ORACLE OPEN WORLD

NOVEMBER 11–15, 2007 | MOSCONE CENTER | SAN FRANCISCO

ORACLE DEVELOP

The Premier Conference for Developers

HANDS-ON LAB

Oracle SQL Developer

SPONSORED BY Dr.Dobb's

ORACLE

Using Oracle SQL Developer

Introduction

This hands on lab is designed to allow you to pick and choose the content you are most interested in learning about. Lessons in this lab cover different levels of experience, including those who are new to Oracle SQL Developer, those who want to find out what's new, and those who are experienced users.

Please read through the summaries of the lessons below before choosing which one you would like to start with. An estimate of the time required to complete each lesson is provided.

Lesson Summaries

Managing Your Database Objects Using Oracle SQL Developer 1.2

Purpose

This tutorial introduces Oracle SQL Developer and shows you how to manage your database objects.

Time to Complete

Approximately 30 minutes

Developing and Debugging PL/SQL using Oracle SQL Developer

Purpose

This tutorial shows you how to create, run, and debug a PL/SQL procedure using Oracle SQL Developer.

Time to Complete

Approximately 30 minutes

Creating XML Extensions using Oracle SQL Developer

Purpose

This tutorial shows you how to add XML extensions to Oracle SQL Developer.

Time to Complete

Approximately 30 minutes

Migrating a Microsoft Access Database to Oracle

Purpose

This tutorial describes how to migrate a Microsoft Access Database to Oracle.

Time to Complete

Approximately 30 minutes

Managing Your Database Using Oracle SQL Developer

Purpose

This tutorial introduces Oracle SQL Developer and shows you how to manage your database objects.

Time to Complete

Approximately 30 minutes

Topics

This tutorial covers the following topics:

Viewing Screenshots

Place the cursor over this icon to load and view all the screenshots for this tutorial. (Caution: This action loads all screenshots simultaneously, so response time may be slow depending on your Internet connection.)

Note: Alternatively, you can place the cursor over an individual icon in the following steps to load and view only the screenshot associated with that step. You can hide an individual screenshot by clicking it.

Overview

Oracle SQL Developer is a free graphical tool that enhances productivity and simplifies database development tasks. Using SQL Developer, users can browse database objects, run SQL statements, edit and debug PL/SQL statements and run reports, whether provided or created.

Developed in Java, SQL Developer runs on Windows, Linux and the Mac OS X. This is a great advantage to the increasing numbers of developers using alternative platforms. Multiple platform support also means that users can install SQL Developer on the Database Server and connect remotely from their desktops, thus avoiding client server network traffic.

Default connectivity to the database is through the JDBC Thin driver, so no Oracle Home is required. To install SQL Developer simply unzip the downloaded file. With SQL Developer users can connect to Oracle Databases 9.2.0.1 and later, and all Oracle database editions including Express Edition.

Back to Topic List

Prerequisites

For this Hands On Session, the following has already been installed for you::

1. Oracle Database 10g.
   Note: To repeat this exercise later you can use any Oracle Database above 9.2.0.1

2. Oracle SQL Developer 1.2.1.
   Note: Oracle SQL Developer is available for download for FREE from **OTN.** To install Oracle SQL Developer, unzip it into any directory on your machine.

3. The files you use throughout the tutorial are located in the **\SQLDev_HOS\GettingStarted\files** directory. The files have also been included in the **sqldev.zip** file.

4. The shipped **HR** schema.

4. There should be no DEPENDENTS table. To delete the DEPENDENTS table open SQL*Plus and execute the following commands:

   connect hr/hr
   drop table dependents;

Back to Topic List

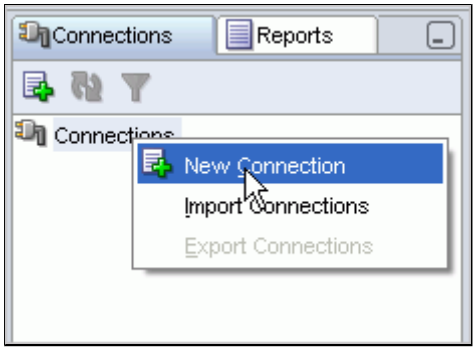
Creating a Database Connection

The first step to managing database objects using Oracle SQL Developer is to create a database connection. Perform the following steps:

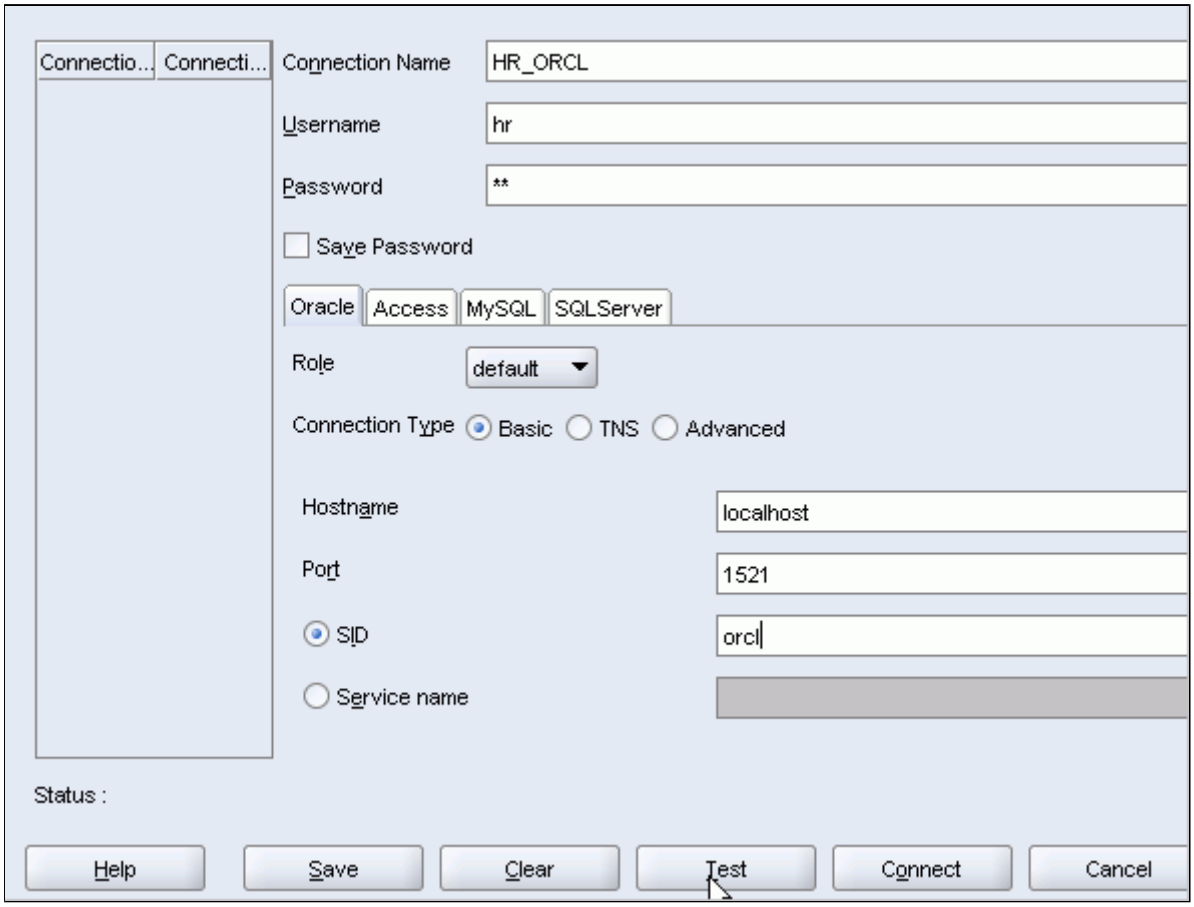
1. Open Windows Explorer and double-click **<your_path>\sqldeveloper\sqldeveloper.exe**.

   Note: you can also execute sqldeveloper in a non-windows environment.

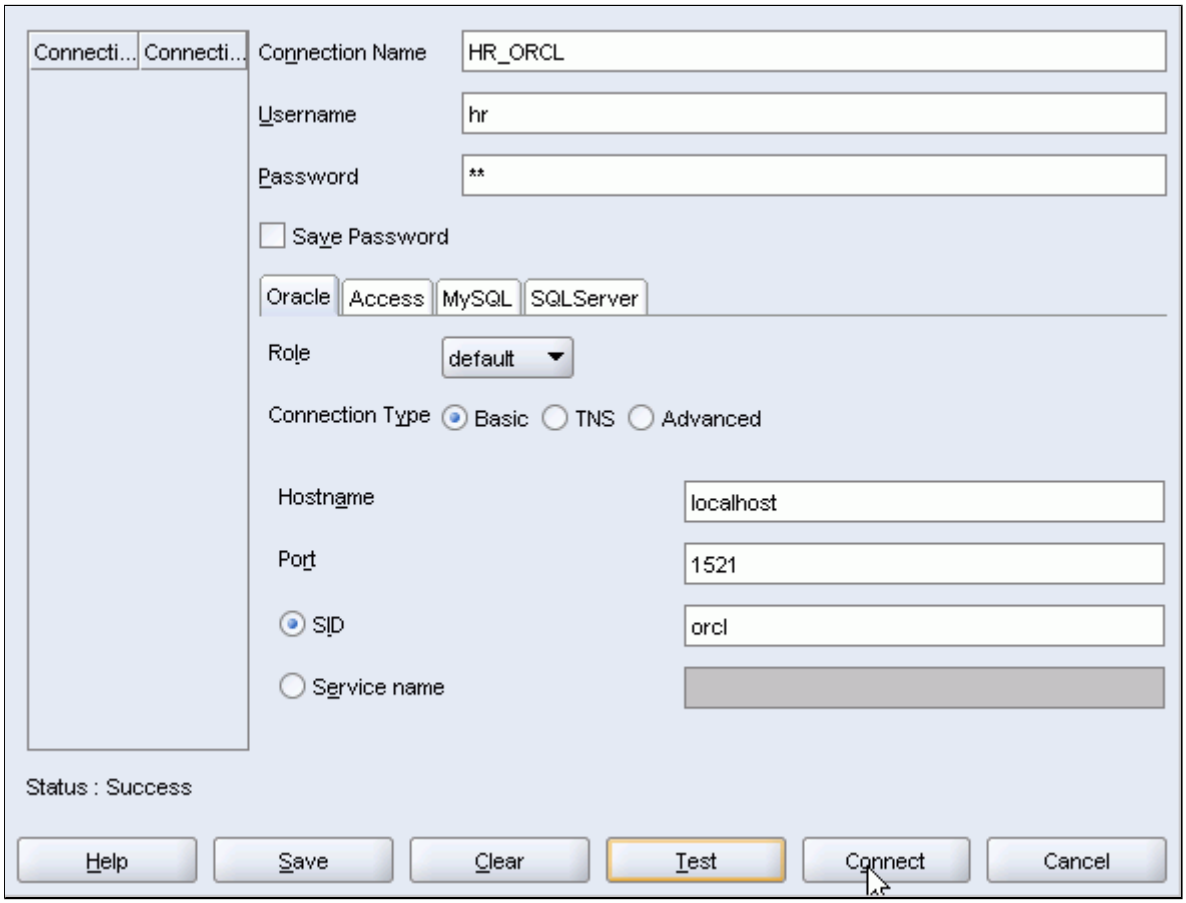
2. In the Connections tab, right-click **Connections** and select **New Database Connection**.



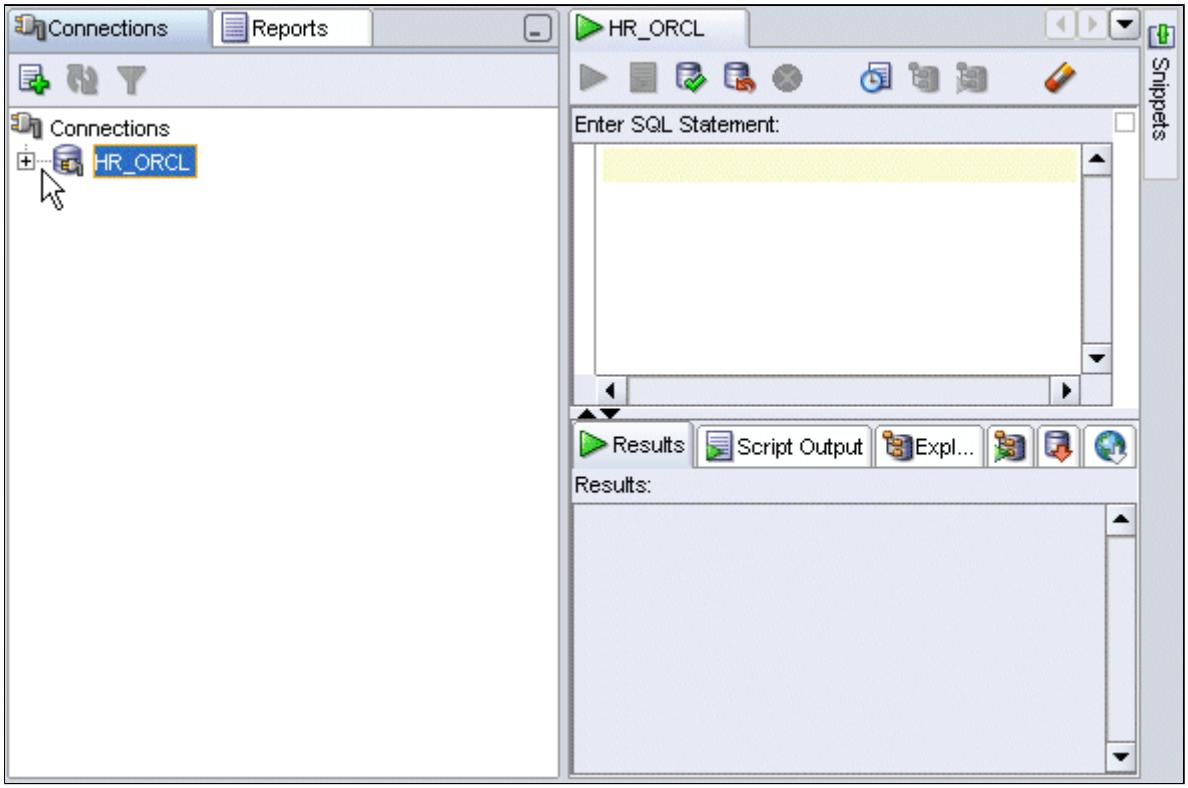
3. Enter **HR_ORCL** for the Connection Name (or any other name that identifies your connection),**HR** for the Username and Password, specify your **<hostname>** for the Hostname and enter **ORCL** for the SID. Click **Test**.



4. The status of the connection was tested successfully. The connection was not saved however. Click **Save** to save the connection, and then click **Connect**.

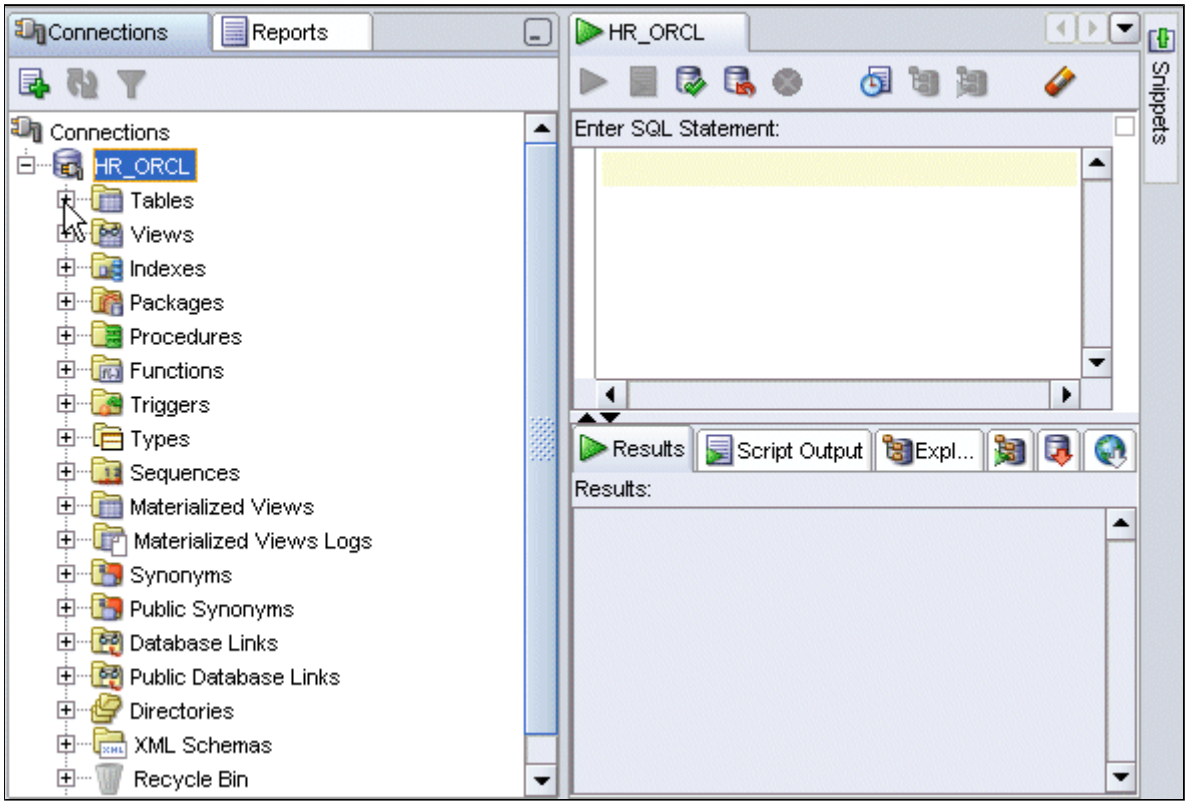


5. The connection was saved and you see the database in the list. Expand **HR_ORCL**.

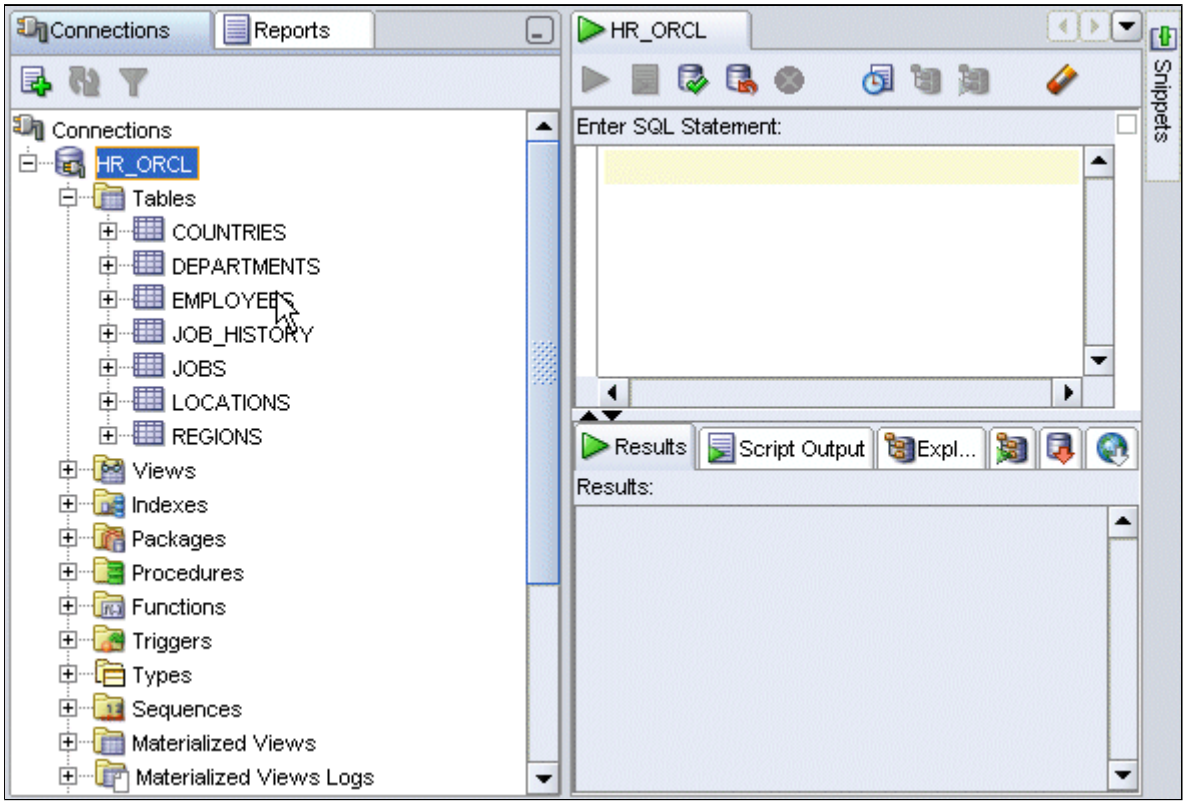


Note: When a connection is opened, a SQL Worksheet is opened automatically. The SQL Worksheet allows you to execute SQL against the connection you just created.

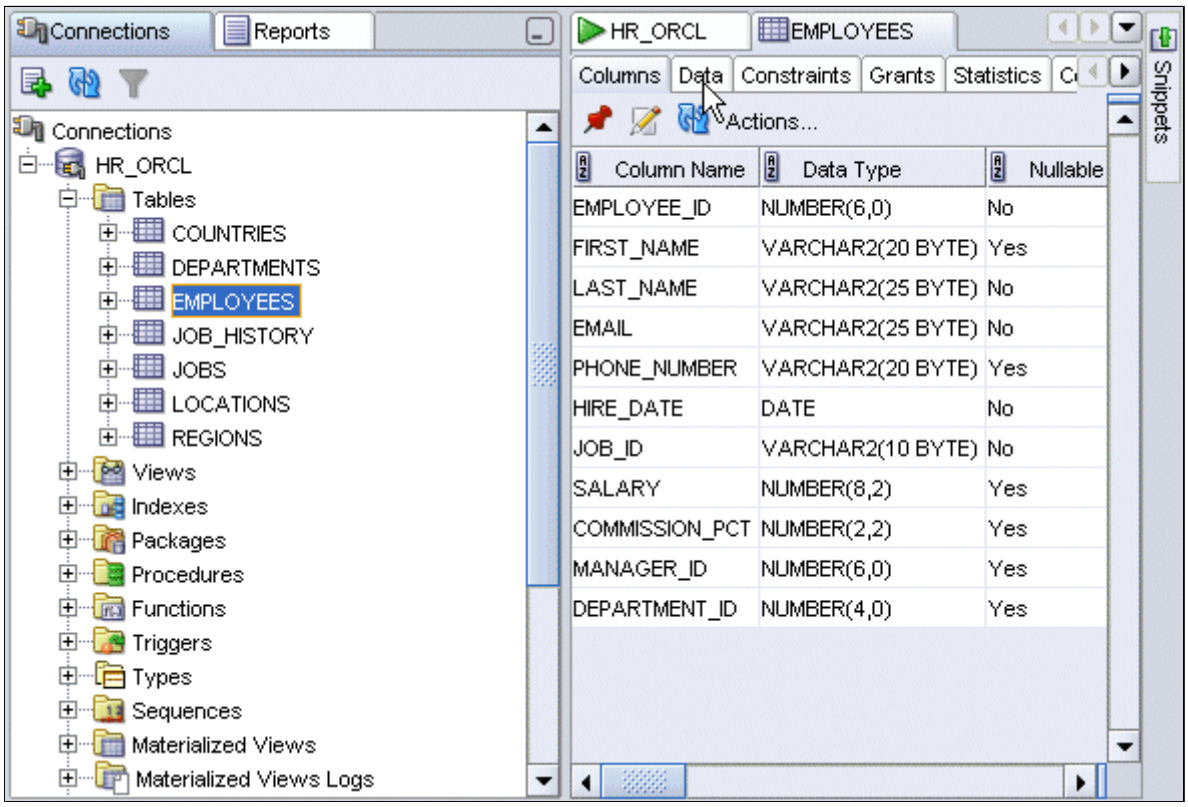
6. Expand **Tables.**



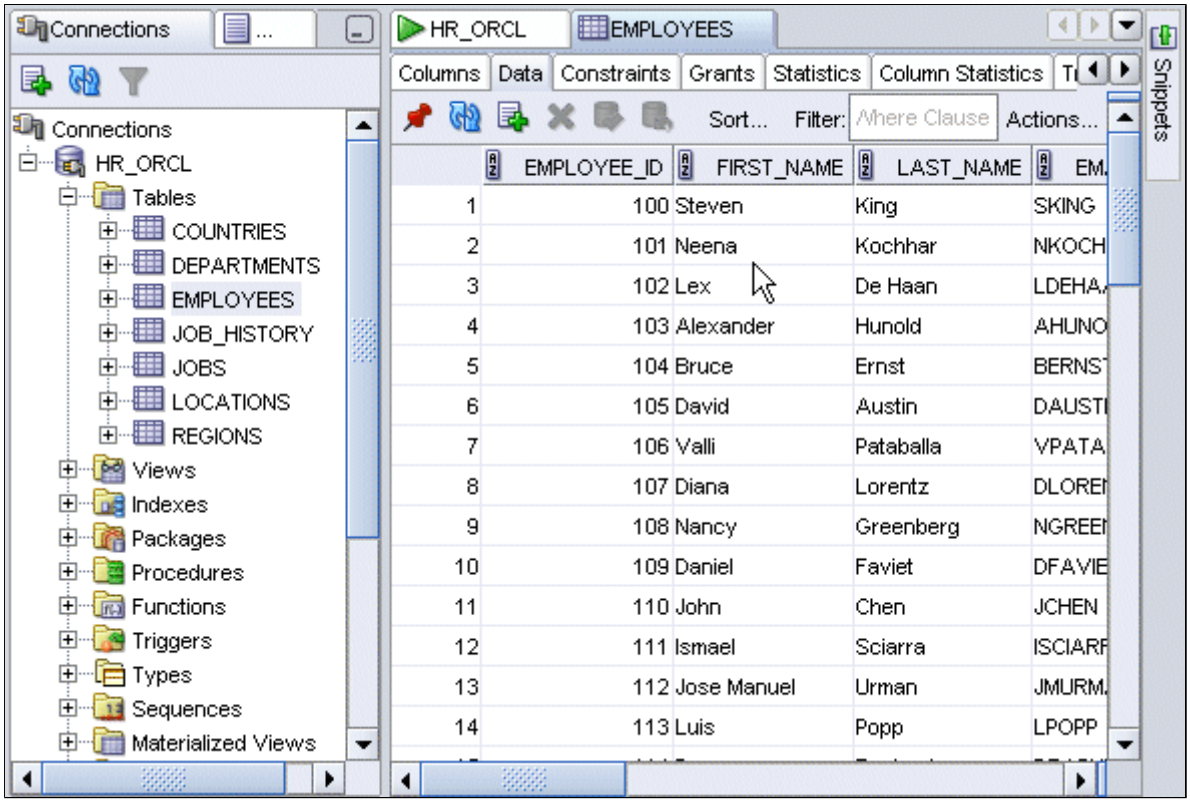
7. Select the Table **EMPLOYEES** to view the table definition.



8. To see the data, click the **Data** tab.



9. The data is shown. In the next topic, you create a new table and populate the table with data.

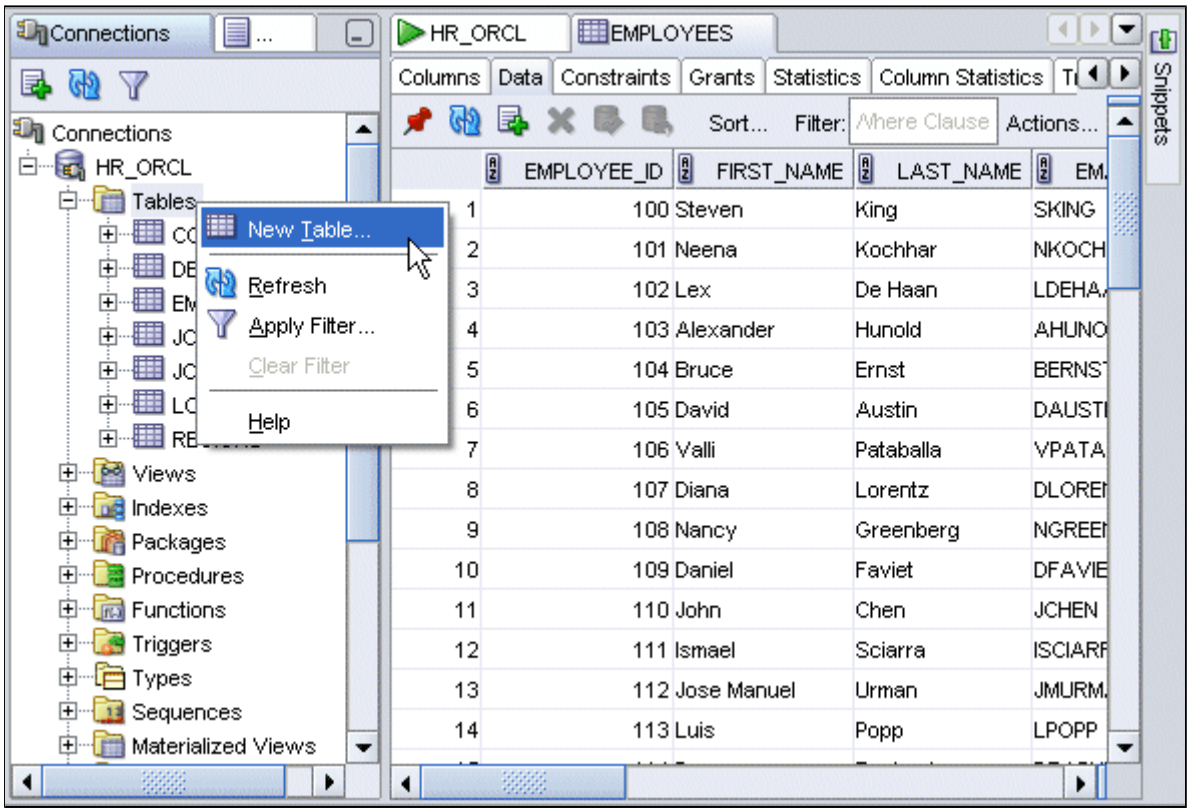


Back to Topic List

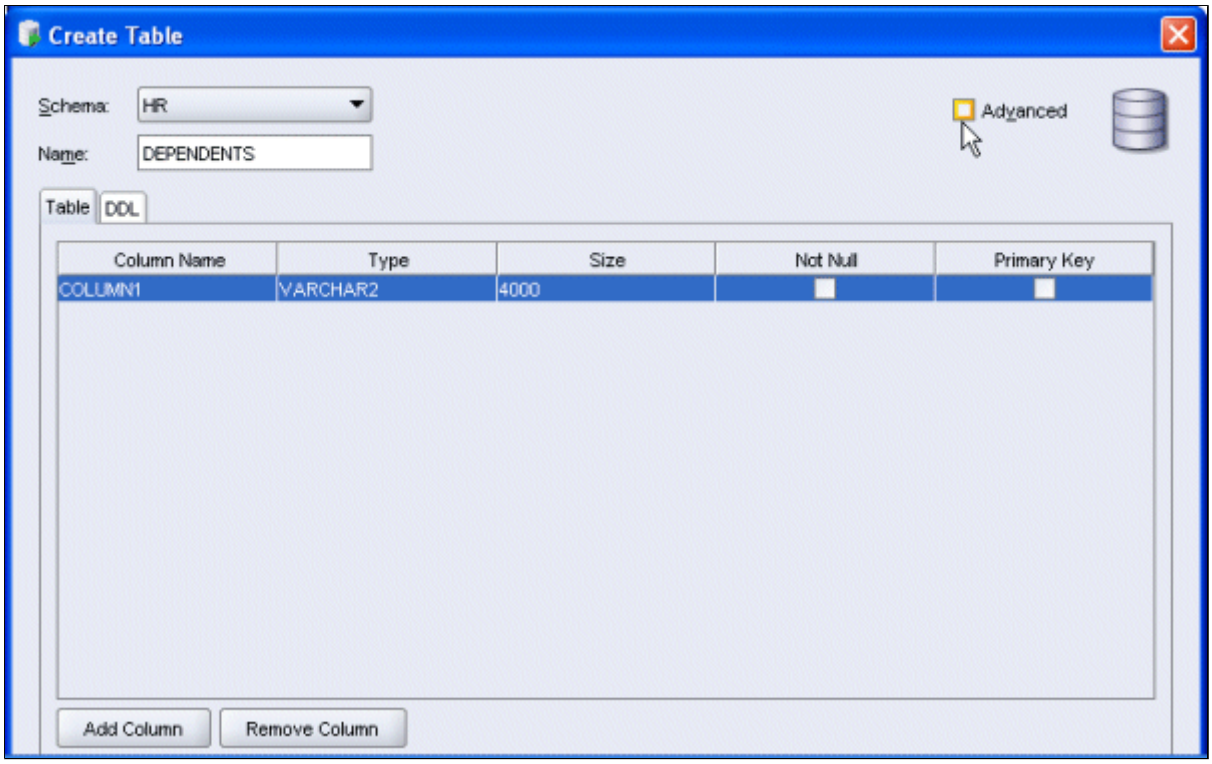
Adding a New Table Using the Create Table Dialog Box

You create a new table called DEPENDENTS which has a foreign key with the EMPLOYEES table. Perform the following steps:

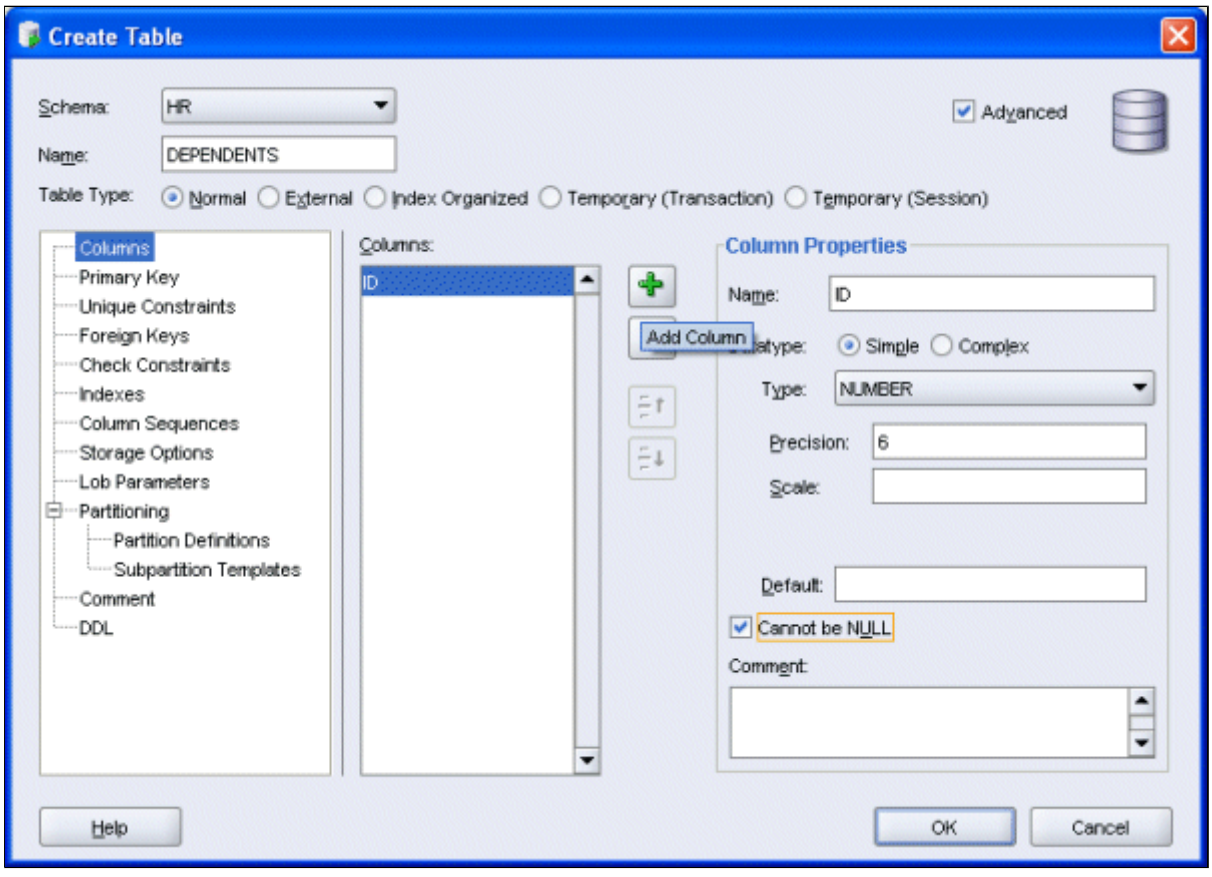
1. Right-click **Tables** and select **New TABLE...**



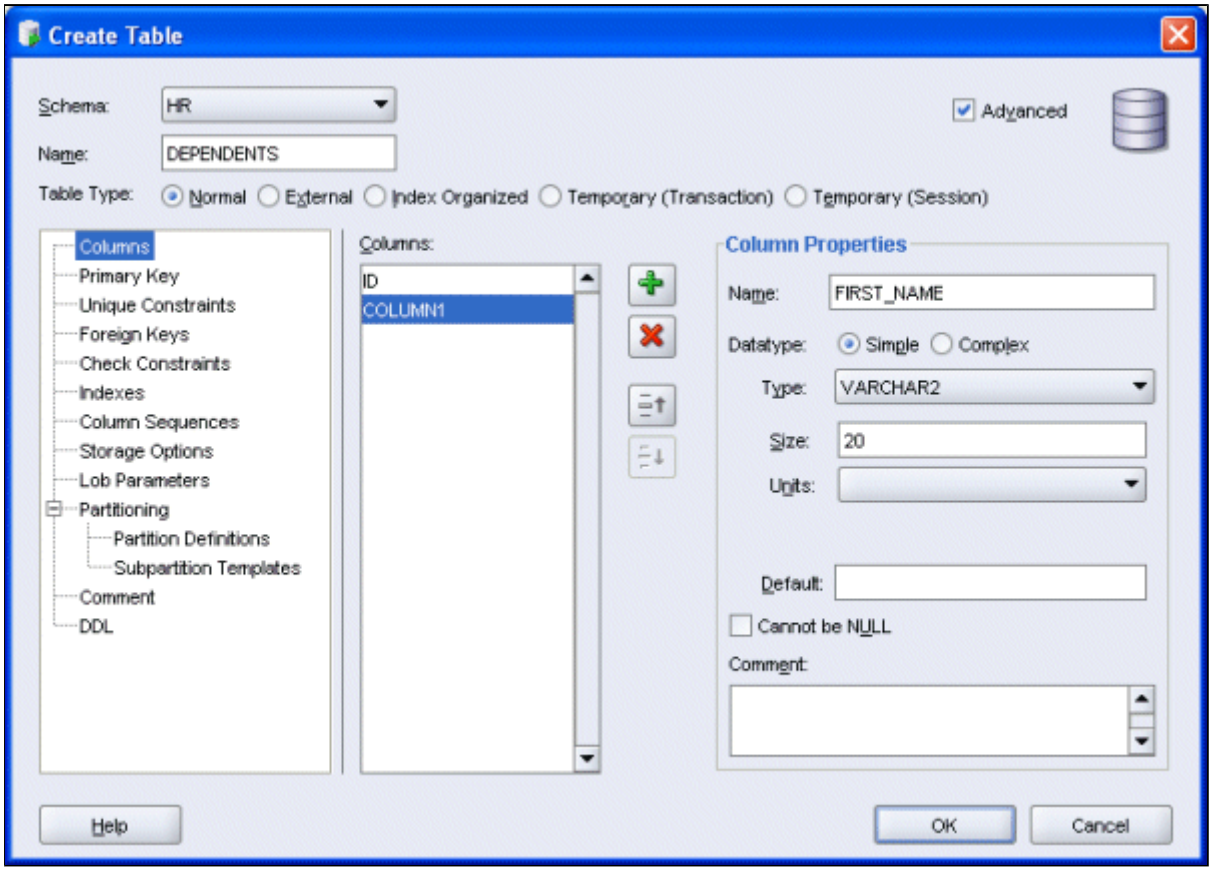
2. Enter **DEPENDENTS** for the Table Name and click the **Advanced** check box.



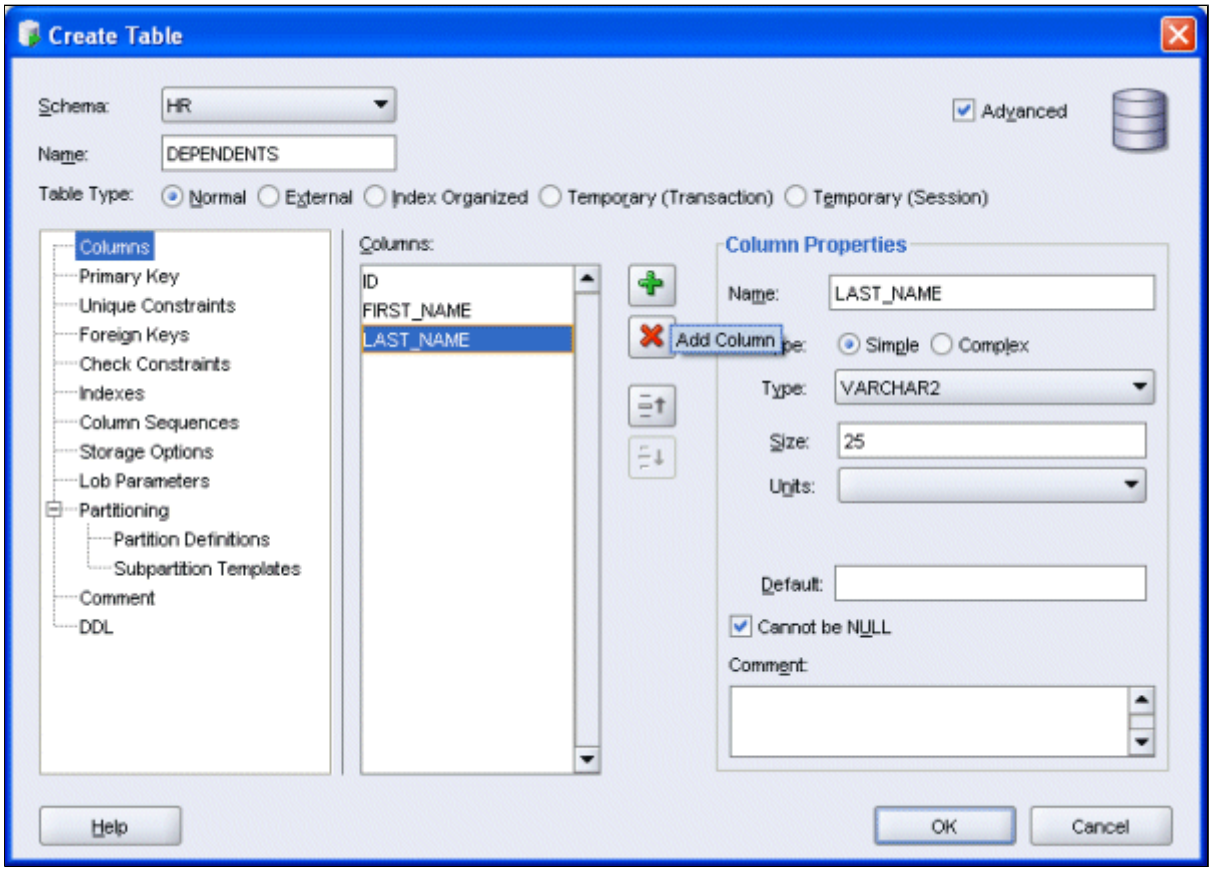
3. Enter **ID** for the Name, select **NUMBER** for the Data type and enter **6** for the Precision. Select the **Cannot be NULL** check box. Then click the **Add Column** icon.



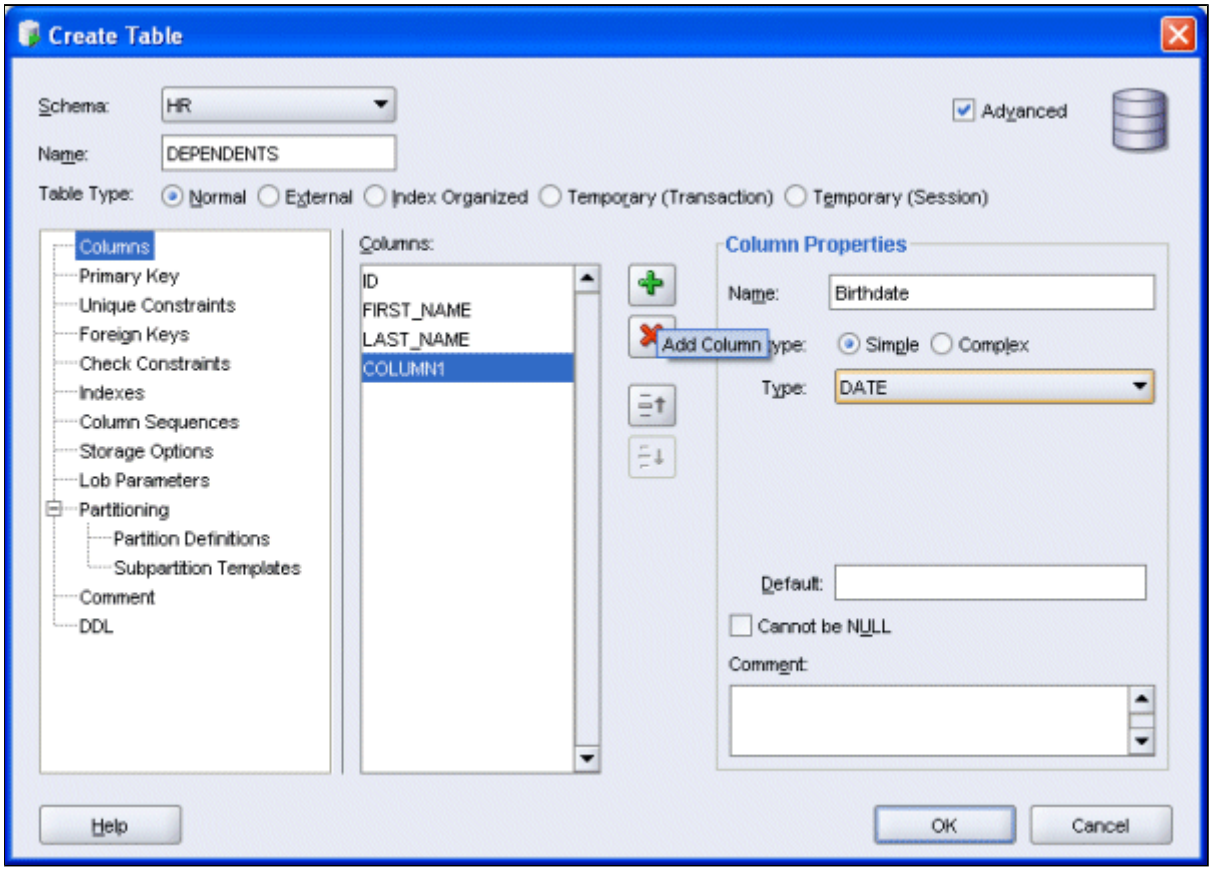
4. Enter **FIRST_NAME** for the Name, and enter **20** for the Size. Then click the **Add Column** icon.



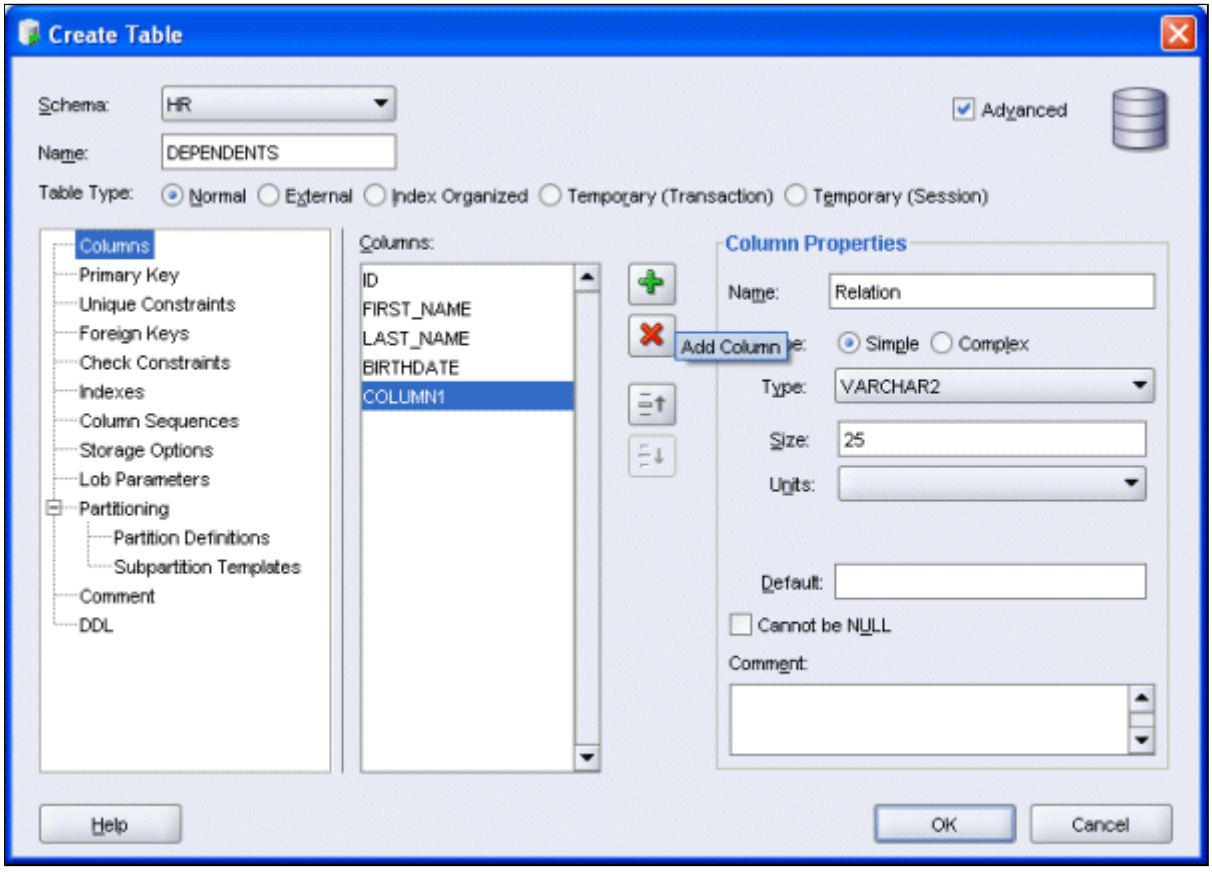
5. Enter **LAST_NAME** for the Name, enter **25** for the Size. Select the **Cannot be NULL** check box. Then click the **Add Column** icon.



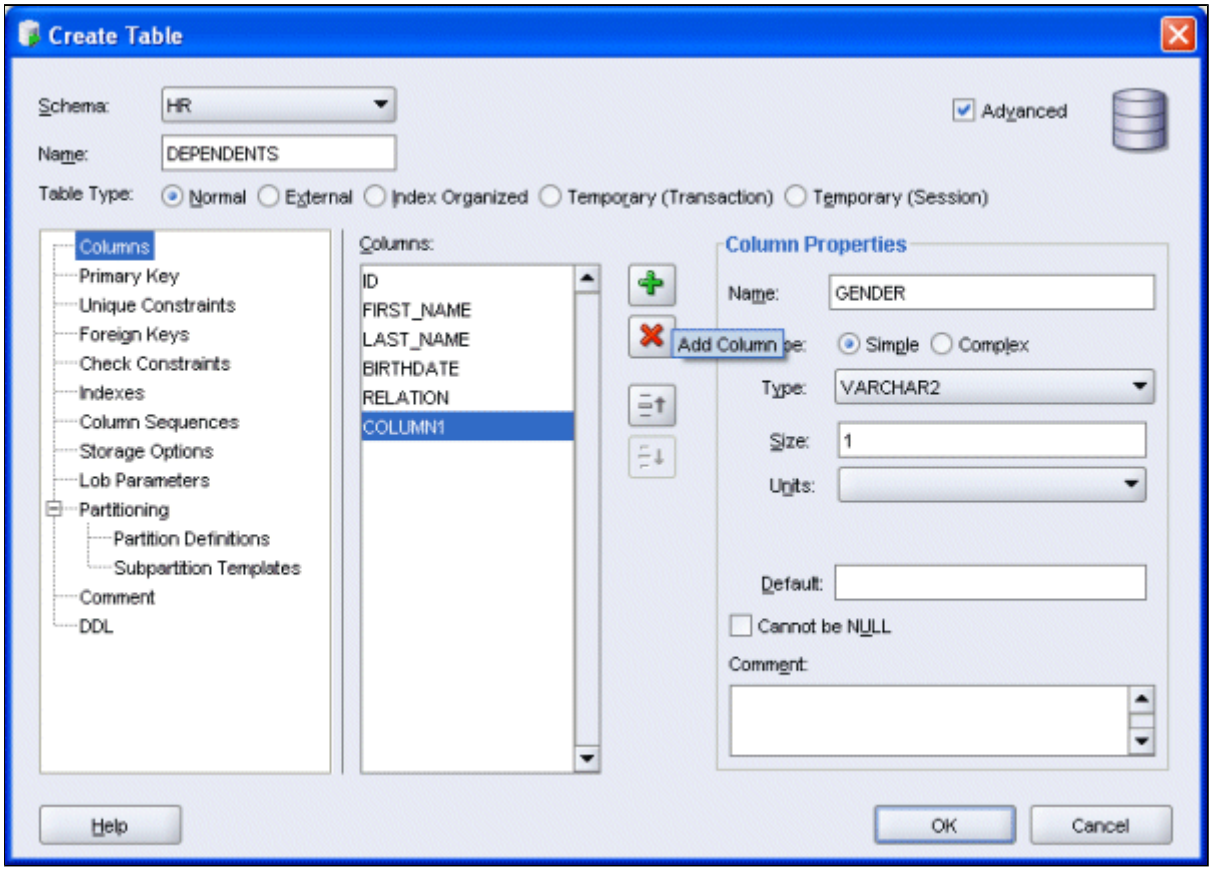
6. Enter **BIRTHDATE** for the Name, select **DATE** for the Data type. Then click the **Add Column** icon.



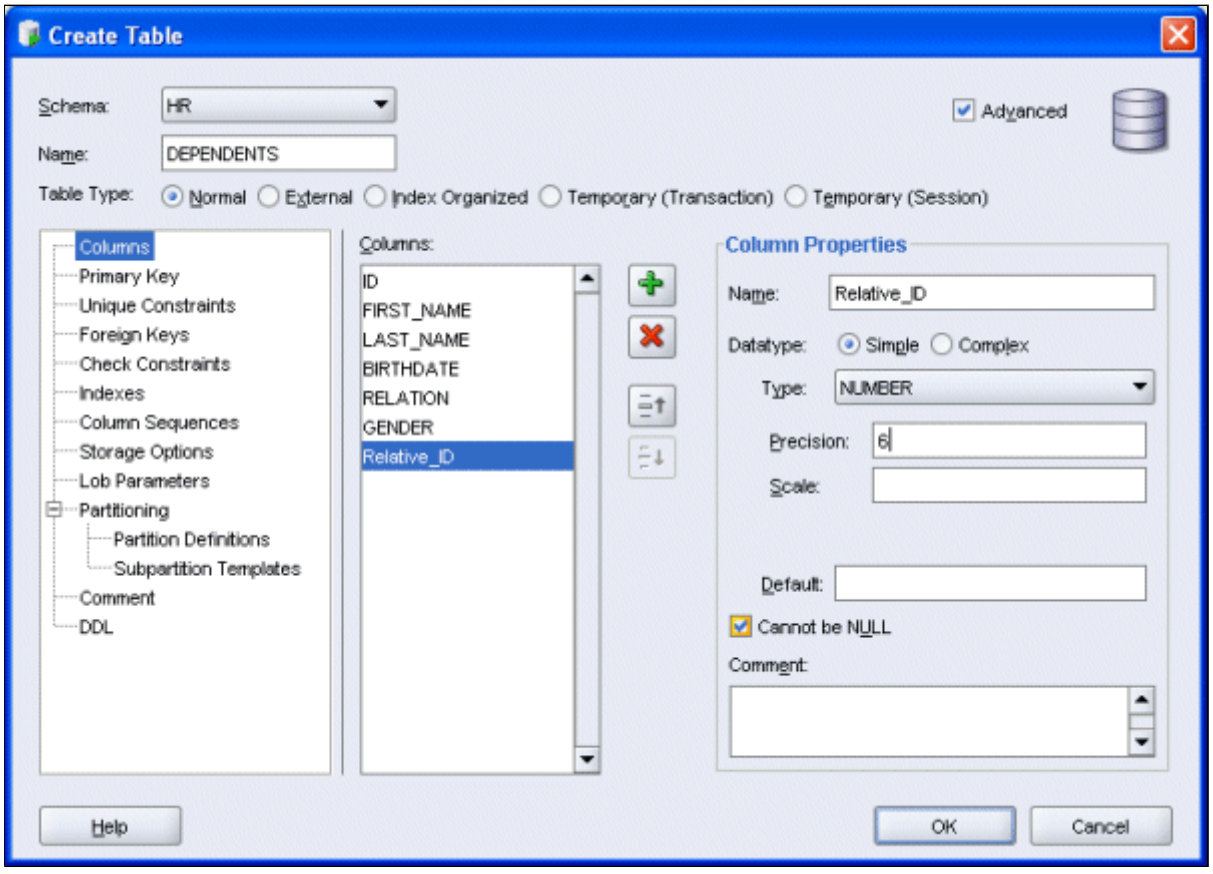
7. Enter **RELATION** for the Name, and enter **25** for the Size. Then click the **Add Column** icon.



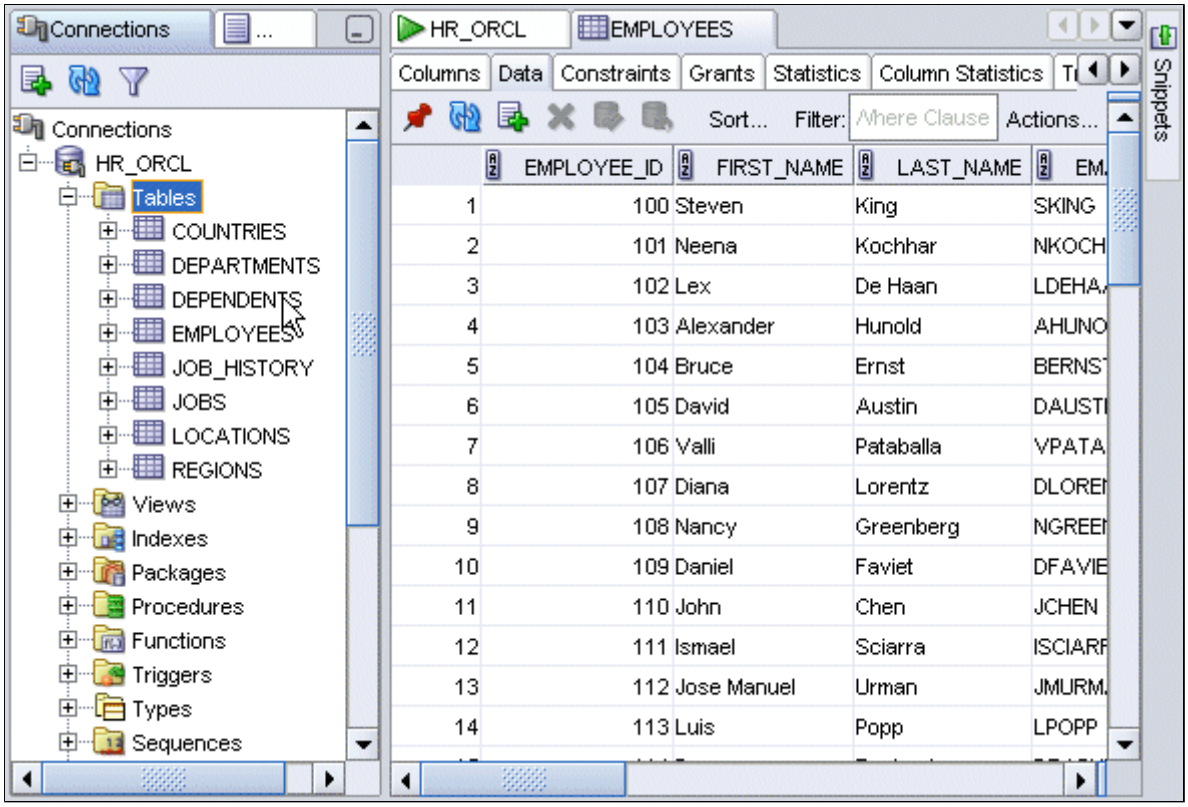
8. Enter **GENDER** for the Name, and enter **1** for the Size. Then click the **Add Column** icon.



9. Enter **RELATIVE_ID** for the Name, select **NUMBER** for the Data type, and enter **6** for the Precision. Select the **Cannot be NULL** check box. Then click **OK** to create the table.



10. Your new table appears in the list of tables. Select **DEPENDENTS** from the list.

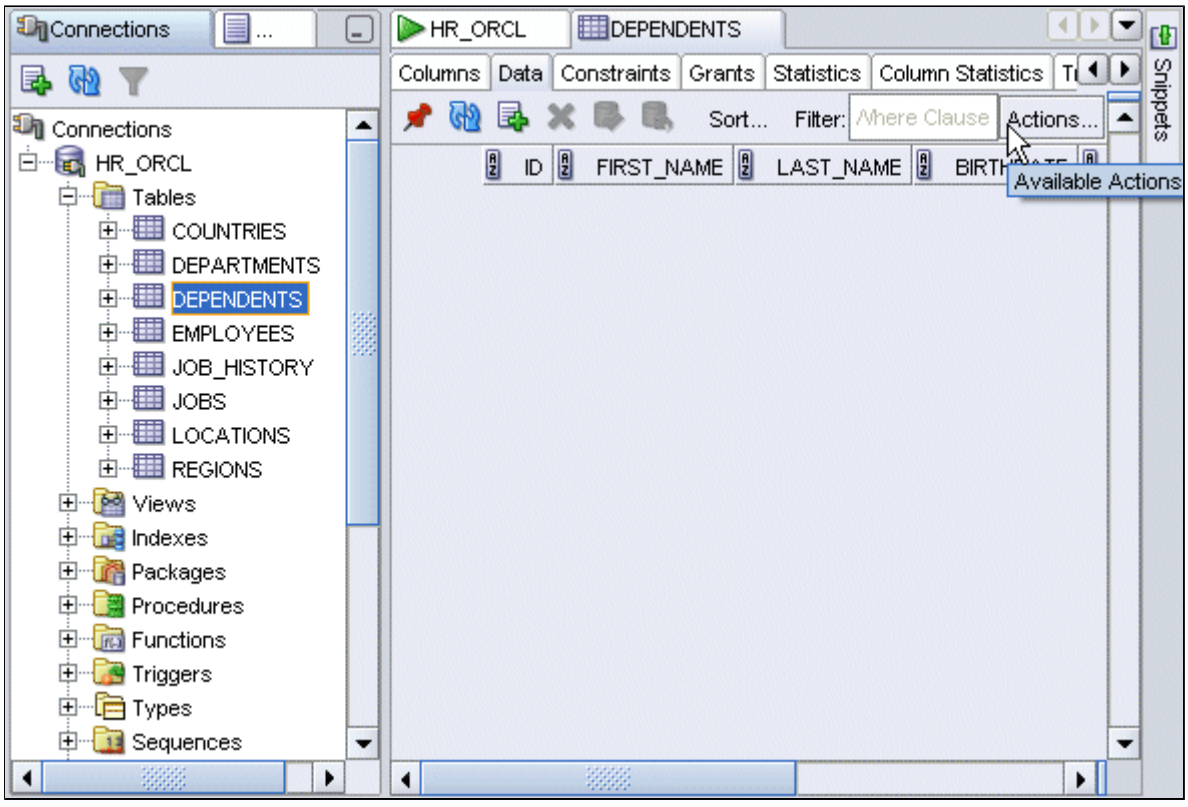


Back to Topic List

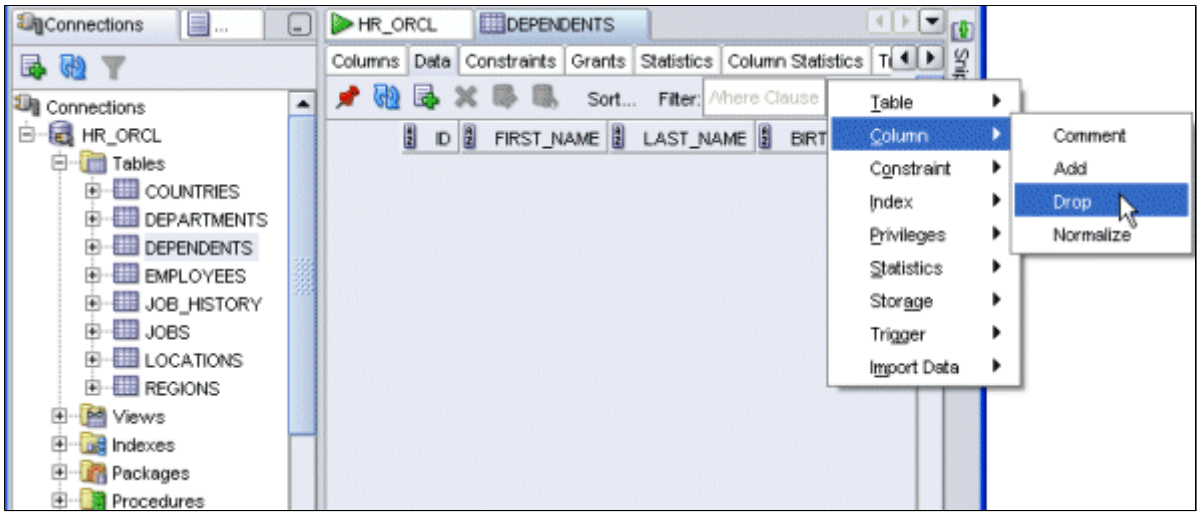
Changing a Table Definition

Oracle SQL Developer makes it very easy to make changes to database objects. In this topic, you delete a column in the DEPENDENTS table you just created. Perform the following steps:

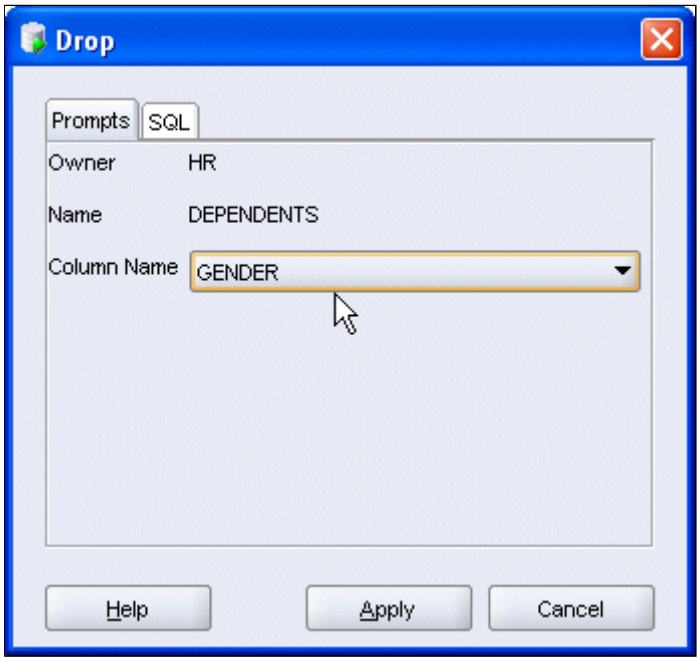
1. Click the **Actions...** button.



2. Select **Column** then **Drop**.

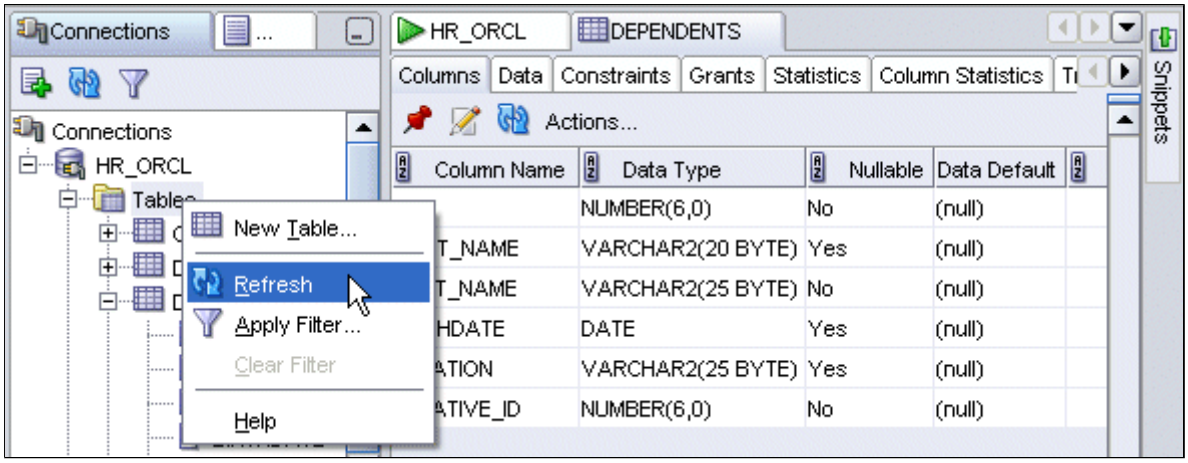


3. Select the Column Name **GENDER** and click **Apply**.

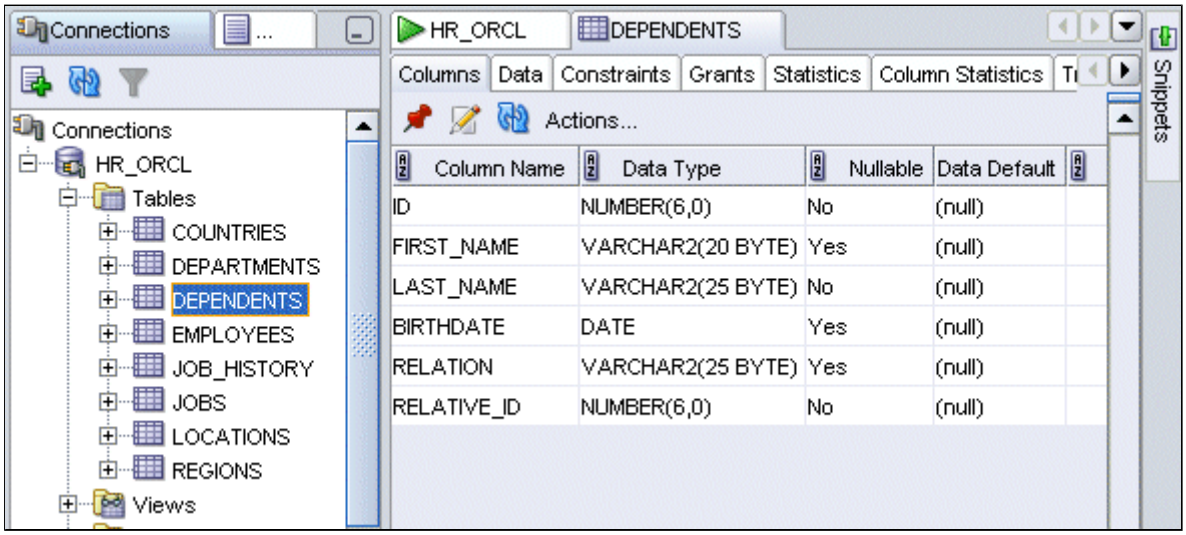


4. The column has been dropped. Click **OK**.

5. Right-click the **tables** node for the **HR_ORCL** Database Connection and click **Refresh** to refresh the Database Connection.



6. Expand **HR_ORCL > Tables**, and select the DEPENDENTS table.

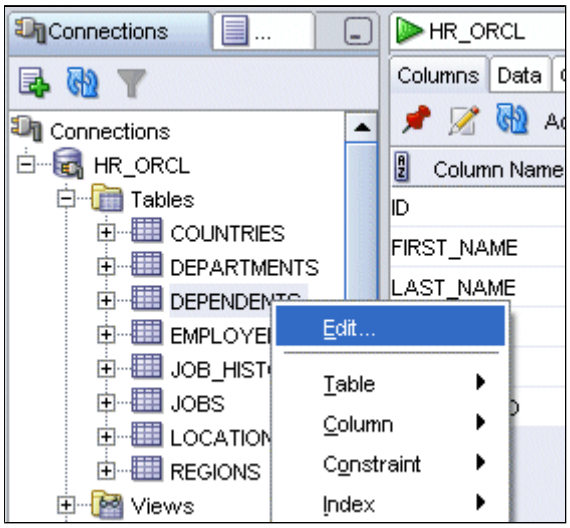


Back to Topic List

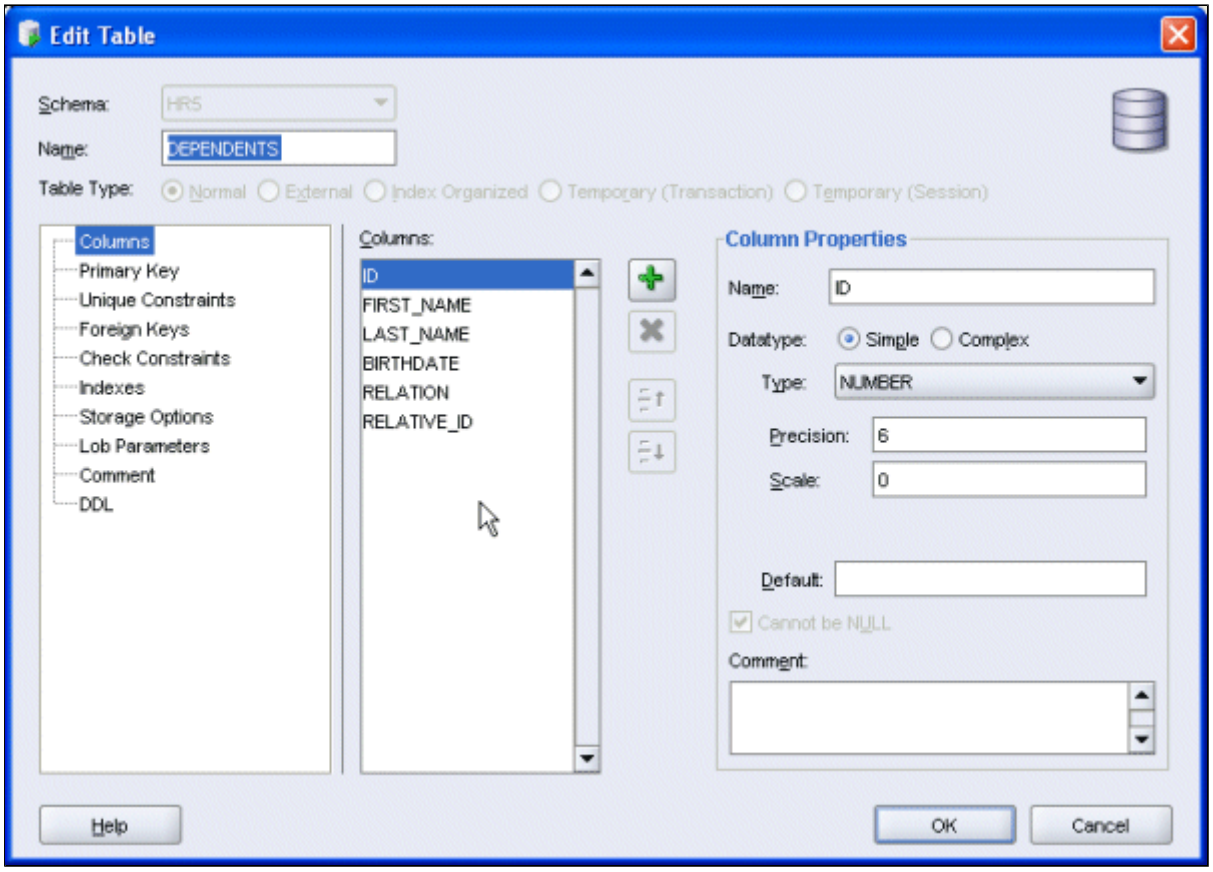
Adding Table Constraints

In this topic, you create the Primary and Foreign Key Constraints for the DEPENDENTS table. Perform the following steps:

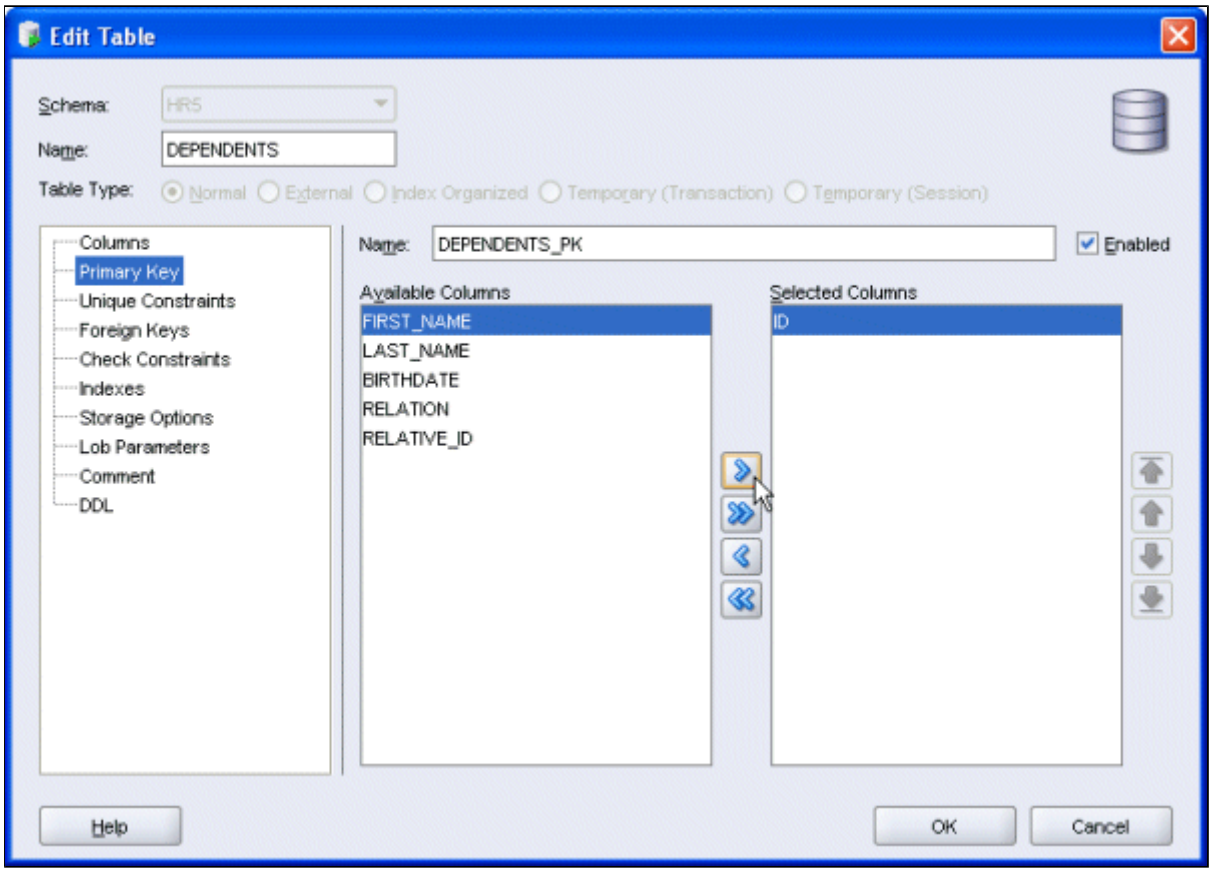
1. Right-click **DEPENDENTS** table and select **Edit**...



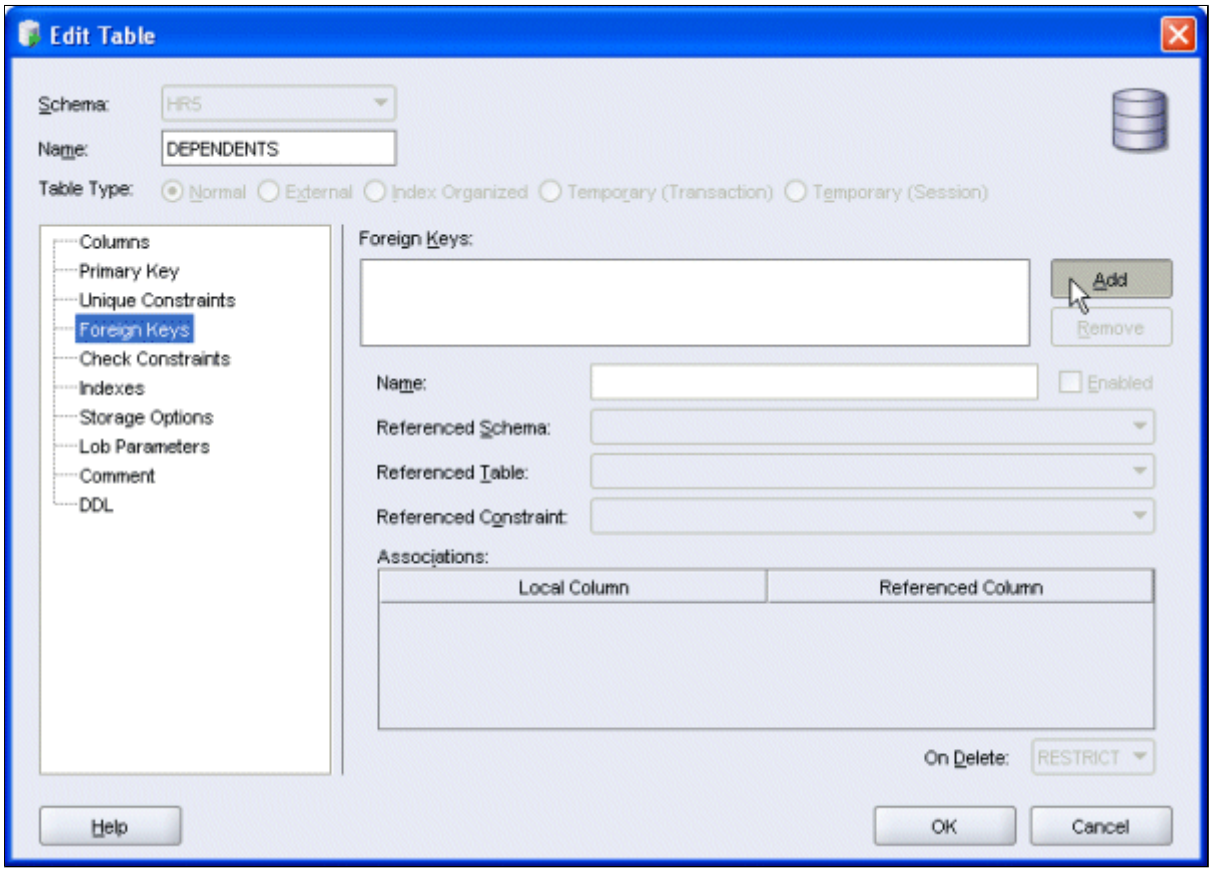
2. Click the **Primary Key** node in the tree.



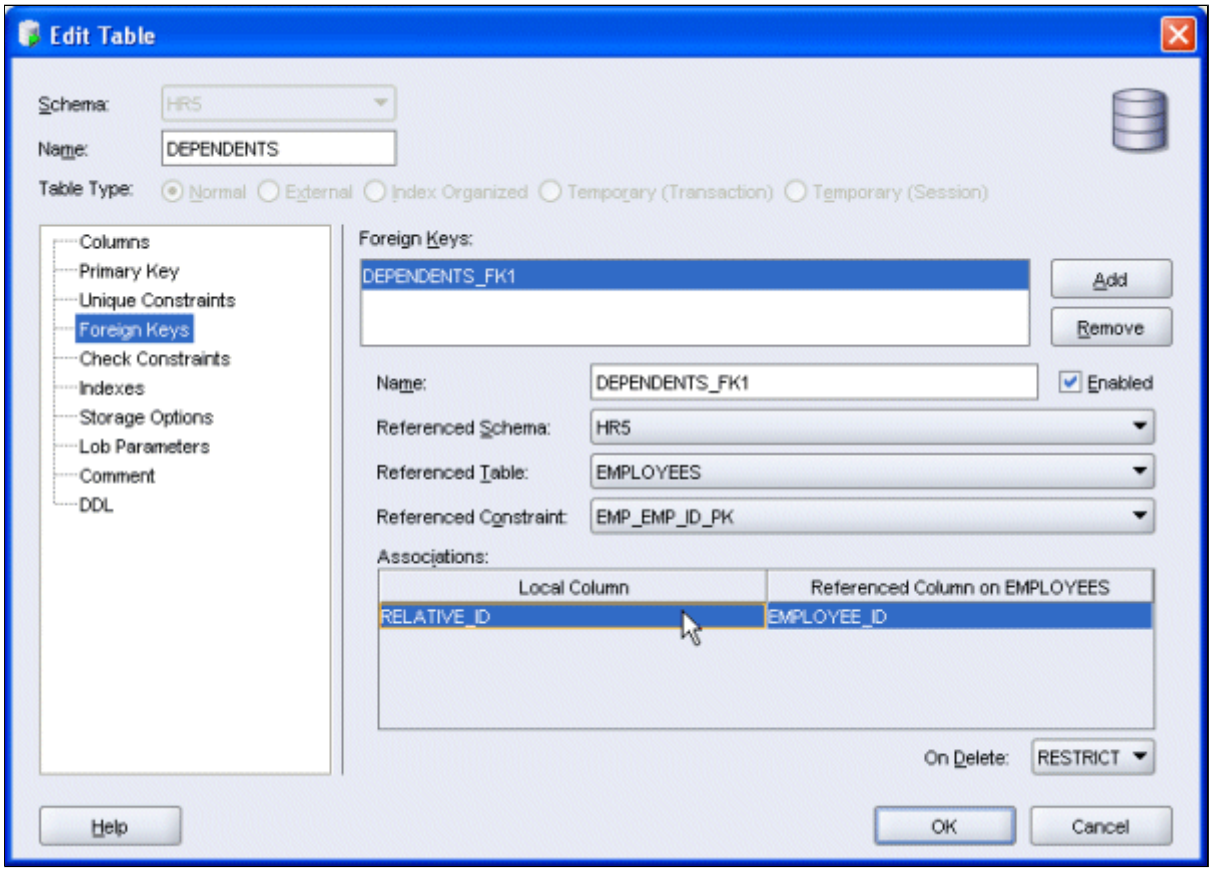
3. Select the **ID** column and click **>** to shuttle the value to the Selected Columns window. Then click the **Foreign Key** node in the tree.



4. Click **Add**.



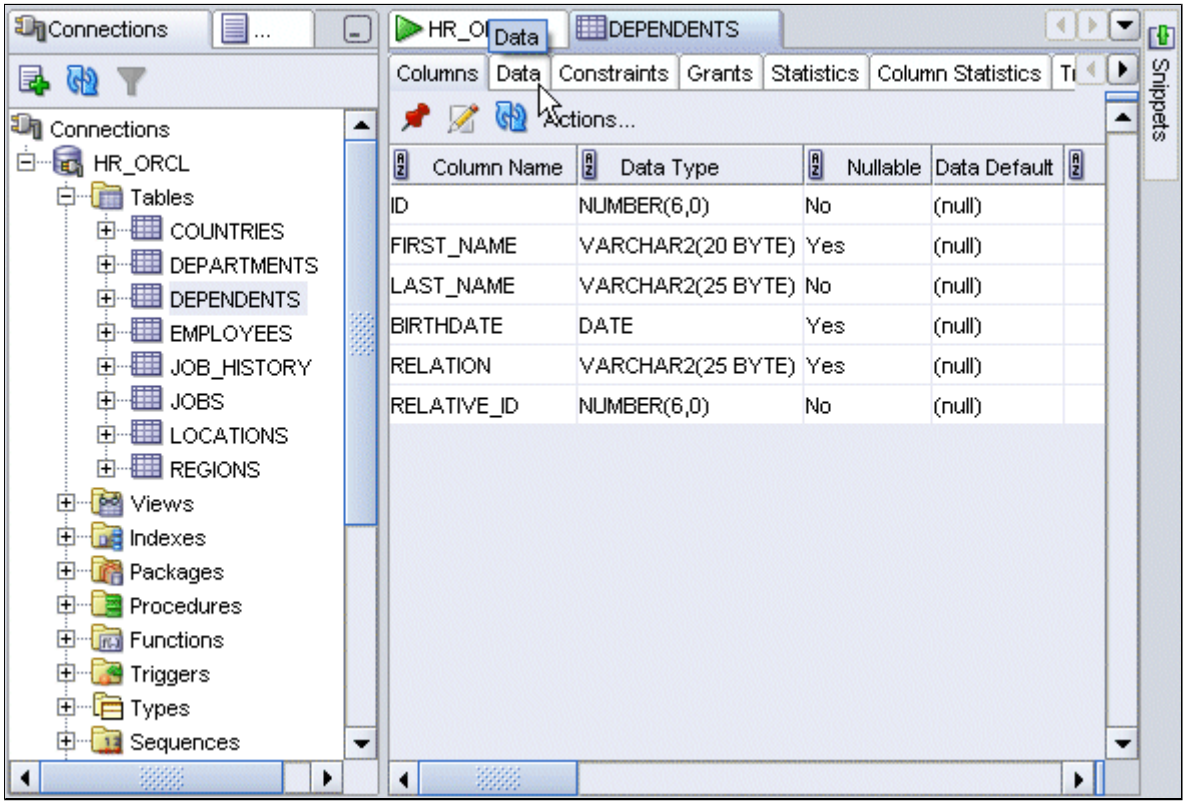
5. Select **EMPLOYEES** for the Referenced Table and select **RELATIVE_ID** for the Local Column and click **OK**.



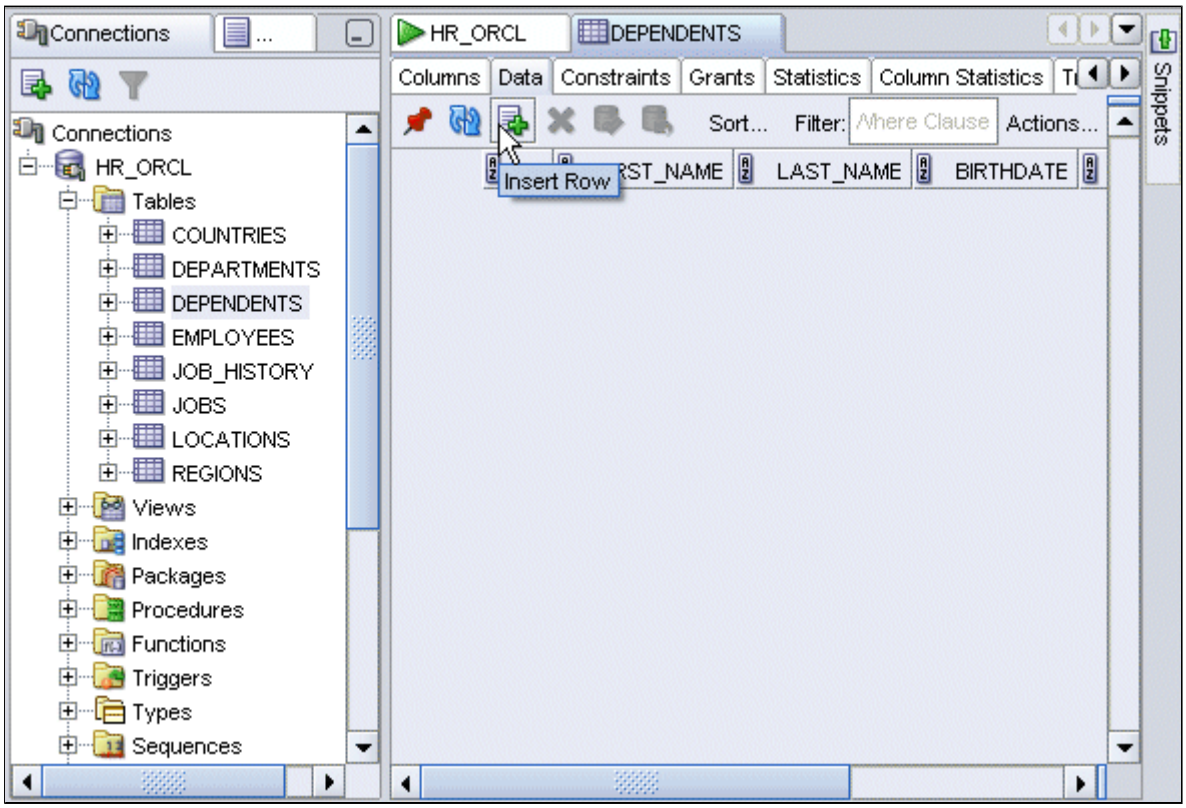
Adding Data to a Table

You can add data to a table by performing the following steps:

1. Click the **Data** tab.

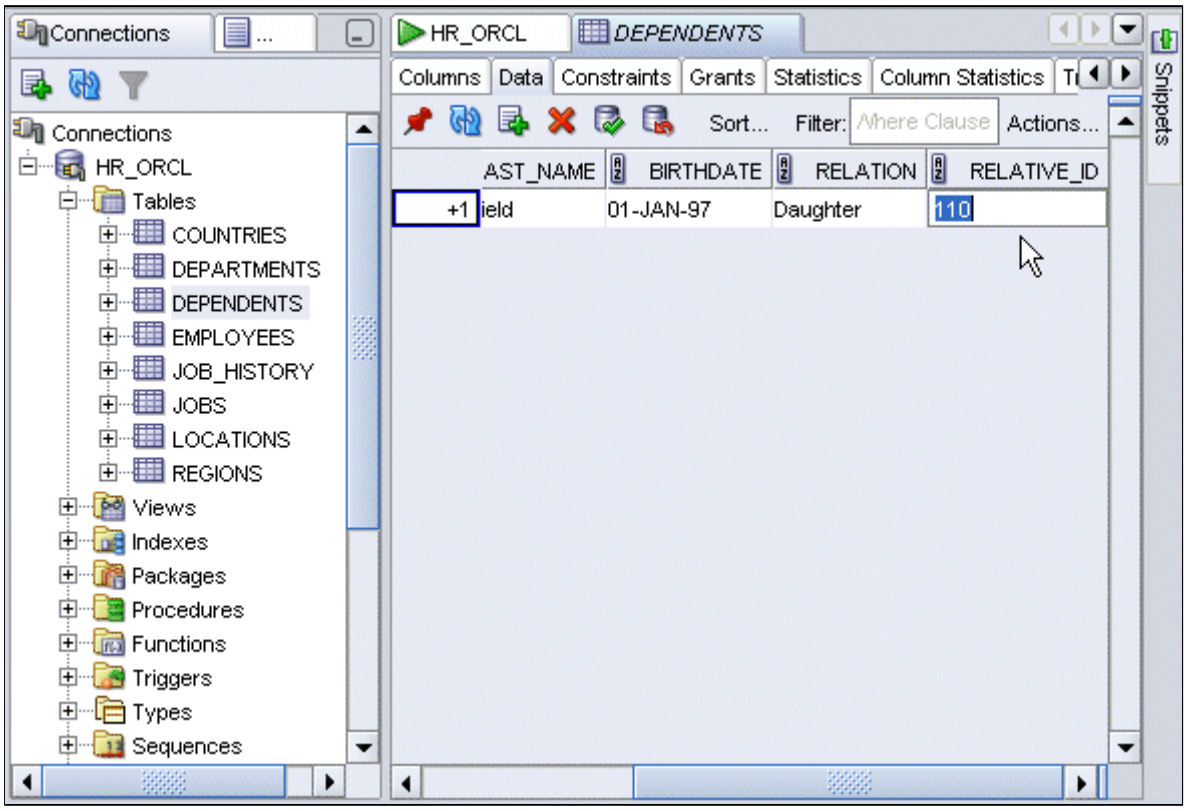


2. Then click the **Insert Row** icon.

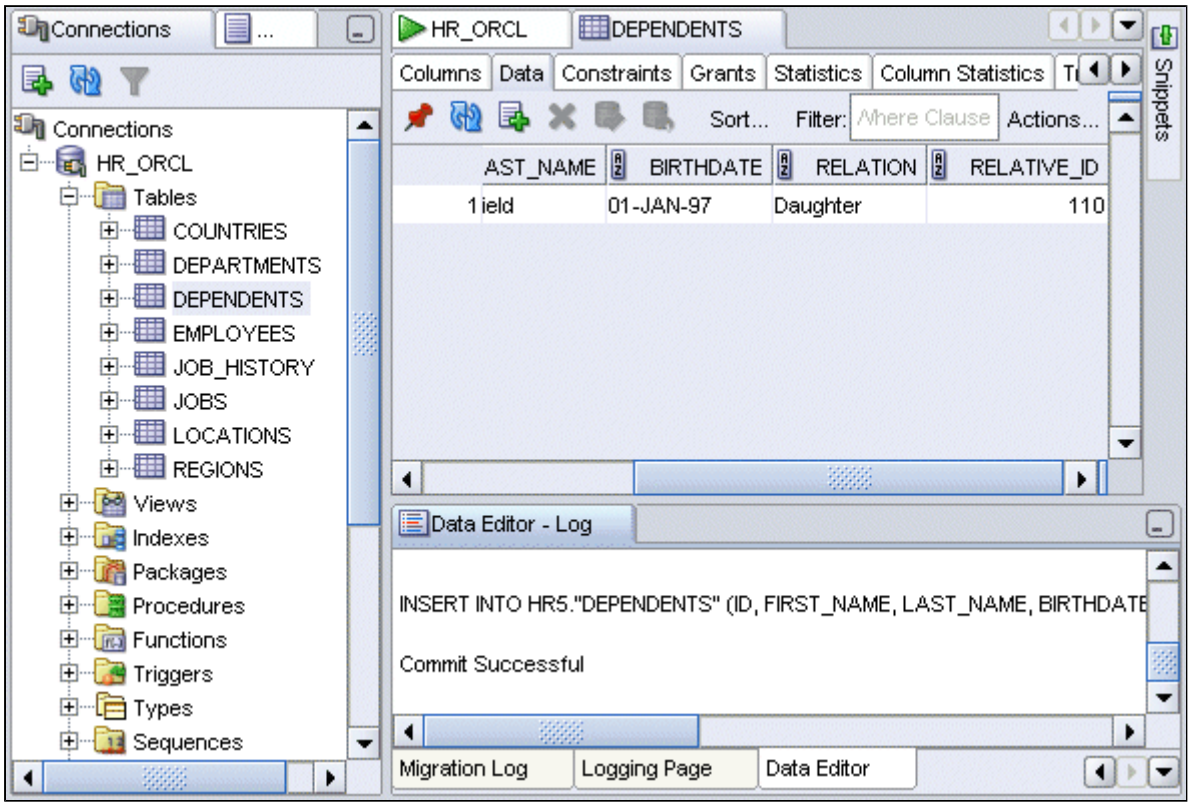


3. Enter the following data and click the **Commit Changes** icon.

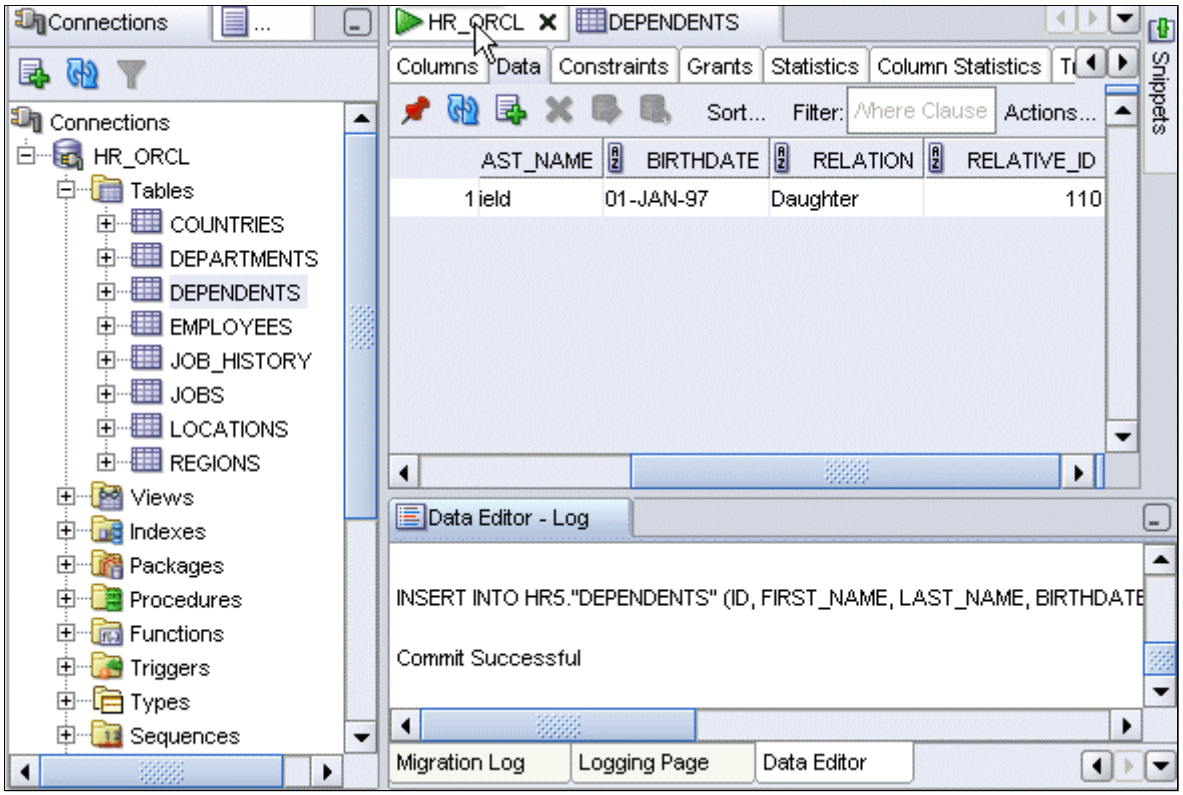
ID	**209**
FIRST_NAME	**Sue**
LAST_NAME	**Littlefield**
BIRTHDATE	**01-JAN-97**
RELATION	**Daughter**
RELATIVE_ID	**110**



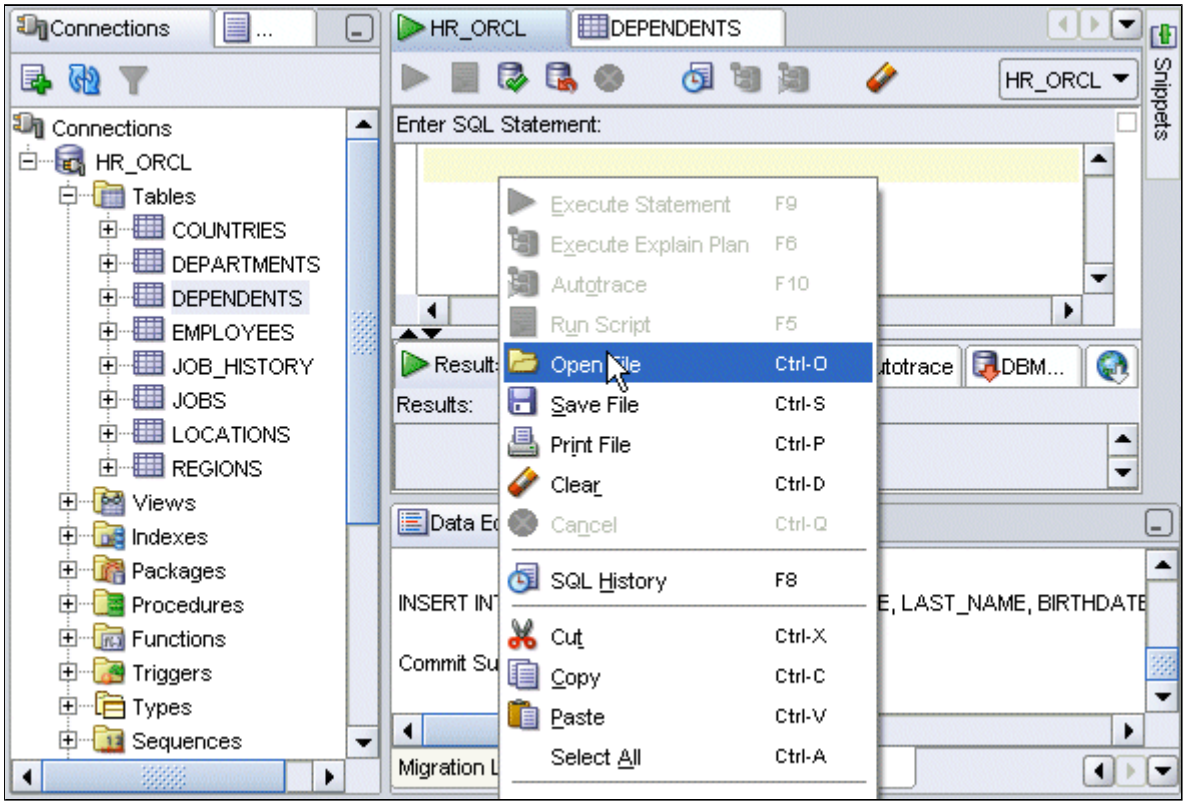
4. The row you just added was committed to the database.



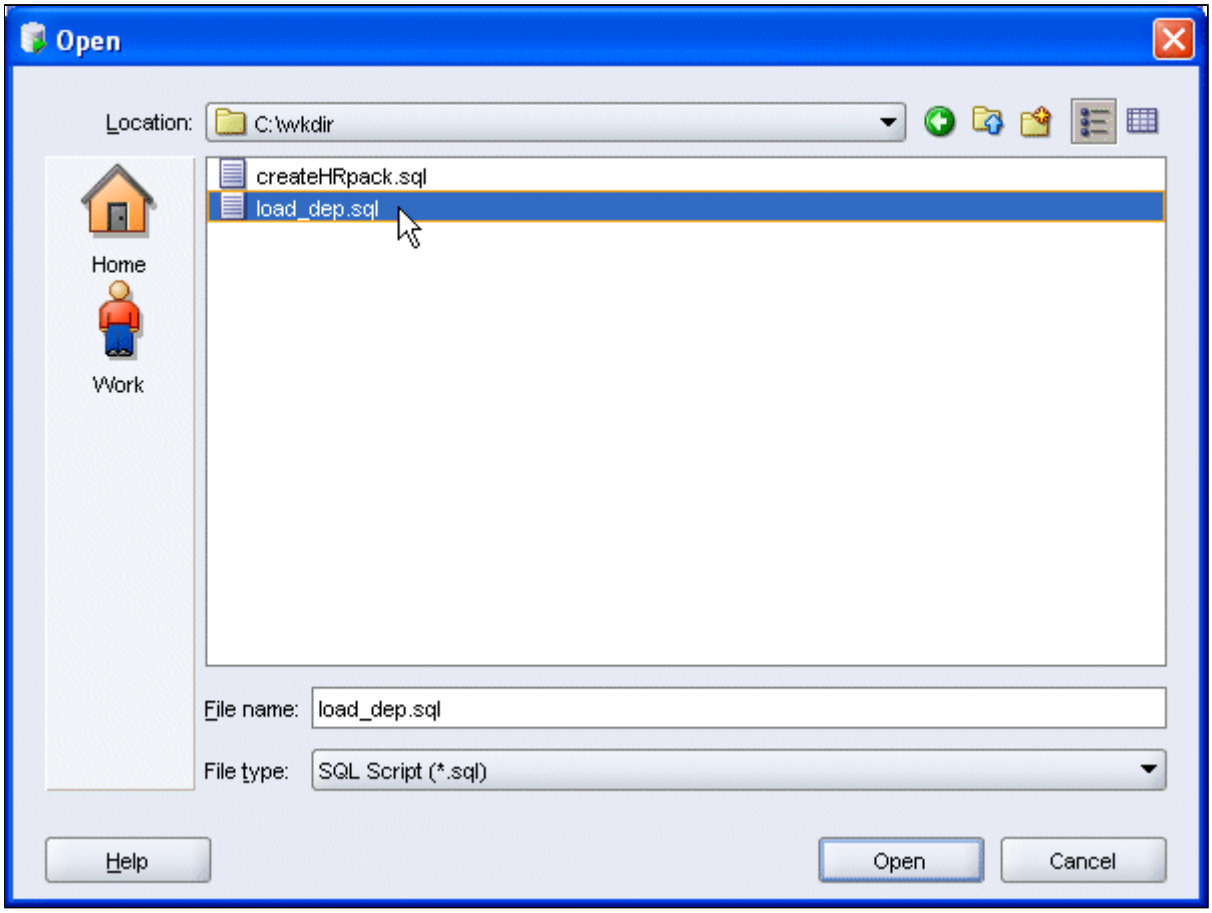
5. You can also load multiple rows at one time using a script. Click the SQL Worksheet **HR_ORCL** tab.



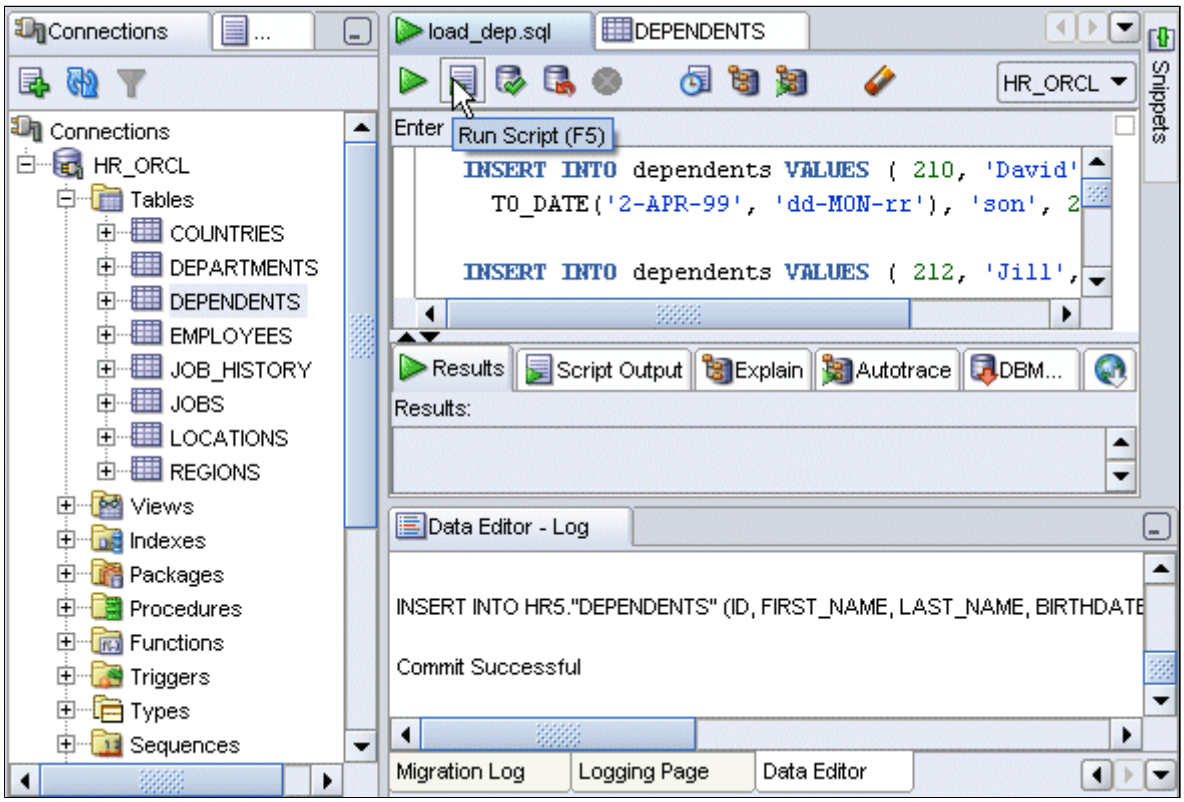
6. Right-click in the SQL Statement area and select **Open File**.



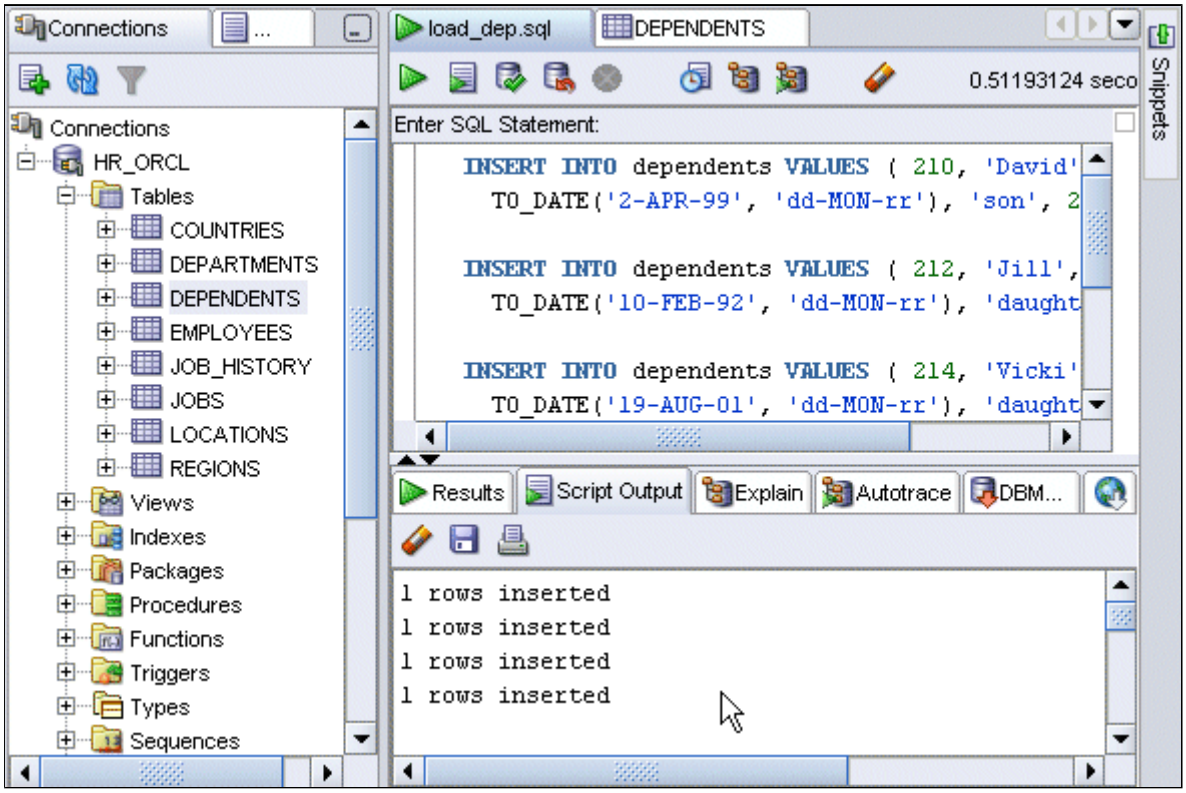
7. Navigate to the **\SQLDev_HOS\GettingStarted\files** directory and select the **load_dep.sql** file and click **Open**.



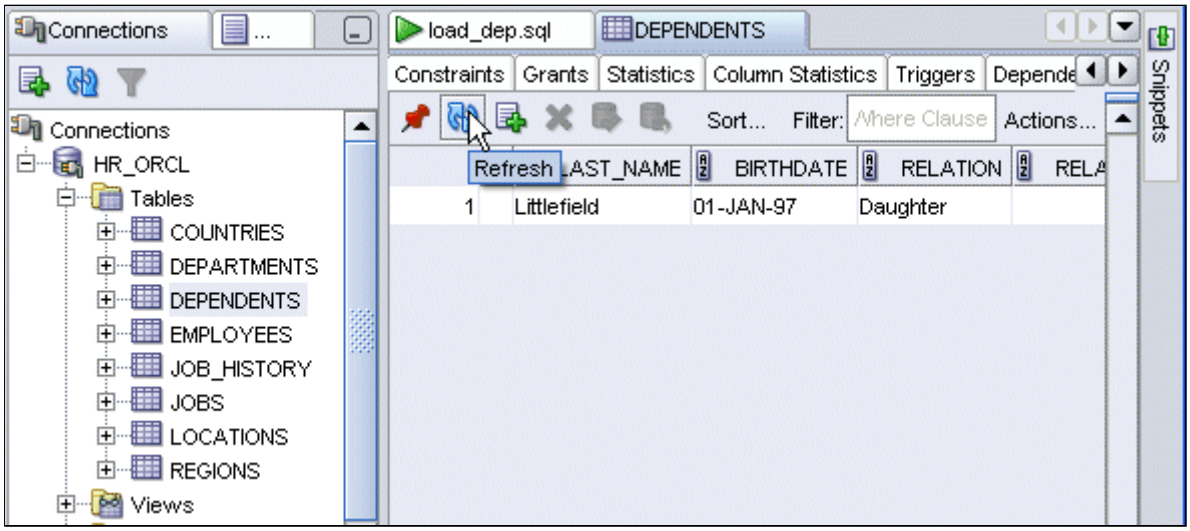
8. The SQL from the script is shown. Click the **Run Script** icon.



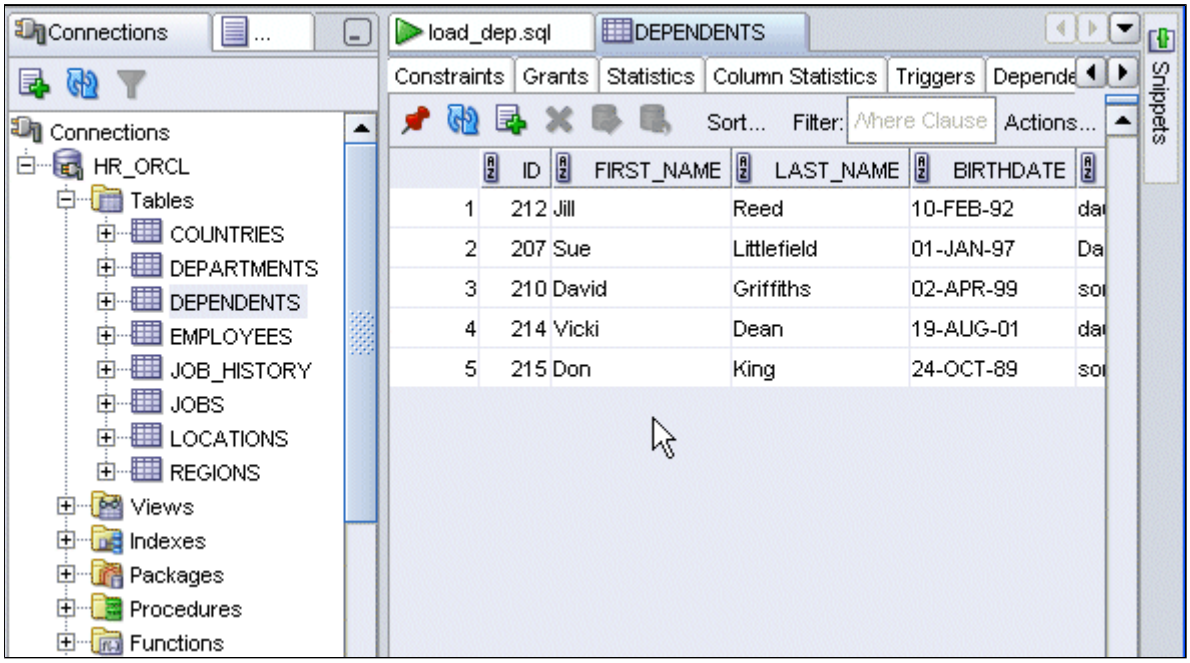
9. The data was inserted. To view the data, click the **DEPENDENTS** tab.



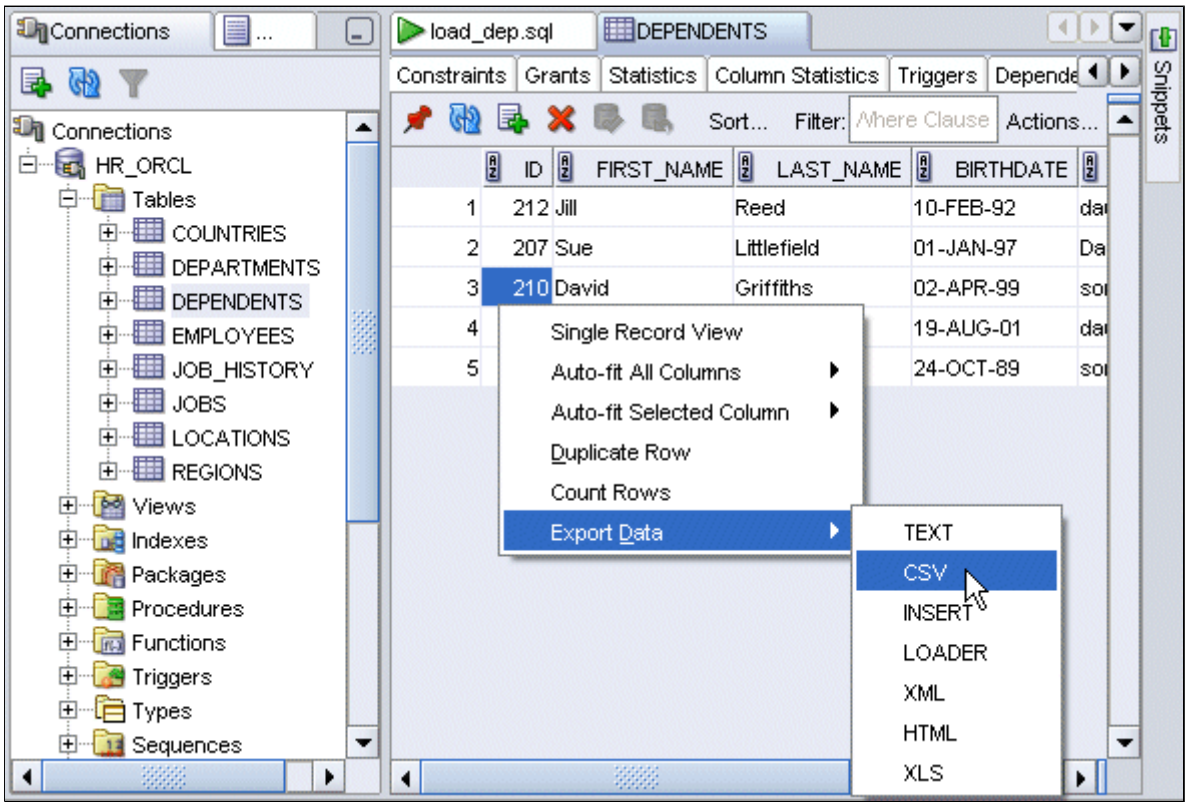
10. Click **Refresh** to show all the data.



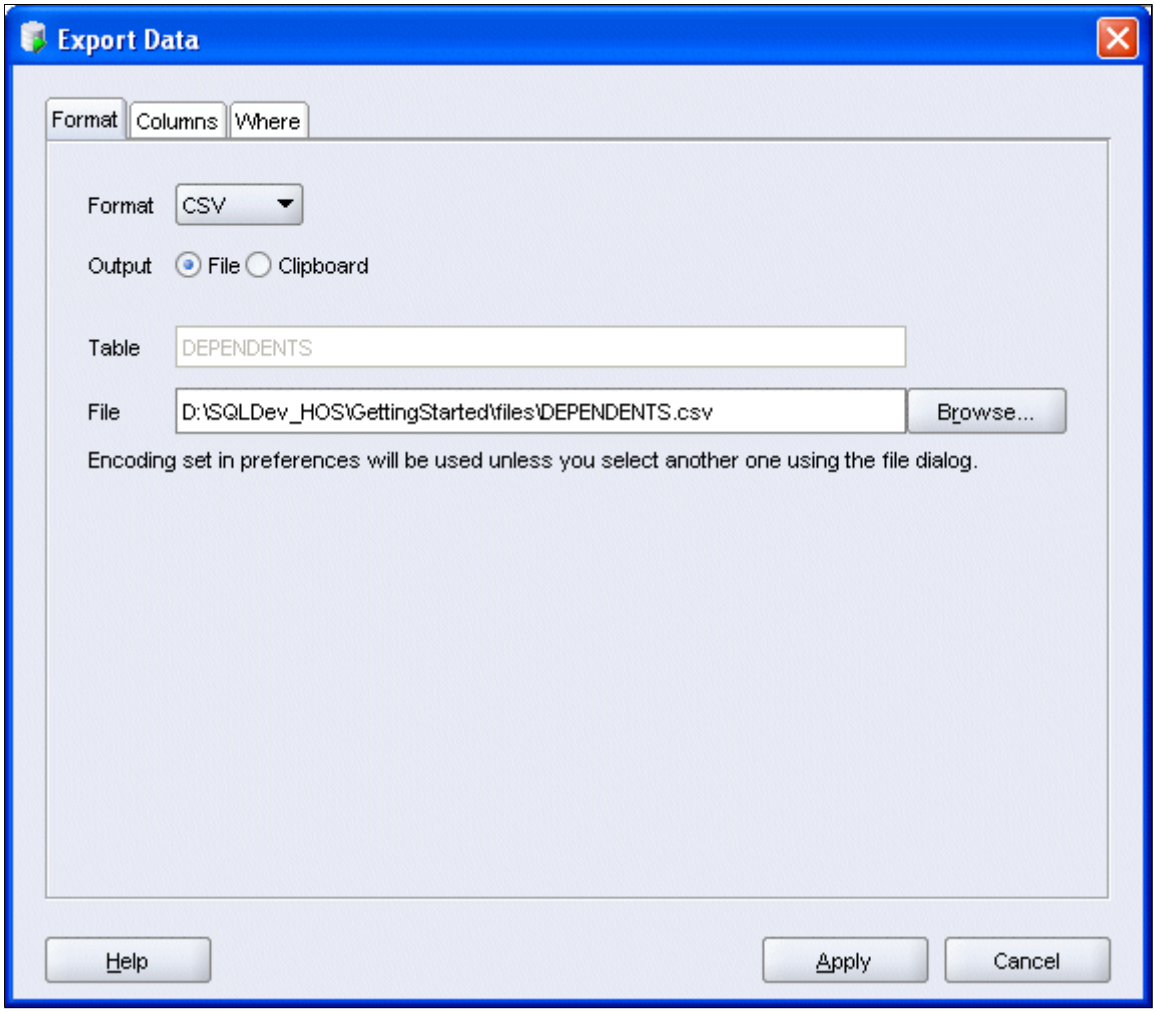
11. All the data is displayed



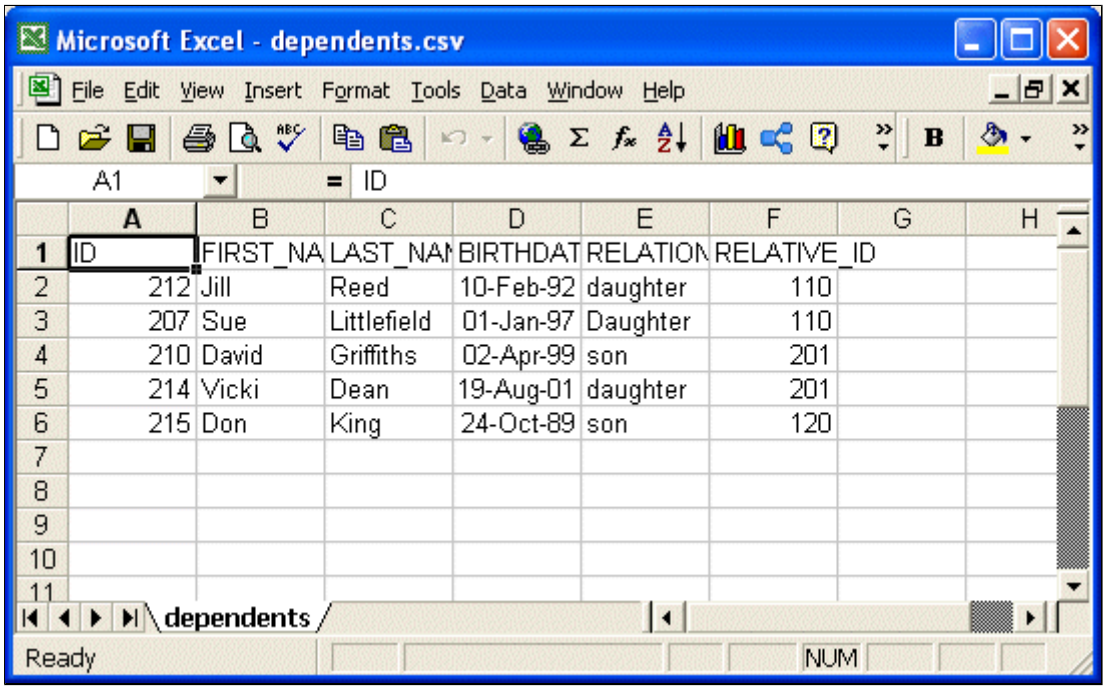
12. You can export the data so it can be used in another tool, for example, Excel. Right-click on one of the values in any column, select **Export** and then one of the file types.



13. Specify the directory and name of the file and click **Apply**.



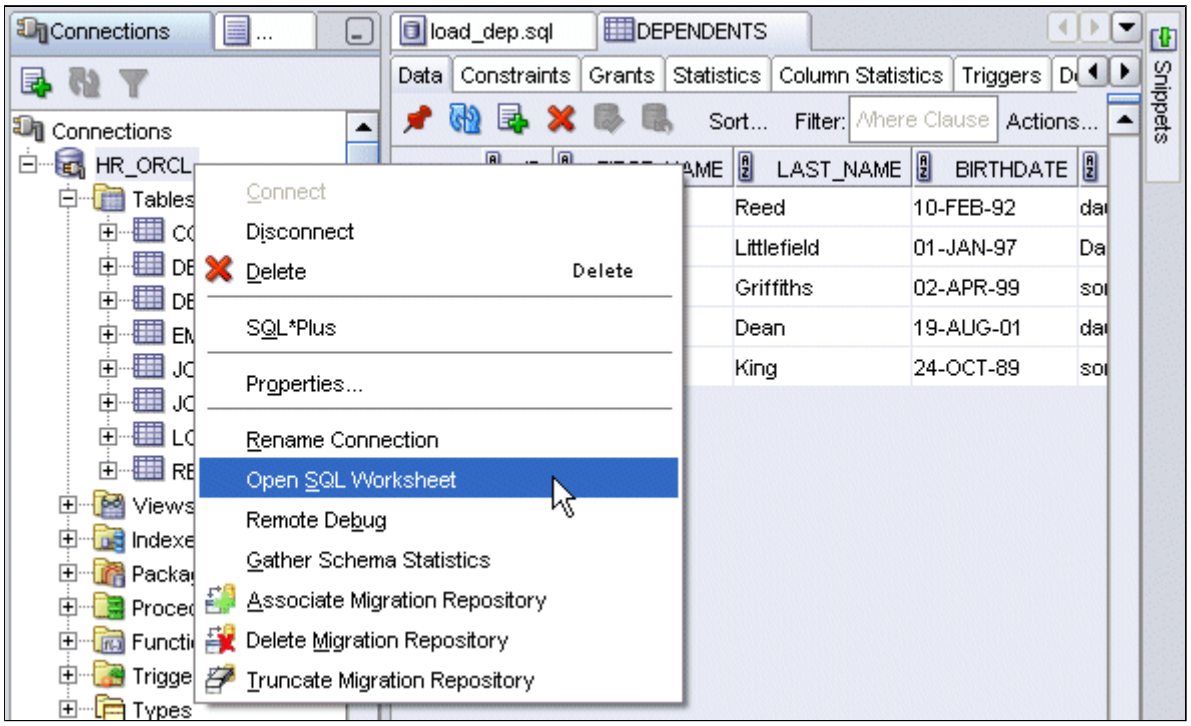
14. If you review the DEPENDENTS.CSV file, you see the following.



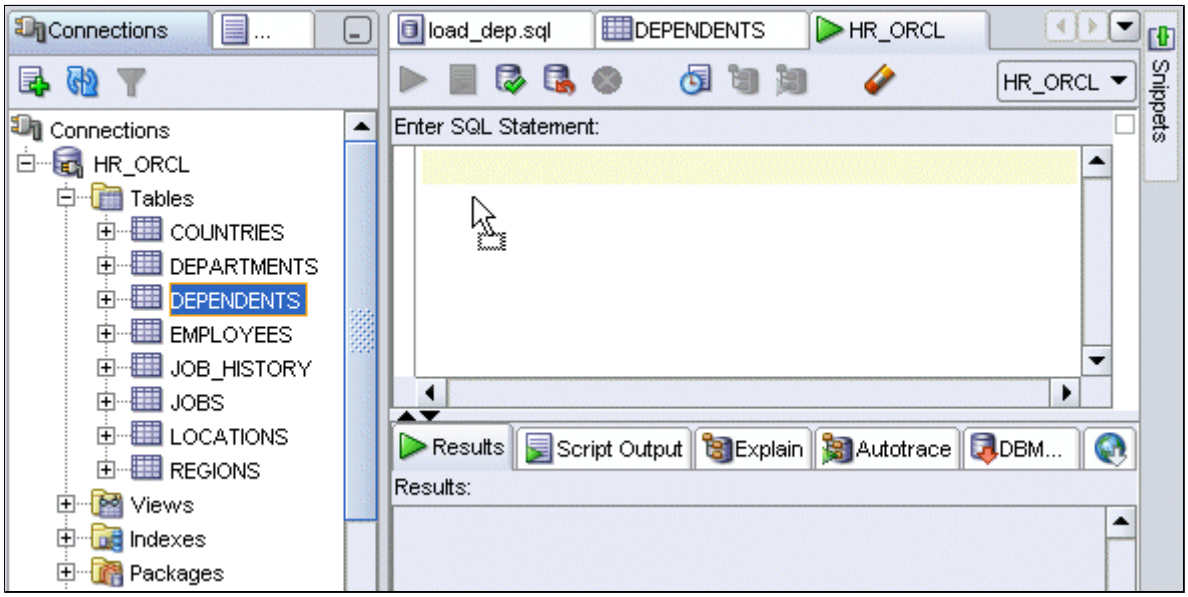
Back to Topic List

One way to access DEPENDENTS data is to generate a SELECT statement on the DEPENDENTS table and add a WHERE clause. Perform the following steps:

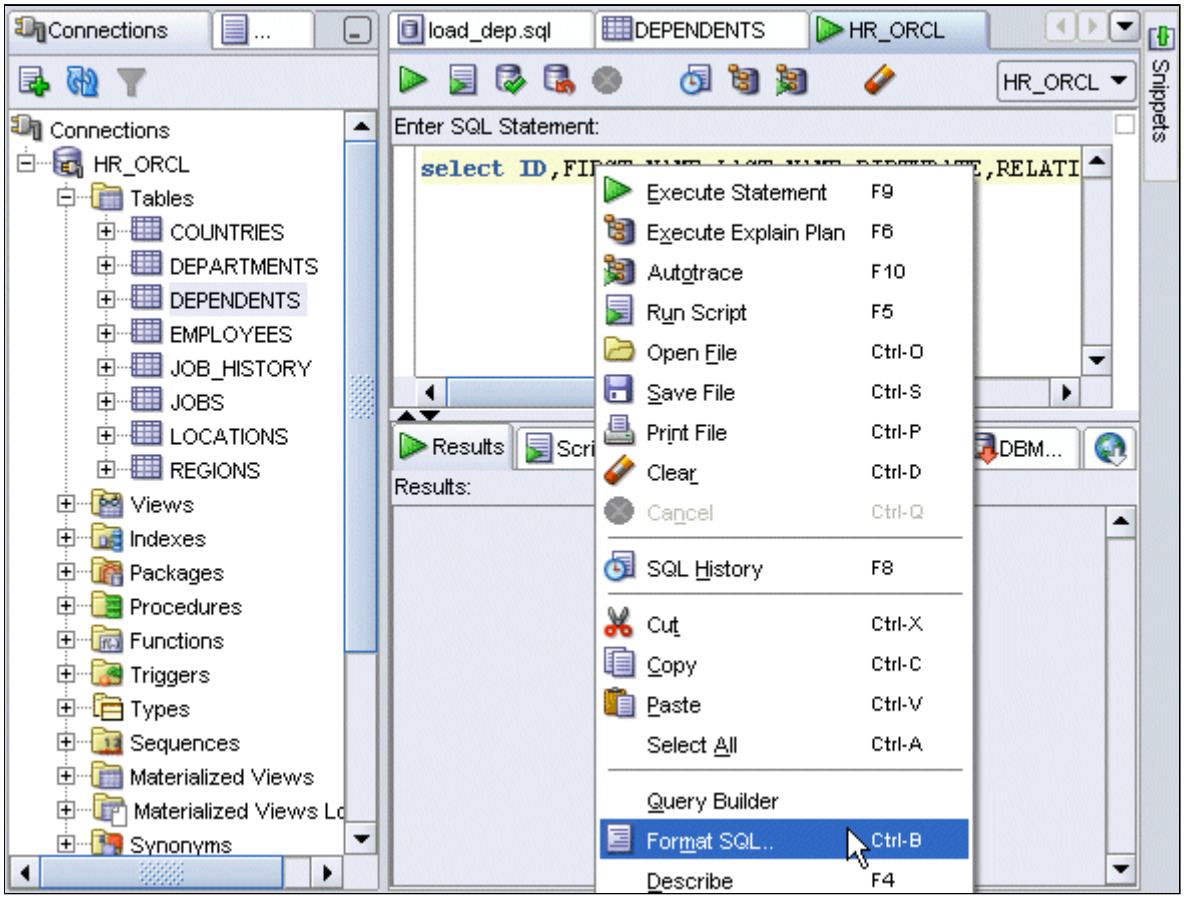
1. Select the **HR_ORCL** Database Connection, right-click and select Open SQL Worksheet.



2. Drag and Drop the **DEPENDENTS** table from the list of database objects to the SQL statement area.

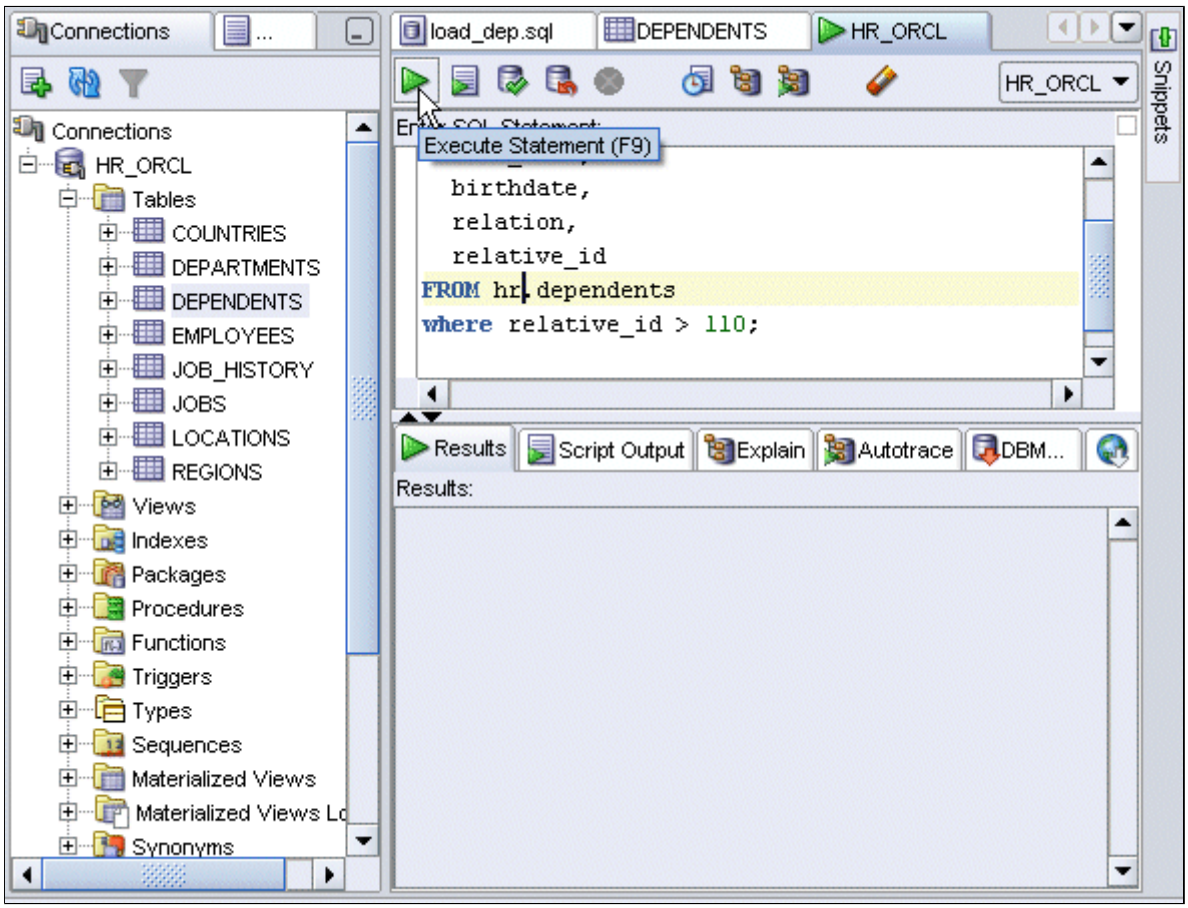


3. A SELECT statement is shown with all the columns contained in the DEPENDENTS table. **Right-click** and select **Format SQL...**

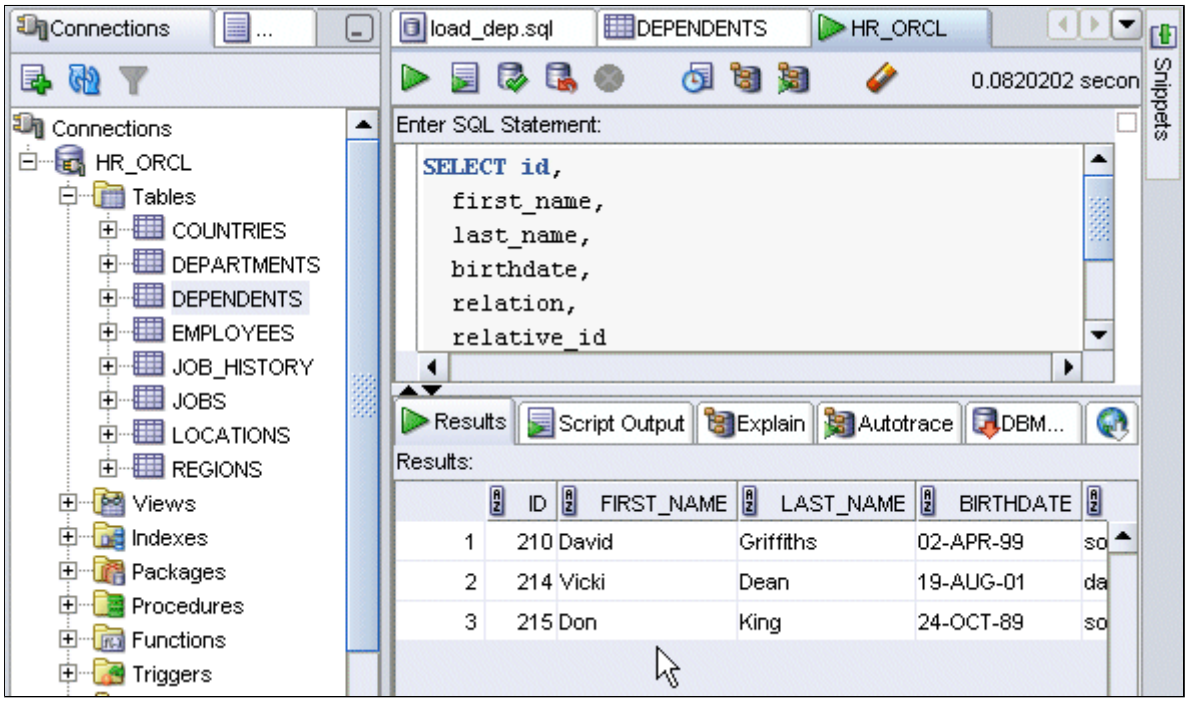


4. Add the WHERE clause *where relative_id > 110* to the end of the SELECT statement BEFORE the ';'.

Then click the **Execute Statement** icon.



5. The results are shown.

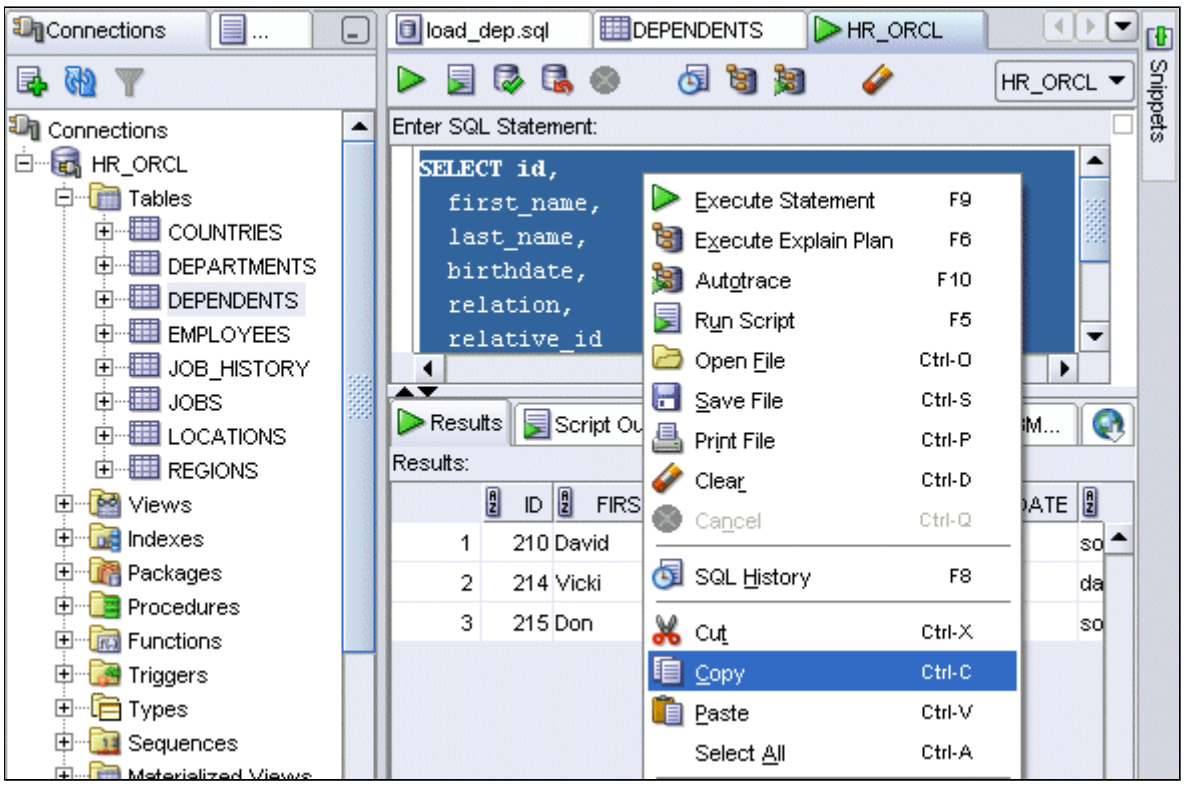


Back to Topic List

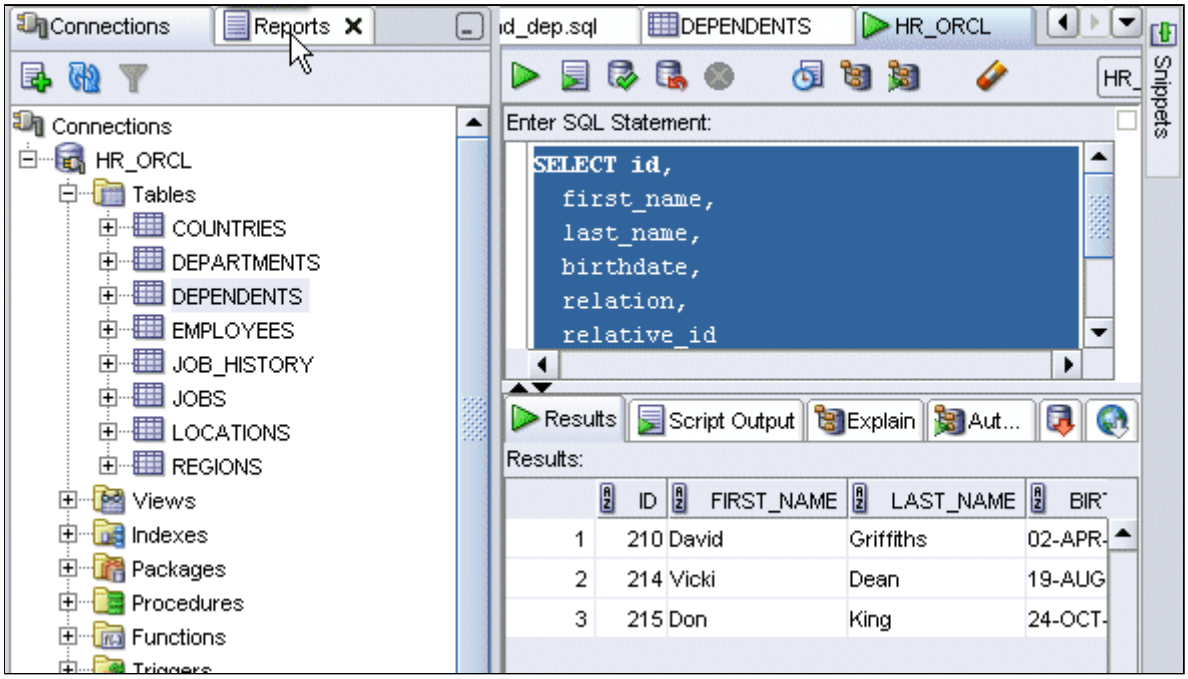
Creating Reports

Since the SQL you just ran in the previous topic needs to be executed frequently, you can create a custom report based on the SQL. In addition, you can run a report of your database data dictionary using bind variables. Perform the following steps:

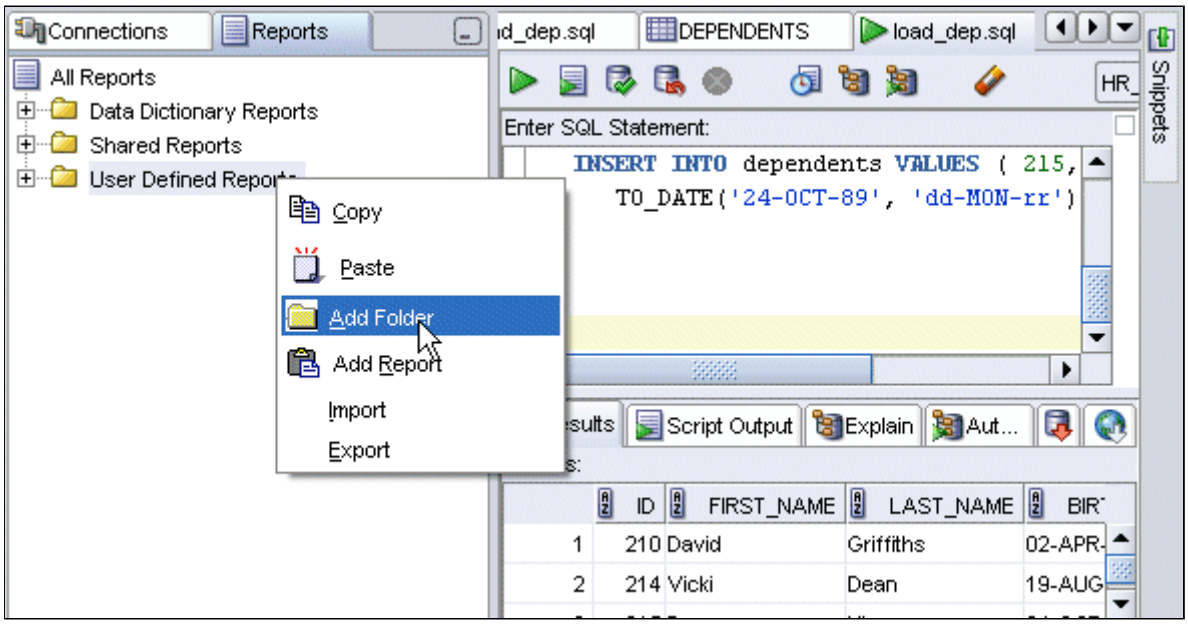
1. Select the SQL in the HR_ORCL SQL Worksheet that you executed, right-click and select **Copy**.



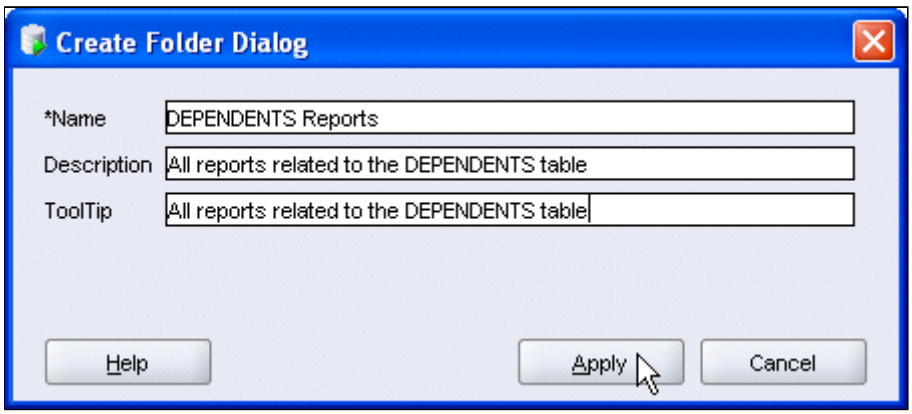
2. Click the **Reports** tab.



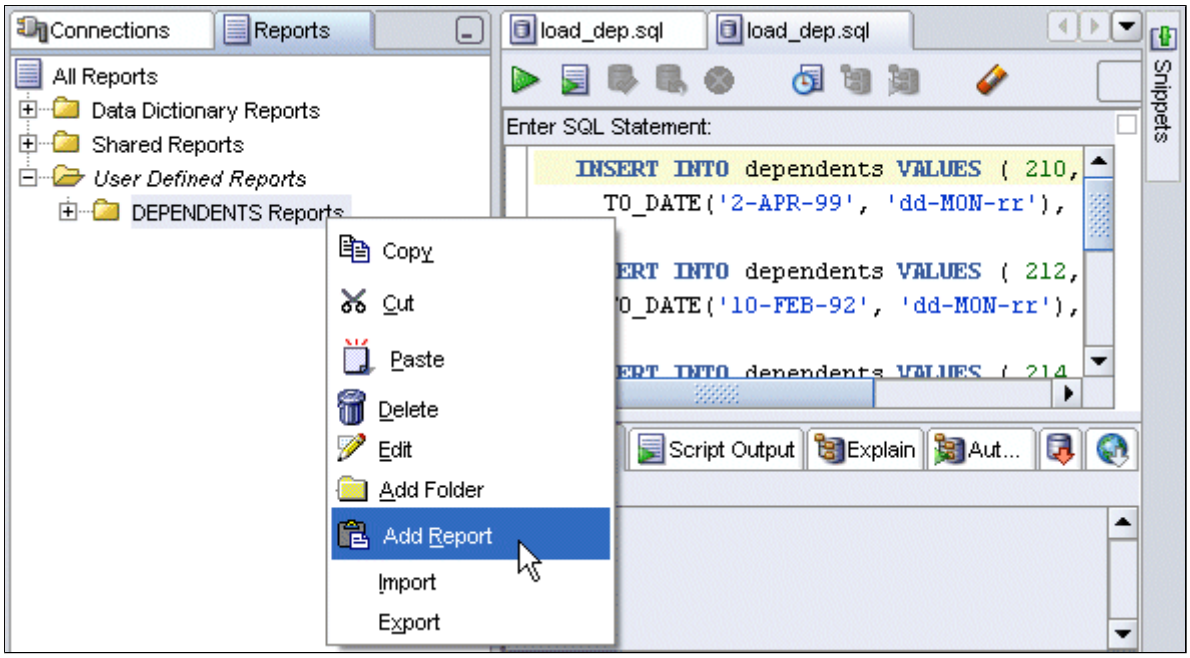
3. Right-click **User Defined Reports** and select **Add Folder**.



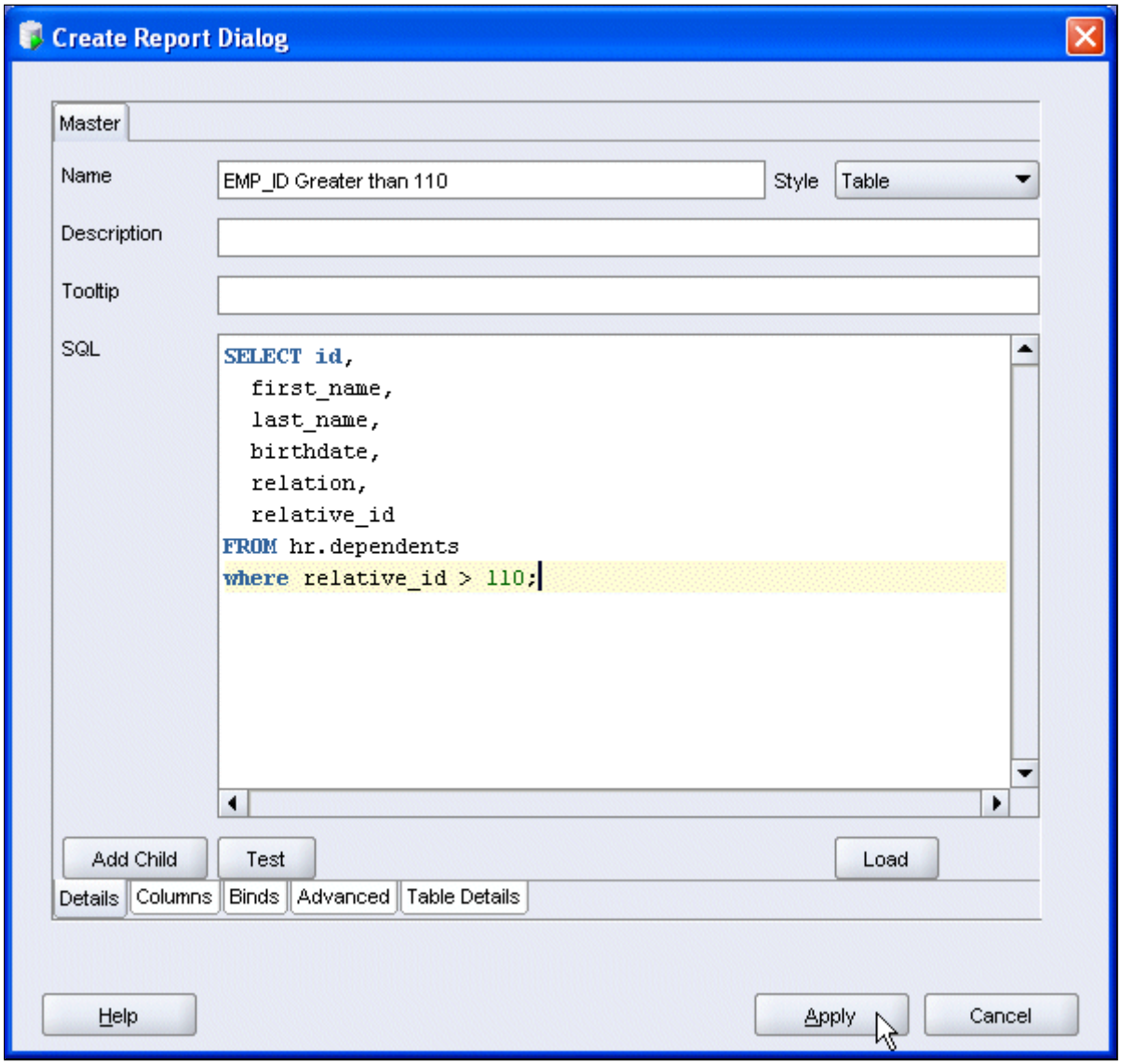
4. Enter the Folder Name **DEPENDENTS Reports**. You can add a Description and Tooltip of your choice. Click **Apply**.



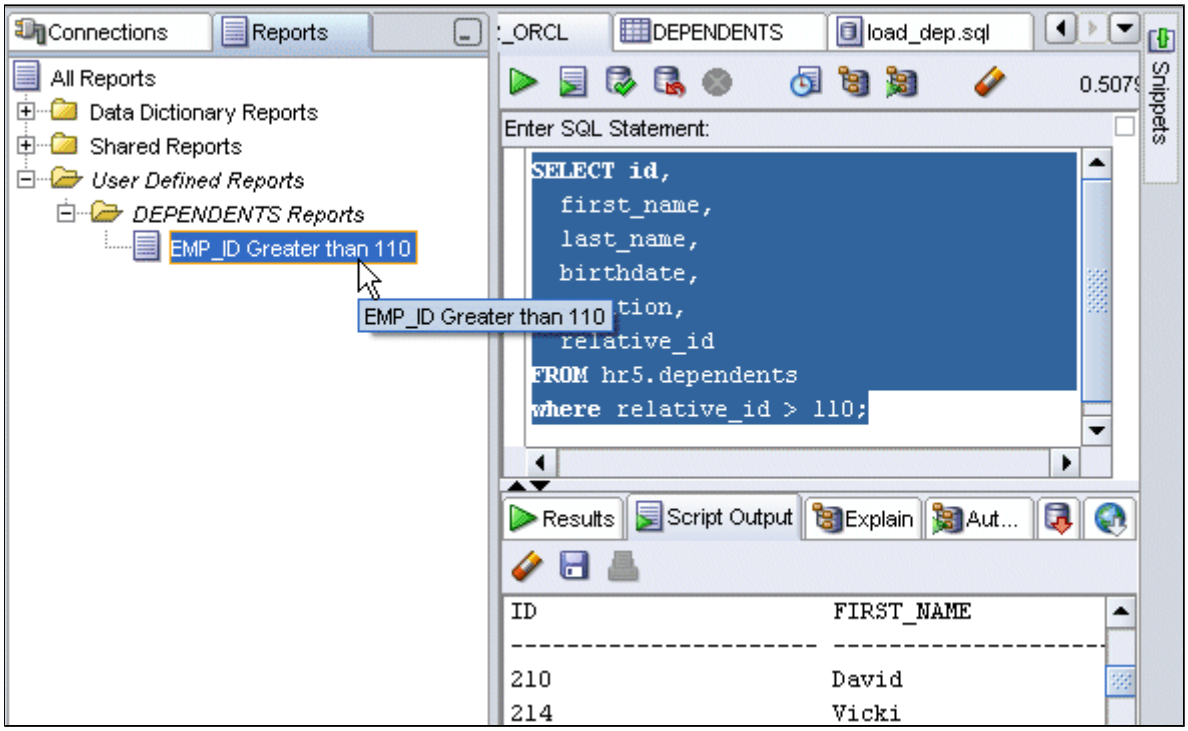
5. Right-click **DEPENDENT Reports** and select **Add Report**.



6. Enter a Name for the report and click **ctrl+v** in the SQL area to paste the SQL you copied from the HR_ORCL SQL Worksheet. Then click **Apply**.

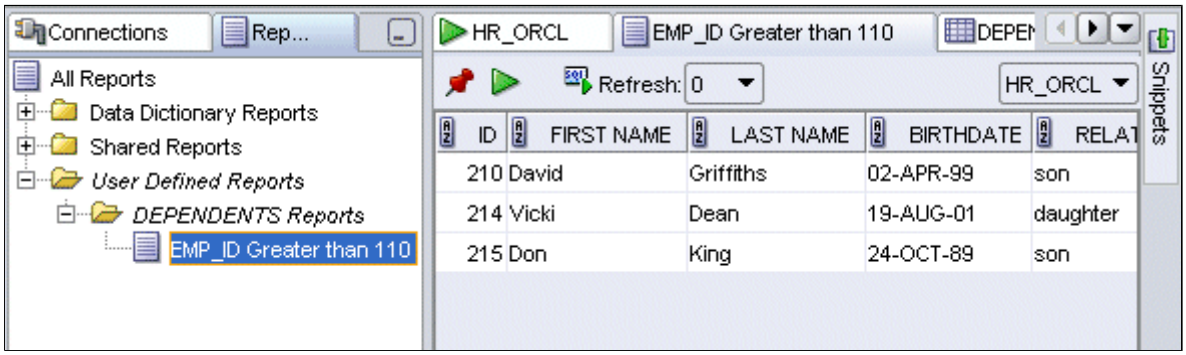


7. Select the Report you just created.

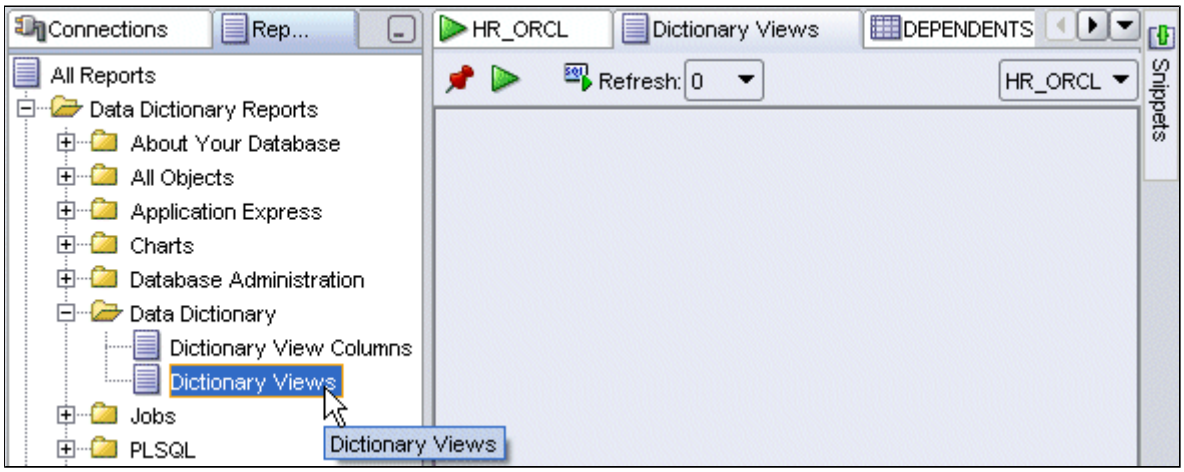


8. Select HR_ORCL from the drop list and click **OK** to connect to your database.

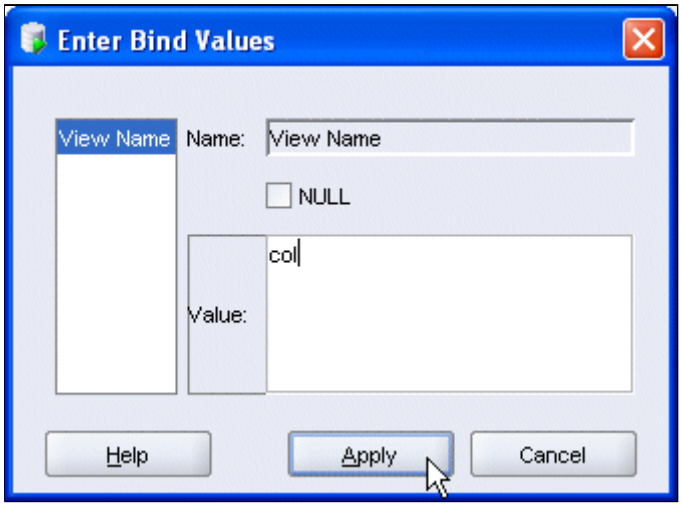
9. The results of your report are shown.



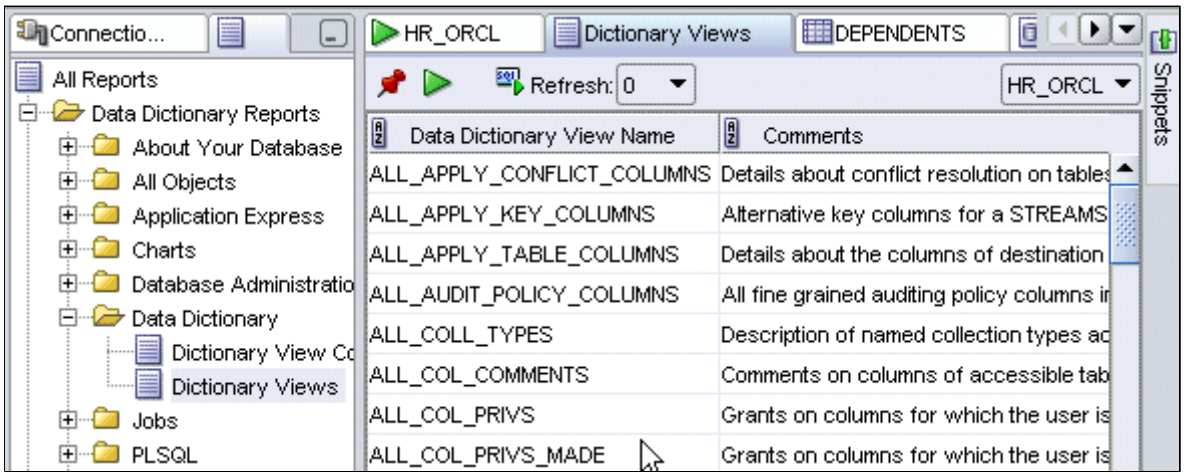
10. You can also run a Data Dictionary report. Expand **Data Dictionary Reports > Data Dictionary**. Then select **Dictionary Views**.



11. Enter **col** for the Value and click **Apply**.



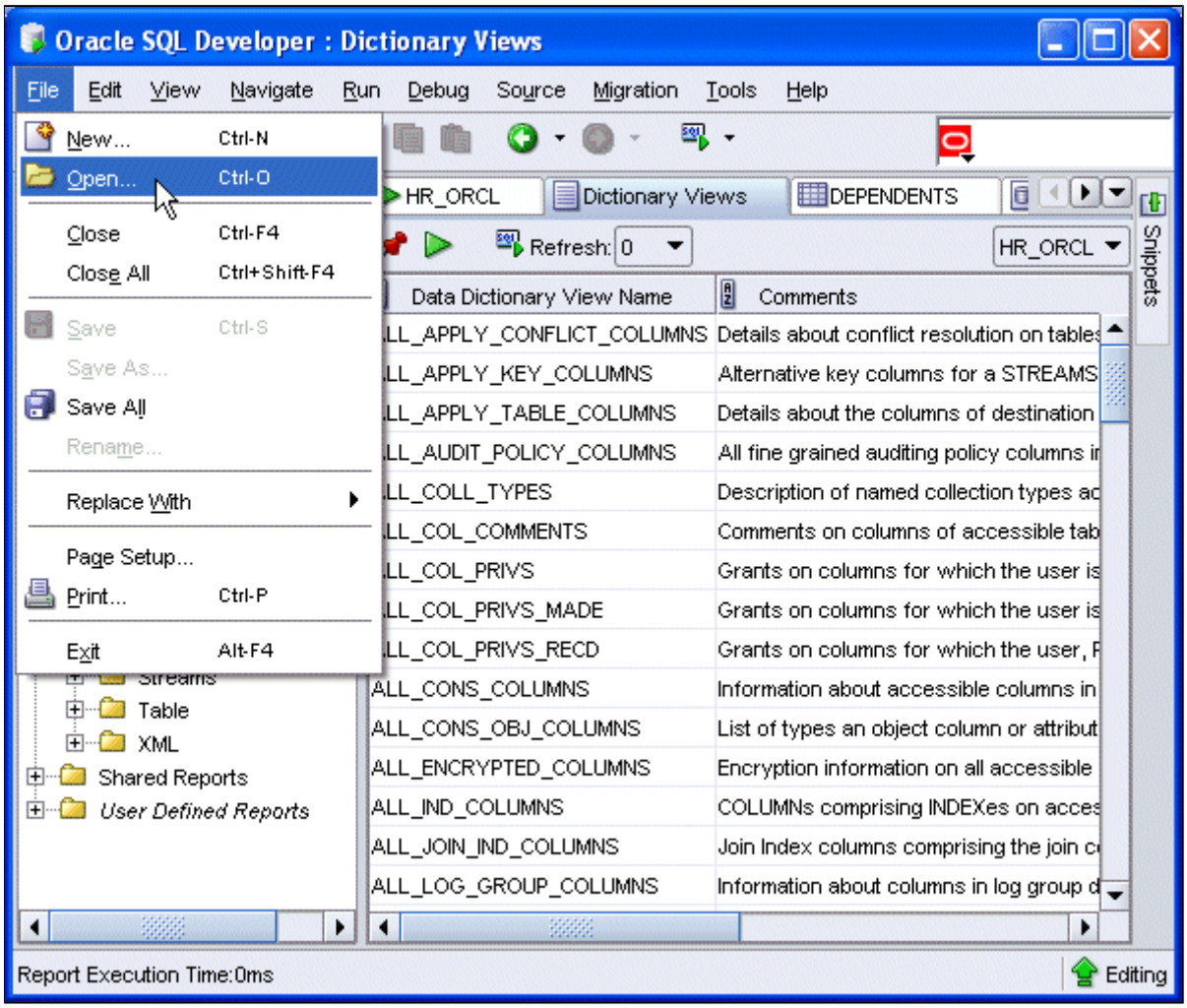
12. All the Data Dictionary views that contain 'col' in its name are displayed.



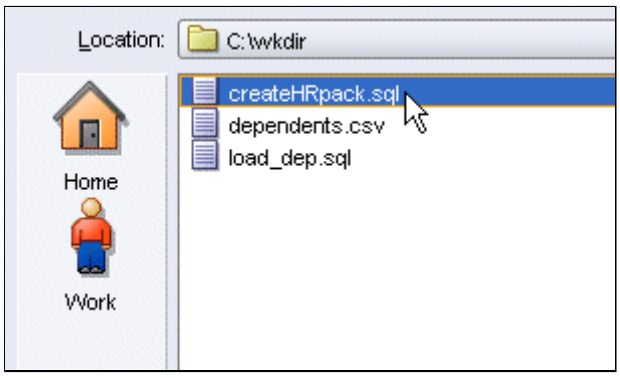
Back to Topic List

Oracle SQL Developer contains extensive PL/SQL debugging capabilities. In this topic, you create a Package Spec and Package Body that adjusts an employee's salary. Perform the following steps:

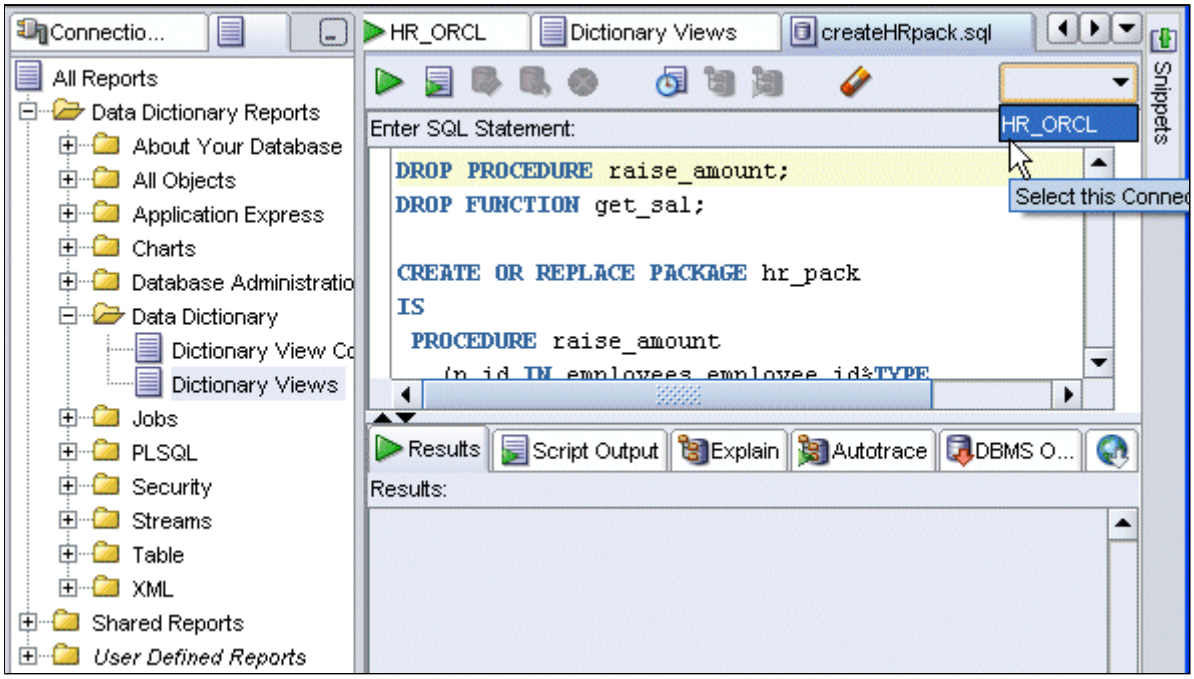
1. Select **File > Open** using the main menu.



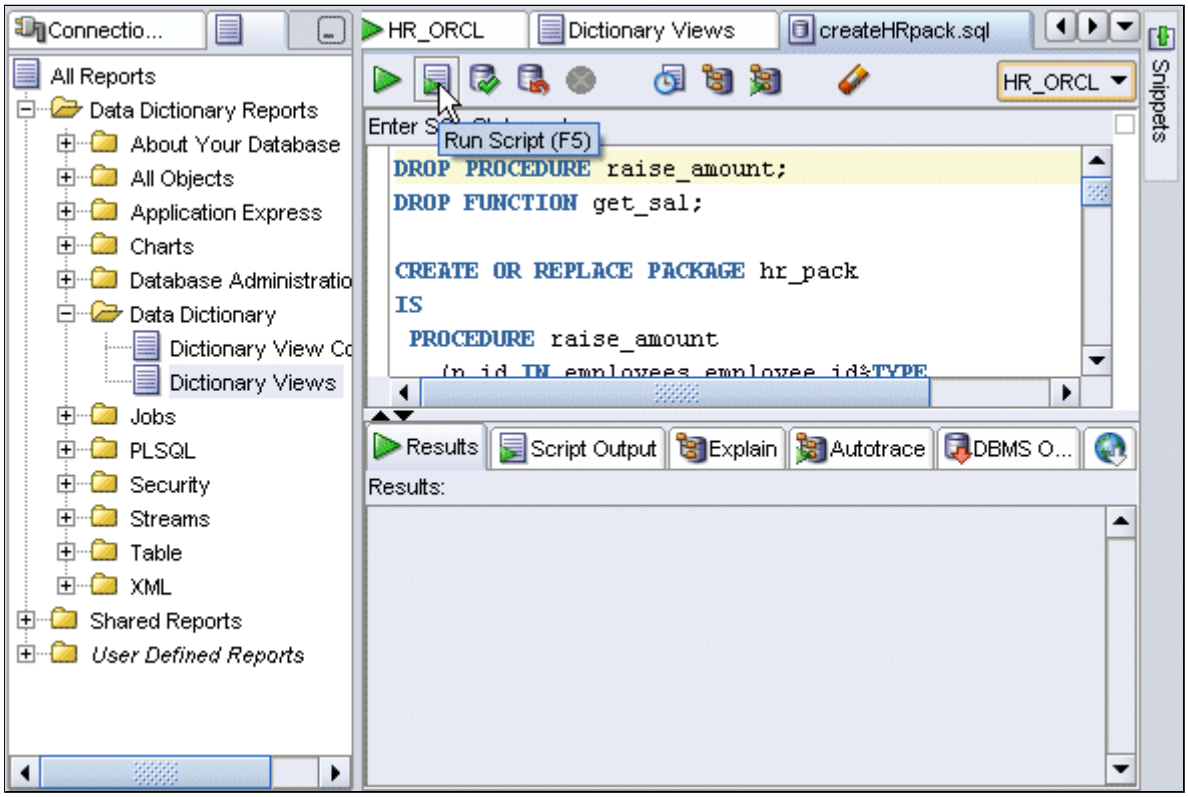
2. Browse to the **\SQLDev_HOS\GettingStarted\files** directory and select **createHRpack.sql** Click **Open**.



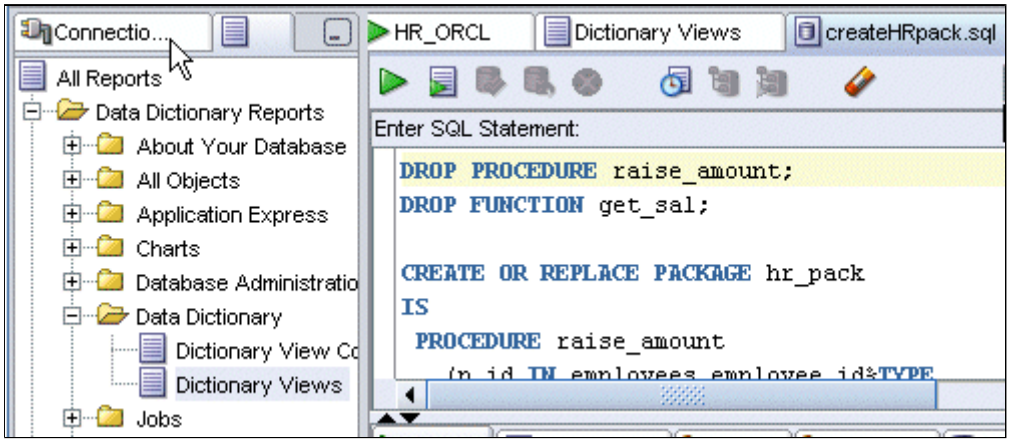
3. Select the HR_ORCL database connection from the the drop list on the right.



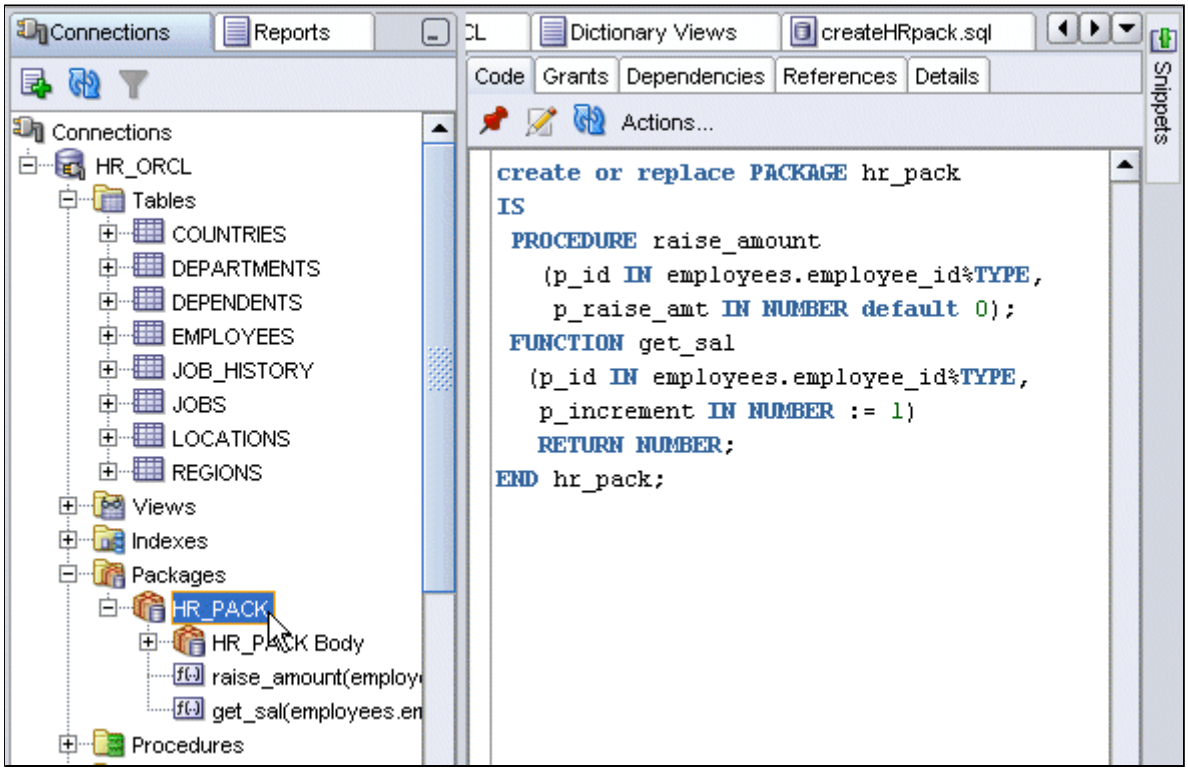
4. Click the **Run Script** icon.



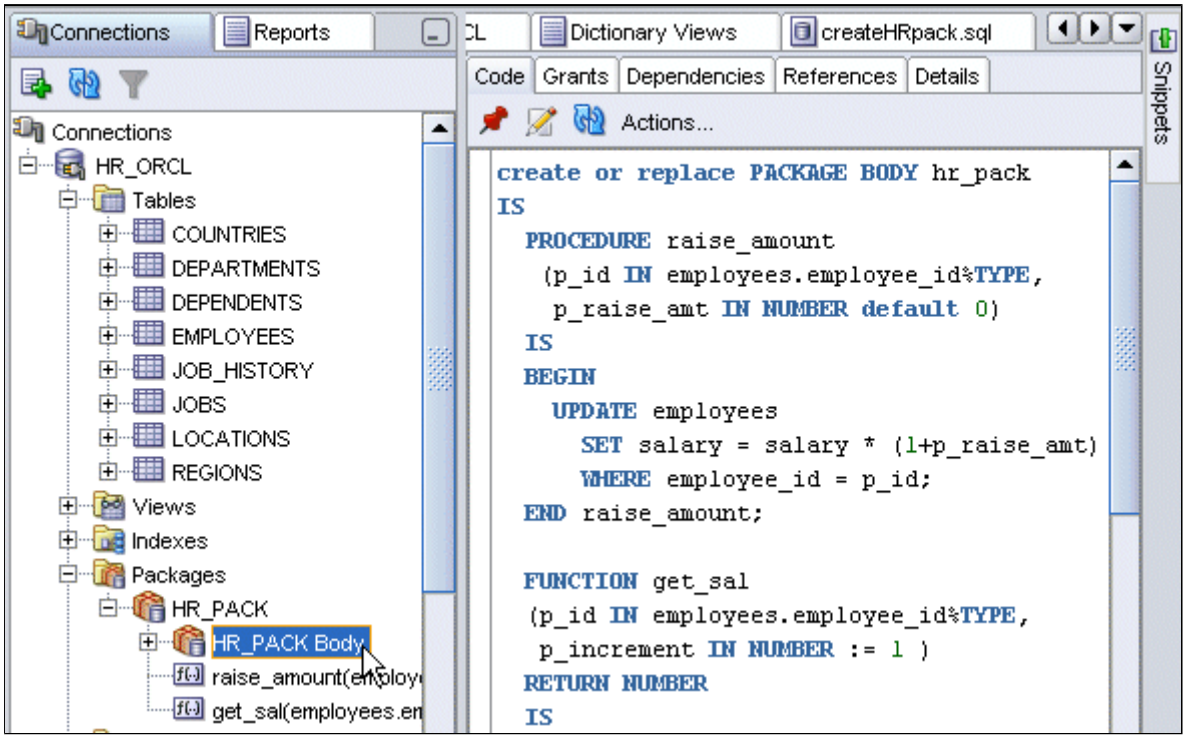
5. Click the **Connections** tab.



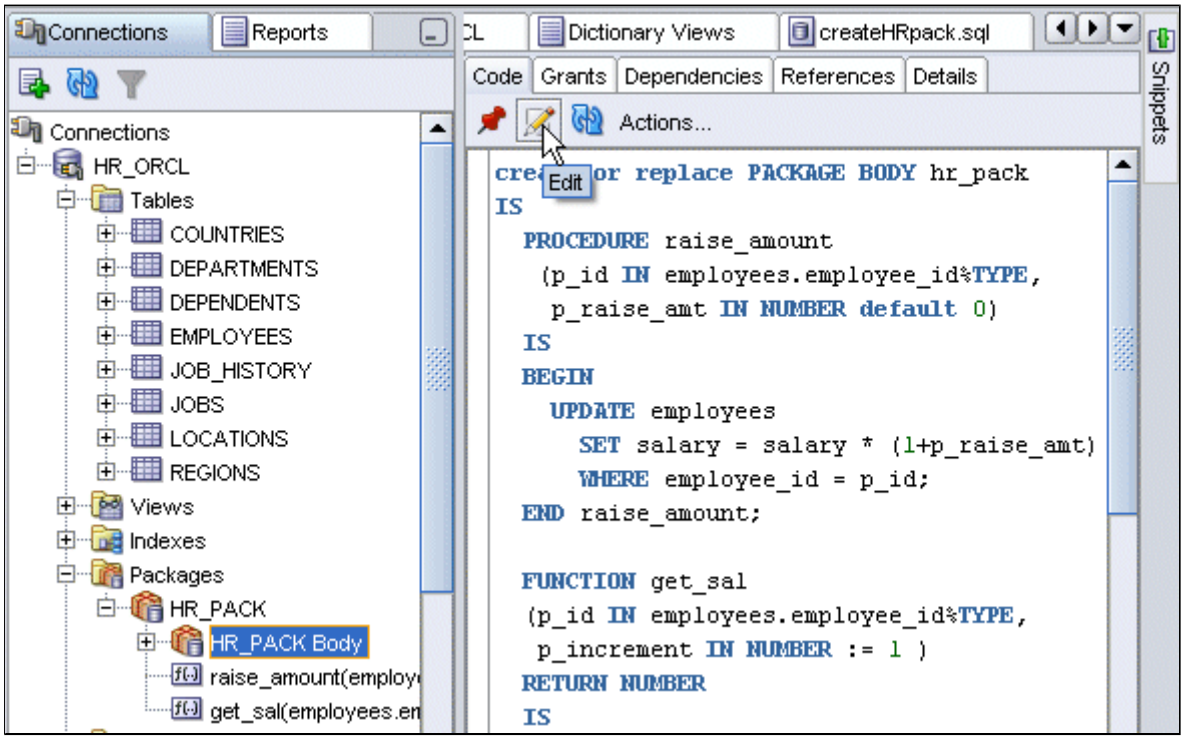
6. Expand **HR_ORCL > Packages > HR_PACK** and select **HR_PACK** to view the package definition.



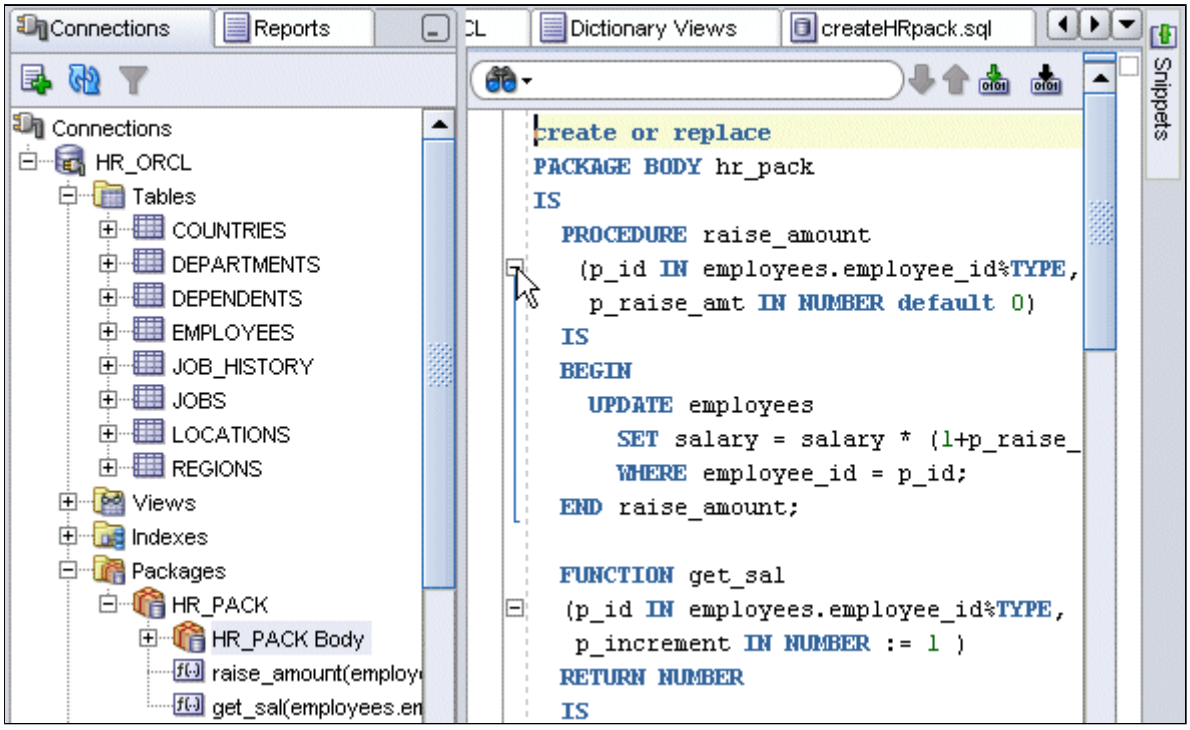
7. Select **HR_PACK BODY** to view the package body definition.



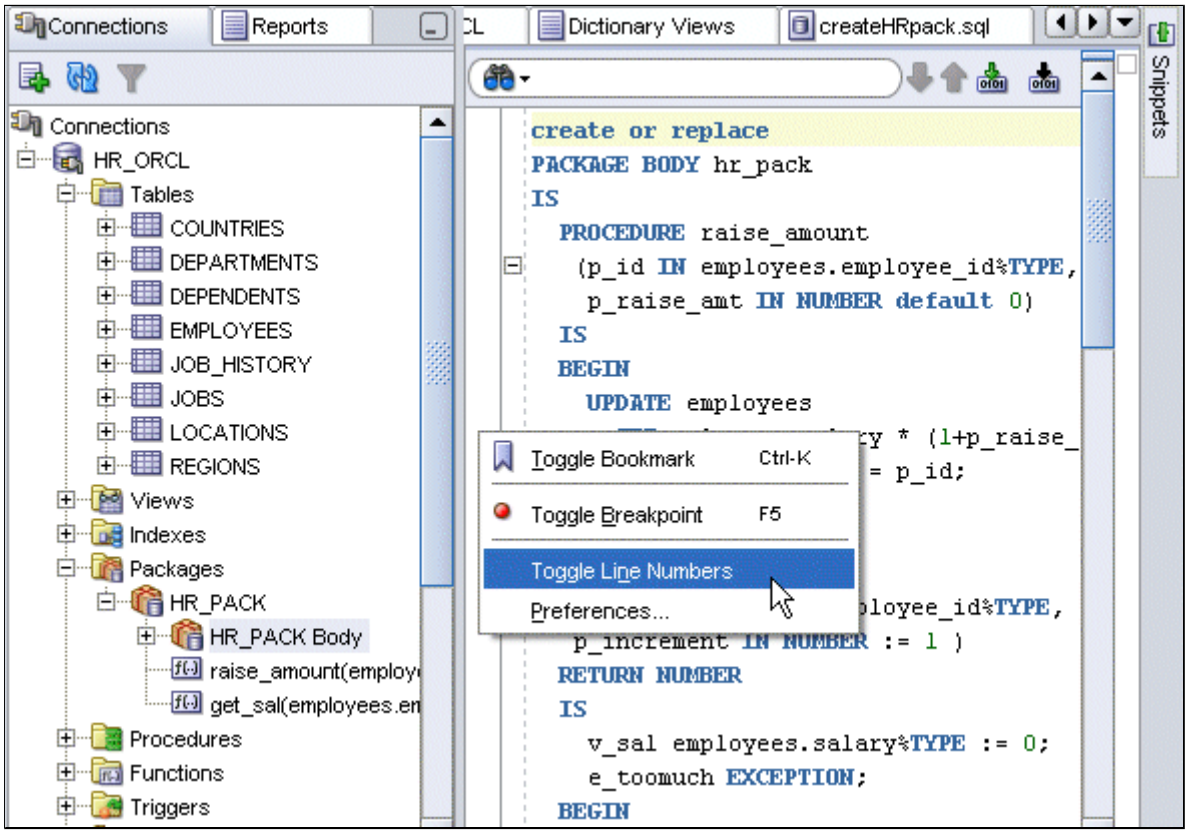
8. To make any changes to the Package Body, click the **Edit** icon.



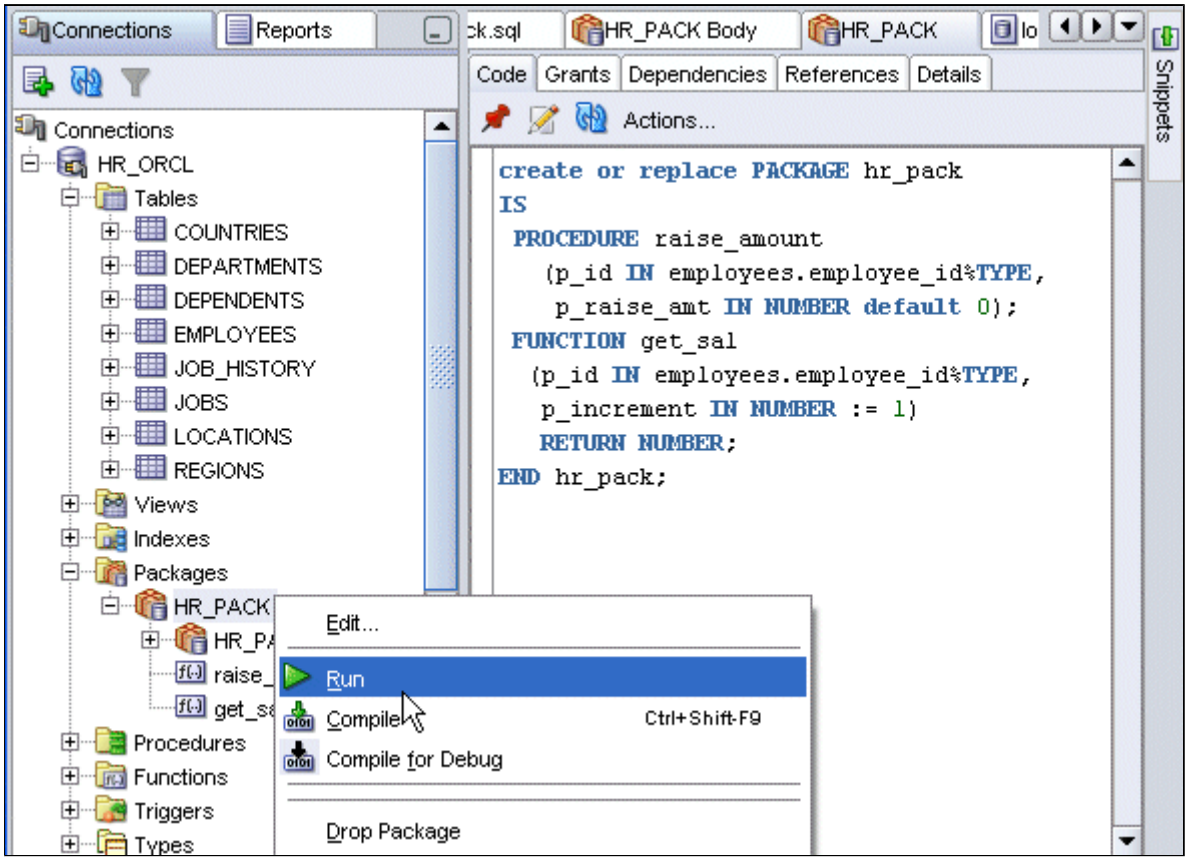
9. Click on any one of the **-** to collapse the code or press **+** to expand the code.



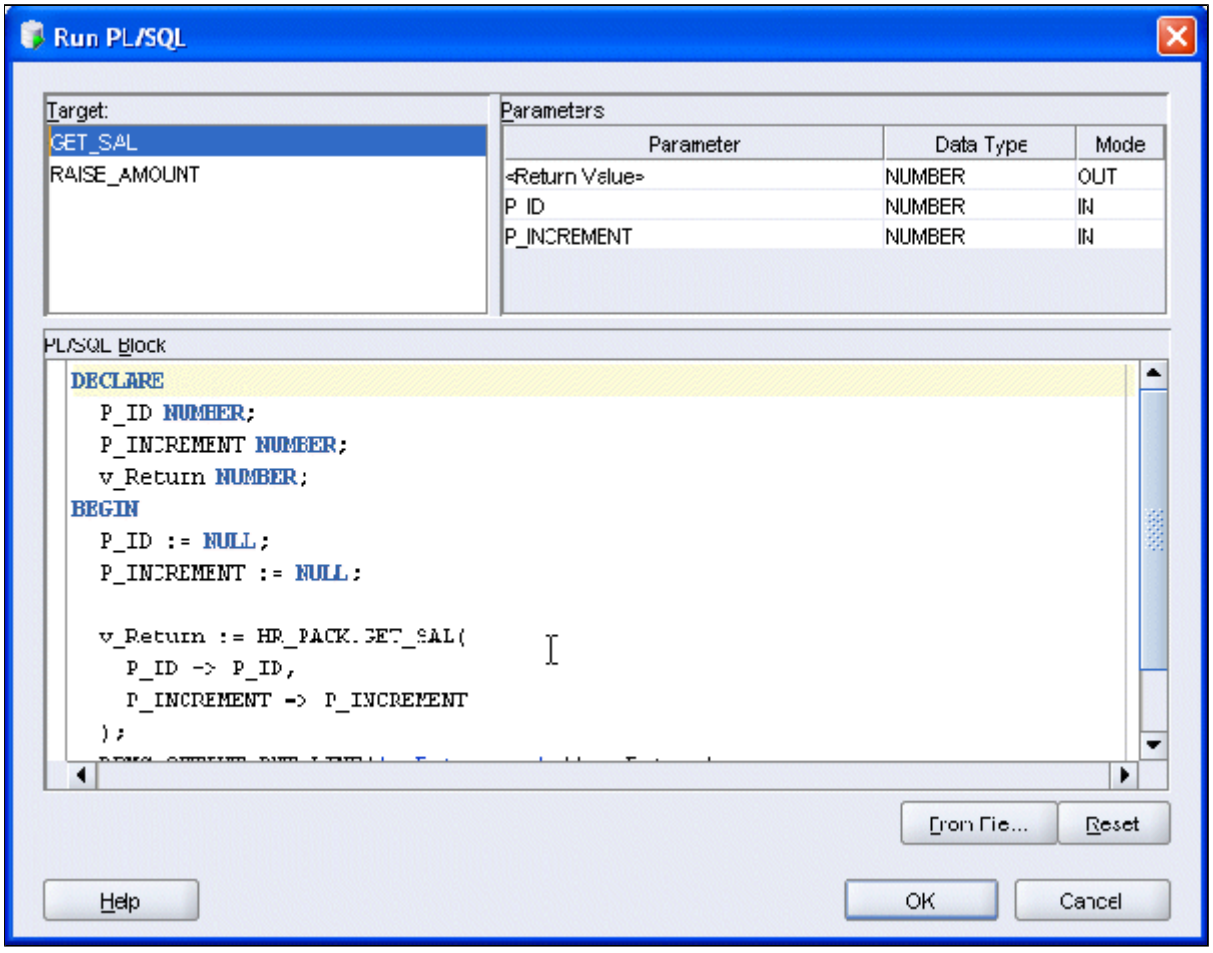
10. If your line numbers do not appear, you can right-click in the line number area and click **Toggle Line Numbers** to turn them on. This is useful for debugging purposes.



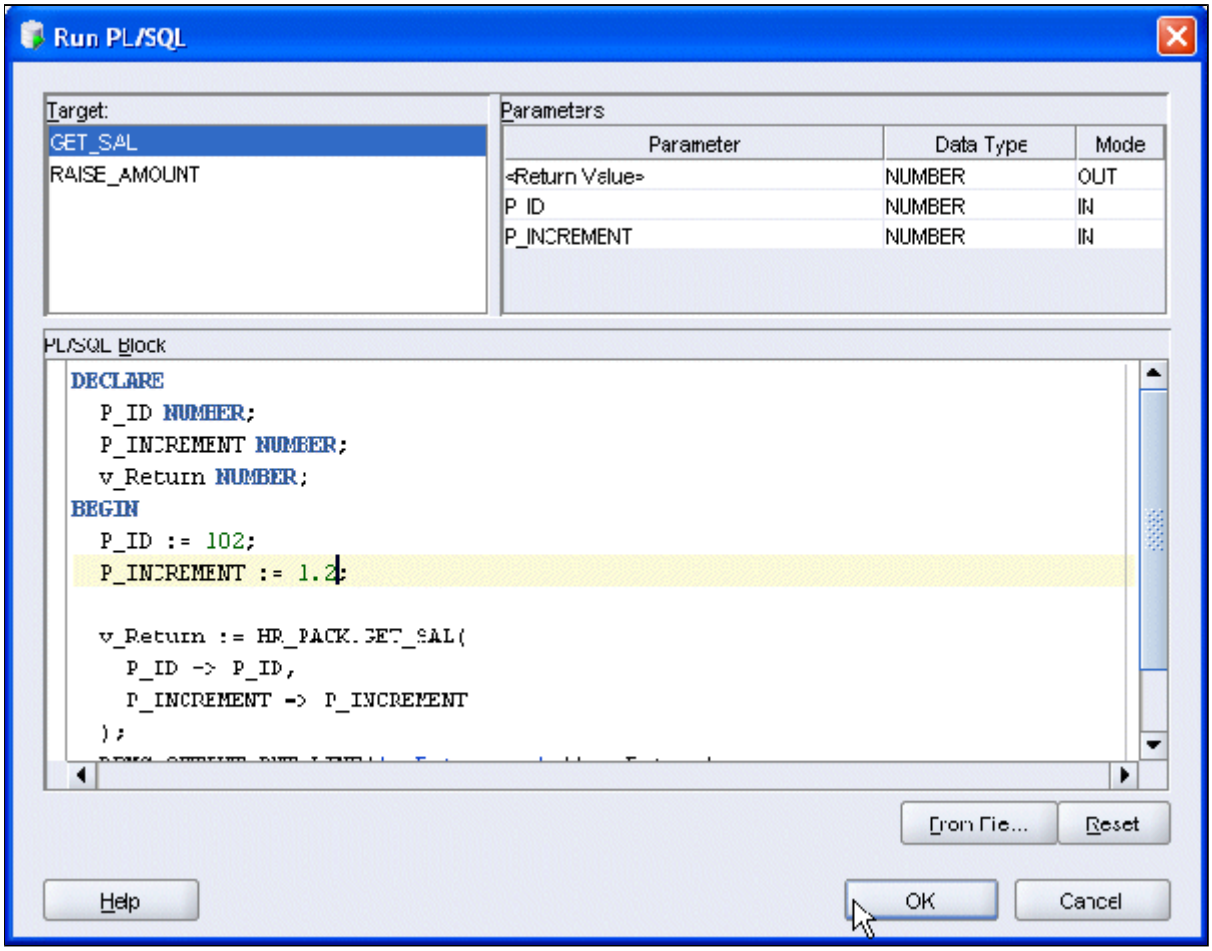
11. In the Connections Navigator, select Packages > **HR_PACK,** right-click and select **Run**.



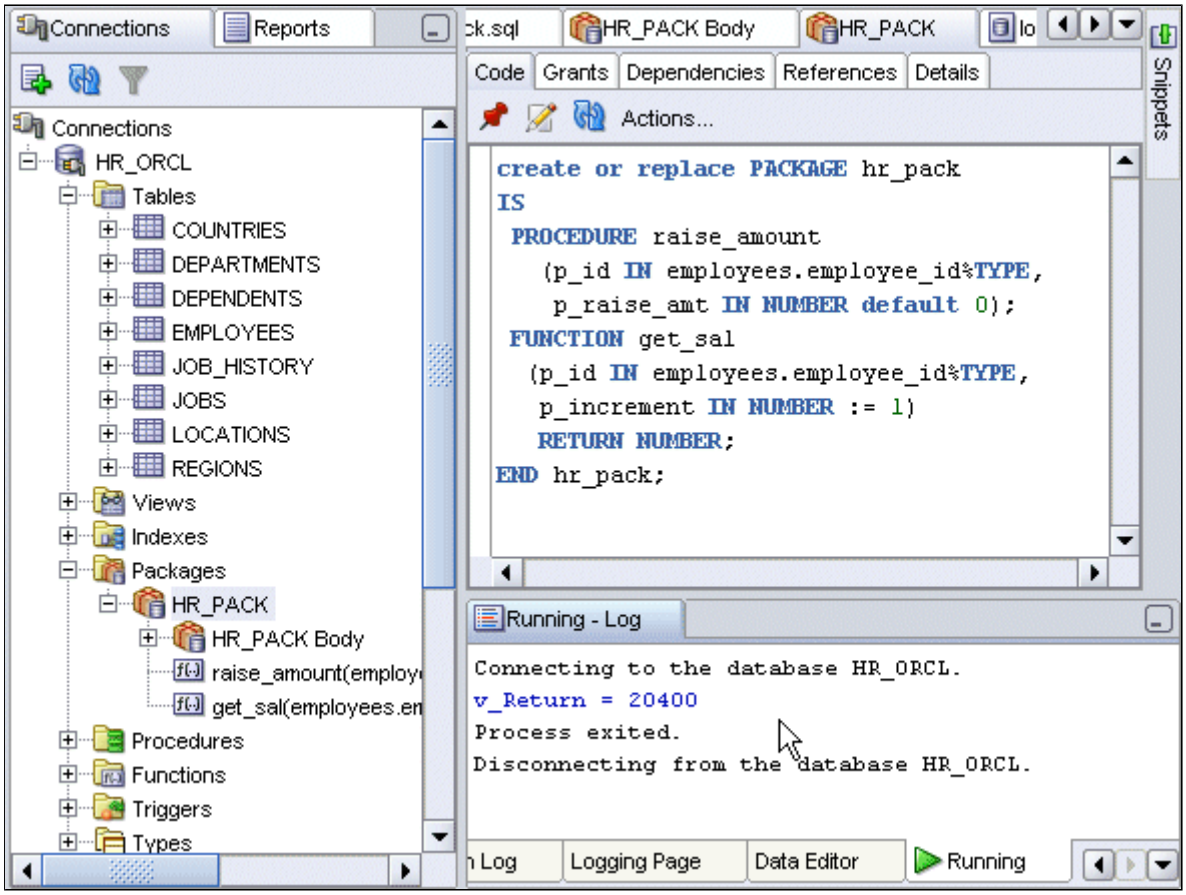
12. A parameter window appears. You need to set the input parameters here.



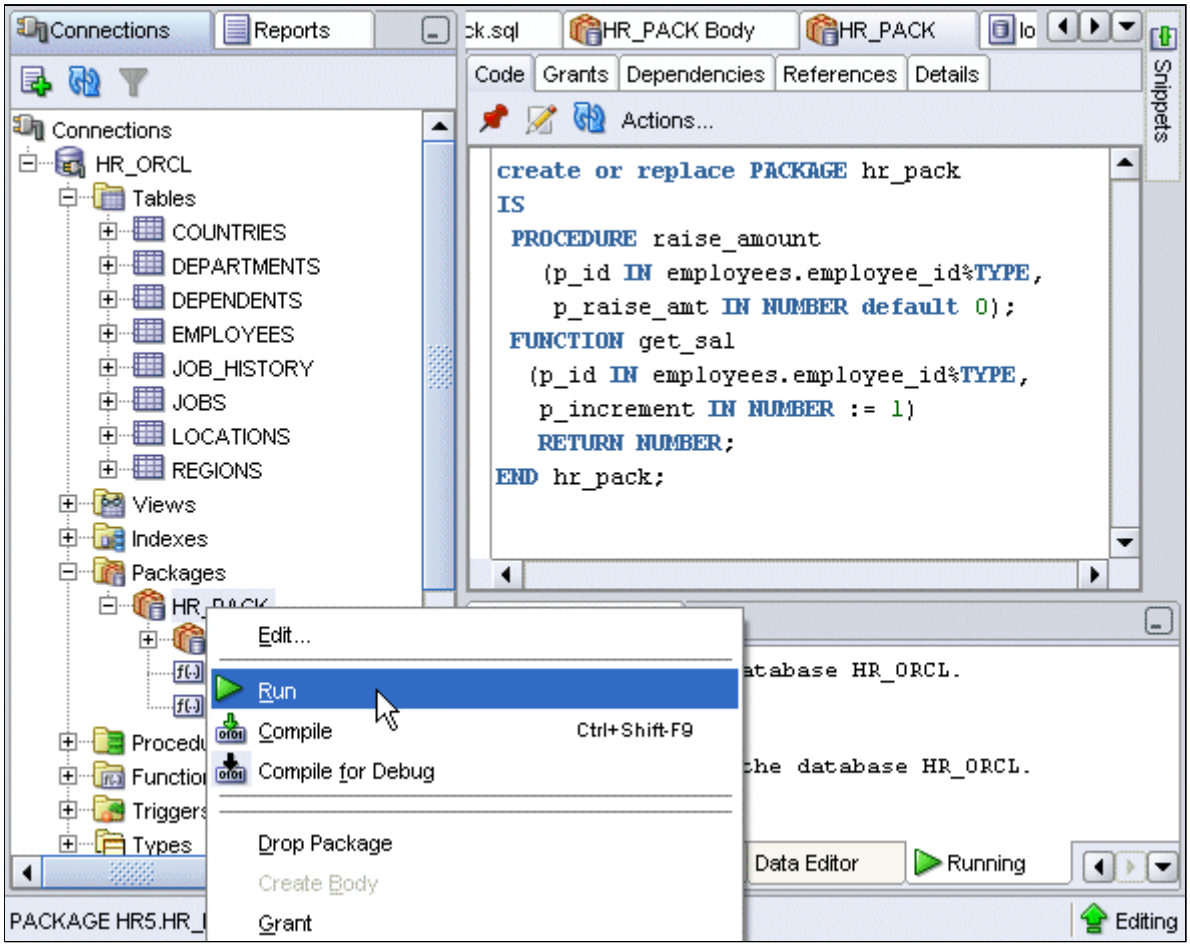
Set the P_ID to **102** and P_INCREMENT to **1.2**. What this means is that the Employee who has the ID 102, their salary is increased by 20%. The current SALARY for EMPLOYEE_ID 102 is 17000. Click **OK**.



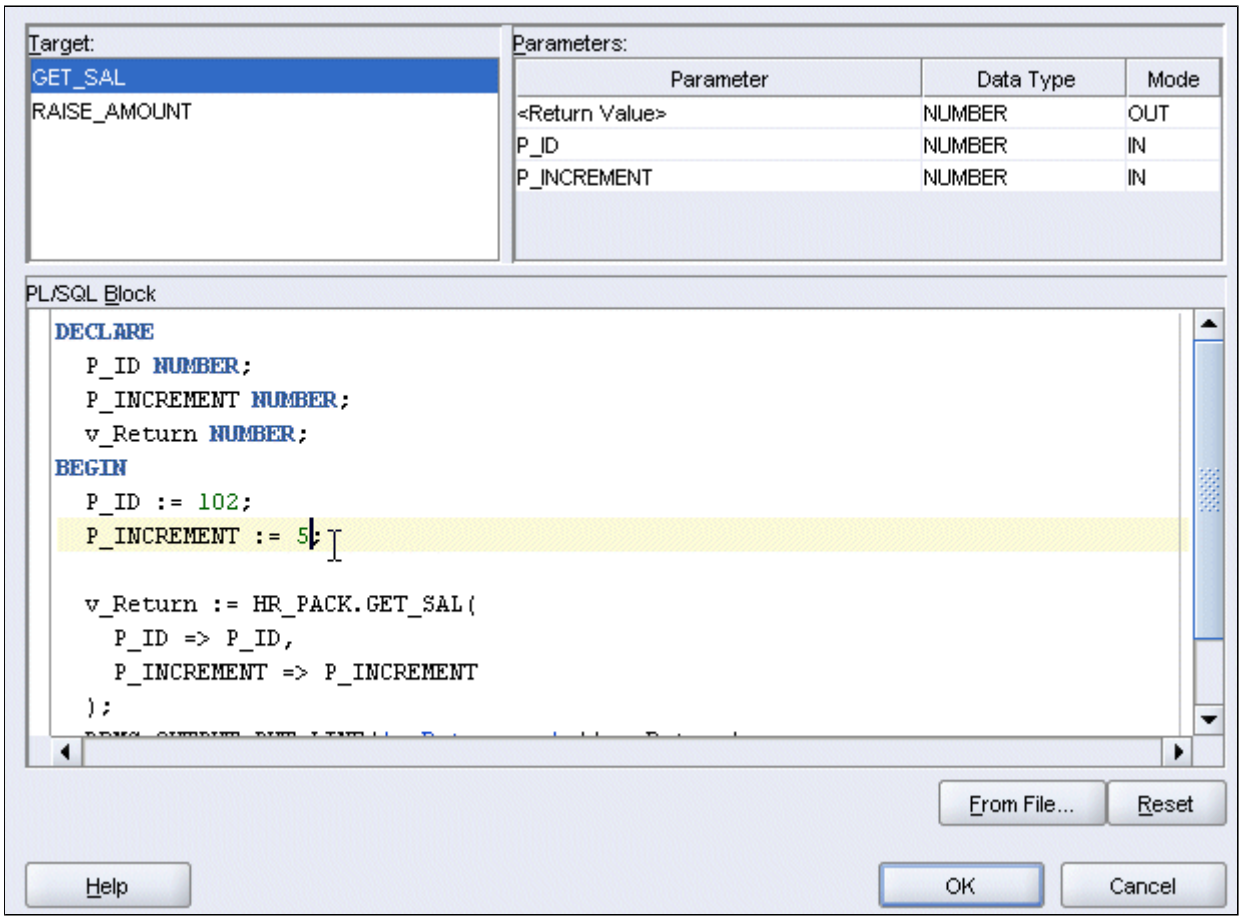
13. The value returned is 20400.



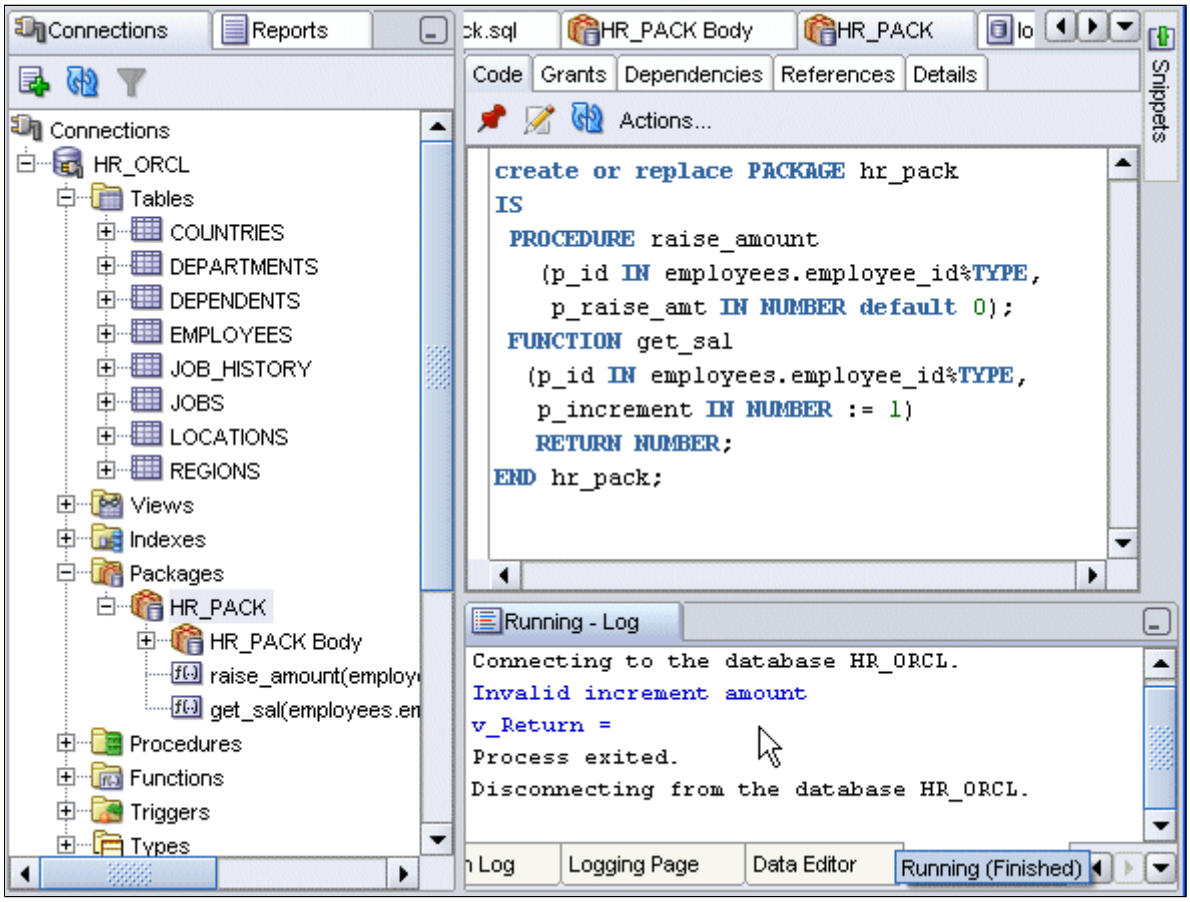
14. To test the Exception Handling, right-click on **HR_PACK** in the navigator and select **Run**.



15. This time, change the P_INCREMENT value to **5** and click **OK**.



16. In this case, an exception was raised with "Invalid increment value" because the P_INCREMENT value was greater than 1.5.



Back to Topic List

Summary

In this tutorial, you learned how to:

- Create a database connection
- Add a new table using the Table Dialog Box
- Change a table definition
- Add data to a table
- Access data
- Generate a report
- Debug and execute PL/SQL

Back to Topic List

Move your mouse over this icon to hide all screenshots.

Developing and Debugging PL/SQL using Oracle SQL Developer

Purpose

This tutorial shows you how to create, run, and debug a PL/SQL procedure using Oracle SQL Developer.

Time to Complete

Approximately 30 minutes

Topics

This tutorial covers the following topics:

Viewing Screenshots

Place the cursor over this icon to load and view all the screenshots for this tutorial. (Caution: This action loads all screenshots simultaneously, so response time may be slow depending on your Internet connection.)

Note: Alternatively, you can place the cursor over each individual icon in the following steps to load and view only the screenshot associated with that step.

Overview

Oracle SQL Developer is a free graphical tool that enhances productivity and simplifies database development tasks. With Oracle SQL Developer, you can browse database objects, run SQL statements and SQL scripts, and edit and debug PL/SQL statements. You can also run any number of provided reports, as well as create and save your own. This tutorial focuses on creating, compiling, running and debugging PL/SQL.

Back to Topic List

Prerequisites

For this Hands On Session, the following has already been installed for you:

1. Oracle Database 10g.

   Note: To repeat this exercise later you can use any Oracle Database above 9.2.0.1

2. Oracle SQL Developer 1.2.1.

   Note: Oracle SQL Developer is available for download for FREE from **OTN.** To install unzip it into any directory on your machine.

3. The files you use throughout the tutorial are located in the **\SQLDev_HOS\PLSQL\files** directory. The files have also been included in the **plsql.zip** file.

4. The shipped HR schema.

   Note: For the PL/SQL debugging portion of this tutorial, the HR user needs a few additional privileges. These have

been granted using the following commands:

GRANT debug any procedure, debug connect session TO hr;

Back to Topic List

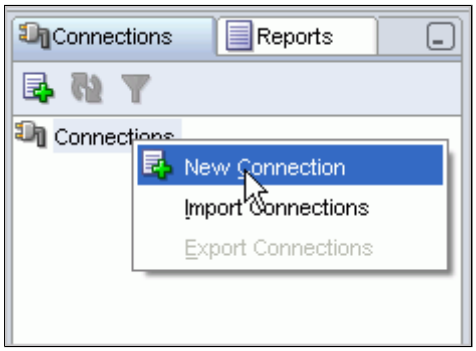
Creating a Database Connection

To create a database connection, perform the following steps:

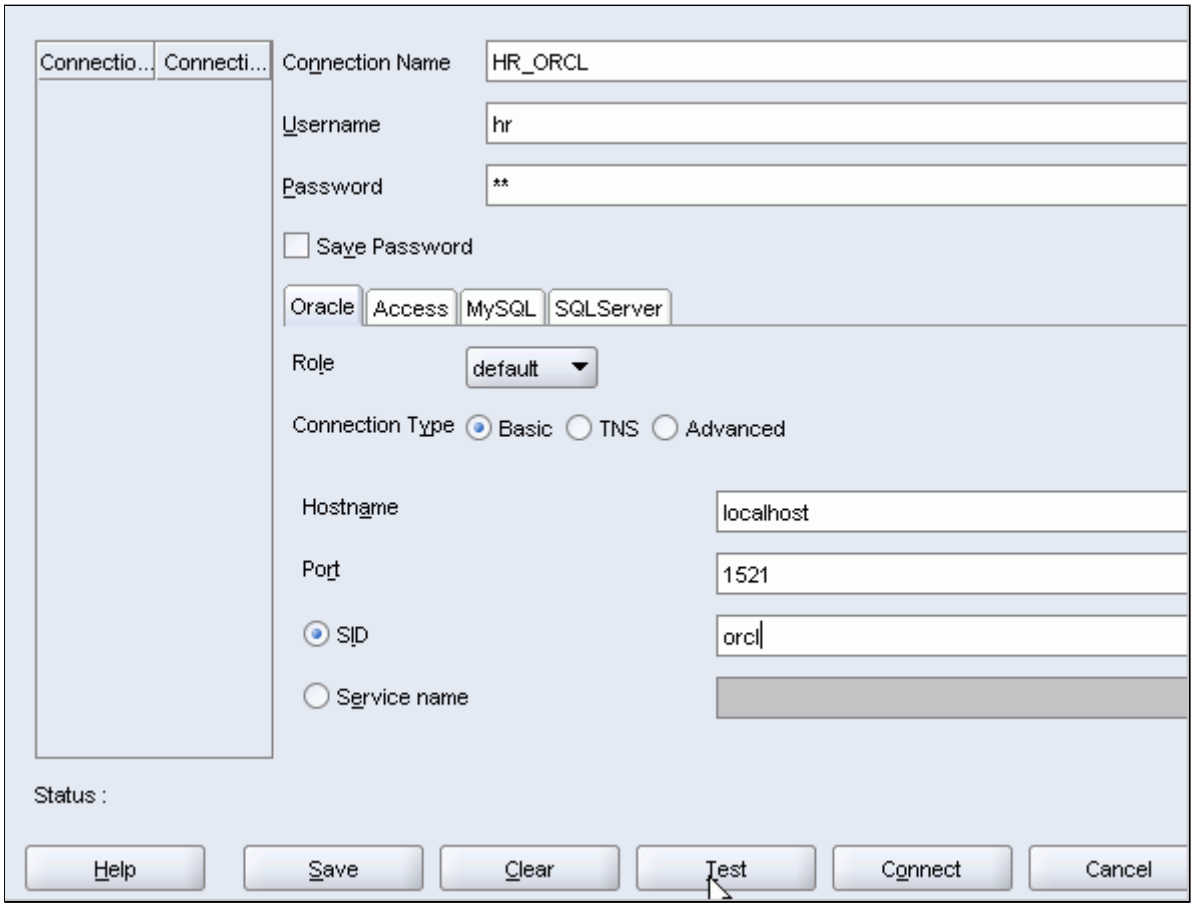
1. Open Windows Explorer and double-click **<your_path>\sqldeveloper\sqldeveloper.exe**.

   Note: If you receive a dialog window asking whether you want to migrate settings from a previous release, click **No**.

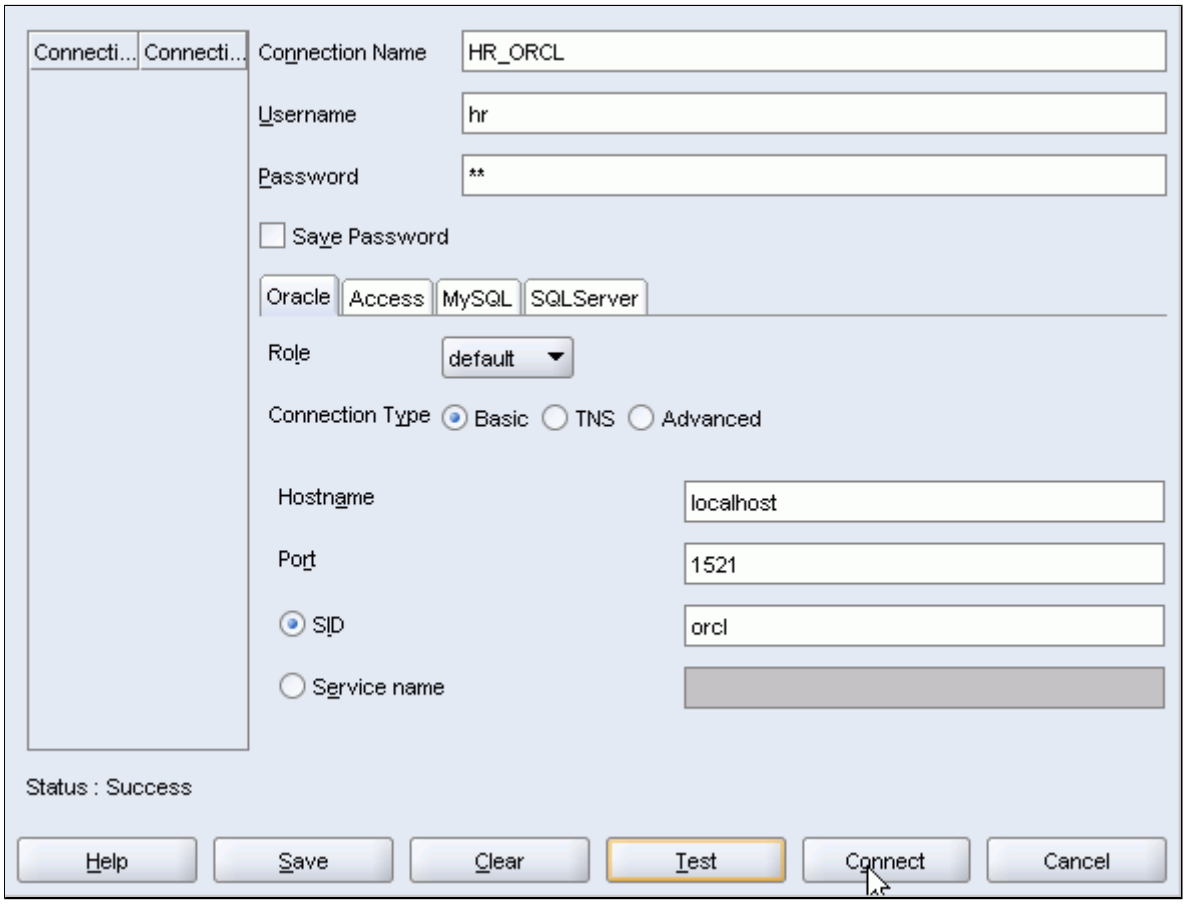
2. In the Connections tab, right-click **Connections** and select **New Connection**.



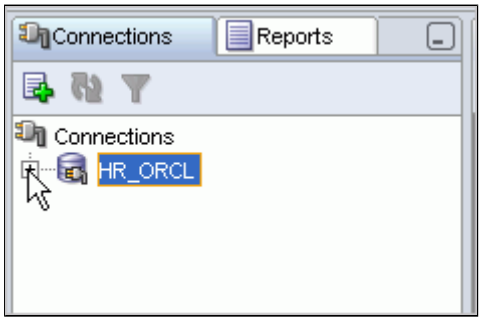
3. Enter **HR_ORCL** for the Connection Name (or any other name that identifies your connection), **HR** for the Username and Password, specify your **<hostname>** for the Hostname and enter **ORCL** for the SID. Click **Test**.



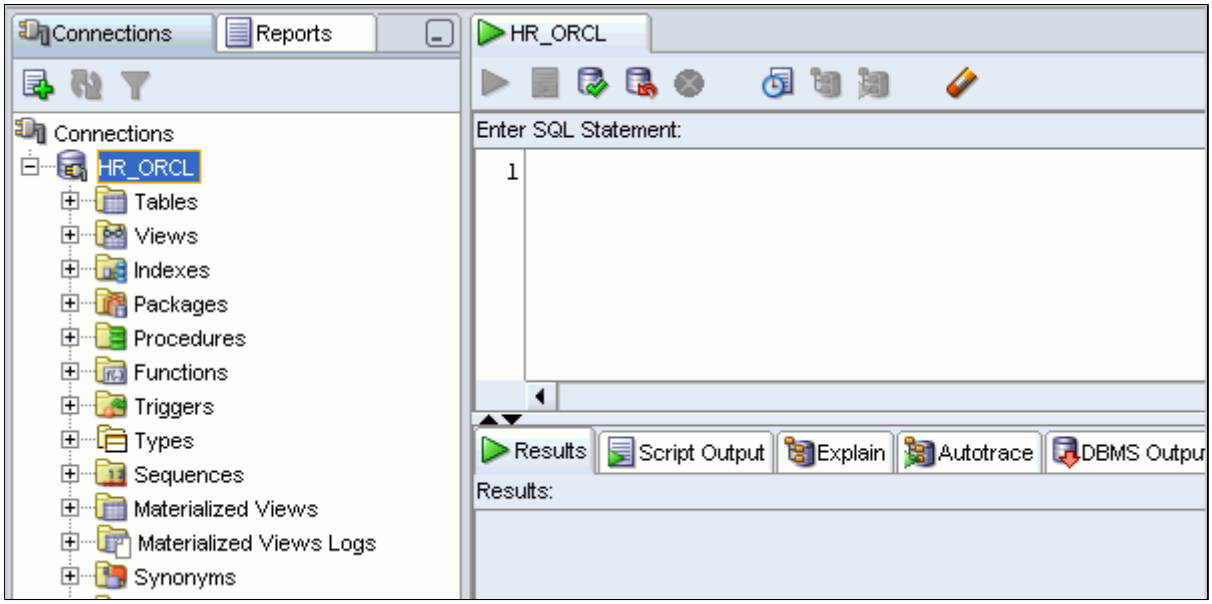
4. The status of the connection was tested successfully. The connection was not saved however. To save the connection, click **Connect**.



5. The connection was saved and you see the database in the list. Expand **HR_ORCL**.



6. When a connection is opened, a SQL Worksheet is opened automatically. The SQL Worksheet allows you to execute SQL against the connection you just created.

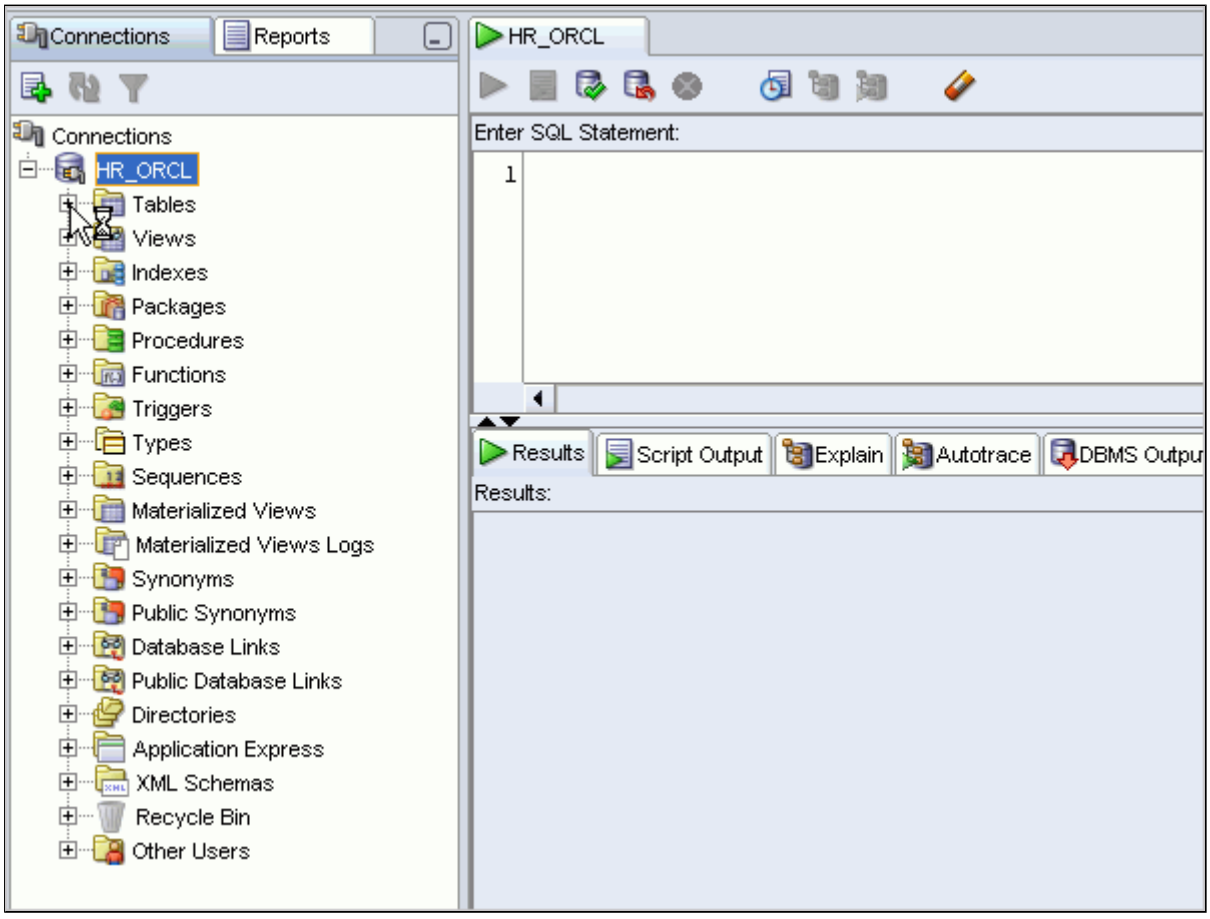


Back to Topic List

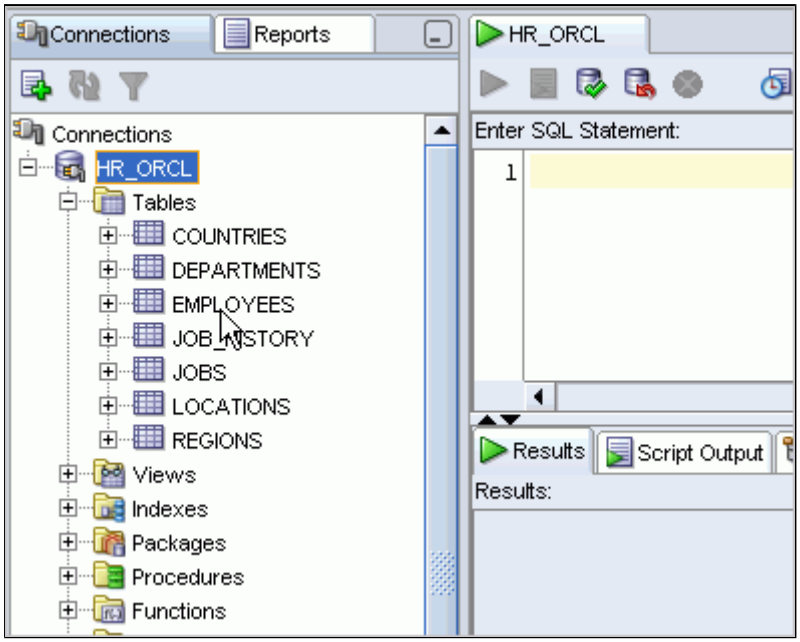
Browsing Your Database

The Connections Navigator in Oracle SQL Developer allows you to browse and edit database objects. This tutorial creates and debugs PL/SQL and uses a selection of tables from the HR schema. In this topic, you review the tables you will use later in the tutorial. Perform the following steps:

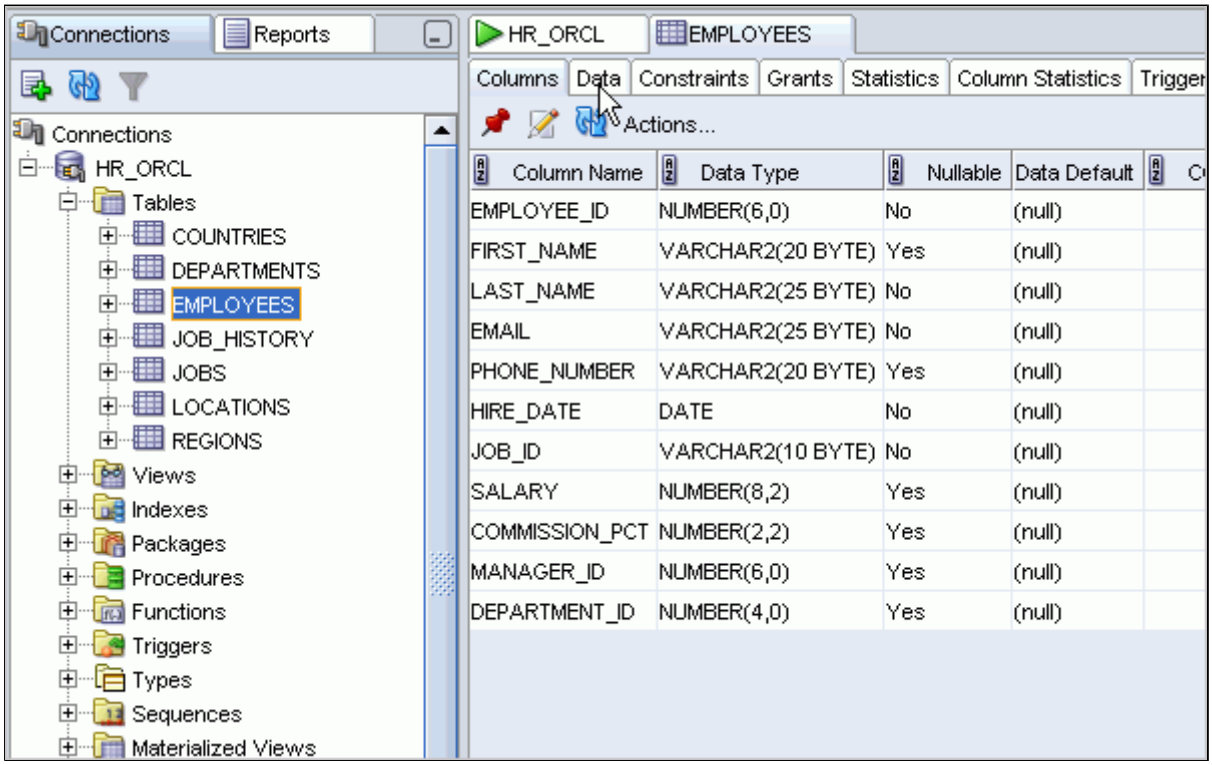
1. Expand the **Tables** node.



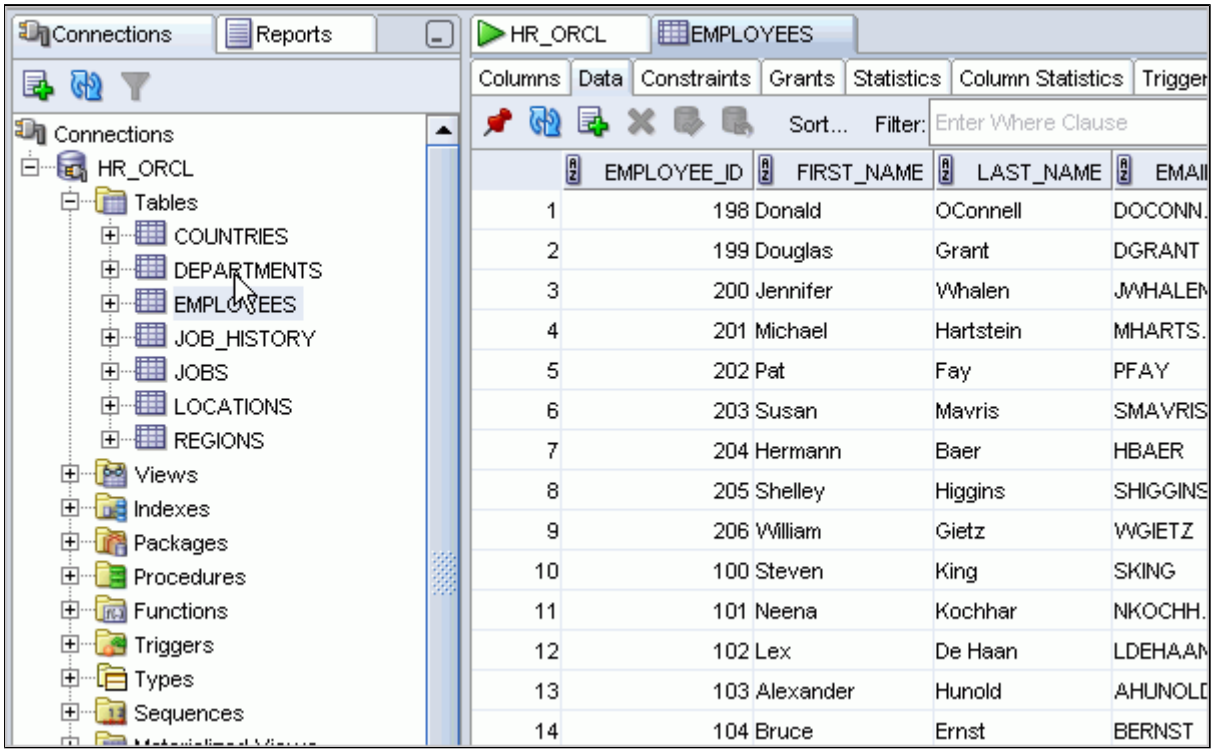
2. Click the **EMPLOYEES** table to view the table definition.



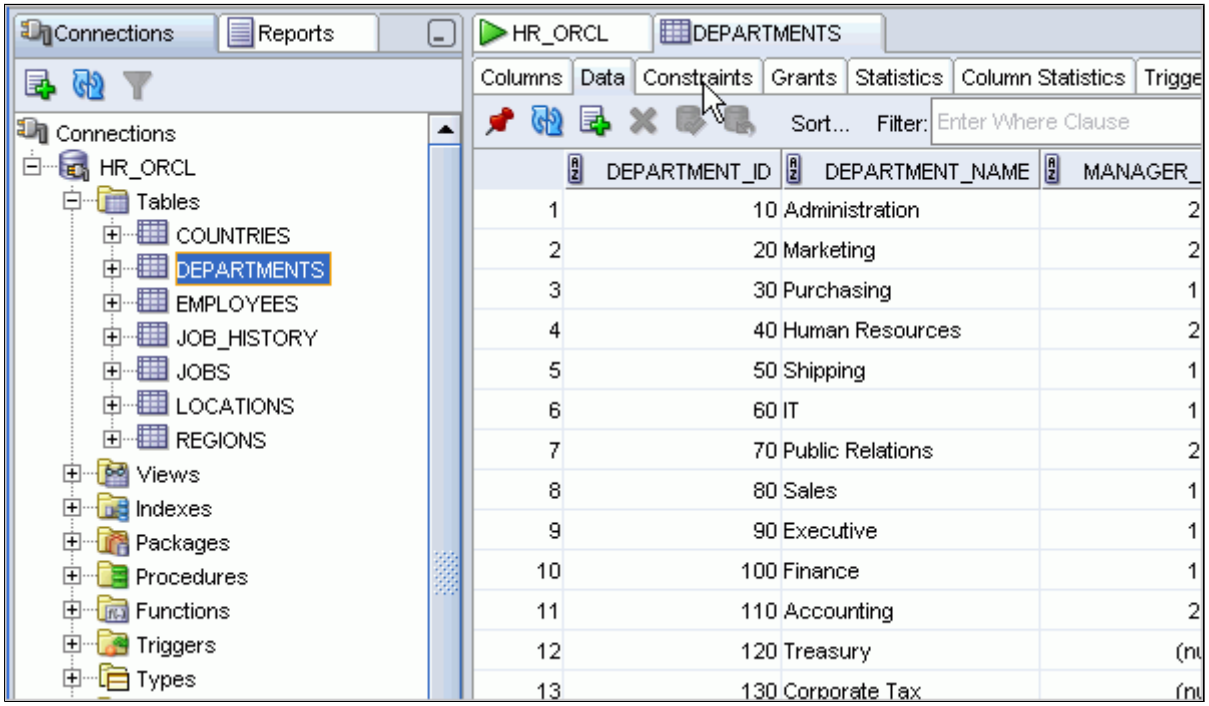
3. To see the data, click the **Data** tab.



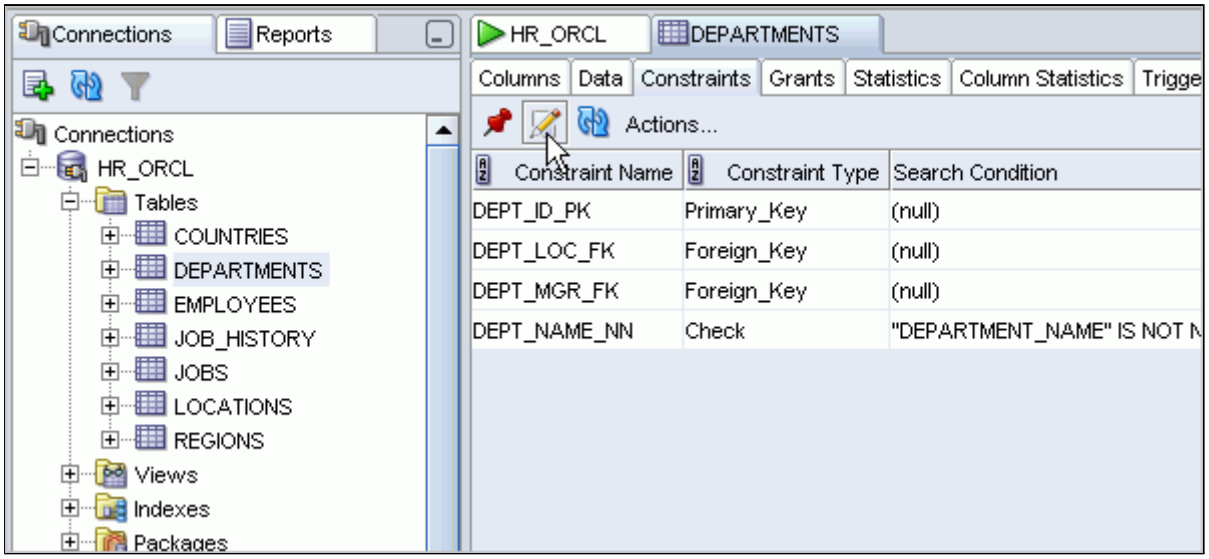
4. Click the **DEPARTMENTS** table in the navigator.



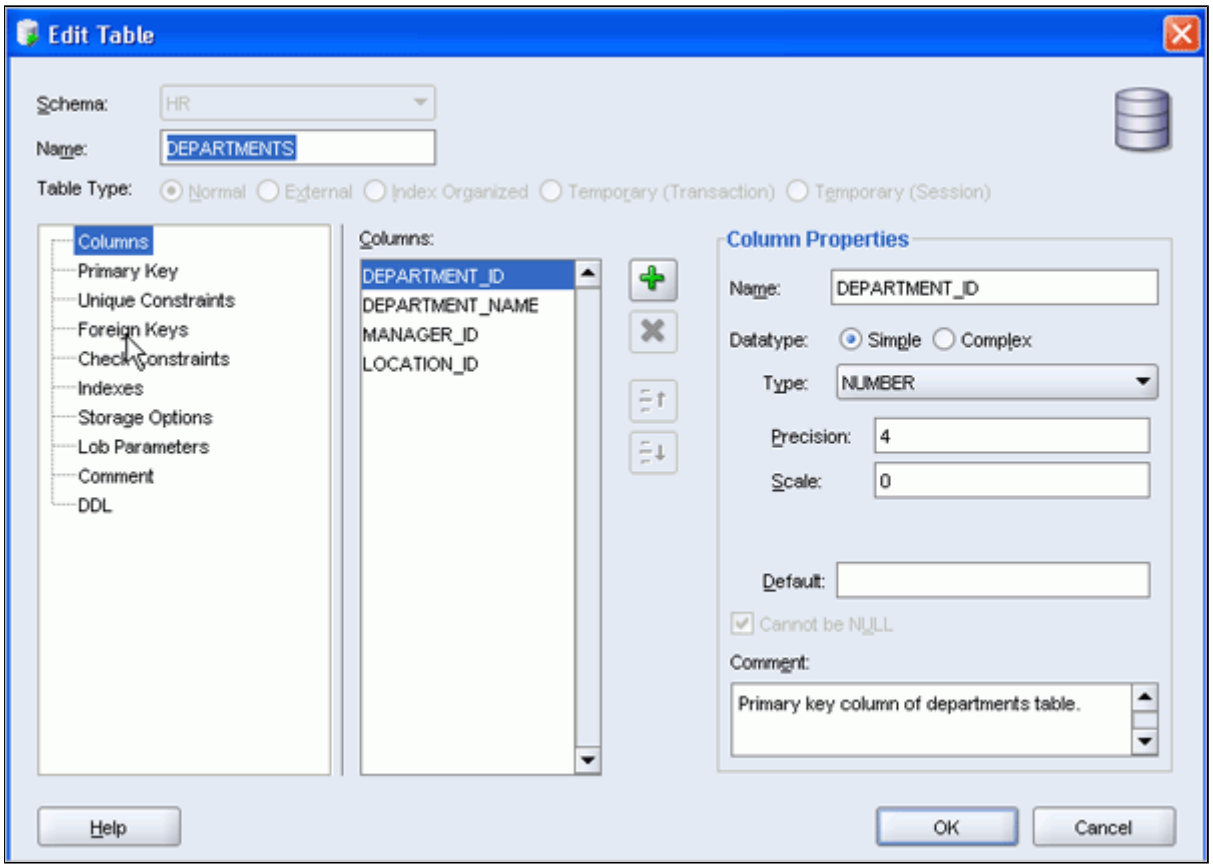
5. There are a number of constraints for the DEPARTMENTS table. When the table definition displays, select the **Constraints** tab.



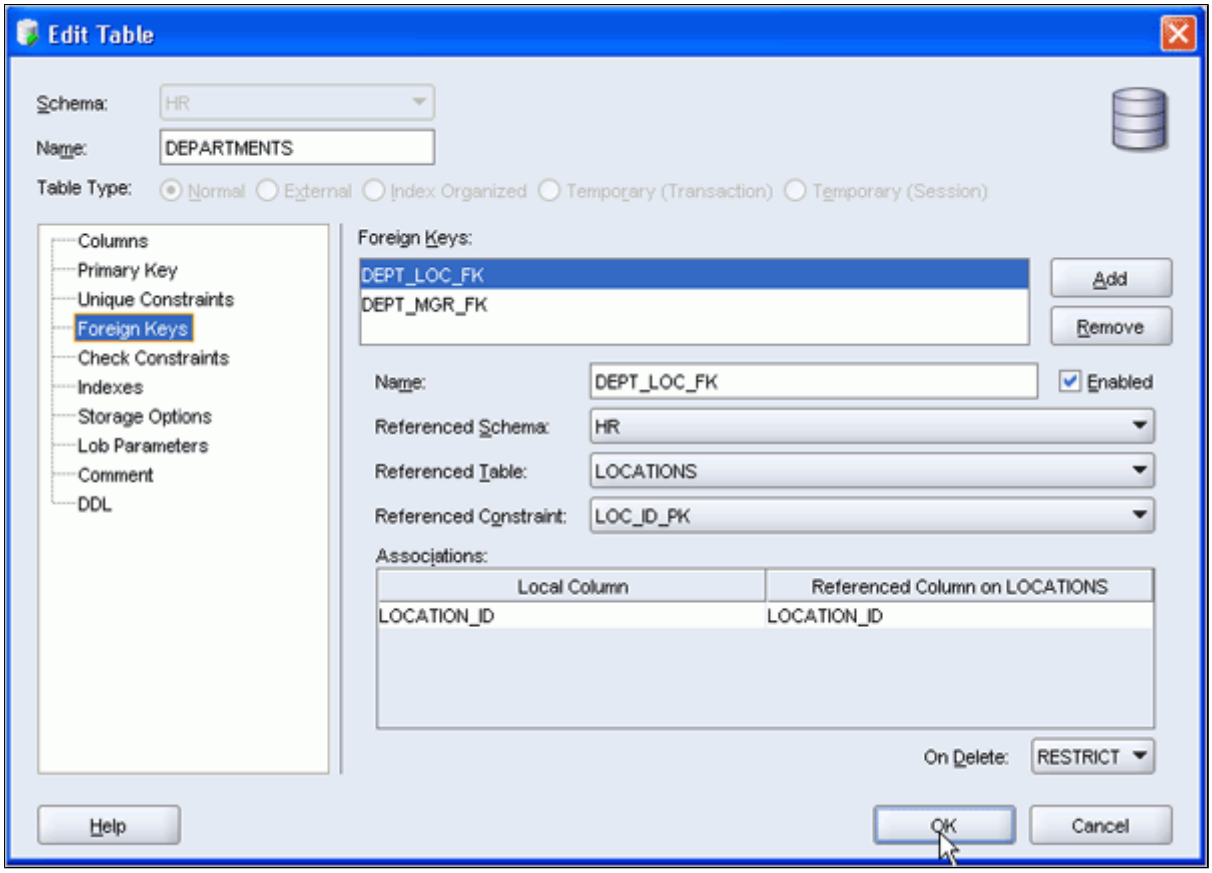
6. Click the **Edit** icon.



7. The dialog has a number of tabs, select the **Foreign Keys** tab.



Review the Foreign Keys. Then click **OK**.



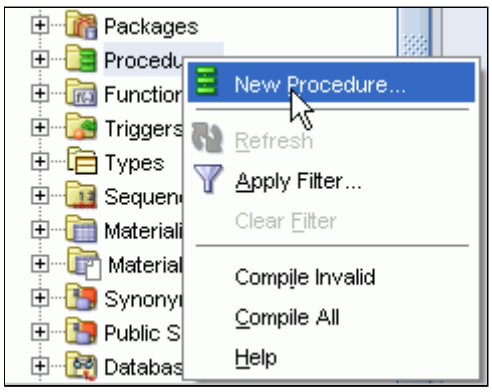
8. Verify that the JOBS and LOCATIONS tables exist, and have data, by selecting each in the Navigator in turn and reviewing the definitions and data.

Back to Topic List

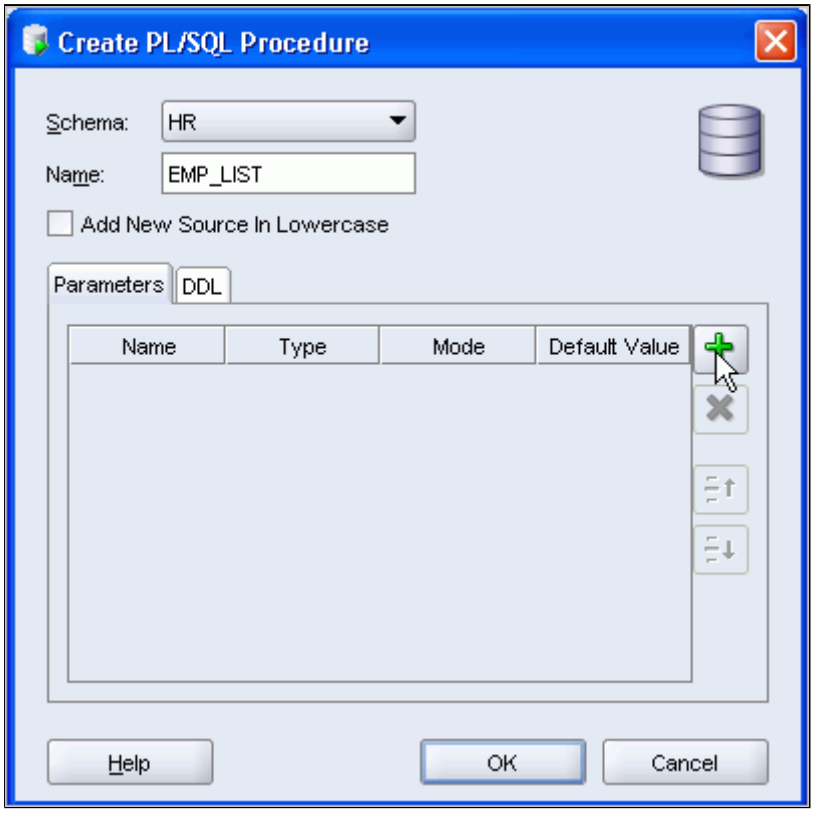
Creating and Compiling a PL/SQL Procedure

In this topic you create, edit and compile a PL/SQL procedure. Perform the following steps:

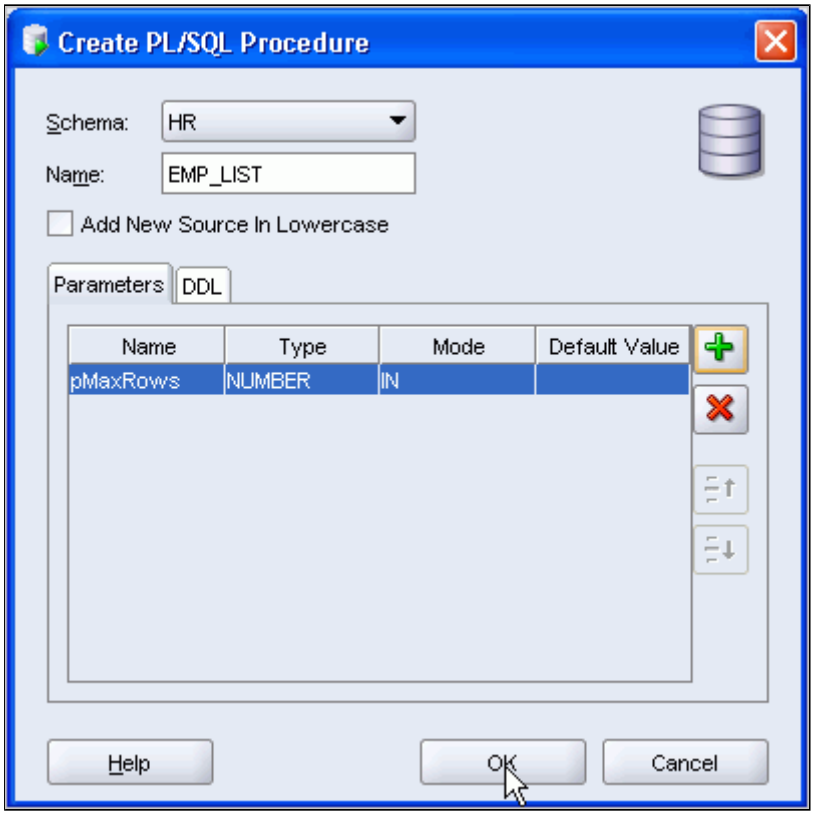
1. Right-click on the **Procedures** node in the Connections Navigator, to invoke the context menu, and select **Create PROCEDURE**.



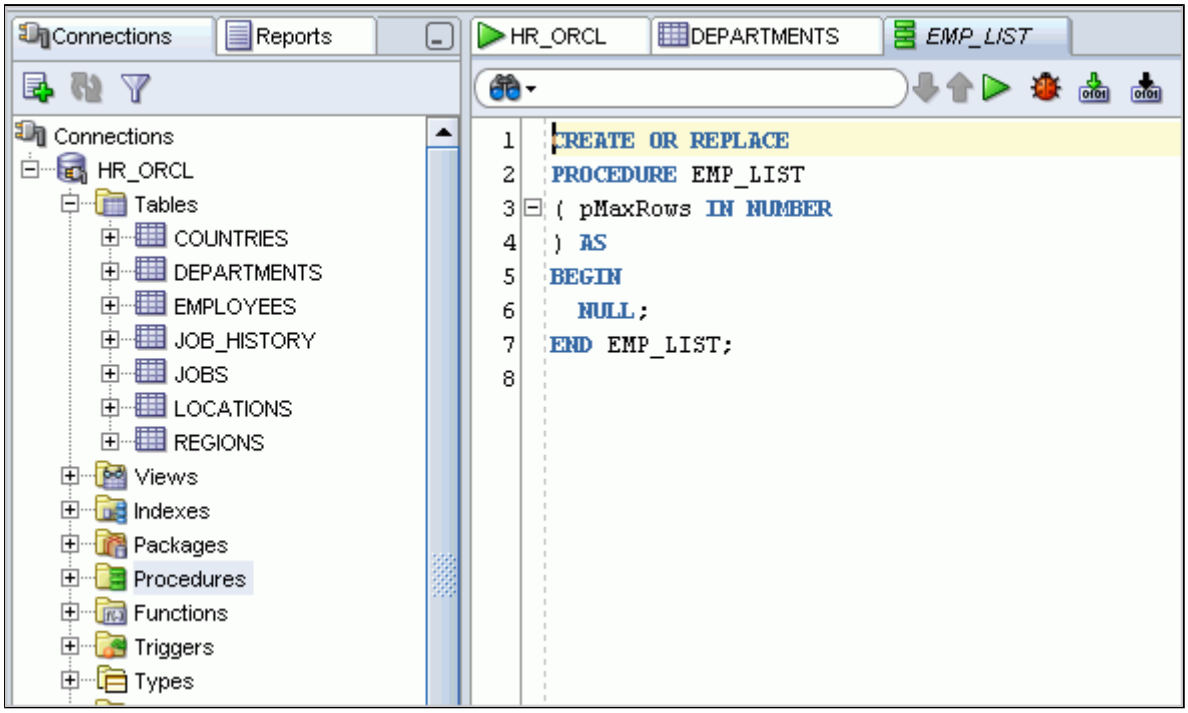
2. Enter **EMP_LIST** as the procedure name. Then click the **+** to add a Parameter.



3. Double-click on param to allow you to change the value to **pMaxRows** and then change VARCHAR2 to **NUMBER**. Make sure you press enter before you click **OK**.



4. The skeleton of the procedure with the parameter specified is displayed.



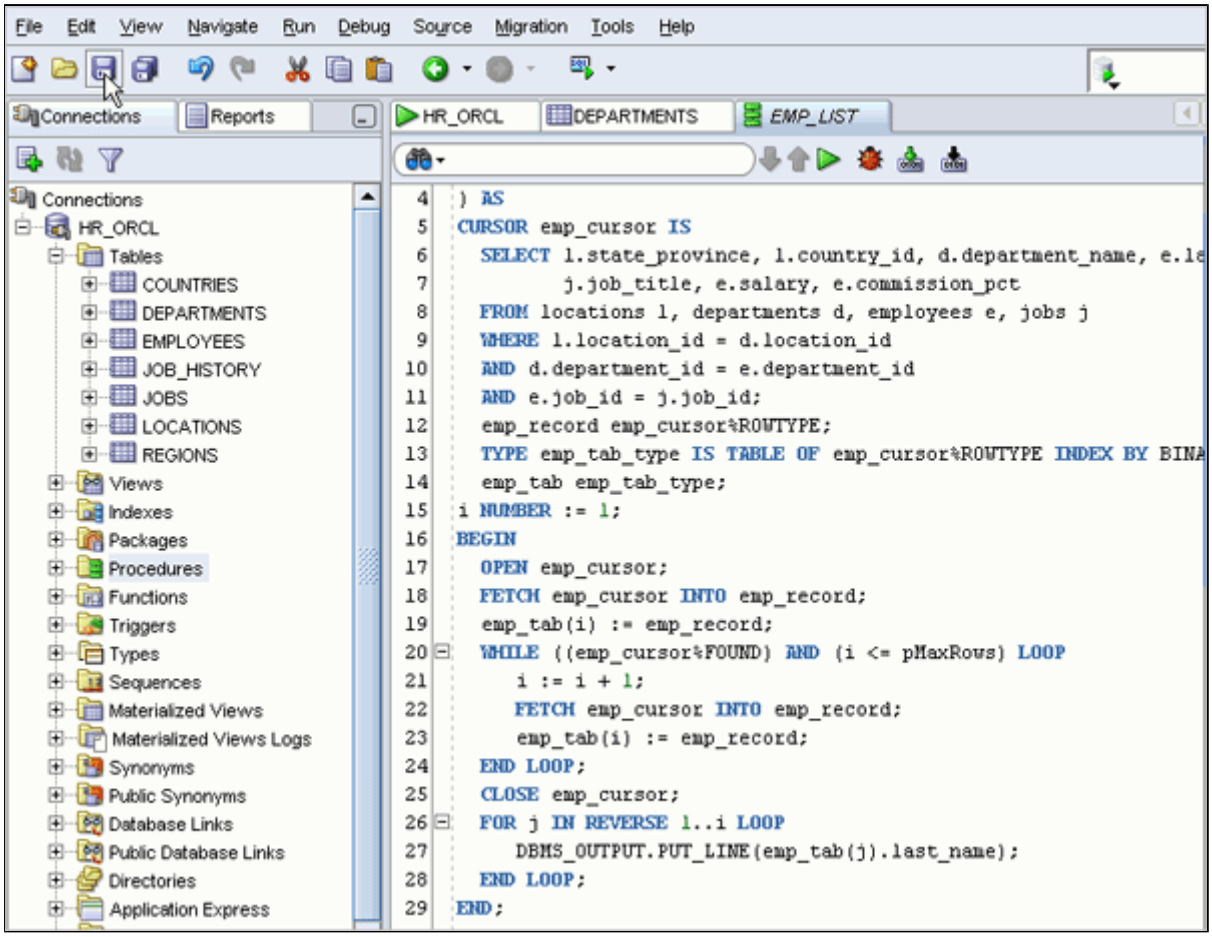
5. Replace the following PL/SQL:

```
BEGIN
  NULL;
END;
```

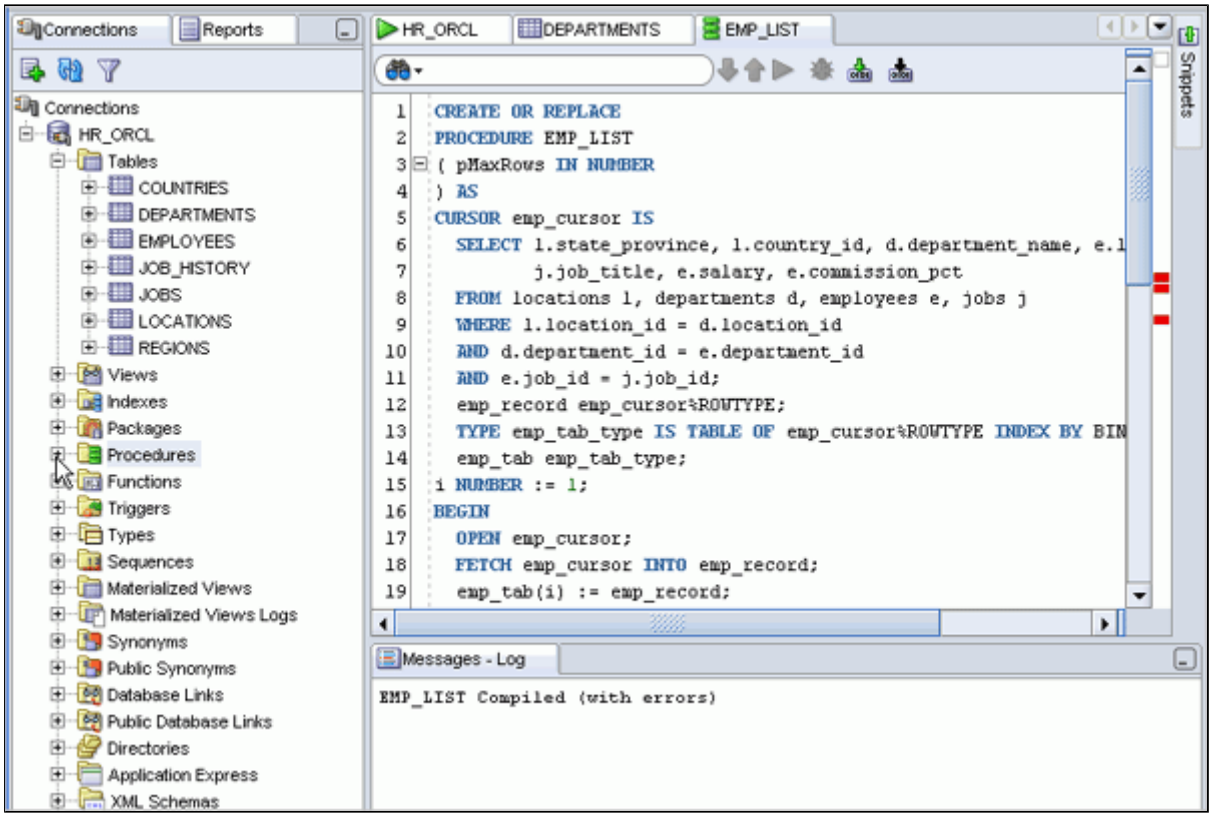
With the following code:
(You will find the code in the file **emp_cursor.sql** in the directory **PLSQL****files**)

```
CURSOR emp_cursor IS
  SELECT l.state_province, l.country_id, d.department_name, e.last_name,
         j.job_title, e.salary, e.commission_pct
  FROM locations l, departments d, employees e, jobs j
  WHERE l.location_id = d.location_id
  AND d.department_id = e.department_id
  AND e.job_id = j.job_id;
  emp_record emp_cursor%ROWTYPE;
  TYPE emp_tab_type IS TABLE OF emp_cursor%ROWTYPE INDEX BY BINARY_INTEGER;
  emp_tab emp_tab_type;
i NUMBER := 1;
BEGIN
  OPEN emp_cursor;
  FETCH emp_cursor INTO emp_record;
  emp_tab(i) := emp_record;
  WHILE ((emp_cursor%FOUND) AND (i <= pMaxRows) LOOP
     i := i + 1;
     FETCH emp_cursor INTO emp_record;
     emp_tab(i) := emp_record;
  END LOOP;
  CLOSE emp_cursor;
  FOR j IN REVERSE 1..i LOOP
     DBMS_OUTPUT.PUT_LINE(emp_tab(j).last_name);
  END LOOP;
END;
```

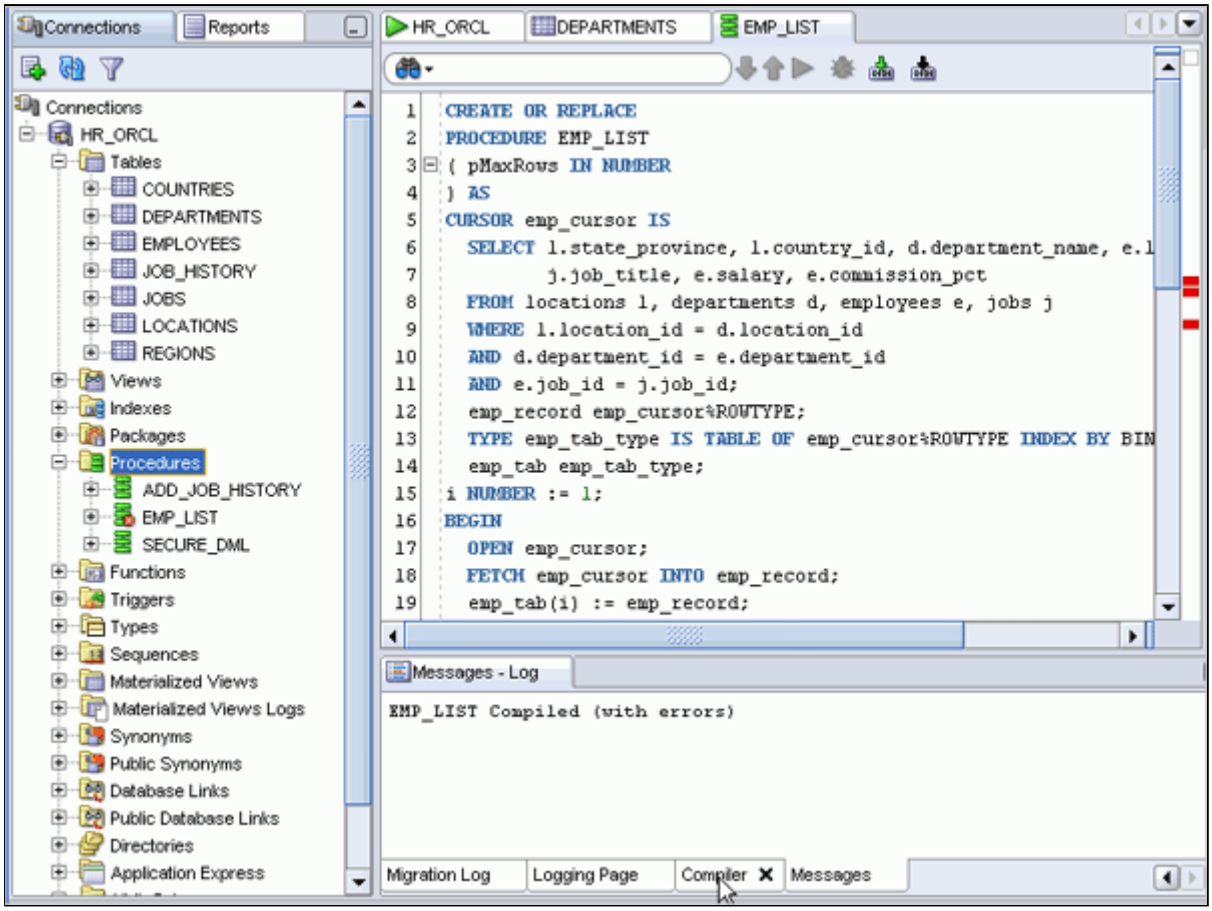
Notice how the reserved words are formatted by Oracle SQL Developer. To format the code further, right-click to invoke the context menuand select Format SQL...
Compile the PL/SQL subprogram by clicking the Save button in the toolbar.



6. Expand **Procedures** in the navigator.

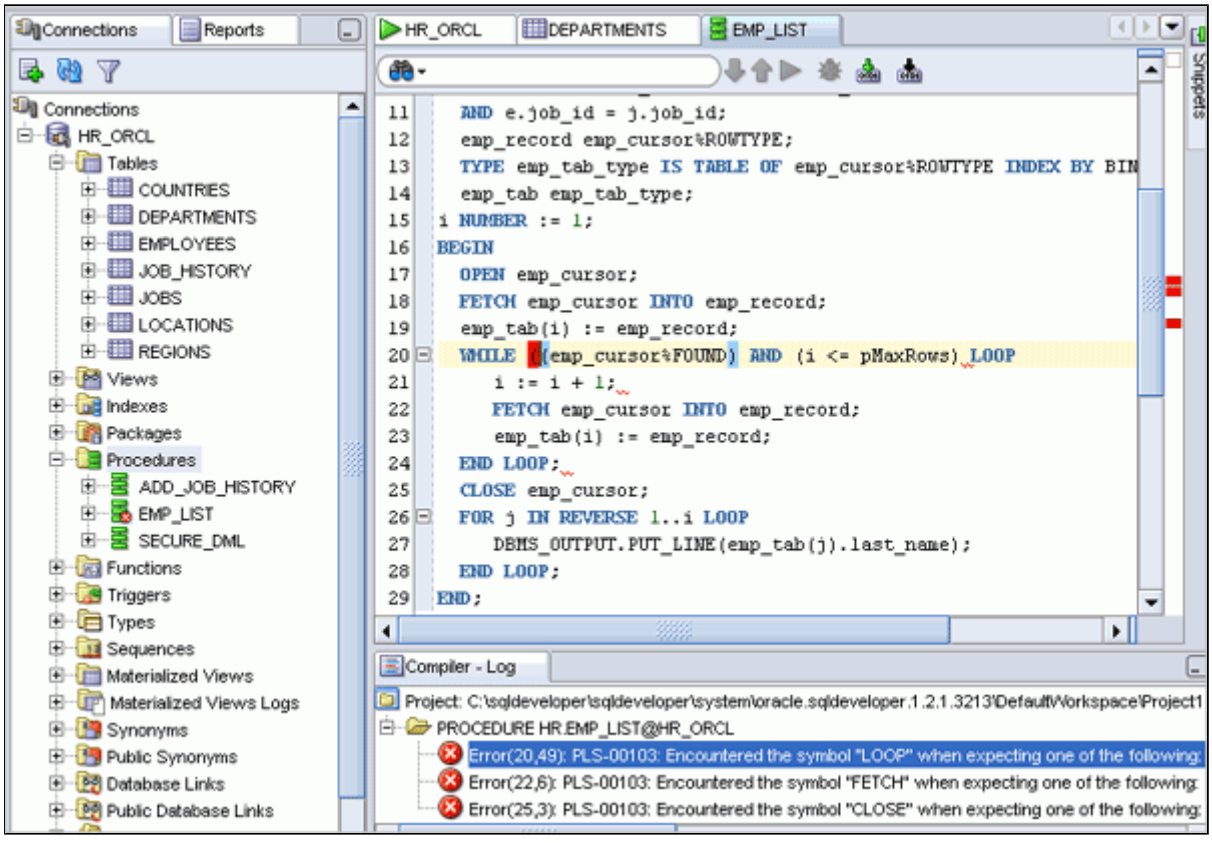


7. Note that when an invalid PL/SQL subprogram is detected by Oracle SQL Developer, the status is indicated with a red X over the icon for the subprogram in the Connections Navigator.

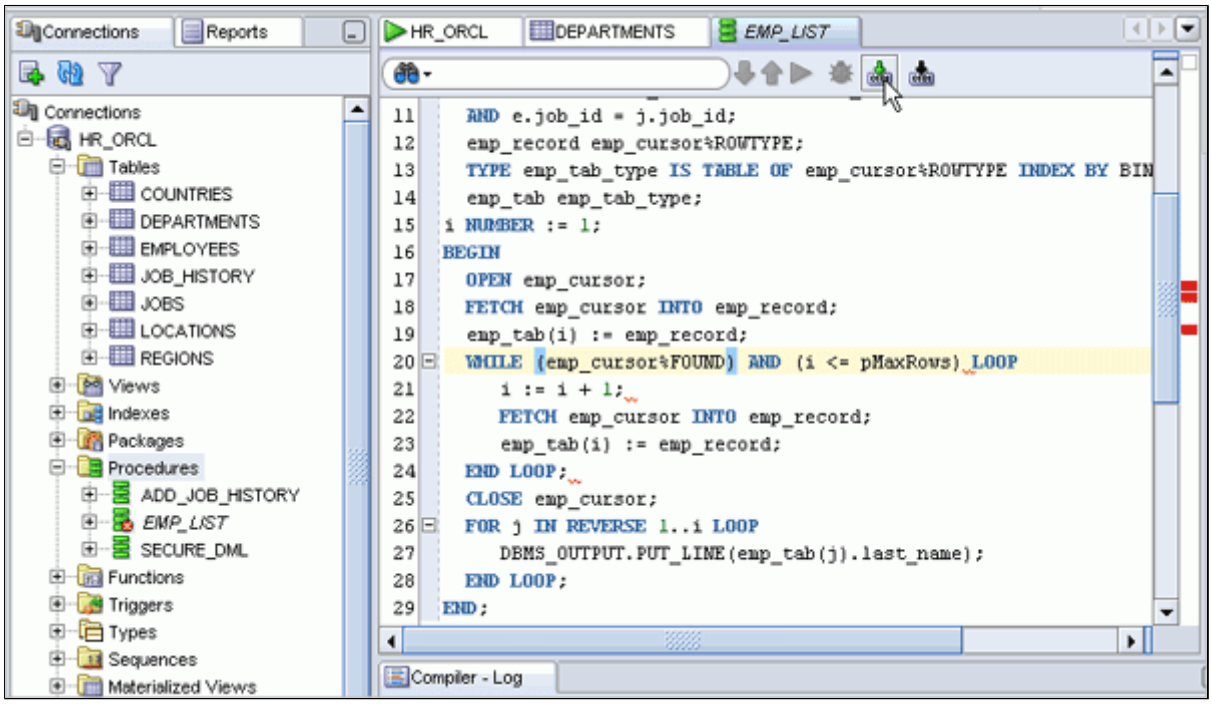


Compilation errors are shown in the log window. You can navigate to the line reported in the error by simply double-clicking on the error. Oracle SQL Developer also displays errors and hints in the right hand gutter. If you hover over each of the red bars in the gutter, the error message displays

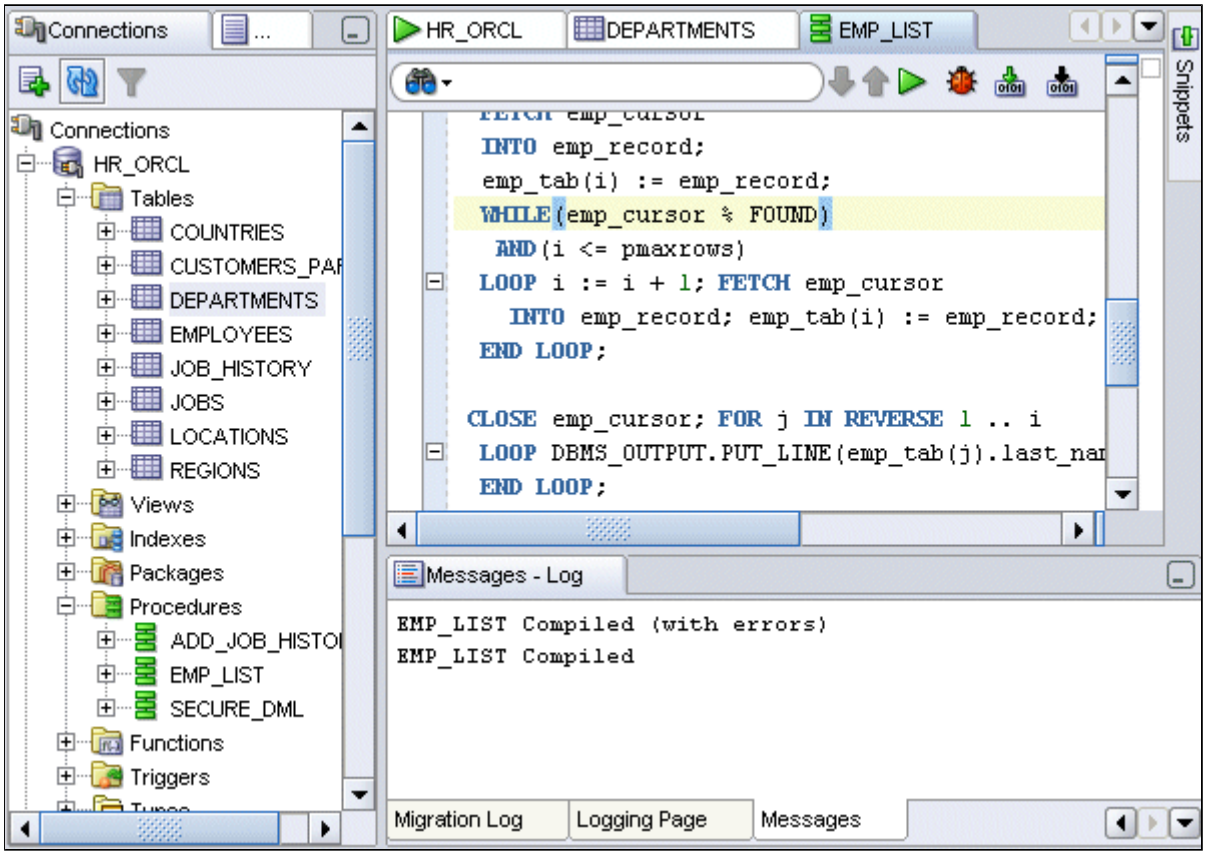
In this case, the error messages indicate that there is a formatting error in the LOOP statement. After reviewing the code further, you see an extra parenthesis in the WHILE statement. Delete the extra parenthesis.



8. Click the **Compile** icon.



9. The Procedure compiled successfully. You are now ready to run the procedure.

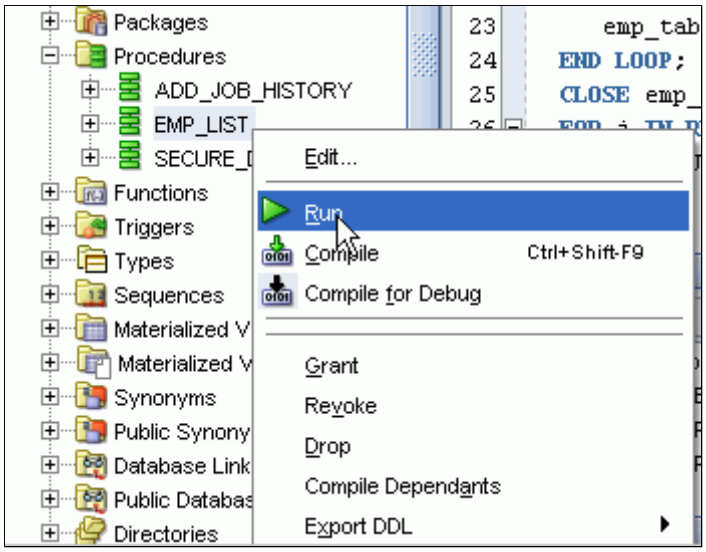


Back to Topic List

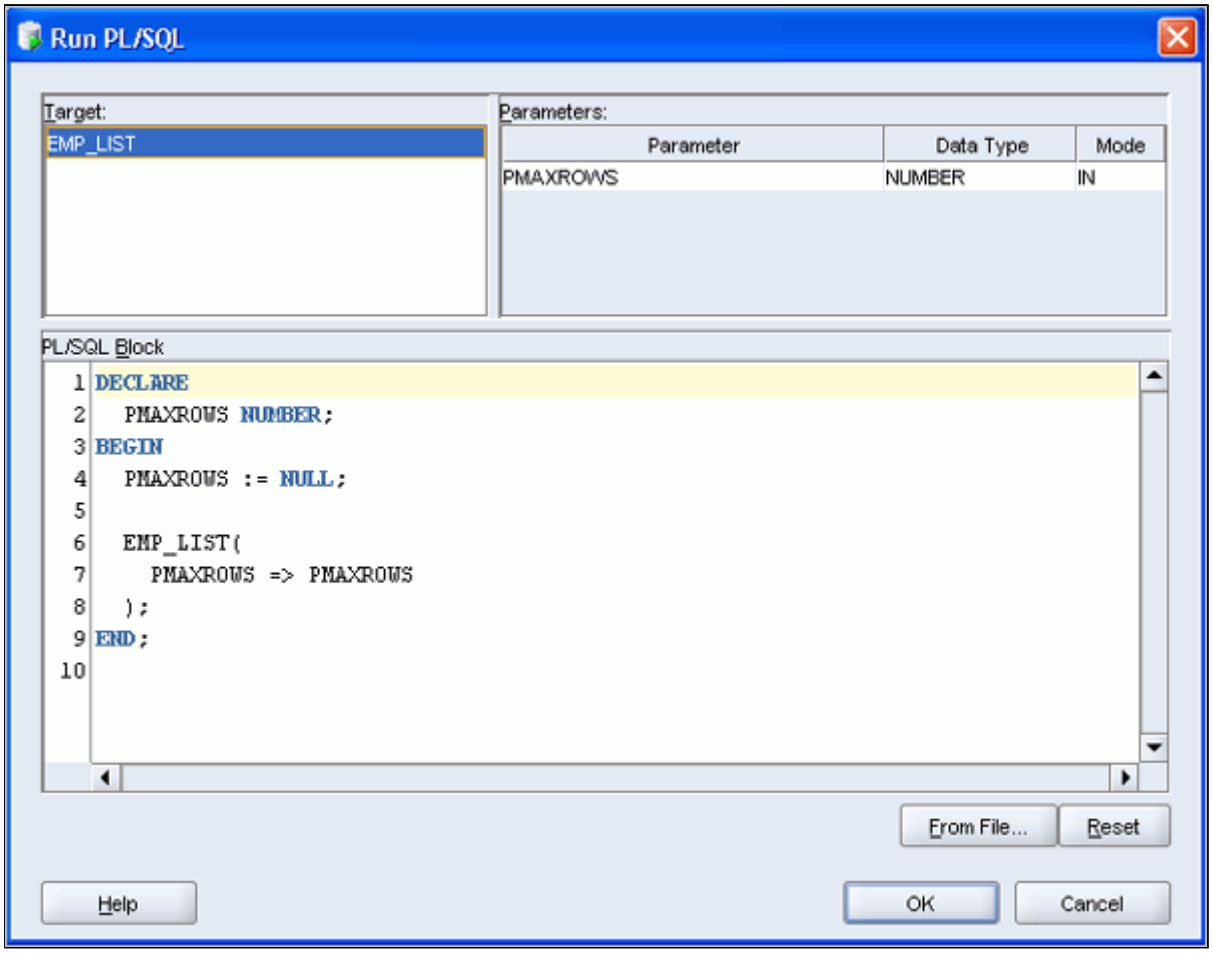
Running a PL/SQL Procedure

Once you have created and compiled a PL/SQL procedure, you can run it using Oracle SQL Developer. Perform the following steps:

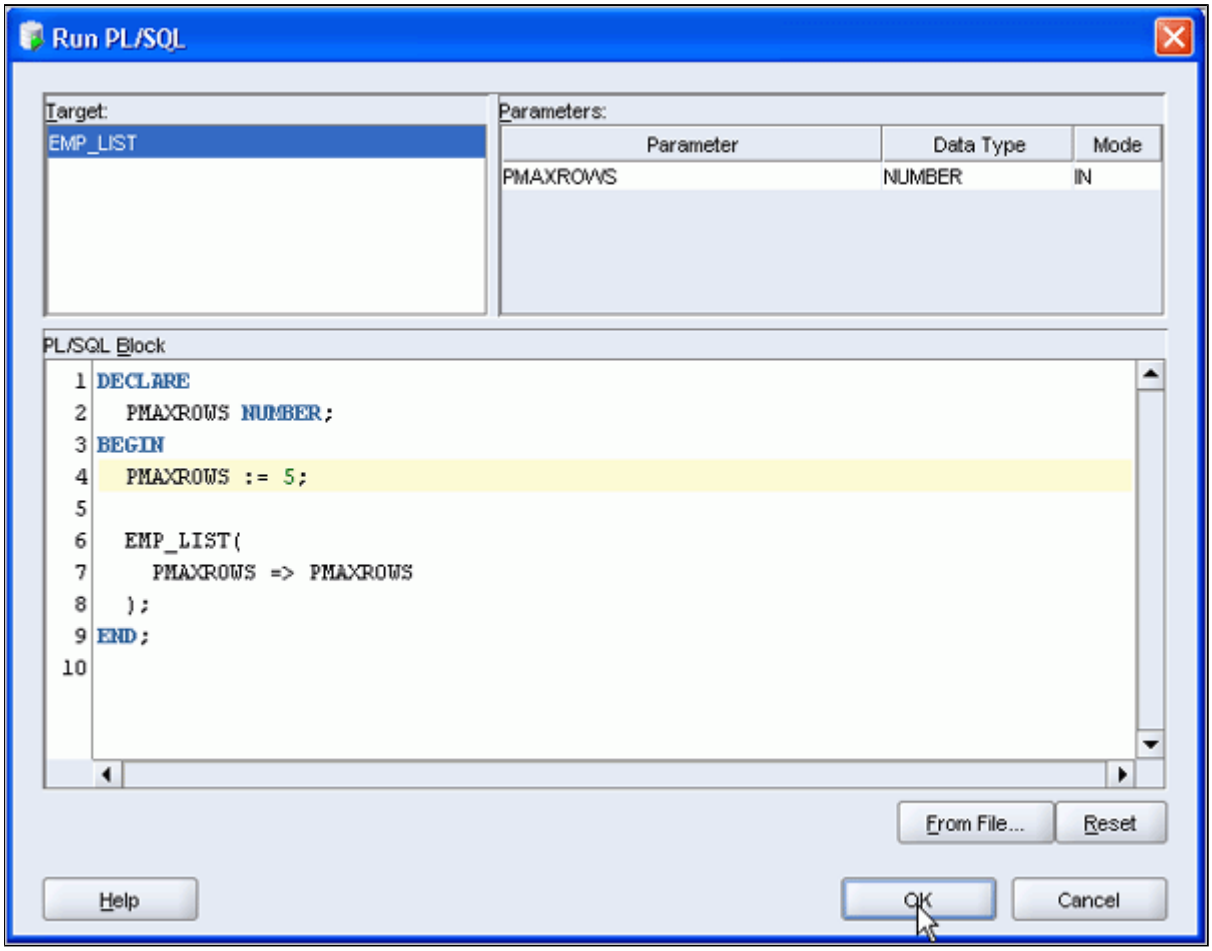
1. Right-click on **EMP_LIST** in the left navigator and select **Run**.



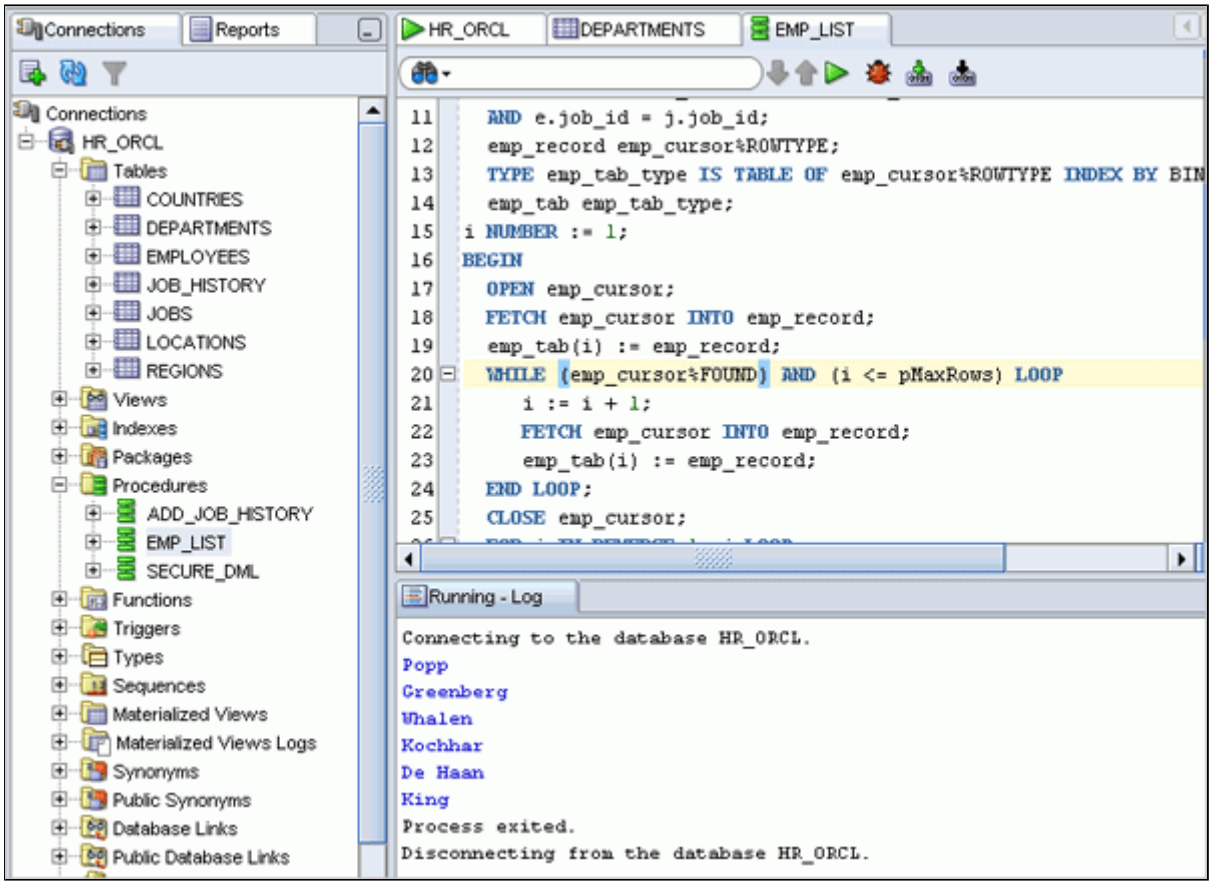
2. This invokes the Run PL/SQL dialog. The Run PL/SQL dialog allows you to select the target procedure or function to run (useful for packages) and displays a list of parameters for the selected target. In the PL/SQL block text area is generated code that Oracle SQL Developer uses to call the selected program. You can use this area to populate parameters to be passed to the program unit and to handle complex return types.



Change `PMAXROWS := NULL;` to `PMAXROWS := 5;` Then click **OK**.



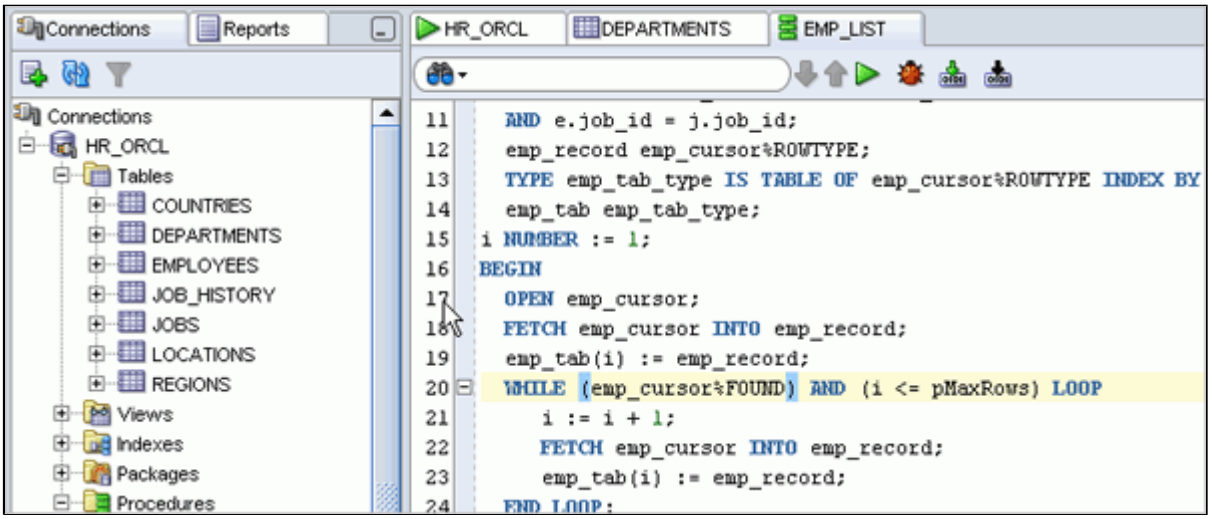
3. The results of the 5 rows returned are displayed in the log window.



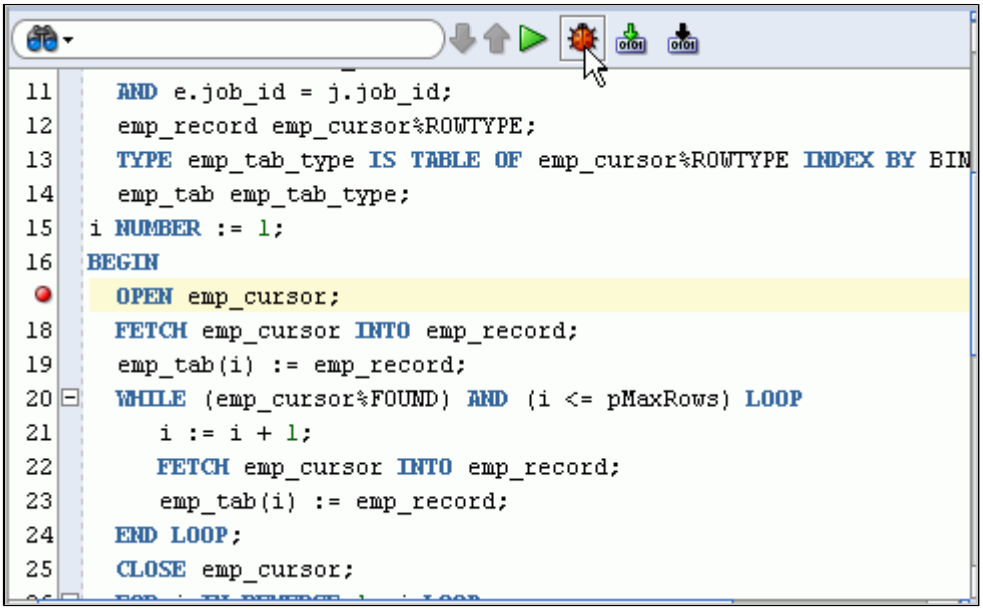
Debugging a PL/SQL Procedure

Oracle SQL Developer also supports PL/SQL debugging with Oracle databases. In this topic, you debug a PL/SQL Procedure, step through the code and modify a value at runtime. Perform the following steps:

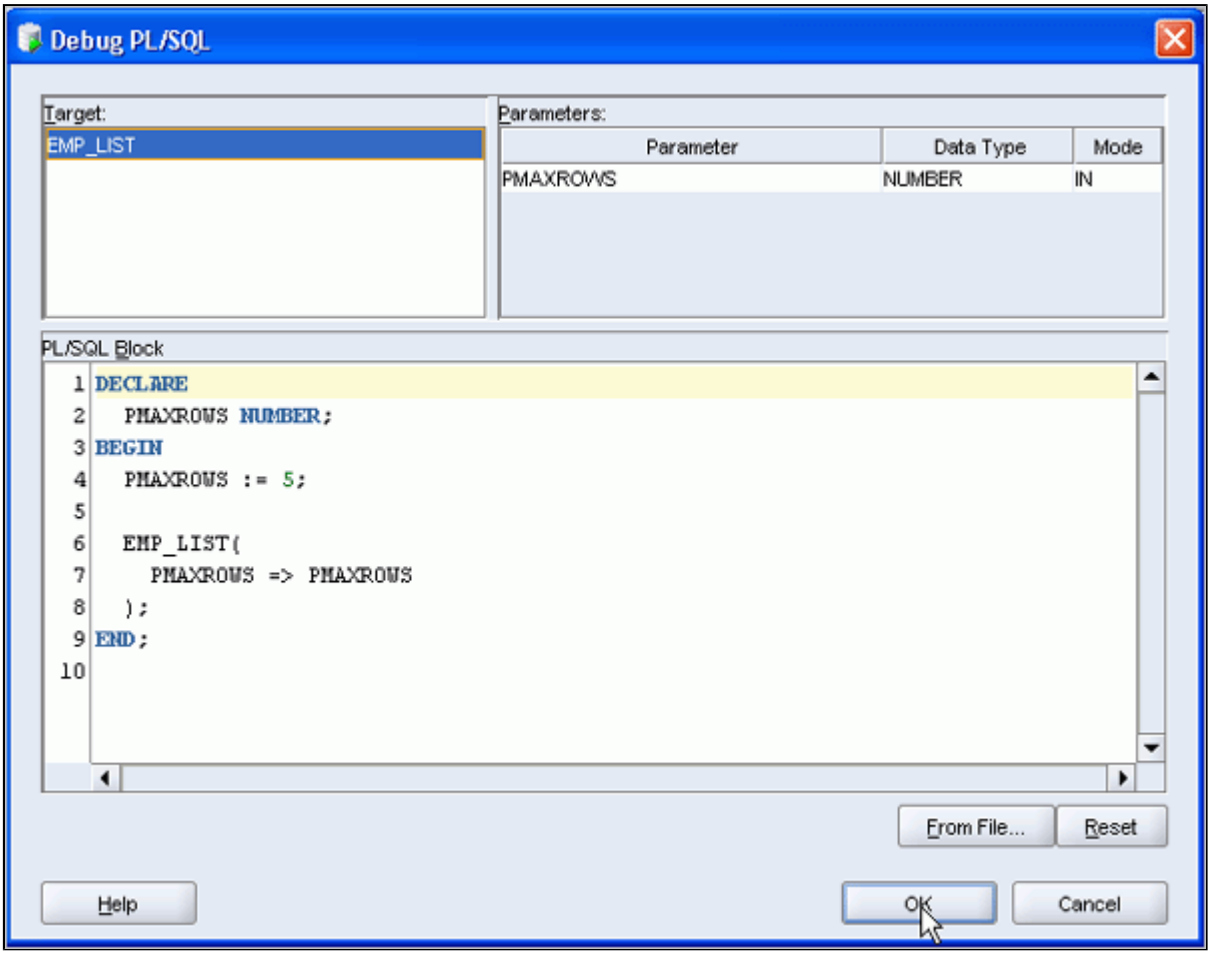
1. Set a breakpoint in the EMP_LIST procedure by clicking in the margin at the line with the **`OPEN emp_cursor;`** statement.



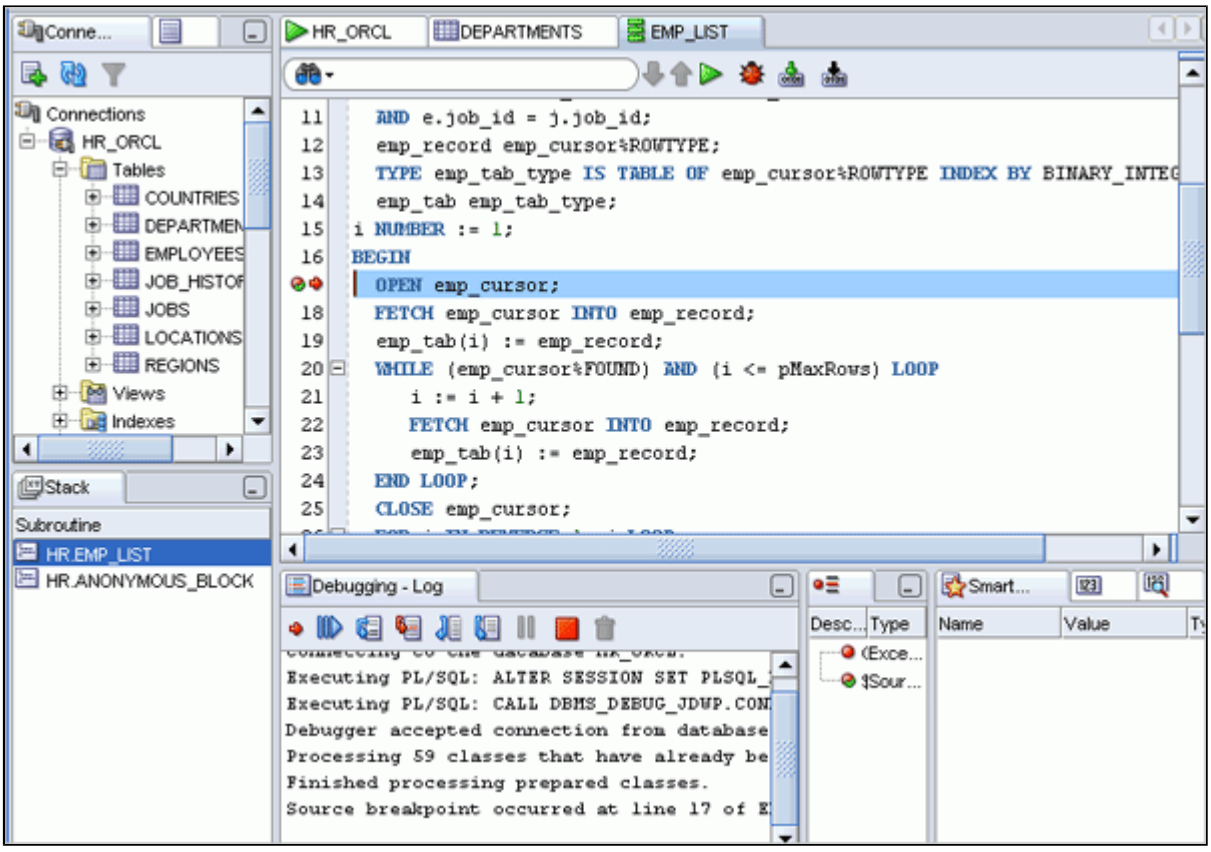
2. Click the **Debug** icon (ladybug).



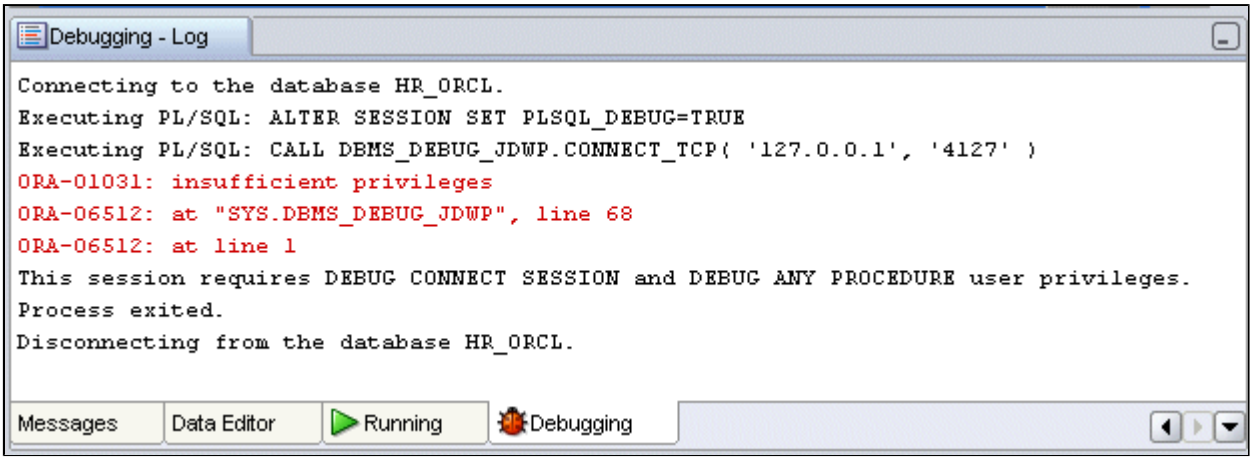
3. The Debug PL/SQL dialog should still show the value `PMAXROWS = 5;` Click **OK**.



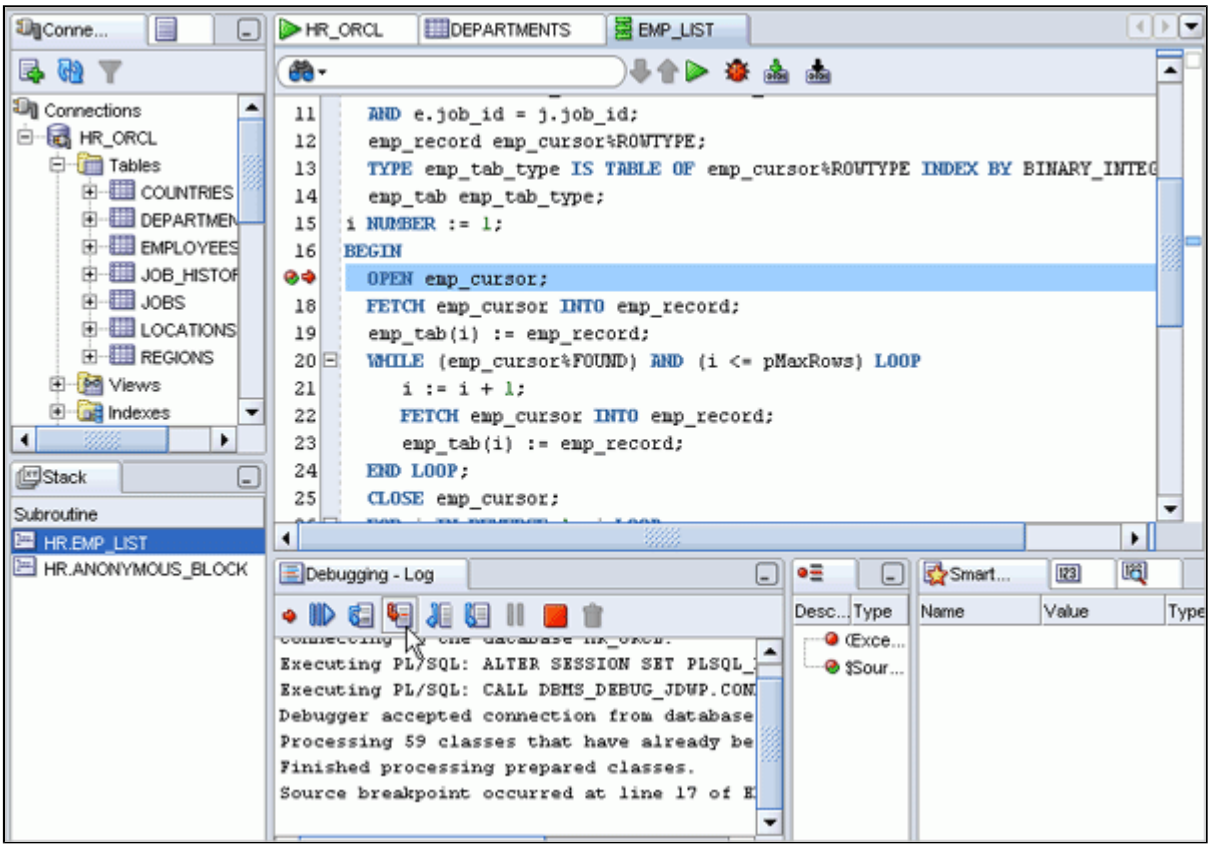
4. The debugger should halt at the line where you placed the breakpoint. You can now control the flow of execution, modify values of variables and perform other debugging functions.



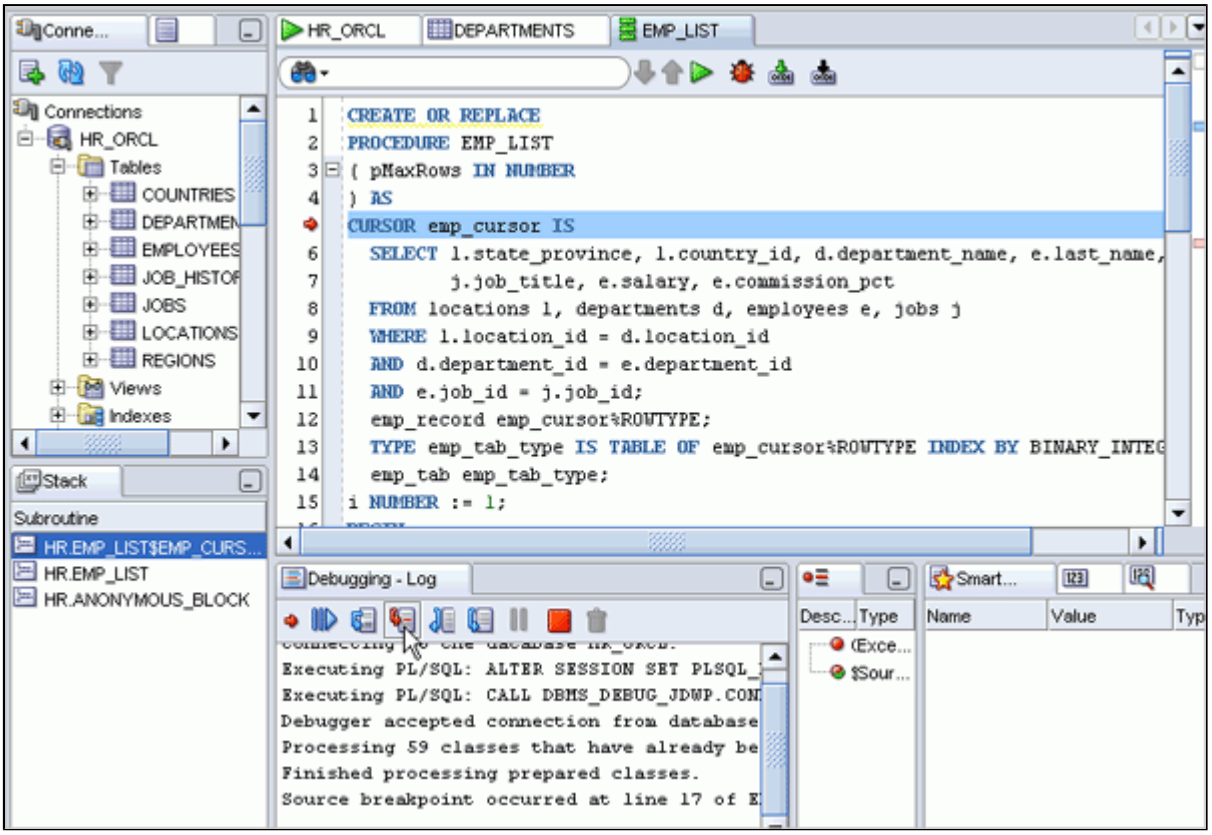
Note: If at this point you receive the error message "This session requires DEBUG CONNECT SESSION and DEBUG ANY PROCEDURE user privileges", you need to complete Step 4 of Prerequisites.



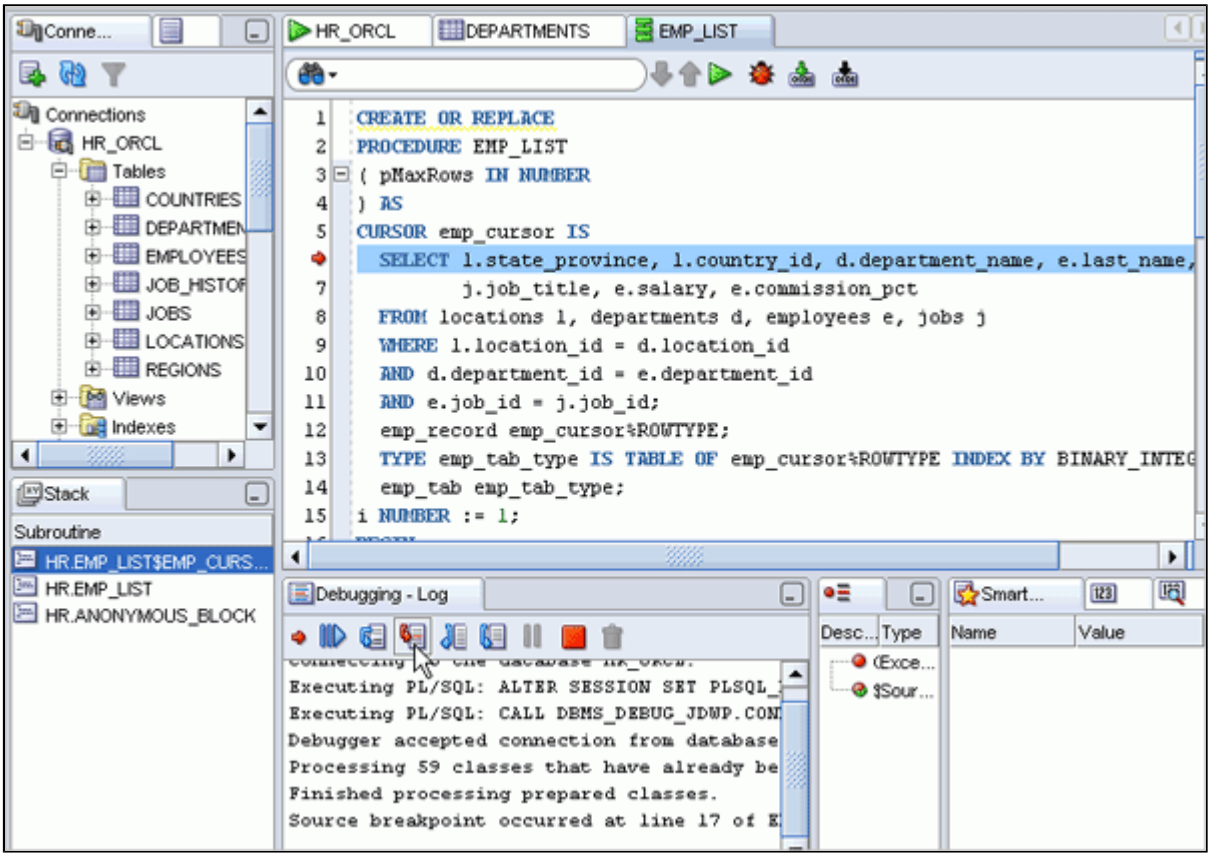
5. Click **Step Into** .



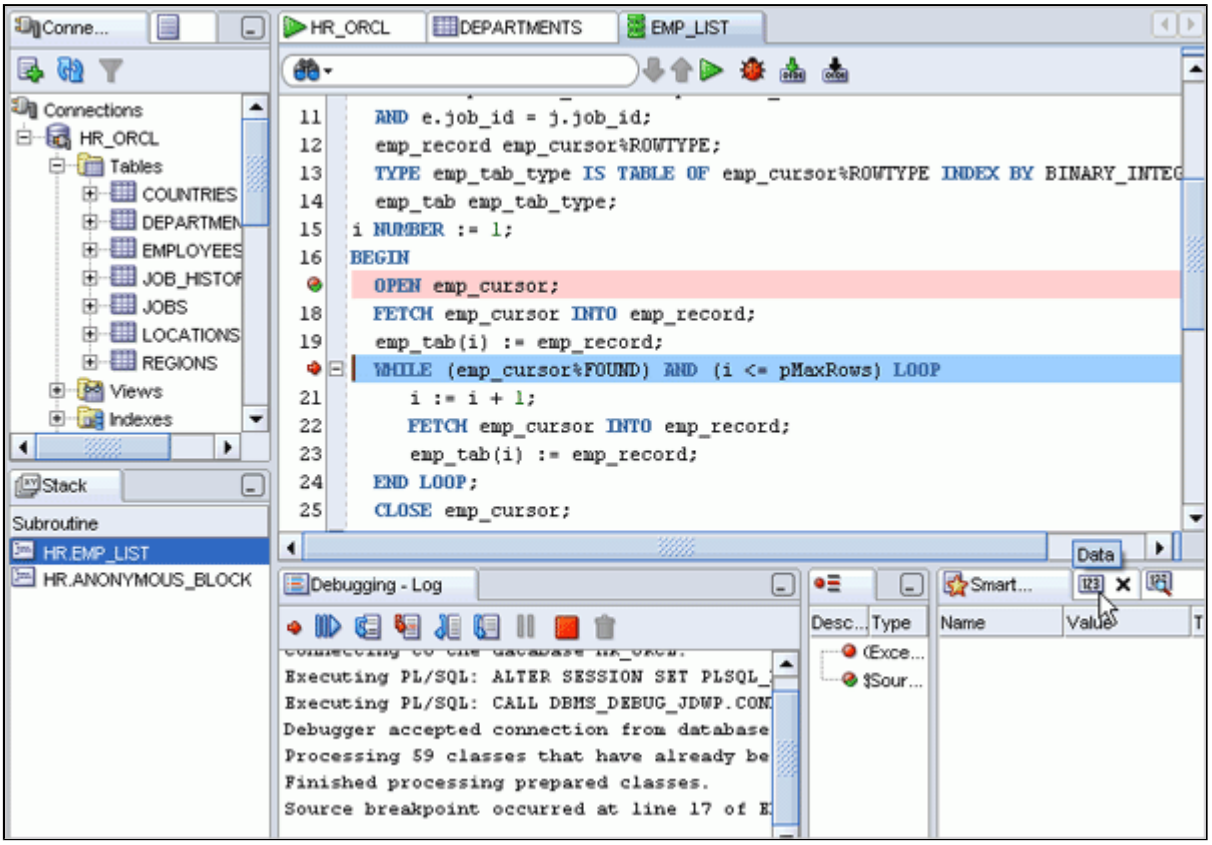
6. This takes you to the first line of the cursor. Click **Step Into** again.



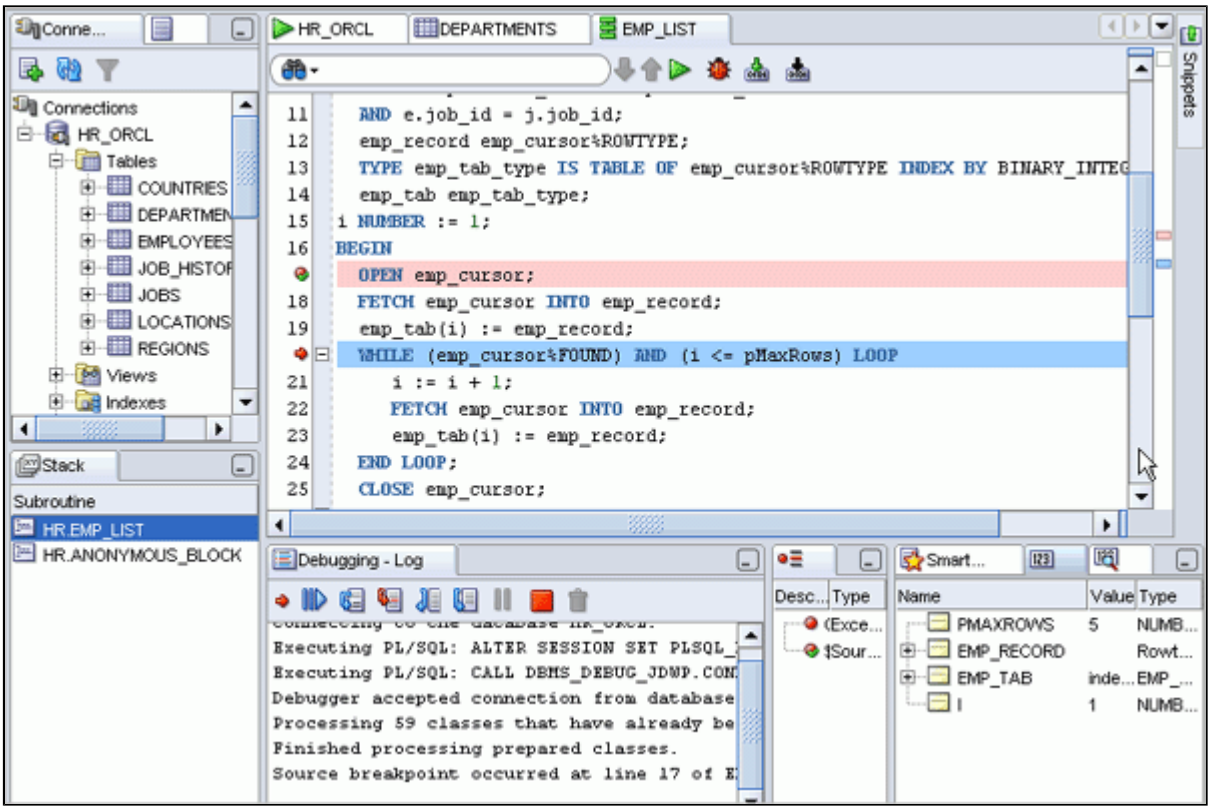
7. You should now be selecting the first row of the cursor. Click **Step Into** 3 more times.



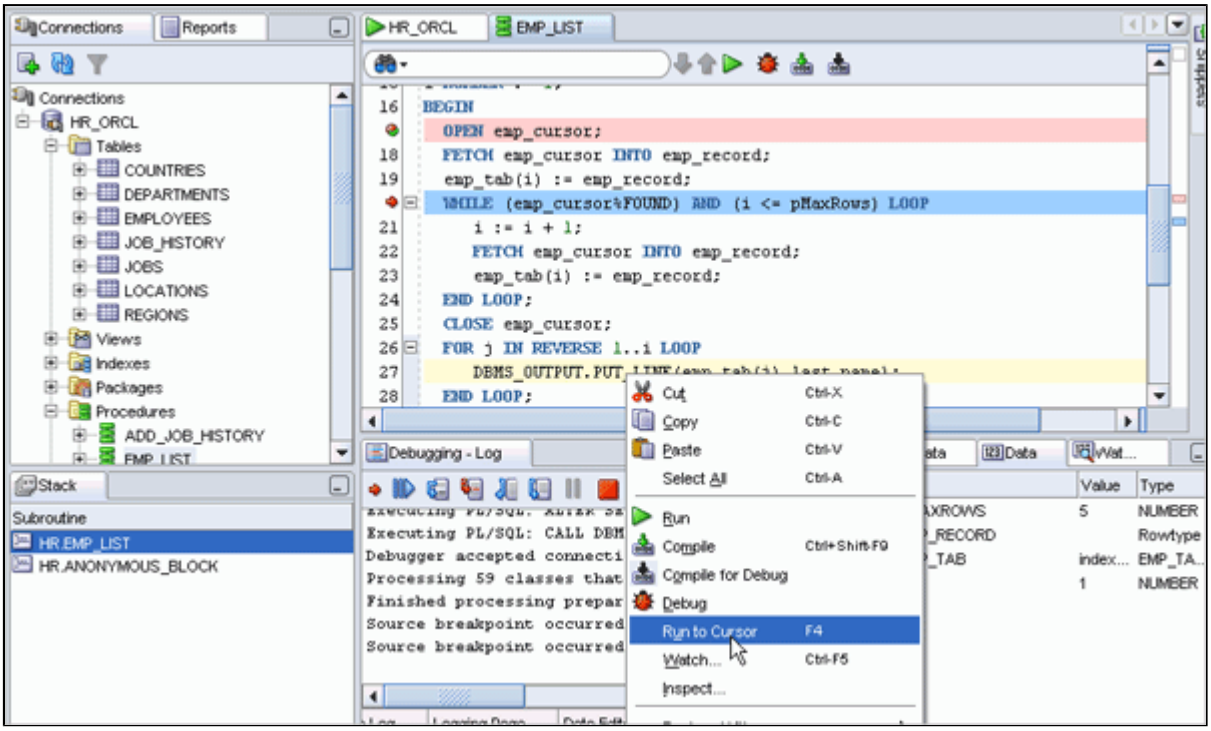
8. Click the **Data** tab.



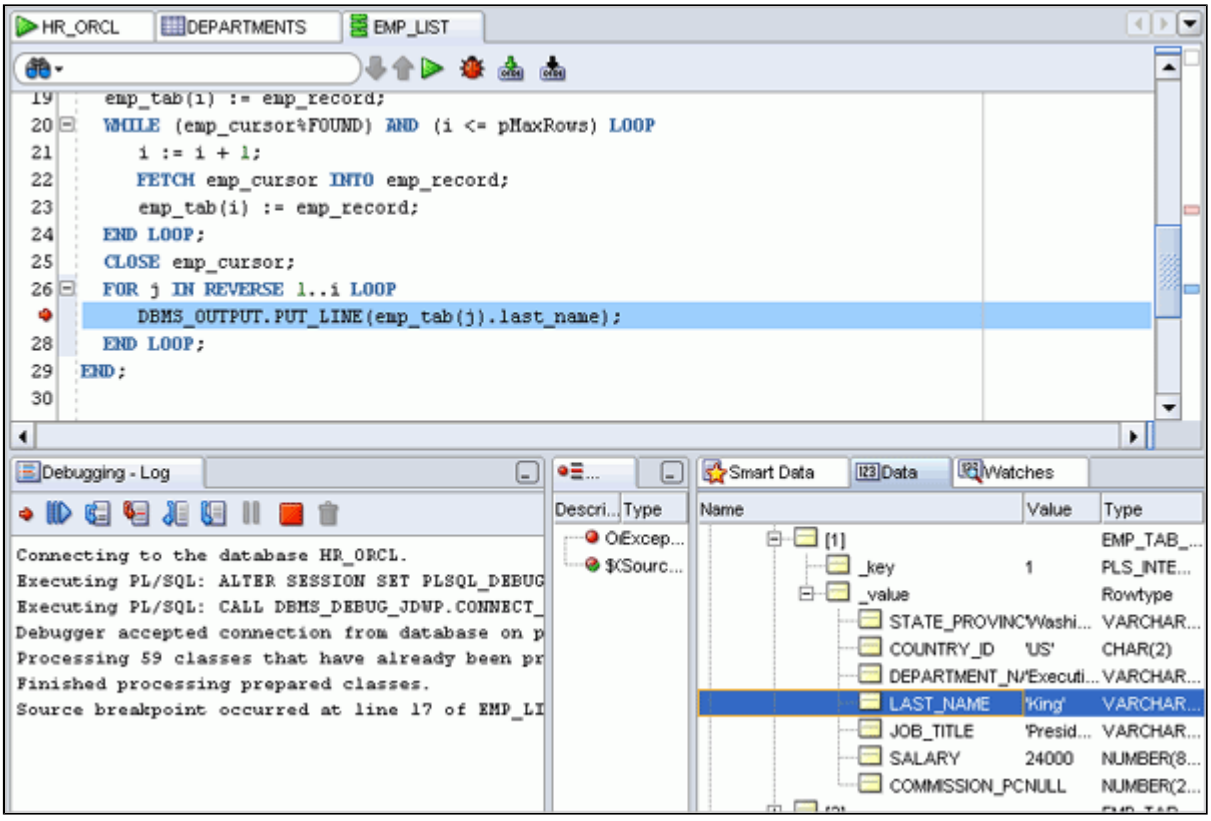
9. The **Data** window starts to show a limited list of variables which are used in the line of code that is about to be executed, and in the previously executed line.



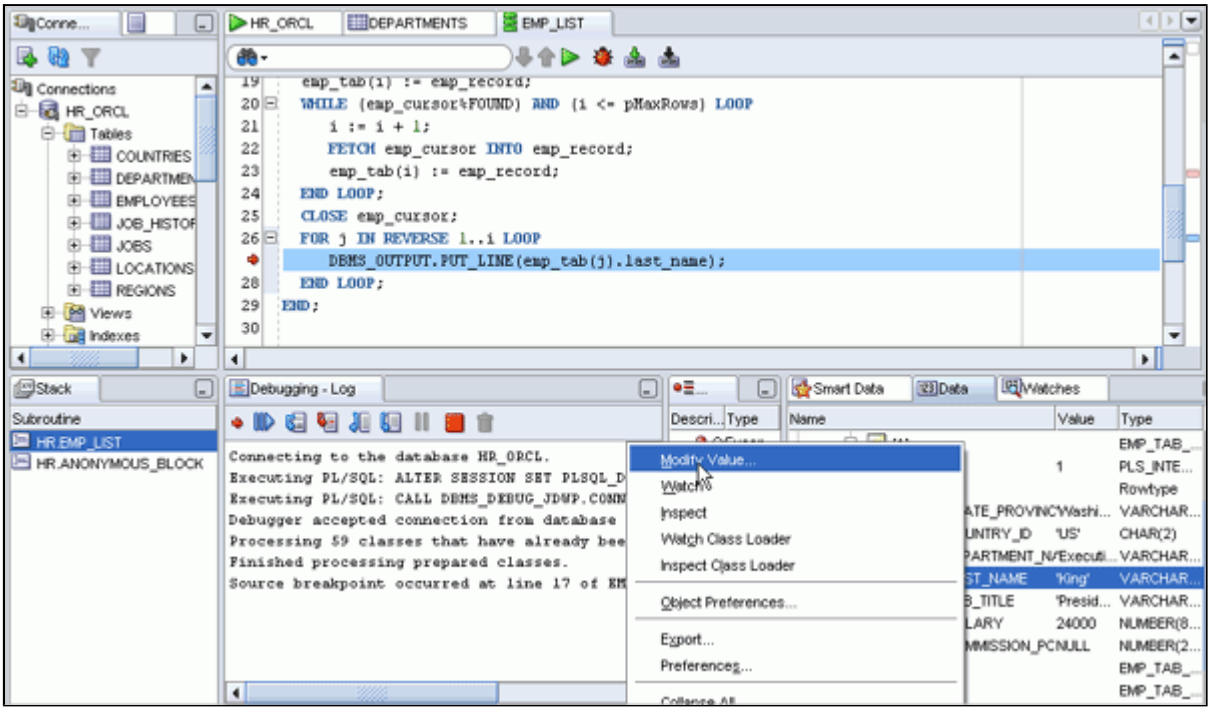
10. Right-click the line that reads `DBMS_OUTPUT.PUT_LINE(emp_tab(j).last_name);` and select **Run to Cursor.**



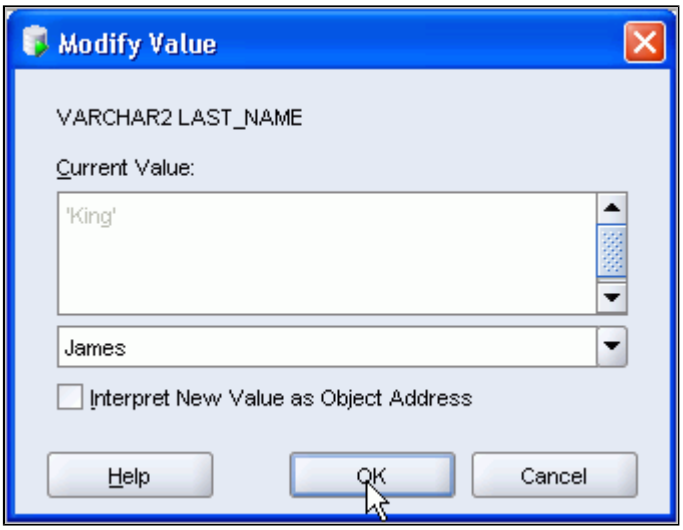
11. Expand **emp_tab > values > [1] > _value.** You see the values of the fields in a given record of the table. Select the **LAST_NAME** field.



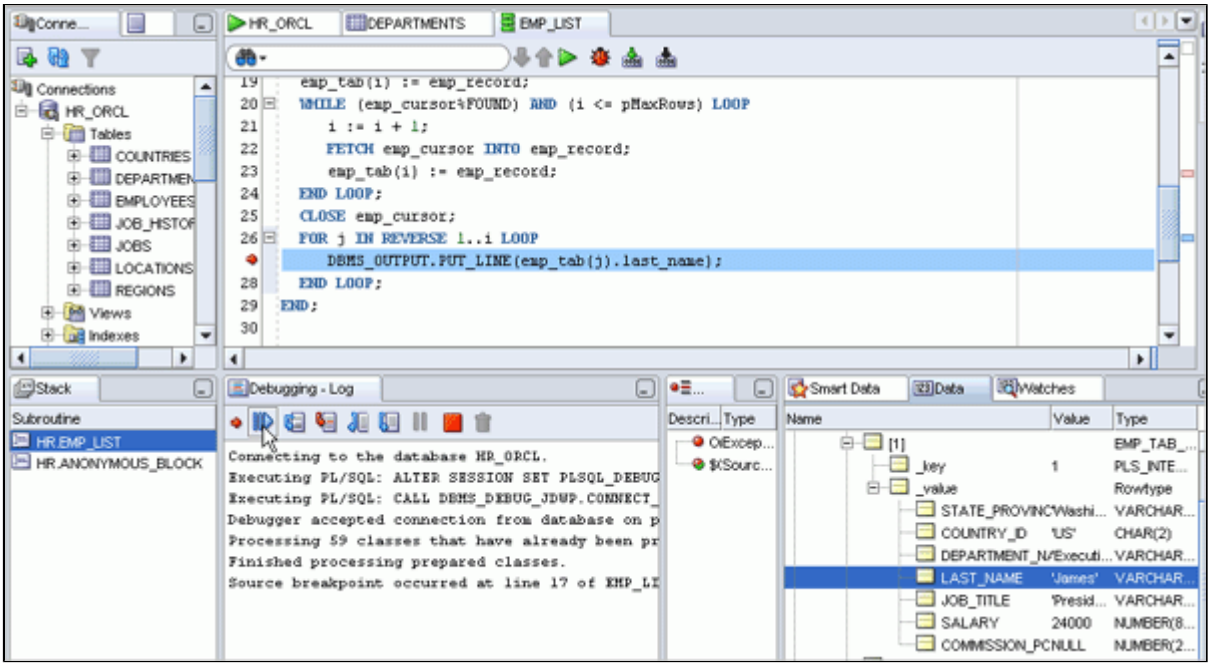
12. Right-click the **LAST_NAME** field and select **Modify Value**.



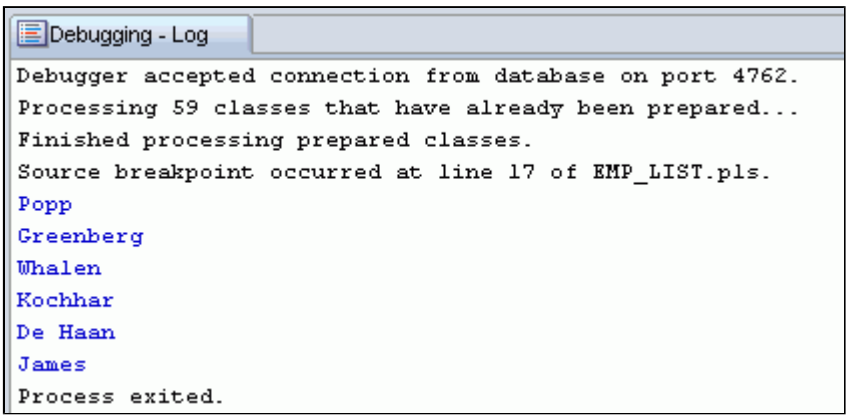
13. Change the name to something else and click **OK**.



14. Click the Resume icon  to allow the PL/SQL to run to completion.



15. Check to see that your modified value is displayed in the Log window.



Summary

In this lesson, you learned how to:

- Create a Database Connection
- Browse the Database
- Create and Compile a PL/SQL Procedure
- Run a PL/SQL Procedure
- Debug a PL/SQL Procedure

Back to Topic List

Place the cursor over this icon to hide all screenshots.

Creating XML Extensions for Oracle SQL Developer

Purpose

This tutorial shows you how to add XML extensions to Oracle SQL Developer.

Time to Complete

Approximately 30 minutes

Topics

This tutorial covers the following topics:

Viewing Screenshots

Place the cursor over this icon to load and view all the screenshots for this tutorial. (Caution: This action loads all screenshots simultaneously, so response time may be slow depending on your Internet connection.)

Note: Alternatively, you can place the cursor over an individual icon in the following steps to load and view only the screenshot associated with that step. You can hide an individual screenshot by clicking it.

Overview

Oracle SQL Developer is a free graphical tool that enhances productivity and simplifies database development tasks. Using Oracle SQL Developer, users can browse database objects, run SQL statements, edit and debug PL/SQL statements and run reports, whether provided or created.

Developed in Java, Oracle SQL Developer runs on Windows, Linux and the Mac OS X. This is a great advantage to the increasing numbers of developers using alternative platforms. Oracle SQL Developer is built on an extensible framework and, as such, is extensible itself. Users can create basic XML extensions or more involved Java extensions to add utilities or other functionality to the product.

Back to Topic List

Prerequisites

For this Hands On Session, the following has already been installed for you:

1. Oracle Database 10g.
   Note: To repeat this exercise later you can use any Oracle Database above 9.2.0.1

2. Oracle SQL Developer 1.2.1.
   Note: Oracle SQL Developer is available for download for FREE from **OTN.** To install Oracle SQL Developer, unzip it into any directory on your machine.

3. The files you use throughout the tutorial are located in the **\SQLDev_HOS\Extensions\files** directory. The files have also been included in the **extensions.zip** file.

4. The shipped **HR** schema.

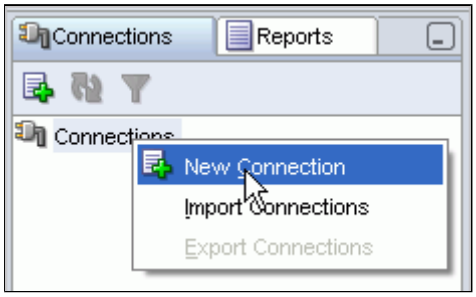
Creating a Database Connection

To create a database connection, perform the following steps:

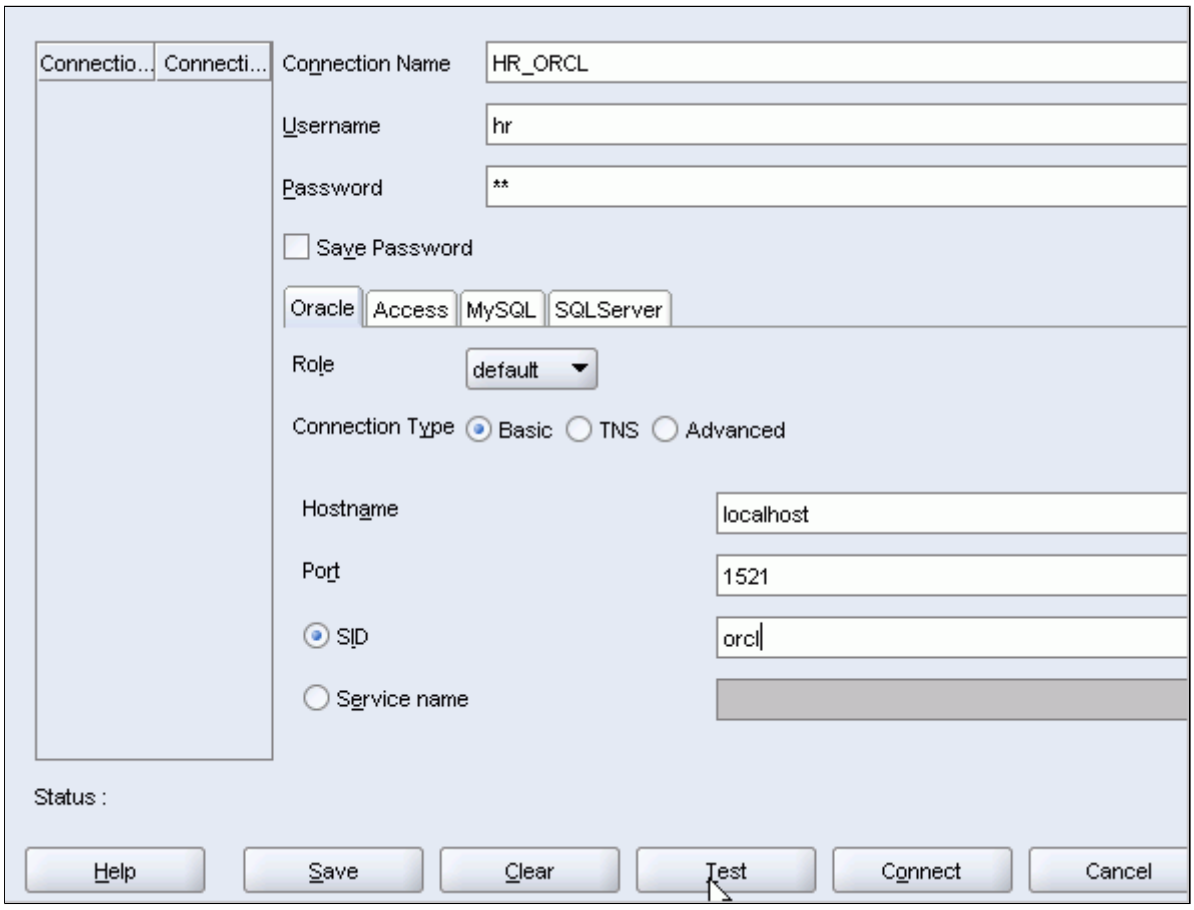
1. Open Windows Explorer and double-click **<your_path>\sqldeveloper\sqldeveloper.exe**.

   Note: If you receive a dialog window asking whether you want to migrate settings from a previous release, click **No**.

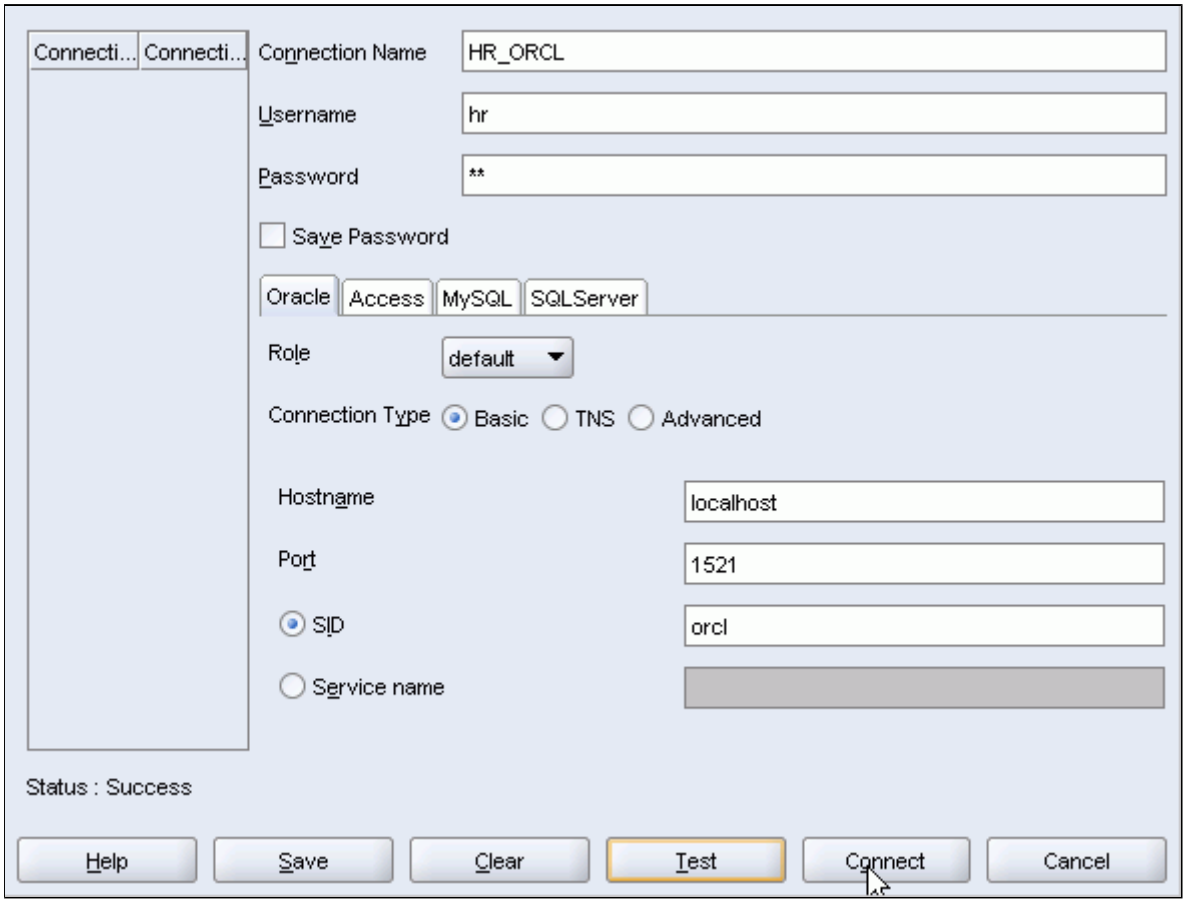
2. In the Connections tab, right-click **Connections** and select **New Connection**.



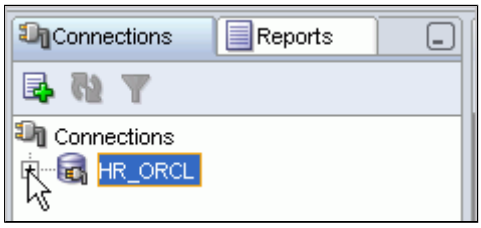
3. Enter **HR_ORCL** for the Connection Name (or any other name that identifies your connection), **HR** for the Username and Password, specify your **<hostname>** for the Hostname and enter **ORCL** for the SID. Click **Test**.



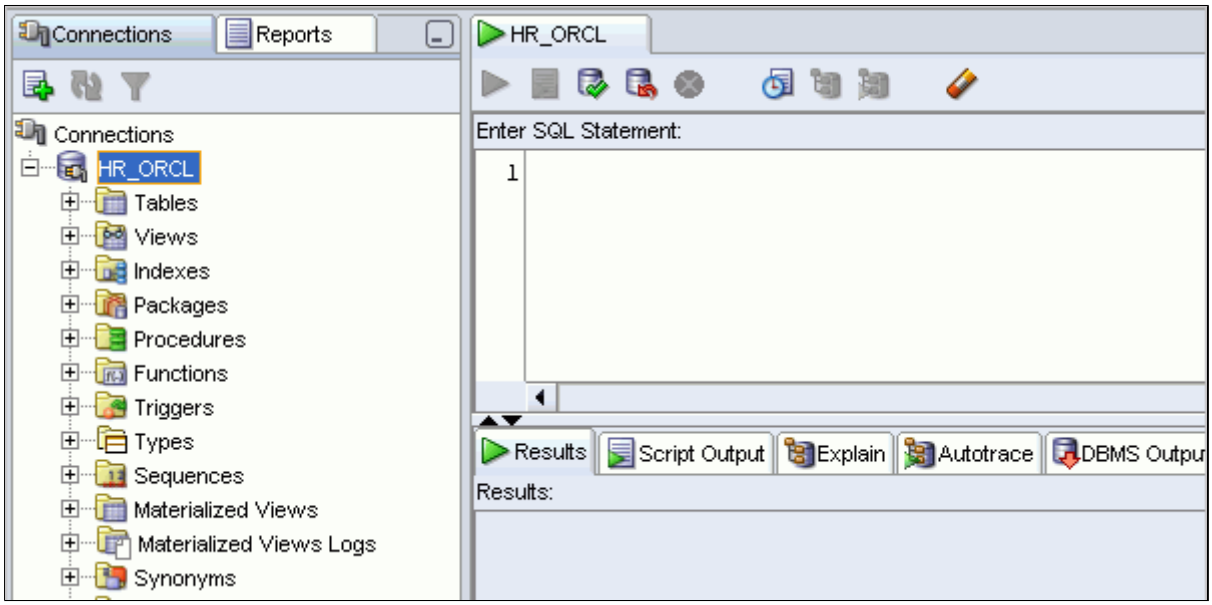
4. The status of the connection was tested successfully. The connection was not saved however. To save the connection, click **Connect**.



5. The connection was saved and you see the database in the list. Expand **HR_ORCL**.



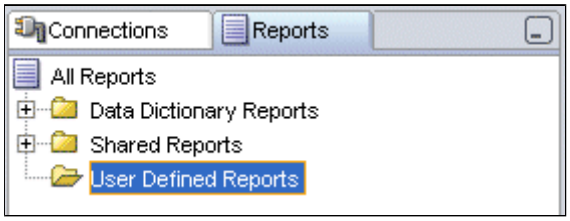
6. When a connection is opened, a SQL Worksheet is opened automatically. The SQL Worksheet allows you to execute SQL against the connection you just created.



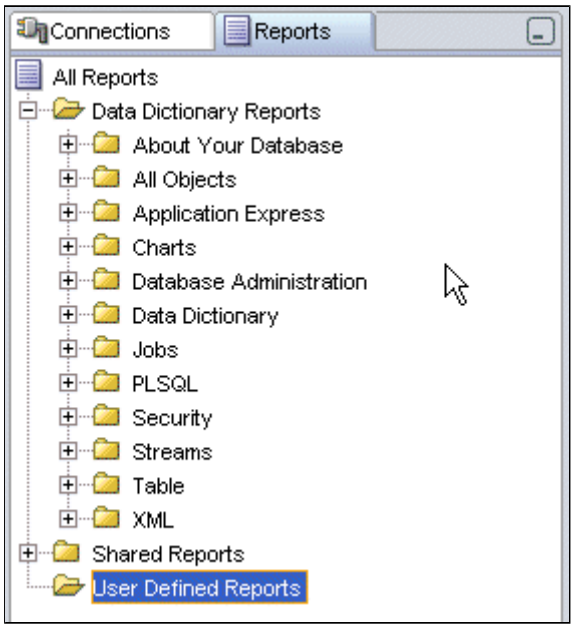
Sharing Reports

Users who want to share user defined reports can add these as an extension to Oracle SQL Developer. To do this perform the following steps:

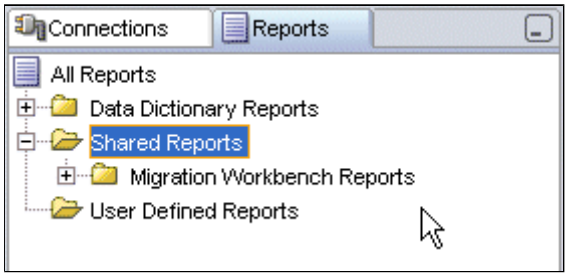
1. Select the Reports tab.



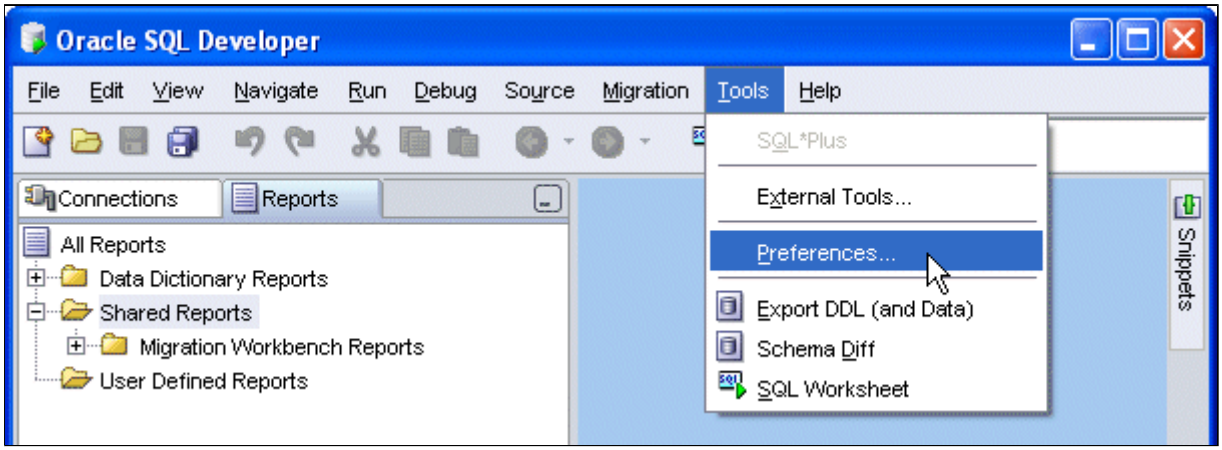
2. Expand each of the nodes. Note that there are a selection of shipped reports under the **Data Dictionary Reports** node.



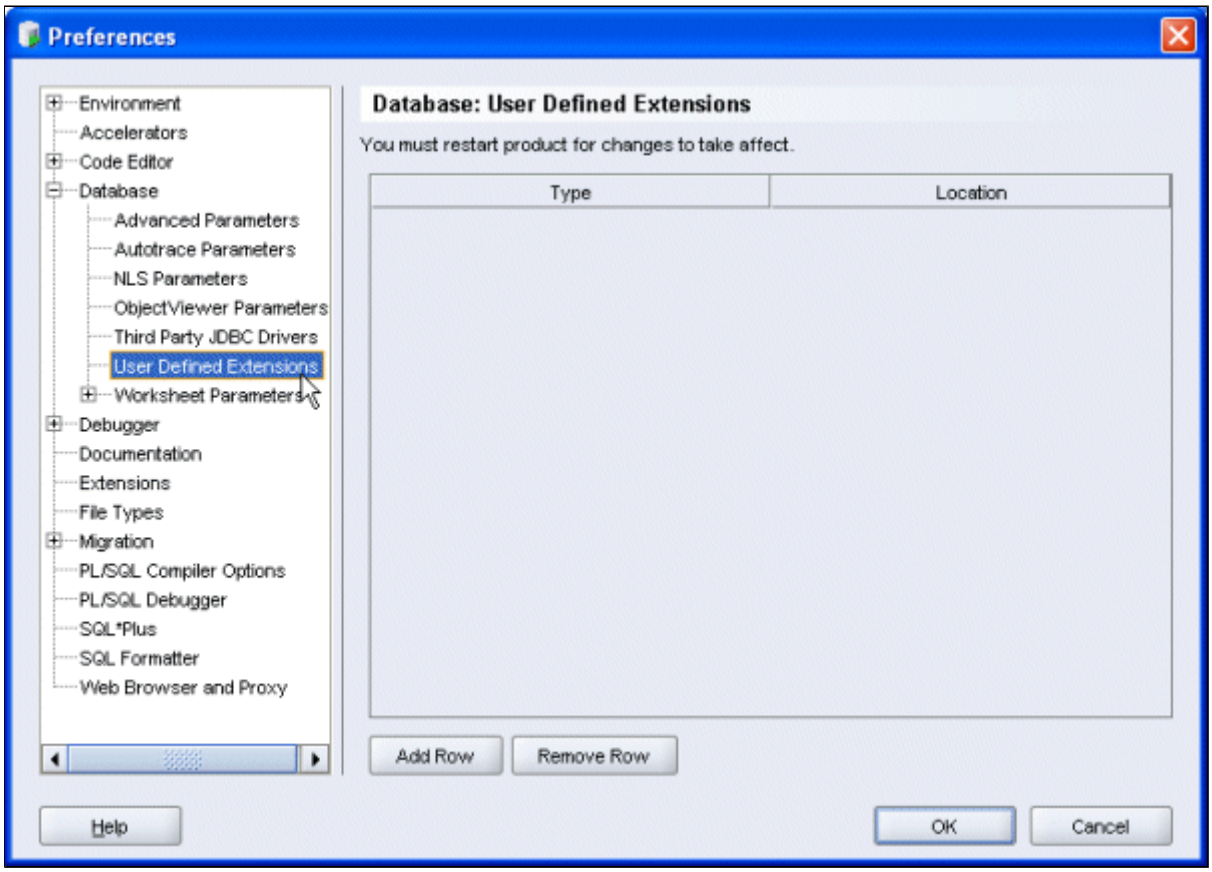
3. There are also shipped Migration Workbench Reports under the **Shared Reports** node. Any reports you create are added under the **User Defined Reports** node.



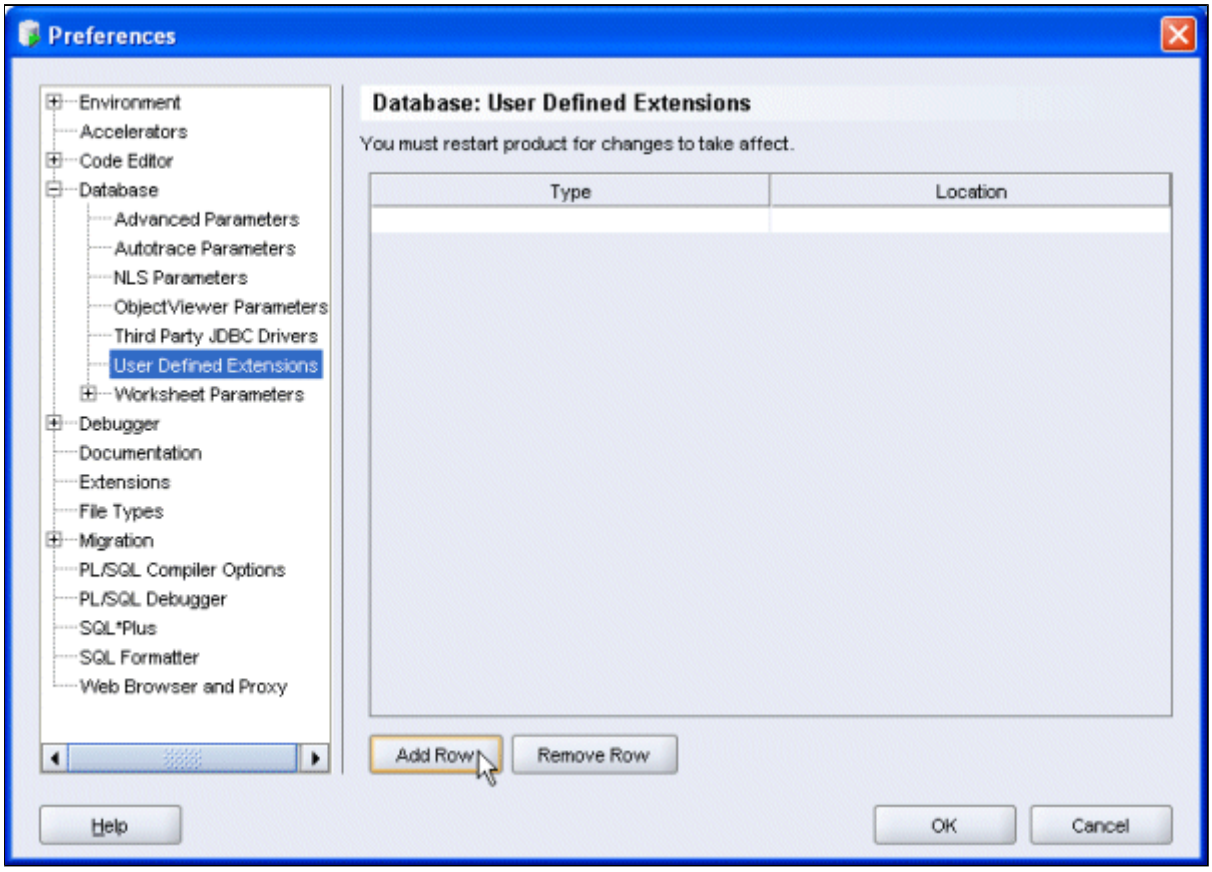
4. To add a shared reports extension, navigate to the **Tools-> Preferences** menu.



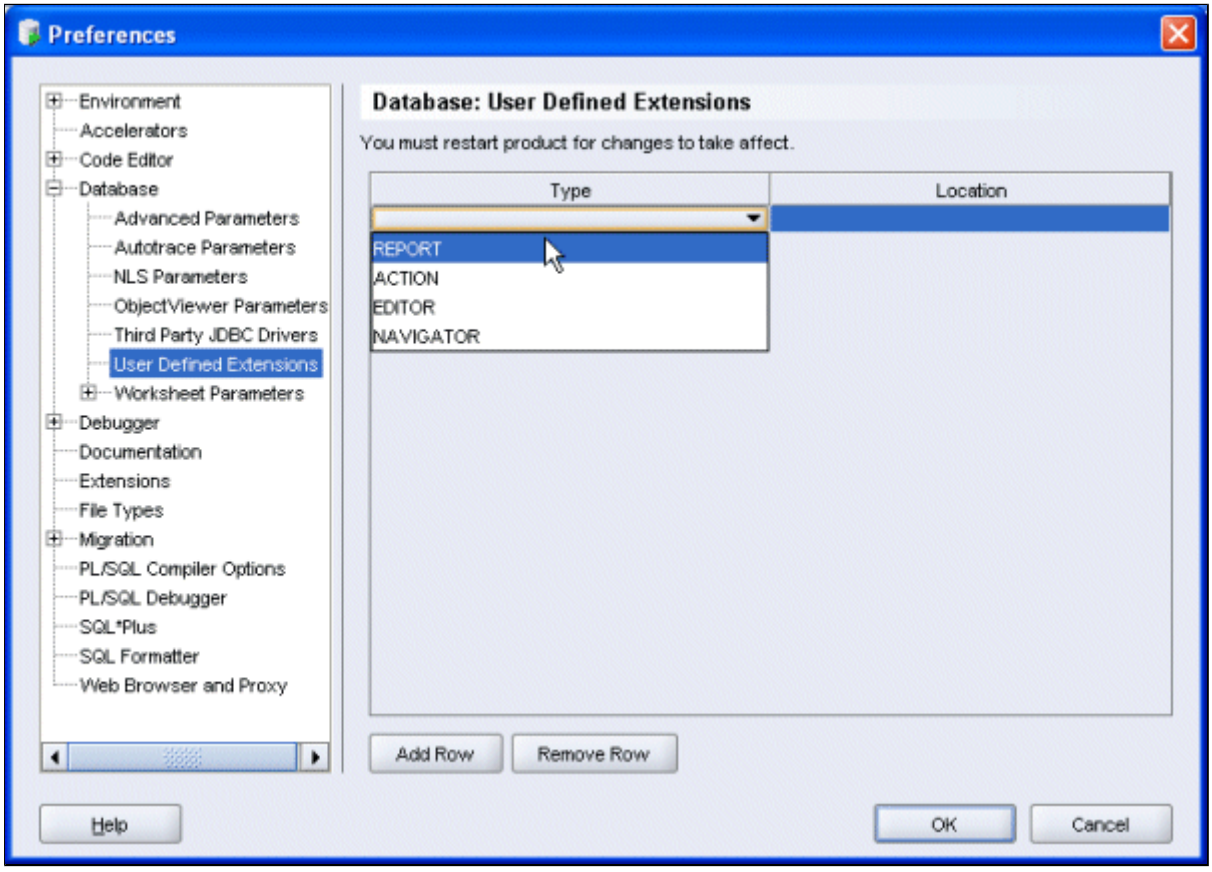
5. Expand the **Database** node and select **User Defined Extensions.**



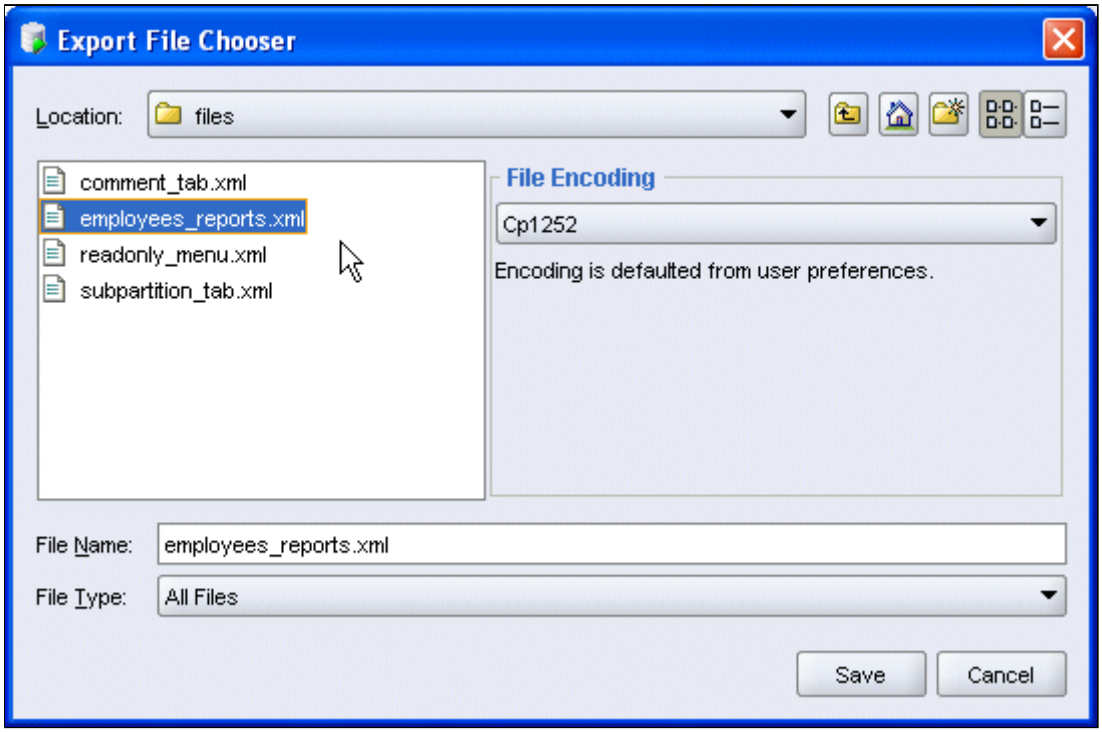
6. Click **Add Row**



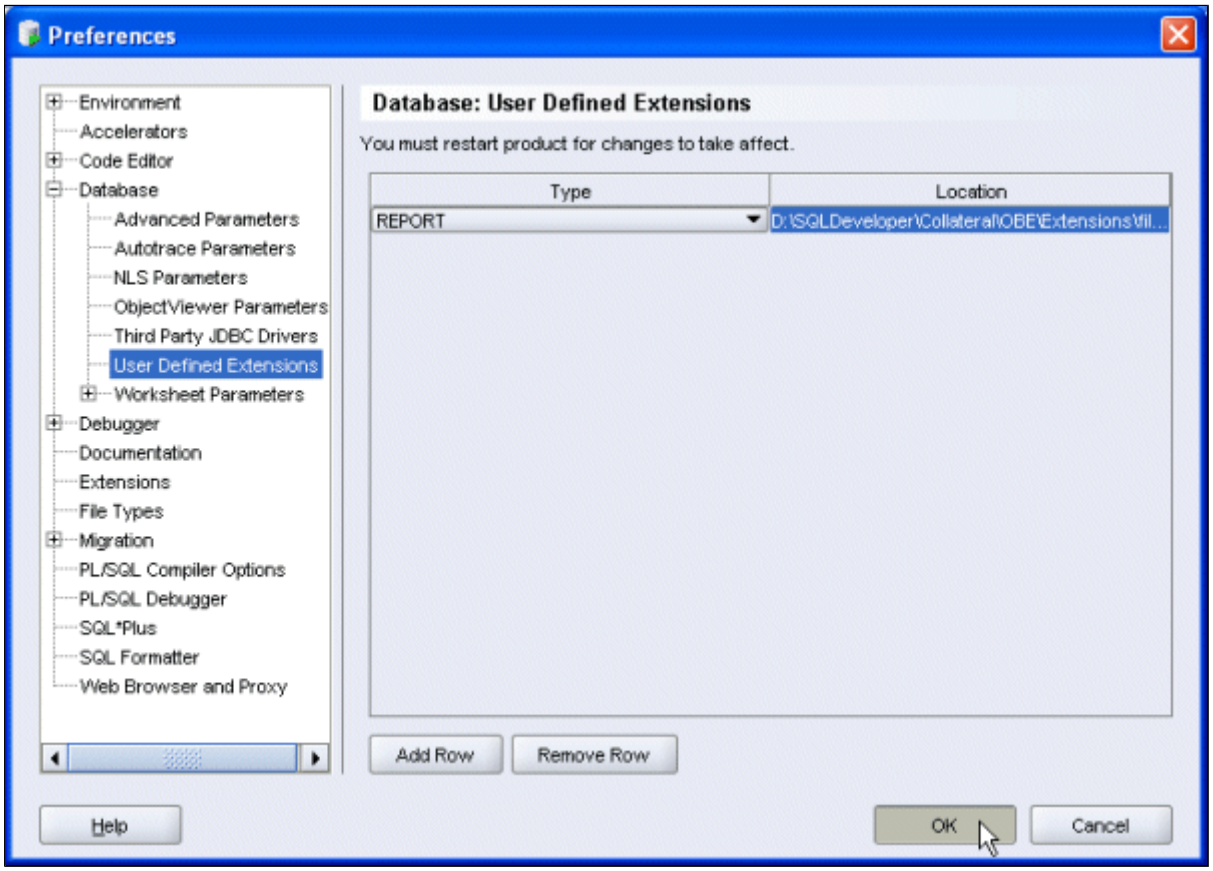
7. Click in the field under **Type** and select **Report.**



8. Click in the field under **Location** and **Browse** to the file location. The files are in the **\SQLDev_HOS\Extensions\files**. Select the **employees_reports.xml** file.

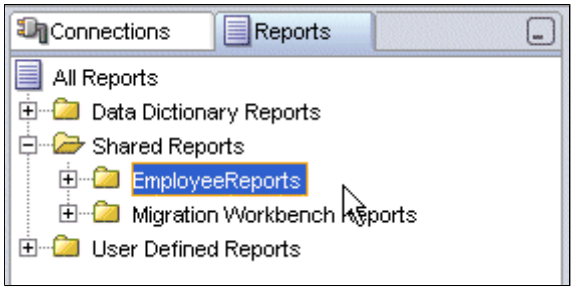


9. Ensure the location is selected, then click **OK**.

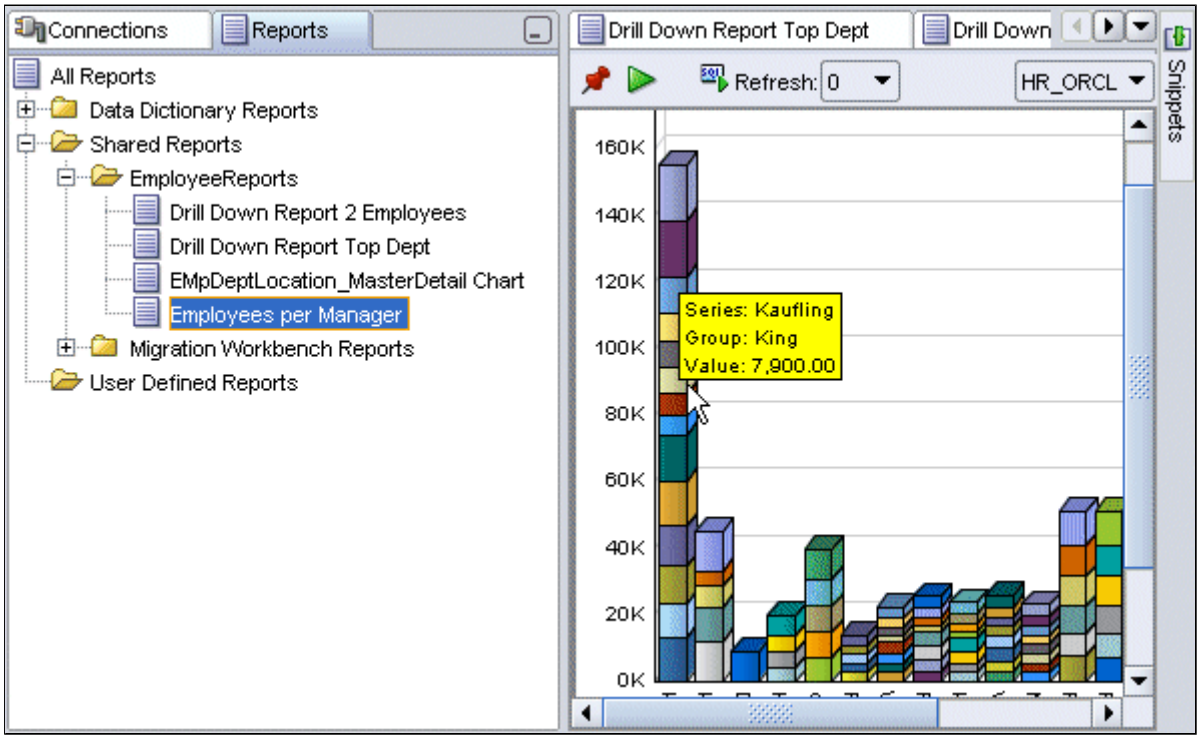


10. Shutdown and restart Oracle SQL Developer for the changes to take effect.

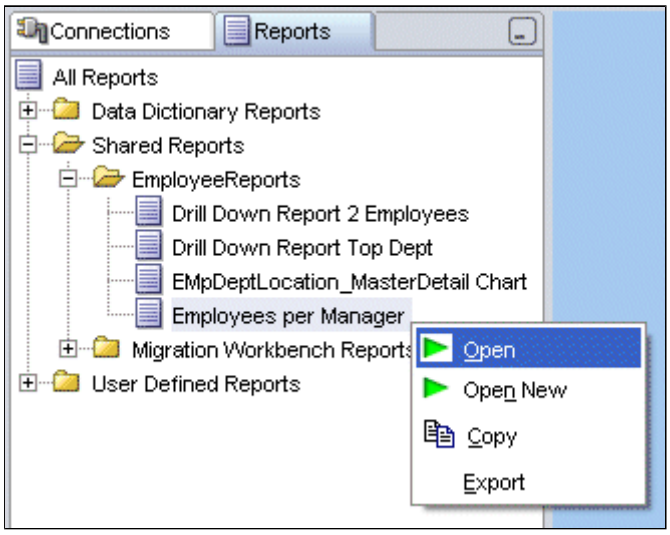
11. Navigate to the Reports tab and select **Shared Reports.**



12. Expand the **EmployeesReports** node and select **Employees per Manager**. Select HR_ORCL from the drop list when prompted and run the report.



13. Right-click on the **Employees per Manager** report. Note that the context menu does not permit you to edit this report. You can copy and recreate it as a User Defined report, but shared reports are not editable.

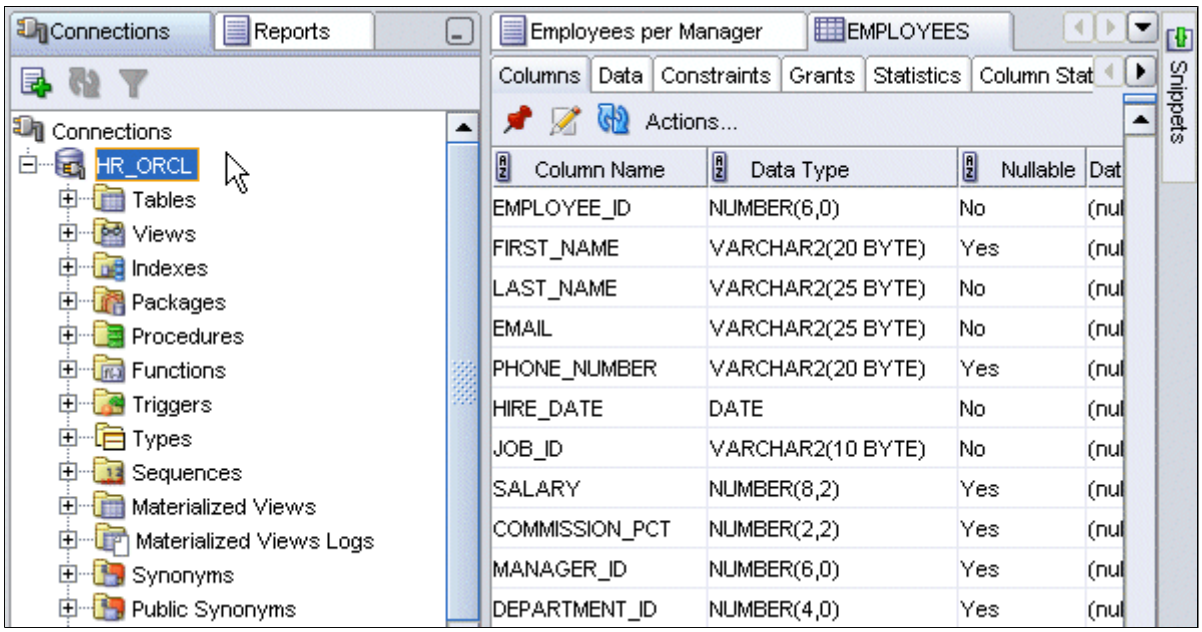


Back to Topic List

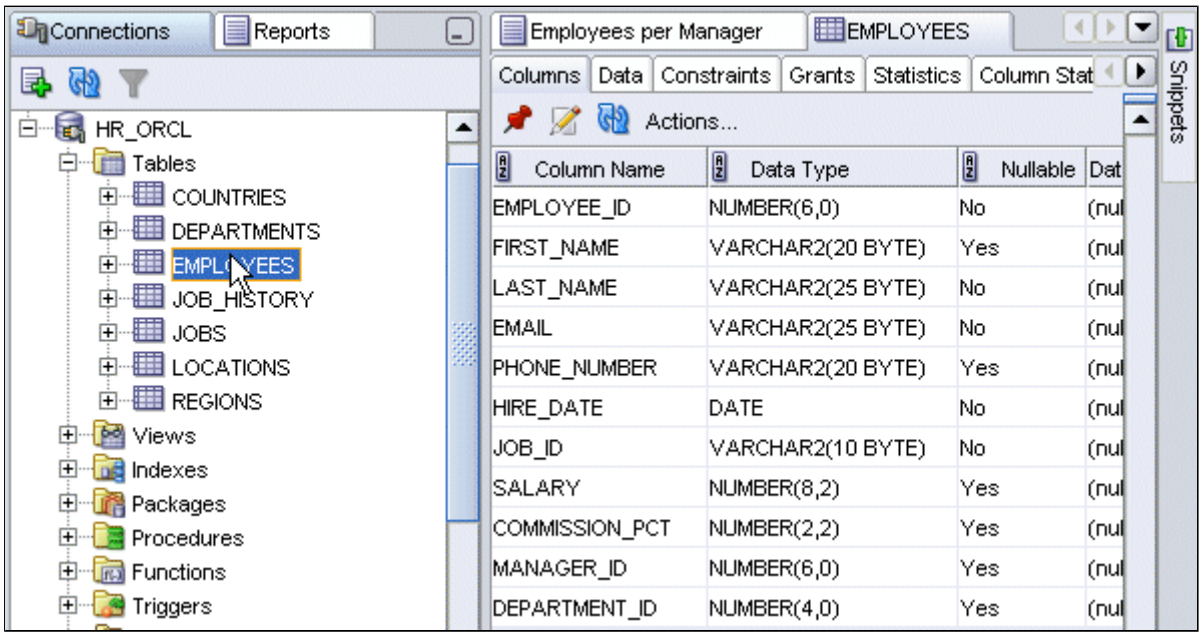
Adding an Extra Tab to Display Column Comments

Each object in the Navigator has a matching set of definition tabs. These are based on queries against the Data Dictionary. Not all features describing an object are displayed in these tabs. To create a Column Comments tab for a table, perform the following steps:

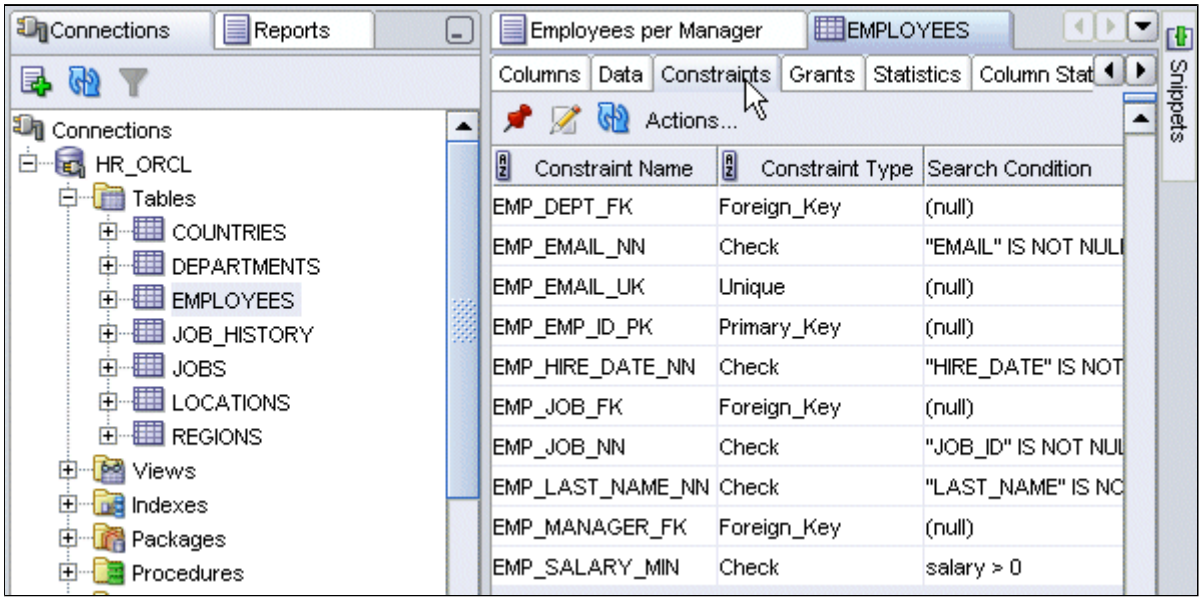
1. Select the **Connections** tab and expand the **HR_ORCL** connection.



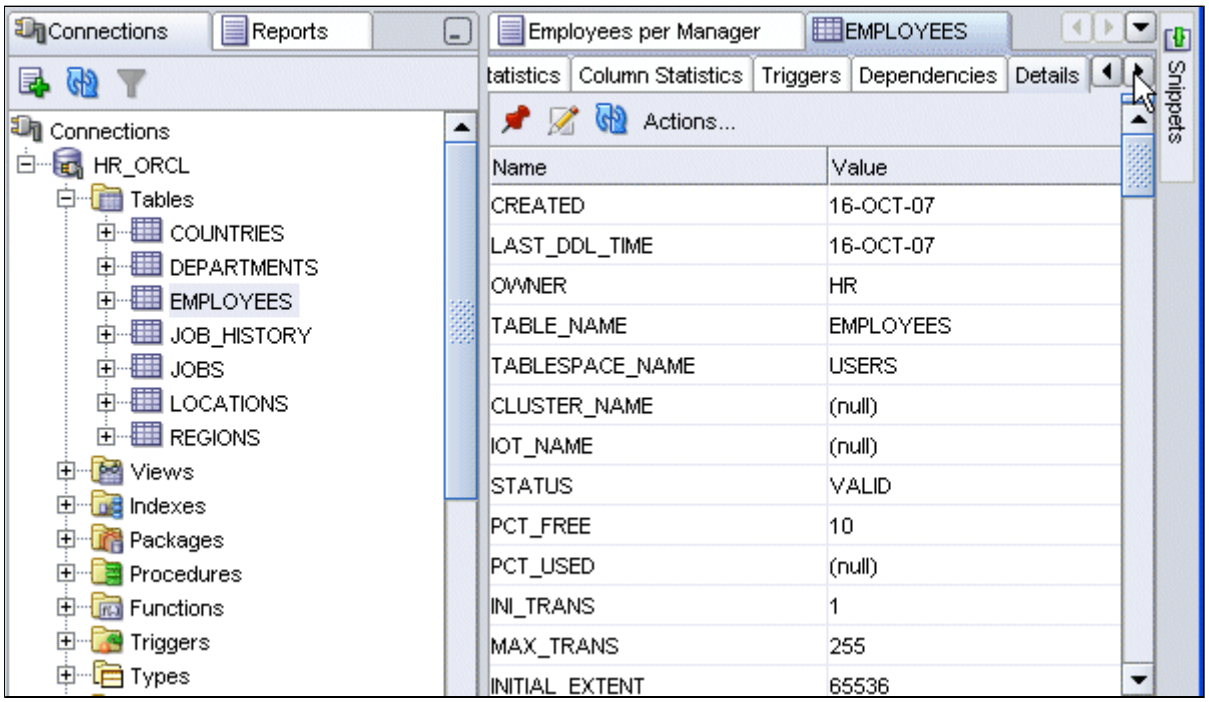
2. Expand the **Tables** node and select **EMPLOYEES**.



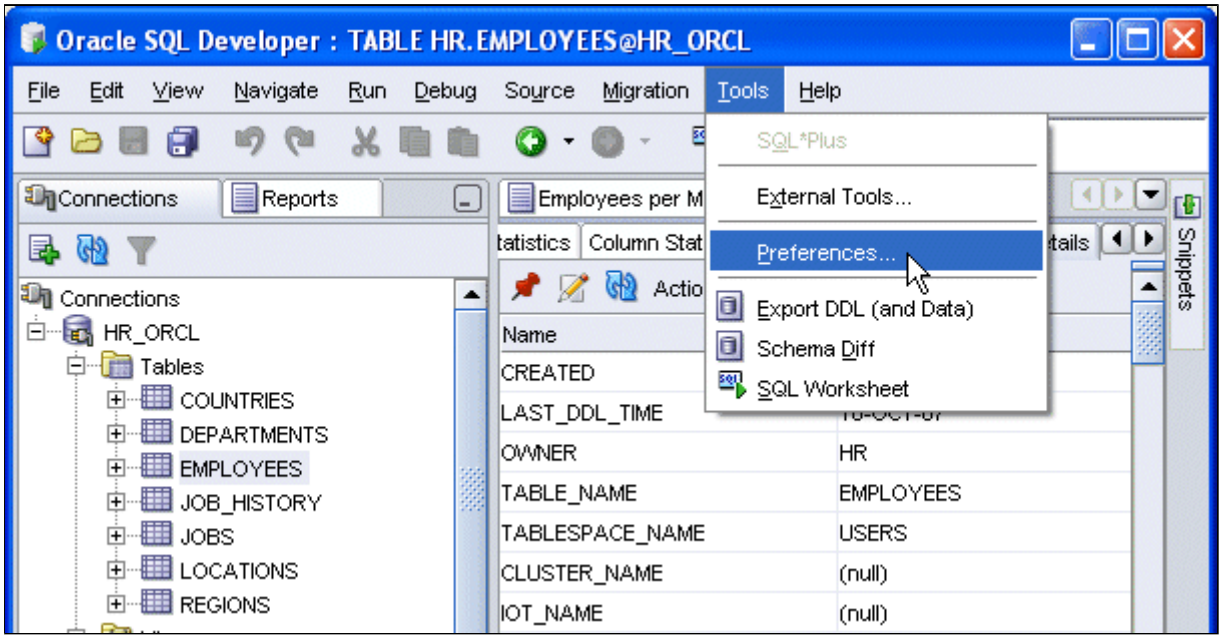
3. The initial tab displayed is the **Columns** tab. Select the **Constraints** tab and view the details displayed.



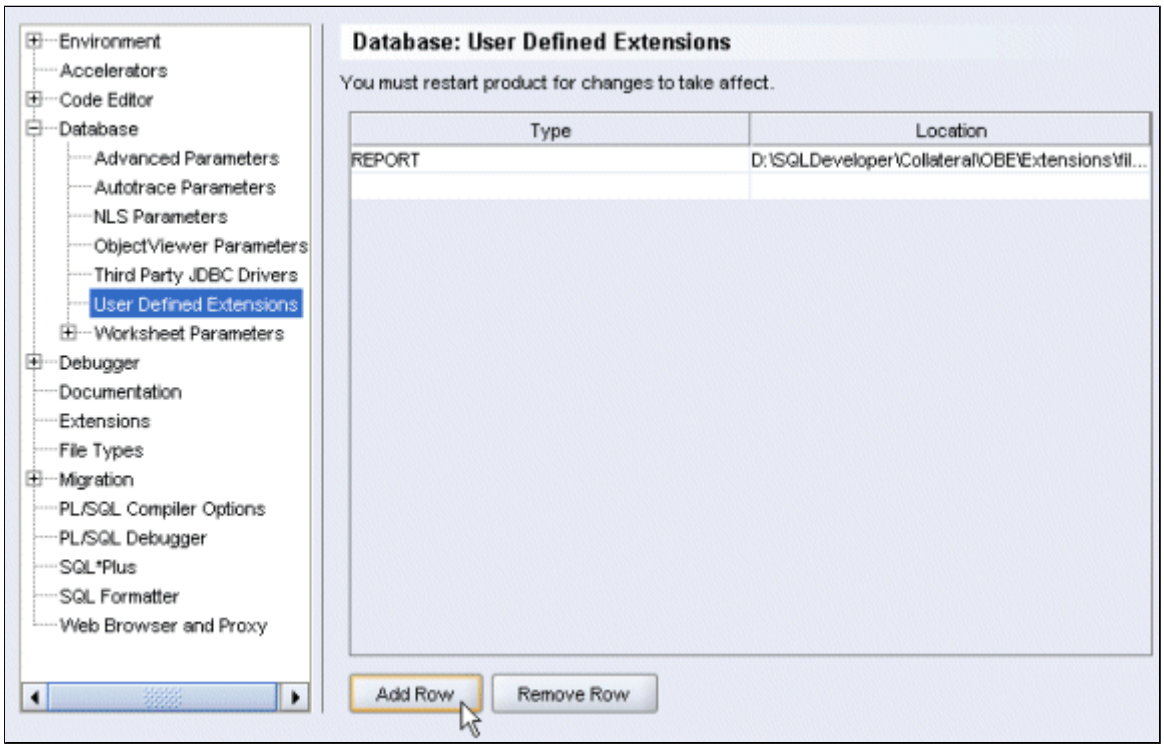
4. Click through the other tabs to see the full details available for a table.



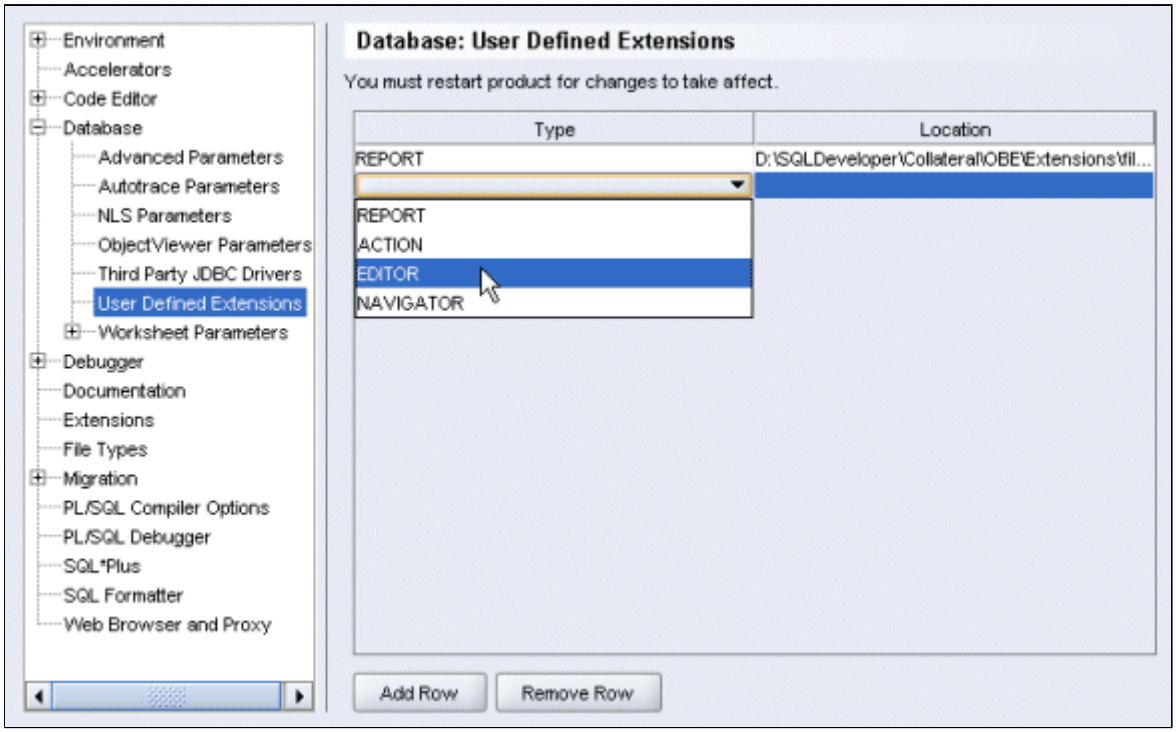
5. Some users would like to display the column comments for a table in a tab too. To do this you need to add in an extension. Select **Tools->Preferences**.



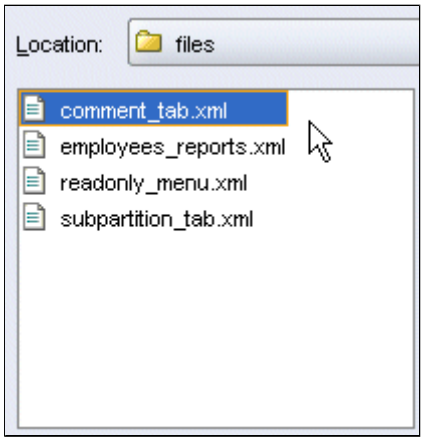
6. As before, select **Database** and **User Defined Extensions.** Click **Add Row.**



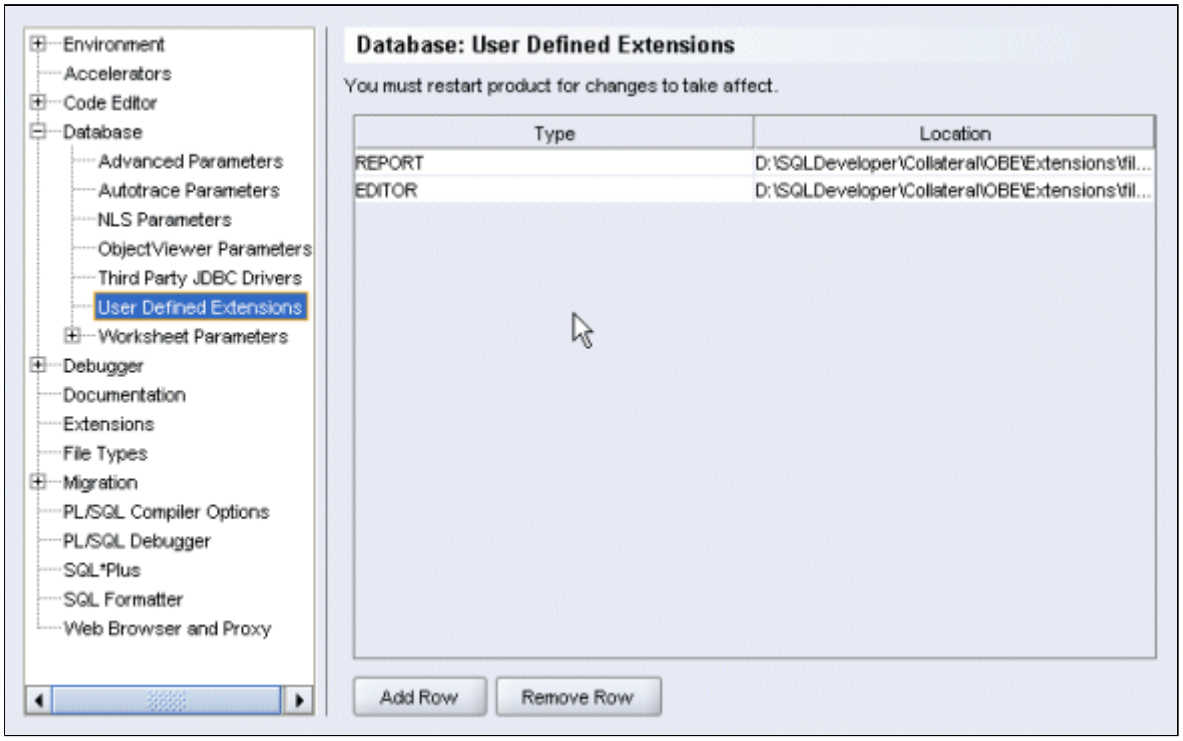
7. Select **Editor** from the drop list for **Type.**



8. Set the **Location** by browsing to the **\SQLDev_HOS\Extensions\files** directory and select the **comment_tab.xml** file.

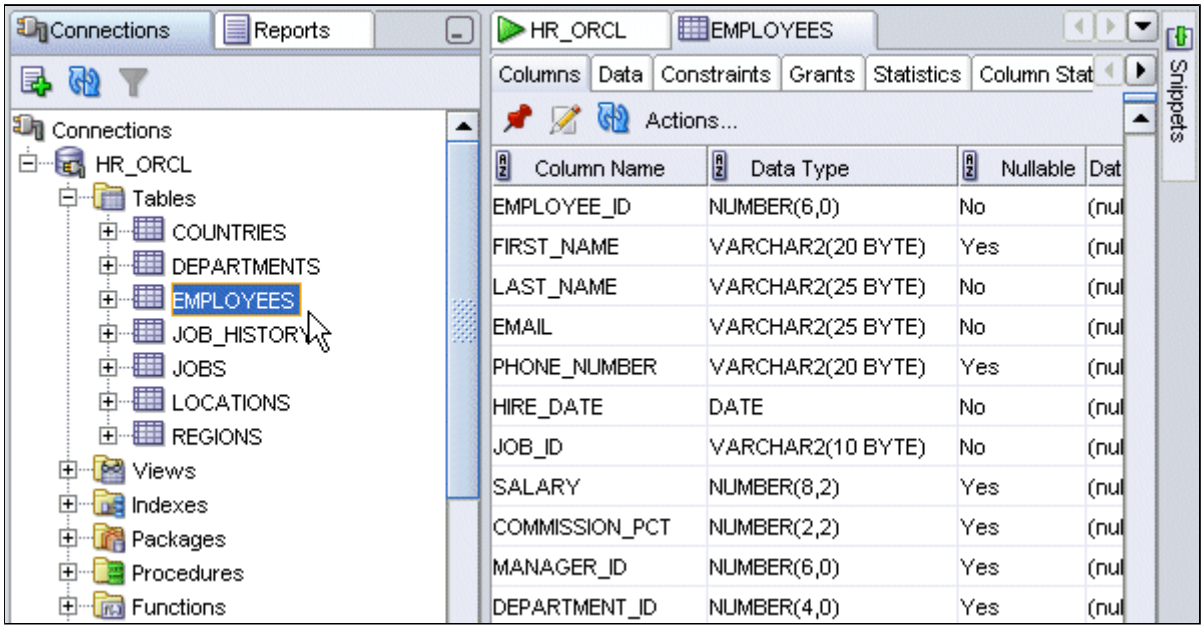


9. Ensure the Location is stored and click **OK.**

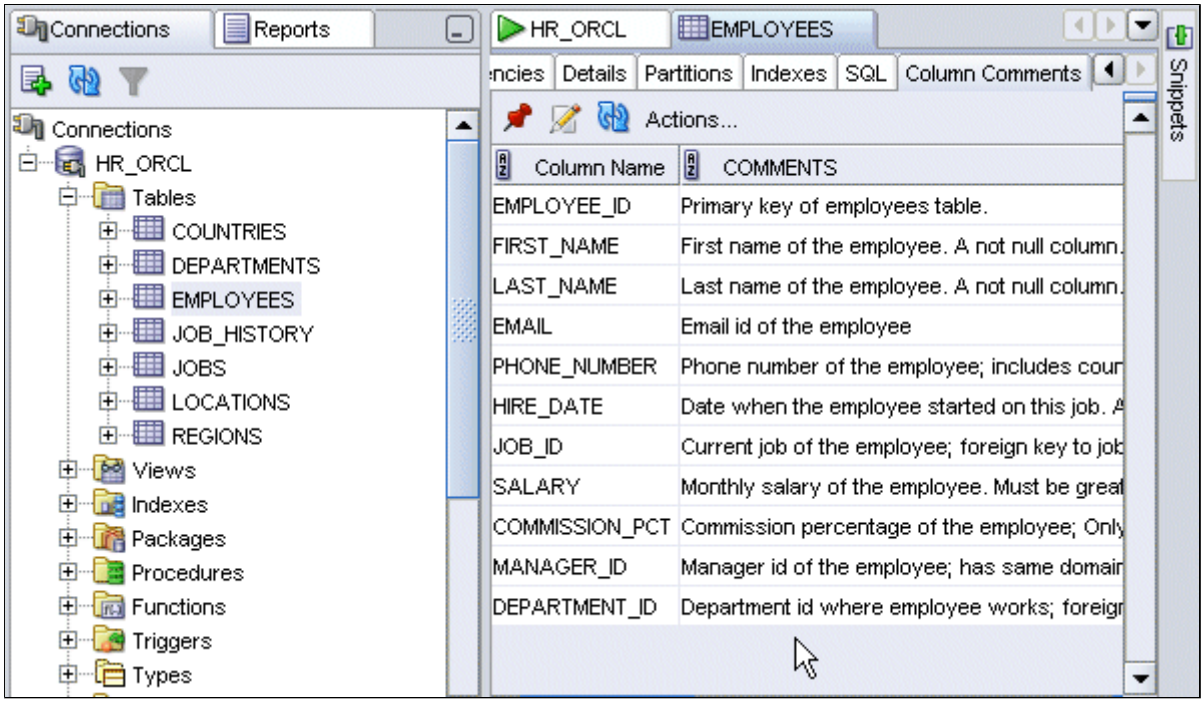


10. Shut down and restart Oracle SQL Developer for your changes to take effect.

11. Select the **HR_ORCL** connection. Expand the tables node and select **EMPLOYEES.**



12. Tab through to the last tab, **Column Comments.** The new tab is added after the other tabs.

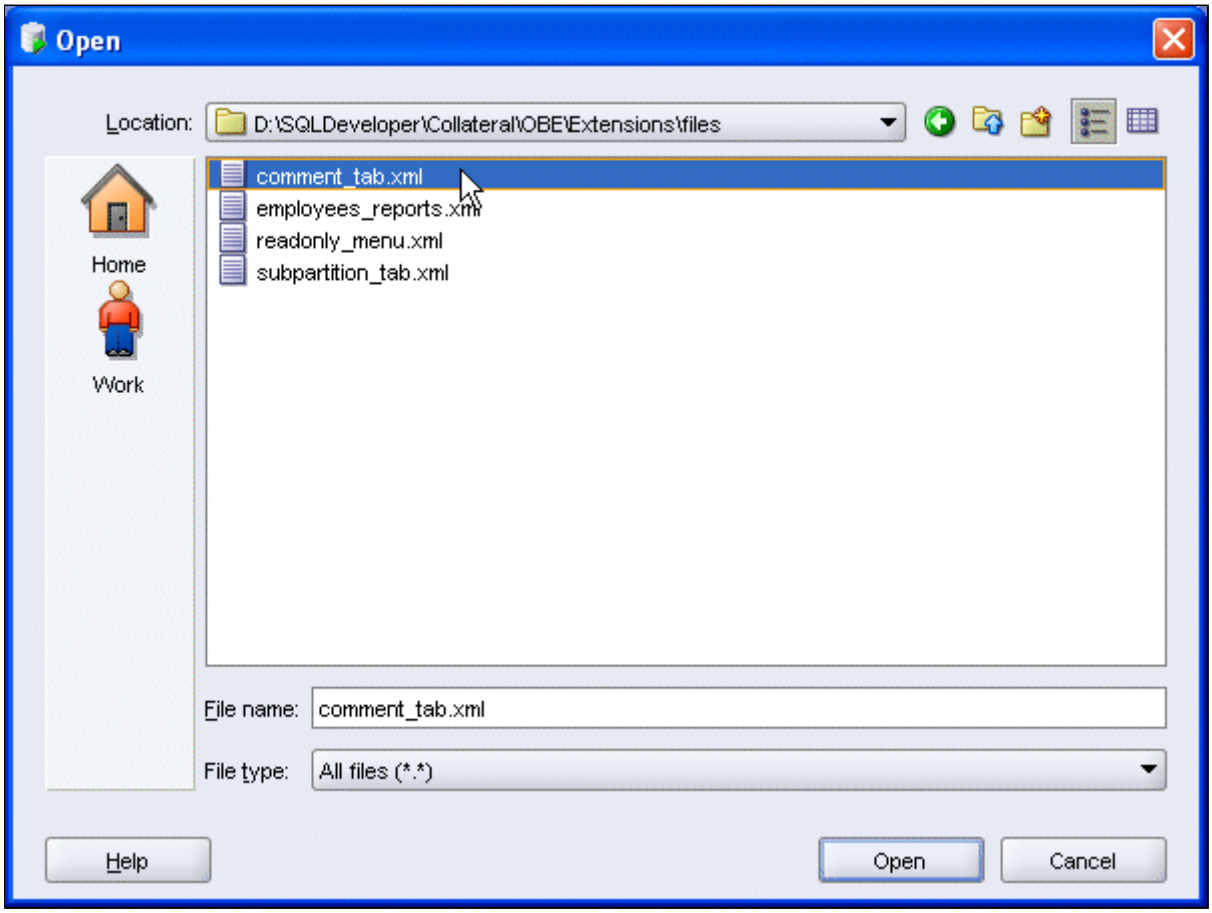


Back to Topic List

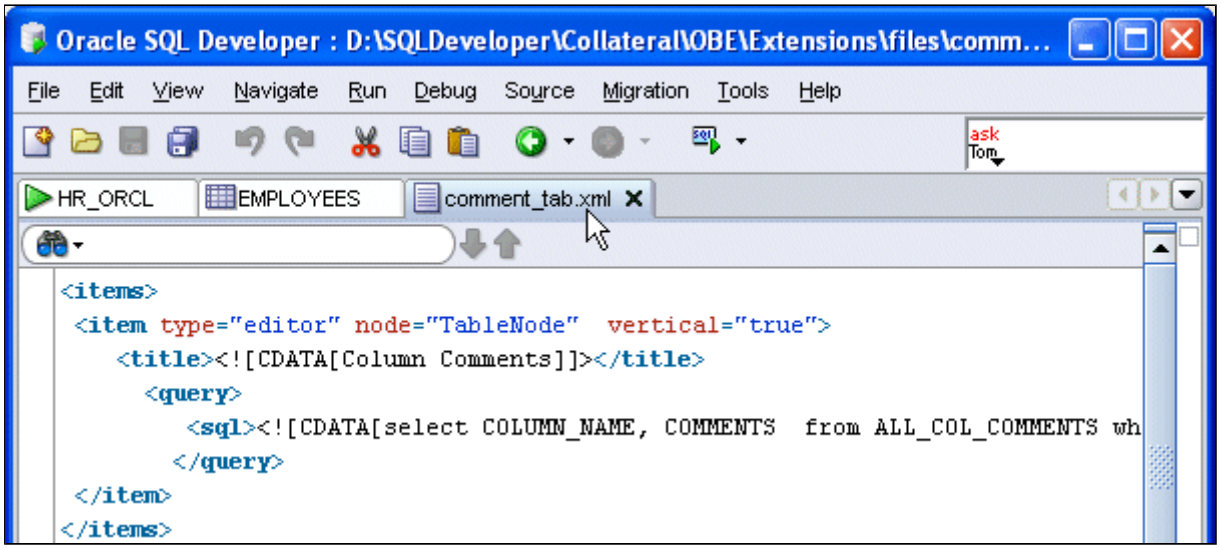
Reviewing the XML File structure

In order to add an XML extension to Oracle SQL Developer, you need to create the XML file. In this section you review the structure of the XML File.

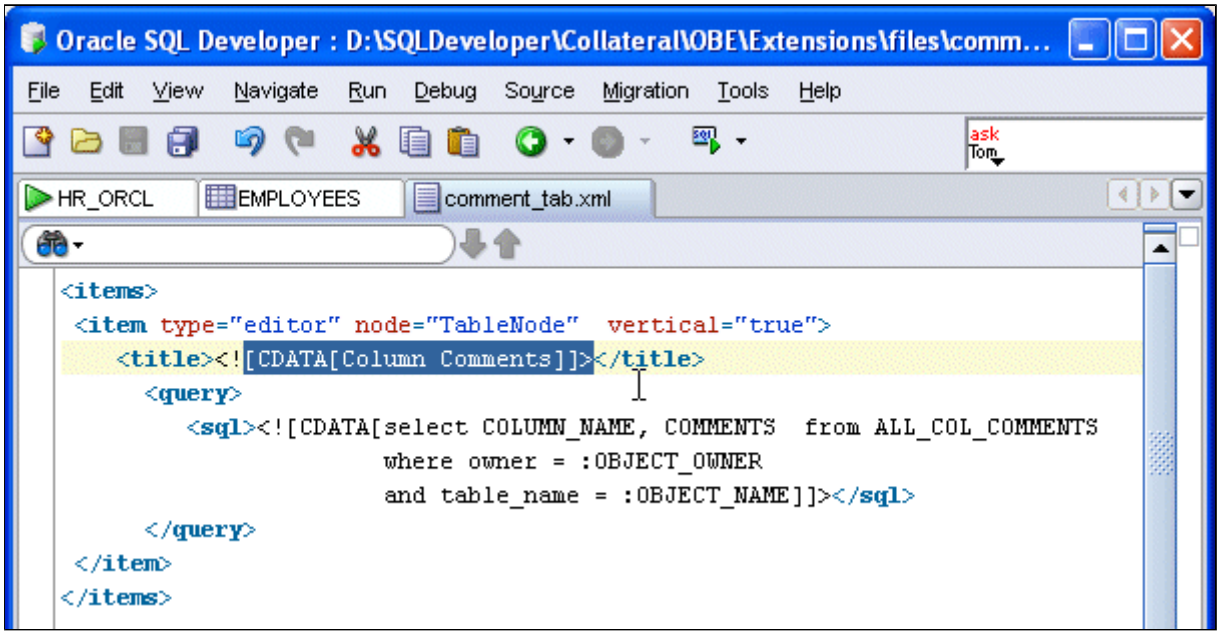
1. In Oracle SQL Developer, select **File ->Open** and browse to the **\SQLDev_HOS\Extensions\files** directory and select the **comment_tab.xml** file.



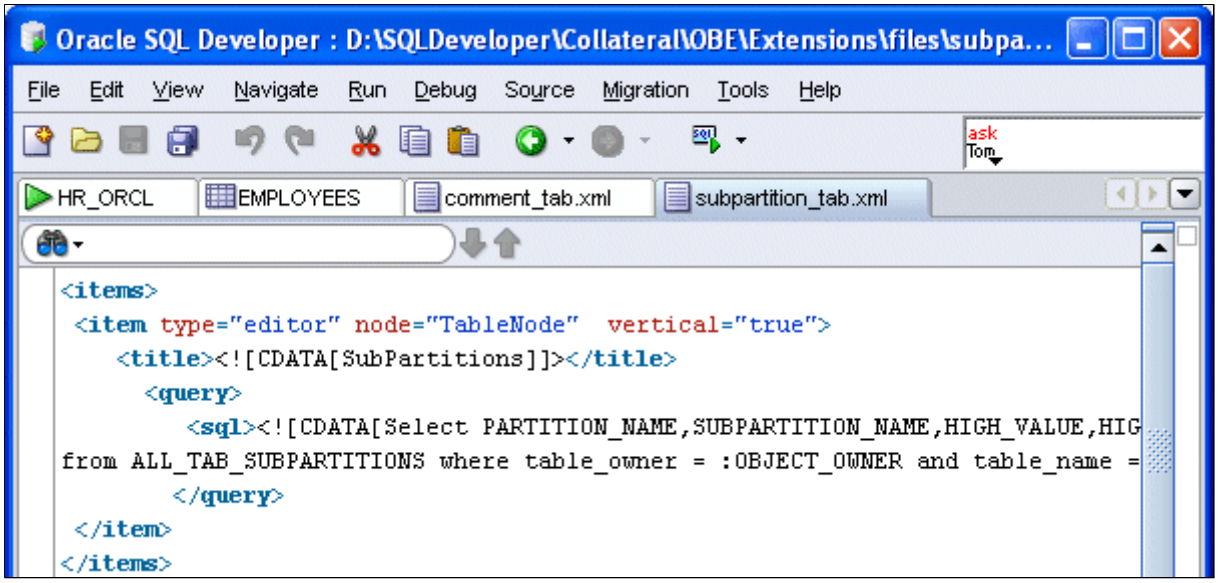
2. Double-click on the tab to maximize the editor.



3. The Oracle SQL Developer framework recognizes the XML tags. The pieces of code you are interested in are Title and the SQL Query. The Title here is **Column Comments** and is used to name to tab.



4. The SQL Query is querying the Data Dictionary for the column comments for the **OWNER** and the **OBJECT_NAME**, in this example the EMPLOYEES Table for schema HR.

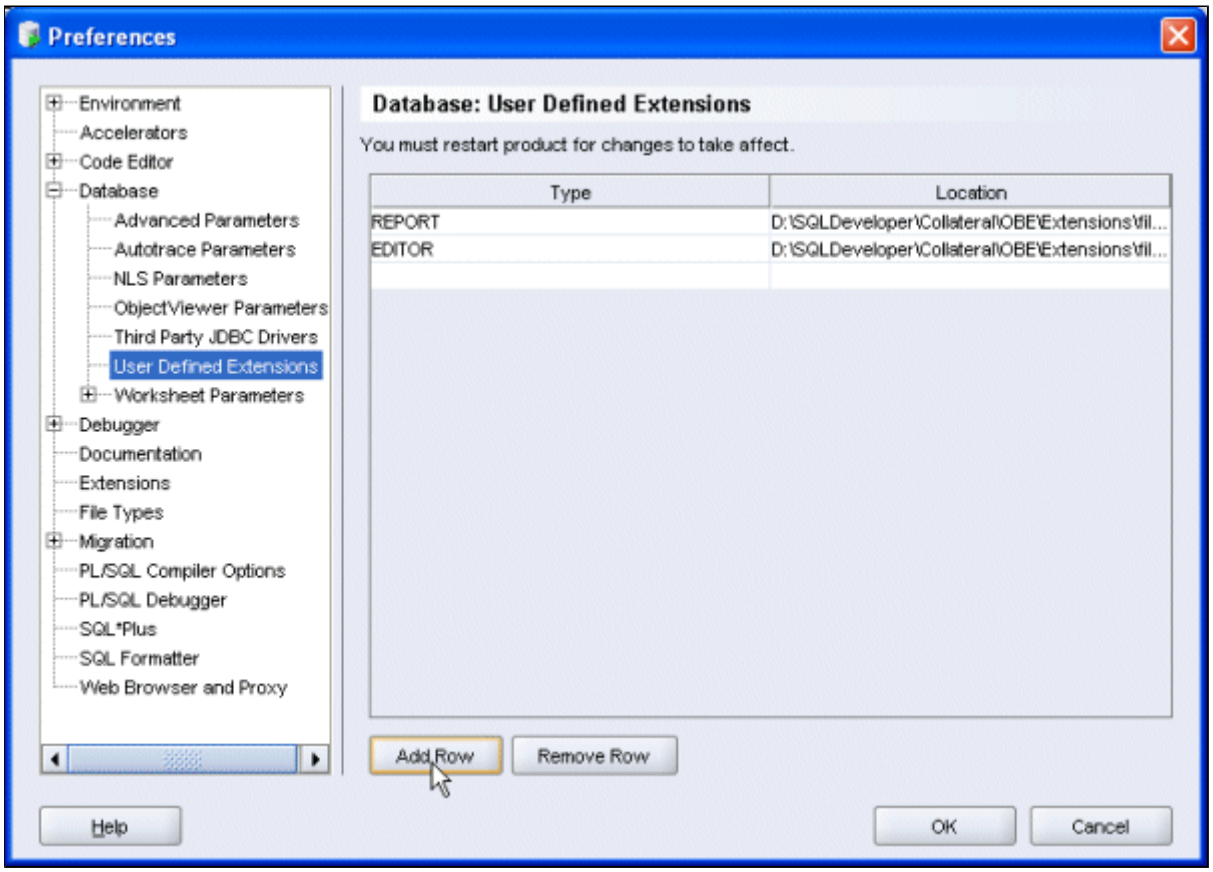


Back to Topic List

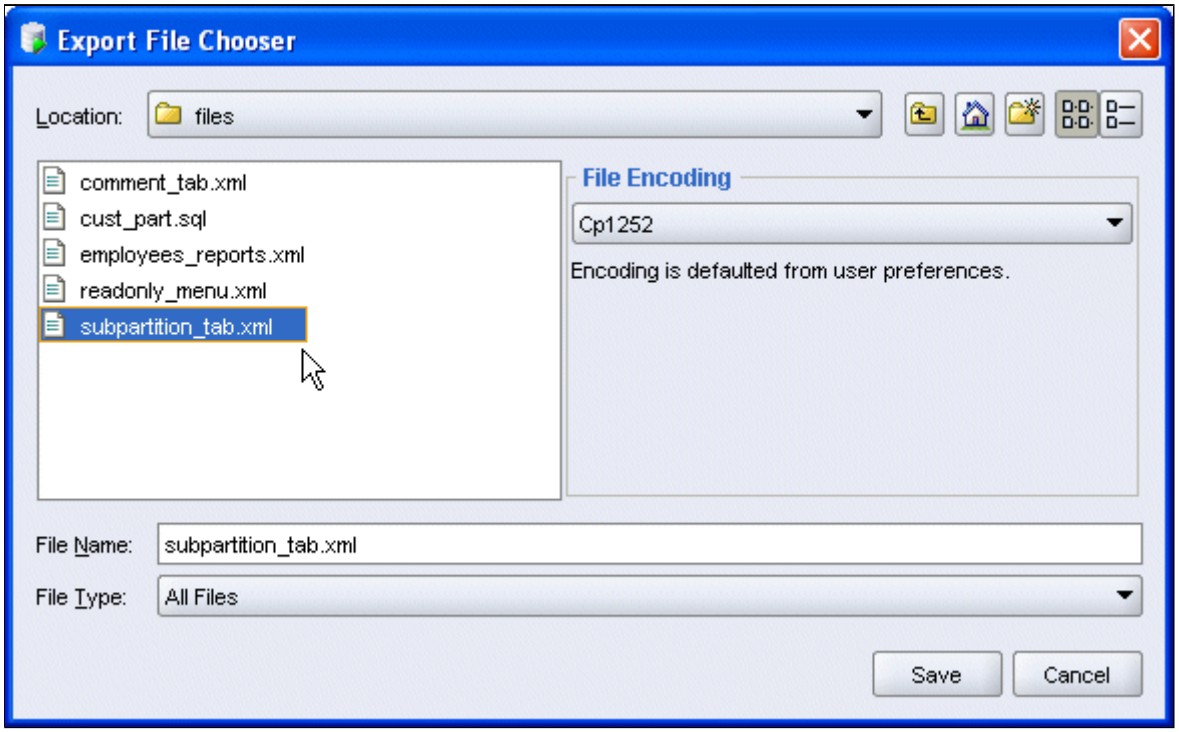
Adding a Tab to Describe Sub-Partitions

This is example is the same as the one just completed above. In this instance you are going to add a tab to describe sub partitions. You create a table with partitions and sub partitions to review the details. To add additional tabs to the Table definitions, perform the following steps:

1. Select **Tools -> Preferences** to invoke the dialog. Expand the **Database** node and select **User Defined Extensions.** Click **Add Row.**

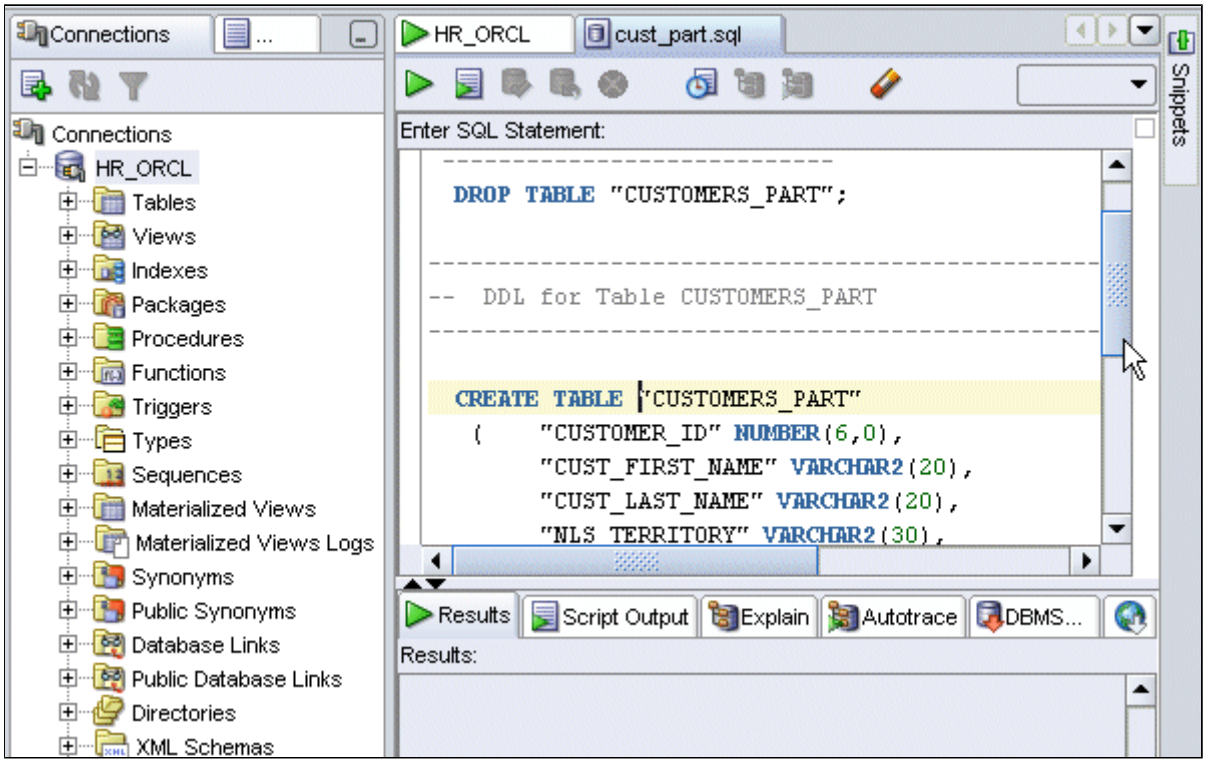


2. The Extension Type is Editor. Browse to the **\SQLDev_HOS\Extensions\files** directory and select the **subpartition_tab.xml** file.

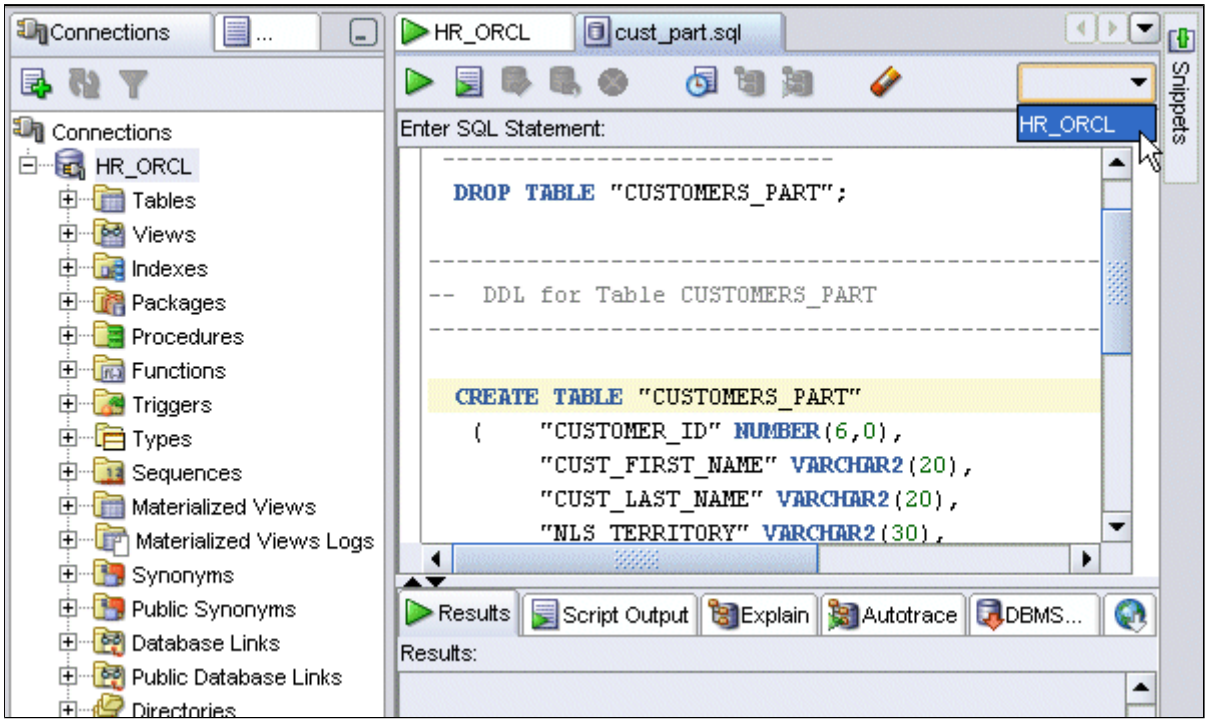


3. Shutdown and restart Oracle SQL Developer for your changes to take effect.

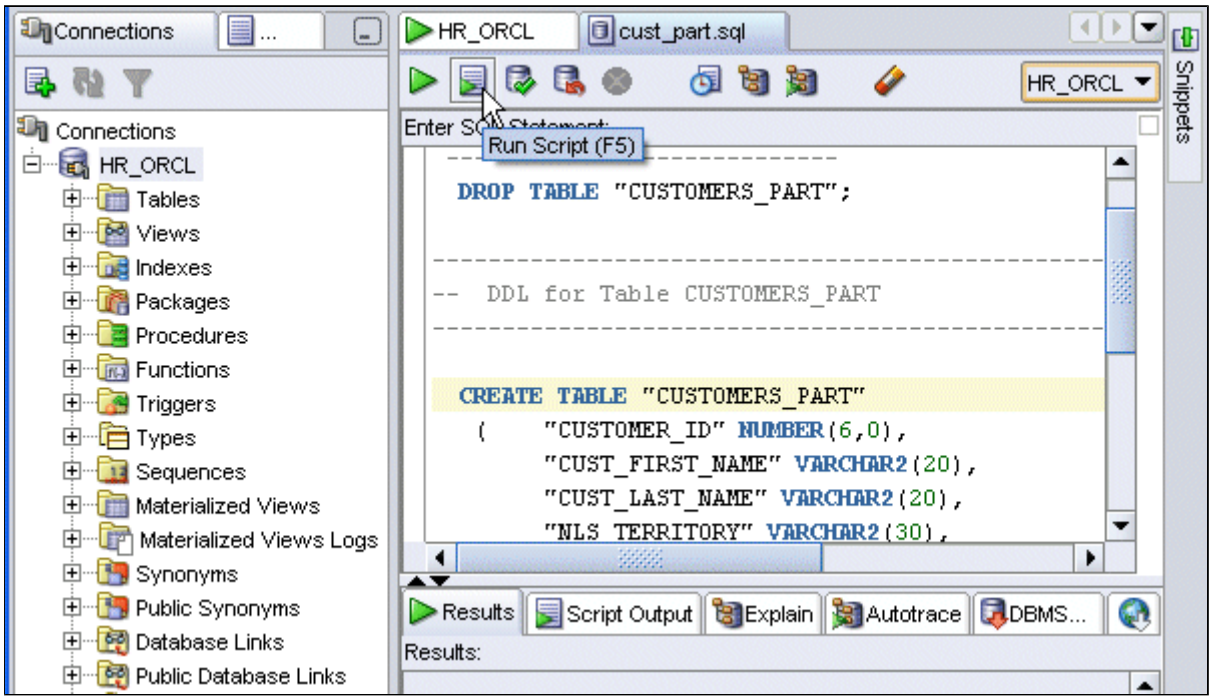
4. Expand the HR_ORCL connection. Select File ->Open and browse to the the **\SQLDev_HOS\Extensions\files** directory. Select the **cust_part.sql** file.



5. Review the file. This script creates a partitioned table, **CUSTOMERS_PART**. Select the **HR_ORCL** connection from the drop list on the right of the SQL Worksheet.

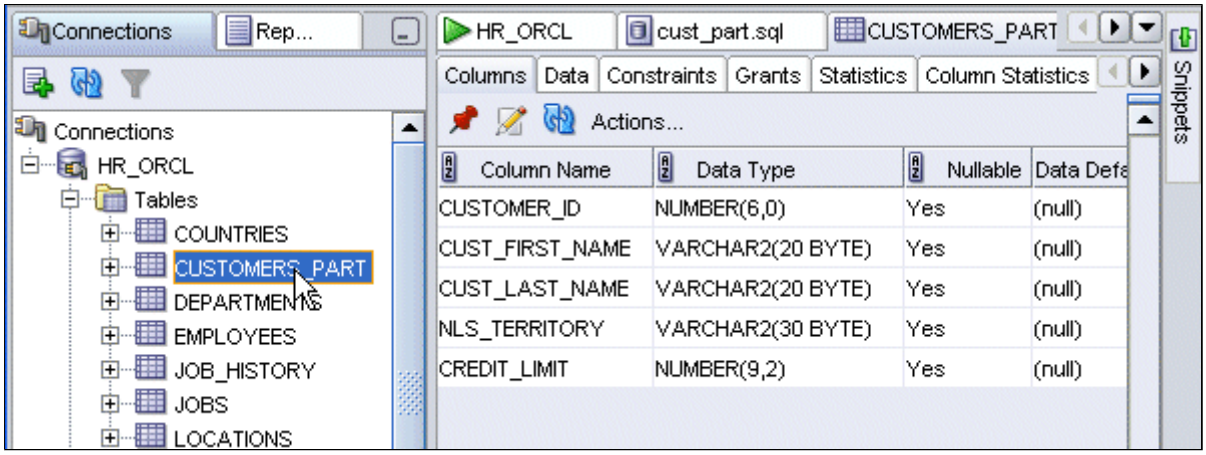


6. Click the **Run Script (F5)** icon.

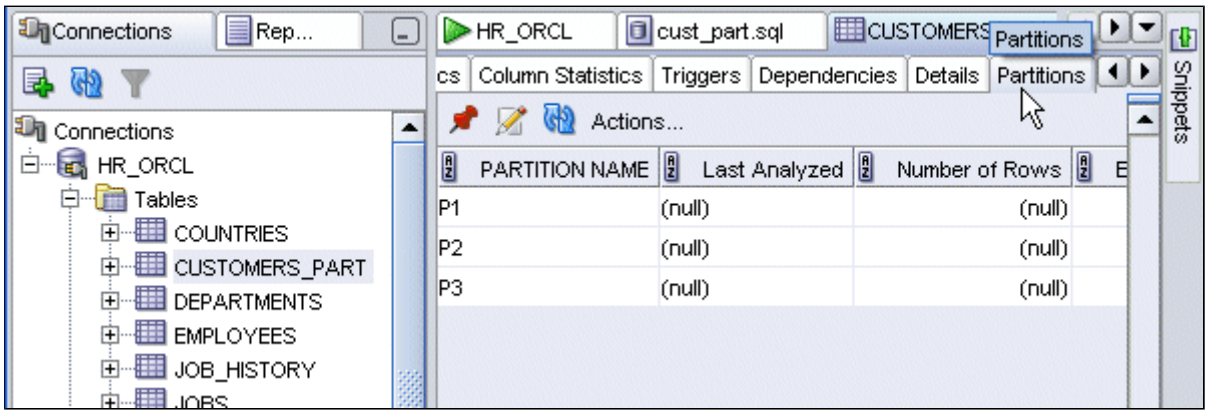


7. Expand the Tables node in the Connections Navigator and select the **CUSTOMERS_PART** table.

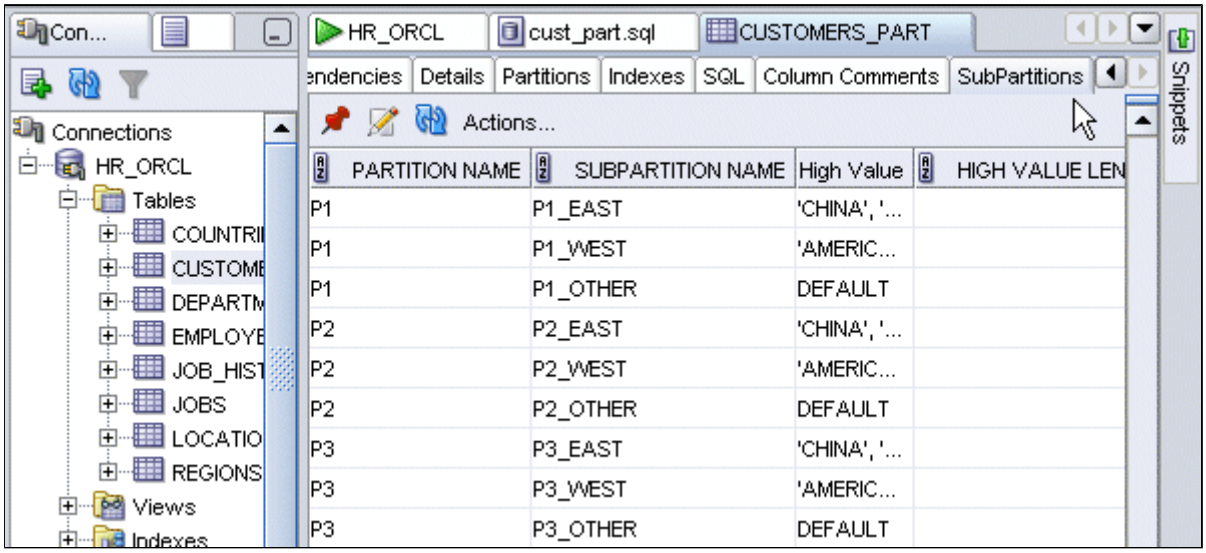
NOTE: If the tables node is already expanded, you need to click Refresh, to see the new table.



8. Select the Partitions Tab.



9. The new **SubPartitions** tab is added at the end of all the tabs. Navigate to the end of the tabs and review the new details.



10. If you have time, consider how you might add or remove columns from this SubPartitions tab.

HINT: The file is **\SQLDev_HOS\Extensions\files\subpartition_tab.xml**

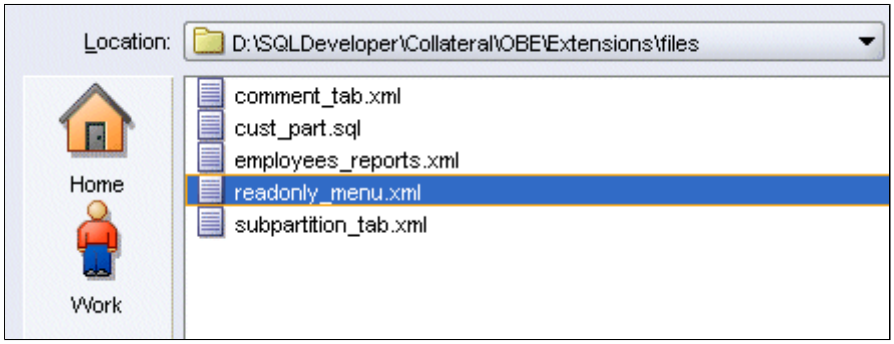
Back to Topic List

Adding a Context Menu

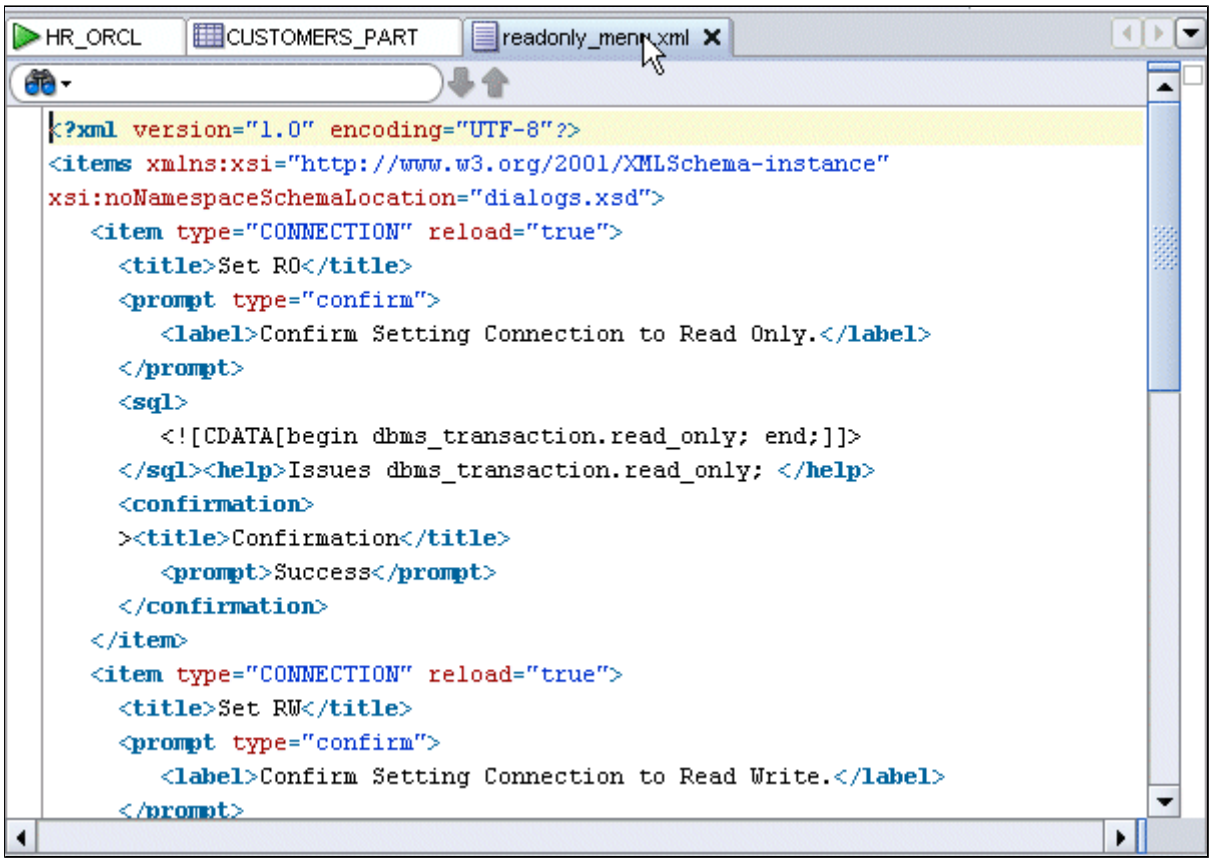
In the above exercises, you added tabs to the table definition, using XML files. You can add context menus using a similar approach. Context menus can be added at any level to the existing context menu structure. In this exercise you add a high-level context menu to set a transaction to READ ONLY.

To add a new context menu perform the following steps:

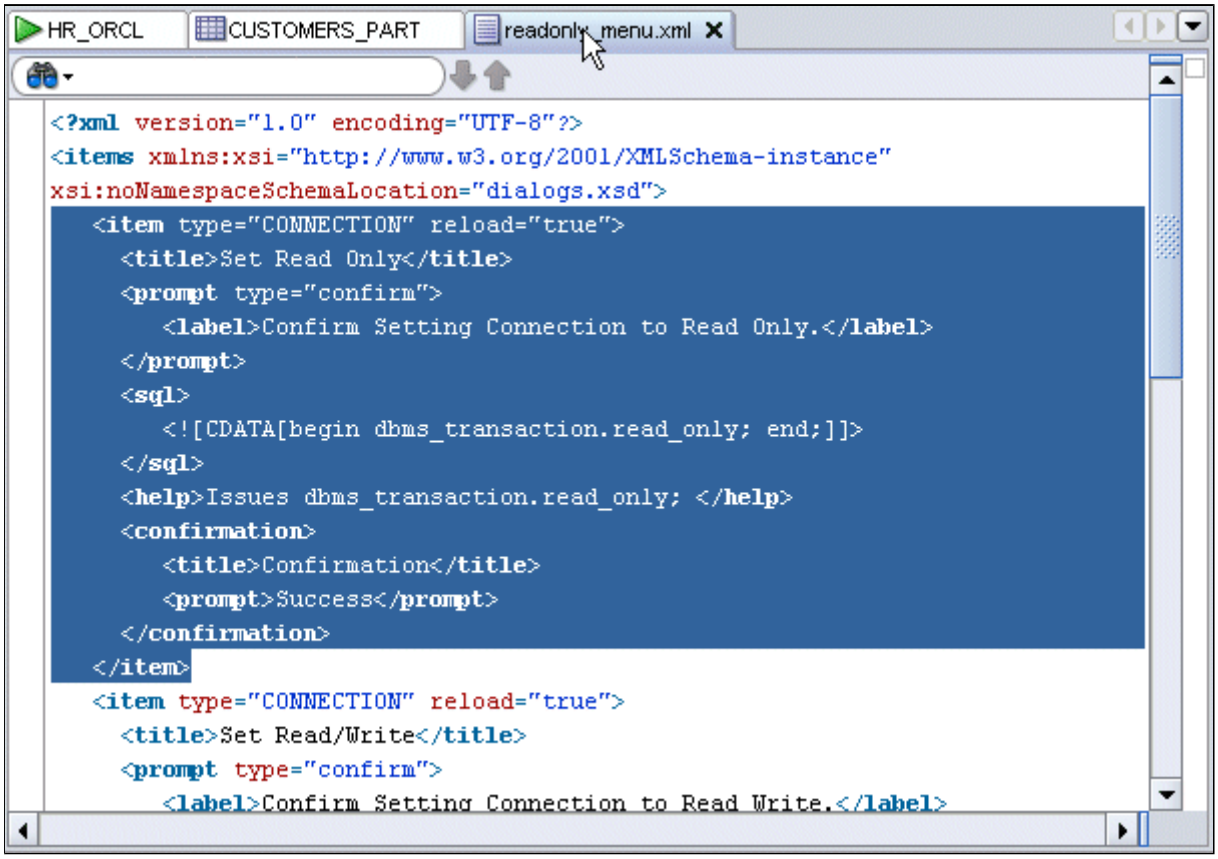
1. Select **File->Open**. Navigate to the **\SQLDev_HOS\Extensions\files** directory and select the **readonly_menu.xml** file.



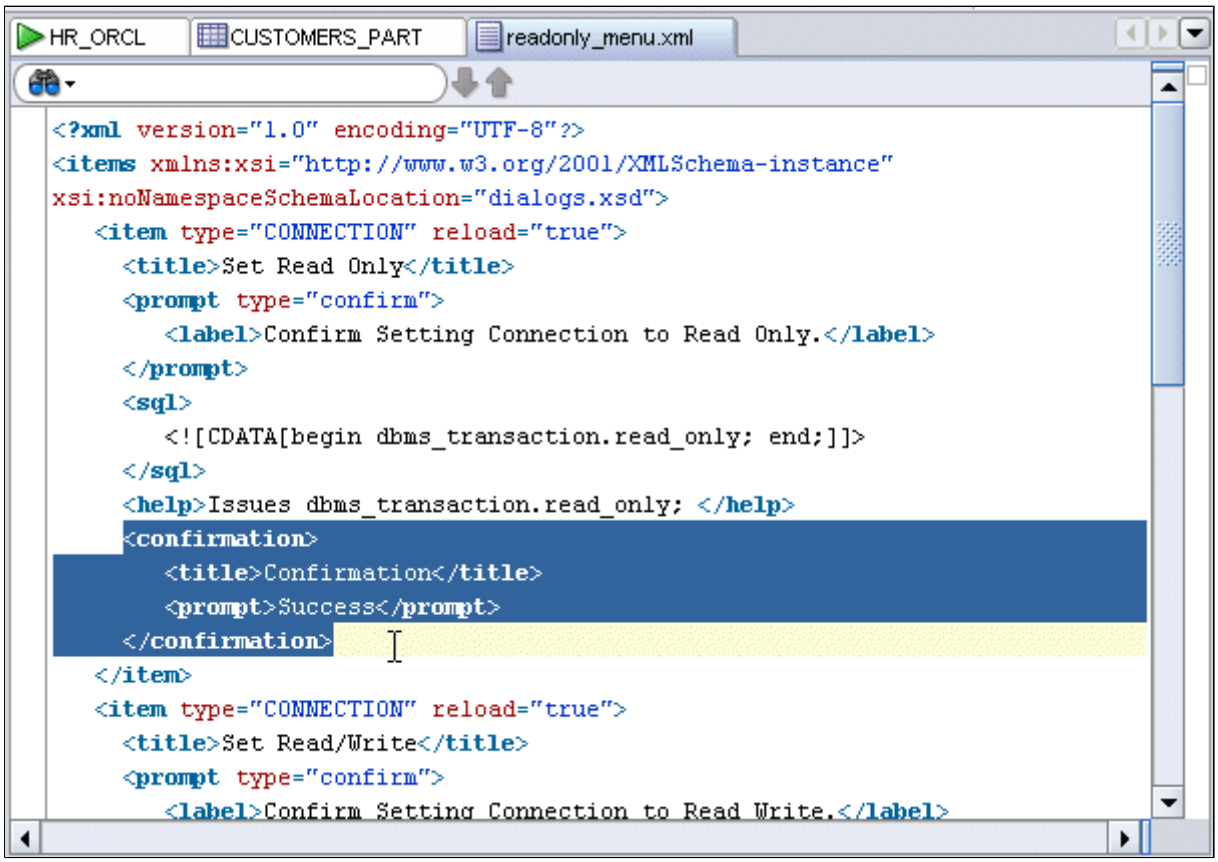
2. Double-click on the **readonly_menu.xml** tab to maximize the editor.



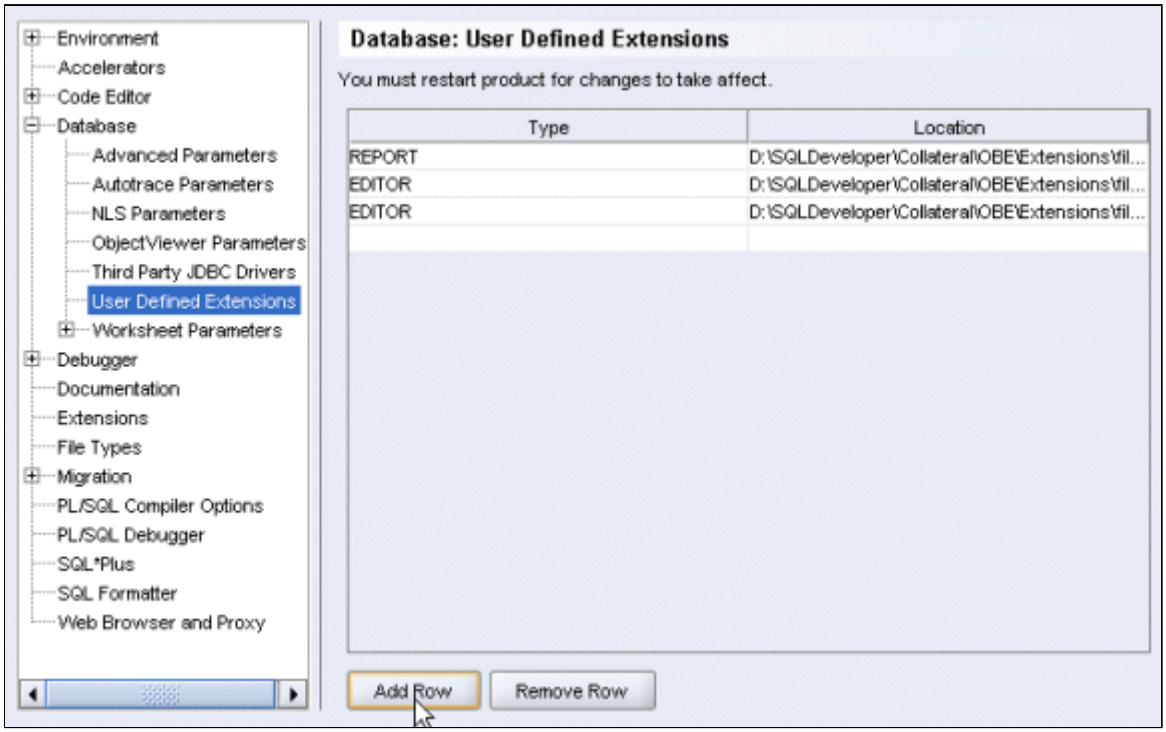
3. This XML file is more involved. Notice that in this case there are two sections; one for **Set Read Only** and the other for **Set Read/Write**. The image below highlights one of the sections between the <item> tags..



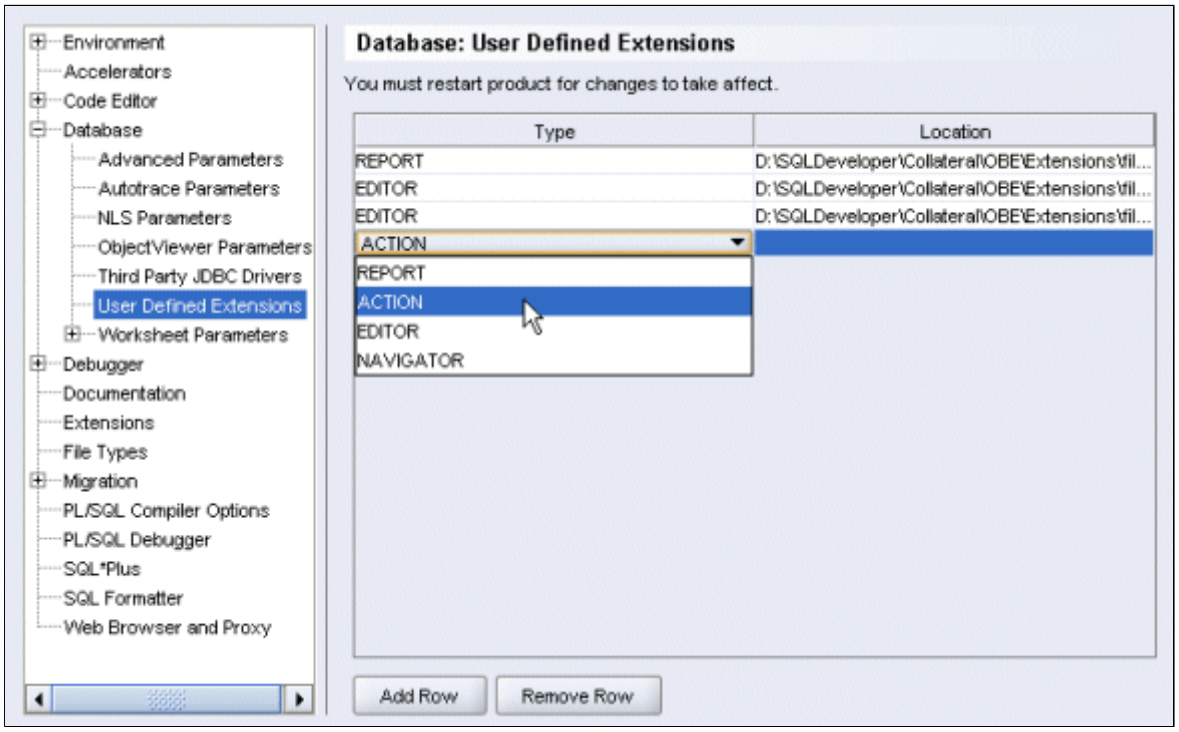
4. Notice the **<prompt>** and **<confirmation>** tags.



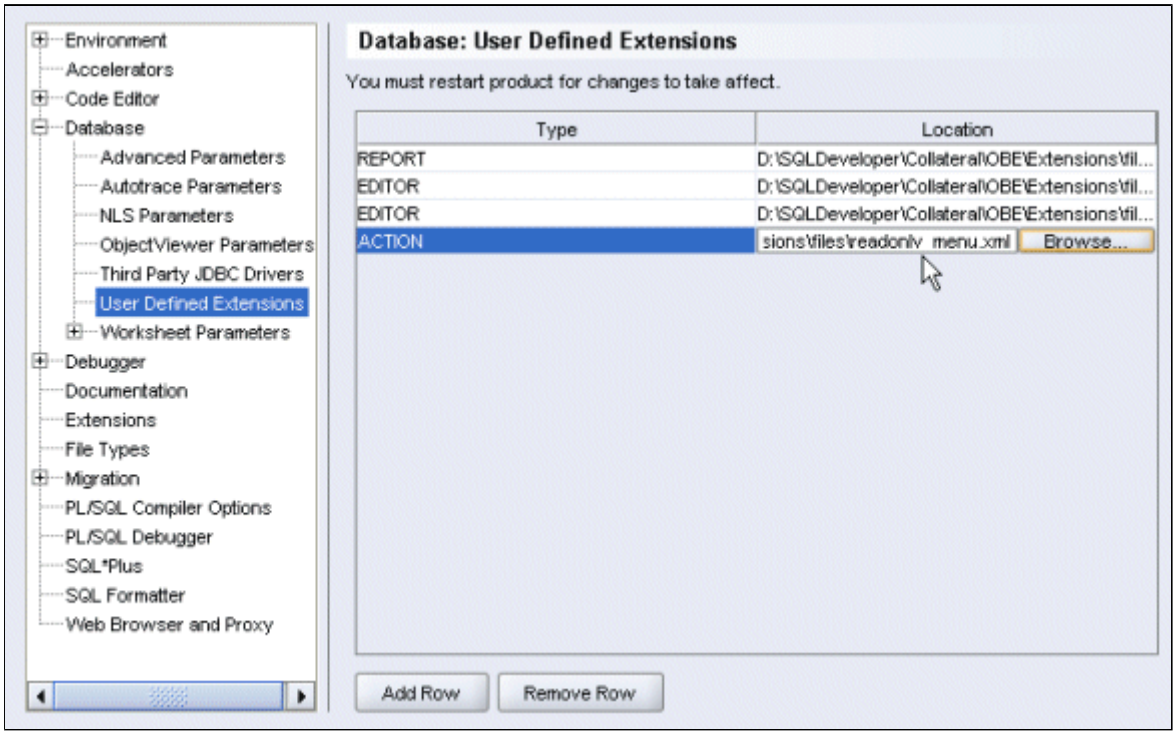
5. To add the extension to Oracle SQL Developer, select **Tools->Preferences**. Expand the **Database** node and select **User Defined Extensions**. Click **Add Row**.



6. Select **ACTION** from the **Extension Type** drop list.

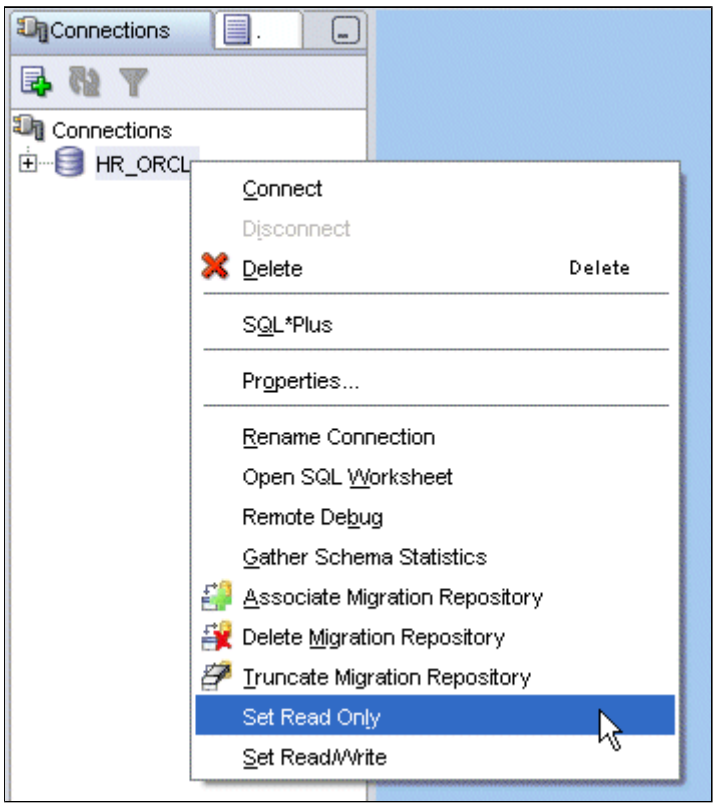


7. Select the Location field and browse to the **\SQLDev_HOS\Extensions\files** directory and select the **readonly_menu.xml** file. Ensure the file location is selected and click OK.



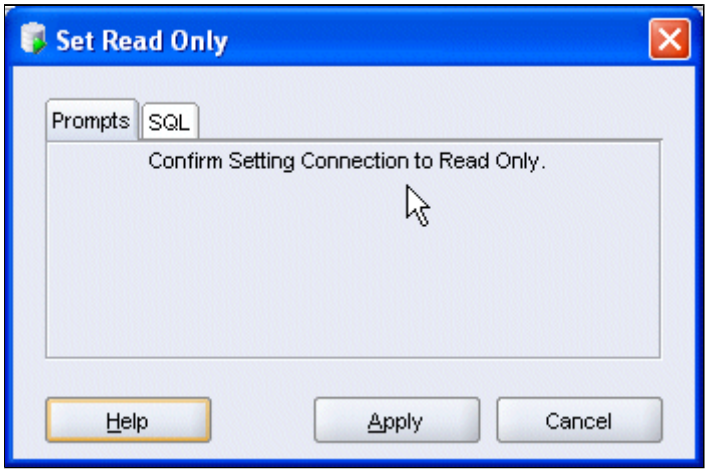
8. Shutdown and restart Oracle SQL Developer for your changes to take effect.

9. Select the **HR_ORCL** connection and right-click to invoke the context menu. Scroll to the bottom and select **Set Read Only.**

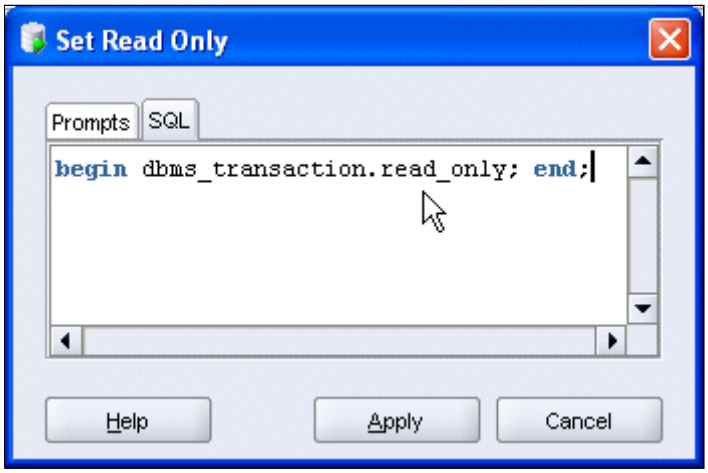


10. The title **"Set Read Only"**is displayed in the dialog header. Notice the Prompt. The text is from the

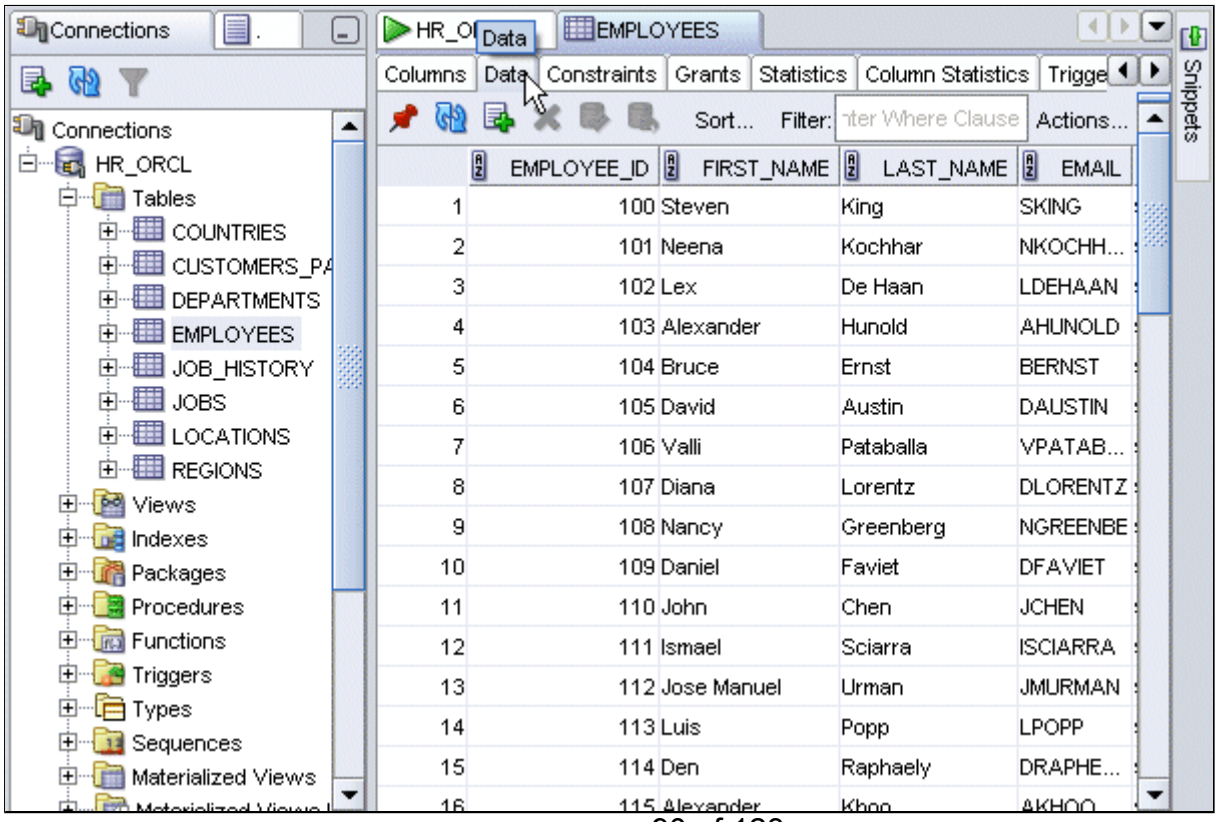
<prompt type="confirm">
<label>Confirm Setting Connection to Read Only.</label>
</prompt>



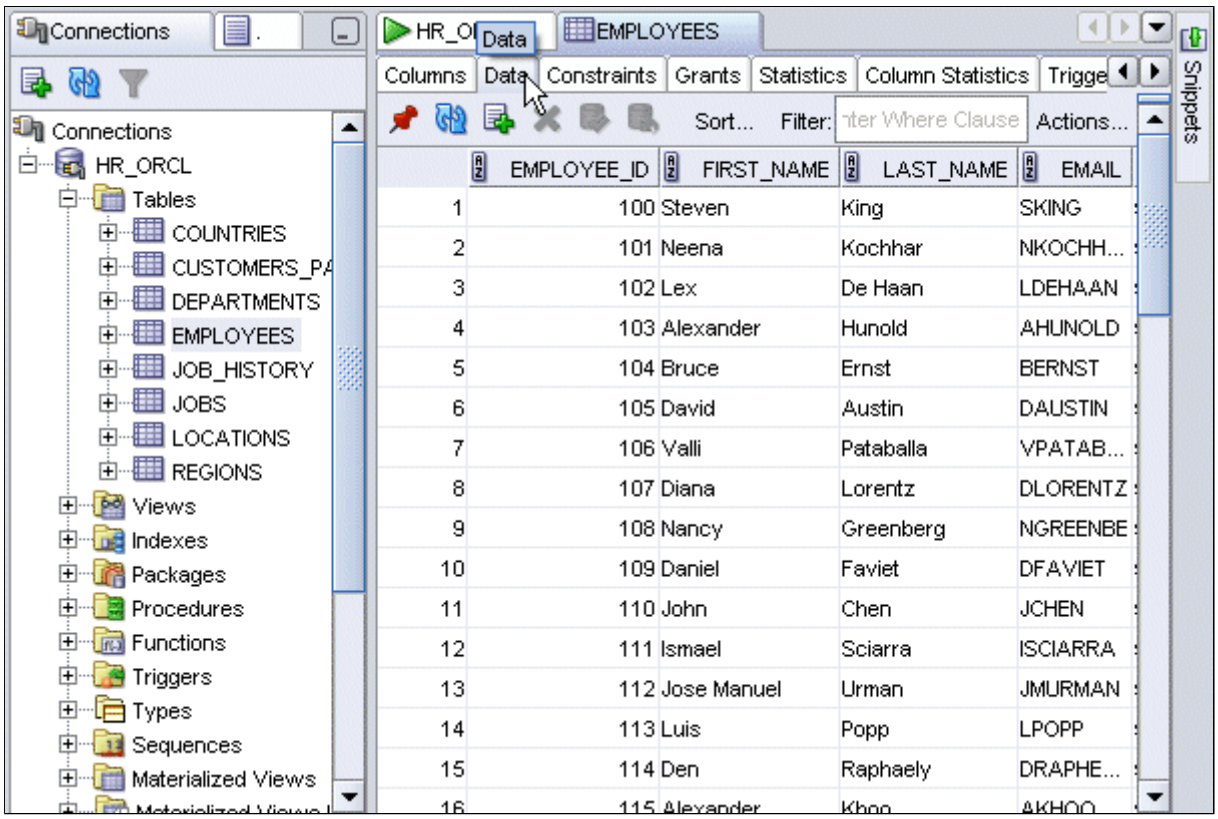
11. Select the **SQL** tab. Find the equivalent SQL code in the XML file. Click **Apply.** The final **Confirmation** dialog is also driven by the XML file.



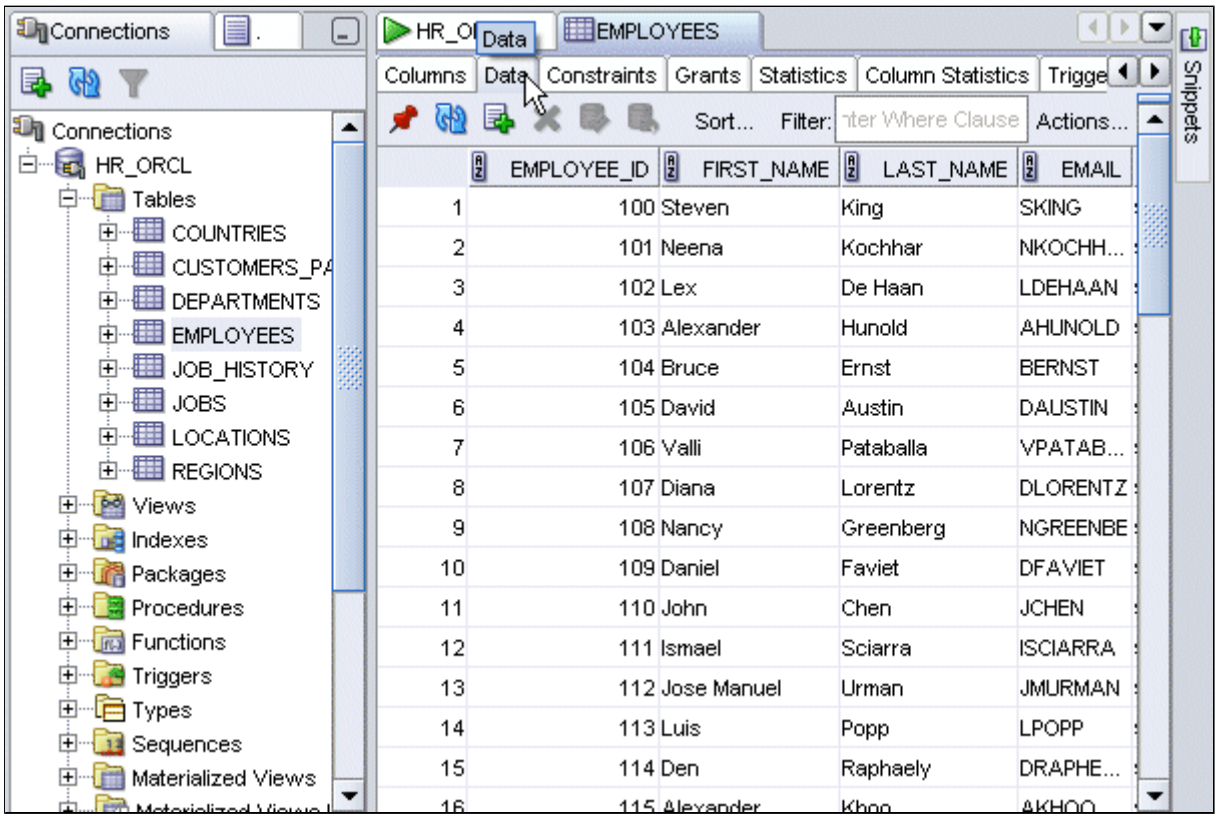
12. Expand the Tables node for the connection **HR_ORCL** and select **EMPLOYEES**. Select the data tab.



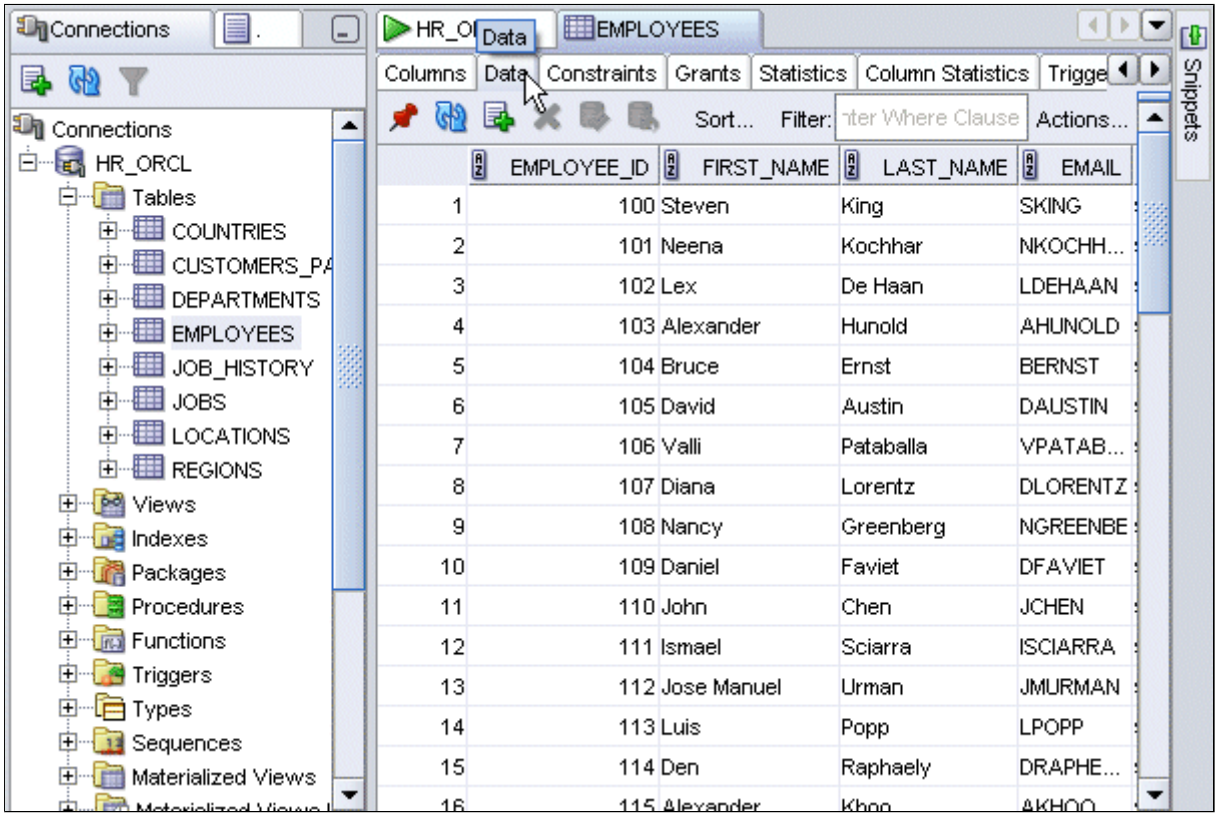
13. Modify the **FIRST_NAME** for one of the records.



14. Select the **Commit** icon to commit the changes.



15. You are not able to commit the records due to the **READ ONLY** transaction.



Back to Topic List

Summary

In this tutorial, you learned how to:

- **Creating a Database Connection**
- **Sharing Reports**
- **Adding an Extra Tab to Display Column Comments**
- **Reviewing the XML File structure**
- **Adding the Tab to describe Subpartitions**
- **Adding a Context Menu**

Back to Topic List

Move your mouse over this icon to hide all screenshots.

Migrating a Microsoft Access Database to Oracle

Purpose

This tutorial describes how to migrate a Microsoft Access Database to Oracle.

Time to Complete

Approximately 30 minutes

Topics

This tutorial covers the following topics:

Viewing Screenshots

Place the cursor over this icon to load and view all the screenshots for this tutorial. (Caution: This action loads all screenshots simultaneously, so response time may be slow depending on your Internet connection.)

Note: Alternatively, you can place the cursor over an individual icon in the following steps to load and view only the screenshot associated with that step. You can hide an individual screenshot by clicking it.

Overview

Using Oracle SQL Developer Migration Workbench, you can quickly migrate your Microsoft Access database to Oracle. There are four main steps in the database migration process:

Capture the Source Database	The Northwind.xml file generated by the Exported tool contains the database schema information for the Microsoft Access Northwind Traders database. Oracle SQL Developer Migration Workbench uses this file as the basis for building a representation of the structure of the source Microsoft Access database. This structure is called the Captured Model.
Convert the Captured Database	Oracle SQL Developer Migration Workbench uses the Captured Model to convert the captured objects to Oracle-format objects, building up a representation of the structure of the destination database. This structure is called the Converted Model.

Generate the Oracle Database	Oracle SQL Developer Migration Workbench generates DDL statements to create the new Oracle database, based upon the Converted Model objects. Running the DDL statements will result in the creation of the objects in the Oracle database.
Migrate the Data	The last step in the process is to migrate the data. You can do this in one of two ways. You can create a connection from within Oracle SQL Developer to your Microsoft Access source database and migrate the data, or you can export the data from Microsoft Access. Microsoft Access will create a series of sqlldr files that you can run from a batch file. The second method is what you will perform in this tutorial.

Back to Topic List

Prerequisites

For this Hands On Session, the following has already been installed for you

1. Oracle Database 10g.
   Note: To repeat this exercise later you can use any Oracle Database above 9.2.0.1

2. Oracle SQL Developer 1.2.1.
   Note: Oracle SQL Developer is available for download for FREE from **OTN.** To install Oracle SQL Developer, unzip it into any directory on your machine.

3. The files you use throughout the tutorial are located in the **\SQLDev_HOS\Migration\files** directory. The files have also been included in the msamigrate.zip file.

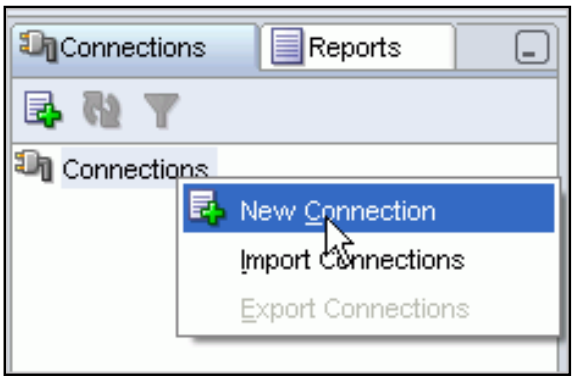
Back to Topic List

Creating the OMWBREP User

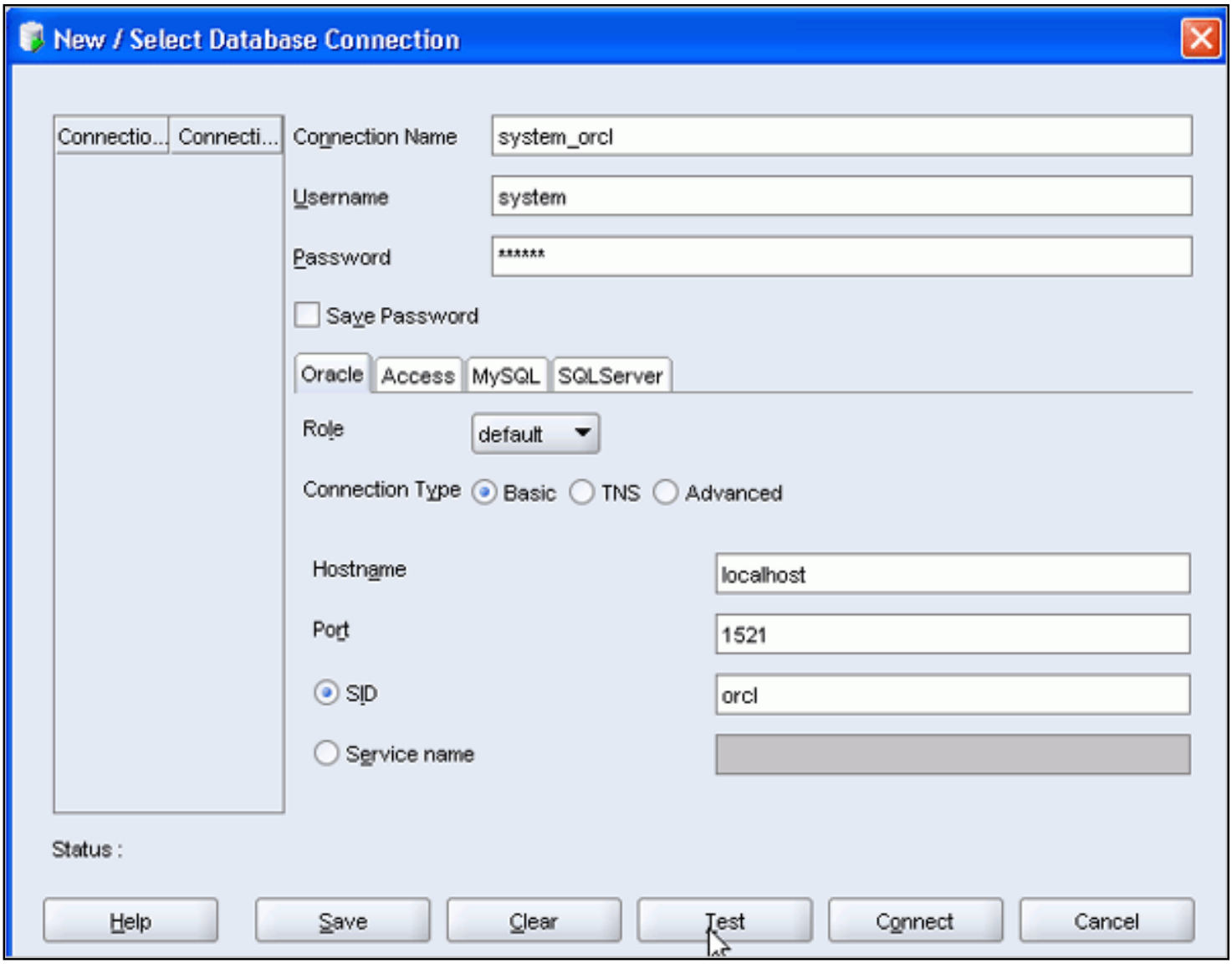
The first step to managing database objects using Oracle SQL Developer is to create a database connection. Perform the following steps:

1. From the directory where you installed Oracle SQL Developer, double-click on **sqldeveloper.exe**.

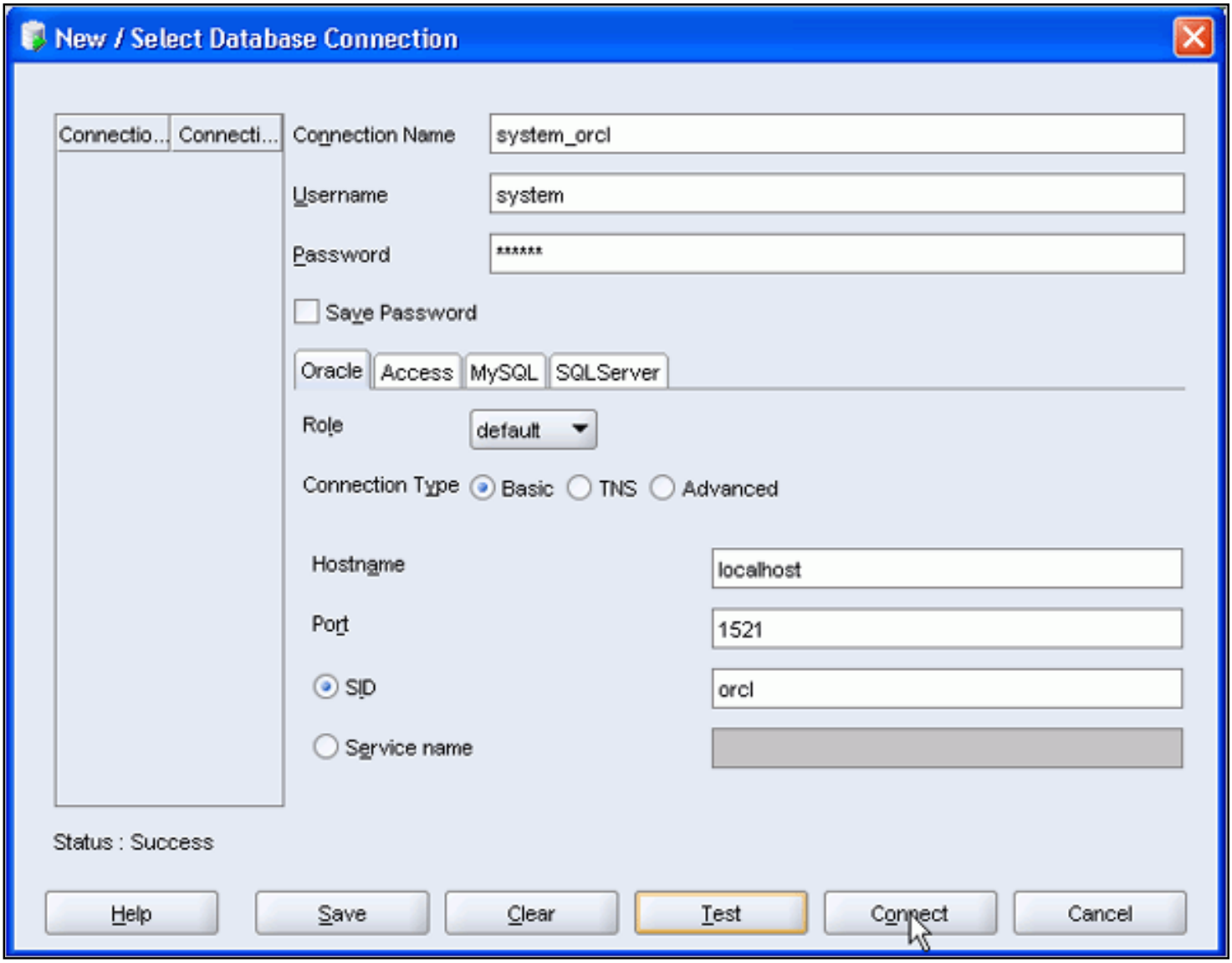
2. In the Connections tab, right-click **Connections** and select **New Connection**.



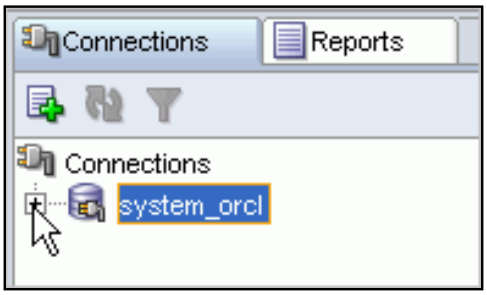
3. Enter **system_orcl** for the Connection Name (or any other name that identifies your connection), **system** for the Username and **<your_password>** for the Password, specify your <hostname> for the Hostname and enter **orcl** for the SID. Then click **Test**.



4. The status of the connection was tested successfully. The connection was not saved however. To save the connection, click **Connect**.

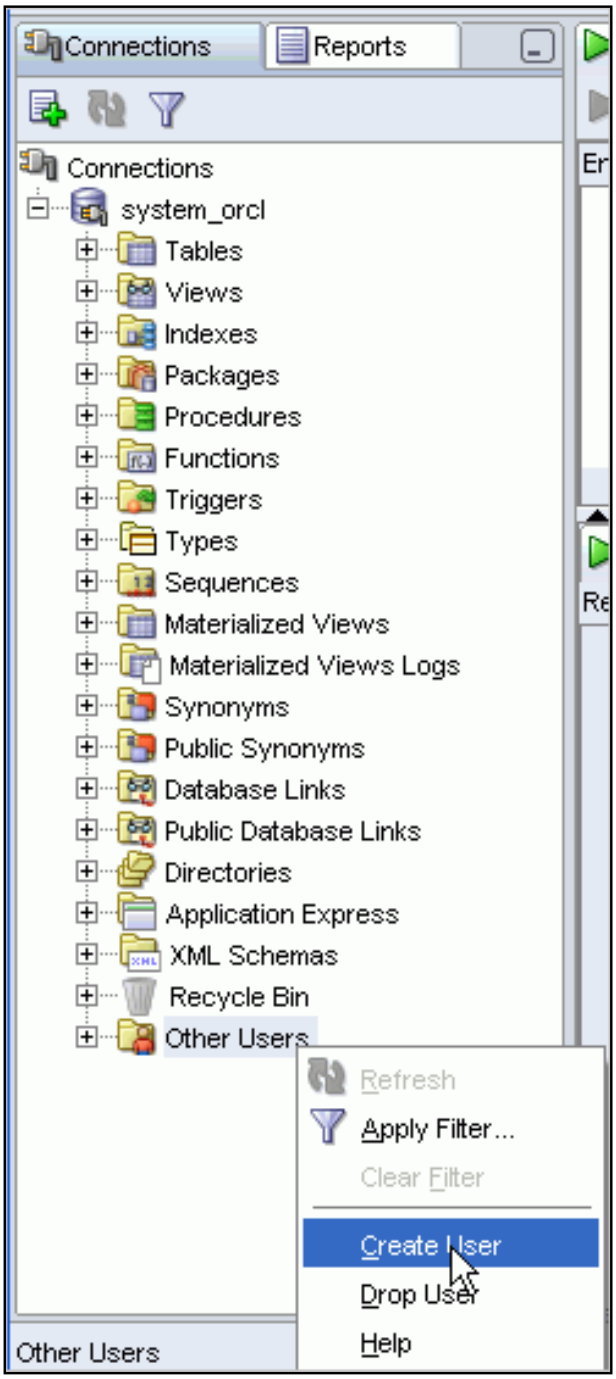


5. The connection was saved and you see the database in the list. Expand **system_orcl**.

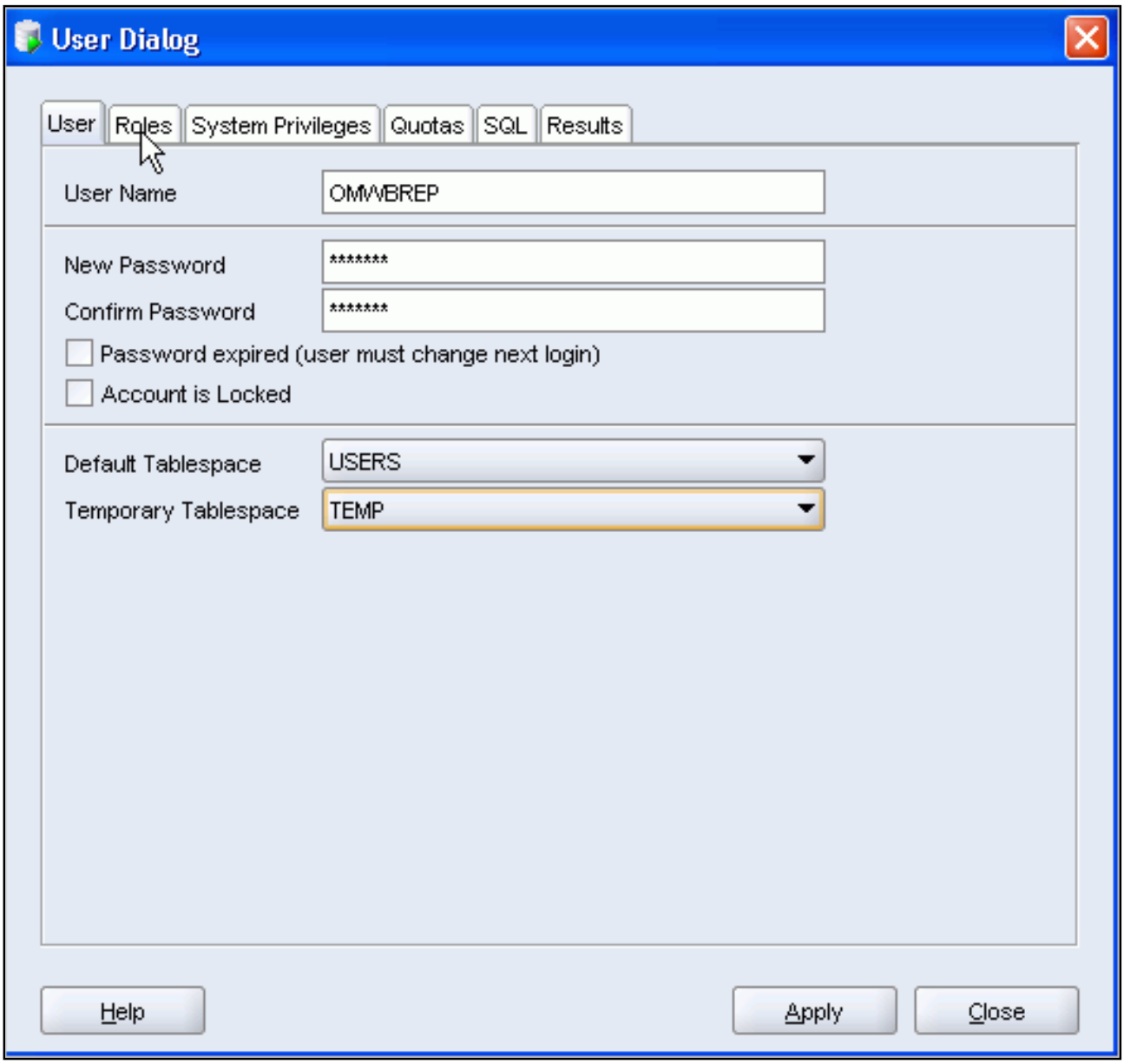


Note: When a connection is opened, a SQL Worksheet is opened automatically. The SQL Worksheet allows you to execute SQL against the connection you just created.

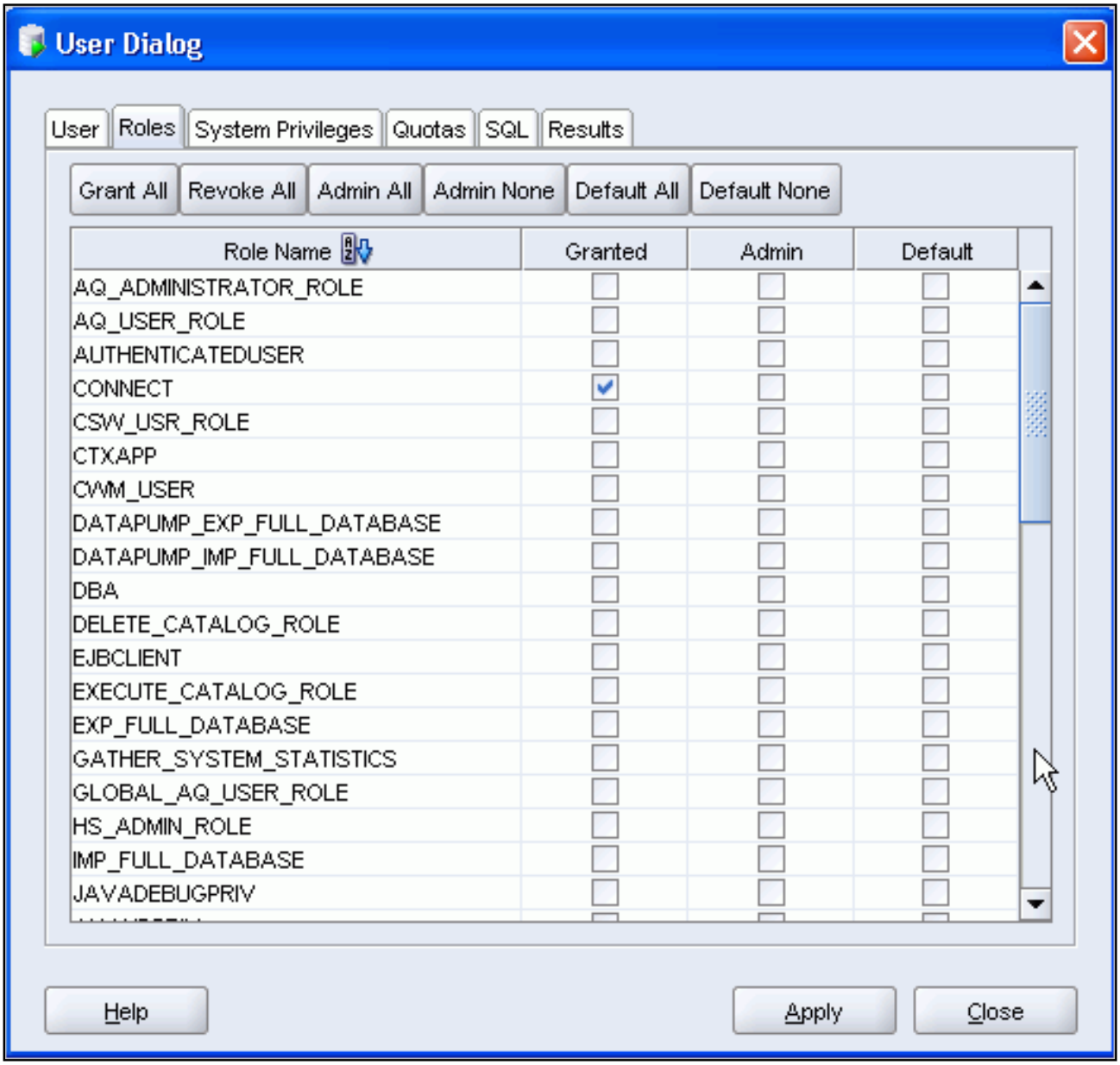
6. Right click on **Other Users** and select **Create User.**



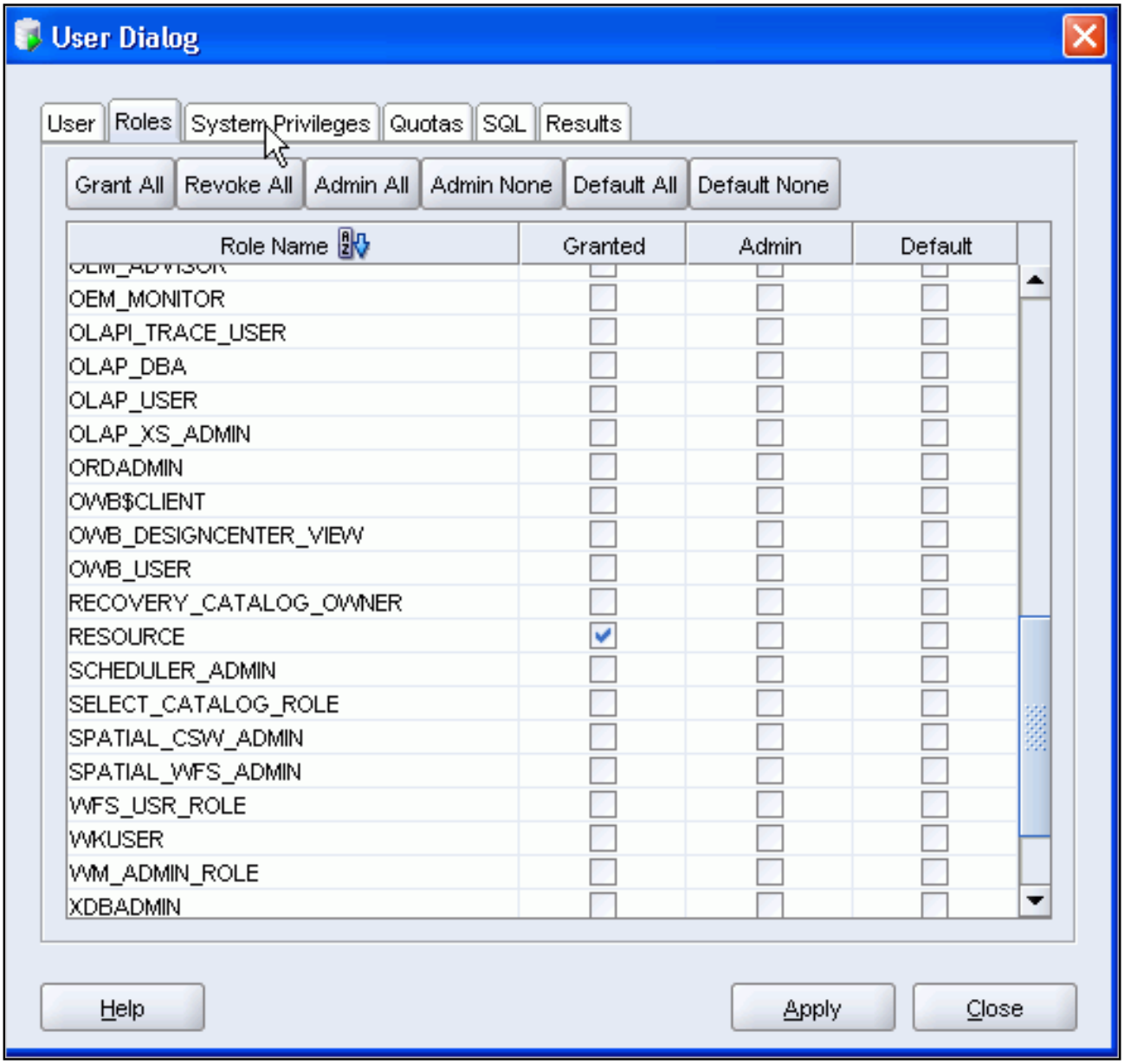
7. Enter **OMWBREP** for the User Name, **omwbrep** for the Password and Confirm Password, **USERS** for the Default Tablespace and **TEMP** for the Temporary Tablespace. Then click the **Roles** tab.



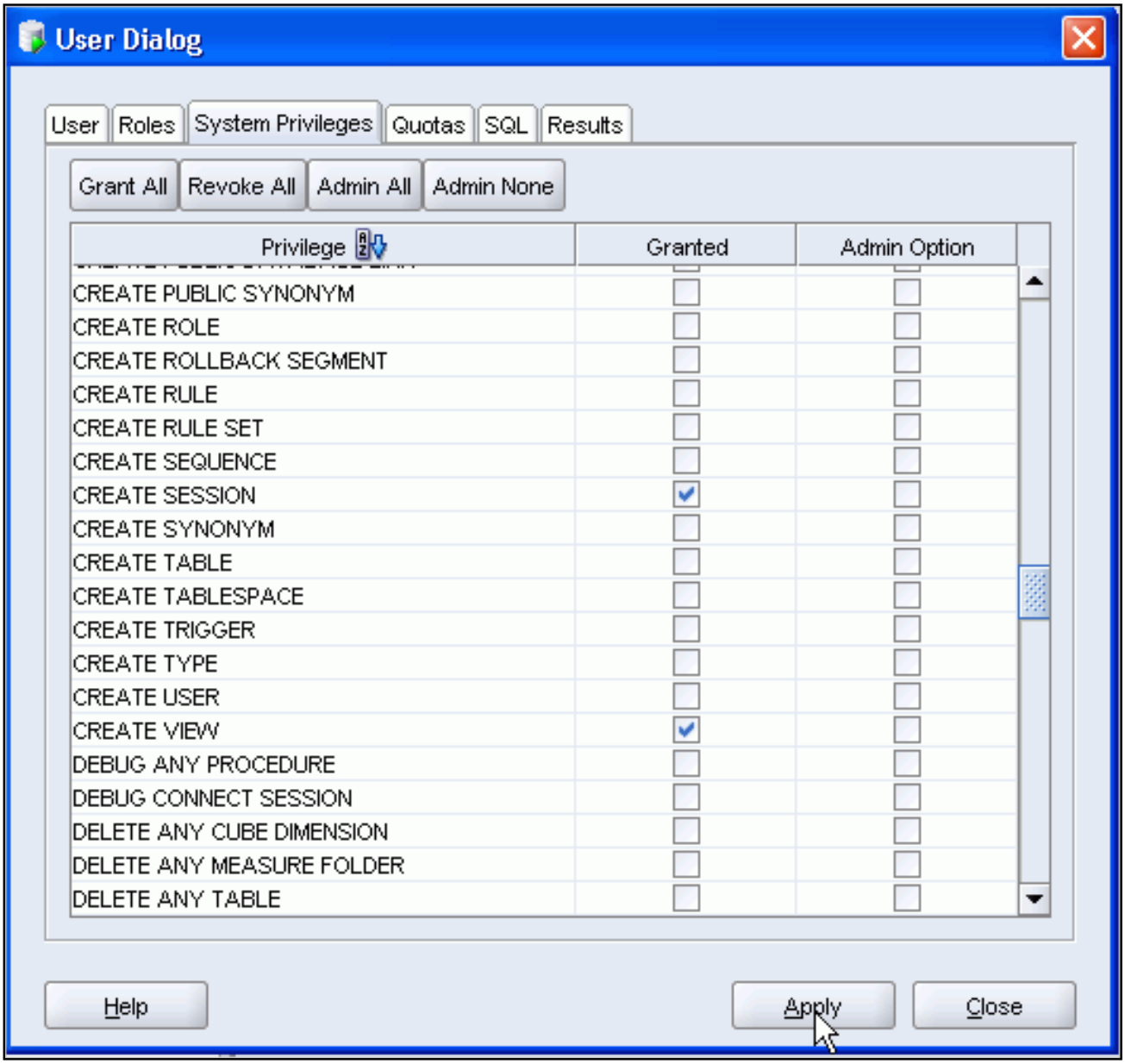
8. Select the Granted checkbox for the **CONNECT** role and scroll down in the list.



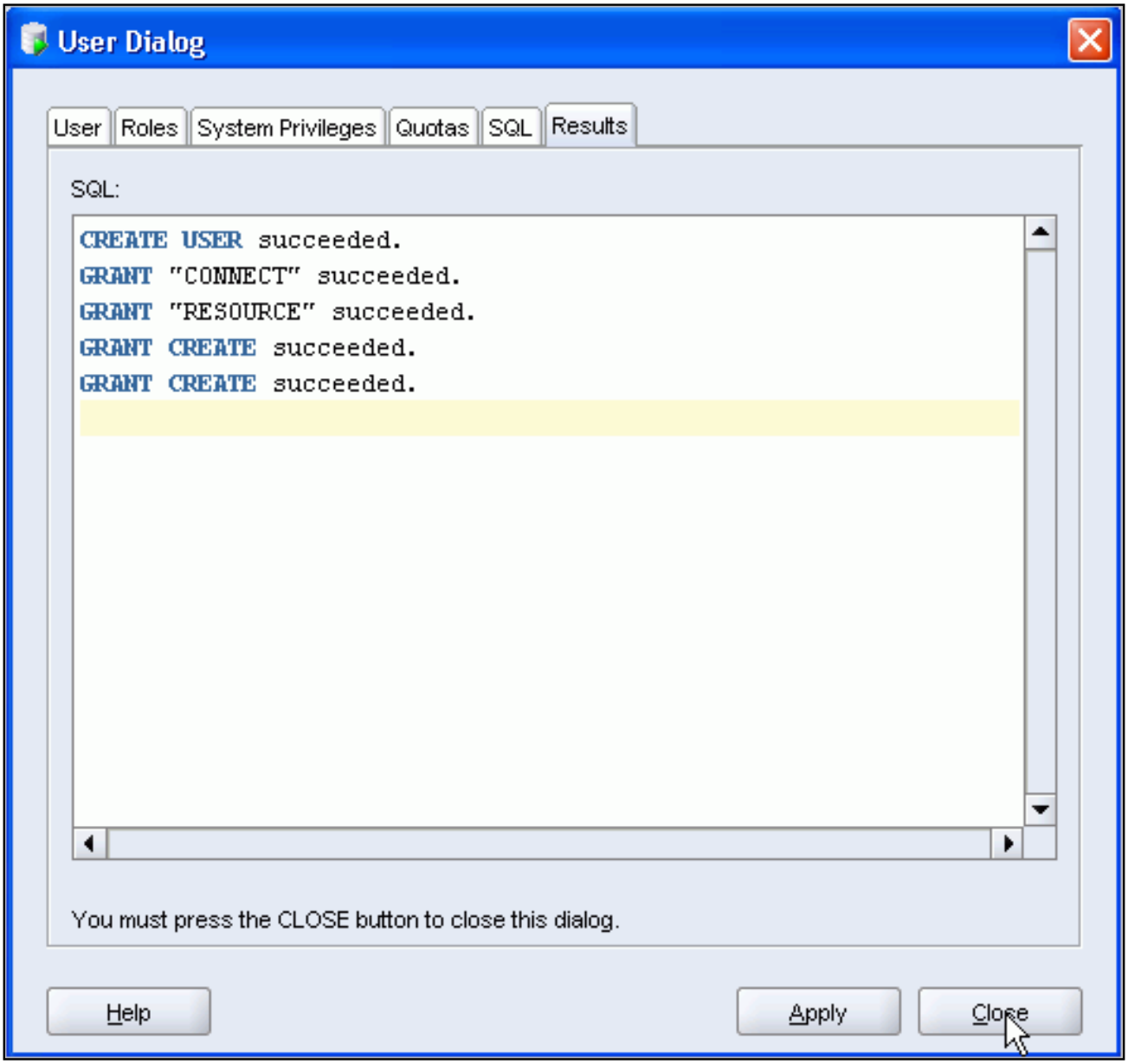
9. Select the Granted checkbox for the **RESOURCE** role and click the **System Privileges** tab.



10. Select the Granted checkbox for the **CREATE SESSION** and **CREATE VIEW** privileges and click **Apply**.



11. The OMWBREP user was created successfully.

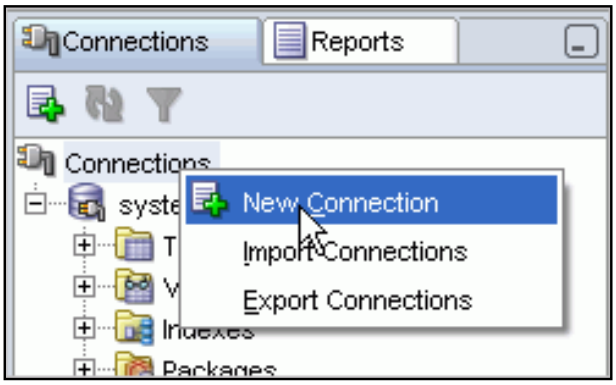


Back to Topic List

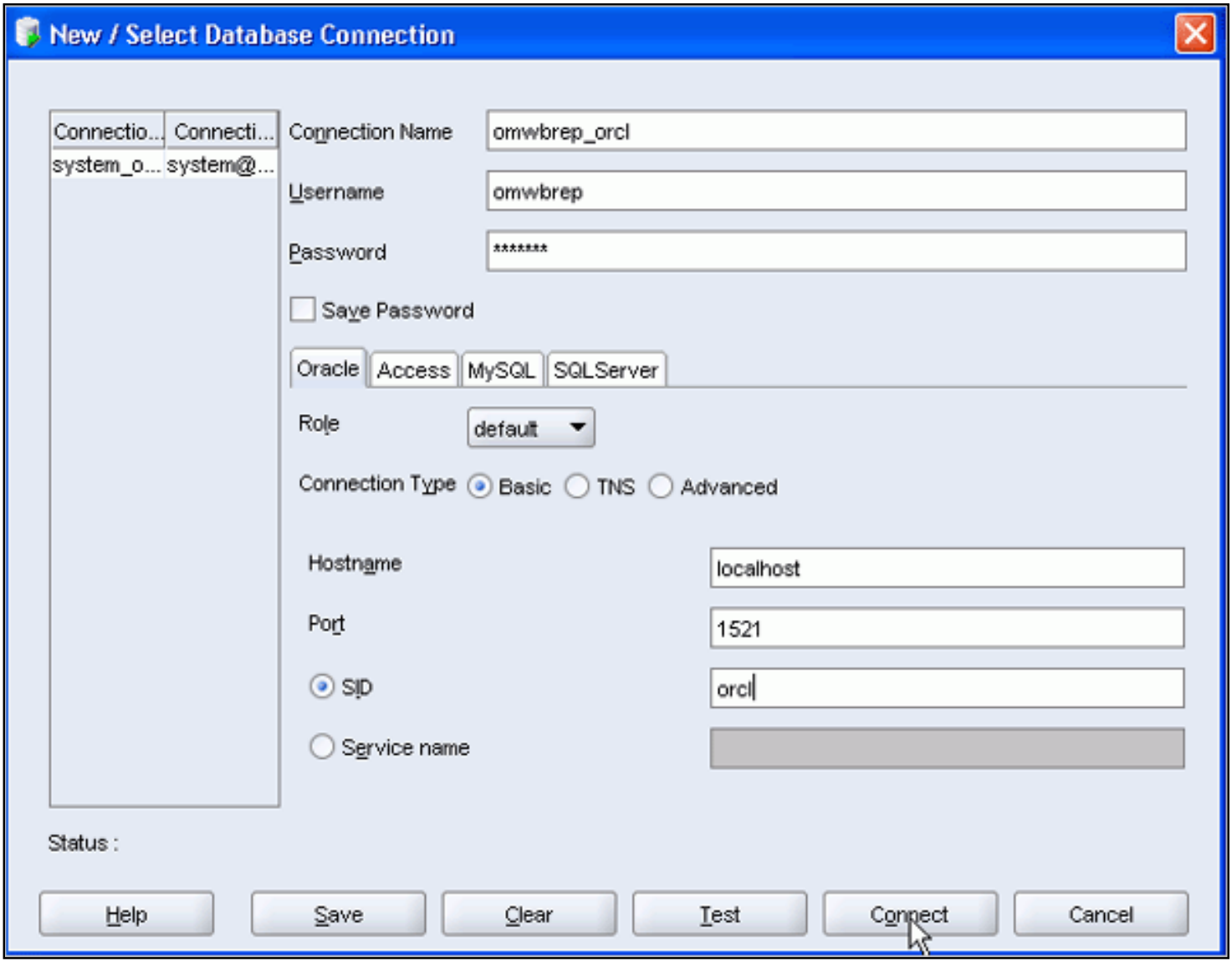
Creating the Migration Repository

In order for the Microsoft Access XML to be converted to Oracle, you need to create a repository to store the required repository tables and PL/SQL packages. Perform the following steps:

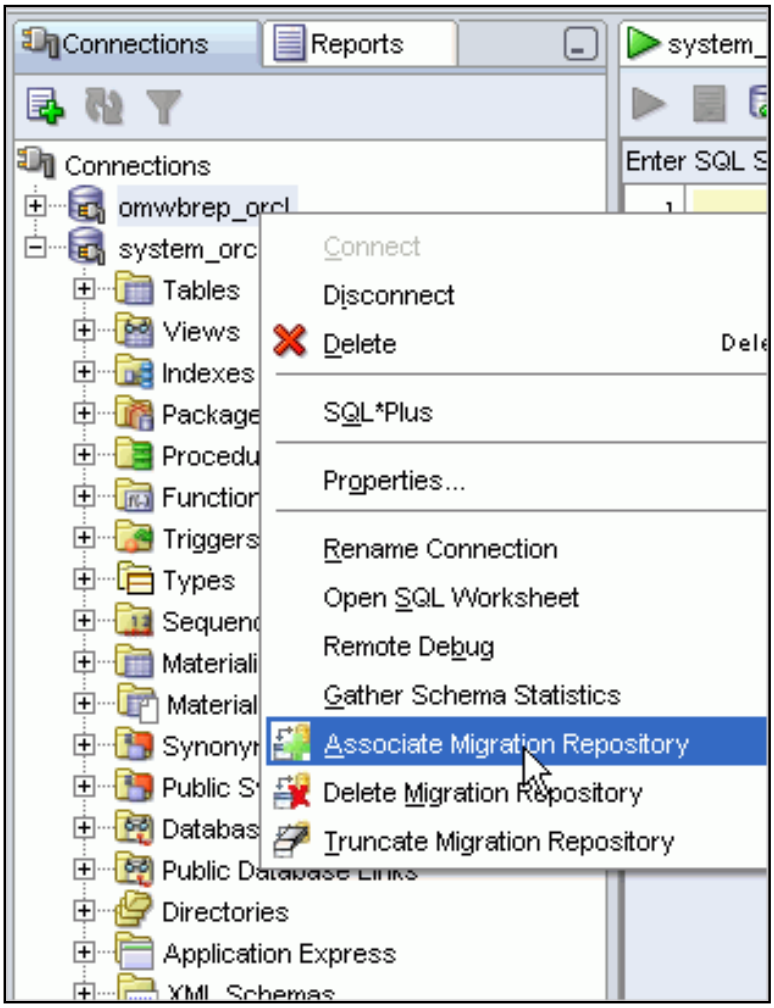
1. Before you create the repository, you need to create a connection to the OMWBREP user. Right click **Connection** and select **New Connection**.



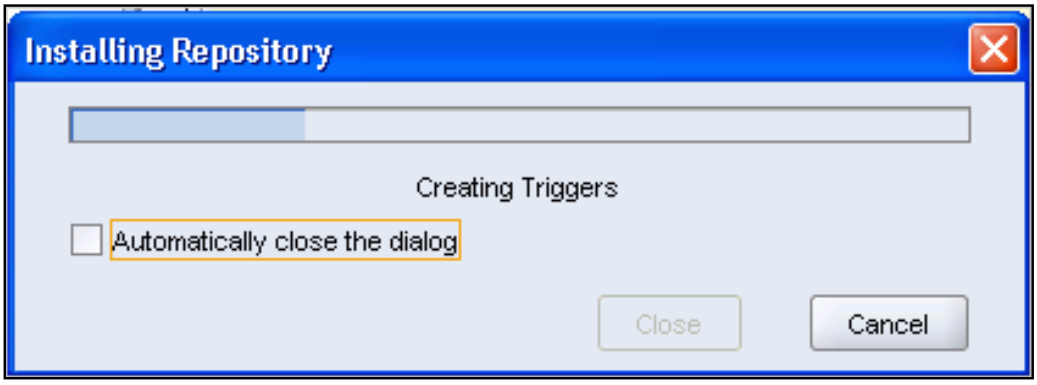
2. Enter **omwbrep_orcl** for the Connection Name (or any other name that identifies your connection), **omwbrep** for the Username and **omwbrep** for the Password and enter **orcl** for the SID. Then click **Connect**.



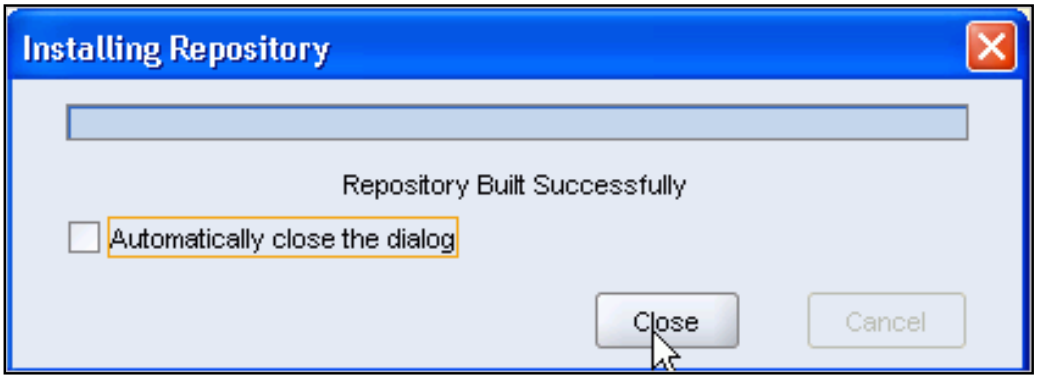
3. Now you can create the repository in this users connection. Right click on the **omwbrep_orcl** connection and select **Associate Migration Repository**.



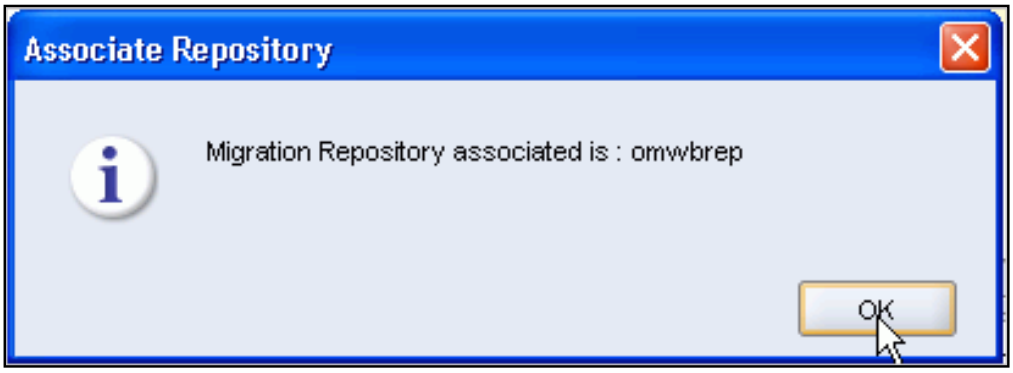
4. The progress window appears.



5. Once the repository is built successfully, click **Close**.



6. Click **OK**.

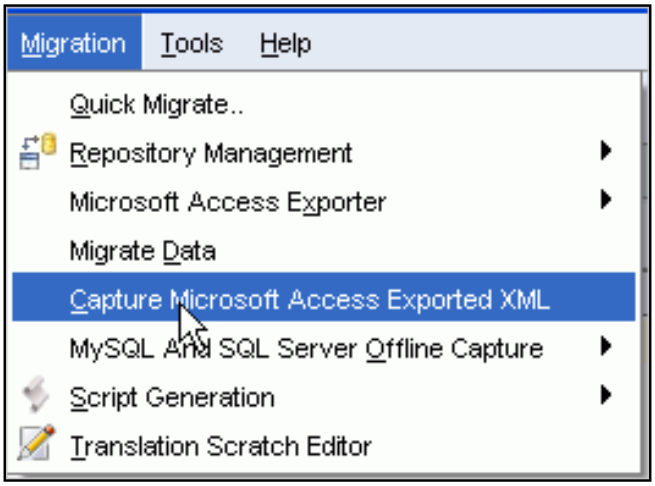


Back to Topic List

Capturing the Microsoft Access Exported XML

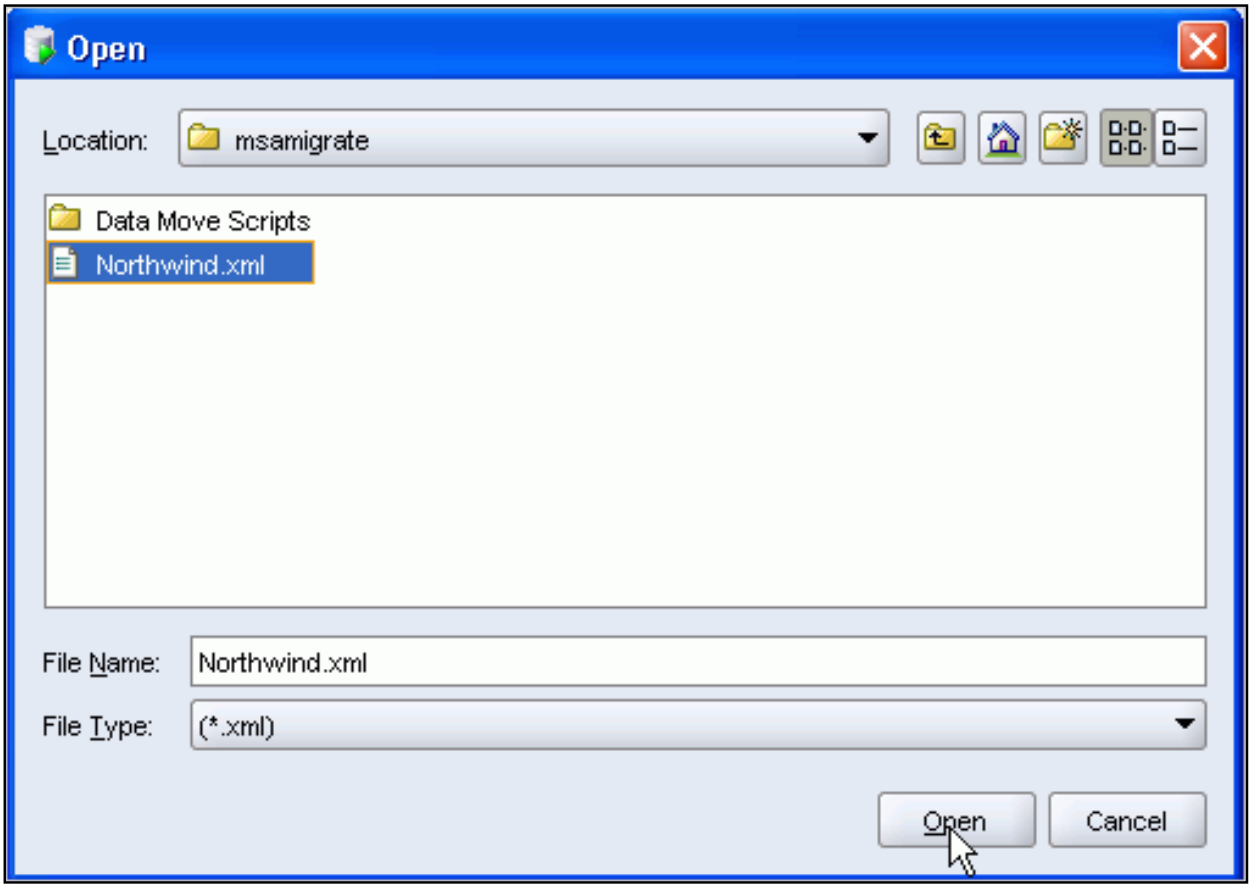
At this point, you can capture the Microsoft Access Exported XML into Oracle SQL Developer. Perform the following steps:

1. Select **Migration > Capture Microsoft Access Exported XML**.



2. Click **Browse...** to find the XML file. The files are in the **\SQLDev_HOS\Migration\files** directory.

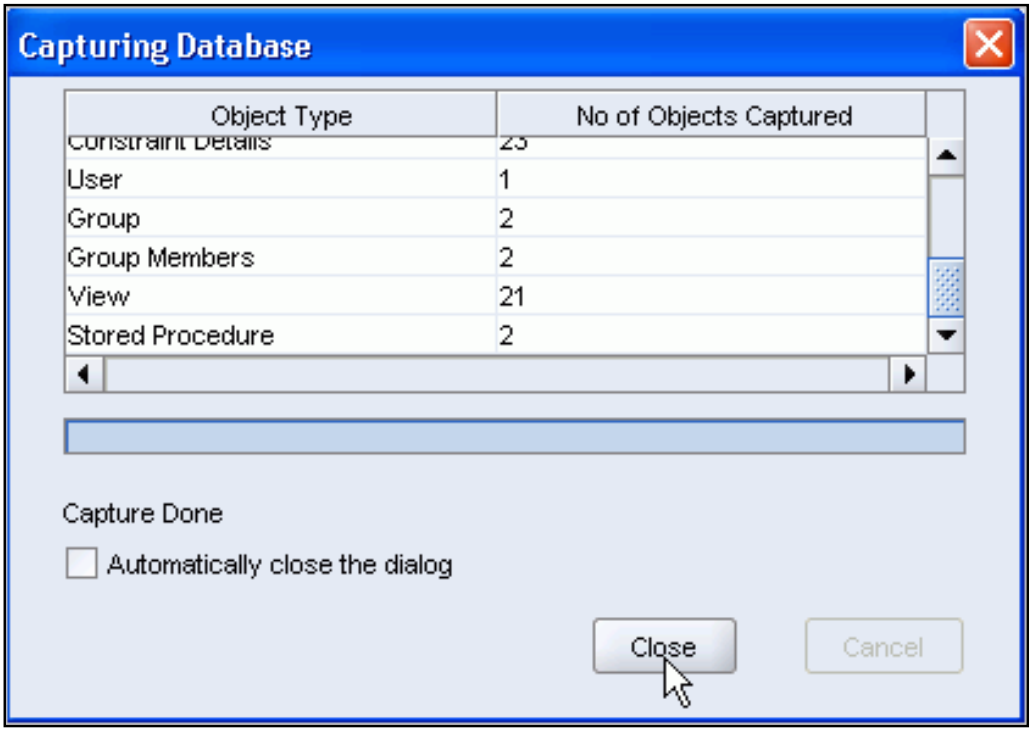
3. Select **Northwind.xml** from the list and click **Open**.


Open
Location:
msamigrate
Data Move Scripts
Northwind.xml
File Name:
Northwind.xml
File Type:
(*.xml)
Open
Cancel

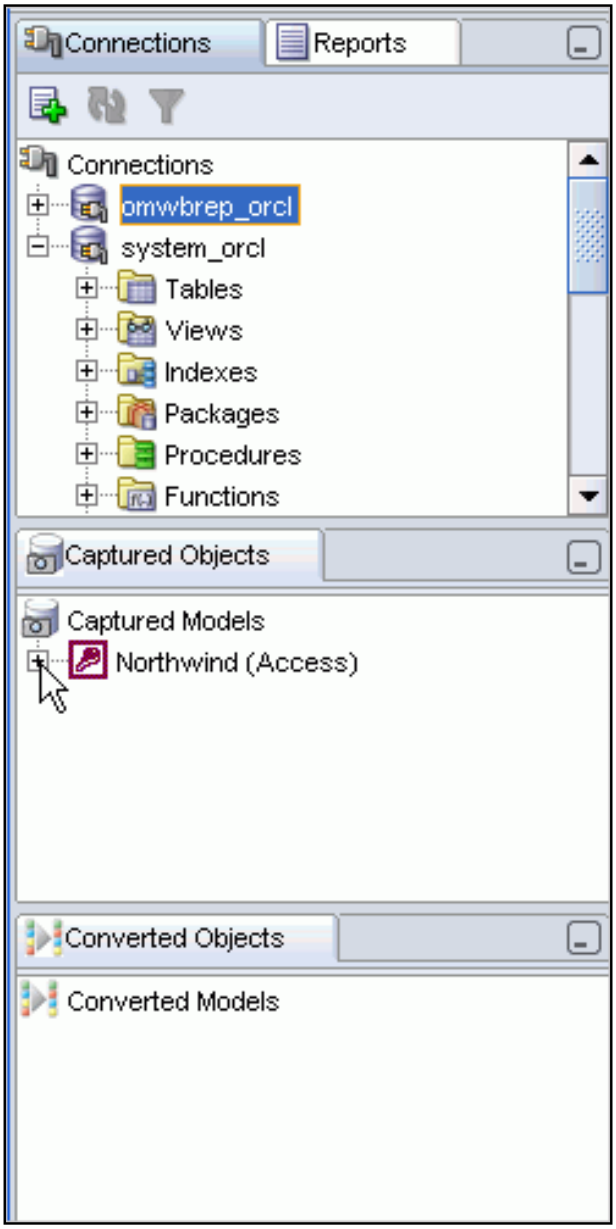
4. Click **OK**.


Microsoft Access Capture
Please select the XML file that represents the Microsoft Access database you would like to migrate to Oracle. The XML file should be generated by activating the appropriate version of the Exporter for Microsoft Access.
File Path:
C:\wkdir\msamigrate\Northwind.xml
Browse...
OK
Cancel

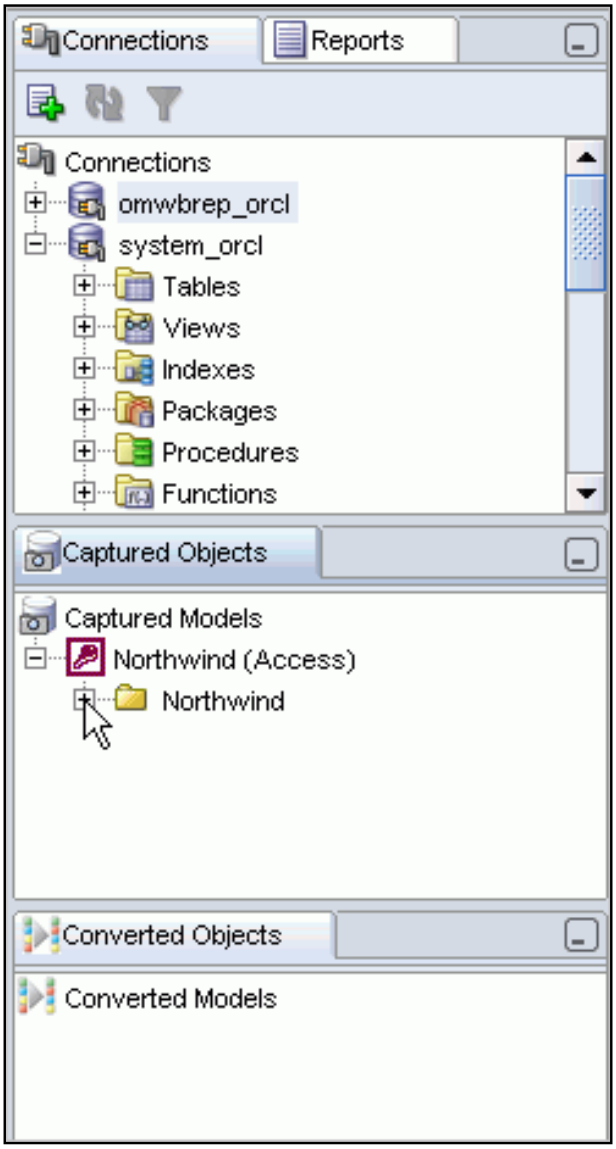
5. The objects are being captured. When done, click **Close**.



6. Under Captured Models, expand **Northwind (Access)**.



7. Expand **Northwind** to see the list of objects that were captured.

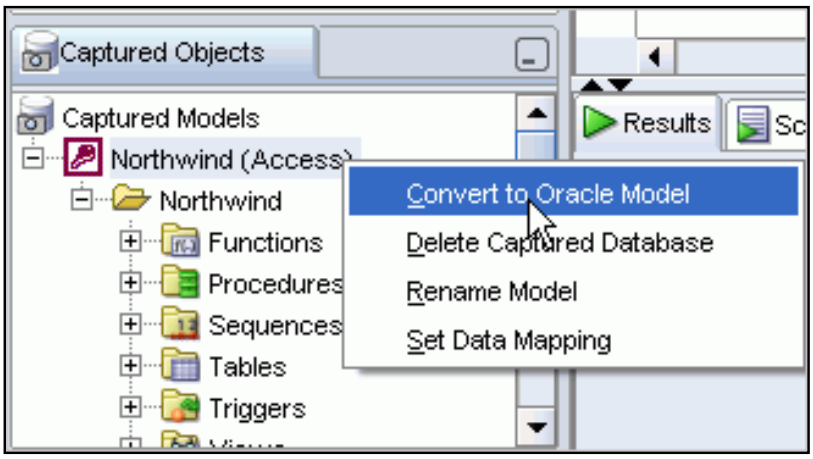


Back to Topic List

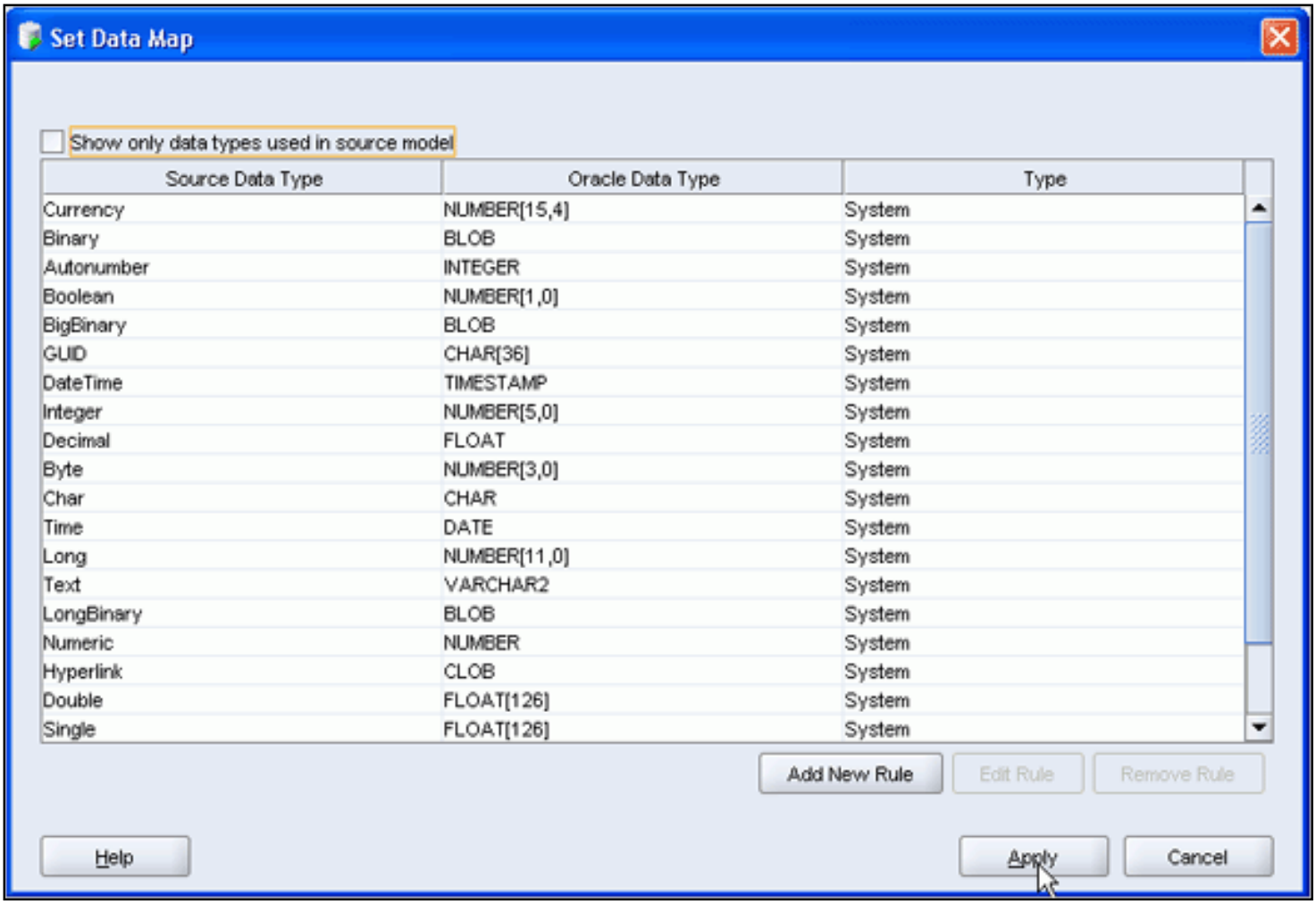
Converting to the Oracle Model

The next step is to convert the captured model to an Oracle model. Perform the following steps:

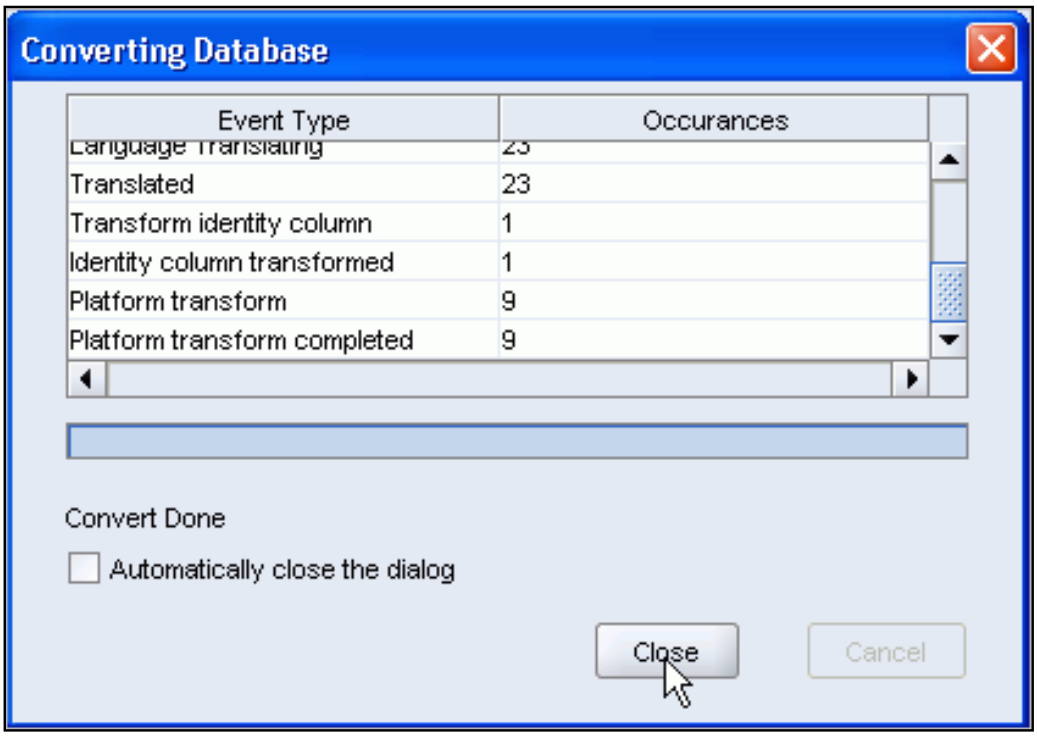
1. Right-click the captured model **Northwind (Access)** and select **Convert to Oracle Model.**



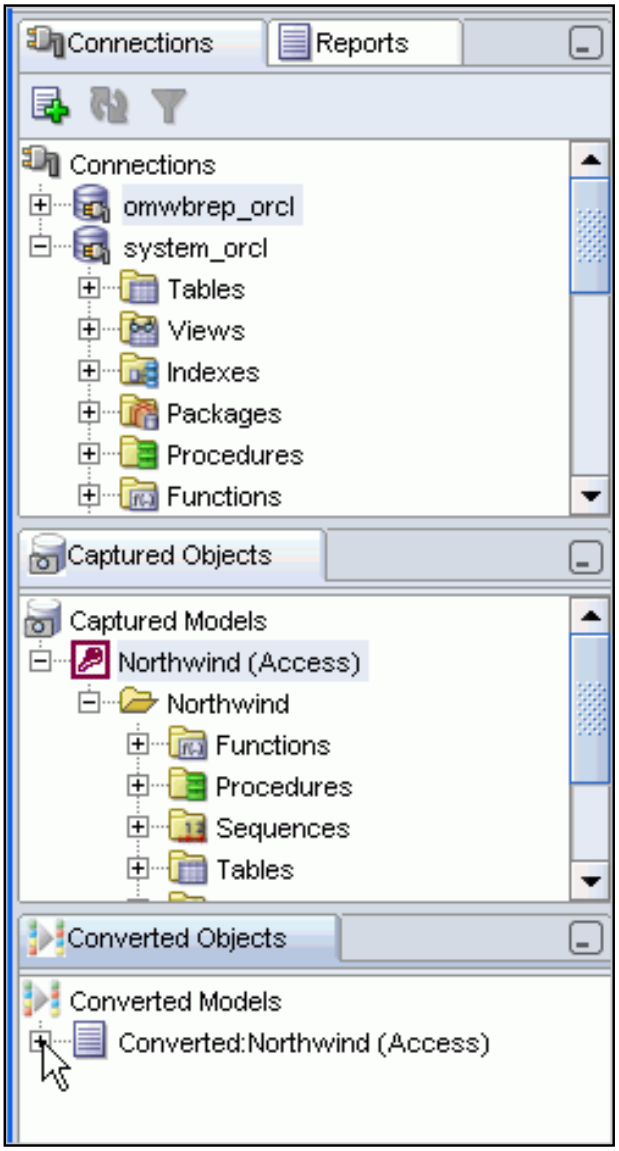
2. The Set Data Map window appears which shows you the Source Data Type and what it will be converted to in the Oracle Model. Click **Apply**.



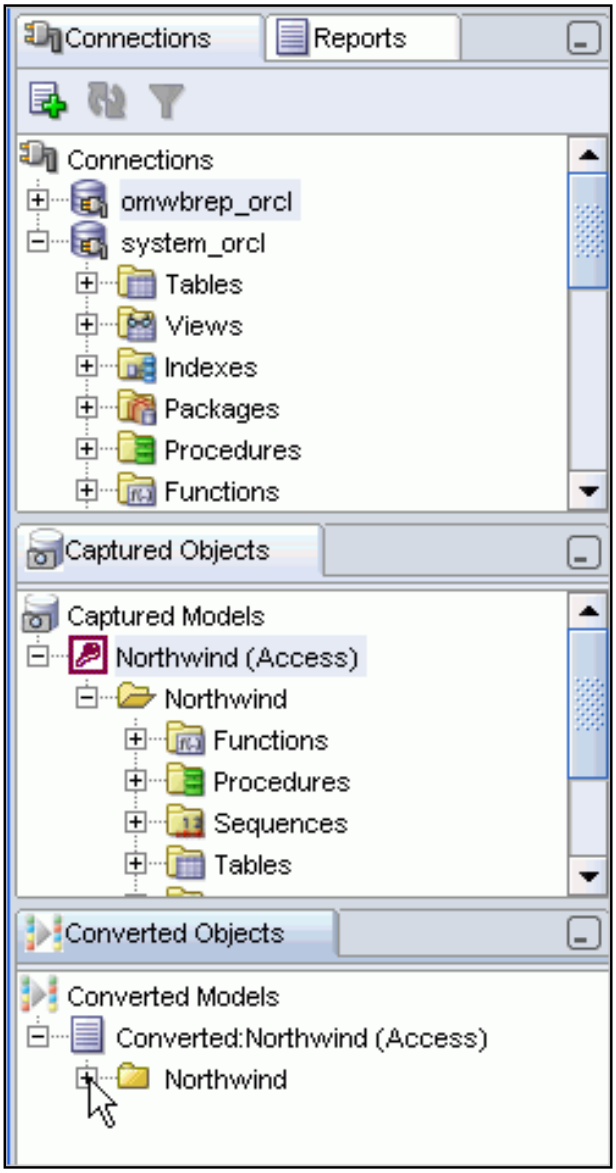
3. The conversion is performed. When done, click **Close**.



4. Expand **Converted Model (Northwind)**.



5. Expand **Northwind**.

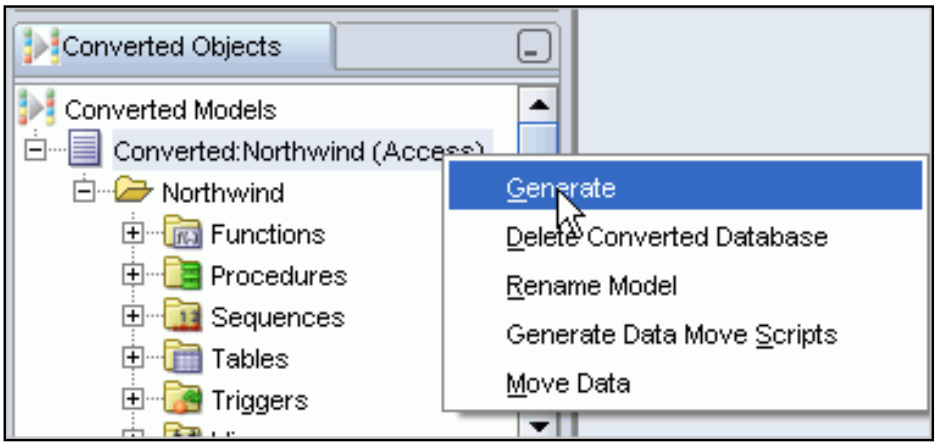


Back to Topic List

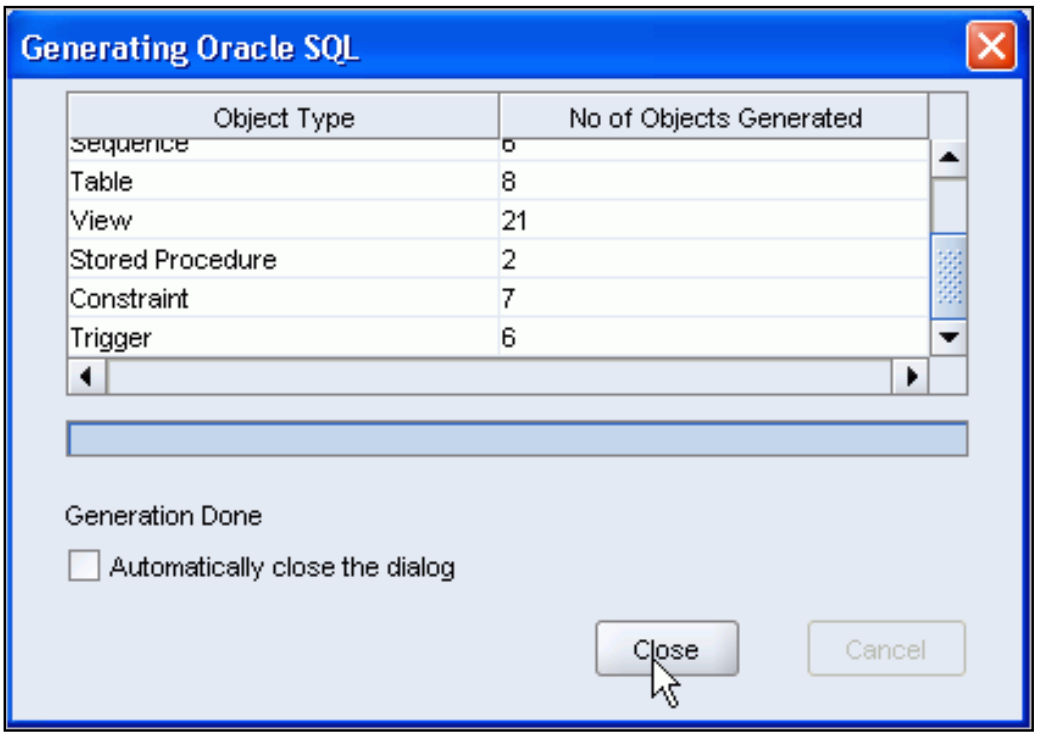
Generating and Executing the Script to Create the Oracle Database Objects

You can now generate the SQL script with DDL statements that can be run to create the objects in an Oracle Database. Perform the following steps:

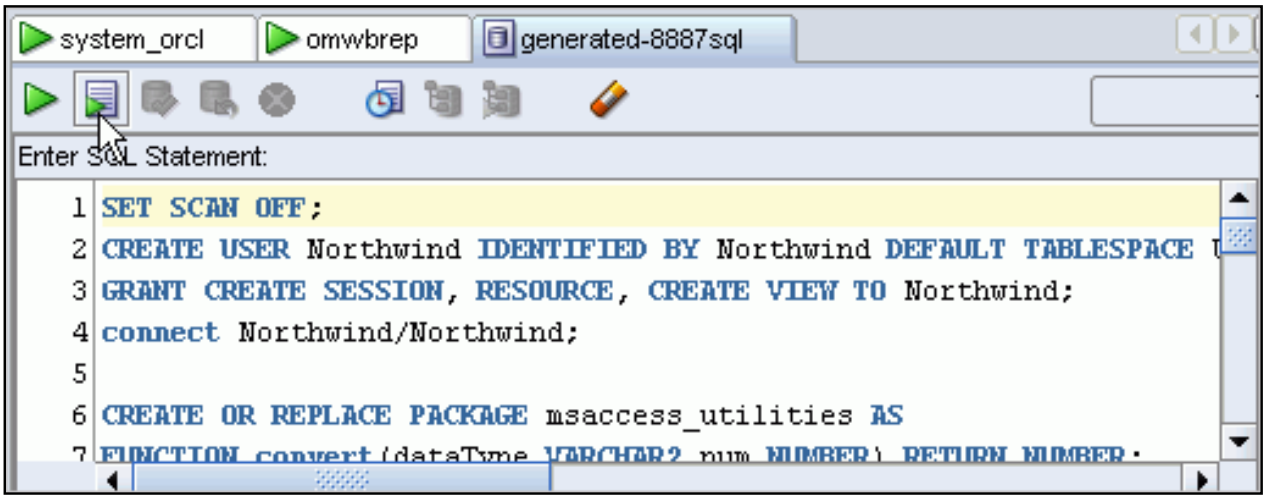
1. Under Converted Objects, right click **Converted Northwind (Access)** and select **Generate**.



2. The Oracle SQL is being generated. When done, click **Close**.

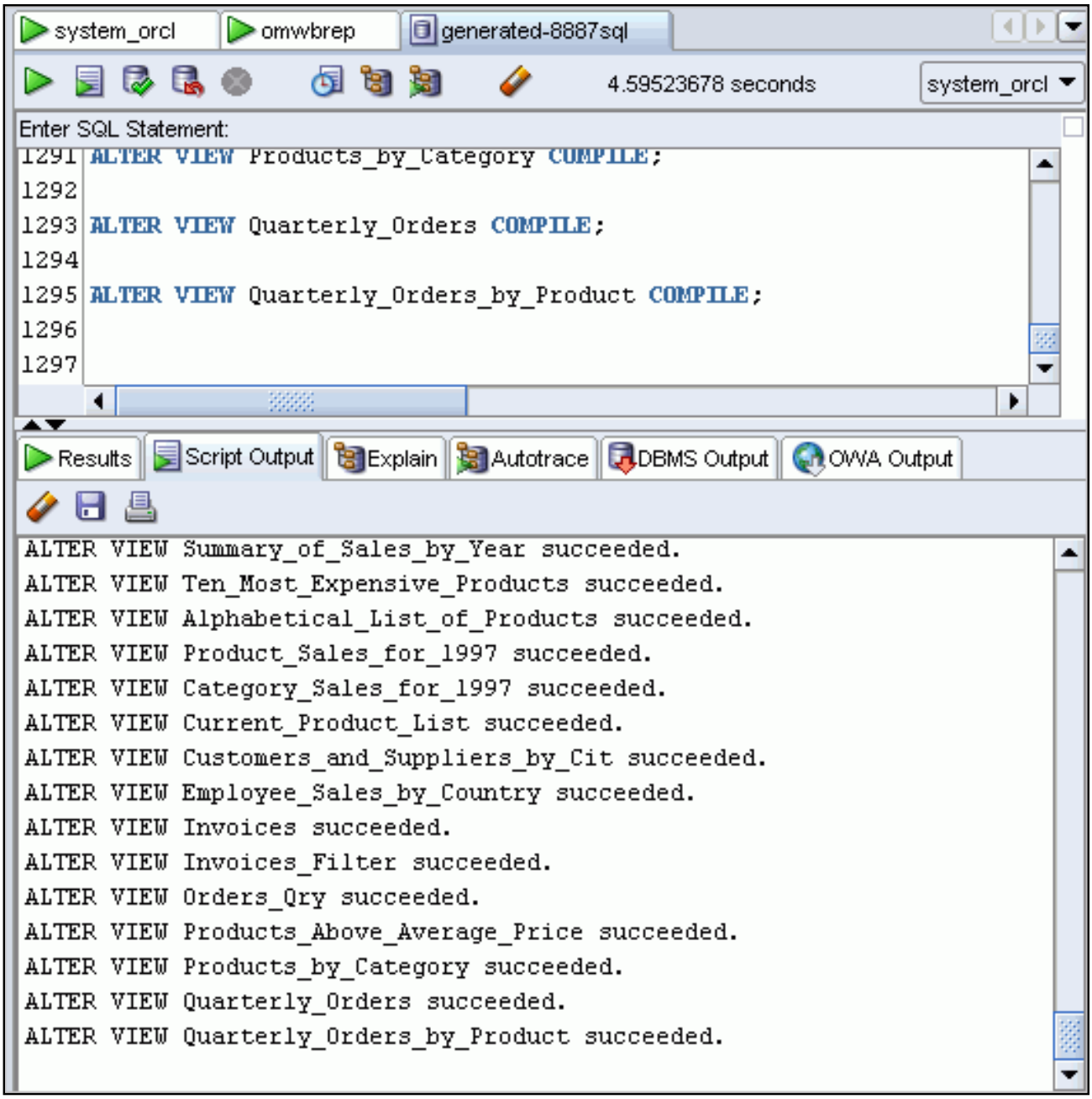


3. The SQL from the script is shown. Click the **Run Script** icon.

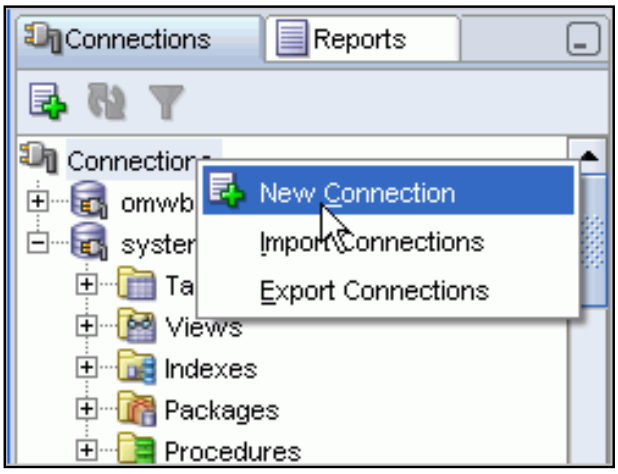


4. Select the **system_orcl** connection and click **OK**.

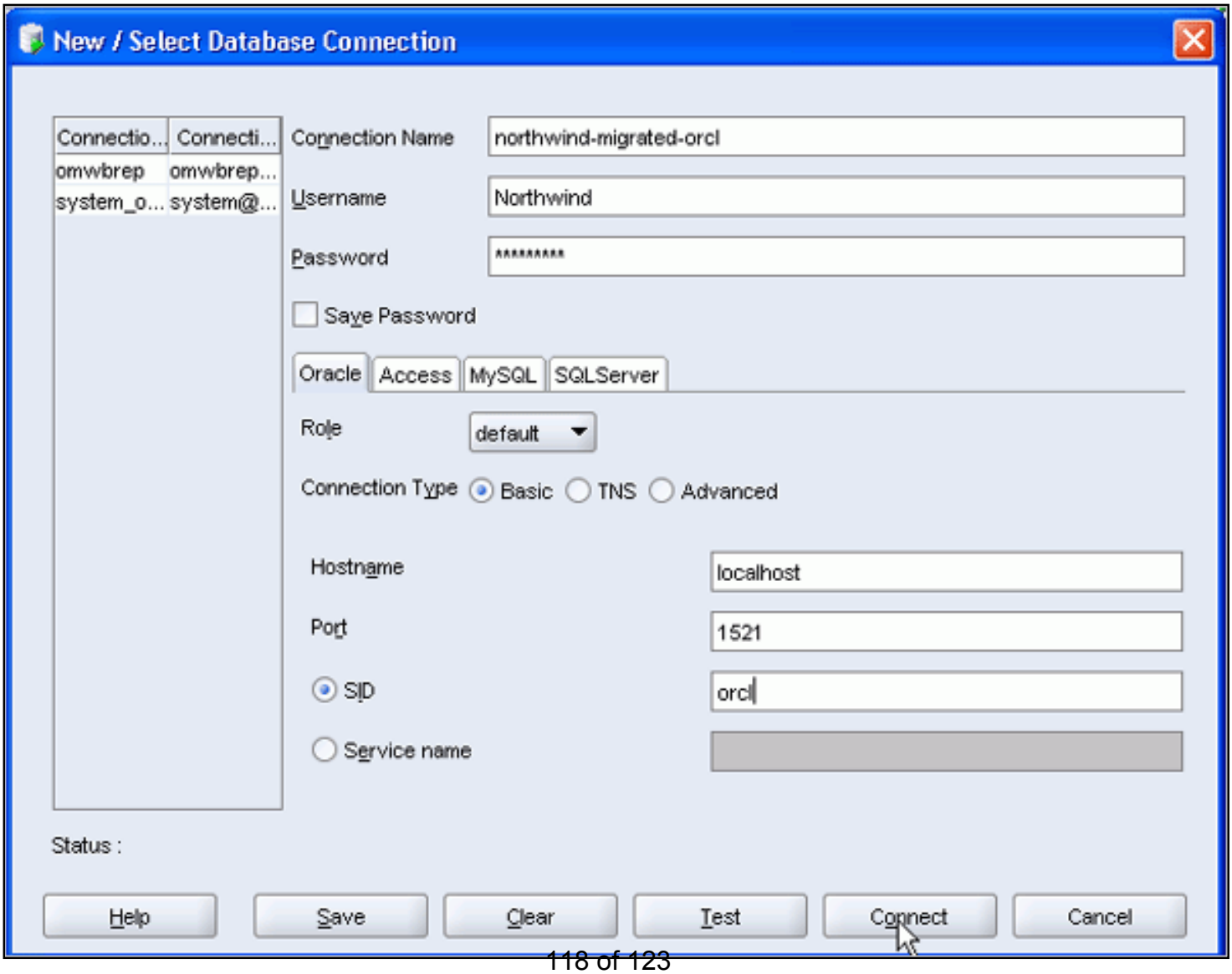
5. The results are display from the script execution.



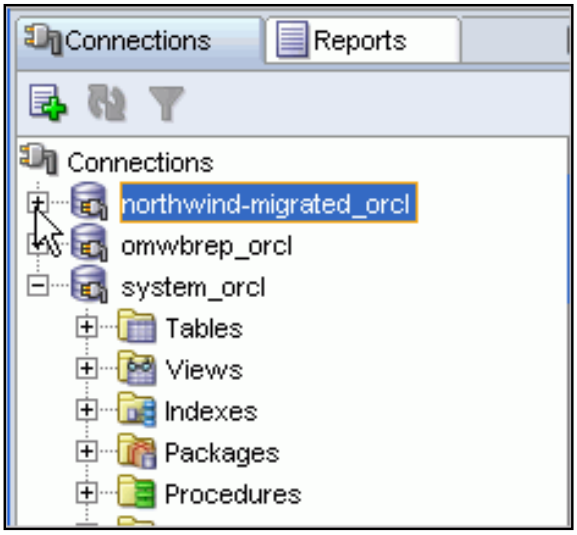
6. Now that your scripts have run successfully, you can create a connection for the Northwind user. Right click **Connection** then select **New Connection**.



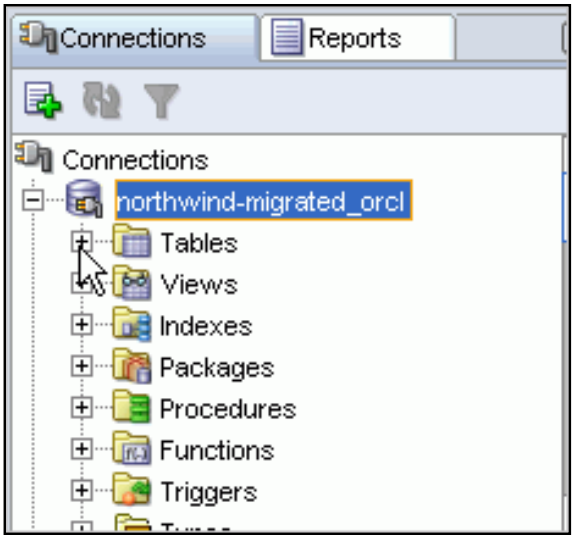
7. Enter **northwind-migrated_orcl** for the Connection Name (or any other name that identifies your connection), **Northwind** for the Username and **northwind** for the Password and enter **orcl** for the SID. Then click **Connect**.



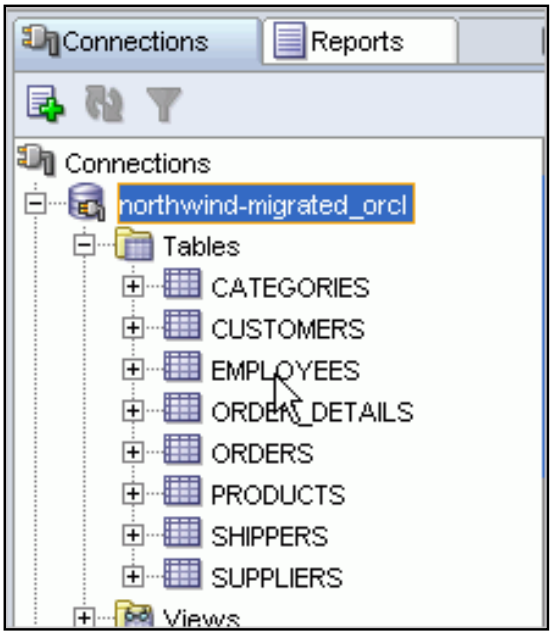
8. Expand **northwind-migrated_orcl.**



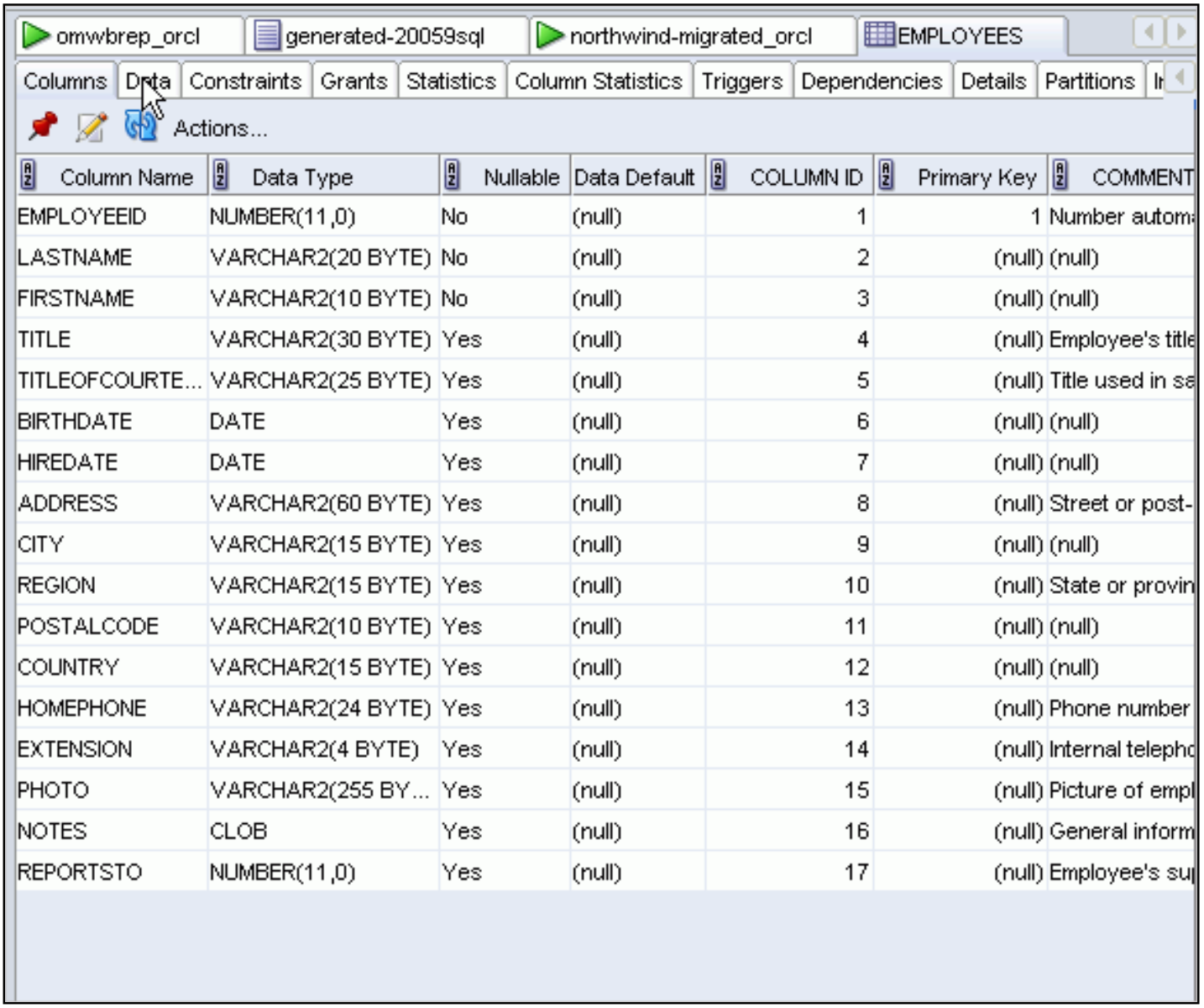
9. Expand **Tables.**



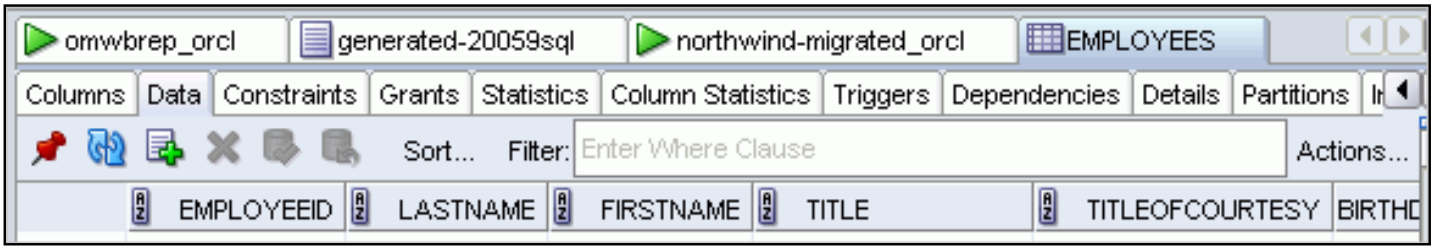
10. Notice that the database tables that were converted to Oracle are listed. Select the **EMPLOYEES** table.



11. Select the **Data** tab.



12. Notice that there currently is no data in the table. You will migrate the data in the next topic of this tutorial.



Back to Topic List

Migrating the Data

The data has already been exported into scripts from Microsoft Access. You will import the data using the scripts

provided. Perform the following steps:

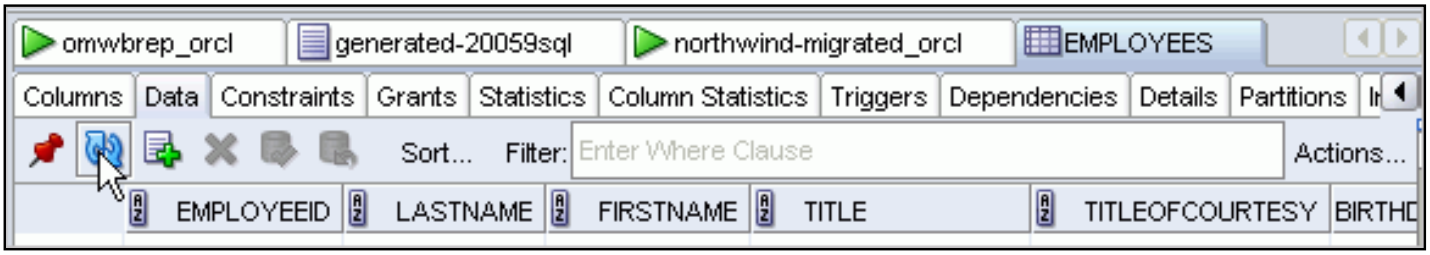
1. Open a DOS command prompt and execute the following commands:

   cd <location where files are>
   oracle_ctl

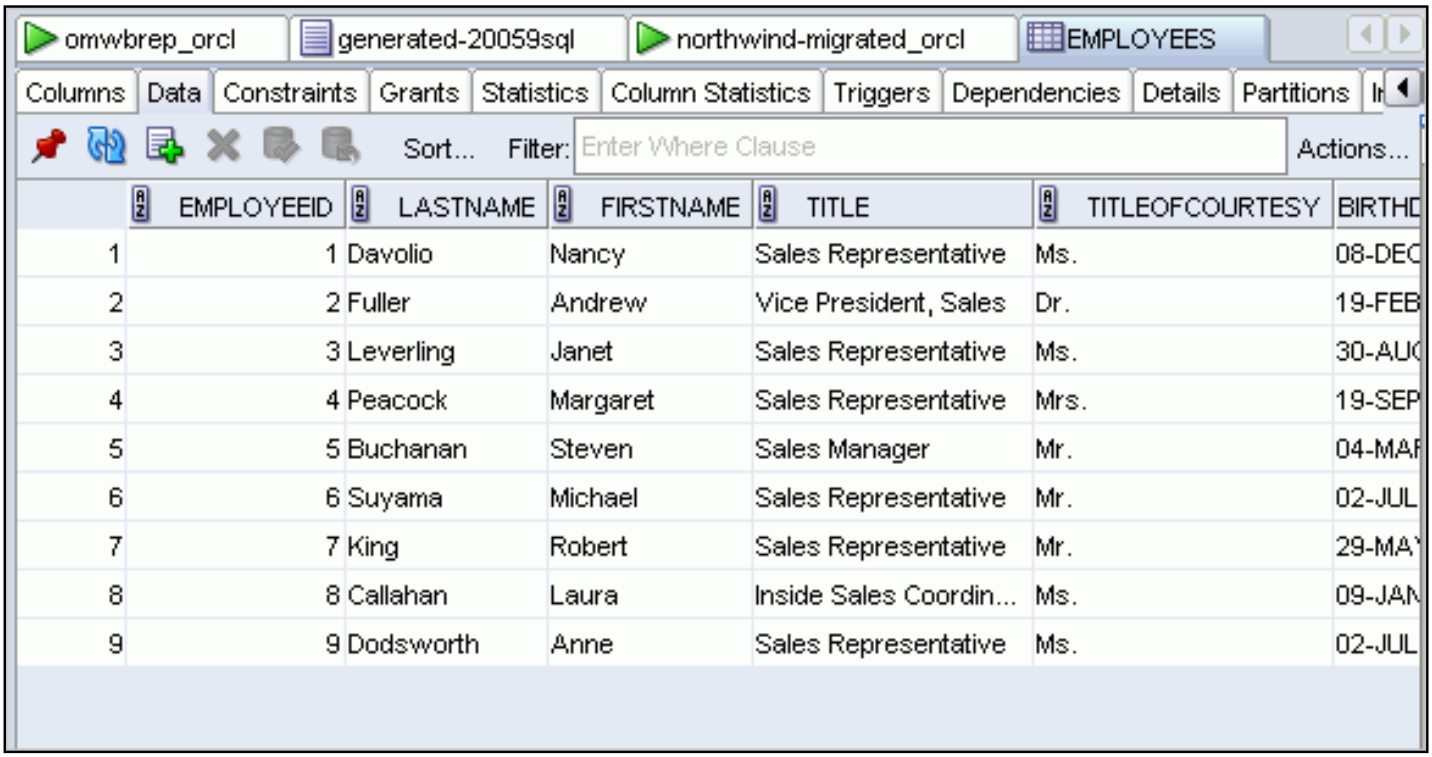
   The location of the files is **`\SQLDev_HOS\Migration\files\data`**

   `oracle_ctl` is a bat file that contains statements to load the data. It uses `sqlldr` to load the data.

2. Switch to Oracle SQL Developer and click **Refresh** to see that the data has been loaded for the `CUSTOMERS` table.



3. The data for the table has been loaded successfully.



Back to Topic List

Summary

In this tutorial, you learned how to:

- Create the OMWBREP User
- Create the Migration Repository
- Capture the Microsoft Access Exported XML
- Convert to the Oracle Model
- Generate and Execute the Script to Create the Oracle Database Objects
- Migrate the Data

Back to Topic List

Move your mouse over this icon to hide all screenshots.