


# 数据库工程作业

要求:

- 1. 完成一个小型的数据库信息管理系统（或部分功能），并填写工程报告；程序和报告请在规定时间之内上传。
- 2. 开发模式（B/S 或 C/S）、开发高级语言任选，后台数据库使用大型数据库管理系统（SQL Server、Oracle、MySQL 等），不要使用桌面数据库。
- 3. 报告中所列举的四种操作，每种操作举一个例子即可。

## 工程作业报告

1. 项目信息（10 分）

|               |   |    |     |    |      |
|---------------|---|----|-----|----|------|
| 学号            | 2011937   | 姓名 | 姜志凯 | 专业 | 信息安全 |
| 项目名称          | 大学信息管理系统  |    |     |    |      |
| 必备环境          | MySQL、Navicat、python 3.9、pycharm、pymysql  |    |     |    |      |
| 系统主要功能简介（4 分） | <p>大学的信息管理系统，包含学生、教师、学院、课程等信息及它们之间的关系，并提供插入、查询、更改、删除等操作，如添加、查询、更改、删除学生或教师信息；增加、查询、更改、删除学生选课信息；增加、查询、更改、删除教师授课信息等，还能按某一属性对元组排序。</p> <p>含有事务应用的删除操作：删除教师时，连同他所教授的课程一并删除；</p> <p>触发器控制下的添加操作：添加新的课程信息时，课时应不超过 17h，否则会提示课时太多，学不完；</p> <p>存储过程控制下的更新操作：在更改授课信息表 teach 时，若更改了课程代码，则会同步更新课程信息表 lesson 中对应的课程代码；</p> <p>含有视图的查询操作：除去不关心的信息，如学生入学年份、学院信息、课程学分、课时等，只保留想要的信息，即哪些学生选了哪些课，得了多少分。</p> |    |     |    |      |
| 系统主要页面截图（6 分） |   |    |     |    |      |

## 管理员登陆

管理员账号：

管理员密码：

登陆

返回首页

## 请选择操作

学生信息

教师信息

课程信息

选课信息

授课信息

退出系统

学生信息

学生信息：

学号：

00001

姓名：

张三

性别：

男

年龄：

网络空间安全学院

操作：

新建学生信息

更新选中学生信息

删除选中学生信息

| 学号    | 姓名 | 性别 | 学院          |
|-------|----|----|-------------|
| 00001 | 张三 | 男  | 网络空间安全学院    |
| 00002 | 李凤 | 女  | 软件学院        |
| 00003 | 刘二 | 男  | 计算机学院       |
| 00004 | 赵四 | 女  | 人工智能学院      |
| 00005 | 王麻 | 男  | 电子信息与光学工程学院 |
| 98725 | 张蛋 | 男  | 网络空间安全学院    |

学生登陆

学生账号：

学生密码：

登陆

返回首页

|  |   |
|--|---|
|  | <div style="background-color: #008000; color: white; padding: 10px; margin-bottom: 20px;"> <h2 style="margin: 0;">学生信息查看</h2> </div> <div style="margin-bottom: 20px;"> <p style="font-size: 1.2em; margin: 5px 0;">学号:00001</p> <p style="font-size: 1.2em; margin: 5px 0;">姓名:张三</p> <p style="font-size: 1.2em; margin: 5px 0;">性别:男</p> <p style="font-size: 1.2em; margin: 5px 0;">入学时间:2020-09-11</p> <p style="font-size: 1.2em; margin: 5px 0;">学院:网络空间安全学院</p> </div> <div style="border: 1px solid #ccc; padding: 5px; display: inline-block;"> <a href="#">返回首页</a> </div> |
|--|---|

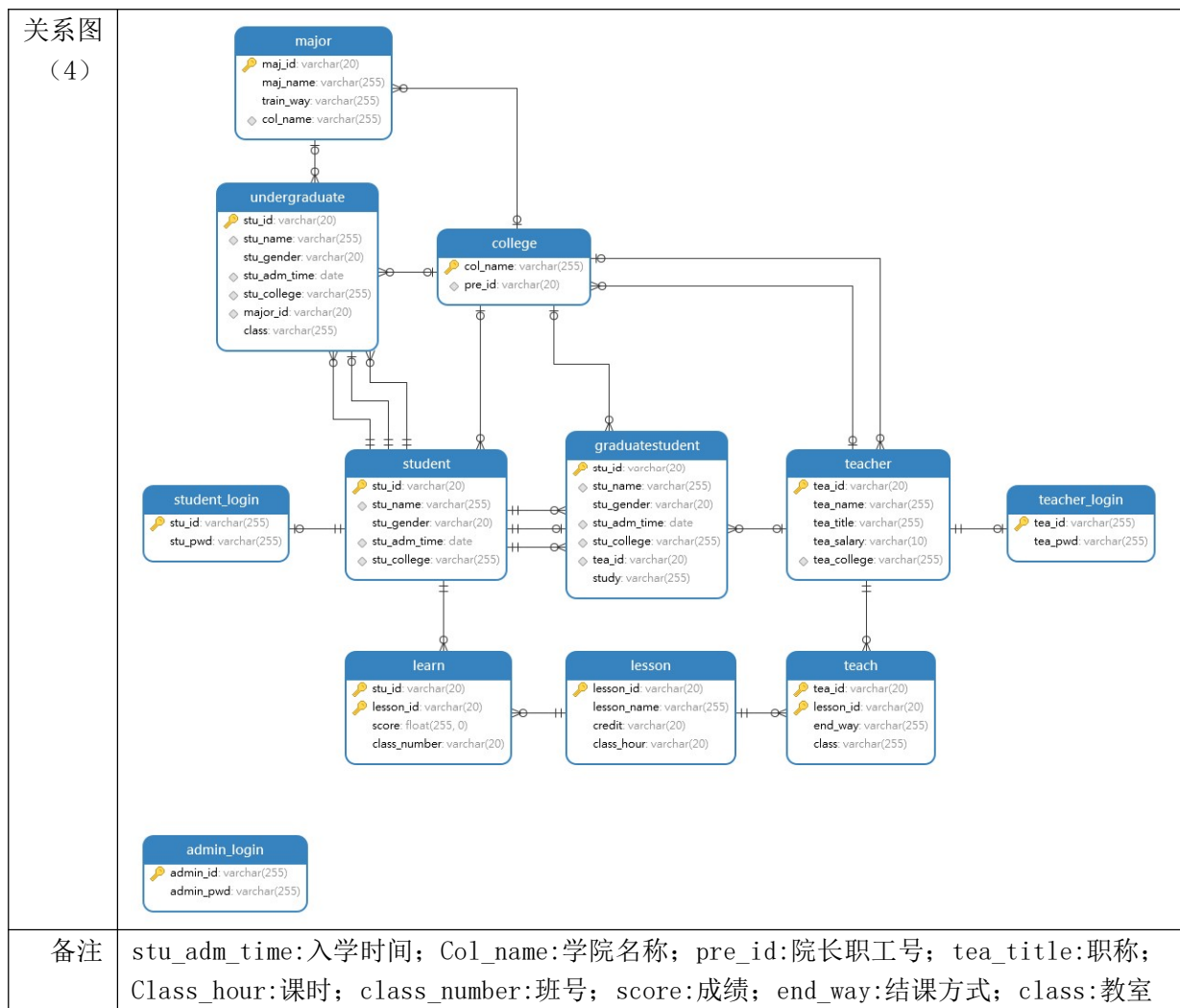
## 2. 系统配置 (10 分)

| 说明                     |      | (2 分) 请说明系统配置情况 (后台数据库, 高级语言);<br>(8 分) 请使用连接串连接高级语言和数据库, 并分析字符串的各个部分。  |            |            |            |
|------------------------|------|---|------------|------------|------------|
| 配置<br>步骤<br>2 分        | DBMS | 1. MySQL  |            |            |            |
|                        | 高级   | 1. python   |            |            |            |
|                        | 语言   | 2. SQL  |            |            |            |
| 连接串<br>分析<br>(6 分)     |      | 序号  | 名称         | 功能说明       | 取值         |
|                        |      | 1   | host       | 要连接的数据库的主机 | localhost  |
|                        |      | 2   | user       | MySQL 用户名  | root       |
|                        |      | 3   | password   | MySQL 密码   | 保密         |
|                        |      | 4   | database   | 选择的数据库     | school     |
|                        |      | 5   | port       | 连接端口       | 3306       |
|                        |      | 6   | autocommit | 自动提交功能     | false (默认) |
| 连接串代码<br>(截屏)<br>(2 分) |      | <pre># 打开数据库连接 db = pymysql.connect(host="localhost", user="root", password="jzk1314.", database="school", port=3306) cursor = db.cursor() # 使用cursor()方法获取操作游标</pre> |            |            |            |
| 备注                     |      | 不想使用自动提交功能, 手动提交, 还能回滚, 所以不加 autocommit 参数。   |            |            |            |

## 3. 数据库设计 (14 分)

|    |  |
|----|--|
| 说明 | (10 分) 按照数据表的创建顺序, 依次给出所涉及数据表的信息, 其中参照字段以“(字 |
|----|--|

|             |   |  |                      |                      |   |
|-------------|---|--|----------------------|----------------------|---|
|             | 段 1, 字段 2, …… , 字段 n)” 的形式给出, 被参照字段以 “表名 (字段 1, 字段 2, …… , 字段 n)” 的形式给出;<br>(4 分) 一般 DBMS 都可以为数据库生成关系图, 请将该图片截屏并粘贴到表格中。 |  |                      |                      |   |
| 数据表<br>(10) | 创建<br>顺序  | 数据表名称  | 主键                   | 参照属性                 | 被参照表及属性   |
|             | 1   | student(stu_id,<br>stu_name,stu_gender,<br>stu_adm_time,stu_college<br>) | stu_id               | stu_college          | college(col_name,<br>pre_id)<br><b>Col_name</b>   |
|             | 2   | college(col_name,pre_id)   | col_name             | Pre_id               | teacher(tea_id,<br>tea_name,tea_title,<br>tea_salary,<br>tea_college)<br><b>Tea_id</b>  |
|             | 3   | teacher(tea_id,<br>tea_name,tea_title,<br>tea_salary,tea_college)        | Tea_id               | Tea_college          | college(col_name,<br>pre_id)<br><b>Col_name</b>   |
|             | 4   | lesson(lesson_id, lesson_<br>name, credit, class_hour)                   | lesson_id            | 无                    | 无   |
|             | 5   | Learn(stu_id, lesson_id,<br>Score, class_number)                         | Stu_id,<br>lesson_id | Stu_id,<br>lesson_id | student(stu_id,<br>stu_name,stu_gender,<br>stu_adm_time,stu_coll<br>ege)<br><b>stu_id</b><br>lesson(lesson_id, less<br>on_name, credit, class_<br>hour)<br><b>lesson_id</b> |
|             | 6   | Teach(tea_id, lesson_id,<br>End_way, class)                              | Tea_id,<br>Lesson_id | Tea_id,<br>Lesson_id | teacher(tea_id,<br>tea_name,tea_title,<br>tea_salary,tea_colleg<br>e)<br><b>Tea_id</b><br>lesson(lesson_id, less<br>on_name, credit, class_<br>hour)<br><b>lesson_id</b>    |



#### 4. 含有事务应用的删除操作（13分）

|              |   |
|--------------|---|
| 说明           | <p>(1分) 简要说明该操作所要完成的功能;</p> <p>(2分) 该操作会涉及的表(必须含有两张或两张以上的关系表,同时以“表名”的形式给出)</p> <p>(1分) 表连接涉及字段描述(描述方式为“表1.属性=表2.属性”)</p> <p>(1分) 删除条件涉及的字段描述(以“表名.属性=?”形式给出)</p> <p>(4分) 实现该操作的关键代码(高级语言、SQL),截图即可;(其中如果删除语句中不包含任何形式的事务应用将扣除3分)</p> <p>(4分) 如何执行该操作,按所述方法能够正常演示程序则给分。</p> |
| 功能描述<br>(1分) | 删除教师时,连同他所教授的课程一并删除   |
| 涉及           | teacher、teach、lesson  |

|                  |   |                              |
|------------------|---|------------------------------|
| 的表<br>(2分)       |   |                              |
| 表连接涉及字段<br>(1分)  | teacher.tea_id = teach.tea_id<br>teach.lesson_id = lesson.lesson_id   |                              |
| 删除条件字段描述<br>(1分) | 字段  | 规则                           |
|                  | teacher.tea_id  | 输入框中将要删除的教师的职工号              |
|                  | teach.tea_id  | 等于输入框中的将要删除的教师职工号            |
|                  | teach.lesson_id   | 将要删除的教师在 teach 表中所对应的课程的课程代码 |
|                  | lesson.lesson_id  | 等于上述对应的将要删除的课程代码             |
| 代码<br>(4分)       | <pre># 打开数据库连接 db = pymysql.connect(host="localhost", user="root", password="jzk1314.", database="school", port=3306) cursor = db.cursor() # 使用cursor()方法获取操作游标 sql1 = "BEGIN" sql2 = "DELETE FROM lesson WHERE lesson_id in (SELECT lesson_id FROM teach WHERE tea_id = '%s')" % (self.row_info[0]) sql3 = "DELETE FROM teacher WHERE tea_id = '%s'" % (self.row_info[0]) sql4 = "COMMIT" # SQL 删除语句 try:     cursor.execute(sql1) # 执行sql语句     cursor.execute(sql2)     cursor.execute(sql3)     cursor.execute(sql4)     db.commit() # 提交到数据库执行     messagebox.showinfo('提示!', '删除成功!')</pre> |                              |
| 程序演示<br>(4分)     | <div>教师信息：</div> <div>职工号：1</div> <div>姓名：张三</div> <div>职称：教授</div> <div>薪水：20000</div>   |                              |

### 课程信息:

课程代码: 015  
课程名称: 医学实验  
学分: 1  
课时: 1

### 授课信息:

教师职工号: 1  
课程代码: 015  
教室: A32  
结课方式: 无

### 教师信息:

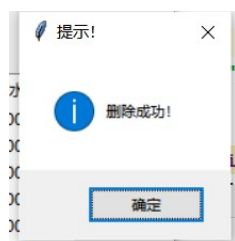
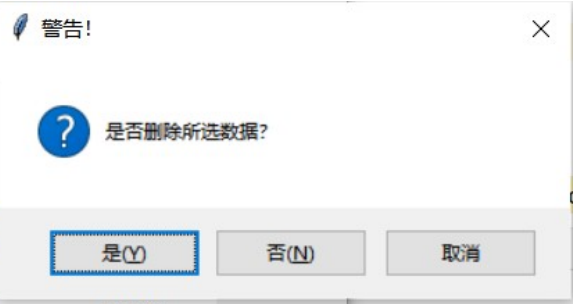
职工号: 1  
姓名: 张三  
职称: 教授  
薪水: 20000

### 操作:

新建教师信息

更新选中教师信息

删除选中教师信息





|    |  |
|----|--|
| 备注 | 因为授课信息表 teach 中的 tea_id 是外键，参照 teacher 表中的 tea_id，而且我在设置外键时设置的是 CASCADE，级联删除，所以不用额外写删除 teach 中相关记录的 SQL 语句，只写删除 lesson 中的就可以了。 |
|----|--|

## 5. 触发器控制下的添加操作（20 分）

|                |   |                 |
|----------------|---|-----------------|
| 说明             | (1 分) 简要说明该操作所要完成的功能；<br>(2 分) 简要说明该触发器所要完成的功能<br>(1 分) 该操作会涉及的表（以“表名”的形式给出）。<br>(2 分) 该操作输入数据以及输入数据应该满足的条件，如：数值范围、是否为空；<br>(6 分) 实现该操作的关键代码（高级语言、SQL），截图即可；<br>(8 分) 如何执行该操作，按所述方法能够正常演示程序则给分。 |                 |
| 功能描述<br>(1 分)  | 添加新的课程信息时，课时应不超过 17h，否则会提示课时太多，学不完  |                 |
| 触发器描述<br>(2 分) | 每次向 lesson 表中插入新记录之前，先检查插入的课程信息的课时是否超过 17 小时，若超过，则不能够插入，报错；若没有超过，则插入成功。   |                 |
| 涉及的表<br>(1 分)  | lesson  |                 |
| 输入数据<br>(2 分)  | 字段  | 规则              |
|                | class_hour  | 课时，小于 17h，可以插入。 |
|                | lesson_id   | 课程代码，此触发器下无要求   |
|                | lesson_name   | 课程名称，此触发器下无要求   |
|                | credit  | 学分，此触发器下无要求     |

|                |   |
|----------------|---|
| 插入操作源码<br>(3分) | <pre>def new_row(self):     print('123')     print(self.var_id.get())     print(self.id)     if str(self.var_id.get()) in self.id:         messagebox.showinfo('警告!', '该课号已被占用!')     else:         if self.var_id.get() != '' and self.var_name.get() != '' and self.var_cre.get() != '':             # 打开数据库连接             db = pymysql.connect(host="localhost", user="root", password="jzk1314.", database="school", port=3306)             cursor = db.cursor() # 使用cursor()方法获取操作游标             if(self.var_hour!=''):                 # SQL 插入语句                 sql = "INSERT INTO lesson(lesson_id, lesson_name, credit, class_hour) \ VALUES ('%s', '%s', '%s', '%s')"% \ (self.var_id.get(), self.var_name.get(), self.var_cre.get(), self.var_hour.get())             else:                 sql = "INSERT INTO lesson(lesson_id, lesson_name, credit, class_hour) \ VALUES ('%s', '%s', '%s', '尚未设置学时')"% \ (self.var_id.get(), self.var_name.get(), self.var_cre.get())              try:                 cursor.execute(sql) # 执行sql语句                 db.commit() # 提交到数据库执行                 messagebox.showinfo('提示!', '插入成功!')                 self.id.append(self.var_id.get())                 self.name.append(self.var_name.get())                 self.cre.append(self.var_cre.get())                 self.hour.append(self.var_hour.get())                 self.tree.insert('', len(self.id) - 1, values=(                     self.id[len(self.id) - 1], self.name[len(self.id) - 1], self.cre[len(self.id) - 1],                     self.hour[len(self.id) - 1]))                 self.tree.update()             except:                 db.rollback() # 发生错误时回滚                 messagebox.showinfo('警告!', '课时太多了!学不完!')             db.close() # 关闭数据库连接          else:             messagebox.showinfo('警告!', '请填写必要课程信息')</pre> |
| 触发器源码<br>(3分)  | <pre>1 DELIMITER // 2 CREATE TRIGGER hour_limit 3 BEFORE INSERT ON lesson FOR EACH ROW 4 BEGIN 5 IF NEW.class_hour &gt; 17 THEN 6 SIGNAL SQLSTATE '45000'; 7 END IF; 8 END; // 9 DELIMITER ; </pre>   |
| 程序演示<br>(4分)   | <div><div>课程信息：</div><div><div>课程代码：015</div><div>课程名称：爱情社会学啊</div><div>学分：1</div><div>课时：16</div></div><div>操作：</div><div><div>新建课程信息</div><div>更新选中课程信息</div><div>删除选中课程信息</div></div></div> <div><div>提示!</div><div>插入成功!</div><div>确定</div></div>   |

|              |   |
|--------------|---|
|              | <div><div>015</div><div>爱情社会学啊</div><div>1</div><div>16</div></div>   |
| 程序演示<br>(4分) | <div><div><div>课程信息：</div><div><div>课程代码：016</div><div>课程名称：爱情社会学呀</div><div>学分：1</div><div>课时：18</div></div></div><div><div>操作：</div><div><div>新建课程信息</div><div>更新选中课程信息</div><div>删除选中课程信息</div></div></div></div> <div><div>警告！</div><div><div>i</div><div>课时太多了！学不完！</div></div><div>确定</div></div> |
| 备注           | <p>这里课程 id 为主键，所以添加新数据时要保持 id 唯一性，若新建的课程的 id 是重复的，则会首先报错“该课号已被占用！”。</p> <div><div>警告！</div><div><div>i</div><div>该课号已被占用！</div></div><div>确定</div></div>  |

## 6. 存储过程控制下的更新操作（18分）

|                  |   |                         |
|------------------|---|-------------------------|
| 说明               | （1分）简要说明该操作所要完成的功能；<br>（1分）简要说明该存储过程所要完成的功能；<br>（2分）说明该操作涉及操作的表（必须包含两张或两张以上的关系表，以“表名形式”描述）<br>（1分）表连接涉及字段描述（描述方式为“表 1. 属性=表 2. 属性”）<br>（2分）该操作会修改字段（以“表名. 字段名”的形式给出），以及修改规则，如新数值的计算方法、在何种条件下予以修改等；<br>（6分）实现该操作的关键代码（高级语言、SQL），截图即可；<br>（5分）如何执行该操作，按所述方法能够正常演示程序则给分。 |                         |
| 功能描述<br>（1分）     | 在更改授课信息表 teach 时，若更改了课程代码，则会同步更新课程信息表 lesson 中对应的课程代码（更改老师的授课信息，老师不变，所以教师职工号不能改变）   |                         |
| 存储过程功能描述<br>（1分） | 每次更新授课信息表时，若课程代码被更新，则先到 lesson 表中更新该课程代码（由于是级联更新，此时授课信息表的课程代码也被更新了），再更新授课信息表中的其他属性即可。（若更新为已存在的授课信息，即（tea_id, lesson_id）元组重复，则会报错）   |                         |
| 涉及的关系表<br>（2分）   | teach、lesson  |                         |
| 表连接涉及字段<br>（1分）  | teach.lesson_id = lesson.lesson_id  |                         |
| 更改字段<br>（2分）     | 字段  | 规则                      |
|                  | lesson_id   | 授课信息表中的课程代码，等于输入框中输入值   |
|                  | end_way   | 等于输入框中输入值               |
|                  | class   | 等于输入框中输入值               |
|                  | tea_id  | 要更改授课信息的教师职工号（不变）       |
|                  | lesson_id（lesson 表中）  | 等于要更改的授课信息的课程代码，将其更改为新值 |

|                                  |   |
|----------------------------------|---|
| 更新<br>代码<br>(3<br>分)             | <pre> def updata_row(self):     res = messagebox.askyesnocancel('警告!', '是否更新所填数据?')     if res == True:         if self.var_id.get() == self.row_info[0]: # 如果所填学号与所选学号一致             # 打开数据库连接             db = pymysql.connect(host="localhost", user="root", password="jzk1314.", database="school", port=3306)             cursor = db.cursor() # 使用cursor()方法获取操作游标             # SQL 插入语句             sql = "call updata_teach('%s', '%s', '%s', '%s', '%s') % \\"                 (self.var_les.get(), self.var_scr.get(), self.var_cla.get(), self.row_info[0], self.row_info[1])             try:                 cursor.execute(sql) # 执行sql语句                 db.commit() # 提交到数据库执行                 messagebox.showinfo('提示!', '更新成功!')                 id_index = self.id.index(self.row_info[0])                 self.les[id_index] = self.var_les.get()                 self.cla[id_index] = self.var_cla.get()                 self.scr[id_index] = self.var_scr.get()                  self.tree.item(self.tree.selection()[0], values=(                     self.var_id.get(), self.var_les.get(), self.var_cla.get(),                     self.var_scr.get())) # 修改对于行信息             except:                 db.rollback() # 发生错误时回滚                 messagebox.showinfo('警告!', '该授课信息已存在!')                 db.close() # 关闭数据库连接          else:             messagebox.showinfo('警告!', '不能修改教师职工号!') </pre> |
| 创建<br>存储<br>过程<br>源码<br>(3<br>分) | <pre> 1 delimiter // 2 CREATE PROCEDURE updata_teach(in l varchar(20), in e varchar(255), in c varchar(255), in i varchar(20), in le varchar(20)) 3 BEGIN 4     UPDATE lesson SET lesson_id = l WHERE lesson_id = le; 5     UPDATE teach SET end_way = e, class = c WHERE tea_id = i; 6 END 7 // 8 delimiter ; </pre>   |
| 存储<br>过程<br>执行<br>源码<br>(1<br>分) | <pre> # 打开数据库连接 db = pymysql.connect(host="localhost", user="root", password="jzk1314.", database="school", port=3306) cursor = db.cursor() # 使用cursor()方法获取操作游标 # SQL 插入语句 sql = "call updata_teach('%s', '%s', '%s', '%s', '%s') % \\"     (self.var_les.get(), self.var_scr.get(), self.var_cla.get(), self.row_info[0], self.row_info[1]) try:     cursor.execute(sql) # 执行sql语句     db.commit() # 提交到数据库执行     messagebox.showinfo('提示!', '更新成功!') </pre>  |
| 程序<br>演示<br>(2<br>分)             | <div> <div>013</div> <div>吉他鉴赏与演奏</div> <div>1</div> <div>9</div> </div>  |

授课信息：

|        |       |
|--------|-------|
| 教师职工号： | 05013 |
| 课程代码：  | 013   |
| 教室：    | P0    |
| 结课方式：  | 平时成绩  |

授课信息：

|        |       |
|--------|-------|
| 教师职工号： | 05013 |
| 课程代码：  | 015   |
| 教室：    | P0    |
| 结课方式：  | 平时成绩  |

操作：

新建授课信息

更新选中授课信息

删除选中授课信息

提示!

更新成功!

确定

授课信息表：

|       |     |    |      |
|-------|-----|----|------|
| 05013 | 015 | P0 | 平时成绩 |
|-------|-----|----|------|

课程信息表：

|     |         |   |   |
|-----|---------|---|---|
| 015 | 吉他鉴赏与演奏 | 1 | 9 |
|-----|---------|---|---|

程序  
演示  
(2  
分)

授课信息:

教师职工号:01001

课程代码:001

教室:A1

结课方式:闭卷

操作:

新建授课信息

更新选中授课信息

删除选中授课信息

| 教师职工号 | 课程代码 | 教室 | 结课方式 |
|-------|------|----|------|
| 01001 | 001  | A1 | 闭卷   |
| 01001 | 012  | A1 | 闭卷   |

授课信息:

教师职工号:01001

课程代码:012

教室:A1

结课方式:闭卷

操作:

新建授课信息

更新选中授课信息

删除选中授课信息

| 教师职工号 | 课程代码 | 教室 | 结课方式 |
|-------|------|----|------|
| 01001 | 001  | A1 | 闭卷   |
| 01001 | 012  | A1 | 闭卷   |

警告!

i

该授课信息已存在!

确定

备注

之前由于 teach 表中的 lesson\_id 是外键,参照于 lesson 表的 lesson\_id,所以之前在 teach 表中不能修改 lesson 中未出现过的 lesson\_id 值;但增加这个存储过程之后,每次更新 teach 表之前,都先更新 lesson,保证 lesson 中有想要的 lesson\_id 值。报错顺序:若更改了教师职工号,则报错;若更新信息为已存在的授课信息,则报错;反之,正常更新两个表。



## 7. 含有视图的查询操作（15 分）

|              |   |
|--------------|---|
| 说明           | (1 分) 简要说明该操作所要完成的功能;<br>(1 分) 简要说明建立的该视图的功能;<br>(2 分) 简要说明该操作涉及的关系数据表 (以 “表名” 的形式给出)<br>(1 分) 简要说明表连接涉及的字段 (以 “表 1. 属性=表 2. 属性”)<br>(6 分) 实现该操作的关键代码 (高级语言、SQL), 截图即可;<br>(4 分) 如何执行该操作, 按所述方法能够正常演示程序则给分。 |
| 操作功能描述 (1 分) | 查看学生表中的学生学号、姓名, 以及在选课表中该学生所选课程的课程和取得的成绩。  |
| 视图功能描述 (1 分) | 除去不关心的信息, 如学生入学年份、学院信息、课程学分、课时等, 只保留想要的信息, 即哪些学生选了哪些课, 得了多少分。   |
| 涉及的关系表 (2 分) | student、learn   |
| 表连接字段 (1 分)  | student.stu_id = learn.stu_id   |
| 创建视图代码       | <pre> 1 CREATE VIEW stu_les AS 2 SELECT student.stu_id, stu_name, lesson_id, score 3 FROM student, learn 4 WHERE student.stu_id = learn.stu_id; </pre>  |

|                              |  |
|------------------------------|--|
| 码<br>( 3<br>分)               |  |
| 查<br>询<br>代<br>码<br>(3<br>分) | <pre> self.id = [] self.name = [] self.gender = [] self.college = []  # 打开数据库连接 db = pymysql.connect(host="localhost", user="root", password="jzk1314.", database="school", port=3306) cursor = db.cursor() # 使用cursor()方法获取操作游标 sql = "SELECT * FROM stu_les" # SQL 查询语句 try:     # 执行SQL语句     cursor.execute(sql)     # 获取所有记录列表     results = cursor.fetchall()     for row in results:         self.id.append(row[0])         self.name.append(row[1])         self.gender.append(row[2])         self.college.append(row[3]) except:     print("Error: unable to fetch data")     messagebox.showinfo('警告!', '数据库连接失败!') db.close() # 关闭数据库连接  # 写入数据 for i in range(min(len(self.id), len(self.name), len(self.gender), len(self.college))):     self.tree.insert('', i, values=(self.id[i], self.name[i], self.gender[i], self.college[i])) </pre> |

程序演示  
(4分)



| 学号    | 姓名 | 课程代码 | 得分   |
|-------|----|------|------|
| 00003 | 刘二 | 005  | 88.0 |
| 00003 | 刘二 | 009  | 94.0 |
| 00001 | 张三 | 001  | 96.0 |
| 00001 | 张三 | 002  | 92.0 |
| 00001 | 张三 | 003  | 95.0 |
| 00001 | 张三 | 009  | None |
| 98725 | 张蛋 | 008  | 98.0 |
| 98725 | 张蛋 | 015  | 77.0 |
| 00002 | 李凤 | 004  | 99.0 |
| 00005 | 王麻 | 007  | 91.0 |
| 00004 | 赵四 | 006  | 87.0 |

备注

学生表（学号，姓名，性别，入学时间，学院）中取出学号和姓名；  
选课表（学生学号，课程代码，得分，教室号）中取出课程代码和得分；  
用学号连接，创建视图并可视化。

