Assignment-6

Practice of Functions and Stored Procedures:-

```
DELIMITER $$
CREATE FUNCTION CustomerLevel(p_creditLimit double) RETURNS
VARCHAR(10)
DETERMINISTIC
BEGIN
   DECLARE lvl varchar(10);

   IF p_creditLimit > 50000 THEN
      SET lvl = 'PLATINUM';
   ELSEIF p_creditLimit <= 50000 AND p_creditLimit >= 10000 THEN
      SET lvl = 'GOLD';
   ELSEIF p_creditLimit < 10000 THEN
      SET lvl = 'SILVER';
   END IF;

   RETURN lvl;
END$$
mysql> create table deptsal as
   -> select deptno,0 as totalsalary from dept;
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from deptsal;
+--------+-------------+
| deptno | totalsalary |
+--------+-------------+
|      1 |           0 |
|     10 |           0 |
|     20 |           0 |
|     30 |           0 |
|     40 |           0 |
+--------+-------------+
5 rows in set (0.00 sec)
```

```
mysql> delimiter $$
mysql> create procedure updatesal (IN paraml int)
    -> begin
    -> update deptsal
    -> set totalsalary = (select sum(salary) from emp where deptno = paraml)
    -> where deptno = paraml;
    -> end;
    -> $$

mysql> call updatesal(1);
    -> call updatesal(20);
    -> call updatesal(10);
    -> call updatesal(30);
    -> call updatesal(40);
    -> $$
Query OK, 1 row affected (0.01 sec)
Query OK, 1 row affected (0.01 sec)
Query OK, 1 row affected (0.01 sec)
Query OK, 1 row affected (0.01 sec)
Query OK, 1 row affected (0.01 sec)

mysql> select * from deptsal;
    -> $$
+--------+-------------+
| deptno | totalsalary |
+--------+-------------+
|      1 |         400 |
|     10 |         400 |
|     20 |        1400 |
|     30 |        1400 |
|     40 |         800 |
+--------+-------------+
5 rows in set (0.00 sec)

mysql> create procedure updatesal2()
    -> begin
    -> declare done int default 0;
```

```
    -> declare current_dnum int;
    -> declare dnumcur cursor for select deptno from deptsal;
    -> declare continue handler for not found set done = 1;
    -> open dnumcur;
    -> repeat
    -> fetch dnumcur into current_dnum;
    -> update deptsal
    -> set totalsalary=(select sum(salary) from emp where deptno = current_dnum)
    -> where deptsal.deptno = current_dnum;
    -> until done
    -> end repeat;
    -> close dnumcur;
    -> end
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql> call updatesal2();
    -> //
Query OK, 0 rows affected (0.02 sec)

mysql> select * from deptsal//
+--------+-------------+
| deptno | totalsalary |
+--------+-------------+
|      1 |         400 |
|     10 |         400 |
|     20 |        1400 |
|     30 |        1400 |
|     40 |         800 |
+--------+-------------+
5 rows in set (0.00 sec)
```

Batch 1 Exercise 1

Consider the employee table:-
Employee (emp_id, first_name,last_name,hiredate)

Write a stored procedure to take the emp_id as input parameter. Procedure must raise the salary of an employee based on following conditions
If experience is less than 2 years then salary raise is 5%
If experience is between 2 to 5 years then raise is 7%
If experience is more than 5 years raise is 10%
Display appropriate messages. And add error handling

```
mysql> CREATE PROCEDURE raise_salary(IN eno INT)
   -> BEGIN
   ->     DECLARE experience INT;
   ->     DECLARE salary DECIMAL(10, 2);
   ->     DECLARE new_salary DECIMAL(10, 2);
   ->
   ->     SELECT hiredate INTO experience FROM emp WHERE eno = eno;
   ->     SELECT salary INTO salary FROM emp WHERE eno = eno;
   ->
   ->     SET new_salary =
   ->         CASE
   ->             WHEN experience < DATE_SUB(NOW(), INTERVAL 2 YEAR) THEN
salary * 1.05
   ->             WHEN experience BETWEEN DATE_SUB(NOW(), INTERVAL 2 YEAR)
AND DATE_SUB(NOW(), INTERVAL 5 YEAR) THEN salary * 1.07
   ->             ELSE salary * 1.1
   ->         END;
   ->
   ->     UPDATE emp SET salary = new_salary WHERE eno = eno;
   ->
   ->     SELECT CONCAT('Salary for employee with ID ', eno , ' has been raised to ',
new_salary) AS message;
   -> END;
   -> //
Query OK, 0 rows affected (0.03 sec)

mysql> call raise_salary(7268);
   -> call raise_salary(7312);
   -> call raise_salary(7315);
   -> call raise_salary(7345);
```

```
    -> call raise_salary(7369);
    -> call raise_salary(7369);
    -> call raise_salary(7370);
    -> call raise_salary(7371);
    -> call raise_salary(7372);
    -> call raise_salary(7373);
    -> call raise_salary(7374);
    -> call raise_salary(7375);
    -> call raise_salary(7568);
    -> //
+------+--------+---------------+------+------------+--------+------------+--------+
| eno  | ename  | job           | MGR  | hiredate   | salary | commission | deptno |
+------+--------+---------------+------+------------+--------+------------+--------+
| 7268 | Eela   | Employee      | 7374 | 2020-01-12 |    200 |          0 |      1 |
| 7312 | Samay  | Employee      | 7372 | 2020-01-12 |    200 |          0 |      1 |
| 7315 | Aman   | Employee      | 7374 | 2021-02-12 |    200 |          0 |     10 |
| 7345 | Sunil  | Employee      | 7373 | 2021-02-12 |    200 |          0 |     10 |
| 7369 | Smit   | BOSS          | NULL | 2017-12-20 |    800 |        300 |   NULL |
| 7370 | Anuj   | Senior Manager | 7369 | 2020-12-20 |    660 |        300 |     20 |
| 7371 | Anup   | Senior Manager | 7369 | 2020-11-20 |    660 |        200 |     20 |
| 7372 | Jay    | Manager       | 7370 | 2020-02-20 |    440 |        200 |     20 |
| 7373 | Amit   | Manager       | 7370 | 2021-03-20 |    440 |        200 |     20 |
| 7374 | Ajay   | Manager       | 7371 | 2020-01-20 |    440 |        200 |     30 |
| 7375 | jaggu  | Manager       | 7371 | 2020-04-21 |    440 |        200 |     30 |
| 7376 | Sumit  | Employee      | 7372 | 2021-02-12 |    200 |          0 |     40 |
| 7568 | Danish | Employee      | 7373 | 2020-01-12 |    200 |          0 |     40 |
| 7615 | Vijay  | Employee      | 7375 | 2021-02-12 |    200 |          0 |     10 |
| 7728 | Era    | Employee      | 7375 | 2020-01-12 |    200 |          0 |      1 |
+------+--------+---------------+------+------------+--------+------------+--------+
15 rows in set (0.00 sec)
```

Batch 1 Exercise 2

- Write a function to return and display the number of years of service for an employee. The function should take the hiredate as a parameter.
- Also write a code to call the function.

```
mysql> CREATE FUNCTION get_years_of_service(hiredate DATE)
```

```
   -> RETURNS INT DETERMINISTIC
   -> BEGIN
   ->     DECLARE years_of_service INT;
   ->     SET years_of_service = YEAR(CURDATE()) - YEAR(hiredate);
   ->     IF MONTH(CURDATE()) < MONTH(hiredate) OR (MONTH(CURDATE()) =
MONTH(hiredate) AND DAY(CURDATE()) < DAY(hiredate)) THEN
   ->         SET years_of_service = years_of_service - 1;
   ->     END IF;
   ->     RETURN years_of_service;
   -> END;
   -> //
Query OK, 0 rows affected (0.01 sec)
mysql> SELECT ename, hiredate, calculate_years_of_service(hiredate) AS
years_of_service FROM emp;
   -> //
mysql> SELECT get_years_of_service(hiredate) AS years_of_service FROM emp;
   -> //
+------------------+
| years_of_service |
+------------------+
|                3 |
|                3 |
|                2 |
|                2 |
|                5 |
|                2 |
|                2 |
|                3 |
|                2 |
|                3 |
|                3 |
|                2 |
|                3 |
|                2 |
|                3 |
+------------------+
15 rows in set (0.01 sec)
```