

**报告名称：**

**班 级：**

**学 号：**

**姓 名：**

目录

[实验一 实验报告 5](#_Toc31413)

[一、 实验目的与要求 5](#_Toc16225)

[二、 实验代码 5](#_Toc12321)

[2.1 创建基本表 5](#_Toc5321)

[2.2 单表查询 5](#_Toc10379)

[2.3 分组统计查询 6](#_Toc13507)

[2.4 连接查询（自身连接） 6](#_Toc21394)

[2.5 连接查询（多表连接） 6](#_Toc7921)

[三、 运行结果 6](#_Toc12001)

[3.1创建基本表 6](#_Toc11416)

[3.2 单表查询 7](#_Toc24521)

[3.3 分组统计查询 7](#_Toc25187)

[3.4 连接查询（自身连接） 7](#_Toc25218)

[3.5 连接查询（多表连接） 8](#_Toc29972)

[四、实验总结 8](#_Toc17536)

[五、问题解决 8](#_Toc16644)

[5.1 问题一 8](#_Toc19086)

[实验二 实验报告 9](#_Toc12898)

[一、 实验目的与要求 9](#_Toc10369)

[二、 实验代码 9](#_Toc19589)

[2.1 带有IN谓词的子查询 9](#_Toc24514)

[2.2 带有比较运算符的子查询 9](#_Toc12509)

[2.3 带有ANY（SOME）或ALL谓词的子查询 9](#_Toc17162)

[2.4 带有EXISTS的子查询 10](#_Toc19001)

[2.5集合查询 10](#_Toc20729)

[三、 运行结果 10](#_Toc10114)

[3.1 带有IN谓词的子查询 10](#_Toc25717)

[3.2 带有比较运算符的子查询 10](#_Toc10531)

[3.3 带有ANY（SOME）或ALL谓词的子查询 11](#_Toc28915)

[3.4 带有EXISTS的子查询 11](#_Toc32421)

[3.5 集合查询 11](#_Toc29208)

[四、 实验总结 11](#_Toc7764)

[五、 问题解决 12](#_Toc12630)

[5.1 问题一 12](#_Toc28388)

[实验三 实验报告 13](#_Toc22891)

[一、 实验目的与要求 13](#_Toc23412)

[二、实验代码 13](#_Toc20210)

[2.1 插入一个元组 13](#_Toc3162)

[2.2 插入子查询结果 13](#_Toc15976)

[2.3 修改某元组的值 13](#_Toc5022)

[2.4 修改多个元组的值 14](#_Toc4937)

[2.5 带子查询的修改 14](#_Toc4629)

[2.6 删除多个元组的值 14](#_Toc26936)

[2.7 带子查询的删除 14](#_Toc17009)

[三、运行结果 15](#_Toc20303)

[3.1 插入一个元组 15](#_Toc457)

[3.2 插入子查询结果 15](#_Toc14091)

[3.3 修改某元组的值 15](#_Toc26979)

[3.4 修改多个元组的值 16](#_Toc29550)

[3.5 带子查询的修改 16](#_Toc27589)

[3.6 删除多个元组的值 16](#_Toc4680)

[3.7 带子查询的删除 16](#_Toc14169)

[四、实验总结 17](#_Toc26829)

[五、问题解决 17](#_Toc29479)

[5.1 问题一 17](#_Toc13798)

[实验四 实验报告 18](#_Toc20085)

[一、 实验目的与要求 18](#_Toc4050)

[二、实验代码 18](#_Toc25164)

[2.1 创建视图 18](#_Toc16409)

[2.2 删除视图 18](#_Toc7215)

[2.3 查询视图 18](#_Toc2022)

[2.4 对非行列子集视图的查询 18](#_Toc25483)

[三、运行结果 19](#_Toc22578)

[3.1 创建视图 19](#_Toc30632)

[3.2 删除视图 19](#_Toc9336)

[3.3 查询视图 19](#_Toc12835)

[3.4 对非行列子集视图的查询 19](#_Toc27584)

[四、实验总结 20](#_Toc31290)

[五、问题解决 20](#_Toc13124)

[5.1问题一 20](#_Toc22385)

[5.2 问题二 20](#_Toc18172)

[实验五 实验报告 21](#_Toc18004)

[一、 实验目的与要求 21](#_Toc30677)

[二、实验代码 21](#_Toc18535)

[2.1 系统主界面(MainWindow) 21](#_Toc3172)

[2.2 登录界面(load) 33](#_Toc12421)

[2.3 学生信息查询界面 34](#_Toc72)

[2.4 选课信息查询界面 35](#_Toc30521)

[2.5 学生操作界面 35](#_Toc20389)

[2.6 选课操作界面 37](#_Toc32470)

[2.7 成绩统计界面 39](#_Toc14164)

[三、运行结果 40](#_Toc11609)

[3.1 系统主界面(MainWindow) 40](#_Toc22462)

[3.2 登录界面(load) 41](#_Toc12893)

[3.3 学生信息查询界面 41](#_Toc22407)

[3.4 选课信息查询界面 42](#_Toc16806)

[3.5 学生操作界面 42](#_Toc19555)

[3.6 选课操作界面 44](#_Toc3855)

[3.7 成绩统计界面 45](#_Toc9170)

[3.8课程操作界面 46](#_Toc32481)

[四、实验总结 48](#_Toc21151)

[五、问题解决 48](#_Toc21974)

[5.1 问题一 48](#_Toc2376)

[5.2 问题二 48](#_Toc20064)

[5.3 问题三 49](#_Toc23742)

# 实验一 实验报告

## 实验目的与要求

(1)掌握SQL程序设计基本规范，熟练运用SQL语言实现数据定义、基本查询，包括单表查询、分组统计查询、连接查询

(2)能按照SQL程序设计规范写出具体的SQL查询语句，并调试通过

(3)支撑课程目标②④

## 实验代码

### 2.1 创建基本表

1. create table student
2. (sno CHAR(9) PRIMARY KEY,
3. sname CHAR(20) UNIQUE,
4. ssex CHAR(2),
5. sage SMALLINT,
6. sdept CHAR(20));
7. create table course
8. (cno CHAR(4) PRIMARY KEY,
9. cname CHAR(40) NOT NULL,
10. cpno CHAR(4),
11. ccredit SMALLUNT,
12. FOREIGN KEY(cpno) REFERENCES course(cno));
13. create table sc
14. (sno CHAR(9),
15. cno char(4),
16. grade smallint,
17. primary key(sno ,cno),
18. foreign key(sno) references student(sno),
19. foreign key(cno) references course(cno));

### 2.2 单表查询

1. select \*
2. from student;

### 2.3 分组统计查询

1. select Cno,count(Sno)
2. from sc
3. group by Cno;

### 2.4 连接查询（自身连接）

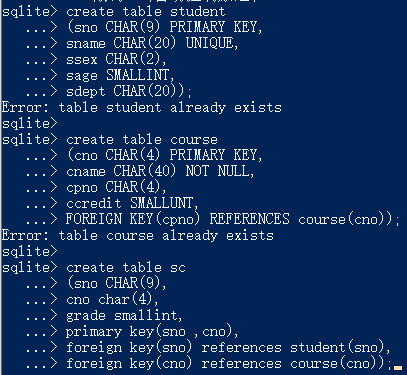
1. select f1.Cno,f2.Cno
2. from course f1,course f2
3. where f1.Cpno=f2.Cpno;

### 2.5 连接查询（多表连接）

1. select student.Sno,Sname,Cname,grade
2. from student,sc,course
3. where student.Sno=sc.Sno AND sc.Cno=course.Cno;

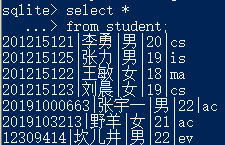
## 运行结果

### 3.1创建基本表

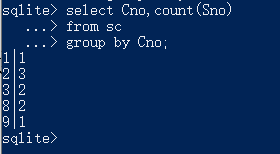


### 3.2 单表查询

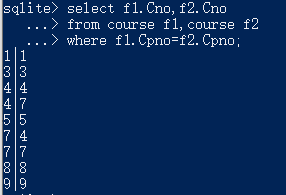
单表查询结果如下：



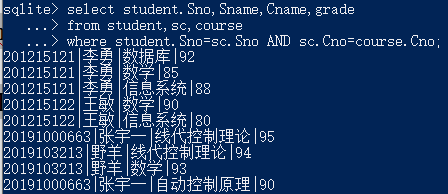
### 3.3 分组统计查询



### 3.4 连接查询（自身连接）



### 3.5 连接查询（多表连接）



## 四、实验总结

在本阶段的实习中，我掌握了基本表的建立、单表查询、分组统计查询、连接查询的使用方法。构建了sqlite3的环境，对数据库编程有了初步的认识。总体来说，实验一内容是比较简单的。

## 五、问题解决

### 5.1 问题一

创建基本表的过程中出现报错Error: table course already exists

原因是该表在之前已经被创建，无需再次创建。

# 实验二 实验报告

## 实验目的与要求

(1)掌握SQL嵌套查询和集合查询等各种高级查询的设计方法等。

(2)能按照SQL程序设计规范写出具体的SQL查询语句，并调试通过。

(3)支撑课程目标②。

## 实验代码

### 2.1 带有IN谓词的子查询

1. select Sno,Sname,Sdept
2. from student
3. where Sdept IN(select Sdept
4. from student where Sname=’刘晨’);

### 2.2 带有比较运算符的子查询

1. select Sno,Sname,Sdept
2. from student
3. where Sdept=(select Sdept from student where Sname=’刘晨’);

### 2.3 带有ANY（SOME）或ALL谓词的子查询

1. select Sname,Sage
2. from student
3. where Sage <(select MIN(Sage) from student where Sdept='cs')
4. AND Sdept <>'cs';

### 2.4 带有EXISTS的子查询

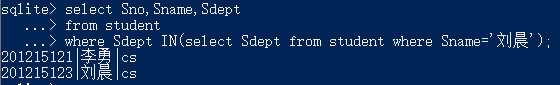
1. select Sname
2. from student
3. where exists(select \* from sc where Sno=student.Sno AND Cno='1');

### 2.5集合查询

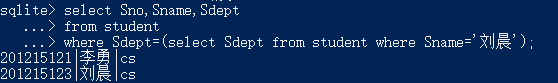
1. select Sno
2. from sc
3. where Cno='1'
4. UNION
5. select Sno
6. from sc
7. where Cno='2';

## 运行结果

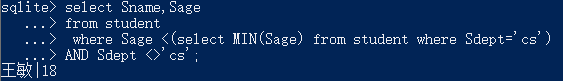
### 3.1 带有IN谓词的子查询



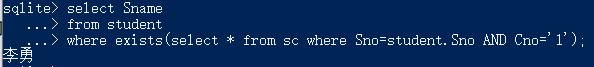
### 3.2 带有比较运算符的子查询



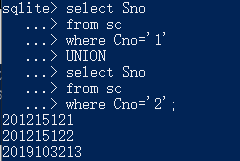
### 3.3 带有ANY（SOME）或ALL谓词的子查询



### 3.4 带有EXISTS的子查询



### 3.5 集合查询



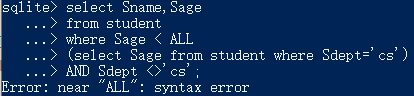
## 实验总结

在本阶段的实验中，我掌握了带有IN的子查询、带有比较运算符的子查询、带有ANY谓词的子查询、带有exists的子查询以及集合查询。在实验过程中，我发现了SQlite中部分SQL语句的用法与我们在课本上所学的SQL语句是不一样的。

## 问题解决

### 5.1 问题一

使用含有ALL的查询时，程序报错，见下图。



原因是ALL的语法再SQlite中与一般的SQL语句有所不同，我们可以用<MIN()等类似的查询方式代替ALL在SQL语句中的作用。

# 实验三 实验报告

## 实验目的与要求

(1)熟悉数据库的数据更新操作，能够使用SQL语句对数据库进行数据的插入、修改、删除。

(2)理解和掌握insert、update和delete语法结构的各个组成成分，结合嵌套SQL子查询，分别设计几种不同形式的插入、修改和删除数据的语句，并调试通过。

(3)支撑课程目标②④。

## 二、实验代码

### 2.1 插入一个元组

1. insert
2. into sc(Sno,Cno)
3. values('201215128','1');

### 2.2 插入子查询结果

1. create table Dept\_age
2. (Sdept CHAR(15),
3. Avg\_age smallint);
4. insert
5. into Dept\_age(Sdept,Avg\_age)
6. select Sdept,AVG(Sage)
7. from student
8. group by Sdept;

### 2.3 修改某元组的值

1. update student
2. set Sage=22
3. where Sno='201215121';

### 2.4 修改多个元组的值

1. update student
2. set Sage=Sage+1;
3. select \* from student;

### 2.5 带子查询的修改

1. update sc
2. set grade=0
3. where Sno IN
4. (select Sno
5. from student
6. where Sdept='cs');

### 2.6 删除多个元组的值

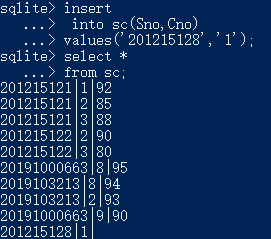
1. delete
2. from sc;

### 2.7 带子查询的删除

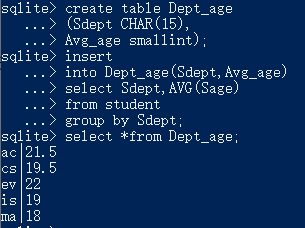
1. delete
2. from sc
3. where Sno IN
4. (select Sno
5. from student
6. where Sdept='cs');

## 三、运行结果

### 3.1 插入一个元组



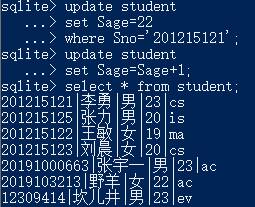
### 3.2 插入子查询结果



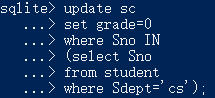
### 3.3 修改某元组的值



### 3.4 修改多个元组的值



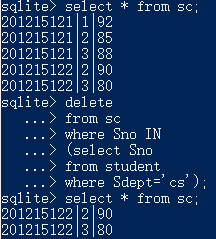
### 3.5 带子查询的修改



### 3.6 删除多个元组的值



### 3.7 带子查询的删除



## 四、实验总结

在该部分实验中，我掌握了insert、update、delete等语句的用法，对SQL语句的理解有了进一步的提升。在使用insert、update、delete等语句时需要时刻注意，不能破坏实体完整性、参照完整性、用户定义完整性。

## 五、问题解决

### 5.1 问题一

问题描述：在SQLite中使用inse、update、delete语句时，系统没有自动进行完整性定义检查。

解决方法：可以通过自定义用户定义完整性、设置断言等方法来解决这一问题。

# 实验四 实验报告

## 实验目的与要求

(1)熟悉SQL语言有关视图的操作，能够熟练使用SQL语句来创建需要的视图，定义数据库外模式，并能使用所创建的视图实现数据管理

(2)理解掌握视图消解执行原理，掌握可更新视图和不可更新视图的区别

(3)支撑课程目标②④

## 二、实验代码

### 2.1 创建视图

1. create view is\_student
2. as
3. select Sno,Sname,Sage
4. from student
5. where Sdept='is';

### 2.2 删除视图

1. drop view IS\_S1;

### 2.3 查询视图

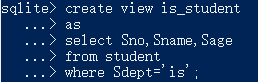
1. create view test1
2. as
3. select sno,sname,sage
4. from student
5. where sdept='ac';

### 2.4 对非行列子集视图的查询

1. create view test3(sno,gavg)
2. as
3. elect sno,AVG(grade)
4. from sc
5. group by sno;
6. select \*
7. from test3
8. where gavg>80;

## 三、运行结果

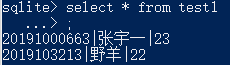
### 3.1 创建视图



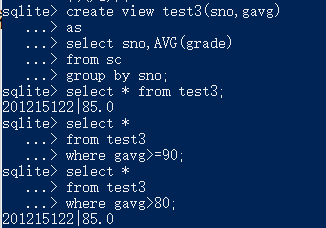
### 3.2 删除视图



### 3.3 查询视图



### 3.4 对非行列子集视图的查询



## 四、实验总结

在该部分实验中，我掌握了视图创建、更新视图、视图消解的方法。在实验过程中我发现在sqlite中部分非行列子集视图也是可以正常消解的。

在创建视图时，子查询中的属性值出错并不会立马报错，而是在视图被调用的时候才会出现报错。这说明了视图建立后并不会立马被执行，而是在被调用的时候才会执行。进而说明了视图只是一个虚表，并不是实际存在的表。

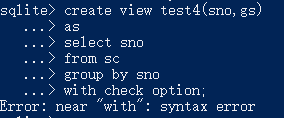
## 五、问题解决

### 5.1问题一

问题描述：无法使用with check option语句，一使用就报错。

问题原因：sqlite版本过低，所以无法使用。

解决方法：将sqlite的版本升级到3.8.3以上的版本。

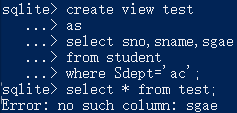


### 5.2 问题二

问题描述：查询视图时出现报错。

问题原因：在创建视图时，属性名输入错误。

解决方法：将视图删除，重新创建。



# 实验五 实验报告

## 实验目的与要求

(1)通过实验，掌握数据库设计基本方法，熟练应用高级语言结合SQL语言完成与信息系统相关的数据管理

(2)支撑课程目标①~④

## 二、实验代码

### 2.1 系统主界面(MainWindow)

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

if(flag\_log=="0"){

ui->find\_stu\_all->setEnabled(false);

ui->find\_stu->setEnabled(false);

ui->find\_stu\_all\_2->setEnabled(false);

ui->find\_sc->setEnabled(false);

ui->find\_sc\_2->setEnabled(false);

ui->find\_sc\_3->setEnabled(false);

ui->find\_sc\_4->setEnabled(false);

ui->grade\_sel->setEnabled(false);

}

sql\_connect();

//find->show();

//connect(find,SIGNAL(sendDataa(int)),this,SLOT(processDataa(int)));

connect(find,&Find::send\_sno,this,&MainWindow::rec\_sno);

connect(find,&Find::send\_sname,this,&MainWindow::rec\_sname);

connect(find\_sc,&Find\_sc::send\_sno,this,&MainWindow::rec\_sno\_sc);

connect(find\_sc,&Find\_sc::send\_sname,this,&MainWindow::rec\_sname\_sc);

connect(stuset,&stu\_set::new\_student,this,&MainWindow::rec\_new\_stu);

connect(stuset,&stu\_set::del\_student,this,&MainWindow::rec\_del\_stu);

connect(stuset,&stu\_set::rename\_stu,this,&MainWindow::rec\_rename\_stu);

connect(stuset,&stu\_set::reage\_stu,this,&MainWindow::rec\_reage\_stu);

connect(stuset,&stu\_set::resex\_stu,this,&MainWindow::rec\_resex\_stu);

connect(stuset,&stu\_set::redept\_stu,this,&MainWindow::rec\_redept\_stu);

connect(coset,&course\_set::new\_course,this,&MainWindow::rec\_new\_course);

connect(coset,&course\_set::del\_course,this,&MainWindow::rec\_del\_course);

connect(coset,&course\_set::rcname\_course,this,&MainWindow::rec\_recname\_course);

connect(coset,&course\_set::rcpno\_course,this,&MainWindow::rec\_recpno\_course);

connect(coset,&course\_set::rcredit\_course,this,&MainWindow::rec\_recredit\_course);

connect(scset,&sc\_set::new\_sc,this,&MainWindow::rec\_new\_sc);

connect(scset,&sc\_set::del\_sc,this,&MainWindow::rec\_del\_sc);

connect(scset,&sc\_set::cha\_sc,this,&MainWindow::rec\_cha\_sc);

connect(gsel,&grade\_sel::stu\_sel,this,&MainWindow::rec\_stu\_sel);

connect(gsel,&grade\_sel::course\_sel,this,&MainWindow::rec\_course\_sel);

connect(ld,&load::man\_mode,this,&MainWindow::rec\_man\_mode);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

bool MainWindow::**sql\_connect**()

{

QSqlDatabase db = QSqlDatabase::addDatabase("QSQLITE");

db.setDatabaseName("student\_zyy.db");

if (!db.open()) {

QMessageBox::critical(nullptr, QObject::tr("Cannot open database"),

QObject::tr("Unable to establish a database connection.\n"

"This example needs SQLite support. Please read "

"the Qt SQL driver documentation for information how "

"to build it.\n\n"

"Click Cancel to exit."), QMessageBox::Cancel);

return false;

}

return true;

}

void MainWindow::**rec\_sno**(QString m\_sno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("select \* from student");

qr.prepare("select \* from student where sno = ?");

qr.addBindValue(m\_sno);//有点类似格式化字符串的作用

qr.exec();

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_sname**(QString m\_sname)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("select \* from student");

qr.prepare("select \* from student where sname = ?");

qr.addBindValue(m\_sname);//有点类似格式化字符串的作用

qr.exec();

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_sno\_sc**(QString sno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("select \* from student");

qr.prepare("select sc.sno,sname,cname,grade from sc,student,course where sc.sno=student.sno AND sc.sno=? AND course.cno=sc.cno");

qr.addBindValue(sno);//有点类似格式化字符串的作用

qr.exec();

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_sname\_sc**(QString sname)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("select \* from student");

qr.prepare("select sc.sno,sname,cname,grade from sc,student,course where sc.sno=student.sno AND sname=? AND course.cno=sc.cno");

qr.addBindValue(sname);//有点类似格式化字符串的作用

qr.exec();

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_new\_stu**(QString sno, QString sname, QString ssex, QString sage, QString sdept)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("insert into student(Sno,Sname,Ssex,Sage,Sdept) values('20191000663','张宇一','男',22,'AC')");

qr.prepare("insert into student(Sno,Sname,Ssex,Sage,Sdept) values(:a,:b,:c,:d,:e)");

qr.bindValue(":a",sno);//有点类似格式化字符串的作用

qr.bindValue(":b",sname);

qr.bindValue(":c",ssex);

qr.bindValue(":d",sage);

qr.bindValue(":e",sdept);

qr.exec();

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_del\_stu**(QString sno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("insert into student(Sno,Sname,Ssex,Sage,Sdept) values('20191000663','张宇一','男',22,'AC')");

qr.prepare("delete from student where sno = ?");

qr.addBindValue(sno);//有点类似格式化字符串的作用

qr.exec();

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_rename\_stu**(QString sno, QString sname)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("insert into student(Sno,Sname,Ssex,Sage,Sdept) values('20191000663','张宇一','男',22,'AC')");

qr.prepare("update student set Sname = :b where Sno = :a");

qr.bindValue(":a",sno);//有点类似格式化字符串的作用

qr.bindValue(":b",sname);

qr.exec();

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_reage\_stu**(QString sno, QString sage)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update student set Sage = :b where Sno = :a");

qr.bindValue(":a",sno);//有点类似格式化字符串的作用

qr.bindValue(":b",sage);

qr.exec();

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_resex\_stu**(QString sno, QString sex)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update student set Ssex = :b where Sno = :a");

qr.bindValue(":a",sno);//有点类似格式化字符串的作用

qr.bindValue(":b",sex);

qr.exec();

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_redept\_stu**(QString sno, QString sdept)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update student set Sdept = :b where Sno = :a");

qr.bindValue(":a",sno);//有点类似格式化字符串的作用

qr.bindValue(":b",sdept);

qr.exec();

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_new\_course**(QString cno, QString cname, QString cpno, QString credit)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("select \* from course");

qr.prepare("insert into course(Cno,Cname,Cpno,Ccredit) values(:a,:b,:c,:d)");

qr.bindValue(":a",cno);//有点类似格式化字符串的作用

qr.bindValue(":b",cname);

qr.bindValue(":c",cpno);

qr.bindValue(":d",credit);

qr.exec();

qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_del\_course**(QString cno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("delete from course where Cno = ?");

qr.addBindValue(cno);//有点类似格式化字符串的作用

qr.exec();

qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_recname\_course**(QString cno, QString cname)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update course set Cname = :b where Cno = :a");

qr.bindValue(":a",cno);//有点类似格式化字符串的作用

qr.bindValue(":b",cname);

qr.exec();

qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_recpno\_course**(QString cno, QString cpno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update course set Cpno = :b where Cno = :a");

qr.bindValue(":a",cno);//有点类似格式化字符串的作用

qr.bindValue(":b",cpno);

qr.exec();

qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_recredit\_course**(QString cno, QString credit)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update course set Ccredit = :b where Cno = :a");

qr.bindValue(":a",cno);//有点类似格式化字符串的作用

qr.bindValue(":b",credit);

qr.exec();

qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_new\_sc**(QString sno, QString cno, QString grade)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

//qr.exec("select \* from course");

qr.prepare("insert into sc(Sno,Cno,grade) values(:a,:b,:c)");

qr.bindValue(":a",sno);//有点类似格式化字符串的作用

qr.bindValue(":b",cno);

qr.bindValue(":c",grade);

qr.exec();

qr.exec("select Sname,sc.Sno,sc.Cno,Cname,grade from student,sc,course where sc.Sno=student.Sno AND sc.Cno=course.Cno");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_del\_sc**(QString sno, QString cno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("delete from sc where Cno = :a AND Sno = :b");

qr.bindValue(":a",cno);//有点类似格式化字符串的作用

qr.bindValue(":b",sno);

qr.exec();

qr.exec("select Sname,sc.Sno,sc.Cno,Cname,grade from student,sc,course where sc.Sno=student.Sno AND sc.Cno=course.Cno");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_cha\_sc**(QString sno, QString cno, QString grade)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("update sc set grade = :b where Cno = :a AND Sno = :c");

qr.bindValue(":a",cno);//有点类似格式化字符串的作用

qr.bindValue(":b",grade);

qr.bindValue(":c",sno);

qr.exec();

qr.exec("select Sname,sc.Sno,sc.Cno,Cname,grade from student,sc,course where sc.Sno=student.Sno AND sc.Cno=course.Cno");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_stu\_sel**(QString sno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("select student.Sno,Sname,AVG(grade) 平均成绩,MAX(grade) 最高分,MIN(grade) 最低分 from sc,student,course where sc.Sno=student.Sno AND sc.Cno=course.Cno AND sc.Sno = ? group by sc.Sno");

qr.addBindValue(sno);//有点类似格式化字符串的作用

qr.exec();

//qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_course\_sel**(QString cno)

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.prepare("select sc.Cno,Cname,AVG(grade) 平均成绩,MAX(grade) 最高分,MIN(grade) 最低分 from sc,student,course where sc.Sno=student.Sno AND sc.Cno=course.Cno AND sc.Cno = ? group by sc.Cno");

qr.addBindValue(cno);//有点类似格式化字符串的作用

qr.exec();

//qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**rec\_man\_mode**(QString)

{

if(flag\_log=="0"){

ui->find\_stu\_all->setEnabled(false);

ui->find\_stu->setEnabled(false);

ui->find\_stu\_all\_2->setEnabled(false);

ui->find\_sc->setEnabled(false);

ui->find\_sc\_2->setEnabled(false);

ui->find\_sc\_3->setEnabled(false);

ui->find\_sc\_4->setEnabled(false);

ui->grade\_sel->setEnabled(false);

}

else if(flag\_log=="1"){

ui->find\_stu\_all->setEnabled(true);

ui->find\_stu->setEnabled(true);

ui->find\_stu\_all\_2->setEnabled(true);

ui->find\_sc->setEnabled(true);

ui->find\_sc\_2->setEnabled(true);

ui->find\_sc\_3->setEnabled(true);

ui->find\_sc\_4->setEnabled(true);

ui->grade\_sel->setEnabled(true);

ld->close();

}

else if(flag\_log=="2"){

ui->find\_stu\_all->setEnabled(false);

ui->find\_stu->setEnabled(true);

ui->find\_stu\_all\_2->setEnabled(false);

ui->find\_sc->setEnabled(true);

ui->find\_sc\_2->setEnabled(true);

ui->find\_sc\_3->setEnabled(false);

ui->find\_sc\_4->setEnabled(false);

ui->grade\_sel->setEnabled(true);

ld->close();

}

}

void MainWindow::**on\_find\_stu\_clicked**()

{

find->show();

}

void MainWindow::**on\_find\_stu\_all\_clicked**()

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.exec("select \* from student");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**on\_find\_stu\_all\_2\_clicked**()

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.exec("select sname,cname,sc.cno,grade from sc,student,course where student.sno=sc.sno AND course.cno=sc.cno");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

}

void MainWindow::**on\_find\_sc\_clicked**()

{

find\_sc->show();

}

void MainWindow::**on\_find\_sc\_2\_clicked**()

{

stuset->show();

}

void MainWindow::**on\_find\_sc\_3\_clicked**()

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.exec("select \* from course");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

coset->show();

}

void MainWindow::**on\_find\_sc\_4\_clicked**()

{

studentqrModel=new QSqlQueryModel(this);

sql\_connect();

QSqlQuery qr;

qr.exec("select Sname,sc.Sno,sc.Cno,Cname,grade from student,sc,course where sc.Sno=student.Sno AND sc.Cno=course.Cno");

studentqrModel->setQuery(qr);

ui->tableView\_manage->*setModel*(*studentqrModel*);

scset->show();

}

void MainWindow::**on\_grade\_sel\_clicked**()

{

gsel->show();

}

void MainWindow::**on\_loadsys\_clicked**()

{

ld->show();

}

void MainWindow::**on\_ex\_clicked**()

{

ui->find\_stu\_all->setEnabled(false);

ui->find\_stu->setEnabled(false);

ui->find\_stu\_all\_2->setEnabled(false);

ui->find\_sc->setEnabled(false);

ui->find\_sc\_2->setEnabled(false);

ui->find\_sc\_3->setEnabled(false);

ui->find\_sc\_4->setEnabled(false);

ui->grade\_sel->setEnabled(false);

flag\_log="0";

ld->show();

}

### 2.2 登录界面(load)

#include "load.h"

#include "ui\_load.h"

#include <QMessageBox>

QString flag\_log="0";

load::**load**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::load)

{

ui->setupUi(this);

}

load::~***load***()

{

delete ui;

}

void load::**on\_pushButton\_clicked**()

{

QString uname;

QString password;

uname=ui->lineEdit->text();

password=ui->lineEdit\_2->text();

if(uname == "root" && password =="123"){

flag\_log="1";

}

else if(uname == "stu" && password =="123"){

flag\_log="2";

}

else{

flag\_log="0";

log->show();

}

emit man\_mode(flag\_log);

ui->lineEdit->clear();

ui->lineEdit\_2->clear();

}

### 2.3 学生信息查询界面

#include "find.h"

#include "ui\_find.h"

Find::**Find**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::Find)

{

ui->setupUi(this);

}

Find::~***Find***()

{

delete ui;

}

void Find::**on\_find\_sure\_clicked**()

{

emit send\_sno(ui->input\_sno->text());

ui->input\_sno->clear();

}

void Find::**on\_find\_sure\_2\_clicked**()

{

emit send\_sname(ui->input\_sname->text());

ui->input\_sname->clear();

}

### 2.4 选课信息查询界面

#include "find\_sc.h"

#include "ui\_find\_sc.h"

Find\_sc::**Find\_sc**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::Find\_sc)

{

ui->setupUi(this);

}

Find\_sc::~***Find\_sc***()

{

delete ui;

}

void Find\_sc::**on\_find\_sure\_3\_clicked**()

{

emit send\_sno(ui->input\_sno\_1->text());

ui->input\_sno\_1->clear();

}

void Find\_sc::**on\_find\_sure\_4\_clicked**()

{

emit send\_sname(ui->input\_sname\_1->text());

ui->input\_sname\_1->clear();

}

### 2.5 学生操作界面

#include "stu\_set.h"

#include "ui\_stu\_set.h"

stu\_set::**stu\_set**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::stu\_set)

{

ui->setupUi(this);

}

stu\_set::~***stu\_set***()

{

delete ui;

}

void stu\_set::**on\_new\_stu\_clicked**()

{

QString Sno;

QString Sname;

QString Ssex;

QString Sage;

QString Sdept;

Sno=ui->lineEdit->text();

Sname=ui->lineEdit\_2->text();

Ssex=ui->lineEdit\_3->text();

Sage=ui->lineEdit\_4->text();

Sdept=ui->lineEdit\_5->text();

emit new\_student(Sno,Sname,Ssex,Sage,Sdept);

ui->lineEdit->clear();

ui->lineEdit\_2->clear();

ui->lineEdit\_3->clear();

ui->lineEdit\_4->clear();

ui->lineEdit\_5->clear();

}

void stu\_set::**on\_pushButton\_clicked**()

{

emit del\_student(ui->lineEdit\_6->text());

ui->lineEdit\_6->clear();

}

void stu\_set::**on\_rename\_clicked**()

{

QString Sno;

QString Sname;

Sno=ui->lineEdit\_7->text();

Sname=ui->lineEdit\_8->text();

emit rename\_stu(Sno,Sname);

ui->lineEdit\_8->clear();

}

void stu\_set::**on\_reage\_clicked**()

{

QString Sno;

QString Sage;

Sno=ui->lineEdit\_7->text();

Sage=ui->lineEdit\_8->text();

emit reage\_stu(Sno,Sage);

ui->lineEdit\_8->clear();

}

void stu\_set::**on\_resex\_clicked**()

{

QString Sno;

QString Ssex;

Sno=ui->lineEdit\_7->text();

Ssex=ui->lineEdit\_8->text();

emit resex\_stu(Sno,Ssex);

ui->lineEdit\_8->clear();

}

void stu\_set::**on\_redept\_clicked**()

{

QString Sno;

QString Sdept;

Sno=ui->lineEdit\_7->text();

Sdept=ui->lineEdit\_8->text();

emit redept\_stu(Sno,Sdept);

ui->lineEdit\_8->clear();

}

### 2.6 选课操作界面

#include "sc\_set.h"

#include "ui\_sc\_set.h"

sc\_set::**sc\_set**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::sc\_set)

{

ui->setupUi(this);

}

sc\_set::~***sc\_set***()

{

delete ui;

}

void sc\_set::**on\_new\_sc\_clicked**()

{

QString sno;

QString cno;

QString grade;

sno=ui->lineEdit->text();

cno=ui->lineEdit\_2->text();

grade=ui->lineEdit\_3->text();

emit new\_sc(sno,cno,grade);

ui->lineEdit->clear();

ui->lineEdit\_2->clear();

ui->lineEdit\_3->clear();

}

void sc\_set::**on\_del\_sc\_clicked**()

{

QString sno;

QString cno;

sno=ui->lineEdit\_4->text();

cno=ui->lineEdit\_5->text();

emit del\_sc(sno,cno);

ui->lineEdit\_4->clear();

ui->lineEdit\_5->clear();

}

void sc\_set::**on\_cha\_sc\_clicked**()

{

QString sno;

QString cno;

QString grade;

sno=ui->lineEdit\_6->text();

cno=ui->lineEdit\_7->text();

grade=ui->lineEdit\_8->text();

emit cha\_sc(sno,cno,grade);//发送

ui->lineEdit\_6->clear();

ui->lineEdit\_7->clear();

ui->lineEdit\_8->clear();

}

### 2.7 成绩统计界面

#include "grade\_sel.h"

#include "ui\_grade\_sel.h"

grade\_sel::**grade\_sel**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::grade\_sel)

{

ui->setupUi(this);

}

grade\_sel::~***grade\_sel***()

{

delete ui;

}

void grade\_sel::**on\_stu\_sel\_clicked**()

{

emit stu\_sel(ui->lineEdit->text());

ui->lineEdit->clear();

}

void grade\_sel::**on\_coure\_sel\_clicked**()

{

emit course\_sel(ui->lineEdit\_2->text());

ui->lineEdit\_2->clear();

}

## 三、运行结果

### 3.1 系统主界面(MainWindow)

系统主界面如下图所示。

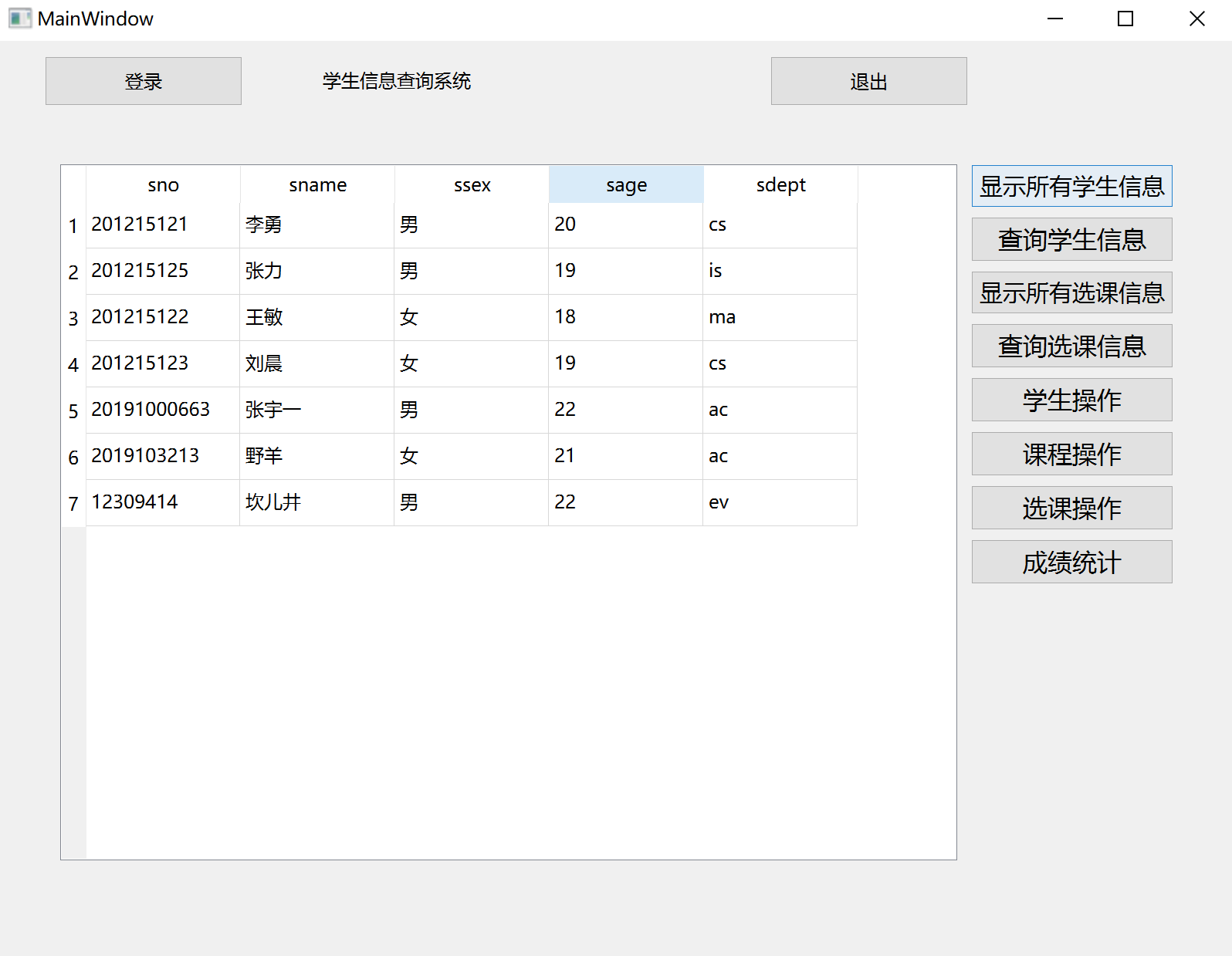


图 1 系统主界面

该界面为教师登录所看到的界面，教师有该系统的最高权限。学生登录系统后会有部分按键变暗无法使用。

### 3.2 登录界面(load)

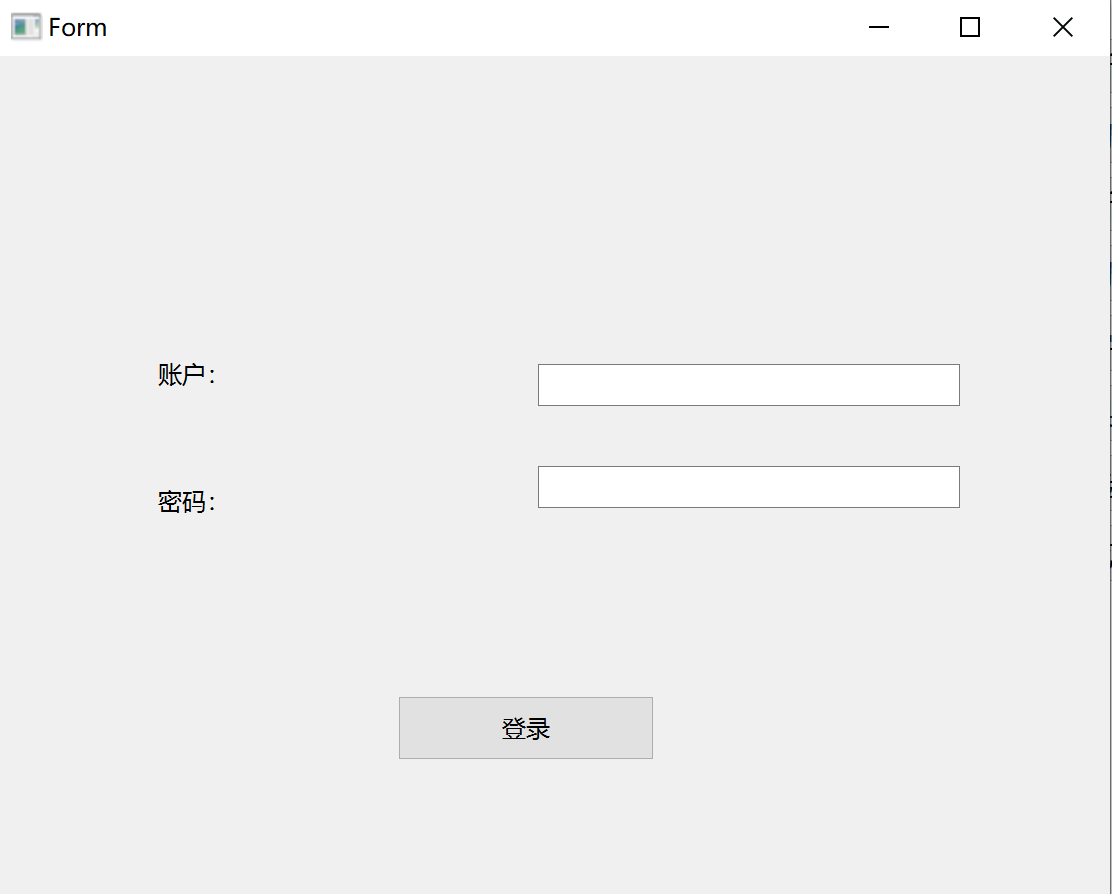


图 2 登录界面

一共有两类账号，一类是教师账号，一类是学生账号，两类账号登录会有不同的界面。

### 3.3 学生信息查询界面

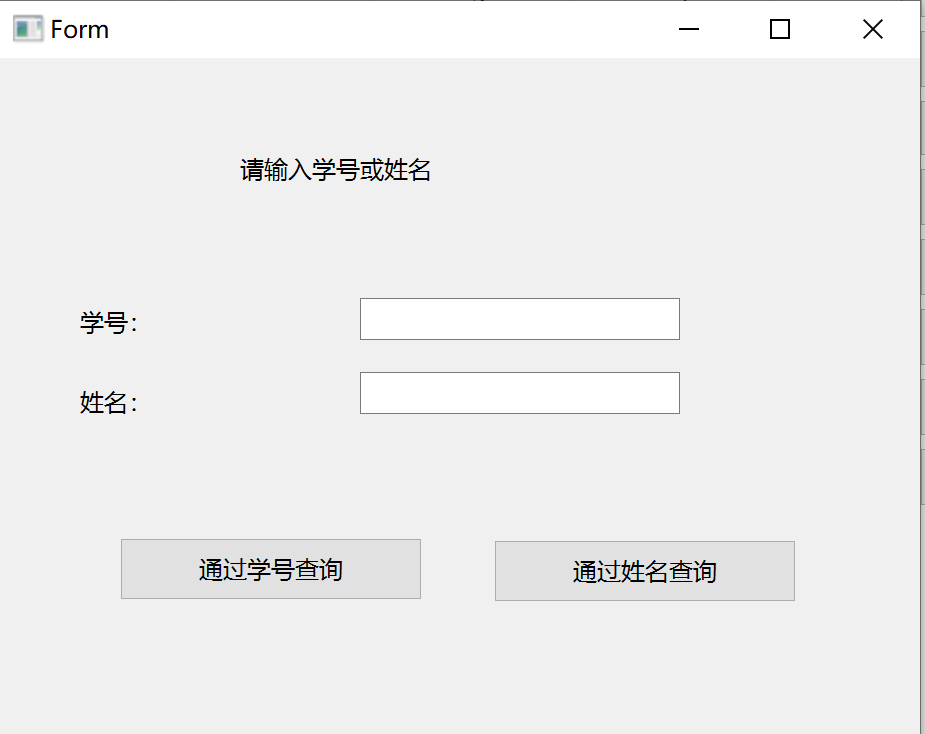


图 3 学生信息查询界面

可通过学号或姓名查询。

### 3.4 选课信息查询界面

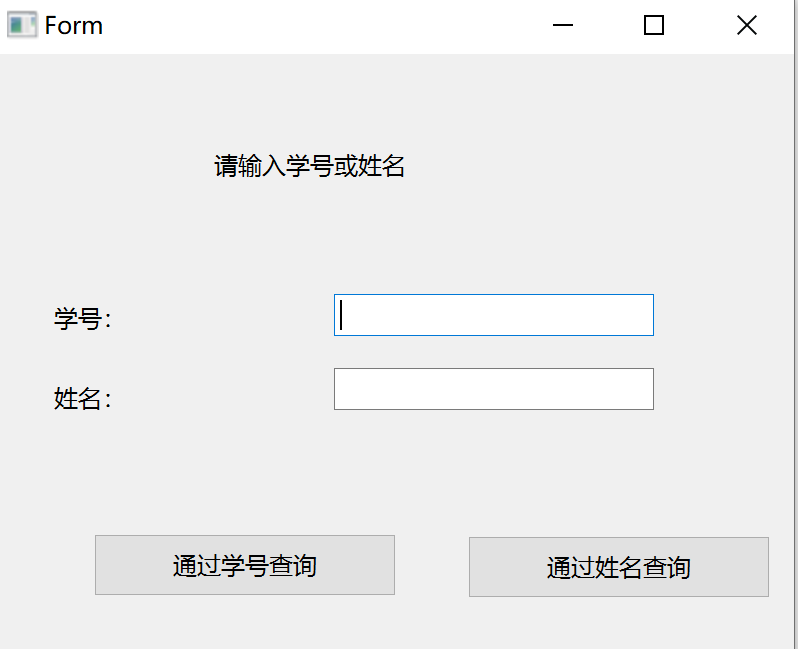


图 4 选课信息查询界面

可通过学号或姓名查询。

### 3.5 学生操作界面



图 5 学生操作新增界面



图 6 学生操作删除界面



图 7 学生操作修改界面

学生界面有新增、删除、修改三个字界面。

### 3.6 选课操作界面



图 8 选课界面新增界面



图 9 选课界面删除界面



图 10 选课界面修改界面

选课界面有新增、删除、修改三个字界面

### 3.7 成绩统计界面



图 11 成绩统计界面学生个人成绩统计



图 12 成绩统计界面课程成绩统计

3.8课程操作界面



图 13 课程操作界面增加部分



图 14 课程操作界面删除部分



图 15 课程操作界面修改部分

## 四、实验总结

我认为本部分是本次数据库实验中最重要的一部分，在这部分实验中，我利用Qt和SQLite构建了自己的学生信息管理系统。这部分的实验安排很好的联系了另外的科目，在学习数据库的同时温习了Qt的使用，使我受益良多。

在完成实验的过程中我也遇到了非常多的问题，例如无法正常连接数据库、connect无法正常连接信号与槽函数、子ui无法和主ui正常连接、字符串无法正常拼接等。通过解决各种问题、调试各类bug使我对SQL语言有了更深入的了解，同时编程能力和调试能力也有了较大的提升。

由于期末时间紧凑，无法全身心的投入本次实验，在本次实验中所写的代码冗余度大、可读性差。在寒假假期时间充裕时，我会将界面再次优化、尝试新的数据库模型、尝试使用全局变量来优化算法等。

## 五、问题解决

### 5.1 问题一

问题描述：connect函数无法正常连接

问题原因：Qt版本不兼容

解决方法：信号与槽通过connect函数连接有多种写法，其中最老的一种写法如下图所示。这种写法的优点是清楚明了，但是当信号函数和槽函数出现错误时并不会报错，这对于调试程序非常不友好，虽说市面上主流的Qt教程均采用这种写法，但是现在部分版本的Qt已经不再兼容这种写法，所以在使用connect函数是建议采用以下这种新写法。



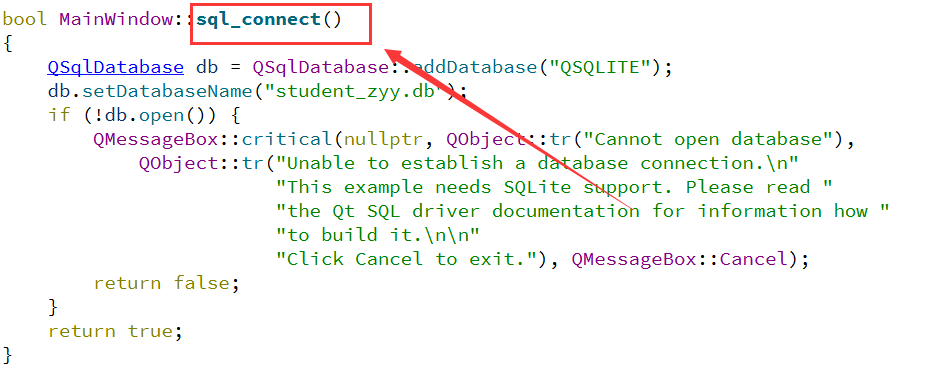
另外需要注意的是，connect函数中信号函数和槽函数不需要带形参，这也是使用connect函数出错的主要原因之一。

### 5.2 问题二

问题描述：无法正常连接数据库

问题原因：连接函数与Qt内置函数connect重名导致不兼容

解决方法：更改连接函数的名称即可，如下图将函数命名为sql\_connect()即可。



### 5.3 问题三

问题描述：需要拼接多个字符串时addBindValue()函数无法正常使用

解决方法：拼接多个字符串理论上是可以通过在prepare()函数后不断加入addBindValue()函数来实现的，但是在实际操作时并不能正常使用。解决方法是可以通过bindValue()将多个字符串拼接起来，如下图所示。

