PLSQL\_Exercises : Exercise 1: Control Structures

CREATE TABLE Customers (

CustomerID NUMBER,

Name VARCHAR2(100),

Age NUMBER,

Balance NUMBER,

IsVIP CHAR(1)

);

CREATE TABLE Loans (

LoanID NUMBER,

CustomerID NUMBER,

DueDate DATE,

InterestRate NUMBER

);

Scenario 1: Apply 1% discount for customers over 60

BEGIN

FOR rec IN (SELECT CustomerID, InterestRate FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

WHERE c.Age > 60) LOOP

UPDATE Loans

SET InterestRate = InterestRate - (InterestRate \* 0.01)

WHERE CustomerID = rec.CustomerID;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Discount applied to customers above 60.');

END;

Scenario 2: Promote customers to VIP if balance > $10,000

BEGIN

FOR rec IN (SELECT CustomerID FROM Customers WHERE Balance > 10000) LOOP

UPDATE Customers

SET IsVIP = 'Y'

WHERE CustomerID = rec.CustomerID;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('VIP status updated.');

END;

Scenario 3: Reminders for loans due in next 30 days

BEGIN

FOR rec IN (SELECT l.LoanID, c.Name, l.DueDate

FROM Loans l

JOIN Customers c ON l.CustomerID = c.CustomerID

WHERE l.DueDate BETWEEN SYSDATE AND SYSDATE + 30) LOOP

DBMS\_OUTPUT.PUT\_LINE('Reminder: Loan ID ' || rec.LoanID || ' for ' || rec.Name ||

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

END LOOP;

END;

PLSQL\_Exercises : Exercise 3: Stored Procedures

Scenario 1: ProcessMonthlyInterest

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Customers

SET Balance = Balance + (Balance \* 0.01);

DBMS\_OUTPUT.PUT\_LINE('Monthly interest applied to all savings accounts.');

END;

Scenario 2: UpdateEmployeeBonus

CREATE TABLE Employees (

EmpID NUMBER,

Name VARCHAR2(100),

DepartmentID NUMBER,

Salary NUMBER

);

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

dept\_id IN NUMBER,

bonus\_percent IN NUMBER

) IS

BEGIN

UPDATE Employees

SET Salary = Salary + (Salary \* bonus\_percent / 100)

WHERE DepartmentID = dept\_id;

DBMS\_OUTPUT.PUT\_LINE('Bonuses applied to Department ID: ' || dept\_id);

END;

Scenario 3: TransferFunds

CREATE TABLE Accounts (

AccountID NUMBER,

CustomerID NUMBER,

Balance NUMBER

);

CREATE OR REPLACE PROCEDURE TransferFunds(

from\_acc IN NUMBER,

to\_acc IN NUMBER,

amount IN NUMBER

) IS

from\_balance NUMBER;

BEGIN

SELECT Balance INTO from\_balance FROM Accounts WHERE AccountID = from\_acc FOR UPDATE;

IF from\_balance < amount THEN

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

END IF;

-- Deduct from source

UPDATE Accounts SET Balance = Balance - amount WHERE AccountID = from\_acc;

-- Add to destination

UPDATE Accounts SET Balance = Balance + amount WHERE AccountID = to\_acc;

DBMS\_OUTPUT.PUT\_LINE('Transferred $' || amount || ' from Account ' || from\_acc || ' to Account ' || to\_acc);

END;