东南大學
SOUTHEAST UNIVERSITY

《数据库原理》课程

实验报告

学　号 09019204

姓　名 曹邹颖

东南大学计算机科学与工程学院

二０二１年 12 月

目录

一、场景介绍

随着智能手机的普及，虚拟图书馆这种模式能更加贴近师生的生活习惯，并且适合师生年龄段的习性，这对师生更多的接触到图书馆的信息有一定得促进作用。一套功能齐全的校内的**虚拟图书馆系统**，使其具有一般图书管理系统所具有的功能，可以实现管理员对图书信息的增删改查功能、对所有读者信息的查阅，也帮助师生读者书籍查询、书籍借阅与归还、续借等功能，方便师生搜索书籍、借阅图书。

二、数据库设计

1. 虚拟图书馆系统的数据流图：

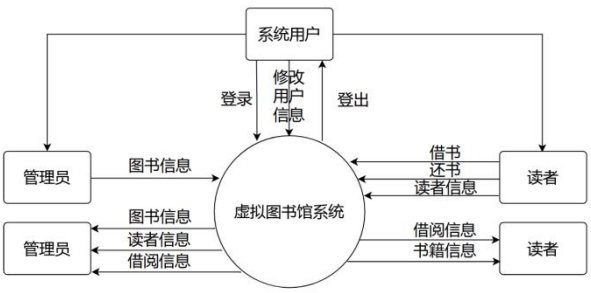


图 1.　顶层数据流图

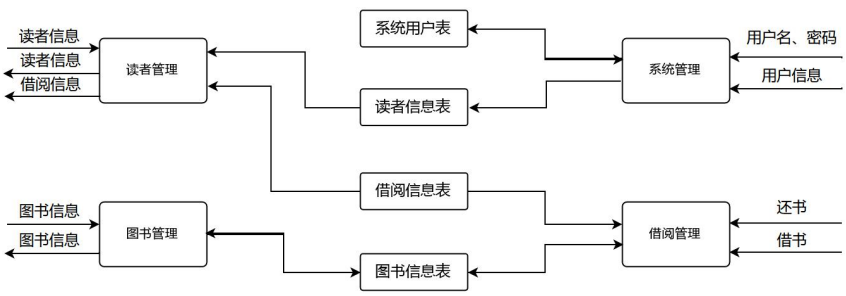


图 2.　0 层数据流图

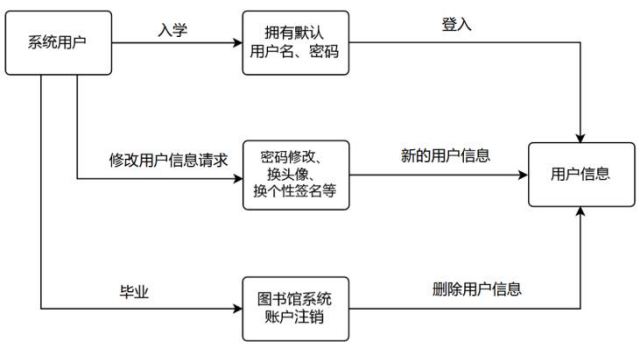


图 3.1　1 层数据流图之系统管理

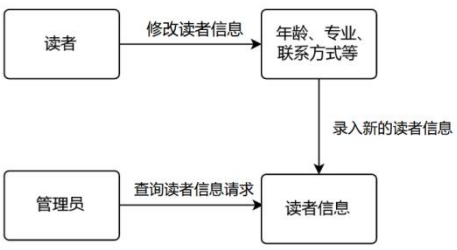


图 3.2　1 层数据流图之读者管理

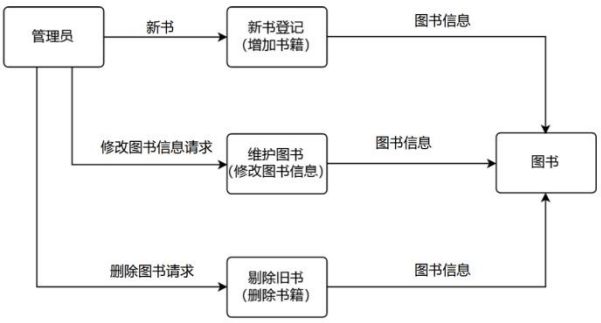


图 3.3　1 层数据流图之图书管理

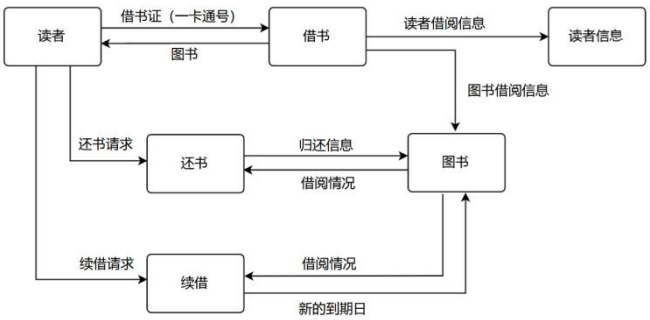
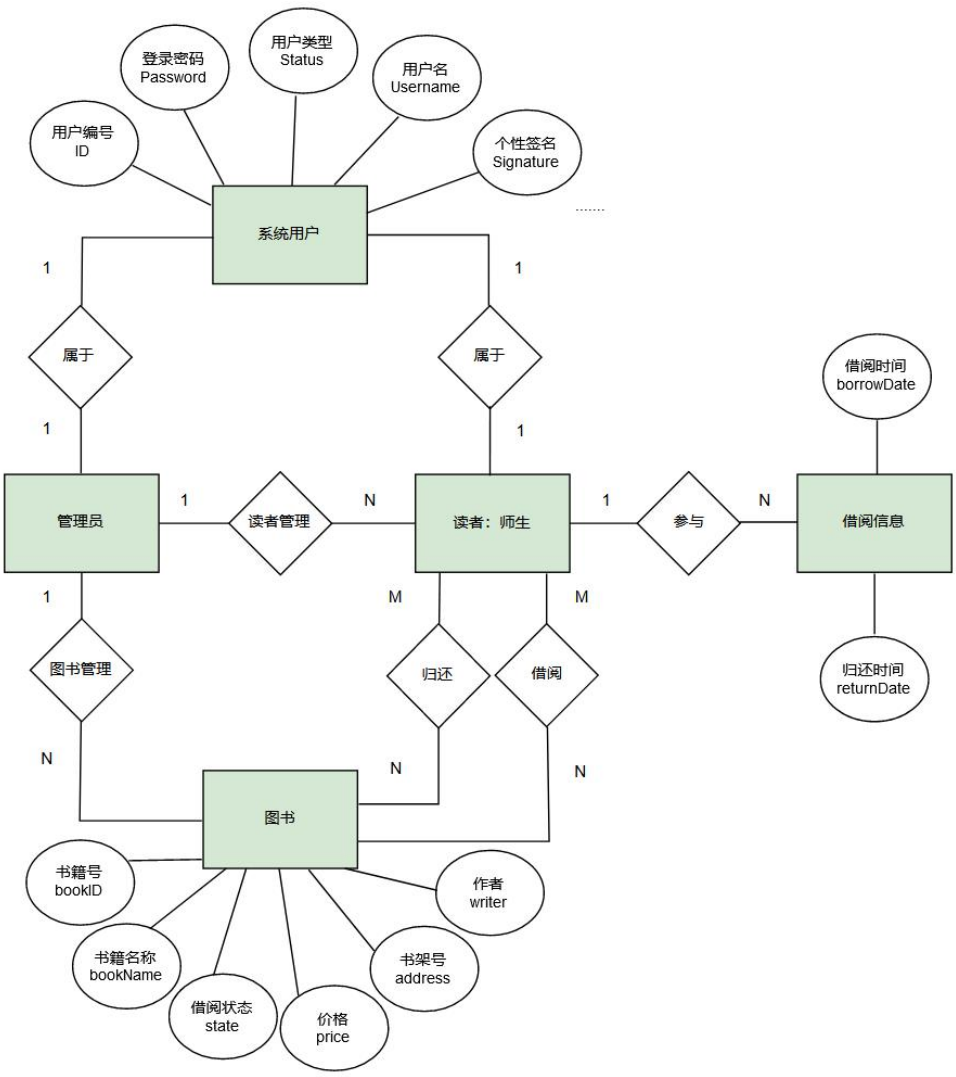


图 3.4　1 层数据流图之借阅管理

2. 虚拟图书馆系统的 E-R 图：



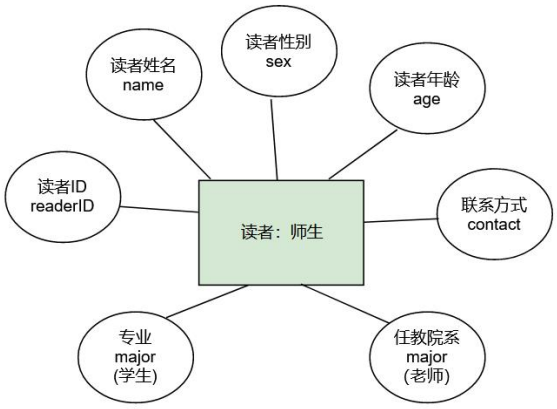


图 4. 虚拟图书馆系统 E-R 图

3. 虚拟图书馆系统的数据模式：

①建立数据库 library：

```
CREATE SCHEMA `library` DEFAULT CHARACTER SET utf8 COLLATE utf8_bin ;
```

②用户信息表 tb_user：

```
CREATE TABLE `library`.`tb_user` (
  `ID` VARCHAR(255) NOT NULL,
  `Password` VARCHAR(255) NOT NULL,
  `status` INT NULL DEFAULT 2,
  `Username` VARCHAR(45) NULL DEFAULT NULL,
  `Signature` VARCHAR(45) NULL DEFAULT NULL,
  `Selfinfo` VARCHAR(45) NULL DEFAULT NULL,
  `Background` VARCHAR(45) NULL DEFAULT NULL,
  `Headpic` BLOB NULL DEFAULT NULL,
  PRIMARY KEY (`ID`));
```

型：

	ID	Password	status	Username	Signature	Selfinfo	Background	Headpic

值：

	ID	Password	status	Username	Signature	Selfinfo	Background	Headpic
▶	000123456	guanliyuan	0	独裁者	天大地大，...	来到独裁的世...	1	BLOB

系统录入一个管理员：

```
INSERT INTO `library`.`tb_user` (`ID`, `Password`, `status`)
VALUES ('000123456', 'guanliyuan', '0');
```

③图书信息表 tb_bookinfo：

```
CREATE TABLE `library`.`tb_bookinfo` (
  `bookID` VARCHAR(45) NOT NULL,
  `bookName` VARCHAR(45) NULL DEFAULT NULL,
  `state` TINYINT NULL DEFAULT 0,
  `price` VARCHAR(45) NULL DEFAULT NULL,
  `writer` VARCHAR(45) NULL DEFAULT NULL,
  `address` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`bookID`));
```

型：

	bookID	bookName	state	price	writer	address

值：

	bookID	bookName	state	price	writer	address
	2701683	算法导论	0	CNY128.00	Thomas H.Cormen	工业技术图书阅览室(九龙湖A401)

④借阅信息表 tb_borrowinfo：

```
CREATE TABLE `library`.`tb_borrowinfo` (
```

```
  `bookID` VARCHAR(45) NOT NULL,
  `borrower` VARCHAR(45) NULL DEFAULT '000123456',
  `borrowDate` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `returnDate` TIMESTAMP NULL DEFAULT NULL,
  PRIMARY KEY (`bookID`, `borrower`, `borrowDate`, `returnDate`));
```

型：

bookID	borrower	borrowDate	returnDate

值：

2701683	213190998	2021-07-30 08:30:19	2021-07-30 08:30:30

⑤读者信息表 tb_readerinfo：

```
CREATE TABLE `library`.`tb_readerinfo` (
  `readerID` VARCHAR(255) NOT NULL,
  `sno` VARCHAR(45) NULL DEFAULT NULL,
  `name` VARCHAR(45) NULL DEFAULT NULL,
  `sex` VARCHAR(45) NULL DEFAULT '男',
  `age` INT NULL DEFAULT NULL,
  `major` VARCHAR(45) NULL DEFAULT NULL,
  `contact` VARCHAR(45) NULL DEFAULT NULL,
  PRIMARY KEY (`readerID`));
```

型：

readerID	sno	name	sex	age	major	contact

值：

213190998	09019204	czy	女	20	计科	18362155818

⑥触发器功能：

A. 对于添加用户信息后添加相应的读者信息

```
DROP TRIGGER IF EXISTS `library`.`tb_user_after_insert`;
DELIMITER $$
USE `library`$$
CREATE DEFINER = CURRENT_USER TRIGGER `library`.`tb_user_after_insert`
AFTER INSERT ON `tb_user` FOR EACH ROW
BEGIN
    INSERT INTO tb_readerinfo(readerID) VALUES (new.ID);
END$$
DELIMITER ;
```

B. 对于删除用户信息成功后删除相应的读者信息和借阅信息

```
DROP TRIGGER IF EXISTS `library`.`tb_user_AFTER_DELETE`;
DELIMITER $$
USE `library`$$
CREATE DEFINER = CURRENT_USER TRIGGER `library`.`tb_user_AFTER_DELETE`
AFTER DELETE ON `tb_user` FOR EACH ROW
BEGIN
    delete from tb_readerinfo where readerID=old.ID;
    delete from tb_borrowrinfo where readerID=old.ID;
END$$
DELIMITER ;
```

C. 对于增加借阅信息（借书）后修改 bookID 对应的图书状态 state=1(表示已借阅)

```
DROP TRIGGER IF EXISTS `library`.`tb_borrowinfo_after_insert`;
DErary`$LIMITER $$
USE `lib$
CREATE DEFINER = CURRENT_USER TRIGGER `library`.`tb_borrowinfo_after_insert`
AFTER INSERT ON `tb_borrowinfo` FOR EACH ROW
BEGIN
    update tb_bookinfo set state=1 where bookID=new.bookID;
END$$
DELIMITER ;
```

D. 对于修改借阅信息（还书）后修改 bookID 对应的图书状态 state=0(表示已归还)

```
DROP TRIGGER IF EXISTS `library`.`tb_borrowinfo_after_update`;
DELIMITER $$
USE `library`$$
CREATE DEFINER = CURRENT_USER TRIGGER `library`.`tb_borrowinfo_after_update`
AFTER UPDATE ON `tb_borrowinfo` FOR EACH ROW
BEGIN
    if not (new.returnDate <=> old.returnDate) then
        update tb_bookinfo set state=0 where bookID=new.bookID;
    end if;
END$$
DELIMITER ;
```

由于 MySQL 中不支持触发器中使用 rollback，所以这里呈现的两个触发器功能采用 sql server 创建触发器时的语言。

E. 对于删除用户信息前检查是否存在对应借阅信息（即该用户还有未归还的图书），若存在，则不予删除该用户信息（rollback）

```
create trigger tb_user_before_delete on tb_user
instead of delete
as
    if not exists(select * from tb_borrowinfo, deleted where tb_borrowinfo.borrower =
                    deleted.ID and tb_borrowinfo.returnDate>GETDATE())
        delete from tb_user where tb_user.ID in (select ID from deleted)
```

F. 对于删除图书信息前检查是否存在对应借阅信息（已借阅但未归还），若存在，则不予删除（rollback）

```
create trigger tb_bookinfo_before_delete on tb_bookinfo
after delete
as
    if exists(select * from tb_borrowinfo, deleted where tb_borrowinfo.bookID =
                deleted.bookID and tb_borrowinfo.returnDate>GETDATE())
    begin
        rollback transaction
    end
```

三、核心代码

1. 数据库连接类 DbUtil

定义一个数据库连接类 DbUtil：

DbUtil 类设计在于将 Java 中嵌入式 SQL 语言封装起来，使得调用数据库操作方便，尽管增删改都对应 execute()函数，但封装成不同函数目的在于增加代码可读性。

```
public class DbUtil {
    // 定义连接对象
    private Connection connection;
    private Statement statement;
    public DbUtil() {
        try {
            this.connection = getCon();
            this.statement = connection.createStatement();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    //获取数据库连接
    public Connection getCon() {
        Connection con = null;
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection(
                    "jdbc:mysql://localhost:3306/library?useUnicode=true&characterEncodeing=UTF-8",
                    "root",
                    "heye010612");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return con;
    }
    //关闭数据库连接
    public void closeCon(Connection con) {
        if (con != null) {
            try {
                con.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
    //添加
    public boolean add(String sql) {
        Boolean b = null;
        try {
            b = statement.execute(sql);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return b;
    }
    //删除
    public boolean delete(String sql){
        Boolean b = null;
        try {
            b = statement.execute(sql);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return b;
    }
    //更新
    public boolean update(String sql) {
        Boolean b = null;
        try {
            b = statement.execute(sql);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return b;
    }
    //查询
    public ResultSet query(String sql) {
        ResultSet rs = null;
        try {
            rs= statement.executeQuery(sql);
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return rs;
    }
}
```

2. 对用户信息表的数据库操作

按 Field 名称修改相应用户信息(密码、个性签名等)：

```
public void updateInfo(String id,String field,String newInfo) throws Exception {
    DbUtil dbUtil = new DbUtil();
    String sql="UPDATE `student`.`tb_user` SET `"+field+"` = '"+newInfo+"' WHERE (`id` = '"+id+"')";
    dbUtil.update(sql);
}
```

3. 对图书信息表的数据库操作

添加书籍信息：(用于管理员)

```
public void addBook(String bookID,String bookName,String state,String price,String writer,String address) {
    DbUtil dbUtil = new DbUtil();
    String sql = "insert into tb_bookinfo values ('" + bookID + "','" + bookName + "','" + state
            + "','" + price + "','" + writer + "','" + address + "')";
    dbUtil.add(sql);
}
```

删除书籍信息：(用于管理员)

```
public void deleteBook(String id) {
    DbUtil dbUtil = new DbUtil();
    String sql = "delete from tb_bookinfo where bookID ='" + id + "' ";
    dbUtil.delete(sql);
}
```

更新书籍信息：(用于管理员)

```
public void upDateBookInfo(String id, String name, int state, String price, String writer, String address) {
    DbUtil dbHelper = new DbUtil();
    String sql = "UPDATE `tb_bookinfo` SET `bookName` = '" + name + "',`state` = '" + state
            + "',`price` = '" + price + "',`writer` = '" + writer + "',`address` = '" + address
            +"' WHERE (`bookID` = '" + id + "')";
    dbHelper.update(sql);
}
```

查询书籍信息：

①按照 book ID 查询：(左连接∵可能没有借阅信息)(用于管理员查询图书信息)

```
public Book searchBook(String ID) throws Exception {
    DbUtil dbHelper = new DbUtil();
    String sql = "select a.bookID,bookName,state,price,writer,address,borrower,borrowDate,returnDate "
            + "from tb_bookinfo a left outer joint b_borrowinfo b on a.bookID=b.bookID and a.bookID='" + ID + "'";
    ResultSet rs = dbHelper.query(sql);
    Book book = new Book();
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"), rs.getString("borrower"),
                rs.getString("borrowDate"), rs.getString("returnDate"));
    }
    if (flag == 0)
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
    return book;
}
```

②按照 bookName 查询：(左连接∵可能没有借阅信息)

```
public ArrayList<Book> searchBookName(String Name) throws Exception {
    DbUtil dbHelper = new DbUtil();
    String sql = "select a.bookID,bookName,state,price,writer,address,borrower,borrowDate,returnDate "
            + "from tb_bookinfo a left outer join tb_borrowinfo b on a.bookID=b.bookID and a.bookName='" + Name + "'";
    ResultSet rs = dbHelper.query(sql);
    ArrayList<Book> bookList = new ArrayList<Book>();
    Book book = new Book();
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"), rs.getString("borrower"),
                rs.getString("borrowDate"), rs.getString("returnDate"));
        bookList.add(book);
    }
    if (flag == 0){
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

③按照 bookName 模糊查询：(用于读者查询图书信息)

```
public ArrayList<Book> searchBookContent(String content) throws Exception {
    DbUtil dbHelper = new DbUtil();
    String sql = "select * from tb_bookinfo where bookName like '%" + content + "%'";
    ResultSet rs = dbHelper.query(sql);
    Book book = new Book();
    ArrayList<Book> bookList = new ArrayList<Book>();
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"));
        bookList.add(book);
    }
    if (flag == 0) {
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

4. 对借阅信息表的数据库操作：

添加借阅信息：（用于借书）

```
public void addBorrowInfo(String id, String borrower, String borrowDate,
        String returnDate) {
    DbUtil dbUtil = new DbUtil();
    String sql = "insert into tb_borrowinfo values ('" + id + "' , '" + borrower
            + "', '" + borrowDate + "', '" + returnDate + "')";
    dbUtil.add(sql);
}
```

修改借阅信息：（用于续借、还书）

```
public void upDateBorrowInfo(String ID, String borrower, String borrowDate,String returnDate) {
    DbUtil dbHelper = new DbUtil();
    String sql = "UPDATE `tb_borrowinfo` SET `borrower` = '" + borrower + "',`borrowDate` = '" + borrowDate
            +"',`returnDate` = '" + returnDate+"' WHERE (`bookID` = '" + ID + "')";
    dbHelper.update(sql);
}
```

查询借阅信息：

①查询已借阅待归还的图书信息：

无特定借阅者：（用于管理员）

```
public ArrayList<Book> searchBorrowing() throws Exception {
    DbUtil dbHelper = new DbUtil();
    Book book = new Book();
    ArrayList<Book> bookList = new ArrayList<Book>();
    String sql = "select a.bookID,bookName,state,price,writer,address,borrower,borrowDate,returnDate "
            + "from tb_bookinfo a,tb_borrowinfo b where a.bookID=b.bookID and "
            + "b.returnDate>= now()";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"), rs.getString("borrower"),
                rs.getString("borrowDate"), rs.getString("returnDate"));
        bookList.add(book);
    }
    if (flag == 0){
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

特定借阅者：（通用：读者只能特指自己，管理员可以指定借阅者）

```
public ArrayList<Book> searchOneBorrowing(String ID) throws Exception {
    DbUtil dbHelper = new DbUtil();
    Book book = new Book();
    ArrayList<Book> bookList = new ArrayList<Book>();
    String sql = "select a.bookID,bookName,state,price,writer,address,borrower,borrowDate,returnDate "
            + "from tb_bookinfo a,tb_borrowinfo b where a.bookID=b.bookID and "
            + "b.returnDate>= now() and borrower='"+ID+"'";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"),ID,
                rs.getString("borrowDate"), rs.getString("returnDate"));
        bookList.add(book);
    }
    if (flag == 0){
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

②查询 dayNum 天内的所有借阅信息：

无特定借阅者：（用于管理员）

```
public ArrayList<Book> searchBorrowedInfo(int dayNum) throws Exception {
    DbUtil dbHelper = new DbUtil();
    Book book = new Book();
    ArrayList<Book> bookList = new ArrayList<Book>();
    String sql = "select a.bookID,bookName,state,price,writer,address,borrower,borrowDate,returnDate "
            + "from tb_bookinfo a,tb_borrowinfo b where a.bookID=b.bookID and "
            + "b.borrowDate>= DATE_SUB(now(), INTERVAL "+dayNum+" DAY)";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"), rs.getString("borrower"),
                rs.getString("borrowDate"), rs.getString("returnDate"));
        bookList.add(book);
    }
    if (flag == 0){
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

特定借阅者：（通用：读者只能特指自己，管理员可以指定借阅者）

```
public ArrayList<Book> searchOneBorrowedInfo(String readerID,int dayNum) throws Exception {
    DbUtil dbHelper = new DbUtil();
    Book book = new Book();
    ArrayList<Book> bookList = new ArrayList<Book>();
    String sql = "select a.bookID,bookName,state,price,writer,address,borrower,borrowDate,returnDate "
            + "from tb_bookinfo a,tb_borrowinfo b where a.bookID=b.bookID and borrower='"+readerID
            + "' and b.borrowDate>= DATE_SUB(now(), INTERVAL "+dayNum+" DAY)";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"), rs.getString("borrower"),
                rs.getString("borrowDate"), rs.getString("returnDate"));
        bookList.add(book);
    }
    if (flag == 0){
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

③查询最受欢迎的 top 10 图书信息：（即借阅次数最多的前 10 本书籍）

```
public ArrayList<Book> searchTopTenBook() throws Exception {
    DbUtil dbHelper = new DbUtil();
    Book book = new Book();
    ArrayList<Book> bookList = new ArrayList<Book>();
    String sql = "select bookName,writer,sum(borrowNum) as popularity from "
            + "(select x.bookID,bookName,writer,y.borrowNum from library.tb_bookinfo as x,"
            + "(select bookID,count(*) as borrowNum from tb_borrowinfo group by bookID) as y "
            + "where x.bookID=y.bookID)as newTable "
            + "group by bookName order by popularity desc limit 0, 10";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        book = new Book(rs.getString("bookID"), rs.getString("bookName"), rs.getString("state"),
                rs.getString("price"), rs.getString("writer"), rs.getString("address"), rs.getString("borrower"),
                rs.getString("borrowDate"), rs.getString("returnDate"));
        bookList.add(book);
    }
    if (flag == 0){
        book = new Book("NotFound", "", "", "", "", "", "", "", "");
        bookList.add(book);
    }
    return bookList;
}
```

④查询博览群书的 top 10 读者信息：（即借阅次数最多的前 10 位读者）

```
public ArrayList<Reader> searchTopTenReader() throws Exception {
    DbUtil dbHelper = new DbUtil();
    Reader reader = new Reader();
    ArrayList<Reader> readerList = new ArrayList<Reader>();
    String sql = "select * from tb_readerinfo where readerID in (select borrower from "
            + "(select borrower,count(*) as borrowNum from tb_borrowinfo group by borrower order by borrowNum limit 0, 10) "
            + "as X)";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        reader = new Reader(rs.getString("readerID"),rs.getString("sno"), rs.getString("name"),
                rs.getString("sex"), rs.getString("age"), rs.getString("major"), rs.getString("contact"));
        readerList.add(reader);
    }
    if (flag == 0){
        reader = new Reader("WithoutInformation", "", "", "", "", "", "");
        readerList.add(reader);
    }
    return readerList;
}
```

5. 对读者信息表的数据库操作：

修改读者信息：(用于读者)

```
public void updateReaderInfo(Reader reader) {
    sql = "UPDATE tb_readerinfo SET sno='"+reader.getSno()+"',"
            + "name='"+reader.getName()+"',sex='"+reader.getSex()+"',"
            + "age='"+reader.getAge()+"',major='"+reader.getMajor()+"',"
            + "contact='"+reader.getContact()+"' "
            + "WHERE readerID='"+reader.getReaderID()+"'";
    dbUtil.update(sql);
}
```

查询读者信息：

①读者查询自己信息：

```
public Reader getReaderInfo(String ID) throws Exception {
    DbUtil dbHelper = new DbUtil();
    Reader reader = new Reader();
    String sql = "select * from tb_readerinfo where readerID='"+ID+"'";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        reader = new Reader(ID,rs.getString("sno"), rs.getString("name"), rs.getString("sex"),
                rs.getString("age"), rs.getString("major"), rs.getString("contact"));
    }
    if (flag == 0)
        reader = new Reader("WithoutInformation", "", "", "", "", "", "");
    return reader;
}
```

②管理员查询某书号所有借阅者的信息：

```
public Reader getCorrespondReader(String bookID) throws Exception {
    DbUtil dbHelper = new DbUtil();
    Reader reader = new Reader();
    String sql = "select * from tb_readerinfo where readerID in "
            + "(select borrower from tb_borrowinfo where bookID='"+bookID+"')";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        reader = new Reader(rs.getString("readerID"),rs.getString("sno"), rs.getString("name"), rs.getString("sex"),
                rs.getString("age"), rs.getString("major"), rs.getString("contact"));
    }
    if (flag == 0)
        reader = new Reader("WithoutInformation", "", "", "", "", "", "");
    return reader;
}
```

③管理员查询某书号当前借阅者的信息：

```
public Reader getCurrentReader(String bookID) throws Exception {
    DbUtil dbHelper = new DbUtil();
    Reader reader = new Reader();
    String sql = "select * from tb_readerinfo where readerID="
            + "(select borrower from tb_borrowinfo where returnDate>=Now() and bookID='"+bookID+"')";
    ResultSet rs = dbHelper.query(sql);
    int flag = 0;
    while (rs.next()) {
        flag = 1;
        reader = new Reader(rs.getString("readerID"),rs.getString("sno"), rs.getString("name"), rs.getString("sex"),
                rs.getString("age"), rs.getString("major"), rs.getString("contact"));
    }
    if (flag == 0)
        reader = new Reader("WithoutInformation", "", "", "", "", "", "");
    return reader;
}
```

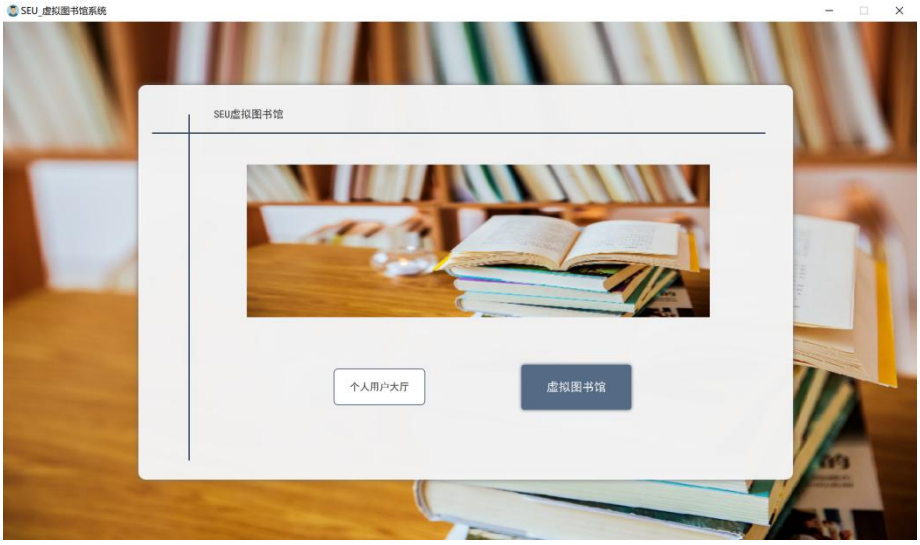
四、实验成效

1. **登录界面**（涉及数据库的用户信息表）

若密码与用户名不匹配会报错：

2. 功能界面

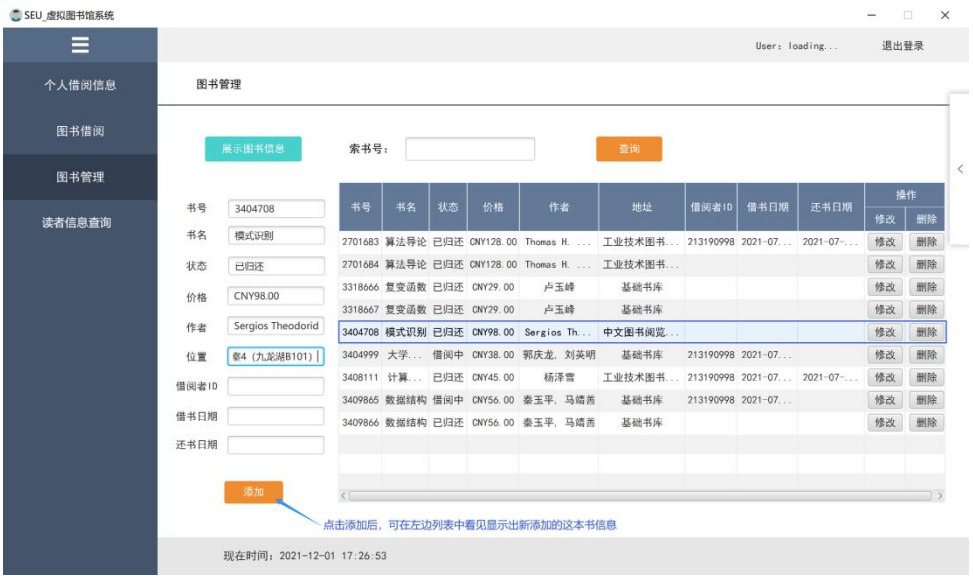
选择进入个人用户大厅还是虚拟图书馆：



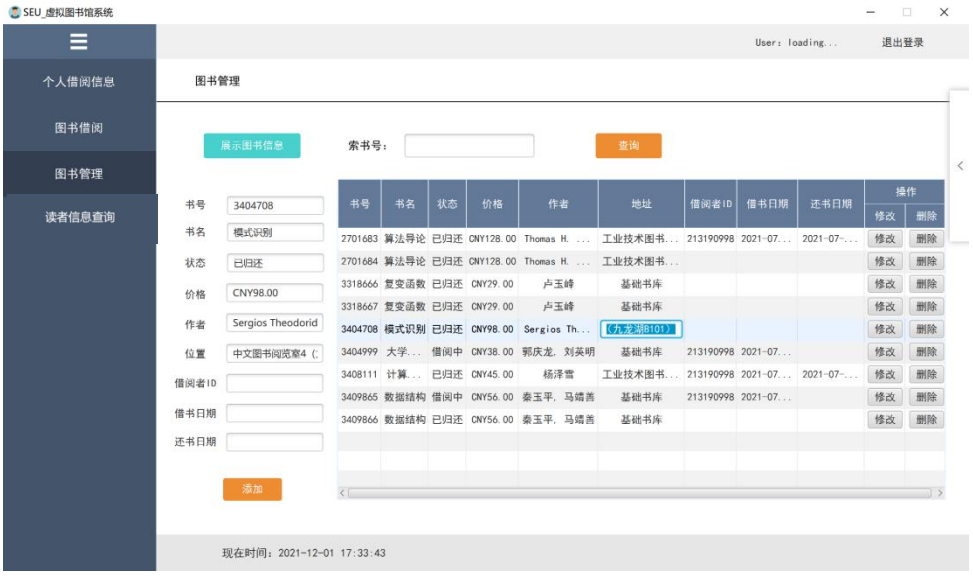
(1) 管理员特有功能：图书管理、读者信息查询

① **图书管理：(增删改图书信息)**

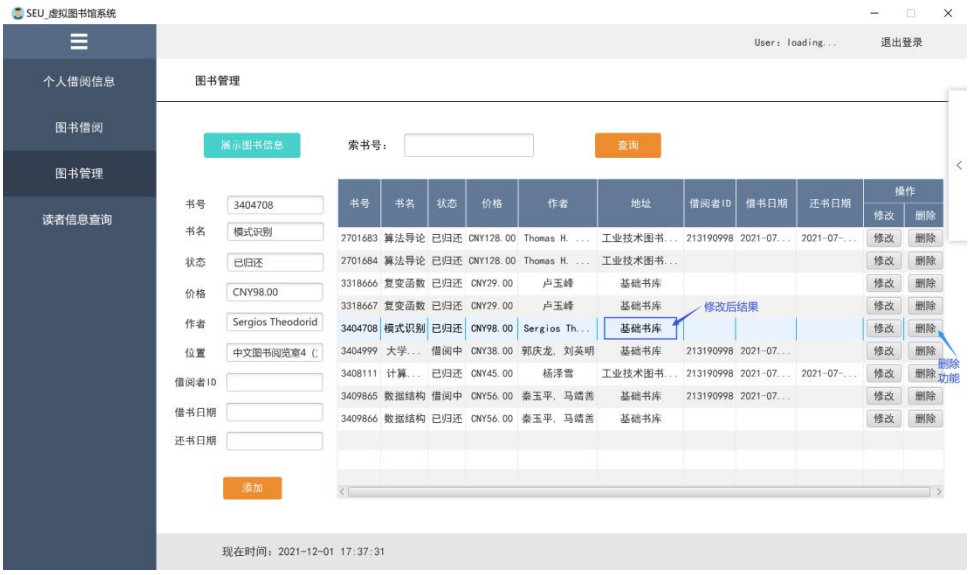
当前图书信息展示如上，使用添加图书功能如下：



点击某一条图书信息修改按钮，便可在其任意信息框中修改，
例如，修改 3404708 书号的图书位置为“基础书库”：



修改后结果如下，同时若点击删除按钮便可将这本书籍从图书馆系统中删除。



删除后结果：

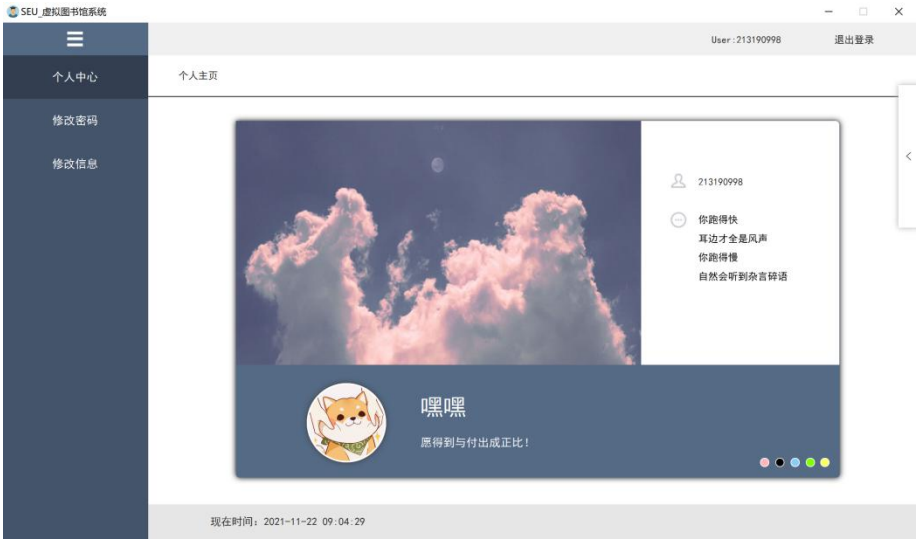
②读者信息查询：

(2) 读者特有功能：修改个人联系方式

修改结果如下：

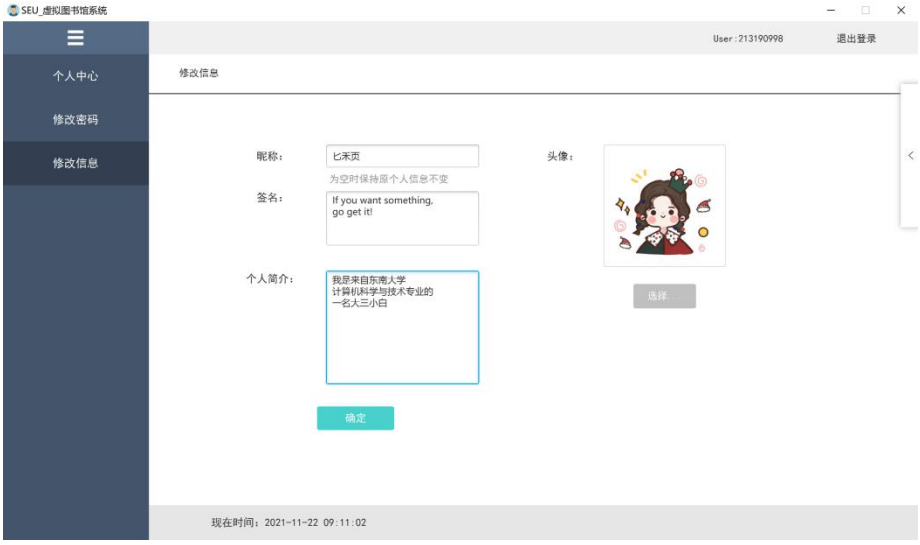
(3) 共有功能

①个人用户大厅（涉及数据库的用户信息表，根据登录的用户 ID 会查询到 TA 的用户名、系统头像、个人简介等显示）：



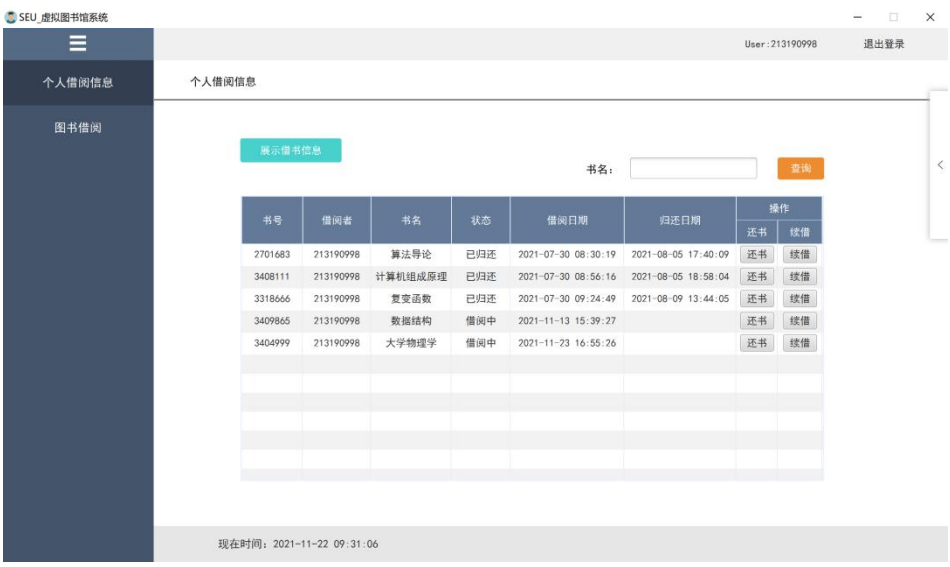
②修改密码功能：(修改完成后会需要重新登陆)

③修改用户个性信息功能：



返回个人中心便可看到个性信息的修改：

④查询借阅信息功能：

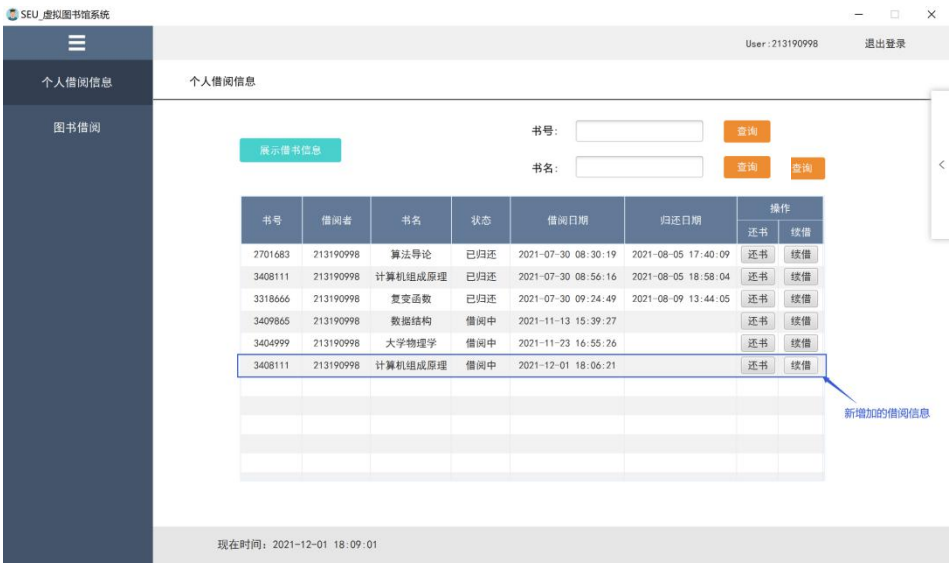


⑤还书、续借功能：

⑥书籍信息查询功能：（根据 book ID 准确查找、bookName 模糊查找）

⑦借书功能：

点击借阅后，查看个人借阅信息界面：



⑧最受欢迎的 top 10 图书信息显示、博览群书的 top 10 读者信息显示：

五、实验小结

在这次数据库课程实验中，又一次运用到 Java、MySQL 数据库、Java FX，与暑期学校的 Java 实训项目不同的是，这一次完成这个虚拟图书馆系统的设计在数据库方面增加了复杂性。

对于 MySQL 数据库，暑期学校从零开始学习时很困难，之后是靠 MySQL Workbench 可视化的界面上手的，然而这次系统学习了数据库原理课程后，就是直接使用 MySQL 中的 SQL 语句去构建数据库、并在工作区输入 SQL 语句对相应表增删改查进行测试，便于自己 SQL 语句书写的纠错。UI 设计关于 FXML 文件和 CSS 的内容真是复杂纷呈，我在有限时间内做出了一些 UI 的修改与美化，还算是较为满意的。

总体下来，自己在这个实验中得到的锻炼主要便是复杂数据库语言的实践，在数据库测试时经常会报错，虽然修改错误的过程并不难，但是要在其中吸取教训，才是有意义的，不管怎样，每次出现问题都是对所学知识加深认识的机会，都是进步的过程！