Database Systems,CCST,Zhejiang University
Laboratory Projects

Library Management System

Author: Classified
Date: 2023-04-01~04-10 Instructed by
Prof. ZHOUBO

一、实验任务

1. 设计简单的图书管理数据库概念模式。
2. 设计相应的关系模式。
3. 实现一个图书管理程序，实现图书、借书证及图书借阅的管理的基本功能。
4. 设计简单的图书管理数据库概念模式。

借此掌握简单数据库应用程序的设计开发方法。

二、实验环境

- 操作系统: macOS Ventura 13.3.1
- 数据库管理系统：MySQL
- 编程语言：Java

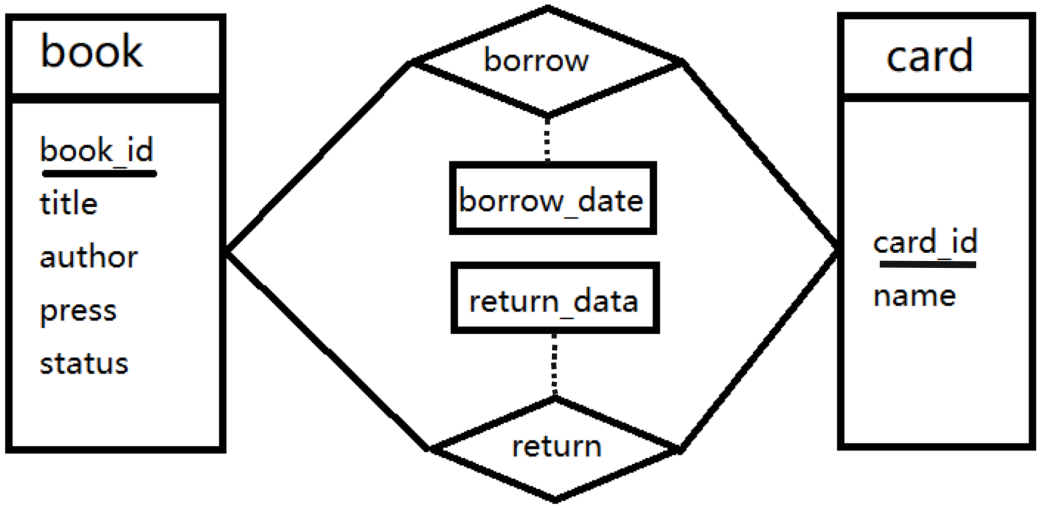
三、实验内容与流程记录

3.0 需求分析

设计的系统需要具备下面的功能：

- 书籍管理：新增馆藏书籍和删除书籍数据
- 书籍查询：根据书名检索书籍，查询是否未被借阅；展示所有藏书
- 借还书：提供借书卡的卡号和要借阅书号，完成借阅和归还的事务
- 借阅卡管理：新增注册借书卡，注销原有借书卡
- 借阅卡查询：展示系统里所有的借阅卡信息

3.1 设计简单的图书管理数据库概念模式(以E-R图的形式呈现)



3.2 设计相应的关系模式。

一个简单的图书管理数据库可能需要以下几个概念模型：

- Book（书籍）： 包含书籍的基本信息，如书号id、书名、作者、借阅状态等。
- Card（借书人）： 包含借书人的基本信息，如借书证ID、姓名等。
- BorrowRecord（借书记录）： 包含借书记录的基本信息，如借书卡ID、书籍、借书日期、归还日期等。

根据分析，可以定义以下关系模式：

- book（book_id, title, author, press ）： 其中book_id为书籍的唯一标识。
- card（card_id, name）： 其中card_id为借书证的唯一标识，borrower_id和library_id分别为借书证的持有者和发行图书馆的唯一标识。
- borrowRecord（record_id, card_id, book_id, borrow_date, return_date）： 其中record_id为借书记录的唯一标识，card_id和book_id分别为借书记录中的借书证和书籍唯一标识。

3.3 实现一个图书管理程序，实现图书、借书证及图书借阅的管理的基本功能。

首先，安装JDK和JDBC for MySQL等环境，运行简单JDBC检测是否环境配置是否正常。

连接数据库，创建相应的表格，插入数据：使用JDBC来实现。

以下是程序的基本流程：

1. 连接到MySQL数据库
2. 创建book、card和borrowRecord三张表
3. 向book表中插入20本书籍的记录
4. 向card表中插入5张借书证的记录
5. 提供基本操作菜单：新增馆藏书籍、删除书籍、根据书名检索书籍、借阅书籍、归还书籍、新增借书卡、注销借书卡、退出程序
6. 根据用户选择，执行相应的操作

> 注意，当检测到目标数据库中已有book/card/borrowRecord表时，程序将不会初始化这些表，而是继承上次运行该管理程序遗留的数据。

实现一个基本的图书管理程序需要以下步骤：

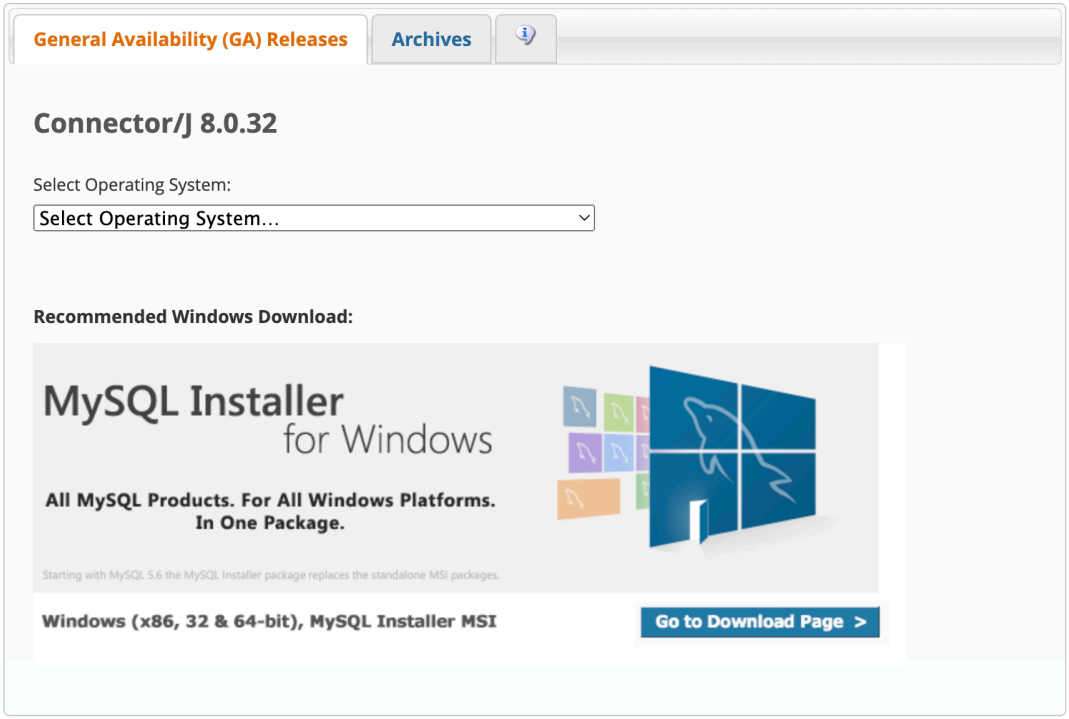
1. 安装MySQL数据库，根据开发者文档配置JDBC环境。
2. 使用JDBC连接到MySQL数据库。
3. 创建图书、借书证和借书记录的数据表。
4. 实现添加、查询和删除图书、借书证和借书记录的功能。
5. 实现借阅和归还图书的功能。

下面是详细的实验过程记录

首先，安装MySQL 以及配置 JDBC环境。详细参考这个网址

MySQL Community Downloads

Connector/J



创建workspace文件夹，编写java程序。import需要的库后，凭借下面的语句连接到数据库

```
    private static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";//JDBC驱动名
    private static final String DB_URL = "jdbc:mysql://localhost:3306/library"; //数据库
连接地址
    //数据库用户名和密码
    private static final String USER = "root";
    private static final String PASSWORD = "*******";
```

分析判断数据库是否已经存在要创建的表，若有，则不额外做操作直接连接；若无，则创建并初始化（添加随机数据记录到book和card）

```
//psuedocode
for each tableName in tableNames do
    executeQuery SQL statement: SELECT * FROM tableName
    if query fails then
        createTable()
    end if
end for
```

> 创建初始化表的JDBC代码集成在createTable()函数中实现

下面是这个CLI程序的交互设计：

- 程序开始显示prompts

```
1. 新增馆藏书籍
2. 删除书籍
3. 根据书名检索书籍
4. 借阅书籍
5. 归还书籍
6. 新增借书卡
7. 注销借书卡
8. 显示所有馆藏书籍
9. 显示所有借阅卡信息
0. 退出程序
```

- 用户选择操作项，hit enter并根据提示进一步补充操作所需的信息
- 系统执行所指定的任务，返回用户需求的内容，并显示操作成或者失败的prompt

> 注意：各个职能都封装在对应的函数中，具有良好的封装特性

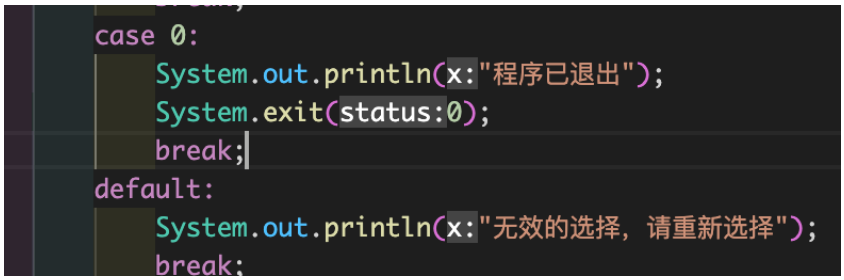
- 每一个函数都内嵌了执行指定功能的sql指令，通常是接受用户的补充输入来complete整条sql语句

  例如下面新增借阅卡的函数实现：

```
// 新增借书卡
private static void addCard() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入借书人姓名：");
```

```
    String name = scanner.nextLine();
    String sql = "INSERT INTO card (name) VALUES (" +
            "'" + name + "'" +
            ")";
    try {
        stmt.executeUpdate(sql);
        System.out.println("新增借书卡成功");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

Prompts会持续出现直至用户输入`0`

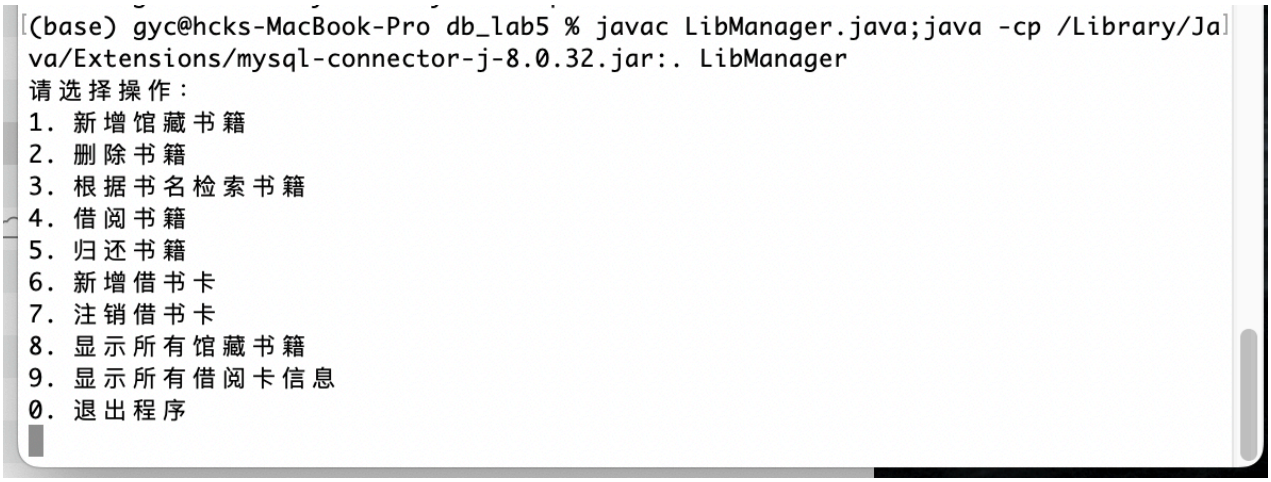


程序中还含有许多try catch的异常处理程式，具体不在报告中展开。

四 实验成果展示

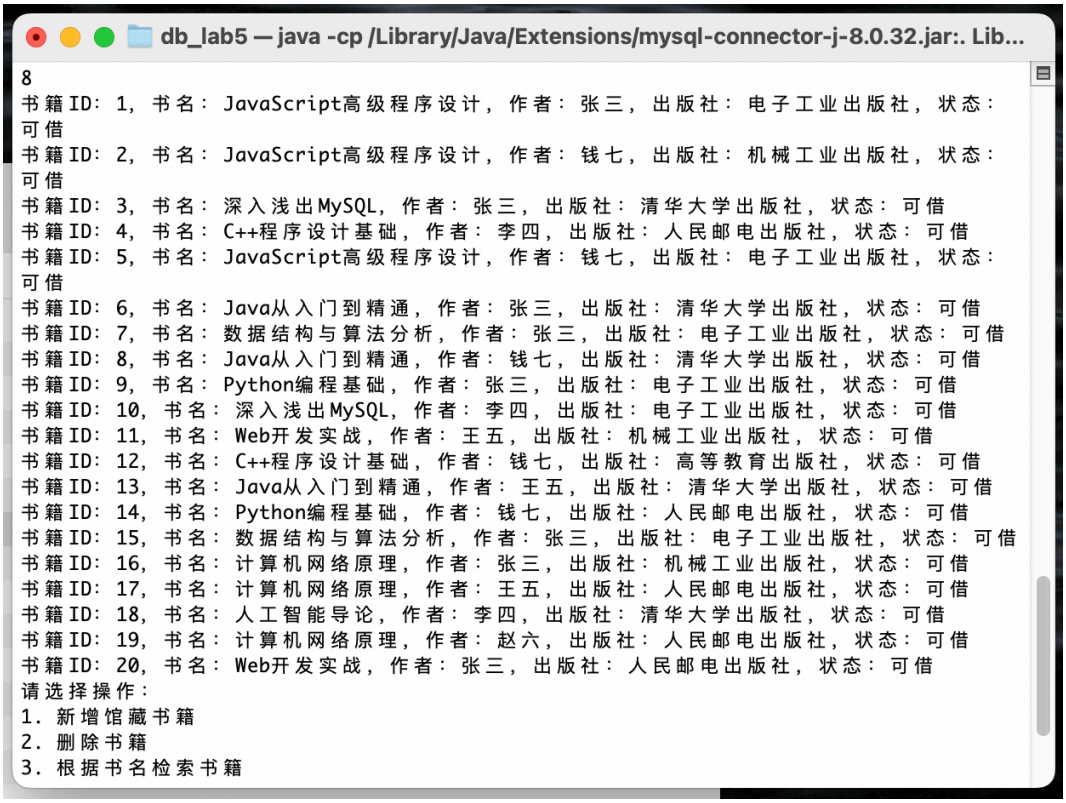
首先cd到workspace工作目录，在使用下面的指令来运行程序：

```
javac LibManager.java;java -cp /Library/Java/Extensions/mysql-connector-j-8.0.32.jar:. LibManager
```



可以看到初始的prompt已经出现。下面将逐个测试项目设计伊始的需求功能。

1. 浏览馆藏图书列表(包含借阅情况)



所有第一次连接随机生成的书目都正常展示，包含借阅情况。

2. 显示登记在案的借阅卡信息

```
9
借阅卡ID: 1, 姓名: 张三
借阅卡ID: 2, 姓名: 王五
借阅卡ID: 3, 姓名: 张三
借阅卡ID: 4, 姓名: 张三
借阅卡ID: 5, 姓名: 张三
请选择操作:
```

3. 新增馆藏书籍

```
1
请输入书名:
生成式AI入门
请输入作者:
胡宸恺
请输入出版社:
浙江大学出版社
新增书籍成功
```

检验：

```
书籍ID: 17, 书名: 计算机网络原理, 作者: 王五, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 18, 书名: 人工智能导论, 作者: 李四, 出版社: 清华大学出版社, 状态: 可借
书籍ID: 19, 书名: 计算机网络原理, 作者: 赵六, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 20, 书名: Web开发实战, 作者: 张三, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 21, 书名: 生成式AI入门, 作者: 胡宸恺, 出版社: 浙江大学出版社, 状态: 可借
请选择操作:
1. 新增馆藏书籍
```

4. 移除馆藏书籍（丢失或者永久性迁移）

```
0. 退出程序
2
请输入要删除的书籍ID:
21
删除书籍成功
请选择操作:
```

检验：21号藏书已经被移除

```
书籍ID: 13, 书名: Java从入门到精通, 作者: 王五, 出版社: 清华大学出版社, 状态: 可借
书籍ID: 14, 书名: Python编程基础, 作者: 钱七, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 15, 书名: 数据结构与算法分析, 作者: 张三, 出版社: 电子工业出版社, 状态: 可借
书籍ID: 16, 书名: 计算机网络原理, 作者: 张三, 出版社: 机械工业出版社, 状态: 可借
书籍ID: 17, 书名: 计算机网络原理, 作者: 王五, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 18, 书名: 人工智能导论, 作者: 李四, 出版社: 清华大学出版社, 状态: 可借
书籍ID: 19, 书名: 计算机网络原理, 作者: 赵六, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 20, 书名: Web开发实战, 作者: 张三, 出版社: 人民邮电出版社, 状态: 可借
请选择操作:
1. 新增馆藏书籍
2. 删除书籍
3. 根据书名检索书籍
4. 借阅书籍
```

5. 借阅书籍

以三号借阅者为例，可以看到借阅时间和默认的最晚归还时间

```
4
请输入借书证号:
3
请输入书籍ID:
19
借阅成功, 借阅日期: 2023-04-16, 归还日期: 2023-05-16
请选择操作:
```

检验

```
书籍ID: 15, 书名: 数据结构与算法分析, 作者: 张三, 出版社: 电子工业出版社, 状态: 可借
书籍ID: 16, 书名: 计算机网络原理, 作者: 张三, 出版社: 机械工业出版社, 状态: 可借
书籍ID: 17, 书名: 计算机网络原理, 作者: 王五, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 18, 书名: 人工智能导论, 作者: 李四, 出版社: 清华大学出版社, 状态: 可借
书籍ID: 19, 书名: 计算机网络原理, 作者: 赵六, 出版社: 人民邮电出版社, 状态: 已借出
书籍ID: 20, 书名: Web开发实战, 作者: 张三, 出版社: 人民邮电出版社, 状态: 可借
请选择操作:
1. 新增馆藏书籍
2. 删除书籍
```

6. 归还书籍

```
5
请输入借书证号:
3
请输入书籍ID:
19
归还成功, 借阅日期: 2023-04-16, 应还日期: 2023-05-16, 实际还书日期: 2023-04-16
请选择操作:
```

检验

```
书籍ID: 12, 书名: C++程序设计基础, 作者: 钱七, 出版社: 高等教育出版社, 状态: 可借
书籍ID: 13, 书名: Java从入门到精通, 作者: 王五, 出版社: 清华大学出版社, 状态: 可借
书籍ID: 14, 书名: Python编程基础, 作者: 钱七, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 15, 书名: 数据结构与算法分析, 作者: 张三, 出版社: 电子工业出版社, 状态: 可借
书籍ID: 16, 书名: 计算机网络原理, 作者: 张三, 出版社: 机械工业出版社, 状态: 可借
书籍ID: 17, 书名: 计算机网络原理, 作者: 王五, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 18, 书名: 人工智能导论, 作者: 李四, 出版社: 清华大学出版社, 状态: 可借
书籍ID: 19, 书名: 计算机网络原理, 作者: 赵六, 出版社: 人民邮电出版社, 状态: 可借
书籍ID: 20, 书名: Web开发实战, 作者: 张三, 出版社: 人民邮电出版社, 状态: 可借
请选择操作:
1. 新增馆藏书籍
2. 删除书籍
```

借阅状态已经恢复。

7. 新增借阅卡

```
6
请输入借书人姓名:
胡宸恺
新增借书卡成功
请选择操作:
```

检验

```
借阅卡ID: 1, 姓名: 张三
借阅卡ID: 2, 姓名: 王五
借阅卡ID: 3, 姓名: 张三
借阅卡ID: 4, 姓名: 张三
借阅卡ID: 5, 姓名: 张三
借阅卡ID: 6, 姓名: 胡宸恺
请选择操作:
1. 新增馆藏书籍
```

8. 注销借阅卡

```
7
请输入借书卡ID:
6
注销借书卡成功
请选择操作:
```

检验

```
借阅卡ID: 1, 姓名: 张三
借阅卡ID: 2, 姓名: 王五
借阅卡ID: 3, 姓名: 张三
借阅卡ID: 4, 姓名: 张三
借阅卡ID: 5, 姓名: 张三
请选择操作:
1. 新增馆藏书籍
```

新增记录已被注销。

9. 退出程序0

```
8. 显示所有馆藏书籍
9. 显示所有借阅卡信息
0. 退出程序
0
程序已退出
(base) gyc@hcks-MacBook-Pro db_lab5 %
```

成功退出程序。

> 若要测试编译运行本项目，请参考submission中的readme文本文档的指导

Appendix: Soure Code

#LibManager.java

```
import java.sql.*;//导入java.sql包中的所有类
import java.util.Scanner;//导入java.util包中的Scanner类,用于获取用户输入

public class LibManager {

    private static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";//JDBC驱动名
    private static final String DB_URL = "jdbc:mysql://localhost:3306/library"; //数据库连接地址

    //数据库用户名和密码
    private static final String USER = "root";
    private static final String PASSWORD = "030531";

    //数据库连接对象，用来连接database
    private static Connection conn = null;
    private static Statement stmt = null;
    private static ResultSet rs = null;

    public static void main(String[] args) {

        // 连接数据库
        try {
            Class.forName(JDBC_DRIVER);
            //连接数据库
            conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);
            stmt = conn.createStatement();
        } catch (ClassNotFoundException | SQLException e) {//捕获ClassNotFoundException和SQLException异常
            e.printStackTrace();
        }

        String[] tableNames = {"book", "card", "borrowRecord"};

        // // debug
        // // 检查是否已经存在book、card和borrowRecord三张表，如果存在则删除
        // for (String tableName : tableNames) {
        //     String sql = "DROP TABLE IF EXISTS " + tableName;
        //     try {
```

```
//             stmt.executeUpdate(sql);
//         } catch (SQLException e) {
//             e.printStackTrace();
//         }
// }

//如果数据库不存在book、card和borrowRecord三张表，则创建它们并初始化
for (String tableName : tableNames) {
    String sql = "SELECT * FROM " + tableName;
    try {
        rs = stmt.executeQuery(sql);
    } catch (SQLException e) {
        // e.printStackTrace();
        // 如果查询失败，则说明表不存在，需要创建
        createTable();
    }
}

//如果数据库已经存在book、card和borrowRecord三张表，则不做任何操作

// 提供基本操作菜单
Scanner scanner = new Scanner(System.in);
boolean flag = true;
while (flag) {
    System.out.println("请选择操作：");
    System.out.println("1. 新增馆藏书籍");
    System.out.println("2. 删除书籍");
    System.out.println("3. 根据书名检索书籍");
    System.out.println("4. 借阅书籍");
    System.out.println("5. 归还书籍");
    System.out.println("6. 新增借书卡");
    System.out.println("7. 注销借书卡");
    System.out.println("8. 显示所有馆藏书籍");
    System.out.println("9. 显示所有借阅卡信息");
    System.out.println("0. 退出程序");

    int choice = scanner.nextInt();
    switch (choice) {
        case 1:
            addBook();
            break;
        case 2:
            deleteBook();
            break;
        case 3:
            searchBook();
            break;
        case 4:
            borrowBook();
```

```
                    break;
                case 5:
                    returnBook();
                    break;
                case 6:
                    addCard();
                    break;
                case 7:
                    deleteCard();
                    break;
                case 8:
                    showAllBooks();
                    break;
                case 9:
                    showAllCards();
                    break;
                case 0:
                    System.out.println("程序已退出");
                    System.exit(0);
                    break;
                default:
                    System.out.println("无效的选择，请重新选择");
                    break;
            }
        }

        // 关闭数据库连接
        try {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            if (conn != null) {
                conn.close();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

// 随机生成书籍的标题
private static String getRandomTitle() {
    String[] titles = {"Java从入门到精通", "Python编程基础", "数据结构与算法分析", "计算机网络原理", "人工智能导论",
            "深入浅出MySQL", "Web开发实战", "JavaScript高级程序设计", "C++程序设计基础", "算法竞赛入门经典"};
    int index = (int) (Math.random() * titles.length);
```

```
    return titles[index];
}

// 随机生成书籍的作者
private static String getRandomAuthor() {
    String[] authors = {"张三", "李四", "王五", "赵六", "钱七"};
    int index = (int) (Math.random() * authors.length);
    return authors[index];
}

// 随机生成书籍的出版社
private static String getRandomPress() {
    String[] presses = {"机械工业出版社", "电子工业出版社", "人民邮电出版社", "清华大学出版社",
"高等教育出版社"};
    int index = (int) (Math.random() * presses.length);
    return presses[index];
}

// 随机生成借书人
private static String getRandomName() {
    String[] names = { "张三", "李四", "王五", "赵六", "钱七" };
    int index = (int) (Math.random() * names.length);
    return names[index];
}

//下面是跟sql语句相关的 各个功能的实现

// 新增馆藏书籍
private static void addBook() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入书名: ");
    String title = scanner.nextLine();
    System.out.println("请输入作者: ");
    String author = scanner.nextLine();
    System.out.println("请输入出版社: ");
    String press = scanner.nextLine();
    String sql = "INSERT INTO book (title, author, press, status) VALUES (" +
            "'" + title + "'," +
            "'" + author + "'," +
            "'" + press + "'," +
            "0" +
            ")";
    try {
        stmt.executeUpdate(sql);
        System.out.println("新增书籍成功");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```
// 删除书籍
private static void deleteBook() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入要删除的书籍ID: ");
    int bookId = scanner.nextInt();
    String sql = "DELETE FROM book WHERE book_id=" + bookId;
    try {
        stmt.executeUpdate(sql);
        System.out.println("删除书籍成功");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// 根据书名检索书籍
private static void searchBook() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入书名关键词: ");
    String keyword = scanner.nextLine();
    String sql = "SELECT * FROM book WHERE title LIKE '%" + keyword + "%'";
    try {
        rs = stmt.executeQuery(sql);
        while (rs.next()) {
            int bookId = rs.getInt("book_id");
            String title = rs.getString("title");
            String author = rs.getString("author");
            String press = rs.getString("press");
            int status = rs.getInt("status");
            String statusStr = status == 0 ? "可借" : "已借出";
            System.out.println("书籍ID: " + bookId + ", 书名: " + title + ", 作者: " +
author + ", 出版社: " + press + ", 状态: " + statusStr);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// 借阅书籍
private static void borrowBook() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入借书证号: ");
    int cardId = scanner.nextInt();
    System.out.println("请输入书籍ID: ");
    int bookId = scanner.nextInt();

    // 检查借书证和书籍是否存在
    boolean cardExists = false, bookExists = false;
    String sql = "SELECT * FROM card WHERE card_id=" + cardId;
```

```
    try {
        rs = stmt.executeQuery(sql);
        if (rs.next()) {
            cardExists = true;
        } else {
            System.out.println("借书证不存在");
            return;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }

    sql = "SELECT * FROM book WHERE book_id=" + bookId;
    try {
        rs = stmt.executeQuery(sql);
        if (rs.next()) {
            int status = rs.getInt("status");
            if (status == 0) {
            bookExists = true;
            } else {
            System.out.println("书籍已借出");
            return;
            }
            } else {
            System.out.println("书籍不存在");
            return;
            }
            } catch (SQLException e) {
            e.printStackTrace();
            }
    // 借阅书籍
    if (cardExists && bookExists) {
        sql = "UPDATE book SET status=1 WHERE book_id=" + bookId;
        try {
            stmt.executeUpdate(sql);
        } catch (SQLException e) {
            e.printStackTrace();
        }

        java.sql.Date borrowDate = new java.sql.Date(System.currentTimeMillis());
        java.sql.Date returnDate = new java.sql.Date(System.currentTimeMillis() + 30 *
24 * 3600 * 1000L); // 借阅期限为30天
        sql = "INSERT INTO borrowRecord (card_id, book_id, borrow_date, return_date)
VALUES (" +
                cardId + "," +
                bookId + "," +
                "'" + borrowDate.toString() + "'," +
                "'" + returnDate.toString() + "'" +
                ")";
```

```
        try {
            stmt.executeUpdate(sql);
        } catch (SQLException e) {
            e.printStackTrace();
        }

        System.out.println("借阅成功，借阅日期：" + borrowDate.toString() + "，归还日期：" +
returnDate.toString());
    }
}

// 归还书籍
private static void returnBook() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入借书证号：");
    int cardId = scanner.nextInt();
    System.out.println("请输入书籍ID：");
    int bookId = scanner.nextInt();

    // 检查借书证和书籍是否存在
    boolean cardExists = false, bookExists = false, borrowed = false;
    String sql = "SELECT * FROM card WHERE card_id=" + cardId;
    try {
        rs = stmt.executeQuery(sql);
        if (rs.next()) {
            cardExists = true;
        } else {
            System.out.println("借书证不存在");
            return;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }

    sql = "SELECT * FROM book WHERE book_id=" + bookId;
    try {
        rs = stmt.executeQuery(sql);
        if (rs.next()) {
            int status = rs.getInt("status");
            if (status == 1) {
                bookExists = true;
            } else {
                System.out.println("书籍未借出");
                return;
            }
        } else {
            System.out.println("书籍不存在");
            return;
        }
```

```
    } catch (SQLException e) {
        e.printStackTrace();
    }

    // 归还书籍
    if (cardExists && bookExists) {
        sql = "SELECT * FROM borrowRecord WHERE card_id=" + cardId + " AND book_id=" +
bookId;
        try {
            rs = stmt.executeQuery(sql);
            if (rs.next()) {
                borrowed = true;
                int recordId = rs.getInt("record_id");
                java.sql.Date borrowDate = rs.getDate("borrow_date");
                java.sql.Date returnDate = rs.getDate("return_date");
                java.sql.Date actualReturnDate = new
java.sql.Date(System.currentTimeMillis());
                long days = (actualReturnDate.getTime() - returnDate.getTime()) / (24 *
3600 * 1000L);
                if (days > 0) {
                    System.out.println("超期" + days + "天，需要缴纳罚款：" + days * 0.1 +
"元");
                }
                sql = "DELETE FROM borrowRecord WHERE record_id=" + recordId;
                try {
                    stmt.executeUpdate(sql);
                } catch (SQLException e) {
                    e.printStackTrace();
                }

                sql = "UPDATE book SET status=0 WHERE book_id=" + bookId;
                try {
                    stmt.executeUpdate(sql);
                } catch (SQLException e) {
                    e.printStackTrace();
                }

                System.out.println("归还成功，借阅日期：" + borrowDate.toString() + "，应还
日期：" + returnDate.toString() + "，实际还书日期：" + actualReturnDate.toString());
            } else {
                System.out.println("借阅记录不存在");
                return;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

```
// 新增借书卡
private static void addCard() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入借书人姓名：");
    String name = scanner.nextLine();
    String sql = "INSERT INTO card (name) VALUES (" +
            "'" + name + "'" +
            ")";
    try {
        stmt.executeUpdate(sql);
        System.out.println("新增借书卡成功");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// 注销借书卡
private static void deleteCard() {
    Scanner scanner = new Scanner(System.in);
    System.out.println("请输入借书卡ID：");
    int cardId = scanner.nextInt();
    String sql = "DELETE FROM card WHERE card_id=" + cardId;
    try {
        stmt.executeUpdate(sql);
        System.out.println("注销借书卡成功");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// 显示所有馆藏书籍
private static void showAllBooks() {
    String sql = "SELECT * FROM book";
    try {
        rs = stmt.executeQuery(sql);
        while (rs.next()) {
            int bookId = rs.getInt("book_id");
            String title = rs.getString("title");
            String author = rs.getString("author");
            String press = rs.getString("press");
            int status = rs.getInt("status");
            String statusStr = status == 0 ? "可借" : "已借出";
            System.out.println(
                    "书籍ID：" + bookId + "，书名：" + title + "，作者：" + author + "，出版
社：" + press + "，状态：" + statusStr);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
```

```
}

// 显示所有借阅卡信息
private static void showAllCards() {
    String sql = "SELECT * FROM card";
    try {
        rs = stmt.executeQuery(sql);
        while (rs.next()) {
            int cardId = rs.getInt("card_id");
            String name = rs.getString("name");
            System.out.println("借阅卡ID: " + cardId + ", 姓名: " + name);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}


private static void createTable() {
        // 创建book、card和borrowRecord三张表
        try {
            String sql = "CREATE TABLE book (" +
                    "book_id INT NOT NULL AUTO_INCREMENT," +
                    "title VARCHAR(50)," +
                    "author VARCHAR(50)," +
                    "press VARCHAR(50)," +
                    "status INT," +
                    "PRIMARY KEY (book_id)" +
                    ")";
            stmt.executeUpdate(sql);

            sql = "CREATE TABLE card (" +
                    "card_id INT NOT NULL AUTO_INCREMENT," +
                    "name VARCHAR(50)," +
                    "PRIMARY KEY (card_id)" +
                    ")";
            stmt.executeUpdate(sql);

            sql = "CREATE TABLE borrowRecord (" +
                    "record_id INT NOT NULL AUTO_INCREMENT," +
                    "card_id INT," +
                    "book_id INT," +
                    "borrow_date DATE," +
                    "return_date DATE," +
                    "PRIMARY KEY (record_id)" +
                    ")";
            stmt.executeUpdate(sql);
        } catch (SQLException e) {
            e.printStackTrace();
```

```
        }

        // 初始化，向book表中插入20本随机书籍的记录
        try {
            for (int i = 0; i < 20; i++) {
                String sql = "INSERT INTO book (title, author, press, status) VALUES ("
+
                        "'" + getRandomTitle() + "'," +
                        "'" + getRandomAuthor() + "'," +
                        "'" + getRandomPress() + "'," +
                        "0" +
                        ")";
                stmt.executeUpdate(sql);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }

        // 初始化，向card表中插入5张随机借书证的记录，对应五个初始的借阅人
        try {
            for (int i = 0; i < 5; i++) {
                String sql = "INSERT INTO card (name) VALUES (" +
                        "'" + getRandomName() + "'" +
                        ")";
                stmt.executeUpdate(sql);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
}
}
```