**Exercise 7: Packages**

**Scenario 1:** Group all customer-related procedures and functions into a package.

**Question:** Create a package **CustomerManagement** with procedures for adding a new customer, updating customer details, and a function to get customer balance.

**Package Specification**

CREATE OR REPLACE PACKAGE CustomerManagement AS

PROCEDURE AddCustomer(

p\_CustomerID NUMBER,

p\_Name VARCHAR2,

p\_DOB DATE,

p\_Balance NUMBER);

PROCEDURE UpdateCustomerDetails(

p\_CustomerID NUMBER,

p\_Name VARCHAR2,

p\_DOB DATE,

p\_Balance NUMBER);

FUNCTION GetCustomerBalance(

p\_CustomerID NUMBER) RETURN NUMBER;

END CustomerManagement;

**Package Body**

CREATE OR REPLACE PACKAGE BODY CustomerManagement AS

PROCEDURE AddCustomer(

p\_CustomerID NUMBER,

p\_Name VARCHAR2,

p\_DOB DATE,

p\_Balance NUMBER)

AS

v\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_count

FROM Customers

WHERE CustomerID = p\_CustomerID;

IF v\_count = 0 THEN

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

VALUES (p\_CustomerID, p\_Name, p\_DOB, p\_Balance, SYSDATE);

DBMS\_OUTPUT.PUT\_LINE('Customer added with ID: ' || p\_CustomerID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Customer with ID: ' || p\_CustomerID || ' already exists.');

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END AddCustomer;

PROCEDURE UpdateCustomerDetails(

p\_CustomerID NUMBER,

p\_Name VARCHAR2,

p\_DOB DATE,

p\_Balance NUMBER)

AS

BEGIN

UPDATE Customers

SET Name = p\_Name,

DOB = p\_DOB,

Balance = p\_Balance,

LastModified = SYSDATE

WHERE CustomerID = p\_CustomerID;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('No customer found with ID: ' || p\_CustomerID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Customer updated with ID: ' || p\_CustomerID);

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END UpdateCustomerDetails;

FUNCTION GetCustomerBalance(

p\_CustomerID NUMBER) RETURN NUMBER

AS

v\_Balance NUMBER;

BEGIN

SELECT Balance INTO v\_Balance

FROM Customers

WHERE CustomerID = p\_CustomerID;

RETURN v\_Balance;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

WHEN OTHERS THEN

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

RETURN NULL;

END GetCustomerBalance;

END CustomerManagement;

**Function Call**

BEGIN

CustomerManagement.AddCustomer(1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1000);

CustomerManagement.UpdateCustomerDetails(1, 'John Doe', TO\_DATE('1985-05-15', 'YYYY-MM-DD'), 1500);

DBMS\_OUTPUT.PUT\_LINE('Customer Balance: ' || CustomerManagement.GetCustomerBalance(1));

END;

**Scenario 2:** Create a package to manage employee data.

**Question:** Write a package **EmployeeManagement** with procedures to hire new employees, update employee details, and a function to calculate annual salary.

**Package Specification**

CREATE OR REPLACE PACKAGE EmployeeManagement AS

PROCEDURE HireEmployee(

p\_EmployeeID NUMBER,

p\_Name VARCHAR2,

p\_Position VARCHAR2,

p\_Salary NUMBER,

p\_Department VARCHAR2,

p\_HireDate DATE);

PROCEDURE UpdateEmployeeDetails(

p\_EmployeeID NUMBER,

p\_Name VARCHAR2,

p\_Position VARCHAR2,

p\_Salary NUMBER,

p\_Department VARCHAR2,

p\_HireDate DATE);

FUNCTION CalculateAnnualSalary(

p\_EmployeeID NUMBER) RETURN NUMBER;

END EmployeeManagement;

**Package Body**

CREATE OR REPLACE PACKAGE BODY EmployeeManagement AS

PROCEDURE HireEmployee(

p\_EmployeeID NUMBER,

p\_Name VARCHAR2,

p\_Position VARCHAR2,

p\_Salary NUMBER,

p\_Department VARCHAR2,

p\_HireDate DATE)

AS

v\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_count

FROM Employees

WHERE EmployeeID = p\_EmployeeID;

IF v\_count = 0 THEN

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

VALUES (p\_EmployeeID, p\_Name, p\_Position, p\_Salary, p\_Department, p\_HireDate);

DBMS\_OUTPUT.PUT\_LINE('Employee hired with ID: ' || p\_EmployeeID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Employee with ID: ' || p\_EmployeeID || ' already exists.');

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END HireEmployee;

PROCEDURE UpdateEmployeeDetails(

p\_EmployeeID NUMBER,

p\_Name VARCHAR2,

p\_Position VARCHAR2,

p\_Salary NUMBER,

p\_Department VARCHAR2,

p\_HireDate DATE)

AS

BEGIN

UPDATE Employees

SET Name = p\_Name,

Position = p\_Position,

Salary = p\_Salary,

Department = p\_Department,

HireDate = p\_HireDate

WHERE EmployeeID = p\_EmployeeID;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('No employee found with ID: ' || p\_EmployeeID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Employee updated with ID: ' || p\_EmployeeID);

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END UpdateEmployeeDetails;

FUNCTION CalculateAnnualSalary(

p\_EmployeeID NUMBER) RETURN NUMBER

AS

v\_Salary NUMBER;

v\_AnnualSalary NUMBER;

BEGIN

SELECT Salary INTO v\_Salary

FROM Employees

WHERE EmployeeID = p\_EmployeeID;

v\_AnnualSalary := v\_Salary \* 12;

RETURN v\_AnnualSalary;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

WHEN OTHERS THEN

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

RETURN NULL;

END CalculateAnnualSalary;

END EmployeeManagement;

**Function Call**

BEGIN

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

EmployeeManagement.UpdateEmployeeDetails(1, 'Alice Johnson', 'Senior Manager', 80000, 'HR', TO\_DATE('2015-06-15', 'YYYY-MM-DD'));

DBMS\_OUTPUT.PUT\_LINE('Annual Salary: ' || EmployeeManagement.CalculateAnnualSalary(1));

END;

**Scenario 3:** Group all account-related operations into a package.

**Question:** Create a package **AccountOperations** with procedures for opening a new account, closing an account, and a function to get the total balance of a customer across all accounts.

**Package Specification**

CREATE OR REPLACE PACKAGE AccountOperations AS

PROCEDURE OpenAccount(

p\_AccountID NUMBER,

p\_CustomerID NUMBER,

p\_AccountType VARCHAR2,

p\_Balance NUMBER);

PROCEDURE CloseAccount(

p\_AccountID NUMBER);

FUNCTION GetTotalBalance(

p\_CustomerID NUMBER) RETURN NUMBER;

END AccountOperations;

**Package Body**

CREATE OR REPLACE PACKAGE BODY AccountOperations AS

PROCEDURE OpenAccount(

p\_AccountID NUMBER,

p\_CustomerID NUMBER,

p\_AccountType VARCHAR2,

p\_Balance NUMBER)

AS

v\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_count

FROM Accounts

WHERE AccountID = p\_AccountID;

IF v\_count = 0 THEN

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

VALUES (p\_AccountID, p\_CustomerID, p\_AccountType, p\_Balance, SYSDATE);

DBMS\_OUTPUT.PUT\_LINE('Account opened with ID: ' || p\_AccountID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Account with ID: ' || p\_AccountID || ' already exists.');

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END OpenAccount;

PROCEDURE CloseAccount(

p\_AccountID NUMBER)

AS

v\_count NUMBER;

BEGIN

DELETE FROM Accounts

WHERE AccountID = p\_AccountID;

IF SQL%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('No account found with ID: ' || p\_AccountID);

ELSE

DBMS\_OUTPUT.PUT\_LINE('Account closed with ID: ' || p\_AccountID);

END IF;

EXCEPTION

WHEN OTHERS THEN

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

END CloseAccount;

FUNCTION GetTotalBalance(

p\_CustomerID NUMBER) RETURN NUMBER

AS

v\_TotalBalance NUMBER;

BEGIN

SELECT SUM(Balance) INTO v\_TotalBalance

FROM Accounts

WHERE CustomerID = p\_CustomerID;

RETURN v\_TotalBalance;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

WHEN OTHERS THEN

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

RETURN 0;

END GetTotalBalance;

END AccountOperations;

**Function Call**

BEGIN

AccountOperations.OpenAccount(1, 100, 'Savings', 5000);

AccountOperations.CloseAccount(1);

DBMS\_OUTPUT.PUT\_LINE('Total Balance: ' || AccountOperations.GetTotalBalance(100));

END;