



图书管理系统

关镇(1171002076) 邓正望(1171002074)

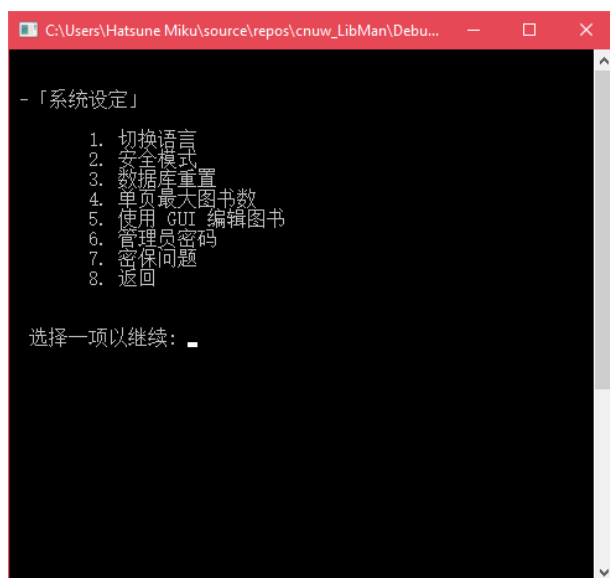
目录

| | |
|--------------------------|----|
| 一. 程序概览 | 2 |
| 1. 简介 | 2 |
| 2. 基础功能 | 3 |
| 二. 程序特点 | 8 |
| · 总览 | 8 |
| 1. SQLite 数据库与 AES256 加密 | 9 |
| 2. 数据云同步 | 10 |
| 3. 可编程性 | 11 |
| 4. I18N (国际化) | 13 |
| 5. Linux 兼容 | 15 |
| 三. 压力测试 | 16 |
| 1. 测试内容 | 16 |
| 2. 结论 | 18 |
| 四. 技术性细节 | 19 |
| 1. libcurl 与 OpenSSL 的利用 | 19 |
| 2. 程序流程图 | 19 |
| 3. “三权分立” | 20 |
| 4. 反省、问题与收获 | 21 |
| 5. 开发、构建/测试环境与组内分工 | 23 |
| 五. 开源许可 | 24 |
| 六. 附录 | 25 |

一. 程序概览

1. 简介

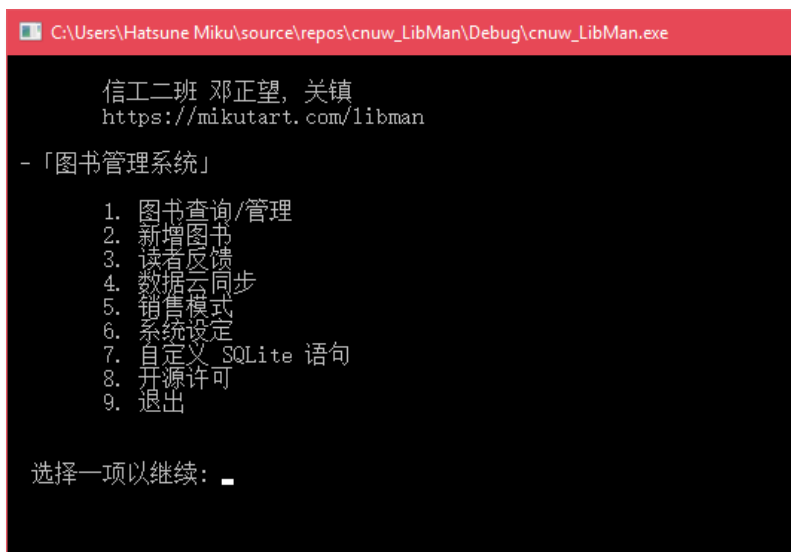
本程序具有图书管理的功能，以 CLI¹作为主要交互方式，辅以少量图形界面²用于优化用户体验。程序兼容 Windows 和各种发行版 Linux 系统。



设置页面



图形界面修改图书

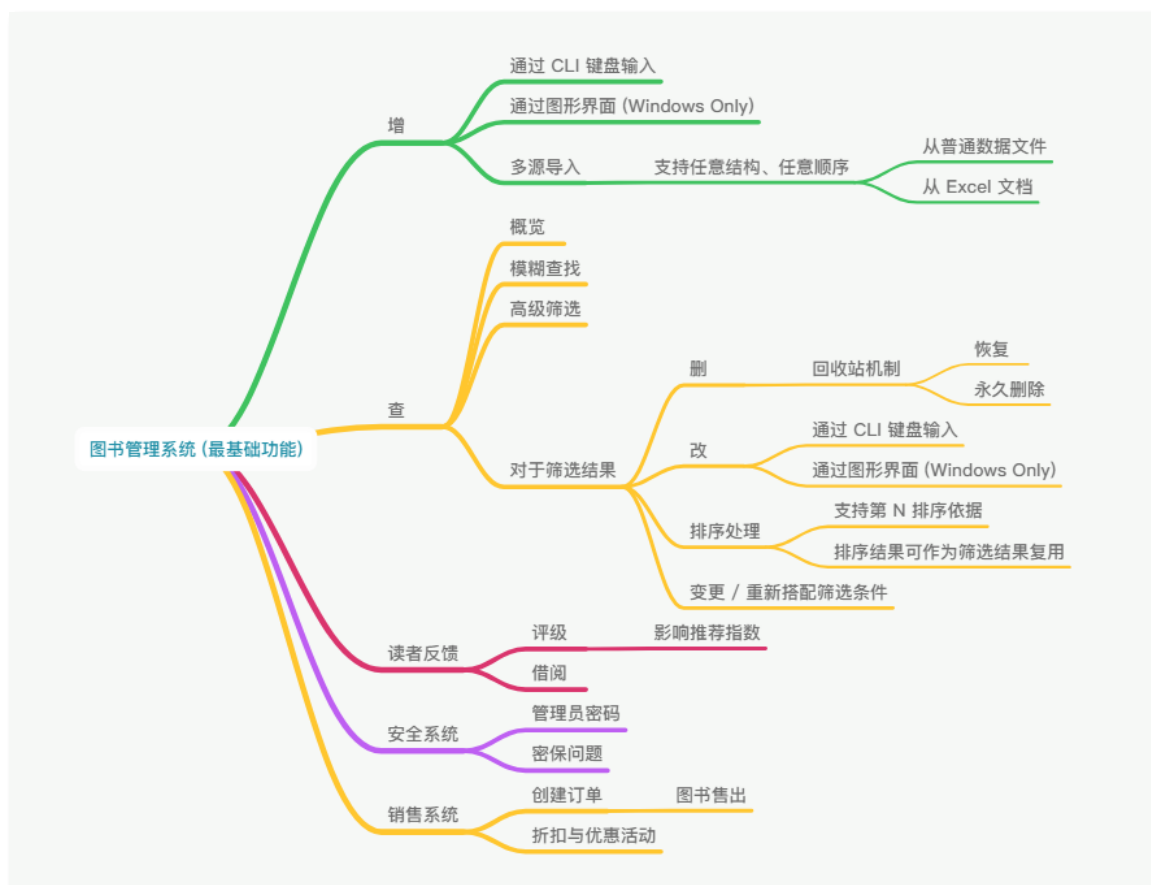


主页面

¹ CLI: 命令行界面 (Command-Line Interface).

² 图形界面: 仅限 Windows 系统。

2. 基础功能



如图所示，本程序作为日常使用具有上述基础功能。在后续章节中将陆续介绍本程序的全部功能。

另外，我们为部分值得一提的基础功能做了进一步的说明。

· 图形界面

在处理数据时，命令行窗口的能力略显不足。图形界面相对来说更加用户友好，而且一本图书的数据可以一目了然，在原地编辑，最后只需一下回车即可完成数据增改。考虑到跨平台问题，图形界面只作为额外功能在 Windows 系统中提供。

· 高级筛选

价格在 15~30 元之间，并且适合儿童阅读、库存充足的书。或是，近期打折力度超过 8 折、阅读量火爆且出自村上春树笔下的书。又或者，由清华大学出版社、高等教育出版社或人民邮电出版社三者之一所出版的图书。高级筛选支持相等（相同）、（数值或字典顺序上的）大于小于、不等于（不同）和形似于共 5 种条件，支持条件混搭，可以完成绝大多数的筛选需求。另外，条件组可以作为文件导入导出。

· 多源导入

一般情况下，从外部数据文件批量导入图书时，要求数据文件严格遵守固定的格式、结构。本程序将“外部文件格式”作为一项独立的设置项目，用户可以定义多种格式约定，并使其同时生效。导入时，只需先选择一项预设约定，便可以导入任意结构的文档，甚至是 Excel 文档³。

· 回收站机制

iOS 和 Android 的官方相册 App 陆续加入了「最近删除」功能，相当实用。我们进行了借鉴，并且在此基础上，去掉了删除图书时的确认提示，使得平时的删除操作既便捷又可逆。

· 读者反馈

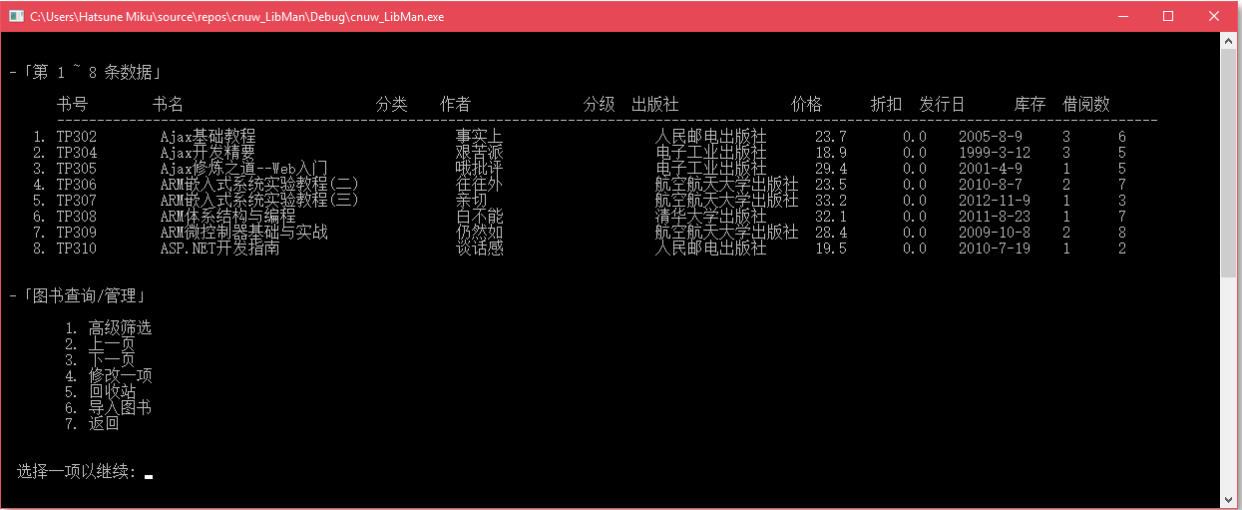
这里是读者归还图书的通道。在归还的同时，对应图书的借阅次数增加 1 次，然后读者自愿为图书进行 0~5 星的评分。程序根据总分数量和总借阅次数计算一本图书的推荐指数。如果读者放弃评分，当次借阅也将视为无效，不计入统计。

· 密码找回机制

用户可随时设置密码保护问题。其采取一问一答形式，用户通常根据一些仅自身拥有的记忆来设问，以确保仅用户本人持有答案。当忘记管理员密码时，如果曾经设置过密码保护问题，可以尝试找回。问题原文将呈现给任何操作者，其中能够给出正确回答的人，将有权立刻重设或清除管理员密码。

³ Excel 文档：仅支持 xls 后缀，且由纯文本描述的表格结构。

以下运行时截图作为本章的补充说明。



图书概览

*测试数据中未提供分类、分级，故未显示。



图形界面修改图书

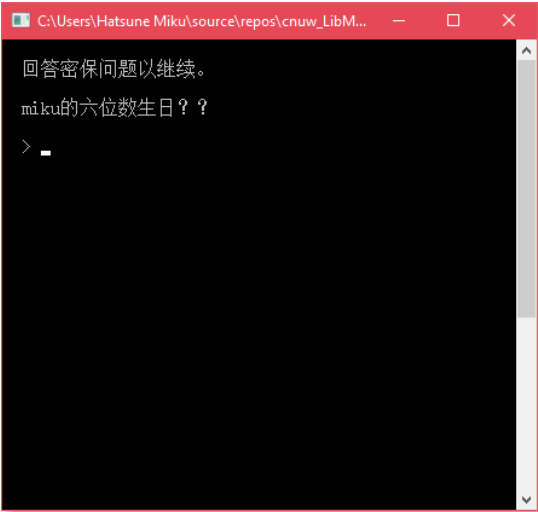
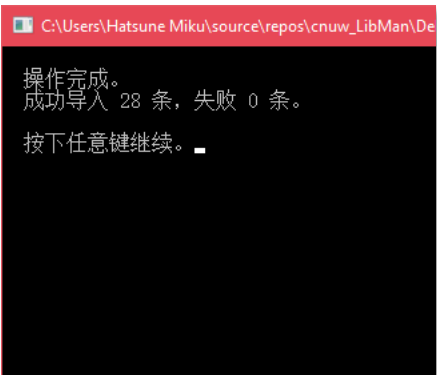
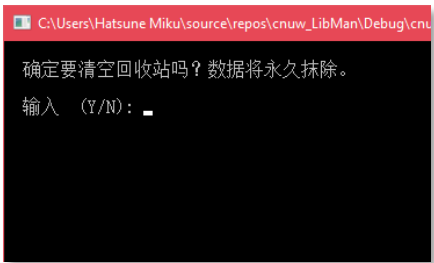


高级筛选

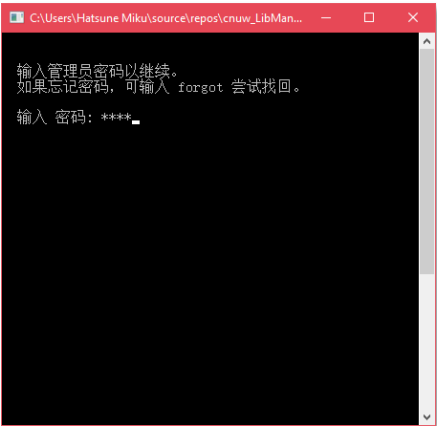


销售系统

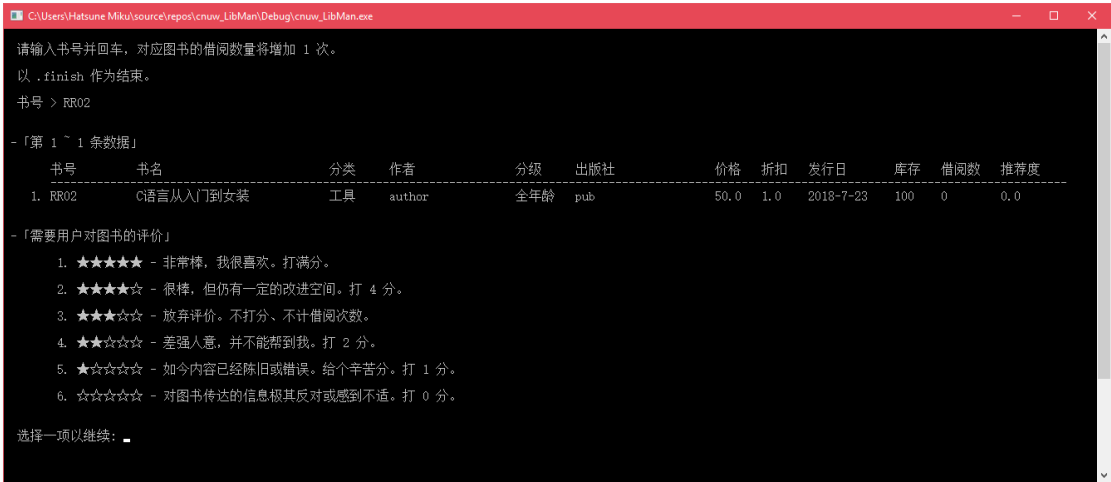
*图中资金流动仅为模拟，并未实现真实货币交易



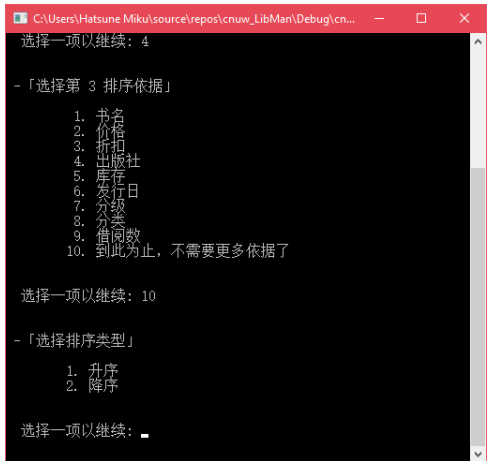
多源导入



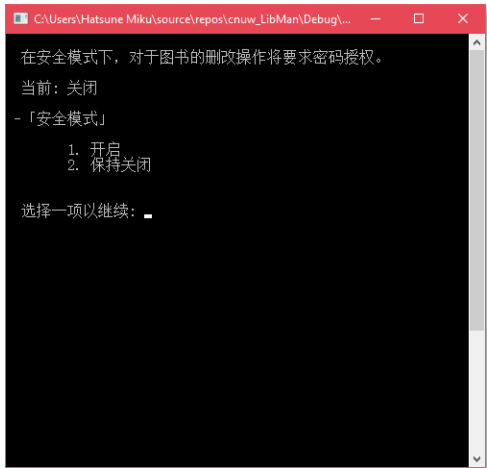
密码验证



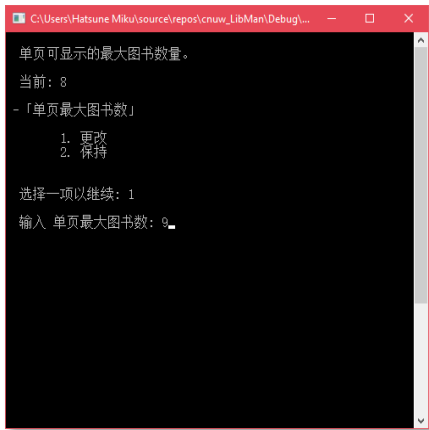
借阅模式



第 N 排序依据



导入条件组

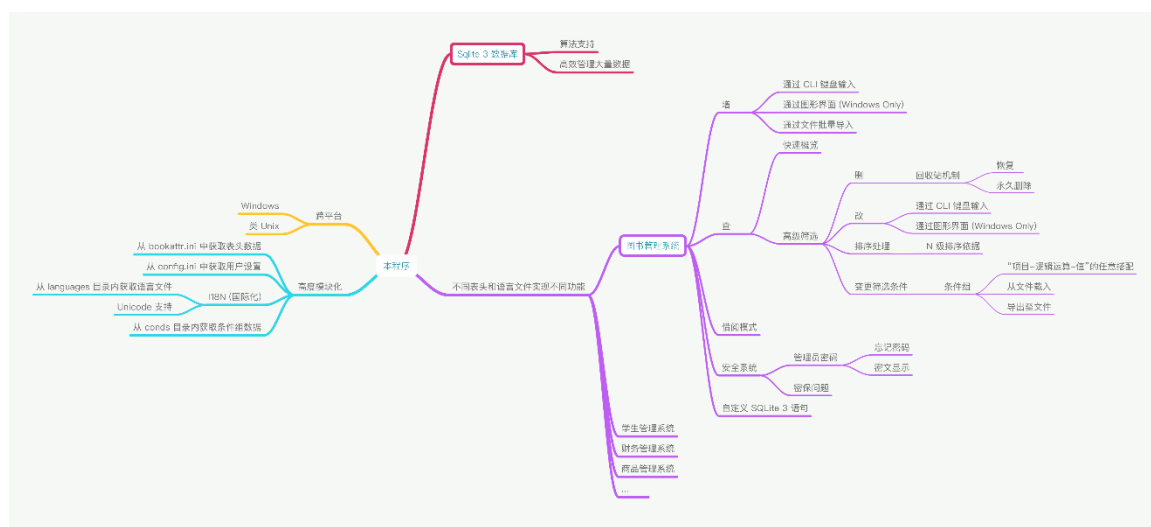


单页最大图书数

安全模式

二. 程序特点

· 总览



*图中“快速概览”即模糊查询功能，“借阅模式”即读者反馈与销售系统。

这是在项目合并时出现的称呼差异。



mindmap.png

图中所展示的是本程序的完整结构。

*高清版本

我们尽全力削减硬编码的数量。因此，本程序所用的全部数据，包括但不限于图书表头、语言文字、图书各属性数据类型，皆外置为配置文件，甚至可以在运行时更改，而程序本身只处理必要逻辑。

这为本程序特点功能的开发打下了基础。

1.SQLite 数据库与 AES256 加密



要维护大量的数据，如果不选用合适的数据存储方案，是很不明智的。我们使用开源、轻量级的 SQLite 数据库作为数据存储方案。

同时，数据安全是必须解决的问题。我们将 AES256 加密应用于 SQLite 中，极大程度保障了数据安全。

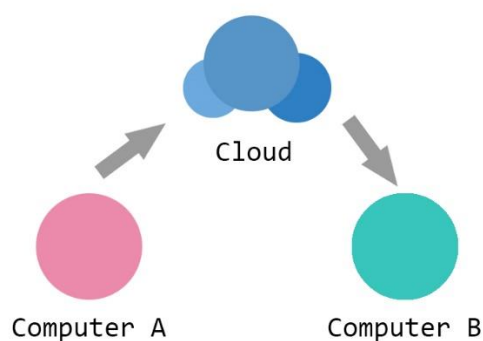
强度较弱的 AES128 或 AES192 加密算法已经足以将绝大多数破译者拒之门外，因此 AES256 目前可以认为非常安全。

```
C:\Users\Hatsune Miku\source\repos\cnuw_LibMan\Debug\cnuw_LibMan.exe
请输入 SQLite 3 语句。
数据库名为 main, 表名为 book .
以 .finish 作为结束。

> select * from book;
1 TP302 Ajax基础教程 事实上 人民邮电出版社 23.7 0.0 2005-8-9 3 6 0
2 TP304 Ajax开发精要 艰苦派 电子工业出版社 18.9 0.0 1999-3-12 3 5 0
3 TP305 Ajax修炼之道--Web入门 哦批评 电子工业出版社 29.4 0.0 2001-4-9 1 5 0
4 TP306 ARM嵌入式系统实验教程(二) 任往外 航空航天大学出版社 23.5 0.0 2010-8-7 2 7 0
5 TP307 ARM嵌入式系统实验教程(三) 亲切 航空航天大学出版社 33.2 0.0 2012-11-9 1 3 0
6 TP308 ARM体系结构与编程 白不能 清华大学出版社 32.1 0.0 2011-8-23 1 7 0
7 TP309 ARM微控制器基础与实战 仍然如 航空航天大学出版社 28.4 0.0 2009-10-8 2 8 0
8 TP310 ASP.NET开发指南 谈话感 人民邮电出版社 19.5 0.0 2010-7-19 1 2 0
9 TP311 J2ME开发精解 督导 电子工业出版社 25.0 0.0 2008-11-25 5 0 0
10 TP312 多智能体模型与实验 饿额 清华大学出版社 33.8 0.0 2009-12-7 3 7 0
11 TP313 关于知识的本体论研究 如同眼 巴蜀书社 43.6 0.0 2012-3-6 2 4 0
12 TP314 机器学习及其应用 轧花机 清华大学出版社 30.5 0.0 2011-10-9 1 3 0
13 TP315 基于MATLAB的系统分析与设计 搞活机 西安电子科技大学 16.9 0.0 2011-4-16 1 3 0
14 TP316 计算机语料库的建设与应用 飞过海 清华大学出版社 23.9 0.0 2012-5-20 1 5 0
15 TP317 计算语言学前瞻 各个 商务印书馆 33.2 0.0 2013-6-30 1 4 0
16 F120 科学计算导论 按时的 清华大学出版社 28.9 0.0 2010-1-17 2 7 0
17 F121 嵌入式实时操作系统ucos-II 方法他 航空航天大学出版社 27.8 0.0 2013-8-7 1 6 0
18 F122 嵌入式系统设计与实践 好看 航空航天大学出版社 45.8 0.0 2009-2-21 1 4 0
19 F123 嵌入式系统设计与应用开发 由于与 航空航天大学出版社 30.6 0.0 2007-6-14 2 0 0
20 F124 人工神经网络与模拟进化计算 下风格 清华大学出版社 19.8 0.0 2012-2-18 2 7 0
21 F125 数据结构——Java语言描述 饿额 清华大学出版社 26.7 0.0 2012-7-10 2 8 0
22 F126 数据库原理 共同 清华大学出版社 32.8 0.0 2010-4-15 1 4 0
```

“自定义 SQLite 语句”功能

2. 数据云同步



此功能将所有用户数据（包括密码，但除条件组文件外）备份至云端，并支持在其它运行本程序的设备上将数据下载。



数据云同步为图书数据迁移提供了极大便利。但是由于个人服务器性能原因，无法保证当数据量大（约十万条以上）时的传输速度。

本程序与云端建立的是符合苹果 ATS 规范的 HTTPS 连接。即使用户数据本身已经是加密的，也要尽可能提升安全等级。

3. 可编程性

右侧的文件，根据本程序设计，是作为一个“图书管理系统”所应具有的头数据。

取其中三行作为示例：

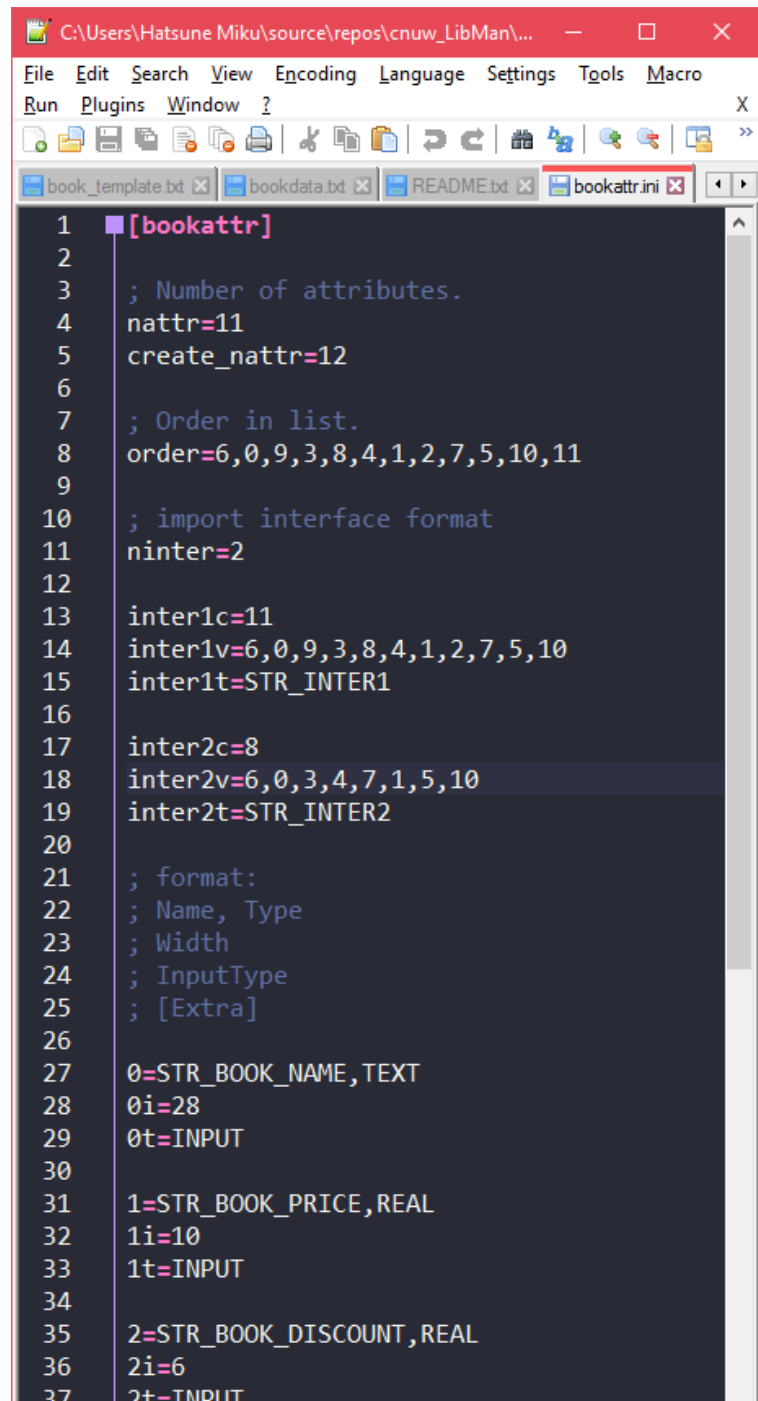
*0=STR_BOOK_NAME,TEXT
1=STR_BOOK_PRICE,REAL*

order=6,0,9,3,8,4...

第一行表明，第 0 项数据的名字是 STR_BOOK_NAME，它的数据类型为 TEXT⁴。

第二行表明，第 1 项数据的名字是 STR_BOOK_PRICE，它的数据类型为 REAL⁵。

第三行表明，表头从左至右的顺序是：第 6 项、第 0 项、第 9 项，以此类推。



```
1 [bookattr]
2
3 ; Number of attributes.
4 nattr=11
5 create_nattr=12
6
7 ; Order in list.
8 order=6,0,9,3,8,4,1,2,7,5,10,11
9
10 ; import interface format
11 ninter=2
12
13 inter1c=11
14 inter1v=6,0,9,3,8,4,1,2,7,5,10
15 inter1t=STR_INTER1
16
17 inter2c=8
18 inter2v=6,0,3,4,7,1,5,10
19 inter2t=STR_INTER2
20
21 ; format:
22 ; Name, Type
23 ; Width
24 ; InputType
25 ; [Extra]
26
27 0=STR_BOOK_NAME,TEXT
28 0i=28
29 0t=INPUT
30
31 1=STR_BOOK_PRICE,REAL
32 1i=10
33 1t=INPUT
34
35 2=STR_BOOK_DISCOUNT,REAL
36 2i=6
37 2t=INPUT
```

⁴ TEXT: Sqlite3 数据库中的数据类型，代表字符串。

⁵ REAL: Sqlite3 数据库中的数据类型，代表浮点数。

简而言之，本程序的这一特点，允许用户自定义所有的数据项。
我们预置了图书的数据项（书名、价格等），它就成为“图书管理系统”。
如果预置学生的数据项（学号、成绩等），它就成为“学生管理系统”。
如果预置票务的数据项（场次、人员等），它就成为“票务管理系统”。
...



如图所示，通过修改配置文件的数据项，“图书管理系统”摇身一变成为“学生管理系统”。



4. I18N (国际化)

```
C:\Users\Hatsune Miku\source\repos\cnuw_LibMan\Deb

信工二班 关镇(1171002076), 邓正望

- 「图书管理系统 CLI」

1. 图书查询/管理
2. 新增图书
3. 借阅模式
4. 系统设定
5. 自定义 SQLite 语句
6. 退出

选择一项以继续: _
```

```
C:\Users\Hatsune Miku\source\repos\cnuw_LibMan\Debug\

Information Project Class 2 关镇(117

- [ Library Manager CLI ]

1. Book Manage & Query
2. New Book
3. Borrow Mode
4. Settings
5. Custom SQLite Command
6. Exit

Select one from options: _
```

程序内部没有任何自然语言字符串，全部自然语言外置在独立的语言文件中，并拥有一定的模板功能。本程序的语言文字可以随时更新。

我们并赋予程序 Unicode 字符集。因此，不论是哪个国家的语言，日语、繁体、法语、俄语、韩语...只需写好语言文件，程序便可支持。

```
C:\Users\Hatsune Miku\source\repos\cnuw_LibMan\Release\languages\1.ini - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
说明.txt 1.ini bookapp.txt book_template.txt bookdata.txt README.txt

160 STR_FILTER_LOAD_TIP=输入其中一个的名字:
161 STR_FILTER_IMP=导入条件
162 STR_FILTER_SAV=保存条件
163 STR_FILTER_RESULT=筛选结果
164 STR_FILTER_HEADER= ID/t项目/t/t条件/t值/n
165 STR_FILTER_TEMPLATE= %d/t%/t%/t%/t%/n
166 STR_FILTER_START_AND=以「并且」的关系开始筛选
167 STR_FILTER_START_OR=以「或者」的关系开始筛选
168 STR_FILTER_SORTING=排序处理
169 STR_FILTER_SORT_DEPTH=选择第 %d 排序依据
170 STR_FILTER_SORT_OK=到此为止，不需要更多依据了
171 STR_FILTER_SORT_TYPE_TIP=选择排序类型
172 STR_FILTER_SORT_ASC=升序
173 STR_FILTER_SORT_DSC=降序
174 STR_FILTER_DELETE=删除全部
175 STR_FILTER_DELETE_ONE=删除一项记录
176 STR_FILTER_EMPTY_IN_TRASHBIN=彻底删除全部
```

```

UI_Printf(L"TITLE_TEMPLATE", L"APP_NAME", L"CREDIT");
// 将 APP_NAME 和作者信息放入 TITLE_TEMPLATE 模板中并打印。

UI_MenuMake(L"STR_MAIN_MENU", 6,      /* 菜单名: 主菜单 */
            L"STR_BOOK_LOOKUP",        /* 图书查询/管理 */
            L"STR_BOOK_ADD",           /* 新增图书 */
            L"STR_BOOK_BROWMODE",      /* 借阅模式 */
            L"STR_SETTINGS",           /* 系统设置 */
            L"STR_ANY_SQLITE_CMD",     /* 自定义 SQLite 语句 */
            L"STR_EXIT");              /* 退出 */
// void UI_MenuMake(const wchar_t* title, int num, ...);
// 创建菜单并打印。

```

但是从代码片段中可以看到，即使顾及了全球语言，开发成本也并不高。

```

603
604 void CP_ModifyOne(void) { // 修改一项图书数据
605     if (getInt(PATH_CONFIG, L"safe_mode") // 读取设置: 是否启用了安全模式
606         // (根据程序设定, 安全模式下, 修改图书需要密码)
607         && !UI_MastercodeCheck(false)) // 密码检验。
608         return; // 密码错误的话, 不可以修改图书。
609

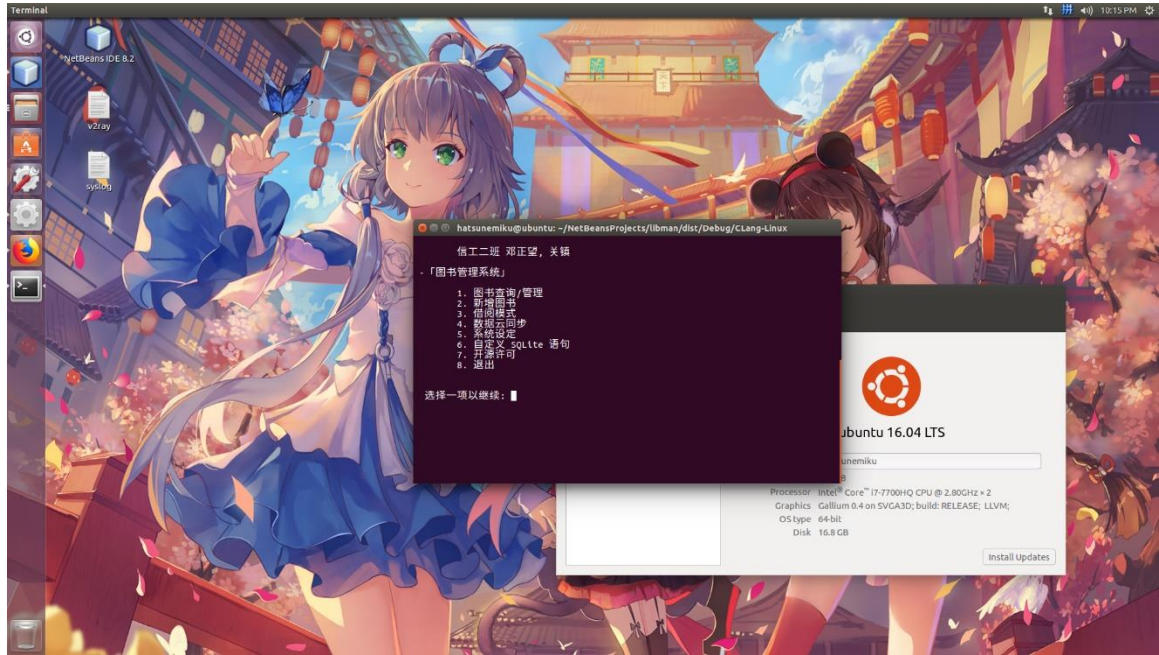
```

另外，这是修改图书信息的代码片段。短短两行代码就完成了读取设置、验证密码的工作。这些都得益于模块化设计。

5. Linux 兼容

“在一台运行 Windows 系统的电脑上添加几本图书，然后云同步至运行 CentOS 的服务器上长期保存。”

借助 C 语言的跨平台优势，这一功能并不难实现。Linux 版可以享受到除图形界面以外的全部功能。

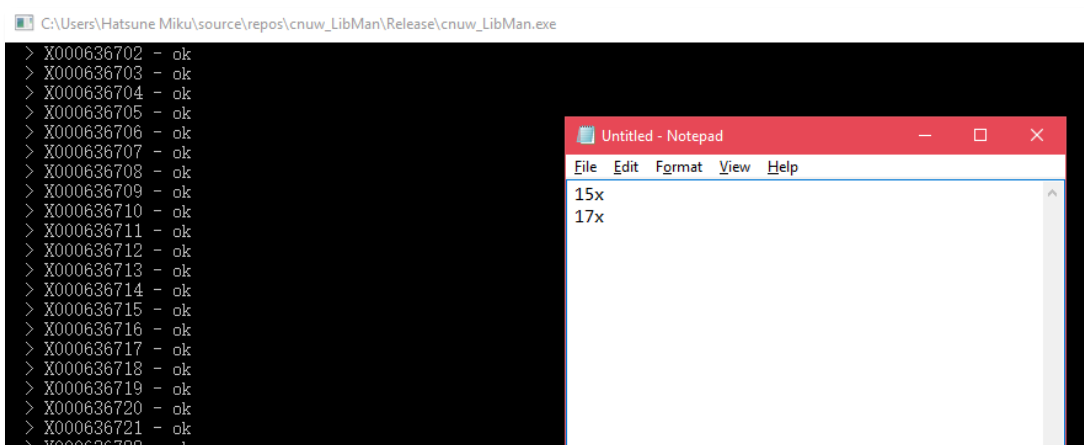


在 Ubuntu 16.04 LTS 环境下编译运行

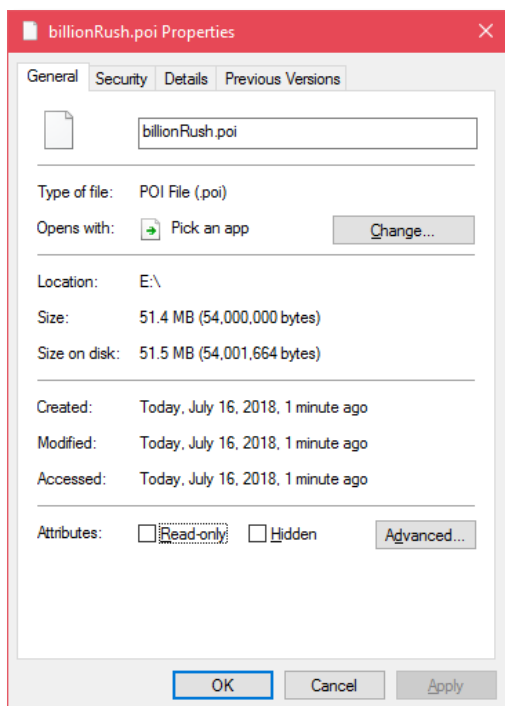
三. 压力测试

1. 测试内容

我们生成了包含 100 万条图书数据文件，并将其导入数据库中。



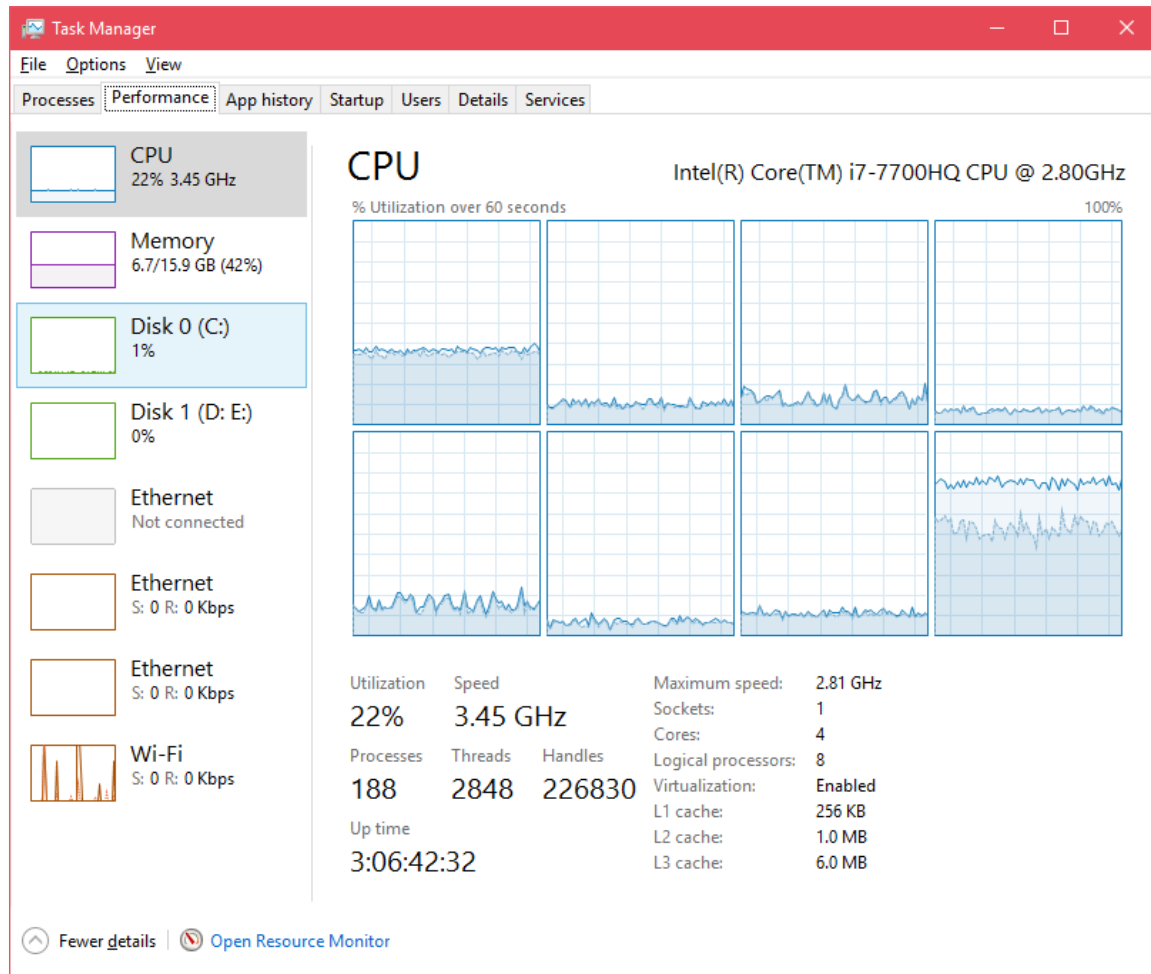
导入时标记时间的截图



数据文档



导入后



测试环境使用 Intel Core™ i7-7700HQ 处理器。数据导入时，硬盘 IO 量保持在极低水平。

在对新数据实时进行 AES256 高强度加密的情况下，导入 100 万条数据用时 18 分钟。导入完成后，程序运行流畅如初。筛选、排序等操作基本未受影响。

2. 结论

数据量在 0~100 万之间时，程序运行稳定、处理数据所需时间均匀上升，但仍然高效（可立即完成筛选、排序等操作）。因此，可以胜任多数图书馆的电子档案管理。

截至 2018 年 7 月 17 日，美国国会图书馆作为世界最大图书馆，存有书籍约 1.5 亿册，其中超过 2/3 的书籍（约 1 亿）通过多媒体存储。1 亿是 100 万的 100 倍，从本程序目前的表现来说，如果不计书籍内容，有可能在效率下降幅度较低的情况下实现 1 亿存储量。

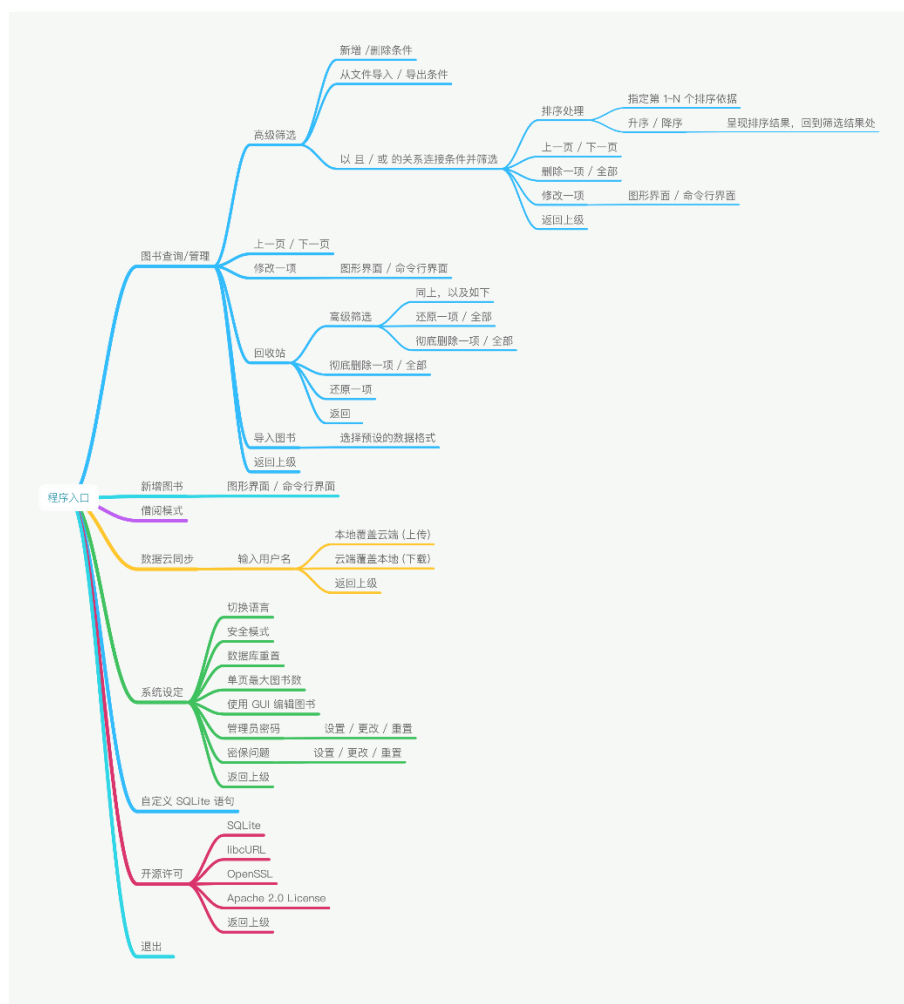
四. 技术性细节

1.libcurl 与 OpenSSL 的利用

本程序的“数据云同步”功能使用了开源项目 libcurl 作为底层库。

然而 libcurl 并不支持 SSL 协议，于是我们使用 OpenSSL 作为支持库重新编译了 libcurl 使其满足我们的要求。

2. 程序流程图

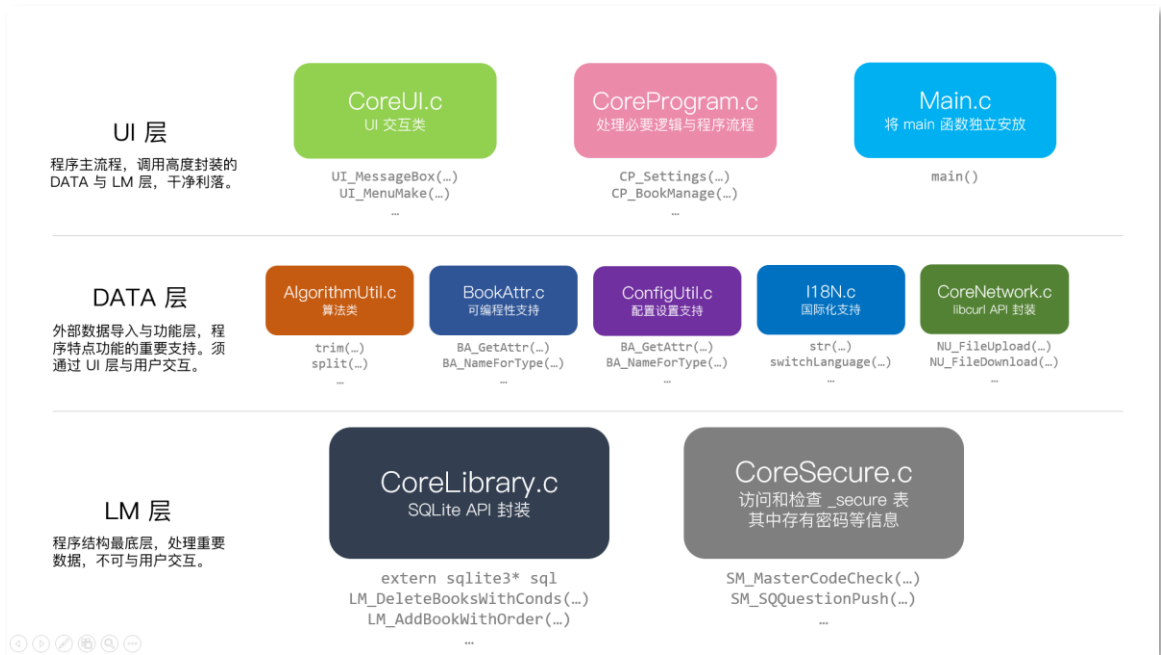


高清版本 program.png

由于功能较多，流程图中没有体现部分细节。

3. “三权分立”

本程序内部结构分层以及各个模块职能大致如下。



高清版本 arch.png

根据各类模块职能不同，整个程序可划分为三层：UI 层、数据层和安全层。其中安全层禁止主动进行 UI 交互。

UI 层处理程序的流程逻辑，如各级菜单间的切换、菜单的打印、语言文字的输出。封装在 UI 层的函数以 UI_或 CP_开头。

DATA 层即数据层。它对图书数据加工、负责编码转换、进行外部库的引入、实现全部特点功能，并为程序提供一套内存管理方案。封装在其中的函数以 BA_或 NU_开头。

LM 层即安全层。它封装各种数据库操作、负责密码和密保问题的验证、更改。它所使用的内存“阅后即焚”，非专业跟踪技巧难以捕捉它的举动。封装的函数以 LM_或 SM_开头。

4. 反省、问题与收获

· 改进空间

- 没有解决多用户使用、用户权限分级这一刚需，以及远程数据库。如果能采用远程数据库，安全性将得到巨大提升。

- 代码注释大多是给自己看的，没考虑到别人能否完全理解。

- 未使用版本控制工具，考虑到团队很小，没有必要。

- 对于一些编码转换函数，应尝试写出不依赖非 C 标准库的更通用的算法，减少在代码中穿插的（用于判断操作系统的）预处理器宏。

- 程序二进制文件未做额外保护（如加壳），且全部密码验证走同一关键跳，易被爆破从而绕过密码验证。

· 遇到的问题

开发过程总体相对顺利，但遇到的唯一问题非常棘手，贯穿整个开发过程。通俗的说，这一问题导致大量语言文字“乱码”。另外，由于类似原因，数据库无法支持非纯英文的 like 子句查询。这一问题导致本程序无法实现“*(11) 可以根据部分书名查询图书，如查询“钢铁”+“炼成”，可以查到“钢铁是怎样炼成的”这本书*”这一基本要求，从而险些导致我们重写整个高级筛选功能。

· 问题的解决

最初我们使用宽字符(wchar_t)替代字符型(char)作为数据存储，即 UCS-2LE，并在 UI 层预置转换函数使其能够呈现。没想到在 Linux 系统中 wchar_t 被规定为占 4 字节（Windows 系统中是 2 字节），这一规定使得程序编码体系完全被破坏，如果继续使用宽字符，将无法兼容 Linux 系统。

随后，我们重新用回 char 类型，重写相关代码，并进行了两天的学习，采用 GBK 编码临时解决了系统兼容问题。

然而 GBK 编码显然不能满足国际化的要求，它只映射了简繁中文。同时我们发现，高级筛选功能中，“形似于”条件在部分情况下异常。在测试环境中，存有一本名为”钢铁是怎样炼成的”的图书，使用

```
select * from book where STR_BOOK_NAME like '%钢铁%' and STR_BOOK_NAME like '%炼成%';
```

无法筛选出任何结果，但

```
select * from book where STR_BOOK_NAME like '%钢铁%';
```

的筛选结果正常。经过半天学习，得知 SQLite 数据库默认使用 UTF-8 编码。我们正有此意，于是在 sqlite3_exec 的 wrapper 中给每条 SQL 语句转换了编码，至此全部问题解决。

· 收获

- 应在开发初期确定程序所用字符集以及编码。
- 外部库的接口应封装一层，以便参数有误时调试。
- 不要用 C++编译器编译 C 程序。
- 使用宽字符串常量时，不要偷懒使用 L”...”，最好封装一个宏，类似

```
#define TEXT(x) L # x
```

- C 语言是支持枚举类型的。

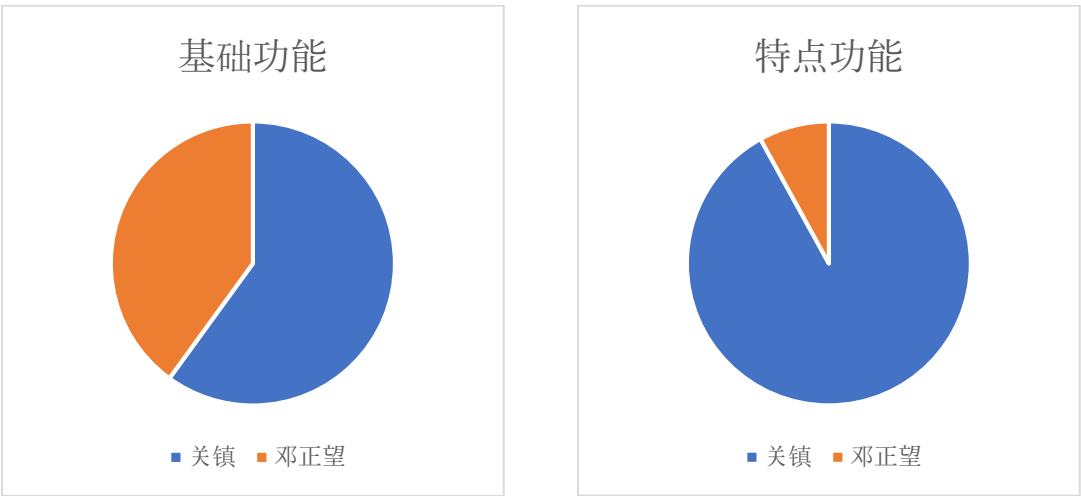
5. 开发、构建/测试环境与组内分工

本程序及其所用开源库，皆由纯 C 语言实现。

*为了实现“数据云同步”功能，在服务器上使用了少量 PHP。

| | |
|---------------|--|
| IDE (Windows) | Microsoft Visual Studio Community 2017 |
| IDE (Linux) | Oracle NetBeans 8.2 |
| 编译器 (Windows) | MSVC (_MSC_VER = 1913) /W4 |
| 编译器 (Linux) | LLVM + Clang 3.8.0 /W4 |
| 开发环境 | Windows 10 (Build 14393) |
| | Windows 7 (Build 7601) |
| 测试环境 | Windows 10 (Build 14393) |
| | Windows XP (SP3) |
| | Ubuntu 16.04 LTS |

组内分工如下。



组员邓正望的分工相对较少，但其工作非常重要。部分分工明细如下。

- 全部函数间调用的传参正确性检查。 (函数的调用)
- char[][]与 char**的误混用检查。 (指针与二维数组)
- 编写 advatoi 函数，字符串转整数。 (指针)
- 从代码层面确保堆上内存得到及时回收。 (手动内存管理)
- 配置项的读操作。 (字符串处理)

- 承担测试组的工作，寻找 BUG 并给出修改建议。
- 参与本文档的编写。

另外，为了使组员能够理解一些特点功能的运作原理，组内在线交流，也约定提前返校见面，初步向组员介绍了如下内容。

- 第三方开源库的使用与遵守开源协议的重要性。
- C 语言中的变长参数。

对应问题：printf 和 scanf 函数为什么能有那么多参数？

- 堆、栈、栈帧与入栈出栈。

*对应问题：为什么静态定义的数组长度最晚在编译时确定？
函数调用是如何完成的？*

- SSL 协议。

对应问题：http 加个 s 是什么意思？

- 一些代码编写规范。
- 几对常用的具有相同作用的 DOS 和 bash 命令。
- Windows 消息机制。
- OllyDbg 的使用。

对应问题：在文档里写的“程序易被爆破从而绕过密码验证”是怎么做到的？

- POSIX 标准。

对应问题：为什么这个程序基本不用修改就能兼容另一种操作系统？

五． 开源许可

各个开源库的开源协议已经内嵌到程序中。

六. 附录

下面包含了本程序的全部头文件定义。由于习惯，注释以英语写成。

程序代码总量约为 4326 行（不含注释、空行），不便全部于文档中展示，请参见源代码。

• NewStd.h

```
/*
    Name:                NewStd.h
    Description:          bool support and other features.
*/

#pragma once

// bool
#include <stdbool.h>

// disable warnings about unreferenced params.
#ifndef UNREFERENCED_PARAMETER
#define UNREFERENCED_PARAMETER(P) (P)
#endif

// wchar_t deprecated due to linux compatible issue.

/*
    #define char char_t
    // make it more convenient to use wide char.
*/
```

• ConfigUtil.h

```
/*
    Name:                ConfigUtil.h
    Description:          Config support.
*/

#pragma once

#include <stdio.h>
#include "NewStd.h"

#define PATH_CONFIG "config.ini"
#define PATH_CONFIGA "config.ini"
#define PATH_CONDS "conds"
#define PATH_CONDSA "conds"
#define defaultConfig(key) config(PATH_CONFIG, key)

#define CU_BUFFER_SIZE 1024
#define CU_TEXT_SIZE 64
```

```

#define CU_INTLEN_SIZE 16

// read a large file line by line.
void massRead(const char* path, void(*Callback)(char* line));

// read config string from file.
char* config(const char* path, const char* key);

// inherits from config.
int getInt(const char* path, const char* key);

// write config to file.
bool configWrite(const char* path, const char* key, const char* value);

// inherits from configWrite.
bool setInt(const char* path, const char* key, int value);

```

• CoreLibrary.h

```

/*
    Name:                CoreLibrary.h
    Description:          Core library management.
*/

#pragma once

#include "sqlite3-secure/sqlite3.h"
#include "NewStd.h"
#include "CoreProgram.h"

#define LM_BUFFER_SIZE 1024
#define PATH_CONDITION_FILE "conds\\"

// order of attrs.
extern int* global_order;

// shared sql object.
extern sqlite3* sql;

// more convenient than a lot of malloc.
char** LM_DynamicArray2Make(int width, int height);
void LM_FreeDynamicArray(void** arr, int n);

// initialize function.
bool LM_Init( void );

// reset database and rebuild attr structure.
bool LM_Reset(bool sure);

// wraps sqlite3_exec.
int LM_SQLExec(
    sqlite3* obj,
    char* exec,
    int(*callback)(void*, int, char**, char**),
    void* preserved,
    char** errmsg);

```

```

// wraps sqlite3_get_table.
int LM_SQLQuery(
    sqlite3* obj,
    char* exec,
    char*** pszResult,
    int* nrow,
    int* ncol,
    char** pszErrMsg);

// fetch all books from sqlite.
void LM_QueryBookAll(
    int* nrow,
    int* ncol,
    char*** result,
    int limit,
    int offset);

// fetch books with custom sqlite exec.
void LM_QueryBookWithSqlExec(
    char* exec,
    int* nrow,
    int* ncol,
    char*** result);

// mark if a book is deleted.
void LM_ChangeBookDeleteStatWithConds(
    int n,
    const char* stat,
    CP_Condition* conds,
    bool AND);

// convert CP_Condition[] to sqlite exec.
void LM_ConditionExecMakeOpt(
    const char* header,
    int n,
    CP_Condition* conds,
    char* result,
    const char* customParam,
    int limit,
    int offset,
    bool AND);

// (generate exec only) - fetch books with CP_Conditions.
void LM_SelectConditionExecMake(
    int n,
    CP_Condition* conds,
    char* result,
    int limit,
    int offset,
    bool AND);

// (generate exec only) - mark some books as deleted.
void LM_DeleteConditionExecMake(
    int n,
    CP_Condition* conds,
    char* result,
    int limit,

```

```

        int offset,
        bool AND);

// (action) - delete books.
void LM_DeleteBooksWithConds(
    int n,
    CP_Condition* conds,
    bool AND);

// (action) - delete books permanently.
void LM_RemoveBooksWithConds(
    int n,
    CP_Condition* conds,
    bool AND);

// (action) - recover deleted books.
void LM_RescueBooksWithConds(
    int n,
    CP_Condition* conds,
    bool AND);

// (action) - modify books.
void LM_UpdateExecMakeWithConds(
    int nConds,
    CP_Condition* conds,
    int n,
    char** modifiedItems,
    char** newValues,
    char* exec);

// fetch a specified book.
void LM_QueryBookWithBookNumber(
    char* bookNumber,
    int* nrow,
    int* ncol,
    char*** result);

// (action) - add books from files in prepared formats.
bool LM_AddBookWithFile(
    char* path,
    int n,
    int* _order,
    int* nsuc,
    int* nfail);
void LM_ConditionFileLoad(
    char* condName,
    CP_Condition* conds,
    int* condition_cnt);

// (action) - add books with given args.
bool LM_AddBookWithOrder(
    char** args,
    int n,
    int* order,
    bool flagUseOrder);

bool UI_BookModifyWithGUI(char* bookNumber);

```

```

bool LM_Add(char** args, int n, bool flagUseOrder);

// generate a CP_Condition file.
void LM_ConditionFileGen(int n, char* condName,
    CP_Condition* conds);

// get a specified value of a book.
void LM_ValueGetWithBookNumber(char* bookNumber, char* key, char** pvalue);
void LM_ValueGetWithBookNumberFree(char** pvalue);

// translates CP_Condition.cond
void LM_CondRulesCat(char* result, char cond);

void LM_GetOrder(int* _order, int n);
void LM_GetOrderWithCondName(const char* condName, int* _order, int n);

// int LM_Callback(void* NotUsed, int argc, char** argv, char** argn);
// deprecated due to a priv violation (directly LM->UI).

```

• AlgorithmUtil.h

```

/*
    Name:            AlgorithmUtil.h
    Description:      Algorithm support.
*/

#pragma once

#include "NewStd.h"
#include <stdlib.h>

#define AU_LINE_SIZE 1024

/*
    Umm... This AlgorithmUtil doesn't have any awesome algorithms.
    coz this program bases on sqlite3 who did everything well...
*/

// converts char[] to wchar_t[].
wchar_t* c2w(char* c);

// convert gb2312 between utf-8.
char* G2U(const char* gb2312);
char* U2G(const char* utf8);

// solve SQLite encoding issue in Linux.
void ascii2Utf8(char* ascii, size_t asciiSize, char* utf8);

// converts int to char[]
void advitoa(int i, char* w);

// has a higher accuracy than strlen :)
int wstrlen(const char* str);

// "1aaa2bb3ccccc4dd5eee" -> 12345

```

```

int advatoi(const char* str);

// they find out a element in an array.
int uniArrIndexOf(void** of, void* with, bool(*Compare)(void* a, void* b), int
limit);
int arrIndexOf(char** of, char* with, int limit);
int arrIndexOfInt(int* of, int with, int limit);

// I wrote this after scanf("[\n]", ) crashed for 1,000,000,000,000,000,000,000
times.
void readline(char* str, int limit);

// drops \r and \n at the end of string.
void trim(char* r, int limit);

// so why doesn't string.h provide split?!
void split(const char* of, const char with, char** ret, int* n);

```

• CoreNetwork.h

```

/*
    Name:                CoreNetwork.h
    Description:          Core network support.
*/

#pragma once

#include "NewStd.h"

#define NU_LINE_SIZE 1024
#define NU_PATH_LENGTH 128
#define NU_SRV_URL "https://mikutart.com/libman/"
#define NU_SRV_HANDLER "handler.php"

bool NU_FileUpload(const char* remoteFileName, const char* localFileName);
bool NU_FileDownload(const char* remoteFileName, const char* localFileName);

```

• CoreProgram.h

```

/*
    Name:                CoreProgram.h
    Description:          Core logics.
*/

#pragma once

#include "NewStd.h"
#include "CoreUI.h"

#define CP_COND_SIZE 64
#define CP_TFORMAT_SIZE 4
#define CP_NATTR_COUNT 64
#define CP_NATTR_SIZE 64
#define CP_FORMAT_SIZE 64
#define CP_COND_NAME_SIZE 64
#define CP_COND_VALUE_SIZE 64

```

```

#define CP_EXEC_SIZE 1024

#define LIC_PATH_SQLITE \
"licenses\\LICENSE_SQLITE.txt"
#define LIC_PATH_LIBCURL \
"licenses\\LICENSE_LIBCURL.txt"
#define LIC_PATH_OPENSSL \
"licenses\\LICENSE_OPENSSL.txt"
#define LIC_PATH_APACHEV2 \
"licenses\\LICENSE_APACHEV2.txt"

#define LPADDING printf(" ");
#define TPADDING printf("  ");
#define NEWLINE putchar('\n');

// a condition includes a name, an operator(cond) and a value.
typedef struct {
    char name[CP_COND_NAME_SIZE];
    char cond;
    char value[CP_COND_VALUE_SIZE];
} CP_Condition;

// (view) - book management.
void CP_BookManage( void );

// (view) - a command line for custom sqlite3 exec.
void CP_AnySqlCmd( void );

// (view) - show open source licenses.
void CP_OpenSourceLic( void );

// (method) - condition make.
void CP_ConditionMake(
    CP_Condition* target,
    const char* name,
    const char op,
    const char* value);

// (view) - sell mode.
void CP_BuyMode( void );
void CP_BuyModeCheck( void );

// (navigation) - add a condition to advanced filter.
void CP_AddCondition(CP_Condition cond);

// (method) - add a condition to advanced filter.
void CP_ConditionCpy(CP_Condition* dst, CP_Condition* src);

// (view) - main view.
void CP_Main( void );

// (view) - cloud service.
void CP_CloudService( void );

// (view) - settings.
void CP_Settings( void );

```



```

// (view) - books with deleted tag.
void CP_TrashBin( void );

// (navigation) - switch language file.
void CP_SwitchLanguage( void );

// (view) - advanced filter.
void CP_AdvancedFilter(void(*from)( void ));

// (method) - filter.
void CP_FilterWithExistingConds(
    bool AND,
    int offset,
    bool flagShowDeleted);

// (view) - borrow mode.
void CP_BorrowMode( void );

// (view) - search.
void CP_Search( void );

// (navigation) - delete one.
void CP_DeleteOne( void );

// (navigation) - export to excel.
void CP_ExportExcelNavi(char* addition, bool AND);

// (navigation) - add one.
void CP_AddBook( void );

// (navigation) - recover one.
void CP_RescueOne( void );

// (navigation) - reset database.
void CP_ResetDatabase( void );

// (navigation) - mass add.
void CP_BookImport( void );

// (method) - mass recover.
void CP_RescueAllWithExistingConds(bool AND);

// (method) - mass delete.
void CP_DeleteAll(bool AND);

// (navigation) - delete a condition.
void CP_DeleteFilter( void );

// (navigation) - import condition file.
void CP_LoadConditionFile( void );

```

• CoreSecure.h

```

/*
    Name:                CoreSecure.h
    Description:          Security question and password management.
*/

```

```
#pragma once

#include <string.h>
#include "NewStd.h"

// check password.
bool SM_MasterCodeCheck(char* uCode);

// update password.
bool SM_MasterCodePush(char* uCode);

// check security question.
bool SM_SQAnswerCheck(char* uAnswer);

// update security question.
bool SM_SQQuestionPush(char* uQuestion);
bool SM_SQAnswerPush(char* uAnswer);

// read security question.
void SM_SQQuestionGet(char* buffer);
```

- CoreUI.h

```
/*
    Name:                CoreUI.h
    Description:          Simulate UI in CLI.
*/

#pragma once

#include <stdio.h>
#include <stdlib.h>

#include "I18N.h"

#define UI_MENU_HEIGHT 64
#define UI_MENU_WIDTH 512
#define UI_ATTR_FORMAT_SIZE 64

// clear CLI.
// Windows: cls
// Linux: clear
void UI_Clear( void );

// set locale to compatible with unicode charset.
void UI_InitWithUnicodeCS( void );

// read a line from CLI and convert to int with advatoi.
int UI_Input( void );

// get command to clear CLI in current operating system.
const char* clearCmd( void );

// get command to list directory in current operating system.
// Windows: dir
// Linux: ls
const char* lsCmd(void);

// print array in menu format.
void UI_MenuMake(const char* title, int num, ...);

// only works on Debug mode.
void dbgprintf(const char* str);

// add book navigation.
void UI_AddBookNavi( void );

// print book attrs table header.
void UI_PrintBookHeader( FILE* target );

// advanced printf which automatically use str().
void UI_Printf(const char* format, ...);
void UI_Fprintf(FILE* target, const char* format, ...);

// simulate a full-screen MessageBox in CLI.
void UI_MessageBoxWithType(const char* msg, const char* caption, bool tryGUI,
unsigned int type);
```

```

void UI_MessageBox(const char* msg, bool tryGUI);

// check password.
bool UI_MastercodeCheck(bool flagFullScreen);

// check security question.
bool UI_SQCheck( void );

// wraps UI_MastercodeInputWithCustomMsg().
void UI_MastercodeInputSolution(char*
    uPassword, bool tryGUI);

// input password in CLI. (uses _getch())
void UI_MastercodeInputWithCustomMsg(
    char* msgTag,
    char* uPassword);

// another way to print a menu.
void UI_MenuMakeWithArray(
    const char* title,
    int num,
    const char** arr);

// wraps UI_MenuMakeWithArray().
char** UI_MenuMakeWithString(
    const char* title,
    const char* str);

// analyze and print sqlite3 result.
void UI_ListMakeWithSqlite3Array(
    const char* title,
    void(*HeaderProcessor)(FILE*),
    int nrow,
    int ncol,
    bool flagShowDeleted,
    int base,
    char** arr,
    void(*callback)(char*, FILE*),
    FILE* target);

// a switch navigation.
void UI_SwitchMake(
    const char* name,
    const char* intro,
    const char* key);

// a setting navigation.
void UI_SettingsMake(
    const char* name,
    const char* intro,
    const char* key,
    bool InputOrSwitch);

// determines raw input or select from given list for each attr.
void UI_AttributedInput(int index, char* name, char* target);

// ask for yes or no.
bool UI_Prompt(const char* msg, bool tryGUI);

```

- I18N.h

```
/*
    Name:                I18N.h
    Description:          a solution for internationalization.
*/

#pragma once

#include "ConfigUtil.h"

#define LANG_PATH "languages/%d%hs"

#define FILE_NAME_SIZE 64
#define LINE_SIZE 1024

extern int global_language;

// switch language.
bool switchLanguage(int language);

// read string from corresponding language file.
char* str(const char* key);
```