Week 3 – Triggers and Routines

SIT DOLOR AMET

Introduction to Triggers and Routines

- **Triggers**: Special kind of stored procedure that automatically executes when certain events occur in the database, such as INSERT, UPDATE, or DELETE operations.
- **Routines**: Includes stored procedures and functions. These are sets of SQL statements that perform specific tasks and can be invoked by the database engine or applications.

Triggers

- **Types of Triggers**:
  - **BEFORE Triggers**: Execute before the triggering event (e.g., before an INSERT).
  - **AFTER Triggers**: Execute after the triggering event.

Use Cases

- Automatically update audit logs.
- Enforce complex integrity constraints that can't be handled by simple constraints.

Example of a Trigger

Scenario: Log employee salary changes in an audit table whenever an update occurs.

```
CREATE TRIGGER log_salary_changes
AFTER UPDATE ON employees
FOR EACH ROW
BEGIN
  INSERT INTO salary_audit (employee_id, old_salary, new_salary, change_date)
  VALUES (OLD.employee_id, OLD.salary, NEW.salary, NOW());
END;
```

- ❑OLD: Refers to the previous state of the row before the update.
- ❑NEW: Refers to the new state of the row after the update.
- ❑Impact: This trigger automatically logs every salary change in the salary_audit table.

Best Practices for Triggers

- **Keep Triggers Simple**: Avoid complex logic that can make debugging difficult.
- **Avoid Overuse**: Triggers can impact performance, so use them only when necessary.
- **Ensure Idempotency**: Triggers should not cause infinite loops or multiple unintended executions.

Routines

❑Stored Procedures:

- A block of code that performs a specific task and can be called with parameters.
- Use Cases: Automate recurring tasks, encapsulate business logic, improve performance by reducing the number of database calls.

❑Functions:

- Similar to stored procedures but designed to return a value. Typically used in queries to encapsulate reusable logic.
- Use Cases: Return computed values, perform calculations, or encapsulate common expressions.

Example of a Stored Procedure

Scenario: Calculate and return the total salary for a given department.

```
CREATE PROCEDURE GetTotalSalary(IN dept_id INT, OUT total_salary DECIMAL(10,2))
BEGIN
  SELECT SUM(salary) INTO total_salary
  FROM employees
  WHERE department_id = dept_id;
END;
```

- **IN**: Input parameter.
- **OUT**: Output parameter.
- **Usage**: This procedure calculates the total salary for a department and returns the result via the total_salary output parameter.

Example of a Function

Scenario: Create a function that returns the number of employees in a department.

```
 CREATE FUNCTION GetEmployeeCount(dept_id INT)
 RETURNS INT
BEGIN
 DECLARE emp_count INT;
 SELECT COUNT(*) INTO emp_count
 FROM employees
 WHERE department_id = dept_id;
 RETURN emp_count;
END;
```

This function can be used directly in SQL queries to return the number of employees in a given department.