

**数据库课程设计报告**

**系 （院）：**  计算机科学学院

**专业班级：** 软工11402班

**姓 名：** 彭程建

目录

1. 课程设计目的........................................3
2. 设计任务及要求......................................3

三、系统概述............................................3

3.1系统边界......................................................3

3.2需求概要......................................................3

3.3系统分析目标..................................................4

四、总体设计............................................13

4.1客户端设计....................................................13

4.2数据库设计....................................................14

4.2.1 需求分析...................................................14

4.2.2 数据库物理结构分析......................................15

4.2.3 数据库概念结构设计......................................15

4.2.4 数据库逻辑结构设计......................................15

五、详细设计与实现[含代码和实现界面].....................17

5.1 客户端........................................................17

5.1.1 表示层（GUI）...........................................17

5.1.2 业务逻辑层..............................................21

5.1.3 数据访问层..............................................30

5.1.4 实体类..................................................31

5.2 数据库........................................................31

六、课程设计小结........................................33

6.1 体会..........................................................33

6.2 参考文献......................................................33

**一、课程设计目的**

通过对图书管理系统的系统分析、系统设计、编码和调试等工作的实践，熟悉管理信息系统的开发过程、设计方法及相关编程技术，熟练掌握数据库设计的基本理论及方法。

**二、设计任务及要求**

要求完成一个具有一定实用价值的图书管理系统，主要任务包括：

在Microsoft SQL Server 2000/2005/2008环境下建立图书管理系统所使用的数据库，利用企业管理器或查询分析器建立各种数据库对象，包括：数据表、视图、约束、存储过程和触发器等；

掌握JDBC编程技术，对MS SQL Server数据库进行连接和操纵；

掌握使用Java语言开发一个数据库应用系统的基本方法和步骤，熟悉一些基础功能的实现方法，如：数据维护（插删改等操作），数据查询、浏览和Excel导出，统计与报表，用户登录和权限管理等。

了解C/S或B/S应用程序的多层体系结构及三层架构方案设计思想，了解迭代式开发，熟悉面向对象设计方法及其分析与设计过程，了解UML文档及其开发过程中的作用。

**三、系统概述**

**系统边界与约定**

* 1. 系统限定在实体书库的借阅和管理等业务范围；
  2. 不考虑图书馆的电子书库、订购、情报、人事管理等业务；
  3. 不考虑图书馆的跨区分布情况，如长江大学图书馆包括多个校区图书馆；
  4. 不考虑图书的通借通还，如长江大学读者可在湖北省高校任何一家图书馆借还图书；
  5. 不考虑珍藏图书的借阅业务；
  6. 暂不考虑与校园一卡通系统的外部接口。一卡通系统为外部系统（外部参与者），卡内记录有身份及相关信息，该系统负责身份验证工作。
  7. 期刊库和论文库的借阅和管理等业务可作为二期项目目标，视本系统使用情况而定。

**需求概要**

(1) 图书管理系统的基本功能需求包括：读者管理、图书管理、借阅管理、用户登录与用户管理等；其它功能需求包括：读者查询与预约借书\*、统计与报表\*、数据备份\*、书架管理\*、期刊管理\*、期刊借阅\*、论文管理\*、论文借阅\*等。（\* 表示留待以后的迭代周期完成，下同）

(2) 相关领域概念：借书证（读者）、图书、借还书记录。

(3) 系统外部参与者：读者、借书证管理员、图书管理员、借阅管理员、系统管理员。将图书馆工作人员划分为借书证管理、图书管理、借阅管理等三类人员；系统管理员负责数据库和软件系统管理，包括各类管理员用户的创建和授权、数据库备份等工作。

**系统分析目标**

找出系统**用例**，书写用例文档，建立**领域模型**(概念模型)。

[**用例]** use case，或译为**使用案例**、**用况**。是指为了完成一个领域目标或任务，提供一个或多个场景来说明其在系统内部与外部（人员或外部系统等）之间的交互过程。如办理借书证、借书、还书、用户登录等均为用例；而借阅管理不是用例，因为它包括多个任务；读者可否借书也不是用例，因为它不是一个领域任务或业务目标。参见“参考资料01 用例.doc”。

用例是一个**事件**流，一个事件对应一个**系统操作**；如借书用例中的“根据借书证号查询读者信息”、“根据借书证号查询超期未归还图书”、“判断读者可否借书”、“判断图书是否在馆”、“确认借书”等均为事件或系统操作。

找出系统用例后，对每个用例的交互操作过程进行描述（即书写用例），通过用例发现领域概念及其属性，并建立领域模型（即概念模型）。用例与领域模型的建立过程一般是交互进行的，并相互参考、印证和补充完善。

通过用例发现事件或系统操作，可进行下一步的设计和实现工作，是系统设计主要内容。

在面向对象方法的开发过程中，分析与设计没有明显的界限，只是侧重面不同而已。

### 5.1 读者管理

读者管理即借书证管理，包括的业务（即用例）有：办理借书证、借书证变更、借书证挂失\*、解除挂失\*、补办借书证\*、注销借书证\*、批量办理新生借书证\*等。（\*表示留待迭代2及以后完成，下同。）

借书证（读者）可分为2种类别：教师、学生。

借书证（教师）=借书证号、姓名、性别、所在单位、办证日期、照片等。

借书证（学生）=借书证号、学号、姓名、性别、专业、班级、办证日期、有效期、照片等。其中，有效期由学生类别决定，本科生4年、专科生3年、硕士生3年等。

相关业务规则：(1)读者凭借书证借书；(2)教师最多借书12本，借书期限最长为60天，可续借2次；学生最多借书8本，借书期限最长为30天，可续借1次；(3)处于挂失、注销状态的读者不能借书；(4)未归还图书者不能注销其借书证。

分析：增加读者类别概念，统一并简化读者信息，采用教师属性项，而学生可在所在单位填写班级，学生特有读者信息可留待以后的开发周期处理。可得到2个概念：读者、读者类别。

读者=借书证号、姓名、性别、所在单位、读者类别、办证日期、照片等。（另可加：电话、邮箱等）

读者类别=读者类别号、类别名称、可借书本数、可借书天数、可续借次数。

思考：学生借书证的有效期如何处理？

**1. 办理借书证**

|  |
| --- |
| 用例名称：办理借书证 |
| 迭 代：1 （说明：第1次迭代期间的用例文档） |
| 参 与 者：借书证管理员（或系统管理员）、读者 |
| 综 述：用例起始于读者来到借书证管理办公室，管理员给读者办理新借书证。 |
| 触 发 器： （说明该用例的触发条件） |
| 前置条件：登录用户具备读者管理权限。 |
| 过程描述： （标题名或为：基本流程、基本事件流、主要成功场景等） |
| 1. 管理员输入：读者类别、所在单位、姓名。 |
| 2. 点击“查询”。 |
| 3. 系统查询读者信息，并显示查询结果。 |
| 4. 管理员判断该读者是否已办理借书证，若是，则可取消办证过程。（人工操作） |
| 5. 管理员输入读者信息：读者类别、姓名、性别、所在单位、办证日期、照片等。（类别、姓名、单位等可预置为过程1中输入值，办证日期可预置为系统日期） |
| 6. 确认办证。 |
| 后置条件：创建新的读者对象，并具有新的借书证号；系统记录读者信息。 |
| 业务规则：借书证号应具有唯一性。 |
| 待解决问题： |
| 1. 打印借书证； |
| 注 释：未考虑校园一卡通做为借书证使用的情况 |
| 作者与日期： 2013-7-24日，LSH，版本号1.0 |

**2. 变更借书证**

变更借书证类别、有效期、所在单位等，以及其它信息；借书证号不可更改。

用例

|  |
| --- |
| 用例名称：变更借书证 |
| 迭 代：1 |
| 参 与 者：借书证管理员（或系统管理员）、读者 |
| 综 述：用例起始于读者来到借书证管理办公室，管理员给读者办理变更借书证业务。 |
| 触 发 器： |
| 前置条件：用户具备读者管理权限，且读者已经存在 |
| 过程描述： （标题名或为：基本流程、基本事件流、主要成功场景等） |
| 1. 管理员输入：读者类别、所在单位、姓名。 |
| 2. 点击“查询”。 |
| 3. 系统查询读者信息，并显示查询结果。 |
| 4. 管理员判断该读者是否已办理借书证，若是否，则可取消办证过程。（人工操作） |
| 5. 管理员更改读者信息：借书证类别、有效期、所在单位等，以及其它信息（办证日期可预置为系统日期） |
| 6. 确认变更借书证。 |
| 后置条件：创建新的读者对象，并具有新的借书证类别、有效期、所在单位等；系统记录读者信息。 |
| 业务规则：借书证号不可更改。 |
| 待解决问题： |
| 1. 打印借书证； |
| 注 释：未考虑校园一卡通做为借书证使用的情况 |
| 作者与日期： 2016-11-24日，Seven，版本号1.0 |

**3. 挂失借书证\***

|  |
| --- |
| 用例名称：挂失借书证 |
| 参 与 者：借书证管理员、读者 |
| 综 述：用例起始于读者来到借书证管理办公室，管理员给读者办理借书证挂失。 |
| 前置条件：登录用户具备读者管理权限，且读者已经存在 |
| 过程描述： |
| 1. 输入借书证号，或输入读者的类别、单位、姓名。 |
| 2. 点击“查询”。 |
| 3. 系统查询并显示读者信息。 |
| 4. 找到其借书证，且读者状态为有效。 |
| 5. 确认挂失。 |
| 后置条件：修改读者状态为挂失；系统记录读者信息。 |
| 业务规则：借书证挂失后禁止读者借书。 |

分析：修改读者概念，增加属性“借书证状态”；并修改办理借书证用例。

思考：如果待挂失的读者有未还的书，应该怎么处理？

**4. 解除挂失\***

|  |
| --- |
| 用例名称：解除挂失 |
| 参 与 者：借书证管理员、读者 |
| 综 述：用例起始于读者来到借书证管理办公室，管理员给读者办理解除借书证挂失。 |
| 前置条件：登录用户具备读者管理权限，且读者已经存在 |
| 过程描述： |
| 1. 输入借书证号，或输入读者的类别、单位、姓名。 |
| 2. 点击“查询”。 |
| 3. 系统查询并显示读者信息。 |
| 4. 找到其借书证，且读者状态为挂失。 |
| 5. 确认解除挂失。 |
| 后置条件：修改读者状态为有效；系统记录读者信息。 |

**5. 补办借书证\***

补办借书证，办理新借书证（新借书证号，其它信息不变），并将原借书证的借阅记录转移到新证上，原借书证注销。

用例（略）

**6. 注销借书证\***

用例（略）

**7. 批量办理新生借书证\***

从教务管理系统导入新生信息，系统生成借书证信息，管理员核实后办理借书证。

用例（略）

**8. 读者类型管理**

从系统的功能完整性来看，还存在一个隐性的功能需求：读者类型管理，即读者类型信息的插、删、改、查等数据维护操作。

用例（略 要求自己完成）

|  |
| --- |
| 用例名称：读者类型管理 |
| 迭 代：1 |
| 参 与 者：借书证管理员（或系统管理员） |
| 综 述：用例起始于管理员来到借书证管理办公室，管理员给读者办理读者类型管理业务。 |
| 触 发 器： |
| 前置条件：借书证管理员（或系统管理员）具备读者管理权限，且读者已经存在 |
| 过程描述： （标题名或为：基本流程、基本事件流、主要成功场景等） |
| 1. 管理员登陆借书证管理系统 |
| 2. 管理员输入：读者类别、所在单位、姓名。 |
| 3. 点击“查询”。 |
| 4. 系统查询读者信息，并显示查询结果。 |
| 5. 管理员进行读者类型信息的插、删、改、查等数据维护操作 |
| 6. 确认变更借书证。 |
| 后置条件：修改读者状态为有效；系统记录读者信息。 |
| 业务规则：借书证号不可更改。 |
| 注 释：未考虑校园一卡通做为借书证使用的情况 |
| 作者与日期： 2016-11-24日，Seven，版本号1.0 |

### 5.2 图书管理

包括业务（用例）：图书编目、新书入库、图书信息维护、图书变卖与销毁处理\*等。

图书信息=书号、书名、作者、出版社、出版日期、ISBN、分类号、语言、页数、单价、内容简介、图书封面、图书状态等；（图书状态包括：在馆、借出、遗失、变卖、销毁）

**1. 图书编目**

新书入库前，图书管理员对图书进行图书编目工作，即根据该馆的现有图书分类目录信息编写新书的分类号，需要时根据图书分类法增加新的图书分类条目。

图书信息= … 、分类号、…

图书分类目录=分类号、分类名。图书分类信息主要用于统计，如中文图书分类条目：

-----------------------------------------------------------

分类号 分类名

TP 自动化技术、计算机技术

TP3 计算技术、计算机技术

TP31 计算机软件

TP311 程序设计、软件工程

TP312 程序语言、算法语言

TP311.13 数据库理论与系统

TP311.131 数据库理论

新书分类号，如图书“C#高级编程(第7版)”可编为：TP312/3033.2

用例名称：图书编目（略）

**2. 新书入库**

|  |
| --- |
| 用例名称：新书入库 |
| 参 与 者：图书管理员 |
| 前置条件：登录用户具备图书管理身份和权限 |
| 过程描述： |
| 1. 输入或系统生成起始书号(BkNO)。 |
| 2. 输入：图书信息(书名、作者、出版社、出版日期、语言、页数、单价、ISBN、分类号、内容简介等)、入库数量(N，N>=1)、入馆日期。 |
| 3. 确认图书入库。 |
| 4. 图书馆工作人员对图书进行贴书签（分类号）、印书号（或条形码）、盖图书馆印章。 |
| 5. 图书上架。 |
| 后置条件：创建N个图书对象，书号分别为BkNO、BkNO+1、…、BkNO+N-1，这N本图书的状态均修改为在馆；系统记录图书信息。 |
| 注 释：BkNO+N可作为下一次新书入库的起始书号。 |

思考：图书信息划分为2个概念是否更为合理？

馆藏图书目录=书名、作者、出版社、出版日期、ISBN、分类号、语言、页数、单价、内容简介、图书封面、入库数量、入馆日期

图书=书号、分类号、图书状态

**3. 图书信息维护**

图书信息的查、删、改等信息维护操作。

用例名称：图书信息维护（略 自己完成，提示：先查找，再删改）

|  |
| --- |
| 用例名称：图书信息维护 |
| 参 与 者：图书管理员 |
| 前置条件：登录用户具备图书管理身份和权限 |
| 过程描述： |
| 1. 输入图书的相关信息。 |
| 2. 检索需要维护的准确书籍。 |
| 3. 管理员对图书信息的查、删、改等信息进行维护操作。 |
| 4. 管理员保存相关修改信息。 |
| 5. 图书信息修改。 |
| 后置条件：系统记录图书信息。 |

**4. 在馆图书变卖与销毁处理**\*

长期没有借阅记录且失去保存价值的图书可以进行变卖或销毁处理。

用例名称：在馆图书销毁和变卖处理（略）

### 5.3 借阅管理

包括业务用例：借书、续借、还书等。还书过程涉及超期罚款、遗失图书罚款等业务规则。

罚款规则：（1）超期罚款规则 应罚款金额=超期天数\*罚款率，罚款率=0.05元/天，罚款率可能随时间或读者类别而变化；实际罚款金额<=应罚款金额，根据实际情况可以进行减免。（2）遗失罚款规则 遗失图书应罚款金额=3\*图书单价；实际罚款金额在（1\*图书单价，3\*图书单价）之间。（3）遗失罚款规则优先于超期罚款规则。

借书记录=借书证号、书号、借书操作员、借书日期、应还日期

续借记录=借书证号、书号、续借操作员、续借日期、应还日期，续借次数

还书记录=借书证号、书号、还书操作员、还书日期、应还日期，超期天数、应罚款金额，实际罚款金额

分析：借书记录、续借记录、还书记录等信息可合并为借阅记录，续借可舍去部分信息。借阅历史记录应该长期保存在数据库中，以便于统计分析，另外，读者对一本书可能多次借阅，故借阅信息应该增加一个标识项：借书顺序号。

借阅信息=借书顺序号、借书证号、书号、借书操作员、借书日期、应还日期，续借次数、还书操作员、还书日期，超期天数、应罚款金额，实际罚款金额

**1. 借书**

|  |
| --- |
| 用例名称：借书 |
| 参 与 者：借阅管理员、读者 |
| 综 述：用例起始于读者带着图书来到图书借阅大厅，管理员给读者办理借书手续。 |
| 前置条件：登录用户具备借阅管理权限。 |
| 过程描述： |
| 1. 管理员输入借书证号，点击“查询读者”。 |
| 1. 系统查询读者信息、未归还图书信息（含超期），并进行显示。 |
| 1. 系统判断读者可否借书（借书证状态为有效，已借书数量小于可借书数量，不存在超期未归还图书）。若不可借书，则禁止借书。 |
| 1. 管理员输入待借图书的书号，点击“查询图书”。 |
| 1. 系统查询，并显示图书信息。 |
| 1. 系统判断图书是否在馆，若不在馆，则禁止借书。 |
| 1. 管理员点击“借书”。 |
| 后置条件：创建借阅记录对象（借书顺序号由系统自动产生，借书证号和书号为上述输入值，借书操作员=登录用户，借书日期=系统日期，应还日期=系统日期+可借书天数，续借次数=0，还书日期=NULL，还书操作员=NULL），并标记为未归还；修改读者对象的已借书数量+1；修改图书状态为借出；系统记录借阅对象、读者对象、图书对象。 |
| 业务规则：(1)借书证状态为挂失、注销者不能借书；(2)借书数量不能超过可借书数量；(3)有超期未归还图书者不能借书。 |
| 注 释：未归还图书状态可采用还书日期为空（NULL）来标记，或另加属性表示是否还书。 |

修改读者信息：增加“已借书数量”；修改用例办理借书证：后置条件中，已借书数量=0。

练习：修改用例以满足读者携带多本图书进行借书的要求。

**2. 续借**

|  |
| --- |
| 用例名称：续借 |
| 参 与 者：借阅管理员、读者 |
| 前置条件：登录用户具备借阅管理权限、图书状态为借出 |
| 过程描述： |
| 1. 输入待续借图书书号。 |
| 1. 系统查询并获取图书对象、未归还状态的借阅记录对象、及其读者对象，并进行显示。 |
| 1. 系统判断可否续借（续借次数<可续借次数, 读者状态为有效）。 |
| 1. 点击“续借”。 |
| 后置条件：修改借阅记录对象（续借次数+1，应还日期+=可借书天数），图书状态为借出，并保存到数据库中。 |
| 业务规则：(1)借书证状态为挂失、注销者不能续借；(2)续借次数不能超过可续借次数。 |
| 注 释：续借前后图书状态应为借出 |

**3. 还书**

还书用例需要处理超期罚款、遗失图书罚款处理2种情况。

用例名称：归还图书（略 要求自己完成，完成罚款处理者，教师可酌情加分）

|  |
| --- |
| 用例名称：还书 |
| 参 与 者：借阅管理员、读者 |
| 综 述：用例起始于读者带着图书来到图书借阅大厅，管理员给读者办理还书手续。 |
| 前置条件：登录用户具备借阅管理权限。 |
| 过程描述： |
| 1. 管理员输入借书证号，点击“查询读者”。 |
| 1. 系统查询读者信息、未归还图书信息（含超期），并进行显示。 |
| 1. 管理员提醒读者是否存在过期未归还的图书。 |
| 1. 管理员输入待还图书的书号，点击“查询图书”。 |
| 1. 系统查询，并显示图书信息。 |
| 1. 系统判断读者是否需要进行罚款处理。 |
| 1. 管理员点击“还书”。 |
| 后置条件：修改借阅记录对象（还书操作员=登录用户，还书日期=系统日期），并标记为已归还；修改读者对象的已借书数量-1；修改图书状态为在库；系统记录借阅对象、读者对象、图书对象。 |
| 业务规则：(1) 教师超期归还图书加重罚款。 |

### 5.4 用户登录与用户管理

包括用例：用户登录、密码修改、用户管理\*，为本系统的基础和主要功能。

用户包括2类：读者、管理员。其中，管理员用户权限是4种角色的组合：借书证管理、图书管理、借阅管理、系统管理；系统管理员负责所有管理员用户及其权限的管理，借书证管理员负责读者管理（即借书证管理）。

管理员是读者，但读者不一定是管理员；读者与管理员间存在(1对0..1)联系。

读者信息+=密码。

管理员信息=用户号、用户名、密码、管理角色

管理角色设计：可采用4位二进制，借书证管理(0001)2=1、图书管理(0010)2=2、借阅管理(0100)2=4、系统管理(1000)2=8。如表示图书管理和借阅管理权限：2+4=6；判断7是否具备图书管理权限：7位与2，即(0111)2位与(0010)2=(0010)2，表示有此权限。

分析：将管理员信息合并到读者中，可简化概念，此时，非管理员的管理角色(0000)2=0。好处：简化用户登录用例；坏处：概念不够清晰，给用户管理带来不便。第1个迭代周期采用简化概念设计。另外也可用4个逻辑数据项分别表示4个管理角色。

读者信息+=密码、管理角色。需修改用例办理借书证（自己完成）。

**1. 用户登录**

|  |
| --- |
| 用例名称：用户登录 |
| 参 与 者：读者或管理员 |
| 前置条件：无 |
| 过程描述： |
| 1. 输入用户号（即借书证号）、密码；密码采用掩码“\*”号显示。 |
| 2. 点击“登录”。 |
| 3. 系统根据借书证号获取读者对象。 |
| 3a. 未获取读者对象时，显示无此用户，转入1； |
| 3b. 密码未匹配时，显示密码错误，并清空密码，转入1； |
| 4. 进入系统主程序界面。 |
| 后置条件：根据登录用户权限显示或隐藏相应的系统功能（菜单等）。 |
| 待解决问题： |
| 1. 挂失或注销借书证者不能登录。 |
| 注 释：该登录用户（读者）对象应为全局可见 |

**2. 密码修改**

|  |
| --- |
| 用例名称：修改密码 |
| 参 与 者：用户 |
| 综 述：用户登录后修改自己的登录密码。 |
| 前置条件：用户已经成功登录 |
| 过程描述： |
| 1. 点击“修改密码” |
| 2. 输入现有密码，新密码，确认新密码。 |
| 3. 系统根据现有密码，判断用户能否修改密码。 |
| 4. 现有密码未匹配时，显示现有密码错误，并清空现有密码，新密码转入2。 |
| 5. 系统修改用户密码； |
| 后置条件：修改读者的密码，系统记录读者信息。 |
| 待解决问题： |
| 1. 挂失或注销借书证者不能登录。 |
| 注 释：该登录用户（读者）对象应为全局可见 |

**3. 用户管理\***

|  |
| --- |
| 用例名称：用户管理 |
| 参 与 者：系统管理员、图书馆工作人员 |
| 综 述：系统管理员对图书馆工作人员进行权限管理。 |
| 前置条件：登录用户为系统管理员，图书馆工作人员需先办理借书证 |
| 过程描述： |
| 1. 输入借书证号，或输入读者的类别、单位、姓名。 |
| 2. 点击“查询”。 |
| 3. 系统查询，并显示读者信息。 |
| 4. 找到借书证后，设置该读者的管理角色。 |
| 5. 确认完成。 |
| 后置条件：修改读者的管理员角色，系统记录读者信息。 |
| 注 释： 管理角色有4种：借书证管理、图书管理、借阅管理、系统管理 |

### 5.5 读者查询与预约借书\*

读者（非管理员用户）的功能需求，包括用例：未归还图书查询与续借\*（含超期、即将到期查询操作）、预约借书\*（含在馆图书查询操作）等。

该类读者所使用的功能最好是采用web程序实现，Web程序功能可包括：读者登录、密码修改、未归还图书查询与续借\*、预约借书\*等。

用例名称：未归还图书查询与续借（略，思考：与续借用例有不同吗？）

用例名称：预约借书（略，思考：对借书用例有何影响？）

### 5.6 统计与报表\*

包括：馆藏图书统计、借阅情况统计、借阅时段统计等。

馆藏图书统计可按图书类别、入馆时间统计图书数量、金额等，可得图书分布情况；借阅情况统计可按单位（含专业、年级）、图书类别、借阅时间（年、学期、月份）等统计借阅次数和天数，统计结果可作为订购新书的参考依据；借阅时段统计按工作日（星期一、二、、、）和时段（8:00-8:30、8:30-9:00、、、）统计借还书次数，得借阅工作量，并可依此进行人员安排。

用例（略）

### 5.7 数据备份\*

用户备份是对SQL Server数据库进行备份，由系统管理员操作。

### 5.8 系统功能与用户角色

表 1 系统功能与用户角色关系表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **系统功能** | **借书证管理** | **图书管理** | **借阅管理** | **系统管理** | **读者** |
| 1 | 读者管理 | √ |  |  | （√） |  |
| 2 | 图书管理 |  | √ |  |  |  |
| 3 | 借阅管理 |  |  | √ |  |  |
| 4 | 用户登录 | √ | √ | √ | √ | √ |
| 5 | 密码修改 | √ | √ | √ | √ | √ |
| 6 | 用户管理 |  |  |  | √ |  |
| 7 | 读者预约 | √ | √ | √ | √ | √ |
| 8 | 统计与报表 | √ | √ | √ |  | √ |
| 9 | 数据备份 |  |  |  | √ |  |

### 5.9 领域模型

**领域概念小结**

读者=读者号（借书证号）、姓名、性别、所在单位、电话、邮箱地址、办证日期、照片、借书证状态、已借书数量、密码、管理角色。（借书证状态：有效、挂失、注销）

读者类别=读者类别号、读者类别名称、可借书数量、可借书天数、可续借次数，罚款率，证件有效期。

图书=书号、书名、作者、出版社、出版日期、ISBN、分类号、语言、页数、单价、内容简介、图书封面、图书状态。（图书状态包括：在馆、借出、遗失、销毁、卖出）

借阅信息=借书顺序号、读者号、书号、借书操作员、借书日期、应还日期、续借次数、还书操作员、还书日期，超期天数、应罚款金额、实际罚款金额。

**领域模型**

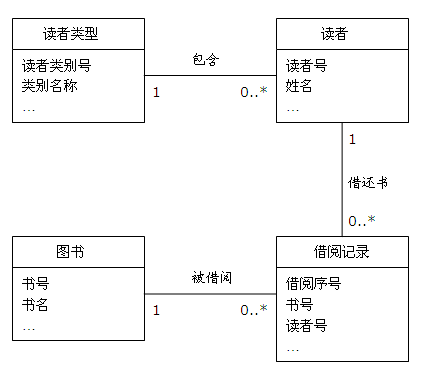


图 1 图书管理系统概念模型(V1)

**四、总体设计**

**1、客户端设计（三层架构）**

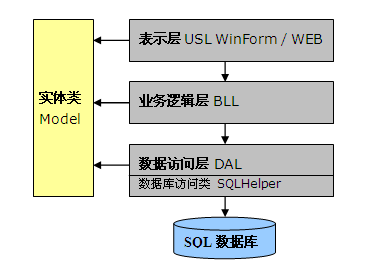


图 2 三层架构示意图

**表示层（USL）**：也称UI，提供交互式界面，形式：Application界面。

**业务逻辑层（BLL）：**实现业务功能，为表示层提供服务，形式：类库。

**数据访问层（DAL）：**实现数据访问功能（如数据库、文件等数据的读取、保存和更新），为业务逻辑层提供服务，形式：类。edu.yangtzeu.lmis.dal.ConnectDB类提供了对SQL Server数据库的一般访问方法。

**实体类(Model)**：描述一个业务实体的类，也即应用系统所涉及的业务对象。对数据库来讲，每个数据表对应于一个实体类，数据表的每个字段对应于类的一个属性。

表示层、业务逻辑层、数据访问层都依赖于业务实体。各层之间数据的传递主要是实体对象，业务信息封装在实体对象中。

**2、数据库设计**

**①需求分析**

(1)图书管理系统的基本功能需求包括：读者管理、图书管理、借阅管理、用户登录与用户管理等；其它功能需求包括：读者查询与预约借书\*、统计与报表\*、数据备份\*、书架管理\*、期刊管理\*、期刊借阅\*、论文管理\*、论文借阅\*等。（\* 表示留待以后的迭代周期完成）

(2)相关领域概念：借书证（读者）、图书、借还书记录。

(3)系统外部参与者：读者、借书证管理员、图书管理员、借阅管理员、系统管理员。将图书馆工作人员划分为借书证管理、图书管理、借阅管理等三类人员；系统管理员负责数据库和软件系统管理，包括各类管理员用户的创建和授权、数据库备份等工作。

**②数据库物理结构分析**

数据库在物理设备上的存储结构与存取方法称为数据库的物理结构，它依赖于选定的数据库管理系统。为一个给定的逻辑数据模型选取一个最适合的应用要求的物理结构的过程，就是数据库的物理设计。 数据库的物理设计通常分为两步：

（1）确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构； （2）对物理结构进行评价，评价的重点是时间和空间的效率。

**●关系模式存取方法的选择** 确定数据库的存取方法，就是确定建立哪些存储路径以实现快速存取数据库中的数据。现行的DBMS一般都提供了多种存取方法，如索引法、HASH法等。其中，最常用的是索引法，本系统也采用的是索引法。我们在经常需要搜索的列和主关键字上建立了唯一索引。

**●确定数据库的存储结构** 由于不同PC机所安装的数据库软件位置不一定相同,所以数据文件与日志文件的存放位置也不一定相同。

**③数据库概念结构设计**

**本系统E-R图：**

**④数据库逻辑结构设计**

1. 读者类别表(TB\_ReaderType)

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 说明 |
| 1 | rdType | SmallInt | 读者类别【主键】 |
| 2 | rdTypeName | Nvarchar(20) | 读者类别名称【唯一、非空】 |
| 3 | CanLendQty | Int | 可借书数量 |
| 4 | CanLendDay | Int | 可借书天数 |
| 5 | CanContinueTimes | Int | 可续借的次数 |
| 6 | PunishRate | Float | 罚款率（元/天） |
| 7 | DateValid | SmallInt | 证书有效期（年）【非空，0表示永久有效】 |

1. 读者信息表(TB\_Reader)

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 说明 |
| 1 | rdID | Int | 读者编号/借书证号【主键】 |
| 2 | rdName | nvarchar(20) | 读者姓名 |
| 3 | rdSex | nchar(1) | 性别，男/女 |
| 4 | rdType | SmallInt | 读者类别【外键TB\_ReaderType】【非空】 |
| 5 | rdDept | nvarchar (20) | 单位代码/单位名称 |
| 6 | rdPhone | nvarchar(25) | 电话号码 |
| 7 | rdEmail | nvarchar(25) | 电子邮箱 |
| 8 | rdDateReg | date | 读者登记日期/办证日期 |
| 9 | rdPhoto | image | 读者照片 |
| 10 | rdStatus | nchar(2) | 证件状态，3个：有效、挂失、注销 |
| 11 | rdBorrowQty | Int | 已借书数量(缺省值0) |
| 12 | rdPwd | nvarchar (20) | 读者密码(初值123)，可加密存储 |
| 13 | rdAdminRoles | SmallInt | 管理角色，0-读者、1-借书证管理、2-图书管理、4-借阅管理、8-系统管理，可组合 |

1. 图书信息表(TB\_Book)

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 说明 |
| 1 | BorrowID | Numeric(12,0) | 借书顺序号【主键】 |
| 2 | rdID | Int | 读者序号【外键TB\_Reader】 |
| 3 | bkID | Int | 图书序号【外键TB\_Book】 |
| 4 | ldContinueTimes | Int | 续借次数（第一次借时，记为0） |
| 5 | ldDateOut | DateTime | 借书日期 |
| 6 | ldDateRetPlan | DateTime | 应还日期 |
| 7 | ldDateRetAct | DateTime | 实际还书日期 |
| 8 | ldOverDay | Int | 超期天数 |
| 9 | ldOverMoney | Money | 超期金额（应罚款金额） |
| 10 | ldPunishMoney | Money | 罚款金额 |
| 11 | lsHasReturn | Bit | 是否已经还书，缺省为0-未还 |
| 12 | OperatorLend | Nvarchar(20) | 借书操作员 |
| 13 | OperatorRet | Nvarchar(20) | 还书操作员 |

1. 借阅信息表(TB\_Borrow)

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名 | 数据类型 | 说明 |
| 1 | bkID | Int | 图书序号【标识列，主键】 |
| 2 | bkCode | Nvarchar (20) | 图书编号或条码号（前文中的书号） |
| 3 | bkName | Nvarchar(50) | 书名 |
| 4 | bkAuthor | Nvarchar(30) | 作者 |
| 5 | bkPress | Nvarchar(50) | 出版社 |
| 6 | bkDatePress | datetime | 出版日期 |
| 7 | bkISBN | Nvarchar (15) | ISBN书号 |
| 8 | bkCatalog | Nvarchar(30) | 分类号（如：TP316-21/123） |
| 9 | bkLanguage | SmallInt | 语言，0-中文，1-英文，2-日文，3-俄文，  4-德文，5-法文 |
| 10 | bkPages | Int | 页数 |
| 11 | bkPrice | Money | 价格 |
| 12 | bkDateIn | DateTime | 入馆日期 |
| 13 | bkBrief | Text | 内容简介 |
| 14 | bkCover | image | 图书封面照片 |
| 15 | bkStatus | NChar(2) | 图书状态，在馆、借出、遗失、变卖、销毁 |

五、详细设计与实现

1、客户端

1.1表示层(GUI)

使用Swing编写界面，(工具：WindowsBuilder)

**①登陆窗口**

设置用户名和密码两个输入框，登陆和退出两个按钮。

**\*核心代码：**

public Login() {

super("Library Manager System");

Image ime = Toolkit.getDefaultToolkit().getImage(

"/images/ico.png");

try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

} catch (ClassNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (InstantiationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IllegalAccessException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (UnsupportedLookAndFeelException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

setIconImage(ime);

initComponents();

setLocationRelativeTo(null);

this.setResizable(false);

}

private void jPasswordField1KeyTyped(java.awt.event.KeyEvent evt) {

if (evt.getKeyChar() == '\n') {

String name = jTextField1.getText(); // 获取用户名

String pass = String.valueOf(jPasswordField1.getPassword());// 获取密码

User user = null;

// 未输入用户名

if (name.equals("") || name == null) {

JOptionPane.showMessageDialog(this, "用户名不允许为空！", "cuowu", 0);

return;

}

try {

user = DAOFactory.getIUserDAOInstance().findById(name);

if (user != null) {

if (user.getPass() != null && user.getPass().equals(pass)) {

GlobalUser.LOGIN\_USER = user; // 记录当前用户

// 进入主界面

Main.start();

this.dispose();

} else {

JOptionPane.showMessageDialog(this, "用户名或密码错误!");

return;

}

}

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

if (user == null) {

JOptionPane.showMessageDialog(this, "用户名或密码错误!", "消息", 0);

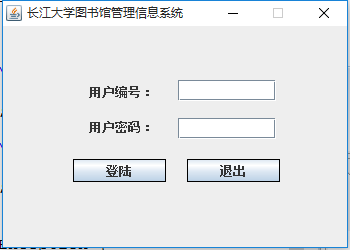
return;

}

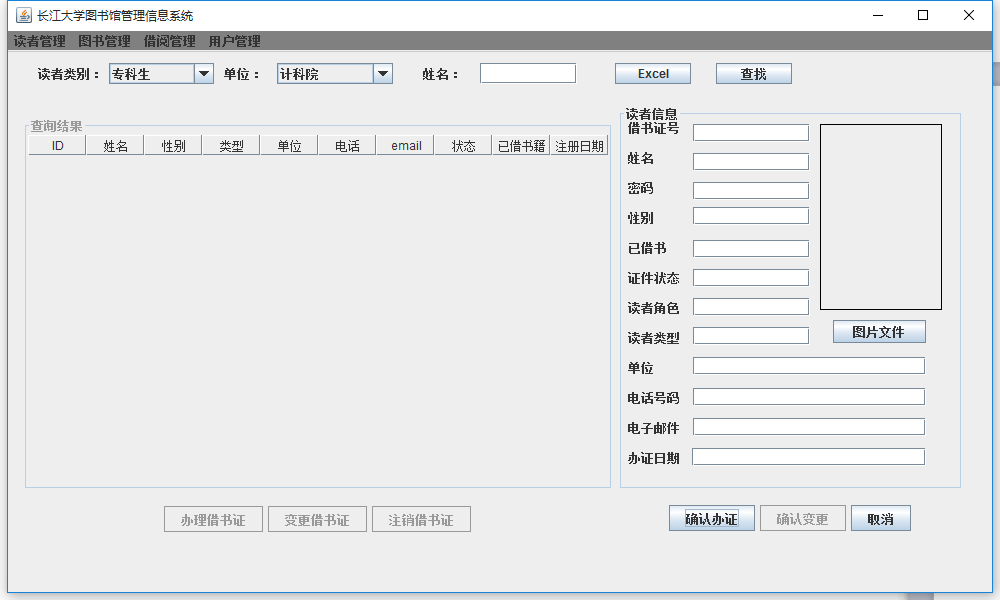
}

}

登陆界面图(Login.java)：



读者管理界面图(ReaderOp.fxml)：



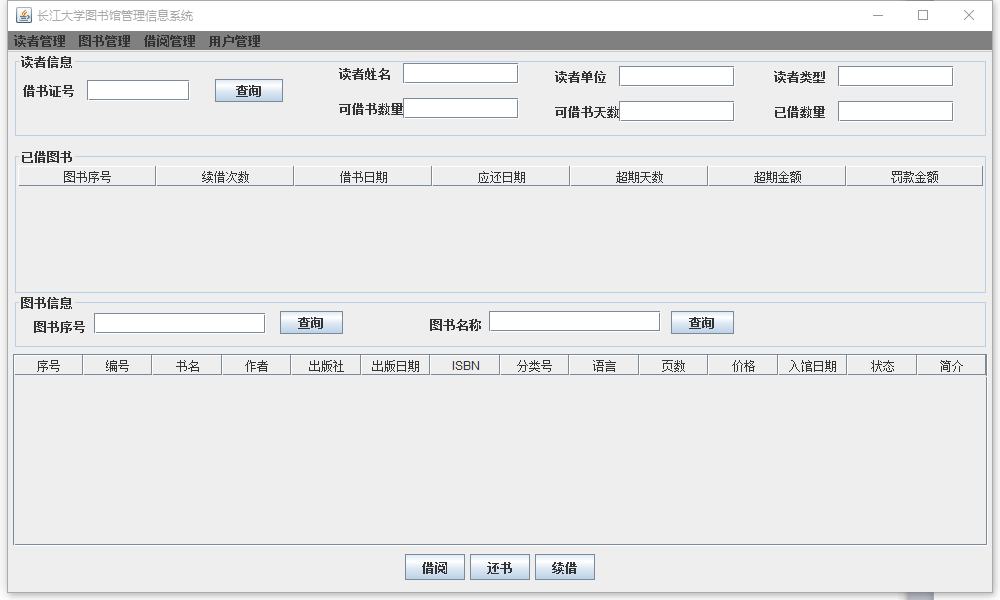
读者类别管理(ReaderTypeOp.fxml)：



图书信息维护图(BookSearch.fxml)：



借阅管理(BookIO.fxml)：



**1.2业务逻辑层**

**①主界面显示：**

（1）登陆成功后，根据登陆者身份判定其权限，并转入相应权限的界面。（采用4位二进制，借书证管理(0001)2=1、图书管理(0010)2=2、借阅管理(0100)2=4、系统管理(1000)2=8。如表示图书管理和借阅管理权限：2+4=6；判断7是否具备图书管理权限：7位与2，即(0111)2位与(0010)2=(0010)2，表示有此权限。）

**\*核心代码：**

private void initMenu() {

MN\_ReaderMgt.setEnabled(Login.reader.isReaderAdmin());

MN\_BookMgt.setEnabled(Login.reader.isBookAdmin());

MN\_BorrowMgt.setEnabled(Login.reader.isBorrowAdmin());

MN\_UserMgt.setEnabled(Login.reader.isSysAdmin());

}

**②读者管理：**

（1）通过选项栏和输入读者信息的方式进行模糊查找。

\*核心代码：

btnQuery.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent arg0) {

String rdType = (String) rdTypeComboBox.getSelectedItem();

String deptType = (String) deptTypeComboBox.getSelectedItem();

String userName = tfUserName.getText().trim();

Reader[] hits = readerBll.retrieveReadersString(rdType, deptType, userName);

// 更新查询结果列表

updateResultTable(hits);

}

});

private void updateResultTable(Reader[] readers) {

CustomizedTableModel<Reader> tableModel = (CustomizedTableModel<Reader>)

searchResultTable.getModel();

if (readers == null) {

JOptionPane.showMessageDialog(null, "没有找到符合要求的记录");

tableModel.setRecords(null);

tableModel.fireTableDataChanged();

return;

}

tableModel.setRecords(readers);

// 更新表格

tableModel.fireTableDataChanged();

}

（3）办理借书证。点击办理借书证后，“确认变更”的按钮变为灰色，“确认办理”的按钮浮起。在右侧输入信息后，点击“确认办理”，将信息输入到新创建的Reader类实例，并通过ReaderDAL类实例连接数据库，将信息输入到数据库中。

变更读者信息。点击需要变更的读者项，在右侧编辑信息，点击确认变更后，将数据库中的数据变更。

**\*核心代码：**

btnAddReader.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

Reader reader = getReaderFromText();

ReaderDAL dal= new ReaderDAL();

try {

dal.add(reader);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

String rdType = (String) rdTypeComboBox.getSelectedItem();

String deptType = (String) deptTypeComboBox.getSelectedItem();

String userName = tfUserName.getText().trim();

Reader[] hits = readerBll.retrieveReadersString(rdType, deptType, userName);

// 更新查询结果列表

updateResultTable(hits);

JOptionPane.showMessageDialog(null, "办证成功!");

}

});

btnSubmitUpdate.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// TODO Auto-generated method stub

Reader reader = getReaderFromText();

readerBll.updateReader(reader);

JOptionPane.showMessageDialog(null, "更改成功!");

String rdType = (String) rdTypeComboBox.getSelectedItem();

String deptType = (String) deptTypeComboBox.getSelectedItem();

String userName = tfUserName.getText().trim();

Reader[] hits = readerBll.retrieveReadersString(rdType, deptType, userName);

// 更新查询结果列表

updateResultTable(hits);

}

});

（4）挂失。点击后将读者项的状态变为挂失。

**\*核心代码：**

btnLost.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = searchResultTable.getSelectedRow();

if (selectedRow < 0) {

JOptionPane.showMessageDialog(null, "请先选中一个借书证!");

return;

} else {

Reader concle =

((CustomizedTableModel<Reader>) searchResultTable.getModel())

.getObjectAt(selectedRow);

concle.setRdStatus("挂失");

ReaderDAL dal = new ReaderDAL();

try {

dal.update(concle);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "挂失成功!");

String rdType = (String) rdTypeComboBox.getSelectedItem();

String deptType = (String) deptTypeComboBox.getSelectedItem();

String userName = tfUserName.getText().trim();

Reader[] hits = readerBll.retrieveReadersString(rdType, deptType, userName);

// 更新查询结果列表

updateResultTable(hits);

}

}

});

（5）解除挂失。与挂失相反的功能。

**\*核心代码：**

gbtnLost.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = searchResultTable.getSelectedRow();

if (selectedRow < 0) {

JOptionPane.showMessageDialog(null, "请先选中一个借书证!");

return;

} else {

Reader concle =

((CustomizedTableModel<Reader>) searchResultTable.getModel())

.getObjectAt(selectedRow);

concle.setRdStatus("有效");

ReaderDAL dal = new ReaderDAL();

try {

dal.update(concle);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "解除挂失成功!");

String rdType = (String) rdTypeComboBox.getSelectedItem();

String deptType = (String) deptTypeComboBox.getSelectedItem();

String userName = tfUserName.getText().trim();

Reader[] hits = readerBll.retrieveReadersString(rdType, deptType, userName);

// 更新查询结果列表

updateResultTable(hits);

}

}

});

（6）注销。注销时会检测该读者是否有未还书籍，若有则操作失败，否则执行操作。注销不可解除。

\*核心代码：

btnCancelReader.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = searchResultTable.getSelectedRow();

if (selectedRow < 0) {

JOptionPane.showMessageDialog(null, "请先选中一条记录!");

return;

} else {

Reader concle = ((CustomizedTableModel<Reader>) searchResultTable.getModel())

.getObjectAt(selectedRow);

ReaderDAL dal = new ReaderDAL();

try {

dal.delete(concle);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "注销成功!");

// 更新查询结果列表

updateResultTable();

}

}

③读者类别管理：

（1）添加条目。编辑读者类别信息，将信息传入新创建的ReaderType类的实例，再通过ReaderTypeDAL传入数据库。

\*核心代码：

btnAdd.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

ReaderType readerType =getReaderTypeFromText();

ReaderTypeDAL dal = new ReaderTypeDAL();

try {

int result = dal.add(readerType);

if (result < 0) {

JOptionPane.showMessageDialog(null, "添加失败! 请重新输入 !");

} else {

JOptionPane.showMessageDialog(null, "添加成功!");

}

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

//更新JTable

updateResultTable();

/\*\*根据输入框中的信息返回一个新建的读者类型对象

\* @return 读者类型对象

\*/

private ReaderType getReaderTypeFromText() {

ReaderType readerType = new ReaderType(Integer.valueOf(tfRdType.getText()));

readerType.setRdTypeName(tfRdTypeName.getText().trim());

readerType.setCanLendQty(Integer.valueOf(tfCanLendQty.getText().trim()));

readerType.setCanLendDay(Integer.valueOf(tfCanLendDay.getText().trim()));

readerType.setCanContinueTimes(Integer.valueOf(tfCanContinueTimes.getText().trim()));

readerType.setPunishRate(Float.valueOf(tfPunishRate.getText().trim()));

readerType.setDateValid(Integer.valueOf(tfDateValid.getText().trim()));

return readerType;

}

}

});

（2）删除条目。将选中的ReaderType实例装入ReaderTypeDAL，通过它来修改数据库的数据。

\*核心代码：

btnDelete.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

int selectedRow = searchResultTable.getSelectedRow();

if (selectedRow < 0) {

JOptionPane.showMessageDialog(null, "请先选中一条记录!");

return;

} else {

ReaderType concle = ((CustomizedTableModel<ReaderType>) searchResultTable.getModel())

.getObjectAt(selectedRow);

ReaderTypeDAL dal = new ReaderTypeDAL();

try {

dal.delete(concle);

updateResultTable();

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "删除成功!");

}

}

});

（3）修改条目。将编辑后的读者类别信息在对应的ReaderType实例中更新，再将该ReaderType实例装入ReaderTypeDAL，通过它来修改数据库的数据。

\*核心代码：

btnUpdate.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = searchResultTable.getRowCount();

if (selectedRow >= 0) {

ReaderType readerType = getReaderTypeFromText();

readerTypeBll.updateReaderType(readerType);

JOptionPane.showMessageDialog(null, "更改成功!");

updateResultTable();

}else {

JOptionPane.showMessageDialog(null, "更改失败!");

}

}

;④新书入库：

当管理员直接点击“新书入库”时，修改键为灰色，添加键浮起。

当管理员在“图书信息维护”处点击修改时，弹出的“新书入库”界面中，添加建为灰色，修改键浮起。

（1）添加图书。

\*核心代码：

btnAddBook.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (tfBkCode.getText().trim().equals("")) {

JOptionPane.showMessageDialog(null, "请输入图书编号");

} else {

Book book = getBookFromText();

BookDAL dal = new BookDAL();

try {

dal.add(book);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "新增成功!");

}

}

});

（2）修改图书。

\*核心代码：

btnSubmitUpdate.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

Book Book = getBookFromText();

bookBll.updateBook(Book);

JOptionPane.showMessageDialog(null, "更改成功!");

updateResultTable();

}

});

（3）取消。将输入清空或置为初始值。

\*核心代码：

button.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

setBookToTextNull();

}

（4）选择封面。

\*核心代码：

btnLoadPictureFile.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// TODO Auto-generated method stub

JFileChooser fc = new JFileChooser();

fc.addChoosableFileFilter(new ImageFilter());

// 实现文件选取功能并添加选择图片文件类型的过滤器ImageFilter

int returnVal = fc.showOpenDialog(SearchBookPanel.this);

if (returnVal == JFileChooser.APPROVE\_OPTION) {

File file = fc.getSelectedFile();

try {

BufferedImage img = ImageIO.read(file);

Image dimg = img.getScaledInstance(lblBkCover.getWidth(), lblBkCover.getHeight(),

Image.SCALE\_SMOOTH);

ImageIcon icon = new ImageIcon(dimg);

lblBkCover.setIcon(icon);

} catch (IOException ea) {

ea.printStackTrace();

}

}

}

});

1. 修改书籍信息。

\*核心代码：

btnSubmitUpdate.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

Book Book = getBookFromText();

bookBll.updateBook(Book);

JOptionPane.showMessageDialog(null, "更改成功!");

updateResultTable();

}

});

（4）删除书籍。

\*核心代码：

btnDeteleBook.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = searchResultTable.getSelectedRow();

if (selectedRow < 0) {

JOptionPane.showMessageDialog(null, "请先选中一条记录!");

return;

} else {

Book concle = ((CustomizedTableModel<Book>) searchResultTable.getModel()).getObjectAt(selectedRow);

BookDAL dal = new BookDAL();

try {

dal.delete(concle);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "删除成功!");

updateResultTable();

}

}

});

⑥借阅管理：

（1）读者信息查找。与前面读者信息查询相同，但是，当以读者身份（非管理员）登录时，读者信息为自己的信息，不可查询。代码略。

（2）图书信息查找。与前面的查找方式类似。代码略。

（3）借阅图书。当读者状态“有效”、借书数未达到最大且图书状态为“在馆”时可借阅。

\*核心代码：

btnBorrow.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = bookSearchResultTable.getSelectedRow();

if(tfRdID.getText().length() == 0){

JOptionPane.showMessageDialog(null, "请先查询借书证!");

}else if(selectedRow <0 ){

JOptionPane.showMessageDialog(null, "请先选择一本书籍!");

}else {

Book book = (Book) ((CustomizedTableModel<Book>) bookSearchResultTable.getModel()).getObjectAt(selectedRow);

int bkID = book.getBkID();

int rdID = Integer.valueOf(tfRdID.getText().trim());

String borrowID = String.valueOf(rdID)+String.valueOf(bkID);

Calendar cal = Calendar.getInstance();

SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");

String ldDateOut = format.format(cal.getTime());

int day = Integer.valueOf(CanLendDay);

cal.add(Calendar.FRIDAY, day);

String ldDateRetPlan = format.format(cal.getTime());

Borrow borrow = new Borrow();

borrow.setBorrowID(borrowID);

borrow.setRdID(rdID);

borrow.setBkID(bkID);

borrow.setLdContinueTimes(0);

borrow.setLdDateOut(ldDateOut);

borrow.setLdDateRetPlan(ldDateRetPlan);

BorrowDAL dal = new BorrowDAL();

try {

dal.add(borrow);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

JOptionPane.showMessageDialog(null, "借阅成功!");

}

}

});

（4）续借图书。当读者状态“有效”且续借数未达到最大可续借。

代码与借阅功能代码类似，略。

（5）还书。使用BackBook类进行还书操作。如图：

\*核心代码：

btnReturn.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int selectedRow = borrowSearchResultTable.getSelectedRow();

if(selectedRow <0 ){

JOptionPane.showMessageDialog(null, "请先选择一个记录!");

return;

}else{

Borrow concle = ((CustomizedTableModel<Borrow>) borrowSearchResultTable.getModel())

.getObjectAt(selectedRow);

BorrowDAL dal = new BorrowDAL();

try {

dal.delete(concle);

} catch (Exception e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

}

String BkID = tfBkID.getText().trim();

Book[] hits = bookBll.retrieveBooksStringByID(BkID);

updateBookResultTable(hits);

JOptionPane.showMessageDialog(null, "还书成功!");

}

});

1.3数据访问层

①读者数据库访问类（ReaderDAL） 代码略

②读者类别数据库访问类（ReaderTypeDAL）代码略

③借阅信息数据库访问类（BorrowDAL）代码略

④图书信息数据库访问类（BookDAL）代码略

1.4实体类

实体类包含读者实体类、读者类别实体类、借阅实体类和图书实体类。（内部属性与数据库表字段基本相同，代码略）

2、数据库

2.1使用MS SQL Server 2008 R2，用户名为sa，密码123

2.2建立Library数据库，架构为dbo

2.3建立表：

①读者表（TB\_Reader）：

CREATE TABLE [dbo].[TB\_Reader](

[rdID] [int] NOT NULL,

[rdName] [nvarchar](20) NOT NULL,

[rdSex] [nchar](1) NOT NULL,

[rdType] [smallint] NOT NULL,

[rdDept] [nvarchar](20) NOT NULL,

[rdPhone] [nvarchar](25) NOT NULL,

[rdEmail] [nvarchar](25) NOT NULL,

[rdDateReg] [date] NOT NULL,

[rdPhoto] [image] NULL,

[rdStatus] [nchar](2) NOT NULL,

[rdBorrowQty] [int] NOT NULL,

[rdPwd] [nvarchar](20) NOT NULL,

[rdAdminRoles] [smallint] NULL,

CONSTRAINT [PK\_tb\_users] PRIMARY KEY CLUSTERED

(

[rdID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

②读者类别表（TB\_ReaderType）：

CREATE TABLE [dbo].[TB\_ReaderType](

[rdType] smallint NOT NULL,

[rdTypeName] nvarchar(20) NOT NULL,

[CanLendQty] int NULL,

[CanLendDay] int NULL,

[CanContinueTimes] int NULL,

[PunishRate] float NULL,

[DateValid] smallint NULL DEFAULT ((0)),

PRIMARY KEY ([rdType]),

UNIQUE ([rdTypeName])

)

GO

insert into [TB\_ReaderType] values(10,'教师',12,60,2,0.05,0);

insert into [TB\_ReaderType] values(20,'本科生',8,30,1,0.05,4);

insert into [TB\_ReaderType] values(21,'专科生',8,30,1,0.05,3);

insert into [TB\_ReaderType] values(30,'硕士研究生',8,30,1,0.05,3);

insert into [TB\_ReaderType] values(31,'博士研究生',8,30,1,0.05,4);

GO

③借阅信息表（TB\_Borrow）：

CREATE TABLE [dbo].[TB\_Borrow](

[BorrowID] [numeric](12, 0) IDENTITY(100000000000,1) NOT NULL,

[rdID] [int] NOT NULL,

[bkID] [int] NOT NULL,

[IdContinueTimes] [int] NOT NULL,

[IdDateOut] [date] NOT NULL,

[IdDateRetPlan] [date] NULL,

[IdDateRetAct] [date] NULL,

[IdOverDay] [int] NULL,

[IdOverMoney] [money] NULL,

[IdPunishMoney] [money] NULL,

[IsHasReturn] [bit] NOT NULL,

[OperatorLend] [nvarchar](20) NOT NULL,

[OperatorRet] [nvarchar](20) NULL,

CONSTRAINT [PK\_TB\_Borrow\_1] PRIMARY KEY CLUSTERED

(

[BorrowID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

④图书信息表（TB\_Book）：

CREATE TABLE [dbo].[TB\_Book](

[bkID] [int] NOT NULL,

[bkCode] [nvarchar](20) NOT NULL,

[bkName] [nvarchar](50) NOT NULL,

[bkAuthor] [nvarchar](30) NOT NULL,

[bkPress] [nvarchar](50) NOT NULL,

[bkDatePress] [date] NULL,

[bkISBN] [nvarchar](15) NULL,

[bkCatalog] [nvarchar](30) NULL,

[bkLanguage] [smallint] NULL,

[bkPages] [int] NULL,

[bkPrice] [money] NULL,

[bkDateIn] [date] NULL,

[bkBrief] [text] NULL,

[bkCover] [image] NULL,

[bkStatus] [nchar](2) NULL,

CONSTRAINT [PK\_TB\_Book] PRIMARY KEY CLUSTERED

(

[bkID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

2.4 为TB\_Borrow表创建触发器check\_punish，判断是否超期，用于更新借阅表。具体代码如下：

ALTER TRIGGER [dbo].[check\_punish] ON [Library].[dbo].[TB\_Borrow]

FOR UPDATE

AS

BEGIN

DECLARE YB CURSOR

FOR SELECT IdDateRetPlan,rdID,BorrowID

FROM inserted

OPEN YB

DECLARE @BookRetPlan DATE

DECLARE @ReaderID INT

DECLARE @PunishRate FLOAT

DECLARE @BorrowID NUMERIC(12, 0)

DECLARE @Overday INT

FETCH NEXT FROM YB INTO @BookRetPlan,@ReaderID,@BorrowID

WHILE @@FETCH\_STATUS = 0

BEGIN

SELECT @PunishRate = PunishRate

FROM Library.dbo.TB\_Reader AS reader,Library.dbo.TB\_ReaderType AS readerType

WHERE reader.rdType = readerType.rdType AND reader.rdID = @ReaderID

IF(@BookRetPlan< CONVERT(nchar(10),GETDATE(),23))

BEGIN

SET@Overday= DATEDIFF(DAY,@BookRetPlan,CONVERT(nchar(10),GETDATE(),23))

UPDATE Library.dbo.TB\_Borrow

SET IdOverDay = @Overday,IdOverMoney = @Overday \* @PunishRate

WHERE BorrowID = @BorrowID

END

FETCH YB INTO @BookRetPlan,@ReaderID,@BorrowID

END

CLOSE YB

DEALLOCATE YB

END

六、课程设计小结

1、设计总结

在我做课程设计时，将书本上所学的知识尽可能地发挥到了实践中。经过需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施及数据库运营与维护6大阶段，我制作并运营成功了一个的以SQLSERVER 2008数据库为后台、在Eclipse中使用JavaFX所制作的应用程序为前台的图书馆管理系统软件。

通过这次设计，我受益非浅，亲身体验了数据库设计的全过程和基于C/S的三层架构模式，在实践中了解了数据库系统设计的步骤、流程以及思路，增长了在数据库设计方面的见识，我深刻认识到以前所学的基础课程的重要性，也使我们掌握了很多新知识，特别是一些课本之外的知识，如JavaFx技术，体会到了理论知识和实践相结合的重要性。

2、参考资料

《数据库课程设计》任务与指导手册（V2.0 Java）

www.csdn.com

www.stackoverflow.com