**Exercise 1: Control Structures**

**Creating tables:**

SET SERVEROUTPUT ON;

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Transactions CASCADE CONSTRAINTS';

EXECUTE IMMEDIATE 'DROP TABLE Accounts CASCADE CONSTRAINTS';

EXECUTE IMMEDIATE 'DROP TABLE Loans CASCADE CONSTRAINTS';

EXECUTE IMMEDIATE 'DROP TABLE Customers CASCADE CONSTRAINTS';

EXECUTE IMMEDIATE 'DROP TABLE Employees CASCADE CONSTRAINTS';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

-- Create tables

CREATE TABLE Customers (

CustomerID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

DOB DATE,

Balance NUMBER,

LastModified DATE,

IsVIP VARCHAR2(5)

);

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20),

Balance NUMBER,

LastModified DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Transactions (

TransactionID NUMBER PRIMARY KEY,

AccountID NUMBER,

TransactionDate DATE,

Amount NUMBER,

TransactionType VARCHAR2(10),

FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID),

CONSTRAINT chk\_transaction\_type CHECK (TransactionType IN ('Deposit', 'Withdrawal'))

);

CREATE TABLE Loans (

LoanID NUMBER PRIMARY KEY,

CustomerID NUMBER,

LoanAmount NUMBER CHECK (LoanAmount > 0),

InterestRate NUMBER,

StartDate DATE,

EndDate 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 data

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

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

INSERT INTO Customers VALUES (3, 'David Lee', TO\_DATE('1962-11-30', 'YYYY-MM-DD'), 15000, SYSDATE, NULL);

INSERT INTO Accounts VALUES (1, 1, 'Savings', 12000, SYSDATE);

INSERT INTO Accounts VALUES (2, 2, 'Checking', 8000, SYSDATE);

INSERT INTO Accounts VALUES (3, 3, 'Savings', 15000, SYSDATE);

INSERT INTO Transactions VALUES (1, 1, SYSDATE, 500, 'Deposit');

INSERT INTO Transactions VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

INSERT INTO Transactions VALUES (3, 3, SYSDATE, 700, 'Deposit');

INSERT INTO Loans VALUES (1, 1, 5000, 7, SYSDATE, SYSDATE + 10);

INSERT INTO Loans VALUES (2, 2, 6000, 6, SYSDATE, SYSDATE + 40);

INSERT INTO Loans VALUES (3, 3, 7000, 5, SYSDATE, SYSDATE + 20);

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'));

**Scenario 1:** The bank wants to apply a discount to loan interest rates for customers above 60 years old.

* **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.

**Scenario 1 code:**

-- Scenario 1: Discount interest for customers > 60 years

BEGIN

FOR cust IN (

SELECT CustomerID, FLOOR(MONTHS\_BETWEEN(SYSDATE, DOB)/12) AS Age

FROM Customers

) LOOP

IF cust.Age > 60 THEN

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE CustomerID = cust.CustomerID;

END IF;

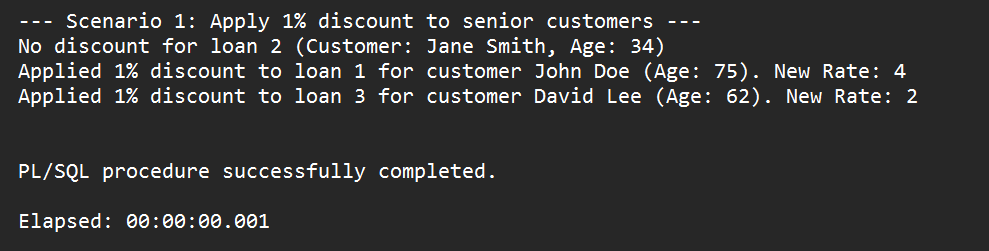
END LOOP;

COMMIT;

END;

/

**Scenario 1 output:**

****

**Scenario 2:** A customer can be promoted to VIP status based on their balance.

* **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.

**Scenario 2 code:**

-- Scenario 2: Set IsVIP = TRUE if balance > 10000

BEGIN

FOR cust IN (SELECT CustomerID, Balance FROM Customers) LOOP

IF cust.Balance > 10000 THEN

UPDATE Customers SET IsVIP = 'TRUE' WHERE CustomerID = cust.CustomerID;

ELSE

UPDATE Customers SET IsVIP = 'FALSE' WHERE CustomerID = cust.CustomerID;

END IF;

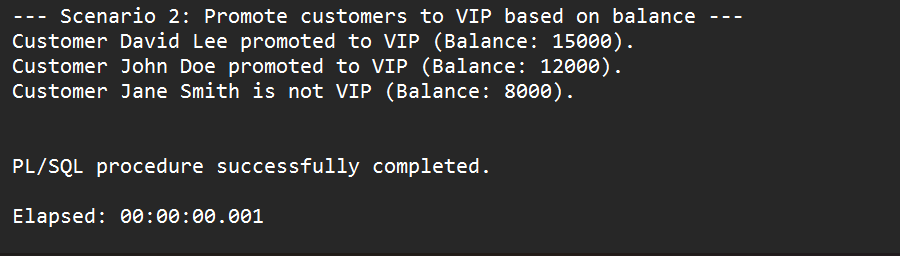
END LOOP;

COMMIT;

END;

/

**Scenario 2 output:**

****

**Scenario 3:** The bank wants to send reminders to customers whose loans are due within the next 30 days.

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

**Scenario 3 code:**

-- Scenario 3: Print reminders for loans due in next 30 days

BEGIN

FOR loan\_rec IN (

SELECT LoanID, CustomerID, EndDate

FROM Loans

WHERE EndDate BETWEEN SYSDATE AND SYSDATE + 30

) LOOP

DECLARE

v\_name VARCHAR2(100);

BEGIN

SELECT Name INTO v\_name FROM Customers WHERE CustomerID = loan\_rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ID ' || loan\_rec.LoanID ||

' for customer ' || v\_name ||

' is due on ' || TO\_CHAR(loan\_rec.EndDate, 'YYYY-MM-DD'));

END;

END LOOP;

END;

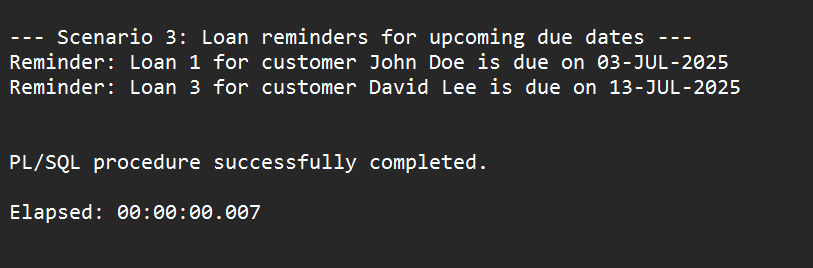
/

-- Check updates

SELECT \* FROM Loans;

SELECT \* FROM Customers;

**Scenario 3 output:**

****