**PL SQL**

**Exercise -1: Control Structures**

**Scenario 1:**

DECLARE

CURSOR c\_customers IS

SELECT CustomerID, LoanID, InterestRate, Age

FROM Customers

JOIN Loans ON Customers.CustomerID = Loans.CustomerID;

BEGIN

FOR rec IN c\_customers LOOP

IF rec.Age > 60 THEN

UPDATE Loans

SET InterestRate = InterestRate - 1

WHERE LoanID = rec.LoanID;

END IF;

END LOOP;

COMMIT;

END;

/

**Scenario 2:**

DECLARE

CURSOR c\_loans IS

SELECT LoanID, CustomerID, DueDate

FROM Loans

WHERE DueDate BETWEEN SYSDATE AND SYSDATE + 30;

v\_name VARCHAR2(100);

BEGIN

FOR rec IN c\_loans LOOP

-- Fetch customer name (optional)

SELECT Name INTO v\_name

FROM Customers

WHERE CustomerID = rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || v\_name ||

', your loan (ID: ' || rec.LoanID ||

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

END LOOP;

END;

/

**Scenario 3:**

DECLARE

CURSOR c\_loans IS

SELECT LoanID, CustomerID, DueDate

FROM Loans

WHERE DueDate BETWEEN SYSDATE AND SYSDATE + 30;

v\_name VARCHAR2(100);

BEGIN

FOR rec IN c\_loans LOOP

-- Fetch customer name (optional)

SELECT Name INTO v\_name

FROM Customers

WHERE CustomerID = rec.CustomerID;

DBMS\_OUTPUT.PUT\_LINE('Reminder: Dear ' || v\_name ||

', your loan (ID: ' || rec.LoanID ||

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

END LOOP;

END;

/

**Exercise – 2: Stored Procedures**

**Scenario 1:**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountType = 'SAVINGS';

COMMIT;

END;

/

**Scenario 2:**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_DepartmentID IN NUMBER,

p\_BonusPercent IN NUMBER

) IS

BEGIN

UPDATE Employees

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

WHERE DepartmentID = p\_DepartmentID;

COMMIT;

END;

/

**Scenario 3:**

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_FromAccountID IN NUMBER,

p\_ToAccountID IN NUMBER,

p\_Amount IN NUMBER

) IS

v\_FromBalance NUMBER;

BEGIN

-- Get source account balance

SELECT Balance INTO v\_FromBalance

FROM Accounts

WHERE AccountID = p\_FromAccountID

FOR UPDATE;

-- Check if sufficient funds

IF v\_FromBalance < p\_Amount THEN

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

END IF;

-- Deduct from source

UPDATE Accounts

SET Balance = Balance - p\_Amount

WHERE AccountID = p\_FromAccountID;

-- Add to destination

UPDATE Accounts

SET Balance = Balance + p\_Amount

WHERE AccountID = p\_ToAccountID;

COMMIT;

END;

/