

Name: Kamal Sharma

UID: 24BAI70380

Course: BE-CSE (AI&ML)

Subject: Database Management System

Experiment: Implementation of Conditional Logic using IF–ELSE, ELSIF Ladder, and CASE in PL/SQL

1. Aim of the Session

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 analyze decision-making capabilities in PL/SQL blocks.

2. Software Requirements

- **Database:**

- Oracle live SQL
- PostgreSQL Database (PgAdmin)

3. Objective of the Session

By the end of the session, the following objectives were achieved:

- To practice writing PL/SQL blocks with proper syntax and structure.
- To declare and initialize variables in PL/SQL effectively.
- To apply conditional logic using IF–ELSE, ELSIF ladder, and CASE statements for decision making.
- To display results using DBMS_OUTPUT for verification and debugging.

4. Practical / Experiment Steps

The experiment was carried out through the following activities:

1. **Variable Declaration:** Declared NUM, MARKS and DAY_NUMBER variables with appropriate data types.
2. **Initialization:** Assigned meaningful values to variables to simulate employee data.
3. **Output Display:** Printed variable values using DBMS_OUTPUT to confirm correct variable assignment.
4. **Conditional Logic:** Applied IF-ELSE,ELSIF ladder and CASE statements to demonstrate decision making.
5. **Result Display:** Printed output using DBMS_OUTPUT to validate the conditional logic.

5. Procedure of the Practical

Execution was performed in the following order:

1. **Environment Setup:** Logged into Oracle LIVE SQL Developer to prepare the workspace.
2. **PL/SQL Block Creation:** Wrote the PL/SQL block with variable declarations and initialization for each task.
3. **Conditional Execution:** Implemented logic for positivity check, grading system, performance evaluation, and day mapping.
4. **Output Verification:** Executed the block and verified the output for correctness.
5. **Documentation:** Saved the PL/SQL script and captured outputs for reporting and future reference.

6. I/O Analysis (Input / Output Analysis)

Input SQL Queries

TASK-1 (Number Check using IF_ELSE) :

```
DECLARE
```

```

NUM NUMBER := -24;

BEGIN

IF NUM > 0 THEN

    DBMS_OUTPUT.PUT_LINE('NUMBER IS POSITIVE');

ELSE

    DBMS_OUTPUT.PUT_LINE('NUMBER IS NON-POSITIVE');

END IF;

END;

```

TASK-2 (Grading System Using ELSIF ladder):

```

DECLARE

MARKS NUMBER := 57;

BEGIN

IF MARKS >= 95 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: A+');

ELSIF MARKS >= 90 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: A');

ELSIF MARKS >= 80 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: B+');

ELSIF MARKS >= 70 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: B');

ELSIF MARKS >= 60 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: C+');

ELSIF MARKS >= 50 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: C');

ELSIF MARKS >= 35 THEN

```

```
    DBMS_OUTPUT.PUT_LINE('GRADE: D');

ELSE

    DBMS_OUTPUT.PUT_LINE('GRADE: F');

END IF;

END;
```

TASK-3 (Performance status using ELSIF ladder):

```
DECLARE

MARKS NUMBER := 63;

BEGIN

IF MARKS >= 95 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: A+');

    DBMS_OUTPUT.PUT_LINE('PERFORMANCE: OUTSTANDING');

ELSIF MARKS >= 90 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: A');

    DBMS_OUTPUT.PUT_LINE('PERFORMANCE: EXCELLENT');

ELSIF MARKS >= 80 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: B+');

    DBMS_OUTPUT.PUT_LINE('PERFORMANCE: VERY GOOD');

ELSIF MARKS >= 70 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: B');

    DBMS_OUTPUT.PUT_LINE('PERFORMANCE: GOOD');

ELSIF MARKS >= 60 THEN

    DBMS_OUTPUT.PUT_LINE('GRADE: C+');

    DBMS_OUTPUT.PUT_LINE('PERFORMANCE: ABOVE AVERAGE');

ELSIF MARKS >= 50 THEN
```

```

DBMS_OUTPUT.PUT_LINE('GRADE: C');

DBMS_OUTPUT.PUT_LINE(' PERFORMANCE: AVERAGE');

ELSIF MARKS >= 35 THEN

DBMS_OUTPUT.PUT_LINE('GRADE: D');

DBMS_OUTPUT.PUT_LINE(' PERFORMANCE: NEEDS IMPROVEMENT');

ELSE

DBMS_OUTPUT.PUT_LINE('GRADE: F');

DBMS_OUTPUT.PUT_LINE(' PERFORMANCE: FAIL');

END IF;

END;

```

TASK-4 (Day-Mapping using CASE statements) :

```

DECLARE

DAY_NUMBER NUMBER := 10;

DAY_NAME VARCHAR2(20);

BEGIN

DAY_NAME := CASE DAY_NUMBER

WHEN 1 THEN 'SUNDAY'

WHEN 2 THEN 'MONDAY'

WHEN 3 THEN 'TUESDAY'

WHEN 4 THEN 'WEDNESDAY'

WHEN 5 THEN 'THURSDAY'

WHEN 6 THEN 'FRIDAY'

WHEN 7 THEN 'SATURDAY'

ELSE 'INVALID DAY NUMBER'

END;

```

```

DBMS_OUTPUT.PUT_LINE('THE DAY IS: ' || DAY_NAME);

END;

```

Output

Task-1:

<pre> 1 DECLARE 2 NUM NUMBER := 24; 3 BEGIN 4 IF NUM > 0 THEN 5 DBMS_OUTPUT.PUT_LINE('NUMBER IS POSITIVE'); 6 ELSE 7 DBMS_OUTPUT.PUT_LINE('NUMBER IS NON-POSITIVE'); 8 END IF; 9 END; 10 </pre>	<pre> 2 DECLARE 3 NUM NUMBER := -24; 4 BEGIN 5 IF NUM > 0 THEN 6 DBMS_OUTPUT.PUT_LINE('NUMBER IS POSITIVE'); 7 ELSE 8 DBMS_OUTPUT.PUT_LINE('NUMBER IS NON-POSITIVE'); 9 END IF; 10 </pre>
Query result	Query result
Script output	Script output
DBMS output	DBMS output
Explain Plan	Explain Plan
SQL	SQL
NUMBER IS POSITIVE	
NUMBER IS NON-POSITIVE	

Task-2:

<pre> 11 DECLARE 12 MARKS NUMBER := 87; 13 BEGIN 14 IF MARKS >= 95 THEN 15 DBMS_OUTPUT.PUT_LINE('GRADE: A+'); 16 ELSIF MARKS >= 90 THEN 17 DBMS_OUTPUT.PUT_LINE('GRADE: A'); 18 ELSIF MARKS >= 80 THEN 19 DBMS_OUTPUT.PUT_LINE('GRADE: B+'); 20 ELSIF MARKS >= 70 THEN 21 DBMS_OUTPUT.PUT_LINE('GRADE: B'); 22 ELSIF MARKS >= 60 THEN 23 DBMS_OUTPUT.PUT_LINE('GRADE: C+'); 24 ELSIF MARKS >= 50 THEN 25 DBMS_OUTPUT.PUT_LINE('GRADE: C'); 26 ELSIF MARKS >= 35 THEN 27 DBMS_OUTPUT.PUT_LINE('GRADE: D'); 28 ELSE 29 DBMS_OUTPUT.PUT_LINE('GRADE: F'); 30 END IF; 31 END; 32 </pre>	<pre> 11 DECLARE 12 MARKS NUMBER := 57; 13 BEGIN 14 IF MARKS >= 95 THEN 15 DBMS_OUTPUT.PUT_LINE('GRADE: A+'); 16 ELSIF MARKS >= 90 THEN 17 DBMS_OUTPUT.PUT_LINE('GRADE: A'); 18 ELSIF MARKS >= 80 THEN 19 DBMS_OUTPUT.PUT_LINE('GRADE: B+'); 20 ELSIF MARKS >= 70 THEN 21 DBMS_OUTPUT.PUT_LINE('GRADE: B'); 22 ELSIF MARKS >= 60 THEN 23 DBMS_OUTPUT.PUT_LINE('GRADE: C+'); 24 ELSIF MARKS >= 50 THEN 25 DBMS_OUTPUT.PUT_LINE('GRADE: C'); 26 ELSIF MARKS >= 35 THEN 27 DBMS_OUTPUT.PUT_LINE('GRADE: D'); 28 ELSE 29 DBMS_OUTPUT.PUT_LINE('GRADE: F'); 30 END IF; 31 END; 32 </pre>
Query result	Query result
Script output	Script output
DBMS output	DBMS output
Explain Plan	Explain Plan
GRADE: B+	
GRADE: C	

Task-3:

<pre> 33 DECLARE 34 MARKS NUMBER := 83; 35 BEGIN 36 IF MARKS >= 95 THEN 37 DBMS_OUTPUT.PUT_LINE('GRADE: A+'); 38 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: OUTSTANDING'); 39 ELSIF MARKS >= 90 THEN 40 DBMS_OUTPUT.PUT_LINE('GRADE: A'); 41 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: EXCELLENT'); 42 ELSIF MARKS >= 80 THEN 43 DBMS_OUTPUT.PUT_LINE('GRADE: B+'); 44 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: VERY GOOD'); 45 ELSIF MARKS >= 70 THEN 46 DBMS_OUTPUT.PUT_LINE('GRADE: B'); 47 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: GOOD'); 48 ELSIF MARKS >= 60 THEN 49 DBMS_OUTPUT.PUT_LINE('GRADE: C'); 50 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: ABOVE AVERAGE'); 51 ELSIF MARKS >= 50 THEN 52 DBMS_OUTPUT.PUT_LINE('GRADE: C'); 53 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: AVERAGE'); 54 ELSIF MARKS >= 35 THEN 55 DBMS_OUTPUT.PUT_LINE('GRADE: D'); 56 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: NEEDS IMPROVEMENT'); 57 ELSE 58 DBMS_OUTPUT.PUT_LINE('GRADE: F'); 59 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: FAIL'); 60 END IF; 61 END; </pre> <p>Query result Script output DBMS output Explain Plan SQL history</p> <p> </p> <p>GRADE: B+ PERFORMANCE: VERY GOOD</p>	<pre> 33 DECLARE 34 MARKS NUMBER := 63; 35 BEGIN 36 IF MARKS >= 95 THEN 37 DBMS_OUTPUT.PUT_LINE('GRADE: A+'); 38 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: OUTSTANDING'); 39 ELSIF MARKS >= 90 THEN 40 DBMS_OUTPUT.PUT_LINE('GRADE: A'); 41 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: EXCELLENT'); 42 ELSIF MARKS >= 80 THEN 43 DBMS_OUTPUT.PUT_LINE('GRADE: B+'); 44 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: VERY GOOD'); 45 ELSIF MARKS >= 70 THEN 46 DBMS_OUTPUT.PUT_LINE('GRADE: B'); 47 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: GOOD'); 48 ELSIF MARKS >= 60 THEN 49 DBMS_OUTPUT.PUT_LINE('GRADE: C+'); 50 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: ABOVE AVERAGE'); 51 ELSIF MARKS >= 50 THEN 52 DBMS_OUTPUT.PUT_LINE('GRADE: C'); 53 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: AVERAGE'); 54 ELSIF MARKS >= 35 THEN 55 DBMS_OUTPUT.PUT_LINE('GRADE: D'); 56 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: NEEDS IMPROVEMENT'); 57 ELSE 58 DBMS_OUTPUT.PUT_LINE('GRADE: F'); 59 DBMS_OUTPUT.PUT_LINE('PERFORMANCE: FAIL'); 60 END IF; 61 END; </pre> <p>Query result Script output DBMS output Explain Plan SQL history</p> <p> </p> <p>GRADE: C+ PERFORMANCE: ABOVE AVERAGE</p>
--	---

Task-4:

<pre> 63 DECLARE 64 DAY_NUMBER NUMBER := 4; 65 DAY_NAME VARCHAR2(20); 66 BEGIN 67 DAY_NAME := CASE DAY_NUMBER 68 WHEN 1 THEN 'SUNDAY' 69 WHEN 2 THEN 'MONDAY' 70 WHEN 3 THEN 'TUESDAY' 71 WHEN 4 THEN 'WEDNESDAY' 72 WHEN 5 THEN 'THURSDAY' 73 WHEN 6 THEN 'FRIDAY' 74 WHEN 7 THEN 'SATURDAY' 75 ELSE 'INVALID DAY NUMBER' 76 END; 77 DBMS_OUTPUT.PUT_LINE('THE DAY IS: ' DAY_NAME); 78 END; 79 </pre> <p>Query result Script output DBMS output Explain Plan</p> <p> </p> <p>THE DAY IS: WEDNESDAY</p>	<pre> 63 DECLARE 64 DAY_NUMBER NUMBER := 10; 65 DAY_NAME VARCHAR2(20); 66 BEGIN 67 DAY_NAME := CASE DAY_NUMBER 68 WHEN 1 THEN 'SUNDAY' 69 WHEN 2 THEN 'MONDAY' 70 WHEN 3 THEN 'TUESDAY' 71 WHEN 4 THEN 'WEDNESDAY' 72 WHEN 5 THEN 'THURSDAY' 73 WHEN 6 THEN 'FRIDAY' 74 WHEN 7 THEN 'SATURDAY' 75 ELSE 'INVALID DAY NUMBER' 76 END; 77 DBMS_OUTPUT.PUT_LINE('THE DAY IS: ' DAY_NAME); 78 END; 79 </pre> <p>Query result Script output DBMS output Explain Plan</p> <p> </p> <p>THE DAY IS: INVALID DAY NUMBER</p>
---	---

7. Learning Outcome

From this practical, the following knowledge and skills were gained:

- Learned how to declare and initialize variables in PL/SQL with appropriate data types.
- Understood the use of conditional logic using IF–ELSE, ELSIF ladder, and CASE statements to control program flow.
- Practiced grading and performance evaluation, reinforcing the concept of conditional branching.
- Gained experience in displaying results using DBMS_OUTPUT, which is essential for debugging and verification.
- Strengthened understanding of PL/SQL procedural programming concepts, preparing for more complex database programming tasks.
- Developed the ability to translate real-world business rules (grading, day mapping, tax calculation) into executable PL/SQL code.