数据库系统概论课程设计

161640111，杜云，15651808839

物联网工程

Email：muyun@nuaa.edu.com

**摘 要:** 在近一个月的时间里，我学习了基于Python语言的Web后端框架Django，并使用其实现了图书管理系统。在数据库设计方面，我设计了用户表，图书表，借阅记录表三个表，并分别对这三个表实现增删改查的操作。图书表中图书数据有1003759条，在中国图书出版数据库(<http://www.cnpub.com.cn/>)上进行爬取，实现了本课程设计中100w数据量的要求。在实现了三个表增删该查之后，我对图书管理系统的实际使用进行了部分调研，并解决了其中两个棘手的问题：第一是数据量较大时，查询结果在web端展示的时间会很长，最后进行分析利用分页技术解决了此问题，并添加了模糊查询的功能。第二是管理员和普通用户的权限控制问题，通过对用户登陆后的Session数据中的用户身份进行条件判断，实现了对相关按钮的隐藏，这是基础的安全控制策略。在进一步分析后，发现web页面可以通过url进行访问，所以还需在其他页面中逻辑的改进，使得普通用户不能获得相应权限，实现了初级权限控制。综上，该图书管理系统实现了对用户表，图书表和借阅记录表的增删改查，在100w条数据的查询时间控制在1秒以内，并且实现了管理员和用户的初级权限控制。在考虑了真实环境下的图书借还的条件后，实现了对应功能的提醒信息，具有一定的鲁棒性。

**关键词: Django框架；模糊查询，查询优化，权限控制**

# 1 数据库设计

## 1.1 系统需求分析

进行了基于结构化分析(Structured Analysis)方法，自顶向下的分析了图书管理系统的需求：图书馆有若干本书，有若干用户，单条借阅记录是由

## 1.2 概念结构设计

## 1.3 逻辑结构设计

## 1.4 物理结构设计

## 1.5 数据库实施

## 1.6 数据库运行与维护

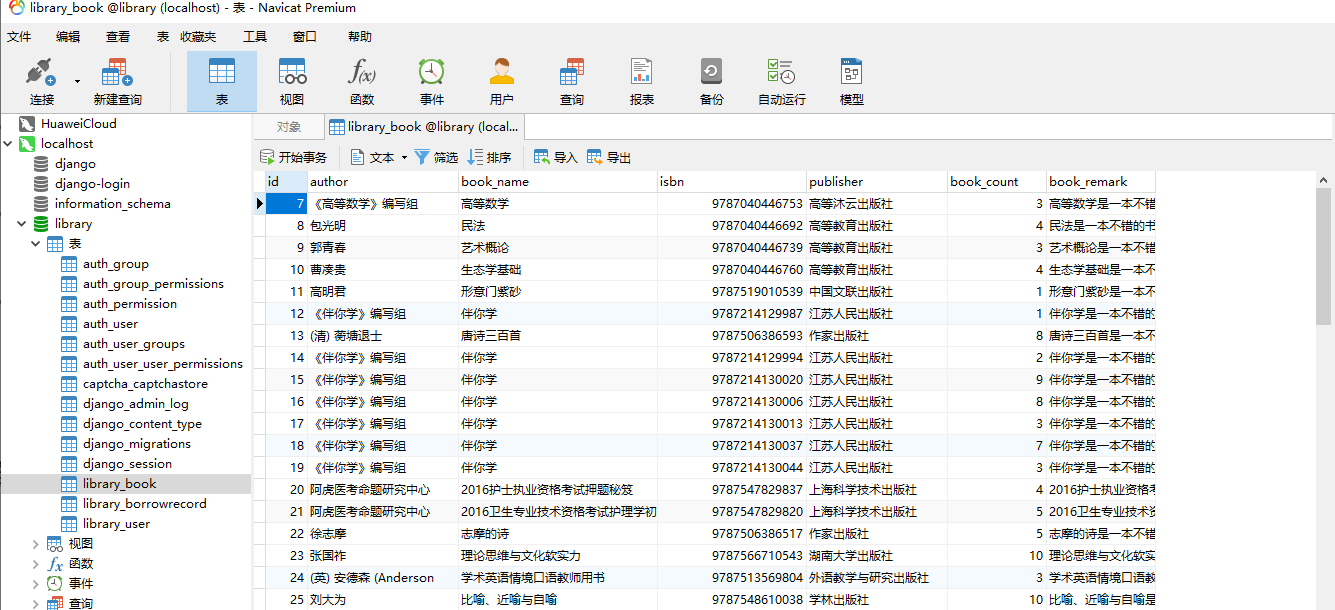


图1 数据库一览

最终，我设计了三个表：用户表、图书表以及借阅记录表，接下来我会分别对这三个表的结构做一个说明

1. 用户表

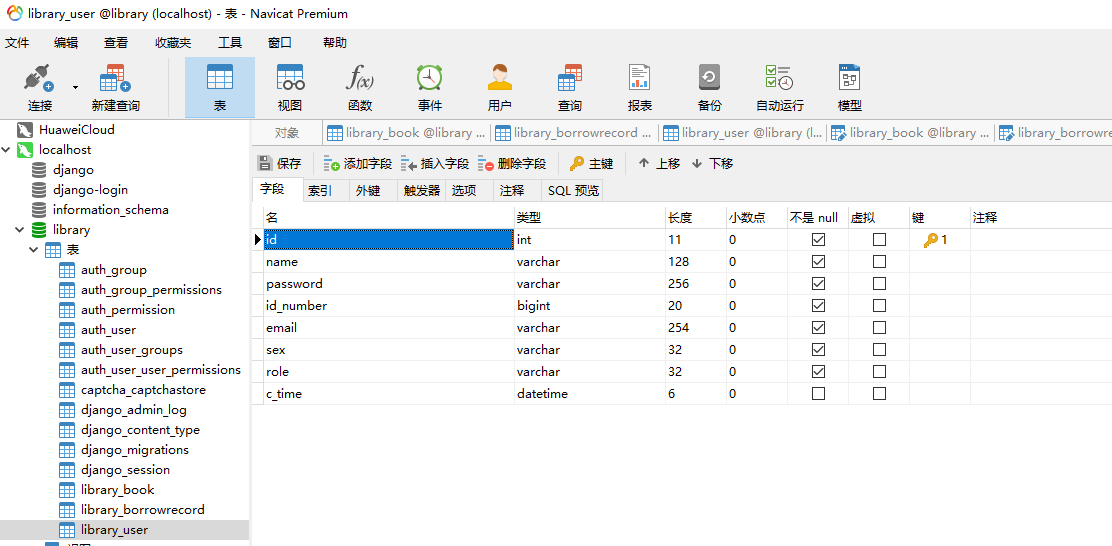


图2 用户表设计

如图1所示，在用户表中，我设计了8个字段(包括自增的主键id)，用户表的字段及其含义如表1所示。

表1 用户表的字段及其含义

|  |  |  |
| --- | --- | --- |
| 字段名 | 字段含义 | 备注 |
| *id* | 记录号 | 自增的主键 |
| *name* | 用户名 | 用户自定义(唯一) |
| *password* | 用户密码 | 用户自定义 |
| *id\_number* | 用户会员卡号 | 会员卡号在注册时分配(唯一) |
| *email* | 用户邮箱 | 用户自定义(唯一) |
| *sex* | 用户性别 | 用户自定义 |
| *role* | 用户身份 | 分为*admin*与*general\_user*，注册默认为*general\_user* |
| *c\_time* | 用户注册时间 | 在创建时自动生成 |

1. 图书表

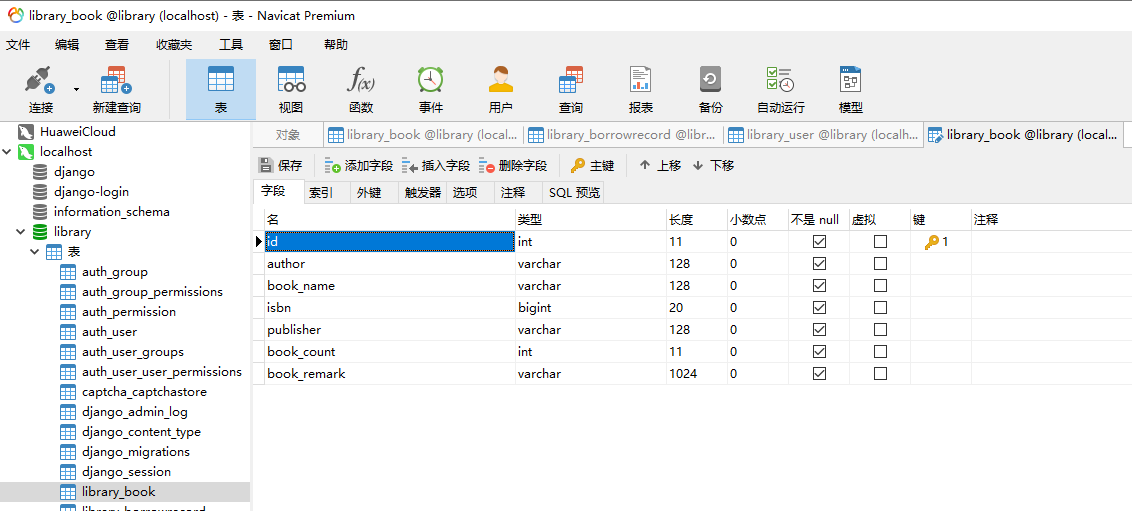


图3 图书表设计

表2 图书表的字段及其含义

|  |  |  |
| --- | --- | --- |
| 字段名 | 字段含义 | 备注 |
| *id* | 记录号 | 自增的主键 |
| *author* | 书籍作者 | 爬取网站数据所得 |
| *book\_name* | 书籍名 | 爬取网站数据所得 |
| *isbn* | 书籍ISBN码 | 爬取网站数据所得(唯一) |
| *publisher* | 书籍出版社 | 爬取网站数据所得 |
| *book\_count* | 书籍数 | 书库中剩余书籍数量 |
| *book\_remark* | 书籍简介/评论 | 由于所爬取网站未爬到数据，此处为自己生成 |

1. 借阅记录表

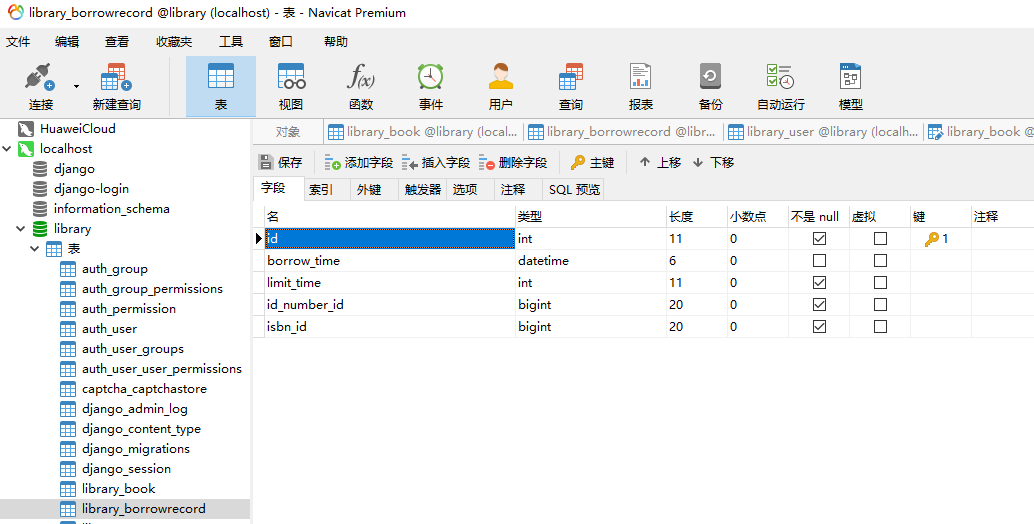


图4 借阅记录表设计

表2 借阅记录表的字段及其含义

|  |  |  |
| --- | --- | --- |
| 字段名 | 字段含义 | 备注 |
| *id* | 记录号 | 自增的主键 |
| *borrow\_time* | 借阅时间 | 借书时自动创建 |
| *limit\_time* | 借书限制时间 | 可借天数 |
| *id\_number\_id* | 借书人 | 外键--user表中id\_number |
| *isbn\_id* | 所借书籍的ISBN码 | 外键--book表中的isbn |

如上图所示，在借阅记录表表中，我设计了5个字段(包括自增的主键id)，借阅记录表的字段及其含义如表2所示。

# 2 对数据库的基础操作

对数据的基础操作包括建表及对单独表的增删改查，此处主要讲讲在对单独表的增删查改中遇到的若干问题

## 2.1 对图书表的增删查改

主要是使用SQL语言对图书表进行增删查改，分以下两种情况：

1. 若其为管理员身份：其可以对

## 2.2 对用户表的增删查改

## 2.3 对借阅记录表的增删查改

# 3 遇到的主要问题及解决方法

## 3.1 解决100w数据量的问题

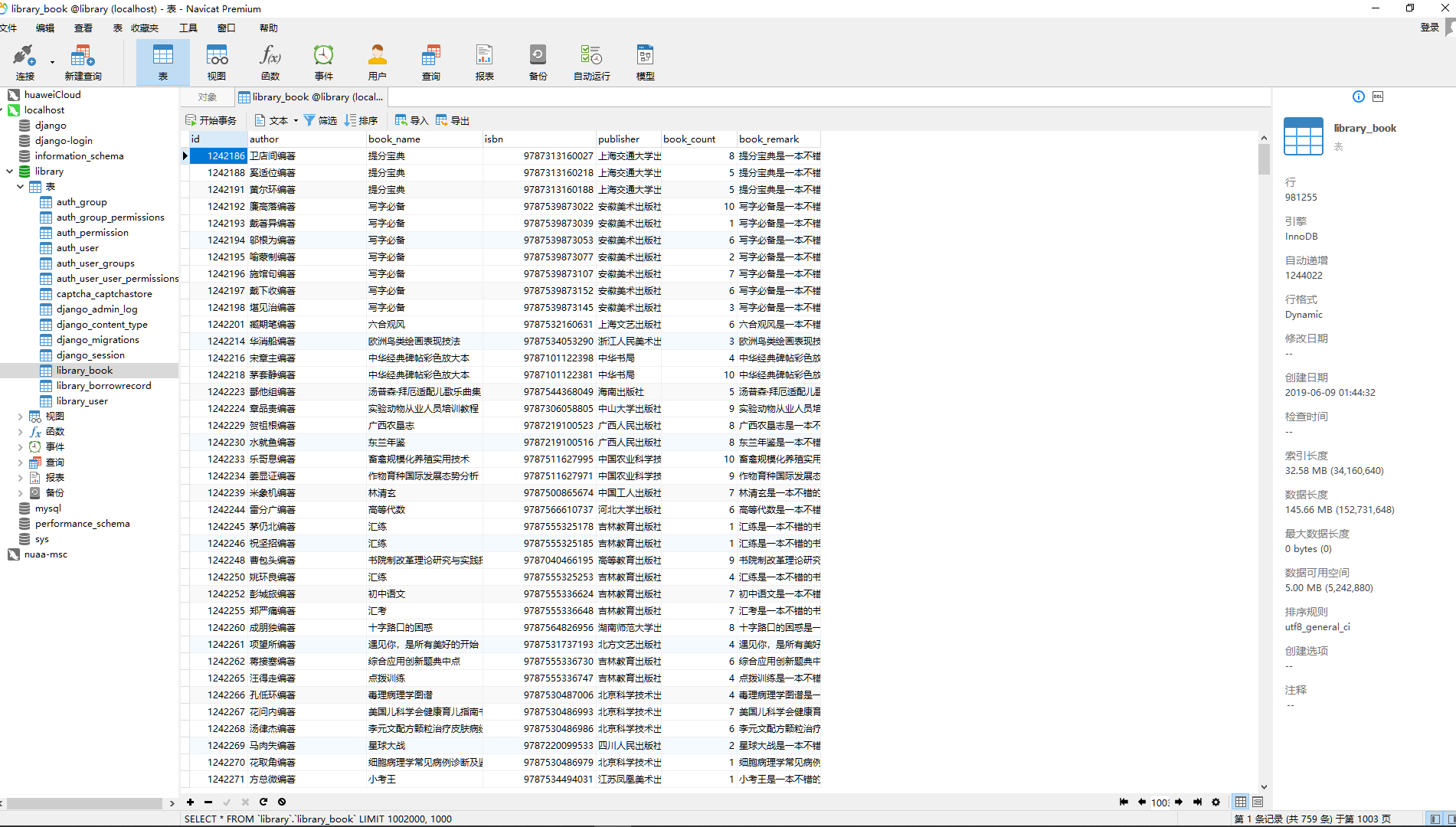


图4 借阅记录表设计

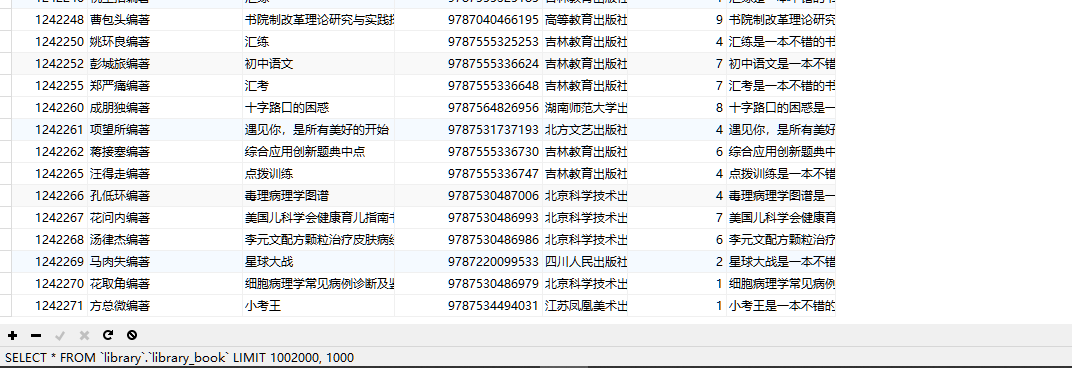


图4 借阅记录表设计



图4 借阅记录表设计

为了解决100w的数据量问题，我在中国图书出版数据库(网址：[http://www.cnpub.com.cn) 上爬取了1002759](http://www.cnpub.com.cn)上爬取了1002759)条数据，由于其网站收录图书条目仅有1368207条，并且有部分书籍缺少作者信息，所以我在缺少作者信息的单条数据中用姓名字典自动生成数据，最终生成了100w+数据。代码如下：

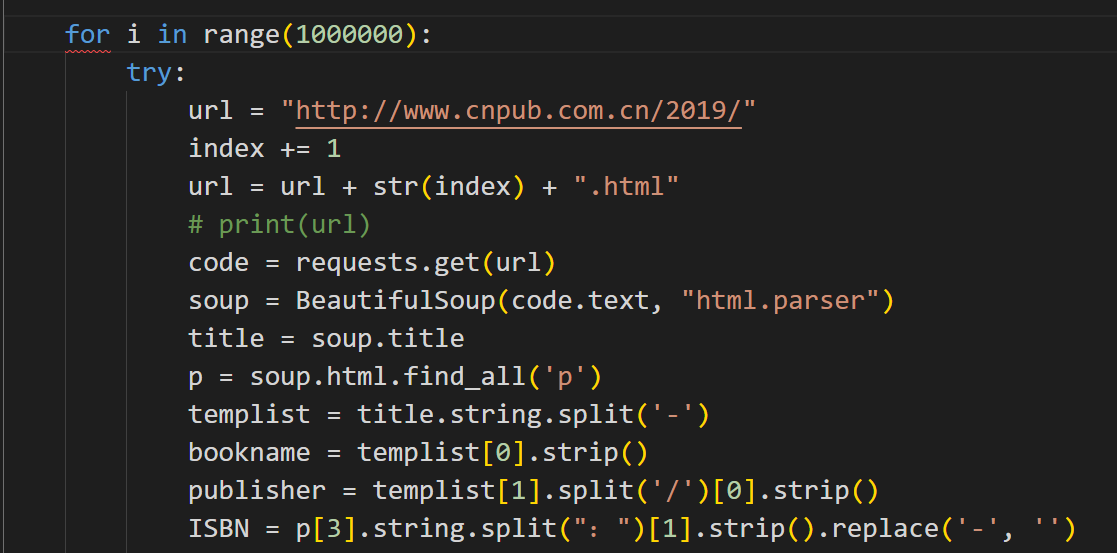


图4 借阅记录表设计



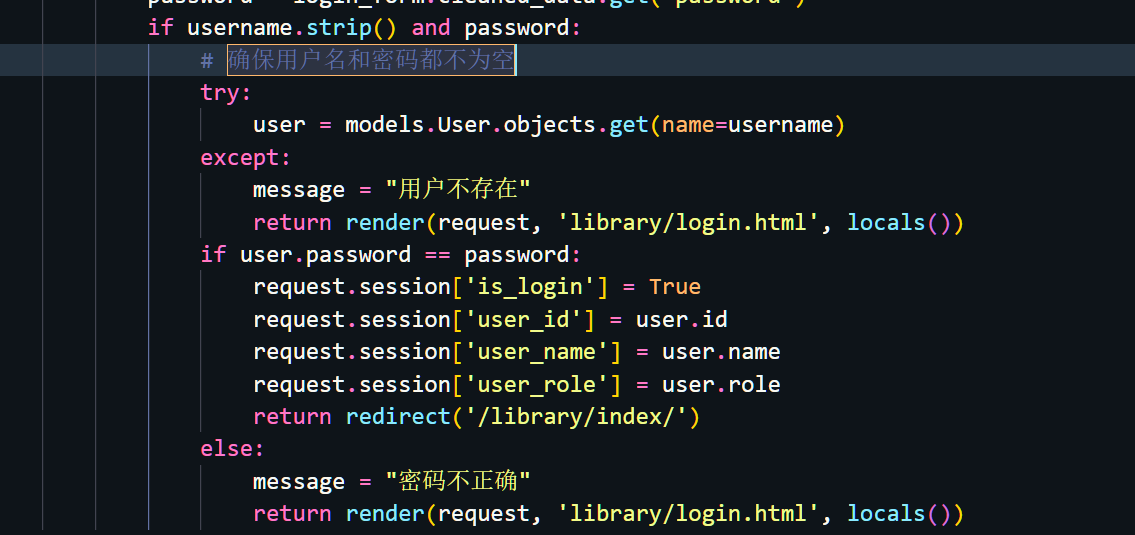
图4 借阅记录表设计



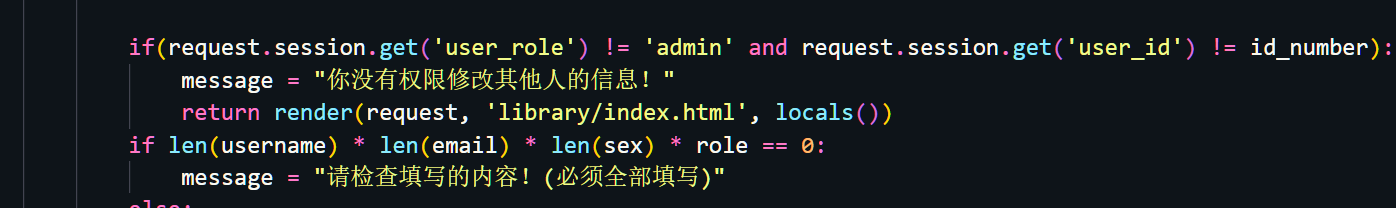
图4 借阅记录表设计

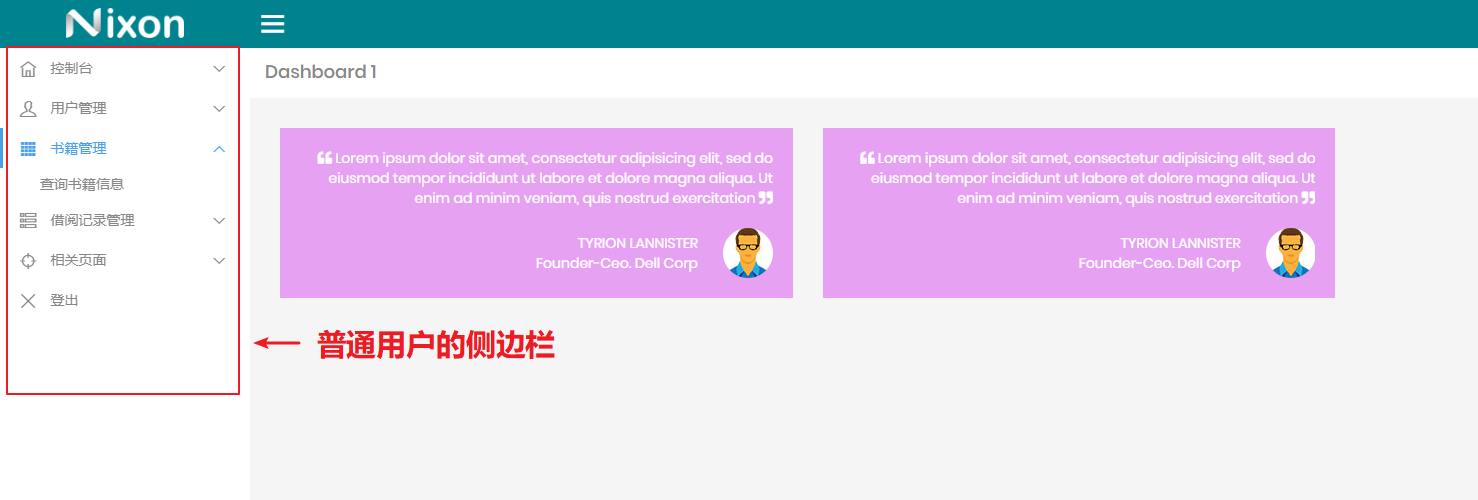
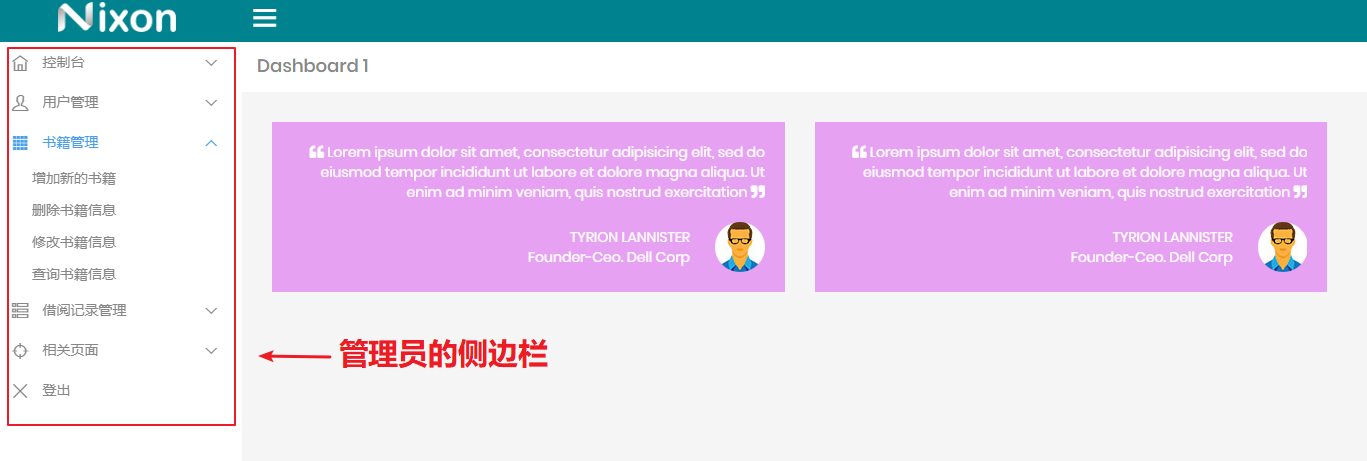
图1自动生成作者的代码片段，有姓与名两个列表的，若解析到的字符串不包含作者信息的话即自动生成。若存在作者信息，就依次解析出书名，出版社，ISBN码，作者，库存数量我生成了10以内的随机数，由于此网站并没有相关书评信息，所以书评信息我也就输入了固定的一串文字。这样大致爬取了5天的时间，中间改进了两三次代码，增加了异常判断和随机姓名提升了效率，最终爬取到了1002759条数据。

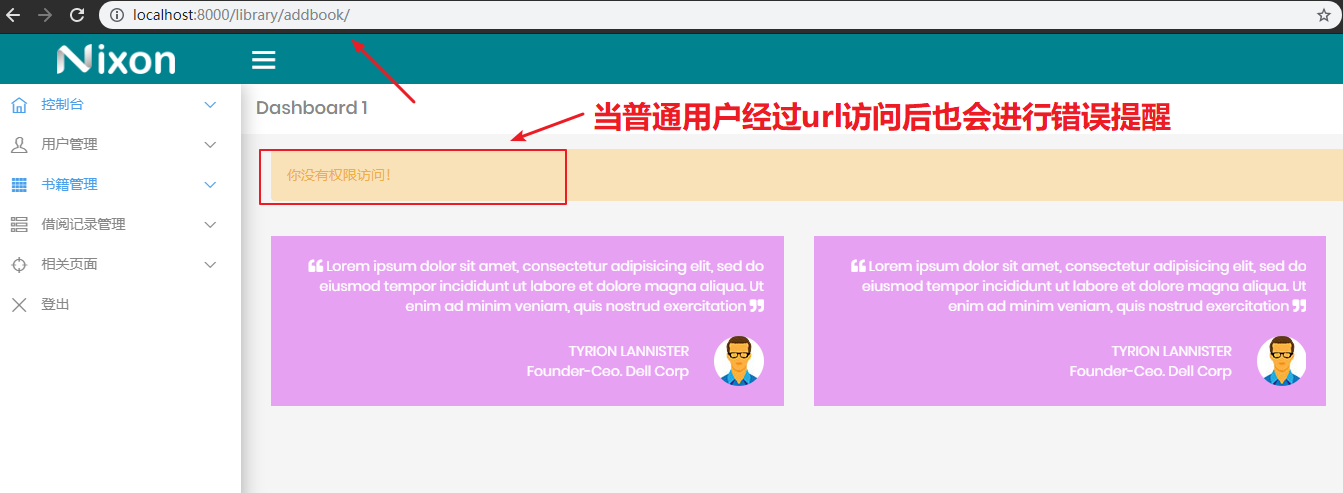
## 3.2 初级的权限控制

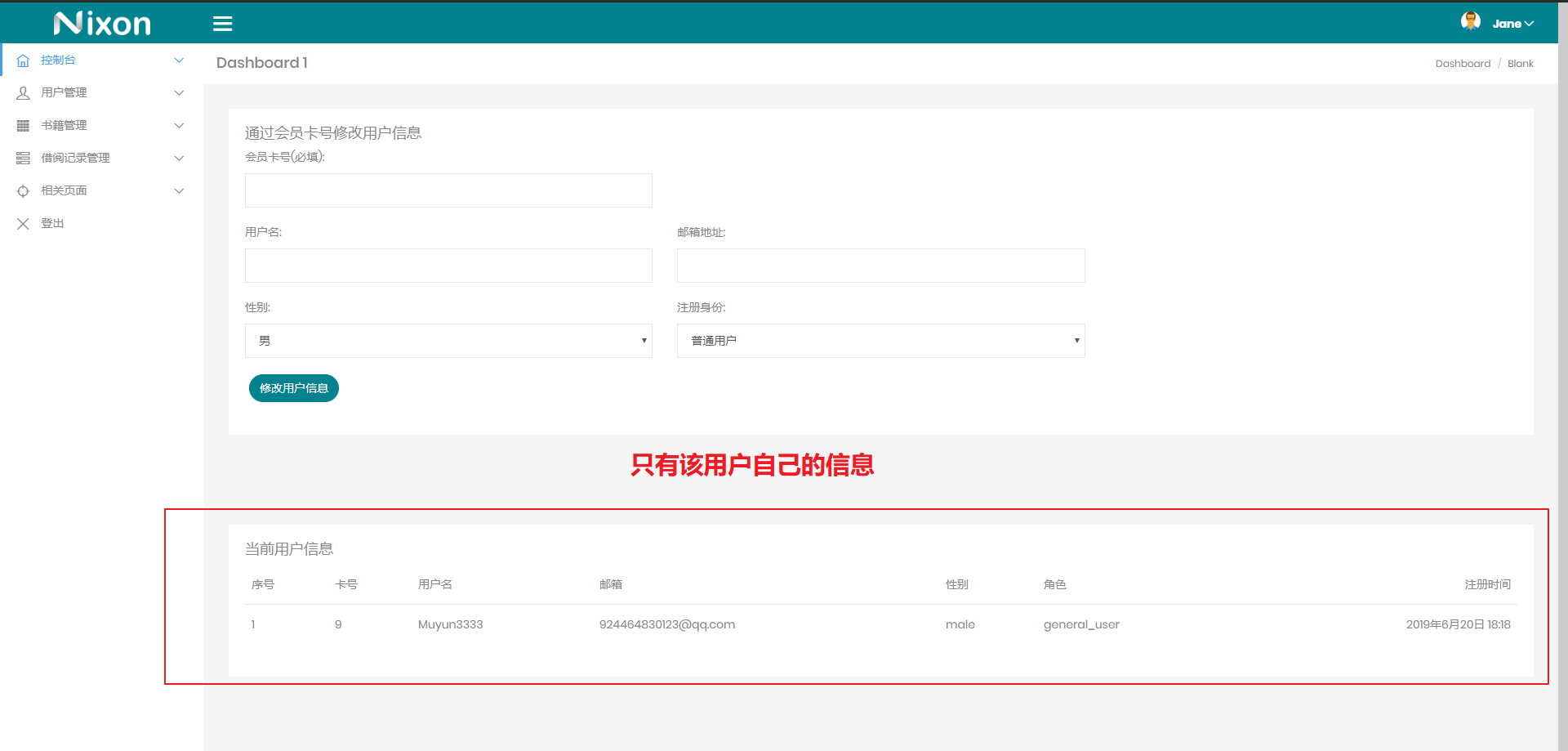










|  |  |  |
| --- | --- | --- |
| 功能页面 | 管理员权限 | 普通用户权限 |
| 增加用户信息 | 能够添加普通用户以及管理员 | 无权限 |
| 删除用户信息 | 能够注销除了*SuperUser*以外的所有人的账号 | 只能够注销自己的账号 |
| 修改用户信息 | 能够修改除了*SuperUser*以外的所有人的信息 | 只能够修改自己的信息 |
| 查询用户信息 | 能够查询所有用户的信息 | 只能够查询自己的信息 |
| 增加新的书籍 | 能够增加新的书籍(图书入库) | 无权限 |
| 删除新的书籍 | 能够删除新的书籍(图书出库) | 无权限 |
| 修改新的书籍 | 能够修改所有的书籍信息 | 无权限 |
| 查询新的书籍 | 能够查询所有的书籍信息 | 能够查询所有书籍信息 |
| 增加新的借阅记录 | 能够为任何用户借阅任意一本还有库存的书 | 只能为自己借阅任意一本有库存的书 |
| 删除新的借阅记录 | 能够为任何用户归还任意一本所借阅的书 | 只能自己归还自己所借阅过的书 |
| 修改新的借阅记录 | 能够为所有用户修改(延期)其借阅时间 | 可以为自己所借的书延长其借阅时间 |
| 查询新的借阅记录 | 能够查询所有的借阅记录 | 可以查询到自己的借阅记录 |

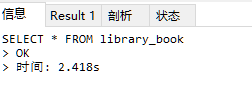
在初步制作网站时，我并没有想好如何设计权限控制。在对Django框架的学习中，我了解到其在登陆时你可以在request里维护一个session的数据结构，其应该是一个字典，所以我在用户登录时在session中存放了“用户是否登录”，“用户id”，“用户名”，“用户角色”的四个属性，并对视图中的其他函数也进行了session的判断。

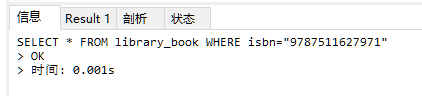
在设计权限控制时，我开始的想法是直接对前端进行改动，若其获取到session中的“用户角色”是普通用户，则隐藏其按键，但是经过思考发现这种十分基础的改动其实并不安全。由于我是制作的网页，用户大可通过暴力破解的方法来破解你的url从而进入到你的相关页面中去，例如我是一个普通用户，本应无权限查看所有用户的信息，但是我凭借直觉跳转到对应的url中去的话，是可以查看到敏感信息的，所以我又对其进行了相应的改进。

第二版的权限控制是这样设计的，由后端获取session中“用户角色”的信息，然后针对普通用户以及管理员做不同的跳转，在这个基础上，我实现了普通用户与管理员的分离并且也针对普通用户与管理员在一些表上的受限制程度不同也做了相应调整，使其变得更加合理，其中普通用户与管理员对表操作的权限我已经列在了表1中。通过后端来判断其用户角色再进行对应的权限控制有了更好的安全控制。不过在其中确是还是存在着一些安全问题，例如如果有用户经过一些手段能够修改session中的”用户身份”的信息的话，这又让整个系统陷入到了危险的境地，我有了一些想法但囿于时间原因还未来得及实现，我想我们保存的session中的值不应该是诸如”admin”这样直白的字符串信息，这样的信息容易被扫描并且容易被爬取，在借鉴了网易有道词典的策略后，我也想到加入随机性这样的技巧来提高整个系统的安全性。例如，我们可以在保存session的时候，加入当前时间以及一个随机数，然后再做一个md5码的加密，session中的信息都以密文的方式设置和验证，这样有助于提高整个系统的安全性。综上，在图1中我在我的图书管理系统中实现了初级的权限控制，具体效果如图2所示，普通用户无法看到无权限的button，并且其只对自己的相关信息进行操作，无法修改别人的信息，并且在其误入了相应页面时，也会进行”你没有相应权限”的错误提醒。

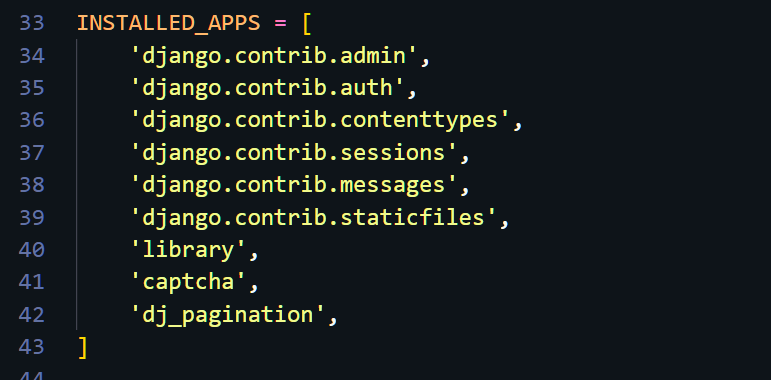
## 3.3 数据量过大导致的问题

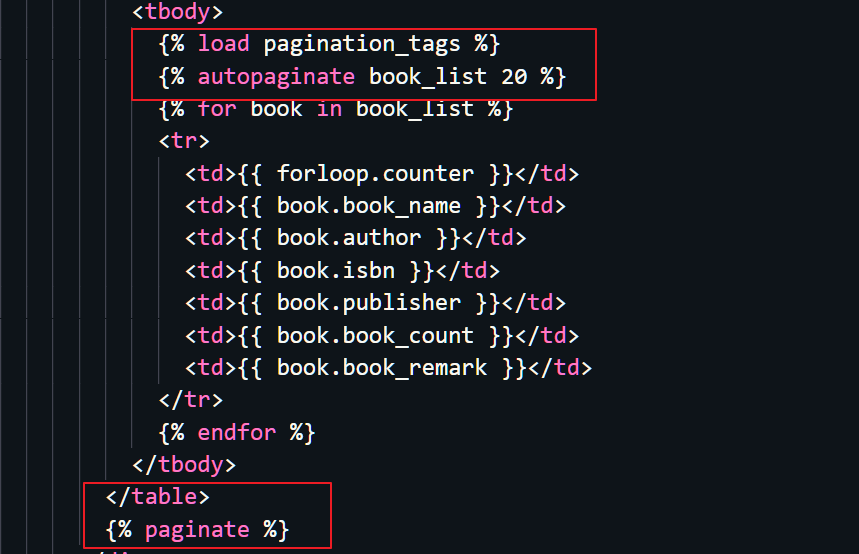
|  |  |  |
| --- | --- | --- |
| 功能 | SQL语句 | 响应时间 |
| 查询100万条数据并返回 | *SELECT \* FROM library\_book;* | 2.418s |
| 查询出版社是“贵州人民出版社”的图书信息 | *SELECT \* FROM library\_book WHERE publisher="贵州人民出版社"* | 0.837s |
| 查询isbn码是“9787511627971”的图书信息 | *SELECT \* FROM library\_book WHERE isbn="9787511627971"* | 0,001s |





在学习Django框架以及MySQL数据库的过程中，也有同学爬取或者自己生成了100万条以上的数据，但是在查询的过程中，他们会发现查询的效率及其低下，甚至在查询单条记录时都会出现5秒以上的延时，这对于用户体验来讲是灾难性的。在反复的测试以及实验中，我发现在拥有一定量的索引之后，其在100万条数据中的查询响应时间并不算久，我查询100万条数据的每一条数据，发现结果响应时间是2s多，这对用户来说是完全能够接受的，接下来我又测试了条件查询，分别返回一批记录以及单条记录，可以看到在条件足够精确的ISBN码查询中，0.001s的响应时间完全可以可以说是实时响应。所以问题到底出在哪里呢，经过我查询资料发现，前端页面的渲染也是需要时间的，也就是说，后端将查询结果返回时，前端取到数据有一定的延时，并且在前端渲染的时候也有一定的延时，在数据量过大的情况下，这会造成糟糕的用户体验。



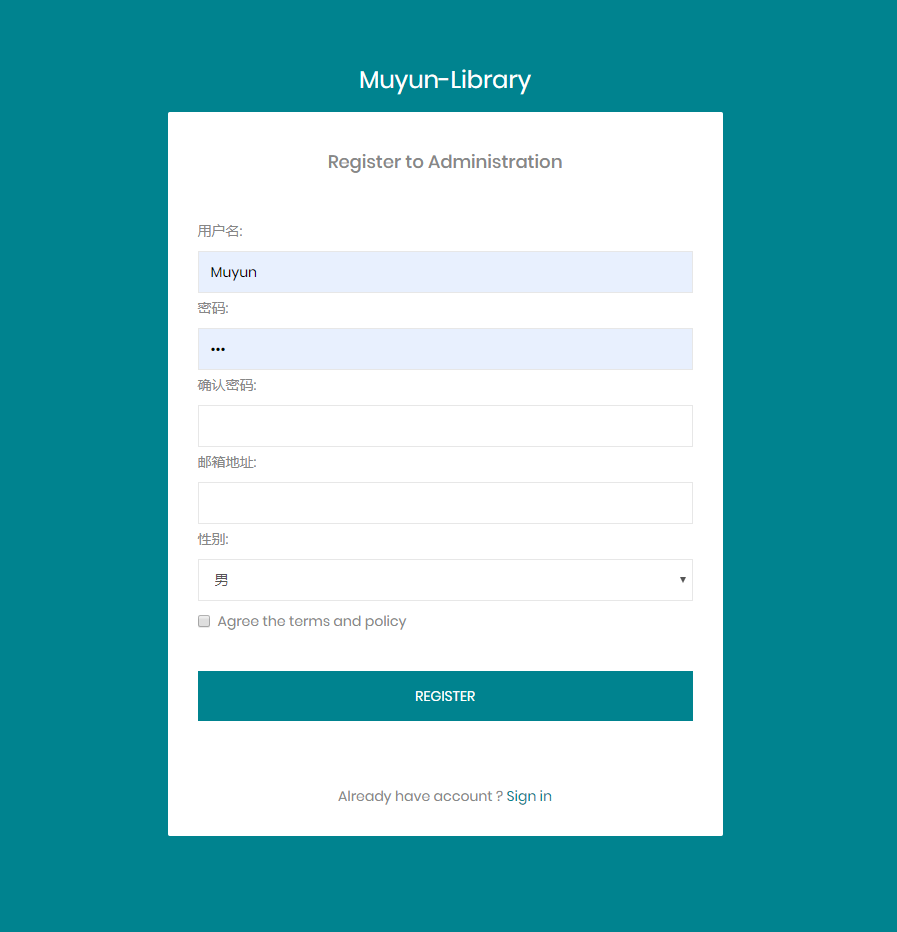


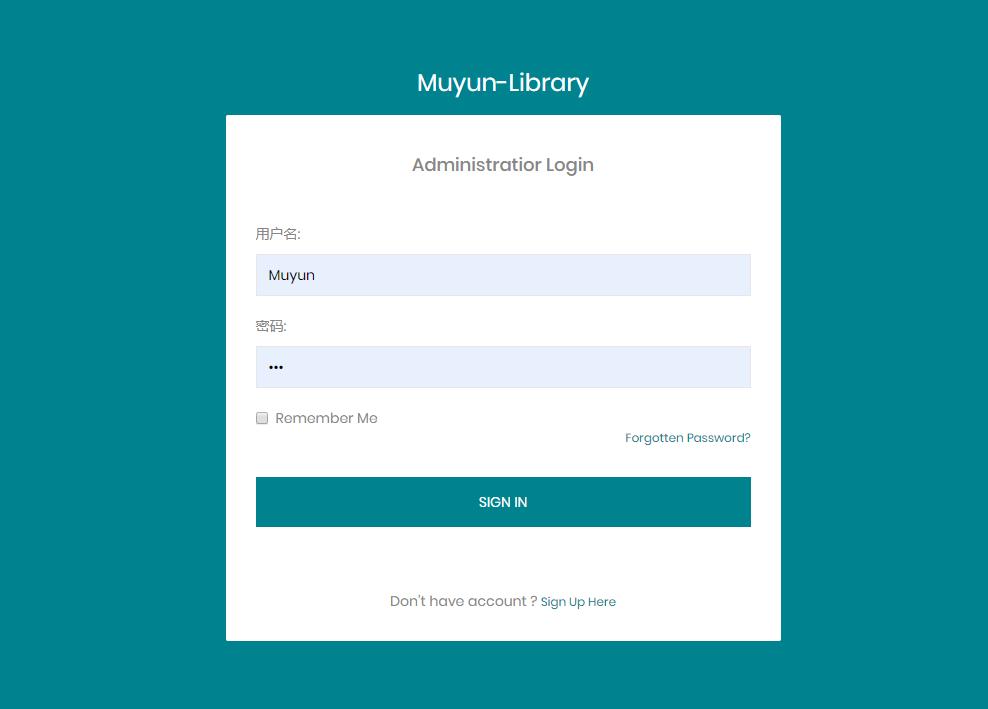


为了减少延时，我觉得减少前端渲染的时间是关键，单页既没有必要将所有查询结果反馈，也不应该以此来拖累性能。所以我采取了分页的方式来解决此问题。在这里我使用了Django自带的一个分页器Django-migration。首先我在Django的设置里安装了dj\_pagination，接着在html文件中load了此app然后对后端传来的列表进行每20条记录为一页的进行分割，在table标签后再加上了翻页的标签，至此，分页的功能完成。图1是分页的效果图，在查询信息时是完全能够接受的。在查阅相关资料后，我又有了一些新想法，在现在流行的一些系统中，是会有缓存数据的，例如我在查询删除书籍信息的时候，总是要后端请求查询所有的数据再到前端进行分页，但其实这一系列操作真正有价值的数据仅仅是首页的前20页罢了，而多余数据带来的开销会严重影响系统的延时以及用户的操作体验。在了解了Redis缓存数据库之后我产生了能不能将数据在第一次访问的时候就缓存到浏览器中，然后在请求相同数据时就不需要经过后端，大大降低了延时。但是这可能会涉及到安全问题，就是将数据缓存到本地会不会让没有权限的人也能通过某种手段来获取，其中涉及到的WEB安全知识较多，我还未去掌握。综上，利用分页减少前端渲染页面的时间，大大提升了用户体验，但是如何进一步提升，仍需学习。

# 4 图书管理系统功能简介

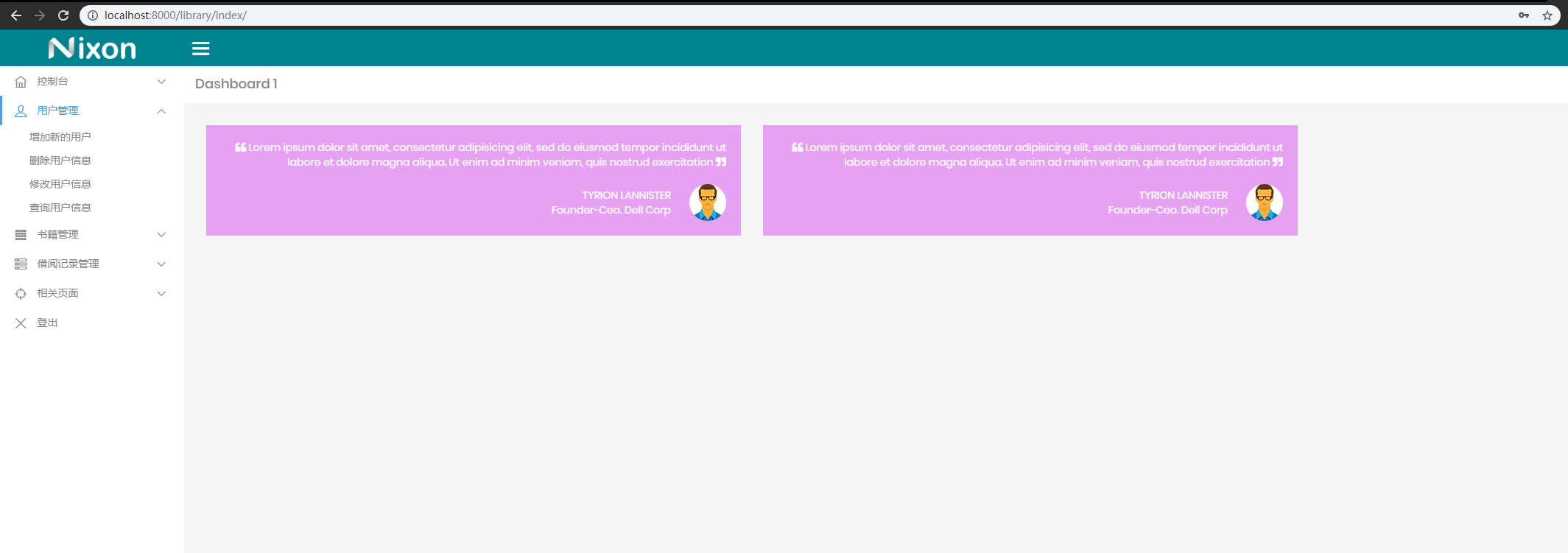
## 4.1 注册/登录模块





首先是注册登录模块，任何人都可以进行注册，但通过此渠道注册默认是普通用户，但可以通过管理员进行修改用户信息将管理员身份赋予给普通用户。此外管理员还能够直接在后台进行用户及管理员的增加。在注册页面中，用户名，用户注册的邮箱都是唯一不可重复的，并且用户密码需要输入两次才可被允许注册。在注册之后，会跳转到登录页面，在输入正确的用户名与密码后会保存所对应用户的session，session信息包括“用户是否登录”，“用户id”，“用户名”，“用户角色”的四个属性，用于之后的权限控制。成功登录之后会跳转到整个网站的主页。

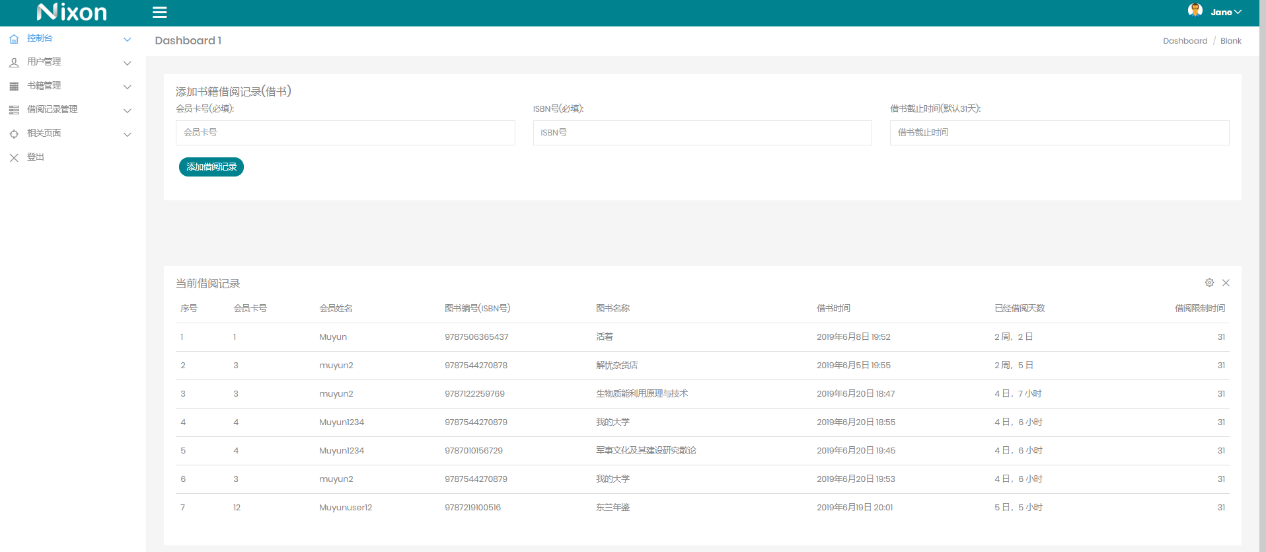
## 4.2网站主页



主页简介：主页左边有5个边栏，前三个边栏“用户管理”、“书籍管理”、“借阅记录管理”分别对应用户表、书籍表、借阅记录表的增删改查；“相关页面”链接到注册登录页面；“登出”一栏将清除登录的Session，并将页面重定向到登陆界面。

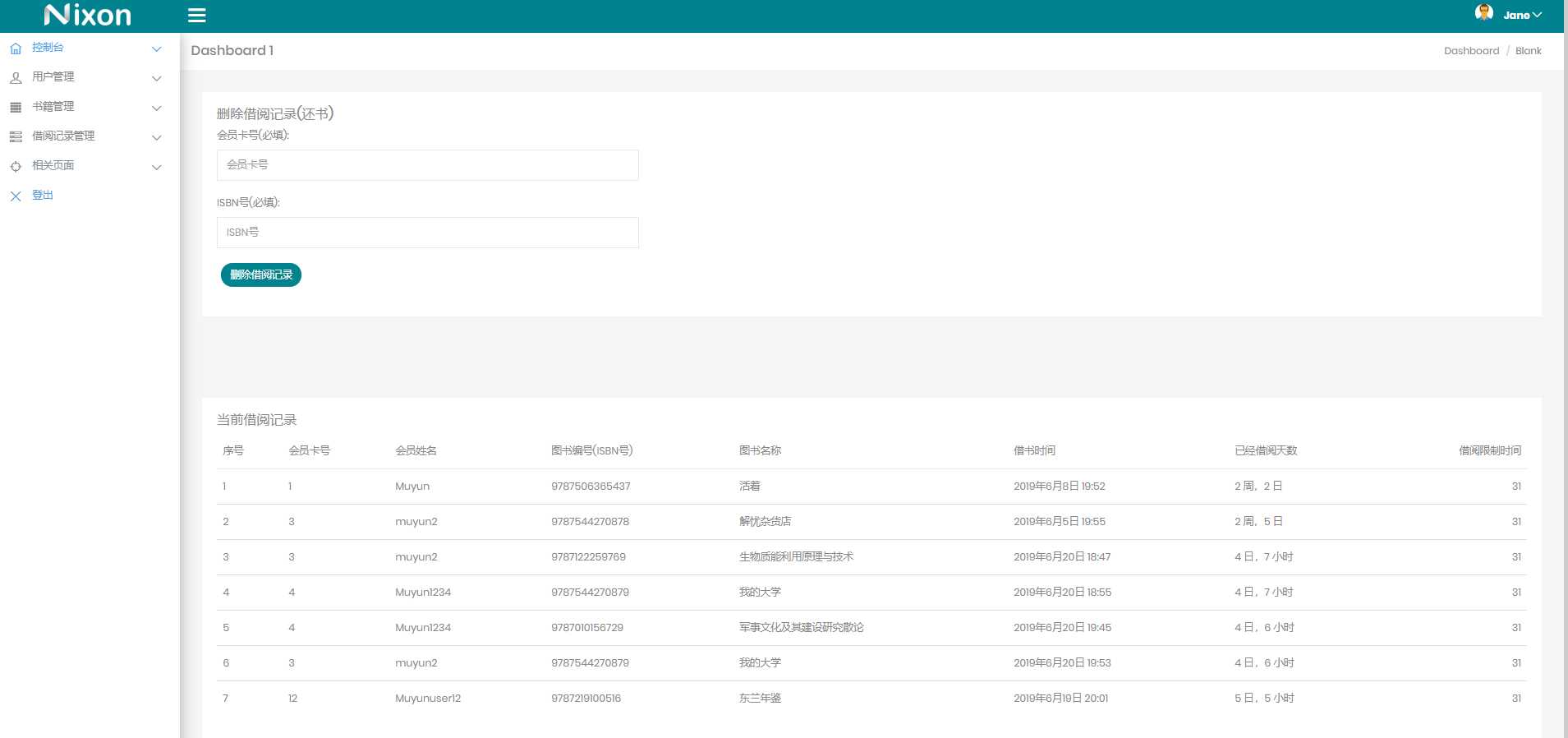
## 4.3 借书/还书/延长借阅时限模块

### 4.3.1借书功能



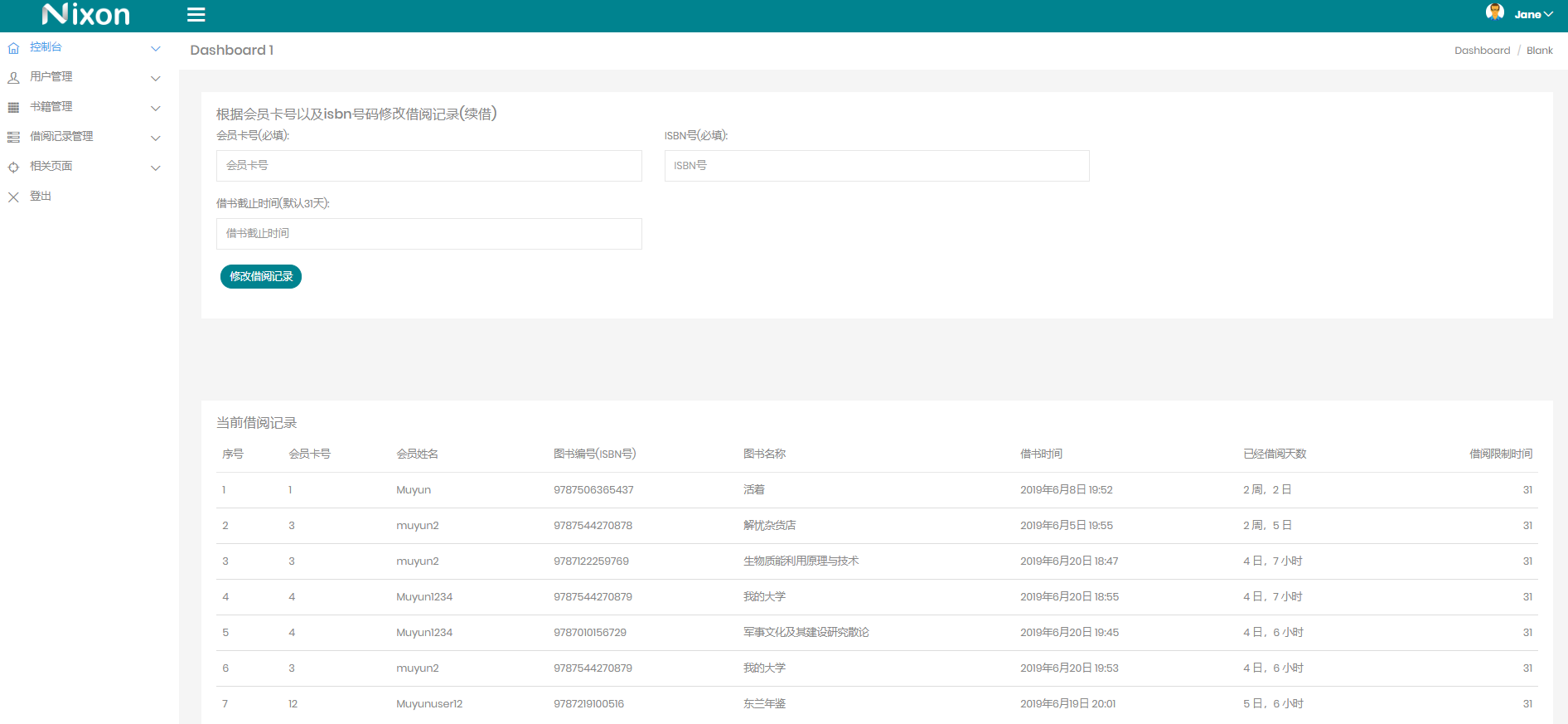
借书功能对应对借阅记录的增加，其中管理员能够为普通用户与管理员借书，而普通用户只能为自己借书。在借书操作正常完成之后，所借阅书籍的库存量会减少为1，当库存量为0时，借阅该书将会返回提示信息“该书库存为零”。借书功能必须需要填写两个信息，会员卡号以及ISBN码，可填的有借阅时限，其不填写的话默认为31天，此外，在借书时还将自动记录下当前的借阅时间，以供之后判断其借书是否超期。

### 4.3.2还书功能



还书功能对应借阅记录的删除，其中管理员能够给普通用户与管理员还书，而普通用户只能归还自己所借阅的书。在还书操作正常完成之后，所归还书籍的库存量会增加1。其中还书功能必须需要填写两个信息，会员卡号以及ISBN码。

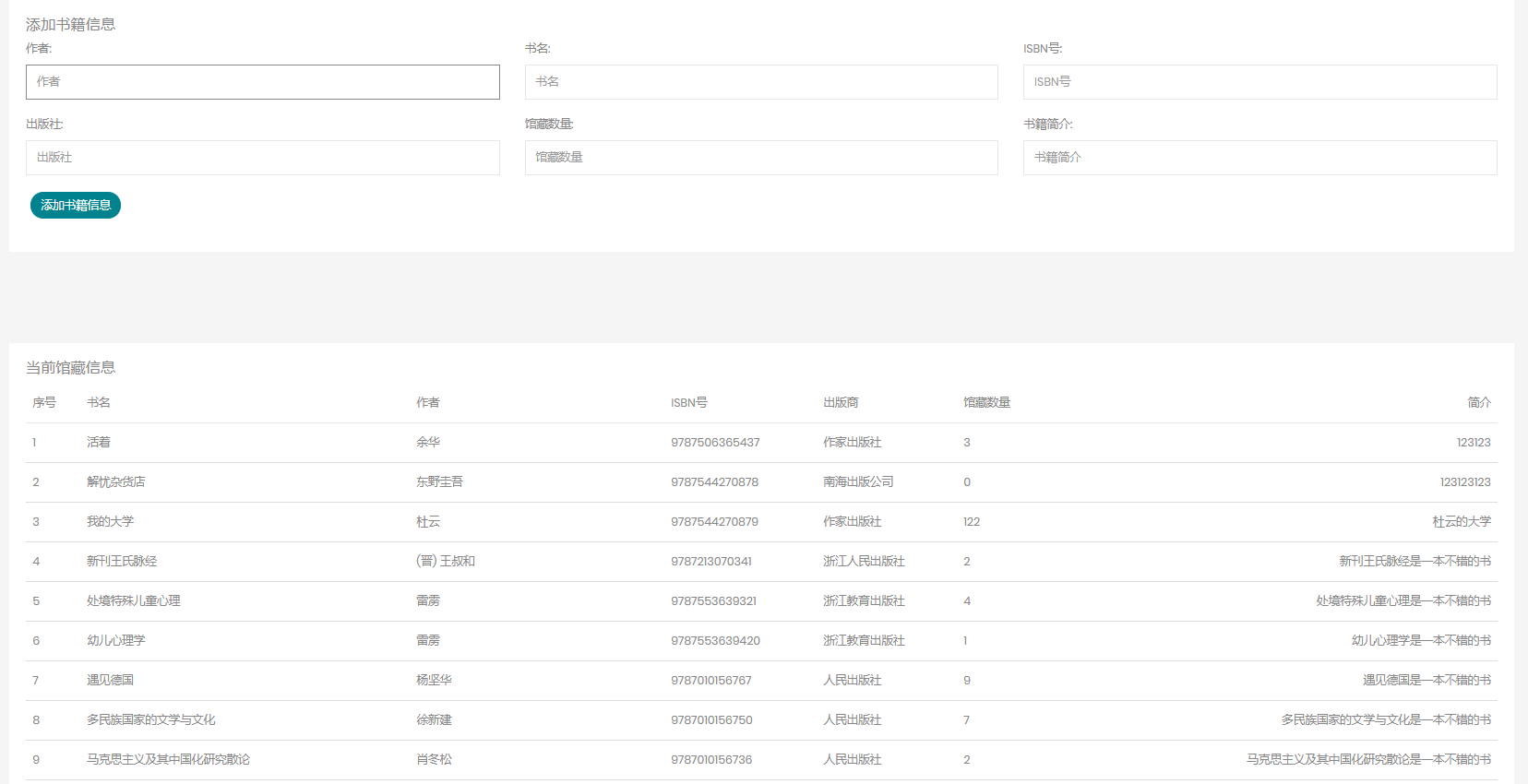
### 4.3.3延长借阅时限功能



延长借阅时限功能对应借阅记录的修改，其中管理员能够延长普通用户与管理员的借书时限，普通用户只能延长自己的借阅时限。在延长借阅时限操作正常完成之后，数据库中所对应的字段会被更新。由于我们的借书时间会自动存储，其会与当前的时间进行做差得到已经借阅的天数，这一信息不会存储到数据库中，只会在前端计算后直接进行显示。

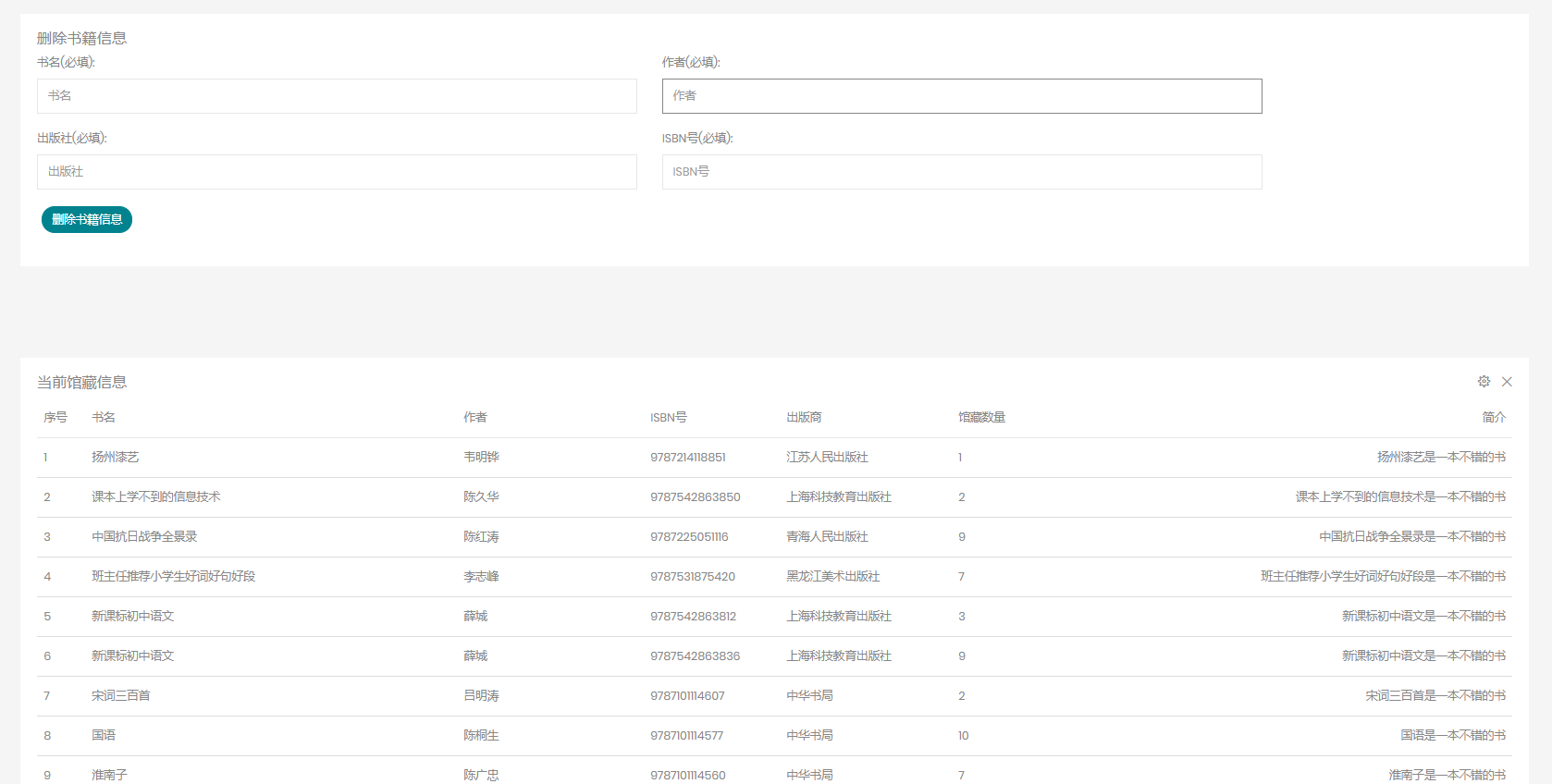
## 4.4 图书入库/出库模块

### 4.4.1图书入库功能



图书入库功能对应的是书籍信息的添加，书籍信息的添加必须填写的信息包括该书籍的作者、书名、ISBN码、出版社、馆藏数量以及书籍简介。图书表的主键是ISBN码，由于其是真实数据，在现实世界中也是保持着其唯一性。所以只要ISBN码不相同的书籍都可以成功入库，若存在ISBN码相同的书籍，会返回提示信息“该书已在书库中”，然后去修改图书的馆藏数量即可。

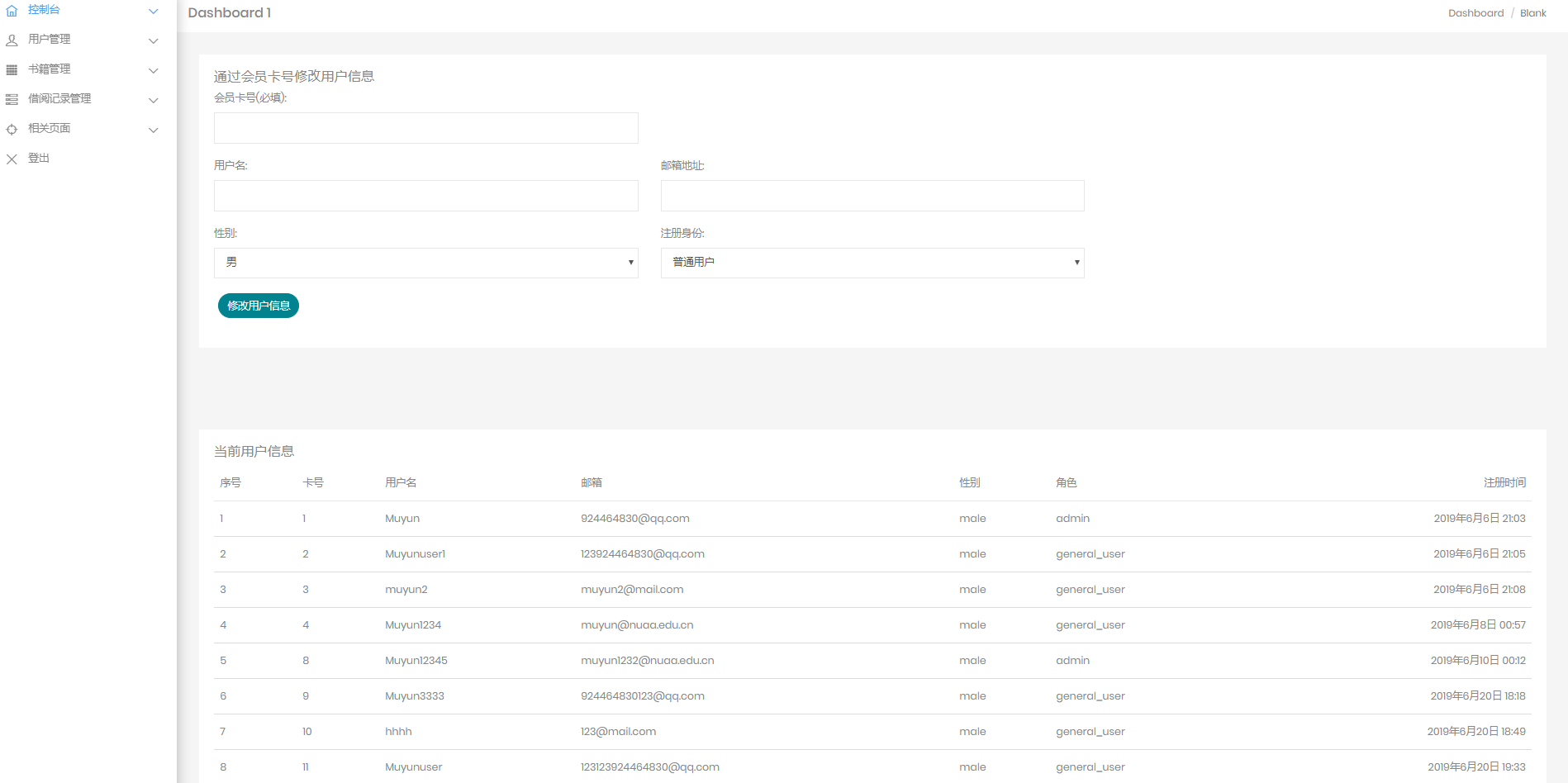
### 4.4.2图书出库功能



图书出库功能对应的是书籍信息的删除，书籍信息的删除必须填写的信息包括该书籍的作者、书名、ISBN码、出版社。图书表的主键是ISBN码，由于其是真实数据，在现实世界中也是保持着其唯一性。在图书出库后该图书的ISBN码也将唯一的删除。

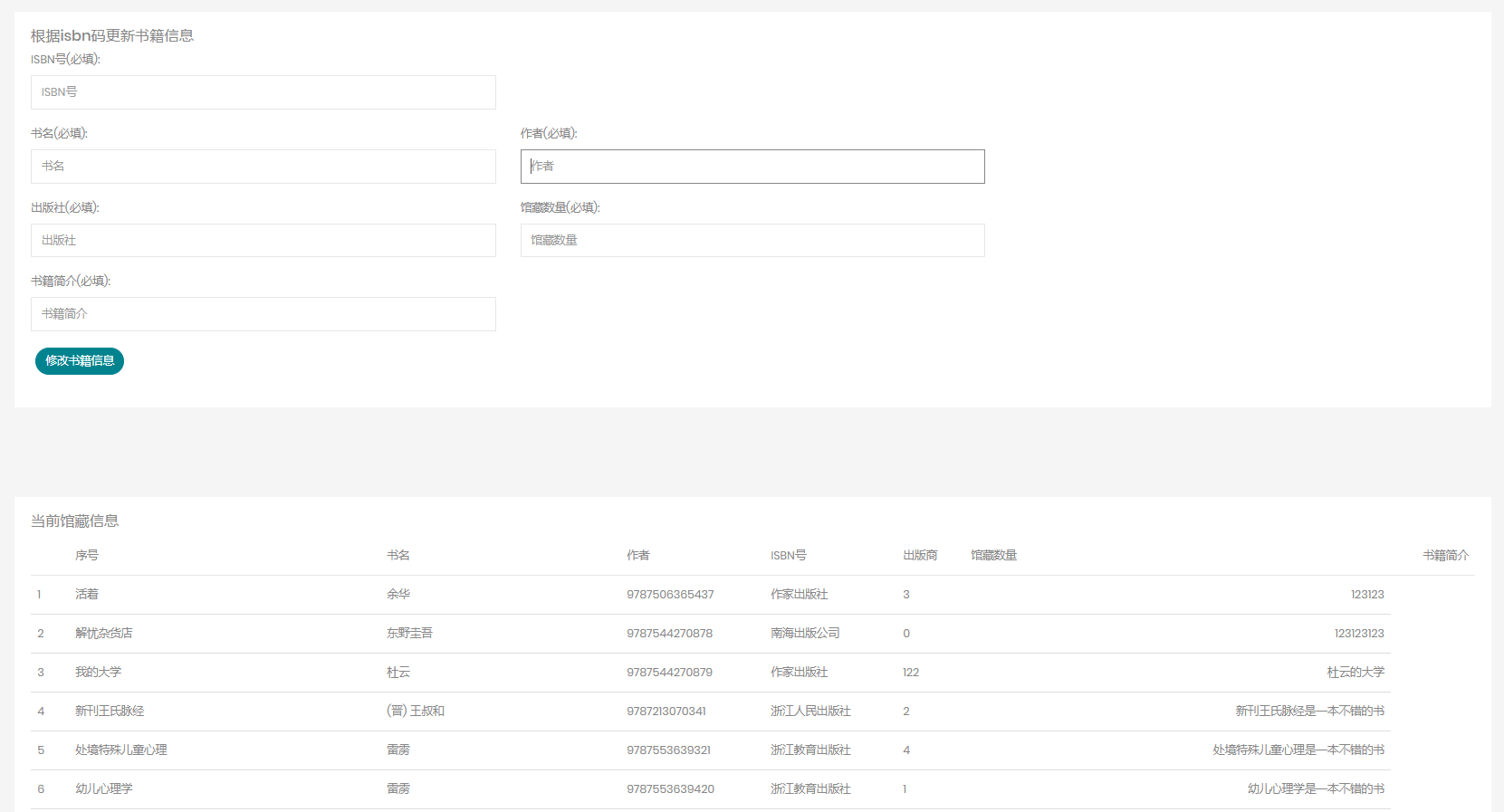
## 4.5 信息修改功能

### 4.5.1用户信息修改功能



用户信息修改功能对应“用户管理”栏的“修改用户信息”的选项，通过输入用户表的主键会员卡号来获得唯一的一条记录。然后再填写用户名、邮箱地址、性别和身份进行修改。管理员可以修改普通用户的信息，也能够赋予普通用户以管理员的权限。但普通用户不需要输入自己的会员卡号即可修改自己的信息，但其修改的信息只有用户名、邮箱地址以及性别，普通用户无权限修改自己的身份。

### 4.5.1图书信息修改功能

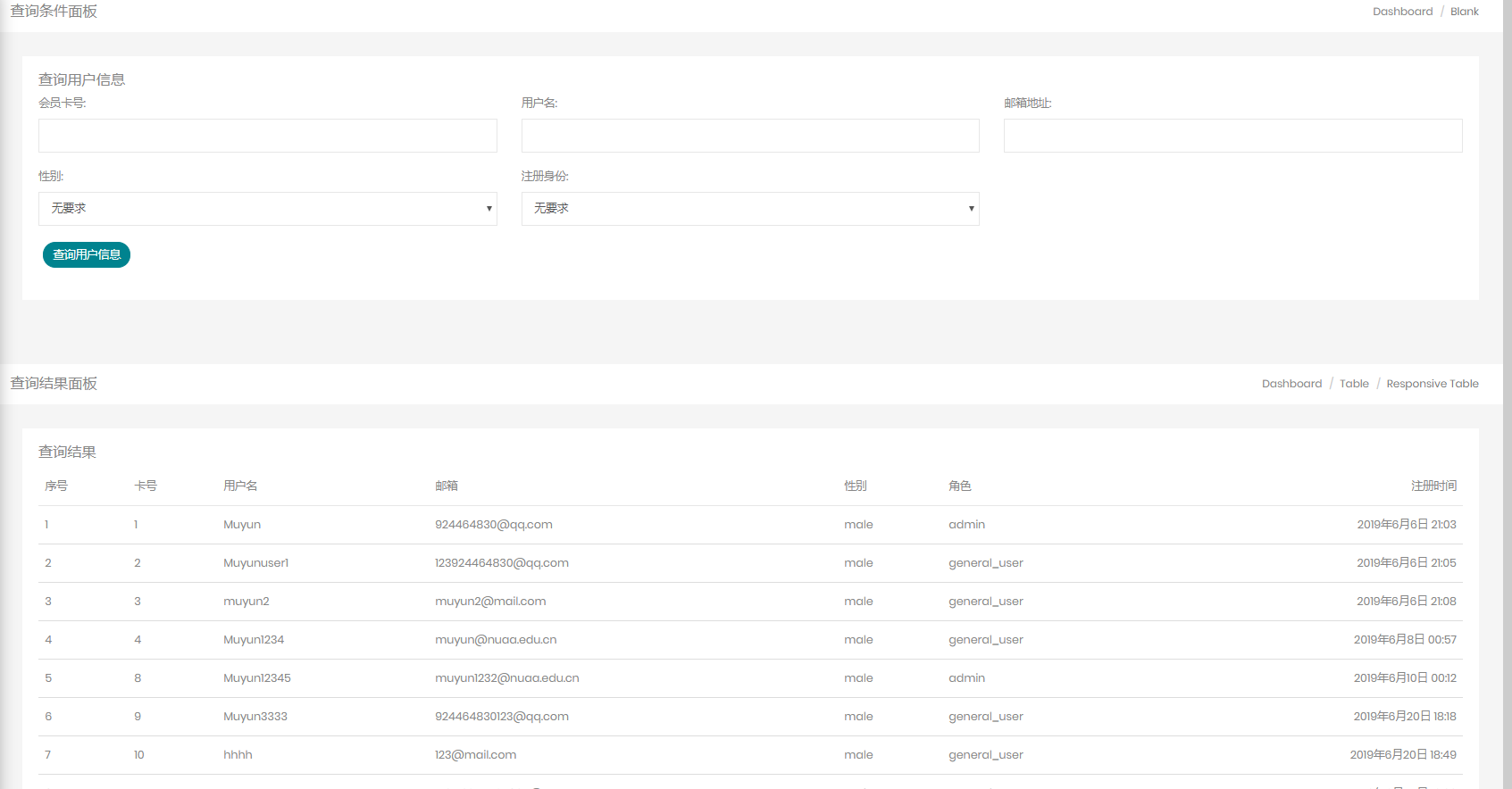


图书信息修改功能对应的是“书籍管理”一栏中的“修改书籍信息”的选项。通过输入图书表的主键ISBN码来获得唯一的一条记录。然后再填写书名、作者、出版社、书籍简介以及馆藏数量进行修改。管理员在进行书籍馆藏数量的修改时就相当于是增加书库中已有书籍的数量，而不是对新图书进行入库，这也符合现实世界的逻辑。普通用户无权限修改图书的信息。

## 4.6 模糊查询功能

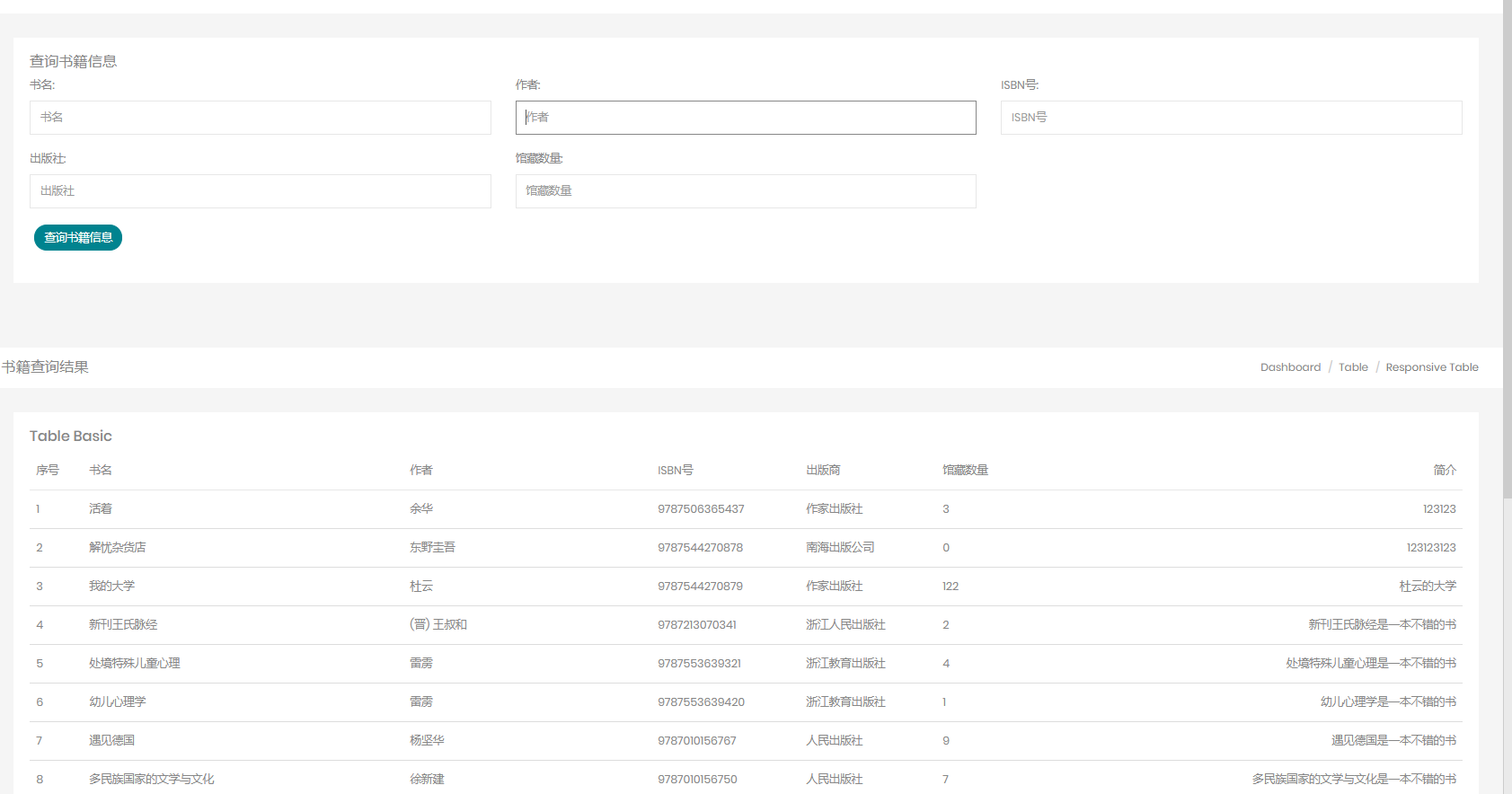
在左边三栏”用户管理”、“书籍管理”、“借阅记录管理中都有对各自表的查询。在三个查询的地方我都使用了模糊查询。

### 4.6.1用户表的模糊查询



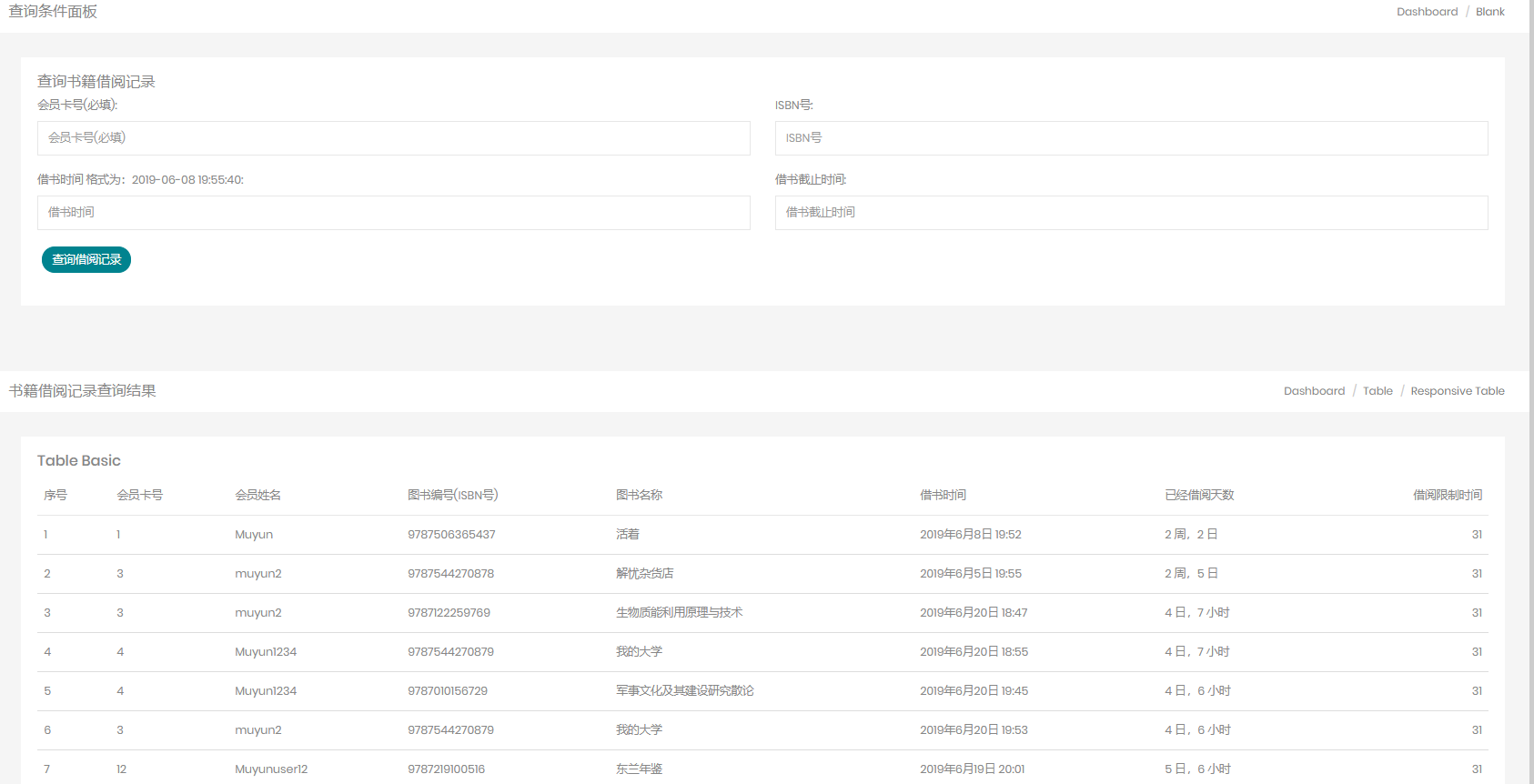
在用户表的模糊查询上，你若每个字段不填写的话将显示全部用户信息，相应的，你可以填写其中的部分信息，下面的查询结果面板将显示满足你所输入条件的一批用户信息。

### 4.6.2图书表的模糊查询



在图书表的模糊查询上，你若每个字段不填写的话将显示全部图书的信息。相应的，你也可以填写其中的部分信息(例如出版社信息)，下面的查询结果面板将显示满足你所输入条件的图书信息。

### 4.6.3借阅记录表的模糊查询



在借阅记录表的模糊查询上，你若每个字段不填写的话将显示全部图书的信息。相应的，你也可以填写其中的部分信息(例如ISBN码信息)，当你填写部分字段信息时，一定需要填写会员卡号的信息，之后下面的查询结果面板将显示满足你所输入条件的一批借阅记录信息。