Experiment: 4

Student Name: Sahil Goyal
UID: 24BDA70148
Branch: CSE
Section/Group: AIT-KRG-GP2
Semester: 4th
Date of Performance: 30/01/26
Subject Name: DBMS

Aim of the Practical:

To understand and implement conditional control structures in PL/pgSQL by creating and executing PL/pgSQL blocks using IF–ELSE, ELSIF, ELSIF ladder, and CASE statements in PostgreSQL to control the flow of execution and demonstrate decision-making capabilities.

Tool Used:

Database Management System:
PostgreSQL

Database Administration Tool:
psql (iTerm2 Terminal)

Objective:

To implement control structures in PL/pgSQL including:

- IF–ELSE
- IF–ELSIF–ELSE
- ELSIF Ladder
- CASE Statement

and analyze how conditional statements control program execution in PostgreSQL.

Practical / Experimental Steps

Step 1: Open iTerm2 terminal.
Step 2: Connect to PostgreSQL using:

```
psql -U postgres -d database_name
```

Step 3: Write and execute a PL/pgSQL DO block using IF–ELSE to check whether a number is positive or non-positive.
Step 4: Write and execute a PL/pgSQL DO block using IF–ELSIF–ELSE to evaluate student grade.
Step 5: Write and execute a PL/pgSQL DO block using an ELSIF ladder to determine performance status.
Step 6: Write and execute a PL/pgSQL DO block using CASE statement to display day name based on day number.
Step 7: Modify input values and re-execute the programs to verify different conditions.
Step 8: Observe and record the outputs displayed using RAISE NOTICE.

• I / O Analysis

1️⃣ IF–ELSE Statement

Program:

```
DO $$
DECLARE
    num INTEGER := -5;
BEGIN
    IF num > 0 THEN
        RAISE NOTICE 'The number % is Positive.', num;
    ELSE
        RAISE NOTICE 'The number % is Non-Positive.', num;
    END IF;
END $$;
```

Output:

NOTICE: The number -5 is Non-Positive.

2️⃣ IF–ELSIF–ELSE Statement

Program:

```
DO $$
DECLARE
    marks INTEGER := 85;
BEGIN
    IF marks >= 90 THEN
        RAISE NOTICE 'Grade: A+';
    ELSIF marks >= 75 THEN
        RAISE NOTICE 'Grade: A';
    ELSIF marks >= 60 THEN
        RAISE NOTICE 'Grade: B';
    ELSIF marks >= 50 THEN
        RAISE NOTICE 'Grade: C';
    ELSE
        RAISE NOTICE 'Grade: Fail';
    END IF;
END $$;
```

Output:

NOTICE: Grade: A

3️⃣ ELSIF Ladder

Program:

```
DO $$
DECLARE
    marks INTEGER := 72;
BEGIN
    IF marks >= 85 THEN
        RAISE NOTICE 'Performance: Excellent';
    ELSIF marks >= 70 THEN
        RAISE NOTICE 'Performance: Very Good';
```

```
    ELSIF marks >= 55 THEN
        RAISE NOTICE 'Performance: Good';
    ELSIF marks >= 40 THEN
        RAISE NOTICE 'Performance: Average';
    ELSE
        RAISE NOTICE 'Performance: Poor';
    END IF;
END $$;
```

Output:

NOTICE: Performance: Very Good

---

4 CASE Statement

Program:

```
DO $$
DECLARE
    day_number INTEGER := 4;
BEGIN
    CASE day_number
        WHEN 1 THEN RAISE NOTICE 'Day: Sunday';
        WHEN 2 THEN RAISE NOTICE 'Day: Monday';
        WHEN 3 THEN RAISE NOTICE 'Day: Tuesday';
        WHEN 4 THEN RAISE NOTICE 'Day: Wednesday';
        WHEN 5 THEN RAISE NOTICE 'Day: Thursday';
        WHEN 6 THEN RAISE NOTICE 'Day: Friday';
        WHEN 7 THEN RAISE NOTICE 'Day: Saturday';
        ELSE
            RAISE NOTICE 'Invalid Day Number';
    END CASE;
END $$;
```

Output:

NOTICE: Day: Wednesday

• Learning Outcomes (What I Have Learnt):

- Understood the structure of a PL/pgSQL block using DO DECLAREBEGINENDDECLARE BEGIN ENDDECLAREBEGINEND.
- Learned to declare and initialize variables in PostgreSQL.
- Gained knowledge of using RAISE NOTICE to display output.
- Practiced implementing decision-making using IF–ELSE and ELSIF ladder.
- Understood the working of CASE statement in PostgreSQL.
- Developed understanding of controlling execution flow inside database programs.