Oracle® Developer Tools for Visual Studio

Help

23.26.0
G47987-01
April 2026

Oracle Developer Tools for Visual Studio Help, 23.26.0

G47987-01

Primary Authors: Maitreyee Chaliha, Christian Shay , Janis Greenberg, Patricia Huey

Contents

2 Setting Up

3 Integration with Server Explorer

4 Wizards, Designers, and Dialogs

5 Oracle Data Window

6 PL/SQL Code Editor

7 Oracle PL/SQL Debugger

8 SQL and PL/SQL File Editor

9 Using Select AI: Natural Language to SQL

10 Oracle Database Project

11 Oracle Database Project Version 2

15 Using Entity Framework Designer

16 Schema Compare

17 Multitenant Container Databases

18 Troubleshooting

Index

List of Tables

Preface

This help introduces Oracle Developer Tools for Visual Studio.

Audience

This document is intended for users of Oracle Developer Tools for Visual Studio.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Passwords in Code Examples

For simplicity in demonstrating this product, code examples do not perform the password management techniques that a deployed system normally uses. In a production environment, follow the Oracle Database password management guidelines, and disable any sample accounts. See *Oracle Database Security Guide* for password management guidelines and other security recommendations.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Changes in This Release for Oracle Developer Tools for Visual Studio

New Features for Oracle Developer Tools for Visual Studio Release 23.26.0

Oracle Developer Tools for Visual Studio release 23.26.0 includes the following:

- New SQL and PL/SQL editor and results window

  Query Window has been replaced with a new SQL and PL/SQL File Editor. The editor includes many advanced and modern editor features including IntelliSense, hover, brace matching, collapsible regions, keyword coloring, and more. The editor supports a subset of SQL*Plus commands and features a new results window that allows you to save results in CSV or JSON format.

- SQL and PL/SQL formatter

  The new SQL and PL/SQL Formatter can be used with the editor to consistently format SQL and PL/SQL as you choose. You can customize this formatter using the SQL and PL/SQL Formatter Settings.

- Select AI for natural language queries

Use Select AI to execute natural language queries rather than SQL. Manage your Select AI credentials and profiles via dialogs in Visual Studio. See Using Select AI: Natural Language to SQL to learn more.

- Support for Vector database column type in designers and data window
- Autonomous Database dialog updates

  Dialogs have been updated to reflected new options in creating and managing Autonomous Databases.
- Context sensitive help

  When using a dialog, press F1 or click a help icon to view this online help.

New Features for Oracle Developer Tools for Visual Studio Release 23.8

Oracle Developer Tools for Visual Studio release 23.8 includes the following:

- Real-Time SQL Monitoring

  Real-Time Monitoring provides automatic monitoring of SQL statements or PL/SQL blocks. You can view a list of monitored SQL, view and save SQL Monitor Active Reports, and generate an Active Report for ad hoc SQL in the Oracle Query Window.
- IntelliSense enhancements in Oracle Query Window and PL/SQL Code Editor including Hover and Stored Procedure and Function Parameter completion.
- HTTPS Proxy settings in Connection Dialog Box to allow connections from behind firewalls and similar scenarios.
- PL/SQL Debugging now allows for the scenario where the database must connect back to Visual Studio using an alternate IP address that is different than the IP address used by the Visual Studio host machine. This is especially useful when the database is inside of a container. See PL/SQL Debugger Options.
- Cloud Management enhancements including Configure Walletless Connectivity and Network Access Dialog which allows the user to configure walletless connections and configure the Access Control List.
- Code generation for C#, Python, Java(JDBC), and Javascript (Node.js)
- Visual Studio theming support

New Features for Oracle Developer Tools for Visual Studio Release 23.6

Oracle Developer Tools for Visual Studio release 23.6 includes the following:

- Supports Entity Framework Designer in combination with code projects that reference ODP.NET 23.x
- Compatible with new datatypes in Oracle Database 23ai (such as Boolean and Vector)
- Duality Views are now visible in Server Explorer

New Features for Oracle Developer Tools for Visual Studio Release 21.7.0.0.0

Oracle Developer Tools for Visual Studio release 21.7.0.0.0 includes the following:

- Support for Microsoft Visual Studio 2022

  Oracle Developer Tools for Visual Studio now supports Visual Studio 2022.

New Features for Oracle Developer Tools for Visual Studio Release 19.3

Oracle Developer Tools for Visual Studio release 19.3 includes the following:

- Support for browsing and managing Oracle Cloud Infrastructure database resources

  An Oracle Cloud Infrastructure Node has been added to Server Explorer making it possible to browse and manage your Oracle Autonomous Databases directly from within Visual Studio.

- Support for Microsoft Visual Studio 2019

  Oracle Developer Tools for Visual Studio now supports Visual Studio 2019.

New Features for Oracle Developer Tools for Visual Studio Release 12.2.0.1.0

Oracle Developer Tools for Visual Studio release 12.2.0.1.0 includes the following:

- Support for Microsoft Visual Studio 2017

  Oracle Developer Tools for Visual Studio now supports Visual Studio 2017.

- Oracle Database Project Version 2 project type

  The Oracle Database Project Version 2 project type is introduced in this release. It allows you to import your database schema into this project as a set of standardized SQL scripts which can be checked into source control.

  Note: The **Generate Create Script to Project** Server Explorer menu item is not supported with the Oracle Database Project Version 2 project type. Instead, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.

- Schema Compare supports comparisons against Oracle Database Project Version 2 project type

  The Schema Compare tool has also been enhanced to allow schema comparisons, *diff* script generation, and updating between Oracle Database Project Version 2 projects and live database instances or between two different Oracle Database Project Version 2 projects.

- Dependency and References Viewer

  The Dependencies and References Viewer enables you to view the dependencies and references that a database schema object has on other schema objects, both for Oracle Database connections as well as Oracle Database Project Version 2 projects.

New Features for Oracle Developer Tools for Visual Studio Release 12.1.0.2.4

Oracle Developer Tools for Visual Studio release 12.1.0.2.4 includes the following:

- Support for Microsoft Visual Studio 2015

  Oracle Developer Tools for Visual Studio now supports Visual Studio 2015.

- Oracle Database Project Version 2 project type (BETA)

  The Oracle Database Project Version 2 project type is introduced in this release. It allows you to import your database schema into this project as a set of standardized SQL scripts which can be checked into source control.

  Note: The **Generate Create Script to Project** Server Explorer menu item is not supported with the Oracle Database Project Version 2 project type. Instead, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool. Functionality involving this beta feature may change in a future release.

- Schema Compare supports comparisons against Oracle Database Project Version 2 project type (BETA)

  The Schema Compare tool has also been enhanced to allow schema comparisons, *diff* script generation, and updating between Oracle Database Project Version 2 projects and live database instances or between two different Oracle Database Project Version 2 projects. Functionality involving this beta feature may change in a future release.

- Schema Compare toolbar changes

  New Update Target toolbar button and Generate Script toolbar button have been added to Schema Compare.

  Note: The **Add to Database Project** and **Export to Editor** toolbar items have been removed.

- Schema Compare support for partial script generation or partial target update

  An Action checkbox in the Schema Compare output window allows you to generate *diff* scripts or update targets for certain schema objects or schema object types, for example all tables and views only, or the `EMPLOYEES` table only.

- New Schema Compare options

  A **Drop objects not in source** option has been added to the Schema Compare Source and Target Dialog Advanced Options.

- Schema Compare supports additional schema object types

  Schema Compare now supports materialized views, XMLType tables/views and Index Organized Tables.

- Generate Create Script supports additional schema object types

  Server Explorer menu items **Generate Create Script** and **Generate Create Script to Project** now supports materialized views, XMLType tables/views and Index Organized Tables.

- Server Explorer changes for Tables, Views, Indexes and Triggers nodes

  Triggers Node and Indexes Node are now top level nodes and contain all triggers and indexes present in the database schema. The Triggers node contains category nodes of

several types including Table Triggers Node, View Triggers Node, Schema Triggers Node, and Database Triggers Node.

The Tables Node now contains category nodes Relational Tables Node, Object Tables Node, and XML Tables Node. Similarly the Views Node has been modified to contain category nodes Relational Views Node, Object Views Node, XML Views Node, and Materialized Views Node.

New Features for Oracle Developer Tools for Visual Studio Release 12.1.0.2.0

Oracle Developer Tools for Visual Studio release 12.1.0.2.0 includes the following:

- Option to enable/disable schema name usage in script generation

  New options in the General Options Page allow you to control whether automatically generated SQL scripts include the schema name when referencing schema objects. This includes scripts generated by Generate Create Script in Server Explorer as well as those generated by Schema Compare.

  > **See Also**
  >
  > General Options Page

- Generate Create Script on Data Connection Node

  The Data Connection node in Server Explorer now includes **Generate Create Script** and **Generate Create Script to Project** menu items. These write the definitions of all objects that appear as children to the Data Connection Node to a `.sql` file or set of `.sql` files.

  > **See Also**
  >
  > Data Connection Nodes

- Option to limit number of objects displayed under a Server Explorer collection node

  A new setting in Data Connections Options Page limits the number of objects that can appear under Server Explorer collection nodes (Tables, Views, Procedures, Functions, and so on). This can improve performance and prevent hangs in the case of schemas that have a large number of objects.

  > **See Also**
  >
  > Data Connections Options Page

  > **See Also**
  >
  > Data Connections Options Page

- Schema Compare now supports XMLType Table and XMLType Views, Index Organized Tables, and Bitmap join indexes

> **See Also**
>
> Schema Compare Source and Target Dialog

New Features for Oracle Developer Tools for Visual Studio Release 12.1.0.1.2

Oracle Developer Tools for Visual Studio release 12.1.0.1.2 includes the following:

- Support for Microsoft Visual Studio 2013

  Oracle Developer Tools for Visual Studio now supports Visual Studio 2013.

New Features for Oracle Developer Tools for Visual Studio Release 12.1.0.1.0

Oracle Developer Tools for Visual Studio release 12.1.0.1.0 includes the following:

- Multitentant Container Database Administrative Features

  Multitenant container database (CDB) administrative features are now exposed through ODT. When connected through Server Explorer to an Oracle Database 12*c* Release 1 (12.1) CDB, a user with `SYSDBA` privileges sees a Pluggable Databases Node. Various menus and dialogs allow the user to create new pluggable databases (PDBs.), clone them, plug and unplug them, open and close them, and delete them. An option to add a connection to a new or cloned PDB to Server Explorer has been added to the General Options Page.

  > **See Also**
  >
  > – About Multitenant Container Databases
  > – Pluggable Databases Node
  > – Pluggable Database Nodes
  > – New Pluggable Database Dialog
  > – Clone Pluggable Database Dialog
  > – Plug Pluggable Database Dialog
  > – Unplug Pluggable Database Dialog
  > – Open Pluggable Database Dialog
  > – Close Pluggable Database Dialog
  > – General Options Page

- Schema Compare

The new Schema Compare tool allows you to compare entire schemas (or subsets of a schema) across two Server Explorer Oracle Database connections.

You can view the comparison results in a results window (see Schema Compare Results Window) and you can generate a deployment script, a *diff script*, that can upgrade a target database schema to match the source schema.

> **See Also**
>
> – About Schema Compare
> – Schema Compare Source and Target Dialog
> – Schema Compare Results Window

- Automatic Implicit `REF CURSOR` Metadata Generation

  You can now use the Stored Procedure Run Dialog Box to automatically generate implicit `REF CURSOR` metadata information and add this metadata information to your `app.config` or `web.config` file. This is useful if you wish to map a `REF CURSOR` parameter in an Oracle Stored Procedure to a Complex Entity type return value of an Entity Function using the Add Function Import Dialog.

  > **See Also**
  >
  > – Using Add Import Function Dialog to Import an Oracle Stored Procedure
  > – Stored Procedure Run Dialog Box

- Support for Oracle Data Provider for .NET, Managed Driver

  You can now connect to Oracle from Server Explorer using the new Oracle Data Provider for .NET, Managed Driver. Any designers or wizards that automatically generate .NET code using this server explorer connection will create code with appropriate references to the Managed Driver.

  (If you need to autogenerate .NET code with references to the Unmanaged Driver, connect in Server Explorer using ODP.NET, Unmanaged Driver).

  > **See Also**
  >
  > Using the Connection Dialog Box

- Connection Dialog Support for SQL*Net EZ Connect and TNSNAMES.ORA Searching

  You can use SQL*Net EZ Connect in the Connection Dialog Box , which allows you to enter a hostname, port, and service name rather than using a TNS connection alias from a `tnsnames.ora` file. You can also use a #unique_83 to assist you in locating and copying over other `tnsnames.ora` files that may exist on your computer.

See Also

- Connection Dialog Box
- #unique_83

- Script Generation to Database Project Creates Master Script with Child Scripts

  The Generate Create Script to Project menu option, when used with multiple selected Server Explorer schema objects nodes, now generates a master SQL script into the Oracle Database Project Scripts folder. This script has calls to individual child SQL scripts, one for each schema object, ordered with dependencies taken into consideration. Dragging and dropping selected schema objects into a Oracle Database Project folder results in all child scripts being created in that target folder.

  See Also

  Managing Oracle Script Files

- Oracle Database 12 Release 1 (12.1) New Data Types Support

  ODT now supports the following new Oracle Database 12.1 data types: Identity Columns (See: Table Designer), 32k extended data types (See: General Options Page), PL/SQL Boolean, Implicit `REF CURSOR`s.

  See Also

  - Table Designer
  - General Options Page
  - Generating an Entity Model from an Oracle Database

- Support for Parameterized SQL and PL/SQL in Query Window

  When you execute SQL queries or PL/SQL blocks containing parameter placeholders (for example `:VAL` or `:1`), you will be prompted to enter the values for the parameters.

  See Also

  - Using Bind Parameters with SQL Statements or PL/SQL blocks
  - Query Parameters Dialog

- Binary XML Support

  Binary XML types are now supported in ODT.

> **See Also**
>
> – Table Designer
> – XML Schema Designer

- Grant Debugging Privileges Dialog

  Oracle Database 12.1 or later now requires additional new privileges for granting explicit access to an IP address and port range when debugging PL/SQL. The Grant Debugging Privileges Dialog provides an easy way to grant these and other privileges that are required.

> **See Also**
>
> Grant Debugging Privileges Dialog

New Features for Oracle Developer Tools for Visual Studio Release 11.2.0.3.20

Oracle Developer Tools for Visual Studio release 11.2.0.3.20 includes the following:

- Support for Microsoft Visual Studio 2012

  Oracle Developer Tools for Visual Studio now supports Visual Studio 2012.

New Features for Oracle Developer Tools for Visual Studio Release 11.2.0.3.1

Oracle Developer Tools for Visual Studio release 11.2.0.3.1 includes the following:

- Support for Entity Designer in Visual Studio 2010.

  Oracle Developer Tools now supports Entity Designer, including a new DDL Generation Template, and new Database Generation Workflows.

> **See Also**
>
> About Entity Framework

New Features for Oracle Developer Tools for Visual Studio Release 11.2.0.1.2

Oracle Developer Tools for Visual Studio release 11.2.0.1.2 includes the following:

- Microsoft Visual Studio 2010 and .NET Framework 4 support

Oracle Developer Tools for Visual Studio now supports Visual Studio 2010 and .NET Framework 4.

- Support for PL/SQL Debugging on Computers using IPv6 or having Multiple IP Addresses

  PL/SQL debugging is now possible on computers that use an IPv6 address. (This requires connecting to Oracle Database version 11.2 or higher). The PL/SQL Debugging Options page and other related dialog boxes also now include the ability to select a specific IP address to use during PL/SQL debugging. If no IP address is chosen in the options page and multiple addresses exist on the computer, the developer is prompted to choose an IP address before debugging begins.

New Features for Oracle Developer Tools for Visual Studio Release 11.1.0.7.20

Oracle Developer Tools for Visual Studio release 11.1.0.7.20 includes the following:

- Oracle Performance Analyzer

  Oracle Performance Analyzer examines an application's use of Oracle Database over a specified period of time and provides tuning recommendations to improve application performance, for example, modifying SQL or adding indexes on a table. To perform this analysis, Oracle Performance Analyzer creates an Automatic Database Diagnostic Monitor (ADDM) task. Starting and ending times define the analysis time period, which is represented by 2 Automatic Workload Repository (AWR) snapshots. The results of the analysis are represented as ADDM Task nodes in Server Explorer. AWR Snapshot nodes have also been added to Server Explorer.

  See Also

  - About Oracle Performance Analyzer
  - Oracle Performance Analyzer Interface
  - ADDM Tasks Node
  - AWR Snapshots Node
  - New ADDM Task Dialog
  - New AWR Snapshot Dialog

- SQL Tuning Advisor

  Developers can tune arbitrary SQL statements with SQL Tuning Advisor. Accessed through the Tune SQL button on the toolbar in the Oracle Query Window or from Oracle Performance Analyzer recommendations, this tool provides instant recommendations explaining how to improve the performance of Oracle SQL statements.

  See Also

  - #unique_98
  - About SQL Tuning Advisor

- Advanced Queuing (AQ) Designers and Integration with Server Explorer

Advanced Queuing Designers enable administration of AQ features in Oracle. Server Explorer now displays queues and queue tables.

See Also

- About Oracle Streams Advanced Queueing
- Queue Table Designer
- Queue Designer
- Advanced Queues Node

- Selection and operation on multiple Server Explorer nodes

  You can select multiple Server Explorer nodes and execute menu operations on them. Multiple SQL scripts, for example, can be generated for multiple schema objects or for entire classes of schema objects, such as all tables. Multiple PL/SQL Stored procedures or packages can be compiled at once. Multiple schema objects can now be dragged and dropped into a Oracle Database Project as well.

  See Also

  Performing Operations on Multiple Database Objects at Once

- Collection Node Filtering

  Collection filtering enables you to display only those Oracle schema objects that you are interested in, using very flexible filter definitions. For example, the Tables collection can be filtered to show tables that begin with a certain character. Collection node filtering is enabled on the Filters tab in the connection dialog box.

  See Also

  - Filters Tab
  - Filtering Collection Nodes

- User and Role designers and Users and Roles Server Explorer nodes

  These designers allow you to create and modify users and roles, which then appear as Server Explorer nodes.

  See Also

  - About Users, Roles, and Privileges
  - User Designer
  - Users Node
  - Role Designer
  - Roles Node

- Grant/Revoke Privileges Wizard supports System Privileges and Roles

  Grant/Revoke Privileges Wizard now supports System Privileges and Roles and also allows users to grant privileges on more than one object at once, or to more than one user or role at once.

  > **See Also**
  >
  > Grant/Revoke Privileges Dialog

- PL/SQL Compiler Settings Options page

  The PL/SQL Compiler Settings Options page controls how Compile or Compile Debug operations are performed from Server Explorer or from the PL/SQL Editor.

  > **See Also**
  >
  > PL/SQL Compiler Settings Options Page

- Table designer enhancements

  The Table designer displays more information about the column data types.

  > **See Also**
  >
  > Table Designer

- Rename schema objects in Server Explorer

  You can rename Oracle database objects by clicking the object name in Server Explorer. You can rename tables, views, sequences, synonym, and indexes.

- Oracle Trigger Designer enhancements

  The Trigger Designer now supports multiple event types.

  > **See Also**
  >
  > Trigger Designer

New Features for Oracle Developer Tools for Visual Studio Release 11.1.0.6.20

Oracle Developer Tools for Visual Studio release 11.1.0.6.20 includes the following:

- Integration with Visual Studio 2005
  - Integration with Server Explorer in Visual Studio 2005

    The Oracle Developer Tools are now fully integrated with Microsoft Server Explorer.

> **Note**
>
> Oracle Explorer is no longer available in Visual Studio 2005. Its functionality has been replaced by Server Explorer.

> **See Also**
>
> About Server Explorer

- Integration with Visual Studio 2005 designers and wizards

  This release is fully integrated with Microsoft data designers including Query Designer.

> **See Also**
>
> Integration with Query Designer

- Oracle Database Project with Source Control Integration

  Oracle Database Project is a Visual Studio project type that enables you to manage SQL scripts, providing the ability to edit and run them, as well as checking them into source control.

> **See Also**
>
> About Oracle Database Project

- Run SQL*Plus Dialog

  This dialog enables you to run any SQL script even those that are not included in an Oracle Database project.

> **See Also**
>
> Run SQL*Plus Script Dialog

- Support for User-Defined Types (UDTs)

  All user-defined types defined in the database can be viewed in the Server Explorer tree controls. Designers have been added for Objects, VARRAYs, and Nested Table Types.

See Also

- User-Defined Types Node
- Object Type Nodes
- Object Type Body Nodes
- Object Type Attribute Nodes
- Object Method Nodes
- VARRAY Type Nodes
- Oracle Custom Class Wizard
- Object Type Designer
- Varray Designer
- Nested Table Type Designer

- Query Window enhancements

  Query Window enhancements support `EXPLAIN PLAN` functionality, auto-commit toggle mode, and shortcut keys.

See Also

- Enabling and Disabling Autocommit
- #unique_130

- Data Window Enhancements

  The data window now has an options page that allows the user to restrict the number of rows fetched and displayed.

See Also

General Options Page

- A new dialog for granting and revoking privileges for schema objects.

See Also

Grant/Revoke Privileges Dialog

- PL/SQL Debugging Enhancements

  The PL/SQL debugger now supports the autos windows in Visual Studio.

See Also

Autos Window

- Import Table Wizard

  This wizard makes it easy to move tables and their data from external data sources such as Microsoft SQL Server, Microsoft Access, and Excel spreadsheets.

  > **See Also**
  >
  > Import Table Wizard

New Features for Oracle Developer Tools for Visual Studio Release 10.2.0.2

Oracle Developer Tools for Visual Studio release 10.2.0.2 includes the following:

- An integrated PL/SQL debugger. This enables you to debug PL/SQL programs in the Visual Studio environment, in much the same manner as you would debug a C# or Visual Basic program.

  > **See Also**
  >
  > About the Oracle PL/SQL Debugger

- Support for Visual Studio 2005

1

Welcome

Welcome to Oracle Developer Tools for Visual Studio

Oracle Developer Tools for Visual Studio 2022 is a free Visual Studio extension that makes it easy to connect to your Oracle Database and Oracle Autonomous Database, browse and modify schema objects and data, edit and debug PL/SQL, generate SQL deployment scripts, perform schema comparisons, tune SQL and .NET app performance, and more. Use Oracle Cloud Explorer to create an Always Free Oracle Autonomous Database instance in the cloud and begin developing your database app quickly.

These tools are a required component to enable development with Entity Designer and Table Adapter Configuration Wizard.

Before you begin, read the following sections to learn about how best to get started using Oracle Developer Tools:

- Oracle Developer Tools Functionality
- Get Help, Provide Feedback, and Report Issues

Oracle Developer Tools Functionality

Oracle Developer Tools provides the following functionality:

- **Server Explorer Integration:** The Server Explorer Window is a component of Visual Studio that consists of a tree control that allows you to browse data connections and servers. Oracle Developer Tools for Visual Studio integrates with Server Explorer and allows you to browse Oracle Database and also to manage and browse your Oracle Autonomous Databases.

  To display Server Explorer, select **Server Explorer** from the Visual Studio View menu.
- **Wizards, Designers, and Dialogs** help you easily create new schema objects or alter existing schema objects. You launch the wizards and designers from the menus that display when you right-click nodes in Server Explorer.
- **Oracle Data Window** displays table and view data in a grid format, which you can view or update. To launch Data Window, double-click the node representing the table or view whose data you want to view or update.
- **About PL/SQL Code Editor** is a database-backed editor (not a file-based editor) that provides syntax coloring, automatic formatting, statement completion, method tips, error task items, and collapsible regions as you work. To start PL/SQL Code Editor, double-click the node for the PL/SQL schema object whose code you want to edit.

  For a file-based editor for Oracle SQL and PL/SQL scripts, see Oracle SQL Script Editor.

- **Oracle PL/SQL Debugger** enables you to debug PL/SQL programs in the Visual Studio environment, in much the same manner as you would debug a C# or Visual Basic .NET program.
- **Oracle Database Project Version 2** is a Visual Studio project type that enables you to manage SQL scripts, providing the ability to edit and run them, as well as checking them into source control.

Get Help, Provide Feedback, and Report Issues

Questions or feedback about this extension? Please post your questions, comments, or report issues on the Oracle Developer Tools forum.

See Also

Server Explorer | Server Explorer Nodes | Wizards and Designers | Oracle Data Window | PL/SQL Code Editor | About SQL and PL/SQL File Editor | Oracle Project | Requirements | Setting Up | Related Documentation

Requirements

This section covers the following topics:

- System Requirements

System Requirements

The requirements are as as follows:

- Oracle Database or Oracle Autonomous Database 19.*x* or later
- Microsoft Visual Studio 2022 or Visual Studio 2026

Related Documentation

In addition to the Oracle documentation provided with Oracle Developer Tools for Visual Studio, you have access to these Oracle resources:

- Oracle Developer Tools for Visual Studio Home Page
- Accessing Documentation about other Oracle Products
- Oracle .NET Developer Center

Oracle Developer Tools for Visual Studio Home Page

Visit the Oracle Developer Tools for Visual Studio home page for the latest news, demos and updates at

https://www.oracle.com/database/technologies/developer-tools/visual-studio/

Oracle .NET Developer Center

Visit the .NET Developer center to learn about all of Oracle's offerings on the .NET platform, including news, white papers, free downloads, forums, and more.

https://www.oracle.com/database/technologies/appdev/dotnet.html

Accessing Documentation about other Oracle Products

If you already have a username and password for Oracle Technology Network, then you can go directly to the documentation section of the OTN Web site at

https://docs.oracle.com/en/

2

Setting Up

About Setting Up

Setting up Oracle Developer Tools for Visual Studio primarily consists of using Server Explorer to create a Data Connection to Oracle Database or Oracle Autonomous Database.

Connecting to Oracle Database from Visual Studio

ODT connects to a database using a host name, port number, and service name as identifying information. This information can be provided to the Server Explorer Data Connections dialog box directly, or as part of a connection string, or these connection details can be referenced via a TNS alias that is located in a `tnsnames.ora` file.

For more information on creating or modifying the `tnsnames.ora` file, see *Oracle Net Services Administrator's Guide*.

After you have ascertained the hostname, port number, and service name for your database (or have obtained a tnsnames.ora file), you can start Visual Studio and connect to Oracle Developer Tools using Server Explorer.

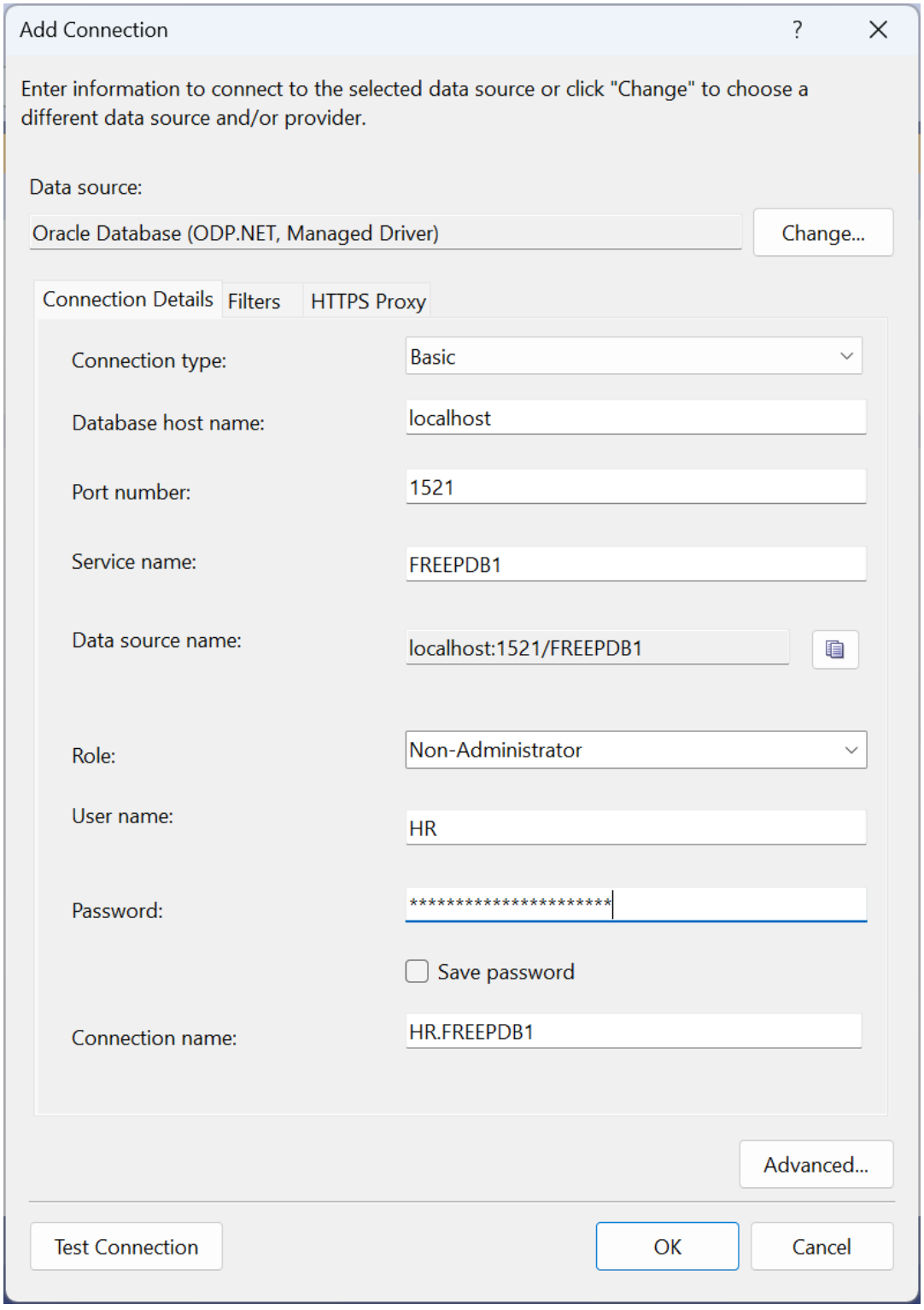
To connect to an Oracle database from Visual Studio:

1. To open Server Explorer, select it from the **View** menu.

   In Server Explorer, right-click the Data Connections node and from the menu, select **Add Connection**.

2. In the connection dialog box if the Data Source field is not set to "Oracle Database (ODP.NET, Managed Driver)" then press the Change button to open the Change Data Source dialog. In that dialog, in the Data Source list, select **Oracle Database**, and in the Data Provider dropdown, select **ODP.NET Managed Driver**. Then check **Always Use This Selection**.

   The Add Connection dialog box appears similar to the following:



If the connection dialog box does not look like this, then please check to make sure the Data Source field is set to "Oracle Database (ODP.NET, Managed Driver)".

3. In the Add Connection dialog box, enter the connection information.

See Also

Connection Dialog Box

Oracle Developer Tools Options Pages

Oracle Developer Tools includes the following options pages.

- General Options Page
- Connection Configuration Options Page
- Data Connections Options Page
- Dependency Viewer Options Page
- Entity Designer Settings Options
- IntelliSense Settings Options Page
- PL/SQL Compiler Settings Options Page
- PL/SQL Debugging Options
- Real-Time SQL Monitoring Options Page
- Query Execution Settings
- SQL and PL/SQL Formatter Settings
- Logging Settings
- Explain Plan Settings
- Schema Compare Options

3

Integration with Server Explorer

- Schema Triggers Node
- Database Triggers Node
- Trigger Nodes
- Views Node
- Relational Views Node
- Object Views Node
- XML Views Node
- Materialized Views Node
- View Nodes
- View Column Node
- Procedures Node
- Procedure Nodes
- Functions Node
- Function Nodes
- Parameter Nodes for Procedures, Functions, Package Methods, and Object Type Methods
- Function Node: Return Value Node
- Packages Node
- Package Nodes
- Package Function Nodes
- Package Procedure Nodes
- Package Body Nodes
- Synonyms Node
- Synonym Nodes
- Sequences Node
- Sequence Nodes
- XML Database Node
- XML Schemas Node
- XML Schema Nodes
- Java Classes Node
- Java Class Nodes
- User-Defined Types Node
- Users Node
- User Nodes
- Roles Node
- Role Nodes
- Object Type Nodes
- Object Type Body Nodes
- Object Type Attribute Nodes

About Server Explorer

Server Explorer is a tree control built into Visual Studio that you can use as the main interface for interacting with Oracle Database or Oracle Autonomous Database.

To view Server Explorer:

From the **View** menu in Visual Studio, select **Server Explorer**.

See the following topics:

About Server Explorer Integration

This section covers the following topics:

What is Server Explorer Integration?

The Server Explorer Window is a component of Visual Studio that consists of a tree control that allows you to browse data connections and servers. Oracle Developer Tools for Visual Studio integrates with Server Explorer and allows you to browse Oracle Database and also to manage and browse your Oracle Autonomous Databases.

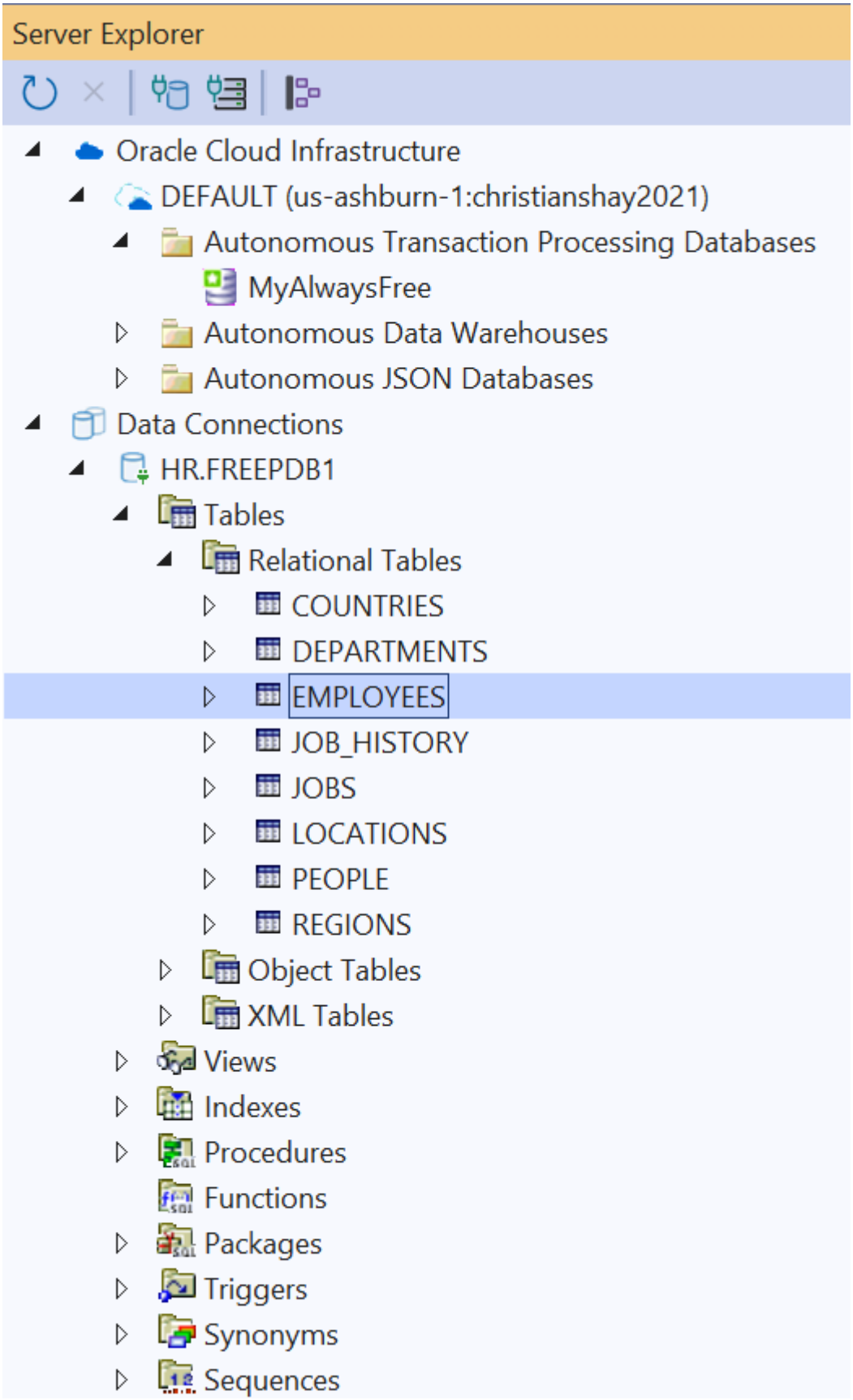
Once a connection has been established to Oracle Database under the Data Connections node, you can expand the tree control and explore your database.

The expanded tree control contains several types of nodes:

- **Connections:** User connections to a particular database.
- **Schema objects:** Objects that a schema defines, such as tables, views, triggers, or other objects that comprise a relational database.
- **Schema Object Categories:** Groupings of database objects, such as tables, functions, views, and so on.

  You can customize the hierarchies of the schema objects using the Server Explorer Nodes Option Page. See Data Connections Options Page.

To display the tree structure, double-click the main node, which is Data Connections. To expand a collection node to display its child nodes, click the carat symbol next to the node. To find out information about a node or perform actions on it, right-click the node to display a menu.

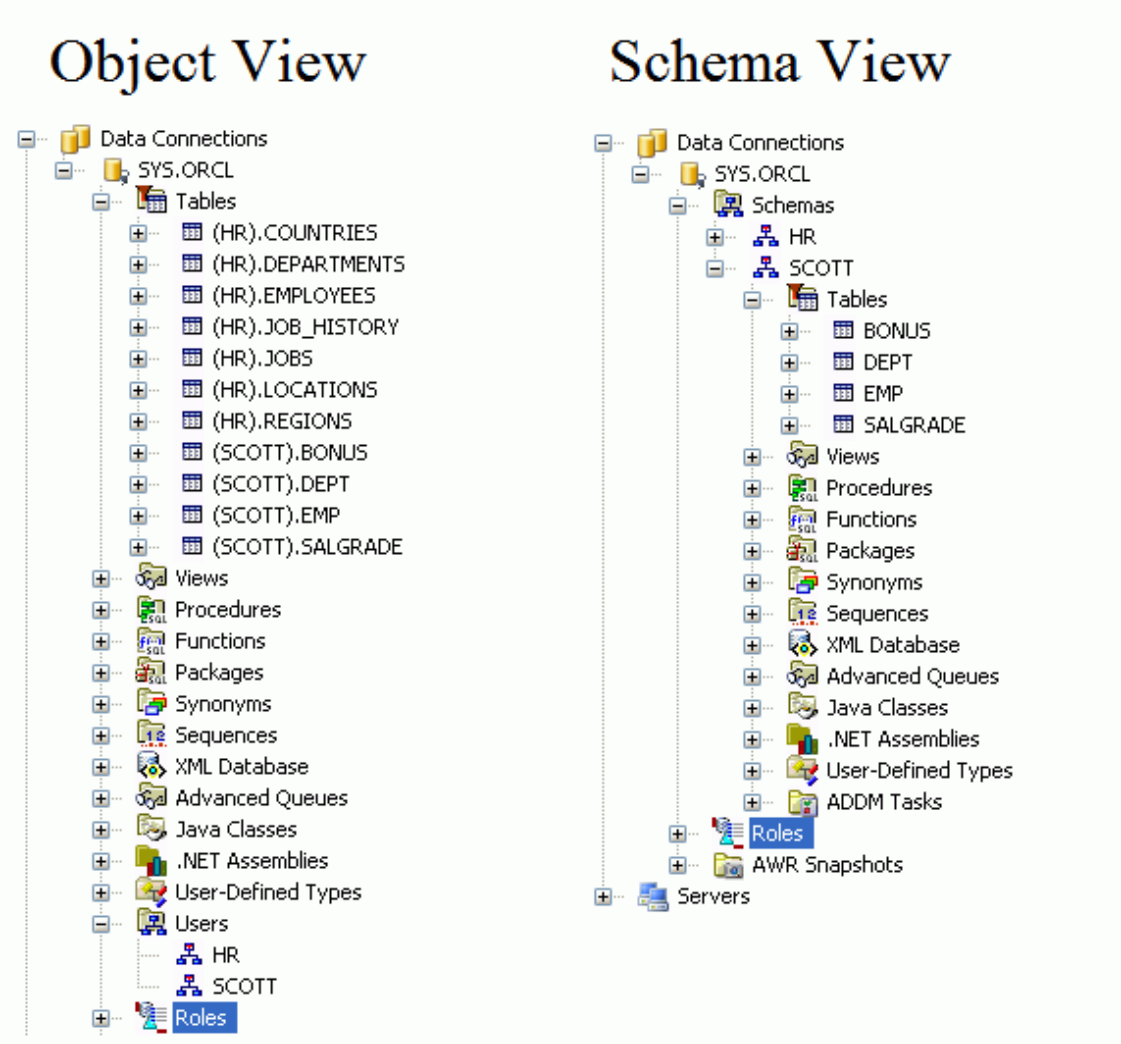


Object View and Schema View

Some users can see schema objects that belong to multiple schemas. For these users Server Explorer offers two different modes for viewing multiple schemas. There is an object view which appends the name of the schema to the front of each schema object. For example, if a table `EMPLOYEE` were accessible to user `SMITH` from the schema `HR`, its name would be `(HR).EMPLOYEE`.

The schema view adds a schema node to segregate the schemas. You can switch between these two views by selecting the connection and choosing **Change View**, then **Objects** or **Schemas**.

The following graphic shows views for a `SYSDBA` user. The object view is on the left, showing all the Table, Views, Procedures, and other objects. The schema view is on the right, and the various schema nodes are visible.



> **Note**
>
> For non-SYSDBA users, the Server Explorer, connection filters must be modified to show additional schemas (by default only the owner's schema is visible). To modify connection filters see Filters Tab.

Refreshing Nodes

All node objects provide a **Refresh** command on their menu. The **Refresh** command refreshes the node and its children in Server Explorer.

Performing Operations on Multiple Database Objects at Once

You can select multiple database objects and collection nodes at once by holding down the shift or control key, and then selecting them. After selecting the nodes, you can right-click to display a menu of operations that can be performed on every selected node.

Only operations supported by all the selected nodes appear in the menu.

Operations that can be Performed on Multiple Selected Objects

For the Server Explorer nodes in which they appear, the following menu options can be performed on multiple selected objects. The descriptions indicate how multiple selection works with this option and supplements the menu option descriptions included with the specific Server Explorer nodes.

Menu Option	Description
Refresh	Refreshes selected nodes.
Properties	Displays in the property grid those properties that are common.
Copy	Copies all selected nodes, but only performs paste operation on types that are supported in the specific drop target.
Delete	Deletes each of the selected nodes in the order in which they were selected, in a batch mode.
Compile	Performs the following actions: 1. If the procedure, function, trigger, or object is open in the PL/SQL Editor and has been changed, saves the changes to the database. 2. Compiles the procedure, function, trigger, or object.
Compile Debug	Compiles the procedure, function, trigger, or object with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information.
Generate Create Script	Writes the Server Explorer script definitions to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the Server Explorer script definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.

Avoiding Performance Problems Due to Large Collections of Database Object Nodes

Some collection nodes may contain thousands of child nodes. For example, the Synonyms collection node typically has many public synonyms. This can lead to performance problems if all of the nodes are retrieved and displayed at once. The following features help with these performance issues:

- Filtering Collection Nodes
- Filtering the Displayed Schema
- Filtering Public Synonyms
- Changing the Visibility of Collection Types
- Limiting the Number of Objects Displayed in a Collection Node

Filtering Collection Nodes

Collection node filters give you the ability to control exactly which child nodes appear under a particular collection node. For example, a filter can be created that shows only Table nodes beginning with the letter A, under the Tables collection node.

To enable a filter, right-click on a collection node such as Tables node and choose Filters. This launches the Filters Tab on the Connection Dialog where you can construct filters of varying levels of complexity.

When filtering is enabled, the collection node icon changes to reflect this. The property page for the collection node contains information about the filters.

Filtering the Displayed Schema

Connections can be filtered so that they only show particular schemas by changing the Displayed Schemas property in the Filters Tab on the Connection Dialog.

Filtering Public Synonyms

Databases typically have hundreds of public synonyms. By default, these are not displayed in Server Explorer. To enable or disable public synonyms, modify the Display Public Synonyms property in the Filters Tab on the Connection Dialog.

Limiting the Number of Objects Displayed in a Collection Node

The Data Connections Options Page allows you to limit the number of objects that can appear under Server Explorer collection nodes (Tables, Views, Procedures, Functions, and so on). If the number of objects exceeds the chosen maximum value, a message box appears to alert you that not all objects were displayed. This can improve performance and prevent hangs in the case of schemas that have a large number of objects.

Changing the Visibility of Collection Types

Collection types may be removed entirely from Server Explorer. For example, the Views node could be removed. This can be achieved on individual connections through the Visible Collections property in the Filters Tab on the Connection Dialog. The default visibility settings for any newly created connections can be set through the Data Connections Options Page. Note that these default visibility settings in the Data Connection Options Page do not affect already created connections, even after refreshing.

Oracle Cloud Infrastructure Node

This section covers the following topics:

- About the Oracle Cloud Infrastructure Node
- How the Oracle Cloud Infrastructure Node Works
- Menu Options

About the Oracle Cloud Infrastructure Node

This node is a top level node of Server Explorer and contains individual Oracle Cloud Infrastructure Profile Nodes, allowing users to explore and manage cloud resources such as Oracle Autonomous Databases.

How the Oracle Cloud Infrastructure Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
Sign up for Oracle Cloud	Launches a web browser to the sign up page for Oracle Cloud

Menu Option	Description
Configure Oracle Cloud Connection	Opens a web page containing instructions showing how to create an Oracle cloud account and how to connect to it.
Refresh	Refreshes the node.
Properties	Displays properties for the node.

Oracle Cloud Infrastructure Profile Nodes

This section covers the following topics:

- About the Oracle Cloud Infrastructure Profile Node
- How the Oracle Cloud Infrastructure Accounts Profile Works
- Menu Options

About the Oracle Cloud Infrastructure Profile Node

This node represents the DEFAULT profile which is defined inside of your Oracle Cloud Infrastructure config file. Under this node, you will find a Transaction Processing/Lakehouse/JSON Node. Users can use these child nodes to explore and manage cloud database resources.

> **Note**
>
> If this node is replaced with the message "[No OCI account profile found]" this means that the config file is not in the correct location (or is not named config with no file extension).

For more information on how to set up the OCI config file, click the **Configure Oracle Cloud Connection** menu item on the Oracle Cloud Infrastructure Node.

How the Oracle Cloud Infrastructure Accounts Profile Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
Change Compartment or Region	Opens the Change Compartment or Region Dialog to change the current compartment
Refresh	Refreshes the node.
Properties	Displays properties for the node.

See Also

Oracle Cloud Infrastructure Profile Nodes | Oracle Cloud Infrastructure Node

Transaction Processing/Lakehouse/JSON Node

This section covers the following topics:

About the Transaction Processing/Lakehouse/JSON Node

The Transaction Processing Node, Lakehouse Node, JSON Node contain collections of Transaction Processing/Lakehouse/JSON Nodes .

How the Transaction Processing/Lakehouse/JSON Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
Create New	Opens the Create Autonomous AI Database Dialog
Refresh	Refreshes the node.
Properties	Displays properties for the node.

See Also

Oracle Cloud Infrastructure Node

Transaction Processing/Lakehouse/JSON Nodes

This section covers the following topics:

About the Transaction Processing/Lakehouse/JSON Nodes

The Transaction Processing Node, Lakehouse Node, and JSON Node represent an Autonomous Transaction Processing Database, an Autonomous Data Warehouse, and an Autonomous JSON Database respectively.

How the Transaction Processing/Lakehouse/JSON Nodes Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
Create Data Connection	Assists in creating a Server Explorer Data Connection so that the database schema can be explored and manipulated. Opens the Create Data Connection Dialog which downloads credentials files if needed and then opens the Server Explorer. See Using the Connection Dialog Box.
Start	Starts the database, after a confirmation dialog. The node icon will change colors to reflect the changing database Lifecycle State.
Stop	Stops the database, after a confirmation dialog. The node icon will change colors to reflect the changing database Lifecycle State.
Terminate	Terminates the database, after a confirmation dialog. The node icon will change colors to reflect the changing database Lifecycle State.
Restore	Opens the Restore Database Dialog to restore the database from a point in time.
Scale Up/Down	Opens the Scale Up/Down Dialog to enable changes to CPU core count and storage allocation.
Update Credentials	Opens the Change Administrator Password Dialog to allow the password to be changed.
Upgrade Instance to Paid (Always Free instances only)	After a confirmation dialog, converts a Always Free database instance into a paid instance
Download Credentials Files	Opens the Download Credentials Files Dialog to assist in downloading the credentials files to a directory.
Update License Type	Opens the Update License Type Dialog [link to new page] to allow the license type to be changed.
Refresh	Refreshes the node.
Properties	Displays properties for the node.

See Also

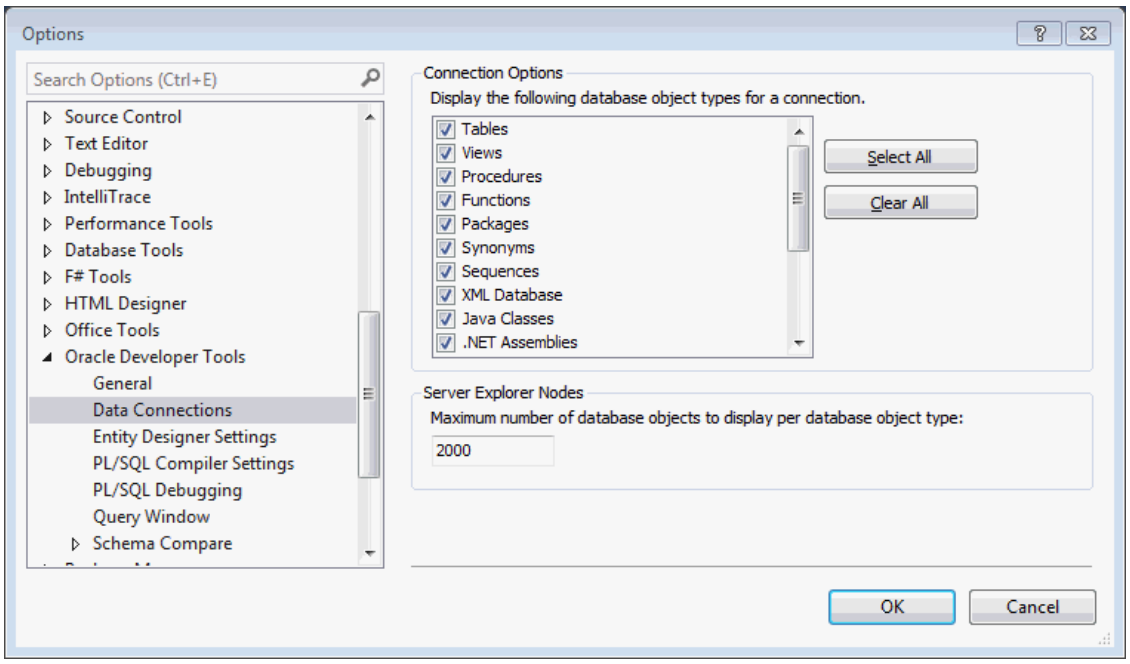
Oracle Cloud Infrastructure Profile Nodes | Oracle Cloud Infrastructure Node

Data Connections Options Page

The Data Connections Options page allows you to customize Server Explorer by choosing which Oracle database schema object types will be displayed for an ODT connection. This may be helpful in improving performance of Server Explorer or in cases where you are getting permissions related errors for certain types of objects.

Changes made to these displayed database object types will not affect any already existing connections even after doing a refresh. To change the displayed object types for already created connections, use the Filters Tab on the Connection Dialog instead.

This page also allows you to limit the number of objects that can appear under Server Explorer collection nodes (Tables, Views, Procedures, Functions, and so on). If the number of objects exceeds the chosen maximum value, a message box appears to alert you that not all objects were displayed. This can improve performance and prevent hangs in the case of schemas that have a large number of objects



Select **OK** to save and apply the changes and **Cancel** to drop the changes.

Using the Data Connections Options Settings

Control	Description
Display the following object types for a connection	Each object type that is selected will appear in the Server Explorer. The others do not. Changes made to the displayed database objects will not affect any already existing connections even after doing a refresh. To change the displayed objects for already created connections, use the Filters Tab on the Connection Dialog instead.
Select All	Selects every object type.
Clear All	Deselects every object type.
Maximum number of database objects to display per database object type	This value limits the number of objects that can appear under Server Explorer collection nodes (Tables, Views, Procedures, Functions, and so on). If the number of objects exceeds the chosen maximum value, a message box appears to alert you that not all objects were displayed. This can improve performance and prevent hangs in the case of schemas that have a large number of objects.

Data Connections Node

This section covers the following topics:

- How the Data Connections Node Works
- Menu Options

How the Data Connections Node Works

The Data Connections node is a top level node in Server Explorer and provides the connections to explore databases. The Data Connections node contains one or more child Data Connection nodes that represent user connections. You can add connections from different data sources and different providers.

To perform actions on this node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

To understand how to establish a connection, see Connection Dialog Box and Connecting to Oracle Database from Visual Studio .

Menu Options

Menu Option	Description
Refresh	Updates the Data Connections tree, including all user connections associated with this database connection.
Delete	Disabled for this node.
Add Connection	Displays the Add Connection dialog box so that you can create a new connection. If the Data Source field does not show Oracle Database (ODP.NET, Managed Driver) or (ODP.NET, Unmanaged Driver), then you must select **Change** and select Data Source, Oracle Database and set to Managed or Unmanaged Driver.
Create New SQL Server Database	Not applicable for Oracle Developer Tools for Visual Studio.
Properties	Disabled for this node

See Also

Data Connection Nodes | Connection Dialog Box | Filters Tab on the Connection Dialog

Data Connection Nodes

This section covers the following topics:

- How the Data Connection Node Works
- Menu Options

How the Data Connection Node Works

An Oracle Developer Tools Data Connection node is a child of the Data Connections node. It represents a user connection to an Oracle database using Oracle Data Provider for .NET. The name of a Data Connection node corresponds to the connection name the user specifies in the Add Connection screen.

Each Data Connection node has a set of nodes under it representing the available database schema object types, such as tables or views.

For example, a Data Connection node named `SMITH.ORCL` connects a user named `SMITH` to the `ORCL` database. This connection node might have a Tables schema collection node under it that contains child nodes for the following:

- Tables `EMP`, `DEPT`, and `BONUS` that are owned by Smith.
- Tables that are owned by another user, such as `HR`, but are visible to `SMITH`.

Closed connections appear with an *x* in the node icon.

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Viewing Database Objects

For Oracle Developer Tools Connections in Server Explorer, you can view the contents of a database (tables, views, procedures, and other objects) in two ways:

- Objects: Displays a hierarchy of objects, grouped by the database schema object types
- Schemas: Displays a hierarchy of schemas, then object types.

You can select one of these views using the **Change View** menu on the Data Connection nodes.

Menu Options

Menu Option	Description
Refresh	Updates the current connection and all nodes displayed under the connection.
Delete	Deletes the connection and its saved connection information.
Change View	Toggles view between Objects and Schemas. This option is only available if the connection is already open.
Modify Connection	Displays the Connection dialog box so that you can edit the existing connection.
Close Connection	Closes the connection.
Open New SQL File	Opens a new SQL file in the SQL and PL/SQL File Editor and associates that file with this database connection.
Open Existing SQL File	Opens a existing SQL file in the SQL and PL/SQL File Editor and associates that file with this database connection.
Open Connection	Opens the connection. This option is available only after the connection is opened for the first time.
Filters	Displays the Filters Tab on Connection Dialog. This option is only available if the connection is already open.
Privileges	Opens the Grant/Revoke Privileges dialog box. This option is only available if the connection is already open.
Generate Create Script	Writes the definition of all objects that appear as children to this Data Connection Node to a `.sql` file, if the child object supports SQL Script generation. See the Visual Studio Output Window for a list of objects for which SQL was not generated.

Menu Option	Description
Generate Create Script to Project	Generates the definitions of all objects that appear as children to this Data Connection Node if the child object supports SQL Script generation. The definitions are written to multiple `.sql` files, one for each schema object. It adds these `.sql` files to the appropriate folders in an open Oracle Database Project. See the Visual Studio Output Window for a list of objects for which SQL was not generated. A master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to the individual child SQL scripts. The master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. See "Managing Oracle Script Files" for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Import Table	Launches the Import Table Wizard. This option is only available if the connection is already open. See Import Table Wizard.
SQL and PL/SQL Formatter Settings	Opens the SQL and PL/SQL Formatter Settings to customize the way the formatter will work when used on this connection.
Explain Plan Settings	Opens the Explain Plan Settings to customize Explain Plan when used on this connection.
Oracle Performance Analyzer	Launches Oracle Performance Analyzer. See Oracle Performance Analyzer.
Oracle Schema Compare	Launches the Schema Compare Source and Target Dialog . See About Schema Compare
Real-Time SQL Monitoring	Opens the Real-Time SQL Monitor Window to display monitored SQL and PL/SQL. See About Real-Time SQL Monitor
Select AI->Set Default AI Profile for Connection	Opens Set Default AI Profile For Connection Dialog where you can select the AI profile to be used in the SQL and PL/SQL editor for natural language queries. For more information see Using Select AI: Natural Language to SQL
Select AI->Configure Select AI Provider Network Access	Opens the Configure Select AI Provider Network Access Dialog for one time configuration of the AI provider. For more information see Using Select AI: Natural Language to SQL
Select AI->Configure Select AI Profile	Opens the Configure Select AI Profile Dialog and allows you to create new credentials and profiles. For more information see Using Select AI: Natural Language to SQL
New Query	Opens a new query in the Microsoft Query Designer. See Integration with Query Designer
Rename	Renames the connection.
Properties	Displays the Properties window.

See Also

Data Connections Node| Connection Dialog Box | Query Window | Filters Tab on the Connection Dialog

Pluggable Databases Node

This section covers the following topics:

About Pluggable Databases

The Pluggable Databases node is visible when a user with the SYSDBA privilege connects to a multitenant container database (CDB). When expanded, a set of child Pluggable Database Nodes are visible, one for each available pluggable database (PDB) in the CDB.

For more information about multitenant architecture features in ODT, see About Multitenant Container Databases.

> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

How the Pluggable Databases Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
New Pluggable Database	Create a new Pluggable Database using the New Pluggable Database Dialog.
Plug	Plugs in a Pluggable Database using the Plug Pluggable Database Dialog.
Filter	Opens the Filters Tab on the Connection Dialog to control which child nodes appear under the collection.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

Pluggable Database Nodes | Clone Pluggable Database Dialog | Close Pluggable Database Dialog | New Pluggable Database Dialog | Open Pluggable Database Dialog | Plug Pluggable Database Dialog | Unplug Pluggable Database Dialog

Pluggable Database Nodes

This section covers the following topics:

About Pluggable Database Nodes

A pluggable database node represents a multitenant pluggable database (PDB). For more information about multitenant architecture features in ODT see About Multitenant Container Databases.

How the Pluggable Database Nodes Work

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
Clone	Clones a pluggable database using the Clone Pluggable Database Dialog (This menu option is not available for the seed database. Use the New Pluggable Database Dialog menu option on the Pluggable Databases node instead.)
Open	Opens the Pluggable Database using the New Pluggable Database Dialog. This option is not available if the Pluggable Database is already open.
Close	Closes the Pluggable Database using the Close Pluggable Database Dialog. This option is not available if the Pluggable Database is already closed.
Unplug	Unplugs the Pluggable Database using the Unplug Pluggable Database Dialog.
Delete	Permanently deletes the pluggable database, including all database files on the server. To remove the pluggable database from the container database without deleting it, use Unplug instead.
Refresh	Refreshes the node..
Properties	Shows properties for the node.

See Also

Pluggable Databases Node | Clone Pluggable Database Dialog | Close Pluggable Database Dialog | New Pluggable Database Dialog | Open Pluggable Database Dialog | Plug Pluggable Database Dialog | Unplug Pluggable Database Dialog

Schemas Node

This section covers the following topics:

- About Schemas
- How the Schemas Node Works
- Menu Options

About Schemas

The Schemas node lists every schema selected in the Filters Tab on the Connection Dialog.

A schema is a collection of database objects, including logical structures such as tables, views, sequences, procedures, synonyms, indexes, clusters, and database links. A schema has the same name as the user who controls it.

You cannot hide the Schemas node through the Data Connections Tools/Option Page of Oracle Developer Tools.

How the Schemas Node Works

The Schemas node is visible from the Server Explorer tree when you right-click the connection node and select **Change View**, then **Schemas**. Otherwise, you see the Users node.

To perform actions on the Schemas node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New User	Launches the designer to create a new user.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the privileges dialog.
Refresh	Refreshes the schemas node.
Properties	Displays the properties window.

See Also

Schema Nodes | User Designer

Schema Nodes

This section covers the following topics:

- How the Schema Nodes Work
- Menu Options

How the Schema Nodes Work

A Schema node is a child of the Schemas Node. Each node represents a database schema that you selected in the Filters Tab on the Connection Dialog. Schema nodes appear if you expand the Schemas node.

If you expand a Schema node, such as `SYSTEM`, the database schema object types nodes such as Tables or Views, appear as child nodes under the `SYSTEM` schema node.

To perform actions on a Schema node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Opens the designer so that you can edit the node.
Generate Create Script	Writes the schema definition to a `.sql` file.
Generate Create Script to Project	Generates the schema definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Grant Debugging Privileges	Opens the Grant Debugging Privileges Dialog which grants to the user privileges neccessary for PL/SQL debugging on this computer. This menu option is only visible to users with SYSDBA privileges.
Privileges	Opens the privileges dialog.
Delete	Deletes the schema selected. Disabled for the current (connected) schema.
Refresh	Refreshes schema properties fetched from database.
Properties	Displays the properties window.

See Also

Schemas Nodes | User Designer

Tables Node

This section covers the following topics:

- About Tables Node
- How the Tables Node Works

- Menu Options

About Tables Node

A table is the basic unit of data storage in an Oracle database. Tables store data in rows and columns. Tables with Invalid Status are not displayed.

Oracle Database supports the following types of tables:

- Relational
- Object
- XML

How the Tables Node Works

The Tables node contains the Relational Tables Node, Object Tables Node, and XML Tables Node category nodes. Each of these category nodes can contain one or more child Table Nodes that represent relational tables, object relational tables, or XML tables respectively.

To perform actions on the Tables node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Relational Table	Creates a new table with relational and XMLType columns in the Table Designer.
New Object Table	Creates a new Object table in the Table Designer.
New XML Table	Creates a new XML table in the Table Designer.
Import Table	Launches the Import Table Wizard. See Import Table Wizard.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Generate Create Script	Writes all table definitions to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the table definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Refresh	Updates the Tables node tree, including its associated table nodes.
Properties	Displays the Properties window.

See Also

Table Nodes | Table Column Node | Table Designer | Oracle Data Window

Relational Tables Node

This section covers the following topics:

- About Relational Tables node
- How the Relational Tables node Works
- Menu Options

About Relational Tables node

This category node is a child node of Tables Node. It contains a set of child Table Nodes which represent all relational tables visible to the user.

How the Relational Tables node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
New Relational Table	Creates a new table with relational and XMLType columns in the Table Designer.
Import Table	Launches the Import Table Wizard. See Import Table Wizard.

Menu Option	Description
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Generate Create Script	Writes all table definitions to a `.sql` file.
Generate Create Script to Project	Generates the table definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Refresh	Updates the Tables node tree, including its associated table nodes.
Properties	Displays the Properties window.

See Also

Table Nodes | Table Column Node | Table Designer | Oracle Data Window

Object Tables Node

This section covers the following topics:

- About Object Tables Node
- How the Object Tables node Works
- Menu Options

About Object Tables Node

This category node is a child node of Tables Node. It contains a set of child Table Nodes which represent all object tables visible to the user.

How the Object Tables node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
New Object Table	Creates a new Object table in the Table Designer.
Import Table	Launches the Import Table Wizard. See Import Table Wizard.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Generate Create Script	Writes all table definitions to a `.sql` file.
Generate Create Script to Project	Generates the table definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Refresh	Updates the Tables node tree, including its associated table nodes.
Properties	Displays the Properties window.

See Also

Table Nodes | Table Column Node | Table Designer | Oracle Data Window

XML Tables Node

This section covers the following topics:

- About XML Tables Node
- How the XML Tables node Works
- Menu Options

About XML Tables Node

This category node is a child node of Tables Node. It contains a set of child TTable Nodes which represent all XML tables visible to the user.

How the XML Tables node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
New XML Table	Creates a new XML table in the Table Designer.
Import Table	Launches the Import Table Wizard. See Import Table Wizard.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Generate Create Script	Writes all table definitions to a `.sql` file.
Generate Create Script to Project	Generates the table definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Refresh	Updates the Tables node tree, including its associated table nodes.
Properties	Displays the Properties window.

See Also

Table Nodes | Table Column Node | Table Designer | Oracle Data Window

Table Nodes

This section covers the following topics:

- How Table Nodes Work
- Menu Options

How Table Nodes Work

The Table node is a child of the Tables Node. It represents any of the following table types:

- Relational: Its child nodes are:
  - Table Column nodes for each column in the table
  - Constraints node for each constraint in the table
  - Indexes node for each index on the table
  - Triggers node for each trigger on the table
- XML: Its child nodes are:

- Constraints node for each constraint in the table
- Indexes node for each index on the table
- Triggers node for each trigger on the table

Notes:

- The Table nodes do not include index-organized tables.
- Tables with Invalid Status are not displayed.

The type of database connection determines the ownership of a table. For more information on the ownership of tables and other schema objects, see Data Connections Node and Schemas Node.

After you have created a table, you can begin to populate it by using Oracle Data Window to edit and view its data.

Starting with Visual Studio 2005, you can edit the name of the table in place by selecting the name and then clicking it to open an edit box or by choosing the Rename menu option to open the edit box.

Note: This feature only works in Visual Studio 2005 or higher.

To perform actions on a Table node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu. To view the table's data, double-click the Table node.

Menu Options

Menu Option	Description
Design	Opens the table in the Table Designer. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the Designer will not open.
Retrieve Data	Opens the table in the Data Window. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and no data window will open.
Add Trigger	Adds a new trigger to the table using the Table Designer. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the Create Trigger dialog box will not open.
Generate Create Script	Writes the table definition to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the table definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Generate Code	Generates example code that executes a SELECT against this table. From the submenu, select **C#**, **Java**, **Python** or **Javascript**. A new code window will open with the example code in the language selected. Generating code additional times for the same language type will place the additional code in the already open code window at the point of the cursor. If you do not want that, then close the open code window before regenerating code.
Copy	Copies the selected table to the clipboard for pasting into Visual Studio designers. This provides the same functionality as dragging and dropping the table to the Visual Studio designers.
Delete	Drops the table. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the node is deleted. If the table cannot be dropped due to dependencies, a dialog box appears with two options. Click **Yes** to delete the table regardless of dependencies. Click **No** to skip the table deletion.
Rename	Allows you to edit the Table name in place in Server Explorer. Alternatively, you can select the Table name and then click one more time to open an edit box.
Refresh	Updates the Table node. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Tables Node | Table Column Node | Table Designer | Oracle Data Window | XML Schema Designer

Table Column Node

This section covers the following topics:

- How the Table Column Nodes Work
- Menu Options

How the Table Column Nodes Work

A Table Column node represents a single column in the enclosing relational table.

An Object type table has no columns. When you expand an Object type table node, the object (user-defined type) used by the table is displayed under the table node.

An XML table always has only one column, `SYS_NC_ROWINFO$`, which is the only table column node that appears under the XML table node.

When you expand an user-defined type column node, the user-defined type appears under the column node.

To perform actions on the Table Column node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Design Table	Edits the table column using the Table Designer.
Copy	Copies the selected column to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping the column to the Visual Studio designers.
Refresh	Updates the Table Column node tree, including all its associated table nodes.
Properties	Displays the Properties window.

See Also

Tables Node | Table Nodes | Table Column Node | Table Designer | Oracle Data Window | XML Schema Designer

Constraints Node

This section covers the following topics:

- About Constraints
- How the Constraints Node Works
- Menu Options

About Constraints

Constraints are rules that you create to restrict values in a database. You can define one or more conditions in the constraint that a user must satisfy before they make changes to the

table to which you have applied the constraint. They are part of the table definition and operate automatically.

How the Constraints Node Works

The Constraints node, which is a child of the Tables node, contains one or more child Constraint nodes for a table. To perform actions on the Constraints node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
Design Table	Edits the table using the Table Designer.
Refresh	Updates the Constraints node, including its associated constraints nodes.
Properties	Displays the Properties window.

See Also

Tables Node | Constraint Nodes | Table Designer

Constraint Nodes

This section covers the following topics:

- How Constraint Nodes Work
- Menu Options

How Constraint Nodes Work

The Constraint node is a child of the Constraints node. It represents a constraint on a table. To perform actions on the child Constraint node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Design Table	Edits the table using the Table Designer. This command is not supported for object tables.
Refresh	Updates the constraints node.
Properties	Displays the Properties window.

See Also

Constraints Node | Tables Node | Table Designer

Indexes Node

This section covers the following topics:

About Indexes

An index is a schema object that contains an entry for each value that appears in the indexed column(s) of the table or cluster. It provides direct, fast access to rows. Oracle Database supports several types of index:

- Normal indexes. (By default, Oracle Database creates B-tree indexes.)
- Bitmap indexes, which store rowids associated with a key value as a bitmap.
- Partitioned indexes, which consist of partitions containing an entry for each value that appears in the indexed column(s) of the table.
- Function-based indexes, which are based on expressions. They enable you to construct SQL statements that evaluate the value returned by an expression, which in turn may include built-in or user-defined functions.
- Domain indexes, which are instances of an application-specific index of type indextype.

How the Indexes Node Works

There are two types of Indexes Node: One is a child of Data Connection Node and contains a set of all indexes in the database that are visible to the user. The other is a child of a Table Node and contains only those indexes that apply to the particular table. Each contains one or more child Index Nodes that represent indexes for a table schema object in the Oracle database.

To perform actions on the Indexes node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

See Also

Data Connection Node | Table Node

Menu Options (child of Data Connection Node)

Menu Option	Description
New Index	Creates a new Index using the Index Designer.

Menu Option	Description
Generate Create Script	Writes the Index definitions to a `.sql` file. If the object associated with the Indexes node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the Index definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Refresh	Updates the Indexes node tree, including its associated Index nodes.
Properties	Displays the Properties window.

See Also

Table Designer

Menu Options (child of Table Node)

Menu Option	Description
Design Table	Creates a new index for the table using the Table Designer. This command is not supported for object tables.
Refresh	Updates the Indexes node tree, including its associated index nodes.
Properties	Displays the Properties window.

See Also

Table Designer

Index Nodes

This section covers the following topics:

- How the Index Nodes Work
- Menu Options

How the Index Nodes Work

The Index node is a child of the Indexes node. It represents an associated index for a table.

Starting with Visual Studio 2005, you can edit the name of the index in place by selecting the name and then clicking it to open an edit box or by choosing the Rename menu option to open the edit box.

Note: This feature only works in Visual Studio 2005 or higher.

To perform actions on an Index node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Design Table	Edits the index using the Table Designer. This command is not supported for object tables.
Rename	Allows you to edit the Index name in place in Server Explorer. Alternatively, you can select the Index name and then click one more time to open an edit box.
Refresh	Updates the index node.
Properties	Displays the Properties window.

See Also

Indexes Node | Table Designer

Triggers Node

This section covers the following topics:

- About Triggers
- How the Triggers Node Works
- Menu Options (child of Data Connection Node)
- Menu Options (child of Table Node)

About Triggers

A trigger is a stored database procedure that is automatically invoked whenever a table or view is modified, for example by `INSERT`, `UPDATE`, or `DELETE` operations. In PL/SQL, triggers can either of the following:

- A stored PL/SQL block associated with a table, a schema, or the database or
- An anonymous PL/SQL block or a call to a procedure implemented in PL/SQL or Java

Oracle Database automatically executes a trigger when specified conditions occur.

How the Triggers Node Works

There are two types of Triggers Nodes: One is a child of Data Connection Node and contains a set of all triggers in the database that are visible to the user. The other is a child of a Table Node and contains only those triggers that apply to the particular table. The Triggers Node that is a child of Data Connection Node contains the Table Triggers Node, View Triggers Node, Schema Triggers Node, and Database Triggers Node category nodes. Each of these nodes can contain one or more child Trigger Nodes.

To perform actions on the Triggers node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

See Also

Data Connection Node | Table Node

Menu Options (child of Data Connection Node)

Menu Option	Description
Add Trigger	Creates a new trigger with the Trigger Designer. This command is not supported for object tables or object views.
Generate Create Script	Writes the trigger definitions to a `.sql` file. If the object associated with the triggers node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the trigger definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Triggers node tree, including its associated trigger nodes.
Properties	Displays the Properties window.

See Also

Add Trigger Dialog Box

Menu Options (child of Table Node)

Menu Option	Description
Add Trigger	Creates a new trigger with the Trigger Designer. This command is not supported for object tables or object views.
Compile	Compiles all triggers.
Compile Debug	Compiles all triggers with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information.
Copy	Copies all triggers.
Refresh	Updates the Triggers node.
Properties	Displays the Properties window.

See Also

Add Trigger Dialog Box

Table Triggers Node

This section covers the following topics:

- About Table Triggers Node
- How the Table Triggers Node Works
- Menu Options

About Table Triggers Node

This category node is a child node of Triggers Node . It contains a set of child Trigger Nodes which represent all table triggers visible to the user.

How the Table Triggers Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Add Trigger	Creates a new trigger with the Trigger Designer. This command is not supported for object tables or object views.
Generate Create Script	Writes the trigger definitions to a `.sql` file. If the object associated with the triggers node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.

Menu Option	Description
Generate Create Script to Project	Generates the trigger definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Triggers node tree, including its associated trigger nodes.
Properties	Displays the Properties window.

See Also

Triggers Node | Add Trigger Dialog Box

View Triggers Node

This section covers the following topics:

- About View Triggers Node
- How the View Triggers Node Works
- Menu Options

About View Triggers Node

This category node is a child node of Triggers Node . It contains a set of child Trigger Nodes which represent all view triggers visible to the user.

How the View Triggers Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Add Trigger	Creates a new trigger with the Trigger Designer. This command is not supported for object tables or object views.

Menu Option	Description
Generate Create Script	Writes the trigger definitions to a `.sql` file. If the object associated with the triggers node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the trigger definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Triggers node tree, including its associated trigger nodes.
Properties	Displays the Properties window.

See Also

Triggers Node | Add Trigger Dialog Box

Schema Triggers Node

This section covers the following topics:

- About Schema Triggers Node
- How the Schema Triggers Node Works
- Menu Options

About Schema Triggers Node

This category node is a child node of Triggers Node . It contains a set of child Trigger Nodes which represent all schema triggers visible to the user.

How the Schema Triggers Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Add Trigger	Creates a new trigger with the Trigger Designer. This command is not supported for object tables or object views.
Generate Create Script	Writes the trigger definitions to a `.sql` file. If the object associated with the triggers node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the trigger definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Triggers node tree, including its associated trigger nodes.
Properties	Displays the Properties window.

See Also

Triggers Node | Add Trigger Dialog Box

Database Triggers Node

This section covers the following topics:

- About Database Triggers Node
- How the Database Triggers Node Works
- Menu Options

About Database Triggers Node

This category node is a child node of Triggers Node . It contains a set of child Trigger Nodes which represent all database triggers visible to the user.

How the Database Triggers Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Add Trigger	Creates a new trigger with the Trigger Designer. This command is not supported for object tables or object views.
Generate Create Script	Writes the trigger definitions to a `.sql` file. If the object associated with the triggers node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the trigger definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Triggers node tree, including its associated trigger nodes.
Properties	Displays the Properties window.

See Also

Triggers Node | Add Trigger Dialog Box

Trigger Nodes

This section covers the following topics:

- How the Trigger Nodes Work
- Menu Options

How the Trigger Nodes Work

The Trigger node is a child of the Triggers node. It represents an associated trigger on a table or view. Double-clicking a Trigger node displays the trigger specification and body in the PL/SQL Code Editor for editing.

To perform actions on a Trigger node, right-click on the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Edits the trigger in the PL/SQL Code Editor. Alternatively, you can double-click the Trigger node to display its specification and body in the PL/SQL Code Editor.
Compile	Compiles the trigger.
Compile Debug	Compiles the trigger with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information.
Generate Create Script	Writes the trigger definition to a `.sql` file. If the object associated with the trigger node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the trigger definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Copy	Copies the trigger.
Delete	Drops the trigger.
Refresh	Updates the Trigger node.
Properties	Displays the Properties window.

See Also

Triggers Node | PL/SQL Code Editor | Add Trigger Dialog Box

Views Node

This section covers the following topics:

- About Views
- How the Views Node Works
- Menu Options

About Views

A view is a custom-tailored presentation of the data in one or more tables. You can think of a view as a *stored query*. Views do not actually contain or store data; they derive their data from the tables on which they are based. Like tables, you can query, update, insert into, and delete from views, with some restrictions. All operations performed on a view affect its base tables.

How the Views Node Works

The Views node contains the Relational Views Node, Object Views Node, XML Views Node, and Materialized Views Node category nodes. Each of these category nodes can contain one or more child View Nodes that represent relational views, object relational views, XML views, or materialized views respectively.

To perform actions on the Tables node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Relational View	Creates a new view with relational and/or `XMLType` columns using the View Designer.
New Object View	Creates a new Object view using the View Designer.
New XML View	Creates a new view of `XMLType` using the View Designer.
Generate Create Script	Writes the view definitions to a `.sql` file. If the object associated with the views node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.

Menu Option	Description
Generate Create Script to Project	Generates the view definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies views to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping views to the Visual Studio designers.
Refresh	Updates the Views node tree, including its associated View nodes.
Properties	Displays the Properties window.

See Also

View Nodes | View Column Node | View Designer

Relational Views Node

This section covers the following topics:

- About Relational Views Node
- How the Relational Views Node Works
- Menu Options

About Relational Views Node

This category node is a child node of Views Node. It contains a set of child View Nodes which represent all relational views visible to the user.

How the Relational Views Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
New Relational View	Creates a new view with relational and/or `XMLType` columns using the View Designer.
Generate Create Script	Writes the view definitions to a `.sql` file. If the object associated with the views node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the view definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies views to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping views to the Visual Studio designers.
Refresh	Updates the Views node tree, including its associated View nodes.
Properties	Displays the Properties window.

See Also

View Nodes | View Column Node | View Designer

Object Views Node

This section covers the following topics:

- About Object Views Node
- How the Object Views Node Works
- Menu Options

About Object Views Node

This category node is a child node of Views Node. It contains a set of child View Nodes which represent all object views visible to the user.

How the Object Views Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
New Object View	Creates a new Object view using the View Designer.
Generate Create Script	Writes the view definitions to a `.sql` file. If the object associated with the views node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the view definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies views to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping views to the Visual Studio designers.
Refresh	Updates the Views node tree, including its associated View nodes.
Properties	Displays the Properties window.

See Also

View Nodes | View Column Node | View Designer

XML Views Node

This section covers the following topics:

About XML Views Node

This category node is a child node of Views Node. It contains a set of child View Nodes which represent all XML views visible to the user.

How the XML Views Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click Properties in the node's menu.

Menu Options

Menu Option	Description
New XML View	Creates a new view of `XMLType` using the View Designer.
Generate Create Script	Writes the view definitions to a `.sql` file. If the object associated with the views node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the view definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies views to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping views to the Visual Studio designers.
Refresh	Updates the Views node tree, including its associated View nodes.
Properties	Displays the Properties window.

See Also

View Nodes | View Column Node | View Designer

Materialized Views Node

This section covers the following topics:

- About Materialized Views Node
- How the Materialized Views Node Works
- Menu Options

About Materialized Views Node

This category node is a child node of Views Node. It contains a set of child View Nodes which represent all materialized views visible to the user.

How the Materialized Views Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Generate Create Script	Writes the view definitions to a `.sql` file. If the object associated with the views node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the view definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies views to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping views to the Visual Studio designers.
Refresh	Updates the Views node tree, including its associated View nodes.

Menu Option	Description
Properties	Displays the Properties window.

See Also

View Nodes | View Column Node | View Designer

View Nodes

This section covers the following topics:

- How the View Nodes Work
- Menu Options

How the View Nodes Work

The View node is a child of the Views node. It represents one or more views associated with this table (or set of tables), and can represent any of the following view types:

- Relational: Its child nodes are:
  - View Column nodes, which represent each column in the view
  - Triggers node, which represents triggers on this view
- XML: Its child nodes are:
  - Triggers node, which represents triggers on this view
  - Object Views node
- Object: Its child nodes are:
  - Triggers node, which represents triggers on this view
  -

Note: The View nodes do not include index-organized or materialized views.

Starting with Visual Studio 2005, you can edit the name of the view in place by selecting the name and then clicking it to open an edit box or by choosing the Rename menu option to open the edit box.

Note: This feature only works in Visual Studio 2005.

To perform actions on the View nodes, right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu. To view the view's data, double-click the View node.

Menu Options

Menu Option	Description
Design	Opens the view in the View Designer. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the Designer will not open. This command is not supported for object views.
Retrieve Data	Opens the view in the Data Window. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and no data window will open. This command is not supported for object views.
Add Trigger	Adds a new trigger to the view using the Table Designer. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the Create Trigger dialog box will not open. This command is not supported for object views.
Generate Create Script	Writes the table definition to a `.sql` file. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated. This command is not supported for object views.
Generate Create Script to Project	Generates the table definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Generate Code	Generates example code that executes a SELECT against this view. From the submenu, select **C#**, **Java**, **Python** or **Javascript**. A new code window will open with the example code in the language selected. Generating code additional times for the same language type will place the additional code in the already open code window at the point of the cursor. If you do not want that, then close the open code window before regenerating code.

Menu Option	Description
Copy	Copies the selected view to the clipboard for pasting into Visual Studio designers. This provides the same functionality as dragging and dropping the view to the Visual Studio designers.
Delete	Drops the view. If the object associated with the table node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the node is deleted. If the table cannot be dropped due to dependencies, a dialog box appears with two options. Click **Yes** to delete the table regardless of dependencies. Click **No** to skip the table deletion.
Rename	Allows you to edit the View name in place in Server Explorer. Alternatively, you can select the View name and then click one more time to open an edit box.
Refresh	Updates the View node. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Tables Node | Table Column Node | Table Designer | Oracle Data Window | XML Schema Designer

View Column Node

This section covers the following topics:

- How the View Column Node Works
- Menu Options

How the View Column Node Works

An Object type view has no columns. When you expand an Object type view node, the object (user-defined type) used by the view appears under the view node.An XML view always has only one column `SYS_NC_ROWINFO$`, which is the only view column node that appears under the XML view node.When you expand an user-defined type column node, the user-defined type used by the column node appears under the column node.

To perform actions on the View Column node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select Properties from the menu.

Menu Options

Menu Option	Description
Design	Edits the column using the View Designer.

Menu Option	Description
Copy	Copies the selected column to the clipboard for pasting to Visual Studio designers. This provides the same functionality dragging and dropping the column to the Visual Studio designers
Refresh	Updates the View Column node.
Properties	Displays the Properties window.

See Also

Views Node | View Nodes | View Designer | XML Schema Designer

Procedures Node

This section covers the following topics:

- About Procedures
- How the Procedures Node Works
- Menu Options

About Procedures

A procedure is a named set of PL/SQL statements designed to perform an action. Unlike functions, procedures do not return values, but they can accept values as input.

How the Procedures Node Works

The Procedures node contains one or more child Procedure nodes. To perform actions on the Procedures node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New PL/SQL Procedure	Creates a new PL/SQL procedure using the Procedure Designer.
New .NET Procedure	Deploys the procedure to an Oracle database as a .NET Assembly using the Oracle Deployment Wizard for .NET. Only enabled for connections with SYSDBA role.
Compile	Performs the following actions: 1. If the procedure is open in the PL/SQL Editor and has been changed, saves the changes to the database. 2. Compiles the procedure. See PL/SQL Compiler Settings Options Page for compile settings.

Menu Option	Description
Compile Debug	Compiles the procedure with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information. See PL/SQL Compiler Settings Options Page for compile settings.
Generate Create Script	Writes the procedure definitions to a `.sql` file. If the object associated with the procedures node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the procedure definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies the selected column to the clipboard for pasting to Visual Studio designers. This provides the same functionality dragging and dropping the column to the Visual Studio designers
Refresh	Updates the Procedures node tree, including its associated procedure and parameter nodes.
Properties	Displays the Properties window.

See Also

Procedure Nodes | Parameter Nodes for Procedures and Functions | Procedure Designer

Procedure Nodes

This section covers the following topics:

- How the Procedure Nodes Work
- Menu Options

How the Procedure Nodes Work

The Procedure node is a child of the Procedures node. It represents a procedure and has a set of child parameter nodes. To perform actions on this node: right-click the node and from the

menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

A procedure with an error is indicated by a red *x* in the procedure icon.

To fix the error, open the procedure code in the PL/SQL editor, make the required changes and save the changes to the database. If the procedure has no errors, the *x* is removed from the procedure node icon.

Procedures that have been compiled with debug information have *DBG* embedded in the icon.

Menu Options

Menu Option	Description
Edit	Displays the procedure code in the PL/SQL Code Editor. Alternatively, you can double-click the Procedure node to display its code in the PL/SQL Code Editor. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.If the body of the procedure is WRAPPED, a warning message will be displayed before the PL/SQL editor window is opened, with the wrapped text. When you make changes to the wrapped text and try to save the changes, you will be prompted to overwrite the wrapped text with your changes.
Compile	Performs the following actions: 1. If the procedure is open in the PL/SQL Editor and has been changed, saves the changes to the database. 2. Compiles the procedure. See PL/SQL Compiler Settings Options Page for compile settings.
Compile Debug	Compiles the procedure with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information. See PL/SQL Compiler Settings Options Page for compile settings.
Run	Executes the procedure using the Run dialog box.
Run Debug	Starts debugging the procedure and runs to the breakpoint. See Debugging a PL/SQL Program Directly.
Step Into	Begins debugging the procedure and stops at the line of code. See Debugging a PL/SQL Program Directly.
Generate Create Script	Writes the procedure definition to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the procedure definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies the selected procedure to the clipboard for pasting to Visual Studio designers. This provides the same functionality dragging and dropping the procedure to the Visual Studio designers.
Delete	Drops the procedure.
Refresh	Updates the Procedure node and its associated parameter nodes. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Procedures Node | Parameter Nodes for Procedures and Functions | PL/SQL Code Editor | Run Dialog Box |

Functions Node

This section covers the following topics:

- About Functions
- How the Functions Node Works
- Menu Options

About Functions

A function is a named set of PL/SQL statements designed to accept and return data values; the returned value is based on the values a user inputs. Functions are either built-in or user-defined PL/SQL expressions. The functions represented by the Function node are user-defined expressions, for example:

```
circle_area(radius)
payroll.tax_rate(empno)
```

```
hr.employees.comm_pct(dependents, empno)@remote
DBMS_LOB.getlength(column_name)
my_function(DISTINCT a_column)
```

If you prefer to write a function that does not return any values, consider writing a procedure.

How the Functions Node Works

A Function node is a child of the Functions node. Each of these child Function nodes has a set of child parameter nodes. To perform actions on this node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

The Functions node contains one or more child Function nodes to represent a function schema object in the Oracle database. Each of the Function Nodes has a set of parameter nodes and a return value node.

To perform actions on the Functions node, right-click on the node and in the menu, choose the appropriate command. To view the node's properties, select **Properties** from the menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New PL/SQL Function	Creates a new PL/SQL function using the Function Designer.
New .NET Function	Deploys the function to an Oracle database as a .NET Assembly using the Oracle Deployment Wizard for .NET. Only visible users with SYSDBA privilege.
Generate Create Script	Writes the function definitions to a `.sql` file. If the object associated with the functions node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the function definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.

Menu Option	Description
Refresh	Updates the Functions node tree, including its associated function and parameter nodes.
Properties	Displays the Properties window.

See Also

Function Nodes | Parameter Nodes for Procedures and Functions | Function Designer

Function Nodes

This section covers the following topics:

- How Function Nodes Work
- Menu Options

How Function Nodes Work

A Function node is a child of the Functions node. Each of these child Function nodes has a set of child parameter nodes.

In addition to PL/SQL functions, Java function wrappers are included.

Double-clicking on a function node opens the PL/SQL editor and displays the PL/SQL code.

A function with an error is indicated by a red *x* in the function icon.

To fix the error, open the function code in the PL/SQL editor, make the required changes and save the changes to the database. If the function has no errors, the *x* is removed from the function node.

Functions that have been compiled with debug information have *DBG* embedded in the icon.

To perform actions on this node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Displays the function code in the PL/SQL Code Editor. Alternatively, you can double-click the Function node to display its code in the PL/SQL Code Editor. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.If the body of the function is WRAPPED, a warning message will be displayed before the PL/SQL editor window is opened, with the wrapped text. When you make changes to the wrapped text and try to save the changes, you will be prompted to overwrite the wrapped text with your changes.

Menu Option	Description
Compile	Performs the following actions: 1. If the function is open in the PL/SQL Editor and has been changed, saves the changes to the database. 2. Compiles the function. See PL/SQL Compiler Settings Options Page for compile settings.
Compile Debug	Compiles the function with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information. See PL/SQL Compiler Settings Options Page for compile settings.
Run	Executes the function using the Run dialog box.
Run Debug	Starts debugging the function and runs to the breakpoint. See Debugging a PL/SQL Program Directly.
Step Into	Begins debugging the function and stops at the line of code. See Debugging a PL/SQL Program Directly.
Generate Create Script	Writes the function definition to a `.sql` file.
Generate Create Script to Project	Generates the function definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Copy	Copies the selected function to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping the procedure onto the Visual Studio designers
Delete	Deletes the function.
Refresh	Updates the Function node and its associated parameter nodes. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Function Nodes | Parameter Nodes for Procedures and Functions | Editing PL/SQL Code | Run Dialog Box |

Parameter Nodes for Procedures, Functions, Package Methods, and Object Type Methods

This section covers the following topics:

- How the Parameter Nodes Work
- Menu Options

How the Parameter Nodes Work

Each Procedure node, Function node, package method, or object method has one or more child parameter nodes, each of which represents a parameter in the procedure, function, or method.

For object methods, UDT type parameters of UDT methods are not expandable and do not have children.

When you expand a parameter node of user-defined type, the user-defined type appears under the parameter node.

To perform actions on this node: right-click on the node and in the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Parameter node.
Properties	Displays the Properties window.

See Also

Procedure Nodes | Procedure Nodes | Function Nodes | Functions Node

Function Node: Return Value Node

This section covers the following topics:

- How Function Return Value Nodes Work
- Menu Options

How Function Return Value Nodes Work

Each Function node has a return value node that represents the return value of that function.

When you expand a return value node of user-defined type, the user-defined type appears under the return value node.

To perform actions on the parameter node: right-click on the node and in the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Refresh	Updates the Function Parameter node.
Properties	Displays the Properties window.

See Also

Function Nodes | Functions Node

Packages Node

This section covers the following topics:

- About Packages
- How the Packages Node Works
- Menu Options

About Packages

A package is an encapsulated collection of related procedures, functions, and other object types grouped together into one programming unit in the database. As such, packages allow you to run a series of related tasks, with each task represented by a procedure or function.

PL/SQL packages have two parts:

- Package specification: The interface to your application. It declares the procedures, functions, types, variables, constants, exceptions, cursors, and subprograms available for use. These are the public declarations that are visible in your application. The package specification is also known as the package header.
- Package body: Defines the implementation details and private declarations that are hidden from your application. You can debug, enhance, or replace a package body without changing the specification. In addition, you can change a package body without recompiling calling programs because the implementation details in the body are hidden from your application. However, depending on the package definition, sometimes the body is not necessary. Another advantage of separating the package body from the package specification is that it lets you keep your proprietary code secure.

For more information on how PL/SQL packages work, including the packages supplied with Oracle Database, see *PL/SQL Packages and Types Reference*.

How the Packages Node Works

The Packages node contains one or more child Package nodes associated with this schema. Within each of these child package nodes are nodes representing procedures and functions.

To perform actions on the Packages node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Package	Creates a new package using the Package Designer.
Generate Create Script	Writes the package definitions to a `.sql` file. If the object associated with the packages node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the package definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Refresh	Updates the Package node tree, including its associated child nodes.
Properties	Displays the Properties window.

See Also

Package Nodes | Package Specification Functions Node | Package Specification Procedures Node | Package Body Node | Package Designer | PL/SQL Code Editor

Package Nodes

This section covers the following topics:

- How the Package Nodes Work
- Menu Options

How the Package Nodes Work

A Package node is a child of the Packages node. It represents a package schema object in the Oracle database. Each package node contains nodes for the following components within a package:

- Functions: Represents any functions that you have added to the package, for example, after clicking **Add** to add methods in the Package Designer. Within each package function node, there can be child nodes representing parameters used for that function. The package function node also contains a child node representing the return value for that function.
- Procedures: Represents any procedures that you have added to the package, similar to package functions. These too can have child nodes that represent the parameters used for that procedure.
- Package body: Appears if you have specified a package body for the package.

A package with an error is indicated by a red *x* in the package icon.

A package that has been compiled with debug information have *DBG* embedded in the icon

To fix the error, open the package specification and/or package body code in the PL/SQL editor, make the required changes, and save the changes to the database. If the package has no errors, the red *x* disappears from the package icon.

To perform actions on the Package node: right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Package Specification	Opens the package specification in the PL/SQL Code Editor.
Edit Package Body	Opens the package body in the PL/SQL Code Editor
Compile	Performs the following actions: 1. If the package is open in the PL/SQL Editor and has been changed, saves the changes to the database. 2. If the package body is open in the PL/SQL Editor and has been changed, saves the changes to the database. 3. Compiles the package specification and package body. Oracle Developer Tools automatically compiles a package and a package body when you save it. You only need to compile packages when you edit them. See PL/SQL Compiler Settings Options Page for compile settings.
Compile Debug	Compiles the package with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information. See PL/SQL Compiler Settings Options Page for compile settings.
Generate Create Script	Writes the package definition to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the package definition and adds it to the open Oracle Database Project. Generates the SQL script file definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Delete	Drops the package specification and package body.
Refresh	Updates the Package node. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Packages Node | Package Specification Functions Node | Package Specification Procedures Node | Package Body Node | Package Designer | PL/SQL Code Editor

Package Function Nodes

This section covers the following topics:

- How the Package Function Nodes Works
- Menu Options

How the Package Function Nodes Works

A Package Function node is a child of the Packages node. It represents a function in a PL/SQL package and can have one or more child Parameter nodes, which represent parameters within that function. It also has a child Return Value node representing the return value for the function.

Double-clicking on a package function node opens the PL/SQL editor with the package specification and takes you to the line of code that contains the function definition.

A package function with an error is indicated by a red *x* in the function icon.

To fix the error, open the package specification or package body code in the PL/SQL editor, make the required changes, and save the changes to the database. If the package function and the package have no errors, the *x* is removed from the package node and all the child nodes of the package node.

Functions that have been compiled with debug information have *DBG* embedded in the icon.

To perform actions on the Package Function node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Specification	Opens the package specification in the PL/SQL Code Editor. If the window for the package specification already exists, Edit Specification makes it the current window.
Run	Executes the package function using the Run dialog box.
Run Debug	Starts debugging the package function and runs to the breakpoint. See Debugging a PL/SQL Program Directly.
Step Into	Begins debugging the package function and stops at the line of code. See Debugging a PL/SQL Program Directly.
Copy	Copies the selected package function to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping the package function to the Visual Studio designers.
Refresh	Updates the package function node tree, including its associated child nodes.
Properties	Displays the Properties window.

See Also

Packages Node | Package Nodes | Package Procedures Node | Package Body Node | Package Designer | PL/SQL Code Editor |

Package Procedure Nodes

This section covers the following topics:

- How the Package Procedure Nodes Works
- Menu Options

How the Package Procedure Nodes Works

A Package Procedure node is a child of the Package node. It represents a procedure in PL/SQL package and can have one or more child Parameter nodes, which represent parameters within that procedure.

Double-clicking on a package procedure node opens the PL/SQL editor with the package specification and takes you to the line of code that contains the procedure definition.

A package procedure with an error is indicated by a red *x* in the package procedure icon.

To fix the error, open the package procedure in the PL/SQL editor, make the required changes, and save the changes to the database. If the package procedure and the package have no errors, the *x* is removed from the package node and all the child nodes of the package node.

Procedures that have been compiled with debug information have *DBG* embedded in the icon.

To perform actions on the Package Procedure node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Specification	Opens the package specification in the PL/SQL Code Editor. If the window for the package specification already exists, Edit Specification makes it the current window.
Run	Executes the package procedure using the Run dialog box.
Run Debug	Starts debugging the package procedure and runs to the breakpoint. See Debugging a PL/SQL Program Directly.
Step Into	Begins debugging the package procedure and stops at the line of code. See Debugging a PL/SQL Program Directly.
Copy	Copies the selected package procedure to the clipboard for pasting to Visual Studio designers. This provides the same functionality as dragging and dropping the package procedure to the Visual Studio designers.
Refresh	Updates the package procedure node tree, including its associated child nodes.
Properties	Displays the Properties window.

See Also

Packages Node | Package Nodes | Package Functions Node | Package Body Node | Package Designer | PL/SQL Code Editor |

Package Body Nodes

This section covers the following topics:

- About the Package Body
- How the Package Body Node Works
- Menu Options

About the Package Body

The package body contains the implementation of every cursor and subprogram declared in the package specification. Subprograms defined in a package body are accessible outside the package only if their specifications appear in the package specification. If a subprogram specification is not included in the package specification, that subprogram can only be called by other subprograms in the same package.

How the Package Body Node Works

The Package Body node is one of three child nodes of the Packages node, with the two representing package specification functions and procedures.

A package body node with an error is indicated by a red *x* in the node icon.

To fix the error, open the package body code in the PL/SQL editor, make the required changes and execute the Compile command on the node. If the package body has no errors, the *x* is removed from the node.

Package body nodes that have been compiled with debug information have *DBG* embedded in the icon.

Double-clicking on the package body node opens the PL/SQL editor and loads the package body PL/SQL code.

To perform actions on the Package Body node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Package Body	Displays if the package has a package body. It opens the package body in the PL/SQL Code Editor. Only the package body is present in the code window containing editable text starting with `PACKAGE BODY` package name. If the window already exists, Edit Package Body places the cursor at the start of the window. This menu item will not appear if the package has no package body. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer. If the body of the package is `WRAPPED`, a warning message will be displayed before the PL/SQL editor window is opened, with the wrapped text. When you make changes to the wrapped text and try to save the changes, you will be prompted to overwrite the wrapped text with your changes.
Compile	Performs the following actions: 1. If the package body is open in the PL/SQL Editor and has been changed, saves the changes to the database. 2. Compiles the package body. See PL/SQL Compiler Settings Options Page for compile settings.
Compile Debug	Compiles the package body with debug information. This is required before PL/SQL code can be debugged. See Compiling a PL/SQL Program with Debug Information. See PL/SQL Compiler Settings Options Page for compile settings.
Generate Create Script	Writes package body definition to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the package body definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Delete	Drops the package body from the database.
Refresh	Updates the Package Body node. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Packages Node | Package Nodes | Package Functions Node | Package Procedures Node | Package Designer | PL/SQL Code Editor

Parameter Nodes for Procedures, Functions, Package Methods, and Object Type Methods

This section covers the following topics:

- How the Parameter Nodes Work
- Menu Options

How the Parameter Nodes Work

Each Procedure node or Function node has one or more child parameter nodes, each of which represents a parameter in the procedure or function.

When you expand a parameter node of user-defined type, the user-defined type appears under the parameter node.

To perform actions on this node: right-click on the node and in the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Refresh	Updates the Parameter node.
Properties	Displays the Properties window.

See Also

Procedure Nodes | Procedures Node | Function Nodes | Functions Node

Synonyms Node

This section covers the following topics:

- About Synonyms
- How the Synonyms Node Works
- Menu Options

About Synonyms

A synonym is an alternative name for a table, view, sequence, procedure, function, package, materialized view, Java class schema object, user-defined object type, or another synonym.

Synonyms provide both data independence and location transparency. They permit applications to function without modification regardless of which user owns the table or view and regardless of which database holds the table or view. However, synonyms are not a substitute for privileges on database objects. Appropriate privileges must be granted to a user before the user can use the synonym.

How the Synonyms Node Works

The Synonyms node contains one or more child Synonym nodes associated with this schema.

To perform actions on the Synonyms node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Synonym	Creates a new synonym using the Synonym Designer.
Generate Create Script	Writes the synonym definitions to a `.sql` file. If the object associated with the synonyms node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.

Menu Option	Description
Generate Create Script to Project	Generates the synonym definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Refresh	Updates the Synonyms node tree, including its associated synonym nodes.
Properties	Displays the Properties window.

See Also

Synonym Nodes | Synonym Designer | Avoiding Performance Problems Due to Large Collections of Database Object Nodes

Synonym Nodes

This section covers the following topics:

- How Synonym Nodes Work
- Menu Options

How Synonym Nodes Work

The Synonym node is a child of the Synonyms node. It represents a synonym object in the database. Each Synonym node has a read-only child node that represents the target of the synonym for this schema. For example, suppose the `EMP_SYN` synonym points to the `EMP` table; this synonym will have an `EMP` Table node as its child. However, if Server Explorer does not support the schema object to which this synonym points, the synonym will not have a child node. The synonym itself is read-only.

If the synonym points to an object over a database link, the child synonym node will be a special database link node (`DBlink`). This node has the standard properties and menu items. The node is named using the following syntax:

schema.object`@dblink_name`

Starting with Visual Studio 2005, you can edit the name of the synonym in place by selecting the name and then clicking it to open an edit box or by choosing the Rename menu option to open the edit box.

Note: This feature only works in Visual Studio 2005.

To perform actions on the Synonym node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Edits an existing synonym using the Synonym Designer. Alternatively, double-clicking a synonym node opens the Synonym Designer for that synonym.
Generate Create Script	Writes the synonym definition to a `.sql` file
Generate Create Script to Project	Generates the synonym definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Delete	Drops the synonym and its child nodes.
Rename	Allows you to edit the Synonym name in place in Server Explorer. Alternatively, you can select the Synonym name and then click one more time to open an edit box.
Refresh	Updates the Synonym node tree, including its associated synonyms nodes. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Synonyms Node | Synonym Designer

Sequences Node

This section covers the following topics:

- About Sequences
- How the Sequences Node Works
- Menu Options

About Sequences

A sequence is an object that generates a serial list of unique numbers for numeric columns of a database's tables. Once a user generates a sequence value, that user can continue to access that value regardless of whether the sequence is incremented by another user. You can use sequences to automatically generate primary key values. You can configure sequences to start and stop within a range of values.

How the Sequences Node Works

The Sequences node contains one or more child Sequence nodes. To perform actions on the Sequences node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Sequence	Creates a new sequence using the Sequence Designer.
Generate Create Script	Writes the sequence definitions to a `.sql` file. If the object associated with the sequences node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.
Generate Create Script to Project	Generates the sequence definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Refresh	Updates the Sequences node tree, including its associated sequence nodes.
Properties	Displays the Properties window.

See Also

Sequence Nodes | Sequence Designer

Sequence Nodes

This section covers the following topics:

- How Sequence Nodes Work
- Menu Options

How Sequence Nodes Work

The Sequence node is a child of the Sequences node.

Starting with Visual Studio 2005, you can edit the name of the sequence in place by selecting the name and then clicking it to open an edit box or by choosing the Rename menu option to open the edit box.

Note: This feature only works in Visual Studio 2005.

To perform actions on the Sequences node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Edits the sequence using the Sequence Designer. Alternatively, double-clicking a sequence node opens the Sequence Designer for that sequence.
Generate Create Script	Writes the sequence definition to a `.sql` file.
Generate Create Script to Project	Generates the sequence definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Delete	Drops the sequence.
Rename	Allows you to edit the Sequence name in place in Server Explorer. Alternatively, you can select the Sequence name and then click one more time to open an edit box.

Menu Option	Description
Refresh	Updates the Sequence node tree, including its associated sequence nodes. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Sequences Node |Sequence Designer

XML Database Node

This section covers the following topics:

- About XML Databases
- How the XML Database Node Works
- Menu Options

About XML Databases

Oracle XML Database is an extension to Oracle Database. It allows faster retrieval and search capabilities, access control, and versioning of XML documents. In addition, it provides SQL access to the same XML data. The XML instance documents saved in the database must conform to an XML schema.

For more information about Oracle XML databases, see *Oracle XML DB Developer's Guide*.

How the XML Database Node Works

The XML Database node contains all XML-related nodes in Server Explorer. It has one child node, the XML Schemas node, which in turn can have one or more child XML Schema nodes.

To perform actions on the XML Database node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Refresh	Updates the XML Database node tree, including its associated XML Schema nodes. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

XML Schemas Node | XML Schema Nodes

XML Schemas Node

This section covers the following topics:

About XML Schemas

An XML schema is written in a schema definition language in XML that describes the structure and other semantics of a conforming instance document. You must register the XML schema in an Oracle XML database before you can use the Oracle XML database. Before you register the XML schema, you must know the following:

- Whether the data for the XML instance documents already exists in relational tables. If this is the case, you must create Object Views for XML.
- Whether the storage model is LOB or Object Relational. This depends on which parts of the document are queried often and thus will need faster retrieval.
- Whether the XML schema document is annotated with comments to generate object data types and object tables. If not, you must create and manually map these objects to the XML schema. If the XML schema is already annotated with information to automatically generate object types and object tables, the XML database automatically generates these objects and you can insert conforming XML documents into it.

Use the XML schema to create complex types and storage elements for XML documents based on the XML schema. This type is available for Oracle version 9.2 and above.

For more information about XML schemas and Oracle XML databases, see *Oracle XML DB Developer's Guide*.

How the XML Schemas Node Works

The XML Schemas node, which is the only child of the XML Database node, contains one or more child XML Schema nodes. To perform actions on the XML Schemas node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New XML Schema	Creates a new XML Schema in the XML Schema Designer.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Schemas node tree, including its associated schema nodes.
Properties	Displays the Properties window.

See Also

XML Database Node | XML Schema Nodes | XML Schema Designer

XML Schema Nodes

This section covers the following topics:

How XML Schema Nodes Work

An XML Schema node represents an XML schema in the database. Each XML schema has a fully qualified URL; this URL represents the XML schema in Server Explorer. Examples of URLS that represent XML schemas are as follows:

- `http://schema1.mycompany.com`
- `http://schema2.yourcompany.com`

The XML Schema node is a child of the XML Schemas node, which in turn is a child of the XML Database node. To perform actions on an XML Schema node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Delete	Drops the XML Schema.
Refresh	Updates the Schema node.
Properties	Displays the Properties window.

See Also

XML Schemas Node | XML Database Node | XML Schema Designer

Java Classes Node

This section covers the following topics:

About Java Classes

If you have installed Oracle Database with the Oracle JVM (Java virtual machine) option, the database is Java-enabled. Oracle JVM is the Java virtual machine provided with the Oracle Database and it supports the database session architecture. This means that any database session can activate a dedicated JVM. All sessions share the same JVM code and statics; however, private states for any given session are held, and subsequently garbage collected, in an individual session space.

Java classes must be loaded in a database schema before they can be invoked. Java class invocation is secured and controlled through database authentication and authorization, Java 2 security, and invoker's or definer's rights.

For more information on OracleJVM, see *Oracle Database Java Developer's Guide*.

How the Java Classes Node Works

The Java Classes node represents a grouping of Java classes. It contains one or more child Java Class nodes. To perform actions on the Java Classes node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select **Properties** from the menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Refresh	Updates the Java Classes node tree, including its associated Java Class nodes.
Properties	Displays the Properties window.

See Also

Java Class Nodes

Java Class Nodes

This section covers the following topics:

- How Java Class Nodes Work
- Menu Options

How Java Class Nodes Work

The Java Class node is a child of the Java Classes node. It represents a Java class. To perform actions on the Java Classes node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Refresh	Updates the Java Class node. If the database object associated with the node was deleted from the database, you will be asked if the node needs to be deleted from the Server Explorer.
Properties	Displays the Properties window.

See Also

Java Classes Node

User-Defined Types Node

This section covers the following topics:

- About User-Defined Types Node
- How User-Defined Types Work
- Menu Options

About User-Defined Types Node

A user-defined type uses Oracle built-in data types and other user-defined types as the building blocks of object types that model the structure and behavior of data in applications.

See Also

Oracle Data Provider for .NET Developer's Guide and *Oracle Database Object-Relational Developer's Guide* for further information about Oracle Objects.

How User-Defined Types Work

The User-Defined Types node contains one or more user-defined type node that represents Object Types, VARRAY Types, and Nested Table Types in the database.

To perform actions on a node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Object Type	Launches the New Object Table Dialog.
New Nested Table	Launches the New Nested Table Dialog.
New Varray	Launches the New VARRAY Dialog.
Generate Create Script	Writes the user-defined type definitions to a `.sql` file. If the object associated with the user-defined types node is deleted from the database backend, this menu option does the following: Displays a message indicating that this object no longer exists in Oracle Database, and the script will not be generated.

Menu Option	Description
Generate Create Script to Project	Generates the user-defined type definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Refresh	Updates the user-defined types node, including its associated nodes.
Properties	Displays the Properties window.

See Also

Object Type Nodes | Object Type Body Nodes | Object Type Attribute Nodes | Object Method Nodes | VARRAY Type Nodes | Nested Table Type Nodes

Users Node

This section covers the following topics:

- About Users Node
- How the Users Node Works
- Menu Options

About Users Node

The Users node displays all local, external and global users (or schemas) that you have selected in the Filters Tab on the Connection Dialog.

How the Users Node Works

The Users node is visible from the Server Explorer tree when using the object view. You enable the object view by right-clicking the connection node and selecting **Change View**, then **Objects**.

To perform actions on the Users node: right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New User	Launches the designer to create a new user.
Generate Create Script	Writes the user definitions to a `.sql` file.
Generate Create Script to Project	Generates the user definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the privileges dialog.
Refresh	Refreshes the users node.
Properties	Displays the properties window.

See Also

User Nodes| User Designer | About Users, Roles, and Privileges

User Nodes

This section covers the following topics:

- How the User Nodes Work
- Menu Options

How the User Nodes Work

The User node is a child of the Users node. It represents a user in the database. If you expand the Users node, a node appears for each user selected in the Filters Tab on the Connection Dialog.

If you expand a user node, for example, `HR`, the database object type nodes such as Tables or Views, appear as child nodes under the `HR` user node.

To perform actions on a User node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Opens the designer so that you can edit the node.
Generate Create Script	Writes the user definition to a `.sql` file.
Generate Create Script to Project	Generates the user definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Grant Debugging Privileges	Opens the Grant Debugging Privileges Dialog which grants to the user privileges neccessary for PL/SQL debugging on this computer. This menu option is only visible to users with SYSDBA privileges.
Privileges	Opens the privileges dialog.
Delete	Deletes the user selected. Disabled for the current (connected) user.
Refresh	Refreshes user properties fetched from database.
Properties	Displays the properties window.

See Also

Users Node| User Designer

Roles Node

This section covers the following topics:

- About Roles
- How Roles Work
- Menu Options

About Roles

A role is a set of privileges that you can grant to users or to other roles.

Users (generally administrators) create roles to group together privileges or other roles. Roles provide a means to facilitate the granting of multiple privileges or roles to users.

There are two types of roles: predefined roles and user-defined roles.

Roles are not part of any schema: Therefore, a user who creates a role can be dropped with no effect on the role.

When you are logged into Server Explorer, in schema view, as `SYSDBA`, you view all roles in the database. If you are a non-`SYSDBA` user, or a `SYSDBA` user viewing Server Explorer in object view, you view the roles that are assigned to you.

See Also

Object View and Schema View

How Roles Work

To perform actions on the Roles node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Option	Description
New Role	Launches the designer to create a new role.
Generate Create Script	Writes the role definitions to a `.sql` file..

Menu Option	Description
Generate Create Script to Project	Generates the role definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Opens the privileges dialog.
Refresh	Refreshes the Roles node.
Properties	Displays the Properties window.

See Also

Role Nodes| Role Designer

Role Nodes

This section covers the following topics:

- How Role Nodes Work
- Menu Options

How Role Nodes Work

To perform actions on a Role node: right-click this node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Opens the designer window for the role being edited.
Generate Create Script	Writes the role definition to a `.sql` file.
Generate Create Script to Project	Generates the role definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Privileges	Opens the privileges dialog.
Delete	Deletes the role selected.
Refresh	Refreshes the node.
Properties	Displays the Properties window.

See Also

Roles Node| Role Designer

Object Type Nodes

This section covers the following topics:

- About Object Type Nodes
- Menu Options

About Object Type Nodes

An Object Type node represents an Object Type defined in the database. Each Object Type node appears as a child of the User-Defined Types collection node.

Object Type nodes can have the following child nodes:

- Object Attribute nodes that represent each of the object attributes.
- Object Method nodes that represent each of the object methods.

Each Object Type node that has a body associated with it in the database also has a corresponding Object Type Body node. See *Oracle Database Object-Relational Developer's Guide*.

To perform actions on this node, right-click the node and from the menu, choose the appropriate command.

To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Object Type Specification…	Opens the Object Type specification in the PL/SQL Editor. The PL/SQL editor contains text you can edit, starting with TYPE *object type name*.
Edit Object Type Body…	Opens the Object Type body in the PL/SQL Editor. The PL/SQL editor contains text you can edit, starting with TYPE BODY *object type name*.
Add Object Type Body…	Opens a PL/SQL Editor window that contains editable text starting with OBJECT TYPE BODY *Object Type name*. Choosing this option creates an empty object type body template based on the properties of the Object Type. You can then modify the Object Type Body present in the code window as needed and save it.
Compile	Performs following actions: If the Object Type specification and/or the Object Type body is open in the PL/SQL Editor and the contents of the window has been changed, saves the changes to the database. Compiles the Object Type specification and Object Type body. If the compilation fails, the node icon changes to indicate an error. See PL/SQL Compiler Settings Options Page for compile settings.
Compile Debug	Performs following actions: If the Object Type specification and/or the Object Type body is open in the PL/SQL Editor and contents of the window have been changed, saves the changes to the database. Compiles the Object Type specification and Object Type body with Debug information. If the object is successfully compiled, the node icon changes to indicate that this node has been compiled in debug mode. If the compilation fails, the node icon changes to indicate an error. See PL/SQL Compiler Settings Options Page for compile settings.
Generate Custom Class	Launches the Oracle Custom Class Wizard for the corresponding object type. See Oracle Custom Class Wizard.
Generate Create Script…	Writes the object type definition to a `.sql` file.

Menu Option	Description
Generate Create Script to Project	Generates the object type definition to a `.sql` file and adds the `.sql` file to the open Oracle Database Project. Generates the object type node definition to a .`sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Privileges	Opens the Grant/Revoke Privileges dialog box.
Delete	Drops the Object Type specification and Object Type body from the database
Refresh	Updates the Object Type node.
Properties	Displays the Properties window.

See Also

Object Type Body Nodes | Object Type Attribute Nodes | Object Method Nodes | VARRAY Type Nodes | Object Type Designer

Object Type Body Nodes

This section covers the following topics:

- About Object Type Body Nodes
- How Object Type Body Nodes Work
- Menu Options

About Object Type Body Nodes

The object type body contains the implementation of the member methods defined in the object type specification. Member methods defined in an object type body can be used on the object type instance only if their specifications appear in the object type specification. The object type body contains the code for the methods that implement the type. For each method specified in an object type specification for which you did not specify the `call_spec`, you must specify a corresponding method body in the object type body.

How Object Type Body Nodes Work

The Object Type body node is one of the child nodes of an Object Type node. This node appears only if the object type has an object type body.

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Object Type Body…	Opens the Object Type body in the PL/SQL Editor. The PL/SQL editor displays editable text starting with `OBJECT TYPE BODY` *`Object Type name`*. If the window already exists, Edit Object Type Body makes it the current window.
Compile	Performs the following actions: If the Object Type body is open in the PL/SQL Editor and has been changed, saves the changes to the database. Compiles only the Object Type body. If the compilation fails, the node icon changes to indicate an error. See PL/SQL Compiler Settings Options Page for compile settings.
Compile Debug	Performs the following actions: If the Object Type body is open in the PL/SQL Editor and has been changed, saves the changes to the database. Compiles the Object Type body with debug information. If the object is successfully compiled, the node icon changes indicate that this node has been compiled in debug mode. If the compilation fails, the node icon changes to indicate an error. See PL/SQL Compiler Settings Options Page for compile settings.
Generate Custom Class	Launches the Oracle Custom Class Wizard for the corresponding object type. See Oracle Custom Class Wizard.
Generate Create Script…	Writes the object type body definition to a `.sql` file. See Managing Oracle Script Files.
Generate Create Script to Project	Generates the object type body definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Delete	Drops the Object Type body from the database
Refresh	Updates the Object Type body node.
Properties	Displays the Properties window.

See Also

Object Type Nodes | Object Type Attribute Nodes | Object Method Nodes | VARRAY Type Nodes | Object Type Designer

Object Type Attribute Nodes

This section covers the following topics:

About Object Type Attribute Nodes

An Object Type Attribute node is a child node of an Object Type node. It represents an attribute in the object type. An inherited attribute displays a different icon than the non-inherited attribute.

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit Object Type Specification…	Opens the Object Type specification in the PL/SQL Editor. The Object Type specification appears in the code window as editable text, starting with `OBJECT TYPE` *`Object_Type_name`*.
Refresh	Updates the Object Type node and the associated child nodes.
Properties	Displays the Properties window.

See Also

Object Type Nodes | Object Method Nodes | VARRAY Type Nodes | Object Type Designer

Object Method Nodes

This section covers the following topics:

About Object Method Nodes

An Object Method node is a child node of an Object Type node. It represents a member method in the object type. A member method can be a procedure or a function on a constructor of the Object Type.The node is named after the method. Overloaded methods in the object type have the same name. Overloaded constructors in the object type have the same name. Specific icons indicate if a method is a function or procedure. Object methods of the base type that are overridden in the derived UDT type are displayed, but those that are not overridden are not displayed.

Menu Options

Menu Option	Description
Edit Object Type Specification…	Opens the Object Type specification in the PL/SQL Editor. The Object Type specification appears in the code window as editable text, starting with `OBJECT TYPE` *`Object_Type_name`*.
Edit Object Type Body…	Opens the Object Type body in the PL/SQL Editor. The Object Type body appears in the code window as editable text, starting with `OBJECT TYPE BODY` *`Object_Type_name`*. If the associated Object Type body (with the node) is deleted from the database, then a message appears indicating that this object no longer exists in the database.
Refresh	Updates the Object Type node and the associated child nodes.
Properties	Displays the Properties window.

See Also

Object Type Nodes | Object Type Attribute Nodes | VARRAY Type Nodes | Object Type Designer

VARRAY Type Nodes

This section covers the following topics:

- About VARRAY Type Nodes
- How VARRAY Type Nodes Work
- Menu Options

About VARRAY Type Nodes

A VARRAY Type node is a child of the User-Defined Types node. It represents a VARRAY type defined in the database.

How VARRAY Type Nodes Work

A VARRAY type node has a different icon than an object type node or nested table type node.A VARRAY type node will have a child node only if the VARRAY type has an element of user-defined type in the database.

For more information, see *Oracle Database Object-Relational Developer's Guide*.

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Opens the type definition for the VARRAY Type in a PL/SQL Editor Window.

Menu Option	Description
Generate Custom Class	Launches the Oracle Custom Class Wizard for the selected VARRAY type. See Using the Oracle Custom Class Wizard.
Generate Create Script…	Writes the VARRY type definition to a `.sql` file.
Generate Create Script to Project	Generates the VARRAY type definition to a `.sql` file and adds the `.sql` file to the open Oracle Database Project. Generates the VARRAY type node definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Delete	Drops the VARRAY type from the database.
Refresh	Updates the VARRAY Type node.
Properties	Displays the Properties window.

See Also

Object Type Nodes | Object Type Attribute Nodes | Object Method Nodes | VARRAY Type Nodes | Varray Designer | Nested Table Type Designer

Nested Table Type Nodes

This section covers the following topics:

- About Nested Table Type Nodes
- How Nested Table Type Nodes Work
- Menu Options

About Nested Table Type Nodes

A Nested Table Type node is a child of the User-Defined Types node. It represents a Nested Table type defined in the database.

How Nested Table Type Nodes Work

A Nested Table type node has a different icon than an object type node or VARRAY type node.A Nested Table type node will have a child node only if the Nested Table type has an element of user-defined type in the database.

For more information, see *Oracle Database Object-Relational Developer's Guide*.

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
Edit	Opens the type definition for the Nested Table type in a PL/SQL Editor Window.
Generate Custom Class	Launches the Oracle Custom Class Wizard for the selected Nested Table type. See Using the Oracle Custom Class Wizard.
Generate Create Script…	Writes the Nested Table type definition to a `.sql` file.
Generate Create Script to Project	Generates the Nested Table type definition to a `.sql` file and adds the `.sql` file to the open Oracle Database Project. Generates the Nested Table type node definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Delete	Drops the Nested Table type from the database.
Refresh	Updates the Nested Table Type node.
Properties	Displays the Properties window.

See Also

Object Type Nodes | Object Type Attribute Nodes | Object Method Nodes | VARRAY Type Nodes | Varray Designer | Nested Table Type Designer

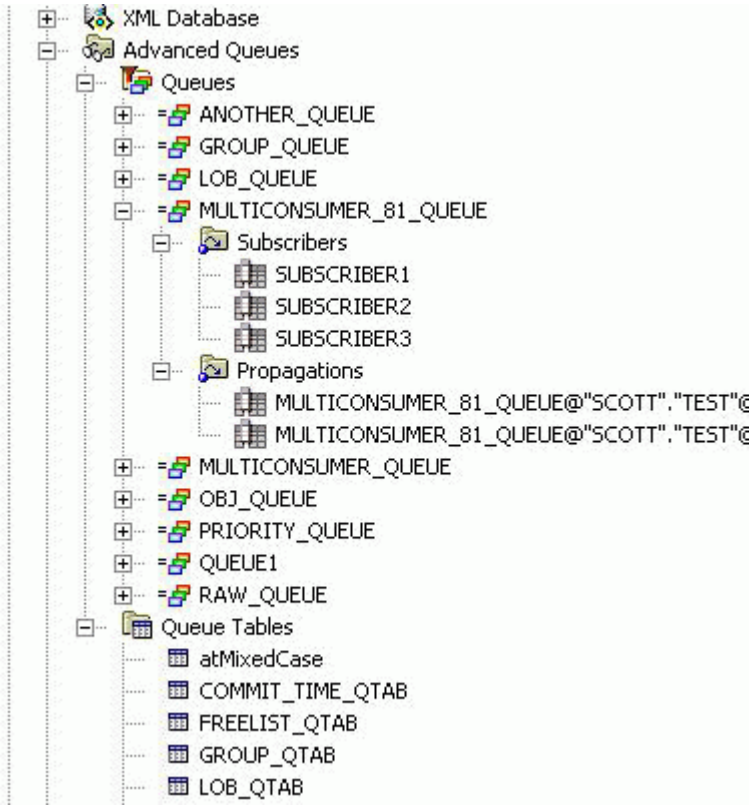
Advanced Queues Node

This section covers the following topics:

- About Advanced Queues Node
- How Advanced Queues Node Works
- Menu Options

About Advanced Queues Node

The Advanced Queues node appears under the connection node in the Object view and under individual schema nodes in Schema View. See About Oracle Streams Advanced Queueing. This section of the Server Explorer is similar to the following:



How Advanced Queues Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Options	Description
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Queues Node

This section covers the following topics:

- About the Queues Node
- How the Queues Node Works
- Menu Options

About the Queues Node

Queues collection node contain queue nodes for each queue in your schema and restriction list. See About Oracle Streams Advanced Queueing.

How the Queues Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Options	Description
New Queue	Launches the Queue Designer to create a new queue. See Queue Designer.
Generate Create Script	Writes the queue definitions to a `.sql` file.

Menu Options	Description
Generate Create Script to Project	Generates the queue definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Launches the existing Grant/Revoke privileges dialog.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Queue Nodes

This section covers the following topics:

- About Queue Nodes
- How Queue Nodes Work
- Menu Options

About Queue Nodes

Each node under Queues node represents an Oracle Streams AQ Queue in the database. See About Oracle Streams Advanced Queueing.

How Queue Nodes Work

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Options	Description
Design Queue	Launches the Queue Designer to modify the queue. See Queue Designer.
Start Enqueuing	Issues a Start Enqueuing command on this queue.
Start Dequeuing	Issues a Start Dequeuing command on this queue.
Generate Create Script	Writes the queue definitions to a `.sql` file.
Generate Create Script to Project	Generates a SQL script to create all queues in this collection and writes it to the Oracle Database project you select. Generates the queue definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Privileges	Launches the existing Grant/Revoke privileges dialog.
Delete	Deletes this queue from the database.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Subscribers Node

This section covers the following topics:

- About the Subscribers Node
- How the Subscribers Node Works
- Menu Options

About the Subscribers Node

Subscribers node is a child to each Queue node in the Server Explorer. It will contain all the subscriber nodes of the queue. See About Oracle Streams Advanced Queueing.

How the Subscribers Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Options	Description
Design Queue	Launches the Queue Designer with Subscribers tab active. See Queue Designer.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Subscriber Nodes

This section covers the following topics:

- About Subscriber Nodes
- How the Subscriber Nodes Work
- Menu Options

About Subscriber Nodes

Each subscriber node represents one subscriber of the queue. These subscribers are also accessible through the Subscribers tab of the Queue Designer. See About Oracle Streams Advanced Queueing.

How the Subscriber Nodes Work

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Options	Description
Design Queue	Launches the Queue Designer with the Subscribers tab active and the selected subscriber selected. See Queue Designer.
Delete	Deletes this subscriber from the queue.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Propagations Node

This section covers the following topics:

- About Propagations Node
- How the Propagations Node Work
- Menu Options

About Propagations Node

Propagation node is a child to each Queue node in the Server Explorer. It contains all the propagation nodes of the queue. See About Oracle Streams Advanced Queueing.

> **Note**
>
> The propagations node is only visible to SYSDBA users.

How the Propagations Node Work

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Options	Description
Design Queue	Launches the Queue Designer with the Propagations tab active. See Queue Designer.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Propagation Nodes

This section covers the following topics:

- About Propagation Nodes
- How the Propagation Nodes Work
- Menu Options

About Propagation Nodes

Each propagation node represents one propagation of the queue. These propagations are also accessible through the Propagations tab page of the Queue Designer. See About Oracle Streams Advanced Queueing.

> **Note**
>
> The propagation nodes are only visible to `SYSDBA` users.

How the Propagation Nodes Work

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Options	Description
Design Queue	Launches the Queue Designer with the Propagation tab active and the propagation selected. See Queue Designer.
Enable/Disable Propagation	Enables or disables the propagation depending on the current state For example if the propagation is already started, the menu shows Disable.
Delete	Deletes this propagation from the queue

Menu Options	Description
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Queue Tables Node

This section covers the following topics:

- About Queue Tables Node
- How Queue Tables Node works
- Menu Options

About Queue Tables Node

This node contains the nodes for all your queue tables. See About Oracle Streams Advanced Queueing.

How Queue Tables Node works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

If this collection node is filtered, the icon changes to include a funnel symbol. For more information about filtering, see Filtering Collection Nodes.

Menu Options

Menu Options	Description
New Queue Table	Launches the Queue Table Designer to create a new queue table. See Queue Table Designer.
Generate Create Script	Writes the queue table definitions to a `.sql` file.

Menu Options	Description
Generate Create Script to Project	Generates the queue table definitions to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Filters	Opens the Filters Tab to control which child nodes appear under the collection.
Privileges	Launches the existing Grant/Revoke privileges dialog.
Copy	Copies all selected nodes
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

Queue Table Nodes

This section covers the following topics:

- About Queue Table Nodes
- How Queue Table Nodes Work
- Menu Options

About Queue Table Nodes

Each node under the Queue Tables node represents a Queue table in the database.

See About Oracle Streams Advanced Queueing.

How Queue Table Nodes Work

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Options	Description
Design Queue Table	Launches the Queue Table Designer to create, modify, and save queue tables. See Queue Table Designer.
Purge Messages	Removes all messages from the queue table. This functionality is only available in Oracle Database 10.1*g* release 10.1 or higher.
Generate Create Script	Writes the queue table definition to a `.sql` file.
Generate Create Script to Project	Generates the queue table definition to a `.sql` file and adds the `.sql` file to an open Oracle Database Project. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into the various schema object folders and this master script is ordered with dependencies taken into consideration. If there are multiple open Oracle Database projects, Visual Studio prompts you to select the project to add the script to. This operation can also be performed by dragging and dropping the Server Explorer node(s) directly onto a database project folder. See Managing Oracle Script Files for more information. **Note**: Oracle Database Project Version 2 projects are not supported. To add scripts to this project type, use Import Schema or Add Existing Item menu items on an Oracle Database Project Version 2 project folder, or use the Schema Compare tool.
Privileges	Launches the existing Grant/Revoke privileges dialog.
Copy	Copies the selected node.
Delete	Deletes this queue table from the database. An error `ORA-24012` is thrown if there are dependent queues for this queue table. The error message asks if you want to force the delete operation. If you select `YES`, all dependent queues will be deleted.
Refresh	Refreshes the node.
Properties	Shows properties for the node.

See Also

About Oracle Streams Advanced Queueing | Queue Table Designer | Queue Designer

ADDM Tasks Node

This section covers the following topics:

- About ADDM Tasks
- How the ADDM Tasks Node Works

About ADDM Tasks

This node contains all the ADDM tasks defined for the current user schema and any schemas the user has access to. Each child node under ADDM Tasks node represents individual ADDM task.

See Also

- Using the Oracle Performance Analyzer
- *Oracle Database Performance Tuning Guide* - Chapter 5, Automatic Performance Statistics; Chapter 6, Automatic Performance and Diagnostics

How the ADDM Tasks Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu Option	Description
New ADDM Task	Launches the New ADDM Task Dialog to create and execute a new ADDM task. Oracle Performance Analyzer is launched to display the execution results.
Filters	Displays the Filters Tab on the Connection Dialog. This option is only available if the connection is already open.
Refresh	Refreshes the node.
Properties	Displays properties for this node.

See Also

About Oracle Performance Analyzer

ADDM Task Nodes

This section covers these topics:

How ADDM Task Nodes Work

An ADDM task is an analysis of Oracle database performance over a period of time. Each ADDM Task node represents a single ADDM task in the database. You can create ADDM Tasks in the following ways:

- Running Oracle Performance Analyzer
- Selecting the **New ADDM Task** by right-clicking on the ADDM Tasks Collection node or the AWR Snapshot nodes
- Calling the ADDM PL/SQL API directly using Query Window or SQL*Plus
- Using external tools such as Oracle Enterprise Manager

You can double-click an existing ADDM task to launch Oracle Performance Analyzer to load and view the findings and recommendations.

See Also

- Using the Oracle Performance Analyzer
- *Oracle Database Performance Tuning Guide* - Chapter 5, Automatic Performance Statistics; Chapter 6, Automatic Performance and Diagnostics

Menu Options

Menu Option	Description
View Analysis	Launches Oracle Performance Analyzer and loads the task. This is the default action when the node is double-clicked. If an ADDM task is ready, it comes to the front.
Delete	Deletes the ADDM tasks from the database.
Refresh	Refreshes the node.
Properties	Displays properties for the ADDM Task.

See Also

About Oracle Performance Analyzer

AWR Snapshots Node

This section covers the following topics:

About AWR Snapshots

This node contains all the AWR snapshots for this database, because AWR snapshots exist at the database level and do not pertain to specific schemas. Each child node under the AWR Snapshots node represents one AWR snapshot in the database.

See Also

- Using the Oracle Performance Analyzer
- *Oracle Database Performance Tuning Guide* - Chapter 5, Automatic Performance Statistics; Chapter 6, Automatic Performance and Diagnostics

How the AWR Snapshots Node Works

To perform actions on this node, right-click the node and from the menu, choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the node's menu.

Menu Options

Menu	Descriptions
New AWR Snapshot	Launches the New AWR Snapshot dialog which creates a new AWR snapshot. See New AWR Snapshot Dialog.
Filters	Displays the Filters Tab on the Connection Dialog. This option is only available if the connection is already open.
Refresh	Refreshes the node.
Properties	Displays properties for the node.

See Also

About Oracle Performance Analyzer

AWR Snapshot Nodes

This section covers the following topics:

- How AWR Snapshot Nodes Work
- Menu Options

How AWR Snapshot Nodes Work

This node represents a single AWR snapshot in the database. An AWR Snapshot is a collection of database statistics and performance metrics gathered at a single point in time. Two snapshots are required to analyze database performance over their time period. This performance analysis is done by creating an ADDM Task.

When you run Oracle Performance Analyzer, it automatically creates begin and end snapshots. The database may also create AWR snapshots automatically at regular intervals, depending on its configuration.

See Also

- Using the Oracle Performance Analyzer
- *Oracle Database Performance Tuning Guide* - Chapter 5, Automatic Performance Statistics; Chapter 6, Automatic Performance and Diagnostics

Menu Options

Menu Option	Description
New ADDM Task	Launches the new ADDM Task dialog. See New ADDM Task Dialog. This is the default action for this node when it is double-clicked.
Delete	Deletes the AWR snapshot from the database.
Refresh	Refreshes the node.
Properties	Displays properties for this node.

4

Wizards, Designers, and Dialogs

About Wizards, Designers, and Dialogs

Oracle Developer Tools provides Wizards, Designers, and Dialogs. They help you easily create new schema objects or alter existing schema objects. You launch the wizards and designers from the menus that display when you right-click nodes in Server Explorer.

See the following topics:

Common Dialog Box Functionality

The dialog boxes provided by the wizards and designers have the following common functionality:

- To proceed with the dialog box actions, either click **OK** or press the **Enter** key.
- To cancel the dialog box actions, either click **Cancel** or press the **Esc** key.
- To get help for the dialog box, either click **Help** or press the **F1** key.

Naming Convention for Schema Names

When you specify the name of any of the schema objects, enclose the schema name in double quotation marks to preserve that name's case.

Connection Dialog Box

Use the Connection dialog box to establish a connection to an Oracle database. This dialog box lets you specify detailed connection, such as the name of the Oracle data source to which you are connecting and the role you are assigned, and apply filters to select the schema nodes to be included in the display.

This section covers the following topics:

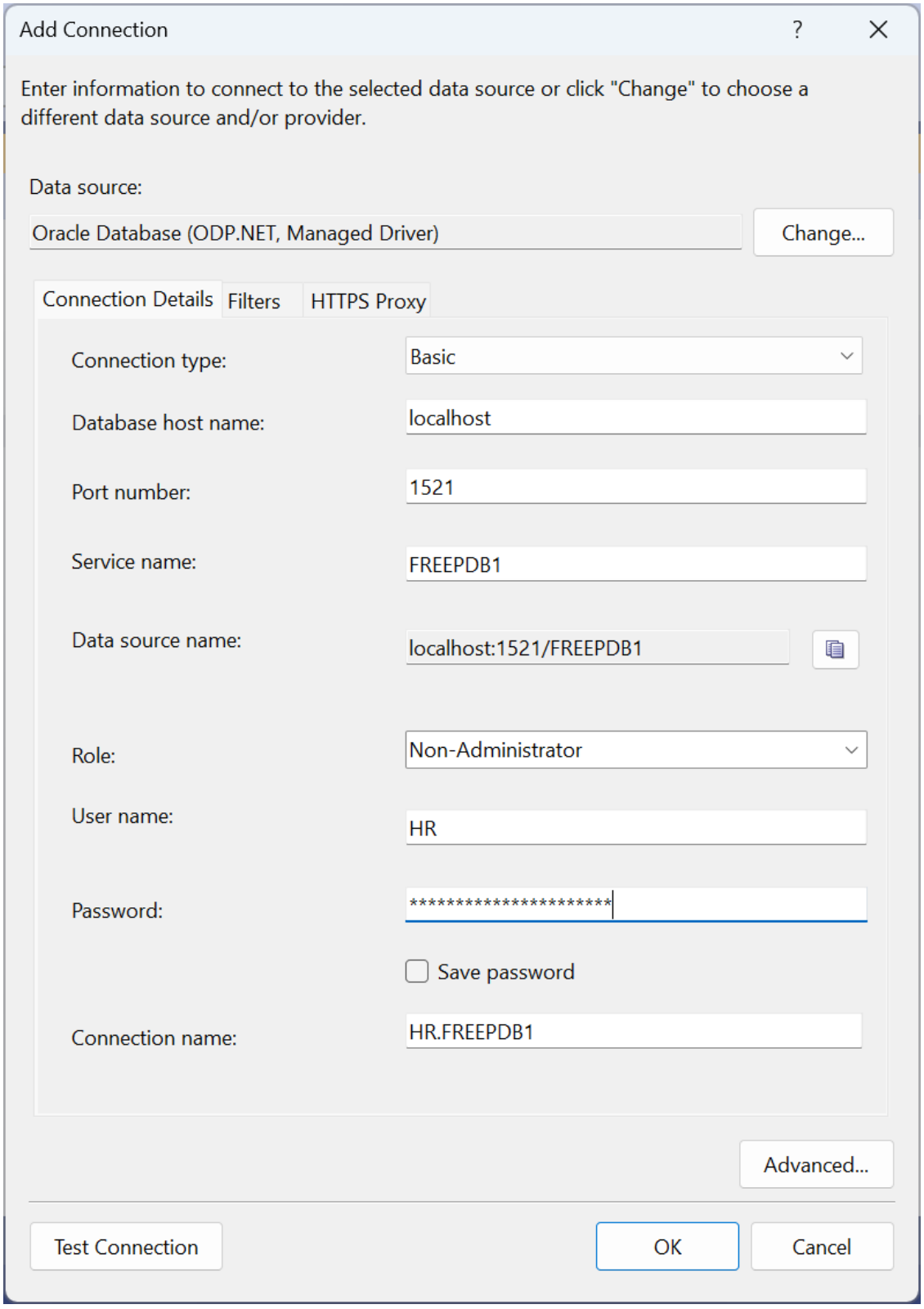
Opening the Connection Dialog Box

Open the Connection dialog box from either of the following locations:

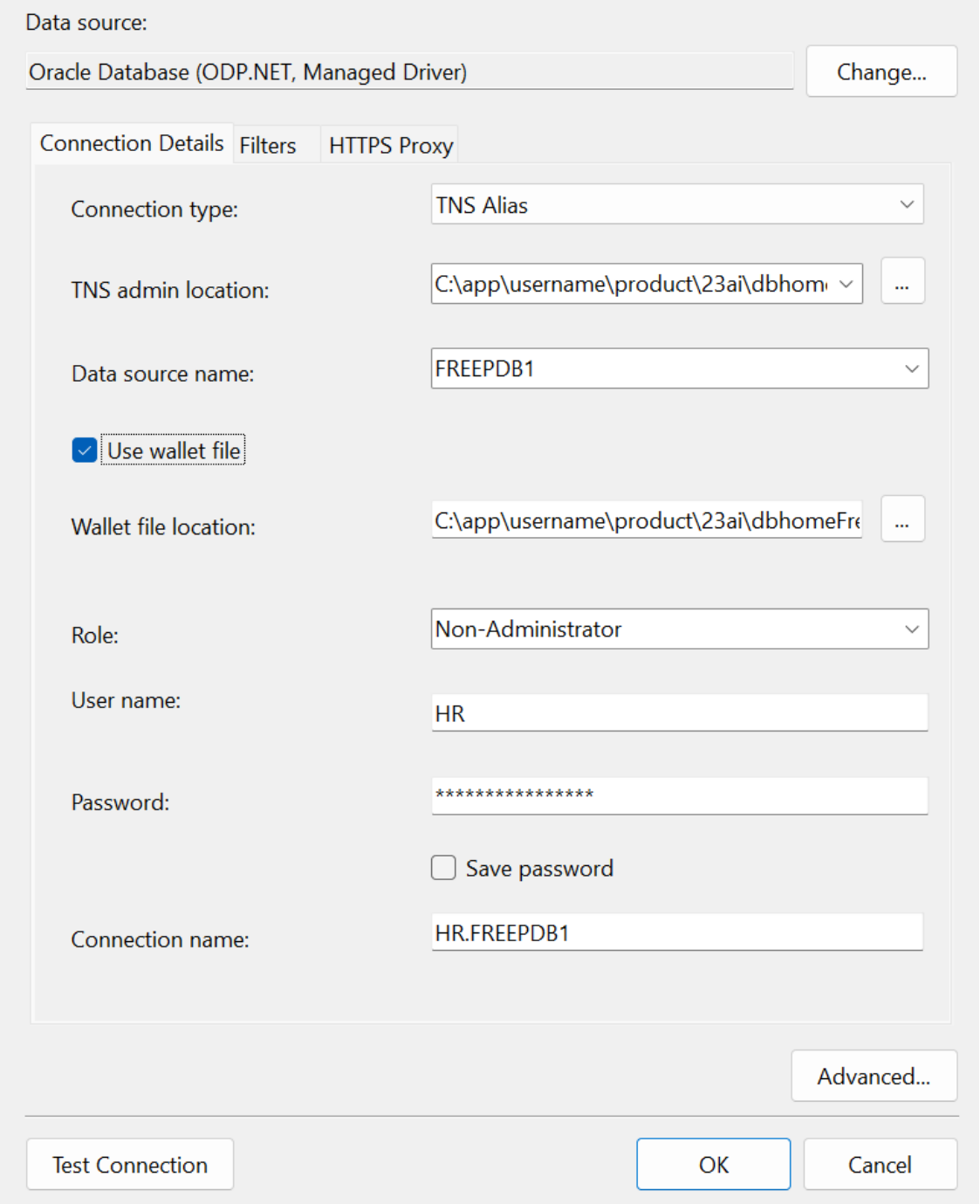
- In the toolbar at the top of the Server Explorer window, click the **Add Connection** icon.
- In the Server Explorer window, right-click the Data Connections node and from the menu, select **Add Connection** for a new connection or select **Modify Connection** to modify the existing connection.
- In the Visual Studio **Tools** menu, select **Connect to Database**.

The Add Connection and Modify Connection dialog boxes are similar to the following:

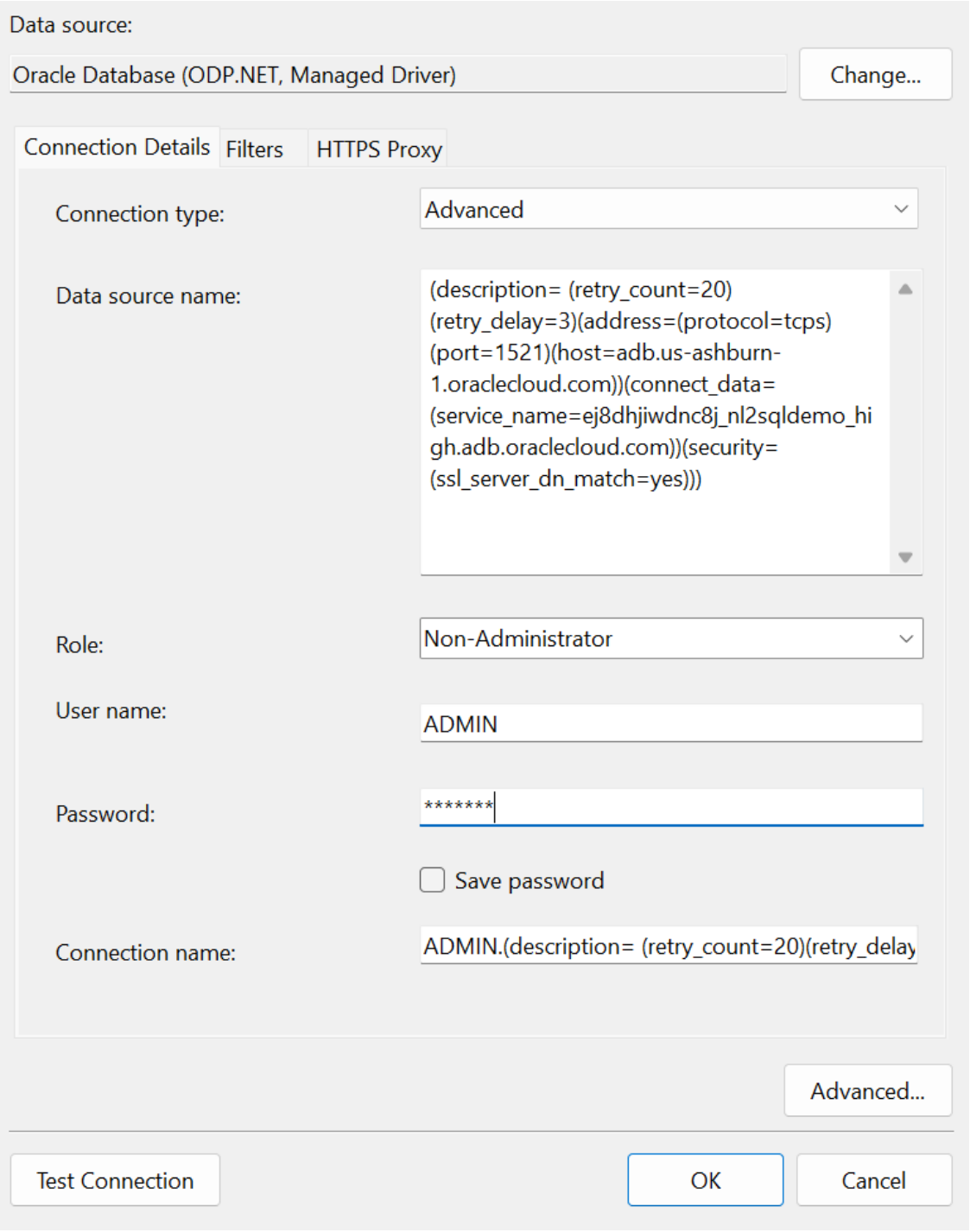
Basic Connection



TNS Alias



Advanced



Using the Connection Dialog Box

In general, to use the Add Connection dialog box, you enter the appropriate connection information, and optionally, click **Test Connection** to make sure the connection works.

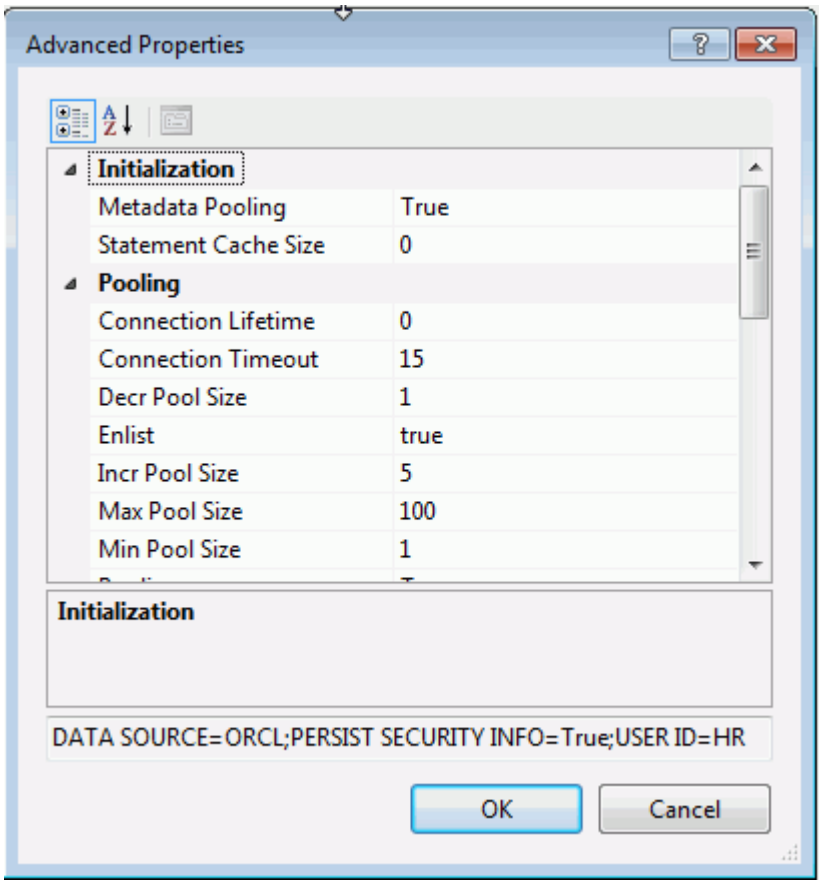
The controls in the Connection dialog box are as follows:

Control	Description
Data Source	This must be set to **Oracle Database (ODP.NET, Managed Driver)** in order to use this extension. If it is not set to that value, then press the **Change** button to open the Change Data Source dialog. In that dialog, in the Data Source list, select **Oracle Database**, and in the Data Provider dropdown, select **ODP.NET Managed Driver**.
Connection Type	Select the connection type representing the style of connection information you wish to provide. Depending on which connection type you choose, the dialog will change. • Basic: You will provide Host/IP, Port, and Service Name • TNS Alias: You will provide a connection alias that is defined in a `tnsnames.ora` file. • Advanced: You will provide a "Easy Connect" connection string or a TNS connect descriptor.
Database host name (Basic)	The host name or IP address of the local or remote computer where Oracle Database resides.
Port number: (Basic)	The port that the database listens on for incoming connection requests (default port is 1521 which most databases are configured to use).
Service name (Basic)	The service name for the database.
Data source name (Basic)	The Easy Connect Connection string that will be used to connect to the database. Use the Data Source Name Copy Button to copy this to the clipboard if needed (for example if you need to use the connection string within your application).
Data Source Name Copy button (Basic)	Copies the Easy Connect connection string to the clipboard for use elsewhere (for example, in code).
TNS Admin Location (TNS Alias)	Use the drop down to select the path to your `tnsnames.ora`, `sqlnet.ora`, and `ldap.ora` files. Alternatively, click the browse button a select the correct directory. You can set the default value for this field in the Connection Configuration Options Page.
Data source name drop-down list (TNS Alias)	Select a database alias from the drop-down list. The contents of the Data source name drop-down list is populated from the contents of your `tnsnames.ora` file and/or from an LDAP server (as configured in `sqlnet.ora` and `ldap.ora`). **NOTE:** For LDAP configurations where `NAMES.DIRECTORY_PATH` is set only to LDAP, the aliases inside a `tnsnames.ora` file will be ignored and will not appear in this drop-down list. To use both LDAP and a `tnsnames.ora` file simultaneously, set `NAMES.DIRECTORY_PATH=(LDAP,TNSNAMES)` in `sqlnet.ora`. For more information on configuring the `tnsnames.ora, sqlnet.ora`, and `ldap.ora` files, see *Oracle Net Services Reference*.
Data Source Name (Advanced)	Enter an Easy Connect connection string or TNS connect descriptor here.
Use wallet file (TNS Alias)	Check this checkbox if you wish to use a wallet file for authentication.
Wallet file location(TNS Alias)	Click the browse button a select the correct directory containing the wallet file. You can set the default value for this field in the Connection Configuration Options Page.
Role	Regular database users should select "Non-Administrator" from the dropdown. Autonomous Database "ADMIN" users should also select "Non-Administrator". For connecting as `SYSDBA` or `SYSOPER` to perform administrative tasks, select the appropriate role.
User name	Enter the username

Control	Description
Password	Enter the password
Save Password	Check this box to save the password for future use
Connection name	The connection name that Server Explorer uses to identify the connection. The name is automatically generated by default, but you can edit it.
Test connection	Click this option to test the connection information you have just entered.
Advanced	Opens the Advanced Properties Dialog Box.

Advanced Properties Dialog Box

You can access the Advanced Properties by clicking Advanced from the Add and Modify Connection Dialogs, or the Filters Tab on the Connection Dialog. Advanced properties allows you to modify advanced properties for the Oracle Connection. The connection type of the Add Connection dialog box determines the properties available.



Filters Tab

You can use filters to restrict the number of visible Server Explorer nodes.

This section covers the following topics:

- About the Filters Tab

About the Filters Tab

Use the Filters tab to create rules that determine which nodes appear in Server Explorer. For example, you can restrict Server Explorer to show only the HR schema, or show only the tables and views collection nodes, or you can filter the stored procedure collection to show only stored procedures that start with the letter a.

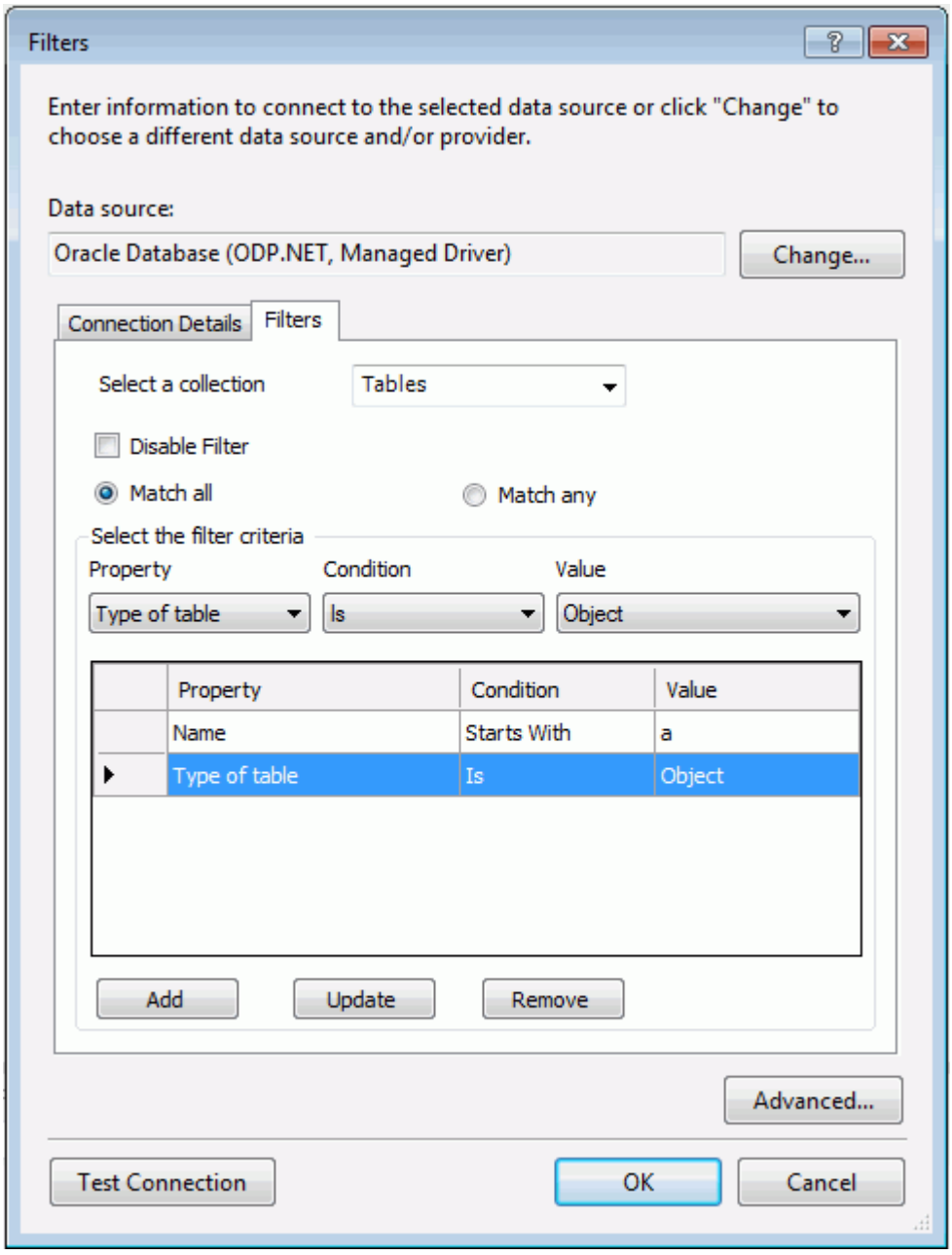
Filtering improves the performance of the Server Explorer by limiting the number of child nodes that are retrieved. This is also helpful when searching for specific subsets of child nodes. See Avoiding Performance Problems Due to Large Collections of Database Object Nodes.

Opening the Filters Tab

There are several ways to open the Filters tab.

- From the Connections node or the Connection nodes, right-click and select **Add Connection** or **Modify Connection**. Then select the Filters tab.
- Right-click on a connection node and select **Filters**. This opens the connection dialog with the Filters tab visible and the connection filters displayed.
- Right-click on a collection node (for example, Tables) and select **Filters**. This opens the connection dialog with the Filters tab visible and any existing collection filters displayed. If a collection node has an active filter, then the icon for the collection changes to indicate this.

The Filters tab appears similar to the following:



Collections of connections may function differently from other collections as indicated in "Connection Filters".

First select the collection type from the list and choose **Match all** or **Match any**, if applicable, and then perform any of these actions:

To add a new condition: Select a Property, Condition, and Value from the lists. Then, click **Add** to add this condition to the filter group. Click **OK** to save your changes and close the dialog. NOTE: Conditions cannot be added for connection filters: they can only be updated.

To update an existing condition: From the filter group, select the condition you wish to modify. Then, change Property, Condition, or Value, as needed, selecting from the specific lists, and click **Update**. Note: If you click on a different row in the filter group before clicking **Update**, the change is lost.

To delete an existing condition: From the filter group, select the condition you wish to delete. Click **Remove**. NOTE: Conditions cannot be deleted for connection filters: they can only be updated.

To disable the filter: Select **Disable Filter**. NOTE: Connection filters cannot be disabled.

To enable the filter: Deselect **Disable Filter**.

Once the filter is modified, click **OK** to save your changes and close the dialog.

Using the Filters Tab

The controls in the Filter tab are as follows:

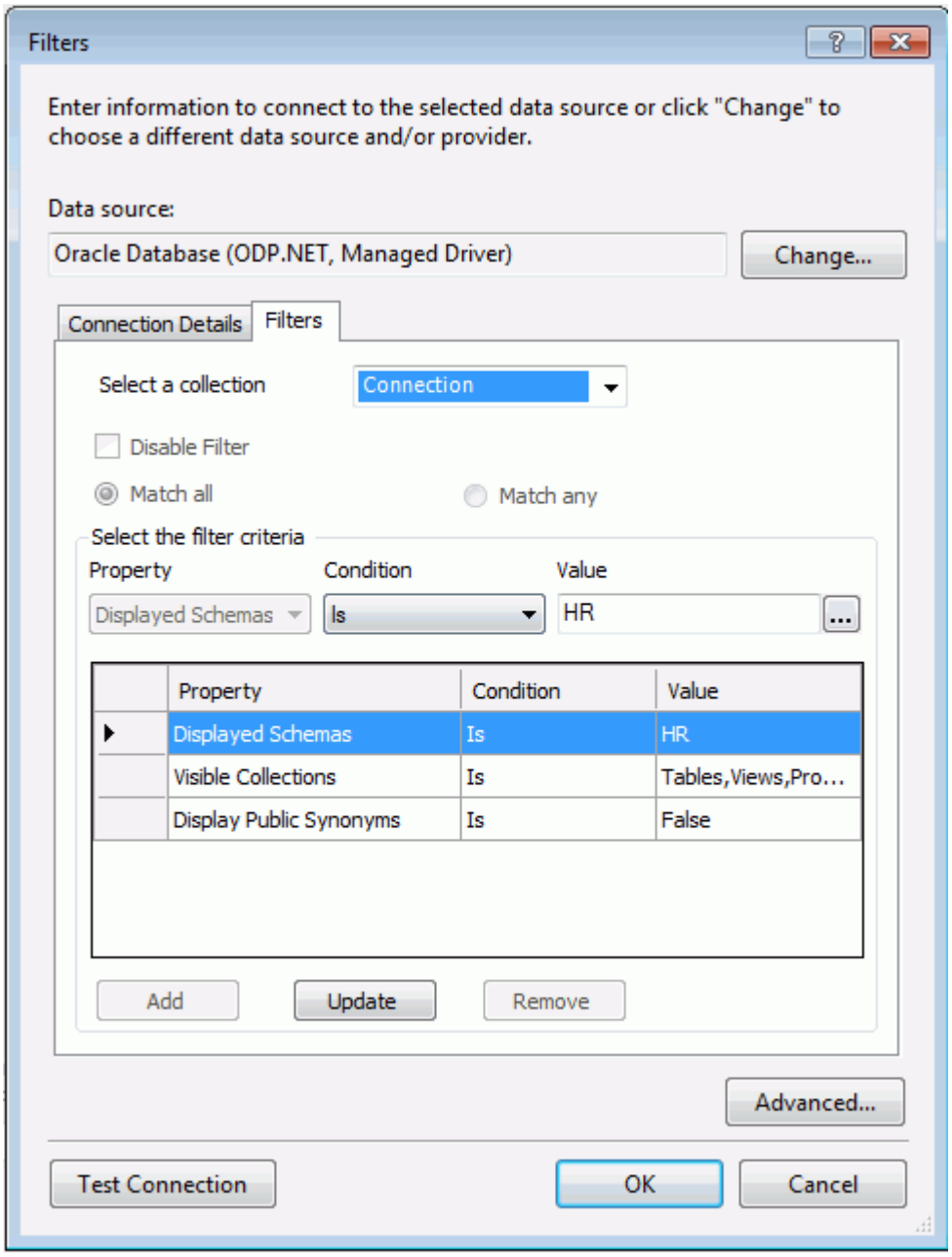
Control	Description
Select a Collection	Select a collection from the list to modify a filter for a particular collection node or select **Connection** to modify a connection filter.
Disable Filter	Disables this filter. Use this control to turn filtering on and off. This is disabled for Connection filters.
Match Condition	Toggles to match either all conditions in the filter or any condition. This is disabled for Connection filters.
Filter Conditions Group	Lists existing conditions and enables you to create new conditions with the following information: • **Property:** Select or change the property to filter. You are limited to properties available to the object type of the collection node. Common properties are: name, owner, creation date, and type. • **Condition:** Select or change the condition operator from the list of condition operators. Condition operators vary depending on the chosen filter property. • **Value:** Enter a custom value or select a standard value from the list of values.
Add	Adds a condition to this filter. This is disabled for Connection filters.
Update	Updates the selected condition.
Remove	Removes the selected condition. This is disabled for Connection filters.
OK	Saves the filter and closes the dialog. If Disable Filter is checked, the filter is saved, but inactive.
Cancel	Cancels any changes to the filter that you made and closes the dialog.

Connection Filters

Connection filters are a special type of filter whose conditions cannot be added, removed, or disabled, only updated. To modify connection filters, choose Connection from the Select a Collection list. Connection filter have three filter group types: Displayed Schemas, Visible Collections, and Displayed Public Synonyms. You can modify the Displayed Schemas and Visible Collections by clicking the ellipse icon that appears besides the value, showing the Select Schemas or Select Collections dialog respectively.

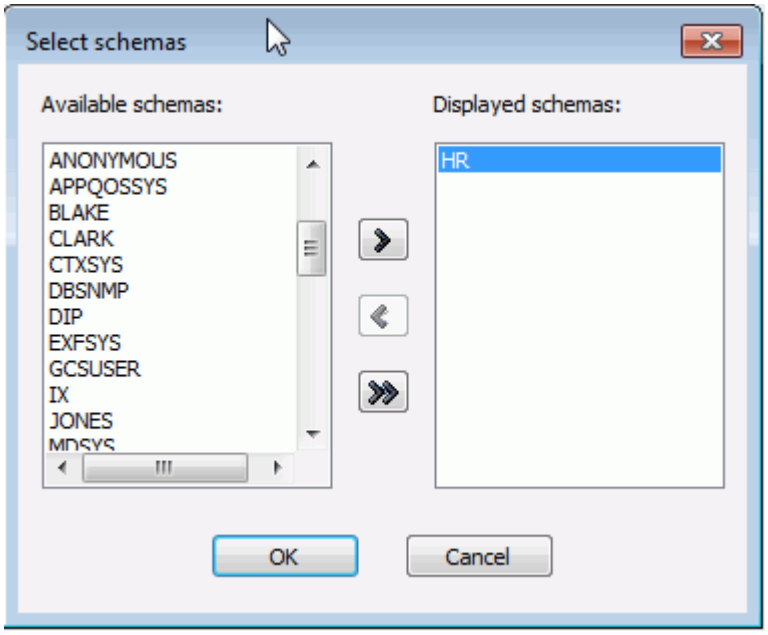
The default values for displayed collections is taken from the Data Connections Options Page at the time that the connection was created. After a connection is created, the Data Connections Options Page values for displayed collections are ignored.

The changes made here persist across Visual Studio sessions.



Select Schemas

This is the Select schemas dialog, which has arrows to move collections from Available schemas to Displayed schemas and vice versa.

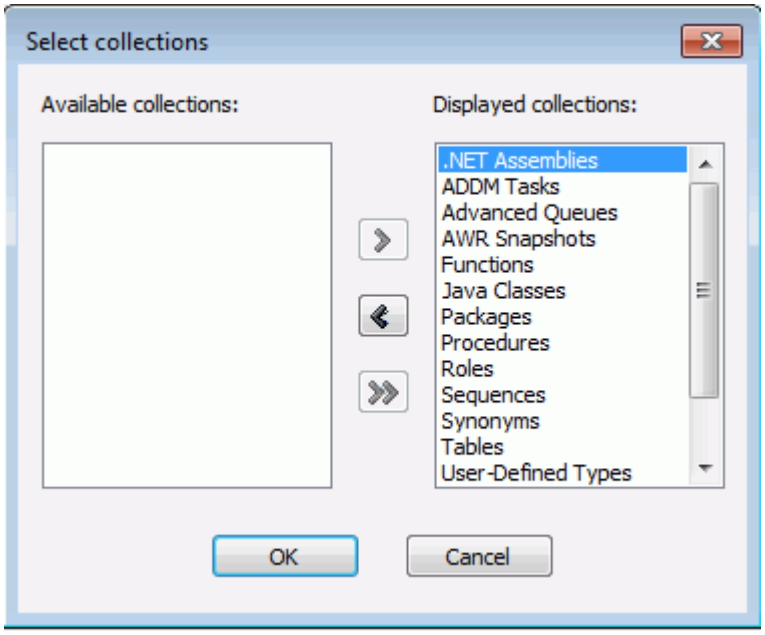


The controls in the Select schemas dialog box are as follows:

Control	Description
Available schemas	Lists schemas available to be displayed.
Displayed schemas	Lists currently displayed schemas. The changes made here persist across Visual Studio sessions.
Arrow buttons	Use the **Left** arrow, **Right** arrow, and **Right All** arrows to move the selection between the lists of available and displayed schemas.
OK	Applies the selected schema names in Server Explorer display for filtering.

Select Collections

This is the Select collection dialog, which has arrows to move collections from Available collections to Displayed collections and vice versa.



The controls in the Select collections dialog box are as follows:

Control	Description
Available collections	Lists collections available to be displayed.
Displayed collections	Lists currently displayed collections. The default values for displayed collections is taken from the Data Connections Options Page at the time that the connection was created. After a connection is created, the Data Connections Options Page values for displayed collections are ignored. The changes made here to the displayed collections persist across Visual Studio sessions.
Arrow buttons	Use the **Left** arrow, **Right** arrow, and **Right All** arrows to move the selection between the lists of available and displayed collections.
OK	Applies the selected collections names in Server Explorer display for filtering.

HTTPS Proxy Tab

You can use an HTTPS proxy to connect to an Oracle Database when the machine that hosts Visual Studio is behind a firewall or equivalent. This is commonly used to connect to Oracle Autonomous Database from behind a firewall.

This section covers the following topics:

- About the HTTPS Proxy Tab
- Opening the HTTPS Proxy Tab
- Using the HTTPS Proxy Tab

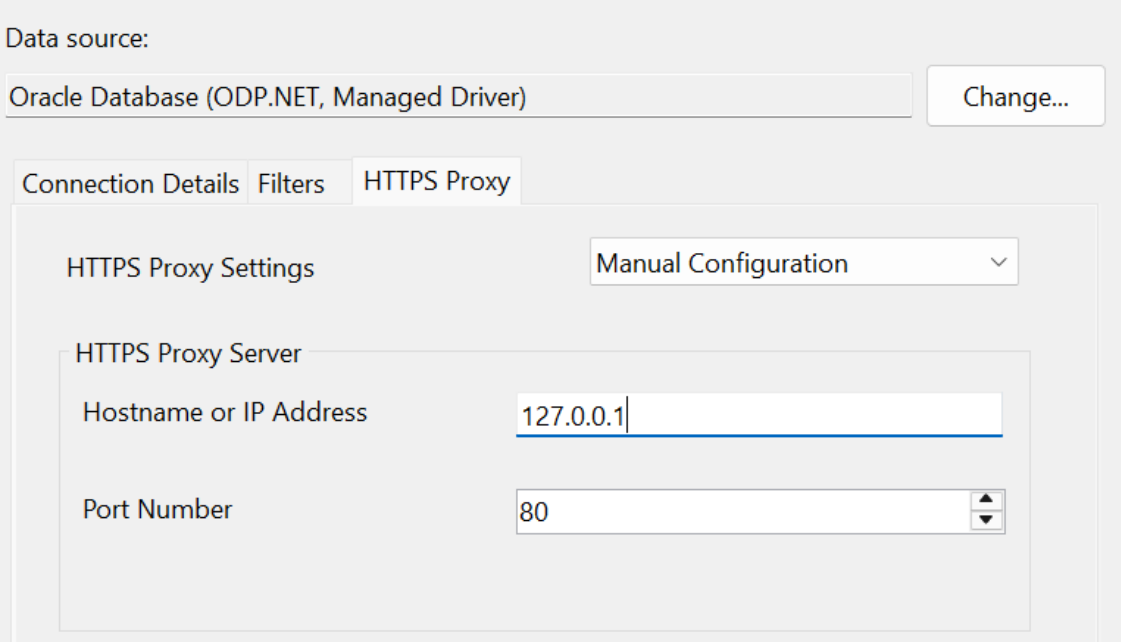
About the HTTPS Proxy Tab

Use the HTTPS Proxy tab to to connect to an Oracle Database when the machine that hosts Visual Studio is behind a firewall or equivalent. This is commonly used to connect to Oracle Autonomous Database from behind a firewall.

Opening the HTTPS Proxy Tab

To open the HTTPS Proxy tab, from the Connections node or the Connection nodes, right-click and select Add Connection or Modify Connection. Then select the HTTPS Proxy tab.

The HTTPS Proxy tab appears similar to the following:



Using the HTTPS Proxy Tab

The controls in the HTTPS Proxy tab are as follows:

Control	Description
HTTPS Proxy Settings	Select **None** for no proxy. Select **Auto-Detect** to use proxy settings configured on the machine that hosts Visual Studio. Select **Manual Configuration** to specify the proxy settings yourself.
Hostname or IP Address	Enter the hostname or IP address of the proxy. This field is only enabled if HTTPS Proxy Settings is set to **Manual Configuration**.
Port Number	Enter the port number of the proxy. This field is only enabled if HTTPS Proxy Settings is set to **Manual Configuration**.

Connection Configuration Options Page

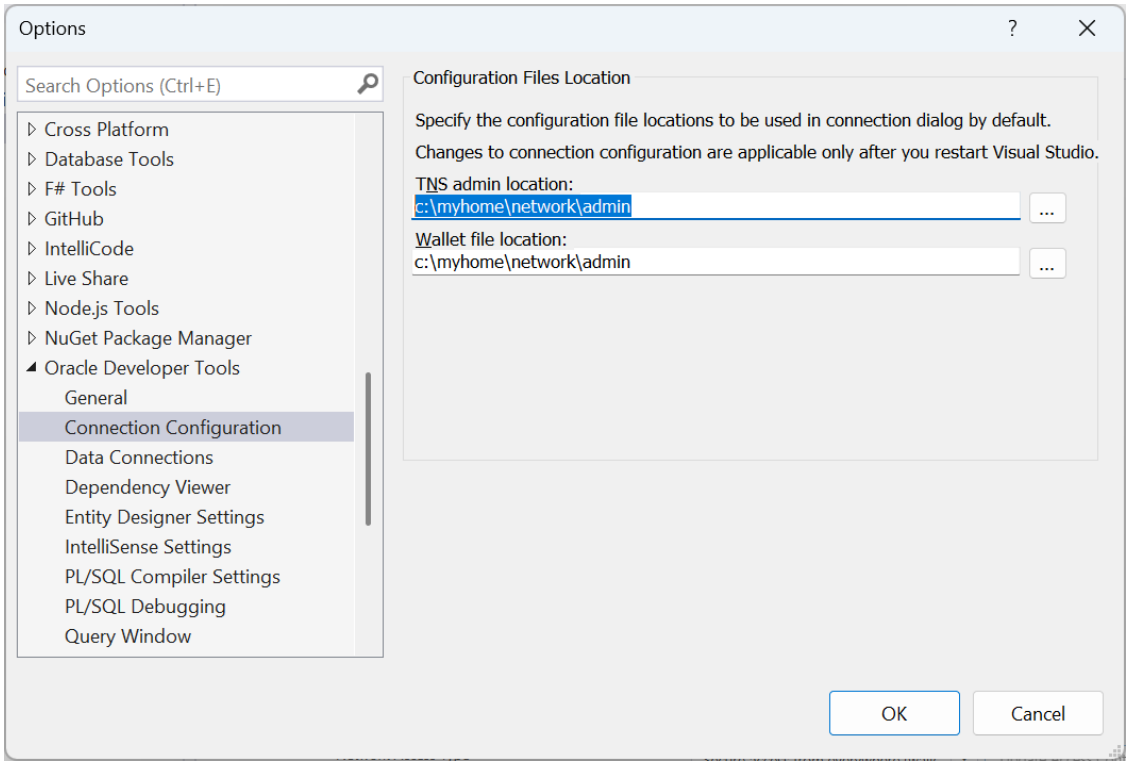
The Connection Configuration Options Page contains settings used by the Connection Dialog Box .

Accessing the Connection Configuration Options Page

To access the Connection Configuration Options Page, select **Options...** from the Tools menu. From the Options menu, select **Oracle Developer Tools**. Then select **Connection Configuration**.

Using the Connection Configuration Options Page

The Connection Configuration Options Page appears similar to the following:

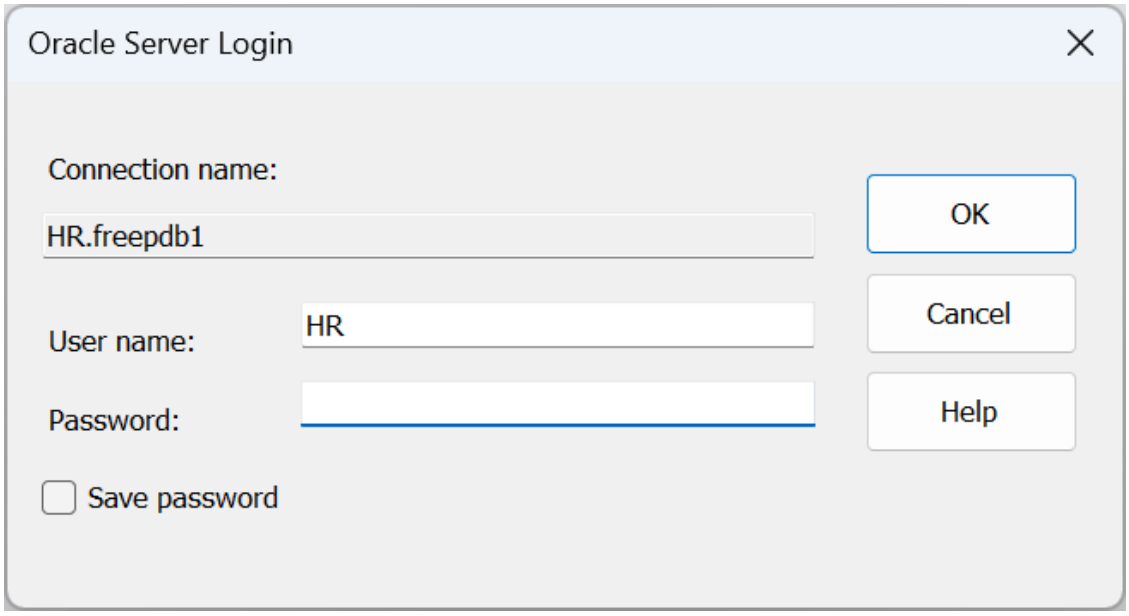


The controls of the Connection Configuration Options are as follows:

Control	Description
TNS Admin Location	This directory is the default directory used in the TNS Admin Location field in the Connection Dialog Box .
Wallet File Location	This directory is the default directory used in the Wallet File Location field in the Connection Dialog Box .

Oracle Server Login Dialog

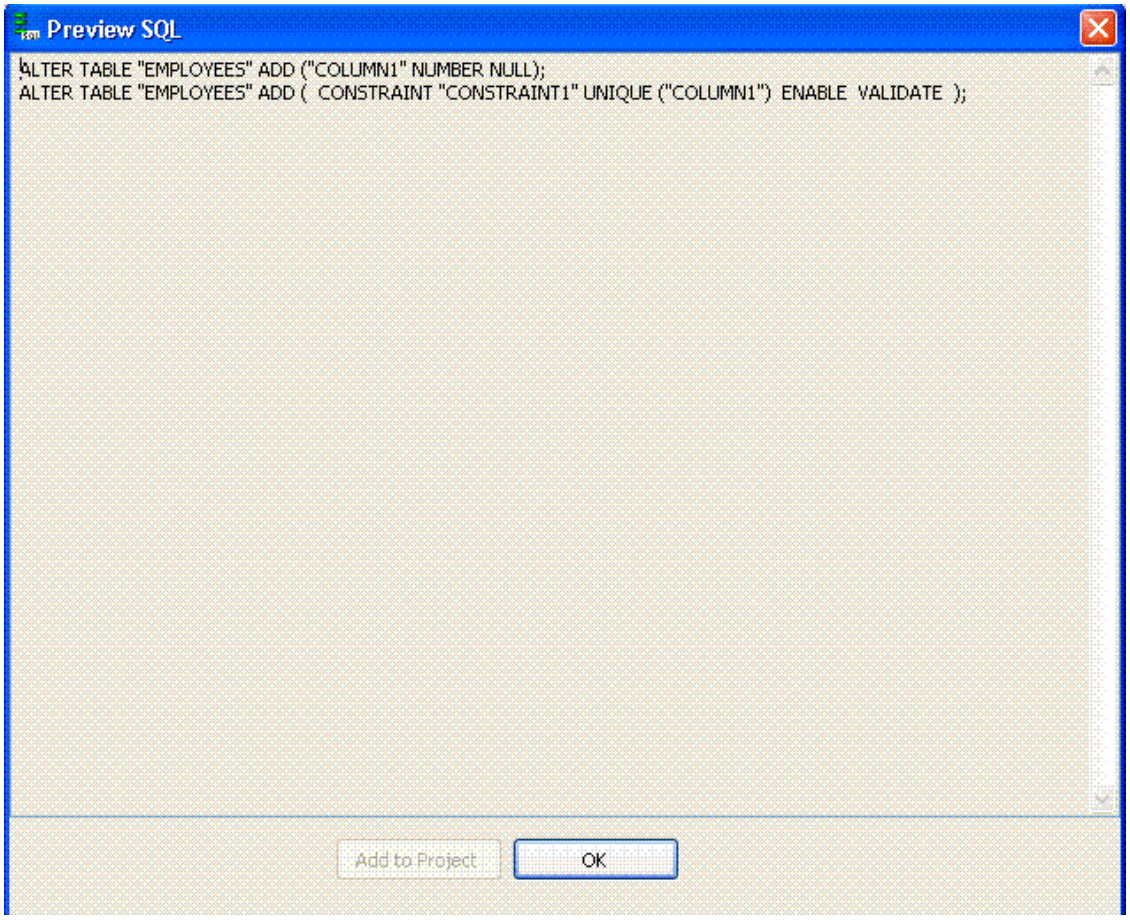
Use the Oracle Server Login dialog to specify the username or password, or both that opens an existing Oracle Developer Tools data connection in Server Explorer.



Control	Description
Connection Name	Displays the name of the connection that is to be opened. This is read-only.
User name	Specify the username to be used for the connection.
Password	Specify the password for the connection.
Save Password	Specify if the password should be saved with the connection information.
OK	Opens the Server Explorer database connection using the specified username/password.
Cancel	Cancels this dialog and does not open the connection.
Help	Launches the help page for this dialog.

Preview SQL Dialog

Most of these wizards and designers contain a Preview SQL button that displays the Preview SQL Dialog. The Preview SQL Dialog shows the SQL DML statements for the pending uncommitted changes in the designer or wizard from where it was launched. You cannot edit the SQL DML statements in this dialog.



Using Preview SQL

The Preview SQL dialog has the following controls:

Control	Description
Add to Project	Copies the displayed SQL DML statements to a `.sql` file and adds the `.sql` file to the open Oracle Database Project. If there are multiple open projects, you will be prompted to select the project to add the script to. See Managing Oracle Script Files.
OK	Closes the dialog.

See Also

Using Preview SQL Dialog Box to Generate Scripts to Projects

Table Designer

The Table Designer lets you create and modify relational, XML, or object tables. It also lets you create, modify, and drop relational table columns, constraints, and indexes.

This section covers the following topics:

Creating Tables in Oracle Developer Tools

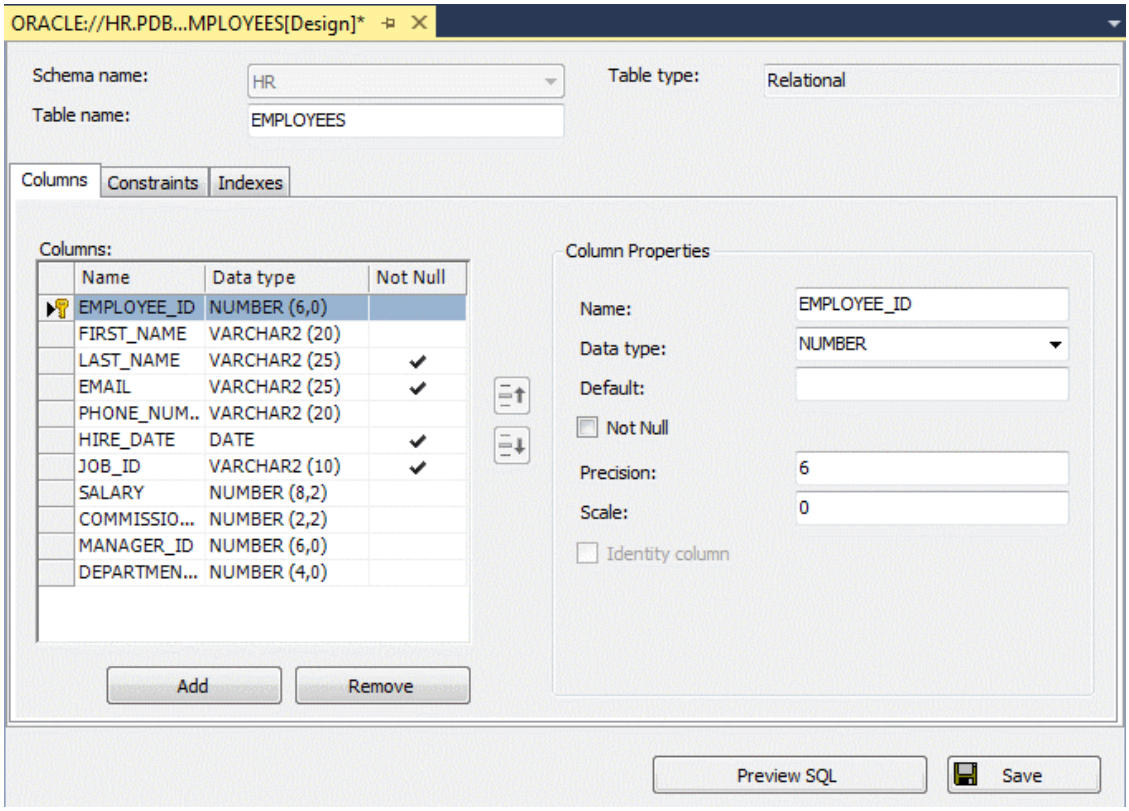
You use the Table Designer to create relational, XML, or object tables. The Oracle Output Window displays the SQL statements that the Table Designer executes, as well as any errors that occurred while saving data. You can view or edit a table's data by using Oracle Data Window.

Starting the Table Designer

Start the Table Designer by using one of the following methods:

- If you are creating a new table, in Server Explorer, right-click the Tables node and from the menu, select **New Relational Table**, **New XML Table**, or **New Object Table**.
- If you are modifying an existing table or one of its columns, constraints, or indexes, in Server Explorer, right-click the node for that table under the Tables node, and from the menu, select **Design**.

The Table Designer appears similar to the following:



Using the Table Designer

The Table Designer has the following components:

- Main Window
- One of the following depending on table type, respectively Relational, XML, or Object. The Relational table is the default.
  - Columns Tab
  - XML Tab
  - Object Tab
- Constraints Tab
- Indexes Tab

In general, to use the Table Designer, make the modifications you need for the table. If you are adding a column, constraint, or index to the table, click **Add** and then make the necessary modifications in the Details pane. If you are dropping a table component, select the component and click **Remove**.

Main Window

The main window for Table Designer is as follows:

Control	Description
Schema name	Select from the list of available schemas in which to create the table.
Table Name	Enter the name of the table you want to create. If you are replacing an existing table, this box displays the table's name.
Table type	Read-only. Type of the table: Relational, Object or XML. The type of table determines the tabs displayed in the Table Designer: • Relational tables: Columns tab, Constraints tab, Indexes tab, and Storage tab • XML tables: XML tab and Storage tab • Object Tables: Object tab, Constraints tab, Indexes tab, and Storage tab
Preview SQL	Displays the CREATE TABLE SQL command code in a pop-up dialog box and in the output window. Read-only.
Save	Saves any changes made to the table to the database. Alternatively, you can click the **Save** button in the Visual Studio toolbar or select **Save** from the **File** menu.

Columns Tab

The Columns tab, which only appears for relational tables, displays the following controls:

Control	Description
Columns	Lists the available columns and their data types with precision and whether or not they are null. The Column Properties pane displays detailed information about the currently selected column. If the name of a column or data type is too long to appear, you can hold the mouse over it to read the full name. A small key icon is displayed next to the primary key column. Click the up and down arrows to change the order of the columns.
Add	Adds a new column to the table and selects it. By default, a new column is named `COLUMN`*n* and its data type is `NUMBER`. To modify the new column, make the appropriate changes in the Column Details pane.
Remove	Drops the selected column from the table.
Properties	Displays the following details about the currently selected column: • **Name:** If you are creating a new column, enter its name here after you click the **Add** button. If you had selected an existing column from the Available Columns list box, its name displays here. To enter a case-sensitive name, enclose the name in double quotation marks. • **Not Null:** Specifies whether the column allows `NULL` values. • **Data type:** Specifies the data type of the column. Available choices are simple Oracle data types and user-defined types. The following components depend on the type of data type you select: • **Default:** Specifies the default value of the column. Specify this value as an expression. For example, to specify the default value of a `VARCHAR2` column as `"MANAGER"`, enter `'MANAGER'` (enclosed in single quotes) into this field. • **Precision:** For numeric types, specifies the precision of the column or the total number of digits. A zero value indicates the minimum precision allowed by the database. For interval types, specifies the precision in terms of fractional seconds, days, or years. • **Scale:** Specifies the column's scale, or the number of digits to the right of the decimal point. A zero value indicates the minimum scale allowed by the database. • **Length semantic:** Specifies the length semantic for the column. Available choices are: – **Default:** The database length semantic – **Byte:** Byte length semantic – **Char:** Character length semantic • **Size:** Specifies the data type size. • **Create as Ref:** Store a User Defined Type as a pointer (REF) to an Object Table rather than inline. • **Store as:** Specifies how to store an `XMLType` column. Available choices are: – Clob: Store XML data as a Clob – Object Relational: Store XML data as a Object Relational Type – Binary: Store XML data as binary data. (Available in Oracle Database 11.2.01 or later) • **Use XML Schema:** Specifies whether or not to use an XML schema. This is automatically checked (mandatory) for XML stored as an Object Relational type. For more information on registering XML Schemas with Oracle Database, see XML Schema Designer . • **XML Schema Owner:** Specifies the owner of the XML Schema.

Control	Description
	• **XML schema name:** Specifies the XML schema for the column, an XML schema to which the column must conform. Available choices are as follows: – **None** – **All XML schemas that you can access** If you specify an XML schema, you must specify an XML Root Element. • **XML root element:** Specifies the XML root element. • **Binary Storage Options:** Specifies the storage options for binary XML storage. Available in Oracle Database 11.2.0.1 or later. Available choices are: – None – Allow Any Schema: (Only available when Use XML Schema is not checked) – Allow Non-Schema: – Allow All: Only available when Use XML Schema is not checked) • **Identity Column:** Specifies that a column is an identity column. Only available in Oracle 12.1 database or later, for columns of a numeric type. This may only be modified on new columns or existing identity columns. There may only be one identity column for each table. • **Identity Column Properties:** This is a modifiable list of identity column properties. Available properties are as follows: • **Generation Type:** Specifies in which cases the identity column value is generated by database. Available choices are: – **ALWAYS:** The database will always generate the value. – **BY DEFAULT:** The database will generate a value when one is not provided. – **BY DEFAULT ON NULL:** The database will generate a value when NULL is provided. During initial column creation, all three options are available. For editing an existing identity column, the options are **ALWAYS** and either **DEFAULT** or **BY DEFAULT ON NULL**. **BY DEFAULT ON NULL** is only available if the column was initially created with that Generation Type. • **Minimum Value:** Specifies the minimum value that the database will generate for this identity column. • **Maximum Value:** Specifies the maximum value that the database will generate for this identity column. • **Interval:** Specifies the interval between the values that the database will generate for this identity column. • **Start With:** Specifies the starting value to be generated by the database for this identity column. For new columns, enter a value. For existing columns, enter a value or choose **Limit Value** from the drop down list. If **Limit Value** is chosen, it will lock the table and find the maximum sequence value found in the table (for increasing sequences) or find the minimum sequence value found in the table (for decreasing sequences). • **Allow Cycle:** Indicates whether or not the database cycles around (that is, starts over) and continues generating values when the minimum or maximum value is reached. • **Order Value:** Guarantees that values are generated in the order of request. This may be needed when using Real Application Clusters. • **Cache Values:** Specifies if values are preallocated by the database in memory for better performance. • **Cache Size:** Specifies how many values are preallocated in the cache.

Constraints Tab

Use the Constraints tab to edit the constraints of the table. Its controls are as follows:

Control	Description
Constraints	Lists the constraints on the table and their constraint types. The Constraint Details pane displays detailed information about the currently selected constraint. • **Add:** Adds a new constraint to the table and selects it. By default, a new constraint is named `CONSTRAINT`*n* and by default it is a unique constraint. To modify the new constraint, make the appropriate changes in the Constraint Details pane. • **Remove:** Removes the selected constraint.
Constraint Properties	Displays the following details about the currently select constraint: • **Name:** If you are creating a new constraint, enter its name here after you click the **Create** button. If you have selected an existing constraint from the Available Constraints list box, its name appears here. To enter a case-sensitive name, enclose the name in double quotation marks. • **Type:** Select from the following constraint types: – **Unique** – **Primary key** – **Check** – **Foreign key** – **UDT columns** – **Ref** • **Deferrable:** Specifies whether the constraint is deferrable, that is, whether the constraint's checking can be deferred until a transaction that the constraint affects has completed. If you deselect **Deferrable**, the constraint must be valid at the time the new or altered table is committed, or the table creation or modification will fail. • **Enabled:** Specifies whether the constraint is enabled or disabled. The default is enabled. • **Validate:** Specifies whether the constraint is validated. A validated constraint is guaranteed to hold for all rows of the table, even if the constraint is disabled. • **Execution**: Establishes the default checking behavior for constraints that are deferrable. It specifies either of the following: – **Immediate:** Oracle will check this constraint at the end of each subsequent SQL statement. If you do not specify Initially at all, the default setting is Initially Immediate. – **Deferred:** Oracle will check this constraint at the end of subsequent transactions.

Control	Description
	The following components depend on the type of constraint you select: • **Primary Key Columns/Unique Key Columns**: Specifies the columns on which the primary key or unique key is based. Available choices are the table's columns. When adding a key, use the drop down list to choose the column. • **Add (Primary/Unique Key):** Adds a new column to the list of primary/unique keys. • **Remove (Primary/Unique Key):** Removes the selected column from the list of primary/unique keys. • **Using Index (Primary/Unique Key):** Specifies the index to use the constraint. Select **Auto** to have the database automatically generate an index. • **Table (Foreign Key)**: Specifies the table where the foreign key constraint is located. • **Constraint (Foreign Key)**: Specifies the foreign key constraint. • **Association (Foreign Key)**: Specifies the referenced column and the local column on which the key operates. When adding a new constraint, click on the local column to choose from the drop-down list. • **On Delete (Foreign Key):** Specifies how Oracle Database automatically maintains referential integrity if you remove a referenced primary or unique key value. Select from the following: – **No action:** Oracle does not allow you to delete referenced key values in the parent table that have dependent rows in the child table. – **Cascade:** Oracle removes dependent foreign key values. – **Set null**: Oracle converts dependent foreign key values to `NULL`. • **Condition (Check Constraint):** Enter the condition to be checked. For example: `JOB_ID IS NOT NULL`. • **Column/Attribute (Ref Constraint):** `REF` constraints let you describe the relationship between a column of type `REF` and the object it references. You can specify the name of a `REF` column of an object table or relational table or specify an embedded `REF` attribute within an object column of a relational table. • **Scope (Ref Constraint):** Select a table name that will be used to restrict the scope of references in the `REF` column. To specify this clause, `scope_table` must be in your own schema or you must have `SELECT` privileges on `scope_table` or `SELECT ANY TABLE` system privileges. You can specify only one scope table for each `REF` column. • **Scope/With Rowid (Ref Constraint):** Select this checkbox to store the rowid along with the `REF` value in the column or attribute specified in the `REF` constraint. Storing the rowid with the `REF` value can improve the performance of dereferencing operations, but will also use more space. Default storage of `REF` values is without rowids. • **OID Based on Primary Key (Primary Key):** For a primary key constraint on an object table, this selection will cause the primary key to be used as the object identifier.

Indexes Tab

Use the Indexes tab to create or modify the table's indexes. Its controls are as follows:

Control	Description
Indexes	Lists the indexes on the table, their index types, and affected columns. The index type can be Bitmap or B-tree. The affected columns are in a comma-delimited list. The Index Details pane displays detailed information about the currently selected index. • **Add:** Adds a new index to the table and selects it. By default, a new index is named `INDEX`*N* and by default it is a B-Tree index. To modify the new index, make the appropriate changes in Index Details. • **Remove:** Removes the selected index.
Index Properties	Displays the following details about the currently selected index: • **Name:** If you are creating a new index, enter its name here after you click the **Add** button. If you have selected an existing index from the Available Indexes list box, its name appears here. To enter a case-sensitive name, enclose the name in double quotation marks. • **Type:** Specifies the type of the index. Available choices are: – **B-Tree:** Create the index as a normal index. – **Bitmap:** Creates the index with a bitmap for each distinct key, rather than indexing each row separately. • **Unique:** Specifies the index as unique. Unique indexes guarantee that no two rows of a table have duplicate values in the key column (or columns). • **Reverse:** Stores the bytes of the index block in reverse order, excluding the rowid. • **Index keys:** Specifies the index keys, the columns or expressions to be indexed. You can also specify whether each key will be indexed in ascending or descending order. To index a column, select the column from the drop-down list in the **Key** column. Remember that the order of the columns is important. To index an expression, type the expression into the **Key** column. To specify the order of the index key, select **Ascending** or **Descending** from the corresponding drop-down list in the **Order** column. Indexes on character data are created in ascending or descending order of the character values in the database character set. • **Add:** Add a key to the list of Index Keys. • **Remove**: Remove the selected Index Key from the list.

XML Tab

Use the XML tab to specify the properties of an XML table. Its components are as follows:

Control	Description
Store As	Specifies how to store the XML. Available choices are: • **CLOB**: Store XML data as a Clob. • **Object Relational:** Store XML data as a Object Relational Type • **Binary**: Store XML data as binary data. (Available in Oracle Database 11.2.01 or later)
Use XML Schema:	Specifies whether or not to use an XML schema. This is automatically checked (mandatory) for XML stored as an Object Relational type. For more information on registering XML Schemas with Oracle Database, see XML Schema Designer .
XML Schema Owner:	Specifies the owner of the XML Schema.

Control	Description
XML schema name	Specifies the XML schema for the column, an XML schema to which the column must conform. Available choices are as follows: • **None** • **All XML schemas that you can access** If you specify an XML schema, you must specify an XML Root Element.
XML root element	Specifies the XML root element.
Binary Storage Options:	Specifies the storage options for binary XML storage. Available in Oracle Database 11.2.0.1 or later. Available choices are: • **None** • **Allow Any Schema:** (Only available when Use XML Schema is not checked) • **Allow Non-Schema:** • **Allow All:** (Only available when Use XML Schema is not checked)

Object Tab

Use the object tab to specify the properties of an object table. Its components are as follows:

Control	Description
Object Type	Lists the object types that you can select.
Object Substitutable At All Levels	Enables substitution at all levels. This means that row objects corresponding to subtypes can be inserted into this object table. Substitution is disabled for all embedded nested tables and arrays.

Import Table Wizard

The Import Table Wizard lets you import table definitions and table data from an external data source to Oracle database.

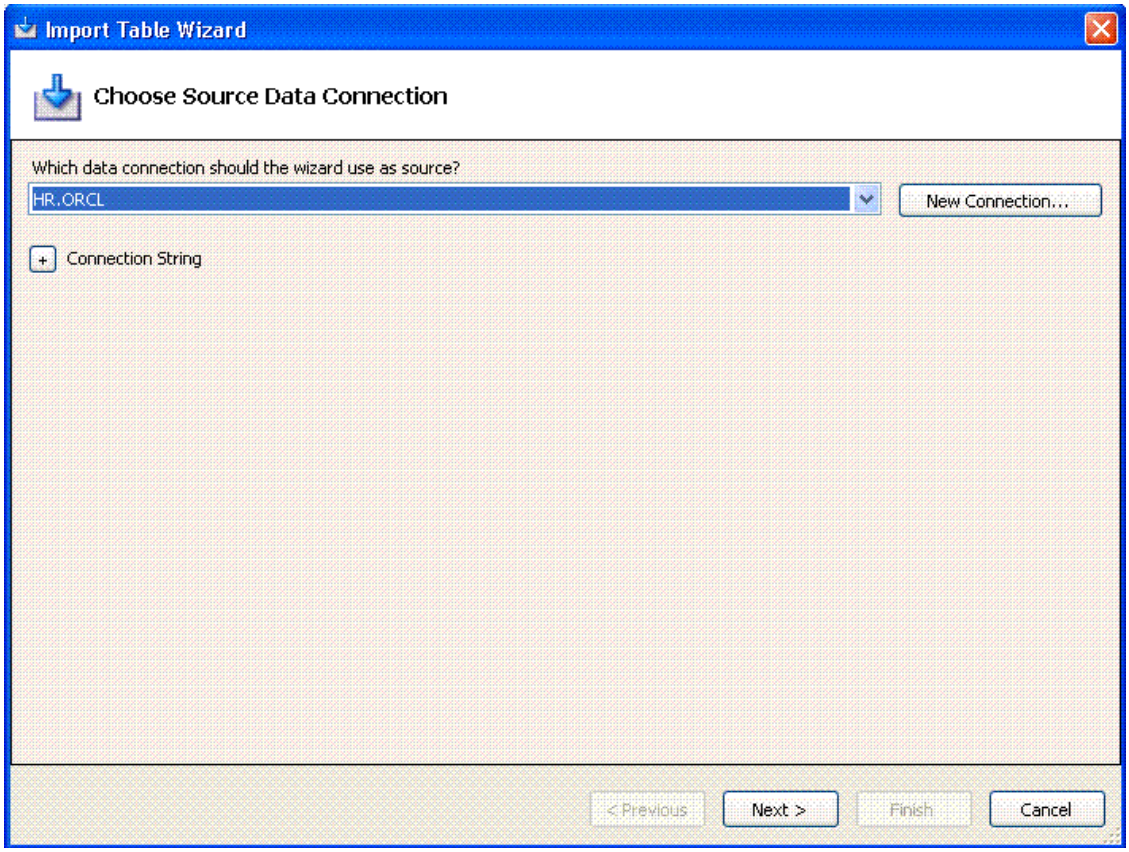
This section covers the following topics:

- Starting the Import Table Wizard
- Using the Import Table Wizard

Starting the Import Table Wizard

To start the Import Table Wizard, from an Oracle Developer Tools connection in Server Explorer, right-click on the connection node or Tables node, and from the menu, select **Import Table**.

The Import Table Wizard's opening screen appears as follows:



Using the Import Table Wizard

The Import Table wizard allows you to do the following:

- Select the existing external data source or create a new external data source in Server Explorer.
- Select one or more tables to import from the external data source.
- Select the columns to import for the specified table.
- View or modify the table definitions of the selected tables in the Oracle database. Only the following table properties are used to create the table in the Oracle database:
  - All columns in the table
  - Table name
  - Column name
  - Column Type
  - Primary Key
  - Allow Null (indicates whether or not the column can be nullable)
- Create the selected tables and import the table data from the external data source to the Oracle database.

You can import table and data using this wizard only from the following Server Explorer data sources.

Data Source	Data Provider
Oracle Database Server	Oracle Data Provider for .NET
Microsoft SQL Server	.NET Framework Data Provider for SQL Server
Microsoft Access Database File	.NET Framework Data Provider for OLE DB
Microsoft Excel File	.NET Framework Data Provider for OLE DB

For information on how to connect to the Microsoft SQL Server, Access Database file, and Excel file, please see Microsoft documentation.

The windows in the Import Table wizard are as follows.

Window	Description
Choose Your Source Data Connection	Specify the data connection for the external data source.
Select Tables to Import	Select the tables and table columns to import to Oracle database. Use Ignore Errors when Importing Table Data to continue importing the next row of data into a table regardless of error. Check the Import Data column to import table data.
Customize Column Settings	For each selected table, customize the name and data type properties for table columns to import to the Oracle database.
Progress Report	Display results after creating tables and importing table data. Click **OK** to exit the wizard.

Trigger Designer

This section covers the following topics:

- Creating Triggers in Oracle Developer Tools
- Starting the Trigger Designer
- Using the Trigger Designer

Creating Triggers in Oracle Developer Tools

Use the Trigger Designer to create triggers for tables or views. After you create a new trigger, you can edit the trigger specification using the PL/SQL Code Editor. When you create a trigger, Oracle Database automatically enables it.

You cannot use the Trigger Designer to create the following types of triggers:

- Procedure calls as triggers
- Nested column triggers if the new version of the trigger refers to the same view as the old version of the trigger
- Triggers that use the `:OLD`, `:NEW`, or `:PARENT` reserved words for row values
- The body of a trigger
- DDL triggers

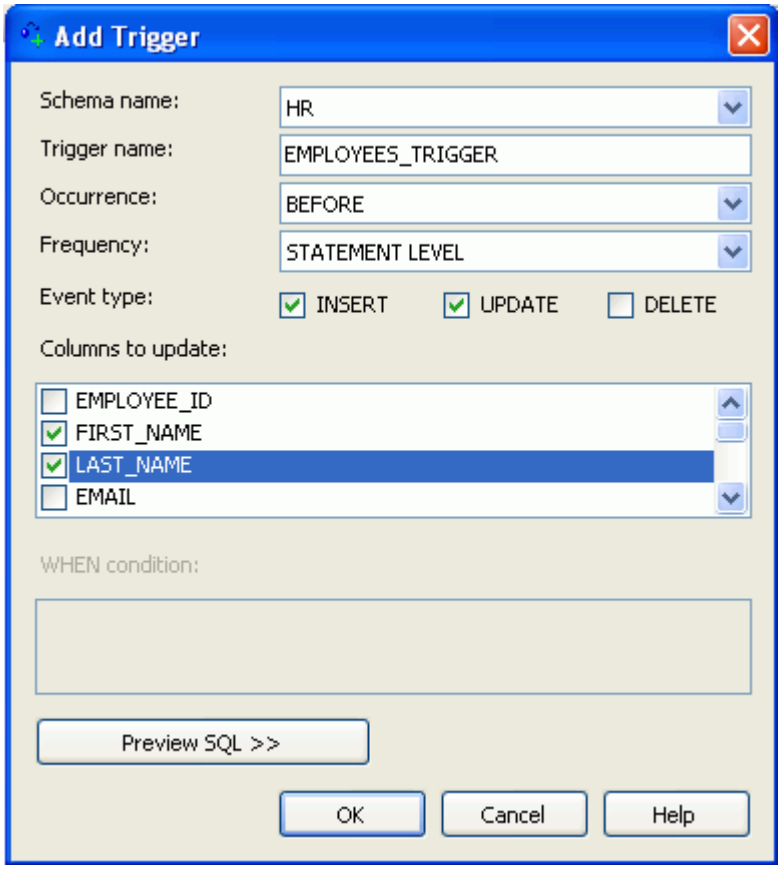
However, after you create the trigger, you can modify it in the PL/SQL Code Editor to use these features.

Starting the Trigger Designer

To start the Trigger Designer:

- If you are creating the trigger on a table, in Server Explorer, open the node for the table. The table node lists nodes for the table's constraints, indexes, and triggers. Right-click the Triggers node and from the menu, select **Add Trigger**.
- If you are creating the trigger on a view, in Server Explorer, select and open the view's node. Right-click the Triggers node and from the menu, select **Add Trigger**.

The Trigger Designer appears similar to the following:



Using the Trigger Designer

In general, use the Trigger Designer to create a new trigger. If you want to create the trigger with functionality not provided in the Trigger Designer, such as adding the `OR REPLACE` keyword, or if you want to edit the trigger later, you can modify the trigger in the PL/SQL Code Editor.

The Trigger Designer has the following controls:

Control	Description
Schema name	Select from the list of available schemas in which to create the trigger.

Control	Description
Trigger name	Enter a name for the trigger. The default name is based on the table for which the trigger was created.
Occurrence	Select from the following: • **BEFORE:** Fires the trigger before executing the triggering event. For row triggers, it fires the trigger before each affected row is changed. • **AFTER:** Fires the trigger after executing the triggering event. For row triggers, it fires the trigger after each affected row is changed.
Frequency	Select from the following: • **ROW LEVEL:** Specifies the trigger as a row trigger. Oracle Database fires a row trigger once for each row that is affected by the triggering statement and meets the optional trigger constraint defined in the `WHEN` condition setting. • **STATEMENT LEVEL:** Specifies the trigger as a statement trigger. Oracle Database fires a statement trigger only once when the triggering statement is issued if the optional trigger constraint is met.
Event type	Check one or more of the following: • **INSERT:** Fires the trigger whenever an `INSERT` statement adds a row to a table or adds an element to a nested table. • **UPDATE:** Fires the trigger whenever an `UPDATE` statement changes a value in one of the columns specified in the Select columns to apply to trigger list. If you do not select a column, the database fires the trigger whenever an `UPDATE` statement changes a value in any column of the table or nested table. If you checked `UPDATE`, the Columns to Update control is enabled. Note: You can specify object type, varray, and `REF` columns in the Select columns to apply to trigger list to fire the trigger whenever an `UPDATE` statement changes a value in one of the columns. However, you cannot change the values of these columns in the body of the trigger itself. • **DELETE:** Fires the trigger whenever a **DELETE** statement removes a row from the table or removes an element from a nested table
Columns to update	Applies only to update triggers. Select from the columns to which the update trigger will apply.
WHEN condition	Enter a SQL condition that must be satisfied in order for the trigger to fire. This condition must contain correlation names and cannot contain a query.
Preview SQL	Displays the `CREATE TRIGGER` SQL command code in a pop-up dialog box and in the output window. Read-only.
OK	Saves your work and displays the trigger code in the PL/SQL Code Editor so that you can customize it. To commit the trigger, click the **Save** button in the Visual Studio .NET toolbar or select Save from the File menu. After you successfully create the trigger, Oracle Developer Tools displays its node in Server Explorer. If you have created the trigger with errors, Oracle Developer Tools displays an error dialog box, and then displays the error messages in the Output window. Click **OK** and correct the error in the dialog box. Afterwards, when you click the Trigger Designer's **OK** button to save your changes, click **Yes** to replace the incorrect trigger with the corrected version. Note: If a trigger produces compilation errors, it is still created, but it fails on execution. This means it effectively blocks all triggering DML statements until it is disabled, replaced by a version without compilation errors, or dropped. You can see the associated compiler error messages in the Oracle Output pane in the Output window.

View Designer

The View Designer lets you create and modify views. This section covers the following topics:

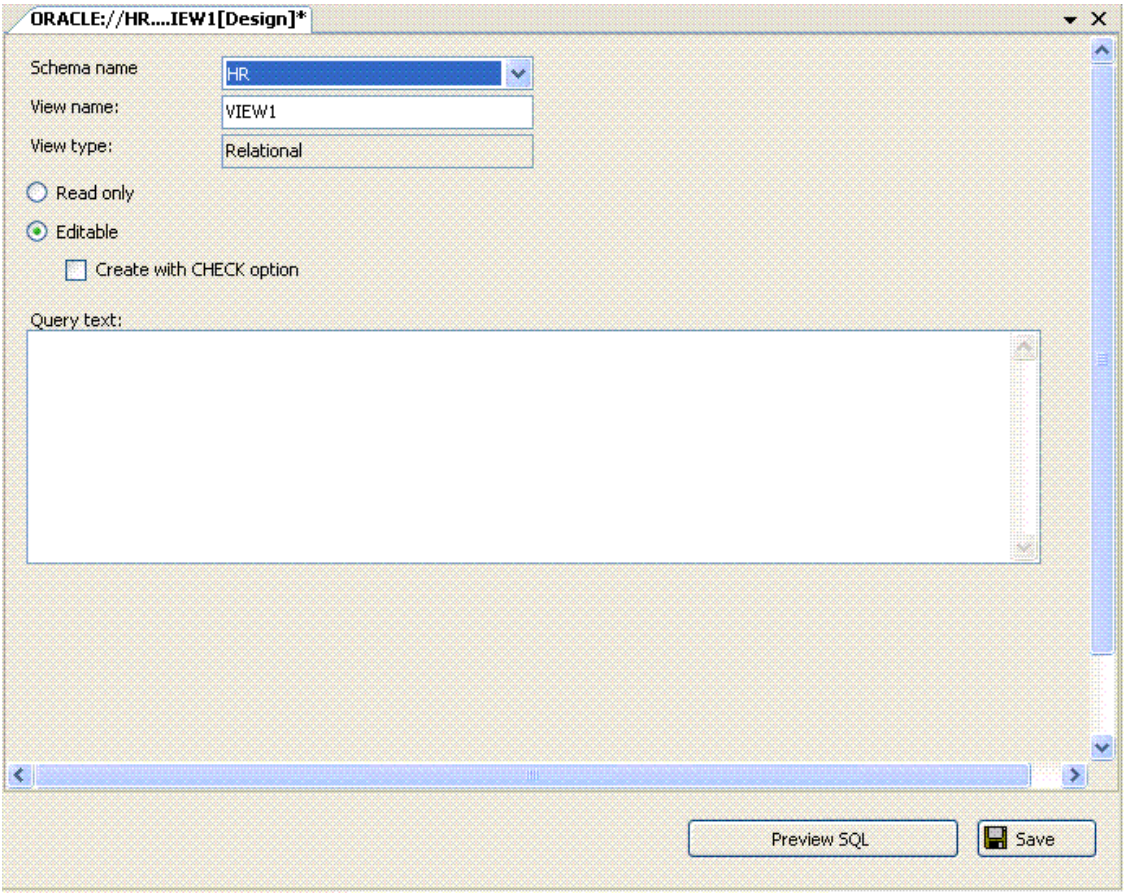
Creating Views in Oracle Developer Tools

Use the View Designer to create a new view or replace an existing view. The Oracle Output Window displays the SQL statements that the View Designer executes, as well as any errors that occurred while saving. After you create a view, you can view or edit a view's data by using Oracle Data Window or you can add a trigger to it using the View node menu.

Starting the View Designer

- If you are creating a new view, right-click the Views node and from the menu, select **New Relational View**, **New XML View**, or **New Object View**.
- If you are modifying an existing view, expand the Views node, right-click the node for the view you want, and from the menu, select **Design**.

The View Designer appears similar to the following:



Using the View Designer

The View Designer creates new views and replaces existing views.

The controls in the View Designer are as follows:

Control	Description
Schema name	Select from the list of available schemas in which to create the view.
View name	Enter the name of the view you want to create. If you are replacing an existing view, this box displays the view name. To enter a case-sensitive name, enclose the name in double quotation marks.
View type	Displays one of the following: • **Relational:** The view is a relational view • **XML:** The view is an XML view • **Object**: The view is an object view
Read only	Ensures that the view cannot be updated. The option applies only to relational views
Editable	Allows the view to be edited. Select the **Create with CHECK option** check box to prohibit any changes to the table or view that would produce rows that are not included in the subquery. If you use the view in subquery of a DML statement, you can specify this clause in a subquery in the `FROM` clause but not in subquery in the `WHERE` clause. The option applies only to relational views.

Control	Description
Base Type	For Object Views only: Specifies the UDT that this object view is based on.
Object Identifier	For Object views: Specifies the object identifier for this view. The default is no text. Fox XML views: Specifies the attributes that will to identify each row. In most cases, these attributes correspond to the primary key columns of the base table. You must ensure that the attribute list is unique and identifies exactly one row in the view. This option does not apply to relational views.
Query text	Enter the SELECT statement to create the view. You do not need to end the statement with a semi-colon.
Super View	For Object views only: Specifies a super view. The list displays all object views that are accessible to the current logged on user according to the filters.
XML schema name	Specifies the XML schema for the column, an XML schema to which the column must conform. Available choices are as follows: • None • All XML schemas that you can access If you specify an XML schema, you must specify an XML Root Element. The option applies only to XML views.
XML root element	For XML views only: Specifies the XML root element for the view.
Preview SQL	Displays the CREATE VIEW SQL command code in a pop-up dialog box and in the output window. Read-only.
Save	Commits the view to the database. Alternatively, you can click the **Save** button in the Visual Studio .NET toolbar or select **Save** from the **File** menu.

Creating Triggers on Existing Views

After you create a view, you can create a trigger on it:

1. In Server Explorer, expand the Views node, and then select the view to which you want to add a trigger. The view will have a Triggers node.
2. Right-click the view's node and from the menu, select **Add Trigger**.
3. Follow the steps in the Trigger Designer to create the trigger.

See Also

Views Node | View Column Node | View Triggers Node | Run Dialog Box | Oracle Query Window

Function Designer

The Function Designer lets you create new functions. This section covers the following topics:

- Creating Functions in Oracle Developer Tools
- Starting the Function Designer
- Using the Function Designer

Creating Functions in Oracle Developer Tools

Use the Function Designer to create the template code for a new standalone function. After you create the function template, the PL/SQL Code Editor opens with the function code displayed so that you can customize the function as needed.

If you want to edit an existing function, in Server Explorer, double-click the node that represents the function or right-click the node and click **Edit**. Then edit the function code in the PL/SQL Code Editor.

If you want to create a function and add it to a package, consider using the Package Designer to create the function.

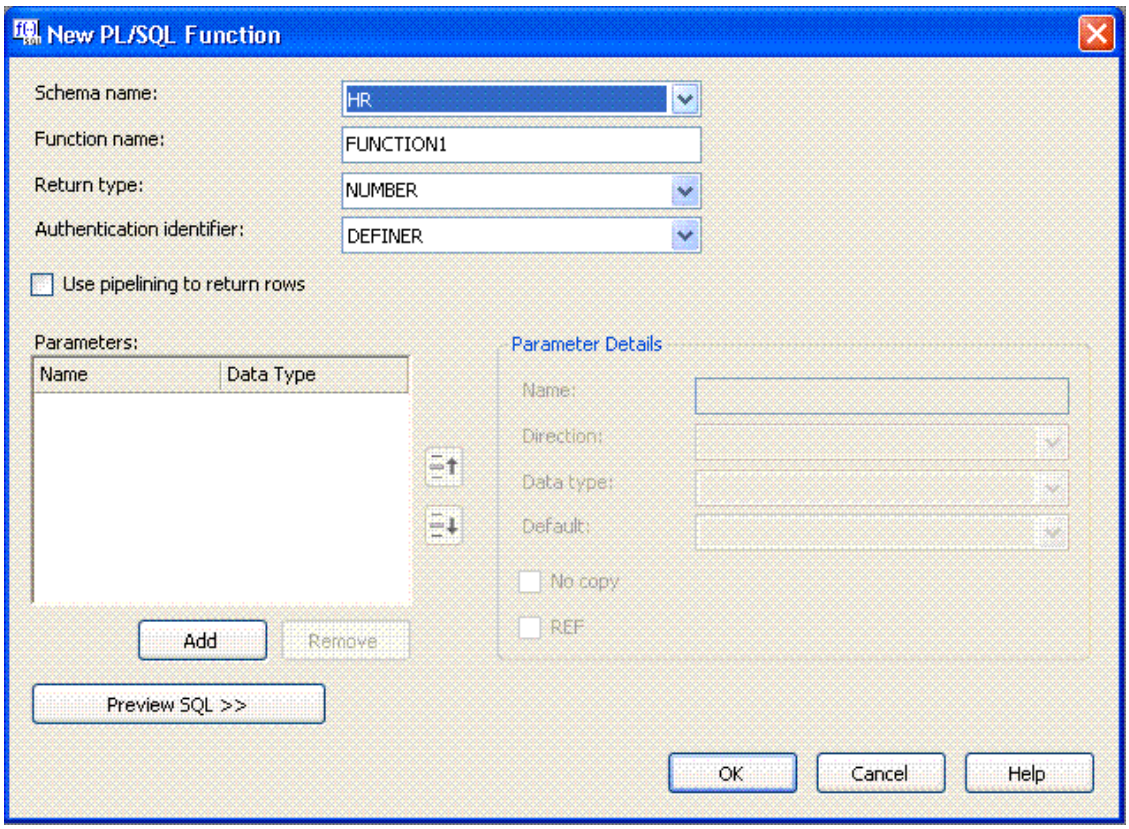
Once you have completed the function, you can compile and run it by right-clicking its function node in Server Explorer and choosing from the menu that appears. To delete the function, right-click its node and from the menu, select **Delete**.

Note: If you want to create a SQLJ, C, or Java call specification, or a pragma clause, use the PL/SQL Code Editor.

Starting the Function Designer

In Server Explorer, right-click the Functions node and from the menu, select **New PL/SQL Function**.

The Function Designer appears similar to the following:



Using the Function Designer

In general, use the Function Designer to create a template for the function. The template contains parameters for the base definition of the function, but not more elaborate code such as statements to add within the template's `BEGIN` and `END` statements. After you create this template, you can customize it in the PL/SQL Code Editor.

The controls in the Function Designer are as follows:

Control	Description
Schema name	Select from the list of available schemas in which to create the function.
Function name	Enter a name for the function.
Return type	Select from the choices of data types listed. If the data type is not listed, you can type it in the **Return type** box.
Authentication identifier	Select from the following choices: • **CURRENT_USER:** Specifies that the function executes with the privileges of the current user. • **DEFINER:** Specifies that the function executes with the privileges of the owner of the schema in which the function resides, and that external names resolve in the schema where the function resides. This is the default setting.
Use pipelining to return rows	Returns the results of a table function iteratively.
Parameters	Lists the parameters and their data types used in this function. To create a new parameter, click **Add**, and then use the Parameter Details pane to modify the parameter as needed. As you create parameters, their PL/SQL code appears in the SQL Preview box. To modify the order of parameters, select the parameter to move and click the **Up** or **Down** arrow. To remove a parameter, select it and click **Remove**.
Parameters detail	Displays detailed information about the selected parameter: • **Name:** Enter a name for the parameter. • **Direction:** Select from the following: – **IN:** Indicates that you must supply a value for the argument when calling the function. IN parameters can have a default value. – **OUT:** Indicates that the function passes a value for this argument back to its calling environment after execution. – **IN-OUT:** Indicates that you must supply a value for the argument when calling the function and that the function passes a value back to its calling environment after execution. • **Data type:** Select from the list. If the data type is not listed, you can type it here. • **Default:** Specify a default value to use if the parameter is not passed. • **No copy:** Instructs the database to pass this argument as fast as possible. This clause can significantly enhance performance when passing a large value like a record, an index-by table, or a varray to an `OUT` or `IN OUT` parameter. `IN` parameter values are always passed using the No copy option. • **REF:** Makes parameter data type objects into reference types.
Preview SQL	Displays the `CREATE FUNCTION` SQL command code in a pop-up dialog box and in the output window. Read-only.

Control	Description
OK	Saves your work, creates the function with the template code in the database, and displays the function template code in the PL/SQL Code Editor so that you can customize it. To commit the changes to the new function template, click the **Save** button in the Visual Studio .NET toolbar or select **Save** from the **File** menu. If you have created the function with errors, Oracle Developer Tools displays an error dialog box, and then displays the error messages in the Output window. Click **OK** and correct the error in the Function Designer. Afterwards, when you click the designer's **OK** button to save your changes, click **Yes** to replace the incorrect function with the corrected version. After you successfully create the function, Oracle Developer Tools displays its node in Server Explorer.

See Also

Functions Node | Run Dialog Box | Oracle Query Window

Procedure Designer

The Procedure Designer lets you create new procedures. This section covers the following topics:

- Creating Procedures in Oracle Developer Tools
- Starting the Procedure Designer
- Using the Procedure Designer

Creating Procedures in Oracle Developer Tools

Use the Procedure Designer to create the template code for a new standalone procedure. After you create the procedure template, the PL/SQL Code Editor opens with the function code displayed so that you can customize the procedure as needed.

If you want to edit an existing procedure, in Server Explorer, double-click the node that represents the procedure or right-click the node and click **Edit**. Then edit the procedure code in the PL/SQL Code Editor.

If you want to create a procedure and add it to a package, consider using the Package Designer to create the procedure.

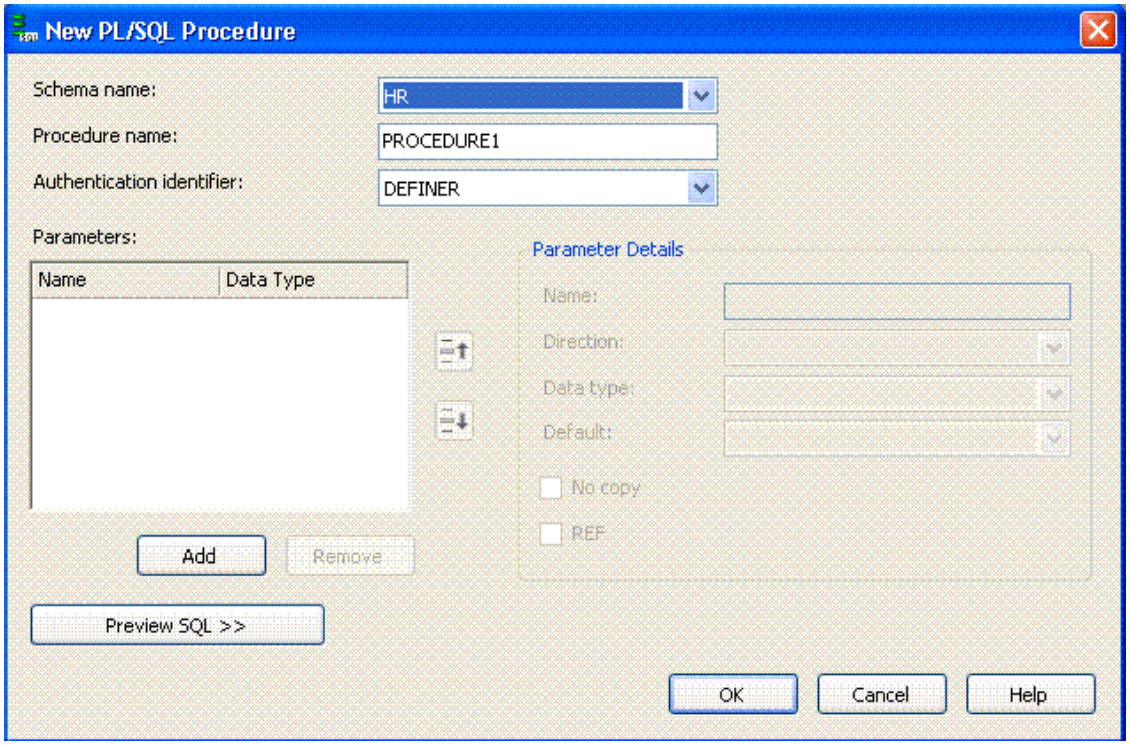
Once you have completed the procedure, you can compile and run it by right-clicking its procedure node in Server Explorer and choosing from the menu that appears. To delete the procedure, right-click its node and from the menu, select **Delete**. For more information, see Procedure Nodes.

Note: If you want to create a SQLJ, C, or Java call specification, or a pragma clause, use the PL/SQL Code Editor.

Starting the Procedure Designer

In Server Explorer, right-click the Procedures node and from the menu, select **New PL/SQL Procedure**.

The Procedure Designer appears as follows:



Using the Procedure Designer

In general, use the Procedure Designer to create a template for the procedure. The template contains parameters for the base definition of the procedure, but not more elaborate code such as statements to add within the template's `BEGIN` and `END` statements. After you create this template, you can customize it in the PL/SQL Code Editor.

The controls in the Procedure Designer are as follows:

Control	Description
Schema name	Select from the list of available schemas in which to create the procedure.
Procedure name	Enter a name for the procedure.
Authentication identifier	Select from the following choices: • **CURRENT_USER:** Specifies that the procedure execute with the privileges of the current user. • **DEFINER:** Specifies that the procedure execute with the privileges of the owner of the schema in which the procedure resides, and that external names resolve in the schema where the procedure resides. This is the default setting.
Parameters	Lists the parameters and their data types used in this procedure. To create a new parameter, click **Add**, and then use the Parameter Details pane to modify the parameter as needed. As you create parameters, their PL/SQL code appears in the SQL Preview box. To modify the order of parameters, select the parameter to move and click the **Up** or **Down** arrow. To remove a parameter, select it and click **Remove**.

Control	Description
Parameters detail	Displays detailed information about the selected parameter: • **Name:** Enter a name for the parameter. • **Direction:** Select from the following: – **IN:** Indicates that you must supply a value for the argument when calling the procedure. `IN` parameters can have a default value. – **OUT:** Indicates that the procedure passes a value for this argument back to its calling environment after execution. – **IN-OUT:** Indicates that you must supply a value for the argument when calling the procedure and that the procedure passes a value back to its calling environment after execution. • **Data type:** Select from the list. If the data type is not listed, you can type it here. • **Default:** Specify a default value to use if the parameter is not passed. • **No copy:** Instructs the database to pass this argument as fast as possible. This clause can significantly enhance performance when passing a large value like a record, an index-by table, or a varray to an `OUT` or `IN OUT` parameter. `IN` parameter values are always passed using the **No copy option**. • **REF:** Makes parameter data type objects into reference types.
Preview SQL	Displays the `CREATE PROCEDURE` SQL command code in a pop-up dialog box and in the output window. Read-only.
OK	Saves your work, creates the procedure with the template code in the database, and displays the procedure template code in the PL/SQL Code Editor so that you can customize it. To commit the changes to the new procedure template, click the **Save** button in the Visual Studio .NET toolbar or select **Save** from the **File** menu. If you have created the procedure with errors, Oracle Developer Tools displays an error dialog box, and then displays the error messages in the Output window. Click **OK** and correct the error in the Procedure Designer. Afterwards, when you click the designer's **OK** button to save your changes, click **Yes** to replace the incorrect function with the corrected version. After you successfully create the procedure, Oracle Developer Tools displays its node in Server Explorer.

See Also

Procedures Node | Stored Procedure Run Dialog Box | Oracle Query Window

Stored Procedure Run Dialog Box

The Run dialog box lets you execute standalone procedures and functions or package procedures or functions. In the case where a stored procedure or function returns a `REF CURSOR`, you can generate the `REF CURSOR` metadata information into the `app.config` or `web.config` configuration file of your application. This metadata information is then used when using the Add Import Function Dialog in Entity Designer or when calling the stored procedure or function from code. For more information on using the Add Import Function Dialog in Entity Designer see Using Add Import Function Dialog to Import an Oracle Stored Procedure. For more information about `REF CURSOR` metadata information, see the Implicit REF CURSOR Binding Support section in *Oracle Data Provider for .NET Developer's Guide*.

This section covers the following topics:

- Running Procedures and Functions in Oracle Developer Tools

Running Procedures and Functions in Oracle Developer Tools

Use the Run dialog box to run compiled or saved standalone procedures and functions or package procedures or functions. To compile a function, stored procedure, package function, or package procedure, right-click its node in Server Explorer and select **Compile** from the menu.

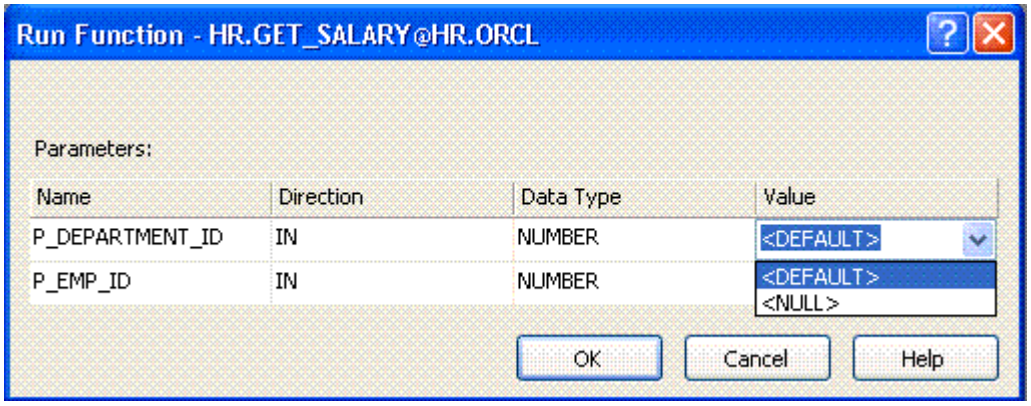
Starting the Run Dialog Box

To start the Run Dialog box:

1. In Server Explorer, locate the node that represents the procedure or function that you want to run.
2. Right-click the node and from the menu, select **Run**.

If the stored procedure or function contains input parameters, the Run Dialog Box will appear.

The Run dialog box appears similar to the following:



Using the Run Dialog Box

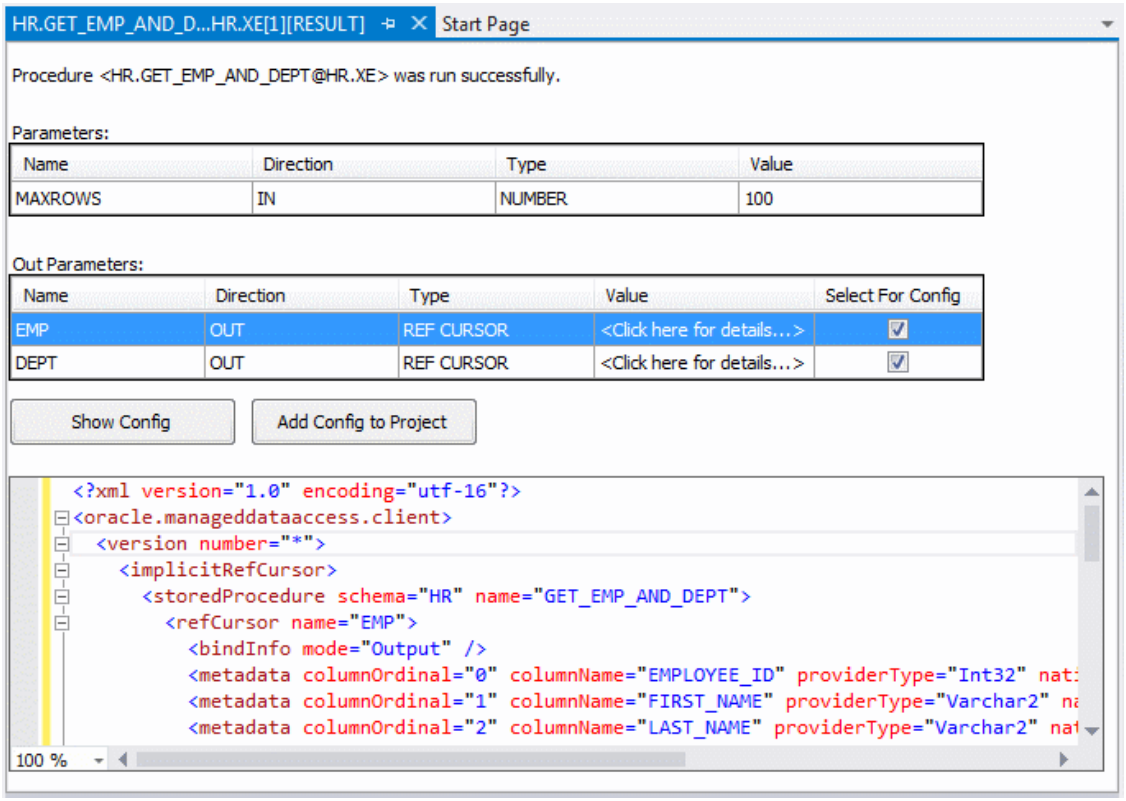
If the procedure or function contains an `IN` or `IN-OUT` parameter, you can enter a value and click `OK`. Otherwise, the procedure or function executes automatically.

You can type in a value or specify a default or null value for a parameter:

- To select a default value for a parameter, select `<DEFAULT>` from the parameter **Value** drop-down list.
- To specify null value for a parameter, select `<NULL>` from the parameter **Value** drop-down list.

Viewing the Results from Running a Procedure or Function

Whenever you run a procedure or function, its results display in a new window. The results window appears similar to the following:



Using the Run Dialog Results Window

The controls in the Run Dialog Results Window are as follows:

Control	Description
Parameters	Lists the input parameters that were entered into the Run Stored Procedure Dialog Box. The name, direction, type and value appear.
Out Parameters	Lists the output parameters exposed by this stored procedure or function. The name, direction, type and value appear. Each parameter of type REF CURSOR appears as a link, which you can click to display its value. For all other data types, the values are displayed inline with a maximum of 4000 characters shown. User Defined Types are displayed in XML format and include only the first level of nestedness. The **Select for Config** check box appears for output parameters of type REF CURSOR only. If this check box is selected, the REF CURSOR metadata information for this parameter is included when the **Show Config** or **Add Config to Project** buttons are pressed.
Show Config	Only appears for procedures or functions that include output parameters of type REF CURSOR. Displays the metadata information for the procedure or function.

Control	Description
Add Config to Project	Only appears for procedures or functions that include output parameters of type `REF CURSOR`. Adds the metadata information for the procedure or function to the `app.config` or `web.config` file of the Visual Studio project. If the metadata information already exists in the config file for this stored procedure or function, an overwrite confirmation dialog appears. If there is no project open, an error is displayed. Note: This metadata information has a different format depending on whether the Server Explorer connection chosen to run this procedure or function uses Managed ODP.NET or Unmanaged ODP.NET as the data source, as described in Connection Dialog Box . Make sure that the data source type used in Server Explorer matches the type of ODP.NET used in the References of your application's project.

See Also

Function Designer | Functions Node | Stored Procedure Designer | Stored Procedures Node

Package Designer

The Package Designer lets you create PL/SQL packages. This section covers the following topics:

- Creating Packages in Oracle Developer Tools
- Starting the Package Designer
- Using the Package Designer
- Add Method Dialog Box

Creating Packages in Oracle Developer Tools

Use the Package Designer to create a new PL/SQL package specification and optionally, an empty package body. If you want to edit an existing package, in Server Explorer, right-click the Package node that represents the package and click **Edit Package Specification**. Then edit the package code in the PL/SQL Code Editor.

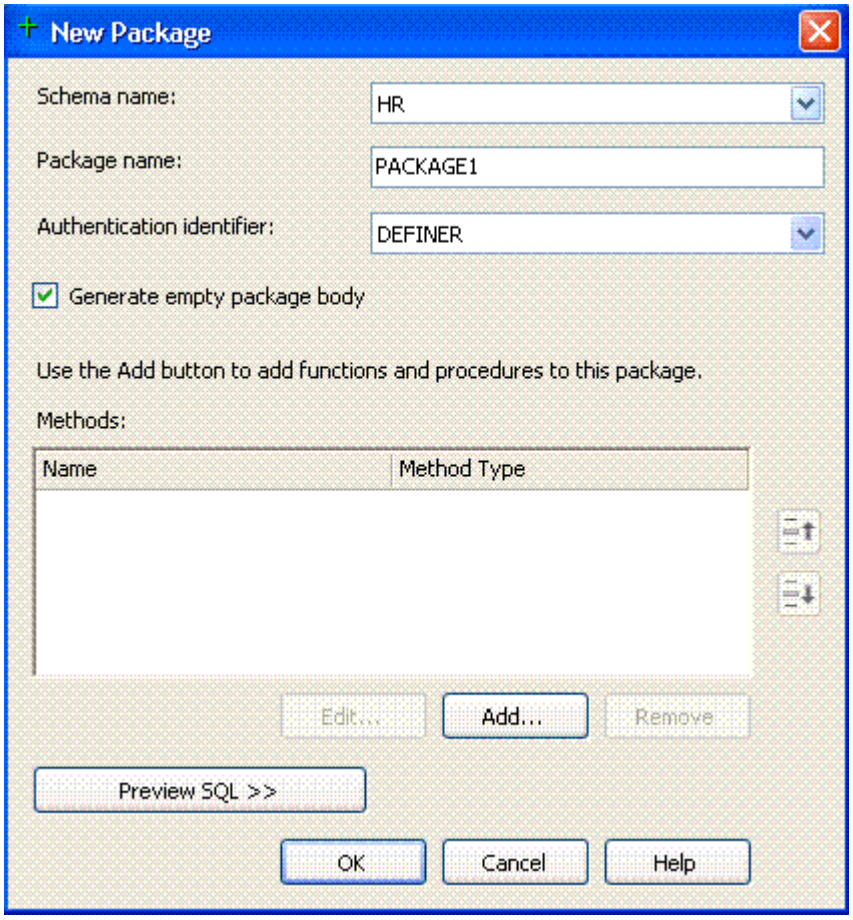
Double-clicking the package node opens the PL/SQL Code Editor. If package has a package body, the PL/SQL Code Editor displays the package body code. Otherwise, the package specification code is available in the PL/SQL Code Editor for editing.

In order to create functions and stored procedures as part of the package, you need to create the package with the **Generate empty package body** check box enabled. To edit the package body, right-click the Package node and select **Edit Package Body** from its menu. The code displays in the PL/SQL Code Editor for you to modify as needed.

Starting the Package Designer

In Server Explorer, right-click the Packages node and from the menu, select **New Package**.

The Package Designer appears similar to the following:



Using the Package Designer

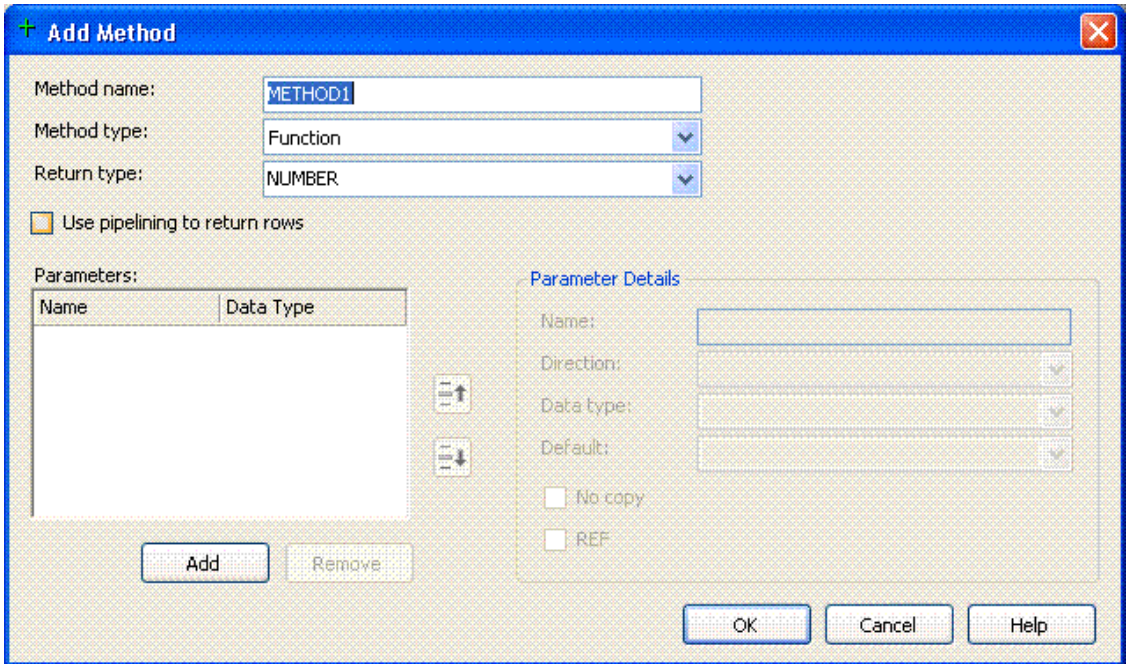
The controls in the Package Designer are as follows:

Control	Description
Schema name	Select from the list of available schemas in which to create the package.
Package name	Enter a name for the package.
Authentication identifier	Select from the following choices: • **CURRENT_USER:** Specifies that the package execute with the privileges of the current user. • **DEFINER:** Specifies that the package execute with the privileges of the owner of the schema in which the package resides, and that external names resolve in the schema where the package resides. This is the default setting.
Generate empty package body	If you want the package specification to include a package body, enable this check box to create an empty package body. You must enable this check box if you want to create the package with stored procedures or functions. The Package Designer lets you generate an empty package body. This package body includes the methods that you defined for the package specification. To customize the package body code, modify it in the PL/SQL Code Editor after you have created the package. You can do so by right-clicking the package node in Server Explorer, and from the menu, selecting **Edit Package Body**.

Control	Description
Methods	Lists the existing methods in this package. To create a new method, click the **Add** button to display the Add Method dialog box. The method specification is added to the package specification. If you have checked **Generate empty package body**, Oracle Developer Tools adds the method specification to the package body also. To change the order of the list of methods, select the method to move and click the **Up** or **Down** arrows. To remove a method, select it and click **Remove**.
Preview SQL	Displays the `CREATE PACKAGE` and `CREATE PACKAGE BODY` SQL statements in a read-only pop-up dialog box.
OK	Saves your work, creates the package (including the package body if you added one) in the database and displays the package code in the PL/SQL Code Editor so that you can customize it. To commit the new package, click the **Save** button in the Visual Studio .NET toolbar or select **Save** from the **File** menu. If you have created the package with errors, Oracle Developer Tools displays an error dialog box, and then displays the error messages in the Output window. Click **OK** and correct the error in the designer. Afterwards, when you click the designer's **OK** button to save your changes, click **Yes** to replace the incorrect package with the corrected version. After you successfully create the package, Oracle Developer Tools displays its node in Server Explorer.

Add Method Dialog Box

The Add Method dialog box appears as follows:



The controls in the Add Method dialog box are as follows:

Control	Description
Method name	Specify the name of a method to add to this package.
Method type	Select from the following choices: • **Function** • **Procedure**
Return type	Select from the choices of data types listed.
Use pipelining to return rows	Returns the results of a table function iteratively.
Parameters	Lists parameters for this method. To create a new parameter, click **Add**, and then use the Parameter Details pane to modify the parameter as needed. When you create parameters, their PL/SQL code appears in the SQL Preview box. To modify the order of parameters, select the parameter to move and click the **Up** or **Down** arrow. To remove a parameter, select it and click **Remove**.
Parameter details	Displays the following information: • **Name:** Enter a name for the parameter. • **Direction:** Select from the following: – **IN:** Indicates that you must supply a value for the argument when calling the method. `IN` parameters can have a default value. – **OUT:** Indicates that the method passes a value for this argument back to its calling environment after execution. – **IN-OUT**: Indicates that you must supply a value for the argument when calling the method and that the method passes a value back to its calling environment after execution. • **Data type:** Select from the list. If the data type is not listed, you can type it here. • **Default:** Specify a default value to use if the parameter is not passed. • **No copy:** Instructs the database to pass this argument as fast as possible. This clause can significantly enhance performance when passing a large value like a record, an index-by table, or a varray to an `OUT` or `IN OUT` parameter. `IN` parameter values are always passed using the **No copy** option. • **REF:** Makes parameter data type objects into reference types.
OK	Creates the method and returns to the New Package dialog box.

See Also

Packages Node| Oracle Query Window

Sequence Designer

The Sequence Designer lets you create or modify sequences. This section covers the following topics:

- Creating Sequences in Oracle Developer Tools
- Starting the Sequence Designer
- Using the Sequence Designer

Creating Sequences in Oracle Developer Tools

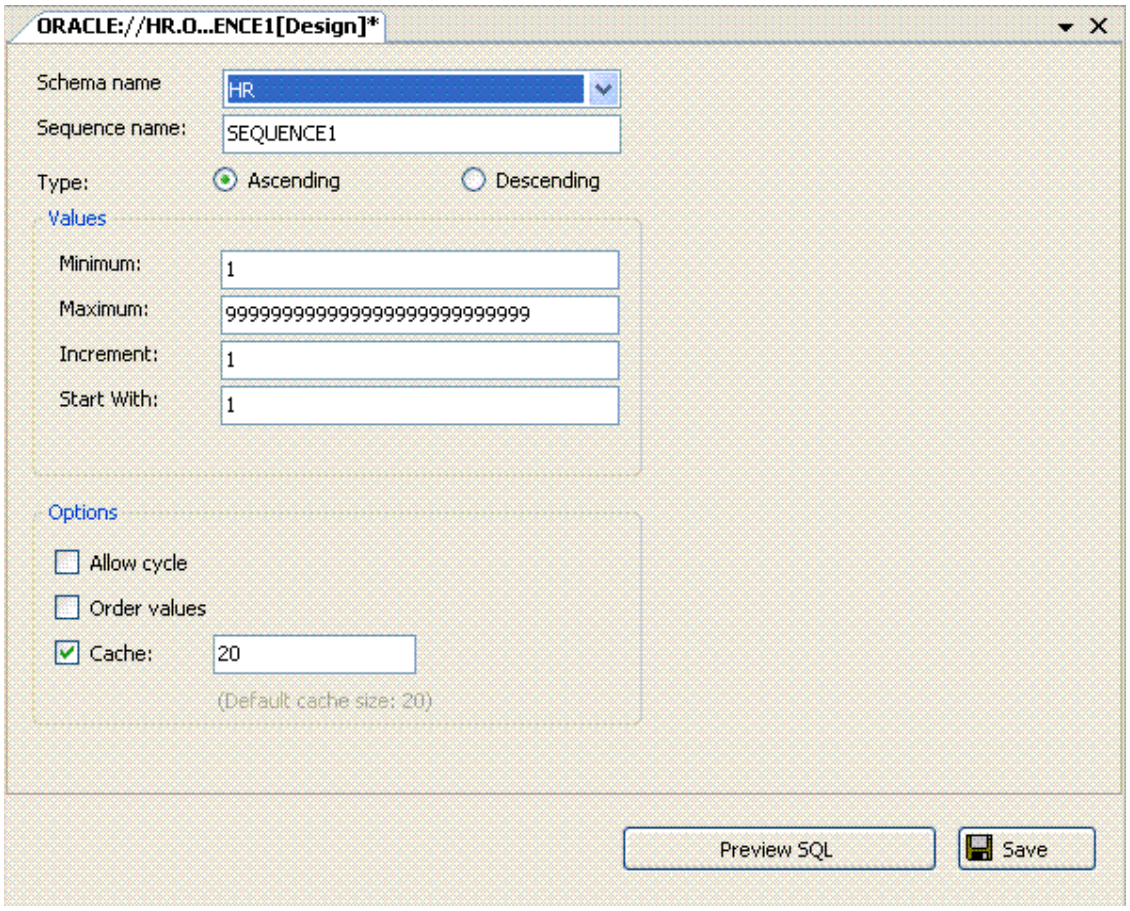
Use the Sequence Designer to create or modify sequences. To create a sequence in your own schema, you must have the `CREATE SEQUENCE` system privilege. To create a sequence in another user's schema, you must have the `CREATE ANY SEQUENCE` system privilege.

Starting the Sequence Designer

Start the Sequence Designer by using one of the following methods:

- If you are creating a new sequence, in Server Explorer, right-click the Sequences node and from the menu, select **New Sequence**.
- If you are modifying an existing sequence, in Server Explorer, double-click the node that represents that sequence, or right-click the node and select **Edit**.

The Sequence Designer appears similar to the following:



Using the Sequence Designer

The Sequence Designer has the following controls:

Control	Description
Schema name	Select from the list of available schemas in which to create the sequence.
Sequence name	Enter a name for the sequence.
Type	Select either of the following: • **Ascending:** Increments each generated sequence value by the interval value. • **Descending:** Decrements each generated sequence value by the interval value.
Values	Specifies the range of values to use for the sequence. • **Minimum:** Minimum value the sequence can generate. This integer value can have 28 or fewer digits. The minimum value must be less than or equal to the initial value and less than the maximum value. • **Maximum:** Maximum value of the sequence. This integer value can have 28 or fewer digits. The maximum value must be equal to or greater than the initial value and greater than the minimum value. • **Interval:** Incremented value between sequences. Enter any positive or negative integer except 0, up to 28 digits. This value can have 28 or fewer digits. The absolute value of **Interval** must be less than the difference of the maximum and minimum values. If this value is negative, the sequence descends. If the value is positive, the sequence ascends. If you omit this setting, the interval defaults to 1. • **Initial value:** The first sequence number to be generated. Use this setting to start an ascending sequence at a value greater than its minimum or to start a descending sequence at a value less than its maximum. For ascending sequences, the default value is the minimum value of the sequence. For descending sequences, the default value is the maximum value of the sequence. This integer value can have 28 or fewer digits.
Options	Select from the following settings: • **Allow cycle:** Lets the sequence continue to generate values after reaching either its maximum or minimum value. After an ascending sequence reaches its maximum value, the sequence generates its minimum value. After a descending sequence reaches its minimum, it generates its maximum value. By default, a sequence cannot generate more values after reaching its maximum or minimum value. • **Order values:** Generates the sequence numbers in order of request. This clause is useful if you are using the sequence numbers as timestamps. Guaranteeing order is usually not important for sequences used to generate primary keys. This setting is necessary only to guarantee ordered generation if you are using Oracle Database with Real Application Clusters. If you are using exclusive mode, sequence numbers are always generated in order. • Cache: Specifies how many values of the sequence the database pre-allocates and keeps in memory for faster access. This integer value can have 28 or fewer digits. The minimum value for this parameter is 2. If you do not select this setting, Oracle Database caches 20 sequence numbers by default.
Preview SQL	Displays the `CREATE SEQUENCE` SQL command code in a pop-up dialog box and in the output window. Read-only.
Save	Commits the sequence to the database. Alternatively, you can click the **Save** button in the Visual Studio .NET toolbar or select **Save** from the **File** menu.

See Also

Sequences Node

Synonym Designer

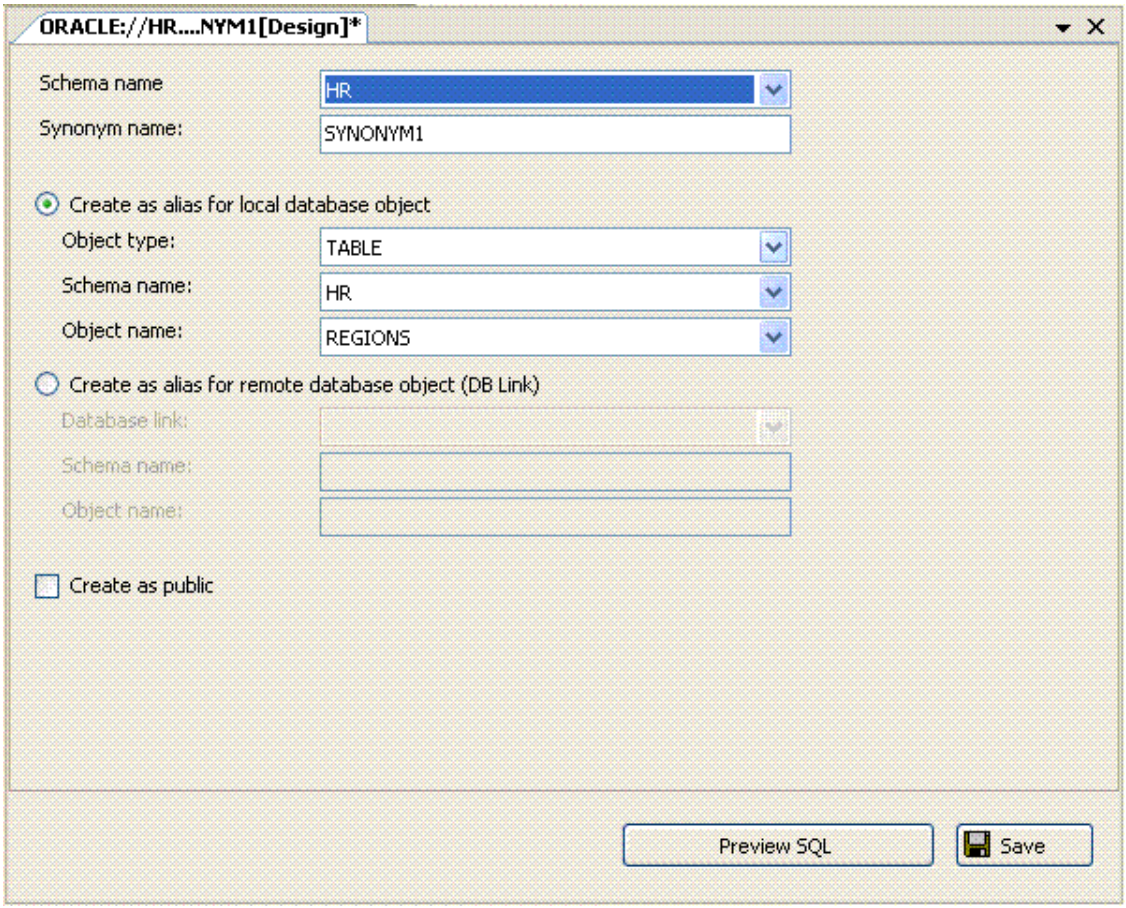
Use the Synonym Designer to create or modify synonyms. This section covers the following topics:

Starting the Synonym Designer

Start the Synonym Designer by using one of the following methods:

- If you are creating a new synonym, in Server Explorer, right-click the Synonyms node and from the menu, select **New Synonym**.
- If you are modify an existing synonym, in Server Explorer, double-click the node that represents the synonym, or right-click the node and select **Edit** from the menu.

The Synonym Designer appears similar to the following:



Using the Synonym Designer

In general, use the Synonym Designer to create or modify synonyms.

The Synonym Designer has the following controls:

Control	Description
Schema name	Select from the list of available schemas in which to create this synonym.
Synonym name	Enter a name for the synonym. The functional maximum length of the synonym name is 32 bytes. Names longer than 30 bytes are permitted for Java functionality only. If you specify a name longer than 30 bytes, Oracle Database encrypts the name and places a representation of the encryption in the data dictionary. The actual encryption is not accessible, and you cannot use either your original specification or the data dictionary representation as the synonym name.
Create as alias for local database object	Specifies the source object of the synonym from the current database. This setting creates a non-database link-based synonym. Settings are as follows: • **Object type:** Type of object, such Table, View, Function, Procedure, Package, Sequence, Synonym, Java class, public Synonym. The schema object cannot be contained in a package. • **Schema name:** Select the schema containing the object for which you are creating a synonym (source object). • **Object name:** Source object name for this synonym. Available choices are objects in the selected schema that are of the specified object type.
Create as alias for remote database object (DB Link)	Specifies a complete or partial database link (DB Link) so that you can create a synonym for a schema object located on a remote database. Note that you cannot specify a database link for a Java class synonym. Settings are as follows: • **Object type:** Type of object, such Table, View, Function, Stored Procedure, and so on. • **Schema name:** Enter a name for the schema in the remote database. • **Object name:** Enter the source object name for this synonym.
Create as public	Specifies the synonym as a public synonym. All users have access to public synonyms. However, each user must have appropriate privileges on the underlying object in order to use the synonym. When resolving references to an object, Oracle Database uses a public synonym only if the object is not prefaced by a schema and is not followed by a database link. If you do not select Create as public, the synonym is private and is accessible only within its schema. A private synonym name must be unique in its schema.
Preview SQL	Displays the `CREATE SYNONYM` SQL command code in a pop-up dialog box and in the output window. Read-only.
Save	Commits the synonym to the database. Alternatively, you can click the **Save** button in the Visual Studio .NET toolbar or select **Save** from the **File** menu.

See Also

Synonyms Node

XML Schema Designer

The XML Schema Designer lets you create XML schemas. For more information on XML schemas, see *Oracle XML DB Developer's Guide*.

This section covers the following topics:

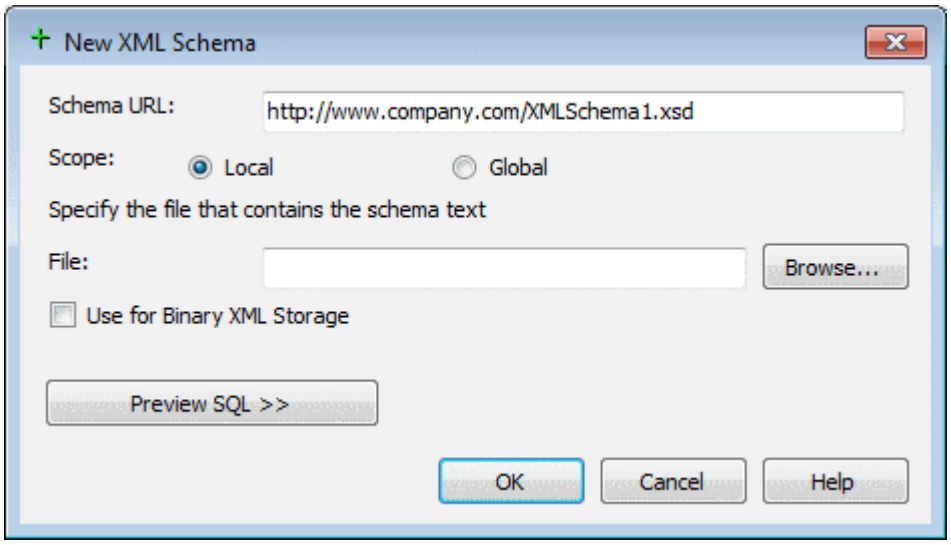
- Starting the XML Schema Designer

- Using the XML Schema Designer

Starting the XML Schema Designer

In Server Explorer, right-click the XML Schemas node and from the menu, select **New XML Schema**.

The XML Schema Designer appears as follows:



Using the XML Schema Designer

In general, to use the XML Schema Designer, you specify the schema's URL, its scope, and the file containing the schema's text.

The XML Schema Designer has the following controls:

Control	Description
Schema URL	Enter the name of the schema in standard URL format, for example: `http://xmls.mycompany.com/myfile.xsd`
Scope	Select from the following: • **Local:** Sets the schema to be visible only to the user creating it. • **Global:** Sets the schema to be visible to the public.
Specify the file that contains the schema text	In the **File** box, specify the file name or click **Browse** to search for the file in the local file system.
Use for Binary XML Storage	Registers the XML Schema to be used for binary XML storage. `XMLType` tables or columns that need to store a schema-based `XMLType` as binary XML must use an XML Schema registered for binary XML storage. XML Schemas registered for binary XML storage cannot be used to create schema-based `XMLType` table or columns with Object relational storage. Binary XML is available in Oracle Database version 11.2.0.1 or later.
Preview SQL	Displays the SQL that will be executed when OK is clicked.
OK	Creates the XML Schema and closes the dialog.

Control	Description
Cancel	Closes the dialog without creating the XML Schema.
Help	Opens this help page.

See Also

XML Schemas Node

Object Type Designer

The Object Type Designer lets you create or modify object types. This section covers the following topics:

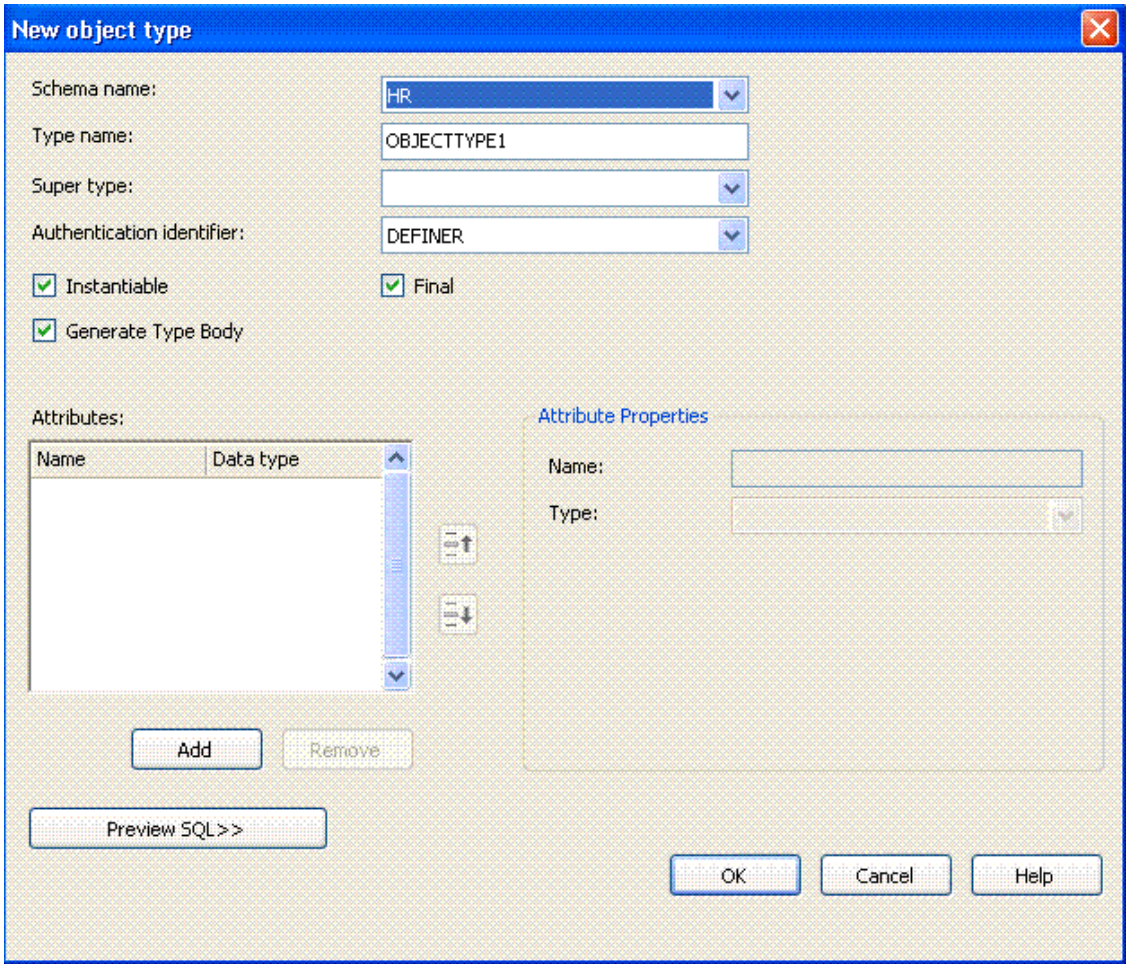
- Creating Object Types in Oracle Developer Tools
- Starting the Object Type Designer
- Using the Object Type Creation Designer

Creating Object Types in Oracle Developer Tools

Use the Object Type Designer to create object types.

Starting the Object Type Designer

Start the Object Type Designer from the Server Explorer, by right-clicking the User Defined Categories node, and selecting the **New Object Type**. The Object Type Designer, similar to the following appears.



Using the Object Type Creation Designer

The Object Type Designer creates new object types.

The controls in the Object Type Designer are as follows:

Control	Description
Schema name	Select the schema for the new type. The list contains all the schemas available to you as defined in the Filters Tab on the Connection Dialog. By default, the schema for the new object type is the one that you logged in with. If that schema was not defined in the filters tab, then the default schema is the first schema in the sorted list of available schemas.
Type name	Displays the name of new object type. By default, the name is generated in the format `OBJECTTYPE`*X* where *X* can be any number.
Super Type	Optional: Type or select the name of a type existing in the database to serve as super type to derive the new object type from. The list displays all types that are accessible to you and that are `NOT FINAL`, meaning that subtypes can be derived from the type. By default, the field is empty.

Control	Description
Authentication identifier	Select the authentication identifier for the type. The possible values are: • **CURRENT_USER:** Specifies that the function executes with the privileges of the current user. • **DEFINER:** Specifies that the function executes with the privileges of the owner of the schema in which the function resides, and that external names resolve in the schema where the function resides. This is the default setting.
Instantiable	Select **Instantiable** if the type is be instantiable. A type is instantiable if instances of the type can be created. Instantiable is selected by default.
Final	Select **Final** if the type is to be final, meaning no subtypes can be derived from the type. Final is selected by default.
Generate Type Body	Select **Generate Type Body** to generate a type body. The default is checked. If this is checked, ODT adds a method named `METHOD1`, with a parameter `PARAM1` of type `NUMBER` before generating the type body.
Attributes	Displays the name of the attribute that have been added to the type and its data type. This grid is read-only. Selecting a row in the data grid changes the display in the Attribute Properties section correspondingly. The properties of the attribute can be modified in the Attribute Properties section. Click the up and down arrows to change the attribute position up in the data grid. This alters the order of the attribute in the type definition in the database.
Add	Adds a new attribute to the type. Properties of the new attribute are displayed in the Attribute Properties section.
Remove	Removes the currently selected attribute from the data grid.
Attribute Properties	This section enables you to specify properties for an attribute. The first two properties, Name and Type, are constant. Other properties change based on the type of the attribute. • **Name:** Enter the name of the attribute. By default, the attribute name is generated in the format `ATTRIBUTE`*X* where *X* is a number. • **Type:** Select the data type of the attribute from the list. By default, the type is a number. The list displays all the accessible database predefined types, ANSI types, and available user-defined types. When the type changes, other attribute properties change. You cannot create attributes of these data types: `LONG`, `LONG RAW`, `ROWID`, and `UROWID`. • **Size:** Represents the size of the character attributes. Default is `20`.
Attribute Properties	• **Length Semantics:** Represents the semantic used in the database for the character type. *Default* indicates that the database default is used. Other valid values are `byte` and `char`. • **Precision:** For types `DECIMAL`, `NUMBER`, `NUMERIC`: Represents precision. The default is an empty value. For types `TIMESTAMP`, `TIMESTAMP_WITH_TIME_ZONE`, `TIMESTAMP_WITH_LOCAL_TIME_ZONE`, and `FLOAT`: Represents the fractional seconds precision. The default is 6.
Attribute Properties	• **Scale:** For types `DECIMAL`, `NUMBER`, and `NUMERIC`: Represents scale. Default is an empty value. • **Day:** For types `INTERVAL_DAY_TO_SECOND`: Represents day precision. Default is 2. • **Fractional Seconds Precision:** For types `INTERVAL_DAY_TO_SECOND`: Represents the seconds precision. Default is 6. • **Year Precision:** For type `INTERVAL_YEAR_TO_MONTH`: Represents the precision in years. Default value is 2. • **Create as ref:** For type Object: Specify whether or not the type of the attribute is a `REF`. Only appears for object types. Default is unchecked.

Control	Description
Preview SQL	Launches the preview SQL Dialog box which lists the CREATE statement that will create the object type in the database.
Cancel	Closes the dialog box without creating the object type in database.
Help	Launches the help page for New Object Type dialog.
Ok	Creates the object type in the database and displays any errors in the output window. The newly created object type is added to the User-Defined Types in Server Explorer and the object type specification or body is opened in the PL/SQL editor.

Varray Designer

The VARRAY Designer lets you create or modify varrays. This section covers the following topics:

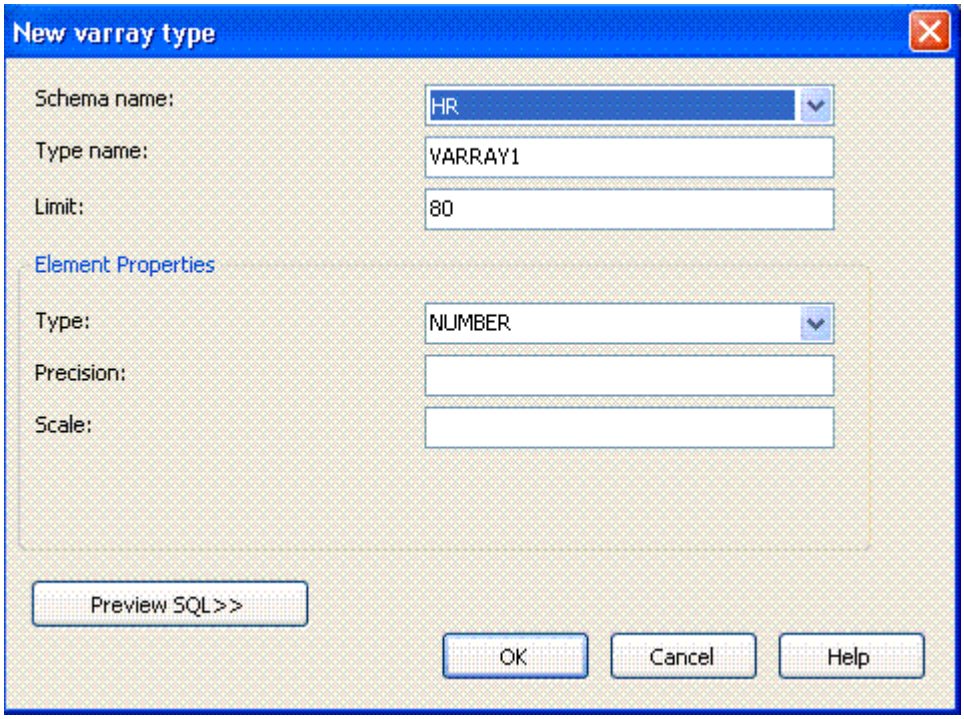
- Creating VARRAYs in Oracle Developer Tools
- Starting the VARRAY Designer
- Using the VARRAY Designer

Creating VARRAYs in Oracle Developer Tools

Use the VARRAY Designer to create VARRAY types.

Starting the VARRAY Designer

Start the VARRAY Designer from the Server Explorer by right-clicking the User-Defined Categories node, and selecting the **New Varray**. The VARRAY Type Designer, similar to the following appears.



Using the VARRAY Designer

The VARRAY Designer creates new VARRAY types.

The controls in the VARRAY Designer are as follows:

Control	Description
Schema name	Select the schema for the new type. The list contains all the schemas that are available to you as defined in the Filters Tab on the Connection Dialog. By default, the schema for the new VARRAY type is the one you have logged in with. If that schema was not defined in the filter tab, then the default schema is the first schema in the sorted list of available schemas.
Type name	Displays the name of the new VARRAY type. By default, the name is generated in the format VARRAY*X* where X can be any number.
Limit	The maximum size of the VARRAY type. Default is 80.

Control	Description
Element Properties	This section enables you to specify properties for the element of the VARRAY type. The first property, **Type**, is constant. Other properties change based on the type. • **Type:** The data type of the element of the VARRAY type. The list contains all the database predefined types, ANSI types, and available user-defined types. When the type changes, other attribute properties change. You cannot create attributes of these data types: LONG, LONG RAW, ROWID, and UROWID. • **Size:** Represents the size of Character types. Default is 20. • **Precision:** For types DECIMAL, NUMBER, NUMERIC: Represents precision for numeric types. You can specify the precision for the type in this field. The default value is empty, meaning no precision is specified. For types TIMESTAMP, TIMESTAMP_WITH_TIME_ZONE, TIMESTAMP_WITH_LOCAL_TIME_ZONE, and FLOAT: Represents the fraction seconds precision for the timestamp type. Default is 6. • **Scale:** For types DECIMAL, NUMBER, NUMERIC: Represents the scale for numeric types. Default value is empty, meaning no scale is specified. • **Day:** For INTERVAL_DAY_TO_SECOND: Represents the day precision for the interval type. Default is 2. • **Fractional second:** For INTERVAL_DAY_TO_SECOND: Represents the seconds precision for the interval type. Default value is 6. • **Year Precision:** For INTERVAL_YEAR_TO_MONTH: Represents the precision in terms of year, Default value is 2.
Preview SQL	Launches the preview SQL Dialog box which displays the CREATE statement that creates the VARRAY type in the database.
Cancel	Closes the dialog box without creating the VARRAY type in database.
Help	Launches the help page for **New Varray** dialog.
Ok	Creates the VARRAY type in the database and displays any errors in the output window. The newly created VARRAY type is added to the User-Defined Types in Server Explorer and the VARRAY type definition is opened in the PL/SQL editor.

Nested Table Type Designer

The Nested TableType Designer creates new nested table types.

This section covers the following topics:

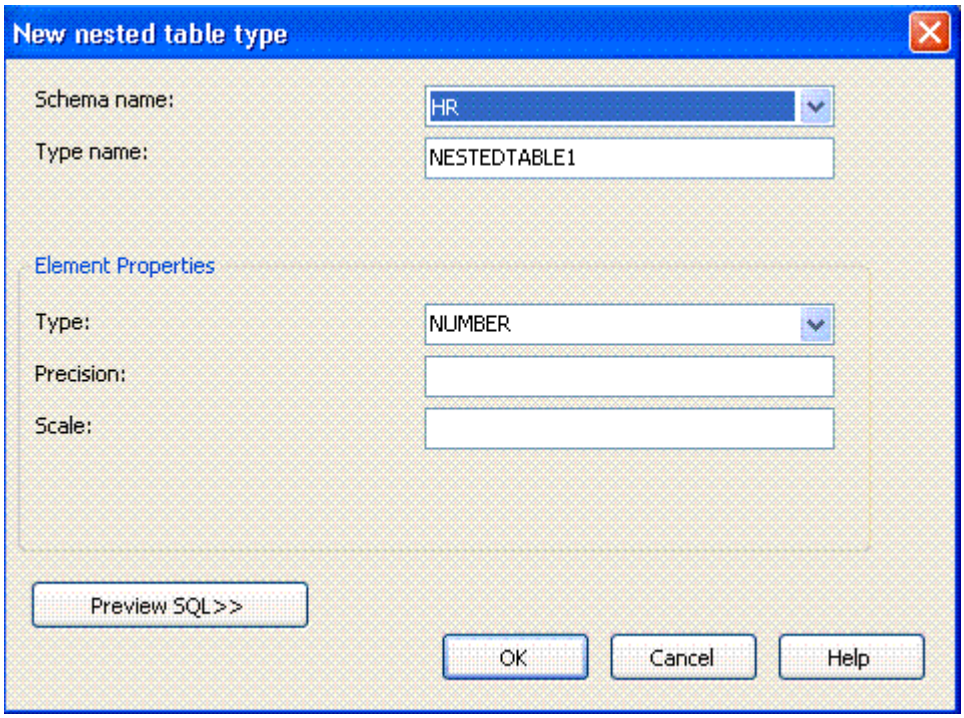
- Creating Nested Tables in Oracle Developer Tools
- Starting the Nested Table Designer
- Using the Nested Table Designer

Creating Nested Tables in Oracle Developer Tools

Use the Nested Table Designer to create Nested Table types.

Starting the Nested Table Designer

Start the Nested Table Designer from the Server Explorer by right-clicking the User-Defined Categories node, and selecting the **New Nested Table**. The Nested Table Type Designer, similar to the following appears.



Using the Nested Table Designer

The Nested Table Designer creates new nested table types.

The controls in the Nested Table Designer are as follows:

Control	Description
Schema name	Select the schema for the new type. The list contains all the schemas that are available to you as defined in the Filters Tab on the Connection Dialog. By default, the schema for the new Nested Table type is the one you have logged in with. If that schema was not defined in the apply filter dialog box, then the default schema is the first schema in the sorted list of available schemas.
Type name	Displays the name of the new Nested Table type. By default, the name is generated in the form NESTEDTABLE*XXX* where X can be any number.

Control	Description
Element Properties	This section enables you to specify properties for the element of the Nested Table type. The first property, **Type**, is constant. Other properties change based on the type. • **Type:** The data type of the element of the Nested Table type. The list contains all the database predefined types, ANSI types, and available user-defined types. When the type changes, other attribute properties change. You cannot create attributes of these data types: `LONG`, `LONG RAW`, `ROWID`, and `UROWID`. • **Size:** Represents the size of Character types. Default is 20. • **Precision:** For types `DECIMAL`, `NUMBER`, `NUMERIC`: Represents precision for numeric types. You can specify the precision for the type in this field. The default value is empty, meaning no precision is specified. For types `TIMESTAMP`, `TIMESTAMP_WITH_TIME_ZONE`, `TIMESTAMP_WITH_LOCAL_TIME_ZONE`, and `FLOAT`: Represents the fraction seconds precision for the timestamp type. Default is 6. • **Scale:** For types `DECIMAL`, `NUMBER`, `NUMERIC`: Represents the scale for numeric types. Default value is empty, meaning no scale is specified. • **Day:** For `INTERVAL_DAY_TO_SECOND`: Represents the day precision for the interval type. Default is 2. • **Fractional second:** For `INTERVAL_DAY_TO_SECOND`: Represents the seconds precision for the interval type. Default value is 6. • **Year Precision:** For `INTERVAL_YEAR_TO_MONTH`: Represents the precision in terms of year, Default value is 2.
Preview SQL	Launches the preview SQL Dialog box which displays the `CREATE` statement that creates the Nested Table type in the database.
Cancel	Closes the dialog box without creating the Nested Table type in database.
Help	Launches the help page for **New Nested Table** dialog.
Ok	Adds the newly created Nested Table type to the User-Defined Types in Server Explorer and opens the Nested Table type definition in the PL/SQL editor.

OracleDataAdapter Wizard

The OracleDataAdapter Wizard generates Oracle Data Provider .NET code so that you can connect Oracle Database to your applications. It lets you modify the `SELECT`, `INSERT`, `UPDATE`, and `DELETE` statements and connection information that are generated in the `OracleDataAdapter`, `OracleConnection`, and `OracleCommand` components.

The OracleDataAdapter Wizard is provided for backward compatibility with Visual Studio 2003.

Visual Studio 2005 and Visual Studio 2008 users typically use Microsoft TableAdapters, using the Microsoft Data Sources window and the Microsoft Table Adapter Wizard to generate the TableAdapters. For more information on TableAdapters, see the Microsoft documentation.

This section covers the following topics:

Starting the OracleDataAdapter Wizard

To start the `OracleDataAdapter` Wizard, use one of the following methods:

- Right-click the OracleDataAdapter component and from the menu, select **Configure with OracleDataAdapter Wizard**.
- Drag an `OracleDataAdapter` component from the toolbox onto the project's form designer.

The `OracleDataAdapter` Wizard's opening screen appears as follows:

Using the OracleDataAdapter Wizard

In general, you use the `OracleDataAdapter` Wizard to modify the Oracle Data Provider data access code for your database objects. After you complete the wizard, the changes are reflected in the `OracleDataAdapter`, `OracleConnection`, and `OracleCommand` components.

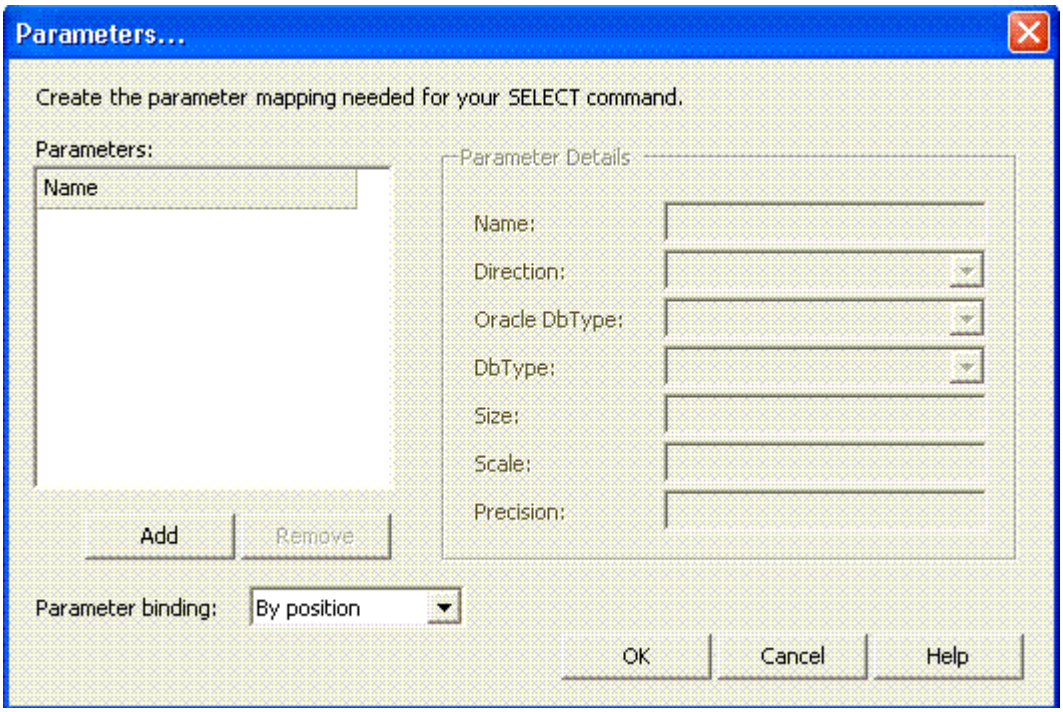
The windows in the `OracleDataAdapter` wizard are as follows:

Window	Description
Welcome to the `OracleDataAdapter` Wizard	Welcome page for the wizard. To bypass this page the next time you run the wizard, select the **Do not show this page again** check box.

Window	Description
Configure your OracleConnection	Specifies the data connection for the Oracle data adapter, with the current connection offered as the default. To use a different connection, click **New Connection**. The new connection information will appear in the OracleConnection component.
Specify your SELECT statement type	Select from the following choices to execute the OracleDataAdapter SELECT command: • **Create a SQL SELECT statement** • **Create anonymous PL/SQL SELECT statement** • **Create SELECT statement using an existing Stored Procedure**
Configure your SELECT statement	Depending on the choices you made in the previous window, do the following: • If you selected either the SQL SELECT statement or the anonymous PL/SQL SELECT statement options in the previous window, enter code to execute the SELECT statement. To create parameters for the SELECT statement, click **Parameters**. To validate that the SELECT statement works, click **Preview Results**. • If you selected the Create SELECT statement using an existing Procedure option, select the procedure that you want from the drop-down list. Procedures appear in the following order: 1. Procedures and functions that the current user owns. 2. Package methods that the current user owns. 3. Procedures and functions that other users own. 4. Package methods that other users own. If you select a function, its parameters (parameter name and data type (all data types and Oracle-specific data types)) appear in the Parameters region. If the parameters cannot be generated, an error appears instead.
Configure your INSERT, UPDATE, and DELETE statements	Select from the following choices: • **Automatic:** Generates the INSERT, UPDATE, and DELETE statements based on the SELECT statement you created in the previous step. • **Custom:** Allows you to customize the INSERT, UPDATE, and DELETE statements. • **Do not create:** Bypasses the creation of INSERT, UPDATE, and DELETE statements.
Specify your INSERT statement type Specify your UPDATE statement Specify your DELETE statement	If you selected Custom in the previous step, use these screens to enter the SELECT statement's INSERT, UPDATE, and DELETE statements. To create parameters for the INSERT, UPDATE, or DELETE statement, click **Parameters**. To validate that the statement works, click **Preview Results**. Note: Not all nodes can generate INSERT, UPDATE, and DELETE statements.
Summary	Displays a summary of all your choices. To complete the wizard, click **Finish**.

Configuring Parameters for Statements

When you configure the `SELECT`, `INSERT`, `UPDATE`, or `DELETE` statement in the `OracleDataAdapter` Wizard, you have the option of setting parameters for the statement by clicking the **Parameters** button. A dialog box similar to the following appears:



The controls in the Parameters dialog box are as follows:

Control	Description
Parameters	Lists parameters for this statement. To create a new parameter, click **Add**, and then use the Parameter Details pane to modify the parameter as needed. To remove a parameter, select it and click **Remove**. The parameters displayed are based on the previous selection: SQL, PL/SQL, or stored procedure.

Control	Description
Parameter Details	Displays the following information, depending on the type of statement used: • **Name**: Enter the parameter's name. This name will be used for parameter binding if you select the **By name** option under **Parameter Binding**. • **Direction:** Select from the following: – **Input:** Indicates that you must supply a value for the parameter when calling the statement. – **Output:** Indicates that the statement passes a value for this parameter back to its calling environment after execution – **Input-Output:** Indicates that you must supply a value for the parameter when calling the statement and that the statement passes a value back to its calling environment after execution. – **ReturnValue:** Indicates that the parameter will hold the return value of a function call. You will not typically have ReturnValue parameters for SQL or PL/SQL statements. • **Type:** Represents either of the following: – **Oracle DbType:** Select from the list of data types supported by Oracle Database. This setting returns the value of the parameter as the appropriate Oracle Data Provider. NET (ODP.NET) type. Alternatively, select the data type from **DbType**. – **DbType:** Select from the list of Microsoft Active X Data Objects .NET (ADO.NET) data types. This setting returns the value of the parameter as the appropriate system type. Alternatively, select the data type from **Oracle DbType**. • **Precision:** Depending on the **Type** selection, specifies the parameter's precision. • **Scale:** Depending on the **Type** selection, specifies the parameter's scale. • **Size:** Depending on the **Type** selection, specifies the size of the parameter.

Control	Description
	• **Source column:** Name of the `DataSet` column to which a parameter is bound. For `INSERT`, `UPDATE`, or `DELETE` statements only. See the description of **Source version** for an example. • **Source version:** Allows you to select which state of a `DataSet` change to write to the database for an `INSERT`, `UPDATE`, or `DELETE` statement. Select from the following: – **Original:** Value before any changes have been applied. Typically, this value is the value that was last retrieved from the database, and is used for parameters in the `WHERE` clause of an update or delete. – **Current:** Current value of the `DataSet`. It is typically used for new values to insert. – **Proposed:** Value for unapplied changes made to the `DataSet` such as edits that are currently in progress, or new rows that have not yet been added to the `DataSet`. – **Default:** Value of the `DataSet` before any changes have been applied. This value has been written to the database by the ODP.NET `AcceptChanges` call. For example, table `EMP` has two columns, `EMPNO` and `ENAME`. Its command text is: `UPDATE EMP SET EMPNO = :0, ENAME = :1 where EMPNO = :2 and ENAME = :3` Setting the `WHERE` clause parameters 2 and 3 to use Original for their source version ensures that updates to the row will not have changed since the last time that row was fetched: – **Parameter 0:** Source version setting = Current; source column = `"EMPNO"` – **Parameter 1**: Source version setting = Current; source column = `"ENAME"` – **Parameter 2:** Source version setting = Original; source column=`"EMPNO"` – **Parameter 3:** Source version setting = Original; source column=`"ENAME"`
Parameter Binding	Select one of the following: • **By position:** Binds the parameters for this command based on the order in which they are added to the `OracleCommand`. • **By name**: Binds the parameters for this command based on the name you enter for the parameter in the **Name** box under **Parameter Details**.
OK	Creates the parameter and returns to the `OracleDataAdapter` wizard.

See Also

| Data Connections Node | Schemas Node | Connecting to an Oracle Database from Visual Studio .NET

Oracle Custom Class Wizard

Using the Oracle Custom Class Wizard, you can generate .NET custom classes for both object types and collection types. These classes can be used to build ODP.NET applications that work with UDTs in Oracle Database.

The Oracle Custom Class Wizard enables you to generate .NET custom classes in one of the following languages: C#, VB.NET, and Managed C.

The wizard allows you to configure some properties of the custom class.

This section covers the following topics:

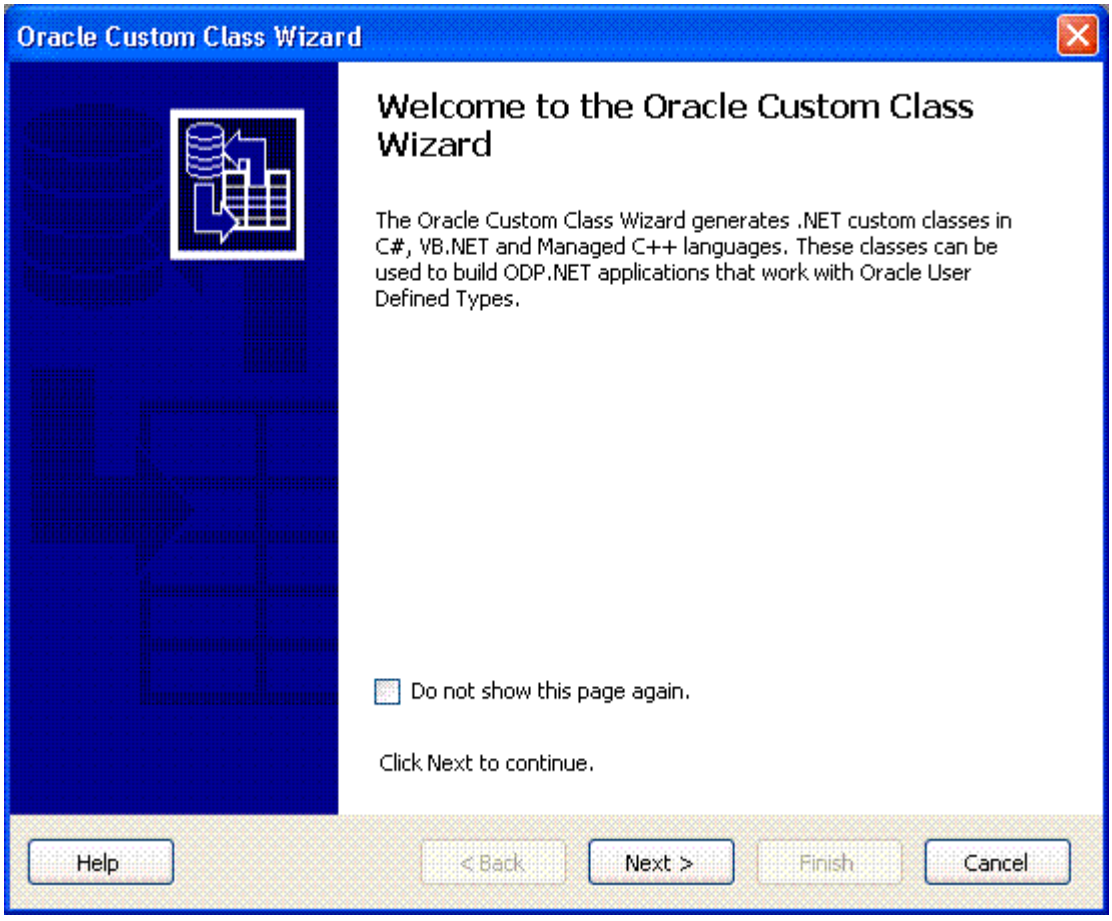
Starting the Oracle Custom Class Wizard

To start the Oracle Custom Class Wizard, do the following:

Select the **Generate Custom Class** item from the menu of an object type or collection type node.

Oracle Custom Class Wizard does not start unless a project is open. If there is a solution without a project, it cannot start.

The opening screen of the Oracle Custom Class Wizard appears as follows:



Using the Oracle Custom Class Wizard

The Oracle Custom Class Wizard gives you the ability to control certain details about the custom class.

Languages in the Oracle Custom Class Wizard

The wizard detects which language the project is in and generates code in that language.

If there is more than one open project from different languages in the solution, then the project that the wizard generates is determined using the following order:

1. The first C# project found.
2. The first Visual Basic project found.
3. The first managed C project found.

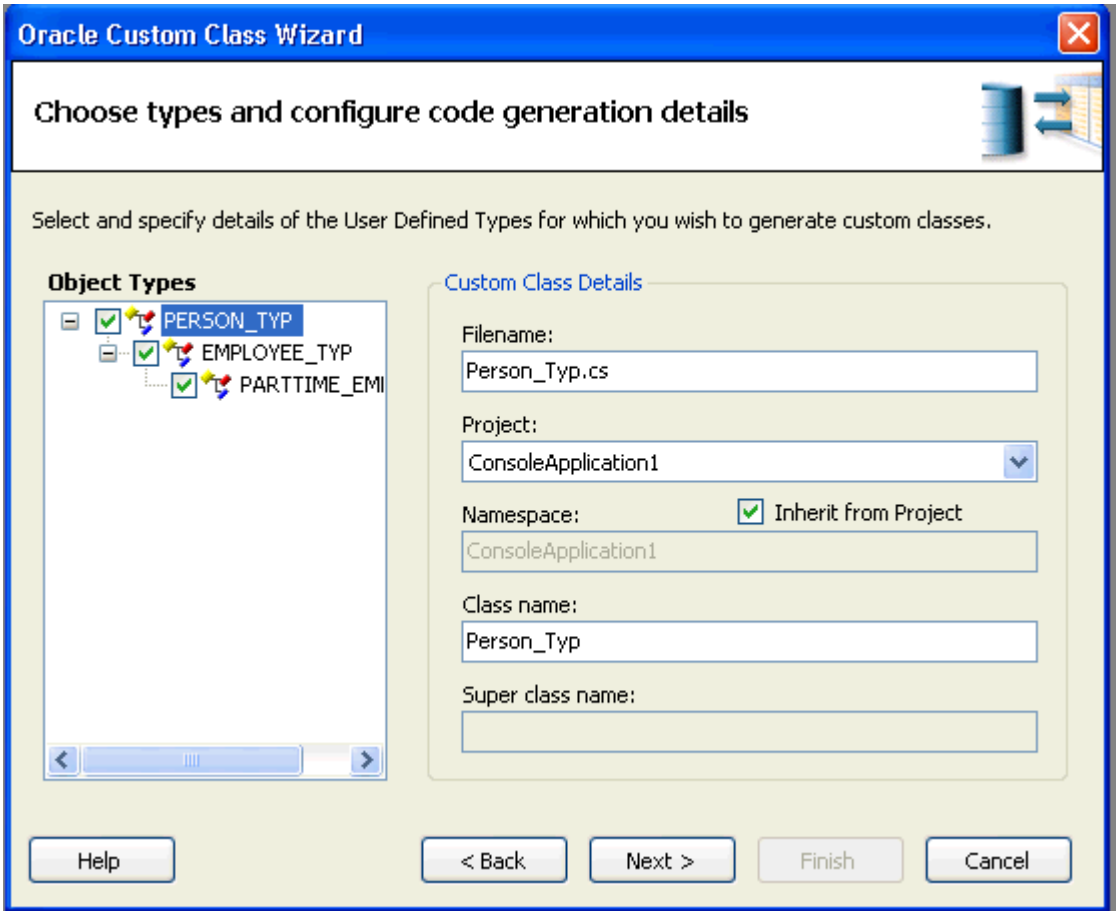
See Also

Generating Classes with the Oracle Custom Class Wizard

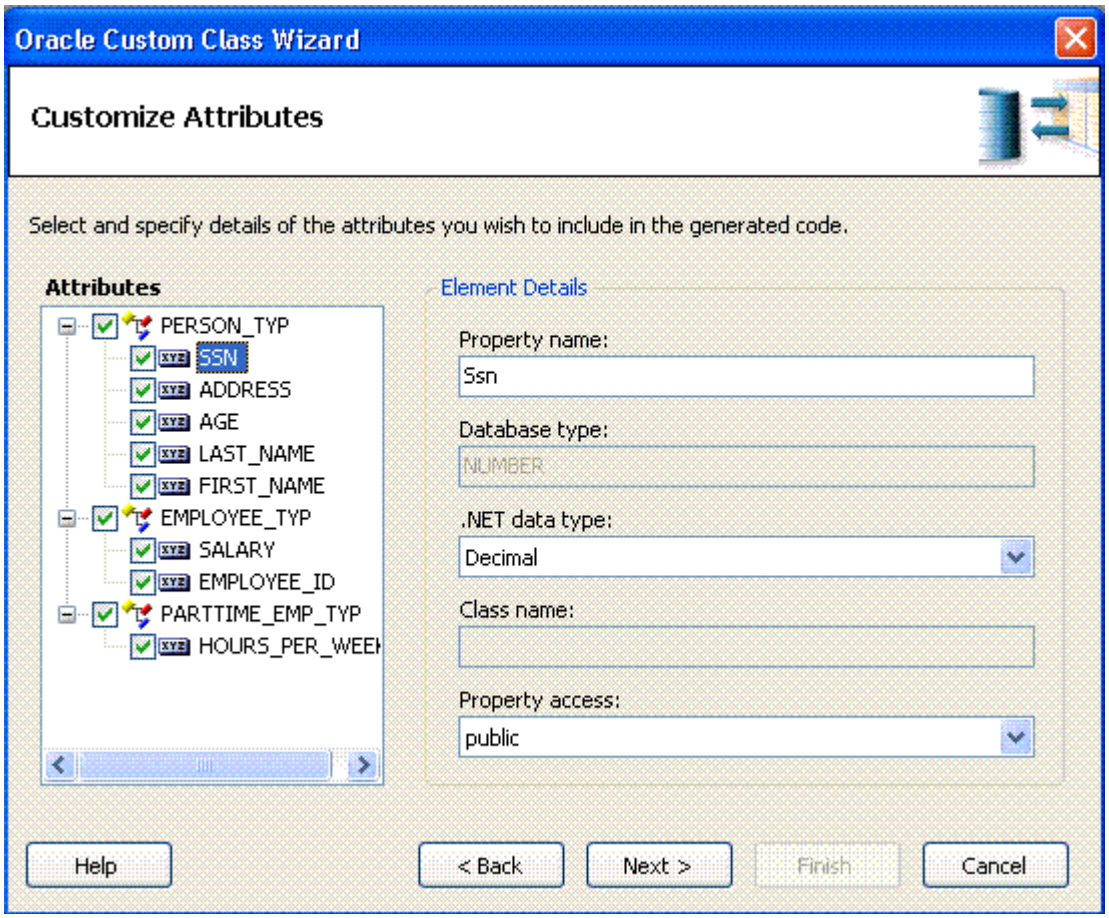
Generating Classes with the Oracle Custom Class Wizard

The Oracle Custom Class Wizard has two primary screens that enable you to customize the generation of code.

The first window enables you to choose types and configure code generation details as follows.



The second window enables you to customize the attributes for the types you have chosen in the first window.



This wizard automatically generates default names for the following based on the related information from the selected user-defined type:

- File name
- Project
- Namespace
- Class name
- Super class Name
- Property Name

All of the auto-generated names will be in mixed case.

The Oracle Custom Class Wizard provides the following windows to generate custom classes corresponding to UDTs of both collection type and object type:

Window	Description
Welcome to the Oracle Custom Class Wizard	Welcome page for the wizard. To bypass this page the next time you run the wizard, select the **Do not show this page** again check box. See "Starting the Oracle Custom Class Wizard".

Window	Description
Choose Types and Configure Code Generation Details	A tree view appears on the left side of the wizard as follows: • For a collection type with an element type of UDT, the tree view shows the base type of the element and below that, the element type. Then it shows the collection type node, in a separate hierarchy, at the same level as the base type of the element. • For an Object Type, the tree view shows the base types in a hierarchical manner. Select the desired user-defined type. See "Using the Oracle Custom Class Wizard" **Next** is disabled if no UDTs are selected. Specify the following information: • **File name**: The file in which the custom class is generated. Default value is name of UDT, with an extension of `.cs`, `.vb`, or `.cpp` based on the type of project you have selected, in the following manner: If UDT is `PERSON_TYPE`, then the file name is `Person_Type.xx`. • **Project:** A list of projects which are loaded in the solution. Each class can be generated in different projects, which you can select from the list. • **Inherit from Project:** If this is checked, then the namespace is derived from the Project name and the Namespace field is read-only. Inherit from Project is checked by default. The field is disabled if the project selected is of type VB.NET. • **Namespace**: The namespace that the class is added to. Default is the name of the project namespace, which is derived from the Project name. You can change this name by deselecting **Inherit from Project**. For VB.NET projects, the custom class is not generated in a separate namespace, but in the default project namespace. The field is disabled if the project selected is of type VB.NET. • **Class name**: The name of the class. Default is name of UDT. • **Super class name**: For Object Types Only: The super class, if you wish to indicate one. If provided, the class you generate derives from the super class entered here. This must be a fully qualified class name with namespace. The default is taken from `ObjectType.SuperClass` as defined in the database. If the type does not define a super type, then this field is disabled.

Window	Description
Customize Attributes	In this window the type (object type or collection type), such as ABS_PERSONAL_TYPE that you selected in the previous screen appears in the left side. The right window displays properties of that type in a read-only mode. These properties are still available by clicking **Back**. The properties include Filename, Project, Namespace, Class Name, and Superclass Name. Click **Next** to view the second part of this window. **Next** is disabled if no UDTs and attributes are selected. The same tree view appears, but the right side contains the properties of the attributes (for object types) and elements (for collection types). When you highlight an attribute or element in the tree view, its properties appear on the right side as follows: • **Property Name:** The name of property that is tied to the attribute. Default value is name of attribute, in title case. For example, if the attribute name is IDNO, the property name is Idno. • **Database Type**: The type of attribute in the database. Read-only. • **.NET Data Type:** The .NET data type that corresponds to the SQL type. You can select from the list of types that are applicable in the context of the SQL type and override the default .NET Types. In case an attribute is itself of type UDT, the .NET Data Type field is blank and read-only. This list also contains Oracle specific types supported by ODP.NET such as OracleBLOB, OracleBFile, and so on, so that properties can map directly to these types. • **Class Name**: Applicable only for attributes of type UDT. Otherwise, this is read-only and blank. For attributes of type UDT, this field is automatically filled with the class name, which is the name of the UDT type in title case. Users can change this name to a class that corresponds to the user-defined type of the attribute that has already been generated or will be generated later. • **Property Access**: The access modifier for the property. Valid values available in the list are public, private, internal, and protected. Default is public, however users can change this value. If either the **Class Name** or **Property Name** field is empty, then the wizard displays an error message.

Window	Description
Summary	Displays a summary of the different tasks to be performed to generate the .NET custom class, if the required information is provided. The Summary indicates the project that the class will be added to and the file that will contain the class definition. This window indicates that the class will not be generated if a UDT Type was de-selected, either in the Configure Class details screen or in the Customize Attributes screen. Click **Finish** to complete the wizard tasks or **Back** to change or add information. During code generation, for each of the files being generated, if a file with a name you provided already exists, you are prompted to overwrite that file as follows: • If you select **Yes**, then the file is overwritten and processing continues. • If you select **No**, then the corresponding files are not created or not overwritten and the wizard remains open, allowing you to rename or overwrite the existing file. For the selected project, the wizard adds reference to `Oracle.DataAccess.dll` (ODP.NET). If the reference already exists, then it is not overridden, but a message displays in the Output Window indicating that the reference was not added. You must have the correct version of ODP.NET in the references section. When the process completes, the wizard closes and all newly created files are opened for editing. The output window also lists the success or failure of code generation.

Integration with Query Designer

> **Note**
>
> This feature works only with Visual Studio 2005 and higher.

Oracle Developer Tools for Visual Studio provides integration with the Microsoft Query Designer which permits you to create and modify queries, using several panes that provide for input and results.

This section includes Oracle-specific limitations for using Query Designer. For generic documentation on using Query Designer, please see the Microsoft documentation at: http://msdn2.microsoft.com/en-us/library/ms172013(vs.80).aspx.

When the Query Designer is launched in Server Explorer or in any node under the connection, the following limitations apply:

- Menu for Query Designer Windows has these limitations:
  - The **Add Table** command shows only Tables and Views
  - The **Verify SQL** command (from the individual panes and the Query Designer toolbar) is not supported by Oracle Developer Tools and displays a message box to indicate this.
  - The Query Type of Make Table is not supported.
- This designer does not generate SQL statements with Oracle specific join syntax

To work around this, you can use syntax such Left Inner Join, Right Outer Join instead of using the plus () symbol. You can modify this query to include Oracle-specific syntax, however, this causes the Query Designer to throw an error. You may ignore the error and proceed with the query. The query executes and a message box appears indicating any errors from the execution.

- Certain properties for Query Designer Window are not supported

  The following Query Designer properties are not supported when launched from Oracle connections in Server Explorer: Destination Table, GROUP BY extension, SQL Comment, Top, Expression, Percent, and With Ties.

- GUI object properties only allow modification of Join properties

  Any GUI objects, including Tables, Views, and Joins, that are added to the diagram, can be viewed, but only the join properties can be modified.

  Other objects support read-only properties.

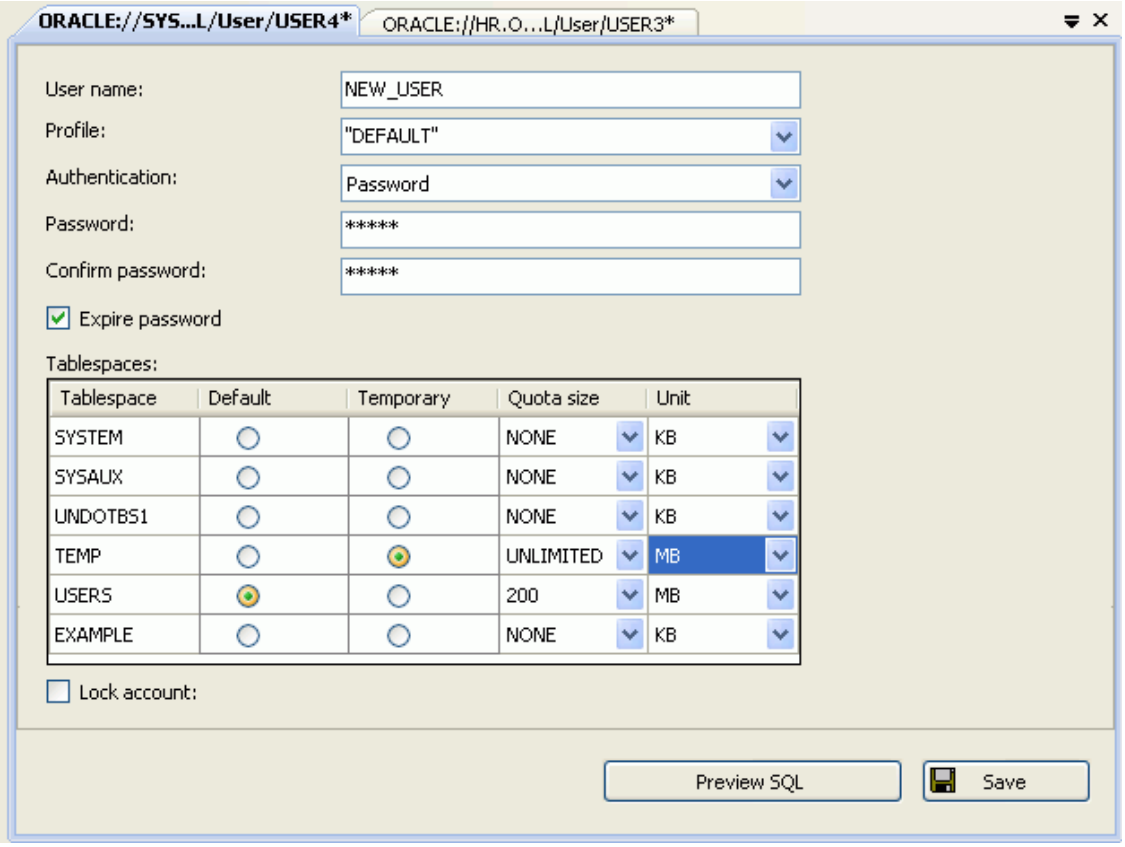
User Designer

The User Designer lets you create a new User or edit existing Users.

This section contains the following topics:

Starting the User Designer

In the Server Explorer, right-click on the users or schemas node, and select **New User**.



Using the User Designer

The User Designer control has some restrictions on non-SYSDBA users as indicated. The controls in the User Designer are as follows:

Control	Description
User name	Enter the username or accept the autogenerated name.
Profile	Enter or select the profile. DEFAULT indicates no restrictions on this resource. Non-SYSDBA users can only select or enter profiles that they have privileges for.
Authentication	Enter or select the authentication type. Valid types are Password (the default), Global, and External. For non-SYSDBA users, the authentication type appears as Password even if it is one of the others.
Password	Enter the password for the user. This is only enabled when the authentication type is Password.
Confirm Password	Enter the password. This is only enabled when the authentication type is Password.
Expire Password	Check to expire the user password. This is only enabled when the Authentication type is password. For non-SYSDBA users, this check box is initially unchecked.
*X.500 Distinguished name	Enter the distinguished name for the user. This is only visible and enabled when the Authentication type is Global.

Control	Description
Tablespaces	Lists the available tablespaces with details in these columns: Tablespace, Default, Temporary, Quota Size, and Unit. For non-SYSDBA users, only tablepaces that they have privileges for appear, and the current values for Default tablespace, Temporary tablespace, and Tablespaces do not appear.
Lock Account	Check to lock the account.
Preview SQL	Displays the CREATE USER SQL dialog. This button is only enabled when there is a pending SQL statement to execute. Verifies that username, password authentication, password, and confirm password are not empty and that password and confirm password are the same.
Save	Verifies that username, password authentication, password, and confirm password are not empty and that password and confirm password are the same. Save the changes.

See Also

Schema Nodes | User Nodes | About Users, Roles, and Privileges

Role Designer

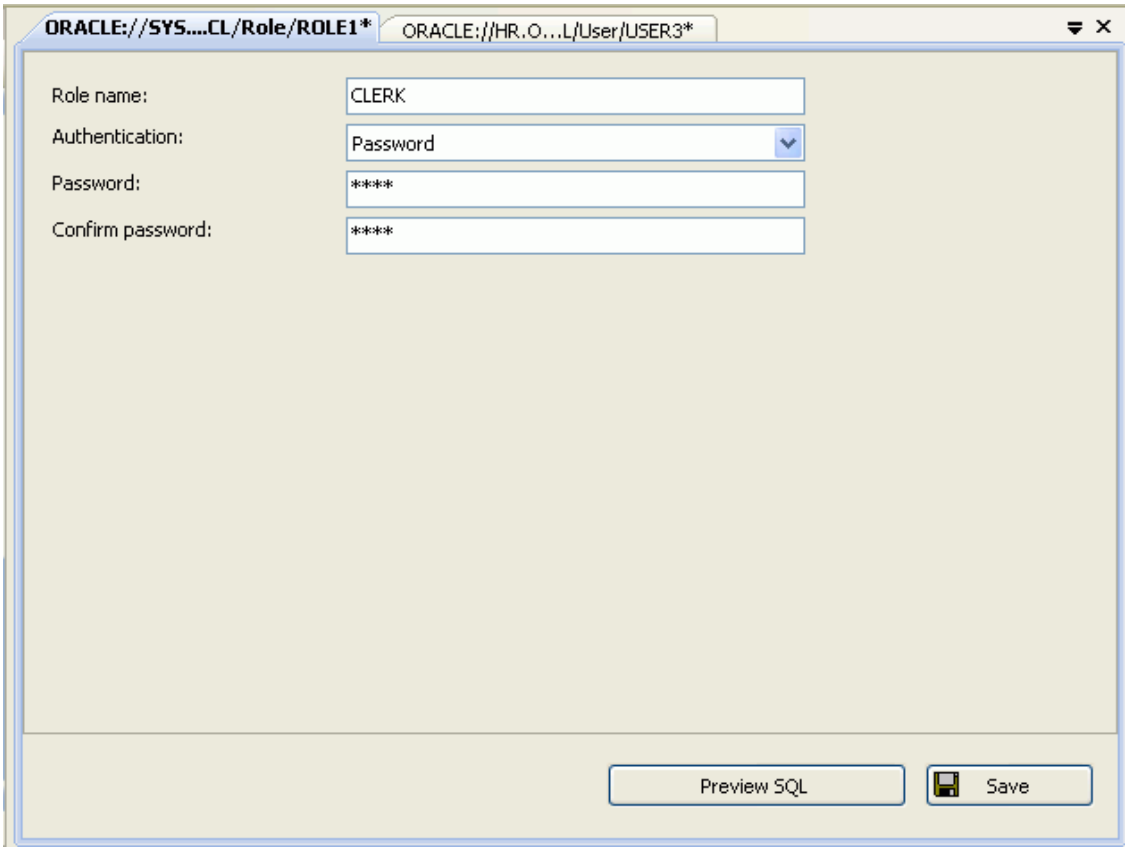
- Creating a Role in Oracle Developer Tools
- Starting the Role Designer
- Using the Role Designer

Creating a Role in Oracle Developer Tools

Use the Role Designer to create new database roles.

Starting the Role Designer

In the Server Explorer, right-click on the Roles node, and select new Role.



Using the Role Designer

The controls in the Role Designer are as follows:

Control	Description
Role name	Enter the role name or accept the autogenerated name.

Control	Description
Authentication	Enter or select the authentication type. The valid values are: None (default), Password, External, and Global. • **None:** Role is automatically authorized by the database and no password is required to enable the role. • **Password:** Lets you create a local role and indicates that the user must specify the password to the database when enabling the role. The password can contain only single-byte characters from your database character set regardless of whether this character set also contains multibyte characters. • **External:** Lets you create an external role. An external user must be authorized by an external service, such as an operating system or third-party service, before enabling the role. Depending on the operating system, the user may have to specify a password to the operating system before the role is enabled. • **Global:** Lets you create a global role. A global user must be authorized to use the role by the enterprise directory service before the role is enabled at login. If a non-SYSDBA user edits a role, the authentication type appears as None even if it is one of the others. For more information on authentication with roles, see `CREATE ROLE` in *Oracle Database SQL Language Reference*.
Password	Enter the password for the user. This is only enabled when the Authentication type is password.
Confirm Password	Enter the password. This is only enabled when the Authentication type is password.
Preview SQL	Displays the `CREATE ROLE` SQL dialog. This button is only enabled when there is a pending SQL statement to execute. Verifies that role name, password authentication, password, and confirm password are not empty and that password and confirm password are the same
Save	Verifies that role name, password authentication, password, and confirm password are not empty and that password and confirm password are the same Save the changes.

See Also

Role Nodes | About Users, Roles, and Privileges

Grant/Revoke Privileges Dialog

This section covers the following topics:

- About the Grant/Revoke Privileges Dialog
- Starting the Grant/Revoke Privilege Dialog
- Using the Grant/Revoke Dialog

About the Grant/Revoke Privileges Dialog

This dialog allows you to grant or revoke:

- Permissions on one or more database objects to a user or role

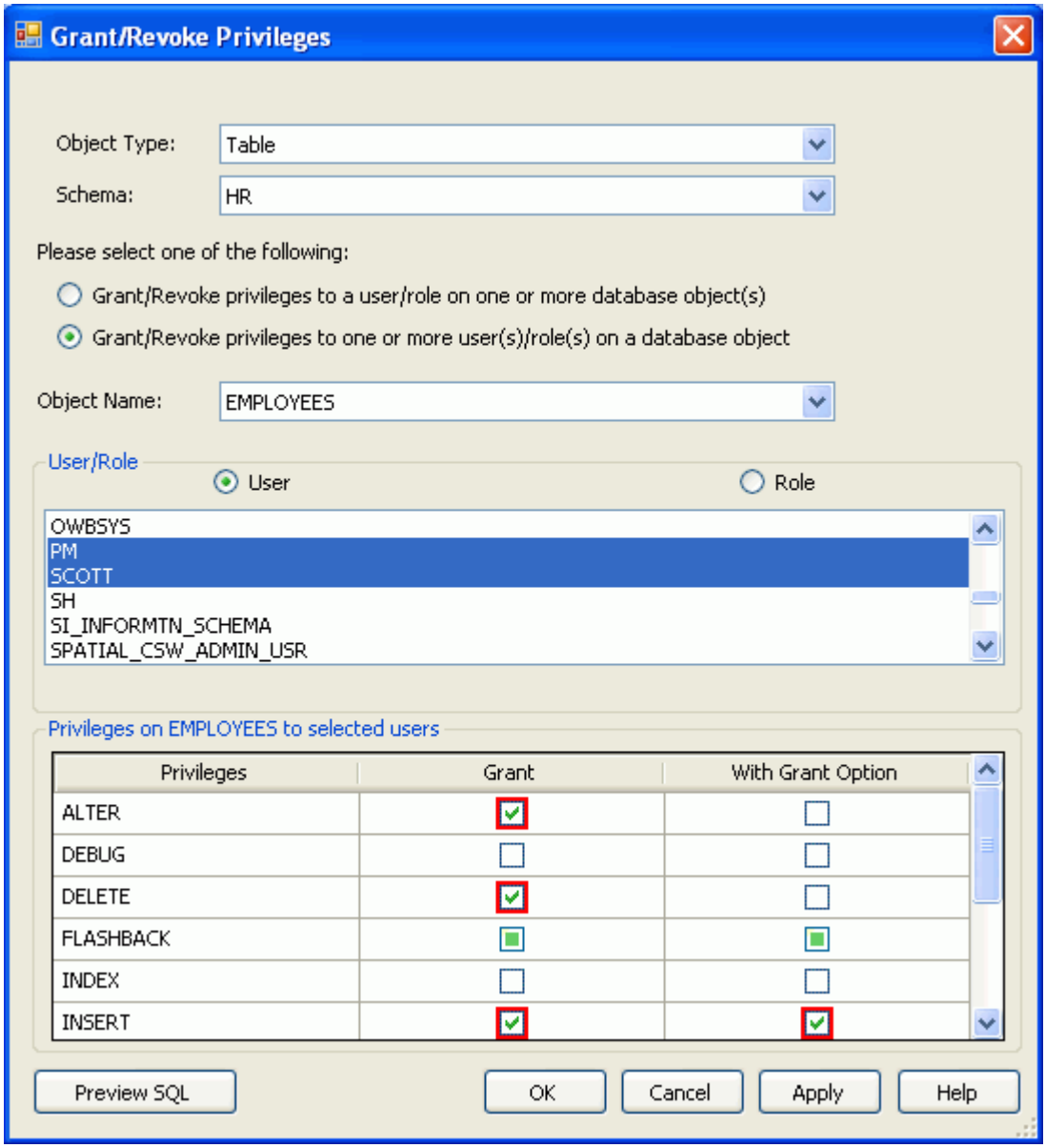
- Permissions on one database object to multiple users or roles
- System privileges to multiple users or roles
- A role to multiple users or roles

See About Users, Roles, and Privileges

Starting the Grant/Revoke Privilege Dialog

To launch the Grant/Revoke Privilege dialog, open Server Explorer, and then select **Privileges** from any node that supports it.

A Grant/Revoke Privilege dialog appears, similar to the following:

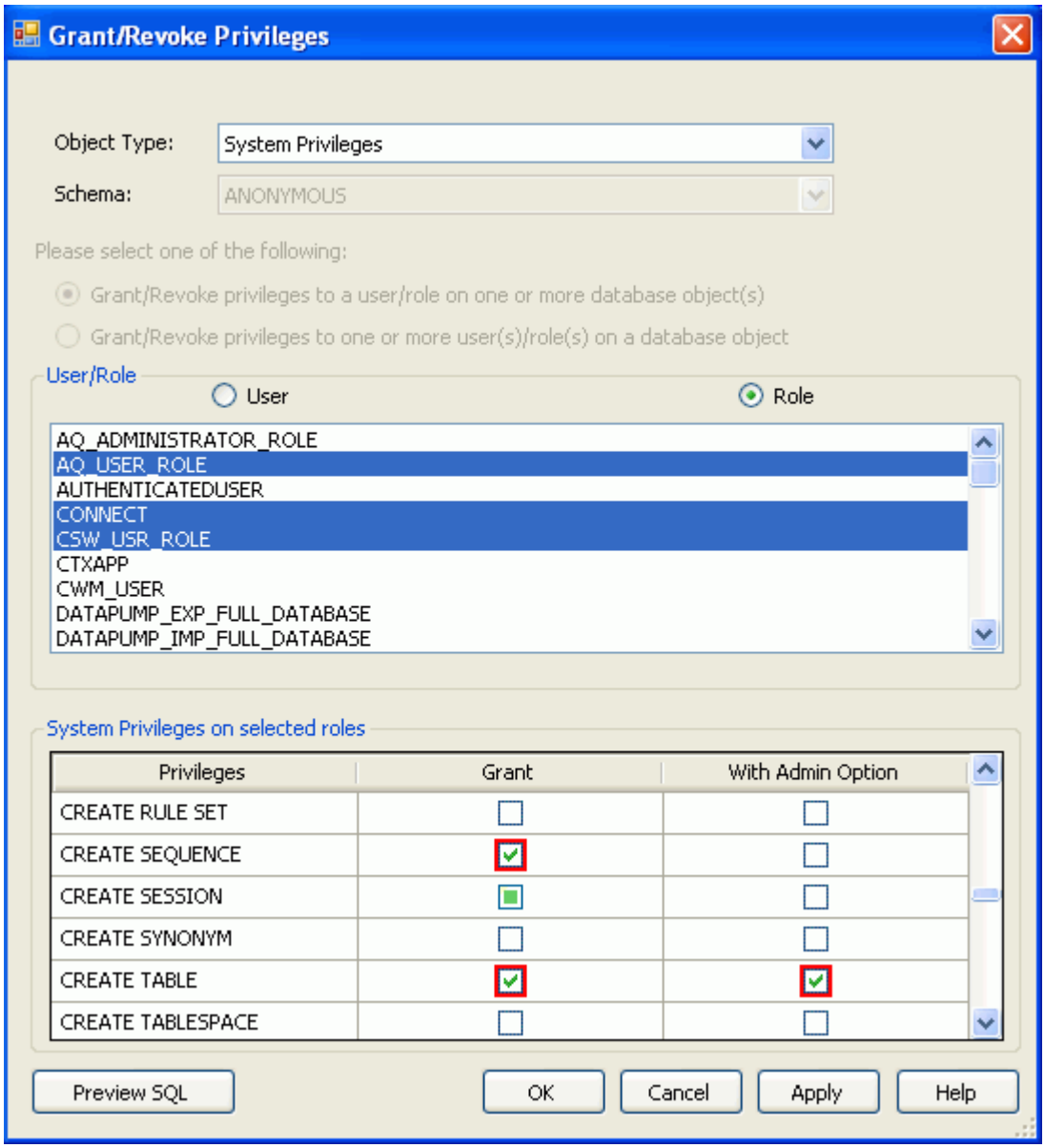


Using the Grant/Revoke Dialog

The controls in the Grant/Revoke privileges dialog box are as follows:

Control	Description
Object Type	Displays the Object Type of the Server Explorer node (can be object node, collection node, or connection node) that launched the dialog, with these exceptions: If the Server Explorer node is `Schema`, `User`, `Role`, `Users`, `Roles`, `Schemas` or Connection Node, then Object Type initially lists System Privileges. Possible values are: `Table`, `View`, `Procedure`, `Function`, `Package`, `Sequence`, `Synonym`, `User-Defined Type`, `User`, `Role`, `Queue`, `Queue Table`, and System Privilege.
Schema	Lists available schemas. Schema is disabled if Object Type is `User` or `Role`.
Please select one of the following	Choose one of the following: • Grant/Revoke privileges to a user/role on one or more database objects. • Grant/Revoke privileges to one ore more user(s)/role(s) on a database object. This control is disabled if Object Type is System Privilege or `Role`.
Object Name	Select from the list of objects of the specified type owned by the selected schema. This control varies depending on the selected option in the previous control as follows: • To select a single database object, select from the drop-down list that appears. • To select multiple database objects, a select list appears. Press the control key or shift key and select individual database objects. Object Name appears for all Object Types except `Role`, `User`, `Schema`, `Roles`, `Users`, `Schemas`, or Connection Node.
User/Role	Select either **User** or **Role** to indicate the target type of the privilege grant. Defaults to User for all nodes except Role or Roles. Then, select available users or roles from the list. The list varies depending on the selected option in the previous controls: • To select a single user or role, select from the drop-down list that appears. • To select multiple users or roles, a select list appears. Press the control key or shift key to make multiple selections. This control is enabled for all object types except for System Privileges.
Privileges	Lists privileges that can or have already been granted to target users or roles. These privileges can be granted with the `With Grant` option, allowing a user or role to grant it to other users or roles, or `With Admin Option` for System Privileges. The `With Grant` or `With Admin option` column is not included if it does not apply. The check boxes have three states: • Checked: All target users or roles have been granted this privilege. • Unchecked: No target users or roles have been granted this privilege. • Shaded: Some target users or roles have been granted this privilege, but not all of them. When appropriate, check boxes have a red border indicating that the user changed the data.
Preview SQL	Displays the SQL that the database executes to grant or revoke privileges.

Control	Description
OK	Saves.
Cancel	Cancels.
Apply	Applies any changes to the database. Any queries that are executed appear in the output pane.
Help	Brings up online help.



The previous screen shot shows the dialog as it appears when Object Type is set to System Privileges.

Definitions of the controls are described above in Using the Grant/Revoke Dialog.

See Also

Oracle Database Security Guide | About Users, Roles, and Privileges| Tables Node | Views Node | Procedures Node | Packages Node | Functions Node | Sequences Node | Object Type Nodes | Synonyms Node

Run SQL*Plus Script Dialog

This section covers these topics:

- Running the Run SQL*Plus Script Dialog
- Using Run SQL*Plus Script

Running the Run SQL*Plus Script Dialog

To run Run SQL*Plus Script, do the following:

1. Select **Run SQL*Plus Script** from the Visual Studio **Tools** menu.

   This launches the Run SQL*Plus Script Dialog box.

2. Enter the full path of the script file in the dialog box or select **Browse** to launch the file selection dialog box. Then select the script file.

3. Select a connection from the drop-down list of connections.

   The Oracle Data Provider for .NET connection string is displayed in the Connection Information area.

4. Select **Run** in the dialog box.

   Run SQL*Plus Script does the following:

   a. Initializes the script execution environment and updates the status/progress bar at the bottom of the dialog box.

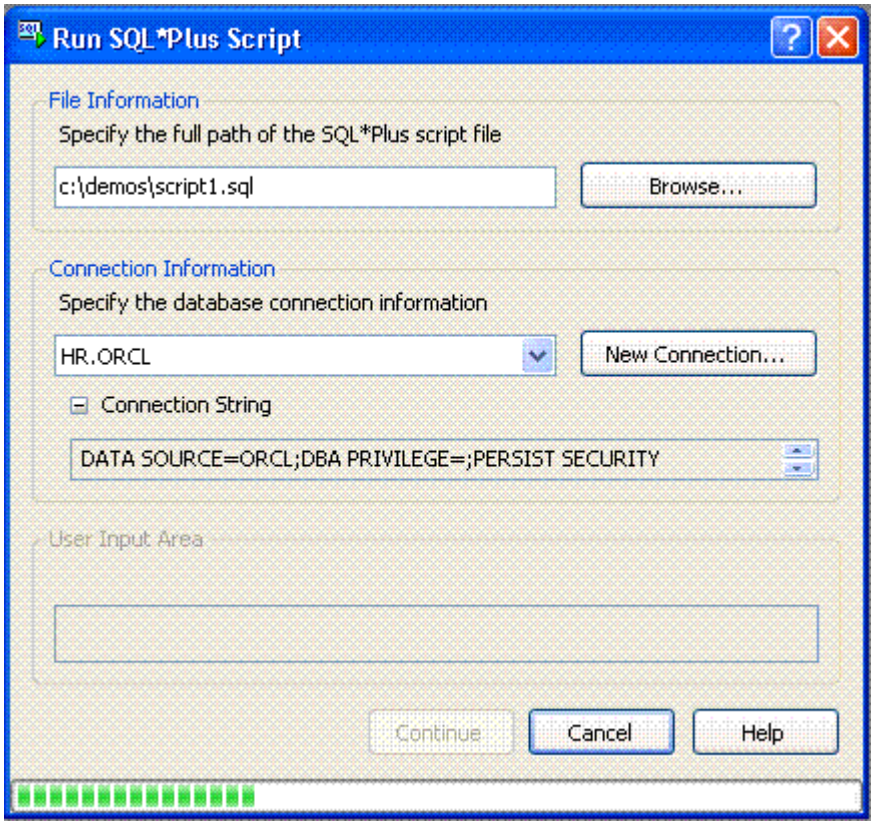
   b. Creates a database connection and updates the status/progress bar at the bottom of the dialog box.

   c. Requests input if the script contains variables.

   d. Executes the script, writes the results to the Oracle Output, and updates the status/progress bar at the bottom of the dialog box.

   e. Closes the database connection and updates the status/progress bar at the bottom of the dialog box.

A dialog box similar to the following appears:



For more information on running Oracle SQL scripts from Visual Studio, see How SQL*Plus Script Execution Works.

Using Run SQL*Plus Script

The controls in Run SQL*Plus Script are as follows:

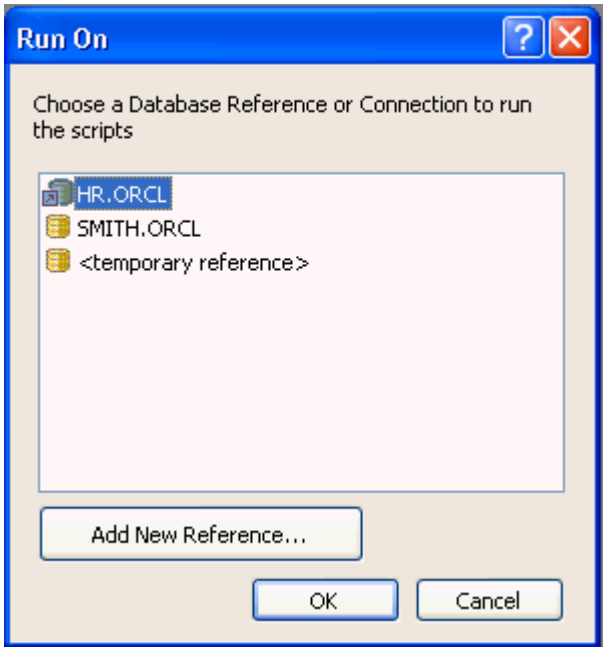
Control	Description
File Information	Displays the file name and full path for the SQL*Plus script file or allows you to enter or edit one.
Browse	Displays the Visual Studio Open File dialog box with the default file name extension set to `.sql` so you can navigate to the SQL*Plus script file.
Connection Information	Displays a list of database connections.
New Connection	Displays the Add Connection Dialog box so that you can create a new database connection.
Connection String	Displays the Oracle Data Provider for .NET connection string for the connection selected in the Connection Source field. This is the connection string used to establish a new connection for executing the SQL*Plus script.
User Input Area	Provides an area for input needed by SQL*Plus commands such as a pause or a parameter value. Every time the script needs input this input area is cleared of all previous entries, and the cursor is positioned on the first row and first column of the input area for new input.

Control	Description
Run	Creates a new database connection using the specified connection information and executes the selected SQL*Plus script in that connection.
Continue	Continues the script execution. Enabled only when you need to be prompted for input. If no input is needed in the script, Continue is disabled until the dialog box closes.
Help	Displays the help for this dialog box.
Status bar	Displays the status bar with the following information: The operation in progress, such as Initializing, Connecting, Executing script, Closing connection. A progress bar indicates the status of the operation by the amount of segmented blocks.

Oracle Database Project Run On Dialog

If you right-click on a script in the Oracle Database Project, and select **Run On**, then the Run On dialog box launches.

A dialog box similar to the following appears:



You can choose which connection to use in the dialog box.

The controls in Oracle Database Project Run On dialog are as follows:

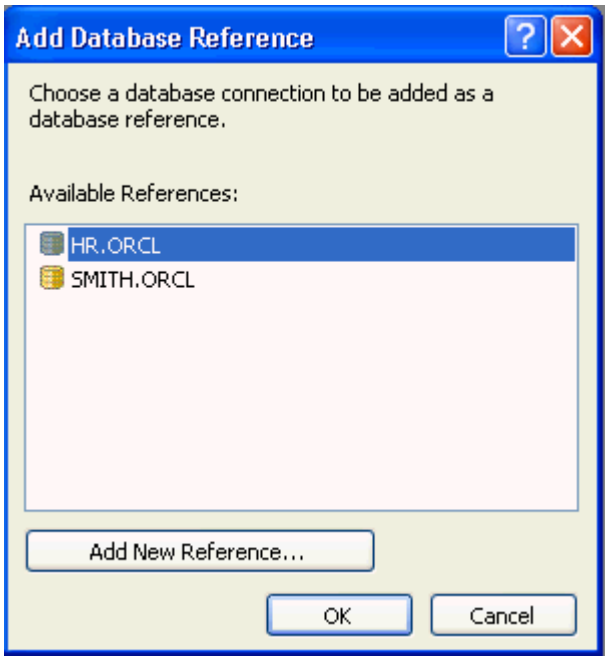
Control	Description
Connections List	Lists the following connections • Connections in database references node of the Oracle Database Project. • Connections from Server Explorer. • A temporary reference. If you select the temporary reference and select **OK**, the **Add Connection** Dialog appears
Add New Reference	Displays the Add Connection Dialog box so that you can create a new database connection. The newly added connection is listed as the last item in the displayed list of connections.
OK	Runs the selected scripts using the connection information from the selected connection. Enabled only when a connection is selected.
Cancel	Closes the dialog.

See Also

About Oracle Database Project | About Running Oracle SQL Scripts

Oracle Database Project Add Database Reference Dialog

If you right-click on the Database References node in the Oracle Database Project, and select **New Database Reference**, then the Add Database Reference dialog box launches.



The controls in Oracle Database Project Add Database Reference dialog are as follows:

Control	Description
Available References	Lists connections from Server Explorer which are not included in the Database References node.

Control	Description
Add New Reference	Displays the Add Connection Dialog box so that you can create a new database connection. The newly added connection is listed as the last item in the Available References.
OK	Adds the selected connection and connection information to the Database References node and updates the project file.
Cancel	Closes the dialog.

See Also

Database References Node | About Oracle Database Project

Queue Table Designer

The Queue Table designer creates or modifies Oracle Streams AQ Queue Tables in the database.

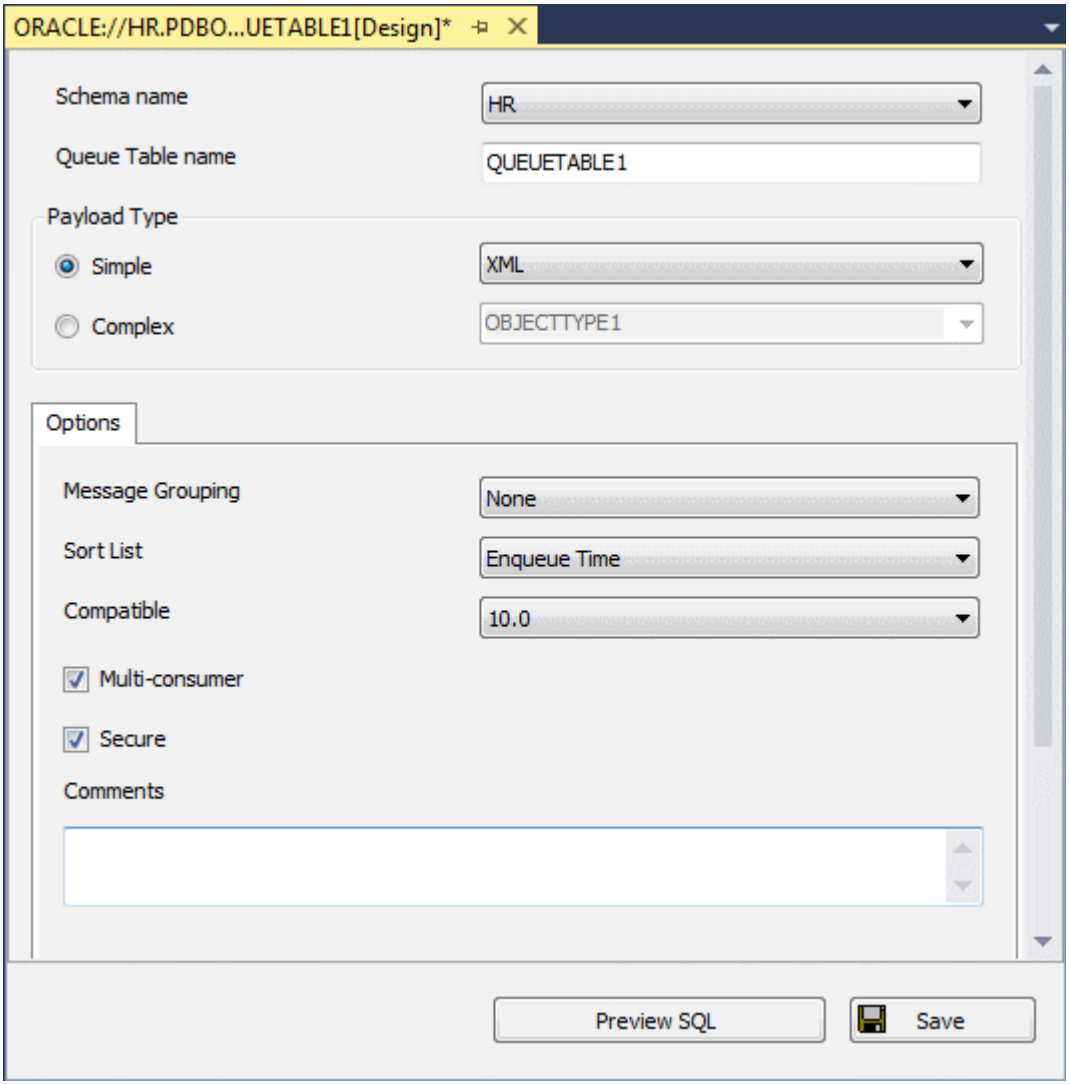
This section covers the following topics:

- Starting the Queue Table Designer
- Using the Queue Table Designer
- Main Window
- Queue Table Designer Options Tab

Starting the Queue Table Designer

Start the Queue Table Designer from Server Explorer by expanding the Advanced Queues node and selecting and right-clicking the **Queue Tables** node. Then select **New Queue Table** from the menu.

The Queue Table Designer appears:



Using the Queue Table Designer

The Queue Table Designer has the following components:

- Main Window
- Queue Table Designer Options Tab

Main Window

The main window of the Queue Designer displays identification information.

All the following controls, except Preview SQL and Save, are disabled when updating an existing queue table.

Control	Description
Schema	Select the schema. By default the current user's schema appears.

Control	Description
Queue Table Name	Enter the name of the queue table. This field cannot be empty. It has a maximum of 24 chars. Names are automatically generated for new tables, such as, QueueTable01.
Payload Type	Select the Payload Type. • Simple results in a selection list containing XML and RAW. • Complex results in a selection list containing all object types in the your schema and other included schemas for this connection. Also, you can enter the name of an object (or UDT) from another schema.
Preview SQL	Displays the SQL that creates or updates the queue if these options are saved.
Save	Creates the queue in the database or updates the existing Queue Table. In case you do not have execute permission on the `DBMS_AQADM` package, the database returns this error when trying to create the queue table: `"PLS-00201: identifier 'DBMS_AQADM' must be declared"`. A detailed error message then appears asking you to check `EXECUTE` permissions on the package and explaining how to grant them.

Queue Table Designer Options Tab

The options tab displays the basic options for the queue table. This tab appears in the previous screen shot.

When you update an existing queue table, all controls except comments are disabled.

The options controls for the Queue Table Designer are as follows:

Control	Description
Message Grouping	Select one of the following values: None (default), Transactional.
Sort List	Select one of the following: • Priority • Enqueue Time (Default) • Commit Time • Priority, Enqueue Time • Enqueue Time, Priority • Priority, Commit Time If you used one of the paired options, then the first option of the pair defines the most significant order.
Compatible	Select one of the following values: • 10.0 for databases that are 10.1 compatible.(Default) • 8.1 for databases that are 8.1 or 9.2 compatible. • 8.0 for databases that are 8.0 compatible. See Chapter 4 in *Oracle Streams Advanced Queuing User's Guide* for further information on `compatibility`.
Multiple Consumers	Specify whether the queue is multiconsumer or not. Checked is default.
Secure	Specify whether or not this queue table is secure. Checked is default. See Chapter 10 in *Oracle Streams Advanced Queuing User's Guide* for further information on creating secure Queue Tables.
Comments	Enter any comments you wish about the Queue Table.

Queue Designer

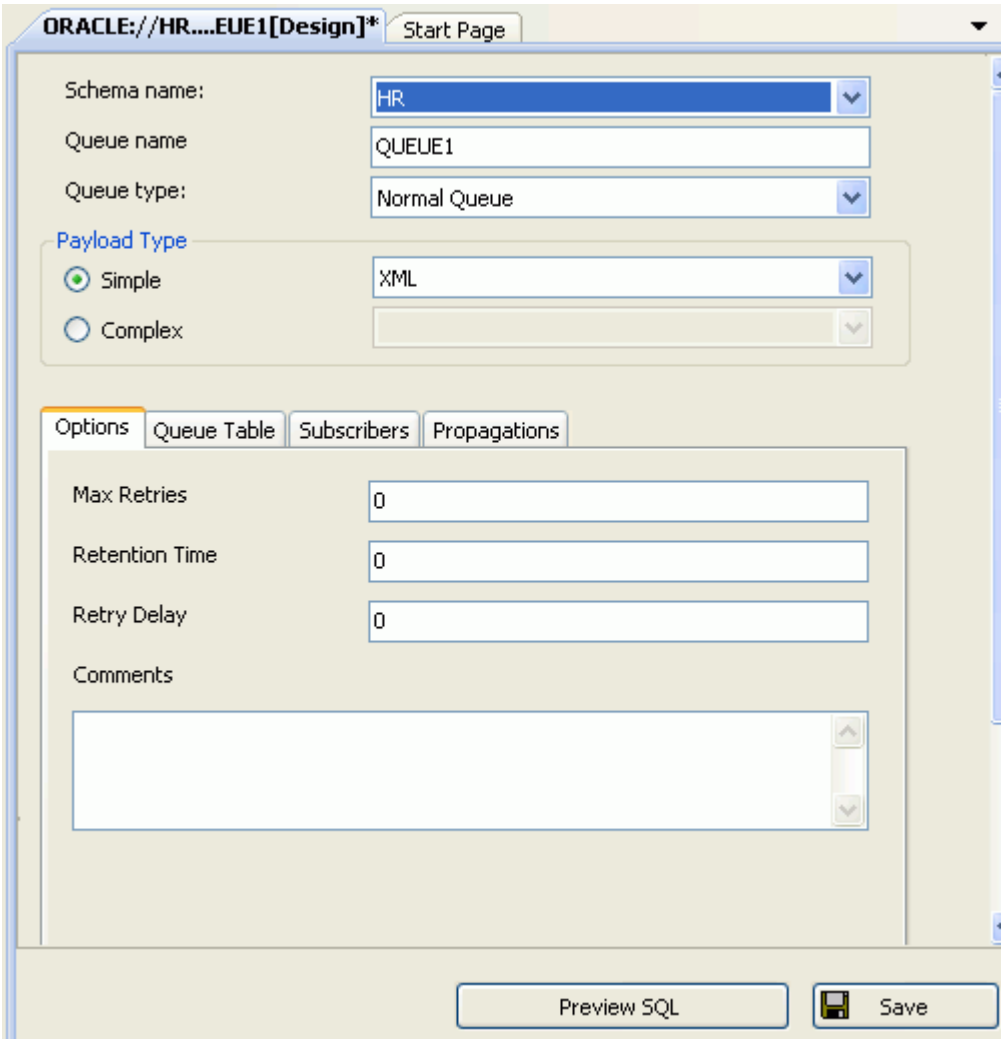
The Queue designer creates or modifies Oracle Steams AQ Queues in the database.

This section covers the following topics:

Starting the Queue Designer

Start the Queue Designer from the Server Explorer by right-clicking the Advanced Queues node and selecting the **Queues** node, then select **New Queue** from the menu.

The Queue Designer appears as follows:



Using the Queue Designer

The Queue Designer has the following components:

- Main Window
- Options Tab
- Queue Table Tab
- Subscribers Tab
- Propagation Tab

Main Window

The main window for the Queue Designer is as follows:

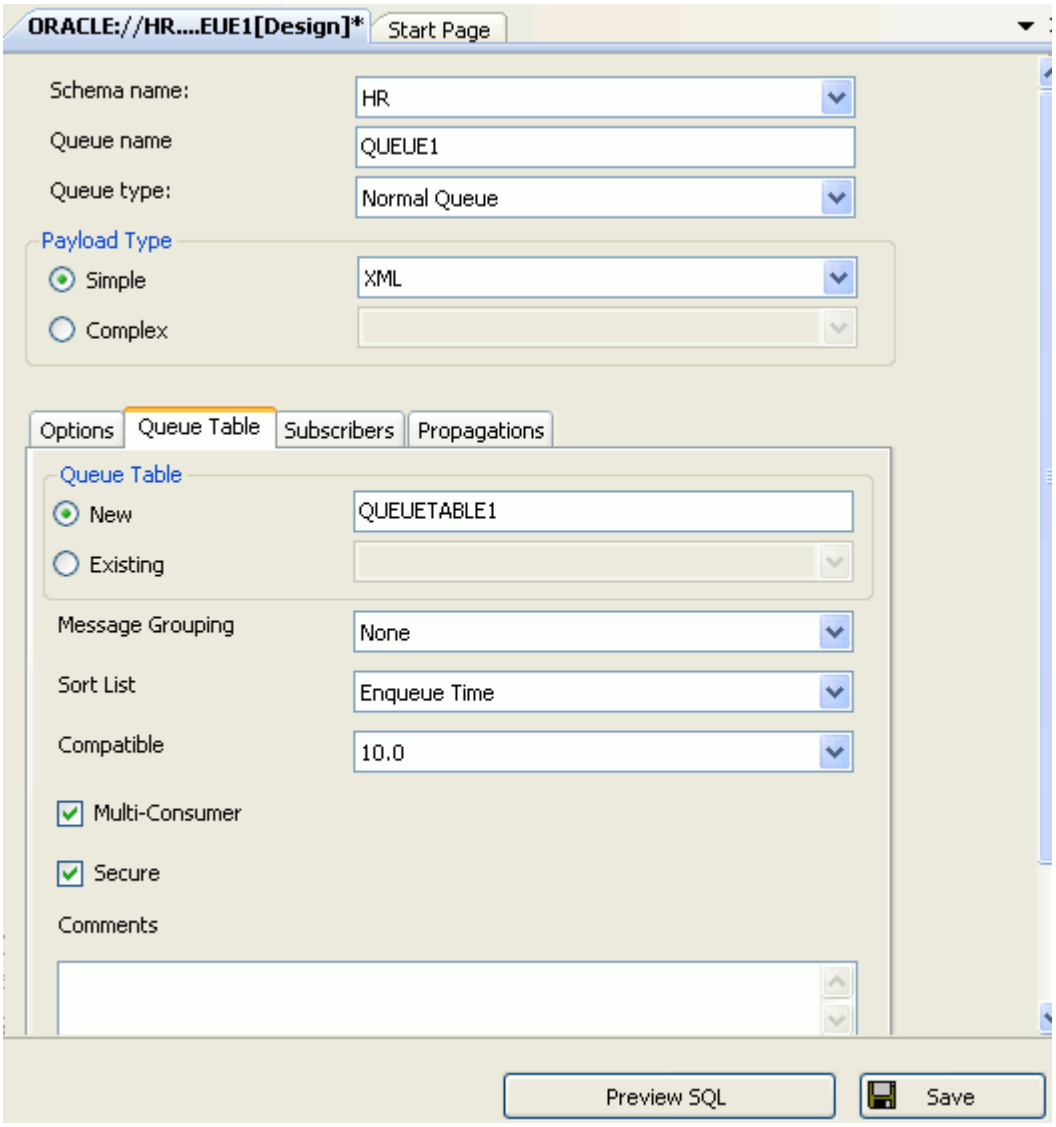
Control	Description
Schema	Select from the list of available schemas for the current connection. Your current schema is the default. This control is disabled when updating existing Queue Tables.
Name	Enter the name for the queue. Name cannot be null and is read-only when updating an existing Queue. When you create new Queue, a unique name is automatically generated, such as, Queue1.
Queue Type	Select the queue type from these values: Normal (default) or Exception. This control is read-only when updating an existing control.
Payload Type	Select the Payload Type. Choosing Simple results in a selection list containing XML and RAW. Choosing Complex results in a selection list containing all object type UDTs in the your schema and other included schemas for this connection. Also, you can enter the name of a UDT from another schema. When updating an existing queue, these controls are read-only.
Save	Creates or updates the queue in the database.
Preview SQL	Displays the SQL that creates or updates the Queue, Queue Table, Subscribers and Propagations when you save.

Options Tab

Control	Description
Max Retires	Enter the number of retries for this queue. There is a maximum value of 2147483647 and a minimum value of 0. This field can be null. Default value is empty/null. This field is disabled for 8.0 compatible queues.
Retention Time	Enter the number of seconds a message is retained in the Queue. The default value is 0, meaning no retention and a value of -1 meaning infinite retention.
Retry Delay	Specify the number of seconds before message is scheduled for processing again after an application rollback. Default is 0. Note that this control is disabled for multiconsumer queues with 8.0 compatibility.
Comments	Enter any comments you wish about the Queue.

Queue Table Tab

The Queue Table tab lets you create or edit a Queue Table associated with this queue. However, you must use the Queue Table designer for setting storage options on this Queue Table.



This tab displays Queue Table Designer options.

These controls, except for Comments, are read-only when updating an existing queue.

Control	Description
Queue Table	Select a new or existing queue table. Choosing New enables you to enter the name of the queue table to be created as part of the queue. Choosing Existing enables you to enter a queue table name or select one from a list containing the existing queue tables in your schema and included schemas. The values of the Queue Table controls change relative to the Queue Table selected.
Message Grouping	Select from the following values: None (default), Transactional.

Control	Description
Sort List	Select one of the following: • Priority • Enqueue Time (Default) • Commit Time • Priority, Enqueue Time • Enqueue Time, Priority • Priority, Commit Time If you used one of the paired options, then the first option of the pair defines the most significant order.
Compatible	Select from the following values: 8.0, 8.1, 10.0. If the database is in 10.1-compatible mode, the default value is 10.0. If the database is in 8.1-compatible or 9.2-compatible mode, the default value is 8.1. If the database is in 8.0 compatible mode, the default value is 8.0.
Secure	Specify whether or not this queue table is secure. See chapter 10 in *Oracle Streams Advanced Queuing User's Guide* for further information on creating secure Queue Tables.
Multiple Consumers	Specify whether the queue is multiconsumer or not. The default value is checked.
Comments	Enter any comments you wish about the Queue Table.

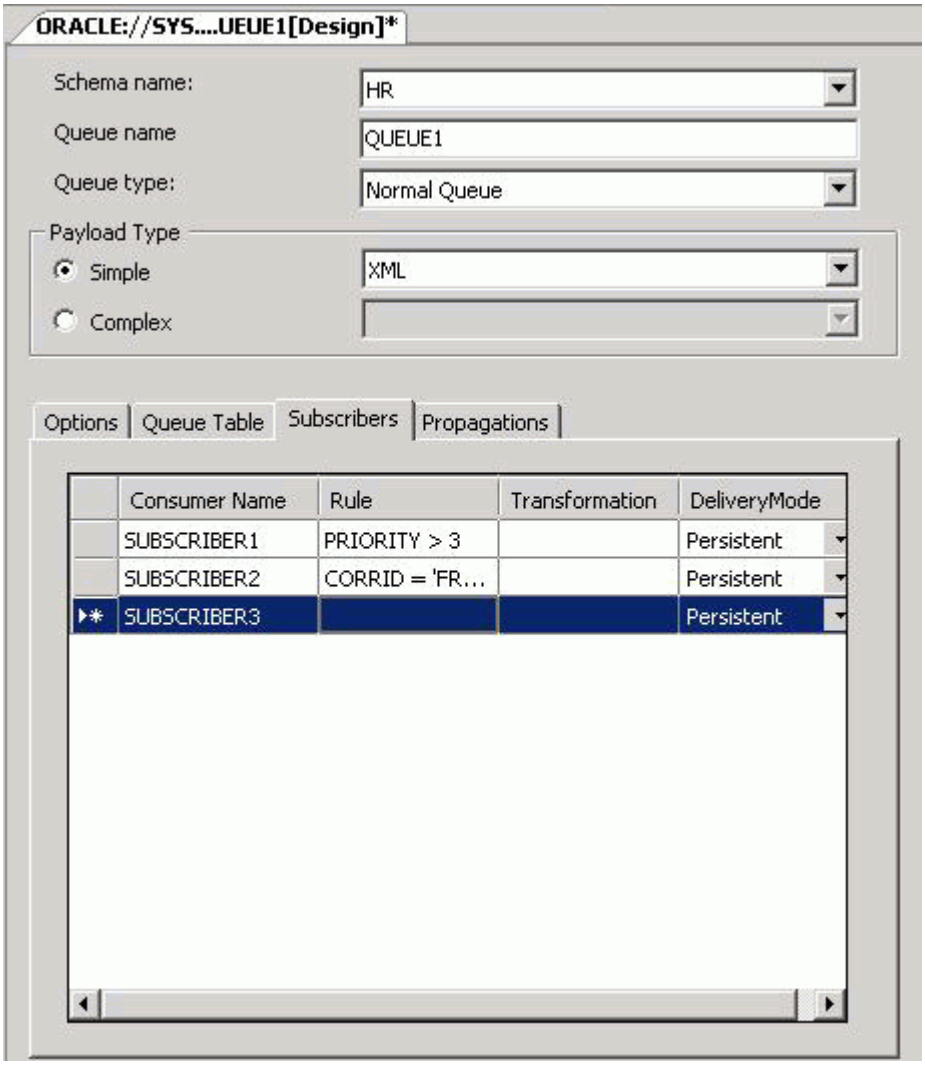
Subscribers Tab

The Subscribers tab allows you to add, delete and modify the Queue subscribers.

The tab contains a grid that lists all the current subscribers of the queue and may be used to add, update or delete the subscribers.

To add a new subscriber, click in the new record row (marked by a * in the row-header column). To delete a subscriber, select the row and press delete. To update a subscriber, click in the field and edit its value.

Only the transformation and rule fields can be modified when you update an existing queue subscriber.



This tab displays the following controls:

Control	Description
Consumer Name	Enter the name of the subscriber. This control cannot be null and is read-only when updating a subscriber.
Rule	Enter a rule for the subscriber. This should be a column that is similar to the `WHERE` clause of a SQL `SELECT` statement and may only use the `PRIORITY` and `CORRID` message properties, for example, `PRIORITY <= 3 AND CORRID = ''FROM JAPAN''`. The rule text cannot exceed 4K characters. This field is empty by default.
Transformation	Enter or select a transformation to be applied when this subscriber dequeues the message. This field is empty by default. Transformation is disabled for non-DBA connections.

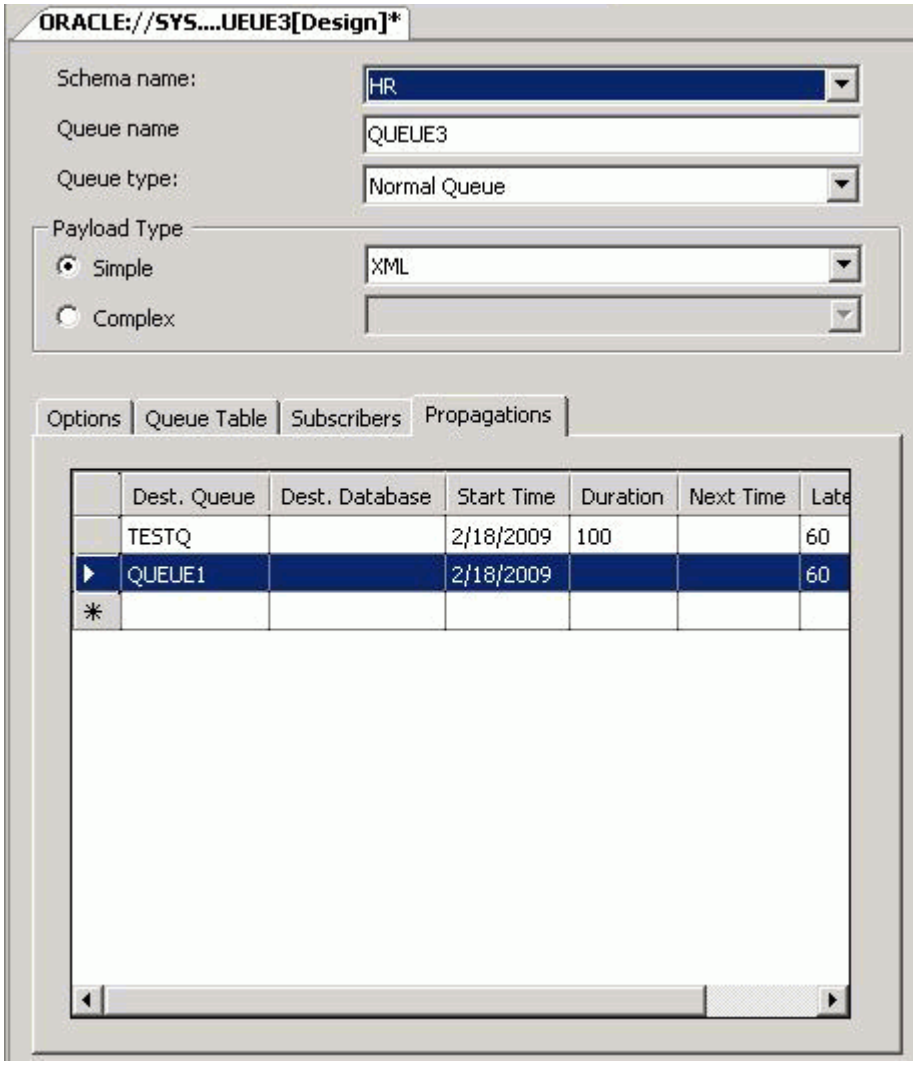
Control	Description
DeliveryMode	Select from the following values: • Persistent (Default) • Buffered • Persistent or Buffered This control is read-only when updating an existing subscriber. This control/column is only visible for Oracle Database 10*g* Release 10.2 or higher.

Propagation Tab

The Propagation tab lets you control Queue Scheduling, also known as Queue Propagation. This tab is only visible to DBA users except during queue creation. A non-DBA user can create a queue, populate this tab and press the save button. However, when the same user attempts to edit the queue, the propagation tab is no longer available.

The tab contains a grid that lists all the current propagations and supports adding, updating, or deleting propagations.

To add a new propagation, click in the new record row (marked by a * in the row-header column). To delete a propagation, select the row and press delete. To update a propagation, click in the field and edit its value.



The Propagation tab displays the following controls:

Control	Description
Dest. Queue	Select from the list or enter a destination queue. The available list contains all queues with matching payload types that you have access to, but only in the current database. Queue names from the destination database link are not fetched. This field will be empty by default and cannot be null. This field is read-only when updating an existing propagation. This control/column is only visible for Oracle Database 10*g* Release 10.2 or higher
Dest. Dababase	Select or enter a destination database link name. The available list contains all the database links defined in the database (see database view `ALL_DB_LINKS`), however you may also enter in the database link name. This field is empty by default and can be null. It is read-only when updating an existing propagation.
Start Time	Specify a Date Time for the initial start time of the propagation. The field shows the current date by default and can be null. A null value means `SYSDATE` is passed to the database as start time. It is disabled when updating an existing propagation.

Control	Description
Duration	Enter the duration of the propagation window in seconds. Duration is empty by default, meaning that the duration of the propagation window is infinite.
Next Time	Enter a date function to compute the start of next propagation window, for example, `"SYSDATE 1 - duration/86400"`. This field is empty by default and can be null.
Latency	Enter a number to specify the latency of the propagation in seconds. The default value is 60 and is used if no value is specified.

See Also

About Oracle Streams Advanced Queueing

Oracle Performance Analyzer

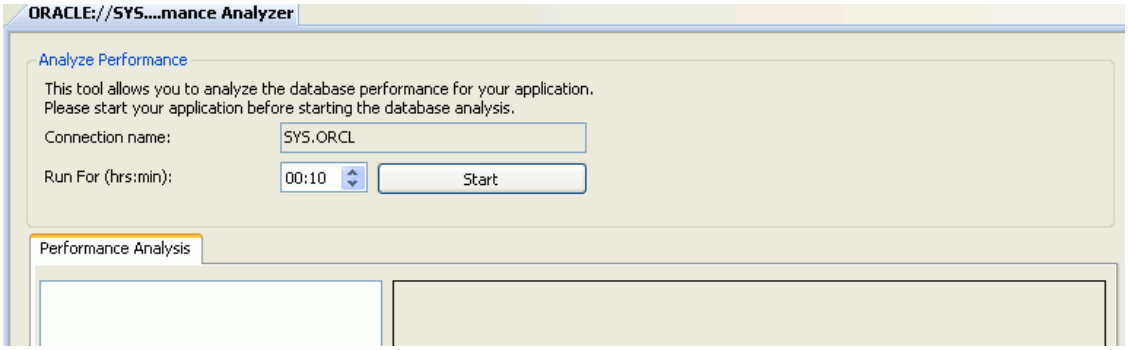
Oracle Performance Analyzer examines the use of an Oracle database over a specified period of time and provides recommendations to improve the performance of the database applications used during that time frame. In some cases, the recommendations can be automatically implemented by clicking a button.

Requirements for setting up Oracle Performance Analyzer and a walkthrough showing how to use it are located in the Performance Tuning section. See: About Oracle Performance Analyzer and Using the Oracle Performance Analyzer .

Starting the Oracle Performance Analyzer

From Server Explorer, select a SYSDBA database connection that represents the database to be tuned. Right-click, and from the menu, select the **Oracle Performance Analyzer** to launch the designer.

The analyzer appears as follows:



The top of the designer contains the following controls:

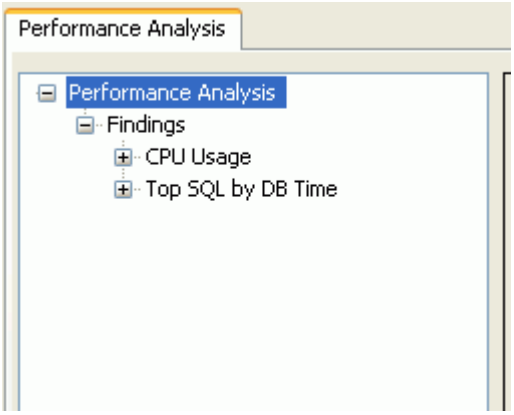
Control	Description
Connection name	Displays the connection that the analyzer was launched from.
Run for (hrs:min)	Enter the hours and minutes that the analyzer is to run. The analyzer requires least 5 minutes of run time to generate any results.

Control	Description
Start	Starts the analyzer.
Performance Analysis Tab	Displays the performance analysis tree on the left and performance analysis summary on the right after the analyzer completes.

Performance Analysis Tab

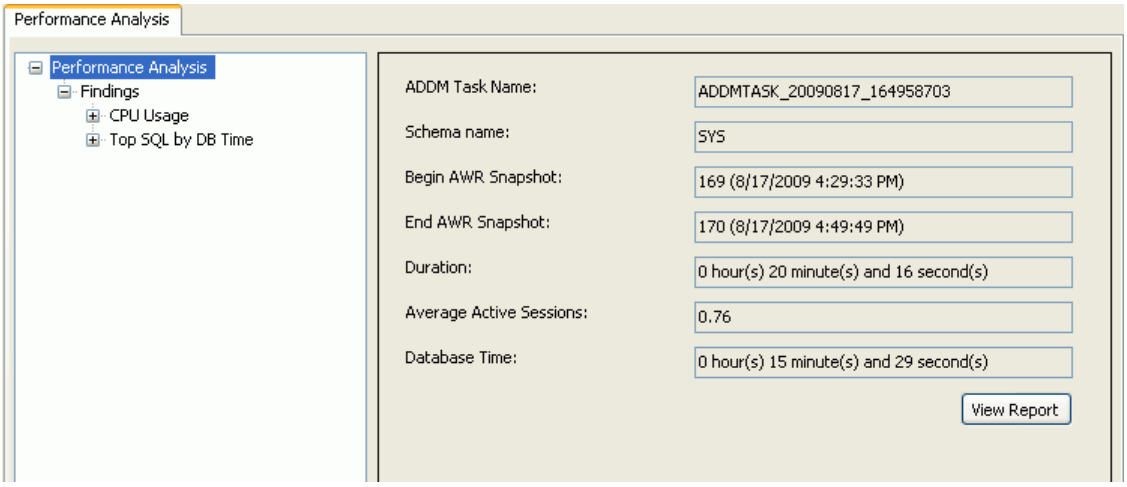
The performance analysis tab has a tree control and an informational section on the right that displays either a summary or details depending on what section of the tree you select.

The performance analysis tree appears similar to the following:



Performance Analysis Summary

When the root of the Performance Analyzer tree is selected, the completed Performance Analysis summary appears on the right, similar to the following:



The performance analysis summary controls are as follows:

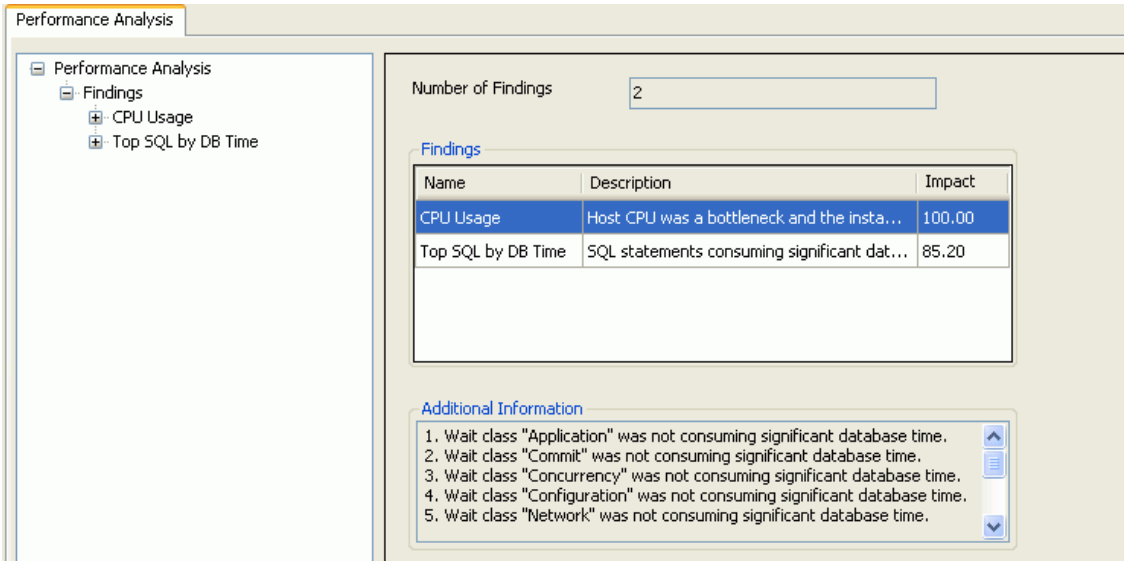
Control	Description
ADDM Task Name	Displays the ADDM Task Name.
Schema Name	Displays the schema.
Begin AWR Snapshot:	Displays the period start time.
End AWR Snapshot:	Displays the period end time
Duration (min)	Displays the length of time the analyzer has run, in minutes.
Average Active Session	Displays the average of active database sessions during this period.
Database Time	Database time is a indicator of the total database workload. It represents the total time spent in database calls.
View Report	Opens the ADDM Advisor report in a text editor which can then be modified and saved to a text file, if desired.

Findings Node

The Findings Node is a collection of nodes about findings that the Oracle Performance Analyzer generates.

For more information about ADDM findings, see section 6.1.1 ADDM Analysis, in Oracle Database Performance Tuning Guide.

When the Findings Node of the tree is selected, the Findings details appears similar to the following:



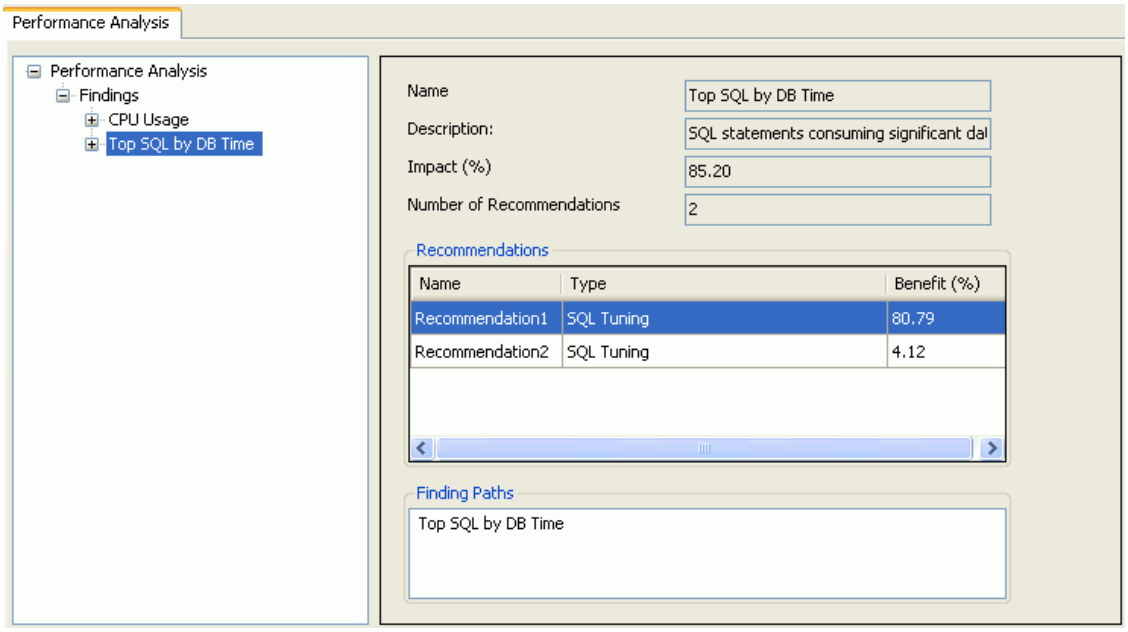
The Findings node controls are as follows:

Control	Description
Number of Findings	Displays the number of findings.
Findings	Lists the findings, their descriptions, and impact as a percent.

Control	Description
Additional Information	Contains general information and warnings pertaining to the findings. If there is insufficient database activity to successfully analyze performance, a message appears in the Additional Information section indicating this. Repeat the test with greater database activity and/or longer run time.

Individual Findings Node

When you select an individual finding node of the tree, the finding node details appear similar to the following:

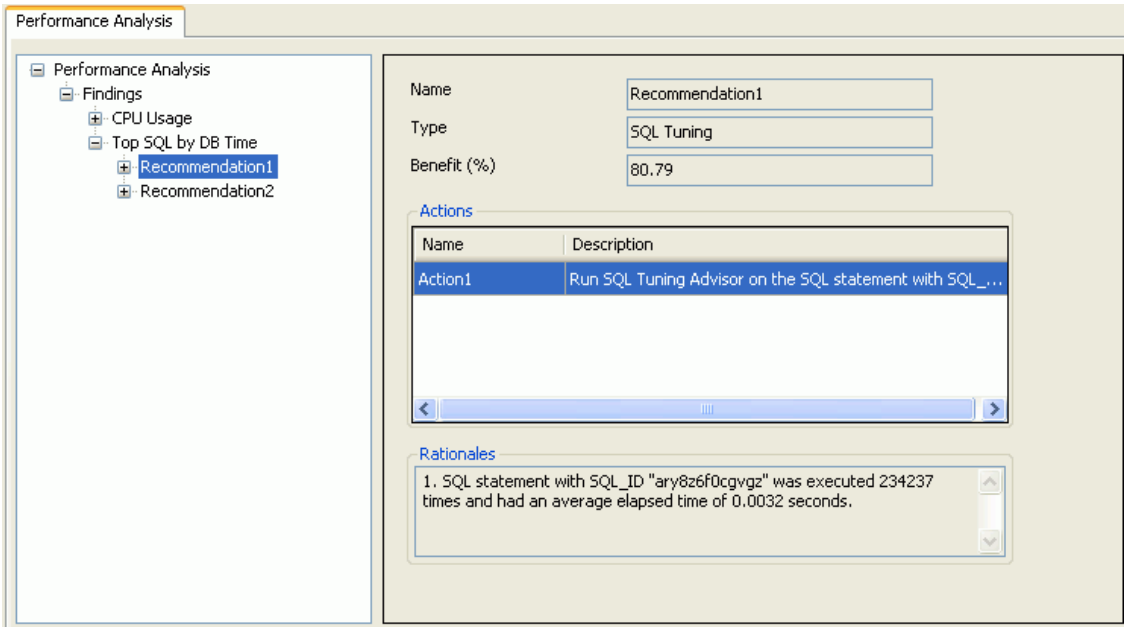


The individual findings node controls are as follows:

Control	Description
Name	Displays name of the finding selected in the Findings node of the tree.
Description	Describes the finding selected.
Impact (%)	Displays the impact of the finding selected as a percent.
Number of Recommendations	Indicates the number of recommendations for the finding selected.
Recommendations	Lists the recommendations, their types, and the estimated benefit, as a percent of `DB time` that can be saved if the recommendation is implemented.
Finding Paths	Displays the path that the Performance Analyzer used to arrive at this finding.

Recommendation Node

When the Recommendations node of the tree is selected, the Recommendation node details appears similar to the following:



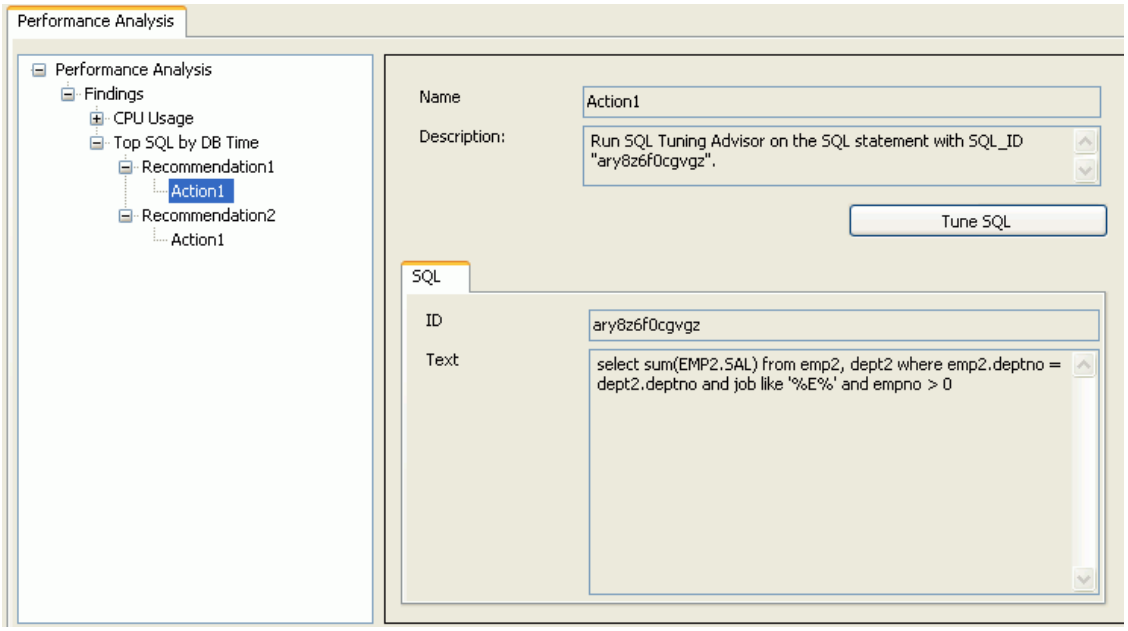
Control	Description
Name	Lists name of the recommendation.
Type	Lists recommendation type, such as SQL Tuning.
Benefit (%)	Lists the expected benefit as a percent.
Actions	Lists the potential actions with their descriptions.
Rationales	Lists the rationales for the recommendation. Rationales explain why the set of actions are recommended and provides additional information to implement the suggested recommendation.

Action Node

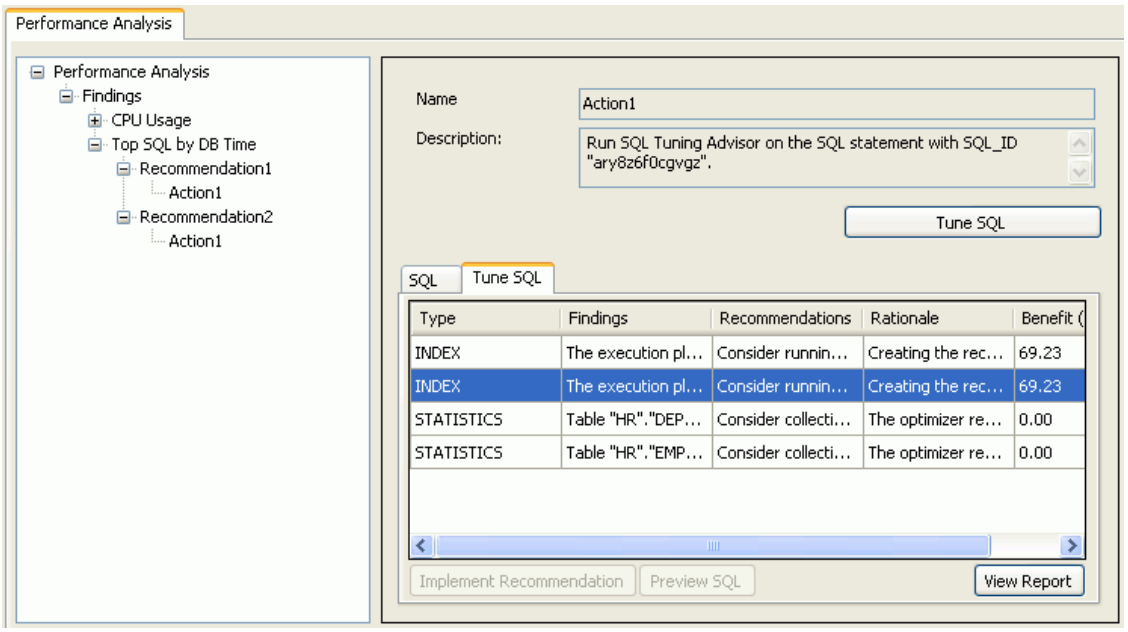
The Action node provides recommendations on how to correct a performance problem. When an Action node of the tree is selected, the following two tabs may appear, in certain scenarios.

- The SQL tab
- The Tune SQL tab

For example, an Action node for a SQL Tuning recommendation may initially appear as follows:



If Tune SQL button appears as part of a recommendation and is clicked, the Tune SQL tab appears similar to the following:



The Action node controls are as follows:

Control	Description
Action Name	Displays the name of the action selected in the tree.
Description	Displays a description of the action.

Control	Description
Tune SQL Button	Runs the SQL Tuning Advisor. This control only appears if there is a recommendation to tune the SQL. SQL Tuning may take some time to complete and during this time the button changes to Cancel. The Cancel button can be clicked to stop the tuning. You may continue to use Visual Studio or switch to other pages in the Performance Analysis tree, however, you will not be able to tune another SQL until the first one is complete or cancelled. Please note that the Cancel operation does not abort the SQL tuning right away, instead it interrupts the process and any recommendations gathered so far will be displayed in the grid.
SQL Tab	Lists the SQL ID and SQL text of the SQL statement if there is an associated SQL.
Tune SQL Tab	Lists the recommendations for the action selected. If there are no recommendations to tune the SQL, a message appears indicating this. This is a grid that contains columns for: • Type - Recommendation Type. • Findings • Recommendations • Rationale - • Benefit (%) - An estimate of the percent of `DB time` that can be saved if the recommendation is implemented.
Implement Recommendation	Implements the highlighted action. If the recommendation Type is SQL Profile and there are existing SQL Profiles, you asked to OK implementing a new SQL profile that overwrites an existing one. Implement Recommendation only appears if you select a Recommendation Type of `STATISTICS` or `SQL Profile`. It does not appear for the other recommendation types.
Preview SQL	Displays SQL for the highlighted action. Preview SQL only appears if you select a Recommendation Type of `STATISTICS` or `SQL Profile`. It does not appear for the other recommendation types.
View Report	Opens the SQL Tuning Advisor report in a text editor which can then be modified and saved to a text file, if desired.

New ADDM Task Dialog

The New ADDM Task dialog allows you to create and execute a new ADDM task.

- Starting a New ADDM Task Dialog
- Using the New ADDM Task Dialog

Creating a New ADDM Task

An ADDM task is an analysis of Oracle database performance over a period of time. Each ADDM Task node represents a single ADDM task in the database.

See Also

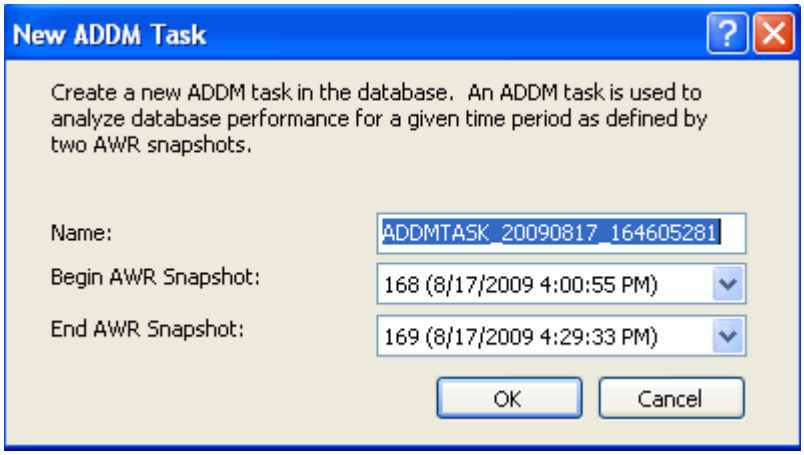
- Using the Oracle Performance Analyzer for more information about performance tuning
- *Oracle Database Performance Tuning Guide* - Chapter 5, Automatic Performance Statistics; Chapter 6, Automatic Performance and Diagnostics

Starting a New ADDM Task Dialog

To start the New ADDM Task dialog, do one of the following:

- From the ADDM Tasks node, right-click **New ADDM Task**.
- From an AWR Snapshot node, right-click **New ADDM Task**.

The New ADDM Task dialog appears:



Using the New ADDM Task Dialog

The New ADDM Task dialog controls are as follows:

Control	Description
Name	Specify the name of the new ADDM task or accept the auto-generated name. Maximum length is 30 characters.
Begin AWR Snapshot	Select the begin snapshot ID for the ADDM task from the list. The list contains all the snapshot IDs in the database except the latest one. The snapshot timestamp appears in parenthesis.
End AWR Snapshot	Select the end snapshot ID for the ADDM task from the list. The list contains all the snapshot IDs in the database except the oldest one. The snapshot timestamp appears in parenthesis.
OK	Closes the dialog and creates and executes the ADDM task. Oracle Performance Analyzer launches and displays the execution results. An error message appears if you select an End Snapshot ID that is equal or older than the Begin snapshot ID.

Control	Description
Cancel	Closes the dialog and no actions are taken.

New AWR Snapshot Dialog

The New AWR Snapshot dialog allows you to create a new AWR snapshot.

- Creating a New AWR Snapshot
- Starting the New AWR Snapshot Dialog
- Using the New AWR Snapshot Dialog

Creating a New AWR Snapshot

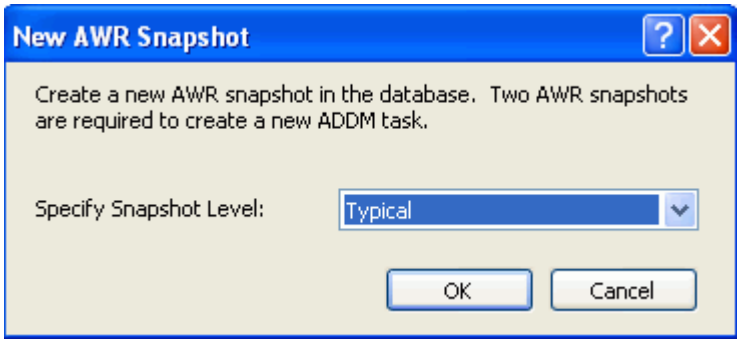
An AWR Snapshot is a collection of database statistics and performance metrics gathered at a single point in time. Two snapshots are required to analyze database performance over their time period. This performance analysis is done by creating an ADDM Task.

See Also

- Using the Oracle Performance Analyzer for more information about performance tuning
- *Oracle Database Performance Tuning Guide* - Chapter 5, Automatic Performance Statistics; Chapter 6, Automatic Performance and Diagnostics

Starting the New AWR Snapshot Dialog

To start the New AWR Snapshot dialog, right-click on AWR Snapshots Node and select New AWR Snapshot from the menu.



When the New AWR Snapshot dialog appears, specify `Typical` or `All` from the list as the Snapshot Level. Snapshot Level determines the amount of performance statistics that are gathered. This can be set to either the default value of `TYPICAL` or a value of `ALL`. When the statistics level is set to `ALL`, the AWR gathers the maximum amount of performance data including additional statistics such as timed OS statistics and plan execution statistics. Note that choosing `ALL` increases the database workload required to create the snapshot.

See Also

Oracle Database Performance Tuning Guide

Using the New AWR Snapshot Dialog

The New AWR Snapshot dialog controls are as follows:

Control	Description
Specify Snapshot Level	Possible values are: `TYPICAL` (default) - This ensures collection of all major statistics required for database self-management functionality and provides the best overall performance. This setting should be adequate for most environments. `ALL` - This includes the statistics collected with the `TYPICAL` setting and additional statistics including timed OS statistics and plan execution statistics.
OK	Closes the dialog and creates the AWR snapshot.
Cancel	Closes the dialog without performing any actions.

Schema Compare Source and Target Dialog

In the Schema Compare Source and Target Dialog, you can select the source and target database connections or Oracle Database Project version 2 projects for a schema comparison. This dialog also allows you to filter results. When you press OK, the Schema Compare Results Window opens to display the results.

See Also

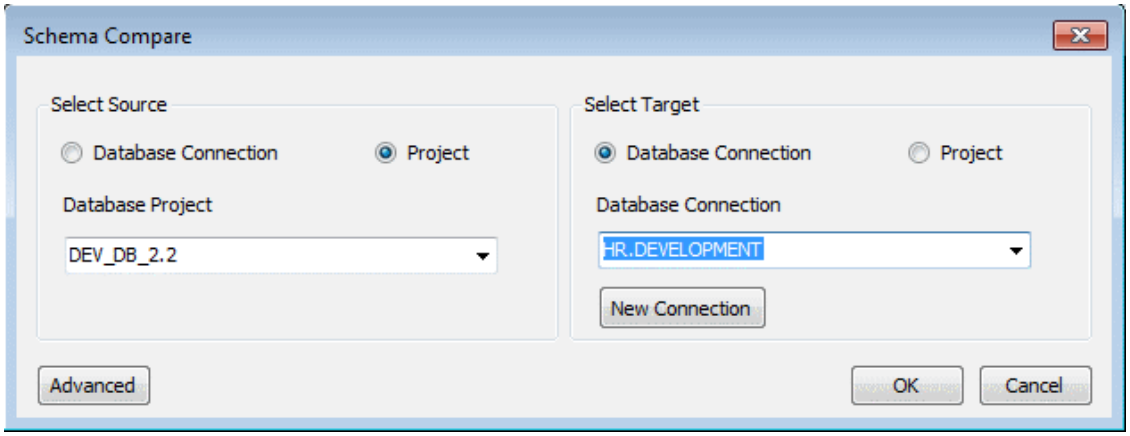
- About Schema Compare
- Schema Compare Results Window

Opening the Schema Compare Source and Target Dialog

You can open the Schema Compare Source and Target Dialog from the Visual Studio Tools menu - Oracle Schema Compare menu item, the context menu for Data Connection nodes described in Menu Options, or the Oracle Database Project Version 2 context menu.

Using Schema Compare Source and Target Dialog

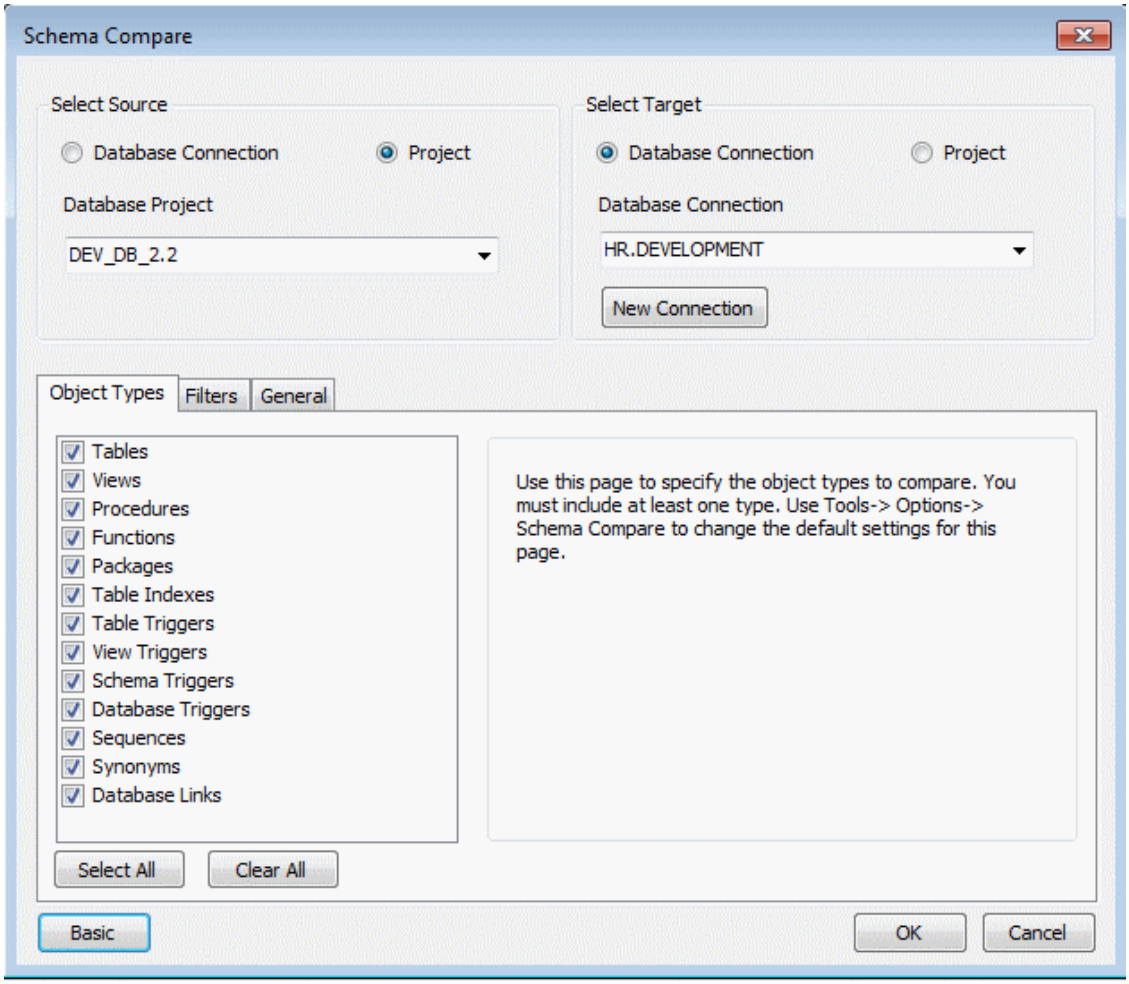
The Schema Compare Source and Target Dialog looks similar to this:



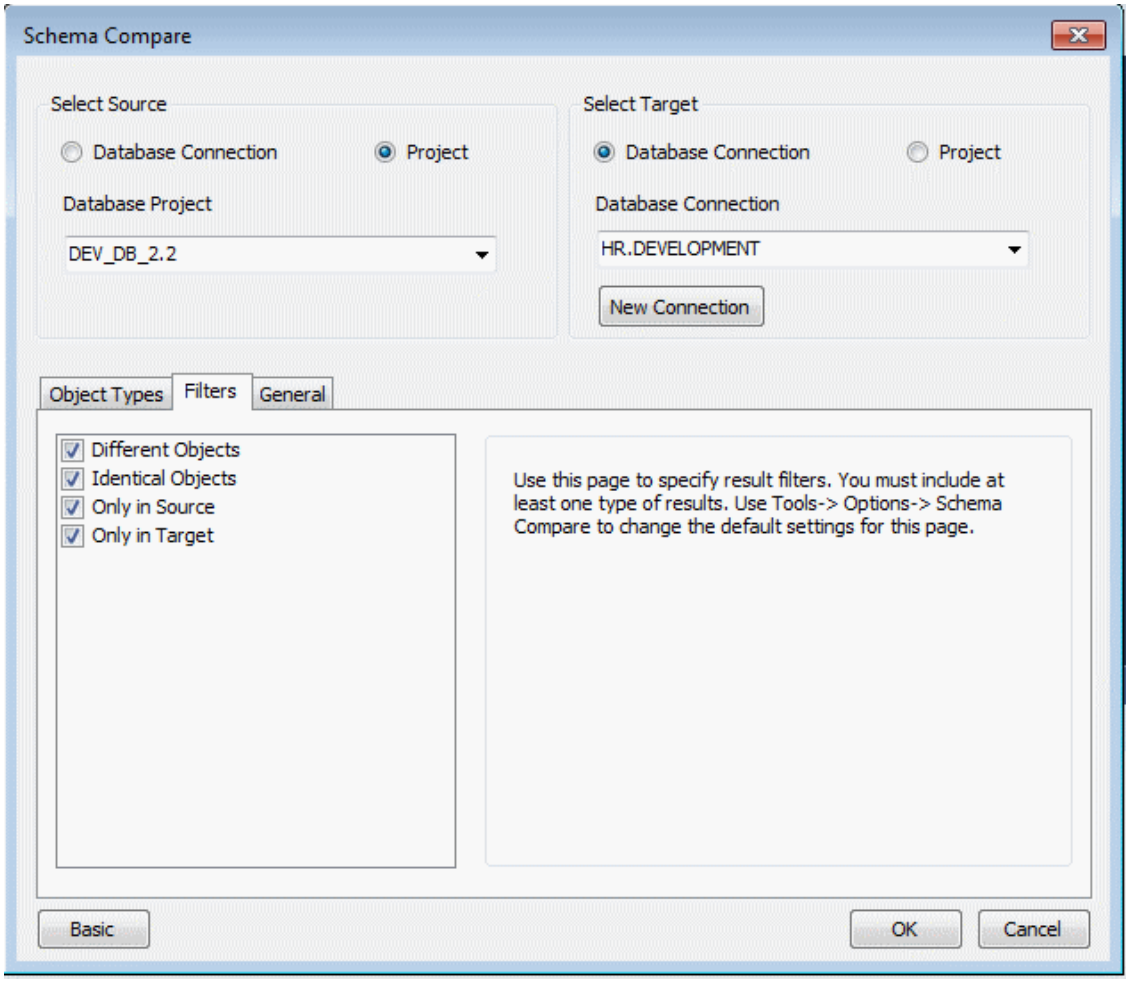
The controls in the basic Schema Compare Source and Target Dialog are as follows:

Control	Description
Source Database Connection/ Project radio button	Selects whether the Source is a Visual Studio Server Explorer Oracle Database connections or an existing Oracle Database Project Version 2 project.
Target Database Connection/ Project radio button	Selects whether the Target is a Visual Studio Server Explorer Oracle Database connections or an existing Oracle Database Project Version 2 project.
Source Database Connection	Select a source from the existing Visual Studio Server Explorer Oracle Database connections for the schema comparison.
Target Database Connection	Select a target from the existing Visual Studio Server Explorer Oracle Database connections for the schema comparison.
Source Database Project	Select a source from an existing Oracle Database Project Version 2 project for the schema comparison.
Target Database Project	Select a target from an existing Oracle Database Project Version 2 project for the schema comparison.
New Connection	Create a new Oracle Database connection to serve as the source or target for the schema comparison by opening the Connection Dialog Box .
OK	Launches the Schema Compare Results Window to display the results of the schema comparison.
Cancel	Closes the dialog without performing a schema comparison.
Advanced	Provides two additional tabs for filtering the results of the schema comparison: Schema Compare Object Types and Schema Compare Filters.

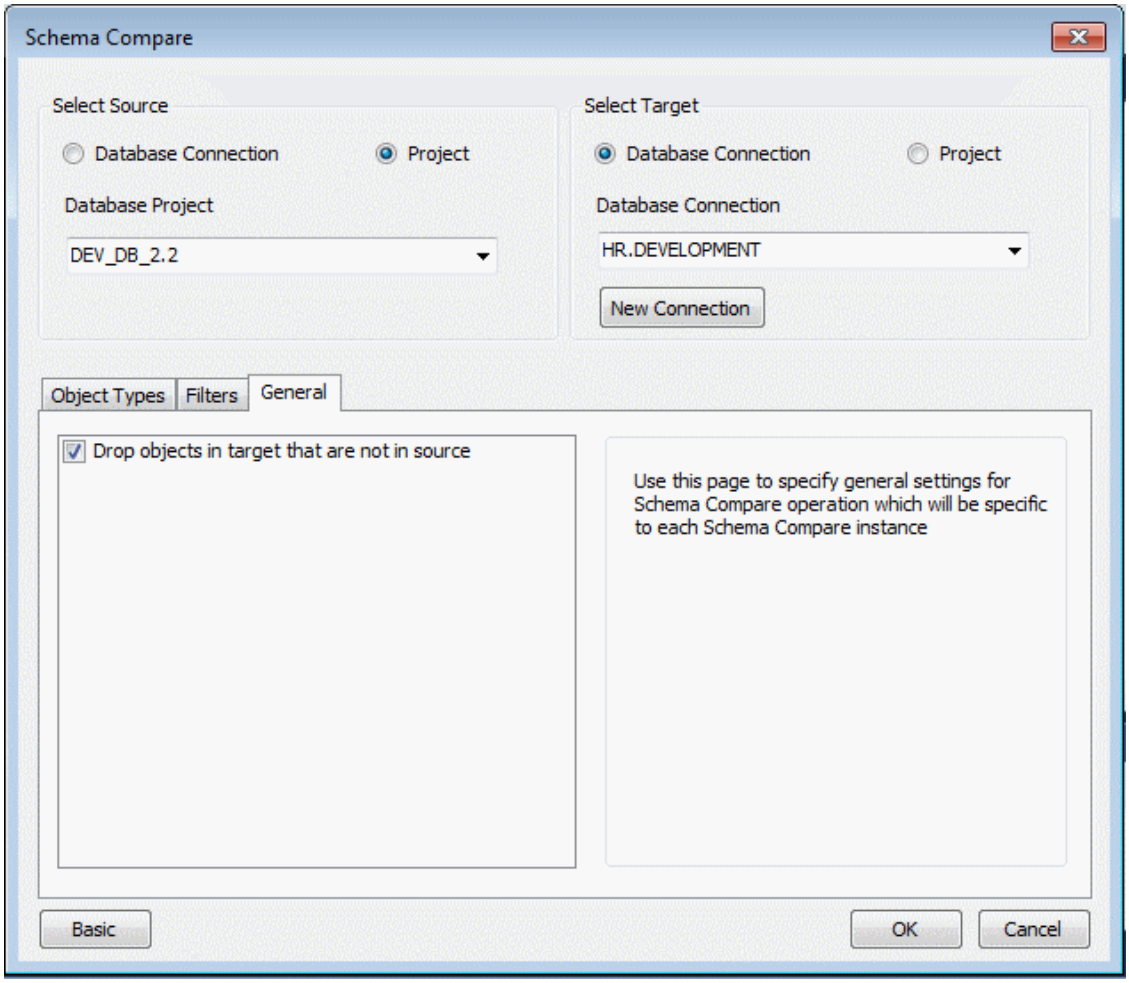
Schema Compare Object Types



Schema Compare Filters



Schema Compare General Options



The additional controls for Object types and Filters in the advanced Schema Compare Source and Target Dialog are as follows:

Control	Description
Object Types tab	Select the schema object types to compare. Default values can be set in the Options for Schema Compare Object Types.
Filters tab	Select the results types you wish to see. Default values can be set in the Options for Schema Compare Filters.
(General Tab) Drop objects in target that are not in source	Drops objects in target that are not in the source.
Select All	Selects all options in the tab.
Clear All	Deselects all options in the tab (except for one).
Basic button	This button closes the Advanced pane.

Schema Compare Results Window

This window displays the results of a schema comparison in several ways:

- Tree control summary format
- Text grid details view (Attributes tab)
- Object Definition changes view (Object Definition tab)
- SQL difference view (Update SQL tab)

The top portion of this window also includes several toolbar buttons.

See Also

- About Schema Compare
- Schema Compare Source and Target Dialog

Opening the Schema Compare Results Window

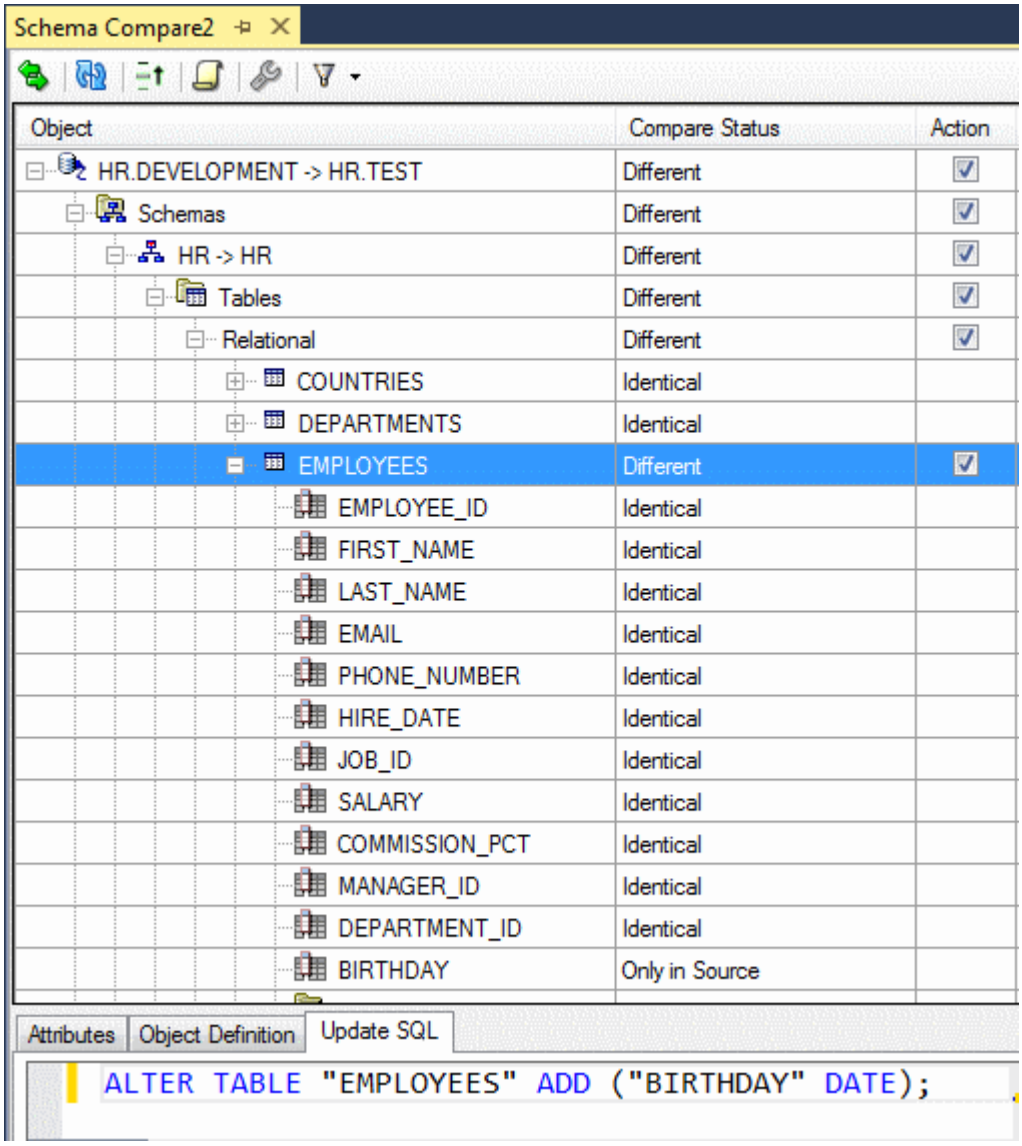
After you provide the Schema Compare Source and Target Dialog with the source and target database connections or Oracle Database Project Version 2 database projects, the Schema Compare Results Window opens. You can open the Schema Compare Source and Target Dialog from the Visual Studio Tools menu, from the Project node in a Oracle Database Project Version 2, or the context menu for a Server Explorer Data Connection node, described in Menu Options.

See Also

- Data Connection Nodes
- Oracle Database Project Version 2

Using the Schema Compare Results Window

The Schema Compare Results window appears similar to the following:



The controls in the Schema Compare Results Window are as follows:

Control	Description
Source to Target toolbar button	Switches the source and target database schemas.
Refresh toolbar button	Clears the results and re-runs the schema compare.
Update Target toolbar button	This will update a target Database Connection or Oracle Database Project Version 2 project so that the target schema will match the source schema.
Generate Script toolbar button	This will generate a deployment script (a *diff* script) and open it in an SQL and PL/SQL File Editor associated with the target Oracle Database connection. The script can be saved as a file and can be run against the target database to upgrade the target schema to match the source schema. This window contains an Execute Query toolbar icon that will run selected SQL against the Oracle Database Connection. Note: Generate Script button is disabled when target is type Oracle Database Project Version 2.

Control	Description
Options button	The options button reopens the Schema Compare Source and Target Dialog with the settings selected when Schema Compare was previously launched. Clicking the OK button on the dialog clears the Schema Compare Results Window and re-runs the schema comparison.
Filter button	The Filter button opens a drop-down list allowing you to choose the filters that control the type of results displayed. The filter takes effect immediately after you make a selection.
Difference hierarchy tree	Displays various levels of the schema hierarchy, indicating differences between the source and target at various points in the hierarchy. You can expand nodes and drill down until you get down to individual schema objects.
Action checkbox	Schema objects or schema object types that have the Action checkbox selected will be included when the Update Target toolbar button or the Generate Script toolbar button is used. The topmost checkbox can be used to select or de-select all checkboxes.
Update SQL tab	Shows the SQL needed to modify the target schema to match the source schema for a selected schema object or any of its child nodes, for example, a table or any of its columns. For higher level nodes in the schema comparison hierarchy, this tab will be blank.
Attributes tab	Displays a grid showing differences between source and target schema objects. For example, if a table column has been changed from a type NUMBER(5,1) to a NUMBER(7,1), you can see this by clicking on the column node in the differences hierarchy and then viewing the Attributes tab. For higher level nodes in the schema comparison hierarchy, this tab will be blank.
Object Definition tab	Displays the difference between the source and target for the selected schema object, using SQL code that might be used to create the object (such as, `CREATE TABLE`). This tab is split vertically, showing both the source and target SQL with the mismatched lines highlighted. For higher level nodes in the schema comparison hierarchy, this tab will be blank.

New Pluggable Database Dialog

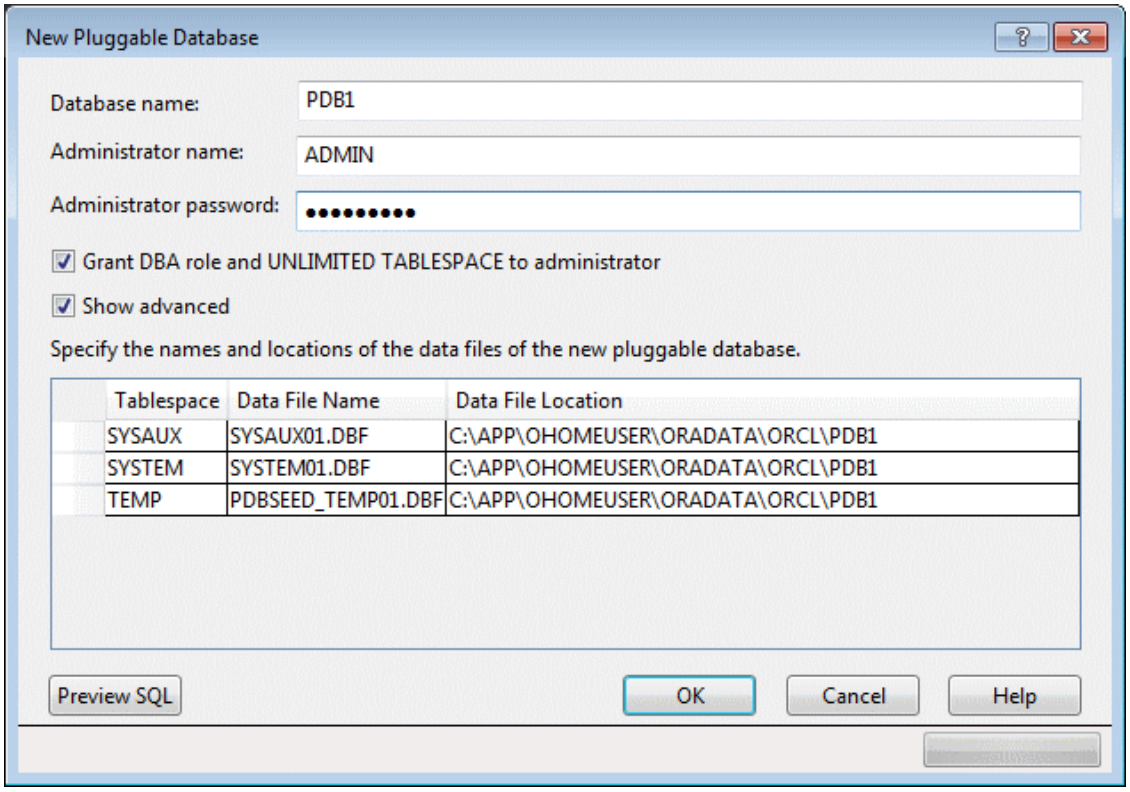
The New Pluggable Database Dialog allows you to create a new pluggable database, which is added to the multitenant container database. New Oracle Database files (DBF files) are added to the database server filesystem. Afterwards, if the option is chosen in the General Options Page, an administrator connection is automatically made to the new pluggable database in Server Explorer, and if the connection to the container database is using a TNS alias, a connection alias is created for it in the `tnsnames.ora` file.

> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

Starting the New Pluggable Database Dialog

To open the New Pluggable Database Dialog, connect to Server Explorer with a user that has SYSDBA privileges. Right-click the Pluggable Databases Node and choose **New Pluggable Database**.

The New Pluggable Database Dialog appears as follows:



Using the New Pluggable Database Dialog

The New Pluggable Database Dialog has the following controls:

Control	Description
Database name	Enter the name of the new pluggable database to be created
Administrator name	Enter the user name of the Administrator account for the pluggable database to be created.
Administrator password	Enter the password for the Administrator account for the pluggable database to be created.
Grant DBA role and UNLIMITED TABLESPACE to administrator	Grants the DBA role and UNLIMITED TABLESPACE privilege to the administrator account.
Show Advanced	Enable this check box to display additional fields, allowing you to specify the data file names and locations for the new pluggable database.

Control	Description
Specify the names and locations of the datafiles of the new pluggable database	Lists new pluggable database datafile names and their locations on the database server filesystem. Both the Data File Name and Data File Location field can be modified as desired.
Preview SQL	View the SQL to be executed with the OK button is pressed.
OK	Creates and opens the pluggable database and then closes this dialog. If the option is chosen in the General Options Page, it also makes a connection to it in Server Explorer as the administrator account and adds an entry to the `tnsnames.ora` file (if the connection to the container database used a TNS alias).
Cancel	Closes this dialog without making any changes to the database.
Help	Opens this help page.

Clone Pluggable Database Dialog

The Clone Pluggable Database Dialog allows you to make a clone of an existing pluggable database. The cloned pluggable database is added to the multitenant container database. New Oracle Database files (DBF files) are added to the database server filesystem. After the clone operation is complete, if the option is chosen in the General Options Page, an administrator connection is automatically made to the new pluggable database in Server Explorer, and if the connection to the container database is using a TNS alias, a connection alias is created for it in the `tnsnames.ora` file.

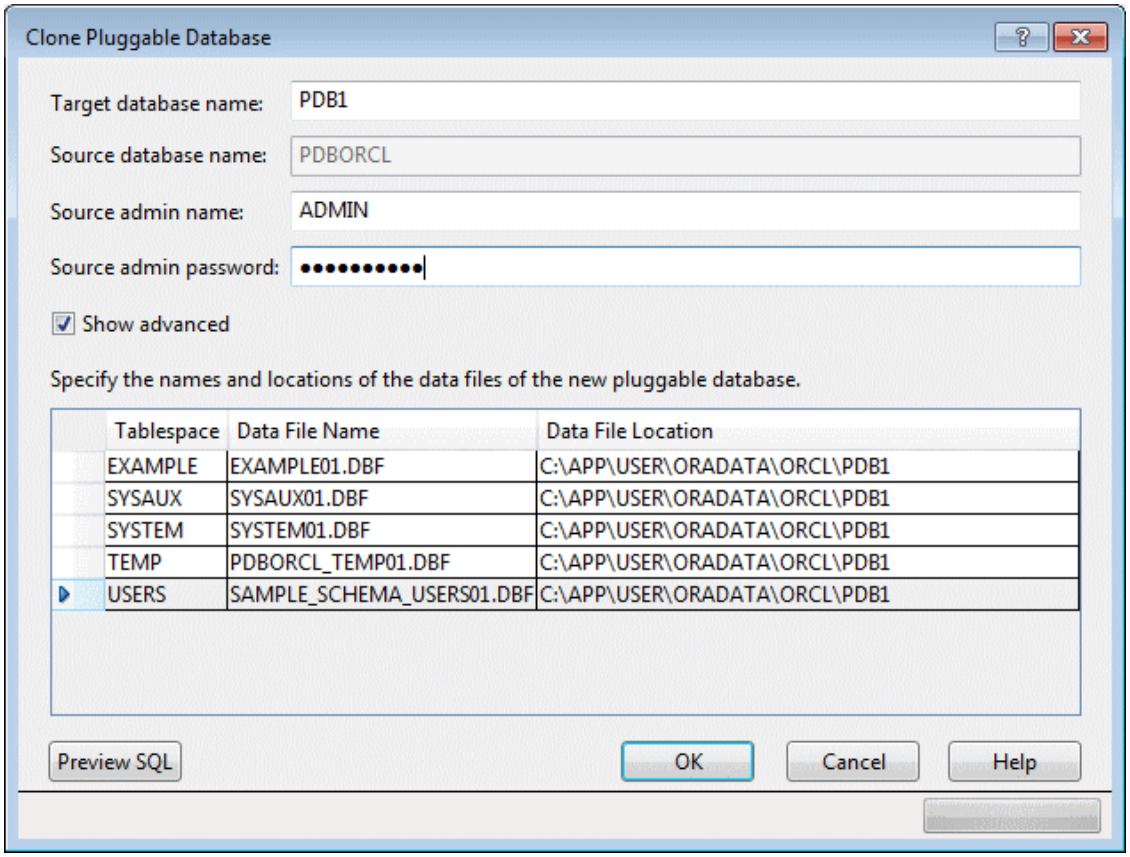
> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

Starting the Clone Pluggable Database Dialog

To open the Clone Pluggable Database Dialog, connect to Server Explorer with a user that has `SYSDBA` privileges. Expand the Pluggable Databases Node and view the list of Pluggable Database Nodes. When you have located the pluggable database that you wish to clone, right-click on it, and choose **Clone**. A confirmation dialog appears, informing you that the pluggable database will be set to Read Only and that all existing connections to it will be lost.

> **Note**
>
> The seed pluggable database cannot be cloned using this dialog. Instead, click on the Pluggable Databases Node and choose **New Pluggable Database**.

The Clone Pluggable Database Dialog appears as follows:



Using the Clone Pluggable Database Dialog

The Clone Pluggable Database Dialog has the following controls:

Control	Description
Target database name	Enter name of the new pluggable database to be created.
Source database name	Displays the name of the pluggable database to be cloned. This field cannot be changed.
Source Admin Name	Enter the user name of the Administrator account for the pluggable database that is to be cloned.
Source Admin password	Enter the password for the Administrator account for the pluggable database that is to be cloned.
Show Advanced	Check this check box to display additional fields, allowing you to specify the data file names and locations for the new pluggable database.
Specify the names and locations of the datafiles of the new pluggable database	Lists new pluggable database datafile names and their locations on the database server filesystem. Both the Data File Name and Data File Location field can be modified as desired.
Preview SQL	View the SQL to be executed when you press the OK button.

Control	Description
OK	Clones the pluggable database, opens the clone database, returns the source database to its original state, and then closes this dialog. If the option is chosen in the General Options Page, it also makes a connection to it in Server Explorer as the administrator account and adds an entry to the `tnsnames.ora` file (if the connection to the container database used a TNS alias).
Cancel	Returns the source database to its original state and then closes this dialog without making any other changes to the database.
Help	Opens this help page.

Plug Pluggable Database Dialog

The Plug Pluggable Database Dialog allows you to plug in a pluggable database. A pluggable database is represented by an XML file plus some Oracle Database files (DBF files). These files must reside on the database server filesystem.

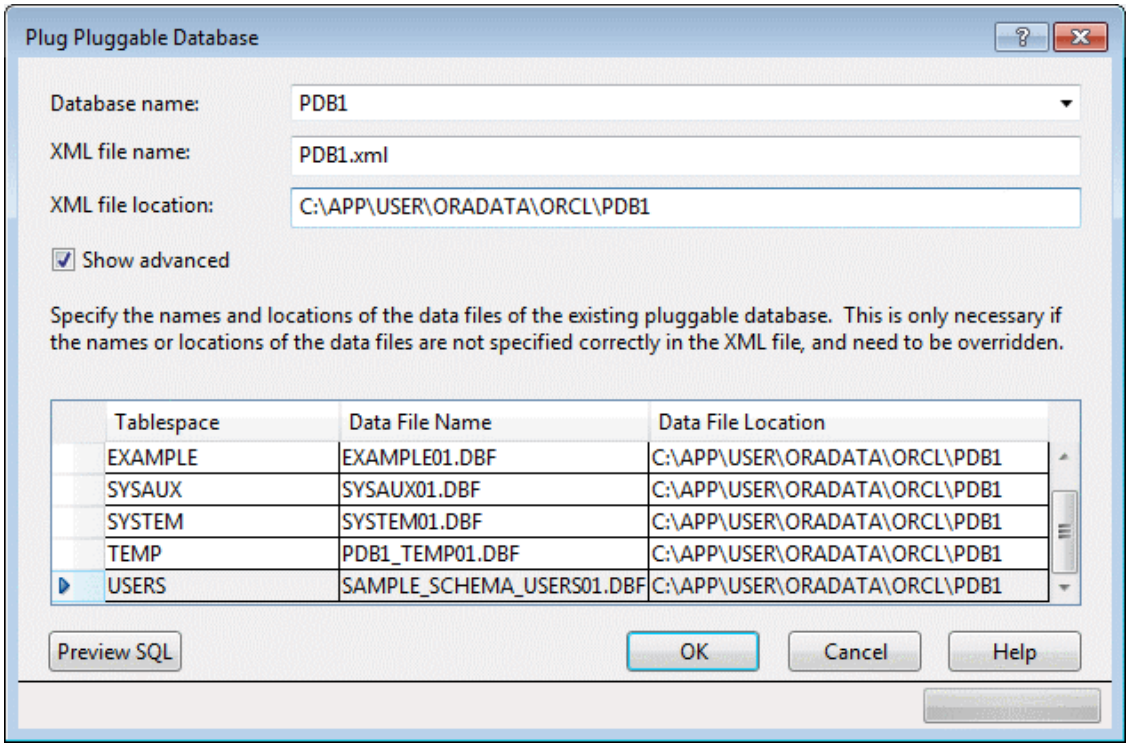
> **Note**
>
> The container database must use the same character set as the pluggable database for the Plug operation to be successful.

> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

Starting the Plug Pluggable Database Dialog

To open the Plug Pluggable Database Dialog, connect to Server Explorer with a user that has `SYSDBA` privileges. Right-click on the Pluggable Databases Node and choose **Plug**.

The Plug Pluggable Database Dialog appears as follows:



Using the Plug Pluggable Database Dialog

The Plug Pluggable Database Dialog has the following controls:

Control	Description
Database name	Select the name of the pluggable database to be plugged in.
XML File Name	Enter the name of the XML file that contains the metadata for the pluggable database.
XML File Location	Enter the path to the XML file that contains the metadata for the pluggable database on the database server filesystem.
Show Advanced	Check this check box to display additional fields, allowing you to specify the data file names and locations for the new pluggable database.
Specify the names and locations of the datafiles of the existing pluggable database	Lists pluggable database datafile names and their locations on the database server filesystem. Both the Data File Name and Data File Location field can be modified as desired.
Preview SQL	View the SQL to be executed when the OK button is pressed.
OK	Plugs in and opens the pluggable database and closes this dialog.
Cancel	Closes this dialog without making any changes to the database.
Help	Opens this help page.

Unplug Pluggable Database Dialog

The Unplug Pluggable Database Dialog allows you to unplug a pluggable database. This closes the pluggable database, removes it from the multitenant container database and creates an XML metadata file. This metadata file and the corresponding Oracle Database files

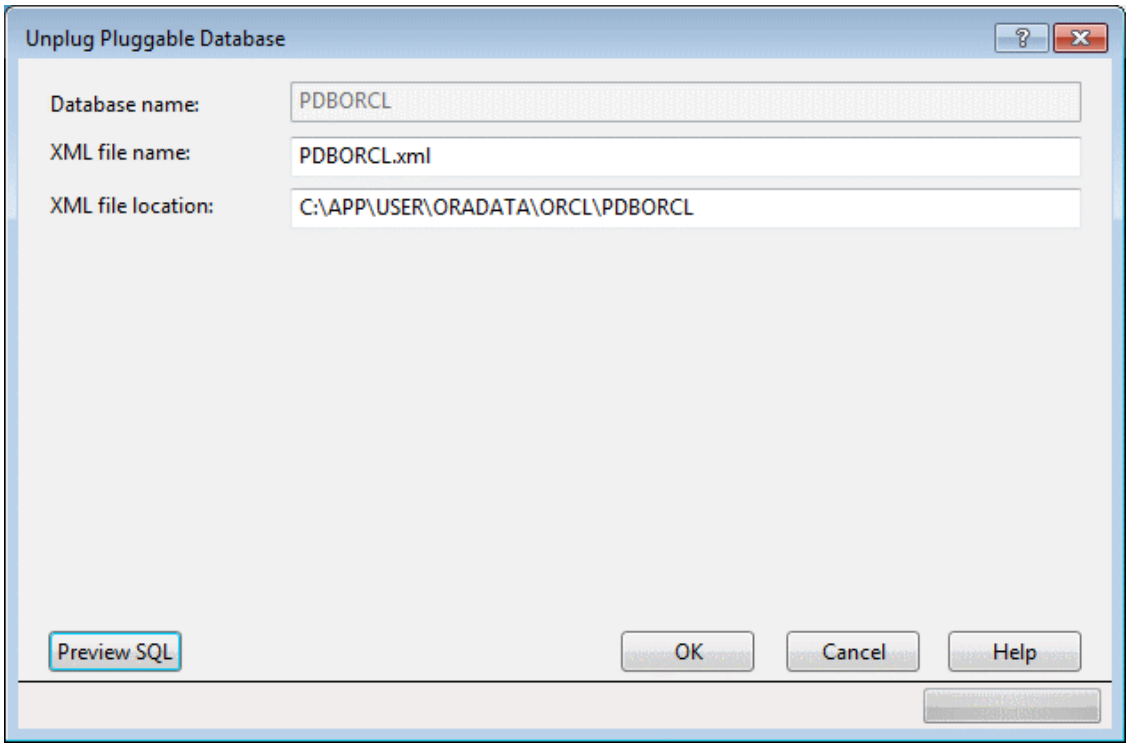
(DBF files) remain on the database server filesystem and can be plugged back into the same multitenant container database at a later time or plugged into other container databases that use the same character set as the pluggable database. In most cases, unplugging a pluggable database is preferred rather than permanently deleting it.

> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

Starting the Unplug Pluggable Database Dialog

To open the Unplug Pluggable Database Dialog, connect to Server Explorer with a user that has SYSDBA privileges. Expand the Pluggable Databases Node and view the list of Pluggable Database Nodes. When you have located the pluggable database that you wish to unplug, right-click on it and choose **Unplug**.

The Unplug Pluggable Database Dialog appears as follows:



Using the Unplug Pluggable Database Dialog

The Unplug Pluggable Database Dialog has the following controls:

Control	Description
Database name	Displays the name of the pluggable database to be unplugged. This field cannot be modified.

Control	Description
XML File Name	Displays name of the XML file that contains the metadata for the pluggable database.
XML File Location	Displays the path on the database server filesystem to the XML file that contains the metadata for the pluggable database.
Preview SQL	View the SQL to be executed with the OK button is pressed.
OK	Closes and unplugs the pluggable database and closes this dialog.
Cancel	Closes this dialog without making any changes to the database.
Help	Opens this help page.

Open Pluggable Database Dialog

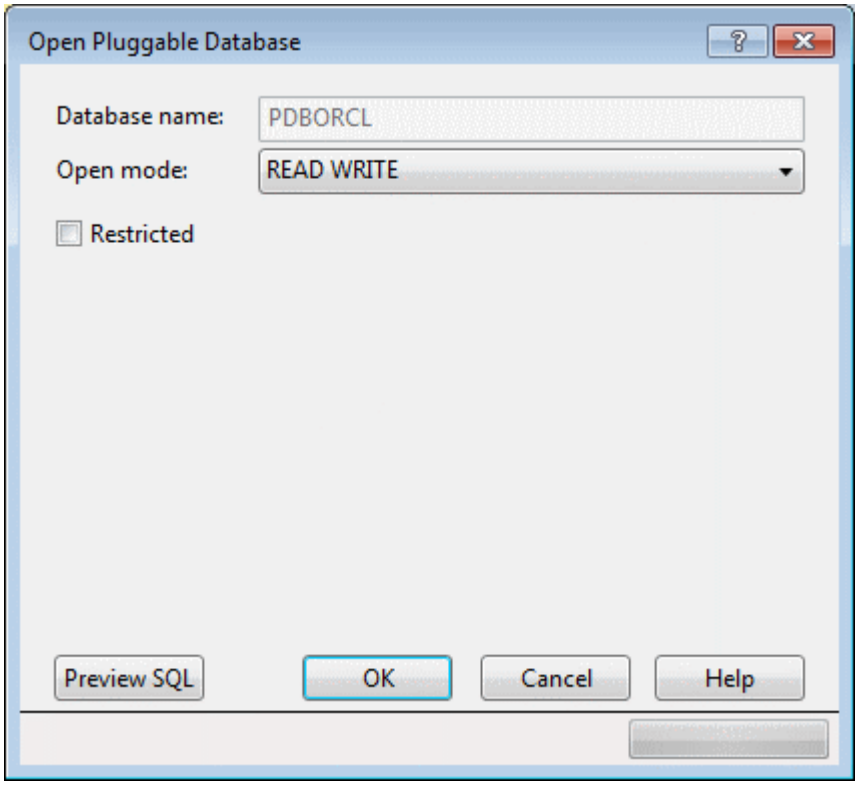
The Open Pluggable Database dialog opens a pluggable database, which makes it accessible to normal users.

> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

Starting the Open Pluggable Database Dialog

To open the Open Pluggable Database Dialog, connect to Server Explorer with a user that has SYSDBA privileges. Expand the Pluggable Databases Node and view the list of Pluggable Database Nodes. Closed pluggable databases have a small red x in their icon. When you have located the pluggable database that you wish to Open, right-click on it and choose **Open**.

The Open Pluggable Database Dialog appears as follows:



Using the Open Pluggable Database Dialog

The Open Pluggable Database Dialog has the following controls:

Control	Description
Database name	Displays the name of the pluggable database to be closed. This field cannot be modified.
Open mode	Specifies how the database opens. If you select `READ WRITE`, the database can be used normally for all operations. If you select `READ ONLY`, no operations that modify the database are allowed.
Restricted	Enable this check box to allow access to this pluggable database only to users that have been granted the `RESTRICTED SESSION` privilege.
Preview SQL	View the SQL to be executed when the OK button is pressed.
OK	Opens the pluggable database and closes this dialog.
Cancel	Closes this dialog without making any changes to the database.
Help	Opens this help page.

Close Pluggable Database Dialog

The Close Pluggable Database dialog closes a pluggable database, which makes it inaccessible to normal users. This is sometimes required in order to perform certain administrative tasks.

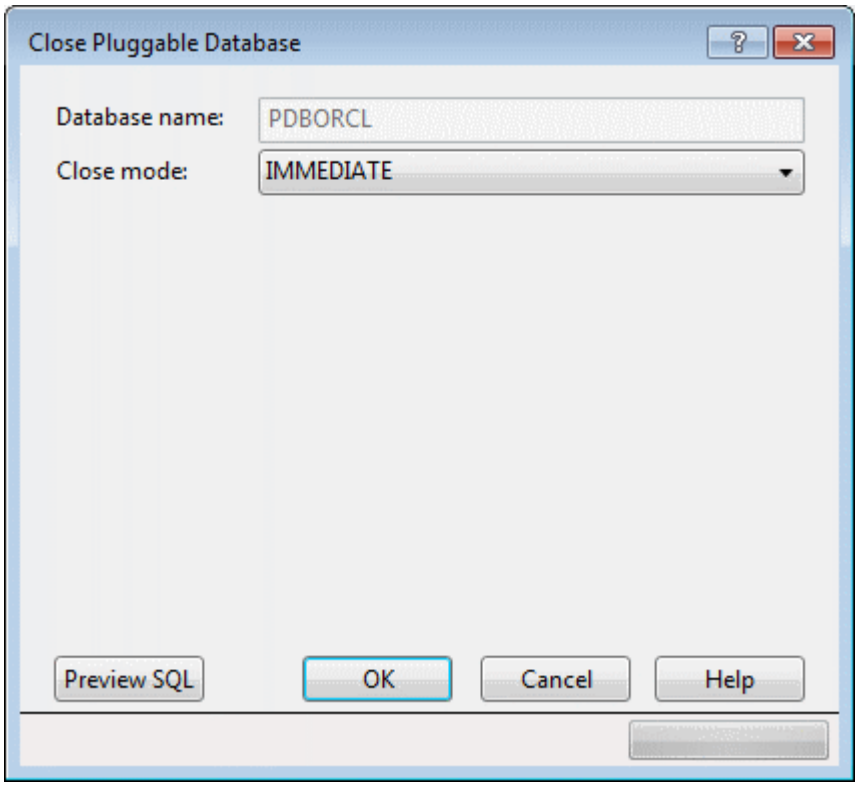
> **Note**
>
> Some pluggable database features in ODT, such as Clone Pluggable Database Dialog and Unplug Pluggable Database Dialog, automatically close the pluggable database or make it read-only, as part of their operation and so explicitly closing the pluggable database may not be required for those operations.

> **Note**
>
> This is a feature of Oracle Database 12.1 or later.

Starting the Close Pluggable Database Dialog

To open the Close Pluggable Database Dialog, connect to Server Explorer with a user that has SYSDBA privileges. Expand the Pluggable Databases Node and view the list of Pluggable Database Nodes. When you have located the pluggable database that you wish to close, right-click on it and choose **Close**.

The Close Pluggable Database Dialog appears as follows:



Using the Close Pluggable Database Dialog

The Close Pluggable Database Dialog has the following controls:

Control	Description
Database name	Displays the name of the pluggable database to be closed. This field cannot be modified.
Close mode	Specifies how the database is closed. If you select IMMEDIATE, all existing connections to the database are disconnected from the database and it closes immediately. If you select NORMAL, the database waits until all existing connections have closed before the database closes.
Preview SQL	View the SQL that will be executed with the OK button is pressed.
OK	Closes the pluggable database and this dialog.
Cancel	Closes this dialog without making any changes to the database.
Help	Opens this help page.

Query Parameters Dialog

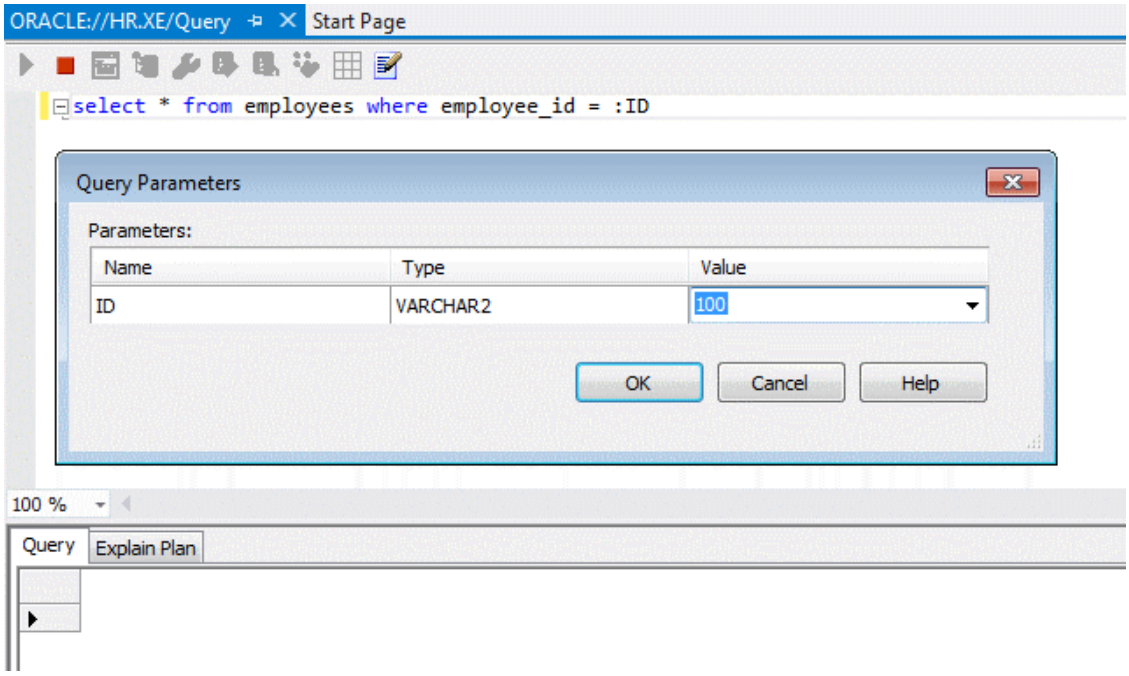
The Query Parameter Dialog appears whenever SQL statements or PL/SQL blocks that contain parameter placeholders are executed within the SQL and PL/SQL File Editor. A parameter placeholder is identified by a colon followed by any arbitrary identifier (for example, `:ID` or `:1`).

Once one or more parameter values are entered into the dialog and the **OK** button is pressed, the SQL statement or PL/SQL block is executed using the parameter values entered.

Starting the Query Parameters Dialog

To use the Query Parameters Dialog, open the SQL and PL/SQL File Editor. Enter a SQL statement or a PL/SQL block that contains a parameter placeholder. Then execute the statement by right-clicking the SQL and PL/SQL File Editor and choosing **Execute SQL**.

The Query Parameters Dialog appears as follows:



Using the Query Parameters Dialog

The Query Parameters Dialog has the following controls:

Control	Description
Name	Displays name of the parameter placeholder.
Type	Displays the external data type of the parameter. This is the data type ODT uses when storing and passing the value to Oracle Database. The default type, `VARCHAR2`, should be sufficient for most cases, because it is convertible to many other Oracle Internal data types. For more information, see External Data Types in the *Oracle Call Interface Programmer's Guide*.
Value	Enter the value for the parameter or select `NULL` from the drop-down list.
OK	Closes this dialog and executes the SQL or PL/SQL using the parameter values provided.
Cancel	Closes this dialog and cancels the execution of the SQL or PL/SQL.
Help	Opens this help page

Grant Debugging Privileges Dialog

The Grant Debugging Privileges Dialog provides an easy way to grant the necessary privileges to a user so that PL/SQL Debugging is possible.

As indicated in PL/SQL Debugging Setup, there are privileges that must be granted by SYSDBA to a user before PL/SQL debugging is possible. For example, in Oracle Database 10*g* Release 1 (10.1) or later, both `DEBUG CONNECT SESSION` and `DEBUG ANY PROCEDURE` must be granted.

In Oracle Database 12.1 or later, additional privileges granting explicit access to an IP address and port range are also required to debug PL/SQL. These privileges are granted using the `DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE` package procedure. To grant privileges on multiple port ranges for the same IP, run this dialog multiple times and provide different port ranges each time. All port range privileges granted from previous executions of this dialog will remain in effect.

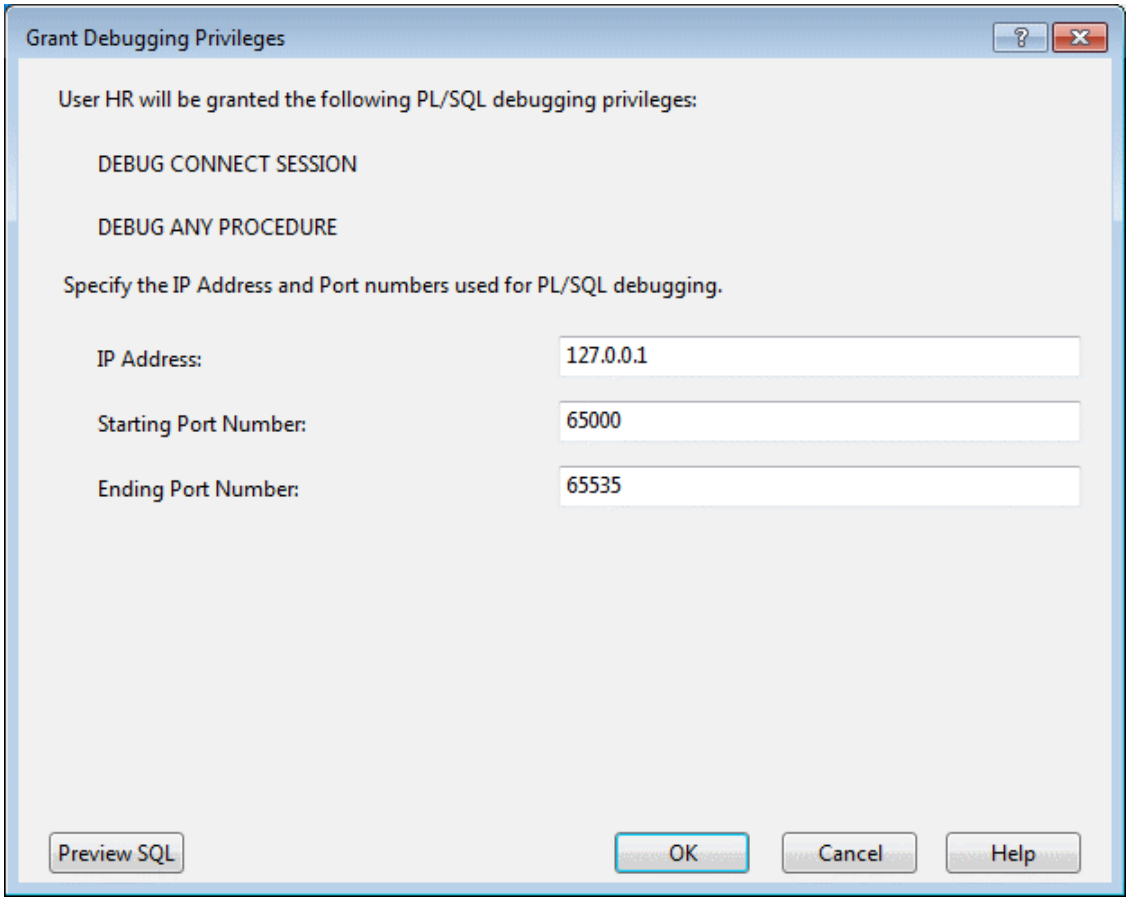
See Also

PL/SQL Debugging Setup

Starting the Grant Debugging Privileges Dialog

To open the Grant Debugging Privileges Dialog, connect in Server Explorer as SYSDBA to the database where the PL/SQL that you wish to debug is located. Next, locate the User Node or Schema Node representing the user who needs to be granted the privileges. From the context menu of the node, select **Grant Debugging Privileges**. This context menu item is visible only in a SYSDBA connection.

The Grant Debugging Privileges Dialog appears as follows:



Using the Grant Debugging Privileges Dialog

The Grant Debugging Privileges Dialog has the following controls:

Control	Description
IP Address	Specify the IP address for the computer where Visual Studio is installed. This control is only visible when connected to Oracle Database 12.1 or later. For more information, see PL/SQL Debugging Options.
Starting Port Number	Specify the starting port number in the range of port numbers for the computer where Visual Studio is installed. This control is only visible when connected to Oracle Database 12.1 or later. For more information, see PL/SQL Debugging Options.
Ending Port Number	Specify the ending port number in the range of port numbers for the computer where Visual Studio is installed. This control is only visible when connected to Oracle Database 12.1 or later. For more information, see PL/SQL Debugging Options.
Preview SQL button	Displays the SQL statements to be executed by this dialog.
OK	Runs the SQL statements to grant the debugging privileges.
Cancel	Closes the dialog and does not execute any SQL statements.
Help	Launches this help page.

Revoking Privileges

The privileges granted by this dialog may be revoked by using the Grant/Revoke Privileges Dialog and for Oracle Database 12.1 or later also executing the `DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACE` package procedure as `SYSDBA`.

For example,

```
BEGIN
   DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACE(
     HOST => '255.255.255.25',
     LOWER_PORT => 61000,
     UPPER_PORT => 65000,
     ACE => XS$ACE_TYPE(PRIVILEGE_LIST => XS$NAME_LIST('jdwp'),
                        PRINCIPAL_NAME => 'HR',
                        PRINCIPAL_TYPE => XS_ACL.PTYPE_DB));
 END;
```

Replace the host, lower port, upper port, and principal name with the values provided in earlier calls to the Grant/Revoke Privileges Dialog.

Import Schema Dialog

The Import Schema Dialog enables you to import a set of SQL scripts into an Oracle Database Project Version 2 project that represent an entire Oracle Database schema. One SQL script is added for each Oracle schema object. Import Schema may only performed once in the lifetime of an Oracle Database Project Version 2 project. Additional scripts may be added to the project via the Schema Compare tool, or using the Add Existing Item project menu item. See the Visual Studio output window for error messages and warnings related to the import.

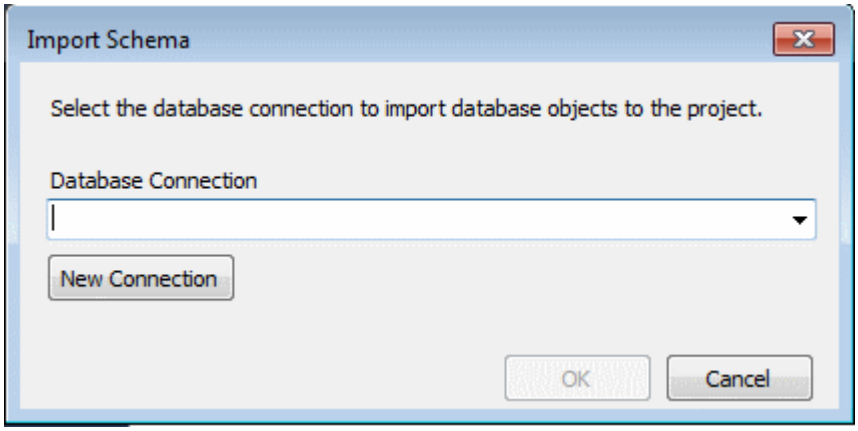
Note: Imported SQL scripts will only include a schema name if the **Include Schema Name for the connected user** option is selected in the General Options Page.

Opening the Import Schema Dialog

You can open the Import Schema Dialog from the Oracle Database Project Version 2 Project Folder, Import Schema menu item.

Using the Import Schema Dialog

The Import Schema Dialog appears as follows:



The Import Schema Dialog has the following controls:

Control	Description
Database Connection	Choose an existing Server Explorer Oracle database connection from the drop down list. The Oracle schema associated with this connection will be imported.
New Connection	Opens the Connection Dialog Box to create a new Server Explorer Oracle database connection. The Oracle schema associated with this connection will be imported.
Ok	Performs the import and closes the dialog. See the Visual Studio output window for error messages and warnings related to the import.
Cancel	Cancels the operation and closes the dialog.

Dependencies and References Viewer

The Dependencies and References Viewer enables you to view the dependencies and references that a database schema object has on other schema objects, both for Oracle Database connections as well as Oracle Database Project Version 2 projects. For example, if a stored procedure `ADD_JOB_HISTORY` performs a insert into database table `JOB_HISTORY`, then the Dependencies and References Viewer shows you that `ADD_JOB_HISTORY` has a dependency on `JOB_HISTORY`, while `JOB_HISTORY` is referenced by `ADD_JOB_HISTORY`.

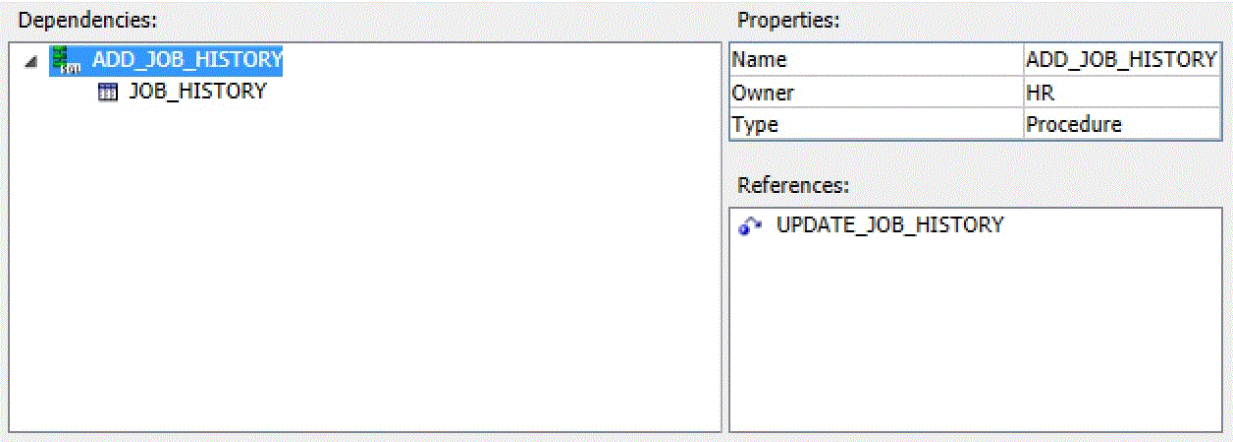
Dependencies are displayed through a tree control where each node in the tree is a dependency of its parent node. When a node in the tree is selected, that object's references are displayed in the References portion of the viewer. Each tree node has a Sync with Explorer menu item which expands either the Server Explorer tree control or the Oracle Database Project Version 2 folder to highlight that object's location.

Opening the Dependencies and References Viewer

To open the Dependencies and References Viewer, right click on supported nodes in Server Explorer or on file names in an Oracle Database Version 2 Project and choose the Dependencies and References menu item.

Using the Dependencies and References Viewer

The Dependencies and References Viewer appears as follows:



The controls in the basic Dependencies and References Viewer are as follows:

Control	Description
Dependencies	Each node in the tree control is a dependency of its parent node. When a node in the tree is selected, that object's references are displayed in the References portion of the viewer. Each tree node has a Sync with Explorer menu item which expands either the Server Explorer tree control or the Oracle Database Project Version 2 folder to highlight that object's location.
References	This is a list of references for the item selected in the Dependencies tree control. Each reference has a Dependencies and References menu item which reloads the Dependencies and References Viewer to show the dependencies for that object.

Dependency Viewer Options Page

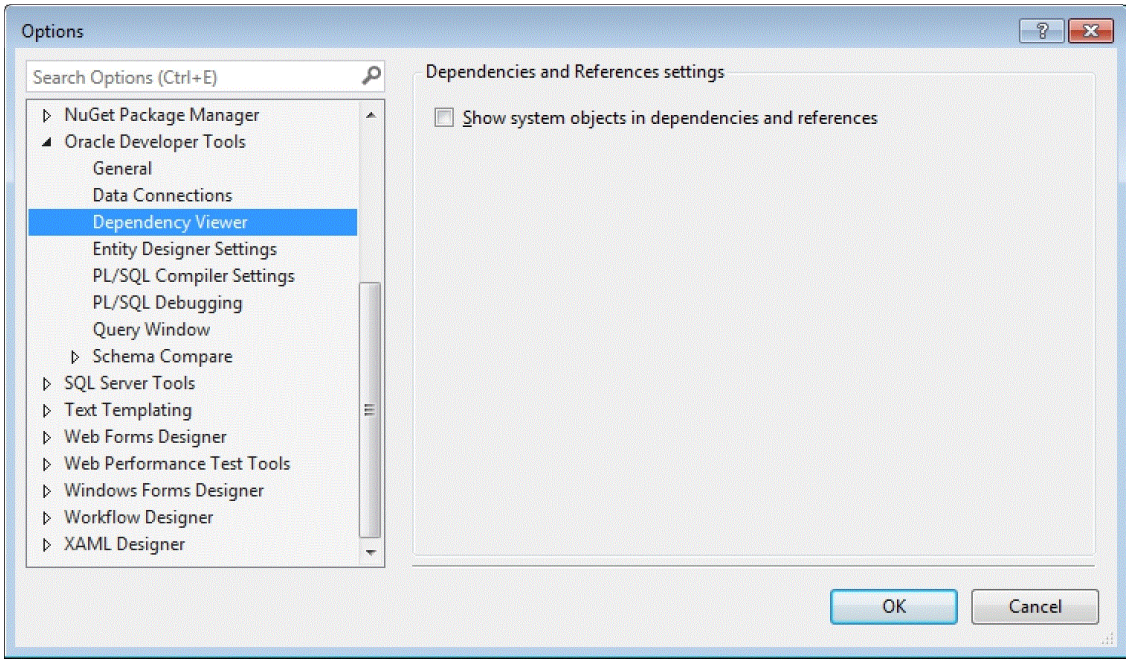
You can control the default settings for the Dependencies and References Viewer on this options page.

Accessing the Dependency Viewer Options

To access the Dependency Viewer Options Page, select Options from the Tools menu. From the Options menu, select Oracle Developer Tools. Then select Dependency Viewer.

Using the Dependency Viewer Options Settings

The Dependency Viewer Options settings appears similar to the following:



The controls of the Dependency Viewer Options are as follows:

Control	Description
Show system objects in dependencies and references	If this option is checked, system objects will be shown in the Dependency and References Viewer.

Change Compartment or Region Dialog

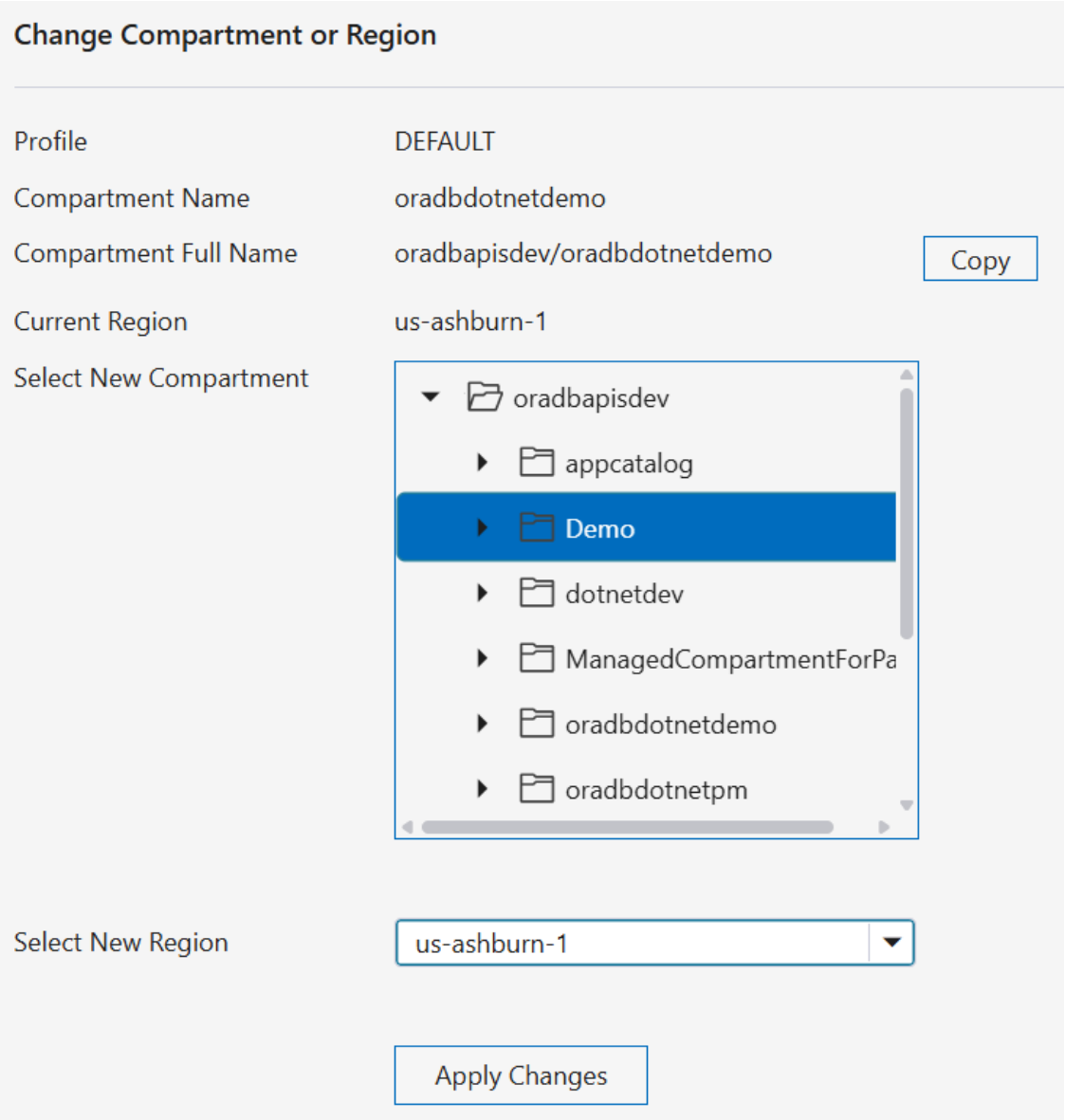
The Change Compartment or Region dialog allows users to choose the compartment where future operations will be performed in.

Opening the Change Compartment or Region Dialog

You can open the Change Compartment or Region dialog from an Oracle Cloud Infrastructure Profile Nodes, Compartments or Region menu item.

Using the Change Compartment or Region Dialog

The Change Compartment or Region Dialog appears as follows:



The Change Compartment or Region Dialog has the following controls:

Control	Description
Profile	Displays the profile being used from your OCI config file. Currently only the DEFAULT profile can be used.
Compartment Name	The name of compartment currently in use
Compartment Full Name	The full name of the compartment currently in use.
Copy button	Click thus button to copy the full name of the compartment currently in use to the clipboard
Current Region	The region that is currently in use
Select New Compartment	Navigate and select the compartment you wish to use via this tree control
Select New Region	Select the region you wish to use from this drop down list

Control	Description
Apply Changes button	Click this button to apply any changes you have made in this dialog box. To cancel any changes, simply close the tab containing the dialog.

Create Autonomous AI Database Dialog

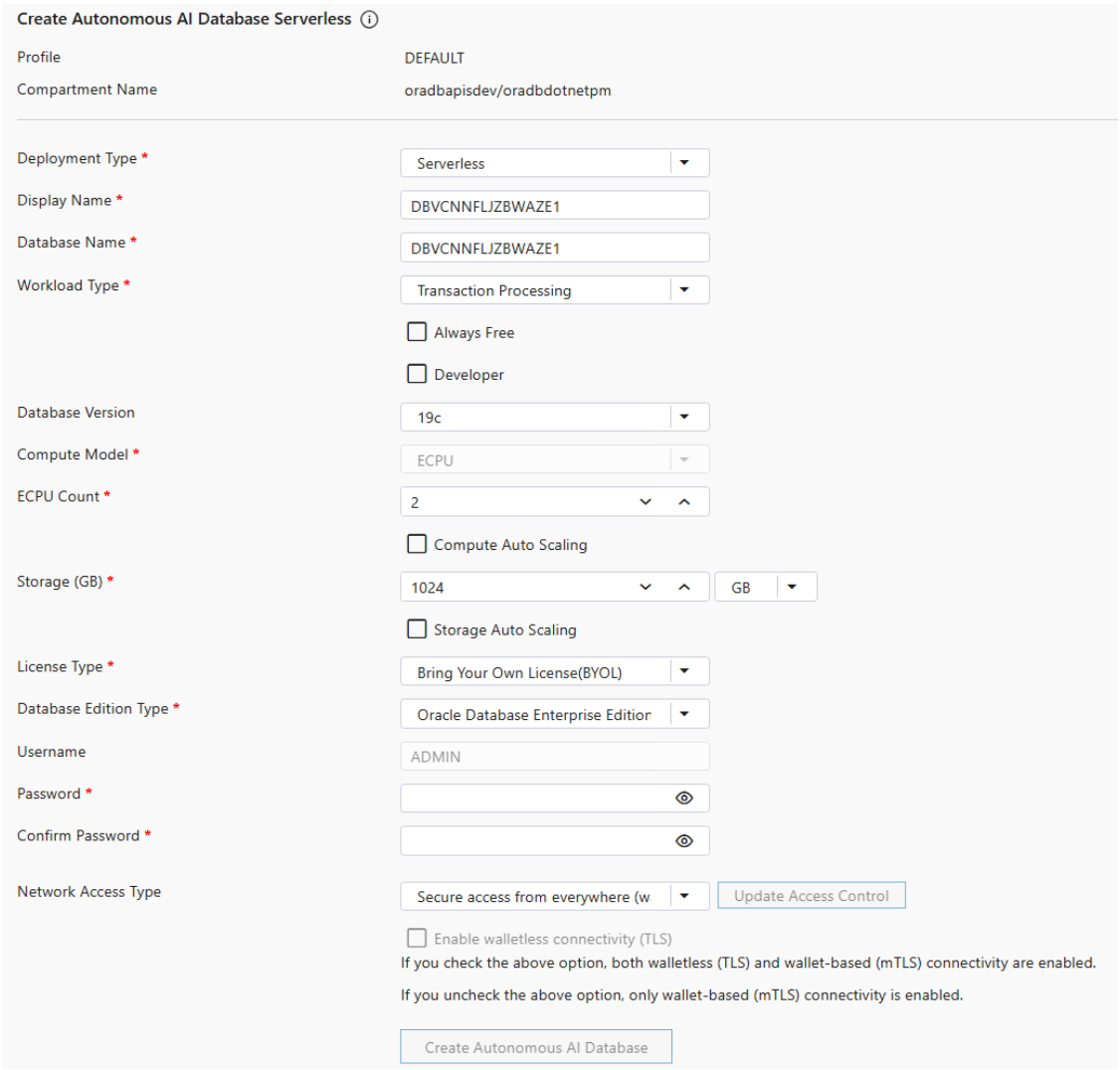
The Create Autonomous AI Database Dialog allows users to create a new Autonomous Transaction Processing Database, an Autonomous Data Warehouse, or Autonomous JSON Database, which are represented by Transaction Processing/Lakehouse/JSON Nodes.

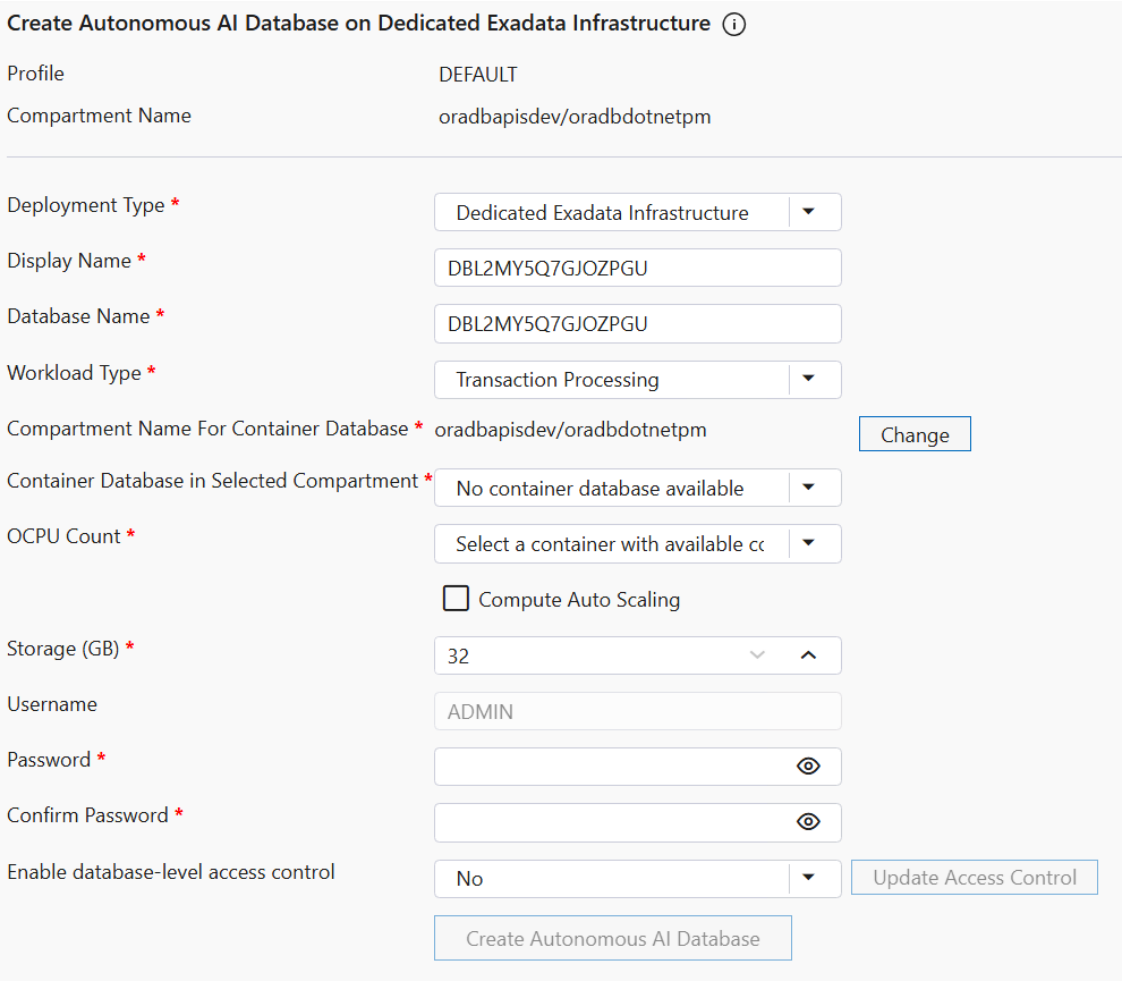
Opening the Create Autonomous AI Database Dialog

You can open the Create Autonomous AI Database Dialog from an Transaction Processing/Lakehouse/JSON Node, Create new menu item. After submitting this dialog an Transaction Processing/Lakehouse/JSON Nodes is created.

Using the Create Autonomous AI Database Dialog

The Create Autonomous AI Database Dialog appears as follows:





The Create Autonomous AI Database Dialog has the following controls:

Control	Description
Profile	Displays the profile being used from your OCI config file. Currently only the DEFAULT profile can be used.
Compartment Name	Displays the compartment that the database will be created in
Deployment Type	Select **Serverless** or **Dedicated Infrastructure**. The fields in this dialog will change depending on which of these options is selected.
Workload Type	Displays **Transaction Processing**, **Lakehouse**, **JSON** depending on the type of instance being created.
Display Name	The name used in Server Explorer to represent this database
Database Name	The name of the database

Control	Description
Always Free	If this box is checked, then this will create an Always Free instance of the database if this is available to you. CPU, Storage, auto scaling and licensing options will be automatically set. There may be limits to the number of free instances you can create. If you receive an error that you have exceeded the limit, delete other free instances (denoted in Server Explorer with a green star on the database instance icon).
Developer	If this box is checked, this will create a Autonomous AI Database for Developers
Database Version (Serverless only)	Select the database version
Compute Model (Serverless only)	Select between **OCPU** and **ECPU**
ECPU Count (Serverless only)	The number of ECPUs allocated for this database
Compartment Name for Container Database (Dedicated only)	Displays the compartment where the container instance is located. Click the Change button to select a different compartment.
Change button (Dedicated only)	Click this button to modify the Compartment Name for Container Database
Container Database in Selected Compartment (Dedicated only)	Choose the container database that will be used to create this database instance.
OCPU Count	The number of OCPUs allocated for this database
Compute Auto Scaling	Enables Compute Auto Scaling for this instance
Storage	The storage allocated for this database
Storage Auto Scaling (Serverless only)	Enables Storage Auto Scaling on this database.
License Type (Serverless only)	Choose **Bring your own license** to rely on a license that you already have. Choose **License Included** to acquire a new license
Database Edition Type (Serverless only)	Select the database edition you wish to use.
Username	The username that will be used for administration
Password	The password for this username. See the dialog for password requirements
Confirm Password	Retype the password to confirm it
Network Access Type (Serverless only)	Select the type of network access you wish to use
Enable database-level access control (Dedicated Only)	Enables database-level access control. If "yes" is selected you can click the **Update Access Control** button to set up the Access Control List.
Update Access Control button	Click this to open the Update Access Control Dialog to select which IPs, CIDR blocks, or VCNs are allowed access.
Enable walletless connectivity (TLS) (Serverless only)	If you check this option, both walletless(TLS) and wallet-based(mTLS) connectivity are enabled. If you uncheck this option, only wallet-based (mTLS) connectivity is enabled.
Create Autonomous AI Database	Creates the database

Scale Up/Down Dialog

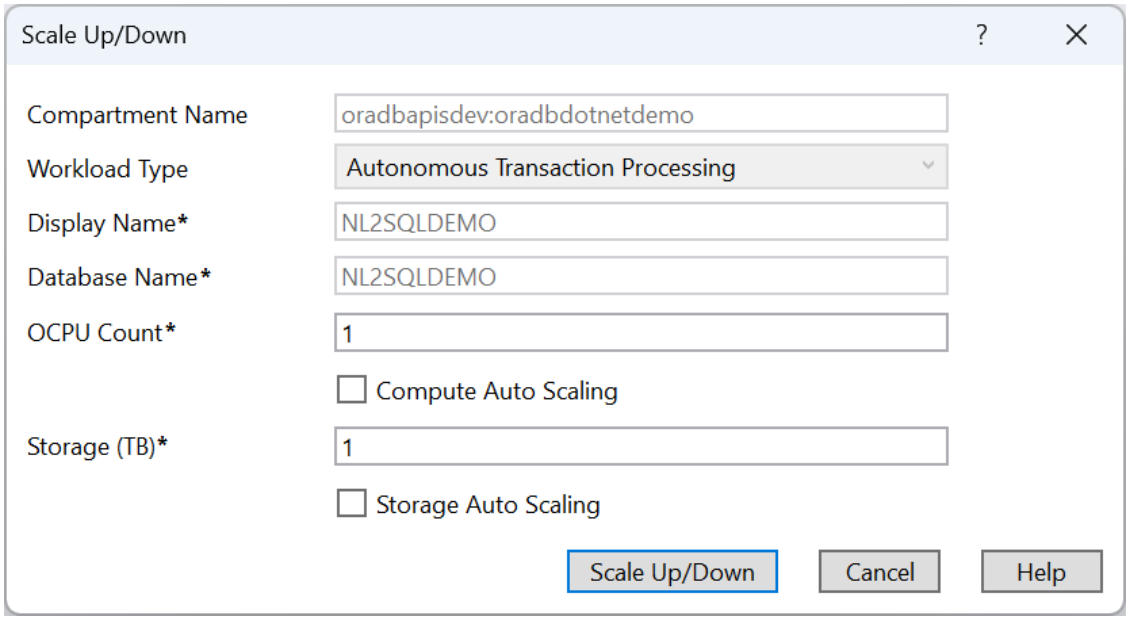
The Scale Up/Down Dialog allows users to modify resource settings for a Autonomous Transaction Processing Database or an Autonomous Data Warehouse.

Opening the Scale Up/Down Dialog

You can open the Scale Up/Down Dialog from an Transaction Processing/Lakehouse/JSON Node, Scale Up/Down menu item.

Using the Scale Up/Down Dialog

The Scale Up/Down Dialog appears as follows:



The Scale Up/Down Dialog has the following controls:

Control	Description
Compartment Name	Displays the compartment that the database is located in
Workload Type	Displays the workload type of this instance: Automatic Transaction Processing, Automatic Data Warehouse, or Automatic JSON Database
Display Name	The name used in Server Explorer to represent this database
Database Name	The name of the database
OCPU Count	The number of CPU Cores allocated for this database
Compute Auto Scaling	Enables Compute Auto Scaling for this instance

Control	Description
Storage	The storage (in TB) allocated for this database
Storage Auto Scaling	Enables Storage Auto Scaling on this instance
Scale Up/Down	Modifies the database and closes the dialog if successful
Cancel	Cancels the operation and closes the dialog
Help	Opens this help page

Change Administrator Password Dialog

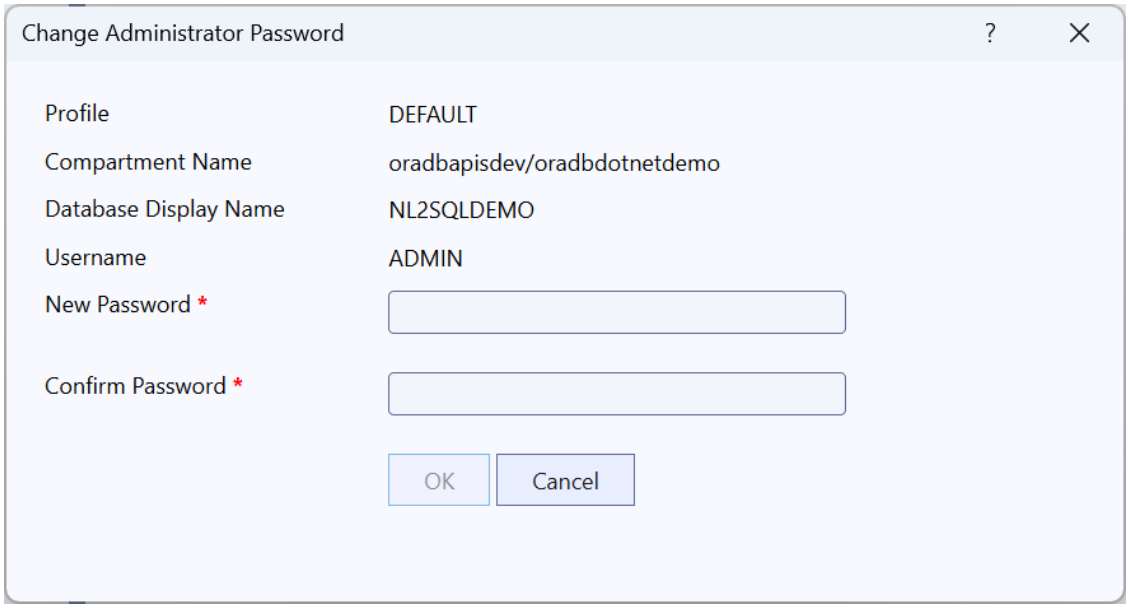
The Change Administrator Password Dialog allows users to modifiy the credentials for a new Autonomous Transaction Processing Database or an Autonomous Data Warehouse.

Opening the Change Administrator Password Dialog

You can open the Change Administrator Password Dialog from an Transaction Processing/ Lakehouse/JSON Node, Change Administrator Password menu item.

Using the Change Administrator Password Dialog

The Change Administrator Password Dialog appears as follows:



The Change Administrator Password Dialog has the following controls:

Control	Description
Profile	The profile being used from your OCI config file. Currently only the DEFAULT profile can be used.

Control	Description
Compartment Name	Displays the compartment that the database is located in
Database Display Name	The name used in Server Explorer to represent this database
Username	The username that is used for administration
Password	The new password for this username. See the dialog for password requirements
Confirm Password	Retype the new password to confirm it
OK	Updates the Password and closes the dialog if successful
Cancel	Cancels the operation and closes the dialog

Download Credentials Files Dialog

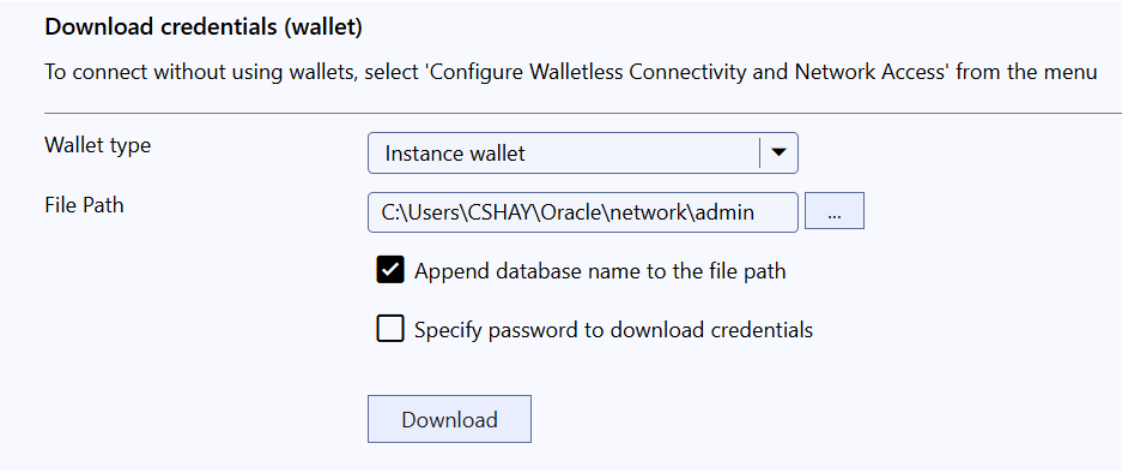
The Download Credentials Files Dialog downloads credentials files. Credentials files include wallet files, `TNSNAMES.ORA` alias file and `SQLNET.ORA` files, among others. If there are existing file conflicts when the credentials files are attempted to be copied, the Copy Credentials Files Dialog will open to assist in resolving the conflict.

Opening the Download Credentials Files Dialog

You can open the Download Credentials Files Dialog from an Transaction Processing/ Lakehouse/JSON Nodes, Download Credentials Files menu item.

Using the Download Credentials Files Dialog

The Download Credentials Files Dialog appears as follows:



The Download Credentials Files Dialog has the following controls:

Control	Description
Wallet Type	Choose the wallet type. An Instance wallet is used to generate a wallet for the current database only. A Regional wallet is used to generate and download the wallet for all databases in the region.
File Path	The location where the credentials files can be downloaded to
Append database name to the file path	If checked, a directory with the name of the database instance will be created in the File Path directory and the credentials files will be extracted into new directory. If unchecked, the credentials files will be extracted directly in the File Path directory.
Specify a password to download credentials	Check this box if you wish to secure the credentials with a password
Password	Enter the credentials files password
Confirm Password	Confirm the credentials files password
Download	Downloads the credentials files. If there are conflicts when downloading credentials files, the Copy Credentials Files Dialog will open to assist in resolving the conflict.

Copy Credentials Files Dialog

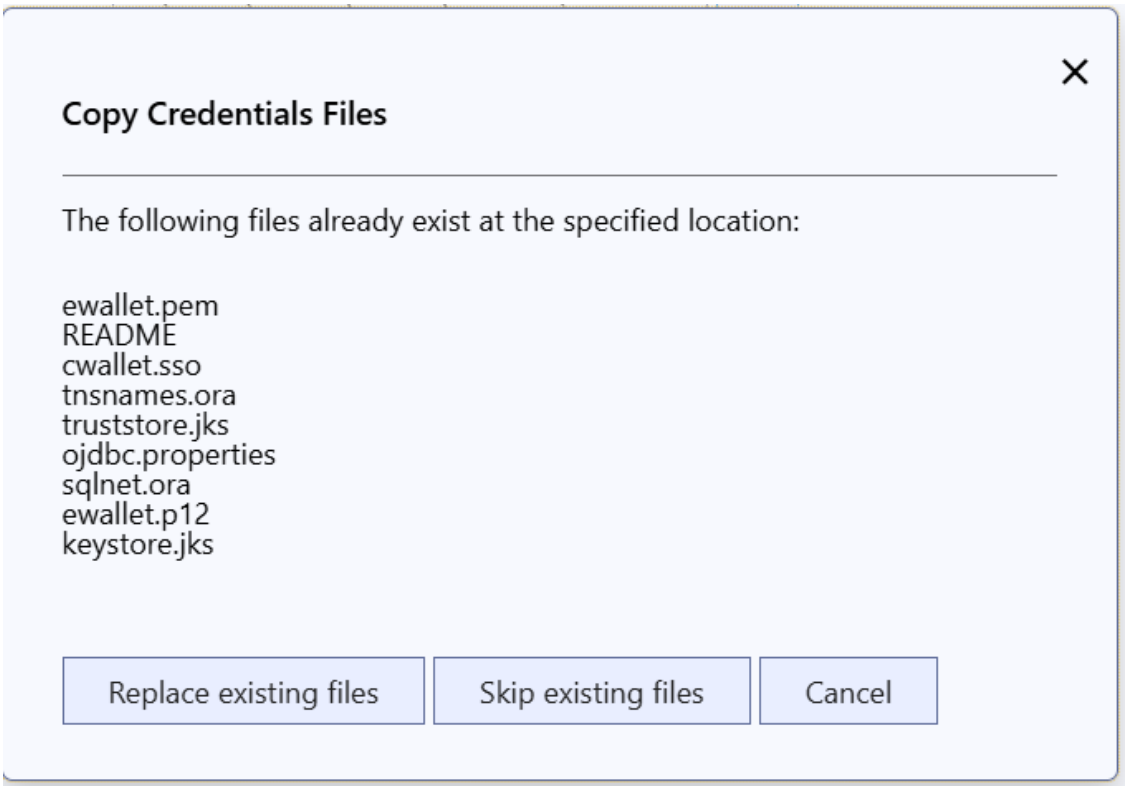
The Copy Credentials Files Dialog opens if there are existing file conflicts when the credentials files are attempted to be copied.

Opening the Copy Credentials Files Dialog

This dialog opens automatically when there are conflicts when credentials files are downloaded.

Using the Copy Credentials Files Dialog

The Copy Credentials Files Dialog appears as follows:



The Copy Credentials Files Dialog has the following controls:

Control	Description
The following files already exist at the specified location	The list of files that already exist when the credentials are attempted to be downloaded
Replace Existing Files	Replaces the listed files with a newly downloaded file
Skip Existing Files	Does not replace any of the listed files
Cancel	Cancels the operation and closes the dialog

Restore Database Dialog

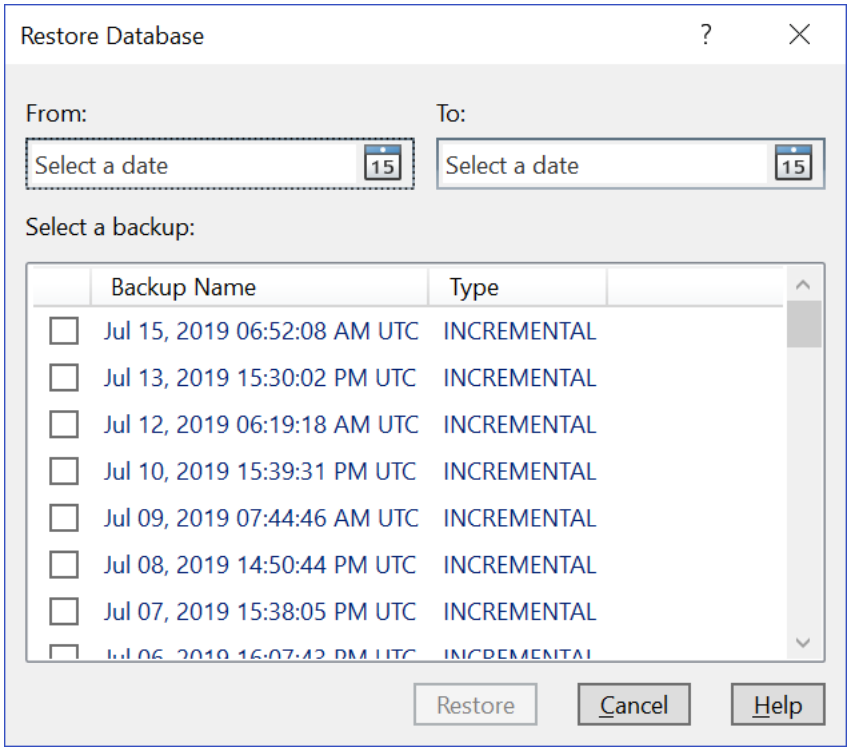
The Restore Database Dialog assists the user in restoring the database from a backup. After the restore operation has begun, the Transaction Processing/Lakehouse/JSON Nodes icon and properties will indicate the status of the operation.

Opening the Restore Database Dialog

You can open the Restore Database Dialog from an Transaction Processing/Lakehouse/JSON Nodes, Restore Database menu item.

Using the Restore Database Dialog

The Restore Database Dialog appears as follows:



The Restore Database Dialog has the following controls:

Control	Description
From:	Select the start date for filtering the list of backups
To:	Select the end date for filtering the list of backups
Select a backup	Choose a single backup to be used to restore the database. The backup name and type are provided.
Restore	Begins the restore operation and the dialog closes. The Transaction Processing/Lakehouse/JSON Nodes icon and properties will indicate the status of the operation.
Cancel	The entire operation is aborted and no file is written to disk. The dialog is closed.
Help	Opens this help page

Update License Type Dialog

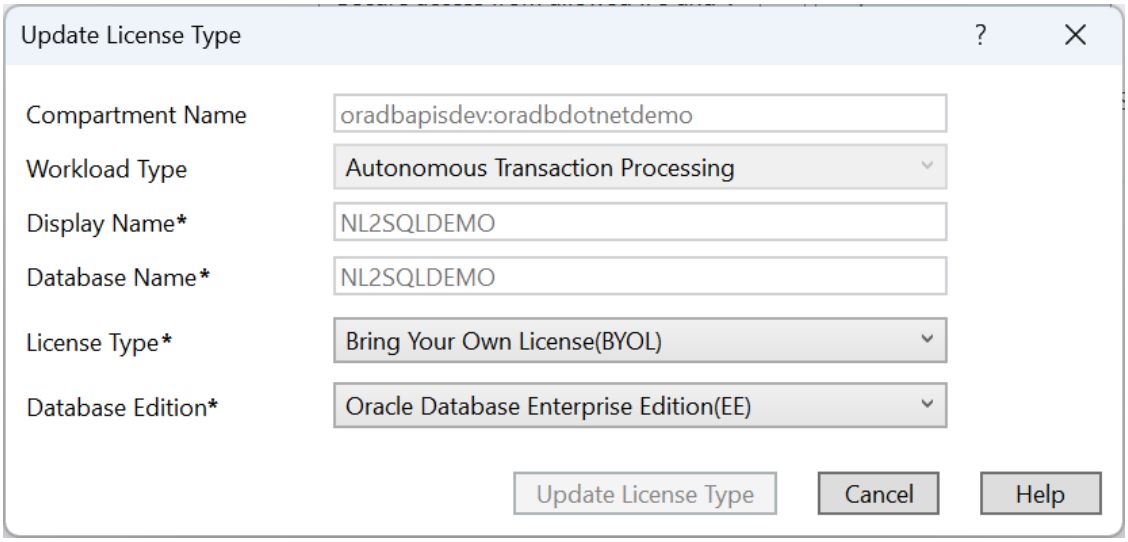
The Update License Type Dialog allows users to update the license type of a Autonomous Transaction Processing Database or an Autonomous Data Warehouse.

Opening the Update License Type Dialog

You can open the Update License Type Dialog from an Transaction Processing/Lakehouse/JSON Nodes, Update License Type menu item.

Using the Update License Type Dialog

The Update License Type Dialog appears as follows:



The Update License Type Dialog has the following controls:

Control	Description
Compartment Name	The compartment that the database is created in
Workload Type	Displays the workload type of this instance: Automatic Transaction Processing, Automatic Data Warehouse, or Automatic JSON Database
Display Name	The name used in Server Explorer to represent this database
Database Name	The name of the database
License Type	Choose ***Bring your own license*** to rely on a license that you already have. Choose ***License Included*** to acquire a new license
Database Edition	Select the edition of the database you have a license for
Update License Type	Updates the license type and closes the dialog if successful
Cancel	The entire operation and closes the dialog
Help	Opens this help page

Configure Walletless Connectivity and Network Access Dialog

The Configure Walletless Connectivity and Network Access Dialog allows you to modify whether or not your Autonomous Database requires a wallet for authentication. This could be useful if you want to connect using only a connection string without needing to have access to

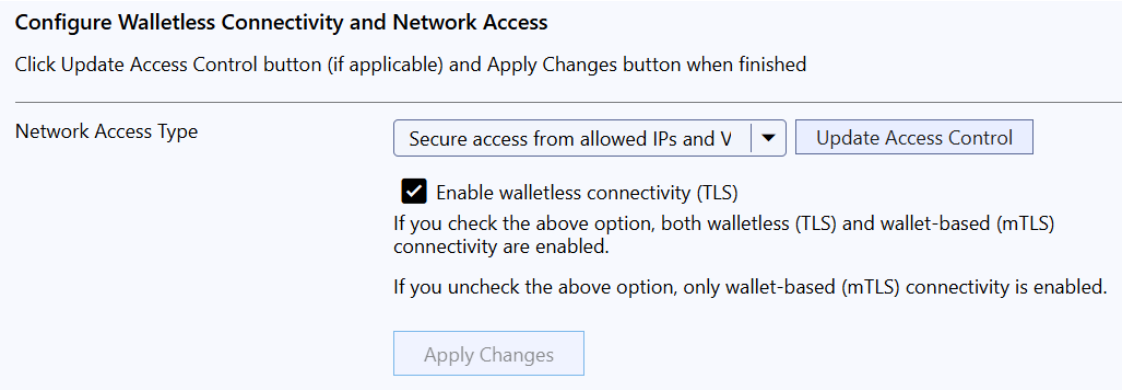
a wallet file on the filesystem. You can also set up the Access Control List and restrict which IP addresses my connect to the Autonomous Database.

Opening the Configure Walletless Connectivity and Network Access Dialog

You can open the Configure Walletless Connectivity and Network Access Dialog from an Autonomous Transaction Processing Database/Autonomous Data Warehouse Nodes, Configure Walletless Connectivity and Network Access menu item.

Using the Configure Walletless Connectivity and Network Access Dialog

The Configure Walletless Connectivity and Network Access Dialog appears as follows:



The Configure Walletless Connectivity and Network Access Dialog has the following controls:

Control	Description
Network Access Type	Select the type of network access you wish to use
Update Access Control button	Click this to open the Update Access Control Dialog to select which IPs, CIDR blocks, or VCNs are allowed access.
Enable walletless connectivity (TLS)	If you check this option, both walletless(TLS) and wallet-based(mTLS) connectivity are enabled. If you uncheck this option, only wallet-based (mTLS) connectivity is enabled.
Apply Changes	Apply changes you have made. The Autonomous Database will briefly be unavailable as the changes are made on the server. If you wish to cancel the operation without making changes, close the tab containing this dialog.

Get Connection Strings Dialog

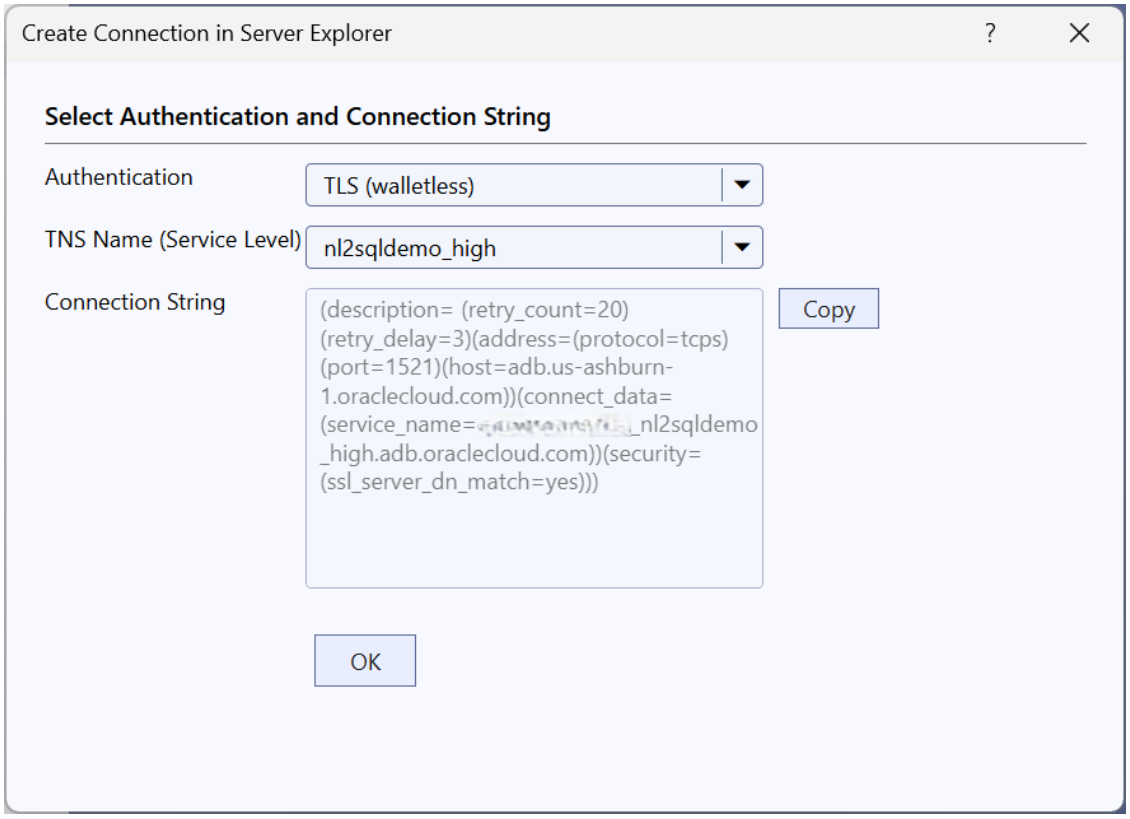
The Get Connection Strings Dialog allows you to obtain a connection string that can be used to connect to your Autonomous Database. This dialog is also used as part of a Create Data Connection operation.

Opening the Get Connection Strings Dialog

You can open the Get Connection Strings Dialog from an Autonomous Transaction Processing Database/Autonomous Data Warehouse Nodes, Configure Walletless Connectivity and Network Access menu item. This dialog is also used as part of a Create Data Connection operation.

Using the Get Connection Strings Dialog

The Get Connection Strings Dialog appears as follows:



The Get Connection Strings Dialog has the following controls:

Control	Description
Authentication	Select Mutual TLS (Wallet required) or TLS (walletless) depending on whether or not you intend to use a wallet.
TNS Name (Service Level)	Select the TNS Name that corresponds to the service level you would like to use.
Connection String	The connection string is displayed here.
Copy	Click the Copy button to copy the connection string to the clipboard

Control	Description
OK	This OK button will only appear as part of a Create Data Connection operation. Clicking the OK button will open the Connection Dialog Box prepopulated with the connection information. If the Authentication drop down is set to Mutual TLS (Wallet required), then the Download Credentials Files Dialog will first open to download the wallet.

Update Access Control Dialog

The Update Access Control Dialog allows you to select which IPs, CIDR blocks, or VCNs are allowed access to your Autonomous Database resources.

> **See Also**
>
> To learn more about the options available on this dialog, please visit: Configuring Network Access with Access Control Rules (ACLs)

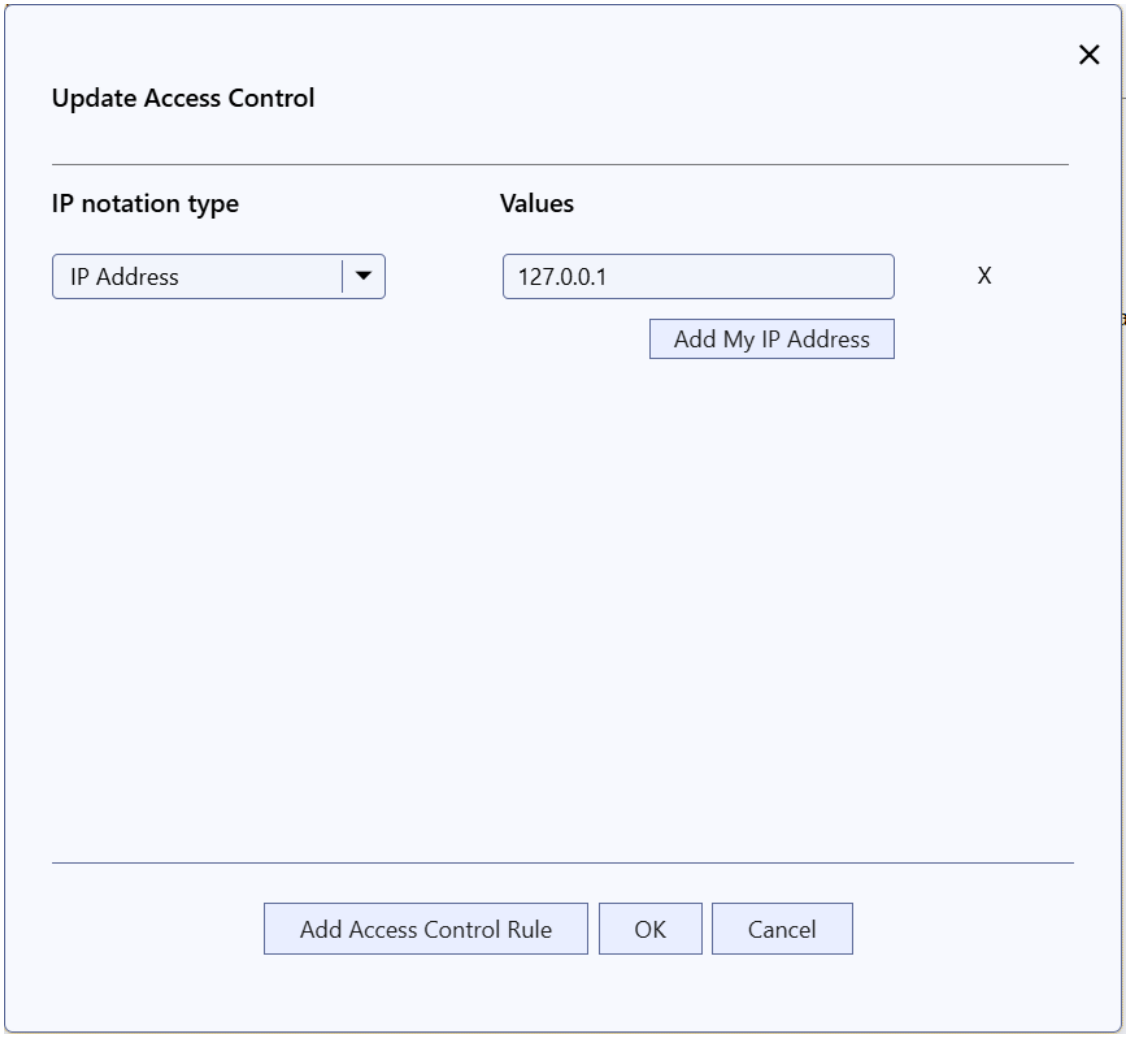
Opening the Update Access Control Dialog

You can open the Update Access Control Dialog by clicking a button on the Configure Walletless Connectivity and Network Access Dialog or on the Create Autonomous AI Database Dialog.

Using the Update Access Control Dialog

You may add access control rules (ACLs) by clicking the Add Access Control button, selecting the IP Notation Type you wish to use and then entering the relevant rules. When you have finished press the OK button. The Autonomous Database will be unavailable for a short period of time while this change is made.

The Update Access Control Dialog appears as follows:



The Update Access Control Dialog has the following controls:

Control	Description
IP Notation Type	Select "IP Address", "CIDR Block", "Virtual Cloud Network", or "Virtual Cloud Network (OCID)". To learn more about these options please visit: Configuring Network Access with Access Control Rules (ACLs)
Values (IP Address only)	Enter an IP address list, separated by commas. These represent the public IP addresses that are visible on the public internet that you want to grant access.
Add My IP Address (IP Address only)	Click this button to automatically add the IP address of the machine where Visual Studio is located to the list.
X button	Click the X button to delete an entry.
Values (CIDR Block only)	Enter a CIDR Block. This represents the public CIDR block for IP addresses that are visible on the public internet that you want to grant access.
Compartment Name (Virtual Cloud Network only)	The compartment that contains the Virtual Cloud Network

Control	Description
Change Compartment (Virtual Cloud Network only)	Opens the Select Compartment Dialog to allow you to select a different compartment.
Virtual cloud network in <compartment name> (Virtual Cloud Network only)	Select a Virtual Cloud Network from the drop down list. If you do not have the privileges to see the VCNs in your tenancy this list is empty. In this case use Virtual cloud network (OCID) instead.
IP addresses or CIDRs (Virtual Cloud Network only)	This field is optional. Enter private IP addresses or private CIDR blocks as a comma separated list to allow specific clients in the VCN.
Values (Virtual cloud network (OCID) only)	Enter the OCID of the VCN you want to grant access from
IP addresses or CIDRs (Virtual Cloud Network (OCID) only)	This field is optional. Enter private IP addresses or private CIDR blocks as a comma separated list to allow specific clients in the VCN.
Add Access Control Rule	Add an additional control rule to the list
OK	Closes the dialog and change the ACLs on the server. The Autonomous Database will be unavailable for a short period of time while this change is made.
Cancel	Closes the dialog and any changes that have been entered are discarded.

Select Compartment Dialog

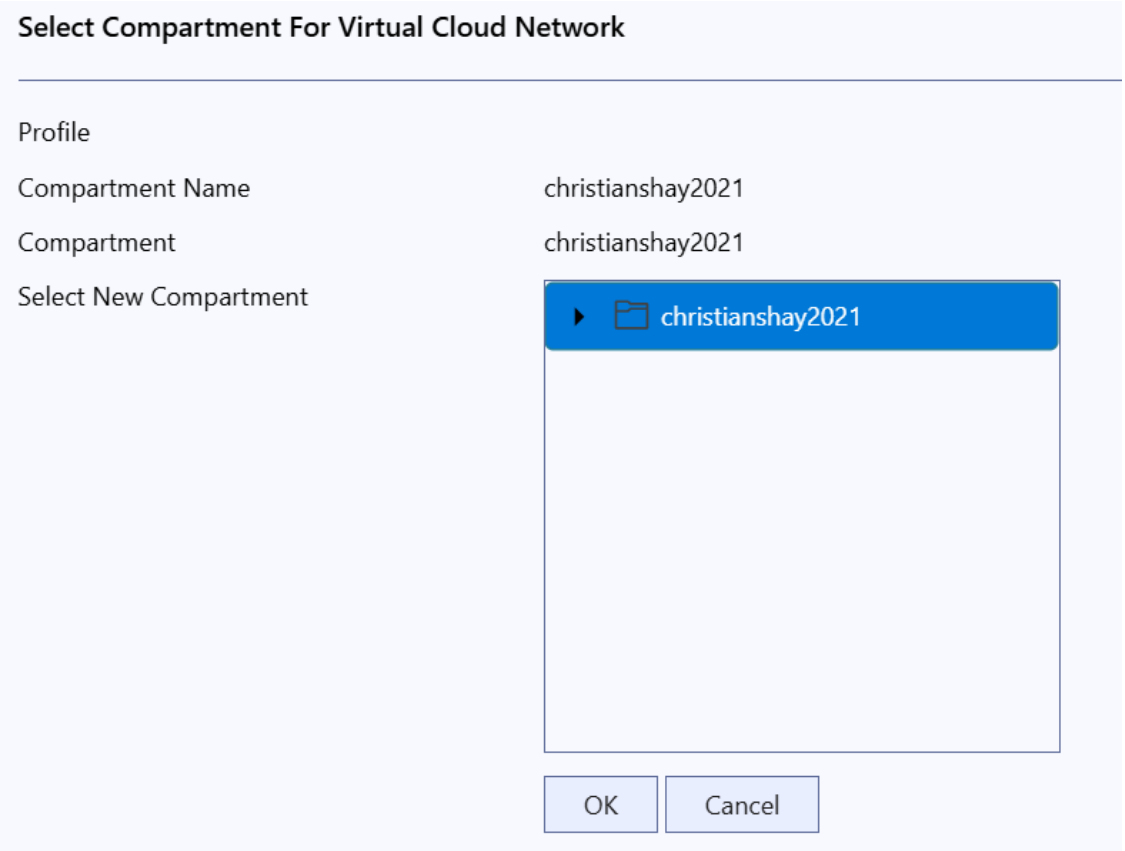
The Select Compartment Dialog allows users to choose an OCI compartment.

Opening the Select Compartment Dialog

You can open the Select Compartment Dialog from an Update Access Control Dialog.

Using the Select Compartment Dialog

The Select Compartment Dialog appears as follows:



The Select Compartment Dialog has the following controls:

Control	Description
Profile	Displays the profile being used from your OCI config file.
Compartment Name	The name of compartment currently in use
Select New Compartment	Navigate and select the compartment you wish to use via this tree control
OK	Click this button to apply any changes you have made in this dialog box.
Cancel	Closes the dialog without making changes

Real-Time SQL Monitor Window

The Real-Time SQL Monitor Window allows you to view the performance of SQL and PL/SQL that is currently or has previously been executed. You can also view an Active Report Window for individual SQL statements or PL/SQL blocks.

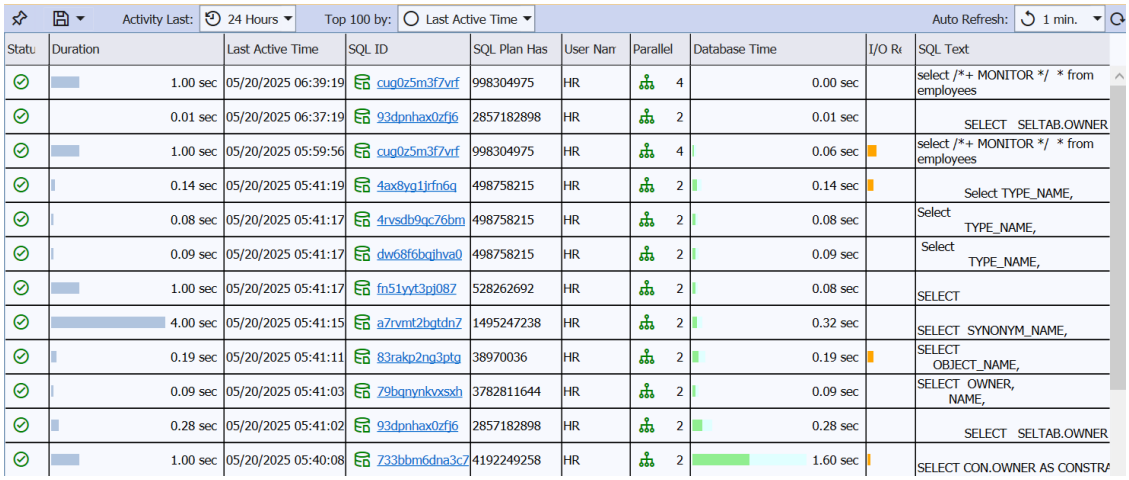
Opening the Real-Time SQL Monitor Window

You can open the Real-Time SQL Monitor Window by clicking the Real-Time SQL Monitor menu item on a Data Connection Nodes .

Using the Real-Time SQL Monitor Window

The Real-Time Monitor Window contains various toolbar icons and SQL IDs that can be clicked on. You may also hover over different grid cells to view additional information about a monitoring task.

The Real-Time SQL Monitor Window appears as follows:



The Real-Time SQL Monitor Window has the following controls:

Control	Description
Freeze Content	Temporarily stops refreshing of the content.
Save Selected Report	Saves an Active Report or XML SQL Monitor Report for SQL or PL/SQL associated the selected row.
Activity Last	Select the timeframe for which you wish to display the monitored SQL
Top 100 by	Choose the metric you wish to use to sort the results
Auto Refresh	Select the frequency with which the results should be auto refreshed. Select **Off** to disable auto refresh.
Refresh	Manually refresh the results
Column Header	Click any column header to sort that metric in ascending order. Click again to sort by that metric in descending order.
SQL ID	Click any SQL ID to open an Active Report for that SQL.

Active Report Window

The Active Report Window allows you to view an Active Report and learn more about a SQL statement or PL/SQL block's performance.

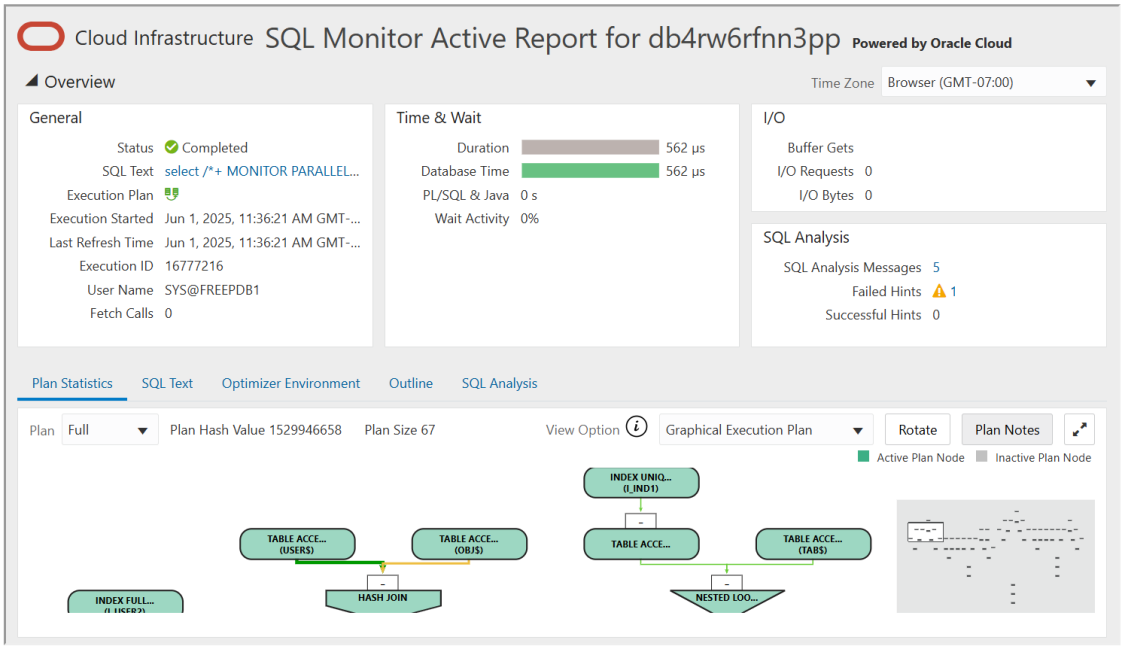
Opening the Active Report Window

You can open Active Report Window by clicking a SQL ID field in the Real-Time SQL Monitor Window or by clicking the **Execute and Monitor SQL** toolbar icon from within SQL and PL/SQL File Editor.

Active reports generated by **Execute and Monitor SQL** in Oracle Query Window will share one Active Report Window that will be updated when another statement in that same Query Window is executed and monitored. Active Reports generated by clicking the SQL-ID in the Real-Time SQL Monitor Window will each have their own separate Active Report Window.

Using the Active Report Window

The Active Report Window appears as follows:



Configure Select AI Provider Network Access Dialog

The Configure Select AI Provider Network Access Dialog enables the database to allow network access to an AI Provider for a specific database user. This dialog is typically only used one time per user when first setting up access. If you do not have the required permissions to execute this dialog, then you can generate the required SQL using the dialog and provide it to your Database Administrator to execute.

See Also

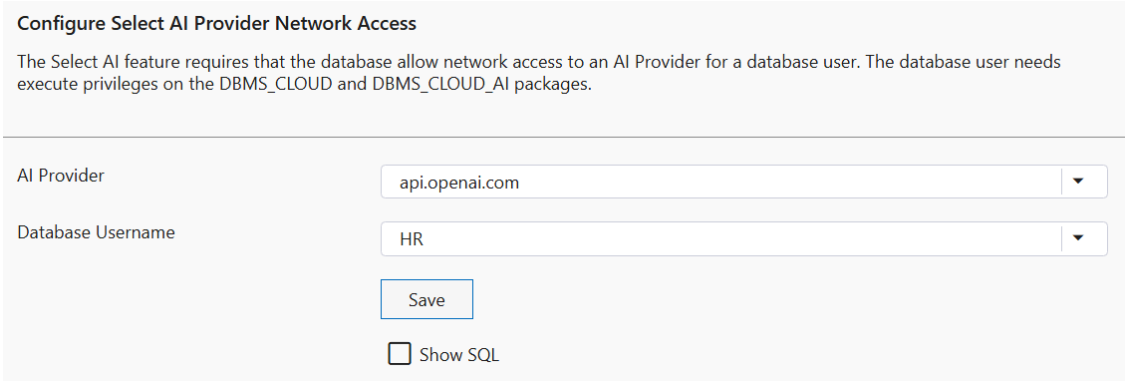
Using Select AI: Natural Language to SQL

Opening the Configure Select AI Provider Network Access Dialog

To open the Configure Select AI Provider Network Access Dialog, right click on the database connection node in Server Explorer and in the menu choose **Select AI** and then **Configure Select AI Provider Network Access**.

Using the Configure Select AI Provider Network Access Dialog

The Configure Select AI Provider Network Access Dialog appears as follows:



The Configure Select AI Provider Network Access Dialog has the following controls:

Control	Description
AI Provider	Choose your AI provider from the dropdown list
Database Username	Select the Database User that will be using Select AI
Save	Executes the commands required to configure Select AI provider network access and then closes the dialog
Show SQL	Check this checkbox to show the SQL that will be executed when the Save button is pressed. If you lack the privileges to execute this SQL you can copy it and provide it to your Database Administrator to execute.

Configure Select AI Profile Dialog

The Configure Select AI Profile Dialog creates a new profile to be used with Select AI. A profile requires a Credential and so this dialog also allows you to create a credential if one does not already exist. Before using this dialog the first time you will need to grant privileges using the Configure Select AI Provider Network Access Dialog.

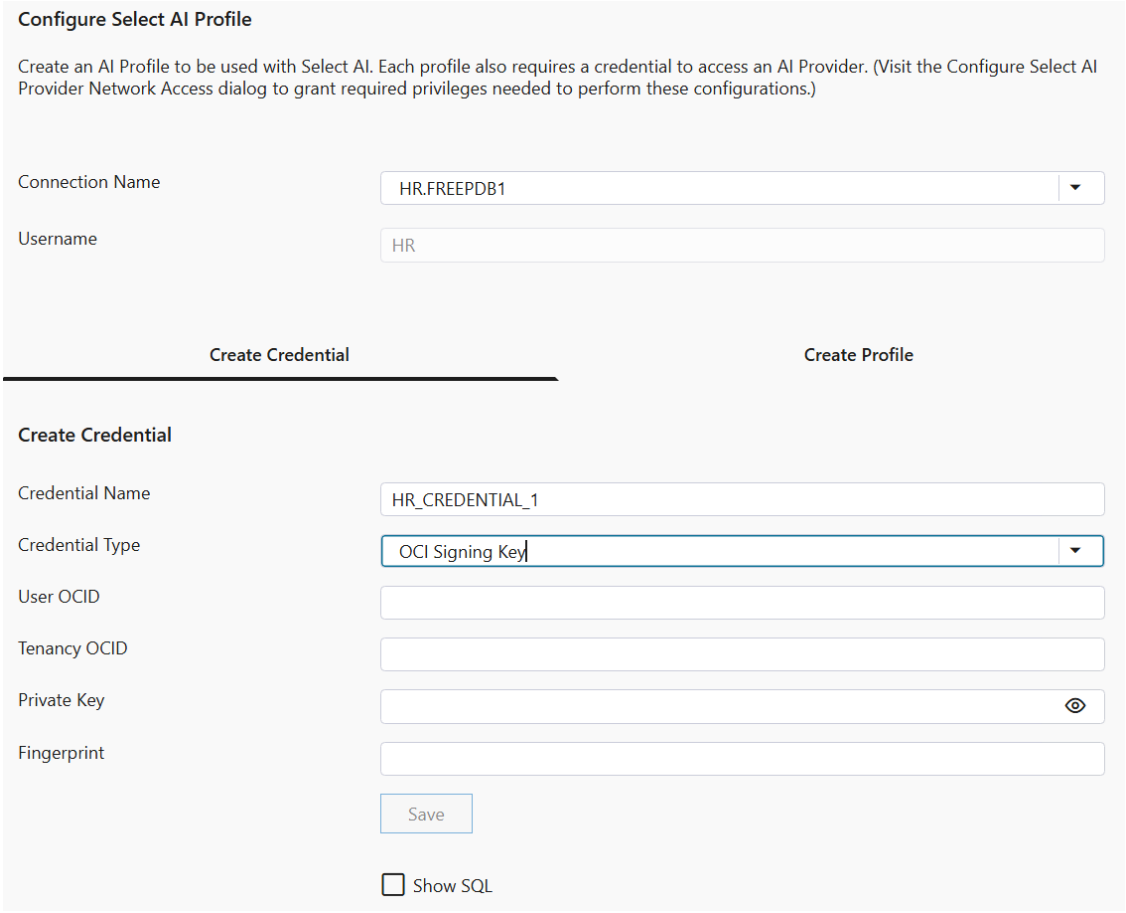
> **See Also**
>
> Using Select AI: Natural Language to SQL

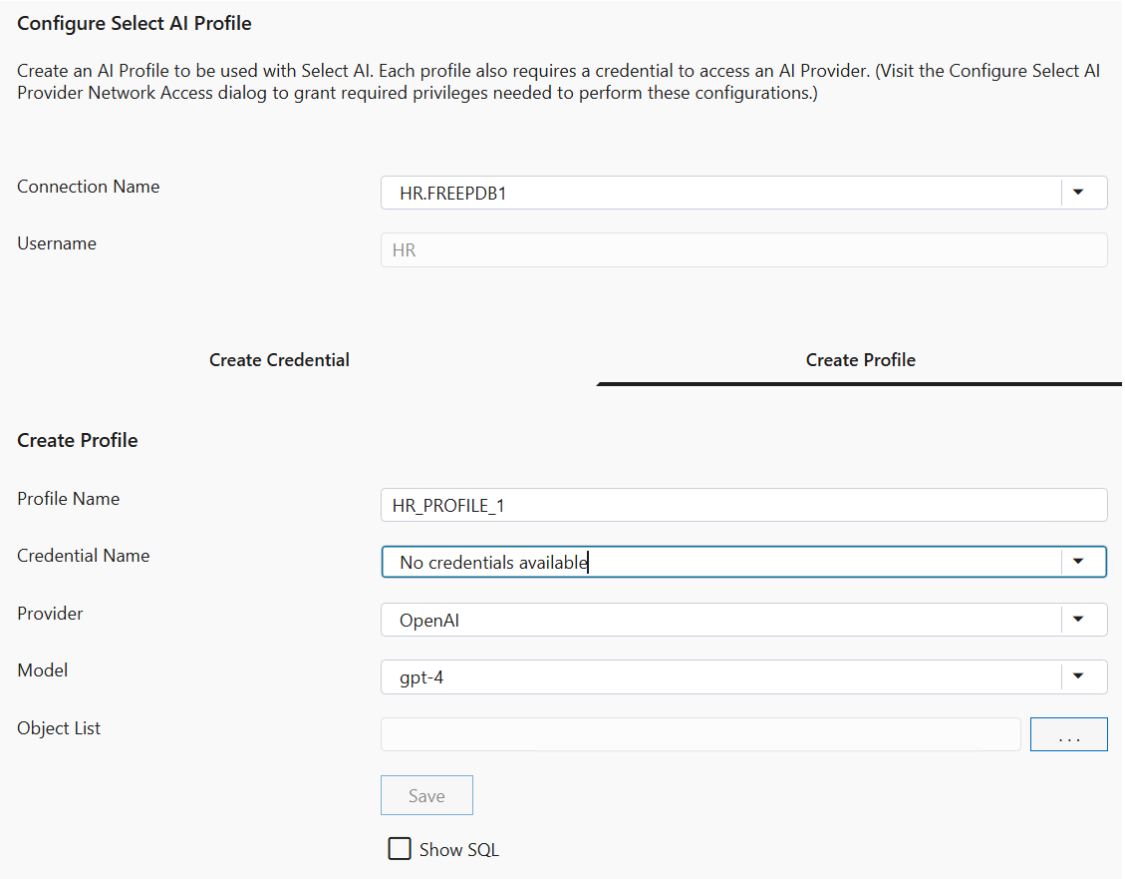
Opening the Configure Select AI Profile Dialog

To open the Configure Select AI Profile Dialog, right click on the database connection node in Server Explorer and in the menu choose **Select AI** and then **Configure Select AI Profile**.

Using the Configure Select AI Profile Dialog

The Configure Select AI Profile Dialog appears as follows:





The Configure Select AI Profile Dialog has the following controls:

Control	Description
Connection Name	This is the database connection that will be used when creating the profile.
Username	This is the database username that will own and use the profile.

The Configure Select AI Profile Dialog (Create Credential tab) has the following controls:

Control	Description
Credential Name	This is the name the credential will be stored under. You will reference this name when creating a profile.
Credential Type	This is the type of credential that will be used. Depending on the type, this dialog will show several additional fields specific to that credential type.
Save	Executes the commands required to create the credential.
Show SQL	Check this checkbox to show the SQL that will be executed when the Save button is pressed.

The Configure Select AI Profile Dialog (Create Profile tab) has the following controls:

Control	Description
Profile Name	Enter the name of the profile you wish to create.
Credential Name	Select a credential from the dropdown list. If none exist, click on the Create Credential tab and create a credential first.
Provider	Select the AI provider you wish to use.
Model	Select the name of the model you wish to use. If the model name does not appear in the list you may type it in the field instead.
Object list	Shows the list of database object whose metadata will be provided to the model.
Select Database Objects button	Click this button to open the Select AI Object List Dialog to select which Database Objects will have their metadata provided to the model. The model requires this metadata to be able to construct SQL queries in response to natural language queries.
Save	Executes the commands required to create the profile.
Show SQL	Check this checkbox to show the SQL that will be executed when the Save button is pressed.

Select AI Object List Dialog

The Select AI Object List Dialog allows you to select which database objects will have their metadata provided to the model. The model requires this metadata to be able to construct SQL queries in response to natural language queries. This dialog is accessed through the Configure Select AI Profile Dialog.

See Also

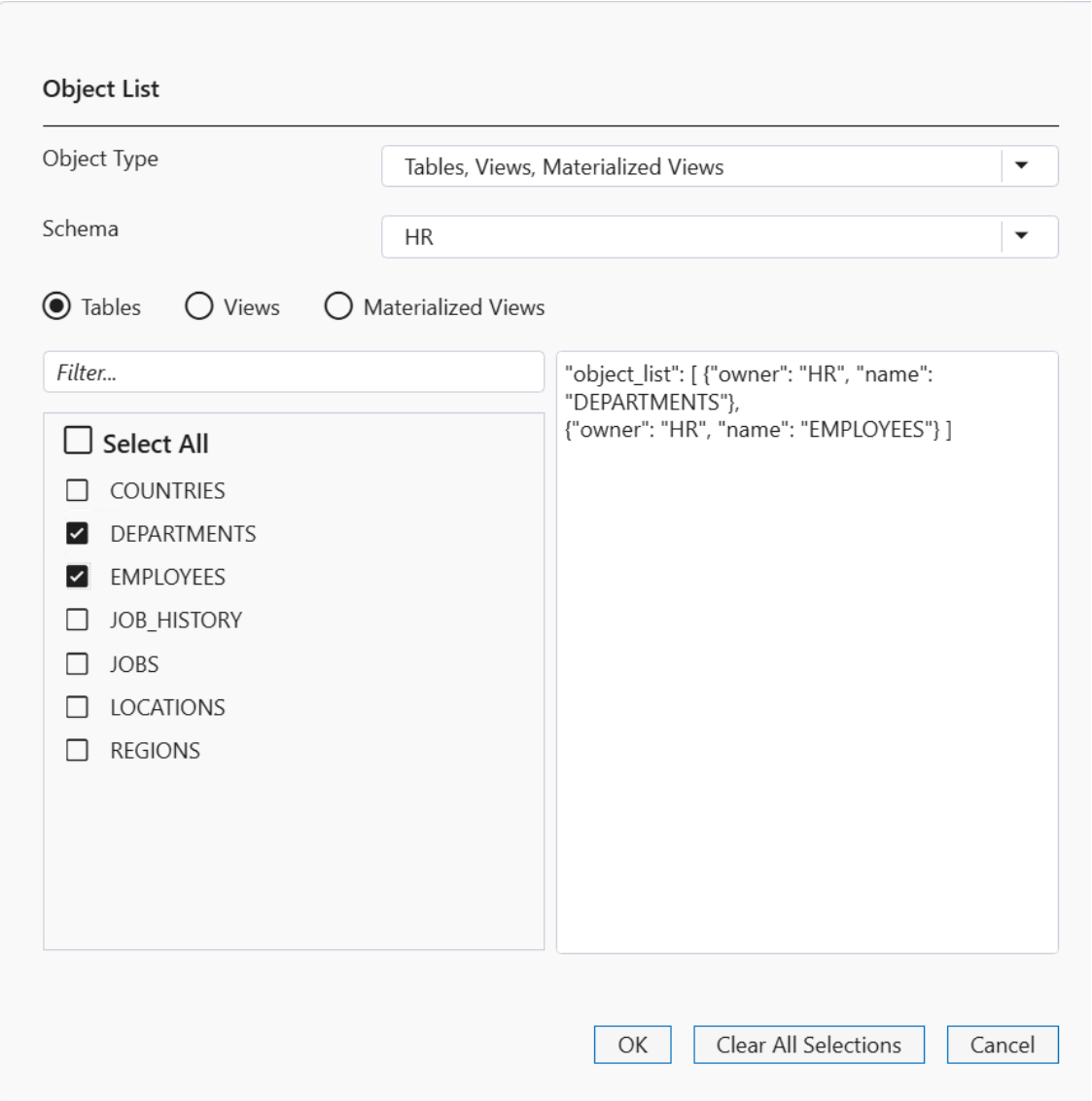
Using Select AI: Natural Language to SQL

Opening the Select AI Object List Dialog

The AI Object List Dialog is opened using the Configure Select AI Profile Dialog.

Using the Select AI Object List Dialog

The Select AI Object List Dialog appears as follows:



The Select AI Object List Dialog has the following controls:

Control	Description
Object Type	Select **Schemas** if you wish the database to provide metadata for all tables, views or materialized views within one or more entire schemas. Select **Tables, Views, Materialized Views** if you wish to select specific Tables, Views, or Materialized Views. You may choose both types as you build up your list, for example you might choose all objects in the HR schema and selected objects in the SCOTT schema.
Schema	If you selected Object type of **Tables, Views, Materialized Views**, then select the schema you wish to choose objects from.
Tables, Views, Materialized Views radio buttons	Select which type of object you wish to view.
Filter field	Enter a text string that will filter the list of objects or schema names

Control	Description
Select All checkbox	Check this checkbox to select all displayed objects or schemas
Ok	Closes the dialog and populates the Configure Select AI Profile Dialog with your choices.
Clear All Selections	Erases those selections you have already made.
Cancel	Discards your selections and closes the dialog.

Set Default AI Profile For Connection Dialog

The Set Default AI Profile For Connection dialog sets an AI Profile on a database connection in Server Explorer. After setting this profile, natural language queries (using SELECT AI) will be enabled in the SQL and PL/SQL File Editor. This profile will be used any time the connection is opened even across Visual Studio launches. Prior to using this dialog, you may need to create an AI profile using the Configure Select AI Profile Dialog.

See Also

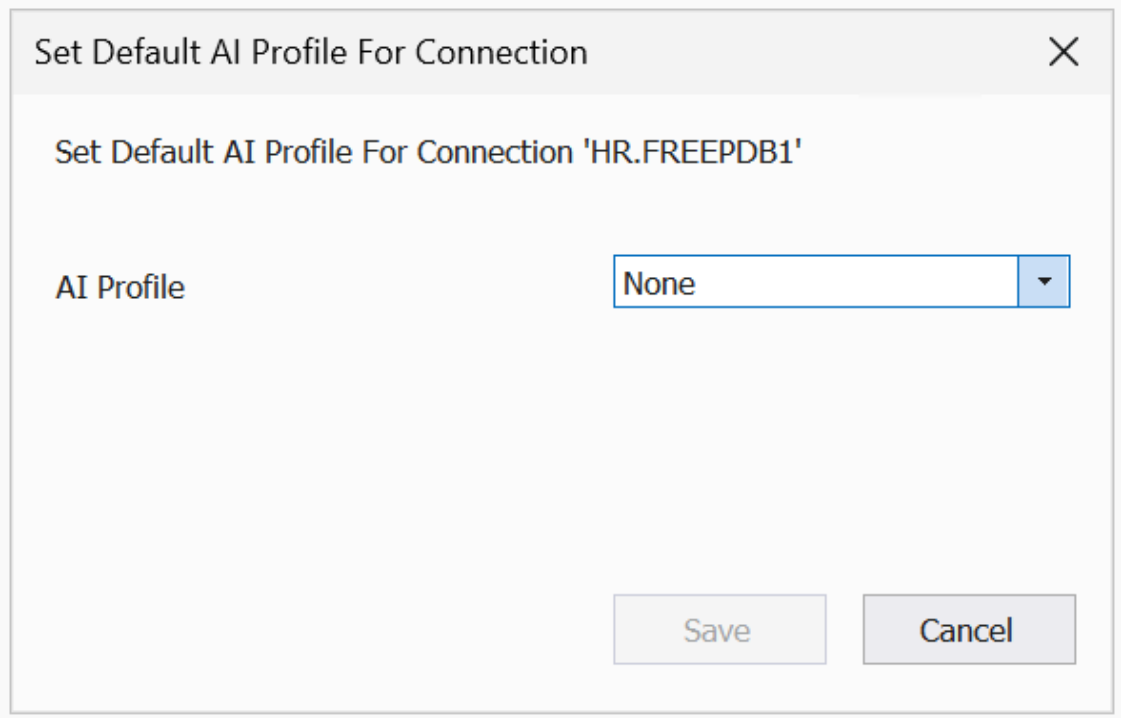
Using Select AI: Natural Language to SQL

Opening the Set Default AI Profile For Connection Dialog

To open the Set Default AI Profile For Connection dialog, right click on the database connection node in Server Explorer and in the menu choose **Select AI** and then **Set Default AI Profile For Connection**.

Using the Set Default AI Profile For Connection Dialog

The Set Default AI Profile For Connection dialog appears as follows:



The Set Default AI Profile For Connection dialog has the following controls:

Control	Description
AI Profile	Select the profile you wish to use from the drop down list. If you do not see a profile listed, you will need to create one via the Configure Select AI Profile Dialog. For more details, see Using Select AI: Natural Language to SQL .
Save	Associates the profile with this connection and closes the dialog. This profile will be used any time the connection is opened even across Visual Studio launches.
Cancel	Discards the changes and closes the dialog.

5

Oracle Data Window

About Oracle Data Window

In the Oracle Data Window, you can view and edit column data for all simple Oracle data types and the following complex Oracle data types: `RAW`, `LONG RAW`, and `XMLType`.

The column data for User-Defined Types (UDTs), `BFILE`, `LONG`, `LONG RAW`, `BLOB`, and `CLOB` data is read-only and can only be modified to the default column value or `NULL`.

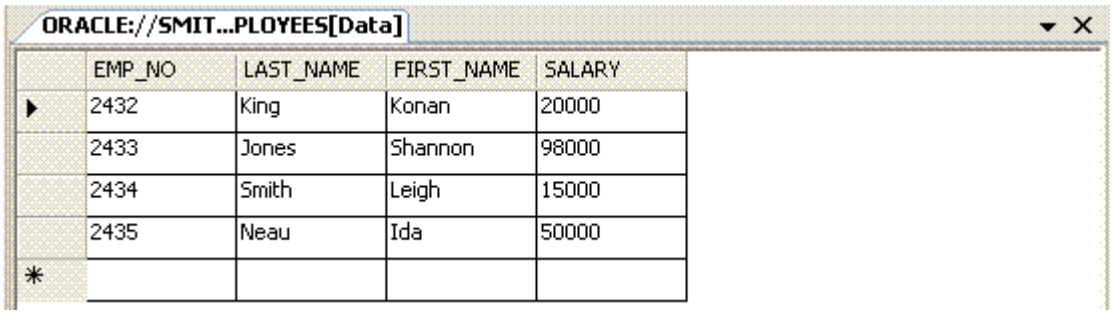
This section covers the following topics:

Opening Oracle Data Window

To open Oracle Data Window, do one of the following:

- Right-click the node representing the table or view whose data you want to view or modify, and from the menu, select **Retrieve Data**.
- Double-click the node that represents the table or view whose data you want to view or modify.

Oracle Data Window appears similar to the following:



	EMP_NO	LAST_NAME	FIRST_NAME	SALARY
▶	2432	King	Konan	20000
	2433	Jones	Shannon	98000
	2434	Smith	Leigh	15000
	2435	Neau	Ida	50000
*				

Using Oracle Data Window

When using Oracle Data Window:

- To edit data, select the cell you want and modify it as needed. You can insert `NULL` into the column by selecting the blank option from the drop-down list, or entering blank into the cell.

For new rows, you can insert the default value into the column by selecting `DEFAULT` from the drop down list.

- To add a row to a table, enter data in the empty last row.
- To remove a row, select the entire row by clicking on the row header at the left-hand side of the row and press **Delete**.
- Your changes are automatically committed to the database when you select a different row. To explicitly commit your changes to the database, click the **Save** button.
- The Oracle Data window does not allow you insert, update, or delete the data of views that have `INSTEAD OF` triggers.
- If the `INSTEAD OF` trigger is created on the view after you launched the data window, an error message appears, informing you that changes were not saved to the database.
- For UDT column data, the actual data is represented in XML format and includes only the level of nested structure.

Menu Options

Oracle Data Window provides the following menu items in the row header for each row of data:

Menu Option	Description
Design	Displays the Table Designer for tables or the View Designer for views.
	Moves cursor to the row of data.
Last	Moves cursor to the last row of data.
Row	Prompts you for the row of data to move the cursor, then moves the cursor to the data row number that you specify.
Refresh	Updates the table or view data displayed in the data window.
Cut	Applies only for a new row or when data is selected in a cell on any row. It copies the contents of the cell to the clipboard and clear the contents of the cell.
Copy	Applies only for a new row or when data is selected in a cell on any row. It copies the contents of the cell to the clipboard.
Paste	Applies only for a new row or when data is selected in a cell on any row. It copies the contents of the clipboard to the cell.
Delete	Deletes the current row of data.

See Also

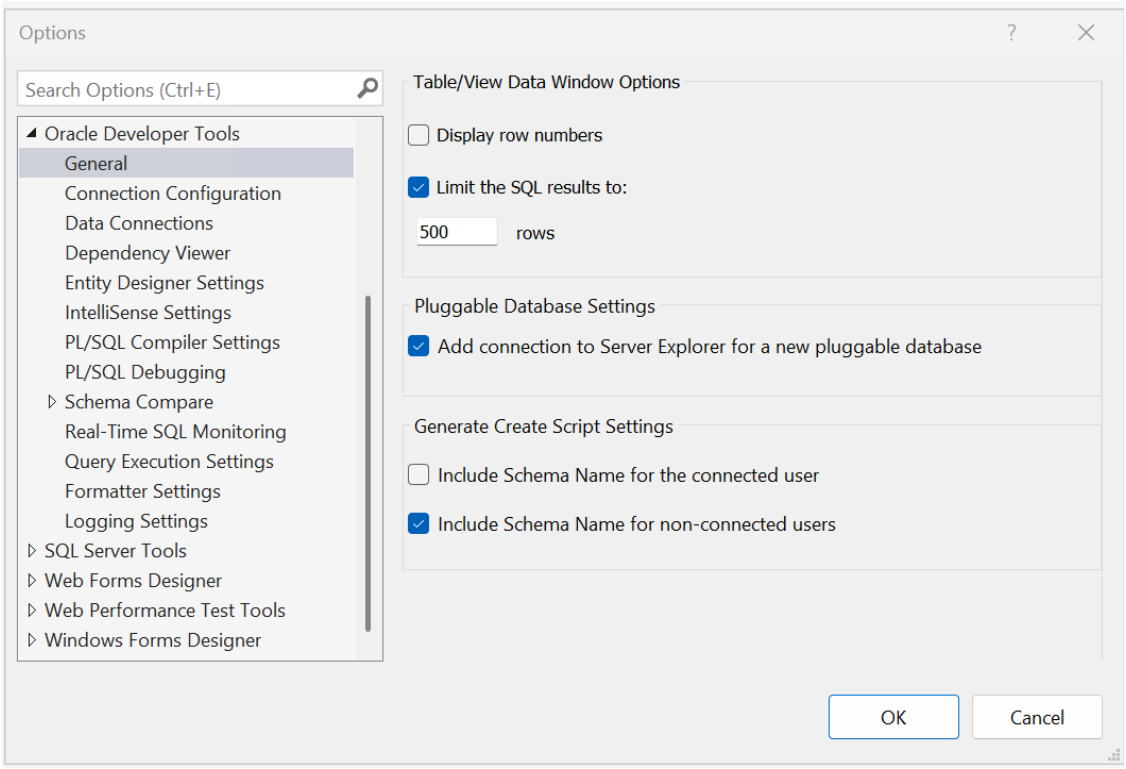
About Server Explorer | Table Designer |View Designer

General Options Page

The General Options page includes various settings that control how Oracle Developer Tools behaves.

Accessing General Options Page

To access the General Options Page, select **Options**... from the **Tools** menu. From the **Options** menu, select **Oracle Developer Tools**. Then select **General Options**.



Using the General Options Setting

The controls of the General Options are as follows:

Control	Description
Display Row Numbers	When enabled, the Data Window will include row numbers when displaying data.
Limit the SQL Results to:	When enabled, the Data Window limits results to the number of rows entered into the text box.
Add connection to Server Explorer for a New Pluggable Database	When enabled, creating a new pluggable database (see New Pluggable Database Dialog) results in a connection being added to Server Explorer for that pluggable database. If a TNS alias is used to connect to the container database, a new entry for the pluggable database is added to the `tnsnames.ora` file. If EZ connect is used to connect to the container, then EZ connect is used to connect to the pluggable database.
Include Schema Name for the connected user	When this is enabled, SQL scripts generated through Server Explorer menus will include the schema name of objects that are owned by the user connected in Server Explorer. This does not apply to Schema Compare nor Oracle Database Project Version 2 script generation.
Include Schema Name for non-connected users	When this is enabled, SQL scripts generated through Server Explorer menus will include the schema name of objects that are not owned by the user connected in Server Explorer. This does not apply to Schema Compare nor Oracle Database Project Version 2 script generation.

6

PL/SQL Code Editor

About PL/SQL Code Editor

Use the PL/SQL Code Editor to edit triggers, procedures, functions, packages, and package bodies. The PL/SQL Editor is database-backed, as opposed to file-based. Changes are saved to the Oracle Database. For file-based editing, such as editing SQL scripts, please use the Oracle SQL Editor.

To start the PL/SQL Code editor, see Starting the PL/SQL Code Editor.

The PL/SQL Code Editor provides several features that help you to quickly and efficiently write PL/SQL code, which include quick access to help information about the PL/SQL or SQL command you are entering.

After you edit and compile or save the PL/SQL code, Oracle Developer Tools commits the updated changes to the database.

You can use the Oracle PL/SQL Debugger to debug PL/SQL code.

See Also

Starting the PL/SQL Code Editor | PL/SQL Code Editor Features |Oracle PL/SQL Debugger | Oracle Query Window | *PL/SQL Database Language Reference* | *Oracle Database SQL Quick Reference* | *Oracle Database Error Messages*

Starting the PL/SQL Code Editor

The PL/SQL Code Editor appears in the Visual Studio Source Window when you select **Edit** from the drop-down menu of the nodes of the following schema objects:

- Functions
- Procedures
- Packages
- Package bodies
- Triggers
- Object Type Nodes
- Object Type Body Nodes
- VARRAY Type Nodes
- Nested Table Type Nodes

See Also

About Editing PL/SQL Code | PL/SQL Code Editor Features

Using PL/SQL Code Editor

This section covers the following topic:

- PL/SQL Code Editor Features

PL/SQL Code Editor Features

The PL/SQL Code Editor provides several features that help you to quickly and efficiently write PL/SQL and SQL code:

- **Syntax coloring:** The PL/SQL Code Editor displays PL/SQL and SQL reserved words in color. Identifiers containing multibyte characters need to be enclosed in double quotes.
- **Brace Matching Feature:** The PL/SQL Code Editor supports highlighting of `BEGIN-END` blocks and matching braces where needed.
- **Collapsible regions:** You can hide or show blocks of PL/SQL code by clicking a plus (+) or minus (-) icon next to the code.
- **IntelliSense:** IntelliSense offers smart suggestions and code completion for SQL and PL/SQL. As you type SQL or PL/SQL you can take utilize IntelliSense like so:
  - As you type, IntelliSense suggestions will automatically appear. You may also press Control+spacebar to obtain suggestions. You can select from the suggestions using arrow keys and the return key or by using the mouse.
  - Type a schema name or object name followed by a period (".") to receive suggestions. In the case of procedure or function names, the full signature of the procedure or function name will be outputted along with placeholder values.
  - After typing a procedure or function name, the left parenthesis ("(") will trigger parameter info to be displayed. The current parameter you are entering will be boldfaced.
  - Hover the mouse over variables and object names to display Quick Info about the variable or object name.

  IntelliSense metadata is cached. If you have recently modified database objects, click the **Rebuild Intellisense** toolbar icon or menu item to receive up to date suggestions.

  To change the casing of suggestions, for example to receive suggestions in lower case, see the IntelliSense Settings Options Page.
- **Compile command:** To compile your PL/SQL code and write it to the database, right-click within the PL/SQL Code Editor window and select **Compile**.
- **Error display:** The Oracle Output Window and Task Pane display errors. To jump to a specific error, double-click its error message to move the cursor to the corresponding source line of code. Errors display until the next time you save or compile the PL/SQL code.
- **Lists of Names:** The PL/SQL Code Editor displays lists at the top which contain the following information.
  - The name of a stored procedure or function, if one is open, in the list on the left. The list on the right is empty.

- The name of the package or package body, if one is open, in the list on the left. The list on the right contains entries for all the procedures and functions declared or defined in the package.

The list on the right can be used to navigate to different procedures and functions.

PL/SQL Compiler Settings Options Page

This section describes the compiler options found on the PL/SQL Compiler Settings page.

See *Oracle Database PL/SQL Language Reference* for detailed information regarding these settings.

About PL/SQL Compiler Settings

The PL/SQL compiler settings page allows you to set the default compile settings that are used when you select the compile or compile debug menu item in Server Explorer. These settings are also used from the PL/SQL Editor.

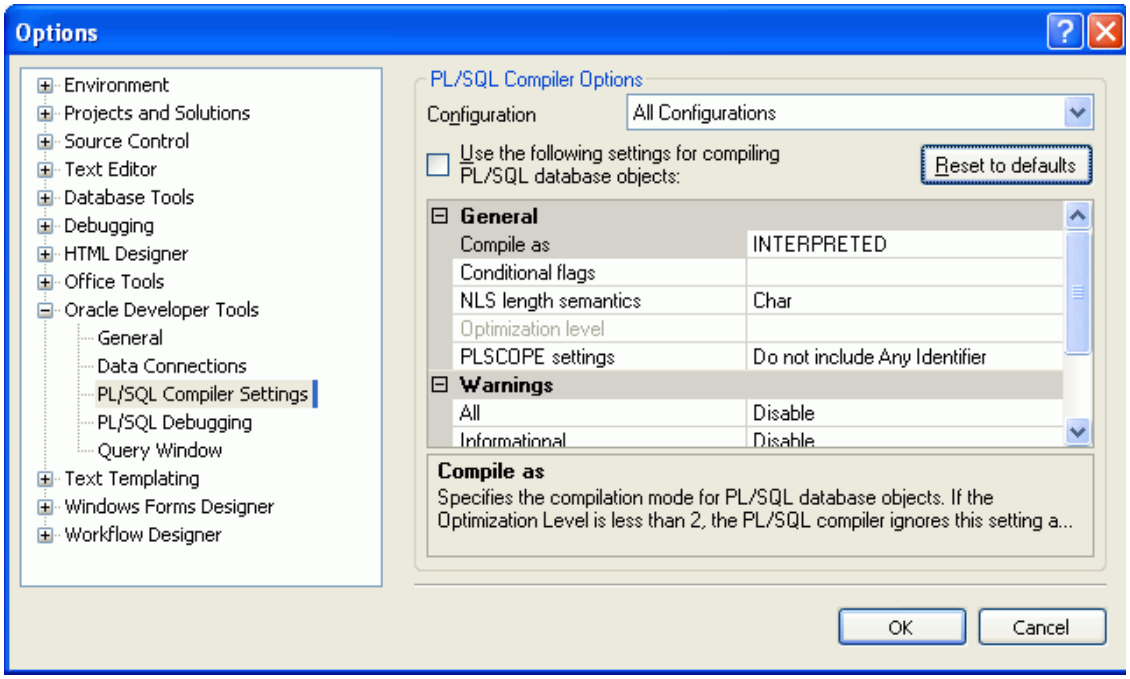
The settings defined in this options page normally carry forward from one Visual Studio session to another.

The PL/SQL Compiler Settings option page also provides a check box to allow you specify whether or not to use these parameters for compile or compile debug operations. A checked box indicates that the parameters specified in this options page are used for compile and compile debug operations. An unchecked box indicates that the code object is compiled using the setting defined in the database session.

> **Note**
>
> These compiler settings only apply to compiling from Server Explorer and the PL/SQL Editor. They do not apply to the SQL*Plus script editor.

Accessing PL/SQL Compiler Settings

To access the PL/SQL Compiler Settings Page, select **Options**... from the **Tools** menu. From the **Options** menu, select **Oracle Developer Tools**. Then select **PL/SQL Compiler Settings**.



Using PL/SQL Compiler Settings

The controls of the PL/SQL Compiler Settings are as follows:

Control	Descriptions
Configuration	Select Compile or Compile Debug, or All Configurations (both) to indicate which compile configurations the settings on this page apply to.
Use the following settings for compiling PL/SQL database objects	Enable the settings on this page. The PL/SQL database objects are compiled with these settings. If this is unchecked (the default), then they are compiled with the parameters settings of the session.
Reset to defaults	Resets to the compile settings to defaults.
Compile As	Select the `PLSQL_CODE_TYPE` parameter, either `NATIVE` or `INTERPRETED` (default).
Conditional Flags	Enter the conditional compilation flags to represent the `PLSQL_CCFLAGS` parameter. Default is an empty string.
NLS length semantics	Select either `Byte` or `Char` (Default).
Optimization Level	Indicate the `PLSQL_OPITIMIZE_LEVEL` parameter. Available values are: 0 and 1 (default) for Compile Debug, 2 and 3 (default) for Compile.
PLSCOPE settings	Select either Include All Identifiers or Do not include Any Identifier (Default).
All	Specify if all warnings are to be `ENABLED`, `DISABLED` (default), or treated as an `ERROR`.
Severe	Specify if `SEVERE` warnings are to be `ENABLED`, `DISABLED` (default), or treated as an `ERROR`.
Informational	Specify if `INFORMATIONAL` warnings are to be `ENABLED`, `DISABLED` (default), or treated as an `ERROR`.
Performance	Specify if `PERFORMANCE` warnings are to be `ENABLED`, `DISABLED` (default), or treated as an `ERROR`.

Control	Descriptions
Others	Specify if specific warning types are to be `ENABLED`, `DISABLED` (default), or treated as an `ERROR`. You can enter different combinations of values.
(description window)	Displays information about the compile option selected.
OK	Saves settings.
Cancel	Cancels settings.

7

Oracle PL/SQL Debugger

About the Oracle PL/SQL Debugger

Oracle Developer Tools for Visual Studio includes an integrated PL/SQL debugger. This enables you to debug PL/SQL programs in the Visual Studio environment, in much the same manner as you would debug a C# or any other language.

This help describes the features of the Oracle PL/SQL Debugger.

This section describes the setup of the Oracle PL/SQL Debugger, introduces the three debugger modes, describes their operation, and provides troubleshooting guidance.

See Also

About PL/SQL Code Editor | *Oracle Database Application Developer's Guide - Fundamentals*, Chapter 7, Debugging Stored Procedures

This section includes the following topics:

PL/SQL Debugging Setup

In order to debug a PL/SQL program in an Oracle Database, you must do the following:

- Grant database privileges that allow you to debug programs.
- Compile the program with *debug information*. You can use the Oracle Developer Tools Compile Debug command to compile the program with debug information.
- In the PL/SQL Debugging Options page, set a valid IP address, port range, and indicate which Server Explorer connection contains the PL/SQL code that will be debugged.

This section includes the following topics:

Granting Database Privileges for Debugging

Performing PL/SQL debugging requires certain database privileges. These can be easily granted through the Grant Debugging Privileges Dialog . The specific privileges are as follows:

- The `DEBUG CONNECT SESSION` privilege is required. This can be granted with `GRANT DEBUG CONNECT SESSION TO` *`username`*.
- The `DEBUG ANY PROCEDURE` privilege or both `DEBUG` and `EXECUTE` privileges on the PL/SQL program being debugged is required. These can be granted with `GRANT DEBUG ANY PROCEDURE TO` *`username`*
- In Oracle Database 12.1 or later, additional privileges are required which grant the database explicit access to an IP address and port range on the computer where Visual Studio is installed. These privileges can be granted through the Grant Debugging Privileges Dialog or by manual execution of the `DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE` package procedure as SYSDBA.

  For example, execute the following command as SYSDBA, replacing the host, lower port, upper port, and principal name:

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
    HOST => '255.255.255.25',
    LOWER_PORT => 61000,
    UPPER_PORT => 65000,
    ACE => XS$ACE_TYPE(PRIVILEGE_LIST => XS$NAME_LIST('jdwp'),
                       PRINCIPAL_NAME => 'HR',
                       PRINCIPAL_TYPE => XS_ACL.PTYPE_DB));
END;
```

  To revoke this privilege, execute:

```
BEGIN
   DBMS_NETWORK_ACL_ADMIN.REMOVE_HOST_ACE(
     HOST => '255.255.255.25',
     LOWER_PORT => 61000,
     UPPER_PORT => 65000,
     ACE => XS$ACE_TYPE(PRIVILEGE_LIST => XS$NAME_LIST('jdwp'),
                        PRINCIPAL_NAME => 'HR',
```

```
                                PRINCIPAL_TYPE => XS_ACL.PTYPE_DB));
END;
```

For more information on the `DBMS_NETWORK_ACL_ADMIN` package, see *Oracle Database PL/SQL Packages and Types Reference*.

Set PL/SQL Debugging Options

A listener is started in Visual Studio for debugging session connections from the database. To ensure that the database can connect to this listener, a valid IP address and port number must be provided. Additionally, Oracle Developer Tools needs to know which Server Explorer connection contains the PL/SQL code to be debugged. All of these values should be set in the PL/SQL Compiler Settings Options Page.

Compiling a PL/SQL Program with Debug Information

You can compile a PL/SQL program so that it does or does not provide debug information in the results. Oracle PL/SQL Debugger cannot debug programs without this debug information.

Compiling with Debug Information

There are two compile commands: Compile and Compile Debug. Both commands compile the PL/SQL program in the database, except that the Compile Debug command provides debug information in its compilation.

> **Note**
>
> If the Compile Debug command is executed, but the user does not have the proper debug privileges, then compile will fail and the program cannot be debugged.

To compile a PL/SQL program, do one of the following:

- From Server Explorer, expand the node that contains the program being compiled, right-click on that program, and from the menu, select **Compile** or **Compile Debug**.
- From a program source window, right-click, and from the menu, select **Compile** or **Compile Debug**.

> **Note**
>
> Oracle Developer Tools only compiles the top-level PL/SQL program.

> **Note**
>
> You cannot compile during debugging. Some menus are disabled (greyed out) to reflect this.

> **Note**
>
> Compiler Settings affecting Compile and Compile Debug operations may be modified through the PL/SQL Compiler Settings Options Page. For more information see PL/SQL Compiler Settings Options Page.

See Also

About Server Explorer

Viewing Compile Debug Property Information

There are several ways that you can view property information for the Compile Debug option to verify whether a program has been compiled with or without debug information. This is typically needed when debugging has been completed, and it is necessary to recompile normally any unit that was previously compiled with debug information.

Server Explorer uses distinct debug icons to indicate at a glance whether or not the program is compiled with debug information.

For example, to view the icons, expand the Procedures node from Server Explorer. The icons appear before the names of the node. Any program unit that been compiled for debug is identified by a distinct icon.

To view the Compile Debug Option property, expand the Procedures node from Server Explorer, right-click on the PL/SQL program, and from the menu, select **Properties**. The Properties list appears.

Debugging Setup Checklist

There are several steps required to configure PL/SQL debugging for first use. If one of the steps is skipped, several possible errors can occur.

To avoid any missing steps and get started debugging your PL/SQL code more quickly, perform the following steps:

1. Make sure the database you use is version 10.2 or later.
2. If you have `SYSDBA` privileges on the database:

   In a `SYSDBA` connection in Server Explorer, from the context menu for the User node (in Objects View) or a Schema node (in Schemas view), representing the user who will be doing the PL/SQL debugging, select **Grant Debugging Privileges** menu item.

   In the Grant Debugging Privileges Dialog that appearsprovide the IP Address for the computer where Visual Studio is installed and the starting port and ending port numbers representing the range of ports Oracle Database could use to connect back to that computer.
3. If you do *not* have `SYSDBA` privileges on the database, ask your DBA to execute the following commands as `SYSDBA` for the user who will be doing the debugging:

   ```
   grant debug any procedure to username
   ```

   and

   ```
   grant debug connect session to username
   ```

For the command, as an alternative, both debug and execute permissions may be granted to the user, on the PL/SQL program being debugged.

Additionally, ask your DBA to execute the following commands as SYSDBA, replacing the host, lower port, upper port, and principal name:

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.APPEND_HOST_ACE(
    HOST => '255.255.255.25',
    LOWER_PORT => 61000,
    UPPER_PORT => 65000,
    ACE => XS$ACE_TYPE(PRIVILEGE_LIST => XS$NAME_LIST('jdwp'),
                       PRINCIPAL_NAME => 'HR',
                       PRINCIPAL_TYPE => XS_ACL.PTYPE_DB));
END;
```

4. Compile the PL/SQL program units with debug information, as follows:

   In Server Explorer, find the node that represents the package, procedure, or function that you want to debug. Right-click it, and from the menu select **Compile Debug**. The icon next to the PL/SQL procedure or function changes to indicate that it has been compiled with debug information.

5. Set the PL/SQL Debugging Options from the Visual Studio Tools menu by selecting **Options**, then **Oracle Developer Tools** in the Options list, and performing the following:

   a. From the IP Address list, choose the IP address for this computer.

      During PL/SQL debugging, the Oracle Database connects to Visual Studio using this IP address. IPv6 addresses are only supported with Oracle Database 11*g* Release 2 (11.2) or later.

      Note: For Oracle Database 12.1 or later, this must be an IP address that was granted privileges in steps 2 and 3 above.

   b. Make sure that the port range specified represents open ports on the computer that Visual Studio is installed on and that the ports are not blocked by a firewall.

      During PL/SQL debugging, the Oracle Database connects to Visual Studio through TCP/IP on a random port within the specified range.

      Note: For Oracle Database 12.1 or later, this port range must be equal to or a subset of the port range that was granted privileges in steps 2 and 3 above.

   c. Check the box next to your connection in the Available database connections list.

      This tells the PL/SQL debugger to look for the PL/SQL code using this connection.

6. If your procedure or function contains input parameters that can be entered by hand (that is, scalar values), begin debugging by right-clicking on the procedure or function name in Server Explorer and choosing **Step into** or **Run Debug** (if a break point is set). This is known as "Direct Database Debugging Mode".

7. If you intend to debug a non-website client-server (Console, WinForm, and so on) application using a single instance of Visual Studio to step from .NET code into PL/SQL code and back ("Multitier Application Debugging Mode"), you must perform steps 8 through 13 in this checklist. If the PL/SQL is being called by a middle tier and/or you wish to use two instances of Visual Studio, one for .NET and one for PL/SQL ("External Application Debugging Mode"), skip to step 14 of this checklist.

8. From the Visual Studio main menu, select **Project**, then select **Properties** (the .NET application project must be loaded).

Click the **Debug** tab and uncheck Enable the Visual Studio hosting process in the Enable Debuggers list. If you do not do this, you must run the debugger once and then stop it in order to make it work correctly.

9. Turn on the Application level debugger by selecting **Tools**, then **Oracle Application Debugging** from the Visual Studio menu, and ensure there is a check next to Oracle Application Debugging.
10. Verify that the ODP.NET config file (`machine.config`, `web.config`, or `app.config`) of the application does not have the `ORA_DEBUG_JDWP` value set. If it is set, disable it.
11. Set a breakpoint in the PL/SQL program. You can also set breakpoints in your .NET application code at some line of code *after* the call to the PL/SQL program, if you want. That stops execution after you return from debugging the PL/SQL.
12. Build and begin debugging your .NET application, and the PL/SQL breakpoint will fire. You can then look at the live data being passed back and forth between your .NET application and the PL/SQL procedure, through the watch window. In case of problems, examine the Oracle Output Window.
13. When done debugging, select **Tools**, then uncheck **Oracle Application Debugging** from the Visual Studio menu.
14. If you wish to debug a Web application or otherwise use a dedicated Visual Studio instance only for PL/SQL debugging (and another Visual Studio instance for .NET debugging), perform the following steps:
    a. In the menu for the Visual Studio instance that will debug the PL/SQL, select **Tools**, then check **External Application Debugging**.
    b. Provide IP address and port number.

       Note: For Oracle Database 12.1 or later, the IP address and port number must have been given permissions in steps 2 and 3 above.
    c. For a web application or any other application that is calling a PL/SQL Stored procedure or function, make sure that `ORA_DEBUG_JDWP` is set either as an environment variable or in the ODP.NET config file (`machine.config`, `web.config`, or `app.config`) of the application. This value must be set before the application or website connects to Oracle. A config file value will take precedence over any environment variable.

       Note: Only ODP.NET version 12.1.0.2 or later supports the `ORA_DEBUG_JDWP` config file entry. For earlier versions, use an environment variable instead.

       Config file example:

       ```
       <oracle.manageddataaccess.client>
         <version number="*">
           <settings>
             <setting name="ORA_DEBUG_JDWP" value="host=127.0.0.1;port=1234"/>
           </settings>
         </version>
       </oracle.manageddataaccess.client>
       ```

       Environment variable example:

       ```
       set ORA_DEBUG_JDWP=host=127.0.0.1;port=1234
       ```
    d. Set a breakpoint in your PL/SQL program.
    e. Run the web application or external application and connect to Oracle. In case of problems, examine the Oracle Output Window.
    f. After the breakpoint fires and you are done debugging, stop debugging from the Visual Studio Menu.

PL/SQL Debugging Options

This section describes the Debug options which are found on the PL/SQL Debugging Options page at Accessing PL/SQL Debugging Options. For more information on PL/SQL debugging, see About the Oracle PL/SQL Debugger.

About Debugger Options

There are three debugger options or settings that you can make.

- Debugger IP Address Configuration

  The Oracle PL/SQL debugger requires a valid TCP/IP address for debugging. A listener is started on the machine that hosts Visual Studio using this IP address and the Oracle Database connects back to Visual Studio using this IP address.

  You can specify one of the TCP/IP addresses on your computer for PL/SQL debugging in the PL/SQL Debugging Options page. See Accessing PL/SQL Debugging Options in the next section.

  If you do not specify a TCP/IP address in the PL/SQL Debugging Options page and you have more than one TCP/IP address on your computer, a dialog box will prompt you to select the TCP/IP address for PL/SQL debugger.

  In most cases the IP address used to start a listener on the machine that hosts Visual Studio is the same as the IP address used by Oracle Database to connect back to Visual Studio. However, in some cases, such as when using the database in a container, they might be different. In these cases, modify **Hostname or IP Address as known by the database** option.

  Changing the TCP/IP address only affects subsequent PL/SQL debugging instances.

- Debugger Port Configurations

  The Oracle PL/SQL debugger requires an open TCP port for debugging. The Oracle Database connects to Visual Studio using this port number.

  For Direct Database and Multitier application debugging, you can specify a range of port numbers in the PL/SQL Debugging Options page. If you do not specify the port number, the Oracle PL/SQL debugger randomly opens an available port, based on the default port range. The default port range is 65000 to 65535. See Accessing PL/SQL Debugging Options in the next section.

  Changing the debugger port only affects subsequent PL/SQL debugging instances.

  When you start debugging, if there is no available port in the range that you specified, a dialog box appears indicating that debugging cannot be started because the specified debugging port is unavailable. For remote database connection, the TCP port must be open for external access.

  When debugging starts, the Debug Output window displays the IP address and the port number.

> **Note**
>
> For External Application Debugging Mode, you set the port number as part of the start process.

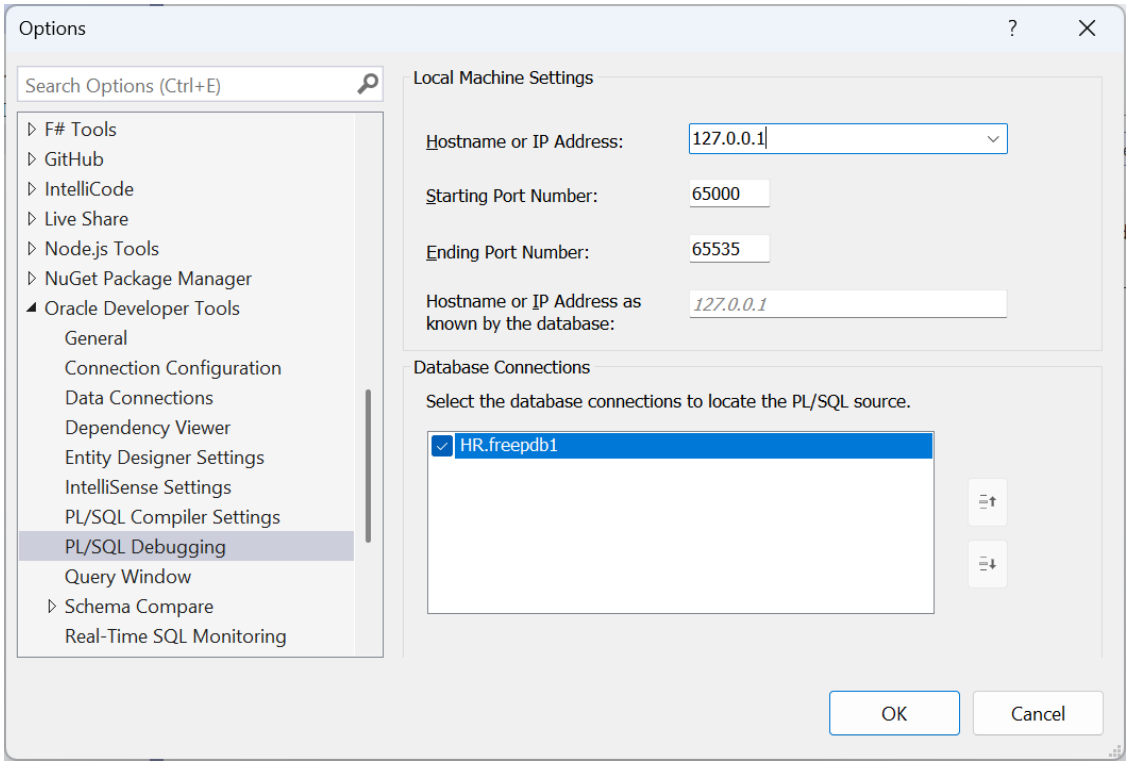
- PL/SQL Source Location

  You can specify the database connection or connections which are possible locations for the PL/SQL source. The choice of connections corresponds to those listed in Server Explorer. During debugging, Oracle Developer Tools looks for the PL/SQL source in the order that you have specified.

  If you do not specify a database connection, Oracle Developer Tools displays a message indicating that the source program could not be located.

Accessing PL/SQL Debugging Options

To access the Debug Options Page, select **Options**... from the **Tools** menu. From the **Options** menu, select **Oracle Developer Tools**. Then select **PL/SQL Debugging Options**.



You can change the IP address and the starting and ending port numbers. You can check which database connections could be the source locations, and, you can change the order of database connections by selecting one and clicking **Move Up** or **Move Down**. This determines the order that the debugger searches for PL/SQL code.

Using the PL/SQL Debugging Options

The controls in the Debug tab of the PL/SQL Debugging Options are as follows:

Control	Description
Hostname or IP Address	The hostname or IP address of the machine where Visual Studio is located.

Control	Description
Starting Port Number	Enter the starting number for a range of ports to be available for the Oracle PL/SQL Debugger. By default, the starting number is 65000.
Ending Port Number	Enter the ending port number for the Oracle PL/SQL Debugger. By default, the ending number is 65535.
Hostname or IP Address as known by the database	This is the IP Address or hostname that the database will use to connect back to the machine Visual Studio is located on. In most cases this will be the same value as the Hostname or IP address field above. In other cases, such as when using the database in a container, it might be different.
Database Connections	Specify a database connection as a location for a PL/SQL source by checking the database connection box. Set the order of database connections that the debugger searches by selecting a connection and clicking **Move Up** or **Move Down**. To deselect a database connection, uncheck the database connection box.

See Also

Connection Dialog Box | Connecting to Oracle Database from Visual Studio

Debugger Modes

The Oracle PL/SQL Debugger operates in these three modes:

- Direct Database Debugging

  The Direct Database Debugging mode allows you to debug PL/SQL code directly from Server Explorer. When you use the Direct Database Debugging mode, you do not need a Visual Studio solution or .NET code. By right-clicking on a procedure or function in Server Explorer, you can step into a PL/SQL procedure or function. Any arguments required by the procedure must be entered by hand. This limits the use of this feature to procedures or functions that have scalar parameter values.

- Multitier Application Debugging

  This mode allows you to seamlessly debug both .NET and PL/SQL code from within a single Visual Studio instance. You can step directly from your .NET code into the PL/SQL code and back out again passing live data and debugging that data.

- External Application Debugging

  External Application Debugging allows you to dedicate an instance of Visual Studio solely for waiting on a breakpoint to be called in a PL/SQL stored procedure or function, and then debug it with that instance of Visual Studio. This is particularly useful when the stored procedure or function is called by a middle tier web server. However any Oracle client running on any operating system may call the stored procedure or function.

See Also

Direct Database Debugging Mode | Multitier Application Debugging Mode | External Application Debugging Mode

Direct Database Debugging Mode

Direct Database Debugging mode allows you to debug PL/SQL programs by using the Step Into command from either a PL/SQL program node in Server Explorer or a PL/SQL program source window.

Before debugging starts, Oracle Developer Tools saves and recompiles the PL/SQL program if the PL/SQL program is updated in Oracle Developer Tools, but has not yet been saved in the database.

This section includes the following topics:

Requirements for Direct Database Debugging Mode

Prior to using Direct Database Debugging, perform all the setup requirements listed in PL/SQL Debugging Setup.

Debugging a PL/SQL Program Directly

This section describes how to debug a PL/SQL program directly. You can begin by executing either the Step Into or Run Debug command.

Both the Step Into and Run Debug command launch the Oracle PL/SQL Debugger:

- **Step Into**

  Stops at the line of the program being debugged.

- **Run Debug**

  Stops at the breakpoint encountered in the program being debugged or in programs that are called by that program.

  If the program is compiled for *release*, and you do not to recompile it with debug information, then the execution stops at the breakpoint encountered in any called program that is compiled with debug information.

> **Note**
>
> To debug triggers, you cannot use Step Into. You must set a breakpoint in the trigger and execute Run Debug. Execution will stop in the trigger when the trigger is fired.

To begin debugging a PL/SQL program, do one of the following:

- From Server Explorer: Expand the node that contains the program being debugged, right-click on that program, and from the menu, select **Step Into** or **Run Debug**.

  To step into a package method, right-click on the method, and select **Step Into** or Run Debug from the menu.

> **Note**
>
> You cannot directly run from Server Explorer or debug any PL/SQL program that contains composite type parameters as `IN` or `INOUT` parameters. You can debug this by creating a PL/SQL wrapper or calling the PL/SQL program from .NET code.

- **From a source window**: Right-click, and from the menu, select **Step Into** or **Run Debug**.

  This option is not available to debug a package procedure or function. Use Server Explorer instead.

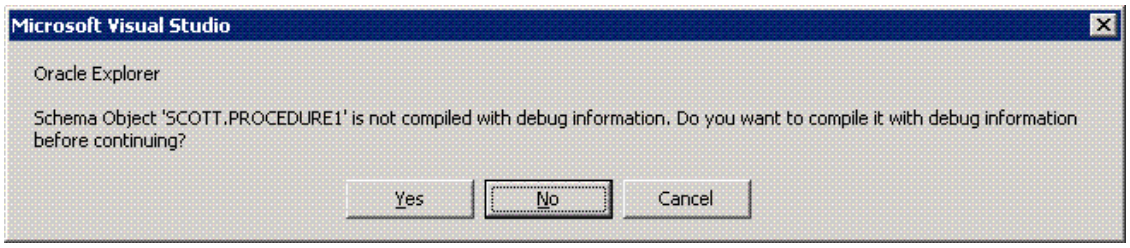
When execution stops, in the PL/SQL program you can proceed to debug the PL/SQL program. See Common PL/SQL Debugger Operations.

See Also

PL/SQL Program Refresh Behavior |Stepping Through a PL/SQL Program

Behavior for PL/SQL Programs Compiled Without Debug Information

If you execute the Step Into or Run Debug commands on a PL/SQL program that is not compiled with debug information, a message box appears indicating this, and asks if you want to compile it with debug information.



Depending on your response, the Oracle PL/SQL Debugger behaves as follows:

Cancel: If you select Cancel or close the message box, then no action is performed and the debugging session is canceled.

No: If you select No, then the code executes and does not compile with debug information.

The execution does not stop at the beginning or at the breakpoints inside the PL/SQL program. However, if this program calls another PL/SQL program that has been compiled with debug information, then the execution stops at the breakpoints set in the that program.

Yes: If you select Yes, then the program is recompiled with debug information, it is launched, and the execution stops at the beginning of the program.

See Also

Stepping Through a PL/SQL Program

Multitier Application Debugging Mode

The Multitier Application Debugging mode allows you to seamlessly debug both .NET and PL/SQL code from within a single Visual Studio instance. You can step directly from your .NET

code into the PL/SQL code and back out again, passing live data and debugging that data. This is most useful in client server applications (rather than ASP.NET web applications).

This section contains the following topics:

How Multitier Application Debugging Works

When an application being debugged from within Visual Studio executes a PL/SQL program that contains an enabled breakpoint, the debugging process switches from the application tier (for debugging the application) to the database tier (for debugging the PL/SQL program). It stops at the enabled breakpoint of the PL/SQL program.

From there on, you can use the Oracle PL/SQL Debugger to control the PL/SQL program execution and examine debugging information. When the PL/SQL program finishes executing, the debugger switches back to the application tier.

The Debug Location Toolbar displays *Oracle Debugger* once the Oracle PL/SQL Debugger starts. You can use the Debug Location Toolbar to switch between the tiers.

Requirements for Multitier Application Debugging

Prior to using Multitier Database Debugging, perform all the setup requirements listed in PL/SQL Debugging Setup.

Before Using Multitier Application Debugging

Before using Multitier Application Debugging, you must do the following:

- Set breakpoints

  In multitier application debugging, the Step Into command does not work on a PL/SQL program called from an application. Therefore, you must set at least one breakpoint each in:

  – The PL/SQL program being called from the application.

  – The calling application, at some point after the call to the PL/SQL procedure or function.

  If there is no breakpoint in the application, the debugger does not stop when it returns to the application tier, it simply continues to execute.
- Select **Application Debugging** from the **Tools** menu to enable multitier application debugging.
- Verify that the ODP.NET config file (`machine.config`, `web.config`, or `app.config`) of the application does not have `ORA_DEBUG_JDWP` value set. If it is set, disable it.

Debugging a Multitier Database Application

For a detailed guide in using multitier database application debugging, follow the steps in the Debugging Setup Checklist.

External Application Debugging Mode

This debugging mode allows you to dedicate an instance of Visual Studio to debug PL/SQL programs that are called by external applications running on any platform, local or remote. Typically, the procedure is called by a web application on a middle tier.

External applications use one of two possible ways to connect to the Oracle PL/SQL Debugger, the `ORA_DEBUG_JDWP` config file value or environment variable, which requires no modifications to the PL/SQL program, or the `DBMS_DEBUG_JDWP` PL/SQL package, which does require modifications to the PL/SQL program.

This section includes the following topics:

How External Application Debugging Works

The Oracle PL/SQL Debugger starts when you select **Start Oracle External Application Debugger** from the Visual Studio Tools menu. A dialog box appears prompting you to enter the IP address and port number for the computer where Visual Studio is located. This must be the same IP Address and port number that you have provided in the `ORA_DEBUG_JDWP` config file of the application or environment variable or through a `DBMS_DEBUG_JDWP` package call inside PL/SQL.

When the external application opens a database connection, the database connects to the Oracle PL/SQL Debugger at the given IP address and port number and eventually the breakpoint is hit, giving control to the developer in Visual Studio.

> **Note**
>
> It is recommended that only one external application connects to the debugger at a time.

The Debug Location Toolbar displays *Oracle Debugger* once the Oracle PL/SQL Debugger starts.

Using the ORA_DEBUG_JDWP Config/Environment Variable

The `ORA_DEBUG_JDWP` variable can be set in the environment of an external application and provides a valid IP address and port number for the computer where Visual Studio is located. When any Oracle application on any operating system connects to Oracle, this variable is read and is passed to the database which then connects back to the Oracle Debugger in Visual Studio, beginning the debugging session.

Using a Environment Variable

Set the value of ORA_DEBUG_JDWP in the format:

host=*IP address or host name*;port=*debugging port number*

where:

- *IP address or host name* is the Internet Protocol dotted address or host name of the computer running Visual Studio.
- *debugging port number* is the TCP port number being used by the PL/SQL debugger.

In the following example, the ORA_DEBUG_JDWP variable is set at the command prompt:

```
set ORA_DEBUG_JDWP=host=130.35.12.111;port=1234
```

No spaces are allowed anywhere in the value of the ORA_DEBUG_JDWP variable.

When the external application creates a database connection, the database connects to the Oracle PL/SQL Debugger located in the IP address, 130.35.12.111, and at the TCP port number, 1234, where the Oracle PL/SQL Debugger listens.

> **Note**
>
> Do not set the ORA_DEBUG_JDWP variable in the Visual Studio runtime environment.

Using a ODP.NET Configuration File Variable

For ODP.NET version 12.1.0.2 or later, ORA_DEBUG_JDWP may be set in the machine.config, web.config or app.config of the application. This has the same effect as setting an environment variable. The configuration file value takes precedence over any environment variable.

Example ODP.NET config file entry:

```
<oracle.manageddataaccess.client>
  <version number="*">
    <settings>
      <setting name="ORA_DEBUG_JDWP" value="host=130.35.12.111;port=1234"/>
    </settings>
  </version>
</oracle.manageddataaccess.client>
```

Using the DBMS_DEBUG_JDWP package

The DBMS_DEBUG_JDWP PL/SQL package allows you to control precisely when the database connects or disconnects to the PL/SQL Debugger in Visual Studio. Instead of beginning the debugging session as soon as the application connects to Oracle, this package allows to you wait until a specific point in the PL/SQL code. This is useful when you only want to debug a small portion of a large package, for example. You must add code to your PL/SQL to call the DBMS_DEBUG_JDWP connect and disconnect procedures.

The DBMS_DEBUG_JDWP package provides two procedures for the application to connect and disconnect from the Oracle PL/SQL Debugger.

You must implement these two procedures in your PL/SQL code:

- Connecting Procedure

  ```
  CONNECT_TCP(HOST VARCHAR2, PORT VARCHAR2)
  ```

  This procedure allows the database to connect to the Oracle PL/SQL Debugger.

  where:

  – `HOST` is the host address where the Oracle PL/SQL Debugger resides.

  – `PORT` is the is the TCP port number for PL/SQL debugging. You must specify the same port number when you start the debugger.

  The application usually opens a database connection. The database connects to the debugger when it executes the `DBMS_DEBUG_JDWP.CONNECT_TCP` method.

- Disconnecting Procedure

  ```
  DISCONNECT()
  ```

  This procedure allows the database to disconnect from the Oracle PL/SQL Debugger.

Requirements for External Application Debugging

Prior to using External Database Debugging, perform all the setup requirements listed in PL/SQL Debugging Setup.

Before Using External Application Debugging Mode

Before using external application debugging mode, you must do the following:

- Set breakpoints.

  To use external application debugging, you must set one breakpoint in the PL/SQL program being called. If you have a second instance of Visual Studio being used to debug .NET code, you may wish to set a breakpoint at some point after the PL/SQL program is executed.

- Set the `ORA_DEBUG_JDWP` environment variable in the environment of the external application or set the `ORA_DEBUG_JDWP` ODP.NET configuration file variable.

  See Using the ORA_DEBUG_JDWP Config/Environment Variable.

  or

- Modify the PL/SQL being debugged to add calls to `DBMS_DEBUG_JDWP.CONNECT_TCP` (with the correct IP address and port number) and `DBMS_DEBUG_JDWP.DISCONNECT`.

  See Using the DBMS_DEBUG_JDWP package.

Debugging Using External Application Debugging Mode

To debug using External Application Debugging Mode, follow all the steps in the Debugging Setup Checklist.

Common PL/SQL Debugger Operations

This section describes PL/SQL debugging operations common to all three debugging modes, through the Visual Studio debugger, including PL/SQL program refresh behavior.

PL/SQL Program Refresh Behavior

When you debug a PL/SQL program, Oracle Developer Tools makes sure that the most updated version of the PL/SQL program is refreshed from the database.

If you have modified the PL/SQL program being debugged, but not saved it to the database (known as a *dirty PL/SQL program*), a Yes/No dialog box shows up to prompt you to save the PL/SQL program to the database before debugging. If you select *Yes*, the PL/SQL program is saved to the database and debugging starts. If you select *No*, debugging is not allowed.

In a scenario where a saved PL/SQL program is being debugged, and it calls a dirty PL/SQL program, the execution stops at the dirty PL/SQL program. An OK dialog box appears indicating that the called PL/SQL program has been modified and debugging might not work properly. When you click OK, the execution stops at the called (dirty) program.

Be aware that the source code might not be synchronized during debugging because the PL/SQL program has been modified.

Controlling PL/SQL Program Execution

You can control the execution of a PL/SQL program using most of the Visual Studio debugging commands available for other types of programs, however, the following debugging commands are not available:

- Set the Execution Point
- Edit and Continue

This section includes the following topics:

Starting the Oracle PL/SQL Debugger

Starting the Oracle PL/SQL Debugger is different for each of the three debugger modes. See the specific mode documentation for information on how to start the debugger:

Stepping Through a PL/SQL Program

You can step through the debugging execution using the debug step commands from the **Visual Studio Debug** menu or the Debug Toolbar.

Step Into

This command single-steps through source statements in a PL/SQL program.

You cannot step into an object method during debugging.

If a source statement is a call to other PL/SQL programs, the debugger enters the called PL/SQL program or programs. From there, how the debugger behaves depends on whether or not any called PL/SQL programs are compiled with the debug information as follows:

- If the called PL/SQL programs are compiled with debug information, the debugger stops at the entry point of each program.
- If the called PL/SQL programs are not compiled with the debug information, a message box appears with several choices. This is described in Behavior for PL/SQL Programs Compiled Without Debug Information.

Step Over

This command single-steps through source statements in a PL/SQL program.

If a source statement is a call to another PL/SQL program, the debugger stops at the next breakpoint or executes that PL/SQL program, and stops at the next source statement or the next breakpoint.

Step Out

This command is generally used from within a called PL/SQL program. It stops at the next breakpoint or completes the called program, returns execution to the calling program, and stops at the next source statement.

Continuing the Debug Execution

When the debugging process stops at a specific location, you can select **Continue** from the **Visual Studio Debug** Menu or from the Debug Toolbar to continue the PL/SQL program execution.

Stopping the Debug Processing

You can select **Stop Debugging** from the **Visual Studio Debug** Menu or from the Debug Toolbar to stop the PL/SQL program execution.

Breaking PL/SQL Program Execution

You can command the debugger to break into the PL/SQL debugging process, by selecting **Break All** from the **Visual Studio Debug** Menu or from the Debug Toolbar.

Running to the Cursor Location

When a PL/SQL program stops, you can command the debugger to stop program execution at a cursor location, as follows:

1. Open the PL/SQL program in the source window.
2. Place the cursor where you want the execution to stop.
3. Right-click and from the menu, select **Run To Cursor**.

Using Breakpoints

You can set breakpoints so that Oracle PL/SQL Debugger breaks program execution at breakpoint locations in the debug process.

> **Note**
>
> The PL/SQL debugger can only bind breakpoints to a PL/SQL program when the debugging process has started and the PL/SQL program is loaded.

PL/SQL debugging supports these breakpoint activities:

- Setting breakpoints: When a breakpoint is set, the debugger suspends execution at the specified location in the PL/SQL program.
- Editing breakpoints information.
- Setting breakpoints with the hit count property.

Oracle PL/SQL debugging does not support the following types of breakpoint:

- Function breakpoint.
- Data breakpoint.
- Address breakpoint.
- Breakpoints with conditional properties.

Breakpoint Status

Breakpoint status is indicated by symbols that appear in the gray margin, at the left, in the source window. They are:

- Enabled: Execution will be suspended at this breakpoint. Breakpoints are only enabled if they can be successfully bound to a PL/SQL program.
- Disabled: The debugger ignores this breakpoint.
- Error: No breakpoint can be set at this location; the location is not valid.
- Warning: No breakpoint can be set because the PL/SQL program has not been loaded yet. The PL/SQL Debugger will bind the breakpoint when the PL/SQL program is loaded.

How Breakpoints Work

To use or set breakpoints, you must make sure that there is a solution opened in the Visual Studio. You cannot set breakpoints if there is no solution open in Visual Studio. When Server Explorer starts or when PL/SQL debugging starts, the solution is created automatically.

Saving and Retrieving Breakpoints

Breakpoint information is saved in a solution. To save breakpoint information, you must open the solution, set breakpoints, and save the solution. When you open the solution, breakpoint information is retrieved from the solution. If you set breakpoints and close the solution without saving, the breakpoint information is gone.

Controlling Breakpoints

You can control breakpoints in the PL/SQL source window in two ways:

- You can click in the gray area to insert or delete a breakpoint.
- You can place the cursor where you want the breakpoint, then right-click and select a breakpoint command from the menu. The breakpoint commands vary depending on the context.

Inserting a Breakpoint

To insert a breakpoint, either click in the gray margin next to the area where you want to set the breakpoint, or place the cursor at the desired location in the PL/SQL source window, and select **Insert Breakpoint** from the menu.

Removing a Breakpoint

To delete a breakpoint, either click on the breakpoint symbol that you want to delete, or right-click the line in the source window that contains the breakpoint, and select **Remove Breakpoint** from the menu.

Enabling and Disabling Breakpoints

The enable and disable breakpoint commands are toggles.

To enable a disabled breakpoint, either click on the disabled breakpoint symbol that you want to enable, or right-click in the source window that contains the disabled breakpoint, and select **Enable Breakpoint** from the menu.

The process is the same to disable an enabled breakpoint.

Editing a Breakpoint

To modify a breakpoint, right-click the line that is associated with the breakpoint symbol and select **Breakpoint Properties** from the menu. When the Breakpoint dialog box appears, you can edit the basic breakpoint properties, such as file name, line and character number.

> **Note**
>
> Oracle PL/SQL Debugger can only insert a breakpoint at the beginning of the line, and ignores the character field.

Select **Hit Count** to specify the maximum number of hits that occur before execution breaks.

Examining Debug Information

When you use the Oracle PL/SQL Debugger in break mode, you can examine debugging information through Visual Studio debug windows. This section describes windows used for watching variables in different contexts, and then describes other windows that Oracle PL/SQL debugging uses, and lists non-supported debugging windows.

Watching Variables

This section describes windows used to watch information or evaluate values. Then, information common to all the windows is discussed, such as special compiling requirements and data types supported.

- Watch Window

  You can evaluate or modify variables or parameters in a PL/SQL program.

  You can add a variable to the Watch window in two ways:

  – Double-click an empty row in the Name column of the Watch window, then type the variable name in the selected row and enter.

  – In a PL/SQL program source window, locate the cursor on a PL/SQL variable, then right-click on the PL/SQL program source window and select **Add Watch** from the menu.

  To access the Watch window, select Window from the **Debug** menu, then select the Watch window.

- QuickWatch Dialog Box

  You can quickly evaluate or modify a single variable or parameter of a PL/SQL program using the QuickWatch dialog box.

  To access the QuickWatch Window, from the source window, place the cursor on a PL/SQL variable, right-click on the window, and then from the menu, select **QuickWatch**.

- Locals Window

  You can view local variables or parameters of a PL/SQL program using the Locals windows.

  To access the Locals window, select Window from the **Debug** menu, then select the Locals window.

- Autos Window

  You can view variables used in the current statement and the previous statement using the Autos window.

  To access the Autos window, select Window from the Debug menu, then select the Autos window.

Evaluating Objects, VARRAYs, and Nested Table Types

You must compile objects with debug information in order to be able to evaluate their values in the debugger.

From Server Explorer, expand the User Defined Types node, right-click on the object type node, and select **Compile Debug** or from the PL/SQL editor displaying the object type specification or body, right-click, and from the menu, select **Compile Debug**.

Evaluating Package Variables

To evaluate package variables, you must compile the package with debug information.

From Server Explorer, expand the node that contains the program being compiled, right-click on that program, and from the menu, select **Compile Debug**.

Data Types Not Evaluated

The following data types cannot be evaluated:

- `REF CURSOR`
- Any predefined, weakly typed `REF CURSOR` type such as `SYS_REFCURSOR`
- `REF` object
- `CURSOR`
- `XMLType`
- `ANYTYPE`

Oracle Variable Types

The windows discussed in this section (Watch Window, QuickWatch Dialog Box, Locals Window, and Autos Window) display the name, value, and type of the variables being watched, as follows:

- Oracle scalar types:

  The type contains the data type name.

- Composite data types:

  The value of the composite data type can be expanded as a tree view.

  Composite data type attributes:

  The name, value, and type of each attribute/member is included.

- Oracle package variables:

  The package can be expanded as a tree view, and the name, value, and type of each package variable that is included under the package node.

> **Note**
>
> The display format for the values of variables and parameters are based on the NLS settings in the database.

Data Types Supported in the Local, Watch, Quick Watch and Autos Windows

The following table shows the data types supported and their corresponding type and value representations in the Local, Watch, Quick Watch, and Autos Windows.

Table 7-1 shows the scalar types.

Table 7-1 Scalar Type

DATA TYPES	TYPE INFORMATION	VALUE PRESENTATION
BINARY_INTEGER	PLS_INTEGER	Numeric string value
BOOLEAN	BOOLEAN	true, false, or null
BINARY_FLOAT	BINARY_FLOAT	Numeric string value
BINARY_DOUBLE	BINARY_DOUBLE	Numeric string value
DEC	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
DECIMAL	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
DOUBLE PRECISION	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
FLOAT	FLOAT(Binary precision)	Numeric string value using NLS_NUMERIC_CHARACTERS
INT	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
INTEGER	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
NATURAL	PLS_INTEGER	Numeric string value
NATURALN	PLS_INTEGER	Numeric string value
NUMBER	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
NUMERIC	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
PLS_INTEGER	PLS_INTEGER	Numeric string value
POSITIVE	PLS_INTEGER	Numeric string value
POSITIVEN	PLS_INTEGER	Numeric string value
REAL(Float(63))	NUMBER(Precision, Scale)/ FLOAT(Binary precision)	Numeric string value using NLS_NUMERIC_CHARACTERS
SIGNTYPE	PLS_INTEGER	Numeric string value
SMALLINT	NUMBER(Precision, Scale)	Numeric string value using NLS_NUMERIC_CHARACTERS
CHAR	CHAR(Length)	String value

Table 7-1 (Cont.) Scalar Type

DATA TYPES	TYPE INFORMATION	VALUE PRESENTATION
CHARACTER	CHAR(Length)	String value
LONG	LONG(Max.Length)	String value
LONGRAW	LONGRAW(Max.Length)	String value
MSLABEL	MSLABEL(Max.Length)	String value
NCHAR	NCHAR(Length)	String value
NVARCHAR2	NVARCHAR2(Max.Length)	String value
RAW	RAW(Max.Length)	String value
UROWID	UROWID	String value
ROWID	ROWID	String value
STRING	VARCHAR2(Max.Length)	String value
VARCHAR	VARCHAR(Max.Length)	String value
VARCHAR2	VARCHAR2(Max.Length)	String value
DATE	DATE	String value using NLS_TIMESTAMP_FORMAT
TIMESTAMP	TIMESTAMP(Seconds Precision)	String value using NLS_TIMESTAMP_FORMAT
TIMESTAMP WITH LOCAL TIME ZONE	TIMESTAMP_WITH_LOCAL_TIME_ZONE(Seconds Precision)	String value using NLS_TIMESTAMP_FORMAT
TIMESTAMP WITH TIME ZONE	TIMESTAMP_WITH_TIME_ZONE(Seconds Precision)	String value using NLS_TIMESTAMP_TZ_FORMAT
INTERVAL DAY TO SECOND	INTERVAL_DAY_TO_SECOND(Days Precision, Seconds Precision)	String value
INTERVAL_YEAR_TO_MONTH	INTERVAL YEAR TO MONTH(Years Precision)	String value

Table 7-2 shows the composite data types.

Table 7-2 Composite Data Type

DATA TYPES	TYPE INFORMATION	VALUE PRESENTATION
OBJECT	Object Type name (for example, PERSONTYPE)	Object type can be expanded to show attribute values
RECORD	Record Type Name	Record type can be expanded to show attribute values
ASSOCIATE ARRAY	Associate Array Type Name	Array type can be expanded to show element values
VARRAY	Object Type Name	Array type can be expanded to show element values
NESTED TABLE	Object Type Name	Array type can be expanded to show element values

Table 7-3 shows the LOB types.

Table 7-3 LOB Type

DATA TYPES	TYPE INFORMATION	VALUE PRESENTATION
BFILE	BFILE	The 255 bytes of the value
BLOB	BLOB	The 255 bytes of the value
CLOB	CLOB	The 255 characters of the value
NCLOB	NCLOB	The 255 characters of the value

Evaluating and Modifying PL/SQL Variables and Parameters

In a Watch window, in addition to evaluating a PL/SQL variable, you can also modify the value of the variable. In the following examples, you can view values, and edit the values. When you press Enter the value of the variable changes to the specified value:

Scalar type:

```
x Number(4,2)
```

Composite Type:

```
create type personType as object (
  name varchar2(128),
  age NUMBER(4)
) NOT FINAL;

x persontype;
```

When you expand *x*, the debugger displays the attribute information of *x*.

Package Type:

```
    CREATE OR REPLACE PACKAGE TESTPACK AS
       PACK1 Number;
       PACK2 NUMBER;
       PROCEDURE TEST;
   END TESTPACK;
```

When you expand `TESTPACK`, you can see the value of the package variables.

Viewing the Call Stack Window

The Call Stack window displays programs that are in the stack.

You can use the context menu of the Call Stack window to enable or disable call stack properties.

Properties

The Call Stack window supports the following properties:

Property	Description
Show Parameter Names	The names of the parameters of the PL/SQL program in the current call stack.

Property	Description
Show Parameter Values	The values of the parameters of the PL/SQL program in the current call stack. Currently, the Oracle PL/SQL Debugger displays at most 30 characters for the value.
Show Parameter Types	The types of the parameters of the PL/SQL program in the current call stack.
Show Line Number	The line number in the current call stack.

Viewing the PL/SQL Program Source from the Call Stack Window

To view the PL/SQL program source code on the call stack, do one of the following:

- Double-click the PL/SQL program name in the Call Stack window.
- Right-click on the PL/SQL program name in the Call Stack window, and select **Switch** from the menu.

The source window appears and indicates the execution position, at the yellow arrow.

Viewing the Threads Window

The Visual Studio Threads window displays threads running in the PL/SQL debugging process.

Each thread represents a database connection to the Oracle PL/SQL Debugger.

In the Direct Database Debugging mode, you will only see one thread in the Threads window.

In Multitier Application Database or External Application Debugging modes, you will see one or more threads. Each thread represents a PL/SQL program that is being debugged in a database connection created from the application code.

The Threads window displays the following information:

Name	Description
ID	The current thread ID. This ID uniquely identifies a database connection to the Oracle PL/SQL Debugger.
Name	The name of the current thread.
Location	The execution location of the current thread.
Priority	The priority of the current thread.
Suspend	The suspend count of the current thread.

Viewing the Output Window for Debugger Output

The Output window displays the debugger output for the PL/SQL debugging session. To access the Output Window, from the **View** Menu, select **Other Windows,** and select **Output**.The Output window displays debugging information including:

- The TCP port number used for PL/SQL debugging.
- A message that indicates when the database is connected to the debugger.
- The message that indicates when the database is disconnected from the debugger.
- When the Oracle PL/SQL Debugger starts or ends.

Unsupported Windows

The following Visual Studio debugging windows do not apply to PL/SQL programs and therefore are not supported by Oracle PL/SQL Debugging:

- This Window
- Memory Window
- Registers Window
- Module Window
- Running Documents Window
- Immediate Window

Handling PL/SQL Exceptions

This section describe how the PL/SQL Debugger handles exceptions that are not caught.

When an uncaught PL/SQL exception occurs in the execution, a dialog box appears informing you of this and providing the following information:

Exception Name: If the PL/SQL exception is a user-defined exception, the exception name is displayed as `EXCEPTION_USER`. If the exception is a predefined PL/SQL exception, the exception name is displayed as `EXCEPTION_ORA_`*`ORACLE_ERROR_NUMBER`*.

Line number: The line where the PL/SQL exception occurred.

Program Name: The name of the PL/SQL program where the PL/SQL exception occurs. If the execution breaks at the package method, it displays the package name.

The dialog box allows you to select from the following operations:

Break: The debugger breaks at the location where PL/SQL exception occurs.

Continue: The debugger continues the program execution.

Ignore: This option does not appear in **Visual Studio**.

See Also

Oracle Database Advanced Application Developer's Guide, Debugging Stored Subprograms

Troubleshooting the Debugger

This section describes how to troubleshoot the Oracle PL/SQL Debugger:

Problems debugging PL/SQL for the very first time

There are several steps required to configure PL/SQL debugging for use. If one of the steps is skipped, one of several errors can occur. To make sure that you have performed all configuration steps, please use the Debugging Setup Checklist.

Why can't I start my debugger?

The problem can be caused by one or all of the following:

- No available port for debugging.

To solve the problem, you must specify the PL/SQL debug port number range in the PL/SQL Debug Option Page.

- The TCP port is not open for external access. This usually happens if you are connecting to a remote database.

  To solve the problem, you must specify all the ports in the debug port range that can be opened for external access. For Visual Studio running on a Windows XP computer, you need to do one of the following:

  – Add Visual Studio in the Exception list for the Windows Firewall.
  – Add all the ports from the PL/SQL debug port number range in the Exception list.
  – Turn off the Windows Firewall (Not recommended).

Why can't I debug the PL/SQL program?

The problem can be caused by one or all of the following:

- Appropriate debug privilege is not granted to the connected user.

  To solve the problem, see PL/SQL Debugging Setup.

- The PL/SQL program is not compiled with debug information.

  To solve the problem, compile the PL/SQL program with debug information. See PL/SQL Debugging Setup.

- Why can't the debugger locate the PL/SQL source during debugging.

  To solve the problem, you have to specify connection source in the PL/SQL Debug Option Page. See PL/SQL Debugging Setup.

Why are my breakpoints gone?

Breakpoint information is saved in the solution. You have to explicitly save the solution to save the breakpoint. See Using Breakpoints.

The debugger is hanging

In certain cases, for example, when Visual Studio has crashed, it may be necessary to search for and kill extraneous `OracleDebugger.exe` processes. This process can continue to run even if Visual Studio has been closed.

Close Visual Studio, press **control**-**alt**-**delete** and choose **Task List**. Locate and any kill `OracleDebugger.exe` processes that are still running. Then restart Visual Studio.

8

SQL and PL/SQL File Editor

About SQL and PL/SQL File Editor

SQL and PL/SQL File Editor lets you enter, edit, execute, and view the results of SQL and PL/SQL statements. Additionally, you can perform an Explain Plan on a SQL statement to aid in performance tuning.

The SQL and PL/SQL File Editor consists of an editor window and a separate results window.

This section covers the following topics:

Opening SQL and PL/SQL File Editor

To open the SQL and PL/SQL File Editor, right click on a connection node in Server Explorer and from the menu select **Open New SQL File** or **Open Existing SQL File**. The opened file will be associated with the Oracle Database Connection as indicated in the SQL and PL/SQL File Editor toolbar.

You can also open files containing SQL using the Visual Studio **File**->**Open**->**File** menu, or by clicking on files located in a Visual Studio solution.

> **Note**
>
> Files opened using the Visual Studio File menu or the Visual Studio solutions will *not* be associated with a Server Explorer Oracle connection and therefore you will need to use the connection dropdown in the SQL and PL/SQL File Editor toolbar to associate the file with an Oracle connection.

Only files with certain file extensions can be opened using the Visual Studio **File->Open->File** menu, or by clicking on files located in a (non Oracle Database Project) Visual Studio solution. The file extensions that can be opened using those methods are those associated with the SQL and PL/SQL File Editor:

- `.plsql`
- `.pls`
- `.bdy`
- `.fnc`
- `.pck`
- `.plb`
- `.pkb`
- `.pkh`
- `.pks`
- `.prc`
- `.spc`
- `.trg`
- `.osql`

> **Note**
>
> Files with the extension `.sql` or other files with extensions not in the list above *must* be opened using the **Open New SQL File**, **Open Existing SQL File** menus or by clicking on a file in Oracle Database Project or Oracle Database Project v2. They cannot be opened using Visual Studio **File**->**Open** menus, or by clicking on files located in a (non-Oracle Database Project) Visual Studio Solution.

Executing SQL Statements Using SQL and PL/SQL File Editor

This section tells you how to execute SQL statements using SQL and PL/SQL File Editor.

This section includes these topics:

Using IntelliSense

IntelliSense offers smart suggestions and code completion for SQL and PL/SQL. As you type SQL or PL/SQL you can take utilize IntelliSense like so:

- As you type, IntelliSense suggestions will automatically appear. You may also press Control+spacebar to obtain suggestions. You can select from the suggestions using arrow keys and the return key or by using the mouse.
- Type a schema name or object name followed by a period (".") to receive suggestions. In the case of procedure or function names, the full signature of the procedure or function name will be outputted along with placeholder values.

- After typing a procedure or function name, the left parenthesis ("(") will trigger parameter info to be displayed. The current parameter you are entering will be boldfaced.
- Hover the mouse over variables and object names to display Quick Info about the variable or object name.

IntelliSense metadata is cached. If you have recently modified database objects, click the **Rebuild Intellisense** toolbar icon or menu item to receive up to date suggestions.

To change the casing of suggestions, for example to receive suggestions in lower case, see the IntelliSense Settings Options Page.

Executing a SQL Statement

To execute a SQL or PL/SQL statement in the SQL and PL/SQL File Editor:

1. If you have multiple SQL statements and want to run one of them, place the cursor on the line containing the SQL you wish to execute.
2. Click **Execute**, either in the menu or the toolbar.

Afterwards, a results window appears, if not already visible, showing the results. The results window displays the following:

- The SQL statement that was executed.
- Query results in grid format.
- Any errors

You can customize the behavior of the results window in the Query Execution Settings.

Executing Multiple SQL Statements

SQL and PL/SQL File Editor can process an entire script file or multiple selected SQL statements and PL/SQL calls.

To execute an entire SQL script file, right click in the editor and select **Execute All** from the menu or click the Execute All toolbar icon. To execute multiple selected SQL statements, highlight the SQL or PL/SQL statements that you wish to execute and then right click in the editor and select **Execute** from the menu or click the Execute toolbar icon.

Enabling and Disabling Autocommit

To enable or disable autocommit, go to the Query Execution Settings and check or uncheck **Commit each SQL statement executed**

When autocommit is disabled, you will need to explicitly execute COMMIT or ROLLBACK statements.

Using Bind Parameters with SQL Statements or PL/SQL blocks

You can include one or more bind variable placeholders in your SQL statements or PL/SQL blocks. A parameter placeholder is identified by a colon followed by any arbitrary identifier (for example, `:ID` or `:1`). When parameterized SQL or PL/SQL is executed, the Query Parameters Dialog opens, allowing you to enter the corresponding values. For more information, see Query Parameters Dialog.

SQL and PL/SQL File Editor Menu Options

To access the following commands, right-click within the SQL and PL/SQL File Editor and select from the menu that appears. Many of these commands are also available via the toolbar.

Menu Option	Description
Execute	Executes the SQL or PL/SQL on the line containing the cursor. If you highlight multiple lines, it will execute all of them.
Execute All	Executes the entire script file.
Format	Formats the entire file using the SQL and PL/SQL Formatter Settings. If you highlight one or more SQL or PL/SQL statements it will format only those statements.
Rebuild IntelliSense Data	This menu option forces an immediate update of the IntelliSense Data. Use this if database objects have recently been modified. IntelliSense Data is cached when a connection is first established. In certain circumstances some of this data might be automatically updated. See IntelliSense Settings Options Page for more details.
Explain Plan (Grid)	Displays an explain plan in a grid format. See Explain Plan Settings for information on how to customize this plan.
Explain Plan (Text)	Displays an explain plan in a text format. See Explain Plan Settings for information on how to customize this plan.
SQL and PL/SQL Formatter Settings	Opens the SQL and PL/SQL Formatter Settings which controls how formatting will be done.
Disconnect Connection	Disconnects the editor from the database connection. The editor can be reassociated with a connection by using the connection dropdown in the editor toolbar.
Execute and Monitor SQL	Executes the statement and then opens Real-Time SQL Monitor Window to allow you to monitor the performance. See About Real-Time SQL Monitor.

See Also

PL/SQL Code Editor | *Oracle Database PL/SQL Language Reference* | *Oracle Database SQL Language Reference.*

IntelliSense Settings Options Page

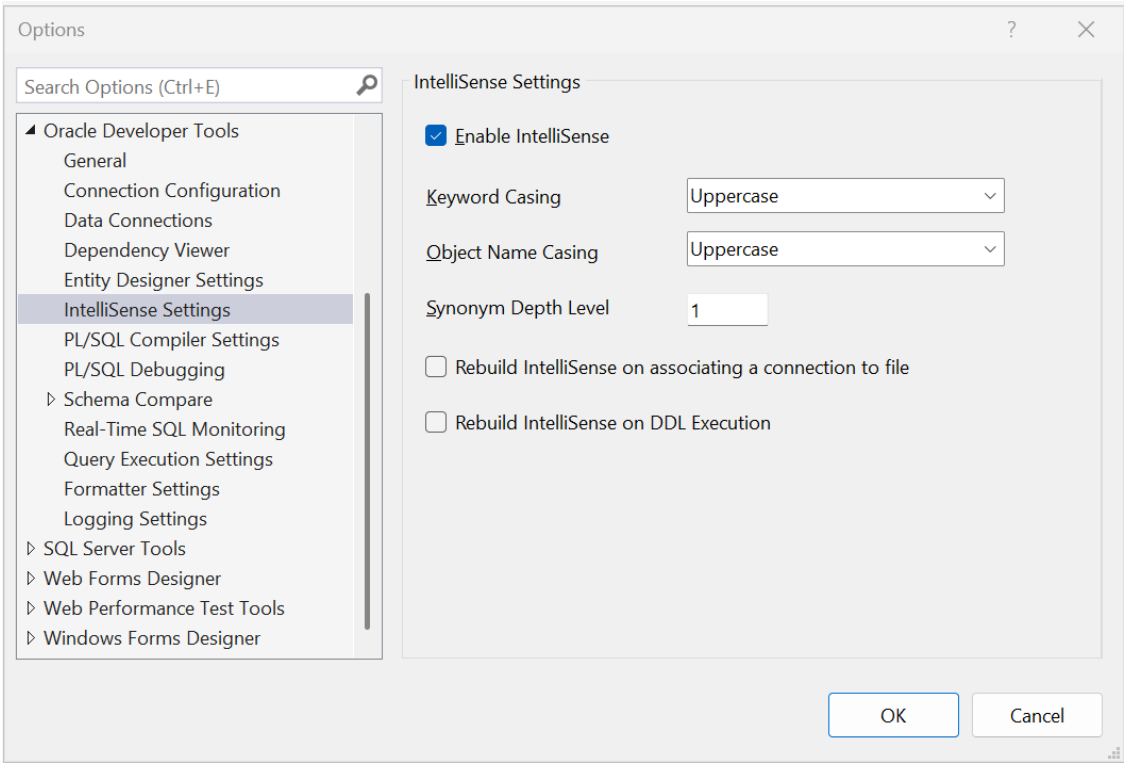
The IntelliSense Settings Options Page provides options controlling the behavior of IntelliSense in the SQL and PL/SQL File Editor and the PL/SQL Code Editor.

Accessing the IntelliSense Settings Options Page

To access the IntelliSense Settings Options Page, select **Options...** from the Tools menu. From the Options menu, select **Oracle Developer Tools**. Then select **IntelliSense Settings**.

Using the IntelliSense Settings Options Page

The IntelliSense Settings Options Page appears similar to the following:



The controls of the IntelliSense Settings Options are as follows:

Control	Description
Enable IntelliSense	Check this box to enable IntelliSense. Uncheck it to disable IntelliSense.
Keyword Casing	Select whether suggested keywords should be in uppercase, lowercase, or titlecase.
Object Casing	Select whether suggested object names should be in uppercase, lowercase, or titlecase.
Synonym depth level	Specify the maximum depth when performing synonym resolving. Synonyms can refer to other synonyms which can refer to other synonyms. Higher values of the synonym depth level can impact the performance of IntelliSense.
Rebuild IntelliSense on associating a connection to a file	Rebuilds the IntelliSense cache when a file is associated with a connection. Note that IntelliSense may be also manually rebuilt from the editor if desired.
Rebuild IntelliSense on DDL Execution	Rebuilds the IntelliSense cache when SQL is executed in the editor which creates, updates or destroys database object definitions.

Query Execution Settings

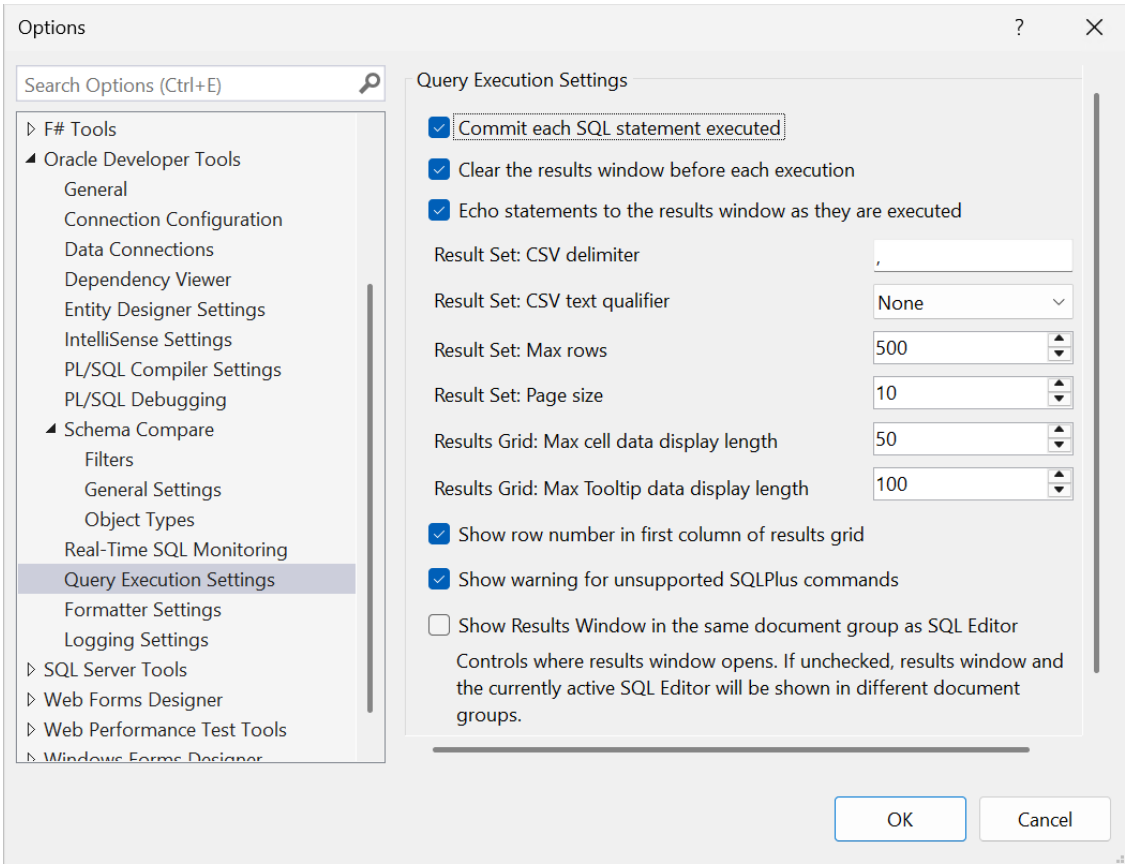
You can choose a variety of settings that control how the SQL and PL/SQL File Editor behaves including whether autocommit is enabled, whether the results window is cleared with each execution, and maximum number of rows that will be fetched.

Accessing the Query Execution Settings

To access the Query Execution Settings Page, select **Options...** from the Tools menu. From the Options menu, select **Oracle Developer Tools**. Then select **Query Execution Settings**.

Using the Query Execution Settings

The Query Execution Settings page appears similar to the following:



The controls of the Query Execution Settings page are as follows:

Control	Description
Commit each SQL statement executed	Turns autocommit on or off
Clear the results windows before each execution	Clears the results before each execution so you don't have to manually do it
Echo statements to the results window as they are executed	Copies each statement being executed to the results window
Result Set:CSV Delimiter	For results saved as a CSV file, this sets the delimiter. Enter a single character or a valid escape character. Use \\t for tab
Result Set: CSV Text qualifier	For results saved as a CSV file, this character is used to quote text strings

Control	Description
Result Set: Max Rows	These are maximum number of rows that can be fetched. A value of -1 means to fetch all rows.
Result Set: Page size	Specify the number of rows displayed on a page of results
Results Grid: Max cell data display length	Maximum number of characters or bytes that can be displayed in a cell for certain data types. Additional data will be viewable via an overflow window.
Results Grid: Max tooltip data display length	Maximum number of characters or bytes that can be displayed in a tooltip for a cell for certain data types.
Show row number in first column of results grid	Includes the row number in the results
Show warning for unsupported SQL*Plus commands	Not all SQL*Plus commands are supported. If this option is checked a dialog will be displayed to warn about unsupported commands.
Show Results Window in the same document group as the SQL Editor	Controls where the results window opens. If unchecked, results window and editor will be shown in different document groups.
OK	Updates the options and closes the dialog
Cancel	Cancels the operation and makes no changes to settings.

Explain Plan Settings

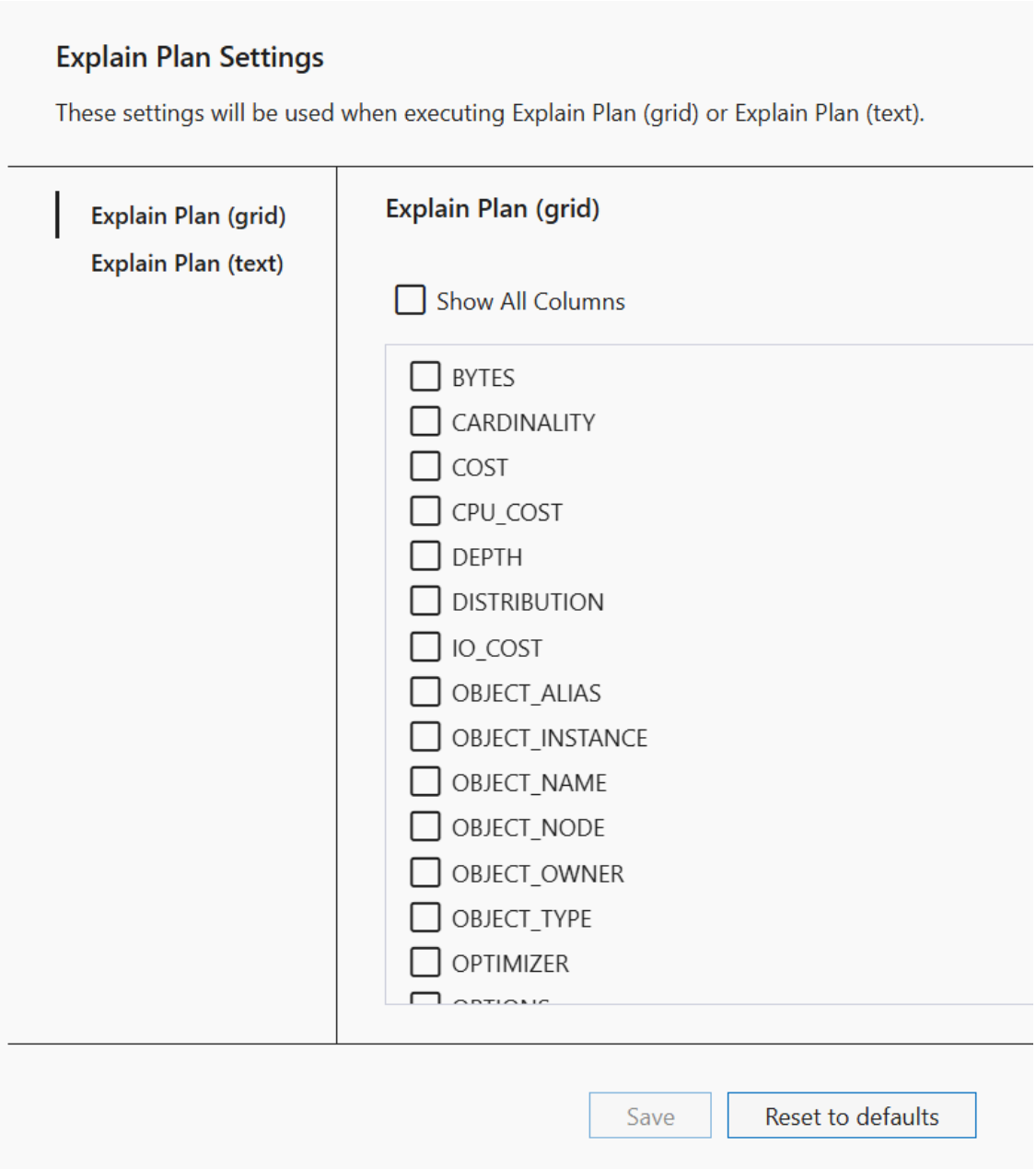
You can how execution plans are displayed by using these settings.

Accessing the Explain Plan Settings

To access the Explain Plan Settings right click on a Database Connection node in Server Explorer. Then select **Explain Plan Settings**. Select the tab for "Explain Plan (grid)" if you wish to modify grid output settings. Select the tab for "Explain Plan (text)" if you wish to modify text output settings.

Using the Explain Plan Settings

The Explain Plan Settings page appears similar to the following:



The controls of the Explain Plan Settings page are as follows:

Control	Description
Show all columns (Grid output only)	Selects all possible columns
Column checkboxes (Grid output only)	Select the Explain Plan columns you wish to display
Format Type (Text output only)	Selects the formatting that is used
Customize keywords (Text output only)	Enables you to be able to select which keywords you wish to use
Keyword checkboxes (Text output only)	Select the Explain Plan keywords you wish to display

SQL and PL/SQL Formatter Settings

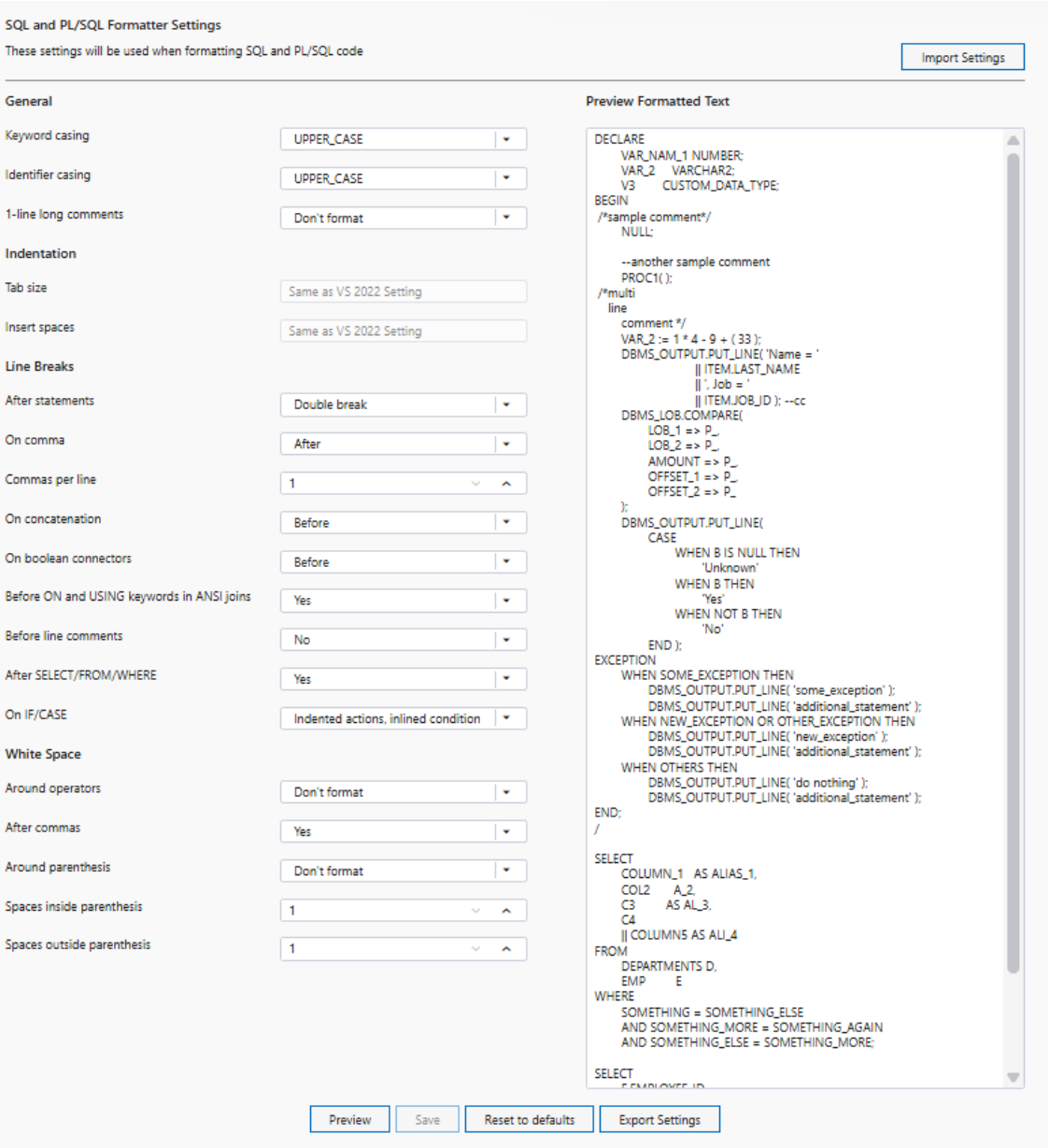
The SQL and PL/SQL File Editor and the PL/SQL Code Editor each have a Format option which formats the SQL or PL/SQL. You control how the SQL and PL/SQL formatter works by using these settings.

Accessing the SQL and PL/SQL Formatter Settings

To access the SQL and PL/SQL Formatter Settings right click on a Database Connection node in Server Explorer. Then select SQL and PL/SQL Formatter Settings. You may also access these settings by right clicking within the SQL and PL/SQL File Editor or the PL/SQL Code Editor and selecting SQL and PL/SQL Formatter Settings from the menu. There is also a SQL and PL/SQL Formatter Settings toolbar icon.

Using the SQL and PL/SQL Formatter Settings

The SQL and PL/SQL Formatter Settings page appears similar to the following:



The controls of the SQL and PL/SQL Formatter Settings page are as follows:

Control	Description
Import Settings	Import a formatter settings file that you have saved earlier using the **Export Settings** button. This will open a file browser dialog where you can locate the file.
Various formatter options	Select items from the various formatter option dropdown lists to modify the behavior of the formatter. You can preview the effect that a setting will have by clicking the **Preview** button and then viewing the **Preview Formatted Text** pane.

Control	Description
Preview Formatted Text	This pane allows you to view the impact of a change you make to a formatter option. After making a formatting change, click the **Preview** button and then view this window. You can choose to use the default SQL and PL/SQL provided in the pane or you may delete it replace it with your own example.
Preview	Updates the **Preview Formatted Text** pane to reflect the current formatter options.
Save	Saves any formatter settings changes. These changes will persist across Visual Studio launches.
Reset to defaults	Resets all the settings in the dialog to the default formatter settings.
Export Settings	Saves the settings in a JSON file which can be imported at a later time using the **Import Settings** button.

When you change various formatter settings and click the **Preview** button, the **Preview Formatted Text** pane will display the effect that the setting has on the displayed SQL or PL/SQL. You can modify the SQL or PL/SQL in the Preview pane. You may also export or import the formatter settings. Press the **Save** button to save any formatter settings changes. These changes will persist across Visual Studio launches. Close the dialog without saving if you do not wish to save your changes.

9

Using Select AI: Natural Language to SQL

About Select AI

Select AI is a feature of Oracle Autonomous Database that enables you to query your data using natural language. Select AI uses generative AI with Large Language Models (LLMs) to convert a user's input text into Oracle SQL. Select AI processes the natural language prompt, supplements the prompt with metadata from your database, and then generates and (optionally) runs a SQL query.

> **See Also**
>
> Use Select AI for Natural Language Interaction with your Database for more information about Select AI

Configuring Select AI for First Use

Perform the following steps to check the requirements and configure Select AI for first use.

1. Select AI requires the following:
   - Access to an Autonomous Database instance, such as an Always Free Autonomous Database
   - A paid API account for a supported AI provider.
2. The Select AI feature requires that the database allow network access to an AI Provider for a database user. The database user also needs execute privileges on the `DBMS_CLOUD` and `DBMS_CLOUD_AI` packages.

   Perform the following steps:

   a. In Server Explorer, Connect to the Autonomous Database as the `ADMIN` user.

   b. Right click on the connection in Server Explorer and select **Select AI**, then **Configure Select AI Provider Network Access**. This opens the Configure Select AI Provider Network Access Dialog.

   c. Choose your **AI Provider** and the **Database Username** from the drop down lists and click the **Save** button.

If you do not have access to the ADMIN account, then you can connect to your database user account and perform the preceding steps. Instead of clicking **Save**, check the **show SQL** checkbox and provide the SQL to the person with access to the ADMIN user.

3. You will need to create a credential that stores the authentication information from your AI provider for use by Oracle Database. The authentication information is obtained by logging into the AI provider's web site. You can create more than one credential.

   Perform the following steps to create a new credential:

   a. In Server Explorer, Connect to the Autonomous Database as your database user.

   b. Right click on the connection in Server Explorer and select **Select AI**, then **Configure Select AI Profile**. This opens the Configure Select AI Profile Dialog.

   c. Go to the **Create Credential** tab.

   d. Provide a credential name, credential type, and additional authentication details.

   e. Click **Save** to create the credential.

   > **Note**
   >
   > You can only create a new credential. An error will be raised if you provide an existing credential name.

4. You will need to create a profile that specifies which AI provider, credential, and model will be used and which database schema object metadata will be provided to the LLM. You may have more than one profile.

   a. In Server Explorer, right click on the connection and select **Configure Select AI Profile**. This opens the Configure Select AI Profile Dialog.

   b. Under the **Create Profile** tab, enter a profile name.

   > **Note**
   >
   > You can only create a new profile. An error will be raised if you provide an existing profile name.

   c. Choose a **Credential Name** from the drop down list.

   d. Select the **AI Provider** from the dropdown list

   e. Select the Model from the dropdown list. If you do not see the model you wish to use, then you may type the name instead.

   f. Click on the button with three dots next to the **Object List** field. The **Select AI Object List Dialog** dialog will open.

   g. In the Object List dialog, select **Object type of Tables, Views, Materialized Views** and then select the schema name from the dropdown.

   h. Using the radio button choose to show either **Tables**, **Views**, or **Materialized Views**.

   i. Choose the tables or views for which you would like to send metadata to the LLM.

   j. Cick **OK** to accept the Object List.

   k. Click **Save** to create the profile.

> **See Also**
>
> For more information about configuring Select AI, see:
>
> - Use Select AI to Generate SQL from Natural Language Prompts
> - DBMS_CLOUD_AI Package

Setting an AI Profile on a Database Connection

You must associate one of the AI Profiles that you have created with the Oracle Database connection that you wish to use with Select AI.

1. In the Server Explorer, connect to the Autonomous Database as your database user.
2. Right click on the database connection node and in the menu choose **Select AI**, then **Set Default AI Profile for Connection**.
3. In the Set Default AI Profile For Connection Dialog that opens, select the AI profile from the dropdown list.
4. Right click on the connection and select **Open New SQL File**, or **Open Existing SQL File**.
5. In the SQL file, if you did not choose a **Default AI Profile** in the connection dialog, then run this PL/SQL block (you will need to do this each time you connect):

```
begin
 dbms_cloud_ai.SET_PROFILE('yourprofilename');
end;
/
```

Translating and Executing Natural Language Queries

Once your AI Profile has been set on a connection, you can execute natural language queries by executing a statement such as:

```
SELECT AI <natural language query>
```

For example (if using the HR schema):

```
SELECT AI What are the departments and where are they located;

SELECT AI Rank the departments by how many employees work in each department;

SELECT AI Who works in the IT department;
```

You may wish to follow a workflow where you first translate the natural language to SQL, inspect it, and (optionally) edit the SQL before executing it.

1. Use the SHOWSQL option in your SELECT statement.

   For example:

   ```
   SELECT AI SHOWSQL What are the departments and where are they located;
   ```

2. Execute the statement and view the proposed SQL in the output window.

3. Paste the SQL into the SQL editor.
4. Modify if needed, then execute it.

Aside from the `SHOWSQL` option, there are many other options available such as `EXPLAINSQL` and `NARRATE`.

> **See Also**
>
> Use AI Keyword to Enter Prompts for more information

10

Oracle Database Project

About Oracle Database Project

> **Note**
>
> Oracle Database Project Version 2 is a new project type that replaces much of the functionality of Oracle Database Project. It enables you to manage a set of automatically generated standardized SQL scripts representing your entire Oracle Database schema, and to perform schema comparison, updates and deployment script generation between the database project and other database instances. For more information about these features, please see Oracle Database Project Version 2.

Oracle Database Project is a Visual Studio project type that enables you to manage Oracle SQL scripts. The Oracle Database Project can be added to source control once you have installed a source control package or plug-in.

> **See Also**
>
> For a general overview of source control support in Visual Studio, see the Microsoft documentation

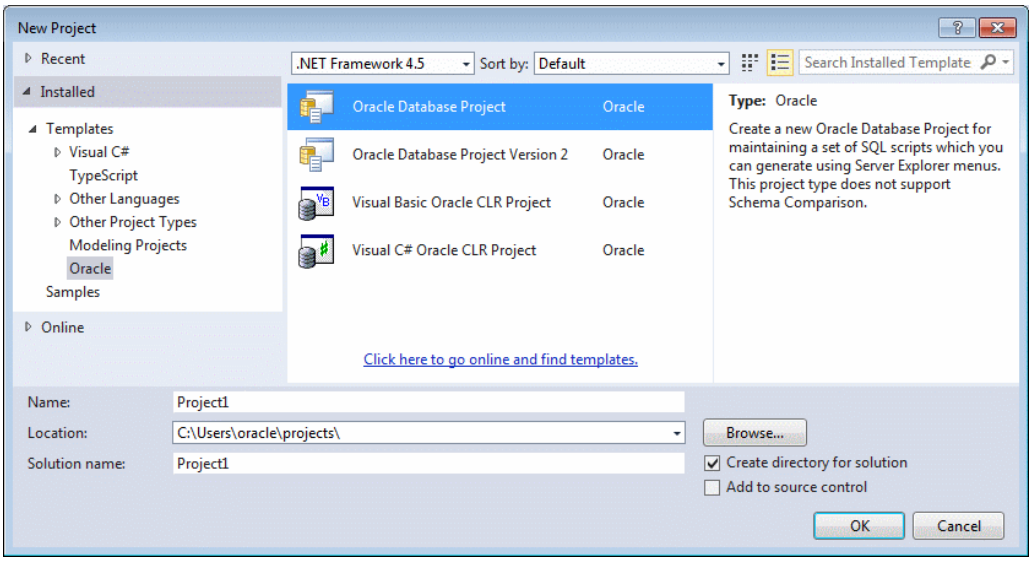
You can perform the following actions related to script files:

- Add script files to the project
- Generate scripts from existing Server Explorer objects and add them to the project
- Edit script files in the SQL editor and run the script
- Select one or more script files and run the selected scripts

Creating the Oracle Database Project

Create the Oracle Database Project as follows:

1. From the **File** menu select **New**, then **Project**.
2. Do the following when the new Project dialog appears:
   - Under Installed Templates, select **Oracle**. From the templates that appear on the right side, select **Oracle Database Project**.



Follow the same process to add a new Oracle Database Project to a currently open solution.

You can select the **Add to Source Control** check box. Visual Studio automatically adds the solution to source control after it is created. Alternatively, you can add the solution to source control in the Solution Explorer. See Viewing Oracle Database Project in Solution Explorer.

This section contains the following topics:

- Adding Database References
- Viewing Oracle Database Project in Solution Explorer

Adding Database References

After you enter a project name and location, the Add Database Reference dialog box appears. Select one Server Explorer connection to associate with the project. If there are no Server Explorer connections, then the Connection dialog box appears. You can also add a database reference after creating the project.

To create a new Server Explorer connection click **Add New Reference**. You may also click **Cancel**, specifying that no Server Explorer connections are associated with the project.

After you select **OK** or **Cancel**, the project is created and opens in Solution Explorer.

See Also

About Database References

Viewing Oracle Database Project in Solution Explorer

You can use Solution Explorer to view Oracle Database Project and to add its solution to source control.

To add a Oracle Database Project to source control, select the solution containing the Oracle Database Project and right-click on it. You will see the command **Add Solution to Source Control**. Select this and follows the commands indicated by your source control provider.

The default layout in Solution Explorer can be changed. For example, you can rename or delete folders. However, when you generate scripts, if the appropriate folder cannot be found, then the scripts go to the root of the project.

About Database Project Nodes and Folders

Oracle Database Project integration with Solution Explorer includes the following nodes and folders:

- Oracle Database Project Node
- Oracle Database Project Folders
- Project Script Files
- Database References Node
- Database Reference Nodes

Oracle Database Project Node

This section covers the following topics:

- How Oracle Database Project Node Works
- Menu Options
- Properties

How Oracle Database Project Node Works

You can view the Oracle Database Project in Visual Studio Solution Explorer.

When source control is enabled, project menus contain applicable source control operations.

To perform actions on an Oracle Database Project node, right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the nodes menu.

Menu Options

Menu Option	Description
Add New Item	Opens the Add New Item dialog box to add a new script to the project. See Adding New Scripts.
Add Existing Item	Opens the Add Existing Item dialog box to add an existing item to the project. See Adding Existing Scripts.

Menu Option	Description
Add SQL Script	Opens Add New Item dialog, which selects SQL Script project item template by default.
New Folder	Creates a new folder.
Paste	Paste scripts or folders which have been cut or copied.
Cut	Copies selected Oracle Database Project to the clipboard and prepares the node to be removed after a paste operation.
Remove	Removes the project from the solution.
Rename	Rename the project.
Save *ProjectName*	Save the project.
Properties	Displays the properties window.

Properties

Property Name	Description
Name	The name of the Oracle Database Project node.
Default Database Reference	The default database reference for the project, or blank if there is none.
File Name	Full path specification of the project file.

Oracle Database Project Folders

The default Oracle Database Project folders are named for Oracle schema types, for example, the Tables folder contains scripts to create database tables. You can rename, or delete the default folders or add your own folders. You can also drag and drop scripts from one folder to another.

When source control is enabled, folder menus also provide applicable source control operations.

To perform actions on a Oracle Database Project folder node, right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the nodes menu.

This section contains these topics:

Menu Options

Menu Option	Description
Add New Item	Opens the Add New Item dialog box to add a new script to the project. See Adding New Scripts
Add Existing Item	Opens the Add Existing Item dialog box to add an existing item to the project. See Adding Existing Scripts
Add SQL Script	Opens the Add New Item dialog box.

Menu Option	Description
New Folder	Creates a new folder.
Cut	Copies selected folder to the clipboard and prepares the folder to be removed after a paste operation.
Copy	Copies the folder.
Paste	Pastes a folder or script. This is only available if the folder or script has been copied.
Remove	Deletes the folder. If any scripts in the folder are open in an editor, then you are given the following options: • Yes: to close the editor, and delete the script. • No: to keep the editor open and not delete the script. If multiple scripts are being deleted, then the operation proceeds to the next script after you select Yes or No. • Cancel: to abort the operation. Any scripts that have already been deleted remain deleted.
Rename	Rename the folder.
Properties	Displays the properties window.

Properties

Property	Description
Name	The name of the folder.

Project Script Files

In the Oracle Database Project, script file project items represent Oracle SQL scripts.

When source control is enabled, the Menu Options also contain applicable source control operations.

Menu Options

Menu Item	Description
Open	Edits the item in the Oracle SQL editor.
Open With	Edits the item in the editor that the user selects.
Run	Runs the script using the default database reference. If there is no default database reference, then Run works the same as Run On. The results appear in the Output Pane. If there is a currently running script, then this menu item does not appear.
Run On	Prompts the user for a database reference, then runs the script. The results appear in the Output Pane. If there is a currently running script, then this menu item does not appear.
Cancel	Cancels the currently running script. If there is no script currently running, then this menu item does not appear.
Cut	Copies selected script to the clipboard and prepares the script to be removed after a paste operation.
Copy	Copies the script.

Menu Item	Description
Remove	Deletes the script file. If the script is open in an editor, then you are given the following options: • Yes: to close the editor, and delete the script. • No: to keep the editor open and not delete the script. If multiple scripts are being deleted, then the operation proceeds to the next script after you select Yes or No. • Cancel: to abort the operation. Any scripts that have already been deleted remain deleted.
Rename	Renames the script file
Properties	Displays the properties window

Properties

Property Name	Description
Name	The file name (without path).
File name	The file name including the full path.
Last Modified	The last modified time of the file.

See Also

Adding New Scripts|Adding Existing Scripts | Adding Automatically Generated Scripts for Schema Objects | Editing Script Files | Launching the Oracle SQL Script Editor | | Running a Script from Oracle Database Project

Database References Node

This section covers the following topics:

About Database References

Database references are used to run scripts.

They associate zero or more Server Explorer connections with a project. There can be, at most, one database reference which is used as the default for the project. You can delete the default database reference and have no default database reference.

A database reference is not actually a connection, but it contains the connection information that is necessary to establish the connection.

Passwords are not stored in the database reference. When a script is run, the password is taken from the Server Explorer connection that corresponds to the database reference, if one exists. If there is no existing connection, one is created. You will be prompted for the password, if necessary. After that, the password is remembered.

If the corresponding connection in Server Explorer is deleted, Oracle Database Project continues to work because the database reference provides all the information necessary to establish the connection. If an Oracle Database Project is moved to a different machine, it continues to work. Also, if the corresponding connection in Server Explorer is modified, the changes are not reflected in the database reference.

How Database References Node Work

The Oracle Database Project contains a special project item called the Database References project item. This is the parent node for the Server Explorer connections associated with the project. You cannot delete or edit this item.

To perform actions on a Database Project References node, right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the nodes menu.

Menu Options

Menu Option	Description
New Database Reference	Displays the Oracle Database Project Add Database Reference Dialog , which allows you to associate a new connection with the project. If there is no Server Explorer connection, the Connection dialog box appears instead. See Adding Database References
Properties	Displays the properties window.

Properties

Property Name	Description
Name	Database References.

Database Reference Nodes

The Database References item has Database Reference nodes as child items. Each child item represents a database reference in the project.

The default database reference has a special icon that distinguishes it from the other database references.

To perform actions on a Database Reference node, right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the nodes menu.

Menu Options

Menu Option	Description
Set as Project Default	Sets this database reference as the default for the project.
Delete	Deletes the database reference from the project (but does not delete the connection from Server Explorer). If you attempt to delete the default database reference, then you are prompted to confirm if you really want to delete.
Rename	Allows you to rename the database reference.

Menu Option	Description
Properties	Displays the properties window.

Properties

Property	Description
Name	For selected nodes, the name of the database reference.
Connection String	The connection string.

See Also

Data Connections Node| Connection Dialog Box

Managing Oracle Script Files

All scripts managed by the Oracle Database Project must have the `.sql` file extension and valid SQL or SQL*Plus syntax.

This section includes the following topics:

- Oracle SQL Script Compatibility Requirements
- Adding New Scripts
- Adding Existing Scripts
- Adding Automatically Generated Scripts for Schema Objects

Oracle SQL Script Compatibility Requirements

Please note that the script execution engine used by default with Visual Studio does not honor Oracle specific SQL syntax nor SQL*Plus directives.

To allow the execution of Oracle SQL scripts, the Oracle Developer Tools for Visual Studio includes a SQL*Plus engine.

Oracle SQL scripts are subject to the following requirements:

Auto-Commit Mode

The SQL script is executed in auto-commit mode by default, which you can change using a SQL*Plus command in the script to turn auto-commit on or off.

SQL*Plus Command Support

Oracle Developer Tools supports most SQL*Plus script commands. The following commands are not supported:

- `HOST`
- `EDIT`
- `GET`
- `SAVE`

- SPOOL
- START
- STORE

Character set Encoding of Script File

The Oracle SQL script file should be in the same encoding as specified in the Oracle NLS_LANG environment variable setting. Please refer to the *Oracle Database Globalization Support Guide*, 11*g* release 1 (11.1) for more information.

Special Characters in Script File Name

The SQL*Plus script file name cannot contain any of the following special characters:

*, ?, ", <, >, |, /, \

For file names containing these characters an "Invalid file name" error message appears in the output pane and the script will not run.

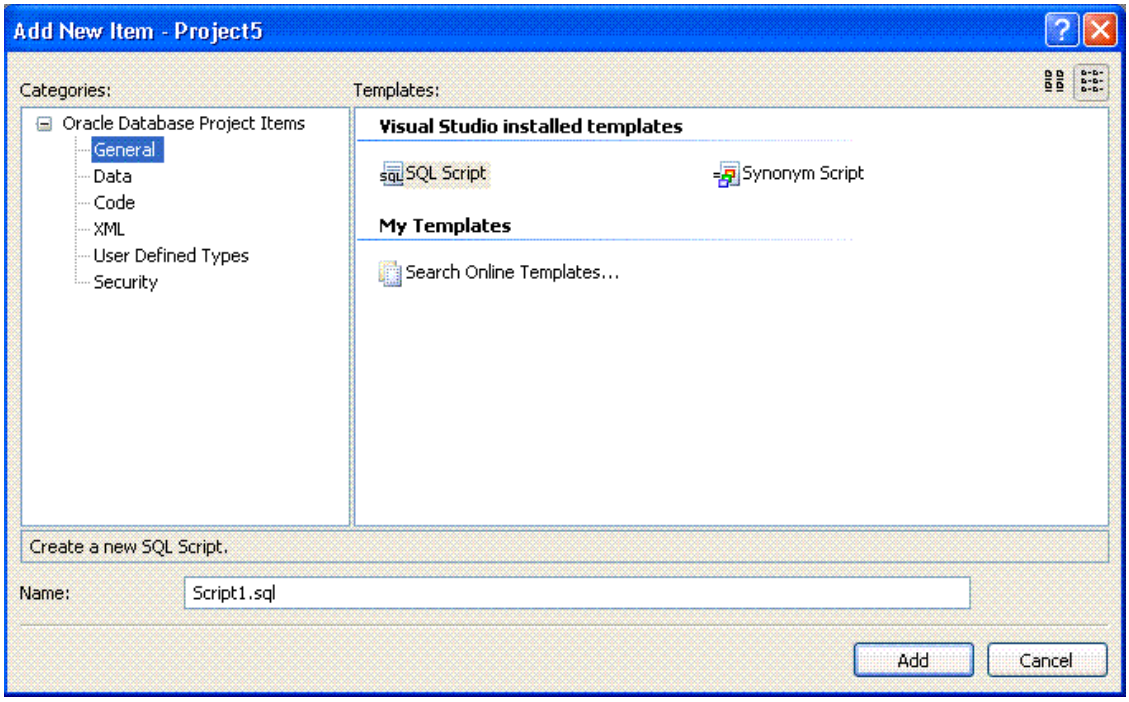
Adding New Scripts

You can add new scripts to an Oracle Database Project using Visual Studio.

To add a new script, do one of the following:

- Right-click the Oracle Database Project root node or the individual type folder node and select **Add New Item**
- Select the root node or folder node in Solution Explorer, then select the **Project Menu** and **Add New Item**.

The Add New Item dialog box appears.



The **Add New Item** dialog box lists categories on the left under Oracle Database Project Items, and script templates on the right. When you select a category, a list of the available script templates is displayed. The main categories are:

- General
- Data
- Code
- XML
- User Defined Types
- Security

Once you click **Add**, the newly created file opens in the SQL editor.

See Also

How Oracle Database Project Node Works

Adding Existing Scripts

You can add existing scripts to an Oracle Database Project using Visual Studio.

To add an existing script, do one of the following:

- Right-click the Oracle Database Project root node or folder node and select **Add Existing Item**
- Select the root node or folder node in Solution Explorer, then select the **Project Menu** and **Add Existing Item**.

The Add Existing Item dialog box appears. You can browse to the desired script, text, XML, or other file, and select it.

This command does not automatically open the added script.

See Also

How Oracle Database Project Node Works

Adding Automatically Generated Scripts for Schema Objects

Scripts can be automatically generated and added to the Oracle Database Project in several ways.

This section covers these topics:

- Using Drag-and-Drop to Generate Scripts
- Using Generate Create Script to Project Directly from Schema Objects
- Using Preview SQL Dialog Box to Generate Scripts to Projects

Using Drag-and-Drop to Generate Scripts

You can drag and drop an Oracle schema object directly from Server Explorer onto an Oracle Database Project folder. This generates an Oracle SQL script in the folder.

Multiple schema objects can be selected and dragged and dropped in one operation. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for

each schema object) generated into the target folder. The master script is ordered with dependencies taken into consideration.

Using Generate Create Script to Project Directly from Schema Objects

In Server Explorer, most database schema object nodes have a **Generate Create Script to Project** menu item. To use this menu item, an Oracle Database Project must exist within Solution Explorer.

Using the **Generate Create Script to Project** menu item, you can generate a SQL script for a database schema object to an Oracle Database Project in the folder corresponding to its schema object type. If that folder no longer exists or has been renamed, the script is written to the root of the project.

Multiple schema objects can be selected and then the **Generate Create Script to Project** menu item can be called from one of them. If multiple objects in Server Explorer are selected, a master SQL script is created in the Oracle Database Project Scripts folder. This master script contains calls to individual child SQL scripts (one for each schema object) generated into folders corresponding to the schema object types. The master script is ordered with dependencies taken into consideration.

If there are multiple Oracle Database projects open in your solution, you are prompted to select the project to add the script to.

Using Preview SQL Dialog Box to Generate Scripts to Projects

A Preview SQL dialog box can be launched from various Oracle designers and wizards. This Preview SQL Dialog contains the SQL needed to effect the changes made to the Schema object. The dialog includes a button for generating the SQL script to an Oracle Database Project for the following designers: Table, View, Procedure, Function, Trigger, Package, Synonym, and Sequence.

When you click on **Add to Project** in the Preview SQL dialog, the generated script is written to a new file in the folder for that Oracle Schema type.

If that folder no longer exists or has been renamed, the script is written to the root of the project. If there are multiple Oracle Database projects open in your solution, you are prompted to select the project to add the script to, but not the script name or location. If a file or files already exist by the specified name, a dialog box pops up asking whether or not you want to replace the existing file(s).

See Also

Preview SQL Dialog

Source Control Integration

Oracle Database Project is integrated with Visual Studio source control. All files and folders in an Oracle Database Project, including the project file, can be saved or reopened from source control. Visual Studio source control can be customized using the Visual Studio Tools Options page. For more information about Visual Studio source control, please see the Microsoft documentation.

Upgrading a Database Project from Earlier Versions

You can upgrade a Visual Studio .NET 2003 project to Visual Studio 2005, 2008, or 2010, a Visual Studio 2005 project to Visual Studio 2008 or 2010, and a Visual Studio 2008 project to Visual Studio 2010.

> **Note**
>
> Project upgrades are also needed between different versions of Oracle Developer Tools.

There are three ways to upgrade a project:

- Open the solution from the earlier version of Visual Studio to the later version of Visual Studio.
- Add the solution from the earlier version of Visual Studio to the later version of Visual Studio
- Upgrade from the command line.

About Editing Oracle SQL Scripts

This section describes the Oracle SQL Script Editor, including the following topics:

About Oracle SQL Script Editor

The Oracle SQL Script Editor is a file-based editor for SQL scripts that use Oracle syntax. Along with editing functionality, it supports keyword coloring, F1 help, and the ability to run scripts. It should be noted that there is a generic SQL script editor with Visual Studio, however, this editor does not process Oracle SQL Script files. The Oracle SQL Script editor should be used instead.

Setting the Oracle SQL Script Editor as the Default Editor

In the context of an Oracle Database project, scripts are always opened using the Oracle SQL Script Editor.

However, for external SQL files, the Microsoft SQL Editor opens `.sql` files. It does not support Oracle SQL syntax and does not use the SQL*Plus engine to execute scripts.

Therefore, for external SQL files, and also, in case a configuration setting is set incorrectly, you should set the Oracle SQL Script Editor to be the default for `.sql` files. You can do this the time you open a `.sql` file as follows:

If the **.sql** script is in the Oracle Database Project, you can set the default by selecting the script, then choosing **Open With**, selecting Oracle SQL Script Editor and clicking **Set as Default**.

For external SQL files, use the Visual Studio **File** Menu, **Open**, **File**, select the desired SQL file, then click the arrow next to **Open** button and select **Open With** then select Oracle SQL Script Editor, **Set as Default** and click **Ok**.

As it is easy to launch the wrong SQL editor, you can verify that you are using the Oracle SQL Script Editor by right-clicking and comparing your menu items to this list: Menu Options.

See Oracle SQL Script Compatibility Requirements for details on compatibility.

Launching the Oracle SQL Script Editor

The Oracle SQL Script Editor can be launched in the following ways:

- In the Oracle Database Project, double-click the script file or select it and press **Enter**
- **Open With** from the project node menu

Editing Script Files

Once script files are opened, you can edit them in the Oracle SQL Script Editor, which supports keyword coloring, and F1 help.

When you right-click the Oracle SQL Script Editor, menu options appear.

Menu Options

When source control is enabled, this menu will also contain applicable source control operations.

Menu Option	Description
Run	Runs the script using the default database reference. If there is no default database reference, then Run works the same as Run On.The results appear in the Output Pane. If there is a script running, then this menu item does not appear. You can only use Oracle SQL Script Editor to run scripts opened from an Oracle Database Project. See Running a Script from Oracle Database Project.
Run Selection	Runs the selected text using the default database reference. If there is no default database reference, then Run Selection works the same as Run On, except only the selected text runs. The results appear in the Output Pane. If there is no selected text, or if there is a currently running query, then this menu item is disabled. See Running a Script from Oracle Database Project.
Cut	Copies selected text to the clipboard, and removes it from the script.
Copy	Copies selected text to the clipboard.
Paste	Pastes text from the clipboard into the script.
Cancel	Cancels the currently running script. If there is no script running, then this menu option does not appear.
Delete	Deletes the selected text.

About Running Oracle SQL Scripts

Oracle Developer Tools for Visual Studio includes a built-in, SQL*Plus compliant execution engine which runs SQL*Plus scripts. SQL*Plus scripts can include SQL and PL/SQL, as well as SQL*Plus specific commands.

To run script files, use one of the following methods:

- Right-click on a script file project item, and select **Run** or **Run On**.
- Right-click on the Oracle SQL Script Editor, and select **Run** or **Run Selection**.
- Launch **Run SQL*Plus Script** from the Visual Studio **Tools** menu.
- Drag and drop a script file to a database reference. This is supported within a single project, but not between projects.

When you run a script, the results appear in the Output Pane.

You can cancel running scripts. To cancel long-running queries, in the script project items of the Solution Explorer, select **Cancel** from the SQL editor menu, or from the Run SQL Script Dialog Box, select **Cancel**.

See Oracle SQL Script Compatibility Requirements for details on compatibility.

This section covers the following topics:

How SQL*Plus Script Execution Works

The following apply to Run SQL*Plus Script:

- Script Execution

  The script is executed in auto-commit mode by default, however, you can disable auto-commit in the script.

  When it is required, script execution prompts for input parameters with an informational message box.

  Execution continues even if any statements or commands in the SQL*Plus script return errors.

- Results and Errors

  The results and errors from the SQL*Plus script execution appear to the Oracle Output Pane in the Visual Studio Output Window. Results and errors are written after every statement in the script is executed.

- Database Connection usage

  Run SQL*Plus Script creates a new database connection every time you select OK to execute a SQL*Plus script. This database connection is closed after the script has been completed.

- BOM Mark

Running scripts with BOM mark is not supported.

- Special Characters in Script File Name

  The SQL*Plus script file name cannot contain any of the following special characters:

  *, ?, ", <, >, |, /, \

  For file names containing these characters an "`Invalid file name`" error message appears in the output pane and the script will not run.

Running a Script from Oracle Database Project

If you right-click on a script in the Oracle Database Project, and select **Run On**, then the Run On dialog box launches.

You can choose which connection to use in the dialog box.

You can also drag and drop a script file to a database reference. This is supported within a single project, but not between projects.

See Also

Oracle Database Project Run On Dialog

Running a Script from Oracle SQL Script Editor

You can right-click on a script in the Oracle SQL Script Editor and select **Run**. The Run On dialog box launches. To cancel long-running queries select **Cancel** from the SQL editor menu.

Running a SQL Script Outside of the Oracle Database Project

To run a SQL script outside of the Oracle Database Project, you can use the Run SQL*Plus Script Dialog as follows:Select **Run SQL*Plus Script** from the Visual Studio Tools menu.This launches the Run SQL*Plus Script Dialog box.

See Also

Run SQL*Plus Script Dialog

11

Oracle Database Project Version 2

About Oracle Database Project Version 2

Oracle Database Project Version 2 is a Visual Studio project type that enables you to manage a set of automatically generated standardized SQL scripts representing one or more Oracle Database schemas, one SQL script per schema object. It also allows you to use the Schema Compare tool to perform schema comparison, updates, and deployment script generation between the database project and other database instances. The Oracle Database Project Version 2 can be added to source control once you have installed a Visual Studio source control plug-in.

After adding a Oracle Database Project Version 2 to your solution, you can select the Import Schema menu item on the Project Folder. After providing a Server Explorer Oracle Database Connection as a source, a set of SQL scripts - one for each schema object - will be automatically generated into the project. Imported SQL scripts always include the schema name for each object. An Import Schema operation may be repeated to import additional schemas into the project, but only one import per schema is allowed for the lifetime of an Oracle Database Project Version 2 project.

You may then edit the SQL scripts manually in the Oracle SQL Script editor or you may also add your own SQL scripts for new schema objects as long as they conform to the standard of only one `CREATE` statement per SQL script (any additional SQL statements beyond the single `CREATE` statement are ignored).

You can execute individual scripts from the Oracle Database Project Version 2 project or from the Oracle SQL Script Editor. SQL*Plus syntax is fully supported.

You can use the Schema Compare tool to visually compare the Oracle Database Project Version 2 project to Server Explorer Oracle Database Connections or to other Oracle Database Project Version 2 projects. You can also generate a deployment script, a *diff* script, that can be used to upgrade the target database schema to match the source schema (for Oracle Database connection targets only). You can also tell Schema Compare to update the target directly without generating a script (For either Oracle Database connection or Oracle Database Project Version 2 target types).

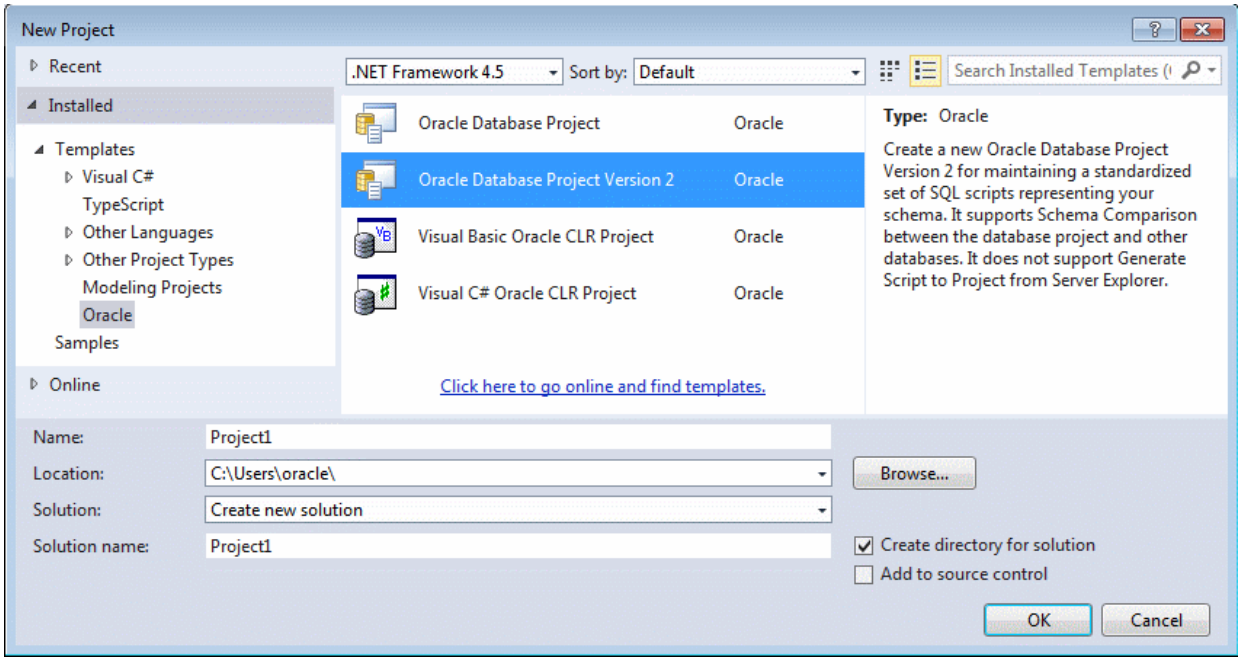
Note: The Generate Create Script to Project menu item on Server Explorer nodes is not supported with this project type. Use the Import Schema menu item on the Project folder. Or you can use the Generate Create Script menu item on Server Explorer nodes to generate a

SQL statement into a file, and then manually add the file into the Oracle Database Project Version 2. A third option is to use the Schema Compare tool to update the project.

Creating the Oracle Database Project Version 2

Create the Oracle Database Project Version 2 as follows:

1. From the **File** menu select **New**, then **Project**.
2. Do the following when the new Project dialog appears:
   - Under Installed Templates, select **Oracle**. From the templates that appear on the right side, select **Oracle Database Project Version 2**.



Follow the same process to add a new Oracle Database Project Version 2 to a currently open solution.

You can select the **Add to Source Control** check box. Visual Studio automatically adds the solution to source control after it is created. Alternatively, you can add the solution to source control in the Solution Explorer. See Viewing Oracle Database Project Version 2 in Solution Explorer.

Viewing Oracle Database Project Version 2 in Solution Explorer

You can use Solution Explorer to view Oracle Database Project Version 2 and to add its solution to source control.

To add a Oracle Database Project Version 2 to source control, select the solution containing the Oracle Database Project Version 2 and right-click on it. You will see the command **Add Solution to Source Control**. Select this and follows the commands indicated by your source control provider.

About Database Project Folders and Project Items

Oracle Database Project Version 2 integration with Solution Explorer includes the following nodes and folders:

- Oracle Database Project Version 2 Project Folder
- Oracle Database Project Version 2 Folders
- Project Script Files
- Database References Folder
- Database Reference Project Items

Oracle Database Project Version 2 Project Folder

This section covers the following topics:

- How Oracle Database Project Version 2 Project Folder Works
- Menu Options
- Properties

How Oracle Database Project Version 2 Project Folder Works

You can view the Oracle Database Project Version 2 in Visual Studio Solution Explorer.

When source control is enabled, project menus contain applicable source control operations.

To perform actions on an Oracle Database Project Version 2 project node, right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the nodes menu.

Menu Options

Menu Option	Description
Add New Item	Opens the Add New Item dialog box to add a new script to the project. See Adding New Scripts.
Add Existing Item	Opens the Add Existing Item dialog box to add an existing item to the project. See Adding Existing Scripts.
Add SQL Script	Opens Add New Item dialog, which selects SQL Script project item template by default.
New Folder	Creates a new folder.
Import Schema	Opens the Import Schema Dialog. You can provide a source Server Explorer Oracle Database Connection or create a new one. One SQL script per schema object will be generated into the Oracle Database Project Version 2. Imported SQL scripts will always include a schema name. An Import Schema operation may be repeated to import additional schemas into the project, but only one import per schema is allowed for the lifetime of an Oracle Database Project Version 2 project. Additional scripts may be added to the project via the Schema Compare tool, or using the Add Existing Item menu item. See the Visual Studio output window for error messages and warnings related to the import.

Menu Option	Description
Oracle Schema Compare	Opens the Schema Compare tool.
Paste	Paste scripts or folders which have been cut or copied.
Cut	Copies selected Oracle Database Project Version 2 to the clipboard and prepares the node to be removed after a paste operation.
Remove	Removes the project from the solution.
Rename	Rename the project.
Save *ProjectName*	Save the project.
Properties	Displays the properties window.

Properties

Property Name	Description
Name	The name of the Oracle Database Project Version 2 project Node.
Build Output File Name	When the Build menu item is chosen, the Build deployment script will use this name.
Build Output Path	The path or partial path relative to the project where the Build deployment script will be stored.
Database Project Version	Version of the database project (currently Version 2).
Default Database Reference	The default database reference for the project, or blank if there is none.
File Name	Full path specification of the project file.
Imported Schemas	List of the schema names that have been imported into this project.
Target Platform	The database version (or later) that the scripts in this project are compatible with. This value is set when the schema is imported and may be changed by the user. Errors will result if scripts contain SQL not compatible with this database version.

Oracle Database Project Version 2 Folders

The default Oracle Database Project Version 2 folders are named for Oracle schema types, for example, the Tables folder contains scripts to create database tables.

When source control is enabled, folder menus also provide applicable source control operations.

To perform actions on a Oracle Database Project Version 2 folder node, right-click the node and from the menu choose the appropriate command. To view the node's properties, select the node and/or click **Properties** in the nodes menu.

This section contains these topics:

- Menu Options
- Properties

Menu Options

Menu Option	Description
Add New Item	Opens the Add New Item dialog box to add a new script to the project. See Adding New Scripts
Add Existing Item	Opens the Add Existing Item dialog box to add an existing item to the project. See Adding Existing Scripts
Add SQL Script	Opens the Add New Item dialog box.
New Folder	Creates a new folder.
Cut	Copies selected folder to the clipboard and prepares the folder to be removed after a paste operation.
Copy	Copies the folder.
Paste	Pastes a folder or script. This is only available if the folder or script has been copied.
Remove	Deletes the folder. If any scripts in the folder are open in an editor, then you are given the following options: • Yes: to close the editor, and delete the script. • No: to keep the editor open and not delete the script. If multiple scripts are being deleted, then the operation proceeds to the next script after you select Yes or No. • Cancel: to abort the operation. Any scripts that have already been deleted remain deleted.
Rename	Rename the folder.
Properties	Displays the properties window.

Properties

Property	Description
Name	The name of the folder.

Project Script Files

In an Oracle Database Project Version 2, script file project items represent Oracle SQL scripts. These script files follow the convention of one CREATE statement per SQL file per schema object. Any additional SQL statements beyond a single CREATE statement will be ignored. See the errors and warnings window of Visual Studio and the Visual Studio output window for notifications of this.

When source control is enabled, the Menu Options also contain applicable source control operations.

Menu Options

Menu Item	Description
Open	Edits the item in the Oracle SQL editor.
Open With	Edits the item in the editor that the user selects.

Menu Item	Description
Run	Runs the script using the default database reference. If there is no default database reference, then Run works the same as Run On. The results appear in the Output Pane. If there is a currently running script, then this menu item does not appear.
Run On	Prompts the user for a database reference, then runs the script. The results appear in the Output Pane. If there is a currently running script, then this menu item does not appear.
Cancel	Cancels the currently running script. If there is no script currently running, then this menu item does not appear.
Dependencies and References	Opens the Dependencies and References Viewer for viewing the dependencies that this object has on other database schema objects.
Cut	Copies selected script to the clipboard and prepares the script to be removed after a paste operation.
Copy	Copies the script.
Remove	Deletes the script file. If the script is open in an editor, then you are given the following options: • Yes: to close the editor, and delete the script. • No: to keep the editor open and not delete the script. If multiple scripts are being deleted, then the operation proceeds to the next script after you select Yes or No. • Cancel: to abort the operation. Any scripts that have already been deleted remain deleted.
Rename	Renames the script file
Properties	Displays the properties window

Properties

Property Name	Description
Name	The file name (without path).
File name	The file name including the full path.
Last Modified	The last modified time of the file.

See Also

Adding New Scripts|Adding Existing Scripts | Adding Automatically Generated Scripts for Schema Objects | Editing Script Files | Launching the Oracle SQL Script Editor | | Running a Script from Oracle Database Project Version 2

Database References Folder

This section covers the following topics:

- About Database References
- How Database References Folder Works
- Menu Options
- Properties

About Database References

Database references are used to run scripts.

They associate zero or more Server Explorer connections with a project. There can be, at most, one database reference which is used as the default for the project. You can delete the default database reference and have no default database reference.

A database reference is not actually a connection, but it contains the connection information that is necessary to establish the connection.

Passwords are not stored in the database reference. When a script is run, the password is taken from the Server Explorer connection that corresponds to the database reference, if one exists. If there is no existing connection, one is created. You will be prompted for the password, if necessary. After that, the password is remembered.

If the corresponding connection in Server Explorer is deleted, Oracle Database Project Version 2 continues to work because the database reference provides all the information necessary to establish the connection. If an Oracle Database Project Version 2 is moved to a different machine, it continues to work. Also, if the corresponding connection in Server Explorer is modified, the changes are not reflected in the database reference.

How Database References Folder Works

The Oracle Database Project Version 2 contains a special project item called the Database References Folder. This is the parent folder for the Server Explorer connections associated with the project. You cannot delete or edit this folder.

To perform actions on a Database Project References Folder, right-click the folder and from the menu choose the appropriate command. To view the folder's properties, select the folder and/or click **Properties** in the folder's menu.

Menu Options

Menu Option	Description
New Database Reference	Displays the Oracle Database Project Add Database Reference Dialog , which allows you to associate a new connection with the project. If there is no Server Explorer connection, the Connection dialog box appears instead.
Properties	Displays the properties window.

Properties

Property Name	Description
Name	Database References.

Database Reference Project Items

The Database References Folder has Database Reference Project Items as child items. Each child item represents a database reference in the project.

The default database reference has a special icon that distinguishes it from the other database references.

To perform actions on a Database Reference Project Item, right-click the item and from the menu choose the appropriate command. To view the item's properties, select the item and/or click **Properties** in the item's menu.

Menu Options

Menu Option	Description
Set as Project Default	Sets this database reference as the default for the project.
Delete	Deletes the database reference from the project (but does not delete the connection from Server Explorer). If you attempt to delete the default database reference, then you are prompted to confirm if you really want to delete.
Rename	Allows you to rename the database reference.
Properties	Displays the properties window.

Properties

Property	Description
Name	The name of the database reference.
Connection String	The connection string.

See Also

Data Connections Node| Connection Dialog Box

Managing Oracle Script Files

All scripts managed by the Oracle Database Project Version 2 must have the `.sql` file extension and valid SQL or SQL*Plus syntax.

This section includes the following topics:

- Oracle SQL Script Compatibility Requirements
- Adding New Scripts
- Adding Existing Scripts
- Adding Automatically Generated Scripts for Schema Objects

Oracle SQL Script Compatibility Requirements

Please note that the script execution engine used by default with Visual Studio does not honor Oracle specific SQL syntax nor SQL*Plus directives.

To allow the execution of Oracle SQL scripts, the Oracle Developer Tools for Visual Studio includes a SQL*Plus engine.

Oracle SQL scripts are subject to the following requirements:

Auto-Commit Mode

The SQL script is executed in auto-commit mode by default, which you can change using a SQL*Plus command in the script to turn auto-commit on or off.

SQL*Plus Command Support

Oracle Developer Tools supports most SQL*Plus script commands. The following commands are not supported:

- `HOST`
- `EDIT`
- `GET`
- `SAVE`
- `SPOOL`
- `START`
- `STORE`

Character set Encoding of Script File

The Oracle SQL script file should be in the same encoding as specified in the Oracle `NLS_LANG` environment variable setting. Please refer to the *Oracle Database Globalization Support Guide*, 11*g* release 1 (11.1) for more information.

Special Characters in Script File Name

The SQL*Plus script file name cannot contain any of the following special characters:

*, ?, ", <, >, |, /, \

For file names containing these characters an "`Invalid file name`" error message appears in the output pane and the script will not run.

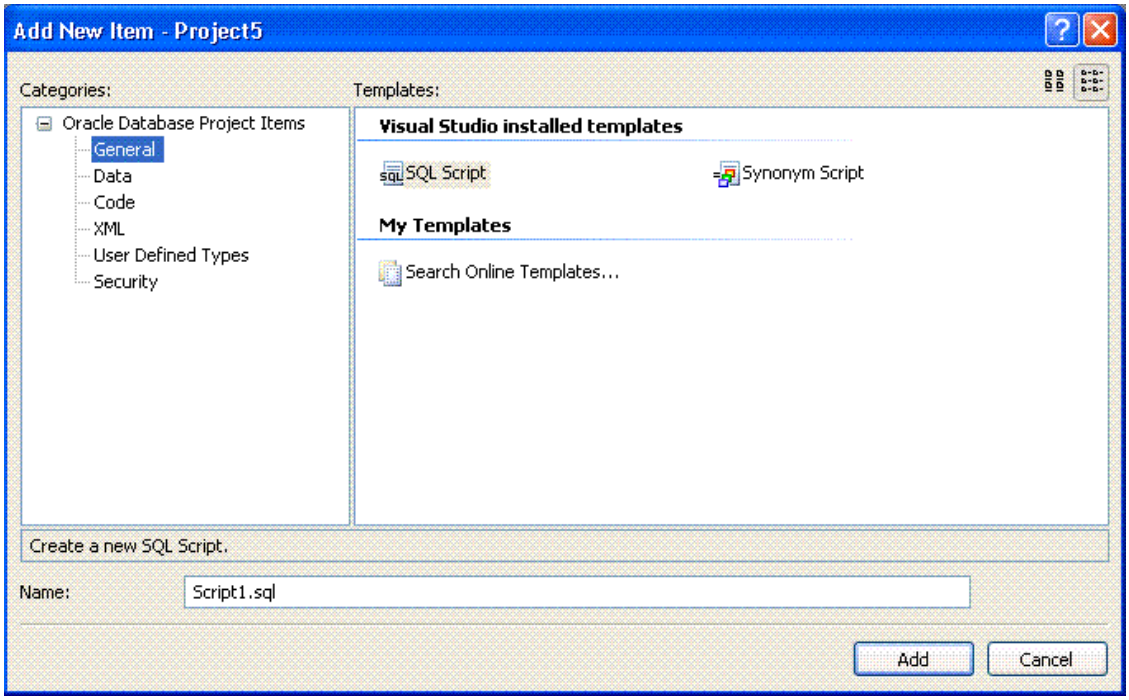
Adding New Scripts

You can add new scripts to an Oracle Database Project Version 2 using Visual Studio.

To add a new script, do one of the following:

- Right-click the Oracle Database Project Version 2 root node or the individual type folder node and select **Add New Item**
- Select the root node or folder node in Solution Explorer, then select the **Project Menu** and **Add New Item**.

The Add New Item dialog box appears.



The **Add New Item** dialog box lists categories on the left under Oracle Database Project Version 2 Items, and script templates on the right. When you select a category, a list of the available script templates is displayed. The main categories are:

- General
- Data
- Code
- XML
- User Defined Types
- Security

Once you click **Add**, the newly created file opens in the SQL editor.

See Also

How Oracle Database Project Version 2 Project Folder Works

Adding Existing Scripts

You can add existing scripts to an Oracle Database Project Version 2 using Visual Studio.

To add an existing script, do one of the following:

- Right-click the Oracle Database Project Version 2 root node or folder node and select **Add Existing Item**
- Select the root node or folder node in Solution Explorer, then select the **Project Menu** and **Add Existing Item**.

The Add Existing Item dialog box appears. You can browse to the desired script, text, XML, or other file, and select it.

This command does not automatically open the added script.

See Also

How Oracle Database Project Version 2 Project Folder Works

Adding Automatically Generated Scripts for Schema Objects

Scripts can be automatically generated and added to the Oracle Database Project Version 2 by selecting the Import Schema menu item on the Project folder. This opens the Import Schema Dialog. You can provide a source Server Explorer Oracle Database Connection or create a new one. One SQL script per schema object will be generated into the Oracle Database Project Version 2. You can also use the Schema Compare tool to add new scripts or update existing scripts in the project

Source Control Integration

Oracle Database Project Version 2 is integrated with Visual Studio source control. All files and folders in an Oracle Database Project Version 2, including the project file, can be saved or reopened from source control. Visual Studio source control can be customized using the Visual Studio Tools Options page. For more information about Visual Studio source control, please see the Microsoft documentation.

About Editing Oracle SQL Scripts

This section describes the Oracle SQL Script Editor, including the following topics:

- About Oracle SQL Script Editor
- Setting the Oracle SQL Script Editor as the Default Editor
- Launching the Oracle SQL Script Editor
- Editing Script Files

About Oracle SQL Script Editor

The Oracle SQL Script Editor is a file-based editor for SQL scripts that use Oracle syntax. Along with editing functionality, it supports keyword coloring, F1 help, and the ability to run scripts. It should be noted that there is a generic SQL script editor with Visual Studio, however, this editor does not process Oracle SQL Script files. The Oracle SQL Script editor should be used instead.

Setting the Oracle SQL Script Editor as the Default Editor

In the context of an Oracle Database Project Version 2, scripts are always opened using the Oracle SQL Script Editor.

However, for external SQL files, the Microsoft SQL Editor opens `.sql` files. It does not support Oracle SQL syntax and does not use the SQL*Plus engine to execute scripts.

Therefore, for external SQL files, and also, in case a configuration setting is set incorrectly, you should set the Oracle SQL Script Editor to be the default for `.sql` files. You can do this the time you open a `.sql` file as follows:

If the **.sql** script is in the Oracle Database Project Version 2, you can set the default by selecting the script, then choosing **Open With**, selecting Oracle SQL Script Editor and clicking **Set as Default**.

For external SQL files, use the Visual Studio **File** Menu, **Open**, **File**, select the desired SQL file, then click the arrow next to **Open** button and select **Open With** then select Oracle SQL Script Editor, **Set as Default** and click **Ok**.

As it is easy to launch the wrong SQL editor, you can verify that you are using the Oracle SQL Script Editor by right-clicking and comparing your menu items to this list: Menu Options.

See Oracle SQL Script Compatibility Requirements for details on compatibility.

Launching the Oracle SQL Script Editor

The Oracle SQL Script Editor can be launched in the following ways:

- In the Oracle Database Project Version 2, double-click the script file or select it and press **Enter**
- **Open With** from the project node menu

Editing Script Files

Once script files are opened, you can edit them in the Oracle SQL Script Editor, which supports keyword coloring, and F1 help. Oracle Database Project Version 2 script files follow the convention of one CREATE statement per SQL file per schema object. Any additional SQL statements beyond a single CREATE statement will be ignored. See the errors and warnings window of Visual Studio and the Visual Studio output window for notifications of this.

When you right-click the Oracle SQL Script Editor, menu options appear.

Menu Options

When source control is enabled, this menu will also contain applicable source control operations.

Menu Option	Description
Run	Runs the script using the default database reference. If there is no default database reference, then Run works the same as Run On.The results appear in the Output Pane. If there is a script running, then this menu item does not appear. You can only use Oracle SQL Script Editor to run scripts opened from an Oracle Database Project Version 2. See Running a Script from Oracle Database Project Version 2.
Run Selection	Runs the selected text using the default database reference. If there is no default database reference, then Run Selection works the same as Run On, except only the selected text runs. The results appear in the Output Pane. If there is no selected text, or if there is a currently running query, then this menu item is disabled. See Running a Script from Oracle Database Project Version 2.
Cut	Copies selected text to the clipboard, and removes it from the script.
Copy	Copies selected text to the clipboard.
Paste	Pastes text from the clipboard into the script.
Cancel	Cancels the currently running script. If there is no script running, then this menu option does not appear.
Delete	Deletes the selected text.

About Running Oracle SQL Scripts

Oracle Developer Tools for Visual Studio includes a built-in, SQL*Plus compliant execution engine which runs SQL*Plus scripts. SQL*Plus scripts can include SQL and PL/SQL, as well as SQL*Plus specific commands.

To run script files, use one of the following methods:

- Right-click on a script file project item, and select **Run** or **Run On**.
- Right-click on the Oracle SQL Script Editor, and select **Run** or **Run Selection**.
- Launch **Run SQL*Plus Script** from the Visual Studio **Tools** menu.
- Drag and drop a script file to a database reference. This is supported within a single project, but not between projects.

When you run a script, the results appear in the Output Pane.

You can cancel running scripts. To cancel long-running queries, in the script project items of the Solution Explorer, select **Cancel** from the SQL editor menu, or from the Run SQL Script Dialog Box, select **Cancel**.

See Oracle SQL Script Compatibility Requirements for details on compatibility.

This section covers the following topics:

How SQL*Plus Script Execution Works

The following apply to Run SQL*Plus Script:

- Script Execution

  The script is executed in auto-commit mode by default, however, you can disable auto-commit in the script.

  When it is required, script execution prompts for input parameters with an informational message box.

  Execution continues even if any statements or commands in the SQL*Plus script return errors.

- Results and Errors

  The results and errors from the SQL*Plus script execution appear to the Oracle Output Pane in the Visual Studio Output Window. Results and errors are written after every statement in the script is executed.

- Database Connection usage

  Run SQL*Plus Script creates a new database connection every time you select OK to execute a SQL*Plus script. This database connection is closed after the script has been completed.

- BOM Mark

Running scripts with BOM mark is not supported.

- Special Characters in Script File Name

  The SQL*Plus script file name cannot contain any of the following special characters:

  *, ?, ", <, >, |, /, \

  For file names containing these characters an "`Invalid file name`" error message appears in the output pane and the script will not run.

Running a Script from Oracle Database Project Version 2

If you right-click on a script in the Oracle Database Project Version 2, and select **Run On**, then the Run On dialog box launches.

You can choose which connection to use in the dialog box.

You can also drag and drop a script file to a database reference. This is supported within a single project, but not between projects.

See Also

Oracle Database Project Run On Dialog

Running a Script from Oracle SQL Script Editor

You can right-click on a script in the Oracle SQL Script Editor and select **Run**. The Run On dialog box launches. To cancel long-running queries select **Cancel** from the SQL editor menu.

Running a SQL Script Outside of the Oracle Database Project Version 2

To run a SQL script outside of the Oracle Database Project Version 2, you can use the Run SQL*Plus Script Dialog as follows:Select **Run SQL*Plus Script** from the Visual Studio Tools menu.This launches the Run SQL*Plus Script Dialog box.

See Also

Run SQL*Plus Script Dialog

12

Oracle Streams Advanced Queueing

About Oracle Streams Advanced Queueing

Oracle Streams Advanced Queueing (known as AQ) provides message management and communication for application integration.

The Oracle Data Provider for .NET provides programmatic interfaces for writing applications that use Oracle Streams and Advanced Queueing.

Oracle Developer Tools includes visual designers for managing the administrative features of Advanced Queuing, for example, creating queues and queue tables. This is much easier than using SQL to perform the same tasks.

See Also

Oracle Streams Concepts and Administration | *Oracle Streams Advanced Queuing User's Guide* |Advanced Queuing Support in *Oracle Data Provider for .NET Developer's Guide*

Setting Up

In order to administer AQ streams, you must be granted appropriate privileges to enqueue, dequeue, and manage queues.

The following Oracle user privileges are required to use this functionality:

- To Create, Drop or Monitor your own Queues

  You must be granted `EXECUTE` rights on `DBMS_AQADM`.

- To Create, Drop or Monitor any Queues

  You must be granted `EXECUTE` rights on `DBMS_AQADM` and be granted `AQ_ADMINISTRATOR_ROLE` by another user who has been granted this role (`SYS` and `SYSTEM` are the granters of `AQ_ADMINISTRATOR_ROLE`).

- To `ENQUEUE` or `DEQUEUE` your own queues

  You must be granted `EXECUTE` rights on `DBMS_AQ`.

You can use grant Grant/Revoke Privileges Dialog to achieve this.

See Also:

Grant/Revoke Privileges Dialog | Chapter 9 in *Oracle® Streams Concepts and Administration*

AQ Server Explorer Nodes

Using Oracle Developer Tools and Server Explorer, you can view and perform actions on AQ nodes.

The following Server Explorer nodes are specific to Advanced Queueing:

- Advanced Queues Node
- Queue Tables Node
- Queue Table Nodes
- Queues Node
- Queue Nodes
- Subscribers Node
- Subscriber Nodes
- Propagations Node
- Propagation Nodes

AQ Designers

Oracle Developer Tools provides the following designers to use with Oracle Streams AQ.

- Queue Designer
- Queue Table Designer

13

Users, Roles, and Privileges

About Users, Roles, and Privileges

The Users and Roles Management feature in Oracle Developer Tools allows you to manage users, roles, and privileges in the database. You can view users and roles through the Server Explorer. There are designers to create and modify them.

About Users and Roles

Before users can connect to an Oracle database, you need to create user accounts and roles for them.

A role is a set of privileges that you can grant to users or to other roles. The default Oracle Database installation include several sample users and roles, such as the `HR` user and the `DBA` role.

You can create user accounts and roles from the Users and Roles Management feature in Oracle Developer Tools as well as in several Oracle Database tools, such as SQL*Plus or the Oracle Enterprise Manager Database Control, which is installed with Oracle Database.

For detailed information on how user accounts and roles work, or how to create and manage user accounts from SQL*Plus, see *Oracle Database Administrator's Guide*. For more information on Enterprise Manager Database Control, see *Oracle Database 2 Day DBA*.

About Privileges

A privilege is the right to run a particular type of SQL statement or to access another user's object. Examples of privileges on a table are `ALTER`, `INSERT`, and `DELETE`.

Viewing Users and Roles in Server Explorer

Server Explorer displays users and roles in two different views, in the object view and in the schema view. You can switch between these two views by selecting the connection node and choosing **Change View**, then **Objects** or **Schemas** from the menu. In the object view, all schema objects of the same type are grouped under the same collection node (such as Tables), and the name of the schema object is prefixed with the schema name in parenthesis. In the schema view, the schema objects are grouped under a Schema node.

The labels Users and Schemas are used interchangeably depending on the view. When you are in schema view, the Schemas label is used and when you are in object view, the Users label is used.

There is also a Roles node in Server Explorer. When you are logged in as SYSDBA, in Server Explorer, in schema view, you view all roles in the database. If you are a non-SYSDBA user, or a SYSDBA user viewing Server Explorer in object view, you view the roles that are assigned to you.

See Also

Object View and Schema View

Managing Users

You can create, modify, and delete users.

This section covers the following topics:

- Creating a User
- Modifying a User
- Deleting a User

See Also

User Designer | Users Node | User Nodes

Creating a User

To create a new user, select the Users node or Schemas node in Server Explorer, then select **New User**. The User Designer appears.

See Also

User Designer

Modifying a User

To modify a user, select the Users node or Schemas node in Server Explorer. Select the user to modify, right-click, and select **Edit**. The User Designer appears.

See Also

User Designer

Deleting a User

To delete a user, select the Users node or Schemas node in Server Explorer. Select the user to delete, right-click and select **Delete**. When the confirmation message appears, select **Yes**.

> **Note**
>
> Only a user that has been granted the DROP USER system privilege can drop other users. A connected user cannot be deleted.

See Also

User Designer

Managing Roles

You can create, modify, and delete roles.

This section covers the following sections:

- Creating a Role
- Modifying a Role
- Deleting a Role

See Also

Role Designer | Roles Node | Role Nodes

Creating a Role

To create a new role, select the Roles node in Server Explorer, then select **New Role**. The Role Designer appears.

See Also

Role Designer

Modifying a Role

To modify a role, select the Roles node in Server Explorer. Select the role to modify, right-click, and select **Edit**. The Role Designer appears.

> **Note**
>
> The role name cannot be modified.

See Also

Role Designer

Deleting a Role

To delete a role, select the Roles node in Server Explorer. Select the role to delete, right-click, and select **Delete**. When the confirmation message appears, select **Yes**.

> **Note**
>
> Only a user that has been granted the `DROP ANY ROLE` system privilege or has been granted the role with `ADMIN OPTION` can drop other roles.

See Also

Role Designer

Granting Privileges to Roles and Users

Typically, privileges are granted to roles which are in turn granted to users. Privileges can also be granted directly to a user.

- Example of an explicit grant to a User: You can explicitly grant the privilege to insert records into the `EMPLOYEES` table to the user `SCOTT`.
- Example of a grant to a Role: You can grant the privilege to insert records into the `EMPLOYEES` table to the role named `CLERK`, and, in turn you can grant that role to users `SCOTT` and `HR`.

In Visual Studio, you can use the Granting and Revoking Privileges Dialog which allows you to grant or revoke:

- Permissions on one or more database objects to a user or role
- Permissions on one database object to multiple users or roles
- System privileges to multiple users or roles
- A role to multiple users or roles

See Also

About Privileges | Grant/Revoke Privileges Dialog

14

Performance Tuning

Performance Tuning Overview

ODT offers several different tools to assist in performance tuning your application's use of the Oracle Database:

- Oracle Performance Analyzer

  Examines the actual use of an Oracle database over a specified period of time and provides recommendations to improve application performance. See About Oracle Performance Analyzer.

- SQL Tuning Advisor

  Provides tuning recommendations for arbitrary SQL statements. Access the SQL Tuning Advisor through the Tune SQL Toolbar in the SQL and PL/SQL File Editor. See About SQL Tuning Advisor.

- Real-Time SQL Monitor

  Real-Time SQL Monitoring provides automatic monitoring of SQL statements and PL/SQL blocks. You can view a list of monitored SQL, view and save SQL Monitor Active Reports, and generate an Active Report for ad hoc SQL. See About Real-Time SQL Monitor.

About Oracle Performance Analyzer

Oracle Performance Analyzer examines the use of an Oracle database over a specified period of time and provides recommendations to improve the performance of the database applications used. In some cases, the recommendations can be automatically implemented by clicking a button.

Oracle Performance Analyzer is implemented using AWR snapshots and ADDM tasks For more information on AWR and ADDM, see *Oracle Database Performance Tuning Guide*.

This section covers the following topics:

Privileges

You must have the SYSDBA role to use Oracle Performance Analyzer.

Licensing

Oracle Performance Analyzer requires the Oracle Diagnostic Pack license and SQL Tuning Advisor requires Oracle Tuning Pack License. Dialog boxes appear to notify you of licensing requirements.

Oracle Performance Analyzer Interface

The Oracle Performance Analyzer user interface details are documented in the Oracle Performance Analyzer topic of Wizards, Designers, and Dialogs online help section.

Using the Oracle Performance Analyzer

Oracle Performance Analyzer creates an ADDM task that analyzes database activity between two points in time (represented by AWR snapshots) and displays the results. Sufficient database activity is required to obtain any tuning recommendations and often this requires many minutes of run time. There are different ways to invoke Oracle Performance Analyzer, depending on the length of time you plan to analyze.

This section covers the following topics:

- Analyzing Performance Over Short Periods of Time
- Analyzing Performance Over Longer Periods of Time
- Viewing Past Performance Analyzer Results

Analyzing Performance Over Short Periods of Time

Oracle Performance Analyzer includes a timer that automatically creates AWR snapshots and an ADDM task and displays the results. This is useful for shorter periods of time where Visual Studio can remain open during the entire analysis. This is also the simplest way to analyze performance and is a good starting point in learning about Oracle performance tuning features.

To use the Oracle Performance Analyzer timer, perform the following steps:

1. Select a SYSDBA database connection in Server Explorer that represents the database to be tuned. Right-click, and from the menu, select the **Oracle Performance Analyzer** to launch the designer.
2. Start the application that accesses the database to be tuned. This can be any application running in Visual Studio, any executable running outside of Visual Studio, any application running in a web server, and so on.

> **Note**
>
> In most cases, you should perform this step (start the application) before Step #3 (running the Oracle Performance Analyzer) to allow the application time to connect and do other start-up activities. These activities are generally not of interest when doing performance tuning. However, if your application executes SQL that runs immediately after the application begins, you may wish to run Oracle Performance Analyzer to make sure that the SQL is tuned. If so, perform Step #3 before Step #2.

3. Enter the amount of time for the analyzer to run, in hours and minutes, and click **Start**. The analyzer requires at least 5 minutes of database time to generate any results. Database time is a indicator of the total database workload. It represents the total time spent in database calls. Generally, database time is a small fraction of the actual Performance Analyzer run-time.

   A status message appears, showing the time before the analysis completes.

4. When the analysis is complete, select the root of the Performance Analyzer tree, which populates the details side of the Performance Analyzer tab.

5. Click on each finding in the tree to view its details. Findings can have one or more recommendations. Click on recommendation nodes to view them. These recommendations may contain actions that you can take to resolve issues. Click on action nodes to view them.

   If there is insufficient database activity to successfully analyze performance, a message appears in the Additional Information section indicating this. This message also tells you the actual database time that was used. Repeat the test with greater database activity and longer run-time, or both.

6. Click **Tune SQL** to display recommendations, if the Tune SQL control appears.

   The **Tune SQL** control only appears when findings suggest that you should run SQL tuning advisor.

7. Read the recommendations in the Tune SQL tab.

   Recommendations of type `STATISTICS` or `SQL PROFILE` activate the Implement Recommendation and Preview SQL controls.

8. Click **Implement Recommendation** to perform the recommendations or click **Preview SQL** to view the SQL that implements the recommendation.

Analyzing Performance Over Longer Periods of Time

If the period of time being analyzed is so long that Visual Studio may need to be closed, or for more flexibility, you can create AWR Snapshots and ADDM Tasks manually. Manually created AWR snapshots can also be configured to include more statistics than the default.

To analyze performance by manually creating AWR snapshots and ADDM tasks, perform the following:

1. Start the application that accesses the database to be tuned. This can be any application running in Visual Studio, any executable running outside of Visual Studio, any application running in a web server, and so on.

> **Note**
>
> In most cases, you should perform this step (start the application) to allow the application time to connect and do other start up activities. These activities are generally not of interest when doing performance tuning. However, if your application executes SQL that runs immediately after the application begins, you may wish to create a snapshot to make sure that the SQL is tuned. If so, perform Step #2 before Step #1.

2. Select a `SYSDBA` database connection in Server Explorer that represents the database to be tuned.
3. Right-click on the AWR Snapshots node and select **New AWR Snapshot** from the menu.
4. In the New AWR Snapshot Dialog, select **Typical** or **All** for snapshot level depending on the depth of analysis you wish to achieve. Then press **OK**. At this point, you may close Visual Studio if desired.
5. After the time period being analyzed has elapsed, perform steps 2-5 again to create a second new snapshot. Note: The analyzer requires at least 5 minutes of database time to generate any results. Database time is a indicator of the total database workload. It represents the total time spent in database calls. Generally, database time is a small fraction of the actual Performance Analyzer run-time.
6. Right-click on the ADDM Tasks node and select **New ADDM Task** from the menu.
7. In the New ADDM Task Dialog, from the lists, choose the snapshot you created as the Begin time and choose the second snapshot you created as the End time. Then Press **OK**.

   Oracle Performance Analyzer launches, displaying the results of the analysis.
8. Select the root of the Performance Analyzer tree, which populates the details side of the Performance Analyzer tab.
9. Click on each finding in the tree to view its details. Findings can have one or more recommendations, which may contain actions that you can take to resolve issues. Click on the recommendation nodes to view them, and click on any action nodes to view them.

   If there is insufficient database activity to successfully analyze performance, a message appears in the Additional Information section indicating this. This message also tells you the actual database time that was used. Wait for a longer period of time and/or increase database activity. Then return to step 2 to create another snapshot and another ADDM Task.
10. Click **Tune SQL** to display recommendations, if the Tune SQL control appears.

    The **Tune SQL** control only appears when findings suggest that you should run SQL Tuning Advisor.
11. Read the recommendations in the Tune SQL tab.

    Recommendations of type `STATISTICS` or `SQL PROFILE` activate the Implement Recommendation and Preview SQL controls.
12. Click **Implement Recommendation** to perform the recommendations or click **Preview SQL** to view the SQL that implements the recommendation.

Viewing Past Performance Analyzer Results

Oracle database stores all performance tuning results in the database as ADDM Tasks. These are available in the ADDM Tasks Server Explorer collection.

To view the results of past Performance Analyzer results, do the following:

1. In Server Explorer, select a SYSDBA database connection that represents the database that was tuned.
2. Expand the ADDM Tasks node to reveal individual ADDM Tasks.
3. Double-click the ADDM Task of interest, to launch Oracle Performance Analyzer and display the tuning results.

See Also:

Starting the Oracle Performance Analyzer

About SQL Tuning Advisor

SQL Tuning Advisor provides recommendations for improving the performance of SQL statements. You can run SQL Tuning Advisor directly from Query Window on SQL that you have entered. You can also run Oracle Performance Analyzer to tune your running application and if it detects poorly performing queries, it will prompt you to run SQL Tuning Advisor directly from the Oracle Performance Analyzer interface.

You can find the documentation for the SQL Tuning Advisor interface in the following location:

Action Node - Describes how to run SQL Tuning Advisor in Oracle Performance Analyzer in response to recommendations that SQL needs to be tuned.

Privileges

A user must be granted the ADVISOR privilege to be able to use SQL Tuning Advisor.

Licensing

SQL Tuning Advisor requires an Oracle Tuning Pack License. Dialog boxes appear to notify you of licensing requirements.

See Also

Oracle Performance Analyzer Interface | #unique_98

About Real-Time SQL Monitor

Real-Time SQL Monitoring provides automatic monitoring of SQL statements, PL/SQL blocks, or composite database operations that are considered expensive. You can view a list of monitored SQL, view and save SQL Monitor Active Reports, and generate an Active Report for ad hoc SQL. The information provided by Real-Time SQL Monitoring is especially useful for long-running SQL statements.

You can find documentation for Real-Time SQL Monitoring in the following three locations:

- Real-Time SQL Monitor Window - View the list of monitored SQL and PL/SQL and sort by metrics of your choice. View and/or save Active Reports for any of the monitored SQL and PL/SQL.
- Active Report Window - View Active Reports from inside of Visual Studio
- SQL and PL/SQL File Editor - Enter Ad-hoc SQL or PL/SQL and then click the **Execute and Monitor SQL** toolbar icon or select it from the menu.

Privileges

No special privileges are required to use Real-Time SQL Monitoring. However, the `SELECT_CATALOG_ROLE` and `ALTER SESSION` privileges are useful.

Any user can view complete Active Reports for SQL they execute using schema objects they own. If your user has been granted the `SELECT_CATALOG_ROLE`, then you will be able to view Active Reports for SQL executed by other users. When this role has not been granted, and the schema objects involved are not owned by the user, the Active Report may be missing information, such as the Explain Plan.

If a user does not have `ALTER SESSION` privileges, using Execute and Monitor SQL will result in a `/*+ MONITOR +*/` hint being added to the SQL. If the user does have `ALTER SESSION` privileges, then the SQL will not be modified.

Licensing

The use of Real-time SQL Monitoring feature requires an Oracle Tuning Pack license (except for Oracle Database Free which requires no license). For more details, please refer to the *Database Licensing Information User Manual* for the database version being used.

Real-Time SQL Monitoring Options Page

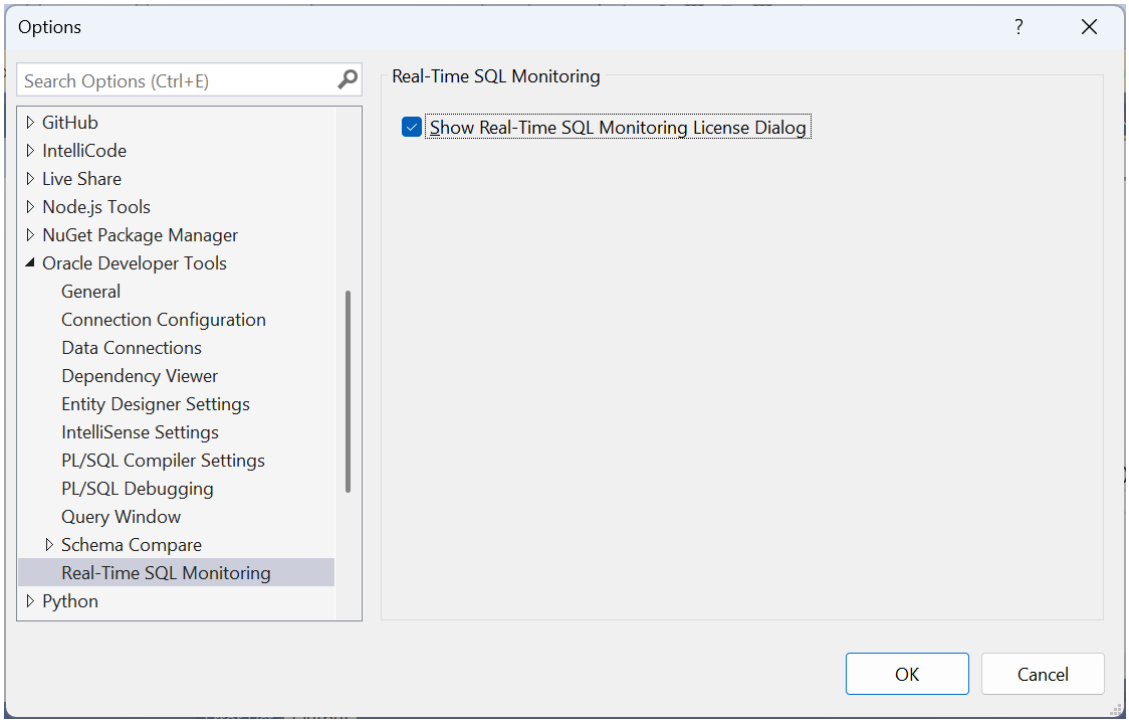
The Real-Time SQL Monitoring Options Page provides options controlling the behavior of Real-Time SQL Monitor Window.

Accessing the Real-Time SQL Monitoring Options Page

To access the Real-Time SQL Monitoring Options Page, select **Options...** from the Tools menu. From the Options menu, select **Oracle Developer Tools**. Then select **Real-Time SQL Monitoring**.

Using the Real-Time SQL Monitoring Options Page

The Real-Time SQL Monitoring Options Page appears similar to the following:



The controls of the Real-Time SQL Monitoring Options are as follows:

Control	Description
Show Real-Time SQL Monitoring License Dialog	Uncheck this option to disable the informational dialog reminding users which licenses are required to use this database feature. This option can also be set from the informational dialog itself.

15

Using Entity Framework Designer

About Entity Framework

Oracle supports Microsoft Entity Framework, an Object-Relational Mapping (ORM) technology that enable developers to write object-oriented code against a conceptual model of their data rather than accessing the database directly. A data provider translates operations on this conceptual model into SQL that executes against the database.

For example, you can update an instance of an `EMPLOYEE` class or request a collection of `EMPLOYEE` instances rather than providing the equivalent `UPDATE` or `SELECT` SQL statements to the `EMPLOYEES` table in an Oracle Database. The mapping layer generates SQL behind the scenes for you.

Oracle Data Provider for .NET (ODP.NET) provides the mapping layer for Entity Framework and Oracle Database. Oracle Developer Tools for Visual Studio (ODT) integrates with the Visual Studio Entity Data Model Designer (Entity Designer) to enable .NET developers to target Oracle Database with their Entity Framework applications when designing or editing the conceptual model.

> **See Also**
>
> Oracle Data Provider for .NET Developer's Guide

Requirements for using Entity Framework Designer

You must reference Oracle Data Provider for .NET (ODP.NET) version 23 in your Visual Studio project. You can do this by using the Nuget Package Manager and searching for and including `Oracle.ManagedDataAccess` version 23.

Any other version of ODP.NET will result in the Entity Framework Designer crashing or throwing errors.

If you must use older versions of ODP.NET, then you will also need to use older versions of Oracle Developer Tools for Visual Studio. The major version of Oracle Developer Tools for Visual Studio must match the major version of the Oracle Data Provider for .NET referenced by your project.

Visual Studio Entity Designer

The Visual Studio Entity Designer (Entity Designer) allows you to create or modify the entity model. With Oracle Developer Tools for Visual Studio installed, this designer can connect to the Oracle Database to create entities based on existing Oracle schema objects.

Alternately, you can design the entities in the designer. Then, you can have the designer generate a SQL script that creates Oracle schema objects based on this design.

Oracle Stored procedures or functions can also be mapped to Entity Functions for explicit calls from the code, or they can be mapped (and automatically called) on `INSERT`, `UPDATE`, and `DELETE` operations on entities.

This section covers these topics:

Generating an Entity Model from an Oracle Database

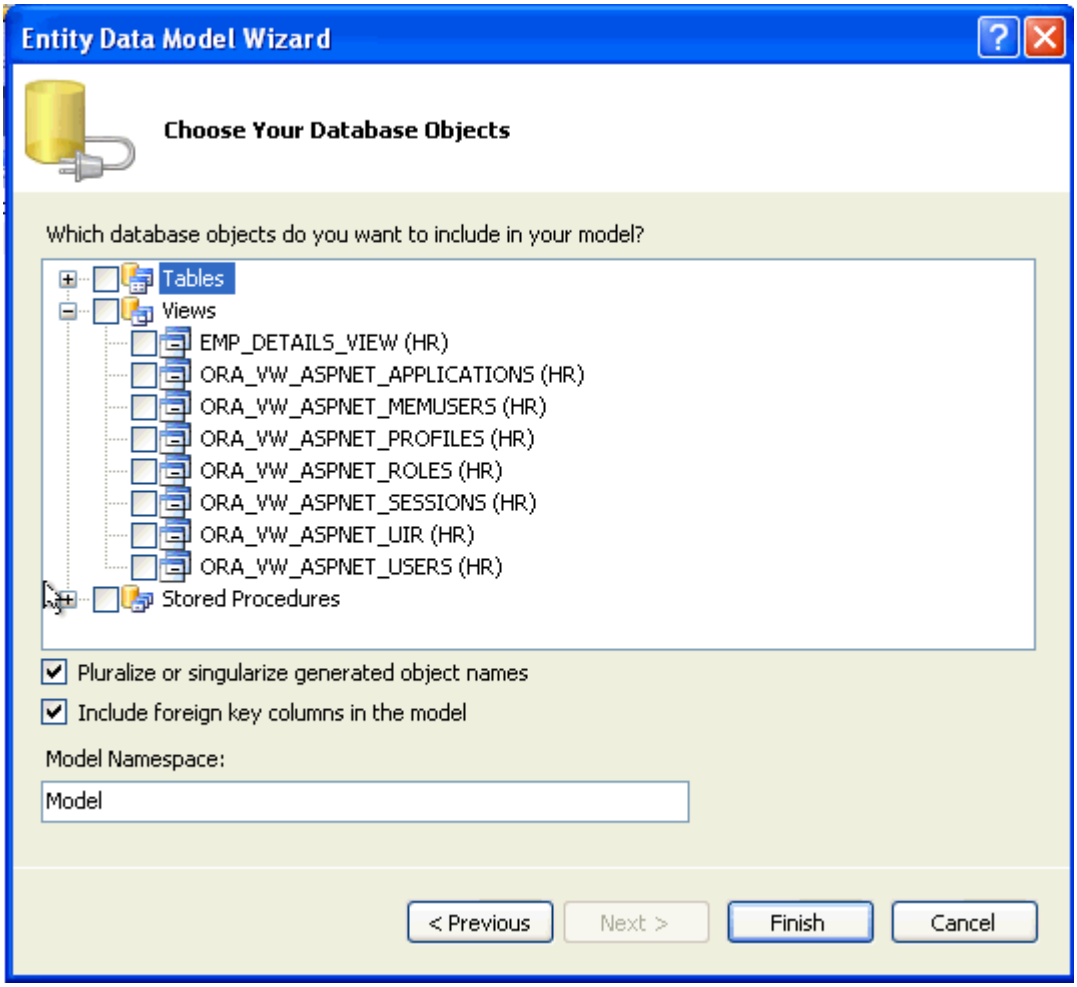
To generate an Entity Model from an Oracle Database, follow these steps:

1. Create a project, such as a C# Windows Forms project.
2. In the Solution Explorer, highlight the project, right-click and select **Add**, then select **New Item** to add a new item to the project.
3. Select **ADO.NET Entity Data Model** and click **Add**.

   The Entity Designer appears.
4. Select **Generate from database** and click **Next**.
5. Choose a connection to an Oracle database.
6. Choose the database objects you want in the Entity Model.

   Check boxes as needed and provide the Model Namespace. Click **Next**.

   The Entity Data Model appears in the designer.

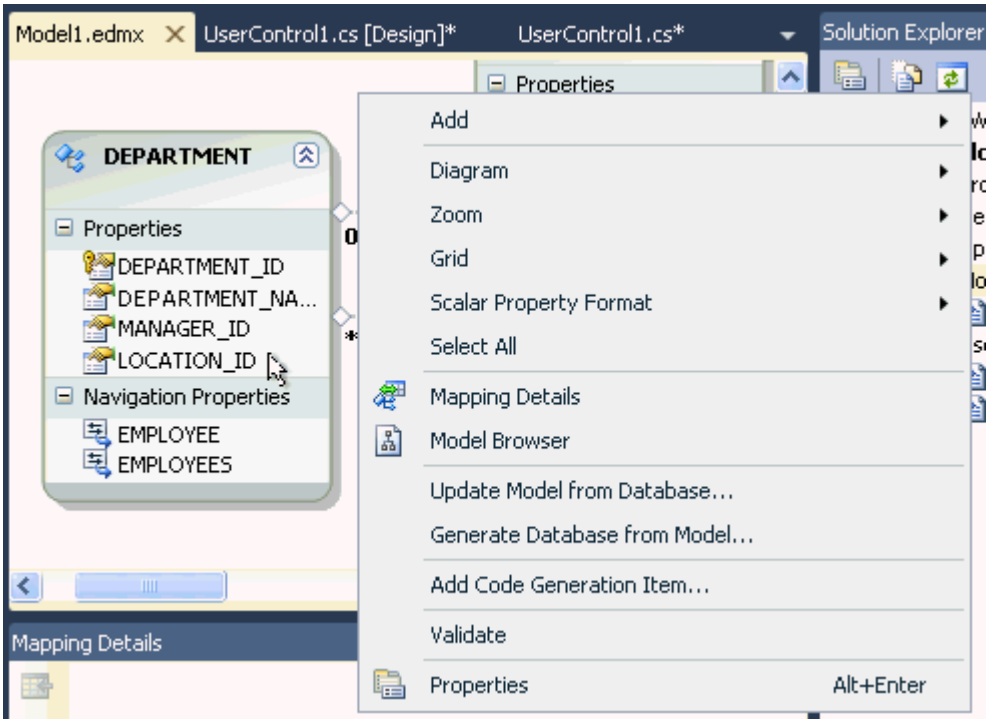


Updating an Entity Data Model from an Oracle Database

You can update an entity data model from an Oracle database.

To update an existing Entity Data Model from an Oracle Database, do the following:

1. Right-click in the **Entity Designer** and select **Update Model from Database**.



2. If you have not done so previously, choose a connection to an Oracle database.
3. Choose the database objects you want in the Entity Model.

Generating an Oracle Database Create Script from an Entity Model

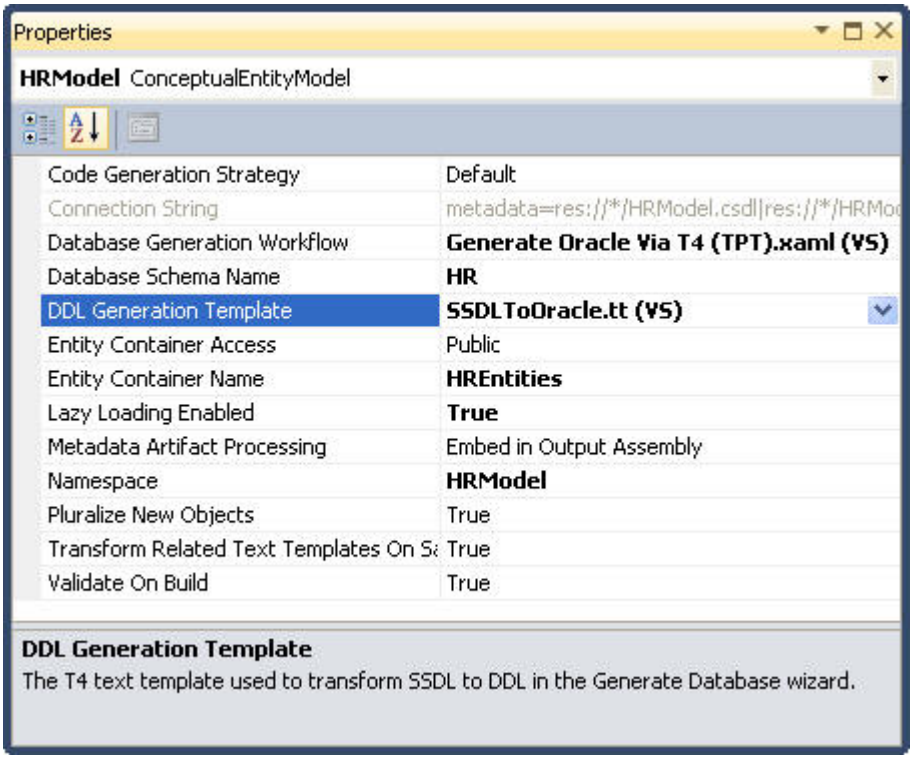
You can generate an Oracle database create script from an entity data model using one of two strategies each of which determines how entities are mapped to database tables:

- **Table-per-type:** Using table-per-type, each entity is mapped to its own database table.
- **Table-per-hierarchy:** Using table-per-hierarchy, an entity and its derived types are mapped to a single database table. This database table includes columns for all possible properties of an entity and its derived types. The database table also includes a discriminator column that indicates for each row, what type of entity it is.

To generate an Oracle database create script, do the following:

1. Open up the Entity Data Model designer to display your entities.
2. In the Properties Window, for the ConceptualEntityModel, set the following properties:
   - Database Generation Workflow, select one of the Oracle Workflows:
     - Table-per-type strategy, select **Generate Oracle Via T4 (TPT).xaml**
     - Table-per-hierarchy, select **Generate Oracle Via T4 (TPH).xaml**
   - For DDL Generation Template, select Oracle template, **SSDLToOracle.tt**
   - For Database Schema Name, specify a schema in the Oracle Database.

> **Note**
>
> The schema name is case-sensitive.



3. Right-click the Entity Designer.

   Select **Generate Database from Model**.

   The Generate Database Wizard appears.

4. Choose a connection to an Oracle Database.

5. Preview the DDL in the Summary and Settings panel, and click **Finish** to generate the SQL script. Remove the comments in front of destructive DDL commands such as "DROP".Execute the SQL script to generate a database by selecting the Visual Studio Tools menu and starting the Run SQL*Plus Script Dialog.

> **Note**
>
> With Oracle Database Release 12.1 or later, on the Entity Designer Settings Options page, you can enable the **Set maximum size for extended datatype columns to 32767** option. The generated DDL then uses a maximum size of 32767 for extended types. You can also enable the **Generate IDENTITY column(s)** option, which will cause the generated DDL to use `IDENTITY` columns rather than a sequence and trigger combination.

Using Add Import Function Dialog to Import an Oracle Stored Procedure

You can map an Entity Function to an Oracle Stored Procedure by using the Add Function Import Dialog.

To map an Oracle stored procedure to the Entity Model, perform the following steps:

1. Generate or update an Entity Model from an Oracle Database.
2. If you have not done so previously, choose a connection to an Oracle database.

3. In the window to Choose Your Database Objects, select some stored procedures or functions or both.

> **Note**
>
> To add entity function imports that have return values, such as scalars, complex types, or entities, you must select Oracle stored procedures that include a `REF CURSOR OUT` or `IN OUT` parameter. The `REFCURSOR` is then mapped to the Entity Function return value. If an Oracle stored procedure or function returns multiple `REF CURSORs`, only the one is used as the return value for an import function.
>
> It is also possible to import Oracle Stored procedures or functions that do not include a `REF CURSOR`, but may have `OUT` or `IN OUT` parameters. In that case, the imported entity function has no return value, and you can bind to the output parameters to retrieve data.

4. To add function imports that have a return value (such as an Oracle Stored procedure that contains a `REF CURSOR`), configure the `REF CURSOR` metadata information in the `app.config` or `web.config` file in your Visual Studio project.

   You can automatically generate this metadata information as follows:

   a. Connect in Server Explorer to the Oracle Database containing the stored procedure or function. Be sure to connect as the data source of the type of ODP.NET that your Entity Framework application uses, as described in Connection Dialog Box , either Managed ODP.NET or Unmanaged ODP.NET. This is required because automatically generated metadata information has different formats for the two types of ODP.NET.

   b. Navigate the Server Explorer tree control to the stored procedure or function node that you wish to generate the metadata information for.

   c. Right-click on the stored procedure or function node and select **Run** from the menu.

   d. If there are input parameters, enter a value or a set of values into the Stored Procedure Run Dialog Box .

      The results window appear in the Stored Procedure Run Dialog Box .

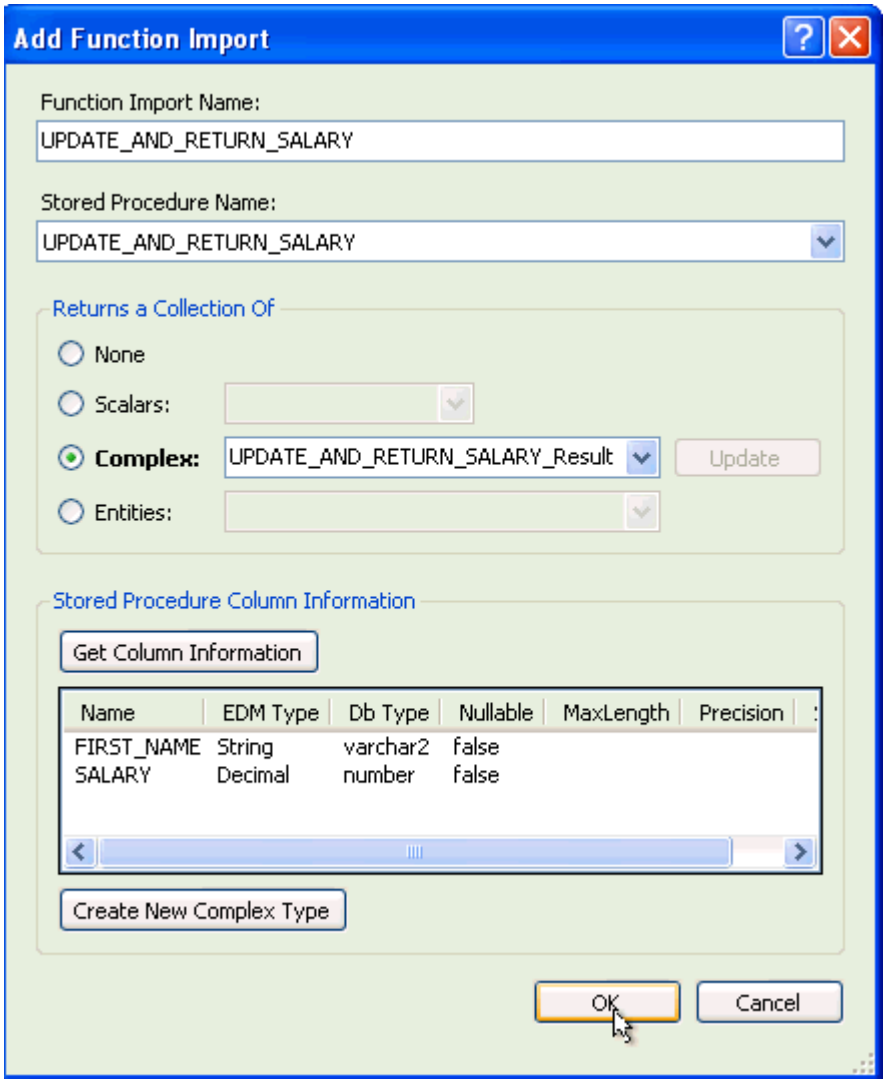
   e. If you intend to use the **Add Import Function Dialog** to create an Import Function that returns a collection of complex types, select all of the **Select for Config** check boxes. For Import Functions that return a collection of entities or scalars, this is not required.

   f. Click the **Add Config to Project** button to add the metadata information to your `app.config` or `web.config`.

   Note: For more information on configuring `REF CURSOR` metadata information, see Implicit REF CURSOR Binding Support in *Oracle Data Provider for .NET Developer's Guide*.

5. In the Model Browser, under Store, click the Stored Procedures node, and then right-click the name of the Oracle Stored Procedure. From the menu, select **Add**, then **Function Import**.

   The Add Function Import dialog appears.



6. Select one of the **Returns a Collection of** options. If you select **Complex**, you can then select **Get Column Information**. This will read your `app.config` or your `web.config` to get metadata information about the `REF CURSOR` that contains the return values.

> **Note**
>
> A return value mapped to a Complex Type is not supported with Oracle Stored Functions. The error `Function Imports cannot be created for composable functions` will result. Consider creating a wrapper Oracle Stored Procedure to work around this.
>
> If the `app.config` or `web.config` metadata information is incorrect, no column information appears when Get Column Information is selected. You must correct the `config` file.

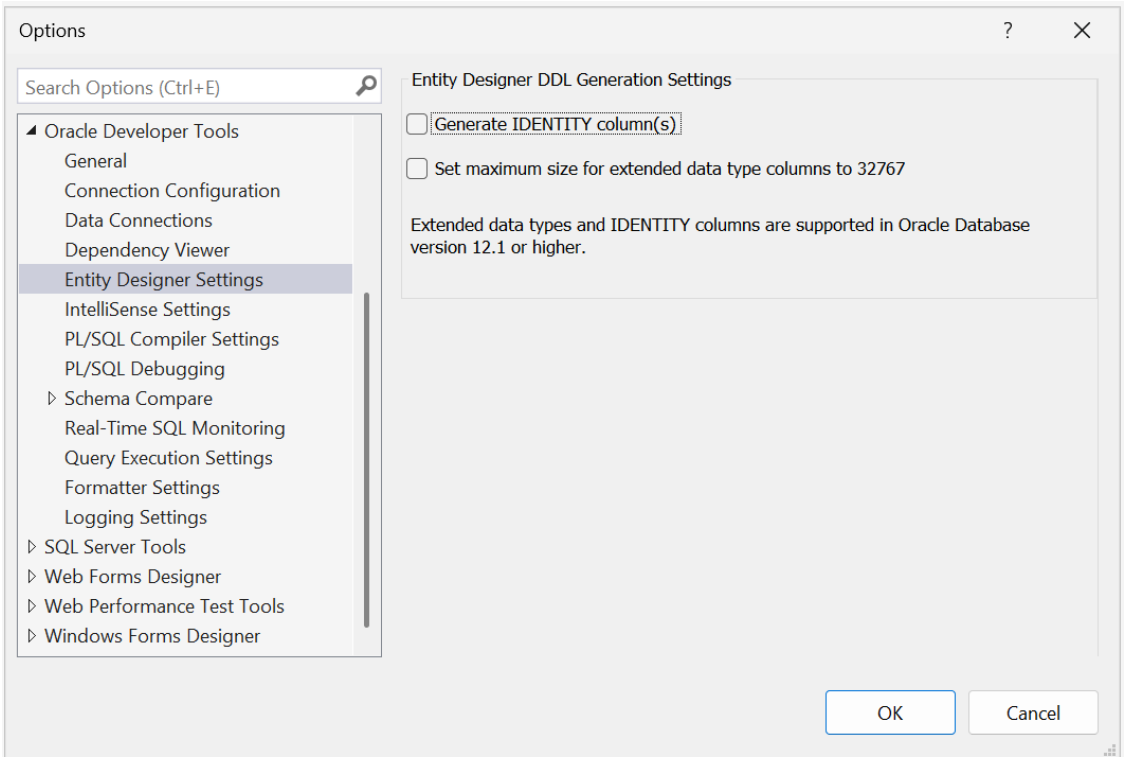
7. After column information appears, select **Create New Complex Type**.
8. Select **OK** to add the function import to the conceptual model.

Entity Designer Settings Options

The Entity Designer Settings Options page allows you to control certain aspects of how Entity Designer generates an Oracle Database create script from an Entity Model. These options only apply when connected to an Oracle database release 12.1 or higher.

Accessing Entity Designer Settings Options Page

To access the General Options Page, select **Options**... from the **Tools** menu. From the **Options** menu, select **Oracle Developer Tools**. Then select **Entity Designer Settings**.



Using the Entity Designer Settings Options

The controls of the Entity Designer Settings Options are as follows:

Control	Description
Generate IDENTITY column(s)	When enabled, Entity Framework Model DDL generation (described in Generating an Oracle Database Create Script from an Entity Model) uses `IDENTITY` columns (rather than a sequence and trigger combination). This is only available in Oracle Database Release 12.1 or later.
Set maximum size for extended data type columns to 32767	When enabled, Entity Framework Model DDL generation (described in Generating an Oracle Database Create Script from an Entity Model) uses Oracle extended data types of size 32767. This is only available in Oracle Database Release 12.1 or later.

16

Schema Compare

About Schema Compare

The Schema Compare tool allows you to compare entire schemas or portions of schemas across two Server Explorer Oracle Database connections, between an Oracle Database Project Version 2 and an Oracle Database Connection, or between two Oracle Database Project Version 2 projects.

You can view the comparison results in a results window to see the differences. You can also generate a deployment script, a *diff* script, that can be used to upgrade the target database schema to match the source schema (for Oracle Database connection targets only). You can also tell Schema Compare to update the target directly without generating a script (for either Oracle Database connection or Oracle Database Project Version 2 target types).

Using Schema Compare

You can launch the Schema Compare tool from the Visual Studio Tools menu, from the Project folder in a Oracle Database Project Version 2, or from the context menu for a Data Connection node in Server Explorer described in the Data Connection Node Menu Options.

Once launched, the Schema Compare Source and Target Dialog appears.

You choose source and target Server Explorer database connections and/or Oracle Database Project Version 2 projects. You can also choose to restrict the comparison to particular schema objects types. You can also filter the results to show only certain types of results, for example, only objects that are missing in the target.

When comparing a Oracle Database Project Version 2 project to a Oracle Database connection, the database connection should contain one of the schema names that is included in the database project.

The Schema Compare Results Window opens.

This displays a tree control representing the results hierarchy. You can drill down and view the differences.

When you select an Object Type node (or any of its children), the Attributes and Object Definition tabs show detailed differences between the source and target. The Update SQL tab shows the SQL required to modify the target so that the object matches the source.

Clicking the Update Target toolbar button on the Results window will update a target Database Connection or Oracle Database Project Version 2 project to upgrade the target schema to match the source schema.

If the target is a Oracle Database connection, then clicking the Generate Script toolbar button will generate a deployment script (a *diff* script). This can be run against the target database to upgrade the target schema to match the source schema.

Schema Compare Options

You can choose the default settings for filtering comparison results by object type and results type in the Schema Compare Source and Target Dialog in the Schema Compare Options page, shown in this section.

The selections that you make determine what options are checked by default when you click the **Advanced** button in Schema Compare Source and Target Dialog .

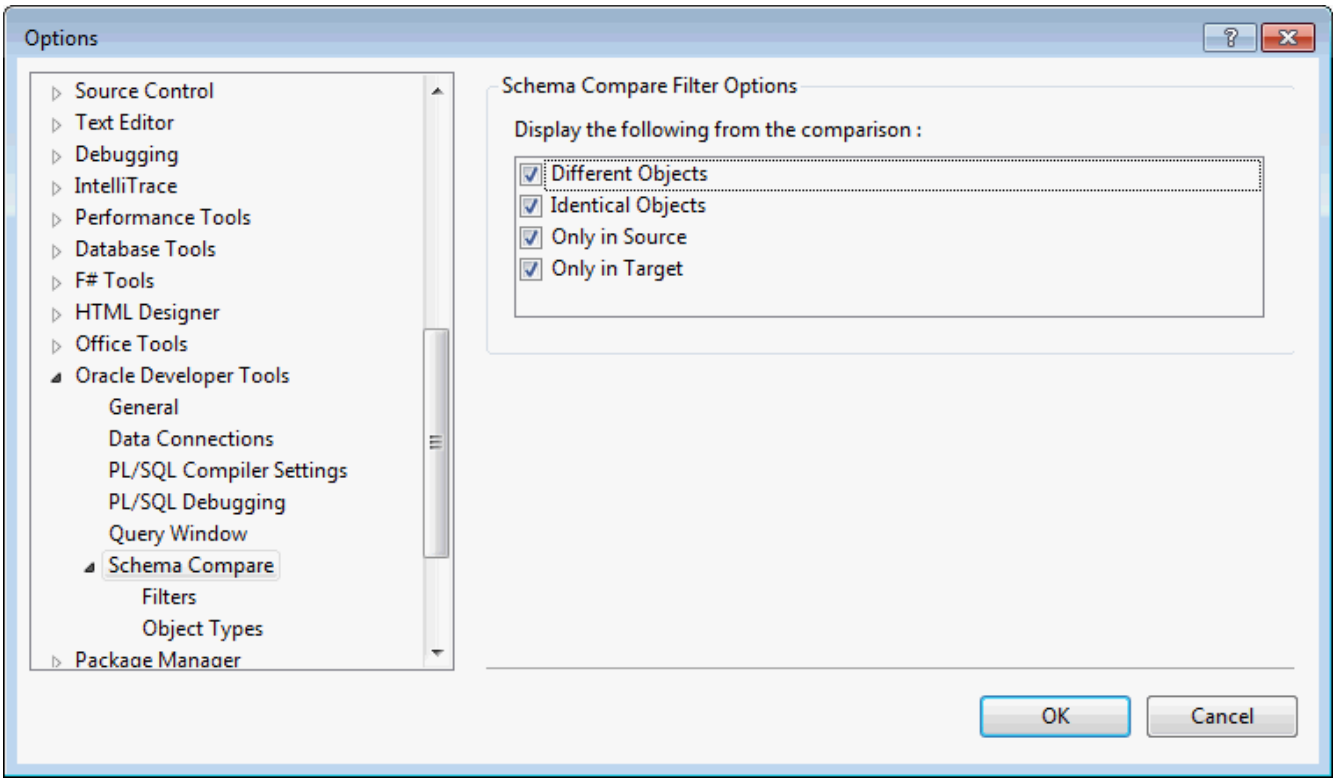
Accessing the Schema Compare Options

To access the Schema Compare Filters Options Page, select **Options**... from the **Tools** menu. From the **Options** menu, select **Oracle Developer Tools**. Then select **Schema Compare**..

Using the Schema Compare Filter Options Settings

The Schema Compare Filter Options page appears similar to the following:

Options for Schema Compare Filters



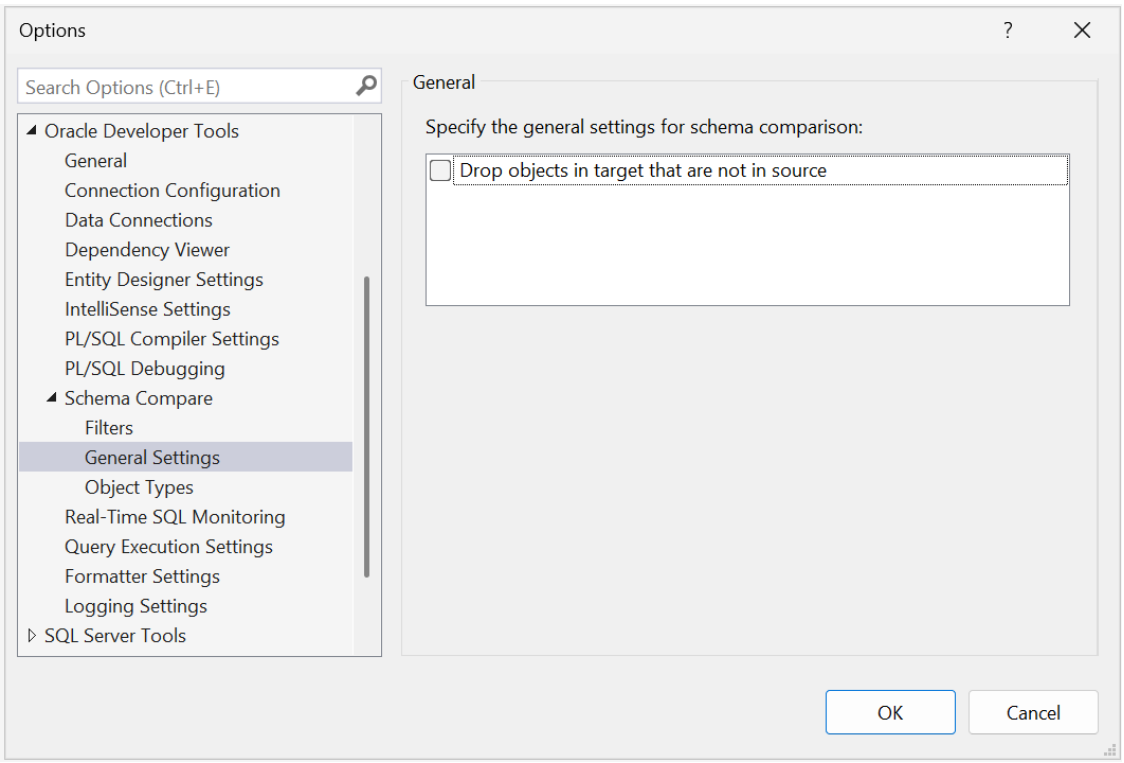
Items selected in the Schema Compare Filter Options page become the defaults for the Schema Compare Source and Target Dialog , Filter Types.

The controls of the Schema Compare Filter Options page are as follows:

Control	Description
Display the following from the comparison	Lists the types of results that will be displayed, with check boxes to enable them.
OK	Saves and closes the tab.
Cancel	Cancels any changes and closes the tab.

Options for Schema Compare General Settings

The Schema Compare General Settings dialog appears similar to the following:



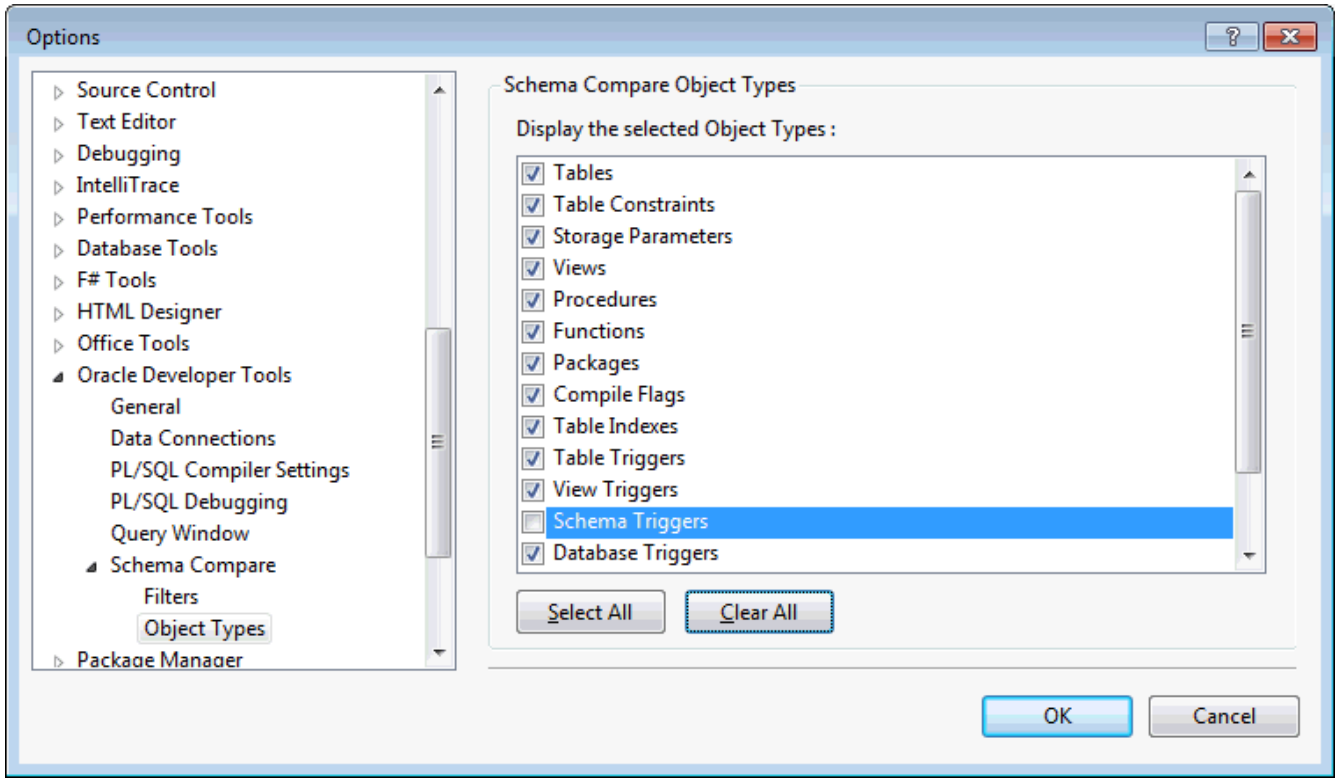
The Schema Compare General Settings dialog has the following controls:

Control	Description
Drop Objects in Target that are not in Source	Objects that are not in the source but that exist in the target will be dropped (deleted) from the target database.

Using the Schema Compare Object Types Options Settings

The Schema Compare Object Types Options page appear similar to the following:

Options for Schema Compare Object Types



Items selected in the Schema Compare Object Types Options page become the defaults for the Schema Compare Source and Target Dialog , Object Types.

The controls of the Schema Compare Object Types Options are as follows:

Control	Description
Display the selected Object Types	Lists the types of objects that will be compared, with check boxes to enable them.
Select All	Enables you to select all of the available Object Types.
Clear All	Enables you to clear any checked selections.
OK	Saves and closes the tab.
Cancel	Cancels any changes and closes the tab.

17

Multitenant Container Databases

Using CDBs

To manage the multitenant architecture from Visual Studio, connect as `SYSDBA` in Server Explorer to a CDB.

Once connected, you should see a Pluggable Databases Node. If you do not see it, you may not have `SYSDBA` privileges, the database may not be 12.1 or higher, or you may have accidentally connected to a PDB instead of a CDB.

This node has context sensitive menus including:

- New Pluggable Database: Opens the New Pluggable Database Dialog and allows you to create a new PDB based on the seed PDB. At the end of this operation, if the option is chosen in the General Options Page, a new connection to the PDB is made in Server Explorer, and if the CDB used a TNS connection, a connection alias for the PDB is added to the `tnsnames.ora` file.
- Plug: Opens the Plug Pluggable Database Dialog which accepts paths to an XML metadata file as well as to database files on the same computer as the CDB. Once plugged in, the PDB is visible to the other PDBs in Server Explorer. The PDB must have the same character set as the CDB or this operation will fail.

When the Pluggable Databases Node is expanded, child Pluggable Database Nodes are shown, one for each PDB. Each of these Pluggable Database nodes have menu options that allow the following operations.

- Clone: Opens the Clone Pluggable Database Dialog and makes a clone of the PDB. At the end of this operation, if the option is chosen in the General Options Page, a new connection to the PDB is made in Server Explorer, and if the CDB used a TNS connection, a connection alias for the PDB is added to the `tnsnames.ora` file.
- Unplug: Opens the Unplug Pluggable Database Dialog, creates an XML metadata file, and removes the PDB from the CDB. The PDB then exists only as a set of files on the server filesystem.
- Close: Opens the Close Pluggable Database Dialog. Performs the equivalent of a non-CDB database `SHUTDOWN`.
- Open: Opens the Open Pluggable Database Dialog. Opens the PDB so that it is accessible to users.
- Delete: Deletes the PDB from the CDB and removes its files from the server.

Connecting to a PDB in Server Explorer

If the option is enabled in General Options Page, creating a new PDB or cloning one using ODT results in a connection being made in Server Explorer using the administrator account, and if applicable, a connect alias being added to the `tnsnames.ora` file.

To manually connect to a PDB, you will need the hostname, port number, and database service name of the container database. Then you can connect to the PDB using the same hostname and port number and a database service name that consist of the pluggable database name followed by the remainder of the container database service name identifier.

For example: If a CDB is identified by:

Host: `myhost.oracle.com`

Port: `1521`

Database Service Name: `CONTAINERDB.oracle.com`

and the PDB is named `PDBORCL`,

then the PDB could be connected to using the following information:

Host: `myhost.oracle.com`

Port: `1521`

Database Service Name: `PDBORCL.oracle.com`

PDB and CDB Compatibility

To plug a PDB into a CDB, they must meet the following requirements:

- They must use the same character set.
- They must have the same endianness.
- The original CDB, which created the PDB, and the target CDB must have the same set of database options installed.

To verify if a PDB is compatible with a given CDB, execute the following SQL*Plus script as `SYSDBA` while connected to the CDB and with the database files copied to the server:

```
SET SERVEROUTPUT ON
DECLARE
  compatible CONSTANT VARCHAR2(3) :=
    CASE DBMS_PDB.CHECK_PLUG_COMPATIBILITY(
           pdb_descr_file => '<<<PDB PATH>>>',
           pdb_name       => '<<<PDB NAME>>>')
    WHEN TRUE THEN 'YES'
    ELSE 'NO'
END;
BEGIN
  DBMS_OUTPUT.PUT_LINE(compatible);
END;
/
select type, message from PDB_PLUG_IN_VIOLATIONS where name=''<<<PDB NAME>>>';
```

> **See Also**
>
> *Oracle Database Administrator's Guide* for more information on PDB and CDB compatibility

About Multitenant Container Databases

Multitenant architecture, introduced in Oracle Database 12*c* Release 1 (12.1), enables an Oracle database to function as a multitenant container database (CDB) that includes zero, one, or many customer-created pluggable databases (PDBs). A PDB is a portable collection of schemas, schema objects, and nonschema objects that appears to any Oracle client application as a *non-CDB.* Note: all Oracle database versions prior to Oracle Database 12*c* Release 1 (12.1) are known as *non-CDB*s.

A CDB contains one seed database, which is a system-supplied template that the CDB can use to create new PDBs. The seed PDB is named `PDB$SEED`. You cannot add or modify objects in `PDB$SEED`. Because new PDBs are essentially copied from the seed rather than created from scratch, PDB creation time is generally much faster than creating a non-CDB.

PDBs can also be *cloned* from other PDBs in a similarly fast amount of time.

PDBs can be *unplugged*, which means they are removed from the CDB, and exist on the database server simply as a set of files: a metadata XML file along with some database files (`.DBF`s). This combination of metadata file and database files can then be easily *plugged into* any other CDB that is compatible with the PDB. For compatibility requirements, see PDB and CDB Compatibility.

From the point of view of .NET developers who have `SYSDBA` privileges, this makes testing a database configuration with your application as simple as *cloning* an existing PDB, or *plugging in* a PDB in the form of some provided database files. Similarly, a database configuration can be shared with other developers, either by cloning or through unplugging and sharing the database files.

If a user with `SYSDBA` privileges makes a connection to a CDB in Visual Studio Server Explorer, a Pluggable Databases Node appears in the tree control. When that node is expanded, the seed PDB, along with any other available PDBs, appear as child nodes. From there, operations such as Create New, Plug, Unplug, Clone, Delete, Open, and Close can all be easily achieved.

For more information on the multitenant architecture and its benefits, see *Oracle Database Concepts* and the *Oracle Database Administrator's Guide*.

18

Troubleshooting

About Troubleshooting

Oracle Support may request that you generate some trace files and logs to assist in diagnosing a problem. This chapter will show you how to generate these files.

Logging Settings

Logging is enabled via the Logging Settings. After modifying the Logging Settings you can reproduce the problem and close Visual Studio and then retrieve the log files.

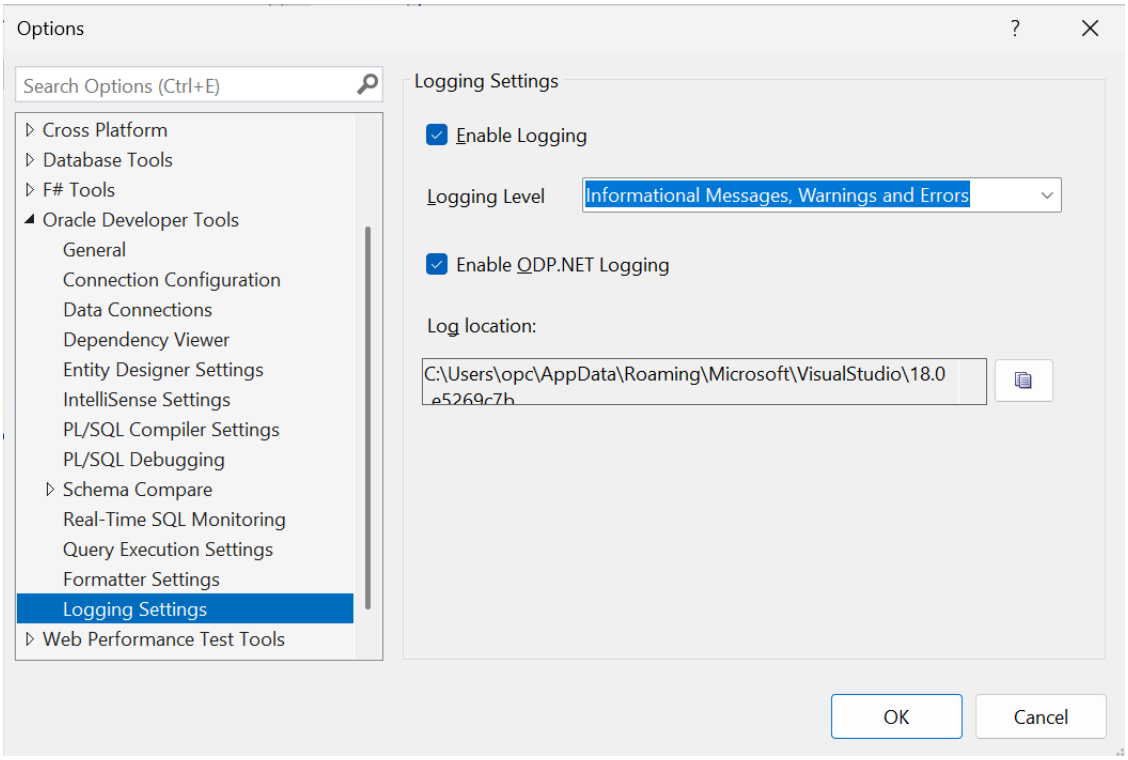
> **Note**
>
> Logging can impact the performance of Visual Studio. Disable logging as soon as you have captured the log files.

Accessing the Logging Settings

To access the Logging Settings Page, select **Options...** from the Tools menu. From the Options menu, select **Oracle Developer Tools**. Then select **Logging Settings**.

Using the Logging Settings

The Logging Settings dialog appears similar to the following:



The Logging Settings dialog has the following controls:

Control	Description
Enable logging	When this is checked logging will be enabled.
Logging Level	This controls the level of detail included in the extension's logs. Oracle Support will tell you what setting to use.
Enable ODP.NET Logging	This check box enables additional database driver level logging. Check this if requested by Oracle Support.
Log Location	This box contains the location where the log files will be created.
Copy log location to clipboard icon	Clicking this icon copies the path to the log files to the clipboard
Ok	Enables the changes and closes the dialog
Cancel	Cancels the changes and closes the dialog

Index

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X