**WEEK – 2**

**PLSQL**

**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 on Savings Accounts**

DECLARE

TYPE AccountRec IS RECORD (

acc\_id NUMBER,

balance NUMBER

);

TYPE AccountTable IS TABLE OF AccountRec INDEX BY PLS\_INTEGER;

savings\_accounts AccountTable;

PROCEDURE ProcessMonthlyInterest IS

BEGIN

FOR i IN savings\_accounts.FIRST .. savings\_accounts.LAST LOOP

savings\_accounts(i).balance := savings\_accounts(i).balance + (savings\_accounts(i).balance \* 0.01);

DBMS\_OUTPUT.PUT\_LINE('Account ' || savings\_accounts(i).acc\_id ||

' updated balance: ' || savings\_accounts(i).balance);

END LOOP;

END;

BEGIN

-- Sample savings accounts

savings\_accounts(1) := AccountRec(201, 10000);

savings\_accounts(2) := AccountRec(202, 15000);

savings\_accounts(3) := AccountRec(203, 8000);

-- Apply monthly interest

ProcessMonthlyInterest;

END;

**Output:**

Account 201 updated balance: 10100

Account 202 updated balance: 15150

Account 203 updated balance: 8080

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

DECLARE

TYPE EmployeeRec IS RECORD (

emp\_id NUMBER,

name VARCHAR2(50),

dept VARCHAR2(30),

salary NUMBER

);

TYPE EmployeeTable IS TABLE OF EmployeeRec INDEX BY PLS\_INTEGER;

employees EmployeeTable;

PROCEDURE UpdateEmployeeBonus(department VARCHAR2, bonus\_percent NUMBER) IS

BEGIN

FOR i IN employees.FIRST .. employees.LAST LOOP

IF employees(i).dept = department THEN

employees(i).salary := employees(i).salary + (employees(i).salary \* bonus\_percent / 100);

DBMS\_OUTPUT.PUT\_LINE('Updated salary for ' || employees(i).name ||

': ' || employees(i).salary);

END IF;

END LOOP;

END;

BEGIN

-- Sample employees

employees(1) := EmployeeRec(1, 'Alice', 'HR', 30000);

employees(2) := EmployeeRec(2, 'Bob', 'IT', 40000);

employees(3) := EmployeeRec(3, 'Charlie', 'IT', 35000);

-- Apply bonus to IT department

UpdateEmployeeBonus('IT', 10);

END;

**Output:**

Updated salary for Bob: 44000

Updated salary for Charlie: 38500

**Scenario 3: Transfer Funds Between Accounts**

DECLARE

TYPE AccountRec IS RECORD (

acc\_id NUMBER,

balance NUMBER

);

TYPE AccountTable IS TABLE OF AccountRec INDEX BY PLS\_INTEGER;

accounts AccountTable;

PROCEDURE TransferFunds(from\_acc NUMBER, to\_acc NUMBER, amount NUMBER) IS

from\_index PLS\_INTEGER := -1;

to\_index PLS\_INTEGER := -1;

BEGIN

-- Find account indexes

FOR i IN accounts.FIRST .. accounts.LAST LOOP

IF accounts(i).acc\_id = from\_acc THEN

from\_index := i;

ELSIF accounts(i).acc\_id = to\_acc THEN

to\_index := i;

END IF;

END LOOP;

-- Error checks

IF from\_index = -1 OR to\_index = -1 THEN

DBMS\_OUTPUT.PUT\_LINE('One or both accounts not found.');

RETURN;

END IF;

IF accounts(from\_index).balance < amount THEN

DBMS\_OUTPUT.PUT\_LINE('Insufficient balance in account ' || from\_acc);

RETURN;

END IF;

-- Transfer

accounts(from\_index).balance := accounts(from\_index).balance - amount;

accounts(to\_index).balance := accounts(to\_index).balance + amount;

DBMS\_OUTPUT.PUT\_LINE('Transfer of ' || amount || ' from account ' || from\_acc ||

' to account ' || to\_acc || ' successful.');

DBMS\_OUTPUT.PUT\_LINE('New balance - From Account: ' || accounts(from\_index).balance ||

' | To Account: ' || accounts(to\_index).balance);

END;

BEGIN

-- Sample accounts

accounts(1) := AccountRec(301, 5000);

accounts(2) := AccountRec(302, 3000);

-- Valid transfer

TransferFunds(301, 302, 1000);

-- Invalid transfer

TransferFunds(302, 301, 10000);

END;

**Output:**

Transfer of 1000 from account 301 to account 302 successful.

New balance - From Account: 4000 | To Account: 4000

Insufficient balance in account 302