浙江大学

数据库系统实验报告

作业名称：数据库程序设计

姓　　名：

学　　号：

电子邮箱：

联系电话：

指导老师：

2020 年 4 月 25 日

实验名称

一、 实验目的

1. 掌握数据库应用开发程序设计方法。

二、 系统需求

这部分我针对实验指导中做了一些小修改，考虑实际情况加了点自己的理解。

比如借书证增加了一个属性是“已借书的数量”，上限为 10，防止一个借书证借太多的书

再比如，借书记录里面多了一条属性“是否可以续借”。因为现实中经常有这种需求。我在程序中规定，借书时限为 30 天，允许续借一次延长 30 天。

1. 基本数据对象

对象名称	属性
书	书号，书名，类别，出版社，作者，年份，价格，总藏书量，库存
借书证	借书证号，姓名，单位（学院），类别（教师、学生），已借书的数量
借书记录	书号，借书证号，借书时间，归还期限，是否可以续借

2. 基本功能模块

模块名称	功能描述
图书入库（修改）	1. 单本入库 在单本入库的模式下同时支持修改图书的信息 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)
借书证管理	增加或删除一个借书证，同时支持修改已有的借书证（例如修改所在单位/学院）
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

3.用户界面

图形界面，采用 python 的原生 tkinter 及 ttk 构建

4.数据库

在 MySQL 中建立 library 数据库,并且按照第四部分中的数据库逻辑结构建立相应的 table。登录数据库的账号密码默认为我个人的账号密码（root,0309miku0831）。可以在 config.txt 中改成运行程序的电脑上的账号和密码。

三、 实验环境

1.数据库管理系统：MySQL

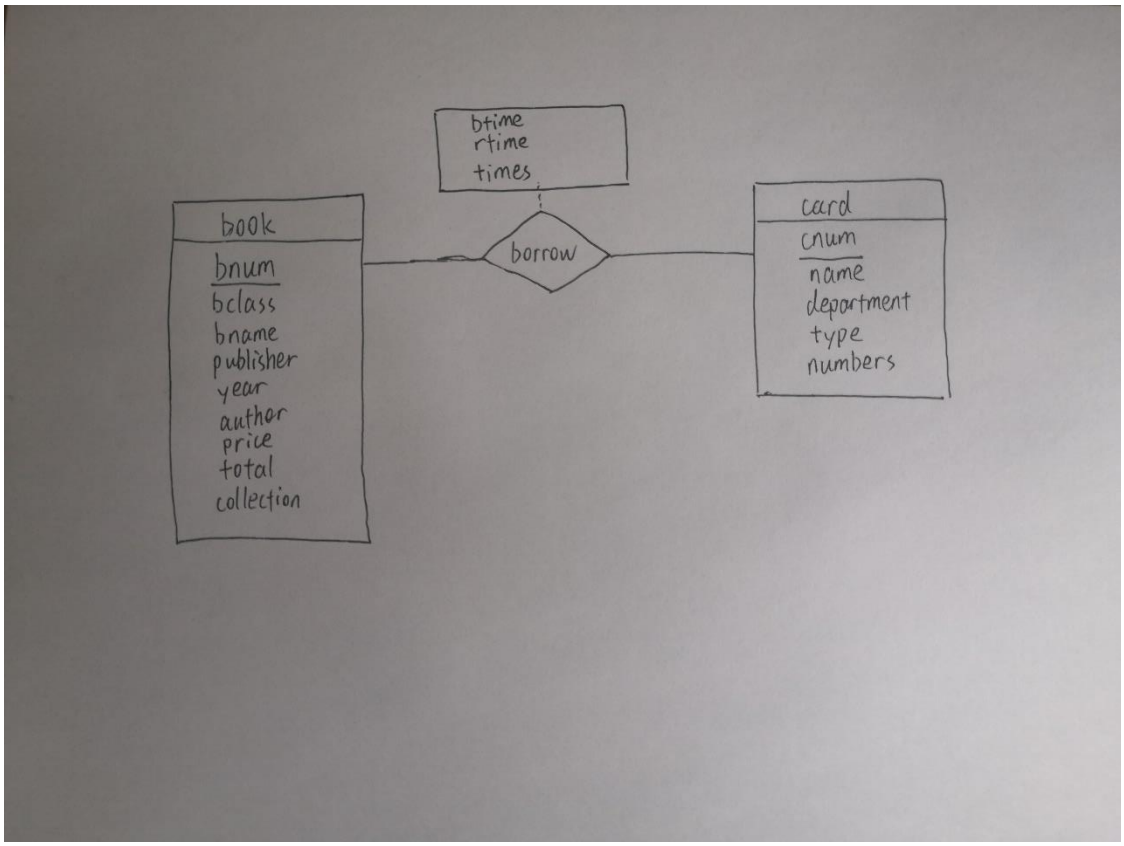
2.开发语言：python（通过 pymysql 于 mysql 连接）

3.图形界面：tkinter

4.开发工具：vscode+anaconda

四、 系统设计及实现

1. 实体之间的关系 E-R 图



2. 数据库逻辑结构设计

Book:

```
create table book
(bnum char(20),
bclass varchar(50),
bname varchar(50),
publisher varchar(50),
year int,
author varchar(20),
price decimal(7,2),
total int,
collection int,
```

```
primary key(bnum)
);
```

分别对应前文基本数据对象描述中，书的各个属性。其中 total 代表图书馆一共有藏书多少本，collection 代表图书馆目前所剩的这本书的数量

Card:

```
create table card
(cnum char(7), -- 注意卡号长度为 7
name varchar(20),
department varchar(40),
type char(1),
numbers int,
primary key(cnum),
check(type in ('T','S'))
);
```

分别于前文中提到的借书证的属性相对应，分别代表借书证号，持证者姓名，单位（学院），类型，以及当前借书数量（限制为 10 本）

Borrow:

```
create table borrow
(cnum char(7),
bnum char(20),
btime datetime,
rtime datetime,
times int, -- 用于记录可续借的次数
primary key(cnum,bnum),
foreign key(cnum) references card(cnum),
foreign key(bnum) references book(bnum)
);
```

对应前文中提到的借书记录的各个属性，与实验指导中给出的示例不同，我将借书时间和归还期限统统设成了 datetime 类型，因为这样更加贴近现实中的情况，也可以很方便的和 python 交互，可以直接调用 python 中的 datetime 模块然后稍加转换就可以直接

写进 sql 语句当中，并且 datetime 类型也支持聚合函数中的选取最小值，对于后续的操作也很方便。

3. 程序运行结果场景以及截图说明

首先是初始界面。注意在登录前请确保 mysql 中有 library 这个数据库，和相应的 table。并且修改 config.txt 中的账号和密码为本机 MySQL 登录用的账号和密码。

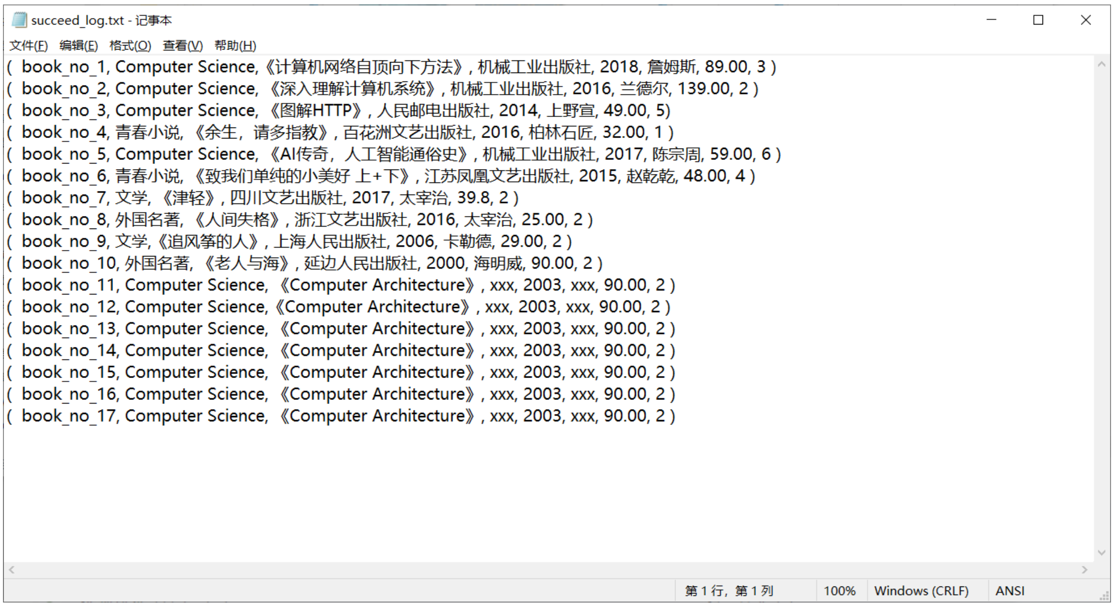
a. 图书入库和修改：

首先我们来看批量入库功能：（批量入库是通过反复调用单本入库实现的，因此对于 sql 语句的介绍放在单本入库里面一起讲）

我们输入文件名字后，点击确认，右侧状态栏会显示导入的结果。其中，成功导入的书会被记录在 succeed_log.txt 中。而考虑到我们往往更加关心有哪些书被导入失败了，因此我们在下面会用一个列表来展示导入失败的书的名单（这里是书全部导入成功了），下面我们来看一个导入失败的例子

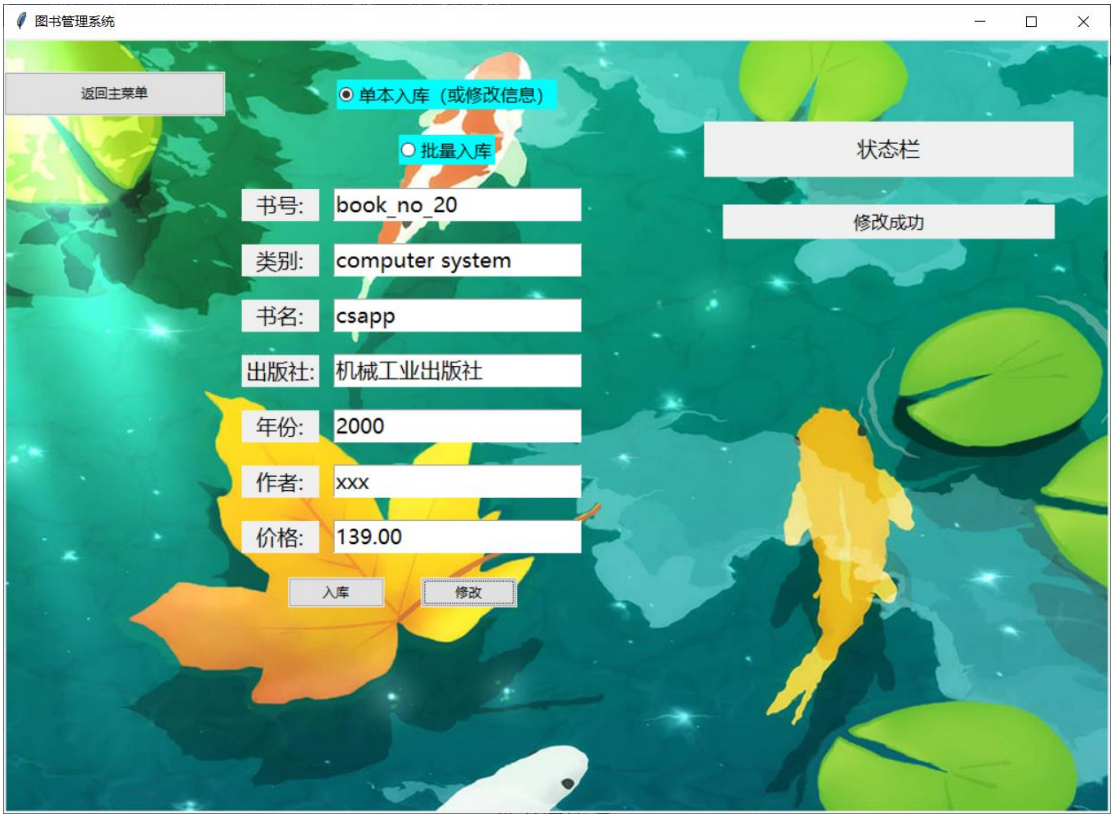
我们在 data2.txt 中放了 7 条与原来不一致的数据（书号为主键，但是其他信息与原来不一致，因此导入失败）

下面是导入记录日志（succeed_log.txt）(原谅我手边真的书不多，又懒得找了，所以后面基本是凑数的)

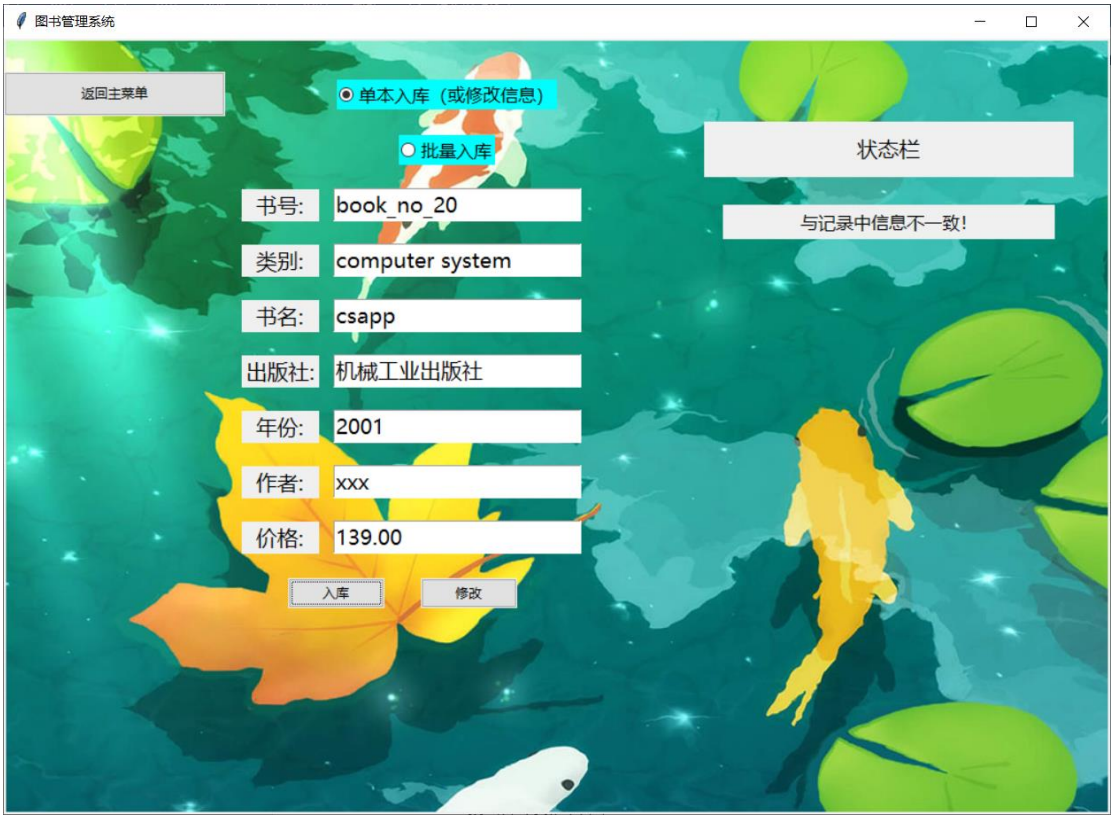


接下来我们看单本入库（或修改）：

我们输入完信息后点入库，即可添加一本书进去。而如果我们要修改这本书的信息，比如把他的类别改成 computer system，



则在输入完信息后点击修改即可（因为实验指导中对于修改信息这一点描述得比较模糊，所以我特别增加了一个选项，让操作者可以明确选择，明确是修改原书信息还是单纯输入有误，以防误操作而把书的信息改错）。如果我此时输入的信息与数据库中记录的不一致，然后直接点击“入库”而不是“修改”，则会提示错误：（我将年份改成 2001 年，原记录是 2000 年）



下面是构建 sql 语句的部分。首先是最普通的插入部分

```
sql="insert into book values(%s,%s,%s,%s,%s,%s,%s,%s,%s)"
cursor.execute(sql,(BookNum,ClassOfBook,BookName,Publisher,Year,Author,
Price,total,collection))
```

这其中就用到了 pymysql 的参数化执行。执行的时候，参数列表中的特殊符号会被自动转义，也就避免了 sql 注入的发生。而如果是入库一本已经存在的书，则要这样：

```
 sql="update book set total=%s,collection=%s where bnum=%s"
                    cursor.execute(sql ,(total,collection,BookNum))
```

接下来是修改部分：

```
sql="update book set bclass=%s,bname=%s,publisher=%s,year=%s,author=%s,
price=%s where bnum=%s"
cursor.execute(sql ,(ClassOfBook,BookName,Publisher,Year,Author,Price,B
ookNum))
```

其中这两条语句是在事务内执行的。如果后面发现所做的动作是“入库”而并非“修改”，事务会被 rollback。

总的来说这一部分可能是最复杂的，因为情况比较多。首先要判断图书馆有没有这本书，再看所做的动作是“入库”还是“修改”。然后再比对填入的书本信息和原记录中的信息，如果不一致则更新书的信息或者直接报错。

不过幸亏 python 是动态语言，非常的灵活，我可以直接在不同的情况下 return 不同的类型，这样就可以很容易的判断执行的结果到底如何，是成功还是失败。

b. 借书证管理

借书证管理界面是这样的，支持曾涛或修改，和删除两种操作

其中添加或修改的部分和图书单本入库一样，如果输入的是新的借书证号，并且点击增添，则会新增一个借书证。而如果输入的是一个已经存在的借书证号，并且执行修改操作，则会修改原借书证的信息。其他情况下则会报错。这个修改功能是考虑到有些学生可能会从一个学院转到另一个学院，或者是老师的职位调动。

执行效果如下：（添加*3，删除*1，修改*1）


图书管理系统
返回主菜单
增添或修改
删除借书证
状态栏
借书证号:
cnum_1
添加成功
姓名:
孟庆昊
所属单位(学院):
计算机科学与技术学院
身份类别:
老师
学生
增添
修改


图书管理系统
返回主菜单
增添或修改
删除借书证
状态栏
借书证号:
cnum_2
添加成功
姓名:
孙建伶
所属单位(学院):
计算机科学与技术学院
身份类别:
老师
学生
增添
修改


图书管理系统
返回主菜单
增添或修改
删除借书证
状态栏
借书证号:
cnum_3
添加成功
姓名:
test
所属单位(学院):
test
身份类别:
老师
学生
增添
修改


图书管理系统
返回主菜单
增添或修改
删除借书证
状态栏
删除成功
请输入借书证号:
cnum_3
删除

这个模块比之前图书入库的模块要简单一些，因为不涉及统计数量之类的。之前在统计书的数量的时候，在一开始一本书没有的时候，select 得到的结果类型不是 int，因此不能直接对结果+1。而借书证管理这一部分，则考虑的东西少很多。关于构造 sql 语句的部分如下：

添加部分

```
sql="insert into card values(%s,%s,%s,%s,0)"
cursor.execute(sql,(CardNum,Name,Dept,Type))
```

其中插入的最后一个属性是固定的，0，表示已经借走的书的数量，上限为 10。而新增加的借书证一定是没有借过别的书的，所以这一项固定为 0。

修改部分

```
sql="update card set name=%s,department=%s,type=%s where cnum=%s"
cursor.execute(sql,(Name,Dept,Type,CardNum))
```

删除部分

```
sql="delete from card where cnum=%s"
cursor.execute(sql,CardNum)
```

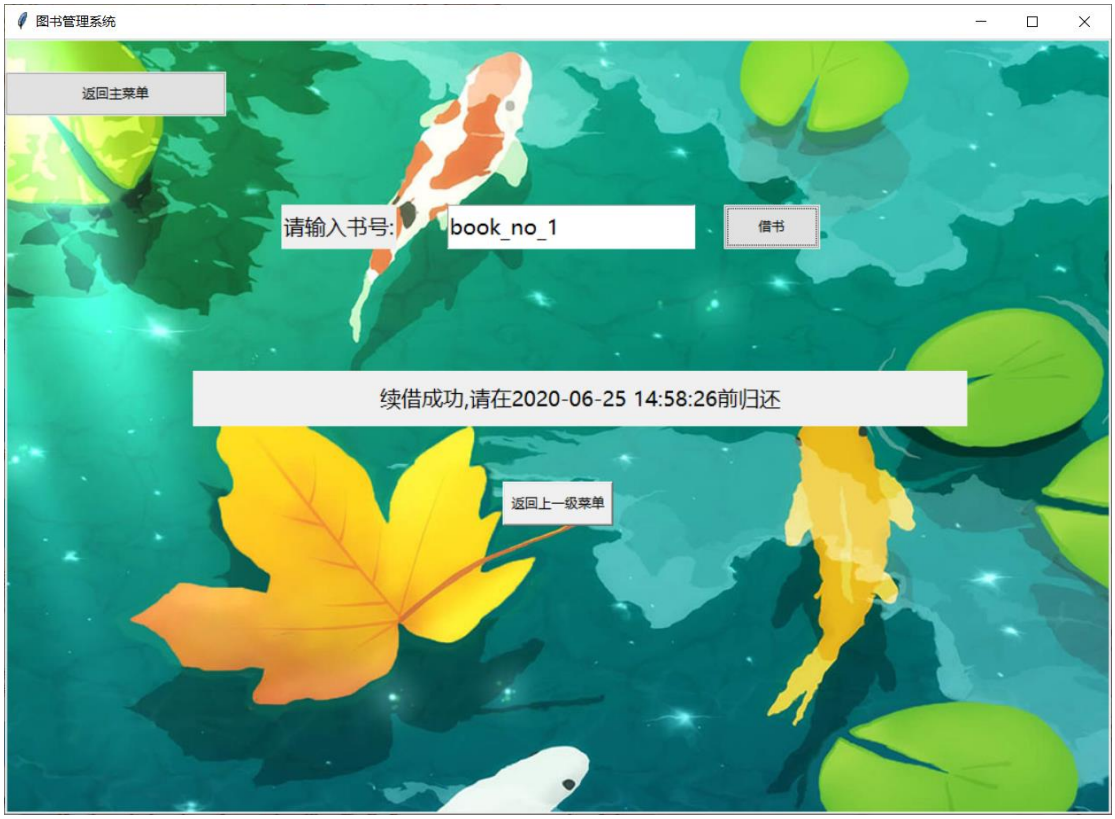
c. 借书：

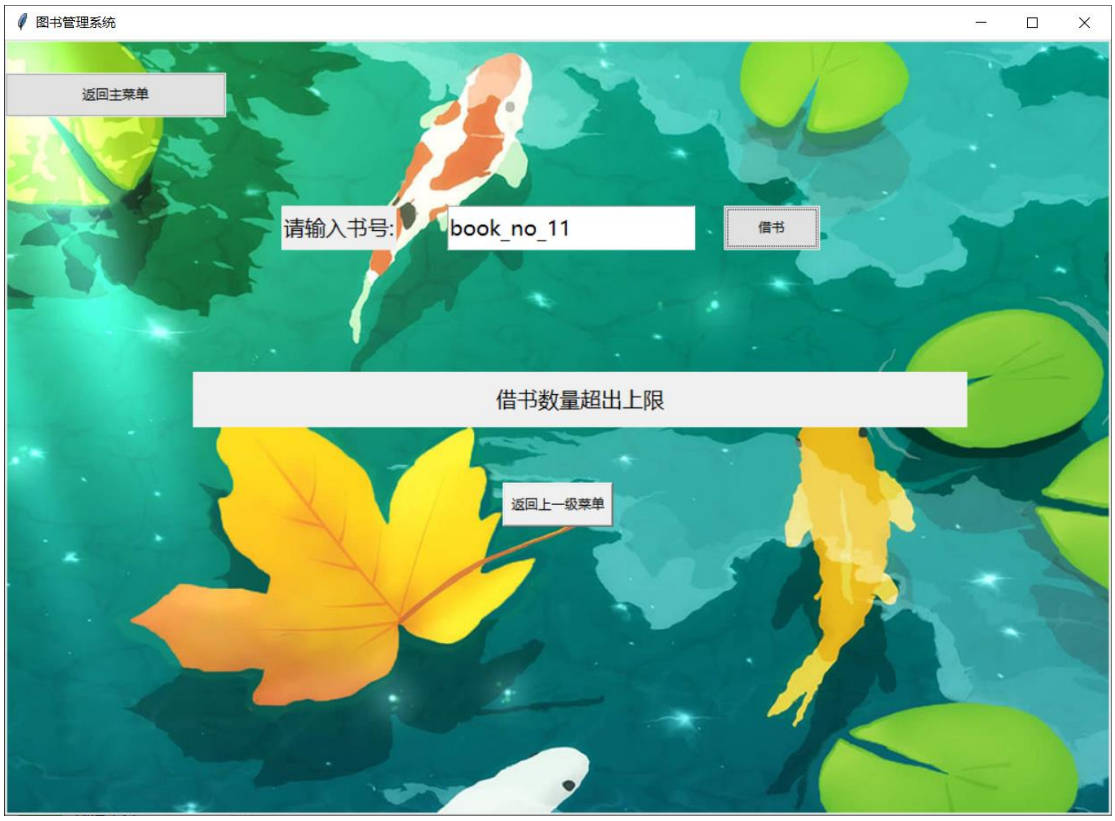
借书部分功能介绍：首先输入借书证号，然后会展示出这个借书证已经借走并且未归还的书。然后再输入要借的书号，如果有库存则借书成功，没有库存则会返回最早归还的时间。特别地，允许续借一次。一次借期为 30 天，续借一次可以再延长 30 天。

我们先借几本书，然后输入卡号：

可以看到我们借书的信息。在 table 中，我们有一个属性 times，就代表是次数的意思，即当前还可以再借几次这本书（1 次或 0 次），1 次就代表可以续借，0 次代表不能续借。

并且这个列表正好能容纳十条借书记录，如果列表被装满了，就代表这个借书证已经不能再借书了。下面我们再放几张借书的截图（借新书*1，续借*1，借书的数量超限，借书失败*1，借的书不存在*1，借的书没有库存*1）：


图书管理系统
返回主菜单
请输入书号:
book_no_1
借书
续借成功,请在2020-06-25 14:58:26前归还
返回上一级菜单


图书管理系统
返回主菜单
请输入书号:
book_no_11
借书
借书数量超出上限
返回上一级菜单

（其中最后一张图是用 cnum_2 这张卡借的，因为 cnum_1 已经借过了，再借的话不会显示无库存，会显示续借）

这一部分也不复杂，只是要实现两个功能：1.查询这个借书证号已经借走的书的清单 2.借书。

构造 sql 语句的部分如下：

```
sql="select bnum,bname,btime,rtime,times from book natural join borrow
where cnum=%s"
cursor.execute(sql,CardNum)
```

这样即可很简单得到借书的清单。

然后在借书的时候，如果是借新书：

```
sql="select bname,collection from book where bnum=%s"
cursor.execute(sql,BookNum)
```

如果书号是存在的，则借书成功，并且要在一个事务内执行下面的语句：

```
sql="update book set collection=%s where bnum=%s"
cursor.execute(sql,(collection,BookNum))
b_time= datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
r_time=(datetime.datetime.now()+datetime.timedelta(days=30)).strftime("
%Y-%m-%d %H:%M:%S")
sql="insert into borrow values(%s,%s,%s,%s,1)"
cursor.execute(sql,(CardNum,BookNum,b_time,r_time))
sql="update card set numbers=%s where cnum=%s"
cursor.execute(sql,(numbers,CardNum))
```

这些语句的作用就是，写入借书时间和归还时间，并且更新当前借书证已借书的数目，同时再把书的库存减一。以上这些统统是在一个事务内完成的。如果出错，则会直接到 except 部分，会把事务回滚。

d. 还书：

下面是还书部分，具体流程和借书差不多，都实现查看借书清单，然后再输入要还的书的书号。其中查看清单调用的是借书模块中的函数。

运行截图如下：

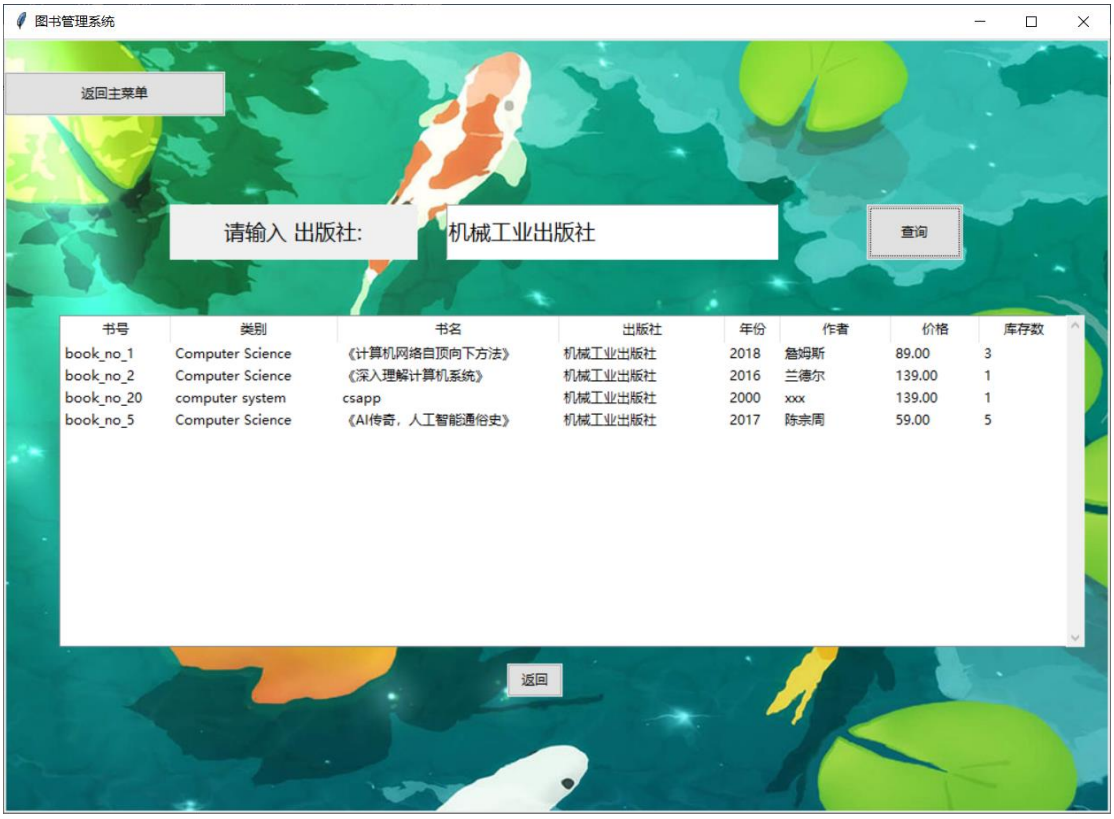
而如果输入的书号并没有被你借走，则会提示错误，如下：

这部分的 sql 语句构建也和借书部分差不多，都是需要在一个事务内执行好多条语句，包括了删除借书记录，同时库存加 1，并且借书证已经借书的数量减 1.

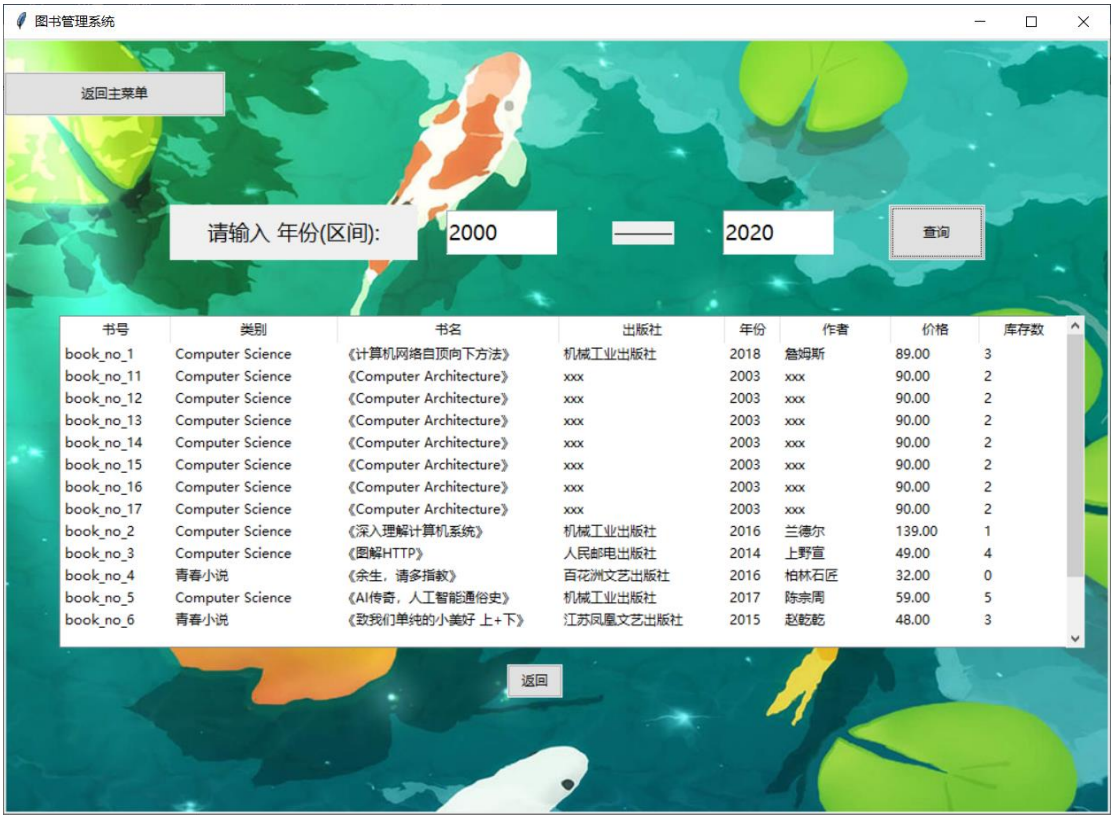
e. 图书查询：

图书查询界面我们首先要选择查询的依据，有以下几种选择：

当我们任意点进一种后，即可输入要查询的信息，然后返回结果。 例如：



并且如果返回的结果过多，还支持滚动查看，也支持右侧的滚动条上下滚动



书号	类别	书名	出版社	年份	作者	价格	库存数
book_no_1	Computer Science	《计算机网络自顶向下方法》	机械工业出版社	2018	詹姆斯	89.00	3
book_no_11	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_12	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_13	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_14	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_15	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_16	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_17	Computer Science	《Computer Architecture》	xxx	2003	xxx	90.00	2
book_no_2	Computer Science	《深入理解计算机系统》	机械工业出版社	2016	兰德尔	139.00	1
book_no_3	Computer Science	《图解HTTP》	人民邮电出版社	2014	上野宣	49.00	4
book_no_4	青春小说	《余生，请多指教》	百花洲文艺出版社	2016	柏林石匠	32.00	0
book_no_5	Computer Science	《AI传奇，人工智能通俗史》	机械工业出版社	2017	陈宗周	59.00	5
book_no_6	青春小说	《致我们单纯的小美好 上+下》	江苏凤凰文艺出版社	2015	赵乾乾	48.00	3

这一部分构造 sql 语句的情况比较多样，主要是看我们选择的查询依据是什么

```
        if index==0:
            sql="select bnum,bclass,bname,publisher,year,author,price,c
                ollection from book where bclass=%s"
        elif index==1:
            sql="select bnum,bclass,bname,publisher,year,author,price,c
                ollection from book where bname=%s"
        elif index==2:
            sql="select bnum,bclass,bname,publisher,year,author,price,c
                ollection from book where publisher=%s"
        elif index==4:
            sql="select bnum,bclass,bname,publisher,year,author,price,c
                ollection from book where author=%s"
        elif index==3:
            y1=int(para1)
            y2=int(para2)
```

```
            sql="select bnum,bclass,bname,publisher,year,author,price,c
                ollection from book where year>%s and year<%s"
        elif index==5:
            p1=float(para1)
            p2=float(para2)
            sql="select bnum,bclass,bname,publisher,year,author,price,c
                ollection from book where price>%s and price<%s"
```

和下面一部分：

```
            if index==0 or index==1 or index==2 or index==4:
                cursor.execute(sql,para1)
            elif index==5:
                cursor.execute(sql,(p1,p2))
            else:
                cursor.execute(sql,(y1,y2))
```

，即可得到查询语句

五、 遇到的问题及解决方法

1. 最大的问题就是这个作业中间重构了两遍…一开始没打算弄图形界面的，但是希望能拿到好一点的分数，想认真对待一下。于是在写完了一整个模块后，重新用 tkinter 写图形界面。然后 tkinter 真的是一言难尽…因为之前 python 用得不多，因此就选了个比较简单的原生的图形库，不过还是折腾了好一番。结果被丑到了…然后后来知道 tkinter 新版本里面自带了稍微美化了一点点的 ttk 库，于是再改用 ttk。

2. 第二个问题前面也提到了，对于 python 我还没有写过大的项目，尤其是 tkinter 也不熟，如何实现两个界面间的跳转我都不知道。于是也上网找了别人的最简单的 demo，中间也学习到了很多。至于为什么选择 python，是因为 python 和 c、Java 比起来真的感觉会轻松很多。和课堂上讲过的 ODBC 和 JDBC 比起来，python 的数据库接口真的是非常简单了。

3.关于 pymysql，我一开始是用的 python 的字符串格式化语句，构造出 sql，然后再调用 cursor.execute(sql)去执行。但是这样有一点就很不好,就是会被 sql 注入攻击。而我一开始找了相关的资料，但是没有注意到一些细节，关于参数化执行的时候，要求是用%s 占位，无论是 int 型还是 float 型还是字符串，都用%s，并且还不要加引号。我一开始没有理解透彻,因此就放弃了。不过后来还是搞懂了，现在的程序是能够防止 sql 注入的

六、 总结

这次的实验真的花费了我比较多的精力,因为是第一次用 python 写大型的程序，并且还是图形界面。总的来说我感觉这次实验中更加花费我精力的倒不是 pymysql 模块，而是 tkinter 图形库模块。我基本上相当于是从零开始，然后遇到了很多问题，幸亏网上的解决方法也比较多。而且万事开头难，一开始遇到的问题相当多，而到后来就顺畅了许多，因为基本上能遇到的问题前面都遇到过了。

除此之外，这也是我第一次用面向对象的语言写大程序。也是体会到了面向对象的方便之处。同时也多亏了 python 灵巧简便的特性，这次的代码量总的来说还可以，大约 1200 行，不知道如果用 c++或者 Java 来写的话需要多少行，不过至少和课上讲的 ODBC 和 JDBC 相比，pymysql 真的简洁很多，现在 web 的后端也有很多大公司都采用了 python 开发。这次的实验也给了我一个很好的机会去深入的了解这样一门语言。

最后有点遗憾就是这个 tkinter 真的是太丑了,不过由于时间原因

我只能选择这样一个稍微简陋的图形库。其实之前一直想用 pyqt5 来开发图形界面的，一直想有机会学习一下，可惜一是没有时间，二是缺乏动力，因此这次也不得以只能做出这样一个没那么美观的作品出来，不过我也真的是倾注了很多心血的，也收获了很多。