Introduction to Oracle PL/SQL

G. M. Shahariar
Lecturer
Department of CSE
Ahsanullah University of Science & Technology

What is PL/SQL?

- **Oracle PL/SQL** is an extension of SQL language
- combines the data manipulation power of SQL with the processing power of procedural language to create super powerful SQL queries.
- PL/SQL means instructing the compiler 'what to do' through SQL and 'how to do' through its procedural way.
- Similar to other database languages, it gives more control to the programmers by the use of loops, conditions and object-oriented concepts.
- The PL/SQL Full form is "Procedural Language extensions to SQL".

https://www.guru99.com/introduction-pl-sql.html

PL/SQL syntax

```
SET SERVEROUTPUT ON;  -- if you have anything to output
[DECLARE]
   Declaration statements;
BEGIN
   Execution statements;
[EXCEPTION]
      Exception handling statements;
END;
/
```

The declaration section allows you to define data types, structures, and variables. You often declare variables in the declaration section by giving names, data types, and initial values.

The execution section is required in a block structure and it must have at least one statement. The execution section is the place where you put the execution code or business logic code. You can use both procedural and SQL statements inside the execution section.

The exception handling section is starting with the EXCEPTION keyword. The exception section is the place that you put the code to handle exceptions. You can either catch or handle exceptions in the exception section.

Notice that the single forward slash (/) is a signal to instruct SQL*Plus to execute the PL/SQL block.

- PL/SQL program units organize the code into blocks.

- A block without a name is known as an anonymous block.

- The anonymous block is the simplest unit in PL/SQL. It is called anonymous block because it is not saved in the Oracle database.

- An anonymous block is an only one-time use and useful in certain situations such as creating test units.

Your Task

- Open SQLPLUS
- Log In using credentials
- Open Notepad++
- Create a new file
- Name it
- Save it as sql file
- Follow me, repeat after me.

Variables

Declare

Anchors

Assignment

Initialization

IF THEN

Problem

Check if a number is even.

If the number is even, print EVEN.

Use IF…THEN syntax to solve the problem.

“IF…THEN” Syntax

```
SET SERVEROUTPUT ON;

DECLARE
            ..............................................
BEGIN
        IF …………..  THEN

                  ......................................

        END IF;
END;
/
```

IF THEN … ELSE

Problem

Check whether a number is even or odd.

If the number is Even, print EVEN.

If the number is Odd, print ODD.

Use IF…THEN…ELSE syntax to solve the problem.

“IF…THEN ELSE” Syntax

```
SET SERVEROUTPUT ON;

DECLARE
      ………………………………………….
BEGIN
    IF ………….. THEN

         ……………………………………….
    ELSE
         ……………………………………….

    END IF;
END;
/
```

IF THEN … ELSIF THEN … ELSE

Problem

Mod a number by 3.

There can be three possible results –

If the result is 0, print ZERO
If the result is 1, print ONE
If the result is 2, print TWO

Use IF..THEN..ELSIF…THEN…ELSE syntax to solve the problem.

“IF…THEN…ELSIF….THEN…ELSE” Syntax

```
SET SERVEROUTPUT ON;
DECLARE
     ...........................................................
BEGIN
     IF ………….. THEN
               .......................................
     ELSIF……………… THEN
               .......................................
     ELSE
               .......................................
     END IF;
END;
/
```

CASE … WHEN THEN… ELSE

Problem

Mod a number by 3.

There can be three possible results –

- If the result is 0, print ZERO
- If the result is 1, print ONE
- If the result is 2, print TWO

Use CASE…WHEN…THEN…ELSE syntax to solve the problem.

“CASE…WHEN…THEN…ELSE” Syntax

```
SET SERVEROUTPUT ON;
DECLARE
     ………………………………………………………….
BEGIN
     CASE
          WHEN ………….. THEN
               …………………………………….
          WHEN ……………… THEN
               …………………………………….
          ELSE
               …………………………………….
     END CASE;
END;
/
```

Another CASE…WHEN…THEN…ELSE Syntax !

“CASE…WHEN…THEN…ELSE” Syntax

```
SET SERVEROUTPUT ON;
DECLARE
        ………………………………………………………………….
BEGIN
        CASE ……………….
              WHEN ………….. THEN
                     …………………………………….
              WHEN ………………. THEN
                     …………………………………….
              ELSE
                     …………………………………….
        END CASE;
END;
/
```

LOOP

Problem

Print 1 2 3 4 5.

Use LOOP, WHILE LOOP, FOR LOOP.

Observe the breaking condition in each case.

“PL/SQL LOOP” Syntax

```
SET SERVEROUTPUT ON;
DECLARE
        ..................................................................
BEGIN
    LOOP
        ..........................................................
        ..............................................................
        IF ..........................THEN
            EXIT;
        END IF;

    END LOOP;
END;
/
```

- Another LOOP Syntax
- Change in break condition

“PL/SQL LOOP” Syntax

```
SET SERVEROUTPUT ON;
DECLARE
        ………………………………………………..
BEGIN
    LOOP
        …………………………………………….
        ……………………………………………….
        EXIT WHEN ……………….
    END LOOP;
END;
/
```

WHILE LOOP

“WHILE LOOP” Syntax

```
SET SERVEROUTPUT ON;
DECLARE
          .......................................................
BEGIN
     WHILE .........................................
     LOOP
          ...................................................
          .......................................................
     END LOOP;
END;
/
```

FOR LOOP

“FOR LOOP” Syntax

```
SET SERVEROUTPUT ON;

DECLARE

            ……………………………………………..
BEGIN
      FOR………….  IN ……………..  LOOP

            ………………………………………….

            ……………………………………………

      END LOOP;
END;
/
```

Type

```
clear screen;
```

in SQLPLUS

User Input in PLSQL

```
SET VERIFY OFF;
SET SERVEROUTPUT ON;

DECLARE
        A number := &x;
        B number := &y;
        C number := 0;
BEGIN
        C := A+B;
        DBMS_OUTPUT.PUT_LINE(C);
END;
/
```

Your Task

- Run “1.sql” file in SQLPLUS
- Observe the output Table
- Open a new sql file in Notepad++

Task 1

- Declare three variables (Say, A, B, C) with the data type of ‘taka’ attribute of ‘money’ table.
- Take input the values of A & B from user.
- In the BEGIN section, Sum A & B (C := A+B)
- If C is less than 170, then insert a new row in ‘money’ table with id = 7, name = ‘A’ and taka = C+10
- If C is in between 170 and 210, then insert a new row in ‘money’ table with id = 7, name = ‘B’ and taka = C+30
- Otherwise, insert a new row in ‘money’ table with id = 7, name = ‘C’ and taka = C

Task 2

- Declare three variables (A, B, C) with the data type of ID, Name, Taka attribute of ‘money’ table.
- Take input the values of A, B, C from user.
- Insert same row for 5 times in ‘money’ table where the ID will be incremented each time.

OFFLINE

- Run ‘DB.sql’
- Create one question that requires the following topics (VARIABLES, IF ELSE/WHEN CASE)
- Solve the question using PL/SQL
- ALL code will be in a single sql file.
- There will be an ONLINE in the Next Lab.

Good Luck ☺