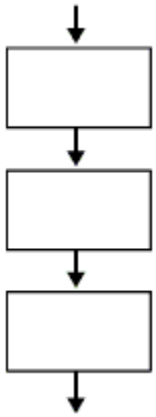
Oracle PL/SQL – Sequential Control Structures

codingcircuit23@gmail.com

Sequential control structures

- The sequence control structure simply executes a sequence of statements in the order in which they occur.



GOTO Statement

- A GOTO statement provides an unconditional jump from the GOTO to a labelled statement in the same sub program.
- GOTO provides the ability to jump through a program from one place to another.
- It is better to have a limited usage of GOTO in program code.
- When a jump is proposed using GOTO , it is associated with an appropriate label.
- The GOTO statement branches to a label unconditionally.
- The label must be unique within its scope and must precede an executable statement.
- When executed, the GOTO statement transfers the control to the labelled statements or block.
- The labelled statement or block can be down or up in the sequence of statements.
- The labels are enclosed between angular brackets << and >>

Syntax:

```
GOTO label_name;
..
..
<< label >>
statement;
```

Example: 1

```
DECLARE
num NUMBER(2) := 1;
BEGIN
      LOOP
               num := num+1;
               IF num > 5 THEN
                   GOTO LBL_ENDOFLOOP;
               END IF;
               DBMS_OUTPUT.PUT_LINE('value = ' || num);
      END LOOP;
      <<LBL_ENDOFLOOP>> -- label
      DBMS_OUTPUT.PUT_LINE('OK');
END;
/
```

Output :

```
value = 2
value = 3
value = 4
value = 5
OK

PL/SQL procedure successfully completed
```

GOTO Statement in PL/SQL imposes the following restrictions. A GOTO statement cannot branch

- Into an IF statement, CASE statement, LOOP statement or sub-block.
- From one IF statement clause to another or from one CASE statement WHEN clause to another.
- From an outer block into a sub-block (that is, an inner BEGIN-END block).
- Directly out of a sub program.
- From an exception handler back into the current BEGIN-END block.

NULL statement :

- PL/SQL uses NULL statement when there is nothing to execute.
- The NULL statement does nothing, and passes control to the next statement.

Syntax:

NULL;

Example: 2

```
DECLARE
        num NUMBER(3) := &num1;
BEGIN
        IF num > 5 THEN
        GOTO then_clause;
        num := 3;
        <<then_clause>>
        NULL;
        ELSE
        dbms_output.put_line(num);
        END IF;
END;
/
```

Output :

If input for num value is given as 7, then nothing will be printed.

```
PL/SQL procedure successfully completed
```