Assignment No.	**6**
Title	Write a PL/SQL block to implement all types of cursors
Roll No.	
Class	T.E. (C.E.)
Date	
Subject	**Database Management System Laboratory**
Signature	

Aim :	Write a PL/SQL block of code using Cursors: (All types: Implicit, Explicit Cursor etc). Write a PL/SQL block of code using parameterized Cursor, that will merge the data available in the newly created table N_RollCall with the data available in

	the table O_RollCall. If the data in the first table already exist in the second table then that data should be skipped.

Objectives **:** Learn the concept of PL/SQL.

Theory **:- CURSOR:-**

For the processing of any SQL statement, database needs to allocate memory. This memory is called context area. The context area is a part of PGA (Process global area) and is allocated on the oracle server.

cursor is associated with this work area used by ORACLE, for multi row queries. A cursor is a handle or pointer to the context area .The cursor allows to process contents in the context area row by row. There are two types of cursors.

1) Implicit cursor:-Implicit cursors are defined by ORACLE implicitly. ORACLE defines implicit cursor for every DML statements.
2) Explicit cursor:-These are user-defined cursors which are defined in the declaration section of the PL/SQL block. There are four steps in which the explicit cursor is processed.
    1) Declaring a cursor
    2) Opening a cursor
    3) Fetching rows from an opened cursor
    4) Closing cursor

General syntax for CURSOR:-

```
DECLARE
        Cursor cursor_name IS select_statement or query;
BEGIN
        Open cursor_name;
Fetch cursor_name into list_of_variables;
       Close cursor_name;
END;
```

Where

1) Cursor_name:-is the name of the cursor.
2) Select_statement:-is the query that defines the set of rows to be processed by the cursor.
3) Open cursor_name:-open the cursor that has been previously declared.
   When cursor is opened following things happen
   i) The active set pointer is set to the first row.
   ii) The value of the binding variables are examined.
4) Fetch statement is used to retrieve a row from the selected rows, one at a time, into PL/SQL variables.
5) Close cursor_name:-When all of cursor rows have been retrieved, the cursor should be closed.

Explicit cursor attributes:-

Following are the cursor attributes

1. %FOUND: - This is Boolean attribute. It returns TRUE if the previous fetch returns a row and false if it doesn't.

2. %NOTFOUND:-If fetch returns a row it returns FALSE and TRUE if it doesn't. This is often used as the exit condition for the fetch loop;

3. %ISOPEN:-This attribute is used to determine whether or not the associated cursor is open. If so it returns TRUE otherwise FALSE.

4. %ROWCOUNT:-This numeric attribute returns a number of rows fetched by the cursor.

Cursor Fetch Loops

1) **Simple Loop**

Syntax:-

```
LOOP

        Fetch cursorname into list of variables;

EXIT WHEN cursorname%NOTFOUND

        Sequence_of_statements;

END LOOP;
```

2) **WHILE Loop**

Syntax:-

```
FETCH cursorname INTO list of variables;

WHILE cursorname%FOUND LOOP

        Sequence_of_statements;

        FETCH cursorname INTO list of variables;

END LOOP;
```

3) **Cursor FOR Loop**

Syntax:

```
FOR variable_name IN cursorname LOOP

        -- an implicit fetch is done here.

        -- cursorname%NOTFOUND is also implicitly checked.

        -- process the fetch records.

        Sequence_of_statements;

END LOOP;
```

There are two important things to note about :-

i) Variable_name is not declared in the DECLARE section. This variable is implicitly declared by the PL/SQL compiler.
ii) Type of this variable is cursorname%ROWTYPE.

Implicit Cursors

PL/SQL issues an implicit cursor whenever you execute a SQL statement directly in your code, as long as that code does not employ an explicit cursor. It is called an "implicit" cursor because you, the developer, do not explicitly declare a cursor for the SQL statement.

If you use an implicit cursor, Oracle performs the open, fetches, and close for you automatically; these actions are outside of your programmatic control. You can, however, obtain information about the most recently executed SQL statement by examining the values in the implicit SQL cursor attributes.

PL/SQL employs an implicit cursor for each UPDATE, DELETE, or INSERT statement you execute in a program. You cannot, in other words, execute these statements within an explicit

cursor, even if you want to. You have a choice between using an implicit or explicit cursor only when you execute a single-row SELECT statement (a SELECT that returns only one row).

In the following UPDATE statement, which gives everyone in the company a 10% raise, PL/SQL creates an implicit cursor to identify the set of rows in the table which would be affected by the update:

```
UPDATE employee
   SET salary = salary * 1.1;
```

The following single-row query calculates and returns the total salary for a department. Once again, PL/SQL creates an implicit cursor for this statement:

```
SELECT SUM (salary) INTO department_total
  FROM employee
 WHERE department_number = 10;
```

If you have a SELECT statement that returns more than one row, you must use an explicit cursor for that query and then process the rows returned one at a time. PL/SQL does not yet support any kind of array interface between a database table and a composite PL/SQL datatype such as a PL/SQL table.

Conclusion: *Thus we have studied Importance & implementation of CURSOR.*