**CODE:**

SET SERVEROUTPUT ON;

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE accounts CASCADE CONSTRAINTS';

EXCEPTION

WHEN OTHERS THEN

IF SQLCODE != -942 THEN

RAISE;

END IF;

END;

/

CREATE TABLE accounts (

AccountID NUMBER PRIMARY KEY,

CustomerID NUMBER,

AccountType VARCHAR2(20),

Balance NUMBER,

LastModified DATE

);

/

CREATE OR REPLACE PACKAGE AccountOperations AS

PROCEDURE OpenAccount(p\_accID NUMBER, p\_custID NUMBER, p\_type VARCHAR2, p\_balance NUMBER);

PROCEDURE CloseAccount(p\_accID NUMBER);

FUNCTION GetTotalBalance(p\_custID NUMBER) RETURN NUMBER;

END AccountOperations;

/

CREATE OR REPLACE PACKAGE BODY AccountOperations AS

PROCEDURE OpenAccount(p\_accID NUMBER, p\_custID NUMBER, p\_type VARCHAR2, p\_balance NUMBER) IS

BEGIN

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

VALUES (p\_accID, p\_custID, p\_type, p\_balance, SYSDATE);

END;

PROCEDURE CloseAccount(p\_accID NUMBER) IS

BEGIN

DELETE FROM accounts WHERE AccountID = p\_accID;

END;

FUNCTION GetTotalBalance(p\_custID NUMBER) RETURN NUMBER IS

v\_total NUMBER := 0;

BEGIN

SELECT NVL(SUM(Balance), 0) INTO v\_total

FROM accounts

WHERE CustomerID = p\_custID;

RETURN v\_total;

END;

END AccountOperations;

/

BEGIN

AccountOperations.OpenAccount(101, 1, 'Savings', 2500);

AccountOperations.OpenAccount(102, 1, 'Checking', 1500);

END;

/

DECLARE

v\_balance NUMBER;

BEGIN

v\_balance := AccountOperations.GetTotalBalance(1);

DBMS\_OUTPUT.PUT\_LINE('Total Balance for Customer 1: ' || v\_balance);

END;

/

BEGIN

AccountOperations.CloseAccount(102);

END;

/

DECLARE

v\_balance NUMBER;

BEGIN

v\_balance := AccountOperations.GetTotalBalance(1);

DBMS\_OUTPUT.PUT\_LINE('Balance after closing one account: ' || v\_balance);

END;

/

**OUTPUT:**

