

# Experiment 4

Student Name: Nandesh T  
Branch: CSE - AIML  
Semester: 4  
Subject Name: DBMS

UID: 24BAI70121  
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

## Objectives

- 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 Input Queries

```

DECLARE
NUM NUMBER:= -21;

BEGIN
  IF NUM > 0 THEN
    DBMS_OUTPUT.PUT_LINE('IT IS A POSITIVE NUMBER');
  ELSE
    DBMS_OUTPUT.PUT_LINE('IT IS A NON-POSITIVE NUMBER');
  END IF;
END;
```

# Output

The screenshot shows the Oracle FreeSQL web interface at 3:57. The browser address bar displays 'freesql.com'. The interface includes a 'Navigator' on the left with a tree view of 'Human Resources (HR)' tables: HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB\_HISTORY, HR.LOCATIONS, and HR.REGIONS. The main editor, titled '[ SQL Worksheet ]\*', contains the following SQL code:

```
1 DECLARE
2   NUM NUMBER:=10;
3
4   BEGIN
5     IF NUM>0 THEN
6       DBMS_OUTPUT.PUT_LINE('IT IS A POSITIVE NUMBER');
7     ELSE
8       DBMS_OUTPUT.PUT_LINE('IT IS A NON-POSITIVE NUMBER');
9     END IF;
10  END;
```

Below the editor, the 'Script output' tab is active, showing the execution results:

```
SQL> DECLARE
      NUM NUMBER:=10;
      BEGIN...
Show more...

IT IS A POSITIVE NUMBER

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.004
```

On the right, the 'Library' section shows two tutorials: 'Creating Tables: Databases for...' (created 8 years ago, 3 likes, 8.7K executions) and 'Joining Tables: Databases for...' (created 7 years ago, 8 likes, 1.8K executions).

The screenshot shows the Oracle FreeSQL web interface at 3:59. The browser address bar displays 'freesql.com'. The interface is identical to the previous one, but the SQL code in the main editor has been updated to use a different input value:

```
1 DECLARE
2   NUM NUMBER:=21;
3
4   BEGIN
5     IF NUM>0 THEN
6       DBMS_OUTPUT.PUT_LINE('IT IS A POSITIVE NUMBER');
7     ELSE
8       DBMS_OUTPUT.PUT_LINE('IT IS A NON-POSITIVE NUMBER');
9     END IF;
10  END;
```

The 'Script output' tab shows the execution results for this new code:

```
PL/SQL procedure successfully completed.
Elapsed: 00:00:00.004

SQL> DECLARE
      NUM NUMBER:=21;
      BEGIN...
Show more...

IT IS A NON-POSITIVE NUMBER
```

The 'Library' section on the right remains the same, displaying the same two tutorials.

## SQL Queries Input

DECLARE

MARKS NUMBER:=68;

GRADE VARCHAR(1);

BEGIN

IF MARKS>=90 THEN

GRADE:='A';

ELSIF MARKS>=80 THEN

GRADE:='B';

ELSIF MARKS>=70 THEN

GRADE:='C';

ELSIF MARKS>=60 THEN

GRADE:='D';

ELSE

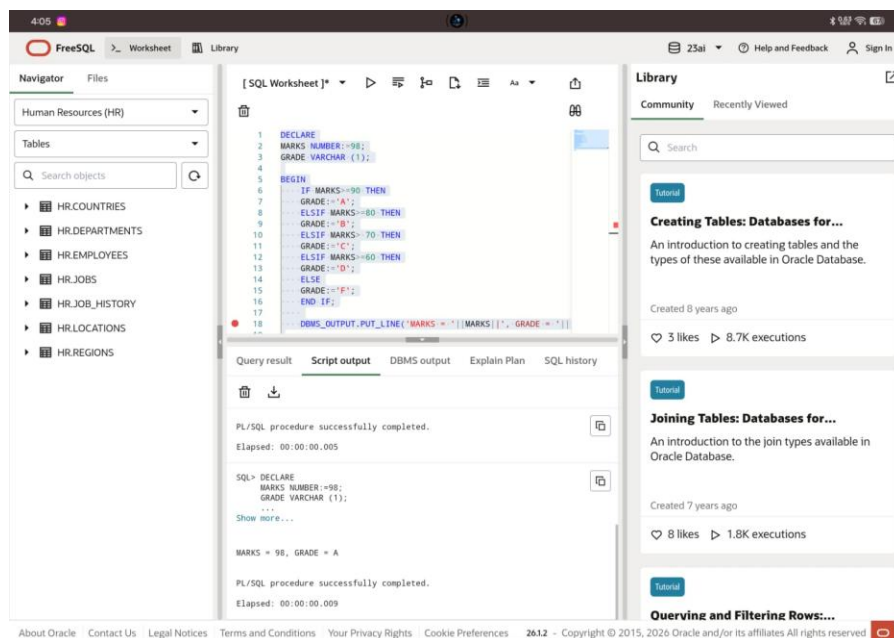
GRADE:='F';

END IF;

DBMS\_OUTPUT.PUT\_LINE('MARKS = '||MARKS||', GRADE = '||GRADE);

END;

## Output



The screenshot displays the FreeSQL web interface. On the left, a sidebar shows a database structure for 'Human Resources (HR)' with tables like HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB\_HISTORY, HR.LOCATIONS, and HR.REGIONS. The main area shows a SQL worksheet with the following code:

```
1 DECLARE
2 MARKS NUMBER:=98;
3 GRADE VARCHAR (1);
4
5 BEGIN
6 IF MARKS<=90 THEN
7 GRADE:='A';
8 ELSIF MARKS<=80 THEN
9 GRADE:='B';
10 ELSIF MARKS<=70 THEN
11 GRADE:='C';
12 ELSIF MARKS<=60 THEN
13 GRADE:='D';
14 ELSE
15 GRADE:='F';
16 END IF;
17
18 DBMS_OUTPUT.PUT_LINE('MARKS = '||MARKS||', GRADE = '||GRADE);
19
```

The 'Script output' tab is active, showing the execution results:

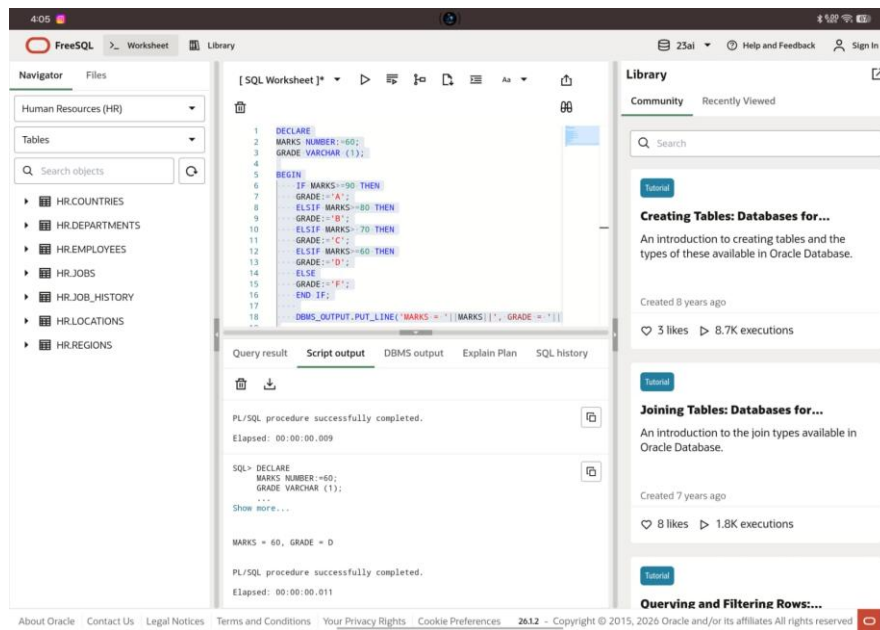
```
PL/SQL procedure successfully completed.
Elapsed: 00:00:00.005

SQL> DECLARE
MARKS NUMBER:=98;
GRADE VARCHAR (1);
...
Show more...

MARKS = 98, GRADE = A

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.009
```

On the right, a 'Library' section shows a list of tutorials, including 'Creating Tables: Databases for...' and 'Joining Tables: Databases for...'. The bottom of the page contains a footer with links to 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', 'Cookie Preferences', and a copyright notice for 2015, 2020 Oracle and/or its affiliates.



## SQL Queries Input

DECLARE

MARKS NUMBER:=58;

PERFORMANCE VARCHAR(20);

BEGIN

IF MARKS>=75 THEN

PERFORMANCE:='DISTINCTION';

ELSIF MARKS>=60 THEN

PERFORMANCE:='FIRST CLASS';

ELSIF MARKS>=50 THEN

PERFORMANCE:='SECOND CLASS';

ELSIF MARKS>=35 THEN

PERFORMANCE:='PASS';

ELSE

PERFORMANCE:='FAIL';

END IF;

DBMS\_OUTPUT.PUT\_LINE('MARKS = '||MARKS||' AND PERFORMANCE  
= '||PERFORMANCE);  
END;

# Output

The image displays two screenshots of the Oracle SQL Developer interface, showing the execution of an SQL query and its results.

**Top Screenshot:**

- Navigator:** Shows the 'Human Resources (HR)' schema with tables like HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB\_HISTORY, HR.LOCATIONS, and HR.REGIONS.
- SQL Worksheet:** Contains the following SQL code:

```
1 DECLARE
2 MARKS NUMBER:=38;
3 PERFORMANCE VARCHAR (20);
4
5 BEGIN
6 IF MARKS<=75 THEN
7 PERFORMANCE:='DISTINCTION';
8 ELSEIF MARKS<=60 THEN
9 PERFORMANCE:='FIRST CLASS';
10 ELSEIF MARKS<=50 THEN
11 PERFORMANCE:='SECOND CLASS';
12 ELSEIF MARKS<=35 THEN
13 PERFORMANCE:='PASS';
14 ELSE
15 PERFORMANCE:='FAIL';
16 END IF;
17 DBMS_OUTPUT.PUT_LINE('MARKS: '||MARKS||', PERFORMANCE: '||PERFORMANCE);
18
```
- Query result:** Shows the output of the query:

```
PL/SQL procedure successfully completed.
Elapsed: 00:00:00.008

SQL> DECLARE
MARKS NUMBER:=38;
PERFORMANCE VARCHAR (20);
...
Show more...

MARKS = 38, PERFORMANCE = PASS

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.007
```
- Library:** Shows a list of tutorials, including 'Creating Tables: Databases for...', 'Joining Tables: Databases for...', and 'Quervining and Filtering Rows...'.

**Bottom Screenshot:**

- Navigator:** Same as the top screenshot.
- SQL Worksheet:** Contains the same SQL code as the top screenshot.
- Query result:** Shows the output of the query:

```
PL/SQL procedure successfully completed.
Elapsed: 00:00:00.008

SQL> DECLARE
MARKS NUMBER:=88;
PERFORMANCE VARCHAR (20);
...
Show more...

MARKS = 88, PERFORMANCE = DISTINCTION

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.008
```
- Library:** Same as the top screenshot.

SQL Queries Input  
DECLARE  
DAYNUM NUMBER:=3;  
DAYNAME VARCHAR(20);

BEGIN

```

DAYNAME:=CASE DAYNUM
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'
END;

```

```

DBMS_OUTPUT.PUT_LINE('IT IS '||DAYNAME);
END;

```

## Output

The screenshot displays the FreeSQL web interface. On the left, a 'Navigator' pane shows a tree structure for 'Human Resources (HR)' with tables like HR.COUNTRIES, HR.DEPARTMENTS, HR.EMPLOYEES, HR.JOBS, HR.JOB\_HISTORY, HR.LOCATIONS, and HR.REGIONS. The main area shows a SQL script in a 'Worksheet' editor. The script is as follows:

```

1 DECLARE
2   DAYNUM NUMBER:=3;
3   DAYNAME VARCHAR (20);
4 BEGIN
5   DAYNAME:=CASE DAYNUM
6     WHEN 1 THEN 'SUNDAY'
7     WHEN 2 THEN 'MONDAY'
8     WHEN 3 THEN 'TUESDAY'
9     WHEN 4 THEN 'WEDNESDAY'
10    WHEN 5 THEN 'THURSDAY'
11    WHEN 6 THEN 'FRIDAY'
12    WHEN 7 THEN 'SATURDAY'
13    ELSE 'INVALID DAY'
14  END;
15
16  DBMS_OUTPUT.PUT_LINE ('IT IS '||DAYNAME);
17 END;

```

Below the script, the 'Script output' tab is active, showing the execution results:

```

SQL> DECLARE
      DAYNUM NUMBER:=3;
      DAYNAME VARCHAR (20);
      BEGIN...
Show more...

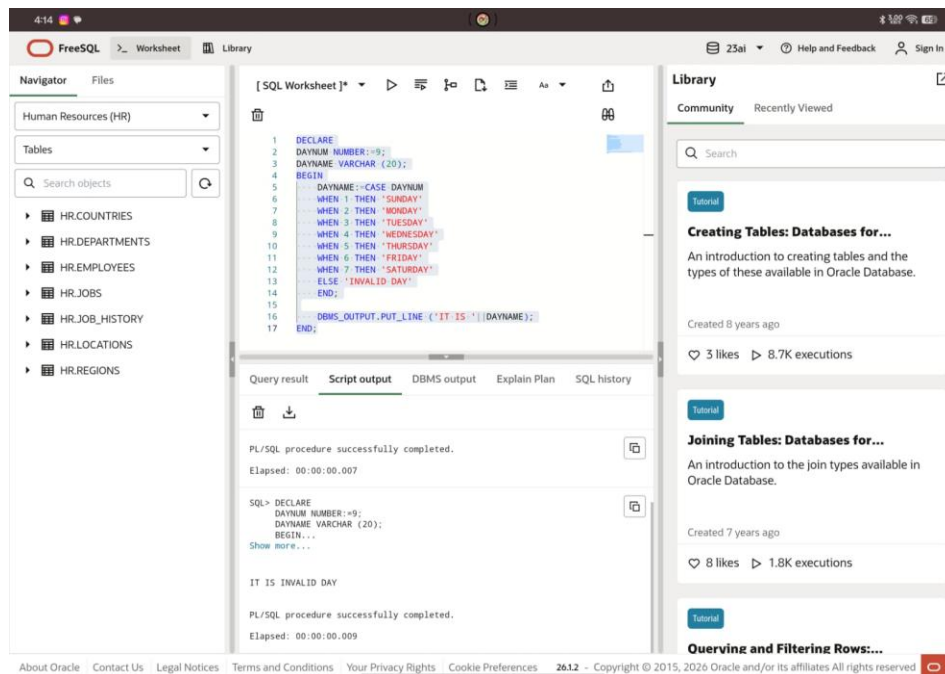
IT IS TUESDAY

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.007

```

On the right side, a 'Library' pane shows a list of tutorials, including 'Creating Tables: Databases for...', 'Joining Tables: Databases for...', and 'Quervoring and Filtering Rows...'. The bottom of the interface contains a footer with links to 'About Oracle', 'Contact Us', 'Legal Notices', 'Terms and Conditions', 'Your Privacy Rights', and 'Cookie Preferences', along with the version number '26.12' and copyright information for 2015, 2026 Oracle and/or its affiliates.





## 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