浙江大学

本科实验报告

课程名称：	数据库系统
姓　　名：	姜雨童
学　　院：	计算机科学与技术学院
系：	计算机科学与技术
专　　业：	计算机科学与技术
学　　号：	3220103450

2024 年　4 月 28 日

1 实验目的

设计并实现一个精简的图书管理系统，要求具有图书入库、查询、借书、还书、借书证管理等功能。通过该图书馆系统的设计与实现，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。

2 实验需求

（1）基本数据对象

对象名称	包含属性
书	书号, 类别, 书名, 出版社, 年份, 作者, 价格, 总藏书量, 库存
借书证	卡号, 姓名, 单位, 类别 (教师 学生等)
管理员	管理员 ID, 密码, 姓名, 联系方式
借书记录	卡号, 借书证号 ,借期, 还期, 经手人 (管理员 ID)

（2）基本功能模块

模块名称	功能描述
管理员登陆	输入管理员 ID, 密码; 登入系统 或 返回 ID/密码 错误.
图书入库	1. 单本入库 2. 批量入库 (方便最后测试) 图书信息存放在文件中, 每条图书信息为一行. 一行中的内容如下 (书号, 类别, 书名, 出版社, 年份, 作者, 价格, 数量) Note: 其中 年份、数量是整数类型; 价格是两位小数类型; 其余为字符串类型 Sample: (book_no_1, Computer Science, Computer Architecture, xxx, 2004, xxx, 90.00, 2)
图书查询	要求可以对书的 类别, 书名, 出版社, 年份(年份区间), 作者, 价格(区间) 进行查询. 每条图书信息包括以下内容: (书号, 类别, 书名, 出版社, 年份, 作者, 价格, 总藏书量, 库存) 可选要求: 可以按用户指定属性对图书信息进行排序. (默认是书名)
借书	1.输入借书证卡号 显示该借书证所有已借书籍 (返回, 格式同查询模块) 2.输入书号 如果该书还有库存，则借书成功，同时库存数减一。 否则输出该书无库存，且输出最近归还的时间。
还书	1.输入借书证卡号 显示该借书证所有已借书籍 (返回, 格式同查询模块) 2.输入书号 如果该书在已借书籍列表内, 则还书成功, 同时库存加一.

	否则输出出错信息.
借书证管理	增加或删除一个借书证.

除图书查询功能外，其余功能模块都应该由图书管理员操作。

（3）用户界面

可采用图形界面或字符界面。如果采用图形或网页界面，酌情加分。

（4）数据库平台

SQL Server 或 MySQL

其中 MySql 详细信息请参见 http://www.mysql.com

MySql APIs:

1. MySql ODBC 3.51
2. MySql JDBC 5.0
3. MySql PHP APIs

有关 MySql 的安装，请参看有关参考书。

（5）开发工具

任选（如 VC++, PHP, Java, Delphi, PowerBuilder 等）

3 实验环境

使用 IntelliJ IDEA 作为集成开发环境，使用 sql server 作为数据库

根据自己的主机名等配置端口如下：

```
host: "FINE"
port: "1433"
user: "sa"
password: "databasesa"
db: "library"
type: "sqlserver"
```

4 系统各模块的设计思路和实现

数据表定义：

```
create table `book` (
    `book_id` int not null auto_increment,
    `category` varchar(63) not null,
    `title` varchar(63) not null,
    `press` varchar(63) not null,
    `publish_year` int not null,
    `author` varchar(63) not null,
    `price` decimal(7, 2) not null default 0.00,
    `stock` int not null default 0,
    primary key (`book_id`),
    unique (`category`, `press`, `author`, `title`, `publish_year`)
);
```

```
create table `card` (
    `card_id` int not null auto_increment,
    `name` varchar(63) not null,
    `department` varchar(63) not null,
    `type` char(1) not null,
    primary key (`card_id`),
    unique (`department`, `type`, `name`),
    check ( `type` in ('T', 'S') )
);

create table `borrow` (
  `card_id` int not null,
  `book_id` int not null,
  `borrow_time` bigint not null,
  `return_time` bigint not null default 0,
  primary key (`card_id`, `book_id`, `borrow_time`),
  foreign key (`card_id`) references `card`(`card_id`) on delete cascade on update
cascade,
  foreign key (`book_id`) references `book`(`book_id`) on delete cascade on update
cascade
);
```

功能函数接口：

- ApiResult storeBook(Book book)：图书入库模块。向图书库中注册(添加)一本新书，并返回新书的书号。如果该书已经存在于图书库中，那么入库操作将失败。当且仅当书的<类别，书名，出版社，年份，作者>均相同时，才认为两本书相同。请注意，book_id 作为自增列，应该插入时由数据库生成。插入完成后，需要根据数据库生成的 book_id 值去更新 book 对象里的 book_id。
- ApiResult incBookStock(int bookId, int deltaStock)：图书增加库存模块。为图书库中的某一本书增加库存。其中库存增量 deltaStock 可正可负，若为负数，则需要保证最终库存是一个非负数。
- ApiResult storeBook(List<Book> books)：图书批量入库模块。批量入库图书，如果有一本书入库失败，那么就需要回滚整个事务(即所有的书都不能被入库)。
- ApiResult removeBook(int bookId)：图书删除模块。从图书库中删除一本书。如果还有人尚未归还这本书，那么删除操作将失败。
- ApiResult modifyBookInfo(Book book)：图书修改模块。修改已入库图书的基本信息，该接口不能修改图书的书号和存量。
- ApiResult queryBook(BookQueryConditions conditions)：图书查询模块。根据提供的查询条件查询符合条件的图书，并按照指定排序方式排序。查询条件包括：类别点查(精确查询)，书名点查(模糊查询)，出版社点查(模糊查询)，年份范围查，作者点查(模糊查询)，价格范围差。如果两条记录排序条件的值相等，则按 book_id 升序排序。
- ApiResult borrowBook(Borrow borrow)：借书模块。根据给定的书号、卡号和借书时间添加一条借书记录，然后更新库存。若用户此前已经借过这本书但尚未归还，那么借书操作将失败。
- ApiResult returnBook(Borrow borrow)：还书模块。根据给定的书号、卡号和还书时间，查询对应的借书

记录，并补充归还时间，然后更新库存。

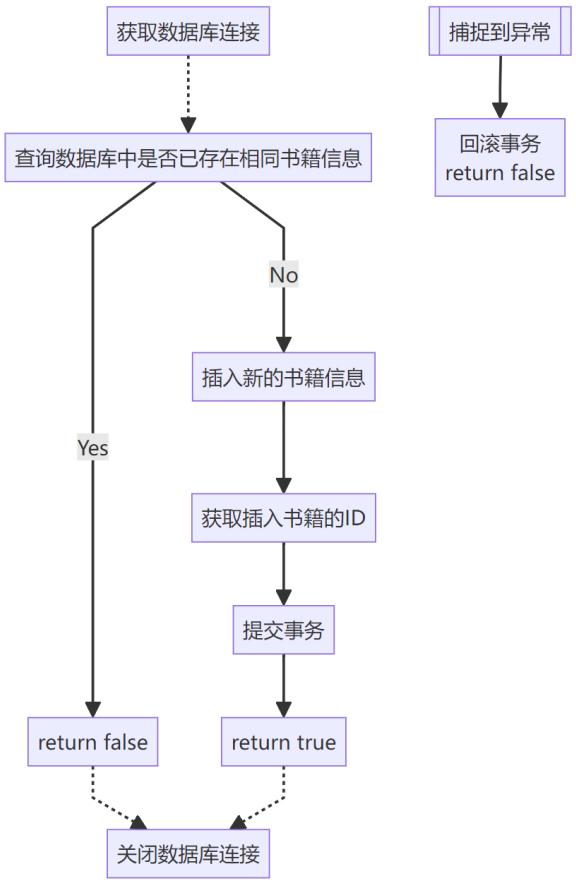
- ApiResult showBorrowHistory(int cardId)：借书记录查询模块。查询某个用户的借书记录，按照借书时间递减、书号递增的方式排序。
- ApiResult registerCard(Card card)：借书证注册模块。注册一个借书证，若借书证已经存在，则该操作将失败。当且仅当<姓名，单位，身份>均相同时，才认为两张借书证相同。
- ApiResult removeCard(int cardId)：删除借书证模块。如果该借书证还有未归还的图书，那么删除操作将失败。
- ApiResult showCards()：借书证查询模块。列出所有的借书证。

Implement：

完整源代码较多，可以查看 java 文件夹下相关文件，此处不做赘述。

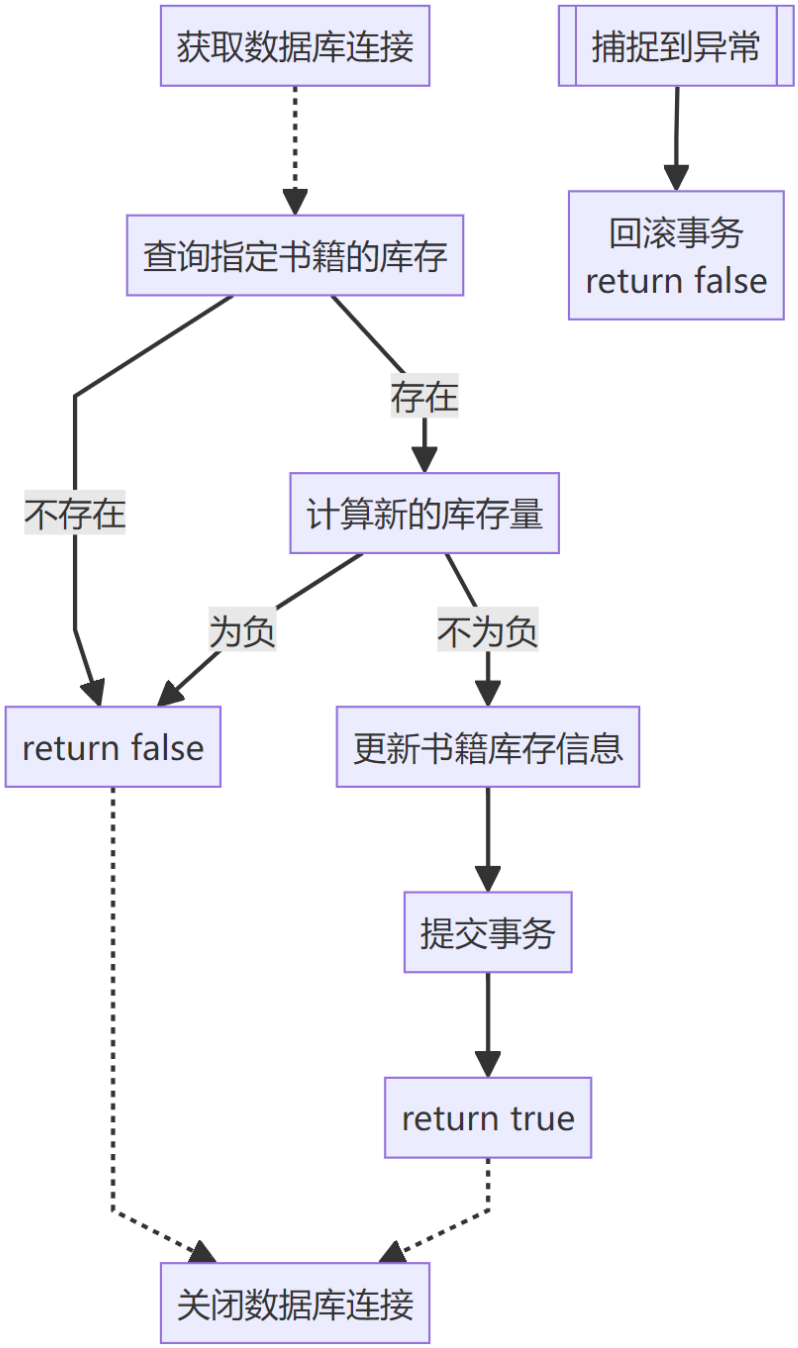
1.StoreBook(Book book)

```
1. 获取数据库连接
2. 查询数据库中是否已存在相同的书籍信息
3. 如果书籍已存在，返回错误信息
4. 如果书籍不存在，插入新的书籍信息
5. 获取插入的书籍 ID
6. 提交事务
7. 返回成功信息和书籍 ID
8. 异常处理：回滚事务并返回错误信息
9. 关闭数据库连接
```



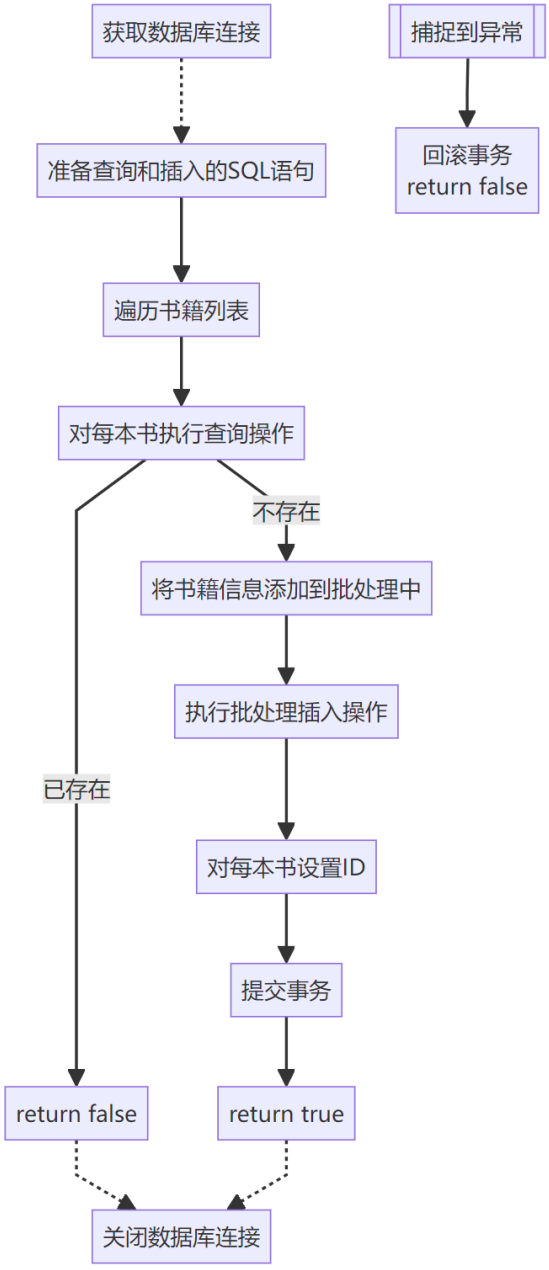
2.incBookStock(int bookId, int deltaStock)

1. 获取数据库连接
2. 查询指定书籍的库存
3. 如果书籍不存在，返回错误信息
4. 计算新的库存量
5. 如果新库存量为负数，返回错误信息
6. 更新书籍库存信息
7. 提交事务
8. 返回成功信息
9. 异常处理：回滚事务并返回错误信息
10. 关闭数据库连接



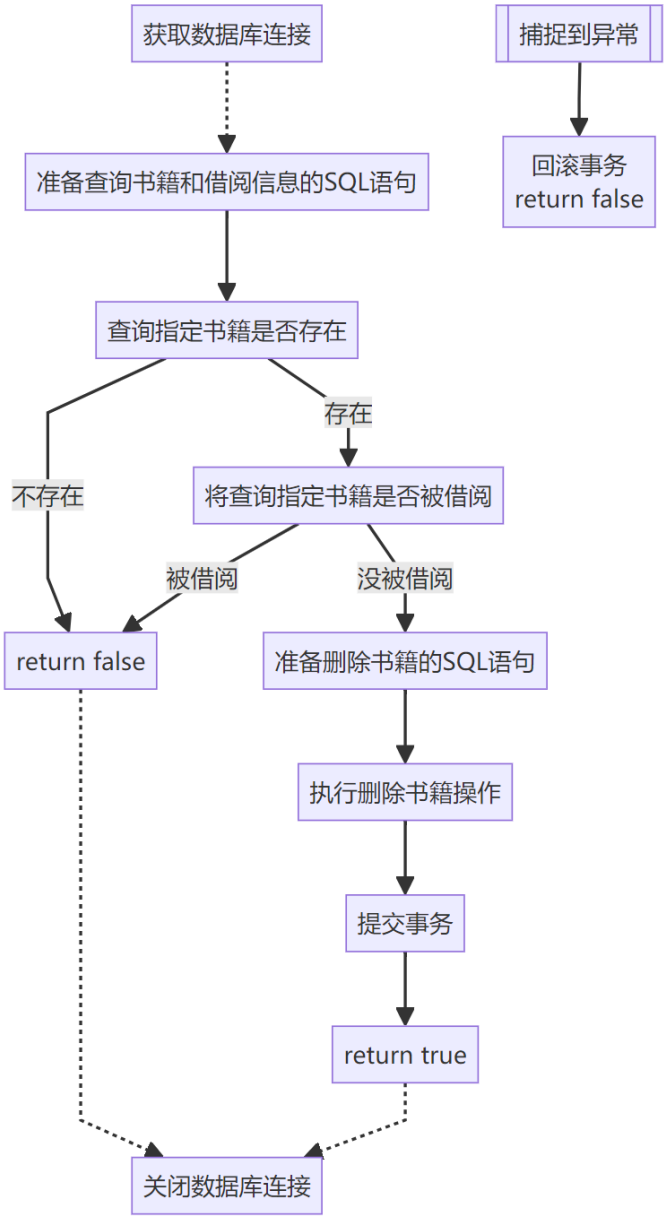
3. storeBook(List<Book> books)

1. 获取数据库连接
2. 准备查询和插入的 SQL 语句
3. 遍历书籍列表
4. 对每本书执行查询操作
5. 如果书籍已存在，回滚操作并返回错误信息
6. 否则，将书籍信息添加到批处理中
7. 执行批处理插入操作
8. 对每本书设置书籍 ID
9. 提交事务
10. 返回成功信息
11. 异常处理：回滚事务并返回错误信息
12. 关闭数据库连接



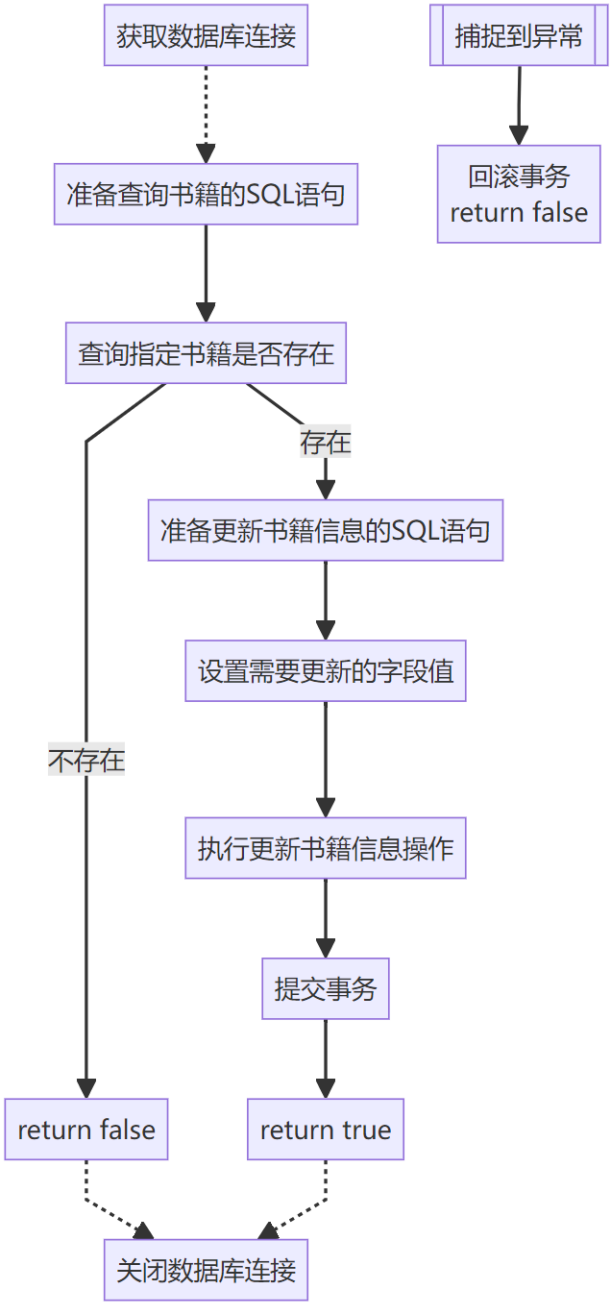
4. removeBook(int bookId)

1. 获取数据库连接
2. 准备查询书籍和借阅信息的 SQL 语句
3. 查询指定书籍是否存在
4. 如果书籍不存在，返回错误信息
5. 查询指定书籍是否被借阅
6. 如果书籍被借阅，返回错误信息
7. 准备删除书籍的 SQL 语句
8. 执行删除书籍操作
9. 提交事务
10. 返回成功信息
11. 异常处理：回滚事务并返回错误信息
12. 关闭数据库连接



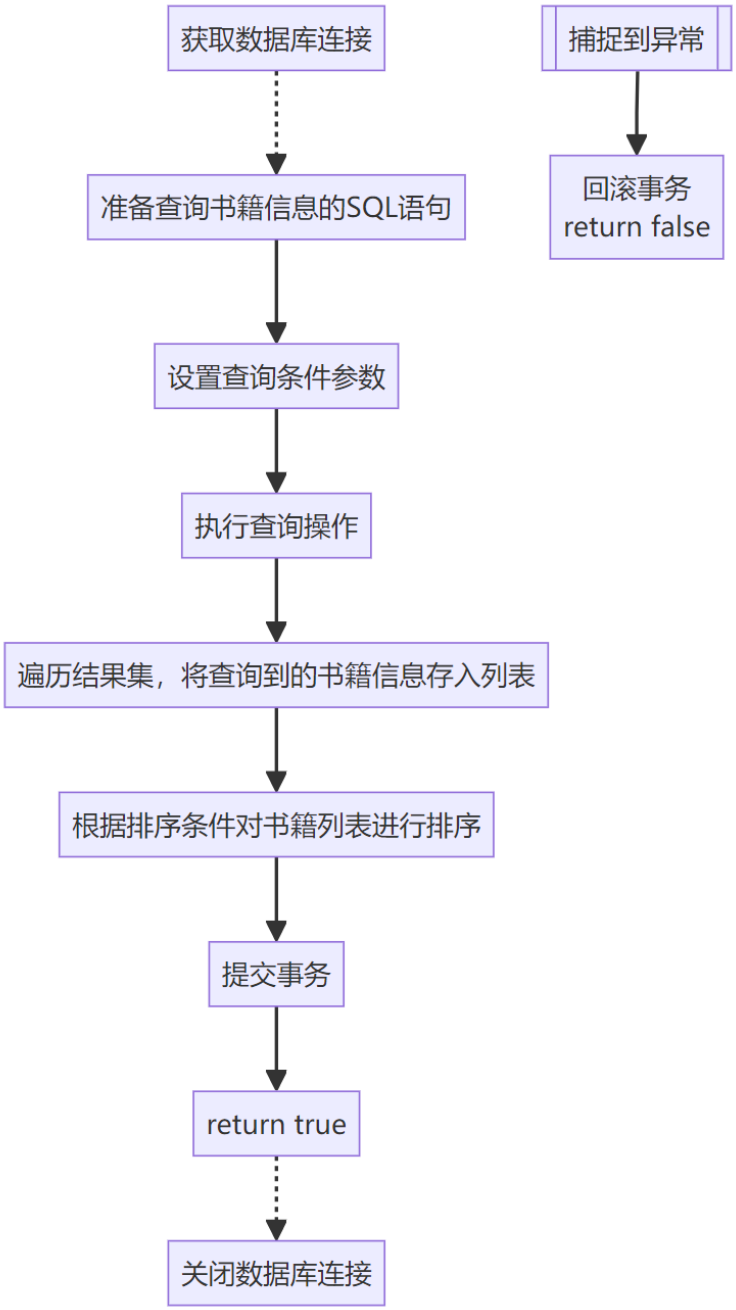
5. modifyBookInfo(Book book)

1. 获取数据库连接
2. 准备查询书籍信息的 SQL 语句
3. 查询指定书籍是否存在
4. 如果书籍不存在，返回错误信息
5. 准备更新书籍信息的 SQL 语句
6. 设置需要更新的字段值
7. 执行更新书籍信息操作
8. 提交事务
9. 返回成功信息
10. 异常处理：回滚事务并返回错误信息
11. 关闭数据库连接



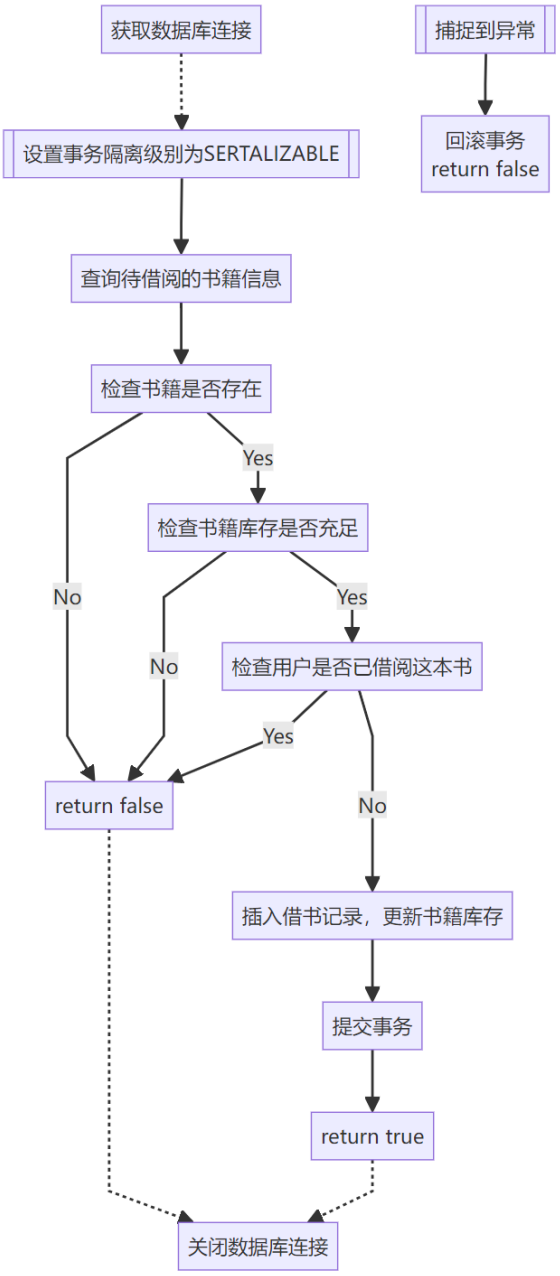
6. queryBook(BookQueryConditions conditions)

1. 获取数据库连接
2. 准备查询书籍信息的 SQL 语句
3. 设置查询条件参数
4. 执行查询操作
5. 遍历结果集，将查询到的书籍信息存入列表
6. 根据排序条件对书籍列表进行排序
7. 提交事务
8. 返回成功信息和查询结果
9. 异常处理：回滚事务并返回错误信息
10. 关闭数据库连接



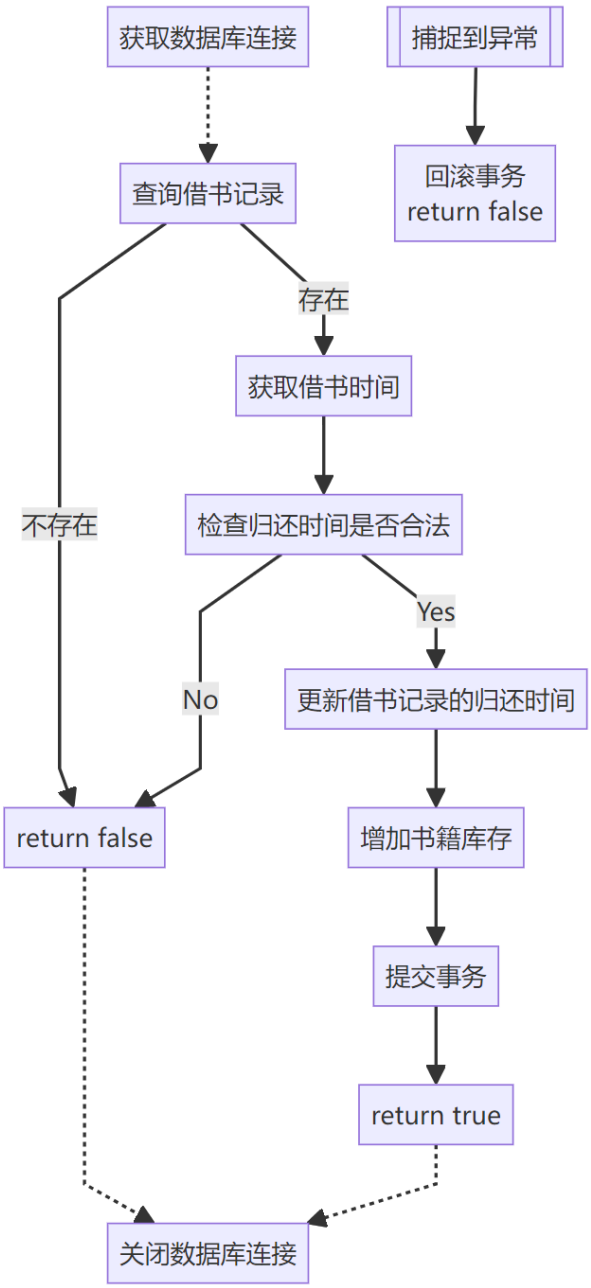
7. borrowBook(Borrow borrow)

1. 获取数据库连接
2. 设置事务隔离级别为 SERIALIZABLE
3. 查询待借阅的书籍信息
4. 检查书籍是否存在
5. 检查书籍库存是否充足
6. 检查用户是否已经借阅了这本书
7. 如果未借阅，则执行借书操作：插入借书记录，更新书籍库存
8. 提交事务
9. 返回成功信息
10. 异常处理：回滚事务并返回错误信息
11. 关闭数据库连接



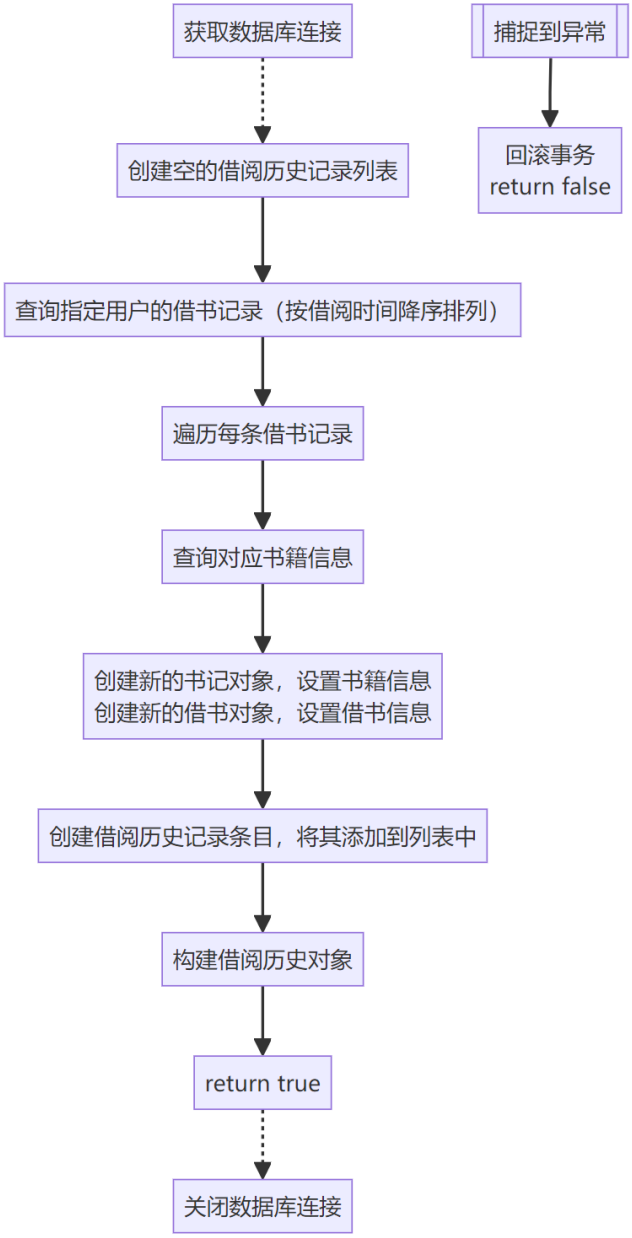
8. returnBook(Borrow borrow)

1. 获取数据库连接
2. 查询借书记录
3. 检查是否存在借书记录
4. 获取借书时间
5. 检查归还时间是否合法
6. 更新借书记录的归还时间
7. 增加书籍库存
8. 提交事务
9. 返回成功信息
10. 异常处理：回滚事务并返回错误信息
11. 关闭数据库连接



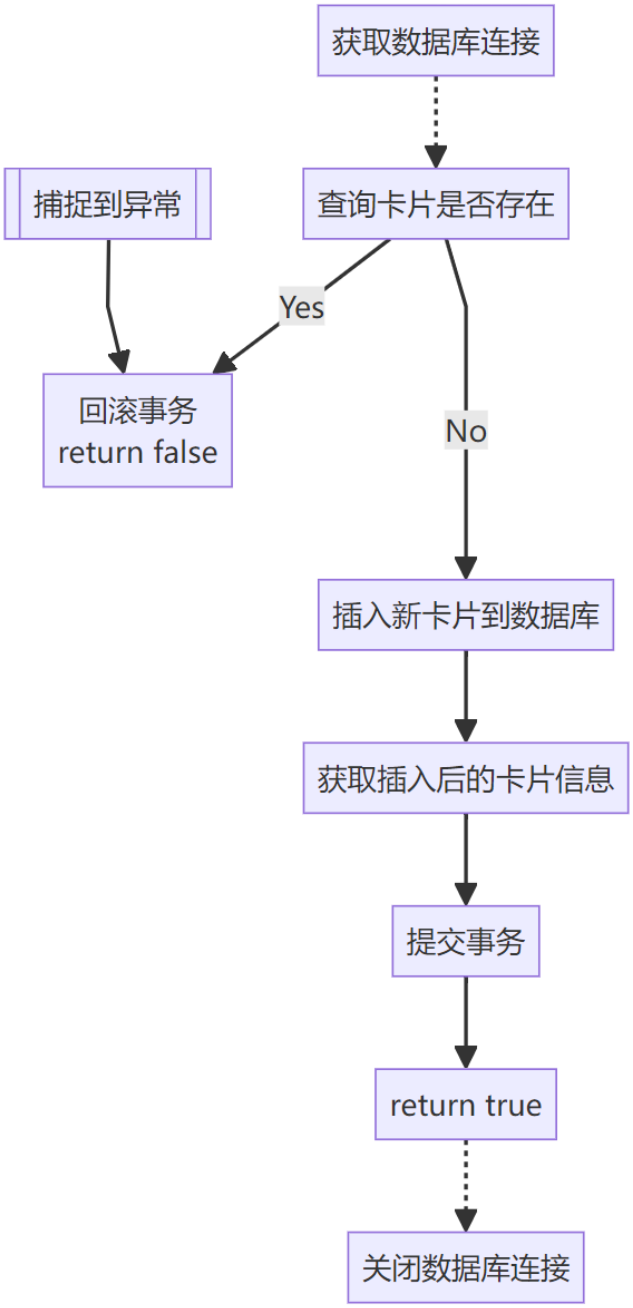
9. showBorrowHistory(int cardId)

1. 获取数据库连接
2. 创建空的借阅历史记录列表
3. 查询指定用户的借书记录，按借阅时间降序和书籍 ID 升序排序
4. 遍历每条借书记录
5. 查询对应的书籍信息
6. 创建新的书籍对象，并设置书籍信息
7. 创建新的借书对象，并设置借书信息
8. 创建借阅历史记录条目，将其添加到历史记录列表中
9. 构建借阅历史对象
10. 返回成功信息和借阅历史对象
11. 异常处理：回滚事务并返回错误信息
12. 关闭数据库连接



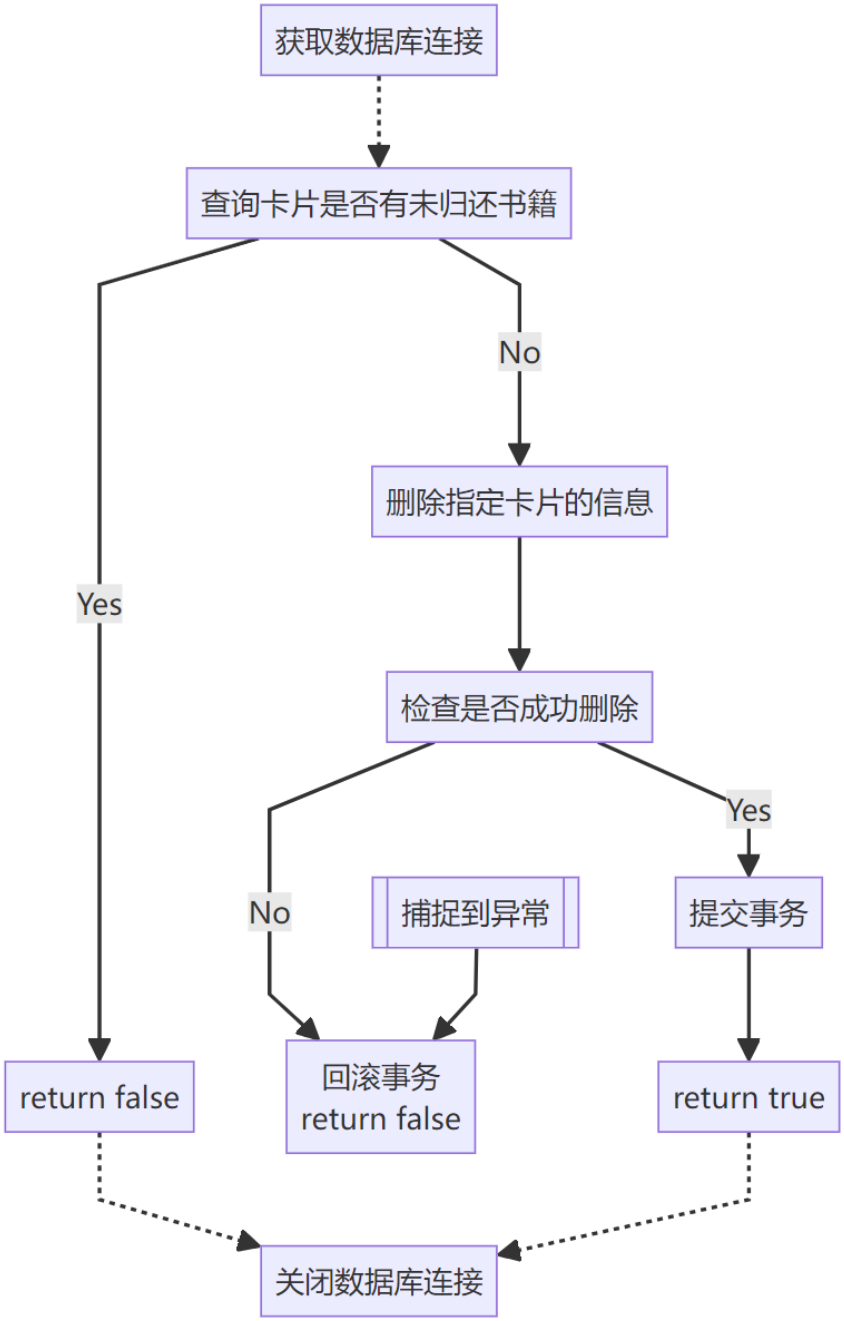
10. registerCard(Card card)

1. 获取数据库连接
2. 查询卡片是否已存在
3. 如果卡片已存在，回滚事务并返回错误信息
4. 如果卡片不存在，插入新卡片信息到数据库
5. 获取插入后的卡片信息
6. 提交事务
7. 返回成功信息
8. 异常处理：回滚事务并返回错误信息
9. 关闭数据库连接



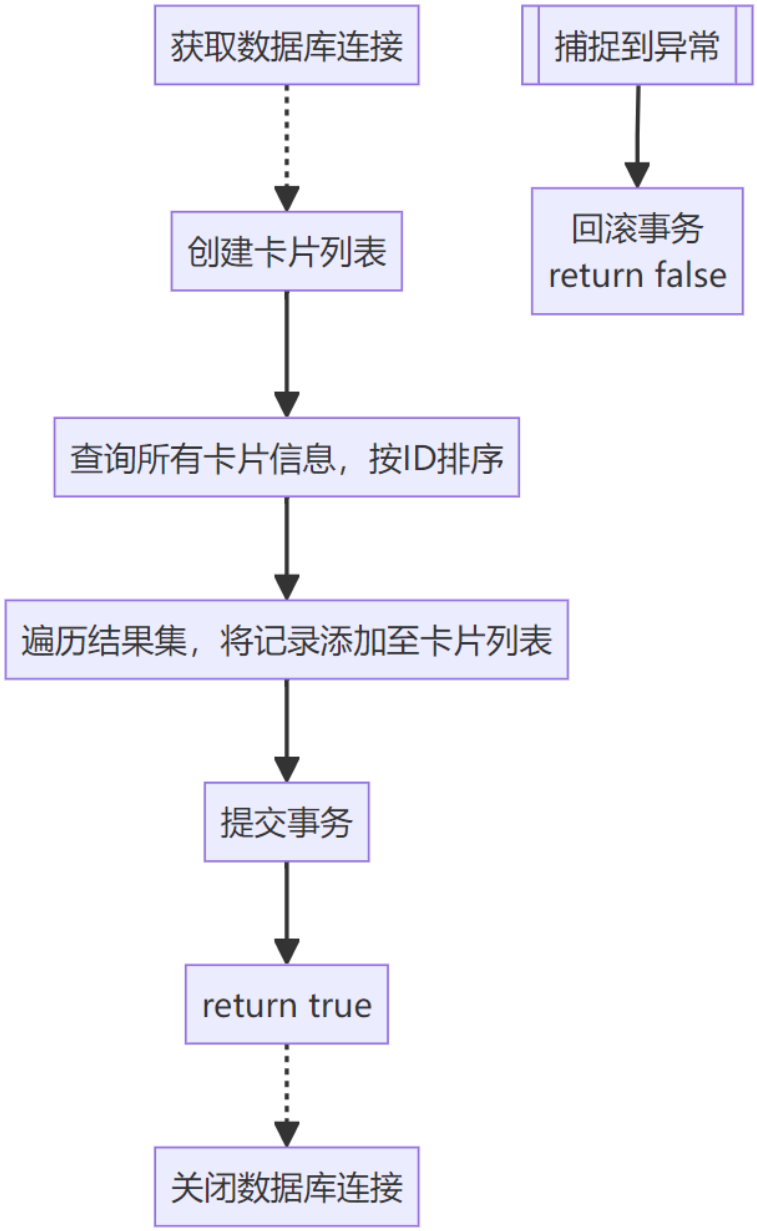
11. removeCard(int cardId)

1. 获取数据库连接
2. 查询该卡片是否有未归还的书籍
3. 如果有未归还的书籍，返回错误信息
4. 删除指定卡片信息
5. 检查是否成功删除卡片信息
6. 如果删除失败，回滚事务并返回错误信息
7. 提交事务
8. 返回成功信息
9. 异常处理：回滚事务并返回错误信息
10. 关闭数据库连接



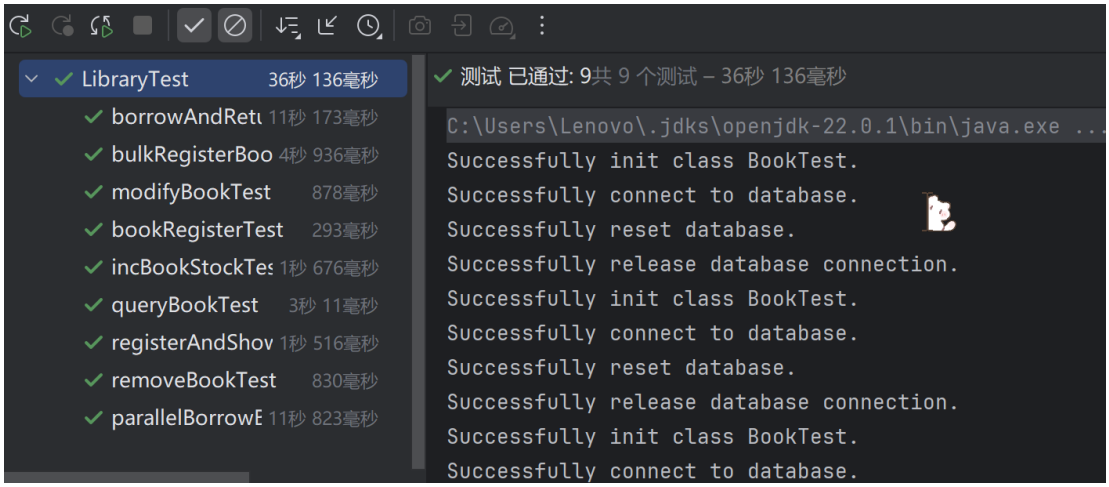
12. showCards()

1. 获取数据库连接
2. 创建卡片列表
3. 查询所有卡片信息并按卡片 ID 排序
4. 遍历结果集，将每条记录转换为卡片对象并添加到卡片列表中
5. 提交事务
6. 异常处理：回滚事务并返回错误信息
7. 返回成功信息和结果对象
8. 关闭数据库连接



5 系统验证测试

正确性测试：



功能性测试：

（仅实现了字符界面，没有实现图形化界面）

初始界面和退出界面：

```
---Welcome to the library---
Input the number to operate:
 [1] store book          [2] delta book stock        [3] batch store books
 [4] remove book         [5] modify book info        [6] query book
 [7] borrow book         [8] return book             [9] show borrow history
[10] register card      [11] remove card            [12] show cards
[13] reset database      [0] exit

Please enter your choice:
0
---EXIT---
```

输入不恰当选项时，系统给出相应反馈：

```
Please enter your choice:
aaa
Invalid input, please enter integer
Please enter your choice:
15
No such choice
Please enter your choice:
```

下面针对各选项进行功能性测试：

1.StoreBook(Book book)

图书入库：

```
1
Please enter the book information:
Category:test
Title:aaa
Press:bbb ccc ddd
PublishYear:2023
Author:aut
Price:12.3
Stock:2
Successfully store book, book_id:5
```

fine.library - dbo.book

book...	cate...	title	press	publi...	author	price	stock
1	aaa	bbb	ccc	2024	ddd	12.30	0
2	c1	t1	p1	2021	a1	193.80	1
3	c2	t2	p2	2008	a2	56.80	2
4	c3	t3	p3	1987	a3	87.30	7
5	test	aaa	bbb c...	2023	aut	12.30	2

再次尝试插入时，操作失败，显示图书已入库：

```
1
Please enter the book information:
Category:test
Title:aaa
Press:bbb ccc ddd
PublishYear:2023
Author:aut
Price:12.3
Stock:4
Book already exists
Please enter your choice:
```

只有当类别、书名、出版社、发行年份、作者均相等时才认为两本书相同，下面给出一个改变了类别（test 变为 testt）后入库成功的例子：

```
1
Please enter the book information:
Category:testt
Title:aaa
Press:bbb ccc ddd
PublishYear:2023
Author:aut
Price:12.3
Stock:1
Successfully store book, book_id:7
Please enter your choice:
```

2.incBookStock(int bookId, int deltaStock)

Case1 的截图中可以看到 book1 的库存为零，尝试调整库存减一后，操作失败。再次尝试，增加三后减少二，均成功（此时库存为一，符合结果预期）：

```
2
Please enter the book_id and the delta_number:
Book_id:1
Delta_number:-1
Stock can't be negative number
Please enter your choice:
2
Please enter the book_id and the delta_number:
Book_id:1
Delta_number:3
Successfully delta book stock
Please enter your choice:
2
Please enter the book_id and the delta_number:
Book_id:1
Delta_number:-2
Successfully delta book stock
Please enter your choice:
```

fine.library - dbo.book

book...	cate...	title	press	publi...	author	price	stock
1	aaa	bbb	ccc	2024	ddd	12.30	1
2	c1	t1	p1	2021	a1	193.80	1
3	c2	t2	p2	2008	a2	56.80	2
4	c3	t3	p3	1987	a3	87.30	7
5	test	aaa	bbb c...	2023	aut	12.30	2
6	testtt	aaa	bbb c...	2023	aut	12.30	2
7	testt	aaa	bbb c...	2023	aut	12.30	1

尝试给不存在的图书修改库存时，返回报错信息：

```
Please enter your choice:
2
Please enter the book_id and the delta_number
Book_id:11
Delta_number:2
Can not find the book
Please enter your choice:
```

3. storeBook(List<Book> books)

批量插入图书时，采用逗号作为分隔符，若输入不符合规范，则输出报错信息：

```
3
Please enter the book information:
Format(split by ','): category, title, press, publishYear, author, price, stock
ccc, ttt, p p p, 2021, aaa, 45.83, 3
cc2, tt2, pp2, 2003, aa2, 43.9, 2
cc3, tt3, pp3, 2009, aa3, 239.8, 1

Successfully batch store books
Please enter your choice:
3
Please enter the book information:
Format(split by ','): category, title, press, publishYear, author, price, stock
ccc ddd we
Invalid input
```

fine.library - dbo.book

	book...	cate...	title	press	publi...	author	price	stock
▶	1	aaa	bbb	ccc	2024	ddd	12.30	1
	2	c1	t1	p1	2021	a1	193.80	1
	3	c2	t2	p2	2008	a2	56.80	2
	4	c3	t3	p3	1987	a3	87.30	7
	5	test	aaa	bbb c...	2023	aut	12.30	2
	6	testtt	aaa	bbb c...	2023	aut	12.30	2
	7	testt	aaa	bbb c...	2023	aut	12.30	1
	8	ccc	ttt	p p p	2021	aaa	45.83	3
	9	cc2	tt2	pp2	2003	aa2	43.90	2
	10	cc3	tt3	pp3	2009	aa3	239.80	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

若批量添加的书中有已经存在的书，则返回报错，这一批图书均不会入库：

```
Please enter your choice:
3
Please enter the book information:
Format(split by ','): category, title, press, publishYear, author, price, stock
ccc1, t1, p1, 2021, a1, 193.8, 1
cccc2, t2, p2, 2008, a2, 56.8, 2
c3, t3, p3, 1987, a3, 87.3, 7

Some of the books already exist
Please enter your choice:
```

4. removeBook(int bookId)

移除不存在或是被借阅仍未归还的图书，系统返回报错，否则成功移除该图书：

```
4
Please enter the book_id:
Book_id:21
Can not find the book
Please enter your choice:
4
Please enter the book_id:
Book_id:1
Book is borrowed
Please enter your choice:
4
Please enter the book_id:
Book_id:3
Successfully remove the book
Please enter your choice:
```

fine.library - dbo.book

	book...	cate...	title	press	publi...	author	price	stock
▶	1	aaa	bbb	ccc	2024	ddd	12.30	1
	2	c1	t1	p1	2021	a1	193.80	1
	4	c3	t3	p3	1987	a3	87.30	7
	5	test	aaa	bbb c...	2023	aut	12.30	2
	6	testtt	aaa	bbb c...	2023	aut	12.30	2
	7	testt	aaa	bbb c...	2023	aut	12.30	1
	8	ccc	ttt	p p p	2021	aaa	45.83	3
	9	cc2	tt2	pp2	2003	aa2	43.90	2
	10	cc3	tt3	pp3	2009	aa3	239.80	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

5. modifyBookInfo(Book book)

在修改图书信息模块，所有可被修改的项目都会被列出，如果不需要修改，可用回车跳过该项。

尝试修改不存在的图书，系统返回报错信息：

```
5
Please enter the book_id of book to be modified:
Book_id:32
Please enter the book information to modify:
Tip: input [Enter] to skip the modification of one item
Category:
Title:
Press:
PublishYear:
Author:
Price:
Can not find the book
```

下面分别给出修改 String 类型变量和 Int/Double 类型变量的例子：

```
5
Please enter the book_id of book to be modified:
Book_id:5
Please enter the book information to modify:
Tip: input [Enter] to skip the modification of one item
Category:modify
Title:
Press:
PublishYear:
Author:
Price:
Successfully modify book
Please enter your choice:
5
Please enter the book_id of book to be modified:
Book_id:6
Please enter the book information to modify:
Tip: input [Enter] to skip the modification of one item
Category:
Title:
Press:
PublishYear:2024
Author:
Price:999.0
Successfully modify book
Please enter your choice:
```

fine.library - dbo.book

book...	cate...	title	press	publi...	author	price	stock
1	aaa	bbb	ccc	2024	ddd	12.30	1
2	c1	t1	p1	2021	a1	193.80	1
4	c3	t3	p3	1987	a3	87.30	7
5	modify	aaa	bbb c...	2023	aut	12.30	2
6	testtt	aaa	bbb c...	2024	aut	999.00	2
7	testt	aaa	bbb c...	2023	aut	12.30	1
8	ccc	ttt	p p p	2021	aaa	45.83	3
9	cc2	tt2	pp2	2003	aa2	43.90	2
10	cc3	tt3	pp3	2009	aa3	239.80	1

6. queryBook(BookQueryConditions conditions)

给出条件后，系统查询满足条件的图书，操作成功后返回结果数量并将图书按 Id 升序打印（同样的，不需要查询的条件，使用回车跳过）：

```
6
Please enter the conditions to query books:
Tips: input [Enter] to skip the current condition
Category:
Title:
Press:
MinPublishYear:2000
MaxPublishYear:2023
author:
MinPrice:
MaxPrice:
Successfully query books
6 records found:
No 1: Book {bookId=2, category='c1', title='t1', press='p1', publishYear=2021, author='a1', price=193.80, stock=1}
No 2: Book {bookId=5, category='modify', title='aaa', press='bbb ccc ddd', publishYear=2023, author='aut', price=12.30, stock=2}
No 3: Book {bookId=7, category='testt', title='aaa', press='bbb ccc ddd', publishYear=2023, author='aut', price=12.30, stock=1}
No 4: Book {bookId=8, category='ccc', title='ttt', press='p p p', publishYear=2021, author='aaa', price=45.83, stock=3}
No 5: Book {bookId=9, category='cc2', title='tt2', press='pp2', publishYear=2003, author='aa2', price=43.90, stock=2}
No 6: Book {bookId=10, category='cc3', title='tt3', press='pp3', publishYear=2009, author='aa3', price=239.80, stock=1}
Please enter your choice:
```

```
6
Please enter the conditions to query books:
Tips: input [Enter] to skip the current condition
Category:
Title:
Press:
MinPublishYear:
MaxPublishYear:
author:
MinPrice:200
MaxPrice:
Successfully query books
2 records found:
No 1: Book {bookId=6, category='testtt', title='aaa', press='bbb ccc ddd', publishYear=2024, author='aut', price=999.00, stock=2}
No 2: Book {bookId=10, category='cc3', title='tt3', press='pp3', publishYear=2009, author='aa3', price=239.80, stock=1}
```

```
6
Please enter the conditions to query books:
Tips: input [Enter] to skip the current condition
Category:
Title:
Press:bbb ccc ddd
MinPublishYear:
MaxPublishYear:
author:
MinPrice:
MaxPrice:
Successfully query books
3 records found:
No 1: Book {bookId=5, category='modify', title='aaa', press='bbb ccc ddd', publishYear=2023, author='aut', price=12.30, stock=2}
No 2: Book {bookId=6, category='testtt', title='aaa', press='bbb ccc ddd', publishYear=2024, author='aut', price=999.00, stock=2}
No 3: Book {bookId=7, category='testt', title='aaa', press='bbb ccc ddd', publishYear=2023, author='aut', price=12.30, stock=1}
Please enter your choice:
```

7. borrowBook(Borrow borrow)

对于不存在或是没有余量导致无法被借阅的图书，系统分别返回相应报错：

```
Please enter your choice:
7
Please enter the borrow information:
Card_id:1
Book_id:39
Can not find the book
Please enter your choice:
7
Please enter the borrow information:
Card_id:1
Book_id:2
No stock
```

对于已经被同一张图书卡借阅的图书，系统同样返回报错信息：

```
7
Please enter the borrow information:
Card_id:1
Book_id:1
You had borrowed this book
Please enter your choice:
```

对于已入库、有库存且未被同一图书卡借阅的图书，系统允许借阅并调整库存（借阅状态下归还时间置 0）：

```
7
Please enter the borrow information:
Card_id:1
Book_id:4
Successfully borrow the book
Please enter your choice:
```

fine.library - dbo.borrow | fine.library - dbo.book

	card_...	book...	borrow_time	retur...
▶	1	1	1714332230528	0
	1	4	1714334864358	0
*	NULL	NULL	NULL	NULL

8. returnBook(Borrow borrow)

不存在对应图书 ID 和借书证 ID 的借阅记录时，系统返回报错信息：

```
8
Please enter the return information:
Card_id:1
Book_id:2
No borrow record
Please enter your choice:
8
Please enter the return information:
Card_id:2
Book_id:1
No borrow record
Please enter your choice:
```

存在对应借阅记录时，系统成功执行归还操作，修改记录中的归还时间，并更新库存：

```
8
Please enter the return information:
Card_id:1
Book_id:1
Successfully return the book
Please enter your choice:
```

fine.library - dbo.borrow | fine.library - dbo.book

	card_...	book...	borrow_time	return_time
▶	1	1	1714332230528	1714335290139
	1	4	1714334864358	0
*	NULL	NULL	NULL	NULL

fine.library - dbo.borrow | fine.library - dbo.book

	book...	cate...	title	press	publi...	author	price	stock
	1	aaa	bbb	ccc	2024	ddd	12.30	1
	2	c1	t1	p1	2021	a1	193.80	1
	4	c3	t3	p3	1987	a3	87.30	7

前

fine.library - dbo.book | fine.library - dbo.borrow

	book...	cate...	title	press	publi...	author	price	stock
▶	1	aaa	bbb	ccc	2024	ddd	12.30	2

后

9. showBorrowHistory(int cardId)

输入卡号后系统打印对应的借阅记录（按照借阅时间递减的顺序）：


```
9
Please enter the card_id to search the borrow history:
Card_id:2
Successfully search the borrow history:
Card 2: 0 records in total:
```

10. registerCard(Card card)

对于符合规范的输入，系统成功注册借书卡，若同一人再次进行注册，则返回报错：

```
10
Please enter the information of the card to register:
Name:Candy
Department:math
Type(S/T):S
Successfully register the card
Please enter your choice:
10
Please enter the information of the card to register:
Name:Candy
Department:math
Type(S/T):S
Card already exists
```

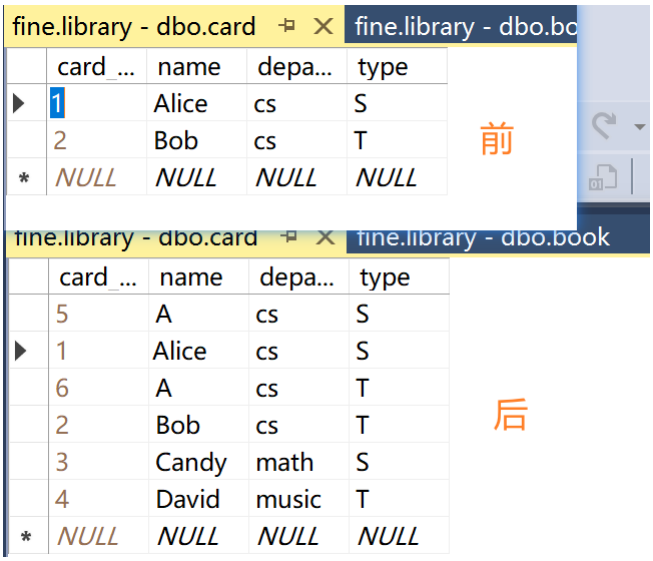
输入的身份状态仅为 S 或 T，否则系统返回报错：

```
10
Please enter the information of the card to register:
Name:David
Department:music
Type(S/T):T
Successfully register the card
Please enter your choice:
10
Please enter the information of the card to register:
Name:Elsa
Department:magic
Type(S/T):U
Unimplemented Function
```

只有当姓名、单位、身份全部相等时，才视作同一人：

```
10
Please enter the information of the card to register:
Name:A
Department:cs
Type(S/T):S
Successfully register the card
Please enter your choice:
10
Please enter the information of the card to register:
Name:A
Department:cs
Type(S/T):T
Successfully register the card
```

本模块内操作前后数据表 card 变化如下：



fine.library - dbo.card

	card_...	name	depa...	type
▶	1	Alice	cs	S
	2	Bob	cs	T
*	NULL	NULL	NULL	NULL

前

fine.library - dbo.card　fine.library - dbo.book

	card_...	name	depa...	type
	5	A	cs	S
▶	1	Alice	cs	S
	6	A	cs	T
	2	Bob	cs	T
	3	Candy	math	S
	4	David	music	T
*	NULL	NULL	NULL	NULL

后

11. removeCard(int cardId)

对于不存在或有未归还图书的卡号，系统均会给出相应报错信息，否则，系统成功注销该借书卡：

```
11
Please enter the card_id to remove:
Card_id:20
The card to be deleted doesn't exist
Please enter your choice:
11
Please enter the card_id to remove:
Card_id:1
There are un-returned books
Please enter your choice:
11
Please enter the card_id to remove:
Card_id:4
Successfully remove the card
Please enter your choice:
```

Card4 被注销：

fine.library - dbo.card | fine.library - dbo.book

	card_...	name	depa...	type
▶	5	A	cs	S
	1	Alice	cs	S
	6	A	cs	T
	2	Bob	cs	T
	3	Candy	math	S
*	NULL	NULL	NULL	NULL

12. showCards()

按卡号递增顺序打印借书卡信息：

```
12
Successfully show the card
5 cards in total:
No 1: Card {cardId=1, name='Alice', department='cs', type=Student}
No 2: Card {cardId=2, name='Bob', department='cs', type=Teacher}
No 3: Card {cardId=3, name='Candy', department='math', type=Student}
No 4: Card {cardId=5, name='A', department='cs', type=Student}
No 5: Card {cardId=6, name='A', department='cs', type=Teacher}
```

13.resetDatabase()

执行前 library 的 book 表如下：

fine.library - dbo.book

book...	cate...	title	press	publi...	author	price	stock
1	Others	The ...	Press...	2011	DaDa	33.00	9
2	Others	Oper...	Press...	2014	Yuuku	39.62	31
3	Horror	Euge...	Press-E	2012	ZaiZai	195.62	0

成功执行[13]后，整个数据库被重置：

```
Please enter your choice:
13
Successfully reset database
```

fine.library - dbo.book

book...	cate...	title	press	publi...	author	price	stock
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

6 遇到的问题和解决方法

Implement：

1.嵌套类的调用

有几个类型中嵌套了类的定义，在调用和更改时需要格外注意使用格式。如借书卡类型、查询条件的升序与降序等：

```
card.setName(res.getString( columnLabel: "name"));
card.setDepartment(res.getString( columnLabel: "department"));
card.setType(Card.CardType.values(res.getString( columnLabel: "type")));  // card type !!!
cards.add(card);
```

```
if (conditions.getSortOrder() == SortOrder.ASC) // asc !!!
    books.sort(conditions.getSortBy().getComparator());
else
    books.sort(conditions.getSortBy().getComparator().reversed());  // desc
commit(conn);
```

2.事务隔离级别串行化

在修改为串行化之前，有出现过报错信息，修改后虽然效率相对低了一点，但是可以避免并发问题：

```java
@Override  14 个用法
public ApiResult borrowBook(Borrow borrow) {
    Connection conn = connector.getConn();
    try { // !!!
        conn.setTransactionIsolation(Connection.TRANSACTION_SERIALIZABLE);
    } catch (Exception e) {
        return new ApiResult( ok: false,  message: "Unimplemented Function");
    }
    try {
```

3.借书时间

测试中途有出现借阅图书的测试点过不去的情况：

排查原因后发现是因为输入的数据只有图书 ID 和借书证 ID，不包含借阅时间。但是后续会使用到借阅时间，因此需要手动添加。

```java
try {
    String query_sql = "SELECT * FROM borrow WHERE book_id = ? AND card_id = ? AND return_time = 0";
    PreparedStatement query_stmt = conn.prepareStatement(query_sql);
    query_stmt.setInt( parameterIndex: 1, borrow.getBookId());
    query_stmt.setInt( parameterIndex: 2, borrow.getCardId());
    ResultSet res = query_stmt.executeQuery();
    if (!res.next())
        return new ApiResult( ok: false,  message: "No borrow record");
    else
        borrow.setBorrowTime(res.getLong( columnLabel: "borrow_time")); // do not be given the borrow time
    if (borrow.getReturnTime() <= borrow.getBorrowTime())
        return new ApiResult( ok: false,  message: "Invalid return time");
```

Main：

1.换行符的消耗

输入数字来选择操作时，发现输入数字并敲击回车后仍卡在输入界面，没有进行下一个环节。

仔细检查代码发现是 scanner.nextInt()只读取了输入的数字，不会读取换行符，因此该换行符一直未被读取，就卡在输入界面了。消耗换行符后，程序可以继续运行：

```java
public static void main(String[] args) {
        System.out.println("Please enter your choice:");
        Scanner scanner = new Scanner(System.in);
        choice = scanner.nextInt();
        scanner.nextLine(); // !!!
        switch(choice){
```

2.空字符串判断

修改图书信息时，直接用回车跳过某项的修改，发现被跳过的项目信息被 null 代替了：

fine.library - dbo.book

	book...	cate...	title	press	publi...	author	price	stock
▶	1	aaa	bbb	ccc	2024	ddd	12.30	3
	2	aaab	bbb	ccc	2001	fff	23.80	4
	3				2001		3.00	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

排查问题后发现是 implement 中对应函数部分没有检查字符串是否为空，直接将读入的字符串替换了原有信息。添加判断语句后解决：

```
ResultSet res = query_stmt.executeQuery();
if (!res.next())
    return new ApiResult( ok: false,  message: "Can not find the book");
String modify_sql = "UPDATE book SET category = ?, title = ?, press = ?, publish_y
PreparedStatement modify_stmt = conn.prepareStatement(modify_sql);
if (!book.getCategory().isEmpty())
    modify_stmt.setString( parameterIndex: 1, book.getCategory());
else
    modify_stmt.setString( parameterIndex: 1, res.getString( columnLabel: "category"));
```

后续测试如下，结果符合预期：

```
Please enter your choice:
5
Please enter the book_id of book to be modified:
Book_id:20
Please enter the book information to modify:
Tip: input [Enter] to skip the modification of one item
Category:a
Title:
Press:
PublishYear:
Author:
Price:
Successfully modify book
```

```
5
Please enter the book_id of book to be modified:
Book_id:21
Please enter the book information to modify:
Tip: input [Enter] to skip the modification of one item
Category:
Title:
Press:2024
PublishYear:
Author:
Price:999.0
Successfully modify book
```

19	History	Algor...	Press...	2022	Soon...	57.97	19
20	a	Analy...	Press-E	2002	None...	166.51	9
21	Philo...	Oper...	2024	2010	DouD...	999.00	31
22	Nature	Com...	Press...	2010	Cola...	210.23	83

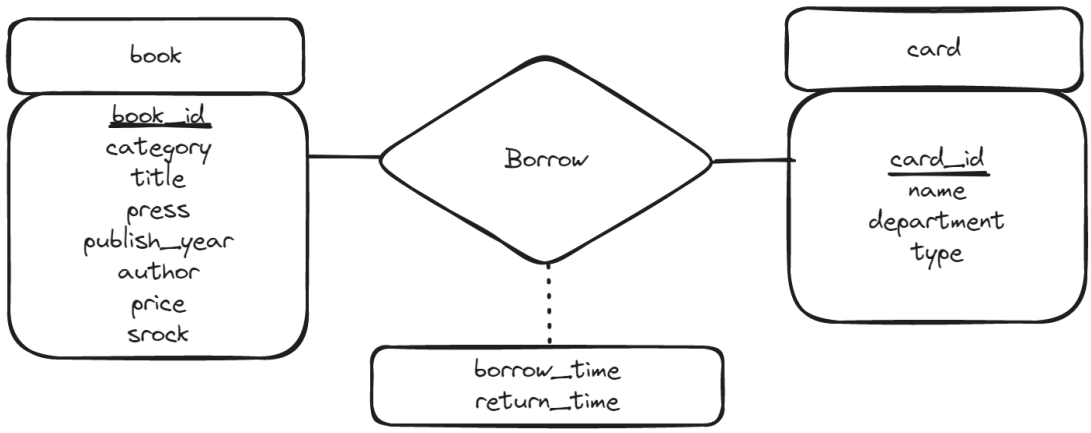
3.跳过 Int/Double

在问题 1-换行符的消耗中我们已经了解到，nextInt()不会消耗换行符，而 nextLine()可以消耗换行符。要实现可选择地跳过一个 Int/Double 输入的功能，就得满足“输入数字，则数字可以被读取到，且后面的换行符也可以被消耗；否则就只消耗换行符”的条件。因此需要判断读入的行是否为空，并对数据进行相应赋值（数据不为空时，需要用系统函数将读取到的字符串转化为 Int/Double 型变量）：

```
System.out.print("PublishYear:");
nextline = scanner.nextLine();
if (nextline.equals("")) {
    publishYear = -1;
      System.out.println("ohhh"); // test
}
else
    publishYear = Integer.parseInt(nextline);
```

7 思考题解答

1. 绘制该图书管理系统的 E-R 图



2. 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？

攻击者利用应用程序对用户输入数据的处理不当，通过在输入中注入恶意的 SQL 代码来实现对数据库的非法访问或控制，达到 SQL 注入攻击的目的。

原理：

- 攻击者在应用程序的输入框或参数中输入恶意的 SQL 代码。
- 应用程序未对用户输入进行充分验证和过滤，直接将用户输入拼接到 SQL 查询语句中。
- 恶意代码被执行，攻击者可以通过该代码来执行数据库操作，获取数据或控制数据库。

举例：

攻击者可以输入代码’OR 1 = 1’，应用程序如果未加检查便将其拼接到 SQL 查询语句中，会导致该语句后跟的条件作废，攻击者很可能绕过了某些条件来获得自身未得到授权的数据或授权。

解决：

- 使用参数化查询或预编译语句：将用户输入的数据作为参数传递给 SQL 查询，而不是直接拼接到 SQL

语句中，从而避免 SQL 注入攻击。

- 输入验证和过滤：确保输入数据符合预期格式和范围，过滤掉恶意代码。

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

- 行级锁：类似于 cpu 流水线，InnoDB 引擎使用行级锁来保护数据，当一个事务在读取或修改某一行数据时，会对该行数据加上行级锁，其他事务需要等待该行级锁释放后才能访问该行数据。
- 快照读：在 RR 隔离级别下，读取数据时使用的是快照读的方式，即每个事务在启动时会创建一个事务开始的快照，事务中所有的读操作都是基于这个快照的数据版本，而不受其他事务的修改影响。