

华北电力大学

课程设计报告

(2021 — 2022 年度第 1 学期)

名 称: 数据库原理课程设计

题 目: 图书管理信息系统

院 系: 控制与计算机工程学院计算机系

班 级: 软件工程 2001 班

学 号: 220201090123

学生姓名: 杨照

指导教师: 崔克彬

设计周数: 1 周

成 绩: _____

日期: 2022 年 11 月 14 日

《数据库原理》课程设计

任 务 书

一、 目的与要求

1. 本实验是为计算机各专业的学生在学习数据库原理后，为培养更好的解决问题和实际动手能力而设置的实践环节。通过这个环节，使学生具备应用数据库原理对数据库系统进行设计的能力。为后继课程和毕业设计打下良好基础。
2. 通过该实验，培养学生在建立数据库系统过程中使用关系数据理论的能力。
3. 通过对一个数据库系统的设计，培养学生对数据库需求分析、数据库方案设计、系统编码、界面设计和软件调试等各方面的能力。是一门考查学生数据库原理、面向对象设计方法、软件工程和信息系统分析与设计等课程的综合实验。

二、 主要内容

针对一个具有实际应用场景的中小型系统（见题目附录）进行数据库设计，重点分析系统涉及的实体、实体之间的联系，实现增加、删除、更新、查询数据记录等基本操作。大致分为如下步骤：

1. 理解系统的数据库需求，分析实体及实体间联系，画出 E-R 图：
 - 1) 分析确定实体的属性和码，完成对该实体的实体完整性、用户自定义完整性的定义。
 - 2) 设计实体之间的联系，包括联系类型和联系的属性。最后画出完整的 E-R 图。
2. 根据设计好的 E-R 图及关系数据库理论知识设计数据库模式：
 - 1) 把 E-R 图转换为逻辑模式；
 - 2) 规范化设计。使用关系范式理论证明所设计的关系至少属于 3NF 并写出证明过程；如果不属于 3NF 则进行模式分解，直到该关系满足 3NF 为止，要求写出分解过程。
 - 3) 设计关系模式间的参照完整性，要求实现级联删除和级联更新。
 - 4) 用 SQL 语言完成数据库内模式的设计。
3. 数据库权限的设计：
 - 1) 根据系统分析，完成授权操作；
 - 2) 了解学习收回权限的操作。
4. 完成用户界面的设计，对重要数据进行加密。
5. 连接数据库，用宿主语言实现系统所需的各种操作：
 - 1) 实现数据记录的录入、删除、查询和修改。
 - 2) 以视图的形式完成复杂查询，比如多表、多条件等。

三、 进度计划

序号	设计(实验)内容	完成时间	备注
1	根据任务书完成信息模型（概念模型、逻辑模型、完整性、规范化）的设计，并基于选用的DBMS 实现该信息模型，然后录入初始数据	第 1 天	
2	根据任务书完成各种数据定义和数据操作，并保留所有 SQL 语句。	第 2 天	
3	数据库权限设计，用户界面设计	第 3 天	
4	用可视化开发工具环境开发学生选定的信息系统（C/S 或者 B/S 模式）	第 4 天	
5	系统的完善与验收	第 5 天	

四、 设计（实验）成果要求

1. 在 DBMS（如 oracle, SQL Server 2008/2012/2016, DB2 等）上完成完整的数据库的设计；
2. 使用可视化开发平台完成信息系统，要求可以正确运行；
3. 完成实验报告。

五、 考核方式

1. 在微机上检查数据库模式的设计、三大完整性的设计、关系属于几范式等；
2. 在微机上检查系统的运行结果，要求学生阐述使用的相关技术；
3. 实验报告的检查。

六、 题目附录

1. 图书管理信息系统
2. 物资管理信息系统
3. 汽车销售管理信息系统
4. 超市管理信息系统
5. 通讯录管理信息系统
6. 工资管理信息系统
7. 酒店管理信息系统
8. 小区物业管理信息系统
9. 大学生个人消费管理系统
10. 学生成绩管理系统

学生姓名： 杨照

指导教师：崔克彬

2022 年 8 月 26 日

一、课程设计的目的与要求

1. 本实验是为计算机各专业的学生在学习数据库原理后，为培养更好的解决问题和实际动手能力而设置的实践环节。通过这个环节，使学生具备应用数据库原理对数据库系统进行设计的能力。为后继课程和毕业设计打下良好基础。
2. 通过该实验，培养学生在建立数据库系统过程中使用关系数据理论的能力。
3. 通过对一个数据库系统的设计，培养学生对数据库需求分析、数据库方案设计、系统编码、界面设计和软件调试等各方面的能力。是一门考查学生数据库原理、面向对象设计方法、软件工程和信息系统分析与设计等课程的综合实验。

二、设计正文

1. 系统分析

1.1 主要功能

用户（图书管理员、学生）登录，查询用户信息（管理员），查询图书信息，借书业务、还书业务（管理员）、修改用户信息（管理员）、查询借阅记录（管理员）。

1.2 模型分析

学生信息实体：属性（学号 Sno、学生姓名 Sname、性别 Ssex、所在系 Sdept）

主码：学号 Sno

自定义约束：Ssex in('男','女')

图书信息实体：属性（图书编号 Bno、图书名 Bname）

主码：图书编号 Bno

自定义约束：Bnum > 0

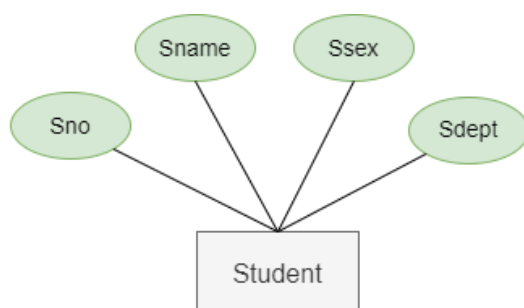
借还书联系：属性（借阅记录 Rid、学号 Sno、图书编号 Bno、借阅时间 Btime）

主码：借阅记录 Rid

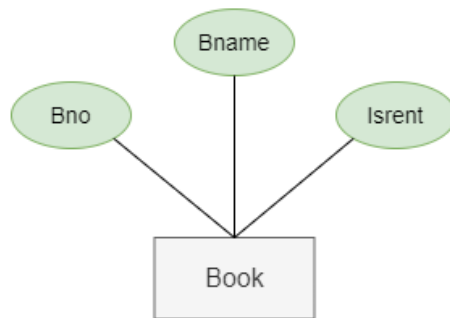
外码：Sno（references student）、Bno（references Book）

1.3 概念模型（ER 图）

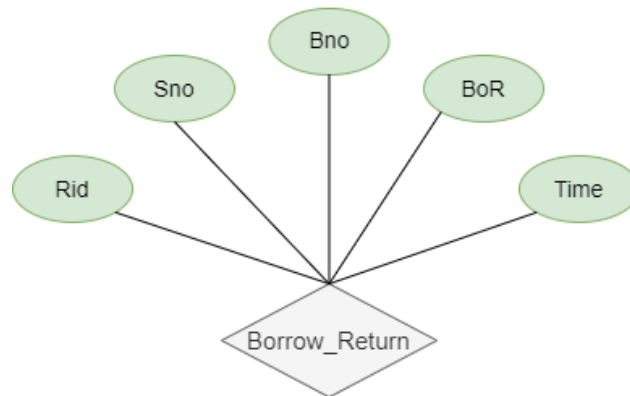
学生实体：



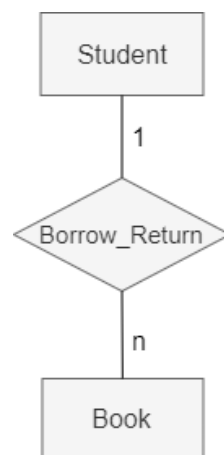
图书实体：



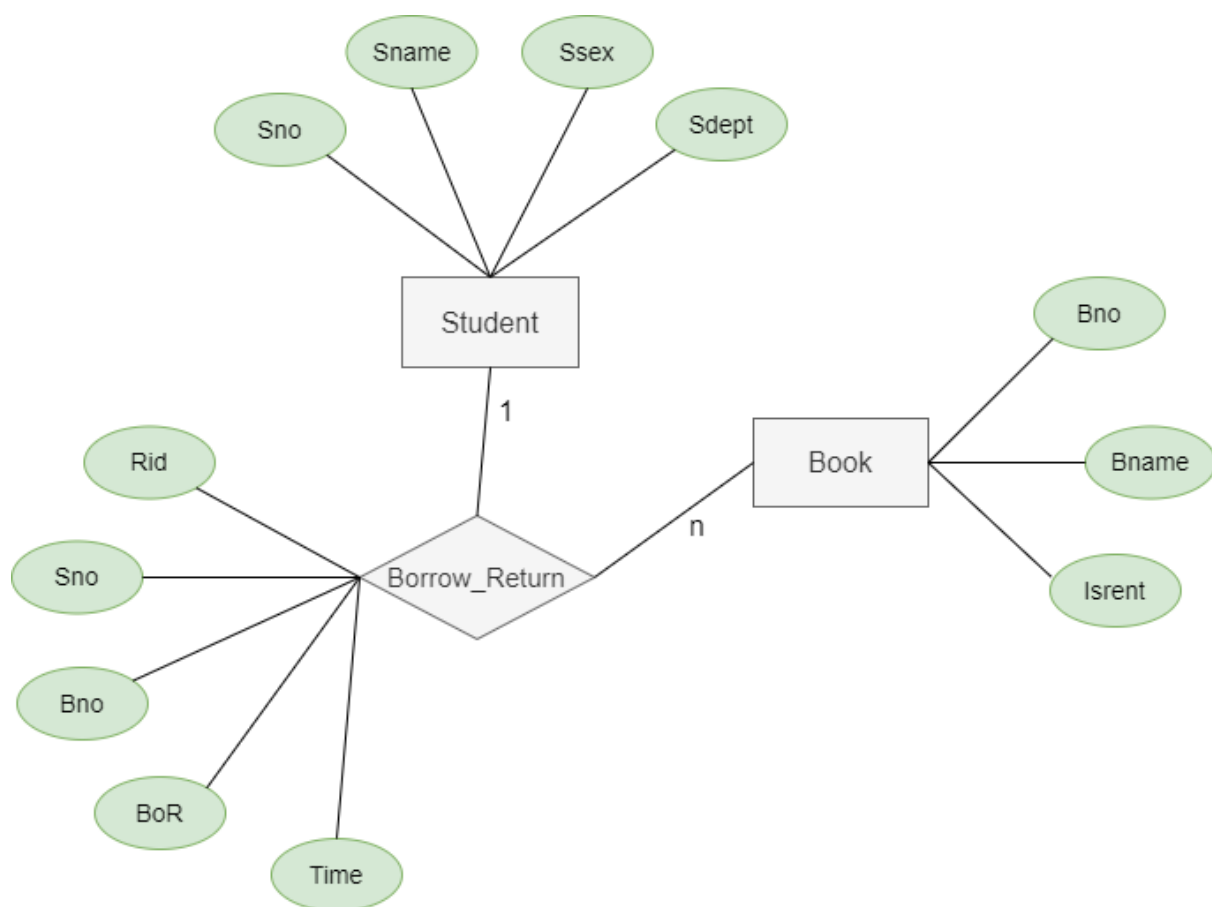
借还书联系：



总体(简)：



总体（详）：



2. 数据库设计

2.1 学生表、借阅表、图书表

	列名	数据类型	允许 Null 值
🔑	Sno	char(5)	<input type="checkbox"/>
	Sname	varchar(10)	<input checked="" type="checkbox"/>
	Ssex	varchar(2)	<input checked="" type="checkbox"/>
	Sdept	varchar(20)	<input checked="" type="checkbox"/>

	列名	数据类型	允许 Null 值
🔑	Rid	int	<input type="checkbox"/>
	Sno	char(5)	<input checked="" type="checkbox"/>
	Bno	char(5)	<input checked="" type="checkbox"/>
	BoR	varchar(4)	<input checked="" type="checkbox"/>
	Time	smalldatetime	<input checked="" type="checkbox"/>

	列名	数据类型	允许 Null 值
🔑	Bno	char(5)	<input type="checkbox"/>
	Bname	varchar(20)	<input checked="" type="checkbox"/>
	Isrent	char(2)	<input checked="" type="checkbox"/>

2.2 规范化设计

证明

关系模式 $student(\underline{Sno}, Sname, Ssex, Sdept)$ 中

$F = \{Sno \rightarrow Sname, Sno \rightarrow Ssex, Sno \rightarrow Sdept\}$

不存在非主属性对主属性的传递函数依赖

\therefore 该关系属于 3NF.

关系模式 $Borrow_Return(\underline{Rid}, Sno, Bno, BoR, Time)$ 中

$F = \{Rid \rightarrow Sno, Rid \rightarrow Bno, Rid \rightarrow BoR, Rid \rightarrow Time\}$

不存在非主属性对主属性的传递函数依赖

\therefore 该关系属于 3NF.

关系模式 $Book(Bno, Bname, Isrent)$ 中

$F = \{Bno \rightarrow Bname, Bno \rightarrow Isrent\}$

不存在非主属性对主属性的传递函数依赖

\therefore 该关系属于 3NF.

2.3 参照完整性

```
CREATE TABLE Borrow_Return(  
    Rid int primary key,  
    Sno char(5),  
    Bno char(5),  
    BoR varchar(4),  
    Time smalldatetime],  
    Foreign key(Sno) references Student(Sno),  
    Foreign key(Bno) references Book(Bno))
```

3. 数据库权限设计

3.1 权限分析

admin (管理员): 可以对图书信息、学生信息进行增删改查, 可以办理借书、还书业务, 拥有数据库的大部分权限

student (学生): 只有查看图书信息、借阅信息, 以及借阅图书的权限

3.2 权限授予

```
/**管理员权限授予**/  
grant select on booklist to admin  
grant select on studentlist to admin  
grant select, update, delete, insert on student to admin  
grant select on book to admin  
grant select, insert on borrow_return to admin  
grant select on borrowreturnrecord to admin  
grant select on rent to admin
```

/**学生权限授予**/

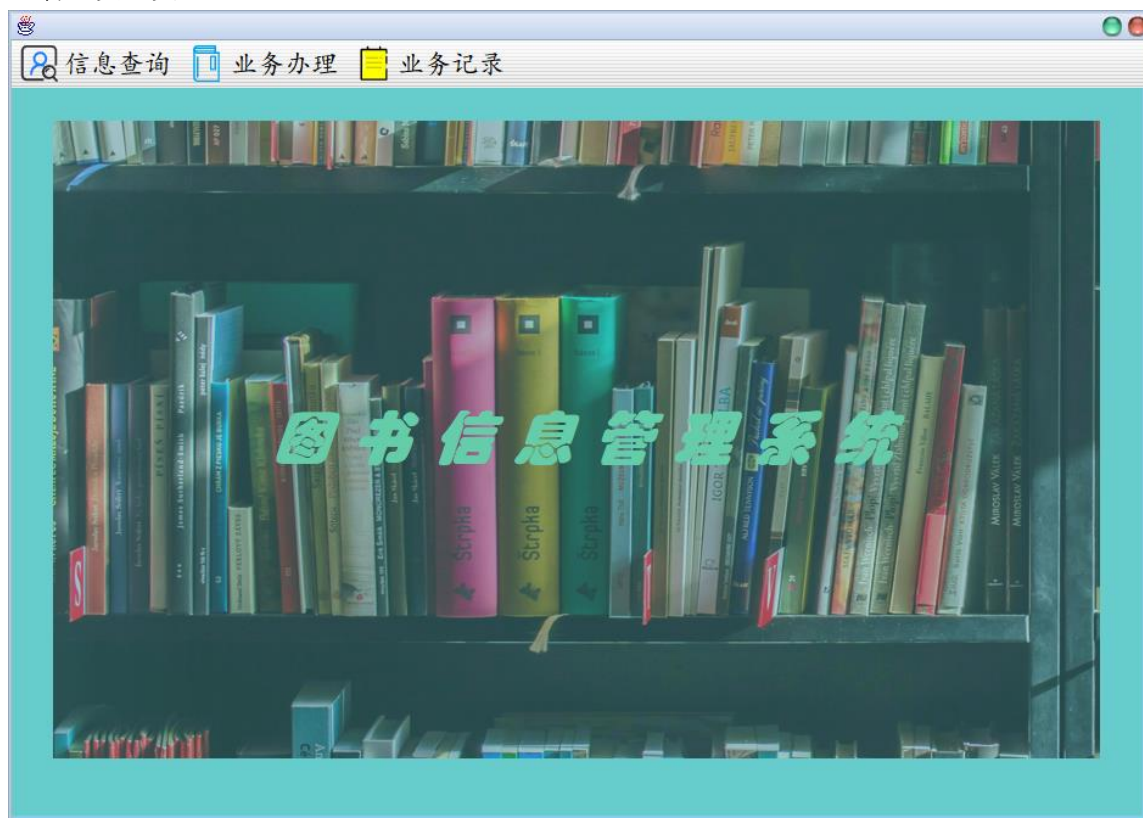
```
grant select on booklist to student  
grant select,insert on borrow_return to student  
grant select on book to student  
grant select on student to student
```

4. 用户界面设计

4.1 登录界面



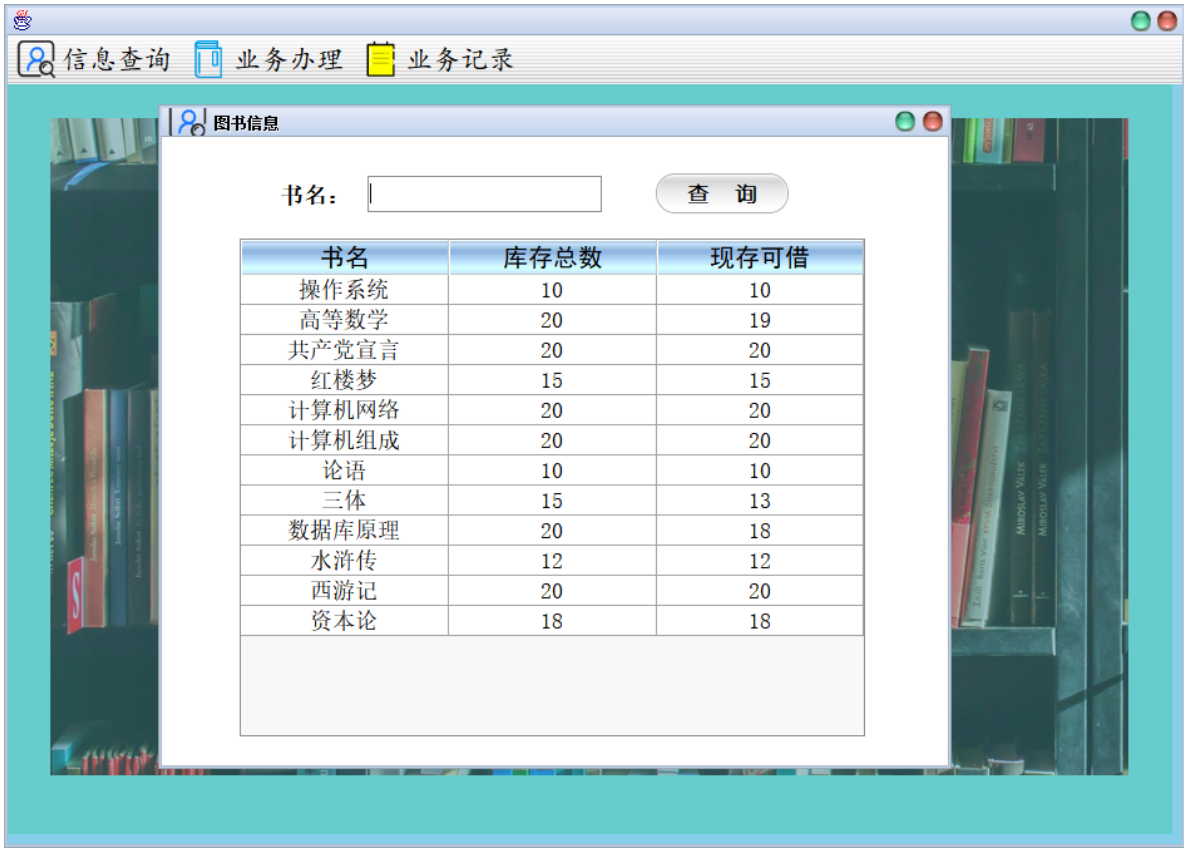
4.2 管理员主页



4.3 学生主页



4.4 图书、学生信息（管理员）



信息查询 业务办理 业务记录

学生信息

学号 姓名 性别 所在系

添加记录 查询记录

修改信息 删除记录

学号	姓名	所在系
20001	王涛	电气
20002	施博睿	电气
20003	曹行	电气
20004	褚文佳	电气
20005	孔超	电气
20006	许旭	电气
20007	曹德容	电气
20008	张勤	电气
20009	费念梅	电气
20010	赵念梅	电气
20011	童姣	电气
20012	明天春	电气
20013	朱舒	电气

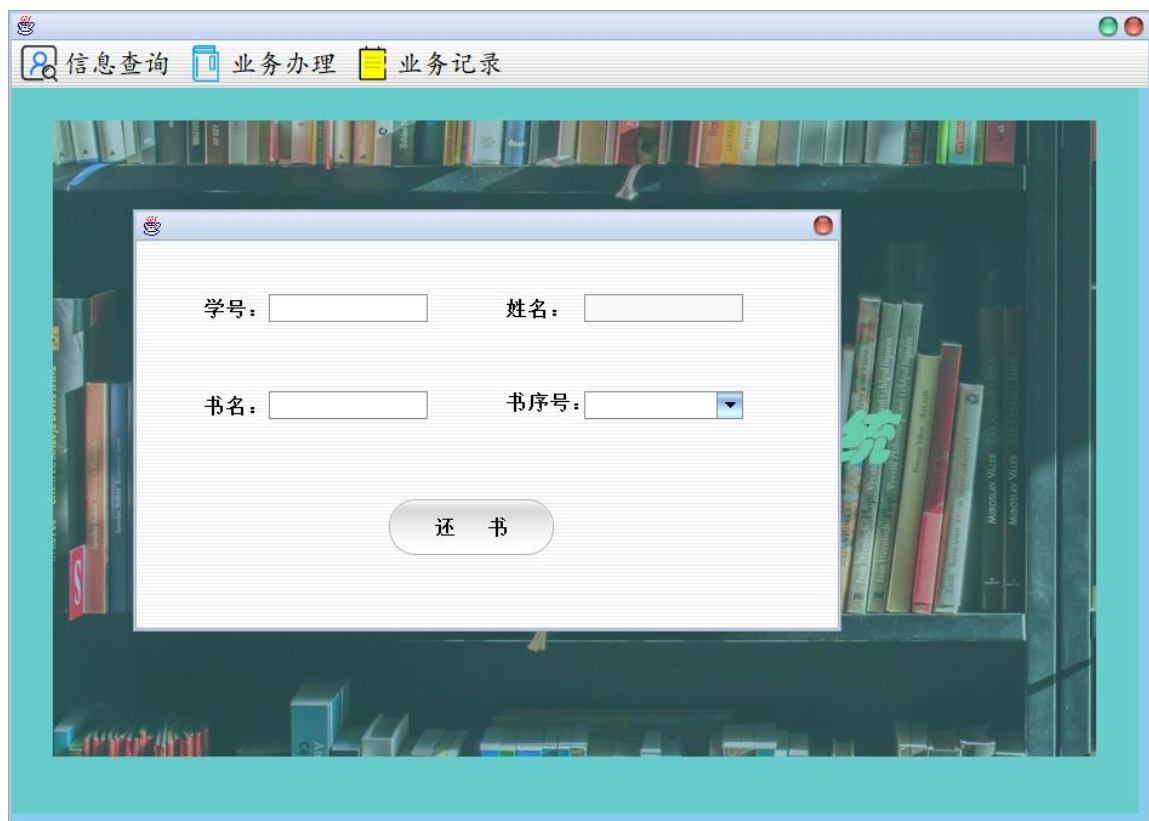
4.5 借还书业务（管理员）

信息查询 业务办理 业务记录

学号: 姓名:

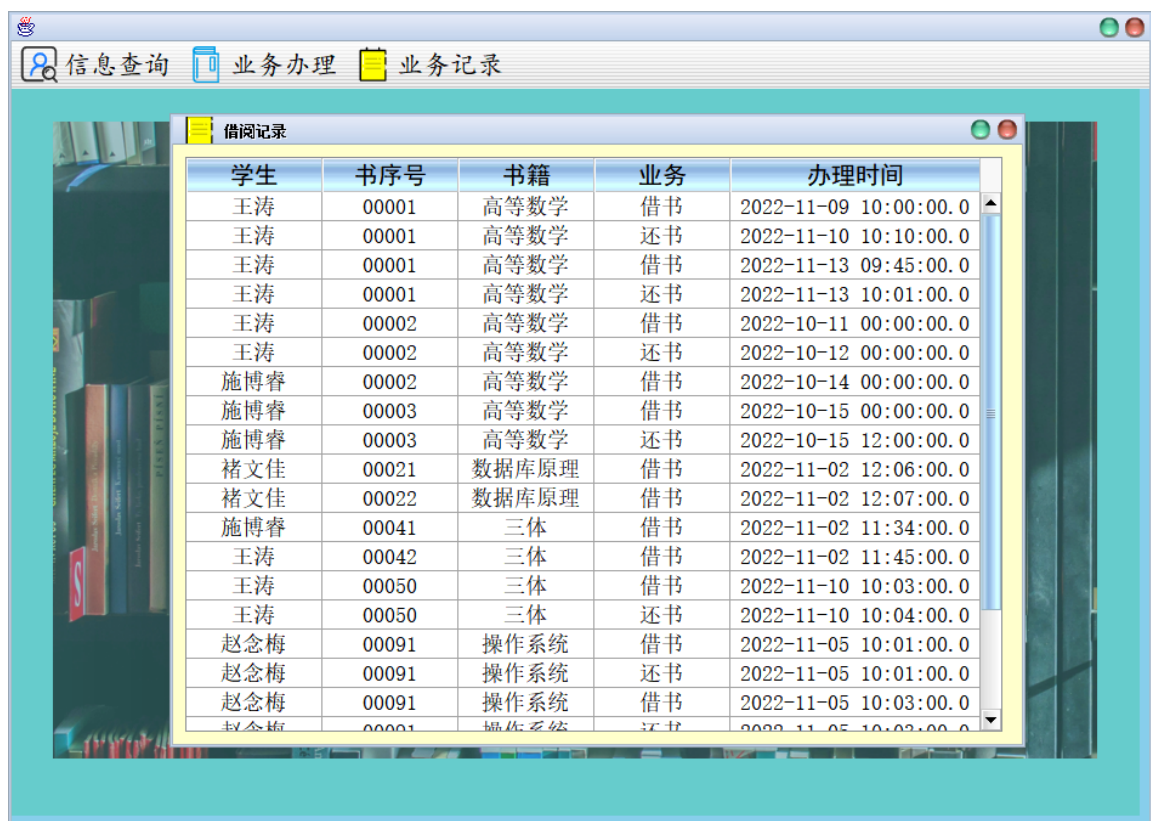
书名: 书序号:

借 阅



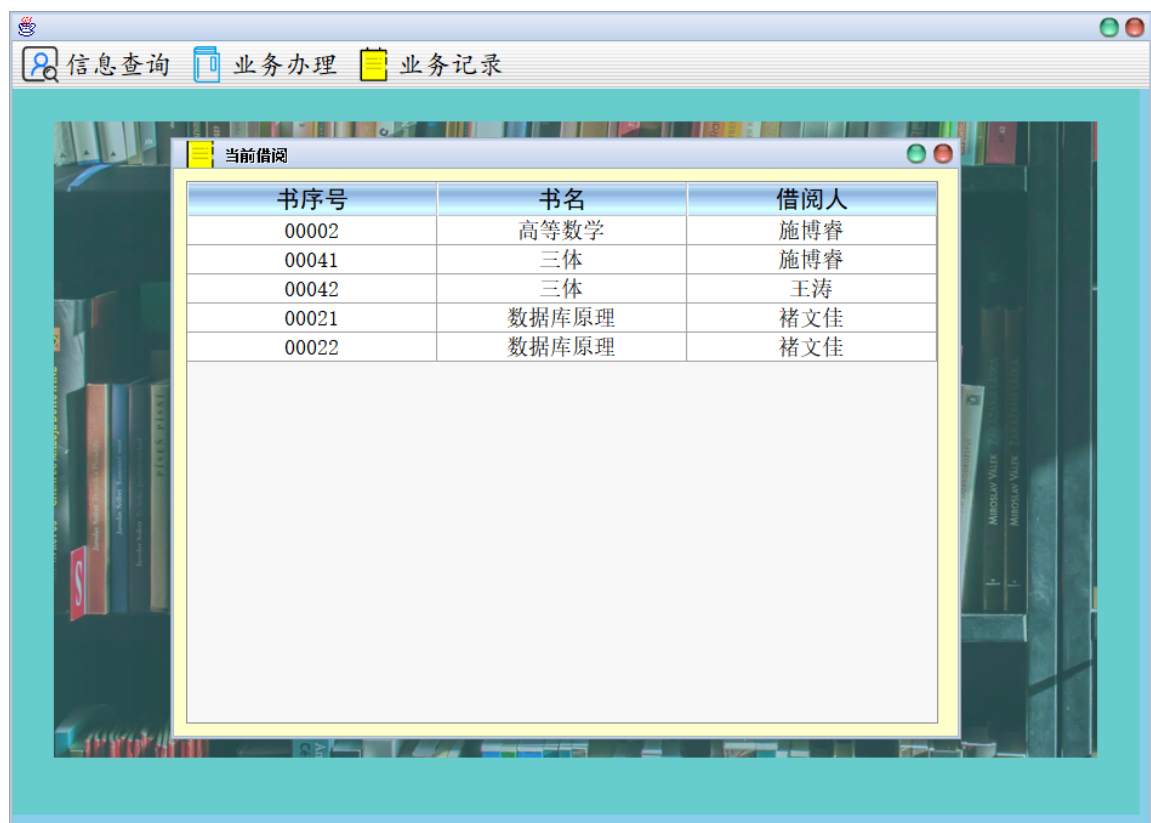
The interface shows a window titled "还书" (Return Book) with a background image of a library bookshelf. The window contains four input fields: "学号:" (Student ID), "姓名:" (Name), "书名:" (Book Name), and "书序号:" (Book Number) with a dropdown arrow. A green "还" (Return) button is positioned to the right of the "书序号:" field. Below the input fields is a large, rounded "还 书" (Return Book) button.

4.6 业务记录（管理员）

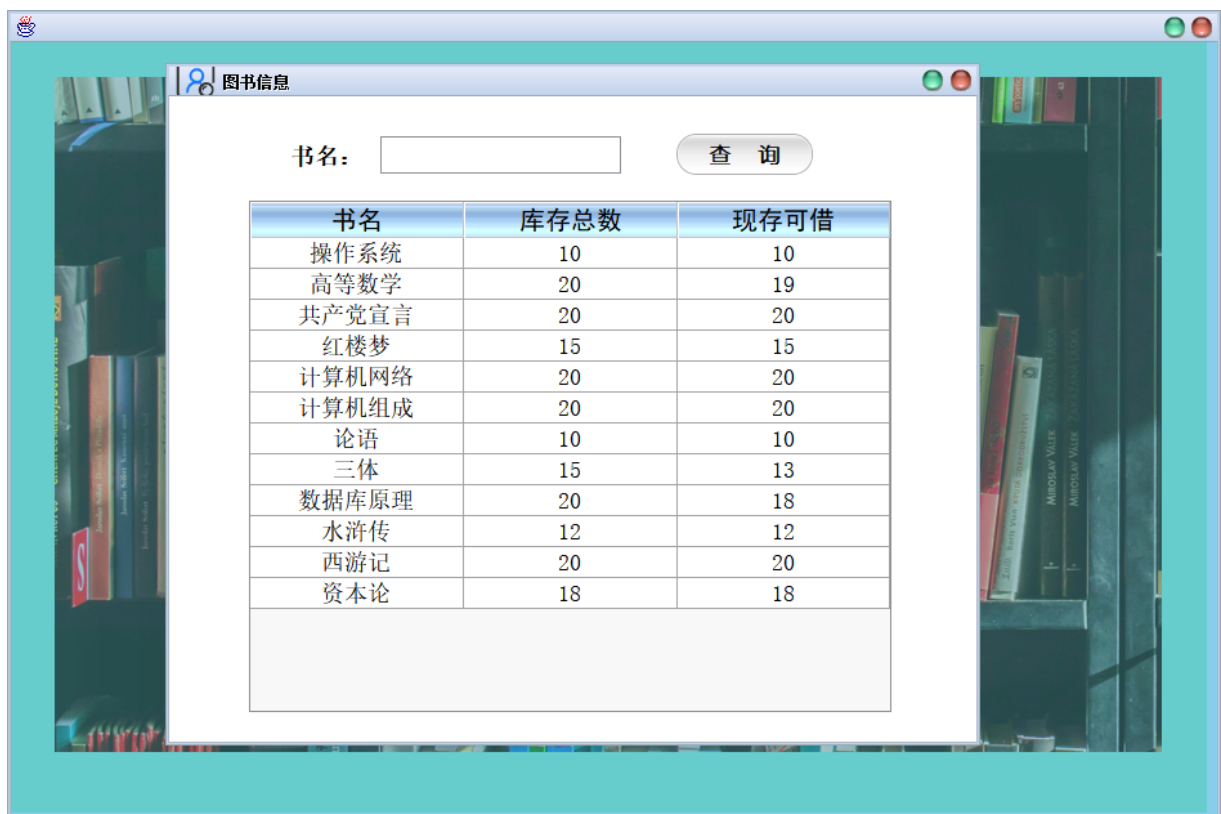


The interface shows a window titled "借阅记录" (Borrowing Record) with a background image of a library bookshelf. The window contains a table with the following data:

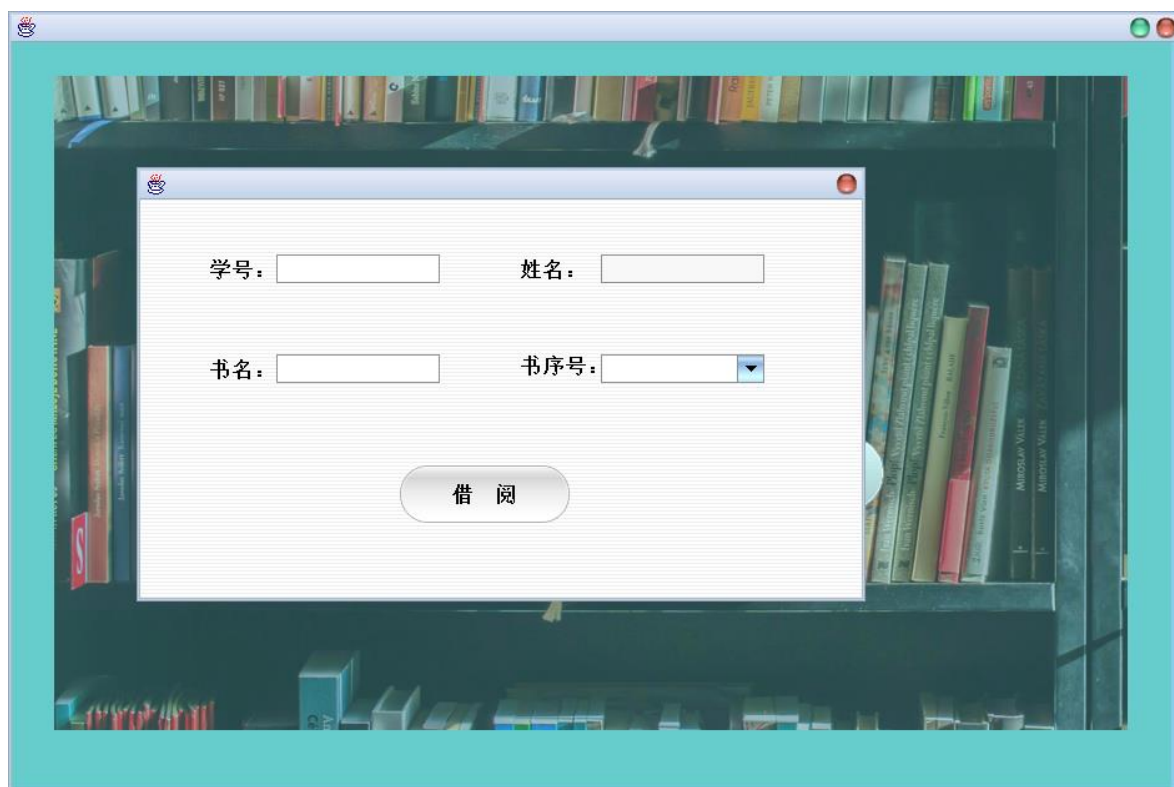
学生	书序号	书籍	业务	办理时间
王涛	00001	高等数学	借书	2022-11-09 10:00:00.0
王涛	00001	高等数学	还书	2022-11-10 10:10:00.0
王涛	00001	高等数学	借书	2022-11-13 09:45:00.0
王涛	00001	高等数学	还书	2022-11-13 10:01:00.0
王涛	00002	高等数学	借书	2022-10-11 00:00:00.0
王涛	00002	高等数学	还书	2022-10-12 00:00:00.0
施博睿	00002	高等数学	借书	2022-10-14 00:00:00.0
施博睿	00003	高等数学	借书	2022-10-15 00:00:00.0
施博睿	00003	高等数学	还书	2022-10-15 12:00:00.0
褚文佳	00021	数据库原理	借书	2022-11-02 12:06:00.0
褚文佳	00022	数据库原理	借书	2022-11-02 12:07:00.0
施博睿	00041	三体	借书	2022-11-02 11:34:00.0
王涛	00042	三体	借书	2022-11-02 11:45:00.0
王涛	00050	三体	借书	2022-11-10 10:03:00.0
王涛	00050	三体	还书	2022-11-10 10:04:00.0
赵念梅	00091	操作系统	借书	2022-11-05 10:01:00.0
赵念梅	00091	操作系统	还书	2022-11-05 10:01:00.0
赵念梅	00091	操作系统	借书	2022-11-05 10:03:00.0
赵念梅	00091	操作系统	还书	2022-11-05 10:03:00.0



4.7 馆藏检索（学生）



4.8 借阅图书（学生）



三、课程设计总结或结论

本次实验整体难度不算太大，但是考察内容十分全面，也存在很多小细节需要我们去实现，实验过程中，我遇到了许多问题，比如实验一开始，我就面临着不知道怎样通过 Java 连接数据库、怎么在 Java 环境下运行 SQL 语句，怎么更高效地实现图形界面等等，但是大部分问题通过在网上搜索资料以及询问老师，都很容易地被解决了。比较难的部分还是在于编写源代码时，有些功能可能并不复杂，但是在实验中，就需要综合考虑各种因素，比如一开始在设计 Borrow_Return 表时，我准备将 Rid 属性设置为自增长，通过触发器实现，但是运行时却不断报错，尝试了各种方法也没能解决，于是我便换了一个思路，从原先的通过 SQL 设置属性自增长改为了在 Java 中编写方法，读取 Borrow_Return 目前的最大 Rid 值，将其+1，然后通过一般的 insert 语句插入数据，实现了预期功能。

通过这次实验，我不仅进一步加深了对本学期数据库课程的学习内容的掌握，巩固了数据库的相关知识，也学习到了更多 Java 的相关内容，比如 JDBC、GUI 等等。虽然实验中遇到 bug 的时候很痛苦，不断修正错误的过程更痛苦，但在最后完成自己的系统，切身感受到自己的进步时，一切便也都值了。最后，感谢老师指出的问题以及悉心的教导和帮助！

四、参考文献

[1] 作者 1, 作者 2. 书名. 出版单位, 版本. 出版日期

附录（设计流程图、程序、表格、数据等）

1) 服务类（包含项目中用到的各种方法）

```
public class Service {
    public static String connect(String User,String Password) {
        // 连接数据库
        try {
            Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
            System.out.println("数据库驱动成功");
            String connectDB =
                "jdbc:sqlserver://localhost:1433;DatabaseName=LIBRARY";
            Connection con = DriverManager.getConnection(connectDB,
                User,Password);
            System.out.println("连接数据库成功");
        } catch (ClassNotFoundException e) {
            return "未找到驱动";
        } catch (SQLException e) {
            return "error! "+e.getMessage();
        }
        return null;
    }

    // 根据给定视图名，返回数据向量
    public static Vector<Vector<String>> show(String view) throws
        Exception {
        Connection con
            =DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databa
            seName=LIBRARY", Login.username,"123456");
```

```

Statement st=con.createStatement();
ResultSet rs=st.executeQuery("select * from "+view);
ResultSetMetaData rsmd = rs.getMetaData() ;
Vector<Vector<String>> data=new Vector<Vector<String>>();
Vector colname = null;
while(rs.next()) {
    colname=new Vector<String>();
    for(int i=1;i<=rsmd.getColumnCount();i++) {
        colname.add(rs.getString(i));
    }
    data.add(colname);
}
return data;
}

```

```

    public static Vector<String> Colname(String view) throws
Exception {
    Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;DatabaseName=LIBRARY", Login.username,"123456");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("select * from "+view);
    ResultSetMetaData rsmd = rs.getMetaData() ;
    Vector<String> colname=new Vector<String>();
    for (int i = 0; i < rsmd.getColumnCount(); i++)
        colname.add(rsmd.getColumnName(i+1));
    return colname;
}

```

// 返回表格，展示给定视图数据

```

public static JTable show_table(String view) {
    Vector<Vector<String>> data = null;
    Vector colname=new Vector<String>();
    try {
        data = show(view);
        colname=Colname(view);
    } catch (Exception e1) {
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }
    JTable table= new JTable(data,colname);
    setTableStyle(table);
    return table;
}

```

```

    }

    // 增加学生记录

    public static void insertstudent(String Sno,String
Sname,String Ssex,String Sdept) throws Exception {
        Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
eName=LIBRARY", Login.username,"123456");
        Statement st=con.createStatement();
        st.execute("insert into Student
values(\"'+Sno+'\", \"'+Sname+'\", \"'+Ssex+'\", \"'+Sdept+'\"");
    }

    // 删除学生记录

    public static void deletestudent(String Sno,String
Sname,String Ssex,String Sdept) throws Exception {
        Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
eName=LIBRARY", Login.username,"123456");
        Statement st=con.createStatement();
        String s=new String();
        if(!Sno.equals("")) {
            s=s.concat(" Sno="+\"'+Sno+'\"");
        }
        if(!Sname.equals("")) {
            if(!s.equals(""))
                s=s.concat(" and Sname="+\"'+Sname+'\"");
            else
                s=s.concat(" Sname="+\"'+Sname+'\"");
        }
        if(!Ssex.equals("")) {
            if(!s.equals(""))
                s=s.concat(" and Ssex="+\"'+Ssex+'\"");
            else
                s=s.concat(" Ssex="+\"'+Ssex+'\"");
        }
        if(!Sdept.equals("")) {
            if(!s.equals(""))
                s=s.concat(" and Sdept="+\"'+Sdept+'\"");
            else
                s=s.concat(" Sdept="+\"'+Sdept+'\"");
        }
        if(s.equals("")) {

```



```

        s=" 1=0";
    }
    st.execute("delete from Student where"+s);
}

// 查找学生记录并返回表格

    public static JTable selectstudent(String Sno,String
Sname,String Ssex,String Sdept) throws Exception {
        Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;DatabaseName=LIBRARY", Login.username,"123456");
        Statement st=con.createStatement();
        String s=new String();
        if(!Sno.equals("")) {
            s=s.concat(" Sno="+ "\"" +Sno+" \"");
        }
        if(!Sname.equals("")) {
            if(!s.equals(""))
                s=s.concat(" and Sname="+ "\"" +Sname+" \"");
            else
                s=s.concat(" Sname="+ "\"" +Sname+" \"");
        }
        if(!Ssex.equals("")) {
            if(!s.equals(""))
                s=s.concat(" and Ssex="+ "\"" +Ssex+" \"");
            else
                s=s.concat(" Ssex="+ "\"" +Ssex+" \"");
        }
        if(!Sdept.equals("")) {
            if(!s.equals(""))
                s=s.concat(" and Sdept="+ "\"" +Sdept+" \"");
            else
                s=s.concat(" Sdept="+ "\"" +Sdept+" \"");
        }
        if(s.equals("")) {
            s=" 1=1";
        }
        ResultSet rs=st.executeQuery("select Sno,Sname,Sdept from
Student where"+s);
        ResultSetMetaData rsmd = rs.getMetaData() ;
        Vector<Vector<String>> data=new Vector<Vector<String>>();
        Vector colname = null;
        while(rs.next()) {
            colname=new Vector<String>();

```

```

        for(int i=1;i<=rsmd.getColumnCount();i++) {
            colname.add(rs.getString(i));
        }
        data.add(colname);
    }
    colname=new Vector<String>();
    colname.add("学号");

    colname.add("姓名");

    colname.add("所在系");

    JTable table= new JTable(data,colname);
    setTableStyle(table);
    return table;
}

```

// 修改学生记录

```

    public static void updatestudent(String Sno,String
    Sname,String Ssex,String Sdept) throws Exception {
        Connection con =
    DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
    eName=LIBRARY", Login.username,"123456");
        Statement st=con.createStatement();
        String s=new String();
        if(Sno.equals("")) {
            return;
        }
        if(!Sname.equals("")) {
            s=s.concat(" Sname="+ "\"" +Sname+"\"");
        }
        if(!Ssex.equals("")) {
            if(!s.equals(""))
                s=s.concat(",Ssex="+ "\"" +Ssex+"\"");
            else
                s=s.concat(" Ssex="+ "\"" +Ssex+"\"");
        }
        if(!Sdept.equals("")) {
            if(!s.equals(""))
                s=s.concat(",Sdept="+ "\"" +Sdept+"\"");
            else
                s=s.concat(" Sdept="+ "\"" +Sdept+"\"");
        }
        if(s.equals("")) {

```

```

        return;
    }
    st.execute("update Student set"+s+" where
Sno=\"'+Sno+'\"");
}

// 查找图书记录并返回表格

    public static JTable selectbook(String bname) throws Exception
{
    Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
eName=LIBRARY", Login.username,"123456");
    Statement st=con.createStatement();
    String sql=new String();
    if(!bname.equals(""))
        sql="select * from booklist where 书名=\"'+bname+'\"";
    else
        sql="select * from booklist";
    ResultSet rs=st.executeQuery(sql);
    ResultSetMetaData rsmd = rs.getMetaData() ;
    Vector<Vector<String>> data=new Vector<Vector<String>>();
    Vector colname = null;
    while(rs.next()) {
        colname=new Vector<String>();
        for(int i=1;i<=rsmd.getColumnCount();i++) {
            colname.add(rs.getString(i));
        }
        data.add(colname);
    }
    colname=new Vector<String>();
    colname.add("书名");

    colname.add("库存总数");

    colname.add("现存可借");

    JTable table= new JTable(data,colname);
    setTableStyle(table);
    return table;
}

// 根据学号返回学生姓名

```

```

        public static String getsname(String sno) throws Exception {
            Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;DatabaseName=LIBRARY", Login.username,"123456");
            Statement st=con.createStatement();
            ResultSet rs=st.executeQuery("select sname from student
where sno='"+sno+"'");
            if(rs.next())
                return rs.getString(1);
            return null;
        }

// 根据图书名返回图书序号列表-未借出

        public static Vector<String> getbnolist(String bname) throws
Exception{
            Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;DatabaseName=LIBRARY", Login.username,"123456");
            Statement st=con.createStatement();
            ResultSet rs=st.executeQuery("select bno from book where
isrent='否' and bname='"+bname+"'");

            Vector<String> v=new Vector<String>();
            while(rs.next())
                v.add(rs.getString(1));
            return v;
        }

//返回目前Borrow_Return表中最大rid

        public static int getrid() throws Exception {
            Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;DatabaseName=LIBRARY", Login.username,"123456");
            Statement st=con.createStatement();
            ResultSet rs=st.executeQuery("select rid from borrow_return
where rid>= all(select rid from Borrow_Return)");
            if(rs.next())
                return rs.getInt(1);
            return -2;
        }

// 借书业务办理

```

```

        public static String borrowbook(int rid,String sno,String
bno,String bor,String time) {
            Connection con;
            try {
                con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
eName=LIBRARY", Login.username,"123456");
                Statement st=con.createStatement();
                st.execute("insert into borrow_return
values("+rid+", "+"\'"+sno+\'', "+"\'"+bno+\'', "+"\'"+bor+\'', "+"\'"+ti
me+"\'"+")");
            } catch (SQLException e) {
                return e.getMessage();
            }
            return null;
        }

```

// 根据图书名返回图书序号列表-已借出

```

        public static Vector getrentbno(String bname) throws Exception
{
            Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
eName=LIBRARY", Login.username,"123456");
            Statement st=con.createStatement();
            ResultSet rs=st.executeQuery("select bno from book where
isrent=\'是\' and bname=\'"+bname+"\'");

            Vector<String> v=new Vector<String>();
            while(rs.next())
                v.add(rs.getString(1));
            return v;
        }

```

// 还书业务办理

```

        public static String returnbook(int rid,String sno,String
bno,String bor,String time) {
            Connection con;
            String s =new String();

            s="还书失败! ";

            try {
                con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas

```

```

eName=LIBRARY", Login.username, "123456");
        Statement st=con.createStatement();
        st.execute("insert into borrow_return
values("+rid+", "+"\'"+sno+\'', "+"\'"+bno+\'', "+"\'"+bor+\'', "+"\'"+ti
me+\'"+")");
        s="";
    } catch (SQLException e) {
        return e.getMessage();
    }
    return s;
}

// 新书入库

    public static void newbook(String bname,int num) throws
Exception {
        Connection con =
DriverManager.getConnection( "jdbc:sqlserver://localhost:1433;Databas
eName=LIBRARY", "yangxz", "123456");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("select bno from book where
bno>= all(select bno from book)");
        rs.next();
        String temp=rs.getString(1);
        int ti=Integer.parseInt(temp);
        for(int i=0;i<num;i++) {
            int j=ti+i+1;
            String bno=String.valueOf(j);
            String str="";
            for(int k=0;k<(5-bno.length());k++)
                str+="0";
            bno=str+bno;
            st.execute("insert into book(bno,bname)
values(\'"+bno+\'', \'"+bname+\'"+")");
        }
    }

//设置表格格式

    public static void setTableStyle(JTable jtb) {
        //设置表格每行的高度
        jtb.setRowHeight(24);
        // 设置表格中的数据居中显示

```

```

        DefaultTableCellRenderer r=new DefaultTableCellRenderer();
        r.setHorizontalAlignment(JLabel.CENTER);
        jtb.setDefaultRenderer(Object.class,r);

        // 设置表头格式

        Dimension size = jtb.getTableHeader().getPreferredSize();
        size.height = 28;
        jtb.getTableHeader().setPreferredSize(size);
        jtb.getTableHeader().setFont(new Font("黑体",Font.PLAIN,20));

        // 设置表头文字居中显示

        DefaultTableCellRenderer renderer =
        (DefaultTableCellRenderer) jtb.getTableHeader().getDefaultRenderer();
        renderer.setHorizontalAlignment(renderer.CENTER);

        // 设置点击表头自动实现排序

        jtb.setAutoCreateRowSorter(true);
        jtb.setFocusable(false);

        jtb.setFont(new Font("新宋体", Font.PLAIN, 18));
    }
}

```

（其他类基本都是采用调用 Service 类的方法+事件监听等实现相应功能，下面只以管理员主界面为例，不再一一展示）

2) 管理员主界面关键代码

```

public MainView() {    // 主界面

    // 查看图书信息

    mntmNewMenuItem.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            BookInformation book=new BookInformation();
            desktopPane.add(book);
            book.setVisible(true);
        }
    });

    JMenuItem mntmNewMenuItem_3 = new JMenuItem("学生信息 ");
}

```

```
// 查看学生信息

mntmNewMenuItem_3.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        StudentInformation student=new StudentInformation();
        desktopPane.add(student);
        student.setVisible(true);
    }
});

// 借书业务

JMenuItem mntmNewMenuItem_1 = new JMenuItem("借书 ");

mntmNewMenuItem_1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        BorrowBook bb=new BorrowBook();
        desktopPane.add(bb);
        bb.setVisible(true);
    }
});

// 还书业务

JMenuItem mntmNewMenuItem_2 = new JMenuItem("还书 ");

mntmNewMenuItem_2.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        ReturnBook rb=new ReturnBook();
        desktopPane.add(rb);
        rb.setVisible(true);
    }
});

// 新书入库

JMenuItem mntmNewMenuItem_2_1 = new JMenuItem("新书入库");

mntmNewMenuItem_2_1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        NewBook nb=new NewBook();
        desktopPane.add(nb);
        nb.setVisible(true);
    }
});
```



```

JMenu mnNewMenu_2 = new JMenu("业务记录");

JMenuItem mntmNewMenuItem_1_1 = new JMenuItem("历史记录");

// 查看历史借还书记录

mntmNewMenuItem_1_1.addActionListener(new ActionListener() {

    public void actionPerformed(ActionEvent e) {
        BorrowReturnRecord bd=new BorrowReturnRecord();
        desktopPane.add(bd);
        bd.setVisible(true);
    }
});

mntmNewMenuItem_1_1.setFont(new Font("宋体", Font.PLAIN, 16));

mnNewMenu_2.add(mntmNewMenuItem_1_1);

JMenuItem mntmNewMenuItem_1_1_1 = new JMenuItem("当前借阅");

// 查看当前图书借阅情况

mntmNewMenuItem_1_1_1.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        RentRecord bd;
        try {
            bd = new RentRecord();
            desktopPane.add(bd);
            bd.setVisible(true);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
});

```