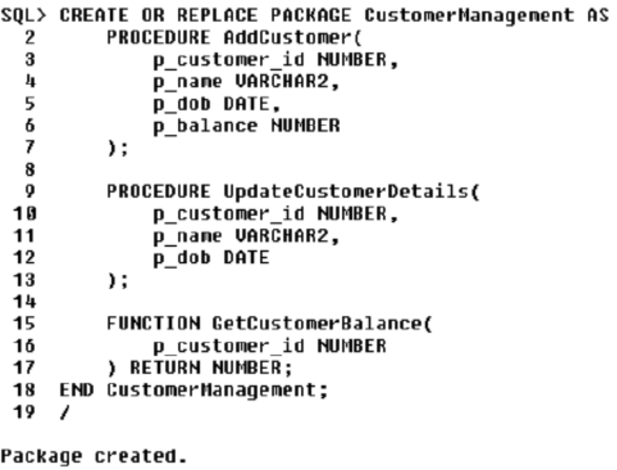
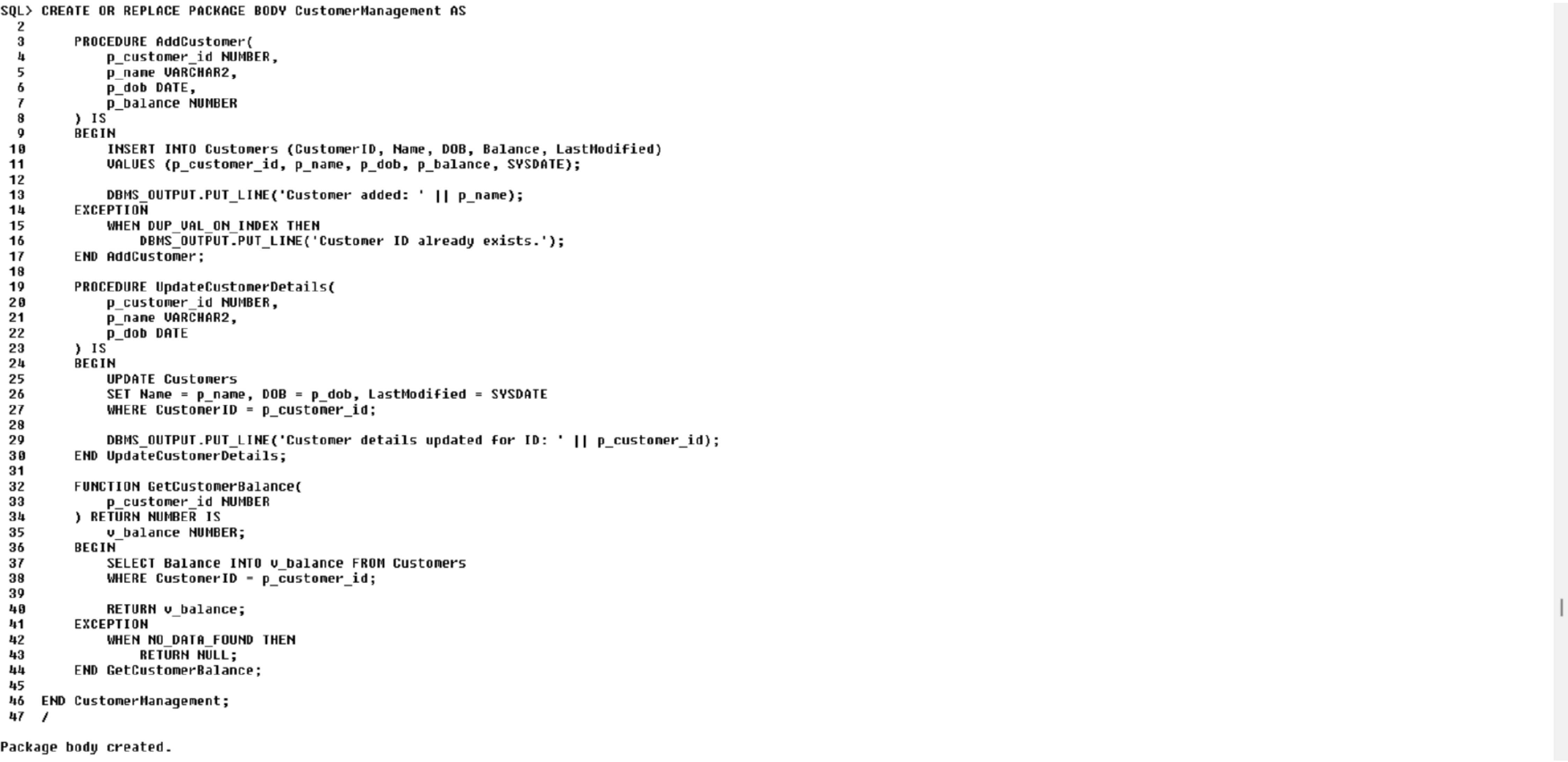
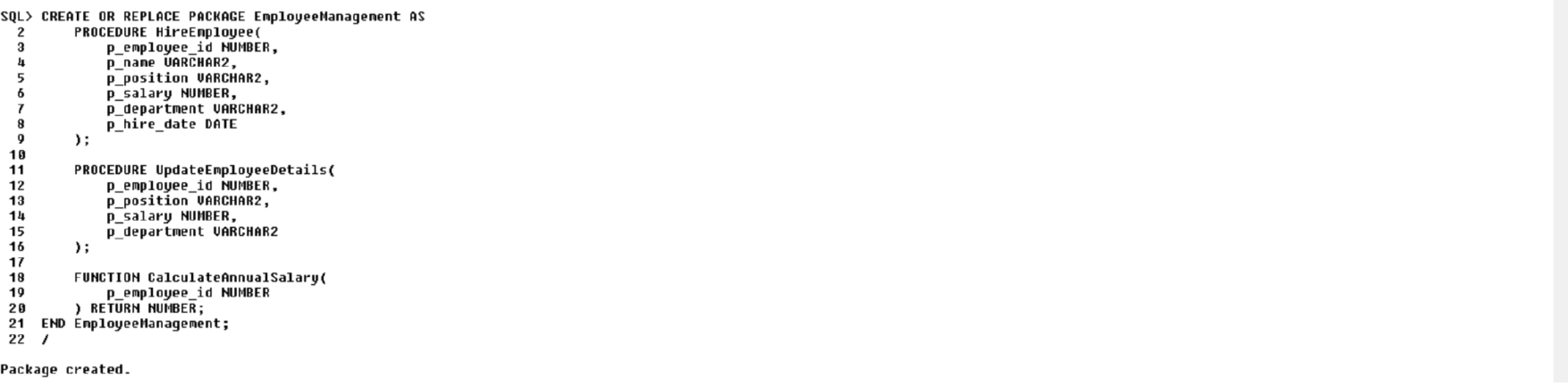


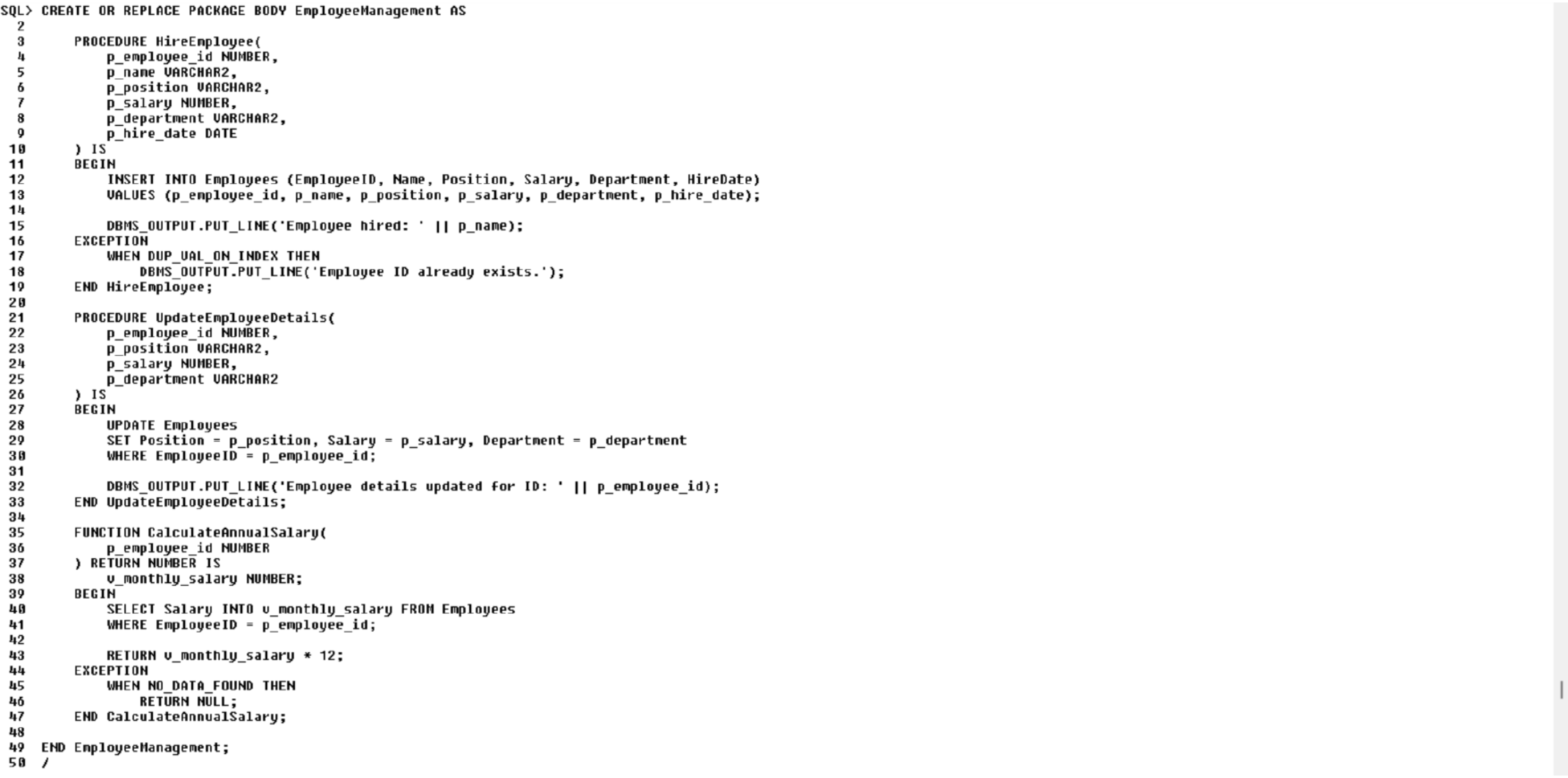
```
SQL> CREATE OR REPLACE PACKAGE CustomerManagement AS
    PROCEDURE AddCustomer(
        p_customer_id NUMBER,
        p_name VARCHAR2,
        p_dob DATE,
        p_balance NUMBER
    );

    PROCEDURE UpdateCustomerDetails(
        p_customer_id NUMBER,
        p_name VARCHAR2,
        p_dob DATE
    );

    FUNCTION GetCustomerBalance(
        p_customer_id NUMBER
    ) RETURN NUMBER;
END CustomerManagement;
/

Package created.
```



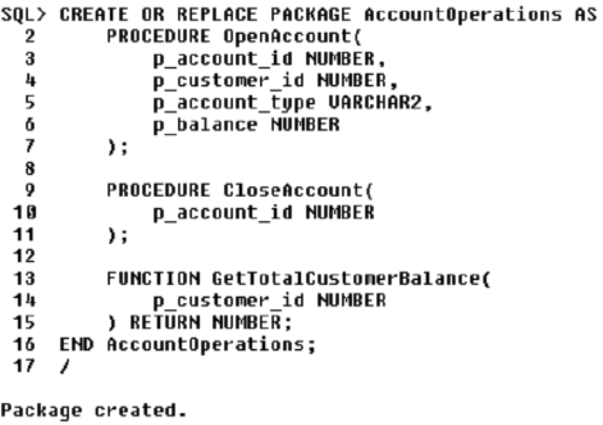
```
SQL> CREATE OR REPLACE PACKAGE BODY CustomerManagement AS

    PROCEDURE AddCustomer(
        p_customer_id NUMBER,
        p_name VARCHAR2,
        p_dob DATE,
        p_balance NUMBER
    ) IS
    BEGIN
        INSERT INTO Customers (CustomerID, Name, DOB, Balance, LastModified)
        VALUES (p_customer_id, p_name, p_dob, p_balance, SYSDATE);

        DBMS_OUTPUT.PUT_LINE('Customer added: ' || p_name);
    EXCEPTION
        WHEN DUP_VAL_ON_INDEX THEN
            DBMS_OUTPUT.PUT_LINE('Customer ID already exists.');
    END AddCustomer;

    PROCEDURE UpdateCustomerDetails(
        p_customer_id NUMBER,
        p_name VARCHAR2,
        p_dob DATE
    ) IS
    BEGIN
        UPDATE Customers
        SET Name = p_name, DOB = p_dob, LastModified = SYSDATE
        WHERE CustomerID = p_customer_id;

        DBMS_OUTPUT.PUT_LINE('Customer details updated for ID: ' || p_customer_id);
    END UpdateCustomerDetails;

    FUNCTION GetCustomerBalance(
        p_customer_id 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 NULL;
    END GetCustomerBalance;

END CustomerManagement;
/

Package body created.
```



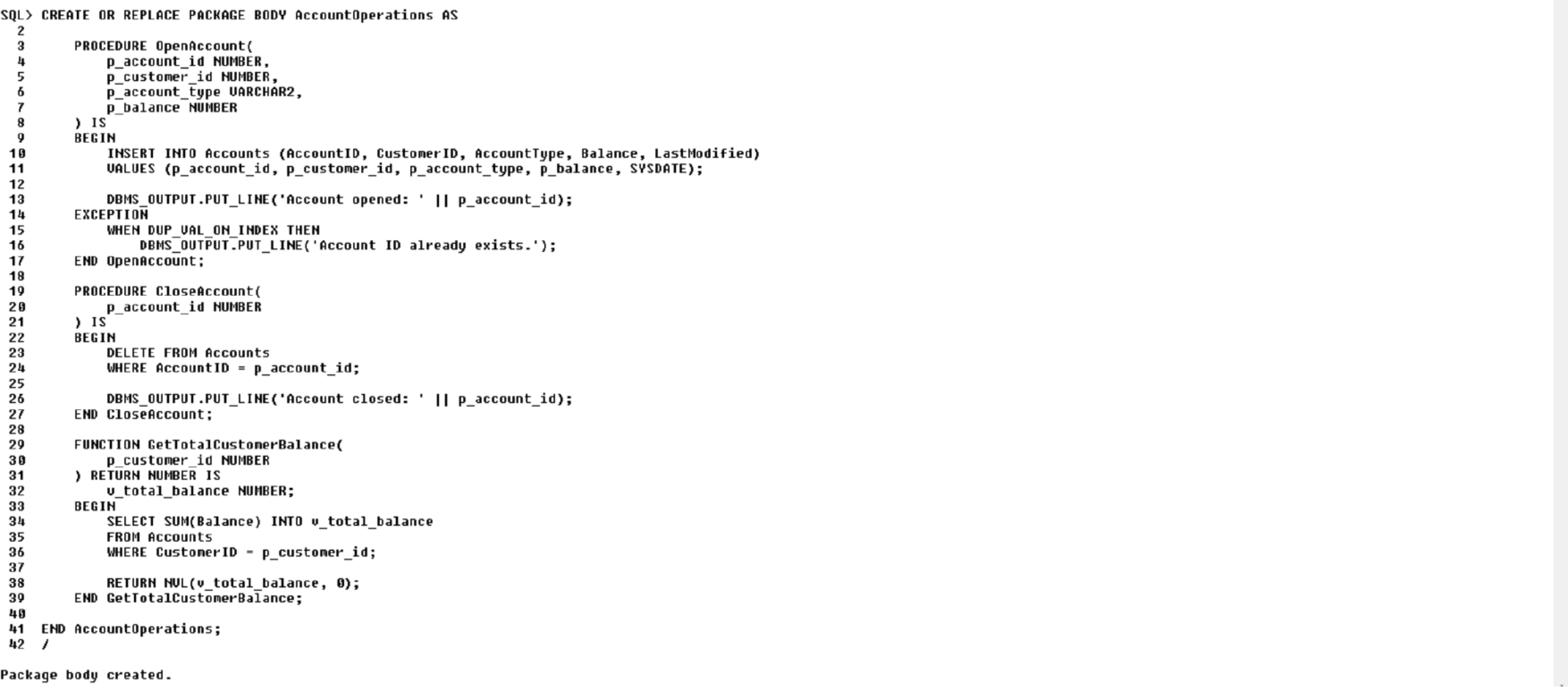
```
SQL> CREATE OR REPLACE PACKAGE EmployeeManagement AS
  2      PROCEDURE HireEmployee(
  3          p_employee_id NUMBER,
  4          p_name VARCHAR2,
  5          p_position VARCHAR2,
  6          p_salary NUMBER,
  7          p_department VARCHAR2,
  8          p_hire_date DATE
  9      );
 10
 11      PROCEDURE UpdateEmployeeDetails(
 12          p_employee_id NUMBER,
 13          p_position VARCHAR2,
 14          p_salary NUMBER,
 15          p_department VARCHAR2
 16      );
 17
 18      FUNCTION CalculateAnnualSalary(
 19          p_employee_id NUMBER
 20      ) RETURN NUMBER;
 21  END EmployeeManagement;
 22  /

Package created.
```



```
SQL> CREATE OR REPLACE PACKAGE BODY EmployeeManagement AS

    PROCEDURE HireEmployee(
        p_employee_id NUMBER,
        p_name VARCHAR2,
        p_position VARCHAR2,
        p_salary NUMBER,
        p_department VARCHAR2,
        p_hire_date DATE
    ) IS
    BEGIN
        INSERT INTO Employees (EmployeeID, Name, Position, Salary, Department, HireDate)
        VALUES (p_employee_id, p_name, p_position, p_salary, p_department, p_hire_date);

        DBMS_OUTPUT.PUT_LINE('Employee hired: ' || p_name);
    EXCEPTION
        WHEN DUP_VAL_ON_INDEX THEN
            DBMS_OUTPUT.PUT_LINE('Employee ID already exists.');
    END HireEmployee;

    PROCEDURE UpdateEmployeeDetails(
        p_employee_id NUMBER,
        p_position VARCHAR2,
        p_salary NUMBER,
        p_department VARCHAR2
    ) IS
    BEGIN
        UPDATE Employees
        SET Position = p_position, Salary = p_salary, Department = p_department
        WHERE EmployeeID = p_employee_id;

        DBMS_OUTPUT.PUT_LINE('Employee details updated for ID: ' || p_employee_id);
    END UpdateEmployeeDetails;

    FUNCTION CalculateAnnualSalary(
        p_employee_id NUMBER
    ) RETURN NUMBER IS
        v_monthly_salary NUMBER;
    BEGIN
        SELECT Salary INTO v_monthly_salary FROM Employees
        WHERE EmployeeID = p_employee_id;

        RETURN v_monthly_salary * 12;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            RETURN NULL;
    END CalculateAnnualSalary;

END EmployeeManagement;
/
```



```
SQL> CREATE OR REPLACE PACKAGE AccountOperations AS
         PROCEDURE OpenAccount(
             p_account_id NUMBER,
             p_customer_id NUMBER,
             p_account_type VARCHAR2,
             p_balance NUMBER
         );

         PROCEDURE CloseAccount(
             p_account_id NUMBER
         );

         FUNCTION GetTotalCustomerBalance(
             p_customer_id NUMBER
         ) RETURN NUMBER;
     END AccountOperations;
     /

Package created.
```



```
SQL> CREATE OR REPLACE PACKAGE BODY AccountOperations AS

         PROCEDURE OpenAccount(
             p_account_id NUMBER,
             p_customer_id NUMBER,
             p_account_type VARCHAR2,
             p_balance NUMBER
         ) IS
         BEGIN
             INSERT INTO Accounts (AccountID, CustomerID, AccountType, Balance, LastModified)
             VALUES (p_account_id, p_customer_id, p_account_type, p_balance, SYSDATE);

             DBMS_OUTPUT.PUT_LINE('Account opened: ' || p_account_id);
         EXCEPTION
             WHEN DUP_VAL_ON_INDEX THEN
                 DBMS_OUTPUT.PUT_LINE('Account ID already exists.');
         END OpenAccount;

         PROCEDURE CloseAccount(
             p_account_id NUMBER
         ) IS
         BEGIN
             DELETE FROM Accounts
             WHERE AccountID = p_account_id;

             DBMS_OUTPUT.PUT_LINE('Account closed: ' || p_account_id);
         END CloseAccount;

         FUNCTION GetTotalCustomerBalance(
             p_customer_id NUMBER
         ) RETURN NUMBER IS
             v_total_balance NUMBER;
         BEGIN
             SELECT SUM(Balance) INTO v_total_balance
             FROM Accounts
             WHERE CustomerID = p_customer_id;

             RETURN NVL(v_total_balance, 0);
         END GetTotalCustomerBalance;

     END AccountOperations;
     /

Package body created.
```