Name: Sudhir Singh
Class: TE - A
Batch: A3
Roll No: 23152
Subject: SL-I (A)

Lab Assignment 6

Implicit Cursor

1. The bank manager has decided to activate all those accounts which were previously marked as inactive for performing no transaction in last 365 days. Write a PL/SQ block (using implicit cursor) to update the status of account, display an approximate message based on the no. of rows affected by the update.
(Use of %FOUND, %NOTFOUND, %ROWCOUNT)

```
mysql> USE college_bank;
Database changed
mysql> -- 1: Implicit Cursor(Activate Inactive Accounts)
mysql> CREATE TABLE accounts(
    ->     acc_no INT PRIMARY KEY,
    ->     acc_status VARCHAR(10),
    ->     last_transaction DATE
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO accounts VALUES
    -> (101, 'Inactive', '2024-08-01'),
    -> (102, 'Active', '2025-03-10'),
    -> (103, 'Inactive', '2023-06-10'),
    -> (104, 'Inactive', '2024-09-01');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from accounts;
+--------+------------+------------------+
| acc_no | acc_status | last_transaction |
+--------+------------+------------------+
|    101 | Inactive   | 2024-08-01       |
|    102 | Active     | 2025-03-10       |
|    103 | Inactive   | 2023-06-10       |
|    104 | Inactive   | 2024-09-01       |
+--------+------------+------------------+
4 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE ActivateAccounts()
    -> BEGIN
    ->     DECLARE rows_affected INT DEFAULT 0;
    ->
    ->     UPDATE accounts SET acc_status = 'Active' WHERE acc_status='Inactive' AND last_transaction <= DATE_SUB(CURDATE(), INTERVAL 365 DAY);
    ->
    ->     SET rows_affected = ROW_COUNT();
    ->
    ->     IF rows_affected > 0 THEN
    ->         SELECT CONCAT(rows_affected, ' accounts activated') AS Message;
    ->     ELSE
    ->         SELECT 'No accounts were activated' AS Message;
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL ActivateAccounts();
+----------------------+
| Message              |
+----------------------+
| 3 accounts activated |
+----------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> SET SQL_SAFE_UPDATES = 1;
Query OK, 0 rows affected (0.00 sec)

mysql> -- 2.Explicit Cursor - Salary Increment
mysql> CREATE TABLE emp(
    ->     e_no INT PRIMARY KEY,
    ->     salary DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

EXPLICIT CURSOR:

2. Organization has decided to increase the salary of employees by 10% of existing salary, who are having salary less than average salary of organization, Whenever such salary updates takes place, a record for the same is maintained in the increment_salary table.

EMP (E_no , Salary) increment_salary(E_no , Salary)

```
mysql> -- 2.Explicit Cursor - Salary Increment
mysql> CREATE TABLE emp(
    ->     e_no INT PRIMARY KEY,
    ->     salary DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE increment_salary(
    ->     e_no INT,
    ->     salary DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO emp VALUES
    -> (1, 20000),
    -> (2, 40000),
    -> (3, 30000),
    -> (4, 50000);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from emp;
+------+----------+
| e_no | salary   |
+------+----------+
|    1 | 20000.00 |
|    2 | 40000.00 |
|    3 | 30000.00 |
|    4 | 50000.00 |
+------+----------+
4 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE IncreaseSalary()
    -> BEGIN
    ->     DECLARE done INT DEFAULT FALSE;
    ->     DECLARE v_e_no INT;
    ->     DECLARE v_salary DECIMAL(10,2);
    ->     DECLARE avg_salary DECIMAL(10,2);
    ->
    ->     DECLARE cur CURSOR FOR SELECT e_no, salary FROM emp;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    ->
    ->     SELECT AVG(salary) INTO avg_salary FROM emp;
    ->     OPEN cur;
    ->     read_loop: LOOP
    ->         FETCH cur INTO v_e_no, v_salary;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         IF v_salary < avg_salary THEN
    ->             UPDATE emp SET salary = salary * 1.10 WHERE e_no = v_e_no;
    ->             INSERT INTO increment_salary VALUES(v_e_no, v_salary * 1.10);
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE cur;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL IncreaseSalary();
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> SELECT * FROM emp;
+------+-----------+
| e_no | salary    |
+------+-----------+
|    1 | 22000.00  |
|    2 | 40000.00  |
|    3 | 33000.00  |
|    4 | 50000.00  |
+------+-----------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM increment_salary;
+------+-----------+
| e_no | salary    |
+------+-----------+
|    1 | 22000.00  |
|    3 | 33000.00  |
+------+-----------+
2 rows in set (0.00 sec)

mysql> SET SQL_SAFE_UPDATES = 1;
Query OK, 0 rows affected (0.00 sec)
```

3. Write PL/SQL block using explicit cursor for following requirements:
College has decided to mark all those students detained (D) who are having attendance less than 75%. Whenever such update takes place, a record for the same is maintained in the D_Stud table.
create table
stud21(roll number(4), att number(4), status varchar(1));
create table d_stud(roll number(4), att number(4));

EXPLICIT CURSOR: Cursor for loop

6. Write PL/SQL block using explicit cursor: Cursor FOR Loop for following requirements:
College has decided to mark all those students detained (D) who are having attendance less than 75%. Whenever such update takes place, a record for the same is maintained in the D_Stud table.
create table
stud21(roll number(4), att number(4), status varchar(1));
create table d_stud(roll number(4), att number(4));

```
mysql> -- 3&6 Cursor FOR Loop - Detain Students
mysql> CREATE TABLE stud21(
    ->     roll INT PRIMARY KEY,
    ->     att DECIMAL(5,2),
    ->     status VARCHAR(1)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE d_stud(
    ->     roll INT,
    ->     att DECIMAL(5,2)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO stud21 VALUES
    -> (1, 80, 'P'),
    -> (2, 70, 'P'),
    -> (3, 60, 'P'),
    -> (4, 90, 'P');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

```
mysql> DELIMITER $$
mysql> CREATE PROCEDURE DetainStudents()
    -> BEGIN
    ->     DECLARE done INT DEFAULT FALSE;
    ->     DECLARE v_roll INT;
    ->     DECLARE v_att DECIMAL(5,2);
    ->
    ->     DECLARE cur CURSOR FOR SELECT roll, att FROM stud21;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    ->
    ->     OPEN cur;
    ->
    ->     read_loop: LOOP
    ->         FETCH cur INTO v_roll, v_att;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         IF v_att < 75 THEN
    ->             UPDATE stud21 SET status='D' WHERE roll = v_roll;
    ->             INSERT INTO d_stud VALUES(v_roll, v_att);
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE cur;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL DetainStudents();
Query OK, 0 rows affected (0.02 sec)

mysql> CALL DetainStudents();
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM stud21;
+------+-------+--------+
| roll | att   | status |
+------+-------+--------+
|    1 | 80.00 | P      |
|    2 | 70.00 | D      |
|    3 | 60.00 | D      |
|    4 | 90.00 | P      |
+------+-------+--------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM d_stud;
+------+-------+
| roll | att   |
+------+-------+
|    2 | 70.00 |
|    3 | 60.00 |
+------+-------+
2 rows in set (0.00 sec)

mysql> SET SQL_SAFE_UPDATES = 1;
Query OK, 0 rows affected (0.00 sec)
```

Parameterized Cursor

4. 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.

```
mysql> -- 4.Parameterized Cursor - Merge Tables
mysql> CREATE TABLE N_RollCall(roll INT PRIMARY KEY, name VARCHAR(50));
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE O_RollCall(roll INT PRIMARY KEY, name VARCHAR(50));
Query OK, 0 rows affected (0.02 sec)

mysql> INSERT INTO N_RollCall VALUES (1,'Sudhir'),(2,'Ayyub'),(3,'Saish');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> INSERT INTO O_RollCall VALUES (2,'Rahul'),(4,'Anuj');
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from N_RollCall;
+------+--------+
| roll | name   |
+------+--------+
|    1 | Sudhir |
|    2 | Ayyub  |
|    3 | Saish  |
+------+--------+
3 rows in set (0.00 sec)

mysql> select * from O_RollCall;
+------+-------+
| roll | name  |
+------+-------+
|    2 | Rahul |
|    4 | Anuj  |
+------+-------+
2 rows in set (0.00 sec)

mysql> DELIMITER $$
mysql> CREATE PROCEDURE MergeRollCall()
    -> BEGIN
    ->     DECLARE done INT DEFAULT FALSE;
    ->     DECLARE v_roll INT;
    ->     DECLARE v_name VARCHAR(50);
    ->
    ->     DECLARE cur CURSOR FOR SELECT roll, name FROM N_RollCall;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    ->
    ->     OPEN cur;
    ->     read_loop: LOOP
    ->         FETCH cur INTO v_roll, v_name;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         IF NOT EXISTS (SELECT 1 FROM O_RollCall WHERE roll = v_roll) THEN
    ->             INSERT INTO O_RollCall VALUES(v_roll, v_name);
    ->         END IF;
    ->     END LOOP;
    ->     CLOSE cur;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
```

```
mysql> SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL MergeRollCall();
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM O_RollCall;
+------+--------+
| roll | name   |
+------+--------+
|    1 | Sudhir |
|    2 | Rahul  |
|    3 | Saish  |
|    4 | Anuj   |
+------+--------+
4 rows in set (0.00 sec)

mysql> SET SQL_SAFE_UPDATES = 1;
Query OK, 0 rows affected (0.00 sec)
```

Parameterized Cursor

5. Write the PL/SQL block for following requirements using parameterized Cursor:
Consider table EMP(e_no, d_no, Salary), department wise average salary should be inserted into new
table dept_salary(d_no, Avg_salary)

```
mysql> -- 5: Parameterized Cursor - Department Average Salary
mysql> CREATE TABLE emp_dept(
    ->     e_no INT PRIMARY KEY,
    ->     d_no INT,
    ->     salary DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE dept_salary(
    ->     d_no INT PRIMARY KEY,
    ->     avg_salary DECIMAL(10,2)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO emp_dept VALUES
    -> (1,101,20000),(2,101,25000),(3,102,30000),(4,102,40000);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from emp_dept;
+------+------+----------+
| e_no | d_no | salary   |
+------+------+----------+
|    1 |  101 | 20000.00 |
|    2 |  101 | 25000.00 |
|    3 |  102 | 30000.00 |
|    4 |  102 | 40000.00 |
+------+------+----------+
4 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE DeptAvgSalary()
    -> BEGIN
    ->     DECLARE done INT DEFAULT FALSE;
    ->     DECLARE v_dno INT;
    ->
    ->     DECLARE cur CURSOR FOR SELECT DISTINCT d_no FROM emp_dept;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    ->
    ->     OPEN cur;
    ->     read_loop: LOOP
    ->         FETCH cur INTO v_dno;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         INSERT INTO dept_salary(d_no, avg_salary)
    ->         SELECT v_dno, AVG(salary) FROM emp_dept WHERE d_no = v_dno;
    ->     END LOOP;
    ->
    ->     CLOSE cur;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL DeptAvgSalary();
Query OK, 0 rows affected (0.01 sec)
```

```
mysql>
mysql> DELIMITER ;
mysql> SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL DeptAvgSalary();
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM dept_salary;
+------+------------+
| d_no | avg_salary |
+------+------------+
|  101 |   22500.00 |
|  102 |   35000.00 |
+------+------------+
2 rows in set (0.00 sec)

mysql> SET SQL_SAFE_UPDATES = 1;
Query OK, 0 rows affected (0.00 sec)
```