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

**Objective: 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:**

sql

CopyEdit

CREATE OR REPLACE PROCEDURE ProcessMonthlyInterest IS

v\_account\_balance NUMBER;

BEGIN

-- Loop through all savings accounts

FOR account IN (SELECT AccountID, Balance FROM Accounts WHERE AccountType = 'SAVINGS') LOOP

-- Calculate the new balance after applying 1% interest

v\_account\_balance := account.Balance \* 1.01;

-- Update the account balance with the new interest-applied balance

UPDATE Accounts

SET Balance = v\_account\_balance

WHERE AccountID = account.AccountID;

-- Log the updated balance

DBMS\_OUTPUT.PUT\_LINE('Account ' || account.AccountID || ' new balance: ' || v\_account\_balance);

END LOOP;

-- Commit the transaction

COMMIT;

-- Final message indicating the process completion

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

EXCEPTION

WHEN OTHERS THEN

-- In case of any errors, rollback and print error message

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

END ProcessMonthlyInterest;

/

**Expected Output:**

1. **Successful Execution** (if everything goes smoothly):

plaintext

CopyEdit

Account 101 new balance: 1020.00

Account 102 new balance: 2020.00

Account 103 new balance: 3050.00

...

Monthly interest applied successfully to all savings accounts.

1. **Error Occurrence** (in case of errors, e.g., database connection issues):

plaintext

CopyEdit

Error occurred: ORA-XXXXX: some error message

**Scenario 2: Update Employee Bonus Based on Department**

**Objective: 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:**

sql

CopyEdit

CREATE OR REPLACE PROCEDURE UpdateEmployeeBonus (

p\_department\_id IN NUMBER,

p\_bonus\_percentage IN NUMBER

) IS

v\_current\_salary NUMBER;

BEGIN

-- Loop through all employees in the specified department

FOR emp IN (SELECT EmployeeID, Salary FROM Employees WHERE DepartmentID = p\_department\_id) LOOP

-- Calculate the new salary after applying the bonus

v\_current\_salary := emp.Salary \* (1 + p\_bonus\_percentage / 100);

-- Update the employee's salary

UPDATE Employees

SET Salary = v\_current\_salary

WHERE EmployeeID = emp.EmployeeID;

-- Log the updated salary

DBMS\_OUTPUT.PUT\_LINE('Employee ' || emp.EmployeeID || ' new salary: ' || v\_current\_salary);

END LOOP;

-- Commit the transaction

COMMIT;

-- Final message indicating the process completion

DBMS\_OUTPUT.PUT\_LINE('Bonus applied successfully to all employees in department ' || p\_department\_id || '.');

EXCEPTION

WHEN OTHERS THEN

-- In case of any errors, rollback and print error message

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

END UpdateEmployeeBonus;

/

**Expected Output:**

1. **Successful Execution** (if everything goes smoothly):

plaintext

CopyEdit

Employee 1001 new salary: 55000.00

Employee 1002 new salary: 58000.00

Employee 1003 new salary: 49000.00

...

Bonus applied successfully to all employees in department 2.

1. **Error Occurrence** (in case of errors, e.g., database issues):

plaintext

CopyEdit

Error occurred: ORA-XXXXX: some error message

**Scenario 3: Transfer Funds Between Accounts**

**Objective: 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:**

sql

CopyEdit

CREATE OR REPLACE PROCEDURE TransferFunds (

p\_from\_account IN NUMBER,

p\_to\_account IN NUMBER,

p\_amount IN NUMBER

) IS

v\_from\_balance NUMBER;

v\_to\_balance NUMBER;

BEGIN

-- Check the balance of the source account

SELECT Balance INTO v\_from\_balance

FROM Accounts

WHERE AccountID = p\_from\_account;

-- Check if there are sufficient funds in the source account

IF v\_from\_balance < p\_amount THEN

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

END IF;

-- Check the balance of the destination account

SELECT Balance INTO v\_to\_balance

FROM Accounts

WHERE AccountID = p\_to\_account;

-- Perform the transfer

UPDATE Accounts

SET Balance = v\_from\_balance - p\_amount

WHERE AccountID = p\_from\_account;

UPDATE Accounts

SET Balance = v\_to\_balance + p\_amount

WHERE AccountID = p\_to\_account;

-- Commit the transaction

COMMIT;

-- Log the success message

DBMS\_OUTPUT.PUT\_LINE('Transfer of ' || p\_amount || ' from account ' || p\_from\_account || ' to account ' || p\_to\_account || ' completed successfully.');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

-- Handle case where the account does not exist

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error: One or both of the accounts do not exist.');

WHEN OTHERS THEN

-- Handle other unforeseen errors

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END TransferFunds;

/

**Expected Output:**

1. **Successful Transfer** (if there are sufficient funds):

plaintext

CopyEdit

Transfer of 500 from account 101 to account 102 completed successfully.

1. **Insufficient Funds** (if the source account has less than the transfer amount):

plaintext

CopyEdit

Error: Insufficient funds in source account.

1. **Account Not Found** (if either the source or destination account does not exist):

plaintext

CopyEdit

Error: One or both of the accounts do not exist.

1. **Other Errors** (in case of unexpected errors):

plaintext

CopyEdit

Error: ORA-XXXXX: some other error message