Assignment - VI

Title :- Write a PL/SQL program using parameterized cursor, that will merge the data available in newly created table.

Problem Statement :-

Write a PL/SQL block of code using parameterized cursor that will merge the data available in newly created table.

objective :-

- To understand the types of cursors
- To understand how to use cursor with PL/SQL block.

S/W & H/W requirement :-

SQL, 64 bit OS Fedora.

Outcome :-

The students will be able to

- Implement PL/SQL block code
- Implement type of cursor.

Theory :-

PL/SQL :-

PL/SQL offers a set of procedural commands (If statement, loops) Organized within blocks,

Cursors :-

A cursor is a temparory work area created in the system memory. when a SQL statement is executed. A cursor contains info on a select statement & rows of data accessed by it.

This tempory work area is used to store data, retrived from the data-base & manipulate the data. A cursor can hold more than one row but can process. Only one row at a time. The set of rows the cursor holds is called active set.

Two types of cursors

1) Implicit cursor
2) Explicit Cursor.

Implicit cursor :-

These are created by default when DML statement like Insert, Update & delete statements are executed they are also created when a select statement that returns just one row is executed.

Explicit cursor :-

They must be created when you are executing a select statement that returns ~~value~~ more than one row. Even though the cursor stores multiple records only one recor can be processed at a time which i called as cursor row.

When you fetch the row the curren row position moves to the next row. Both implicit & explicit cursors have the same functionality but they differ in the way they are accessed.

Syntax :-

Declare cursorname cursor for select -column name from table.
Fetch cursor into variable as :-
Fetch cursorname into variable.

Test cases :-

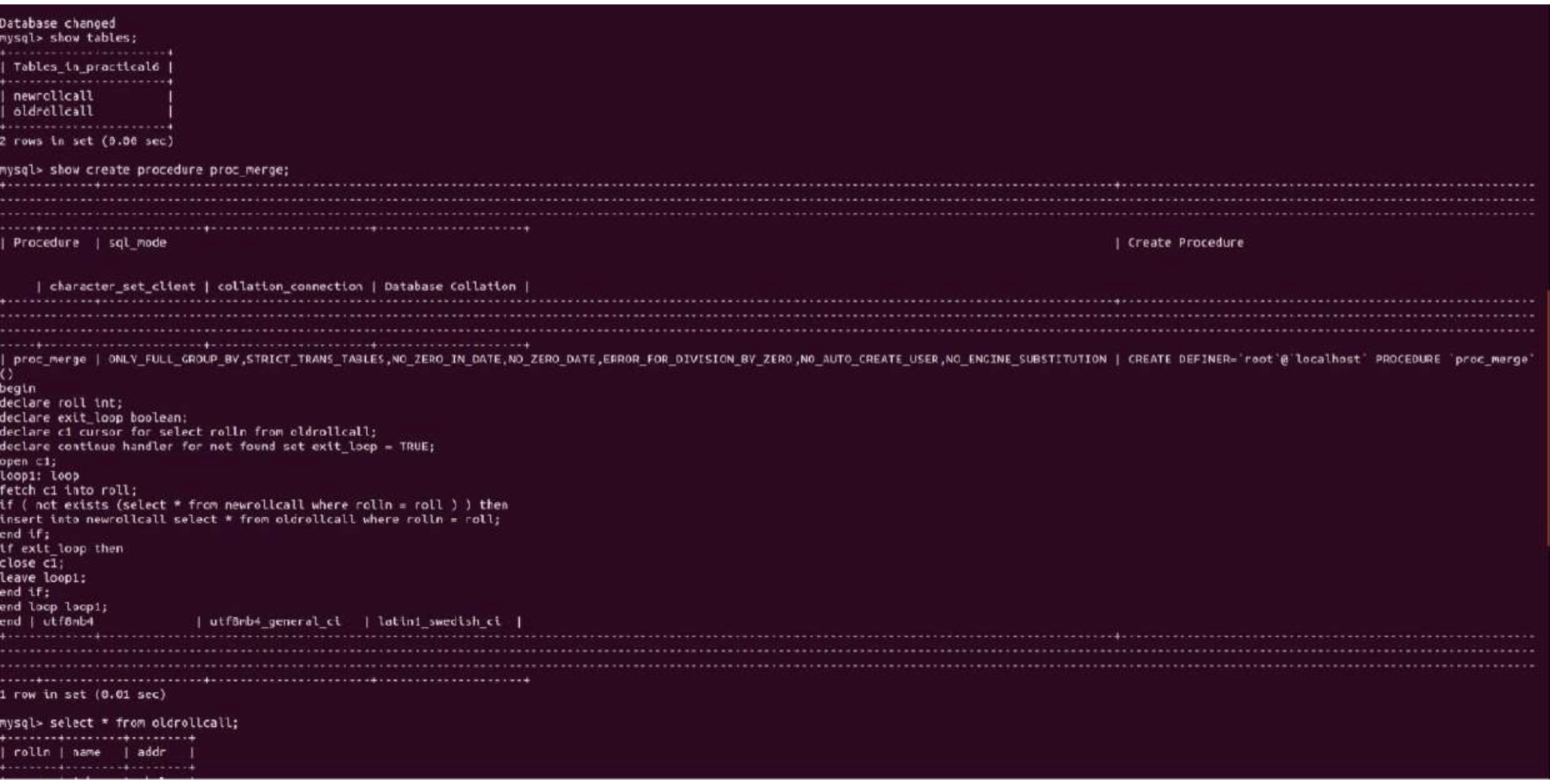
operation	I/P	O/P	Result
call procedure and update status	old table:- 1, 'Raj' 2, 'john' call proc-merge();	New Table 1, 'Raj' 2, 'John'	success
	old table 1, 'mark' New Table 2, 'Aditya' call proc_merge();	New Table 2, 'Aditya' 2, 'mark'	Succ

Conclusion :-

In this assignment we learn to write program using parameterized cursor that will merge the data available in newly created table.

Source Code:

```
Create procedure proc_merge()
begin
declare roll int;
declare exit_loop boolean;
declare c1 cursor for select rolln from oldRollCall;
declare continue handler for not found set exit_loop = TRUE;
open c1;
loop1: loop
fetch c1 into roll;
if ( not exists (select * from newRollCall where rolln = roll ) ) then
insert into newRollCall select * from oldRollCall where rolln = roll;
end if;
if exit_loop then
close c1;
leave loop1;
end if;
end loop loop1;
end
```

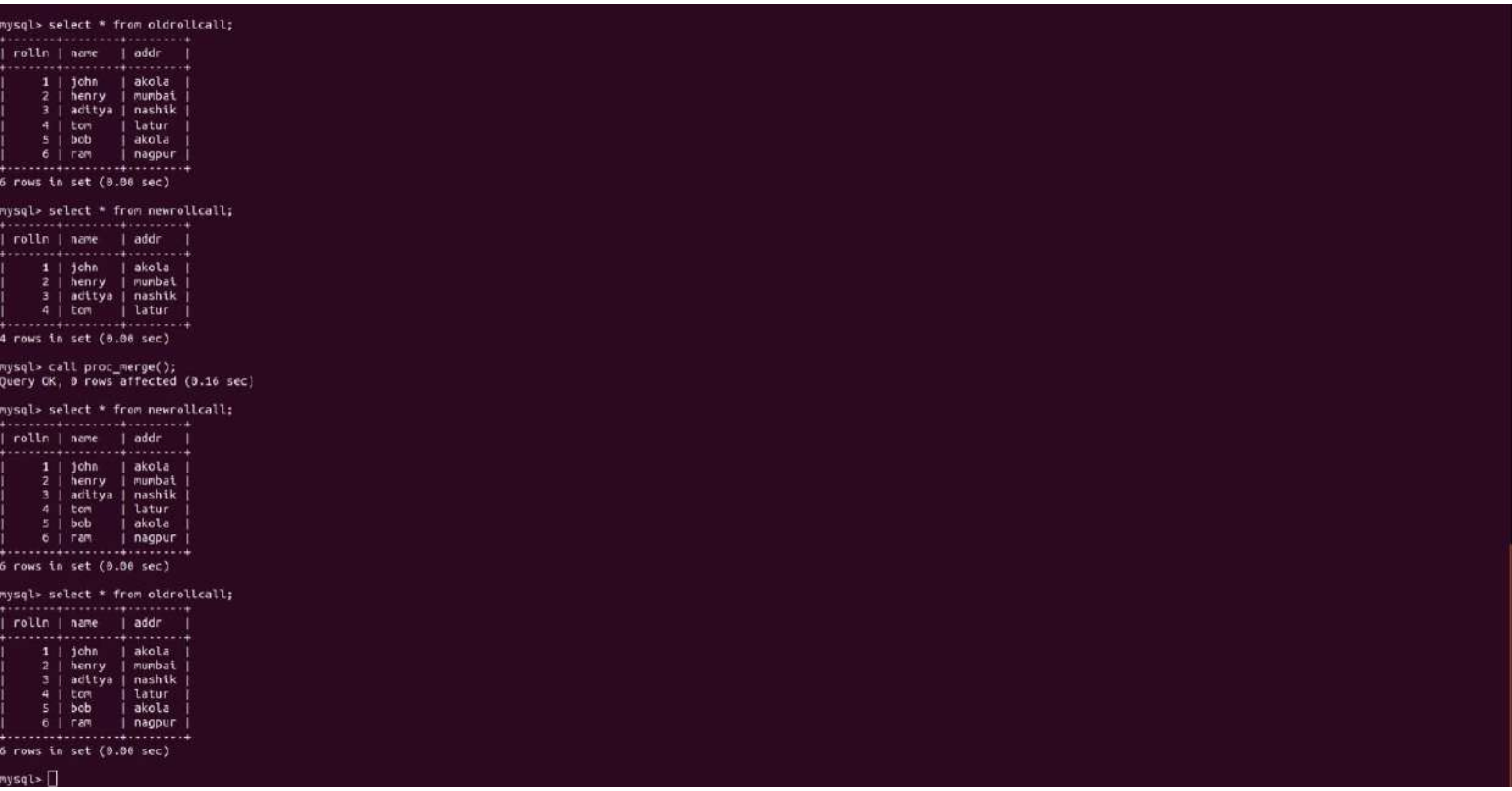


```
Database changed
mysql> show tables;
+----------------------+
| Tables_in_practical6 |
+----------------------+
| newrollcall          |
| oldrollcall          |
+----------------------+
2 rows in set (0.00 sec)

mysql> show create procedure proc_merge;
+------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------+----------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----+----------------------+----------------------+--------------------+
| Procedure  | sql_mode                                                                                                                                                         | Create Procedure


     | character_set_client | collation_connection | Database Collation |
+------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------+----------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----+----------------------+----------------------+--------------------+
| proc_merge | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION | CREATE DEFINER=`root`@`localhost` PROCEDURE `proc_merge`
()
begin
declare roll int;
declare exit_loop boolean;
declare c1 cursor for select rolln from oldrollcall;
declare continue handler for not found set exit_loop = TRUE;
open c1;
loop1: loop
fetch c1 into roll;
if ( not exists (select * from newrollcall where rolln = roll ) ) then
insert into newrollcall select * from oldrollcall where rolln = roll;
end if;
if exit_loop then
close c1;
leave loop1;
end if;
end loop loop1;
end | utf8mb4              | utf8mb4_general_ci   | latin1_swedish_ci  |
+------------+------------------------------------------------------------------------------------------------------------------------------------------------------------------+----------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----+----------------------+----------------------+--------------------+
1 row in set (0.01 sec)

mysql> select * from oldrollcall;
+-------+--------+--------+
| rolln | name   | addr   |
+-------+--------+--------+
```



```
mysql> select * from oldrollcall;
+-------+--------+--------+
| rolln | name   | addr   |
+-------+--------+--------+
|     1 | john   | akola  |
|     2 | henry  | mumbai |
|     3 | aditya | nashik |
|     4 | tom    | latur  |
|     5 | bob    | akola  |
|     6 | ram    | nagpur |
+-------+--------+--------+
6 rows in set (0.00 sec)

mysql> select * from newrollcall;
+-------+--------+--------+
| rolln | name   | addr   |
+-------+--------+--------+
|     1 | john   | akola  |
|     2 | henry  | mumbai |
|     3 | aditya | nashik |
|     4 | tom    | latur  |
+-------+--------+--------+
4 rows in set (0.00 sec)

mysql> call proc_merge();
Query OK, 0 rows affected (0.16 sec)

mysql> select * from newrollcall;
+-------+--------+--------+
| rolln | name   | addr   |
+-------+--------+--------+
|     1 | john   | akola  |
|     2 | henry  | mumbai |
|     3 | aditya | nashik |
|     4 | tom    | latur  |
|     5 | bob    | akola  |
|     6 | ram    | nagpur |
+-------+--------+--------+
6 rows in set (0.00 sec)

mysql> select * from oldrollcall;
+-------+--------+--------+
| rolln | name   | addr   |
+-------+--------+--------+
|     1 | john   | akola  |
|     2 | henry  | mumbai |
|     3 | aditya | nashik |
|     4 | tom    | latur  |
|     5 | bob    | akola  |
|     6 | ram    | nagpur |
+-------+--------+--------+
6 rows in set (0.00 sec)

mysql> 
```