Oracle Database: Develop PL/SQL Program Units Ed 2

Duration: 3 Days

What you will learn

This course is designed for developers with basic PL/SQL and SQL language skills. Students learn to develop, execute, and manage PL/SQL stored program units such as procedures, functions, packages, and database triggers. Students also learn to manage, PL/SQL subprograms and triggers.

Learn To:

Create, and execute stored procedures and functions.
Design and use PL/SQL packages.
Create overloaded package subprograms for more flexibility.
Utilize Oracle-Supplied Packages in Application Development.
Create triggers to solve business challenges.
Build and execute SQL statements dynamically.

Benefits To You

Students are introduced to the utilization of some of the Oracle-supplied packages. Additionally students learn to use Dynamic SQL, understand design considerations when coding using PL/SQL, understand and influence the PL/SQL compiler, and manage dependencies. In this course, students learn and use Oracle SQL Developer as the main environment tool to develop these program units. SQL*Plus is introduced as optional tools. Demonstrations and hands-on practice reinforce the fundamental concepts.

Audience

Database Administrators
Forms Developer
PL/SQL Developer
Portal Developer
System Analysts
Technical Consultant

Related Training

Required Prerequisites

Familiarity with data processing concepts and technique

Familiarity with programming languages

Suggested Prerequisites
Familiarity with data processing concepts and techniques

Familiarity with programming languages

Course Objectives
Use conditional PL/SQL compilation and obfuscate (hide) code

Create triggers to solve business challenges

Manage dependencies between PL/SQL subprograms

Design PL/SQL code for predefined data types, local subprograms, additional pragmas and standardized constants and exceptions

Use the compiler warnings infrastructure

Create, use, and debug stored procedures and functions

Design and use PL/SQL packages to group and contain related constructs

Create overloaded package subprograms for more flexibility

Use the Oracle supplied PL/SQL packages to generate screen output, file output, and mail output

Write dynamic SQL for more coding flexibility

Course Topics

Introduction
Course Objectives, Course Agenda and Appendixes Used in this Course
Full Human Resources (HR) Schema
Online Oracle Database 12c SQL and PL/SQL documentation
PL/SQL development environments available in this course
Using the SQL Worksheet
Executing SQL Statements
Working With Script Files
Creating and Executing Anonymous Blocks

Working with Oracle Database Exadata Express Cloud Service
Overview of Oracle Database Exadata Express Cloud Service
Accessing Cloud Database using SQL Workshop
Connecting to Exadata Express Database using Database Clients
Using SQL Developer to work with Exadata Express Database
Using SQLcl to work with Exadata Express Database

Copyright © 2013, Oracle. All rights reserved.

Using SQL*Plus to work with Exadata Express Database

Creating Stored Procedures
PL/SQL blocks and subprograms
Uses and benefits of procedures
Working with procedures
Using formal and actual parameters
Identify the available parameter-passing modes
Passing parameters using the positional, named, or combination techniques
Handling exceptions in procedures
Viewing the procedure information

Creating Functions and Debugging Subprograms
Creating Stored Functions
The Difference Between Procedures and Functions
Working with Functions
Identifying the Advantages of Using Stored Functions in SQL Statements
Using User-Defined Functions in SQL Statements
Using a PL/SQL Function in the SQL WITH Clause
Defining and executing PL/SQL functions in SQL statements
Restrictions When Calling Functions from SQL statements

Creating Packages
Using PL/SQL Packages
Components of a PL/SQL Package
Visibility of a Package’s Components
Developing a PL/SQL Package
Creating the Package Specification and Package Body
Invoking the Package Constructs
Creating and Using Bodiless Packages
Removing a Package

Working With Packages
Overloading Subprograms
Using Forward Declarations to Solve Illegal Procedure Reference
Initializing Packages
Using Package Functions in SQL and Restrictions
Controlling Side Effects of PL/SQL Subprograms
Persistent State of Packages
Persistent State of Package Variables and Cursors
Using PL/SQL Tables of Records in Packages

Using Oracle-Supplied Packages in Application Development
Using Oracle-Supplied Packages
Examples of Some of the Oracle-Supplied Packages
Working of DBMS_OUTPUT Package
Using the UTL_FILE Package to Interact With Operating System Files
Using the UTL_MAIL Package

Using Dynamic SQL
Introduction to Dynamic SQL
The Execution Flow of SQL

Copyright © 2013, Oracle. All rights reserved.

Working With Dynamic SQL
When Do You Need Dynamic SQL?
Using Native Dynamic SQL (NDS)
Dynamic SQL with mock up application
Using BULK COLLECT and FORALL
Dynamic SQL using DBMS_SQL package

Creating Triggers
Different types of triggers
Database triggers and their use
Creating database triggers
Database trigger firing rules
Removing database triggers

Creating Compound, DDL, and Event Database Triggers
Compound triggers
Mutating tables
Creating triggers on DDL statements
Creating triggers on system events
Displaying information about triggers

Design Considerations for PL/SQL Code
Standardizing constants with a constant package
Standardizing exceptions with an exception package
Writing PL/SQL code that uses local subprograms
Grant Roles to PL/SQL Packages and Standalone Stored Subprograms
Using the NOCOPY compiler hint to pass parameters by reference
Using the PARALLEL ENABLE hint for optimization
Using the AUTONOMOUS TRANSACTION pragma
Describing the differences between invoker rights and definer rights

Using PL/SQL compiler
Using the PL/SQL Compiler with initialisation parameters
Using the PL/SQL Compile Time Warnings
Viewing the Current Setting of PLSQL_WARNINGS
Viewing the Compiler Warnings
Guidelines for using PLSQL_WARNINGS
Conditional Compilation

Managing Dependencies
Dependent and referenced objects
Tracking procedural dependencies with dictionary views
Predicting the effect of changing a database object
Managing local and remote procedural dependencies

Copyright © 2013, Oracle. All rights reserved.