04/03/2022

Ex.No: 9

PL/SQL CONDITIONAL AND ITERATIVE STATEMENTS

AIM:

To write PL/SQL programs using conditional and iterative statements.

PROCEDURE:

Steps to write and execute PL/SQL

*. As we want output of PL/SQL program on screen, before starting writing anything type.

```
SQL > SET SERVEROUTPUT ON
```

*. To write program, use Notepad through Oracle using ED command

```
SQL> ED ProName.
```

*. Type the program save and exit.

*. To run the program

```
SQL >@ ProName
```

BASIC SYNTAX OF PL/SQL:

```
DECLARE
/* variables can be declared here */
BEGIN
/* Executable statements can be written here */
EXCEPTION
/* Error handlers can be written here */
END;
```

Decision Making with If statement :

Syntax:

```
If(Test_condition) then
set of statements
else
set of statements
End if;
```

For Nested IF-ELSE Statement we can use IF--ELSIF - ELSE as follows:

```
If(Test_condition) then
set of statements
Elsif (condition)
set of statements
End if ;
```

LOOPING STATEMENTS :

*. For executing the set of statements repeatedly we use loops.

*. Oracle supports looping statements like GOTO, FOR, WHILE & LOOP.

GOTO :

```
<<LaBEL>>
set of statements
Goto label;
```

FOR LOOP:

```
For<var> in [Reverse] <INI_value> . . . . <End_value>
set of statements.
End loop;
```

WHILE LOOP:

```
While (condition) loop
set of statements
End loop;
```

09/03/2022

Ex.NO.10

PL|SQL PROCEDURES ON SAMPLE EXERCISES

AIM:

To write PL/SQL programs using procedures.

PROCEDURE:

Basic syntax of PL|SQL:

```
DECLARE
/*variables can be declared here*/
BEGIN
/* Executable statements can be written here */
EXCEPTION
/* Error handlers can be written here */
END;
```

SYNTAX:

```
CREATE [OR REPLACE] PROCEDURE procedure_name
[(Parameter [, parameter])]
IS
[declaration_section]
BEGIN
executable_section
[EXCEPTION
 exception_section]
END[procedure_name];
```

11/03/2022

EX.NO: 11

PL/SQL FUNCTIONS

AIM:

To write PL/SQL programs using functions

PROCEDURE:

Syntax of PL/SQL:

```
DECLARE
/* variables can be declared here*/
BEGIN
/* Executable statements can be written here*/
EXCEPTION
/* Error handlers can be written here */
END;
```

Steps to write and Execute PL/SQL:

*. As we want output of PL/SQL program on screen, before starting writing anything type.

```
SQL> SET serveroutput on
```

*. To write program, use Notepad through Oracle using ED. command.

```
SQL> ED ProName
```

*. Type the program save and exit

*. To run the program

```
SQL> @ Proname
```

SYNTAX FOR FUNCTIONS:

```
CREATE [OR REPLACE] FUNCTION function_name
[(parameter [, parameter])]
RETURN return-datatype
IS|AS
[declaration_section]
BEGIN
executable_section
[EXCEPTION
    Exception_section]
END [function_name];
```

16.03.2022

EX.NO: 12

PL/SQL CURSORS

AIM:

To create a database using implicit and explicit cursors.

PROCEDURE:

Cursor:

A cursor is a pointer to the context area. PL/SQL controls the context area through a cursor. A cursor holds the rows (one or more) returned by a SQL statement. The set of rows the cursor holds is referred to as the active set.

Types of Cursors:

There are two types of cursors:

1] Implicit cursors

2] Explicit cursors.

IMPLICIT CURSORS:

⇒ Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement.

⇒ Programmers cannot control the implicit cursors and the information in it.

⇒ Implicit cursors has attributes such as %Found, %ISOPEN, %Notfound and %Rowcount. SQL cursor has additional attributes such as %Bulk_rowcount and %Bulk_Exceptions, to use with FORALL statement.

EXPLICIT CURSORS:

⇒ Explicit cursors are programmer-defined cursors for gaining more control over the context area.

⇒ It should be defined in the declaration section of the PL/SQL Block.

⇒ It is created on a SELECT statement which returns more than one row.

Syntax:

```
CURSOR cursor_name IS select_statement;
```