UNIT IV

PL/SQL: A Programming Language: History–Fundamentals–Block Structure–Comments – Data Types–Other Data Types–Variable Declaration–Anchored Declaration-Assignment operation – Bind Variables–Substitution Variables–Printing in PL/SQL–Arithmetic Operators. Control Structures and Embedded SQL: Control Structures–Nested Blocks–SQL in PL/SQL–Data Manipulation –Transaction Control statements. PL/SQL Cursors and Exceptions: Cursors – Implicit & Explicit Cursors and Attributes–Cursor FOR loops–SELECT…FOR UPDATE – WHERE CURRENT OF Clause– Cursor with Parameters– Cursor Variables– Exceptions– Types of Exceptions.

Text Book: “DATABASE SYSTEMS USING ORACLE”-NILESH SHAH,2nd Edition, PHI

Prepared by Dr.P.Radha

PL/SQL: A Programming Language

- History
- Fundamentals
- Block Structure
- Comments
- Data Types
- Other Data Types
- Variable Declaration
- Anchored Declaration
- Assignment operation
- Bind Variables
- Substitution Variables
- Printing
- Arithmetic Operators

A BRIEF HISTORY OF PL/SQL

- Before PL/SQL was developed, users embedded SQL statements into hot languages like C++ and Java.
- PL/SQL version 1.0 was introduced with Oracle 6.0 in 1991.
- Version 1.0 had very limited capabilities, however, and was far from being a full-fledged programming language.
- It was merely used for batch processing.

Cont…

- With versions 2.0, 2.1, and 2.2, the following new features were introduced:
- The transaction control statements SAVEPOINT, ROLLBACK, and COMMIT.
- The DML statements INSERT, DELETE, and UPDATE.
- The extended data types Boolean, BINARY_INTEGER, PL/SQL records, and PL/SQL tables.
- Built-in functions—character, numeric, conversion, and date functions.
- Built-in packages.
- The control structures sequence, selection, and looping.
- Database access through work areas called cursors.
- Error handling.

Cont…

- Modular programming with procedures and functions.
- Stored procedures, functions, and packages.
- Programmer-defined subtypes.
- DDL support through the DBMS_SQL package.
- The PL/SQL wrapper.
- The DBMS_JOB job scheduler.
- File I/O with the UTF_FILE package

FUNDAMENTALS OF PL/SQL

- A PL/SQL program consists of statements. You may use upper- or lowercase letters in your program.
- In other words, PL/SQL is not case sensitive except for character string values enclosed in single quotes.
- Like any other programming language, PL/SQL statements consist of reserved words, identifiers, delimiters, literals, and comments.

Cont…

Reserved Words

- The reserved words, or key words, are words provided by the language that have a specific use in the language.
- For example, DECLARE, BEGIN, END, IF, WHILE, EXCEPTION, PROCEDURE, FUNCTION, PACKAGE, and TRIGGER are some of the reserved words in PL/SQL.

Cont…

User-Defined Identifiers

- User-defined identifiers are used to name variables, constants, procedures, functions, cursors, tables, records, and exceptions.
- A user must obey the following rules in naming these identifiers:

Cont…

- The name can be from 1 to 30 characters in length.
- The name must start with a letter.
- Letters (A–Z, a–z), numbers, the dollar sign ($), the number sign (#) and the underscore (_) are allowed.
- Spaces are not allowed.
- Other special characters are not allowed.
- Key words cannot be used as user-defined identifiers.
- Names must be unique within a block.
- A name should not be the same as the name of a column used in the block.

Cont…

Literals

- Literals are values that are not represented by user-defined identifiers. Literals are of three types: Numeric, Character, and Boolean. For example:
- Numeric **100, 3.14,** — **55**, **5.25E7, or NULL**
- Character**‘A’**, **‘this is a string’**, **‘0001’**, **‘25-MAY-00’, ‘ ’, or NULL**
- Boolean **TRUE, FALSE, or NULL**

PL/SQL BLOCK STRUCTURE

- PL/SQL is a block structured language. A program can be divided into logical blocks
- The block structure gives modularity to a PL/SQL program, and each object within a block has “scope.” Blocks are of two types:
- An ***anonymous block*** is a block of code without a name. It can be used anywhere in a program and is sent to the server engine for execution at runtime.

Cont…

- A ***named block*** is a block of code that is named. A *subprogram* is a named block that can be called and can take arguments.
- A *procedure* is a subprogram that can perform an action, whereas a *function* is a subprogram that returns a value.
- A *package* is formed from a group of procedures and functions. A *trigger* is a block that is called implicitly by a DML statement.

Cont…

A PL/SQL block consists of three sections:

- A **declaration section**.
- An **executable section**.
- An **exception-handling section**.

COMMENTS

- Comments are used to document programs. They are written as part of a program, but they are not executed.
- In fact, comments are ignored by the PL/SQL engine.
- It is a good programming practice to add comments to a program, because this helps in readability and debugging of the program

Cont…

There are two ways to write comments in PL/SQL:

- To write a single-line comment, two dashes (--) are entered at the begin- ning of a new line. For example,

  - -This is a single-line comment.

- To write a multiline comment, comment text is placed between /* and */. A multiline comment can be written on a separate line by itself, or it can be used on a line of code as well. For example,

  /*   This is a
  multiline comment
  that ends here. */

DATA TYPES

Each constant and variable in the program needs a data type. The data type decides the type of value that can be stored in a variable. PL/SQL has four data types:

- Scalar.
- Composite.
- Reference.
- LOB

Cont…

- A scalar data type is not made up of a group of elements.
- It is atomic in nature.
- The composite data types are made up of elements or components.
- PL/SQL supports three composite data types—***records, tables***, and V***arrays***

Cont…

There are four major categories of scalar data types:

- Character.
- Number.
- Boolean.
- Date.

OTHER DATA TYPES

NLS

The National Language Support (NLS) data type is for character sets in which multiple bytes are used for character representation. NCHAR and NVARCHAR2 are examples of NLS data types.

Cont…

LOB

- Like Oracle9i, PL/SQL also supports Large Object (LOB) data types to store large values of character, raw, or binary data.
- The LOB types allow up to 4 gigabytes of data. LOB variables can be given one of the following data types:

Cont…

- The ***BLOB*** type contains a pointer to the large binary object inside the database.
- The ***CLOB*** type contains a pointer to a large block of single-byte character data of fixed width.
- The ***NCLOB*** type contains a pointer to a large block of multibyte character data of fixed width.
- The ***BFILE*** type contains a pointer to large binary objects in an external operating system file. It would contain the directory name and the filename

VARIABLE DECLARATION

- A scalar variable or a constant is declared with a data type and an initial value assignment. The declarations are done in the DECLARE section of the program block.
- The initial value assignment for a variable is optional unless it has a NOT NULL constraint. The constants and NOT NULL type variables must be initialized.

Cont…

The general syntax is

DECLARE

identifiername [CONSTANT] datatype [NOT NULL] [:= | DEFAULT expression];

ANCHORED DECLARATION

- PL/SQL uses %TYPE attribute to anchor a variable’s data type. Another variable or a column in a table can be used for anchoring

The general syntax is

Variablename typeattribute%TYPE [value assignment];

ASSIGNMENT OPERATION

- The assignment operation is one of the ways to assign a value to a variable.
- You have already learned that a variable can be initialized at the time of declaration by using the DEFAULT option or :=.
- The assignment operation is used in the executable section of the program block to assign a literal, another variable’s value, or the result of an expression to a variable.

Cont…

- The general syntax is

  VariableName := Literal* | *VariableName* | *Expression;

BIND VARIABLES

- Bind variables are also known as host variables.
- These variables are declared in the host SQL* Plus environment and are accessed by a PL/SQL block.
- Anonymous blocks do not take any arguments, but they can access host variables with a colon prefix (:) and the host variable name.

Cont…

The syntax of a host variable declaration is

VARIABLE variablename datatype

For example,

```
SQL> VARIABLE double NUMBER
```

SUBSTITUTION VARIABLES IN PL/SQL

- PL/SQL does not have any input capabilities in terms of having an input statement.
- There are no explicit input/output (I/O) statements, but substitution variables of SQL are available in PL/SQL.
- Substitution variables have limitations, which become apparent in a loop.

PRINTING IN PL/SQL

- There is no explicit output statement in PL/SQL. Oracle does have a built-in package called DBMS_OUTPUT with the procedure PUT_LINE to print.
- An environment variable named SERVEROUTPUT must be toggled ON to view output from it.

Cont…

- The DBMS_OUTPUT is the most frequently used package because of its ca- pabilities to get lines from a file and to put lines into the buffer.
- The PUT_LINE procedure displays information passed to the buffer and puts a new-line marker at the end.

For example,

DBMS_OUTPUT.PUT_LINE (‘This line will be displayed’);

ARITHMETIC OPERATORS

- Five standard arithmetic operators are available in PL/SQL for calculations.
- If more than one operator exists in an arithmetic expression, the following order of precedence is used:

Cont…

- Exponentiation is performed first, multiplication and division are performed next, and addition and subtraction are performed last.
- If more than one operator of the same priority is present, they are performed from left to right.
- Whatever is in parentheses is performed first.

Cont…

Arithmetic Operator	Use
+	Addition
-	Subtraction
/	Division
*	Multiplication
**	Exponentiation

Control Structures and Embedded SQL:

- Control Structures
- Nested Blocks
- SQL in PL/SQL
- Data Manipulation
- Transaction Control statements.

CONTROL STRUCTURES

- In a procedural language like PL/SQL, there are three basic programming control structures:
- In a ***sequential structure***, a series of instructions are performed from the beginning to the end in a linear order. None of the instructions is skipped, and none of the instructions is repeated.

Cont..

- The ***selection structure*** is also known as a ***decision structure*** or an ***IF- structure***.
- It involves conditions with a TRUE or FALSE outcome.
- Based on the outcome, one of the options is performed, and the other option is skipped. Selection statements are also available for multiple options.

Cont..

- In a ***looping structure***, a series of instructions is performed repeatedly.
- There are different looping statements appropriate for a variety of situations.
- A programmer has to write a loop correctly to make it perform a specific number of times.

Cont..

Selection Structure

- There are three selection or conditional statements in PL/SQL.
- Relational operators, logical operators, and other special operators are used to create Boolean expressions or conditions

Cont..

- The AND and OR operators are binary operators, because they work on two conditions.
- The NOT operator is a unary operator, because it works on a single condition.

Cont..

Nested IF.

- The nested IF statement contains an IF statement within another IF statement. If the condition in the outer IF statement is TRUE, the inner IF statement is performed. Any IF statement with a compound condition can be written as a nested IF statement. For example, the program segment in Figure 11-12 assigns an insurance surcharge based on an individual’s gender and age. There are four categories:
- Male 25 or over.
- Male under 25.
- Female 25 or over.
- Female under 25.

Cont…

Looping Structure

- Looping means iterations. A loop repeats a statement or a series of statements a specific number of times, as defined by the programmer.
- Three types of looping statements are available in PL/SQL:
- Basic loop.
- WHILE loop.
- FOR loop.

Cont…

Basic loop

- A basic loop is a loop that is performed repeatedly. Once a loop is entered, all statements in the loop are performed.
- When the bottom of the loop is reached, control shifts back to the top of the loop. The loop will continue in- finitely.
- An infinite loop, or a “never-ending loop,” is a logical error in programming.
- The only way to terminate a basic loop is by adding an EXIT statement inside the loop.

Cont…

The general syntax is

```
 LOOP
Looping statement1; Looping statement2;
. . .
Looping statementN; EXIT [WHEN condition];
END LOOP;
```

Cont…

WHILE loop

- The WHILE loop is an alternative to the basic loop and is per- formed as long as the condition is true. It terminates when the condition becomes false. If the condition is false at the beginning of the loop, the loop is not performed at all.
- The WHILE loop does not need an EXIT statement to terminate.

Cont..

The general syntax is

```
WHILE condition
 LOOP Looping statement1;
Looping statement2;
...
Looping statement n;
END LOOP;
```

Cont…

FOR loop

- The FOR loop is the simplest loop you can write. Unlike the basic and WHILE loops, you do not have to initialize, test, and increment/decrement the loop control variable separately

Cont…

The general syntax is

```
FOR counter IN [REVERSE] lower..upper LOOP
Looping statement1
Looping statement2
.. .
Looping statementN
 END LOOP;
```

Cont..

Nested loops

- Loops can be nested to many levels. When the inner loop ends, it does not automatically end the outer loop enclosing it.
- The loop labels use the same naming rules as those used for identifiers. The loops are labeled before the key word LOOP on the same line or on a separate line.

NESTED BLOCKS

- PL/SQL block may contain another PL/SQL block; in other words, PL/SQL blocks can be nested.
- The execution starts with the outer block and continues with the inner block.
- The variables declared in the outer block are global to the inner block, and they are accessible in the inner block.
- The variables declared in the inner block, however, are not accessible in the outer block.

SQL IN PL/ SQL

- The PL/SQL statements have control structures for calculations, decision making, and iterations
- PL/SQL does not support Data Definition Language (DDL) statements, such as CREATE, ALTER, and DROP.
- The Data Control Language (DCL) statements GRANT and REVOKE also are not available in PL/SQL.

DATA MANIPULATION IN PL/SQL

The three DML statements to manipulate data are:

- The **INSERT** statement to add a new row in a table.
- The **DELETE** statement to remove a row or rows.
- The **UPDATE** statement to change values in a row or rows.

TRANSACTION CONTROL STATEMENTS

- The COMMIT statement to commit the current transaction.
- The SAVEPOINT statement to mark a point in your transaction.
- The ROLLBACK [TO SAVEPOINT *n*] statement to discard all or part of the transaction.

PL/SQL Cursors and Exceptions

- Cursors
- Implicit & Explicit Cursors and Attributes
- Cursor FOR loops
- SELECT…FOR UPDATE
- WHERE CURRENT OF clause
- Cursor with Parameters
- Cursor Variables
- Exceptions
- Types of Exceptions

Cursors

- When you execute an SQL statement from a PL/SQL block, Oracle assigns a private work area for that statement.
- The work area, called a cursor, stores the statement and the results returned by execution of that statement.
- A cursor is created either implicitly or explicitly by you.

Cont…

Types of Cursors

- In a ***static cursor***, the contents are known at compile time. The cursor ob- ject for such an SQL statement is always based on one SQL statement.
- In a ***dynamic cursor***, a cursor variable that can change its value is used. The variable can refer to different SQL statements at different times.

IMPLICIT CURSORS

- PL/SQL creates an implicit cursor when an SQL statement is executed from within the program block.
- The implicit cursor is created only if an explicit cursor is not attached to that SQL statement. Oracle opens an implicit cursor, and the pointer is set to the first (and the only) row in the cursor. Then, the SQL statement is fetched and executed by the SQL engine on the Oracle server

EXPLICIT CURSORS

- An explicit cursor is declared as a SELECT statement in the PL/SQL block. It is given a name, and you can use explicit statements to work with it.
- You have total control of when to open the cursor, when to fetch a row from it, and when to close it.
- There are cursor attributes in PL/SQL to get the status information on explicit cursors.

Cont…

Four actions can be performed on an explicit cursor:

- Declare it.
- Open it.
- Fetch row(s) from it.
- Close it.

Cont..

Declaring an Explicit Cursor

- A cursor is declared as a SELECT statement. The SELECT statement must not have an INTO clause in a cursor's declaration.
- If you want to retrieve rows in a specific order into a cursor, an ORDER BY clause can be used in the SELECT statement.
- The general syntax is

```
DECLARE
CURSOR cursorname IS
SELECT statement;
```

Cont…

Opening a Cursor

- When a cursor is opened, its SELECT query is executed.
- The active set is created using all tables in the query and then restricting to rows that meet the criteria.
- The data retrieved by the SELECT query is brought into the cursor or the work area.
- The cursor points to the first row in the active set. PL/SQL uses an OPEN statement to open a cursor.
- The general syntax is

OPEN cursorname;

Cont..

Fetching Data from a Cursor

- The SELECT statement creates an active set based on tables in the FROM clause, column names in the SELECT clause, and rows based on conditions in the WHERE clause.
- The number of variables must match the number of columns in the row. In PL/SQL, a FETCH statement is used for this action.
- The general syntax is

FETCH cursorname INTO variablelist / recordname;

Cont…

Closing a cursor

- When you are done with a cursor, you should close it.
- A closed cursor can be reopened again. If you terminate your PL/SQL program with- out closing an open cursor, it will not result in an exception.
- A user releases memory by closing a cursor.
- PL/SQL uses the CLOSE statement to close a cursor. The general syntax is

  CLOSE cursorname;

EXPLICIT CURSOR ATTRIBUTES

Actions can be performed on cursors with OPEN, FETCH, and CLOSE statements. The four explicit cursor attributes are:

%ISOPEN	It returns TRUE if the cursor is open; otherwise, it returns FALSE.
%FOUND	It returns TRUE if the last fetch returned a row; otherwise, it returns FALSE.
%NOTFOUND	It returns TRUE if the last fetch did not return a row; otherwise, it returns FALSE. It complements the %FOUND attribute.
%ROWCOUND	It returns total number of rows returned.

IMPLICIT CURSOR ATTRIBUTES

- An implicit cursor cannot be opened, fetched from, or closed with a statement. You do not name implicit cursors.
- The cursor attributes are available for an implicit cursor with the name SQL as a prefix.

Cont..

The four attributes for a implicit cursor are:

- SQL%ISOPEN.
- SQL%ROWCOUNT.
- SQL%NOTFOUND.
- SQL%FOUND.

CURSOR FOR LOOPS

- The cursor FOR loop is the easiest way to write a loop for explicit cursors. The cursor is opened implicitly when the loop starts.
- A row is then fetched into the record from the cursor with every iteration of the loop.
- The cursor is closed automatically when the loop ends, and the loop ends when there are no more rows

Cont…

- The general syntax is

```
FOR recordname IN cursorname LOOP
 Loop statements;
. . .
END LOOP;
```

Cont…

Cursor FOR Loop Using a Subquery

- Use of a subquery in the cursor FOR loop eliminates declaration of an explicit cursor.
- The cursor is created by a subquery in the FOR loop statement itself.

SELECT . . . FOR UPDATE CURSOR

- When you type a SELECT query, the result is returned to you without locking any rows in the table.
- Row locking is kept to a minimum. You can explicitly lock rows for update before changing them in the program.
- The FOR UPDATE clause is used with the SELECT query for row locking.

Cont…

- The locked rows are not available to other users for DML statements until you release them with COMMIT or ROLLBACK commands.
- Rows that are locked for update do not have to be updated.

- The general syntax is

```
CURSOR cursorname IS
SELECT columnnames
FROM tablename(s) [WHERE condition]
FOR UPDATE [OF columnnames] [NOWAIT];
```

WHERE CURRENT OF CLAUSE

- In a cursor, data manipulation in the form of UPDATE or DELETE is performed on rows fetched. The WHERE CURRENT OF clause allows you to perform data manipulation only on a recently fetched row.
- The general syntax is

```
UPDATE tablename SET clause
WHERE CURRENT OF cursorname;
DELETE FROM tablename
WHERE CURRENT OF cursorname;
```

CURSOR WITH PARAMETERS

- A cursor can be declared with parameters, which allow you to pass values to the cursor.
- These values are passed to the cursor when it is opened, and they are used in the query when it is executed.
- With the use of parameters, you can open and close a cursor many times with different values.

Cont…

- When parameters are passed, you need not worry about the scope of variables.

The general syntax is

```
CURSOR cursorname
[(parameter1 datatype, parameter2 datatype, )]
IS
SELECT query;
```

CURSOR VARIABLES: AN INTRODUCTION

- An explicit cursor is the name of the work area for an active set.
- A cursor variable is a reference to the work area.
- A cursor is based on one specific query, whereas a cursor variable can be opened with different queries within a program.
- A static cursor is like a constant, and a cursor variable is like a pointer to that cursor.

Cont…

REF CURSOR Type

- Two steps are involved in creating a cursor variable. First, you have to create a referenced cursor type.
- Second, you have to declare an actual cursor variable with the referenced cursor type.
- The general syntax is

TYPE cursortypename IS REF CURSOR [RETURN returntype];

cursorvarname cursortypename;

Cont…

Opening a Cursor Variable

You assign a cursor to the cursor variable when you OPEN it.

The general syntax is

- ***OPEN cursorname / cursorvarname FOR SELECT query;***

Cont…

Fetching from a Cursor Variable

- The fetching action is same as that of a cursor. The compiler checks the data structure type after the INTO clause to see if it matches the query linked to the cursor.

The general syntax is

FETCH cursorvarname INTO recordname / variablelist;

EXCEPTIONS

- An exception occurs when an unwanted situation arises during the execution of a program.
- Exceptions can result from a system error, a user error, or an application error.
- When an exception occurs, control of the current program block shifts to another section of the program, known as the exception section, to handle exceptions.

TYPES OF EXCEPTIONS

There are three types of exceptions in PL/SQL:

- ***Predefined Oracle server exceptions*** are exceptions that are named by PL/SQL and are raised implicitly when a PL/SQL or DBMS error occurs.
- ***Non-predefined Oracle server exceptions*** are standard Oracle server errors that are not named by the system. They can be declared in the declaration section but are raised implicitly by the server. These exceptions do not have a name, but they do have an associated error number.

Cont…

3. **User-defined exceptions** are exceptions that are declared in the declaration section and are raised by the user explicitly.

- The user decides which abnormal condition is an exception.
- The Oracle server does not consider these conditions to be errors.