**DBMS Practical No: 6**

**# Write a PL/SQL code block using Cursors (All types:**

**Implicit, Explicit, Cursor FOR Loop, Parameterized Cursor)**

**1. Oracle Database:**

* This is a relational database management system developed by Oracle Corporation. It's one of the most widely used database systems for storing and managing data.

**2. PL/SQL:**

* PL/SQL stands for "Procedural Language/Structured Query Language." It's a programming language used to interact with an Oracle Database. PL/SQL allows you to create stored procedures, functions, and triggers for database management.

**3. Implicit Cursor:**

* Implicit cursors are automatically created by the Oracle Database to handle SQL statements within PL/SQL code. They are used for processing query results without the need to explicitly declare, open, fetch, or close them.

**4. Explicit Cursor:**

* Explicit cursors are user-defined cursors in PL/SQL. Unlike implicit cursors, you need to explicitly declare, open, fetch, and close them. This provides more control over the cursor's behaviour.

**5. Cursor FOR Loop:**

* A Cursor FOR Loop is a simplified way to work with cursors. It allows you to process query results without explicitly opening, fetching, and closing the cursor. The loop handles these operations automatically.

**6. Parameterized Cursor:**

* A parameterized cursor is a cursor that can accept input parameters. These parameters are used to filter query results based on specific conditions. You can pass values to the cursor when you open it.

**7. DECLARE:**

* In PL/SQL, the DECLARE keyword is used to declare variables and cursors. This section is where you declare all the variables and cursors that will be used in your PL/SQL block.

**8. BEGIN:**

* The BEGIN keyword marks the beginning of the executable part of a PL/SQL block. This is where the actual code and logic are written.

**9. OPEN:**

* The OPEN statement is used to open an explicit cursor. It prepares the cursor to fetch rows from the result set.

**10. FETCH: -** The FETCH statement retrieves rows from the cursor and stores them in variables. This is where you extract data from the result set row by row.

**11. EXIT WHEN: -** The EXIT WHEN statement is used to exit a loop. It specifies a condition under which the loop should terminate.

**12. CLOSE: -** The CLOSE statement is used to close an explicit cursor once you've finished fetching data. It releases resources associated with the cursor.

**13. DBMS\_OUTPUT.PUT\_LINE: -** DBMS\_OUTPUT.PUT\_LINE is a PL/SQL procedure that is used to display messages or output to the screen. It's often used for debugging and displaying results within PL/SQL blocks.

**Step 1: Set Up Your Environment**

Make sure you have access to an Oracle Database or any other database system that supports PL/SQL. For this example, we'll use Oracle Database.

**Step 2: Create a Sample Table and Insert Data**

To demonstrate cursors, you'll need a sample table with data. Let's create a simple table and insert some data:

**sql**

**CREATE TABLE employees (**

**employee\_id NUMBER,**

**first\_name VARCHAR2(50),**

**last\_name VARCHAR2(50),**

**salary NUMBER**

**);**

**INSERT INTO employees VALUES (1, 'John', 'Doe', 50000);**

**INSERT INTO employees VALUES (2, 'Jane', 'Smith', 60000);**

**INSERT INTO employees VALUES (3, 'Bob', 'Johnson', 55000);**

**COMMIT;**

**Step 3: Implicit Cursor**

Implicit cursors are automatically created by Oracle to process SQL statements. The simplest implicit cursor is used to loop through query results.

**sql**

**-- Implicit cursor**

**DECLARE**

**v\_employee\_name employees.first\_name%TYPE;**

**v\_employee\_salary employees.salary%TYPE;**

**BEGIN**

**-- Open the cursor**

**FOR emp\_record IN (SELECT first\_name, salary FROM employees) LOOP**

**v\_employee\_name := emp\_record.first\_name;**

**v\_employee\_salary := emp\_record.salary;**

**-- Process the data (e.g., print it)**

**DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_employee\_name || ', Salary: ' || v\_employee\_salary);**

**END LOOP;**

**END;**

**/**

**Step 4: Explicit Cursor**

Explicit cursors are user-defined cursors that provide more control. They need to be declared, opened, fetched, and closed manually.

**sql**

**-- Explicit cursor**

**DECLARE**

**CURSOR employee\_cursor IS**

**SELECT first\_name, salary FROM employees;**

**v\_employee\_name employees.first\_name%TYPE;**

**v\_employee\_salary employees.salary%TYPE;**

**BEGIN**

**-- Open the cursor**

**OPEN employee\_cursor;**

**LOOP**

**-- Fetch a row from the cursor**

**FETCH employee\_cursor INTO v\_employee\_name, v\_employee\_salary;**

**EXIT WHEN employee\_cursor%NOTFOUND;**

**-- Process the data (e.g., print it)**

**DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_employee\_name || ', Salary: ' || v\_employee\_salary);**

**END LOOP;**

**-- Close the cursor**

**CLOSE employee\_cursor;**

**END;**

**/**

**Step 5: Cursor FOR Loop**

A Cursor FOR Loop is a simplified way to work with cursors without explicitly opening, fetching, and closing them.

**sql**

**-- Cursor FOR Loop**

**DECLARE**

**v\_employee\_name employees.first\_name%TYPE;**

**v\_employee\_salary employees.salary%TYPE;**

**BEGIN**

**FOR emp\_record IN (SELECT first\_name, salary FROM employees) LOOP**

**v\_employee\_name := emp\_record.first\_name;**

**v\_employee\_salary := emp\_record.salary;**

**-- Process the data (e.g., print it)**

**DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_employee\_name || ', Salary: ' || v\_employee\_salary);**

**END LOOP;**

**END;**

**/**

**Step 6: Parameterized Cursor**

Parameterized cursors allow you to pass parameters to your SQL query.

**sql**

**-- Parameterized cursor**

**DECLARE**

**CURSOR employee\_cursor (p\_min\_salary NUMBER) IS**

**SELECT first\_name, salary**

**FROM employees**

**WHERE salary >= p\_min\_salary;**

**v\_employee\_name employees.first\_name%TYPE;**

**v\_employee\_salary employees.salary%TYPE;**

**BEGIN**

**OPEN employee\_cursor(55000); -- Provide a minimum salary parameter**

**LOOP**

**FETCH employee\_cursor INTO v\_employee\_name, v\_employee\_salary;**

**EXIT WHEN employee\_cursor%NOTFOUND;**

**DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_employee\_name || ', Salary: ' || v\_employee\_salary);**

**END LOOP;**

**CLOSE employee\_cursor;**

**END;**

**/**