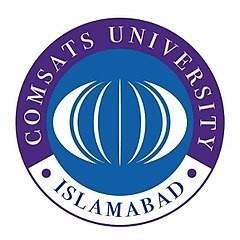
**COMSATS University Islamabad, Lahore Campus**

**Block–B, Department of Computer Engineering**

**COMSATS University Islamabad, Lahore Campus 1.5KM Defence Road, Off Raiwind Road, Lahore**

Assignment

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Course Title: | Database Systems | | | | Course Code: | CSC270 | Credit Hours: | 4(3,***1***) |
| Course Instructor/s: | Modassir Ishfaq | | | | Programme Name: | BCE | | |
| Semester: | 4th | Batch: | Fa23 | Section: | A&B | Date: |  | |
| **Time Allowed:** | **Within 3 days** | | | | **Maximum Marks:** | | **10** | |
| Student’s Name: | Ali Hussain Sindhu | | | | Reg. No. | FA23-BCE-013 | | |
|  | | | | | | | | |

Question:

You are managing the backend of an online store. Customers place orders, and the system must deduct inventory and record the transaction safely. The following tables are in use:

*CREATE TABLE Products ( ProductID INT PRIMARY KEY,*

*ProductName VARCHAR(100), Stock INT,*

*Price DECIMAL(10,2), flag BOOL*

*);*

*CREATE TABLE* *Customer ( customerID INT PRIMARY KEY, CustomerName VARCHAR(100), Balance INT,*

*DateofJoining DATETIME*

*);*

*CREATE TABLE Orders (*

*OrderID INT IDENTITY(1,1) PRIMARY KEY,*

*ProductID INT, CustomerID INT, Quantity INT,*

*TotalAmount DECIMAL(12,2),*

*OrderDate DATETIME DEFAULT GETDATE()*

*);*

Write a script for SQL Transaction to handle customers’ orders ensuring the following:

* Product stock validation (enough quantity available to meet the order requirements)
* Customer balance validation (enough balance to process the whole order)
* Products get flagged when the stock is less than 10. Such items will not be available for newer customer (base this on DateofJoining in customer’s table)
* Customers who joined 5 years or more ago can order low stock items with a limit of 2 per order.
* Any situation where order is not processable will result in the rollback of the whole transaction and generate relevant prompt.
* On success of order placement, all tables must be updated accordingly.

**Solution**

CREATE OR ALTER PROCEDURE sp\_ProcessOrder

@custID INT,

@prodID INT,

@qty INT

AS

BEGIN

DECLARE

@available INT,

@unitPrice DECIMAL(10,2),

@custBalance INT,

@orderTotal DECIMAL(12,2),

@joinDate DATETIME,

@currentDate DATETIME = GETDATE();

BEGIN TRY

BEGIN TRANSACTION;

-- Product stock validation

SELECT @available = Stock, @unitPrice = Price

FROM Products

WHERE ProductID = @prodID;

IF @available IS NULL

BEGIN

RAISERROR('Item not found in catalog.', 16, 1);

ROLLBACK;

RETURN;

END

IF @available < @qty

BEGIN

RAISERROR('Insufficient stock for the requested item.', 16, 1);

ROLLBACK;

RETURN;

END

-- Customer balance and eligibility check

SELECT @custBalance = Balance, @joinDate = DateOfJoining

FROM Customer

WHERE CustomerID = @custID;

IF @custBalance IS NULL

BEGIN

RAISERROR('Customer ID not registered.', 16, 1);

ROLLBACK;

RETURN;

END

SET @orderTotal = @unitPrice \* @qty;

IF @custBalance < @orderTotal

BEGIN

RAISERROR('Not enough funds in the account.', 16, 1);

ROLLBACK;

RETURN;

END

-- Low stock restriction based on customer age

IF @available < 10

BEGIN

IF DATEDIFF(YEAR, @joinDate, @currentDate) < 5

BEGIN

RAISERROR('Low-stock items are not purchasable by new customers.', 16, 1);

ROLLBACK;

RETURN;

END

ELSE IF @qty > 2

BEGIN

RAISERROR('Old customers can only purchase up to 2 low-stock units.', 16, 1);

ROLLBACK;

RETURN;

END

END

-- All validations passed, place order

INSERT INTO Orders (ProductID, CustomerID, Quantity, TotalAmount)

VALUES (@prodID, @custID, @qty, @orderTotal);

UPDATE Products

SET Stock = Stock - @qty

WHERE ProductID = @prodID;

UPDATE Customer

SET Balance = Balance - @orderTotal

WHERE CustomerID = @custID;

UPDATE Products

SET flag = 1

WHERE ProductID = @prodID AND Stock < 10;

COMMIT;

PRINT 'Transaction completed: Order successful.';

END TRY

BEGIN CATCH

ROLLBACK;

RAISERROR(ERROR\_MESSAGE(), 16, 1);

END CATCH

END;