**Exercise 3: Stored Procedures**

**-- Customers table**

CREATE TABLE Customers (

CustomerID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

DOB DATE,

Balance NUMBER,

LastModified DATE

);

**-- Accounts table**

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20), -- 'Savings', 'Checking', etc.

Balance NUMBER,

LastModified DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

**-- Employees table**

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(100),

Position VARCHAR2(50),

Salary NUMBER,

Department VARCHAR2(50),

HireDate DATE

);

**Question:** Write a stored procedure **ProcessMonthlyInterest** that calculates and updates the balance of all savings accounts by applying an interest rate of 1% to the current balance.

**Solution :**

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR acc IN (

SELECT AccountID, Balance

FROM Accounts

WHERE AccountType = 'Savings'

) LOOP

UPDATE Accounts

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

LastModified = SYSDATE

WHERE AccountID = acc.AccountID;

END LOOP;

COMMIT;

END;

**Question:** Write a stored procedure **UpdateEmployeeBonus** that updates the salary of employees in a given department by adding a bonus percentage passed as a parameter.

**Solution :**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

deptName IN VARCHAR2,

bonusPercent IN NUMBER

) IS

BEGIN

FOR emp IN (

SELECT EmployeeID, Salary

FROM Employees

WHERE Department = deptName

) LOOP

UPDATE Employees

SET Salary = Salary + (Salary \* bonusPercent / 100)

WHERE EmployeeID = emp.EmployeeID;

END LOOP;

COMMIT;

END;

**Question:** Write a stored procedure **TransferFunds** that transfers a specified amount from one account to another, checking that the source account has sufficient balance before making the transfer.

**Solution :**

CREATE OR REPLACE PROCEDURE TransferFunds (

fromAccountID IN NUMBER,

toAccountID IN NUMBER,

amount IN NUMBER

) IS

fromBalance NUMBER;

BEGIN

-- Get source account balance

SELECT Balance INTO fromBalance

FROM Accounts

WHERE AccountID = fromAccountID

FOR UPDATE;

-- Check for sufficient balance

IF fromBalance < amount THEN

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

END IF;

-- Deduct from source account

UPDATE Accounts

SET Balance = Balance - amount,

LastModified = SYSDATE

WHERE AccountID = fromAccountID;

-- Add to destination account

UPDATE Accounts

SET Balance = Balance + amount,

LastModified = SYSDATE

WHERE AccountID = toAccountID;

COMMIT;

END;