



浙江大学
ZHEJIANG UNIVERSITY

实验 5 图书管理系统

2024-2025 春夏学期 数据库系统
课程实验报告

姓名 王浩雄

学号 3230106032

年级 2023 级

专业 混合班（计算机科学与技术）

班级 混合 2303 班

2025 年 4 月 2 日

目录

1 实验综述	3
1.1 实验目的	3
1.2 实验内容	3
2 实验环境	3
3 后端设计	4
3.1 数据库接口定义	4
3.2 方法实现细节	5
3.3 数据库模式	7
4 前端设计	8
4.1 图书管理 & 借书操作页面	8
4.2 借书记录查询 & 还书操作页面	8
4.3 借书证管理页面	9
5 系统功能验证与测试	10
5.1 正确性测试	10
5.2 功能性测试	10
5.2.1 图书入库	11
5.2.2 图书批量入库	12
5.2.3 增加图书库存	13
5.2.4 修改图书信息	14
5.2.5 查询借书证	15
5.2.6 添加借书证	16
5.2.7 删除借书证	17
5.2.8 图书查询与借书操作	18
5.2.9 借书记录查询与还书操作	19
6 思考题	20

实验 5 图书管理系统

1 实验综述

1.1 实验目的

1. 设计并实现一个基于 MySQL（或 OpenGauss, SQL Server）精简的图书管理程序，要求具有图书入库、查询、借书、还书、借书证管理等功能。该图书管理程序应具备较好的可扩展性、鲁棒性和安全性，并且在高并发场景下仍能正确运行。
2. 使用前端框架，完成图书管理系统的前端页面，使其成为一个用户能真正使用的图书管理系统。
3. 通过本实验，提高学生的系统编程能力，加深对数据库系统原理及应用的理解。

1.2 实验内容

1. 完成类 `LibraryManagementSystemImpl` 中各功能模块的函数，实现新增图书、修改图书信息、修改图书库存、删除图书、新增借书证、修改借书证信息、删除借书证、借书、还书等功能，并通过所有测试样例。
2. 使用提供的前端框架，正确完成图书管理系统的前端页面，使其成为一个用户能真正使用的图书管理系统。

2 实验环境

- 操作系统：
Windows 11 Pro 24H2（64 位操作系统，基于 x64 的处理器）
- DBMS 版本：
SQL Server Developer（64-bit）v16.0.1135.2
SQL Server Management Studio v20.2.30.0
- 集成开发环境：
IntelliJ IDEA 2024.3.4.1 (Ultimate Edition)
- JDK 版本：
Amazon Corretto 18.0.2
- NPM 版本：
10.9.2

3 后端设计

本项目的后端使用 SQL Server 和 Java 实现，接收来自前端的请求，调用实现的接口实现处理逻辑，并作出响应。

3.1 数据库接口定义

LibraryManagementSystem 接口定义了图书馆管理系统的核心功能，主要包括以下方法：

- `storeBook(Book book)`: 新增单本图书
- `incBookStock(int bookId, int deltaStock)`: 调整图书库存
- `storeBook(List<Book> books)`: 批量新增图书
- `removeBook(int bookId)`: 删除图书
- `modifyBookInfo(Book book)`: 修改图书信息
- `queryBook(BookQueryConditions conditions)`: 多条件复合查询图书
- `borrowBook(Borrow borrow)`: 处理借书操作
- `returnBook(Borrow borrow)`: 处理还书操作
- `showBorrowHistory(int cardId)`: 查询借书证下所有借阅记录
- `showBorrowHistory()`: 查询全部借书证所有借阅记录
- `registerCard(Card card)`: 新增借书证
- `removeCard(int cardId)`: 删除借书证
- `showCards()`: 查询所有借书证
- `modifyCardInfo(Card card)`: 修改借书证信息
- `resetDatabase()`: 重置数据库

3.2 方法实现细节

系统通过 `LibraryManagementSystemImpl` 类实现核心功能，并针对并发访问情形进行特殊处理。下面以 `borrowBook` 方法为例具体阐述关键实现要点。

1. 排他锁控制库存访问

```
1 String checkStockSql = "SELECT stock FROM book WITH (XLOCK)
  WHERE book_id = ?";
2 PreparedStatement checkStockStmt = conn.prepareStatement(
  checkStockSql);
```

- 使用 SQL Server 的 `WITH (XLOCK)` 提示获取排他锁，防止其他事务同时修改同一图书库存
- 防止出现多个事务同时进行借书操作，导致库存为负的情况发生

2. 库存可用性验证

```
1 if (checkStockRs.next()) {
2     int stock = checkStockRs.getInt("stock");
3     if (stock <= 0) {
4         rollback(conn);
5         return new ApiResult(false, "库存不足");
6     }
7 }
```

- 检查库存是否大于 0，保证借书操作的库存有效性

3. 借阅状态验证

```
1 String checkBorrowSql = "SELECT return_time FROM borrow ...";
2 if (checkBorrowRs.next() && checkBorrowRs.getLong("return_time")
  == 0) {
3     rollback(conn);
4     return new ApiResult(false, "存在未归还记录");
5 }
```

- 通过 `return_time=0` 标识未归还状态
- 防止同一用户在未归还的情况下重复借阅同一图书

4. 原子事务操作

```
1 insertBorrowStmt.executeUpdate(); // 插入借阅记录
2 updateStockStmt.executeUpdate(); // 更新库存
3 commit(conn); // 事务提交
```

- 借阅记录与库存更新保持原子性
- 显式提交，保证操作持久化
- 异常时执行完整回滚

5. 时间格式管理

```
1 insertBorrowStmt.setLong(3, borrow.getBorrowTime()); // Unix 时  
   间戳
```

- 数据库采用整型存储 Unix 毫秒时间戳,而前端使用"YYYY-MM-DD HH:MM"的格式存储时间字符串
- 通过两种时间格式的转换,实现时间的正确处理

该实现方案通过排他锁 + 事务隔离保障并发安全,其操作时序可表示为:

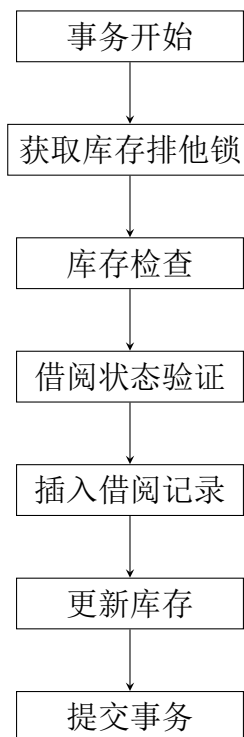


图 1: 借书操作事务流程图

3.3 数据库模式

核心表结构如下：

```
1 create table book (
2 book_id int not null identity,
3 category varchar(63) not null,
4 title varchar(63) not null,
5 press varchar(63) not null,
6 publish_year int not null,
7 author varchar(63) not null,
8 price decimal(7, 2) not null default 0.00,
9 stock int not null default 0,
10 primary key (book_id),
11 unique (category, press, author, title, publish_year)
12 );
13
14 create table card (
15 card_id int not null identity,
16 name varchar(63) not null,
17 department varchar(63) not null,
18 type char(1) not null,
19 primary key (card_id),
20 unique (department, type, name),
21 check ( type in ('T', 'S') )
22 );
23
24 create table borrow (
25 card_id int not null,
26 book_id int not null,
27 borrow_time bigint not null,
28 return_time bigint not null default 0,
29 primary key (card_id, book_id, borrow_time),
30 foreign key (card_id) references card(card_id) on delete cascade on
    update cascade,
31 foreign key (book_id) references book(book_id) on delete cascade on
    update cascade
32 );
```

4 前端设计

本项目的前端使用 Vue3 为框架，并使用 Element Plus 组件库，与用户进行交互，向后端发出请求并处理响应。前端由三个页面组成，分别对应图书管理与借书操作功能、借书记录查询与还书操作功能、借书证管理功能。

4.1 图书管理 & 借书操作页面

该页面实现如下功能：

1. 按检索条件查询图书信息
2. 新增图书，支持单本添加与 CSV 批量导入
3. 编辑图书信息，修改图书库存
4. 删除图书
5. 进行借书操作



图 2: 图书管理 & 借书操作页面

4.2 借书记录查询 & 还书操作页面

该页面实现如下功能：

1. 按检索条件查询借书记录，支持按借书证 ID 与图书 ID 进行查询
2. 在检索结果中，展示图书名，使操作更加直观便捷
3. 进行还书操作



图 3: 借书记录查询 & 还书操作页面

4.3 借书证管理页面

该页面实现如下功能：

1. 按检索条件查询借书证信息
2. 新增借书证
3. 编辑借书证信息
4. 删除借书证



图 4: 借书证管理页面

5 系统功能验证与测试

5.1 正确性测试

正确性测试通过测试用例进行评判。本地测试中，程序已通过全部 9 组测试用例，验证了其功能实现的正确性。

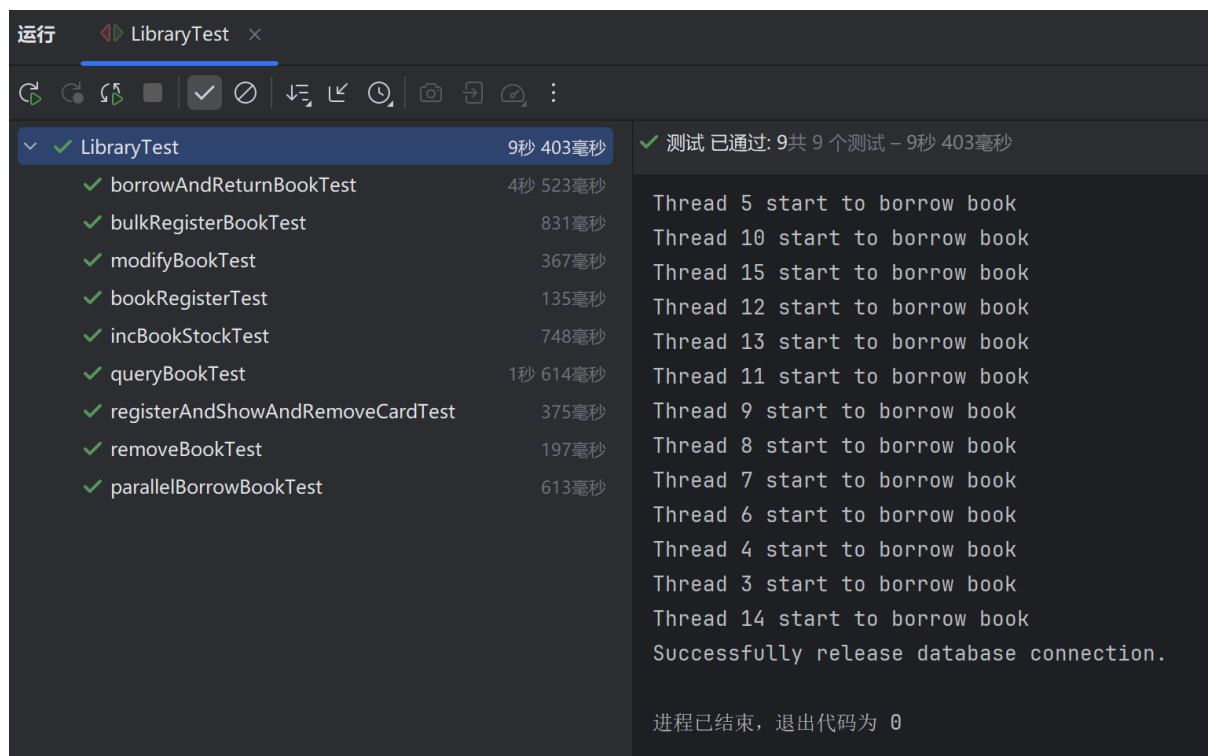


图 5: 测试用例执行结果

5.2 功能性测试

功能性测试通过随机运行模拟场景的结果进行评判，考察软件使用时的交互友好程度、效率、正确性等指标。完成前端实现后，我对下述几种典型使用场景进行了测试。在测试中，程序已通过全部测试场景，验证了其功能实现的正确性。

5.2.1 图书入库

在“图书管理 & 借书操作”页面单击“新增图书”按钮，在弹窗内输入新增图书的相关信息，然后单击“提交”按钮，可以观察到新增图书对应的检索条目。



新增图书弹窗，包含以下输入项：

- 图书名: 数据库系统
- 类别: Computer Science
- 出版社: 浙江大学出版社
- 出版年份: 2024
- 作者: 王浩雄
- 价格: 23.98
- 库存: 12

底部有“取消”和“提交”按钮。

图 6: “新增图书”弹窗

图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
2	数据库系统	Computer Science	浙江大学出版社	2024	王浩雄	23.98	12	借书 编辑信息 修改库存 删除

图 7: 图书检索条目

5.2.2 图书批量入库

在“图书管理 & 借书操作”页面单击“批量导入”按钮，在弹窗内按指定格式上传 CSV 格式文档，然后单击“提交”按钮，可以观察到新增图书对应的检索条目。测试时上传的 CSV 文件内容如下：

```
1 title,category,press,author,publishYear,price,stock
2 Database System,Computer Science,Zhejiang University Press,Wang
   Haoxiong,2022,128.88,21
3 Algorithms in C++,Computer Science,China Machine Press,Chen Yue
   ,2018,39.99,12
```



图 8: “批量导入”弹窗

<div>查询 新增图书 批量导入</div>								
图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
1	Compiler Designs	Philosophy	Press-B	2003	Nonehyo	63.77	0	<button>借书</button> <button>编辑信息</button> <button>修改库存</button> <button>删除</button>
2	数据库系统	Computer Science	浙江大学出版社	2024	王浩雄	23.98	12	<button>借书</button> <button>编辑信息</button> <button>修改库存</button> <button>删除</button>
4	Database System	Computer Science	Zhejiang University Press	2022	Wang Haoxiong	128.88	21	<button>借书</button> <button>编辑信息</button> <button>修改库存</button> <button>删除</button>
5	Algorithms in C++	Computer Science	China Machine Press	2018	Chen Yue	39.99	12	<button>借书</button> <button>编辑信息</button> <button>修改库存</button> <button>删除</button>

图 9: 图书检索条目

5.2.3 增加图书库存

在“图书管理 & 借书操作”页面单击图书条目右侧的“修改库存”按钮，在弹窗内输入库存增量，然后单击“提交”按钮，可以观察到图书库存量的变更。

图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
1	Compiler Designs	Philosophy	Press-B	2003	Nonehyo	63.77	0	借书 编辑信息 修改库存 删除

图 10: 图书库存修改前

修改库存

库存增量

[取消](#) [提交](#)

图 11: “修改库存”弹窗

图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
1	Compiler Designs	Philosophy	Press-B	2003	Nonehyo	63.77	5	借书 编辑信息 修改库存 删除

图 12: 图书库存修改后

5.2.4 修改图书信息

在“图书管理 & 借书操作”页面单击图书条目右侧的“编辑信息”按钮，在弹窗内修改图书信息，然后单击“提交”按钮，可以观察到图书信息的变更。

图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
1	Compiler Designs	Philosophy	Press-B	2003	Nonehyo	63.77	5	借书 编辑信息 修改库存 删除

图 13: 图书信息修改前

编辑信息

图书名

Compiler Designs

类别

Philosophy

出版社

Press-B

出版年份

2025

作者

Nonehyo

价格

88.48

取消

提交

图 14: “编辑信息”弹窗

图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
1	Compiler Designs	Philosophy	Press-B	2025	Nonehyo	88.48	5	借书 编辑信息 修改库存 删除

图 15: 图书信息修改后

5.2.5 查询借书证

在“借书证管理”页面右上角的“筛选条件”文本框内输入筛选条件“3”，可以观察到筛选后的借书证卡片结果，它们的姓名都包含字符“3”。



图 16: “筛选条件”文本框

借书证管理



图 17: 筛选结果

5.2.6 添加借书证

在“借书证管理”页面单击加号卡片，在弹窗内输入新借书证的信息，然后单击“确定”按钮，可以观察到新增的借书证卡片。



图 18: “新建借书证”弹窗



图 19: 新增的借书证卡片

5.2.7 删除借书证

在“借书证管理”页面单击借书证卡片下方的“删除”按钮，在弹窗内单击“确定”按钮进行二次确认，可以观察到借书证卡片被移除。

借书证管理



图 20: 借书证删除前

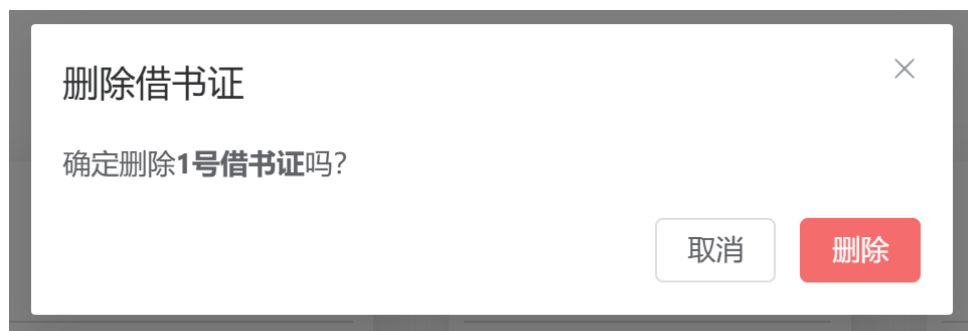


图 21: “确认删除”弹窗

5.2.8 图书查询与借书操作

在“图书管理 & 借书操作”页面键入若干检索条件，然后单击“查询”按钮，可以观察到图书对应的检索条目，条目右侧显示一个“借书”按钮，单击后在弹窗内输入借书证 ID 即可完成借书。

Figure 22 shows the search condition input bar. It includes input fields for Book ID, Book Name, Category (set to Computer Science), Publisher, Author, Publication Year (range 2020-2024), Price (range minimum-maximum), and Inventory (range minimum-maximum). There are buttons for Search, Add New Book, and Batch Import.

图 22: 检索条件输入栏

图书ID	图书名	类别	出版社	出版年份	作者	价格	库存	操作
2	数据库系统	Computer Science	浙江大学出版社	2024	王浩雄	23.98	12	借书 编辑信息 修改库存 删除
4	Database System	Computer Science	Zhejiang University Press	2022	Wang Haoxiong	128.88	26	借书 编辑信息 修改库存 删除

图 23: 图书检索条目

Figure 24 shows the "Borrow Book" modal window. It has a title "借书" and a close button. The "借书证ID" (Borrower ID) field contains the value "71". There are "取消" (Cancel) and "确认借阅" (Confirm Borrow) buttons.

图 24: “借书”弹窗

5.2.9 借书记录查询与还书操作

在“图书管理 & 借书操作”页面键入若干检索条件，然后单击“查询”按钮，可以观察到借书记录对应的检索条目。在检索时，同时支持按借书证 ID 和图书 ID 进行检索。若图书未归还，则对应的检索条目右侧显示一个“还书”按钮，单击即可完成还书。

借书证ID 17

图书ID 输入图书ID

筛选条件 输入筛选条件

查询

借书证ID	图书ID	图书名	借出时间	归还时间	操作
17	5	Algorithms in C++	2025.04.02 15:37	未归还	还书
17	2	数据库系统	2025.04.02 15:37	未归还	还书

图 25: 按借书证 ID 进行的检索

借书证ID 输入借书证ID

图书ID 5

筛选条件 输入筛选条件

查询

借书证ID	图书ID	图书名	借出时间	归还时间	操作
17	5	Algorithms in C++	2025.04.02 15:37	未归还	还书
7	5	Algorithms in C++	2025.04.02 15:40	2025.04.02 15:40	还书
2	5	Algorithms in C++	2025.04.02 15:40	2025.04.02 15:40	还书

图 26: 按图书 ID 进行的检索

借书证ID	图书ID	图书名	借出时间	归还时间	操作
17	2	数据库系统	2025.04.02 15:37	未归还	还书
17	5	Algorithms in C++	2025.04.02 15:37	2025.04.02 15:41	还书

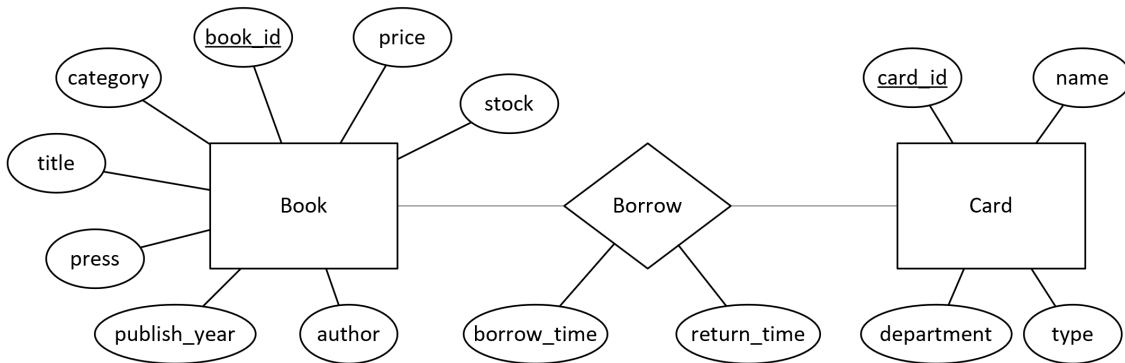
图 27: 已还书和未还书的条目区别

6 思考题

第一题

题目

绘制该图书管理系统的 E-R 图。



第二题

题目

描述 SQL 注入攻击的原理并简要举例。在图书管理系统中，哪些模块可能会遭受 SQL 注入攻击？如何解决？

SQL 注入攻击原理：

SQL 注入是通过将恶意 SQL 代码插入用户输入的参数中，利用应用程序未正确过滤输入的漏洞，欺骗数据库执行非授权操作。

假设一个管理系统的登录场景，登录功能的 SQL 语句为：

```
1 SELECT * FROM users WHERE username = '$user' AND password = '$pass'
```

若用户输入的用户名为 admin' --，密码任意，则进行 SQL 语句拼接后，得到的语句为：

```
1 SELECT * FROM users WHERE username = 'admin' --' AND password = '
  $pass'
```

由于--表示注释，因此该语句可视同为如下语句，从而跳过了对 password 的检验。

```
1 SELECT * FROM users WHERE username = 'admin'
```

易受攻击的模块：

- **图书查询接口：**联合查询注入攻击（如 红楼梦'; DROP TABLE books; --）可能导致数据被破坏。
- **借书证查询接口：**若将本系统的功能进一步扩展，存储借书证密码等敏感信息，非法的查询注入攻击可能导致个人信息泄露。
- **借还书操作接口：**使用注入攻击，可能导致借还书记录被篡改或删除，导致图书不可追踪，造成财产损失。

解决办法：

- **使用参数化查询：**本系统使用 JDBC 的参数化查询语句，自动对注入攻击行为进行屏蔽。
- **正则表达式匹配：**若必须使用 SQL 语句拼接，可在输入框等处使用正则表达式匹配非法符号，限制非法符号的输入。
- **设置最小权限：**数据库连接账号仅授予必要权限（如禁止 DROP、DELETE 等高风险操作）。
- **错误处理：**避免暴露数据库错误详情、表的定义、SQL 语句构造方法等信息，防止被攻击者利用。

第三题

题目

在 InnoDB 的默认隔离级别（Repeated Read）下，当出现并发访问时如何保证借书结果的正确性？

本程序使用 SQL Server 排他锁控制相关表的访问，使用 SQL Server 的 WITH (XLOCK) 提示获取排他锁，防止其他事务同时修改同一图书库存，导致库存为负的情况发生。

```
1 String checkStockSql = "SELECT stock FROM book WITH (XLOCK) WHERE  
    book_id = ?";  
2 PreparedStatement checkStockStmt = conn.prepareStatement(  
    checkStockSql);
```