NAME:=SHREYASI GIDMARE

PRN:=23070521146

Introduction to PL/SQL *Conditions*

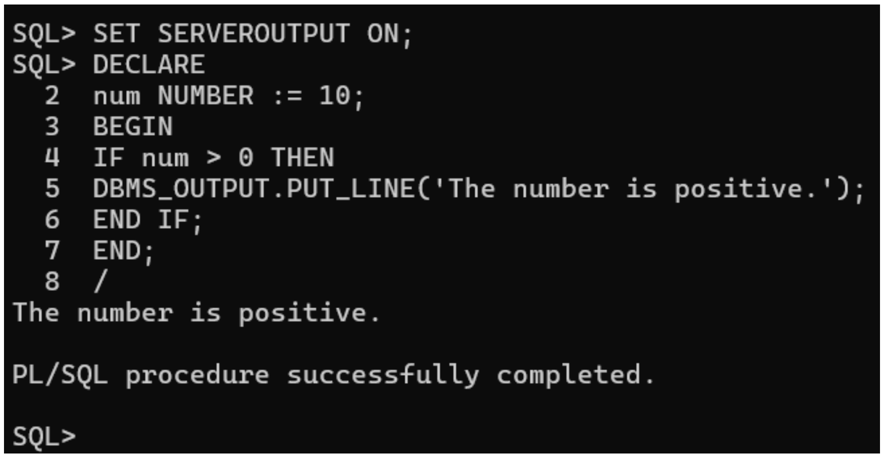
In PL/SQL, conditions allow decision-making in programs. The two main types of conditional statements are: **IF-THEN**
IF-THEN-ELSE
IF-THEN-ELSIF-ELSE
CASE Statement

IF-THEN Statement

Executes a block of code if the condition is `TRUE`.

Example: Check if a number is positive

```
SET SERVEROUTPUT ON;
 DECLARE
    num NUMBER := 10;
BEGIN
    IF num > 0 THEN
        DBMS_OUTPUT.PUT_LINE('The number is positive.');
    END IF;
END; /
```



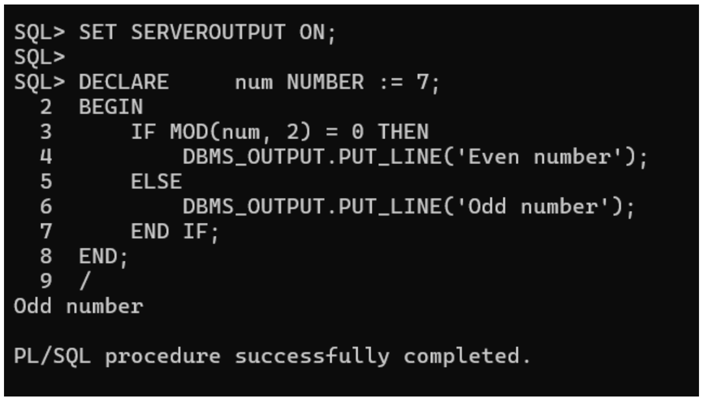
```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2  num NUMBER := 10;
  3  BEGIN
  4  IF num > 0 THEN
  5  DBMS_OUTPUT.PUT_LINE('The number is positive.');
  6  END IF;
  7  END;
  8  /
The number is positive.

PL/SQL procedure successfully completed.

SQL>
```

IF-THEN-ELSE Statement

Executes one block if the condition is `TRUE`, otherwise executes another block.

Example: Check if a number is even or odd

```
SET SERVEROUTPUT ON;

DECLARE      num
NUMBER := 7;
BEGIN
    IF MOD(num, 2) = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Even number');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Odd number');
    END IF;
END; /
```



```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> DECLARE     num NUMBER := 7;
  2  BEGIN
  3      IF MOD(num, 2) = 0 THEN
  4          DBMS_OUTPUT.PUT_LINE('Even number');
  5      ELSE
  6          DBMS_OUTPUT.PUT_LINE('Odd number');
  7      END IF;
  8  END;
  9  /
Odd number

PL/SQL procedure successfully completed.
```

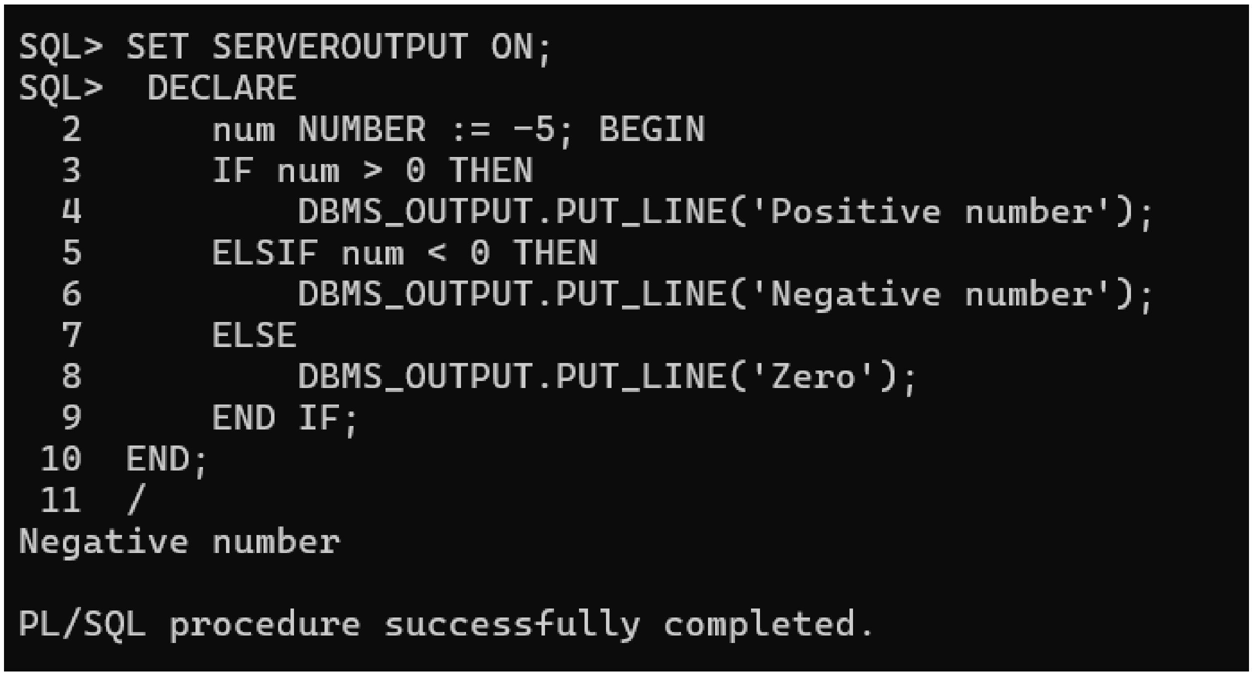
IF-THEN-ELSIF-ELSE Statement

Check multiple conditions one by one.

Example: Check if a number is positive, negative, or zero

```
SET SERVEROUTPUT ON;
 DECLARE
    num NUMBER := -5; BEGIN
    IF num > 0 THEN
```

```
        DBMS_OUTPUT.PUT_LINE('Positive number');
    ELSIF num < 0 THEN
        DBMS_OUTPUT.PUT_LINE('Negative number');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Zero');
    END IF;
END; /
```



```
SQL> SET SERVEROUTPUT ON;
SQL>  DECLARE
        num NUMBER := -5; BEGIN
        IF num > 0 THEN
            DBMS_OUTPUT.PUT_LINE('Positive number');
        ELSIF num < 0 THEN
            DBMS_OUTPUT.PUT_LINE('Negative number');
        ELSE
            DBMS_OUTPUT.PUT_LINE('Zero');
        END IF;
    END;
    /
Negative number

PL/SQL procedure successfully completed.
```

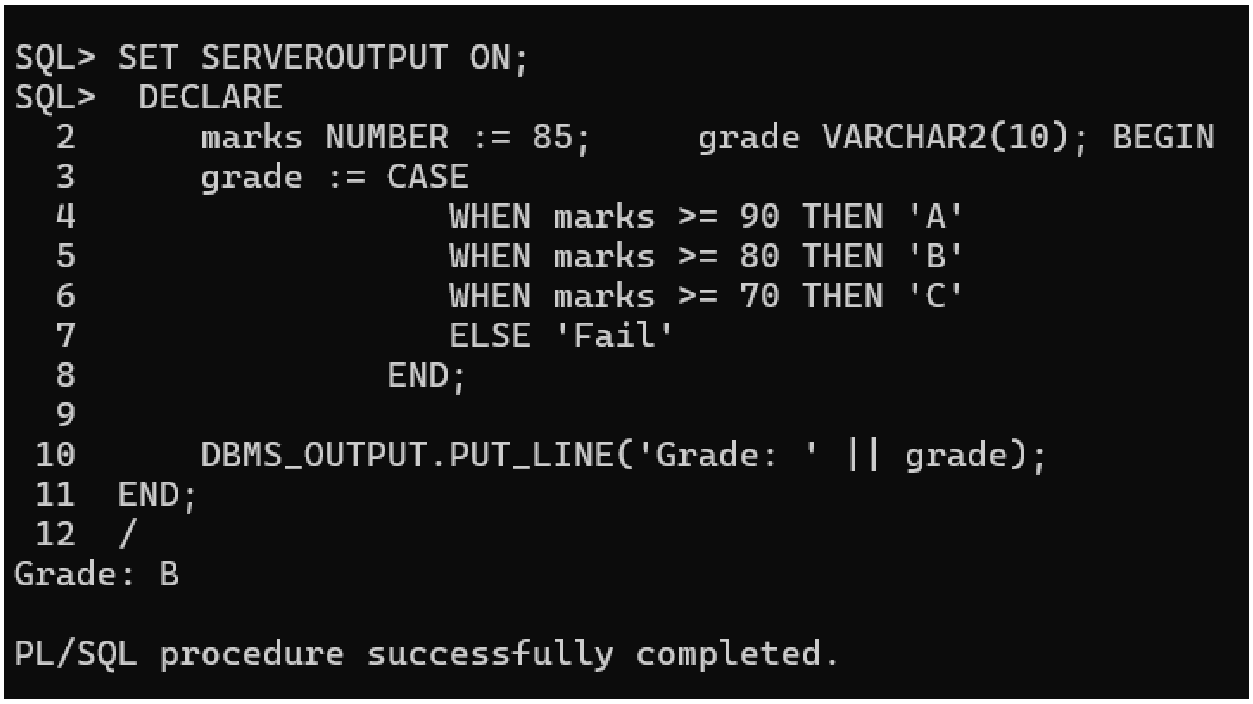
CASE Statement

The `CASE` statement is used to handle multiple conditions more efficiently.

Example: Grade Calculation Using CASE

```
SET SERVEROUTPUT ON;
 DECLARE
    marks NUMBER := 85;
grade VARCHAR2(10); BEGIN
    grade := CASE
                WHEN marks >= 90 THEN 'A'
```

```
                    WHEN marks >= 80 THEN 'B'
                    WHEN marks >= 70 THEN 'C'
                    ELSE 'Fail'
                END;

    DBMS_OUTPUT.PUT_LINE('Grade: ' || grade);
END; /
```

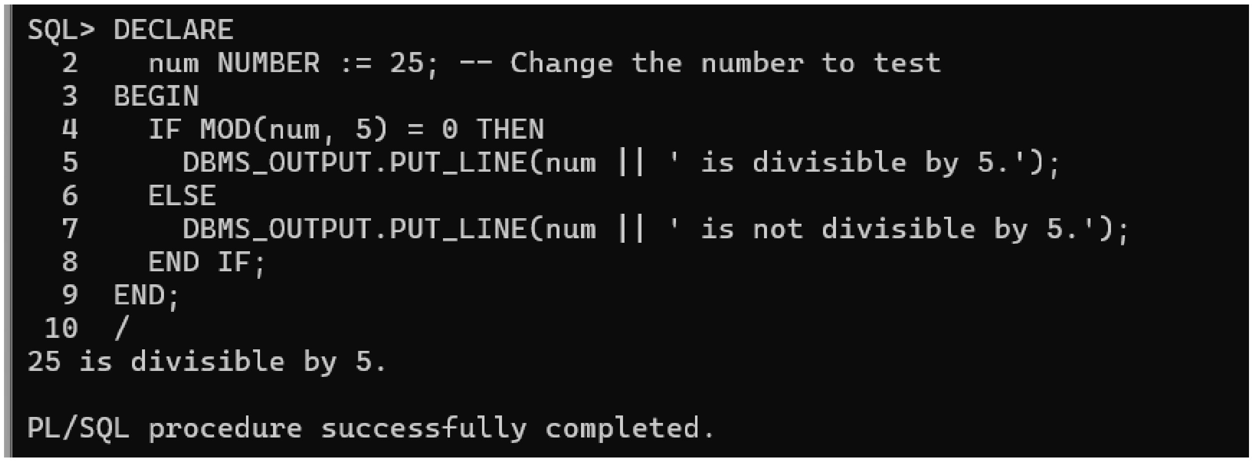


```
SQL> SET SERVEROUTPUT ON;
SQL>  DECLARE
        marks NUMBER := 85;     grade VARCHAR2(10); BEGIN
        grade := CASE
                    WHEN marks >= 90 THEN 'A'
                    WHEN marks >= 80 THEN 'B'
                    WHEN marks >= 70 THEN 'C'
                    ELSE 'Fail'
                END;

        DBMS_OUTPUT.PUT_LINE('Grade: ' || grade);
     END;
     /
Grade: B

PL/SQL procedure successfully completed.
```

BBB B B

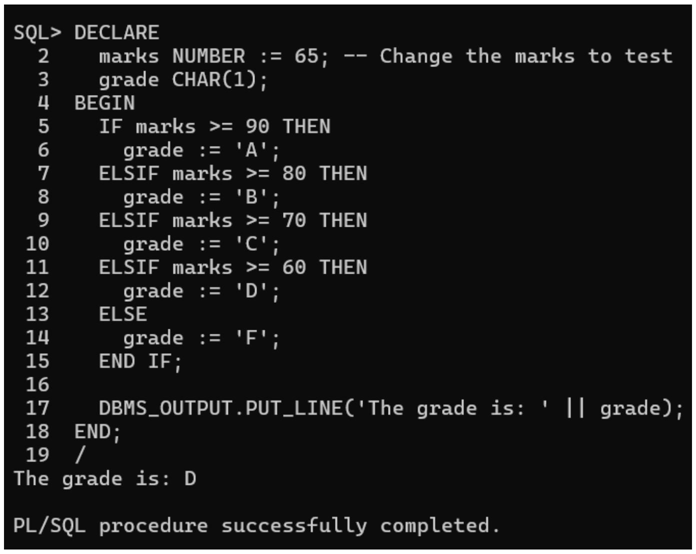
Simple Tasks for Practice

1. Write a PL/SQL program to check whether a number is **divisible by 5**.



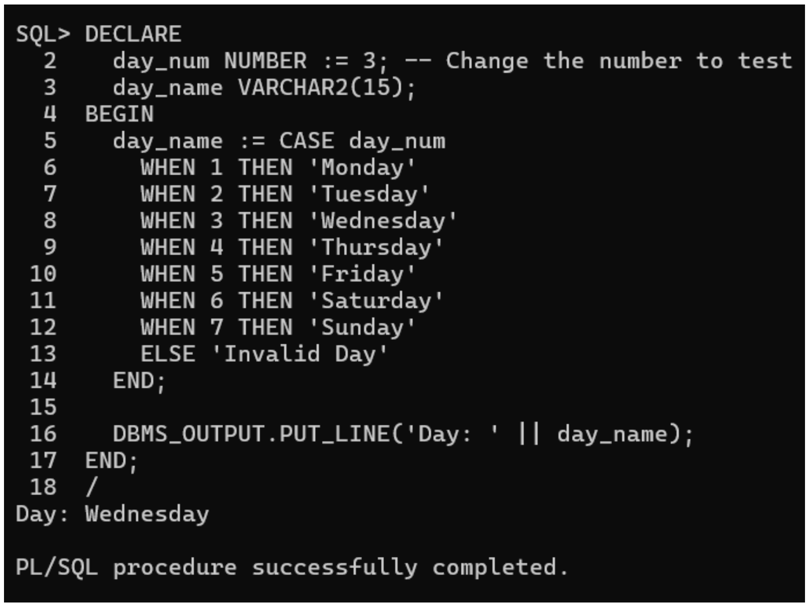
```
SQL> DECLARE
       num NUMBER := 25; -- Change the number to test
     BEGIN
       IF MOD(num, 5) = 0 THEN
         DBMS_OUTPUT.PUT_LINE(num || ' is divisible by 5.');
       ELSE
         DBMS_OUTPUT.PUT_LINE(num || ' is not divisible by 5.');
       END IF;
     END;
     /
25 is divisible by 5.

PL/SQL procedure successfully completed.
```

2. Modify the **grade program** to include more conditions (e.g., `60-70` for **D**, `below 60` for **F**).



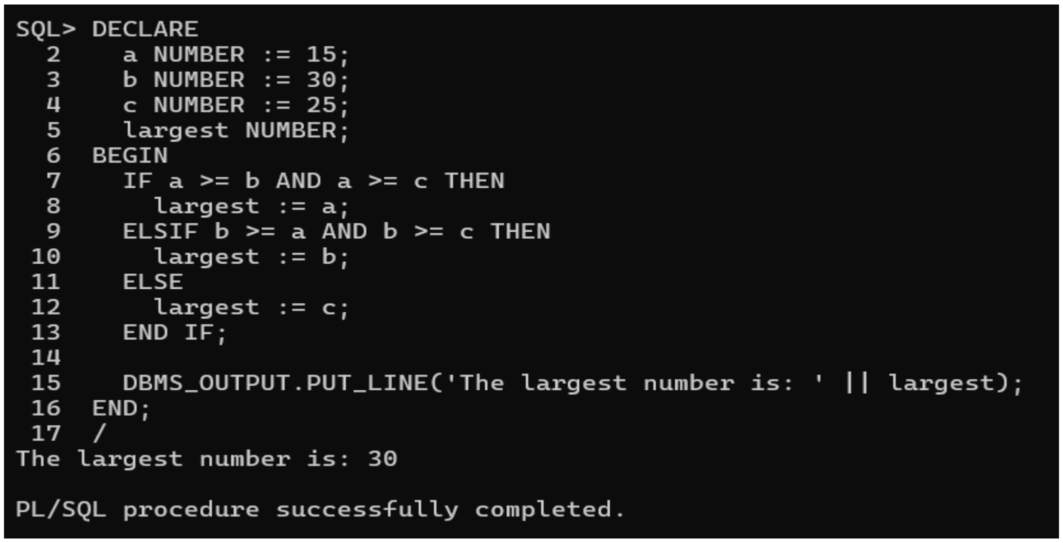
```
SQL> DECLARE
    marks NUMBER := 65; -- Change the marks to test
    grade CHAR(1);
  BEGIN
    IF marks >= 90 THEN
      grade := 'A';
    ELSIF marks >= 80 THEN
      grade := 'B';
    ELSIF marks >= 70 THEN
      grade := 'C';
    ELSIF marks >= 60 THEN
      grade := 'D';
    ELSE
      grade := 'F';
    END IF;

    DBMS_OUTPUT.PUT_LINE('The grade is: ' || grade);
  END;
  /
The grade is: D

PL/SQL procedure successfully completed.
```

3. Write a **CASE statement** to display the day of the week based on a number input (1 = Monday, 2 = Tuesday, etc.).



```
SQL> DECLARE
    day_num NUMBER := 3; -- Change the number to test
    day_name VARCHAR2(15);
  BEGIN
    day_name := CASE day_num
      WHEN 1 THEN 'Monday'
      WHEN 2 THEN 'Tuesday'
      WHEN 3 THEN 'Wednesday'
      WHEN 4 THEN 'Thursday'
      WHEN 5 THEN 'Friday'
      WHEN 6 THEN 'Saturday'
      WHEN 7 THEN 'Sunday'
      ELSE 'Invalid Day'
    END;

    DBMS_OUTPUT.PUT_LINE('Day: ' || day_name);
  END;
  /
Day: Wednesday

PL/SQL procedure successfully completed.
```

4. Create a program that **checks the largest of three numbers** using `IF-THEN-ELSIF`.



```
SQL> DECLARE
       a NUMBER := 15;
       b NUMBER := 30;
       c NUMBER := 25;
       largest NUMBER;
     BEGIN
       IF a >= b AND a >= c THEN
         largest := a;
       ELSIF b >= a AND b >= c THEN
         largest := b;
       ELSE
         largest := c;
       END IF;

       DBMS_OUTPUT.PUT_LINE('The largest number is: ' || largest);
     END;
     /
The largest number is: 30

PL/SQL procedure successfully completed.
```