drop database TestDB

drop table Products1

drop table Customers1

drop table Transactions1

drop procedure GetCitiesWithCustomerCountGreaterThan

drop procedure DeleteTransaction

drop trigger AfterTransactionDelete-- Create Database

CREATE DATABASE TestDB;

GO

-- Use the Database

USE TestDB;

GO

-- Create Products Table

CREATE TABLE Products1 (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100) NOT NULL,

Price DECIMAL(10,2) NOT NULL,

Quantity INT DEFAULT 0 -- Added Quantity column

);

GO

-- Create Customers Table

CREATE TABLE Customers1 (

CustomerID INT PRIMARY KEY,

Name VARCHAR(100) NOT NULL,

City VARCHAR(100) NOT NULL

);

GO

-- Create Transactions Table

CREATE TABLE Transactions1 (

TransactionID INT PRIMARY KEY,

CustomerID INT,

ProductID INT,

Quantity INT CHECK (Quantity > 0),

FOREIGN KEY (CustomerID) REFERENCES Customers1(CustomerID) ON DELETE CASCADE,

FOREIGN KEY (ProductID) REFERENCES Products1(ProductID) ON DELETE CASCADE

);

GO

-- Insert values into Products Table

INSERT INTO Products1 (ProductID, ProductName, Price, Quantity)

VALUES

(1, 'Laptop', 800.50, 10),

(2, 'Smartphone', 500.99, 15),

(3, 'Headphones', 79.99, 20);

GO

-- Insert values into Customers Table

INSERT INTO Customers1 (CustomerID, Name, City)

VALUES

(1, 'Alice Johnson', 'New York'),

(2, 'Bob Smith', 'Los Angeles'),

(3, 'Charlie Brown', 'Chicago');

GO

-- Insert values into Transactions Table

INSERT INTO Transactions1 (TransactionID, CustomerID, ProductID, Quantity)

VALUES

(1, 1, 1, 2), -- Alice buys 2 Laptops

(2, 2, 2, 1), -- Bob buys 1 Smartphone

(3, 3, 3, 3); -- Charlie buys 3 Headphones

GO

-- Create Procedure to Get Cities with More than X Customers

CREATE PROCEDURE GetCitiesWithCustomerCountGreaterThan

@CustomerCount INT

AS

BEGIN

SELECT City, COUNT(\*) AS TotalCustomers

FROM Customers1

GROUP BY City

HAVING COUNT(\*) > @CustomerCount;

END;

GO

-- Create Procedure to Delete a Transaction

CREATE PROCEDURE DeleteTransaction

@TransactionID INT

AS

BEGIN

DELETE FROM Transactions1

WHERE TransactionID = @TransactionID;

END;

GO

-- Create Trigger to Handle Actions After Deleting a Transaction

CREATE TRIGGER AfterTransactionDelete

ON Transactions1

AFTER DELETE

AS

BEGIN

-- Ensure Products1 has a Quantity column before updating

IF EXISTS (SELECT 1 FROM INFORMATION\_SCHEMA.COLUMNS WHERE TABLE\_NAME = 'Products1' AND COLUMN\_NAME = 'Quantity')

BEGIN

UPDATE p

SET p.Quantity = p.Quantity + d.Quantity

FROM Products1 p

INNER JOIN deleted d ON p.ProductID = d.ProductID;

END

END;

GO

-- Execute Procedures and Statements

EXEC GetCitiesWithCustomerCountGreaterThan @CustomerCount = 0;

EXEC DeleteTransaction @TransactionID = 1;

-- Corrected DELETE statement

DELETE FROM Transactions1 WHERE TransactionID = 2;

GO

select \*from Customers1

select \*from Products1

select \*from Transactions1