

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

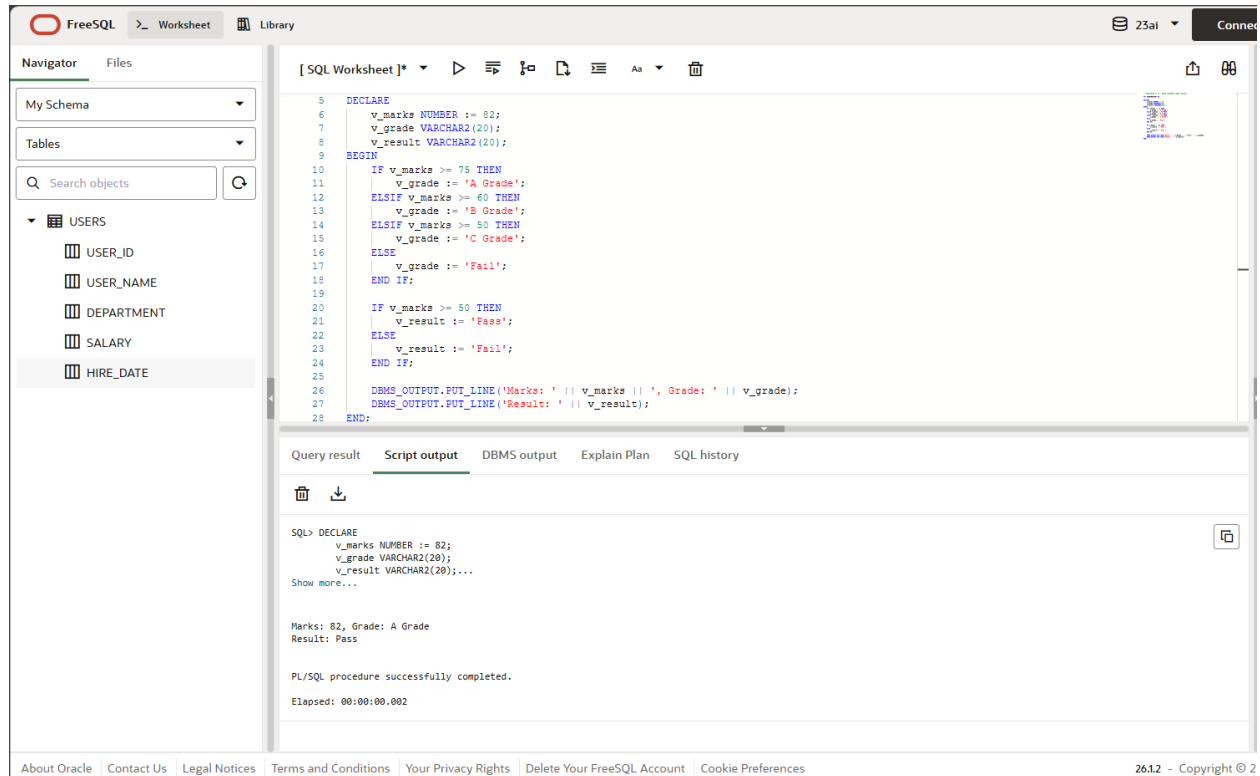
```

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 displays the FreeSQL web interface. On the left, the 'Navigator' pane shows a schema named 'My Schema' with a table 'USERS' containing columns: USER_ID, USER_NAME, DEPARTMENT, SALARY, and HIRE_DATE. The main editor area, titled '[SQL Worksheet]*', contains a PL/SQL script. The script declares variables v_marks (NUMBER), v_grade (VARCHAR2(20)), and v_result (VARCHAR2(20)). It then uses a series of IF-ELSE statements to determine the grade based on v_marks and the result based on v_marks. Finally, it uses DBMS_OUTPUT.PUT_LINE to display the marks, grade, and result. The 'Script output' tab is active, showing the execution results: 'Marks: 82, Grade: A Grade' and 'Result: Pass'. The output also confirms that the PL/SQL procedure was successfully completed and shows an elapsed time of 00:00:00.002.

```
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;
```

Query result | **Script output** | DBMS output | Explain Plan | SQL history

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

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 displays the FreeSQL web interface. On the left, the 'Navigator' pane shows a schema named 'My Schema' with a table 'USERS' containing columns: USER_ID, USER_NAME, DEPARTMENT, SALARY, and HIRE_DATE. The main editor shows a PL/SQL script with the following code:

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

The 'Script output' pane shows the execution results:

```
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
```

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 displays the FreeSQL web-based IDE interface. On the left, a 'Navigator' pane shows a database schema named 'My Schema' with a table 'USERS' containing columns: USER_ID, USER_NAME, DEPARTMENT, SALARY, and HIRE_DATE. The main editor area, titled '[SQL Worksheet]', contains a PL/SQL script with line numbers 47 to 70. The script declares a variable `v_week_num` as a NUMBER (3) and `v_day` as a VARCHAR2 (15). It uses a `BEGIN` block with an `IF-ELSIF-ELSE` ladder to map week numbers 1 through 7 to the days of the week (Monday through Sunday). For any other week number, it sets `v_day` to 'Invalid'. The script concludes with an `END IF;` statement and a `DBMS_OUTPUT.PUT_LINE` call to display the week number and day. Below the editor, the 'Script output' tab is active, showing the execution results. It indicates the script elapsed in 00:00:00.007 seconds. The output text shows the SQL prompt, the declaration of variables, the start of the `BEGIN` block, and the final output: 'Week No: 3, Day: Wednesday'. A message at the bottom states 'PL/SQL procedure successfully completed.' and shows another elapsed time of 00:00:00.003 seconds.

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