

目录

目录

SQL数据安全性

- 一、实验目的
- 二、实验环境
- 三、实验流程
 - 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
 - 3.1.2 E-R图
 - 3.2 关键代码展示
 - 3.2.1 QT连接数据库
 - 3.2.2 打开数据库
 - 3.2.3 查询数据库中的内容
 - 3.2.4 修改数据库中内容
 - 3.2.5 删除数据库中信息
 - 3.2.6 新建管理员并授权
 - 3.3 程序运行展示
 - 3.3.1 输入目标数据库
 - 3.3.2 管理员登录
 - 3.3.3 管理员操作
 - 3.3.3.1 增加管理员
 - 3.3.3.2 查看全部管理员
 - 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
 - 3.3.4.2 批量入库
 - 3.3.5 图书查询
 - 3.3.6 借书
 - 3.3.6.1 借书操作
 - 3.3.6.2 查询已借书籍
 - 3.3.7 借书
 - 3.3.7.1 借书操作
 - 3.3.7.2 查询已借书籍
 - 3.3.8 借书证管理
 - 3.3.8.1 新增借书证
 - 3.3.8.2 修改借书证
 - 3.3.8.3 删除借书证
 - 3.3.8.4 查询借书证
- 四、遇到的问题及解决方法
 - error1 不能正确加载mysql.dll驱动
 - error2 通过sqlquery语句连接数据库存在问题
- 五、总结

#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)

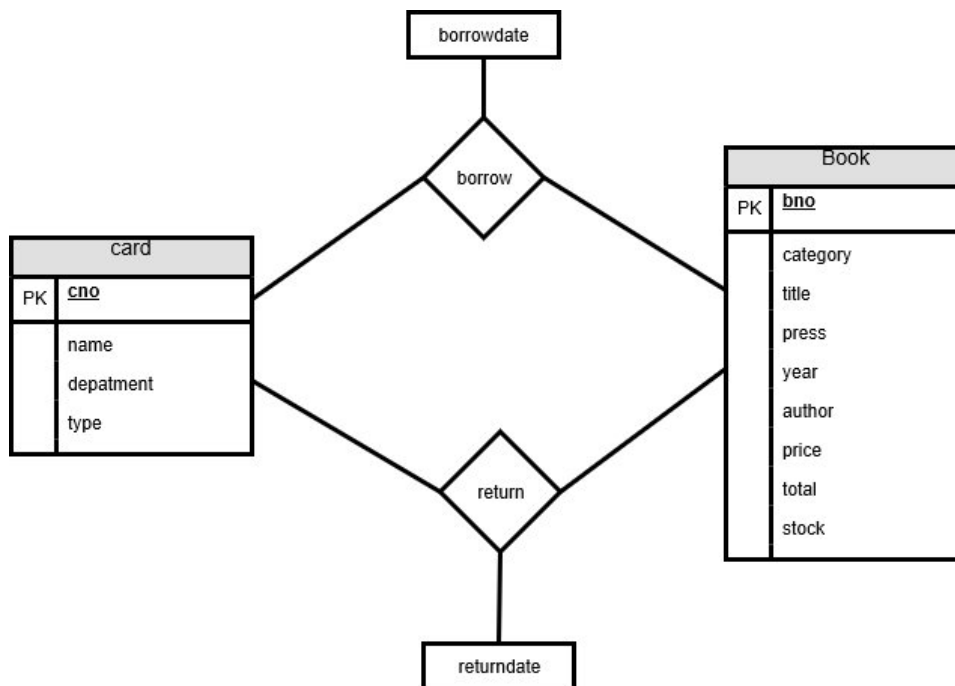
#2 三、实验流程

#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

```
1  create table book
2      (bno char(10),
3       category varchar(10),
4       title varchar(20),
5       press varchar(20),
6       year int,
7       author varchar(10),
8       price decimal(7,2),
9       total int,
10      stock int,
11      primary key(bno));
12
13  create table card
14      (cno char(7),
15       name varchar(10),
16       department varchar(40),
17       type char(1),
18       primary key(cno),
19       check(type in('T','S')));
20
21  create table borrow
22      (cno char(7),
23       bno char(8),
24       borrow_date int,
25       return_date int,
26       primary key(cno,bno),
27       foreign key (cno) references card(cno),
28       foreign key (bno) references book(bno));
29
```

3.1.2 E-R图



#3 3.2 关键代码展示

3.2.1 QT连接数据库

```

1 QT += sql
2 INCLUDEPATH += D:\mysql-8.0.19-win64\mysql-8.0.19-win64\include
  
```

3.2.2 打开数据库

```

1 void Addcard::opendb(){
2     db=QSqlDatabase::addDatabase("QMYSQL");
3     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）
4     db.setPort(3306); //连接数据库端口号，与设置一致
5     db.setDatabaseName(database); //连接数据库名，与设置一致
6     db.setUserName(username); //数据库用户名，与设置一致
7     db.setPassword(password); //数据库密码，与设置一致
8     db.open();
9 }
  
```

3.2.3 查询数据库中的内容

```

1 //*****举例：查询是否已经存在该借书证*****
2 QString str1="select * from card where cno='"+cno+"'";
3 qDebug()<<str1;
4 QSqlQuery query1;
5 query1.exec(str1);
6 if(query1.next()){
7     qDebug()<<"存在该卡号";
8     ui->label_info->setText("已经存在卡号"+cno);
9 }
  
```

3.2.4 修改数据库中内容

```

1 //*****举例：修改书本类型*****
2 QSqlQuery query2;
3 QString str2,str0;
4
5 str0=" WHERE bno='"+lst[0]+'";
6 bool successupdate=1;
7 if(!lst[1].isEmpty()){
8     str2="UPDATE book SET category='"+lst[1]+'"+str0;
9     qDebug()<<str2;
10    successupdate=successupdate*query2.exec(str2);
11 }

```

3.2.5 删除数据库中信息

```

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

```

3.2.6 新建管理员并授权

```

1     QSqlQuery query;
2     QString str1;
3     str1="CREATE USER '"+new_username+"'@'localhost' IDENTIFIED BY
4     '"+new_password+'";
5     qDebug()<<str1;
6     QString str2;
7     str2="GRANT select ON libmanage.* TO
8     '"+new_username+"'@'localhost'";
9     qDebug()<<str2;
10    QString str3;
11    str3="GRANT create ON libmanage.* TO
12    '"+new_username+"'@'localhost'";
13    QString str4;
14    str4="GRANT delete ON libmanage.* TO
15    '"+new_username+"'@'localhost'";
16    QString str5;
17    str5="GRANT insert ON libmanage.* TO
18    '"+new_username+"'@'localhost'";
19    bool success1=query.exec(str1);
20    bool success2=query.exec(str2);
21    bool success3=query.exec(str3);
22    bool success4=query.exec(str4);
23    bool success5=query.exec(str5);
24    if(success1*success2*success3*success4*success5){
25        qDebug()<<"创建用户成功";
26        ui->label_createinfo->setText("创建用户"+new_username+"成功");
27    }
28    else{
29        qDebug()<<"创建用户失败";
30        ui->label_createinfo->setText("创建用户"+new_username+"失败");
31    }

```

#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

Dialog

管理员登录

用户名

密码

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

MainWindow

用户名: root

图书管理系统

Created by Miracle_Zero 2020/4/11

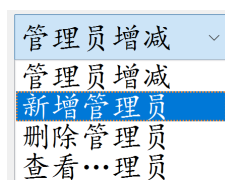


登出后，失去全部权限，并回归游客身份



3.3.3 管理员操作

3.3.3.1 增加管理员



Dialog

创建管理员

用户名

密码

创建用户test0411成功

不可重复创建同名管理员

用户名

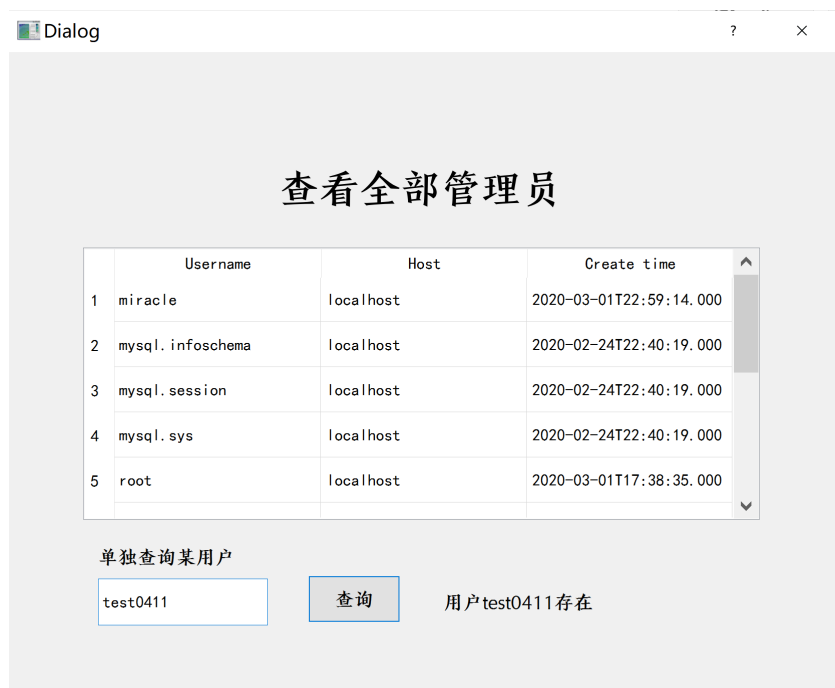
密码

创建用户test0411失败

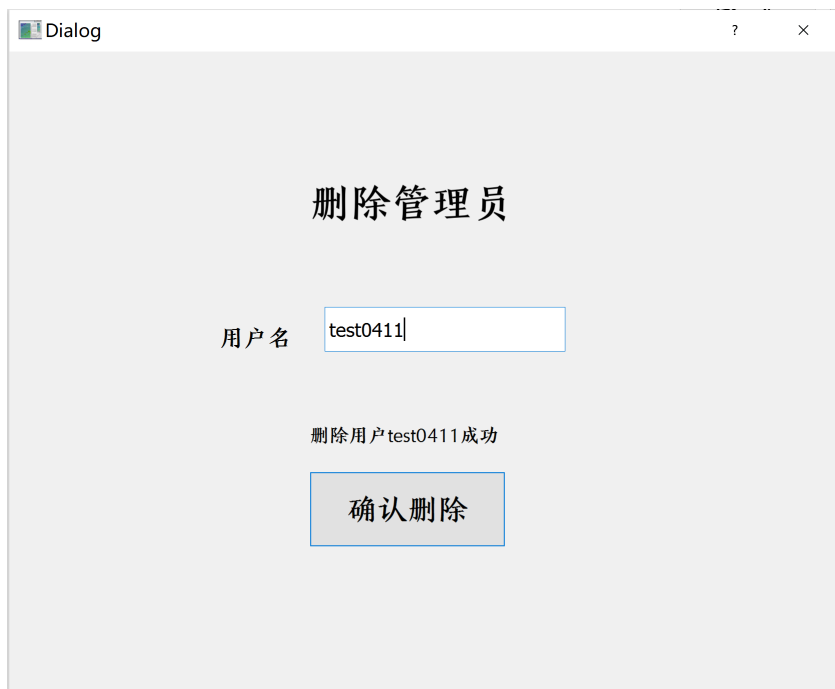
3.3.3.2 查看全部管理员

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

类别

书名

出版

年份

作者

价格

总藏书量

库存

入库

插入图书lllvvv成功

	bno	category	title
10	1002001022	小说	少年维特之烦恼
11	1002003001	小说	平凡的世界
12	1002011020	小说	雷锋塔
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

类别

书名

出版

年份

作者

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

书号

类别

书名

出版

年份

作者

价格

总藏书量

库存

入库

3.3.4.2 批量入库

图书批量入库

打开文件

导入图书

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

图书批量入库

C:/Users/ljy28/Desktop/input.txt

打开文件

导入图书

	tips	bno	category	title	press	year	
1	更新信息	book_no_1	Computer new	Computer Architecture	press1	2004	au
2	更新信息	book_no_2	Computer Data	Computer 111	press2	2005	ne

	bno	category	title
1			
2	0000	11	111
3	1001958021	小说	战争与和平
4	1001962035	小说	The best short stories of Ja
5	1001992032	小说	David Copperfield
6	1001993031	小说	The woman in white
7	1001993033	小说	A tale of two cities
8	1001993034	小说	The moonstone
9	1001994030	小说	Crime and punishment

	bno	category	title
1			
2	0000	11	111
3	1001958021	小说	战争与和平
4	1001962035	小说	The best short stories of Ja
5	1001992032	小说	David Copperfield
6	1001993031	小说	The woman in white
7	1001993033	小说	A tale of two cities
8	1001993034	小说	The moonstone
9	1001994030	小说	Crime and punishment

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

Dialog

?

×

图书查询

☒ 类别

小说

☐ 书名

☐ 出版社

☐ 年份

到

☐ 作者

☒ 价格

30

到

40

查询

3.3.6 借书

3.3.6.1 借书操作

Dialog

?

×

借书

卡号

1200002

书号

1001993033

借书日期

2020. 4. 16

借书

借阅成功

若书本已无库存

Dialog

?

×

借书

卡号

1200002

书号

0000

借书日期

2020. 4. 16

借书

本书无库存，最近借阅时间为1970年1月1

若输入错误书号

Dialog

借书

卡号 1200002

书号 000011

借书日期 2020. 4. 16 借书

不存在该书号

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

Dialog

借书

卡号 1200002

书号 0000

借书日期 2020. 4. 16 借书

本书无库存，最近借阅时间为1970年1月1

仅显示该借书证未还图书

Dialog

查询已借图书

卡号 1200002 ☒ 仅显示未还 查询

	借书时间	归还时间	书号	类别
1	1970年1月13	未归还	1001992032	小说

若输入错误的卡号，则显示不存在该借书证

查询已借图书

卡号

12000021

☐ 仅显示未还

查询

	借书时间	归还时间	书号	类别
1				

不存在该借书证

3.3.7 借书

3.3.7.1 借书操作

Dialog

还书

卡号

1200002

书号

1001993033

还书日期

2020. 4. 16

还书

归还成功

成功归还

查询已借图书

卡号

1200002

☐ 仅显示未还

查询

	借书时间	归还时间	书号	类别
1	1970年1月13	未还	1001992032	小说
2	2020年4月15	2020年4月15	1001993033	小说
3	1970年4月10	2020年4月11	1002001022	小说
4	1970年1月27	1970年1月15	1002011023	小说

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

Dialog

?

×

新增借书证

卡号

00001

姓名

test

单位

cs

类别

教师

确认新增

新建借书证00001成功

若卡号已存在，则不能成功新增

Dialog

?

×

新增借书证

卡号

1200002

姓名

test

单位

cs

类别

教师

确认新增

已经存在卡号1200002

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

修改借书证

卡号

0000111

Check

×

☐

姓名

☐

部门

☐

类别

▼

更新

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

Dialog

?

×

删除借书证

卡号

00001

确认删除

删除借书证00001成功

若输入不存在的卡号

删除借书证

卡号

0000111

确认删除

不存在卡号0000111

借书卡查询

☐ 卡号

☐ 姓名

☒ 部门

哲学

☒ 类型

学生

查询



#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/qg_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。