

软件学院

数据库报告：图书销售管理系统

|  |  |
| --- | --- |
| 选课编号： | 113007 |
| 分组编号： | 24 |
| 姓名： | 程鹏宇 |
| 学号： | 3124336012 |
| 班级： | 1班 |
| 指导老师： | 白秀秀 |
| 完成日期： | 2024年11月31号 |

**目录**

[1. 需求分析 1](#_Toc24341)

[2. 概念结构设计 1](#_Toc336)

[3. 逻辑结构设计 2](#_Toc27173)

[3.1建立关系模式 2](#_Toc69)

[3.2关系模式规范化处理 3](#_Toc20090)

[1)类别 3](#_Toc27336)

[3)图书 3](#_Toc5897)

[4)客户 3](#_Toc3322)

[5)订单 4](#_Toc126)

[6)订单详情 4](#_Toc5529)

[4. 数据库物理设计 4](#_Toc30787)

[5．数据库实施与测试 6](#_Toc22406)

[5.1数据入库 6](#_Toc7268)

[5.2建立视图 6](#_Toc31881)

[视图1 显示每个客户的订单总数和总金额 6](#_Toc3023)

[视图2 显示每个类别的图书总数和总库存量 7](#_Toc23308)

[5.3数据库测试 8](#_Toc4501)

[5.3.1增删改查操作 8](#_Toc28395)

[5.3.2建立并测试触发器 9](#_Toc19207)

[6．实验总结 10](#_Toc13706)

# 需求分析

根据题目需求分析，可知该系统应能够支持图书的分类、库存管理、客户管理、订单管理等功能。具体来说，每种图书应包含ISBN编号、书名、价格、作者、出版社、出版时间、库存数量等属性。图书属于不同的类别，并且每册图书只能属于一个类别，类别数据包括类别编号、类别名称、类别描述等。不同的出版社供应不同数量的各种图书用于销售，出版社数据包括出版社编号、出版社名称、联系方式等。书店客户具有客户编号、客户名、密码、注册日期、登录次数、账户余额等信息。客户可以根据自己的需求创建订单购买图书，每个订单只能属于一位客户，每个订单允许购买多种多册图书。

为了实现这些功能和需求，数据库设计是关键。系统中包含几个主要的实体：类别、出版社、图书、客户、订单和订单详情。类别表存储图书类别的信息，包括类别编号、类别名称和类别描述。出版社表存储出版社的信息，包括出版社编号、出版社名称和联系方式。图书表存储每本书的详细信息，包括ISBN编号、书名、价格、作者、出版社编号、出版时间、库存数量和类别编号。客户表存储客户的详细信息，包括客户编号、客户名、密码、注册日期、登录次数和账户余额。订单表存储每个订单的信息，包括订单编号、客户编号、订单日期和总金额。订单详情表存储每个订单的详细信息，包括订单编号、图书ISBN、数量和单价。这些实体之间存在多种关系，例如图书与类别是一对一关系，图书与出版社也是一对一关系，订单与客户是一对多关系，订单与订单详情是一对多关系，图书与订单详情也是一对多关系。

# 概念结构设计

根据上述需求分析，创建E-R图如图1所示。

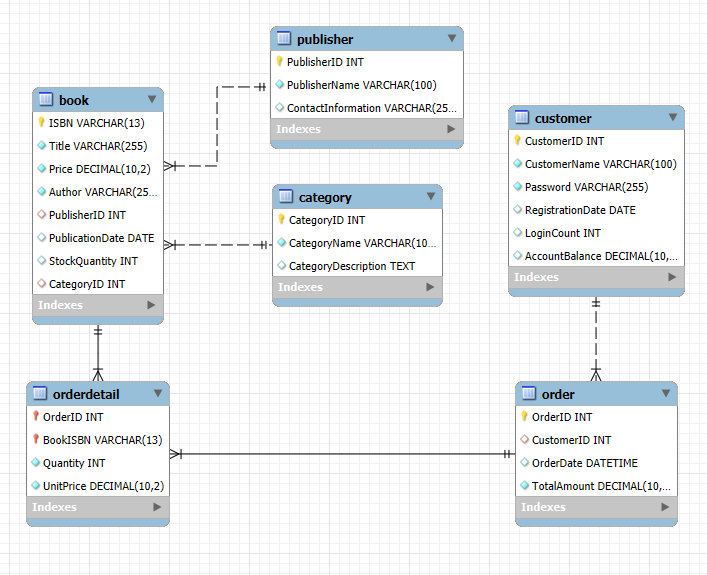


图1图书管理系统E-R图

# 逻辑结构设计

## 3.1建立关系模式

根据上述E-R图，建立关系模式如下：

Category(CategoryID,CategoryName,CategoryDescription)

Publisher(PublisherID,PublisherName,ContactInformation)

Book(ISBN,Title,Price,Author,PublisherID,PublicationDate,StockQuantity,CategoryID)

Customer(CustomerID,CustomerName,Password,RegistrationDate,LoginCount,AccountBalance)

Order(OrderID,CustomerID,OrderDate,TotalAmount)

OrderDetail(OrderID,BookISBN,Quantity,UnitPrice)

## 3.2关系模式规范化处理

### 1)类别

主码：CategoryID

非主属性：CategoryName,CategoryDescription

函数依赖：CategoryName完全函数依赖于CategoryID；CategoryDescription完全函数依赖于CategoryID

结论：类别关系已经满足3NF，因为所有非主属性都完全函数依赖于主码，且不存在传递函数依赖。

#### 2)出版社

主码：PublisherID

非主属性：PublisherName,ContactInformation

函数依赖：PublisherName完全函数依赖于PublisherID；ContactInformation完全函数依赖于PublisherID

结论：出版社关系已经满足3NF，因为所有非主属性都完全函数依赖于主码，且不存在传递函数依赖。

### 3)图书

主码：ISBN

非主属性：Price,Author,PublisherID,PublicationDate,StockQuantity,CategoryID，Title

函数依赖：Title完全函数依赖于ISBN；Price完全函数依赖于ISBN；Author完全函数依赖于ISBN；PublisherID完全函数依赖于ISBN；PublicationDate完全函数依赖于ISBN；StockQuantity完全函数依赖于ISBN；CategoryID完全函数依赖于ISBN

结论：图书关系已经满足3NF，因为所有非主属性都完全函数依赖于主码，且不存在传递函数依赖。

### 4)客户

主码：CustomerID

非主属性：Password,RegistrationDate,LoginCount,AccountBalance,CustomerName

函数依赖：CustomerName完全函数依赖于CustomerID；Password完全函数依赖于CustomerID；RegistrationDate完全函数依赖于CustomerID；LoginCount完全函数依赖于CustomerID；AccountBalance完全函数依赖于CustomerID

结论：客户关系已经满足3NF，因为所有非主属性都完全函数依赖于主码，且不存在传递函数依赖。

### 5)订单

主码：OrderID

非主属性：CustomerID,OrderDate,TotalAmount

函数依赖：CustomerID完全函数依赖于OrderID；OrderDate完全函数依赖于OrderID；TotalAmount完全函数依赖于OrderID

结论：订单关系已经满足3NF，因为所有非主属性都完全函数依赖于主码，且不存在传递函数依赖。

### 6)订单详情

主码：OrderID,BookISBN

非主属性：Quantity,UnitPrice

函数依赖：Quantity完全函数依赖于(OrderID,BookISBN)；UnitPrice完全函数依赖于(OrderID,BookISBN)

结论：订单详情关系已经满足3NF，因为所有非主属性都完全函数依赖于主码，且不存在传递函数依赖。

# 数据库物理设计

数据库的SQL脚本代码如下

-- 创建数据库

CREATE DATABASE book\_sales\_system;

USE book\_sales\_system;

-- 创建类别表

CREATE TABLE Category (

    CategoryID INT AUTO\_INCREMENT PRIMARY KEY,

    CategoryName VARCHAR(100) NOT NULL,

    CategoryDescription TEXT

);

-- 创建出版社表

CREATE TABLE Publisher (

    PublisherID INT AUTO\_INCREMENT PRIMARY KEY,

    PublisherName VARCHAR(100) NOT NULL,

    ContactInformation VARCHAR(255)

);

-- 创建图书表

CREATE TABLE Book (

    ISBN VARCHAR(13) PRIMARY KEY,

    Title VARCHAR(255) NOT NULL,

    Price DECIMAL(10, 2) NOT NULL,

    Author VARCHAR(255) NOT NULL,

    PublisherID INT,

    PublicationDate DATE,

    StockQuantity INT,

    CategoryID INT,

    FOREIGN KEY (PublisherID) REFERENCES Publisher(PublisherID),

    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID)

);

-- 创建客户表

CREATE TABLE Customer (

    CustomerID INT AUTO\_INCREMENT PRIMARY KEY,

    CustomerName VARCHAR(100) NOT NULL,

    Password VARCHAR(255) NOT NULL,

    RegistrationDate DATE,

    LoginCount INT DEFAULT 0,

    AccountBalance DECIMAL(10, 2) DEFAULT 0.00

);

-- 创建订单表

CREATE TABLE `Order` (

    OrderID INT AUTO\_INCREMENT PRIMARY KEY,

    CustomerID INT,

    OrderDate DATETIME DEFAULT CURRENT\_TIMESTAMP,

    TotalAmount DECIMAL(10, 2) NOT NULL,

    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)

);

-- 创建订单详情表

CREATE TABLE OrderDetail (

    OrderID INT,

    BookISBN VARCHAR(13),

    Quantity INT NOT NULL,

    UnitPrice DECIMAL(10, 2) NOT NULL,

    PRIMARY KEY (OrderID, BookISBN),

    FOREIGN KEY (OrderID) REFERENCES `Order`(OrderID),

    FOREIGN KEY (BookISBN) REFERENCES Book(ISBN)

);

# 5．数据库实施与测试

## 5.1数据入库

使用SQL脚本进行数据入库，脚本如下，成功创建。

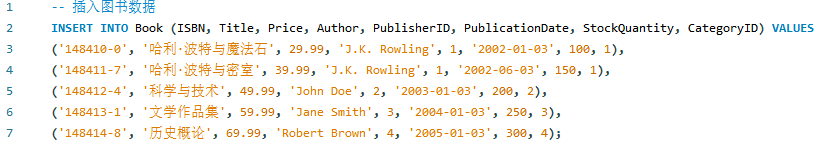


图 2 数据入库操作



图 3 成功入库

## 5.2建立视图

### 视图1 显示每个客户的订单总数和总金额

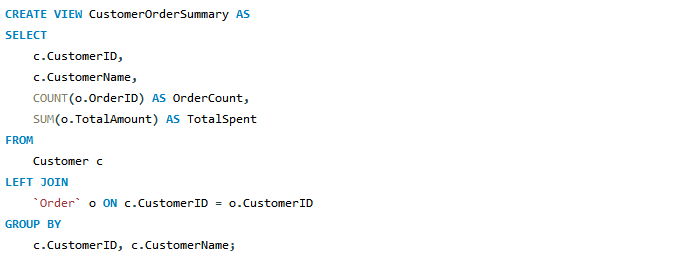


图 4 订单总数和总金额视图

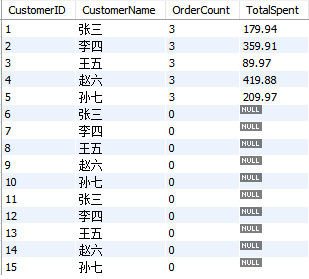


图 5 订单总数和总金额输出

### 视图2 显示每个类别的图书总数和总库存量

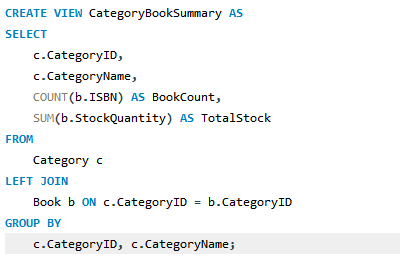


图 6 图书总数和库存总量视图

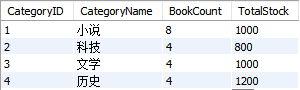


图 7 图书总量和库存总量输出

## 5.3数据库测试

### 5.3.1增删改查操作

#### 增加

图 8 增加操作



图 9 成功增加

#### 删除



图 10 删除操作

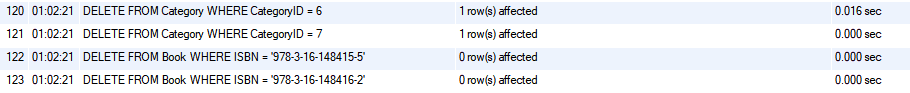


图 11 成功删除

#### 更新

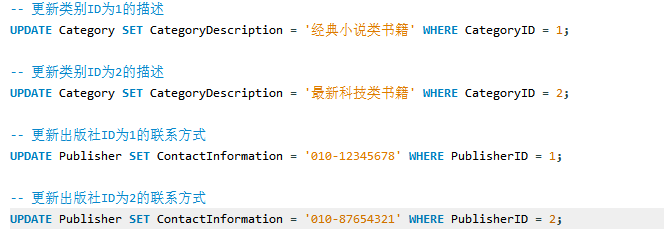


图 12 更新操作

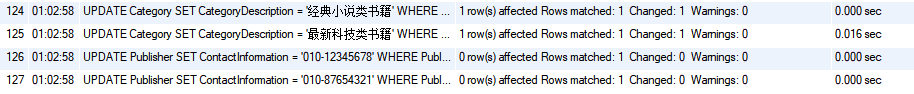


图 13 成功更新

### 5.3.2建立并测试触发器

#### 1)在插入订单详情时减少图书库存

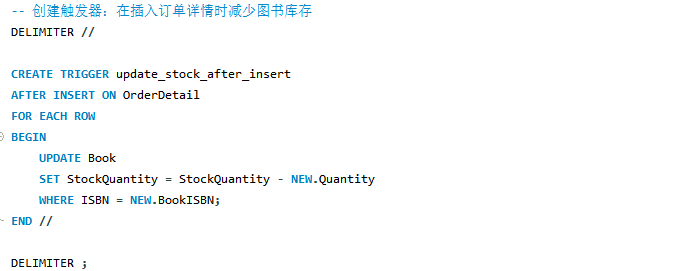


图 14 触发器一脚本



图 15 触发器一成功建立

#### 在删除订单详情时增加图书库存

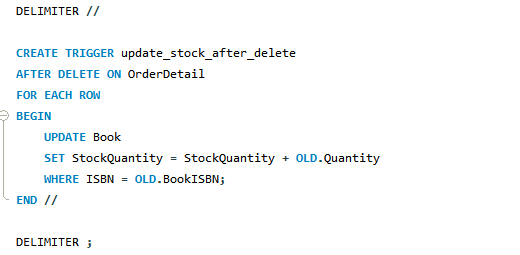


图 16 触发器二脚本



图 17 触发器二成功建立

#### 测试触发器

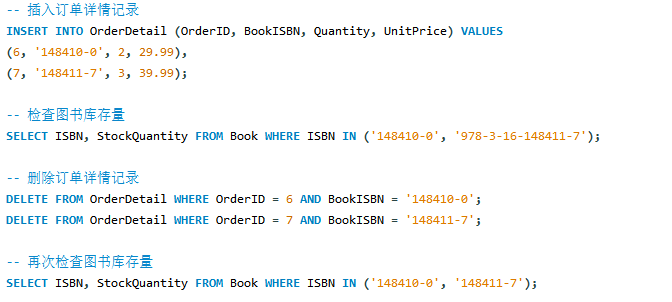


图 18 触发器测试脚本

若触发器正确，结果应为148410-0的库存为100和148411-7的库存为150.结果如下所示，和预想的一样，这就验证了结果的正确性

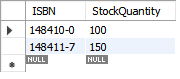


图 19 测试结果正确

# 6．实验总结

在本次数据库设计过程中，我完成了创建数据库、表结构设计、数据插入、视图创建以及触发器的设计和测试等任务。通过这次实践，我对数据库的设计和管理有了更深的理解。