Experiment 3

Student Name: Diwansh
Branch: CSE - AIML
Semester: 4
Subject Name: DBMS

UID: 24BAI70294
Section/Group: 24AIT_KRG G1
Date of Performance: 28/1/26
Subject Code: 24CSH-298

AIM:
To understand the basic structure of a PL/SQL program by creating and executing a simple PL/SQL block that includes declaration and execution sections, performs calculations, and uses conditional statements to make decisions.

OBJECTIVES:
● To create a simple PL/SQL program demonstrating **Declaration Section** and **Execution Section**.
● To understand the use of **conditional statements (IF–ELSE)** in PL/SQL for decision-making operations.
● To perform salary-based calculations inside a PL/SQL block.

SOFTWARE REQUIREMENTS:
• Oracle FreeSQL

PRACTICAL / EXPERIMENT STEPS:
1. The PL/SQL block is started using the **DECLARE** keyword to define required variables.

2. Variables such as worker ID, worker name, and basic pay are declared with appropriate data types and initialized with values.
3. A variable for **House Rent Allowance (HRA)** is declared to store the calculated value.
4. Worker details like ID, name, and basic pay are displayed using the **DBMS_OUTPUT.PUT_LINE** procedure.
5. HRA is calculated as **40% of the basic pay**.
6. An **IF–ELSE** conditional statement is used to check whether tax is applicable.
7. If the basic pay is greater than **60000**, tax is applicable; otherwise, it is not.
8. The calculated HRA and tax status are displayed as output.
9. The PL/SQL block ends successfully after execution.

PROCEDURE:

1. Start the Oracle FreeSQL environment.
2. Write the code to declare the required variables.

```
DECLARE
    WORKER_ID NUMBER := 555;
    WORKER_NAME VARCHAR2(25) := 'DIWANSH';
    BASIC_PAY NUMBER := 48000;
    HRA_AMOUNT NUMBER;
```

3. Inside the **BEGIN** block, write the code to calculate HRA.
4. Display the worker details using **DBMS_OUTPUT.PUT_LINE**.

```
BEGIN
    HRA_AMOUNT := BASIC_PAY * 0.40;

    DBMS_OUTPUT.PUT_LINE('WORKER ID: ' || WORKER_ID);
    DBMS_OUTPUT.PUT_LINE('WORKER NAME: ' || WORKER_NAME);
    DBMS_OUTPUT.PUT_LINE('BASIC PAY: RS. ' || BASIC_PAY);
    DBMS_OUTPUT.PUT_LINE('HOUSE RENT ALLOWANCE: RS. ' || HRA_AMOUNT);
```

5. Use the **IF–ELSE** statement to check tax applicability based on basic pay.
6. Display the HRA and tax message.

```
IF BASIC_PAY > 60000 THEN
    DBMS_OUTPUT.PUT_LINE('TAX APPLICABLE');
ELSE
    DBMS_OUTPUT.PUT_LINE('TAX NOT APPLICABLE');
END IF;
```

7. End the PL/SQL block using the **END;** statement and execute the program.

```
END;
/
```

8. Verify the output displayed on the screen.

CODE:

```
DECLARE
    WORKER_ID NUMBER := 555;
    WORKER_NAME VARCHAR2(25) := 'DIWANSH';
    BASIC_PAY NUMBER := 48000;
    HRA_AMOUNT NUMBER;

BEGIN
    HRA_AMOUNT := BASIC_PAY * 0.40;

    DBMS_OUTPUT.PUT_LINE('WORKER ID: ' || WORKER_ID);
    DBMS_OUTPUT.PUT_LINE('WORKER NAME: ' || WORKER_NAME);
    DBMS_OUTPUT.PUT_LINE('BASIC PAY: RS. ' || BASIC_PAY);
    DBMS_OUTPUT.PUT_LINE('HOUSE RENT ALLOWANCE: RS. ' || HRA_AMOUNT);

    IF BASIC_PAY > 60000 THEN
        DBMS_OUTPUT.PUT_LINE('TAX APPLICABLE');
    ELSE
        DBMS_OUTPUT.PUT_LINE('TAX NOT APPLICABLE');
    END IF;

END;
```

I/O ANALYSIS:

This PL/SQL program demonstrates the basic structure of a PL/SQL block. It declares variables for worker details in the **DECLARE** section and performs HRA calculation in the **BEGIN** section. The program calculates **House Rent Allowance as 40% of basic pay** and checks tax applicability using an **IF–ELSE** condition. Finally, it displays worker ID, name, basic pay, HRA amount, and tax status using **DBMS_OUTPUT.PUT_LINE**.

```
WORKER ID: 555
WORKER NAME: DIWANSH
BASIC PAY: RS. 48000
HOUSE RENT ALLOWANCE: RS. 19200
TAX NOT APPLICABLE


PL/SQL procedure successfully completed.

Elapsed: 00:00:00.001
```

LEARNING OUTCOMES:

1. Understand the basic structure of a PL/SQL block, including Declaration and Execution sections.
2. Gain hands-on experience in declaring variables and performing calculations in PL/SQL.
3. Learn to use conditional statements (IF–ELSE) for decision-making.
4. Learn to display output using built-in PL/SQL procedures like **DBMS_OUTPUT.PUT_LINE**.