五、详细设计说明书

[1．引言 1](#_Toc506973482)

[1.1编写目的 1](#_Toc506973483)

[1.2项目背景 1](#_Toc506973484)

[1.3定义 2](#_Toc506973485)

[1.4参考资料 2](#_Toc506973486)

[2．总体设计 2](#_Toc506973487)

[2.1需求概述 2](#_Toc506973488)

[2.2软件结构 2](#_Toc506973489)

[3．程序描述 2](#_Toc506973490)

[3.1功能 3](#_Toc506973491)

[3.2性能 3](#_Toc506973492)

[3.3输入项目 3](#_Toc506973493)

[3.4输出项目 3](#_Toc506973494)

[3.5算法 3](#_Toc506973495)

[3.6程序逻辑 3](#_Toc506973496)

[3.7接口 3](#_Toc506973497)

[3.8存储分配 3](#_Toc506973498)

[3.9限制条件 3](#_Toc506973499)

[3.10测试要点 3](#_Toc506973500)

# 1．引言

## 1.1编写目的

本文档为图书管理系统的详细设计说明书，向阅读人员和图书馆管理人员介绍本图书管理系统的框架和详细设计。编写这份详细设计说明书的目的在于明确系统范围，明确图书管理系统流程并规范化记录该系统需求和非功能性系统需求。

开发团队，包括项目经理、文档经理、技术经理和测试经理项目开发参考使用。

其他相关用户，了解系统需求范围和实现目标，目的在于更好的使用系统

## 1.2项目背景

校园图书馆是高等院校的重要组成部门，是教师和学生获取知识的重要场所。由于图书馆主要从事大量的图书资料的储存和流通。所以一直以来，计算机在图书管理中得到了广泛应用。

目前，本校学生数量较多，教师人数其次，同时借书数量一直保持较高记录。为了解决工作人员图书管理时繁琐复杂，读者借阅图书时效率不高等相关问题，本系统主要就教师和学生的日常借阅、查询、续借、归还图书的流程和图书管理人员的管理图书的工作加以简化，以提高效率，减轻图书馆工作人员的工作压力，更好的发挥高校图书馆的潜在价值。

同时，我们也结合校园图书馆的资金投入有限，图书需求量大等特点，对图书管理中各业务环节加强管理。所以系统在实现时，实现传统的验收、编目、典藏、流通和管理等环节。

## 1.3定义

**Pentium:**英特尔奔腾处理器

**Android:**谷歌公司开发的操作系统

**IOS:**苹果公司开发的操作系统

**RS232:**常用的串行通信接口标准之一

**RJ45:**布线系统中信息插座（即通信引出端）连接器的一种,公用电信网络的接口

**索书号:**是图书馆藏书排架用的编码，是文献外借和馆藏清点的主要依据,一般由分行排列的几组号码组成,一个索书号只能代表一种书

**中图分类号**:中图分类号，是指采用《中国图书馆分类法》对科技文献进行主题分析，并依照文献内容的学科属性和特征，分门别类地组织文献，所获取的分类代号

**ISBN**:国际标准书号（International Standard Book Number）,是专门为识别图书等文献而设计的国际编号，由13位数字组成，分为5段，即在原来的10位数字前加上3位EAN（欧洲商品编号）图书产品代码“978”

**模糊查询**：系统允许被搜索信息和搜索提问之间存在一定的差异，这种差异就是“模糊”在搜索中的含义

**分布式存储**：分布式存储是一种数据存储技术，通过网络使用企业中的每台机器上的磁盘空间，并将这些分散的存储资源构成一个虚拟的存储设备，数据分散的存储在企业的各个角落

**B+树**：一种树数据结构，通常用于[数据库](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93" \t "https://baike.baidu.com/item/_blank)和[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/_blank)的[文件系统](https://baike.baidu.com/item/%E6%96%87%E4%BB%B6%E7%B3%BB%E7%BB%9F" \t "https://baike.baidu.com/item/_blank)中。B+树的特点是能够保持数据稳定有序，其插入与修改拥有较稳定的对数时间复杂度

**文件索引系统**：在文件中随机存取记录，需要知道记录的地址

**时间复杂度**：在[计算机科学](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6" \t "https://baike.baidu.com/item/_blank)中，时间复杂性，又称时间复杂度，[算法](https://baike.baidu.com/item/%E7%AE%97%E6%B3%95" \t "https://baike.baidu.com/item/_blank)的时间复杂度是一个[函数](https://baike.baidu.com/item/%E5%87%BD%E6%95%B0" \t "https://baike.baidu.com/item/_blank)，它定性描述该算法的运行时间

**unicode码**：[Unicode](https://baike.baidu.com/item/Unicode" \t "https://baike.baidu.com/item/Unicode%E7%A0%81/_blank)是一个编码方案，Unicode 是为了解决传统的字符编码方案的局限而产生的，它为每种语言中的每个字符设定了统一并且唯一的[二进制](https://baike.baidu.com/item/%E4%BA%8C%E8%BF%9B%E5%88%B6" \t "https://baike.baidu.com/item/Unicode%E7%A0%81/_blank)编码，以满足跨语言、跨平台进行文本转换、处理的要求

**哈希**：把任意长度的[输入](https://baike.baidu.com/item/%E8%BE%93%E5%85%A5/5481954" \t "https://baike.baidu.com/item/Hash/_blank)通过散列算法变换成固定长度的[输出](https://baike.baidu.com/item/%E8%BE%93%E5%87%BA/11056752" \t "https://baike.baidu.com/item/Hash/_blank)，该输出就是散列值

**提要**：摘出要领

**全列匹配**：使用数据的所有信息进行匹配

**最左前缀匹配**：使用信息中能够匹配的最长的左前缀进行匹配

**分库分表**：将数据分散到多个数据库和数据表中进行存储

**数据一致性**：指关联数据之间的逻辑关系是否正确和完整。而数据存储的一致性模型则可以认为是存储系统和数据使用者之间的一种约定

**数据错乱**：数据一致性被破坏，出现不一致的问题

**数据回滚**：指的是程序或数据处理错误，将程序或数据恢复到上一次正确状态的行为。回滚包括程序回滚和数据回滚等类型

**缓存**：访问更快的高速存储器

**操作日志**：系统所有操作的记录

## 1.4参考资料

**图书管理系统详细设计说明：**

<https://blog.csdn.net/m18580239676/article/details/48367949/>

<https://wenku.baidu.com/view/609920a18662caaedd3383c4bb4cf7ec4afeb668.html>

<https://www.co188.com/jh/t30643.html>

**之前的需求分析分档，项目合同文档，项目建议书，总体设计书**

# 2．总体设计

## 2.1需求概述

1.对软硬件相关需求：

**（1）服务器：**

服务器用作后台的数据处理，需求如下

处理器（CPU）：Pentium 2G 单核以上

内存容量（RAM）：2G以上

**（2）PC客户端：**

PC客户端的硬件需求如下

处理器（CPU）：Pentium 1G或更高

内存容量（RAM）：256M或更高

硬盘空间：512MB 剩余空间及以上

**（3）手机客户端：**

手机客户端的硬件需求如下

系统要求：Android 5.0 | IOS 10.0 及以上

内存容量（RAM）：256M或更高

硬盘空间：512MB 剩余空间及以上

**（4）自助借还机：**

自助借还机用于在图书馆内进行图书借还操作。硬件需求如下

工作频率：13.56 MHz

响应速度：不少于每秒8个标签

阅读范围：确保250mm及250mm范围以内为有效阅读区域（半径）

通信接口：USB、RS232、RJ45

防冲突：≥30个标签

主机：采用触摸式屏工业电脑

2.功能相关需求：

**（1）**能够存储一定数量的图书信息，并方便有效的进行相应的书籍数据操作管理。

**（2）**能够对一定数量的读者进行相应的信息存储与管理

**（3）**能够对需要的系统结果提供列表显示输出。

**（4）**能够提供一定的安全机制，提供数据信息授权访问，防止随意删改，同时提供备份的服务。

## 2.2软件结构

# 软件结构3．程序描述

**查询模块：**

## 3.1 功能

从系统数据库中查找出对应或相关的书籍信息，共分为模糊查询和精确查询。

模糊查找的功能为在查询页输入书名，查找符合条件的所有书籍的大致信息，并可设置排序方式。

精确查询为进入某本书的详情页面时，查找该本书的所有具体信息。

## 3.2 性能

对于模糊查询：大多数数据库都支持模糊查询，在MySQL关系型数据库中建立索引后，对于一百万量级的数据进行英文模糊匹配用时大约为3.596秒、进行中文模糊匹配用时大约为3.773秒；对于四百万量级的数据进行模糊匹配需要的时间则高达13.335秒和14.511秒。因此对于模糊匹配需要进行分布式存储等性能上的优化才能满足定量的需求。

对于精确查询：若在数据库中对每本书建立哈希索引可大大提高性能，大多数数据库底层使用B+树的数据结构作为文件索引系统。当数据量为百万级时，单次精确查询的时间复杂度平均为log(n)，并与树的度成正比。假设每一棵B+树有十个分支，那么单词查找需要6次左右查询。性能可以满足大规模高并发的业务需求。

## 3.3 输入项目

1. 书名（小于64个unicode字符，且不包含特殊符号）

2. 每本书的唯一标识符（一段哈希值，通常不是由用户进行输入）

## 3.4 输出项目

所有匹配书籍信息的数组（包含书籍编号、书名、作者、ISBN号码、出版社、价格、版次、提要、书籍图片路径、书籍状态、入库时间、删除时间）

## 3.5 算法

模糊查询和精确查询的算法本质都是利用了数据库底层建立索引所使用的B+树。对于模糊查询即匹配某列的前缀字符串；对于精确查询则使用的是全列匹配或最左前缀匹配。

**3.6 程序逻辑**

## 

## 3.7接口

1. 查询书籍是否存在

Url：/isExistBook

方式：get

参数：id=书籍Id

返回值:{result:True/False}

1. 模糊查询相关书籍，返回相关书籍ID

Url：/getBookIdByKey

方式：post

参数：key=关键字

返回值:

不存在返回 {result:False}

存在返回

{result:True,

Ids:[1,2,3,4,5]

}

1. 查询书籍详细信息

Url：/getBookByID

方式：post

参数：id=书籍id

返回值:

{result:True,

Book:

{id:书籍Id,

name:书籍名,

author:作者名,

ISBN:isbn号,

publisher:出版社,

version:书籍版次,

typeId:中图法分类号,

summary:书籍提要,

cover:书籍图片路径,

state:书籍状态,

entry\_time:入库时间,

delete:删除时间,},

}

1. 查询书籍状态

Url：/getBookStateByID

方式：get

参数：id=书籍id

返回值:

{result:True,

state:(free/checkout/otherwhere)}

## 3.8存储分配

1. 书籍信息表用于存储所有同一本图书的基本信息，主要是通过检索该表来实现书籍信息的查询
2. 图书分类表以中图分类法为基础，存储了对图书分类的信息。

将所有的信息存储在mysql数据库中，同时将查询过的数据存储在缓存redis中，保证高并发下的快速读取，当后台数据量较大的时候，需要对数据库进行分库分表，才能保证对于性能的需求

## 3.9限制条件

1. 表中数据过大的时候，即便建立了索引，匹配时间也会到达几十秒，需要进行分布式存储
2. 高并发情形下，对于查询的速度和准确度存在较高的要求

## 3.10测试要点

1. 满足高并发下的快速成功的查询
2. 对于模糊查询，需要准确给出书籍的所有id，并且按照相关性进行排序
3. 完成在不存在以及非正常输入下提供正常的返回信息
4. 当后台存在大量数据的时候，也能够进行快速的查询操作

**借书模块：**

## 3.1 功能

借书功能:

通过查找功能从系统数据库中查找符合借出条件的书籍列表。通过确认用户信息查看用户是否拥有图书借阅的权限(如,馆藏书籍非管理员不可查看,或该用户借书证已经不在有效期内,或用户借阅图书数量达到了上限等)

借阅后修改图书的状态,以及修改用户的借阅列表的状态。并更新计算图书的代还日期

## 3.2 性能

性能上没有过多描写的,算法本身进行的操作的复杂度全部是o(n)，没有什么过多可以优化的地方,只是单单针对查询所获得的信息做简单的修改即可,需要注意的是在维护数据一致性上要做到完善,防止用户操作过程中产生某些位置的系统异常而造成数据错乱,为异常数据回滚做好保证即可。

## 3.3 输入项目（小于64个unicode字符，且不包含特殊符号）

1. 书名

2. 作者

3. 出版社

...

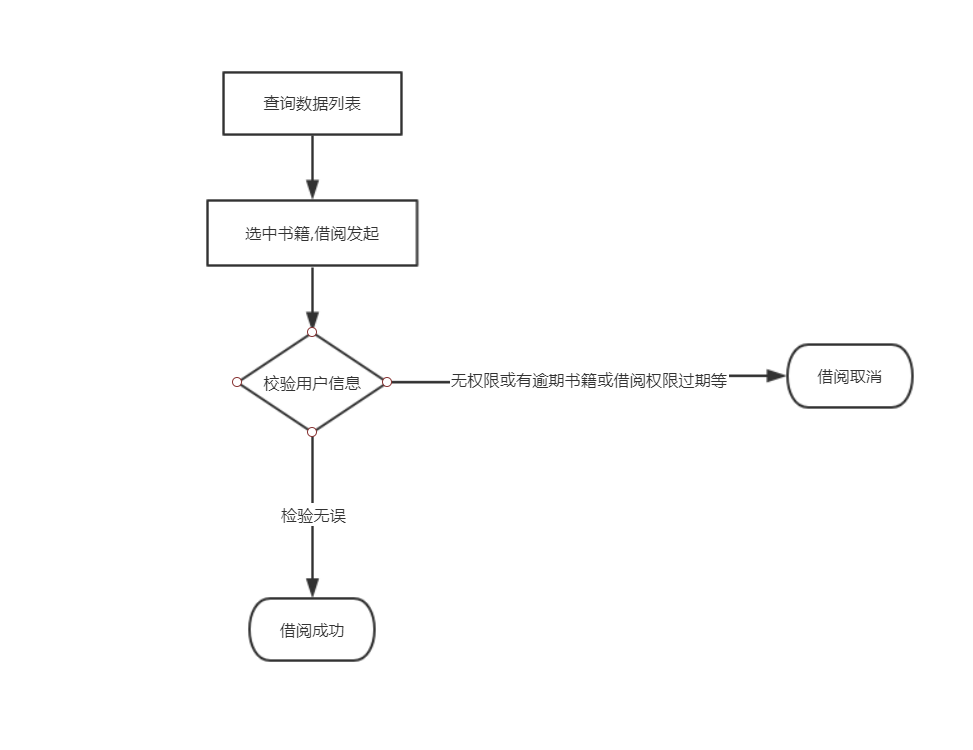
## 3.4 输出项目

借阅成功的信息,以及归还截止日期提醒用户在规定借阅时间内还书。

## 3.5 算法

算法本质没有什么特别复杂的地方,只是单单针对查询所获得的信息做简单的修改即可

## 3.6 程序逻辑

 **3.7接口**

1. 查询书籍能否借阅

Url：/canBorrowBook

方式：get

参数：id=书籍Id

返回值:{result:True/False}

1. 进行借阅

Url：/borrowBookById

方式：post

参数：id=书籍Id

返回值:{result:True/False

date:借阅的天数}

1. 查看总共的借阅次数

Url：/getBorrowBookTimesById

方式：post

参数：id=书籍Id

返回值:{result:True/False

time:次数}

## 3.8存储分配

1. 书籍借阅主要是通过操作书籍信息表以及借还信息表来实现
2. 查询书籍的借阅次数主要是通过查询书籍的借还信息表中对应的户籍的总共记录次数来返回实现

将所有的信息存储在mysql数据库中，同时将查询过的数据存储在缓存redis中，保证高并发下的快速读取，当后台数据量较大的时候，需要对数据库进行分库分表，才能保证对于性能的需求

## 3.9限制条件

由于需要提供对应的总共借阅次数的接口，因此相比较而言对于借还信息表，其中的数据是只增不减得，所以要维持好在大数据量的条件之下的并发查询能力

## 3.10测试要点

1. 保证在测试时对于错误的输入数据有准确的反馈信息
2. 保证一定的反应速度和检索速度
3. 进度安排：

第一阶段准备好数据为期两天

第二阶段搭建测试环境、确认主要功能的实现，为期一天。

第三阶段进行测试，测试本模块的主要功能。

第四阶段是进行分析与综合，优化系统性能。

**还书模块：**

## 3.1功能

在读取借阅者的信息和书籍的信息后，基于借阅者身份和书籍借阅状态的合法性，将书籍的借阅状态修改为未借阅，并将结果返回

## 3.2性能

算法包含两次操作，其一是数据库的操作，其二是算法附带的判断操作

在数据库条目采用了哈希索引，且在数据库底层采用了B+树的情况下，针对单条记录做查找操作的时间复杂度为，且针对单条记录的判断和写操作时间复杂度均为。综合后，得出还书算法的的时间复杂度为

其中值得注意的地方，便是数据的安全性问题。在频繁的数据写操作过程中，若是发生错误，便会产生严重的后果。解决方法是其一是需要添加操作日志，其二是需要合理的数据备份。

## 3.3输入项目

1. 用户编号
2. 书籍编号

## 3.4输出项目

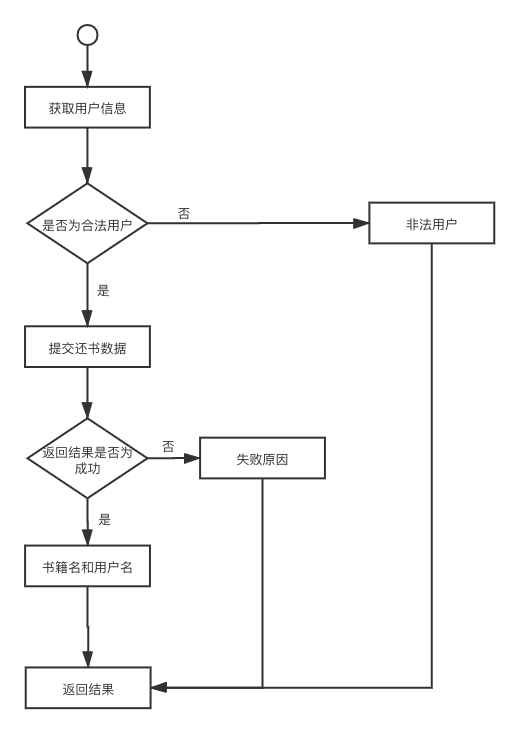
还书结果，如果失败提示错误信息，如果成功则提示还书的对应书籍名称和用户

## 3.5算法

第一步是针对用户的合法性进行查询，并对结果进行分析。如果用户不合法或是没有权限，则会直接返回失败信息，否则执行下一步

第二步是将书籍信息提交给数据库，让数据库根据条件进行合理的查询和写操作，并对结果进行分析。如果返回失败信息，则将对应失败信息输出，如果返回成功信息，则将书籍名和用户名返回。

## 3.6程序逻辑



## 3.7接口

1. 查询书籍是否已还

Url：/isReturnBook

方式：get

参数：id=书籍Id

返回值:{result:True/False}

1. 进行还书

Url：/returnBookById

方式：post

参数：id=书籍Id

返回值:{result:True/False

date:距离还书的天数

returnDate:还书的日期

}

1. 查看书籍距离还书的日期

Url：/getReturnBookDateById

方式：post

参数：id=书籍Id

返回值:{result:True/False

date:距离还书的天数

}

## 3.8存储分配

1. 书籍还书主要是通过操作书籍信息表中的状态字段以及修改借还书表中的记录来实现
2. 还书的日期是在数据库中是通过对应的借书日期加上对应的借阅日期计算出来的

将所有的信息存储在mysql数据库中，同时将查询过的数据存储在缓存redis中，保证高并发下的快速读取，当后台数据量较大的时候，需要对数据库进行分库分表，才能保证对于性能的需求

## 3.9限制条件

还书之后对于数据信息的修改需要及时，保证下一个借阅用户能够及时地、准确地查看到书籍对应的数据状态

## 3.10测试要点

1. 保证在还书之后状态的改变在短时间内就能够完成
2. 对于时间零界点的时候，要有足够明确的定义和反馈来告知用户

**续借模块：**

## 3.1功能

在读取借阅者的信息和书籍的信息后，基于借阅者身份和续借的资格审查，审查通过就将书籍的借阅到期时间往后延期一个月的时间，未通过就不延期；

## 3.2性能

对于续借的操作，数据库查询本身具有难以避免的O(logn)复杂性，但这部分的性能和我们的用户量有关，对于一个图书管理系统的较为狭小的用户范围来说，我们难以达到多于1s的查询延时，同时由于我们尚不具备对数据库底层的操作能力，因此这部分是难以优化的；

而更加重要的是对续借的资格审查，这部分我们会对该用户的不良记录和用户身份进行校验，如何设计资格审查的有效性和复杂度是我们主要关注的性能优化问题；

## 3.3输入项目

1. 用户编号
2. 书籍编号

## 3.4输出项目

续借结果，如果失败提示错误信息，如果成功则提示续借成功，并返回新的归还时间；

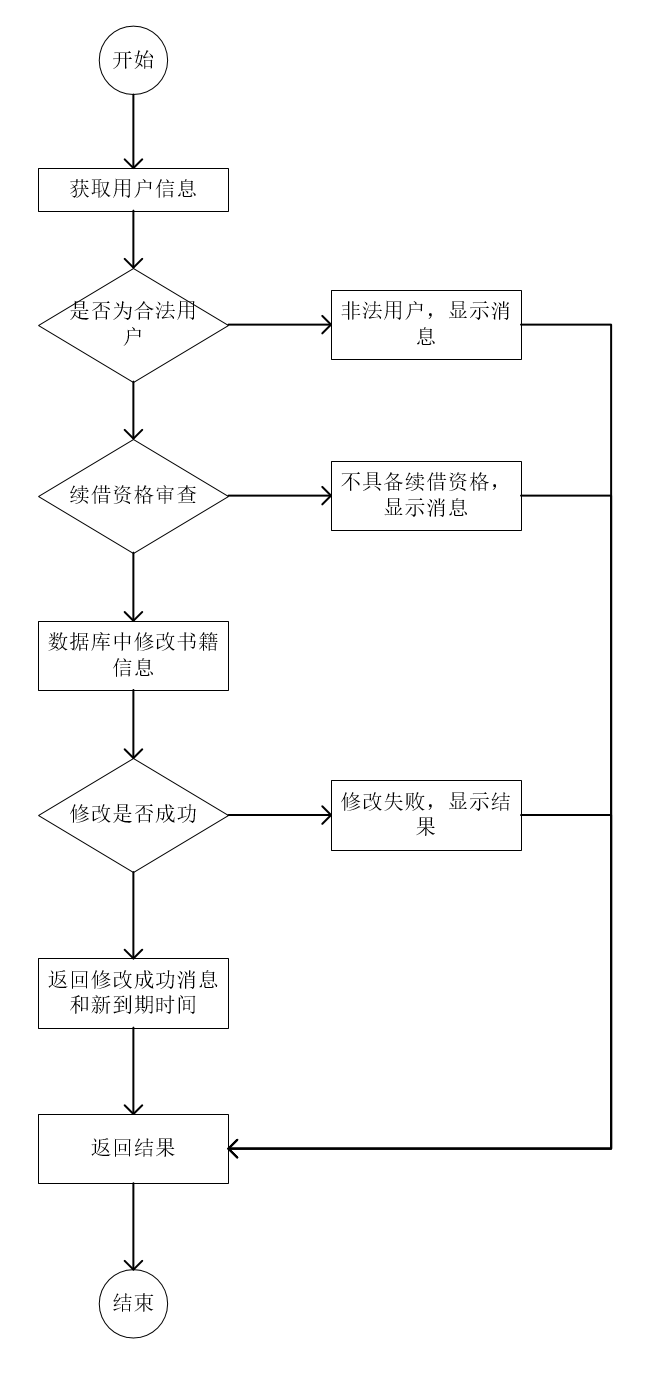
## 3.5算法

第一步是针对用户的合法性进行查询，并对结果进行分析。如果用户不合法或是没有权限，则会直接返回失败信息，否则执行下一步；

第二步是对用户的资格进行审查，着重关注用户的违约情况来确定是否运行续借；

第三步是将书籍信息提交给数据库，让数据库根据条件进行查询和写操作，修改对应ISBN码的书籍的到期时间。如果返回失败信息，则将对应失败信息输出，如果返回成功信息，则将书籍的相关信息和新的归还时间返回。

## 3.6程序逻辑



## 3.7接口

1. 查询书籍能否续借

Url：/canRenewBook

方式：get

参数：id=书籍Id

返回值:{result:True/False}

1. 查询书籍能够续借的天数

Url：/getRenewDateById

方式：post

参数：id=书籍Id

返回值:

不存在返回 {result:False}

存在返回

{result:True,

date:10

}

1. 查询书籍能够续借的次数

Url：/getRenewTimeById

方式：post

参数：id=书籍id

返回值:

{result:True,

time:次数

}

1. 进行续借

Url：/renewBook

方式：get

参数：id=书籍id

返回值:

{result:True/False}

## 3.8存储分配

1. 书籍信息表用于存储所有同一本图书的基本信息，在该表中查询是否能够进行续借，以及还能够进行续借的次数
2. 借还信息表存储了每一次借还图书的过程，每一条记录都是一个用户与一本图书的关系映射，当用户进行续借的时候，记录对应的续借信息

将所有的信息存储在mysql数据库中，同时将查询过的数据存储在缓存redis中，保证高并发下的快速读取，当后台数据量较大的时候，需要对数据库进行分库分表，才能保证对于性能的需求

## 3.9限制条件

1. 续借是一个写入操作，在较大并发的情况下对于数据库的压力比较大，需要进行合理的处理才能够既快速又准确的
2. 在数据量较大的时候，检索和修改都是缓慢的，需要进行合理的分库分表

## 3.10测试要点

1. 满足高并发下成功地完成续借
2. 对于不满足条件下地续借应当给出正常地反馈
3. 满足不同情况下，例如没登陆、书籍已还等，能够顺利即使的完成续借