**Exercise 7: Packages**

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

CREATE OR REPLACE PACKAGE CustomerManagement AS

-- Procedure to add a new customer

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

-- Procedure to update customer details

PROCEDURE UpdateCustomer(p\_CustomerID IN NUMBER, p\_Name IN VARCHAR2, p\_Balance IN NUMBER);

-- Function to get customer balance

FUNCTION GetCustomerBalance(p\_CustomerID IN NUMBER) RETURN NUMBER;

END CustomerManagement;

/

**Package body**

CREATE OR REPLACE PACKAGE BODY CustomerManagement AS

PROCEDURE AddCustomer(p\_CustomerID IN NUMBER, p\_Name IN VARCHAR2, p\_DOB IN DATE, p\_Balance IN NUMBER) IS

BEGIN

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

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

END AddCustomer;

PROCEDURE UpdateCustomer(p\_CustomerID IN NUMBER, p\_Name IN VARCHAR2, p\_Balance IN NUMBER) IS

BEGIN

UPDATE Customers

SET Name = p\_Name, Balance = p\_Balance, LastModified = SYSDATE

WHERE CustomerID = p\_CustomerID;

END UpdateCustomer;

FUNCTION GetCustomerBalance(p\_CustomerID IN NUMBER) RETURN NUMBER IS

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;

END GetCustomerBalance;

END CustomerManagement;

/

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

CREATE OR REPLACE PACKAGE EmployeeManagement AS

-- Procedure to hire a new employee

PROCEDURE HireEmployee(p\_EmployeeID IN NUMBER, p\_Name IN VARCHAR2, p\_Position IN VARCHAR2, p\_Salary IN NUMBER, p\_Department IN VARCHAR2, p\_HireDate IN DATE);

-- Procedure to update employee details

PROCEDURE UpdateEmployee(p\_EmployeeID IN NUMBER, p\_Name IN VARCHAR2, p\_Salary IN NUMBER, p\_Position IN VARCHAR2, p\_Department IN VARCHAR2);

-- Function to calculate annual salary

FUNCTION CalculateAnnualSalary(p\_EmployeeID IN NUMBER) RETURN NUMBER;

END EmployeeManagement;

/

**Package body**

CREATE OR REPLACE PACKAGE BODY EmployeeManagement AS

PROCEDURE HireEmployee(p\_EmployeeID IN NUMBER, p\_Name IN VARCHAR2, p\_Position IN VARCHAR2, p\_Salary IN NUMBER, p\_Department IN VARCHAR2, p\_HireDate IN DATE) IS

BEGIN

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

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

END HireEmployee;

PROCEDURE UpdateEmployee(p\_EmployeeID IN NUMBER, p\_Name IN VARCHAR2, p\_Salary IN NUMBER, p\_Position IN VARCHAR2, p\_Department IN VARCHAR2) IS

BEGIN

UPDATE Employees

SET Name = p\_Name, Salary = p\_Salary, Position = p\_Position, Department = p\_Department

WHERE EmployeeID = p\_EmployeeID;

END UpdateEmployee;

FUNCTION CalculateAnnualSalary(p\_EmployeeID IN NUMBER) RETURN NUMBER IS

v\_Salary NUMBER;

BEGIN

SELECT Salary INTO v\_Salary

FROM Employees

WHERE EmployeeID = p\_EmployeeID;

RETURN v\_Salary \* 12; -- Assuming salary is monthly

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END CalculateAnnualSalary;

END EmployeeManagement;

/

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

CREATE OR REPLACE PACKAGE AccountOperations AS

-- Procedure to open a new account

PROCEDURE OpenAccount(p\_AccountID IN NUMBER, p\_CustomerID IN NUMBER, p\_AccountType IN VARCHAR2, p\_Balance IN NUMBER);

-- Procedure to close an account

PROCEDURE CloseAccount(p\_AccountID IN NUMBER);

-- Function to get the total balance of a customer across all accounts

FUNCTION GetTotalCustomerBalance(p\_CustomerID IN NUMBER) RETURN NUMBER;

END AccountOperations;

/

**Package Body**

CREATE OR REPLACE PACKAGE BODY AccountOperations AS

PROCEDURE OpenAccount(p\_AccountID IN NUMBER, p\_CustomerID IN NUMBER, p\_AccountType IN VARCHAR2, p\_Balance IN NUMBER) IS

BEGIN

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

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

END OpenAccount;

PROCEDURE CloseAccount(p\_AccountID IN NUMBER) IS

BEGIN

DELETE FROM Accounts

WHERE AccountID = p\_AccountID;

END CloseAccount;

FUNCTION GetTotalCustomerBalance(p\_CustomerID IN NUMBER) RETURN NUMBER IS

v\_TotalBalance NUMBER;

BEGIN

SELECT SUM(Balance) INTO v\_TotalBalance

FROM Accounts

WHERE CustomerID = p\_CustomerID;

RETURN NVL(v\_TotalBalance, 0); -- Return 0 if no accounts found

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

END GetTotalCustomerBalance;

END AccountOperations;

/