DO $$

BEGIN

EXECUTE 'DROP TABLE IF EXISTS Transactions CASCADE';

EXECUTE 'DROP TABLE IF EXISTS Accounts CASCADE';

EXECUTE 'DROP TABLE IF EXISTS Loans CASCADE';

EXECUTE 'DROP TABLE IF EXISTS Employees CASCADE';

EXECUTE 'DROP TABLE IF EXISTS Customers CASCADE';

END

$$;

CREATE TABLE Customers (

CustomerID SERIAL PRIMARY KEY,

Name VARCHAR(100),

DOB DATE,

Balance NUMERIC DEFAULT 0,

LastModified DATE DEFAULT CURRENT\_DATE

);

CREATE TABLE Accounts (

AccountID SERIAL PRIMARY KEY,

CustomerID INT REFERENCES Customers(CustomerID),

AccountType VARCHAR(20),

Balance NUMERIC DEFAULT 0,

LastModified DATE DEFAULT CURRENT\_DATE

);

CREATE TABLE Employees (

EmployeeID SERIAL PRIMARY KEY,

Name VARCHAR(100),

Position VARCHAR(50),

Salary NUMERIC,

Department VARCHAR(50),

HireDate DATE DEFAULT CURRENT\_DATE

);

INSERT INTO Customers(Name, DOB, Balance) VALUES ('Martina', '1985-05-15', 5000);

INSERT INTO Employees(Name, Position, Salary, Department) VALUES ('Roselin', 'Manager', 70000, 'HR');

INSERT INTO Accounts(CustomerID, AccountType, Balance) VALUES (1, 'Savings', 5000);

-- ===========================================

-- PACKAGE: CustomerManagement

-- ===========================================

CREATE OR REPLACE PROCEDURE customer\_management\_add\_customer(

p\_name VARCHAR,

p\_dob DATE,

p\_balance NUMERIC

)

LANGUAGE plpgsql AS $$

BEGIN

INSERT INTO Customers(Name, DOB, Balance, LastModified)

VALUES (p\_name, p\_dob, p\_balance, CURRENT\_DATE);

END;

$$;

CREATE OR REPLACE PROCEDURE customer\_management\_update\_customer(

p\_customer\_id INT,

p\_name VARCHAR,

p\_dob DATE

)

LANGUAGE plpgsql AS $$

BEGIN

UPDATE Customers

SET Name = p\_name,

DOB = p\_dob,

LastModified = CURRENT\_DATE

WHERE CustomerID = p\_customer\_id;

END;

$$;

CREATE OR REPLACE FUNCTION customer\_management\_get\_balance(

p\_customer\_id INT

) RETURNS NUMERIC AS $$

DECLARE

v\_balance NUMERIC;

BEGIN

SELECT Balance INTO v\_balance

FROM Customers

WHERE CustomerID = p\_customer\_id;

RETURN v\_balance;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE PROCEDURE employee\_management\_hire\_employee(

p\_name VARCHAR,

p\_position VARCHAR,

p\_salary NUMERIC,

p\_department VARCHAR

)

LANGUAGE plpgsql AS $$

BEGIN

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

VALUES (p\_name, p\_position, p\_salary, p\_department, CURRENT\_DATE);

END;

$$;

CREATE OR REPLACE PROCEDURE employee\_management\_update\_employee(

p\_employee\_id INT,

p\_name VARCHAR,

p\_position VARCHAR

)

LANGUAGE plpgsql AS $$

BEGIN

UPDATE Employees

SET Name = p\_name,

Position = p\_position

WHERE EmployeeID = p\_employee\_id;

END;

$$;

CREATE OR REPLACE FUNCTION employee\_management\_annual\_salary(

p\_employee\_id INT

) RETURNS NUMERIC AS $$

DECLARE

v\_salary NUMERIC;

BEGIN

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

RETURN v\_salary \* 12;

END;

$$ LANGUAGE plpgsql;

CREATE OR REPLACE PROCEDURE account\_operations\_open\_account(

p\_customer\_id INT,

p\_account\_type VARCHAR,

p\_balance NUMERIC

)

LANGUAGE plpgsql AS $$

BEGIN

INSERT INTO Accounts(CustomerID, AccountType, Balance, LastModified)

VALUES (p\_customer\_id, p\_account\_type, p\_balance, CURRENT\_DATE);

END;

$$;

CREATE OR REPLACE PROCEDURE account\_operations\_close\_account(

p\_account\_id INT

)

LANGUAGE plpgsql AS $$

BEGIN

DELETE FROM Accounts WHERE AccountID = p\_account\_id;

END;

$$;

CREATE OR REPLACE FUNCTION account\_operations\_total\_balance(

p\_customer\_id INT

) RETURNS NUMERIC AS $$

DECLARE

v\_total NUMERIC;

BEGIN

SELECT COALESCE(SUM(Balance), 0)

INTO v\_total

FROM Accounts

WHERE CustomerID = p\_customer\_id;

RETURN v\_total;

END;

$$ LANGUAGE plpgsql;

CALL customer\_management\_add\_customer('Revathy', '1990-01-01', 3000);

CALL employee\_management\_hire\_employee('Dev', 'Developer', 60000, 'IT');

CALL account\_operations\_open\_account(2, 'Checking', 4000);

SELECT \* FROM Customers;

SELECT \* FROM Employees;

SELECT \* FROM Accounts;

SELECT customer\_management\_get\_balance(1) AS Customer1\_Balance;

SELECT employee\_management\_annual\_salary(2) AS Employee2\_AnnualSalary;

SELECT account\_operations\_total\_balance(1) AS Customer1\_TotalBalance;