

Testing and Debugging Procedures with SQL Developer

Before You Begin

Purpose

This tutorial covers how to execute a DDL script, review changes to the database objects, create, execute, test and debug a procedure.

Time to Complete

Approximately 60 minutes

Introduction

Oracle SQL Developer is a free and fully supported graphical tool that enhances productivity and simplifies database development tasks. Using SQL Developer, users can browse, edit and create database objects, run SQL statements, edit and debug PL/SQL statements, build PL/SQL unit tests, run reports, and place files under version control.

In this tutorial, you use SQL Developer Release 4.1 to examine various tasks.

Prerequisites

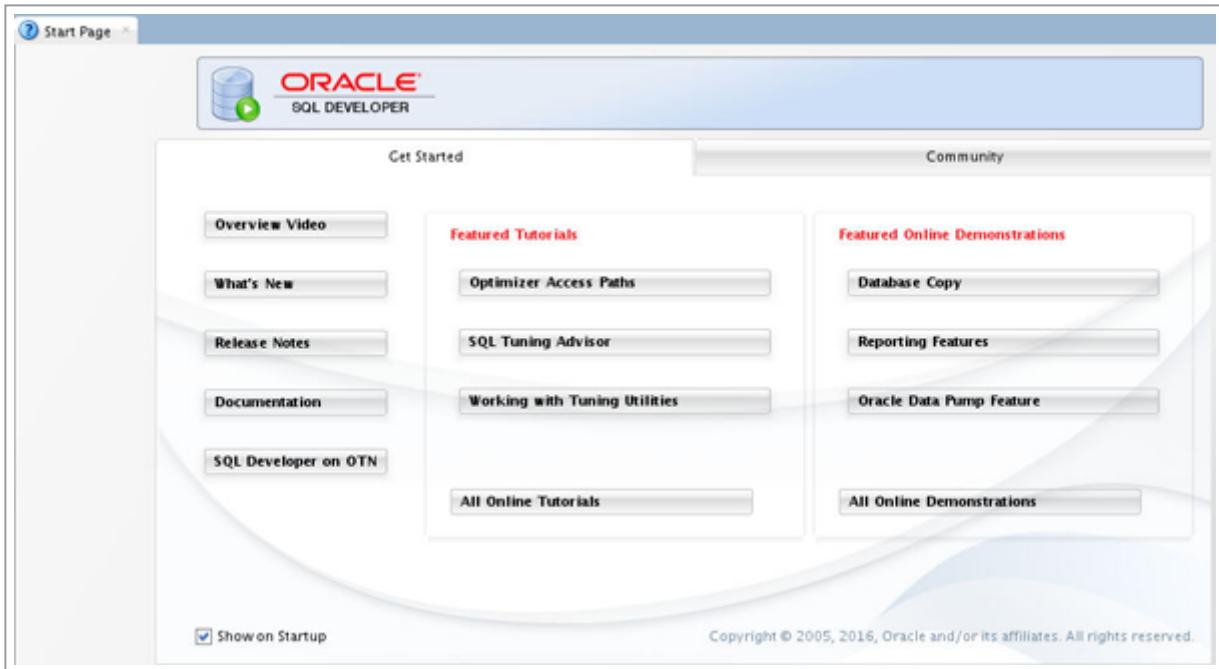
Before starting this tutorial, you should:

- Have installed Oracle SQL Developer Release 4.1 or above
- Have access to an Oracle Database 11g database that has the sample schema installed.
- Grant HR user `DEBUG CONNECT SESSION` and `DEBUG ANY PROCEDURE` privileges.
- Performed the Re-engineering Your Database Using Oracle SQL Developer Data Modeler 4.1 (https://apexapps.oracle.com/pls/apex/f?p=44785:112:0:::P112_CONTENT_ID:19004) tutorial.
- Downloaded and unzipped the `files.zip` (`files/files.zip`) into your working directory.

Creating a Database Connection

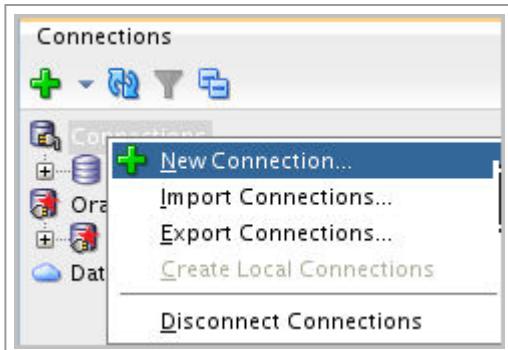
In this topic, you will create a database connection to the HR Schema in SQL Developer:

1. Double click on the SQL Developer icon on the Desktop.
2. The first time SQL Developer is open, the "Start Page" is displayed. You can deselect the **Show on Startup** check box to turn it off.



Description of this image (files/tab_01_01.txt)

3. In the Connections tab, right click Connections and select New Connection



Description of this image (files/tab01_02.txt)

4. Instructions and result (including collapsible image with text file for accessibility):

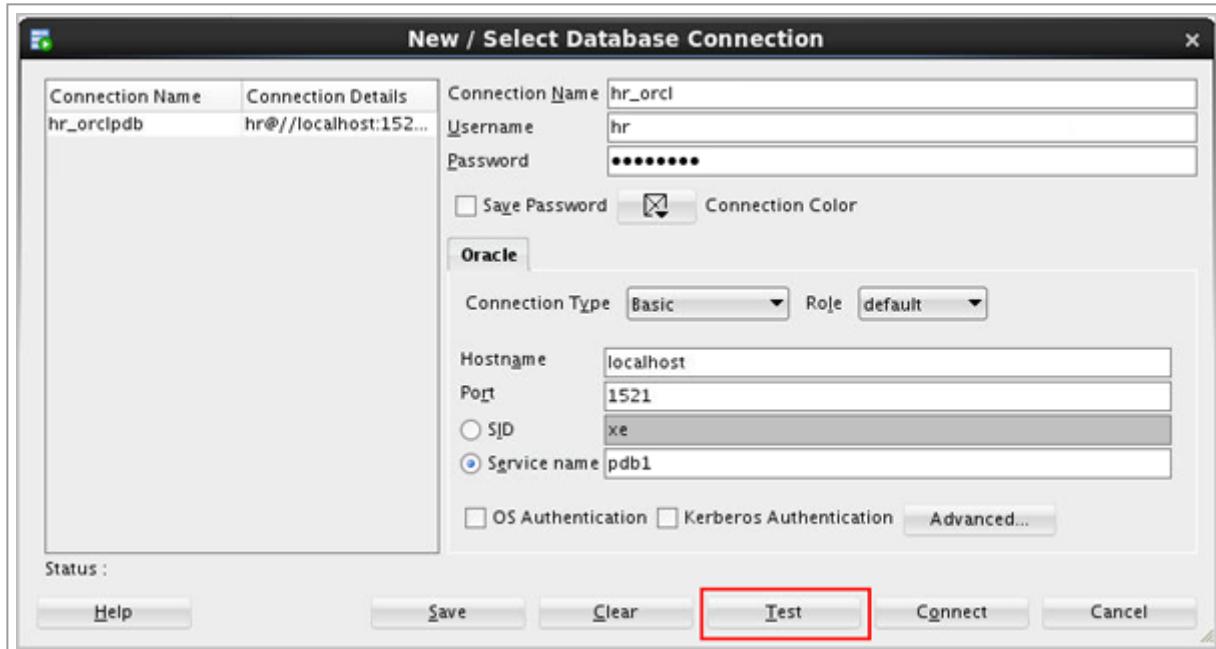
A New / Update Database Connection dialog opens. Enter the connection details as follows and click **Test**.

Connection Name: **hr_orcl**

User Name: **hr**

Password: <your password> (Select the **Save Password** checkbox)

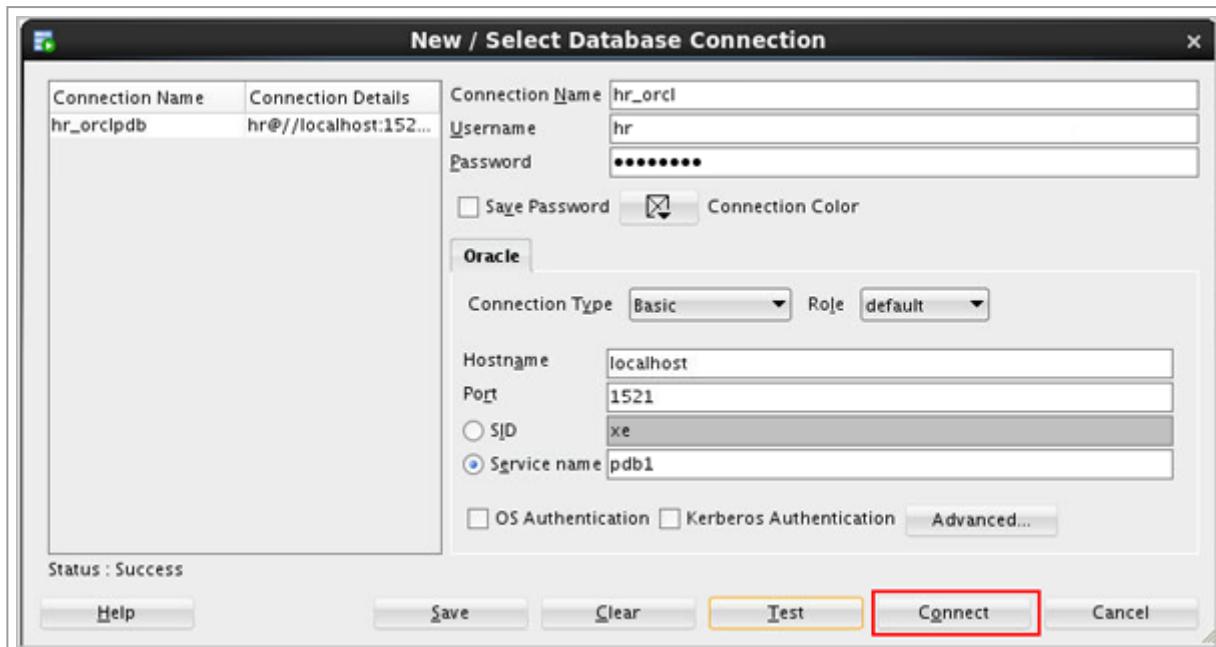
SID: <your own SID>



Description of this image (files/tab01_03.txt)

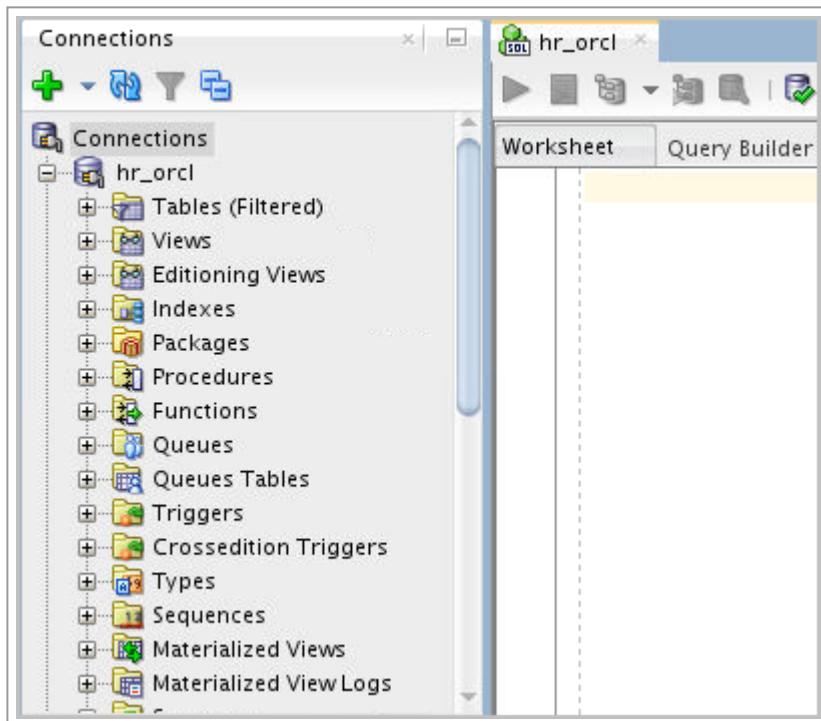
Note: In this tutorial the Service Name is specified instead of SID.

5. The status of your test is 'Success'. Click **Connect** in the New/ Update Database Connection dialog to create the connection.



Description of this image (files/tab01_04.txt)

6. Expand the **hr_orcl** connection. Notice all the object types. Expand **Tables**. In the next section, you examine the objects currently in the hr schema.



Description of this image (files/tabc01_05.txt)

Review Existing Objects in the HR Schema

In this topic, you review the existing objects in the hr schema.

1. Expand the **EMPLOYEES** table. Notice that the column definitions are listed.

COLUMN_NAME	DATA_TYPE	NULLABLE
1 EMPLOYEE_ID	NUMBER(6,0)	No
2 FIRST_NAME	VARCHAR2(20 BYTE)	Yes
3 LAST_NAME	VARCHAR2(25 BYTE)	No
4 EMAIL	VARCHAR2(25 BYTE)	No
5 PHONE_NUMBER	VARCHAR2(20 BYTE)	Yes
6 HIRE_DATE	DATE	No
7 JOB_ID	VARCHAR2(10 BYTE)	No
8 SALARY	NUMBER(8,2)	Yes
9 COMMISSION_PCT	NUMBER(2,2)	Yes
10 MANAGER_ID	NUMBER(6,0)	Yes
11 DEPARTMENT_ID	NUMBER(4,0)	Yes

Description of this image (files/tabc02_01.txt)

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

The screenshot shows the SQL Developer interface. On the left, the Connections navigator displays a connection to 'hr_orcl' with several tables listed under 'Tables (Filtered)': COUNTRIES, DEPARTMENTS, EMPLOYEES, JOB_HISTORY, JOBS, and LOCATIONS. The 'DEPARTMENTS' table is currently selected and highlighted in blue. On the right, the main workspace shows the 'hr_orcl' connection and the 'DEPARTMENTS' table. The 'Columns' tab is selected, displaying the following table structure:

COLUMN_NAME	DATA_TYPE	NULLABLE
1 DEPARTMENT_ID	NUMBER(4,0)	No
2 DEPARTMENT_NAME	VARCHAR2(30 BYTE)	No
3 MANAGER_ID	NUMBER(6,0)	Yes
4 LOCATION_ID	NUMBER(4,0)	Yes

Description of this image (files/tab02_02.txt)

3. Notice that the information in the **EMPLOYEES** tab was replaced by the **DEPARTMENTS** table information. If you want the table information in the tab to remain frozen, select the **Pin** icon to Freeze the pane.

This screenshot is identical to the one above, showing the SQL Developer interface with the 'hr_orcl' connection and the 'DEPARTMENTS' table selected. The 'Columns' tab is active, displaying the same table structure. However, the 'Actions...' button, which contains a pin icon, is highlighted with a red box to draw attention to it.

Description of this image (files/tab02_03.txt)

4. Then click the **EMPLOYEES** table again in the navigator.

The screenshot shows the SQL Developer interface. On the left, the Connections pane displays a tree view of database objects under the connection 'hr_orcl'. The 'Tables (Filtered)' node is expanded, showing 'COUNTRIES', 'DEPARTMENTS', and 'EMPLOYEES'. The 'EMPLOYEES' node is highlighted with a red box. On the right, the main workspace shows the 'hr_orcl' connection selected. Below it, the 'DEPARTMENTS' tab is active, displaying its column definitions in a grid:

COLUMN_NAME	DATA_TYPE	NULLABLE
1 DEPARTMENT_ID	NUMBER(4,0)	No
2 DEPARTMENT_NAME	VARCHAR2(30 BYTE)	No
3 MANAGER_ID	NUMBER(6,0)	Yes
4 LOCATION_ID	NUMBER(4,0)	Yes

Description of this image (files/tab02_04.txt)

- Notice this time you have 2 tabs, one for each of the tables because the DEPARTMENTS table pane is frozen.

The screenshot shows the SQL Developer interface. On the left, the Connections pane displays a tree view of database objects under the connection 'hr_orcl'. The 'Tables (Filtered)' node is expanded, showing 'COUNTRIES', 'DEPARTMENTS', and 'EMPLOYEES'. Both 'DEPARTMENTS' and 'EMPLOYEES' nodes are highlighted with red boxes. On the right, the main workspace shows the 'hr_orcl' connection selected. Below it, the 'DEPARTMENTS' tab is active, displaying its column definitions in a grid, and the 'EMPLOYEES' tab is also visible in the tab bar:

COLUMN_NAME	DATA_TYPE	NULLABLE
1 EMPLOYEE_ID	NUMBER(6,0)	No
2 FIRST_NAME	VARCHAR2(20 BYTE)	Yes
3 LAST_NAME	VARCHAR2(25 BYTE)	No
4 EMAIL	VARCHAR2(25 BYTE)	No
5 PHONE_NUMBER	VARCHAR2(20 BYTE)	Yes
6 HIRE_DATE	DATE	No
7 JOB_ID	VARCHAR2(10 BYTE)	No
8 SALARY	NUMBER(8,2)	Yes
9 COMMISSION_PCT	NUMBER(2,2)	Yes
10 MANAGER_ID	NUMBER(6,0)	Yes
11 DEPARTMENT_ID	NUMBER(4,0)	Yes

Description of this image (files/tab02_05.txt)

- You can see the data in the EMPLOYEES table. Click the Data subtab.

Connections

hr_orcl

Tables (Filtered)

- COUNTRIES
- DEPARTMENTS
- EMPLOYEES**
- JOB_HISTORY

EMPLOYEES

Columns Data Model Constraints Grants Statistics Triggers Actions...

COLUMN_NAME	DATA_TYPE	NULLABLE
1 EMPLOYEE_ID	NUMBER(6,0)	No
2 FIRST_NAME	VARCHAR2(20 BYTE)	Yes
3 LAST_NAME	VARCHAR2(25 BYTE)	No
4 EMAIL	VARCHAR2(25 BYTE)	No
5 PHONE_NUMBER	VARCHAR2(20 BYTE)	Yes
6 HIRE_DATE	DATE	No
7 JOB_ID	VARCHAR2(10 BYTE)	No
8 SALARY	NUMBER(8,2)	Yes
9 COMMISSION_PCT	NUMBER(2,2)	Yes
10 MANAGER_ID	NUMBER(6,0)	Yes
11 DEPARTMENT_ID	NUMBER(4,0)	Yes

Description of this image (files/tab02_06.txt)

7. The data in the EMPLOYEES table is displayed. You can also enter a SQL statement in the SQL Worksheet. Click the hr_orcl tab.

EMPLOYEES

DEPARTMENTS

hr_orcl

Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies Details Partitions Sort... Filter:

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE
1	100 Steven	King	SKING	515.123.4567	17-JUN-03
2	101 Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-05
3	102 Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-01
4	103 Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-06
5	104 Bruce	Ernst	BERNST	590.423.4568	21-MAY-07
6	105 David	Austin	DAUSTIN	590.423.4569	25-JUN-05
7	106 Valli	Pataballa	VPATABAL	590.423.4560	05-FEB-06
8	107 Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-07

Description of this image (files/tab02_07.txt)

8. Enter the following SQL statement and select the Execute SQL Statement icon.

```
select * from employees
where job_id like '%SA%';
```

EMPLOYEES

DEPARTMENTS

hr_orcl

Worksheet Query Builder

```
select * from employees
where job_id like '%SA%';
```

Description of this image (files/tab02_08.txt)

9. The Query Results are displayed. In the next topic, you run the script you generated in the previous tutorial on Data Modeler.

The screenshot shows the Oracle SQL Developer interface. The top navigation bar has tabs for EMPLOYEES, DEPARTMENTS, and hr_orcl. The hr_orcl tab is active, showing a connection status of 0.122 seconds. Below the tabs is a toolbar with various icons. The main area has two tabs: Worksheet and Query Builder; the Worksheet tab is selected. A SQL query is entered in the worksheet:

```
select * from employees  
where job_id like '%SA%';
```

Below the worksheet is a Script Output window showing the results of the query execution:

Task completed in 0.122 seconds

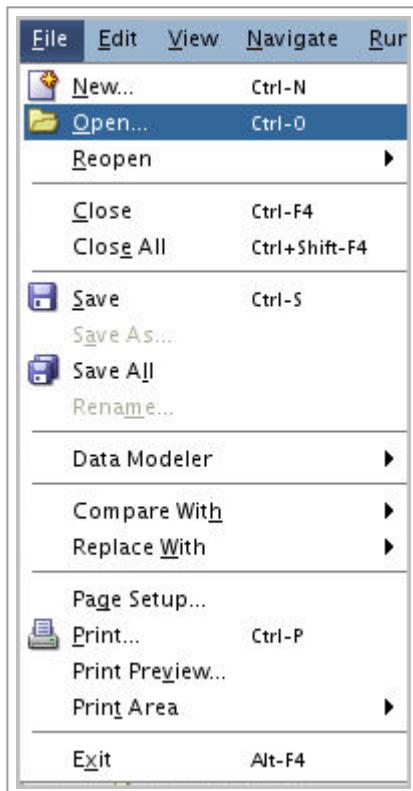
EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL
145	John	Russell	JRUSSEL
146	Karen	Partners	KPARTNER
147	Alberto	Errazuriz	AERRAZUR
148	Gerald	Cambrault	GCAMBRAU
149	Eleni	Zlotkey	EZL0TKEY
150	Peter	Tucker	PTUCKER
151	David	Bernstein	DBERNSTE
152	Peter	Hall	PHALL
153	Christopher	Olsen	COLSEN
154	Nanette	Cambrault	NCAMBRAU
155	Oliver	Tuvault	OTUVault

Description of this image (files/tab02_09.txt)

Executing a DDL Script

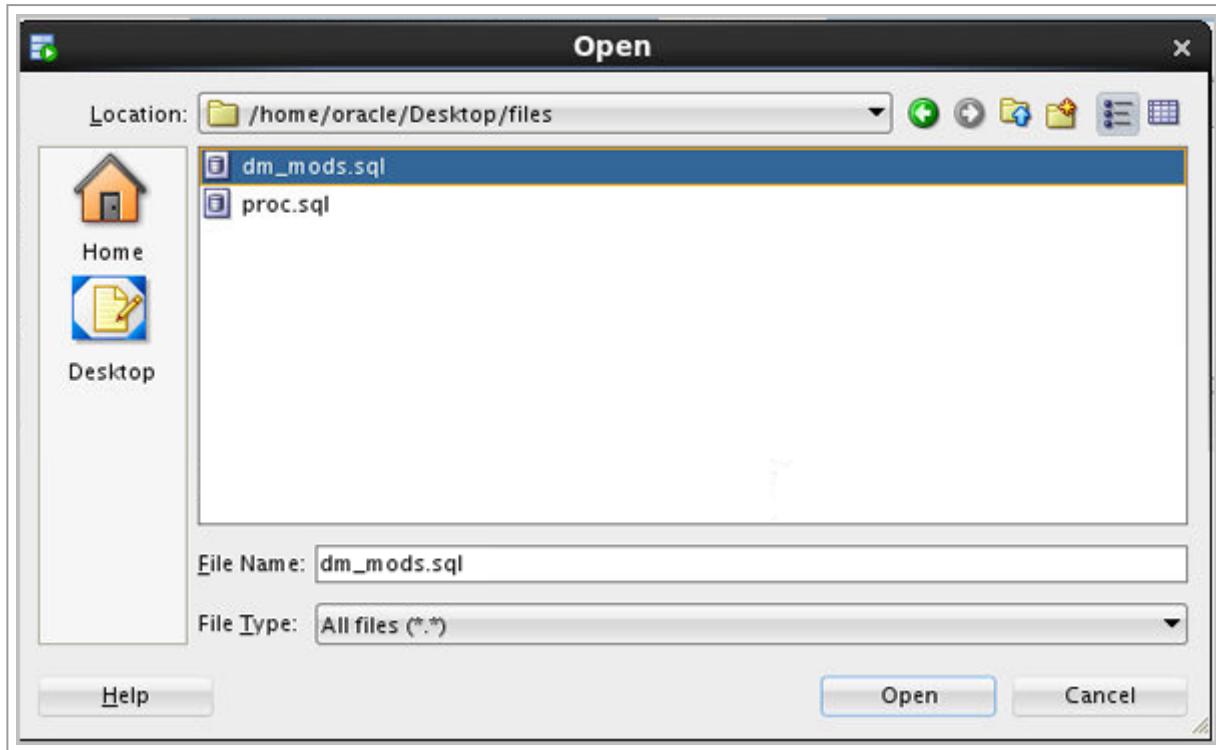
In this topic, you execute the DDL script you generated in the Data Modeler tutorial. If you did not complete the previous tutorial, you can access the solution using the **dm_mods.sql** in the files folder.

1. Select **File > Open**.



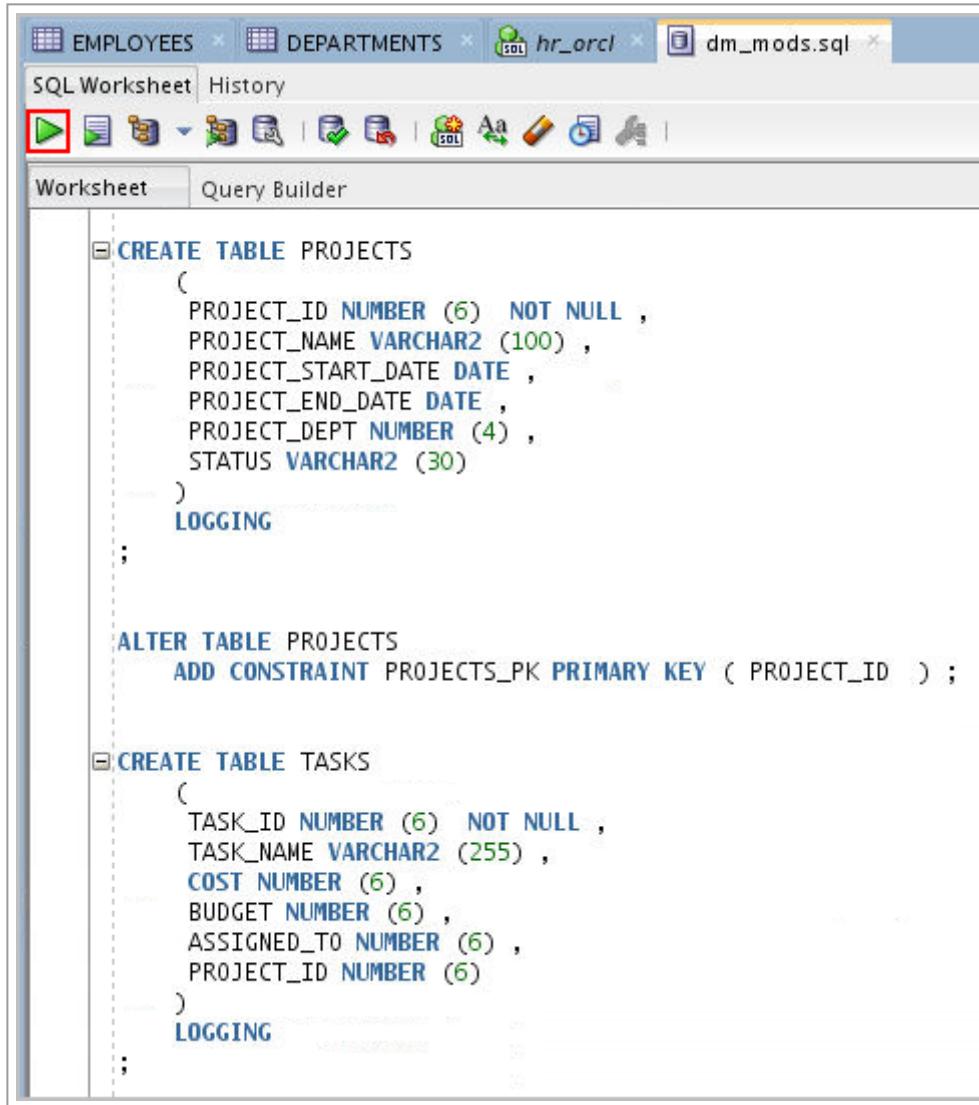
Description of this image (files/tab03_01.txt)

2. Locate the `dm_mods.sql` file and click **Open.**



Description of this image (files/tab03_02.txt)

3. This SQL file contains all the DDL to change the HR Schema so that it is synchronized with the model changes you made in the previous tutorial. When you execute this script, the `PROJECTS` and `TASKS` tables will be created, and the new `COST_CENTER` column will be added to the `DEPARTMENTS` table. Scroll down to review the DDL.



The screenshot shows the SQL Worksheet tab selected in the top navigation bar. The main area displays a DDL script for creating two tables, PROJECTS and TASKS, and adding a primary key constraint. The script includes logging clauses and semicolons at the end of each section.

```

CREATE TABLE PROJECTS
(
    PROJECT_ID NUMBER (6) NOT NULL ,
    PROJECT_NAME VARCHAR2 (100) ,
    PROJECT_START_DATE DATE ,
    PROJECT_END_DATE DATE ,
    PROJECT_DEPT NUMBER (4) ,
    STATUS VARCHAR2 (30)
)
LOGGING
;

ALTER TABLE PROJECTS
ADD CONSTRAINT PROJECTS_PK PRIMARY KEY ( PROJECT_ID ) ;

CREATE TABLE TASKS
(
    TASK_ID NUMBER (6) NOT NULL ,
    TASK_NAME VARCHAR2 (255) ,
    COST NUMBER (6) ,
    BUDGET NUMBER (6) ,
    ASSIGNED_TO NUMBER (6) ,
    PROJECT_ID NUMBER (6)
)
LOGGING
;

```

Description of this image (files/tab03_03.txt)

4. Select the **hr_orcl** connection from the list and click **OK**.



Description of this image (files/tab03_04.txt)

5. All the statements in the DDL script executed successfully

```

CREATE TABLE PROJECTS
(
    PROJECT_ID NUMBER(6) NOT NULL ,
    PROJECT_NAME VARCHAR2(100) ,
    PROJECT_START_DATE DATE ,
    PROJECT_END_DATE DATE ,
    PROJECT_DEPT NUMBER(4) ,
    STATUS VARCHAR2(30)
)
LOGGING
;

ALTER TABLE PROJECTS
ADD CONSTRAINT PROJECTS_PK PRIMARY KEY (PROJECT_ID) ;

```

Script Output

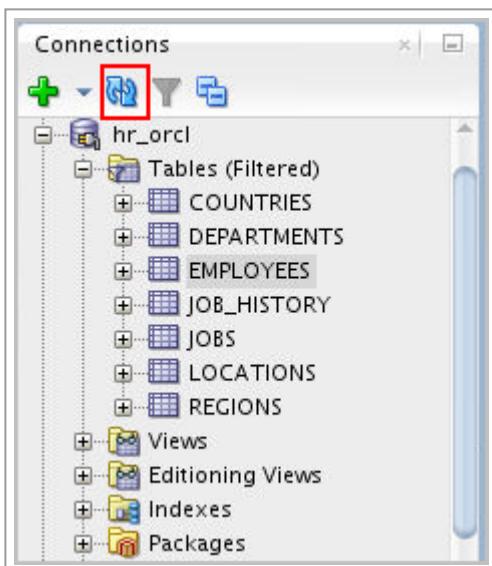
Table PROJECTS altered.

Table TASKS altered.

Table TASKS altered.

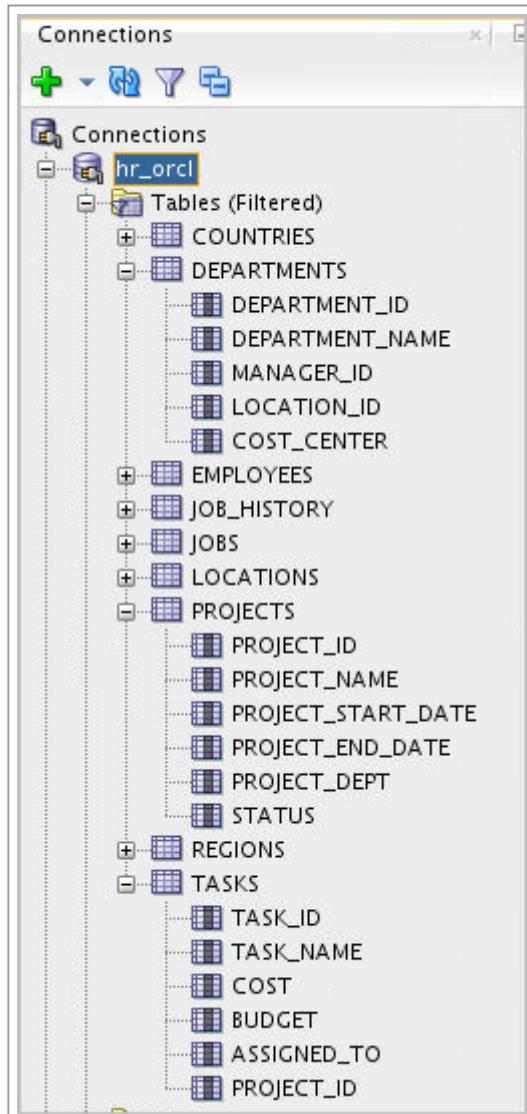
Description of this image (files/tab03_05.txt)

6. Click the Refresh icon to refresh the list of tables.



Description of this image (files/tab03_06.txt)

7. Notice that the new tables **PROJECTS** and **TASKS** are contained in the list. Expand the **DEPARTMENTS**, **PROJECTS** and **TASKS** table nodes and review the results. In the next topic, you create a procedure and run it.



Description of this image (files/tab03_07.txt)

Creating and Executing a Procedure

Debugging a Procedure

The procedure created in the earlier section was created with an error. You can locate errors in the code by debugging the code. You have to run a script before you actually start the debug process

Before you Debug

In order to debug a sub program you should have **DEBUG** privileges.

1. To check the privileges you can execute a SQL statement

The screenshot shows the Oracle SQL Developer interface. In the top tab bar, the 'hr_orcl' connection is selected. Below the tabs, there are two tabs: 'Worksheet' and 'Query Builder', with 'Worksheet' currently active. The main workspace contains the following SQL code:

```
select * from user_sys_privs;
```

Below the workspace is a 'Query Result' window. It has a toolbar with icons for Run, Stop, Refresh, and Save, followed by a dropdown menu set to 'SQL'. The status bar at the bottom of the window indicates 'All Rows Fetched: 7 in 0.033 seconds'. The results are displayed in a table:

USERNAME	PRIVILEGE	ADMIN_OPTION	COMMON
1 HR	CREATE VIEW	NO	NO
2 HR	UNLIMITED TABLESPACE	NO	NO
3 HR	CREATE DATABASE LINK	NO	NO
4 HR	CREATE SEQUENCE	NO	NO
5 HR	CREATE SESSION	NO	NO
6 HR	ALTER SESSION	NO	NO
7 HR	CREATE SYNONYM	NO	NO

Description of this image (files/tab05_01_01.txt)

You can see that the hr user doesn't have **DEBUG CONNECT** and **DEBUG** privileges

2. If you don't have the required **DEBUG** privileges, a **SYSDBA** role user has to assign them. Login as a **SYSDBA** user

The screenshot shows the 'New / Select Database Connection' dialog box. On the left, a list of existing connections is shown: 'hr_orcl' and 'hr_orclpdb'. On the right, connection details are being entered for a new connection:

- Connection Name:** hr_orcl
- Username:** sys
- Password:** (redacted)
- Role:** **SYSDBA** (highlighted with a red box)
- Hostname:** localhost
- Port:** 1521
- Service name:** pdb1

At the bottom of the dialog, there are buttons for **Help**, **Save**, **Clear**, **Test**, **Connect** (highlighted with a red box), and **Cancel**.

Description of this image (files/tab05_01_02.txt)

3. Execute the grant commands and ACL(Access Control List) script shown as a **SYSDBA** user

The screenshot shows the Oracle SQL Developer interface. In the top tab bar, there are tabs for 'dm_mods.sql', 'proc.sql', 'hr_orcl', and 'hr_orcl~3'. The 'hr_orcl~3' tab is active. The main area is a 'Worksheet' tab, which contains the following SQL code:

```
GRANT DEBUG ANY PROCEDURE TO hr;
GRANT DEBUG CONNECT SESSION TO hr;
```

Below the worksheet, a 'Script Output' window is visible, showing the message 'Grant succeeded.' twice.

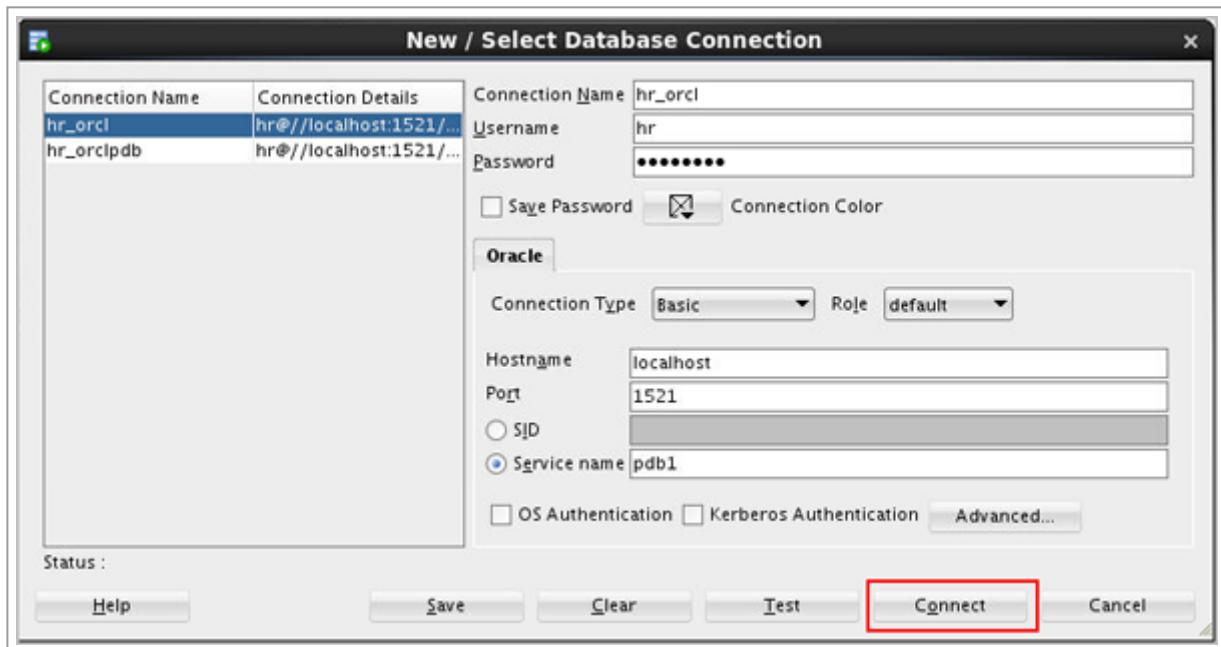
Description of this image (files/tab05_01_03-2.txt)

The screenshot shows the Oracle SQL Developer interface. The top tab bar has tabs for 'dm_mods.sql', 'proc.sql', and 'hr_orcl~3'. The 'hr_orcl~3' tab is active. The main area is a 'Worksheet' tab, displaying a PL/SQL block:

```
BEGIN
  DBMS_NETWORK_ACL_ADMIN.append_host_ace(
    host => '127.0.0.1',
    lower_port => null,
    upper_port => null,
    ace => xs$ace_type(privilege_list => xs$name_list('jdwp'),
    principal_name => 'hr',
    principal_type => xs_acl.ptype_db)
  );
END;
```

Description of this image (files/tab05_01_03.txt)

4. Now login as hr user, who has a non-sysdba user role



Description of this image (files/tab05_01_04.txt)

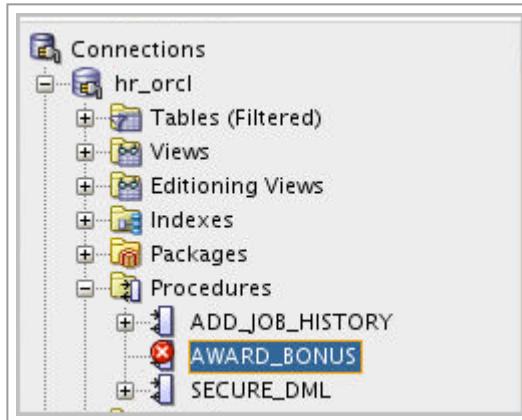
5. Execute the SQL statement shown to check whether required privileges are granted to hr user

USERNAME	PRIVILEGE	ADMIN_OPTION	COMMON
1 HR	CREATE VIEW	NO	NO
2 HR	UNLIMITED TABLESPACE	NO	NO
3 HR	DEBUG CONNECT SESSION	NO	NO
4 HR	CREATE DATABASE LINK	NO	NO
5 HR	CREATE SEQUENCE	NO	NO
6 HR	CREATE SESSION	NO	NO
7 HR	DEBUG ANY PROCEDURE	NO	NO
8 HR	ALTER SESSION	NO	NO
9 HR	CREATE SYNONYM	NO	NO

Description of this image (files/tab05_01_05.txt)

Debugging

1. Now open the AWARD_BONUS procedure you created earlier. Compile the procedure



Description of this image (files/tab05_02_01.txt)

You can see the error message in the compiler log. It specifies a line number where the error occurred

2. Modify the code in line 13 by adding a semi colon. Select the **Compile** icon.

The screenshot shows the PL/SQL editor for the 'hr_orcl' connection. The current tab is 'AWARD_BONUS'. The code is as follows:

```

1  create or replace PROCEDURE award_bonus (
2      emp_id NUMBER, sales_amt NUMBER) AS
3      l_salary    REAL;
4      l_commission    REAL;
5      comm_missing  EXCEPTION;
6  BEGIN
7      SELECT salary, commission_pct INTO l_salary, l_commission
8          FROM employees
9          WHERE employee_id = emp_id;
10     dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
11     dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
12     IF l_commission IS NULL THEN
13         RAISE comm_missing;
14     ELSE
15         l_salary := l_salary + sales_amt*l_commission;
16         dbms_output.put_line('Salary for Employee ID '||emp_id||' will be changed');
17         UPDATE employees
18             SET salary = l_salary
19             WHERE employee_id = emp_id;

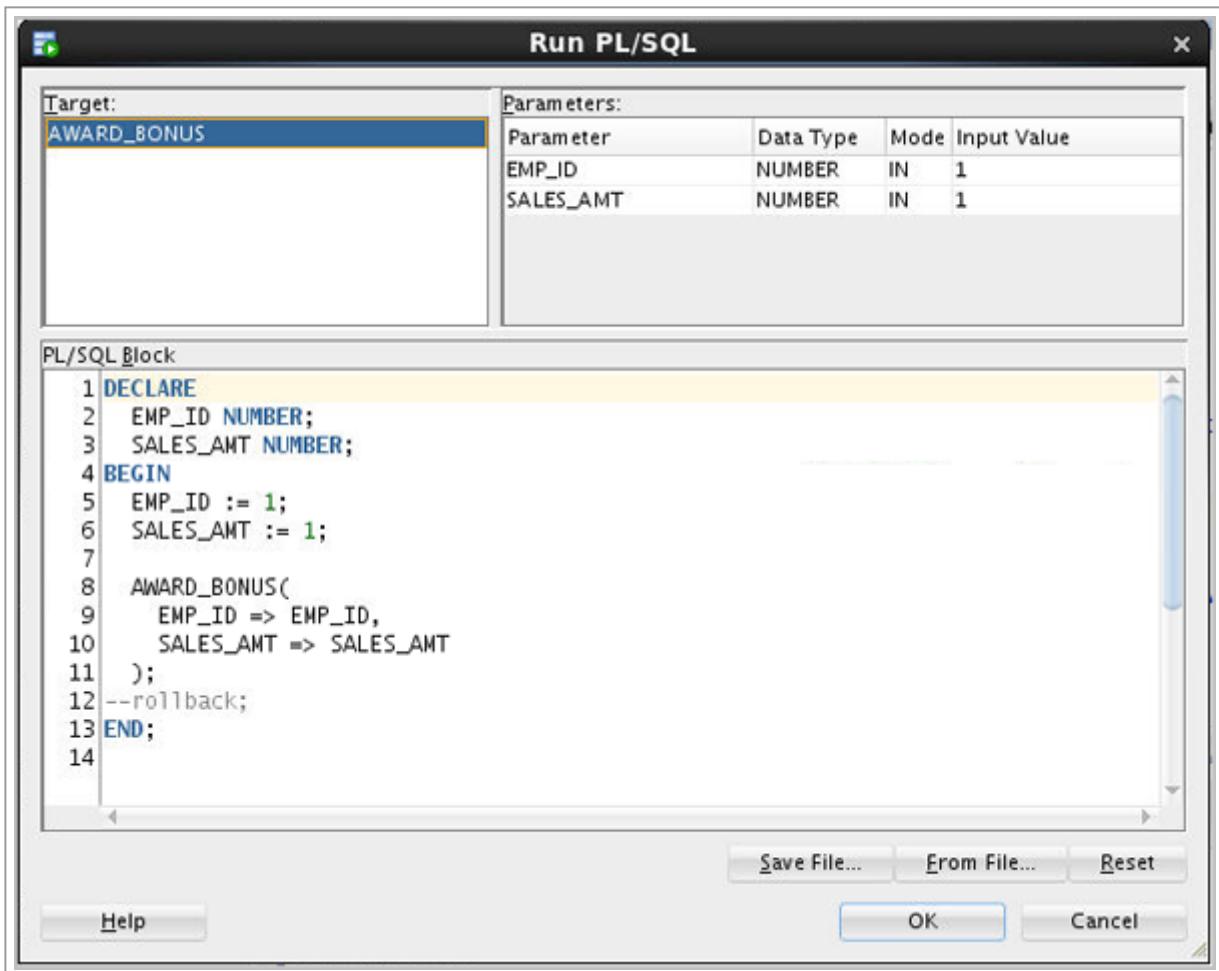
```

The 'Run' icon (a green triangle) on the toolbar is highlighted with a red box. The status bar at the bottom right shows '13:24'.

Description of this image (files/tab05_02_05.txt)

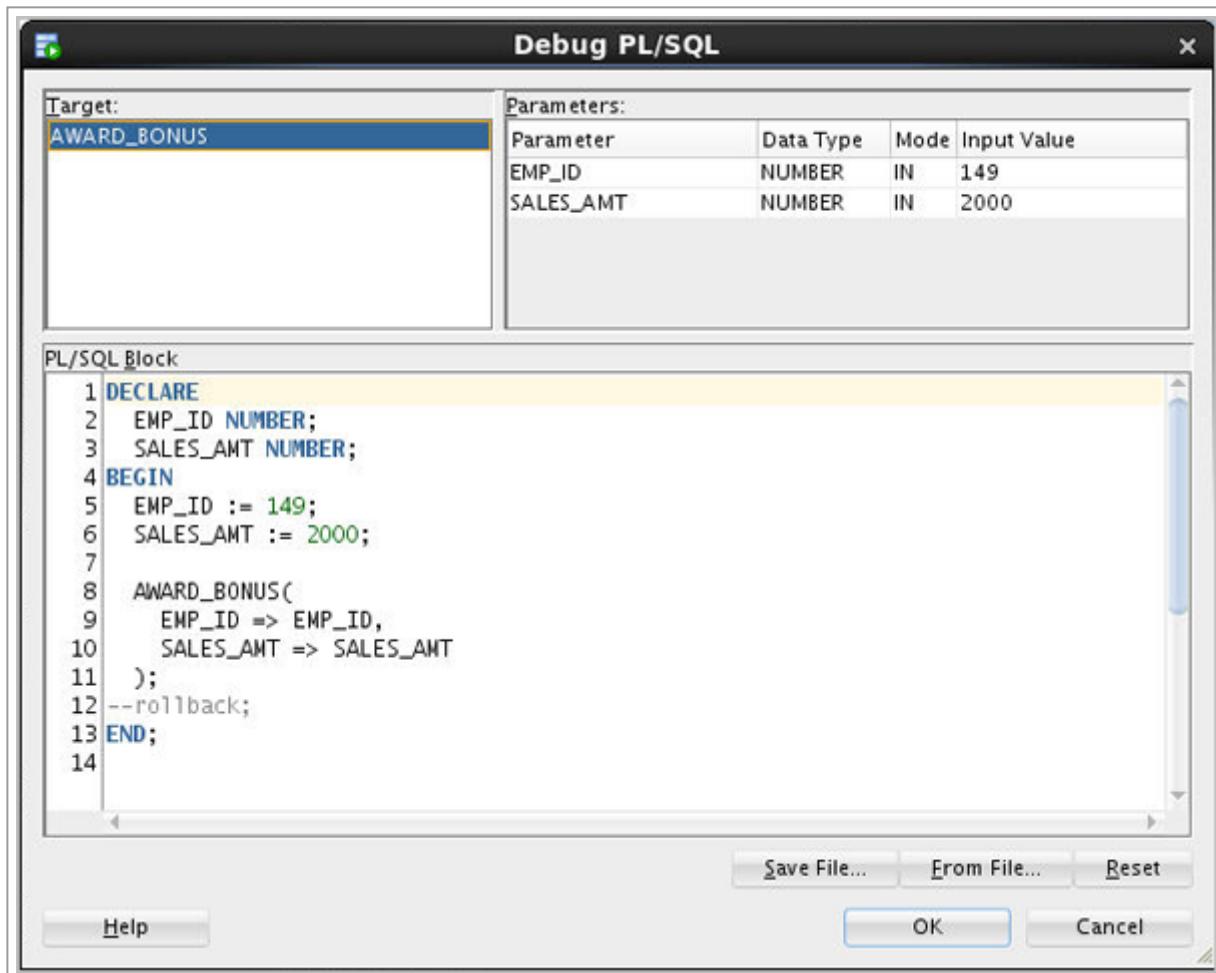
Run the procedure by clicking on Run icon.

3. The Run PL/SQL dialog window appears. Notice that the values for EMP_ID and SALES_AMT are currently set to 1.



Description of this image (files/tab05_02_07.txt)

4. Change the default values to **149** for EMP_ID and **2000** for SALES_AMT and click **OK**.



Description of this image (files/tab05_02_10.txt)

5. Note that the procedure executed successfully and the value for salary was changed. To see how debug works, you create a break point. Click the line number 7.

The screenshot shows the Oracle SQL Developer interface. The top window displays the PL/SQL code for the `AWARD_BONUS` procedure. The code includes logic to check if a commission is null and raise an error if so, or update the salary if it's not. It also prints the current salary and commission percentage for employee ID 149. The bottom window, titled "Debugging: IdeConnections%23hr_orcl.jpr - Log", shows the output of the executed code, including the database connection details, the execution of the `ALTER SESSION SET PLSQL_DEBUG=TRUE` command, and the printed values for employee 149.

```

8   FROM employees
9     WHERE employee_id = emp_id;
10    dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
11    dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
12  IF l_commission IS NULL THEN
13    RAISE comm_missing;
14  ELSE
15    l_salary := l_salary + sales_amt*l_commission;
16    dbms_output.put_line('Salary for Employee ID '||emp_id||' will be changed to:');
17    UPDATE employees
18      SET salary = l_salary
19        WHERE employee_id = emp_id;
20
21 END IF;
END award_bonus;

```

Debugging: IdeConnections%23hr_orcl.jpr - Log

Connecting to the database hr_orcl.
 Executing PL/SQL: ALTER SESSION SET PLSQL_DEBUG=TRUE
 Executing PL/SQL: CALL DBMS_DEBUG_JDWP.CONNECT_TCP('127.0.0.1', '49685')
 Debugger accepted connection from database on port 49685.
 Source breakpoint: AWARD_BONUS.pls:7
 Executing PL/SQL: CALL DBMS_DEBUG_JDWP.DISCONNECT()
 Salary for Employee ID 149 currently is: 10400
 Commission Percentage for 149 currently is: .2
 Salary for Employee ID 149 will be changed to: 10800
 Process exited.
 Disconnecting from the database hr_orcl.
 Debugger disconnected from database.

Description of this image (files/tab05_02_17.txt)

- When a break point is created at line 7, the execution will break at line 7 and allows developer to monitor the data held in different variables. Click the **Debug** icon.

```

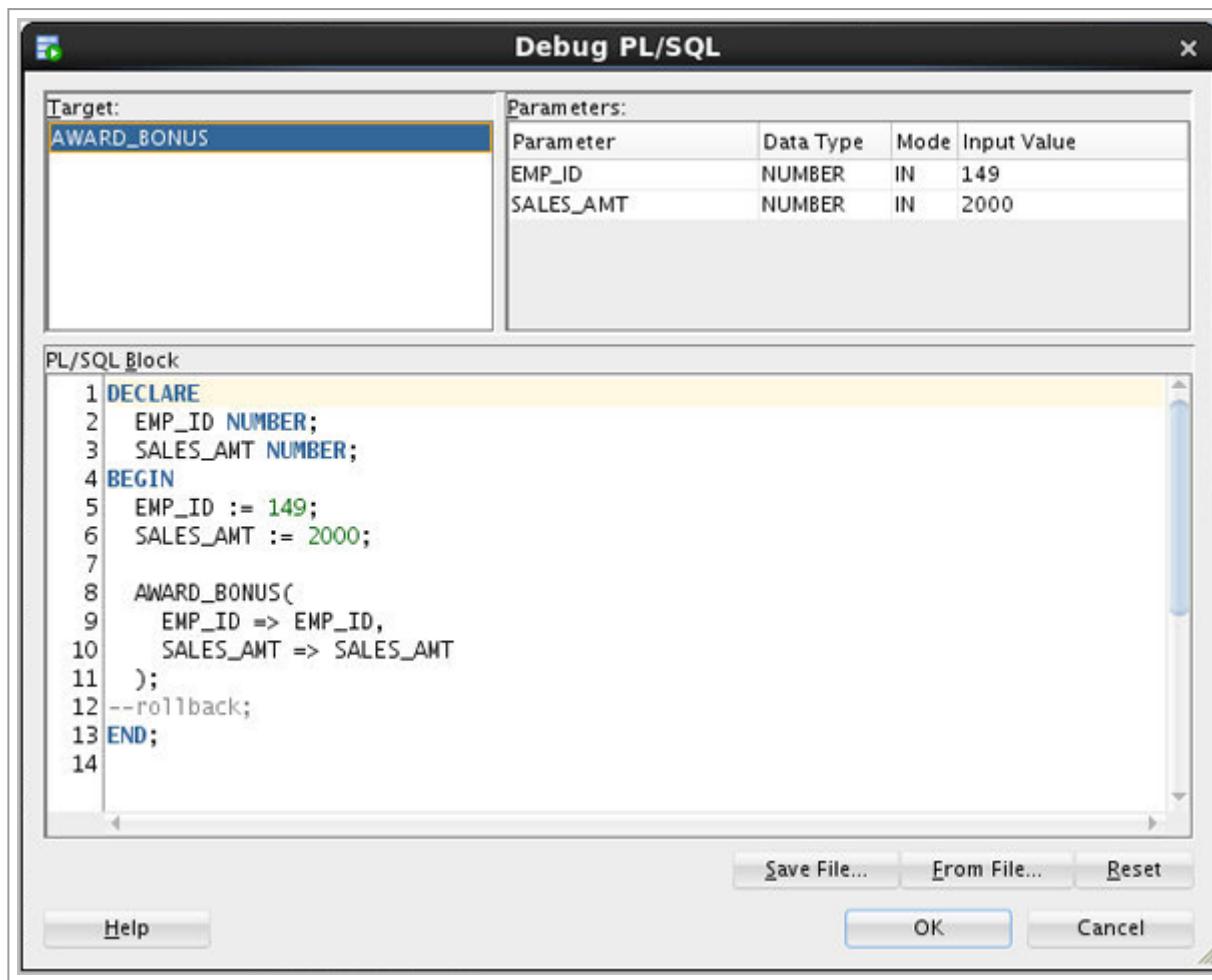
1  CREATE OR REPLACE PROCEDURE award_bonus (
2      emp_id NUMBER, sales_amt NUMBER) AS
3      l_salary    REAL;
4      l_commission    REAL;
5      comm_missing EXCEPTION;
6
7      BEGIN
8          SELECT salary, commission_pct INTO l_salary, l_commission
9              FROM employees
10             WHERE employee_id = emp_id;
11
12         dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
13         dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
14
15         IF l_commission IS NULL THEN
16             RAISE comm_missing;
17         ELSE
18             l_salary := l_salary + sales_amt*l_commission;
19             dbms_output.put_line('Salary for Employee ID '||emp_id||' will be changed');
20             UPDATE employees
21                 SET salary = l_salary
22                   WHERE employee_id = emp_id;

```

13:24

Description of this image (files/tab05_02_09.txt)

- Click **OK** to accept the same input values as before.



Description of this image (files/tab05_02_10.txt)

- The debugger is running and has stopped at line 8. Click the **Smart Data** tab. The Smart Data tab holds the values of variables in the PL/SQL block. These are currently set to NULL.

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes tabs for dm_mods.sql, proc.sql, hr_orcl, and AWARD_BONUS. Below the menu is a toolbar with icons for code navigation and execution. The main area displays the PL/SQL code for the AWARD_BONUS procedure:

```

1  create or replace PROCEDURE award_bonus (
2      emp_id NUMBER, sales_amt NUMBER) AS
3      l_salary    REAL;
4      l_commission  REAL;
5      comm_missing EXCEPTION;
6
7  BEGIN
8      SELECT salary, commission_pct INTO l_salary, l_commission
9          FROM employees
10         WHERE employee_id = emp_id;
11         dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
12         dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
13
14      IF l_commission IS NULL THEN
15          RAISE comm_missing;
16      ELSE
17          l_salary := l_salary + l_salary * l_commission;

```

A red arrow icon on the left indicates a breakpoint is set on the first line of the SELECT statement. The status bar at the bottom right shows '7:1'. Below the code editor is a 'Debugging' toolbar with tabs for Log, Breakpoints, Smart Data (which is selected), Data, and Watches.

The 'Data' tab displays the current values of variables:

Name	Value	Type
L_COMMISSION	NULL	NUMBER
L_SALARY	NULL	NUMBER

Description of this image (files/tab05_02_11.txt)

9. You can see all the data manipulated in the procedure in the **Data** tab.

```

1  create or replace PROCEDURE award_bonus (
2      emp_id NUMBER, sales_amt NUMBER) AS
3          l_salary    REAL;
4          l_commission    REAL;
5          comm_missing EXCEPTION;
6
7  BEGIN
8      SELECT salary, commission_pct INTO l_salary, l_commission
9          FROM employees
10         WHERE employee_id = emp_id;
11         dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
12         dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
13     IF l_commission IS NULL THEN
14         RAISE comm_missing;
15     ELSE
16         -- additional code here
17     END IF;
18  END;
  
```

Debugging: IdeConnections%23hr_orcl.jpr - Log | Smart Data | **Data** [x] Watches | Breakpoints

Name	Value	Type
EMP_ID	149	NUMBER
SALES_AMT	2000	NUMBER
L_SALARY	NULL	NUMBER
L_COMMISION	NULL	NUMBER

Description of this image (files/tab05_02_12.txt)

You see that the current values of `l_salary` and `l_commission` are `NULL`.

10. Click the **Step Over** icon to move to the next statement in the procedure.

```

1 create or replace PROCEDURE award_bonus (
2     emp_id NUMBER, sales_amt NUMBER) AS
3     l_salary    REAL;
4     l_commission    REAL;
5     comm_missing EXCEPTION;
6 BEGIN
7     SELECT salary, commission_pct INTO l_salary, l_commission
8         FROM employees
9             WHERE employee_id = emp_id;
10            dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
11            dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
12    IF l_commission IS NULL THEN
13        RAISE comm_missing;
14    ELSE
15        l_salary := l_salary + sales_amt*l_commission;

```

Debugging: IdeConnections%23hr_orcl.jpr - Log

Connecting to the database hr_orcl.
 Executing PL/SQL: ALTER SESSION SET PLSQL_DEBUG=TRUE
 Executing PL/SQL: CALL DBMS_DEBUG_JDWP.CONNECT_TCP('127.0.0.1', '32709')
 Debugger accepted connection from database on port 32709.
 Source breakpoint: AWARD_BONUS.pls:7

Description of this image (files/tab05_02_13.txt)

11. Notice the values for l_salary and l_commission have changed to the existing values in the database, as the execution of select statement is complete, you can see the values from the database are fetched into the variables in the procedure.

Description of this image (files/tab05_02_14.txt)

- 12.** Click the **Step Over** icon again to move to the next statement. As the execution of the update statement completes, you can see the new values of salary and commission in the Data tab

```

8   FROM employees
9      WHERE employee_id = emp_id;
10     dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
11     dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
12  IF l_commission IS NULL THEN
13    RAISE comm_missing;
14  ELSE
15    l_salary := l_salary + sales_amt*l_commission;
16    dbms_output.put_line('Salary for Employee ID '||emp_id||' will be changed to:');
17    UPDATE employees
18      SET salary = l_salary
19      WHERE employee_id = emp_id;
20
21  END IF;
22 END award bonus;

```

Debugging: IdeConnections%23hr_orcl.jpr - Log | Smart Data **Data** | Watches | Breakpoints

Name	Value	Type
EMP_ID	149	NUMBER
SALES_AMT	2000	NUMBER
L_SALARY	10800	NUMBER
L_COMMISION	.2	NUMBER

Description of this image (files/tab05_02_15.txt)

13. Notice that the debugger moved to the next statement in the procedure. You want to run the rest of the procedure, click the **Resume** icon.

The screenshot shows the Oracle SQL Developer interface. The top window displays the PL/SQL code for the AWARD_BONUS procedure. The code includes a SELECT statement, a WHERE clause, DBMS_OUTPUT.PUT_LINE statements, an IF block, and an UPDATE statement. A breakpoint is marked at line 21. The bottom window shows the debug log, which includes the connection message, the execution of ALTER SESSION SET PLSQL_DEBUG=TRUE, and the acceptance of a debugger connection from port 21372.

```

8   FROM employees
9      WHERE employee_id = emp_id;
10     dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
11     dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
12  IF l_commission IS NULL THEN
13    RAISE comm_missing;
14  ELSE
15    l_salary := l_salary + sales_amt*l_commission;
16    dbms_output.put_line('Salary for Employee ID '||emp_id||' will be changed');
17    UPDATE employees
18      SET salary = l_salary
19      WHERE employee_id = emp_id;
20
21  END IF;
22 END award bonus;

```

Debugging: IdeConnections%23hr_orcl.jpr - Log

Connecting to the database hr_orcl.
 Executing PL/SQL: ALTER SESSION SET PLSQL_DEBUG=TRUE
 Executing PL/SQL: CALL DBMS_DEBUG_JDWP.CONNECT_TCP('127.0.0.1', '21372')
 Debugger accepted connection from database on port 21372.
 Source breakpoint: AWARD_BONUS.pls:7

Description of this image (files/tab05_02_16.txt)

14. Procedure execution and debugging is complete. In the next topic, you create a test repository so that you can create and run a unit test.

```

8   FROM employees
9     WHERE employee_id = emp_id;
10    dbms_output.put_line('Salary for Employee ID '||emp_id||' currently is:');
11    dbms_output.put_line('Commission Percentage for '||emp_id||' currently is:');
12  IF l_commission IS NULL THEN
13    RAISE comm_missing;
14  ELSE
15    l_salary := l_salary + sales_amt*l_commission;
16    dbms_output.put_line('Salary for Employee ID '||emp_id||' will be changed to:');
17    UPDATE employees
18      SET salary = l_salary
19        WHERE employee_id = emp_id;
20
21 END IF;
END award_bonus;

```

Debugging: IdeConnections%23hr_orcl.jpr - Log

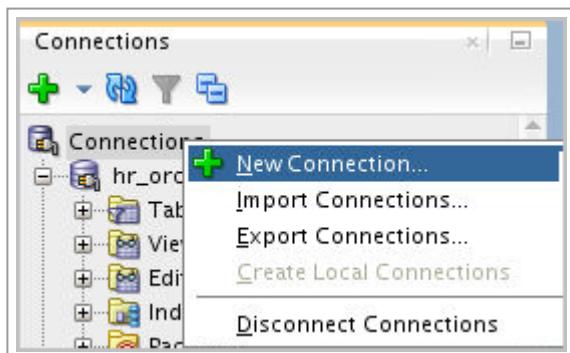
Connecting to the database hr_orcl.
 Executing PL/SQL: ALTER SESSION SET PLSQL_DEBUG=TRUE
 Executing PL/SQL: CALL DBMS_DEBUG_JDWP.CONNECT_TCP('127.0.0.1', '49685')
 Debugger accepted connection from database on port 49685.
 Source breakpoint: AWARD_BONUS.pls:7
 Executing PL/SQL: CALL DBMS_DEBUG_JDWP.DISCONNECT()
 Salary for Employee ID 149 currently is: 10400
 Commission Percentage for 149 currently is: .2
 Salary for Employee ID 149 will be changed to: 10800
 Process exited.
 Disconnecting from the database hr_orcl.
 Debugger disconnected from database.

Description of this image (files/tab05_02_17.txt)

Creating a Unit Test Repository

In this topic, you create a database user called UNIT_TEST_REPO. You create this user to hold the Unit Testing Repository data. You will then create the repository in the schema of the user that you created.

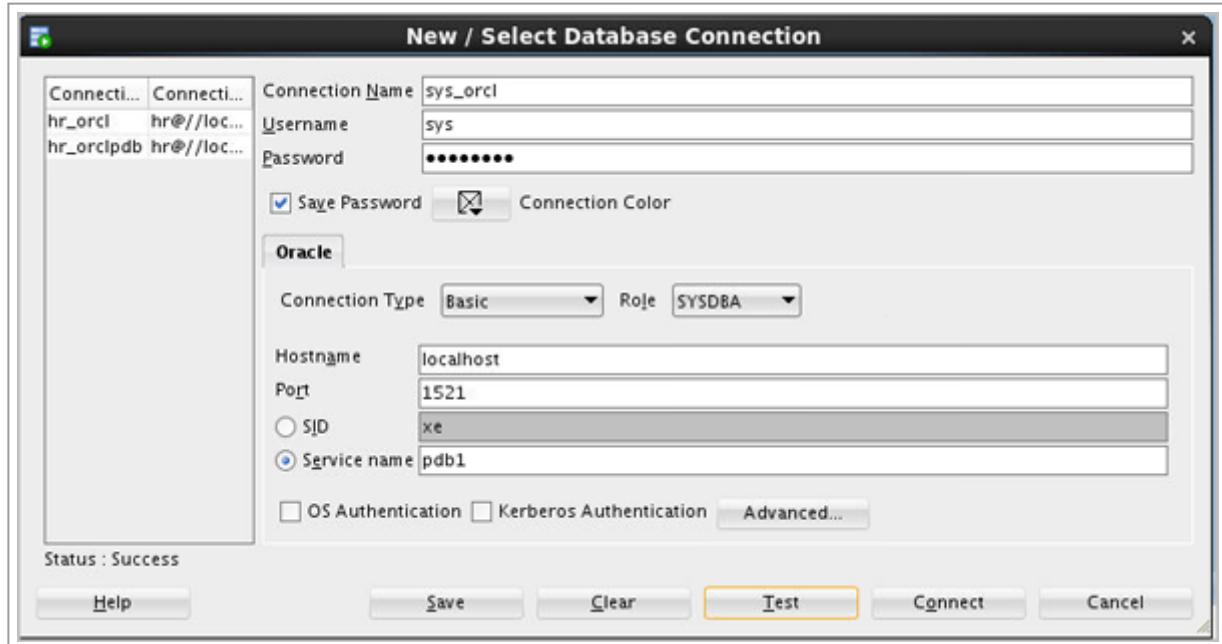
1. Create a connection for the SYS User. Right-click **Connections** and select **New Connection**.



Description of this image (files/tab06_01.txt)

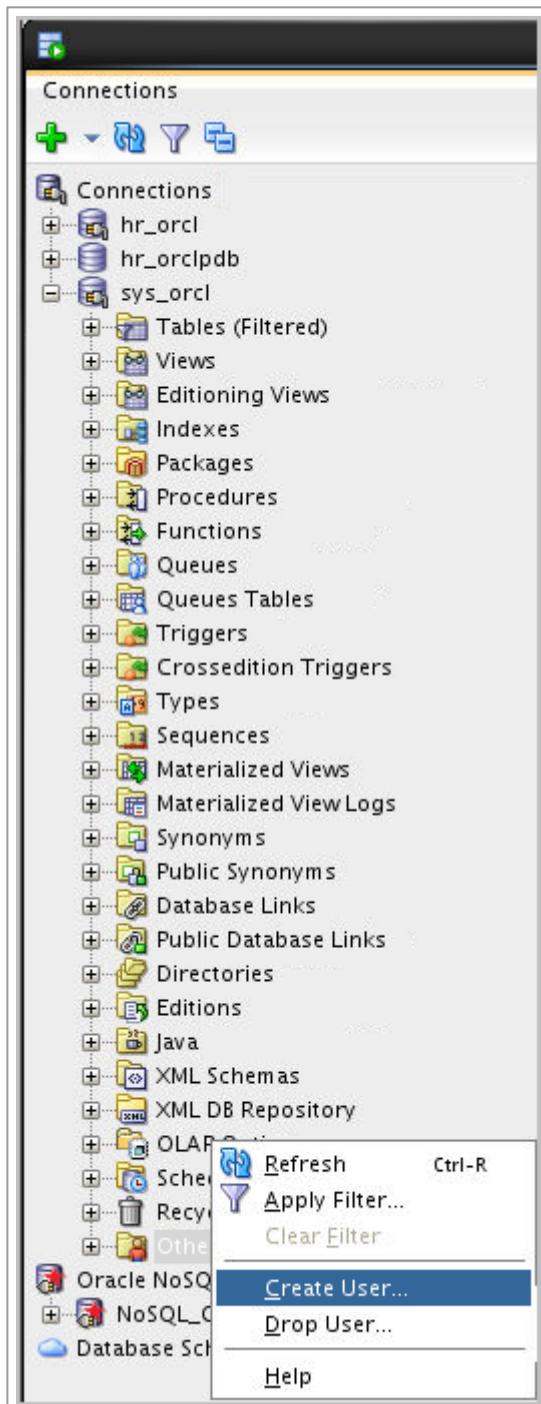
2. Enter the following information and click **Connect**.

Connection Name: **sys_orcl**
Username: **sys**
Password: **oracle**
Select **Save Password** checkbox
Role: **SYSDBA**
Service Name: **pdb1**



Description of this image (files/tab06_02.txt)

3. Your connection was created successfully. Collapse the **hr_orcl** connection. Expand the **sys_orcl** connection and right-click **Other Users** and select **Create User**.



Description of this image (files/tab06_03.txt)

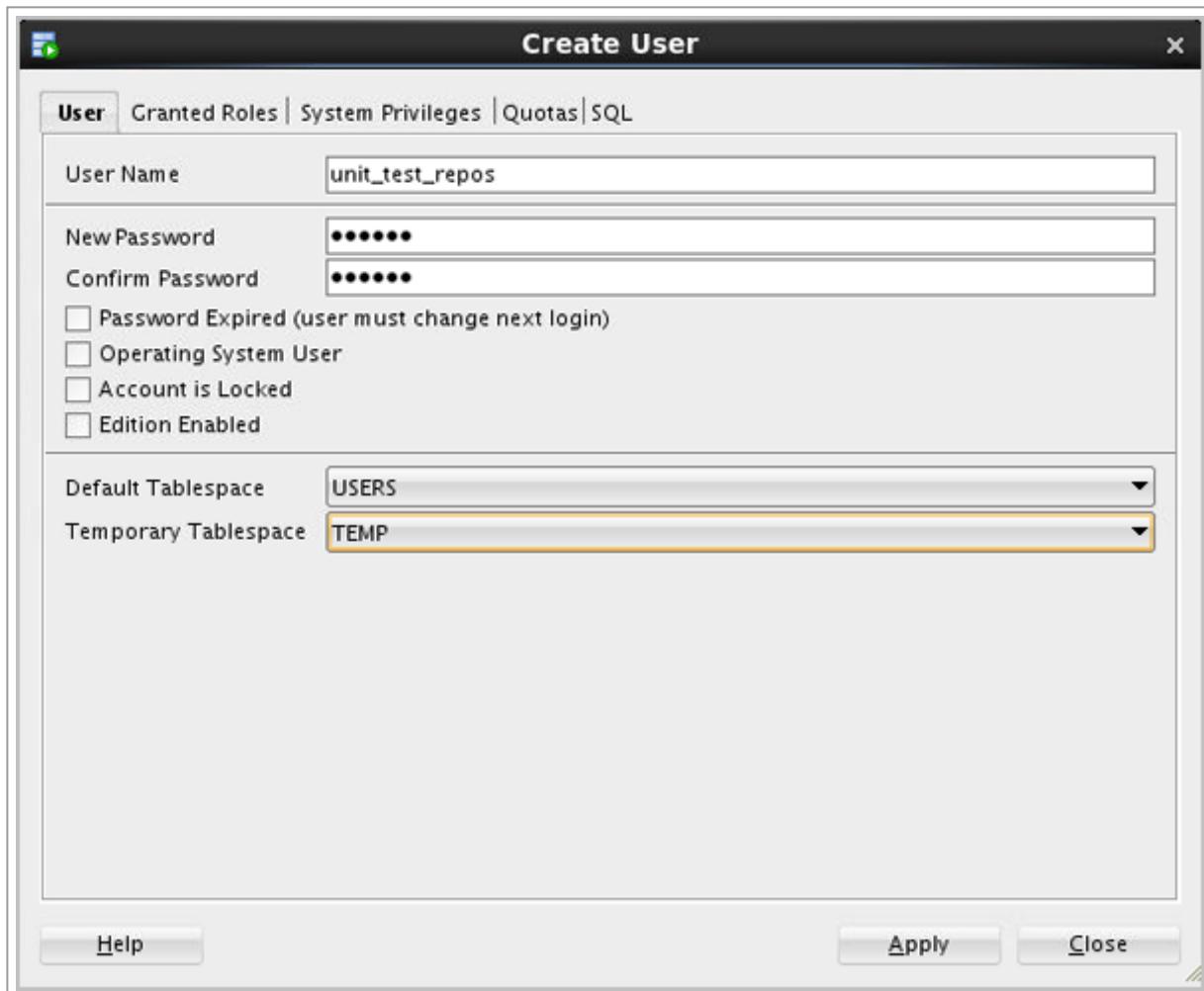
4. Enter the following information and select the **Granted Roles** tab.

Username: **unit_test_repos**

Password: **oracle**

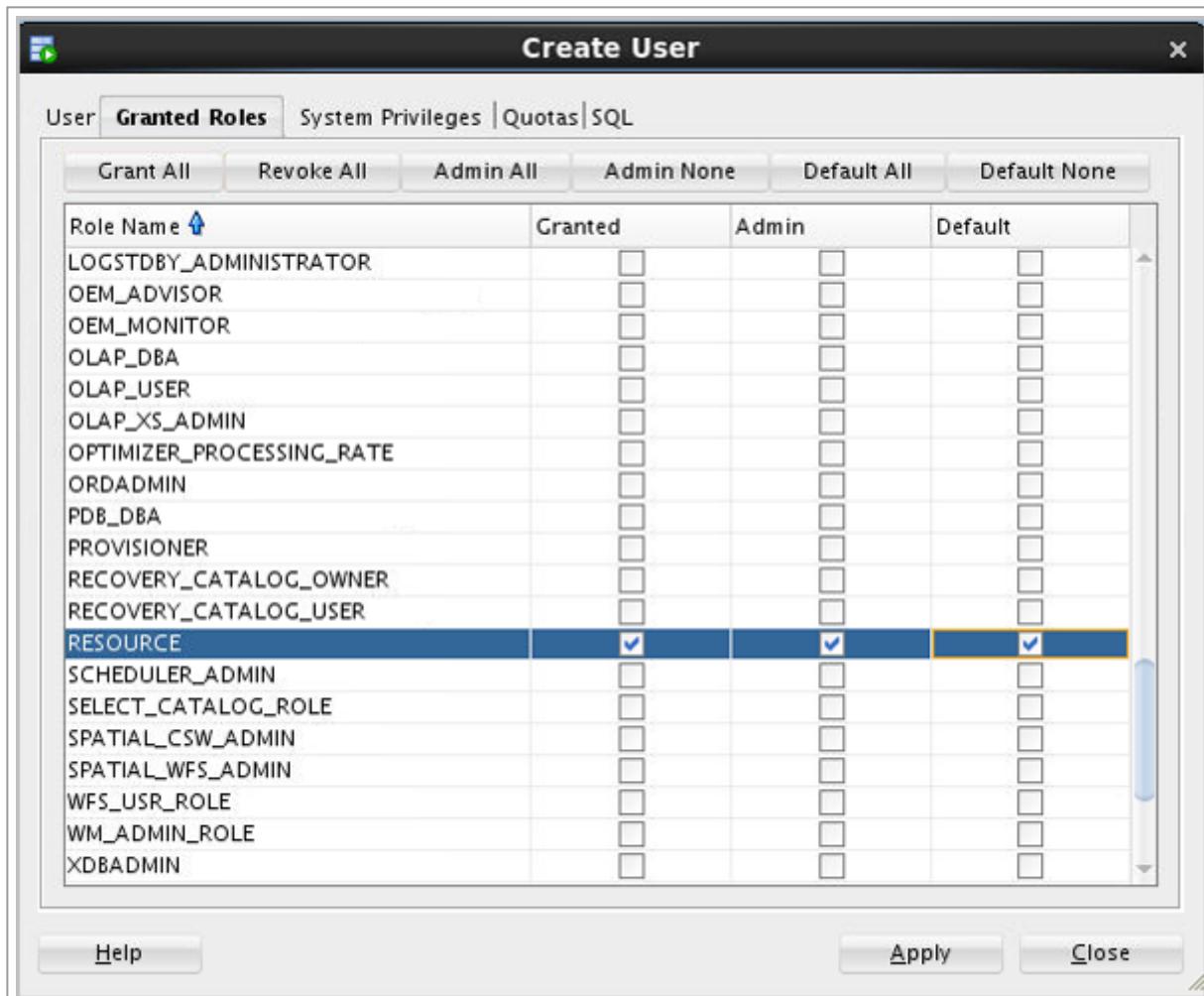
Default Tablespace: **USERS**

Temporary Tablespace: **TEMP**



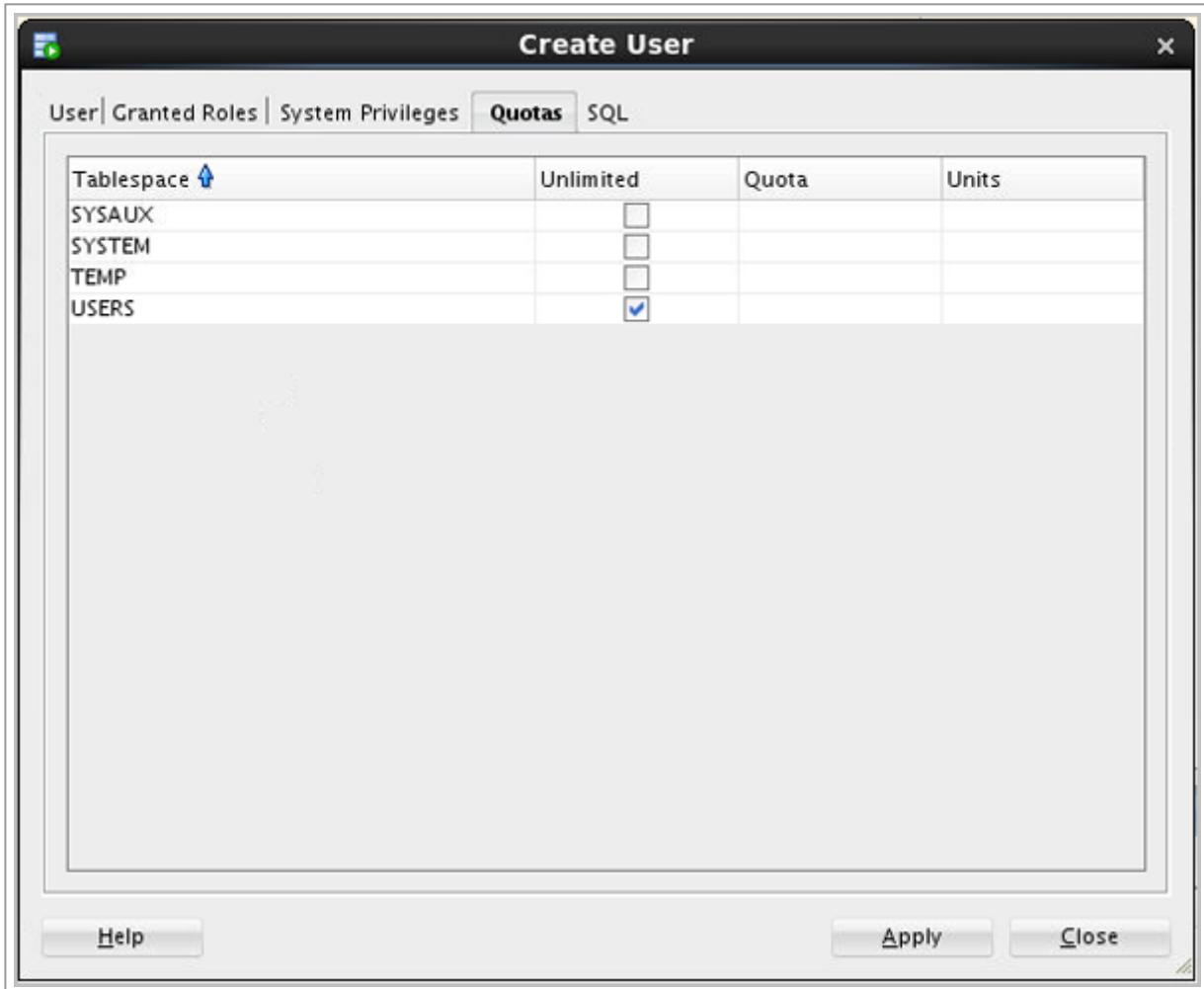
Description of this image (files/tab06_04.txt)

5. Select the **Connect** and **Resource** roles and click **Apply**.



Description of this image (files/tab06_05.txt)

6. In the Quotas tab, check the Unlimited check box for the USERS tablespace



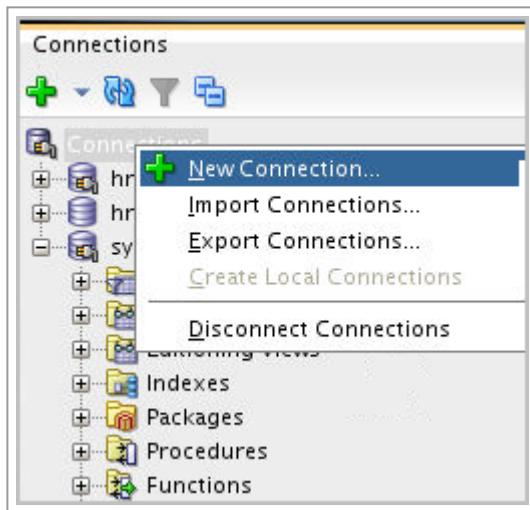
Description of this image (files/tab06_06.txt)

7. The unit_test_repos user was created successfully. Click **OK**.



Description of this image (files/tab06_07.txt)

8. You now need to create a connection to the unit_test_repos user. This user will hold the unit testing repository data. Right-click **Connections** and select **New Connection**.



Description of this image (files/tab06_08.txt)

9. Enter the following information and click **Connect**.

Connection Name: **unit_test_repos_orcl**

Username: **unit_test_repos**

Password: **oracle**

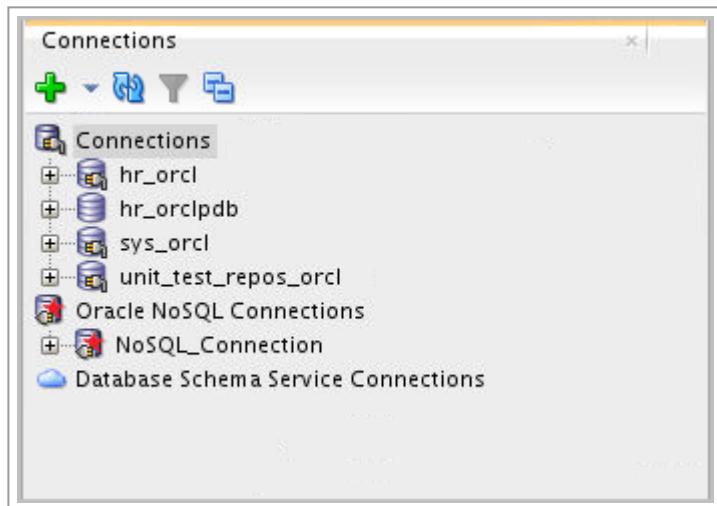
Select **Save Password** checkbox

Service Name: **pdb1**



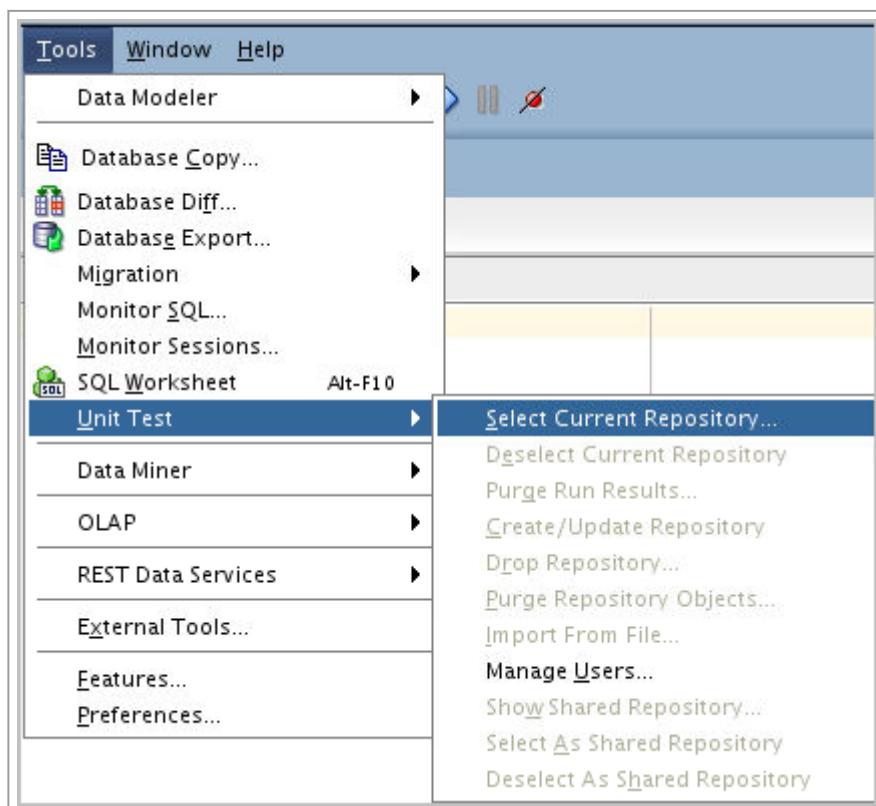
Description of this image (files/tab06_09.txt)

10. The unit_test_repos user and unit_test_repos_orcl connection were created successfully.



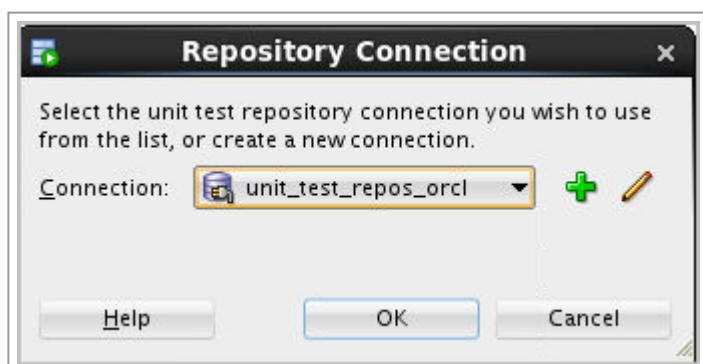
Description of this image (files/tab06_10.txt)

11. Select Tools > Unit Test > Repository, then select Select Current Repository.



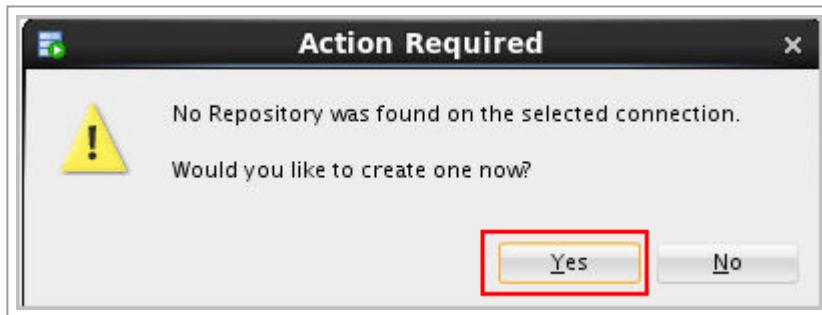
Description of this image (files/tab06_11.txt)

12. Select the unit_test_repos_orcl connection and click OK.



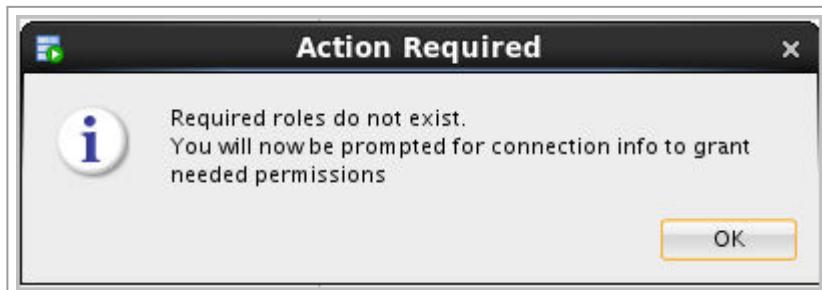
Description of this image (files/tab06_12.txt)

13. You would like to create a new repository. Click **Yes**.



Description of this image (files/tab06_13.txt)

14. This connection does not have the permissions it needs to create the repository. Click **OK** to show the permissions that will be applied.



Description of this image (files/tab06_14.txt)

15. Enter **oracle** for the sys password and click **OK**.



Description of this image (files/tab06_15.txt)

16. The grant statement is shown. Click **Yes**.



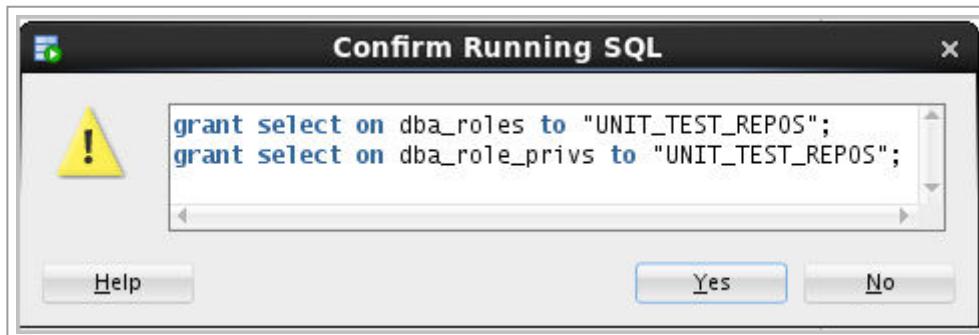
Description of this image (files/tab06_16.txt)

17. The UNIT_TEST_REPO\$ user needs select access to some required tables. Click **OK**.



Description of this image (files/tab06_17.txt)

18. The grant statements are displayed. Click **Yes**.



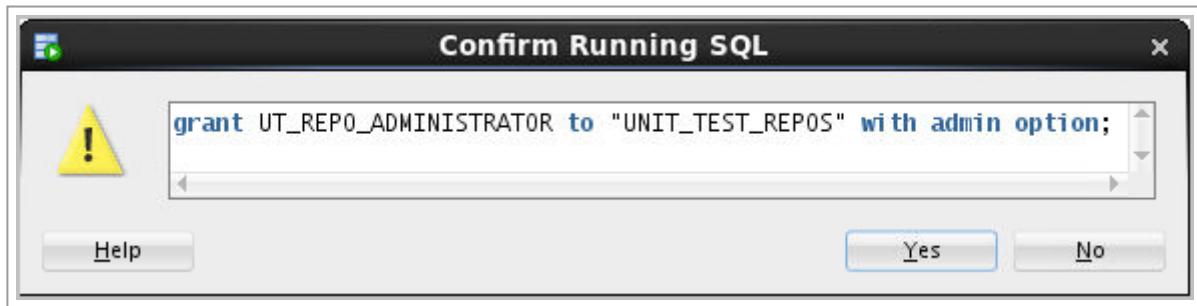
Description of this image (files/tab06_18.txt)

19. The UNIT_TEST_REPO\$ user does not currently have the ability to manage repository owners. Click **OK** to see the grant statements that will be executed.



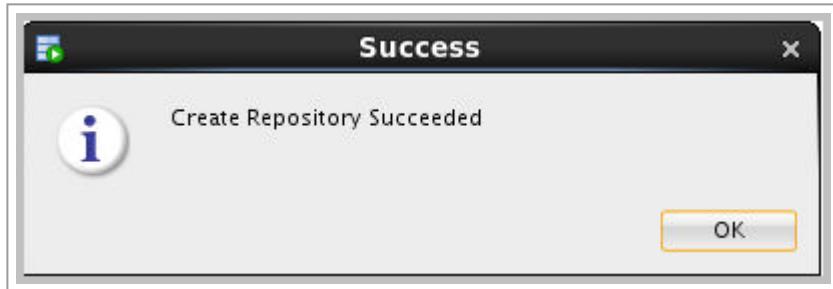
Description of this image (files/tab06_19.txt)

20. The grant statements are displayed. Click **Yes**.



Description of this image (files/tab06_20.txt)

21. Your repository was created successfully. Click **OK**.

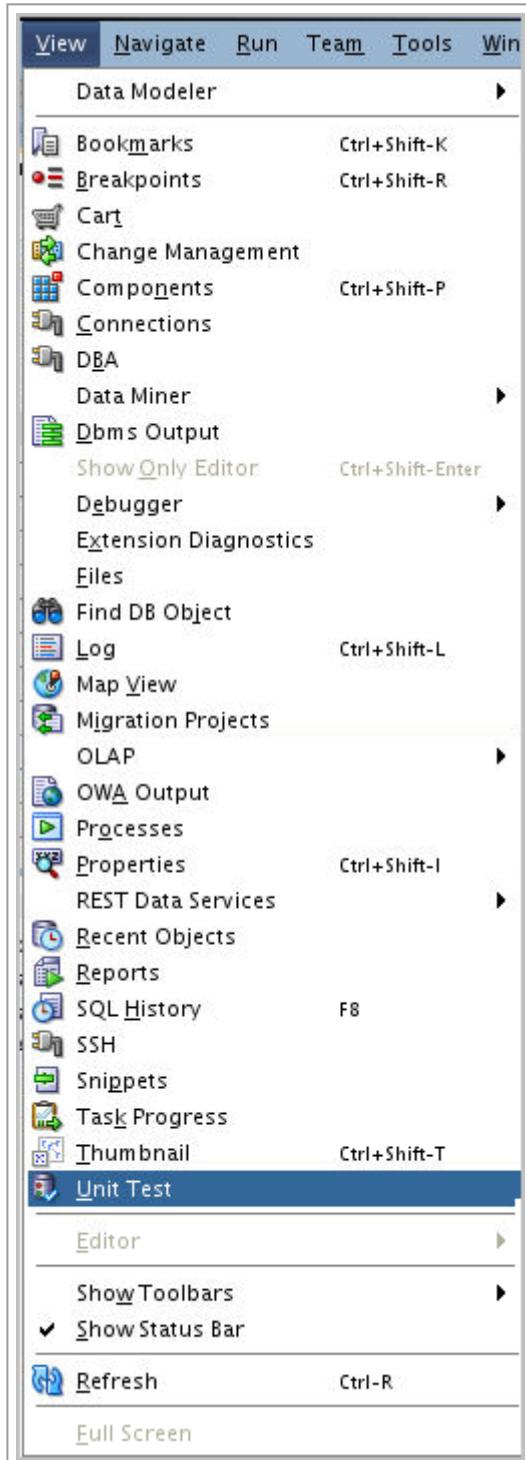


Description of this image (files/tab06_21.txt)

Creating and Running a Unit Test

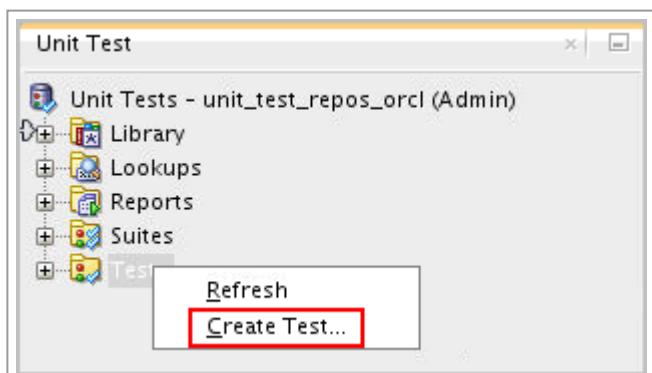
Now that the Unit Testing Repository has been created, you will create a unit test for the PL/SQL procedure you created earlier in this tutorial. Then you will run the unit test to see if various values will work.

1. Select **View > Unit Test**.



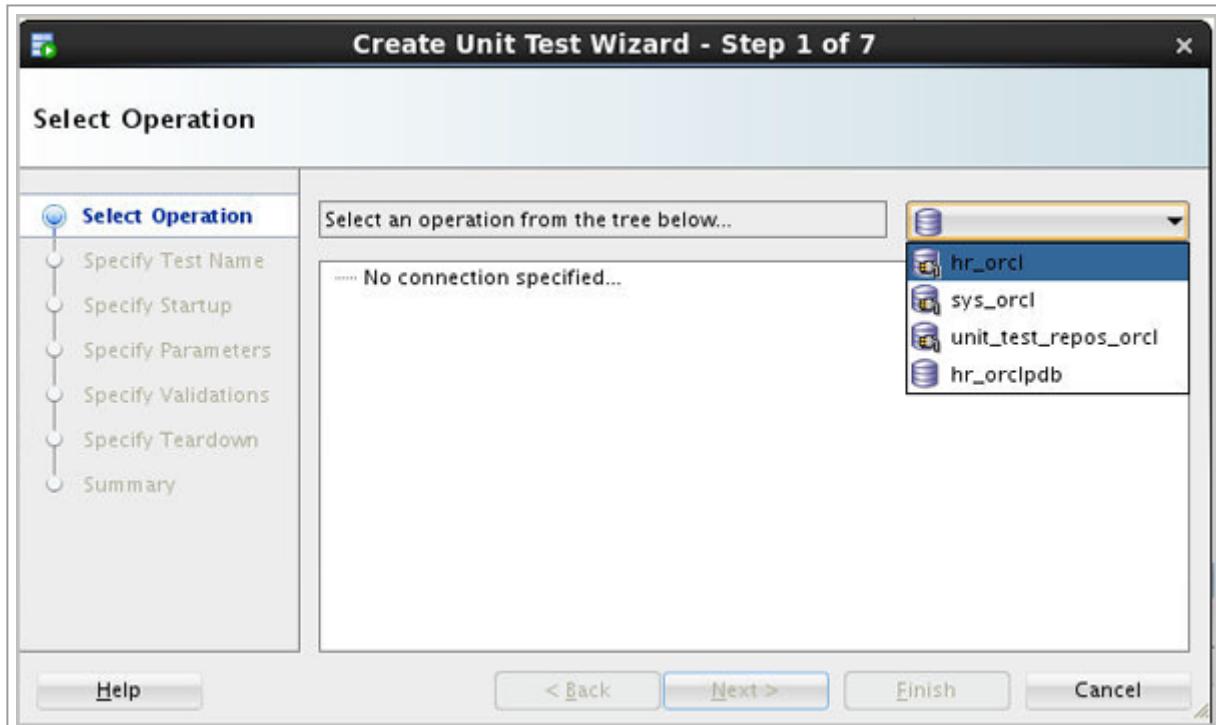
Description of this image (files/tab07_01.txt)

2. In the Unit Test navigator, right-click **Tests** and select **Create Test**.



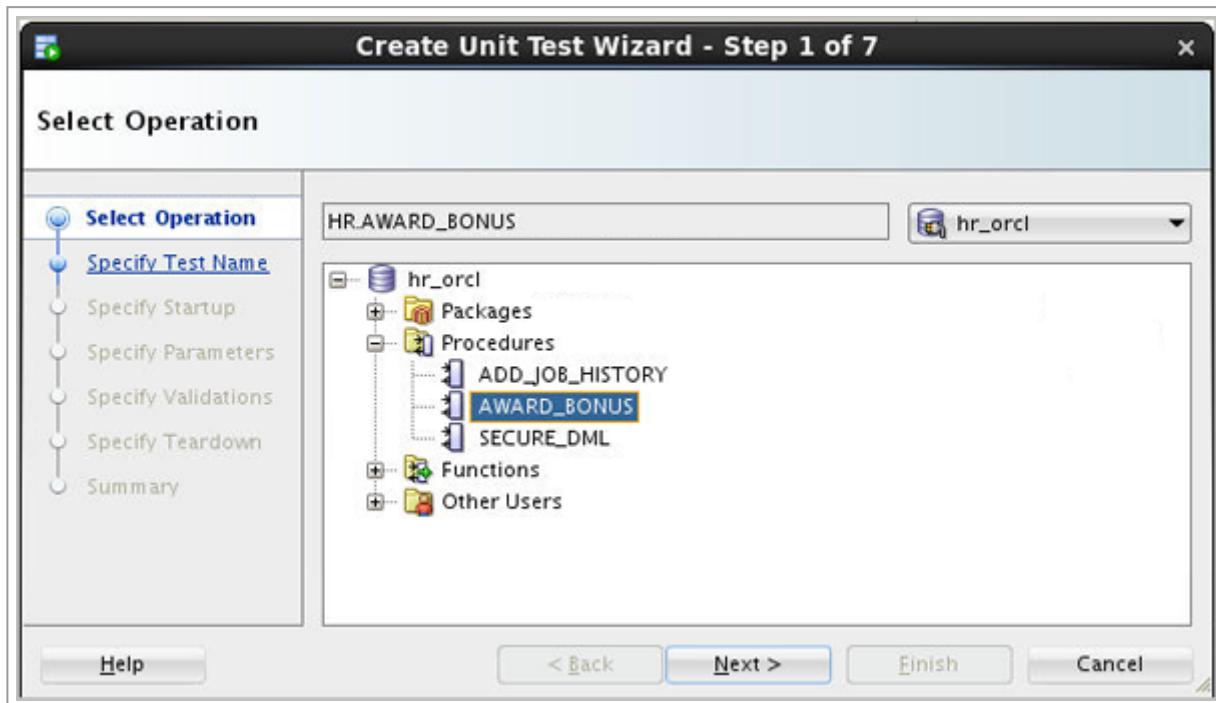
Description of this image (files/tab07_02.txt)

3. In Select Operation, select the **hr_orcl** connection that you used to create the **AWARD_BONUS** procedure.



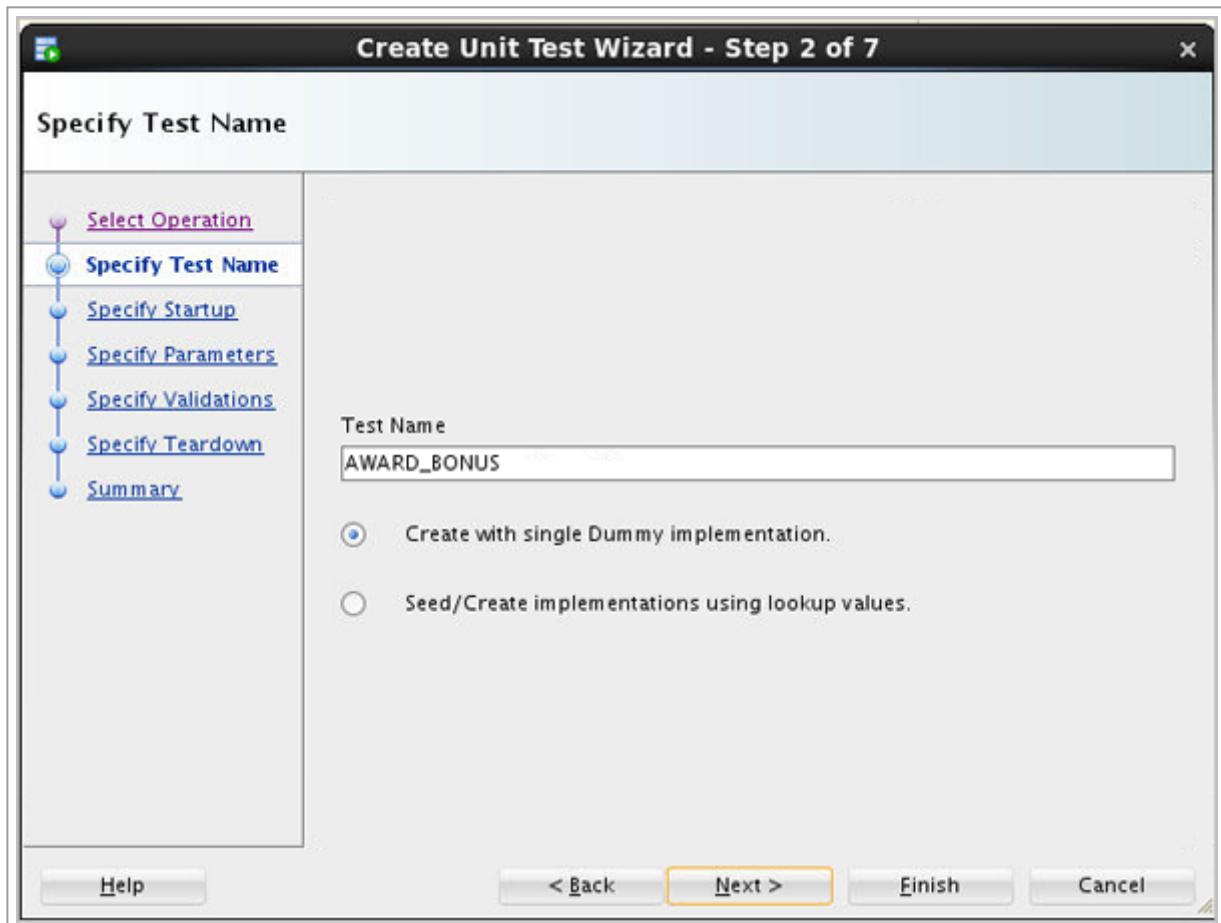
Description of this image (files/tab07_03.txt)

4. Expand **Procedures**, select **AWARD_BONUS** and click **Next**.



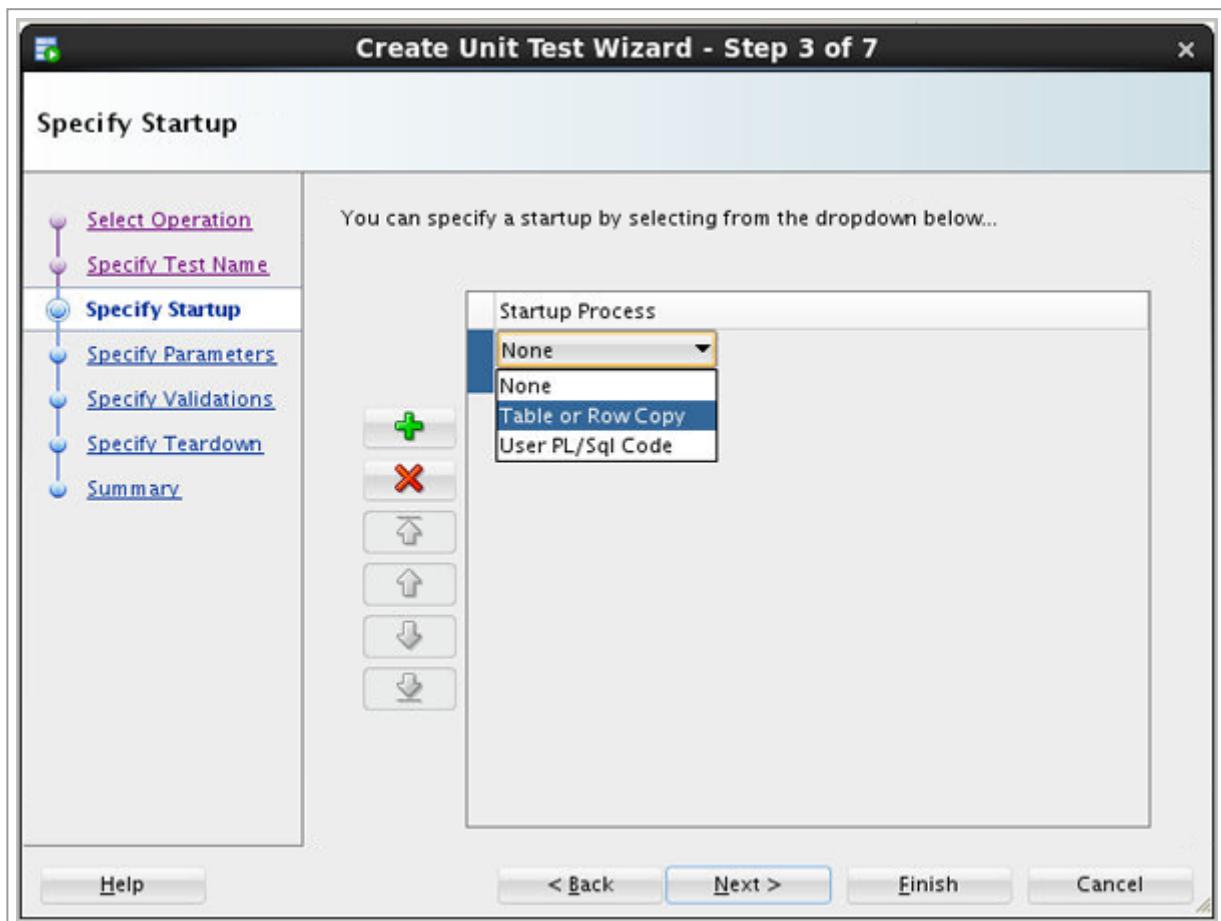
Description of this image (files/tab07_04.txt)

5. In Specify Test Name window, make sure that **AWARD_BONUS** is specified for Test Name and that **Create with single Dummy implementation** is selected, then click **Next**.



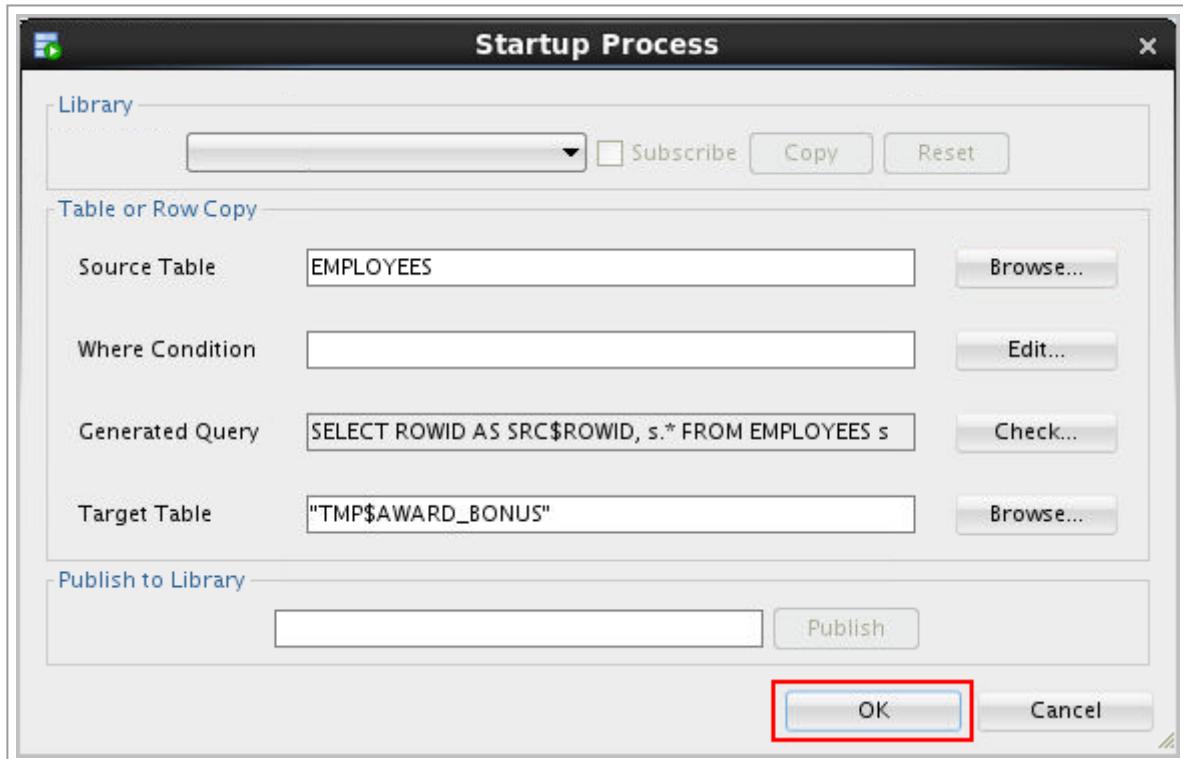
Description of this image (files/tab07_05.txt)

6. In Specify Startup window, click '+' icon and select **Table or Row Copy** from the drop down list box.



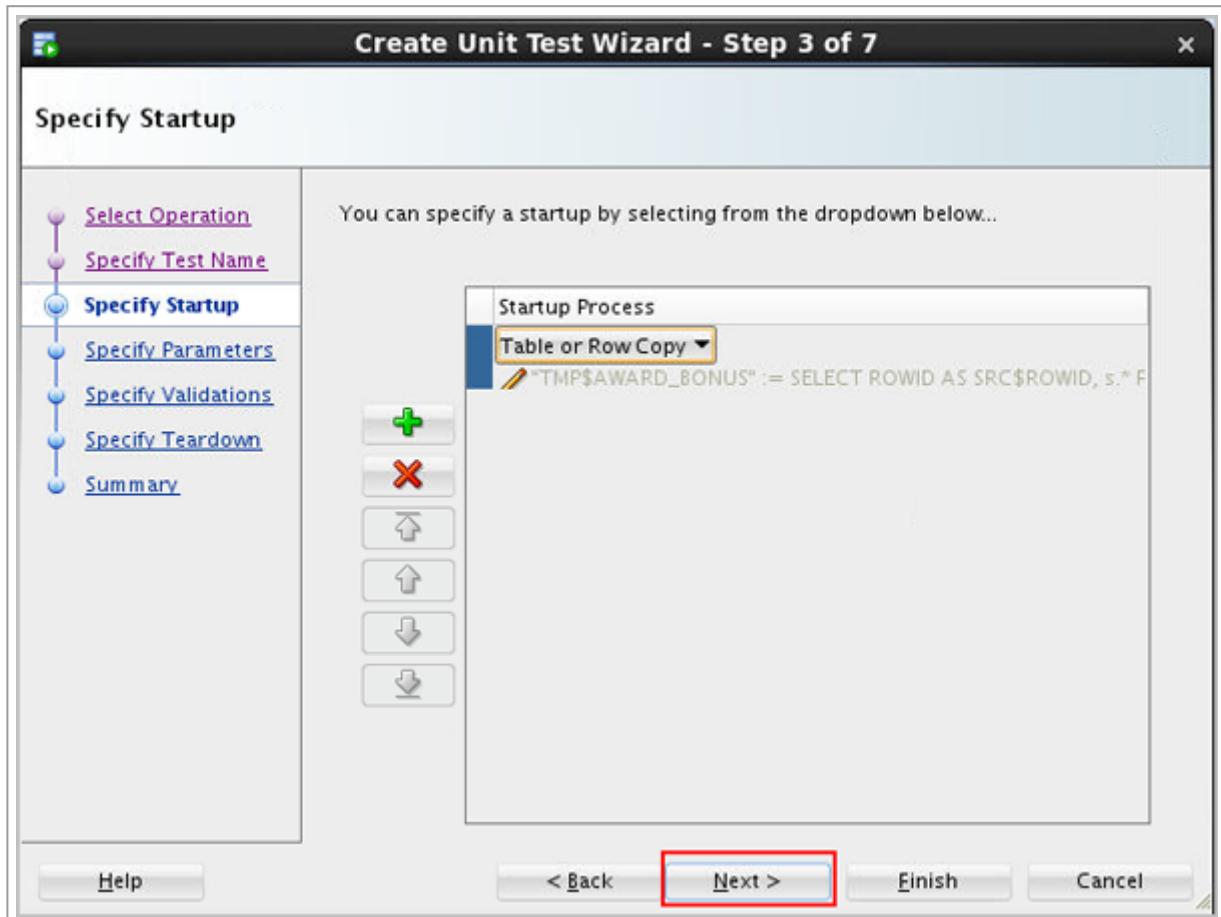
Description of this image (files/tab07_06.txt)

7. Enter **EMPLOYEES** for Source Table and click **OK**. Note that the table affected by the test will be saved to a temporary table and the query to the table is automatically generated.



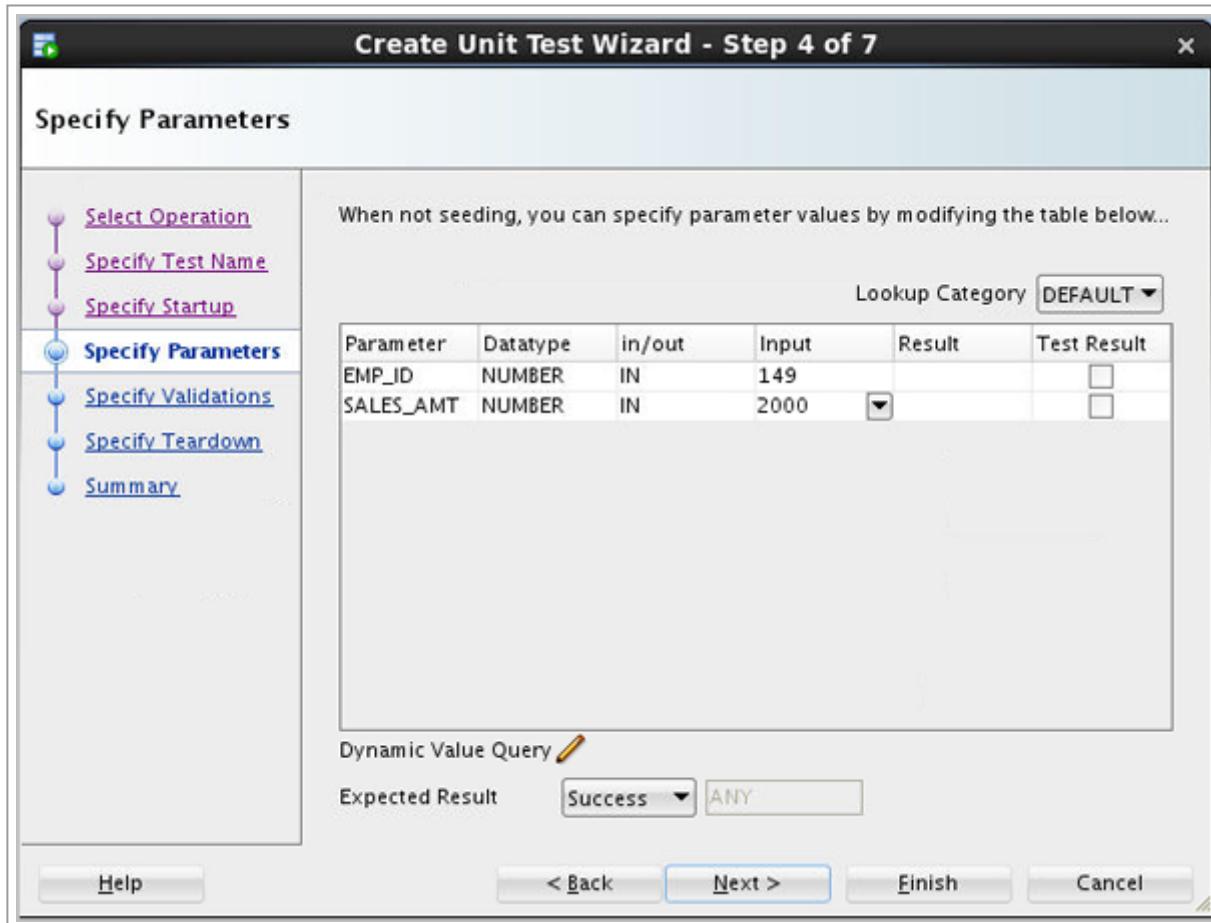
Description of this image (files/tab07_07.txt)

8. Click **Next**.



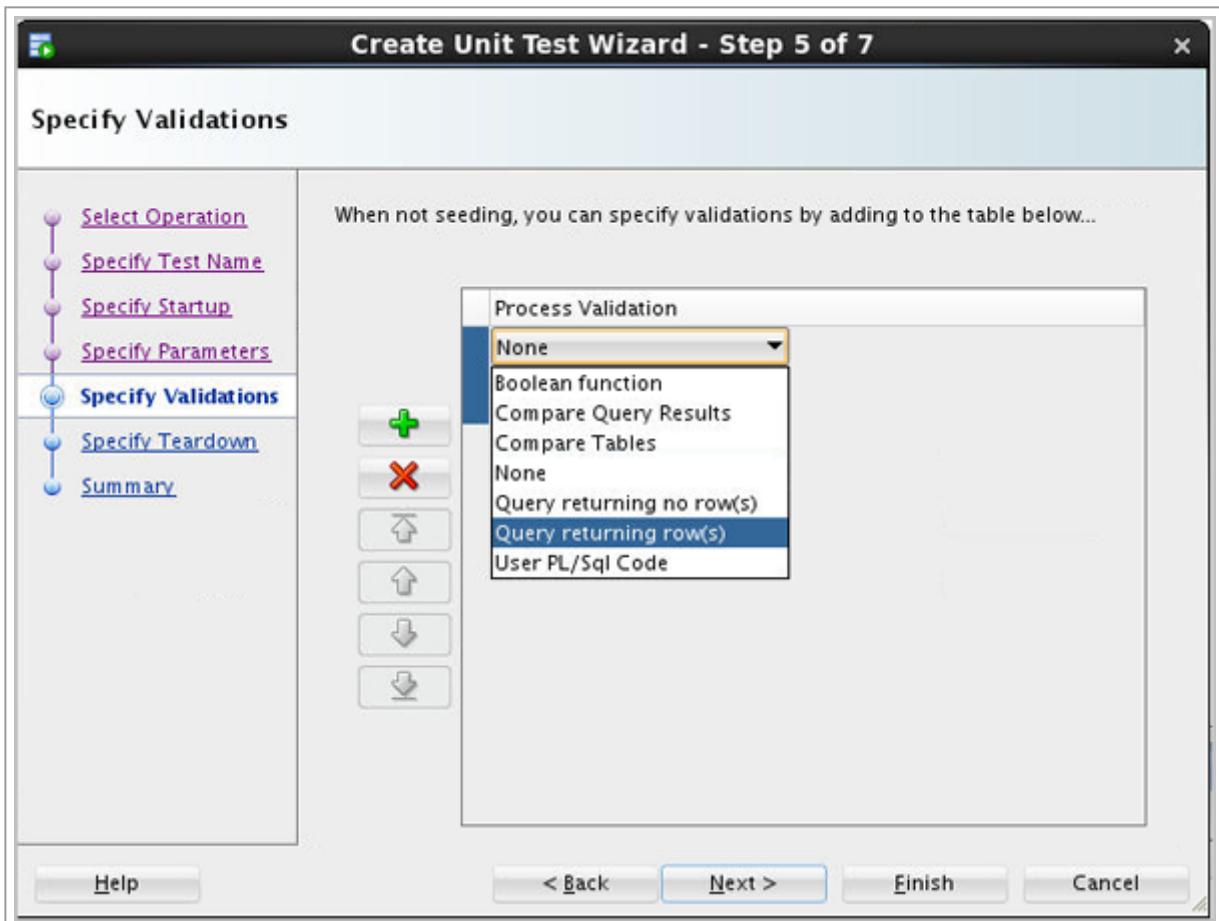
Description of this image (files/tab07_08.txt)

9. In the Specify Parameters window, change the Input string for EMP_ID to **149** and SALES_AMT to **2000** and click **Next**.



Description of this image (files/tab07_09.txt)

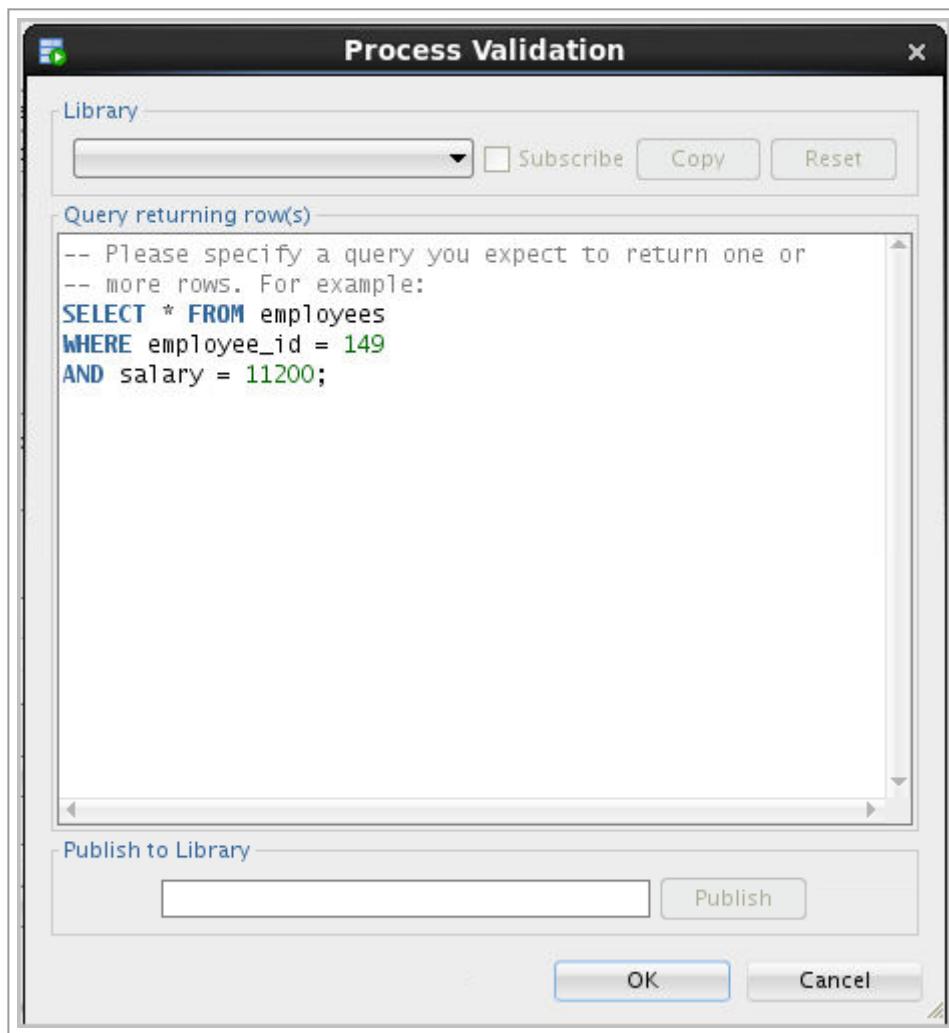
10. Select the '+' icon to add a validation and select **Query returning row(s)** from the drop down list.



Description of this image (files/tab07_10.txt)

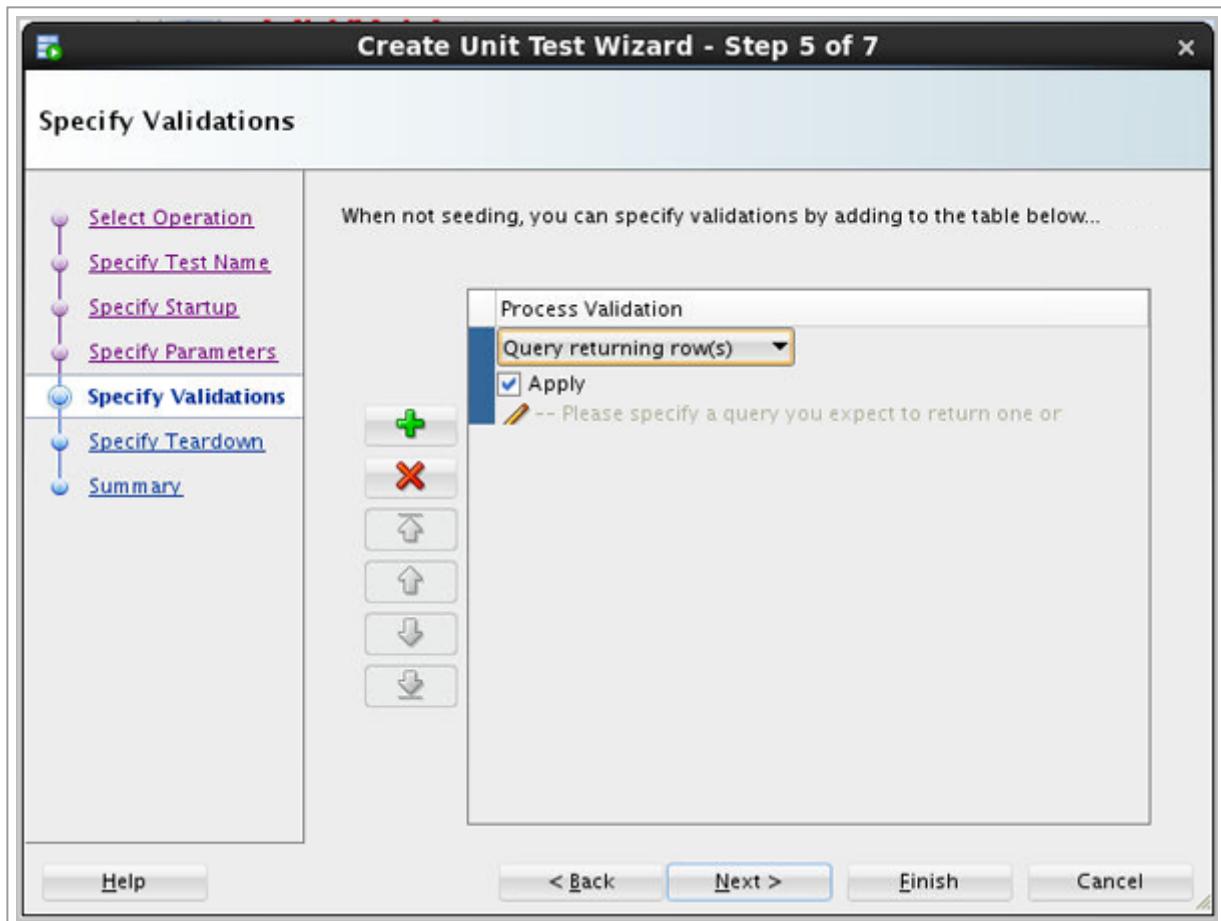
11. Specify the following query and click **OK**. This query will test the results of the change that the unit test performed.

```
SELECT * FROM employees
WHERE employee_id = 149
AND salary = 11200;
```



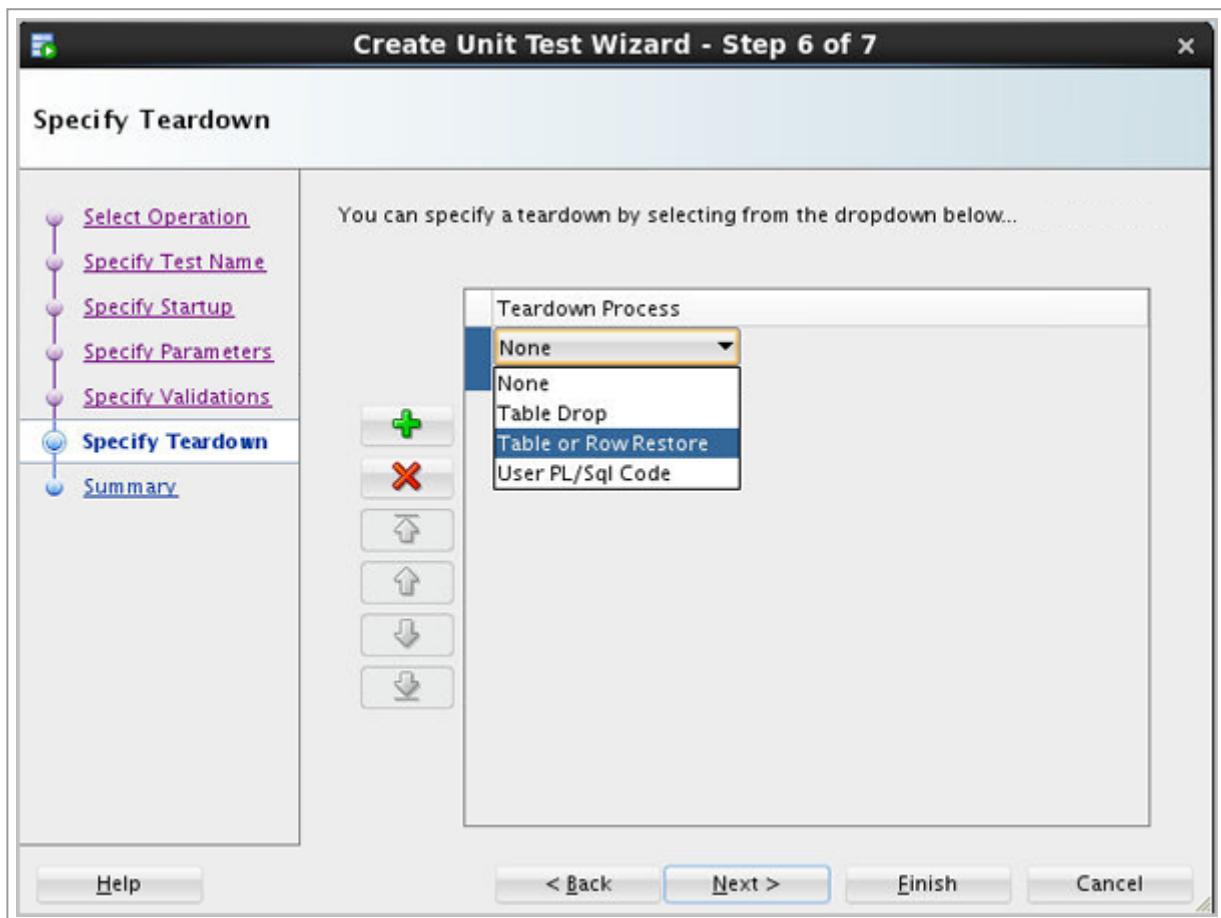
Description of this image (files/tab07_11.txt)

12. Click Next.



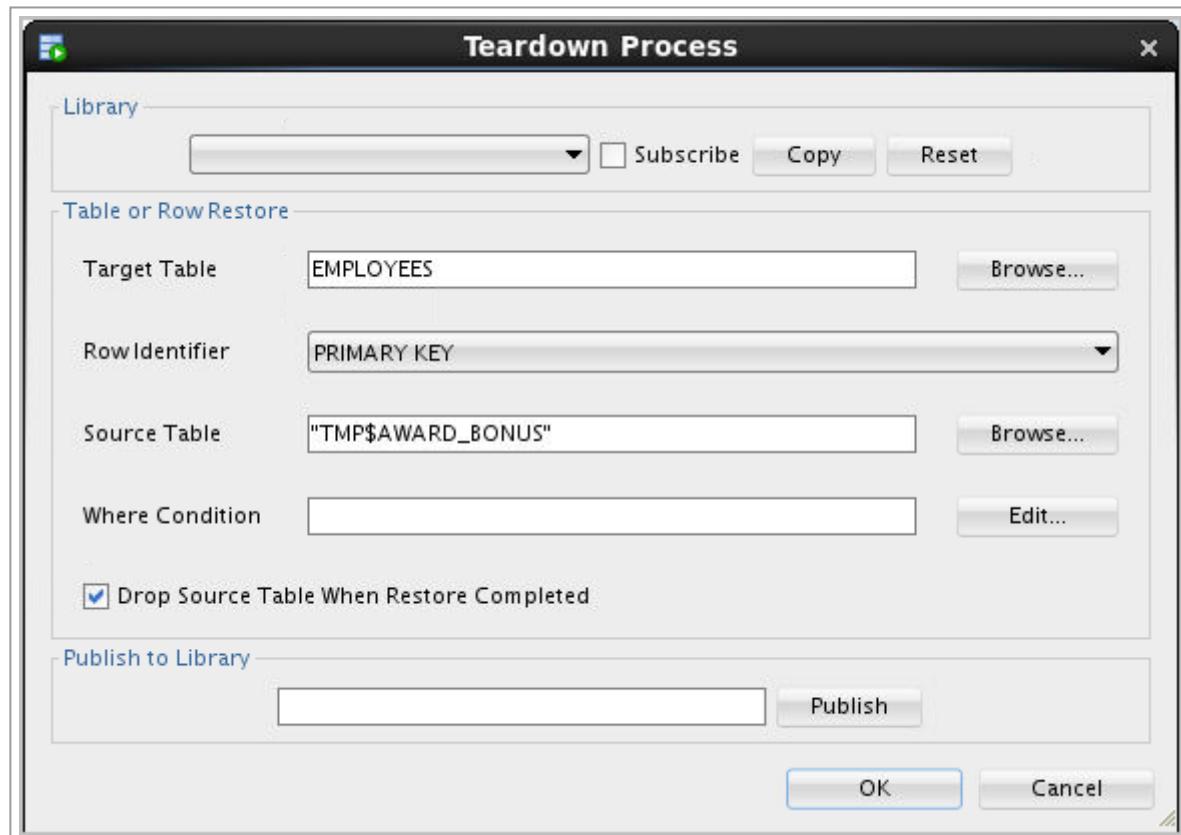
Description of this image (files/tab07_12.txt)

13. In the Specify Teardown window, click the '+' icon and select **Table or Row Restore** from the drop down list.



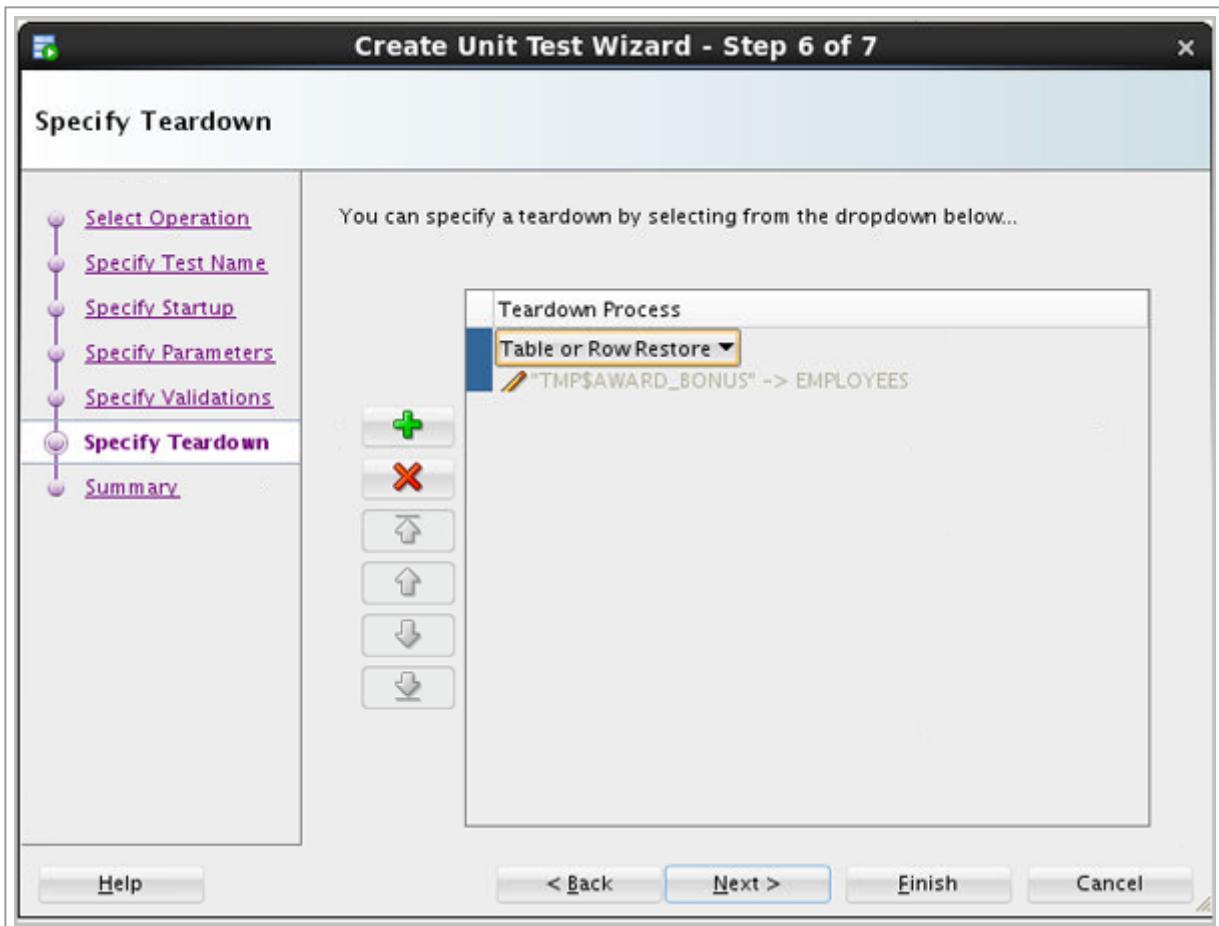
Description of this image (files/tab07_13.txt)

14. Leave the Row Identifier as Primary Key and click **OK**.



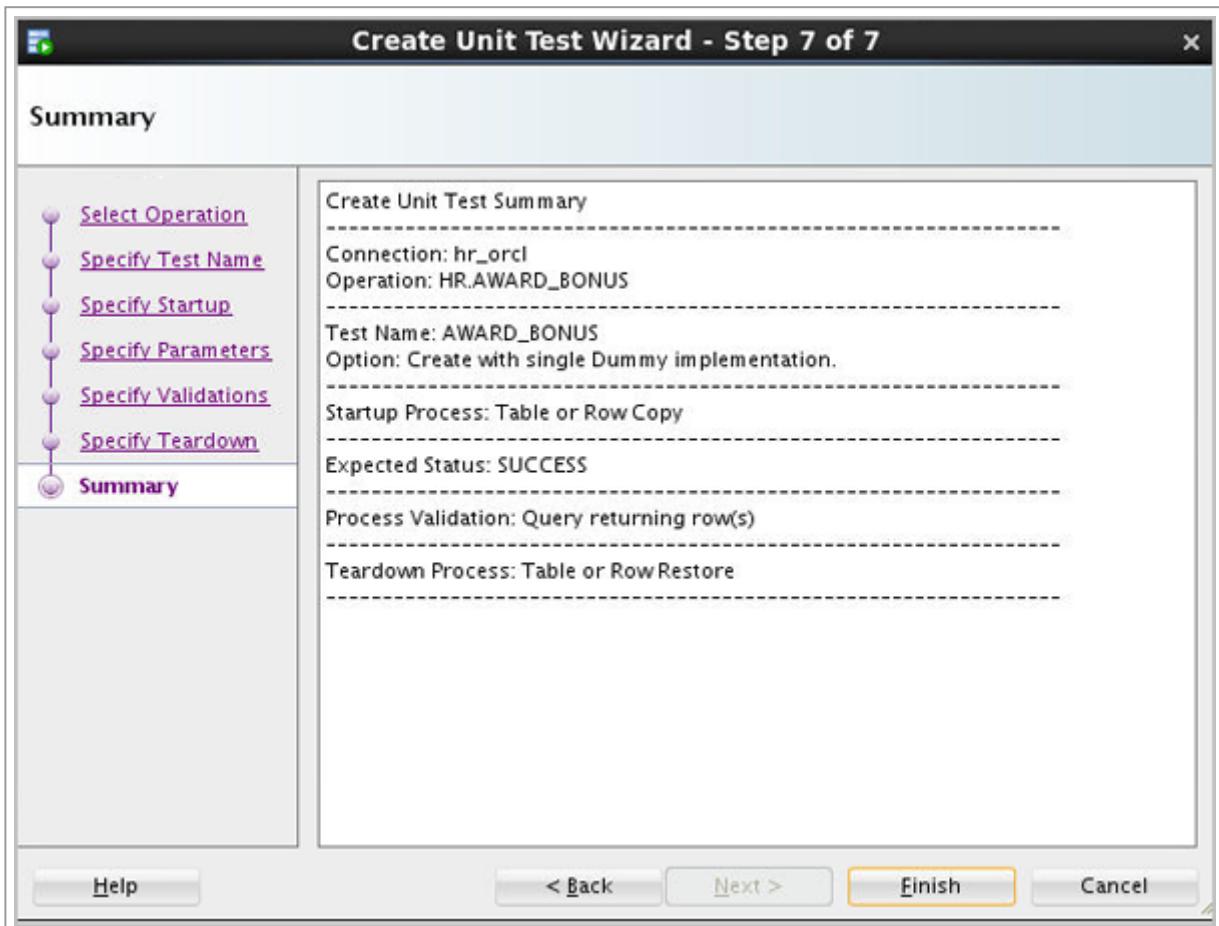
Description of this image (files/tab07_14.txt)

15. Click **Next**.



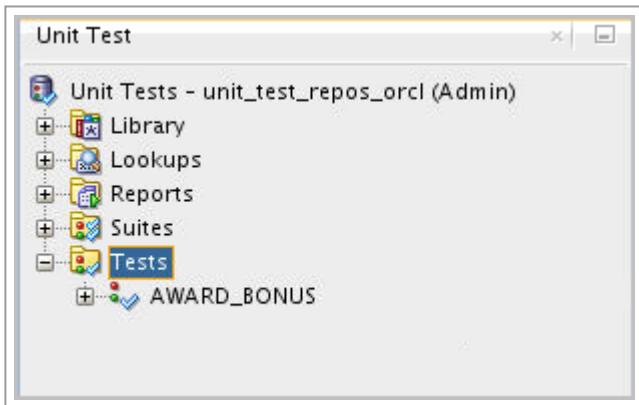
Description of this image (files/tab07_15.txt)

16. Click Finish.



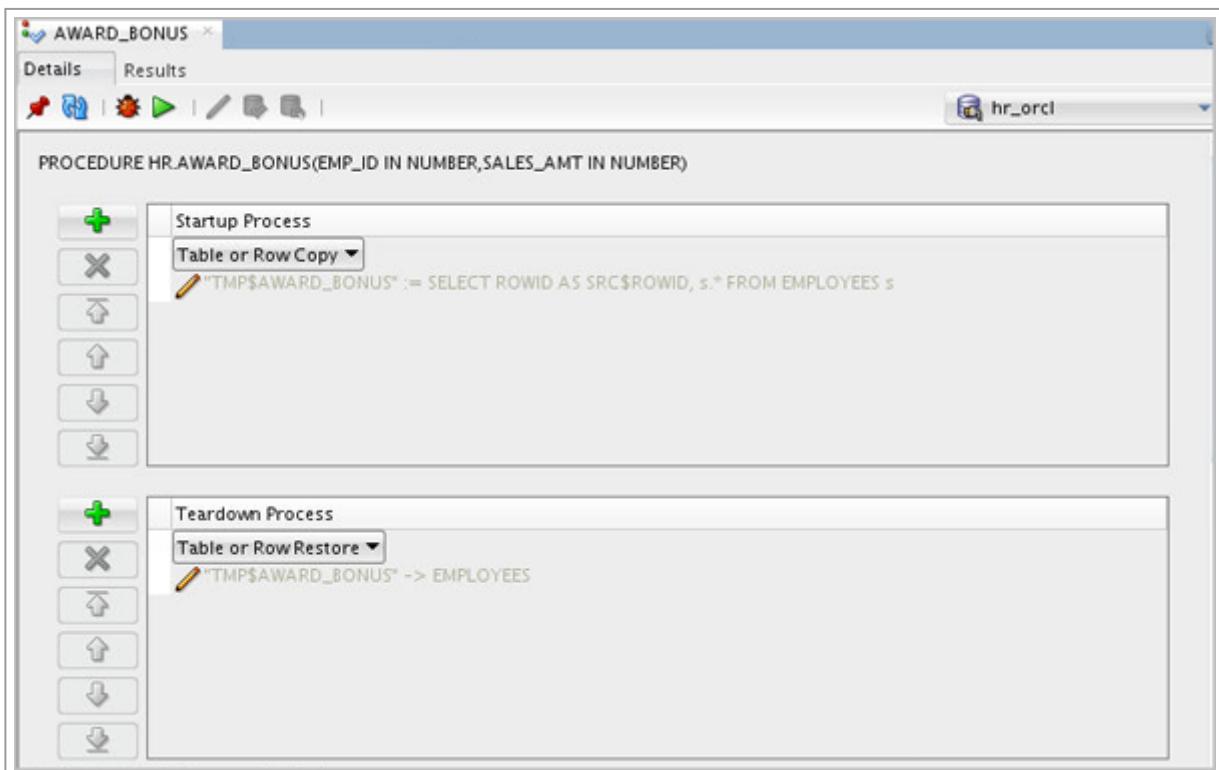
Description of this image (files/tab07_16.txt)

17. Expand **Tests**. Your test appears in the list.



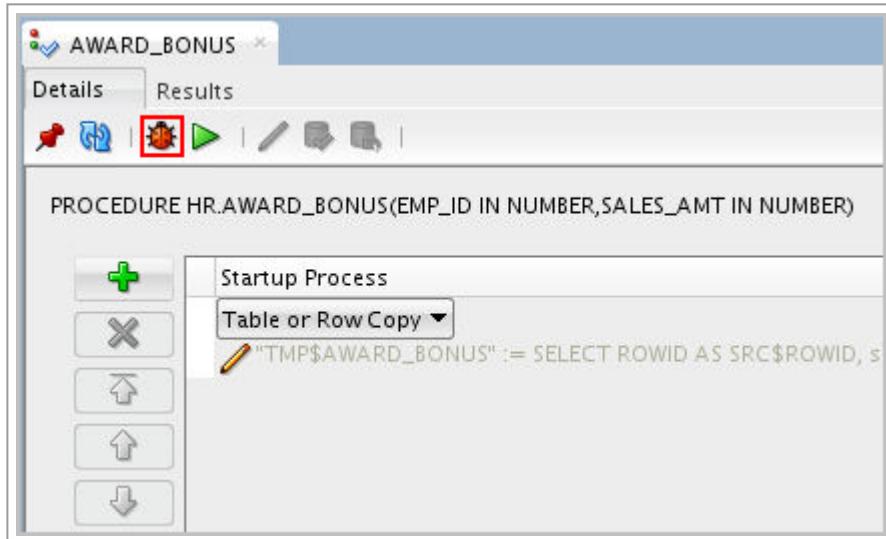
Description of this image (files/tab07_17.txt)

18. Select the **AWARD_BONUS** test in the left navigator. Notice that the test details are displayed on the right panel.



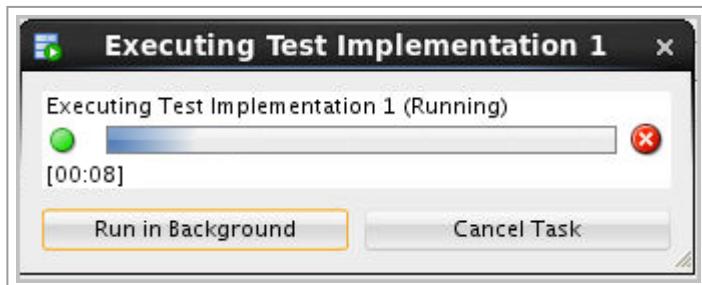
Description of this image (files/tab07_18.txt)

19. Run the test by clicking the **Debug Implementation** icon.



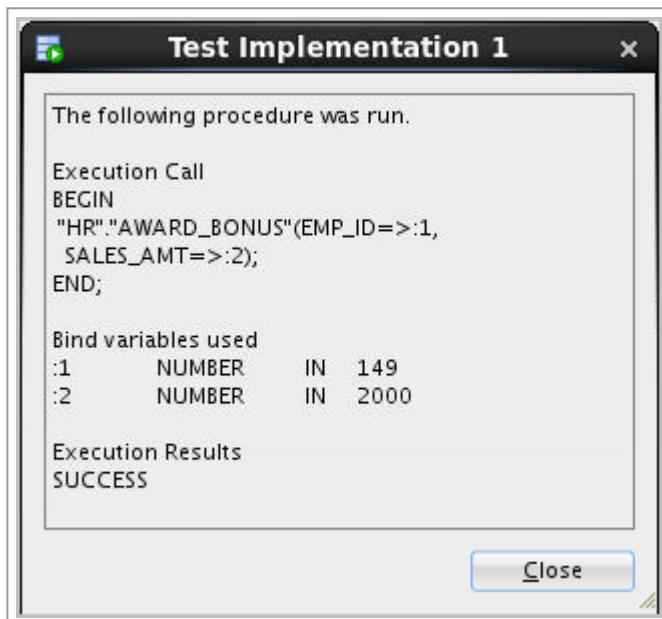
Description of this image (files/tab07_19.txt)

- 20.** The Progress window appears.



Description of this image (files/tab07_20.txt)

- 21.** When the test completes, the results are displayed. Click **Close**.



Description of this image (files/tab07_21.txt)

Want to Learn More?

- Oracle SQL Developer Data Modeler on OTN (<http://www.oracle.com/technetwork/developer-tools/datamodeler/overview/index.html>)
- Oracle Data Modeling and Relational Database Design course (http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getCourseDesc?dc=D56497GC10&p_org_id=1080544&lang=US)
- Oracle Learning Library (<http://www.oracle.com/oll>)

About Oracle (<http://www.oracle.com/corporate/index.html>) Contact Us (<http://www.oracle.com/us/corporate/contact/index.html>)

Legal Notices (<http://www.oracle.com/us/legal/index.html>) Terms of Use (<http://www.oracle.com/us/legal/terms/index.html>)

Your Privacy Rights (<http://www.oracle.com/us/legal/privacy/index.html>)

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.