Name : Anushka M Mulay

Roll No 35040

TE IT(A)

**Assignment No. 7**

* **Title:** To study cursors in MySQL.
* **Problem statement:** Write a PL/SQL block to implement all types of cursor.
* **Requirements:** MySQL command line client.
* **Prerequisites:** Basic of MySQL DML commands.
* **Theory:**
* **Cursor :** A cursor is a temporary work area created in memory when a SQL statement is executed. A cursor contains information of select statement and the rows of data accessed by it. A cursor can hold more than one row, but can process only one row at a time. Used only with stored procedure.
* **Pupose** : Cursors are used by database programmers to process individual rows returned by database system queries. Cursors enable manipulation of whole result sets at once. In this scenario, a cursor enables the sequential processing of rows in a result set.
* **Types**

There are two major different types of cursors:

* **Implicit cursor**

When you execute DML statement like DELETE, INSERT, UPDATE and SELECT statements, implicit statements are created to process these statements.

Oracle provides few attributes called as implicit cursor attributes to check the status of DML operations.

**Example –**

Consider the PL/SQL block that uses implicit cursor attributes:

DECLARE var\_rows number (5);

BEGIN

UPDATE employee SET salary = salary + 1000;

IF SQL%NOTFOUND THEN

Dbms\_output.put\_line(‘none of the salaries are updated’);

ELSIF SQL%FOUND THEN

Var\_rows := SQL%ROWCOUNT;

Dbms\_output.put\_line(‘salaries for’ || var\_rows || ‘employees are updated’);

END IF;

END;

In the above PL/SQL block, the salaries of all the employees in the ‘employee’ table are updated. If none of the employee’s salary is updated we get a message ‘none of the salaries are updated’, else we get, ‘salaries for 1000 employees are updated’.

* **Explicit cursor**

They are created when you are executing a SELECT statement that returns more than one rows. When you fetch a row, the current row position moves to next row.

Both types of cursor have same functionality but they differ in the way they are accessed.

Cursor declaration syntax – CURSOR cursor\_name IS select\_statement;

* Declaring a cursor

DECLARE

CURSOR emp\_cur IS

SELECT \* FROM emp\_tbl WHERE salary > 5000;

* Accessing the cursor

We can access the cursor in the execution section of the PL/SQL program.

1. Open

In order to use an explicit cursor, you must open it.

Syntax – OPEN cursor\_name;

1. Fetch

Besides opening the cursor, we also have to grab the results of the SELECT statement one by one.

Syntax – FETCH cursor\_name INTO record\_name;

1. Close

Syntax – CLOSE cursor\_name;

**General syntax –**

Declare

Variables;

Create a cursor;

Begin

Open cursor;

Fetch cursor;

Process the records;

Close cursor;

End;

**What is continue handler? Why is it used?**

* When an error occurs inside a stored procedure, it is important to handle it appropriately, such as continuing or exiting the current code block’s execution, and issuing a meaningful error message.
* This is relevant within the context of cursors and is **used** to control what happens when a cursor reaches the end of a data set.

**Not found continue handler**

This is relevant within the context of cursors and is used to control what happens when a cursor reaches the end of a data set. If no more rows are available, a no data condition occurs with sqlstate value '02000'. To detect this condition, you can set up a handler for it or for a not found condition.

Syntax –

* Declare continue handler for not found
* Begin
* body of handler
* end;

Example – Cursor with continue handler to display name and credits of student.

> delimiter //

> create procedure cursor1()

-> begin

-> declare done int default 0;

-> declare i int;

-> declare n varchar(10);

-> declare cur1 cursor for select tot\_credits, sname from student;

-> declare continue handler for not found set done=1;

-> open cursor1;

-> loop1 : loop fetch cursor1 into i,n;

-> if done then leave loop1;

-> end if;

-> select i,n;

-> end loop;

-> close cursor1;

-> end;//

**Conclusion**: Hence, we have studied and implemented cursor and types of cursors in MySQL.