哈尔滨工业大学

<<数据库系统>>

实验报告一

(2022 年度春季学期)

姓名：	冯开来
学号：	1190201215
学院：	计算机学院
教师：	程思瑶

实验一

一、实验目的

在熟练掌握 MySQL 基本命令、SQL 语言以及用 C 语言编写 MySQL 操作程序的基础上，学习简单数据库系统的设计方法，包括数据库概要设计、逻辑设计。

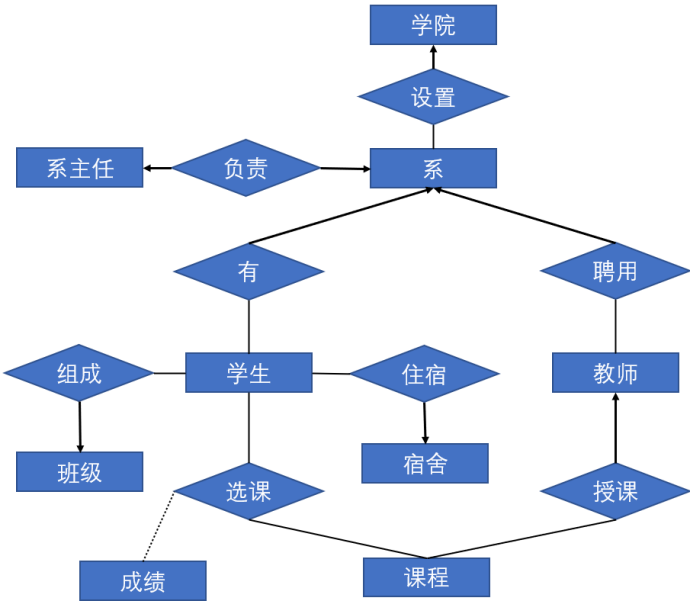
二、实验环境

Windows XP 操作系统、MySQL 关系数据库管理系统、MinGW 编译器或 Microsoft Visual C++编译器。(可以用其他操作系统和编程环境，数据库尽量选择 MySQL)

本次实验可使用 C，C++，JAVA，PHP 或其他语言均可。

三、实验过程及结果

ER 图：



一共有学院、系、系主任、学生、教师、班级、住宿、课程 8 个实体集，还有成绩这一个联系集。其中系主任和系是一对一的关系，学生和课程是多对多的关系，剩余皆为一对多的关系，如一门课只由一位老师教，一个老师可以教多门课。

关系表：

学生和课程是多对多关系，所以转换成逻辑数据库的时候添加了联系集成绩。其余的一对多关系，均在 n 的关系表中加入了 1 的主码，如在学生的表中添加了班级的主码班号，宿舍的主码宿舍号。具体如下：

学院：college(name, dname) // dname 为学院领导姓名

系：department(name, did, cname) // did 为系主任工号，cname 为学院名

系主任：director(id, name, phone)

学生：student(sid, sname, de_name, class, dorm) // de_name 为所在系名

教师：teacher(tid, tname, de_name) // de_name 为所在系名

宿舍：dorm(id, place) // id 为宿舍号，place 为所在公寓

班级：class(id)

课程：course(cid, cname, tid) // tid 为授课教师工号

成绩：grade(cid, sid, score)

关系的完整性约束：

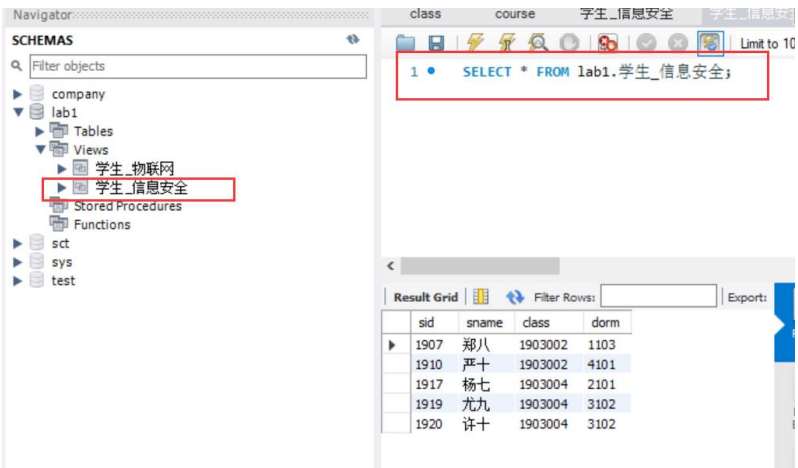
主键约束如上图划线部分已经给出。

外键约束：系 department 中 cname，did 是外键。学生 student 中 de_name，class，dorm 是外键。教师 teacher 中 de_name 是外键。课程 course 中 tid 为外键。

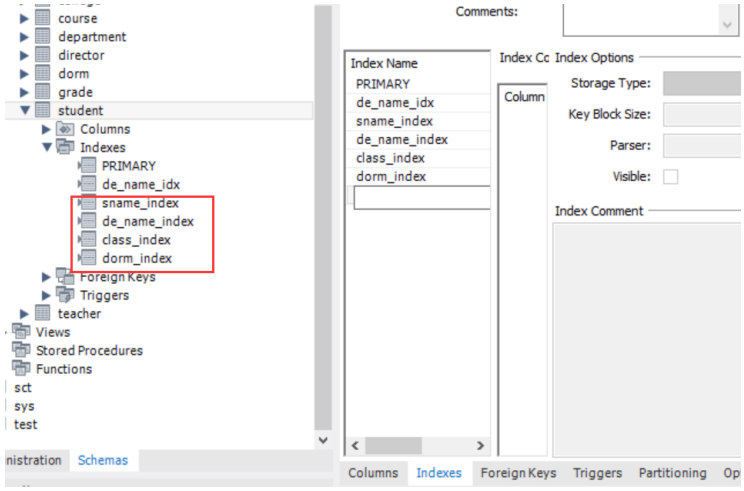
空值约束全为非空属性，具体约束方法后面程序中会体现。

常用查询的视图和常用属性的索引：

我们为同一院系的同学建立了视图，建立的方法在后续程序中会体现。如信息安全系的学生：



对于非主键的属性索引，我们以学生为例，为除了主键学号的其他属性都建立了索引，如下图：



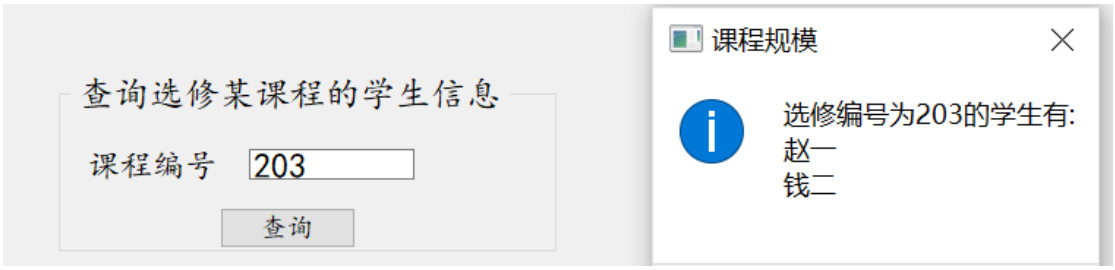
查询操作：

如查询选课数目超过 n 的学生姓名。首先进行了输入 n 的检查，是否为空值，错误值并且给予提示。在具体查询中的 sql 语句使用了关系的连接、分组操作，使用 having 语句进行筛选，下图中的 query。关键代码如下：

```
def query_stu(self):
    text = self.lineEdit.text()
    if not text:
        QMessageBox.warning(self, '警告', '请输入数目')
    elif int(text)<0:
        QMessageBox.warning(self, '警告', '请输入正确的数量')
    else:
        con = pymysql.connect(host='localhost', port=3306, user='root', password='fkl001206', charset='utf8',
                              database='lab1')  # 连接数据库
        cur = con.cursor()  # 执行sql语句的游标
        # 连接查询，体现分组，having
        query = 'select sname from student natural join grade group by grade.sid having count(*) > %s'
        cur.execute(query, [text])
        curr = cur.fetchall()
        if len(curr)>0:
            msg = '课程数目超过{num}的学生有:\n'.format(num=text)
            for item in curr:
                msg += item[0]+'\n'
        else:
            msg = '无结果'
        QMessageBox.information(self, '学习状况', msg)
        con.close()
        cur.close()
```

如查询选修了某特定课程号的课程的学生姓名。这里面的 sql 语句用到了嵌套查询，下图中的 query。

```
cur = con.cursor()  # 执行sql语句的游标
query = 'select sname from student where sid in (select sid from grade where cid = %s)'# 嵌套查询
cur.execute(query, [c_num])
curr = cur.fetchall()
if len(curr) > 0:
    msg = '选修编号为{num}的学生有:\n'.format(num=c_num)
    for item in curr:
        msg += item[0] + '\n'
else:
    msg = '无结果'
QMessageBox.information(self, '课程规模', msg)
con.close()
cur.close()
```



此外，我还提供了其他查询，如查询所有学生信息，所有教师信息，某学生所有及格课程的平均分。如下图：

插入操作：

这里提供了学生和课程还有成绩的插入操作。我们以学生为例，插入操作首先需要判断是否所有值都是非空的，如果少了一条信息，会提供相关提示。此外

还要通过主键学号判断插入的信息是否重复，如果重复给予提示。还有如果该专业未出现在系 department 表中，也不会允许正确插入。代码如下图：

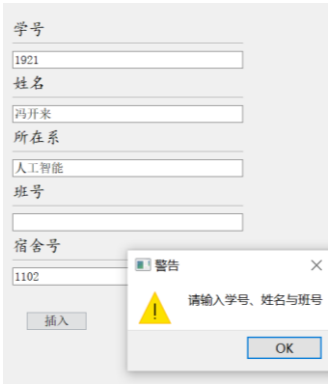
```
def insert(self):
    num, name, de_name, classid, dorm = self.lineEdit.text(), self.lineEdit_2.text(), self.lineEdit_3.text(), \
                                        self.lineEdit_4.text(), self.lineEdit_5.text()
    if not num or not name or not de_name or not classid or not dorm:# 插入错误值，空值，提醒
        QMessageBox.warning(self, '警告', '请输入学号、姓名与班号')
    elif len(num) != 4:
        QMessageBox.warning(self, '警告', '学号为4位数字')
    else:
        con = pymysql.connect(host='localhost', port=3306, user='root', password='fkl001206', charset='utf8',
                              database='lab1')  # 连接数据库
        cur = con.cursor()  # 执行sql语句的游标
        query = 'select * from student where sid=%s'
        if cur.execute(query, [num]):
            QMessageBox.warning(self, '插入异常', '该学号已存在，请重新输入')# 重复值提示
        elif not cur.execute('select * from department where name = %s', [de_name]):
            QMessageBox.warning(self, '插入异常', '该专业不存在，请尝试插入正确条目')
        else:
            QMessageBox.information(self, '成功', '成功插入一条学生数据')
            query = 'insert into student(sid, sname, de_name, class, dorm) values (%s, %s, %s, %s, %s)'
            cur.execute(query, [num, name, de_name, classid, dorm])
            con.commit()
```

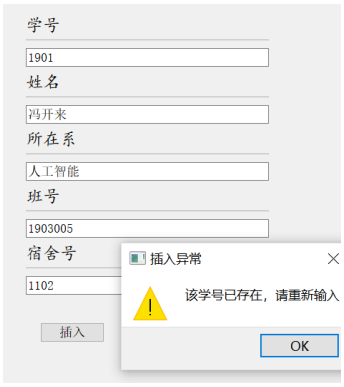
空值判断

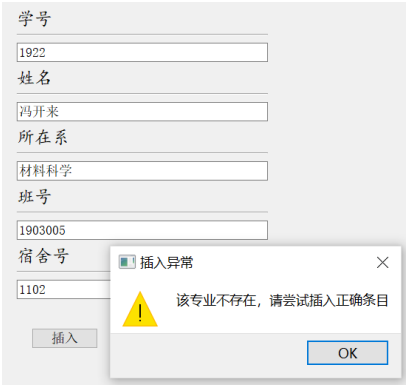
错误值判断

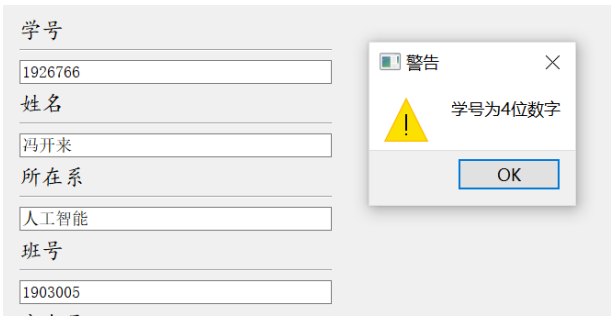
重复信息

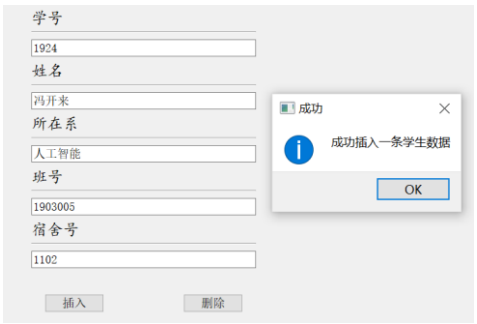
外键约束











删除操作：

我们同样以删除学生信息为例，在删除操作中只需要提供正确的学号就能进行操作。同时这里进行了触发器功能，如果检测到该学号在成绩 grade 表中也出现，则一并删去成绩 grade 表中的元组。同样的对于删除课程信息也有可能删除成绩中的元组，删除成绩信息则不会影响到学生和课程信息。

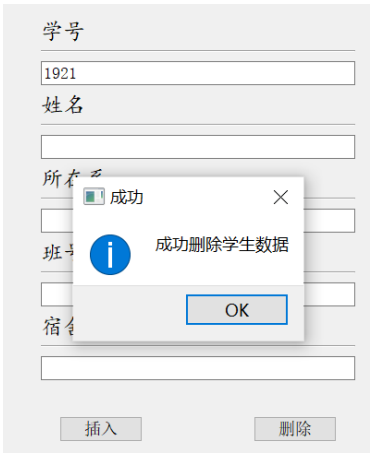
具体代码和截图如下：

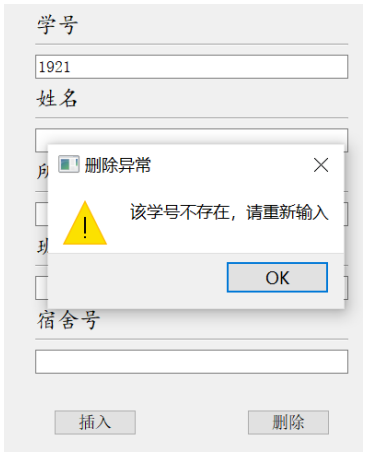
```python
def delete(self):
    num= self.lineEdit.text()
    if not num:
        QMessageBox.warning(self, '警告', '学号为空')
    elif not num.isdigit() or len(num) != 4:
        QMessageBox.warning(self, '警告', '学号只可为4位数字')
    else:
        con = pymysql.connect(host='localhost', port=3306, user='root', password='fkl001206', charset='utf8',
                              database='lab1')  # 连接数据库
        cur = con.cursor()  # 执行sql语句的游标
        query = 'select * from student where sid=%s'
        if not cur.execute(query, [num]):
            QMessageBox.warning(self, "删除异常", "该学号不存在，请重新输入")
        else:
            if cur.execute('select * from grade where sid=%s', [num]):
                QMessageBox.information(self, '提示', '该学号在其他表中被作为外键引用，触发器已默认删除其他表中对应条目的数据')
                query = 'delete from grade where sid=%s'
                cur.execute(query, [num])
            QMessageBox.information(self, '成功', '成功删除学生数据')
            query = 'delete from student where sid=%s'
            cur.execute(query, [num])
            con.commit()
```

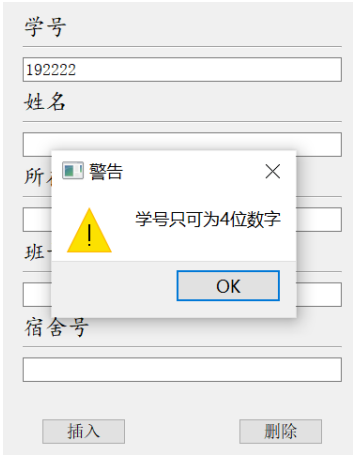
空值和错误值判断

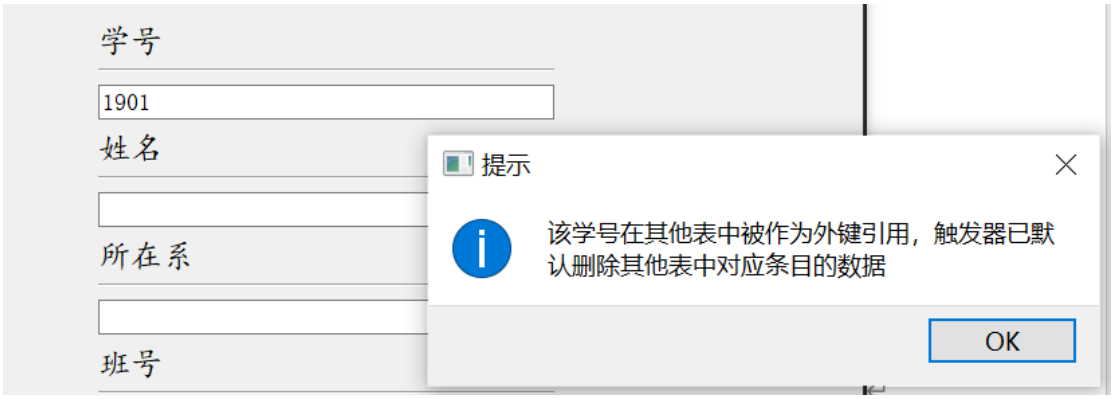
外键约束

触发器：删除成绩表中的元组









建立视图和索引：

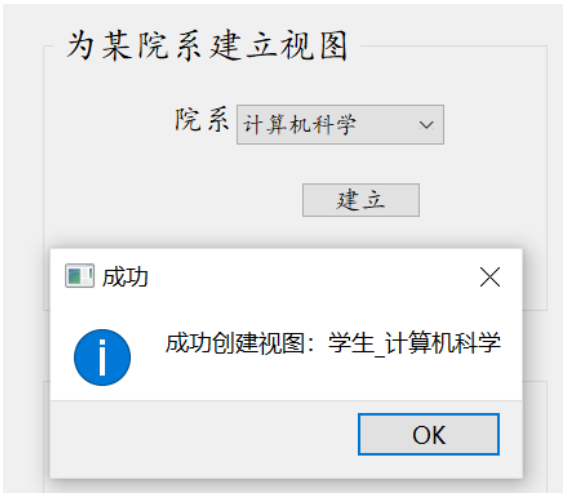
在本次实验中，我添加了建立视图和建立索引的功能。并且建立视图是以学生的院系，为相同系的学生建立视图。这里采用了选框的组件，所以不需要判断空值和错误值。但是需要判断该视图是否已经被创建。

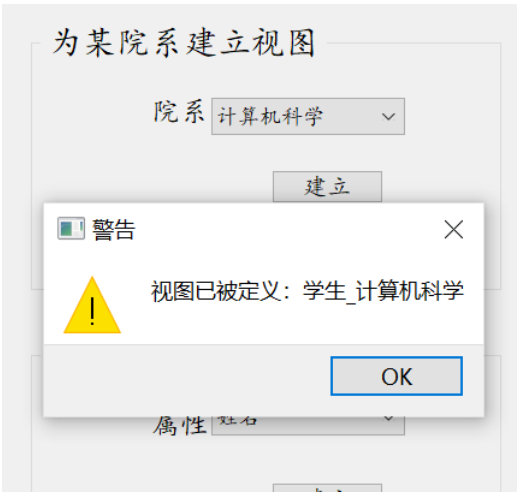
具体代码和结果如下图：

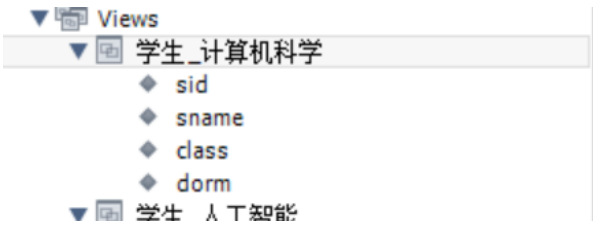
```
def create_department_view(self):
    d_name = self.comboBox.currentText()
    view_name = '学生_' + d_name
    con = pymysql.connect(host='localhost', port=3306, user='root', password='fkl001206', charset='utf8',
                          database='lab1')  # 连接数据库
    cur = con.cursor()
    query = 'select count(*) from information_schema.VIEWS where TABLE_SCHEMA="lab1" and TABLE_NAME=%s'
    cur.execute(query, [view_name])  # 先查询视图是否已被定义
    if cur.fetchone()[0] == 1:
        QMessageBox.warning(self, '警告', '视图已被定义：' + view_name)
    else:
        query = 'create view ' + view_name + ' as select sid ,sname, class, dorm from student where de_name = %s'
        cur.execute(query, [d_name])
        QMessageBox.information(self, '成功', '成功创建视图：' + view_name)
        cur.close()
        con.close()
```

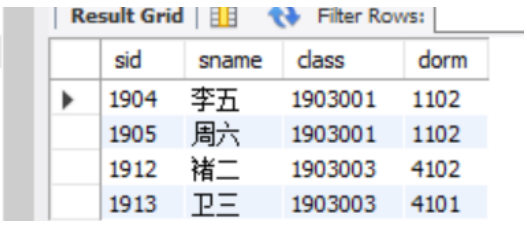
判断该视图是否重复创建

创建语句







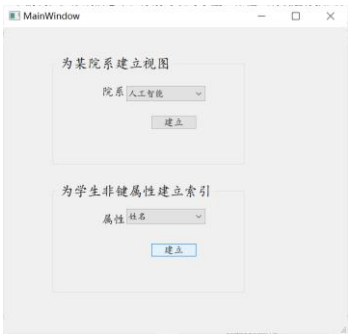


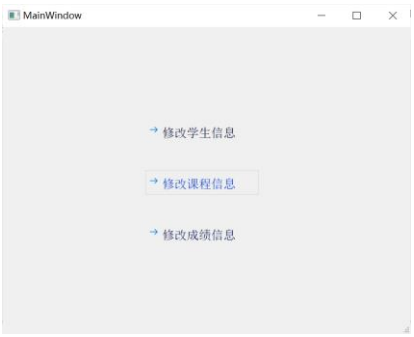
Result Grid | Filter Rows:

sid	sname	class	dorm
1904	李五	1903001	1102
1905	周六	1903001	1102
1912	褚二	1903003	4102
1913	卫三	1903003	4101

界面设计：

本实验界面基于 python 的 PyQt5 库。主页面设计为三个功能，分别是查询功能，修改功能和建立视图和索引。在查询功能中分为查看学生、查看老师和其他查询三个板块；在修改信息中，分别可以对学生、课程、成绩进行插入和删除操作。具体界面如下图：





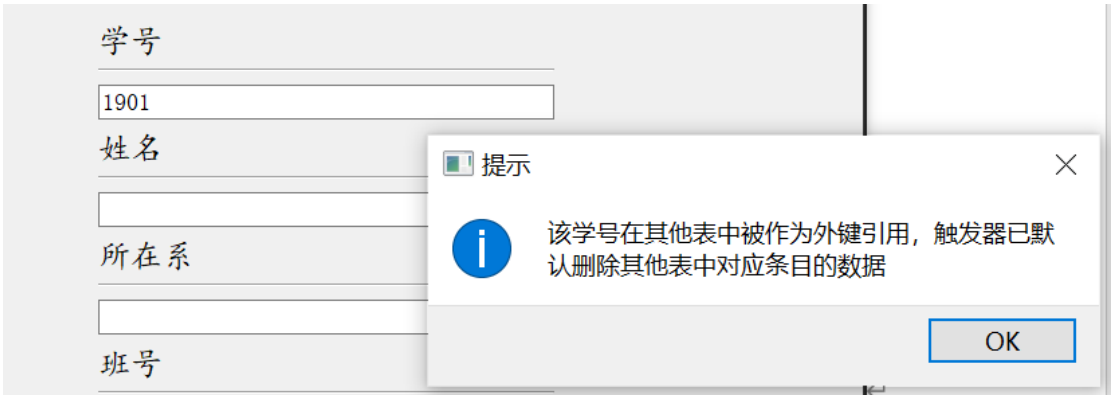
触发器功能：

在进行删除信息的时候增加了触发器功能，以学生为例如果检测到该学号在成绩 grade 表中也出现，则一并删去成绩 grade 表中的元组。同样的对于删除课程信息也有可能删除成绩中的元组，删除成绩信息则不会影响到学生和课程信息。具体代码和截图如下：

```
cur = con.cursor()  # 执行sql语句的游标
query = 'select * from student where sid=%s'
if not cur.execute(query, [num]):
    QMessageBox.warning(self, "删除异常", "该学号不存在，请重新输入")
else:
    if cur.execute('select * from grade where sid=%s', [num]):
        QMessageBox.information(self, '提示', '该学号在其他表中被作为外键引用，触发器已默认删除其他表中对应条目的数据')
        query = 'delete from grade where sid=%s'
        cur.execute(query, [num])
    QMessageBox.information(self, '成功', '成功删除学生数据')
    query = 'delete from student where sid=%s'
    cur.execute(query, [num])
    con.commit()
```

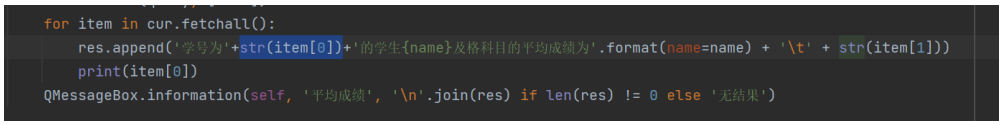
外键约束

触发器：删除成绩表中的元组



四、实验心得

1. 关于 Qt 库的使用和各个组件的设置代码。因为之前没接触过 Qt，所以这一部分花了很多时间，查了很多博客和资料，一步一步摸索做出来的。
2. 在做插入操作的时候，从数据库查询到的学号是整型，但是将学号 item[0] 作为输出的时候没有转换为字符串类型，所以在这一部分一直出错。后来添加了 str(item[0])后，就解决了问题。



3. 还有一些问题就是没考虑全面，比如在学生表中插入的新信息，它的班号不在班级 class 表中的班号时理论上应该在 class 表中也同时插入，作为一个插入操作的触发器，但是因为时间原因也只做了删除操作的触发器功能。且事务管理也没有做。