

Oracle Database 11g: Program with PL/SQL

This course starts with an introduction to PL/SQL and proceeds to list the benefits of this powerful programming language. Participants are made aware of how to create PL/SQL blocks of application code that can be shared by multiple forms, reports, and data management applications. In addition, creation of anonymous PL/SQL blocks as well as stored procedures and functions are covered in this course.

Participants enhance their developer skills by learning to develop, execute, and manage PLSQL stored program units such as procedures, functions, packages, and database triggers. Understanding the basic functionality of how to debug functions and procedures using the SQL Developer Debugger gives way to refined lines of code. Participants also learn to manage PL/SQL subprograms, triggers, declaring identifiers, and trapping exceptions. The utilization of some of the Oracle-supplied packages is also in the course. Additionally participants learn to use Dynamic SQL, understand design considerations when coding using PL/SQL, understand and influence the PL/SQL compiler, and manage dependencies.

This course is a bundle of Oracle Database: PL/SQL Fundamentals and Oracle Database: Develop PL/SQL Program Units courses.

Students use Oracle SQL Developer to develop these program units. SQL*Plus and JDeveloper are introduced as optional tools.

This is appropriate for a 10g and 11g audience. There are minor changes between 10g and 11g features.

Détails

- **Code** : O11GPPLSQL
 - **Durée** : 5 jours (35 heures)
- Public**
- Administrateurs de data warehouse
 - Analystes
 - Analystes système
 - Database Administrators
 - Développeurs
 - Développeurs Forms
 - Développeurs PL/SQL
- Pré-requis**
- Oracle Database 11g: SQL Fundamentals II Release 2

Objectifs

- Perform tablespace point-in-time recovery
- Describe the Oracle Database architecture components related to backup and recovery operations
- Describe additional high availability features such as Oracle Data Guard and Oracle Secure Backup
- Plan effective backup and recovery procedures
- Describe Oracle Database backup methods and recovery operations that can be used to resolve database failure

Programme

Introduction

- Course Objectives
- Course Agenda
- Describing the Human Resources (HR) Schema
- PL/SQL development environments Available in this course
- Introduction to SQL Developer

Introduction to PL/SQL

- PL/SQL Overview
- Benefits of PL/SQL Subprograms
- Overview of the Types of PL/SQL blocks
- Creating and Executing a Simple Anonymous Block
- Generating Output from a PL/SQL Block

Declaring PL/SQL Identifiers

- Different Types of Identifiers in a PL/SQL subprogram
- Using the Declarative Section to Define Identifiers

- Storing Data in Variables
- Scalar Data Types
- %TYPE Attribute
- Bind Variables
- Using Sequences in PL/SQL Expressions

Writing Executable Statements

- Describing Basic PL/SQL Block Syntax Guidelines
- Commenting Code
- SQL Functions in PL/SQL
- Data Type Conversion
- Nested Blocks
- Operators in PL/SQL

Interacting with the Oracle Server

- Including SELECT Statements in PL/SQL to Retrieve data
- Retrieving Data in PL/SQL with the SELECT Statement

- The SQL Cursor concept
- Avoiding Errors by Using Naming Conventions When Using Retrieval and DML Statements
- Manipulating Data in the Server Using PL/SQL
- Understanding the SQL Cursor concept
- Using SQL Cursor Attributes to Obtain Feedback on DML
- Saving and Discarding Transactions

Writing Control Structures

- Conditional processing Using IF Statements
- Conditional processing Using CASE Statements
- Simple Loop Statement
- While Loop Statement
- For Loop Statement
- The Continue Statement

Working with Composite Data Types

- Using PL/SQL Records to Hold Multiple Values of Different Types
- Using the %ROWTYPE Attribute
- Inserting and Updating with PL/SQL Records
- INDEX BY Tables to Hold Multiple Values of the Same Data Type
- INDEX BY Table Methods
- INDEX BY Table of Records

Using Explicit Cursors

- Understanding Explicit Cursors
- Declaring the Cursor
- Opening the Cursor
- Fetching data from the Cursor
- Closing the Cursor
- Cursor FOR loop
- The %NOTFOUND and %ROWCOUNT Attributes
- FOR UPDATE Clause and WHERE CURRENT Clause
- Handling Exceptions
- Understanding Exceptions

Handling Exceptions with PL/SQL

- Trapping Predefined Oracle Server Errors
- Trapping Non-Predefined Oracle Server Errors
- Trapping User-Defined Exceptions
- Propagate Exceptions
- RAISE_APPLICATION_ERROR Procedure

Creating Stored Procedures

- Creating a Modularize and Layered Subprogram Design
- Modularizing Development With PL/SQL Blocks
- Understanding the PL/SQL Execution Environment
- The Benefits of Using PL/SQL Subprograms
- The Differences Between Anonymous Blocks and Subprograms
- Creating, Calling, and Removing Stored Procedures Using the CREATE Command and SQL Developer
- Using Procedures Parameters and Parameters Modes
- Viewing Procedures Information Using the Data Dictionary Views and SQL Developer

Creating Stored Functions and Debugging

Subprograms

- Creating, Calling, and Removing a Stored Function Using the CREATE Command and SQL Developer
- Identifying the Advantages of Using Stored Functions in SQL Statements
- Identify the steps to create a stored function
- Using User-Defined Functions in SQL Statements
- Restrictions When Calling Functions from SQL statements
- Controlling Side Effects When Calling Functions from SQL Expressions
- Viewing Functions Information
- Debugging Functions and Procedures

Creating Packages

- Listing the Advantages of Packages
- Describing Packages
- The Components of a Package
- Developing a Package
- The Visibility of a Package's Components
- Creating the Package Specification and Body Using the SQL CREATE Statement and SQL Developer
- Invoking the Package Constructs
- Viewing the PL/SQL Source Code Using the Data Dictionary

Working With Packages

- Overloading Subprograms in PL/SQL
- Using the STANDARD Package
- Using Forward Declarations to Solve Illegal Procedure Reference
- Using Package Functions in SQL and Restrictions
- Persistent State of Packages
- Persistent State of a Package Cursor
- Controlling Side Effects of PL/SQL Subprograms
- 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
- How Does the DBMS_OUTPUT Package Work?
- Using the UTL_FILE Package to Interact With Operating System Files
- Using the UTL_MAIL Package
- Using the UTL_MAIL Subprograms

Using Dynamic SQL

- The Execution Flow of SQL
- What is Dynamic SQL?
- Declaring Cursor Variables
- Dynamically Executing a PL/SQL Block
- Using Native Dynamic SQL to Compile PL/SQL Code
- Using DBMS_SQL Package
- Using DBMS_SQL with a Parameterized DML Statement
- Dynamic SQL Functional Completeness

Design Considerations for PL/SQL Code

- Standardizing Constants and Exceptions
- Using Local Subprograms

- Using Autonomous Transactions
- Using the NOCOPY Compiler Hint
- Using the PARALLEL_ENABLE Hint
- Using the Cross-Session PL/SQL Function Result Cache
- Using the DETERMINISTIC Clause with Functions
- Using Bulk Binding to Improve Performance

Creating Triggers

- Working With Triggers
- Identifying the Trigger Event Types and Body
- Business Application Scenarios for Implementing Triggers
- Creating DML Triggers Using the CREATE TRIGGER Statement and SQL Developer
- Identifying the Trigger Event Types, Body, and Firing (Timing)
- Statement Level Triggers Versus Row Level Triggers
- Creating Instead of and Disabled Triggers
- Managing, Testing, and Removing Triggers

Creating Compound, DDL, and Event Database Triggers

- Working With Compound Triggers
- Identifying the Timing-Point Sections of a Table Compound Trigger
- Compound Trigger Structure for Tables and Views
- Using a Compound Trigger to Resolve the Mutating Table Error
- Comparing Database Triggers to Stored Procedures
- Creating Triggers on DDL Statements
- Creating Database-Event and System-Events Triggers
- System Privileges Required to Manage Triggers

Using the PL/SQL Compiler

- Using the PL/SQL Compiler
- Using the Initialization Parameters for PL/SQL Compilation

- Using the New PL/SQL Compile Time Warnings
- Overview of PL/SQL Compile Time Warnings for Subprograms
- The Benefits of Compiler Warnings
- The PL/SQL Compile Time Warning Messages Categories
- Setting the Warning Messages Levels: Using SQL Developer, PLSQL_WARNINGS Initialization Parameter, and the
- Viewing the Compiler Warnings: Using SQL Developer, SQL*Plus, or the Data Dictionary Views

Managing PL/SQL Code

- What Is Conditional Compilation and How Does it Work?
- Using Selection Directives
- Using Predefined and User-Defined Inquiry Directives
- The PLSQL_CCFLAGS Parameter and the Inquiry Directive
- Using Conditional Compilation Error Directives to Raise User-Defined Errors
- Using the DBMS_DB_VERSION Package
- Using DBMS_PREPROCESSOR Procedures to Print or Retrieve Source Text
- Obfuscating and Wrapping PL/SQL Code

Managing Dependencies

- Overview of Schema Object Dependencies
- Querying Direct Object Dependencies Using the USER_DEPENDENCIES View
- Querying an Object's Status
- Invalidation of Dependent Objects
- Displaying Direct and Indirect Dependencies
- Fine-Grained Dependency Management in Oracle Database 11g
- Understanding Remote Dependencies
- Recompiling a PL/SQL Program Unit

Modalités

- **Type d'action** :Acquisition des connaissances
- **Moyens de la formation** :Formation présentielle – 1 poste par stagiaire – 1 vidéo projecteur – Support de cours fourni à chaque stagiaire
- **Modalités pédagogiques** :Exposés – Cas pratiques – Synthèse
- **Validation** :Exercices de validation – Attestation de stages