**书店租赁管理系统**

1. **需求分析**

## 1.系统总体的功能需求

系统在界面设计方面要尽可能的人性化，对用户使用而言应该是简单易用的，在布局和设计上要科学化。

就目前而言,该系统是为中小型书店研发的。系统开发的目标是实现书店图书租赁的系统化、规范化和自动化，这是在用户要求的基础上提出来的，功能要求如下：

1、管理员能对书店租赁系统里的会员信息、图书信息、借阅信息、收入信息等进行数据的添加、修改、删除、查询以及统计的功能操作。

2、会员能够登录系统和修改密码，并且只能够查询自己的借阅信息和个人信息查询以及查询书店书库信息（包括书名、作者、出版社、库存数量等）。

## 2.系统结构设计

可以将书店管理系统分为两个模块——用户模块和管理员模块.所谓的用户模块也就是前台服务，管理员模块也就是后台服务。

会员信息管理

图书信息管理

收入信息查询

系统管理

书店租赁管理**系**统

新增会员

信息修改

删除会员

会员信息查询

图书信息查询

新增图书

删除图书

库存量

借书管理

还书管理

借阅天数

登录管理

退出

收入信息

信息查询

借书时间

图书价格

借书会员

借阅图书

### 3.主要功能模块

<1>图书信息维护模块

该模块用于维护图书信息。当有新书时可以将新书信息添加到数据库中，也可以对图书信息进行修改，当有旧书要淘汰时可以将旧书信息从数据库中删除。

<2>借阅管理模块

借阅管理模块是本系统重要的组成部分，这一模块可以包含了借书和还书两个模块。其中借书模块用于实现用户借书操作，借书操作完成后该图书的库存数和最大可借数自动减一。还书模块用于实现用户还书操作，还书操作完成后该图书的库存数和最大可借数自动加一。

<3>图书信息查询模块

图书信息查询模块用于查询所有图书信息。当读者想要了解图书信息时可以进入此模块进行查询。

1. **概念结构设计**

E-R模型图：

用户性别

最大可借数

用户名称

用户类型

可借天数

用户编号

用户

M

借阅时间

借阅

归还时间

图书编号

图书名称

图书类型

作者

译者

出版社

价格

出版时间

库存数量

图书

N

**三、逻辑结构设计**

**图书信息表**：

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 说明 | 类型（长度） | 备注 |
| Book\_id | 图书编号 | Varchar（32） | 不允许空，主键 |
| name | 图书名称 | Varchar（100） | 允许空 |
| type | 图书类型 | Varchar（50） | 允许空 |
| author | 作者 | Varchar（50） | 允许空 |
| translator | 译者 | Varchar（50） | 允许空 |
| publisher | 出版社 | Varchar（1024） | 允许空 |
| Publish\_time | 出版时间 | date | 允许空 |
| stock | 库存数量 | int(11) | 允许空 |
| price | 价格 | double | 允许空 |

**用户信息表：**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 说明 | 类型 | 备注 |
| id | 用户编号 | Varchar（32） | 不允许空，主键 |
| name | 用户名称 | Varchar（50） | 允许空 |
| type | 用户类型 | Varchar（20） | 允许空 |
| sex | 用户性别 | char(2) | 允许空 |
| max\_num | 最大可借数 | int(3) | 允许空 |
| days\_num | 可借天数 | int(11) | 允许空 |

**借阅信息表：**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 说明 | 类型 | 备注 |
| book\_id | 图书编号 | Varchar(32) | 允许为空，主键 |
| reader\_id | 用户编号 | Varchar(32) | 允许为空，主键 |
| borrow\_date | 借出时间 | date | 允许为空 |
| back\_date | 到期时间 | date | 允许为空 |
| is\_back | 是否归还 | smallint(1) | 允许为空 |

**用户信息表：**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 说明 | 类型 | 备注 |
| Reader\_id | 用户流水号 | int(11) | 不允许为空,主键 |
| name | 用户名称 | Varchar（50） | 允许为空 |
| pass | 用户密码 | Varchar（50） | 允许为空 |
| is\_admin | 是否为管理员 | smallint(1) | 允许为空 |

**销售信息表：**

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 说明 | 类型 | 备注 |
| book\_id | 图书编号 | Varchar(32) | 允许为空.主键 |
| buy\_date | 借出时间 | date | 允许为空 |

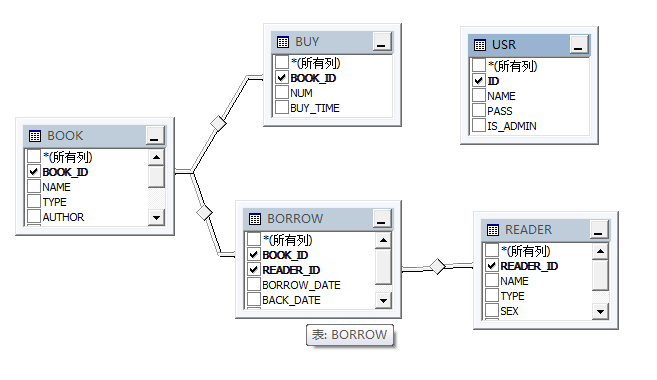
1. **物理结构设计**

****

****

**四、系统功能程序设计**

关系图：



**存储过程脚本**：

（创建存储过程统计指定时间段内各种图书、期刊的借阅、零售数量；统计指定客户借阅、购买情况（书名（刊物名）、数量））

GO

IF EXISTS(SELECT \* FROM SYS.OBJECTS WHERE NAME='BOOK\_ID\_TJ')

DROP PROC BOOK\_ID\_TJ

GO

CREATE PROCEDURE BOOK\_ID\_TJ

@COUNT INT OUTPUT,

@BOOK\_ID\_TJ VARCHAR(32),

@DATE\_BEGIN DATETIME,

@DATE\_END DATETIME

AS

SELECT @COUNT=COUNT(BOOK\_ID) FROM BORROW

WHERE BOOK\_ID=@BOOK\_ID\_TJ AND BORROW\_DATE BETWEEN @DATE\_BEGIN AND @DATE\_END

//测试存储过程

GO

DECLARE @COUNT INT ,

@BOOK\_ID\_TJ VARCHAR(32),

@DATE\_BEGIN DATETIME,

@DATE\_END DATETIME

SELECT @DATE\_BEGIN='2013/1/16 0:00:00'

SELECT @DATE\_END='2013/1/17 1:00:00'

SELECT @BOOK\_ID\_TJ='1'

EXEC BOOK\_ID\_TJ

@COUNT OUTPUT,@BOOK\_ID\_TJ ,@DATE\_BEGIN, @DATE\_END

PRINT @COUNT

GO

**触发器脚本：**

（借阅或归还时自动修改图书或期刊的在册数量，零售时修改图书或期刊 的总量和在册数量）

触发器1

CREATE TRIGGER INSERT\_BOOK\_READER

ON BORROW FOR INSERT

AS DECLARE @BOOK\_ID\_NEW VARCHAR(32),@READER\_ID\_NEW VARCHAR(32)

SELECT @BOOK\_ID\_NEW=BOOK\_ID,@READER\_ID\_NEW=READER\_ID FROM INSERTED

UPDATE BOOK SET STOCK=STOCK-1 WHERE BOOK.BOOK\_ID=@BOOK\_ID\_NEW

AND EXISTS(SELECT \* FROM BORROW WHERE BOOK\_ID=@BOOK\_ID\_NEW

AND READER\_ID=@READER\_ID\_NEW AND IS\_BACK='0')

UPDATE READER SET MAX\_NUM=MAX\_NUM-1 WHERE READER\_ID=@READER\_ID\_NEW

AND EXISTS(SELECT \* FROM BORROW WHERE BOOK\_ID=@BOOK\_ID\_NEW

AND READER\_ID=@READER\_ID\_NEW AND IS\_BACK='0')

触发器2

CREATE TRIGGER UPDATE\_BOOK\_READER

ON BORROW FOR UPDATE

AS DECLARE @BOOK\_ID\_NEW VARCHAR(32),@READER\_ID\_NEW VARCHAR(32)

SELECT @BOOK\_ID\_NEW=BOOK\_ID,@READER\_ID\_NEW=READER\_ID FROM INSERTED

UPDATE BOOK SET STOCK=STOCK+1 WHERE BOOK.BOOK\_ID=@BOOK\_ID\_NEW

AND EXISTS(SELECT \* FROM BORROW WHERE BOOK\_ID=@BOOK\_ID\_NEW

AND READER\_ID=@READER\_ID\_NEW AND IS\_BACK='1')

UPDATE READER SET MAX\_NUM=MAX\_NUM+1 WHERE READER\_ID=@READER\_ID\_NEW

AND EXISTS(SELECT \* FROM BORROW WHERE BOOK\_ID=@BOOK\_ID\_NEW

AND READER\_ID=@READER\_ID\_NEW AND IS\_BACK='1')

触发器3

CREATE TRIGGER BUYUPDATE\_BOOK\_READER

ON BUY FOR INSERT

AS DECLARE @BOOK\_ID\_NEW VARCHAR(32)

SELECT @BOOK\_ID\_NEW=BOOK\_ID FROM INSERTED

UPDATE BOOK SET STOCK=STOCK+1 WHERE BOOK.BOOK\_ID=@BOOK\_ID\_NEW

**数据库恢复与备份：**

数据库的完全备份

USE SUN

GO

BACKUP DATABASE SUN

TO DISK='F:\备份\FULLBACKUP\_SUN'

WITH INIT

GO

数据库差异备份

USE SUN

GO

BACKUP DATABASE SUN

TO DISK='F:\备份\CBACKUP\_SUN'

WITH DIFFERENTIAL

GO

数据库的备份还原

USE MASTER

GO

RESTORE DATABASE SUN

FROM DISK='F:\备份\FULLBACKUP\_SUN'

WITH NORECOVERY

GO

RESTORE DATABASE SUN

FROM DISK='F:\备份\CBACKUP\_SUN'

WITH REPLACE

GO