|  |  |
| --- | --- |
| 总评成绩 |  |

数据库课程设计

说明书

学生成绩管理系统

学生姓名： 何锋 学号： 1707004130

学 院： 大数据学院

专 业： 计算机科学与技术

指导教师： 杨晓雯

2020年1月

目 录

1. 引言 …………………………………………………………………………… 1
   1. 编写目的 ……………………………………………………………… 1
   2. 项目背景及意义 ……………………………………………………… 1
2. 需求分析 ……………………………………………………………………… 2
   1. 功能需求分析 ………………………………………………………… 2
   2. 设计分析 ……………………………………………………………… 2
3. 应用环境 ……………………………………………………………………… 4
   1. 系统运行网络环境 …………………………………………………… 4
   2. 系统软硬件环境 ……………………………………………………… 4
4. 概念结构设计 ………………………………………………………………… 5
   1. 学生实体结构设计 …………………………………………………… 5
   2. 教师实体结构设计 ……………………………………………………… 5
   3. 管理员实体结构设计 …………………………………………………… 5
   4. 班级实体结构设计 ……………………………………………………… 6
   5. 课程实体结构设计 …………………………………………………… 6
   6. 系统整体结构设计 ……………………………………………………… 7
5. 逻辑结构设计 ……………………………………………………………… 8
   1. 学生逻辑结构设计 …………………………………………………… 8
   2. 教师逻辑结构设计 ……………………………………………………… 9
   3. 课程逻辑结构设计 …………………………………………………… 9
   4. 成绩逻辑结构设计 …………………………………………………… 10
   5. 管理员逻辑结构设计 ………………………………………………… 10
   6. 班级逻辑结构设计 …………………………………………………… 11
   7. 班级课程逻辑结构设计 ……………………………………………… 11
   8. 班级教师逻辑结构设计 ……………………………………………… 11
   9. 考试逻辑结构设计 …………………………………………………… 12
   10. 授课逻辑结构设计 …………………………………………………… 12
6. 物理结构设计 ……………………………………………………………… 13
   1. 存取方法的设计 ……………………………………………………… 13
7. 概要设计 …………………………………………………………………… 14
   1. 系统总体设计概述 …………………………………………………… 14
   2. 系统功能模块划分 …………………………………………………… 14
   3. 系统功能模块清单 …………………………………………………… 14
8. 详细设计 …………………………………………………………………… 17
   1. 登录模块设计 ………………………………………………………… 17
      1. SQL语句设计 …………………………………………………… 17
      2. LoginGUI类设计 ……………………………………………… 17
      3. 界面效果 ……………………………………………………… 18
   2. 添加课程模块设计 …………………………………………………… 19
      1. SQL语句设计 …………………………………………………… 19
      2. CourseAddJPaneL类设计 ……………………………………… 19
      3. 界面效果 ……………………………………………………… 20
   3. 查询课程模块设计 …………………………………………………… 20
      1. SQL语句设计 …………………………………………………… 20
      2. CourseSearchJPanel类设计 ………………………………… 20
      3. 界面效果 ……………………………………………………… 22
   4. 安排课程功能模块设计 ……………………………………………… 22
      1. SQL语句设计 …………………………………………………… 22
      2. CourseArrangeJPanel类设计 ………………………………… 22
      3. 界面效果 ……………………………………………………… 23
   5. 添加师生功能模块设计 ……………………………………………… 23
      1. SQL语句设计 …………………………………………………… 23
      2. StuAddJPanel类和TeaAddJPanel类设计 …………………… 24
      3. 界面效果 ……………………………………………………… 25
   6. 师生管理功能模块设计 ……………………………………………… 26
      1. SQL语句设计 …………………………………………………… 26
      2. StuManageJPanel类和TeaManageJPanel类设计 …………… 27
      3. 界面效果 ……………………………………………………… 29
9. 测试 ………………………………………………………………………… 30
   1. 功能测试 ……………………………………………………………… 30
      1. 登录功能测试 ……………………………………………… 30
      2. 添加课程功能测试 …………………………………………… 35
      3. 查询课程功能测试 ………………………………………… 37
      4. 修改和删除课程功能测试 …………………………………… 39
      5. 安排课程功能测试 …………………………………………… 41
      6. 添加师生功能测试 …………………………………………… 42
      7. 修改和删除师生功能测试 …………………………………… 44
   2. 用户界面测试 ………………………………………………………… 47
   3. 测试结论 ……………………………………………………………… 47
10. 结论 ………………………………………………………………………… 48

参考文献 ………………………………………………………………………… 49

**1 引言**

**1.1 编写目的**

学生成绩管理一直以来都是学校教学管理中十分重要又相当复杂的管理工作之一，随着学校的规模不断扩大、学生数量的急剧增长，有关学生的各种信息也在增加，传统的手工处理已经无法满足当下教学和管理的要求。而计算机具有运算速度快、处理能力强等特点，非常适合应用到该领域上。

因此，为了保证流畅、安全、高效的存储大量学生的成绩信息，有必要设计一个学生成绩管理系统。这不但可以使教务人员从复杂的成绩管理中解脱出来，而且对于推动教学的发展也起到非常重要的作用。

**1.2 项目背景及意义**

随着我国教育改革的不断深入以及自学教育的不断扩招，参加考试的人越来越多,考试科目的多样化,使得学校对自学考试成绩的管理越趋繁琐、复杂，工作业务繁杂,工作量大,这种传统的学生管理模式已经暴露出种种弊端:难以统一调配和处理，效率极低，缺乏科学性以及合理性。

随着计算机应用的普及与深入，利用计算机能够对所有自学考试成绩进行统一管理，并进行分析，大大减少教学秘书的工作量,提高工作效率，为教学办公带来了极大的方便。使操作人员只需输入一些简单的汉字、数字,便可以存储、查找、修改、打印学生成绩信息等。学生成绩管系统的开发可实现学生成绩管理的系统化、规范化、自动化、达到提高学生成绩管理效率的目的,本着实用性、通用、开放和安全的原则 ,实现学生信息管理、课程信息管理、学生成绩管理、成绩查询等功能。本着一切从实际出发,充分考虑成绩的内部管理、信息交流等方面的复杂需求,实现成绩的有效管理,真正为学生成绩管理提供一个电子平台。

因此为了做好学生成绩管理工作,提高办事效率,实现全面的、相对集中的办公自动化,开发学生成绩管理系统就成了当务之急,其目的主要为了彻底改变这种繁杂的管理模式,把广大教师从繁重的成绩管理工作中解脱出来、把学校从传统的成绩管理模式中解放出来,实现全面的、相对集中的、智能化的信息综合管理,为学校的教学管理工作带来方便。

**2 需求分析**

**2.1 功能需求分析**

每当考试结束后，管理员开放考试成绩录入功能，教师录入考试成绩，学生可以查询自己的考试成绩。

学生成绩管理系统的用户有管理员，可以进行个人信息查询和修改，修改个人密码、开放考试成绩录入功能，录入、安排课程，安排、查询考试，管理学生成绩、学生、教师等操作；其次还有任课老师，可以查询、修改自己的个人信息，修改个人密码，查询考试信息，录入、查询、修改学生考试成绩；最后，学生也是不可或缺的用户，学生可以登录查询、修改个人信息，修改个人密码，查询各科成绩，查询考试，查询课程等。

**2.2 设计分析**

因为我们做的是学生成绩管理系统，所以数据库中有一个表存储学生各科成绩，有了学生成绩，就要有一个表存储学生信息，一个表存储课程信息，教师能录入学生成绩，所以要有一个教师表，管理员可以管理录入功能的开放与关闭，也可以管理学生教师以及课程的信息，所以要有一个管理员表，每个学生都有一个班级，所以要有一个表来存储班级信息，每个班级都有不同的课程，所以要建立一个班级课程表，每个班级也都有教师，所以要建立一个班级的教师表，教师授课的情况也要知道，所以要建立一个授课表。

综上所述，学生成绩管理系统有11个表：班级表class，班级课程表classcourse，班级教师表classteacher，课程表course，成绩表csgrade，考试表exam，成绩录入功能表gradesystem，管理员表manager，学生表student，教师表teacher，授课表teaching。

1. 班级表class：应包括班级号、班级人数、系别、专业。
2. 班级课程表classcourse：除了班级号、课程号必要字段，还应包括学年以及学期等。
3. 班级教师表classteacher：除了班级教师号、教师号必要字段，还应包括学年以及学期等。
4. 课程表course：应包括课程号、课程名、学时、学分和属性。
5. 成绩表csgrade：应包括学号、课程号、成绩。
6. 考试表exam：除了考试号、考试课程号必要字段，还应包括时间、地点等。
7. 成绩录入功能表gradesystem：用于管理成绩录入功能的开启和关闭，0为关闭，1为开启。
8. 管理员表manager：除了管理员号、姓名、性别、密码必要字段，还应包括年龄、电话、电子邮箱等。
9. 学生表student：除了学号、姓名、性别、班级、密码必要字段，还应包括年龄、电话、电子邮箱等。
10. 教师表teacher：除了教师号、姓名、性别、院系、密码必要字段，还应包括年龄、职称、学历、电话、电子邮箱等。
11. 授课表teaching：除了教师号、课程号、地点、时间、周次，还应包括学年、学期等。

**3 应用环境**

**3.1 系统运行网络环境**

网络结构图如图3-1所示，用户（学生、教师和管理员）可以通过网络登录学生成绩管理系统，进行相应的操作，应用服务器接收到相应操作做出相应响应之后对数据库进行相应操作，并返回进行展示。

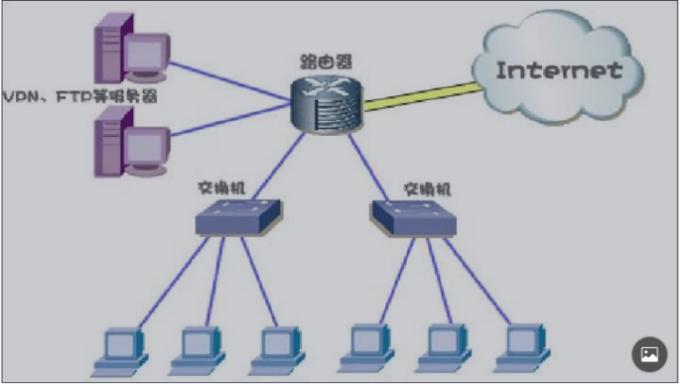


图3-1 系统网络结构图

**3.2 系统软硬件环境**

服务器配置要求如表3.1所示。

表3.1 服务器配置

|  |  |
| --- | --- |
| 配置项 | 配置参数 |
| CPU 型号 | Intel(R)Core(TM)i5-7200U |
| CPU 主频 | 2.50GHz |
| 内存容量 | 16.0GB |
| 硬盘容量 | 256G + 448G |
| 操作系统 | Windows 10 专业版 |

系统运行软件环境如下表3.2所示。

表3.2 系统运行软件环境

|  |  |
| --- | --- |
| 环境项 | 环境参数 |
| 操作系统 | 具有Java虚拟机的操作系统均可 |
| 数据库 | MySQL |
| 数据库管理工具 | Navicat Premium 12、Navicat 12 for MySQL |
| 开发工具 | IntelliJ IDEA 2019.2、Eclipse |
| 第三方开发工具包 | mysql-connector-java-8.0.18.jar、jfreechart-1.0.19.jar、jcommon-1.0.23.jar、datepicker.jar、jdatepicker-1.3.4.jar、poi-3.9.jar、poi-ooxml-3.9.jar、poi-ooxml-schemas-3.9.jar |
| 图标设计工具 | Adobe PhotoShop CC 2019 |

**4 概念结构设计**

本章在需求分析的基础之上对学生成绩管理系统的数据库进行设计，设计原则是实体合理、联系精简、范式标准等。学生成绩管理系统数据库有5个实体，分别为学生、教师、管理员、班级和课程。各实体的属性以及实体E-R图如下所述。

**4.1 学生实体结构设计**

学生基本信息包括：学号、姓名、性别、年龄、班级、联系电话、电子邮箱、密码等，其中学号为主码。

学生实体的E-R图如图4-1所示。

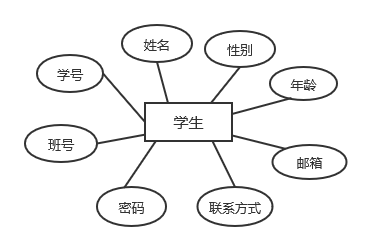


图4-1 学生实体E-R图

**4.2 教师实体结构设计**

教师基本信息包括：教师号、密码、姓名、性别、年龄、系别、学历、职称、职务、联系电话、电子邮箱等，其中教师号为主码。

教师实体的E-R图如图4-2所示。

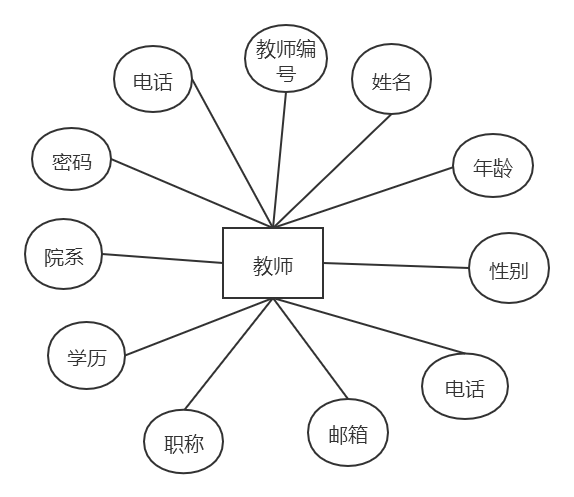


图4-2 教师实体E-R图

**4.3 管理员实体结构设计**

管理员基本信息包括：管理员号、姓名、性别、年龄、密码、联系电话、电子邮箱等，其中管理员号为主码。

管理员实体的E-R图如图4-3所示。

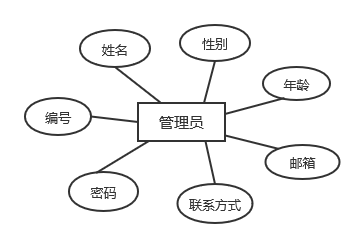


图4-3 管理员实体E-R图

**4.4 班级实体结构设计**

班级基本信息包括：班级号、班级人数、系别、专业等，其中班号为主码。

班级实体的E-R图如图4-4所示。

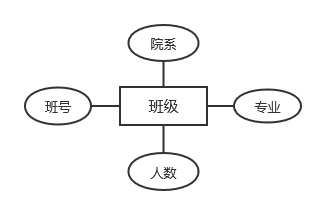


图4-4 班级实体E-R图

**4.5 课程实体结构设计**

课程基本信息包括：课程号、课程名、学时、学分、属性等，其中课程号为主码。

课程实体的E-R图如图4-5所示。

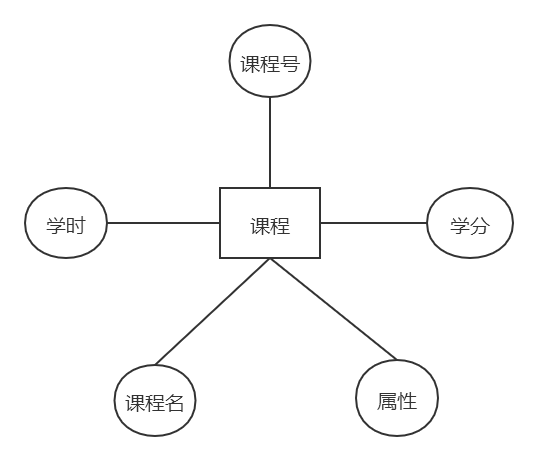


图4-5 课程实体E-R图

**4.6 系统整体结构设计**

系统的实体E-R图如图4-6和4-7所示。

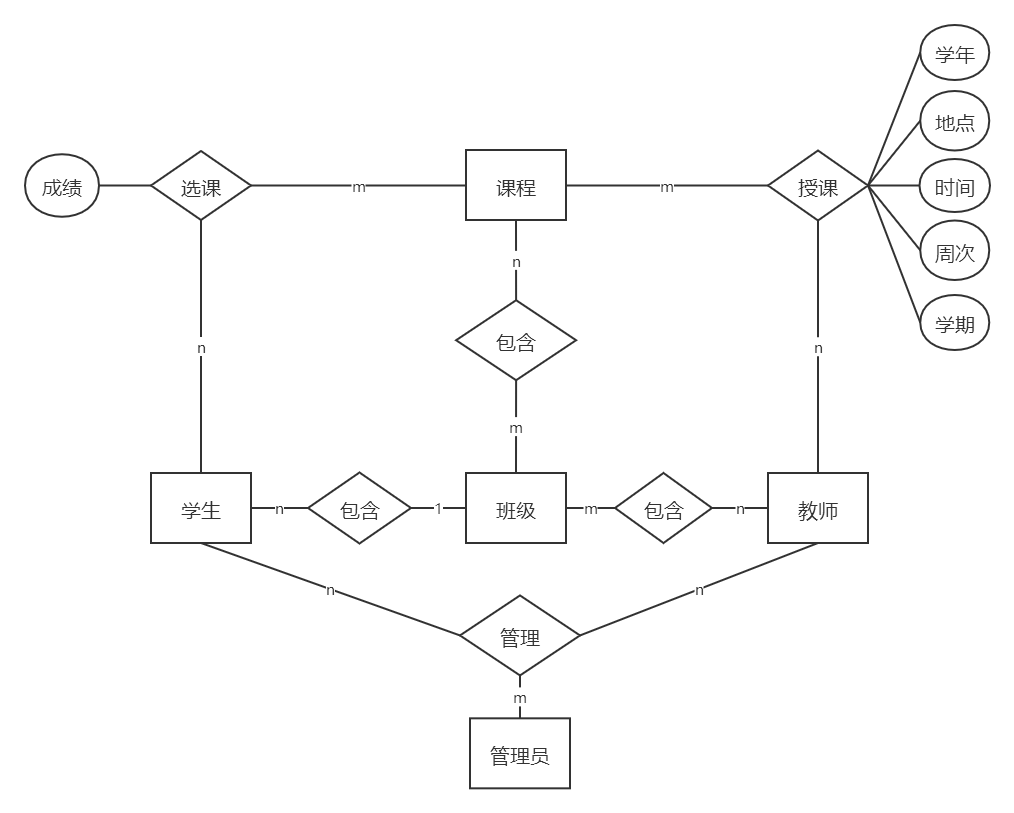


图4-6 系统实体E-R图

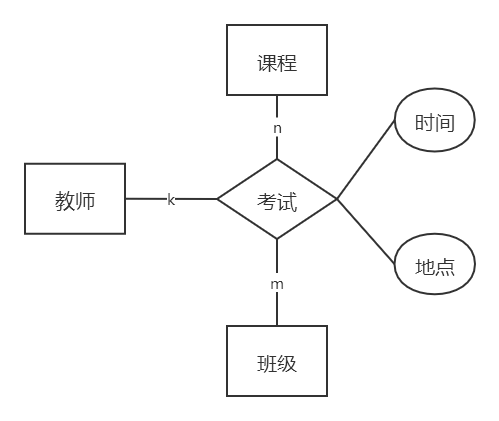


图4-7 系统实体E-R图

**5 逻辑结构设计**

根据E-R图向关系模型的转换原则，一个实体型转换为一个关系模式，实体的属性就是关系的属性。因此按照图3-6、3-7中所画的E-R图，本数据库中的表为：学生表（Student表）、教师表（Teacher表）、课程表（Course表）、成绩表（CSGrade表）、管理员表（Manager表）、班级表（Class表）、班级课程表（ClassCourse表）、班级教师表（ClassTeacher表）和教授课程表（Teaching表）。

根据实体型间联系的转换原则：

（1）一个1:1联系可以转-换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。如果转换为一个独立的关系模式，则与该联系相连的各实体的码以及联系本身的属性转换为联系的属性，每个实体的码均是该关系的候选码。如果与某一端实体对应的关系模式合并，则需要在该关系模式的属性中加入另一个关系模式的码和联系本身的属性。

（2）一个1:n联系可以转换为一个独立的关系模式，也可以与n端对应的关系模式合并。如果转换为一个独立的关系模式，则与该联系相连的各实体的码以及联系本身的属性转换为联系的属性，而关系的码为n端实体的码。

（3）一个m:n联系可以转换为一个关系模式，与该联系相连的各实体的码以及联系本身的属性转换为联系的属性，各实体的码组成关系的码或关系码的一部分。

综上所述，该数据库中的逻辑结构设计如下。

**5.1 学生逻辑结构设计**

学生的逻辑结构设计如表5.1所示。

表5.1 学生表（Student表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 学号 | Sno | varchar | 10 | No | 主码 |
| 姓名 | Sname | varchar | 10 | Yes |  |
| 性别 | Ssex | char | 2 | Yes | 男、女 |
| 年龄 | Sage | int | 2 | Yes |  |
| 班级 | Sclass | varchar | 10 | Yes |  |
| 密码 | Spassword | varchar | 10 | No |  |
| 联系电话 | Stel | varchar | 11 | Yes |  |
| 电子邮箱 | Semail | varchar | 20 | Yes |  |

**5.2 教师逻辑结构设计**

教师的逻辑结构设计如表5.2所示。

表5.2 教师表（Teacher表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 教师号 | Tno | varchar | 10 | No | 主码 |
| 姓名 | Tname | varchar | 10 | Yes |  |
| 性别 | Tsex | char | 2 | Yes | 男、女 |
| 年龄 | Tage | int | 2 | Yes |  |
| 系别 | Tdept | varchar | 20 | No |  |
| 学历 | Tdegree | varchar | 8 | Yes | 本科、硕士、博士、博士后 |
| 职称 | Ttitle | varchar | 4 | Yes | 助教、讲师、副教授、教授 |
| 职务 | Tduty | varchar | 20 | Yes |  |
| 联系电话 | Ttel | varvhar | 11 | No |  |
| 电子邮箱 | Temail | varchar | 20 | Yes |  |
| 密码 | Tpassword | varchar | 10 | No |  |

**5.3 课程逻辑结构设计**

课程的逻辑结构设计如表5.3所示。

表5.3 课程表（Course表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 课程号 | Cno | varchar | 10 | No | 主码 |
| 课程名 | Cname | varchar | 50 | No |  |
| 学时 | Cperiod | int | 2 | Yes |  |
| 学分 | Ccredit | double | 4 | Yes |  |
| 属性 | Cattribute | char | 4 | No | |  | | --- | | 必修、专业选修、公共选修 | |

**5.4 成绩逻辑结构设计**

成绩的逻辑结构设计如表5.4所示。

表5.4 成绩表（CSGrade表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 学号 | Sno | varchar | 10 | No | 主码 |
| 课程号 | Cno | varchar | 10 | No | 主码 |
| 成绩 | Grade | double | 4 | Yes |  |

**5.5 管理员逻辑结构设计**

管理员的逻辑结构设计如表5.5所示。

表5.5 管理员表（Manager表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 管理员号 | Mno | varchar | 10 | No | 主码 |
| 姓名 | Mname | varchar | 10 | No |  |
| 性别 | Msex | varchar | 2 | No | 男、女 |
| 年龄 | Mage | int | 2 | Yes |  |
| 密码 | Mpassword | varchar | 10 | Yes |  |
| 联系电话 | Mtel | varchar | 11 | No |  |
| 邮箱 | Memail | varchar | 25 | Yes |  |

**5.6 班级逻辑结构设计**

班级的逻辑结构设计如表5.6所示。

表5.6 班级表（Class表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 班级号 | CLno | varchar | 10 | No | 主码 |
| 班级人数 | CLnum | varchar | 11 | Yes |  |
| 学院 | CLdept | char | 50 | Yes |  |
| 专业 | CLmajor | char | 50 | Yes |  |

**5.7 班级课程逻辑结构设计**

班级课程的逻辑结构设计如表5.7所示。

表5.7 班级课程表（ClassCourse表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 班级号 | CLno | varchar | 20 | No | 主码 |
| 课程号 | Cno | varchar | 20 | No | 主码 |
| 学年 | Cyear | varchar | 20 | Yes |  |
| 学期 | Cterm | varchar | 20 | Yes |  |

**5.8 班级教师逻辑结构设计**

班级教师的逻辑结构设计如表5.8所示。

表5.8 班级教师表（ClassTeacher表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 班级号 | CLno | varchar | 20 | No | 主码 |
| 教师号 | Tno | varchar | 20 | No | 主码 |
| 学年 | Cyear | varchar | 20 | Yes |  |
| 学期 | Cterm | varchar | 20 | Yes |  |

**5.9 考试逻辑结构设计**

考试的逻辑结构设计如表5.9所示。

表5.9 考试表（Exam表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 班级号 | Eclno | varchar | 2 | No | 主码 |
| 课程号 | Ecno | varchar | 10 | No | 主码 |
| 考试地点 | Eposition | varchar | 10 | Yes |  |
| 开始时间 | Etime | Datetime | 6 | Yes |  |
| 结束时间 | Endtime | Datetime | 6 | Yes |  |
| 考试编号 | Eno | char | 20 | No | 主码 |

**5.10 授课逻辑结构设计**

教师的逻辑结构设计如表5.10所示。

表5.10 教授课程表（Teaching表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 中文含义 | 字段名称 | 数据类型 | 长度 | 是否为空 | 备注 |
| 教师号 | Tno | varchar | 10 | No | 主码 |
| 课程号 | Cno | varchar | 10 | No | 主码 |
| 上课地点 | Cposition | varchar | 20 | Yes |  |
| 上课周次 | Cweek | varchar | 20 | Yes |  |
| 上课时间1 | Ctime1 | varchar | 30 | Yes |  |
| 上课时间2 | Ctime2 | varchar | 30 | Yes |  |
| 学年 | Cyear | varchar | 20 | Yes |  |
| 学期 | Cterm | varchar | 20 | Yes |  |

**6 物理结构设计**

为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程，就是数据库的物理设计。确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构。

**6.1 存取方法的设计**

常用的存取方法为索引方法和聚簇方法。B+树索引和hash索引是数据库中经典的存取方法，使用最普遍。

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | 建立索引的属性 | 索引类型 | 存取方法 |
| class表 | 班号 | NORMAL类型 | B+树索引 |
| classcourse表 | 班号、课程号 | NORMAL类型 | B+树索引 |
| Classteacher表 | 教师号、班号 | NORMAL类型 | B+树索引 |
| course表 | 课程号 | NORMAL类型 | B+树索引 |
| csgrade表 | 学号、课程号 | NORMAL类型 | B+树索引 |
| Manager表 | 管理员号 | NORMAL类型 | B+树索引 |
| student表 | 学号、班级号 | NORMAL类型 | B+树索引 |
| teacher表 | 教师号 | NORMAL类型 | B+树索引 |
| teaching表 | 教师号、课程号 | NORMAL类型 | B+树索引 |

**7 概要设计**

**7.1 系统总体设计概述**

经过需求分析，已经明确系统必须要“做什么”，接下来就要确定“怎么做”了。系统总体设计的一个很好的方法就是要按照系统要实现的功能进行模块划分，把联系紧密的功能实现划分到相同模块中，模块之间的划分遵循“高内聚、低耦合”的原则。

**7.2 系统功能模块划分**

该系统应该完成以下两个方面的内容:学生信息资料的管理和学生成绩的管理(包括必修课和选修课),每个内容均需要提供添加、修改和查询的功能。该系统涉及到的用户包括教师、学生和管理员，各类用户的权限不一样。因此，把学生成绩管理系统划分为登录模块、学生模块、教师模块、管理员模块四大模块。模块划分如图7-1所示。

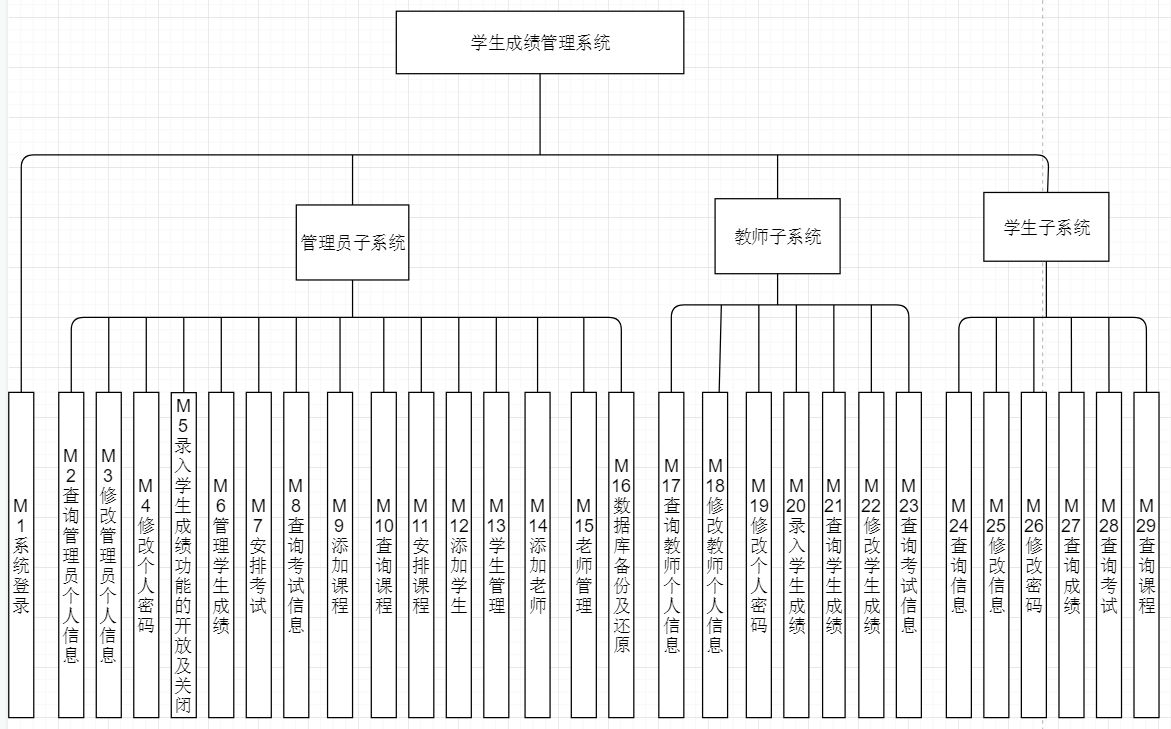


图7-1 系统总功能图

**7.3 系统功能模块清单**

对图7-1所示的系统各个功能模块进行细化：

* M1系统登录：用户根据自己的身份选择相应选项进行登录，进入对应身份的界面。
* M2个人信息查询: 管理员查看个人的信息
* M3个人信息修改：管理员修改个人信息。
* M4修改个人密码：管理员通过输入旧密码、输入新密码和再次输入新密码确认密码来修改自己的密码。
* M5教师录入学生成绩功能的开启和关闭：管理员开启成绩录入功能，教师才能录入成绩，管理员关闭此功能，教师不能再录入成绩。
* M6管理学生成绩：通过学年、学期、班级查询、修改、删除学生成绩，班级成绩情况可以用饼状图显示各科成绩平均分以及所占比重，也可通过学年、学期、学号来查询、修改、删除学生成绩，所查询到的学生成绩表可保存到本地文件夹。
* M7安排考试：管理员通过输入考试编号和考试地点、选择考试开始时间和结束时间、选择已存在的班级和已安排的课程来安排考试。
* M8查询考试：课程号为空时，管理员可通过学年、学期和班级号查询整个班全部课程的考试安排，课程号为已安排课程时，管理员通过学年、学期、班级和课程号查询班级的相应课程的考试安排。
* M9添加课程：管理员通过输入课程编号、课程名称、课程学时、课程学分和选择课程属性来添加课程。
* M10查询课程：管理员选择学院和专业来查询、修改、删除课程，学院、专业为空时查询所有课程，课程可分为已安排课程和未安排课程， 查询到的课程表可导出保存到本地。
* M11安排课程：管理员选择上课班级、学年、学期、科目和没有教课的老师、输入上课地点、上课周次、上课时间来安排课程。
* M12添加学生：管理员输入学生的信息包括学号、姓名、性别、年龄、班级、联系方式和邮箱来添加学生。
* M13学生管理：管理员可通过班级查询整个班级的学生信息，也可通过学号查询某个学生的信息，可修改选中的学生的学号、密码和班级，可删除选中的学生。
* M14添加老师：管理员输入老师的信息包括教师号、姓名、性别、年龄、学院、学历、职称、职务、联系方式和邮箱来添加老师。
* M15老师管理：管理员通过教师号查询老师信息，可修改、删除选中老师的信息，可初始化老师的密码为123456。
* M16数据库备份及还原：管理员进行数据库备份和还原。
* M17查询个人信息：老师查看个人信息。
* M18修改个人信息：老师修改个人信息。
* M19修改密码：老师通过输入旧密码、输入新密码和再次输入新密码确认密码来修改自己的密码。
* M20录入学生成绩：在管理员开启成绩录入功能后，老师选择班级、学生、科目、输入成绩来录入学生成绩。
* M21查询学生成绩：老师通过选择班级和科目来查询整个班级的学生的成绩，整个班级的成绩可以按升序或降序排序、可以求平均成绩，也可通过输入学号并选择课程来查询某个学生某课程的成绩，查询得到的成绩表可以导出到本地保存。
* M22修改学生成绩：老师可以通过输入学生学号和姓名、选择科目和输入成绩来修改学生的成绩。
* M23查询考试信息：老师通过选择科目来查询考试信息。
* M24查询信息：学生查看个人信息。
* M25修改信息：学生修改个人信息。
* M26修改密码：学生通过输入旧密码、输入新密码和再次输入新密码确认密码来修改自己的密码。
* M27查询成绩：学生通过选择学年、学期查询自己的成绩，成绩可以图表显示，查询得到的成绩表可以导出保存到本地。
* M28查询考试：学生可以查询所修课程的考试安排。
* M29查询课程：学生通过选择学年和学期查询课程。

**8 详细设计**

**8.1 登录模块设计**

8.1.1 SQL语句设计

* 根据用户输入的学号查询该学生的学号和密码

select Sno,Spassword from student where Sno=' student.getSno() ';

* 根据用户输入的教师号查询该教师的教师号和密码

select Tno,Tpassword from teacher where Tno= ' teacher.getTno() ';

* 根据用户输入的管理员号查询该管理员的管理员号和密码

select Mno,Mpassword from manager where Mno=' admin.getMno()';

8.1.2 LoginGUI类设计

* 判断用户输入的用户名和密码以及登录身份是否正确：

void handle\_enter(String userId, String userPwd, String identity) throws SQLException {

int result = 0;

if (identity.equals("学生")) { //如果用户选择的登录身份是 学生

student = new Student(userId, userPwd);

result = handlestu.checkStu(student);//检测用户登录名,密码是否正确

} else if (identity.equals("教师")) { //如果用户选择的登录身份是 教师

teacher = new Teacher(userId, userPwd);

result = handletea.checkTea(teacher);//检测用户登录名,密码是否正确

} else if (identity.equals("管理员")) { //如果用户选择的登录身份是 管理员

admin = new Admin(userId, userPwd);

result = handleadm.checkAdn(admin);//检测用户登录名,密码是否正确

}

/\*

\* -1:用户名不存在(用户名输入不正确)

\* 0:用户密码错误

\* 1:用户名密码均正确

\*/

if (result == -1) {

JOptionPane.showMessageDialog(this, "该用户名不存在，请重新输入用户名！", "消息对话框", JOptionPane.ERROR\_MESSAGE);

} else if (result == 0) {

JOptionPane.showMessageDialog(this, "密码输入错误，请重新输入密码！", "消息对话框", JOptionPane.ERROR\_MESSAGE);

} else if (result == 1 && identity.equals("学生")) {

Student new\_student = HandleStu.getStuInfo(student.getSno());

StudentMainJFrame studentMainJFrame = new StudentMainJFrame(new\_student);//进入学生主界面

studentMainJFrame.setVisible(true);

this.dispose();

} else if (result == 1 && identity.equals("教师")) {

TeaMainGUI teamaingui = new TeaMainGUI(teacher);//进入教师主界面

teamaingui.setVisible(true);

this.dispose();

} else if (result == 1 && identity.equals("管理员")) {

AdminMainGUI adminMainGUI = new AdminMainGUI(admin);//进入管理员主界面

adminMainGUI.setVisible(true);

this.dispose();

}

}

}

8.1.3 界面效果



图8-1 登录界面

**8.2 添加课程功能模块设计**

8.2.1 SQL语句设计

* 通过输入的课程号、课程名、学时、学分和属性向课程表里面添加一门课程

Insert into course (Cno,Cname,Cperiod,Ccredit,Cattribute) values (?,?,?,?,?);

8.2.2 CourseAddJpanel类设计

* 添加课程：

public static boolean InsertCourse(String Con, String Cname, int Cperiod, float Ccredit, String Cattribute) {

try {

statement = connect.prepareStatement(insertSQL);

statement.setString(1, Con);

statement.setString(2, Cname);

statement.setInt(3, Cperiod);

statement.setFloat(4, Ccredit);

statement.setString(5, Cattribute);

int result = statement.executeUpdate();

if (result > 0) {

return true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

8.2.3 界面效果

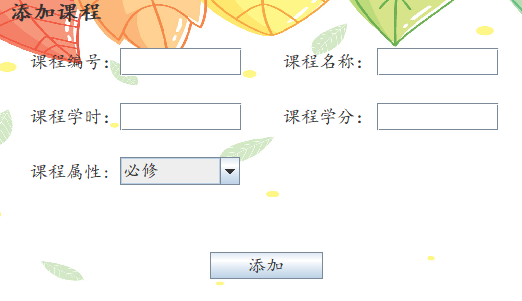


图8-2 添加课程界面

**8.3 查询课程功能模块设计**

8.3.1 SQL语句设计

* 查询所有学院的已安排或未安排课程的信息

已安排：select \* from course where course.Cno in (select teaching.Cno from teaching);

未安排：select \* from course where course.Cno not in (select teaching.Cno from teaching);

* 通过选定学院和选定专业来查询已安排或未安排课程的信息

已安排：select distinct course.\*,class.CLmajor,classcourse.Cyear,classcourse.Cterm from course,class,classcourse where course.Cno = classcourse.Cno and class.CLno = classcourse.CLno and class.CLdept=? and class.CLmajor=? and course.Cno in (select teaching.Cno from teaching);

未安排：select distinct course.\*,class.CLmajor,classcourse.Cyear,classcourse.Cterm from course,class,classcourse where course.Cno = classcourse.Cno and class.CLno = classcourse.CLno and class.CLdept=? and class.CLmajor=? and course.Cno not in (select teaching.Cno from teaching);

8.3.2 CourseSearchJPanel类设计

* 通过选定学院和选定专业来查询已安排或未安排课程的信息

public static String[][] getCourseByMajor(String dept,String major,boolean arr){

String[][] result = new String[0][];

ResultSet resultSet = null;

int count = 0;

try{

if (arr){

statement = connect.prepareStatement(searchArrByMajor);

}else {

statement = connect.prepareStatement(searchNotArrByMajor);

}

statement.setString(1,dept);

statement.setString(2,major);

resultSet = statement.executeQuery();

while (resultSet.next()){

count ++;

}

result = new String[count][8];

count = 0;

resultSet = statement.executeQuery();

while (resultSet.next()){

result[count][0] = resultSet.getString(1);

result[count][1] = resultSet.getString(2);

result[count][2] = String.valueOf(resultSet.getInt(3));

result[count][3] = String.valueOf(resultSet.getDouble(4));

result[count][4] = resultSet.getString(5);

result[count][5] = resultSet.getString(6);

result[count][6] = resultSet.getString(7);

result[count][7] = resultSet.getString(8);

count ++;

}

} catch (SQLException e) {

e.printStackTrace();

}

return result;

}

8.3.3 界面效果

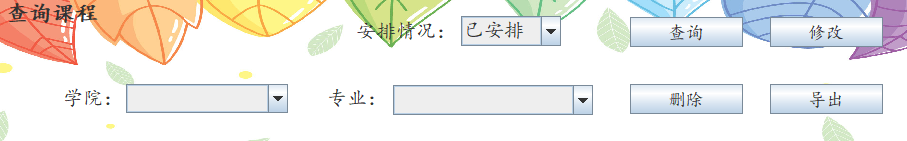


图8-3 查询课程界面

**8.4 安排课程功能模块设计**

8.4.1 SQL语句设计

* 通过用户选定的教师、课程名、输入的上课地点、周次、上课时间、学年和学期向教授表(teaching表)中添加一条课程安排记录

insert into teaching (Tno,Cno,Cposition,Cweek,Ctime1,Ctime2,Cyear,Cterm) values (?,?,?,?,?,?,?,?);

8.4.2 CourseArrangeJPanel类设计

* 添加课程安排

public static boolean InsertCourse(String Con, String Cname, int Cperiod, float Ccredit, String Cattribute) {

try {

statement = connect.prepareStatement(insertSQL);

statement.setString(1, Con);

statement.setString(2, Cname);

statement.setInt(3, Cperiod);

statement.setFloat(4, Ccredit);

statement.setString(5, Cattribute);

int result = statement.executeUpdate();

if (result > 0) {

return true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

8.4.3 界面效果



图8-4 安排课程界面

**8.5 添加师生功能模块设计**

8.5.1 SQL语句设计

* 通过输入的学号、姓名、性别、年龄、班级、默认密码、联系方式和邮箱向学生表(student表)里添加一个学生记录

insert into student(Sno,Sname,Ssex,Sage,Sclass,Spassword,Stel,Semail)

value(?,?,?,?,?,?,?,?);

* 通过输入的教师号、姓名、性别、年龄、学院、学历、职称、职务、联系方式、邮箱、默认密码向教师表(teacher表)里添加一个教师记录

insert into teacher(Tno,Tname,Tsex,Tage,Tdept,Tdegree,Ttitle,Tduty,Ttel,Temail,Tpassword) value(?,?,?,?,?,?,?,?,?,?,?);

8.5.2 StuAddJPanel类和TeaAddJPanel类设计

* 添加学生

public static boolean InsertStudent(String Sno, String Sname, String Ssex, String Sage, String Sclass, String Spassword, String Stel, String Semail) {

try {

PreparedStatement statement = connection.prepareStatement(InsertStudentSQL);

statement.setString(1, Sno);

statement.setString(2, Sname);

statement.setString(3, Ssex);

statement.setString(4, Sage);

statement.setString(5, Sclass);

statement.setString(6, Spassword);

statement.setString(7, Stel);

statement.setString(8, Semail);

int result = statement.executeUpdate();

if (result > 0) {

return true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

* 添加教师

public static boolean InsertTeacher(String Tno,String Tname,String Tsex,String Tage,String Tdept,String Tdegree,String Ttitle,String Tduty,String Ttel,String Temail,String Tpassword) {

try{

PreparedStatement statement = con.prepareStatement(InsertSQL);

statement.setString(1,Tno);

statement.setString(2,Tname);

statement.setString(3,Tsex);

statement.setString(4,Tage);

statement.setString(5,Tdept);

statement.setString(6,Tdegree);

statement.setString(7,Ttitle);

statement.setString(8,Tduty);

statement.setString(9,Ttel);

statement.setString(10,Temail);

statement.setString(11,Tpassword);

int result = statement.executeUpdate();

if(result > 0){

return true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

8.5.3 界面效果



图8-5 添加学生界面



图8-5 添加教师界面

**8.6 师生管理功能模块设计**

8.6.1 SQL语句设计

* 通过选定的班级号查询该班级的所有学生的信息

select \* from student where Sclass=?;

* 通过输入的学号查询指定学生的信息

select \* from student where Sno=?;

* 通过指定的教师号查询教师的信息

select Tno,Tname,Tsex,Tage,Tdept,Tdegree,Ttitle,Tduty,Ttel, Temail from Teacher where Tno=?;

* 通过学号从student表里删除指定的学生

DELETE FROM Student WHERE Sno=?;

* 通过学号修改指定学生的相关信息

update student set Spassword=?,Sclass=? where Sno=?;

* 通过教师号修改指定教师的相关信息

update teacher set Tname=?,Tsex=?,Tage=?,Tdept=?,Ttel=?,Temail=? where Tno=?;

* 通过教师号从teacher表里删除指定的教师

delete from teacher where Tno=?;

8.6.2 StuManageJPanel类和TeaManageJPanel类设计

* 通过班级号或者学号查询学生信息

String sClass = classComboBox.getSelectedItem().toString();

String Sno = textField.getText();

Object name[] = {"学号","密码","姓名", "性别", "年龄", "班级", "电话","邮箱"};

remove(table);

//查询某个班级的学生

if (Sno.equals("")) {

Object data[][] = HandleStu.getAllStudent(sClass);

table = new JTable(data, name);

}

else { //查询指定学号的学生

Object data[][] = HandleStu.getStudentBySno(Sno);

table = new JTable(data, name);}

执行查询语句并返回查询结果：

public static String[][] getStudentBySno(String Sno) {

String[][] result = new String[0][];

try {

statement = connection.prepareStatement(getStuBySnoSQL);

statement.setString(1, Sno);

result = getAll(statement);

return result;

} catch (SQLException e) {

e.printStackTrace();

}

return result;

}

* 删除表格指定的教师

int index = table.getSelectedRow();

if (index == -1){

DialogUtil.showMessage("提示信息","未选择老师，无法进行删除！");

}else {

int row = table.getSelectedRow();

String Tno = (String) table.getValueAt(row, 0);

boolean result = HandleTea.DeleteTeacher(Tno);

if (result) {

DialogUtil.showMessage("提示信息", "删除信息成功！");

sureButton.setEnabled(false);

deleteButton.setEnabled(false);

changeButton.setEnabled(false);

} else {

DialogUtil.showMessage("提示信息", "删除信息失败，请检查是否有误！");

}

}

执行删除语句：

public static boolean DeleteTeacher(String Tno) {

try {

PreparedStatement statement = con.prepareStatement(DeleteSQL);

statement.setString(1,Tno);

int result = statement.executeUpdate();

if(result > 0){

return true;

}

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

8.6.3 界面效果



图8-6 学生管理界面

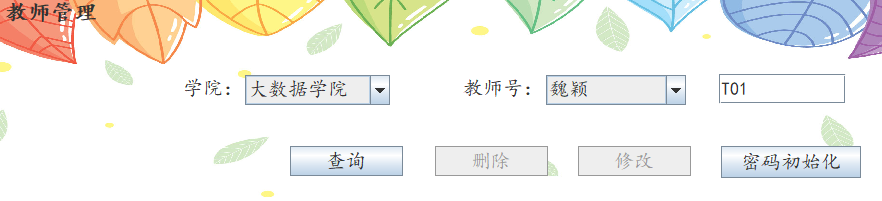


图8-6 教师管理界面

**9 测试**

**9.1 功能测试**

9.1.1 登录功能测试

测试目的：用户在登录过程中需要正确的用户名和用户密码，同时，用户身份的不同，进入不同的主界面，显示的功能不同。具体登录功能等价类表如表9.1所示。

表9.1 登录功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 无效等价类 | 用户名：空  密码：空  登录身份：选择或未选择 | 用户名：  密码：  登录身份：选择或未选择 | 登录不成功 | 弹出出错消息，提示请输入用户名和密码 |
| 用户名：非空  密码：空  登录身份：选择或未选择 | 用户名：111  密码：  登录身份：选择或未选择 | 登录不成功 | 弹出出错消息，提示请输入密码 |
| 用户名：空  密码：非空  登录身份：选择或未选择 | 用户名：  密码：111  登录身份：选择或未选择 | 登录不成功 | 弹出出错消息，提示请输入用户名 |
| 用户名：正确  密码：不正确  登录身份：正确 | 用户名：T01  密码：111111  登录身份：教师 | 登录不成功 | 弹出出错消息，提示密码错误 |
| 用户名：正确  密码：不正确  登录身份：不正确 | 用户名：T01  密码：111111  登录身份：学生 | 登录不成功 | 弹出出错消息，提示该用户名不存在 |
| 用户名：正确  密码：不正确  登录身份：未选择 | 用户名：T01  密码：111111  登录身份：未选择 | 登录不成功 | 弹出出错消息，提示选择登录身份 |
| 用户名：不正确  密码：正确  登录身份：正确 | 用户名：T01  密码：111111  登录身份：教师 | 登录不成功 | 弹出出错消息，提示该用户名不存在 |
| 用户名：不正确  密码：正确  登录身份：不正确 | 用户名：T01  密码：111111  登录身份：学生 | 登录不成功 | 弹出出错消息，提示该用户名不存在 |
| 用户名：不正确  密码：正确  登录身份：未选择 | 用户名：T01  密码：111111  登录身份：未选择 | 登录不成功 | 弹出出错消息，提示选择登录身份 |
| 有效等价类 | 用户名：正确  密码：正确  登录身份：正确 | 用户名：T01  密码：123456  登录身份：教师 | 登录成功 | 进入用户的主界面 |

**用例 1：**

* 测试条件：用户名和密码均为空，登录身份选择或未选择
* 测试用户：学生、教师、管理员
* 测试步骤：不输入用户名和用户密码，选择或不选择登录身份，直接点击登录按钮
* 预期结果：登录失败，提示请输入用户名和密码
* 实际结果：登录失败，提示请输入用户名和密码

具体结果图如图9-1所示。

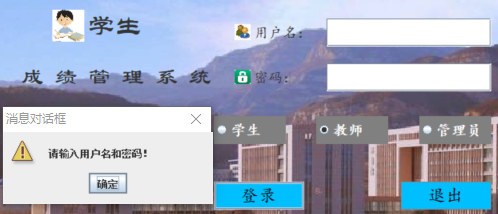


图9-1 登录功能测试用例一结果图

**用例 2：**

* 测试条件：用户名或密码为空，登录身份选择或未选择
* 测试用户：学生、教师、管理员
* 测试步骤：不输入用户名或密码，选择或不选择登录身份，点击登录按钮
* 预期结果：登录失败，提示输入用户名或密码
* 实际结果：登录失败，提示输入用户名或密码

具体结果图如图9-2和图9-3所示。

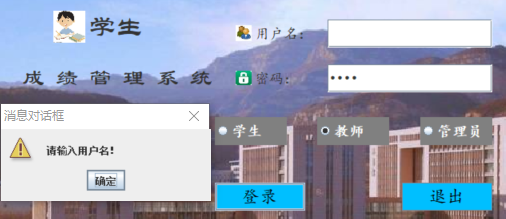


图9-2 登录功能测试用例二结果图

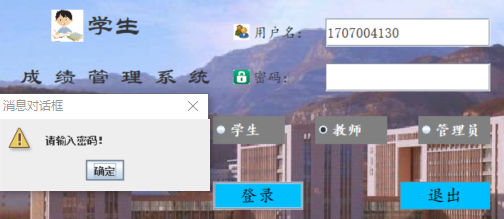


图9-3 登录功能测试用例二结果图

**用例 3：**

* 测试条件：用户名或密码错误，登录身份正确
* 测试用户：学生、教师、管理员
* 测试步骤：输入错误的用户名或密码，登录身份正确，点击登录按钮
* 预期结果：登录失败，提示用户名不存在或密码错误
* 实际结果：登录失败，提示用户名不存在或密码错误

具体结果图如图9-4和图9-5所示。



图9-4 登录功能测试用例三结果图



图9-5 登录功能测试用例三结果图

**用例 4：**

* 测试条件：用户名或密码错误，登录身份不正确
* 测试用户：学生、教师、管理员
* 测试步骤：输入错误的用户名或密码，登录身份不正确
* 预期结果：登录失败，提示用户名不存在
* 实际结果：登录失败，提示用户名不存在

具体结果图如图9-6和图9-7所示。



图9-6 登录功能测试用例四结果图



图9-7 登录功能测试用例四结果图

**用例 5：**

* 测试条件：用户名或密码错误，登录身份未选择
* 测试用户：学生、教师、管理员
* 测试步骤：输入错误的用户名或密码，登录身份未选择
* 预期结果：登录失败，提示选择登录身份
* 实际结果：登录失败，提示选择登录身份

具体结果图如图9-8和图9-9所示。



图9-8 登录功能测试用例五结果图

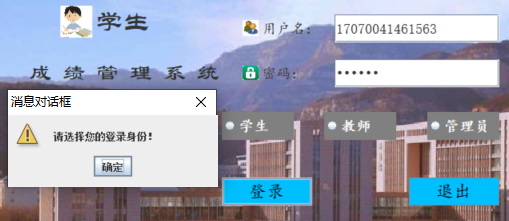


图9-9 登录功能测试用例五结果图

**用例 6：**

* 测试条件：用户名、密码、登录身份均正确
* 测试用户：学生、教师、管理员
* 测试步骤：输入正确的用户名和密码，登录身份正确
* 预期结果：登录成功，进入各自的主界面
* 实际结果：登录成功，进入各自的主界面

具体结果图如图9-10、图9-11和图9-12所示。



图9-10 登录功能测试用例六结果图

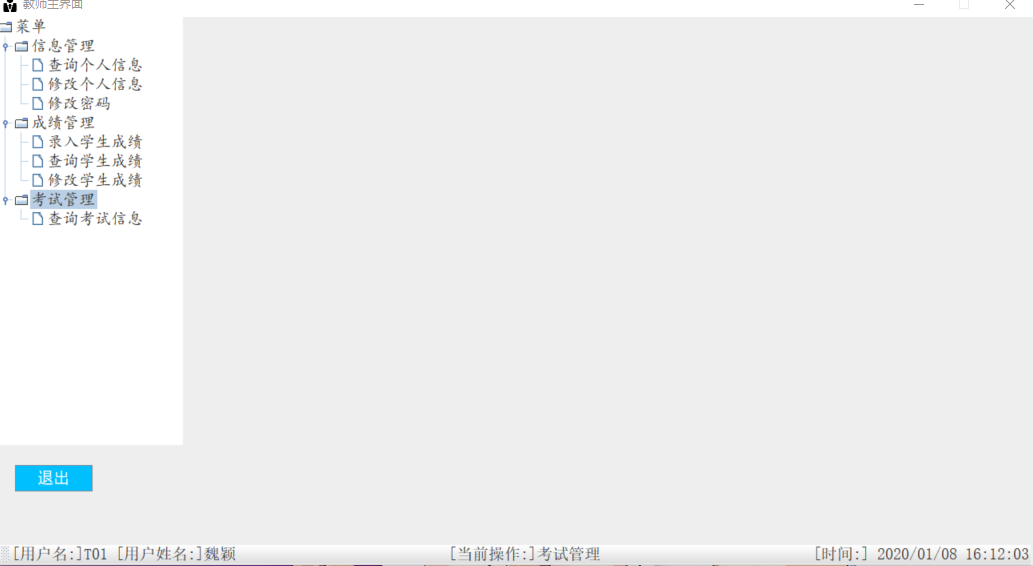


图9-11 登录功能测试用例六结果图

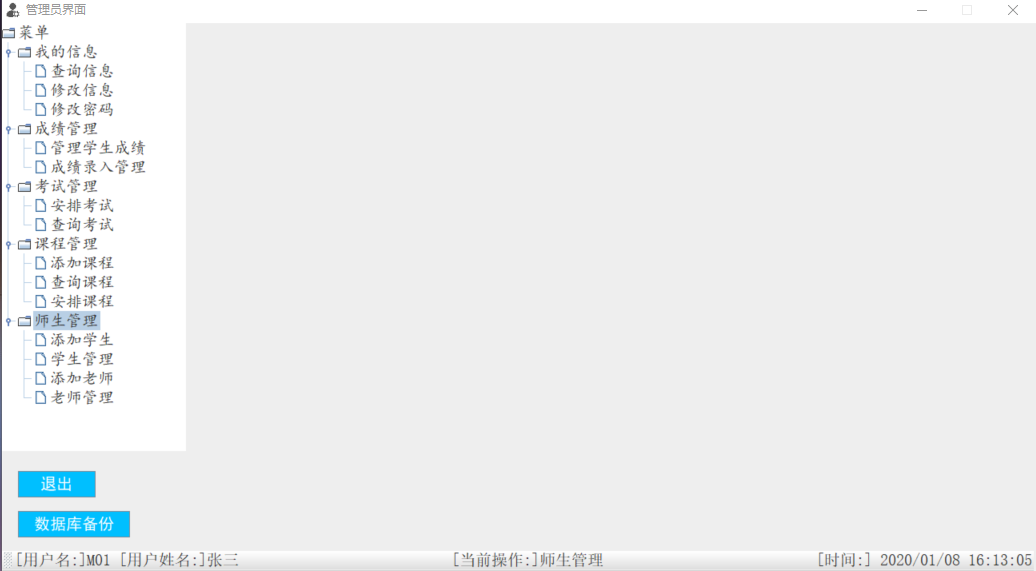


图9-12 登录功能测试用例六结果图

9.1.2 添加课程功能测试

测试目的：管理员能添加新课程的信息到数据库中。具体添加功能等价类表如表9.2所示。

表9.2 添加功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 无效等价类 | 课程编号、课程名称、课程学时和课程学分至少一个为空 | 课程编号、课程名称、课程学时和课程学分至少一个为空 | 无效输入 | 提示请输入完整的信息 |
| 有效等价类 | 课程编号、课程名称、课程学时、课程学分、课程属性均非空 | 课程编号：A01482675  课程名称：电路电子技术  课程学时：48  课程学分：4  课程属性：必修 | 有效输入 | 提示添加课程成功 |

**用例 1：**

* 测试条件：输入新课程信息时有的信息为空
* 测试用户：管理员
* 测试步骤：在输入新课程时，不输入必填项中的全部信息，点击添加按钮
* 预期结果：系统提示必填项不能为空，请输入完整信息
* 实际结果：系统提示必填项不能为空，请输入完整信息

具体结果图如图9-13所示。



图9-13 添加课程功能测试用例一结果图

**用例 2：**

* 测试条件：输入新课程信息时所有信息均非空
* 测试用户：管理员
* 测试步骤：在输入新课程时，输入必填项中的全部信息，点击添加按钮
* 预期结果：添加课程成功，并提示添加课程成功
* 实际结果：添加课程成功，并提示添加课程成功

具体结果图如图9-14所示。



图9-14 添加课程功能测试用例二结果图

9.1.3 查询课程功能测试

测试目的：管理员根据查询条件能查询到想要的课程信息

具体添加课程功能等价类表如表9.3所示。

表9.3 查询课程功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 有效等价类 | 安排情况：已安排或未安排 | 安排情况：已安排或未安排 | 有效查询 | 所有已安排或未安排的课程 |
| 安排情况：已安排或未安排  学院：非空  专业：非空 | 安排情况：已安排或未安排  学院：大数据学院  专业：计算机科学与技术 | 有效查询 | 大数据学院计算机科学与技术专业所有已安排或未安排的课程 |

**用例 1：**

* 测试条件：安排情况已安排或未安排
* 测试用户：管理员
* 测试步骤：在查询课程时，安排情况为已安排或未安排，点击查询按钮
* 预期结果：界面显示所有已安排或未安排的课程
* 实际结果：界面显示所有已安排或未安排的课程

具体结果图如图9-15和图9-16所示。



图9-15 查询课程功能测试用例一结果图



图9-16 查询课程功能测试用例一结果图

**用例 2：**

* 测试条件：安排情况已安排或未安排，学院和专业非空
* 测试用户：管理员
* 测试步骤：在查询课程时，安排情况为已安排或未安排，学院和专业非空点击查询按钮
* 预期结果：界面显示指定学院和指定专业所有已安排或未安排的课程
* 实际结果：界面显示指定学院和指定专业所有已安排或未安排的课程

具体结果图如图9-17所示。



图9-17查询课程功能测试用例一结果图

9.1.4 修改和删除课程功能测试

测试目的：管理员能修改或删除选定的课程

具体修改或删除课程功能等价类表如表9.4所示。

表9.4 修改删除课程功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 无效等价类 | 修改或删除时选择框未选择课程 | 在修改或删除时选择框未选择课程 | 无法修改或删除 | 无法修改或删除，并提示选中课程 |
| 有效等价类 | 修改或删除时选中课程 | 在修改或删除时选中课程 | 可以修改或删除 | 可以修改或删除课程 |

**用例 1：**

* 测试条件：修改或删除时选择框未选择课程
* 测试用户：管理员
* 测试步骤：在修改或删除时选择框未选择课程，点击修改或删除按钮
* 预期结果：无法修改或删除，并提示选中课程
* 实际结果：无法修改或删除，并提示选中课程

具体结果图如图9-18所示。

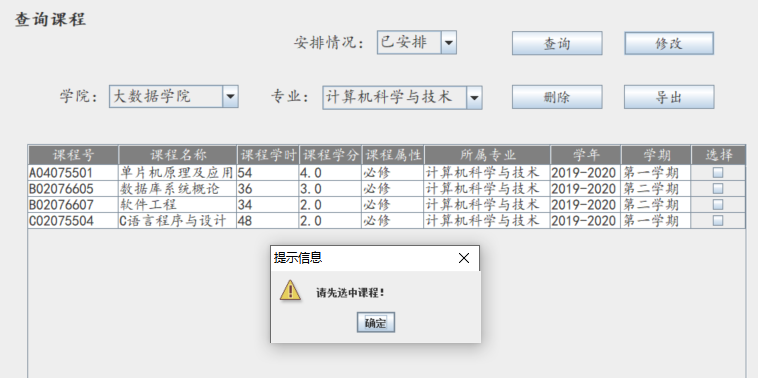


图9-18 修改或删除课程功能测试用例一结果图

**用例 2：**

* 测试条件：修改或删除时选择框选中课程
* 测试用户：管理员
* 测试步骤：在修改或删除时选择框选中课程，点击修改或删除按钮
* 预期结果：可以修改或删除课程，并提示修改或删除成功
* 实际结果：可以修改或删除课程，并提示修改或删除成功

具体结果图如图9-19和图9-20所示。



图9-19 修改和删除课程功能测试用例二结果图



图9-20 修改和删除课程功能测试用例二结果图

9.1.5 安排课程功能测试

测试目的：管理员能安排具体课程的上课地点、上课老师、上课时间等信息

具体安排课程功能等价类表如表9.5所示。

表9.5 安排课程功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 无效等价类 | 安排课程时必填信息中有为空的 | 上课地点、上课周次、上课时间1和上课时间2至少有一个为空的 | 无法安排课程 | 无法安排课程并提示输入完整的信息 |
| 有效等价类 | 安排课程时必填信息均非空 | 上课地点、上课周次、上课时间1和上课时间2均非空 | 可以安排课程 | 可以安排课程 |

**用例 1：**

* 测试条件：安排课程时必填信息中有为空的
* 测试用户：管理员
* 测试步骤：在安排课程时，上课地点、上课周次、上课时间1和上课时间2至少有一个为空的，点击确定按钮
* 预期结果：无法安排课程并提示输入完整的信息
* 实际结果：无法安排课程并提示输入完整的信息

具体结果图如图9-21所示。

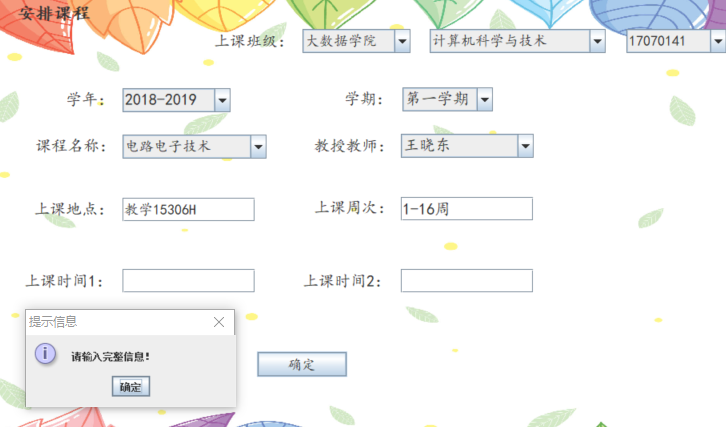


图9-21 安排课程功能测试用例一结果图

**用例 2：**

* 测试条件：安排课程时必填信息均非空
* 测试用户：管理员
* 测试步骤：在安排课程时，上课地点、上课周次、上课时间1和上课时间2均非空，点击确定按钮
* 预期结果：可以为选定学院的选定专业的指定班级安排某一学年某一学期的课程，安排成功，并提示安排课程成功
* 实际结果：可以为选定学院的选定专业的指定班级安排某一学年某一学期的课程，安排成功，并提示安排课程成功

具体结果图如图9-22所示。



图9-22 安排课程功能测试用例一结果图

9.1.6 添加师生功能测试

测试目的：管理员能添加教师和学生到数据库中

具体添加师生功能等价类表如表9.6所示。

表9.6添加师生功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 无效等价类 | 添加师生时必填信息中有为空的 | 添加师生时必填信息中有为空的 | 无法添加师生 | 无法添加师生，并提示输入完整信息 |
| 有效等价类 | 添加师生时必填信息均非空 | 添加师生时必填信息均非空 | 可以正常添加师生 | 可以正常添加师生，并提示添加成功 |

**用例 1：**

* 测试条件：添加师生时必填信息中有为空的
* 测试用户：管理员
* 测试步骤：在添加教师或学生到数据库中时，必填信息至少有一个为空的，点击确定按钮
* 预期结果：无法添加师生并提示输入完整的信息
* 实际结果：无法添加师生并提示输入完整的信息

具体结果图如图9-23和图9-24所示。



图9-23 添加师生功能测试用例一结果图



图9-24 添加师生功能测试用例一结果图

**用例 2：**

* 测试条件：添加师生时必填信息均非空
* 测试用户：管理员
* 测试步骤：在添加教师或学生到数据库中时，必填信息均非空，点击确定按钮
* 预期结果：可以添加教师和学生到数据库，并提示添加成功
* 实际结果：可以添加教师和学生到数据库，并提示添加成功

具体结果图如图9-25和图9-26所示。

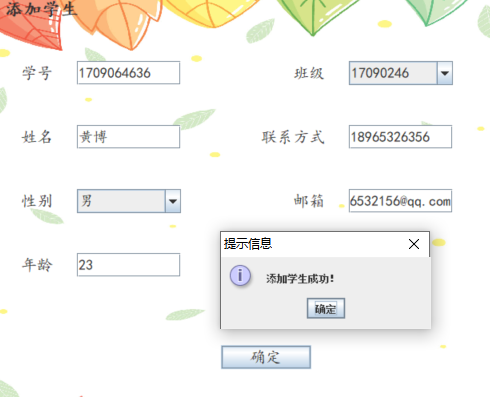


图9-25 添加师生功能测试用例二结果图



图9-26 添加师生功能测试用例二结果图

9.1.7 修改和删除师生功能测试

测试目的：管理员能修改或删除选定的教师或学生

具体修改或删除师生功能等价类表如表9.7所示。

表9.7 修改删除师生功能测试等价类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **输入等价类** | **测试用例** | **测试数据** | **期望结果** | **实际结果** |
| 无效等价类 | 修改或删除时选择框未选择教师或学生 | 在修改或删除时选择框未选择教师或学生 | 无法修改或删除 | 无法修改或删除，并提示选中教师或学生 |
| 有效等价类 | 修改或删除时选中教师或学生 | 在修改或删除时选中教师或学生课程 | 可以修改或删除 | 可以修改或删除教师或学生，并提示修改或删除成功 |

**用例 1：**

* 测试条件：修改或删除时选择框未选择教师或学生
* 测试用户：管理员
* 测试步骤：在修改或删除时选择框未选择教师或学生，点击修改或删除按钮
* 预期结果：无法修改或删除师生，并提示选中教师或学生
* 实际结果：无法修改或删除师生，并提示选中教师或学生

具体结果图如图9-27和图9-28所示。

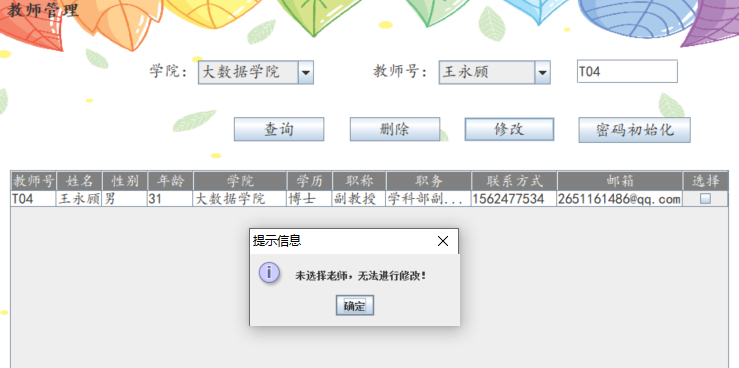


图9-27 修改或删除师生功能测试用例一结果图



图9-28 修改或删除师生功能测试用例一结果图

**用例 2：**

* 测试条件：修改或删除时选择框选中教师或学生
* 测试用户：管理员
* 测试步骤：在修改或删除时选择框选中教师或学生，点击修改或删除按钮
* 预期结果：可以修改或删除教师或学生，并提示修改或删除成功
* 实际结果：可以修改或删除教师或学生，并提示修改或删除成功

具体结果图如图9-29和图9-30所示。

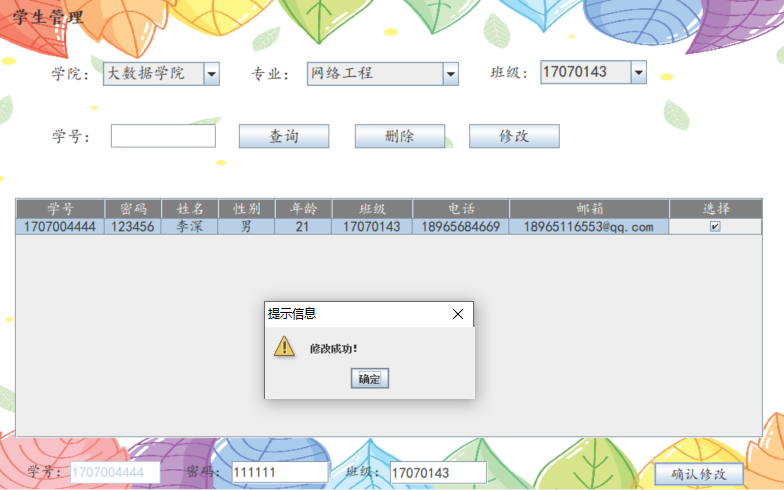


图9-29 修改和删除师生功能测试用例二结果图



图9-30 修改和删除师生功能测试用例二结果图

**9.2 用户界面测试**

用户界面（UI）测试用于核实用户与软件之间的交互。UI测试的目标是确保用户界面会通过测试对象的功能来为用户提供相应的访问或浏览功能。另外，UI测试还可确保UI中的对象按照预期的方式运行。

**9.3 测试结论**

具体测试结论表如表9.8所示。

表9.8 测试结论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能名称** | **功能描述** | **通过分析** | **通过否** | **备注** |
| 登录系统 | 用户输入用户名和密码并选择登录身份，可以进入各自的主界面 | 基本功能已实现 | 通过 |  |
| 课程管理 | 能够添加、删除、查询、修改、安排课程信息 | 基本功能已实现 | 通过 |  |
| 师生管理 | 能够添加、删除、查询、修改教师或学生信息 | 基本功能已实现 | 通过 |  |

**10 结论**

**10.1**

总的来说，是把以前老师讲的知识又串了一下，又熟悉了许多，很多以前没有弄明白的知识点，通过这次课程设计，差不多都掌握了，对于那些像数据库、表、视图、索引等的建立、使用方法和作用，我又有了更深一步的了解和掌握，也对SQL查询、修改、删除等语句的运用熟练了很多。同时对数据库设计的过程和方法有了更深的理解并将它运用到具体的管理系统的设计过程中，更深刻地理解了要做一个管理系统，需求分析、概念结构设计和逻辑结构设计是基础中的基础，是必要的也是必须的，尤其是需求分析，得先分析清楚这个系统涉及到哪些实体，它们各有什么属性，它们之间的联系是什么样的，否则的话，之后的设计过程就无法进行，或者要多次修改数据库中所建的表。

但是还有很多欠缺，就像我做这些东西，感觉没有一定的逻辑顺序，好像它们之间没有多大的联系似的，所以还要更进一步的完善，但是由于时间紧迫，没有办法完善它。做此课程设计过程中，得到老师和同学的大力支持和帮助，在此特别感谢。  
 在这次课程设计的过程中,我们体会到要想开发一个系统软件,不仅需要相当的专业技术知识，还要有严谨缜密的思维能力。只有思想上清晰了，编程才有意义，否则就是白费力气。同时还要善于捕获细小的方面，因为那往往是这个程序的致命因素。这次毕业设计培养了我的细心和耐性，更树立了一种科学的态度。这对我以后的工作和学习也有很大的帮助和指导作用。同时也深刻认识到了本身不存在很多不足之处，还需要不断地学习来充实完善自己，只有这样才能学有所成，求得更大的发展。

**参 考 文 献**

[1] 王珊,萨师煊. 数据库系统概论（第5版). 北京：高等教育出版社，2014.

[2] 马忠贵. 数据库技术及应用——Microsoft SQL Server 2008+Java. 北京：国防工业出版社, 2012.

[3] 李刚. Java程序员之旅--Java数据库技术详解. 北京：化学工业出版社，2010.

[4] 贾蓉生，胡大源，林金池. Java典型应用彻查1000例-数据库应用基础.

.北京：科学出版社.2009.

1. 郑阿奇. SQL Server实用教程（第4版）.北京：电子工业出版社，2015.