

1. 实验目的

数据库原理课程设计时计算机科学与技术专业几种实践性环节之一，旨在让我们加深对数据库基础理论和基本知识的理解，掌握设计数据库管理系统的基本方法，锻炼运用知识解决实际问题的动手能力。通过完成从用户需求分析、概念结构设计、逻辑结构设计等一系列的数据库设计到编程、调试和应用等全过程，进一步理解和掌握教材中的相关内容。

2. 概要设计

2.1. 引言

图书馆作为一种信息资源的集散地，对图书的高效管理是一项重大需求，传统的基于文本、表格等纸介质的手工处理，由于数据处理工作量大，容易出错；且由于数据繁多，容易丢失，不易查找。尽管有的图书馆由计算机，但是尚未用于信息管理，没有发挥它的效力，资源限制比较突出，这就是信息管理系统的开发的基本环境。

图书管理系统中，需要为管理员建立一个账户，账户中存储了图书类别和各个图书类别下的对应的图书的各种信息。读者不直接与系统进行交互，而是管理员充当读者的代理与系统进行交互。在进行图书的管理时，第一步需要输入管理员信息来验证身份，之后对应所需要检索的是图书类别还是某一类别下的某本图书进行操作，在添加、删除时进行条件的判断，条件满足时，请求才被接受，才能成功的执行所做的操作。

调查各种图书馆的图书及其类别的组织机构的总体状况以及管理员需要管理的具体信息，管理系统需要支持对图书类别的增添删改查功能以及对图书的增添删改查功能。

2.2. 系统任务

调查方法：结合网上资料和生活环境中的实际应用，对成熟商家的模式考察。
主要任务：设计完整的信息数据库系统。分别为管理员设计了功能完备、安全性较高、一致性优秀的数据库管理系统。完成各项基于信息管理系统的数据库操作，并根据系统操作特点，采用一些数据库技术对数据库操作或数据库运行进行优化，使其更加高效地运行。

数据库完整性要求：各个数据库表在主码上不能为空，要符合实体完整性和参照完整性。

数据安全性要求：对于一个数据表，有时不能使数据库用户可以看到这个数据表上的全部信息，可以通过建立视图的方式，与授权结合实现数据库的安全性，由于时间关系，本项目只实现了管理员相关的功能，用户相关的并未实现。

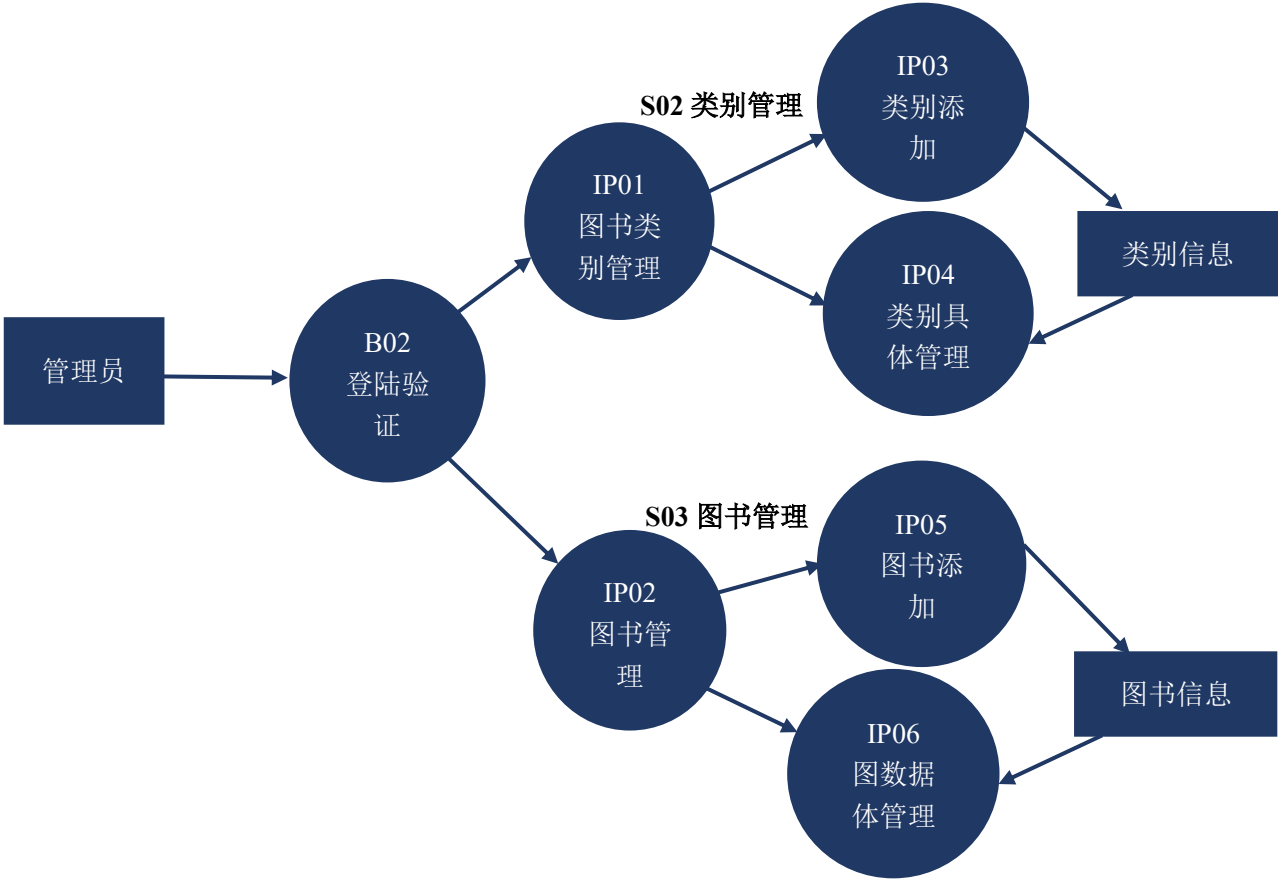
一致性要求：事物的执行结果必须是数据库从一个一致性状态转变到另一个一致性状态。在使用主语言操作数据库时，不满足数据库完整性约束的操作会被数据库拒绝，如果在主语言中没有及时发现该问题，会导致主程序与数据库的不一致，因此在编写系统时，需要严格要求对数据库的操作，当且仅当数据全部合法时才能操作数据库。

2.3. 数据流图

顶层数据流图：



0 层数据流图：



2.4. 数字字典：

2.4.1. 数据加工：

元素编号	名称	类型	说明
B01	管理系统	处理	对图书类别和图书进行管理的系统加工。
S01	管理员信息	数据流	管理员的信息
S02	类别信息	数据流	图书类别的名称以及描述。
S03	图书信息	数据流	图书的名称、作者、作者性别、价格以及图书描述信息。
IP01	类别管理	处理	分为图书类别添加和类别维护两个模块。
IP02	图书管理	处理	分为图书添加和图书维护两个模块。

IP03	类别添加	处理	添加类别的名称以及描述。
IP04	类别具体管理	处理	对类别的具体信息进行查询以及增删改添及查询。
IP05	图书添加	处理	对图书的名称、作者、作者性别、价格、类别及其相关描述信息进行添加。
IP06	队员具体管理	处理	对图书的具体信息进行增删改添及查询。

2.4.2. 数据流：

数据流名称：S01 管理员信息

数据项	数据类型	长度	备注
id	Int	整型	管理员编号
userName	Varchar	20	管理员账号
password	Varchar	20	管理员密码

数据流名称：S02 类别信息

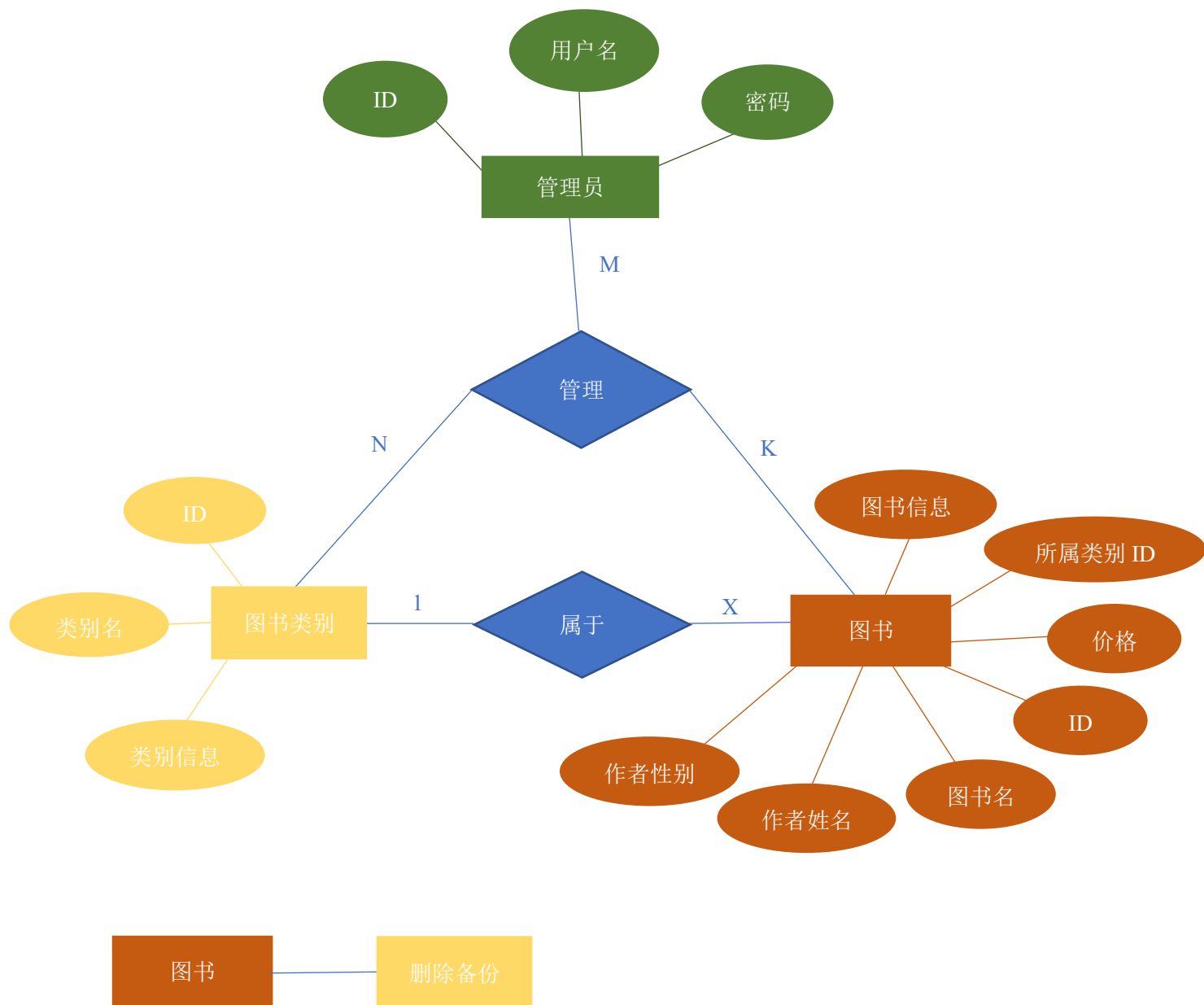
数据项	数据类型	长度	备注
Id	Int	整型	图书类别编号
bookTypeName	Varchar	45	图书类别名称
bookTypeDesc	Varchar	1000	图书类别描述

数据流名称：S03 队员信息

数据项	数据类型	长度	备注
Id	Int	整型	图书编号
bookName	Varchar	20	图书名称
author	Varchar	20	图书作者姓名
sex	Varchar	10	图书作者性别
price	Float	浮点型	图书价格
bookTypeId	Int	整型	所属类别编号
bookDesc	Varchar	1000	图书具体描述

2.5. 数据库概念设计：

通过之前的需求调研，基本上了解了图书管理系统所需要的数据，再结合构建出的数据流图和数据字典，基本明白数据所需的存储方式和数据之间的关系结构需求，通过将这些需求进行数据抽象，抽象出“管理员”、“图书类别”、“图书”三个实体。再通过每个实体的特点以及系统的需求抽象出每个实体的组成成分，以实体“图书”为例，可抽象出其属性：ID、图书名称、图书作者、作者性别、图书价格、图书类别、图书描述。最后，通过现实需求，确定实体间的联系以及其结构约束，共抽象出以下联系：管理员对图书类别的多对多联系、管理员对图书的多对多联系、图书类别对图书的一对多联系。最根据实体间的联系，设计出最终的 ER 图。最终得到的 ER 图如下图所示：



2.6. 数据库逻辑设计

数据库逻辑设计主要思想是将概念设计阶段得到的 ER 图转化为关系模式，并进行规范化和泛化，并为应用设置外模式。

首先根据 ER 图，将实体集和联系集分别转化成关系模式，而后合并拥有相同码的关系模式形成新的关系模式，并根据语义进行命名。

设计出的关系模式如下：

管理员（管理员编号，姓名，密码）

图书类别（类别编号，名称，类别信息）

说明：图书类别的编号设计成主键自增，即使某一类别被删除，后来新添加的类别也不会填充到由于删除类别而空闲出的编号中，而是会在当前已创建的类别的最大编号(包括已删除的)的基础上加一；在删除类别时会进行判断，如果当前类别下有图书，则不能删除该类别，并发出提示。

图书（图书编号，名称，作者姓名，作者性别，价格，所属类别 id，图书信息）

说明：图书所属的类别是外键，依赖于图书类别表的主键，用于连接图书类别表，便于在添加和删除等操作时的判断和统一管理。

通过语义推敲每个表的函数依赖集，并进行范式验证，可以得出每个关系的主属性和非主属性都依赖于码，因此关系模式符合 BCNF。

2.7. 数据库物理设计

基本思想：数据库的物理结构设计，就是为一个给定数据库的逻辑结构选取一个最适合应用环境的物理结构和存取方法的过程，其目的是提高数据库的访问速度并有效地利用存储空间。

索引：索引属于数据库的物理结构，起作用类似于书本的目录。通过在频繁检索的字段闪光增加索引，可以提高数据库检索的效率。但是，索引并不是越多越好，其会增加空间开销，不必要的索引反而会降低数据库的效率。但本系统数据量相对较少，且检索多为主码，所以并无此方面的顾虑。

脚本描述：

主键约束：管理员的主键为管理员编号，图书类别的主键为类别编号，图书的主键为图书编号。

外键约束：需要一个外键 con 将图书的所属类别 id 外联到图书类别的 id 中。

触发器：主要用 JAVA 语言实现，使用 JOptionpane 中的 showconfirmdialog 来触发信息，比如在出现错误的时候提示信息；在图书表中设计一个触发器，每当删除一个图书，都会在 delete_book 表中添加该图书，实现该图书的备份：

```
225  /**
226   * 图书类别删除事件处理
227   * @param e
228   */
229  private void bookTypeDeleteActionEvent(ActionEvent evt) {
230      // TODO Auto-generated method stub
231      String id=idTxt.getText();
232      if(StringUtil.isEmpty(id)) {
233          JOptionPane.showMessageDialog(null, "请选择要删除的记录");
234          return;
235      }
236      int n=JOptionPane.showConfirmDialog(null, "确定要删除该记录吗?");
237      if(n==0) {
238          Connection con=null;
239          try {
240              con=dbUtil.getCon();
241              boolean flag=bookDao.existBookByBookTypeId(con, id);
242              if(flag) {
243                  JOptionPane.showMessageDialog(null, "当前图书类别下有图书，不能删除此类别!");
244                  return;
245              }
246              int deleteNum=bookTypeDao.delete(con, id);
247              if(deleteNum==1) {
248                  JOptionPane.showMessageDialog(null, "删除成功!");
249                  this.resetValue();
250                  this.fillTable(new BookType());
251              }else {
252                  JOptionPane.showMessageDialog(null, "删除失败");
253              }
254          }catch(Exception e){
255              e.printStackTrace();
256              JOptionPane.showMessageDialog(null, "删除失败");
257          }
258      }
```

```
mysql> DELIMITER $
mysql> CREATE TRIGGER before_delete_t_book
-> BEFORE DELETE
-> ON t_book FOR EACH ROW
-> BEGIN
-> INSERT INTO delete_book(id,bookName,author,sex,price,bookTypeId,bookDesc)
-> VALUES (OLD.id, OLD.bookName, OLD.author, OLD.sex, OLD.price, OLD.bookTypeId, OLD.bookDesc);
-> END $
Query OK, 0 rows affected (0.04 sec)
```

与 mysql 语言中的 showmessge 报异常时一样的。每一个弹窗都是用 showmessagedialog 来进行触发结果显示。

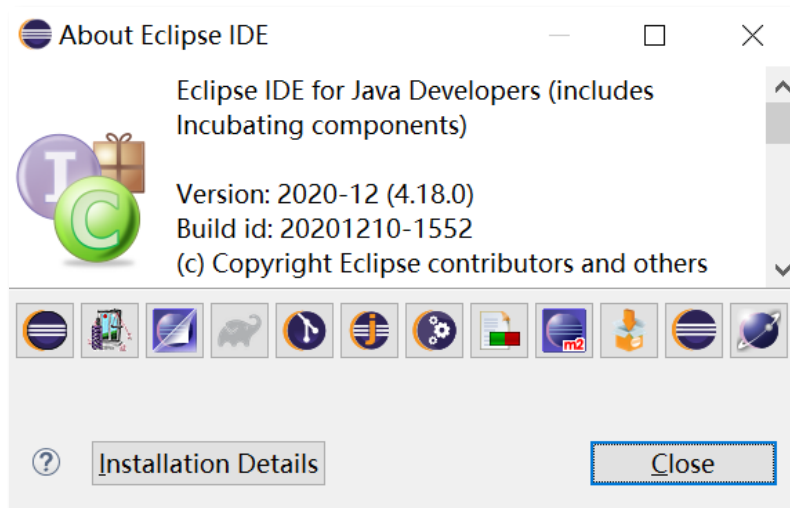
Java 语言实现的存储过程：给每一个 jbutton 设置事件 actionPerformed，当 actionPerformed 触发，然后调用 dao 类中的相关 sql 代码，即相当于运行存储过程，以次来执行相关操作。

索引：在图书表中添加 bookid 索引以提高搜索效率。

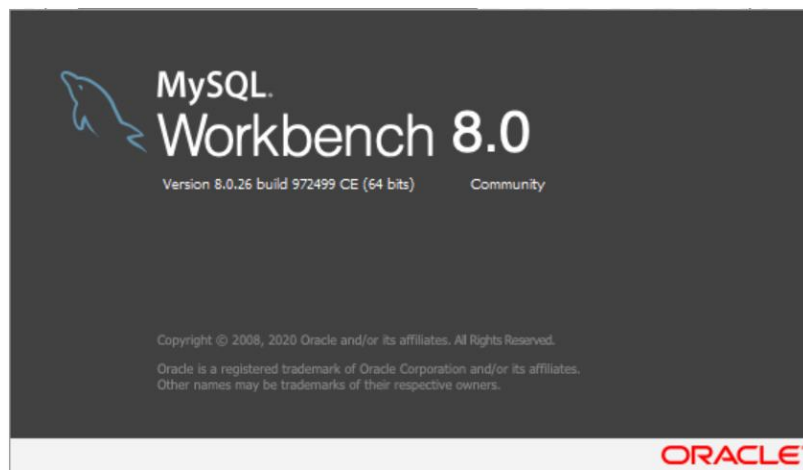
3. 开发环境需求

语言：java

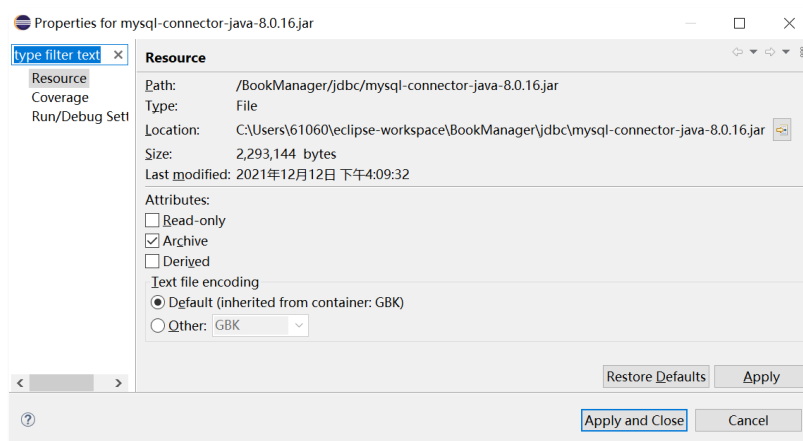
平台：eclipse，具体版本如下图所示：



数据库：mysql，具体版本如下图所示：



JDBC：具体版本如下图所示：



4. 具体实现

具体代码请见附件。

4.1. com.LiQi.dao:

```
▼ com.LiQi.dao
  > BookDao.java
  > BookTypeDao.java
  > UserDao.java
```

BookDao.java 中包含图书添加、图书信息查询、图书信息删除、图书信息修改、指定图书类别下是否存在图书等功能的实现，在连接数据库后使用 sql 语言来实现操作。以图书信息查询为例，先设置基本查询语句，在进行判断，根据查询的属性类别添加不同的 sql 语句，以此来实现最终的查询：

```
/**
 * 图书信息查询
 * @param con
 * @param book
 * @return
 * @throws Exception
 */
public ResultSet list(Connection con, Book book) throws Exception {
    StringBuffer sb = new StringBuffer("select * from t_book b, t_bookType bt where b.bookTypeId=bt.id");
    if (StringUtil.isEmpty(book.getBookName())) {
        sb.append(" and b.bookName like '%" + book.getBookName() + "%'");
    }
    if (StringUtil.isEmpty(book.getAuthor())) {
        sb.append(" and b.author like '%" + book.getAuthor() + "%'");
    }
    if (book.getBookTypeId() != null && book.getBookTypeId() != -1) {
        sb.append(" and b.bookTypeId=" + book.getBookTypeId());
    }
    PreparedStatement pstmt = con.prepareStatement(sb.toString());
    return pstmt.executeQuery();
}
```

BookTypeDao.java 包含图书类别添加、图查询图书类别集合、删除图书类别、更新图书类别等功能的实现，同样在连接数据库后使用 sql 语言来实现操作。以图书类别添加为例，先设置基本查询语句，根据输入的顺序来分配字符串到不同的属性变量中，以此来实现最终的正确添加：

```
public class BookTypeDao {
    /**
     * 图书类别添加
     * @param con
     * @param bookType
     * @return
     * @throws Exception
     */
    public int add(Connection con, BookType bookType) throws Exception {
        String sql = "insert into t_bookType values(null,?,?)";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, bookType.getBookTypeName());
        pstmt.setString(2, bookType.getBookTypeDesc());
        return pstmt.executeUpdate();
    }
}
```

UserDao.java 实现了管理员的登陆验证功能，同样通过连接数据库使用 sql 语句进行登录：


```

public class UserDao {

    /**
     * 登陆验证
     * @param con
     * @param user
     * @return
     * @throws Exception
     */
    public User login(Connection con, User user) throws Exception {
        User resultUser = null;
        String sql = "select * from t_user where userName=? and password=?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, user.getUserName());
        pstmt.setString(2, user.getPassword());
        ResultSet rs = pstmt.executeQuery();
        if (rs.next()) {
            resultUser = new User();
            resultUser.setId(rs.getInt("id"));
            resultUser.setUserName(rs.getString("userName"));
            resultUser.setPassword(rs.getString("password"));
        }
        return resultUser;
    }
}

```

4.2. com.LiQi.model:

```

v com.LiQi.model
> Book.java
> BookType.java
> User.java

```

其中包含了图书类别实体、图书实体、用户实体三个实体，各自的属性如下所示：

```

/**
 * 图书实体
 * @author 61060
 */
public class Book {
    private int id;
    private String bookName;
    private String author;
    private String sex;
    private Float price;
    private Integer bookTypeId;
    private String bookTypeName;
    private String bookDesc;
}

```

```

public class BookType {
    private int id;
    private String bookTypeName;
    private String bookTypeDesc;
}

```

```
public class User {  
    private int id; //编号  
    private String userName;  
    private String password;
```

4.3. com.LiQi.util:

```
▼ com.LiQi.util  
  > DbUtil.java  
  > StringUtil.java
```

其中包含了数据库工具类(DbUtil.java)和字符串工具类(StringUtil.java)，其中数据库工具类用来获取数据库连接和关闭数据库连接，字符串工具类用于判断是否为空。当时在连接数据库的是偶遇到了一些小问题，在稍后的部分给出具体的问题及解决方案：


```
package com.LiQi.util;  
  
import java.sql.Connection;  
  
/**  
 * 数据库工具类  
 * @author 61060  
 */  
public class DbUtil {  
    private String dbUrl="jdbc:mysql://localhost:3306/db_book?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";  
    private String dbUserName="root";  
    private String dbPassword="Liqizuihaokan1";  
    private String jdbcName="com.mysql.cj.jdbc.Driver";  
  
    /**  
     * 获取数据库连接  
     * @return  
     * @throws Exception  
     */  
    public Connection getCon()throws Exception{  
        Class.forName(jdbcName);  
        Connection con = DriverManager.getConnection(dbUrl,dbUserName,dbPassword);  
        return con;  
    }  
  
    /**  
     * 关闭数据库连接  
     * @param con  
     * @throws Exception  
     */  
    public void closeCon(Connection con)throws Exception{  
        if(con!=null) {  
            con.close();  
        }  
    }  
  
    public static void main(String[] args) {  
        DbUtil dbUtil = new DbUtil();  
        try {  
            dbUtil.getCon();  
            System.out.println("数据库连接成功!");  
        } catch (Exception e) {  
            // TODO Auto-generated catch block  
            e.printStackTrace();  
            System.out.println("数据库连接失败");  
        }  
    }  
}
```

```
package com.LiQi.util;
/**
 * 字符串工具类
 * @author 61060
 */
public class StringUtil {
    /**
     * 判断是否为空
     * @param str
     * @return
     */
    public static boolean isEmpty(String str) {
        if(str==null || "".equals(str.trim())) {
            return true;
        }else {
            return false;
        }
    }
    /**
     * 判断是否不是空
     * @param str
     * @return
     */
    public static boolean isNotEmpty(String str) {
        if(str!=null && !"".equals(str.trim())) {
            return true;
        }else {
            return false;
        }
    }
}
```

4.4. com.LiQi.view:

```
▼ com.LiQi.view
  > BookAddInterFrm.java
  > BookManagelInterFrm.java
  > BookTypeAddInterFrm.java
  > BookTypeManagelInterFrm.java
  > LiQiInterFrm.java
  > LogOnFrm.java
  > MaimFrm.java
```

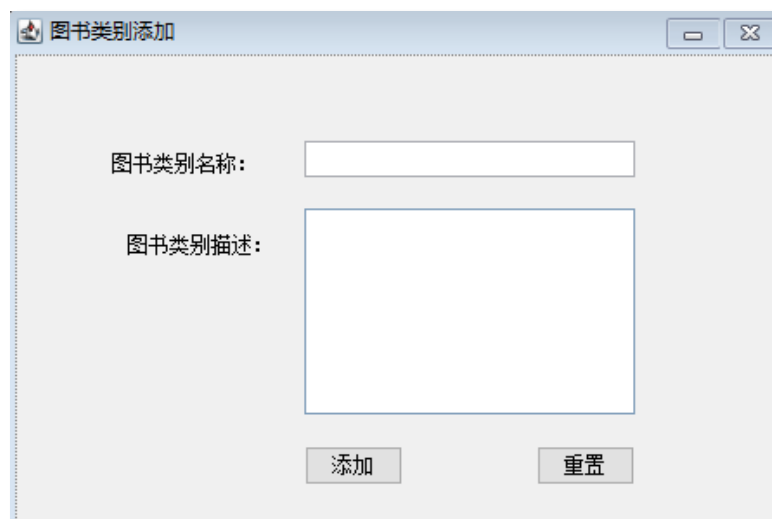
`BookAddInterFrm.java` 是实现图书添加相关事件的处理的图形界面设计模块，具体界面如下图所示，其中实现了图书添加事件处理、重置表单、初始化图书类别下拉框等功能。图书添加事件处理模块主要是实现在输入必要的图书信息后，点击添加按钮，能够成功的在数据库中添加相应图书，重置表单主要是实现将当前界面已经输入的信息清除，以便重新输入新的图书信息，初始化图书类别下拉框用来实现在下拉框中显示当前已经存在的图书类别并可以选择的功能：



BookManageInterFrm.java 是实现图书具体管理功能的图形界面设计模块，具体界面如下图所示，其中实现了图书删除事件处理、图书修改时间处理、重置表单、表格行点击事件处理、图书查询事件处理、初始化下拉框、初始化表格数据等功能。图书删除事件处理实现了对选定图书的删除；图书修改事件处理实现了对选定图书的修改，注意这里图书的编号我们人为设置成了不可修改；重置表单实现了对当前输入的清除功能；表格行点击事件实现了对图书列表中的具体图书的选择，在点击选定图书后，在界面下方会显示该图书的信息，在这里对图书信息进行修改；图书查询事件处理实现了界面上方对要查询的图书信息进行输入，点击查询按钮后下方的表格数据仅会显示被查询图书的功能；初始化图书类别下拉框用来实现在下拉框中显示当前已经存在的图书类别并可以选择的功能；初始化表格数据实现了对现有的所有图书在表格中显示的功能：

编号	图书名称	图书作者	作者性别	图书价格	图书描述	图书类别
----	------	------	------	------	------	------

BookTypeAddInterFrm.java 是实现图书类别添加功能的图形界面设计模块，具体界面如下图所示，其中实现了图书类别添加事件处理，重置事件处理等功能。图书类别添加事件处理实现了对输入的图书类别加入到类别表中的功能；重置事件处理实现了对已经输入的信息进行清除的功能：




图书类别添加

图书类别名称:

图书类别描述:

添加 重置

`BookTypeManageInterFrm.java` 是实现图书类别管理功能的图形界面设计模块，具体界面如下图所示，其中实现了图书类别删除事件处理、图书类别修改事件处理、表格行点击事件处理、图书类别搜索事件处理、初始化表格、重置表单等功能。图书类别删除事件处理实现了对选定图书类别的删除；图书类别修改事件处理实现了对选定图书类别的修改，注意这里图书类别的编号我们人为设置成了不可修改；表格行点击事件实现了对图书类别列表中的具体类别的选择，在点击选定类别后，在界面下方会显示该类别的信息，在这里对类别信息进行修改；初始化表格数据实现了对现有的所有图书类别在表格中显示的功能；重置表单实现了对当前输入的清除功能：



图书类别管理

图书类别名称: 查询

编号	图书类别名称	图书类别描述
----	--------	--------

表单操作

编号: 图书类别名称:

描述:

修改 删除

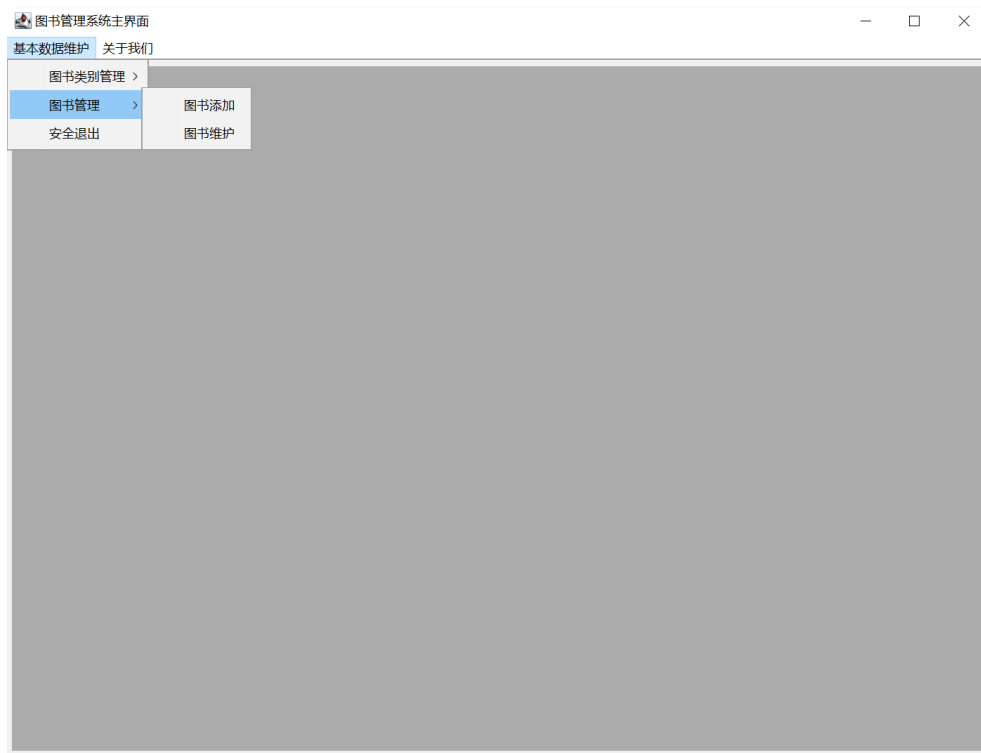
`LiQiInterFrm.java` 是实现显示关于我们的图形界面设计模块，最后的显示效果如下：



LogOnFrm.java 是实现登录界面的图形界面设计模块，具体界面如下图所示，其中实现了登录事件处理、重置事件处理等功能。登陆事件处理实现了对管理员的用户名和密码的输入的判断，如果用户名或者密码为空，则会提醒“用户名不能为空！”或“密码不能为空！”，如果都已输入但有错误，会提醒“用户名或密码错误！”；重置事件处理实现了对当前输入的清除功能：



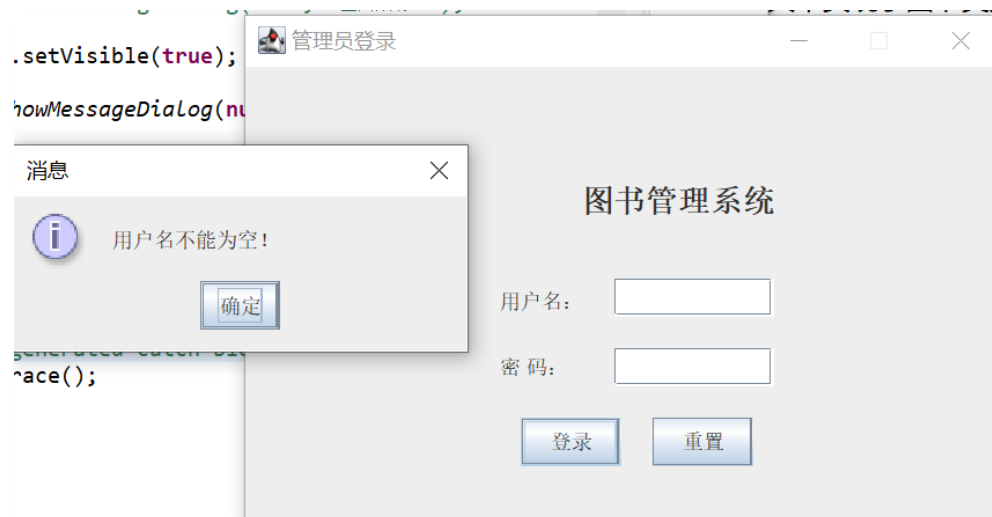
MainFrm.java 是实现图书管理系统主界面的图形界面设计模块，具体界面如下图所示，实现了对各种功能的集成，通过点击某一想要操作的功能按钮即可弹出相应界面，并在相应界面内进行操作：

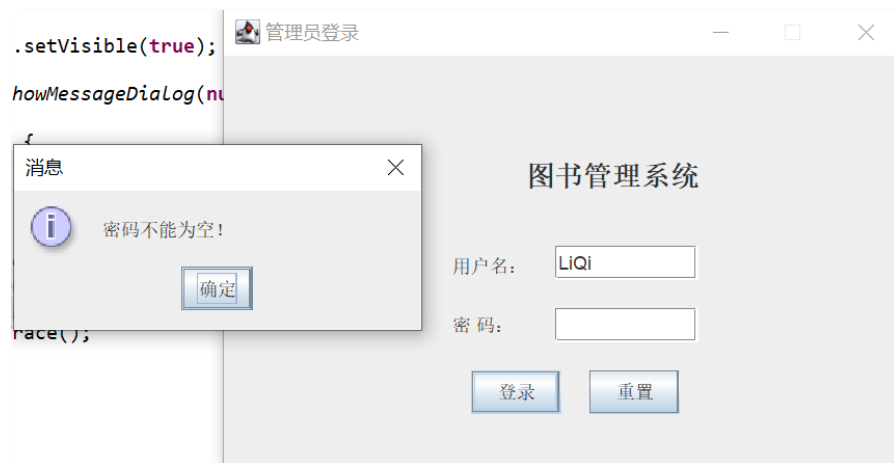


5. 功能测试

5.1. 登录功能

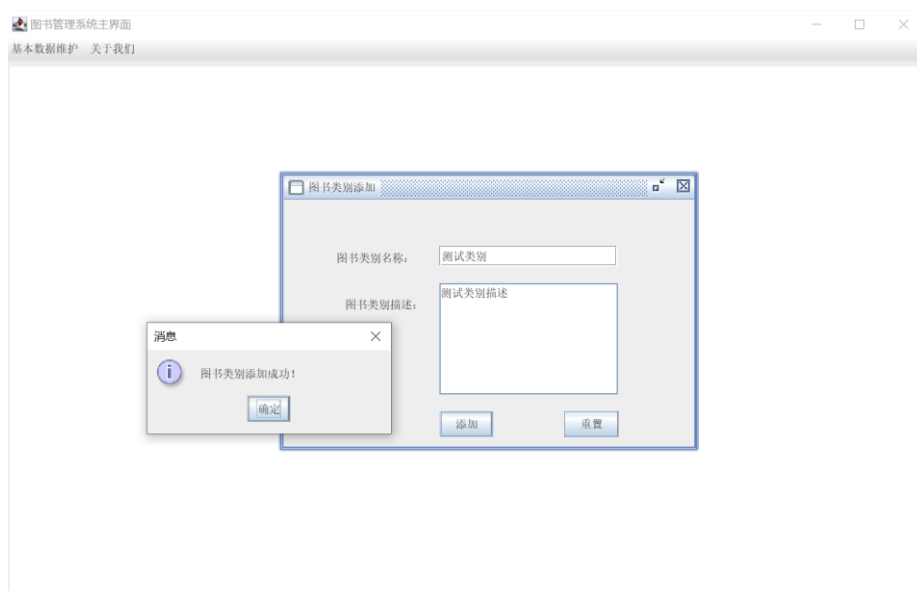
可以发现登录和重置功能都能成功使用，出现不符合要求的输入时会触发相应报错：





5.2. 图书类别添加功能

可以发现类别添加功能能够成功使用，在添加成功后，图书类别管理界面和数据库中都会添加相应的类别信息：



图书类别管理

图书类别名称:

编号	图书类别名称	图书类别描述
15	惊悚类	恐怖又刺激。
19	习题类	各教材配套习题。
20	思想启蒙类	儿童思想启蒙读物。
25	测试类别	测试类别描述

表单操作

编号: 图书类别名称:

描述:

id	bookTypeName	bookTypeDesc
5	儿童益智类	给儿童最好的礼物。
6	生物类	生物的奥秘。
7	化学类	神秘的化学世界。
8	物理类	物理让人发狂。
9	光学类	光，一个神秘的领域。
10	地理类	大千世界，地质万千。
11	政治类	险象环生的政治舞台。
12	散文类	休闲时来一本散文书。
13	科幻类	人类的想象力无穷无...
14	小说类	一个个鲜活的人物，...
15	惊悚类	恐怖又刺激。
19	习题类	各教材配套习题。
20	思想启蒙类	儿童思想启蒙读物。
25	测试类别	测试类别描述

5.3. 图书类别管理功能

以科普类的查询为例，可以看到输入”科普类”，点击查询按钮后能够成功过滤信息，只显示科普类图书的相关信息：

图书管理系统主界面

基本数据维护 关于我们

图书类别管理

图书类别名称:

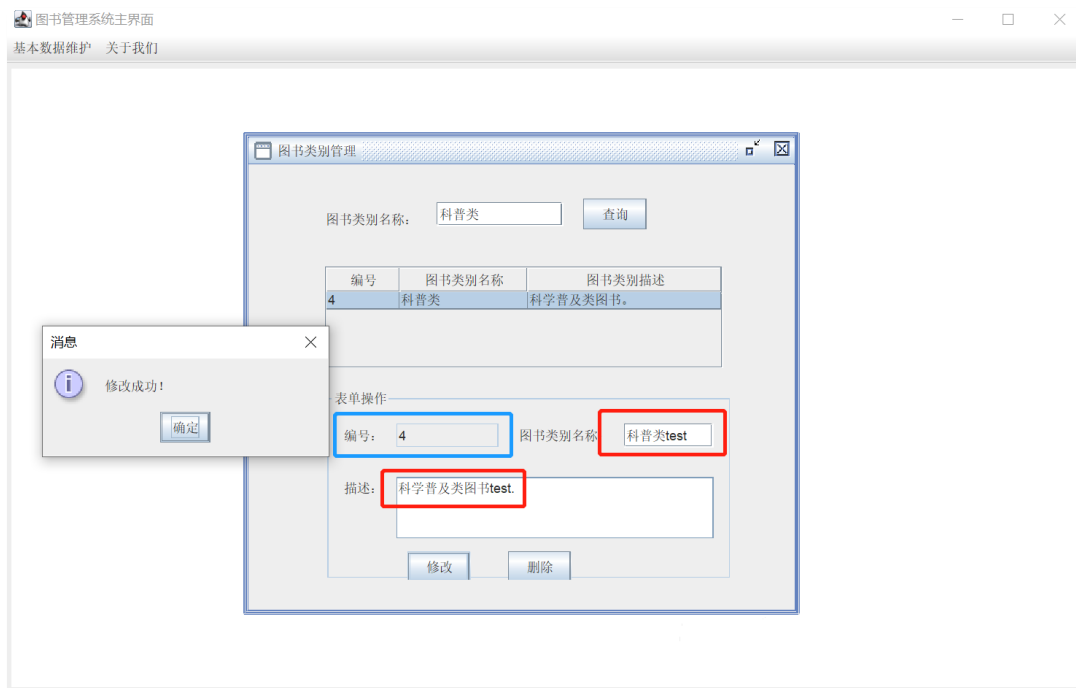
编号	图书类别名称	图书类别描述
4	科普类	科学普及类图书。

表单操作

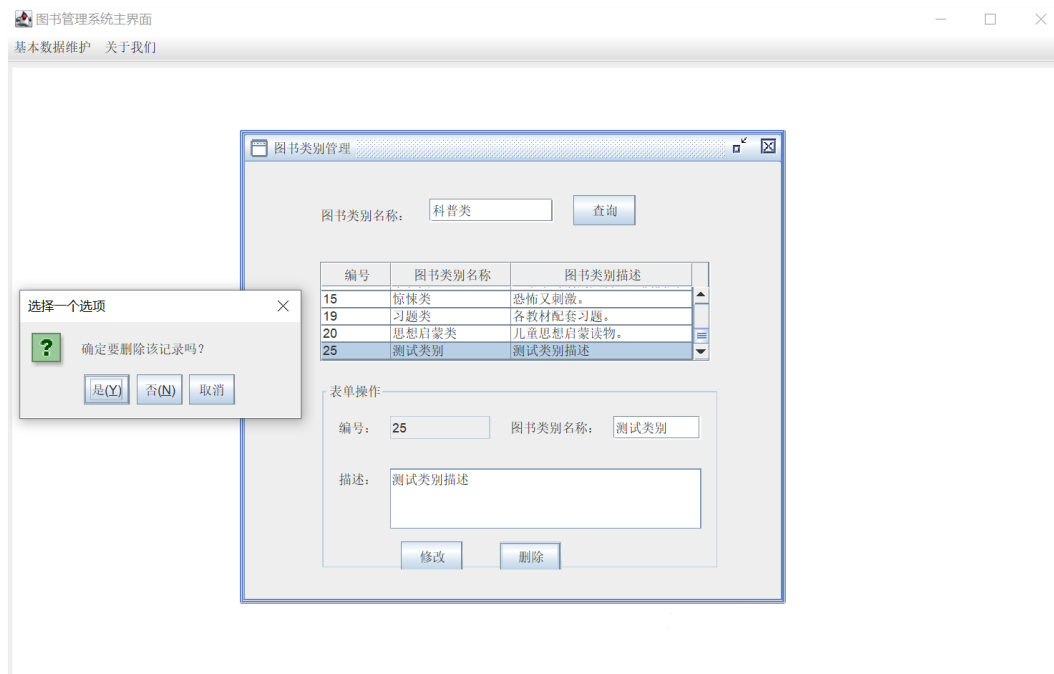
编号: 图书类别名称:

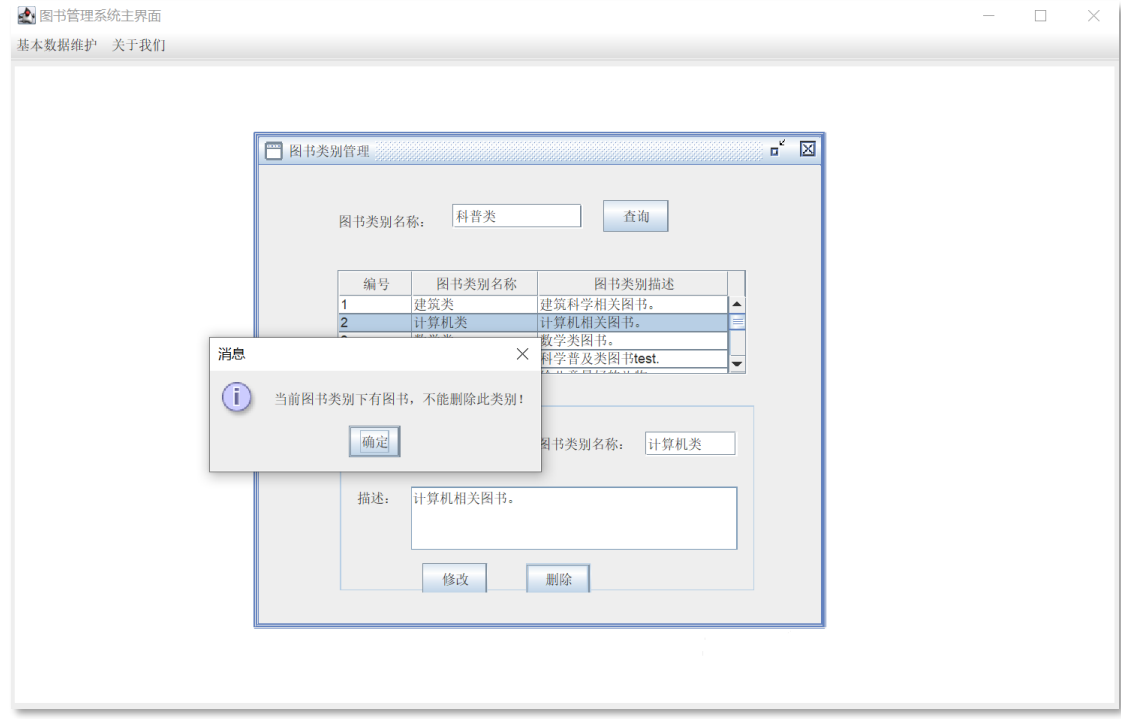
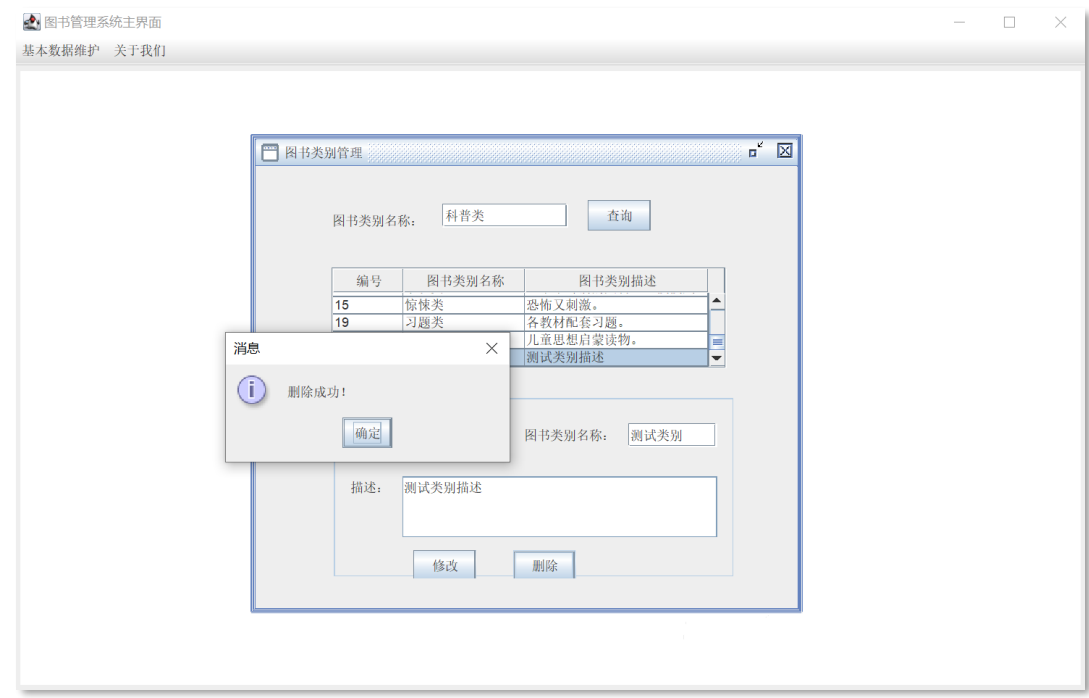
描述:

并且发现点击要修改的类别后下方能显示相应信息，编号信息不可编辑，类别名称和描述信息可以编辑，在修改信息后，点击修改按钮，弹出修改成功窗口：



点击删除按钮会显示“确定要删除该记录吗？”窗口，如果当前类别下没有图书，点击“是”，删除成功，数据库对应的表中的相应行也会删除；如果当前类别下有图书，点击“是”无法删除，会显示“当前类别下有图书，无法删除该类别！”：

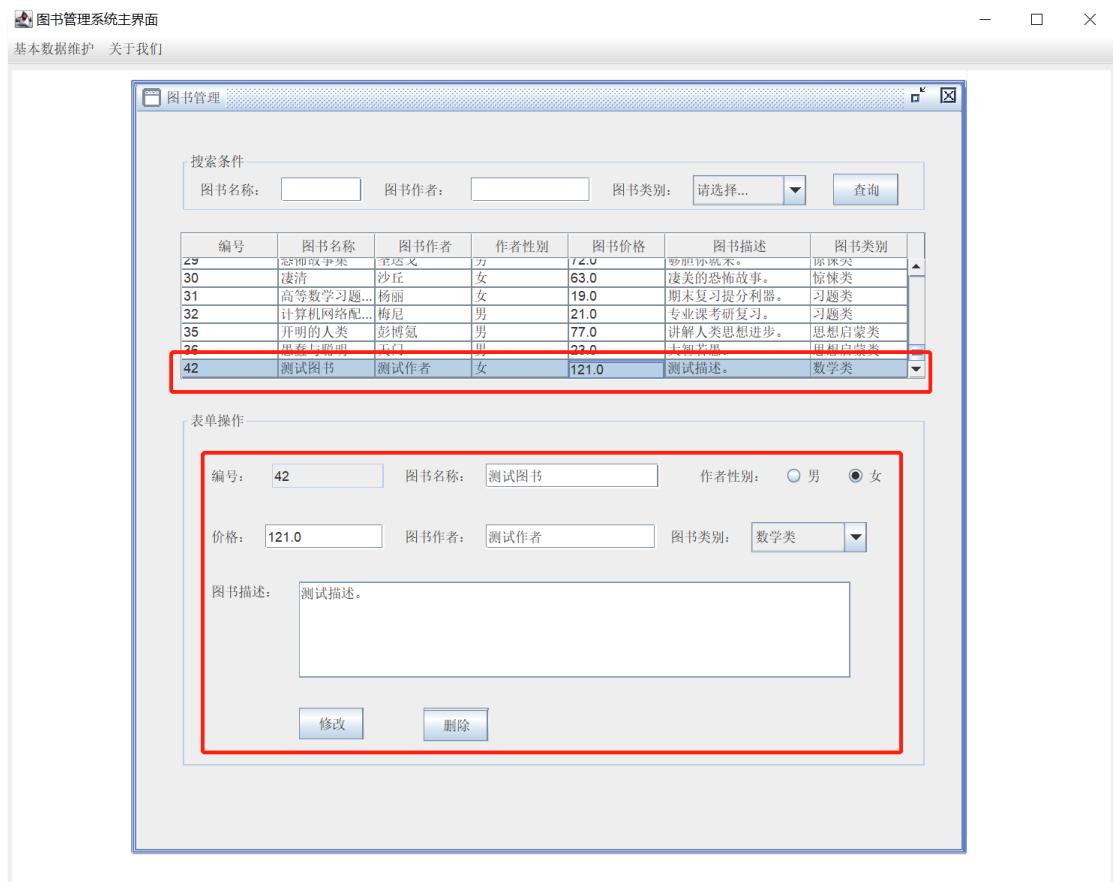
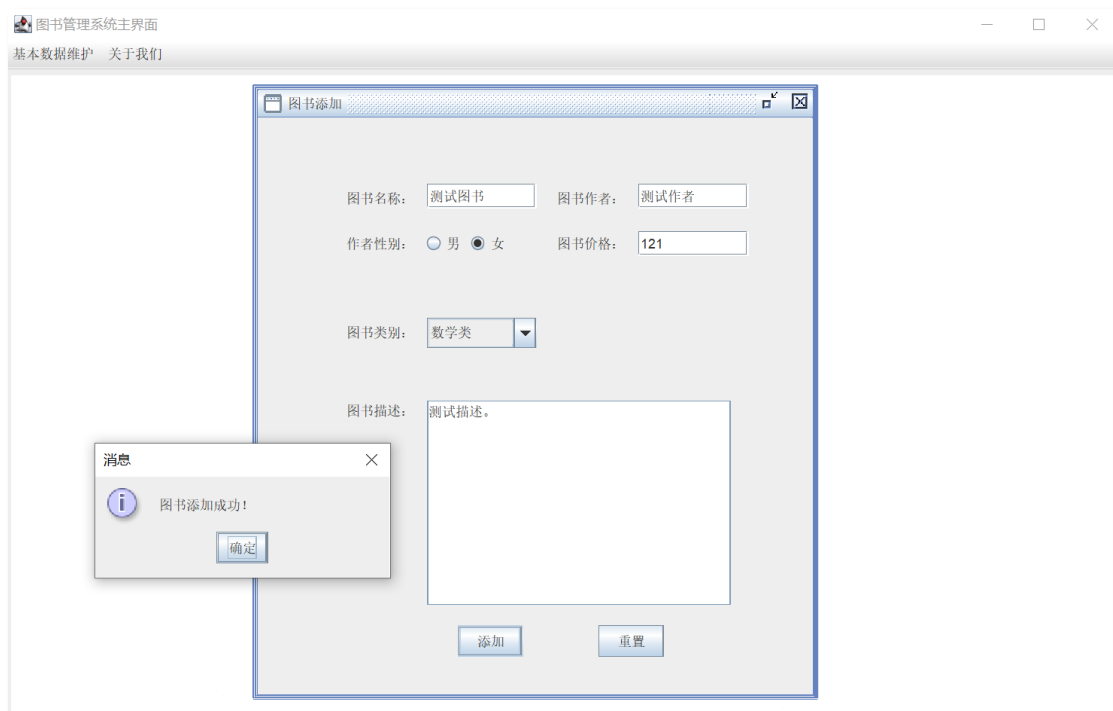




id	bookName	author	sex	price	bookTypeId	bookDesc
22	世界地貌特征	爱德华	女	123	10	世界地貌的综合讲解。
23	纸牌屋	阿岚	男	125	11	米国政坛的真实映照。
24	帝国的陨落	扎德默	男	43	11	一个没落的帝国。
25	陈忠实作品集	微积	女	345	12	陈忠实作品集。
26	精美散文	卫夫	女	54	12	近十年来的出色散文的集...
27	骆驼祥子	老舍	男	87	14	经典中的经典, 社会现实...
28	鼠疫	阿尔贝·加缪	男	45	14	老鼠的力量, 会对人类造...
29	恐怖故事集	圣达戈	男	72	15	够胆你就来。
30	凄青	沙丘	女	63	15	凄美的恐怖故事。
31	高等数学习题册	杨丽	女	19	19	期末复习提分利器。
32	计算机网络配...	梅尼	男	21	19	专业课考研复习。
35	开明的人类	彭博威	男	77	20	讲解人类思想进步。
36	墨菲与聪明	无门	男	23	20	大智若愚。
42	测试图书	测试作者	女	121	3	测试描述。

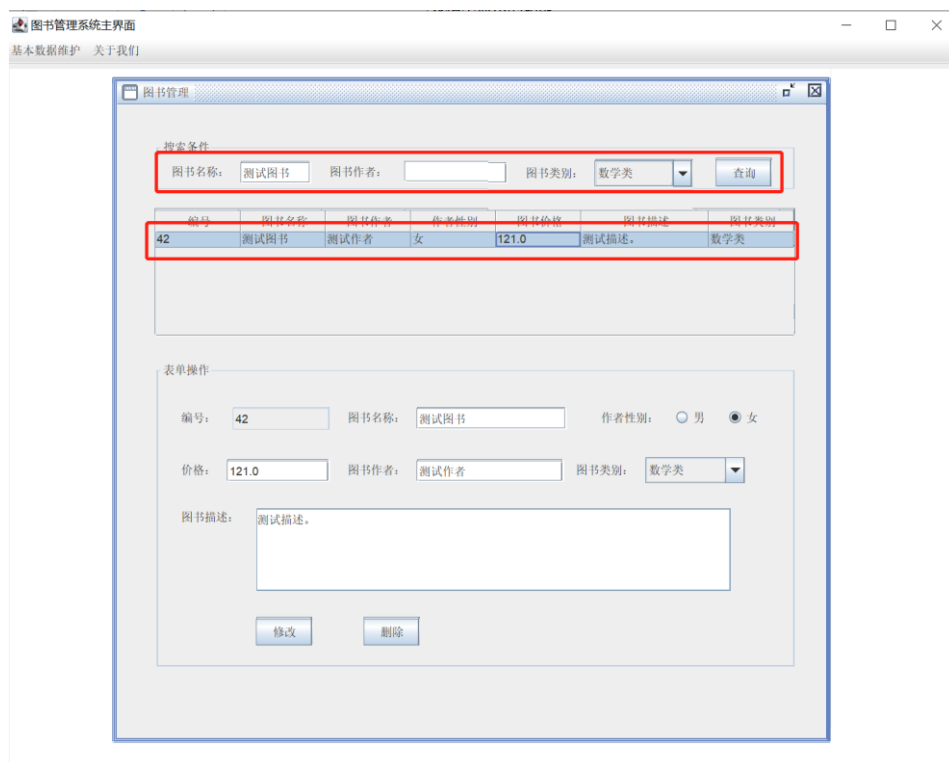
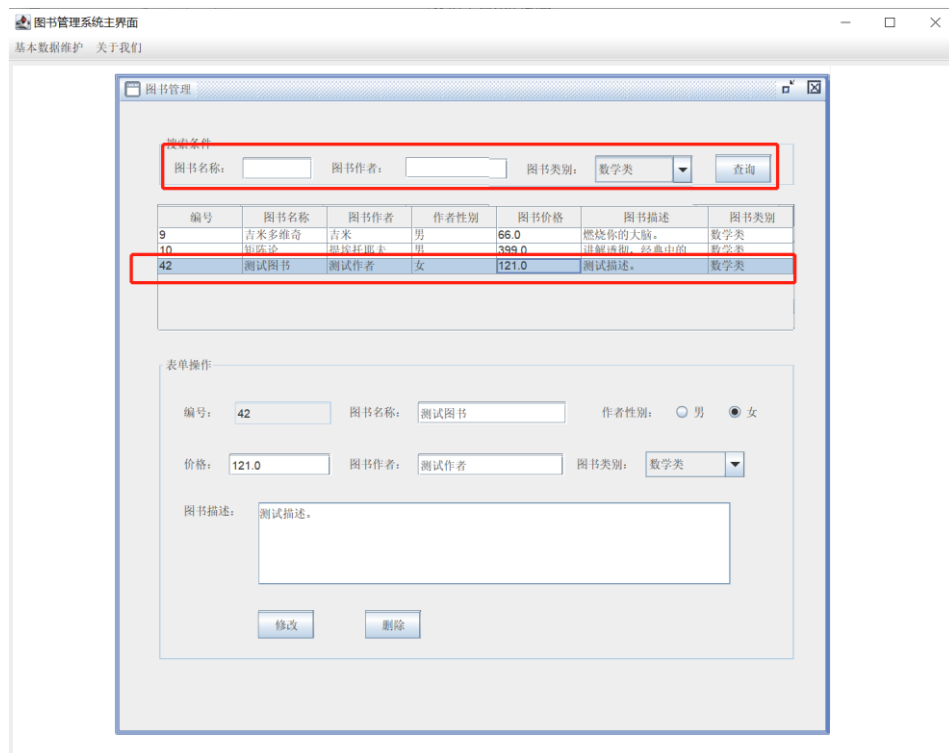
5.4. 图书添加功能

可以发现图书添加功能能够成功使用，在添加成功后，图书维护界面和数据库中都会添加相应的图书信息：

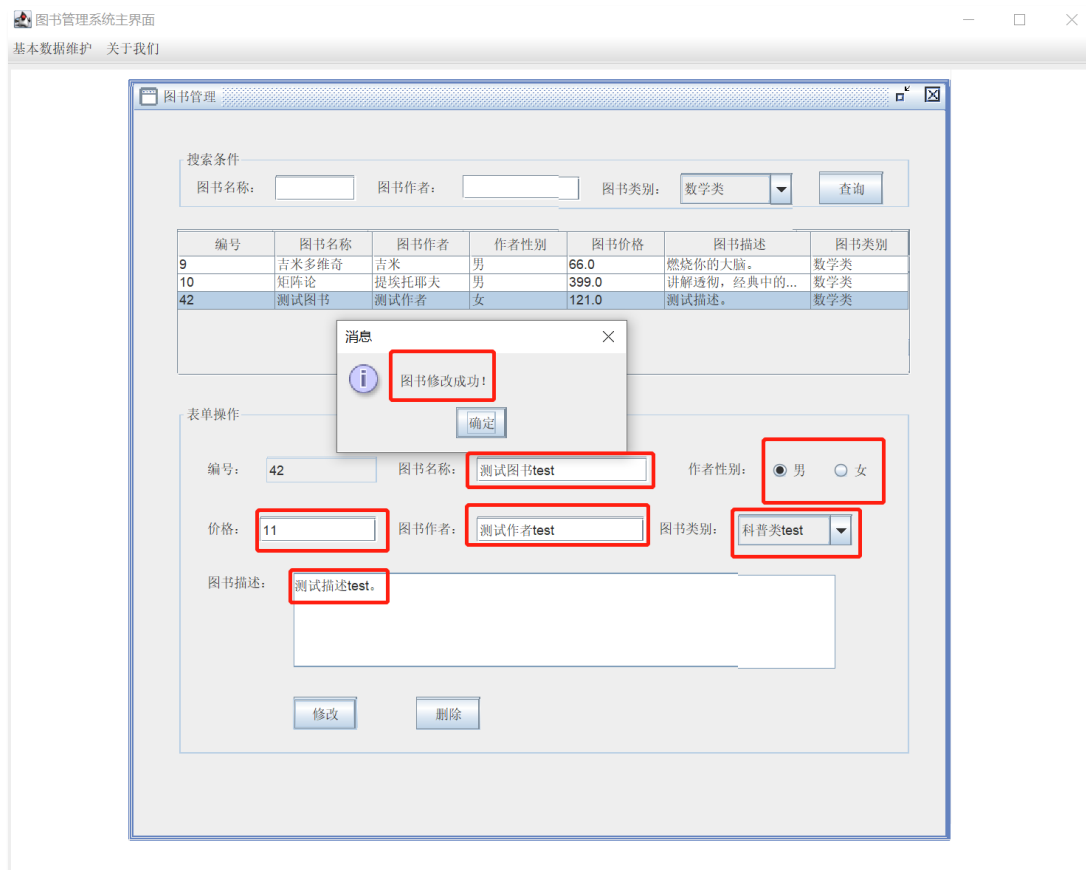


5.5. 图书管理功能

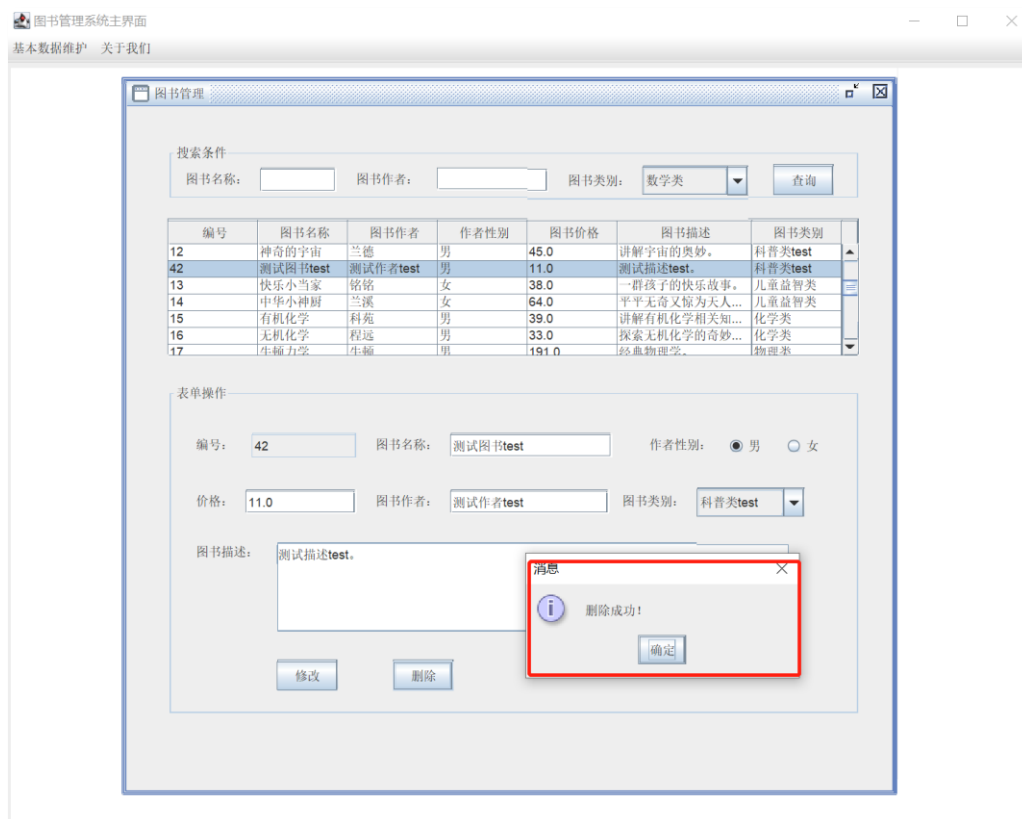
以数学类增加的测试图书为例，可以看到输入”科普类”，点击查询按钮后能够成功过滤信息，只显示科普类图书的相关信息：



并且发现点击要修改的类别后下方能显示相应信息，编号信息不可编辑，类别名称和描述信息可以编辑，在修改信息后，点击修改按钮，弹出修改成功窗口：



点击删除按钮会显示“确定要删除该记录吗？”窗口，点击“是”，删除成功，数据库对应的表中的相应行也会删除：



5.6. 退出系统功能

点击安全退出按钮，弹出“选择一个选项窗口”，点击“是”，系统成功退出。



6. 系统评价

6.1. 通过数据库技术实现主语言操作的简化

在删除某一图书类别时，需要判断该类别下是否存在图书，如果存在则不能删除，在这种情况下，手动控制耗时耗力，也增加调试的复杂度。这是就可以通过用 sql 语言创建触发器，结合修改数据库前利用主语言控制保证数据满足完整性约束，就可以保证所有表的更新都是正确的。

6.2. 各种完整性控制、触发器、事务

在设计本系统中，有关数据库的操作在写入数据库前会进行一系列的完整性约束判断，当且仅当所有条件都满足时才能向数据库发送操作请求并改善数据库。否则回在主语言中产生报错信息并拒绝操作数据库，要求用户提供符合要求的信息。这就减少了数据库出错的概率，提高了系统的可用性。

6.3. 安全性

设置了管理员账号密码，在一定程度上实现了数据库的安全性。

6.4. 高并发的一致性

由于多线程掌握不牢，无法测试高并发下能否通过锁保证一致性。

6.5. SQL 语句的优化

由于系统不存在大量数据运算，故没有进行太多的对 SQL 语句的优化，在设计上，通过先设置主体语句，后通过条件判断来在主体后面添加相应条件来简化语句。

7. 问题及解决

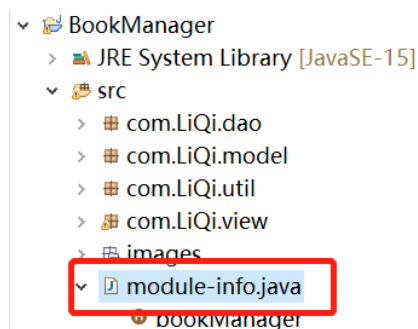
7.1. Java 连接数据库时，jdbc 链接有错误，经过在 csdn 上查询，发现是由于版本问题，数据库连接没有设置时区会出错，需要设置时区才能成功连接。

```
7  * 数据库工具类
8  *  @author 61060
9  *
10 */
11 public class DbUtil {
12     private String dbUrl="jdbc:mysql://localhost:3306/db_book?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";
13     private String dbUserName="root";
14     private String dbPassword="Liqizuihaokan1!";
15     private String jdbcName="com.mysql.cj.jdbc.Driver";
16
17 /**
18  * 获取数据库连接
19  * @return
20  * @throws Exception
21  */
22 public Connection getCon()throws Exception{
23     Class.forName(jdbcName);
24     Connection con = DriverManager.getConnection(dbUrl,dbUserName,dbPassword);
25     return con;
26 }
```

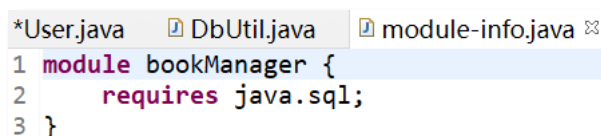
7.2. 在下载配置 Java swing 的 window builder 时，遇到如下问题：

```
1 package view;
2
3 import java.awt.BorderLayout;
4 import java.awt.EventQueue;
5
6 import javax.swing.JFrame;
7 import javax.swing.JPanel;
8 import javax.swing.border.EmptyBorder;
```

解决方案就是在自动生成的 `module-info.java` 类中添加一行代码：



```
BookManager
├── JRE System Library [JavaSE-15]
└── src
    ├── com.LiQi.dao
    ├── com.LiQi.model
    ├── com.LiQi.util
    ├── com.LiQi.view
    ├── images
    └── module-info.java
```



```
*User.java DbUtil.java module-info.java
1 module bookManager {
2     requires java.sql;
3 }
```

8. 设计心得

数据库应用系统是根据 DBMS 所支持的数据模型将数据组织起来，为用户提供数据存储、维护、检索功能。数据库设计的好坏直接影响着整个数据库系统的效率和质量。

本次数据库的课程设计，从设计目的出发，逐步里哦阿姐系统设计的任务与功能，然后对系统进行可行性分析，初步判断自己的技术、操作是否可行，并根据系统功能画出数据流图与数据字典，让我充分了解了系统中数据的流动方式，同时明白了前任设计的符号规范的简洁、高效、优美。接着进行需求分析与详细设计，再到系统实现和数据测试，每一步都逐

渐深化，提升了自己独立实践的能力与知识活学活用的能力。

在数据库设计方面，从需求分析到概念结构设计，从概念结构设计到逻辑结构设计再到物理结构设计，环环相扣，让我体验了完整的数据库设计的基本流程，也让我对相关数据库技术有了更深的理解。独立设计一个系统显著提高了自己的专业能力，开发过程中遇到的困难也让我体会到了一个好的方向比无用的努力更加重重要。

除此之外，我还学习了数据库和 JDBC 结合的相关知识，熟悉了 window builder 图形界面的使用，提升了自己的相关知识，了解了相应的代码规范，总体而言受益匪浅。

批阅者：

批阅日期：

实验成绩：

批注：