DEPARTMENT OF COMPUTER

Discover. Learn. Empower.

Experiment 3

Student Name: Riya Kashyap
UID: 25MCA20033
Branch: MCA-UIC
Section/Group: 1
Semester: 2nd
Date of Performance: 27/1/2026
Subject Name: Technical Training
Subject Code: 25CAP-652

1. Aim: To implement conditional decision-making logic in PostgreSQL using **IF–ELSE constructs** and **CASE expressions** for classification, validation, and rule-based data processing.

2. Objective:

- To understand conditional execution in SQL
- To implement decision-making logic using CASE expressions
- To simulate real-world rule validation scenarios
- To classify data based on multiple conditions
- To strengthen SQL logic skills required in interviews and backend systems

3. Practical/Experiment Steps:

- Create the required table to store schema name and violation count.
- Insert sample records with different violation values.
- Retrieve data using a SELECT statement.
- Apply searched CASE expression to classify violations.
- Display classification results in the output.
- Alter the table to add an approval status column.
- Update approval status using CASE logic.
- Verify updated records using SELECT query.
- Declare variables inside a PL/pgSQL block.
- Implement IF–ELSE and ELSIF conditions.
- Display messages based on condition evaluation.
- Create a student table for real-world classification.
- Insert student marks into the table.
- Use CASE expression to assign grades.

- Display student name, marks, and grade.
- Apply CASE inside ORDER BY clause.
- Sort records based on priority conditions.
- Verify final output for correctness.
- Analyze results based on business rules.
- Conclude the experiment successfully.

4. Procedure of the Practical:

```
-- Step 0
-- Create table for schema violations
CREATE TABLE schema_violations (
   schema_id SERIAL PRIMARY KEY,
   schema_name VARCHAR(50),
   violation_count INT
);

-- Insert sample data
INSERT INTO schema_violations (schema_name, violation_count) VALUES
('HR_SCHEMA', 0),
('FIN_SCHEMA', 2),
('SALES_SCHEMA', 5),
('PAYROLL_SCHEMA', 9),
('AUDIT_SCHEMA', 15);

--step 1
SELECT
   schema_name,
   violation_count,
   CASE
      WHEN violation_count = 0 THEN 'No Violation'
      WHEN violation_count BETWEEN 1 AND 3 THEN 'Minor Violation'
      WHEN violation_count BETWEEN 4 AND 7 THEN 'Moderate Violation'
      ELSE 'Critical Violation'
   END AS violation_status
FROM schema_violations;

-- Step 2
-- Add new column
ALTER TABLE schema_violations
ADD COLUMN approval_status VARCHAR(20);

-- Update using CASE
UPDATE schema_violations
SET approval_status = CASE
   WHEN violation_count = 0 THEN 'Approved'
   WHEN violation_count BETWEEN 1 AND 5 THEN 'Needs Review'
   ELSE 'Rejected'
END;
```

```
SELECT * FROM schema_violations;

-- Step 3
DO $$
DECLARE
   v_count INT := 6;
BEGIN
   IF v_count = 0 THEN
      RAISE NOTICE 'No violations detected.';
   ELSIF v_count BETWEEN 1 AND 5 THEN
      RAISE NOTICE 'Minor issues found. Review needed.';
   ELSIF v_count BETWEEN 6 AND 10 THEN
      RAISE NOTICE 'Moderate violations detected.';
   ELSE
      RAISE NOTICE 'Critical violations! Immediate action required.';
   END IF;
END $$;


-- Step 4
-- Create student table
CREATE TABLE students (
   student_id SERIAL PRIMARY KEY,
   student_name VARCHAR(50),
   marks INT
);

-- Insert data
INSERT INTO students (student_name, marks) VALUES
('Riya', 92),
('Aman', 78),
('Neha', 65),
('Rahul', 48),
('Pooja', 34);


SELECT
   student_name,
   marks,
   CASE
      WHEN marks >= 90 THEN 'A'
      WHEN marks >= 75 THEN 'B'
      WHEN marks >= 60 THEN 'C'
      WHEN marks >= 40 THEN 'D'
      ELSE 'Fail'
   END AS grade
FROM students;

-- Step 5
SELECT
   schema_name,
   violation_count,
   approval_status
```

```
FROM schema_violations
ORDER BY
  CASE
    WHEN violation_count = 0 THEN 1
    WHEN violation_count BETWEEN 1 AND 3 THEN 2
    WHEN violation_count BETWEEN 4 AND 7 THEN 3
    ELSE 4
  END;
```

5. Output:

	schema_name character varying (50)	violation_count integer	violation_status text
1	HR_SCHEMA	0	No Violation
2	FIN_SCHEMA	2	Minor Violation
3	SALES_SCHEMA	5	Moderate Violati...
4	PAYROLL_SCHEMA	9	Critical Violation
5	AUDIT_SCHEMA	15	Critical Violation

	schema_id [PK] integer	schema_name character varying (50)	violation_count integer	approval_status character varying (20)
1	1	HR_SCHEMA	0	Approved
2	2	FIN_SCHEMA	2	Needs Review
3	3	SALES_SCHEMA	5	Needs Review
4	4	PAYROLL_SCHEMA	9	Rejected
5	5	AUDIT_SCHEMA	15	Rejected

Data Output | Messages | Notifications

```
NOTICE:  Moderate violations detected.
DO

Query returned successfully in 257 msec.
```

	student_name character varying (50)	marks integer	grade text
1	Riya	92	A
2	Aman	78	B
3	Neha	65	C
4	Rahul	48	D
5	Pooja	34	Fail

	schema_name character varying (50)	violation_count integer	approval_status character varying (20)
1	HR_SCHEMA	0	Approved
2	FIN_SCHEMA	2	Needs Review
3	SALES_SCHEMA	5	Needs Review
4	PAYROLL_SCHEMA	9	Rejected
5	AUDIT_SCHEMA	15	Rejected

6. Learning Outcome:

- Understand the use of conditional logic in PostgreSQL.
- Apply CASE expressions for data classification.
- Implement IF–ELSE logic using PL/pgSQL blocks.
- Perform rule-based decision making inside the database.
- Classify real-world data using multiple conditions.
- Automate validations without application-level logic.
- Use CASE expressions in SELECT, UPDATE, and ORDER BY clauses.
- Strengthen SQL skills required for backend systems.
- Gain confidence in interview-oriented SQL problems.
- Improve logical thinking for database-driven applications.