**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.

## ****Scenario 1: Process Monthly Interest for Savings Accounts****

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

BEGIN

UPDATE accounts

SET balance = balance \* 1.01 -- Apply 1% interest

WHERE account\_type = 'SAVINGS'; -- Filter savings accounts

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Monthly interest processed for all savings accounts');

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

RAISE;

END;

/

**Scenario 2: Update Employee Bonuses by Department**

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus(

p\_department\_id IN NUMBER,

p\_bonus\_percent IN NUMBER

) IS

BEGIN

UPDATE employees

SET salary = salary \* (1 + p\_bonus\_percent/100)

WHERE department\_id = p\_department\_id;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Bonuses updated for department ' || p\_department\_id);

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

RAISE;

END;

/

## ****Scenario 3: Transfer Funds Between Accounts****

CREATE OR REPLACE PROCEDURE TransferFunds(

p\_from\_account IN NUMBER,

p\_to\_account IN NUMBER,

p\_amount IN NUMBER

) IS

v\_source\_balance NUMBER;

BEGIN

-- Check source account balance

SELECT balance INTO v\_source\_balance

FROM accounts

WHERE account\_id = p\_from\_account

FOR UPDATE; -- Lock row during transfer

IF v\_source\_balance < p\_amount THEN

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

END IF;

-- Deduct from source

UPDATE accounts

SET balance = balance - p\_amount

WHERE account\_id = p\_from\_account;

-- Add to destination

UPDATE accounts

SET balance = balance + p\_amount

WHERE account\_id = p\_to\_account;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Transferred $' || p\_amount || ' from account ' || p\_from\_account || ' to ' || p\_to\_account);

EXCEPTION

WHEN OTHERS THEN

ROLLBACK;

RAISE;

END;

/