**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.

**Code:**

SET SERVEROUTPUT ON

BEGIN

    EXECUTE IMMEDIATE 'DROP TABLE Transactions';

EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN

    EXECUTE IMMEDIATE 'DROP TABLE Accounts';

EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN

    EXECUTE IMMEDIATE 'DROP TABLE Loans';

EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN

    EXECUTE IMMEDIATE 'DROP TABLE Employees';

EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

BEGIN

    EXECUTE IMMEDIATE 'DROP TABLE Customers';

EXCEPTION WHEN OTHERS THEN IF SQLCODE != -942 THEN RAISE; END IF; END;

/

-- Create tables

CREATE TABLE Customers (

    CustomerID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    DOB DATE,

    Balance NUMBER,

    LastModified DATE

);

CREATE TABLE Accounts (

    AccountID NUMBER PRIMARY KEY,

    CustomerID NUMBER,

    AccountType VARCHAR2(20),

    Balance NUMBER,

    LastModified DATE,

    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Transactions (

    TransactionID NUMBER PRIMARY KEY,

    AccountID NUMBER,

    TransactionDate DATE,

    Amount NUMBER,

    TransactionType VARCHAR2(10),

    FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)

);

CREATE TABLE Loans (

    LoanID NUMBER PRIMARY KEY,

    CustomerID NUMBER,

    LoanAmount NUMBER,

    InterestRate NUMBER,

    StartDate DATE,

    EndDate DATE,

    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE Employees (

    EmployeeID NUMBER PRIMARY KEY,

    Name VARCHAR2(100),

    Position VARCHAR2(50),

    Salary NUMBER,

    Department VARCHAR2(50),

    HireDate DATE

);

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

VALUES (1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000, SYSDATE);

INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)

VALUES (2, 'Jane Smith', TO\_DATE('1990-07-20', 'YYYY-MM-DD'), 1500, SYSDATE);

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)

VALUES (1, 1, 'Savings', 1000, SYSDATE);

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)

VALUES (2, 2, 'Checking', 1500, SYSDATE);

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (1, 1, SYSDATE, 200, 'Deposit');

INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

VALUES (2, 2, SYSDATE, 300, 'Withdrawal');

INSERT INTO Loans (LoanID, CustomerID, LoanAmount, InterestRate, StartDate, EndDate)

VALUES (1, 1, 5000, 5, SYSDATE, ADD\_MONTHS(SYSDATE, 60));

INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)

VALUES (1, 'Alice Johnson', 'Manager', 70000, 'HR', TO\_DATE('2015-06-15', 'YYYY-MM-DD'));

INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)

VALUES (2, 'Bob Brown', 'Developer', 60000, 'IT', TO\_DATE('2017-03-20', 'YYYY-MM-DD'));

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest AS

BEGIN

    UPDATE Accounts

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

        LastModified = SYSDATE

    WHERE AccountType = 'Savings';

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

END;

/

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

    p\_Department IN VARCHAR2,

    p\_BonusPercent IN NUMBER

) AS

BEGIN

    UPDATE Employees

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

    WHERE Department = p\_Department;

    DBMS\_OUTPUT.PUT\_LINE('Bonus applied to department: ' || p\_Department);

END;

/

CREATE OR REPLACE PROCEDURE TransferFunds (

    p\_SourceAccountID IN NUMBER,

    p\_TargetAccountID IN NUMBER,

    p\_Amount IN NUMBER

) AS

    v\_SourceBalance NUMBER;

BEGIN

    -- Lock source account row

    SELECT Balance INTO v\_SourceBalance

    FROM Accounts

    WHERE AccountID = p\_SourceAccountID

    FOR UPDATE;

    -- Check balance

    IF v\_SourceBalance < p\_Amount THEN

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

    END IF;

    -- Deduct from source

    UPDATE Accounts

    SET Balance = Balance - p\_Amount,

        LastModified = SYSDATE

    WHERE AccountID = p\_SourceAccountID;

    -- Add to target

    UPDATE Accounts

    SET Balance = Balance + p\_Amount,

        LastModified = SYSDATE

    WHERE AccountID = p\_TargetAccountID;

    -- Record transactions

    INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

    VALUES (Transactions\_seq.NEXTVAL, p\_SourceAccountID, SYSDATE, -p\_Amount, 'Transfer');

    INSERT INTO Transactions (TransactionID, AccountID, TransactionDate, Amount, TransactionType)

    VALUES (Transactions\_seq.NEXTVAL, p\_TargetAccountID, SYSDATE, p\_Amount, 'Transfer');

    DBMS\_OUTPUT.PUT\_LINE('Transferred ' || p\_Amount || ' from account ' || p\_SourceAccountID || ' to ' || p\_TargetAccountID);

END;

/

CREATE SEQUENCE Transactions\_seq START WITH 3 INCREMENT BY 1;

-- Apply monthly interest

BEGIN

    ProcessMonthlyInterest;

END;

/

-- Apply 10% bonus to IT department

BEGIN

    UpdateEmployeeBonus('IT', 10);

END;

/

-- Transfer funds (e.g. 200 from account 1 to 2)

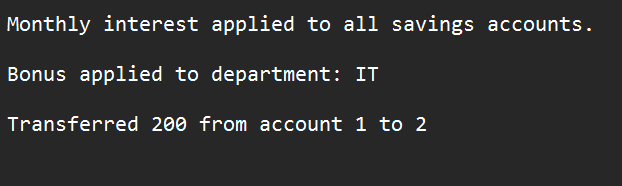
BEGIN

    TransferFunds(1, 2, 200);

END;

/

**Output:**

****