**NAME: ABINAYA T S**

**SUPERSET\_ID: 6412256**

**WEEK 2: PL/SQL EXERCISE**

**Exercise 3: Stored Procedures**

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

* + **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.

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

* + **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.

**Scenario 3:** Customers should be able to transfer funds between their accounts.

**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.

SAMPLE CODE:

-- Enable DBMS output

SET SERVEROUTPUT ON;

-- =====================================

-- Drop Existing Tables if they exist

-- =====================================

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Accounts';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE Employees';

EXCEPTION

WHEN OTHERS THEN NULL;

END;

/

-- =====================================

-- Create Accounts Table

-- =====================================

CREATE TABLE Accounts (

AccountID NUMBER PRIMARY KEY,

AccountType VARCHAR2(20),

Balance NUMBER

);

-- Insert Sample Accounts

INSERT INTO Accounts VALUES (101, 'Savings', 10000);

INSERT INTO Accounts VALUES (102, 'Current', 15000);

INSERT INTO Accounts VALUES (103, 'Savings', 20000);

COMMIT;

-- =====================================

-- Create Employees Table

-- =====================================

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

Name VARCHAR2(50),

DepartmentID NUMBER,

Salary NUMBER

);

-- Insert Sample Employees

INSERT INTO Employees VALUES (1, 'Alice', 10, 50000);

INSERT INTO Employees VALUES (2, 'Bob', 10, 60000);

INSERT INTO Employees VALUES (3, 'Charlie', 20, 70000);

COMMIT;

-- =====================================

-- Procedure 1: ProcessMonthlyInterest

-- =====================================

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE Accounts

SET Balance = Balance + (Balance \* 0.01)

WHERE AccountType = 'Savings';

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

COMMIT;

END;

/

-- =====================================

-- Procedure 2: UpdateEmployeeBonus

-- =====================================

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;

DBMS\_OUTPUT.PUT\_LINE('Bonus updated for department ' || p\_DepartmentID);

COMMIT;

END;

/

-- =====================================

-- Procedure 3: TransferFunds

-- =====================================

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_FromAccountID IN NUMBER,

p\_ToAccountID IN NUMBER,

p\_Amount IN NUMBER

) IS

v\_FromBalance NUMBER;

BEGIN

-- Check source balance

SELECT Balance INTO v\_FromBalance

FROM Accounts

WHERE AccountID = p\_FromAccountID;

IF v\_FromBalance >= p\_Amount THEN

-- 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;

DBMS\_OUTPUT.PUT\_LINE('Transferred ' || p\_Amount || ' from Account ' || p\_FromAccountID || ' to Account ' || p\_ToAccountID);

COMMIT;

ELSE

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

END IF;

END;

/

-- =====================================

-- Execute Procedures

-- =====================================

-- Apply monthly interest

BEGIN

ProcessMonthlyInterest;

END;

/

-- Give 10% bonus to department 10

BEGIN

UpdateEmployeeBonus(10, 10);

END;

/

-- Transfer funds from Account 101 to 102

BEGIN

TransferFunds(101, 102, 1000);

END;

/

-- =====================================

-- View Final Results

-- =====================================

PROMPT === Final Account Balances ===

SELECT \* FROM Accounts;

PROMPT === Final Employee Salaries ===

SELECT \* FROM Employees;  
  
  
  
OUTPUT:  
  
