PRACTICAL - 9

Q1. Create a trigger before insert on employee table on each row.

```
CREATE TRIGGER before_insert_employee
BEFORE INSERT ON employee
FOR EACH ROW
BEGIN
    -- Sample trigger logic: Log the inserted data
    INSERT INTO employee_log (employee_id, employee_name,
action_type, action_time)
    VALUES (NEW.id, NEW.name, 'INSERT', NOW());
END;
```

Q2. Create a trigger after into department table.

```
CREATE TRIGGER after_insert_department
AFTER INSERT ON department
FOR EACH ROW
BEGIN
    -- Sample trigger logic: Update a counter or perform other
actions
    UPDATE department_statistics
    SET department_count = department_count + 1;
END;
```

Q3. Drop a trigger on department table.

```
DROP TRIGGER after_insert_department;
```

Q4. Create a trigger on project table before update.

```
CREATE TRIGGER before_update_project
BEFORE UPDATE ON project
FOR EACH ROW
BEGIN
    -- Sample trigger logic: Ensure that the project budget
doesn't exceed a limit
    IF NEW.budget > 1000000 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Project budget exceeds the allowed
limit.';
    END IF;
END;
```

Q5. Create a trigger on employee table for update.

```
CREATE TRIGGER for_update_employee
BEFORE UPDATE ON employee
FOR EACH ROW
BEGIN
    -- Sample trigger logic: Log the changes made during the
update
    IF NEW.salary <> OLD.salary THEN
        INSERT INTO employee_changes (employee_id, old_salary,
new_salary, change_time)
        VALUES (NEW.id, OLD.salary, NEW.salary, NOW());
    END IF;
END;
```

Q6. Create a trigger on employee table and drop it.

```
CREATE TRIGGER before_insert_employee
BEFORE INSERT ON employee
FOR EACH ROW
BEGIN
    -- Your trigger logic here
    -- For example, you can log the inserted data or perform
validations
END;

DROP TRIGGER before_insert_department;
```