**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.

**CODE:**

CREATE OR REPLACE PACKAGE CustomerManagement AS

PROCEDURE AddCustomer(p\_CID NUMBER, p\_Name VARCHAR2, p\_DOB DATE, p\_Balance NUMBER);

PROCEDURE UpdateCustomerDetails(p\_CID NUMBER, p\_Balance NUMBER);

FUNCTION GetBalance(p\_CID NUMBER) RETURN NUMBER;

END;

/

select \* from Customers;

**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.

**CODE:**

CREATE OR REPLACE PACKAGE BODY EmployeeManagement AS

PROCEDURE HireEmployee(p\_EID NUMBER, p\_Name VARCHAR2, p\_Pos VARCHAR2, p\_Sal NUMBER, p\_Dept VARCHAR2) IS

BEGIN

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

VALUES (p\_EID, p\_Name, p\_Pos, p\_Sal, p\_Dept, SYSDATE);

END HireEmployee;

PROCEDURE UpdateEmployee(p\_EID NUMBER, p\_Sal NUMBER) IS

BEGIN

UPDATE Employees

SET Salary = p\_Sal

WHERE EmployeeID = p\_EID;

IF SQL%ROWCOUNT = 0 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Employee not found for ID: ' || p\_EID);

END IF;

END UpdateEmployee;

FUNCTION AnnualSalary(p\_EID NUMBER) RETURN NUMBER IS

v\_Sal NUMBER;

BEGIN

SELECT Salary INTO v\_Sal

FROM Employees

WHERE EmployeeID = p\_EID;

RETURN v\_Sal \* 12;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL; -- Return NULL if employee not found

END AnnualSalary;

END EmployeeManagement;

/

BEGIN

EmployeeManagement.HireEmployee(41, 'Alice Smith', 'Developer', 60000, 'IT');

END;

/

BEGIN

EmployeeManagement.UpdateEmployee(41, 65000);

END;

/

DECLARE

v\_AnnualSal NUMBER;

BEGIN

v\_AnnualSal := EmployeeManagement.AnnualSalary(1);

DBMS\_OUTPUT.PUT\_LINE('Annual Salary for Employee 1: ' || v\_AnnualSal);

END;

/

select \* from Employees;

**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.

**CODE:**

CREATE OR REPLACE PACKAGE BODY AccountOperations AS

PROCEDURE OpenAccount(p\_AID NUMBER, p\_CID NUMBER, p\_Type VARCHAR2, p\_Balance NUMBER) IS

BEGIN

INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified) VALUES (p\_AID, p\_CID, p\_Type, p\_Balance, SYSDATE);

END;

PROCEDURE CloseAccount(p\_AID NUMBER) IS

BEGIN

DELETE FROM Accounts WHERE AccountID = p\_AID;

END;

FUNCTION TotalBalance(p\_CID NUMBER) RETURN NUMBER IS

v\_Total NUMBER;

BEGIN

SELECT SUM(Balance) INTO v\_Total FROM Accounts WHERE CustomerID = p\_CID;

RETURN NVL(v\_Total, 0);

END;

END;

/

BEGIN

AccountOperations.OpenAccount(31, 101, 'Savings', 5000);

END;

/

SELECT \* FROM Accounts WHERE AccountID = 31;