**结课报告**

课程名称 面向对象程序设计

学 院 经济管理学院

班级学号 2004070128

学生姓名 陈瑞棋

指导教师 吴晓艳

2022年5月15日

项目：图书借阅系统

# 子项目：需求分析

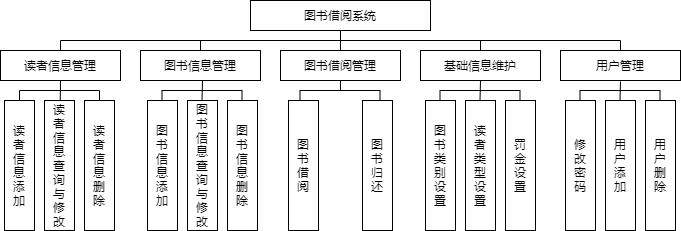
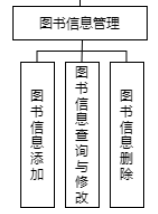


图1 功能模块图

# 图书借阅系统有以下四大模块：读者信息管理、图书信息管理、图书借阅管理、用户信息管理、类型信息管理，我们接下来对每一模块的具体功能进行分析需求。

第一模块：图书信息管理，功能如图1所示



**图1 图书信息管理模块功能展示**

图书信息管理是图书借阅系统最核心的功能，因为图书是最关键的要素， 在系统中有着所有图书的相关信息。

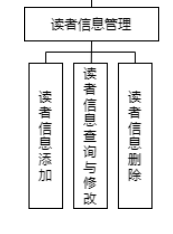
图书增、删、改、查功能

首先我们得能知道系统中有什么书，这本书的相关信息，知道了图书的信息这样我们才能根据读者的需求借书，所以这一模块就需要一个图书查询功能。查询要能查询所有书籍的关键信息，同时也能在知道了书名或者书的ISBN进行指定查找。

如果图书馆新进了一批书籍，我们得把这些书籍给登记进入系统中，这里模块就需要一个图书信息增加功能，我们需要输入图书的关键信息然后添加信息到我们的后台系统中。

如果输入的图书信息有误，我们需要一个图书信息修改功能，如果有的图书被读者弄丢了，最后我们要把这本书的记录在图书系统中删除，所以需要一个图书删除功能。

第二模块：读者信息管理，功能如图2所示

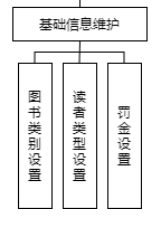


**图2 读者信息管理模块功能展示**

图书借阅系统中读者信息也是很重要的，我们需要对读者的信息进行操作。

如果是第一次来借书，那么必须要在系统中创建读者信息， 在根据读者id进行借阅书籍，所以需要一个读者信息增加功能。系统管理员给长期不借阅书籍的读者注销读者用户，需要读者信息删除功能。系统管理员要知道当前读者的所有信息或者指定信息，此模块需要一个查询读者信息的功能，还包括查询所有和根据id指定查询。

第三模块：类型信息管理，功能如图3所示

**图3 类型信息管理模块功能展示**

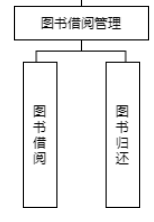
读者类型信息管理

我们借阅书籍，因为读者的身份或者类别不同，读者借阅书籍肯定有不同的规则，参照一般大学图书馆的规则，学生能最多借3本书，3个月内归还，老师最多能借5本书，5个月内归还，所以因为读者身份不同借阅书籍的规则也不同，所以我们要对读者类型信息进行分类，用户可以对读者的类型信息进行增删改查操作。

图书类型信息管理

与之对应的图书类别信息也是如此，真正图书系统中书籍一定是很多的，所以分类就很有必要，所以我们也要对书籍的类别进行管理，对书籍的类别信息进行增删改查操作。

第四模块：借阅信息管理，功能如图4所示

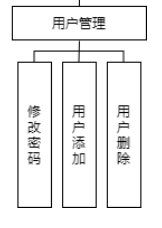
**图4 借阅信息管理模块功能展示**

图书借阅与归还也是系统中较为关键的业务，读者需要根据自己的需求借阅自己想要阅读的图书，此时就需要图书的借阅功能，功能展开，我们需要最为关键的读者信息、借阅的图书信息，以及借书的日期，然后增加到借书的数据库中。

当读者阅读完这本书或者到达了阅读的最长期限，用户必须要归还图书，当前模块需要一个归还功能，根据借阅图书的ISBN或者图书id进行归还，具体就是在借阅的书籍列表中进行删除，把书籍的借阅信息删除。

我们同时也会在借阅和归还的面板中加入显示所有借阅书籍的功能，让读者知道那本书被借阅了，书被借走了没法再借了(默认系统中每本书只有一本)。这里就需要一个查询功能，直接查询数据库中所有被借阅的图书信息以及相关的读者信息。

第五模块：用户信息管理，功能如图5所示

**图5 用户信息管理模块功能展示**

用户信息对于图书系统是至关重要的，为什么重要呢?它关乎与我们是否能够成功登入该系统中。首先登陆需要一个具体的登录功能，用户输入用户名跟或者密码，然后程序在后台中查询数据库进行比对操作，比对成功才能登入

如果用户第一次登陆，那么没有注册用户名，需要我们设置一个注册功能，就是添加用户操作，用户自己设置用户名和密码(前提：不能和数据库中的用户名一致)，增加成功我们就用此用户的信息进行登陆操作。如果该用户想要修改密码，那么就需要写一个修改密码的功能，先输入原密码，然后输入新密码，最后确认新密码。

如果系统管理员想要删除某个用户(长期不登录，或者登陆异常)的用户信息，那么就需要在后台中写一个删除用户的功能，根据用户名进行删除(前提：数据库中对应字段设为唯一)。同时系统管理员想要查找用户的信息，需要写一个查询功能进行查询所有。

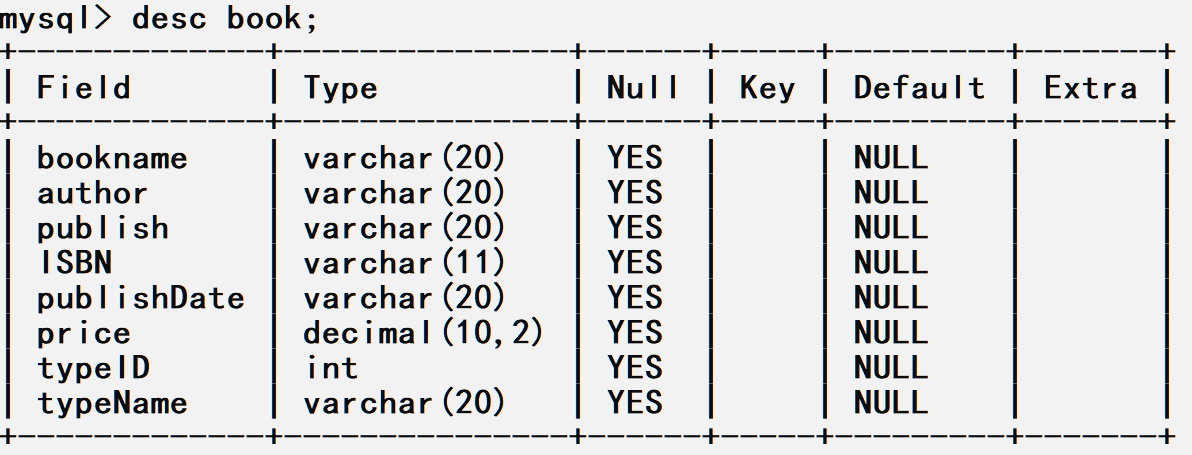
# 子项目：数据库设计

图书信息表用于存储图书基本信息，如表1所示

表 1图书信息表book表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 字段描述 |
| Bookname  Author  Publish  ISBN  pulishDate  Price  TypeId  TypeName | Varchar  Varchar  Varchar  Varchar  Varchar  Decimal(10,2)  Int  Varchar | 20  20  20  11  20  20 | 图书名称  图书作者  图书出版社  图书ISBN  图书出版日期  图书价格  图书类别ID  图书类别名称 |

数据库中具体的表相关信息

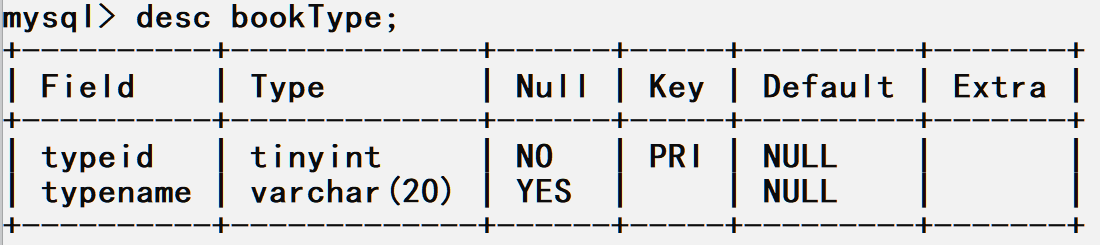


图书类别信息表用于存储图书类别基本信息，如表2所示

表 2图书类别表bookType表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 字段描述 |
| TypeId  TypeName | INT  VARCHAR | 20 | 图书类别ID  图书类别名称 |

数据库中具体的表相关信息

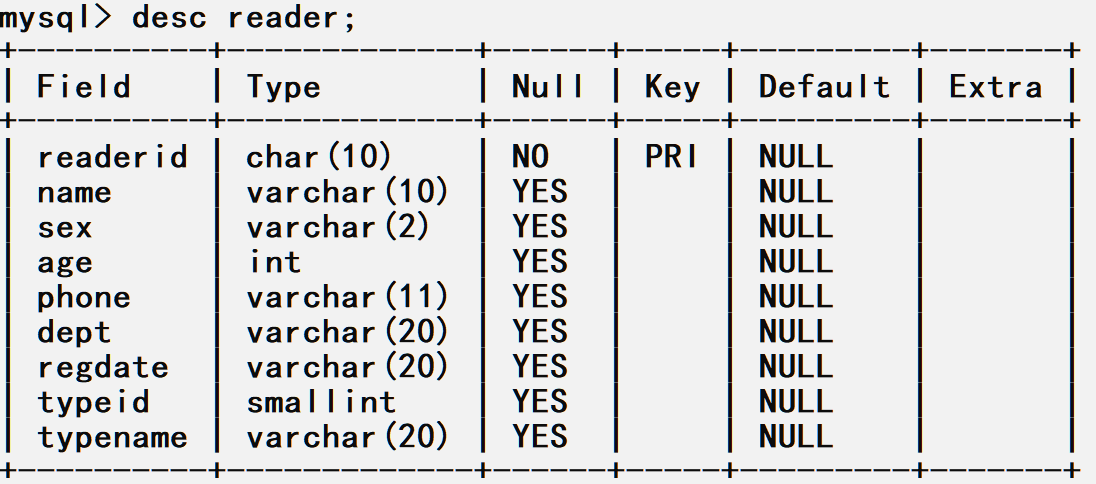


读者信息表用于存储读者基本信息，如表3所示

表3读者reader表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 字段描述 |
| ReaderId  Name  Sex  Age  Phone  Dept  Regdate  Typeid  Typename | VARCHAR  VARCHAR  VARCHAR  Int  VARCHAR  VARCHAR  VARCHAR  Int  VARCHAR | 20  20  20  20  20  20  20 | 读者ID  读者姓名  读者性别  读者年龄  读者手机号  读者所在部门(学院)  读者信息注册时间  读者类别ID  读者类别名称 |

数据库中具体的表相关信息

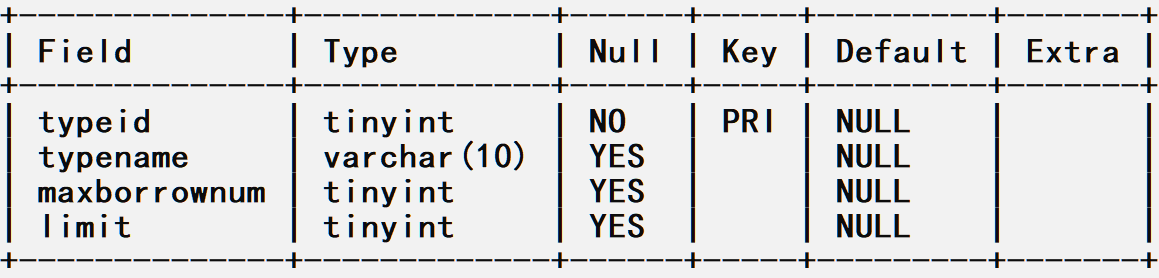


读者类别信息表用于存储读者基本信息，如表4所示

表 4读者类别表ReaderType表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 字段描述 |
| TypeId  TypeName  MaxBorrowbook  Limit | Int  Varchar  Int  Int | 20 | 读者类别ID  读者类别名称  该类读者最多能借数量  该类读者借书的最长期限 |

数据库中具体的表相关信息

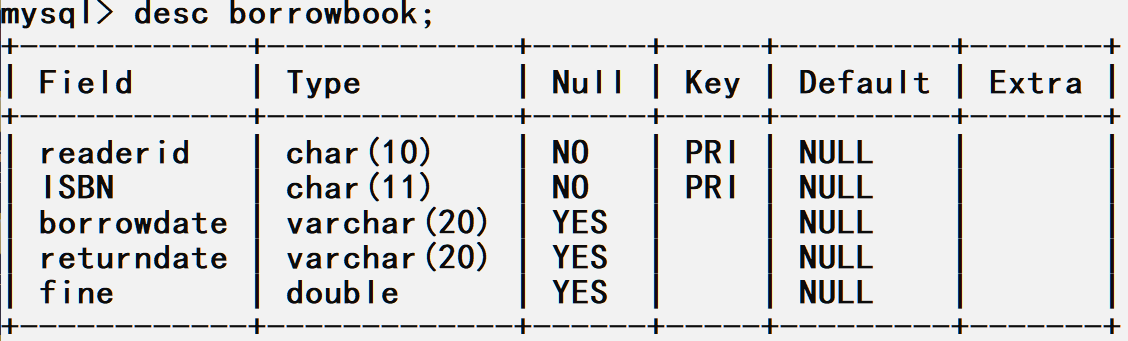


借阅图书信息表用于存储借阅的图书关键信息，如表5所示

表 5借阅图书表borrowBook表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 字段描述 |
| ReaderId  ISBN  borrowDate  returnDate  Fine | Varchar  Varchar  Varchar  Varchar  Double | 20  20  20  20 | 借这本书的读者ID  借阅书籍的ISBN  借书的日期  还书的日期  超期的罚金 |

数据库中具体的表相关信息

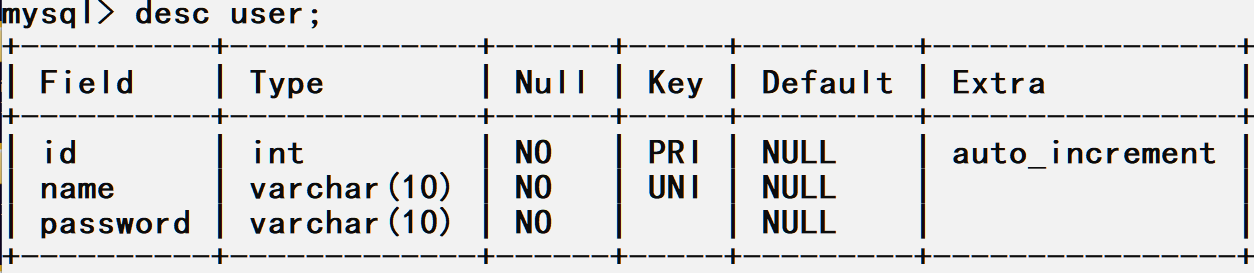


用户表用于存储系统中的用户信息，如表6所示

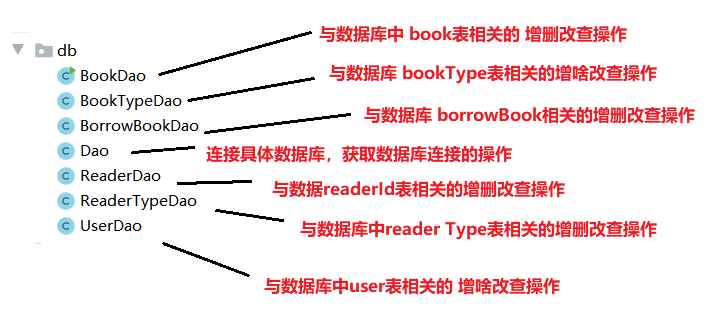
表 6用户表user表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 数据类型 | 长度 | 字段描述 |
| Id  Name  Password | Int  Varchar  Varchar | 20  20 | 用户id(自增主键)  用户名  用户密码 |

数据库中具体的表相关信息



# 子项目：图书借阅系统中类的应用

数据库操作具体表相关的类，如图7所示

# 

**图7操作数据库的JDBC类**

具体表中数据对应的实体类(pojo)，如图8所示



**图8 数据表中对应的实体类**

Swing实现的Java可视化界面相关的类，如图9所示



**图9 Swing实现的可视化界面的类**

实现前端界面与后端数据交互的类(事件)，如图10 所示



**图10 前端界面实现了具体事件的类**

# 子项目：图书借阅系统界面设计与实现

1. 创建登录界面

用户在文本框中输入用户名和密码,成功登录后进入图书系统的主界面，如图11所示

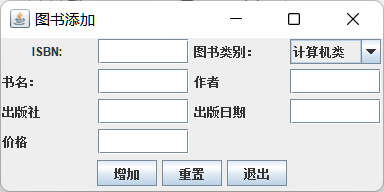


**图11 登陆界面**

package com.bbm.staticview;  
import javax.swing.\*;  
public class Login extends JFrame {  
 private JPanel myPanel;  
 private JLabel labName,labPassword;  
 private JTextField txtName;  
 private JPasswordField txtPassword;  
 private JButton btnConfirm,btnReset;  
 public Login(String name){  
 super(name);*//框架类设标题* setSize(250,150);  
 setLocationRelativeTo(null);  
 myPanel=new JPanel();  
 setContentPane(myPanel);  
 labName=new JLabel("用户名：");   
 labPassword=new JLabel("密 码：");  
 txtName=new JTextField(12);  
 txtPassword=new JPasswordField(12);  
 txtPassword.setEchoChar('\*');  
 btnConfirm=new JButton("登录");  
 btnReset=new JButton("重置");  
 myPanel.add(labName);  
 myPanel.add(txtName);  
 myPanel.add(labPassword);  
 myPanel.add(txtPassword);  
 myPanel.add(btnConfirm);  
 myPanel.add(btnReset);   
 setVisible(true);   
 }  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new Login("登录");  
 }  
  
}

1. 创建图书增加界面，如图12所示

输入对应标签的信息，点击添加，成功增加一条图书信息



**图12 图书增加界面**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class BookAdd extends JFrame {  
 private JPanel panel,bookPanel,btnPanel;  
 private JLabel labISBN,labType,labName,labAuthor,  
 labPublish,labPublishDate,labPrice;  
 private JTextField txtISBN,txtName,txtAuthor,  
 txtPublish,txtPublishDate,txtPrice;  
 JComboBox cmbBookType;*//组合框* private JButton btnAdd,btnReset,btnExit;  
 public BookAdd(String s)  
 {  
 super(s);  
 setSize(400,200);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
 *//图书面板的信息* GridLayout grid1=new GridLayout(4,4);*//网格布局* grid1.setHgap(5);  
 grid1.setVgap(5);  
 bookPanel=new JPanel(grid1);  
 labISBN=new JLabel("ISBN:");  
 labISBN.setHorizontalAlignment(SwingConstants.*CENTER*);*//居中* txtISBN=new JTextField(15);  
 labType=new JLabel("图书类别：");  
 cmbBookType=new JComboBox();  
 cmbBookType.addItem("计算机类");  
 cmbBookType.addItem("外语类");  
 labName=new JLabel("书名：");  
 txtName=new JTextField(12);  
 labAuthor=new JLabel("作者");  
 txtAuthor=new JTextField(12);  
 labPublish=new JLabel("出版社");  
 txtPublish=new JTextField();  
 labPublishDate=new JLabel("出版日期");  
 txtPublishDate=new JTextField(12);  
 labPrice=new JLabel("价格");  
 txtPrice=new JTextField(12);  
 *//其余类似，作者，出版社，出版社日期，价格，课后大家补充。* bookPanel.add(labISBN);  
 bookPanel.add(txtISBN);  
 bookPanel.add(labType);  
 bookPanel.add(cmbBookType);  
 bookPanel.add(labName);  
 bookPanel.add(txtName);  
 bookPanel.add(labAuthor);  
 bookPanel.add(txtAuthor);  
 bookPanel.add(labPublish);  
 bookPanel.add(txtPublish);  
 bookPanel.add(labPublishDate);  
 bookPanel.add(txtPublishDate);  
 bookPanel.add(labPrice);  
 bookPanel.add(txtPrice);  
 *//将各组件加入到面板* panel.add(bookPanel,BorderLayout.*CENTER*);  
 btnPanel=new JPanel();  
 btnAdd=new JButton("增加");  
 btnReset=new JButton("重置");  
 btnExit=new JButton("退出");  
 btnPanel.add(btnAdd);  
 btnPanel.add(btnReset);  
 btnPanel.add(btnExit);  
 panel.add(btnPanel,BorderLayout.*SOUTH*);   
   
 setVisible(true);   
 }  
 public static void main(String[] args) {  
 new BookAdd("图书添加");  
  
 }  
  
}

1. 创建图书修改与查询界面，如图13所示

在中间面板中显示所有图书信息，输入图书信息，可以实现删除与修改功能。

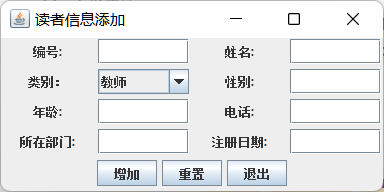


**图13 图书管理界面**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.Component;  
import java.awt.Dimension;  
import java.awt.Graphics;  
import java.awt.GridLayout;  
import java.awt.Insets;  
  
import javax.swing.\*;  
import javax.swing.border.Border;  
  
public class BookSelectModify extends JFrame {  
 private JPanel panel,selectConditionPane,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
 private JComboBox cmbChoice,cmbType;  
 private JTextField txtSelect,txtISBN,txtName,txtAuthor,  
 txtPublish,txtPublishDate,txtPrice;  
 private JLabel labISBN,labType,labName,labAuthor,  
 labPublish,labPublishDate,labPrice;  
 private JButton btnSelect,btnModify,btnDelete,btnExit;  
 private JTable table;  
 private JScrollPane scrollPane;  
   
 public BookSelectModify()  
 {  
 setTitle("图书管理");*//设置标题* setSize(500,500);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
 selectConditionPane=new JPanel();  
 cmbChoice=new JComboBox();  
 cmbChoice.addItem("全部");  
 cmbChoice.addItem("书名");  
 txtSelect=new JTextField(20);  
 selectConditionPane.add(cmbChoice);  
 selectConditionPane.add(txtSelect);  
 panel.add(selectConditionPane,BorderLayout.*NORTH*);  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel();  
 table=new JTable();  
 scrollPane=new JScrollPane(table);  
 scrollPane.setPreferredSize(new Dimension(400,240));*//设大小* selectResultPane.add(scrollPane);  
 bookPane=new JPanel(new GridLayout(4,4));  
 *//4行4列，需要创建每个组件，加到bookPane中，另外6个课后完成* labISBN=new JLabel("ISBN号");  
 labType=new JLabel("图书类型");  
 labName=new JLabel("书名");  
 labAuthor=new JLabel("作者");  
 labPublish=new JLabel("出版社");  
 labPublishDate=new JLabel("出版日期");  
 labPrice=new JLabel("价格");  
 txtISBN=new JTextField(8);  
 cmbType=new JComboBox();  
 cmbType.addItem("计算机类");  
 cmbType.addItem("管理类");  
 txtName=new JTextField(8);  
 txtAuthor=new JTextField(8);  
 txtPublish=new JTextField(8);  
 txtPublishDate=new JTextField(8);  
 txtPrice=new JTextField(8);  
 bookPane.add(labISBN);   
 bookPane.add(txtISBN);  
 bookPane.add(labType);   
 bookPane.add(cmbType);  
 bookPane.add(labName);   
 bookPane.add(txtName);  
 bookPane.add(labAuthor);   
 bookPane.add(txtAuthor);  
 bookPane.add(labPublish);   
 bookPane.add(txtPublish);  
 bookPane.add(labPublishDate);   
 bookPane.add(txtPublishDate);  
 bookPane.add(labPrice);   
 bookPane.add(txtPrice);  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
 btnPanel=new JPanel();  
 btnSelect=new JButton("查询");  
 btnModify=new JButton("修改");  
 btnDelete=new JButton("删除");  
 btnExit=new JButton("退出");  
 btnPanel.add(btnSelect);  
 btnPanel.add(btnModify);  
 btnPanel.add(btnDelete);  
 btnPanel.add(btnExit);  
 panel.add(btnPanel,BorderLayout.*SOUTH*);   
   
 setVisible(true);  
 }   
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new BookSelectModify();  
 }  
  
}

1. 读者信息增加界面，如图14所示

输入对应的信息，点击增加，实现添加信息的功能。

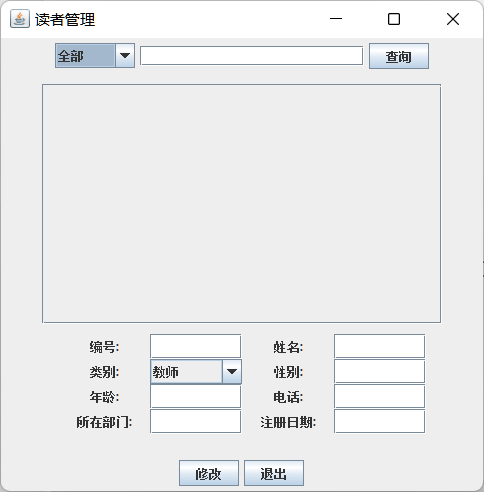


**图14 读者信息增加界面**

package com.bbm.staticview;  
  
import javax.swing.\*;  
import java.awt.\*;  
  
public class ReaderAdd extends JFrame {  
 private JPanel panel,bookPanel,btnPanel;  
  
 *// 标签* private JLabel labReaderID, labName,labType, labSex, labAge,  
 labPhone, labDept, labRegisterDate;  
  
 *// 文本属性* private JTextField txtReaderID,txtName, txtSex,  
 txtAge, txtPhone,txtDept,txtRegisterDate;  
  
 JComboBox cmbType;*//组合框  
  
 // 组件按钮* private JButton btnAdd,btnReset,btnExit;  
  
  
 public ReaderAdd(String s){  
 super(s);  
 setSize(400,200);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
 *//图书面板的信息* GridLayout grid1=new GridLayout(4,4);*//网格布局* grid1.setHgap(5);  
 grid1.setVgap(5);  
 bookPanel=new JPanel(grid1);  
  
 labReaderID =new JLabel("编号:");  
 labReaderID.setHorizontalAlignment(SwingConstants.*CENTER*);*//居中* txtReaderID =new JTextField(15);  
  
 labName = new JLabel("姓名:");  
 labName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 txtName=new JTextField(12);  
  
 labType =new JLabel("类别：");  
 labType.setHorizontalAlignment(SwingConstants.*CENTER*);  
 cmbType =new JComboBox();  
 cmbType.addItem("教师");  
 cmbType.addItem("学生");  
  
  
 labSex =new JLabel("性别:");  
 labSex.setHorizontalAlignment(SwingConstants.*CENTER*);  
 txtSex =new JTextField(12);  
  
  
  
 labAge =new JLabel("年龄:");  
 labAge.setHorizontalAlignment(SwingConstants.*CENTER*);  
 txtAge =new JTextField();  
  
 labPhone =new JLabel("电话:");  
 labPhone.setHorizontalAlignment(SwingConstants.*CENTER*);  
 txtPhone =new JTextField(12);  
  
 labDept =new JLabel("所在部门:");  
 labDept.setHorizontalAlignment(SwingConstants.*CENTER*);  
 txtDept = new JTextField(12);  
  
 labRegisterDate =new JLabel("注册日期:");  
 labRegisterDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 txtRegisterDate =new JTextField(12);  
  
 *// 注册日期  
 //TODO  
  
 //其余类似，作者，出版社，出版社日期，价格，课后大家补充。* bookPanel.add(labReaderID);  
 bookPanel.add(txtReaderID);  
 bookPanel.add(labName);  
 bookPanel.add(txtName);  
 bookPanel.add(labType);  
 bookPanel.add(cmbType);  
 bookPanel.add(labSex);  
 bookPanel.add(txtSex);  
 bookPanel.add(labAge);  
 bookPanel.add(txtAge);  
 bookPanel.add(labPhone);  
 bookPanel.add(txtPhone);  
 bookPanel.add(labDept);  
 bookPanel.add(txtDept);  
 bookPanel.add(labRegisterDate);  
 bookPanel.add(txtRegisterDate);  
  
 *// 注册日期  
 //TODO  
  
 //将各组件加入到面板* panel.add(bookPanel,BorderLayout.*CENTER*);  
 btnPanel=new JPanel();  
 btnAdd=new JButton("增加");  
 btnReset=new JButton("重置");  
 btnExit=new JButton("退出");  
 btnPanel.add(btnAdd);  
 btnPanel.add(btnReset);  
 btnPanel.add(btnExit);  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
 public static void main(String[] args) {  
 new ReaderAdd("读者信息添加");  
 }  
  
}

1. 读者信息查询与修改界面，如图15所示

在中间面板实现查询功能，输入对应的信息实现修改

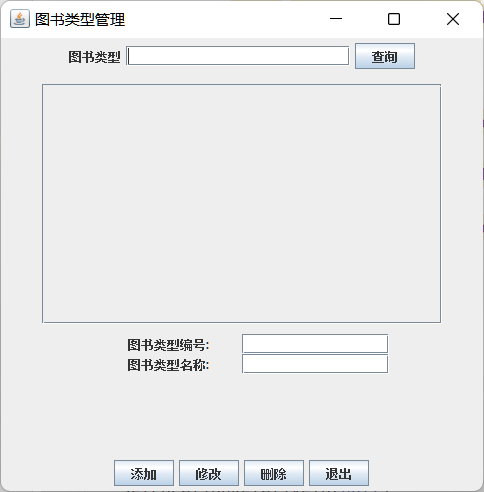


**图15 读者管理界面**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.Dimension;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class ReaderSelectModify extends JFrame {  
 private JPanel panel,selectConditionPane,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
  
 private JComboBox cmbChoice,cmbType;  
  
 private JTextField txtSelect, txtReaderID, txtSex, txtAge,  
 txtPhone, txtDept, txtRegisterDate,txtName;  
  
 private JLabel labReaderID,labType, labSex, labAge,  
 labPhone, labDept, labRegisterDate,labName;  
  
 private JButton btnSelect,btnModify,btnDelete,btnExit;  
  
 private JTable table;  
  
 private JScrollPane scrollPane;  
  
  
 public ReaderSelectModify(){  
 setTitle("读者管理");*//设置标题* setSize(500,500);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
  
 *// 顶部的框* selectConditionPane=new JPanel();  
 cmbChoice=new JComboBox();  
 cmbChoice.addItem("全部");  
 cmbChoice.addItem("读者编号");  
*// cmbChoice.addItem("读者名字");* txtSelect=new JTextField(20);  
 btnSelect=new JButton("查询");  
  
 selectConditionPane.add(cmbChoice);  
 selectConditionPane.add(txtSelect);  
 selectConditionPane.add(btnSelect);  
  
 panel.add(selectConditionPane,BorderLayout.*NORTH*);  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel();  
 table=new JTable();  
 scrollPane=new JScrollPane(table);  
 scrollPane.setPreferredSize(new Dimension(400,240));*//设大小* selectResultPane.add(scrollPane);  
  
  
 *//4行4列，需要创建每个组件，加到bookPane中，另外6个课后完成* bookPane=new JPanel(new GridLayout(4,4));  
 labReaderID =new JLabel("编号:");  
 labName = new JLabel("姓名:");  
 labType=new JLabel("类别:");  
 labSex =new JLabel("性别:");  
 labAge =new JLabel("年龄:");  
 labPhone =new JLabel("电话:");  
 labDept =new JLabel("所在部门:");  
 labRegisterDate =new JLabel("注册日期:");  
 txtReaderID =new JTextField(8);  
  
 cmbType=new JComboBox();  
 cmbType.addItem("教师");  
 cmbType.addItem("学生");  
  
 txtSex =new JTextField(8);  
 txtAge =new JTextField(8);  
 txtPhone =new JTextField(8);  
 txtDept =new JTextField(8);  
 txtRegisterDate =new JTextField(8);  
 txtName = new JTextField(8);  
  
  
 bookPane.add(labReaderID);  
 labReaderID.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtReaderID);  
  
 bookPane.add(labName);  
 labName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtName);  
  
 bookPane.add(labType);  
 labType.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(cmbType);  
  
  
 bookPane.add(labSex);  
 labSex.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtSex);  
  
  
 bookPane.add(labAge);  
 labAge.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtAge);  
  
  
 bookPane.add(labPhone);  
 labPhone.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPhone);  
  
  
 bookPane.add(labDept);  
 labDept.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtDept);  
  
  
 bookPane.add(labRegisterDate);  
 labRegisterDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtRegisterDate);  
  
  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
 btnPanel=new JPanel();  
  
 btnModify=new JButton("修改");  
 btnExit=new JButton("退出");  
  
 btnPanel.add(btnModify);  
 btnPanel.add(btnExit);  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new ReaderSelectModify();  
 }  
  
}

1. 图书类别信息管理界面，如图16所示

根据图书类型id可以进行查询，输入对应的信息可实现类型信息的修改



**图16 图书类型管理**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.Dimension;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class BookTypeManage extends JFrame {  
 private JPanel panel,selectConditionPane,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
  
 private JComboBox cmbChoice,cmbType;  
  
 private JTextField txtSelect, txtBookTypeID, txtBookTypeName;  
  
 private JLabel labBookTypeID,labBookTypeName;  
  
 private JButton btnSelect,btnModify,btnDelete,btnExit,binInsert;  
  
 private JTable table;  
  
 private JScrollPane scrollPane;  
  
  
 public BookTypeManage(){  
 setTitle("图书类型管理");*//设置标题* setSize(500,500);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
 selectConditionPane=new JPanel();  
  
 txtSelect=new JTextField(20);  
 JLabel labSelect = new JLabel("图书类型");  
 btnSelect=new JButton("查询");  
  
 selectConditionPane.add(labSelect);  
 selectConditionPane.add(txtSelect);  
 selectConditionPane.add(btnSelect);  
 panel.add(selectConditionPane,BorderLayout.*NORTH*);  
  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel();  
 table=new JTable();  
 scrollPane=new JScrollPane(table);  
 scrollPane.setPreferredSize(new Dimension(400,240));*//设大小* selectResultPane.add(scrollPane);  
  
 bookPane=new JPanel(new GridLayout(2,4));  
  
 labBookTypeID =new JLabel("图书类型编号:");  
 labBookTypeName = new JLabel("图书类型名称:");  
  
  
 txtBookTypeID =new JTextField(13);  
 txtBookTypeName =new JTextField(13);  
  
  
 bookPane.add(labBookTypeID);  
 labBookTypeID.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtBookTypeID);  
  
 bookPane.add(labBookTypeName);  
 labBookTypeName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtBookTypeName);  
  
  
  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
  
 btnPanel=new JPanel();  
  
 binInsert = new JButton("添加");  
 btnModify=new JButton("修改");  
 btnDelete=new JButton("删除");  
 btnExit=new JButton("退出");  
  
 btnPanel.add(binInsert);  
 btnPanel.add(btnModify);  
 btnPanel.add(btnDelete);  
 btnPanel.add(btnExit);  
  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new BookTypeManage();  
 }  
  
}

1. 读者类型管理，界面如图17所示

根据读者类型id可以进行查询，输入对应的信息可实现类型信息的修改



**图17 读者类型信息修改删除界面**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.Dimension;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class ReaderTypeManage extends JFrame {  
 private JPanel panel,selectConditionPane,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
  
 private JComboBox cmbChoice,cmbType;  
  
 private JTextField txtSelect, txtTypeID, txtLimit,txtTypeName,txtMaxBorrowNum;  
  
 private JLabel labTypeID, labMaxBorrowNum, labLimit,labTypeName,labSelect;  
  
 private JButton btnSelect,btnModify,btnDelete,btnExit,binInsert;  
  
 private JTable table;  
  
 private JScrollPane scrollPane;  
  
  
 public ReaderTypeManage(){  
 setTitle("读者类型管理");*//设置标题* setSize(500,500);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
 selectConditionPane=new JPanel();  
  
 txtSelect=new JTextField(20);  
 JLabel labSelect = new JLabel("读者类型");  
 btnSelect=new JButton("查询");  
  
 selectConditionPane.add(labSelect);  
 selectConditionPane.add(txtSelect);  
 selectConditionPane.add(btnSelect);  
 panel.add(selectConditionPane,BorderLayout.*NORTH*);  
  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel();  
 table=new JTable();  
 scrollPane=new JScrollPane(table);  
 scrollPane.setPreferredSize(new Dimension(400,240));*//设大小* selectResultPane.add(scrollPane);  
  
 bookPane=new JPanel(new GridLayout(2,4));  
  
 labTypeID =new JLabel("读者类型编号:");  
 labTypeName = new JLabel("读者类型名称:");  
 labMaxBorrowNum =new JLabel("可借图书数量:");  
 labLimit =new JLabel("可借图书期限:");  
  
 txtTypeID =new JTextField(8);  
 txtTypeName =new JTextField(8);  
 txtMaxBorrowNum =new JTextField(8);  
 txtLimit =new JTextField(8);  
  
 bookPane.add(labTypeID);  
 labTypeID.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtTypeID);  
  
 bookPane.add(labTypeName);  
 labTypeName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtTypeName);  
  
 bookPane.add(labMaxBorrowNum);  
 labMaxBorrowNum.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtMaxBorrowNum);  
  
  
 bookPane.add(labLimit);  
 labLimit.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtLimit);  
  
  
  
  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
  
 btnPanel=new JPanel();  
  
 binInsert = new JButton("添加");  
 btnModify=new JButton("修改");  
 btnDelete=new JButton("删除");  
 btnExit=new JButton("退出");  
  
 btnPanel.add(binInsert);  
 btnPanel.add(btnModify);  
 btnPanel.add(btnDelete);  
 btnPanel.add(btnExit);  
  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new ReaderTypeManage();  
 }  
  
}

1. 图书借阅界面，如图18所示

输入读者编号确认读者身份，输入图书ISBN确认图书身份，输入当前日期点击借阅，出现在借阅的查询面板当中



**图18图书借阅信息界面**

import java.awt.BorderLayout;  
import java.awt.Dimension;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class BookBorrow extends JFrame {  
 private JPanel panel, ReaderConditionPane,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
  
 private JTextField txtReaderName,txtReaderType,txtReaderID, txtISBN, txtAuthor,txtTypeName, txtBookName,txtPublish,txtPublishdate,txtPrice ,txtBorrowDate,txtUser ;  
  
 private JLabel labISBN, labBookName, labAuthor,labTypeName, labPublish,labPublishDate,labPrice,labBorrowDate,labUser;  
  
 private JButton btnClose, binBorrow;  
  
 private JTable table;  
  
 private JScrollPane scrollPane;  
  
  
 public BookBorrow(){  
 setTitle("图书借阅");*//设置标题* setSize(500,500);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
  
 *// 设置顶部面板* ReaderConditionPane =new JPanel();  
  
 JLabel labReaderID = new JLabel("读者编号:");  
 txtReaderID =new JTextField(8);  
  
 ReaderConditionPane.add(labReaderID);  
 ReaderConditionPane.add(txtReaderID);  
  
 JLabel labReaderName = new JLabel("读者姓名:");  
 txtReaderName = new JTextField(8);  
  
 ReaderConditionPane.add(labReaderName);  
 ReaderConditionPane.add(txtReaderName);  
  
 JLabel labReaderType = new JLabel("读者类别:");  
 txtReaderType = new JTextField(8);  
  
 ReaderConditionPane.add(labReaderType);  
 ReaderConditionPane.add(txtReaderType);  
  
 panel.add(ReaderConditionPane,BorderLayout.*NORTH*);  
  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel();  
 table=new JTable();  
 scrollPane=new JScrollPane(table);  
 scrollPane.setPreferredSize(new Dimension(400,240));*//设大小* selectResultPane.add(scrollPane);  
  
  
 *// 底部面板* bookPane=new JPanel(new GridLayout(5,2));  
  
 labISBN =new JLabel("ISBN:");  
 labTypeName = new JLabel("类别:");  
 labBookName =new JLabel("书名:");  
 labAuthor =new JLabel("作者:");  
 labPublish =new JLabel("出版社:");  
 labPublishDate = new JLabel("出版日期:");  
 labPrice = new JLabel("单价:");  
 labBorrowDate = new JLabel("当前日期:");  
 labUser = new JLabel("操作用户:");  
  
 txtISBN =new JTextField(8);  
 txtTypeName =new JTextField(8);  
 txtBookName =new JTextField(8);  
 txtAuthor =new JTextField(8);  
 txtPublish =new JTextField(8);  
 txtPublishdate = new JTextField(8);  
 txtPrice =new JTextField(8) ;  
 txtBorrowDate = new JTextField(8);  
 txtUser = new JTextField(8);  
  
 bookPane.add(labISBN);  
 labISBN.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtISBN);  
  
 bookPane.add(labTypeName);  
 labTypeName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtTypeName);  
  
 bookPane.add(labBookName);  
 labBookName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtBookName);  
  
  
 bookPane.add(labAuthor);  
 labAuthor.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtAuthor);  
  
 bookPane.add(labPublish);  
 labPublish.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPublish);  
  
 bookPane.add(labPublishDate);  
 labPublishDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPublishdate);  
  
 bookPane.add(labPrice);  
 labPrice.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPrice);  
  
 bookPane.add(labBorrowDate);  
 labBorrowDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtBorrowDate);  
  
 bookPane.add(labUser);  
 labUser.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtUser);  
  
  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
  
  
 btnPanel=new JPanel();  
  
 binBorrow = new JButton("借阅");  
 btnClose =new JButton("关闭");  
  
  
 btnPanel.add(binBorrow);  
 btnPanel.add(btnClose);  
  
  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new BookBorrow();  
 }  
  
}

1. 图书归还界面，如图19所示

输入图书的ISBN点击归还图书从借阅表中删除

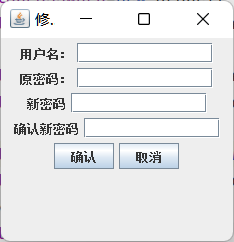


**图19图书归还信息界面**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.Dimension;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class BookReturn extends JFrame {  
 private JPanel panel, ReaderConditionPane,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
  
 private JTextField txtReaderName,txtReaderType,txtReaderID,  
 txtISBN, txtAuthor,txtTypeName, txtBookName,txtPublish,  
 txtPublishdate,txtPrice ,txtBorrowDate,txtUser,txtReturnDate,txtFine ;  
  
 private JLabel labISBN, labBookName, labAuthor,labTypeName, labPublish,  
 labPublishDate,labPrice,labBorrowDate,labUser,labReturnDate,labFine;  
  
 private JButton btnClose, btnReturn;  
  
 private JTable table;  
  
 private JScrollPane scrollPane;  
  
  
 public BookReturn(){  
 setTitle("图书归还");*//设置标题* setSize(500,500);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
  
 *// 设置顶部面板* ReaderConditionPane =new JPanel();  
  
 JLabel labReaderID = new JLabel("读者编号:");  
 txtReaderID =new JTextField(8);  
  
 ReaderConditionPane.add(labReaderID);  
 ReaderConditionPane.add(txtReaderID);  
  
 JLabel labReaderName = new JLabel("读者姓名:");  
 txtReaderName = new JTextField(8);  
  
 ReaderConditionPane.add(labReaderName);  
 ReaderConditionPane.add(txtReaderName);  
  
 JLabel labReaderType = new JLabel("读者类别:");  
 txtReaderType = new JTextField(8);  
  
 ReaderConditionPane.add(labReaderType);  
 ReaderConditionPane.add(txtReaderType);  
  
 panel.add(ReaderConditionPane,BorderLayout.*NORTH*);  
  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel();  
 table=new JTable();  
 scrollPane=new JScrollPane(table);  
 scrollPane.setPreferredSize(new Dimension(400,240));*//设大小* selectResultPane.add(scrollPane);  
  
  
 *// 底部面板* bookPane=new JPanel(new GridLayout(6,2));  
  
 labISBN =new JLabel("ISBN:");  
 labTypeName = new JLabel("类别:");  
 labBookName =new JLabel("书名:");  
 labAuthor =new JLabel("作者:");  
 labPublish =new JLabel("出版社:");  
 labPublishDate = new JLabel("出版日期:");  
 labPrice = new JLabel("单价:");  
 labBorrowDate = new JLabel("当前日期:");  
 labUser = new JLabel("操作用户:");  
 labReturnDate = new JLabel("归还日期:");  
 labFine = new JLabel("罚金:");  
  
 txtISBN =new JTextField(8);  
 txtTypeName =new JTextField(8);  
 txtBookName =new JTextField(8);  
 txtAuthor =new JTextField(8);  
 txtPublish =new JTextField(8);  
 txtPublishdate = new JTextField(8);  
 txtPrice =new JTextField(8) ;  
 txtBorrowDate = new JTextField(8);  
 txtUser = new JTextField(8);  
 txtReturnDate = new JTextField(8);  
 txtFine = new JTextField(8);  
  
 bookPane.add(labISBN);  
 labISBN.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtISBN);  
  
 bookPane.add(labTypeName);  
 labTypeName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtTypeName);  
  
 bookPane.add(labBookName);  
 labBookName.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtBookName);  
  
  
 bookPane.add(labAuthor);  
 labAuthor.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtAuthor);  
  
 bookPane.add(labPublish);  
 labPublish.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPublish);  
  
 bookPane.add(labPublishDate);  
 labPublishDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPublishdate);  
  
 bookPane.add(labPrice);  
 labPrice.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtPrice);  
  
 bookPane.add(labBorrowDate);  
 labBorrowDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtBorrowDate);  
  
 *//TODO* bookPane.add(labReturnDate);  
 labReturnDate.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtReturnDate);  
  
 *//TODO* bookPane.add(labFine);  
 labFine.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtFine);  
  
 bookPane.add(labUser);  
 labUser.setHorizontalAlignment(SwingConstants.*CENTER*);  
 bookPane.add(txtUser);  
  
  
  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
  
  
 btnPanel=new JPanel();  
  
 btnReturn = new JButton("归还");  
 btnClose =new JButton("关闭");  
  
  
 btnPanel.add(btnReturn);  
 btnPanel.add(btnClose);  
  
  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new BookReturn();  
 }  
  
}

1. 用户密码修改界面，如图20所示

输入用户名和原密码得对应，新密码与确认新密码得一致



**图20用户修改密码界面**

package com.bbm.staticview;  
import javax.swing.\*;  
public class UpdatePassword extends JFrame {  
  
 private JPanel myPanel;

private JLabel labName,labPassword,labNewPassword,labConfirmPassword;

private JTextField txtName;

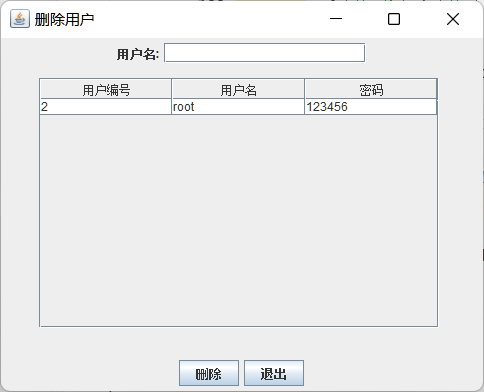
private JPasswordField txtPassword,txtNewPassword,txtConfirmPassword;

private JButton btnConfirm, btnCancel;

public UpdatePassword(String name){  
 super(name);*//框架类设标题* setSize(250,250);  
 setLocationRelativeTo(null);  
 myPanel=new JPanel();  
 setContentPane(myPanel);  
  
 *// 定义标签* labName=new JLabel("用户名：");  
 labName.setHorizontalAlignment(SwingConstants.*CENTER*);  
  
 labPassword=new JLabel("原密码：");  
 labPassword.setHorizontalAlignment(SwingConstants.*CENTER*);  
  
 labNewPassword = new JLabel("新密码");  
 labNewPassword.setHorizontalAlignment(SwingConstants.*CENTER*);  
  
 labConfirmPassword = new JLabel("确认新密码");  
 labConfirmPassword.setHorizontalAlignment(SwingConstants.*CENTER*);  
 *// 原用户名* txtName=new JTextField(12);  
  
 *// 原密码* txtPassword=new JPasswordField(12);  
 txtPassword.setEchoChar('\*');  
  
 *// 新密码* txtNewPassword = new JPasswordField(12);  
 txtPassword.setEchoChar('\*');  
  
 *//确认新密码* txtConfirmPassword = new JPasswordField(12);  
 txtConfirmPassword.setEchoChar('\*');  
  
 btnConfirm=new JButton("确认");  
 btnCancel =new JButton("取消");  
  
 *// 面板添加组件* myPanel.add(labName);  
 myPanel.add(txtName);  
  
 myPanel.add(labPassword);  
 myPanel.add(txtPassword);  
  
 myPanel.add(labNewPassword);  
 myPanel.add(txtNewPassword);  
  
 myPanel.add(labConfirmPassword);  
 myPanel.add(txtConfirmPassword);  
 myPanel.add(btnConfirm);  
 myPanel.add(btnCancel);  
  
 setVisible(true);  
 }  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new UpdatePassword("修改密码");  
 }  
}

1. 用户信息删除界面，如图21所示

输入用户名点击删除，实现删除用户



**图21删除用户界面**

package com.bbm.staticview;  
  
import java.awt.BorderLayout;  
import java.awt.Dimension;  
import java.awt.GridLayout;  
  
import javax.swing.\*;  
  
public class UserDelete extends JFrame {  
 private JPanel panel,btnPanel,  
 centerPanel,selectResultPane,bookPane;  
  
 private JButton btnDelete,btnExit;  
  
 private JTable table;  
  
 private JScrollPane scrollPane;  
  
  
 public UserDelete(){  
 setTitle("删除用户");*//设置标题* setSize(500,400);  
 setLocationRelativeTo(null);  
 panel=new JPanel(new BorderLayout());  
 setContentPane(panel);  
  
  
 *//中间面板* centerPanel=new JPanel();  
 selectResultPane=new JPanel(); *// 查询面板* table=new JTable(); *// 表单* scrollPane=new JScrollPane(table); *// 把表单加入查询面板* scrollPane.setPreferredSize(new Dimension(400,300));*//设查询面板的大小* selectResultPane.add(scrollPane);  
  
 *// 给下面的按钮布局* bookPane=new JPanel(new GridLayout(1,2));  
  
 centerPanel.add(selectResultPane);  
 centerPanel.add(bookPane);  
 panel.add(centerPanel,BorderLayout.*CENTER*);  
  
 btnPanel=new JPanel();  
  
  
 btnDelete=new JButton("删除");  
 btnExit=new JButton("退出");  
  
  
 btnPanel.add(btnDelete);  
 btnPanel.add(btnExit);  
  
 panel.add(btnPanel,BorderLayout.*SOUTH*);  
  
 setVisible(true);  
 }  
  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new UserDelete();  
 }  
  
}

1. 用户信息增加界面，如图22所示

这是一个注册界面，用户名与密码不能为空，用户名不能和存在过的一致。输入用户名和密码后点击添加实现注册功能。



**图22添加用户界面**

package com.bbm.staticview;  
  
import javax.swing.\*;  
  
public class UserAdd extends JFrame {  
 private JPanel myPanel;  
 private JLabel labName,labPassword;  
 private JTextField txtName;  
 private JPasswordField txtPassword;  
 private JButton btnAdd, btnCancel;  
  
 public UserAdd(String name){  
 super(name);*//框架类设标题* setSize(250,150);  
 setLocationRelativeTo(null);  
  
 myPanel=new JPanel();  
 setContentPane(myPanel);  
  
 labName=new JLabel("用户名：");  
 labPassword=new JLabel("密 码：");  
  
 txtName=new JTextField(12);  
 txtPassword=new JPasswordField(12);  
 txtPassword.setEchoChar('\*');  
  
 btnAdd =new JButton("添加");  
 btnCancel =new JButton("取消");  
  
 myPanel.add(labName);  
 myPanel.add(txtName);  
 myPanel.add(labPassword);  
 myPanel.add(txtPassword);  
 myPanel.add(btnAdd);  
 myPanel.add(btnCancel);  
  
 setVisible(true);  
 }  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new UserAdd("添加用户");  
 }  
  
}

1. 图书系统主界面，如图23所示

这是登陆成功之后进入的主界面，有几大功能模块。



**图23图书系统主界面**

package com.bbm.staticview;  
import javax.swing.\*;  
public class Library extends JFrame {  
 private JMenuBar bar;*//菜单条* private JMenu menuBook,menuReader,menuType,menuUser,menuBorrowBook;*//菜单* private JMenuItem itemBookAdd,itemBookSelect,itemReaderAdd,itemReaderSelect,  
 itemBookTypeManage,itemReaderTypeManage,itemUserAdd,  
 itemUserDelete,itemUserUpdate,itemBookBorrow,itemBookReturn;*//菜单项* public Library(String s){  
 super(s);  
 setSize(1000,800);  
 setLocationRelativeTo(null);  
 bar=new JMenuBar();  
 setJMenuBar(bar);  
  
 *// 图书信息管理模块* menuBook=new JMenu("图书信息管理");*//菜单* itemBookAdd=new JMenuItem("图书增加");*//菜单项* itemBookSelect=new JMenuItem("图书查询与修改");  
  
 menuBook.add(itemBookAdd);  
 menuBook.add(itemBookSelect);  
  
 *// 读者信息管理模块* menuReader = new JMenu("读者信息管理");  
  
 itemReaderAdd = new JMenuItem("读者添加");  
 itemReaderSelect = new JMenuItem("读者查询与修改");  
  
 menuReader.add(itemReaderAdd);  
 menuReader.add(itemReaderSelect);  
  
 *// 类型管理* menuType = new JMenu("类型管理");  
  
 itemBookTypeManage = new JMenuItem("图书类型管理");  
 itemReaderTypeManage = new JMenuItem("读者类型管理");  
  
 menuType.add(itemBookTypeManage);  
 menuType.add(itemReaderTypeManage);  
  
 *// 用户管理* menuUser = new JMenu("用户管理");  
  
 itemUserAdd =new JMenuItem("注册用户");  
 itemUserDelete = new JMenuItem("删除用户");  
 itemUserUpdate = new JMenuItem("修改密码");  
  
 menuUser.add(itemUserAdd);  
 menuUser.add(itemUserDelete);  
 menuUser.add(itemUserUpdate);  
  
 *// 借阅管理* menuBorrowBook = new JMenu("借阅管理");  
  
 itemBookBorrow = new JMenuItem("图书借阅");  
 itemBookReturn = new JMenuItem("图书归还");  
  
 menuBorrowBook.add(itemBookBorrow);  
 menuBorrowBook.add(itemBookReturn);  
  
 bar.add(menuBook);  
 bar.add(menuReader);  
 bar.add(menuType);  
 bar.add(menuUser);  
 bar.add(menuBorrowBook);  
 setVisible(true);   
 }  
 public static void main(String[] args) {  
 *// TODO Auto-generated method stub* new Library("图书借阅系统");  
 }

# 子项目：图书借阅系统数据访问方法

1.Dao类连接数据库，获取数据库实例(线程安全)，获取数据库连接，关闭资源

package com.bbm.db;  
import com.mysql.cj.jdbc.MysqlDataSource;  
  
import javax.sql.DataSource;  
import java.sql.\*;  
  
public class Dao {private static final String *URL* = "jdbc:mysql://127.0.0.1:3306/bookSystem?characterEncoding=utf8&useSSL=false&allowPublicKeyRetrieval=true" ;  
 private static final String *USERNAME* = "root";  
 private static final String *PASSWORD* = "123456";  
  
 *// 获取数据库连接* private static volatile DataSource *dataSource* = null;  
  
 *// 单例模式获得数据库实例* private static DataSource getDataSource(){  
 if (*dataSource*==null) {  
 synchronized (Dao.class) {  
 if(*dataSource*==null){  
 *dataSource* = new MysqlDataSource();  
 ((MysqlDataSource)*dataSource*).setURL(*URL*);  
 ((MysqlDataSource)*dataSource*).setUser(*USERNAME*);  
 ((MysqlDataSource)*dataSource*).setPassword(*PASSWORD*);  
 }  
 }  
 }  
 return *dataSource*;  
 }  
  
 *// 获得数据库连接* public static Connection getConnection(){  
 try {  
 return *getDataSource*().getConnection();  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
  
 return null;  
 }  
  
 *//回收资源* public static void close(ResultSet resultSet, PreparedStatement statement ,Connection connection){  
  
 if (resultSet!=null) {  
 try {  
 resultSet.close();  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 }  
  
 if (statement!=null) {  
 try {  
 statement.close();  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 }  
  
 if (connection!=null) {  
 try {  
 connection.close();  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 }  
 }  
   
}

2.BookDao 类实现操作数据库中的book表，并对表中的数据进行增删改查

package com.bbm.db;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.util.\*;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
  
import com.bbm.model.Book;  
  
public class BookDao {  
 *//根据ISBN号删除图书信息* public static int deleteBook(String ISBN) {  
  
 *// 先定义数据库连接，预处理* Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
  
  
 try{  
 *//1.获取到数据库连接* connection = Dao.*getConnection*();  
 *//2.拼装sql语句* String sql="delete from book where ISBN='"+ISBN+"'";  
 statement = connection.prepareStatement(sql);  
 ret = statement.executeUpdate();  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }finally {  
 Dao.*close*(null,statement,connection);  
 }  
 return ret;  
 }  
  
  
 *//增加图书信息* public static int insertBook(Book b,String typeName) {  
 *//稍微复杂：图书表中包括的是类型编号，界面设计时是类型的名称，  
 //所以，在增加之前，需要先将类型名称转换成类型编号  
 //根据类型名查找到相应的类型编号，涉及到图书类型表* Connection connection = Dao.*getConnection*();  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 int ret = 0;  
 *// 根据图书的类型名找到图书编号* try {  
 *// 拼接sql语句* String sql1="select typeid from booktype where typename='" +typeName+"'";  
 statement = connection.prepareStatement(sql1);  
 resultSet = statement.executeQuery();  
  
 int typeID = -1;  
  
 if(resultSet.next()){  
 typeID = resultSet.getInt("typeid");  
 }  
 *// 根据类型名拿到图书类型的id了* String sql2="insert into Book(ISBN,bookname,author,publish,publishdate,price,typeid,typename)"  
 +"values('"+b.getISBN()+"','"+b.getBookName()+"','"+b.getAuthor()  
 +"','"+b.getPublish()+"','"+b.getPublishDate()  
 +"',"+b.getPrice()+","+typeID+",'"+typeName+"'"+")";  
  
 System.*out*.println(sql2);  
 statement = connection.prepareStatement(sql2);  
  
 ret = statement.executeUpdate(sql2);  
  
  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
  
 return ret;  
 }  
  
  
  
 *//查询功能* public static List<Book> selectBook(String s1, String s2){  
 *//根据下拉框的选择 进行查询  
 // 第一种 下拉框是全部，查询全部  
 // 第二种 下拉框是书名，文本框输入具体的值，根据这两个条件进行查询* Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 List<Book> listBook =new ArrayList<Book>();  
  
 try {  
 *//1.连接数据库* connection = Dao.*getConnection*();  
  
 *// 2.拼接sql语句* String sql = "SELECT ISBN,bookname,author,publish,publishDate,price,book.typeid,book.typename from book JOIN booktype "  
 + "on book.typeid=booktype.typeid";  
 if (s1.equals("书名"))  
 sql = sql + " where bookname='" + s2 + "'";  
  
 *//3.执行sql查询语句* statement = connection.prepareStatement(sql);  
  
 resultSet = statement.executeQuery();  
  
 *//可以包括多类的查询条件，根据前面的界面来完善。* while(resultSet.next()) {  
 Book b1 = new Book();  
 b1.setISBN(resultSet.getString("ISBN"));  
 b1.setBookName(resultSet.getString("bookname"));  
 b1.setPublish(resultSet.getString("publish"));  
 b1.setAuthor(resultSet.getString("author"));  
 b1.setPublishDate(resultSet.getString("publishdate"));  
 b1.setPrice(resultSet.getDouble("price"));  
 b1.setTypeID(resultSet.getInt("typeid"));  
 b1.setTypeName(resultSet.getString("typename"));  
 listBook.add(b1);  
 }  
 return listBook;  
  
 }catch (SQLException e){  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
  
 return null;  
  
 }  
   
 *//修改* public static int updateBook(Book b){  
 Connection connection = Dao.*getConnection*();  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 int ret = 0;  
  
  
 try {  
 *// 拼接sql语句* String sql1="select typeid from booktype where typename='" +b.getTypeName()+"'";  
 statement = connection.prepareStatement(sql1);  
 resultSet = statement.executeQuery();  
  
 int typeID = -1;  
  
 if(resultSet.next()){  
 typeID = resultSet.getInt("typeid");  
 }  
  
 *//update book set bookname='计算机网络',author='肖朝晖',  
 //publish='北京理工大学出版社',price=34,  
 //typeid=1,publishdate='20180101' where isbn='003'  
  
 // 拼接sql 修改语句* String sqlUpdat="update book set bookname='"+b.getBookName()  
 +"',author='"+b.getAuthor()+"',publish='"+b.getPublish()  
 +"',price="+b.getPrice()+",typeid="+typeID+",publishdate='"  
 +b.getPublishDate()+"' where isbn='"+b.getISBN()+"'";  
  
 statement = connection.prepareStatement(sqlUpdat);  
 ret = statement.executeUpdate();  
  
 } catch(SQLException e){  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
 return ret;  
 }  
  
 public static void main(String[] args) {  
 BookDao.*selectBook*("","");  
 }  
  
}

3.ReaderDao 类实现操作数据库中的reader表，并对表中的数据进行增删改查

package com.bbm.db;  
  
import com.bbm.model.Book;  
import com.bbm.model.Reader;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class ReaderDao {  
  
 *//增删改查  
  
 // 增* public static int insertReader(Reader reader,String typeName){  
 Connection connection = Dao.*getConnection*();  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 int ret = 0;  
 *// 读者表中 typeid 与 读者类型表中 typeid 相联系  
 // 根据读者的类型名 typename 找到读者类型编号 typeid* try {  
 *// 拼接sql语句 从读者类型表 readertype 中根据 typename查找 typeid* String sql1="select typeid from readertype where typename=?";  
 statement = connection.prepareStatement(sql1);  
 *// 占位符替换* statement.setString(1,typeName);  
  
 *// 执行sql查询语句* resultSet = statement.executeQuery();  
  
 int typeID = -1;  
  
 if(resultSet.next()){  
 typeID = resultSet.getInt("typeid");  
 }  
 *// 根据类型名拿到读者类型的id了* System.*out*.println(typeID);  
  
 String sql2="insert into reader values (?,?,?,?,?,?,?,?,?)";  
  
 *// 占位符替换* statement = connection.prepareStatement(sql2);  
 statement.setString(1,reader.getReaderid());  
 statement.setString(2,reader.getName());  
 statement.setString(3,reader.getSex());  
 statement.setInt(4,reader.getAge());  
 statement.setString(5,reader.getPhone());  
 statement.setString(6,reader.getDept());  
 statement.setString(7,reader.getRegdate());  
 statement.setInt(8,typeID);  
 statement.setString(9,typeName);  
  
*// System.out.println(sql2);  
  
 // 执行sql增加语句* ret = statement.executeUpdate();  
  
  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
  
 return ret;  
 }  
  
  
 *// 删 根据读者的编号进行删除操作* public static int deleteReader(String readerid){  
  
 *// 先定义数据库连接，预处理* Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
  
  
 try{  
 *//1.获取到数据库连接* connection = Dao.*getConnection*();  
 *//2.拼装sql语句* String sql="delete from reader where readerid = ?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setString(1,readerid);  
  
 *// 3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }finally {  
 Dao.*close*(null,statement,connection);  
 }  
 return ret;  
 }  
  
  
 *// 改* public static int updateReader(Reader reader){  
 Connection connection = Dao.*getConnection*();  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 int ret = 0;  
  
  
 try {  
 *// 拼接sql语句 从读者类型表 readertype 中根据 typename查找 readerid* String sql1="select typeid from readertype where typename= ?" ;  
 statement = connection.prepareStatement(sql1);  
 *// 占位符替换* statement.setString(1,reader.getTypename());  
  
 System.*out*.println(reader.getTypename());  
  
 *// 执行sql查询语句* resultSet = statement.executeQuery();  
  
 int typeID = -1;  
  
 if(resultSet.next()){  
 typeID = resultSet.getInt("typeid");  
 }  
  
 System.*out*.println(typeID);  
 *// 根据类型名拿到读者类型的id了  
  
 //update book set bookname='计算机网络',author='肖朝晖',  
 //publish='北京理工大学出版社',price=34,  
 //typeid=1,publishdate='20180101' where isbn='003'  
  
 // 拼接sql 修改语句* String sqlUpdat="update reader set name=?,sex=?,age=?,phone=?,dept=?,regdate=?,typeid=? ,typename=? where readerid=?";  
 statement = connection.prepareStatement(sqlUpdat);  
  
 *// 占位符替换* statement.setString(1,reader.getName());  
 statement.setString(2,reader.getSex());  
 statement.setInt(3,reader.getAge());  
 statement.setString(4,reader.getPhone());  
 statement.setString(5,reader.getDept());  
 statement.setString(6,reader.getRegdate());  
 statement.setInt(7,typeID);  
 statement.setString(8,reader.getTypename());  
 statement.setString(9,reader.getReaderid());  
  
  
 *// 执行sql语句* ret = statement.executeUpdate();  
  
 } catch(SQLException e){  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
 return ret;  
 }  
  
 *// 查* public static List<Reader> selectReader(String s1, String s2){  
 *//一个是下拉框中选中的字符串，一个是* Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 List<Reader> list =new ArrayList<Reader>();  
  
  
 try {  
 *//1.连接数据库* connection = Dao.*getConnection*();  
  
 *// 2.拼接sql语句* String sql = "select readerid ,name,sex,age,phone,dept,regdate,reader.typeid,readertype.typename from reader JOIN readertype on reader.typeid=readertype.typeid";  
*// System.out.println("执行到了if的前一行");* if (s1.equals("读者编号")){  
*// System.out.println("执行了if判断");* sql = sql+" where readerid='" + s2 + "'";  
 }  
  
*// System.out.println("执行了if判断之后的条件");* System.*out*.println(sql);  
  
 *//3.执行sql查询语句* statement = connection.prepareStatement(sql);  
  
 resultSet = statement.executeQuery();  
  
 *//可以包括多类的查询条件，根据前面的界面来完善。* while(resultSet.next()) {  
 Reader reader = new Reader();  
 reader.setReaderid(resultSet.getString("readerid"));  
 reader.setName(resultSet.getString("name"));  
 reader.setSex(resultSet.getString("sex"));  
 reader.setAge(resultSet.getInt("age"));  
 reader.setPhone(resultSet.getString("phone"));  
 reader.setDept(resultSet.getString("dept"));  
 reader.setRegdate(resultSet.getString("regdate"));  
 reader.setTypename(resultSet.getString("typename"));  
 list.add(reader);  
 }  
 return list;  
  
 }catch (SQLException e){  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
  
 return null;  
 }  
  
}

4.BookTypeDao 类实现操作数据库中的bookType表，并对表中的数据进行增删改查

package com.bbm.db;  
  
import com.bbm.model.Book;  
import com.bbm.model.BookType;  
import com.bbm.model.ReaderType;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class BookTypeDao {  
  
  
 *//增删改查  
  
 // 增* public static int insertBookType(BookType bookType) {  
 Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
  
 *//2.拼接sql语句* String sql = "insert into booktype values (?,?)";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符防止sql注入，最好不要直接拼接到sql语句中* statement.setInt(1, bookType.getTypeid());  
 statement.setString(2,bookType.getTypename());  
  
 *//3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return ret;  
 }  
  
 *// 删 因为书籍类型编号是主键，所以是唯一的，所以就根据主键来删除记录* public static int deleteBookType(int typeid) {  
 int ret = 0;  
 Connection connection = null;  
 PreparedStatement statement = null;  
  
  
 try {  
 *//1.拿到数据库连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "delete from booktype where typeid =?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setInt(1, typeid);  
 *// 3.执行sql语句* ret = statement.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
  
 return ret;  
 }  
  
 *// 改 因为主键是唯一的，所以根据主键id进行修改读者类型信息* public static int updateBookType(BookType bookType) {  
 Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "update booktype set typename=? where typeid= ?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符防止sql注入，最好不要直接拼接到sql语句中* statement.setString(1, bookType.getTypename());  
 statement.setInt(2, bookType.getTypeid());  
  
 *//3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return ret;  
 }  
  
 *// 查 查找所有的读者类型* public static List<BookType> selectBookType() {  
 List<BookType> list = new ArrayList<BookType>();  
 Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "select *\** from booktype";  
 statement = connection.prepareStatement(sql);  
  
 *//3.执行sql语句* resultSet = statement.executeQuery();  
 while (resultSet.next()) {  
 BookType bookType = new BookType();  
 bookType.setTypeid(resultSet.getInt("typeid"));  
 bookType.setTypename(resultSet.getString("typename"));  
 list.add(bookType);  
 }  
 return list;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return null;  
 }  
  
  
 *// 查 按照指定的 读者编号进行查找读者类型* public static List<BookType> selectBookTypeById(int typeid) {  
  
 List<BookType> list = new ArrayList<BookType>();  
 Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 BookType bookType = new BookType();  
  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "select *\** from booktype where typeid =?";  
 statement = connection.prepareStatement(sql);  
 statement.setInt(1,typeid);  
  
 *//3.执行sql语句* resultSet = statement.executeQuery();  
 if(resultSet.next()) {  
 bookType.setTypeid(resultSet.getInt("typeid"));  
 bookType.setTypename(resultSet.getString("typename"));  
 list.add(bookType);  
 }  
 return list;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return null;  
 }  
  
  
}

5.ReaderTypeDao 类实现操作数据库中的readerType表，并对表中的数据进行增删改查

package com.bbm.db;  
import com.bbm.model.Reader;  
import com.bbm.model.ReaderType;  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class ReaderTypeDao {  
  
 *//增删改查  
  
 // 增* public static int insertReaderType(ReaderType readerType) {  
  
 Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "insert into readertype values (?,?,?,?)";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符防止sql注入，最好不要直接拼接到sql语句中* statement.setInt(1,readerType.getTypeid());  
 statement.setString(2, readerType.getTypename());  
 statement.setInt(3, readerType.getMaxborrownum());  
 statement.setInt(4, readerType.getLimit());  
 *//3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return ret;  
 }  
  
 *// 删 因为读者类型编号是主键，所以是唯一的，所以就根据主键来删除记录* public static int deleteReaderType(int typeid) {  
 int ret = 0;  
 Connection connection = null;  
 PreparedStatement statement = null;  
  
  
 try {  
 *//1.拿到数据库连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "delete from readertype where typeid =?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setInt(1, typeid);  
 *// 3.执行sql语句* ret = statement.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
  
 return ret;  
 }  
  
 *// 改 因为主键是唯一的，所以根据主键id进行修改读者类型信息* public static int updateReaderType(ReaderType readerType, int typeid) {  
 Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "update readertype set typename=?,maxborrownum=?,`limit`=? where typeid= ?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符防止sql注入，最好不要直接拼接到sql语句中* statement.setString(1, readerType.getTypename());  
 statement.setInt(2, readerType.getMaxborrownum());  
 statement.setInt(3, readerType.getLimit());  
 statement.setInt(4, typeid);  
 *//3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return ret;  
 }  
  
 *// 查 查找所有的读者类型* public static List<ReaderType> selectReaderType() {  
 List<ReaderType> list = new ArrayList<ReaderType>();  
 Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "select *\** from readertype";  
 statement = connection.prepareStatement(sql);  
  
 *//3.执行sql语句* resultSet = statement.executeQuery();  
 while (resultSet.next()) {  
 ReaderType readerType = new ReaderType();  
 readerType.setTypeid(resultSet.getInt("typeid"));  
 readerType.setTypename(resultSet.getString("typename"));  
 readerType.setMaxborrownum(resultSet.getInt("maxborrownum"));  
 readerType.setLimit(resultSet.getInt("limit"));  
 list.add(readerType);  
 }  
 return list;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return null;  
 }  
  
  
 *// 查 按照指定的 读者编号进行查找读者类型* public static List<ReaderType> selectReaderTypeById(int typeid) {  
  
 Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 List<ReaderType> list = new ArrayList<ReaderType>();  
  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "select *\** from readertype where typeid =?";  
 statement = connection.prepareStatement(sql);  
 statement.setInt(1,typeid);  
  
 *//3.执行sql语句* resultSet = statement.executeQuery();  
 if(resultSet.next()) {  
 ReaderType readerType = new ReaderType();  
 readerType.setTypeid(resultSet.getInt("typeid"));  
 readerType.setTypename(resultSet.getString("typename"));  
 readerType.setMaxborrownum(resultSet.getInt("maxborrownum"));  
 readerType.setLimit(resultSet.getInt("limit"));  
 list.add(readerType);  
 }  
 return list;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return null;  
 }  
  
}

6.BookBorrowDao 类实现操作数据库中的borrowBook表，并对表中的数据进行增删改查

package com.bbm.db;  
import com.bbm.model.Book;  
import com.bbm.model.BorrowBook;  
import com.bbm.model.Reader;  
import netscape.security.UserTarget;  
  
import java.sql.\*;  
import java.util.ArrayList;  
import java.util.List;  
  
public class BorrowBookDao {  
*// BorrowBook 表中 放的都是 借走的书，没借走的书都不在这里  
  
// 这个涉及到 图书借阅 与 归还  
  
// 有点儿搞不懂功能需求  
  
 //增删改查  
  
 // 增 就相当于 借阅* public static int borrowBook(String readerid,String ISBN,String borrowDate){  
 *// 借阅需要知道 借书人的信息、书的信息  
  
// 根据图书借阅表的标签和文本框，我们往bookBorrow表中插入信息* Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
  
 try {  
 *//1. 获取数据库连接* connection = Dao.*getConnection*();  
  
 *//2.拼接sql语句* String sql = "insert into borrowbook(readerid,ISBN,borrowdate) values(?,?,?)";  
  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setString(1,readerid);  
 statement.setString(2,ISBN);  
 statement.setString(3,borrowDate);  
  
 *//3.执行sql语句* ret = statement.executeUpdate();  
  
 return ret;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(null,statement,connection);  
 }  
  
 return 0;  
 }  
  
  
 *// TODO  
 // 删 就相当于 归还 , 根据ISBN进行删除 //* public static int returnBook(String ISBN){  
 *// 将借的书信息 从 表中删除* Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
  
 try {  
 *//1. 获取数据库连接* connection = Dao.*getConnection*();  
  
 *//2.拼接sql语句* String sql = "delete from borrowbook where ISBN = ?";  
  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setString(1,ISBN);  
  
 *//3.执行sql语句* ret = statement.executeUpdate();  
  
 return ret;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(null,statement,connection);  
 }  
  
 return 0;  
 }  
  
 *// 改 改变图书的借阅状态，跟增删差不多  
 // 但是前端页面中并没有这个功能* public static int updateBookState(){  
 return 0;  
 }  
  
 *// 查 只显示被借走的书* public static List<BorrowBook> selectBorrowBook(){  
 *// 这个查询功能只 查三个东西 (ISBN，bookname，借书日期)  
  
 // borrowbook 里面没有bookname属性，后来我自己又添加了，否则没办法接收 联表查询的结果* Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
 List<BorrowBook> list = new ArrayList<BorrowBook>();  
  
 try {  
 *//1.获取数据库连接* connection = Dao.*getConnection*();  
  
 *//2. 拼接sql语句* String sql = "select book.ISBN,book.bookname,borrowbook.borrowdate from book join borrowbook on borrowbook.ISBN = book.ISBN";  
  
 *//3. 执行sql语句* statement = connection.prepareStatement(sql);  
 resultSet = statement.executeQuery();  
  
 while(resultSet.next()){  
 BorrowBook borrowBook = new BorrowBook();  
 borrowBook.setISBN(resultSet.getString("ISBN"));  
 borrowBook.setBookname(resultSet.getString("bookname"));  
 borrowBook.setBorrowdate(resultSet.getString("borrowdate"));  
 list.add(borrowBook);  
 }  
  
 return list;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }finally {  
 Dao.*close*(resultSet,statement,connection);  
 }  
  
 return null;  
 }  
}

7.UserDao 类实现操作数据库中的user表，并对表中的数据进行增删改查

package com.bbm.db;  
import com.bbm.model.ReaderType;  
import com.bbm.model.User;  
  
import java.net.UnknownServiceException;  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class UserDao {  
 *//增删改查  
  
 // 增 前端界面写入user的所有信息，封装成一个user对象插入* public static int insertUser(User user) {  
  
 Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "insert into user values (null,?,?)";  
 statement = connection.prepareStatement(sql);  
  
  
 *// 占位符防止sql注入，最好不要直接拼接到sql语句中* statement.setString(1,user.getName());  
 statement.setString(2,user.getPassword());  
 *//3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return ret;  
 }  
  
  
 *// 删 因为用户id是自增主键所以 删除根据id删除* public static int deleteUser(String name){  
 int ret = 0;  
 Connection connection = null;  
 PreparedStatement statement = null;  
  
  
 try {  
 *//1.拿到数据库连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "delete from user where name =?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setString(1,name);  
 *// 3.执行sql语句* ret = statement.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
  
 return ret;  
 }  
  
 *// 改 前端页面只显示指定用户名的情况下，修改密码* public static int updateUser(String name,String password){  
 Connection connection = null;  
 PreparedStatement statement = null;  
 int ret = 0;  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "update user set password=? where name= ?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符防止sql注入，最好不要直接拼接到sql语句中* statement.setString(1, password);  
 statement.setString(2, name);  
  
 *//3.执行sql语句* ret = statement.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return ret;  
 }  
  
 *// 查 前端页面会在删除具体用户的时候 展示所有用户信息，所以直接查询所有信息即可* public static List<User> selectUser(){  
 List<User> list = new ArrayList<User>();  
 Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "select *\** from user";  
 statement = connection.prepareStatement(sql);  
  
 *//3.执行sql语句* resultSet = statement.executeQuery();  
 while (resultSet.next()) {  
 User user = new User();  
 user.setId(resultSet.getInt("id"));  
 user.setName(resultSet.getString("name"));  
 user.setPassword(resultSet.getString("password"));  
 list.add(user);  
 }  
 return list;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return null;  
 }  
  
 public static User selectUserByName(String name){  
 User user = new User();  
 Connection connection = null;  
 PreparedStatement statement = null;  
 ResultSet resultSet = null;  
  
 try {  
 *//1.先拿到数据库的连接* connection = Dao.*getConnection*();  
 *//2.拼接sql语句* String sql = "select *\** from user where name=?";  
 statement = connection.prepareStatement(sql);  
  
 *// 占位符替换* statement.setString(1,name);  
  
 *//3.执行sql语句* resultSet = statement.executeQuery();  
 if(resultSet.next()) {  
 user.setId(resultSet.getInt("id"));  
 user.setName(resultSet.getString("name"));  
 user.setPassword(resultSet.getString("password"));  
 }  
 return user ;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 } finally {  
 Dao.*close*(null, statement, connection);  
 }  
 return null;  
 }  
  
}

# 子项目：图书借阅系统功能设计与实现

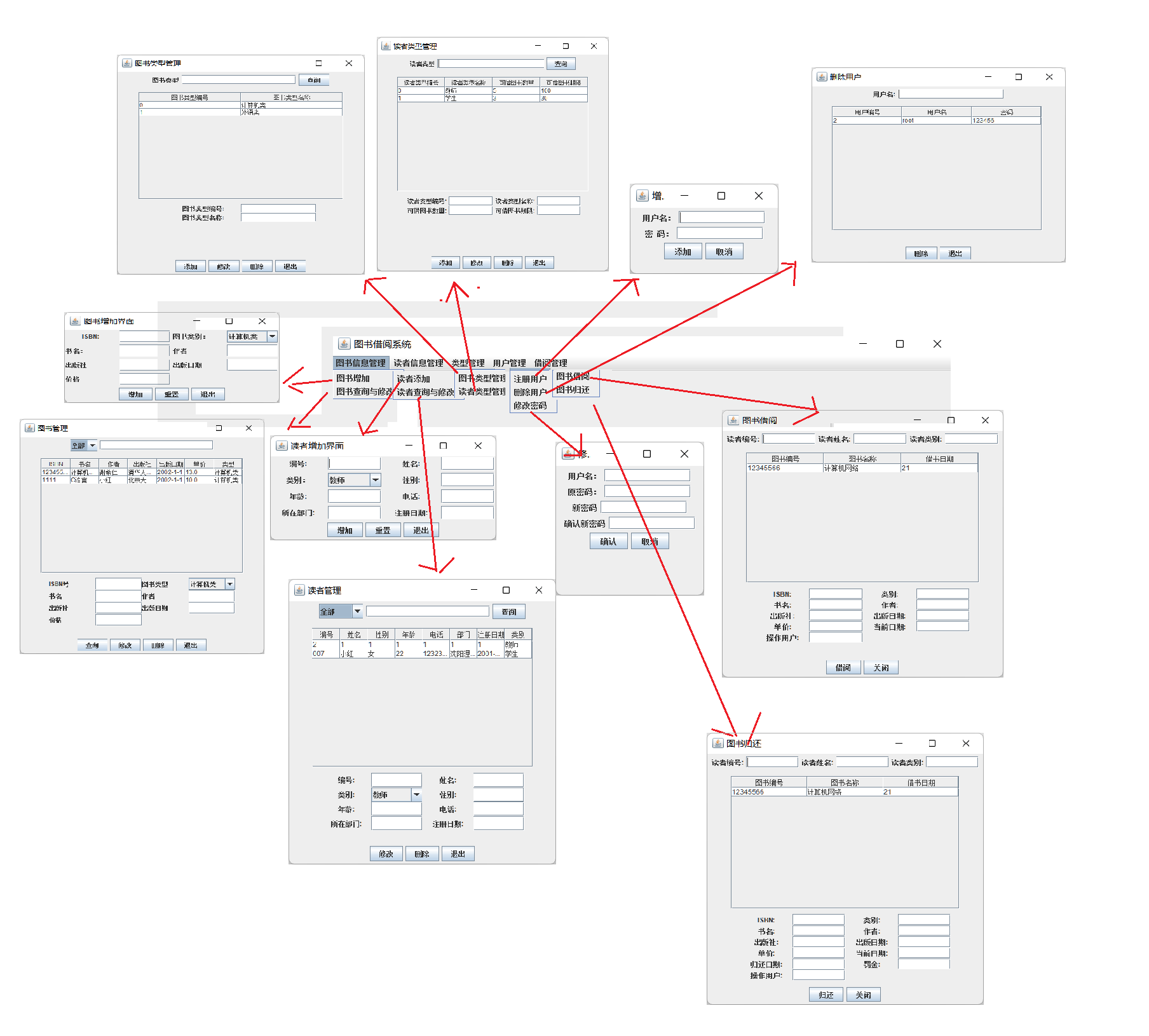
1. 登陆界面功能实现，如图24所示

点击登陆进入主界面，点击重置文本框清空



**如图24 登录功能事件设计**

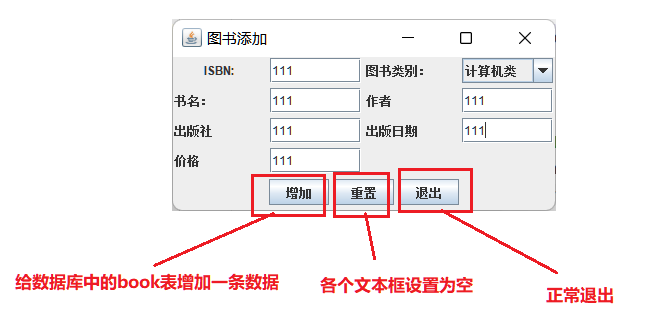
1. 主界面功能实现，如图25所示

点击主页面具体模块的下拉选项，会出现对应的页面

**如图25 图书借阅系统主界面事件设计**

1. 图书增加功能实现，如图26所示

点击增加，给book表中增加一条数据，点击重置设置文本框为空，点击退出正常退出。



**如图26 图书信息增加事件设计**

1. 图书查询与修改功能实现，如图27所示

点击查询，如果上面的下拉框是全部，那么查询所有图书信息，并显示到面板中，如果下拉框是书名，那么根据书名进行查询并把结果显示到中间面板中。

点击修改，输入对应的内容，修改对应图书的信息

点击删除，输入ISBN，删除表中对应的图书信息

点击退出，正常退出界面



**如图 27所示 图书信息修改查询界面事件设计**

1. 读者增加功能实现，如图28所示

点击增加，给reader表中增加一条数据，点击重置设置文本框为空，点击退出正常退出。



**图28读者增加事件设计**

1. 读者查询与修改功能实现，如图29所示

点击查询，如果上面的下拉框是全部，那么查询所有图书信息，并显示到面板中，如果下拉框是书名，那么根据书名进行查询并把结果显示到中间面板中.点击修改，输入对应的内容，修改对应图书的信息.点击删除，输入ISBN，删除表中对应的图书信息。点击退出，正常退出界面



**图29读者信息删除与修改事件设计**

1. 读者类型管理功能实现，如图30所示



**图30读者类型信息事件设计**

1. 图书类型管理功能实现，如图31所示

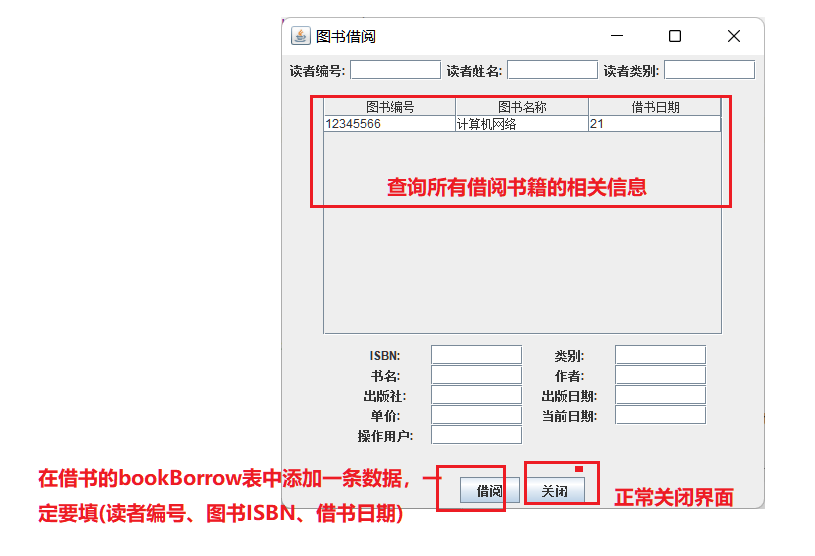
在表单显示所有的类型信息，点击添加，给booktype表中添加数据，点击修改，修改对应类型编号的数据的相关信息。点击删除，删除指定类型编号的图书类型信息，点击退出，正常退出界面。



**图31图书类型管理信息事件设计**

1. 图书借阅功能实现，如图32所示

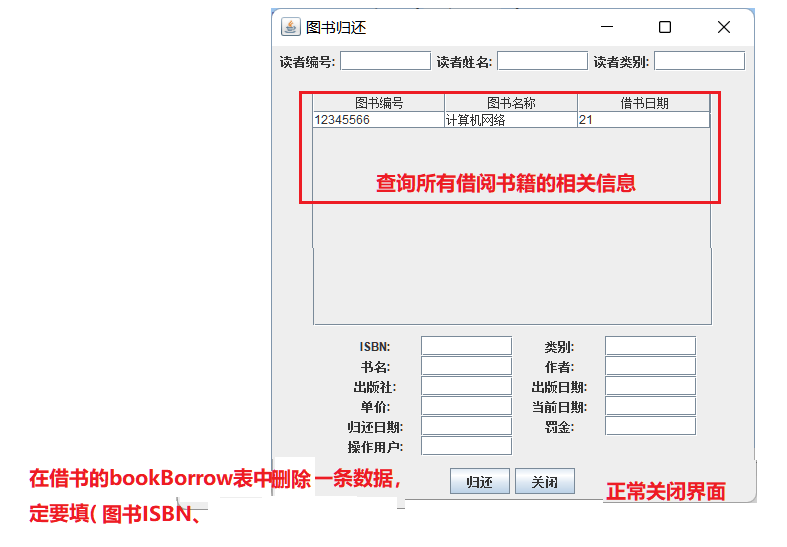
在表单中显示所有已经被借走的书籍信息，点击借阅，将借阅的信息进行增加到bookBorrow的表中。点击关闭页面正常退出。



**图32 图书借阅信息事件设计**

1. 图书归还功能实现，如图33所示

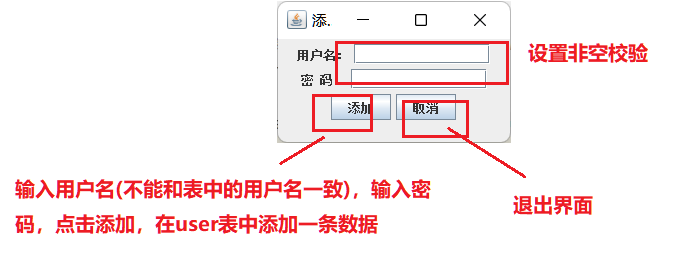
在表单中显示所有已经被借走的书籍信息，点击归还，将借阅的信息进行删除。点击关闭页面正常退出。



**图33图书归还信息事件设计**

1. 用户增加功能实现，如图34所示

在文本框中输入用户名和密码，点击添加，将对应信息放到user类中，在插入到user表中。 用户名都是唯一的，不能输入已经存在过的用户名。



**图34 用户增加事件设计**

1. 用户删除功能实现，如图35所示

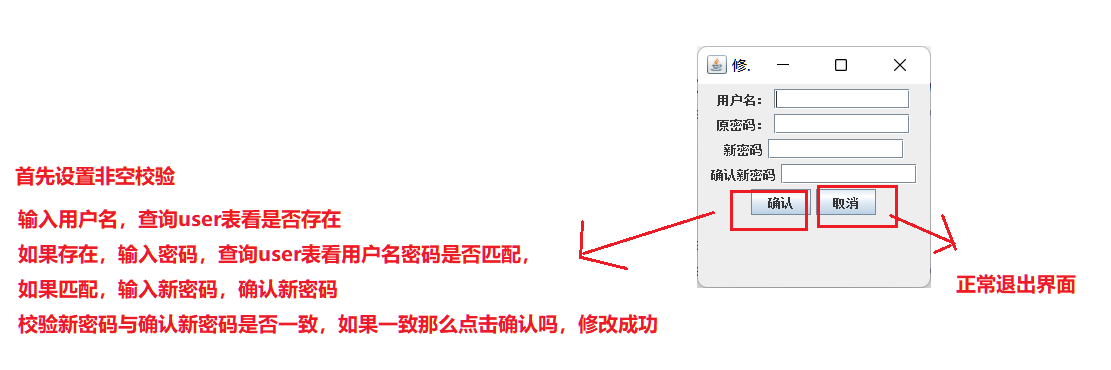
在文本框中输入用户名，点击删除按钮，在user表中对数据进行删除



**图35删除用户事件设计**

1. 修改密码功能实现，如图36所示

设置非空校验。输入用户名，查询user表看是否存在，如果存在，那么输入密码，查询user表，查看原密码与用户名是否匹配，如果匹配，输入新密码与确认新密码，如果不一致那么返回，如果一致，那么修改密码成功。



**图36用户修改密码事件设计**

# 总结

项目总结

这个swing的项目感觉总体上逻辑并不难，可能是我之前学习过后端知识的原因，很容易实现就是繁琐简单的代码得一直重复写，举个例子，jdbc所有的dao都是增删改查操作，都是按照固定套路写，以前我就感觉到了，所以我特别想把jdbc全换成mybatis实现，写个接口在加个xml配置文件搞定，很方便。老师可能考虑大家初学Java的原因不敢讲深，不过我在这个项目中该注意到点到的都有，单例模式+双重校验锁获取数据库唯一实例且线程安全，sql语句用占位符替换不用字符串拼接以防sql注入等等。还可以写一些优化代码的操作，例如可以引入lombook，实体类加注解实现getter、setter等等。

我的收获

以前我就学了Java相关的知识，目前已经到后端框架学习了，所以做这个swing项目感觉还不是太难，但是还是有收获的，在我以前看来swing已经是非常淘汰的东西了，GUI企业根本不会考察，所以我就压根没接触过，经过这个项目的完工，发现对swing的代码不能说掌握，起码算认识了，也能修修改改完成界面的布置，swing可视化图形界面还是挺有趣的。

课程建议

这个项目对初学者感觉不太友好，咱们同学刚接触java语法，可能基础还不太会，然后又接触数据库、jdbc，可能进度开的有点快。建议同学们课下一定要多多的自学才能跟得上。