数据库应用系统

综合设计报告

**题目名称： 教务系统（学生端）**

**2020年7月10日**

# 项目背景

包括题目内容、数据要求，并给出参照设计系统的连接地址；

题目内容：本次数据库大作业模拟了ytu教务系统学生端，并完成了个人管理，选课管理，课表管理，考务管理五大方面。

数据要求：

1. 学生的个人信息

学号、姓名、个人登录教务系统密码、电话、邮箱、个人主页、个人简介

1. 课程的信息

课程号 课程名 学分 学时 授课老师 上课时间 上课地点 考试时间 考试地点等等

1. 教师的信息

教师序号 教师姓名 教师电话 所授课程等

# 2. 系统需求分析

本产品实现了一个基于mySQL数据库下的学生教务管理系统,而本需求分析报告的编写目的是为了提供一个由用户(学生)和开发者双方共同确定的开发系统的业务需求目标，并对本系统所要实现的软件功能做一个全面的规格描述。  
  
 同时，在用户业务需求的基础上，经过需求分析和数据整理，以向整个开发期提供关于软件系统的业务和数据的技术信息和整体描述，做为软件开发的技术基础，也作为系统设计、实现的目标和测试以及维护阶段的依据。  
本软件需求分析报告的适用读者为:软件用户(学生)、软件需求分析人员、软件设计及开发者和相关的测试人员。

本系统为了实现学校在校学生基本信息资源的查询、统计、添加、删除、维护与管理，方便用户的管理与需求。

Ytu教务系统（学生端）包括

1. 查询和修改个人信息，如修改密码，电话，email等
2. 进行选课管理，包括查看所有课程信息，选择课程，取消选课，查看已选课程
3. 查看本学期考试安排

教务系统的基本规定：每个学生可以选修多门课程，每门课程可以有多个教师，课程的上课时间固定，老师的教学地点固定

# 3. 总体设计

本系统分为四大模块：

1. 登录模块

学生填写登录所需的账号（学号），密码，无误后即可登录

1. 个人信息模块

学生可以修改个人信息

1. 选课模块

学生可以查看所有课程信息，选择课程，取消选课，查看已选课程

1. 考务查询模块

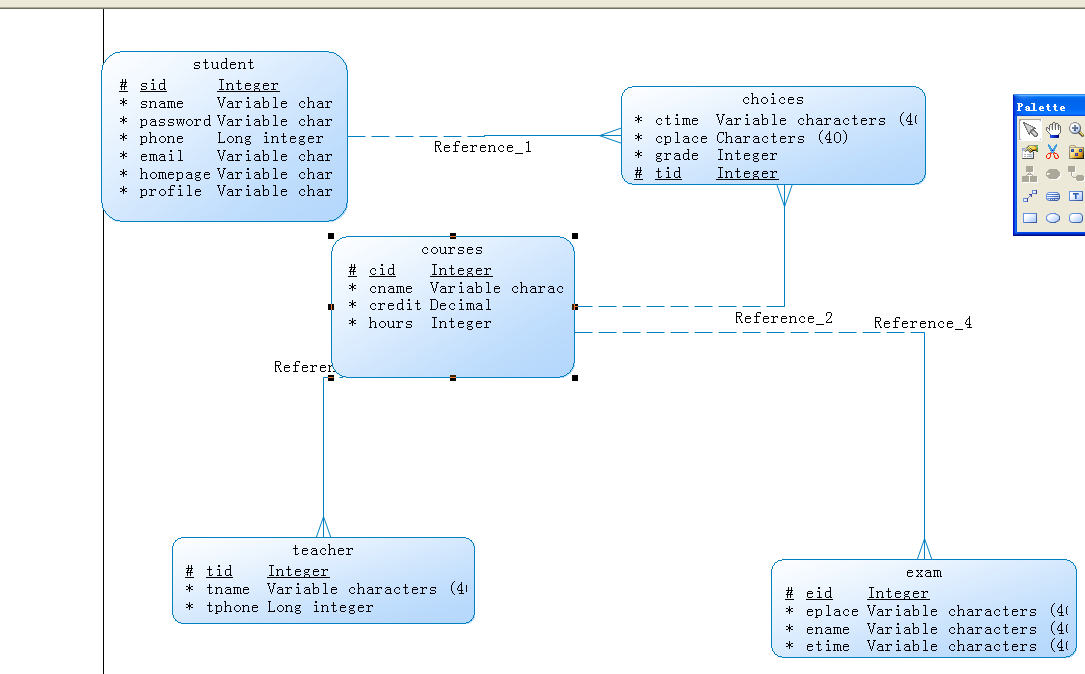
学生可以查询考试信息

# 4. 数据库设计

能够根据系统功能需求设计出合理的和结构清晰的数据库并创建。一般不得少于5个表，需要指明所满足的范式并给出理由。概念结构设计合理、逻辑结构设计、物理结构能够通过工具自动生成且合理。该部分主要体现课程目标2的要求。

## 4.1概念结构设计

Powerdesigner设计的概念模型



上述模型为初次设计结构，最终版本在choices表中删去了除sid cid tid以外所有属性。

## 4.2逻辑结构设计

逻辑结构设计采用下表形式：

1. 学生信息表

在该表中保存了学生的基本资料

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 字段长度 | 约束 | 备注 |
| Sid | Int | 8 | 主键 | 学号 |
| Sname | Varchar | 40 | Not null | 学生姓名 |
| Password | Varchar | 40 | Not null | 登录密码 |
| Sphone | Bigint | 11 |  | 学生电话 |
| Email | Varhcar | 40 |  | 学生邮箱 |
| homepage | Varchar | 1024 |  | 个人主页 |
| profile | Varchar | 1024 |  | 个人简介 |

1. 课程信息表

在该表中保存了课程的基本信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 字段长度 | 约束 | 备注 |
| Cid | Int | 4 | 主键 | 课程号 |
| Cname | Varchar | 40 | Not null | 课程名称 |
| Credit | Decimal(2,1) | 3 | Not null | 学分 |
| Hours | int | 2 | Not null | 学时 |
| Ctime | Varchar | 40 | Not null | 上课时间 |

1. 学生选课表

在该表中保存了学生所选课程的基本信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 字段长度 | 约束 | 备注 |
| Cid | Int | 4 | 外键 | 课程号 |
| Sid | Int | 8 | 外键 | 学号 |
| Tid | Int | 8 | 外键 | 教师序号 |

其中课程号和教师序号共同组成主键

1. 考试安排表

在该表中保存了学生所选课程的基本信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 字段长度 | 约束 | 备注 |
| Eid | Int | 8 | 主键 | 考试序号 |
| Cid | Int | 4 | 外键 | 课程号 |
| Eplace | Varchar | 40 | Not null | 考试地点 |
| Ename | Varchar | 40 | Not null | 考试名称 |
| Etime | Varchar | 40 | Not null | 时间 |

1. 教师信息表

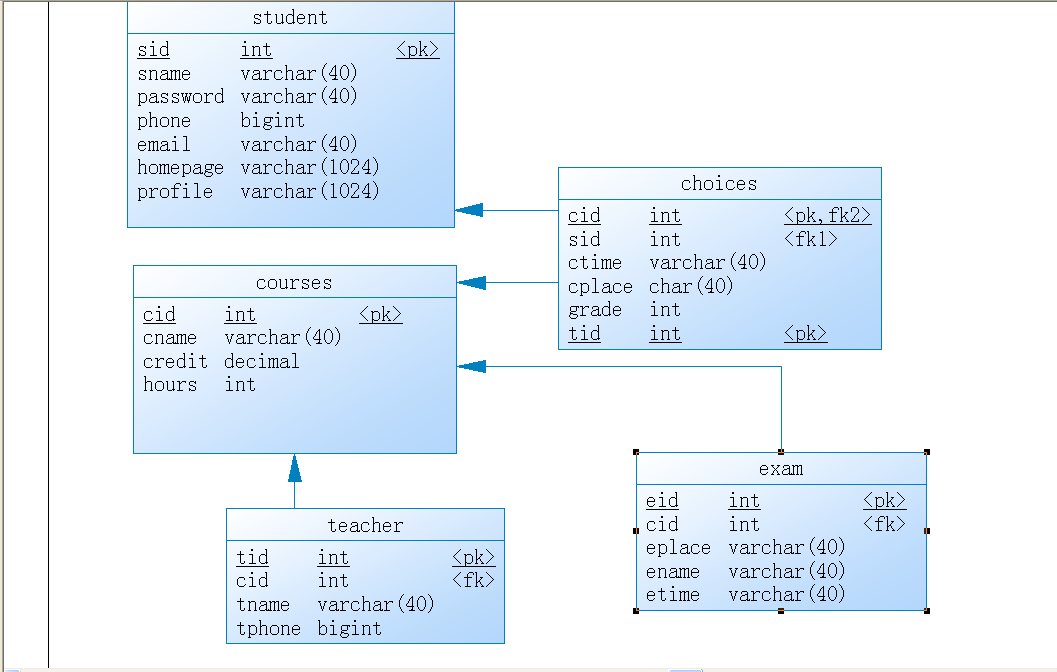
在该表中保存了教师的信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 字段长度 | 约束 | 备注 |
| Tid | Int | 8 | 主键 | 教师序号 |
| Cid | Int | 4 | 外键 | 课程号 |
| Tname | Varchar | 40 | Not null | 教师姓名 |
| Tphone | Long int | 11 |  | 教师电话 |
| Tplace | Varchar | 40 | Not null | 教学地点 |



## 4.3物理结构设计

Powerdesigner设计的物理模型



采用mysql原生物理结构。

聚簇索引，数据文件存放在服务器数据目录下的数据库目录和文件，其中frm文件包含了表结构的描述信息，每一个表都有一个.frm文件在对应的数据库目录下，它与表使用哪种引擎没有关系，myd文件由myisam存储引擎在适当的数据库目录中创建的数据库数据文件和索引文件

# 5. 系统实现及测试结果

## 5.1数据库创建

创建语句及执行结果。（powerdesign创建）

/\*==============================================================\*/

/\* DBMS name: MySQL 5.0 \*/

/\* Created on: 2020-7-10 21:57:19 \*/

/\*==============================================================\*/

drop table if exists choices;

drop table if exists courses;

drop table if exists exam;

drop table if exists student;

drop table if exists teacher;

/\*==============================================================\*/

/\* Table: choices \*/

/\*==============================================================\*/

create table choices

(

cid int not null,

sid int not null,

tid int not null,

primary key (cid, tid)

);

/\*==============================================================\*/

/\* Table: courses \*/

/\*==============================================================\*/

create table courses

(

cid int not null,

cname varchar(40) not null,

credit decimal not null,

hours int not null,

ctime varchar(40) not null,

primary key (cid)

);

/\*==============================================================\*/

/\* Table: exam \*/

/\*==============================================================\*/

create table exam

(

eid int not null,

cid int not null,

eplace varchar(40) not null,

ename varchar(40) not null,

etime varchar(40) not null,

primary key (eid)

);

/\*==============================================================\*/

/\* Table: student \*/

/\*==============================================================\*/

create table student

(

sid int not null,

sname varchar(40) not null,

password varchar(40) not null,

phone bigint not null,

email varchar(40) not null,

homepage varchar(1024) not null,

profile varchar(1024) not null,

primary key (sid)

);

/\*==============================================================\*/

/\* Table: teacher \*/

/\*==============================================================\*/

create table teacher

(

tid int not null,

cid int not null,

tname varchar(40) not null,

tphone bigint not null,

tplace char(40) not null,

primary key (tid)

);

alter table choices add constraint FK\_Reference\_1 foreign key (sid)

references student (sid) on delete restrict on update restrict;

alter table choices add constraint FK\_Reference\_2 foreign key (cid)

references courses (cid) on delete restrict on update restrict;

alter table exam add constraint FK\_Reference\_4 foreign key (cid)

references courses (cid) on delete restrict on update restrict;

alter table teacher add constraint FK\_Reference\_3 foreign key (cid)

references courses (cid) on delete restrict on update restrict;

导入：



## 5.2单一功能的SQL实现

能够根据系统功能需求，自己设计6个或以上不同要求的数据库增删改查等功能，并用SQL实现。功能要求都有一定难度、并能正确实现。

1.根据学号查找改学生全部信息

sql = """  
 select \*  
 from student  
 where sid = '%d'  
""" % sid

2.按学号更新该学生信息（加锁）

lock = """  
 lock tables student write  
 """  
cursor.execute(lock)  
sql = """  
 update student  
 set sname = '%s',password = '%s',phone = '%ld',email='%s',homepage='%s',profile='%s'  
 where sid = '%d'  
""" % (student[1], student[2], int(student[3]), student[4], student[5], student[6], int(student[0]))  
try:  
 cursor.execute(sql)  
 db.commit()  
 unlock = """  
 unlock tables  
 """  
 cursor.execute(unlock)  
 return 1  
except:

db.rollback()  
 return 0  
cursor.close()  
db.close()

3.多表联合查询查询两表中的课程号，课程名，教师名，学时，学分并按课程号升序排列

sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid  
 order by courses.cid asc  
   
 """

1. 多表联合查询：

Index=0：根据课程号来查询相关课程信息

Index=1：根据课程名来查询相关课程信息

Index=2：根据教师名来查询相关课程信息

if index == 0:  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid and courses.cid= '%d'  
 """ % int(message)  
if index == 1:  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid and courses.cname= '%s'  
 """ % message  
if index == 2:  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid and teacher.tname= '%s'  
 """ % message

6.根据教师名插入选课信息

sql0 = """  
 select tid  
 from teacher  
 where tname='%s'  
""" % tname  
sql = """  
 insert into choices  
 values('%d','%d','%d')  
 """ % (int(cid), int(sid), int(tid))

7.多表联合查询：根据学生号查询该学生选课全部信息，升序排序

sql = """  
 select courses.cname,teacher.tname,courses.ctime,teacher.tplace  
 from courses,teacher,choices  
 where choices.sid = '%d' and courses.cid = choices.cid and choices.tid = teacher.tid  
 order by courses.cid asc  
  
 """ % int(sid)

1. 删除课程信息

sql = """  
 delete from choices  
 where cid = '%d' and sid = '%d' and tid = '%d'  
 """ % (int(cid), int(sid), int(tid))

## 5.3数据库服务器端编程

编写服务器端存储过程、触发器等实现相应的应用功能并进行测试。

存储过程存放在服务器端数据库中，存储过程在第一次执行时进行语法检查和编译，执行后关闭数据库。存储过程执行速度快，改善系统性能，减少网络流量，增加代码的重要性和共享性，提供了安全机制。

存储过程：登陆时，调用存储过程p进行学生的账号密码的匹配

cursor = db.cursor(pymysql.cursors.DictCursor)  
cursor.callproc("p", (sid, 0)) # 参数为存储过程名称和存储过程接收的参数  
db.commit()  
password = cursor.fetchall()[0]['password']  
if pwd == password:  
 return 1

存储过程p：

MariaDB [homework]> create procedure p(in id int,out res int)

-> begin

-> select \* from student where sid = id;

-> set res = 0;

-> end //

触发器：当学生选课时，输入的课程id小于零时自动触发，会将id置为0

MariaDB [homework]> create trigger t1

-> before insert on choices

-> for each row

-> begin

-> if new.cid<0 then set new.cid=0;

-> end if;

-> end //

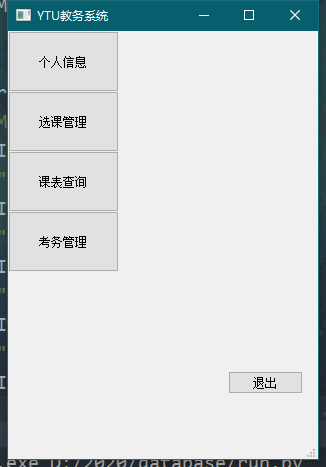
## 5.4客户端设计与编程实现

编写客户端程序实现相应的应用功能并进行测试。应该包含客户端服务器的联接代码。

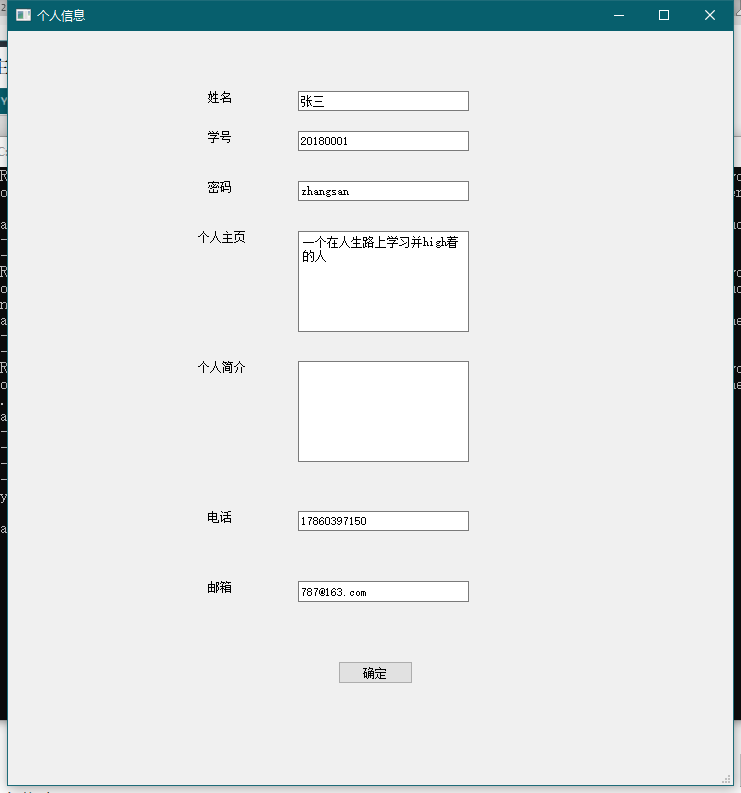
1.登录界面



1. 主界面



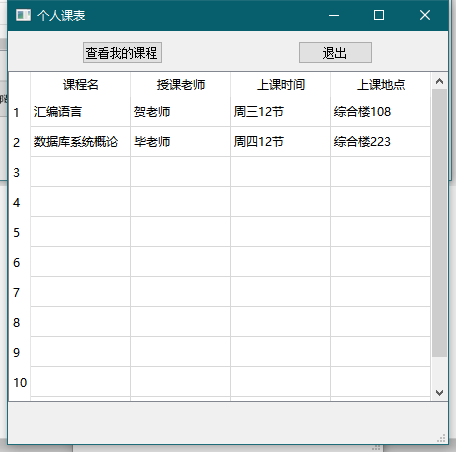
1. 个人信息



1. 选课界面



5.个人课表



1. 考务管理

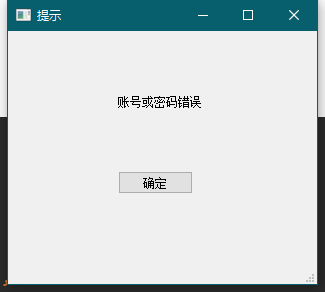


服务器端连接代码

db = pymysql.connect(  
 host='localhost',  
 port=3306,  
 user='root',  
 passwd='',  
 database='homework',  
 charset='utf8'  
)  
cursor = db.cursor()

测试：

登录失败：



成功登录后进入主界面

选课成功后



测试完成

## 5.5系统性能方面的设计与实现

影响数据库安全性、完整性、并发性、可恢复性等方面的考虑

安全性：用户通过学号和密码登录（密码唯一）

完整性：用户无法更改自身学号，也无法插入空值（即使插入，也不会响应），存在触发器使用户（学生）选课时无法输入负数课程号，同时choices（选课管理表）中存在外键

并发性：用户更新个人信息时加入了表锁，保证了信息不会发生“丢失修改，不可重复读，脏读”等情况

lock = """  
 lock tables student write  
 """  
cursor.execute(lock)  
sql = """  
 update student  
 set sname = '%s',password = '%s',phone = '%ld',email='%s',homepage='%s',profile='%s'  
 where sid = '%d'  
""" % (student[1], student[2], int(student[3]), student[4], student[5], student[6], int(student[0]))  
try:  
 cursor.execute(sql)  
 db.commit()  
 unlock = """  
 unlock tables  
 """  
 cursor.execute(unlock)  
 return 1  
except:

db.rollback()  
 return 0  
cursor.close()  
db.close()

可恢复性：在sql文件中，每当sql语句执行出错，都会执行rollback指令，保证了数据的可恢复性

每次调用sql函数都添加了rollback函数

except:

db.rollback()  
 return 0

# 6. 总结

本次数据库大作业中，我查了许多资料，一点点从懵懂到了解再到编写代码直到完成这次大作业，我收获了很多。我写的代码比较简洁明了，两个py文件一个存放界面初始化和相关函数调用，另一个存放设计的sql功能函数，使用了powerdesigner设计了数据库的逻辑结构和物理结构并生成了相关代码，其次使用了python以及相关第三方库pymysql，库函数的调用是代码结构更加清晰，我还设计了四大模块以及九个功能函数。当然也有许多不足，我感觉界面并不是十分的美观，也没有插入漂亮的图片，而且只写了学生端（以后有时间的话我会完成教师端），但是不论如何，我对自己的作业还是十分喜欢的，这是我第一个写出的带有界面和数据库的软件，收获良多。

本次数据库大作业中，我对数据库这门课程产生了一种新的认识，一种在编写程序上的认识，数据库技术是处理信息，管理数据最有效的方法，而且具有操作方便，操作简单实用等优点。它让我知道了在一个程序中，数据库所扮演的角色和它的地位，python与sql语言的完美融合使我对编程产生了更加浓厚的兴趣。还记得刚刚开始学数据库的时候，一上去接触了许多新的名词，让人云里雾里搞不明白，还好老师讲的非常细致，有时遇到比较难的问题会反复讲好几遍直到我们都明白。在这一个学期的数据库学习中，我只是对数据库有了一个相关的认识，对于如何调用数据库以及在编程语言中对数据库的书写，我基本上只有一个笼统的了解，是这次数据库大作业给了我机会，让我真正意义上弄明白了数据库的使用。

# 参考文献

1. 佚名 PyQt5教程 <http://code.py40.com/face>
2. 佚名 （PYQt5和QT designer）PyQt5学习入门

<http://www.1118pc.com/show_pdetails.asp?id=922>

1. challenge\_me PowerDesigner的基本使用

<https://blog.csdn.net/challenge_me/article/details/82223806>

1. 花生福 mysql触发器trigger 实例详解

<https://www.cnblogs.com/zh-1721342390/p/9602941.html>

1. weixin\_30684743 mysql中lock tables与unlock tables(锁表/解锁)使用总结<https://blog.csdn.net/weixin_30684743/article/details/99768183?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.nonecase&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.nonecase>
2. 马又虫 MySQL多次单表查询和多表联合查询

<https://www.cnblogs.com/youmingDDD/p/11921187.html>

# 源代码选摘

客户端代码节选

# 登录界面  
class LOGIN(QtWidgets.QMainWindow, login.Ui\_MainWindow):  
 def \_\_init\_\_(self):  
 super(LOGIN, self).\_\_init\_\_()  
 self.setupUi(self)  
 self.ts = TISHI(self)  
 self.mw = MAIN(self)  
 # 判断登录账号密码是否正确  
 def sure(self):  
 sid = int(self.lineEdit.text())  
 pwd = self.lineEdit\_2.text()  
 # 通过数据库查询账号密码  
 if sql.SqlConnect\_Login(sid, pwd):  
 self.mw.show() # mainwindow主界面显示  
 self.mw.sid = sid # 传递用户账户，方便后续操作  
 # print("self.mw.sid="+str(self.mw.sid))  
 self.close() # 登录成功后，关闭当前窗口  
 else:  
 # print("学号或密码错误\n")  
 self.ts.show()  
# 主界面  
class MAIN(QtWidgets.QMainWindow, stu.Ui\_MainWindow):  
 def \_\_init\_\_(self, last\_form):  
 super(MAIN, self).\_\_init\_\_()  
 self.setupUi(self)  
 self.last\_form = last\_form  
 self.sid = -1  
 self.stumessage = StudentMessage(self) # 个人信息界面  
 self.xuanke = CHOICES(self) # 选课界面  
 self.kebiao = COURSE(self)  
 self.kaoshi = EXAM(self)  
 def no(self):  
 self.close()  
 # 显示个人信息界面  
 def personal(self):  
 self.stumessage.sid = self.sid  
 Student\_ID, Student\_Name, Student\_Password, Student\_phone, Student\_email, Student\_homepage, Student\_profile = sql.SqlConnect\_Person(  
 self.sid)  
 self.stumessage.lineEdit.setText(str(Student\_Name))  
 self.stumessage.lineEdit\_2.setText(str(Student\_ID))  
 self.stumessage.textEdit.setText(str(Student\_homepage))  
 self.stumessage.textEdit\_2.setText(str(Student\_profile))  
 self.stumessage.lineEdit\_3.setText(str(Student\_phone))  
 self.stumessage.lineEdit\_4.setText(str(Student\_email))  
 self.stumessage.lineEdit\_5.setText(str(Student\_Password))  
 self.stumessage.show()  
 # 选课界面  
 def choices(self):  
 self.xuanke.sid = self.sid  
 self.xuanke.show()  
 # 查看我的课程  
 def courses(self):  
 self.kebiao.sid = self.sid  
 self.kebiao.show()  
 def exam(self):  
 self.kaoshi.sid = self.sid  
 self.kaoshi.show()  
# 账号或密码错误界面  
class TISHI(QtWidgets.QMainWindow, tishi.Ui\_MainWindow):  
 def \_\_init\_\_(self, last\_form):  
 super(TISHI, self).\_\_init\_\_()  
 self.setupUi(self)  
 self.last\_form = last\_form  
# 提示选课成功界面  
class TISHI2(QtWidgets.QMainWindow, tishi2.Ui\_MainWindow):  
 def \_\_init\_\_(self, last\_form):  
 super(TISHI2, self).\_\_init\_\_()  
 self.setupUi(self)  
 self.last\_form = last\_form  
# 个人信息界面  
class StudentMessage(QtWidgets.QMainWindow, studentmessage.Ui\_MainWindow):  
 def \_\_init\_\_(self, last\_form):  
 super(StudentMessage, self).\_\_init\_\_()  
 self.setupUi(self)  
 self.last\_form = last\_form  
 self.sid = -1  
 # 初始化信息  
 self.lineEdit.setText('')  
 self.lineEdit\_2.setText('')  
 self.lineEdit\_5.setText('')  
 self.textEdit.setText('')  
 self.textEdit\_2.setText('')  
 self.lineEdit\_3.setText('')  
 self.lineEdit\_4.setText('')  
 def S\_save(self):  
 Student\_Name = self.lineEdit.text()  
 Student\_Password = self.lineEdit\_5.text()  
 Student\_phone = self.lineEdit\_3.text()  
 Student\_email = self.lineEdit\_4.text()  
 Student\_homepage = self.textEdit.toPlainText()  
 Student\_profile = self.textEdit\_2.toPlainText()  
 student = (  
 self.sid, Student\_Name, Student\_Password, Student\_phone, Student\_email, Student\_homepage, Student\_profile)  
 sql.SqlConnect\_Save\_Student(student)  
 self.close()  
# 选课界面  
class CHOICES(QtWidgets.QMainWindow, xuanke.Ui\_MainWindow):  
 def \_\_init\_\_(self, last\_form):  
 super(CHOICES, self).\_\_init\_\_()  
 self.setupUi(self)  
 self.last\_form = last\_form  
 self.sid = -1  
 self.tishi2 = TISHI2(self)  
 def c\_save(self):  
 self.close()  
 def all(self):  
 Am = sql.SqlConnect\_Allcourses()  
 # print(type(Am[0][0]))  
 for i in range(10):  
 for j in range(5):  
 try:  
 self.tableWidget.setItem(i, j, QtWidgets.QTableWidgetItem(str(Am[i][j])))  
 except:  
 self.tableWidget.setItem(i, j, QtWidgets.QTableWidgetItem(""))  
 def search(self):  
 index = self.comboBox.currentIndex()  
 message = self.lineEdit.text()  
 Am = sql.SqlConnect\_Search\_Courses(index, message)  
 for i in range(10):  
 for j in range(5):  
 try:  
 self.tableWidget.setItem(i, j, QtWidgets.QTableWidgetItem(str(Am[i][j])))  
 except:  
 self.tableWidget.setItem(i, j, QtWidgets.QTableWidgetItem(""))  
 def add(self):  
 cid = self.lineEdit\_2.text()  
 tname = self.lineEdit\_3.text()  
 if sql.SqlConnect\_Add\_Course(cid, tname, self.sid):  
 self.tishi2.show()  
 def delete(self):  
 cid = self.lineEdit\_2.text()  
 tname = self.lineEdit\_3.text()  
 if sql.SqlConnect\_Del\_Course(cid, tname, self.sid):  
 self.tishi2.show()

Sql代码节选

import pymysql  
# 学生账号密码匹配  
def SqlConnect\_Login(sid, pwd):  
 try:  
 cursor = db.cursor(pymysql.cursors.DictCursor)  
 cursor.callproc("p", (sid, 0)) # 参数为存储过程名称和存储过程接收的参数  
 db.commit()  
 password = cursor.fetchall()[0]['password']  
 if pwd == password:  
 return 1

except:

db.rollback()  
 return 0  
 cursor.close()  
 db.close()  
# 查询学生信息  
def SqlConnect\_Person(sid):  
 cursor = db.cursor()  
 sql = """  
 select \*  
 from student  
 where sid = '%d'  
 """ % sid  
 try:  
 cursor.execute(sql)  
 studentmessage = cursor.fetchall()[0]  
 return studentmessage  
 except:

db.rollback()  
 return (0, 0, 0, 0, 0, 0, 0)  
  
 cursor.close()  
 db.close()  
  
  
# 更新学生信息  
def SqlConnect\_Save\_Student(student):  
lock = """  
 lock tables student write  
 """  
cursor.execute(lock)  
sql = """  
 update student  
 set sname = '%s',password = '%s',phone = '%ld',email='%s',homepage='%s',profile='%s'  
 where sid = '%d'  
""" % (student[1], student[2], int(student[3]), student[4], student[5], student[6], int(student[0]))  
try:  
 cursor.execute(sql)  
 db.commit()  
 unlock = """  
 unlock tables  
 """  
 cursor.execute(unlock)  
 return 1  
except:

db.rollback()  
 return 0  
cursor.close()  
db.close()

# 查看所有课程  
def SqlConnect\_Allcourses():  
 cursor = db.cursor()  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid  
 order by courses.cid asc  
   
 """  
 try:  
 cursor.execute(sql)  
 all = cursor.fetchall()  
 return all  
 except:

db.rollback()  
 return None  
 cursor.close()  
 db.close()  
# 搜索课程  
def SqlConnect\_Search\_Courses(index, message):  
 cursor = db.cursor()  
 if index == 0:  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid and courses.cid= '%d'  
 """ % int(message)  
 if index == 1:  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid and courses.cname= '%s'  
 """ % message  
 if index == 2:  
 sql = """  
 select courses.cid,courses.cname,teacher.tname,courses.hours,courses.credit  
 from courses,teacher  
 where courses.cid = teacher.cid and teacher.tname= '%s'  
 """ % message  
 try:  
 cursor.execute(sql)  
 all = cursor.fetchall()  
 return all  
 except:

db.rollback()  
 return None  
 cursor.close()  
 db.close()  
# 添加课程  
def SqlConnect\_Add\_Course(cid, tname, sid):  
 cursor = db.cursor()  
 cursor1 = db.cursor()  
 sql0 = """  
 select tid  
 from teacher  
 where tname='%s'  
 """ % tname  
 try:  
 cursor.execute(sql0)  
 tid = cursor.fetchall()[0][0]  
 except:  
 return 0  
 sql = """  
 insert into choices  
 values('%d','%d','%d')  
 """ % (int(cid), int(sid), int(tid))  
 try:  
 cursor1.execute(sql)  
 db.commit()  
 cursor.close()  
 db.close()  
 return 1  
 except:

db.rollback()  
 return 0