



**专项设计报告**

**基本Web的图书管理系统**

**学 院 计算机学院**

**专 业 网络工程**

**年级班别 1604**

**学 号 3216005086**

**学生姓名 张微**

**指导教师 张凡龙**

**2019年 10月**

目录

[1 需求分析 4](#_Toc21462)

[1.1需求定义 4](#_Toc13151)

[1.2 功能需求 5](#_Toc5948)

[2 概念结构设计 6](#_Toc4778)

[2.1读者信息模块 6](#_Toc24577)

[2.2 图书信息模块 6](#_Toc23308)

[2.3 管理员信息模块 6](#_Toc6887)

[2.4 完整的E-R图 6](#_Toc31181)

[3 逻辑结构设计 7](#_Toc11621)

[3.1数据库逻辑设计 7](#_Toc24238)

[3.2数据库逻辑结构 7](#_Toc28625)

[4 物理设计 9](#_Toc29149)

[4.1 建立图书馆数据库（library） 9](#_Toc7692)

[4.2 建立图书表（book） 9](#_Toc19068)

[4.3 建立读者表（reader） 9](#_Toc6758)

[4.4 建立借阅表（borrow） 10](#_Toc12903)

[4.5 建立管理员表（administartors） 10](#_Toc31958)

[4.6 建立还书信息表（return\_table） 11](#_Toc10526)

[5 实施过程 12](#_Toc30907)

[5.1装载数据过程 12](#_Toc26359)

[5.2设计视图 13](#_Toc13750)

[5.3 存储过程设计 14](#_Toc11025)

[5.4触发器设计 17](#_Toc23658)

**1 需求分析**

本图书管理系统主要实现对图书馆的管理：图书、读者、管理员、借阅。

由此，结构可分为：图书管理、读者管理、管理员管理、借还管理、罚单管理、还书信息。

**1.1需求定义**

1.1.1图书管理

1).浏览图书信息

2).编辑图书信息：修改、入库、删除

1.1.2 读者管理

1).浏览读者信息

2).编辑读者信息：修改、添加、删除

1.1.3管理员管理

1).浏览管理员信息

2).编辑管理员信息：修改、添加、删除

1.1.4借还管理

1).浏览图书借书信息

2).借书（每次只能操作一次）-> 可对新增借书的删除

4).还书（逾期欠费需缴费才能还书）

1.1.5罚单管理

1).浏览罚单信息

2).缴费

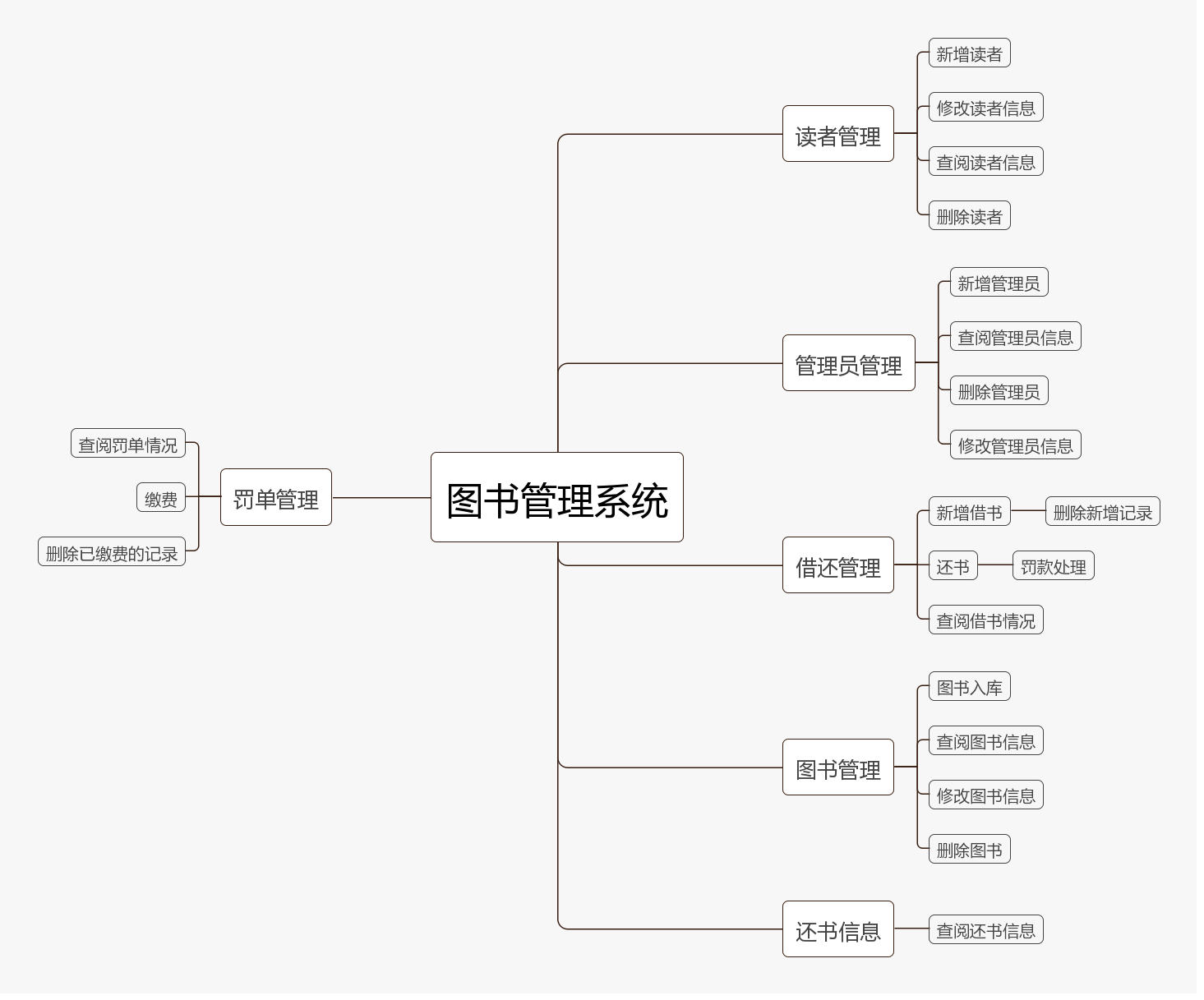
3).删除已缴费的记录

1.1.6还书信息

1. .浏览还书信息

**1.2 功能需求**

1. 图书的查阅、修改、删除、图书入库：编号、书名、作者、出版社、在库数、类别
2. 读者的查询、修改、删除、新增读者：借书证号、姓名、班级、院系
3. 管理员的查询、修改、删除、新增管理员：编号、姓名、负责书库
4. 借还信息的查询、还书（逾期欠费需缴费才能还书）、新增借书：借书证号、读者名、编号、书名、借书日期、预计归还日期，可删除新增记录
5. 罚单的查阅、缴费、删除（已缴费）
6. 还书信息的查阅

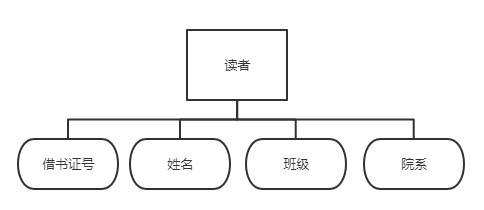


1.1图书管理系统功能图

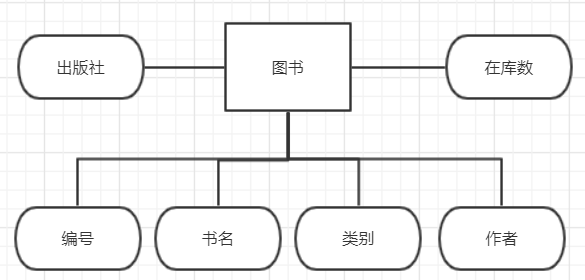
**2 概念结构设计**

根据需求分析我们规划出实体有：读者信息实体、管理员信息实体、图书信息实体。各个实体具体的描述E-R实体如下。

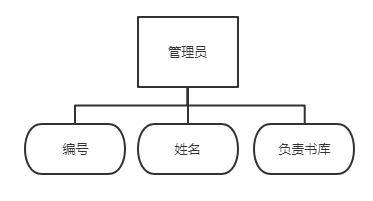
**2.1读者信息模块**



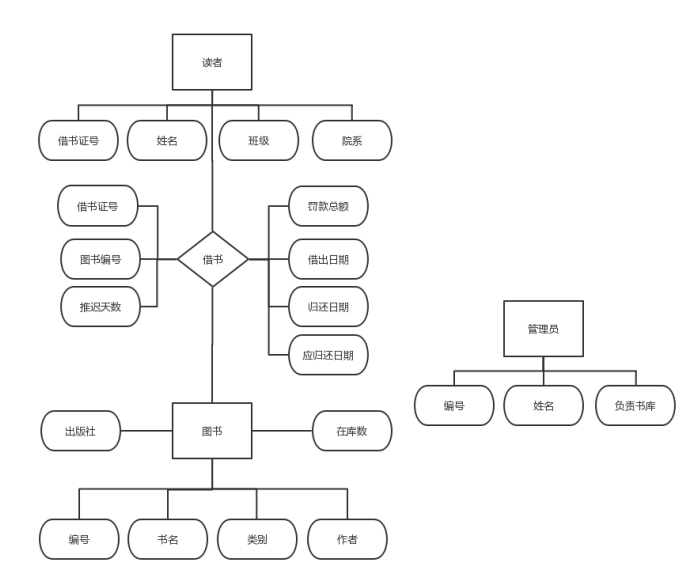
**2.2 图书信息模块**



**2.3 管理员信息模块**



**2.4 完整的E-R图**



**3 逻辑结构设计**

**3.1数据库逻辑设计**

由E-R模型得到关系模型：

3.1.1 借阅关系

属性：借书证号、图书编号、借出日期、应归还日期、归还日期、推迟天数、罚款总额

主键：借书证号、图书编号

**3.2数据库逻辑结构**

3.2.1 图书信息表



3.2.2 读者信息表



3.2.3 借书信息表



3.2.4 还书信息表



3.2.5管理员信息表



3.2.6 罚单信息表



**4 物理设计**

由上述数据库逻辑结构转换为实际数据模型，本图书管理系统的各个表如下：

**4.1 建立图书馆数据库（library）**

**CREATE** **DATABASE** library;

**4.2 建立图书表（book）**

**CREATE** **TABLE** `book` (

`id` **INT**(11) **NOT** **NULL** **DEFAULT** '0',

`name` **VARCHAR**(20) **NOT** **NULL**,

`author` **VARCHAR**(20) **NOT** **NULL**,

`publisher` **VARCHAR**(20) **NOT** **NULL**,

`num` **INT**(11) **NOT** **NULL**,

`sort` **VARCHAR**(20) **NOT** **NULL**,

**PRIMARY** **KEY** (`id`)

)

**COLLATE**='utf8\_general\_ci'

**ENGINE**=**InnoDB**

;

desc book;



**4.3 建立读者表（reader）**

**CREATE** **TABLE** `reader` (

`id` **INT**(11) **NOT** **NULL** **DEFAULT** '0',

`name` **VARCHAR**(20) **NOT** **NULL**,

`class` **VARCHAR**(10) **NOT** **NULL**,

`department` **VARCHAR**(10) **NOT** **NULL**,

**PRIMARY** **KEY** (`id`)

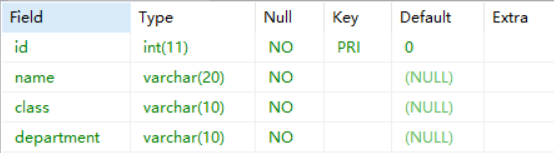
)

**COLLATE**='utf8\_general\_ci'

**ENGINE**=**InnoDB**

;

**desc** reader;



**4.4 建立借阅表（borrow）**

**CREATE** **TABLE** `borrow` (

`bookId` **INT**(11) **NOT** **NULL** **COMMENT** '学生编号',

`readerId` **INT**(11) **NOT** **NULL** **COMMENT** '图书编号',

`borrow\_date` **DATE** **NULL** **DEFAULT** **NULL** **COMMENT** '借书时间',

`expect\_return\_date` **DATE** **NULL** **DEFAULT** **NULL** **COMMENT** '预期归还时间',

**PRIMARY** **KEY** (`bookId`, `readerId`),

**INDEX** `READERID` (`readerId`, `bookId`),

**CONSTRAINT** `BOOKID` **FOREIGN** **KEY** (`bookId`) **REFERENCES** `book` (`id`),

**CONSTRAINT** `READERID` **FOREIGN** **KEY** (`readerId`) **REFERENCES** `reader` (`id`)

)

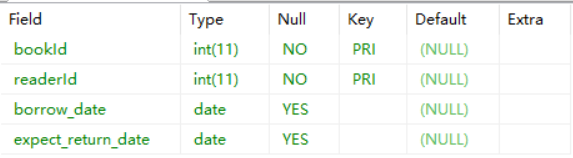
**COMMENT**='存储学生的借书信息'

**COLLATE**='latin1\_swedish\_ci'

**ENGINE**=**InnoDB**

;

**desc** borrow;



**4.5 建立管理员表（administartors）**

**CREATE** **TABLE** `administrators` (

`id` **INT**(11) **NOT** **NULL** **DEFAULT** '0',

`name` **VARCHAR**(20) **NOT** **NULL**,

`stackroom` **VARCHAR**(10) **NOT** **NULL**,

**PRIMARY** **KEY** (`id`)

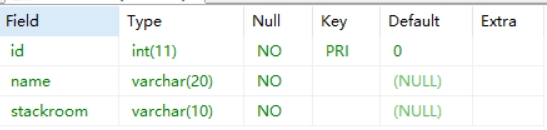
)

**COLLATE**='utf8\_general\_ci'

**ENGINE**=**InnoDB**

;

**desc** administrators;



**4.6 建立还书信息表（return\_table）**

**CREATE** **TABLE** `return\_table` (

`bookId` **INT**(11) **NOT** **NULL**,

`readerId` **INT**(11) **NOT** **NULL**,

`borrow\_date` **DATE** **NULL** **DEFAULT** **NULL**,

`return\_date` **DATE** **NULL** **DEFAULT** **NULL** **COMMENT** '实际还书情况',

**PRIMARY** **KEY** (`bookId`, `readerId`),

**INDEX** `readerId` (`readerId`, `bookId`),

**CONSTRAINT** `BOOKID\_R` **FOREIGN** **KEY** (`bookId`) **REFERENCES** `book` (`id`),

**CONSTRAINT** `READERID\_R` **FOREIGN** **KEY** (`readerId`) **REFERENCES** `reader` (`id`)

)

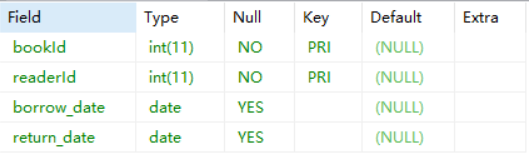
**COMMENT**='存储学生的归还信息'

**COLLATE**='latin1\_swedish\_ci'

**ENGINE**=**InnoDB**

;

**desc** return\_table;



**4.7建立罚单信息表（ticket）**

**CREATE** **TABLE** `ticket` (

`readerId` **INT**(11) **NOT** **NULL** **COMMENT** '学生编号',

`bookId` **INT**(11) **NOT** **NULL** **COMMENT** '书籍编号',

`over\_date` **INT**(11) **NULL** **DEFAULT** **NULL** **COMMENT** '超期天数',

`ticket\_fee` **FLOAT** **NULL** **DEFAULT** **NULL** **COMMENT** '处罚金额',

`payoff` **INT**(11) **NULL** **DEFAULT** **NULL**,

**PRIMARY** **KEY** (`readerId`, `bookId`),

**INDEX** `BOOKID\_T` (`bookId`),

**CONSTRAINT** `BOOKID\_T` **FOREIGN** **KEY** (`bookId`) **REFERENCES** `book` (`id`),

**CONSTRAINT** `READERID\_T` **FOREIGN** **KEY** (`readerId`) **REFERENCES** `reader` (`id`)

)

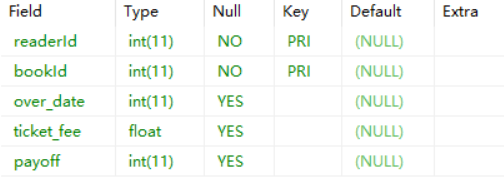
**COMMENT**='存储学生的罚单信息'

**COLLATE**='latin1\_swedish\_ci'

**ENGINE**=**InnoDB**

;

**desc** ticket;



**5 实施过程**

**5.1装载数据过程**

5.1.1 用DML语句对book表加入数据

**INSERT** **INTO** `library`.`book` (`id`, `name`, `author`, `publisher`, `num`, `sort`) **VALUES** ('8', '时事报告', '张习文', '时事报告杂志社', '1', '政治');

**INSERT** **INTO** `library`.`book` (`id`, `name`, `author`, `publisher`, `num`, `sort`) **VALUES** ('9', '计算机网络安全', '马利', '清华大学出版社', '1', '计算机');

**select** \* **from** book;



5.1.2 用DML语句对reader表加入数据

**INSERT** **INTO** `library`.`reader` (`id`, `name`, `class`, `department`) **VALUES** ('6', '张三', '1603', '机器人学院');

**INSERT** **INTO** `library`.`reader` (`id`, `name`, `class`, `department`) **VALUES** ('7', '李四', '1602', '金融学院');

**select** \* **from** reader;



5.1.3 用DML语句对administartors表加入数据

**INSERT** **INTO** `library`.`administrators` (`id`, `name`, `stackroom`) **VALUES** ('4', '王五', '外国文学');

**INSERT** **INTO** `library`.`administrators` (`id`, `name`, `stackroom`) **VALUES** ('5', '江六', '中国文学');

**select** \* **from** administrators;



**5.2设计视图**

5.2.1 在reader,borrow和book上创建借书者的全面信息视图reader\_borrow

**create** **view** reader\_borrow **as**

**select** reader.id **AS** reader\_id,reader.name **AS** reader\_name,book.id **AS** book\_id,book.name **AS** book\_name,borrow\_date,**addDate**(borrow\_date,30) expect\_return\_date

**from** reader,book,borrow

**where** reader.id = borrow.readerId **and** book.id = borrow.bookId;



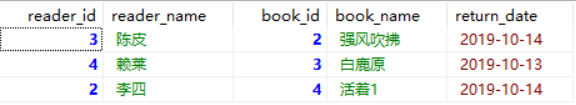
5.2.3 创建个人所有借书归还记录视图 reader\_borrow\_return

**create** **view** reader\_borrow\_return **as**

**select** reader.id **AS** reader\_id,reader.name **AS** reader\_name,book.id **AS** book\_id,book.name **AS** book\_name,return\_date

**from** reader,book,return\_table

**where** reader.id = return\_table.readerId **and** book.id = return\_table.bookId;



**5.3 存储过程设计**

5.3.1 产生罚单proc\_gen\_ticket

当日期超过预定归还日期，产生罚单，并将记录写入表ticket中，这个存储过程在定时器eventJob中调用

**CREATE** **DEFINER**=`root`@`%` **PROCEDURE** `proc\_gen\_ticket`(

**IN** `currentdate` **date**

)

**BEGIN**

**replace** **into** ticket(readerId,bookId,over\_date,ticket\_fee,payoff)

**SELECT** reader\_id,book\_id,**dateDiff**(currentdate,reader\_borrow.expect\_return\_date),0.1\***datediff**(currentdate,reader\_borrow.expect\_return\_date),1

**FROM** reader\_borrow

**WHERE** currentdate>reader\_borrow.expect\_return\_date;

**END**

5.3.2 借书过程实现

1) 设计存储函数，func\_get\_booknum，返回书籍是否在架上

**CREATE** **DEFINER**=`root`@`%` **FUNCTION** `func\_get\_booknum`(

`book\_id` **INT**

)

**RETURNS** **int**(11)

**BEGIN**

**return**(**select** num **from** book **where** book.id=book\_id);

**END**

2) 设计存储过程proc\_borrow，调用func\_get\_booknum，判断书籍是否在架上，若为真，则借书成功，在borrow表中插入记录；否则提示失败

**CREATE** **DEFINER**=`root`@`%` **PROCEDURE** `proc\_borrow`(

**IN** `reader\_id` **INT**,

**IN** `book\_id` **INT**,

**IN** `borrow\_date` **DATE**,

**IN** `expect\_return\_date` **DATE**

)

**BEGIN**

**if** func\_get\_booknum(book\_id) > 0 **then**

**insert** **into** borrow(readerId,bookId,borrow\_date,expect\_return\_date)

**values**(reader\_id,book\_id,borrow\_date,expect\_return\_date);

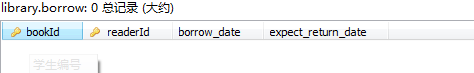
**else**

**select** -1 **as** msg;

**end** **if**;

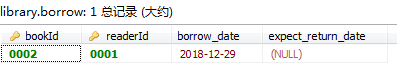
**END**

目前borrow记录如下

****

执行函数，读者0001借图书0002

**call** proc\_borrow('0001','0002',**CURDATE**());

****

****

执行函数，读者0002借图书0002

**call** proc\_borrow('0002','0002',**CURDATE**());

****

5.3.3 还书存储过程proc\_return

还书时，查看是否超期，即查询ticket表项，当发现超期，提示交罚单后再还书；如未超期，则记录归还项目到return\_table中，并且删除借书记录（以免还书后定时器仍然扫描这个记录）

**CREATE** **DEFINER**=`root`@`%` **PROCEDURE** `proc\_return`(

**IN** `reader\_id` **INT**(11),

**IN** `book\_id` **INT**(11),

**IN** `return\_date` **DATE**

)

**BEGIN**

**declare** borrowdate **date**;

**if**(**select** payoff **from** ticket **where** ticket.readerId = reader\_id **and** ticket.bookId = book\_id) = 1 **then**

**select** -1 **as** msg;

**else**

**if**(**select** **count**(\*) **from** return\_table **where** return\_table.readerId = reader\_id **and** return\_table.bookId = book\_id ) > 0 **then**

**DELETE** **FROM** return\_table **WHERE** readerId = reader\_id;

**set** borrowdate = (**select** borrow\_date **from** borrow **where** borrow.readerId = reader\_id **and** borrow.bookId = book\_id);

**insert** **into** return\_table(readerId,bookId,borrow\_date,return\_date)

**values**(reader\_id,book\_id,borrowdate,return\_date);

**delete** **from** borrow

**where** borrow.readerId = reader\_id **and** borrow.bookId = book\_id;

**else**

**set** borrowdate = (**select** borrow\_date **from** borrow **where** borrow.readerId = reader\_id **and** borrow.bookId = book\_id);

**insert** **into** return\_table(readerId,bookId,borrow\_date,return\_date)

**values**(reader\_id,book\_id,borrowdate,return\_date);

**delete** **from** borrow

**where** borrow.readerId = reader\_id **and** borrow.bookId = book\_id;

**end** **if**;

**end** **if**;

**END**

读者0001借了0001图书，超期产生了罚单

****

**call** proc\_return('0001','0001',**CURDATE**());

****

交罚单

**call** proc\_payoff('0001','0001');

****

****

再次还书

****

5.3.4 交罚单存储过程proc\_payoff

修改罚单中payoff段为0，表明罚单已交

**CREATE** **DEFINER**=`root`@`%` **PROCEDURE** `proc\_payoff`(

**IN** `reader\_id` **INT**,

**IN** `book\_id` **INT**

)

**BEGIN**

**update** ticket

**set** ticket.payoff = 0

**where** ticket.readerId = reader\_id **and** ticket.bookId = book\_id;

**select** 'succeed';

**END**

5.3.5 删除图书存储过程proc\_deleteBook

删除图书时，除了 book表，还要把关联表 return\_table 对应的书也删掉

**CREATE** **DEFINER**=`root`@`%` **PROCEDURE** `proc\_deleteBook`(

**IN** `book\_id` **INT**

)

**BEGIN**

**DELETE** **FROM** return\_table **WHERE** return\_table.bookId = book\_id;

**DELETE** **FROM** book **WHERE** book.id = book\_id;

**END**

**5.4触发器设计**

5.4.1 设计触发器borrow\_after\_insert

当某学生借书成功后，图书表相应的图书不在架上，变为0；

**CREATE** **TRIGGER** `borrow\_after\_insert` **AFTER** **INSERT** **ON** `borrow` **FOR** **EACH** **ROW** **BEGIN**

**update** book **set** num = num-1 **where** id = **new**.bookId;

**END**

在插入表borrow之前，id=0003的图书还在架上，为1



学生0001借了这本书后，在boorow中插入记录



在borrow中插入这条记录后，id=0003的图书，不在架上，为0

****

5.4.2 设计触发器return\_after\_insert

还书成功后，对应的书籍num变为1

**CREATE** **TRIGGER** `return\_after\_insert` **AFTER** **INSERT** **ON** `return` **FOR** **EACH** **ROW** **BEGIN**

**update** book **set** num = num+1 **where** id = **new**.bookId;

**END**

还书时在return\_table插入

****

此时图书归还架上

****

5.4.3 定义定时器（事件）eventJob

每天自动触发一次，扫描视图reader\_borrow，若发现当前有预期归还时间小于当前时间，则判断为超期，生成处罚记录，这个定时器将每天定时触发存储过程proc\_gen\_ticket

**CREATE** **EVENT** `eventJob`

**ON** **SCHEDULE**

**EVERY** 1 **DAY** **STARTS** '2018-08-01 16:22:42'

**ON** **COMPLETION** **PRESERVE**

**ENABLE**

**COMMENT** ''

**DO** **call** proc\_gen\_ticket(**CURDATE**())

读者0003借了图书0001，生成借书记录reader\_borrow视图



当他在1月9日前还书，没有生成罚单



当他在1月9日后还书，生成罚单



