**What Is PL/SQL?**

**PL/SQL (Procedural Language/SQL)** is Oracle’s extension to SQL that allows procedural programming features like loops, conditions, functions, and error handling within SQL statements.

**Key Features:**

* **Portable**: Runs across all Oracle environments
* **Efficient**: Reduces network traffic with block execution
* **Exception Handling**: Robust error management
* **Development Tools**: Integrated with Oracle IDEs
* **Error Checking**: Compile-time and runtime diagnostics

**Basic PL/SQL Block Structure**

DECLARE

-- Variable declarations

BEGIN

-- Executable statements

EXCEPTION

-- Error handling

END;

**3. Example: Find Maximum of Two Numbers**

DECLARE

a1 NUMBER := 10;

b NUMBER := 100;

c1 NUMBER;

BEGIN

IF a1 > b THEN

c1 := a1;

ELSE

c1 := b;

END IF;

DBMS\_OUTPUT.PUT\_LINE('Maximum number in 10 and 100: ' || c1);

END;

**4. Data Types and Constants**

**Declaring a Constant:**

DECLARE

school\_name CONSTANT VARCHAR2(20) := 'DPS';

BEGIN

DBMS\_OUTPUT.PUT\_LINE('I study in ' || school\_name);

END;

**5. Even/Odd Check**

DECLARE

a1 NUMBER := :a1;

BEGIN

IF MOD(a1, 2) = 0 THEN

DBMS\_OUTPUT.PUT\_LINE(a1 || ' is Even');

ELSE

DBMS\_OUTPUT.PUT\_LINE(a1 || ' is Odd');

END IF;

END;

**6. Grade Evaluation Using CASE**

DECLARE

c\_grade CHAR(1) := :c\_grade;

c\_rank VARCHAR2(20);

BEGIN

CASE c\_grade

WHEN 'A' THEN c\_rank := 'Excellent';

WHEN 'B' THEN c\_rank := 'Very Good';

WHEN 'C' THEN c\_rank := 'Good';

WHEN 'D' THEN c\_rank := 'Fair';

WHEN 'F' THEN c\_rank := 'Poor';

ELSE c\_rank := 'Invalid Grade';

END CASE;

DBMS\_OUTPUT.PUT\_LINE('Grade: ' || c\_grade);

DBMS\_OUTPUT.PUT\_LINE('Rank: ' || c\_rank);

END;

**7. Multiplication Table Using LOOP**

DECLARE

i INT := :i;

j INT := 1;

BEGIN

LOOP

EXIT WHEN j > 10;

DBMS\_OUTPUT.PUT\_LINE(i || '\*' || j || '=' || (i \* j));

j := j + 1;

END LOOP;

END;

**8. Functions in PL/SQL**

**Example: Odd or Even Function**

DECLARE

a1 NUMBER := 50;

c1 NUMBER;

FUNCTION oddOrEven(x IN NUMBER) RETURN NUMBER IS

BEGIN

IF MOD(x, 2) = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('EVEN NUMBER');

ELSE

DBMS\_OUTPUT.PUT\_LINE('ODD NUMBER');

END IF;

RETURN -1;

END;

BEGIN

c1 := oddOrEven(a1);

END;

**9. Cursors**

**Implicit Cursor**

Used automatically for single-row INSERT, UPDATE, DELETE, or SELECT.

**Explicit Cursor Example:**

DECLARE

CURSOR c\_customer IS SELECT name FROM customer;

TYPE c\_list IS TABLE OF customer.name%TYPE;

name\_list c\_list := c\_list();

counter INTEGER := 0;

BEGIN

FOR n IN c\_customer LOOP

counter := counter + 1;

name\_list.EXTEND;

name\_list(counter) := n.name;

DBMS\_OUTPUT.PUT\_LINE('Customer: ' || n.name);

END LOOP;

END;

**10. Triggers and Packages**

You mentioned **Triggers** and **Packages**, which are advanced PL/SQL features:

* **Triggers**: Automatically execute code in response to events (e.g., BEFORE INSERT)
* **Packages**: Group related procedures, functions, variables, and cursors into a single unit

If you'd like, I can help you write a sample trigger or package next. Want to explore that?