**9. Write a PL/SQL block to implement all types of cursor.**

**Ans :**

**-- Step 1: Create the database (if not already created)**

CREATE DATABASE IndianCollegeDB;

**-- Step 2: Use the created database**

USE IndianCollegeDB;

**-- Step 3: Create the Instructor table with Primary Key**

CREATE TABLE Instructor (

InstructorID INT PRIMARY KEY,

Name VARCHAR(100),

Department VARCHAR(100)

);

**-- Step 4: Create the Student table with Primary Key**

CREATE TABLE Student (

StudentID INT PRIMARY KEY AUTO\_INCREMENT,

Name VARCHAR(100),

Email VARCHAR(100)

);

**-- Step 5: Create the Course table with Foreign Key reference to Instructor table**

CREATE TABLE Course (

CourseID INT PRIMARY KEY,

Title VARCHAR(100),

InstructorID INT,

FOREIGN KEY (InstructorID) REFERENCES Instructor(InstructorID)

);

**-- Step 6: Create the Enrollment table (junction table) with Foreign Keys**

CREATE TABLE Enrollment (

EnrollmentID INT PRIMARY KEY AUTO\_INCREMENT,

StudentID INT,

CourseID INT,

EnrollmentDate DATE,

FOREIGN KEY (StudentID) REFERENCES Student(StudentID),

FOREIGN KEY (CourseID) REFERENCES Course(CourseID)

);

**-- Step 7: Insert sample data into Instructor table**

INSERT INTO Instructor (InstructorID, Name, Department) VALUES

(1, 'Dr. Rajesh Kumar', 'Computer Science'),

(2, 'Prof. Meena Agarwal', 'Mathematics'),

(3, 'Dr. Arvind Sharma', 'Physics');

**-- Step 8: Insert sample data into Student table**

INSERT INTO Student (Name, Email) VALUES

('Amit Patel', 'amit.patel@example.com'),

('Priya Sharma', 'priya.sharma@example.com'),

('Ravi Kumar', 'ravi.kumar@example.com'),

('Neha Singh', 'neha.singh@example.com'),

('Vikram Joshi', 'vikram.joshi@example.com');

**-- Step 9: Insert sample data into Course table**

INSERT INTO Course (CourseID, Title, InstructorID) VALUES

(1, 'Introduction to Programming', 1),

(2, 'Data Structures', 1),

(3, 'Calculus I', 2),

(4, 'Quantum Mechanics', 3),

(5, 'Linear Algebra', 2);

**-- Step 10: Insert sample data into Enrollment table**

INSERT INTO Enrollment (StudentID, CourseID, EnrollmentDate) VALUES

(1, 1, '2025-04-01'),

(2, 2, '2025-04-02'),

(3, 3, '2025-04-03'),

(4, 4, '2025-04-04'),

(5, 5, '2025-04-05');

**-- Step 11: PL/SQL block demonstrating different types of cursors**

DECLARE

**-- 1. Implicit Cursor example: Using SELECT INTO**

v\_StudentName VARCHAR2(100);

v\_StudentEmail VARCHAR2(100);

**-- 2. Explicit Cursor example: Declaring an explicit cursor**

CURSOR c\_enrollments IS

SELECT e.EnrollmentID, s.Name AS StudentName, c.Title AS CourseTitle

FROM Enrollment e

JOIN Student s ON e.StudentID = s.StudentID

JOIN Course c ON e.CourseID = c.CourseID;

**-- 3. Cursor FOR Loop example: Using a FOR loop for an implicit cursor**

v\_EnrollmentCount INT := 0;

**-- 4. Parameterized Cursor example: A cursor that accepts a parameter**

CURSOR c\_student\_enrollment(p\_StudentID INT) IS

SELECT c.Title

FROM Enrollment e

JOIN Course c ON e.CourseID = c.CourseID

WHERE e.StudentID = p\_StudentID;

BEGIN

**-- 1. Implicit Cursor: Using SELECT INTO to fetch student info**

SELECT Name, Email

INTO v\_StudentName, v\_StudentEmail

FROM Student

WHERE StudentID = 1;

**-- Display the student information**

DBMS\_OUTPUT.PUT\_LINE('Student Name: ' || v\_StudentName);

DBMS\_OUTPUT.PUT\_LINE('Student Email: ' || v\_StudentEmail);

**-- 2. Explicit Cursor: Opening and fetching data from explicit cursor**

OPEN c\_enrollments;

LOOP

FETCH c\_enrollments INTO v\_StudentName, v\_EnrollmentCount;

EXIT WHEN c\_enrollments%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Enrollment ID: ' || v\_EnrollmentCount || ', Student: ' || v\_StudentName);

END LOOP;

CLOSE c\_enrollments;

**-- 3. Cursor FOR Loop: A more concise way to handle cursors**

FOR rec IN (SELECT e.EnrollmentID, s.Name AS StudentName, c.Title AS CourseTitle

FROM Enrollment e

JOIN Student s ON e.StudentID = s.StudentID

JOIN Course c ON e.CourseID = c.CourseID) LOOP

DBMS\_OUTPUT.PUT\_LINE('Enrollment ID: ' || rec.EnrollmentID ||

', Student: ' || rec.StudentName ||

', Course: ' || rec.CourseTitle);

END LOOP;

**-- 4. Parameterized Cursor: Fetching courses for a specific student**

OPEN c\_student\_enrollment(2); -- Passing StudentID 2 as a parameter

FOR rec IN c\_student\_enrollment LOOP

DBMS\_OUTPUT.PUT\_LINE('Student 2 is enrolled in: ' || rec.Title);

END LOOP;

CLOSE c\_student\_enrollment;

END;

/

**-- Step 12: Show the output using SELECT statements**

-- Show all enrollments

SELECT \* FROM Enrollment;

-- Show all students

SELECT \* FROM Student;

**Explanation of the Code:**

1. **Implicit Cursor**:
   * **What it does**: This is automatically created by Oracle when performing a SELECT INTO statement. It implicitly handles the result of the query.
   * **Example**: The block first uses an implicit cursor to select the Name and Email of the student with StudentID = 1 and stores these values into variables v\_StudentName and v\_StudentEmail.
2. **Explicit Cursor**:
   * **What it does**: An explicit cursor is declared by the user, and the programmer manually opens, fetches, and closes it.
   * **Example**: The c\_enrollments cursor is explicitly declared to fetch student enrollment details. The cursor is opened and then fetched inside a loop, displaying enrollment details.
3. **Cursor FOR Loop**:
   * **What it does**: This is a simplified version of an explicit cursor. It automatically handles the opening, fetching, and closing of the cursor for you.
   * **Example**: The FOR rec IN (...) loop processes the SELECT statement, which retrieves the student enrollments, and automatically closes the cursor once the loop completes.
4. **Parameterized Cursor**:
   * **What it does**: A cursor that takes a parameter as input, which is used to filter the results dynamically.
   * **Example**: The c\_student\_enrollment cursor is parameterized with p\_StudentID, and the cursor is opened with the parameter value 2 to fetch courses for student 2. It processes the result set in a FOR loop and displays the courses.