```
CREATE OR REPLACE PACKAGE CustomerManagement AS
   PROCEDURE AddNewCustomer(p_CustomerID IN NUMBER, p_Name IN VARCHAR2, p_DOB IN DATE, p_Balance IN NUMBER);
   PROCEDURE UpdateCustomerDetails(p_CustomerID IN NUMBER, p_Name IN VARCHAR2, p_Balance IN NUMBER);
   FUNCTION GetCustomerBalance(p_CustomerID IN NUMBER) RETURN NUMBER;
END CustomerManagement;
CREATE OR REPLACE PACKAGE BODY CustomerManagement AS
   PROCEDURE AddNewCustomer(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 AddNewCustomer;

   PROCEDURE UpdateCustomerDetails(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 UpdateCustomerDetails;

   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; -- Or handle exception as needed
   END GetCustomerBalance;
```

```
END CustomerManagement;

CREATE OR REPLACE PACKAGE EmployeeManagement AS

  PROCEDURE HireNewEmployee(p_EmployeeID IN NUMBER, p_Name IN VARCHAR2, p_Position IN VARCHAR2, p_Salary IN NUMBER, p_Department IN VARCHAR2);

  PROCEDURE UpdateEmployeeDetails(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 HireNewEmployee(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);

  END HireNewEmployee;


  PROCEDURE UpdateEmployeeDetails(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;

  END UpdateEmployeeDetails;


  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; -- Or handle exception as needed
```

```
    END CalculateAnnualSalary;
END EmployeeManagement;
CREATE OR REPLACE PACKAGE AccountOperations AS
    PROCEDURE OpenNewAccount(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 OpenNewAccount(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 OpenNewAccount;

    PROCEDURE CloseAccount(p_AccountID IN NUMBER) IS
    BEGIN
        DELETE FROM Accounts WHERE AccountID = p_AccountID;
    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
        GROUP BY CustomerID;
        RETURN v_TotalBalance;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            RETURN 0; -- Or handle exception as needed
```

```
    END GetTotalBalance;
END AccountOperations;
```