**1. 引言**

本文档描述了图书管理系统的需求，包括功能、用户界面、技术要求和其他相关信息。图书管理系统旨在帮助管理者完成对图书的管理。本系统将支持管理员和读者两种角色，提供从图书入库到借阅、归还、检索、罚款以及统计分析在内的一系列功能。

图书管理系统的设计初衷是为了简化图书的管理流程，减少人工操作的复杂性，提高管理效率和用户体验。通过使用软件技术，我们希望能够实现一个功能完备、界面友好、性能稳定的系统，满足图书馆、学校、企业等各种场景的需求。

本文档将详细描述系统的各项需求，包括功能需求、技术要求、用户界面设计等内容，以便开发团队在后续的开发过程中能够有据可依，确保系统的各项功能能够按时、高质量地实现。

**2. 功能需求**

##### 2.1系统管理需求

2.1.1 准备工作

双击目录下的“Library.exe”即可运行

2.1.2 建立数据库

通过以下对图书管理系统的各个模板功能的划分以及接下来对系统功能需求分析，得到系统的整体结构图（如2.1.2所示）

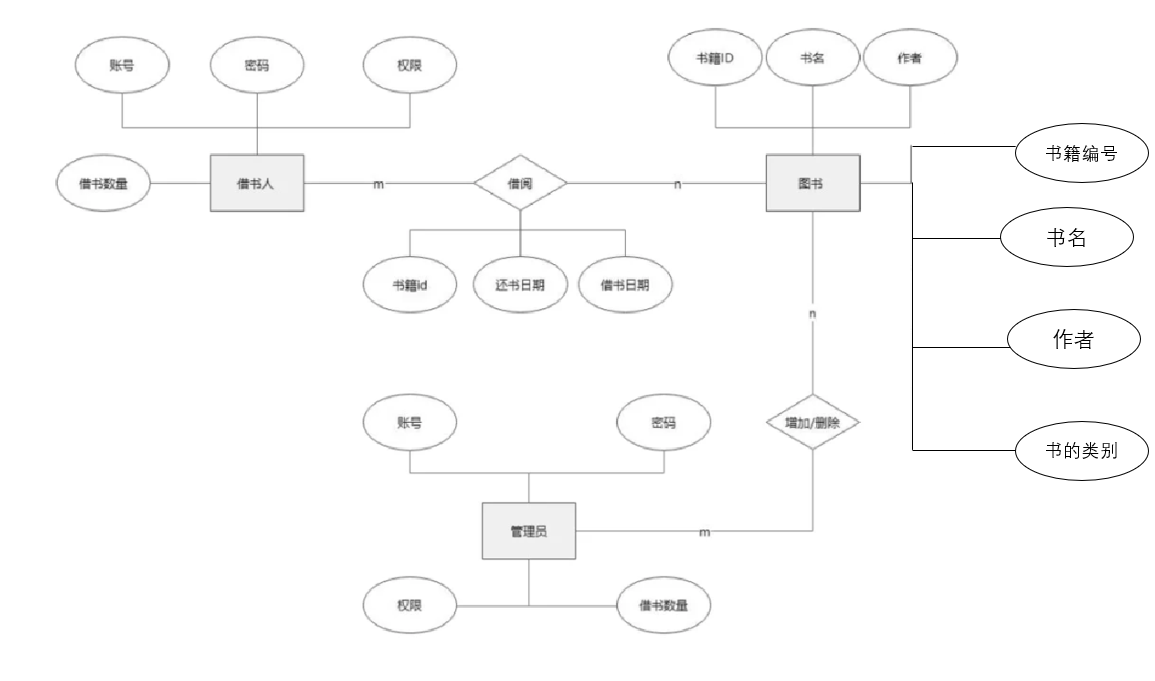


图2.12

透过分析设计，利用SQLite设计系统的数据库并名为LirbraryDatabase。

2.1.3数据库备份与恢复

定期备份系统数据，并在需要时进行数据恢复（ex.书本信息）

2.1.4 用户权限设置

为不同用户分配不同的权限。

##### 2.2 图书管理需求

2.2.1图书入库与下架

用户故事：作为一名图书馆管理员，我想要简单完成图书入库与下架，以便我快速完成入库和删除工作。

功能需求:

* 手动录入新书信息，包括书籍编号，书名、作者、出版社和书的类别
* 系统应提供字段校验功能(ex.字段是否为空，编号是否符号格式)
* 管理员可通过 excel批量导入图片信息



2.2.2图书状态与修改

（1）状态查询

用户故事：作为一名图书馆管理员，我想要快速查询以及修改图书状态，以便增加工作效率。

功能需求:

* 检视所有书籍的借阅状态为空闲/已借出
* 删除书籍
* 进入修改书籍页面



（2）修改书籍具体资料

用户故事：作为一名图书馆管理员，我想要快速修改书籍的资料，以便完成书库的更新。

功能需求:

* 修改书籍具体资料
* 返回



2.2.3 用户管理

用户故事：作为一名图书馆管理员，我想要清楚看到和简单的修改读者用户的状态，以便管理用户。

功能需求:

* 修改用户资料
* 查看用户借书、欠款情况
* 删除用户



2.2.4 数据备份

用户故事：作为一名图书馆管理员，我想要每天保存数据，以便防止意外情况。

（1）备份步骤：

1. 打开系统管理界面。
2. 选择“数据库备份”选项。
3. 选择备份文件保存位置。
4. 点击“备份”按钮，系统会自动进行数据备份。

（2）恢复步骤

1. 打开系统管理界面。
2. 选择“数据库恢复”选项。
3. 选择需要恢复的备份文件。
4. 点击“恢复”按钮，系统会自动恢复数据。

2.2.5非功能性要求:

* 易用性:系统界面应简洁易懂
* 安全性:系统需要足够的数据保护，防止数据被恶意篡改
* 性能:提供系统响应速度去高效应对大量信息的录入与保存

##### 2.3.读者管理

2.3.1 读者注册

新读者登记注册，获取借阅权限。

2.3.2  读者信息修改

更新读者个人信息。

2.3.3  读者借阅记录查询

查询个人借阅历史和当前借阅状态

2.2.3 图书查询

用户故事：作为一名读者，我想要快速地查询想要的书本，以便找到想要的书。

功能需求：

* 手动输入书名来搜寻书本。
* 输入IBSN码来搜寻书本。



2.3.4.支付罚款

用户故事：作为一名读者，我想要在线上完成罚款的缴纳，以便不用亲身到图书馆缴纳罚款。

功能需求：

* 缴纳罚款



2.3.5用户反馈系统

用户故事：作为一名读者，我想要有个简单的途径反馈我的建议与意见，以便得到更好的服务。

功能需求：

* 传送反馈建议与意见的内容



**3. 技术要求**

* 使用Qt界面开发框架，以实现跨平台的用户界面。
* 使用文件I/O操作来保存和读取用户的任务和历史数据。并且利用文件I/O操作来保存和读取用户的图书借阅历史和其他相关数据，确保数据的持久化存储和可靠读取。
* 通过多线程技术实现系统中的各种计时功能和任务管理，提高系统响应速度和用户操作的流畅性。
* 图形处理库来绘制系统中的各种图表，如图书借阅情况、用户借阅历史等，使数据展示更加直观和美观。
* 利用轻量级的SQLite数据库来存储系统中的用户数据、图书信息和借阅记录，确保数据存储的高效性和可靠性。
* 通过网络编程技术，实现客户端与服务器端之间的数据通信，确保用户在不同设备上的数据同步和一致性。

**4. 其他要求**

用户界面应该友好、直观，以便用户方便地操作和管理任务。

系统应具有良好的性能和稳定性，能够处理大量的任务和数据。

系统应具备良好的安全性，确保用户的数据不会被未经授权的访问所泄露。

数据保护 性能要求 接口（用户界面）c++开发