



Oracle Database 10g : Program with PL/SQL Course

36 Contact Hours

Course Overview

This course introduces students to PL/SQL and helps them understand the benefits of this powerful programming language. In the class, students learn to create PL/SQL blocks of application code that can be shared by multiple forms, reports, and data management applications.

What you will learn

Students learn to create anonymous PL/SQL blocks, stored procedures, and functions. They learn about declaring variables and trapping exceptions. Students will also learn to develop stored procedures, functions, packages and database triggers. Students will learn to manage PL/SQL program units manage dependencies, manipulate large objects, and use some of the Oracle-supplied packages. Students use iSQL*Plus to develop these program units. Demonstrations and hands-on practice reinforce the fundamental concepts. This is one of the three exams required to achieve the OCP Developer Certification, and one of the three exams required to achieve the OCP Developer Certification.

Audience :

PL/SQL Programmer, Database Designers, Forms Developer, Technical Consultant

Prerequisites :

Oracle Database 10g: Introduction to SQL

Course Objectives:

Design PL/SQL anonymous blocks that execute efficiently

Write PL/SQL code to interface with the database

Describe the features and syntax of PL/SQL



Use PL/SQL programming constructs and conditionally control code flow (loops, control structures, and explicit cursors)

Handle runtime errors

Create simple procedures and functions

Design PL/SQL packages to group and contain related constructs

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

Schedule PL/SQL jobs to run independently

Write dynamic SQL for more coding flexibility

Create triggers to solve business challenges

Manage dependencies between PL/SQL subprograms

Course Outline

Introduction to PL/SQL

What is PL/SQL

PL/SQL Environment

Benefits of PL/SQL

Overview of the Types of PL/SQL blocks

Create and Execute a Simple Anonymous Block

Generate Output from a PL/SQL Block

iSQL*Plus as PL/SQL Programming Environment

Declaring PL/SQL Identifiers

Identify the Different Types of Identifiers in a PL/SQL subprogram

Use the Declarative Section to Define Identifiers



List the Uses for Variables

Store Data in Variables

Declare PL/SQL Variables

Writing Executable Statements

Describe Basic Block Syntax Guidelines

Use Literals in PL/SQL

Customize Identifier Assignments with SQL Functions

Use Nested Blocks as Statements

Reference an Identifier Value in a Nested Block

Qualify an Identifier with a Label

Use Operators in PL/SQL

Use Proper PL/SQL Block Syntax and Guidelines

Interacting with the Oracle Server

Identify the SQL Statements You Can Use in PL/SQL

Include SELECT Statements in PL/SQL

Retrieve Data in PL/SQL with the SELECT Statement

Avoid Errors by Using Naming Conventions When Using Retrieval and DML Statements

Manipulate Data in the Server Using PL/SQL

The SQL Cursor concept

Use SQL Cursor Attributes to Obtain Feedback on DML

Save and Discard Transactions



Writing Control Structures

Control PL/SQL Flow of Execution

Conditional processing Using IF Statements

Conditional Processing CASE Statements

Handle Nulls to Avoid Common Mistakes

Build Boolean Conditions with Logical Operators

Use Iterative Control with Looping Statements

Working with Composite Data Types

Learn the Composite Data Types of PL/SQL Records and Tables

Use PL/SQL Records to Hold Multiple Values of Different Types

Inserting and Updating with PL/SQL Records

Use INDEX BY Tables to Hold Multiple Values of the Same Data Type

Using Explicit Cursors

Cursor FOR Loops Using Subqueries

Increase the Flexibility of Cursors By Using Parameters

Use the FOR UPDATE Clause to Lock Rows

Use the WHERE CURRENT Clause to Reference the Current Row

Use Explicit Cursors to Process Rows

Explicit Cursor Attributes

Cursors and Records



Handling Exceptions

Handling Exceptions with PL/SQL

Predefined Exceptions

Trapping Nonpredefined Oracle Server Errors

Functions that Return Information on Encountered Exceptions

Trapping User-Defined Exceptions

Propagate Exceptions

Use The RAISE_APPLICATION_ERROR Procedure To Report Errors To Applications

Creating Stored Procedures

Describe PL/SQL blocks and subprograms

Describe the uses of procedures

Create procedures

Differentiate between formal and actual parameters

List the features of different parameter modes

Create procedures with parameters and invoke a procedure

Handle exceptions in procedures

View source code in the data dictionary

Creating Stored Functions

Describe stored functions

List the CREATE OR REPLACE FUNCTION syntax

Identify the steps to create a stored function

Create a stored function in iSQL*Plus and execute a stored function

Identify the advantages of using stored functions in SQL statements

ENGO SOFT



Identify the restrictions of calling functions from SQL statements

Describe how procedures and functions differ

Creating Packages

List the benefits of using PL/SQL packages

Differentiate between a package specification and a package body

Create packages

Include public and private constructs in a package

Call public and private constructs in a package

Remove packages

Using More Package Concepts

Overload procedure and function definitions

Use forward declarations

Create a one-time package initialization block

Follow the persistent state of constructs in packages

Use PL/SQL tables and records in packages

Wrap code to hide the source

Utilizing Oracle Supplied Packages in Application Development

List the various uses for the Oracle supplied packages

Reuse pre-packaged code to complete various tasks from developer to DBA purposes

Use the DESCRIBE command to view the package specifications and overloading

Describe how DBMS_OUTPUT works

Use UTL_FILE to direct output to operating system files



Use the HTP package to generate a simple web page

Describe the main features of UTL_MAIL

Call the DBMS_SCHEDULER package to schedule PL/SQL code to run

Dynamic SQL and Metadata

Describe using native dynamic SQL

List the execution flow of SQL

Write dynamic SQL using the EXECUTE IMMEDIATE syntax

Write dynamic SQL with the DBMS_SQL package

Generate DDL from metadata using the DBMS_METADATA package

Design Considerations for PL/SQL Code

Standardize constants with a constant package

Standardize exceptions with an exception package

Write PL/SQL code that uses local subprograms

Use the NOCOPY compiler hint to pass parameters by reference

Use the PARALLEL ENABLE hint for optimization

Use the AUTONOMOUS TRANSACTION pragma to run independent transactions within a single transaction

Set the AUTHID directive to execute programs with the privileges of the calling user instead of the creating user

Use bulk binding for multi-row operations

Managing Dependencies

Describe dependent and referenced objects

Track procedural dependencies with dictionary views



Predict the effect of changing a database object upon stored procedures and functions

Manage local and remote procedural dependencies

Manipulating Large Objects

Describe a LOB object

Create and maintain LOB data types

Differentiate between internal and external LOBs

Use the DBMS_LOB PL/SQL package to control LOBs

Describe the use of temporary LOBs

Creating Triggers

Describe different types of triggers

Describe database triggers and their use

Create database triggers

Describe database trigger firing rules

Remove database triggers

Applications for Triggers

Create database and system event triggers

Create triggers on DDL statements

Use the CALL statement in triggers to invoke procedures

Explain the rules for reading and writing to tables with triggers

Describe business application scenarios for implementing with triggers

Manage trigger code



Understanding and Influencing the PL/SQL Compiler

Describe native compilation and interpreted compilation

List the features of native compilation

Switch between native and interpreted compilation for compiled PL/SQL code

Set the parameters to control aspects of PL/SQL compilation

Write a query to retrieve information from the dictionary views on how the PL/SQL code is compiled

Explain the compiler warning mechanism

List the steps to use the compiler warnings

Use DBMS_WARNING to implement compiler warnings

ENGO SOFT