

EXPERIMENT - 7

Student Name: Tanay Manish Nesari

Branch: BE-CSE

Semester: 5th

Subject Name: ADBMS

UID: 23BCS13761

Section/Group: KRG 1-B

Date of Performance: 30/09/2025

Subject Code: 23CSP-333

Aim: To understand and implement Triggers.

Objective:

1) MEDIUM LEVEL PROBLEM-----

--REQUIREMENTS: DESIGN A TRIGGER WHICH:

--1. WHENEVER THERE IS A INSERTION ON STUDENT TABLE THEN, THE CURRENTLY
INSERTED OR DELETED

--ROW SHOULD BE PRINTED AS IT AS ON THE OUTPUT CONSOLE WINDOW

2) HARD LEVEL PROBLEM-----

Requirements: DESIGN A PORSTGRESQL TRIGGERS THAT:

Whenever a new employee is inserted in tbl_employee, a record should be added to tbl_employee_audit like:
"Employee name <emp_name> has been added at <current_time>"

Whenever an employee is deleted from tbl_employee, a record should be added to tbl_employee_audit like:
"Employee name <emp_name> has been deleted at <current_time>"

The solution must use PostgreSQL triggers.

Code:

```
1. CREATE OR REPLACE FUNCTION fn_student_audit()  
RETURNS TRIGGER  
LANGUAGE plpgsql  
AS  
$$  
BEGIN  
IF TG_OP = 'INSERT' THEN  
    RAISE NOTICE 'Inserted Row -> ID: %, Name: %, Age: %, Class: %',  
        NEW.id, NEW.name, NEW.age, NEW.class;  
    RETURN NEW;  
  
ELSIF  
    TG_OP = 'DELETE' THEN  
    RAISE NOTICE 'Deleted Row -> ID: %, Name: %, Age: %, Class: %',  
        OLD.id, OLD.name, OLD.age, OLD.class;  
    RETURN OLD;  
END IF;
```

```
    RETURN NULL;
END;
$$
```

```
2. CREATE TABLE tbl_employee (
emp_id SERIAL PRIMARY KEY,
emp_name VARCHAR(100) NOT NULL,
emp_salary NUMERIC
);
```

```
CREATE TABLE tbl_employee_audit (
sno SERIAL PRIMARY KEY,
message TEXT
);
```

```
CREATE OR REPLACE FUNCTION audit_employee_changes()
RETURNS TRIGGER
LANGUAGE plpgsql
AS
$$
BEGIN
    IF TG_OP = 'INSERT' THEN
        INSERT INTO tbl_employee_audit(message)
        VALUES ('Employee name ' || NEW.emp_name || ' has been added at ' || NOW());
        RETURN NEW;

    ELSIF TG_OP = 'DELETE' THEN
        INSERT INTO tbl_employee_audit(message)
        VALUES ('Employee name ' || OLD.emp_name || ' has been deleted at ' || NOW());
        RETURN OLD;
    END IF;

    RETURN NULL;
END;
$$
```

```
CREATE TRIGGER trg_employee_audit
AFTER INSERT OR DELETE
ON
tbl_employee
FOR EACH ROW
EXECUTE FUNCTION audit_employee_changes();
```

```
INSERT INTO tbl_employee(emp_name, emp_salary) VALUES ('Aman', 50000);
```

```
DELETE FROM tbl_employee WHERE emp_name = 'Aman';
```

```
SELECT * FROM tbl_employee_audit;
```