Oracle SQL Course Content

Introduction

- Describe the features of Oracle Database 12c
- Describe the salient features of Oracle Cloud 12c
- Explain the theoretical and physical aspects of a relational database
- Describe Oracle servers implementation of RDBMS and object relational database management system (ORDBMS)

Basic DATABASE Concept and SQL

- Basic history of database concept: DBMS, RDBMS, ORDBMS
- Advantage of ORACLE database and version information
- Interface tools usage: sqlplus, isqlplus, sqldeveloper, Toad
- SQL Language overview : DQL, DML, DDL, DCL, TCL
- What is the usage of ANSI standard.
- SELECT Command – Column Alias Rules, String data,
- Concatenations with various data
- Null Value handling with number and characters,
- Arithmetic Operator
- Concatenation Operator,
- Eliminating Duplicate Rows

Restricting and Sorting Data

- WHERE Clause – Character Strings and Dates, number
- General Comparison Conditions = > >= < <= <>
- Other Comparison BETWEEN , IN , LIKE , NULL
- Logical Conditions AND OR NOT
- ORDER BY Clause, Sorting by Column Alias , Column Position, Multiple Columns

Single-Row Functions

- Character Functions: UPPER, LOWER, INITCAP, LENGTH, SUBSTR, INSTR, LPAD, RPAD,
- CONCAT, LTRIM, RTRIM, TRIM, REPLACE, TRANSLATE, REVERSE
- Number Functions: ROUND, TRUNC, MOD, POWER, CEIL , FLOOR, ABS

- Dates Functions: SYSDATE, MONTHS_BETWEEN, NEXT_DAY, LAST_DAY, ADD_MONTHS, ROUND, TRUNC, Arithmetic on Date
- Conversion Functions: Implicit Data-Type Conversion & Explicit Data-Type Conversion, TO_CHAR TO_NUMBER ,TO_DATE
- General Functions: NVL , NVL2 , NULLIF, COALESCE
- CASE Expression, DECODE
- Nested function with real-time usage

JOINS

- Equi join / simple join / normal join
- Ansi join, left outer, right outer, full outer
- Natural join, natural outer joins
- Inner join, join … using clause, join … on clause,
- Cross join, non-equi join, self join
- Oracle standard outer joins.
- Multi table Joins, Complex Joins " How to simplified complex joins.

Multi-row Functions

- Group Functions Rules, SUM, MIN, MAX, COUNT, AVG
- Creating Groups of Data: GROUP BY Clause
- Filtering Group Results: The HAVING Clause

Sub-queries

- Single-Row Subqueries- Rules, Operators : = > >= < <= <>
- Null Values in a Subquery
- Multi-Row Subqueries- Rules, Operators : IN, ANY , ALL

Reporting data using interface commands

- Pagesize, linesize , column heading , column format , colsep
- Ttitle , btitle , break on column, spool , csv file generation, text file generation

Data Manipulation Language DML and Transaction Control Language TCL

- DML : INSERT, UPDATE, DELETE, MERGE
- TCL : COMMIT, ROLLBACK, SAVEPOINT

Data Definition Language – DDL

- DDL : CREATE, ALTER, RENAME, DROP, TRUNCATE
- DEFAULT OPTION.
- Constrain table copy

Constraints

- NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK
- Column Level Constraint, Table Level Constraint " Naming constraints and usage
- Adding a Constraint, Dropping a Constraint,
- Disabling Constraints, Enabling Constraints
- Validating Constraints

Views

- Simple Views and Complex Views " Create, Drop, Source Code
- Rules for Performing DML Operations on a View
- WITH CHECK OPTION , WITH READ ONLY
- Inline Views
- Materialized View " Create, Refresh, Drop – Usage

Other Database Objects

- Sequence- NEXTVAL and CURRVAL
- Index – When to Create an Index, When Not to Create an Index.
- Synonyms

Dcl commands

- Creating Users
- Granting / Revoking Privileges
- Creating and Granting Privileges to a Role

Dictionary Tables

- Tables, Views, Synonyms, Index, Sequence, Constrains, Source and other Dictionary

SET Operators

- UNION ,
- UNION ALL ,
- INTERSECT ,
- MINUS

Advanced Date-time Functions

- Time zones ,
- Sysdate, systimestamp,
- Current_date , current_timestamp
- Sessiontimezone ,
- Storing time zone data in table

- Extract , To _Yminterval

Advanced GROUP BY Clause

- Group by with ROLLUP,
- Group by with CUBE,
- GROUPING SETS

Advanced Subqueries

- Pairwise Comparison Subquery ,
- Nonpairwise Comparison Subquery
- Correlated Subqueries,
- Correlated UPDATE,
- Correlated DELETE
- EXISTS , NOT EXISTS Operator

Hierarchical Retrieval

- Walking the Tree: From the Bottom Up , From the Top Down
- LEVEL Pseudo column,
- Connect by prior

Multi-table Insert

- Unconditional INSERT ALL
- Conditional INSERT ALL
- Conditional FIRST INSERT

Data loader

- SQLLDR Loading CSV file / Flat file into ORACLE table.

Analytic Functions

- WM_CONCAT, LAG, LEAD, RANK, DENSE_RANK
- Query_by partition_clause with sum, min, max, avg, count,
- order_by_clause with sum, min, max, avg, count,
- Psudo column : Rownum, Rowid, – Elimination duplicate data
- Connect by rownum , Connect by Level Generating random numbers, random dates,
- Quote Operator syntax and usage

Analytic SQL Training For Datawarehouse Developers

Grouping and Aggregating Data Using SQL

- Generating Reports by Grouping Related Data
- Review of Group Functions
- Reviewing GROUP BY and HAVING Clause
- Using the ROLLUP and CUBE Operators
- Using the GROUPING Function
- Working with GROUPING SET Operators and Composite Columns
- Using Concatenated Groupings with Example

Hierarchical Retrieval

- Using Hierarchical Queries
- Sample Data from the EMPLOYEES Table
- Natural Tree Structure
- Hierarchical Queries: Syntax
- Walking the Tree: Specifying the Starting Point
- Walking the Tree: Specifying the Direction of the Query
- Using the WITH Clause
- Hierarchical Query Example: Using the CONNECT BY Clause

Working with Regular Expressions

- Introducing Regular Expressions
- Using the Regular Expressions Functions and Conditions in SQL and PL/SQL
- Introducing Metacharacters
- Using Metacharacters with Regular Expressions
- Regular Expressions Functions and Conditions: Syntax
- Performing a Basic Search Using the REGEXP_LIKE Condition
- Finding Patterns Using the REGEXP_INSTR Function
- Extracting Substrings Using the REGEXP_SUBSTR Function

Analyzing and Reporting Data Using SQL

- Overview of SQL for Analysis and Reporting Functions
- Using Analytic Functions
- Using the Ranking Functions
- Using Reporting Functions

Performing Pivoting and Unpivoting Operations

- Performing Pivoting Operations

- Using the PIVOT and UNPIVOT Clauses
- Pivoting on the QUARTER Column: Conceptual Example
- Performing Unpivoting Operations
- Using the UNPIVOT Clause Columns in an UNPIVOT Operation
- Creating a New Pivot Table: Example

Pattern Matching using SQL

- Row Pattern Navigation Operations
- Handling Empty Matches or Unmatched Rows
- Excluding Portions of the Pattern from the Output
- Expressing All Permutations
- Rules and Restrictions in Pattern Matching
- Examples of Pattern Matching

Modeling Data Using SQL

- Using the MODEL clause
- Demonstrating Cell and Range References
- Using the CV Function
- Using FOR Construct with IN List Operator, incremental values and Subqueries
- Using Analytic Functions in the SQL MODEL Clause
- Distinguishing Missing Cells from NULLs
- Using the UPDATE, UPSERT and UPSERT ALL Options
- Reference Models

SQL Statement Tuning Training

Exploring the Oracle Database Architecture

- Describe the major architectural components of Oracle Database server
- Explain memory structures
- Describe background processes
- Correlate logical and physical storage structure

Introduction to SQL Tuning

- Describe what attributes of a SQL statement can make it perform poorly
- Describe the Oracle tools that can be used to tune SQL
- Explain the tuning tasks

Introduction to the Optimizer

- Describe the execution steps of a SQL statement
- Explain the need for an optimizer

- Explain the various phases of optimization
- Control the behavior of the optimizer

Interpreting Execution Plans

- Gather execution plans
- Display execution plans, display xplan
- Interpret execution plans

Application Tracing

- Configure the SQL Trace facility to collect session statistics
- Use the trcsess utility to consolidate SQL trace files
- Format trace files using the tkprof utility
- Interpret the output of the tkprof command

Optimizer Operations

- Describe the SQL operations for tables and indexes
- Describe the possible access paths for tables and indexes

Optimizer: Join Operations

- Describe the SQL operations for joins
- Describe the possible access paths for joins

Other Optimizer Operations

- Describe Clusters, In-List, Sorts, Filters and Set Operations
- Use Result Cache operations

Case Study: Star Transformation

- Define a star schema, a star query plan without transformation and a star query plan after transformation

Optimizer Statistics

- Gather optimizer statistics
- Gather system statistics
- Set statistic preferences
- Use dynamic sampling
- Manipulate optimizer statistics

Using Bind Variables

- Explain the benefits of using bind variables

- Use bind peeking
- Use adaptive cursor sharing

SQL Tuning Advisor

- Describe statement profiling
- Use SQL Tuning Advisor

Using SQL Access Advisor

- Use SQL Access Advisor

Automating SQL Tuning

- Use Automatic SQL Tuning

SQL Plan Management

- Manage SQL performance through changes
- Set up SQL Plan Management
- Set up various SQL Plan Management scenarios

Using Optimizer Hints

- Use hints when appropriate
- Specify hints for Optimizer mode, Query transformation, Access path, Join orders, Join methods and Views

Parallel Queries

Parallel Processing Concepts

- Explain what parallel processing is and why is it useful

Basics of Parallel Execution

- Describe operations that can be parallelized
- Explain parallel execution theory
- Understand impact of initialization parameter on parallel execution

Manual DOP Management

- Understand an explain plan of a parallel query
- Understand an explain plan of parallel DML and DDL

Simplified Auto DOP

- Understand the new parameters of Auto DOP

- Explain when to use Auto DOP
- Use Auto DOP

Statement Queuing

- Explain statement queuing, concurrency and DBRM

In-Memory Parallel execution

- Use in-memory parallel execution

Data Warehouse Administration

Partitioning Concepts

- Explain the available partitioning strategies
- Explain partition pruning
- Implement partition enhancements in star query optimization

Materialized Views

- Use summaries to improve performance
- Differentiate materialized view types

Oracle Pl/Sql Course Content

Introduction

- Course Objectives
- Course Agenda
- Human Resources (HR) Schema
- Introduction to SQL Developer

Introduction to PL/SQL

- PL/SQL Overview
- Benefits of PL/SQL Subprograms
- Overview of the Types of PL/SQL blocks
- Create a Simple Anonymous Block
- Generate Output from a PL/SQL Block

PL/SQL Identifiers

- List the different Types of Identifiers in a PL/SQL subprogram
- Usage of the Declarative Section to define Identifiers
- Use variables to store data
- Identify Scalar Data Types
- The %TYPE Attribute
- What are Bind Variables?
- Sequences in PL/SQL Expressions

Write Executable Statements

- Describe Basic PL/SQL Block Syntax Guidelines
- Comment Code
- Deployment of SQL Functions in PL/SQL
- How to convert Data Types?
- Nested Blocks

Interaction with the Oracle Server

- Invoke SELECT Statements in PL/SQL to Retrieve data
- Data Manipulation in the Server Using PL/SQL
- SQL Cursor concept/save and Discard Transactions
- Usage of SQL Cursor Attributes to Obtain Feedback on DML

Control Structures

- Conditional processing Using IF Statements

- Conditional processing Using CASE Statements
- Use simple Loop Statement
- Use While Loop Statement
- Use For Loop Statement
- Describe the Continue Statement

Composite Data Types

- Use PL/SQL Records
- The %ROWTYPE Attribute
- Insert and Update with PL/SQL Records
- Associative Arrays (INDEX BY Tables)
- Examine INDEX BY Table Methods
- Use INDEX BY Table of Records

Explicit Cursors

- What are Explicit Cursors?
- Declare the Cursor
- Open the Cursor
- Fetch data from the Cursor
- Close the Cursor
- Cursor FOR loop
- Explicit Cursor Attributes
- FOR UPDATE Clause and WHERE CURRENT Clause

Exception Handling

- Understand Exceptions
- Handle Exceptions with PL/SQL
- Trap Predefined Oracle Server Errors
- Trap Non-Predefined Oracle Server Errors
- Trap User-Defined Exceptions
- Propagate Exceptions
- RAISE_APPLICATION_ERROR Procedure

Stored Procedures and Functions

- Understand Stored Procedures and Functions
- Differentiate between anonymous blocks and subprograms
- Create a Simple Procedure
- Create a Simple Procedure with IN parameter
- Create a Simple Function

- Execute a Simple Procedure
- Execute a Simple Function

Create Stored Procedures

- Create a Modularized and Layered Subprogram Design
- Modularize Development With PL/SQL Blocks
- Describe the PL/SQL Execution Environment
- Identity the benefits of Using PL/SQL Subprograms
- List the differences Between Anonymous Blocks and Subprograms
- Create, Call, and Remove Stored Procedures Using the CREATE Command and SQL Developer
- Implement Procedures Parameters and Parameters Modes
- View Procedures Information Using the Data Dictionary Views and SQL Developer

Create Stored Functions

- Create, Call, and Remove a Stored Function Using the CREATE Command and SQL Developer
- Identity the advantages of Using Stored Functions in SQL Statements
- List the steps to create a stored function
- Implement User-Defined Functions in SQL Statements
- Identity the restrictions when calling Functions from SQL statements
- Control Side Effects when calling Functions from SQL Expressions
- View Functions Information

Create Packages

- Identity the advantages of Packages
- Describe Packages
- List the components of a Package
- Develop a Package
- How to enable visibility of a Packages components?
- Create the Package Specification and Body Using the SQL CREATE Statement and SQL Developer
- Invoke Package Constructs
- View PL/SQL Source Code Using the Data Dictionary

Packages

- Overloading Subprograms in PL/SQL
- Use the STANDARD Package
- Use Forward Declarations to Solve Illegal Procedure Reference
- Implement Package Functions in SQL and Restrictions
- Persistent State of Packages
- Persistent State of a Package Cursor

- Control Side Effects of PL/SQL Subprograms
- Invoke PL/SQL Tables of Records in Packages

Implement Oracle-Supplied Packages in Application Development

- What are Oracle-Supplied Packages?
- Examples of Some of the Oracle-Supplied Packages
- How Does the DBMS_OUTPUT Package Work?
- Use the UTL_FILE Package to Interact With Operating System Files
- Invoke the UTL_MAIL Package
- Write UTL_MAIL Subprograms

Dynamic SQL

- The Execution Flow of SQL
- What is Dynamic SQL?
- Declare Cursor Variables
- Dynamically executing a PL/SQL Block
- Configure Native Dynamic SQL to Compile PL/SQL Code
- Invoke DBMS_SQL Package
- Implement DBMS_SQL with a Parameterized DML Statement
- Dynamic SQL Functional Completeness

Design Considerations for PL/SQL Code

- Standardize Constants and Exceptions
- Understand Local Subprograms
- Write Autonomous Transactions
- Implement the NOCOPY Compiler Hint
- Invoke the PARALLEL_ENABLE Hint
- The Cross-Session PL/SQL Function Result Cache
- The DETERMINISTIC Clause with Functions
- Usage of Bulk Binding to Improve Performance

Triggers

- Describe Triggers
- Identify the Trigger Event Types and Body
- Business Application Scenarios for Implementing Triggers
- Create DML Triggers Using the CREATE TRIGGER Statement and SQL Developer
- Identify the Trigger Event Types, Body, and Firing (Timing)
- Statement Level Triggers Versus Row Level Triggers
- Create Instead of and Disabled Triggers
- How to Manage, Test, and Remove Triggers?

Create Compound, DDL, and Event Database Triggers

- What are Compound Triggers?
- Identify the Timing-Point Sections of a Table Compound Trigger
- Compound Trigger Structure for Tables and Views
- Implement a Compound Trigger to Resolve the Mutating Table Error
- Compare Database Triggers to Stored Procedures
- Create Triggers on DDL Statements
- Create Database-Event and System-Event Triggers
- System Privileges Required to Manage Triggers

The PL/SQL Compiler

- What is the PL/SQL Compiler?
- Describe the Initialization Parameters for PL/SQL Compilation
- List the New PL/SQL Compile Time Warnings
- Overview of PL/SQL Compile Time Warnings for Subprograms
- List the benefits of Compiler Warnings
- List the PL/SQL Compile Time Warning Messages Categories
- Setting the Warning Messages Levels: Using SQL Developer, PLSQL_WARNINGS Initialization Parameter, and the DBMS_WARNING Package Subprograms
- View Compiler Warnings: Using SQL Developer, SQL*Plus, or the Data Dictionary Views

Manage PL/SQL Code

- What Is Conditional Compilation?
- Implement Selection Directives
- Invoke Predefined and User-Defined Inquiry Directives
- The PLSQL_CCFLAGS Parameter and the Inquiry Directive
- Conditional Compilation Error Directives to Raise User-Defined Errors
- The DBMS_DB_VERSION Package
- Write DBMS_PREPROCESSOR Procedures to Print or Retrieve Source Text
- Obfuscation and Wrapping PL/SQL Code

Manage Dependencies

- Overview of Schema Object Dependencies
- Query Direct Object Dependencies using the USER_DEPENDENCIES View
- Query an Object€™s Status
- Invalidation of Dependent Objects
- Display the Direct and Indirect Dependencies
- Fine-Grained Dependency Management in Oracle Database 11g
- Understand Remote Dependencies
- Recompile a PL/SQL Program Unit

Designing PL/SQL Code

- Describe the predefined data types
- Create subtypes based on existing types for an application
- List the different guidelines for cursor design
- Cursor variables

Using Collections

- Overview of collections
- Use Associative arrays
- Use Nested tables
- Use VARRAYs
- Compare nested tables and VARRAYs
- Write PL/SQL programs that use collections
- Use Collections effectively

Manipulating Large Objects

- Describe a LOB object
- Use BFILEs
- Use DBMS_LOB.READ and DBMS_LOB.WRITE to manipulate LOBs
- Create a temporary LOB programmatically with the DBMS_LOB package
- Introduction to SecureFile LOBs
- Use Secure File LOBs to store documents
- Convert BasicFile LOBs to SecureFile LOB format
- Enable reduplication and compression

Using Advanced Interface Methods

- Calling External Procedures from PL/SQL
- Benefits of External Procedures
- C advanced interface methods
- Java advanced interface methods

Performance and Tuning

- Understand and influence the compiler
- Tune PL/SQL code
- Enable intra unit inlining
- Identify and tune memory issues
- Recognize network issues

Improving Performance with Caching

- Describe result caching
- Use SQL query result cache
- PL/SQL function cache
- Review PL/SQL function cache considerations

Analyzing PL/SQL Code

- Finding Coding Information
- Using DBMS_DESCRIBE
- Using ALL_ARGUMENTS
- Using DBMS_UTILITY.FORMAT_CALL_STACK
- Collecting PL/Scope Data
- The USER/ALL/DBA_IDENTIFIERS Catalog View
- DBMS_METADATA Package

Profiling and Tracing PL/SQL Code

- Tracing PL/SQL Execution
- Tracing PL/SQL: Steps

Implementing VPD with Fine-Grained Access Control

- Understand how fine-grained access control works overall
- Describe the features of fine-grained access control
- Describe an application context
- Create an application context
- Set an application context
- List the DBMS_RLS procedures
- Implement a policy
- Query the dictionary views holding information on fine-grained access

Safeguarding Your Code Against SQL Injection Attacks

- SQL Injection Overview
- Reducing the Attack Surface
- Avoiding Dynamic SQL
- Using Bind Arguments
- Filtering Input with DBMS_ASSERT
- Designing Code Immune to SQL Injections
- Testing Code for SQL Injection Flaws

Contact Info:

+91 9884412301 | +91 9884312236

info@credosystemz.com

Know more about Oracle PL/SQL

New # 30, Old # 16A, Third Main Road,
Rajalakshmi Nagar, Velachery, Chennai
(Opp. to MuruganKalyanaMandapam)

BOOK A FREE DEMO