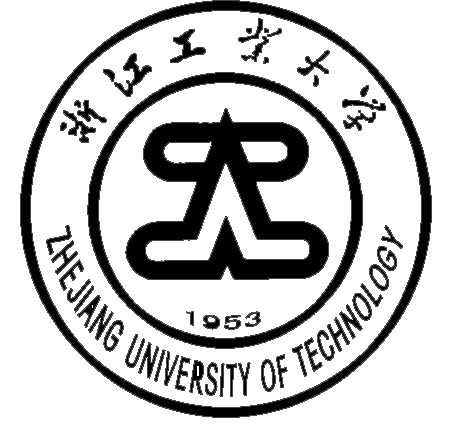


**《数据库设计》期末课程作业**

**2021/2022（1）**



题目 **学习强国数据库设计复现**

班级 软工1904

学号 201906062126

姓名 张成强

指导教师 李伟

所在学院 计算机学院

提交日期 2021年12月 2日

目录

[1、](#_Toc44251151)[需求分析 1](#_Toc44251151)

[1.1 数据需求描述 1](#_Toc44251152)

[1.2 系统功能需求 10](#_Toc44251153)

[1.3 其他性能需求 13](#_Toc44251154)

[2、概念结构设计 13](#_Toc44251155)

[2.1 局部E-R图 1](#_Toc44251156)4

[2.2 全局E-R图 15](#_Toc44251157)

[2.3 优化E-R图 16](#_Toc44251158)

[3、逻辑结构设计 17](#_Toc44251159)

[3.1 关系模式设计 17](#_Toc44251160)

[3.2 数据类型定义 18](#_Toc44251161)

[3.3 关系模式的优化 18](#_Toc44251162)

[4、物理结构设计 20](#_Toc44251163)

[4.1 聚簇设计 20](#_Toc44251164)

[4.2 索引设计 20](#_Toc44251165)

[4.3 分区设计 21](#_Toc44251166)

[5、数据库实施 20](#_Toc44251167)

[5.1 基本表建立 21](#_Toc44251168)

[5.2 视图的建立 27](#_Toc44251169)

[5.3 索引的建立 31](#_Toc44251170)

[5.4 触发器建立 32](#_Toc44251171)

[5.5 建存储过程 33](#_Toc44251172)

[6、应用系统开发与试运行 34](#_Toc44251173)

[6.1 开发平台和开发环境介绍。 34](#_Toc44251174)

[6.2 前台界面与后台数据库连接说明，代码实现。 34](#_Toc44251175)

[6.3 系统各功能设计和运行界面截图。 35](#_Toc44251176)

[7、实验总结 65](#_Toc44251177)

[7.1 遇到的问题和解决的办法 65](#_Toc44251178)

[7.2 系统设计的不足 66](#_Toc44251179)

[7.3 进一步改进思路和体会 67](#_Toc44251180)

**高校成绩管理数据库系统的设计与实现**

**小组组长：张成强** **小组成员：卢鹏宇，黄益妙**

**小组详细分工情况：**

**设计数据库：张成强负责数据字典，数据流图**

**卢鹏宇负责E-R图的编辑，和表的设计**

**黄益秒负责索引分区的设计，和具体实现**

**实现功能：三人各自完成前端、后端、数据库连接的功能。**

## 1、需求分析

随着中国高等教育的蓬勃发展，大学生人数逐步上升。同时，高校是教学和科研的重要基地，同时还是培养人才的重要场所。随着计算机技术的发展和互联网时代的到来，学习的教学管理受到了极大的挑战。随着学校的规模不断扩大，学生数量急剧增加，学生的成绩信息也成倍增长。这使得部门工作人员的工作量也急剧增加，出现的错误也会随之增加。因此学校的成绩管理面临新的需求，即在疏通各部门尤其是各系之间信息渠道的基础上，建立全校共享数据库。由授课教师来完成成绩录入，而学生可以自行查询成绩。这样除了减轻教学管理部门人员的工作负担，还可以减少信息错误的发生，同时为其他部门甚至全校教职工提供信息服务。建立现代化办公的工作环境，极大地减轻了管理人员的工作量，提高工作效率，并且实现了部门信息管理与学校自动化办公的有机结合。高校成绩管理系统存在的主要问题就是“信息孤岛”的问题。运行计算机技术与网络技术，可以极大便利学生与教师，管理效率大大提高。运行本系统，需要配置好mysql数据库，安装相应的java环境，并下载对应的数据库连接jar包。

### 1.1 数据需求描述

系统的数据主要包括学生、教师等的人员信息，课程、专业、学院等基本信

息，最重要的是学生成绩信息。下面以数据流图与数据字典的形式详细描述系统

的数据需求。

1. **初级数据流图**

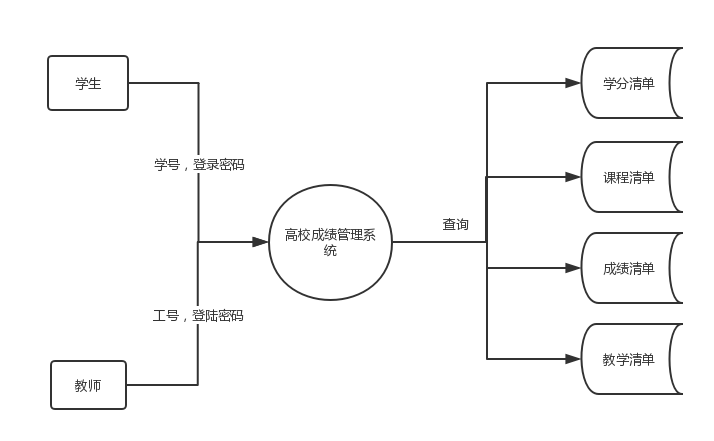
****

图1.1 学生成绩管理系统初级数据流图

1. **详细数据流图**

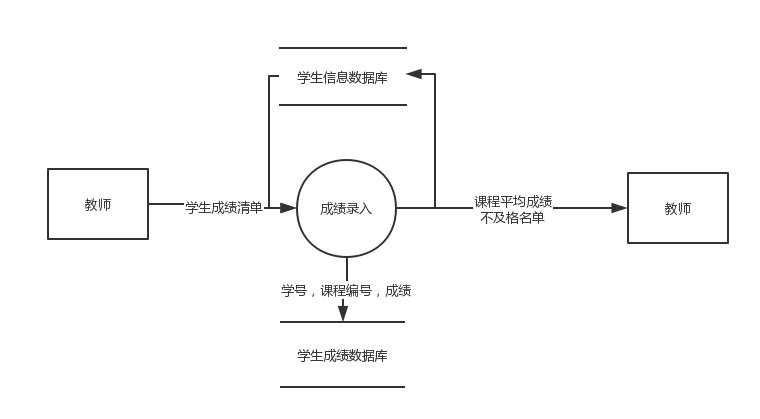
****

图1.2 教师录入成绩数据流图

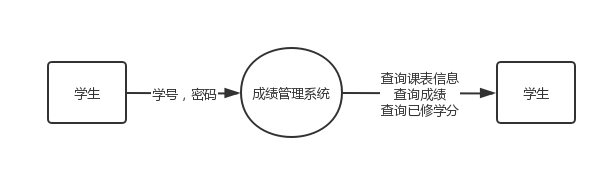


图1.3 学生信息查询数据流图

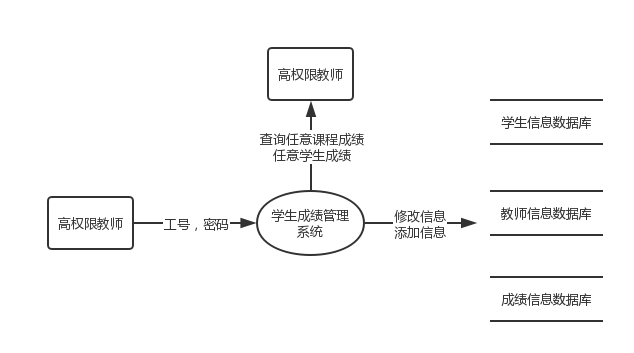


图1.4 高权限教师数据流图

1. 数据字典

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 学生 |
| 别 名 | 学生基本信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一个学生的基本信息 |
| 组 成 | 学号，专业（外键），姓名，性别，年龄，平均绩点，已修学分，联系电话，生源地，登录密码，班级编号（外键） |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 学号 |
| 别 名 | 学生编号 |
| 描 述 | 唯一标识每个学生 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 顺序编号 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 专业 |
| 别 名 | 学生专业 |
| 描 述 | 表示这个学生所属专业 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 按实际需求 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 姓名 |
| 别 名 | 学生名称 |
| 描 述 | 学生生活中的称呼 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 性别 |
| 别 名 | 学生性别 |
| 描 述 | 区分学生的一个指标 |
| 定 义 | CHAR（1） |
| 取值范围 | 男或女 |
| 取值含义 | 从{男，女}中选择 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 班级 |
| 别 名 | 所属班级 |
| 描 述 | 这个学生的所属班级 |
| 定 义 | CHAR（20） |
| 取值范围 | 1到20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 年龄 |
| 别 名 | 出生日期 |
| 描 述 | 一个人什么时候出生的 |
| 定 义 | INT |
| 取值范围 | 0-100 |
| 取值含义 | 按照身份证上取值 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 平均绩点 |
| 别 名 | 平均成绩 |
| 描 述 | 该学生的平均成绩 |
| 定 义 | DOUBLE |
| 取值范围 | 0-5 |
| 取值含义 | 按照成绩取值 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 已修学分总数 |
| 别 名 | 学生取得的学分 |
| 描 述 | 一个学生从入学到现在获得的学分 |
| 定 义 | INT |
| 取值范围 | 0-300 |
| 取值含义 | 按照课程学分取值 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 联系电话 |
| 别 名 | 电话号码 |
| 描 述 | 该学生电话号码 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 生源地 |
| 别 名 | 地区 |
| 描 述 | 一个学生之前就读的地区 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 学生表 |

|  |  |
| --- | --- |
| 数据项名 | 登录密码 |
| 别 名 | 密码 |
| 描 述 | 该学生登录系统的密码 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 学生表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 课程信息 |
| 别 名 | 课程基本信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一门课程的基本信息 |
| 组 成 | 课程编号,课程名，课程学分，课程学时，开设学期，考核方式，任课教师（外键），开设学院（外键） |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 课程编号 |
| 别 名 | 课程的编码 |
| 描 述 | 一个课程的唯一标识符 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 顺序取值 |
| 位 置 | 课程信息表 |

|  |  |
| --- | --- |
| 数据项名 | 课程名 |
| 别 名 | 课程称谓 |
| 描 述 | 一个课程的名称 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 课程信息表 |

|  |  |
| --- | --- |
| 数据项名 | 课程学分 |
| 别 名 | 学分 |
| 描 述 | 修完该课程获得的学分 |
| 定 义 | INT |
| 取值范围 | 0-10 |
| 取值含义 | 按照实际课程需求 |
| 位 置 | 课程信息表 |

|  |  |
| --- | --- |
| 数据项名 | 课程学时 |
| 别 名 | 学时 |
| 描 述 | 该课程这一学期所需要的上课时间 |
| 定 义 | INT |
| 取值范围 | 0-100 |
| 取值含义 | 按照实际课程需求 |
| 位 置 | 课程信息表 |

|  |  |
| --- | --- |
| 数据项名 | 开设学期 |
| 别 名 | 教学学期 |
| 描 述 | 课程开设的学期 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 自定义 |
| 位 置 | 课程信息表 |

|  |  |
| --- | --- |
| 数据项名 | 考核方式 |
| 别 名 | 考试或考查 |
| 描 述 | 该课程的考察方式 |
| 定 义 | CHAR(2) |
| 取值范围 | {考试，考查} |
| 取值含义 | 按照实际课程需求 |
| 位 置 | 课程信息表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 教师信息 |
| 别 名 | 教师基本信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一门课教师的基本信息 |
| 组 成 | 工号，姓名，性别，职称，联系电话，权限级别，登录密码，年龄，学院编号（外键） |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 工号 |
| 别 名 | 教师编号 |
| 描 述 | 参与教学教师的编号 |
| 定 义 | CHAR(20) |
| 取值范围 | 1-20个字符 |
| 取值含义 | 顺序取值 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 姓名 |
| 别 名 | 老师姓名 |
| 描 述 | 老师的称呼 |
| 定 义 | CHAR(20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 身份证上的称谓 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 性别 |
| 别 名 | 老师性别 |
| 描 述 | 老师的性别 |
| 定 义 | CHAR(1） |
| 取值范围 | {男，女} |
| 取值含义 | 选择其中一个 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 职称 |
| 别 名 | 老师职称 |
| 描 述 | 老师的职称 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 教师的职称 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 联系电话 |
| 别 名 | 电话号码 |
| 描 述 | 老师的联系方式 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 权限级别 |
| 别 名 | 系统权限 |
| 描 述 | 该教师登录系统获得的权限 |
| 定 义 | INT |
| 取值范围 | 0--10 |
| 取值含义 | 顺序取值 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 登录密码 |
| 别 名 | 密码 |
| 描 述 | 教师登录系统的密码 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个 |
| 取值含义 | 教师的登录密码 |
| 位 置 | 教师信息表 |

|  |  |
| --- | --- |
| 数据项名 | 年龄 |
| 别 名 | 出生日期 |
| 描 述 | 教师的年龄 |
| 定 义 | INT |
| 取值范围 | 0-100 |
| 取值含义 | 教师年龄大小 |
| 位 置 | 教师信息表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 班级信息 |
| 别 名 | 班级基本信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一个班级的基本信息 |
| 组 成 | 班级编号，班级名 |

|  |  |
| --- | --- |
| 数据项名 | 班级编号 |
| 别 名 | 班级代号 |
| 描 述 | 该班级的唯一标识符 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 班级信息表 |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 班级名 |
| 别 名 | 班级名称 |
| 描 述 | 该班级的称呼 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 班级信息表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 专业信息 |
| 别 名 | 专业基本信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一个专业的基本信息 |
| 组 成 | 专业编号，专业名，所属学院（外键） |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 专业编号 |
| 别 名 | 专业代号 |
| 描 述 | 该专业的唯一表示符 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 专业信息表 |

|  |  |
| --- | --- |
| 数据项名 | 专业名 |
| 别 名 | 专业称呼 |
| 描 述 | 该专业的称谓 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 专业信息表 |

|  |  |
| --- | --- |
| 数据项名 | 所属学院 |
| 别 名 | 管辖学院 |
| 描 述 | 该专业在哪个学院下开设 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 专业信息表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 学院信息 |
| 别 名 | 专业基本信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一个学院的基本信息 |
| 组 成 | 学院编号，学院名 |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 学院编号 |
| 别 名 | 学院的代号 |
| 描 述 | 该学院的唯一标识符 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 学院信息表 |

|  |  |
| --- | --- |
| 数据项名 | 学院名 |
| 别 名 | 学院名称 |
| 描 述 | 该学院的称呼 |
| 定 义 | CHAR（20） |
| 取值范围 | 1-20个字符 |
| 取值含义 | 实际需求取值 |
| 位 置 | 学院信息表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 学生成绩 |
| 别 名 | 学生成绩信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一个学生成绩的基本信息 |
| 组 成 | 学号（外键），课程编号（外键），成绩 |

**b.数据项条目**

|  |  |
| --- | --- |
| 数据项名 | 成绩 |
| 别 名 | 绩点 |
| 描 述 | 学生在这门课的成绩 |
| 定 义 | INT |
| 取值范围 | 0-100 |
| 取值含义 | 实际需求取值 |
| 位 置 | 学生成绩表 |

**a.数据结构条目**

|  |  |
| --- | --- |
| 数据结构名 | 开设信息 |
| 别 名 | 课程开设信息 |
| 描 述 | 这是学生成绩管理系统的主要数据结构，定义了一个课程开设的基本信息 |
| 组 成 | 课程编号（外键），班级编号（外键） |

### c.数据存储条目

|  |  |
| --- | --- |
| 数据存储名 | 学生成绩单 |
| 别名 | 学生成绩清单 |
| 描述 | 学生考试成绩的具体信息 |
| 数据流来源 | 教师 |
| 数据流去向 | 学生成绩数据库 |
| 组成 | 学生名，教师名，课程名，分数 |

**d.数据处理条目**

|  |  |
| --- | --- |
| 处理过程名 | 学生成绩录入 |
| 说明 | 教师将成绩单录入到学生成绩数据库 |
| 流入数据流 | 学生成绩单 |
| 流出数据流 | 课程平均成绩，不及格学生名单 |
| 处理逻辑：将学生成绩添加到学校成绩数据库中，更新学生的绩点。  如果成绩几个，则将该学生的已修学分增加相应的学分，平均绩点进行更新。  否则，将该条目添加不及格名单中，并将平时绩点更新。 | |

根据上面的数据字典，我们创建了如下的UML图。

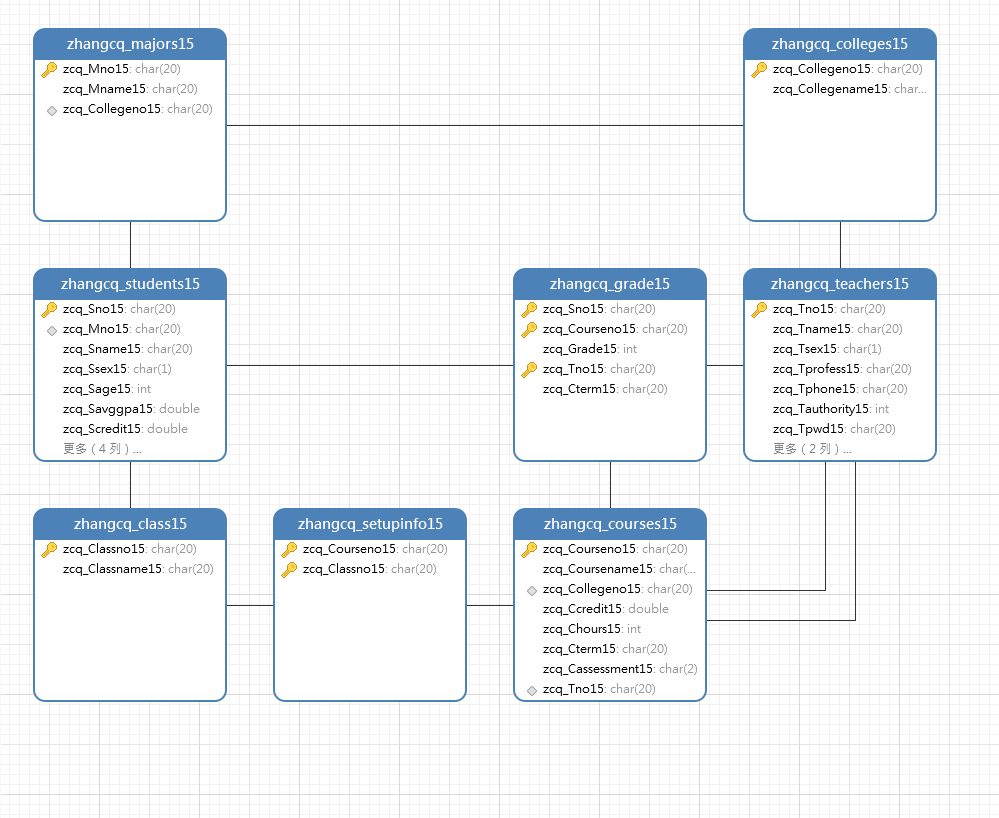


图1.5成绩管理系统UML图

### 1.2 系统功能需求

成绩管理主要是教务处对教师、对课程、对学生成绩管理，以便于教师，学生能够随时对学生成绩进行各种查询。

他服务的对象不同，提供的功能也不同。

1. 高权限教师（系统管理员）

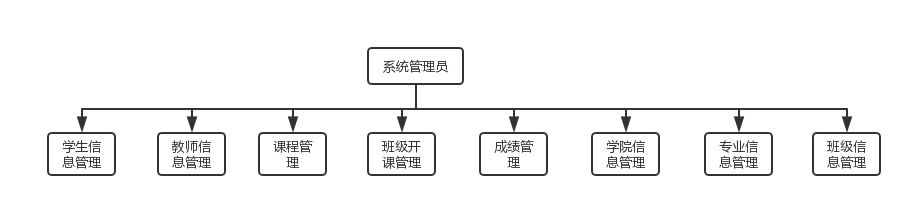


图1.6 系统管理员功能模块图

1. 学生信息管理：按学好可以查找到学生的详细信息，并可以添加、删除、修改学生信息
2. 教师信息管理：教师以学院为单位划分，可按学院，按教师名，教师号等方式查看教师信息，并由添加，删除、修改教师信息的功能。
3. 课程管理：可以按课程编号查看课程的详细信息，可以添加，删除修改开课记录。
4. 班级开课管理，以班级为单位，根据班级进行开课管理
5. 成绩管理：按学号可以查找学生的全部成绩信息。可以查看该学生的成绩但没有修改成绩功能。成绩录入应由相应老师完成。
6. 学院信息管理，查看，添加，修改，删除学院信息。
7. 专业信息管理，查看，添加，修改，删除专业信息。
8. 班级信息管理，查看，添加，修改，删除班级信息。
9. 普通教师

教师登录后可以修改自己的教师信息，可以修改登录密码，以班级为单位查询学生的课程信息，以班级为单位录入成绩。可以查看任课情况。

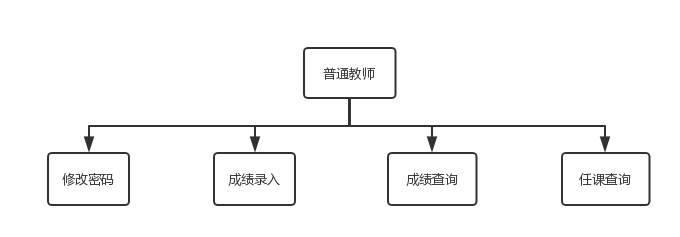


图1.7教师功能模块图

1. 成绩录入模块：根据教师编号，上课学期，查询所有开课记录，以及开课记录对应班级的所有学生，对每个学生的该科成绩进行录入修改。

1. 成绩查询模块：教师只能查询自己教授的课程的所有同学的成绩。可获得不及格学生名单。课程的平均成绩。
2. 教师任课情况查询：自动获得教师的所有任课情况， 包括课程名称，班级名称，平均成绩。
3. 学生

学生登陆后可以修改自己的信息，可以修改登陆密码；查看全部已考科目的成绩，查看本学期成绩以及不及格科目成绩，查看本学期或学年的班级以及年级名次，以及班级课表、已修学分、总平均绩点、专业教学计划查询等查询。

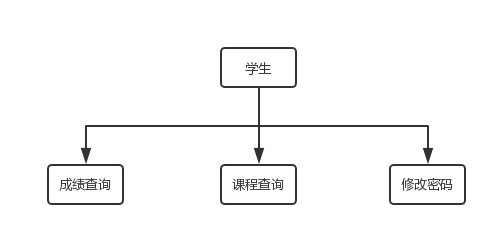


图1.8学生功能模块图

1.成绩查询：

a) 学生成绩查询：按学期（学年）查询，获得该学期（学年）成绩单，平均绩点，不及格科目，班级排名，年级排名。

b) 课程平均成绩查询：按班级、学期（学年）查询，获得该学期（学年）

各个课程的平均成绩。

2. 课程查询：

a) 班级课程开课查询：根据学期查询，上课科目，任课教师

b) 学生教学计划查询：专业教学计划，已修学分，平均绩点。（毕业最低学分，毕业最低绩点）已修科目各科成绩。

### 1.3 其他性能需求

该系统应支持多用户同时对数据库进行访问，即支持并发。

实现方法主要是通过java数据源进行数据库连接，并且Servlet编程自带多线程可以支持并发，以此来提升响应速度。

当然响应时间越短越好，但考虑到服务器成本问题，实际响应时间不需要很段，因为也没有必要，在数据访问高峰期，如期末考后，则应该增强服务器的性能。

因此，数据库服务器性能最好能动态变化。因此需要在统计高峰期后做好高并发准备。

## 2、概念结构设计

1. 下图2-1-1为教师ER图，其属性包括姓名、教师编号、性别、职称、年龄、联系电话、登录密码、权限级别。

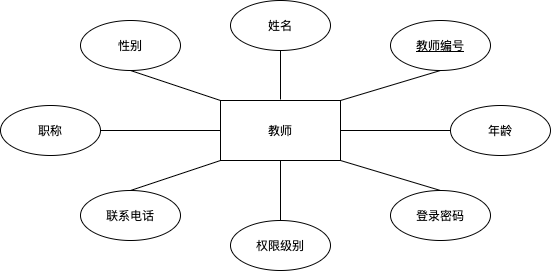


图 2-1-1 教师ER图

1. 下图2-1-2为学生ER图，其属性包括姓名、学号、性别、专业编号、平均绩点、已修学分总数、联系电话、登录密码、生源所在地、班级编号、年龄。

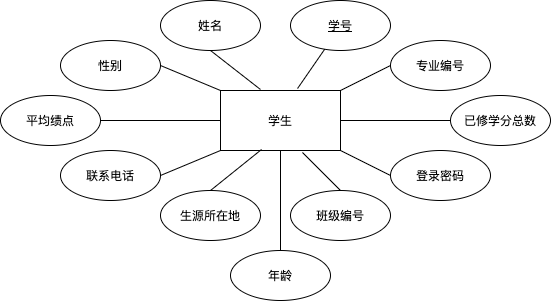


图 2-1-2 学生ER图

1. 下图2-1-3为课程ER图，其属性包括课程编号、课程名称、开设学院、课程学分、课程学时、开设学期、考核方式、任课教师。

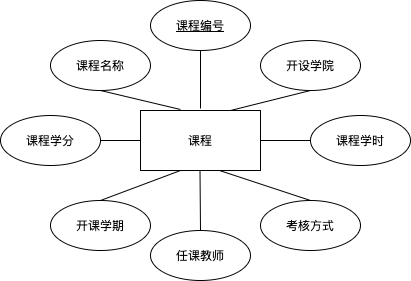


图 2-1-3 课程ER图

1. 下图2-1-4为学院ER图，其属性包括学院编号、学院名

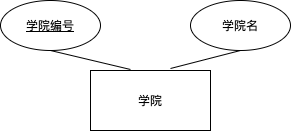


图 2-1-4 学院ER图

1. 下图2-1-5为专业ER图，其属性包括专业编号、专业名、所属学院编号。

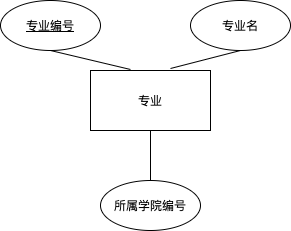


图 2-1-5 专业ER图

1. 下图2-1-6为班级ER图，其属性包括班级编号、班级名、所属学院编号、所属专业编号。

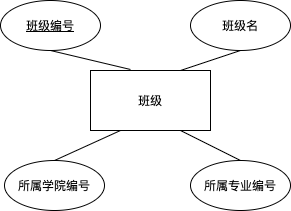


图 2-1-6 班级ER图

### 2.2 全局E-R图

下图2-2-1为局部ER图合并成的全局ER图，其中课程和班级为n：m的联系，联系名为“开设”；课程和学生为n：m的联系，联系名为“选修”，“选修”具有成绩属性；班级和学生为1:n的联系，联系名为“属于”；专业和学生为1:n的联系，联系名为“属于”；学院和专业为1:n的联系，联系名为“属于”；学院和教师为1:n的联系，联系名为“属于”；课程和教师为n：m的联系，联系名为“任教”。

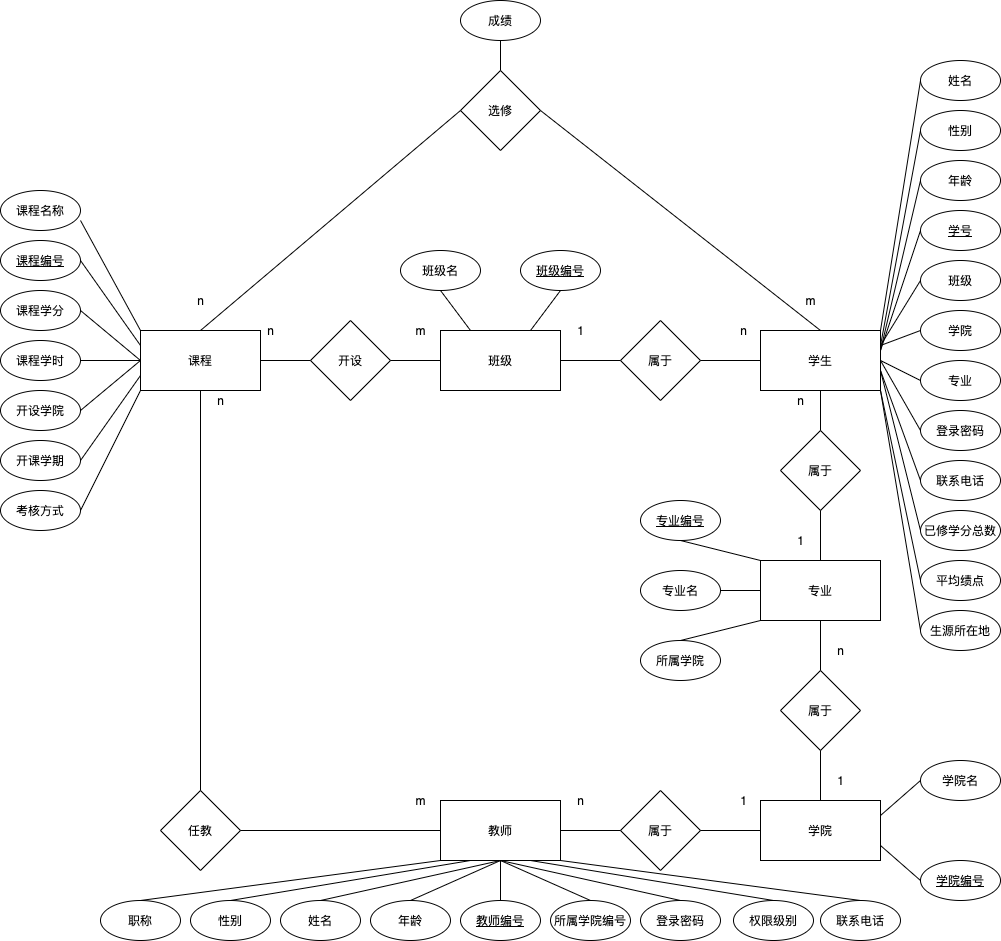


图 2-2-1 全局ER图

### 2.3 优化E-R图

下图2-3-1为优化后的ER图，删除了学生、专业、教师实体中冗余的属性，将课程信息的开设学院设为外键。

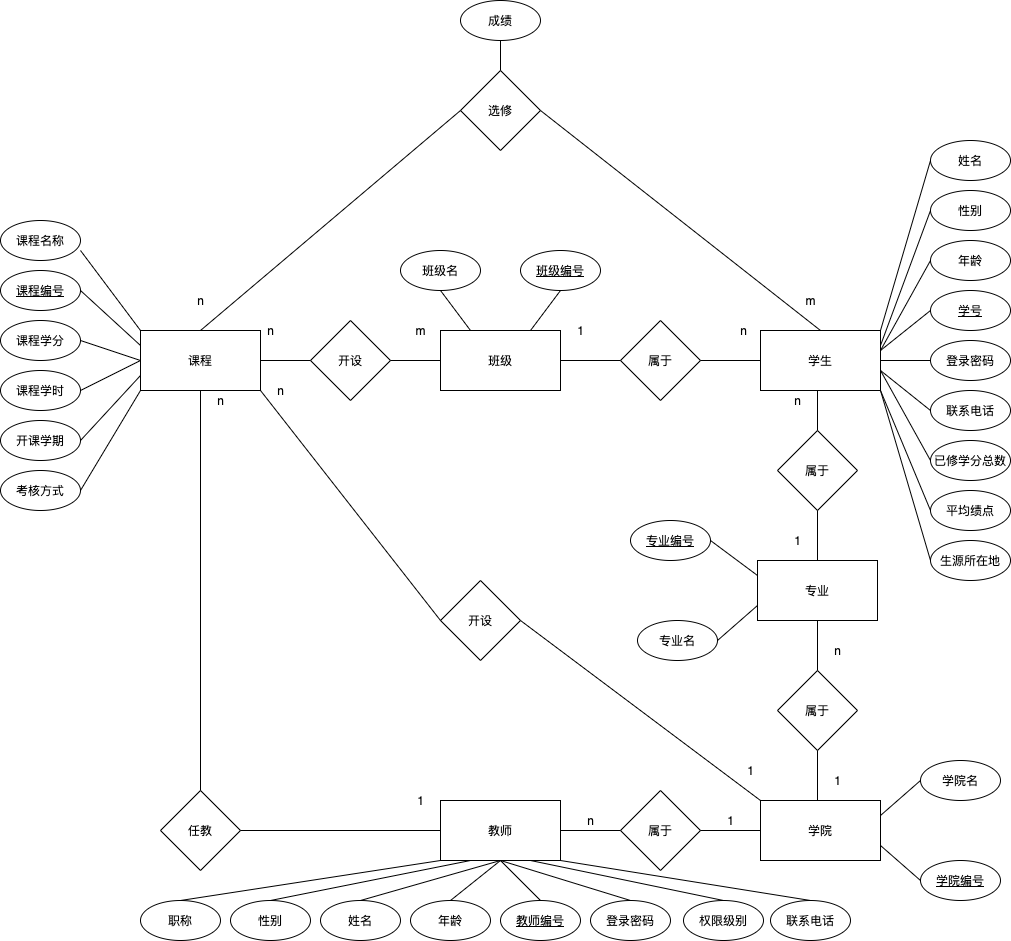


图 2-3-1 优化后的全局ER图

在做课程设计的过程中，考虑到一门课在不同的学期可能是不同的老师教，所以把课程字段中的开设学期和任课老师去掉了。

## 3、逻辑结构设计

### 3.1 关系模式设计

加红的是主键，加波浪线的是外键

1. 课程（课程编号，开设学院，课程名称，课程学分，课程学时，考核方式）
2. 班级（班级编号，班级名）
3. 学生（学号，班级编号，专业编号，姓名，性别，年龄，登录密码，联系电话，已修学分总数，平均绩点，生源所在地）
4. 专业（专业编号，学院编号，专业名）
5. 学院（学院编号，学院名）
6. 教师（教师编号，学院编号，姓名，性别，年龄，职称，登录密码，权限级别，联系电话）
7. 选修信息（学号，课程编号，任课教师，课程名称，成绩，学期）
8. 开设信息（课程编号，班级编号）

### 3.2 关系模式的优化

检查各个关系模式是否满足1NF、2NF，消除关系模式中的冗余与传递依赖。使整个关系模式都满足BCNF，从而优化关系模式。

加红的是主键，加波浪线的是外键

1. 课程（课程编号，开设学院，课程名称，课程学分，课程学时，考核方式）
2. 班级（班级编号，班级名）
3. 学生（学号，班级编号，专业编号，姓名，性别，年龄，登录密码，联系电话，已修学分总数，平均绩点，生源所在地）
4. 专业（专业编号，学院编号，专业名）
5. 学院（学院编号，学院名）
6. 教师（教师编号，学院编号，姓名，性别，年龄，职称，登录密码，权限级别，联系电话）
7. 选修信息（学号，课程编号，教师编号，成绩，学期）
8. 开设信息（课程编号，班级编号）

### 3.3 数据类型定义

1. 课程表

表 3-2-1 课程表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 课程编号 | CHAR | 20 | 主键，唯一，非空 |  |
| 开设学院 | CHAR | 20 | 外键 |
| 课程名称 | CHAR | 20 |  |
| 课程学分 | DOUBLE | 0-10 |  |
| 课程学时 | DOUBLE | 0-100 |  |
| 考核方式 | CHAR | 2 |  |

1. 班级表

表 3-2-2 班级表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 班级编号 | CHAR | 20 | 主键，唯一，非空 |  |
| 班级名 | CHAR | 20 |  |

1. 学生表

表 3-2-3 学生表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 学号 | CHAR | 20 | 主键，唯一，非空 |  |
| 班级编号 | CHAR | 20 | 外键 |
| 专业编号 | CHAR | 20 | 外键 |
| 姓名 | CHAR | 20 |  |
| 性别 | CHAR | 1 |  |
| 年龄 | INT | 0-100 |  |
| 登录密码 | CHAR | 20 |  |  |
| 联系电话 | CHAR | 20 |  |  |
| 已修学分总数 | DOUBLE | 0-300 |  |  |
| 平均绩点 | DOUBLE | 0-5 |  |  |
| 生源所在地 | CHAR | 20 |  |  |

1. 专业表

表 3-2-4 专业表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 专业编号 | CHAR | 20 | 主键，唯一，非空 |  |
| 学院编号 | CHAR | 20 | 外键 |
| 专业名 | CHAR | 20 |  |

1. 学院表

表 3-2-5 课程表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 学院编号 | CHAR | 20 | 主键，唯一，非空 |  |
| 学院名 | CHAR | 20 |  |

1. 教师表

表 3-2-6 课程表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 教师编号 | CHAR | 20 | 主键，唯一，非空 |  |
| 学院编号 | CHAR | 20 | 外键 |
| 姓名 | CHAR | 20 |  |
| 性别 | CHAR | 1 |  |
| 年龄 | INT | 0-100 |  |
| 职称 | CHAR | 20 |  |
| 登录密码 | CHAR | 20 |  |  |
| 权限级别 | CHAR | 20 |  |  |
| 联系电话 | CHAR | 20 |  |  |

1. 选修信息表

表 3-2-7 课程表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 学号 | CHAR | 20 | 外键 | 学号、课程编号和任课教师一起构成主键 |
| 课程编号 | CHAR | 20 | 外键 |
| 教师编号 | CHAR | 20 | 外键 |
| 成绩 | DOUBLE | 255 |  |
| 学期 | CHAR | 20 |  |

1. 开设信息表

表 3-2-8 课程表数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项 | 数据类型 | 长度 | 完整性约束 | 备注 |
| 课程编号 | CHAR | 20 | 外键 | 课程编号和班级编号一起构成主键 |
| 班级编号 | CHAR | 20 | 外键 |

## 4、物理结构设计

### 4.1 聚簇设计

班级：需要在“班级编号”列上建立聚簇，该列为主键，经常用于连接查询并且装入属性后属性值很少修改 。

课程：需要在“课程编号”列上建立聚簇，该列为主键，经常用于连接查询并且装入属性后属性值很少修改 。开设信息：需要在“课程编号”和“班级编号”列上建立聚簇，该列为主键，经常用于连接查询并且装入属性后属性值很少修改 。教师：需要在“教师编号”列上建立聚簇，该列为主键，经常用于连接查询并且装入属性后属性值很少修改 。学生：需要在“学号”列上建立聚簇，该列为主键，经常用于连接查询并且装入属性后属性值很少修改 。选修信息：需要在“课程编号”和“学号”和“任课教师”列上建立聚簇，该列为主键，经常用于连接查询并且装入属性后属性值很少修改 。

### 4.2 索引设计

班级：对属性“班级编号”建立升序唯一索引课程：对属性“课程编号”建立升序唯一索引开设情况：建立按“班级编号”升序排列的普通索引教师：对“教师编号”建立升序唯一索引学生：对“学号”建立升序唯一索引选修信息：建立按“学号”升序和“成绩”升序排列的普通索引

### 4.3 分区设计

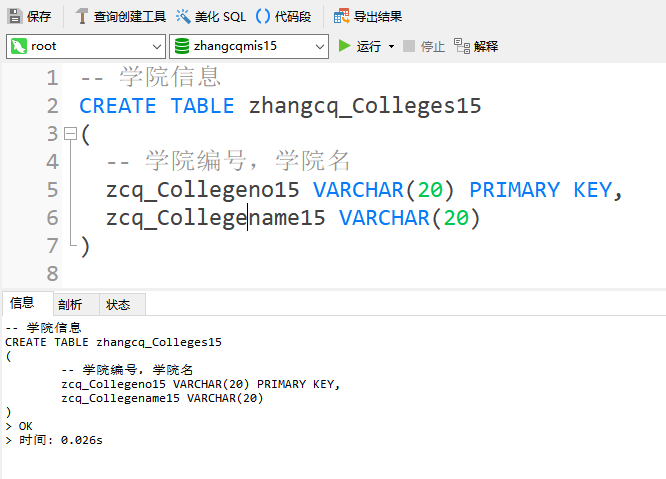
### 根据磁盘分区设计的一般原则：1. 减少访问冲突，提高I/O并发性。多个事物并发访问同一磁盘时，会产生磁盘访问冲突而导致效率低下，如果事务访问数据均能分布于不同磁盘上，则I/O可并发执行，从而提高数据库访问速度。2. 分散热点数据，均衡I/O负担。在数据库中数据访问的频率是不均匀的，那些经常被访问的数据成为热点数据，此类数据宜分散存在于不同的磁盘上，以均衡各个磁盘的负荷，充分发挥多磁盘的并行操作的优势。3. 保证关键数据快速访问，缓解系统瓶颈。在数据库中有些数据如数据字典等的访问频率很高，为保证对它的访问不直接影响整个系统的效率，可以将其存放在某一固定磁盘上，以保证其快速访问。

由于我们设计的系统数据量较小，因此不需要分区。

## 5、数据库实施

本实验的全部操作在mysql中的查询语句中实现。

### 5.1 基本表建立



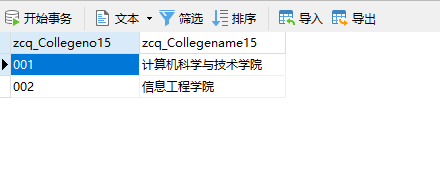


图5.1.1 创建基本表”zhangcq\_colleges15”

描述：zcq\_Collegeno15是主键。



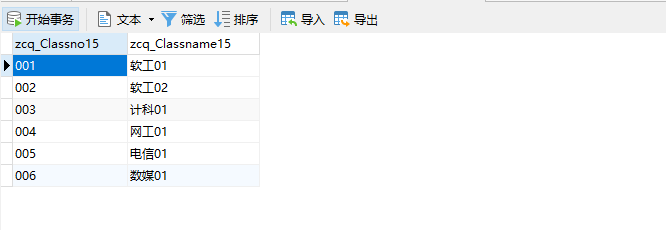


图5.1.2 创建基本表”zhangcq\_Class15”

描述：zcq\_Classno15是主键。



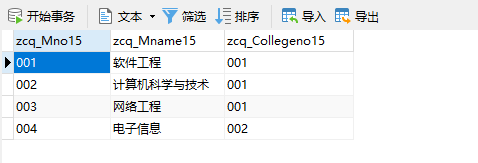
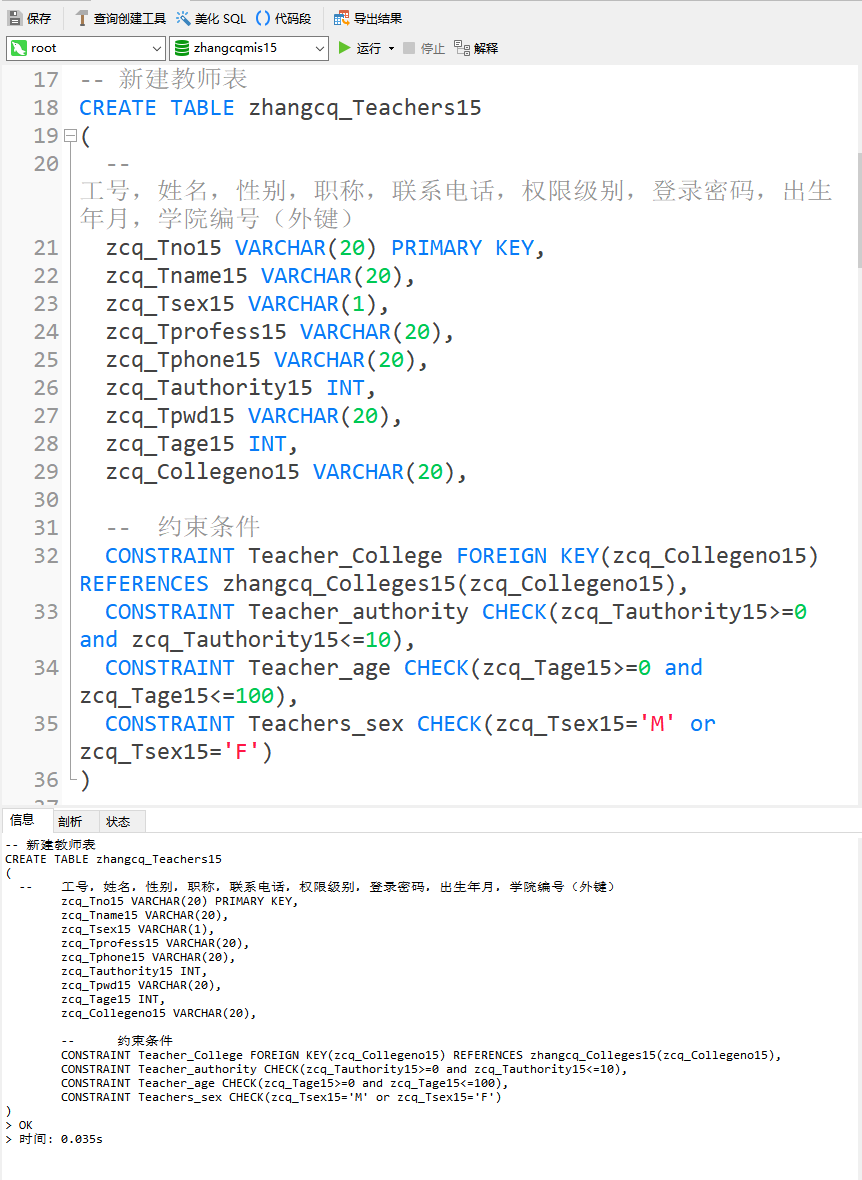


图5.1.3 创建基本表”zhangcq\_Major15”

描述：zcq\_Mno15是主键，zcq\_Collegeno是引用zhangcq\_College15表的外键。



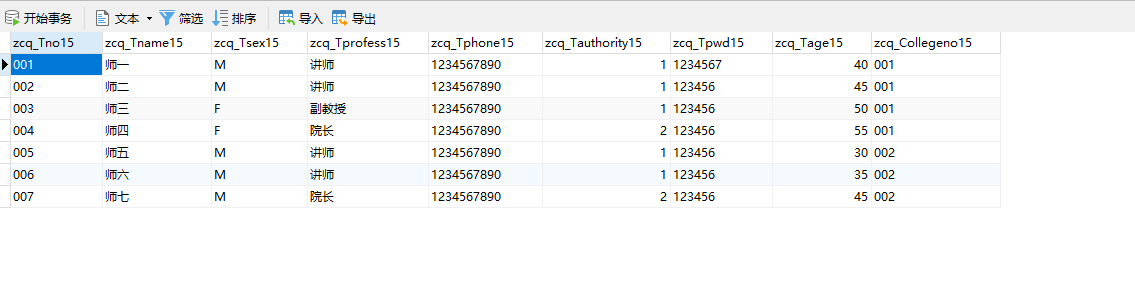
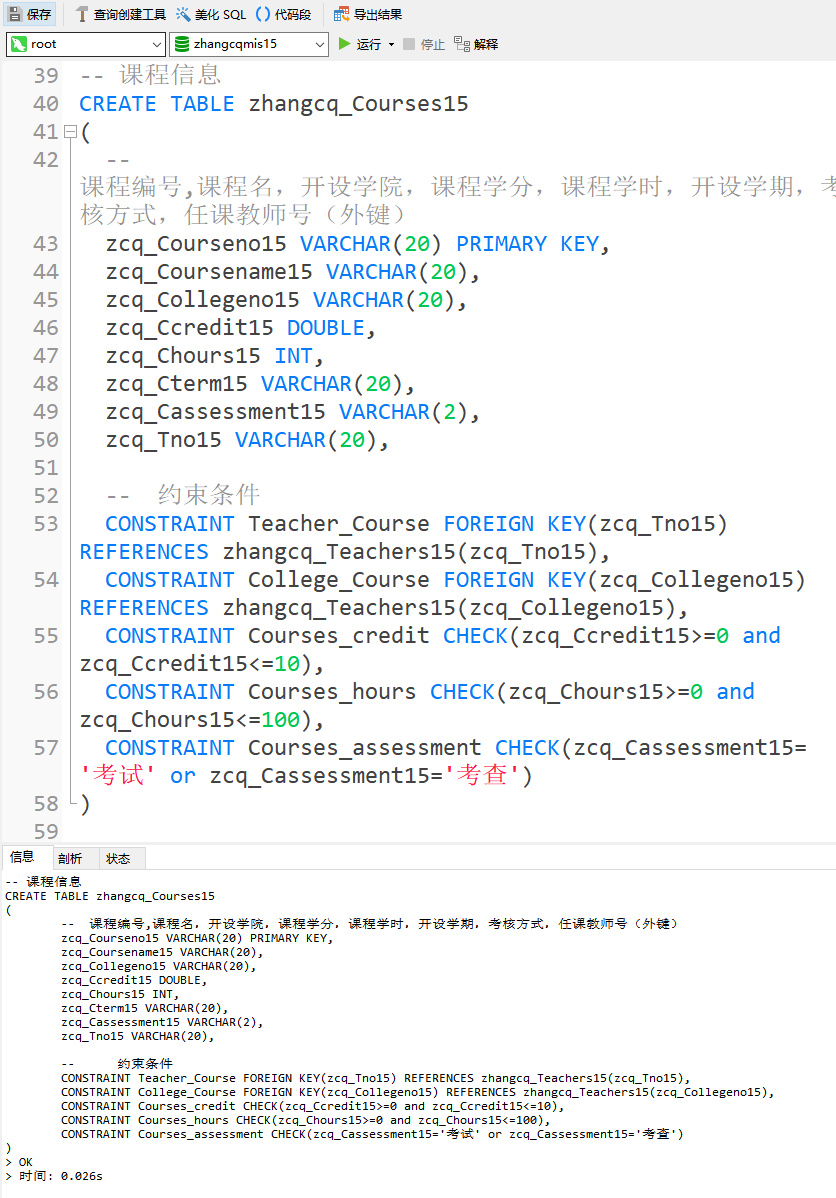


图5.1.4 创建基本表”zhangcq\_Teachers15”

描述：zcq\_Tno是主键，zcq\_Collegeno15是引用zhangcq\_College的外键。

自定义完整性约束：zcq\_Tage15的范围是0到100。Teacher\_authority的范围是0-10，代表管理权限 的10个层级。Teachers\_sex只能是M或F代表男女。



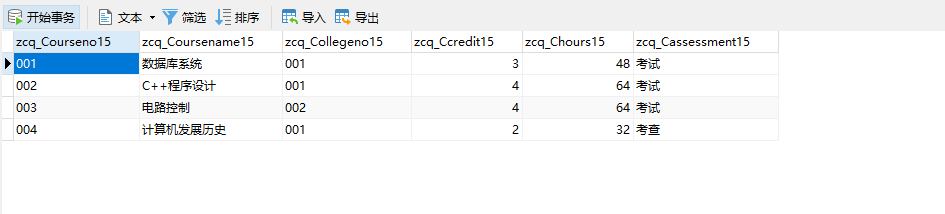
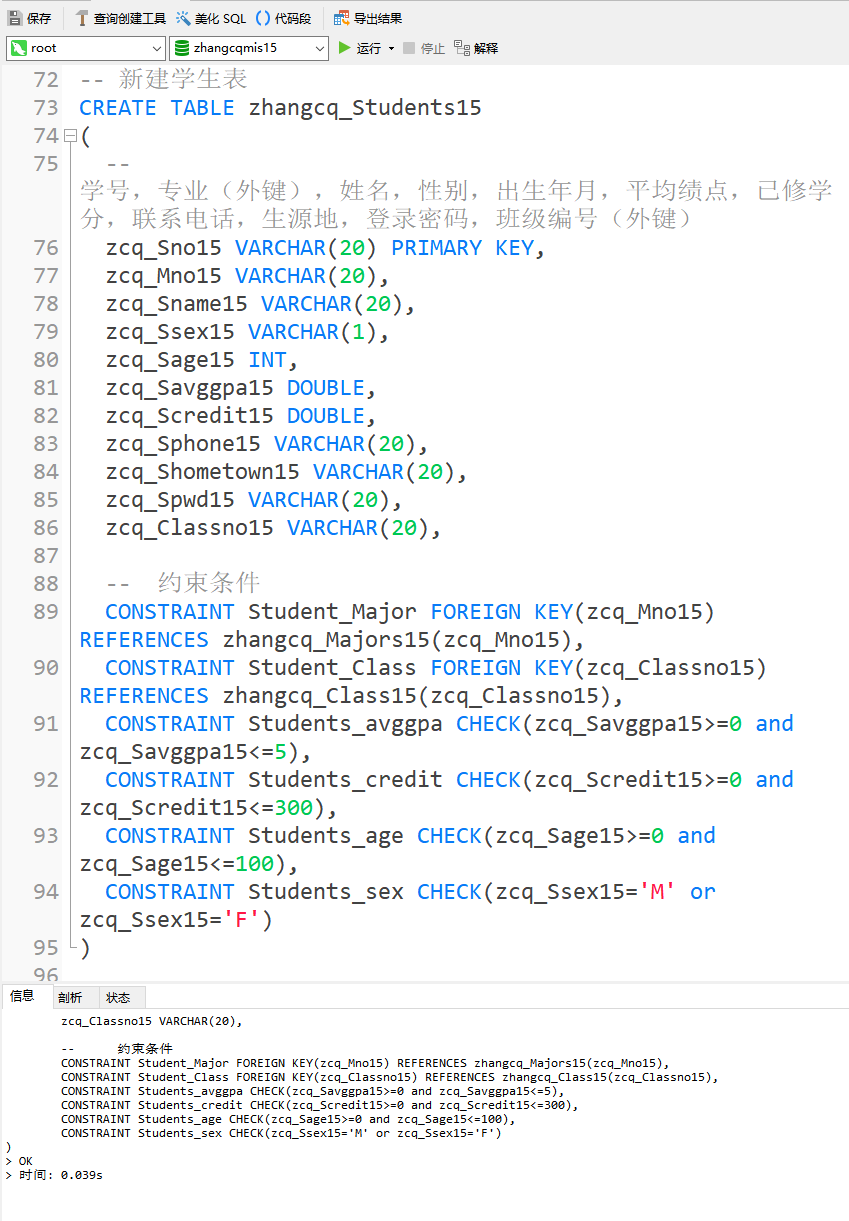


图5.1.5创建基本表”zhangcq\_Courses15”

描述：zcq\_Courseno是主键，zcq\_Tno15是引用zhangcq\_Teacher15的外键。zcq\_Collegeno15是引用 zhangcq\_colleges15的外键。

自定义完整性约束：Courses\_credit的范围是0到10，代表学分从0个到10个。Courses\_hours的范围是0-100，代表课程学时从0个到100个。Courses\_assessment只能是“考察”或“考试”。



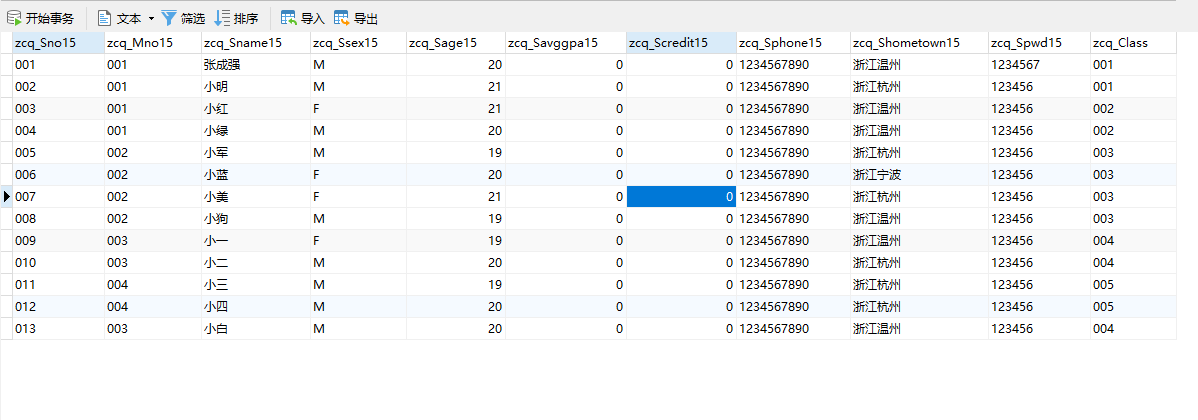


图5.1.6创建基本表”zhangcq\_Students15”

描述：zcq\_Sno是主键，zcq\_Mno15是引用zhangcq\_Majors15的外键。zcq\_Classno15是引用 zhangcq\_Class15的外键。

自定义完整性约束：zcq\_Savggpa15的范围是0到5，代表平均绩点从0个到5。zcq\_Scredit15的范围是0-300，代表已修学分从0个到300个。zcq\_Sage15从0-100，代表年龄从0-100。zcq\_Ssex15只能是M或F，代表男女。

### 

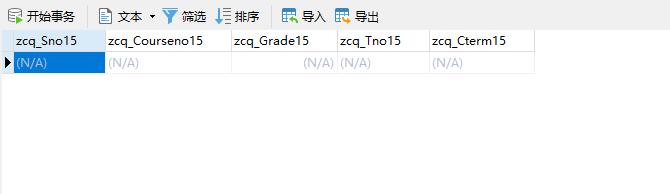


图5.1.7创建基本表”zhangcq\_Grade15”

描述：zcq\_Sno15,zcq\_Courseno15,zcq\_Tno15三个共同构成主键。

自定义完整性约束：zcq\_Grade15的范围是0-100，代表成绩从0-100。

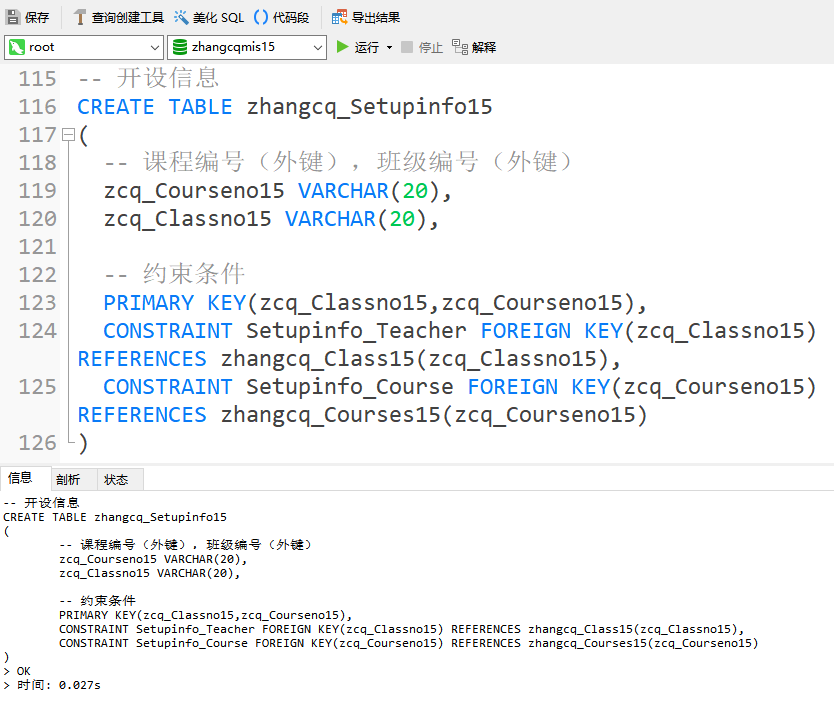




图5.1.8创建基本表”zhangcq\_Setupinfo15”

描述：zcq\_Classno15,zcq\_Courseno15两个共同构成主键。

### 5.2 视图的建立

以下视图都是为了提升用户体验而进行设计的，用户在进行查询时。如果只按照基本表中的信息给出的话，那些班级编号，课程编号，教师编号等用户会极难理解，所以利用视图将多个表中的字段提取出来，让后呈现给用户，提升用户体验。

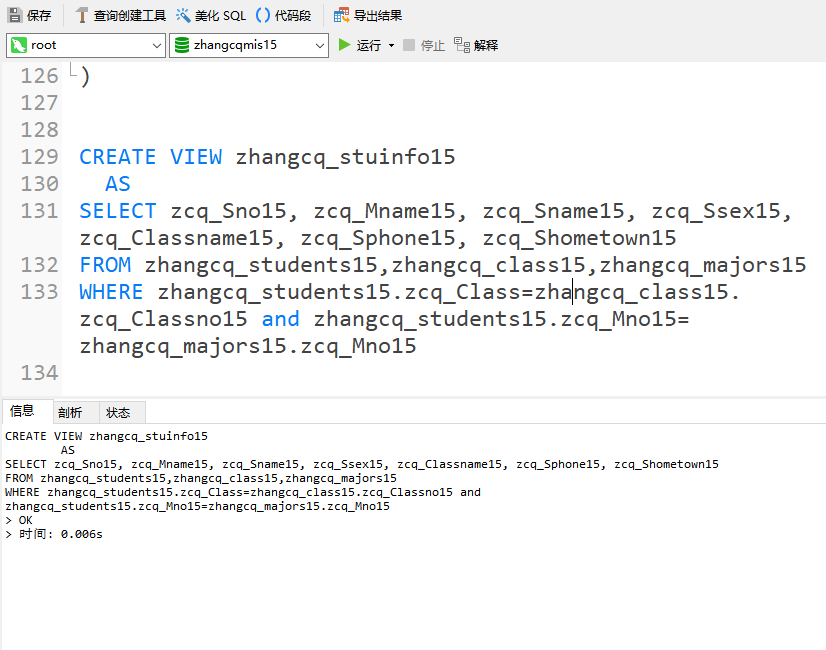


图5.2.1创建视图”学生信息”

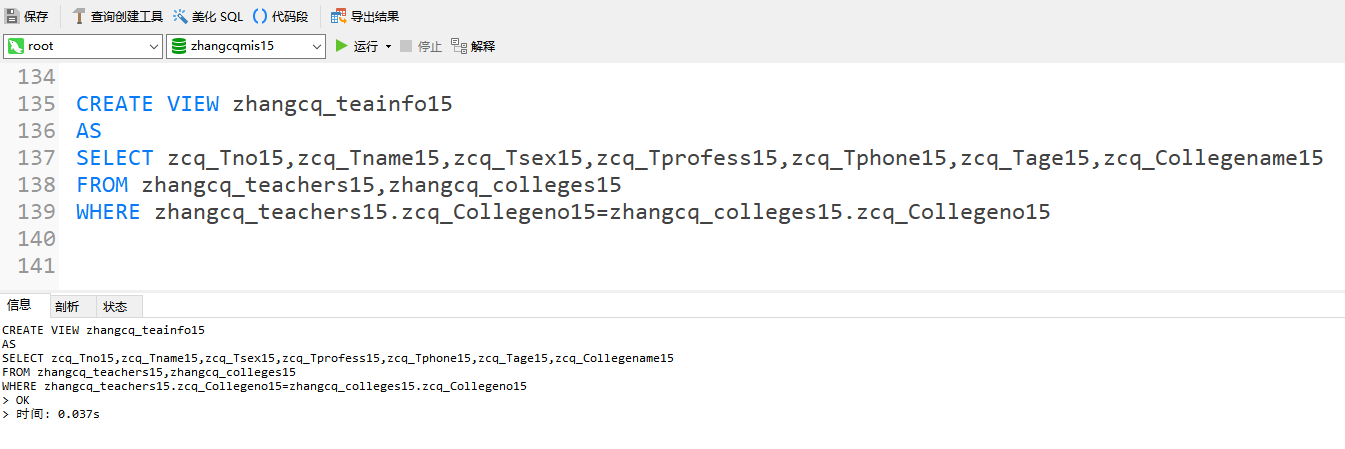


图5.2.2创建视图”教师信息”



图5.2.3创建视图”课程信息”



图5.2.4创建视图”课程开设信息”

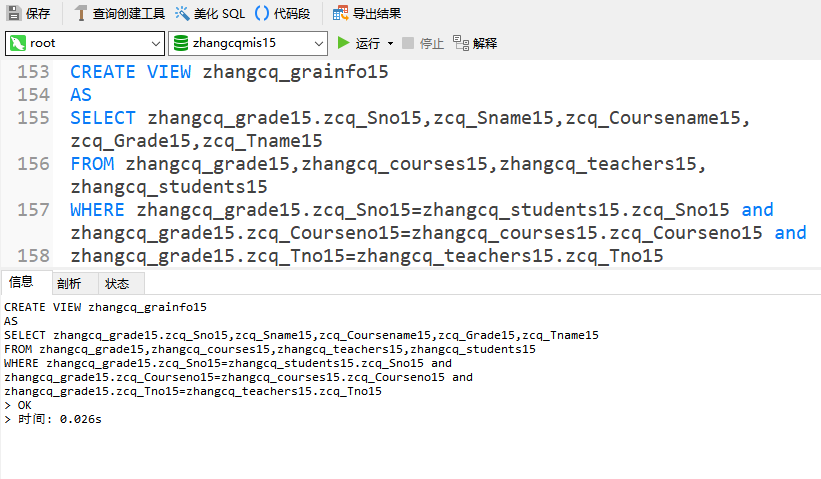


图5.2.5创建视图”成绩清单信息”

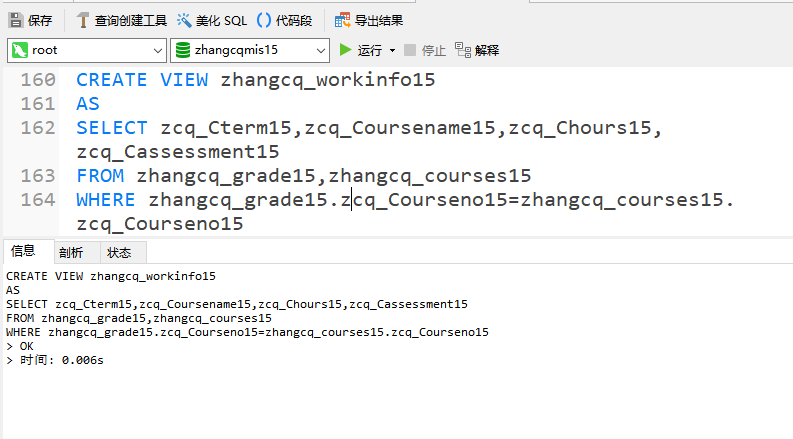


图5.2.6创建视图”任教课程信息”

### 5.3 索引的建立

以下索引是为了方便系统更好的查找数据，提升搜索效率和速度。



图5.3.1创建索引“对班级编号唯一升序索引”

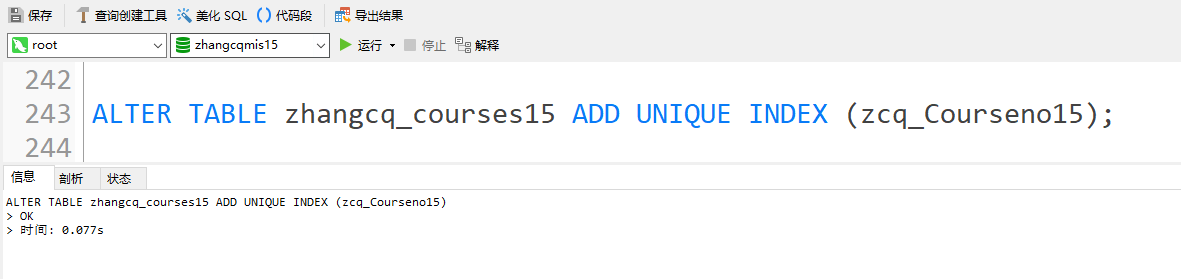


图5.3.2创建索引“对课程编号唯一升序索引”

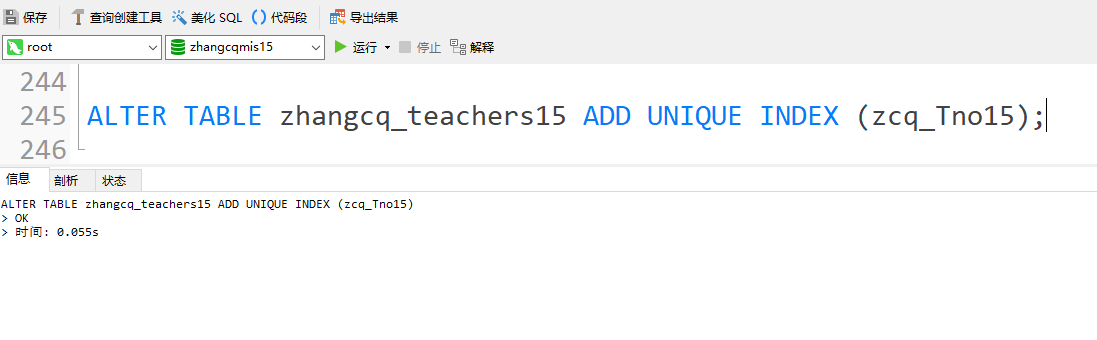


图5.3.3创建索引“对教师编号唯一升序索引”

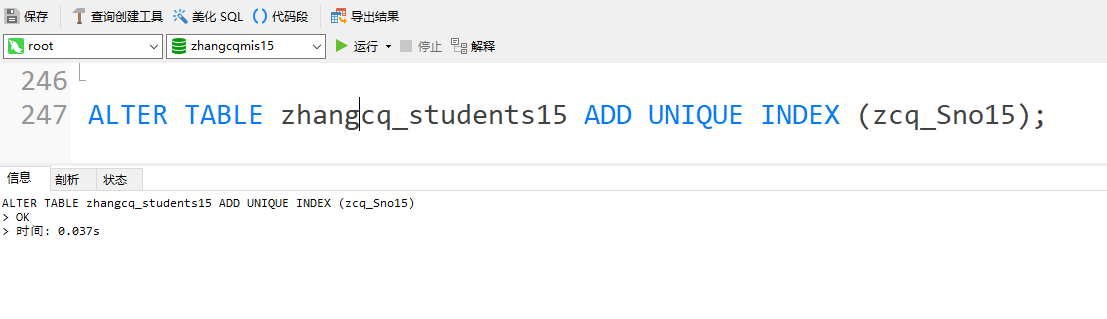


图5.3.4创建索引“对学生编号唯一升序索引”

### 5.4 触发器建立

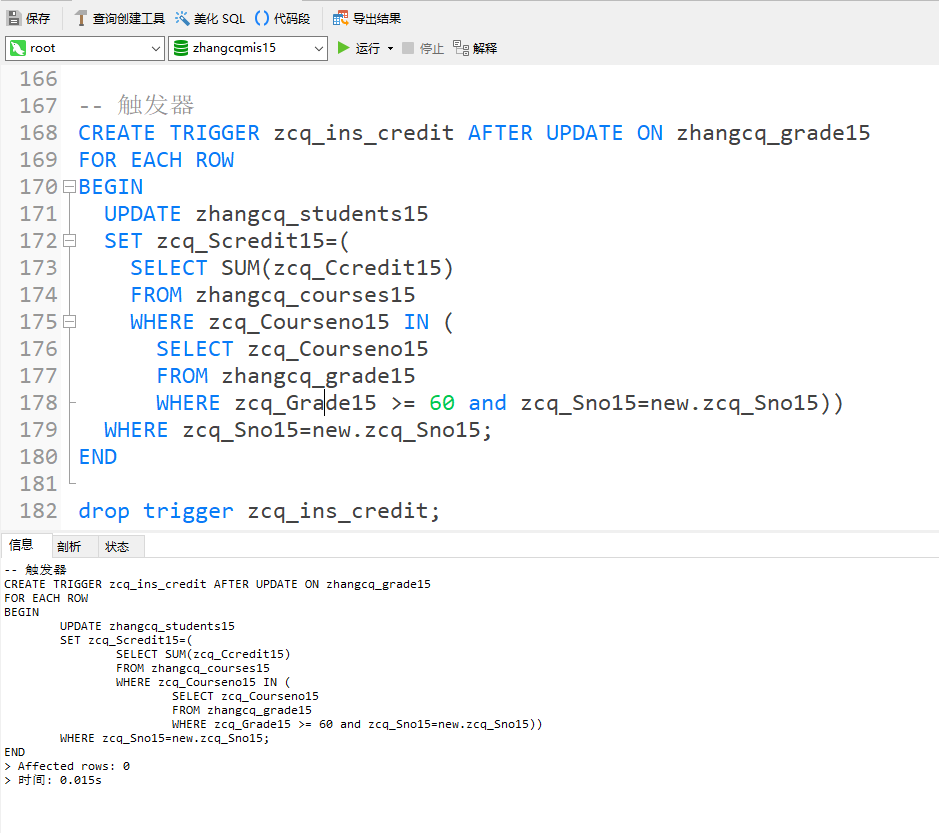


图5.4.1 创建触发器”自动更新学生学分”

描述：当教师录入这个学生的成绩时，这个触发器会被调用，从而根据学生的成绩表，对学生个人的已修学分这一数据项进行更新。

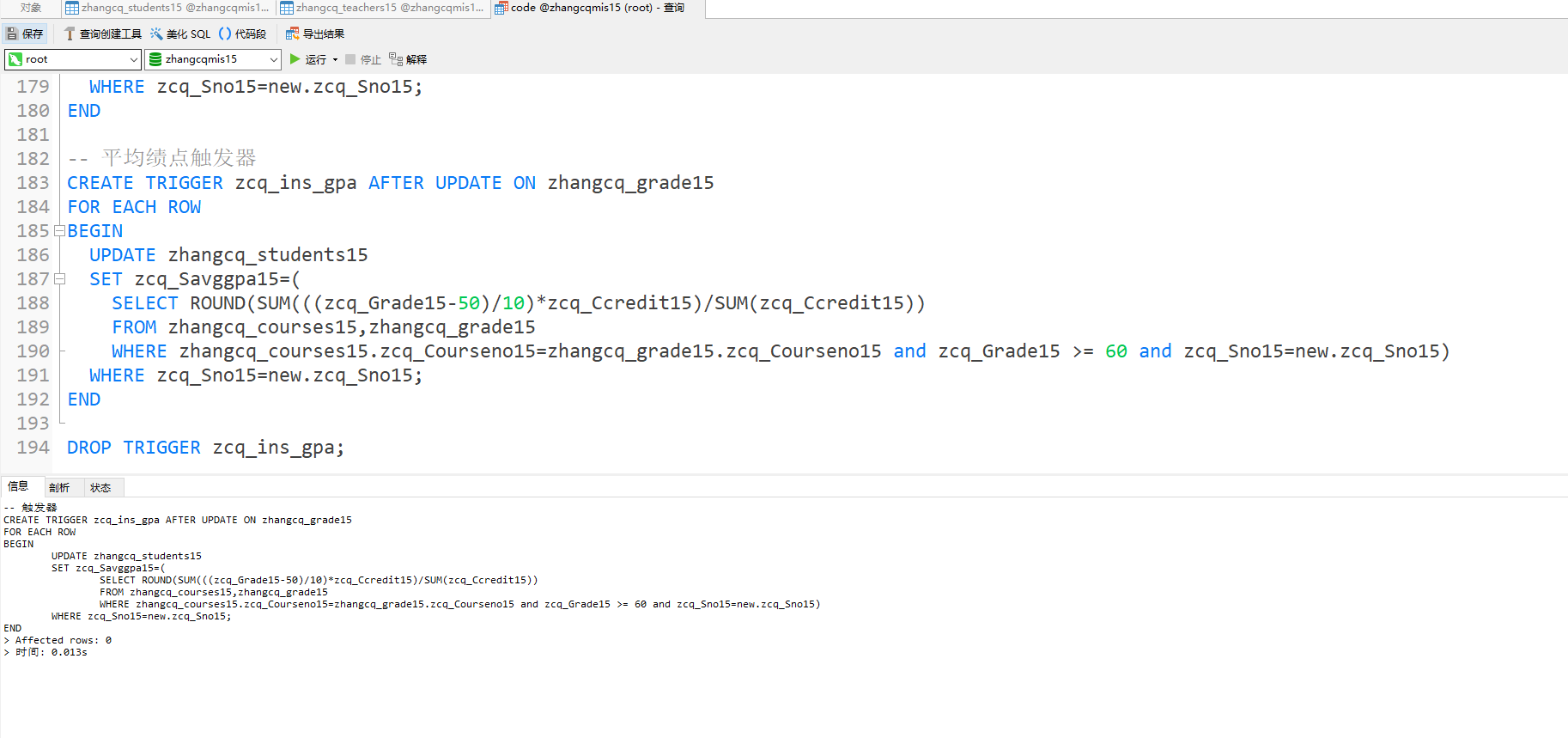


图5.4.2 创建触发器”自动更新学生平均绩点”

描述：当教师录入这个学生的成绩时，这个触发器会被调用，从而根据学生的成绩表，对学生个人的平均绩点这一数据项进行更新。

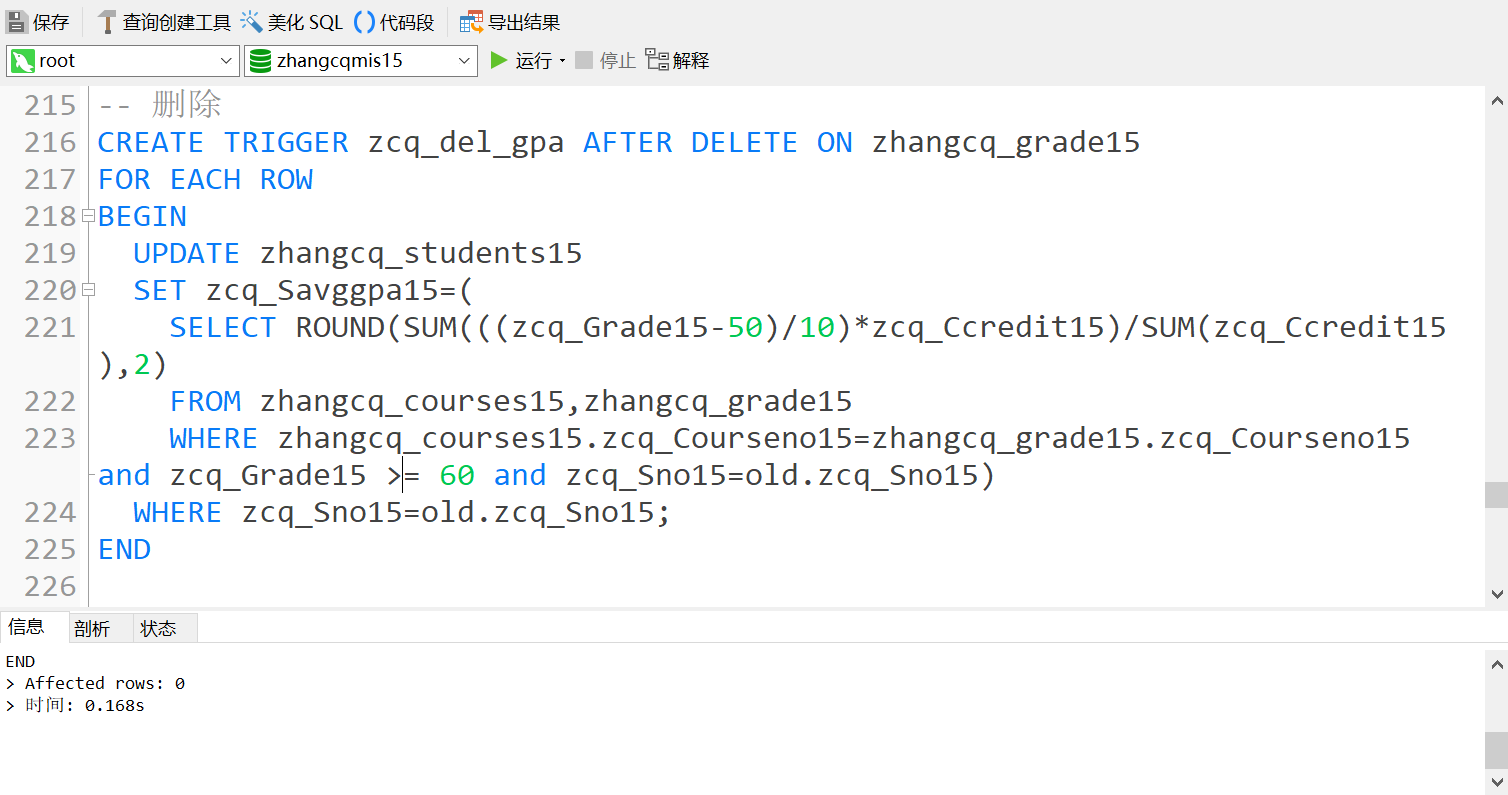


图5.4.3 创建触发器”删除学生成绩自动更新学分”

描述：当教师删除这个学生的成绩时，这个触发器会被调用，从而根据学生的成绩表，对学生个人的已修学分这一数据项进行更新。

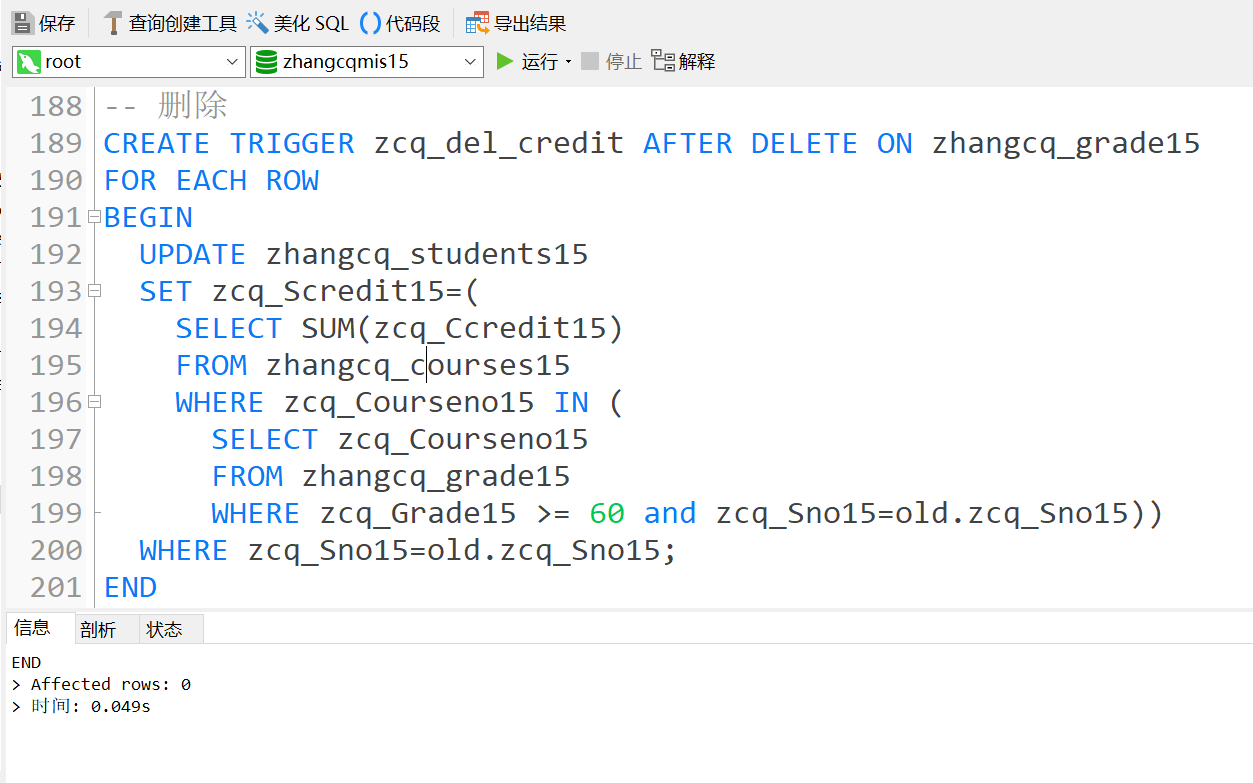


图5.4.4 创建触发器”删除学生成绩自动更新绩点”

描述：当教师删除这个学生的成绩时，这个触发器会被调用，从而根据学生的成绩表，对学生个人的平均绩点这一数据项进行更新。

### 5.5 建存储过程

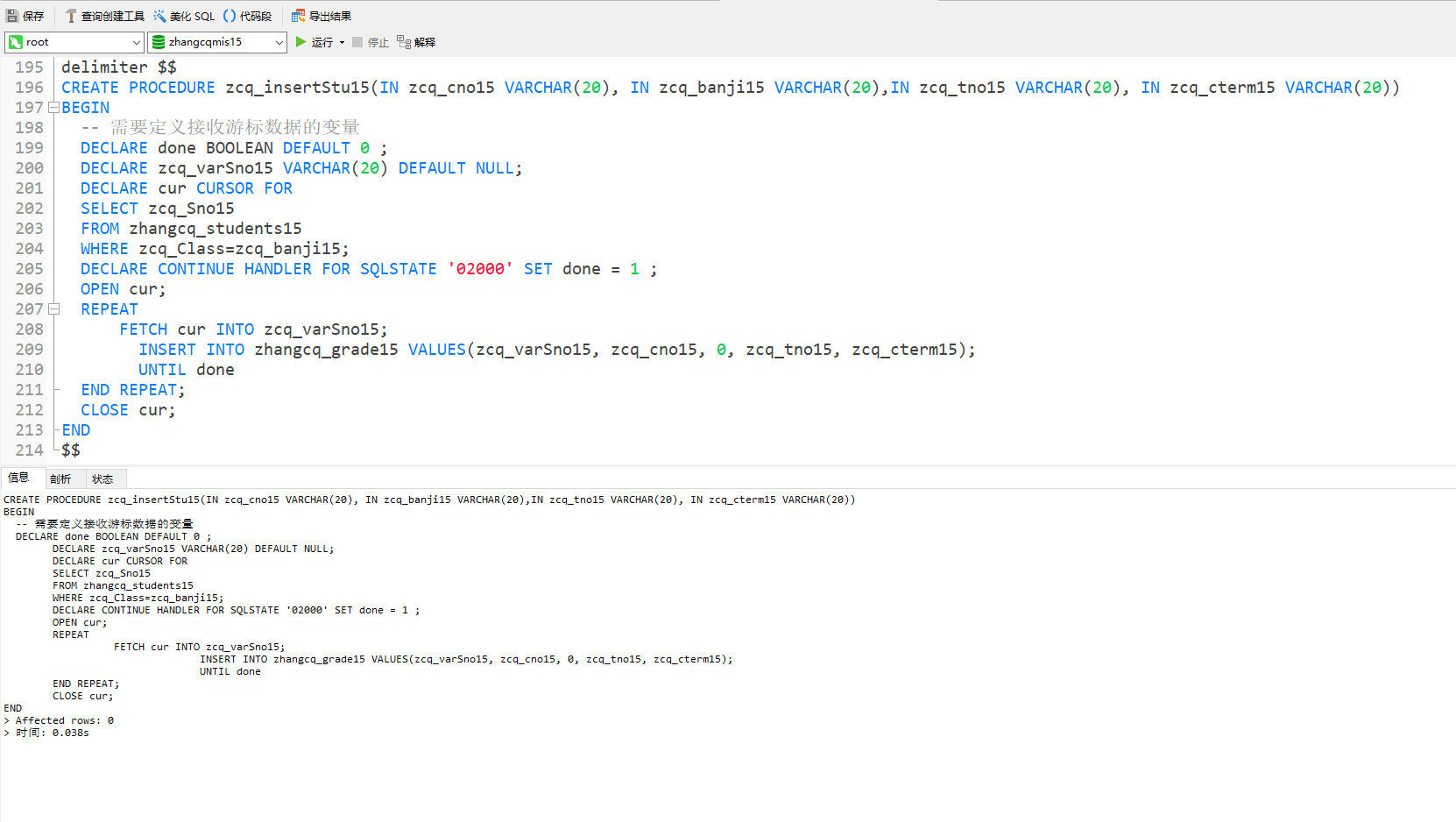


图5.5.1 创建存储过程”为一个班级的学生添加课程记录”

描述：当管理员想要为一个班级开课时，管理员会调用该存储过程，这个存储过程将会去学生表中检索出所有属于给定班级的学生，然后用游标的方式，将这些学生一一加入到成绩表中，这时他们的成绩是null，说明教师还没有录入成绩。

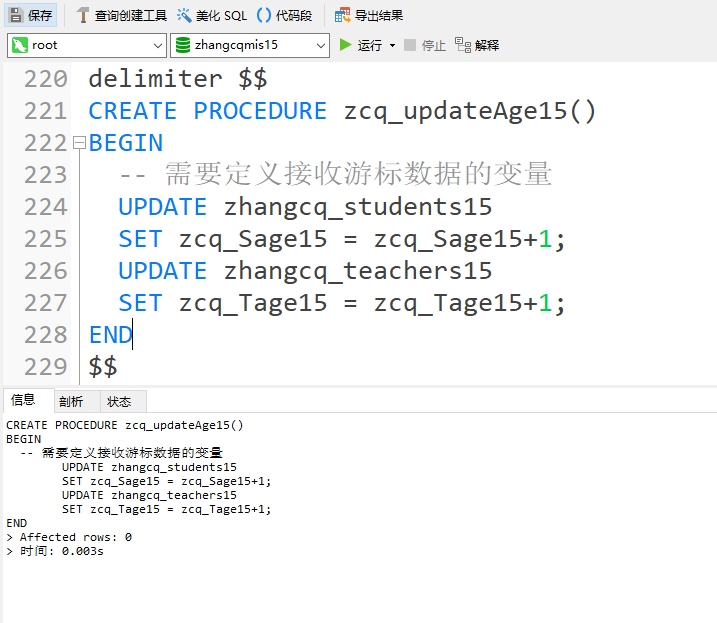


图5.5.2 创建存储过程”为所有老师学生年龄加一”

描述：当时间过完一年时，系统将会自动调用这个存储过程。这个存储过程会将学生和教师的年龄都自动的加上一。

## 6、应用系统开发与试运行

### 6.1 开发平台和开发环境介绍。

本实验由IntelliJ IDEA 2020.3.3 x64开发，运用了javaBean，Servlet，DAO等框架，由Servlet进行处理数据，并且设置了过滤器，以免用户使用权限不足的功能。数据库存储用了mysql数据库。后台与数据库的连接主要通过数据池进行连接，它可以进一步提高并发的效率和速度。在前端bootstrap模板的基础上进行进一步的UI设计。最终，本款产品在jdk8.0的环境下，试运行成功。

### 6.2 前台界面与后台数据库连接说明，代码实现。

本实验采取mysql数据库，由java用数据源的方式进行链接。

<?xml version="1.0" encoding="utf-8"?>  
<Context reloadable = "true">  
<Resource  
 name="jdbc/sampleDS"  
 type="javax.sql.DataSource"  
 driverClassName="com.mysql.cj.jdbc.Driver"  
 url="jdbc:mysql://localhost:3306/zhangcqmis15?useUnicode=true&amp;characterEncoding=utf-8&amp;useSSL=false&amp;serverTimezone = GMT"  
 username="root"  
 password="zerostart"  
 maxActive="400"  
 maxIdle="20"  
 maxWait="500"  
/>  
</Context>

首先，先在context.xml中配置好mysql的相应信息，包括驱动Driver所在地，mysql链接账户名和密码，允许的最大链接数和最大等待时间。

之后，由BaseDAO负责建立链接并调用。其他DAO对BaseDAO进行继承就可以用sql语句对数据库进行操作了。

package com.example.final\_database.dao;  
  
import javax.naming.Context;  
import javax.naming.InitialContext;  
import javax.naming.NamingException;  
import javax.sql.DataSource;  
import java.sql.Connection;  
  
public class Basedao {  
 DataSource dataSource;  
 public Basedao(){  
 try{  
  
 Context context = new InitialContext();  
 dataSource =  
 (DataSource)context.lookup("java:comp/env/jdbc/sampleDS");  
 }catch (NamingException ne){  
 System.*out*.println("Exception:"+ne);  
 }  
 }  
 public Connection getConnection()throws Exception{  
 return dataSource.getConnection();  
 }  
  
}

### 6.3 系统各功能设计和运行界面截图。

**6.3.1学生信息管理系统登录界面**



图6.3.1 学生成绩管理系统登录展示图

功能设计：在这个页面，用户填写好表单后交到后台进行审核，然后根据用户类型到相应的表中搜索账号密码，若账号密码输入正确。如果是学生则跳转到学生界面，如果是老师或管理员，也相应的跳转到对应界面。

后端代码：

package com.example.final\_database.controller;  
  
import com.example.final\_database.dao.loginDao;  
import com.example.final\_database.model.Student;  
import com.example.final\_database.model.Teacher;  
  
import javax.servlet.\*;  
import javax.servlet.http.\*;  
import javax.servlet.annotation.\*;  
import java.io.IOException;  
  
@WebServlet(name = "loginServlet", value = "/loginServlet")  
public class loginServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 }  
  
 @Override  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 request.setCharacterEncoding("utf-8"); //1  
 response.setContentType("text/html;charset=utf-8"); //2  
 response.setCharacterEncoding("utf-8"); //3  
 String kind =request.getParameter("kind");  
 String username = request.getParameter("username");  
 String password = request.getParameter("password");  
 HttpSession session=request.getSession();  
 String message=null;  
 RequestDispatcher dispatcher=null;  
 loginDao userdao=new loginDao();  
 if(kind.equals("student")){  
  
 Student user=userdao.StudentLogin(username,password);  
 if(user==null){  
 message="用户名或密码错误";  
 dispatcher=request.getRequestDispatcher("./VIEW/index.jsp");  
 }  
 else{  
 message="登录成功！欢迎!";  
 session.setAttribute("user",user);  
 dispatcher=request.getRequestDispatcher("./VIEW/student.jsp");  
 }  
 session.setAttribute("message",message);  
 session.setAttribute("kind","学生");  
 dispatcher.forward(request,response);  
 }  
 else if(kind.equals("teacher")){  
 Teacher user=userdao.TeacherLogin(username,password);  
 if(user==null){  
 message="用户名或密码错误";  
 dispatcher=request.getRequestDispatcher("./VIEW/index.jsp");  
 }  
 else{  
 message="登录成功！欢迎!";  
 session.setAttribute("user",user);  
 if(user.getTauthority()<2) {  
 dispatcher = request.getRequestDispatcher("./VIEW/teacher.jsp");  
 }  
 else{  
 dispatcher = request.getRequestDispatcher("./VIEW/admin.jsp");  
 }  
 }  
 session.setAttribute("message",message);  
 session.setAttribute("kind","老师");  
 dispatcher.forward(request,response);  
 }  
 }  
}

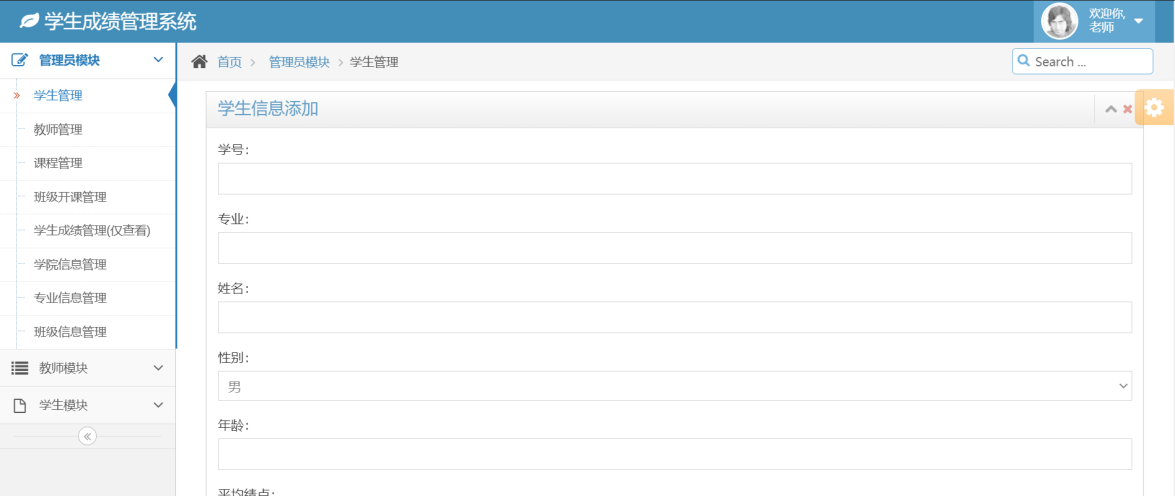
**6.3.2管理员**

输入账号密码后，进行登录将会跳转到界面一。





**界面一：学生管理界面**



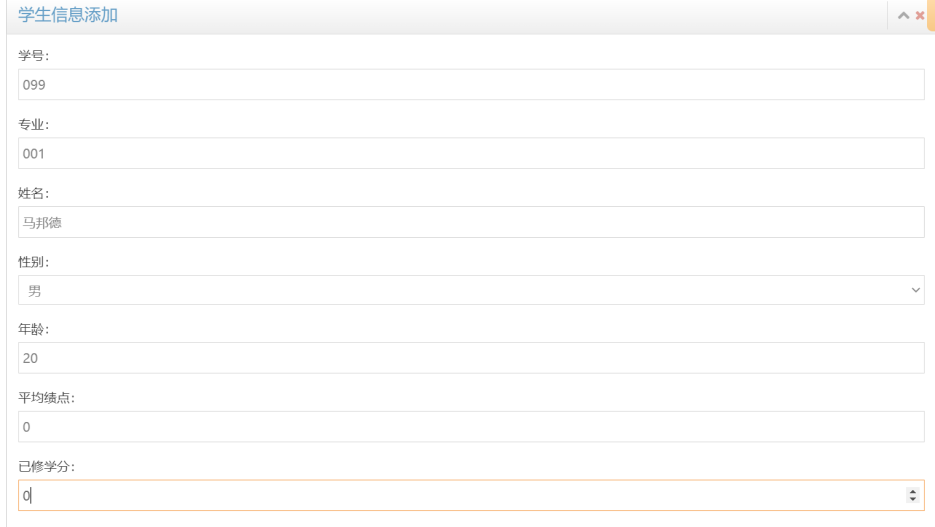
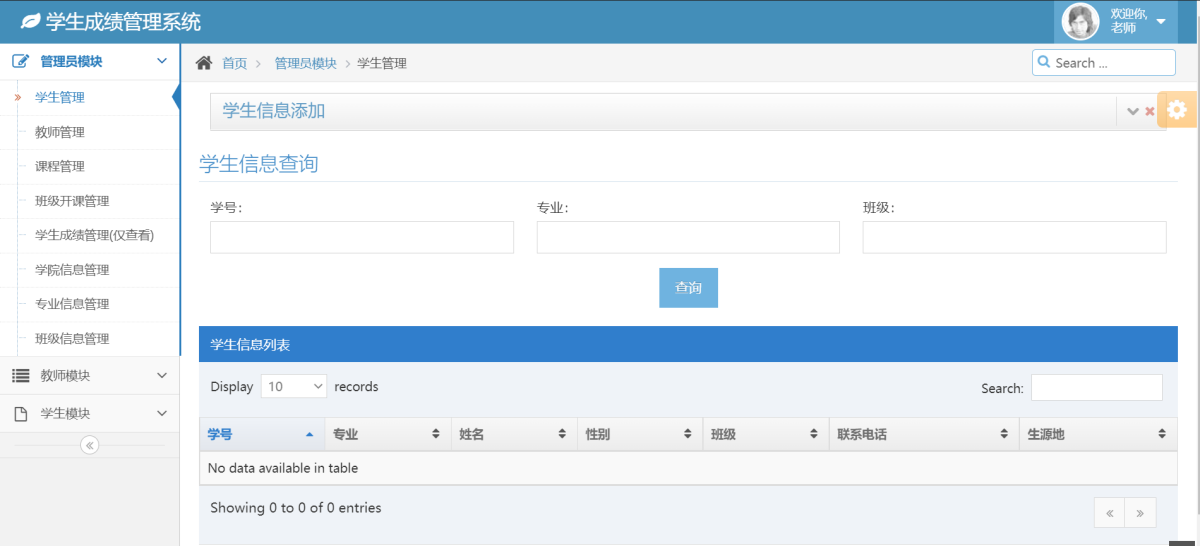




图6.3.2.1 学生信息管理展示图：学生添加



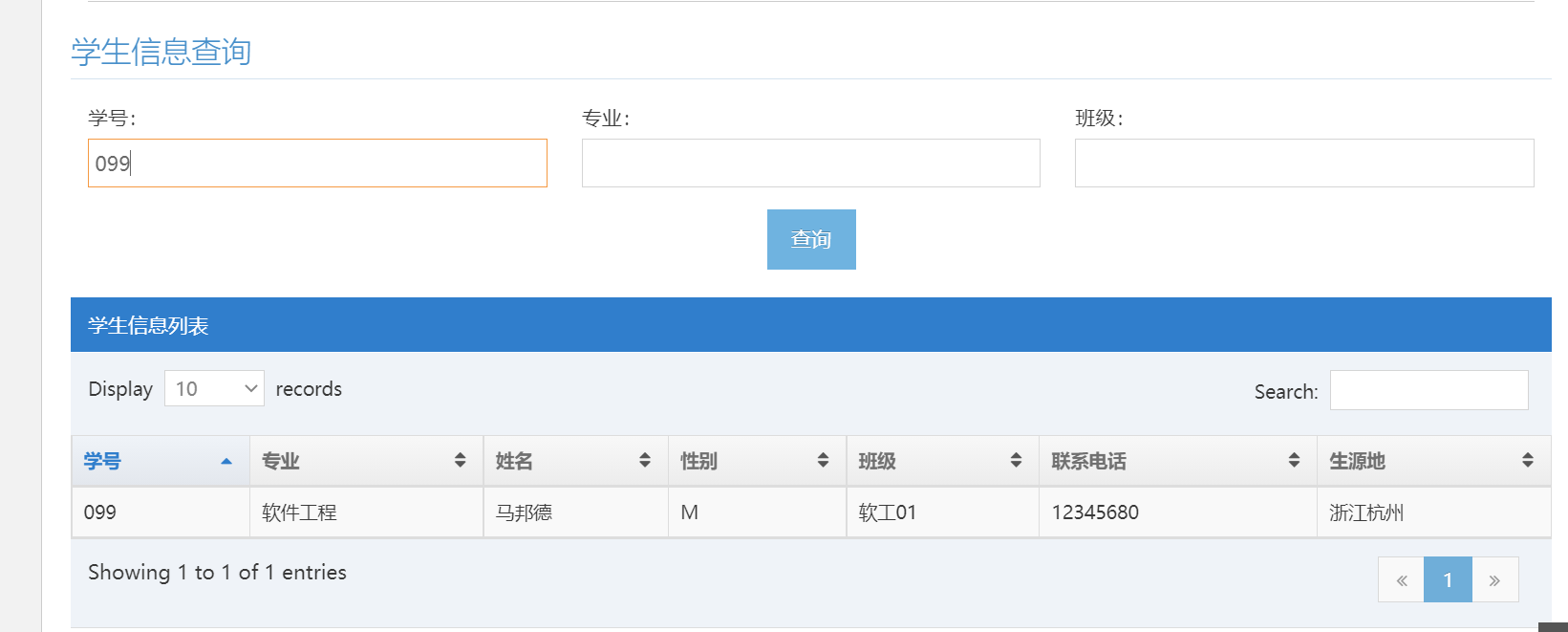


图6.3.2.2 学生信息管理展示图：学生查询

功能设计：这个界面一共有两个功能，一是学生添加，二是学生查询。学生添加主要是通过表单交给后台，后台通过INSERT语句插入信息到相应的表中。学生查询则是通过zhangcq\_stuinfo15视图进行SELECT查询，因为普通表中的字段都是编号，对管理人员使用十分不友善，所以专门新建了一个视图进行查询。

核心代码：

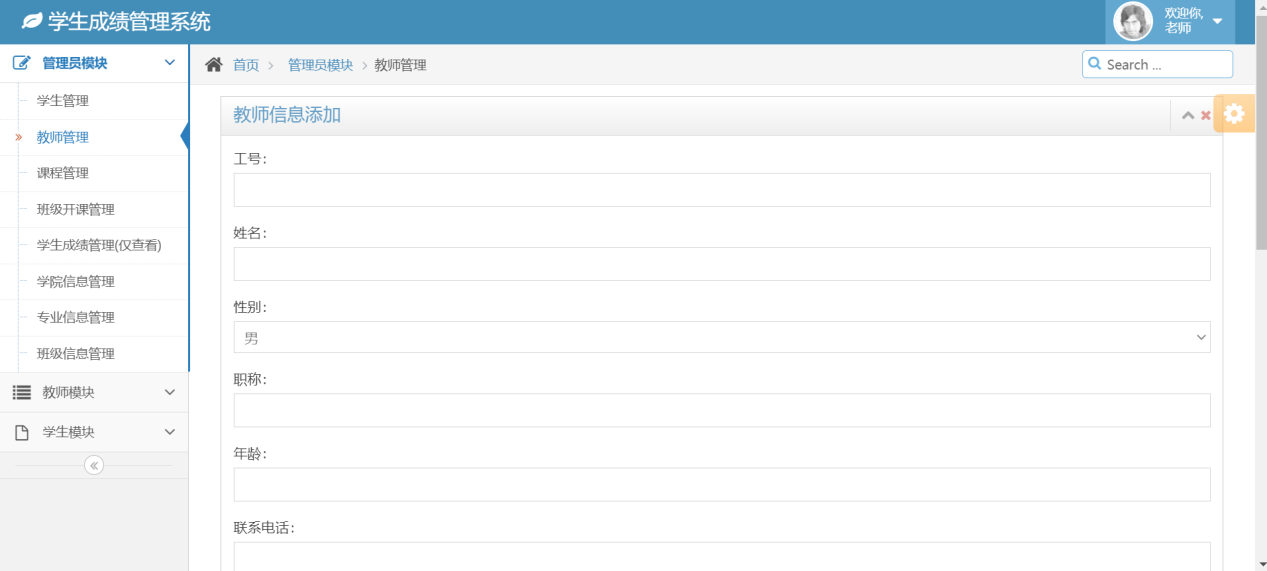
（学生添加）

public boolean addStudennt(String Sno,String Mno ,String Sname,String Ssex,String Sage,String Savggpa,String Scredit,String Sphone,String Shometown,String Spwd,String Classno) {  
 String sql = "INSERT INTO zhangcq\_students15 VALUES(?,?,?,?,?,?,?,?,?,?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Sno);  
 psmt.setString(2,Mno);  
 psmt.setString(3,Sname);  
 psmt.setString(4,Ssex);  
 psmt.setInt(5,Integer.*parseInt*(Sage));  
 psmt.setDouble(6,Double.*parseDouble*(Savggpa));  
 psmt.setDouble(7,Double.*parseDouble*(Scredit));  
 psmt.setString(8,Sphone);  
 psmt.setString(9,Shometown);  
 psmt.setString(10,Spwd);  
 psmt.setString(11,Classno);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

学生查询：

public ArrayList<Student> queryStudent(String Sno, String Mno, String Classname) {  
 String sql="select \* from zhangcq\_stuinfo15 WHERE 1=1";  
 if(!Sno.equals("")){  
 sql=sql+" and zcq\_Sno15='"+Sno+"'";  
 }  
 if(!Mno.equals("")){  
 sql=sql+" and zcq\_Mname15='"+Mno+"'";  
 }  
 if(!Classname.equals("")){  
 sql=sql+" and zcq\_Classname15='"+Classname+"'";  
 }  
 ArrayList<Student> list=new ArrayList<Student>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Student student=new Student();  
 student.setSno(rst.getString(1));  
 student.setMajor(rst.getString(2));  
 student.setSname(rst.getString(3));  
 student.setSsex(rst.getString(4));  
 student.setBanji(rst.getString(5));  
 student.setSphone(rst.getString(6));  
 student.setShometown(rst.getString(7));  
 list.add(student);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面二：教师管理界面**



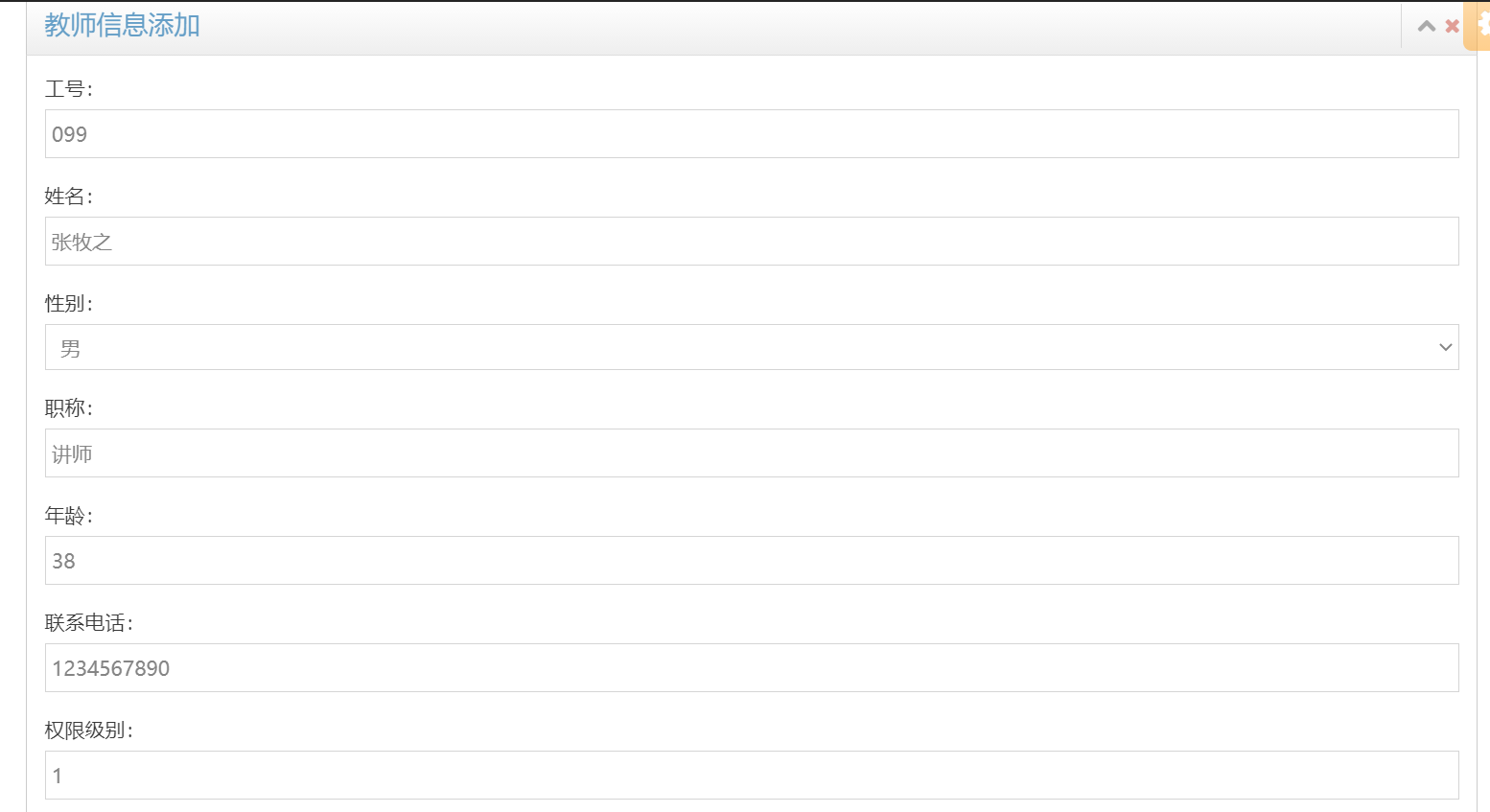
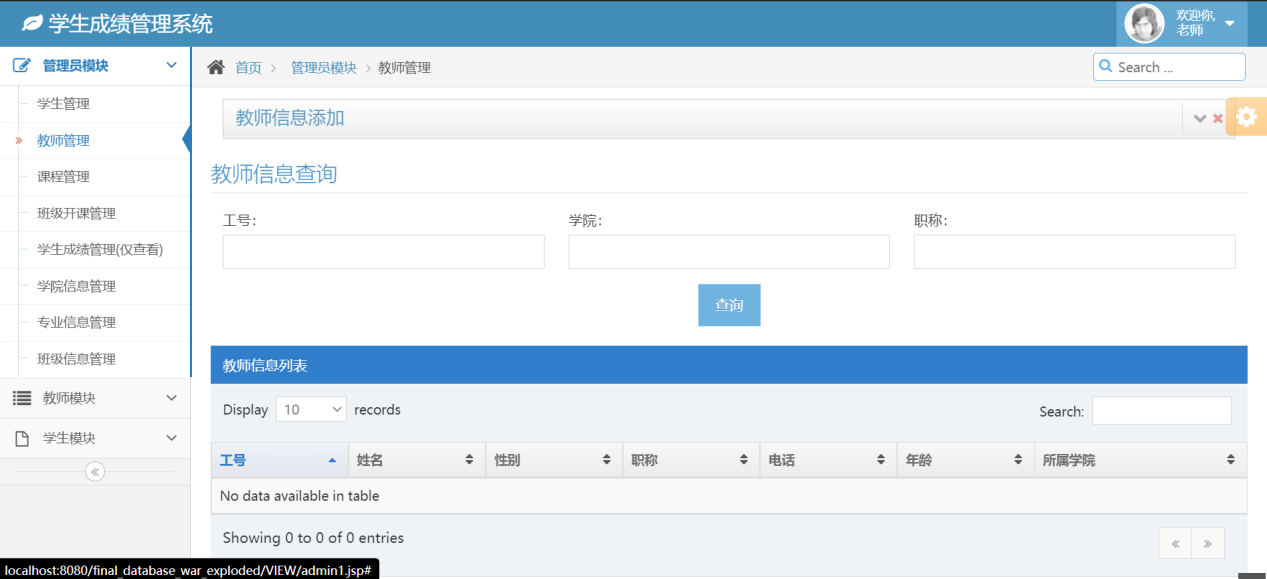




图6.3.2.3 教师信息管理展示图：教师添加



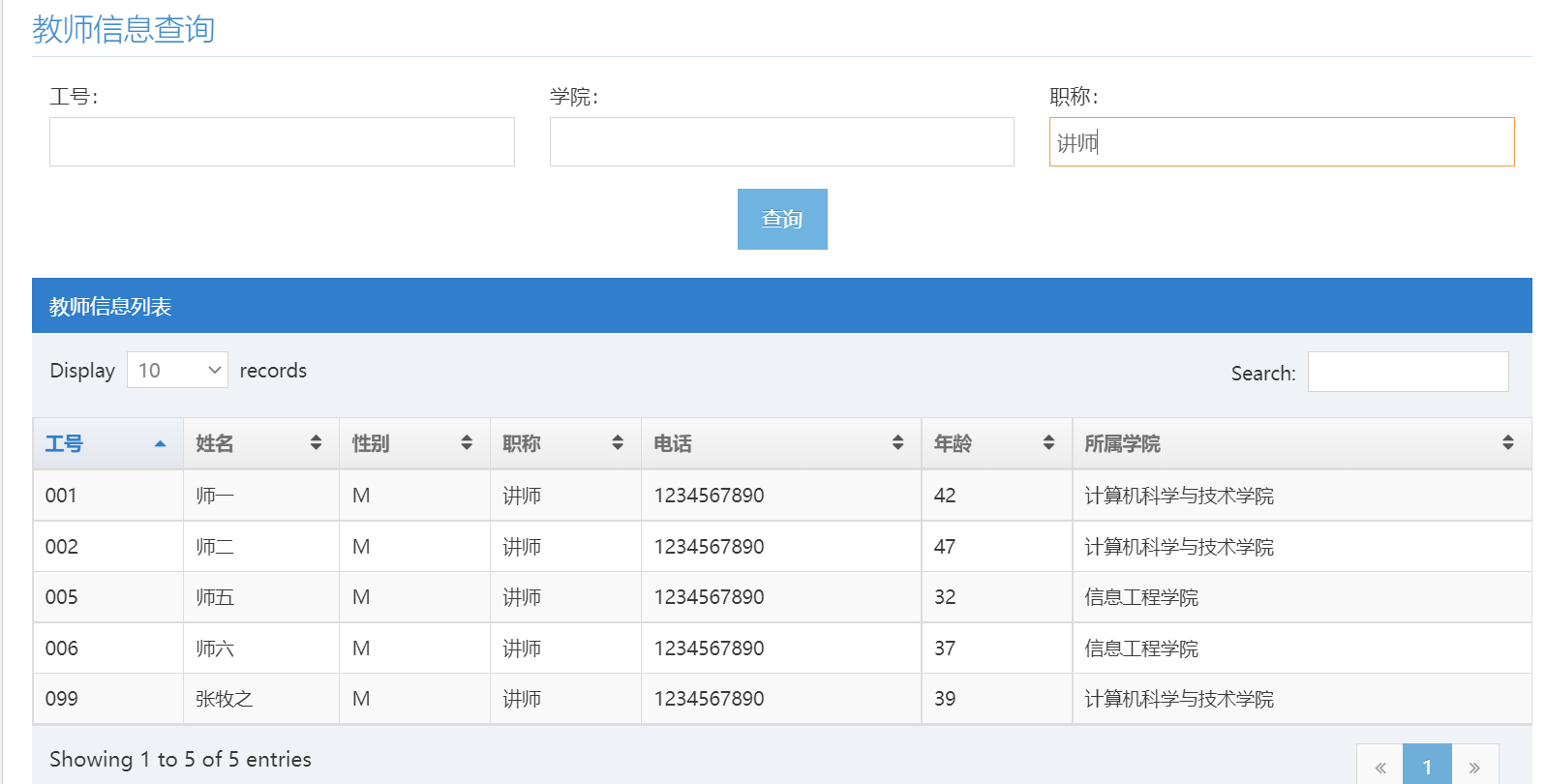


图6.3.2.4 教师信息管理展示图：教师查询

功能设计：这个界面一共有两个功能，一是教师添加，二是教师查询。教师添加主要是通过表单交给后台，后台通过INSERT语句插入信息到相应的表中。教师查询则是通过zhangcq\_teainfo15视图进行SELECT查询，因为普通表中的字段都是编号，对管理人员使用十分不友善，所以专门新建了一个视图进行查询。

核心代码：

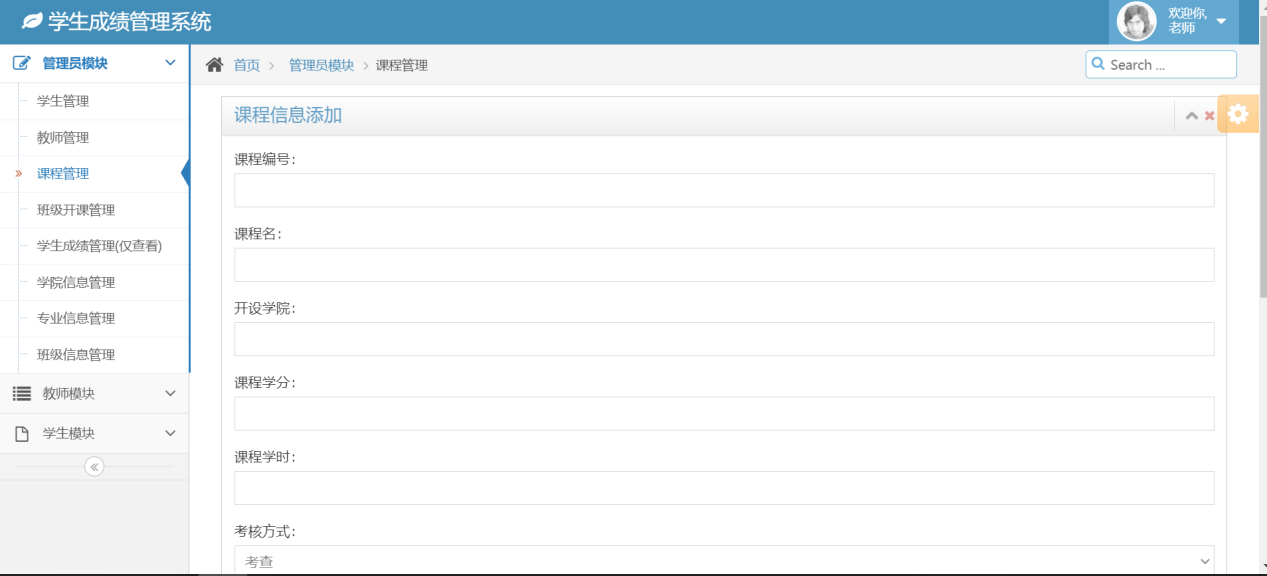
(教师添加)

public boolean addTeacher(String tno,String tname ,String tsex,String tprofess,String tage,String tphone,String tauthority,String tpwd,String collegeno) {  
 String sql = "INSERT INTO zhangcq\_teachers15 VALUES(?,?,?,?,?,?,?,?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,tno);  
 psmt.setString(2,tname);  
 psmt.setString(3,tsex);  
 psmt.setString(4,tprofess);  
 psmt.setInt(5,Integer.*parseInt*(tage));  
 psmt.setString(6,tphone);  
 psmt.setInt(7,Integer.*parseInt*(tauthority));  
 psmt.setString(8,tpwd);  
 psmt.setString(9,collegeno);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

(教师查询)

public ArrayList<Teacher> queryTeacher(String Tno, String College, String profess) {  
 String sql="select \* from zhangcq\_teainfo15 WHERE 1=1";  
 if(!Tno.equals("")){  
 sql=sql+" and zcq\_Tno15='"+Tno+"'";  
 }  
 if(!College.equals("")){  
 sql=sql+" and zcq\_Collegename15='"+College+"'";  
 }  
 if(!profess.equals("")){  
 sql=sql+" and zcq\_Tprofess15='"+profess+"'";  
 }  
 ArrayList<Teacher> list=new ArrayList<Teacher>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Teacher teacher=new Teacher();  
 teacher.setTno(rst.getString(1));  
 teacher.setTname(rst.getString(2));  
 teacher.setTsex(rst.getString(3));  
 teacher.setTprofess(rst.getString(4));  
 teacher.setTphone(rst.getString(5));  
 teacher.setTage(rst.getInt(6));  
 teacher.setCollegename(rst.getString(7));  
 list.add(teacher);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面三：课程管理界面**



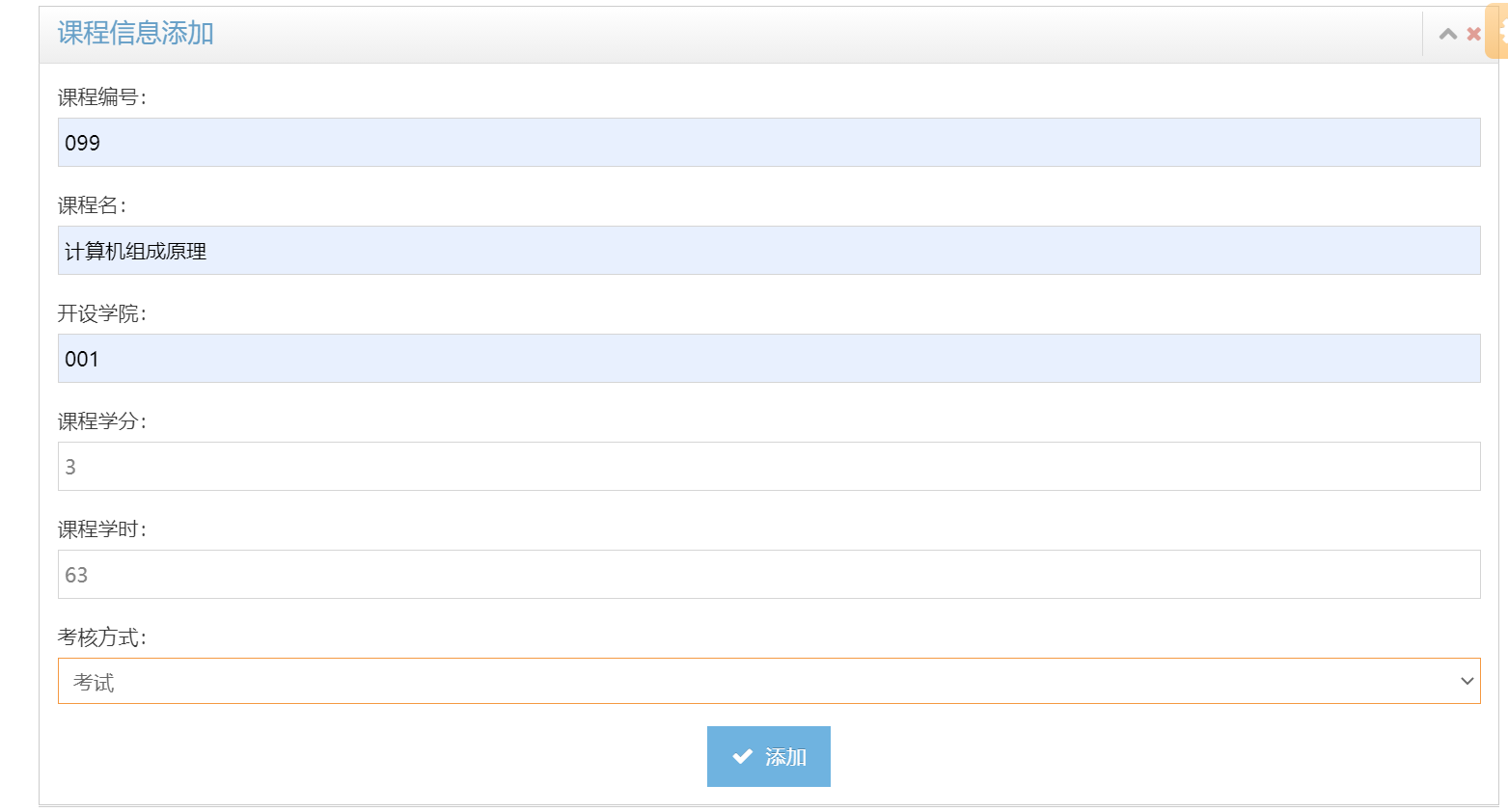




图6.3.2.5 课程信息管理展示图：课程查询

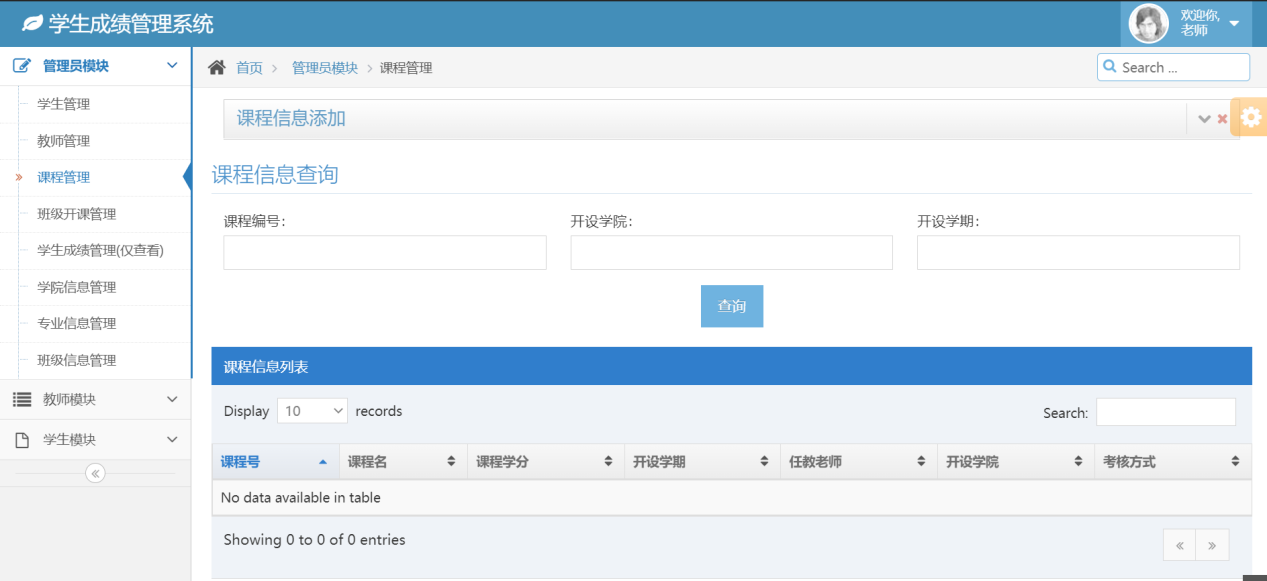




图6.3.2.6 课程信息管理展示图：课程查询

功能设计：这个界面一共有两个功能，一是课程添加，二是课程查询。课程添加主要是通过表单交给后台，后台通过INSERT语句插入信息到相应的表中。课程查询则是通过zhangcq\_couinfo15视图进行SELECT查询，因为普通表中的字段都是编号，对管理人员使用十分不友善，所以专门新建了一个视图进行查询。

核心代码：

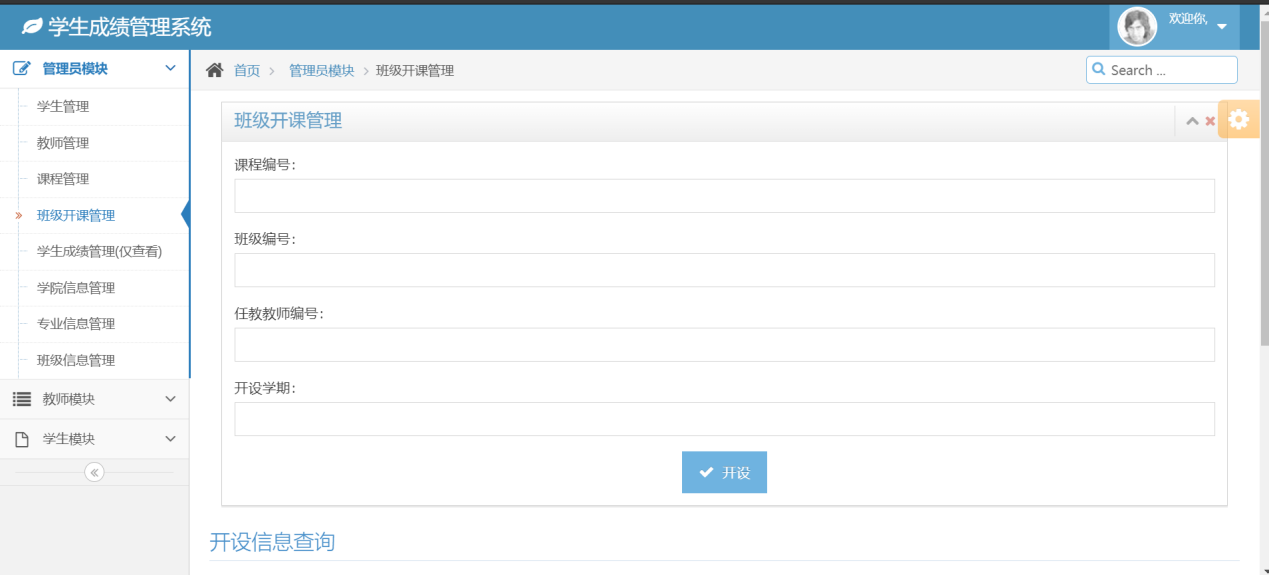
(课程添加)

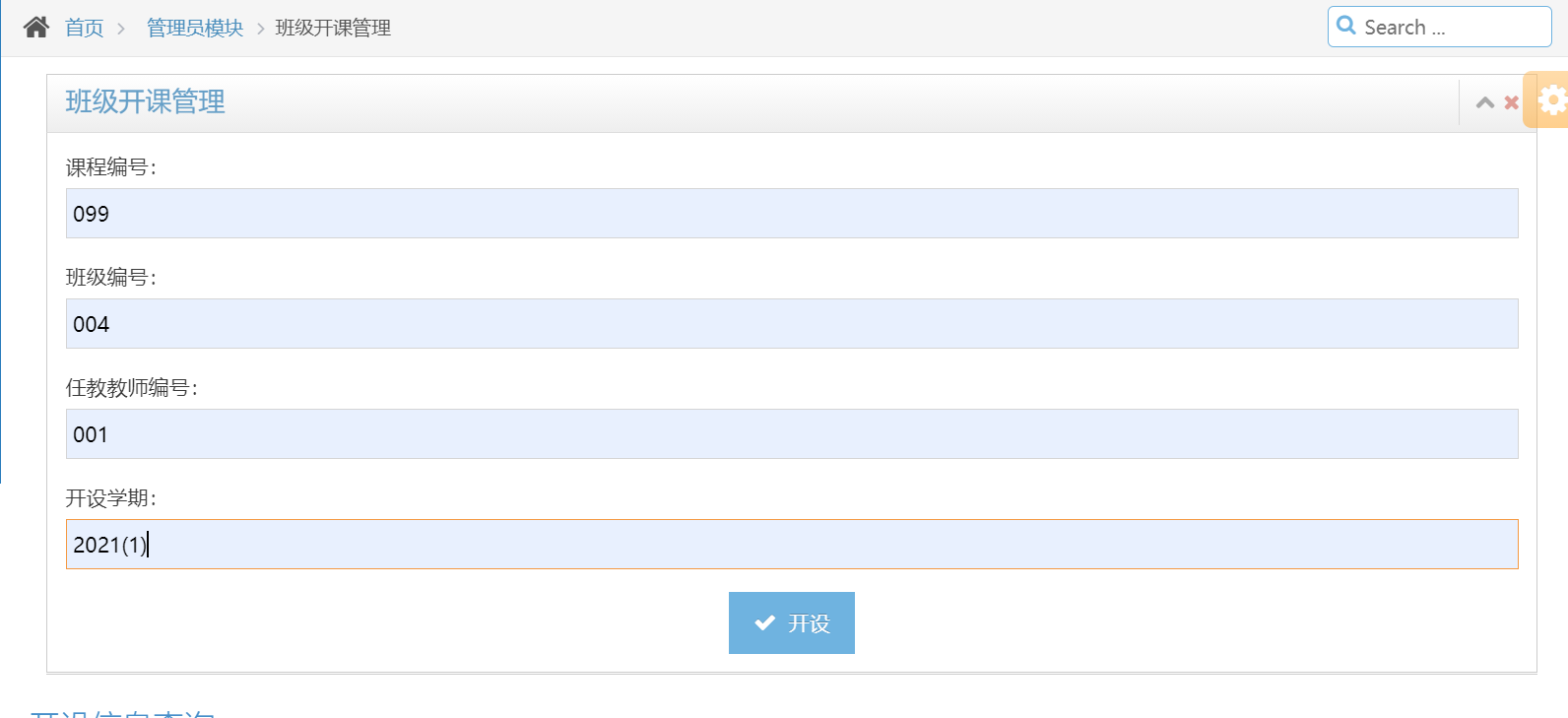
public boolean addCourse(String Courseno,String Coursename ,String Collegeno,String Ccredit,String Chours,String Cassessment) {  
 String sql = "INSERT INTO zhangcq\_courses15 VALUES(?,?,?,?,?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Courseno);  
 psmt.setString(2,Coursename);  
 psmt.setString(3,Collegeno);  
 psmt.setDouble(4,Double.*parseDouble*(Ccredit));  
 psmt.setInt(5,Integer.*parseInt*(Chours));  
 psmt.setString(6,Cassessment);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

(课程查询)

public ArrayList<Course> queryCourse(String Courseno, String Collegename, String Cterm) {  
 String sql="select distinct \* from zhangcq\_couinfo15 WHERE 1=1";  
 if(!Courseno.equals("")){  
 sql=sql+" and zcq\_Courseno15='"+Courseno+"'";  
 }  
 if(!Collegename.equals("")){  
 sql=sql+" and zcq\_Collegename15='"+Collegename+"'";  
 }  
 if(!Cterm.equals("")){  
 sql=sql+" and zcq\_Cterm15='"+Cterm+"'";  
 }  
 ArrayList<Course> list=new ArrayList<Course>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Course course=new Course();  
 course.setCourseno(rst.getString(1));  
 course.setCoursename(rst.getString(2));  
 course.setCcredit(rst.getDouble(3));  
 course.setCterm(rst.getString(4));  
 course.setTname(rst.getString(5));  
 course.setCollegename(rst.getString(6));  
 course.setCassessment(rst.getString(7));  
 list.add(course);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面四：班级开课管理界面**





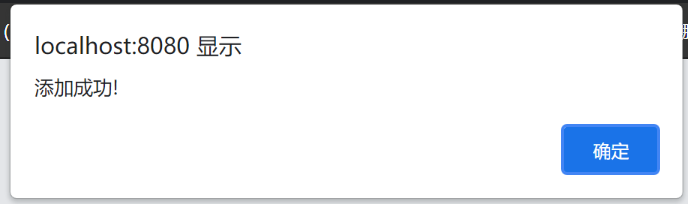
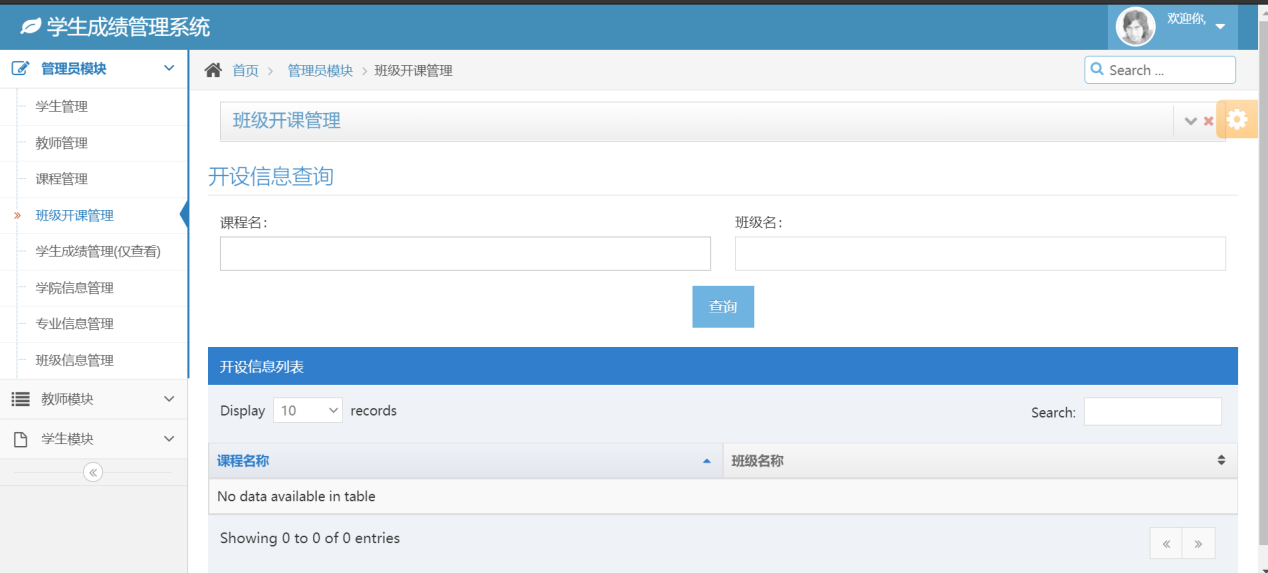


图6.3.2.7 开课信息信息管理展示图：班级开课



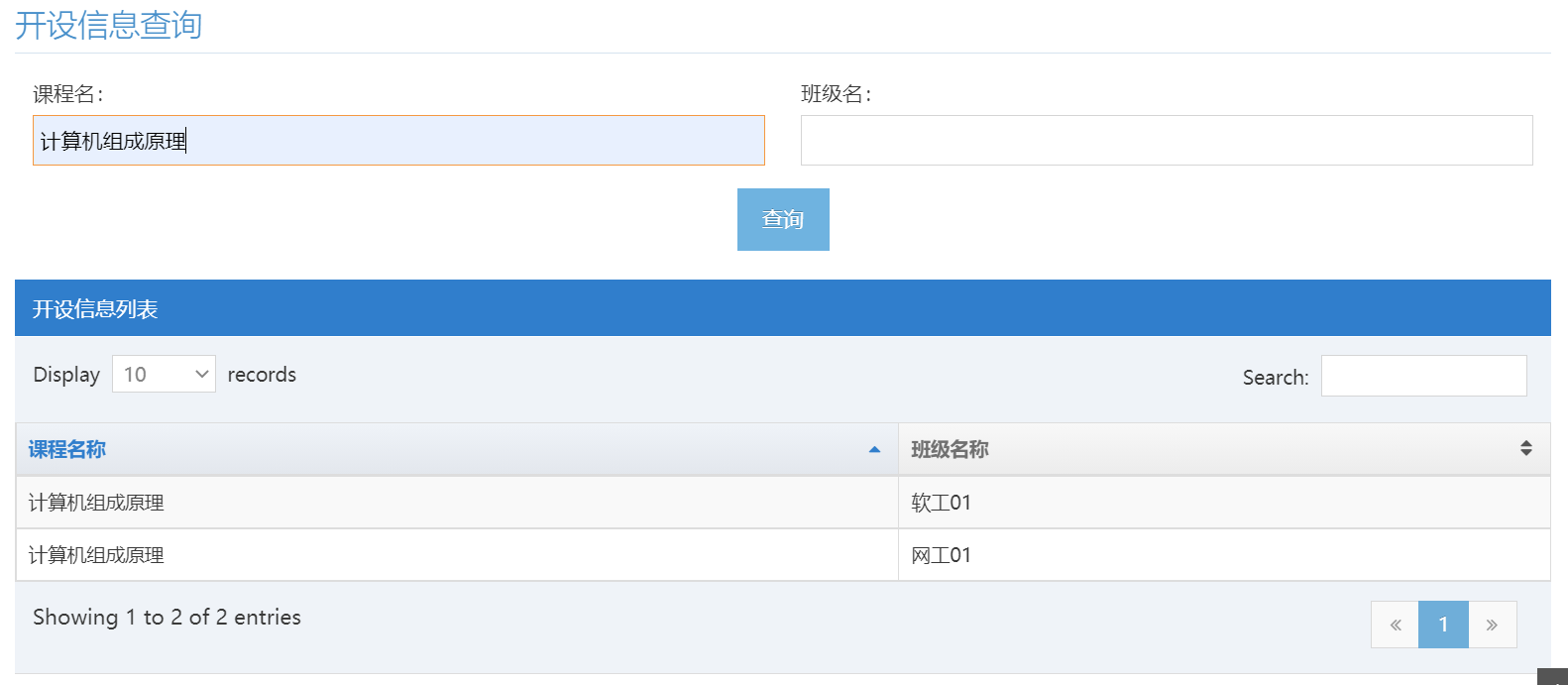


图6.3.2.8 开课信息管理展示图：开课信息查询

功能设计：这个界面一共有两个功能，一是开课信息添加，二是开设课程信息查询。课程添加主要是通过表单交给后台，后台通过事先设计的存储过程zcq\_insertStu15，将班级中的每一个学生，新建信息到zhangcq\_grade15表中。课程查询则是通过zhangcq\_setinfo15视图进行SELECT查询，因为普通表中的字段都是编号，对管理人员使用十分不友善，所以专门新建了一个视图进行查询。

核心代码

(开设课程)

public boolean addSetupinfo(String Courseno,String Classno,String Tno,String Cterm) {  
 String sql = "INSERT INTO zhangcq\_setupinfo15 VALUES(?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Courseno);  
 psmt.setString(2,Classno);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}  
public boolean addSetupinfo1(String Courseno,String Classno,String Tno,String Cterm) {  
 String sql = "CALL zcq\_insertStu15(?,?,?,?);";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Courseno);  
 psmt.setString(2,Classno);  
 psmt.setString(3,Tno);  
 psmt.setString(4,Cterm);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

(查询课程)

public ArrayList<Setupinfo> querySetup(String Coursename, String Classname) {  
 String sql="select distinct \* from zhangcq\_setinfo15 WHERE 1=1";  
 if(!Coursename.equals("")){  
 sql=sql+" and zcq\_Coursename15='"+Coursename+"'";  
 }  
 if(!Classname.equals("")){  
 sql=sql+" and zcq\_Classname15='"+Classname+"'";  
 }  
 ArrayList<Setupinfo> list=new ArrayList<Setupinfo>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Setupinfo setupinfo=new Setupinfo();  
 setupinfo.setCoursename(rst.getString(1));  
 setupinfo.setClassname(rst.getString(2));  
 list.add(setupinfo);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面五：学生成绩管理（仅查看）**

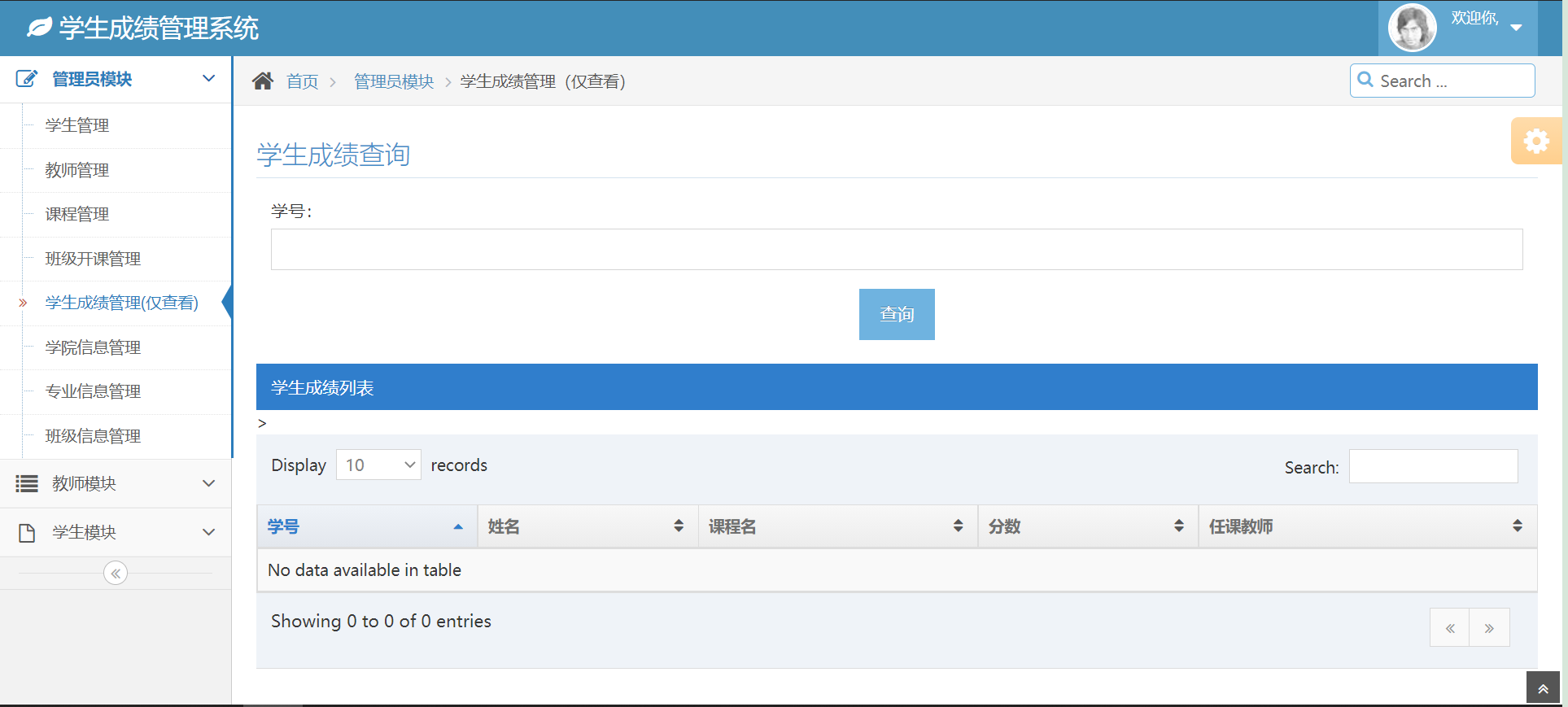




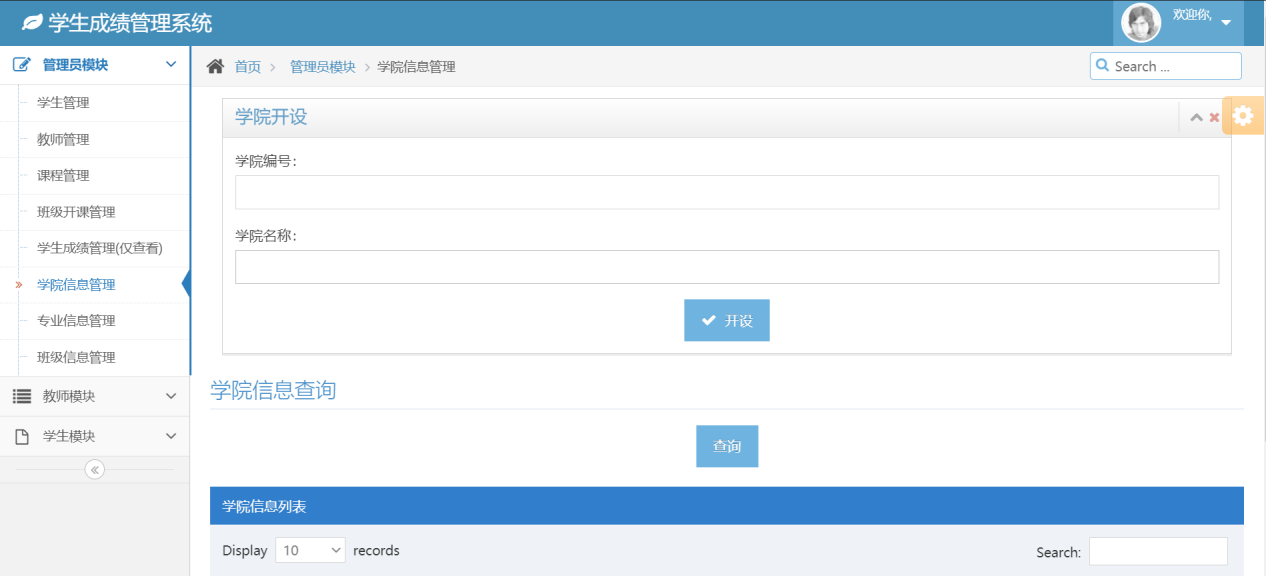
图6.3.2.9 学生成绩管理展示图：学生成绩信息查询

功能设计：这个界面就一个学生成绩查询功能。学生成绩查询则是通过zhangcq\_grainfo15视图进行SELECT查询，因为普通表中的字段都是编号，对管理人员使用十分不友善，所以专门新建了一个视图进行查询。

核心代码：

public ArrayList<Grade> queryGrade(String sno,String cterm,String tno) {  
 String sql="select distinct \* from zhangcq\_grainfo15 WHERE 1=1";  
 if(!sno.equals("")){  
 sql=sql+" and zcq\_Sno15='"+sno+"'";  
 }  
 if(!cterm.equals("")){  
 sql=sql+" and zcq\_Cterm15='"+cterm+"'";  
 }  
 if(!tno.equals("")){  
 sql=sql+" and zcq\_Tno15='"+tno+"'";  
 }  
 ArrayList<Grade> list=new ArrayList<Grade>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Grade grade=new Grade();  
 grade.setSno(rst.getString(1));  
 grade.setSname(rst.getString(2));  
 grade.setCoursename(rst.getString(3));  
 grade.setGrade(rst.getInt(4));  
 grade.setTno(rst.getString(5));  
 grade.setTname(rst.getString(6));  
 grade.setCterm(rst.getString(7));  
 list.add(grade);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面六：学院信息管理**



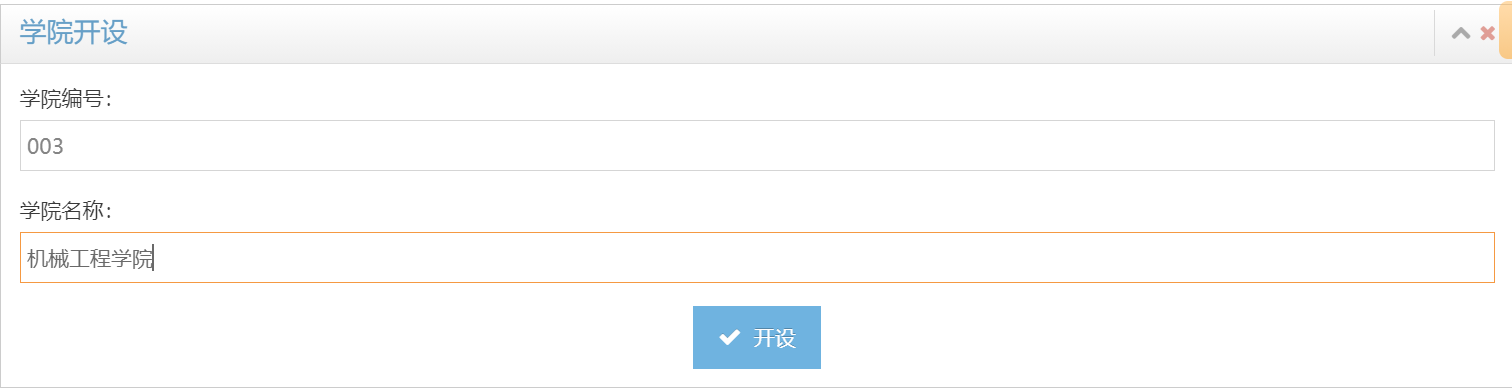




图6.3.2.10 学院管理展示图：学院信息添加

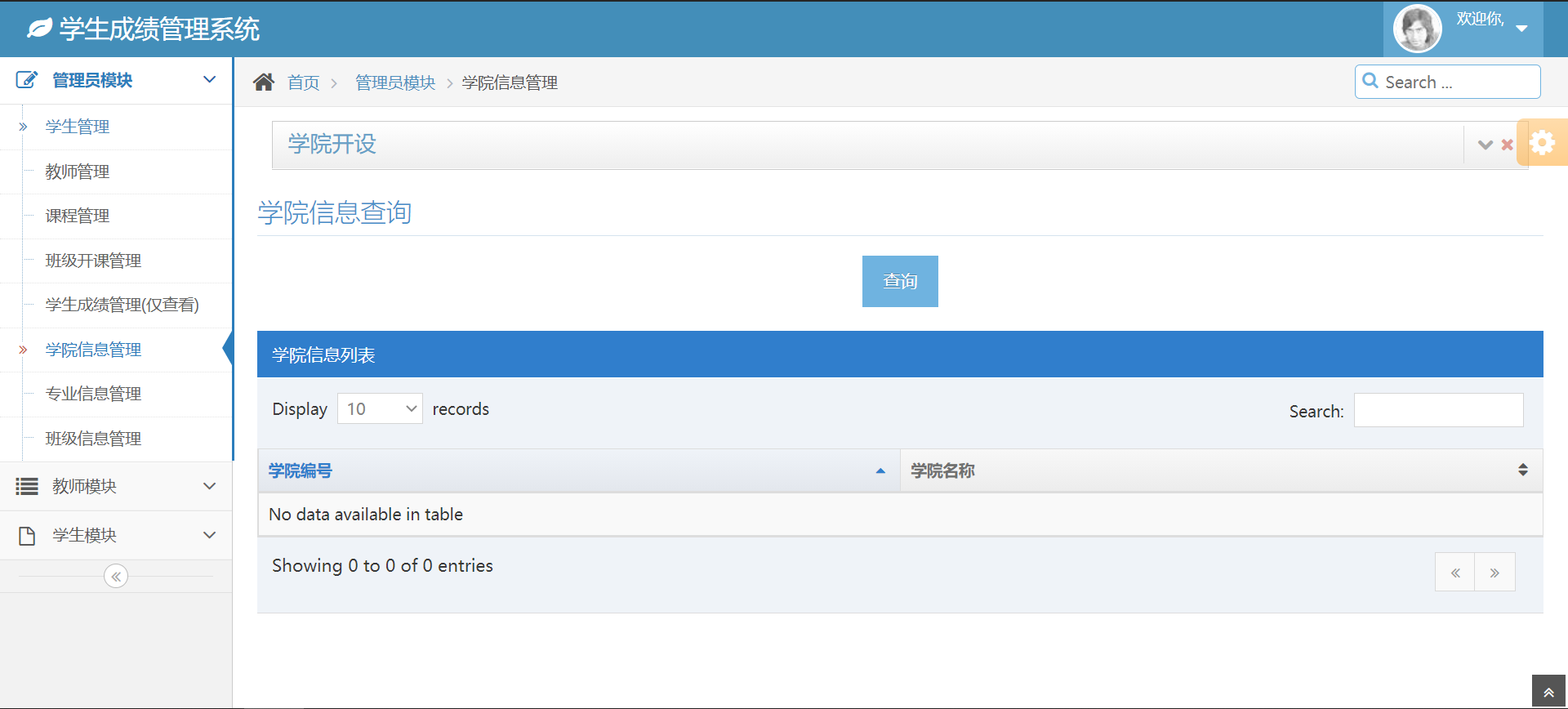




图6.3.2.11 学院管理展示图：学院信息查询

功能设计：这个界面一共有两个功能，一是学院添加，二是学院查询。学院添加主要是通过表单交给后台，后台通过INSERT语句插入信息到相应的表中。学院查询则是通过直接SELECT表zhangcq\_colleges15得到查询结果。

核心代码：

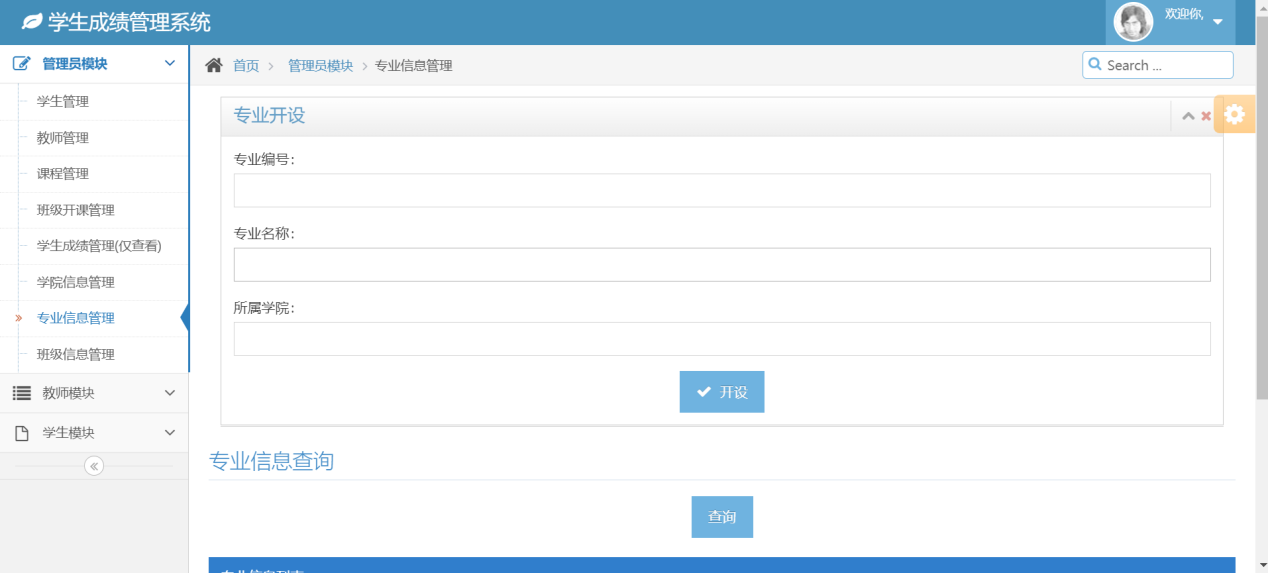
(学院添加)

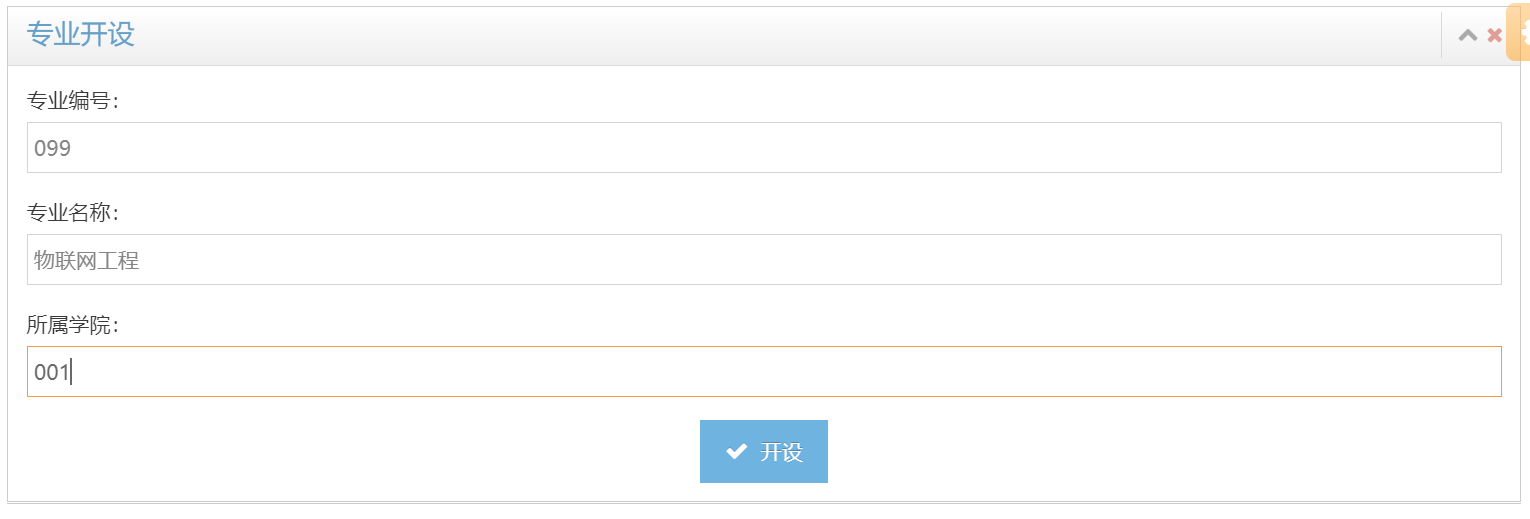
public boolean addCollege(String Collegeno,String Collegename) {  
 String sql = "INSERT INTO zhangcq\_colleges15 VALUES(?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Collegeno);  
 psmt.setString(2,Collegename);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

(学院查询)

public ArrayList<College> queryCollege(){  
 String sql="select \* from zhangcq\_colleges15";  
 ArrayList<College> list=new ArrayList<College>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 College college = new College();  
 college.setCollegeno(rst.getString(1));  
 college.setCollegename(rst.getString(2));  
 list.add(college);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面七：专业信息管理**





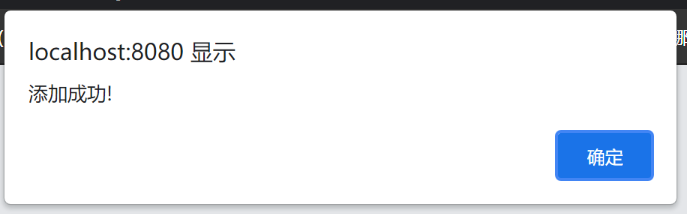


图6.3.2.11 专业管理展示图：专业信息添加

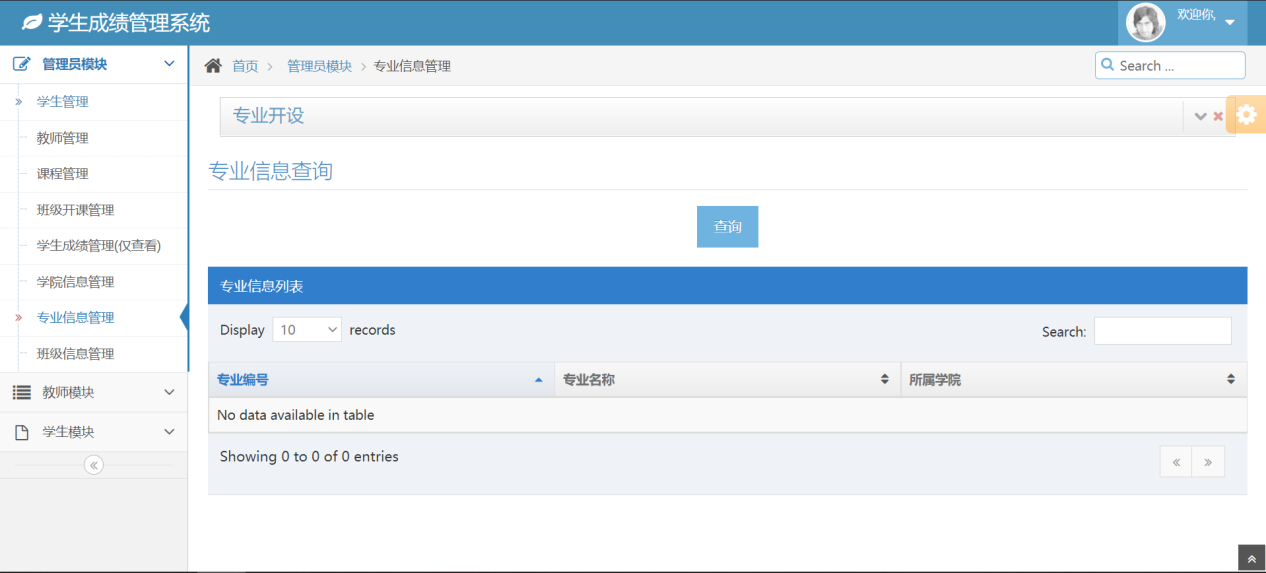




图6.3.2.12 专业管理展示图：专业信息查询

功能设计：这个界面一共有两个功能，一是专业添加，二是专业查询。专业添加主要是通过表单交给后台，后台通过INSERT语句插入信息到相应的表中。专业查询则是通过直接SELECT表zhangcq\_majors15得到查询结果。

核心代码：

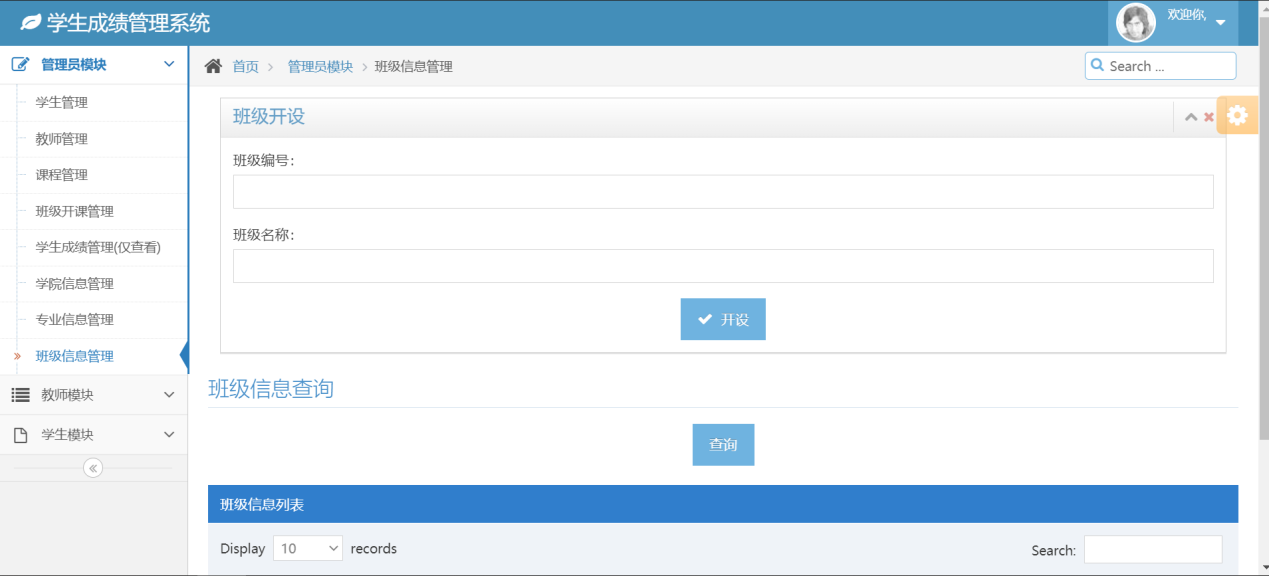
(专业添加)

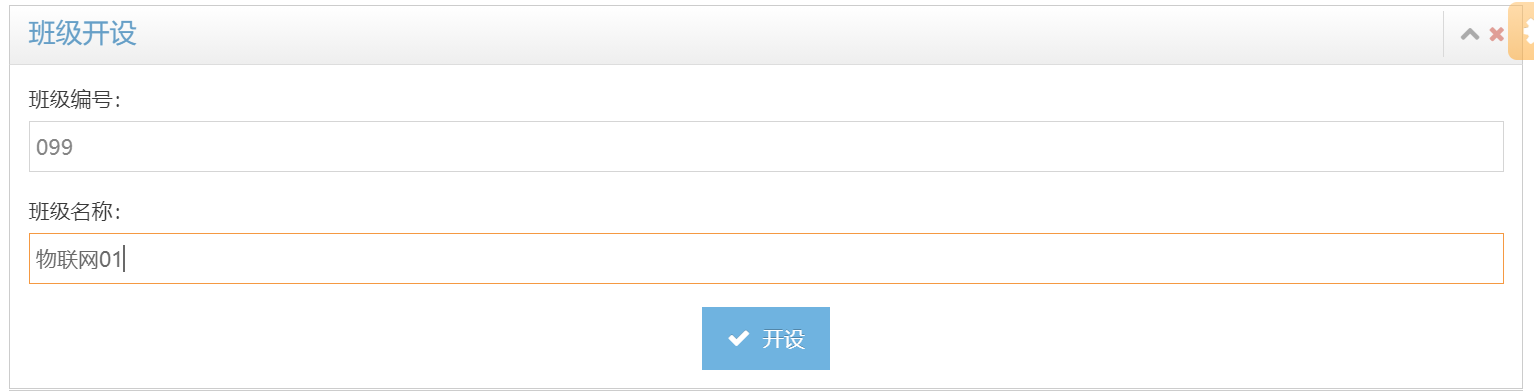
public boolean addMajor(String Mno,String Mname,String Collegeno) {  
 String sql = "INSERT INTO zhangcq\_majors15 VALUES(?,?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Mno);  
 psmt.setString(2,Mname);  
 psmt.setString(3,Collegeno);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

(专业查询)

public ArrayList<Major> queryMajor(){  
 String sql="select \* from zhangcq\_majors15";  
 ArrayList<Major> list=new ArrayList<Major>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Major major = new Major();  
 major.setMno(rst.getString(1));  
 major.setMname(rst.getString(2));  
 major.setCollegeno(rst.getString(3));  
 list.add(major);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面八：班级信息管理**





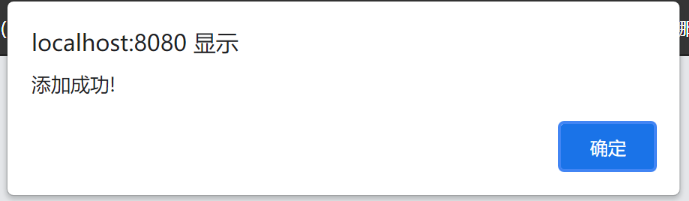
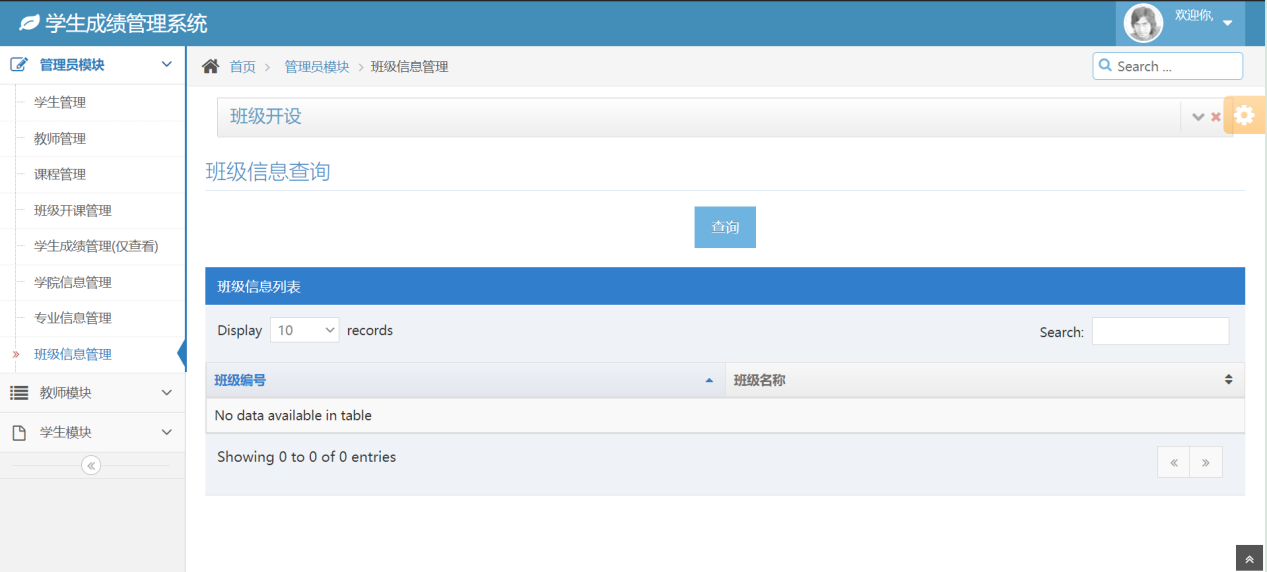


图6.3.2.13 班级管理展示图：班级信息添加



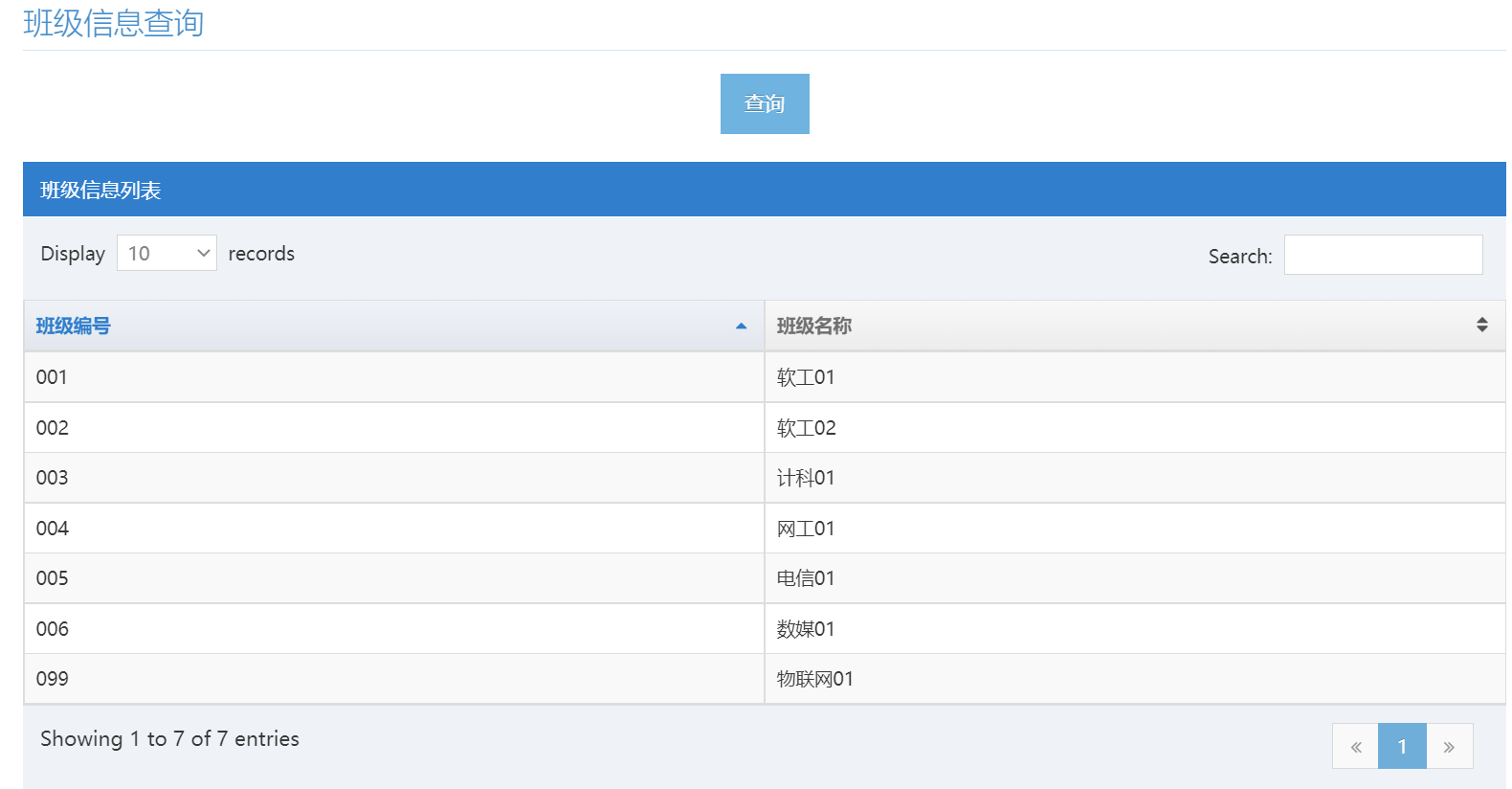


图6.3.2.14 班级管理展示图：班级信息查询

功能设计：这个界面一共有两个功能，一是班级添加，二是班级查询。班级添加主要是通过表单交给后台，后台通过INSERT语句插入信息到相应的表中。班级查询则是通过直接SELECT表zhangcq\_class15得到查询结果。

核心代码：

(班级添加)

public boolean addClass(String Classno,String Classname) {  
 String sql = "INSERT INTO zhangcq\_class15 VALUES(?,?)";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,Classno);  
 psmt.setString(2,Classname);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

(班级查询)

public ArrayList<Banji> queryClass(){  
 String sql="select \* from zhangcq\_class15";  
 ArrayList<Banji> list=new ArrayList<Banji>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Banji banji = new Banji();  
 banji.setClassno(rst.getString(1));  
 banji.setClassname(rst.getString(2));  
 list.add(banji);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**6.3.3 教师**

**界面一：成绩录入**



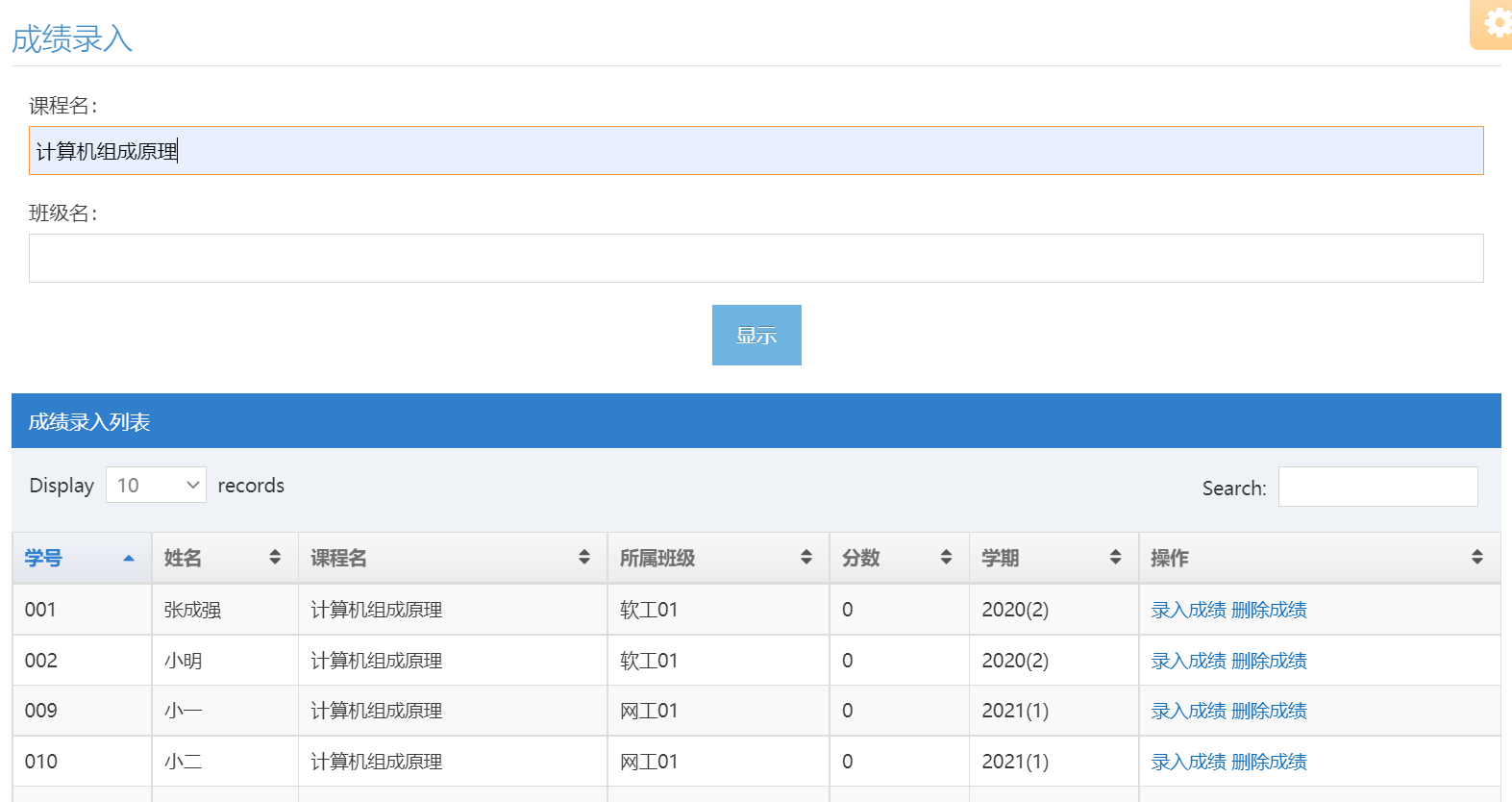
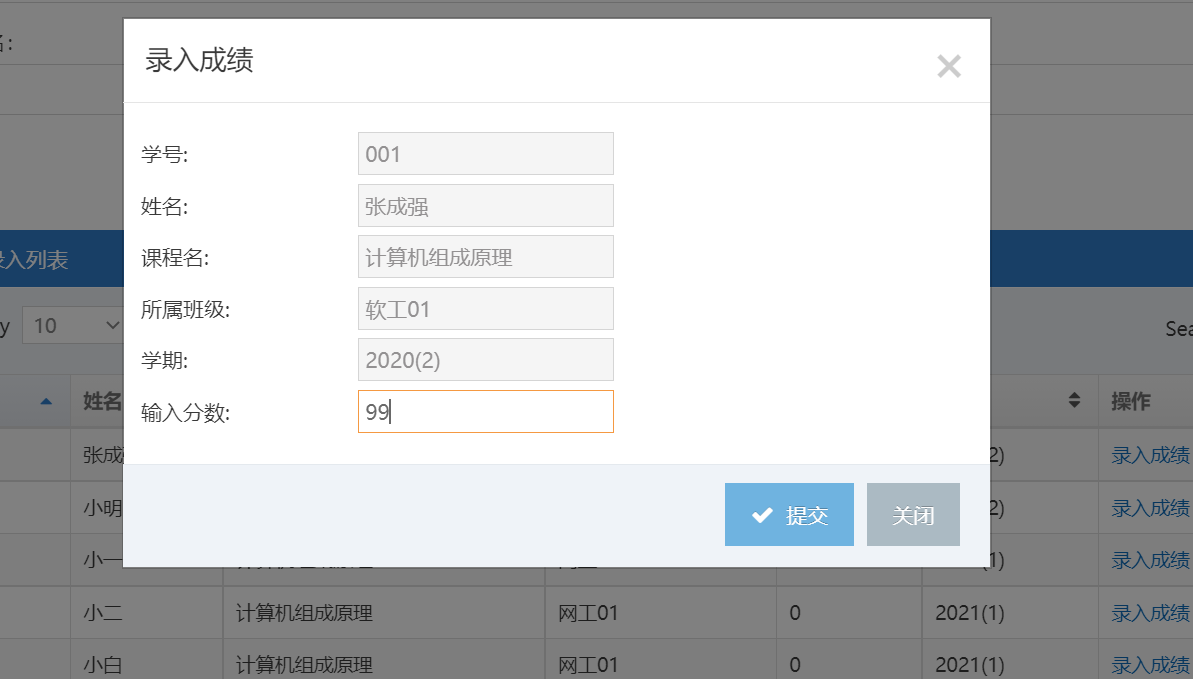


图6.3.3.1 录入成绩列表





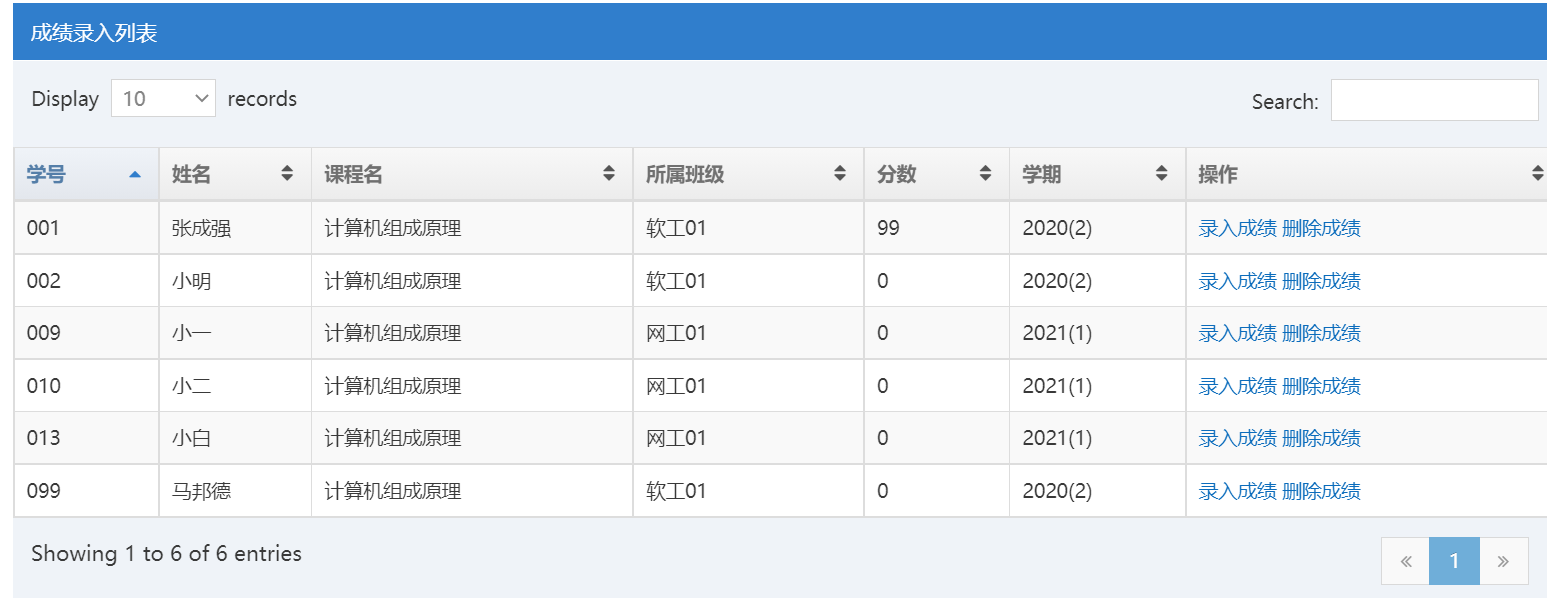


图6.3.3.2 录入成绩模态框

功能设计：这个页面主要功能就是根据查询的课程名和班级查，列出所有符合条件的学生，前提是任课教师是登录账号的教师。然后教师可以通过输入成绩模态框，对他们的成绩进行输入和修改。

核心代码：

(查询成绩)

public ArrayList<Grade> insertGrade1(String tno,String coursename,String banji) {  
 String sql="select distinct \* from zhangcq\_grainfo15 WHERE 1=1";  
 if(!coursename.equals("")){  
 sql=sql+" and zcq\_Coursename15='"+coursename+"'";  
 }  
 if(!banji.equals("")){  
 sql=sql+" and zcq\_Classname15='"+banji+"'";  
 }  
 if(!tno.equals("")){  
 sql=sql+" and zcq\_Tno15='"+tno+"'";  
 }  
 ArrayList<Grade> list=new ArrayList<Grade>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Grade grade=new Grade();  
 grade.setSno(rst.getString(1));  
 grade.setSname(rst.getString(2));  
 grade.setCoursename(rst.getString(3));  
 grade.setGrade(rst.getInt(4));  
 grade.setTno(rst.getString(5));  
 grade.setTname(rst.getString(6));  
 grade.setCterm(rst.getString(7));  
 grade.setClassname(rst.getString(8));  
 grade.setCno(rst.getString(9));  
 list.add(grade);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

(录入成绩)

public boolean modify\_grade(String sno,String tno,String cno,String cterm,String grade) {  
 String sql = "UPDATE zhangcq\_grade15 SET zcq\_Grade15 = ? WHERE zcq\_Sno15 = ? and zcq\_Tno15 = ? and zcq\_Courseno15 = ? and zcq\_Cterm15 = ?";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setInt(1,Integer.*parseInt*(grade));  
 psmt.setString(2,sno);  
 psmt.setString(3,tno);  
 psmt.setString(4,cno);  
 psmt.setString(5,cterm);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

**界面二：成绩查询**

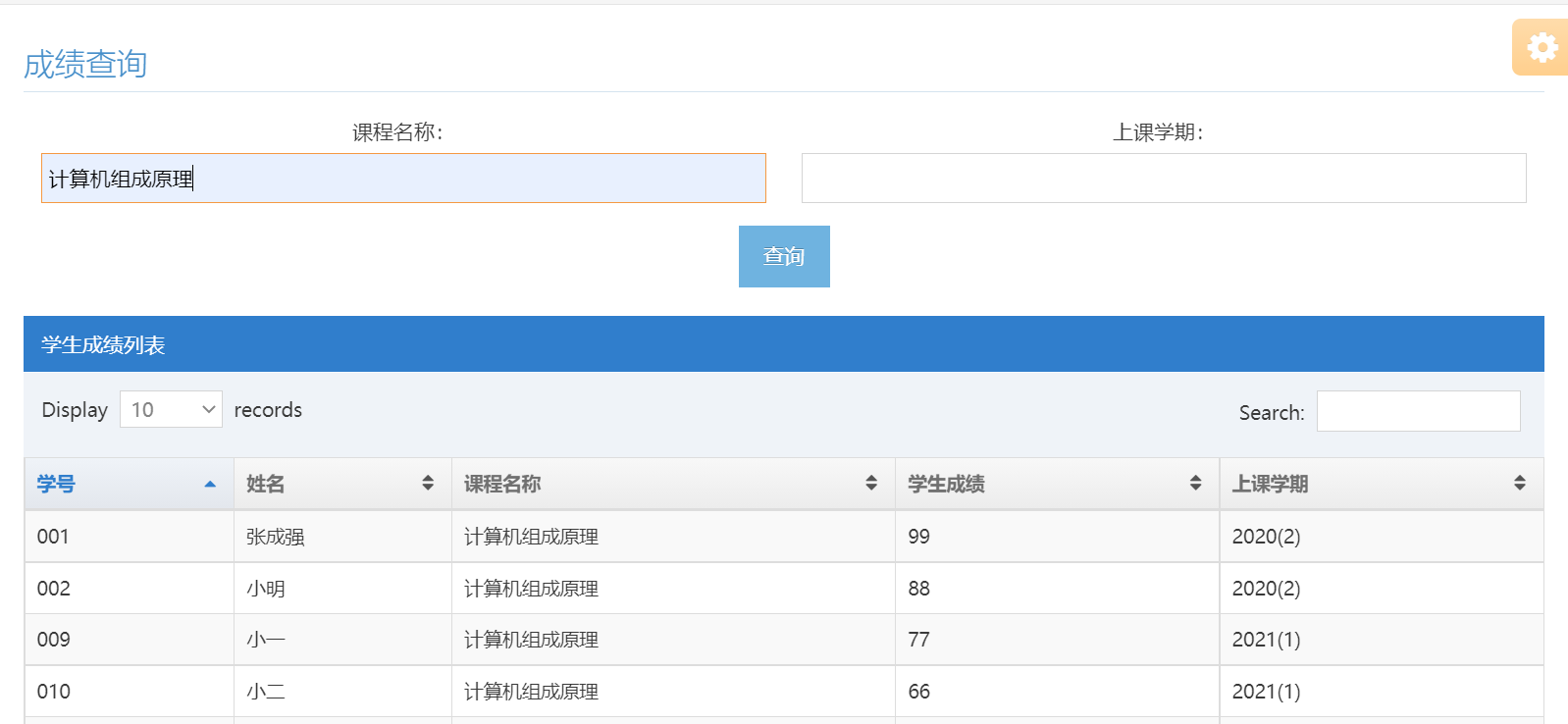


图6.3.3.3 教师查询对应任教学生成绩



图6.3.3.4 教师查询对应任教学生成绩饼状图

功能设计：这个页面主要功能就是根据查询的课程名和上课学期查，列出所有符合条件的学生，前提是任课教师是登录账号的教师。然后教师就可以知道他们的成绩的具体分布。

核心代码：

public ArrayList<Grade> t\_queryGrade(String coursename,String cterm,String tno) {  
 String sql="select distinct \* from zhangcq\_grainfo15 WHERE 1=1";  
 if(!coursename.equals("")){  
 sql=sql+" and zcq\_Coursename15='"+coursename+"'";  
 }  
 if(!cterm.equals("")){  
 sql=sql+" and zcq\_Cterm15='"+cterm+"'";  
 }  
 if(!tno.equals("")){  
 sql=sql+" and zcq\_Tno15='"+tno+"'";  
 }  
 ArrayList<Grade> list=new ArrayList<Grade>();  
 Grade.*setNine*(0);  
 Grade.*setEight*(0);  
 Grade.*setSeven*(0);  
 Grade.*setSix*(0);  
 Grade.*setDown*(0);  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Grade grade=new Grade();  
 grade.setSno(rst.getString(1));  
 grade.setSname(rst.getString(2));  
 grade.setCoursename(rst.getString(3));  
 grade.setGrade(rst.getInt(4));  
 grade.setTno(rst.getString(5));  
 grade.setTname(rst.getString(6));  
 grade.setCterm(rst.getString(7));  
 if(grade.getGrade()>90){  
 Grade.*setNine*(Grade.*getNine*()+1);  
 }  
 else if(grade.getGrade()>80){  
 Grade.*setEight*(Grade.*getEight*()+1);  
 }  
 else if(grade.getGrade()>70){  
 Grade.*setSeven*(Grade.*getSeven*()+1);  
 }  
 else if(grade.getGrade()>60){  
 Grade.*setSix*(Grade.*getSix*()+1);  
 }  
 else if(grade.getGrade()<60 && grade.getGrade()>0){  
 Grade.*setDown*(Grade.*getDown*()+1);  
 }  
 list.add(grade);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面三：任课查询**



图6.3.3.5 教师查询任教课程

功能设计：这个页面主要功能就是根据查询的上课学期查，列出所有符合条件的课程，前提是任课教师是登录账号的教师。然后教师就可以知道这个学期的具体任教课程信息。

核心代码：

public ArrayList<Grade> queryWork(String tno,String cterm){  
 String sql="select distinct zcq\_Cterm15, zcq\_Coursename15, zcq\_Chours15, zcq\_Cassessment15 from zhangcq\_workinfo15 WHERE 1=1";  
 if(!tno.equals("")){  
 sql=sql+" and zcq\_Tno15='"+tno+"'";  
 }  
 if(!cterm.equals("")){  
 sql=sql+" and zcq\_Cterm15='"+cterm+"'";  
 }  
 ArrayList<Grade> list=new ArrayList<Grade>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Grade grade = new Grade();  
 grade.setCterm(rst.getString(1));  
 grade.setCoursename(rst.getString(2));  
 grade.setChour(rst.getInt(3));  
 grade.setAssessment(rst.getString(4));  
 list.add(grade);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面四：修改密码**

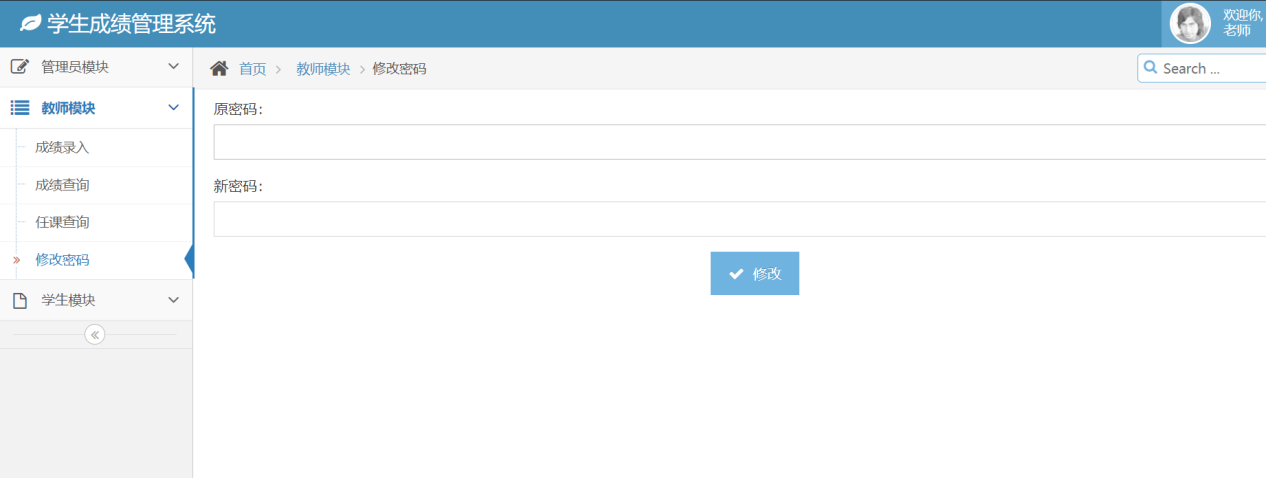


图6.3.3.6 教师修改密码界面

功能设计：这个页面主要功能就是修改教师密码。根据教师输入的旧密码是否一致保证安全，然后进行新密码修改。

核心代码：

public boolean modify\_pwd(String new\_pwd,String tno) {  
 String sql = "UPDATE zhangcq\_teachers15 SET zcq\_Tpwd15 = ? WHERE zcq\_Tno15 = ?";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,new\_pwd);  
 psmt.setString(2,tno);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

**6.3.4 学生**

**界面一：成绩查询**





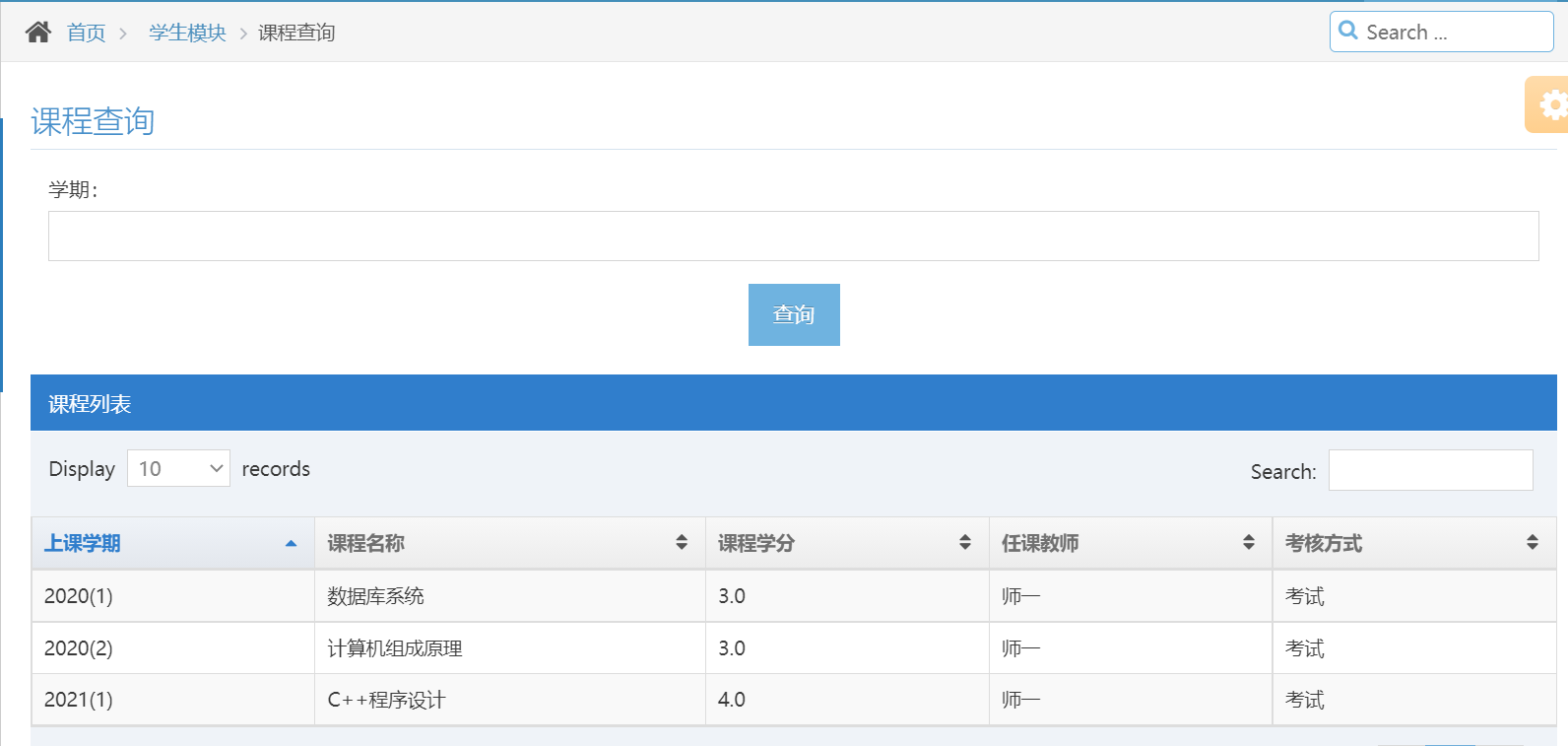
图6.3.4.1 学生成绩查询图

功能设计：这个页面主要功能就是根据查询的上课学期查，列出所有符合条件的成绩，前提是学生信息是登录账号的学生。然后学生就可以知道这个学期的具体的课程成绩信息。

核心代码:

public ArrayList<Grade> queryGrade(String sno,String cterm,String tno) {  
 String sql="select distinct \* from zhangcq\_grainfo15 WHERE 1=1";  
 if(!sno.equals("")){  
 sql=sql+" and zcq\_Sno15='"+sno+"'";  
 }  
 if(!cterm.equals("")){  
 sql=sql+" and zcq\_Cterm15='"+cterm+"'";  
 }  
 if(!tno.equals("")){  
 sql=sql+" and zcq\_Tno15='"+tno+"'";  
 }  
 ArrayList<Grade> list=new ArrayList<Grade>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Grade grade=new Grade();  
 grade.setSno(rst.getString(1));  
 grade.setSname(rst.getString(2));  
 grade.setCoursename(rst.getString(3));  
 grade.setGrade(rst.getInt(4));  
 grade.setTno(rst.getString(5));  
 grade.setTname(rst.getString(6));  
 grade.setCterm(rst.getString(7));  
 list.add(grade);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面二：课程查询**



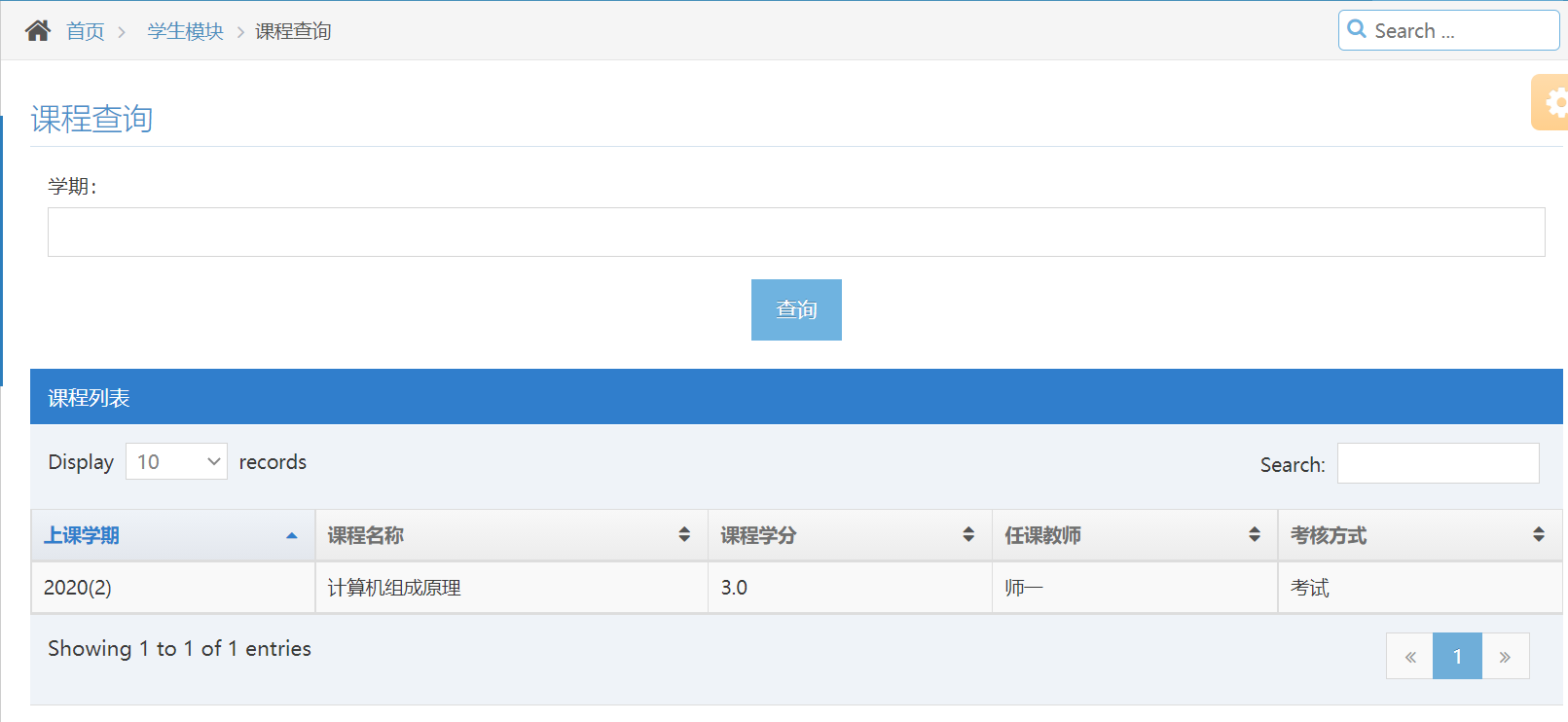


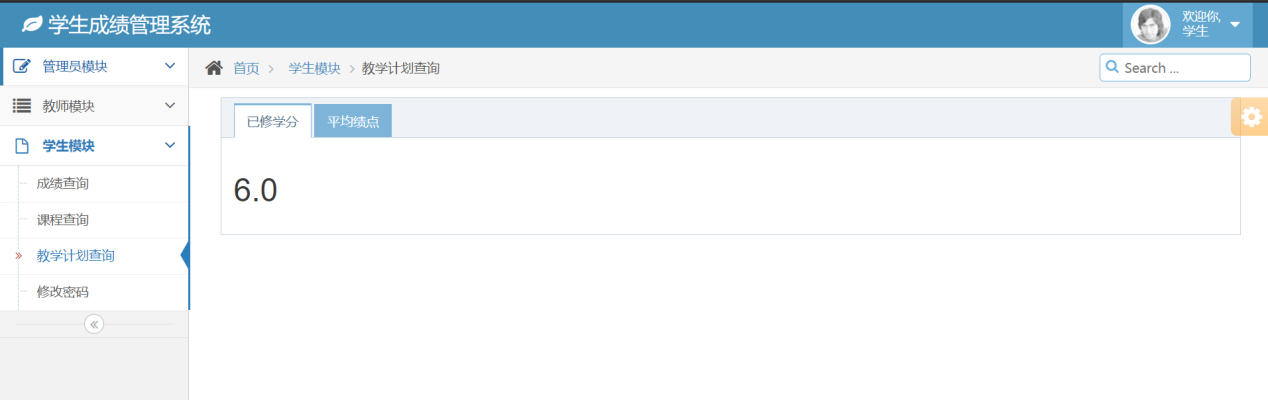
图6.3.4.2 学生课程查询界面

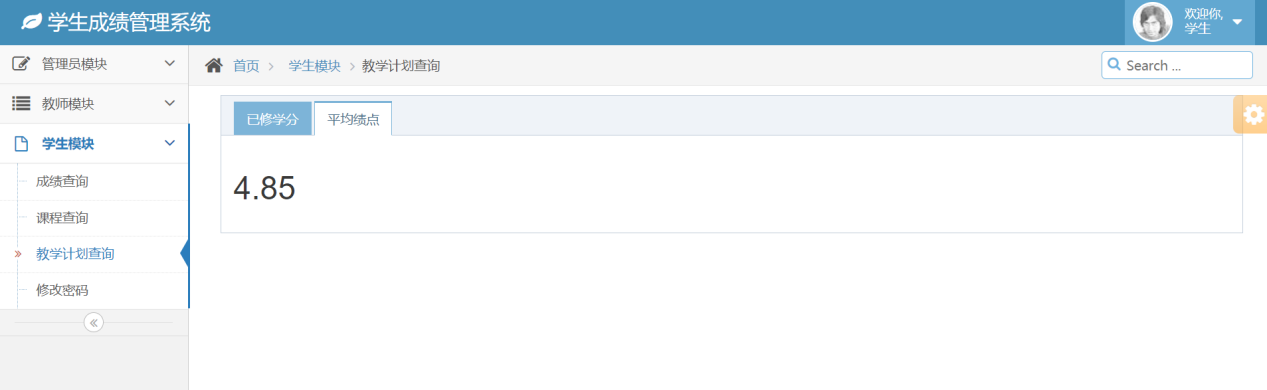
功能设计：这个页面主要功能就是根据查询的上课学期查，列出所有符合条件的课程，前提是学生信息是登录账号的学生。然后学生就可以知道这个学期的具体的课程信息。

核心代码：

public ArrayList<Grade> queryWork(String sno,String cterm){  
 String sql="select \* from zhangcq\_workinfo15 WHERE 1=1";  
 if(!sno.equals("")){  
 sql=sql+" and zcq\_Sno15='"+sno+"'";  
 }  
 if(!cterm.equals("")){  
 sql=sql+" and zcq\_Cterm15='"+cterm+"'";  
 }  
 ArrayList<Grade> list=new ArrayList<Grade>();  
 try(Connection conn=dataSource.getConnection();PreparedStatement psmt=conn.prepareStatement(sql)){  
 try(ResultSet rst = psmt.executeQuery()){  
 while(rst.next()){  
 Grade grade = new Grade();  
 grade.setCterm(rst.getString(4));  
 grade.setCoursename(rst.getString(5));  
 grade.setCredit(rst.getDouble(6));  
 grade.setTname(rst.getString(3));  
 grade.setAssessment(rst.getString(8));  
 list.add(grade);  
 }  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }  
 return list;  
}

**界面三：教学计划查询**



图6.3.4.3 学生教学计划查询界面

功能设计：就是取出登录账号的已修学分和平均绩点。这些数据是教师在输入成绩的时候，根据先前设置的触发器马上进行实时更新的。

核心代码：

request.setCharacterEncoding("utf-8"); //1  
response.setContentType("text/html;charset=utf-8"); //2  
response.setCharacterEncoding("utf-8"); //3  
String kind =request.getParameter("kind");  
String username = request.getParameter("username");  
String password = request.getParameter("password");  
HttpSession session=request.getSession();  
String message=null;  
RequestDispatcher dispatcher=null;  
loginDao userdao=new loginDao();  
if(kind.equals("student")){  
  
 Student user=userdao.StudentLogin(username,password);  
 if(user==null){  
 message="用户名或密码错误";  
 dispatcher=request.getRequestDispatcher("./VIEW/index.jsp");  
 }  
 else{  
 message="登录成功！欢迎!";  
 session.setAttribute("user",user);  
 dispatcher=request.getRequestDispatcher("./VIEW/student.jsp");  
 }  
 session.setAttribute("message",message);  
 session.setAttribute("kind","学生");  
 dispatcher.forward(request,response);  
}

**界面四：修改密码**

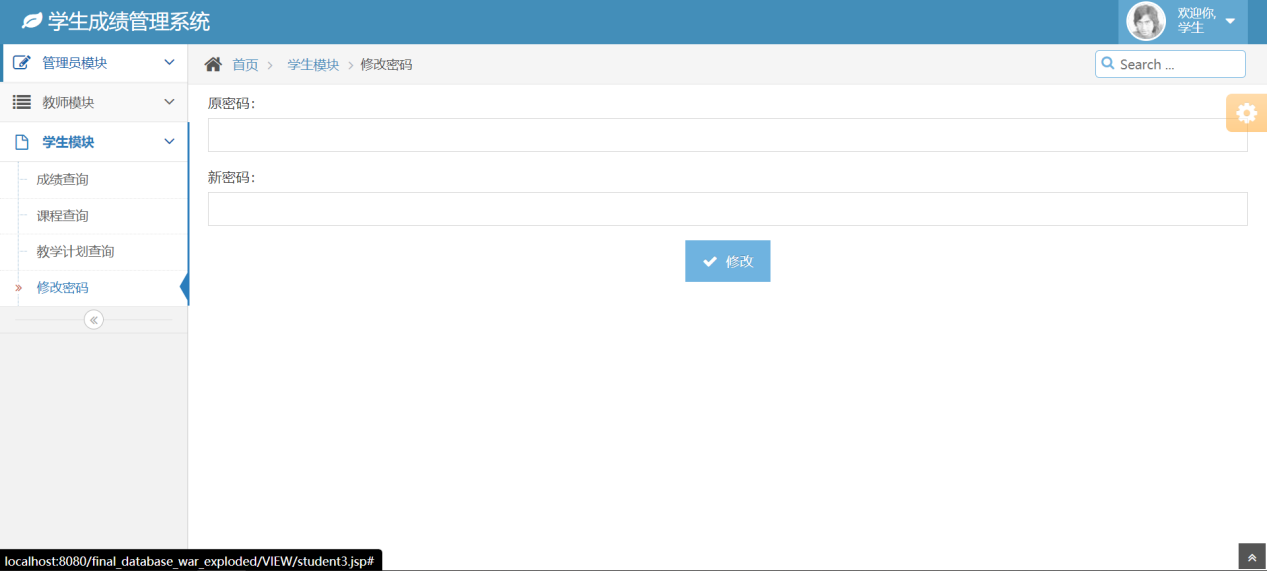


图6.3.4.4 学生修改密码界面

功能设计：这个页面主要功能就是修改教师密码。根据教师输入的旧密码是否一致保证安全，然后进行新密码修改。

核心代码：

public boolean modify\_pwd(String new\_pwd,String tno) {  
 String sql = "UPDATE zhangcq\_teachers15 SET zcq\_Tpwd15 = ? WHERE zcq\_Tno15 = ?";  
 try (Connection conn = dataSource.getConnection(); PreparedStatement psmt = conn.prepareStatement(sql)) {  
 psmt.setString(1,new\_pwd);  
 psmt.setString(2,tno);  
 psmt.execute();  
 return true;  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 return false;  
 }  
}

**6.3.5 访问无权限页面时**



图6.3.5.1 访问无权页面展示图

## 7、实验总结

### 7.1 遇到的问题和解决的办法

1.在表中，我们使用编号来代表专业、学院、班级、课程，但是这些在查询时直接展现给用户就会显得很不友好。但如果要用名字代表，用原来表字段创建的模型就会失效。

所以我们在原来的基础上扩充模型，虽然会有一定的冗余，但大大提升了用户的可操作性，和减少了代码的编写量。

1. 当前端用户查询时输入了空字符串，此时其实是用户不希望用这个查询条件的，但按我们平时的代码的话，就会查询到空结果。于是我们希望将空字符串查询到的结果为全部。

实现过程如下，

string sql="select \* from 表 where 1=1"

查询表里所有内容

它增加1=1的目的来了,是为下面的拼凑做准备.

if(username!="")

{sql=sql+" and username='admin'";}

//查询用户名为admin的用户

if(password!="")

{sql=sql+" and password='123'"}

//查询用户名为admin,密码为123的用户

1. 当用课程和成绩表联合查询时会出现许多重复项，这个时候会影响用户的搜索效率，于是我再查询语句的主字段前加了distinct 有效的删除了视图中的重复项。
2. 在创建存储过程的时候一直报1064的错误，后来才知道，mysql解释器一碰到；就自动停止了。但是存储过程中避免不了有很多的“；”。所以我们要在开头用delimiter $$把停止符号换成$$这样MySQL遇到$$就不会停止了。
3. 在调运插入存储过程的时候，他总是会在最后一个地方报1062主键重复的错误。后来大量查阅资料发现必须要将FETCH 放在判断之前才不会重复取值，而我之前用的REPEAT函数中的UTIL判断是永远在最开始执行的。导致最后一个元组取值重复。之后我就把REPEATE函数用LOOP代替，大功告成！

### 7.2 系统设计的不足

1.不同类型的用户登录这个系统会有不同的操作界面。但是为了防止一些用户恶意访问其他不属于自己权限的界面，本实验采取了修改链接的方式。但是一些内部用户仍可通过get方式直接访问越权界面。

2.在表的设计中，为了兼顾课程需要，本实验不能很好的表示选修课的班级状态，本实验的班级为实际的行政班，而并非现实生活中的走读班。实际上我们每个人都属于很多不同的班级。这点本实验没有考虑周全。

### 7.3 进一步改进思路和体会

1.如果有时间的话，我想用过滤器进行用户过滤用户访问请求会更加的合理和安全。

2.之后可以将表进行进一步的修改拆分，把班级单独进行表示。让每个学生可以属于不同的班级，每个班级可以拥有很多不同的学生。