**《数据库系统原理》实验报告**

### **实验题目：**实验18 数据库设计

**姓名：**

**实验日期：**

**实验内容及完成情况：**（可续页）

### 实验18 数据库设计

**实验学时：4学时**

**实验类型：设计**

**实验要求：必做**

**一、实验目的**

1、熟悉数据库应用系统需求分析；

2、熟悉数据库设计的基本方法；

3、掌握数据库和数据表的创建方法；

4、掌握视图的创建方法；

5、了解各种数据完整性约束；

6、对存储过程和触发器有一个完整的认识。

**二、实验要求**

给出网上书店管理系统的需求分析报告（见四），要求根据功能分析和数据分析设计概念模型，实现完整的数据库（包括：各种表，视图，必要的存储过程，触发器等）。

**三、实验内容和步骤**

1、表结构设计：设计文档要求说明每一张表以及每一个字段的含义，说明每个表的主键、外键、约束等；

2、建立数据库；

3、建立表及必要的约束；

4、建立必要的视图；

5、建立必要的存储过程和触发器。

**四、相关资料**

**网上书店管理系统的需求分析报告**

设计一个小型的数据库应用系统，服务于一个网上书店的日常处理业务及网上购书业务，包括供书目录、库存管理、采购管理、客户管理，顾客订单管理、发货管理，供应商管理及网上浏览等。

总体设计要求：书店内部日常处理业务通过C/S模式来实现，而顾客可通过Internet进行网上订书及查询浏览，为改善性能，应尽量将业务规则放在中间层或数据库服务器一端。

该系统的主要功能如下：

1、供书目录及库存管理

供书目录管理及库存管理为书店的主要日常业务，应包括建立和更新供书目录，新书入库功能等。需要记录供书的信息有：书号，书名，作者（最多可包括四个作者，有序），出版社，价格，关键字（最多10个），本书的目录（可选），封皮（可选），存货量，供书商（一本书可有多个供应商，可选）等，并且需考虑丛书的问题（即一个书号包含多本书），库存存放位置（可采用示意图表示，可选）。

2、采购管理

采购管理为书店的主要日常业务之一，包括缺书登记管理及采购单管理。

缺书登记可通过以下几种方法生成：

1）直接进行缺书登记，包括书号，书名，出版社，供书商，数量，登记日期。

2）存书量低于一定限度时，自动生成缺书记录（注：不能有重复）,最低存书量可设定。（可选）

3）顾客在网上进行缺书登记或订货数量超出库存时，生成缺书记录（注：不能有重复），必要时，应记录下顾客的缺书请求，以便答复。（可选）

采购单管理：根据缺书记录单可选择生成采购单，到货时对采购单作相应处理，并增加库存量，删除缺书记录，EMAIL通知缺书登记的顾客（可选）等。

3、客户管理

客户管理包括客户信息管理及信用管理。需要记录客户的信息包括：网上ID，登录密码，名称，地址，帐户余额，信用等级等。

有关客户信息的管理，分别由书店管理和客户网上管理完成，书店管理包括收到客户的款项后，增加客户的帐户余额；调整信用额度。客户其他信息由客户在网上维护，如注册一个新的用户，修改用户信息等等。

信用等级分为五级，一级为10%的折扣，不能透支；二级为15%的折扣，不能透支；三级为15%的折扣，可先发书再付款，透支有额度限制；四级为20%的折扣，可先发书再付款，透支有额度限制；五级为25%的折扣，可先发书再付款，透支无额度限制。信用等级管理有两种形式：一种是书店进行管理；一种是自动进行管理，即在每月月初根据该帐户的余额或累计购书总金额进行调整，如帐户余额超过一定金额，提升其信用额度，或者累计购书总金额超过一定金额提升其信用额度（可选）。

4、顾客订单管理和发货管理

顾客订单包括订单号，订货日期，客户ID，书号，订书数量，金额，发货地址，发货情况等。

订单由顾客在网上申请生成。一个订单可订多本书（可选）。并仅能对书库中已有的书目进行订购，库存量不足时可以先订货；当客户查询的书目在书库中没有找到时，将会询问客户是否希望做进一步的询价及报价。

发货管理为书店日常业务管理之一，必须根据订单情况及客户的信用等级判断其付款额是否到帐，并相应扣减其帐户余额后，才能发货。对一个订单可分次发货（可选）

5、供应商管理

供应商相关信息，包括：供应商基本信息，供应商供货信息，并且不同的供应商发布自己现有书目信息。

6、网上浏览查询

可查询以下二类信息：

1）客户相关信息，包括：客户基本信息，客户历史订单信息，相关订单的发货信息等。

2）书目信息：可按照书号，书名，出版社，关键字（可指定匹配程度（可选））、作者（按第一，第二……作者查询（可选）），进行查询，并可进行模糊查询（可选）。

要求：必须根据所有必选的功能和数据需求设计实现完整的数据库（包括：各种表，视图，必要的存储过程，触发器等）。

**五、课后习题**

1.试选择一个应用，练习数据库设计。

### ****1. 表结构设计****

设计的表包括供书目录、库存、采购管理、客户管理、订单管理、供应商管理、及相关视图。以下是表的概要说明

1. Books (供书目录及库存表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 书号 | BookID | INT | 是 | 是 | 书籍的唯一标识 |
| 2 | 书名 | Title | VARCHAR(255) | 是 | 否 | 书籍名称 |
| 3 | 作者 | Author | JSON | 是 | 否 | 作者列表（JSON 格式） |
| 4 | 出版社 | Publisher | VARCHAR(255) | 是 | 否 | 出版社名称 |
| 5 | 单价 | Price | DECIMAL(10,2) | 是 | 否 | 书籍单价 |
| 6 | 关键字 | Keywords | JSON | 否 | 否 | 关键字列表（JSON 格式） |
| 7 | 目录 | Catalog | TEXT | 否 | 否 | 书籍的目录 |
| 8 | 封面图片 | CoverImage | TEXT | 否 | 否 | 封面图片路径 |
| 9 | 库存量 | Stock | INT | 是 | 否 | 当前库存数量 |
| 10 | 存储位置 | StorageLocation | TEXT | 否 | 否 | 库存位置描述 |

2. BookSuppliers (书籍供应商关联表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 供应商ID | SupplierID | INT | 是 | 是 | 供应商的唯一标识 |
| 2 | 书号 | BookID | INT | 是 | 是 | 书籍的唯一标识 |
| 3 | 供货价格 | SupplyPrice | DECIMAL(10,2) | 是 | 否 | 供应商提供的单价 |

3. PurchaseRequests (缺书登记表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 缺书记录ID | RequestID | INT | 是 | 是 | 缺书记录的唯一标识 |
| 2 | 书号 | BookID | INT | 是 | 否 | 书籍的唯一标识 |
| 3 | 供应商ID | SupplierID | INT | 否 | 否 | 建议供应商的唯一标识 |
| 4 | 请求日期 | RequestDate | DATE | 是 | 否 | 缺书请求的登记日期 |
| 5 | 缺书数量 | Quantity | INT | 是 | 否 | 缺书的数量 |

1. PurchaseOrders (采购订单表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 订单ID | OrderID | INT | 是 | 是 | 采购订单的唯一标识 |
| 2 | 缺书记录ID | RequestID | INT | 是 | 否 | 缺书记录的唯一标识 |
| 3 | 下单日期 | OrderDate | DATE | 是 | 否 | 采购订单的下单日期 |
| 4 | 到货日期 | ArrivalDate | DATE | 否 | 否 | 到货日期 |
| 5 | 状态 | Status | ENUM | 是 | 否 | 采购订单状态（如已到货） |

1. Customers (客户信息表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 客户ID | CustomerID | INT | 是 | 是 | 客户的唯一标识 |
| 2 | 登录密码 | Password | VARCHAR(255) | 是 | 否 | 客户的登录密码 |
| 3 | 姓名 | Name | VARCHAR(255) | 是 | 否 | 客户姓名 |
| 4 | 地址 | Address | VARCHAR(255) | 是 | 否 | 客户的地址 |
| 5 | 帐户余额 | Balance | DECIMAL(10,2) | 是 | 否 | 客户的账户余额 |
| 6 | 信用等级 | CreditLevel | INT | 是 | 否 | 客户的信用等级 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 信用等级 | Level | INT | 是 | 是 | 信用等级 |
| 2 | 折扣率 | DiscountRate | DECIMAL(5,2) | 是 | 否 | 信用等级对应的折扣率 |
| 3 | 透支额度 | OverdraftLimit | DECIMAL(10,2) | 否 | 否 | 信用等级对应的透支额度 |

1. CreditRules (信用规则表)

7. Orders (订单表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 订单ID | OrderID | INT | 是 | 是 | 订单的唯一标识 |
| 2 | 客户ID | CustomerID | INT | 是 | 否 | 客户的唯一标识 |
| 3 | 下单日期 | OrderDate | DATE | 是 | 否 | 订单的创建日期 |
| 4 | 发货地址 | ShippingAddress | VARCHAR(255) | 是 | 否 | 订单的发货地址 |
| 5 | 总金额 | TotalAmount | DECIMAL(10,2) | 是 | 否 | 订单的总金额 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 订单明细ID | OrderDetailID | INT | 是 | 是 | 订单明细的唯一标识 |
| 2 | 订单ID | OrderID | INT | 是 | 否 | 对应订单的唯一标识 |
| 3 | 书号 | BookID | INT | 是 | 否 | 书籍的唯一标识 |
| 4 | 数量 | Quantity | INT | 是 | 否 | 书籍订购数量 |

8.OrderDetails (订单明细表)

9.Suppliers (供应商信息表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 供应商ID | SupplierID | INT | 是 | 是 | 供应商的唯一标识 |
| 2 | 名称 | Name | VARCHAR(255) | 是 | 否 | 供应商名称 |
| 3 | 联系信息 | ContactInfo | VARCHAR(255) | 是 | 否 | 供应商的联系方式 |

1. Series (丛书表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 丛书ID | SeriesID | INT | 是 | 是 | 丛书的唯一标识 |
| 2 | 丛书名称 | SeriesName | VARCHAR(255) | 是 | 否 | 丛书的名称 |
| 3 | 描述 | Description | TEXT | 否 | 否 | 丛书的描述信息 |

1. MissingRequests (顾客缺书请求表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 请求ID | RequestID | INT | 是 | 是 | 缺书请求的唯一标识 |
| 2 | 客户ID | CustomerID | INT | 是 | 否 | 顾客的唯一标识 |
| 3 | 书号 | BookID | INT | 是 | 否 | 缺书的书号 |
| 4 | 请求日期 | RequestDate | DATE | 是 | 否 | 请求的日期 |
| 5 | 状态 | Status | ENUM | 是 | 否 | 请求状态（如待处理、已答复） |

1. Shipping (发货表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **数据项名** | **列名** | **数据类型** | **非空** | **主键** | **含义** |
| 1 | 发货ID | ShippingID | INT | 是 | 是 | 发货的唯一标识 |
| 2 | 订单ID | OrderID | INT | 是 | 否 | 对应订单的唯一标识 |
| 3 | 发货时间 | ShippingDate | DATE | 是 | 否 | 发货日期 |
| 4 | 状态 | Status | ENUM | 是 | 否 | 发货状态（如已发货、部分发货） |

**2.表建立**

1. Books 表

CREATE TABLE Books (

BookID INT PRIMARY KEY, -- 主键

Title VARCHAR(255) NOT NULL, -- 书名

Author JSON NOT NULL, -- 作者列表，JSON格式

Publisher VARCHAR(255) NOT NULL, -- 出版社

Price DECIMAL(10, 2) NOT NULL CHECK (Price >= 0), -- 单价，非负

Keywords JSON, -- 关键字列表，JSON格式

Catalog TEXT, -- 目录，可选

CoverImage TEXT, -- 封面图片路径，可选

Stock INT NOT NULL CHECK (Stock >= 0), -- 库存，非负

StorageLocation TEXT -- 存储位置，可选

);

1. BookSuppliers 表

CREATE TABLE BookSuppliers (

SupplierID INT, -- 外键，供应商ID

BookID INT, -- 外键，书号

SupplyPrice DECIMAL(10, 2) NOT NULL CHECK (SupplyPrice > 0), -- 供货价格，正数

PRIMARY KEY (SupplierID, BookID), -- 组合主键

FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID), -- 供应商外键

FOREIGN KEY (BookID) REFERENCES Books(BookID) -- 书号外键

);

1. PurchaseRequests 表

CREATE TABLE PurchaseRequests (

RequestID INT PRIMARY KEY, -- 主键，缺书记录ID

BookID INT NOT NULL, -- 外键，书号

SupplierID INT, -- 外键，供应商ID，可选

RequestDate DATE NOT NULL, -- 请求日期

Quantity INT NOT NULL CHECK (Quantity > 0), -- 缺书数量，正数

FOREIGN KEY (BookID) REFERENCES Books(BookID), -- 书号外键

FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID) -- 供应商外键

);

4. PurchaseOrders 表

CREATE TABLE PurchaseOrders (

OrderID INT PRIMARY KEY, -- 主键，采购订单ID

RequestID INT NOT NULL, -- 外键，缺书记录ID

OrderDate DATE NOT NULL, -- 下单日期

ArrivalDate DATE, -- 到货日期，可选

Status ENUM('Pending', 'Ordered', 'Arrived') NOT NULL, -- 状态

FOREIGN KEY (RequestID) REFERENCES PurchaseRequests(RequestID) -- 缺书记录外键

);

5.Customers 表

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY, -- 主键，客户ID

Password VARCHAR(255) NOT NULL, -- 登录密码

Name VARCHAR(255) NOT NULL, -- 客户姓名

Address VARCHAR(255) NOT NULL, -- 客户地址

Balance DECIMAL(10, 2) NOT NULL CHECK (Balance >= 0), -- 帐户余额，非负

CreditLevel INT NOT NULL CHECK (CreditLevel BETWEEN 1 AND 5) -- 信用等级，1-5

);

6.CreditRules 表

CREATE TABLE CreditRules (

Level INT PRIMARY KEY, -- 主键，信用等级

DiscountRate DECIMAL(5, 2) NOT NULL CHECK (DiscountRate BETWEEN 0 AND 1), -- 折扣率，0-1之间

OverdraftLimit DECIMAL(10, 2) -- 透支额度，可为 NULL

);

7. Orders 表

CREATE TABLE Orders (

OrderID INT PRIMARY KEY, -- 主键，订单ID

CustomerID INT NOT NULL, -- 外键，客户ID

OrderDate DATE NOT NULL, -- 下单日期

ShippingAddress VARCHAR(255) NOT NULL, -- 发货地址

TotalAmount DECIMAL(10, 2) NOT NULL CHECK (TotalAmount >= 0), -- 总金额，非负

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID) -- 客户外键

);

8. OrderDetails 表

CREATE TABLE OrderDetails (

OrderDetailID INT PRIMARY KEY, -- 主键，订单明细ID

OrderID INT NOT NULL, -- 外键，订单ID

BookID INT NOT NULL, -- 外键，书号

Quantity INT NOT NULL CHECK (Quantity > 0), -- 订购数量，正数

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID), -- 订单外键

FOREIGN KEY (BookID) REFERENCES Books(BookID) -- 书号外键

);

9. Suppliers 表

CREATE TABLE Suppliers (

SupplierID INT PRIMARY KEY, -- 主键，供应商ID

Name VARCHAR(255) NOT NULL, -- 供应商名称

ContactInfo VARCHAR(255) NOT NULL -- 联系信息

);

1. Series (丛书表)

CREATE TABLE Series (

SeriesID INT PRIMARY KEY, -- 丛书ID

SeriesName VARCHAR(255) NOT NULL, -- 丛书名称

Description TEXT, -- 丛书描述（可选）

Stock INT NOT NULL DEFAULT 0 CHECK (Stock >= 0) -- 丛书总库存，非负

);

1. MissingRequests (顾客缺书请求表)

CREATE TABLE MissingRequests (

RequestID INT PRIMARY KEY, -- 请求ID

CustomerID INT NOT NULL, -- 顾客ID

BookID INT NOT NULL, -- 缺书书号

RequestDate DATE NOT NULL, -- 请求日期

Status ENUM('Pending', 'Processed', 'Rejected') NOT NULL DEFAULT 'Pending', -- 请求状态

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID) -- 外键，关联顾客表

ON DELETE CASCADE, -- 删除顾客时级联删除请求

FOREIGN KEY (BookID) REFERENCES Books(BookID) -- 外键，关联书籍表

ON DELETE CASCADE -- 删除书籍时级联删除请求

);

1. Shipping (发货表)

CREATE TABLE Shipping (

ShippingID INT PRIMARY KEY, -- 发货ID

OrderID INT NOT NULL, -- 订单ID

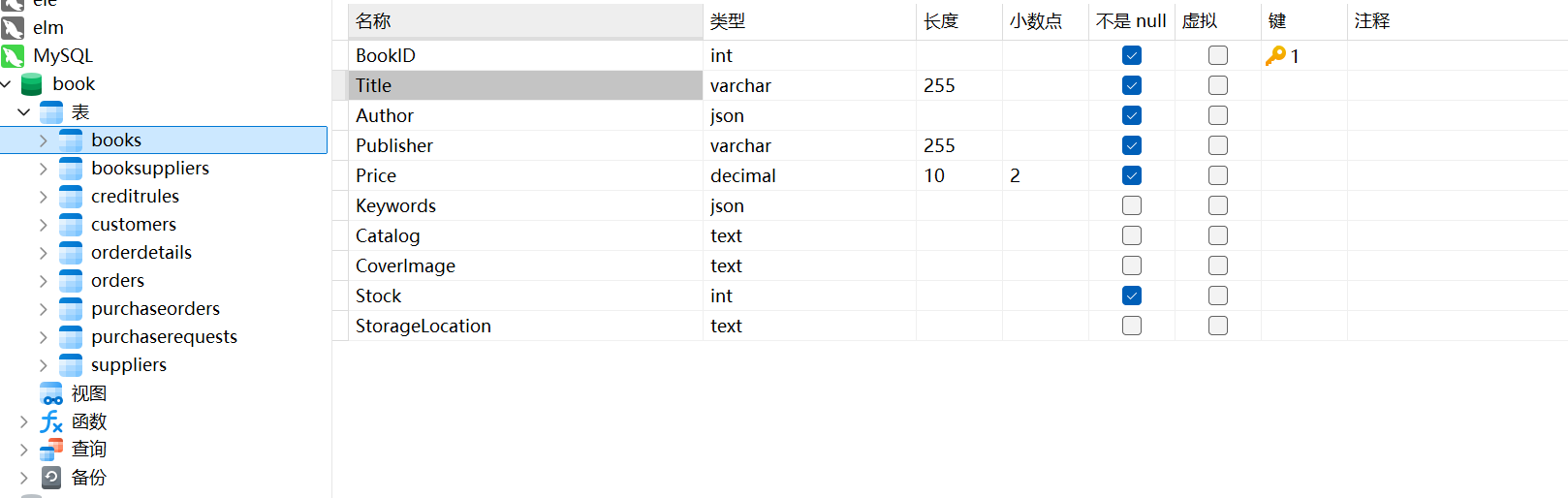
ShippingDate DATE NOT NULL, -- 发货日期

Status ENUM('Pending', 'Partially Shipped', 'Shipped') NOT NULL DEFAULT 'Pending', -- 发货状态

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID) -- 外键，关联订单表

ON DELETE CASCADE -- 删除订单时级联删除发货记录

);



3.触发器设计

### **1. 库存更新时自动生成缺书记录**

**功能说明**：当书籍库存量低于设定的最小库存时，自动生成缺书记录。

CREATE TRIGGER Trigger\_AutoGenerateMissingRecord

AFTER UPDATE ON Books

FOR EACH ROW

BEGIN

IF NEW.Stock < 10 THEN

INSERT INTO PurchaseRequests (BookID, RequestDate, Quantity)

VALUES (NEW.BookID, NOW(), 10)

ON DUPLICATE KEY UPDATE RequestDate = NOW();

END IF;

END;

**解释**：

* 在 Books 表中更新库存时触发。
* 如果库存量低于10，则在 PurchaseRequests 表中生成缺书记录。

### **2. 触发器：采购订单到货后更新库存**

**功能说明**：采购订单到货时自动更新库存量并删除缺书记录。

CREATE TRIGGER Trigger\_UpdateStockOnArrival

AFTER UPDATE ON PurchaseOrders

FOR EACH ROW

BEGIN

IF NEW.Status = 'Arrived' THEN

UPDATE Books

SET Stock = Stock + (

SELECT Quantity

FROM PurchaseRequests

WHERE RequestID = NEW.RequestID

)

WHERE BookID = (

SELECT BookID

FROM PurchaseRequests

WHERE RequestID = NEW.RequestID

);

DELETE FROM PurchaseRequests

WHERE RequestID = NEW.RequestID;

END IF;

END;

**解释**：

* 在 PurchaseOrders 表中状态更新为 Arrived 时触发。
* 更新 Books 表中的库存量，并删除对应的 PurchaseRequests 记录。

### **3. 触发器：客户订单扣减余额并判断信用等级**

**功能说明**：当订单发货时扣减客户余额，并根据余额更新信用等级。

CREATE TRIGGER Trigger\_DeductBalanceOnOrder

AFTER UPDATE ON Orders

FOR EACH ROW

BEGIN

IF NEW.TotalAmount <= (

SELECT Balance FROM Customers WHERE CustomerID = NEW.CustomerID

) THEN

UPDATE Customers

SET Balance = Balance - NEW.TotalAmount

WHERE CustomerID = NEW.CustomerID;

-- 自动更新信用等级（可选功能）

IF (SELECT Balance FROM Customers WHERE CustomerID = NEW.CustomerID) >= 1000 THEN

UPDATE Customers

SET CreditLevel = GREATEST(CreditLevel + 1, 5)

WHERE CustomerID = NEW.CustomerID;

END IF;

ELSE

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Insufficient balance for this order.';

END IF;

END;

**解释**：

* 在订单发货时触发。
* 如果余额足够，扣减客户账户余额。
* 可选功能：余额超过一定值时自动提升信用等级。

### **4. 触发器：删除客户时清理相关订单**

**功能说明**：当客户被删除时，自动删除其相关订单及订单明细。

CREATE TRIGGER Trigger\_CascadeDeleteCustomer

BEFORE DELETE ON Customers

FOR EACH ROW

BEGIN

DELETE FROM OrderDetails

WHERE OrderID IN (

SELECT OrderID FROM Orders WHERE CustomerID = OLD.CustomerID

);

DELETE FROM Orders

WHERE CustomerID = OLD.CustomerID;

END;

**解释**：

* 在 Customers 表删除记录时触发。
* 删除相关联的订单明细和订单记录。

### **5. 触发器：自动调整信用等级**

**功能说明**：每月初根据累计消费或账户余额自动调整客户信用等级。

CREATE TRIGGER Trigger\_AutoUpdateCreditLevel

BEFORE UPDATE ON Customers

FOR EACH ROW

BEGIN

IF NEW.Balance >= 1000 THEN

SET NEW.CreditLevel = LEAST(NEW.CreditLevel + 1, 5);

ELSEIF NEW.Balance < 500 THEN

SET NEW.CreditLevel = GREATEST(NEW.CreditLevel - 1, 1);

END IF;

END;

**解释**：

* 在客户余额更新后触发。
* 根据余额提高或降低信用等级。

### **6. 触发器：防止库存量为负值**

**功能说明**：当尝试减少库存时，阻止库存量变为负值

CREATE TRIGGER Trigger\_PreventNegativeStock

BEFORE UPDATE ON Books

FOR EACH ROW

BEGIN

IF NEW.Stock < 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Stock cannot be negative.';

END IF;

END;

**解释**：

* 在 Books 表更新时触发。
* 如果库存量更新后为负，抛出错误。

### **7.触发器：自动更新丛书库存**

**功能**：当 Books 表的库存更新时，同时更新其关联的丛书库存

CREATE TRIGGER Trigger\_UpdateSeriesStock

AFTER UPDATE ON Books

FOR EACH ROW

BEGIN

IF NEW.Stock <> OLD.Stock THEN

UPDATE Series

SET Stock = (SELECT SUM(Stock) FROM Books WHERE SeriesID = NEW.SeriesID)

WHERE SeriesID = NEW.SeriesID;

END IF;

END;

### **8.触发器：顾客缺书请求的自动处理**

**功能**：当缺书请求被处理后自动将状态更新为 "已答复"。

CREATE TRIGGER Trigger\_ProcessMissingRequest

AFTER INSERT ON PurchaseOrders

FOR EACH ROW

BEGIN

UPDATE MissingRequests

SET Status = 'Processed'

WHERE BookID = (

SELECT BookID FROM PurchaseRequests WHERE RequestID = NEW.RequestID

) AND Status = 'Pending';

END;

### **9.触发器：发货时检查库存和信用**

**功能**：在订单发货时检查库存和顾客信用等级，确保满足发货条件。

CREATE TRIGGER Trigger\_CheckShippingBeforeUpdate

BEFORE INSERT ON Shipping

FOR EACH ROW

BEGIN

DECLARE CurrentStock INT;

DECLARE CreditLevel INT;

DECLARE TotalRequired INT;

-- 获取库存

SELECT SUM(Stock) INTO CurrentStock

FROM Books

WHERE BookID IN (SELECT BookID FROM OrderDetails WHERE OrderID = NEW.OrderID);

-- 获取信用等级

SELECT CreditLevel INTO CreditLevel

FROM Customers

WHERE CustomerID = (SELECT CustomerID FROM Orders WHERE OrderID = NEW.OrderID);

-- 计算订单总需求

SELECT SUM(Quantity) INTO TotalRequired

FROM OrderDetails

WHERE OrderID = NEW.OrderID;

-- 检查库存

IF CurrentStock < TotalRequired THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Not enough stock for shipping.';

END IF;

-- 检查信用等级

IF CreditLevel < 3 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Insufficient credit level for pre-shipping.';

END IF;

END;

### **10. 触发器:自动生成采购单**

**功能**：当缺书记录生成时自动生成采购单。

CREATE TRIGGER Trigger\_AutoGeneratePurchaseOrder

AFTER INSERT ON PurchaseRequests

FOR EACH ROW

BEGIN

INSERT INTO PurchaseOrders (RequestID, OrderDate, Status)

VALUES (NEW.RequestID, NOW(), 'Pending');

END;

### **11. 触发器：顾客累计消费更新信用等级**

**功能**：根据顾客的累计消费金额提升信用等级。

CREATE TRIGGER Trigger\_UpdateCreditLevelOnTotalAmount

AFTER UPDATE ON Orders

FOR EACH ROW

BEGIN

DECLARE TotalSpent DECIMAL(10, 2);

-- 计算累计消费金额

SELECT SUM(TotalAmount) INTO TotalSpent

FROM Orders

WHERE CustomerID = NEW.CustomerID;

-- 更新信用等级

IF TotalSpent >= 5000 THEN

UPDATE Customers

SET CreditLevel = LEAST(CreditLevel + 1, 5)

WHERE CustomerID = NEW.CustomerID;

END IF;

END;

4.视图设计

### **1. 供书目录视图**

显示书籍的基本信息和库存状态，包括关联的丛书信息。

CREATE VIEW View\_BookInventory AS

SELECT

b.BookID,

b.Title,

b.Author,

b.Publisher,

b.Price,

b.Stock,

s.SeriesName AS Series,

CASE

WHEN b.Stock = 0 THEN 'Out of Stock'

WHEN b.Stock < 10 THEN 'Low Stock'

ELSE 'In Stock'

END AS StockStatus

FROM

Books b

LEFT JOIN

Series s ON b.SeriesID = s.SeriesID;

**用途**：

* 提供书籍的基本库存状态，便于管理人员查看哪些书籍需要补货。

### **2. 缺书信息视图**

显示缺书登记的详细信息，包括顾客缺书请求和书籍的库存状态。

CREATE VIEW View\_MissingBooks AS

SELECT

pr.RequestID,

b.BookID,

b.Title,

b.Stock,

pr.Quantity AS RequestedQuantity,

c.Name AS CustomerName,

pr.RequestDate,

pr.Status

FROM

PurchaseRequests pr

LEFT JOIN

Books b ON pr.BookID = b.BookID

LEFT JOIN

Customers c ON pr.CustomerID = c.CustomerID

WHERE

pr.Status = 'Pending';

**用途**：

* 管理人员查看缺书记录及顾客的缺书请求，方便采购管理。

### **3. 客户订单视图**

显示客户的订单及发货状态。

CREATE VIEW View\_CustomerOrders AS

SELECT

o.OrderID,

o.OrderDate,

c.CustomerID,

c.Name AS CustomerName,

od.BookID,

b.Title AS BookTitle,

od.Quantity,

o.TotalAmount,

s.Status AS ShippingStatus,

s.ShippingDate

FROM

Orders o

JOIN

OrderDetails od ON o.OrderID = od.OrderID

JOIN

Books b ON od.BookID = b.BookID

JOIN

Customers c ON o.CustomerID = c.CustomerID

LEFT JOIN

Shipping s ON o.OrderID = s.OrderID;

**用途**：

* 提供客户订单的详细信息及发货情况，便于查询订单状态。

### **4. 客户消费记录视图**

显示每位客户的累计消费金额和信用等级。

CREATE VIEW View\_CustomerSpending AS

SELECT

c.CustomerID,

c.Name AS CustomerName,

c.CreditLevel,

c.Balance,

IFNULL(SUM(o.TotalAmount), 0) AS TotalSpent

FROM

Customers c

LEFT JOIN

Orders o ON c.CustomerID = o.CustomerID

GROUP BY

c.CustomerID, c.Name, c.CreditLevel, c.Balance;

**用途**：

* 用于管理客户的消费信息，便于信用等级调整和促销活动。

### **5. 订单发货视图**

显示订单的发货详情，包括未发货订单。

CREATE VIEW View\_ShippingStatus AS

SELECT

o.OrderID,

o.OrderDate,

c.Name AS CustomerName,

s.ShippingID,

s.ShippingDate,

s.Status AS ShippingStatus,

o.TotalAmount

FROM

Orders o

LEFT JOIN

Shipping s ON o.OrderID = s.OrderID

JOIN

Customers c ON o.CustomerID = c.CustomerID

WHERE

s.Status IS NULL OR s.Status != 'Shipped';

**用途**：

* 查看未完全发货的订单，便于跟进发货。

### **6. 供应商供货视图**

显示供应商的基本信息及供货书籍。

CREATE VIEW View\_SupplierBooks AS

SELECT

s.SupplierID,

s.Name AS SupplierName,

b.BookID,

b.Title AS BookTitle,

bs.SupplyPrice

FROM

Suppliers s

JOIN

BookSuppliers bs ON s.SupplierID = bs.SupplierID

JOIN

Books b ON bs.BookID = b.BookID;

**用途**：

* 管理人员查看供应商的供货书籍信息，便于供货关系维护。

### **7. 顾客浏览书目视图**

显示书籍信息，支持按关键字、作者或出版社进行模糊查询。

CREATE VIEW View\_BrowseBooks AS

SELECT

b.BookID,

b.Title,

b.Author,

b.Publisher,

b.Price,

b.Keywords,

CASE

WHEN b.Stock = 0 THEN 'Out of Stock'

ELSE 'Available'

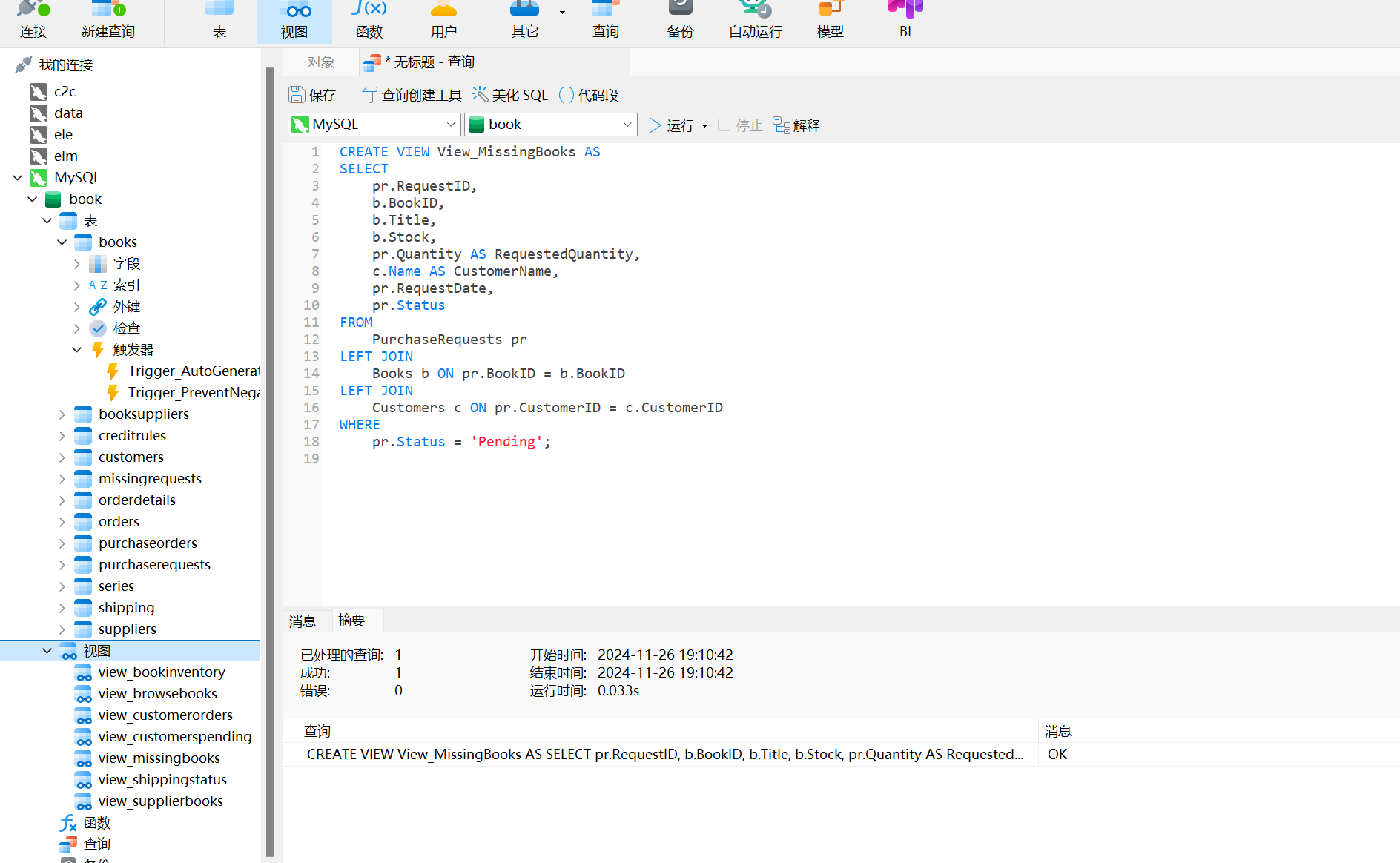
END AS Availability

FROM

Books b;

**用途**：

* 提供顾客查询书籍信息的视图，用于网上浏览。



5.存储过程

### **1. 自动生成缺书记录**

当库存低于设定值时，自动生成缺书记录。

DELIMITER //

CREATE PROCEDURE GenerateMissingRecords()

BEGIN

INSERT INTO PurchaseRequests (BookID, RequestDate, Quantity, Status)

SELECT

BookID,

CURDATE(),

10 AS Quantity,

'Pending' AS Status

FROM

Books

WHERE

Stock < 10

ON DUPLICATE KEY UPDATE

RequestDate = VALUES(RequestDate); -- 防止重复生成缺书记录

END //

DELIMITER ;

**用途**：

* 定期检查库存并生成缺书记录。

### **2. 创建采购订单**

根据缺书记录生成采购订单，并将状态更新为“已处理”。

DELIMITER //

CREATE PROCEDURE CreatePurchaseOrder(IN RequestID INT)

BEGIN

DECLARE BookID INT;

DECLARE Quantity INT;

-- 获取缺书记录信息

SELECT

BookID,

Quantity

INTO

BookID, Quantity

FROM

PurchaseRequests

WHERE

RequestID = RequestID;

-- 创建采购订单

INSERT INTO PurchaseOrders (RequestID, OrderDate, Status)

VALUES (RequestID, CURDATE(), 'Pending');

-- 更新缺书记录状态

UPDATE PurchaseRequests

SET Status = 'Processed'

WHERE RequestID = RequestID;

END //

DELIMITER ;

**用途**：

* 根据特定的缺书记录生成采购订单并更新状态。

### **3. 更新客户信用等级**

根据客户的账户余额或累计消费金额自动调整信用等级。

DELIMITER //

CREATE PROCEDURE UpdateCreditLevel(IN CustomerID INT)

BEGIN

DECLARE TotalSpent DECIMAL(10, 2);

DECLARE CurrentBalance DECIMAL(10, 2);

DECLARE NewCreditLevel INT;

-- 获取累计消费金额和账户余额

SELECT

IFNULL(SUM(TotalAmount), 0),

Balance

INTO

TotalSpent, CurrentBalance

FROM

Orders

JOIN

Customers ON Orders.CustomerID = Customers.CustomerID

WHERE

Customers.CustomerID = CustomerID;

-- 计算新的信用等级

IF TotalSpent >= 5000 OR CurrentBalance >= 1000 THEN

SET NewCreditLevel = LEAST(CreditLevel + 1, 5);

ELSEIF CurrentBalance < 500 THEN

SET NewCreditLevel = GREATEST(CreditLevel - 1, 1);

ELSE

SET NewCreditLevel = CreditLevel;

END IF;

-- 更新信用等级

UPDATE Customers

SET CreditLevel = NewCreditLevel

WHERE CustomerID = CustomerID;

END //

DELIMITER ;

**用途**：

* 按账户余额或消费金额动态调整信用等级。

### **4. 更新库存和删除缺书记录**

采购到货时更新库存，并自动删除相关缺书记录。

DELIMITER //

CREATE PROCEDURE UpdateStockAndClearRequest(IN RequestID INT)

BEGIN

DECLARE BookID INT;

DECLARE Quantity INT;

-- 获取缺书记录信息

SELECT

BookID,

Quantity

INTO

BookID, Quantity

FROM

PurchaseRequests

WHERE

RequestID = RequestID;

-- 更新库存

UPDATE Books

SET Stock = Stock + Quantity

WHERE BookID = BookID;

-- 删除缺书记录

DELETE FROM PurchaseRequests

WHERE RequestID = RequestID;

END //

DELIMITER ;

**用途**：

* 到货后自动更新库存并清理相关缺书记录。

### **5. 查询顾客订单及发货情况**

根据顾客 ID 查询订单及其发货状态。

DELIMITER //

CREATE PROCEDURE GetCustomerOrders(IN CustomerID INT)

BEGIN

SELECT

o.OrderID,

o.OrderDate,

o.TotalAmount,

s.Status AS ShippingStatus,

s.ShippingDate

FROM

Orders o

LEFT JOIN

Shipping s ON o.OrderID = s.OrderID

WHERE

o.CustomerID = CustomerID;

END //

DELIMITER ;

**用途**：

* 提供顾客订单和发货情况查询。

### **6. 自动处理未发货订单**

将所有信用等级满足条件的未发货订单状态更新为“已发货”。

DELIMITER //

CREATE PROCEDURE ProcessPendingOrders()

BEGIN

UPDATE Orders o

SET

o.Status = 'Shipped'

WHERE

o.Status = 'Pending'

AND o.CustomerID IN (

SELECT CustomerID

FROM Customers

WHERE CreditLevel >= 3

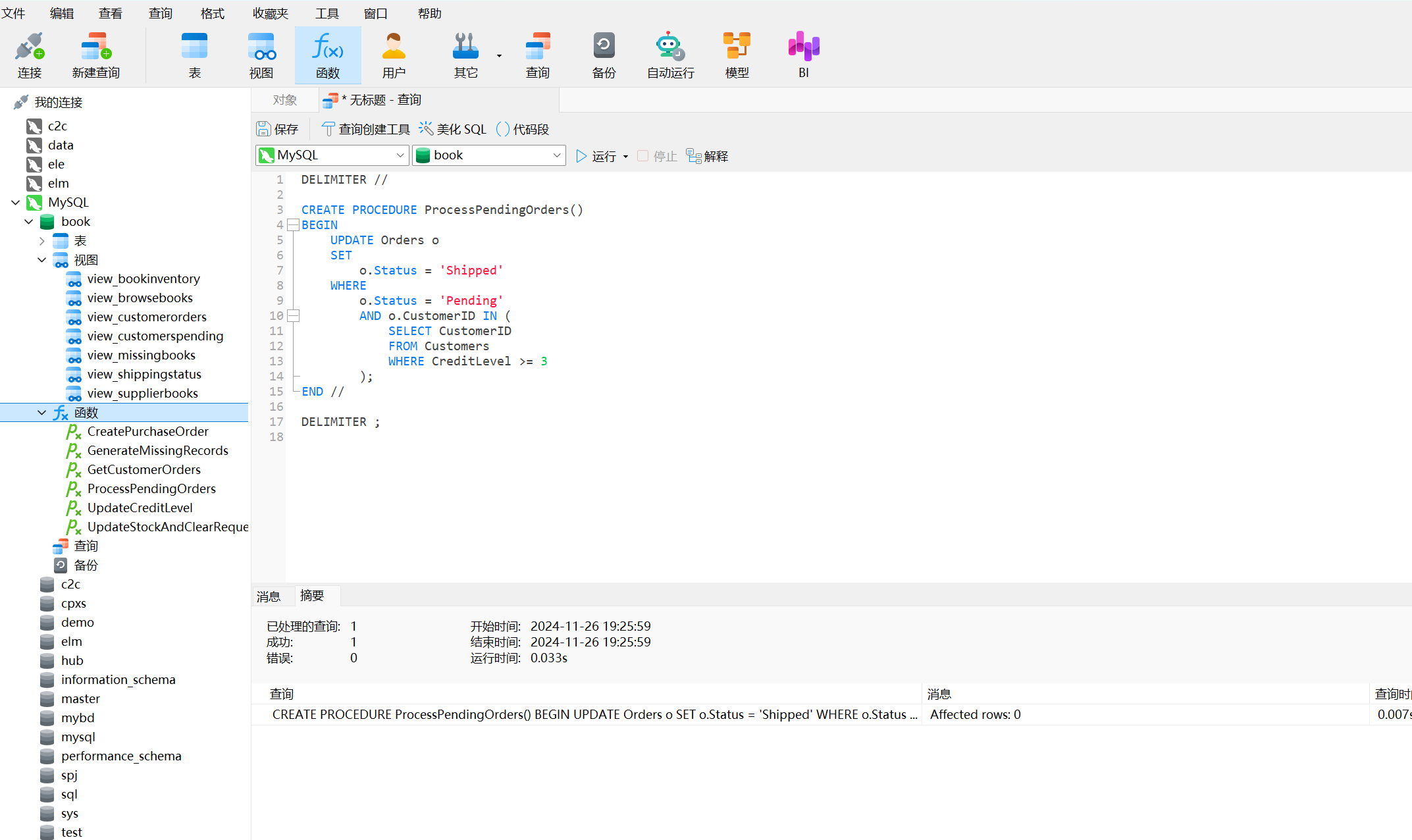
);

END //

DELIMITER ;

**用途**：

* 自动处理未发货订单。



**实验总结：**

通过本次实验，我深入了解了数据库应用系统的需求分析过程，熟悉了数据库设计的基本方法并能够应用到实际场景中。我掌握了数据库和数据表的创建技巧，能够灵活设计视图以满足不同的查询需求。同时，我对数据完整性约束有了全面认识，包括主键、外键以及检查约束等在维护数据一致性和安全性中的重要作用。此外，通过实验，我对存储过程和触发器的应用有了更加深刻的理解，这些工具能够有效简化复杂业务逻辑的实现，提高了数据库操作的效率和安全性。

**教师评语及成绩**：