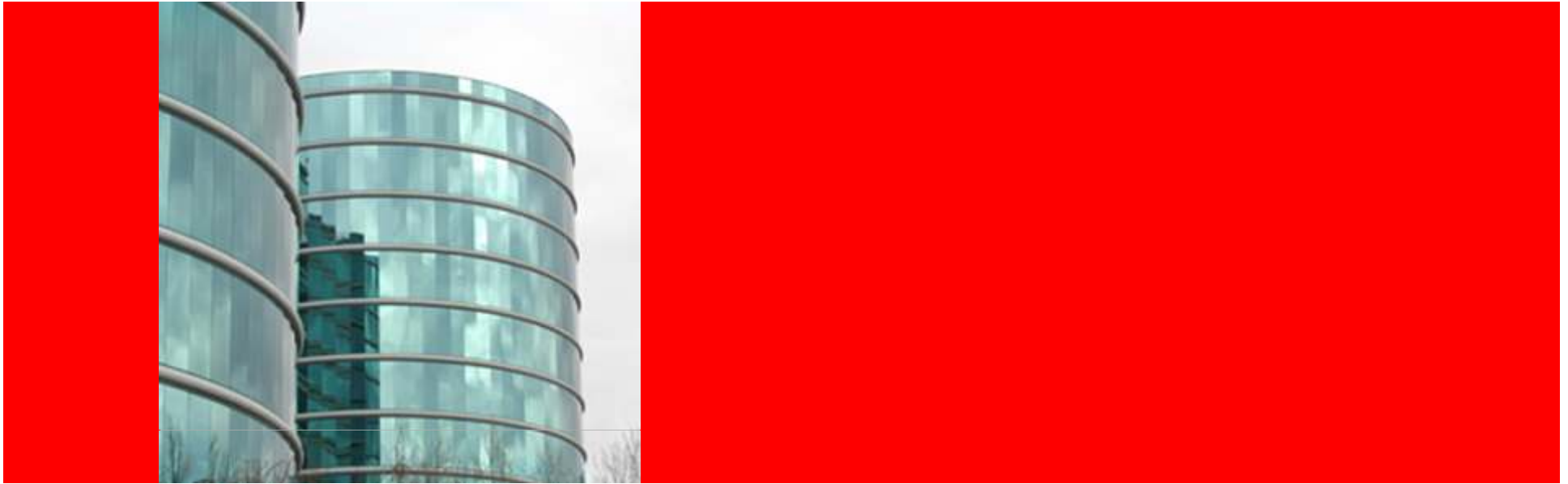





ORACLE®



**ORACLE®**

Oracle SQL Developer: PL/SQL Support and Unit  
Testing



The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.



## Agenda

- SQL Developer 2.1 New Features
- Working with PL/SQL
- Remote Debugging
- Unit Testing
- Other PL/SQL Related Features



## SQL Developer 2.1 – New Feature Overview

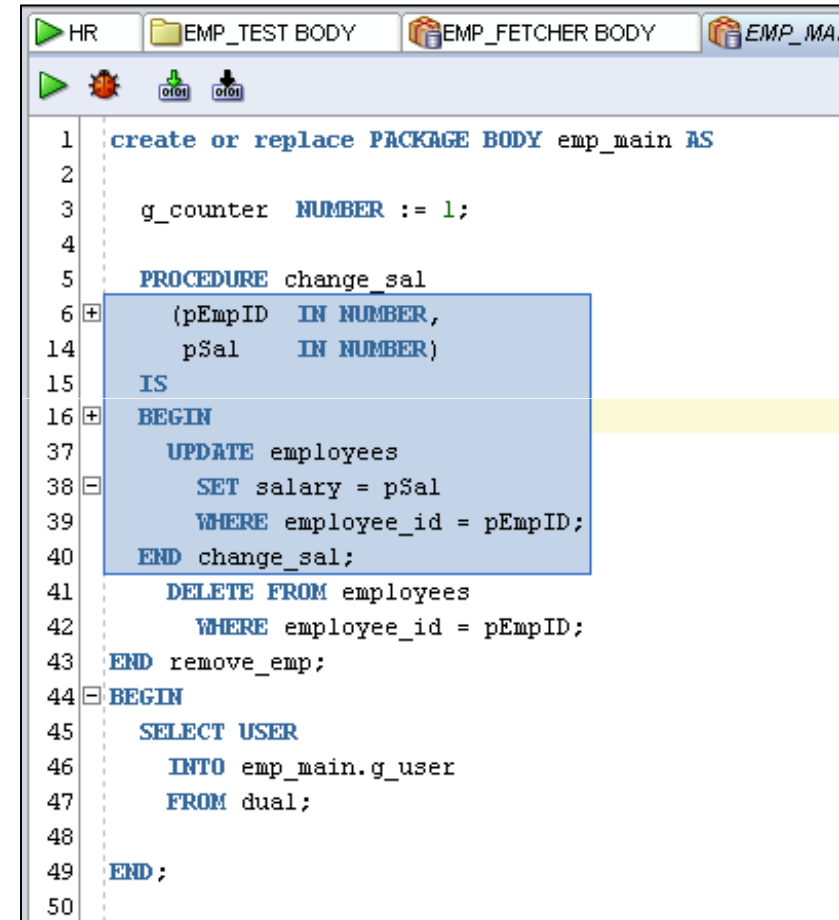
- PL/SQL Unit Testing
- Data Modeler Viewer
- Migration support for IBM DB2 UDB and Teradata
- Updated Data Grids
  - Manage columns, filter on data
- New SQL Worksheet
  - Dockable dbms\_output, multiple worksheets
- Increased Connections navigator support for
  - Jobs, Editions (for 11gR2), XML DB Repository
- Updated display editors
  - PL/SQL edit mode, subpartitions
- Version Control support for Serena Dimensions, Perforce
- Updated filtering mechanism
  - Schema level, generated objects

# Working with PL/SQL



# Creating and Editing PL/SQL

- Code editor
  - Syntax highlighting
  - Code formatter
  - Code insight (auto complete)
  - Code folding
  - Query Builder
- Code snippets
  - Drag and drop code snippets
  - Add and customize snippets
- Code templates

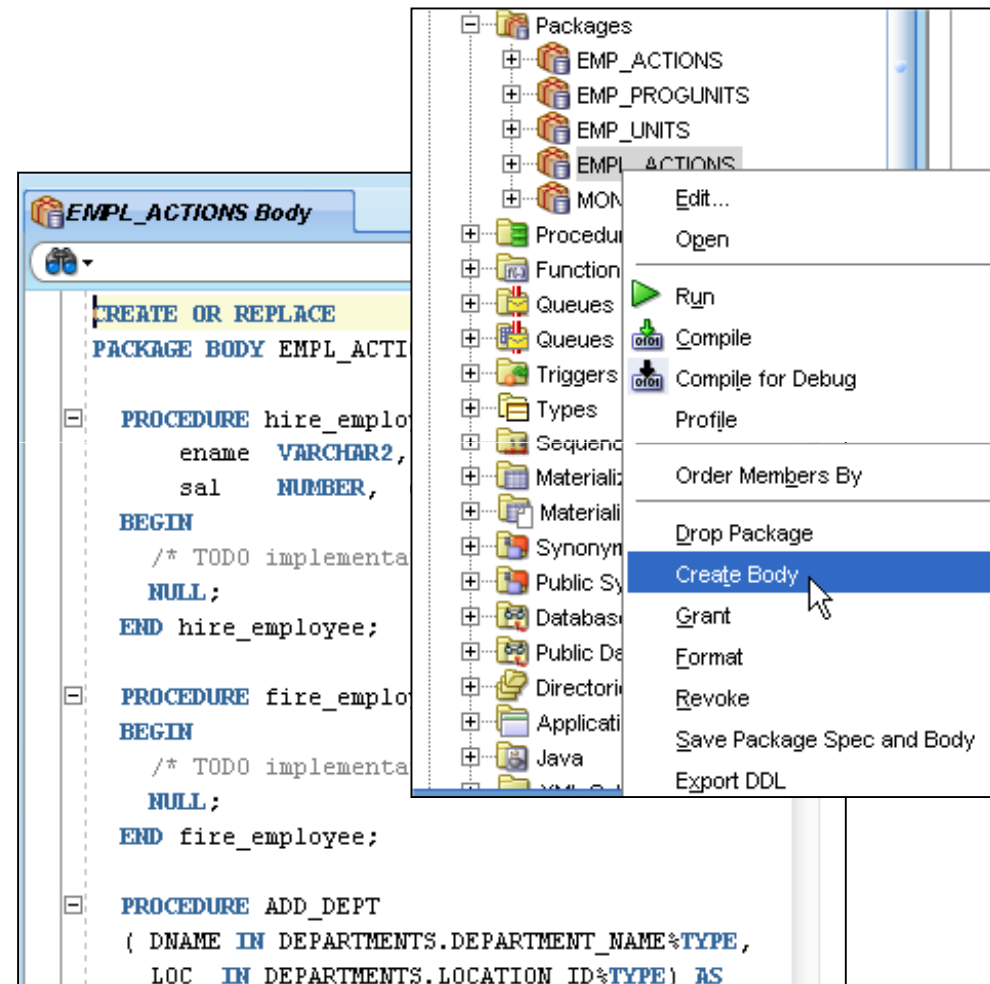


The screenshot shows the Oracle SQL Developer code editor with a file named 'EMP\_TEST BODY'. The code is written in PL/SQL and includes syntax highlighting. A blue selection box highlights the 'change\_sal' procedure. The code is as follows:

```
1  create or replace PACKAGE BODY emp_main AS
2
3      g_counter  NUMBER := 1;
4
5      PROCEDURE change_sal
6      (pEmpID  IN NUMBER,
14     pSal     IN NUMBER)
15  IS
16  BEGIN
37      UPDATE employees
38      SET salary = pSal
39      WHERE employee_id = pEmpID;
40  END change_sal;
41
42      DELETE FROM employees
43      WHERE employee_id = pEmpID;
44  END remove_emp;
45  BEGIN
46      SELECT USER
47      INTO emp_main.g_user
48      FROM dual;
49  END;
50
```

# Preparing Code Skeletons with Dialogs

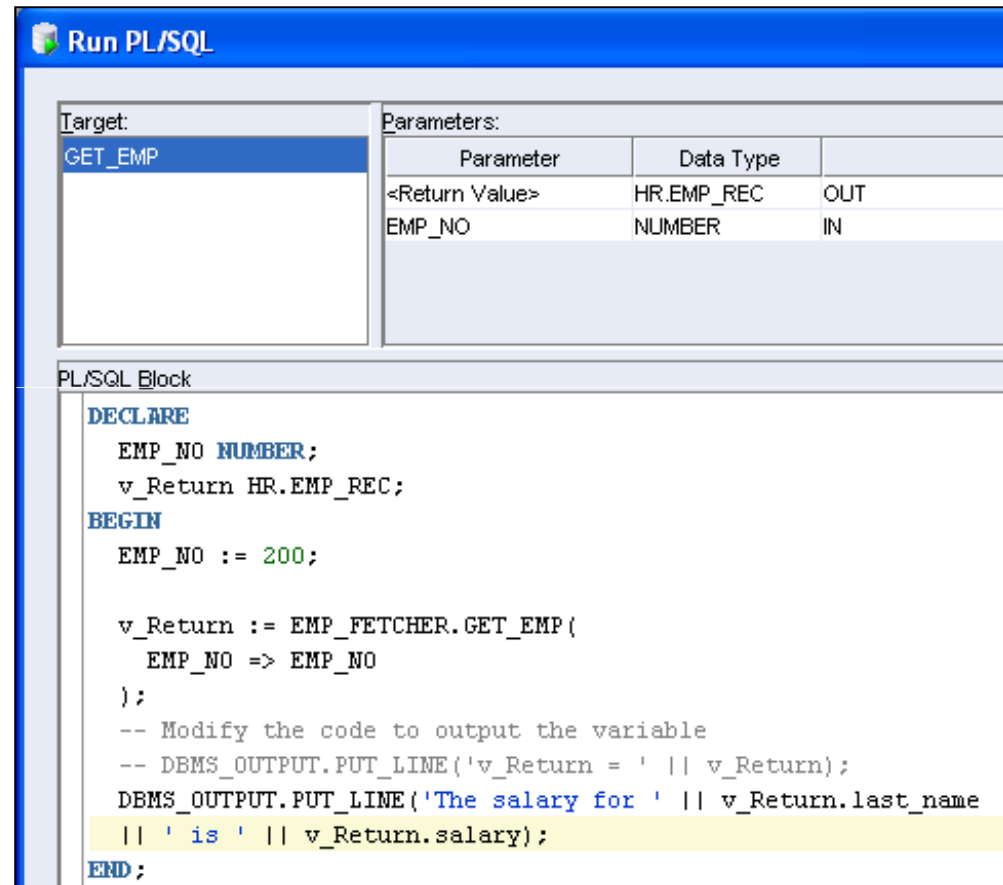
- Use dialog to create
  - Procedures & Functions
  - Triggers
  - Package spec & body
- Create Trigger
  - Table
  - View
  - Schema
  - Database





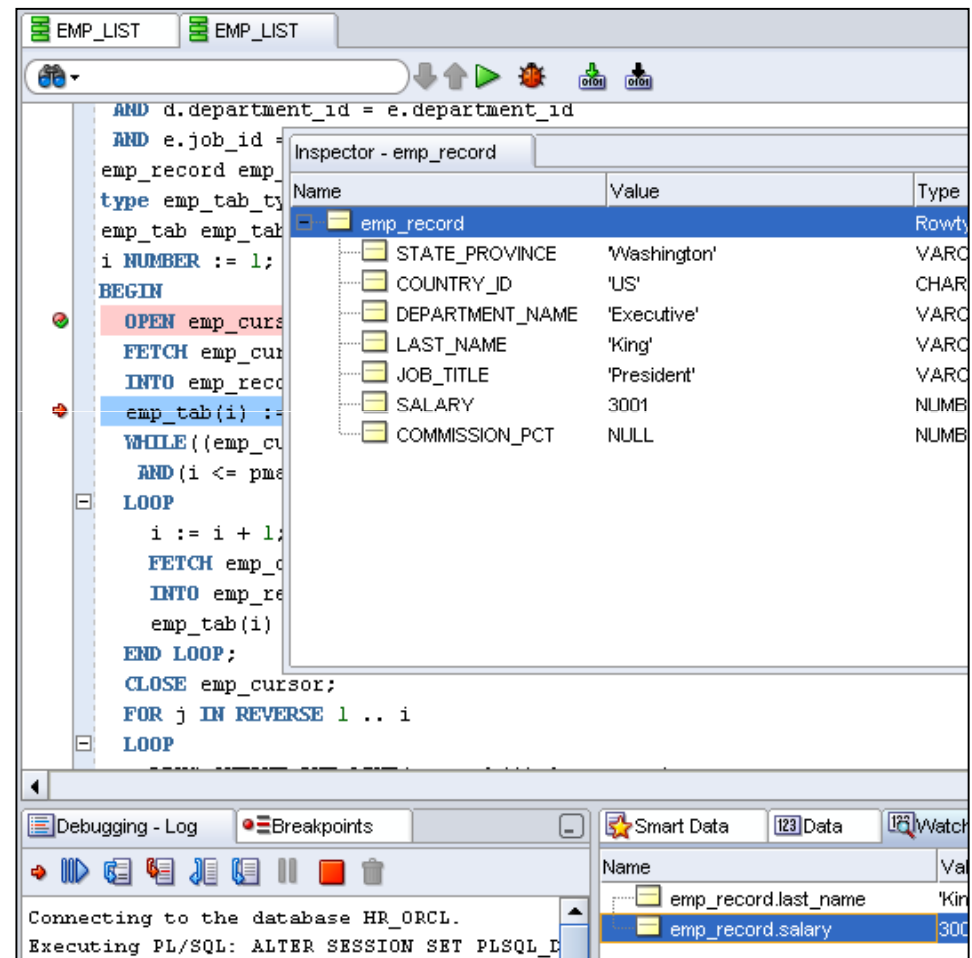
# Compiling and Running PL/SQL

- Compiling
  - Inline error reporting
- Run procedures, functions, and packages
  - Use DBMS\_OUTPUT
  - Function return values
  - OUT parameters
- Run PL/SQL dialog
  - Specifies run targets
  - Shows parameter detail
  - Generates editable PL/SQL block
    - For parameter values
    - For output parameters
    - Works with records



# Debugging PL/SQL

- Set breakpoints
  - Configure conditions
- Compile for Debug
- Control program execution (Step into, over...)
- Run to Cursor
- Inspect and modify variables
- Review
  - Smart data
  - Data
- Watches expressions
- View debug log



# Remote Debugging

**Problem:** Test a procedure being executed in a separate application

- In SQL Developer

- Select Remote Debug
- Set up remote debug detail; machine, port
- Browse to procedure
- Set a breakpoint

- In remote session

- execute DBMS\_DEBUG\_JDWP.CONNECT\_TCP ('127.0.0.1', 4000)
- Execute procedure

- In SQL Developer

- Debug as normal

```
SQL> show user
USER is "HR"
SQL> exec DBMS_DEBUG_JDWP.CONNECT_TCP( '127.0.0.1', 4000 );

PL/SQL procedure successfully completed.

SQL> DECLARE
2     hdate          varchar2(20);
3     hname          varchar2(20);
4 BEGIN
5     Get_emp_name(100, hdate, hname);
6 END;
7 /
```

Use the environment variable

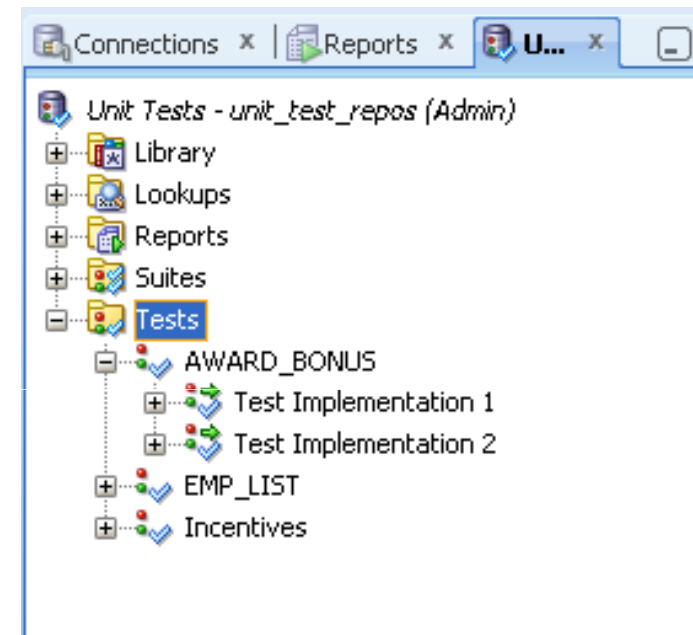
SET ORA\_DEBUG\_JDWP=host=127.0.0.1;port=4000

# Working with PL/SQL Unit Testing



# Unit Testing – Overview

- Tests
- Suites
- Reports
- Library
- Static and dynamic lookups
- Multi user repository based
- Code coverage
- Command line use
- Target any database



# Unit Testing - Tests

- Input/Return
  - Static or Dynamic Values
- Startups/TearDown
  - Table Copy/Restore
  - Row Copy/Restore
  - Custom
- Code coverage
- Success or failure testing
- Validation
  - Custom

The screenshot shows the Oracle Unit Testing interface for a procedure named `PROCEDURE KLRICE.HI(P_IN IN VARCHAR2)`. The interface includes tabs for 'Details' and 'Results'. Under 'Details', there are dropdown menus for 'Startup Process' and 'Teardown Process', both set to 'None'. A checkbox for 'Gather Code Coverage Statistics' is checked. Below this is a section for 'Test Implementation 1' which contains a table of parameters and a dynamic value query.

| Parameter | Datatype | in/out |
|-----------|----------|--------|
| P_IN      | VARCHAR2 | IN     |

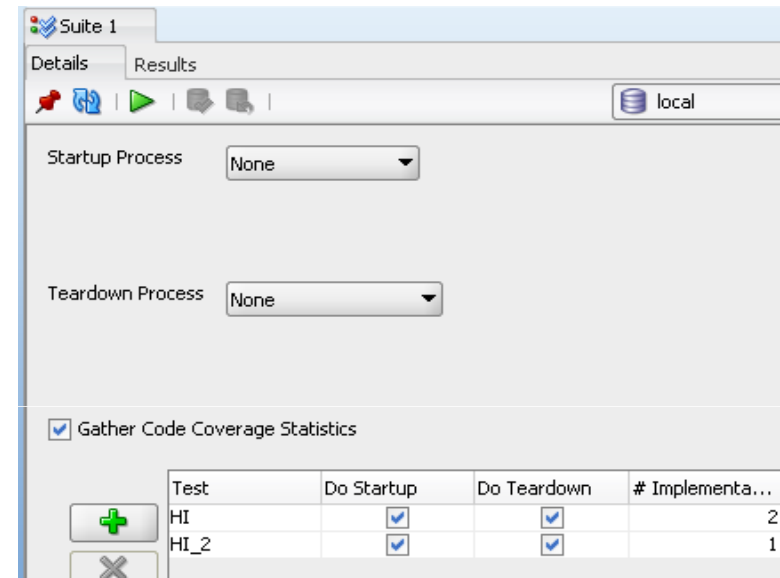
Dynamic Value Query: `select 1 as P_IN from dual union all select 2 as P_IN from dual union all select 3 as P_IN from dual union all select 4 as P_IN from dual union all select 5 as P_IN from dual`

Expected Result: `Success` (dropdown), `6502` (text input), `Enter expected error number` (placeholder text)

Buttons: `+` (Add), `-` (Remove), `Process Validation`

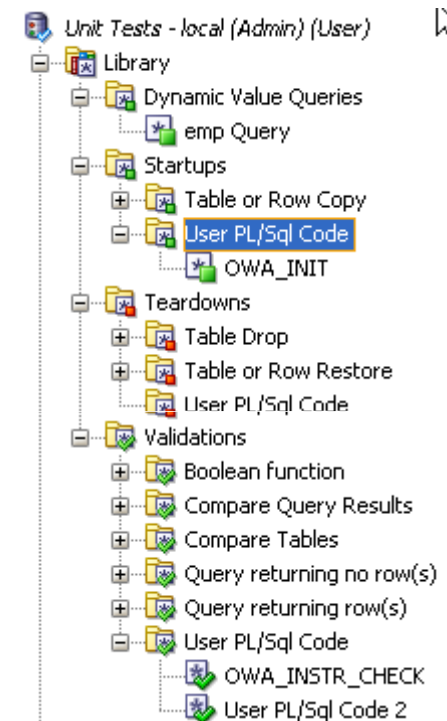
# Unit Testing - Suites

- Startups/TearDown
  - Table Copy/Restore
  - Row Copy/Restore
  - Custom
- Code Coverage
- Tests are run sequentially
- Startup and teardowns for tests can be turned off



# Unit Testing - Library

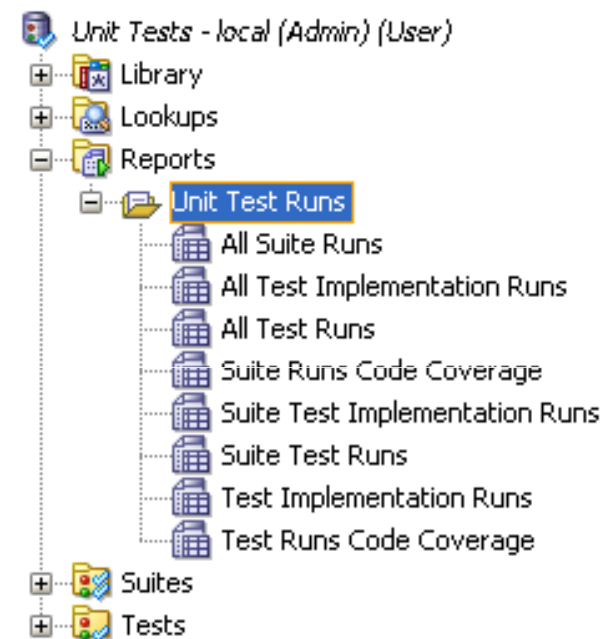
- Stores reusable items
  - Dynamic Values
  - Startups
  - Teardowns
  - Validations
- Referenced or copied to local tests





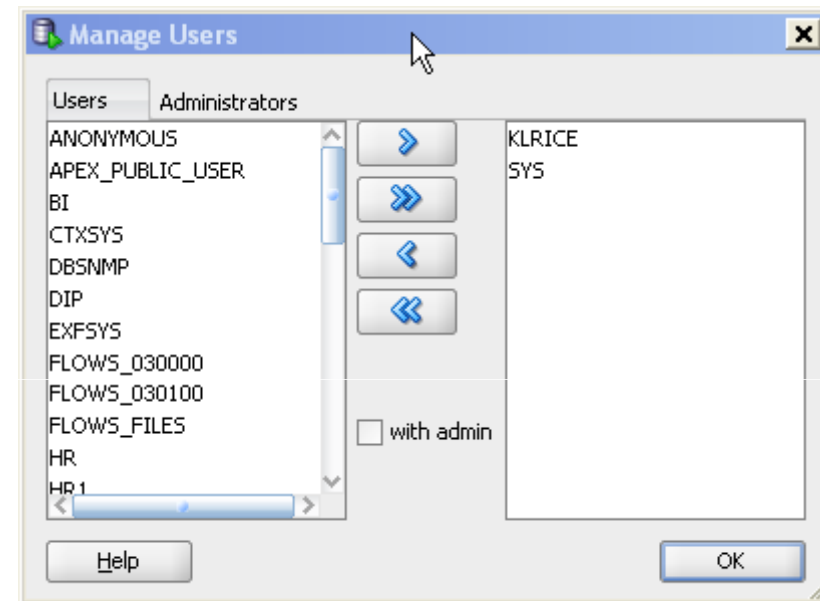
# Unit Testing - Reports

- Includes Standard reports
  - Suites
  - Tests
  - Code Coverage
- Reports against the repository
- Users can query the repository directly



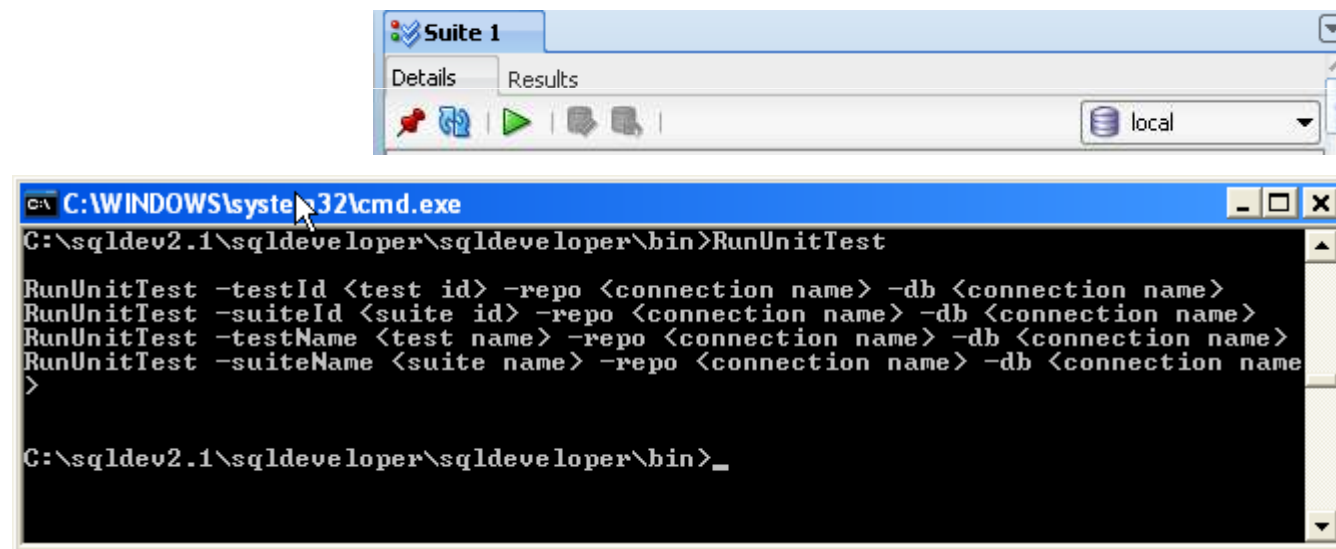
## Unit Testing – Multi User

- Use database users
- Control
  - admin vs. user
- Managed with roles



## Unit Testing – Running Suites/Tests

- Inside SQL Developer simply change the Combo List
- Command line by passing arguments
- Results are stored in the repository



The screenshot shows two overlapping windows. The top window is titled 'Suite 1' and has tabs for 'Details' and 'Results'. It features a toolbar with icons for a pin, a document, a play button, a speech bubble, and a printer. A dropdown menu on the right shows 'local'. The bottom window is a Windows command prompt titled 'C:\WINDOWS\system32\cmd.exe'. The command prompt shows the directory 'C:\sqldev2.1\sqldeveloper\sqldeveloper\bin' and the command 'RunUnitTest'. Below this, four lines of command syntax are displayed: 'RunUnitTest -testId <test id> -repo <connection name> -db <connection name>', 'RunUnitTest -suiteId <suite id> -repo <connection name> -db <connection name>', 'RunUnitTest -testName <test name> -repo <connection name> -db <connection name>', and 'RunUnitTest -suiteName <suite name> -repo <connection name> -db <connection name>'. The prompt ends with a greater-than sign '>' and a cursor.

```
C:\WINDOWS\system32\cmd.exe
C:\sqldev2.1\sqldeveloper\sqldeveloper\bin>RunUnitTest

RunUnitTest -testId <test id> -repo <connection name> -db <connection name>
RunUnitTest -suiteId <suite id> -repo <connection name> -db <connection name>
RunUnitTest -testName <test name> -repo <connection name> -db <connection name>
RunUnitTest -suiteName <suite name> -repo <connection name> -db <connection name>
>

C:\sqldev2.1\sqldeveloper\sqldeveloper\bin>_
```



## **Refactor, Review, Search, Tune and Monitor**

SQL Developer provides a PL/SQL related utilities

- PL/SQL Hierarchical Profiler
- Extended Search using PL/SQL
- SQL Monitor
- SQL Developer PL/SQL Reports
- General refactoring
- APEX refactoring

# Hierarchical Profiler

The screenshot displays the Oracle SQL Developer interface with the Hierarchical Profiler for the procedure `HR1.EMP_LIST`. The left pane shows the database schema tree with `hr1_11g_orcl` selected. The main pane shows the execution profiles for the procedure.

**Execution Profiles Table:**

| RUNID | Timestamp        | Comment | Total Elapsed Time (mksec) |
|-------|------------------|---------|----------------------------|
| 1     | 14-Farvardin ... | (null)  | 2348                       |
| 2     | 14-Farvardin ... | (null)  | 2872                       |
| 3     | 28-Farvardin ... | (null)  | 675                        |
| 4     | 09-Khordad ...   | (null)  | 2346                       |

**Hierarchy Table:**

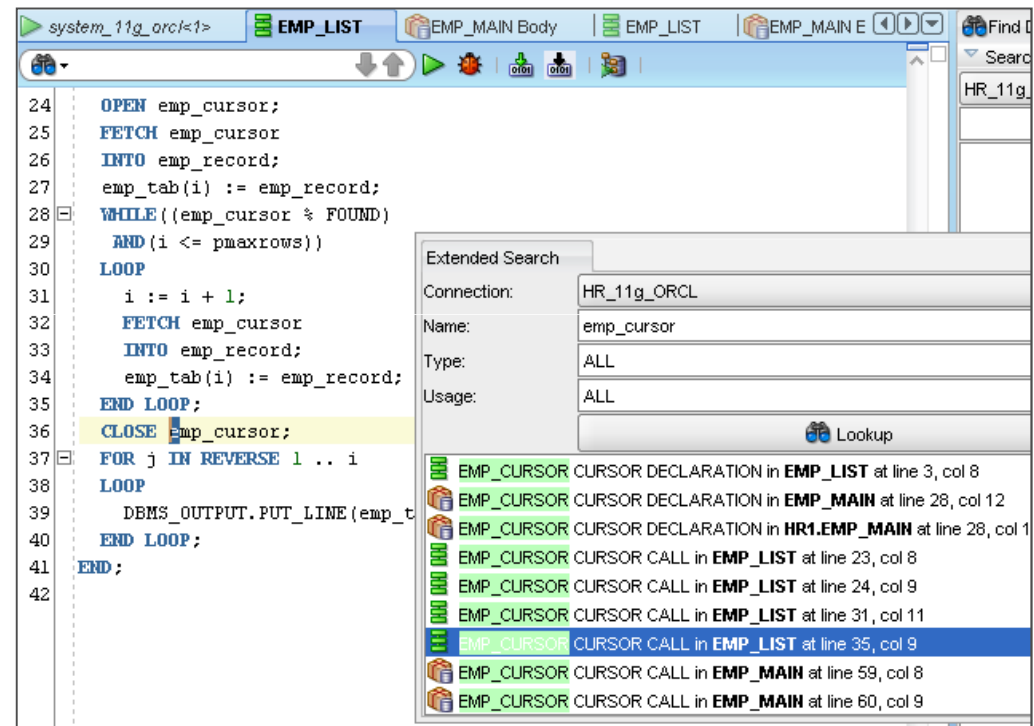
| Callee                               | Elapsed (mksec) | Aggregated | Calls# |
|--------------------------------------|-----------------|------------|--------|
| HR1.EMP_LIST                         | 171             | 2295       | 1      |
| HR1.EMP_LIST.EMP_LIST.EMP_CURSOR     | 18              | 89         | 1      |
| HR1.EMP_LIST.__static_sql_exec_line7 | 71              | 71         | 1      |
| SYS.DBMS_OUTPUT.PUT_LINE             | 11              | 11         | 5      |
| SYS.DBMS_OUTPUT.__pkg_init           | 11              | 11         | 1      |
| HR1.EMP_LIST.__sql_fetch_line25      | 1922            | 1922       | 1      |
| HR1.EMP_LIST.__sql_fetch_line30      | 91              | 91         | 4      |
| SYS.DBMS_HPROF.STOP_PROFILING        | 0               | 0          | 1      |

SQL History

hr1\_11g\_orcl | HR1 | EMP\_LIST Editing

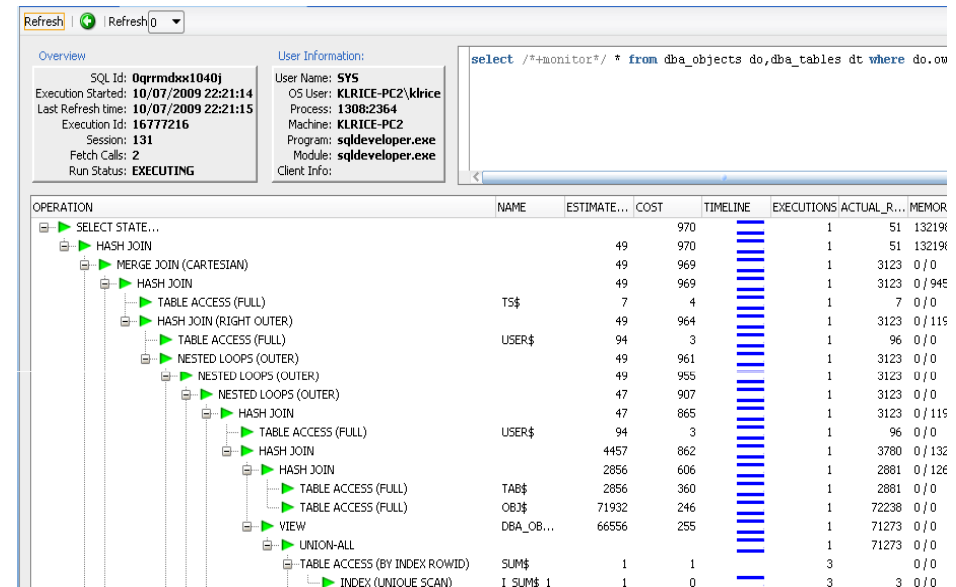
# Searching

- Find (and replace) in all editors
- Find DB Objects
  - Across schemas
  - Navigate to object
- Extended search Across schemas
  - For object types
  - For usages
  - PLScope support (Oracle Database 11g)



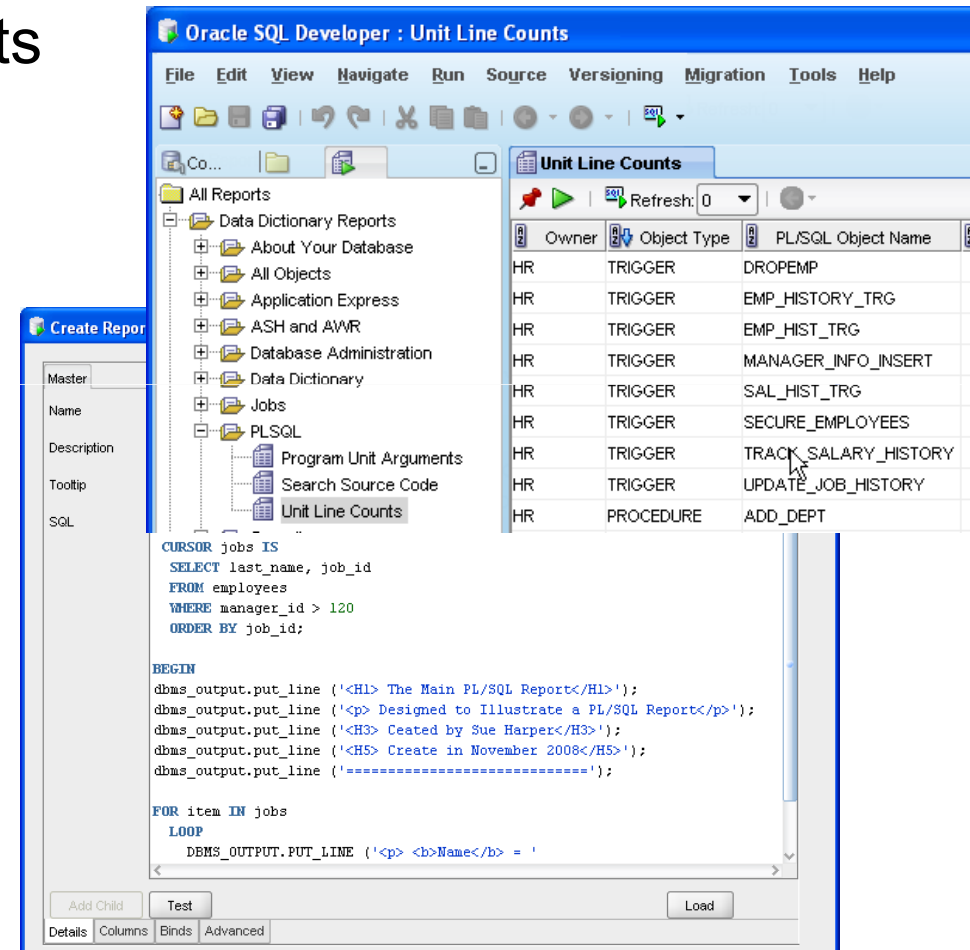
# Real-Time SQL Monitoring

- Real time view of SQL
- Use /\*+MONITOR\*/
- Drill to view details
- Visual indicators for current step
- Queries over 5 seconds monitored
- DBMS\_SQLTUNE.REPORT\_SQL\_MONITOR



# SQL Monitor and PL/SQL Reports

- Shipped PL/SQL Reports
  - SQL Monitor
  - Search Source Code
  - Program Unit Arguments
- User Defined Reports
  - Using plsql-dbms\_output
  - Formatting code







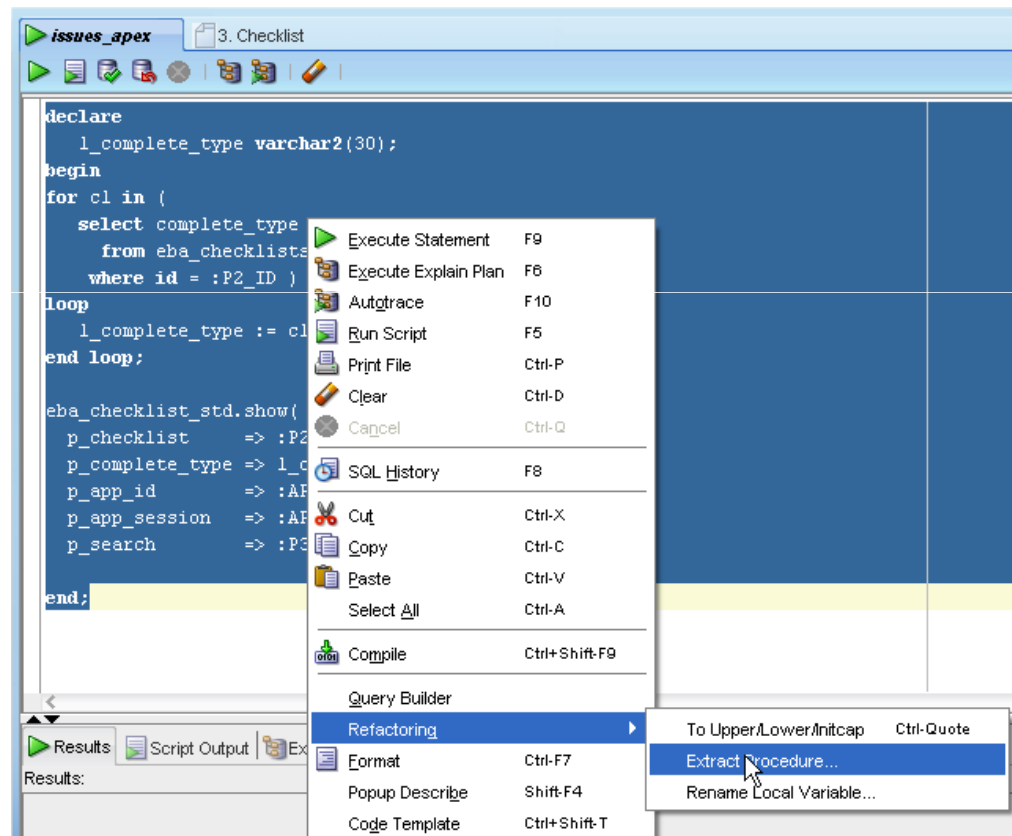
## Refactoring

- Extract a procedure
- Surround blocks with
  - For
  - While
  - Begin block
- Variable renames
- Extract anonymous PL/SQL blocks from APEX apps

# Integrating with Oracle APEX

## Providing integration points to Oracle APEX

- Remote debugging
- Tuning SQL
- Refactoring PL/SQL code





## Finding More Detail

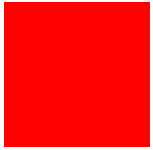
[www.oracle.com/technology/products/database/sql\\_developer](http://www.oracle.com/technology/products/database/sql_developer)

- SQL Developer on OTN
  - White papers, Oracle by Example (OBE) and online demos
  - Team Blogs: [Blogs, Magazine Articles & Podcasts](#)
  - [www.oracle.com/technology/products/database/sql\\_developer](http://www.oracle.com/technology/products/database/sql_developer)
- SQL Developer Exchange
  - Share reports, snippets, code, and add feature requests
  - <http://sqldeveloper.oracle.com>
- Forums
  - SQL Developer  
[forums.oracle.com/forums/forum.jspa?forumID=260](http://forums.oracle.com/forums/forum.jspa?forumID=260)



## Summary

- PL/SQL
  - Creating, editing, compiling and debugging
- Unit Testing
  - Creating, Running, Reporting
  - Batch processing
- Creating SQL Developer extensions
- Real-Time SQL Monitoring
  - Watch SQL as it runs
- Refactoring
  - Convert APEX anonymous blocks into a Package



**For More Information**

**search.oracle.com**

SQL Developer



**or**

**[www.oracle.com/technology/products/database/sql\\_developer](http://www.oracle.com/technology/products/database/sql_developer)**



ORACLE®



**ORACLE IS THE INFORMATION COMPANY**