

《数据库系统概论》课程设计报告

《图书销售系统系统》

班级：

学号：

姓名：

目 录

[**一、系统需求分析** **1**](#_Toc8423_WPSOffice_Level1)

[（一）需求概述 1](#_Toc26487_WPSOffice_Level2)

[（二）组织结构分析 1](#_Toc2103_WPSOffice_Level2)

[（三）管理功能分析 1](#_Toc1531_WPSOffice_Level2)

[（四）业务流分析 2](#_Toc10514_WPSOffice_Level2)

[（五）数据流分析 4](#_Toc30037_WPSOffice_Level2)

[（六）数据字典 5](#_Toc4071_WPSOffice_Level2)

[**二、数据库概念结构设计** **8**](#_Toc26487_WPSOffice_Level1)

[（一）实体分析 8](#_Toc8785_WPSOffice_Level2)

[（二）属性分析 8](#_Toc25633_WPSOffice_Level2)

[（三）联系分析 10](#_Toc5435_WPSOffice_Level2)

[（四）概念模型设计（.cdm图） 11](#_Toc14653_WPSOffice_Level2)

[**三、数据库逻辑结构设计** **12**](#_Toc2103_WPSOffice_Level1)

[（一）概念模型转化为逻辑模型 12](#_Toc5745_WPSOffice_Level2)

[1、一对一关系的转化（标注主、外键） 12](#_Toc26487_WPSOffice_Level3)

[2、一对多关系的转化（标注主、外键） 13](#_Toc2103_WPSOffice_Level3)

[3、多对多关系的转化（标注主、外键） 13](#_Toc1531_WPSOffice_Level3)

[（二）逻辑模型设计（.pdm图） 13](#_Toc5528_WPSOffice_Level2)

[**四、数据库的物理实现** **14**](#_Toc1531_WPSOffice_Level1)

[（一）表设计（SQL server 中各表设计） 14](#_Toc15502_WPSOffice_Level2)

[1. 用户表设计 14](#_Toc10514_WPSOffice_Level3)

[2. 管理员表设计： 14](#_Toc30037_WPSOffice_Level3)

[3. 图书表设计： 15](#_Toc4071_WPSOffice_Level3)

[4. 图书分类表设计： 15](#_Toc8785_WPSOffice_Level3)

[5. 订单列表设计： 15](#_Toc25633_WPSOffice_Level3)

[6. 订单表设计： 15](#_Toc5435_WPSOffice_Level3)

[7. 管理订单： 16](#_Toc14653_WPSOffice_Level3)

[8. 管理图书： 16](#_Toc5745_WPSOffice_Level3)

[9. 管理用户： 16](#_Toc5528_WPSOffice_Level3)

[（二）创建表和完整性约束代码设计 16](#_Toc4468_WPSOffice_Level2)

[（三）创建视图、索引、存储过程和触发器 22](#_Toc19967_WPSOffice_Level2)

[1、创建视图 22](#_Toc4468_WPSOffice_Level3)

[2、创建索引 23](#_Toc19967_WPSOffice_Level3)

[3、创建存储过程 23](#_Toc25595_WPSOffice_Level3)

[4、创建触发器 24](#_Toc20829_WPSOffice_Level3)

[**五、数据库功能调试（包括视图、索引等内容的测试）** **24**](#_Toc4468_WPSOffice_Level1)

[（一）管理员管理图书信息模块 24](#_Toc25595_WPSOffice_Level2)

[1、查询图书信息 24](#_Toc1195_WPSOffice_Level3)

[2、插入图书记录 25](#_Toc7184_WPSOffice_Level3)

[3、删除图书记录 26](#_Toc5858_WPSOffice_Level3)

[4、修改图书记录 26](#_Toc9376_WPSOffice_Level3)

[（二）用户查看并更改用户信息模块 26](#_Toc20829_WPSOffice_Level2)

[1、查询用户信息 26](#_Toc32055_WPSOffice_Level3)

[2、插入用户信息 27](#_Toc14505_WPSOffice_Level3)

[3、修改用户记录 27](#_Toc5789_WPSOffice_Level3)

[（三）订单的创建 28](#_Toc1195_WPSOffice_Level2)

[1、查询订单 28](#_Toc27666_WPSOffice_Level3)

[2、插入订单 29](#_Toc32048_WPSOffice_Level3)

[（四）视图、索引、存储过程和触发器测试 30](#_Toc7184_WPSOffice_Level2)

[1、book视图 30](#_Toc14097_WPSOffice_Level3)

[2、books索引 31](#_Toc7344_WPSOffice_Level3)

[3、bookname存储过程 31](#_Toc24892_WPSOffice_Level3)

[4、book\_in触发器 32](#_Toc26058_WPSOffice_Level3)

[**六、应用程序设计** **32**](#_Toc19967_WPSOffice_Level1)

[（一）动态页面网页设计 32](#_Toc5858_WPSOffice_Level2)

[1、登录界面设计 32](#_Toc2618_WPSOffice_Level3)

[2、数据库连接实现 32](#_Toc19602_WPSOffice_Level3)

[（二）管理员登录模块 32](#_Toc9376_WPSOffice_Level2)

[（三）图书表查看模块 33](#_Toc32055_WPSOffice_Level2)

[（四）图书表插入记录 34](#_Toc14505_WPSOffice_Level2)

[（五）图书表内容修改 35](#_Toc5789_WPSOffice_Level2)

[**七、设计总结** **35**](#_Toc25595_WPSOffice_Level1)

**一、系统需求分析**

（一）需求概述

本次课程设计的题目是《图书销售系统》，系统内容要求有：图书销售系统功能包括零售购买：顾客购书后收银台进行结账。对于书店的会员可以提供相应的折扣。输入需要购买的图书和数量，计算出总金额，由用户选择使用现金或会员卡进行结账。并提供销售小票流水号作为销售的单据。对于会员，还要计算相应的积分。图书零售退货：顾客对已购买的图书进行退货。需要提供图书和销售的小票以作为购买凭证。系统查询数据库进行数据验证，对符合要求的图书进行退货。

本数据库系统对个人书商销售的各个环节实现自动化操作，让书店店主了解自己的销售情况，并且帮助店主管理业务，提供销售情况的统计。

数据库有六个实体集如下：

一是“用户实体集”，属性包含用户注册账号、用户密码、用户地址、用户名字、用户性别、用户电话、用户注册时间、用户邮箱、该用户是否为会员及会员积分。

二是“管理员”实体集，属性包含管理员用户名、管理员密码、管理员真名、管理员性别、管理员电话、管理员邮箱。

三是“图书”实体集，属性包含图书ISBN号、图书名称、图书剩余数量、图书分类编号、图书总数量、图书价格。图书内容介绍、图书出版社、图书作者。

四是“订单列表”实体集，属性包含订单编号、购买时间、总价格、详细附加内容。

五是“订单”实体集，属性包含订单编号、图书ISBN号、图书价格。

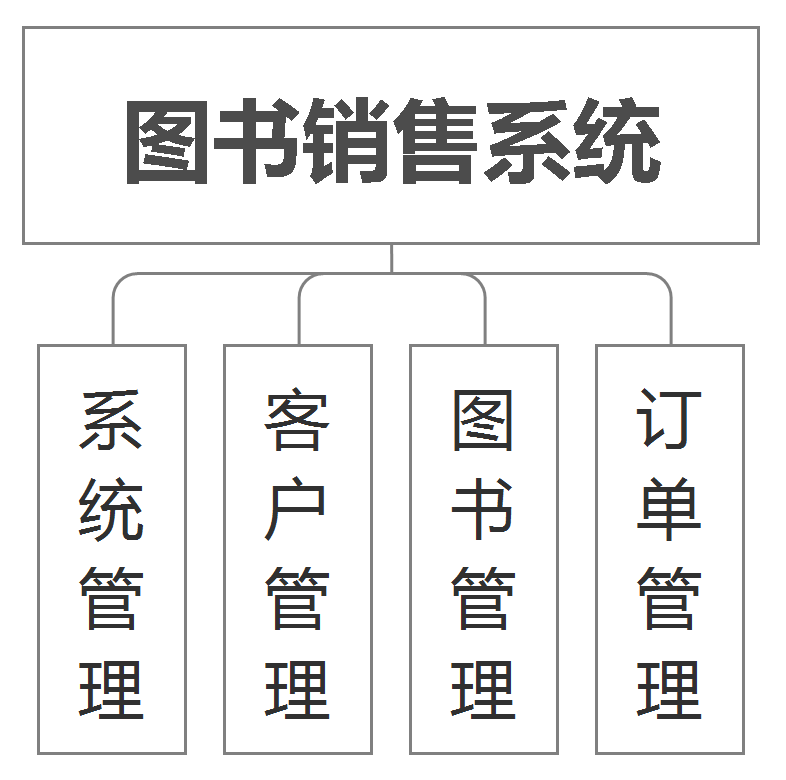
六是“图书分类”实体集，属性包含分类编号、分类名称。

（二）组织结构分析

本系统适用于小型书店的零售购买，提供给店主管理员的权限，可以对于数据库进行一系列的操作，例如对于图书、订单等的增、删、查、改操作，而用户则值提供数据库的查看权限，以便于用户查询书店有的图书和用户自己购买过得订单信息。

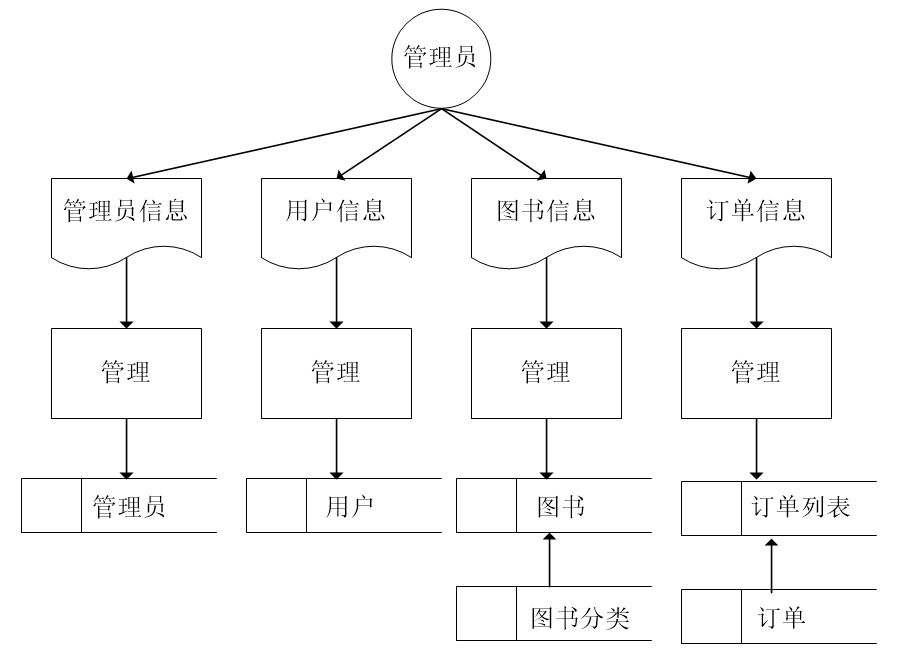
（三）管理功能分析

本图书销售系统主要功能是管理员进行系统的管理，以及管理客户信息、图书信息和详细的各种订单信息。每部分信息的管理均包含增、删、改、查，其中管理员权限最高，可以对所有表进行操作。



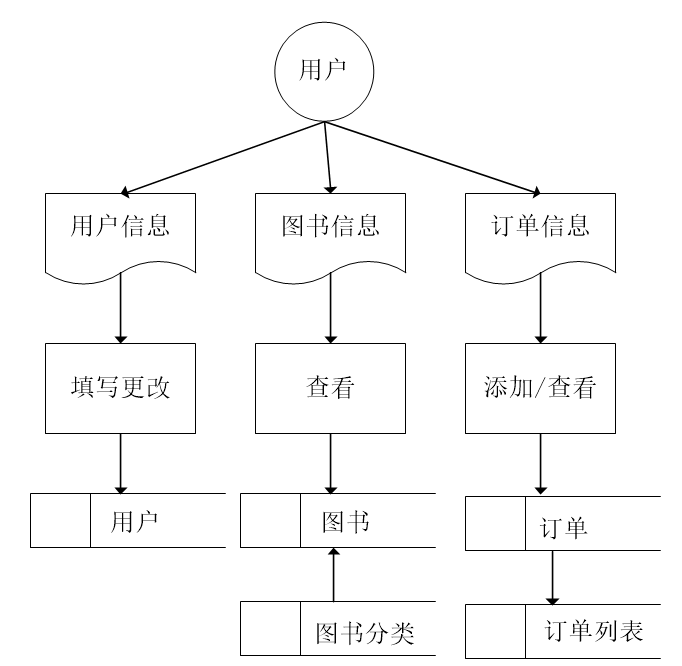
（四）业务流分析

管理员登录：



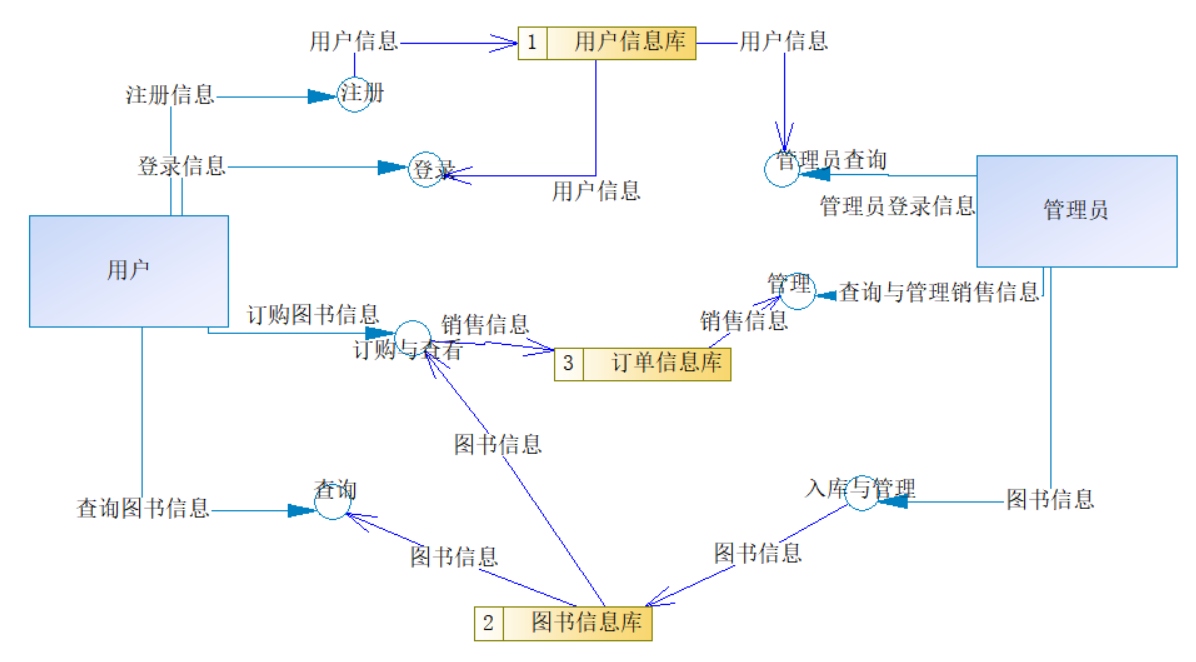
用户登录：





（五）数据流分析

根据系统的功能需求，对其使用结构化分析方法，进一步对需求进行分析整理，得到本系统的数据流图如下：



（六）数据字典

根据数据流图中所涉及的信息，并对信息进行相应的分析，确定出所有数据项的描述内容，其主要分为数据项名称、类型、长度和取值范围，如下表：

数据项：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 数据项名称 | 含义说明 | 数据类型 | 长度 | 取值范围 | 数据项联系说明备注 |
| 用户注册账号 | 顾客注册的账户的账号 | char | 10 |  |  |
| 用户密码 | 顾客账户的登录密码 | char | 10 |  |  |
| 用户地址 | 顾客常用地址 | char | 20 |  |  |
| 用户名字 | 顾客账户名字 | char | 10 |  |  |
| 用户性别 | 顾客的性别 | char | 2 | 男/女 |  |
| 用户电话 | 顾客的联系电话 | char | 20 |  |  |
| 用户注册时间 | 顾客注册的时间 | date |  |  |  |
| 用户邮箱 | 顾客的邮箱号 | char | 30 |  |  |
| 会员 | 标记该顾客是否为会员 | bit |  | 1/0 | 1则是会员，0则不是会员 |
| 积分 | 会员拥有的积分 | int |  |  | 非会员则积分为0 |
| 管理员用户名 | 书店管理员账户的用户名 | char | 10 |  |  |
| 管理员密码 | 书店管理员账户的登录密码 | char | 10 |  |  |
| 管理员真名 | 书店管理员的真实姓名 | char | 10 |  |  |
| 管理员性别 | 书店管理员的性别 | char | 2 | 男/女 |  |
| 管理员电话 | 书店管理员的联系电话 | char | 20 |  |  |
| 管理员邮箱 | 书店管理员的邮箱 | char | 30 |  |  |
| 图书ISBN号 | 图书的ISBN书号 | char | 13 |  |  |
| 图书名称 | 图书的书名 | char | 20 |  |  |
| 图书剩余数量 | 该图书剩余的库存数量 | int |  |  |  |
| 图书价格 | 图书的售卖原价 | float |  |  |  |
| 图书内容介绍 | 图书内容简介 | text |  |  |  |
| 图书出版社 | 图书的出版社 | char | 50 |  |  |
| 图书作者 | 图书的作者 | char | 50 |  |  |
| 分类编号 | 图书分类的类型编号 | char | 10 |  |  |
| 分类名称 | 图书分类的类型名称 | char | 100 |  |  |
| 订单编号 | 订单的编号即为小票流水号 | char | 20 |  |  |
| 购买时间 | 订单购买即产生的时间 | date |  |  |  |
| 总价格 | 该笔订单的总价格 | float |  |  |  |
| 内容 | 订单的附加信息 | text |  |  |  |

根据数据流图中的信息的分析，在数据项描述的基础上确定所有数据结构的描述，主要有数据结构名称、含义和组成说明。

数据结构：

|  |  |  |
| --- | --- | --- |
| 数据结构名称 | 含义 | 组成 |
| 用户 | 各个顾客用户的信息 | 用户注册账号、用户密码、用户地址、用户名字、用户性别、用户电话、用户注册时间、用户邮箱、会员、积分 |
| 管理员 | 书店管理员信息 | 管理员用户名、管理员密码、管理员真名、管理员性别、管理员电话、管理员邮箱 |
| 图书 | 图书的信息 | 图书ISBN号、图书名称、图书剩余数量、图书价格、图书内容介绍、图书出版社、图书作者 |
| 图书分类 | 图书分类的类型信息 | 分类编号、分类名称 |
| 订单列表 | 订单的信息 | 订单编号、购买时间、总价格、内容 |

根据数据流图的数据流向的分析，确定所有数据流的描述，主要有数据流名称、数据流来源。数据流去向和说明。

数据流：

|  |  |  |  |
| --- | --- | --- | --- |
| 数据流名 | 数据流来源 | 数据流去向 | 说明 |
| 用户信息 | 顾客用户注册 | 用户登录、管理员查询 | 用户信息的管理 |
| 订单信息 | 用户顾客购买 | 管理员管理 | 订单信息的管理 |
| 图书信息 | 管理员录入 | 用户查看 | 图书信息的管理 |

**二、数据库概念结构设计**

（一）实体分析

经需求分析，本次图书销售系统中包含六个实体，它们分别是：用户实体、管理员实体、图书实体、图书分类实体、订单列表实体和订单实体。

（二）属性分析

用户实体属性：用户注册账号、用户密码、用户地址、用户名字、用户性别、用户电话、用户注册时间、用户邮箱、会员、积分。

用户实体ER图：



管理员实体属性：管理员用户名、管理员密码、管理员真名、管理员性别、管理员电话、管理员邮箱。

管理员实体ER图：



图书实体属性：图书ISBN号、图书名称、图书剩余数量、图书价格、图书内容介绍、图书出版社、图书作者。

图书实体ER图：



图书分类实体属性：分类编号、分类名称。

图书分类实体ER图：



订单列表实体属性：订单编号、购买时间、总价格、内容。

订单列表实体ER图：



订单实体属性：用户注册账号、订单编号、图书ISBN号、图书价格。

订单实体ER图：



（三）联系分析

图书销售系统中包含六个实体，它们分别是：用户实体、管理员实体、图书实体、图书分类实体、订单列表实体和订单实体。

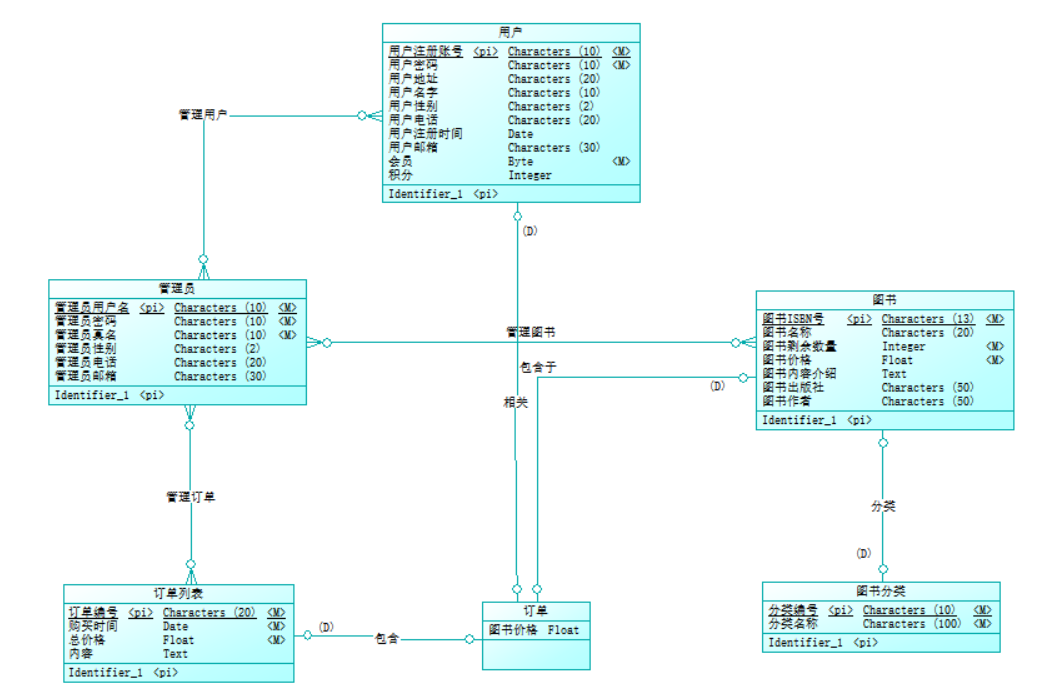
用户与图书有查看和购买联系，用户可以查看并购买图书，同时用户购买图书即生成一个订单，是用户和订单的联系；每个用户生成的不同订单集合到订单列表，两者之间有详细信息的联系；管理员与图书和图书分类有管理联系，管理员通过进货入库和销售出库对于图书详细信息进行更改；同时管理员可以查看订单，对其有管理权限。

各个实体之间的ER图：



（四）概念模型设计（.cdm图）

经过需求分析和实体属性分析，以及各实体之间的关系分析，最终得到概念模型如下：



**三、数据库逻辑结构设计**

（一）概念模型转化为逻辑模型

1、一对一关系的转化（标注主、外键）

一个1:1联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。

①转换为一个独立的关系模式

关系的属性：与该联系相连的各实体的码以及联系本身的属性

关系的候选码：每个实体的码均是该关系的候选码

②与某一端实体对应的关系模式合并

合并后关系的属性：加入对应关系的码和联系本身的属性

合并后关系的码：不变

每本图书属于一种分类，拥有一个分类编号，其中图书分类中的分类编号为主键，图书中的分类编号为外键；

每一个订单拥有唯一的订单编号、用户和图书ISBN号，即每一个订单包含的订单编号、用户注册账号、图示ISBN号为外键，它们分别是订单列表、用户、图书实体的主键。

2、一对多关系的转化（标注主、外键）

一个1:n联系可以转换为一个独立的关系模式，也可以与n端对应的关系模式合并。

①转换为一个独立的关系模式

关系的属性：与该联系相连的各实体的码以及联系本身的属性

关系的码：n端实体的码

②与n端对应的关系模式合并

合并后关系的属性：在n端关系中加入1端关系的码和联系本身的属性

合并后关系的码：不变

可以减少系统中的关系个数，一般情况下更倾向于采用这种方法

本图书销售管理系统没有一对多关系的转化。

3、多对多关系的转化（标注主、外键）

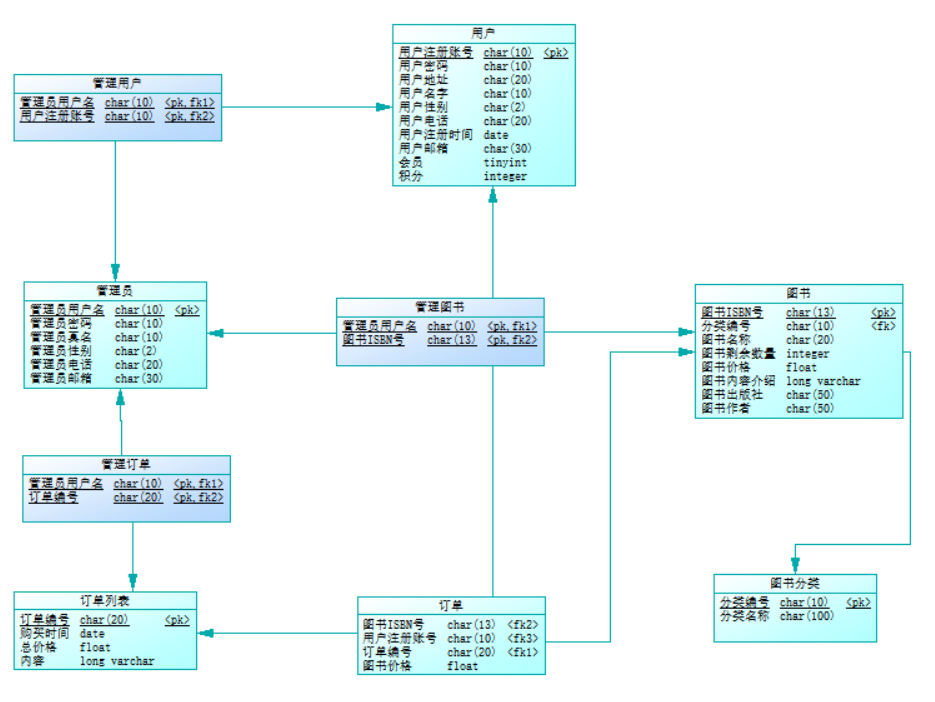
一个m:n联系转换为一个关系模式

关系的属性：与该联系相连的各实体的码以及联系本身的属性

关系的码：各实体码的组合

每个管理员（主键：管理员用户名）可以管理多个用户信息（主键：用户注册账号）、图书信息（主键：图书ISBN号）、图书分类信息（主键：分类编号）、订单列表信息（主键：订单编号）、订单信息。

（二）逻辑模型设计（.pdm图）



**四、数据库的物理实现**

（一）表设计（SQL server 中各表设计）

1. 用户表设计



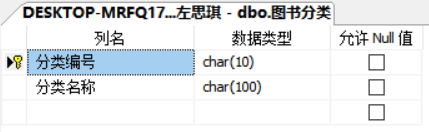
1. 管理员表设计：



1. 图书表设计：



1. 图书分类表设计：



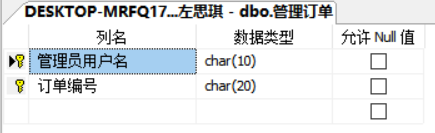
1. 订单列表设计：



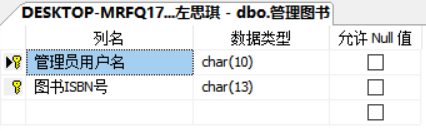
1. 订单表设计：



1. 管理订单：



1. 管理图书：



1. 管理用户：



（二）创建表和完整性约束代码设计

/\*==============================================================\*/

/\* Table: 图书 \*/

/\*==============================================================\*/

create table 图书

(

图书ISBN号 char(13) not null,

分类编号 char(10) null,

图书名称 char(20) null,

图书剩余数量 integer not null,

图书价格 float not null,

图书内容介绍 text null,

图书出版社 char(50) null,

图书作者 char(50) null,

constraint PK\_图书 primary key (图书ISBN号)

);

/\*==============================================================\*/

/\* Index: 图书\_PK \*/

/\*==============================================================\*/

create unique index 图书\_PK on 图书 (

图书ISBN号 ASC

);

/\*==============================================================\*/

/\* Table: 图书分类 \*/

/\*==============================================================\*/

create table 图书分类

(

分类编号 char(10) not null,

分类名称 char(100) not null,

constraint PK\_图书分类 primary key (分类编号)

);

/\*==============================================================\*/

/\* Index: 图书分类\_PK \*/

/\*==============================================================\*/

create unique index 图书分类\_PK on 图书分类 (

分类编号 ASC

);

/\*==============================================================\*/

/\* Table: 用户 \*/

/\*==============================================================\*/

create table 用户

(

用户注册账号 char(10) not null,

用户密码 char(10) not null,

用户地址 char(20) null,

用户名字 char(10) null,

用户性别 char(2) null,

用户电话 char(20) null,

用户注册时间 date null,

用户邮箱 char(30) null,

是否为会员 tinyint not null,

积分 integer null,

constraint PK\_用户 primary key (用户注册账号)

);

/\*==============================================================\*/

/\* Index: 用户\_PK \*/

/\*==============================================================\*/

create unique index 用户\_PK on 用户 (

用户注册账号 ASC

);

/\*==============================================================\*/

/\* Table: 管理员 \*/

/\*==============================================================\*/

create table 管理员

(

管理员用户名 char(10) not null,

管理员密码 char(10) not null,

管理员真名 char(10) not null,

管理员性别 char(2) null,

管理员电话 char(20) null,

管理员邮箱 char(30) null,

constraint PK\_管理员 primary key (管理员用户名)

);

/\*==============================================================\*/

/\* Index: 管理员\_PK \*/

/\*==============================================================\*/

create unique index 管理员\_PK on 管理员 (

管理员用户名 ASC

);

/\*==============================================================\*/

/\* Table: 管理图书 \*/

/\*==============================================================\*/

create table 管理图书

(

管理员用户名 char(10) not null,

图书ISBN号 char(13) not null,

constraint PK\_管理图书 primary key clustered (管理员用户名, 图书ISBN号)

);

/\*==============================================================\*/

/\* Index: 管理图书\_FK \*/

/\*==============================================================\*/

create index 管理图书\_FK on 管理图书 (

管理员用户名 ASC

);

/\*==============================================================\*/

/\* Index: 管理图书\_FK \*/

/\*==============================================================\*/

create index 管理图书\_FK on 管理图书 (

图书ISBN号 ASC

);

/\*==============================================================\*/

/\* Table: 管理用户 \*/

/\*==============================================================\*/

create table 管理用户

(

管理员用户名 char(10) not null,

用户注册账号 char(10) not null,

constraint PK\_管理用户 primary key clustered (管理员用户名, 用户注册账号)

);

/\*==============================================================\*/

/\* Index: 管理用户\_FK \*/

/\*==============================================================\*/

create index 管理用户\_FK on 管理用户 (

管理员用户名 ASC

);

/\*==============================================================\*/

/\* Index: 管理用户\_FK \*/

/\*==============================================================\*/

create index 管理用户\_FK on 管理用户 (

用户注册账号 ASC

);

/\*==============================================================\*/

/\* Table: 管理订单 \*/

/\*==============================================================\*/

create table 管理订单

(

管理员用户名 char(10) not null,

订单编号 char(20) not null,

constraint PK\_管理订单 primary key clustered (管理员用户名, 订单编号)

);

/\*==============================================================\*/

/\* Index: 管理订单\_FK \*/

/\*==============================================================\*/

create index 管理订单\_FK on 管理订单 (

管理员用户名 ASC

);

/\*==============================================================\*/

/\* Index: 管理订单\_FK \*/

/\*==============================================================\*/

create index 管理订单\_FK on 管理订单 (

订单编号 ASC

);

/\*==============================================================\*/

/\* Table: 订单 \*/

/\*==============================================================\*/

create table 订单

(

图书ISBN号 char(13) null,

用户注册账号 char(10) null,

订单编号 char(20) null,

图书价格 float null

);

/\*==============================================================\*/

/\* Index: 包含\_FK \*/

/\*==============================================================\*/

create index 包含\_FK on 订单 (

订单编号 ASC

);

/\*==============================================================\*/

/\* Index: 包含于\_FK \*/

/\*==============================================================\*/

create index 包含于\_FK on 订单 (

图书ISBN号 ASC

);

/\*==============================================================\*/

/\* Index: 相关\_FK \*/

/\*==============================================================\*/

create index 相关\_FK on 订单 (

用户注册账号 ASC

);

/\*==============================================================\*/

/\* Table: 订单列表 \*/

/\*==============================================================\*/

create table 订单列表

(

订单编号 char(20) not null,

购买时间 date not null,

总价格 float not null,

内容 text null,

constraint PK\_订单列表 primary key (订单编号)

);

/\*==============================================================\*/

/\* Index: 订单列表\_PK \*/

/\*==============================================================\*/

create unique index 订单列表\_PK on 订单列表 (

订单编号 ASC

);

alter table 图书

add constraint FK\_图书\_分类\_图书分类 foreign key (分类编号)

references 图书分类 (分类编号)

alter table 管理图书

add constraint FK\_管理图书\_管理图书\_管理员 foreign key (管理员用户名)

references 管理员 (管理员用户名)

alter table 管理图书

add constraint FK\_管理图书\_管理图书\_图书 foreign key (图书ISBN号)

references 图书 (图书ISBN号)

alter table 管理用户

add constraint FK\_管理用户\_管理用户\_管理员 foreign key (管理员用户名)

references 管理员 (管理员用户名)

alter table 管理用户

add constraint FK\_管理用户\_管理用户\_用户 foreign key (用户注册账号)

references 用户 (用户注册账号)

alter table 管理订单

add constraint FK\_管理订单\_管理订单\_管理员 foreign key (管理员用户名)

references 管理员 (管理员用户名)

alter table 管理订单

add constraint FK\_管理订单\_管理订单\_订单列表 foreign key (订单编号)

references 订单列表 (订单编号)

alter table 订单

add constraint FK\_订单\_包含\_订单列表 foreign key (订单编号)

references 订单列表 (订单编号)

alter table 订单

add constraint FK\_订单\_包含于\_图书 foreign key (图书ISBN号)

references 图书 (图书ISBN号)

alter table 订单

add constraint FK\_订单\_相关\_用户 foreign key (用户注册账号)

references 用户 (用户注册账号)

（三）创建视图、索引、存储过程和触发器

1、创建视图

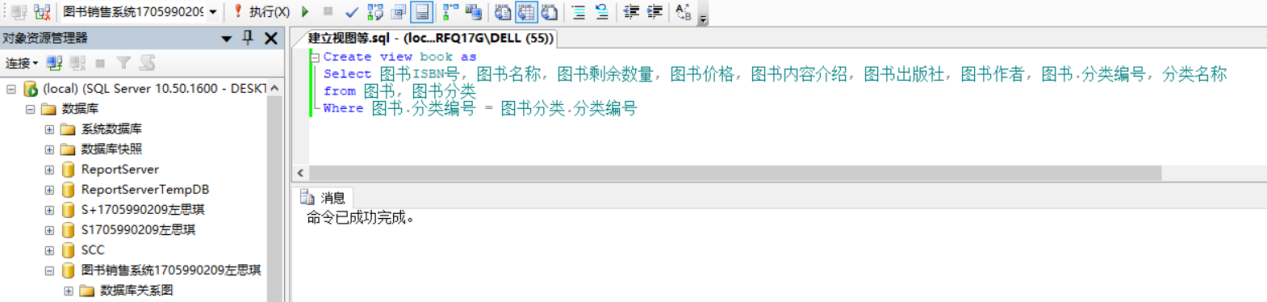
建立了book视图，方便用户查询图书具体信息。

create view book as

Select 图书ISBN号, 图书名称, 图书剩余数量, 图书价格, 图书内容介绍, 图书出版社, 图书作者, 图书.分类编号, 分类名称

from 图书, 图书分类

Where 图书.分类编号 = 图书分类.分类编号



建立的视图如下所示：



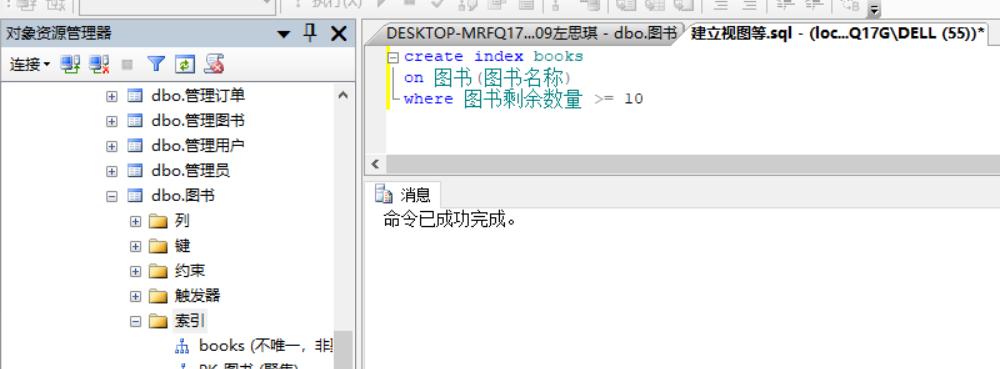
2、创建索引

在图书表创建了一个获得图书存货大于10的图书名称的非聚集索引。

create index books

on 图书(图书名称)

where 图书剩余数量 >= 10



3、创建存储过程

建立存储过程，输入图书名称得到该图书的所有信息。

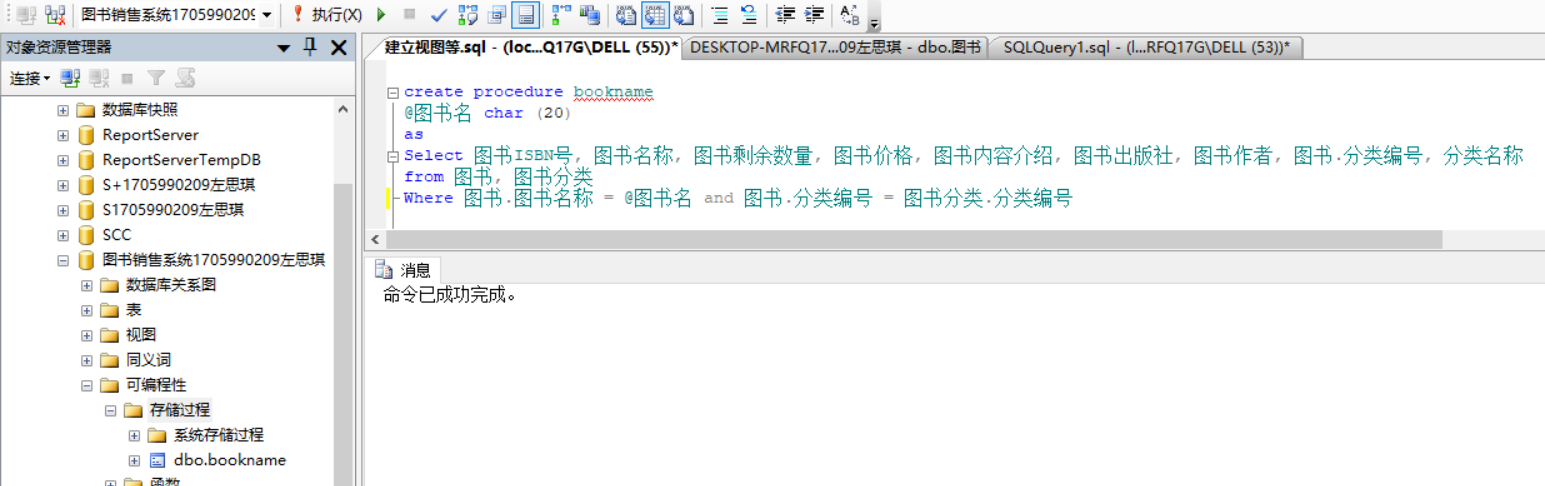
create procedure bookname

@图书名 char (20)

as

Select 图书ISBN号, 图书名称, 图书剩余数量, 图书价格, 图书内容介绍, 图书出版社, 图书作者, 图书.分类编号, 分类名称

from 图书, 图书分类

Where 图书.图书名称 = @图书名 and 图书.分类编号 = 图书分类.分类编号

4、创建触发器

创建触发器，当向图书中插入信息时打印出“图书插入成功”。

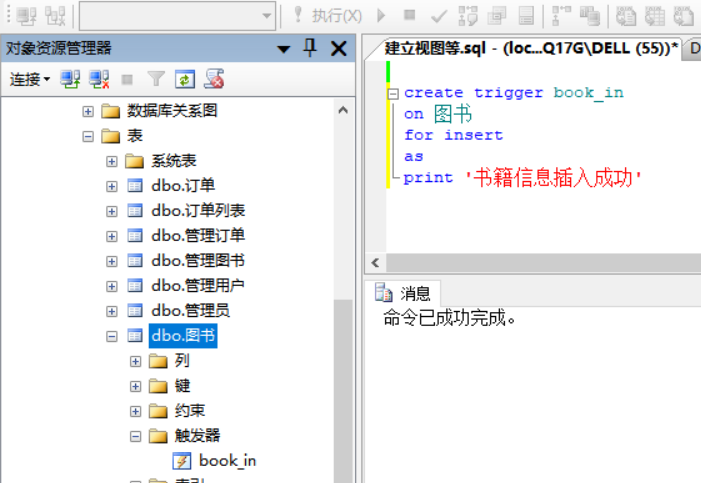
create trigger book\_in

on 图书

for insert

as

print '书籍信息插入成功'



**五、数据库功能调试（包括视图、索引等内容的测试）**

（一）管理员管理图书信息模块

1、查询图书信息

（1）测试数据



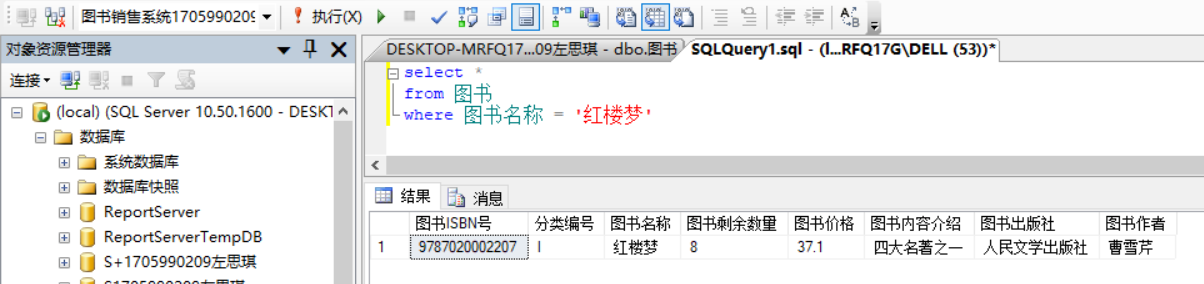
（2）SQL代码

select \*

from 图书

where 图书名称 = '红楼梦'

（3）执行结果



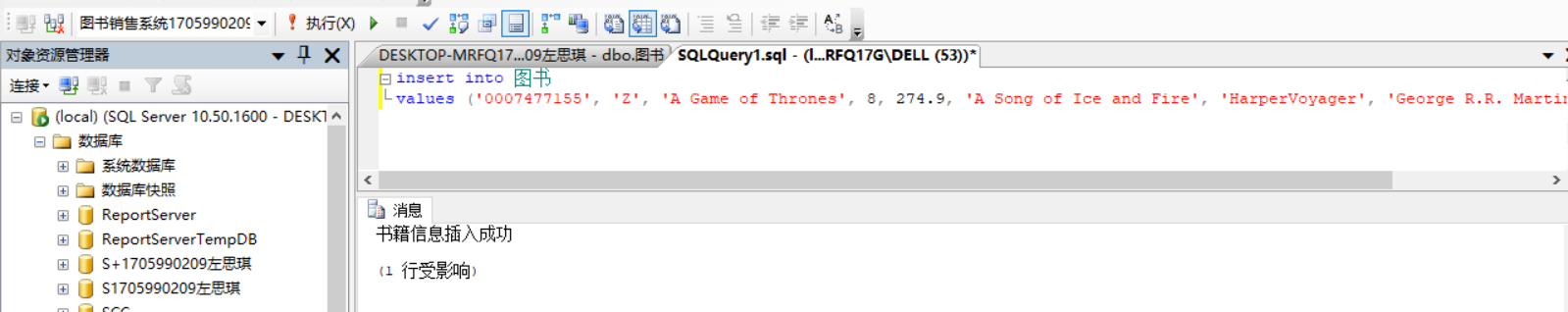
2、插入图书记录

（1）SQL代码

insert into 图书

values ('0007477155', 'Z', 'A Game of Thrones', 8, 274.9, 'A Song of Ice and Fire', 'HarperVoyager', 'George R.R. Martin')

（2）执行结果





3、删除图书记录

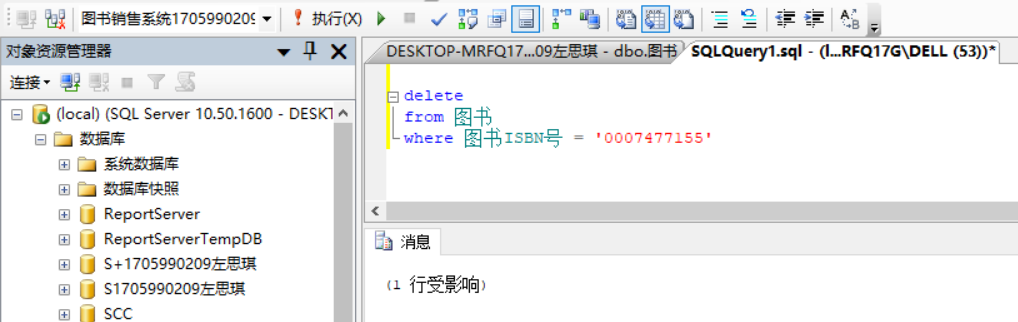
（1）SQL代码

delete

from 图书

where 图书ISBN号 = '0007477155'

（2）执行结果



4、修改图书记录

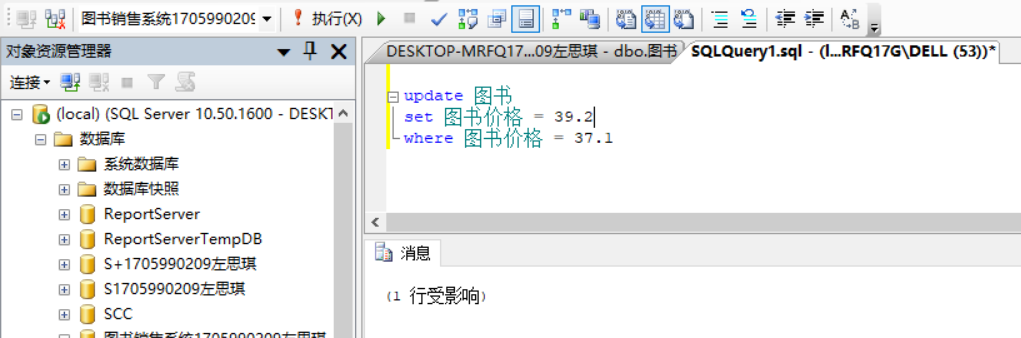
（1）SQL代码

update 图书

set 图书价格 = 39.2

where 图书价格 = 37.1

（2）执行结果



（二）用户查看并更改用户信息模块

1、查询用户信息

（1）测试数据



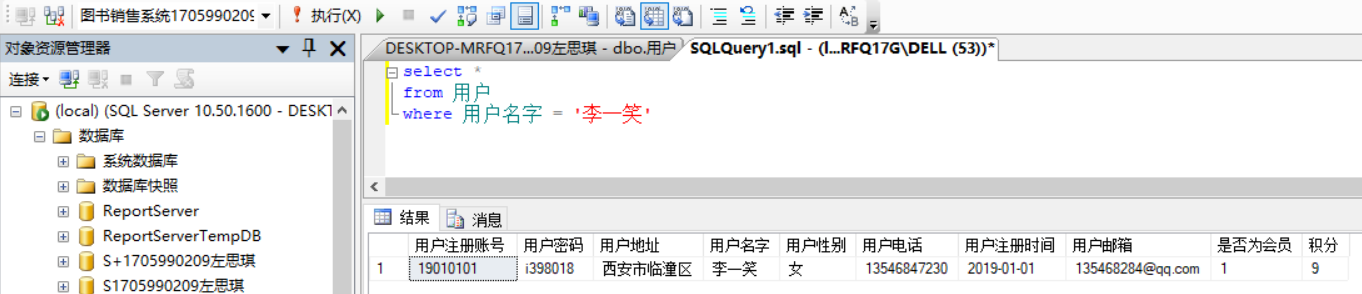
（2）SQL代码

select \*

from 用户

where 用户名字 = '李一笑'

（3）执行结果



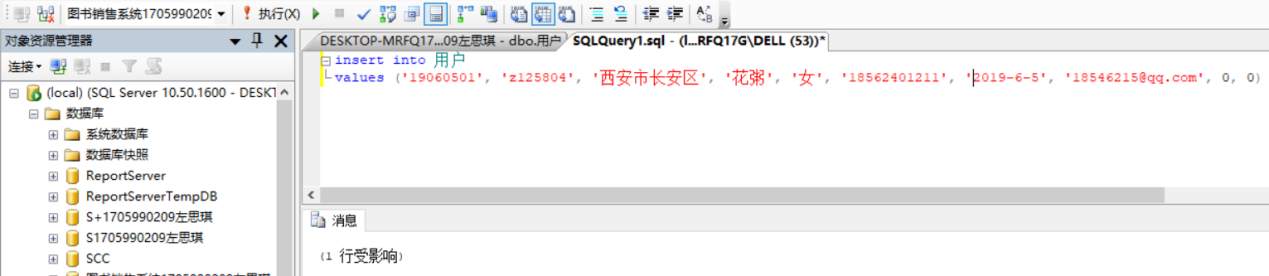
2、插入用户信息

（1）SQL代码

insert into 用户

values ('19060501', 'z125804', '西安市长安区', '花粥', '女', '18562401211', '2019-6-5', '18546215@qq.com', 0, 0)

（2）执行结果





3、修改用户记录

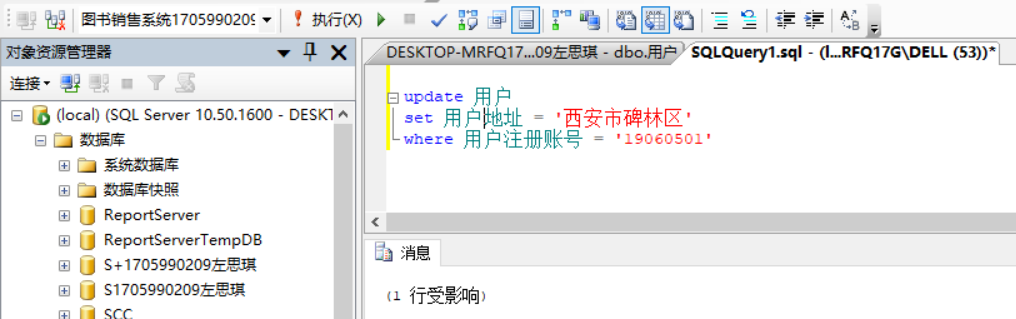
（1）SQL代码

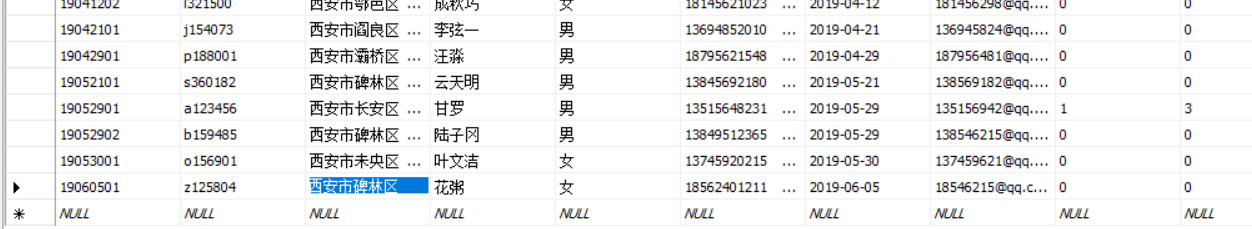
update 用户

set 用户地址 = '西安市碑林区'

where 用户注册账号 = '19060501'

（2）执行结果

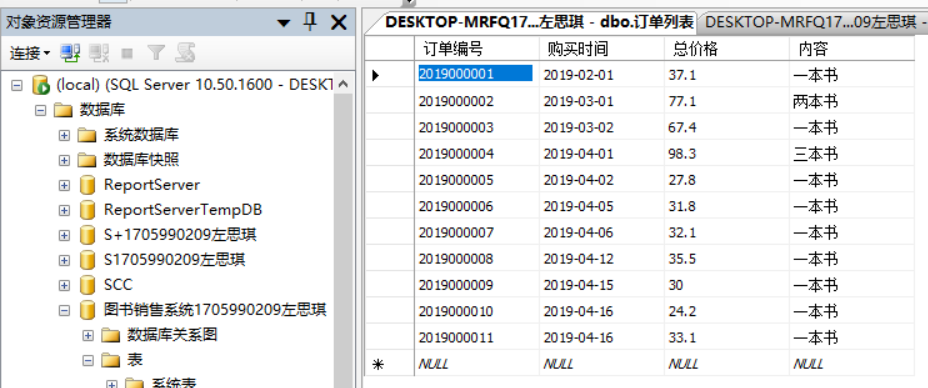


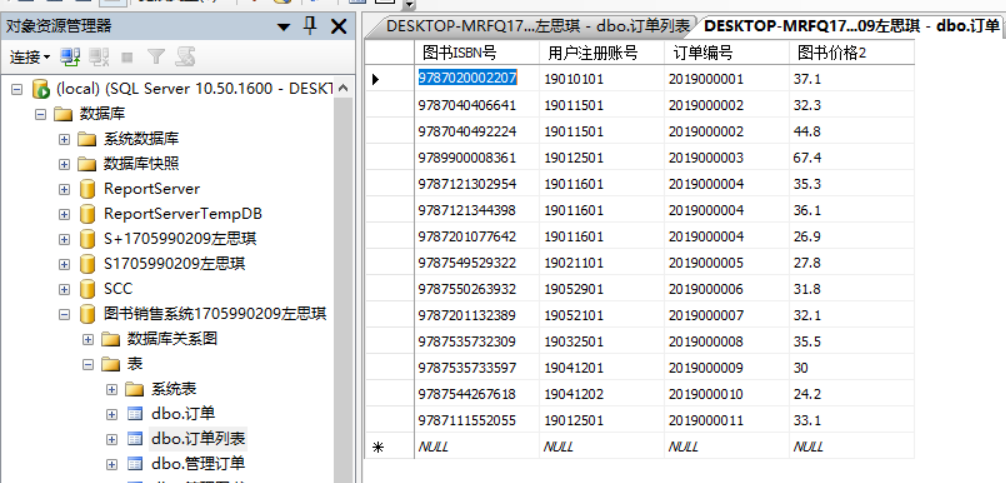


（三）订单的创建

1、查询订单

（1）测试数据





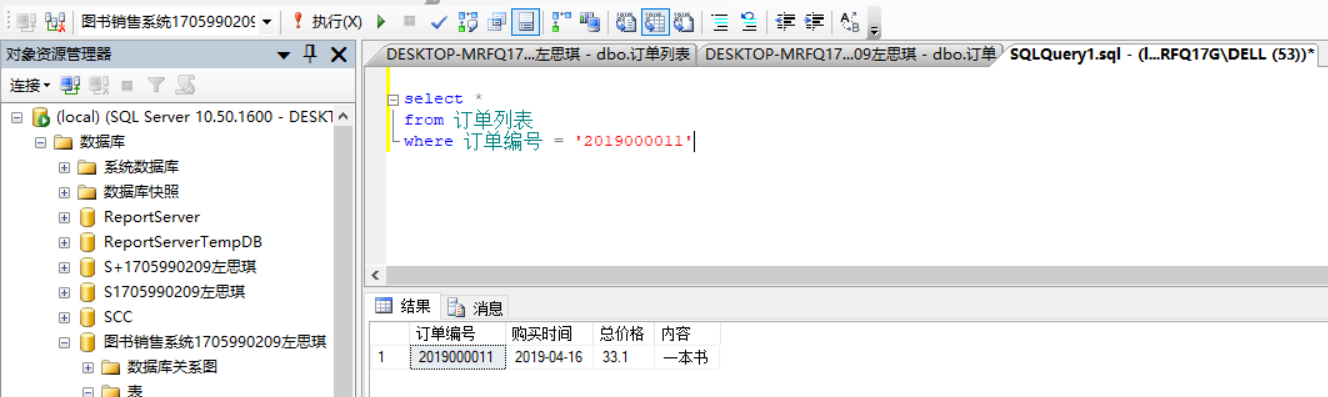
（2）SQL代码

select \*

from 订单列表

where 订单编号 = '2019000011'

（3）执行结果



2、插入订单

（1）SQL代码

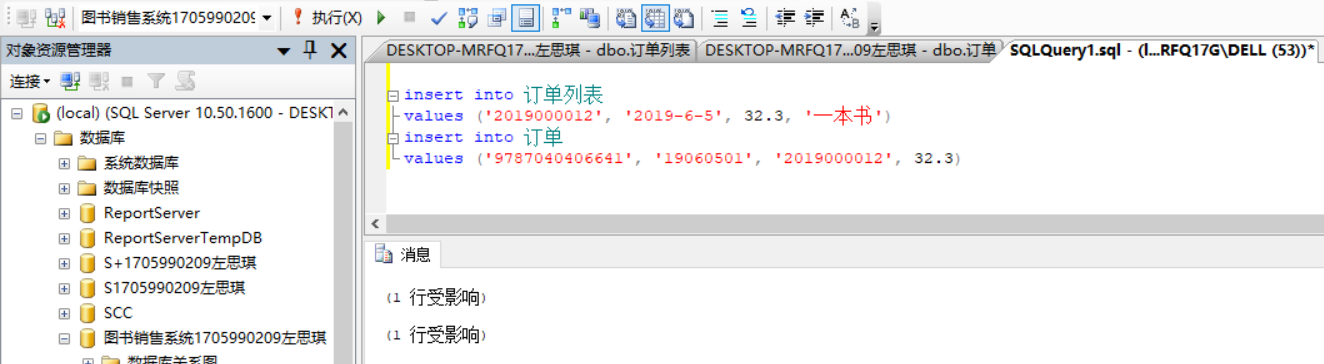
insert into 订单列表

values ('2019000012', '2019-6-5', 32.3, '一本书')

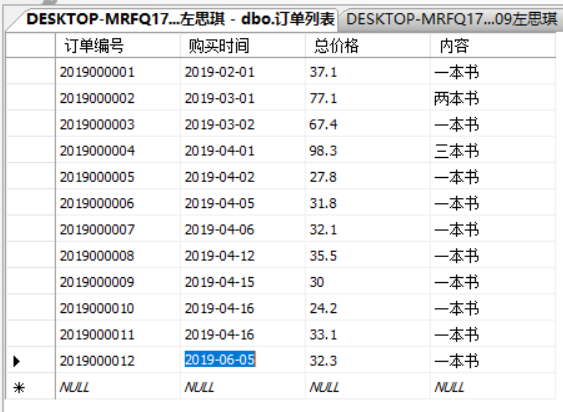
insert into 订单

values ('9787040406641', '19060501', '2019000012', 32.3)

（2）执行结果







（四）视图、索引、存储过程和触发器测试

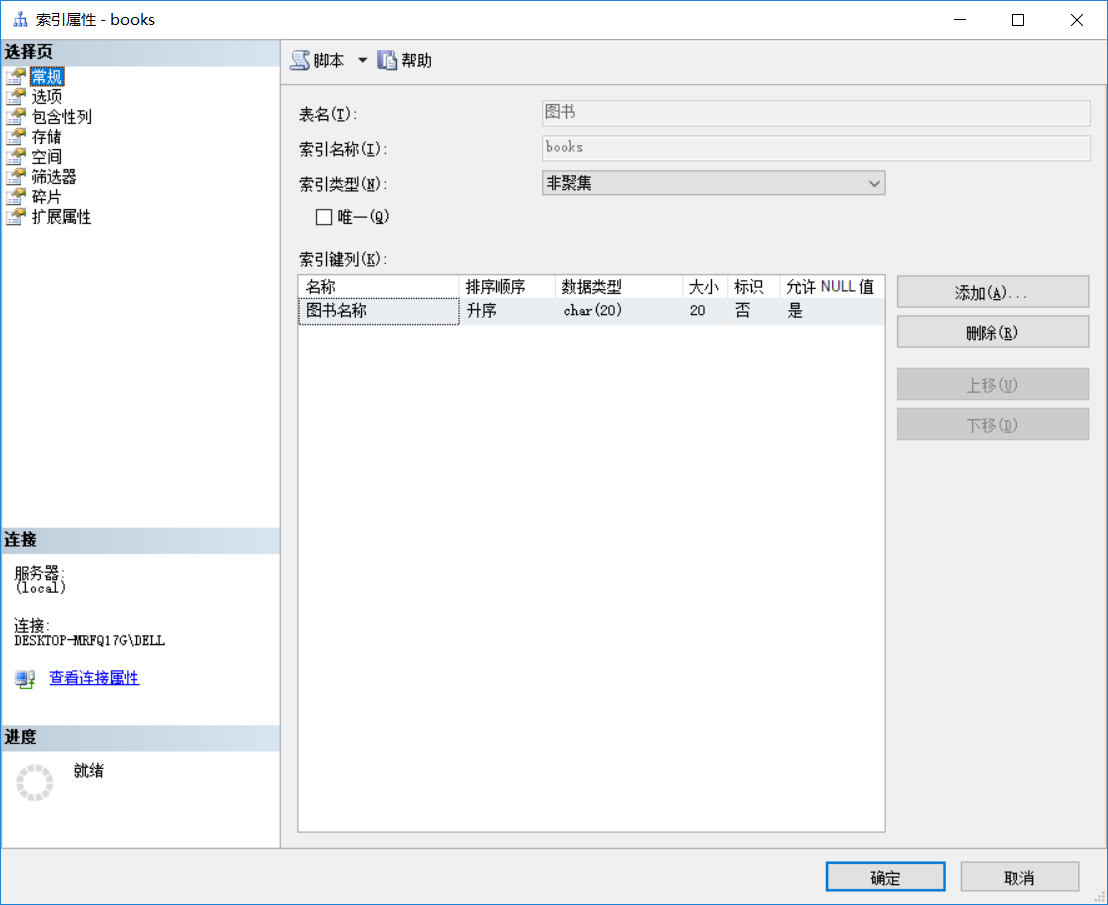
1、book视图

查询图书具体信息，创建的视图如下。



2、books索引

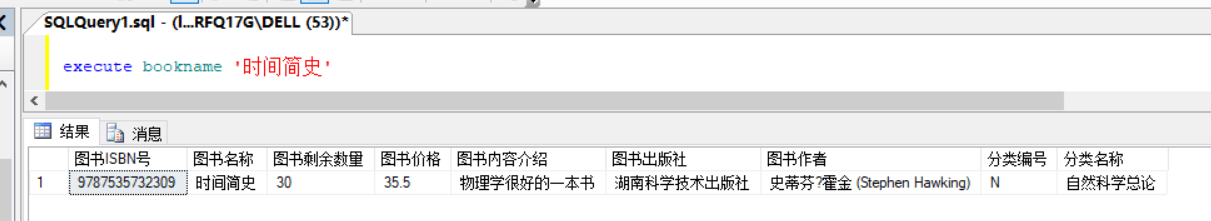
在图书表获得图书存货大于10的图书名称的非聚集索引。



3、bookname存储过程

输入图书名称得到该图书的所有信息。

execute bookname '时间简史'

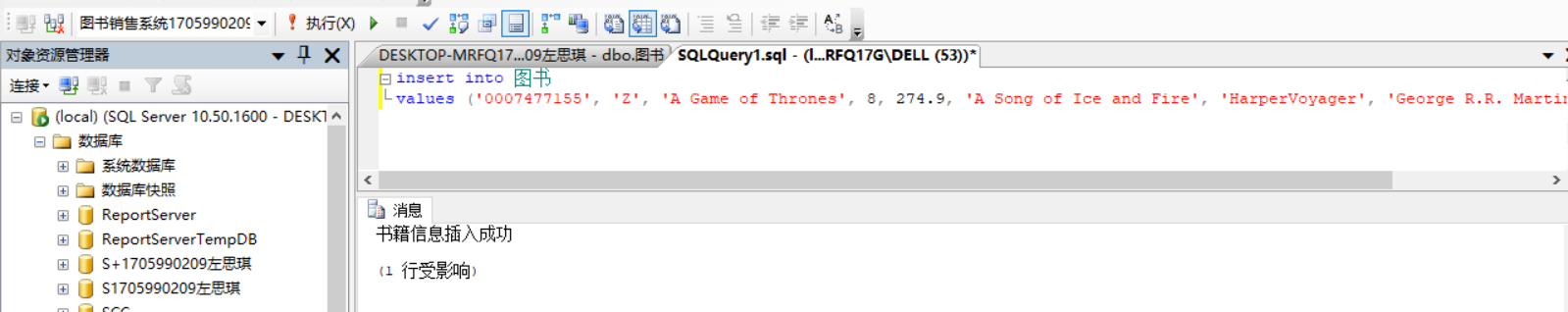


4、book\_in触发器

当向图书中插入信息时打印出“图书插入成功”。

insert into 图书

values ('0007477155', 'Z', 'A Game of Thrones', 8, 274.9, 'A Song of Ice and Fire', 'HarperVoyager', 'George R.R. Martin')





**六、应用程序设计**

（一）动态页面网页设计

本部分可视化界面设计使用Dreamweaver CS6软件进行设计实现，使用ASP作为脚本环境，链接SQL server数据库。

1、登录界面设计

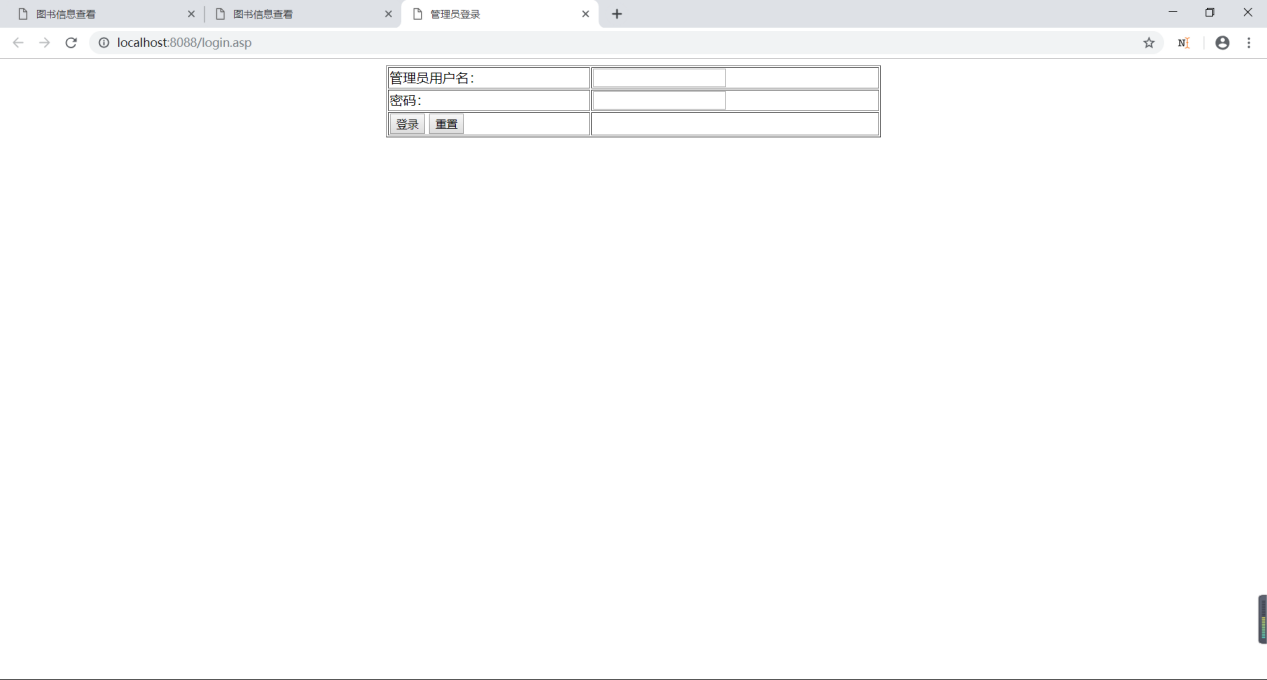
使用Dreamweaver CS6设计界面，利用软件在制作的网页中插入数据对象从而生成动态页面。

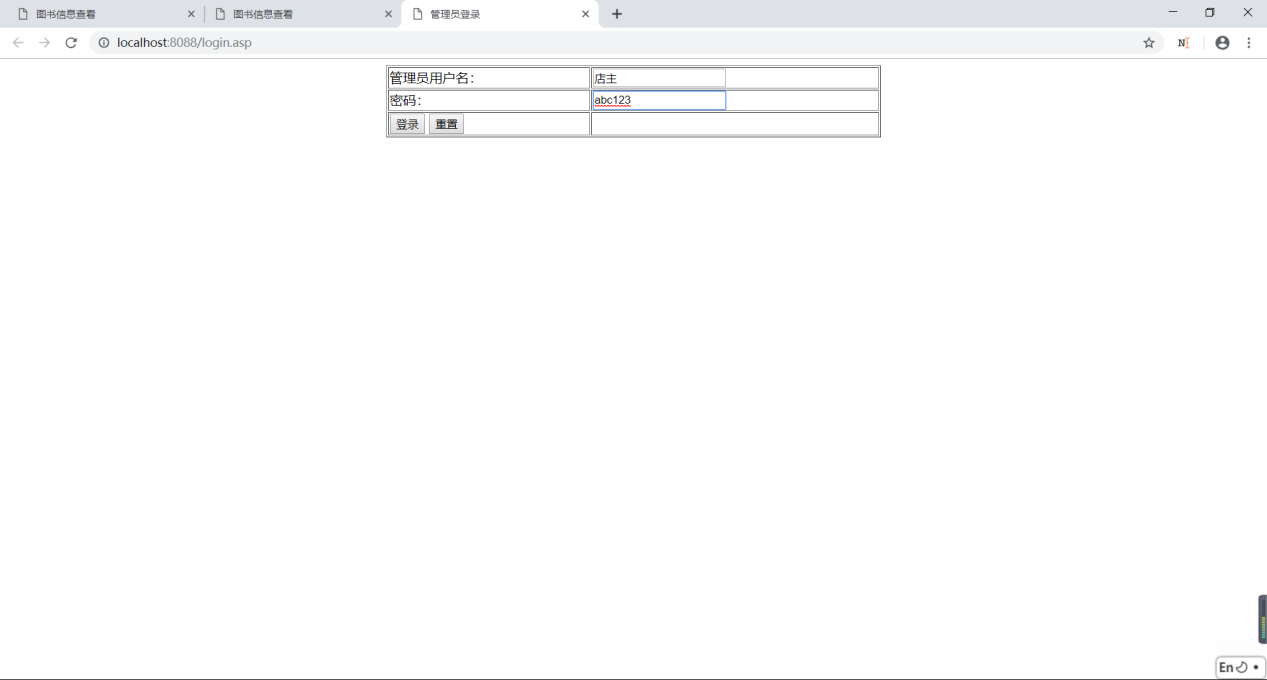
2、数据库连接实现

利用Dreamweaver CS6建立站点并进行连接。

（二）管理员登录模块

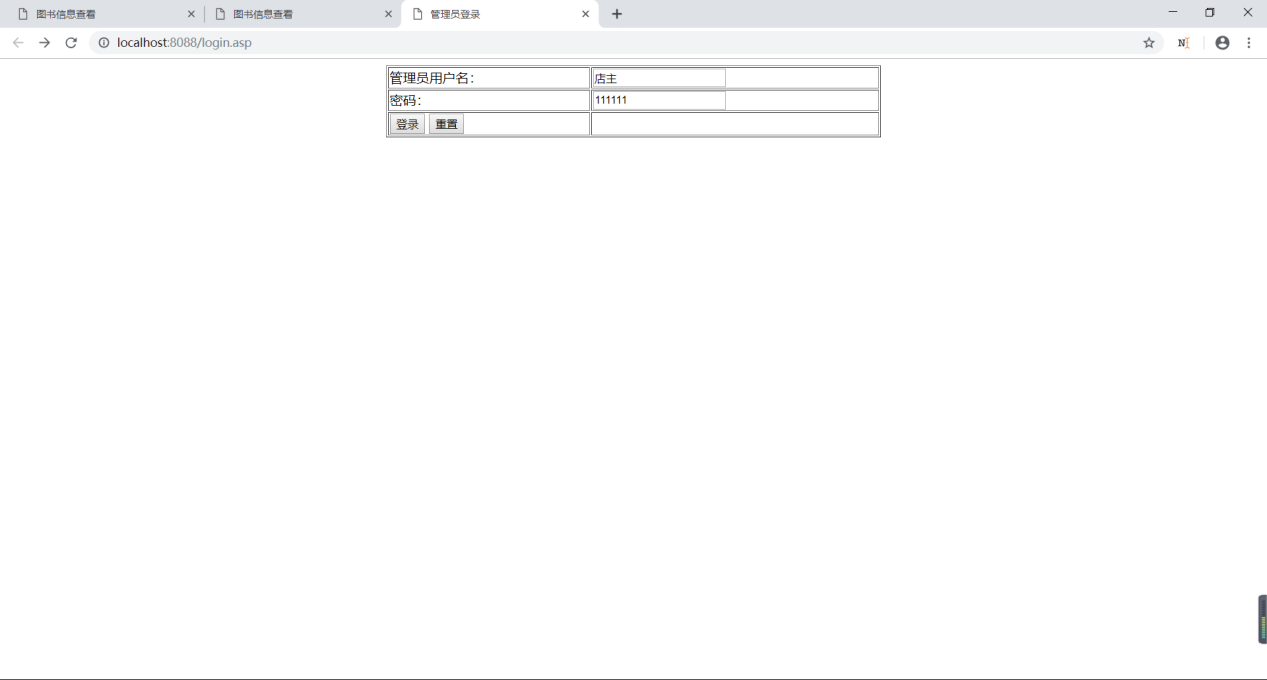
输入正确用户名与密码，登入成功进入图书表查看模块如下：

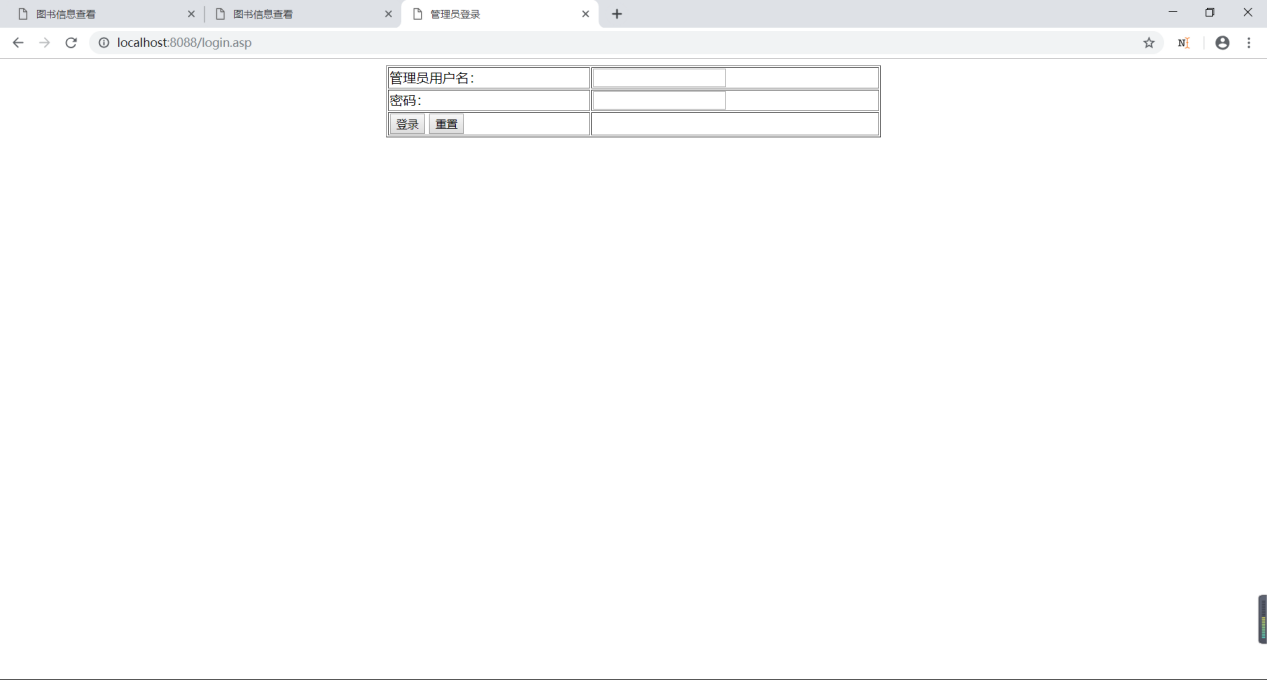






输入用户名或密码错误，则登录失败，重新回到登录界面：

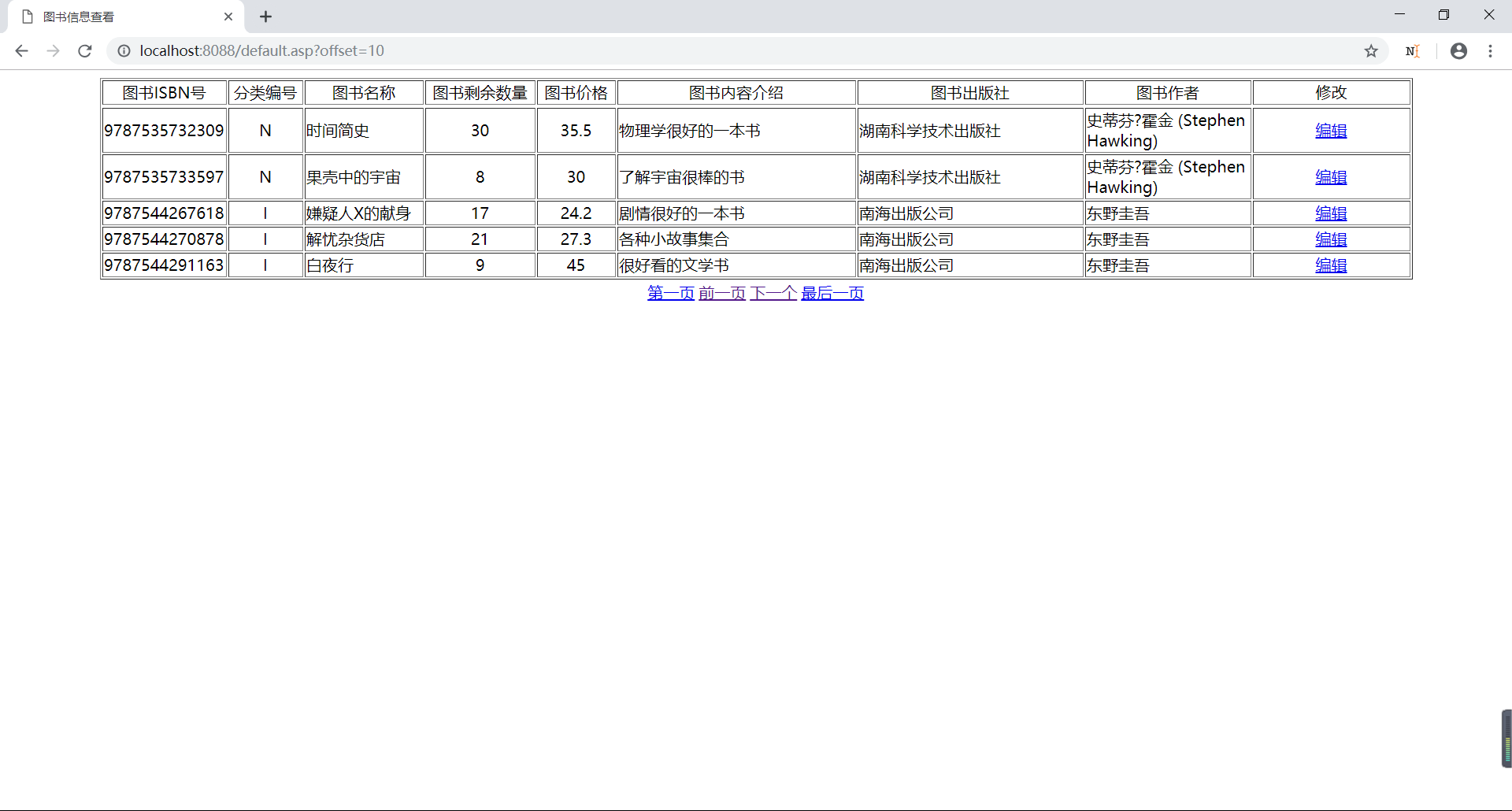




（三）图书表查看模块

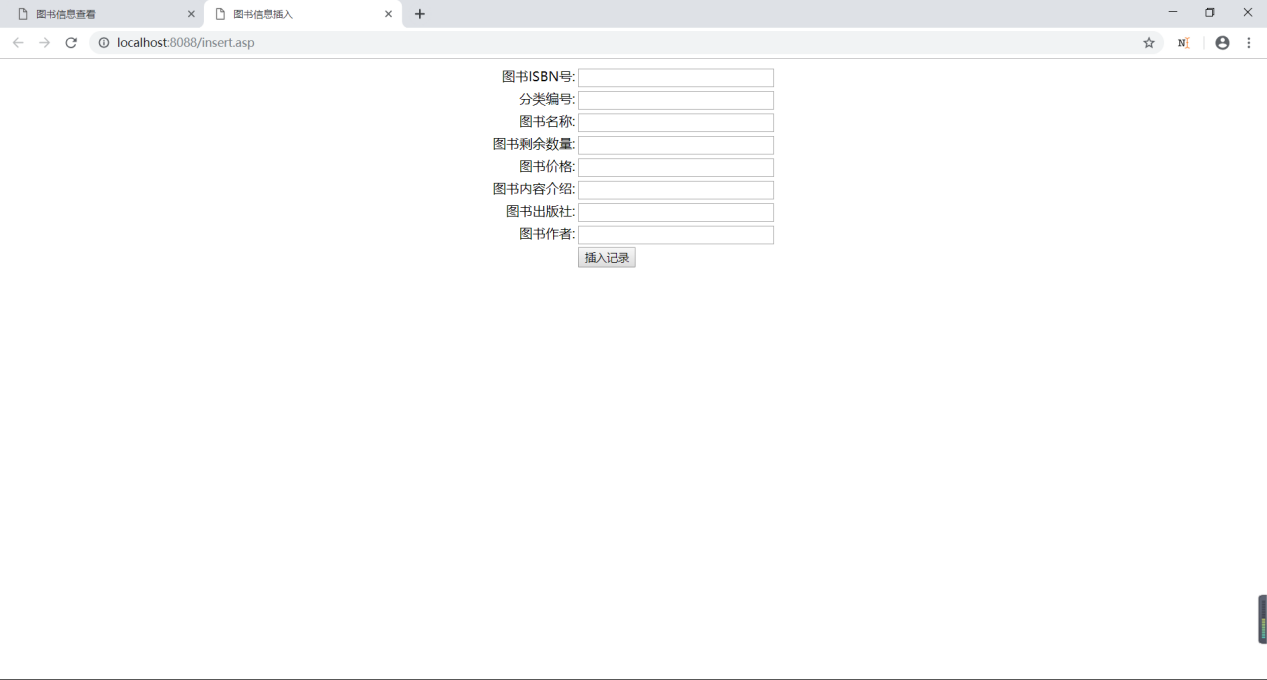


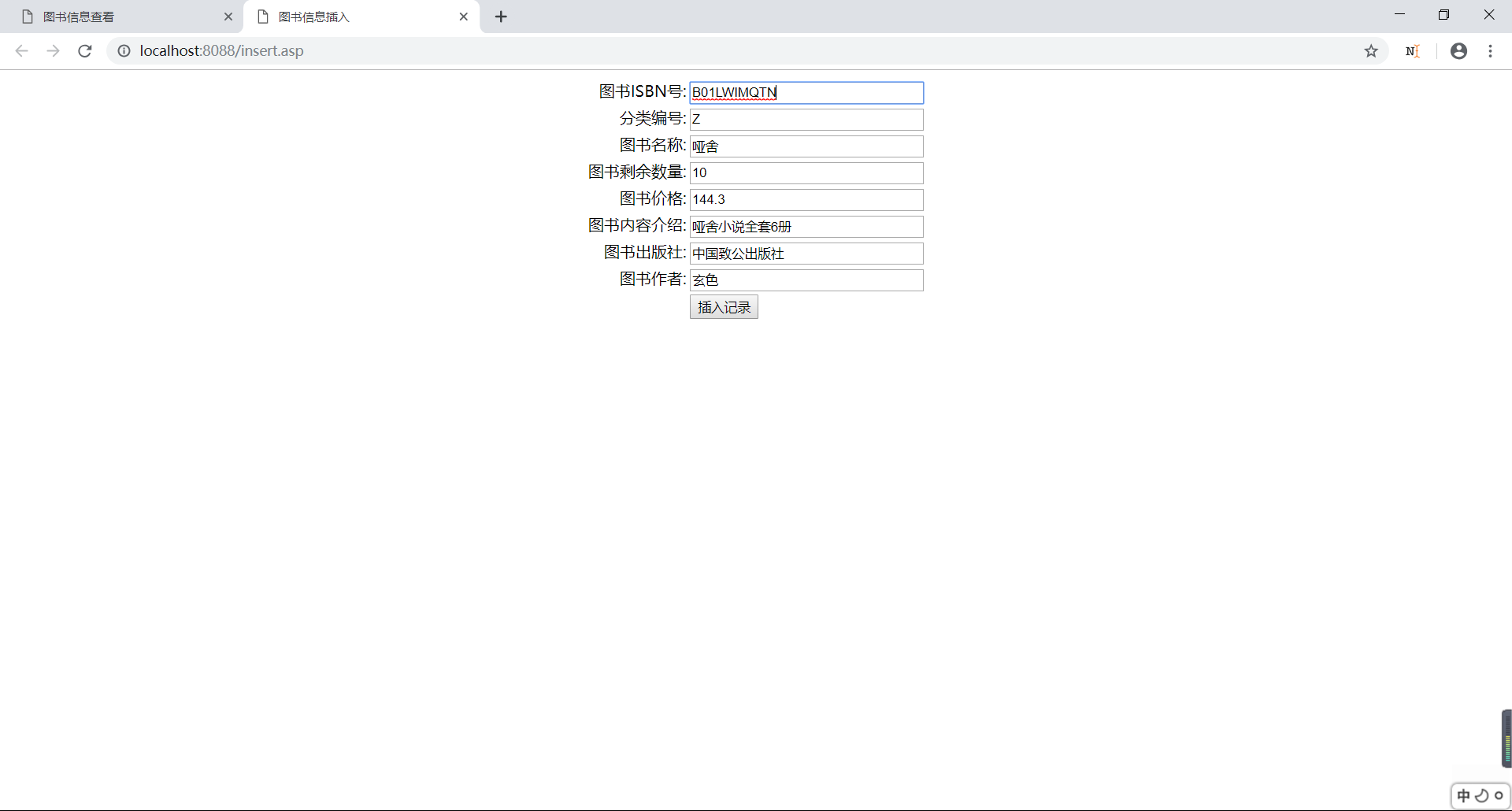


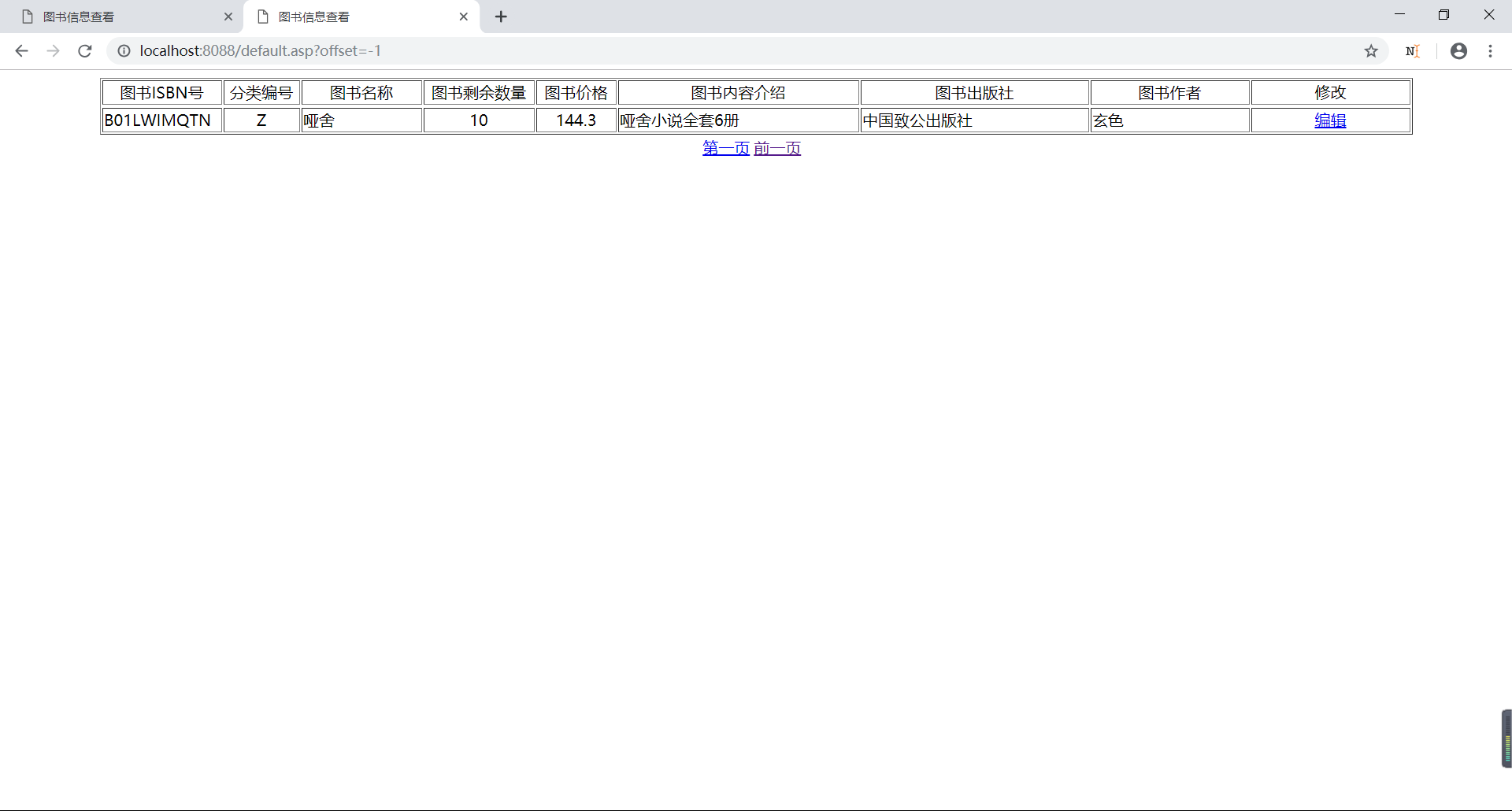




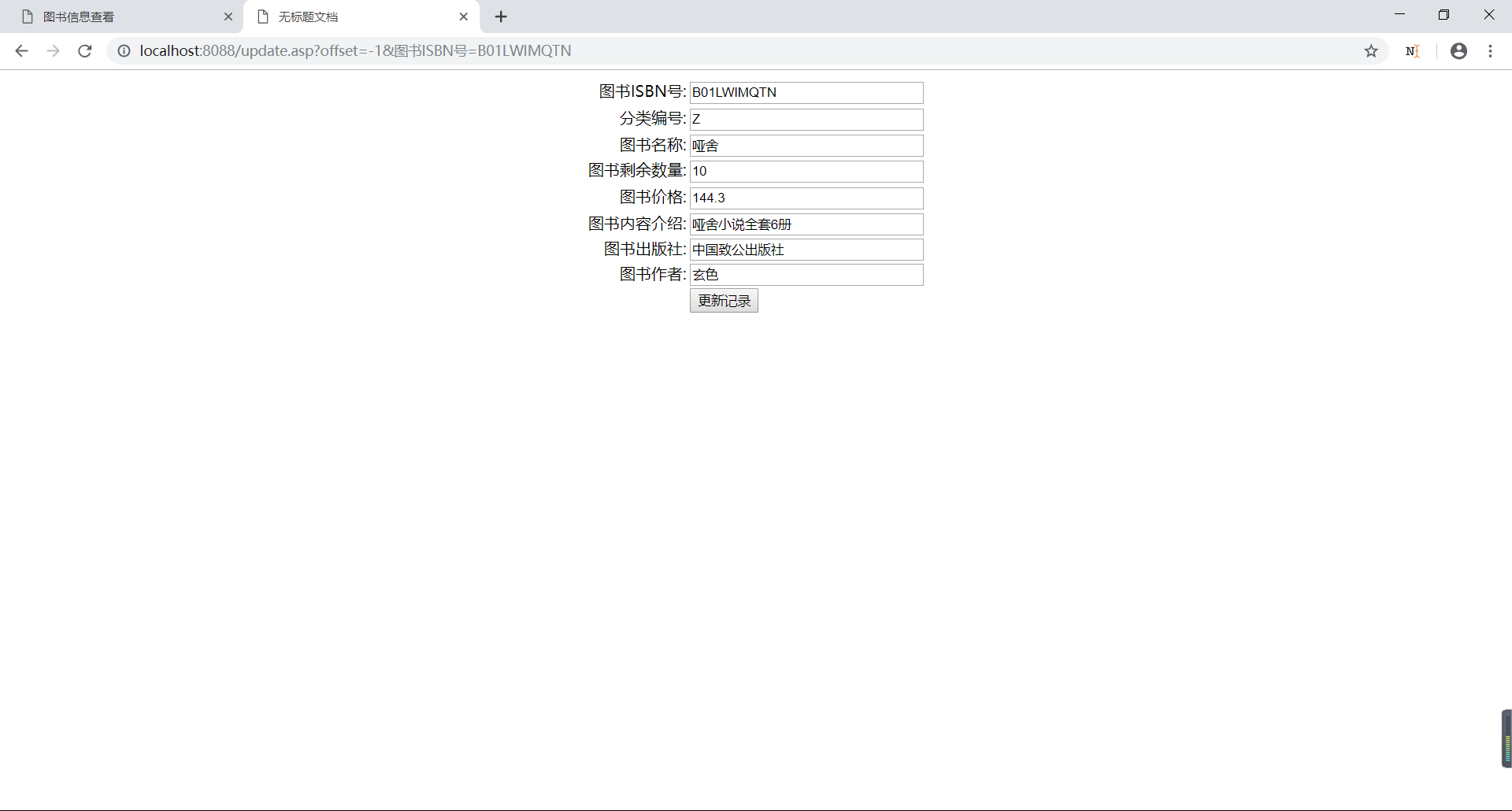
（四）图书表插入记录







（五）图书表内容修改





**七、设计总结**

通过本次课程设计，让我在数据库设计这一方面学会了很多东西，从需求分析到概念结构设计再到逻辑结构设计，然后进行物理实现，一直到最后数据库的建立，然后进行调试并维护运行，这期间让我更加充分的体会到了数据库的原子性、一致性、隔离性、持续性/永久性的四大特征，也更加深刻地明白了只有从实践中才能真正掌握知识，真正学会知识并且更好的去应用。

首先对于问题进行需求分析，这是一个很重要的过程，开始由于对于需求的分析不够充分，导致之后进行绘制实体之间的联系时发现了很多问题，之后重新对于题意进行了更加充分地分析，于是最终确定了系统中的实体以及它们的属性。

在这次的课程设计中，第一次接触到PowerDesigner软件，在建立CDM图的时候遇到了不少困难。首先是对实体之间的关系的确定，两个实体之间究竟是一对一还是一对多或是多对多的关系，后来通过老师的详细讲解，对实体之间的联系有了更深入的了解，并且成功的利用PowerDesigner建立出了概念设计模型，即是CDM图。然后利用PowerDesigner自带的生成逻辑模型的功能，成功生成了PDM图。

在学习PowerDesigner的过程中，发现它可以直接将逻辑模型设计图生成为建立表的SQL语句，充分感受到了这个软件的强大，但是直接生成数据库建表语句并不是一个很简单的过程，它首先需要配置数据源，并且连接数据库，当以上过程都正确时才可以生成正确的SQL代码，但是生成的代码并不能直接当做SQL sever的查询语句进行运行，里面会因为一些语句的版本是否支持等问题有一些报错，通过将导出的SQL代码进行一定的更改之后，才得以成功建立出数据库的各个表。

在测试数据库的过程中，进行了各种增、删、查、改的操作，于是让我对SQL的语言有了更熟练的运用。同时在建立视图、索引、触发器和存储过程的过程中，对于这些东西的作用以及实现方法有了更加清晰的了解，也有了更加熟练的运用。

在进行应用程序设计的过程中，由于能力的原因，采用了动态网页的制作来进行实现。再认真阅读老师的教学幻灯片之后，通过自己一次又一次的调试，一遍又一遍的学习，以及查阅了相关的博客等，终于成功的建立出了一个简单的动态界面。

动态界面的建立是使用ASP作为脚本环境，链接SQL server数据库，同时用Adobe公司出产的Dreamweaver CS6软件进行制作。这是第一次尝试制作动态界面，在这个过程中我学会了如何启动与更改Windows功能，对于控制面板有了更多的了解，配置了Windows中的Internet服务，启用了IIS和与ASP相关的多个服务，然后建立了一个网页，利用DW建立了相应的站点，然后在本机系统DSN中建立了一个32位数据源，链接本系统的数据库，通过这个数据源链接入DW制作的网页中。虽然这个界面做的并不好看，但是在制作的过程中让我学到了很多新的知识，让我对动态页面有了更加深入的了解。

总的来说，通过这次的课程设计让我发现了我的很多知识盲区，并且通过实践中的学习让我的一些能力得到了很好地提升，这次的课程设计对我以后的专业有很大的帮助，也是在这次实践中总结出了一些经验，努力逐个击破自己的弱点，让自己的能力有更好的提升。在之后的学习与实践过程中要更加努力。