

Library-Management-System

18307130252 力维辰

<https://github.com/kleinerbubs/library-management-system>

概述

利用golang和mysql实现了简单的图书馆管理系统。提供了书籍管理、借阅、查询等多种服务，同时设置了管理员、普通读者和游客三种不同的使用权限。

通过编写library_test.go，实现所有功能的交互测试。

提供简单的使用说明文档readme.txt，允许用户通过命令行与程序进行交互。

数据库相关

数据库模式设计

- Booklist(ISBN, title, author, publisher, stock, available, removeInfo)

该表记录所有书籍的相关信息：ISBN、书名、作者名、出版社、库存总数、目前可借阅数量、删除原因

其中，库存总数 = 目前可借阅数量 + 已被借阅数量；

删除原因默认为空，且只有管理员可以看到；

当一本书的所有库存都被管理员移除后，只有管理员可以看见这本书籍的相关信息。

- Userlist(id, name, password, overdue, type)

该表记录所有用户的相关信息：用户名、真实姓名、密码、借阅到期书籍数量、用户类型

其中，用户表中记录的用户类型分为两类：0.管理员 1.普通读者；

借阅到期书籍数量在用户创建时默认为0；

用户是否被禁止借阅通过判断借阅到期书籍数量（overdue）是否超过三本来实现。

- Recordlist(record_id, book_id, user_id, IsReturned, borrow_date, return_date, deadline, extendtimes)

该表记录所有借阅记录的相关信息：记录号、ISBN、用户名、书籍是否已经归还、借阅日期、归还截止日期、还书日期、延期次数

其中，记录号在建表时通过AUTO_INCREMENT设置自增，起始值为1；

归还截止日期在借书时自动生成，默认借阅期限1个月；

还书日期在归还时自动记录，在未归还时空空；

一旦超过截止日期，将自动续借不超过3次；

表格中所有datetime字段的时区都设置为东八区。

数据库连接方式

填写config.ini文件，library.go会自动从config.ini里读取信息，无需对library.go进行额外的修改。

文件前三行的格式为：第一行mysql登录用户名，第二行mysql登陆密码，第三行数据库名称。不要在前三行填写额外的信息。如果mysql的用户名为abc，密码为123，采用的数据库是ass3，那么只要在文件中像下面写的这样填写就可以了：

```
abc
123
ass3
```

功能实现

• 功能测试

修改library.go第22-25行的数据库信息，并在该目录下执行下面的命令

```
go test -v
```

```
=== RUN   TestExtendDeadline
=== RUN   TestExtendDeadline/0
2020/05/07 23:52:35 Book not exists.
=== RUN   TestExtendDeadline/1
2020/05/07 23:52:35 Already extended for three times. Can't extend again.
=== RUN   TestExtendDeadline/2
2020/05/07 23:52:35 Extended successfully.
--- PASS: TestExtendDeadline (0.00s)
    --- PASS: TestExtendDeadline/0 (0.00s)
    --- PASS: TestExtendDeadline/1 (0.00s)
    --- PASS: TestExtendDeadline/2 (0.00s)
=== RUN   TestReturnBook
=== RUN   TestReturnBook/0
2020/05/07 23:52:35 Book not exists.
=== RUN   TestReturnBook/1
2020/05/07 23:52:35 Returned successfully.
=== RUN   TestReturnBook/2
2020/05/07 23:52:35 Returned successfully.
--- PASS: TestReturnBook (0.01s)
    --- PASS: TestReturnBook/0 (0.00s)
    --- PASS: TestReturnBook/1 (0.00s)
    --- PASS: TestReturnBook/2 (0.00s)
PASS
ok      github.com/ichn-hu/IDBS-Spring20-Fudan/assignments/ass3/boilerplate    0.203s
```

由于测试打印内容较多，所以仅截取了最后的内容以作参考

• 建表

```
func (lib *Library) CreateTables() error {}
```

正常情况下返回值nil

• 检查用户是否存在

```
func (lib *Library) CheckUserExists(userid string) error {}
```

根据用户id，检查用户是否存在。存在返回nil，用户不存在时返回ErrUserNotExists，否则返回错误信息err

• 检查书籍是否存在

```
func (lib *Library) CheckBookExists(ISBN string) error {}
```

根据ISBN，检查书籍是否存在。存在返回nil，书籍不存在（或者已经被彻底移除时）返回ErrBookNotExists，否则返回错误信息err

• 增加一名用户

```
func (lib *Library) AddUser(user Users) error {}
```

向Userlist表中插入一条新的用户信息，正常情况下返回值为nil

• 修改用户密码

```
func (lib *Library) ModifyPassword(userid, password string) error {
```

修改Userlist中的用户信息，正常情况下返回值为nil

• 用户登录检查

```
func (lib *Library) IdentifyUser(userid, password string) (Users, error) {
```

当用户名和密码匹配时返回nil以及用户的所有相关信息，否则返回ErrPassword

• 增加书籍

```
func (lib *Library) AddBook(bookTitle, bookISBN, bookAuthor, bookPublisher  
string, bookStock int) (int, error) {}
```

根据输入信息添加书籍。支持一次性添加多本相同的书籍。只能由管理员账号进行操作。

在书籍信息不存在时直接添加书籍信息，否则修改库存信息。

int返回当前该书籍的库存量，error正常情况下返回nil

• 删除书籍

```
func (lib *Library) RemoveBook(bookISBN, bookRemoveInfo string) (int, error) {}
```

根据书号删除书籍。一次只能删除一本书。只能由管理员账号进行操作。

int返回当前书籍库存量，存在错误时int返回-1

error在书籍不存在时返回ErrBookNotExist；在书籍已经全部被删除时返回ErrAllRemoved；在成功操作时返回nil；否则返回错误信息

• 查询书籍

```
func (lib *Library) QueryBookTitle(keyTitle string) ([]Books, error) {}  
func (lib *Library) QueryBookAuthor(keyAuthor string) ([]Books, error) {}  
func (lib *Library) QueryBookISBN(keyISBN string) ([]Books, error) {}
```

支持根据书名、作者名以及ISBN码三种查询方式。

查询时，对于ISBN码要求完全匹配，对于书名和作者名采用模糊匹配，检索所有包含该关键字（忽略大小写）的书籍，并按照匹配度进行排序，检索结果最相似的靠前。

[]Books字段返回书籍信息，error正常时返回nil

在信息打印时，所有库存为0的书本信息以及删除书本的原因都只有管理员能看到。

需要注意的是，这里的匹配度是按照检索字段的长度排序的，所以只能保证检索结果最相似的靠前，但是存在查找包含单词“lie”的书名，但是返回的书名中含有“believe”的情况。该

• 借书

```
func (lib *Library) BorrowBook(bookISBN, userID string, borrowDate time.Time)
error {}
```

根据书号借阅一本书籍。borrowDate为借阅日期。

只有overdue<=3的用户才能进行这一操作。并且为了保证资源最大化利用，对于同一本书，一个读者不能同时借阅多本。

如果书籍不存在返回ErrBookNotExists；如果全部被借走了返回ErrBookNotAvailable；如果读者在尚未归还该书的情况下试图再次借阅返回ErrAlreadyBorrowed；在成功借书时返回nil。

• 还书

```
func (lib *Library) ReturnBook(bookISBN, userID string) (int, error) {
```

根据书号归还一本书。

如果书籍不存在返回ErrBookNotExists；如果用户没有借阅该书返回ErrNotBorrowed；在成功还书时返回nil。

• 续借

```
func (lib *Library) ExtendDeadline(bookISBN, userID string) error {}
```

根据书号进行续借。

如果书籍不存在返回ErrBookNotExists；如果用户没有借阅该书返回ErrNotBorrowed；已经（自动）续借3次，不能再次续借返回ErrNoMoreExtended；在成功续借时返回nil。

• 查询还书截止日期

```
func (lib *Library) CheckDeadline(bookISBN, userID string) (Records, error) {}
```

根据书号查询用户的还书截至日期。

书籍信息不存在时返回ErrBookNotExists；查询的书籍没有被借阅时返回ErrNotBorrowed；成功还书时返回nil。

• 查询历史借阅记录

```
func (lib *Library) CheckBorrowHistory(userID string) ([]Records, error) {}
```

查询用户的所有借阅记录，并按照从借书时间从近到远排序。

[]Records返回所有借阅记录，error返回nil

• 查询未还书籍的借阅记录

```
func (lib *Library) CheckUnreturned(userID string) ([]Records, error) {}
```

查询用户的所有未还书籍的借阅记录，并按照从借书时间从近到远排序。

[]Records返回所有借阅记录，error返回nil

- **查询借阅到期书籍数量**

```
func (lib *Library) CheckOverdue(userID string, now time.Time) (int, []Records, error) {
```

查询用户到当前时刻的所有已经到期的借阅书籍数量

int返回有到期的借阅书籍数量，[]Records返回所有的到期借阅记录，error正常返回nil

命令行交互

请确保先测试系统，再启动系统，否则可能会造成数据丢失以及测试错误。

mysql的连接方式和之前一样，命令行测试中额外提供两个文件mysql_reset.sql和book_sample.sql。

在第一次正式使用前执行mysql_test.sql，实现数据库的重置，注意数据库名称必须和config.ini中的一致

重置时自动添加一条默认root管理员信息，用户名和密码都是root

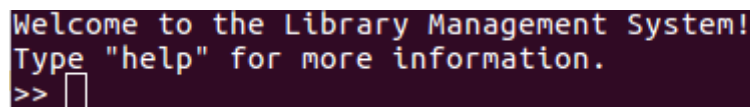
```
mysql -h localhost -u root -p DBName < mysql_reset.sql
```

book_sample.sql中内置了一些书籍信息以供命令行测试(可选，不导入时管理系统内没有任何书籍)

```
mysql -h localhost -u root -p DBName < book_sample.sql
```

完成初始化后，在终端执行下面的命令，启动管理系统

```
go run library.go
```



```
Welcome to the Library Management System!  
Type "help" for more information.  
>> 
```

出现上图即说明登录成功，具体的交互指令说明可以选择输入help获取，也可以直接阅读readme.txt文件

下图是一个简单的交互界面示例。

```
Welcome to the Library Management System!
Type "help" for more information.
>> register
Username: user
RealName: demo
Password: 123abc
ConfirmPassword: 123abc
2020/05/08 10:49:33 Registered Successfully. You can login now.
>> login
Username: user
Password: 123abc
2020/05/08 10:49:38 Login Successfully.
demo@library: title
BookTitle: camino



| ISBN           | Title        | Author       | Publisher                  | Stock | Available |
|----------------|--------------|--------------|----------------------------|-------|-----------|
| 978-0385545938 | Camino Winds | John Grisham | Doubleday (April 28, 2020) | 3     | 2         |



demo@library: borrow
BookISBN: 978-0385545938
2020/05/08 10:49:56 Borrowed successfully.
demo@library: deadline
BookISBN: 978-0385545938
Deadline: 2020/06/08 02:49:57
demo@library: extend
BookISBN: 978-0385545938
2020/05/08 10:50:23 Extended successfully.
demo@library: unreturned



| ISBN           | Title        | BorrowDate          | Deadline            |
|----------------|--------------|---------------------|---------------------|
| 978-0385545938 | Camino Winds | 2020/05/08 02:49:57 | 2020/06/08 02:49:57 |



demo@library: history



| ISBN           | Title        | IsReturned | BorrowDate          | ReturnDate |
|----------------|--------------|------------|---------------------|------------|
| 978-0385545938 | Camino Winds | false      | 2020/05/08 02:49:57 | NULL       |



demo@library: return
BookISBN: 978-0385545938
2020/05/08 10:50:47 Returned successfully.
demo@library: history



| ISBN           | Title        | IsReturned | BorrowDate          | ReturnDate                    |
|----------------|--------------|------------|---------------------|-------------------------------|
| 978-0385545938 | Camino Winds | true       | 2020/05/08 02:49:57 | 2020-05-08 02:50:47 +0000 UTC |



demo@library: adduser
adduser : command not found
demo@library: exit
>> exit
```

Reference

表格打印开源包: <https://github.com/modood/table>

mysql官方api文档: <https://golang.org/pkg/>