**Table Creation:**

CREATE TABLE Customers (

    CustomerID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    DOB DATE,

    Balance NUMBER,

    LastModified DATE,

    IsVIP VARCHAR2(5) DEFAULT 'FALSE'

);

CREATE TABLE Loans (

    LoanID NUMBER PRIMARY KEY,

    CustomerID NUMBER,

    LoanAmount NUMBER,

    InterestRate NUMBER,

    StartDate DATE,

    EndDate DATE,

    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Customers (

    CustomerID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    DOB DATE,

    Balance NUMBER,

    LastModified DATE

);

CREATE TABLE Accounts (

    AccountID NUMBER PRIMARY KEY,

    CustomerID NUMBER,

    AccountType VARCHAR2(20),

    Balance NUMBER,

    LastModified DATE,

    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Employees (

    EmployeeID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    Position VARCHAR2(50),

    Salary NUMBER,

    Department VARCHAR2(50),

    HireDate DATE

);

**Insert Statements**

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1950-05-15', 'YYYY-MM-DD'), 12000, SYSDATE, 'FALSE');

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 8000, SYSDATE, 'FALSE');

INSERT INTO Customers VALUES (1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);

INSERT INTO Customers VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);

INSERT INTO Accounts VALUES (1, 1, 'Savings', 1000, SYSDATE);

INSERT INTO Accounts VALUES (2, 2, 'Checking', 1500, SYSDATE);

INSERT INTO Employees VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', TO\_DATE('2015-06-15', 'YYYY-MM-DD'));

INSERT INTO Employees VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', TO\_DATE('2017-03-20', 'YYYY-MM-DD'));

INSERT INTO Loans VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 15));

COMMIT;

**Exercise 1: Control Structures**

**Scenario 1:**

**Question:** Write a PL/SQL block that loops through all customers, checks their age, and if they are above 60, apply a 1% discount to their current loan interest rates.

BEGIN

  FOR rec IN (

    SELECT CustomerID

    FROM Customers

    WHERE MONTHS\_BETWEEN(SYSDATE, DOB)/12 > 60

  ) LOOP

    UPDATE Loans

    SET InterestRate = InterestRate - 1

    WHERE CustomerID = rec.CustomerID;

  END LOOP;

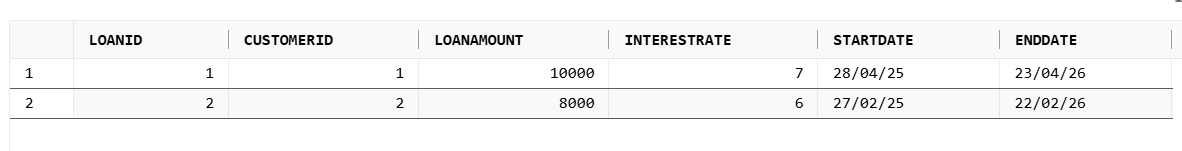
  COMMIT;

END;

/

SELECT \* FROM Loans;

**Output**



**Scenario 2:**

**Question:** Write a PL/SQL block that iterates through all customers and sets a flag IsVIP to TRUE for those with a balance over $10,000.

ALTER TABLE Customers ADD IsVIP VARCHAR2(5) DEFAULT 'FALSE';

BEGIN

  FOR rec IN (

    SELECT CustomerID

    FROM Customers

    WHERE Balance > 10000

  ) LOOP

    UPDATE Customers

    SET IsVIP = 'TRUE'

    WHERE CustomerID = rec.CustomerID;

  END LOOP;

  COMMIT;

END;

/

SELECT CustomerID, Name, Balance, IsVIP FROM Customers;

**Output**



**Scenario 3:**

**Question:** Write a PL/SQL block that fetches all loans due in the next 30 days and prints a reminder message for each customer.

SET SERVEROUTPUT ON;

DECLARE

  v\_name Customers.Name%TYPE;

BEGIN

  FOR rec IN (

    SELECT LoanID, CustomerID, EndDate

    FROM Loans

    WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30

  ) LOOP

    BEGIN

      SELECT Name INTO v\_name

      FROM Customers

      WHERE CustomerID = rec.CustomerID;

      DBMS\_OUTPUT.PUT\_LINE(

        'Reminder: ' || v\_name || ', your loan (ID ' || rec.LoanID ||

        ') is due on ' || TO\_CHAR(rec.EndDate, 'DD-Mon-YYYY')

      );

    EXCEPTION

      WHEN NO\_DATA\_FOUND THEN

        DBMS\_OUTPUT.PUT\_LINE('No customer found for Loan ID: ' || rec.LoanID);

    END;

  END LOOP;

END;

/

**Output**

