERTs can pack pages.
With Data	This setting is only available with **Is Materialized** selected. If selected, a WITH DATA clause is appended to the generated DDL, specifying that the backing query is executed to provide new data. If unselected, a WITH NO DATA clause is appended to the generated DDL, specifying that no new data is generated.
View Autovacuum... settings	Lets you enable or disable the autovacuum daemon as well as specify values for parameters used in autovacuum calculations (Threshold, Scale Factor, Analyze Threshold, Analyze Threshold, Cost Delay, Cost Limit, Freeze Min Age, Freeze Max Age, Freeze Table Age, Multixact Freeze Min Age, Multixact Freeze Max Age, and Multixact Freeze Table Age.
Toast View Autovacuum... settings	Lets you enable or disable the autovacuum daemon for the table's secondary toast table, as well as specify values for parameters used in autovacuum calculations (Threshold, Scale Factor, Cost Delay, Cost Limit, Freeze Min Age, Freeze Max Age, Freeze Table Age, Multixact Freeze Min Age, Multixact Freeze Max Age, and Multixact Freeze Table Age.

Views (PostgreSQL) - Definition

Complete the CREATE VIEW statement by typing or pasting in the relevant query.

Sybase ASE Object Wizards

You create Sybase ASE objects using the following wizards:

- Aliases Wizard (Sybase ASE)
- Databases Wizard (Sybase ASE)
- Defaults Wizard (Sybase ASE)
- Extended Procedures Wizard (Sybase ASE)
- Foreign Keys Wizard (Sybase ASE)
- Functions Wizard (Sybase ASE)
- Groups Wizard (Sybase ASE)
- Indexes Wizard (Sybase ASE)
- Logins Wizard (Sybase ASE)
- Primary Keys Wizard (Sybase ASE)
- Procedures Wizard (Sybase ASE)
- Rules Wizard (Sybase ASE)
- Segment Wizard (Sybase ASE)
- Tables Wizard (Sybase ASE)
- Triggers Wizard (Sybase ASE)
- Unique Keys Wizard (Sybase ASE)
- User Datatypes Wizard (Sybase ASE)
- User Messages Wizard (Sybase ASE)
- Users Wizard (Sybase ASE)
- Views Wizard (Sybase ASE)

Aliases Wizard (Sybase ASE)

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 a creation wizard for an alias. For details, see Opening 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 Aliases (Sybase ASE) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Aliases (Sybase ASE) - 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.

Databases Wizard (Sybase ASE)

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 a creation wizard for a database. For details, see Opening 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 Databases (Sybase ASE) - Properties.
   - **Placement** panel - for details, see Databases (Sybase ASE) - Placement.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Databases (Sybase ASE) - Properties

When creating or editing a database, this tab/panel lets you work with the following settings:

Category	Setting (and availability)	Description
Creation Properties	**Database Name**	Provide a name for the database.
	Type	Generate variations on the CREATE DATABASE DDL generated to create the database, as follows: DEFAULT - generates a generic CREATE DATABASE statement. TEMP - generates a CREATE TEMPORARY DATABASE variation. ARCHIVE - generates a CREATE ARCHIVE DATABASE variation.
	In-memory (available with a **Type** of TEMP or DEFAULT) This feature is available as of Sybase ASE 15.5.	Generates CREATE INMEMORY DATABASE DDL, indicating that this is an in-memory database. If you are using Rapid SQL, an **in-memory** database must be created against a device initialized (DISK INITI) using the **inmemory** parameter.

	Scratch Database (available with a **Type** of ARCHIVE)	Lets you provide a WITH SCRATCH_DATABASE parameter value, specifying the name of an existing database that stores information about the archive database.
	For Proxy Update (available with a **Type** of DEFAULT and not available if **For Load** is selected) This feature is available as of Sybase ASE 12.5.	Adds a FOR PROXY_UPDATE parameter to the generated DDL, forcing resynchronization of proxy tables with the proxy database. For information on how to resynchronize an existing database, see Resynchronize.
	For Load (available with a **Type** of DEFAULT and not available if **For Proxy Update** is selected)	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 (available with a **Type** of TEMP or DEFAULT)	Enable this option to specify whether logs and data are to be kept on the same logical device
	With Default Location (available with a **Type** of DEFAULT)	Lets you add a WITH DEFAULT_LOCATION clause value, specifying a storage location for new tables. If you also select **For Proxy Update**, one proxy table will be automatically created for each remote table or view.
	Durability (available with a **Type** of TEMP or DEFAULT) Relaxed Durability databases are available as of Sybase ASE 15.5.	Lets you select a WITH DURABILITY= parameter value of NO RECOVERY, FULL, or AT SHUTDOWN, specifying the durability level of the database. With **In-memory** selected, the only valid selection is NO RECOVERY.
Options	The GENERATE... DDL is followed by an sp_dboption call for each option selected in this group.	
	With a **Type** of ARCHIVE	**single user** - generates a sp_dboption call specifying the SINGLE USER option, restricting database access to one user at a time.
	With a **Type** of TEMP or DEFAULT	In addition to the **single user** option, the following sp_dboption database options are available: **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**. For Sybase ASE 15.0 databases, **scratch database**, **async log service option**, and **delayed commit** options are available.

Databases (Sybase ASE) - 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. For databases with a **Properties** tab/panel **Type** selection of ARCHIVE, no LOG ONLY **Device Type** selection is available.

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.

Defaults Wizard (Sybase ASE)

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 a creation wizard for a default. For details, see Opening 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 Defaults (Sybase ASE) - Properties.
    - **Dependencies** panel - for details, see Defaults (Sybase ASE) - Dependencies.
    - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Defaults (Sybase ASE) - 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 ASE) - 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 Procedures Wizard (Sybase ASE)

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 a creation wizard for an extended procedure. For details, see Opening 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 Extended Procedures (Sybase ASE) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Extended Procedures (Sybase ASE) - 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 Keys Wizard (Sybase ASE)

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. The 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 a creation wizard for a foreign key. For details, see Opening 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 Foreign Keys (Sybase ASE) - Properties.
   - **Column Mapping** panel - for details, see Foreign Keys (Sybase ASE) - Column Mapping.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Foreign Keys (Sybase ASE) - 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 ASE) - 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 ASE)

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 a creation wizard for a function. For details, see Opening 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 Functions (SQL Server) - Properties.
   - **Definition panel** - for details, see Functions (SQL Server) - Definition.
3. Finally, use the **Execute** button to create the object.

Functions (Sybase ASE) - 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 ASE) - Definition

Complete the CREATE FUNCTION outline provided by typing or pasting the body of the function.

Groups Wizard (Sybase ASE)

A group is a collection of privileges that the DBA assigns to designated users.

To create a new group using a wizard:

1. Open a creation wizard for a group. For details, see Opening 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 Groups (Sybase ASE) - Properties.
   - **Object Permissions** and **System Permissions** panels - see Setting Permissions or Privileges for an Object.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

After you click **Finish** and **Execute**, use the Group Editor to assign users to the new group. For details, see Groups Editor (Sybase ASE).

Groups (Sybase ASE) - Properties

When creating a group, this panel lets you provide a name for the group.

Indexes Wizard (Sybase ASE)

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 a creation wizard for an index. For details, see Opening 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 Indexes (Sybase ASE) - Properties.
   - **Columns** panel - for details, see Indexes (Sybase ASE) - Columns.
   - **Partition** panel - for details, see Indexes (Sybase ASE) - Partition.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Indexes (Sybase ASE) - Properties

When creating or editing an index, this tab/panel lets you work with the following settings:

Setting	Description
Table Owner, **Table Nam**e, 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 ASE) - 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 ASE) - Partition

Click **Add** to open the Index Partition wizard (Sybase ASE).

Index Partition wizard (Sybase ASE)

The Sybase ASE Index Partition wizard can be opened from the following editors and wizards:

- Indexes Wizard (Sybase ASE)
- Indexes Editor (Sybase ASE)

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.

Logins Wizard (Sybase ASE)

The Login wizard lets you build and submit an **sp_addlogin** or CREATE LOGIN 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 a creation wizard for a login. For details, see Opening 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 Logins (Sybase ASE) - Properties.
   - **Roles** panel - for details, see Logins (Sybase ASE) - Roles.
   - **Users** panel - for details, see Logins (Sybase ASE) - Users.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Note: Login triggers, configured for individual logins, are also supported. For details, Logins Editor (Sybase ASE).

Logins (Sybase ASE) - 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.
	Suid (version 15.7 ^)	Lets you optionally add a SUID parameter to the generated DDL, specifying a system user ID.
General	**Default Database**	The database where the user starts each session.
	Default language	The language in which prompts and messages are displayed.

	Profile (version 15.7 ^)	Lets you add a LOGIN PROFILE parameter: **ignore** eliminates profile binding while selecting a login profile (if available) binds the profile to this login.
	Locked or **Account Locked**	If selected, the **sp_addlogin** or CREATE LOGIN 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 ASE).	
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.
	Exempt Inactive Lock (version 15.7 ^)	When selected, an EXEMPT INACTIVE LOCK TRUE parameter is added to the generated DDL, exempting the login from being locked due to inactivity.

Logins (Sybase ASE) - 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 Role Activation.

Logins (Sybase ASE) - 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.

Adding a login account to a database

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 Logins Editor (Sybase ASE).

Primary Keys Wizard (Sybase ASE)

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 a creation wizard for a primary key. For details, see Opening 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 Primary Keys (Sybase ASE) - Properties.
   - **Columns** panel - for details, see Primary Keys (Sybase ASE) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Primary Keys (Sybase ASE) - 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 ASE) - 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.

Procedures Wizard (Sybase ASE)

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 a creation wizard for a procedure. For details, see Opening 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 Procedures (Sybase ASE) - Properties.
   - **Definition** panel - for details, see Procedures (Sybase ASE) - Definition.
3. Finally, use the **Execute** button to create the object.

Procedures (Sybase ASE) - 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 ASE from saving a plan for the procedure.

Procedures (Sybase ASE) - Definition

Complete the provided CREATE PROCEDURE statement by typing or pasting the body of the procedure.

Rules Wizard (Sybase ASE)

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 a creation wizard for a rule. For details, see Opening 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 Rules (Sybase ASE) - Properties.
   - **Dependencies** panel - for details, see Rules (Sybase ASE) - Dependencies.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Rules (Sybase ASE) - 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 ASE) - 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 ASE)

This wizard lets you build and submit an `sp_addsegment` procedure call, optionally followed by `sp_dbextend 'set', 'database'`... and `sp_dbextend 'set', 'threshold'`... calls. This defines a segment on a database device, and specifies the size by which to grow the database, its maximum size, and a free space level.

To create a new segment using a wizard:

1. Opening an Object Wizard for a segment.
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 Segments (Sybase ASE) - Properties.
   - **Location** panel - for details, see Segments (Sybase ASE) - Location.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object. For more information, see Previewing the DDL Generated to Create the New Object.

The Segments Editor opens. You can extend the segment and see the objects stored in the segment. For more information, see Segments Editor (Sybase ASE).

Segments (Sybase ASE) - Properties

When creating or editing a segment, this tab/panel lets you work with the following settings:

Setting	Description
Name	Provide a **Name** for the segment.
Grow By	Provide an `sp_dbextend 'set', 'database'`... growth rate parameter value, specifying the rate at which the segment grows each time threshold procedures are run.
Threshold Free Space	Provide an `sp_dbextend 'set', 'threshold'`... free space parameter value, specifying when the threshold procedure is installed on the segment.
Max Size	Provide an `sp_dbextend 'set', 'database'`... maximum segment size parameter value.

Segments (Sybase ASE) - Location

When creating a segment, this tab/panel lets you select the database device for the segment. Select a device from the **Available Devices** list and click **Place** to include the device in the **Used Devices** list.

Similarly, when editing a segment, this tab/panel lets you extend a segment by adding additional devices. Select a device from the **Available Devices** list and click **Extend** to include the device in the **Used Devices** list.

Tables Wizard (Sybase ASE)

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 a creation wizard for a table. For details, see Opening 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 Tables (Sybase ASE) - Properties.
   - **Columns** panel - for details, see Tables (Sybase ASE) - Columns.
   - **Indexes** panel - for details, see Tables (Sybase ASE) - Indexes.
   - **Constraints** panel - for details, see Tables (Sybase ASE) - Constraints.
   - **Partitions** panel - for details, see Tables (Sybase ASE) - Partition.
   - **Permissions** panel - for details, see Tables (Sybase ASE) - Permissions.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

Tables (Sybase ASE) - 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 ASE) - 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 ASE) - Indexes

Click **Add** to open the Index Wizard. For details, see Indexes Wizard (Sybase ASE).

Tables (Sybase ASE) - Constraints

Selecting a constraint type and clicking **Add** opens the object wizard for that object type. For details see:

- Primary Keys Wizard (Sybase ASE)
- Unique Keys Wizard (Sybase ASE)
- Foreign Keys Wizard (Sybase ASE)
- Create Synonym

Tables (Sybase ASE) - Partition

Click **Add** to open the Table Partition Wizard. For details, see Table Partition wizard for Sybase ASE.

Tables (Sybase ASE) - Permissions

Set up the user permissions for this table.

Table Partition wizard for Sybase ASE

See the following topics for information on editors and wizards that let you open the Sybase ASE Table Partition Wzard:

- **Tables Wizard (Sybase ASE)**
- Tables Editor (Sybase ASE)

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.
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.

Triggers Wizard (Sybase ASE)

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 ASE 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 a creation wizard for a trigger. For details, see Opening 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 Triggers (Sybase ASE) - Properties.
   - **Definition panel** - for details, see Triggers (Sybase ASE) - Definition.
3. Finally, use the **Execute** button to create the object.

Triggers (Sybase ASE) - 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 ASE).

Triggers (Sybase ASE) - 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 Keys Wizard (Sybase ASE)

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 a creation wizard for a unique key. For details, see Opening 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 Unique Keys (Sybase ASE) - Properties.
   - **Columns** panel - for details, see Unique Keys (Sybase ASE) - Columns.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the 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 ASE) - 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 ASE) - 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 Datatypes Wizard (Sybase ASE)

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 a creation wizard for a user datatype. For details, see Opening 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 Datatypes (Sybase ASE) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

User Datatypes (Sybase ASE) - 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 Messages Wizard (Sybase ASE)

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 a creation wizard for a user message. For details, see Opening 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 (Sybase ASE) - Properties.
   - **Information** panel - for details, see User Messages (Sybase ASE) - Information.
   - **Bindings** panel - for details, see User Messages (Sybase ASE) - Bindings.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

User Messages (Sybase ASE) - 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 ASE) - 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 ASE) - 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**.

Users Wizard (Sybase ASE)

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 a creation wizard for a user. For details, see Opening 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 Users (Sybase ASE) - Properties.
   - **DDL View** panel - for details, see Previewing the DDL Generated to Create the New Object.
3. Finally, use the **Execute** button to create the object.

You can use the User Editor to assign System and Object privileges as well as create an alias.

Users (Sybase ASE) - 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.

Views Wizard (Sybase ASE)

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 a creation wizard for a view. For details, see Opening 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 Views (Sybase ASE) - Properties.
   - **Definition** panel - for details, see Views (Sybase ASE) - Definition.
3. Finally, use the **Execute** button to create the object.

You can use the User Editor to assign System and Object privileges as well as create an alias.

Views (Sybase ASE) - 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 ASE) - Definition

Complete the CREATE VIEW statement by typing or pasting in the relevant query.

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 Overview and common usage of object editors.

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.

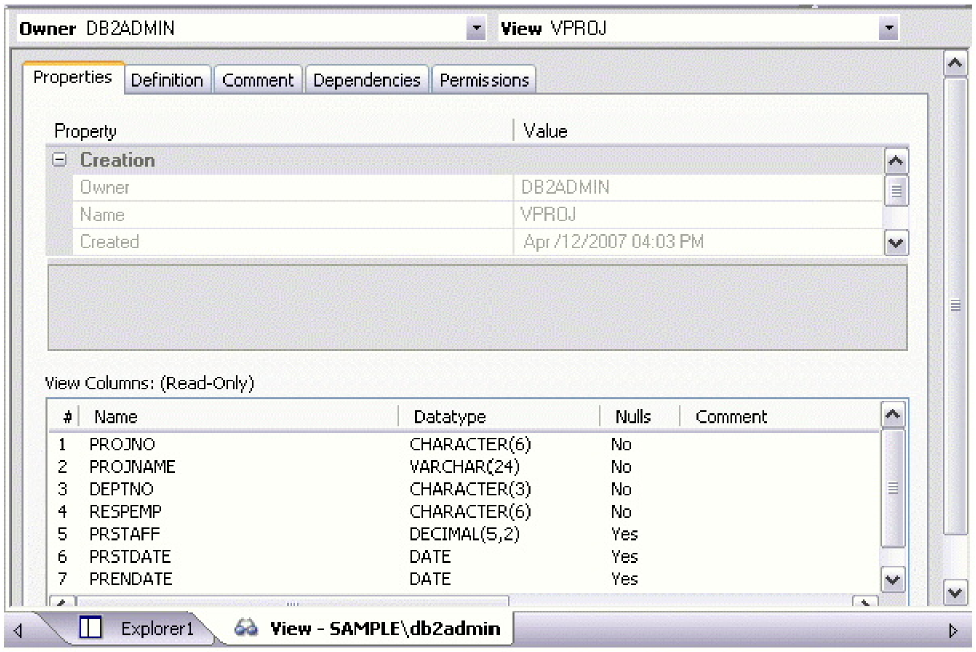
See the following topics for information the object editors available for each supported platform:

- IBM DB2 for Linux, Unix, and Windows Object Editors
- IBM DB2 for z/OS Object Editors
- InterBase/Firebird Object Editors
- Microsoft SQL Server Object Editors
- MySQL Object Editors
- Oracle Object Editors
- PostgreSQL Object Editors
- Sybase ASE Object Editors

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:



Property	Value
Creation	
Owner	DB2ADMIN
Name	VPROJ
Created	Apr /12/2007 04:03 PM

View Columns: (Read-Only)

#	Name	Datatype	Nulls	Comment
1	PROJNO	CHARACTER(6)	No	
2	PROJNAME	VARCHAR(24)	No	
3	DEPTNO	CHARACTER(3)	No	
4	RESPEMP	CHARACTER(6)	No	
5	PRSTAFF	DECIMAL(5,2)	Yes	
6	PRSTDATE	DATE	Yes	
7	PRENDATE	DATE	Yes	

In order to work with object editors, you must be familiar with the following tasks:

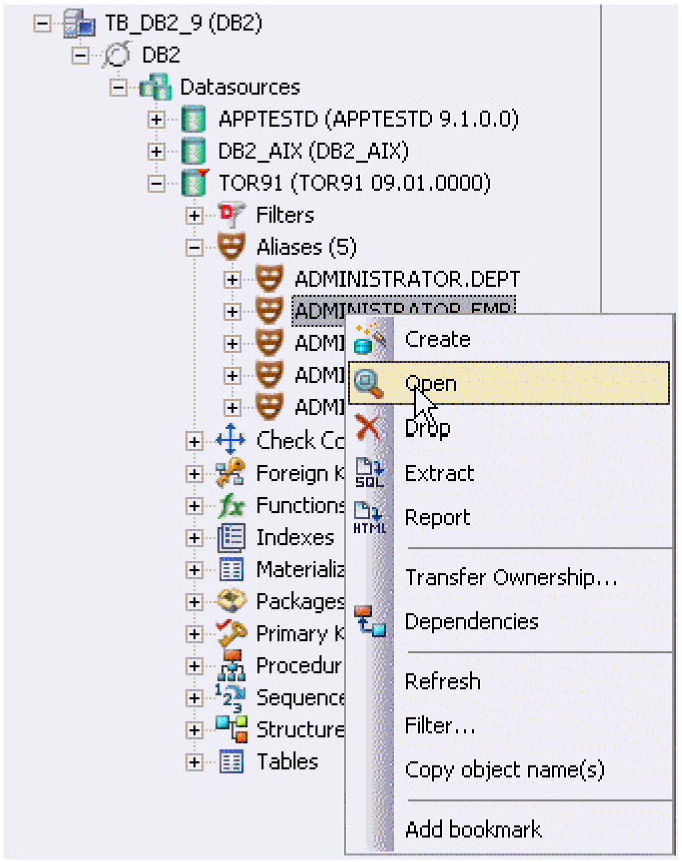
- Opening an Object Editor on a server or database object.
- Viewing and Modifying Object Properties using functionality common to all object editors as well as object-specific functionality available to specific object editors.
- Previewing and Submitting Object Editor Changes to effect your changes to the datasource.

Opening an Object Editor

Object editors are accessed from the Datasource Navigator pane of the main window.

To open an Object editor on a specific database object in Rapid SQL

1. Connect to the datasource with the database object to be edited. For more information, see Datasource and Server Management.
2. On the Datasource Navigator, 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.



An editor for that object type opens 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.

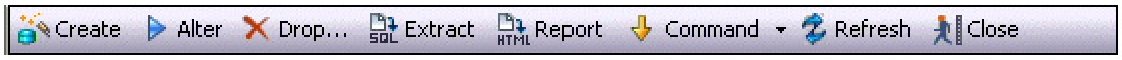
Work with Tasks and Settings Common to Object Editors

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:

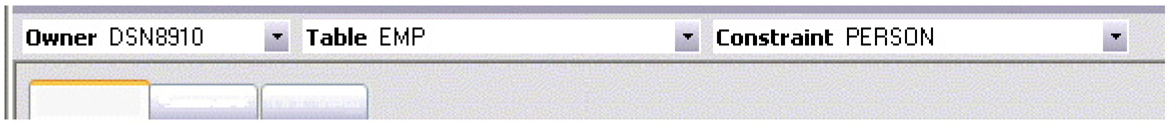
- **Create** - Launches an object creation wizard. For more information, see Creating objects.

- **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** - 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.

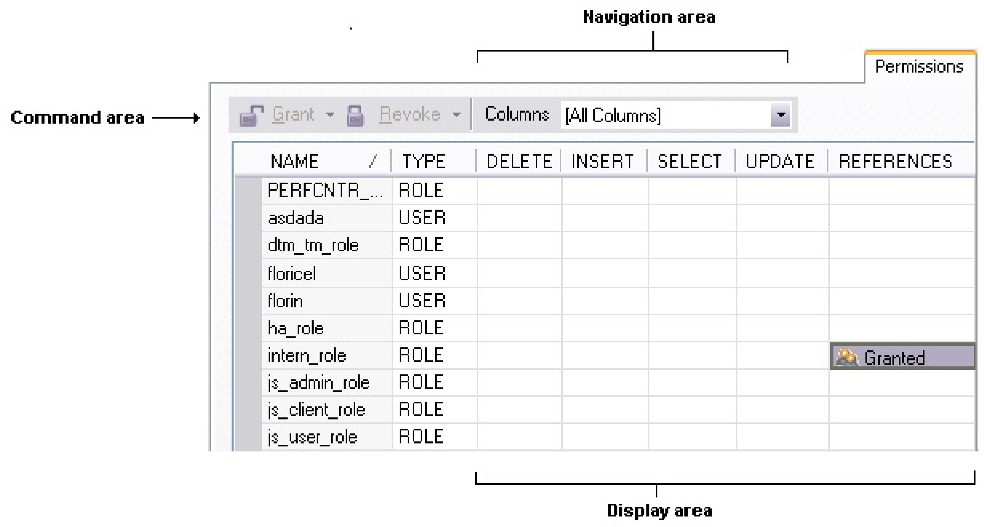
See the topics for specific Object editors later in this chapter for information on whether that object type supports display of underlying SQL code.

2. Click the **DDL** tab.

Working with Privileges and Permissions

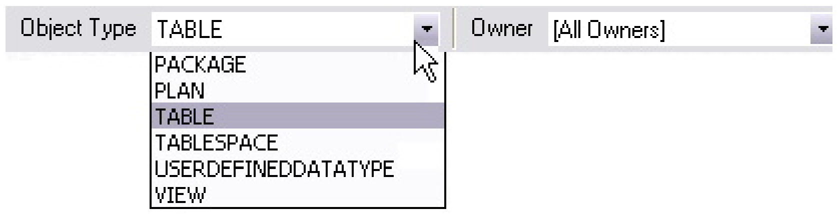
When you open an Object editor on an object with associated privileges, the **Permissions** (or **Object Permissions** or **System Permissions**) tab for that editor displays the relevant privileges and lets you make changes accordingly:

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.



NAME	TYPE	DELETE	INSERT	SELECT	UPDATE	REFERENCES
PERFCNTR_...	ROLE					
asdada	USER					
dtm_tm_role	ROLE					
floricel	USER					
florin	USER					
ha_role	ROLE					
intern_role	ROLE					Granted
js_admin_role	ROLE					
js_client_role	ROLE					
js_user_role	ROLE					

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.

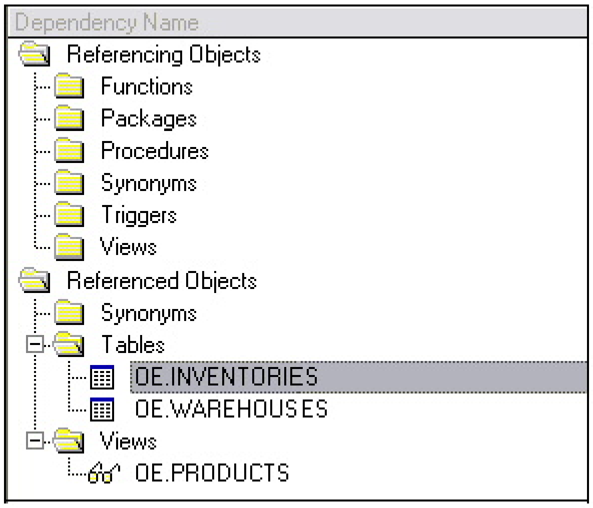
The Command area lets you initiate granting or revocation of permissions.

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, you can 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:

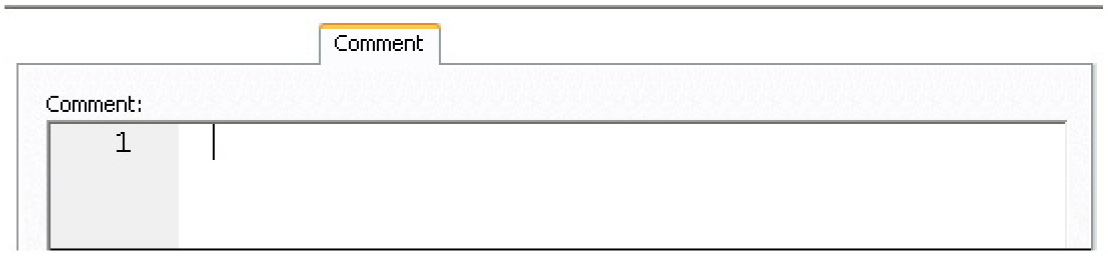
   - Use the **Edit** button to open an Object editor on that object
   - Use the **Drop** button to initiate dropping that object.

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.

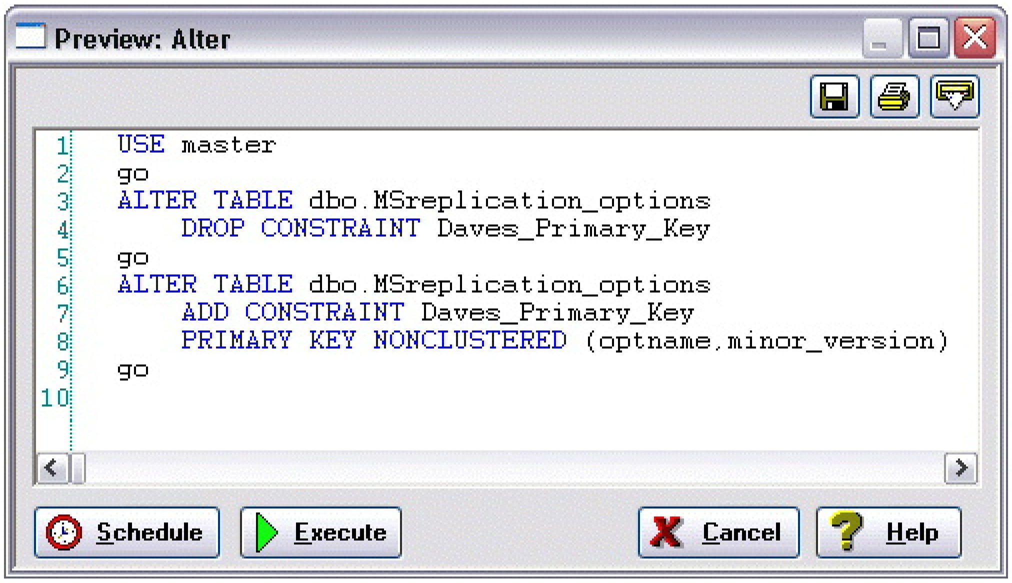
Previewing and Submitting Object Editor Changes

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.



```
USE master
go
ALTER TABLE dbo.MSreplication_options
    DROP CONSTRAINT Daves_Primary_Key
go
ALTER TABLE dbo.MSreplication_options
    ADD CONSTRAINT Daves_Primary_Key
    PRIMARY KEY NONCLUSTERED (optname,minor_version)
go
```

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

An Object Editor is offered 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)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

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 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 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 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.

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 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
Placement	Select a tablespace and click the **Edit** button to open an object editor on that tablespace. For more information, see Tablespaces Editor (IBM DB2 LUW).
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 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 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 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 **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 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.

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 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 **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.
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 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. 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**, **Allow Reverse Scans**, and **Compression**.
	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 Adding a Comment to an object.	
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.

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 initiallyData 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 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 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 Using the SQL Editor.
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.

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 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 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.	
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 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.

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 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 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 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.

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 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 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 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.

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 Types 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 Types 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.
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 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 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 Opening an Object Editor.
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. 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**	[JavaScript:popup.TextPopup(Poptext1,Popfont,9,9,-1,-1) Lets you view the **Database partition Group**], **Managed By**, **Page Size**, and **Extent Size** properties. Lets you choose the **Name** of a tablespace.
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, SQL script that completes the following steps is constructed:

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, any permissions on them are also rebuilt.

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.
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 **DatabasePartitions**, **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 Adding a Comment to an object.
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 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.

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 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**, **Define**r, 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 view and modify the CREATE TRIGGER DDL that will implement your changes.

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	[JavaScript:popup.TextPopup(Poptext1,Popfont,9,9,-1,-1) 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, the existiing view is automatically dropped and then recreated.
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

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)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

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.
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 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 specify a **Delete Rule** (CASCADE, NO ACTION, RESTRICT, SET NULL).
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 (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 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:	
	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.
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 Adding a Comment to an object.
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 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 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 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.
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.
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 Using the SQL 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 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	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.	
Storage	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 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.

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 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:	
	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 Adding a Comment to an object.
Body	Lets you modify the SQL code for procedures on the current datasource
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.

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	[JavaScript:popup.TextPopup(Poptext1,Popfont,9,9,-1,-1) 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.
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 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 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, an SQL script that completes the following steps is constructed:

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, any permissions on them are also rebuilt.

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.

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 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	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 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 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 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** and **Size** settings.
	Character Options	Lets you set the **For Data** and **CCSID** properties.
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.	
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 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 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 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.
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)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

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 Wizard (ITB/FBD).
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 (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD) - 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 (ITB/FBD) - 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.
Return Type	Displays datatype details of the value returned by the external function.
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.

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 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 **Delete rule** and **Update Rule** actions. For details on these properties, see Foreign keys (ITB/FBD) - 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.

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 (ITB/FBD) - 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 (ITB/FBD) - 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.
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 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 **Shadow Set** number and lets you modify the **Conditional** and **Behavior** properties. For details on these properties, see Shadows (ITB/FBD) - Properties.
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 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 (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 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. For details on the properties, see Tables (ITB/FBD) - Columns. 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**.
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 (ITB/FBD). 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 (ITB/FBD) - 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 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 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 Users (ITB/FBD) - Properties.
Comment	For details on using this tab, see Adding a Comment to an object.
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 Users (ITB/FBD) - Roles. To assign roles to the user, select the associated check box.
Object 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.

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 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 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 Adding a Comment to an object.
Definition	Lets you view and modify the CREATE VIEW DDL for the view.
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.

Microsoft SQL Server Object Editors

An Object Editor is provided for all supported Microsoft SQL Server objects. To see an Editor for a specific object, click the corresponding link below:

- Asymmetric Keys Editor (SQL Server)
- Certificates Editor (SQL Server)
- 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)
- Roles Editor (SQL Server)
- Rules Editor (SQL Server)
- Schemas Editor (SQL Server)
- Sequences Editor (SQL Server)
- Symmetric Key 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)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

Asymmetric Keys Editor (SQL Server)

The Asymmetric Keys Editor lets you view and modify asymmetric key properties.

To edit an asymmetric key

1. Open an editor on the asymmetric 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	This tab lets you view all properties assigned when the asymmetric key was created. For more information, see Asymmetric Keys (SQL Server) - Properties. In addition, this tab lets you modify the following properties: **Authorization Owner** - the AUTHORIZATION argument. **Change Encryption Mechanism** - activates the **Encryption Type** control, letting you change the encryption method used for this certificate. **Encryption Type** - lets you select among encryption using the MASTER KEY, a USER_PASSWORD, or NONE. **Encryption Password** and **Decryption Password** - available if you select an **Encryption Type** of USER_PASSWORD, they lets you provide the specific encryption/decryption passwords.
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.

Certificates Editor (SQL Server)

The Certificates Editor lets you view and modify check certificate properties.

To edit a certificate

1. Open an editor on the certificate. 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	This tab lets you view all properties assigned when the certificate was created. For more information, see Certificates (SQL Server) - Properties. In addition, this tab lets you modify the following properties: **Authorization Owner** - the AUTHORIZATION argument. **Active for Begin Dialog** - the ON/OFF values for a ACTIVE FOR BEGIN_DIALOG = option. **Change Encryption Mechanism** - activates the **Encryption Type** control, letting you change the encryption method used for this certificate. **Encryption Type** - lets you select among encryption using the MASTER KEY, a USER_PASSWORD, or NONE. **Encryption Password** - available if you select an **Encryption Type** of USER_PASSWORD, it lets you provide the specific encryption password.
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 (SQL Server)

The Check Constraints Editor lets you 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 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	
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**, **FileGrowth 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**, **FileGrowth 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 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 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.

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 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 provide a dynamic-link library (DLL) **Library Name** for the extended 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.

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 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 specify **Enabled**, **With NOCHECK**, 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 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 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.
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 Indexes Wizard (SQL Server).
	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 Indexes Wizard (SQL Server). **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).
	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 Viewing the SQL/DDL for an Object.

3. When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

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 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 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 Logins Wizard (SQL Server). 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 Logins (SQL Server) - Server Roles.
Users	Lets you add and remove user accounts for this login to specified databases. For more detailed information, see Logins (SQL Server) - Users.
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.

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 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**, **Input Parameter Type**, **Range**, and **Function Values** properties for the partition function. For more information on these properties, see Partition Functions (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.

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.
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 Keys 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 Previewing and Submitting Object Editor Changes.

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 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 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 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.

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 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 whether the rule is **Enabled** and view the **Fire On Inser**t, **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 Working with Object Dependencies.

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.

Sequences Editor (SQL Server)

This editor lets you view creation properties for a sequence as well as modify the scope, starting value and increment value.

To edit a sequence:

1. Open an editor on the sequence. 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**, **Create Date**, **Data Type Schema**, **Base Data Type**, **Data Type Name**, and **Data Type Precision** properties for the synonym. Lets you modify the **Start With**, **Increment By**, **Minimum Value**, **Maximum Value**, **Cycle**, **Cache**, and **Cache Size** properties. For more information on these properties, see Sequences (SQL Server) - Properties. This tab also includes the following, display-only properties: **Current Value** - The value most recently returned by the NEXT VALUE FOR function or the last value resulting from an `sp_sequence_get_range` procedure execution. **Is Exhausted** - Indicates that the sequence object has reached the **Maximum Value** and **Cycle** is not set. The NEXT VALUE FOR function will return an error until the sequence is restarted using ALTER SEQUENCE.
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.

Symmetric Key Editor (SQL Server)

The Symmetric Key Editor lets you view creation details for a key and modify its encryption mechanisms.

To edit a symmetric key:

1. Open an editor on the symmetric 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	Lets you view the **Authorization Owner**, **Name**, **Create Date**, **Update Date**, **Provider Name**, **Key Algorithm**, and **Key Length** properties. For more information, see Symmetric Keys (SQL Server) - Properties.
Encryption Mechanisms	Lets you manage the encryption mechanisms for this key. For detailed instructions, see Symmetric Keys (SQL Server) - 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.

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.
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 (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 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 settings in the following categories:	
	Physical Storage	Lets you view the **Partitioned**, **Text Image Filegroup**, and **FileStream On** (.version 2008^) properties and modify the **Filegroup** property. For details on these properties, see Tables (SQL Server) - Properties.
	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.
	Filetable (Version 2012^)	Lets you view the **As Filetable**, **Filetable Collate Filename**, **FiletablePK Constraint Name**, **Filetable StreamID Unique Constraint Name**, and **Filetable Fullpath Unique Constraint Name** properties and lets you modify the **Filetable Directory Name** property. For details on these properties, see Tables (SQL Server) - Properties. This group also lets you modify an **Enabled** property. That control specifies whether system-defined constraints are enabled.

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 Indexes Wizard (SQL Server). Select an index and click **Edit** to open an wizard that lets you modify that index. For more information, see Indexes Editor (SQL Server). 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 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.

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 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 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.
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. The existing trigger is automatically dropped and then recreated.
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 (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 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 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.
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 Keys 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 Defaults Wizard (SQL Server). **Rule Binding** - Lets you select a rule. For information on creating rules, see Rules Wizard (SQL Server).
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 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 (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 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:	
	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.	
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 (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 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 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 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.

MySQL editors

An Object Editor is offered for all supported MySQL objects. To see an Editor for a specific object, click the corresponding link below:

- Database editor (MySQL)
- Foreign Keys editor (MySQL)
- Functions editor (MySQL)
- Indexes, Primary Keys, and Unique Keys editors (MySQL)
- Tables editor (MySQL)
- Users editor (MySQL)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

Database editor (MySQL)

The Databases Editor lets you manage basic properties and privileges 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	Lets you modify the **Default Character Set** and **Collation** properties. For details on these properties, see Databases - Properties.
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.

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.
Foreign Keys	For INNODB tables, this tab lets you add, delete, or modify table foreign keys. For details on available options, see Tables - Foreign Keys.
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 Tables - MERGE Tables.
Privileges	For details on using this tab, see Working with Privileges and Permissions.
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 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 (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 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
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 Users wizard (MySQL).
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 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.

Oracle Object Editors

An Object Editor is available 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)
- Jobs 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)
- Programs Editor (Oracle)
- Profiles Editor (Oracle)
- Redo Log Groups Editor (Oracle)
- Roles Editor (Oracle)
- Rollback Segments Editor (Oracle)
- Schedules 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)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

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 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	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.
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 Previewing and Submitting Object Editor Changes.

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 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 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.

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 **Partition Commands** menu items to initiate object actions. For details, see the following topics: Allocate ExtentAnalyzeCoalesceDeallocate 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 Editor lets you change the sql procedure specified for the job as well as manage the job's schedule and status.

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
Properties	Lets you modify the **Submit as disabled**, **Next Date**, **Run Again**, **Every**, **Time Unit**, **Use Custom Expression**, and **Custom Expression** properties. For details on these properties, see Job Queues (Oracle) - Properties. This tab also displays the following properties: **Job** - the unique identifier assigned when the job was created. **User** - the user who submitted the job. **Schema User** - the user schema for which to parse the job **Priv User** - the user whose privileges the job is to run under.
Definition	This tab lets you modify the DBMS_JOB.SUBMIT call's WHAT parameter, specifying the name of the PL/SQL procedure to run.
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.

Jobs Editor (Oracle)

The Jobs Editor lets you modify details of a job.

To edit a job

1. Open an editor on the job. 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	For details on using these settings, see Jobs (Oracle) - 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.

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.
	Partitioning Columns	Displays partitioning columns.
	Subpartitioning Columns	Displays subpartitioning columns.
	Partitions	Click **Add** or **Edit** to open the **Partition** dialog. 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. Click **Drop** to drop a subpartition.
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 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 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 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.
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 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.

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 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 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 Viewing the SQL/DDL for an Object.

3. When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

Package Bodies Editor (Oracle)

While listed as separate objects in the Datasource Navigator, 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 ModifiedSource 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.
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 **Partition Commands** menu items to initiate object actions. For details, see the following topics: Allocate ExtentAnalyzeCoalesceDeallocate 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.

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 (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 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:	
	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 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.

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**, **ReuseTime**, **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.

Programs Editor (Oracle)

The Programs Editor lets you modify details of a program.

To edit a program

1. Open an editor on the program. 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
Action	For details on using these settings, see Programs (Oracle) - Action.
Arguments	For details on using these settings, see Programs (Oracle) - Arguments.
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.

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 Role Activation.

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.

Schedules Editor (Oracle)

The Schedules Editor lets you modify details of a schedule.

To edit a schedule

1. Open an editor on the schedule. 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
Action	For details on using these settings, see Schedules (Oracle) - 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.

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 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 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 Working with Object Dependencies.	
Privileges	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.

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 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 type, owner, name, and other details of the object referenced by the synonym.
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 (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.

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	
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.
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:

- o Users with privileges to create certain types of objects can create those objects in the specified tablespace.
- o 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.

The table below describes the options and functionality on the **Add User Quota...** or **Edit User Quota...** dialogs:

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.

Oracle 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.

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, COMPOUND), 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. For detailed information on these properties, see Triggers (Oracle) - Properties.
Events	Lets you select the DDL, DML, or Database manipulation events that fire this trigger. For detailed information,see Triggers (Oracle) - Events.
Column Selection	For triggers firing on UPDATE events, lets you work with the associated columns. For detailed information,see Triggers (Oracle) - Column Selection.
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. For detailed information,see Triggers (Oracle) - Action.
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 Navigator, 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 Types 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 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 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.
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 **Partition Commands** menu items to initiate object actions. For details, see the following topics: Allocate ExtentAnalyzeCoalesceDeallocate Unused Space**Mark UnusableRebuildSplit**
	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 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: **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 Profiles Wizard (Oracle). **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 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 (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 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 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 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.

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 Opening an Object Editor.
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.
Hash Definitions (only available for a **Partition Type** of HASH)	Lets you specify a Partition method: **NoneNumber 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**.

PostgreSQL Object Editors

You modify PostgreSQL objects using the following wizards:

- Check Constraints Editor (PostgreSQL)
- Domains Editor (PostgreSQL)
- Exclusion Constraints, Primary Keys, or Unique Keys Editors (PostgreSQL)
- Foreign Keys Editor (PostgreSQL)
- Functions Editor (PostgreSQL)
- Indexes Editor (PostgreSQL)
- Roles Editor (PostgreSQL)
- Rules Editor (PostgreSQL)
- Schemas Editor (PostgreSQL)
- Tables Editor (PostgreSQL)
- Tablespaces Editor (PostgreSQL)
- Triggers Editor (PostgreSQL)
- Types Editor (PostgreSQL)
- Views Editor (PostgreSQL)

Check Constraints Editor (PostgreSQL)

The Check Constraints Editor lets you view or modify check constraints properties.

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
Properties	Lets you view or modify properties defined when creating the object. For details on these properties, see Check Constraints Wizard (PostgreSQL).
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.

Domains Editor (PostgreSQL)

The Domains Editor lets you view or modify domain properties.

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	Lets you view or modify properties defined when creating the object. For details on these properties, see Domains Wizard (PostgreSQL).
Constraints	Lets you add, drop, or edit constraints on this object. For details on these properties, see Domains Wizard (PostgreSQL).
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.

Exclusion Constraints, Primary Keys, or Unique Keys Editors (PostgreSQL)

The Exclusion Constraints Editor, Primary Keys Editor, and Unique Keys Editor offer the same steps in modifying these PostgreSQL objects.

To edit an exclusion constraint, primary key, or unique key

1. Open an editor on the target object. 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 or modify properties defined when creating the object. For details on these properties, see Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL).
Columns	Lets you work with the columns designated for this object. For details, see Exclusion Constraints, Primary Keys, and Unique Keys Wizards (PostgreSQL).
Dependencies	For details on using this tab, see Working with Object Dependencies.
Comment	For details on using this tab, see Adding a Comment to an object.
Statistics	Displays the following pg_stat_all_indexes or pg_statio_all_table view statistics: **Scans**, **Keys Read**, **Rows Fetched**, **Blocks Read**, **Buffer Hits**, and **Index Size**.
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 (PostgreSQL)

The Foreign Keys Editor lets you view or modify column mapping and properties 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	Lets you work with the columns of the main table and mapped referenced table columns for this foreign key. For details, see Foreign Keys Wizard (PostgreSQL).
Properties	Lets you view or modify properties defined when creating the object. For details on these properties, see Foreign Keys Wizard (PostgreSQL).
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.

Functions Editor (PostgreSQL)

The Functions Editor lets you view or modify the function body, parameters, and properties 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 modify the settings defined when the function was created or last edited. For details on these properties, see Functions Wizard (PostgreSQL).
Parameters	Lets you modify the function inputs and outputs defined when the function was created or last edited. For details on these settings on this tab, see Functions Wizard (PostgreSQL).
Body	Lets you modify the function definition specified when the function was created or last edited. For details on working with this tab, see Functions Wizard (PostgreSQL).
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.
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 (PostgreSQL)

The Indexes Editor lets you view or modify the columns and 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
Properties	Lets you view or modify properties defined when creating the object. For details on these properties, see Indexes Wizard (PostgreSQL).
Columns	Lets you modify the columns that make up the index. For details, see Indexes Wizard (PostgreSQL).
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.

Roles Editor (PostgreSQL)

The Roles Editor lets you manage properties and users for a 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	Lets you modify the properties set for the role when it was created or last edited. For details on these properties, see Roles Wizard (PostgreSQL).
Membership	Lets you modify the set of users GRANTed to the role. For details, see Roles Wizard (PostgreSQL).
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.

Rules Editor (PostgreSQL)

The Rules Editor lets you view or modify rules properties.

To edit a rule

1. Open an editor on the rules. 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 or modify properties defined when creating the object. For details on these properties, see Rules Wizard (PostgreSQL).
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.

Schemas Editor (PostgreSQL)

The Schemas Editor lets you change the owner of a schema, manage its permissions, and view referencing and referenced dependencies.

To edit a schema

1. Open an editor on the schema. 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.
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.
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 (PostgreSQL)

The Tables Editor lets you manage settings for an existing 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
Properties	In addition to displaying the identification properties for this table, this tab lets you view the **Persistence** setting. You can modify the **Tablespace**, **With OIDS**, **Fill Factor**, **Table Autovacuum**, and **Toast Table Autovacuum** settings. For details on these properties, see Tables Wizard (PostgreSQL).
Ancestors	This tab/panel lets you modify a table's INHERITS clause.. For each table to be added to the list, use the New button to enable a dropdown that lets you select a table. Use the Delete button to remove a selected table from the list.
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. For details on working with this tab, see Tables Wizard (PostgreSQL). You can add a column by clicking **Add Column**, providing a name for the new column, and set its property values. Similarly, select a column and click **Delete** to remove the column from the table. Lastly, change a columns position by selecting the column and using the arrow buttons.
Dependencies	For details on using this tab, see Working with Object Dependencies.
Constraints	Lets you manage constraints for the table. Constraints are grouped by type, under folders. Optionally take one of the following actions: To add a constraints, select a constraint type folder and click **Add** to open a dialog that lets you add a constraint of that type. To edit a constraint, select a constraint and click **Edit** to open a dialog that lets you modify the constraint details. To delete a constraint, 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.
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 (PostgreSQL)

The Tablespaces Editor lets you manage properties and permissions as well as view dependencies 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 view or modify settings specified when the tablespace was created or most recently edited. For details on these properties, see Tablespaces Wizard (PostgreSQL).
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.
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 (PostgreSQL)

The Triggers Editor lets you manage properties and targeted columns 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 or modify settings specified when the trigger was created or most recently edited. For details on these properties, see Triggers Wizard (PostgreSQL).
Column Selection	Lets you view or modify columns targeted when the trigger was created or most recently edited. For details on these settings on this tab, see Triggers Wizard (PostgreSQL).
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.

Types Editor (PostgreSQL)

The Types Editor lets you view or modify type properties.

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
Properties	Lets you view or modify properties defined when creating the object. For details on these properties, see Types Wizard (PostgreSQL).
Range Kind	Lets you view or modify the Range Kind defined when creating the object. For details on these properties, see Types Wizard (PostgreSQL).
Base Kind	Lets you view or modify the Base Kind defined when creating the object. For details on these properties, see Types Wizard (PostgreSQL).
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.

Views Editor (PostgreSQL)

The Views Editor lets you manage settings for an existing 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	Lets you modify the settings defined when the function was created or last edited. For details on these properties, see Views Wizard (PostgreSQL).
Definition	Lets you modify the DDL originally generated to create the view.
Dependencies	For details on using this tab, see Working with Object Dependencies.
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.
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.

Sybase ASE Object Editors

An Object Editor is offered 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 ASE)
- Check Constraints Editor (Sybase ASE)
- Databases Editor (Sybase ASE)
- Defaults Editor (Sybase ASE)
- Extended Procedures Editor (Sybase ASE)
- Foreign Keys Editor (Sybase ASE)
- Functions Editor (Sybase ASE)
- Groups Editor (Sybase ASE)
- Indexes Editor (Sybase ASE)
- Logins Editor (Sybase ASE)
- Primary Keys Editor (Sybase ASE)
- Procedures Editor (Sybase ASE)
- Rules Editor (Sybase ASE)
- Segments Editor (Sybase ASE)
- Tables Editor (Sybase ASE)
- Triggers Editor (Sybase ASE)
- Unique Keys Editor (Sybase ASE)
- User Datatypes Editor (Sybase ASE)
- User Messages Editor (Sybase ASE)
- Users Editor (Sybase ASE)
- Views Editor (Sybase ASE)

For access to object editors for different supported DBMS platforms and an introduction to editor usage, see Modifying objects using editors.

Aliases Editor (Sybase ASE)

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 ASE)

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 ASE)

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	For information on working with these settings, see Databases (Sybase ASE) - 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 ASE)

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 ASE)

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 Previewing and Submitting Object Editor Changes.

Foreign Keys Editor (Sybase ASE)

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 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 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 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 (Sybase ASE)

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 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 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 Hide Text.
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.

Groups Editor (Sybase ASE)

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 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
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 Working with Privileges and Permissions.
System Privileges	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.

Indexes Editor (Sybase ASE)

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 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.
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 Viewing the SQL/DDL for an Object.	

3. When finished, you can submit your changes. For details, see Previewing and Submitting Object Editor Changes.

Logins Editor (Sybase ASE)

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 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 set when creating the login: **Name**, **FullName**, **Default database**, **Default language**, **Locked** (or **Account Locked**), **Password**, **Password expiration**, **Minimum password length**, **Maximum Login attempts**, and **Authentication Mechanism**. Against Sybase 15.7 and higher, servers, **Profile** (with an additional **default** option available) and **Exempt Inactive Lock** parameters are available. For detailed information on these settings, see Logins Wizard (Sybase ASE). 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 Add/Modify Login Trigger and Drop Login Trigger.
Users	Lets you maintain database user accounts for a login. For more detailed information, see Logins (Sybase ASE) - Users.
Roles	Lets you maintain the roles granted to this login. For more detailed information, see Logins (Sybase ASE) - Users. For information on activating and deactivating roles for the current login in the current session, see Role Activation.
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 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 (Sybase ASE)

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 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 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 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 (Sybase ASE)

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 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 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 Hide Text.
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.

Rules Editor (Sybase ASE)

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 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 a [JavaScript:popup.TextPopup(Poptext1,Popfont,9,9,-1,-1) **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 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.

Segments Editor (Sybase ASE)

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
Properties	This tab lets you view the **Name** and **Max Size** settings you specified when creating the segment. It also lets you change the **Grow By** and **Threshold Free Space** settings. For information on these settings, see Segments (Sybase ASE) - Properties. This tab also displays a **Hysteresis** value, the global variable `@@thresh_hysteresis`, that controls threshold sensitivity to variations in free space.
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. For information on specific editors, see Sybase ASE Object Editors.
Space	Displays a summary of segment space distribution, broken down into data, indexes, unused, and other. Also provides reserved, data, indexes, and unused space values for each object on the segment.
Threshold	Displays a list of the free pages thresholds for the segment. For a selected threshold, the **Property/Value** list displays the associated procedure name. Optionally you can: Click the **Add** button to specify the number of free pages and procedure name for a new segment threshold. Select an existing threshold and modify the threshold's procedure name value. Select an existing threshold and click the **Delete** 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 ASE)

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.
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 Partitioning Sybase ASE Tables.
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.

Partitioning Sybase ASE 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 Opening an Object Editor.
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.
Partitions Definitions (only available for a **Partition Type** of HASH)	Lets you specify a Partition method: **NoneNumber 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 ASE)

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 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 basic identifying properties of the trigger and its parent view or table: **Parent Type**, **Parent Owner**, **Parent Name**, **Owner**, **Name**, **Create Date**, **Fire On Insert**, **Fire On Update**, and **Fire On Delete**. For details on these properties, see Triggers (Sybase ASE) - Properties. 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, the trigger must be dropped then recreated. 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 Hide Text.
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 (Sybase ASE)

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 ASE)

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 ASE)

The User Messages Editor lets you manage user messages and their respective object bindings.

Note: On Sybase ASE, 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 ASE) - 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 ASE) - 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 ASE)

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 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 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 Working with Privileges and Permissions.
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.

Views Editor (Sybase ASE)

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 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	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 Hide Text.
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.

Object actions

In addition to letting you create and perform basic editing on various object types, there are a number of actions 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 Navigator, 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 in Rapid SQL

1. Connect to the datasource that contains the object you want to perform the operation against. For more information, see Connecting to Datasources.
2. On the Database Navigator, expand nodes until the target object type is visible and then expand the target object type. See the following topics for more information:
    - The Rapid SQL Datasource Navigator - for general information on working with the Navigator.
    - Supported Objects, specific object type topic - for information on navigating Datasource Navigator nodes to the target object type.
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.

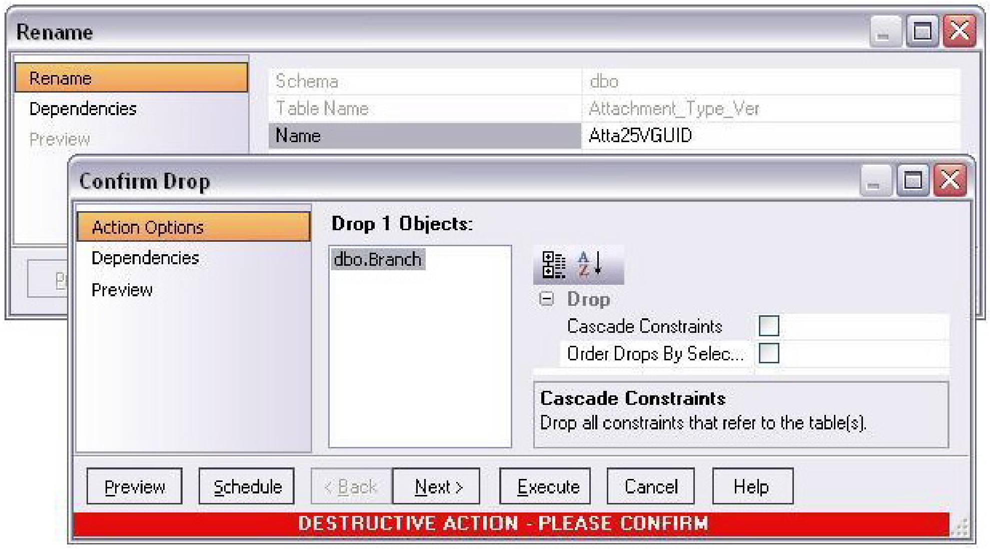
Note: Not all object operations can be applied to multiple objects.

The wizard for the selected object operation opens.

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:



Schema	dbo
Table Name	Attachment_Type_Ver
Name	Atta25VGUID

dbo.Branch

Drop
Cascade Constraints ☐
Order Drops By Selec... ☐

Cascade Constraints
Drop all constraints that refer to the table(s).

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 more 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. No attempt is made 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.
SQL	Opens an SQL Editor on the SQL contained in the Preview dialog. For more information, see Using the SQL 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 available object actions by DBMS platform.

	DB2 LUW	DB2 z/OS	IITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ

Add or Modify Check Constraint	✓	✓	✓		✓		✓	✓
Add/Modify Login Trigger								✓
Add Private Key							✓	
Allocate Extent					✓		✓	
Analyze					✓		✓	
Analyze Tables				✓				
Attach Database							✓	
Backup Certificate							✓	
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							✓		
Disable Index			✓				✓		
Disable Job					✓				
Disable Keys				✓					
Disable/Enable Triggers							✓	✓	
Drop	✓	✓	✓	✓	✓		✓	✓	✓
Drop Automatic Storage Path(s)	✓								
Drop By Category					✓				
Drop Clone		✓							
Drop Java					✓				
Drop Login Trigger								✓	
Drop Materialized Query Table	✓								
Drop Unused					✓				
Drop Unused					✓				
Backup Certificate							✓		
Enable Job (Job Queue)					✓				
Enable Keys				✓					
Enable Recycle Bin					✓				

Enable/Disable (Oracle Jobs)					✓			
Estimate Size							✓	
Exchange Data With Clone		✓						
Execute			✓		✓		✓	
Extract	✓	✓	✓	✓	✓	✓	✓	✓
Extract Data as XML					✓	✓		
Flashback Recycle Bin Entry					✓			
Flashback Table					✓			
Flush Cache	✓							
Flush Tables				✓				
Free (Packages)		✓						
Free Plan		✓						
Generate Package/Procedure/Statement	✓	✓	✓		✓	✓	✓	
Hide Text							✓	
Import Data From File	✓	✓	✓	✓	✓	✓	✓	
Load Java					✓			
Lock	✓	✓						
Lower High Water Mark	✓							
Move Log							✓	
Move Table	✓							
Next Used Filegroup						✓		
Object Properties								✓
Optimize Tables				✓				
Place						✓	✓	
Population status						✓		
Purge Recycle Bin					✓			

Purge Recycle Bin Entry					✓			
Quiesce	✓							
Reassign By Category					✓			
Rebalance	✓							
Rebind Packages	✓	✓						
Rebind Plans	✓	✓						
Rebuild (Full-text Catalogs)							✓	
Rebuild Index		✓	✓		✓		✓	
Rebuild Outlines					✓			
Rebuild Table				✓				
Recompile							✓	✓
Refresh Materialized View						✓		
Refresh Table	✓							
Rename	✓	✓	✓	✓	✓		✓	✓
Reorganize /Reorg	✓	✓			✓		✓	✓
Repair Tables				✓				
Report	✓	✓	✓	✓	✓		✓	✓
Restart (Object Action)	✓							
Restart Sequence(s)							✓	
Resynchronize		✓						
Run Job (Oracle Jobs)					✓			
Run Job (job queues)	✓							
Role Activation					✓			✓
Run Job (job queues)					✓			
Schema (Object Action)	✓	✓	✓	✓	✓		✓	✓
Select * From	✓	✓	✓	✓	✓		✓	✓

Set Default					✓		
Set Integrity	✓	✓					
Set Online/Offline							✓
Set Statistics			✓				
Set UNDO					✓		
Shrink					✓	✓	
Start Database		✓					
Stop Database		✓					
Stop Job					✓		
Switch Online	✓						
Transfer Ownership	✓						
Truncate	✓	✓		✓	✓	✓	✓
Unbind From Temporary Database							✓
Update Statistics	✓					✓	✓

Add or Modify Check Constraint

This action opens a Check Constraints Editor or Check Constraints Wizard.

Platform Availability

- o DB2 LUW, DB2 z/OS,

  ITB/FBD *,

  MySQL, ORCL, SQL SVR, SYB ASE

To Open the Check Constraint Editor or Check Constraint Wizard

1. On the Navigator, right-click the **Check Constraints** node and select **Create**.
2. Use the following table as a guide to creating or editing the constraint. Note that settings available varies by DBMS type.

Step	Settings and tasks	
Definition	**Table Owner** or **Table Schema**	Select the owner of the table for which the check constraint is to be created.
	Table or **Table Name**	Lets you select where you want to place the check constraint.
	Name	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**	Enables or disables the check constraint.
	Not for Replication	Enabling this feature ensures that the constraint is not enforced if the table is replicated for another database.
	Check Condition	Type the check constraint condition. As a convenience, use the **Table Columns** button to paste in column names as part of the condition.
Comment	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** or ALTER LOGIN 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

Platform Availability

- SYB ASE

To Specify the Stored Procedure Used as a Login Trigger for a Login:

1. If using Rapid SQL, 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. The **Add/Modify Login Trigger** dialog opens.

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 ASE).

Related Information

- Drop Login Trigger

Add Private Key

The **Add Private Key** object action builds and submits an ALTER CERTIFICATE... WITH PRIVATE KEY statement, optionally including a DECRYPTION BY PASSWORD argument. This lets you change the private key for a certificate, add a private key to a certificate that currently does not have one, and optionally, specify a decryption password.

Note: This functionality is available for Microsoft SQL Server only.

To change a certificate's private key

1. Initiate an **Add Private Key** action against a certificate. For more information see Initiating an object operation.

The **Add Private Key** dialog box opens.

2. Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks	
Action options	Lets you work with the following settings:	
	Private Key File	The full path and filename (local or UNC) of the file to which the private key is to be written.
	Decryption Password	The password required to decrypt the key.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see Dependencies.	
Preview	Displays the DDL that will execute the object action. For details, see Preview.	

3. Finally, use the **Execute** button to create the object.

Topics

- Backup Certificate.

Allocate Extent

The Allocate Extent dialog 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.

Platform Availability

- ORCL

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 Initiating an object operation.
2. Use the following table as a guide to setting the options on this dialog:

Option	Description
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 scheduling option, see Scheduling.

Analyze

This action lets you work with statistics for tables, indexes (primary keys and unique keys), and clusters.

Platform Availability

- ORCL

This action lets you build and submit the following ANALYZE statement variations or DBMS_STATS package procedure calls:

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 Accessing Third Party Documentation.

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 Initiating an object operation.
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 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 Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Analyze Tables

This action lets you build and submit an ANALYZE TABLE statement, analyzing and storing the key distribution for a table.

Platform Availability

- MySQL

To analyze a table:

1. Initiate an **Analyze Tables** 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 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see 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

Platform Availability

- SQL SVR

To Attach a Database

1. Initiate an **Attach Database** action against a database. For more information see Initiating an object operation.

The **Attach Database File(s)** dialog opens.

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.

The appropriate *.ldf file is automatically added.

4. To drop database file(s), click **Drop** and then select the target file(s).
5. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Detach Database

Backup Certificate

The **Backup Certificate** object action builds and submits a BACKUP CERTIFICATE... TO FILE statement with optionally, a WITH PRIVATE KEY clause. This lets you back up a certificate to file.

Platform Availability

- SQL SVR

To write a certificate to disk

1. Initiate a **Backup Certificate** action against a certificate. For more information see Initiating an object operation.

The **Attach Database File(s)** dialog box opens.

2. Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks	
Action options	Lets you work with the following settings:	
	File	The full path and filename (local or UNC) of the file to which the certificate is to be written.
	Private Key File	The full path and filename (local or UNC) of the file to which the private key is to be written.
	Encryption Password	The password used to encrypt the private key before writing the key to the backup file.
	Decryption Password	The password used to decrypt the private key before backing up the key.
Dependencies	Lets you review the objects potentially impacted by this action. 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. Finally, use the **Execute** button to create the object.

Topics

- Add Private Key

Bind Package

The Bind Package Wizard lets you set package parameters, add environments, and set package properties.

Platform Availability

- DB2 z/OS

To Package Parameters, Add Environments, and Set Package Properties

1. Expand an owning schema, right-click **Packages**, and select **Bind** from the context menu.

The **Bind Package** wizard opens.

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.	
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.

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.

Platform Availability

- SYB ASE

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 Initiating an object operation.
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 Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

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**.

Platform Availability

- DB2 z/OS

To Build a Procedure

1. Initiate a **Build** action against a procedure. For more information see Initiating an object operation.
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 Query Builder.

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: In Rapid SQL, the Browser listing for an Oracle table has a **Read-Only** field that displays the current status for a table. For more information, see Browsers.

Platform Availability

- ORCL 11g

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 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 **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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Change Category

You can change the category of a target Stored Outline.

Platform Availability

- ORCL

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. For more information, see Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Drop By Category
- Reassign By Category

Change Owner

This action lets you generate and submit an **ALTER***objecttype objectname***OWNER TO***newobjectname* statement, for one or more selected objects of the following types:

Domains	Functions	Schema	Tables
Tablespaces	Types	Views	

This lets you specify a new owner for the object.

Platform Availability

- PSTGRS *

To change the owner of an object

1. Initiate a **Change Owner** action against one or more objects of an object type shown in 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 **Change Category** wizard:

Step	Settings and tasks
Action options	From the **New Owner** dropdown, choose a new owner for the object.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Change Schema

Change Password

The **Change Password** dialog lets you change user or login passwords, which you should do on a regular basis to guard against security leaks.

Platform Availability

- MySQL, ORCL, SQL SVR, SYB ASE

To Change the Password for a User or Login

1. Take one of the following actions:

   - In Rapid SQL, for Sybase ASE or SQL Server, expand the Logins node.
   - In Rapid SQL, for MySQL and Oracle, expand the Users node.

2. Right-click the user or login, and then select **Change Password** from the context menu.

The **Change Password** dialog opens.

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 scheduling option, see Scheduling.

Change Schema

This action lets you generate and submit an **ALTER***objecttype objectname***SET SCHEMA***newobjectname* statement, for one or more selected objects of the following types:

Domains Functions Tables Types Views

This lets you specify a new schema for the object.

Platform Availability

- PSTGRS *

To change the schema for an object

1. Initiate a **Change Schema** action against one or more objects of an object type shown in 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 **Change Category** wizard:

Step	Settings and tasks
Action options	From the **New Owner** dropdown, choose a new owner for the object.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Change Owner

Change Status

You can change the status of a number of relevant object types. For details, see the following topics:

- Change Status (Rapid SQL - InterBase Triggers)
- Change Status (Oracle, SQL Server, and Sybase ASE triggers)
- Change Status (SQL Server DDL triggers)
- Change Status (Full-text Indexes)
- Change Status (check constraints)
- Change Status (tablespaces)

Change Status (Rapid SQL - 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.

To Change the Status of a Trigger:

1. Initiate a **Change Status** action against one or more triggerss. 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	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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Change Status (Oracle, SQL Server, and Sybase ASE triggers)

The **Change Status** object action lets you build an issue an ALTER TABLE statement that specifies a DISABLE (DISABLE TRIGGER) or ENABLE (ENABLE TRIGGER) option. This lets you enable or disable any triggers.

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 scheduling option, see Scheduling.

Change Status (SQL Server DDL triggers)

In Rapid SQL, 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: The Rapid SQL 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.
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 scheduling option, see 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.

Platform Availability

- SQL SVR

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Change Status (check constraints)

You can 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 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, 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		Settings and tasks
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. For more information, see Dependencies.	
Preview	Preview the DDL generated from your choices. For more information, see Preview.	

3. Use the **Execute** or **Schedule** button to complete the operation.

Setting Constraint Status for Oracle Objects

The **Set Constraint(s) Status** dialog lets you change the status of check constraints, foreign key constraints, primary key constraints, and unique key constraints. You can 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

The table below describes the options and functionality on the **Set Constraint(s) Status** dialog.

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. Initiate a **Change Status** action against one or more check constraints. For more information see Initiating an object operation.
2. Use the table above to select dialog box options.
3. 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.

Platform Availability

- ORCL

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

Check Tables

This action 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.

Platform Availability

- MySQL

To Check a Table for Errors

1. Initiate a **Check Tables** 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 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. For more information, see Preview.	

3. Click **Execute**. For information on the scheduling option, see Scheduling.

The results set opens on the **Results** tab of an ISQL editor window. For more information, see Using the 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.

Platform Availability

- SQL SVR, SYB ASE

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

Step	Description
Action options	Displays the names of the database(s) you chose.
Preview	Displays the DDL generated to execute the Checkpoint operation. For more information, see Preview.

3. Use one of the **Schedule** or **Execute** buttons to execute the Checkpoint.

Checksum Tables

This action lets you issue a CHECKSUM TABLE statement, returning a live checksum.

Platform Availability

- MySQL

To Check a Table for Errors:

1. Initiate a **Checksum Tables** 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 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. For more information, see Preview.	

3. Click **Execute**. For information on the scheduling option, see Scheduling.

The results set opens on the **Results** tab of an ISQL editor window. For more information, see Using the 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, the tablespace is locked.

Platform Availability

- ORCL

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

Compile

You can 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.

The **Compile** wizard opens. 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. For more information, see Dependencies.		
Preview	Displays the DDL generated to execute the Checkpoint operation. For more information, see Preview.		

3. Click **Execute**. For information on the scheduling option, see 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

You can 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. An 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.

Notes on Compiling Oracle Java Sources

Oracle lets you compile a Java source. Oracle resolves references to other Java sources.

Notes on Compiling Oracle Materialized Views

You can 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.

Notes on Compiling Oracle Packages and Package Bodies

You can recompile a package and recompile all package objects together.

Recompiling a package compiles both the package specification and the package body by issuing two consecutive ALTER statements. However, only the ALTER statement is shown for the specification in the **Preview: Confirm Compile** dialog. 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.

Notes on Compiling Oracle Procedures

You can compile a procedure that is part of a package, by compiling the package itself. The ALTER PROCEDURE statement is used 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.

Notes on Compiling Oracle Types and Type Bodies

You can recompile a type. Both the type specification and the type body are recompiled.

Notes on Compiling Oracle Views

You can recompile a view when you have altered an object referenced by that view.

When you recompile a view, Oracle invalidates all dependent objects.

Convert Tables

This action lets you build and submit an ALTER TABLE statement, specifying an ENGINE= option, specifying any available storage engine.

Note: For information on how to quickly convert a table to use the INNODB storage engine, see Rebuild Table.

Platform Availability

- MySQL

To Change the Storage Engine for a Table

1. Initiate a **Convert Tables** 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 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Copy Object Names

The Copy Object Names functionality lets you copy and then paste object name(s) into other applications.

Platform Availability

- ORCL, SQL SVR

To Copy an Object Name to be Pasted into Another Application

1. On the Rapid SQL Datasource Navigator, expand nodes until the target object type is visible and then expand the target object type.

2. Right-click on the specific database object or objects and select **Copy object name(s)**.

Copy Schema

This action lets you quickly make copies of a database schema and its corresponding objects.

Platform Availability

- o DB2 LUW

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. For more information, see Preview.

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	✓	✓	✓	✓
DB2 z/OS			✓	✓

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.

The Alias Wizard opens. For information on using the Alias editor, see the topic corresponding to the DBMS you are working against:

- Aliases Wizard (DB2 LUW)
- Aliases Wizard (DB2 Z/OS)

Create Clone

This action provides support for 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: The base table must reside on a universal tablespace. For more information, see Tablespaces Wizard (DB2 Z/OS).

Platform Availability

- DB2 z/OS version 9 ^

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

The clone for a table is not displayed in the Datasource Navigator 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.

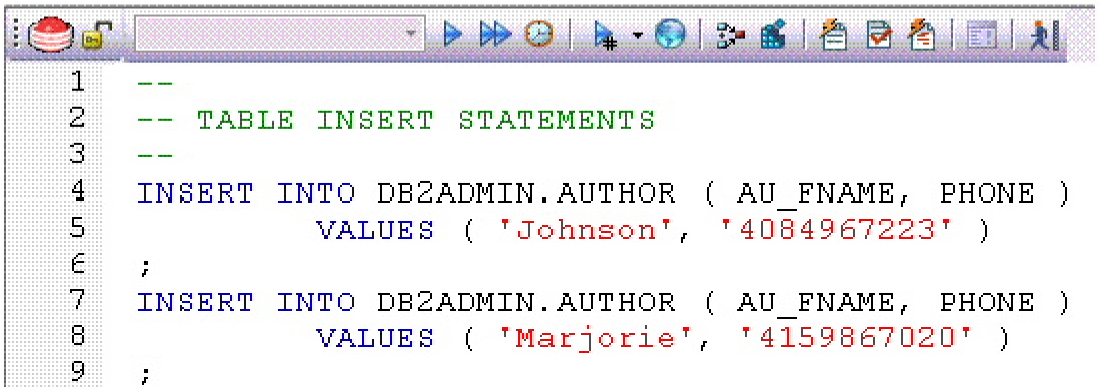
After creating a clone for a table, you can exchange data between the base table and the clone.

Related Information

- 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.



```
--
-- TABLE INSERT STATEMENTS
--
INSERT INTO DB2ADMIN.AUTHOR ( AU_FNAME, PHONE )
        VALUES ( 'Johnson', '4084967223' )
;
INSERT INTO DB2ADMIN.AUTHOR ( AU_FNAME, PHONE )
        VALUES ( 'Marjorie', '4159867020' )
;
```

Platform Availability

- DB2 LUW
- DB2 z/OS
- ITB/FBD *
- MySQL
- ORCL
- SQL SVR
- SYB ASE

To Create Insert Statements Based on Existing Table Data

1. In the left pane of the application, select the Tables node.
2. In the right pane of the application, right-click a table, and then select **Create Insert Statements**.

The **Create Insert Statements** dialog opens.

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 Using the SQL Editor.

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 ASE
Materialized Query Tables Tables	Tables	Tables	Tables	Logins Tables Users	Tables	Logins Tables Users

To Create an Object Bbased on Another Object

1. Initiate a **Create Like** action against a table. For more information see Initiating an object operation.

A Create Like wizard opens.

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)

o Tables editor (InterBase/Firebird)

o Tables editor (MySQL)

o Logins Editor (SQL Server)

o Tables Editor (SQL Server)

o Users Editor (SQL Server)

o Tables Editor (Oracle)

o Logins Editor (Sybase ASE)

o Tables Editor (Sybase ASE)

o Users Editor (Sybase ASE)

4. Click **Execute** to create the new object.

Create or Edit Java Source

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, 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.

Platform Availability

- ORCL

Create Synonym

You can perform an ad hoc Create Synonym operation against a specific object displayed in the Datasource Navigator. For background information, see Synonyms.

Platform Availability

- ORCL

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 Synonyms Wizard (Oracle).

Create View

Right-clicking one or more tables in the Navigator and selecting **Create View** opens a wizard that lets you create a new view. For more information, see the following topics:

- Views Wizard (DB2 LUW)
- Views Wizard (DB2 Z/OS)
- Views Wizard (ITB/FBD)
- Views Wizard (SQL Server)
- Views Wizard (Oracle)
- Views Wizard (Sybase ASE)

DBCC

The 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.

Platform Availability

- SQL SVR, SYB ASE

To Perform a Database Consistency Check

1. Initiate a **DBCC** action against a database, table, or index, as follows:
   - If you are checking consistency of a Sybase ASE 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 ASE 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 ASE 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**, or **DBCC > Show Statistics** from the context menu, as appropriate.

For more information see Initiating an object operation.

A dialog that that lets you specify DBCC options opens.

2. 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

3. Click **Execute**. For information on the scheduling option, see Scheduling.

DBCC for Microsoft SQL Server

The **DBCC (Database Consistency Check**) dialog 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 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

DBCC for Microsoft SQL Server Databases

The **DBCC** dialog 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 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.

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).

DBCC for Microsoft SQL Server Tables

The **DBCC** dialog for tables lets you perform table-level validations of databases. The fundamental difference between the **DBCC** dialog 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.

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.
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.
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.

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.

DBCC for Microsoft SQL Server Indexes

The following Database Console Commands are supported, letting you perform index-level validations of databases.

Option	Description
Check Fragmentation	Lets you build and submit a DBCC SHOWCONTIG command, to display fragmentation information for the selected index, letting you determine 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. For details, see Rebuild Index (SQL Server).
Update Usage	Lets you build and submit a DBCC UPDATEUSAGE command, reporting on and correcting pages and row count inaccuracies that can result in incorrect space usage reports. The **Count Rows** control lets you add a WITH COUNT_ROWS argument, dictating that the row count column be corrected as necessary. The **No Informational Messages** control lets you add a WITH NO_INFOMSGS argument, suppressing all informational messages.
Check Index	Lets you build and submit a DBCC CHECKTABLE command, verifying the integrity of pages and structures of the table or indexed view. Disabling the **Show Informational Messages** control adds a WITH NO_INFOMSGS argument, suppressing informational messages.
Show Statistics	Lets you build and submit a DBCC SHOW_STATISTICS command, displaying current query optimization statistics for the owning table or indexed view.

DBCC for Sybase ASE

The **DBCC** (Database Consistency Check) dialog 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 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

DBCC for Sybase ASE Databases

The **DBCC** dialog for databases lets you perform database-wide validations. The fundamental difference between the **DBCC** dialog 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.

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.

DBCC for Sybase ASE Tables

The **DBCC** dialog for tables lets you perform table-level validations of databases. The fundamental difference between the **DBCC** dialog 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.

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

Additional options are offered 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 have any allocation errors fixed. 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.

DBCC for Sybase ASE Indexes

The **DBCC** dialog for indexes lets you perform index-level validations of databases. Unlike the **DBCC** dialog for tables, this **DBCC** dialog 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.

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	Any allocation errors detected are fixed. 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.

Deallocate Unused Space

The **Deallocate Unused Space** dialog 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 editor.

Platform Availability

- ORCL

To Deallocate Unused Space from an Index or Table

1. Initiate a **Deallocate Unused Space** action against one or more Clusters, Indexes, Primary Keys, Tables, or Unique Keys. For more information see Initiating an object operation.

The **Deallocate Unused Space** dialog opens.

2. 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.

3. Click **Execute**. For information on the scheduling option, see 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.

Platform Availability

- SYB ASE

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 ASE 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 operationn. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Describe

RIght-clicking a table, view, procedure or function in the Navigator and selecting **Describe** opens a window that lets you view columnar information (for tables and views) or input parameter information (for procedures and functions).

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.

Platform Availability

- SQL SVR

To Detach Data and Transaction Files for a Database

1. Initiate a **Detach Database** action against a database. For more information see Initiating an object operation.

The **Detach Database(s)** dialog opens.

2. 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.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Related Information

- Attach Database

Disable Index

The **Disable Index(es)** action lets you mark an index as disabled or inactive and unavailable for use.

Platform Availability

- ITB/FBD *,

  SQL SVR

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Disable Job

You can 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.

Platform Availability

- ORCL

To Disable a Job Queue

1. Initiate a **Disable Job** action against a job queue. For more information see Initiating an object operation.

Tthe **Disable** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Enable Job (Job Queue)
- Run Job (job queues).

Disable Keys

This action lets you build and submit an ALTER TABLE statement that specifies the DISABLE KEYS option. This action tells MySQL stop updating nonunique indexes.

Platform Availability

- MySQL

To Stop Updates to Nonunique Indexes

1. Initiate a **Disable Keys** action against one or more tables. For more information see Initiating an object operation.

The **Disable Keys** dialog opens.

2. 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Note: You can subsequently recreate missing indexes. For details, see Enable Keys.

Disable/Enable Triggers

These actions let you build and submit ALTER TABLE statements that specify ENABLE TRIGGER or DISABLE TRIGGER options.

Platform Availability

- SQL SVR, SYB ASE

To Enable or Disable Triggers for a Table

1. On the Database Navigator, select the **Tables** node.
2. On the Database Navigator, right-click one or more selected tables and select **Enable Triggers** or **Disable Triggers**.

The **Enable Triggers** or **Disable Triggers** dialog opens, as appropriate.

3. Click **Execute**.

Drop

The **Confirm Drop** dialog lets you drop one or more database objects and remove their definition from the system catalog.

Platform Availability

- DB2 LUW, DB2 z/OS,

  ITB/FBD *,

  MySQL, ORCL, SQL SVR, SYB ASE, SYB IQ

To Drop an Object

1. Initiate a **Drop** action against one or more objects. For more information see Initiating an object operation.

The **Confirm Drop** dialog opens on the **Action Options** panel.

2. On the **Action Options** panel, select any options specific to the object type or configuration of the object you are dropping.
3. Navigate to the **Dependencies** pane to view database objects that are dependent on the object you are dropping.

Note: For Microsoft SQL Server, 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.

4. Navigate to the **Preview** pane to verify that the correct DDL was created, and if necessary navigate backward and modify your choices.
5. Click **Execute** to drop the object.

Drop Automatic Storage Path(s)

This action lets you remove tablespace storage paths on a datasource and subsequently rebalance any affected tablespaces. It builds and submits:

- An ALTER DATABASE DROP STORAGE ON... statement specifying the storage paths to be removed
- Optionally, one ALTER TABLESPACE... REBALANCE statement for each tablesapce impacted by the ALTER DATABASE DROP STORAGE ON... statement

Platform Availability

- DB2 LUW 9.7^

Prerequisites or Setup Tasks

This object action is only available for datasources configured for autostorage and that have at least two autostorage paths configured.

To drop storage paths on a datasource

1. Initiate a **Drop Automatic Storage Path(s)** action against a datasource. For more information see Initiating an object operation.
2. Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Path(s) Selection	**Path:** Select the check box associated with each storage path that is to be dropped. This object action will not allow you to select all storage paths on a datasource. At least one path must be left active. **Rebalance the affected tablespaces:** Selecting this check box generates an ALTER TABLESPACE... REBALANCE statement for each tablesapce affected by the storage paths you selected to drop.
Preview	Displays the DDL that will execute the object action. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

For related information, see Rebalance.

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.

Platform Availability

- ORCL

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Related Information

- 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.

Platform Availability

- DB2 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 scheduling option, see Scheduling.

Related Information

- Create Clone
- Exchange Data With Clone

Drop Java

This action lets you make use of the `dropjava` tool, deleting schema objects corresponding to Java files.

Platform Availability

- ORCL

To Drop Schema Objects Assocaited with a Java File

1. Initiate a **Drop Java** action against one or more Java Classes. For more information see Initiating an object operation.
2. Use the following table as a guide to understanding and modifying settings on the **Drop Jave Objects** dialog:

Setting	Description
Who Owns the Java Object	Select the owner of objects to be dropped.
File to be dropped	Use these controls to build a list of Java files for which associated objects are to be dropped.
Drop Synonym	Drops any synonyms defined for objects being dropped.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Load Java

Drop Login Trigger

The **Drop Login Trigger** action lets you build and submit an **sp_modifylogin** or ALTER LOGIN 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: 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 ASE).

Platform Availability

- SYB ASE

To Remove the Currently Configured Login Trigger for a Login

1. In Rapid SQL expand the **Logins** node.

2. Right-click a login and select Login Trigger Actions > Drop Login Trigger from the context menu. The **Drop Login Trigger** 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 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. For more information, see Preview.

Topics

- Add/Modify Login Trigger

Drop Materialized Query Table

The **Drop Materialized Query** action lets you convert a materialized query table to a regular table.

Platform Availability

- DB2 LUW

To Convert a Materialized Query Table to a Regular Table

1. Initiate a **Drop Materialized Query Table** action against one or more materialized query tables. For more information see Initiating an object operation.

The **Confirm MQT Conversion to Regular** dialog opens.

2. Click **Execute**. For information on the scheduling option, see 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.

Platform Availability

- ORCL

To Drop One or More Unused Stored Outlines

1. Initiate a **Drop Unused** action against one or more stored outlines. For more information see Initiating an object operation.

The **Drop Unused** dialog opens.

2. 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

This action lets you build and submit an ALTER TABLE... DROP UNUSED COLUMNS statement, letting you remove columns currently marked as unused.

Platform Availability

- ORCL

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 Initiating an object operation.
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 Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Edit Data

Right-clicking a table and selecting Edit Data lets you edit table data. For details, see Data Editor.

Enable/Disable (Oracle Jobs)

This action builds and submits a DBMS_SCHEDULER.DISABLE or DBMS_SCHEDULER.ENABLE call, letting you enable or disable a job or program.

Platform Availability

- ORCL 10g ^

To enable or disable a job or program

1. In Rapid SQL, on a connected Oracle datasource, expand the **Scheduler** nodes and select the **Jobs** or **Programs** node.

2. Right-click a job or program and select **Enable/Disable** from the context menu.

3. 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 job or program. Leave the **Enabled** check box deselected to disable the job or program.
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.

4. Click **Execute**. For information on the scheduling option, see Scheduling.

Enable/Disable Filetable(s)

This action lets you build and submit an ALTER TABLE statement, specifying either an ENABLE FILETABLE_NAMESPACE or DISABLE FILETABLE_NAMESPACE argument. This enables or disables system-defined constraints on a FileTable.

Platform Availability

- SQL SVR

To enable or disable system-defined constraints on a FileTable

1. Initiate an **Enable/Disable Filetables** action against one or more tables. For more information see Initiating an object operation.

The **Enable/Disable Filetable(s)** dialog box opens.

2. Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks	
Action options	**Enabled**	If selected, an ALTER TABLE with an ENABLE FILETABLE_NAMESPACE argument is generated. If unselected, an ALTER TABLE with a DISABLE FILETABLE_NAMESPACE argument is generated.
Dependencies	Lets you review the objects potentially impacted by this action. 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. Finally, use the **Execute** button to create the object.

For information on setup of filegroups, FileTable, and Filestream, see the following topics:

- Databases Wizard (SQL Server)
- Tables Wizard (SQL Server)

Enable/Disable Stored Outline Auto-creation/Auto-use

There are two object actions 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.

Platform Availability

- ORCL

To Specify Whether Stored Outlines are Automatically Created and Used

1. Initiate an **Enable/disable stored outline auto-use** or **Enable/disable stored outline auto-creation** 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 wizard:

Step	Settings and tasks	
Action Options	**Automatically create stored outlines** or **Automatically use stored outlines**	Lets you enable or disable the feature associated 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. For more information, see Preview.	

3. When ready, click **Execute**.

Enable Job (Job Queue)

This action 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.

Platform Availability

- ORCL

To Enable a Job Queue

1. Initiate an **Enable Job** action against a job queue. For more information see Initiating an object operation.

The **Enable** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Disable Job
- Run Job (job queues)

Enable Keys

You can issue an ALTER TABLE statement that specifies the ENABLE KEYS option. This action tells MySQL to recreate missing indexes.

Platform Availability

- MySQL

To Recreate Missing Indexes

1. Initiate an **Enable Keys** action against one or more tables. For more information see Initiating an object operation.

The **Enable Keys** dialog opens.

2. 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see 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.

To Enable or Disable the Recycle Bin

1. In Rapid SQL, 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 scheduling option, see Scheduling.

Topics

- Recycle Bin
- 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 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.

Platform Availability

- ORCL, SYB ASE

To Estimate Size for a Table or Index

1. Initiate an **Estimate Size** action against one or more tables or indexes. For more information see Initiating an object operation.

The **Estimate Size** dialog opens.

2. 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 ASE 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:

- o Use the **Report** button to generate an HTML report of results
- o Use the **Save As** button to save a result grid file
- o Use the **Owner** and **Name** controls to select another table or index for size estimates.
- o 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 ASE 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:

- o Use the **Report** button to generate an HTML report of results
- o Use the **Save As** button to save a result grid file
- o 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.

Platform Availability

- DB2 z/OS version 9 ^

To Exchange Data Between One or More Tables and their Clones

1. Initiate a **Clone Actions > Exchange Data With Clone** action against one or more tables. For more information see Initiating an object operation.

The **Exchange Data With Clone** dialog opens.

2. 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.
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 more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Create Clone
- Drop Clone

Execute

The **Execution** dialog 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 ASE
Extended procedures							✓
Functions					✓		
Procedures	✓	✓	✓		✓	✓	✓

To Compile an Extended Procedure, Function, or Procedure

1. In Rapid SQL, expand the node corresponding to valid object type. Objects of that type are displayed below the object type node.

2. Right click the target object and select **Execute** from the context menu.

Note: If the extended procedure, function, or procedure takes input parameters, you are prompted to provide values.

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 Using the Results Editor.

Working With Execution Parameters

If you initiate an **Execute** action against an extended procedure, function, or procedure that takes input parameters, a dialog prompts you to provide parameter values and otherwise manage parameters.

The **Procedure Execution** dialog 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:

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, 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.

Extract

You can extract the statements required to create existing objects into an Interactive SQL window.

Note: For information on setting preferences for this action, see DDL Extract Options.

Platform Availability

- DB2 LUW, DB2 z/OS,

  ITB/FBD *,

  MySQL, ORCL, SQL SVR, SYB ASE, SYB IQ

To Extract the DDL for an Object

1. Initiate an **Extract** action against one or more objects. For more information see Initiating an object operation.

The DDL Editor opens with the DDL to create the selected objects. For more information, see Using the DDL Editor.

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.

Caution: To use the **Extract Data as XML** feature in Sybase ASE, you must first purchase and install an XML Services license from Sybase.

Platform Availability

- ORCL 9i ^, SQL SVR, SYB ASE 12.5.1 ^

To Open the XML Editor Filter

1. Initiate an **Extract Data as XML** action against a table. For more information see Initiating an object operation.

An **XML Editor Filter** dialog opens.

2. In the **Columns** area, select the check boxes corresponding to the columns with data you want to extract.
3. 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.

4. 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 ASE	Refer to Sybase online documentation for details onXML mapping information. For assistance, see Accessing Third Party Documentation.	
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

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.

5. When ready, click **OK**. An XML representation of the extracted data opens 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: The Datasource Navigator 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

Platform Availability

- ORCL 10g^

To Recover a Table Currently in the Recycle Bin

1. In Rapid SQL, 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 Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

4. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- 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: 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.

Platform Availability

- ORCL 10g ^

Note: Available options differ by the version of Oracle that you are working against.

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.

Dependencies	For details on using this tab, see Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Flashback Recycle Bin Entry

Flush Cache

This action 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.

Platform Availability

- DB2 LUW

To Flush the Cache for a Package

1. Initiate a **Flush Cache** action against a package. For more information see Initiating an object operation.

The **Flush Package Cache** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

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.

Platform Availability

- MySQL

To Flush One or More Tables

1. Initiate a **Flush Tables** action against one or more stored tables. For more information see Initiating an object operation.
2. 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Free (Packages)

You can delete one or more packages.

Platform Availability

- DB2 z/OS

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Free Plan

In Rapid SQL, this action lets you drop plans associated with packages.

To Drop a Plan

- In Rapid SQL, 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

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.

Platform Availability

- DB2 z/OS, ITB/FBD *, ORCL, SQL SVR, SYB ASE

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 Code Generator dialog box, which is accessible from the application's **Tools** menu and Tools toolbar.

The **Select 1 or More Columns** dialog 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.

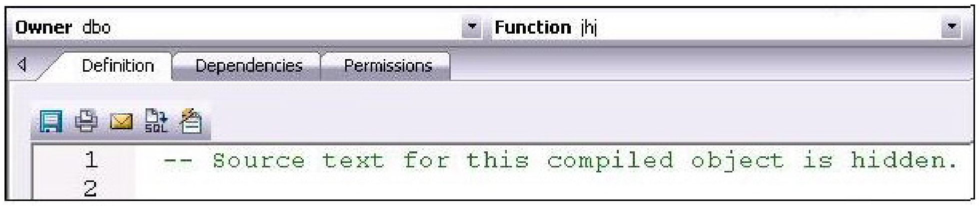
Hide Text

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:

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.

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.



```
-- Source text for this compiled object is hidden.
```

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 application-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.

Platform Availability

- SYB ASE

To Permanently Hide the Source Text for One or More Objects

1. For Rapid SQL in the Database Navigator, expand the node corresponding to one of the target object types. Objects of that type are shown below the object type node.

2. 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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

4. Click **Execute**. For information on the scheduling option, see Scheduling.

Rapid SQL 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 Navigator 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 only.

Note: For related information, see Drop Java.

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.

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 to add a new resolver option in the list. **Edit** - Click to open the **Resolver Edit** dialog to modify a resolver option. **Remove** - Select one or more resolver option and click to delete.

Lock

The **Lock Table** dialog 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. Locks are released at the end of a transaction.

Platform Availability

- DB2 LUW

To Lock a Table

1. Initiate a **Lock** action against one or more tables. For more information see Initiating an object operation.

The **Lock Table** dialog opens.

2. 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.
3. Click **Execute**. For information on the scheduling option, see Scheduling.

Lower High Water Mark

Against automatically-managed tablespaces, this action lets you build and submit an ALTER TABLESPACE... REDUCE statement, specifying that existing containers be reduced by a specified size.

Against database-managed tablespaces, you build and submit an ALTER TABLESPACE... LOWER HIGH WATER MARK statement followed by an ALTER TABLESPACE... REDUCE statement, explicitly lowering the high water mark before reducing the size of the tablespace.

Note: Before using this object action, consult DB2 LUW documentation for information on automatic and non-automatic storage, storage reclaimasbility, as well as database versus system-managed tablespaces. For more information, see Accessing Third Party Documentation.

Platform Availability

- DB2 LUW 9.7 ^

To Lower the High Water Mark for a Tablespace

1. Initiate a **Lower High Water Mark** action against a tablespace. For more information see Initiating an object operation.
2. Use the following table as a guide to understanding and choosing options in the **Lower High Water Mark** dialog:

Pane	Options
Action Options	Against automatically-managed tablespaces, this pane lets you provide the REDUCE option value for the generated ALTER TABLESPACE statement. Two options are provided. Selecting **MAX** specifies that the high water mark be lowered by moving the maximum number of extents to the beginning of the tablespace. Specifying a **Size** and **Unit** specifies the numeric value by which the tablespace is to be reduced through extent movement. Against database-managed tablespaces, this pane only allows you to specify that **All Containers** are to be decreased by a specified **Size** (in Pages, KB, MB, or GB **Unit**s). If you want to specify individual containers, use the **Containers** panel.
Containers	This tab is only available for database-managed tablespaces. As well, the functionality provided depends on whether you selected the **All Containers** option on the **Action Options** pane. With **All Containers** selected, this panel lets you specify an ON DBPRATITIONNUM clause on the ALTER TABLESPACE... REDUCE statement generated, specifying partitions for the container operations. Click the **Partitions** button, select one or more partitions on the **Select Partitions** dialog, and click **OK**. With **All Containers** deselected, this panel lets you specify an ON DBPRATITIONNUM clause and also lets you specify a database container clause on the ALTER TABLESPACE... REDUCE statement generated. For each container to be included in the databae container clause, click the Add Container button, select a container from the dropdown, and select a **Size** and **Unit** (Pages, KB, MB, or GB) for the container.
Preview	Displays the DDL generated to execute the operation. For more information, see Preview.

3. **Execute** or **Schedule** the action.

Move Log

You can move a transaction log from one device to another.

Platform Availability

- SYB ASE

To Move a Database Transaction Log to Another Device

1. Initiate a **Move Log** action against one or more databases. For more information see Initiating an object operation.

The **Move Log** dialog opens.

2. From the **New Device** dropdown, select the defined device to which you want to move the transaction log.
3. Click **Execute**. For information on the scheduling option, see Scheduling.

Move Table

The **Move Table** action lets you move data in an active table into a new, identically-named table, leaving the data online and available for use. This action lets you build and submit one or more SYSPROC.ADMIN_MOVE_TABLE calls. A single call, corresponding to the MOVE command, lets you execute all move operations in a single step. Multiple calls correspond to use of the INIT, COPY, REPLAY, VERIFY, SWAP, CLEANUP or CANCEL commands, and let you execute the move operations in individual steps.

Note: Before using this object action, consult DB2 LUW documentation for information on the ADMIN_MOVE_TABLE procedure, specifically the options and parameters used by the procedure. For more information, see Accessing Third Party Documentation.

Platform Availability

- DB2 LUW 9.7 ^

To move table data

1. Initiate a **Move Table** action against a table. For more information see Initiating an object operation.
2. Use the following table as a guide to understanding and choosing options the options in the dialog:

Pane		Categories and Option Descriptions
Action Options	Method	The **Use Target Table** option lets you select from the two methods available with use of the ADMIN_MOVE_TABLE procedure. Select the **Use Target Table** check box if you have already created the target tabl , and are using this action to move the data to that table. Leave the **Use Target Table** check box unselected to specify tablespaces used by the new table and have this action create the table before moving data to the new table.
	Tablespace	These options are only available with the **Use Target Table** check box deselected. They let you optionally specify new data, index, and LOB tablespaces for the target table.
	Target Table	This option is only available with the **Use Target Table** check box selected. It lets you specify an existing table to be used as the target table during the move.
	Operations	The **Generate...** options correspond to the ADMIN_MOVE_TABLE Operation input parameters. These can be used in one of two mutually exclusive ways. You can generate a single ADMIN_MOVE_TABLE call specifying MOVE to have all operations performed at one time. Alternatively, you can generate one or more ADMIN_MOVE_TABLE calls corresponding to each of the selected INIT, COPY, REPLAY, VERIFY, SWAP, CLEANUP, or CANCEL options, to have the table moved one step at a time.

	MOVE, INIT, COPY, REPLAY, VERIFY, SWAP, CLEANUP, and CANCEL Options	A set of options is displayed for each selected **Generate...** option. Depending on the **Generate...** selection, KEEP, COPY_USE_LOAD, COPY_WITH_INDEXES, FORCE, NO_STATS, COPY_STATS, NO_AUTO_REVAL, REORG, NO_TARGET_LOCKSIZE, CLUSTER, NON_CLUSTER, and LOAD_MSGPATH options may be available. Keep in mind that CLUSTER and NON_CLUSTER options are mutually exclusive and that LOAD_MSGPATH is only available when COPY_USE_LOAD is selected.
Preview	Displays the DDL generated to execute the operation. For more information, see Preview.	

3. Click the **Execute** or **Schedule** button.

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.

Platform Availability

- SQL SVR

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Object Properties

This action submits a `sp_iqhelp` procedure call against a selected object of one of the following types:

Domains	Events	Functions	Procedures
System Tables	Tables	Views	

Standard `sp_iqhelp` output, providing information about the object, is opened in a Results Editor.

Platform Availability

- o SYB IQ

To display sp_iqhelp information for an object

1. Initiate an **Object Properties** action against an object of one of the types in the table above. For more information see Initiating an object operation.

The output opens in a Results Editor.

Optimize Tables

You can issue an OPTIMIZE TABLE statement, reclaiming unused space and defragmenting the data file.

Platform Availability

- MySQL

To Optimize Table Storage

1. Initiate an **Optimize Tables** action against one or more tables. For more information see Initiating an object operation.

The **Optimize Tables** dialog opens.

2. 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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Place

The **Placement** dialog 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.

Platform Availability

- SYB 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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operationn. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see 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: 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.

Platform Availability

- SQL SVR

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see 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

Platform Availability

- ORCL 10g ^

To Purge Objects in the Recycle Bin Tables and Associated Objects

1. In Rapid SQL, 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 scheduling option, see Scheduling.

Topics

- 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: The Datasource Navigator 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

Platform Availability

- ORCL 10g

To Permanently Purge Selected Tables or Indexes Currently in the Recycle Bin

1. In Rapid SQL, 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. For more information, see Preview.

4. Click **Execute**. For information on the scheduling option, see Scheduling.

Note: When dropping tables on a datasource with the Recycle Bin feature enabled, the **Drop** action has an option that lets you immediately purge the table.

Topics

- Recycle Bin
- Enable Recycle Bin
- Flashback Recycle Bin Entry
- Purge Recycle Bin

Quiesce

You can 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.

Platform Availability

- DB2 LUW

Note:Quiesce Database is not supported for a DB2 8 server from an IBM DB2 for Linux, Unix, and Windows client or for an IBM DB2 for Linux, Unix, and Windows 7 server from an IBM DB2 for Linux, Unix, and Windows client.

To Quiesce a Database

1. On the Navigator, 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**.

Quiesce (Instance)

This action lets you restrict user access to an instance.

Platform Availability

- DB2 LUW

To Quiesce an Instance

1. On the Navigator, 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**.

Topics

- Unquiesce

Reassign By Category

This action lets you reassign the category of stored outlines in 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.

Platform Availability

- ORCL

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Change Category
- Drop By Category

Rebalance

This action lets you build and submit one or more ALTER TABLESPACE... REBALANCE statements. This creates containers on recently added storage paths and drops containers from storage paths that are in DROP PENDING state.

Platform Availability

- DB2 LUW 9.7^

Prerequisites or Setup Tasks

This object action is only available for tablespaces that meet the following criteria:

- The tablespace must use automatic storage. That is, in the Tablespace Editor, the **Use Automatic Storage** property is selected.For more information see Tablespaces Editor (IBM DB2 LUW).
- In the Tablespace Editor, the **Type** property specifies LARGE or REGULAR.

To rebalance a tablespace

1. Initiate a **Rebalance** action against one or more tablespaces.
2. Use the following table as a guide to understanding and modifying settings in this wizard:

Step	Settings and tasks
Action Options	Displays the tablespaces selected for this operation.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Drop Automatic Storage Path(s)

Rebind Packages

The **Rebind Package** dialog lets you update the best access path for SQL statements when the contents of a package changes.

Tip: If the physical storage of a package is changed or dropped, rebinding updates the path of the SQL statements.

Platform Availability

- DB2 LUW, DB2 z/OS

To Rebind One or More Packages

1. Initiate a **Rebind Packages** action against one or more packages. For more information see Initiating an object operation.

The **Rebinding Package** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

Rebind Plans

This action lets you update the best access path for SQL statements when the contents of a plan change.

Tip: If the physical storage of a plan is changed or dropped, rebinding updates the path of the SQL statements.

Platform Availability

- DB2 z/OS

To Rebind a Plan

- In Rapid SQL, 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.

Platform Availability

- SQL SVR

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see 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)

The **Rebuild Indexes** dialog 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 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.

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. Initiate a **Rebuild** action against one or more indexes. For more information see Initiating an object operation.

The **Rebuild Indexes** dialog opens.

2. To move the index to a new tablespace, click the **New Tablespace** list and then click the new tablespace.
3. 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.
4. 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.
5. 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.

6. Click **Execute**. For information on the scheduling option, see Scheduling.

Rebuild Index (SQL Server)

The **Rebuild Indexes** dialog 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.

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 Indexes Wizard (SQL Server).
Dependencies	Lets you review the objects potentially impacted by this action.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see 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 Database Navigator, 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 scheduling option, see Scheduling.

Rebuild Outlines

You can rebuild one or more stored outlines in a single operation.

Platform Availability

- ORCL

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Rebuild Table

This action builds and submits an ALTER TABLE statement with an ENGINE=InnoDB option.

Platform Availability

- MySQL

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Recompile

You can recompile one or more tables. Recompilation causes each procedure and trigger that uses the target table to be recompiled the next time it runs.

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.

Platform Availability

- SQL SVR, SYB ASE

To Recompile One or More Tables

1. Initiate a **Recompile** action against one or more tables. For more information see Initiating an object operation.

The **Recompile Tables** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

Refresh Table

The **Refresh Table** dialog lets you reload materialized query tables that have been defined with refresh options.

Platform Availability

- DB2 LUW

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

For procedures on restoring a damaged master database, consult the Commands Reference Manual.

Refresh Materialized View

This action lets you build and submit a REFRESH MATERIALIZED VIEW statement, to replace the contents of a materialized view.

Platform Availability

PSTGRS

Note: This action is only available against views created with the Is Materialized control selected. For more information, see Views Wizard (PostgreSQL).

To replace the contents of a materialized view

1. Initiate an **Refresh Materialized View** action against a view. 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	If **With No Data** is selected, a WITH NO DATA clause is included in the generated DDL. No new data is generated and the materialized view will be in an unscannable state.
Dependencies	Review the referring and referred objects that will be automatically resolved when you execute this operation. For details, see Dependencies.
Preview	Displays the DDL that will execute the object action. For details, see Preview.

3. Finally, use the **Execute** button to create the object.

Topics

- Backup Certificate

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 ASE

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	You can 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. Databases with a **Type** of ARCHIVE or with the **In-memory** property selected cannot be renamed. For more information, see Databases (Sybase ASE) - Properties.

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. For more information, see Dependencies.	
Preview	Preview the DDL generated for the operation. For more information, see Preview.	

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Reorganize /Reorg

Reorganize or Reorg options are offered 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 ASE Indexes and Tables)
- Reorg Index (DB2 z/OS)

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.

Step	Description
Action Options	o Choose a **Reorg Type** of table or all indexes of the table. o Optionally, choose an **Index Schema** and specify an **Index name**. o Enable or disable **Inplace Reorg** and if you enable, choose an **Inplace reorgmode** of Start, Stop, Pause, or Resume. o Select an **Access mode** to control read and write access. o Select an **Index reorgmode** to clean up empty pages, delete after cleaning up empty pages, or convert to a type 2 index. o If you did not enable **Inplace Reorg**, select a **Tablespace**. o Select a **Long Tablespace**. o Enable or disable the following options: **Index scan**, **Reorglong 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. For more information, see Preview..

Reorganize Dialog Box (Multiple Tables)

The table below describes the options and functionality on the **Reorganize** dialog.

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 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: 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 Indexes Editor (SQL Server).

To reorganize an index

1. Initiate a **Reorganize** action against an index, primary key, or unique key. For more information see Initiating an object operation.

The **Reorganize Index** wizard opens.

2. 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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Reorganizing Oracle Objects

The **Reorganize** dialog 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 scheduling option, see 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.

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Reorganize (Sybase ASE Indexes and Tables)

The R**eorganize** dialog 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 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:

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, you can reorganize them in increments over multiple sessions by specifying a maximum duration for each session. After tables are reorganized for the specified duration, the operation stops until you resume it again from the Options box of the **Reorganize Table** dialog. 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, the reorg executes 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, the reorg runs 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 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 it is estimated that the switch phase will not start by the deadline, the reorganization terminates. **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.

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**.

Deadline Expression Builder

The table below describes the options and functionality on the **Deadline Expression Builder** dialog

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.

Condition Dialog Box

The **Condition** dialog lets you type free form condition text.

Note: The correctness of the condition text is not tested.

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.

Platform Availability

- MySQL

To Repair a Possibly Corrupted Table

1 On the Datasource Navigator, expand nodes until the **Tables** node is visible.

2 In Rapid SQL, expand the **Tables** node.

3 Right-click one or more selected tables and select **Repair Tables** from the context menu.

The **Repair Tables** dialog opens.

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.	
Preview	Displays the DDL that will execute the object action. For more information, see Preview.	

5 Click **Execute**. For information on the scheduling option, see Scheduling.

The results set opens on the **Results** tab of an ISQL editor window. For more information, see Using the 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.

Platform Availability

- DB2 LUW, DB2 z/OS,

  ITB/FBD *,

  MySQL, ORCL, SQL SVR, SYB ASE

To Generate a Detail Report on One or More Objects

1. In Rapid SQL, on the Datasource Navigator, expand the target object node.

2. Right-click one or more selected objects, and then select **Report** from the context menu.

The **Report** dialog opens.

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**.

The report opens in the built-in browser and writes a copy to disk.

Restart (Object Action)

The **Restart Sequence** dialog lets you restart a sequence, starting at the value specified in the sequence definition or starting at a specified value.

Platform Availability

- DB2 LUW

To Rebuild a Stored Outline

1. Initiate a **Restart** action against a sequence object. For more information see Initiating an object operation.

The **Restart Sequence** dialog opens.

2. Select one of the following **Restart Sequenceat** options:
   - **Originating Value** use the value specified in the object definition
   - **This value** lets you specify a custom start value
3. Click **Execute**. For information on the scheduling option, see Scheduling.

Restart Sequence(s)

This action builds and submits an ALTER SEQUENCE... RESTART/RESTART WITH statement. This lets you reset the current value to either the initial value specified when the sequence was created, or reset it to a specified value.

Platform Availability

- SQL SVR

To reset the value of a sequence

1. Initiate a **Restart Sequence(s)** action against one or more sequences. 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	Start With	An integer that will be returned as the next value of the sequence object. The integer must be within the range specified for the sequence. If no value is provided, sequence numbering restarts with the initial value specified when the sequence was created.
Dependencies	Lets you review the objects potentially impacted by this action. For more information, see Dependencies.	
Preview	Preview the DDL generated for the operation. For more information, see Preview.	

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Resynchronize

This action executes an ALTER DATABASE... FOR PROXY_UPDATE statement, forcing resynchronization of proxy tables within the proxy database.

Note: This functionality is available for Sybase ASE 12.5.

To manually resynchronize proxy tables

1. Initiate a **Resynchronize** 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 settings in the dialog:

Step	Settings and tasks
Action options	Lists the database or databases you selected to resynchronize.
Preview	Lets you preview the DDL generated for the operation.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Role Activation

You can activate or deactivate roles for the current session. The **Role Activation** action builds and submits one or more SET ROLE commands.

Platform Availability

- ORCL, SYB ASE

To Enable or Disable One or More Roles in the Current Session

1. Right-click on a connected Sybase ASE 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 ASE). 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**.

3. Click **Execute**.

Run Job (Oracle Jobs)

This action builds and submits a DBMS_SCHEDULER.RUN_JOB call, letting you run a job immediately.

Platform Availability

- ORCL 10g ^

To run a job immediately

1. In Rapid SQL, on a connected Oracle datasource, expand the **Scheduler** nodes and select the **Jobs** node.

2. Right-click a job and select **Run Job** from the context menu.

3. Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action options	Select the **Use current session** check box to provide a USE_CURRENT_SESSION attribute value of TRUE. Deselect the **Use current session** check box to provide a use_current_session attribute value of FALSE.
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.

4. Click **Execute**. For information on the scheduling option, see Scheduling.

Run Job (job queues)

You can 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.

Platform Availability

- ORCL

To Run a Job

1. Initiate a **Run Job** action against a job queue. For more information see Initiating an object operation.

The **Run** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

Topics

- Enable Job (Job Queue)
- Disable Job.

Schema (Object Action)

The **Schema** dialog lets you view key properties of the columns in a table or view.

Platform Availability

- DB2 LUW, DB2 z/OS,

  ITB/FBD *,

  MySQL, ORCL, SQL SVR, SYB ASE

To View a Column Summary for a Table or View

1. Initiate a **Schema** action against a table or view. For more information see Initiating an object operation.

Select * From

This action lets you retrieve all data contained in tables or views. Results are displayed in a row and column grid.

Platform Availability

- DB2 LUW, DB2 z/OS,

  ITB/FBD *,

  MySQL, ORCL, SQL SVR, SYB ASE

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, a two-tabbed window opens:

- 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 Using the 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 Using the SQL 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.

Platform Availability

- ORCL 10g ^

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. For more information, see Dependencies.
Preview	Preview the DDL generated for the operation. For more information, see Preview.

Set Integrity

DB2 SET INTEGRITY statement support lets 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 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.

Platform Availability

- DB2 LUW

To Open the Set Integrity Wizard

1. On the Navigator, expand a DB2 database, and click or expand 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	
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/NoForce 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	
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 lets you disable your databases to prevent access, and enable your databases to grant access through the **Datasource** menu.

Platform Availability

- SQL SVR, SYB ASE

Important Notes

For Sybase ASE, you can only 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

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 scheduling option, see Scheduling.

Set Statistics

This action creates and submits a SET STATISTICS statement, letting you recompute the selectivity of an index.

Platform Availability

- ITB/FBD *

To Recompute the Selectivity of an Index:

1. On the Database Navigator, 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.

The **Set Statistics** dialog opens.

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 scheduling option, see Scheduling.

Set UNDO

Set UNDO Tablespace dialog lets you dynamically set an UNDO tablespace if the tablespace is running in AUTO UNDO mode.

Platform Availability

- ORCL 9i ^

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. For more information, see Preview.

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Shrink

For details on using 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

You can 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. For more information, see Preview.	

3. Click **Execute**. For information on the other options, see Preview and Scheduling.

Shrinking Oracle rollback segments

You can 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. For more information, see Preview.	

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Shrinking Oracle tables or indexes

You can 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 Tables 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. For more information, see Preview.	

3. Click **Execute**. For information on the scheduling option, see Scheduling.

Start Database

The **Start Database** dialog lets you start a database:

- When the database that has been stopped with a **Stop Database** dialog. For more information, see Stop Database.
- 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.

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.

Platform Availability

- DB2 z/OS

To Start a Database

1. In Rapid SQL, on the Datasource Navigator, expand the **Databases** node.

2. Right-click the target object and select **Start Database** from the context menu.

The **Start Database** dialog opens.

3. Use the following table as a guide to understanding and settings options on this dialog:

Option	Description
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 scheduling option, see Scheduling.

Stop Database

The **Stop Database** dialog lets you stop a database, optionally letting you allow running applications access until their next COMMIT.

Platform Availability

- DB2 z/OS

To Stop a Database

1. In Rapid SQL, on the Database Navigator, expand the Databases node.

2. Right-click the target object and select Stop Database from the context menu.

The **Stop Database** dialog opens.

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 scheduling option, see Scheduling.

Stop Job

This action builds and submits a DBMS_SCHEDULER.STOP_JOB call, letting you stop a job immediately.

Platform Availability

- o ORCL 10g ^

To stop a job

1. In Rapid SQL, on a connected Oracle datasource, expand the **Scheduler** nodes and select the **Jobs** node.

2. Right-click a job and select **Stop Job** from the context menu.

3. Use the following table as a guide to understanding and modifying settings in the dialog:

Step	Settings and tasks
Action options	Select the **Force Stop** check box to provide a FORCE attribute value of TRUE forcing the job to terminate immediately. Deselect the **Use current session** check box to provide a FORCE attribute value of FALSE.
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.

4. Click **Execute**. For information on the scheduling option, see Scheduling.

Switch Online

The **Switch Online** dialog lets you access a tablespace by switching it online after the parent container(s) have been recovered or restored.

Platform Availability

- DB2 LUW

To Switch a Tablespace Online:

1. Initiate a **Switch Online** action against one or more tablespaces. For more information see Initiating an object operation.

The **Switch Online** dialog opens.

2. Click **Execute**. For information on the scheduling option, see Scheduling.

Transfer Ownership

An object owner or a user with SECADM authority can 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.

Platform Availability and object type support

- DB2 LUW 9.1 ^

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. For more information, see Preview.

Truncate

This action 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, all the rows are deleted. These rows are not logged as individual drops and cannot be recovered from a transaction log or other type of log.

Platform Availability

- DB2 LUW, DB2 z/OS, MySQL, ORCL, SQL SVR, SYB ASE

To Truncate a Table or Cluster

1. Initiate a **Truncate** or **Truncate Table** action against one or more tables or clusters. For more information see Initiating an object operation.

The **Truncate** wizard/dialog box opens.

2. 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
3. Click **Execute**. For information on the scheduling option, see 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.

Notes on truncating Sybase ASE objects

Truncate against a Sybase ASE 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.

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.

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.

Platform Availability

- SYB ASE

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 scheduling option, see Scheduling.

Unquiesce

The Unquiesce operation lets you restore user access to a single database or all databases of an instance previously quiesced.

Platform Availability

- DB2 LUW

Note: RUnquiesce Database for supported servers from IBM DB2 for Windows/Unix 7 or earlier clients is not supported.

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.

The **Unquiesce Database** or **Unquiesce Database** dialog opens.

2. Click **Execute**.

Topics

- Quiesce

Update Statistics

The **Update Statistics** dialog 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 version 8.2.2)
- 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 ASE)

Updating statistics for tables or indexes (IBM DB2 for Linux, Unix, and Windows version 8.2.2)

To Update Statistics for an Index or Table

1. Initiate an **Update Statistics** action against one or more tables or indexes. For more information see Initiating an object operation.

The **Update Statistics** dialog opens.

2. 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	A CPU sampling technique is used 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 the statistics are being updated.
	Allow read/write access during collection	Allows read and write access while the statistics are being updated.

3. 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, you can 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. For more information, see Dependencies.
Preview	Lets you view the DDL generated for the operation. For more information, see Preview.

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 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 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 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. For more information, see Preview.	

3. Use the **Execute** or **Schedule** button to perform the operation.

Updating statistics for indexes or tables (Sybase ASE)

The **Update Statistics** dialog 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.
Columns	(available only when a single table is selected)	Lets you specify all columns or specific columns to be used for the operation.
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. For more information, see Preview.	

3. Use the **Execute** or **Schedule** button to perform the operation.

Coding Environments/Editors

Rapid SQL provides coding environments that let you create, edit, and where appropriate, execute scripts containing for example, SQL and DDL. 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.

The following topics describe use of the Rapid SQL coding environments:

- Common Word-processing Operations describes common editor options such as file and clipboard operations.
- Common Coding Environment Features describes coding environment options such as commenting and working with line numbers.
- Using the SQL Editor provides details on the SQL Editor.
- Using the DDL Editor provides details on using the SQL Editor to generate object-creation code.

Common Word-processing Operations

The following topics describe features available in all coding environments/editors:

- Clipboard Functions
- File operations
- Find/Replace Options in Coding Environments
- Regular Expressions Support
- Changing Case
- Printing Options

Clipboard Functions

The coding/scripting environments all offer the most common clipboard-related options involving selecting text, cutting, copying, and pasting text. The following topic describes the clipboard option offered:

- Cutting/Copying and Pasting Columns

Cutting/Copying and Pasting Columns

To cut or copy and then 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, select **Copy**s or **Copy**.
4. In the Editor window, position the pointer where you want to paste the column.
5. On the **Edit** menu, select **Paste**.

File operations

The following topics provide key information on file options available in coding environments/editors:

- Opening Files
- Inserting a File Into an Open Script
- Saving and Closing Scripts

Opening Files

All text-based editors offer an **Open File** facility. It provides dedicated support for files of the following types:

- Script (**.sql**), data definition languge (**.ddl**), query (**.qry**), and results (**.rsl**) files
- Query Builder files (**.eqb**) - for more information see Query Builder.
- XML (**.xml**) and text (**.txt**) files
- Web (**.htm**, **.html**, **.asp**) files
- Procedure (**.prc**, **.pro**), VBScript (**.vbs**),
- JScript (**.js**), and Java (**.java**) files

To open a supported-type file

1. Select File > Open. An **Open File(s)** dialog opens.
2. Use the dialog to locate, select a file type, and select one or more files to open.

If you select a file type of **All Files**, you can open a file with a suffix not corresponding to one of the supported file types. One of two actions is taken:

- It automatically opens the file in the SQL Editor
- It prompts you for additional information on the file before opening

Note: The specific action taken is dictated by an Options Editor selection. For details, see ISQL Options - Main Tab.

If you selected the Options Editor setting to prompt, when you select an unknown file type for opening, a **What Type of File Is** dialog opens. The table below describes the options and functionality on the dialog:

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 or the file contains Oracle Anonymous PL/SQL.	Lets you select object type and owner and provide an 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.

An SQL Editor or DDL Editor opens, as appropriate. For details, see the following topics:

- Using the SQL Editor
- Using the DDL Editor

For more general information, see Coding Environments/Editors.

Inserting a File Into an Open Script

The ISQL Editor facilitates the reuse of SQL scripts by letting you insert an existing file into another script.

To insert a file at the current cursor location

1. In an open editor, place the cursor where you want to insert the file.
2. Select Edit > Insert File. Tthe **Insert File into Current File** dialog opens.
3. Use the dialog to locate, select a file type, and select the files to insert.

For more general information, see Coding Environments/Editors.

Saving and Closing Scripts

In Rapid SQL, untitled scripts are named according to the following conventions:

Environment	Default File Name	Default File Suffix
SQL Editor	**SQL1...SQLn**	**.sql**
DDL Editor	The object name provided when invoking the DDL EDitor	**.sql**
HTML Editor	**HTML1...HTMLn**	**.htm**
Text Editor		**.txt**

Note: If you have set the **Auto-Save** feature in the Options Editor, a temporary copy of your scripts is automatically saved periodically as you work. For details, see ISQL Options.

To close a script

1. On the on the Main menu, click **Close**.
2. If you have not saved your script, you are prompted to save the file. Click **Yes** to save and **No** to close without saving.

Explicitly Saving a Script

With an ISQL window open, selecting File > Save or File > Save As opens a standard **Save As** dialog. It lets you save a new script, save an existing script with a new name and optionally with a different file suffix. The following **Encoding** schemes are supported:

ANSI	UNICODE	UTF-16LE (No BOM)
UTF-16 Big Endian	UTF-16BE (No BOM)	UTF-8
UTF-8 (No BOM)	UTF-32 Little Endian	UTF-32LE (No BOM)
UTF-32 Big Endian	UTF-32BE (No BOM)	

Automatic Save Operations

If you shut down the application or close an editor window while there are modified, unsaved scripts, you are prompted to save before completing the operation. The table below describes the options and functionality on the **Save Modified Documents** dialog

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.

Find/Replace Options in Coding Environments

Each text-based editor offers basic find and find/replace text functions.

To invoke the Find or Find/Replace facility

1. With a text editor active, select Edit > Find or Edit > Replace.
2. Use the following table as a guide to working with the resulting dialog:

Option	Description
Find What	Lets you type your search string.
Replace With (Replace only)	Lets you type the replacement text.
Match whole word only	Select to search for only the complete word.
Match Case	Select to make the search case-sensitive.
Regular expression	Indicates the the find string contains regular expressions. For more information, see Regular Expressions Support.
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	Click to find the next occurrence of your search string.
Bookmark All (Find only)	Places a bookmark at each line containing the find string. For details, see Working with Coding Environment Bookmarks.
In Selection	Replaces all instance of the find string with the replace string within a selected block of text.
Replace Button (Replace only)	Click replace the current selection.
Replace All Button (Replace only)	Click to automatically find and replace all occurrences of your search string within the current window.

Selecting Edit > Repeat Find finds the next instance of the most recently used find string without opening the **Find** or **Replace** dialog.

Regular Expressions Support

Some features, such as file search, SQL Editor search, and Browser list filtering, support the use of regular expressions. This provides flexibility in in searches and other situations in which text string qualification is required.

Detailed documentation on regular expressions is beyond the scope of this document. For descriptions of syntax and metacharacters, consult third-party documentation on the regular expressions standard. The following table provides short descriptions of the most commonly used regular expression metacharacters used with Rapid SQL features:

Metacharacter	Description
.	Matches any single character.
*	Matches zero or more characters.
+	Matches one or more characters.
^	The start of a string.
$	The end of a string.
\b	Word boundary.
\d, \w and \s	Shorthand character classes matching digits, word characters (letters, digits, and underscores), and whitespace (spaces, tabs, and line breaks).
\D, \W and \S	Negated versions of the above that should only be used outside character classes.
[ ]	Matches any one character enclosed between the square brackets.
[^]	Matches any character NOT enclosed between the square brackets.
{}	Matches aspecified number of repetations. For example `ab{2,4}` matches a**bb**, a**bbb**, and **abbbb**.
|	OR.
\	Escape special character.
()	Tag expression.
\s	Space or Tab.

In the case of some specific features, regular expression usage must be enabled or configured. For more information, see the Rapid SQL feature topics for details:

- Find/Replace Options in Coding Environments
- Find in Files
- Column filtering in the Browser

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, select **Upper Case**.

OR

On the **Edit** menu, select **Lower Case**.

Printing Options

A number of standard print options are provided from text-based editors:

- Selecting File > Print opens a dialog that lets you print the contents of the editor with options such as choosing a printer, specifying a page range, and specifying the number of copies to print
- Selecting File > Print Preview opens a new window showing a printed page appearance for the current document
- Selecting File > Print Setup opens a dialog that lets you select a printer, select printing options, and specify preferences

Common Coding Environment Features

The following features are avaiable in all coding environments/editors:

Working with Comments

You can comment and uncomment blocks of code using the commenting scheme appropriate to the DBMS platform you are using.

To comment out code

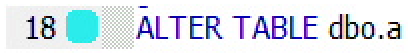
1. Select the block of code you want to comment out and then select Edit > Comment Out.

To uncomment code

1. Select a complete block of commented code or a single line of the block and then select Edit > Undo Comment Out.

Working with Coding Environment 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. A blue dot is inserted in the gutter next to lines that you bookmark.



```
18 ● ALTER TABLE dbo.a
```

To set a bookmark on a line or clear an existing bookmark from a line

1. In the Editor window, position the pointer in front of the line and then select Edit > Toggle Bookmark.

To move the cursor back and forth among bookmarked lines

1. Select Edit > Next Bookmark or Edit > Previous Bookmark.

To clear all bookmarks

1. Select Edit > Clear Bookmark.

GoTo Line x

The **Goto** dialog lets you move to a specific line or column in your script.

To complete the **Goto** dialog, do the following:

1. On the **Edit** menu, select **Goto**. The **Go To** dialog opens.
2. In **Line Number**, type or select the target line number.
3. Click **OK**.

The cursor moves to the target line.

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.

To find matching tokens

1. Position the insertion pointer just to the left of the first token in a token pair you want to locate.
2. Click the **Match Token** button.

The pointer jumps to the next available token.

Using the SQL Editor

Rapid SQL incorporates a powerful SQL scripting environment, the SQL Editor. The SQL Editor lets you write, debug, test and deploy solid SQL code for your database applications.

You can open multiple SQL windows. In addition, when you open an SQL file, it opens in an SQL Editor window. For more information, see Opening Files.

To open a SQL Editor window

1. Connect to the datasource that you want to write a script against and ensure that it is the active datasource. For more information, see Connected/Selected Datasource options.
2. On the **File** menu, select **New ISQL**.

An SQL Editor window opens in your current workspace.

The following topics provide details on functionality specific to the SQL Editor:

- SQL Editor Windows Basics
- Valid Content in the SQL Editor
- Automated Error Detection and Coding Aid Features
- Execution and Execution-related SQL Editor options
- Using the Results Editor
- Miscellaneous SQL Editor Tasks

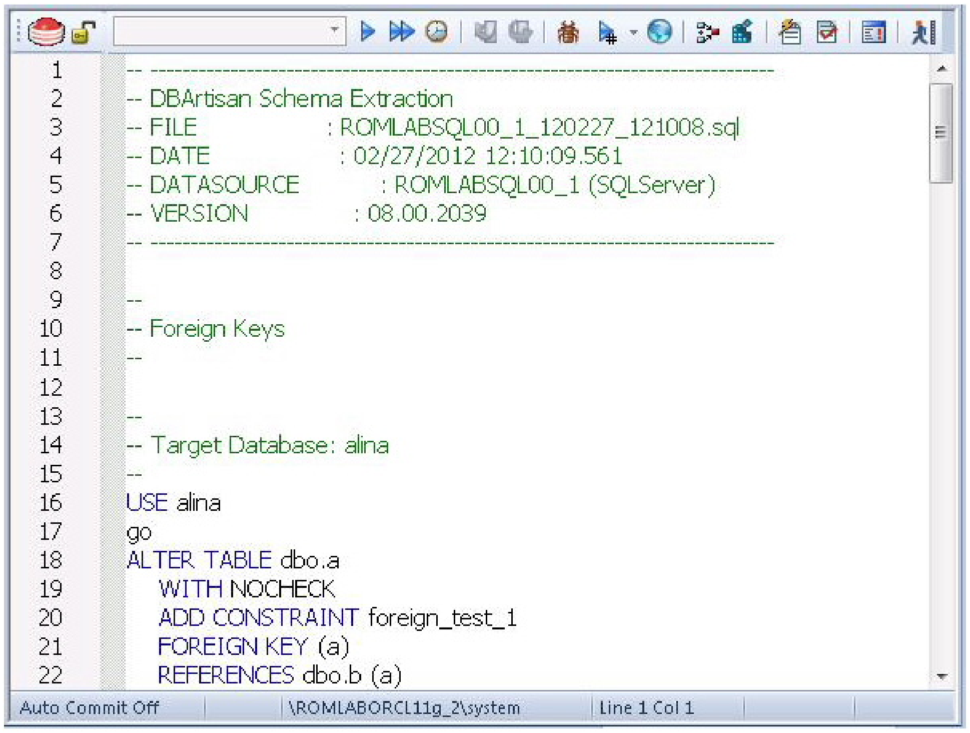
See the following topics for information that the SQL Editor shares with other Rapid SQL environments:

- Common Word-processing Operations
- Common Coding Environment Features

Note: The **Tools** menu offers a number of code generation and execution utilities that can be used in conjunction with the SQL Editor. For details, see Tools.

SQL Editor Windows Basics

The SQL Editor consists of a shortcuts toolbar, an editing area, and a status bar.



```
-- ---------------------------------------------------------------------------------
-- DBArtisan Schema Extraction
-- FILE                : ROMLABSQL00_1_120227_121008.sql
-- DATE                : 02/27/2012 12:10:09.561
-- DATASOURCE          : ROMLABSQL00_1 (SQLServer)
-- VERSION             : 08.00.2039
-- ---------------------------------------------------------------------------------

--
-- Foreign Keys
--

--
-- Target Database: alina
--
USE alina
go
ALTER TABLE dbo.a
    WITH NOCHECK
    ADD CONSTRAINT foreign_test_1
    FOREIGN KEY (a)
    REFERENCES dbo.b (a)
```

Auto Commit Off | \ROMLABORCL11g_2\system | Line 1 Col 1

For information on the toolbar and status bar, see the following topics:

- o SQL Editor Toolbar Options
- o ISQL Window Status Bar

For information on using functionality available in the editing area, see the following topics:

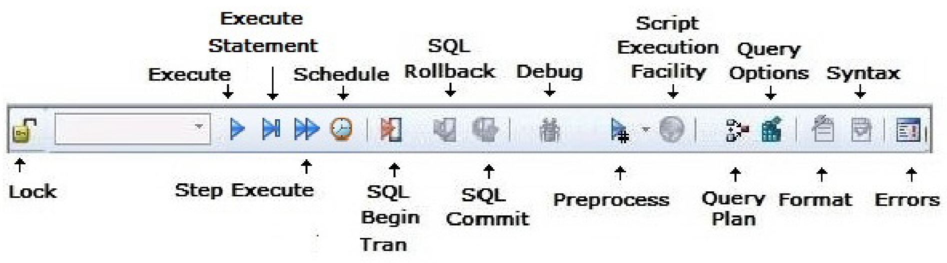
- o Common Word-processing Operations
- o Common Coding Environment Features
- o Valid Content in the SQL Editor
- o Automated Error Detection and Coding Aid Features
- o Execution and Execution-related SQL Editor options
- o Miscellaneous SQL Editor Tasks

Topics

- o Toggling Display of the Error Pane.

SQL Editor Toolbar Options

The SQL Editor toolbar is displayed above the editing window.



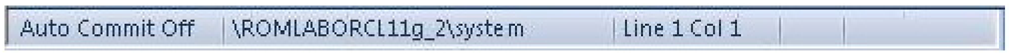
The table below describes the options of the ISQL Editor toolbar:

Option	Description
Lock - Unlock Connection	Lets you lock an ISQL Window to a specific datasource connection or unlock a currently locked window. For details, see Locking a SQL Editor Window to a Datasource.
Execute	Executes the script.
Execute Statement	Execute the statement at the cursor. For more information, see Executing the Current Statement.
Step Execute	Initiates step execution of the script. For more information, see Step Executing a Script.
Schedule	Opens a dialog 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. For more information, see 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 being 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 SQL Preprocessing: #define and #include.
Script Execution Facility	Provides a shortcut to opening the Script Execution Facility. For more information, see Script/File Execution Facilities.
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 Using the Query Plan Facility.
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 automatic syntax checking is therefore disabled. For more information, see Syntax Checking.
Errors	Toggles the **Errors** pane. For more information, see Toggling Display of the Error Pane.

ISQL Window Status Bar

The ISQL window Status bar is located below the editable area of the ISQL window.


Auto Commit Off | \ROMLABORCL11g_2\system | Line 1 Col 1

It 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 ASE)
- Keyboard Emulation Mode - Lets you customize your editing environment to an emulation mode.

Tip: For Microsoft SQL Server and Sybase ASE, 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 set emulation and auto commit preferences. For details, see ISQL Options.

Valid Content in the SQL Editor

The SQL Editor lets you create, edit, and execute scripts written in the SQL flavor native to each supported DBMS platform. The following topics cover additional content that can be included in your scripts.

- SQL Preprocessing: #define and #include
- ETStart and ETEnd tags
- Bind Variable Parameterization in Prepared Statements (Dynamic SQL)

Topics

- Automated Error Detection and Coding Aid Features
- Execution and Execution-related SQL Editor options

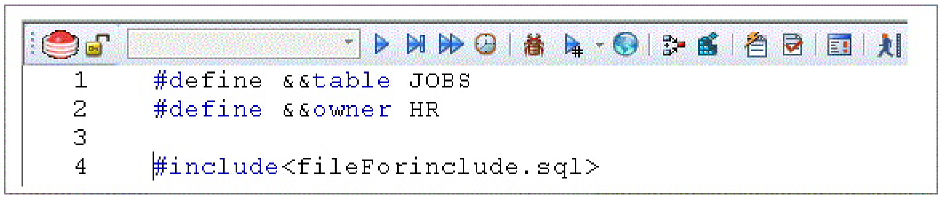
SQL Preprocessing: #define and #include

SQL preprocessing provides similar functionality to that 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

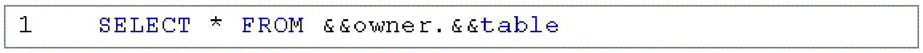
The following figure illustrates the result of basic preprocessing of a script, nesting of **#define** directives/references, and the notations required by the ISQL editor. The original script includes two **#define** directives and a **#include** reference to a one-line file named **fileForInclude.sql**. The referenced file includes two identifiers to be replaced with **#define** processing.

Script content before SQL preprocessing



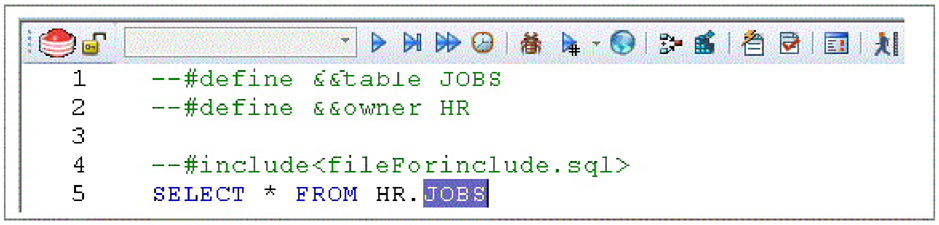
```
#define &&table JOBS
#define &&owner HR

#include<fileForinclude.sql>
```

Contents of fileForinclude.sql



```
SELECT * FROM &&owner.&&table
```

Script content produced by SQL preprocessing



```
--#define &&table JOBS
--#define &&owner HR

--#include<fileForinclude.sql>
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 Setting up Rapid SQL to Preprocess #include directives.
- **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 #include Functionality and Syntax and #define Functionality and Syntax.

- **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, the following locations are searched, 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 Datasources.
2. The location specified on the Directories tab of the Options editor. For details, see Directories Options.

Note: For detailed information on setting options, see Specifying Application Preferences 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

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 &&*IdentifierReplacement_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 &amp;&amp;myTable Clients
#define &amp;&amp;embtClients Embarcadero
#define &amp;&amp;tempTable New&amp;&amp;myTable
#define &amp;&amp;embtTempTable &amp;&amp;embtClients&amp;&amp;myTable

SELECT * FROM &amp;&amp;tempTable;
SELECT * FROM &amp;&amp;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:

- o The script can be simply preprocessed, 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.
- o The script can be preprocessed and executed 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 Script/File Execution Facilities for other script execution options
- See **SQL Preprocessing: #define and #include** for an introduction to SQL preprocessing in the ISQL editor

ETStart and ETEnd tags

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. Two tag pairs, **ETStart** and **ETEnd**, 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

...

Bind Variable Parameterization in Prepared Statements (Dynamic SQL)

You can provide bind variable values when executing scripts containing prepared statements. Bind variables, or parameterized literals, can help optimize explain/execution plan usage by minimizing the requirement to "hard parse" queries or statements that differ by one or more literal values.

In scripts, Rapid SQL recognizes named or unnamed variables that use the following notation:

DBMS Platform/Driver Type	Notation
Oracle with native driver	:*<name>* or :*<number>*
Oracle With JDBC driver	?
All other platforms/drivers	?

For example:

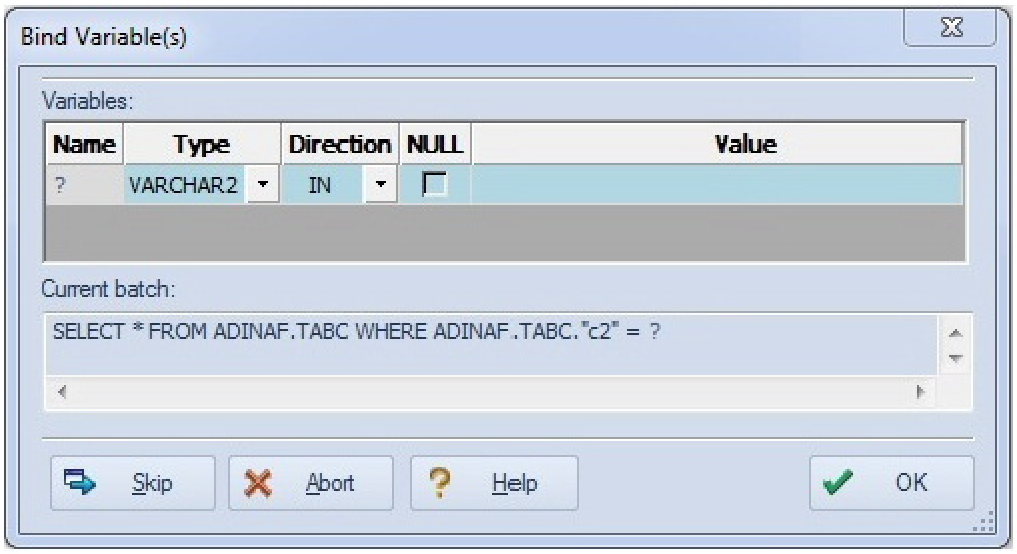
```
SELECT * FROM MYTABLE WHERE MYCOLUMN = ?
```

OR

```
CALL MYPROC ( :a, :b, :c )
```

Note: Support for bind variables is enabled on a editor window-by-editor window basis. Bind variable parameterization is enabled by setting the **Prepare Batch** Query Option, available on all DBMS platforms. For details, see Setting up the Execution Environment with Query options.

For an enabled SQL Editor window, when using execution options, whenever a statement containing bind variable notation is encountered, a **Bind Variables** dialog opens.



Name	Type	Direction	NULL	Value
?	VARCHAR2	IN	☐	

```
SELECT * FROM ADINAF.TABC WHERE ADINAF.TABC."c2" = ?
```

For each bind variable in the current statement, the dialog lets you specify:

- A type
- An IN, OUT, or INOUT designation for elements such as function or procedure parameters or arguments, variables used in INTO clauses of an INSERT statements and variables used in RETURNING clauses of an Update statements.
- A value

You can then execute, skip the statement, or abort execution for the currently running script.

Automated Error Detection and Coding Aid Features

Code is analyzed as you add content to an ISQL editor session and offers the following automated features:

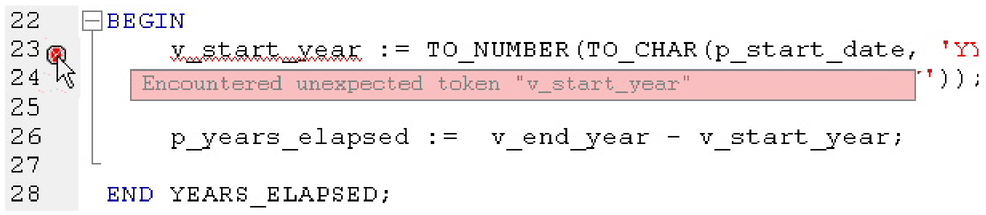
- Syntax Checking: Automatically flag syntactical errors in your scripts.
- Semantic Validation: Detect object name references to objects not present in the datasource index.
- Code Complete: Insert or replace object names, selected from suggestion lists, as you edit a script.
- Hyperlink Object Actions: Invoke common object operations such as **Open** and **Extract**, as well as object type-specific operations, from object names referenced in a script.
- Paste SQL Syntax: Paste syntax for SQL statements, system functions calls, and other commonly used elements into an SQL Editor.
- Paste SQL Statements: Generate and paste complete SELECT, INSERT, DELETE, UPDATE or EXEC statements or function call syntax into an SQL Editor.
- Using Code Templates: Create templates for statements or code blocks that can be quickly added to scripts in Rapid SQL.
- Using Auto Replace Expressions: Create short text representations of larger text strings that can be used to quickly add the larger string to a script.
- Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor: Make use of Embarcadero Team Server 2016 metadata in the SQL Editor.
- Toggling Display of the Error Pane: View details on errors.
- ISQL Performance: Provides information and statistics about SQL statements and tables in the SQL editor. **ISQL Performance** provides recommendations to improve the performance of the system.

Syntax Checking

The ISQL editor can perform on-the-fly or manually-initiated syntax checking. The syntax of the contents of an SQL Editor widow is checked 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, the script is annotates assistance is offered as follows:

- o Line numbers for lines containing syntax errors are flagged with an error icon.
- o Hovering the mouse over an error icon displays a tooltip with an error message
- o The specific error in the line of code is underlined in red



```
BEGIN
    v_start_year := TO_NUMBER(TO_CHAR(p_start_date, 'YY
    Encountered unexpected token "v_start_year"          '));

    p_years_elapsed :=  v_end_year - v_start_year;

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 Using the DDL Editor.

Syntax checks can be configured to be performed automatically or you can require a syntax check to be initiated manually:

- o **Automatic** - Automatic syntax checking is enabled from the Options Editor. For details, see ISQL Options - Code Assist Tab.

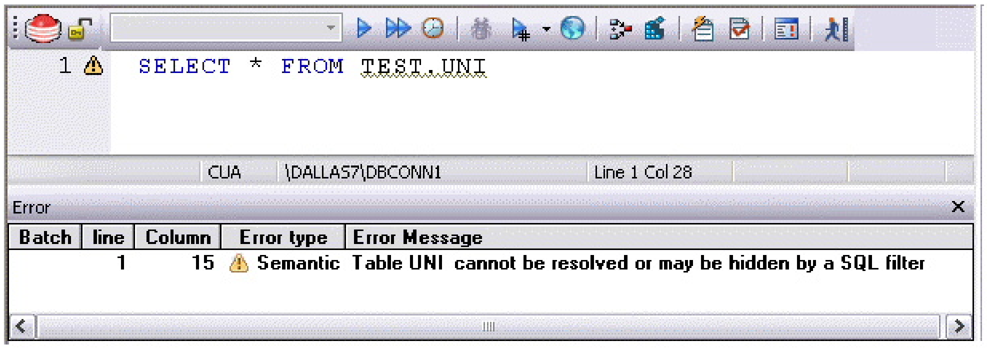
If the **Enable Real-time syntax checking** check box is selected, a syntax check is performed whenever there is an interval of 1.5 seconds or more between key strokes. Syntax error annotations persist until the error is corrected and the next automatic syntax check is executed.

- o **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, the names of supported object types are analyzed as you type, and an error condition is raised when it detects a name not present on the datasource.



```
1 ⚠ SELECT * FROM TEST.UNI
```

CUA | \DALLAS7\DBCONN1 | Line 1 Col 28

Error

Batch	line	Column	Error type	Error Message
	1	15	⚠ Semantic	Table UNI cannot be resolved or may be hidden by a SQL filter

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 obtained 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

- Semantic validation is only available for objects and object types configured to be included in the datasource index. For more information, see ISQL Options - Code Assist - Advanced Tab on page 214.
- For SQL Server and Sybase ASE 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 ASE 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.

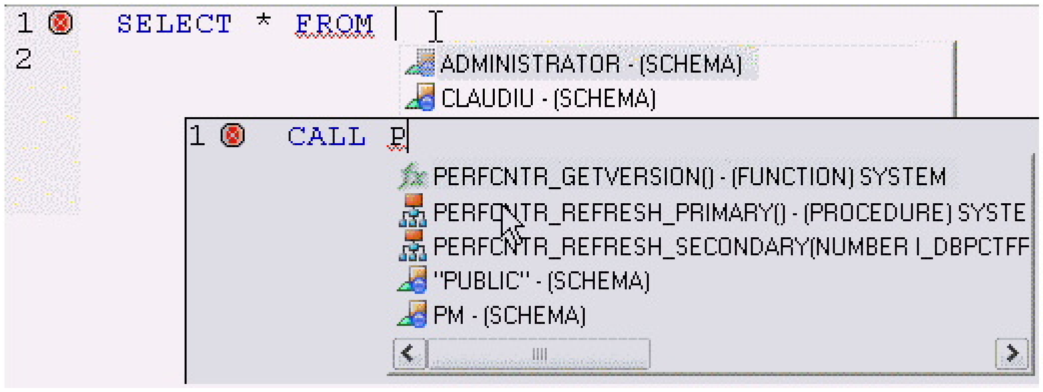
- Ensure that the ISQL Editor window is connected to a datasource. A datasource name is required in order to resolve an object name.

In the case of SQL Server and Sybase ASE, 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 Using the DDL Editor.

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.



```
SELECT * FROM
    ADMINISTRATOR - (SCHEMA)
    CLAUDIU - (SCHEMA)
```

```
CALL P
    PERFCNTR_GETVERSION() - (FUNCTION) SYSTEM
    PERFCNTR_REFRESH_PRIMARY() - (PROCEDURE) SYSTE
    PERFCNTR_REFRESH_SECONDARY(NUMBER I_DBPCTFF
    "PUBLIC" - (SCHEMA)
    PM - (SCHEMA)
```

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

Before using the Code Complete feature, complete the following setup tasks:

- Ensure that the object types for which Code Complete is to be available, are being indexed in the datasource index. For details, see ISQL Options - Code Assist - Advanced Tab.
- 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 ISQL Options - Code Assist Tab.

When the Code Complete feature is enabled, it can be invoked in two ways:

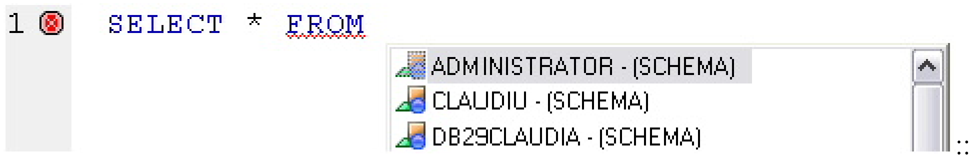
- o Automatically, if auto-activation is enabled. A Code Complete suggestion box is offered when you start typing an object name in a relevant location and then stop typing for an interval that exceeds the specified auto-activation delay.
- o Manually, if auto-activation is disabled or if the specified 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 ISQL Options - Code Assist Tab.

Code Complete offers context sensitive suggestions based on the cursor location within the statement when Code Complete is invoked. For example:

- o **When invoked without any preceding text** - Code Complete will supply basic suggestions appropriate to that clause.

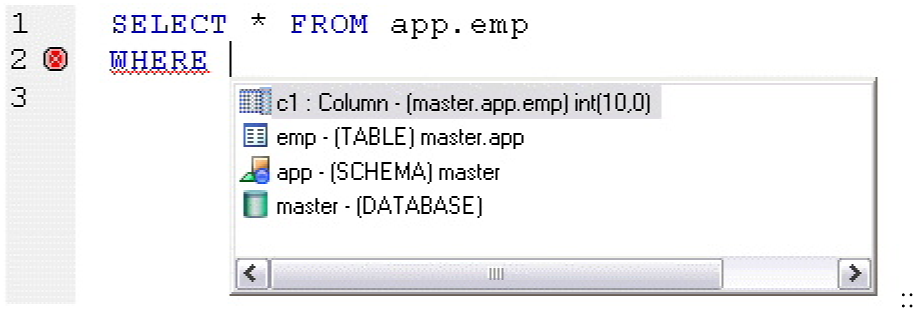
For example, suggestions for the table name in a FROM clause would help a user construct a qualified table or view name. Against a SQL Server datasource for example, this would include schemas and databases.



```
SELECT * FROM
```

ADMINISTRATOR - (SCHEMA)
CLAUDIU - (SCHEMA)
DB29CLAUDIA - (SCHEMA)

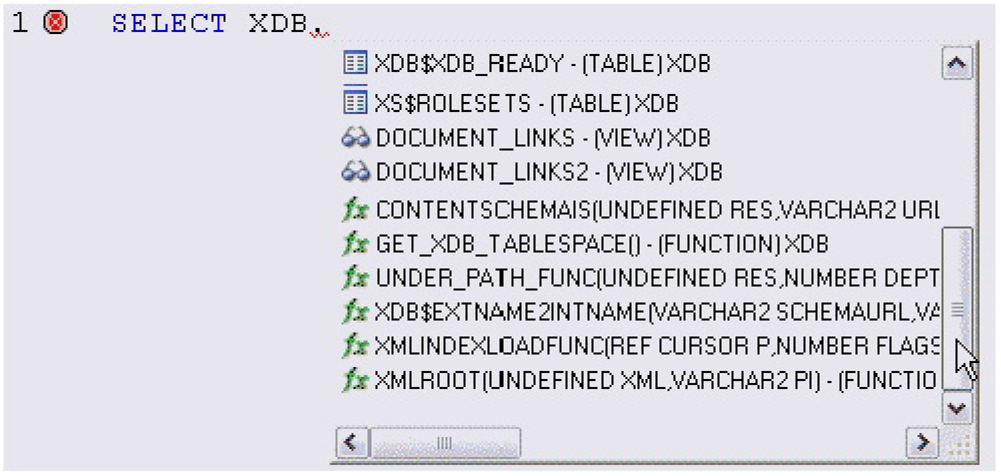
Suggestions for the column name in a WHERE clause in the same statement would contain the schemas, databases and tables\views referenced in the FROM clause as well as a listing of the applicable columns based on the referenced tables\views.



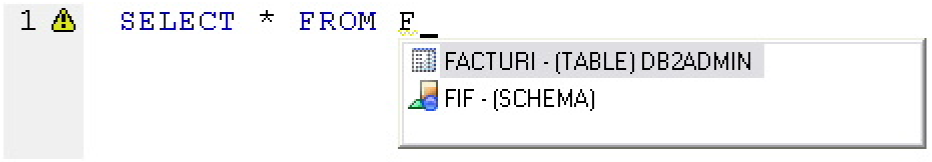
```
SELECT * FROM app.emp
WHERE
```

c1 : Column - (master.app.emp) int(10,0)
emp - (TABLE) master.app
app - (SCHEMA) master
master - (DATABASE)

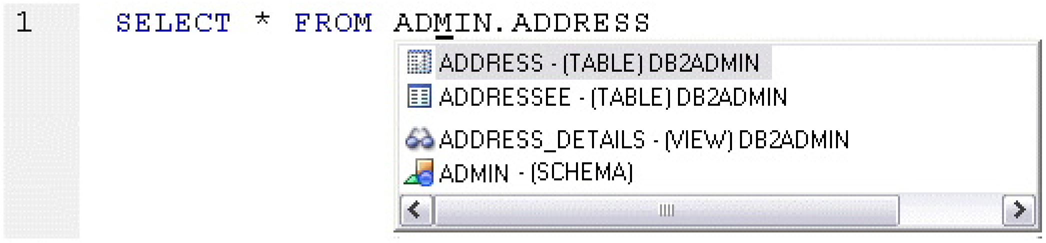
- o **After the delimiter within an object name** - Code Complete offers a list that lets you select the next element to build a fully qualified object name.

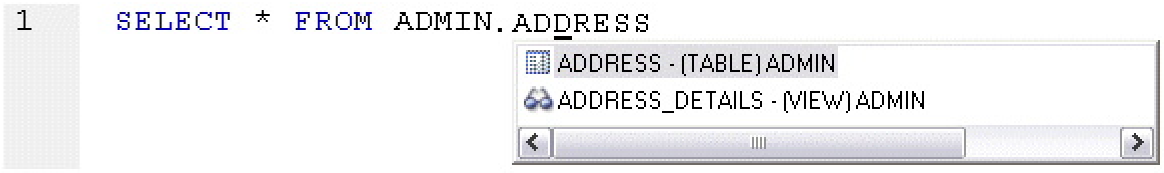

```
SELECT XDB.
XDB$XDB_READY - (TABLE) XDB
XS$ROLESETS - (TABLE) XDB
DOCUMENT_LINKS - (VIEW) XDB
DOCUMENT_LINKS2 - (VIEW) XDB
CONTENTSCHEMAIS(UNDEFINED RES,VARCHAR2 URI
GET_XDB_TABLESPACE() - (FUNCTION) XDB
UNDER_PATH_FUNC(UNDEFINED RES,NUMBER DEPT
XDB$EXTNAME2INTNAME(VARCHAR2 SCHEMAURL,VA
XMLINDEXLOADFUNC(REF CURSOR P,NUMBER FLAGS
XMLROOT(UNDEFINED XML,VARCHAR2 PI) - (FUNCTIO
```

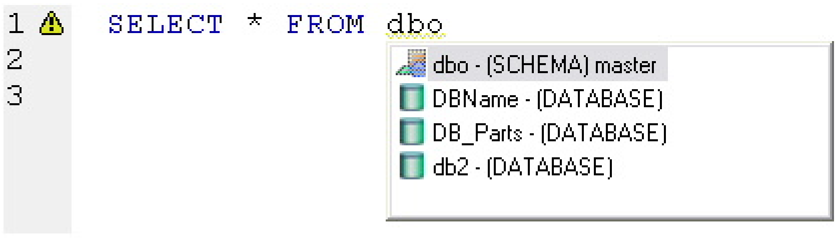
- o **After typing one or more characters of an object name** - Code Complete offers a listing of the names of all object name elements, relevant to the cursor location within the object name, that start with the typed characters.


```
SELECT * FROM F
FACTURI - (TABLE) DB2ADMIN
FIF - (SCHEMA)
```

- o **At any point within a partially-specified or complete object name** - Code Complete offers all selections appropriate to the location of the cursor.


```
SELECT * FROM ADMIN.ADDRESS
ADDRESS - (TABLE) DB2ADMIN
ADDRESSEE - (TABLE) DB2ADMIN
ADDRESS_DETAILS - (VIEW) DB2ADMIN
ADMIN - (SCHEMA)
```


```
SELECT * FROM ADMIN.ADDRESS
ADDRESS - (TABLE) ADMIN
ADDRESS_DETAILS - (VIEW) ADMIN
```


```
SELECT * FROM dbo
dbo - (SCHEMA) master
DBName - (DATABASE)
DB_Parts - (DATABASE)
db2 - (DATABASE)
```

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.

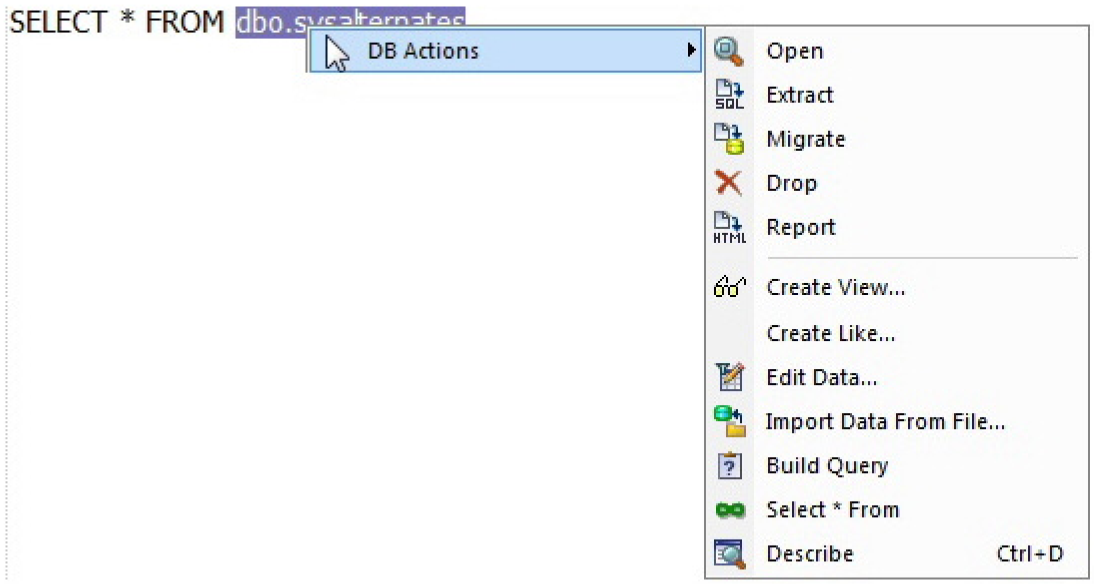
To dismiss the suggestion list without making a selection:

1. Click anywhere outside the suggestion list or press ESC.

Note: The schema information obtained 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.

Hyperlink Object Actions

Hyperlinks let you invoke object actions against object names referenced in an ISQL Editor script. A configured keystroke combination (default, CTRL+right-click) opens a menu with commands specific to the supported object type:



```
SELECT * FROM dbo.sysalternates
```

Note: In addition to the object type-specific options, the context menu also includes SQL editor options typically available when right-clicking in the editor.

The following table shows the object action availability in Rapid SQL for supported object type hyperlinks in the SQL editor:

Action (and notes)	Object type					
	Table	View	Function	Procedure	Package	Package Body

Build Query	For more information, see Query Builder	✓	✓				
Compile	For more information, see Execution and Execution-related SQL Editor options					✓	✓
Create Like		✓					
Create View		✓					
Debug	For more information, see Embarcadero SQL Debugger			✓	✓		
Describe	For more information, see The Rapid SQL Describe Window	✓	✓	✓	✓		
Drop		✓	✓	✓	✓	✓	✓
Edit Data	For more information, see Data Editor	✓					
Execute	For more information, see Execution and Execution-related SQL Editor options			✓	✓		
Extract		✓	✓	✓	✓	✓	✓
Import Data From File	For more information, see Import Data	✓					
Open	For more information, see Modifying objects using editors	✓	✓	✓	✓	✓	✓
Report		✓	✓	✓	✓	✓	✓
Select * From		✓	✓				

Before using the Hyperlinks feature, complete the following setup tasks:

- Ensure that the object types for which Hyperlink object actions are to be available, are being indexed in the datasource index. For details, see ISQL Options - Code Assist - Advanced Tab on page 214.
- Ensure that the Hyperlink feature has been enabled, the appropriate keystroke combination configured, and the menu details selected. For details, see ISQL Options - Code Assist - Main Tab on page 211.

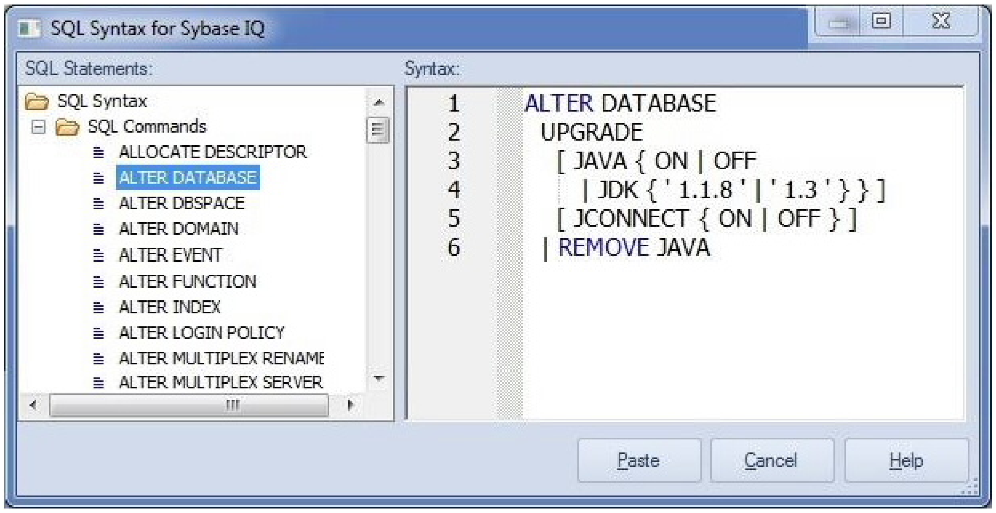
Once setup tasks are complete, the Hyperlinks feature is available in the SQL Editor.

To invoke an object action using the Hyperlink feature

1. Use the configured keystroke combination anywhere within the text of an object name reference for a supported object type.
2. Select an object action from the context menu. For more information, see the online help for the selected action.

Paste SQL Syntax

The Paste SQL Syntax facility lets you paste syntax for SQL statements, system functions calls, and other commonly used elements into an SQL Editor.



```
ALTER DATABASE
 UPGRADE
  [ JAVA { ON | OFF
    | JDK { '1.1.8' | '1.3' } }]
  [ JCONNECT { ON | OFF } ]
 | REMOVE JAVA
```

Once pasted into the SQL Editor, placeholders can be replaced with values specific to your implementation. The following table lists the categories of elements for which syntax is available for pasting on a DBMS-by-DBMS basis.

DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS	SQL SVR	SYB ASE	SYB IQ

Aggregate Functions			✓	✓			✓	✓	✓
Analytical Functions									✓
Character Functions	✓				✓				
Column Functions		✓							
Conversion Functions	✓				✓				
Datatype Conversion Functions			✓	✓			✓	✓	✓
Date/Time Functions	✓			✓	✓		✓	✓	✓
Encryption Functions				✓					
Group Functions	✓				✓				
HTTP Functions									✓
Mathematical and Number/Numeric Functions	✓			✓	✓		✓	✓	✓
Optimizer Hints					✓				
Other/Miscellaneous Functions (User, NVL, etc.)	✓				✓				✓
Scalar Functions		✓							
SQL Commands	✓	✓	✓	✓	✓		✓	✓	✓
String Functions				✓			✓	✓	✓
System Functions			✓	✓			✓	✓	✓
System Diagnostics								✓	
Text/Image Functions							✓		
Time Series Functions									✓

To paste SQL Syntax for an element into an SQL Editor

1. Open a SQL Editor against the DBMS platform you want work against. For details, see Coding Environments/Editors.
2. Place the cursor where you want to paste the SQL syntax.
3. Select Edit > Paste SQL Syntax.

4. In **SQL Statements**, select the element (such as an SQL statement or system function call) you want to paste. The syntax for the selected element is displayed in the **Syntax** pane.

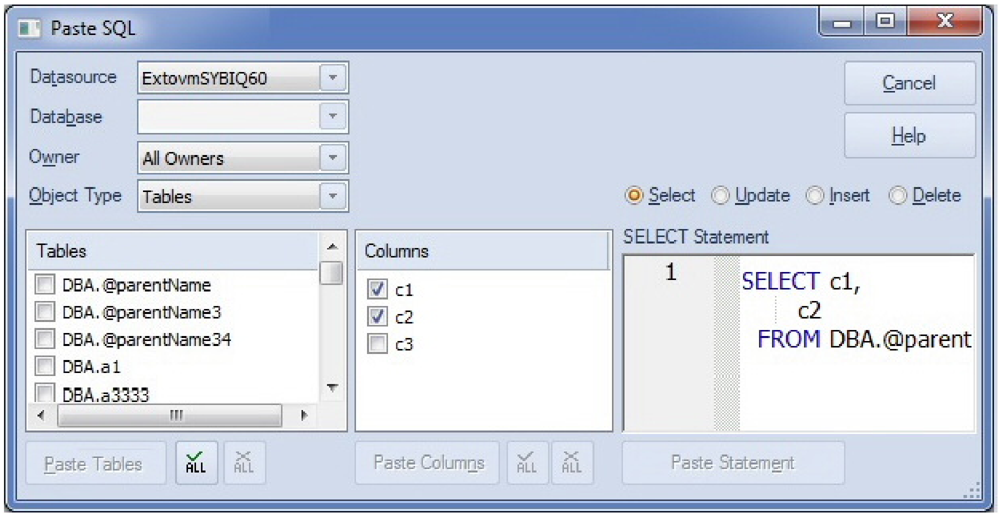
5. Click **Paste**.

Note: For information on generating and pasting complete SELECT, INSERT, UPDATE, or DELETE statements, see Paste SQL Statements.

Paste SQL Statements

The multi-purpose pasting facility offers the following:

- Build SELECT, UPDATE, INSERT, DELETE, or EXEC statements and subsequently paste into the SQL Editor. Function calls can also be generated and pasted.
- Paste comma-separated lists of defaults, functions, indexes, packages, procedures, rules, tables, triggers, user datatypes, or views into the SQL Editor. Similarly, you can paste lists of table or view column names, procedure parameter names, or function argument names into the SQL Editor.



Datasource: ExtovmSYBIQ60
Database:
Owner: All Owners
Object Type: Tables

[✓ Select] Update Insert Delete

Tables
☐ DBA.@parentName
☐ DBA.@parentName3
☐ DBA.@parentName34
☐ DBA.a1
☐ DBA.a3333

Columns
✓ c1
✓ c2
☐ c3

SELECT Statement

```
SELECT c1,
    c2
 FROM DBA.@parent
```

To generate a SELECT, INSERT, UPDATE, DELETE or EXEC statement or a function call

1. Open a SQL Editor against the DBMS platform you want to work against. For details, see Coding Environments/Editors.
2. Select Edit > Paste SQL Statement. A **Paste SQL** dialog opens.
3. Use the **Datasource**, **Database** (if available), and **Owner** controls to specify the location and ownership of the object you want to generate the statement or call from.
4. From the **Object Type** list, select **Tables** or **Views** to generate a SELECT, UPDATE, INSERT, or DELETE, statement, **Procedures** to generate an EXEC statement, or **Functions** to generate a function call.
5. If you selected an **Object Type** of **Tables** or **Views**, select one of the **Select**, **Update**, **Insert**, or **Delete** radio buttons corresponding to the type of statement you want to generate and paste.
6. In the left list (which will be labelled **Functions**, **Procedures**, **Tables**, or **Views**), select the specific object on which the generated statement or function call will be based. The middle list is populated with the columns, arguments, or parameters for the left list object you selected.

7. In the middle list, select the specific columns, arguments, or parameters that are to appear in the statement or function call.

The right list is populated with the generated statement or function call.

8. If necessary, edit the right list, to provide valid argument values for example.
9. Click **Paste Statement** (or **Paste Syntax**).

At any time that the left or middle lists are populated, you can select multiple items in that list and click the associated **Paste**... button. This pastes a comma-separated list of those elements to the SQL Editor.

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.

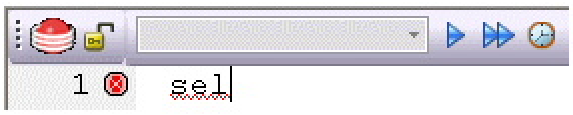
Before making use of this feature, you should be familiar with the following tasks:

- Enabling the code templates feature. For details, see Code Workbench.
- Maintaining the list of available templates.

Using Auto Replace Expressions

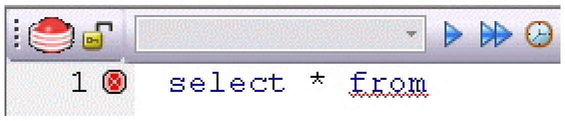
You can set up shortcuts consisting of a few characters that represent longer character strings. Instances of these Auto Replace expressions are automatically replaced by the replacement string on activation events such as typing SPACE, TAB, or RETURN. This feature is useful for creating shortcuts for 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**:



```
sel
```

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



```
select * from
```

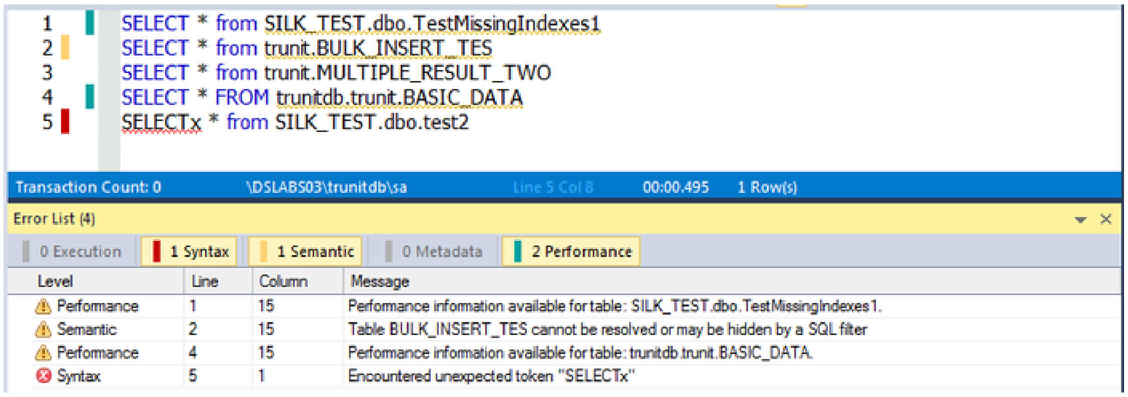
A default set of Auto Replace definitions are loaded at startup but you can also add, edit, and delete Auto Replace definitions. In addition, you can save sets of definitions to file and subsequently load specific Auto Replace definitions, allowing you to customize your templates to different platforms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

Before making use of this feature, you should be familiar with the following:

- Enabling the Auto Replace feature. For details, see Code Workbench.
- For Rapid SQL users, creating and maintaining the list of available Auto Replace definitions.

Toggling Display of the Error Pane

You can view errors currently present in an ISQL editor window in a separate pane appearing below the main editor window.



```
1  SELECT * from SILK_TEST.dbo.TestMissingIndexes1
2  SELECT * from trunit.BULK_INSERT_TES
3  SELECT * from trunit.MULTIPLE_RESULT_TWO
4  SELECT * FROM trunitdb.trunit.BASIC_DATA
5  SELECTx * from SILK_TEST.dbo.test2
```

```
Transaction Count: 0        \DSLABS03\trunitdb\sa        Line 5 Col 8        00:00.495        1 Row(s)
```

Level	Line	Column	Message
Performance	1	15	Performance information available for table: SILK_TEST.dbo.TestMissingIndexes1.
Semantic	2	15	Table BULK_INSERT_TES cannot be resolved or may be hidden by a SQL filter
Performance	4	15	Performance information available for table: trunitdb.trunit.BASIC_DATA.
Syntax	5	1	Encountered unexpected token "SELECTx"

The error pane provides a dynamic, line-by-line listing of the following error types:

- **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.
- **Syntax** - syntax problems associated with the SQL dialect of the datasource 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.
- **Semantic** - errors in specified object names. Semantic problems persist in the Error pane until they are corrected. For more information, see Semantic Validation.
- **Metadata** - If a database or schema has associated model data in Team Server 2016, is indicated in the Error Pane. See Viewing Embarcadero Team Server 2016 Model Metadata in the SQL Editor.
- **Performance** - performance information related to an specific statement written in the **ISQL Editor**. See ISQL Performance.

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**.

The window is automatically open when any error/warning is detected.

ISQL Performance

ISQL Performance provides information and statistics about the SQL statements in the SQL editor. While you are writing SQL statements in the SQL Editor, the system is analyzing the query. **ISQL Performance** generates a report according to the statistics gathered by SQL Server for tables and columns.

Two types of reports are available for each statement:

- **Report related to the statement**: Report which provides recommendations to the user for improving the system, based on the state of the database and the information gathered from previous queries.
- **Report related to the table:** Report which provides information like **Statistics**, **Overlapping Statistics**, Missing Indexes and **Unused Indexes** related to the table. This information appears in different tabs of the **Performance** window.

When any **Performance Report** is available, the information appears in two different places:

- **Error Pane**: The Error Pane add a new Performance entry to the list, in case a **Performance Report** is available. In case the two types of reports are available for the statements, the **Error Pane** adds two entries to the list.
- **Performance Window**: Includes a report for **Statistics**, **Overlapping Statistics**, Missing Indexes and **Unused Indexes** in different tabs.

Enabling ISQL Performance

In order to enable the **ISQL Performance** feature, you must:

1. Go to Tools > Options... > ISQL > Code Assist
2. Ensure that **Enable performance analysis in ISQL** is checked.

For more information, see ISQL Options.

Performance Window

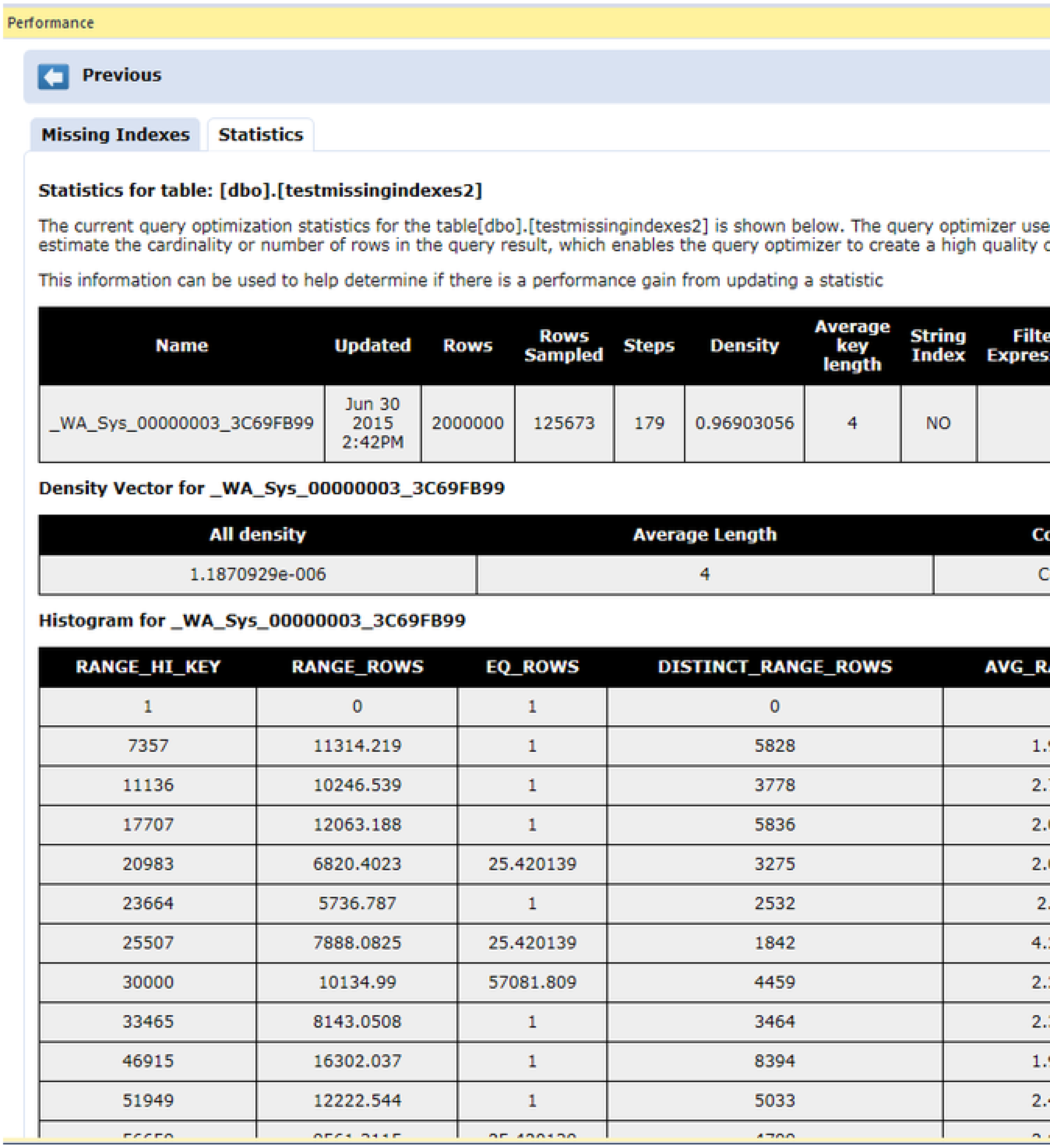
The performance window appears automatically while you are writing in the **Code Editor** in case a report is available for any of the written SQL statements. Two different types of information may be available for each SQL statement: the **Performance Window** includes an **Statistics** tab and a **Missing Indexes** tab, in case there is any missing index detected.

Performance Window also includes navigation buttons for navigating between reports when performance reports are available for more than one SQL statement.

Statistics Tab

The **Statistics** tab shows the current query optimization for the table in the query. The query optimizer uses statistics to estimate the cardinality or number of rows in the query result, which enables the query optimizer to create a high-quality query plan.

You can use this information to determine if there is a performance gain from updating a statistic.



Statistics for table: [dbo].[testmissingindexes2]

The current query optimization statistics for the table[dbo].[testmissingindexes2] is shown below. The query optimizer use
estimate the cardinality or number of rows in the query result, which enables the query optimizer to create a high quality c

This information can be used to help determine if there is a performance gain from updating a statistic

Name	Updated	Rows	Rows Sampled	Steps	Density	Average key length	String Index	Filte Expres
_WA_Sys_00000003_3C69FB99	Jun 30 2015 2:42PM	2000000	125673	179	0.96903056	4	NO	

Density Vector for _WA_Sys_00000003_3C69FB99

All density	Average Length	Co
1.1870929e-006	4	C

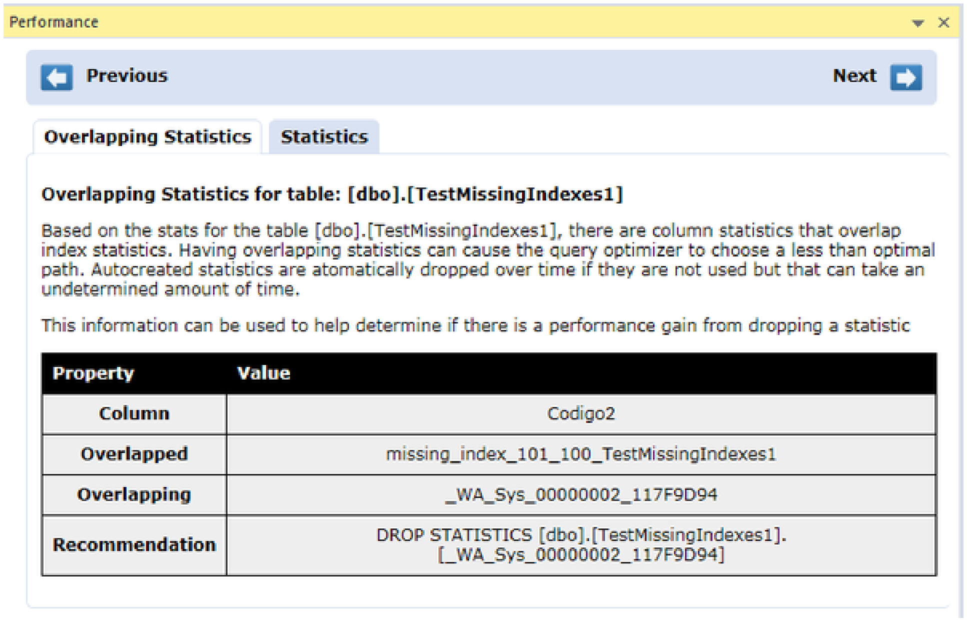
Histogram for _WA_Sys_00000003_3C69FB99

RANGE_HI_KEY	RANGE_ROWS	EQ_ROWS	DISTINCT_RANGE_ROWS	AVG_R
1	0	1	0	
7357	11314.219	1	5828	1.
11136	10246.539	1	3778	2.
17707	12063.188	1	5836	2.
20983	6820.4023	25.420139	3275	2.
23664	5736.787	1	2532	2.
25507	7888.0825	25.420139	1842	4.
30000	10134.99	57081.809	4459	2.
33465	8143.0508	1	3464	2.
46915	16302.037	1	8394	1.
51949	12222.544	1	5033	2.

Overlapping Statistics Tab

This tab appears in case there are column statistics that overlap index statistics. Having overlapping statistics can cause the query optimizer to choose a sub-optimal path. Autocreated statistics are automatically dropped over time if they are not used but that can take an undetermined amount of time.

This information helps you determine if there is a performance gain from dropping a statistic.



Overlapping Statistics for table: [dbo].[TestMissingIndexes1]

Based on the stats for the table [dbo].[TestMissingIndexes1], there are column statistics that overlap index statistics. Having overlapping statistics can cause the query optimizer to choose a less than optimal path. Autocreated statistics are atomatically dropped over time if they are not used but that can take an undetermined amount of time.

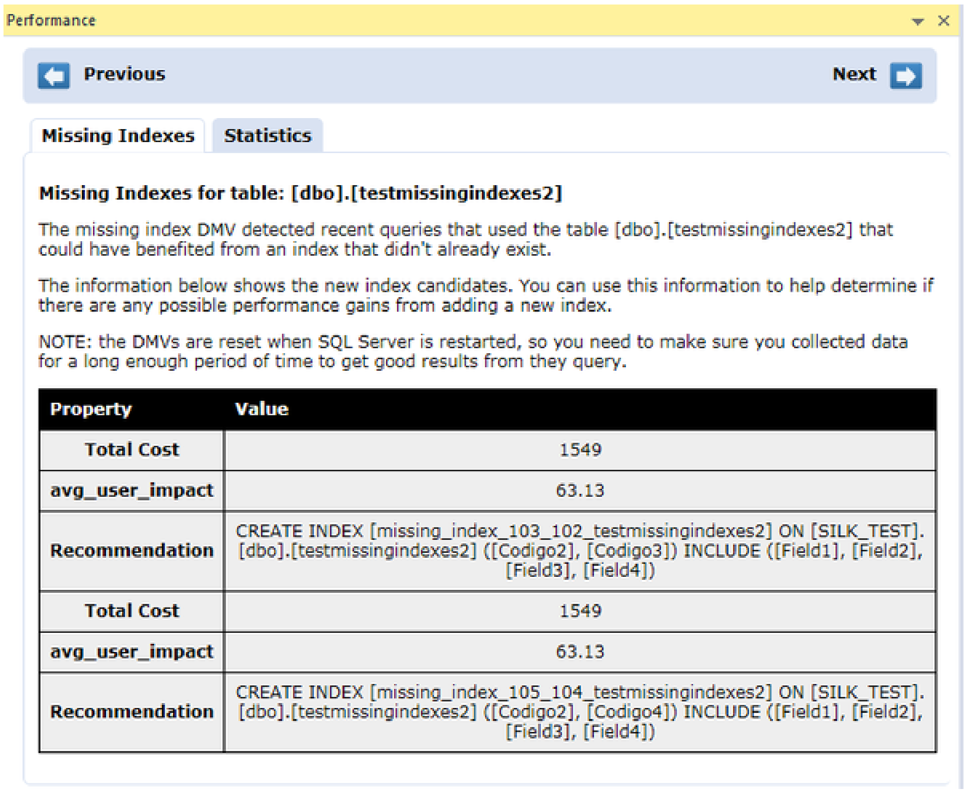
This information can be used to help determine if there is a performance gain from dropping a statistic

Property	Value
Column	Codigo2
Overlapped	missing_index_101_100_TestMissingIndexes1
Overlapping	_WA_Sys_00000002_117F9D94
Recommendation	DROP STATISTICS [dbo].[TestMissingIndexes1].[_WA_Sys_00000002_117F9D94]

Missing Indexes Tab

This tab appears when recent queries would improve the performance if a new Index was created.

The information in the tab shows the index candidates. You can use the information to determine if there are any possible performance gains from adding a new index.



Missing Indexes for table: [dbo].[testmissingindexes2]

The missing index DMV detected recent queries that used the table [dbo].[testmissingindexes2] that could have benefited from an index that didn't already exist.

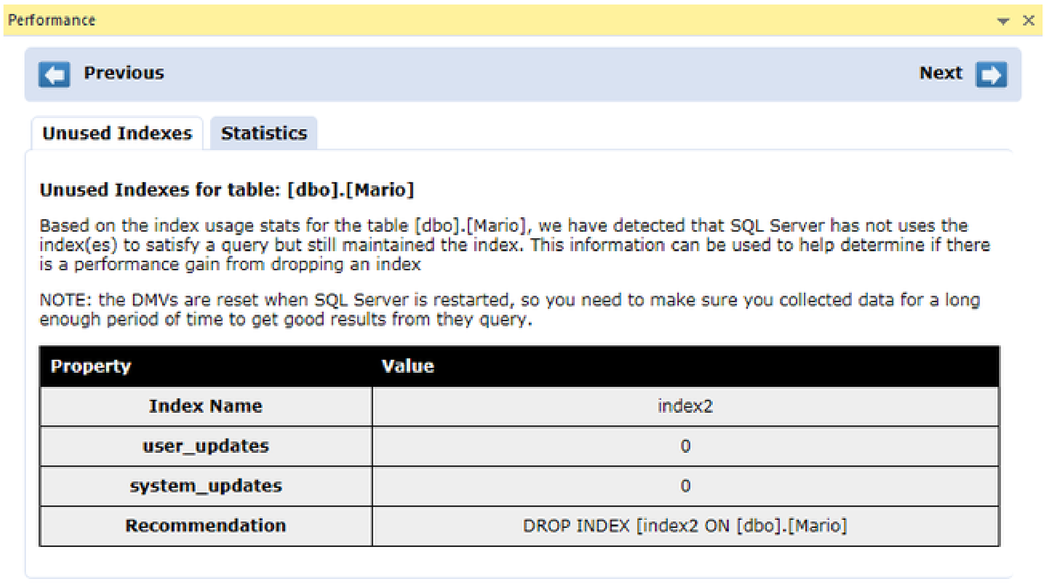
The information below shows the new index candidates. You can use this information to help determine if there are any possible performance gains from adding a new index.

NOTE: the DMVs are reset when SQL Server is restarted, so you need to make sure you collected data for a long enough period of time to get good results from they query.

Property	Value
Total Cost	1549
avg_user_impact	63.13
Recommendation	CREATE INDEX [missing_index_103_102_testmissingindexes2] ON [SILK_TEST].[dbo].[testmissingindexes2] ([Codigo2], [Codigo3]) INCLUDE ([Field1], [Field2], [Field3], [Field4])
Total Cost	1549
avg_user_impact	63.13
Recommendation	CREATE INDEX [missing_index_105_104_testmissingindexes2] ON [SILK_TEST].[dbo].[testmissingindexes2] ([Codigo2], [Codigo4]) INCLUDE ([Field1], [Field2], [Field3], [Field4])

Unused Indexes Tab

This tab appears when there are indexes defined to satisfy a query which are not used. You can use this information to determine if there is a performance gain from dropping an index.



Unused Indexes for table: [dbo].[Mario]

Based on the index usage stats for the table [dbo].[Mario], we have detected that SQL Server has not uses the index(es) to satisfy a query but still maintained the index. This information can be used to help determine if there is a performance gain from dropping an index

NOTE: the DMVs are reset when SQL Server is restarted, so you need to make sure you collected data for a long enough period of time to get good results from they query.

Property	Value
Index Name	index2
user_updates	0
system_updates	0
Recommendation	DROP INDEX [index2 ON [dbo].[Mario]

NOTE: the DMVs are reset when SQL Server is restarted, so you need to make sure you collected data for a long enough period of time to get good results from the query.

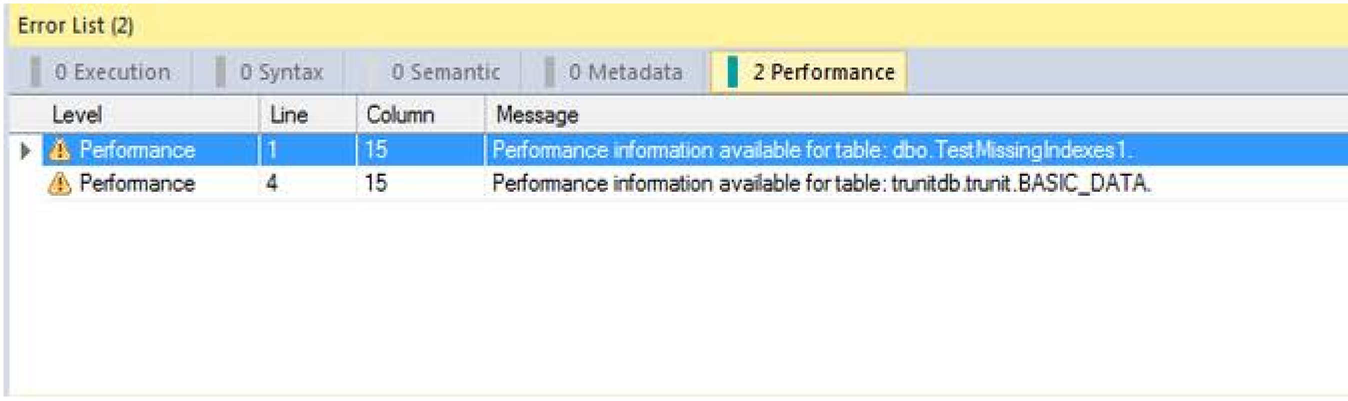
Recommendations

This tab shows recommendations about the query you are writing in order to improve the performance of the system. Recommendations may be:

- o Misuses of `DEALLOCATE`
- o Avoiding "Work tables" when "table variables" can be used.
- o Improvements on a SQL statement to improve the performance
- o ...

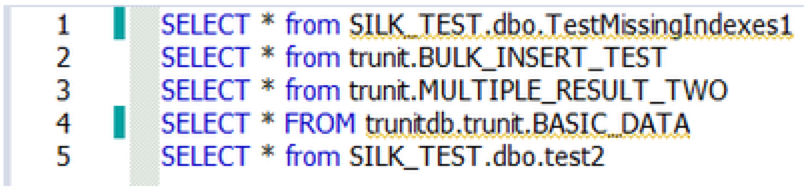
Error Pane

When **ISQL Performance** creates a new report, it includes an entry in the Error Pane to inform the user about the availability of that new report. As you can see in the next figure, the Error Pane includes a **Performance** tab where information about the available performance information is indicated.



Level	Line	Column	Message
Performance	1	15	Performance information available for table: dbo.TestMissingIndexes1.
Performance	4	15	Performance information available for table: trunitdb.trunit.BASIC_DATA.

The **Performance** tab indicator is green which is also added to the **ISQL Editor** gutter, indicating which SQL statements have a specific performance report.



```
SELECT * from SILK_TEST.dbo.TestMissingIndexes1
SELECT * from trunit.BULK_INSERT_TEST
SELECT * from trunit.MULTIPLE_RESULT_TWO
SELECT * FROM trunitdb.trunit.BASIC_DATA
SELECT * from SILK_TEST.dbo.test2
```

Execution and Execution-related SQL Editor options

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.

The following topic describe the execution related options available from the SQL Editor.

- o Locking a SQL Editor Window to a Datasource
- o Setting up the Execution Environment with Query options
- o Executing Entire Scripts in the SQL Editor
- o Executing Partial Scripts in the SQL Editor
- o Executing the Current Statement
- o Step Executing a Script
- o Using the Query Plan Facility
- o Canceling a Query

See Also

- o Script/File Execution Facilities.

Locking a SQL Editor Window to a Datasource

When executing scripts or setting query options, all directives are issued to the currently active datasource. Since the selected datasource can be changed, locking the SQL Editor window to a datasource ensure that all directives are issued to the correct datasource and that sessions are set up to terminate properly.

Note: For information on the ways in which the selected datasource can change, see Connected/Selected Datasource options.

A SQL Editor window's locked/unlocked status is shown by the **Lock - Unlock Connection** button on the SQL Editor Toolbar. For information on the toolbar, see SQL Editor Windows Basics.

Lock - Unlock Connection button state	Description
🔒 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.

To toggle the datasource locked/unlocked status of a SQL Editor window

1. Click the **Lock - Unlock Connection** button.

Setting up the Execution Environment with Query options

A set of options corresponding to DBMS-specific query options are available. 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.

See the following topics for descriptions of the options available for the DBMS you are working against:

- DB2 LUW Query options
- DB2 z/OS Query options
- MySQL Query options
- Oracle Query options
- SQL Server Query options
- Sybase ASE Query options
- Sybase IQ Query options
- Teradata Query options

If you do not change the default settings, a set of default query option settings are sent 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.

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 SQL 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.
- 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 ISQL Options.

- 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 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 **Setting up the Execution Environment with Query options**.

When query options are sent to the server

Locked connections optimize sending query options to the server. 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 SQL Editor Toolbar Options.
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. These rules apply as long as the connection remains locked.

DB2 LUW Query options

The following settings are available when setting query options against a DB2 LUW datasource:

Query Option	Description
Batch Delimiter	The batch separator must be a viewable character and not a space, new line, or tab. The default (semi-colon) has 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.
Check syntax when executing	TRUE/FALSE
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.
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, those tables will continue to be used. 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.
Max Errors Before Aborting	Select the maximum number of errors encountered before a script is aborted. Setting this value to zero disables the feature.
Row Count	When set to TRUE, a query is tereminated after returning the specified number of rows.
Run Script with batch execution	TRUE/FALSE
Isolation Level	Lets you set DB2 Isolation Levels of UNCOMMITED READ, RESET, CURSOR STABILITY, REPEATABLE READ, or READ STABILITY.
Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

DB2 z/OS Query options

The following settings are available when setting query options against a DB2 z/OS datasource:

Query Option	Description
Batch Delimiter	The batch separator must be a viewable character and not a space, new line, or tab. The default (semi-colon) has 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.
Check syntax when executing	TRUE/FALSE
Max Errors Before Aborting	Select the maximum number of errors encountered before a script is aborted. Setting this value to zero disables the feature.
Row Count	When set to TRUE, a query is terminated after returning the specified number of rows.
Run Script with batch execution	TRUE/FALSE
Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

MySQL Query options

The following settings are available when setting query options against a MySQL datasource:

Query Option	Description
Batch Delimiter	The batch separator must be a viewable character and not a space, new line, or tab. The default (semi-colon) has 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.
Big Tables	When set to TRUE, allows big result sets by saving all temporary sets to file. This can slow queries.
Client Character	Default character set.
Insert ID	Choose a value to be used a following INSERT or ALTER TABLE statement when you supply an AUTO_INCREMENT value.
Interactive Timeout	28800 default
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.
Max Errors Before Aborting	Select the maximum number of errors encountered before a script is aborted. Setting this value to zero disables the feature.
Query Cache Type	The query is cached for ON or DEMAND.
Row Count	When set to TRUE, a query terminates after returning the specified number of rows.
SET Other Variables	Variables to be set at runtime.
SQL Auto IS NULL	When set to TRUE, enables you to find the last inserted row for a table.
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)
SQL Big Tables	TRUE/FALSE
SQL Buffer Result	When set to TRUE, forces results from SELECT statements into temporary tables.
SQL Log Bin	When set to TRUE, allows logging to the binary log.
SQL Log Off	When set to TRUE, no logging is done to the general query log.

SQL Log Update	When set to TRUE, allows logging to the binary log.
SQL Low Priority Updates	When set to TRUE, gives table modifying operations lower priority than SELECT operations.
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.
SQL Quote Show Create	When set to TRUE, table and column names will be quoted.
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
SQL Select	The maximum number of records that should be returned from SELECT statements.
SQL Warnings	Defines whether or not single row insert statements generate an information string in the event of a warning.
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.
Unique Checks	Performs uniqueness checks for secondary indexes of MyISAM tables.
Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

Oracle Query options

The following settings are available when setting query options against an Oracle datasource:

Category	Query Option	Description
Query Analysis	**Network Activity**	True/False
	Execution Information	True/False
	I/O Activity	True/False
	Sort Activity	TRUE/FALSE

	Index Activity	True/False
	Parse Activity	TRUE/FALSE
	Table Activity	True/False
	OS Activity	TRUE/FALSE
	Wait Activity	True/False
Miscellaneous	Batch Delimiter	The batch separator must be a viewable character and not a space, new line, or tab. The default ("/") has 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.
	Check syntax when executing	TRUE/FALSE
	LONG Size Bytes	8,192 is the default
	Max Errors Before Aborting	Select the maximum number of errors encountered before a script is aborted. Setting this value to zero disables the feature.
	Row Count	When set to TRUE, a query terminates after returning the specified number of rows.
	Run Script with batch execution	TRUE/FALSE
	Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

SQL Server Query options

The following settings are available when setting query options against a SQL Server datasource:

Category	Query Option	Description
Query Analysis	**Show Plan**	When set to TRUE, reports data retrieval methods chosen by the Microsoft SQL Server query optimizer.
	No Count	Terminates the message indicating the number of rows affected by a Transact-SQL statement from being returned as part of the results.
	No Exec	When set to TRUE, compiles each query without executing it.
	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.
	Statistics I/O	Lets you display information regarding the amount of disk activity generated by Transact-SQL statements.
	Statistics Time	Displays the number of milliseconds required to parse, compile, and execute each statement.
	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.
Arithmetic	**Ignore Overflow**	When set to TRUE, any overflow caused by a resulting value that is larger than a column's specified size is ignored.
	Abort On Overflow	If set to TRUE, queries will be aborted on encountering a value greater than the specified size.

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**, **IMPLICIT_TRANSACTIONS**, and **CONCAT_NULL_YIELDS_NULL**. The initial default values are hard-coded, not obtained from server settings.	
Transactions	**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.
Miscellaneous	**Batch Delimiter**	The batch separator must be a viewable character and not a space, new line, or tab. The default ("go") has 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.
	Max Errors Before Aborting	Select the maximum number of errors encountered before a script is aborted. Setting this value to zero disables the feature.
	Row Count	When set to TRUE, a query terminates after returning the specified number of rows.
	Run Script with batch execution	TRUE/FALSE
	Text Size	8,192 is the default

Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

Sybase ASE Query options

The following settings are available when setting query options against a Sybase ASE datasource:

Category	Query Option	Description
Query Analysis	**Show Plan**	When set to TRUE, reports data retrieval methods chosen by the Microsoft SQL Server query optimizer.
	No Count	Terminates the message indicating the number of rows affected by a Transact-SQL statement from being returned as part of the results.
	No Exec	When set to TRUE, compiles each query without executing it.
	Statistics I/O	Lets you display information regarding the amount of disk activity generated by Transact-SQL statements.
	Statistics Time	Displays the number of milliseconds required to parse, compile, and execute each statement.
	Statistics Subquery Cache	Displays the number of cache hits, misses, and the number of rows in the subquery cache for each subquery.
	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.
	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.
	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.
Transactions	**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.
	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.

Arithmetic	**Ignore Overflow**	When set to TRUE, any overflow caused by a resulting value that is larger than a column’s specified size is ignored.
	Abort On Overflow	If set to TRUE, queries will be aborted on encountering a value greater than the specified size.
	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, query results are truncated processing continues.
DBCC Traceflags	**Index Selection**	Valuable when tuning query performance.
	Join Selection	Valuable when tuning query performance.
	Output to Execution Window	TRUE/FALSE
	Output to Server Error Log	TRUE/FALSE
Miscellaneous	**ANSI NULL**	When set to TRUE, controls results of logical operations with NULL values.
	Set Quoted Identifier	TRUE/FALSE
	Batch Delimiter	The batch separator must be a viewable character and not a space, new line, or tab. The default (“go”) has 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.
	Row Count	When set to TRUE, a query terminates after returning the specified number of rows.
	Table Count	Sets the number of tables that Sybase ASE considers at one time while optimizing a join.
	Text Size	8,192 is the default
	Max Errors Before Aborting	Select the maximum number of errors encountered before a script is aborted. Setting this value to zero disables the feature.
	Run Script with batch execution	TRUE/FALSE

Text Formatting (Sybase ASE 15)	These options let you generate diagnostic output in text format, using the Sybase ASE **set option** command. To use these options, you must be using the **sa_role** or have the **set tracing** permission. You can select ON, OFF BRIEF, or LONG standard Sybase ASE settings. If you are not using the **sa_role** or do not have the **set tracing** permission, use only the DEFAULT selection for each **Text Formatting** Set option is available. That setting has no effect. Using other selections without the required role or permission will result in an error on the next execution. For detailed information on these options and required permissions or roles, see the Sybase documentation. For access, see Accessing Third Party Documentation. Short descriptions message content displayed for Text Formatting **set option** command options are:

Show	A collection of details, where the collection depends on the choice of BRIEF, LONG, ON, or OFF.
Show Lop	Logical operators used.
Show Managers	Data structure managers used during optimization.
Show Props	Logical properties evaluated.
Show Parallel	Details of parallel query optimization.
Show Histograms	Processing of histograms associated with SARG/join columns.
Show Abstract Plan	Details of an abstract plan.
Show Search Engine	Details of the join-ordering algorithm.
Show Counters	Optimization counters.
Show Best Plan	Details of the best query plan selected by the optimizer.
Show Code Gen	Details of code generation.
Show Pio Costing	Estimates ofreads/writes from/to disk.
Show Lio Costing	Estimates of reads/writes from/to memory.
Show Pll Costing	Estimates relating to costing for parallel execution.
Show Elimination	Partition elimination.

Show Missing Stats	Details of useful statistics missing from SARG/join columns.
Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

Sybase IQ Query options

The following settings are available when setting query options against a Sybase IQ datasource:

Query Option	Description
Send SET Options	The **Send Set Options** setting applies to all settings except those in the bottom-most **Miscellaneous** category. It dictates whether the child options are sent to the server. The default for this option is set using the **Enable SET query option** setting on the ISQL tab of the Options editor. For details, see ISQL Options.
Isolation Level	Specifies the locking isolation level for Catalog Store tables.
Allow Nulls By Default	Sets the ALLOW_NULLS_BY_DEFAULT option, controling whether new columns created without specifying either NULL/NOT NULL are allowed to contain NULL values.
ANSI Null	Sets the ANSINULL option, controlling the interpretation of using = and != with NULL.
Conversion Error	Sets the CONVERSION_ERROR option, controlling reporting of data type conversion failures on fetching information.
Divide-by-zero Error	Sets the DIVIDE_BY_ZERO_ERROR option, controlling reporting of division by zero.
SET quoted_identifier	Sets the QUOTED_IDENTIFIER option, controlling the interpretation of strings enclosed in double quotes.
SQL Flagger Error Level	Sets the SQL_FLAGGER_ERROR_LEVEL option, controlling the behavior in response to any SQL code that is not part of the specified standard.
SQL Flagger Warning Level	Sets the SQL_FLAGGER_WARNING_LEVEL option, controlling the response to any SQL that is not part of the specified standard.
String R-Truncation	Sets the STRING_RTRUNCATION option, determining whether an error is raised when an INSERT or UPDATE truncates a CHAR or VARCHAR string.
T-SQL Variables	Sets the TSQL_VARIABLES option, controlling whether the **@** sign can be used as a prefix for Embedded SQL host variable names.
Disable RI Check	Sets the DISABLE_RI_CHECK option, allowing LOAD, INSERT, UPDATE, or DELETE operations to bypass the referential integrity check to improve performance.
Early Predicate Execution	Sets the EARLY_PREDICATE_EXECUTION option, controlling whether simple local predicates are executed before query optimization.
Extended Join Syntax	Sets the EXTENDED_JOIN_SYNTAX option, controlling whether queries with an ambiguous syntax for multi-table joins are allowed or reported as an error.
Index Advisor	Sets the INDEX_ADVISOR option, generateing messages suggesting additional column indexes that may improve performance of queries.
Index Advisor Max Rows	Sets the INDEX_ADVISOR_MAX_ROWS option, setting the maximum number of unique advice messages stored by the index advisor.

Infer Subquery Predicates	Sets the INFER_SUBQUERY_PREDICATES option, controlling the optimizer's inference of additional subquery predicates.
Join Optimization	Sets the the JOIN_OPTIMIZATION option, enabling or disabling the optimization of the join order.
Large Doubles Accumulator	Sets the LARGE_DOUBLES_ACCUMULATOR option, controlling which accumulator to use for SUM or AVG of floating-point numbers.
Minimize Storage	Sets the MINIMIZE_STORAGE option, minimizing use of disk space for newly created columns.
No Exec	Sets the NOEXEC option, generating the optimizer query plans instead of executing the plan.
Non ANSI NULL Varchar	Sets the NON_ANSI_NULL_VARCHAR option, controlling whether zero-length VARCHAR data is treated as NULLs for INSERT, ILOAD, and UPDATE operations.
On Charset Conversion Error	Sets the ON_CHARSET_CONVERSION_FAILURE option, controlling the action taken on a character conversion error.
Query Name	Sets the QUERY_NAME option, giving a name to an executed query in its query plan.
Query Detail	Sets the QUERY_DETAIL option, specifying whether additional query information is included in the Query Detail section of the query plan.
Query Plan	Sets the QUERY_PLAN option, specifying whether or not additional query plans are printed to the message file.
Query Plan After Run	Sets the QUERY_PLAN_AFTER_RUN option, printing the entire query plan after query execution is complete.
Query Plan as HTML	Sets the QUERY_PLAN_AS_HTML option, generating graphical query plans in HTML format.
Query Plan as HTML Directory	Sets the QUERY_PLAN_AS_HTML_DIRECTORY option, specifying the directory into which Sybase IQ writes HTML query plans.
Query Rows Returned Limit	Sets the QUERY_ROWS_RETURNED_LIMIT option, setting the row threshold for rejecting queries based on an estimated size of the result set.
Query TempSpace Limit	Sets the QUERY_TEMP_SPACE_LIMIT option, specifying the maximum estimated amount of temp space before rejecting a query.
Query Timing	Sets the QUERY_TIMING option, determining whether specific timing statistics are collected and displayed.
Batch Delimiter	A delimiting character used to separate multiple SQL statements within a statement batch.
Row Count	Lets you limit the number of rows returned by a query.

Max Errors Before Aborting	Lets you limit the number of errors returned before aborting a query.
Run Script With Batch Execution	When selected, scripts are executed statement by statement (batch delimited).
Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

Teradata Query options

The following settings are available when setting query options against a Teradata datasource:

Query Option	Description
Batch Delimiter	A delimiting character used to separate multiple SQL statements within a statement batch.
Row Count	Lets you limit the number of rows returned by a query.
Max Errors Before Aborting	Lets you limit the number of errors returned before aborting a query.
Run Script With Batch Execution	When selected, scripts are executed statement by statement (batch delimited).
Prepare Batch	For the current SQL Editor window, setting this check box enables bind variable parameterization. This lets you provide named and unnamed variable values as a script executes. For details, see Bind Variable Parameterization in Prepared Statements (Dynamic SQL).

Executing Entire Scripts in the SQL Editor

You can execute the contents of an SQL editor window.

To execute an entire script

1. On the **Query** menu, select **Execute**.

The script starts to execute. If there are results, a **Results** tab opens letting you work with the results. For details, see Using the Results Editor.

Executing Partial Scripts in the SQL Editor

You can execute a selected portion of a script within an SQL editor window.

To execute a partial script

1. Ensure that partial script execution is enabled. For details, see ISQL Options - Main Tab.
2. Select the portion of the script to be executed.
3. On the **Query** menu, select **Execute**.

The selected statements execute. If there are results, a **Results** tab opens letting you work with the results. For details, see Using the Results Editor.

Executing the Current Statement

The **Execute Statement** command lets you execute an individual statement where statements in the script are terminated with a valid delimiter (**;** or **/**).

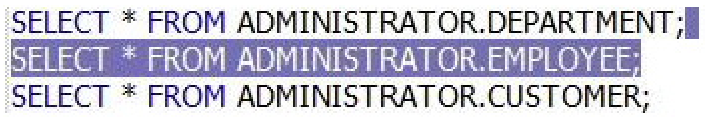
Note: The **Execute Statement** command is only available if the **Scan entire file for context** option in the Options Editor is selected. For details, see ISQL Options - Code Assist - Advanced Tab.

To execute a delimiter-terminated statement

1. Place the cursor anywhere within the statement.
2. From the **Query** menu, select **Execute Statement**.

The statement is executed and if results are returned, the Results Editor opens. For details, see Using the Results Editor.

As a visual aid, the executed statement is highlighted.



```
SELECT * FROM ADMINISTRATOR.DEPARTMENT;
SELECT * FROM ADMINISTRATOR.EMPLOYEE;
SELECT * FROM ADMINISTRATOR.CUSTOMER;
```

Topics

- Execution and Execution-related SQL Editor options.

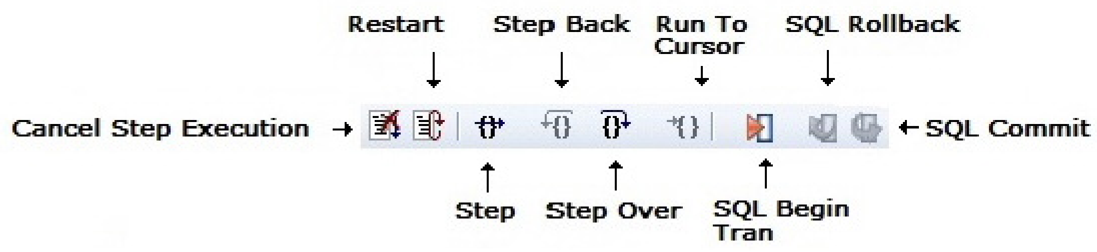
Step Executing a Script

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. The Step Execution facility parses the statements and moves from batch to batch during the step execution process, each step controlled by you clicking the step execution button.

To use the Step Execute Facility

1. Open a script. For details, see Opening Files.
2. On the **Query** menu, select **Step-Execute**.

The procedure starts to execute and displays errors at the bottom of the Editor window. A toolbar offering subsequent step execution actions opens:



Note: Each executing line is indicated 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, the arrow moves to indicate your current location.

3. Use the following table as a guide to subsequent actions you take:

Feature	
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 i**con 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.
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.

Keep the following in mind when using this feature:

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, errors are flagged with a red dot in the ISQL window gutter. The errors are flagged sequentially as they are stepped into.
Point and Click Error Navigation	Errors are displayed in the output window at the bottom of the screen and selects the errors as they occur. You can click each error and be taken directly to the line where that error occurred.

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 ASE)
- Viewing a Tree-based or Graphical Query Plan (Oracle)

Viewing a Tree-based Query Plan (IBM DB2 for LUW, SQL Server, Sybase ASE)

For IBM DB2 for Windows, Unix, and Linux, Microsoft SQL Server, and Sybase ASE, you can 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.

The 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, a tree view of statements and associated costs is included.

To view a tree-based representation of a query plan

1. Open a script.
2. On the **Query** menu, select **Query Plan.**

The Show Plan mode starts.

3. To generate the Show Plan in a separate result window, click **Execute**.

Viewing a Tree-based or Graphical Query Plan (Oracle)

For Oracle, you can 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.

Note: The Options editor has a **Default Query Plan** setting that controls whether the default query plan display is tree-based or graphical. For details, see ISQL 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, select **Query Plan**.

The Show Plan mode starts.

3. To generate the Show Plan in a separate result window, click **Execute**.

Options when working with the graphical Query Plan view 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

Canceling a Query

The ISQL Editor lets you cancel a query while the rows are still being returned.

To Cancel a Query

1. On the Editor window tool bar, click **Cancel**. For more information, see SQL Editor Toolbar Options.

Note: This button is only enabled after a script has begun executing.

Using the 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, REF CURSOR contents are displayed 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.

Topics

Results Window Toolbar Options

The table below describes the options of the Results window toolbar:

Option	Description
Lock	Lets you lock an ISQL Window to a specific datasource connection. When locked, the ISQL Window does not respond to datasource change events in other windows, and only uses the current connection for subsequent execution.
Options	Open the Options editor, letting you specify results window preferences. For more information, see Results (ISQL) Options.
ReExecute	Lets you execute the script again without switching to the Query tab.
Close	Lets you close the current results window.

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.

To configure result set windows

1. On the **File** menu, select **Options**.
2. In the Options Editor, click the list, and then click **Results** or click the **Results** tab.
3. Refer to Results (ISQL) Options for details on how to set result windows options.

Configuring Results Window Grid Properties

You can alter the physical appearance in a results set window. Effects include items such as 3-D buttons, highlighted headings, and enabled or disabled row and column gridlines.

To configure the appearance of result set grids

1. On the **File** menu, select **Options**.
2. In the Options Editor, click the list, and then click **Grid Properties** or click the **Grid Properties** tab.
3. Refer to Grid Properties (Results window) Options for details on how to set result windows options.

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
- Pipe delimited ('|") files
- HTML
- Excel 2000 or higher
- XML
- User-specified delimited files

Topics

- Using the 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, select **Read Only**.

A check-mark is placed next to read only.

Note: The Read Only command is a toggle. When it is set, the formatting buttons on the **Edit** menu are not available.

Topics

- Using the 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.

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:
2. On the **File** menu, select **Send**.

The open **Message** dialog opens.

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**.

The result set is sent to the specified addressee(s).

Topics

- o Using the Results 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 Execution and Execution-related SQL Editor options

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**.

The **Rename Tab** dialog opens.

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**.

The name is changed and the **Rename Tab** dialog closes. 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 SQL window, right-click the **Query** tab, and then click **Close** or **Close All**.

The Query closes.

Closing Result Window Tabs

You can close tabbed Result set windows.

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**.

The **Result Window** tab closes.

Topics

- Using the 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

1. On the **File** menu, select **Save**. The **Save Results** dialog opens.
2. In the **File name** box, type the name of the result set.
3. From the **Save as type** dropdown, select a file type of **Excel 2000 or later .xls**, t**ab-delimited**, **comma-delimited**, **pipe-delimited**, **user-specified delimited .txt**, **HTML**, or **XML**.
4. From the **Encoding** dropdown, select an encoding scheme of ANSI, UNICODE, UTF-16LE (No BOM), UTF-16 Big Endian, UTF-16BE (No BOM), UTF-8, UTF-8 (No BOM), UTF-32 Little Endian, UTF-32LE (No BOM), UTF-32 Big Endian, or UTF-32BE (No BOM).
5. To include column titles, select **Include column titles when saving**.
6. If you chose **User Specified Delimiter**, in **User Specified Delimiter** type the delimiter.
7. Click **Save**.

Closing a Result Set

To close a result set, do the following:

1. On the **Main** menu, click **Close**.

The Result Set closes.

2. If you have not saved your result set, you are prompted to save the file. Click **Yes** to save and **No** to close without saving.

Topics

- Using the 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

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, select **Cut**.

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, select **Copy**.

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, select **Paste**.

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, select **Cut**.

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, select **Copy**.

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, select **Paste**.

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, select **Cut**.

Copying Columns

To copy columns, do the following:

1. In the Results window, click the column heading.
2. On the **Edit** menu, select **Copy**.

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, select **Paste**.

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, select **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, select **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, select **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, select **Insert Row**.

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.

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, select **Delete 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, select **Delete Column**.

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.

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. Double click the column header 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.

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.

- Changing the Displayed View
- Format Border Styles
- Format Font Styles
- Format Color Styles

For information on general appearance of results, see Configuring Results Window Grid Properties.

Changing the Displayed View

You can view results in a standard grid, in HTML format, or as flat ASCII. By default, when a result window is generated, the results are displayed according to the view option currently selected on the Results (ISQL) Options. You can change the display in the results window.

To change the result view on the fly:

1. Right-click in the results window.
2. On the shortcut menu, choose one of the **View as Grid**, **View as HTML**, or **View as ASCII** options.

Format Border Styles

You can use the shortcut menu to alter border properties.

Completing the Format Styles Dialog Box

To complete the **Format Styles** dialog, do the following:

1. Right-click the **Result** data grid, and then click **Border**.
2. 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.
3. 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. To select the **Range** property, click the range box.

4. In the **Type** box, you can select the type of line you want to border the cell or cells by clicking the corresponding boxes.
5. To select a color, click the **Color** list, and then click the border color.
6. Click **OK**.

Changes are saved and the **Format Styles** dialog closes.

Format Font Styles

You can use the shortcut menu to alter font properties. Selecting the Font command displays the **Format Styles** tabbed dialog.

Completing the Format Styles Dialog Box

To complete the **Format Styles** dialog, do the following:

1. Right-click the **Result** data grid, and then click **Font**.

The **Format Styles** dialog opens.

2. In the **Font** box, type or click the font you want to use.
3. In the **Style** box, type or click the font style you want to use.
4. In the **Size** box, type or click the size you want to use.
5. To make a line cross through the length of the text, in the **Effects** box, select the **Strikeout** check box.
6. To underline the text, in the **Effects** box, select the **Underline** check box.
7. To change the script style, click the **Script** list, and then click the new script style.

The Sample box displays the sample text of your selections

Format Color Styles

You can use the shortcut menu to alter color properties. Selecting the Color command displays the **Cell Properties** dialog.

Completing the Cell Properties Dialog Box

To complete the **Format Styles** dialog, do the following:

1. Right-click the **Result** data grid, and then click **Color**.

The **Cell Properties** dialog opens.

2. Change the **Text Color** and **Background Color** options.

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 Unicode characters is provided as follows:
- IBM DB2 for Windows, Unix, and Linux V8 and V9: **character**, **clob**, **varchar**, and **longvarchar** types
- SQL Server 2005: **nchar**, **nvarchar**, **ntext**, and **nvarchar(max)** types
- Oracle 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 ASE 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

Miscellaneous SQL Editor Tasks

The following supplementary function is available from the SQL Editor:

Emailing 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

1. Open a SQL Editor and open, paste, or type a script. For more information, see Using the SQL Editor.
2. On the **File** menu, select **Send**. Your E-mail application opens and automatically attaches a file containing the script.
3. In the **Address** box, type the name of the addressee(s) and when ready click **Send**.

Using the DDL Editor

You use the 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.

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. For more information, see the DBMS platform-specific topics under DDL Extract Options.

To create an object using the DDL EDitor

1. On the **File** menu, select New > DDL Editor.
2. Use the following table as a guide to completing the dialog:

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.

1. Click **OK**.

The DDL opens in the SQL Editor. A subset of SQL Editor functionality becomes available. For more information, see Using the SQL Editor.

Project Management

The following project management-oriented functions are available:

- Project support
- Version Control

Project support

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. Version control functions and build management facilities are also incorporated and help you manage and build projects. Once a project has been created, you can:

- 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. 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 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. The Project Management facilities let you effectively monitor revisions for your database servers.

Topics

- Create a New Project
- Working With Projects

Create a New Project

Step-by-step wizards guide you through the process of creating a new project. You have a variety of options to choose from when you create a new project. You can 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.

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 Project - 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.

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**.

The New Project Dialog Box opens.

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

A context-sensitive wizard reverse-engineers all or part of an existing database. It lets you create a project containing files based on the relevant object types. A separate file is created 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.

Note: The project script file build order is automatically discovered and set by referencing the system catalog to determine dependencies.

To start a new project from a database

1. Select **File > New > Project**. The **New Project** dialog box opens.
2. Complete the New Project dialog box, ensuring you select the **From Database** option, and then click **OK**. The **Reverse Engineer Wizard** opens.
3. Use the following table as a guide to understanding and setting options in the wizard:

Panel	Tasks and Settings	Description
Connection	**Source Datasource**	Lets you select the datasource for which you want to start a project..
Catalogs (only displayed for multiple database DBMS datasources)	**Source Database**	Lets you select one or more databases for which you want to start a project..
Object Selection	**Schemas**	Opens a dialog that lets you narrow the choice of candidate objects by selecting only those associated with specified schema/owners.
	Object Types	Lets you select the object types that will be extracted. You can use the expand/collapse icons to hide and show the dependent object types for each object. As you select object types, all objects of that type (that satisfy the **Schemas** criteria) are made available for selection in the **Objects** list. Selected dependent object types are also made available in the **Objects** list. For information on setting the dependent object type automatically selected to be extracted along with each object type, see DDL Extract Options.
	Objects	Lets you select the specific objects that will be extracted. For each object selected, you can also select the specific dependent objects that are to be extracted.

Options	This panel lets you choose extraction options and view an **Example Preview** script that is updated as you choose options. It also lets you save your options choices as a template, with an option to use the saved template as the default. Click **Finish** when ready to proceed with the extraction.
Execute	Displays the status of the extraction with details on the number of objects retrieved, the number of errors detected, and the elapsed time. On completion, the schema definition is stored in the file you specified on the previous panel. Filter options (Show All, Show Errors, and Show Warnings) let you display specific types of messages generated during the extraction process. When the extraction completes, you can also use the following options:

Continue	Adds the individual files for each obect to the project.
Report	Opens a detailed report on the results of the schema extraction.

Create a New Project From Existing Files

When creating a new project, you can start by adding files of the following types:

- Scripts (.sql)
- DDL (.ddl)
- Procedures (.prc, .pro)
- ER/Studio files (.dml)
- Queries (.qry)
- Text files (.txt)
- VB Scrpt files (.vbs)
- JScript files (.js)
- Web files files (.html, .htm, .asp)

To create a new project from an existing file, do the following:

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

The **New Project** dialog box opens.

2. Complete the New Project dialog box and select the **From Existing Files** option.
3. Click **OK**.

The **Add File(s) to Project** dialog box opens.

4. Use the dialog to locate the directory containing the files, and select one or more files to add to the project.
5. When you finish, click **Open**.

Topics

- Add File(s) to a Project.

Create a New Project From a Version Control Project

You can 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.

You can slsocreate a project from a version control project that contains sub-directories, while including files from those sub-directories in your 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**.

The **New Project** dialog box opens.

2. Complete the New Project dialog box and select **From Version Control Project**</span>.
3. Click **OK**.

The **Choose project from (Version Control name)** dialog box opens.

4. Select the project or files you want to include, and then click **OK**.

The **Files to be included in (project path)** dialog box opens.

5. Select the project of files you want to include, and then click **OK**.

Create a New Project Without Initialization

You can simply create a project with any further initialization. You can subsequently use the following topics to add additional resources:

- 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**.

The **New Project** dialog box opens.

2. Complete the New Project dialog box and select the **Do Not Initialize** option.
3. Click **OK**.

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 subfolder under another project, and nest subfolders. Once you have created a new project, use the following topics to help you maintain and modify the project.

Topics

Opening an Existing Project

To open an existing project, do the following:

1. On the **File** menu click **Open Project**.

The **Open Project** dialog box opens.

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**.

The **Project Tab** containing the project opens.

Topics

- Working with Projects.

Opening a Recent Project

To open a recent project, do the following:

1. On the **File** menu select **Recent Projects,** and then select a project.

The **Project Tab** containing the project opens.

Topics

- 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.

Topics

- Working with Projects.

Build Project

The Build Project dialog box lets you:

- o Generate a project build script and display it in a SQL window.
- o Execute the project build immediately on build target.
- o Schedule a project build.

The table below describes the Build Project dialog box:

Option	Description
Datasource	Displays the target datasource.
Database	Displays the target database.
Build subfolders	Select to include subfolders.
Generate a Project Build Script and Display it in a SQL window.	Select to generate the project build script and have it opened 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.

Completing the Build Project Dialog Box

To complete the Build Project dialog box, do the following:

1. On the **File** menu click **Open Project**.

The **Open Project** dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the Project Tab, right-click the target project, and then select Build.

The **Build Project** dialog box opens.

5. Select options.
6. Click **OK**.

Set Build Order

You can specify the order in which you want your project files built 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**.

The **Open Project** dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the **Project** menu, click **Build Order**.

The **Set Build Order** dialog box opens.

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**.

Note: The next time you build the project, the new build order is used.

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.

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**.

The Open Project dialog box opens.

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.

The Project Tab containing the project opens.

4. On the **Project** menu, click Add Database Objects.

The Reverse Engineer Wizard opens.

5. Use the information in Create a New Project From a Database to complete the 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.

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**.

The Script/File Execution Facilities dialog box opens.

Add File(s) to a Project

You can add external files to your project. This lets you manipulate and expand your project as needed.

To add files to a project from, do the following:

1. On the **Project** menu, click **Add Files**.

The **Add Files to Project** dialog box opens.

2. Use the dialog to locate the directory containing the files you want to add to the project and then select the individual files to be added.

3. When ready, click **Open**.

Open a File from a Project

You can open a file directly from the Project Tab using a number of different methods.

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**.

The file(s) open in a new SQL window.

Subfolders

Subfolders help categorize your source code files. On the Project Tab of the Database Navigator, you can:

- Create subfolders.
- Delete subfolders.
- Rename subfolders.
- Sort subfolders.

Creating a New Subfolder

To create a new subfolder:

1. On the **File** menu click **Open Project**.

The Open Project dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the **Project** menu, click **New Subfolder**.
5. Type the name of the new subfolder and then press **Enter**.

Deleting a Subfolder

To delete a project, do the following:

1. On the **File** menu click **Open Project**.

The **Open Project** dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the **Project Tab**, right-click a subfolder, and then click **Delete**.

The **Confirm Delete** dialog box opens.

5. In the **Confirm Delete** dialog box, select the **Delete** local copy to delete the local copy of the subfolder, and then click **OK**.

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 subfolder.

For more information, see Deleting a Subfolder.

Renaming a Subfolder

To rename a subfolder, do the following:

1. On the **File** menu click **Open Project**.

The **Open Project** dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the **Project Tab**, right-click a subfolder, and then click **Rename**.
5. Type the name of the new subfolder and then press **Enter.**

Sorting Subfolders

To sort a subfolder, do the following:

1. On the **File** menu click **Open Project**.

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**.

The **Project Tab** containing the project.

4. On the **Project Tab**, right-click the subfolders you want to sort and then click **Sort**.

Topics

- Working with Projects.

Project Properties

You can view properties of Projects, Subfolders, and individual files. The Project Properties, Subfolder Properties, and File Properties dialog boxes display information about the Projects, Subfolders, 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**.

The Open Project dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the **Project Tab**, right-click one or more selected projects and then click **Properties**.

The **Project File Properties** or **Multiple Selection Properties** dialog box opens.

5. Complete the Project File Properties dialog box or Multiple Selection Properties Dialog Box.

6. Click **OK**.

Multiple Selection Properties Dialog Box

The table below describes the options and functionality of the Project Properties dialog box:

Option	Description
Description	Lets you type or edit the common description for the selected files.
Include In Build	Sets the files to be included in the build.
Object Type	Lets you associate a database object type with the file, as appropriate.

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 Subfolder Properties

To view subfolder properties, do the following:

1. On the **File** menu click **Open Project**.

The **Open Project** dialog box opens.

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**.

The **Project Tab** containing the project opens.

4. On the **Project Tab**, select a subfolder.
5. Right-click the subfolder, and then click **Properties**.

The **Subfolder Properties** dialog box opens.

6. Complete the Subfolder Properties dialog box.
7. Click **OK**.

Subfolder Properties Dialog Box

The table below describes the options and functionality of the Subfolder Properties dialog box:

Option	Description
Name	Displays the name of the subfolder. You can enter or edit the subfolder name.
Full Specification	Displays the subfolder location.
Status	Displays the subfolder status.

For more information, see Viewing Subfolder Properties.

Viewing File Properties

To view file properties, do the following:

1. Click **File** and then **Open Project**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project opens on the **Projects Tab**.

4. On the **Project Tab**, select a file.
5. Right-click the file, and then click **Properties**.

The **File Properties** dialog box opens.

6. Complete the File Properties dialog box.
7. Click **OK**.

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, 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, you can immediately place the project into version control or you can Create a New Project From a Version Control Project. You can also add projects and files to version control later selecting **Project** menu, **Version Control** and **Add To Version Control...**.

Version Control Integration

To use integrated version control features, you must have the version control client software installed on the same computer, or the MSSCCI plug-in for the selected Version Control System. You must select the appropriate version control on the Version Control Tab of the Option Editor.

Rapid SQL supports all Version Control Systems compatible with MSSCCI. Other Version Control Systems not compatible with MSSCCI may work under certain conditions.

Note: 64 bit versions of Rapid SQL can now use 32 bit MSSCCI providers by choosing to do so in the version control option page. Most providers are 32 bits only, so this allows users to continue to use the 64 bit version of our application even if the have a source control that has no 64 bit provider available.

Version Control Configuration

Information about how to configure a Version Control system is described in Version Control Options.

Using Version Control

Once you add a file or project to the version control, most functions found in the underlying version control system are available directly. Basic version control procedures include:

- Get Latest Version
- Check Out
- Check In
- Undo Check Out
- Open
- Delete
- Remove from Version Control
- Sort
- Show History
- Show Difference
- Version Control Properties
- VC Files Tab
    - Expand All
    - Collapse All
    - Refresh
    - Close Files List

Note: 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.

Version Control Icons

The table below describes the icons related to Version Control files:

Icon	Description
📄	File is not checked out by anyone.
☑	File is checked out non-exclusively and only by the user logged into source control.
☑	File is checked out non-exclusively only by a one user who is not the user logged into source control.
☑	File is checked out exclusively and only by the user logged in to source control.
☑	File is checked out exclusively and only by a user who is not the user logged in to source control.
☑	File is checked out by multiple users, including the user logged in to source control.
☑	File is checked out by multiple users, not including the user logged in to source control.
📄	Project file is not checked out by anyone.
☑	Project file is checked out non-exclusively and only by the user logged into source control.
☑	File is checked out non-exclusively only by a one user who is not the user logged into source control.
☑	Project file is checked out exclusively and only by the user logged in to source control.
☑	Project file is checked out exclusively and only by a user who is not the user logged in to source control.
☑	Project file is checked out by multiple users, including the user logged in to source control.
☑	Project file is checked out by multiple users, not including the user logged in to source control.

Topics

- Working with Projects in Version Control

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...**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project on the **Projects Tab** opens.

4. Click the project or target files.
5. On the **Project** menu, click **Version Control**.

The **Version Control** menu opens.

6. Click **Get Latest Version...**.

The **Get From Version Control** dialog box opens.

7. In the **Files to Get** box, click the project or files.
8. For advanced options, click **Advanced**.
9. Click **OK**.

The most current version of the file is written to your working directory.

Getting Latest Version of a File

To get the latest version of a file, do the following:

1. On the **Project** Tab, right-click the target file(s) and select **Get Latest Version...**.

The **Get From Version Control** dialog box opens.

2. In the **Files to Get** box, select the file(s).
3. For advanced options, click **Advanced**.
4. Click **OK**.

The most current version of the file is added to the **Project** Tab.

Getting Latest Version of a File from VC Files Tab

To get the latest version of a file, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Get Latest Version...'**.

The **Get From Version Control** dialog box opens.

2. In the **Files to Get** box, select the file(s).
3. For advanced options, click **Advanced**.
4. Click **OK**.

The most current version of the file is added to the **VC Files** Tab.

Topics

- Using Version Control.

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. A red check mark over the file icon indicates 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.

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 Version Control Options.

Checking Out a Project

To check-out a project, do the following:

1. Click **File** and then **Open Project**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click Open.

The project opens on the **Projects** Tab.

4. Select the target project.
5. On the **Project** menu, click **Version Control**.
6. Select **Check Out**.

The **Check Out** dialog box opens.

7. In the **Files to Be Checked Out<** box select the project.
8. For advanced options, click **Advanced**.
9. Click **OK**.

The project or files are checked out 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 **Project** Tab, right-click the target file(s) and select **Check Out**.

The **Check Out File(s)** dialog box opens.

2. In the **Check Out File(s)** dialog box select the files.
3. For advanced options, click **Advanced**.
4. Click **OK**.

The file(s) is checked out from version control and the most current version of the file is added to the **Project** Tab.

Check Out a File from VC Files Tab

To check out a file, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Check Out**.

The **Check Out File(s)** dialog box opens.

2. In the **Check Out File(s)** dialog box select the files.
3. For advanced options, click **Advanced**.
4. Click **OK**.

The file(s) is checked out from version control and the most current version of the file is added to the **VC Files** Tab.

Topics

- 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**.
2. The **Open Project** dialog box opens.
3. Type the project name or select the project.
4. Click **Open**.

The project opens on the **Project** Tab.

5. On the **Project** menu, click **Version Control**, and then **Check In**.

The **Check In** dialog box opens.

6. In the **Files to Be Checked In** box click the project or files.
7. 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.
8. **OPTIONAL:** In the **Comment** text box type a description of the changes.
9. Click **OK**.

Check In a File

To check in a file, do the following:

1. On the **Project** Tab, right-click the target file(s) and select **Check In**.

The **Check In File(s)** dialog box opens.

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. **OPTIONAL:** In the **Comment** text box type a description of the changes.
5. Click **OK**.

Check In a File from VC Files Tab

To check in a file, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Check In**.

The **Check In File(s)** dialog box opens.

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**.

Topics

- 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**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project opens on the **Project** Tab.

4. Click the project.
5. On the **Project** menu, click **Version Control**.
6. Click **Undo Check Out**.

The **Undo Check Out** dialog box opens.

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**.

   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 **Project** Tab, right-click the target file(s) and select **Undo Checkout**.

The **Undo Checkout** dialog box opens.

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**.

   Note: If you Undo Check Out, you lose any changes you have made to the local copy of your file(s).

Undoing Checkout for a File from VC Files Tab

To undo checkout for a file, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Undo Checkout**.

The **Undo Checkout** dialog box opens.

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**.

   Note: If you Undo Check Out, you lose any changes you have made to the local copy of your file(s).

Topics

- 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**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project opens on the **Project** Tab.

4. Click the target file.
5. On the **Project** menu, click **Version Control**, and then select **Show History**.

The **History** dialog box opens.

> **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 **Project** Tab, right-click the target file(s) and select **Show History**.

The **History** dialog box opens.

> **Note:** The **History** dialog box depends on your version control system.

Showing History for a File from VC Files Tab

To show history, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Show History**.

The **History** dialog box opens.

Topics

- 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**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project opens on the **Project** Tab.

4. Click the target file.
5. On the **Project** menu, click **Version Control**.
6. Click **Show Differences**.

The **Differences** dialog box opens.

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 **Project** Tab, right-click the target file(s) and select **Show History**.

The **Difference** dialog box opens.

2. In the **Difference** dialog box, you can maneuver through the files by using the **Up** and **Down** arrows.

   Note: The **Difference** dialog box and options depend on your version control system.

Showing History for a File from VC Files Tab

To show history, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Show History**.

The **Difference** dialog box opens.

2. In the **Difference** dialog box, you can maneuver through the files by using the **Up** and **Down** arrows.

Topics

- 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**.**

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project opens on the **Project** Tab.

4. On the **Project** menu, click **Version Control**, and then click **Remove from Version Control**.

The **Remove File(s)** dialog box opens.

5. To permanently destroy the project select the check box, click **OK**.

Removing a File from Version Control

To remove a file from Version Control, do the following:

1. On the **Project**, right-click the target file(s) and select **Remove from Version Control**.

The **Remove File(s)** dialog box opens.

2. Select the file(s) in the **Files to be Removed from Version Control** box.
3. Click **OK**.

Removing a File from Version Control from VC Files Tab

To remove a file from Version Control, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Remove from Version Control**.

The **Remove File(s)** dialog box opens.

2. Select the file in the **Files to be Removed** box.

3. Click **OK**.

Topics

- o [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**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.
3. Click **Open**.

The project opens on the **Project** Tab.

4. Click the target file.
5. On the **Project** menu, click **Version Control**, and then **Version Control Properties**.

The **Version Control Properties** dialog box opens.

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 **Project** Tab, right-click the target file(s) and select **Version Control Properties**.

The **Version Control Properties** dialog box opens.

2. Review properties.
3. Click **Close**.

Viewing Version Control Properties for a File from VC Files Tab

To view the version control properties, do the following:

1. On the **VC Files** Tab, right-click the target file(s) and select **Version Control Properties**.

The **Version Control Properties** dialog box opens.

2. Review properties.

3. Click **Close**.

Topics

- Using Version Control.

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, a project is created 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. 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.

Completing the Add to Version Control Dialog Box

To add a project to version, do the following:

1. Click **File** and then **Open Project**.

The **Open Project** dialog box opens.

2. Type the project name or select the project.

3. Click **Open**.

The project opens on the **Project** Tab.

4. On the **Project** menu, click **Version Control**.

The **Add to Version Control** dialog box opens.

5. Click **Add to Version Control**.
6. Select options needed.
7. Click **OK** to add a project, and then click **Create**.

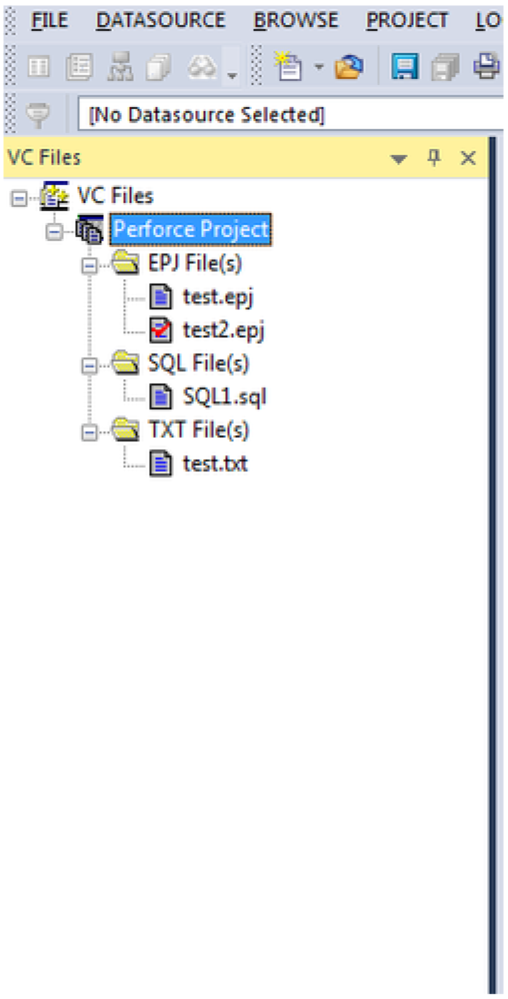
   Note: The project file icon dims to indicate that the project has been placed under version control.

Topics

- Working with Projects.

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.



Opening the VC Files Tab

1. Select File, Open Version Control File List.

If you have files on your version control list, the VC Files Tab opens. If you do not have files on your version control list, the Add Version Control Files dialog box opens.

Closing the VC Files Tab

1. Select File, Close Version Control File List.

VC Files Tab Available Functionality

Once the files are on the list in the VC Files Tab, you can use the version control functionalities described in Version Control#Using Version Control as well as some functionalities specific for the VC Files Tab:

- Version Control Functionality - Expand All
- Version Control Functionality - Collapse All

- Version Control Functionality - Refresh
- Version Control Functionality - Close Files List

Different functionalities are available depending on the level of the tree:

	At Version Control Files Level	At Project Level	At Directory Level	At File Level
Add Files	✓	✓	✓	
Open				✓
Delete		✓	✓	✓
Sort	✓	✓	✓	
Get Latest Version	✓	✓	✓	✓
Check Out	✓	✓	✓	✓
Check In	✓	✓	✓	✓
Undo Check Out	✓	✓	✓	✓
Show History				✓
Show Differences				✓
Remove from Version Control	✓	✓	✓	✓
Version Control Properties				✓
Expand All	✓	✓	✓	
Collapse All	✓	✓	✓	
Refresh	✓	✓	✓	✓
Close Files List	✓	✓	✓	✓

Topics

- Working with Files in Version Control
- Working with Projects in Version Control

Version Control Functionality - Expand All

The **Expand All** functionality expands all tree items under the selected item(s).

Topics

- Working with Files in Version Control

Version Control Functionality - Collapse All

The **Collapse All** functionality collapses all tree items under a selected item(s).

Topics

- Working with Files in Version Control

Version Control Functionality - Refresh

The **Refresh** functionality obtains the current version control status for the file(s).

Topics

- Working with Files in Version Control

Version Control Functionality - Close Files List

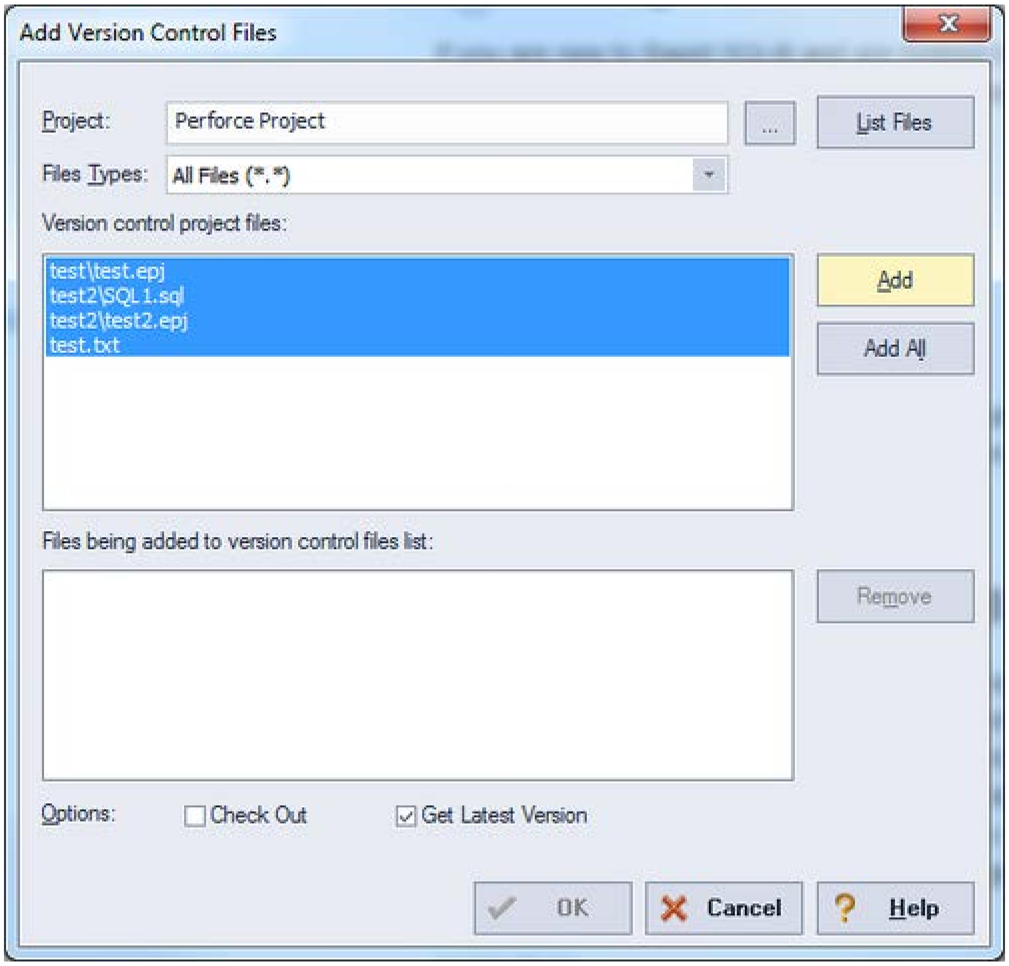
The **Close Files List** functionality closes the list of files of the **VC Files** Tab.

Topics

- Working with Files in Version Control

Working with Files in Version Control

You can view and work with files stored in various projects in a version control system without including the files in a project. The Add Version Control Files dialog box lets you create a list of files in a version control system.



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.

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.

The version control system login dialog box opens.

2. Type login information.

The Add Version Control Files dialog box opens.

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.

The files are displayed 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.

The Check Out File or Get File dialog box opens.

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. Click OK.

The files are added in the Files being added to version control files list box to the VC Files Tab. If you selected the Check Out option, the Check Out (Files) dialog box opens. If you selected the Get Latest Version option, the Get from Version Control dialog box opens.

Tools

The **Tools** menu offers the following:

Tool	Description
Find in Files	Lets you find a phrase or character in your files.
Database Search	Lets you search for objects whose DDL contains specified strings.
Script/File Execution Facilities	Two stand-alone utilities that let you type or paste SQL statements to be executed or execute a script in a file.
Scheduling	Describes tools you can use to schedule tasks.
Visual Difference	Lets you compare two files or database objects.
Query Builder	Lets you build queries using a visual interface.
Code Analyst (Tools Menu)	Provides a means to identify costly, time-consuming code.
Code Generation Facility	Lets you generate DML-based procedures or packages.
Import Data	Lets you import data from **.xls**, **.sql**, **.tab**, or **.csv** files.
SQL Profiler	Provides access to the PL/SQL Profiler.
Table/Index Size Estimator	Provides access to the **Estimate Size** object action.
Embarcadero Product Menu Commands	Correspond to installed or available Embarcadero Technologies products.
Available DBMS Utility Menu Commands	Lets you start third party utilities.
Code Workbench	Lets you set up keystroke-saving shortcuts that can be used in the SQL Editor.
Customize (Tools menu)	Lets you customize the user interface and set up shortcuts.
Options (Tools menu)	Lets you configure features and specify preferences.
Data Editor	Provides a live data editor.

Find in Files

The Find in Files tool lets you search files for a phrase or character.

To search files for specified characters:

1. On the **Tools** menu, select **Find in Files**. A **Find in Files** dialog opens.
2. Use the table below as a guide to choosing settings on this 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.
Other common Search options	The **Find in Files** dialog lets you specify the following common search options: **Match whole word only**, **Match case**, **Regular Expression** (tells the application whether the specified character(s) is a regular expression), **Look in subfolders**, and **Output to Pane 2** (displays the results in another window).

3. When ready, click **Find**.

Database Search

The powerful Database Search tool lets you search for objects whose DDL contains a specified character string, across multiple databases.

Platform Availability

DB2 LUW	DB2 z/OS	ITB/FBD *	MySQL	ORCL	PSTGRS *	SQL SVR	SYB ASE	SYB IQ
✓	✓	✓	✓	✓	✓	✓	✓	

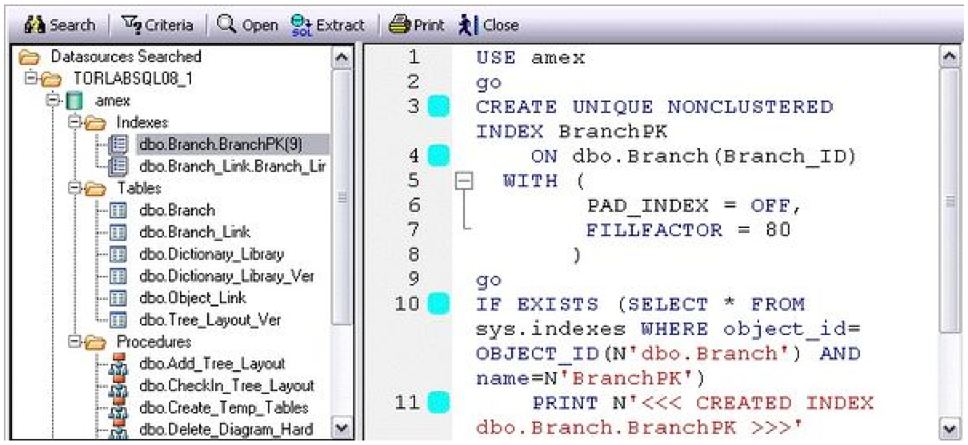
To find instances of a string on multiple databases:

1. On the **Tools** menu, click **Database Search**. The **Database Search Wizard** opens.
2. Use the following table as a guide to understanding and modifying the settings on the panels of this wizard:

Panel	Settings and tasks
Select the datasources you want to search	Depending on the DBMS platform, you may also be able to further qualify by database, user, or schema.
Select the objects you want to search	Lets you specify the character string to search for, select a case matching setting, and specify the object types that are to be searched. Depending on the DBMS platform, searchable object types are **defaults**, **foreign keys**, **functions**, **indexes**, **materialized tables**, **package bodies**, **packages**, **primary keys**, **procedures**, **rules**, **tables**, **triggers**, **unique keys**, and **views**.

1. Click **Finish** to start the search.

A progress dialog box is displayed while the search runs. When the search completes, the results open in a **Database Search** window. The left pane lists all objects whose DDL satisfied the search. The right pane displays the DDL of a selected object. Lines containing the specified character string are flagged.



```
USE amex
go
CREATE UNIQUE NONCLUSTERED
INDEX BranchPK
    ON dbo.Branch(Branch_ID)
  WITH (
      PAD_INDEX = OFF,
      FILLFACTOR = 80
     )
go
IF EXISTS (SELECT * FROM
sys.indexes WHERE object_id=
OBJECT_ID(N'dbo.Branch') AND
name=N'BranchPK')
    PRINT N'<<< CREATED INDEX
dbo.Branch.BranchPK >>>'
```

When viewing results, the following options are available from the **Database Search** window toolbar:

Button	Description
Search	Reopens the **Database Search** wizard with the most recent settings specified. This lets you further qualify settings for a new search.
Criteria	Opens a window displaying the case matching option and character string specified for the search.
Open	Opens the selected object in an object editor. For information on using object editors, see Modifying objects using editors.
Extract	Extracts the DDL for a selected object to a SQL editor window. For more information, see Using the SQL Editor.
Print	Lets you print the target object SQL.

Script/File Execution Facilities

Two stand-alone utilities establish multiple threads and database connections, letting you simultaneously execute SQL statements against multiple cross-DBMS platform datasources.

- Script Execution Facility lets you type or paste SQL statements to be executed
- File Execution Facility lets you load files containing SQL statements to be executed

Both utilities also work in conjunction with supported scheduling facilities, letting you schedule execution jobs. After completing a scheduled job, a results report lists errors, verifies script execution, and details the output of the job. Results can be automatically sent to any e-mail or network recipients.

Both utilities use a tabbed dialog box where you set the parameters and options for the script execution.

To execute SQL using one of these facilities

1. On the **Tools** menu, select either **Script Execution Facility** or **File Execution Facility**.

A tabbed dialog opens. It lets you provide the SQL to be executed and provide additional details.

2. Use the following table as a guide to selecting settings on the tab of this dialog:

Tab	Setting	Description
Script (Script Execution Facility only)	**Script** box	Lets you type or paste a script.
Files (File Execution Facility only)	**Show Full File Paths**	Select to display the full path. Deselect to display only the file name.
	File Name	Displays the file names.
	Add and **Remove**	Use these controls to add or remove files.
	View	Opens the View File dialog box.
	Up and **Down**	Use these controls to move the selected file up or down in the list.

Target	**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.
Output	**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.
Notify	**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 (;).

3. When ready, click **Execute**.

Scheduling

You can schedule routine tasks and jobs using the following utilities:

- Microsoft Task Scheduler
- ETSQLX Command Line Utility

Microsoft Task Scheduler

You can 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, you are offered a link to the Microsoft Web site where you can download the Microsoft Task Scheduler at no cost.

Note: The ETSQLX command line utility runs a scheduled job even if the application is not running. For more information, see ETSQLX Command Line Utility.

To open the Microsoft Job Scheduler

1. On the **Tools** menu, click **Scheduler**.
2. Use the associated online Help for assistance with using this Microsoft utility.

ETSQLX Command Line Utility

The 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

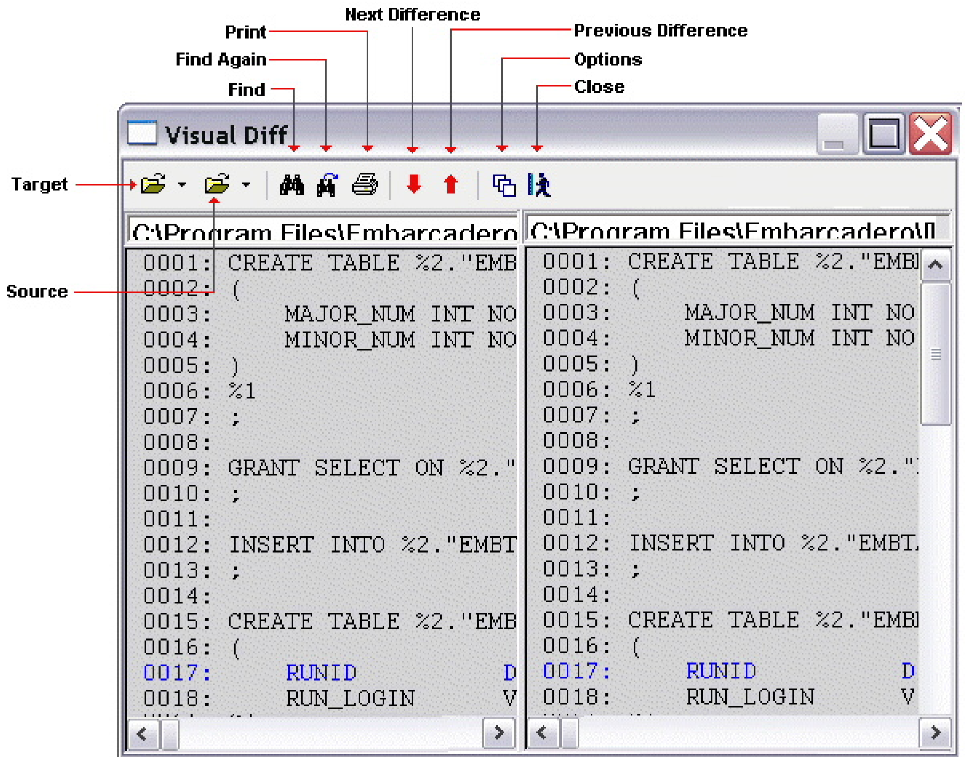
You can compare two text files or database objects (extracted DDL). Using the **Visual Diff** dialog box, you can easily synchronize and analyze database objects or files across multiple database platforms. The files are displayed side by side. The Visual Difference utility highlights any differences 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 Diff** dialog box are read-only, you cannot modify files or objects directly from this dialog box.

To open the Visual Diff dialog box

1. On the **Tools** menu, click **Visual Diff**.

The **Visual Diff** dialog box opens.



```
C:\Program Files\Embarcadero
CREATE TABLE %2."EMB
(
     MAJOR_NUM INT NO
     MINOR_NUM INT NO
)
%1
;

GRANT SELECT ON %2."
;

INSERT INTO %2."EMBT
;

CREATE TABLE %2."EMB
(
     RUNID           D
     RUN_LOGIN       V
```

```
C:\Program Files\Embarcadero\I
CREATE TABLE %2."EMB
(
     MAJOR_NUM INT NO
     MINOR_NUM INT NO
)
%1
;

GRANT SELECT ON %2."
;

INSERT INTO %2."EMBT
;

CREATE TABLE %2."EMB
(
     RUNID           D
     RUN_LOGIN       V
```

The **Visual Diff** 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 toolbar provides the basic visual search functions.

Topics

- Setting Visual Difference options
- Specifying the Files or Database Objects to Compare

- Navigating to the next or previous highlighted difference
- Finding character strings in the source or target
- Printing a Pane of the Visual Difference Dialog Box

Setting Visual Difference options

The **Visual Diff** dialog box lets you set display and comparison options to help you customize the dialog box to view differences in a comprehensive manner.

To set options for visual difference operations

1. On the **Tools** menu, click **Visual Diff**.
2. Click the **Options** icon on the toolbar.

The **Visual Diff Option**s dialog box opens.

3. Use the following table as a guide to setting options in this dialog:

Option	Description	Default
Display Line Numbers	Indicates that line numbers should be displayed.	Off
Display Hidden Characters	Indicates that hidden, nonprintable characters 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 nonprintable characters should be excluded.	Off
Ignore Case	Indicates that case should not be a differentiating factor.	On

4. Click **OK**.

Specifying the Files or Database Objects to Compare

You can compare two files or database objects side-by-side in the **Visual Diff** dialog box. The file or object you want to compare is called the source. The file or object you want to compare the source to is the target.

To open the source and target

1. On the **Tools** menu, click **Visual Diff**.
2. On the toolbar, click the down arrow next to the Source icon and then click **File** or **Database Object**.

The **Select the 1st File to Compare** or **Select the 1st Database Object to Compare** dialog box opens.

3. Take one of the following actions:
   - Use the **Select the 1st File to Compare** dialog box to locate and select the source file
   - Use the **Select the 1st Database Object to Compare** dialog box to navigate the datasource tree to locate and select the source database object.
4. Repeat those actions to select a target file or database object. A progress dialog is briefly displayed, showing processing of the differences between the source and target.

Note: The Visual Difference Utility highlights all differences between the two files.

Once you have added the source and target, you can navigate back and forward among the highlighted differences, search the source or target for character strings, or print the contents of the source or target pane.

Topics

- Navigating to the next or previous highlighted difference
- Finding character strings in the source or target
- Printing a Pane of the Visual Difference Dialog Box
- Visual Difference

Navigating to the next or previous highlighted difference

You can navigate through the **Visual Diff** 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.

To select the next or previous difference in the source and targets

1. Open the **Visual Diff** dialog box and load the source and target files or database objects. For details, see Specifying the Files or Database Objects to Compare.
2. On the toolbar, click the down arrow to go to the next difference or the up arrow to go to the previous difference.

⬇⬆

Topics

- Visual Difference.

Finding character strings in the source or target

The **Visual Diff** dialog box lets you search for text in your files or database objects.

To search for text in the source or target pane

1. Open the **Visual Diff** dialog box and load the source and target files or database objects. For details, see Specifying the Files or Database Objects to Compare.
2. Place your cursor inside the pane you want to search.
3. Click the Find icon on the Visual Difference toolbar.

The **Find** dialog box opens.

4. In the **Find What** box, enter the search string.
5. Use the **Match**... controls specify whole word searching or case sensitivity.
6. Click **Find Next** to find the next occurrence of the search string.

After specifying the search string in a visual difference session, you can repeat the search without opening the **Find** dialog

To find the next instance of the search string

1. Click the Find Next icon on the Visual Difference toolbar.

Topics

- Visual Difference.

Printing a Pane of the Visual Difference Dialog Box

You can print each pane of the **Visual Diff** dialog box.

To print the target or source pane

1. Open the **Visual Diff** dialog box and load the source and target files or database objects. For details, see Specifying the Files or Database Objects to Compare.
2. Position your cursor inside the pane you want to print.
3. Click the Print icon on the Visual Difference toolbar.

4. Click **OK.**

Note: You can only print one pane of the **Visual Diff** dialog box at a time.

Topics

o Visual Difference.

SQL Profiler

The **Tools** menu provides access to the SQL Profiler, a license-specific module:

- For licensing information, see Licensing Your Product
- For detailed information on using the profiler, see PL/SQL Profiler.

Table/Index Size Estimator

The **Tools** menu provides access to the **Estimate Size** object action, available against Oracle and Sybase ASE tables and indexes. For details, see Estimate Size.

Import Data

It is often necessary to import data into database tables from external sources. You may need to bring sample data into a database to establish test case scenarios or import data for use in the real world. Business analysts, for example, often use spreadsheets from outside sources. Putting such data into tables can help them perform detailed queries and make informed decisions based on meaningful data.

The **Import Data** tool lets you import data from a Microsoft Excel (**.xls**) file or any SQL data file format in which data values are stored in plain-text form and delimited by tab, semicolon, comma, space, or other user-specified character (such as **.sql**, **.tab**, or **.csv**).

Note: When you import a text file, the wizard may give the impression of stalling. Press ESC and you should be able to continue without further incident.

To Open the Import Data Wizard:

1. Connect to the datasource to which you want to import data and ensure that the datasource is selected. For more information, see Connecting to Datasources.
2. On the **Tools** menu, select **Import Data**. The **Import Data Wizard** opens.
3. Use the following table as a guide to providing information as you work through the panels of this wizard:

Panel	Setting	Description
Data Properties	Specify the file to be used in this data load operation	Type the path and name of the file that you want to imprt from or use the browse button to locate and select the file.
	Which table do you want to load data into?	If working against a Sybase or SQL Server datasource, select the **Catalog** of tables. Select the **Schema** that owns the target table. Select the particular **Table** where you want to import the data.
Delimiter	What character delimited the columns in the data file?	Choose among **Tab**, **Semicolon**, **Comma**, or **Space**. Alternatively, select **Other** and type the specific delimiting character.
	First Row Contains Field Names	By choosing this, if the first row contains column names, these will be displayed as the first row of the file. You can still change column mappings if you want. If you do not select this, generic **Field 1**, **Field 2**,... **Field***n* will indicate column mappings.

Column Mapping		**Note:** You must map all mandatory (NOT NULL) column fields to a column heading before you can enable the Finish button and import the data. To map columns, click the grey bar at the top of the grid and a drop-down list appears. The names in this drop-down list are columns in the table that's going to receive the data. These need to sync up to the data that's being imported. Red, or mandatory columns, turn blue in the column list when they have been assigned. A maximum of 10 rows are displayed here to make mapping columns easier. All rows will be imported when the job completes. If there are more columns in the file you are importing than in the receiving table, the extra columns will be ignored and you can create the insert statements. You cannot complete the import operation if columns are mapped more than once. The import operation will not let you generate invalid columns.
Excel Column Mapping (.xls files only)	Sheet	If the file you selected has more than one sheet, identify the worksheet that has the data you want to import.
	Start/End Cell	Identify the starting and ending cells of data you want to import and click Refresh. Or, accept the default cells.

4. Click **Finish**.

Code Generation Facility

The Code Generation Facility offers a quick way to generate DML statements for tables and views:

- Packages and procedures for Oracle.
- 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 user interface lets you select the items you need to generate a generic block of code such as the following:

```
CREATE OR REPLACE PROCEDURE ADDRESS_SEL
(
    city_in IN SYSTEM.ADDRESS.CITY%TYPE,
    addressline1_out OUT SYSTEM.ADDRESS.ADDRESSLINE1%TYPE,
    addressline2_out OUT SYSTEM.ADDRESS.ADDRESSLINE2%TYPE,
    city_out OUT SYSTEM.ADDRESS.CITY%TYPE,
    stateprovinceid_out OUT SYSTEM.ADDRESS.STATEPROVINCEID%TYPE,
    postalcode_out OUT SYSTEM.ADDRESS.POSTALCODE%TYPE
)
IS
BEGIN
    SELECT
    t.ADDRESSLINE1,
    t.ADDRESSLINE2,
    t.CITY,
    t.STATEPROVINCEID,
    t.POSTALCODE
    INTO
        addressline1_out,
        addressline2_out,
        city_out,
        stateprovinceid_out,
        postalcode_out
    FROM SYSTEM.ADDRESS t
    WHERE
     t.CITY = city_in;

END;
```

The procedure (or package) can be executed directly from the choices on the Code Generation Facility window or opened in the SQL Editor after generation. For more information, see Using the SQL Editor.

To generate a procedure or package using the Code Generation Facility

1. Select Tools > Code Generation Facility to open the **Embarcadero Code Generator** window.
2. Use the following table as a guide to understanding the user interface elements as you make the choices in generating a procedure or package.

Option	Description
Datasource/database/Owner lists	Let you qualify the table or views that you want to generate a procude ot package against. **NOTE:** If you select a currently unconnected datasource, a Connect button becomes available, letting you connect to the target datasource. The current datasource, and database and datasource user name where applicable, are preselected when you open Code Generation Facility.
Tables/Views buttons and list	Let you select the table or view that the SELECT/INSERT/UPDATE/DELETE statement is to be generated against.
Select 1 or More Where Clause Columns	Select the check boxes that correspond to the target where clauses. **NOTE:** Columns of primary keys are preselected.
Select 1 or More Output Columns	Select the check boxes that correspond to the target output columns.
Provide an Output File Name and **Open**	Lets you specify that the procedure or package is to be written to a file. Optionally, you can have the file opened after writing the file.
Execute Immediately	Select to execute the file immediately.
Generate	Specify whether you want to generate a procedure or package, and select one or more DML statement types.
Grant Execute to	Use these controls to grant execute privileges.

3. Click **OK** when ready to generate the procedure or package.

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
Building a SELECT Statement	Create, manipulate and execute SELECT Statements for tables and views.
Building an INSERT Statement	Create and manipulate INSERT Statements for tables.
Building an UPDATE Statement	Create and manipulate UPDATE Statements for tables.
Building a DELETE Statement	Create and manipulate DELETE Statements for tables.
Building a CREATE VIEW Statement	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.

Topics

- 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. 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 Dialog Boxes
- Workspace Windows
- Query Builder Toolbar
- Tables and Views Shortcut Menus
- Tables and Views Keyboard Commands

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.

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 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 Navigator, and then open another Query Builder session. Query Builder prompts you to save the current session prior to opening a new session. For details, see Saving and Reopening Queries.

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.

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:

- Working with Tables and Views in the SQL Diagram Pane

- Joins
- Working with Columns in the SQL Diagram Pane

All changes in the SQL diagram reflect in correct SQL code in the SQL Statement Pane.

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

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. You can 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 details, see Using Data Editor with Query Builder.

Topics

- Query Builder
- Creating a Clause Using the SQL Statement Tree

Query Builder Toolbar

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.

Topics

- Query Builder.

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	For more information, see Table Properties.

Note: Your selection applies to all selected tables and views.

Topics

- Query Builder

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.

Topics

- For more information, see Query Builder

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.

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. These properties override the global options set in the Options editor for the current session. For details, see Query Builder Options.

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 in the Options editor while Query Builder is running, a warning indicates that you are about to change options or properties. For details, see Query Builder Options.

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**.
2. Set options.
3. Click **OK**.

Query Builder saves the options.

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.

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.

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.
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**.

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 for Linux, Unix, and Windows and IBM DB2 for z/OS and OS/390 servers, there is an additional join object in the SQL Statement Tree. The Join On node displays join relations between columns in IBM DB2 for Linux, Unix, and Windows and IBM DB2 for z/OS and OS/390 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**.

Topics

- Query Builder

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. Query Builder lets you build five separate types of queries simultaneously:

- Building a SELECT Statement
- Building an INSERT Statement
- Building an UPDATE Statement
- Building a DELETE Statement
- Building a CREATE VIEW Statement

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. Saving and Reopening Queries.

You can 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. You can use different tables and views for each query. You can also toggle back and forth among each of the queries. For more information, see Working with Tables and Views in the SQL Diagram Pane.

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. For details, see Saving and Reopening Queries.

Topics

- Main Steps in Using Query Builder
- Building a Query
- Working with Tables and Views in the SQL Diagram Pane
- Working with Columns in the SQL Diagram Pane
- Selecting ALL or DISTINCT Columns
- Joins
- Auto Layout

- Auto Joins
- Creating a Clause Using the SQL Statement Tree
- Subqueries
- Syntax Checker
- Saving and Reopening Queries
- Using Data Editor with Query Builder

Main Steps in Using Query Builder

To use Query Builder, do the following:

- Selecting a Database
- Selecting a Statement
- Selecting Tables and Views
- Selecting 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.

Query Builder opens.

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, IBM DB2 for z/OS and OS/390 or IBM DB2 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.

The current query clears and a warning prompt is displayed.

5. To save the current query, click **Yes**.

The **Save As** dialog box opens.

6. To continue without saving, click **No**.

Selecting a Statement

Query Builder lets you build SELECT, INSERT, UPDATE, DELETE, and CREATE VIEW 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.

Selecting Tables and Views

To build a query, open one or more tables or views in the SQL Diagram Pane. For details, see SQL Diagram Pane.

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**.

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. For details, see:

- SQL Diagram Pane.

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.

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.

Selecting Columns in the SQL Statement Tree

Query Builder lets you select and set individual properties using the **Selected Column Properties** Dialog Box. For details, see Column Properties.

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**.

Topics

- Saving and Reopening Queries
- Using Query Builder

Building a Query

Query Builder lets you build SELECT, INSERT, UPDATE, SELECT, and CREATE VIEW statements, which you can run separately or simultaneously, depending on your needs.

To build a Query, do the following:

- Selecting a Database
- Selecting a Statement
- Selecting Tables and Views
- Selecting 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.

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.

The Query Builder opens.

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.

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. For details, see Saving and Reopening Queries.

Building an INSERT Statement

To build an INSERT Statement, do the following:

1. On the **Tools** menu, click **Query Builder**.

The Query Builder opens.

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.

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. For details, see Saving and Reopening Queries.

Building an UPDATE Statement

To build an UPDATE statement, do the following:

1. On the Tools menu, click Query Builder.

The Query Builder opens.

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.

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. For details, see Statement Properties.

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. For details, see Saving and Reopening Queries.

Building a DELETE Statement

To build a DELETE statement, do the following:

1 On the Tools menu, click Query Builder.

The Query Builder opens.

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.

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. For details, see Saving and Reopening Queries.

To build a CREATE VIEW statement, do the following:

1. On the Tools menu, click Query Builder.

The Query Builder opens.

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.

The ISQL Editor opens.

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.

The CREATE VIEW query executes.

11. To close the Editor, click Close.

The ISQL Editor save message opens.

12. Click No.
13. To add the view to the table tree, on the **Query Builder** menu, click Refresh.

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.

Topics

- Query Builder
- Working with Tables and Views in the SQL Diagram Pane
- Working with Columns in the SQL Diagram Pane
- Joins
- Syntax Checker
- Creating a Clause Using the SQL Statement Tree
- Selecting Columns in the SQL Statement Tree
- Subqueries
- Aliases

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 a Table or View

Query Builder can automatically dictate a layout in the SQL Diagram Pane using the **Auto Layout** button. For details, see Auto Layout.

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.

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.

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.

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.

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 **Selected Column Properties** dialog box. For details, see Column Properties.

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 SQL Statement Tree. For more information, see Selecting Columns in the SQL Statement Tree.

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.

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**.

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 Joins	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 Joins	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 Joins	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 Joins	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. For details, see Editing Joins.

You can set Query Builder options, in the Options editor, to automatically create joins. For details, see Query Builder Options.

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.

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.

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.

Right Outer Joins

Right outer joins return opposite results from Left Outer Joins. 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.

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.

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.

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.
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 for Linux, Unix, and Windows and IBM DB2 for z/OS and OS/390 servers, there is an additional join object in the SQL Statement Tree. The Join On node displays join relations between columns in IBM DB2 for Linux, Unix, and Windows and IBM DB2 for z/OS and OS/390 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.

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.

Query Builder organizes your tables in the SQL Diagram Pane.

Topics

- Auto Joins

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 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.

Query Builder joins columns in the SQL Diagram Pane.

Topics

- Statement Properties

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.

For more information, see the following topics:

- Creating a WHERE Clause
- Deleting a WHERE Clause
- Creating an AND Clause in a WHERE Clause
- Deleting an AND Clause
- Inserting an AND or OR Clause
- Deleting an OR Clause
- Creating an ORDER BY Clause
- Changing the Sort Order in an ORDER BY Clause
- Deleting an ORDER BY Clause
- Creating a GROUP BY Clause
- Deleting a GROUP BY Clause
- Creating a HAVING Clause
- Deleting a HAVING Clause

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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

Deleting a GROUP BY Clause

To delete a GROUP BY clause, do the following:

1. On the SQL Statement Tree expand the GROUP BY node, and then on the Query Builder tool bar, click Delete.

OR

On the SQL Statement Tree Expand 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.

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.

Deleting a HAVING Clause

To delete a HAVING clause, do the following:

1. On the SQL Statement Tree expand 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.

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.

Topics

- Selecting Columns in 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.

Topics

- o Selecting Columns in 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. The options for the Syntax Checker tool are set in the Options Editor. For details,see Query Builder Options.

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 on application shutdown.

Query Builder runs Schema Change detection any time you set a query to execute, refresh the data, or open a saved query.

You can 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.

The Save As dialog box opens.

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: You can save data in text (*.txt) and XML (*.xml) file formats.

3. Click OK.

The file is saved and the Save As dialog box closes.

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.
- Columns 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. You can 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.

Query Builder opens.

2. Select a database or instance. For details, see Selecting a Database.
3. Select a table. For details, see Selecting Tables and Views.
4. Select a column, or columns. For details, see Selecting Columns.
5. On the Query Builder menu, click Edit Data.

The Data Editor opens.

Topics

- Data Editor

Code Analyst (Tools Menu)

The Code Analyst is a tool you can use to identify time-consuming lines of code.

Topics

- Code Analyst

Embarcadero Product Menu Commands

The **Tools** menu lists all installed Embarcadero Technologies products. This lets you toggle to or start another Embarcadero product.

Of the products listed among these menu commands, the most closely integrated with Rapid SQL is Embarcadero Team Server 2016. Team Server 2016 provides access to datasource definitions, model metadata, and alerts/notifications streams.

Topics

- Team Server 2016 Support

Available DBMS Utility Menu Commands

The **Tools** menu has commands that let you start installed third party utilities.

Code Workbench

The Code Workbench reduces the time needed for common day-to-day coding tasks. You can:

- Define auto replacement expressions that can be used to quickly insert commonly used blocks of SQL syntax or commands. For details, see Working with Code Workbench Auto Replace Shortcuts.
- Import and Export Code Workbench settings for client sharing purposes. For details, see Importing and Exporting Settings in Code Workbench.
- If using Rapid SQL, you can create customized, complete code templates of blocks of code that you can add to scripts and enable their usage in the SQL Editor. For details, see Working With Code Workbench Code Templates.

Working With Code Workbench 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 comment blocks or add common exit and error handling routines to new or existing objects.

For information on configuring and using this feature, see the following topics:

- Enabling Code Templates - tells you how to turn on the feature.
- Defining the Hot Key for the Code Templates Dialog Box - tells you how to designate the keystroke that results in shortcuts being replaced by full code templates in the SQL Editor.
- Creating Code Templates - tells you how to set up code templates and their associated shortcuts.
- Using Code Templates in the SQL Editor - tells you how to substitute complete code templates for their shortcuts in the SQL Editor.

Enabling Code Templates

In order to have shortcuts substituted with code templates in the SQL Editor, you must first enable the feature.

To enable the use of code templates

1. Select Tools > Code Workbench to open the **Code Workbench** dialog.
2. Select **Enable Code Templates**.

For related information, see the following topics:

- Defining the Hot Key for the Code Templates Dialog Box - tells you how to designate the keystroke that results in shortcuts being replaced by full code templates in the SQL Editor.
- Creating Code Templates - tells you how to set up code templates and their associated shortcuts.
- Using Code Templates in the SQL Editor - tells you how to substitute complete code templates for their shortcuts in the SQL Editor.

Defining the Hot Key for the Code Templates Dialog Box

In order to have shortcuts substituted with code templates in the SQL Editor, you must first designate the keystroke that triggers the substitution.

To designate the keystroke that triggers code template substitution

1. Ensure that the Code Templates feature in enabled. For details, see Enabling Code Templates.
2. Select Tools > Code Workbench to open the **Code Workbench** dialog.
3. Click **Edit Hot Keys** to open the **Customize** dialog.
4. On the **Keyboard** tab, select **Code Assist** from the **Commands** list, in the **Press New Shortcut Key** box, type CTRL+SPACE, and click **Assign**.
5. Click **Close** to dismiss the **Customize** dialog and click **OK** to close the **Code Workbench** dialog.

For related information, see the following topics:

- Creating Code Templates - tells you how to set up code templates and their associated shortcuts.
- Using Code Templates in the SQL Editor - tells you how to substitute complete code templates for their shortcuts in the SQL Editor.

Creating Code Templates

In order to have shortcuts substituted with code templates in the SQL Editor, you must first designate the keystroke that triggers the substitution. The **Code Workbench** window's **Code Templates** tab lets you maintain your list of code templates and their associated shortcuts.

To access the Code Templates tab

1. Select Tools > Code Workbench and select the **Code Templates** tab.

To add or edit a code template

1. Click **Add** or **Edit** to open the **Edit Code Template** expression dialog.
2. In the **Shortcut** box, type a shortcut of your choosing (or replace the existing one).
3. In **Description**, type a comment about this code template.
4. In **Platform** assign the template to a specific DBMS platform. Groups and sorts in the **Code Templates** tab help you browse available code templates from an open, editable window (ISQL, text, HTML, Java, etc.).
5. In **Type** assign the template to a specific code type. You can type or select a type. Groups and sorts in the Code Templates tab and helps you browse available code templates from an open, editable window (ISQL, text, HTML, Java, etc.).
6. In the **Template** window, type or replace the block of code.
7. Click **OK**.

Code Workbench creates the template and displays it on the Code Templates Tab.

To delete a code template

1. Select a template from the list, click **Delete**, and confirm when prompted.

For related information, see the following topics:

- Enabling Code Templates - tells you how to turn on the Code Template feature.
- Defining the Hot Key for the Code Templates Dialog Box - tells you how to designate the keystroke that results in shortcuts being replaced by full code templates in the SQL Editor.
- Using Code Templates in the SQL Editor - tells you how to substitute complete code templates for their shortcuts in the SQL Editor.

Using Code Templates in the SQL Editor

Code templates are complete code blocks that can be easily added to scripts with a few keystrokes. Templates let you define standard comment blocks or add common exit and error handling routines to new or existing objects.

1. Ensure that the Code Templates feature is enabled, a hot key designated, and that one or more templates have been defined. For access to more information on those tasks, see Working With Code Workbench Code Templates.

2. Open the SQL Editor and ensure there is an active datasource. For details, see Connected/Selected Datasource options.

3. Include one or more template shortcuts in a script.

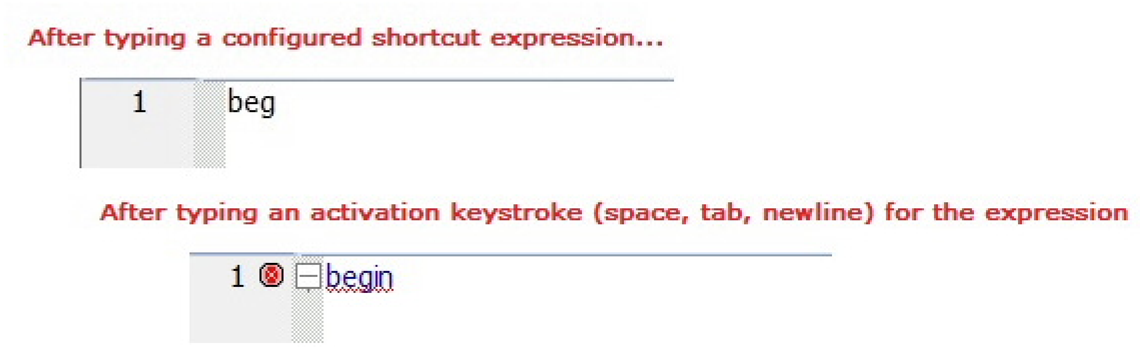
4. At any time, press CTRL+SPACE. All instances of shortcuts are replaced by the associated, complete code template.

For related information, see the following topics:

- Enabling Code Templates - tells you how to turn on the Code Template feature.
- Defining the Hot Key for the Code Templates Dialog Box - tells you how to designate the keystroke that results in shortcuts being replaced by full code templates in the SQL Editor.
- Creating Code Templates - tells you how to create and maintain your list of code emplates and shortcuts.

Working with Code Workbench Auto Replace Shortcuts

The Code Workbench's Auto Replace feature lets you create a set of shortcuts associated with words, terms, and other character strings commonly used when scripting. Subsequently, when using the SQL Editor, you can have the full character string automatically substituted for the shortcut. For example, if you have a **beg** shortcut set up for the word **begin**, if you type **beg** in the the SQL Editor, followed by an activation keystroke (space, tab, or newline), **beg** is replaced by the full word **begin**.



After typing a configured shortcut expression...

```
beg
```

After typing an activation keystroke (space, tab, newline) for the expression

```
begin
```

The following topics describe the steps involved in working with Auto Replace shortcuts:

- Enabling Auto Replace Shortcut Substitution
- Configuring Auto Replace Shortcuts
- Using Code Workbench Auto Replace Shortcuts in the ISQL Window

Enabling Auto Replace Shortcut Substitution

Before configured Auto Replace shortcuts can be automatically substituted with the associated full terms, you must first enable the Auto Replace feature.

To enable Auto Replace

1. Select Tools > Code Workbench to open the **Code Workbench** dialog
2. Select **Enable Auto Replace**.

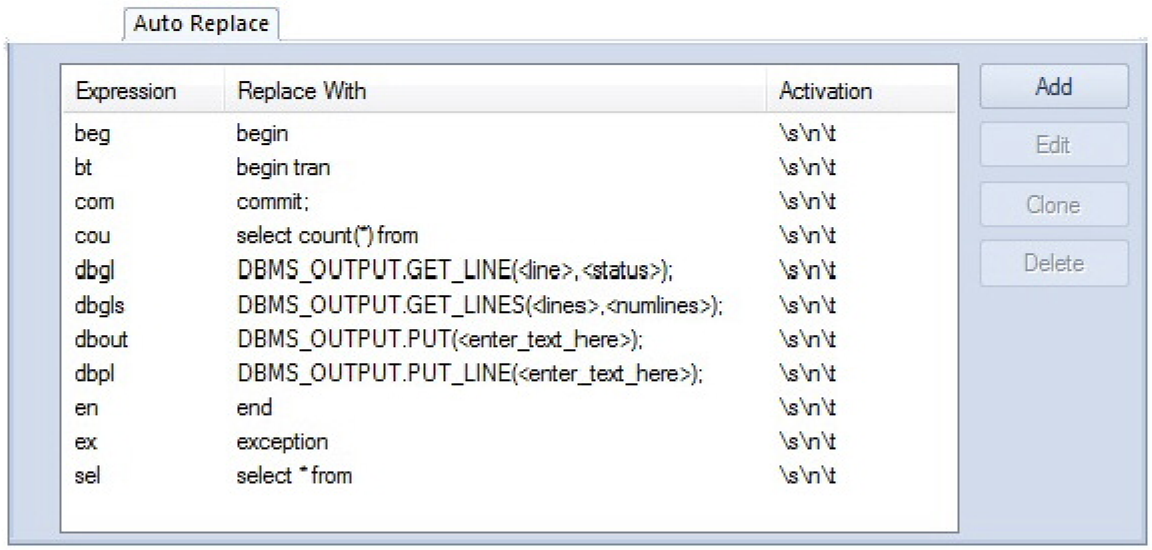
Note: If **Replace substrings** is unselected, only blank-delimited occurrences of auto replace expressions are replaced when an activation event occurs. If selected, all occurrences of auto replace expressions are replaced. For example, consider an Auto Replace expression of **BEG** that is to be replaced 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.

Also see the following topics:

- Working with Code Workbench Auto Replace Shortcuts, for an introduction
- Configuring Auto Replace Shortcuts, for details on setting up shortcuts
- Using Code Workbench Auto Replace Shortcuts in the ISQL Window, for details on invoking Auto Replace.

Configuring Auto Replace Shortcuts

The **Code Workbench** window's **Auto Replace** tab lets you maintain your list of Auto Replace shortcuts.



Expression	Replace With	Activation
beg	begin	\s\n\t
bt	begin tran	\s\n\t
com	commit;	\s\n\t
cou	select count(*) from	\s\n\t
dbgl	DBMS_OUTPUT.GET_LINE(<line>,<status>);	\s\n\t
dbgls	DBMS_OUTPUT.GET_LINES(<lines>,<numlines>);	\s\n\t
dbout	DBMS_OUTPUT.PUT(<enter_text_here>);	\s\n\t
dbpl	DBMS_OUTPUT.PUT_LINE(<enter_text_here>);	\s\n\t
en	end	\s\n\t
ex	exception	\s\n\t
sel	select * from	\s\n\t

Each entry in the list consists of the following:

- The shortcut
- The full term that will be substituted for the shortcut
- The keystroke events that trigger substitution of the term for the short cut.

To open the Code Workbench and navigate to the Auto Replace tab

1. Select Tools > Code Workbench.

To add a shortcut to the list

1. On the **Auto Replace** tab, click **Add**.
2. In the **Expression** box, type a new expression.
3. In the **Activation** box, press one or more of SPACE, TAB, or NEWLINE to set those as activation keys for this shortcut.
4. In the **Replace With** box, type the replacement string.
5. Click **OK**.

The replace expression is now ready for use in the ISQL Window.

To delete a shortcut from the list

1. Select one or more shortcuts from the list, click **Delete**, and confirm when prompted.

To edit an existing shortcut

1. Select a shortcut from the list, click **Edit**, and modify the **Expression**, **Activation**, or **Replace With** boxes, as necessary.

Also see the following topics:

- Working with Code Workbench Auto Replace Shortcuts, for an introduction
- Enabling Auto Replace Shortcut Substitution, for details on enabling the feature
- Code Analyst (Tools Menu), for details on invoking Auto Replace.

Using Code Workbench Auto Replace Shortcuts in the ISQL Window

When Auto Replace is enabled and shortcuts are configured, you can type a replace shortcut in the SQL Editor instead of typing the associated expression. Each time you type one of the activation keys, all shortcuts associated with the activation key are replaced with the defined replacement expression.

Also see the following topics:

- Working with Code Workbench Auto Replace Shortcuts, for an introduction
- Enabling Auto Replace Shortcut Substitution, for details on enabling the feature
- Configuring Auto Replace Shortcuts, for details on setting up shortcuts

Importing and Exporting Settings in Code Workbench

This feature helps standardize your application settings. Code Workbench settings can be exported to an **.xml** file, sent to another user, and subsequently imported by that user.

Exporting Settings

1. Select Tools > Code Workbench.
2. Click **Export Settings**. Code Workbench opens a **Save as** dialog box.
3. Use the dialog to specify a location and file name, and then click **Save**. Code Workbench saves your settings as an **.xml** file.

Importing Settings

1. Select Tools > Code Workbench.
2. Click **Import Settings**. Code Workbench opens an **Open** dialog box.
3. Use the dialog to locate and select the **.xml** file, and then click **Open**.

Code Workbench imports the settings.

Topics

- Code Workbench

Customize (Tools menu)

The **Customize** dialog lets you specify user interface preferences and set up shortcuts.

Topics

- Customizing the Rapid SQL General User Interface Appearance

Options (Tools menu)

The **Tools** menu provides access to the Options Editor, used to configure features and set preferences. For details on using the editor and available options, see Specifying Application Preferences and Feature Options.

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. For details, see Query Builder.

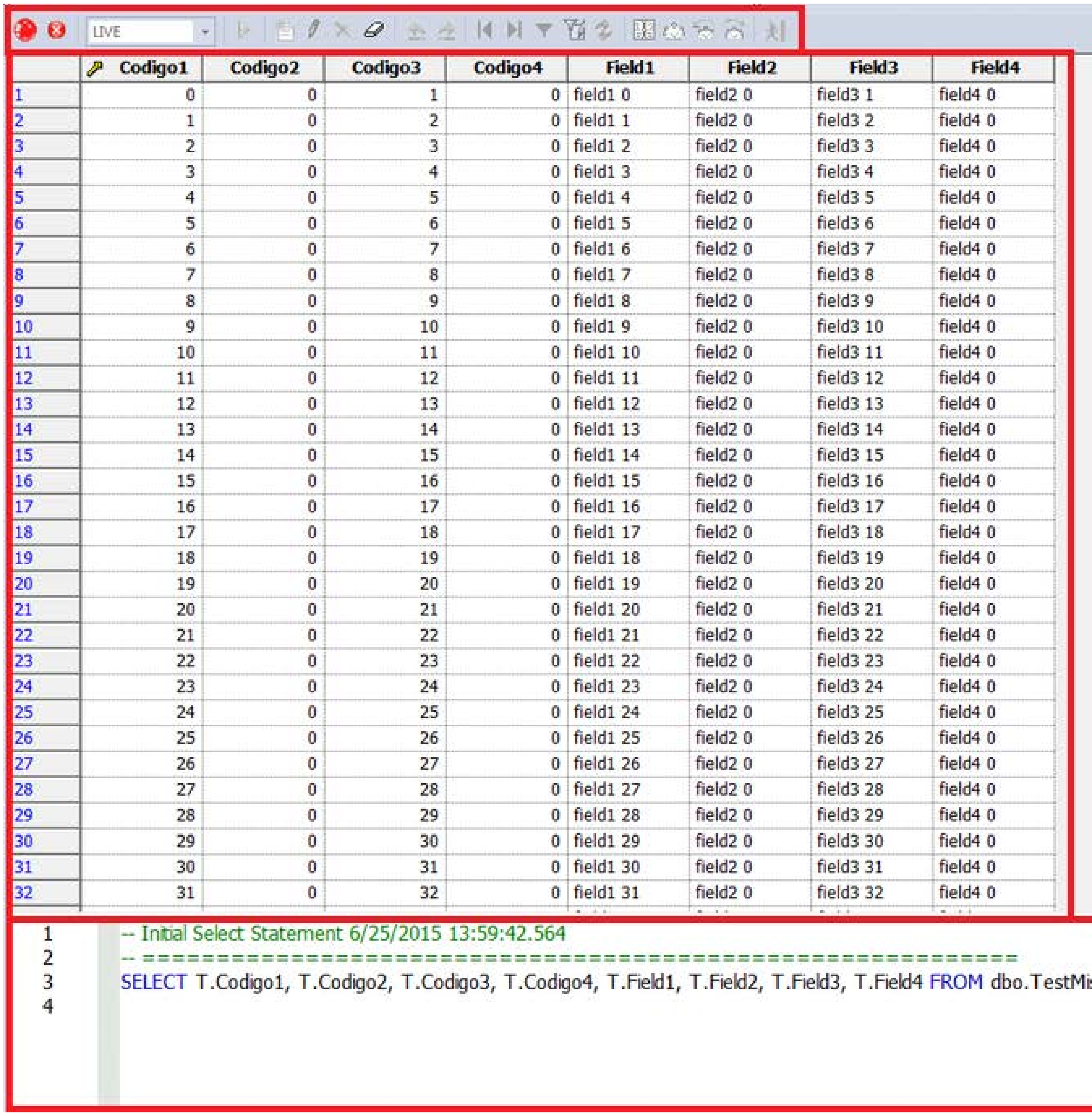
The Data Editor includes a Data Editor Filter that lets you select the columns in your table that you want to edit. You must select at least one column to use the Data Editor. The Data Editor Filter is not available for the Query Builder. For more information, see Data Editor Filter.

Tip: You can customize Data Editor options in the Options editor. For detals, see Data Editor Options.

You can find information about how to access the **Data Editor** in Using Data Editor.

As you can see in the figure, **Data Editor** consists of three main parts:

- Edit Window
- SQL Window
- Toolbar



	Codigo1	Codigo2	Codigo3	Codigo4	Field1	Field2	Field3	Field4
1	0	0	1	0	field1 0	field2 0	field3 1	field4 0
2	1	0	2	0	field1 1	field2 0	field3 2	field4 0
3	2	0	3	0	field1 2	field2 0	field3 3	field4 0
4	3	0	4	0	field1 3	field2 0	field3 4	field4 0
5	4	0	5	0	field1 4	field2 0	field3 5	field4 0
6	5	0	6	0	field1 5	field2 0	field3 6	field4 0
7	6	0	7	0	field1 6	field2 0	field3 7	field4 0
8	7	0	8	0	field1 7	field2 0	field3 8	field4 0
9	8	0	9	0	field1 8	field2 0	field3 9	field4 0
10	9	0	10	0	field1 9	field2 0	field3 10	field4 0
11	10	0	11	0	field1 10	field2 0	field3 11	field4 0
12	11	0	12	0	field1 11	field2 0	field3 12	field4 0
13	12	0	13	0	field1 12	field2 0	field3 13	field4 0
14	13	0	14	0	field1 13	field2 0	field3 14	field4 0
15	14	0	15	0	field1 14	field2 0	field3 15	field4 0
16	15	0	16	0	field1 15	field2 0	field3 16	field4 0
17	16	0	17	0	field1 16	field2 0	field3 17	field4 0
18	17	0	18	0	field1 17	field2 0	field3 18	field4 0
19	18	0	19	0	field1 18	field2 0	field3 19	field4 0
20	19	0	20	0	field1 19	field2 0	field3 20	field4 0
21	20	0	21	0	field1 20	field2 0	field3 21	field4 0
22	21	0	22	0	field1 21	field2 0	field3 22	field4 0
23	22	0	23	0	field1 22	field2 0	field3 23	field4 0
24	23	0	24	0	field1 23	field2 0	field3 24	field4 0
25	24	0	25	0	field1 24	field2 0	field3 25	field4 0
26	25	0	26	0	field1 25	field2 0	field3 26	field4 0
27	26	0	27	0	field1 26	field2 0	field3 27	field4 0
28	27	0	28	0	field1 27	field2 0	field3 28	field4 0
29	28	0	29	0	field1 28	field2 0	field3 29	field4 0
30	29	0	30	0	field1 29	field2 0	field3 30	field4 0
31	30	0	31	0	field1 30	field2 0	field3 31	field4 0
32	31	0	32	0	field1 31	field2 0	field3 32	field4 0

```
-- Initial Select Statement 6/25/2015 13:59:42.564
-- ==========================================================
SELECT T.Codigo1, T.Codigo2, T.Codigo3, T.Codigo4, T.Field1, T.Field2, T.Field3, T.Field4 FROM dbo.TestMi
```

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.

Data Editor SQL 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.

Data Editor Toolbar

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
Calendar button	Sets correct format for target date/time cell. Enables the Calendar window.
Date/Time Format Builder button	For details, see Editing Date and Time Functions.
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.

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.

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 Unicode characters is provided as follows:
- IBM DB2 for Windows, Unix, and Linux: **character**, **clob**, **varchar**, and **longvarchar** types
- SQL Server: **nchar**, **nvarchar**, **ntext**, and **nvarchar(max)** types.
- Oracle 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 ASE 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

Topics

- Using Data Editor

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. For details, see Editing Date and Time Functions.

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.

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:

- Editing Date and Time Functions

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, 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, the changes are committed 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 from the Data Editor Tab of the Options Editor. For details, see Editing date/time globally.	Options Editor
Grid	Lets you make changes to the date display of the entire Data Editor grid for that session only. For details, see Editing grid date/time.	Data Editor grid
Column	Lets you make changes to the date display of a single column in the Data Editor for that session only. For details, see Editing column date/time.	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 or Editing column date/time.

To edit the date and time globally, do the following:

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

The Options Editor opens.

2. On the Options Editor, click the Data Editor Tab.
3. On the Data Editor Tab, click ...

The Date/Time Format Builder dialog box opens.

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.

The Date/Time Format Builder dialog box opens.

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.

The date/time format changes are accepted the Date/Time Format Builder dialog box closes.

8. On the Options Editor, select the appropriate Default Date/Time Format options:

Option	Description
Use Calendar Control as default	If selected, the **Calendar Control** window is used. For details, see Editing Date and Time Functions.
Two-digit year system setting warning	If selected, a warning is sent when you use a two-digit year system setting.

9. Click OK.

The Default Date/Time Format changes are accepted the Options Editor closes.

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. For details, see Editing date/time globally.

To edit the grid date and time, do the following:

1. On the Datasource Navigator, select the target table.
2. Right-click the table, and then click Edit Data.

The Data Editor opens.

3. On the Data Editor tool bar, click Date/Time Format Builder.

The Date/Time Format Builder opens.

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.

The Date/Time Format Builder dialog box opens.

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.

The date/time format changes are accepted and the Date/Time Format Builder dialog box closes.

- 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. For details, see Editing grid date/time.

To make this display permanent, Editing Global Date/Time Format. For details, see Editing date/time globally.

To edit the column date and time, do the following:

1. On the Datasource Navigator, select the target table.

2. Right-click the table, and click Edit Data.

The Data Editor opens.

3. On the Data Editor, click the column header to select the column.
4. Right-click the column and click Format.

The Date/Time Format Builder opens.

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.

The Date/Time Format Builder dialog box opens.

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.

The date/time format changes are accepted and the Date/Time Format Builder dialog box closes.

- 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.

Add-on tools

The following are the available, license-specific tools.

- Embarcadero SQL Debugger - Provides detailed information about using the Embarcadero SQL Debugger. It Includes a step-by-step walk through to familiarize you with the features of the Embarcadero SQL Debugger.
- PL/SQL Profiler - Provides detailed information about using the Rapid SQL PL/SQL Profiler. Includes a step-by-step walk through to familiarize you with the features of the PL/SQL Profiler.
- Code Analyst - Identifies time-consuming lines of code.

Topics

- Licensing Your Product

Embarcadero SQL Debugger

The Embarcadero SQL Debugger lets you locate and fix bugs in the following elements:

- Procedures (IBM DB2 for Linux, Unix, and Widows, Microsoft SQL Server, Oracle, Sybase ASE)
- Functions (Oracle)
- Triggers (IBM DB2 for Linux, Unix, and Widows, Microsoft SQL Server, Oracle, Sybase ASE)

Note: The SQL Debugger lets you debug triggers by debugging the procedures that call them.

Note: For Oracle, you cannot debug packages, but you can debug the functions and procedures within packages.

Note: For Oracle, you cannot debug any objects contained in the Exclusion List. For more information, see Editing the Exclusion List.

The table below describes the sections of this chapter:

Section	Description
SQL Debugger Modules and DBMS Support	Describes the specific SQL Debugger modules and DBMS versions supported.
Embarcadero SQL Debugger Features	Provides an overview of SQL Debugger functionality.
Setting up the Embarcadero SQL Debugger	Provides details on requirements and setup tasks for each supported DBMS.
Embarcadero SQL Debugger Interface	Describes the Embarcadero SQL Debugger graphical interface that includes an editor window and four debug view windows.
Embarcadero SQL Debugger Functionality	This section describes the functionality on the SQL Debugger.
Using the Embarcadero SQL Debugger	This section describes how to run a debug session.
Tutorial Sessions	Provides walkthrough sessions for new users.

SQL Debugger Modules and DBMS Support

There are four Embarcadero SQL Debugger modules, each corresponding to a supported DBMS. Modules and DBMS support are as follows:

- **Embarcadero SQL Debugger for IBM DB2 for Linux, Unix, and Windows:** supports all versions supported by Rapid SQL
- **Embarcadero SQL Debugger for Microsoft:** supports all versions supported by Rapid SQL
- **Embarcadero SQL Debugger for Oracle:** supports all versions supported by Rapid SQL
- **Embarcadero SQL Debugger for Sybase ASE:** supports all versions supported by Rapid SQL

Each SQL Debugger version is an optional and separate add-on module.

Embarcadero SQL Debugger Features

The Embarcadero SQL Debugger lets you identify problems within your code. The Embarcadero SQL Debugger 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 main menu, the Procedures window, the DDL Editor and ISQL windows.

The Embarcadero SQL Debugger 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.
Insert or Remove Breakpoint	Lets you specify positions in a program where the debugger stops execution.

To set specific Debugger values on the Options Editor, see Embarcadero SQL Debugger Options.

Setting up the Embarcadero SQL Debugger

The following topics describe requirements and tasks for those DBMS that have setup requirements:

- Embarcadero SQL Debugger for Linux, Unix, and Windows Setup
- The Embarcadero SQL Debugger for Microsoft Setup
- Embarcadero SQL Debugger for Oracle Setup

In addition, you should set your debugger preferences at this time. For details, see Embarcadero SQL Debugger Options.

Embarcadero SQL Debugger for Linux, Unix, and Windows Setup

Embarcadero SQL Debugger for IBM DB2 for Linux, Unix, and Windows requires proper configuration of the server and client.

For more information, see:

- Prerequisites for Adding and Compiling Stored Procedures
- Configuring the IBM DB2 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 for Linux, Unix, and Windows requires the following products and components.

Client

- IBM DB2 for Linux, Unix, and Windows
- 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 for Windows/Unix Support
- Administration and Configuration Tools

Server

- IBM DB2 for Linux, Unix, and Windows
- DB2 Enterprise Edition
- Communications Protocols
- Stored Procedure Builder
- Applications Development Interfaces
- System Bind Files
- DB2 Connect Server Support
- Documentation
- Base DB2 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 and includes the C compiler options needed to compile the procedure on the server. If it is not found, install the IBM DB2 Administration and Configuration Tools option on the server.

Configuring the IBM DB2 for Linux, Unix, and Windows Server for Procedures

You can create procedures on the targeted server.

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: Availability of this feature depends on your licensing.

Topics

- Licensing Your Product

The Embarcadero SQL Debugger for Microsoft Setup

Embarcadero SQL Debugger for Microsoft requires proper configuration of Server Requirements and Client Requirements.

Important Note about Microsoft SQL Server 2000 Service Pack 4 (SP4)

When running Windows XP SP2, SQL Server 2000 SP4 is required on both the client and the server. To verify this is the case start MS Query Analyzer and press Help > About. The Version SQL should be 8.00.2039.

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 T-SQL Debugger.

You may receive the following error message if you try to use the T-SQL Debugger:

"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 has disabled T-SQL Debugger for Application using earlier T-SQL Debugger clients for security reasons. To enable T-SQL Debugger for these Applications, a member of the sysadmins server role, such as sa must explicitly enable debugging by running the following code:

```
Exec sp_sdidebug 'legacy_on'
```

Note: You must repeat this procedure whenever you restart the server.

Server Requirements

Embarcadero SQL Debugger for Microsoft requires:

- Windows 2005
- Windows 2000
- Microsoft SQL Server version 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. 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, install them before running the Embarcadero SQL Debugger for Microsoft.

To install the Debugger Interface subcomponent on the server after the initial installation:

1. Start Microsoft Visual Studio, Enterprise Edition Setup.

OR

Start Microsoft SQL Server Setup.

2. Select Custom Install.

Microsoft SQL Server opens the **Select Components** dialog.

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

To configure the service, see the instructions for your server operating system.

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.

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 who will be using the Embarcadero SQL Debugger for Microsoft.

Note: This person needs 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 Note about Microsoft SQL Server 2000 Service Pack 4 (SP4)

When running Windows XP SP2, SQL Server 2000 SP4 is required on both the client and the server. When running Windows XP SP2, SQL Server 2000 SP4 is required on both the client and the server. To verify this is the case start MS Query Analyzer and press Help > About. The Version SQL should be 8.00.2039.

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 T-SQL Debugger.

You may receive the following error message if you try to use the T-SQL Debugger:

"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 has disabled T-SQL Debugger for Application using earlier T-SQL Debugger clients for security reasons. To enable T-SQL Debugger for these Applications, a member of the sysadmins server role, such as sa must explicitly enable debugging by running the following code:

```
Exec sp_sdidebug 'legacy_on'
```

Note: You must repeat this procedure whenever you restart the server.

Configuring DCOM on the Server

To configure DCOM on the server, 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.

4. Click the **Default Security** tab.
5. In the Default Access Permissions box, click **Edit Default**.

Windows opens the **Registry Value Permissions** dialog.

6. Click **Add**.

Windows opens the **Add Users and Groups** dialog.

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, grant them remote access on the server. To grant remote access, 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
- Client Connectivity
- Installing the Microsoft SQL Debugger Interface Subcomponent

Operating System

The client must be running one of the following operating systems:

- Microsoft Windows 95
- Microsoft Windows 98
- Microsoft Windows XP

Client Connectivity

When running the Debugger on SQL Server Clients

Before you proceed, verify that the file `ssdebugps.dll` is registered on the client machine. This file is REQUIRED for debugging and Microsoft only installs it on the server machine and not as a part of a client-only install.

Next you must configure DCOM by following these steps:

1. At a command prompt, type `dcomcnfg`, and then press ENTER.

Component Services opens.

2. In Component Services, expand Component Services, expand Computers, and then expand My Computer.
3. On the toolbar, click the **Configure My Computer** button.

The **My Computer** dialog appears.

4. In the **My Computer** dialog, click the **COM Security** tab.
5. Under Access Permission, click **Edit Limits**.

The **Access Permission** dialog appears.

6. Under Group or user names, click **ANONYMOUS LOGON**.
7. Under **Permissions** for ANONYMOUS LOGON, select the **Remote Access** check box, and then click **OK**.

Finally, you need to go to the Default Properties tab of the MyComputer dialog.

1. Select None in the Default Authentication Level dropdown menu.
2. Select Impersonate in the Default Impersonation Level dropdown menu.

For more detailed information, you can refer to Microsoft's Help: :http://support.microsoft.com/default.aspx?kbid=833977#XSLTH4134121124120121120120

When running Windows XP SP2, SQL Server 2000 SP4 is required on both the client and the server.

For Microsoft SQL the client must have the Client Connectivity component of Microsoft SQL Server.

For Microsoft SQL Server

The client must have the Development Tools, Debugger Interface subcomponent of Microsoft SQL Server. 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, 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, do the following:

1. Start the Microsoft SQL Server Setup program.
2. Select Custom Install.

Microsoft SQL Server opens the Select Components dialog.

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 Oracle Setup

The only task required in setting up the Embarcadero SQL Debugger for Oracle is Editing the Exclusion List.

Editing the Exclusion List

Upon installation, an Exclusion List is set up on your computer which includes packages that the application cannot debug. The Exclusion List is located in the main product directory. You can add or remove packages from this file by 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 Microsoft 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: Press ENTER 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 not debug a package function or package procedure listed on the Exclusion List.

4. Save the changes to `deborcex.etd`.

Embarcadero SQL Debugger Options

You can specify debugger options from the **Debug** tab of the Options editor.

Topics

- Debug Options

Embarcadero SQL Debugger Interface

The Embarcadero SQL Debugger includes an editor window and four debug view windows. When you open a debug session, the code for the object is extracted 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 includes five windows:

- DDL Editor Window
- Watch Window
- Variables Window
- Call Stack Window
- Dependency Tree Window

Working with SQL Debugger Windows

You can resize, move, dock and float the following windows:

- Watch Window
- Variables Window
- Call Stack Window
- Dependency Tree Window

The process to do it is:

1. To resize the target window, click its frame and drag it.
2. To move and dock the target window, click its grab bar and drag it.
3. To float the target window, press Shift, then click its grab bar and drag it.

DDL Editor Window

The DDL Editor displays your code in read-only format. When you start debugging, the Embarcadero SQL Debugger extracts your code into a DDL Editor. The DDL Editor uses the default syntax coloring.

Note: For Oracle, LOB datatypes and REF CURSOR variables are displayed in the **Results** tab.

Topics

- Embarcadero SQL Debugger Interface

Watch Window

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 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: 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 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 @. Drag the @ to the Watch Window.

2. In the Watch window, change the value of the variable.
3. On the DDL Editor, click **Debug** or **Go**.

The Embarcadero SQL Debugger 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 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 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.

A ScreenTip displaying the current value of that variable displays.

Topics

- Embarcadero SQL Debugger Interface

Call Stack Window

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, a green arrow indicates the line of the referenced call.

Topics

- Embarcadero SQL Debugger Interface

Dependency Tree Window

The Dependency Tree window displays any external database objects the script accesses. These database objects are displayed 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.

The SQL of the target object is displayed in the DDL Editor window.

Topics

- Embarcadero SQL Debugger Interface

Embarcadero SQL Debugger Functionality

The Embarcadero SQL Debugger offers 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

You must be running a debugging session to use these functions.

Topics

- Opening 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, a dialog box prompts you for the input parameters when you open a debugging session.

The **Procedure Execution** dialog 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 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, 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 or Continue button	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.

Topics

- Embarcadero SQL Debugger Functionality

Step Into

Step Into lets you execute the current instruction. If the current instruction makes a call to a stored SQL object, the Embarcadero SQL Debugger steps inside the nested child object.

To use the Step Into facility, do the following:

1. On the **Debug** menu, select **Step Into**.

OR

Press F11.

The Embarcadero SQL Debugger moves the arrow to execute the current instruction.

Topics

- Embarcadero SQL Debugger Functionality

Step Out

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, select **Step Out**.

OR

Press SHIFT+F11.

The Embarcadero SQL Debugger stops stepping through the current object and executes the remainder of the script.

Topics

- Embarcadero SQL Debugger Functionality

Step Over

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, select **Step Over**.

OR

Press F10.

The Embarcadero SQL Debugger executes the current instruction.

Topics

- Embarcadero SQL Debugger Functionality.

Run To Cursor

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 places the cursor on the target line.

3. On the **Debug** menu, select **Run to Cursor**.

OR

Press CTRL+F10.

The Embarcadero SQL Debugger executes all instructions between the pointer and the cursor.

Topics

- Embarcadero SQL Debugger Functionality

Insert or Remove Breakpoint

A breakpoint is a position in a program where a debugger stops execution. When you start debugging, Embarcadero SQL Debugger opens the script in a DDL Editor. A yellow arrow pointer indicates which line the Embarcadero SQL Debugger executes next.

The Embarcadero SQL Debugger executes all lines of code between the yellow arrow and the first breakpoint. If no breakpoints are present, Embarcadero SQL Debugger 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 Breakpoints, temporarily disable or enable breakpoints without having to add or remove breakpoints.

The Embarcadero SQL Debugger 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.

Breakpoints are stored so that when you debug the same script on separate occasions, you can reuse the same breakpoints. 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, select **Breakpoint**.

OR

Press F9.

The Embarcadero SQL Debugger inserts a new breakpoint or removes an existing breakpoint on the target line of code.

Topics

- Embarcadero SQL Debugger Functionality

Toggle Breakpoint

After inserting a breakpoint, Toggle Breakpoint lets you enable or disable that breakpoint. Embarcadero SQL Debugger 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 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, select **Enable/Disable Breakpoint**.

OR

Press CTRL+F9.

The Embarcadero SQL Debugger toggles the breakpoint indicated by the pointer.

Topics

- Insert or Remove Breakpoint

Go

Go lets you execute all instructions stopping only when it encounters a breakpoint or when the program is complete.

To use the Go facility, do the following:

1. On the **Debug** menu, select **Go**.

OR

Press F5.

The Embarcadero SQL Debugger executes all instructions.

Topics

- Embarcadero SQL Debugger Functionality

Stop

Stop lets you halt the script execution and terminate the session.

To use the Stop facility, do the following:

1. On the **Debug** menu, select **Stop Debugging**.

OR

Press SHIFT+F5.

The Embarcadero SQL Debugger stops the script execution and terminates the session.

Topics

- Embarcadero SQL Debugger Functionality

Restart

Restart lets you terminate the current debug session and open a new one. When the new session opens, you are prompted for new input parameters.

To use the Restart facility, do the following:

1. On the **Debug** menu, select **Restart**.

OR

Press CTRL+SHIFT+F5.

The Embarcadero SQL Debugger restarts the debug session.

Topics

- Embarcadero SQL Debugger Functionality.

Break

Break lets you pause the debug session.

To use the Break facility, do the following:

1. On the **Debug** menu, select **Break**.

The Embarcadero SQL Debugger suspends the debug session.

Topics

- Embarcadero SQL Debugger Functionality.

Close

Close lets you close the DDL Editor and the Embarcadero SQL Debugger.

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 closes the debug session.

Topics

- Embarcadero SQL Debugger Functionality

Using the Embarcadero SQL Debugger

This section offers a general overview of how to use Embarcadero SQL Debugger's full range of debugging functionality.

- Opening a Debugging Session
- Debugging an SQL Script with Embarcadero SQL Debugger

Opening a Debugging Session

When you start a debugging session, the Embarcadero SQL Debugger interface opens. For details, see Embarcadero SQL Debugger Interface.

If the target script requires input parameters, a dialog box prompts you for the necessary input parameters before displaying the target code in the SQL Editor window. When the target script is displayed in the SQL Editor window, you can begin debugging. For details, see Debugging an SQL Script with Embarcadero SQL Debugger

Note: Embarcadero SQL Debugger for IBM DB2 for Linux, Unix, and Windows, Embarcadero SQL Debugger for Sybase, and Embarcadero SQL Debugger for Microsoft only let you debug the SQL script of procedures or triggers.

To debug an object, do the following:

1. In the Navigator, click the object node.

The objects are displayed.

2. Right-click the target object, and select **Open**.

The Object Editor opens.

OR

Right-click the target object and select **Extract**.

The DDL Editor opens, displaying the code of the target object.

3. On the toolbar, click **Debug**.

OR

In the DDL Editor window, right-click, and then click **Debug**.

If the script requests input parameters, a dialog box prompts for parameter values. If the script does not require input parameters, the script is displayed in the DDL Editor window for you to begin debugging. For details, see Debugging an SQL Script with Embarcadero SQL Debugger.

Note: You cannot use the Embarcadero SQL Debugger until it has fully initialized.

4. In the dialog box, specify the appropriate parameter values, and then click **OK** or **Continue**.

The script opens in the DDL Editor window.

Note: If the script requires Sybase ASE or Oracle types (tables, records, or Booleans) as input parameters, the Embarcadero SQL Debugger generates an anonymous block.

Debugging an SQL Script with Embarcadero SQL Debugger

After you open a debugging session and enter any required input parameters, you can begin working with your script in the Embarcadero SQL Debugger.

Debugging an SQL Script

To debug a SQL Script, do the following:

1. On the **Debug** menu, click one of the SQL Debugger options (Step Into, Step Over, and so forth) or click **Go**.

Note: You can monitor the progress of your debug session in the Variables window.

2. On the **Debug** menu, select **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 DDL Editor).

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.

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.

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 Dependency Tree Window.
3. When you finish debugging the script, click **Close** or **Exit**.

The T-SQL Debugger DDL Editor closes.

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.

Tutorial Sessions

The following topics provide walkthrough sessions geared to new users of the Embarcadero SQL Debugger:

- Debugging a Sample Script with Embarcadero SQL Debugger for Microsoft
- Debugging a Sample Script with Embarcadero SQL Debugger for Oracle

Debugging a Sample Script with Embarcadero SQL Debugger for Microsoft

This Getting Started section demonstrates basic debugging functionality. You will debug two procedures using the Embarcadero SQL Debugger for Microsoft.

The section topics are designed to familiarize you with basic debugging features and functionality:

- Getting Started with Embarcadero SQL Debugger for Microsoft
- Testing a Procedure
- Starting the Debugging Session
- Breakpoints
- Step Into
- Step Out
- Correcting the Script

Getting Started with Embarcadero SQL Debugger for Microsoft

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 will use the Embarcadero SQL Debugger for Microsoft to identify this bug.

The Getting Started section includes:

- 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, 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".

1. Connect to a Microsoft SQL Server datasource.
2. On the **Datasource** menu, select the database node, and then select the target database.

Note: For this walk-through, we recommend that you select a non-production database.

3. On the Main toolbar, click **New**.

OR

Press CTRL+N.

An SQL Editor opens in the current workspace.

4. In the SQL Editor, type the DDL for procedure check_modulo.

Note: Substitute your user name once in the DDL for this procedure.

5. On the SQL Editor toolbar, click **Execute**.

The script executes 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 will use the Embarcadero SQL Debugger for Microsoft to identify this bug.

Caution: When you input parameters, enter the smaller number in the @p_num1_in int box.

To create this procedure, 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

1. Connect to a Microsoft SQL Server datasource.
2. On the **Datasource** menu, select the database node, and then select the target database.

Note: For this walk-through, we recommend that you select a non-production database.

3. On the Main toolbar, click **New**.

OR

Press CTRL+N.

An SQL Editor opens in the current workspace.

4. In the SQL Editor, type the DDL for procedure calculate_sum_with_overflow_bug.

Note: Substitute your user name twice in the DDL for this procedure.

5. On the SQL Editor toolbar, click **Execute**.

The script executes 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 the Datasource Navigator.

Confirming the Creation of the Procedures

To confirm the creation of the procedures, do the following:

1. On the **Navigator** tab, click the Navigatorlist, and then click **Organize By Owner**.

The **Navigator** tab refreshes with the new display configuration.

2. On the **Navigator** tab, double-click the **Databases** node, and then double-click the target database node.

Tthe list of object owners is displayed.

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.

Testing a Procedure

After you confirm the creation of the procedures, 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, enter the smaller number in the @p_num1_in int box.

Testing a Procedure

To test a procedure, do the following:

1. On the **Navigator** tab, right-click **calculate_sum_with_overflow_bug**, and then click **Execute**.

The Procedure Execution window opens.

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**.

The procedure is compiled a **Results** tab opens, 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 **Navigator** tab, right-click **calculate_sum_with_overflow_bug**, and then click **Execute**.

The Procedure Execution window opens.

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**.

The error returned states “Arithmetic overflow occurred”.

Starting the Debugging Session

After you test the procedure, open the procedure in Embarcadero SQL Debugger for Microsoft and enter input parameters before debugging.

To start the debugging session, do the following:

1. On the **Navigator**tab, right-click the procedure, **calculate_sum_with_overflow_bug**, and then click **Debug** to start the debug session.

The DDL for the procedure is extracted into a DDL Editor and opens the **Procedure Execution** dialog.

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**.

The dialog box closes.

The Embarcadero SQL Debugger includes the following five windows:

- DDL Editor Window
- Watch Window
- Variables Window
- Call Stack Window
- Dependency Tree Window

Breakpoints

After you start the debugging session, insert a breakpoint into the code of the procedure calculate_sum_with_overflow_bug. Then 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

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, select **Breakpoint**.

OR

Press F9.

A breakpoint (indicated by a dot) is inserted next to the number of the target line.

3. On the **Debug** menu, select **Go**.

OR

Press F5.

Embarcadero SQL Debugger for Microsoft displays the value of the variables before the breakpoint in the Variables Window.

Step Into

After setting the breakpoint, step into the dependent procedure, check_modulo.

To use the Step Into facility, do the following:

1. On the **Debug** menu, select **Step Into**.

OR

Press F11.

The DDL for the dependent, nested procedure is extracted into the DDL Editor.

2. Step Into again.

The next part of the code executes and displays the values for the variables in the Variables Window.

The Call Stack Window displays calls to the procedures.

Step Out

After you Step Into the modulo_check (nested procedure) code, step back out and return to the calculate_sum_with_overflow_bug (outside procedure) code.

To use the Step Out facility, do the following:

1. On the **Debug** menu, select **Step Out**.

OR

Press SHIFT+F11.

The DDL Editor opens, containing the code for calculate_sum_with_overflow_bug.

2. On the **Debug** menu, select **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, the Debugger closes a DDL Editor opens to the **Results** tab.

Correcting the Script

When you finished Stepping Out of the nested code and encounter the error, to fully fix the bug, do the following:

1. Locate the source of the error
2. Scroll to the line in the script displaying the error
3. Analyze the code
4. Correct the error
5. Compile the corrected script

When you first executed the procedure, the error message “Arithmetic overflow error for data type tinyint, value = 450” was displayed. 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, the error message was returned and the procedure fails.

To correct the script, do the following:

1. On the **Navigator**tab, right-click **calculate_sum_with_overflow_bug**, and then click **Extract**.

Tthe DDL for the procedure is extracted into a DDL Editor.

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

The **Find** dialog opens.

3. In the **Find What** box, type **TINYINT**.
4. Click **Find Next**.

The first occurrence of TINYINT is selected.

5. Change the data type for @result from TINYINT to INT.

6. On the DDL Editor toolbar, click **Execute** to execute the modified script.

The script executes and the **Results** tab opens.

7. On the **Navigator** tab, right-click **calculate_sum_with_overflow_bug**, and then click **Execute**.

The **Procedure Execution** dialog opens.

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**.

The procedure executes with the new data type and opens the **Results** tab, returning the value 450. You successfully corrected the script and debugged the procedure.

Debugging a Sample Script with Embarcadero SQL Debugger for Oracle

The 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 familiarize you with basic debugging features and functionality:

- Getting Started with Embarcadero SQL Debugger for Oracle guides you through creating the package you use in Debugging Sample Script 1 and Debugging Sample Script 2.
- Debugging Sample Script 1 guides you through debugging functionality and demonstrates the Embarcadero SQL Debugger for Oracle interface features.
- Debugging Sample Script 2 guides you through debugging functionality and error correction.

Note: For the purposes of this walk-though we have created this package under the user name DEMO_SPENCE.

Getting Started with Embarcadero SQL Debugger for Oracle

The 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.

On installation the script is placed in the \UsrScrpt subfolder of the main installation directory.

If you create the package included with the installation, you can delete it and its objects from your system when you finish working with them. The objects to delete:

- 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

Embarcadero SQL Debugger for Oracle Overview

The Getting Started section guides you through:

- Opening the sample debug script.
- Executing sample debug script.
- Changing the **Navigator** tab display.
- Confirming the creation of the package, including its functions and procedures.

Getting Started

1. On the **File** menu, select **Open**.

The **Open File(s)** dialog opens.

2. In the **Open File(s)** dialog, go to DBA900\UsrScrpt\DEBUGGER_DEMO.sql, and then click **Open**.

Note: During the installation the DEBUGGER_DEMO.sql is placed in the \UsrScrpt subfolder of the main installation folder.

The **What type of file** dialog opens.

3. On the **What type of file** dialog, click **The file includes the DDL to create a database object**, and then click **OK**.

The target script opens in an SQL Editor.

4. On the SQL Editor toolbar, click **Execute** to execute the script and create the package.

The target script executes 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.

5. On the **Navigator** tab list, click **Organize by Owner**.

A list of owners is displayed.

6. On the Navigator, double-click your owner name.

A list of your schema objects is displayed.

7. Under your owner node, double-click the **Packages** node.

A COUNT_TIME_INTERVAL displayed, confirming the package’s creation.

Debugging Sample Script 1

Sample Script 1 demonstrates Embarcadero SQL Debugger's basic features and functionality with the function WORKING_DAYS( ), which counts the number of business days between two dates.

Debugging Sample Script 1 is divided into five parts:

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( ), start a debug session.

Starting the Debug Session

To start the debug session, do the following:

1. On the **Navigator** tab, under the **Packages** node, double-click the COUNT_TIME_INTERVAL node.

The COUNT_TIME_INTERVAL node opens and displays the following items:

2. Under the COUNT_TIME_INTERVAL node, double-click **Functions**.

The Functions node opens and displays the following items:

3. Under the **Functions** node, right-click WORKING_DAYS ( ), and then click **Debug** to start the debug session.

The **Function Execution** dialog opens with the current date in the boxes.

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.

Input Parameters

To enter input parameters, do the following:

1. Click the **P_START_DATE DATE** box, and then click the drop-down arrow.

A calendar opens.

2. On the calendar, click the left arrow to set the month to **November 1999**.

3. Click **1**.

11/01/1999 is displayed in the Value column of P_START_DATE.

4. Click the **P_END_DATE DATE** box, and then click the drop-down arrow.

A new calendar opens.

5. On the calendar, click the left arrow to set the month to **November 1999**.
6. Click **8**.

11/08/1999 is displayed in the Value column of P_END_DATE.

7. Click **OK**.

The **Function Execution** dialog closes and then opens the following five Embarcadero SQL Debugger for Oracle interface windows:

- DDL Editor Window, which displays the SQL code for the function.
- Watch Window
- Variables Window
- Call Stack Window
- Dependency Tree Window, which displays the dependent objects.

Sample Script 1- Inserting Breakpoints

After you input parameters in the **Input Parameters** dialog, you can begin inserting breakpoints. In this example, you insert the breakpoints in the extracted dependent object code. After you extract this code, 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.

The SQL code for the package body opens in the SQL Editor window.

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

The **Find** dialog opens.

3. On the **Find** dialog, in the **Find What** box, type DBMS_OUTPUT.
4. Click **Find Next**.

In the SQL Editor, the first occurrence of DBMS_OUTPUT, on line 22 is highlighted.

5. On the SQL Editor toolbar, click **Breakpoint**.

A breakpoint is inserted next to the target line number.

6. On the **Find** dialog, click **Find Next**.

Tthe next occurrence of DBMS_OUTPUT is highlighted.

7. Click **Find Next** a third time.

The next occurrence of DBMS_OUTPUT, on line 35, is highlighted.

8. On the **Find** dialog, click **Cancel**.

The **Find** dialog closes.

9. On the **Edit** toolbar, click **Breakpoint** to insert a second breakpoint.

You should now have breakpoints set at lines 22 and 35.

Sample Script 1- Stepping Into

After you insert breakpoints, you can step into the function code.

Step Into

To use the Step Into facility, 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 runs to the next breakpoint.

7. On the SQL Editor toolbar, click **Go** once more.

Embarcadero 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 Results

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

To debug session results, do the following:

1. Click **OK**.

The Debug Session Results box closes and your debug session terminates.

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, use Embarcadero SQL Debugger for Oracle to identify the bug, and then correct the script so that it can execute successfully.

Debugging Sample Script 2 is divided into six parts:

- Sample Script 2 - Executing the Function
- Sample Script 2 - Starting the Debug Session
- Sample Script 2 - Entering Input Parameters
- Sample Script 2- Inserting Breakpoints
- Sample Script 2- Stepping Into
- Sample Script 2 - Correcting the Script

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 ( ), 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 **Navigator** tab, under the **Packages** node, double-click the **COUNT_TIME_INTERVAL** node.

The **COUNT_TIME_INTERVAL** node opens.

2. Double-click the **Functions** node.

The Functions node opens.

3. Click **WEEKEND_DAYS ( )**, then right-click it and click **Execute**.

The **Function Execution** dialog opens.

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**.

The attempt to execute the function fails and returns an error indicating that the character string buffer is too small.

Sample Script 2 - Starting the Debug Session

After you unsuccessfully execute the function WEEKEND_DAYS( ) and determine 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 **Navigator** tab, under the COUNT_TIME_INTERVAL node, under the Functions node, right-click **WEEKEND_DAYS ( )**, and then click **Debug** to start the debug session.

The **Function Execution** dialog opens.

Sample Script 2 - Entering Input Parameters

After you start the debug session, you can enter input parameters in the **Function Execution** dialog.

Entering Input Parameters

To enter input parameters, do the following:

1. At the end of the **P_START_DATE** row, click the drop-down arrow.

The calendar opens.

2. On the calendar, click Left Arrow to set the month to **November 1999**.
3. Click **1**.

11/01/1999 is displayed 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.

A new calendar opens.

5. On the calendar, click Left Arrow to set the month to **November 1999**.
6. Click **30**.

11/08/1999 is displayed in the **Value** column of the **P_END_DATE** row.

7. Click **Continue**.

Tthe **Function Execution** dialog gloses and then opens the following five Embarcadero SQL Debugger for Oracle interface windows:

- DDL Editor Window, which displays the SQL code for the function
- Watch Window
- Variables Window
- Call Stack Window
- Dependency Tree Window, which displays the dependent objects.

Sample Script 2- Inserting Breakpoints

After you enter input parameters, you can begin inserting breakpoints. In this example, you insert the breakpoints in the extracted dependent object code. After you extract this code, 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.

Tthe SQL code for the package body opens in the SQL Editor.

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

The **Find** dialog opens.

3. On the **Find** dialog, 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, click **Cancel**.

The **Find** dialog closes.

5. Click line 60, the first line of executable code:
6. On the **SQL Editor** toolbar, click **Breakpoint**.

A breakpoint is inserted 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.

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, monitor the Variables window. As you step through the code, the Variables window updates with the value of the variables.

Step Into

1. On the SQL Editor toolbar, click **Step Into**.

The yellow arrow moves 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**.

The yellow arrow moves to the next line of the code, line 66.

4. On the SQL Editor toolbar, click **Step Into**.

The yellow arrow moves 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**.

The yellow arrow moves 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**.

The yellow arrow moves 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.

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:

- 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, 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 **Navigator** tab, under the **Packages** node, under the COUNT_TIME_INTERVAL. node, right-click **Package Body**, and then click **Extract**.

The package body is extracted into an SQL Editor.

2. In the SQL Editor, scroll to line 57, the line defining the variable **v_theda**y.
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.

The scrip executes successfully.

5. On the **Navigator** tab, under the **COUNT_TIME_INTERVAL** package node, under the **Functions** node, click **WEEKEND_DAYS ( )**.
6. Right-click **WEEKEND_DAYS ( )**, and then click **Execute**.

The Function Execution dialog opens.

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.

The corrected function executes successfully.

PL/SQL Profiler

The PL/SQL Profiler module lets Oracle developers 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. The PL/SQL Profiler collects and stores data in database tables that let you identify and isolate performance problems and provide code coverage information. The 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

Requirements

- Oracle built-in package DBMS_PROFILER
- Oracle tables:
- PLSQL_PROFILER_RUNS
- PLSQL_PROFILER_UNITS
- PLSQL_PROFILER_DATA (user's schema)

The table below describes the sections of this chapter:

Section	Description
Setting up PL/SQL Profiler	Describes the process of setting up the PL/SQL Profiler.
PL/SQL Profiler Functionality	Describes the functionality of the PL/SQL Profiler.
Using PL/SQL Profiler	Describes how to run a profile session.

Note: The PL/SQL Profiler is an optional add-on module.

Topics

Setting up PL/SQL Profiler

The Oracle profiling tables must be on the Oracle server before you can use the PL/SQL Profiler. The first time you open the PL/SQL Profiler, the server is checked for the profiling tables. If the profiling tables are not on the server, Oracle SQL*Plus is atarted, which installs profiling tables on the Oracle server.

To run Oracle SQL*Plus (which installs the profiling tables) your Oracle server and client must meet the following conditions:

- The Oracle server and the client have the same version of Oracle.
- The client has the Oracle\BIN directory on the path.
- The client has the Oracle file, SQLPLUS.exe in the Oracle\BIN directory.
- The following Oracle files are in the Oracle\RDBMS\ADMIN directory:
- DBMSPBP.sql
- PROFLOAD.sql
- PROFTAB.sql
- PRVTPBP.blp

Note: If the Oracle server and the client machines are running different versions of Oracle, after running SQL*Plus, the following error message is displayed: Version of package is incompatible.

PL/SQL Profiler Explorer

The 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 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.
Run level	Contains all Unit, Unit Name, Unit Type, Run Date\time data for the specific Run ID.

PL/SQL Profiler Functionality

The PL/SQL Profiler offers the following functionality:

- Start
- Flush
- Run Summary
- Run Detail
- Unit Summary
- Clear Profile Table
- Unit Detail
- Stop (PL/SQL Profiler)

Start

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 profile session, do the following:

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Start.

The **PL/SQL Profiler - Start** dialog opens.

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**.

Profiling begins.

Starting an Existing Profile Session

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Start.

The **PL/SQL Profiler - Start** dialog opens.

2. Click the **Profile Label** list, and then click the existing profile.

3. Click **OK**.

Profiling begins.

Topics

- PL/SQL Profiler Functionality

Flush

The PL/SQL Profiler lets you move the data from the dynamic tables into analysis tables with the flush command.

The table below describes the options and functionality on the **PL/SQL Profiler - Flush** dialog:

Option	Description
Flush	Click to delete the data in a running profile.
Flush & Analyze	Click to open the **PL/SQL Profiler Run Detail** window. For details, see Run Detail.
Cancel Button	Click to abort the flush and continue the profiling session.

Note: You can only Flush a running Profile.

Flushing a Profile

To flush a profile, do the following:

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Flush.

The **PL/SQL Profiler - Flush** dialog opens.

Topics

- PL/SQL Profiler Functionality

Run Summary

The PL/SQL Profiler Run Summary window lets you to 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. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Run Summary.

The **PL/SQL Profiler - Run Summary** window opens.

2. In the **PL/SQL Profiler - Run Summary** window, click the **Label** list, and then click the target profile to populate the table.

Topics

- PL/SQL Profiler Functionality

Run Detail

The PL/SQL Profiler Run Detail window lets you to view the following information for each of your profiles:

- Run Number
- Run Date
- Run Time

The **Run Detail** tab lets you:

- View the information for all runs or you can view profile information based on the unit type or unit owner.
- View results in milliseconds, seconds and minutes.
- View graphical displays of the profiling data that let you can navigate to the specific unit within the summary portion of the window.
- Specify the number of top lines to display in the graphical portion of the interface.

Tip: Each graph is a working object. You can select data in a graph and the corresponding line of source displays in the lower pane of the interface.

Opening the Run Detail Window

To open the Run Detail Window, do the following:

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Run Detail.

The **PL/SQL Profiler - Run Detail** window opens.

2. 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.

Topics

- PL/SQL Profiler Functionality

Unit Summary

The PL/SQL Profiler Unit Summary window lets you to view the following information for each of your profiles:

- Run ID
- Run Date
- Run Time
- Unit Time
- Percentage of Run Time

The 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 navigate 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. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Unit Summary.

The **PL/SQL Profiler - Unit Summary** window opens.

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.

Topics

- PL/SQL Profiler Functionality

Clear Profile Table

The 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. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Clear Profile Table.

The profile table is cleared.

2. If you are sure that you want to clear out the profiler tables, click **Yes**.

Topics

- PL/SQL Profiler Functionality

Unit Detail

The PL/SQL Profiler Unit Detail window lets you to view the following information for each of your profiles:

- Average Time
- Source
- PL/SQL Script

The 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). Additionally, also displays graphs of execution statistics for the top N run. You can use the graphical displays to navigate 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 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. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Unit Detail.

Tthe **PL/SQL Profiler - Unit Detail** window opens.

2. 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** window, right-click, and then click **Advanced View** to populate the table with **Advanced View** information.

Topics

- PL/SQL Profiler Functionality

Stop (PL/SQL Profiler)

The PL/SQL Profiler Stop command pauses the data gathering operation. Stop & Analyze populates the summary tables so that you can view the Unit Detail and Run Summary windows.

The table below describes the options and functionality on the **PL/SQL Profiler - Stop** dialog:

Option	Description
Stop	Click to stop the profiling session.
Stop & Analyze	For details, see Run Detail..
Cancel	Click to continue the profiling session.

Stopping a Profiling Session

To stop a Profiling Session, do the following:

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Stop.

The **PL/SQL Profiler - Stop** dialog opens.

Topics

- PL/SQL Profiler Functionality

Using PL/SQL Profiler

The steps in this section provide a high level overview of running a profiling session, and cover the following processes:

- Starting the Session.
- Executing the Sample Script.
- Stopping and Analyzing the Session.

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.

Using the PL/SQL Profiler

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Start.

The **PL/SQL Profiler - Start** dialog opens.

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**.

Profiling begins.

4. On the Datasource Navigator, execute on 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. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Stop.

The **PL/SQL Profiler - Stop** dialog opens.

The table below describes the options and functionality on the **PL/SQL Profiler - Stop** dialog:

Option	Description
Stop	Click to stop the profiling session.
Stop & Analyze	Click to open the **PL/SQL Profiler Run Detail** window. For details, see Run Detail. 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.
Cancel	Click to continue the profiling session.

6. If using Rapid SQL, use the **Tools** menu to open any of the following PL/SQL Profiler windows. For more information, see:

- o Run Summary
- o Unit Summary
- o Unit Detail

Topics

- o Sample Profiling Session
- o PL/SQL Profiler Functionality

Sample Profiling Session

The Rapid SQL installations include two scripts for the sample profiling session:

- PROFILER_BUILD_DEMO.SQL
- PROFILER_DEMO.SQL

The PROFILER_BUILD_DEMO.SQL creates the objects that you profile in the walk through, and the PROFILER_DEMO.SQL is what you profile during the walk through.

Note: To create the objects in the PROFILER_BUILD_DEMO.SQL script, you need CREATE privileges.

The sample script demonstrates the following features of the PL/SQL Profiler:

- Unit Detail
- Run Detail
- Show Only Hit Lines
- Advanced View

The scripts are located in the \UsrScrpt subfolder of the main installation folder.

Overview

Sample Profiling Session is divided into six parts:

- Getting Started
- Starting the Session
- Executing the Sample Script
- Stopping the Session
- Re-running & Re-executing the Session
- Stopping & Analyzing

Sample Profiling Session - Getting Started

In this step of Sample Profiling Session, you create the objects that you profile in the walk through.

Overview

The Getting Started section guides you through:

- Opening PROFILER_BUILD_DEMO.SQL.
- Changing the Datasource Navigator Display.

- o Confirming the Creation of the Package.

Getting Started

1. On the **File** menu, select **Open**.

The **Open Files** dialog opens.

2. In the **Open Files** dialog, navigate to the UsrScrpt subfolder of the main installation folder, press ENTER, and then double-click **PROFILER_BUILD_DEMO.SQL** to open the script in a SQL Editor window.

The **PROFILER_BUILD_DEMO.SQL** script opens in an SQL Editor window.

3. On the SQL Editor window, click **Execute**.

Tthe script executes and create the package.

4. On the Navigator window list, click **Organize by Owner**.
5. On the Navigator window, click the node of your owner name.

Your schema objects are displayed.

6. 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, select **Open**.

The **Open Files** dialog opens.

2. In the **Open Files** dialog, type the path to the UsrScrpt directory, press ENTER, and then double-click **PROFILER_DEMO.SQL**.

The script opens in a SQL Editor window.

3. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Start.

The **PL/SQL Profiler - Start** dialog opens.

4. In the **Profile Label** list, enter **DemoProfile**.

5. Click **OK**.

The profiling session begins.

Note: If this is the first time you start the PL/SQL Profiler, a dialog box opens.

Note: Click **Yes** to have SQL*Plus create the tables. You need to start the profiling session again (see step 3 above.)

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**.

The script executes 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. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Stop.

The **PL/SQL Profiler - Stop** dialog opens.

2. Click **Stop**.

Sample Profiling Session - Re-running & Re-executing the Session

In this step of Sample Profiling Session, you run the same profile session and execute the DEMO script again.

Sample Profiling Session - Re-running & Re-executing the Session

To re-run and re-execute the session, do the following:

1. In the SQL Editor, click the **Query** tab.

2. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Start.

The **PL/SQL Profiler - Start** dialog opens again.

3. Click the down arrow on the **Profile Label** list, and then click **DemoProfile**.

4. Click **OK**.

The profiling session begins.

5. On the SQL Editor toolbar, click **Execute**.

The script executes again and the **Results** tab opens.

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 the sample profiling session, do the following:

1. If using Rapid SQL, on the **Tools** menu, select SQL Profiler > Stop.

The **PL/SQL Profiler - Stop** dialog opens again.

2. Click **Stop & Analyze**.

The PL/SQL Profiler - Run Detail window opens.

3. Click the **Run** list, and then click **Run#x**.

Note: A number is assigned 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.

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#x** 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**.

The **PL/SQL Profiler - Unit Detail** window opens, the grid is populated 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 PL/SQL Profiler shows only the lines of code that executed.

9. Right-click, and then click **Advanced View**.

The **Advanced View** window opens.

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 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_( ), YEARS_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 (Rapid SQL)

Topics

- Code Analyst DBMS Notes
- Code Analyst Requirements
- Installing Code Analyst
- Uninstalling Code Analyst
- Code Analyst Product Design
- Code Analyst Tutorial
- Using the Code Analyst

Code Analyst DBMS Notes

Code Analyst is available for:

- Microsoft SQL Server
- Oracle 7
- IBM DB2 LUW
- 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.

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 details concerning set up.

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.

IBM DB2 LUW 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 &lt;SQL statement&gt;
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.

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.

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
- Embarcadero SQL Debugger

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 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 ASE – 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.
	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.
	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

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 Installing Code Analyst. 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.

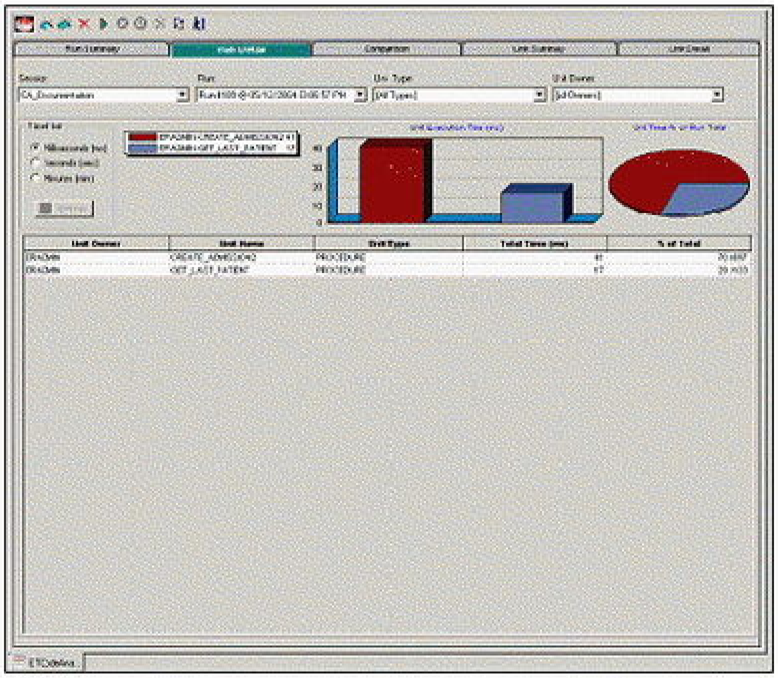
4. Click Next.

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 Code Analyst Options.



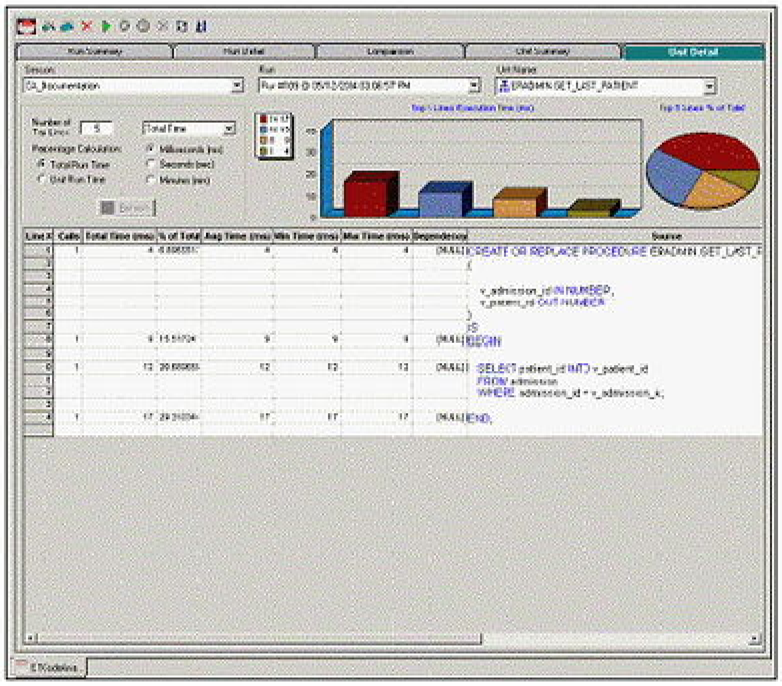
Unit Owner	Unit Name	Unit Type	Total Time (ms)	% of Total
ERADMIN	CREATE_ADMISSION2	PROCEDURE	41	70.6897
ERADMIN	GET_LAST_PATIENT	PROCEDURE	17	29.3103

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.



Line #	Calls	Total Time (ms)	% of Total	Avg Time (ms)	Min Time (ms)	Max Time (ms)	Dependency	Source
1	1	4	6.896551	4	4	4	(N/A)	CREATE OR REPLACE PROCEDURE ERADMIN.GET_LAST_P
2								(
3								
4								v_admission_id IN NUMBER,
5								v_patient_id OUT NUMBER
6								)
7								IS
8	1	9	15.51724	9	9	9	(N/A)	BEGIN
9								
10	1	12	20.68965	12	12	12	(N/A)	SELECT patient_id INTO v_patient_id
11								FROM admission
12								WHERE admission_id = v_admission_id;
13								
14	1	17	29.31034	17	17	17	(N/A)	END;

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.

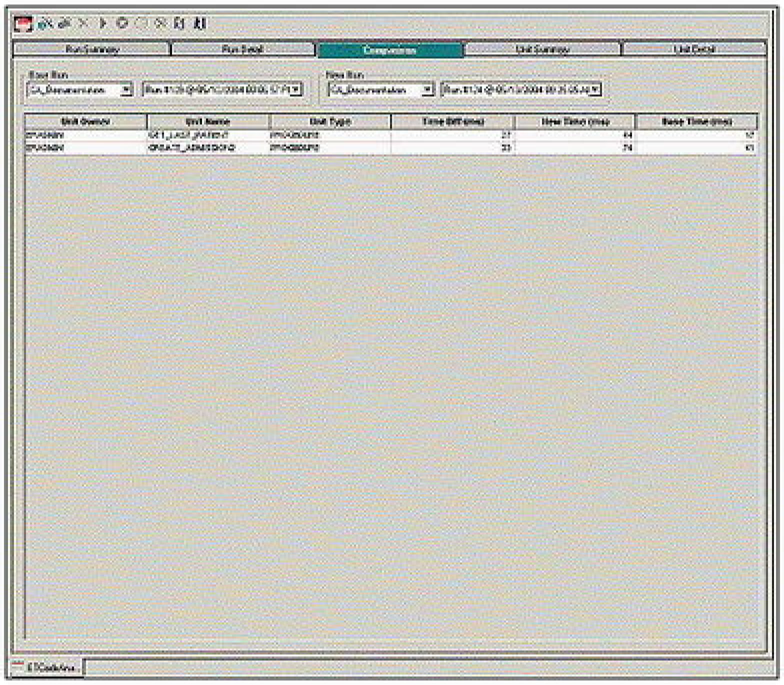
2. In Rapid SQL, open the object editor, and then modify the code on the Definition tab.

3. Click Alter.

4. In Code Analyst, on the Unit Detail Tab, click Execute.

Comparing Code Analyst Sessions

1. Click the Comparison tab.



Unit Owner	Unit Name	Unit Type	Time Diff (ms)	New Time (ms)	Base Time (ms)
[illegible]	GET_LAST_PATIENT	PROCEDURE	27	44	17
[illegible]	CREATE_ADMISSION	PROCEDURE	23	74	51

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
- 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 (Rapid SQL)

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

- 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

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.

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.

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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.

Topics

- 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.
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.

Topics

- 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.

Topics

- Execute
- Code Analyst Product Design
- Using the Code Analyst