Group A

Assignment No 7

Title of the Assignment: Cursors: (All types: Implicit, Explicit, Cursor FOR Loop, Parameterized Cursor)

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.

Objective: To understand the concept of cursors and its types.

Outcome: Students will be able to learn and understand All types: Implicit, Explicit, Cursor FOR Loop, and Parameterized Cursor.

Theory:

❖ PL/SQL Cursor

- When an SQL statement is processed, Oracle creates a memory area known as context area.
- A cursor is a pointer to this context area.
- It contains all information needed for processing the statement.
- In PL/SQL, the context area is controlled by Cursor.
- A cursor contains information on a select statement and the rows of data accessed by it.
- A cursor is used to referred to a program to fetch and process the rows returned by the SQL statement, one at a time.

o **There are two types of cursors:**

1. Implicit Cursors
2. Explicit Cursors

1. PL/SQL Implicit Cursors

The implicit cursors are automatically generated by Oracle while an SQL statement is executed, if you don't use an explicit cursor for the statement.

These are created by default to process the statements when DML statements like INSERT, UPDATE, DELETE etc. are executed.

Oracle provides some attributes known as Implicit cursor's attributes to check the status of DML operations. Some of them are: %FOUND, %NOTFOUND, %ROWCOUNT and %ISOPEN.

For example: When you execute the SQL statements like INSERT, UPDATE, DELETE then the cursor attributes tell whether any rows are affected and how many have been affected. If you run a SELECT INTO statement in PL/SQL block, the implicit cursor attribute can be used to find out whether any row has been returned by the SELECT statement. It will return an error if there no data is selected.
The following table specifies the status of the cursor with each of its attribute.

Attribute	Description
%FOUND	Its return value is TRUE if DML statements like INSERT, DELETE and UPDATE affect at least one row or more rows or a SELECT INTO statement returned one or more rows. Otherwise it returns FALSE.
%NOTFOUND	Its return value is TRUE if DML statements like INSERT, DELETE and UPDATE affect no row, or a SELECT INTO statement return no rows. Otherwise it returns FALSE. It is a just opposite of %FOUND.
%ISOPEN	It always returns FALSE for implicit cursors, because the SQL cursor is automatically closed after executing its associated SQL statements.
%ROWCOUNT	It returns the number of rows affected by DML statements like INSERT, DELETE, and UPDATE or returned by a SELECT INTO statement.

2. PL/SQL Explicit Cursors

The Explicit cursors are defined by the programmers to gain more control over the context area. These cursors 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 of explicit cursor

Following is the syntax to create an explicit cursor:

```
CURSOR cursor_name IS select_statement;;
```

Steps:

You must follow these steps while working with an explicit cursor.

1. Declare the cursor to initialize in the memory.
2. Open the cursor to allocate memory.
3. Fetch the cursor to retrieve data.
4. Close the cursor to release allocated memory.

1) Declare the cursor:

It defines the cursor with a name and the associated SELECT statement.

Syntax for explicit cursor declaration

```
CURSOR name IS
 SELECT statement;
```

2) Open the cursor:

It is used to allocate memory for the cursor and make it easy to fetch the rows returned by the SQL statements into it.

Syntax for cursor open:

```
OPEN cursor_name;
```

3) Fetch the cursor:

It is used to access one row at a time. You can fetch rows from the above-opened cursor as follows:

Syntax for cursor fetch:

```
FETCH cursor_name INTO variable_list;
```

4) Close the cursor:

It is used to release the allocated memory. The following syntax is used to close the above-opened cursors.

Syntax for cursor close:

```
Close cursor_name;
```

Program

```
create procedure n1(in rno1 int)
begin
declare rno2 int;
declare exit_cond boolean;
declare c1 cursor for select rollno from o_rollcall where rollno>rno1;
declare continue handler for not found set exit_cond=TRUE;
open c1;
l1:loop
fetch c1 into rno2;
if not exists(select * from n_rollcall where rollno=rno2)then
insert into n_rollcall select * from o_rollcall where rollno=rno2;
end if;
if exit_cond then
close c1;
leave l1;
end if;
end loop l1;
end
//
```

```
mysql> create table o_rollcall(rollno int,name varchar(90),address varchar(90));
Query OK, 0 rows affected (0.53 sec)

mysql> create table n_rollcall(rollno int,name varchar(90),address varchar(90));
Query OK, 0 rows affected (0.50 sec)

mysql> insert into o_rollcall values(1,'komal','abc');
Query OK, 1 row affected (0.07 sec)

mysql> insert into o_rollcall values(2,'raj','pbc');
Query OK, 1 row affected (0.06 sec)

mysql> insert into o_rollcall values(3,'rudra','pune');
Query OK, 1 row affected (0.07 sec)

mysql> insert into n_rollcall values(1,'komal','abc');
Query OK, 1 row affected (0.06 sec)

mysql> insert into n_rollcall values(4,'lina','nashik');
Query OK, 1 row affected (0.08 sec)

mysql> insert into n_rollcall values(5,'sheetal','aaldi');
Query OK, 1 row affected (0.06 sec)
```

```
mysql> select * from o_rollcall;
+--------+-------+---------+
| rollno | name  | address |
+--------+-------+---------+
|      1 | komal | abc     |
|      2 | raj   | pbc     |
|      3 | rudra | pune    |
+--------+-------+---------+
3 rows in set (0.00 sec)

mysql> select * from n_rollcall;
+--------+---------+---------+
| rollno | name    | address |
+--------+---------+---------+
|      1 | komal   | abc     |
|      4 | lina    | nashik  |
|      5 | sheetal | aaldi   |
+--------+---------+---------+

////////////////OUTPUT///////////////
mysql> delimiter ;
mysql> call n1(1);
Query OK, 0 rows affected (0.15 sec)

mysql> select * from n_rollcall;
+--------+---------+---------+
| rollno | name    | address |
+--------+---------+---------+
|      1 | komal   | abc     |
|      4 | lina    | nashik  |
|      5 | sheetal | aaldi   |
|      2 | raj     | pbc     |
|      3 | rudra   | pune    |
+--------+---------+---------+
5 rows in set (0.00 sec)

Mysql> select * from o_rollcall;
+--------+-------+---------+
| rollno | name  | address |
+--------+-------+---------+
|      1 | komal | abc     |
|      2 | raj   | pbc     |
|      3 | rudra | pune    |
+--------+-------+---------+
3 rows in set (0.00 sec)
```

Conclusion: Performed implementation of cursors in MYSQL successfully.

Viva Questions:

- What is cursor? Explain with example?
- Write advantages of cursor?
- Write application of cursor?
- Explain the types of cursor?
- Explain the explicit cursor with example**?**

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator:	