**Exercise 1: control Structures**

PL/SQL *Programming*

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

BEGIN

FOR cust\_rec IN (SELECT CustomerID, Age, LoanInterestRate FROM Customers WHERE Age > 60) LOOP

UPDATE Customers

SET LoanInterestRate = LoanInterestRate - 1

WHERE CustomerID = cust\_rec.CustomerID;

END LOOP;

COMMIT;

END;

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

BEGIN

FOR cust\_rec IN (SELECT CustomerID, Balance FROM Customers WHERE Balance > 10000) LOOP

UPDATE Customers

SET IsVIP = 'TRUE'

WHERE CustomerID = cust\_rec.CustomerID;

END LOOP;

COMMIT;

END;

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

DECLARE

v\_due\_date DATE := SYSDATE + 30;

BEGIN

FOR loan\_rec IN (

SELECT LoanID, CustomerID, DueDate

FROM Loans

WHERE DueDate BETWEEN SYSDATE AND v\_due\_date

) LOOP

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

' for Customer ' || loan\_rec.CustomerID ||

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

END LOOP;

END;

**Exercise 3: Stored Procedures**

**Scenario 1: The bank needs to process monthly interest for all savings accounts.**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountType = 'SAVINGS';

COMMIT;

END;

**Scenario 2: The bank wants to implement a bonus scheme for employees based on their performance.**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_DepartmentID IN Employees.DepartmentID%TYPE,

p\_BonusPercent IN NUMBER

) IS

BEGIN

UPDATE Employees

SET Salary = Salary + (Salary \* (p\_BonusPercent / 100))

WHERE DepartmentID = p\_DepartmentID;

COMMIT;

END;

**Scenario 3: Customers should be able to transfer funds between their accounts.** CREATE OR REPLACE PROCEDURE TransferFunds(

p\_FromAccountID IN Accounts.AccountID%TYPE,

p\_ToAccountID IN Accounts.AccountID%TYPE,

p\_Amount IN NUMBER

) IS

v\_FromBalance Accounts.Balance%TYPE;

BEGIN

SELECT Balance INTO v\_FromBalance

FROM Accounts

WHERE AccountID = p\_FromAccountID

FOR UPDATE;

IF v\_FromBalance < p\_Amount THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Insufficient balance in source account.');

END IF;

UPDATE Accounts

SET Balance = Balance - p\_Amount

WHERE AccountID = p\_FromAccountID;

UPDATE Accounts

SET Balance = Balance + p\_Amount

WHERE AccountID = p\_ToAccountID;

COMMIT;

END;