Name: Kamal Sharma

UID: 24BAI70380

Course: BE-CSE (AI&ML)

Subject: Database Management System

Experiment: Implementation of Conditional Logic using IF–ELSE in PL/SQL

1. Aim of the Session

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, and to display output using built-in procedures.

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-ELSIF-ELSE statements for decision making.
- To display results using DBMS_OUTPUT for verification and debugging.

4. Practical / Experiment Steps

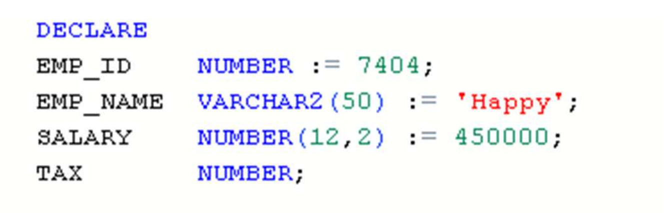
The experiment was carried out through the following activities:

1. **Variable Declaration**: Declared EMP_ID, EMP_NAME, SALARY, and TAX variables with appropriate data types.
2. **Initialization**: Assigned meaningful values to EMP_ID, EMP_NAME, and SALARY to simulate employee data.
3. **Output Display**: Printed employee details using DBMS_OUTPUT to confirm correct variable assignment.
4. **Conditional Logic**: Applied IF-ELSIF-ELSE statements to calculate tax based on salary slabs, demonstrating control flow.
5. **Result Display**: Printed calculated tax using DBMS_OUTPUT to validate the conditional logic.

5. Procedure of the Practical

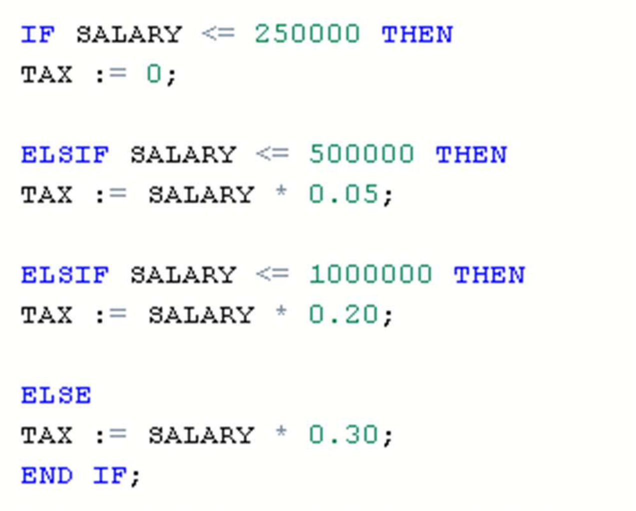
Execution was performed in the following order:

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.**



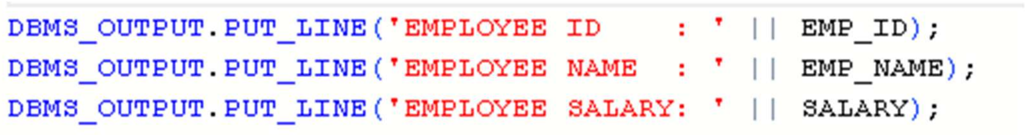
```
DECLARE
EMP_ID     NUMBER := 7404;
EMP_NAME   VARCHAR2(50)  := 'Happy';
SALARY     NUMBER(12,2)  := 450000;
TAX        NUMBER;
```

3. **Conditional Execution: Implemented salary-based tax calculation logic using IF-ELSIF-ELSE.**

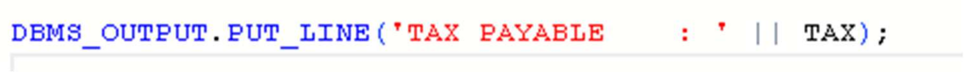


```
IF SALARY <= 250000 THEN
TAX := 0;

ELSIF SALARY <= 500000 THEN
TAX := SALARY * 0.05;

ELSIF SALARY <= 1000000 THEN
TAX := SALARY * 0.20;

ELSE
TAX := SALARY * 0.30;
END IF;
```

4. **Output Verification: Executed the block and verified the output for correctness.**



```
DBMS_OUTPUT.PUT_LINE('EMPLOYEE ID     : ' || EMP_ID);
DBMS_OUTPUT.PUT_LINE('EMPLOYEE NAME   : ' || EMP_NAME);
DBMS_OUTPUT.PUT_LINE('EMPLOYEE SALARY: ' || SALARY);
```



```
DBMS_OUTPUT.PUT_LINE('TAX PAYABLE     : ' || TAX);
```

5. **Documentation: Saved the PL/SQL script and captured outputs for reporting and future reference.**

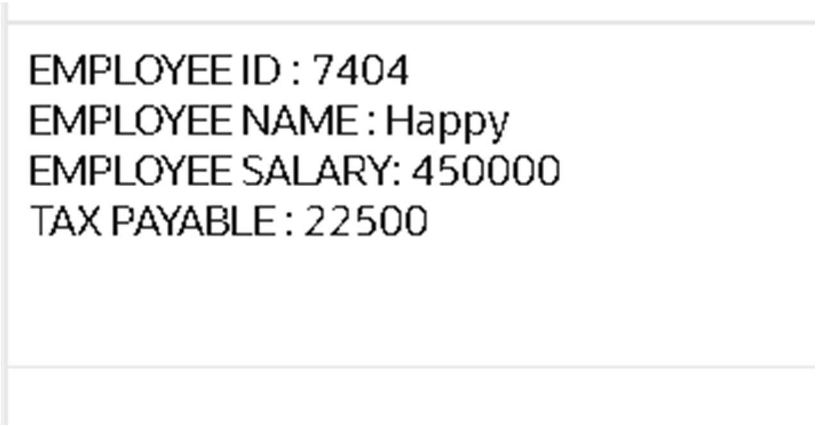
6. I/O Analysis (Input / Output Analysis)

Input SQL Queries

```
DECLARE

EMP_ID     NUMBER:= 7404;

EMP_NAME VARCHAR2(50)  := 'Happy';

SALARY     NUMBER(12,2): = 450000;

TAX        NUMBER;
```

```
BEGIN

DBMS_OUTPUT.PUT_LINE('EMPLOYEE ID    : ' || EMP_ID);

DBMS_OUTPUT.PUT_LINE('EMPLOYEE NAME  : ' || EMP_NAME);

DBMS_OUTPUT.PUT_LINE('EMPLOYEE SALARY: ' || SALARY);


IF SALARY <= 250000 THEN

TAX := 0;

ELSIF SALARY <= 500000 THEN

TAX := SALARY * 0.05;

ELSIF SALARY <= 1000000 THEN

TAX := SALARY * 0.20;

ELSE

TAX := SALARY * 0.30;

END IF;


DBMS_OUTPUT.PUT_LINE('TAX PAYABLE    : ' || TAX);

END;
```

Output



```
EMPLOYEE ID : 7404
EMPLOYEE NAME : Happy
EMPLOYEE SALARY: 450000
TAX PAYABLE : 22500
```

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-ELSIF-ELSE statements to control program flow.
- Practiced tax calculation based on salary ranges, reinforcing the concept of conditional branching.
- Gained experience in displaying results using DBMS_OUTPUT, which is essential for debugging and output 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 into executable PL/SQL code.