**Packages**

**Scenario 1: Customer Management Package**

**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\_customerID IN NUMBER, p\_name IN VARCHAR2, p\_dob IN DATE, p\_balance IN NUMBER);

PROCEDURE UpdateCustomer(p\_customerID IN NUMBER, p\_name IN VARCHAR2, p\_dob IN DATE, p\_balance IN NUMBER);

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

END CustomerManagement;

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);

DBMS\_OUTPUT.PUT\_LINE('Customer ' || p\_name || ' added.');

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

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

END AddCustomer;

PROCEDURE UpdateCustomer(p\_customerID IN NUMBER, p\_name IN VARCHAR2, p\_dob IN DATE, p\_balance IN NUMBER) IS

BEGIN

UPDATE Customers

SET Name = p\_name, DOB = p\_dob, Balance = p\_balance, LastModified = SYSDATE

WHERE CustomerID = p\_customerID;

DBMS\_OUTPUT.PUT\_LINE('Customer ID ' || p\_customerID || ' updated.');

EXCEPTION

WHEN OTHERS THEN

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

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: Employee Management Package**

**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\_employeeID IN NUMBER, p\_name IN VARCHAR2, p\_position IN VARCHAR2, p\_salary IN NUMBER, p\_department IN VARCHAR2);

PROCEDURE UpdateEmployee(p\_employeeID IN NUMBER, p\_name IN VARCHAR2, p\_position IN VARCHAR2, p\_salary IN NUMBER, p\_department IN VARCHAR2);

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

END EmployeeManagement;

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) IS

BEGIN

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

VALUES (p\_employeeID, p\_name, p\_position, p\_salary, p\_department, SYSDATE);

DBMS\_OUTPUT.PUT\_LINE('Employee ' || p\_name || ' hired.');

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

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

END HireEmployee;

PROCEDURE UpdateEmployee(p\_employeeID IN NUMBER, p\_name IN VARCHAR2, p\_position IN VARCHAR2, p\_salary IN NUMBER, p\_department IN VARCHAR2) IS

BEGIN

UPDATE Employees

SET Name = p\_name, Position = p\_position, Salary = p\_salary, Department = p\_department

WHERE EmployeeID = p\_employeeID;

DBMS\_OUTPUT.PUT\_LINE('Employee ID ' || p\_employeeID || ' updated.');

EXCEPTION

WHEN OTHERS THEN

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

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;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

END CalculateAnnualSalary;

END EmployeeManagement;

**Scenario 3: Account Operations 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.

CREATE OR REPLACE PACKAGE AccountOperations AS

PROCEDURE OpenAccount(p\_accountID IN NUMBER, p\_customerID IN NUMBER, p\_accountType IN VARCHAR2, p\_balance IN NUMBER);

PROCEDURE CloseAccount(p\_accountID IN NUMBER);

FUNCTION GetTotalBalance(p\_customerID IN NUMBER) RETURN NUMBER;

END AccountOperations;

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);

DBMS\_OUTPUT.PUT\_LINE('Account ' || p\_accountID || ' opened for Customer ID: ' || p\_customerID);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

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

END OpenAccount;

PROCEDURE CloseAccount(p\_accountID IN NUMBER) IS

BEGIN

DELETE FROM Accounts WHERE AccountID = p\_accountID;

DBMS\_OUTPUT.PUT\_LINE('Account ' || p\_accountID || ' closed.');

EXCEPTION

WHEN OTHERS THEN

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

END CloseAccount;

FUNCTION GetTotalBalance(p\_customerID IN NUMBER) RETURN NUMBER IS

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 NULL;

END GetTotalBalance;

END AccountOperations;