

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

CS33503 数据库系统实验

实验检查记录

实验结果的正确性 (60%)		表达能力 (10%)	
实验过程的规范性 (10%)		实验报告 (20%)	
加分 (5%)		总成绩 (100%)	

实验报告

一、实验目的（介绍实验目的）

1. 熟悉 SQL 语言，熟练掌握关系数据库系统的使用
2. 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
3. 能够向数据库中添加一定规模的数据
4. 能够对数据库性能进行简单的评估与优化

二、实验环境（介绍实验使用的硬件设备、软件系统、开发工具等）

本实验采用 MySQL 关系数据库，使用 Python 语言进行开发。用户界面部分则使用 PyQt5 进行生成，在测试、数据导入、可视化过程中，使用了 Navicat 数据库管理工具。

三、实验过程（介绍实验过程、设计方案、实现方法、实验结果等）

1. 实验内容概述

本次实验中，我完成了一个用于学校管理的综合性系统，系统包括老师，学生，教室等实体，可以实现成绩录入，显示课程排班，教学任务安排等功能。在完成数据库的概念性设计后，我使用 PyQt5 完成了用户界面的设计，并使其能够与数据库进行交互。

2. 实验内容 1：数据库设计

2.1 需求分析

预想的该系统具有的功能如下：

1. 学生选课
2. 学生成绩录入
3. 学生成绩查询
4. 课程成绩查询
5. 教学任务安排
6. 教室安排

根据这些功能，需要设计以下 9 个实体，分别是：院系，班级，学生，老师，成绩，课

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

程，课程的教学任务，课程的教室安排，教室。关于它们的具体定义以及约束以表的方式列举如下：

表 1: Student

列名	数据类型	完整性约束	含义
s_id	char(9)	主键	学号
cl_id	char(6)	外键，NOT NULL	班号
s_name	varchar(12)	NOT NULL	学生姓名
s_sex	char(2)	NOT NULL	学生性别
s_age	int	NOT NULL	学生年龄
s_tel	char(11)	NOT NULL	学生联系方式

表 2: Teacher

列名	数据类型	完整性约束	含义
t_id	char(8)	主键	教师工号
d_id	char(4)	外键，NOT NULL	专业代码
t_name	varchar(12)	NOT NULL	教师姓名
t_sex	char(2)	NOT NULL	教师性别
t_age	int	NOT NULL	教师年龄
t_tel	char(11)	NOT NULL	教师联系方式

表 3: Department

列名	数据类型	完整性约束	含义
d_id	char(4)	主键	专业代码
d_name	varchar(16)	NOT NULL	专业名称

表 4: Class

列名	数据类型	完整性约束	含义
cl_id	char(6)	主键	班号
t_id	char(8)	NOT NULL	班主任工号
d_id	char(4)	外键，NOT NULL	专业代码

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

表 5: Course

列名	数据类型	完整性约束	含义
c_id	char(6)	主键	课程号
c_name	varchar(20)	NOT NULL	课程名称
d_id	char(4)	外键, NOT NULL	开课专业
credit	float	NOT NULL	课程学分
hours	int	NOT NULL	课程学时

表 6: Classroom

列名	数据类型	完整性约束	含义
r_id	varchar(8)	主键	教室代号
r_addr	varchar(20)	NOT NULL	教室地点

表 7: sc (StudentCourse)

列名	数据类型	完整性约束	含义
s_id	char(9)	主键, 外键	学号
c_id	char(6)	主键, 外键	课程号
score	float	NOT NULL	课程成绩
date	date	NOT NULL	成绩录入时间

表 8: tt (TeachingTask)

列名	数据类型	完整性约束	含义
t_id	char(8)	主键, 外键	班主任工号
c_id	char(6)	主键, 外键	课程号
start_week	int		开课周数
end_week	int		结课周数

表 9: ttr (TeachingTaskClassroom)

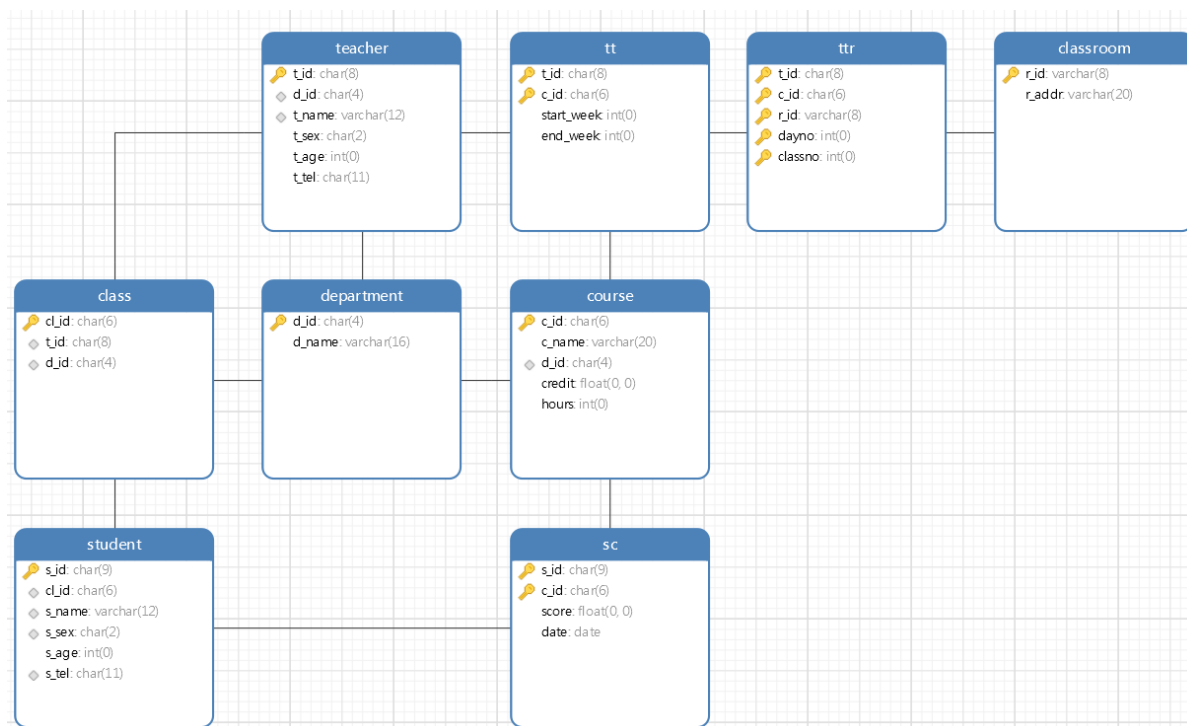
列名	数据类型	完整性约束	含义
c_id	char(6)	主键, 外键	课程号
t_id	char(8)	主键, 外键	班主任工号
r_id	varchar(8)	主键, 外键	教室代号
dayno	int	NOT NULL	一周中第几天
classno	int	NOT NULL	课程节数

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李昊翀

在我所想的设定中，包含了各种联系：班级与班主任的联系属于一对一联系，班级与学生的联系属于一对多联系，而教室与教学任务安排属于多对多联系。

2.2 系统 ER 图

由前述，整个管理系统包含 9 个实体和 10 个联系，具体的系统 ER 图如下所示：



2.3 数据库索引

按照实际经验，针对经常修改，查看的实体：Student，Teacher，ttr 分别建立了索引，以优化数据库的性能。具体而言，对 student 表中的 s_name 属性、s_sex 属性和 s_tel 属性创建了索引；对 teacher 表中的 t_name 属性创建了索引；对 ttr 表中的 classNo 属性创建了索引。

在 Navicat 中查看创建好的索引，如图所示：

名	字段	索引类型	索引方法	注释
▶ s_clid_idx	'cl_id'	NORMAL	BTREE	
idx_stu_name	's_name'	NORMAL	BTREE	
idx_stu_tel	's_tel'	NORMAL	BTREE	
idx_stu_sex	's_sex'	NORMAL	BTREE	

2.4 数据库外模式

在该管理系统中，创建视图（外模式）主要是满足成绩查询等功能的要求，同时在查询过程中方便得到复杂的统计结果（如平均成绩等）。目前，我为该管理系统设计了如下三个视图：

1. studStat

包括如下信息：学生学号，学生姓名，以及该学生目前修读的总学分，目前的平均学分

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

绩。该视图只用于查询，其构造语句如图所示：

```
create view studStat (s_id, s_name, total_credits, avg_score) as
(select A.s_id, A.s_name, sum(credit), sum(total)/sum(credit) from
(select s.s_id, s.s_name, credit * score total, credit from sc , student s , course c
where sc.s_id = s.s_id and sc.c_id = c.c_id and sc.score is not null ) as A group by A.s_id);
```

2. studNoScoreCourse

包括如下信息：学生学号，学生姓名，目前暂无成绩的课程门数，没出成绩的课程总学分，该视图只用于查询，其构造语句如图所示：

```
create view studNoScoreCourse (s_id, s_name, credits_no_score, courseNum) as
(select A.s_id, A.s_name, sum (credit), count (*) from
(select s.s_id, s.s_name, credit from sc, student s, course c
where sc.s_id = s.s_id and sc.c_id = c.c_id and sc.score is null) as A group by A.s_id );
```

3. courseStat

包括如下信息：课程的课程号、课程名称，选修该课程的学生成绩的最高分、最低分以及平均分，该视图只用于查询，其构造语句如图所示：

```
create view courseStat (c_id, c_name, max_score, min_score, avg_score) as
(select c.c_id, c_name, max (score), min (score), avg (score) from course c, sc
where c.c_id = sc.c_id and sc.score is not null group by c.c_id );
```

2.5 批量数据生成与添加

为了测试数据库性能，需要向数据库中注入一定规模的数据。由于在现实场景中，学生和成绩这两个实体的数量往往多于其余实体的数量，因此在本实验中，我通过 Python 程序分别生成了 1000 个 Student 和 Score 实体，将其保存在.xls 文件中，在需要时可以写入数据库，便于后续进行数据库的性能测试。

生成的数据格式如下：

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

A	B	C	D	E	F
s_id	cl_id	s_name	s_sex	s_age	s_tel
100403000	100403	TEST0	女	21	13000000000
100403001	100403	TEST1	女	18	13000000001
100403002	100403	TEST2	男	19	13000000002
100403003	100403	TEST3	女	23	13000000003
100403004	100403	TEST4	女	22	13000000004
100403005	100403	TEST5	男	19	13000000005
100403006	100403	TEST6	女	21	13000000006
100403007	100403	TEST7	男	22	13000000007
100403008	100403	TEST8	男	20	13000000008
100403009	100403	TEST9	女	22	13000000009
100403010	100403	TEST10	男	20	13000000010
100403011	100403	TEST11	男	19	13000000011
100403012	100403	TEST12	女	23	13000000012
100403013	100403	TEST13	男	20	13000000013
100403014	100403	TEST14	女	19	13000000014
100403015	100403	TEST15	女	23	13000000015
100403016	100403	TEST16	男	22	13000000016
100403017	100403	TEST17	男	23	13000000017
100403018	100403	TEST18	女	19	13000000018
100403019	100403	TEST19	男	21	13000000019
100403020	100403	TEST20	男	21	13000000020
100403021	100403	TEST21	女	18	13000000021

图 1：生成的 Student 实体

A	B	C	D
s_id	c_id	score	date
100403000	130001	51	2022-01-01
100403001	130001	18	2022-01-01
100403002	130001	86	2022-01-01
100403003	130001	98	2022-01-01
100403004	130001	33	2022-01-01
100403005	130001	87	2022-01-01
100403006	130001	1	2022-01-01
100403007	130001	21	2022-01-01
100403008	130001	48	2022-01-01
100403009	130001	18	2022-01-01
100403010	130001	66	2022-01-01
100403011	130001	36	2022-01-01
100403012	130001	18	2022-01-01
100403013	130001	56	2022-01-01
100403014	130001	37	2022-01-01
100403015	130001	93	2022-01-01
100403016	130001	41	2022-01-01
100403017	130001	36	2022-01-01
100403018	130001	94	2022-01-01

图 2：生成的 sc 实体

在生成完毕后，可以通过 Navicat 数据库管理软件进行数据写入。具体流程如下：

导入从:

表: ☐ student ☒ sc

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

字段名行:

第一个数据行:

最后一个数据行:

格式

日期排序: YMD

日期分隔符: -

时间分隔符: :

小数点符号: .

日期时间排序: 日期 时间

源表	目标表	新建表
student	student	<input type="checkbox"/>
sc	sc	<input type="checkbox"/>

导入向导

我们已收集向导导入数据时所需的全部信息。点击 [开始] 按钮进行导入。

表: 2
已处理: 2,000
错误: 0
已添加: 2,000
已更新: 0
已删除: 0
时间: 00:00.55

[IMP] Import start
[IMP] Import type - Excel file
[IMP] Import from - C:\UserData\Desktop\数据库Lab3\Lab3\gen_data.xls
[IMP] Import table [student]
[IMP] Import table [sc]
[IMP] Processed: 2000, Added: 2000, Updated: 0, Deleted: 0, Errors: 0
[IMP] Finished successfully

保存 日志 << < 上一步 开始 关闭

导入完成后，效果如下：

s_id	cl_id	s_name	s_sex	s_age	s_tel
100101001	100101	张三	男	18	14648466354
100101002	100101	张四	男	21	12565925663
100101003	100101	张五	男	19	13511245565
100101004	100101	张六	男	18	16545959266
100102001	100102	李一	女	19	18945663574
100102002	100102	李二	女	21	13349568558
100102003	100102	李三	女	19	13112882293
100202002	100202	李四	女	22	12300005656
100202004	100202	李五	男	20	16546599990
100403000	100403	TEST0	女	21	13000000000
100403001	100403	TEST1	女	18	13000000001
100403002	100403	TEST2	男	19	13000000002
100403003	100403	TEST3	女	23	13000000003
100403004	100403	TEST4	女	22	13000000004
100403005	100403	TEST5	男	19	13000000005
100403006	100403	TEST6	女	21	13000000006
100403007	100403	TEST7	男	22	13000000007
100403008	100403	TEST8	男	20	13000000008
100403009	100403	TEST9	女	22	13000000009
100403010	100403	TEST10	男	20	13000000010
100403011	100403	TEST11	男	19	13000000011
100403012	100403	TEST12	女	23	13000000012
100403013	100403	TEST13	男	20	13000000013
100403014	100403	TEST14	女	19	13000000014
100403015	100403	TEST15	女	23	13000000015
100403016	100403	TEST16	男	22	13000000016

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李昊翀

3. 数据库应用开发

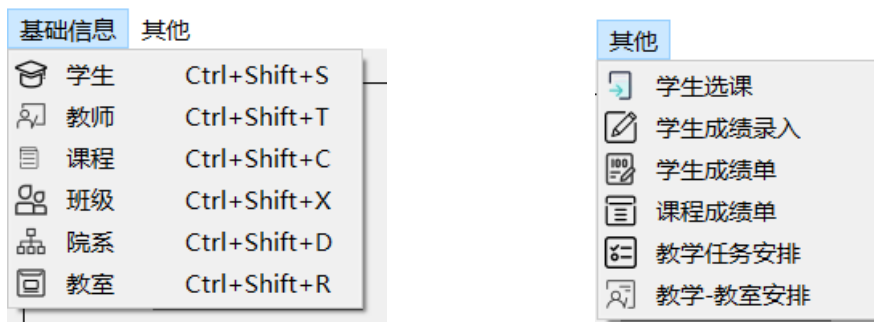
3.1 应用界面简介

按照前述的思想使用 PyQt5 设计管理系统，完成后得到 sms.ui 文件。使用 pyuic5 将 .ui 文件转化为 .py 文件，再另外编写一个 main 函数即可启动程序界面。

下面按顺序来介绍应用界面的每个部分。

1. 菜单栏

菜单栏可以选择具体执行操作的两部分，分别是基础信息部分和其他功能部分。如图所示。



2. 基础信息部分

在打开程序时，系统默认进入的部分也就是学生的基础信息部分。如图所示（默认表部分内容为空，需要在 SELECT 操作下点击“确认”才能显示数据内容）：

	学号	班级	姓名	性别	年龄	电话
1	100101001	100101	张三	男	18	14648466354
2	100101002	100101	张四	男	21	12565925663
3	100101003	100101	张五	男	19	13511245565
4	100101004	100101	张六	男	18	16545959266
5	100102001	100102	李一	女	19	18945663574
6	100102002	100102	李二	女	21	13349568558
7	100102003	100102	李三	女	19	13112882293
8	100202002	100202	李四	女	22	12300005656
9	100202004	100202	李五	男	20	16546599990
10	100403000	100403	TEST0	女	21	13000000000
11	100403001	100403	TEST1	女	18	13000000001
12	100403002	100403	TEST2	男	19	13000000002

基础信息部分主要完成对实体数据的查看，另外我完成了对实体的增，删，查三种功能的支持。如上图所示，可以在对应的框中输入数据，以便完成增，删，查

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

三种功能。具体如何执行将在 3.2 部分详细阐述。

3. 其他功能部分

其他部分包含了完成五种不同功能的模块：学生选课，学生成绩录入，显示学生成绩，显示某门课程的所有学生成绩，以及教学任务安排和教室安排。其界面如图所示（以教室安排为例）：

	教师编号	教师姓名	课程编号	课程名	星期	节次	教室
1	10011045	郑一	140001	大数据算法	1	1	正心5楼10号
2	10020605	何一	120004	计算机视觉导论	2	1	正心3楼12号
3	10021012	何二	110001	高级算法设计	3	1	正心5楼10号
4	10021012	何二	120004	计算机视觉导论	2	1	诚意5楼1号
5	10011045	郑一	130001	信息检索	1	2	正心1楼1号
6	10011045	郑一	130001	信息检索	2	2	正心1楼1号
7	10011045	郑一	140001	大数据算法	3	2	正心5楼10号
8	10021012	何二	110001	高级算法设计	1	2	正心5楼10号
9	10021020	何三	110001	高级算法设计	3	2	正心3楼12号
10	10021020	何三	120003	人工智能导论	4	2	正心5楼10号
11	10021065	何四	120002	模式识别与机...	1	2	致知4楼1号
12	10021065	何四	120002	模式识别与机...	2	2	致知4楼1号

关于该部分的具体功能演示，放在 3.2 部分详细阐述。

3.2 基本功能与功能测试

以下具体讲解该系统实现的功能。

1. 基本查询功能

用户在界面中选择进行 SELECT 操作，在输入任意查询条件的组合后，系统后端会根据按键反馈，组合成相应的 SQL 查询语句，执行后，系统返回 MySQL 数据库中满足条件的全部元组，同时在 Python 后端中打印输出用户所执行的操作，如图所示，用户在教师页面选中 SELECT 操作并点击确认，前端与后端界面的显示如下：

```
select * from teacher;
```

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

教师信息 操作: SELECT

教师编号: 姓名:

系号: 性别:

确认 清除

	教师编号	系号	姓名	性别	年龄	电话
1	10011045	1001	郑一	男	35	12546825462
2	10012005	1001	郑二	女	42	15678912356
3	10016215	1001	郑三	女	29	16846262625
4	10020605	1002	何一	男	56	17582452555
5	10021012	1002	何二	女	45	11645021234
6	10021020	1002	何三	男	50	13984505856
7	10021065	1002	何四	男	38	19462632546
8	10030608	1003	毛一	男	65	14654626599
9	10035075	1003	毛二	男	36	16598783423
10	10040106	1004	林一	女	32	19563552215
11	10042210	1004	林二	男	52	17256658522
12	20010102	2001	邓一	男	60	16548596231

2. 模糊查询

在基本的查询功能之外，用户也可以输入“%”表示通配符，进行模糊匹配查询。以学生信息查询为例，如图所示，学号部分输入“1001%”，系统会查询所有学号以“1001%”开头的元组并返回：

```
select * from student where s_id like '1001%';
```

学生信息 操作: SELECT

学号: 年龄: 性别:

班号: 姓名: 联系方式:

确认 清除

	学号	班级	姓名	性别	年龄	电话
1	100101001	100101	张三	男	18	14648466354
2	100101002	100101	张四	男	21	12565925663
3	100101003	100101	张五	男	19	13511245565
4	100101004	100101	张六	男	18	16545959266
5	100102001	100102	李一	女	19	18945663574
6	100102002	100102	李二	女	21	13349568558
7	100102003	100102	李三	女	19	13112882293

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

当输入的信息在表中没有与其匹配的元组时，系统则会提示报错：

The screenshot shows a software window titled 'MainWindow' with a tab labeled '基础信息 其他'. The main content is a form titled '学生信息' (Student Information). The form includes a '操作' (Action) dropdown menu currently set to 'SELECT'. There are input fields for '学号' (Student ID) with value '1001W', '年龄' (Age), '性别' (Gender), '班号' (Class ID) with value '200102', '姓名' (Name), and '联系方式' (Contact Info). To the right of the form are '确认' (Confirm) and '清除' (Clear) buttons. Below the form is a table with 7 rows of student data. A modal dialog box is overlaid on the table, displaying the error message: 'QUERY Wrong 1054 Unknown column 's_class' in 'where clause''. The dialog has an 'OK' button.

	学号	班级	姓名	性别	年龄	电话
1	100101001	100101				14648466354
2	100101002	100101				12565925663
3	100101003	100101				13511245565
4	100101004	100101				16545959266
5	100102001	100102	李一	女	19	18945663574
6	100102002	100102	李二	女	21	13349568558
7	100102003	100102	李三	女	19	13112882293

另外，在点击“清除”按钮后，可以清除该页面所有输入内容和显示区内容。

3. 插入（增加）功能

插入功能允许用户向表中插入一行数据，插入的数据需满足数据库的完整性约束，如主键约束、外键约束（不能插入外键表中不存在的内容）、非空约束（不允许为空的属性都不能插入为空）等等，若插入的数据不满足完整性约束，将弹出提示信息。

在本系统中，插入功能一般要求所有输入框或者选择框都有相应的内容，（因为大部分属性都有 NOT NULL 的约束）。以教学任务安排表 tt 为例，选择教师编号“10012005”和课程号“110001”，设置开课星期为 4，结课星期为 12，点击确认，提示插入成功，如下图所示。

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

MainWindow

基础信息 其他

课程信息

操作: INSERT

课程编号: 110009 课程名称: 测试课程

开课院系号: 1001 学分: 3.0 学时: 24

确认 清除

	课程编号	课程名	开课系号	学分	学时
1	110001	高级算法设计		3.0	24
2	110004	计算建模		3.0	24
3	120002	模式识别与机器学习		3.0	24
4	120003	人工智能导论		4.5	36
5	120004	计算机视觉导论	1002	4.5	36
6	130001	信息检索	1003	3.0	24
7	140001	大数据算法	1004	4.5	36
8	210020	基因生物学	2001	5.0	40
9	220010	系统开发	2002	5.0	40

Success X
Insert Success!
OK

改而选中 SELECT 进行查询，同样能查到对应的结果。

MainWindow

基础信息 其他

课程信息

操作: SELECT

课程编号: 110009 课程名称: 测试课程

开课院系号: 1001 学分: 3.0 学时: 24

确认 清除

	课程编号	课程名	开课系号	学分	学时
1	110009	测试课程	1001	3.0	24

若是不满足插入条件，如：某一栏为空，则会弹窗显示错误提示，如图所示：

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

The screenshot shows a software interface for managing course information. The main window is titled 'MainWindow' and has tabs for '基础信息' (Basic Information) and '其他' (Other). The '课程信息' (Course Information) section contains several input fields: '课程编号' (Course ID) with value '110010', '课程名称' (Course Name) with value '测试课程2', '开课系号' (Department ID), '学分' (Credits), and '学时' (Hours). A dropdown menu for '操作' (Action) is set to 'INSERT'. To the right are '确认' (Confirm) and '清除' (Clear) buttons. Below the form is a table with columns: '课程编号', '课程名', '开课系号', '学分', and '学时'. The table contains one row with values: '1', '110009', '测试课程', and '24'. A modal dialog box titled 'INSERT Wrong 1265' is open, displaying the error message 'Data truncated for column 'credit' at row 1' and an 'OK' button.

另外，为了部分地解决插入过程中可能出现的问题，同时提升程序的用户友好度，我在部分页面将输入框更改为了选择框，这样就有效地满足了外键约束（不会出现没有的选项），例如在教学任务安排的界面中，课程号和教师编号就是由选择框直接选择，而不是由用户输入，这样可以有效地降低输入错误率。

在代码方面，插入的代码会相对复杂一些。与查询和删除不一样，插入过程中要求每一项都是确认的，要罗列插入值，所以查询中的 where 条件查询函数是不可复用的，因此对于每一种插入，都西药重新书写 SQL 插入语句的生成函数。以 insert_tt_sql() 函数为例，当判断 index = 2，即需要进行插入操作时，执行插入函数 insert()，在每次执行该 sql 语句之后都需要进行“commit”操作。其具体的代码如下：

```
def insert_tt_sql(self):
    insert_tt_sql = "insert into tt (t_id, c_id, start_week, end_week) values ("
    tid_text = self.comboBox_ttid.currentText()
    cid_text = self.comboBox_ttcid.currentText()
    start_week_text = self.lineEdit_startweek.text().strip()
    end_week_text = self.lineEdit_endweek.text().strip()
    insert_value = "''" + tid_text + "''," + cid_text + "''," + start_week_text + "''," + end_week_text + "''"
    insert_value.replace("''", "null")
    insert_tt_sql += insert_value + ");"
    return insert_tt_sql
```

4. 删除功能

与查询相似，用户可以通过完整的属性值来删除某一特定元组，也可以只输入部分属性的值来删除所有满足该条件的元组（包括模糊匹配功能）。因此，此处删除所使用的 where 条件可以复用查询的 where 条件。若用户的删除操作破坏了完整性约束，则系统会报错，主要体现在外键约束带来的错误，例如当某个教室被安排了教学任务删除该教室就会报错。对于删除操作，在用户选择执行

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

时，首先会提示确认是否要删除，若选择 Yes，则运行删除 SQL 语句，并检测用户想要执行的删除是否能够正确执行。若是不满足外键约束，则弹出错误提示。如图所示：

MainWindow
基础信息 其他

课程信息

操作: DELETE

课程编号: 112345 课程名称: 测试课程

开课院系号: 1001 学分: 3.0 学时: 24

确认 清除

	课程编号	课程名	开课系号	学分	学时
1	110001	高级算法			24
2	110004	计算建模			24
3	112345	测试课程			24
4	120002	模式识别			24
5	120003	人工智能导论	1002	4.5	36
6	120004	计算机视觉导论	1002	4.5	36
7	130001	信息检索	1003	3.0	24
8	140001	大数据算法	1004	4.5	36
9	210020	基因生物学	2001	5.0	40
10	220010	系统开发	2002	5.0	40

Success
Delete success! The left data as below.
OK

MainWindow
基础信息 其他

教学任务安排

操作: DELETE

课程号: 110004 教师编号: 10030608

开课星期: 2 结束星期: 7

确认 清除

	教师编号	教师姓名	课程编号	课程名	开始星期	结束星期
5	10012005					5
6	10016215					6
7	10016215					10
8	10016215					11
9	10020605					11
10	10021012	何二	110001	高级算法设计	2	7
11	10021012	何二	120004	计算机视觉导论	3	11
12	10021020	何三	110001	高级算法设计	2	7
13	10021020	何三	120003	人工智能导论	1	9
14	10021065	何四	120002	模式识别与机器学习	2	7
15	10030608	毛一	110004	计算建模	2	7
16	10035075	毛二	220010	系统开发	2	11

DELETE Wrong 1451
Cannot delete or update a parent row: a foreign key constraint fails ('sct'.ttr', CONSTRAINT 'ttr_ctid' FOREIGN KEY ('t_id', 'c_id') REFERENCES 'tt' ('t_id', 'c_id') ON DELETE RESTRICT ON UPDATE RESTRICT)
OK

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

5. 连接查询

连接查询其实在实验中的许多部分都有所体现，最典型的的就是相对复杂的查询功能，如学生成绩单和课程成绩统计。这两项除了需要创建视图来获取统计数据外，都需要对 sc、student、course 三个表进行连接来得到更详细的信息。这两个功能的实现代码如下图所示：

```
# student_transcript
elif self.stackedWidget.currentWidget() == self.page_stu_transcript:
    student_transcript_header = ['学号', '姓名', '课程编号', '课程名', '成绩', '课程学分']
    student_transcript_query_sql = \
        "select student.s_id, student.s_name, sc.c_id, c_name, score, credit " + \
        "from sc inner join student inner join course " + \
        "on sc.s_id = student.s_id and sc.c_id = course.c_id " + \
        self.student_transcript_where()
    self.query(student_transcript_query_sql, student_transcript_header)
# student_statistics
self.student_statistics()
```

```
# course_transcript
elif self.stackedWidget.currentWidget() == self.page_course_transcript:
    course_transcript_header = ['课程编号', '课程名', '学号', '姓名', '成绩']
    course_transcript_query_sql = \
        "select sc.c_id, c_name, student.s_id, student.s_name, score " + \
        "from sc inner join student inner join course " + \
        "on sc.s_id = student.s_id and sc.c_id = course.c_id " + \
        self.course_transcript_where()
    self.query(course_transcript_query_sql, course_transcript_header)
# course_statistics
self.course_statistics()
```

这两部分的具体运行结果如图：

学生查询成绩

☒ 是否出成绩 学号: 100101001 姓名: 张三 确认 清除

请输入正确的学号和对应的姓名，否则无法查询！！

	学号	姓名	课程编号	课程名	成绩	课程学分
1	100101001	张三	110001	高级算法设计	80.0	3.0
2	100101001	张三	130001	信息检索	48.0	3.0
3	100101001	张三	140001	大数据算法	58.0	4.5
4	100101001	张三	210020	基因生物学	78.0	5.0
5	100101001	张三	220010	系统开发	55.0	5.0

学生统计结果

该同学已出成绩的总学分为: 20.5分, 平均分绩有: 63.90243902439025

OK

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

课程成绩单

课程号: 110001 课程名:

请输入正确的课程号 (可以不输入课程名, 若输入则一定要正确)

	课程编号	课程名	学号	姓名	成绩
1	110001	高级算法设计	100101001	张三	80.0
2	110001	高级算法设计	200101002	王二	100.0
3	110001	高级算法设计	200101003	王三	80.5
4	110001	高级算法设计	200102002	丁二	None

课程统计结果

该课程最高分: 100.0分, 最低分: 80.0平均分: 86.83333333333333分.

6. 嵌套查询

嵌套查询在复杂功能中也具有很高的重要性。如, 在查询班级情况时, 为防止班级因为存在学生而被漏统计, 就将满足条件的班级元组作为表 A, 与 student 表中的班级-学生数统计结果进行外连接, 再在“班级学生数”这一项中用“0”来代替全部的 null。查询全部的班级及其相关信息, 其运行结果如图所示, 这里有 2 个班级的学生人数为 0。

班级信息 操作: SELECT

班号: 系号: 班主任编号:

	班号	班主任编号	班主任名字	班级院系	班级学生数
1	100101	10012005	郑二	计算机科学	4
2	100102	10011045	郑一	计算机科学	3
3	100201	10021065	何四	计算机视觉	0
4	100202	10021012	何二	计算机视觉	2
5	100301	10035075	毛二	自然语言处理	0
6	100403	10042210	林二	大数据	1000
7	200101	20010102	邓一	生物信息学	5
8	200102	20013050	邓二	生物信息学	2

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

7. 分组查询

如上述嵌套查询一样，对于分组查询，在实验中查询班级时，引入班级学生数的概念，首先要将学生按照班级进行划分并且用“count(*)”进行统计，具体来说就是获得相应班级的学生数。例子如嵌套查询所示。

3.4 系统优化

正如前文所述，我为比较常被修改的实体类中的特定属性增加了索引。如：student 表中的 s_name 属性、s_sex 属性和 s_tel 属性创建了索引；teacher 表中的 t_name 属性创建了索引；ttr 表中的 classNo 属性创建了索引。索引方式均为 BTREE。

为测试创建索引对性能的优化，在 MySQL 中对 student 表的 s_name 属性进行索引前后，测试如下语句的执行时间：

SELECT s_name from student;

在未创建索引时，结果如图所示：

```

TEST992
TEST993
TEST994
TEST995
TEST996
TEST997
TEST998
TEST999
王一
王二
王三
王四
王五
丁一
丁二
+-----+
1016 rows in set (0.01 sec)

```

创建索引后，结果如图：

```

TEST998
TEST999
丁一
丁二
张三
张五
张六
张四
李一
李三
李二
李五
李四
王三
王二
王五
王四
+-----+
1016 rows in set (0.00 sec)

```

可以看出，创建索引对提升查询速度产生了一定效果，但由于数据总量仍不够大，因此效果不算明显。

实验题目	数据库设计与应用开发			实验日期	2022. 5. 1
班级	1903103	学号	1190200208	姓名	李旻翀

四、实验结论（总结实验发现及结论）

1. 通过本次实验，我强化掌握了数据建模的思想和方法，并体会到良好的数据库设计对后续数据库开发的重要作用。
2. 更加熟练地掌握了 SQL 语言的使用。包括数据库的建立、视图和索引的建立、表的建立、插入、删除、查询等方面，同时我也更加掌握到了数据库完整性约束的作用和重要性（尤其是外键约束）。
3. 初步掌握了 python 环境下的嵌入 SQL 语言的使用和动态构建 SQL 语句的方法，构建数据库应用界面的过程也让我熟悉了 PyQt5 的使用。
4. 了解了数据库对于大规模系统构建的重要性。以此实验构建的数据库，后续可以进行丰富的拓展，这让我体会到了软件构造的复杂性与乐趣。与此同时，系统中涉及数据库的部分也让我知道要实现更加复杂的功能，还需要进一步对数据库进行探索和学习。