SQL Intern Task- 8

NAME : SWETHA R
ROLL NUMBER : 6380459779
Email: rswetha2807@gmail.com

Task - 8 : Stored Procedures and Functions

EXAMPLE TABLE :

```
CREATE TABLE employees (
      emp_id INT AUTO_INCREMENT PRIMARY KEY ,
       name VARCHAR (100),
       Salary DECIMAL (10, 2)
);

INSERT INTO employees (name, salary ) VALUES
('Arun' , 40000) ,
('Meena' , 60000) ,
('Raj' , 75000) ;
```

1. What is DELIMITER in MYSQL?

In MYSQL, the default command terminator is a semicolon (;)
SELECT * FROM employees;

The ; tells MYSQL:
"End of the command, now execute it"

2. WHY do we change the delimiter when writing a procedure or function?

Stored procedures and functions contain multiple SQL statements inside them and they often also use semicolons (;) inside the body.

```
CREATE PROCEDURE my_proc()
BEGIN
  SELECT * FROM employees;
  UPDATE employees SET salary = salary + 1000;
END;
```

- ➢ If we don't change the delimiter, MySQL gets **confused** and thinks the procedure is over **after the first ;**.

➢ **So we use `DELIMITER //` or `DELIMITER $$`**

<u>This tells MySQL:</u>

"Use `//` (or `$$`) instead of `;` to know where the *entire block* ends."

- DELIMITER // - Tell MySQL: "I'll finish the command using `//` instead of `;`."
- END // - Now the command ends here (not earlier inside BEGIN...END block).
- DELIMITER ; - Switches back to normal `;` after finishing the procedure.

1. CREATE PROCEDURE with Parameters & Logic

<u>Example : Increase an Employee's Salary by a % Value</u>

```
DELIMITER //
CREATE PROCEDURE IncreaseSalary (
        IN empId INT ,
        IN percent DECIMAL (5 , 2)
)
BEGIN
      UPDATE employees
      SET salary = salary + (salary * percent / 100 )
      WHERE emp_id = empId;
END //
DELIMITER ;
```

<u>Call it:</u>

```
CALL IncreaseSalary(1, 10);
```

2.CREATE FUNCTION with Logic and Return Value

<u>Example : Calculate Yearly Salary from Monthly Salary</u>

```
DELIMITER //
CREATE FUNCTION GetYearlySalary (
       Monthly Decimal (10,2)
)
RETURNS DECIMAL (10,2)
DETERMINISTIC
BEGIN
      RETURN monthly * 12;
END //
```

```
DELIMITER ;
```

<u>USE IT :</u>

```
SELECT name, salary, GetYearlySalary(salary) AS yearly_salary
FROM employees;
```

<u>Drop if Needed</u>

- DROP PROCEDURE IF EXISTS IncreaseSalary;
- DROP FUNCTION IF EXISTS GetYearlySalary;

<u>SYNTAX :</u>
<u>Procedure Syntax</u>

```
DELIMITER //
 CREATE PROCEDURE  procedure_name ( IN param1 TYPE, OUT parama2 TYPE, …)
BEGIN
--- sql logic
END //
DELIMITER ;
```

<u>Function Syntax</u>

```
DELIMITER //
CREATE FUNCTION function_name (param TYPE)
RETURNS return_type
BEGIN
    RETURN something;
END //
DELIMITER:
```

<u>OUTCOME:</u>

- `DELIMITER` is used to **change the command-end symbol temporarily**
- It avoids confusion when **semicolon `;` is used inside** procedures/functions
- You always set it **back to `;` at the end**
  Understand the difference between procedures and functions
- **Be able to modularize SQL logic**
- **Use input parameters and conditional logic**

- **Write reusable, efficient, and clean SQL code blocks**