1: Track Salary Updates
Context:
A company wants to maintain a log of all salary changes for employees. Every time an employee’s
salary is updated, the old and new values should be stored in a separate table for audit purposes.
Tables:
● employees(emp_id INT PRIMARY KEY, name VARCHAR(50), salary DECIMAL(10,2))
● salary_log(log_id INT AUTO_INCREMENT PRIMARY KEY, emp_id INT, old_salary DECIMAL(10,2), new_salary DECIMAL(10,2), change_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP)
Objective:
Create a BEFORE UPDATE trigger on the employees table that:
● Captures the old and new salary values whenever salary is updated
● Inserts them into the salary_log table

```
-- Step 1: Create Tables
-- Employees table
CREATE TABLE employees (
    emp_id INT PRIMARY KEY,
    name VARCHAR(50),
    salary DECIMAL(10,2)
);

-- Salary log table
CREATE TABLE salary_log (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    emp_id INT,
    old_salary DECIMAL(10,2),
    new_salary DECIMAL(10,2),
    change_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Step 2: Create Trigger
DELIMITER $$

CREATE TRIGGER before_salary_update
BEFORE UPDATE ON employees
FOR EACH ROW
BEGIN
    -- Only log if salary actually changes
    IF OLD.salary != NEW.salary THEN
        INSERT INTO salary_log (emp_id, old_salary, new_salary)
        VALUES (OLD.emp_id, OLD.salary, NEW.salary);
    END IF;
END$$
```

```
DELIMITER ;

-- Example Usage
-- Insert an employee
INSERT INTO employees (emp_id, name, salary)
VALUES (1, 'Anita Sharma', 30000.00);

-- Update salary (this will trigger the log)
UPDATE employees
SET salary = 35000.00
WHERE emp_id = 1;

-- View the log
SELECT * FROM salary_log;
```