All Online Learning

www.allonlinelearning.com

PL/SQL Introduction

- The programs of PL/SQL are logical blocks that can contain any number of nested sub-blocks.
- Pl/SQL stands for "Procedural Language extension of SQL" that is used in Oracle.
- PL/SQL is integrated with Oracle database (since version 7).
- The functionalities of PL/SQL usually extended after each release of Oracle database

Syntax for declaring variable:

variable_name [CONSTANT] datatype [NOT NULL] [:= | DEFAULT initial_value]

Initializing Variables in PL/SQL

Every time you declare a variable, PL/SQL defines a default value NULL to it. If you want to initialize a variable with other value than NULL value**, you can do so during the declaration, by using any one of the following methods.**

- The DEFAULT keyword
- The assignment operator

Example:

```
counter binary_integer := 0;

greetings varchar2(20) DEFAULT 'Hello World';
```

You can also specify NOT NULL constraint to avoid NULL value. If you specify the NOT NULL constraint, you must assign an initial value for that variable.

Example of initilizing variable

```
1.  DECLARE
2.  a integer := 30;
3.  b integer := 40;
4.  c integer;
5.  f real;
6.  BEGIN
7.  c := a + b;
8.  dbms_output.put_line('Value of c: ' || c);
9.  f := 100.0/3.0;
10. dbms_output.put_line('Value of f: ' || f);
11. END;
```

After the execution, this will produce the following result:

1. Value of c: 70
2. Value of f: 33.33333333333333333

PL/SQL procedure successfully completed.

Variable Scope in PL/SQL:

PL/SQL allows nesting of blocks.

A program block can contain another inner block. If you declare a variable within an inner block, it is not accessible to an outer block.

There are two types of variable scope:

- **Local Variable**: Local variables are the inner block variables which are not accessible to outer blocks.
- **Global Variable:** Global variables are declared in outermost block.

Example of Local and Global variables

```
1.  DECLARE
2.   -- Global variables
3.     num1 number := 95;
4.     num2 number := 85;
5.  BEGIN
6.     dbms_output.put_line('Outer Variable num1: ' || num1);
7.     dbms_output.put_line('Outer Variable num2: ' || num2);
8.     DECLARE
9.        -- Local variables
10.       num1 number := 195;
11.       num2 number := 185;
12.    BEGIN
13.       dbms_output.put_line('Inner Variable num1: ' || num1);
14.       dbms_output.put_line('Inner Variable num2: ' || num2);
15.    END;
16. END;
17. /
```

After the execution, this will produce the following result:

1. Outer Variable num1: 95
2. Outer Variable num2: 85
3. Inner Variable num1: 195
4. Inner Variable num2: 185

Example of PL/SQL constant

Let's take an example to explain it well:

```
1.  DECLARE
2.     -- constant declaration
3.     pi constant number := 3.141592654;
4.     -- other declarations
5.     radius number(5,2);
6.     dia number(5,2);
7.     circumference number(7, 2);
8.     area number (10, 2);
9.  BEGIN
10.    -- processing
11.    radius := 9.5;
12.    dia := radius * 2;
13.    circumference := 2.0 * pi * radius;
14.    area := pi * radius * radius;
15.  -- output
16.    dbms_output.put_line('Radius: ' || radius);
17.    dbms_output.put_line('Diameter: ' || dia);
18.    dbms_output.put_line('Circumference: ' || circumference);
19.    dbms_output.put_line('Area: ' || area);
20. END;
21. /
```

After the execution of the above code at SQL prompt, it will produce the following result:.

```
1.  Radius: 9.5
2.  Diameter: 19
3.  Circumference: 59.69
4.  Area: 283.53
```

Pl/SQL procedure successfully completed.

Example of PL/SQL If Statement

Let's take an example to see the whole concept:

```
1.  DECLARE
2.    a number(3) := 500;
3.  BEGIN
4.    -- check the boolean condition using if statement
5.    IF( a < 20 ) THEN
6.       -- if condition is true then print the following
7.       dbms_output.put_line('a is less than 20 ' );
8.    ELSE
9.       dbms_output.put_line('a is not less than 20 ' );
10.   END IF;
11.   dbms_output.put_line('value of a is : ' || a);
12. END;
```

After the execution of the above code in SQL prompt, you will get the following result:

```
a is not less than 20
value of a is : 500
PL/SQL procedure successfully completed.
```

Example of PL/SQL EXIT Loop

Let's take a simple example to explain it well:

```
1.  DECLARE
2.  i NUMBER := 1;
3.  BEGIN
4.  LOOP
5.  EXIT WHEN i>10;
6.  DBMS_OUTPUT.PUT_LINE(i);
7.  i := i+1;
8.  END LOOP;
9.  END;
```

After the execution of the above code, you will get the following result:

```
1
2
3
4
5
6
7
8
9
10
```