

Experiment 4

Student Name: Yashraj Jangra
Branch: CSE - AIML
Semester: 4
Subject Name: DBMS

UID: 24BAI70476
Section/Group: 24AIT_KRG G1
Date of Performance: 4/2/26
Subject Code: 24CSH-298

Aim

To design and implement PL/SQL programs utilizing conditional control statements such as IF-ELSE, ELSIF, ELSIF ladder, and CASE constructs in order to control the flow of execution based on logical conditions and to analyse decision-making capabilities in PL/SQL blocks.

Software Requirements

- Database Management System:
 - PostgreSQL
- Database Administration Tool:
 - pgAdmin
- Implement control structures in PL/SQL (IF-ELSE, ELSE-IF, ELSE-IF LADDER, CASE STATEMENTS in PL-SQL BLOCK).

Problem Statement

Develop and execute PL/SQL programs that demonstrate the use of conditional control statements. The programs should employ IF-ELSE, ELSIF, ELSIF ladder, and CASE statements to evaluate given conditions and control the flow of execution accordingly, thereby illustrating decision-making capabilities in PL/SQL blocks.

1. Problem Statement – IF-ELSE Statement

Write a PL/SQL program to check whether a given number is positive or non-positive using the IF-ELSE conditional control statement and display an appropriate message.

2. Problem Statement – IF-ELSIF-ELSE Statement

Write a PL/SQL program to evaluate the grade of a student based on the obtained marks using the IF–ELSIF–ELSE statement and display the corresponding grade.

3. Problem Statement – ELSIF Ladder

Write a PL/SQL program to determine the performance status of a student based on marks using an ELSIF ladder and display the appropriate result.

4. Problem Statement – CASE Statement

Write a PL/SQL program to display the name of the day based on a given day number using the CASE conditional statement.

Practical/Experiment Steps

- Control Structure Implementation: Designed multiple PL/SQL blocks to explore diverse conditional logic formats, including simple branching and multi-path evaluation.
- Logic Branching Analysis: Utilised IF-ELSE and ELSIF ladders to categorize numerical data into specific ranges, such as student grades and performance statuses.
- Selection Optimisation: Implemented the CASE statement as a streamlined alternative to multiple conditional checks for mapping discrete values like day numbers to names.
- Dynamic Messaging: Integrated variable-driven output strings to provide real-time feedback based on the evaluation of input conditions.
- Execution Flow Control: Validated the decision-making capabilities of the PL/SQL engine by testing various input scenarios to ensure the correct code path was activated.

Procedure

- Enabled the output server environment to ensure all procedural results would be visible in the console window.
- Constructed a basic IF-ELSE block to perform a binary check on a numerical variable for positive or non-positive properties.
- Developed an IF-ELSIF-ELSE structure to map student marks to specific letter grades based on defined percentage thresholds.
- Expanded the conditional logic into a comprehensive ELSIF ladder to categorise performance into tiers such as Distinction, First Class, and Pass.

- Implemented a CASE statement block to translate integer inputs into corresponding day names, including a default handler for invalid entries.
- Initialised diverse test values for each variable, such as negative numbers for sign checks and specific marks for grading, to verify logic accuracy.
- Nested the procedural logic within standard BEGIN...END; blocks to maintain structured programming principles.
- Executed each individual block sequentially and monitored the DBMS output console for the expected string concatenations.
- Verified that the output correctly reflected the logic branch associated with the assigned variable values and documented the results.
- Verified the console output against the manual calculations to ensure the logic and variables were handled correctly.

Input/Output Analysis

SQL Queries Input

```

DECLARE
    v_marks NUMBER := 82;
    v_grade VARCHAR2(20);
    v_result VARCHAR2(20);
BEGIN
    IF v_marks >= 75 THEN
        v_grade := 'A Grade';
    ELSIF v_marks >= 60 THEN
        v_grade := 'B Grade';
    ELSIF v_marks >= 50 THEN
        v_grade := 'C Grade';
    ELSE
        v_grade := 'Fail';
    END IF;

    IF v_marks >= 50 THEN
        v_result := 'Pass';
    ELSE
        v_result := 'Fail';
    END IF;

    DBMS_OUTPUT.PUT_LINE('Marks: ' || v_marks || ', Grade: ' || v_grade);
    DBMS_OUTPUT.PUT_LINE('Result: ' || v_result);
END;

```

Output

The screenshot shows the FreeSQL Worksheet interface. On the left, the Navigator pane displays the schema structure, including tables like USERS, USER_ID, USER_NAME, DEPARTMENT, SALARY, and HIRE_DATE. The main workspace contains a PL/SQL script:

```
5  DECLARE
6      v_marks NUMBER := 82;
7      v_grade VARCHAR2(20);
8      v_result VARCHAR2(20);
9  BEGIN
10     IF v_marks >= 75 THEN
11         v_grade := 'A Grade';
12     ELSIF v_marks >= 60 THEN
13         v_grade := 'B Grade';
14     ELSIF v_marks >= 50 THEN
15         v_grade := 'C Grade';
16     ELSE
17         v_grade := 'Fail';
18     END IF;
19
20     IF v_marks >= 50 THEN
21         v_result := 'Pass';
22     ELSE
23         v_result := 'Fail';
24     END IF;
25
26     DBMS_OUTPUT.PUT_LINE('Marks: ' || v_marks || ', Grade: ' || v_grade);
27     DBMS_OUTPUT.PUT_LINE('Result: ' || v_result);
28 END;
```

The "Script output" tab is selected, showing the execution results:

```
SQL> DECLARE
      v_marks NUMBER := 82;
      v_grade VARCHAR2(20);
      v_result VARCHAR2(20)...
Show more...

Marks: 82, Grade: A Grade
Result: Pass

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.002
```

At the bottom, there are links for About Oracle, Contact Us, Legal Notices, Terms and Conditions, Your Privacy Rights, Delete Your FreeSQL Account, and Cookie Preferences. The page footer indicates version 26.1.2 - Copyright © 2.

SQL Queries Input

```
DECLARE
    v_num NUMBER := 17;
    v_type VARCHAR2(10);
BEGIN
    IF MOD(v_num, 2) = 0 THEN
        v_type := 'Even';
    ELSE
        v_type := 'Odd';
    END IF;

    DBMS_OUTPUT.PUT_LINE('Number: ' || v_num || ', Type: ' || v_type);
END;
```

Output

The screenshot shows the FreeSQL SQL Worksheet interface. On the left, the Navigator pane displays the schema structure under 'My Schema', including tables like USERS, USER_ID, USER_NAME, DEPARTMENT, SALARY, and HIRE_DATE. The main area is titled '[SQL Worksheet]*' and contains a PL/SQL script. The script declares variables v_num and v_type, initializes v_num to 17, and v_type to 'VARCHAR2(10)'. It then begins a loop where it checks if v_num is even or odd. If even, v_type is set to 'Even'; if odd, v_type is set to 'Odd'. Finally, it prints 'Number: ' || v_num || ', Type: ' || v_type. The right side of the interface shows a graphical Explain Plan for the current query.

```
26     DBMS_OUTPUT.PUT_LINE('Marks: ' || v_marks || ', Grade: ' || v_grade);
27     DBMS_OUTPUT.PUT_LINE('Result: ' || v_result);
28   END;
29
30
31  -- Even or odd number
32  DECLARE
33    v_num NUMBER := 17;
34    v_type VARCHAR2(10);
35  BEGIN
36    IF MOD(v_num, 2) = 0 THEN
37      v_type := 'Even';
38    ELSE
39      v_type := 'Odd';
40    END IF;
41
42    DBMS_OUTPUT.PUT_LINE('Number: ' || v_num || ', Type: ' || v_type);
43  END;
44
45
46  -- Week number to weekday
47  DECLARE
48    v_week_num NUMBER := 3;
49    v_day VARCHAR2(15);
```

Query result Script output DBMS output Explain Plan SQL history

Elapsed: 00:00:00.002

```
SQL> DECLARE
      v_num NUMBER := 17;
      v_type VARCHAR2(10);
      BEGIN...
Show more...
```

Number: 17, Type: Odd

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.007

About Oracle | Contact Us | Legal Notices | Terms and Conditions | Your Privacy Rights | Delete Your FreeSQL Account | Cookie Preferences 26.1.2 - Copyright © 2023

SQL Queries Input

```
DECLARE
  v_week_num NUMBER := 3;
  v_day VARCHAR2(15);
BEGIN
  IF v_week_num = 1 THEN
    v_day := 'Monday';
  ELSIF v_week_num = 2 THEN
    v_day := 'Tuesday';
  ELSIF v_week_num = 3 THEN
    v_day := 'Wednesday';
  ELSIF v_week_num = 4 THEN
    v_day := 'Thursday';
  ELSIF v_week_num = 5 THEN
    v_day := 'Friday';
  ELSIF v_week_num = 6 THEN
    v_day := 'Saturday';
  ELSIF v_week_num = 7 THEN
    v_day := 'Sunday';
```

```

ELSE
    v_day := 'Invalid';
END IF;

DBMS_OUTPUT.PUT_LINE('Week No: ' || v_week_num || ', Day: ' || v_day);
END;

```

Output

The screenshot shows the FreeSQL SQL Worksheet interface. The code in the worksheet window is:

```

47  DECLARE
48      v_week_num NUMBER := 3;
49      v_day VARCHAR2(15);
50  BEGIN
51      IF v_week_num = 1 THEN
52          v_day := 'Monday';
53      ELSIF v_week_num = 2 THEN
54          v_day := 'Tuesday';
55      ELSIF v_week_num = 3 THEN
56          v_day := 'Wednesday';
57      ELSIF v_week_num = 4 THEN
58          v_day := 'Thursday';
59      ELSIF v_week_num = 5 THEN
60          v_day := 'Friday';
61      ELSIF v_week_num = 6 THEN
62          v_day := 'Saturday';
63      ELSIF v_week_num = 7 THEN
64          v_day := 'Sunday';
65      ELSE
66          v_day := 'Invalid';
67      END IF;
68
69      DBMS_OUTPUT.PUT_LINE('Week No: ' || v_week_num || ', Day: ' || v_day);
70  END;

```

The query result pane shows the output of the execution:

```

Elapsed: 00:00:00.007

SQL> DECLARE
      v_week_num NUMBER := 3;
      v_day VARCHAR2(15);
      BEGIN...
Show more...

Week No: 3, Day: Wednesday

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.003

```

Learning Outcomes

- Gained proficiency in using IF-ELSE, ELSIF ladders, and CASE statements to control program execution flow.
- Evaluated data variables to automate specific outcomes, such as student grading or performance status.
- Using CASE statements as a streamlined method for mapping discrete values like day numbers to names.
- Skills in setting logical thresholds to categorize raw numerical marks into descriptive classifications