#1

目录

#1

SQL数据安全性

#2 一、实验目的

1. 掌握数据库应用开发程序设计方法。

#2 二、实验环境

1. 操作系统：Windows 10
2. 数据库管理系统： MySQL
3. 编程语言：C++
4. 可视化图形编程：QT Creator 4.9.1+QT5.13.0(MinGW 64bit)

#2 三、实验流程

#3 3.1 图书管理系统数据库的设计

3.1.1 逻辑结构设计DDL语句

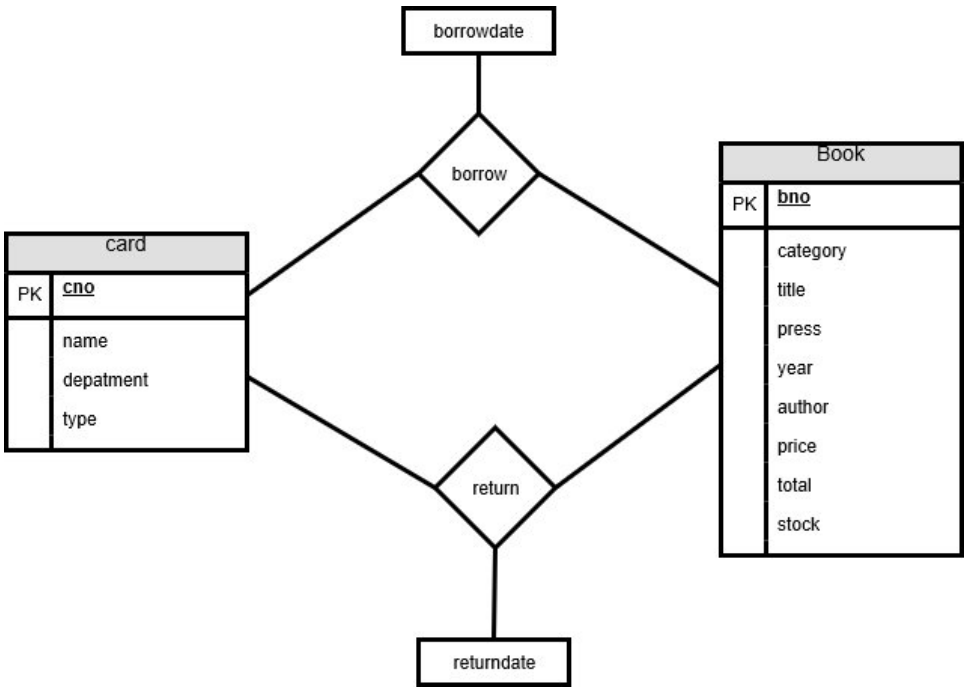
```
create  table book
   (bno char(10),
   category     varchar(10),
   title    varchar(20),
   press    varchar(20),
   year int,
   author varchar(10),
   price    decimal(7,2),
   total    int,
   stock    int,
   primary key(bno));

  create table card
  (cno char(7),
  name varchar(10),
  department varchar(40),
  type char(1),
  primary key(cno),
  check(type in('T','S')));

  create table borrow
  (cno char(7),
  bno  char(8),
  borrow_date int,
  return_date int,
  primary key(cno,bno),
  foreign key (cno) references card(cno),
  foreign key (bno) references book(bno));

```

3.1.2 E-R图



3.2 关键代码展示

3.2.1 QT连接数据库

```
QT += sql
INCLUDEPATH += D:\mysql-8.0.19-winx64\mysql-8.0.19-winx64\include
```

3.2.2 打开数据库

```
void Addcard::opendb(){
    db=QSqlDatabase::addDatabase("QMYSQL");
    db.setHostName("localhost");       //连接数据库主机名，这里需要注意（若填的为”127.0.0.1“，出现不能连接，则改为localhost)
    db.setPort(3306);                  //连接数据库端口号，与设置一致
    db.setDatabaseName(database);       //连接数据库名，与设置一致
    db.setUserName(username);           //数据库用户名，与设置一致
    db.setPassword(password);     //数据库密码，与设置一致
    db.open();
}
```

3.2.3 查询数据库中的内容

```
//*****************举例：查询是否已经存在该借书证**********
QString str1="select * from card where cno='"+cno+"';";
qDebug()<<str1;
QSqlQuery query1;
query1.exec(str1);
if(query1.next()){
    qDebug()<<"存在该卡号";
    ui->label_info->setText("已经存在卡号"+cno);
}
```

3.2.4 修改数据库中内容

```
//*****************举例：修改书本类型***********
QSqlQuery query2;
QString str2,str0;

str0=" WHERE bno='"+lst[0]+"';";
bool successupdate=1;
if(!lst[1].isEmpty()){
    str2="UPDATE book SET category='"+lst[1]+"'"+str0;
    qDebug()<<str2;
    successupdate=successupdate*query2.exec(str2);
}
```

3.2.5 删除数据库中信息

```
//*****************举例：删除管理员***********
str1="drop user "+delete_name+"@'localhost';";
qDebug()<<str1;
bool success1=query.exec(str1);
//*****************举例：借书证***********
str1="delete from card where cno='"+cno+"';";
QSqlQuery query2;
bool success2=query2.exec(str1);
```

3.2.6 新建管理员并授权

```
        QSqlQuery query;
        QString str1;
        str1="CREATE USER '"+new_username+"'@'localhost' IDENTIFIED BY
'"+new_password+"';";
        qDebug()<<str1;
        QString str2;
        str2="GRANT select ON libmanage.* TO
'"+new_username+"'@'localhost';";
        qDebug()<<str2;
        QString str3;
        str3="GRANT create ON libmanage.* TO
'"+new_username+"'@'localhost';";
        QString str4;
        str4="GRANT delete ON libmanage.* TO
'"+new_username+"'@'localhost';";
        QString str5;
        str5="GRANT insert ON libmanage.* TO
'"+new_username+"'@'localhost';";
        bool success1=query.exec(str1);
        bool success2=query.exec(str2);
        bool success3=query.exec(str3);
        bool success4=query.exec(str4);
        bool success5=query.exec(str5);
        if(success1*success2*success3*success4*success5){
            qDebug()<<"创建用户成功";
            ui->label_createinfo->setText("创建用户"+new_username+"成功");
        }
        else{
            qDebug()<<"创建用户失败";
            ui->label_createinfo->setText("创建用户"+new_username+"失败");
        }
```

#3 3.3 程序运行展示

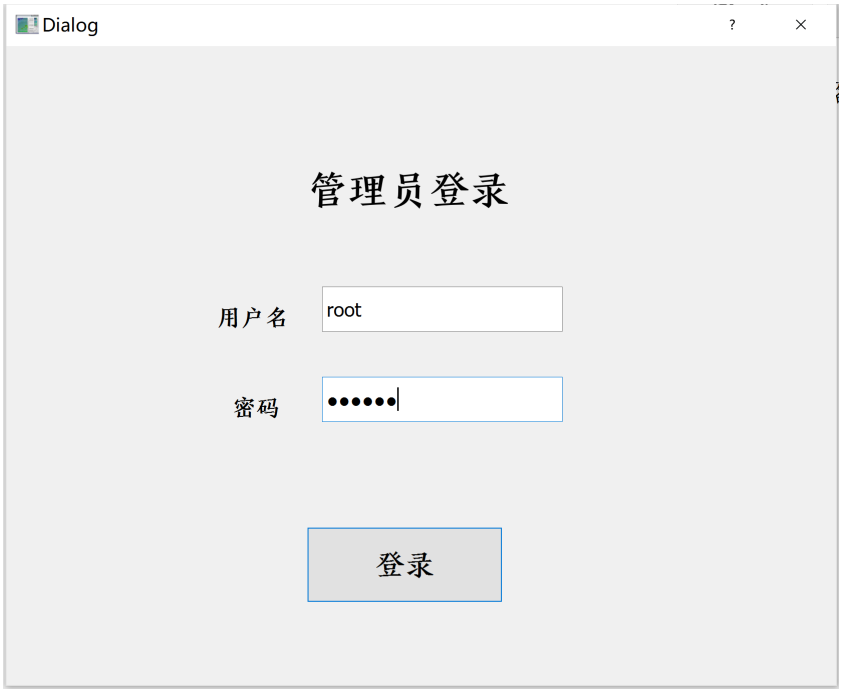
3.3.1 输入目标数据库

> 因不同环境中，图书管理系统的目标数据库名称没有被严格规定，因此需要在程序运行之初输入目标数据库名称，如libmanage等，否则无法进入管理程序

3.3.2 管理员登录

> 第一次进入主界面时，身份为游客，游客身份不具有图书管理的任何权限
>
> 点击“登入”按钮，进行管理员登录操作

> 管理员登录界面
>
> 输入本地的图书管理系统数据库的管理员信息（不局限于root管理员）
>
> 其中密码以暗码形式输入，避免信息泄露

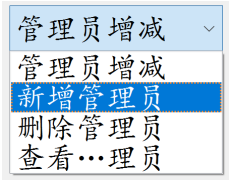


> 若以root身份登录，具有所有权限
>
> 否则不具有管理员删改权限

> 登出后，失去全部权限，并回归游客身份

3.3.3 管理员操作

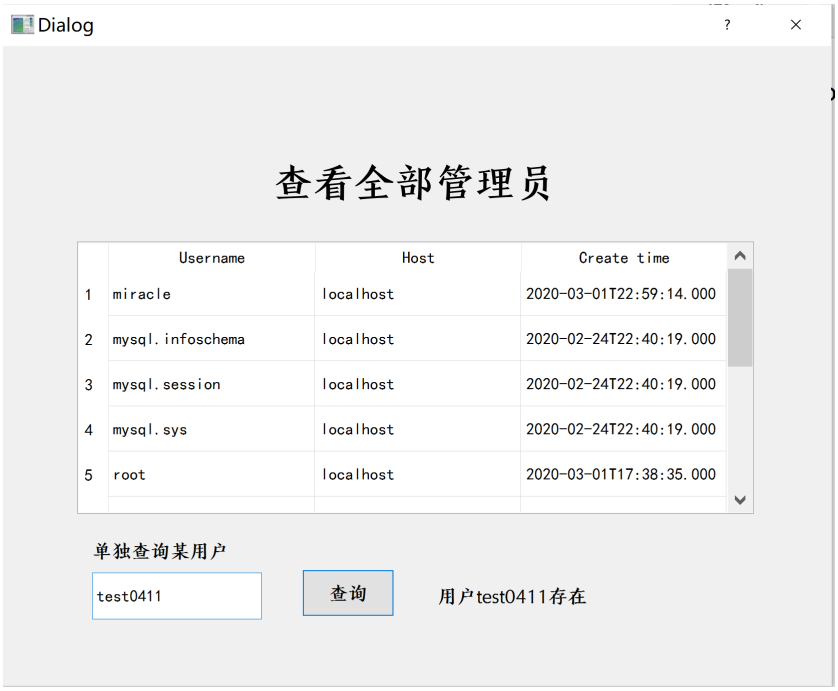
3.3.3.1 增加管理员



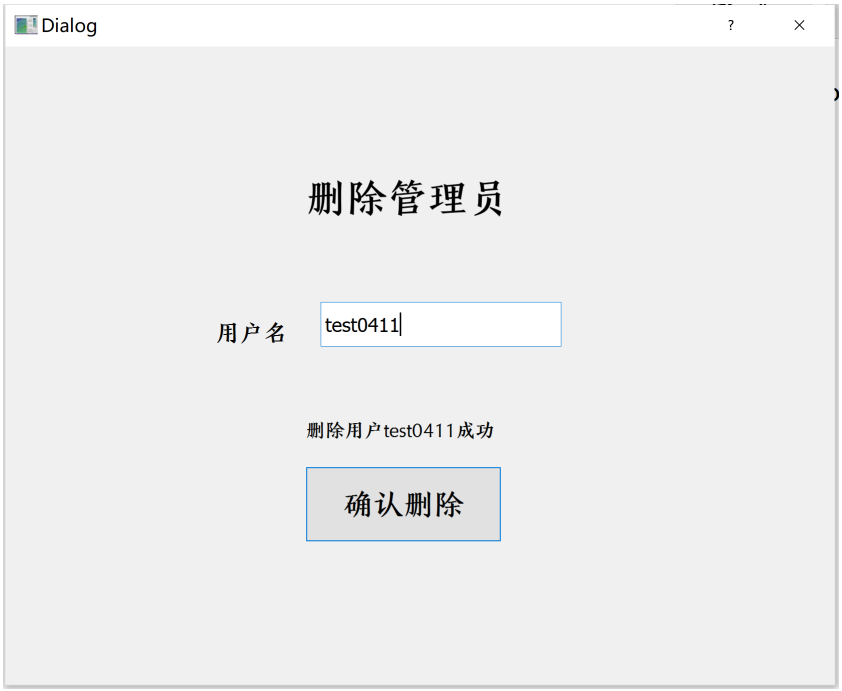
> 不可重复创建同名管理员

3.3.3.2 查看全部管理员

> 进入该页面列表显示该数据库所有管理员信息
>
> 也可通过查询按钮单独查询某管理员是否存在



3.3.3.3 删除管理员

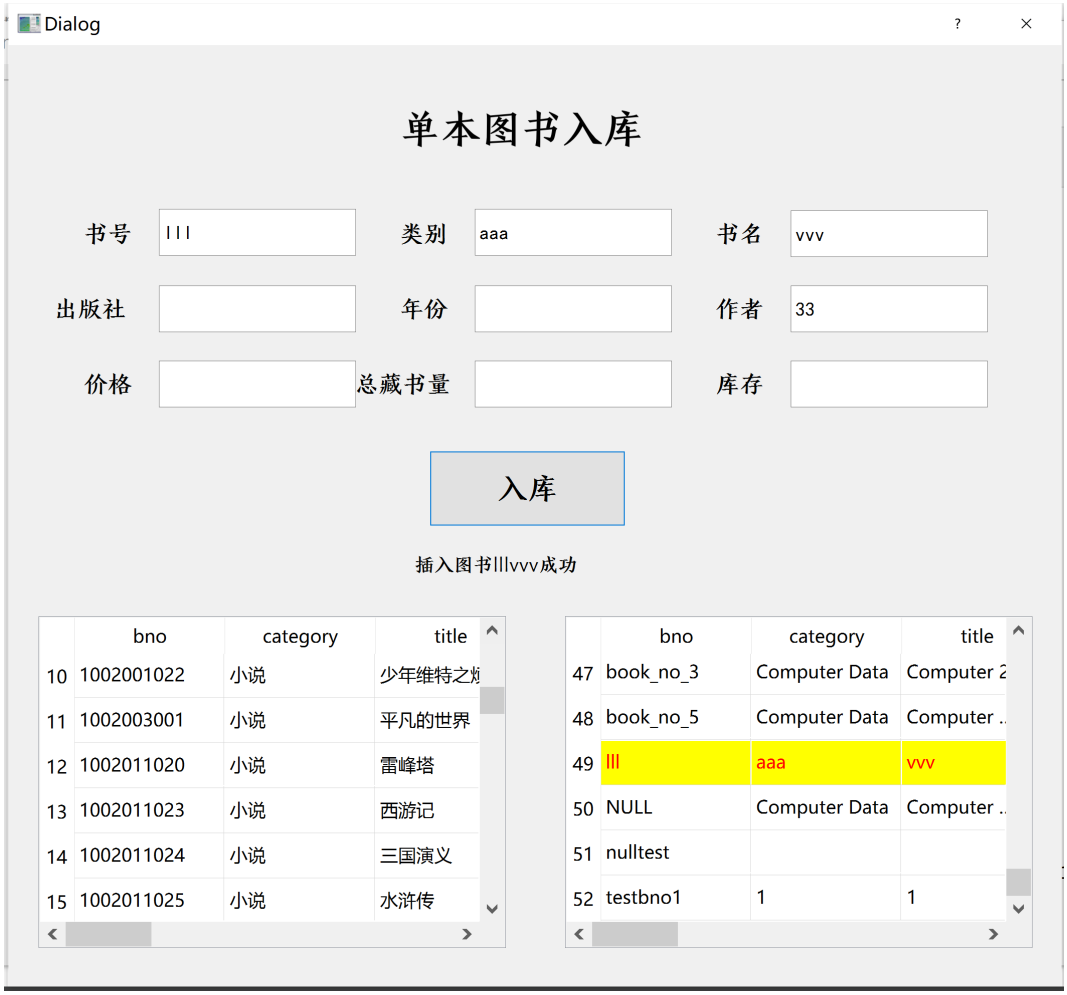


3.3.4 图书入库

3.3.4.1 单本图书入库

> 年份文本框限定只能输入四位数字
>
> 价格文本框限定最多输入两位小数
>
> 总藏书量文本框与库存文本框限定只能输入数字

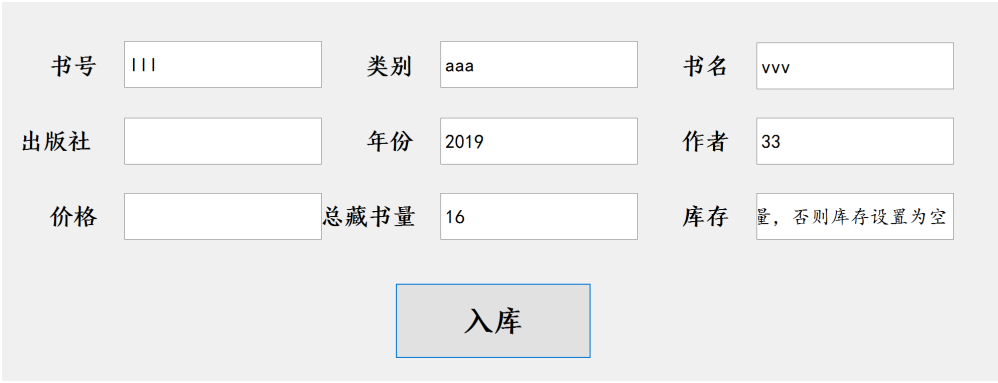
> 新书插入


Dialog
单本图书入库
书号
lll
类别
aaa
书名
vvv
出版社
年份
作者
33
价格
总藏书量
库存
入库
插入图书lllvvv成功
bno category title
10 1002001022 小说 少年维特之烦
11 1002003001 小说 平凡的世界
12 1002011020 小说 雷峰塔
13 1002011023 小说 西游记
14 1002011024 小说 三国演义
15 1002011025 小说 水浒传
bno category title
47 book_no_3 Computer Data Computer 2
48 book_no_5 Computer Data Computer ..
49 lll aaa vvv
50 NULL Computer Data Computer ..
51 nulltest
52 testbno1 1 1

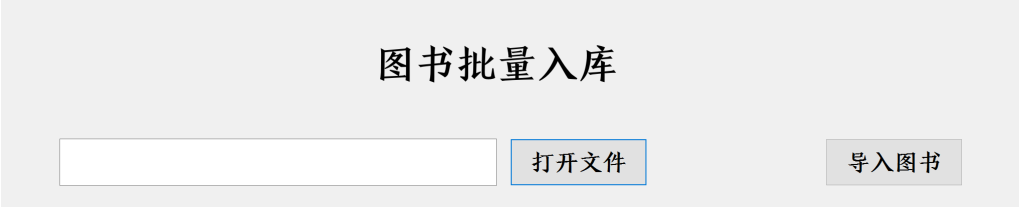
图书更新


Dialog
单本图书入库
书号
lll
类别
aaa
书名
vvv
出版社
年份
2019
作者
33
价格
总藏书量
库存
入库
更新信息lll成功
title press year
47 Computer 222 new 2005
48 Computer ... press2 2005
49 vvv 0
50 Computer ... press2 2005
51 0
52 1 1 1
title press year
47 Computer 222 new 2005
48 Computer ... press2 2005
49 vvv 2019
50 Computer ... press2 2005
51 0
52 1 1 1

> 只有填写总藏书量后才能填写库存
>
> 库存大于总藏书量时，库存输入无效



3.3.4.2 批量入库

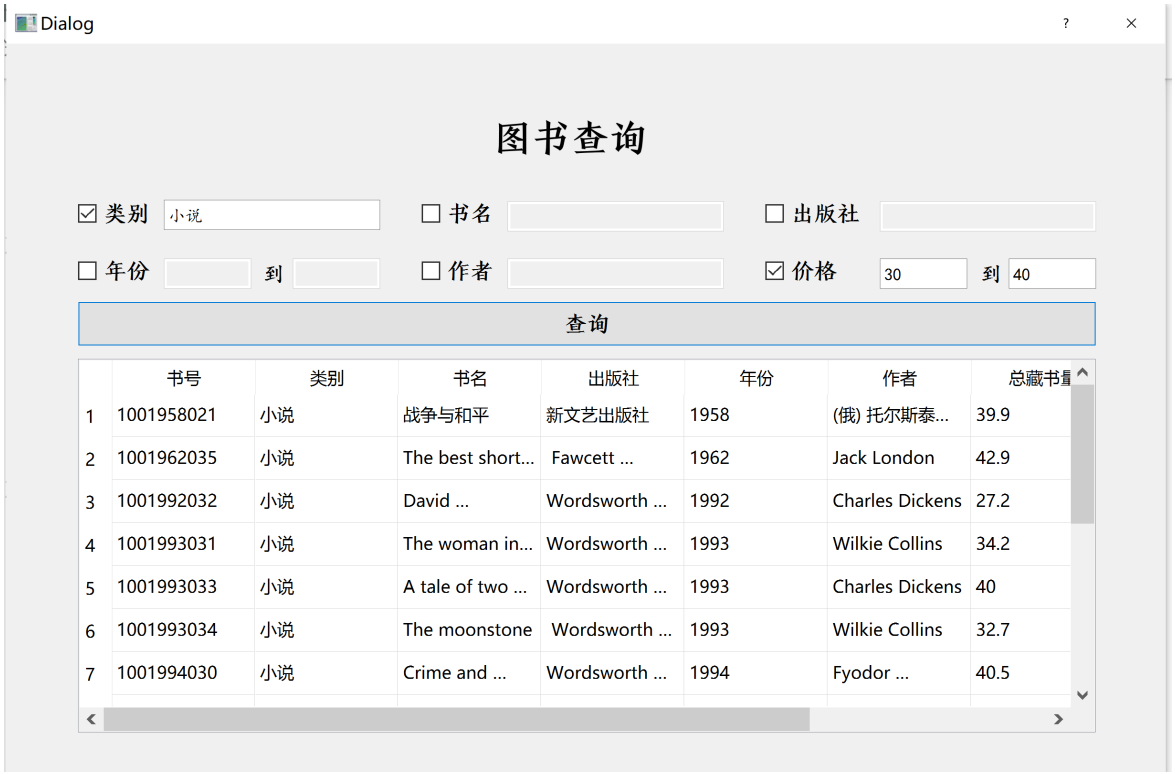


> 点击打开文件后，选择批量入库的图书文件，默认格式为.txt
>
> 选择导入图书

> 上方表格为导入的文件信息，与是否是新书导入或旧书更新
>
> 左表为导入前表格，右表为导入后表格

3.3.5 图书查询

> 支持多属性查询



3.3.6 借书

3.3.6.1 借书操作

> 若书本已无库存

> 若输入错误书号

3.3.6.2 查询已借书籍

> 数据库中关于归还日期和借书日期均以int类型记录
>
> 因此默认初始时间为1970年1月1日，单位天
>
> 通过函数获得具体的借书与还书日期

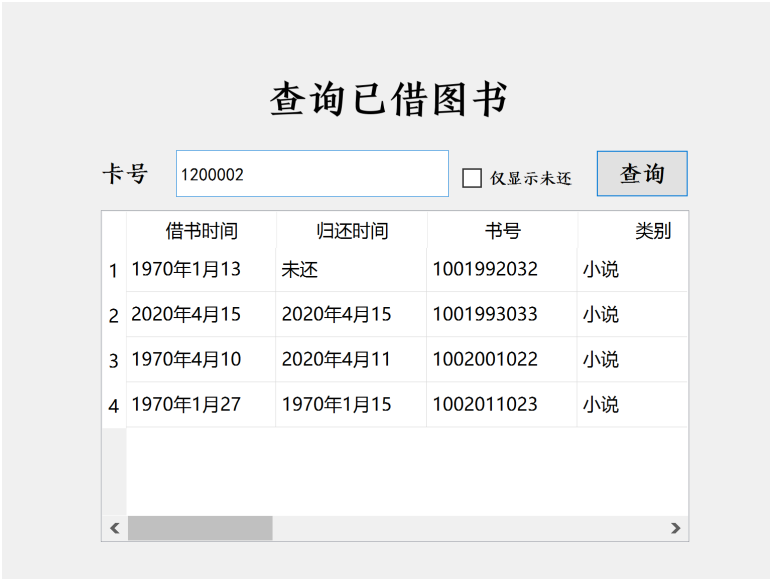
> 仅显示该借书证未还图书

> 若输入错误的卡号，则显示不存在该借书证

3.3.7 借书

3.3.7.1 借书操作

> 成功归还

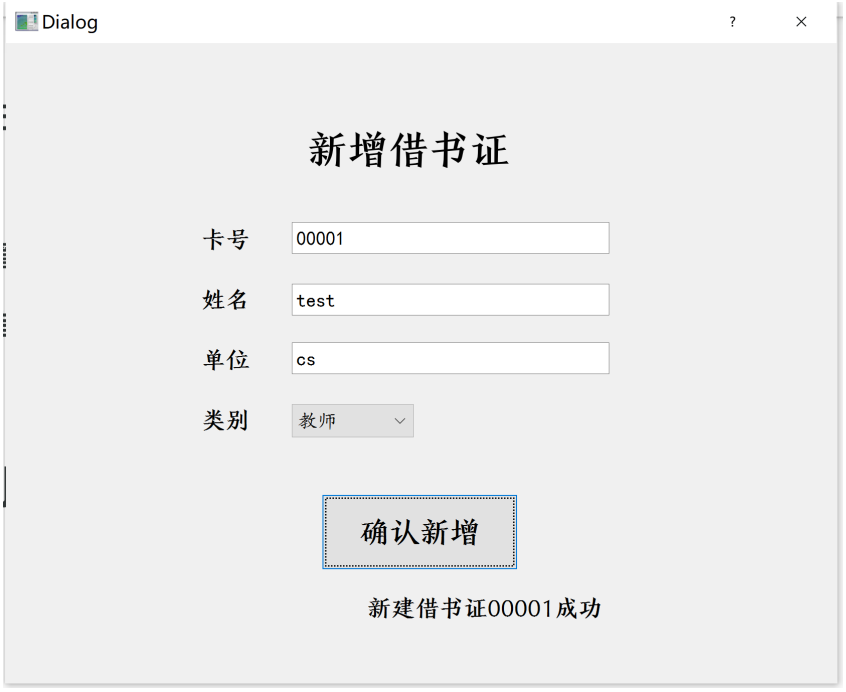


3.3.7.2 查询已借书籍

同3.3.6.2

3.3.8 借书证管理

3.3.8.1 新增借书证

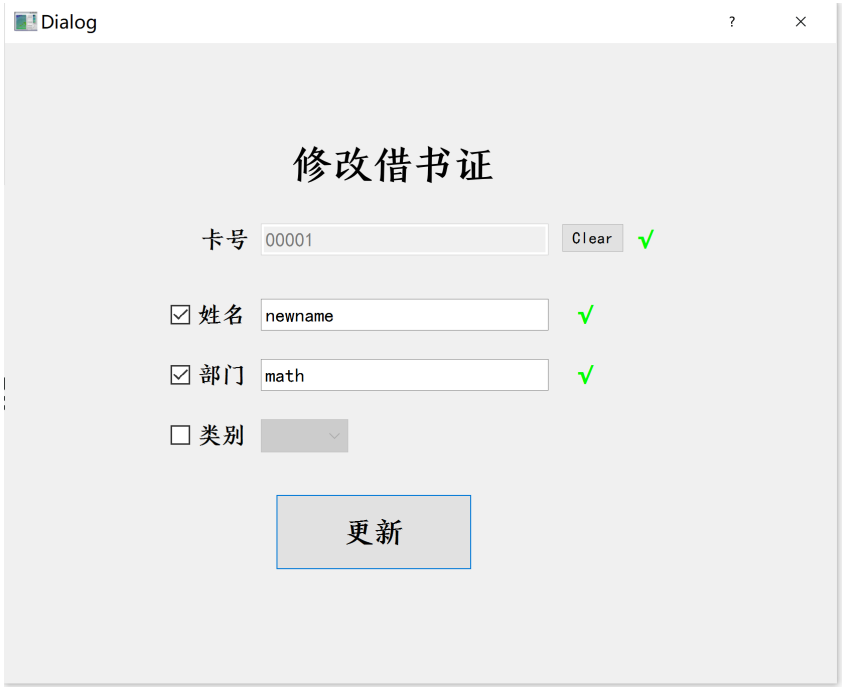


> 若卡号已存在，则不能成功新增

3.3.8.2 修改借书证

> 需输入卡号后，按旁边按钮确认卡号存在，才能进行后续操作

> 成功修改借书证后，会出现绿色对勾，否则说明信息有误


Dialog
修改借书证
卡号
00001
Clear
姓名
newname
部门
math
类别
更新


Dialog
删除借书证
卡号
00001
确认删除
删除借书证00001成功

3.3.8.3 删除借书证

若输入不存在的卡号

3.3.8.4 查询借书证

#2 四、遇到的问题及解决方法

#3 error1 不能正确加载mysql.dll驱动

问题：qt官方只提供32位环境，而我的电脑安装的是64位的mysql，驱动程序存在但不能适配

解决方案：对qt的sql.pro进行重新编译

参考：https://blog.csdn.net/qq_38812184/article/details/99955301?tdsourcetag=s_pcqq_aiomsg

#3 error2 通过qsqlquery语句连接数据库存在问题

问题： 在qt环境中，sql的操作方式有很多，例如qsqlquery与qsqlquerymodel，二者虽然类似但使用方式不同，不能混合使用

#2 五、总结

通过这次实验，我对于mysql的嵌入式开发与odbc语言的运用有了更深的认识，也渐渐认识到开发一款可视化数据库应用软件的重要性与难度，为日后的软件开发奠定了基础。此外，我对于QT图形库的运用有了更加熟练的操作。