Exercise 7: Packages

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

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

CREATE OR REPLACE PACKAGE CustomerManagement

AS

PROCEDURE AddCustomer (

p\_customer\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_dob IN DATE,

p\_balance IN NUMBER

);

PROCEDURE UpdateCustomer (

p\_customer\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_dob IN DATE,

p\_balance IN NUMBER

);

FUNCTION GetCustomerBalance (p\_customer\_id IN NUMBER) RETURN NUMBER;

END CustomerManagement;

/

CREATE OR REPLACE PACKAGE BODY CustomerManagement

AS

PROCEDURE AddCustomer (

p\_customer\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_dob IN DATE,

p\_balance IN NUMBER

)

IS

BEGIN

END AddCustomer;

PROCEDURE UpdateCustomer (

p\_customer\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_dob IN DATE,

p\_balance IN NUMBER

)

IS

BEGIN

END UpdateCustomer;

FUNCTION GetCustomerBalance (p\_customer\_id IN NUMBER)

RETURN NUMBER

IS

v\_balance NUMBER;

BEGIN

SELECT Balance INTO v\_balance FROM Customers WHERE CustomerID = p\_customer\_id;

RETURN v\_balance;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

END GetCustomerBalance;

END CustomerManagement;

/

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

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

CREATE OR REPLACE PACKAGE EmployeeManagement

AS

PROCEDURE HireEmployee (

p\_employee\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_position IN VARCHAR2,

p\_salary IN NUMBER,

p\_department IN VARCHAR2,

p\_hire\_date IN DATE

);

PROCEDURE UpdateEmployee (

p\_employee\_id IN NUMBER,

p\_name IN VARCHAR2,

p\_position IN VARCHAR2,

p\_salary IN NUMBER,

p\_department IN VARCHAR2

);

FUNCTION CalculateAnnualSalary (p\_employee\_id IN NUMBER) RETURN NUMBER;

END EmployeeManagement;

/

CREATE OR REPLACE PACKAGE BODY EmployeeManagement

AS

PROCEDURE HireEmployee (

)

IS

BEGIN

END HireEmployee;

PROCEDURE UpdateEmployee (

)

IS

BEGIN

END UpdateEmployee;

FUNCTION CalculateAnnualSalary (p\_employee\_id IN NUMBER)

RETURN NUMBER

IS

v\_salary NUMBER;

BEGIN

SELECT Salary INTO v\_salary FROM Employees WHERE EmployeeID = p\_employee\_id;

RETURN v\_salary \* 12

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 0;

END CalculateAnnualSalary;

END EmployeeManagement;

/

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

**o 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.**

CREATE OR REPLACE PACKAGE AccountOperations

AS

PROCEDURE OpenAccount (

p\_account\_id IN NUMBER,

p\_customer\_id IN NUMBER,

p\_account\_type IN VARCHAR2,

p\_balance IN NUMBER

);

PROCEDURE CloseAccount (p\_account\_id IN NUMBER);

FUNCTION GetTotalBalance (p\_customer\_id IN NUMBER) RETURN NUMBER;

END AccountOperations;

/

CREATE OR REPLACE PACKAGE BODY AccountOperations

AS

PROCEDURE OpenAccount (

)

IS

BEGIN

END OpenAccount;

PROCEDURE CloseAccount (p\_account\_id IN NUMBER)

IS

BEGIN

END CloseAccount;

FUNCTION GetTotalBalance (p\_customer\_id IN NUMBER)

RETURN NUMBER

IS

v\_total\_balance NUMBER := 0;

BEGIN

FOR r IN (SELECT Balance FROM Accounts WHERE CustomerID = p\_customer\_id) LOOP

v\_total\_balance := v\_total\_balance + r.Balance;

END LOOP;

RETURN v\_total\_balance;

END GetTotalBalance;

END AccountOperations;

/