FUDAN Library Management System实验报告

18307130024 谭一凡

项目地址

目录

功能概述

我设计的图书馆管理系统实现了以下几方面功能：

- 用户管理与登录：

- ☑ 管理员和普通用户信息存储在同一个表中，通过“权限”属性加以区分。
- ☑ 用户名以明文存储，**密码经sha256算法加密后存储。**
- ☑ 管理员可添加普通用户与管理员用户。
- ☑ **用户修改密码**

- 书籍管理：

- ☑ 书籍分两个表存储，为：书单(ISBN, 书名, 作者, 借阅次数（默认为0）)以及单册(书号, $\underset{\sim}{I}$$\underset{\sim}{S}$$\underset{\sim}{B}$$\underset{\sim}{N}$, 是否被借阅（默认为未借阅）)。
- ☑ 用删除记录(书号, 删除信息)来记录单册书的删除记录。
- ☑ 管理员添加书时，需管理书单和单册书两个关系表，以**减少数据冗余。**未加进书单的，不允许加入单册表中。
- ☑ **支持从csv文件中读取数据，以及batch insert。**

- 查询、借阅与归还：

- ☑ 可根据ISBN、书名与作者在书单表中查询，其中：作者支持单关键词模糊查询，**书名支持多关键词模糊查询。**
- ☑ 用借阅记录(借阅编号（自增），$\underset{\sim}{用}$$\underset{\sim}{户}$$\underset{\sim}{名}$，$\underset{\sim}{书}$$\underset{\sim}{号}$，借阅时间，应还时间，归还时间，罚款，续借次数)来记录每次借阅与归还信息。
- ☑ 每次借阅时查询用户的已借阅数和过期数，以判断用户是否可以再借书。
- ☑ 用书号进行借阅与归还。
- ☑ 查询用户的借阅记录、哪些书未归还。
- ☑ 查询书籍的到期时间。

- ☑ 用户续借。
- ☑ **用户预约。**
- ☑ 添加书本的借阅次数属性，**用户可查询借阅次数前10的热门书籍。**

- 控制台界面

- ☑ 系统初始以“游客”身份登录。一个终端同一时间内只允许一个用户登录。用户退出后进入游客模式。
- ☑ **用户输入密码时，控制台不会回显。**
- ☑ 对不同权限的用户，有为其定制的帮助界面。
- ☑ 查询记录表格化输出，较为美观。

用户系统

1. 关系模式与存储

管理员和普通用户存储于一个表中，模式为users(username, password, root)。

```
create table if not exists users
(
    username nvarchar(200) primary key,
    password nvarchar(200),
    root bool default 0
);
```

建表时会生成一个初始管理员用户，用户名和密码都是root。

用户的密码会经过sha256函数处理后存储到users表中。

```
func getSHA256(input string) string {
    hash := sha256.New()
    hash.Write([]byte(input))
    bytes := hash.Sum(nil)
    hashCode := hex.EncodeToString(bytes)
    return hashCode
}
```

2. 修改密码

用户可以修改密码。修改密码时，需要输入原密码→输入新密码→重复新密码，以完成密码的修改。

3. 用户权限

有三种用户权限：游客、普通用户权限以及管理员权限。不同用户权限能执行的操作不同。

用户权限	查询类	借阅类	图书、用户管理
游客	✔	x	x

用户权限	查询类	借阅类	图书、用户管理
普通用户	✔	✔	x
管理员	✔	✔	✔

进入图书馆系统后，默认为游客模式。用户登录后，进入用户对应的权限模式；退出后，进入游客模式。

图书系统

1. 关系模式

实验handout中提到图书的属性有: ISBN, title, author。
在图书馆中，同一本书会有很多复本，所以我加了一个bookid的属性，作为单册书的主键。 上述四个属性有以下函数依赖：bookid → ISBN, ISBN → {title, author}。所以我分了两个表来存储书，以减少冗余：

```
    create table if not exists booklist #书单，以ISBN为主键
(
    title     nvarchar(200),
    author    nvarchar(200),
    ISBN      nvarchar(200) primary key,
    visits    int default 0 #被借阅次数，后面的热门榜会用到
);
    create table if not exists singlebook #单册书，以bookid为主键，ISBN为外键
(
    ISBN      nvarchar(200),
    bookid    nvarchar(200) primary key,
    available bool default true, #是否可被借阅
    foreign key (ISBN) references booklist(ISBN) on delete cascade
);
```

2. 查询

对booklist，支持以下几种查询：

- 通过ISBN（精确查询）
- 通过author（单关键字查询）
- **通过title（多关键字查询，书的title同时包含所有关键字）**
- **热门书单（被借阅次数前十的书籍）**
- 通过bookid查询书目的到期时间

对singlebook，支持精确搜索ISBN，得到bookid和available属性。

其中title的多关键字搜索，实现如下:

```
// query the booklist by title
func querybookbytitle(title []string, lib *Library) ([]Book, error) {
    var books []Book
    query := "select title, author, ISBN from booklist "
```

```
    for index, value := range title {
        if index == 0 {
            query = query + "where "
        } else {
            query = query + "and "
        }
        query = query + fmt.Sprintf("title like '%%%s%%' ", value) // 关键词逐一加
入查询语句中
    }
    rows, err := lib.db.Queryx(query)
    // 以下为查询处理部分，略去
}
```

热门书单查询的实现，就是booklist表中按借阅次数降序排序，然后limit 10。

借阅系统

用户可根据bookid来进行借阅。
除了基本的借还书、续借，我还实现了**预约功能**。详见下文。

1. 关系模式

借阅记录表的关系模式定义如下：

```
    create table if not exists rent
(
    rentdate nvarchar(200), #日期用字符串存储，通过后端进行处理
    duedate nvarchar(200),
    returndate nvarchar(200) default "not returned yet", #该属性也可区分书是否归还
    fine float default 0, #在还书时更新罚款
    rentid int primary key auto_increment,
    username nvarchar(200),
    bookid nvarchar(200),
    extend int default 0, #已续借次数
    foreign key (username) references users(username),
    foreign key (bookid) references singlebook(bookid)
);
```

预约记录表的关系模式定义如下：

```
create table if not exists appointment
(
    id int primary key auto_increment, #预约id也作为预约先后的判断标准
    username nvarchar(200),
    bookid nvarchar(200),
    borrowed nvarchar(200) default "No",
    foreign key (username) references users(username),
    foreign key (bookid) references singlebook(bookid)
)
```

2. 借还书

用户通过bookid来进行借书。

借书前，检查以下几件事：

> 检查借阅资格：逾期书是否超过上限（默认3）、未还书是否超过上限（默认30）。
> 检查书是否存在，书是否已归还。
> 检查是否有人已预约，该用户是否为预约等待队列中最早预约的。

如果检查都通过，则更新以下信息：

> 在rent表中添加相关记录；
> 在singlebook表中把这本书的available置0；
> 在book表中把这本书的visit加1；
> 在预约表中更新相关的预约信息。

rent表中有duedate属性，需要用到时间操作。计算duedate的代码如下：

```
rentdate := time.Now().Format(dateformat) //dataformat为"2006-01-02 15:04:05"
duedate := time.Unix(time.Now().Unix(), int64(du*time.Hour*24)).Format(dateformat)
//du是1个借阅周期的时间，默认为30
```

还书时，检查以下信息：

> 用户是否借了这本书，并未归还

如果检查通过，则更新以下信息：

> 更新借阅记录的returndate；
> 更新借阅记录的fine；
> 如果该书不在预约队列里，则在singlebook中把该书的available设置为1。

3. 查询借阅记录

支持查询历史借阅记录、当前未还书记录、未还且逾期的记录。
普通用户只能查询自己的记录，管理员可查询所有人的记录。

记录按以下形式储存：

```
type Rent struct {
    Rentdate   string
    Duedate    string
    Returndate string
    Fine       float32
    Bookid     string
    ISBN       string
    Title      string
```

```
    Author      string
}
```

这几种查询的实现类似。以查询未还且逾期的记录为例：

```
var rentdate, duedate, returndate, bookid, ISBN, title, author string
var fine float32
var rent []Rent
query := fmt.Sprintf(`select rentdate, duedate, returndate, fine, rent.bookid,
booklist.ISBN, title, author
                            from rent, booklist, singlebook
                            where username = '%s' and returndate = 'not returned
yet' and rent.bookid = singlebook.bookid
                            and singlebook.ISBN = booklist.ISBN
                            order by rentid`, username) //从数据库中查询用户未归还的
借阅信息，在后端检查是否逾期
rows, err := lib.db.Queryx(query)
if err != nil {
    return nil, err
}
now, _ := time.Parse(dateformat, time.Now().Format(dateformat)) //删掉时区信息
for rows.Next() {
    err = rows.Scan(&rentdate, &duedate, &returndate, &fine, &bookid, &ISBN,
&title, &author)
    if err != nil {
        return nil, err
    }
    duedate1, _ := time.Parse(dateformat, duedate)
    if duedate1.Unix() < now.Unix() { // 如果逾期
        rent = append(rent, Rent{Rentdate: rentdate, Duedate: duedate, Returndate:
returndate, Fine: fine, Bookid: bookid, ISBN: ISBN, Title: title, Author: author})
    }
}
```

4. 续借

用户借阅书籍后，可以进行续借操作。
每次借阅的续借次数有上限（默认为3）。

续借的实现是，将数据库中借阅记录的duedate（varchar类型）读取到本地，为string类型；将string类型转换成时间戳进行处理后再转换成string类型，存回数据库，具体如下：

```
// duedate为数据库中借阅记录的到期时间，为string类型
newduedate, err := time.Parse(dateformat, duedate) //dataformat为"2006-01-02
15:04:05"
if err != nil {
    return err
}
```

```
newduedate1 := time.Unix(newduedate.Unix(),
int64(du*time.Hour*24)).Format(dateformat)
```

5. 预约

用户可以预约书籍。

当用户发出预约请求时，系统检查以下几点：

> 请求的bookid是否合法（不在singlebook表里，或被管理员移除）？
> 是否可直接借阅?
> 该用户是否已借了这本书？ 该用户对这本书是否有未处理的预约请求?

如果检测通过，则添加预约请求，并返回该用户之前还有多少个没处理的预约请求。

管理员系统

相对普通用户来说，管理员对图书馆系统拥有更高的权限。

1. 添加用户、书籍

- 向users表中添加用户
- 向booklist、singlebook表中添加书籍
- **支持从csv文件中读取数据进行批量添加**

批量添加的实现如下（以批量添加用户为例）：

```
// add accounts to table users, using batch insert
func adduser_batch(user *[]User, lib *Library) error {
    exec := "insert ignore into users(username, password, root) values "
    if len(*user) < 1 {
        return nil
    }
    for index, value := range *user {
        u, p, r := value.username, getSHA256(value.password), value.root
        exec = exec + fmt.Sprintf("('%s','%s',%d)", u, p, r)
        if index < len(*user)-1 {
            exec = exec + ","
        }
    }
    _, err := lib.db.Exec(exec)
    return err
}
```

从csv中读取数据的实现如下（以批量添加用户为例）：

```
func readuser(filename string) ([]User, error) {
    fs, err := os.Open(filename)
    if err != nil {
```

```
            return nil, err
        }
        defer fs.Close()
        reader := csv.NewReader(fs)
        var users []User
        for {
            row, err := reader.Read()
            if err == io.EOF {
                break
            }
            if err != nil {
                return nil, err
            }
            u, p := row[0], row[1] //username, password
            r, err := strconv.Atoi(row[2]) //root
            if err != nil {
                return nil, err
            }
            users = append(users, User{u, p, r})
        }
        return users, nil
    }
```

2. 移除书籍

管理员可以移除单册书。
被移除的书籍会加进removelist表中，但不会从singlebook表中删除掉（singlebook的主键bookid是rent表中属性bookid的外键，不删的作用是保留rent中记录）。
移除前，需检查这本书是否归还。如果未归还即删除，当用户归还时会出错。
书被移除后，会在removelist表中有相应记录，并将singlebook表中该书的available置0。

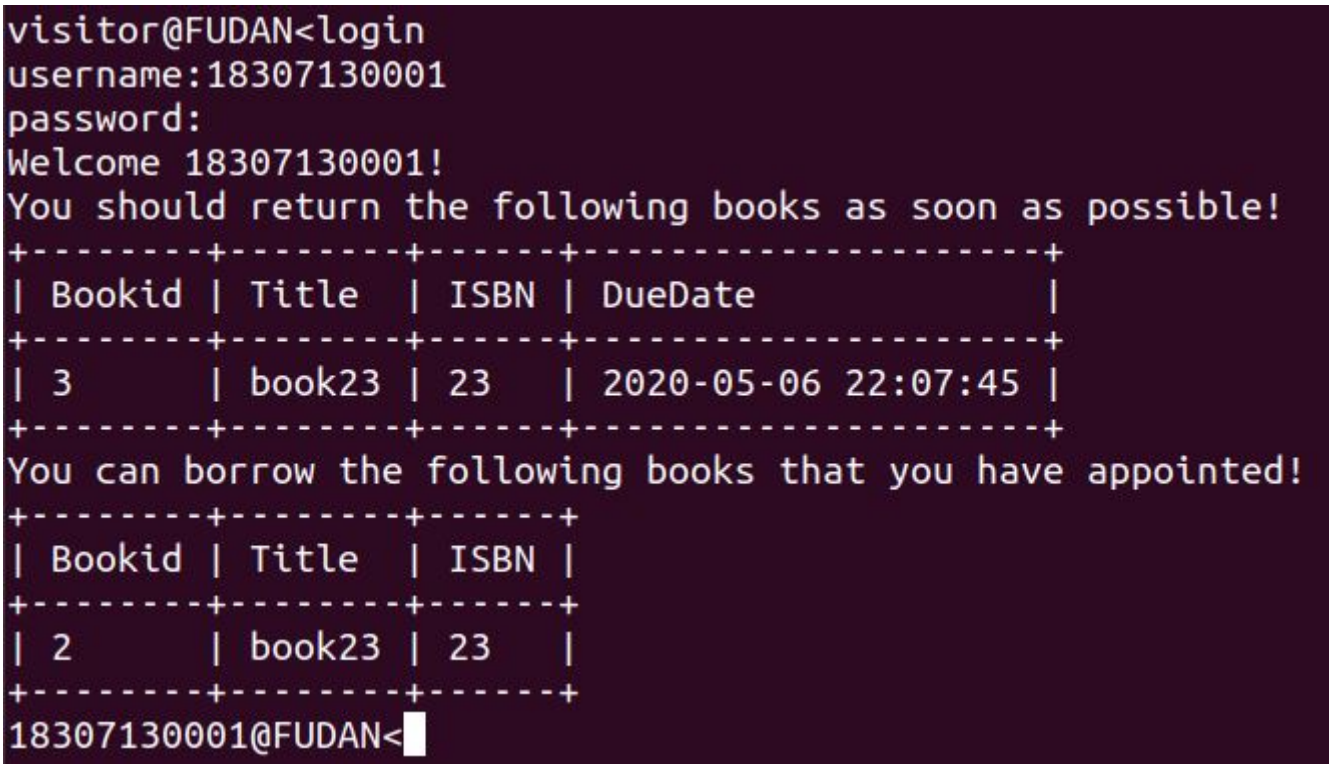
前端

我模仿linux的bash终端，为我的后端管理系统设计了一个控制台前端界面。
用户可以输入命令来调用对应的函数接口。

下面介绍几点我设计的前端的一些特点：

1. 登录提示，以及密码保护

下图为登录时的界面：



用户登录时，系统会提示你七天内将到期（或已到期）的图书，以及你预约过的且轮到你借阅的图书。
登陆后，命令行前缀会变成"<用户名>@FUDAN"的形式。

同时，输入密码时，不会回显。其原理是生成一个进程，执行"stty -echo"。代码如下：

```
attrs := syscall.ProcAttr{
        Dir:   "",
        Env:   []string{},
        Files: []uintptr{os.Stdin.Fd(), os.Stdout.Fd(), os.Stderr.Fd()},
        Sys:   nil}
var ws syscall.WaitStatus
pid, err := syscall.ForkExec(
    "/bin/stty",
    []string{"stty", "-echo"},
    &attrs)
if err != nil {
    return
}
_, err = syscall.Wait4(pid, &ws, 0, nil)
if err != nil {
    return
}
```

密码输入完毕后，其他数据输入时需要回显，则执行"stty echo"即可。

2. 帮助界面

不同用户权限的用户可进行不同的操作。输入非法指令或者help会显示这些用户可执行的操作。
以下三张图分别展示了游客、普通用户、管理员用户可执行的指令，以及这些指令的描述。

```
visitor@FUDAN<Invalid command will call help!
Type "help" can get this help.
+-----------------------------+-------------------------------------------+---------------------+
| Command                     | Description                               | Example             |
+-----------------------------+-------------------------------------------+---------------------+
| quit                        | quit                                      | quit                |
| login                       | login                                     | login               |
| logout                      | logout                                    | logout              |
| ISBN <ISBN>                 | search by ISBN                            | ISBN abc            |
| title <title> [title, ...] | search by title in mutiple keywords       | title math analysis |
| author <author>             | search by author                          | author a            |
| bookid <ISBN>               | get bookid of books whose ISBN is <ISBN>  | bookid abc          |
| duedate <bookid>            | query the duedate of a borrowed book      | duedate 1           |
| topten                      | query the top10 bestsellers               | topten              |
+-----------------------------+-------------------------------------------+---------------------+
```

```
18307130001@FUDAN<help
Type "help" can get this help.
+-----------------------------+-------------------------------------------------+---------------------+
| Command                     | Description                                     | Example             |
+-----------------------------+-------------------------------------------------+---------------------+
| quit                        | quit                                            | quit                |
| login                       | login                                           | login               |
| logout                      | logout                                          | logout              |
| ISBN <ISBN>                 | search by ISBN                                  | ISBN abc            |
| title <title> [title, ...] | search by title in mutiple keywords             | title math analysis |
| author <author>             | search by author                                | author a            |
| bookid <ISBN>               | get bookid of books whose ISBN is <ISBN>        | bookid abc          |
| borrow <bookid>             | borrow the book whose id is <bookid>            | borrow 2            |
| return <bookid>             | return the book whose id is <bookid>            | return 2            |
| extend <bookid>             | extend the duedate of book whose id is <bookid> | extend 2            |
| changepassword              | change your password                            | changepassword      |
| list                        | query your borrow record                        | list                |
| duedate <bookid>            | query the duedate of a borrowed book            | duedate 1           |
| overdue                     | query overdue books of your account             | overdue             |
| topten                      | query the top10 bestsellers                     | topten              |
| unreturn                    | query the unreturned books of your acoount      | unreturn            |
+-----------------------------+-------------------------------------------------+---------------------+
```

```
root@FUDAN<help
Type "help" can get this help.
+-------------------------------------+-----------------------------------------------------------------------+----------------------+
| Command                             | Description                                                           | Example              |
+-------------------------------------+-----------------------------------------------------------------------+----------------------+
| quit                                | quit                                                                  | quit                 |
| login                               | login                                                                 | login                |
| logout                              | logout                                                                | logout               |
| ISBN <ISBN>                         | search by ISBN                                                        | ISBN abc             |
| title <title> [title, ...]          | search by title in mutiple keywords                                   | title math analysis  |
| author <author>                     | search by author                                                      | author a             |
| bookid <ISBN>                       | get bookid of books whose ISBN is <ISBN>                              | bookid abc           |
| duedate <bookid>                    | query the duedate of a borrowed book                                  | duedate 1            |
| borrow <bookid>                     | borrow the book whose id is <bookid>                                  | borrow 2             |
| return <bookid>                     | return the book whose id is <bookid>                                  | return 2             |
| extend <bookid>                     | extend the duedate of book whose id is <bookid>                       | extend 2             |
| changepassword                      | change your password                                                  | changepassword       |
| add user <username> <password> <root> | add user                                                            | add user root1 root1 1 |
| add users [filepath]                | add user from csv file, default filepath'../data/users.csv'           | add users            |
| add book <title> <author> <ISBN>    | add book to booklist                                                  | add book a b c       |
| add books [filepath]                | add book to booklist from csv file, default filepath'../data/books.csv' | add books          |
| add sbook <bookid> <ISBN>           | add singlebook                                                        | add a b              |
| add sbooks [filepath]               | add singlebook from csv file, default filepath'../data/sbooks.csv'    | add sbooks           |
| list [username]                     | query borrow record of [username], default yours                      | list 18307130001     |
| overdue [username]                  | query overdue books of the account, default yours                     | overdue 18307130001  |
| topten                              | query the top10 bestsellers                                           | topten               |
| remove <bookid> <detail>            | remove a singlebook with detail                                       | remove 3 lost        |
| unreturn [username]                 | query unreturned books of a user, default yours                       | unreturn             |
+-------------------------------------+-----------------------------------------------------------------------+----------------------+
```

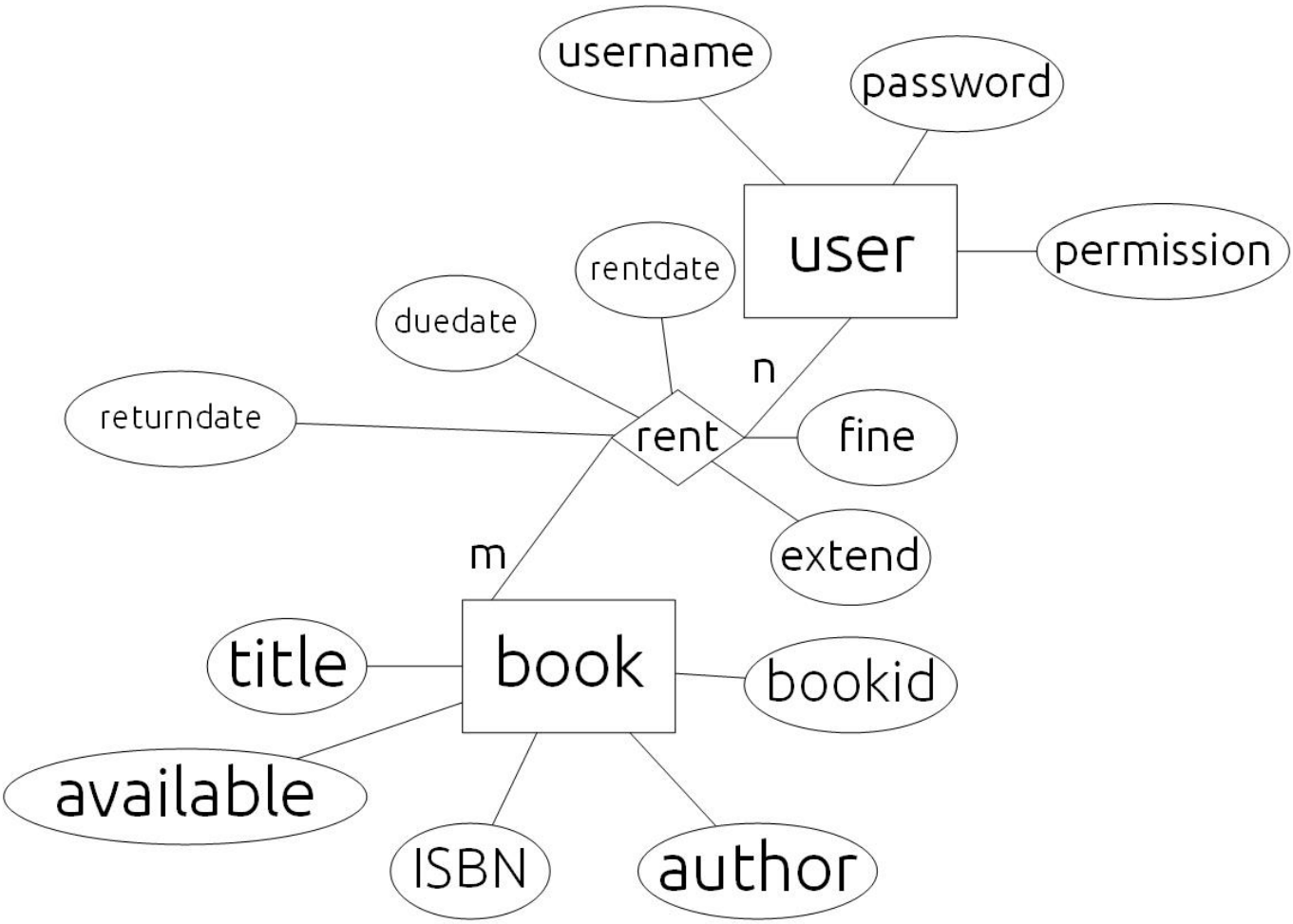
3. 表格化输出

```
18307130001@FUDAN<title b o o k
+---------------------------------------+---------------+------+
| Title                                 | Author        | ISBN |
+---------------------------------------+---------------+------+
| book20                                | 1             | 20   |
| book21                                | 2             | 21   |
| book21                                | 3             | 22   |
| book23                                | 3             | 23   |
| Database systems : the complete book | Garcia-Molina | 800  |
+---------------------------------------+---------------+------+
18307130001@FUDAN<
```

我调用了第三方包(github.com/modood/table)，把切片进行表格化输出，效果如上图所示。
这个库用加号、减号构成表格的框，同时进行了对齐。

借阅关系的ER图

图书馆管理系统中，查询和借阅是核心功能。借阅关系的ER图如下：



Reference

Boilerplate by ichn-hu
表格化输出工具