

成

绩



**课 程 设 计**

课程名称  数据库课程设技

题目名称 学生信息数据库管理

学生学院 计算机学院

专业班级 计算机科学与技术15级7班

学 号 3115005141

学生姓名 方锦基

指导教师 路 璐

2017 年 12 月10 日

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **评价标准** | | | | | **分数比例（%）** | **成绩** |
| 论文 | 论文结构包含：  1、相关技术介绍、需求分析、  2、概念结构设计（涉及的实体至少三个以上）、  3、逻辑结构设计（有完整性约束说明）、  4、数据库物理设计、  5、数据库完整性设计（违反实体、参照完整性时的解决办法，比如触发器、存储过程等）  5、数据库安全性设计、  6、数据库实施、系统测试方案和测试报告、  7、系统的主要功能和使用说明、系统安装说明。  要求论文完整、内容详细，格式规范。 | | | | | 40 |  |
| 程序 | 1. 系统运行正确；   2、功能完善：有增、删、改、查功能，输入、输出功能；  3、有基本的统计、报表功能  4、有多表连接查询、自身连接查询、字符串匹配查询、模糊查询、分组查询等。  5、工作量饱满；  6、系统实现技术的难度。 | | | | | 30 |  |
| 数据  库设  计 | E-R图设计正确，至少3个实体；  数据库逻辑结构设计规范化；  数据库物理设计合理。 | | | | | 30 |  |
| 总评成绩 | 优 | 良 | 中 | 及格 | 不及格 | 总分 |  |

**目 录**

[1 引言 1](#_Toc503874746)

[1.1 课程设计选题 1](#_Toc503874747)

[1.2 课程设计目的 1](#_Toc503874748)

[1.3 本选题的设计背景 1](#_Toc503874749)

[1.4 主要工作 1](#_Toc503874750)

[2 需求分析 2](#_Toc503874751)

[2.1 业务流程分析 2](#_Toc503874752)

[2.2 功能分析 3](#_Toc503874753)

[2.2.1 学生端 3](#_Toc503874754)

[2.2.2 教师端 4](#_Toc503874755)

[2.3 数据需求分析 4](#_Toc503874756)

[2.3.1 数据项 4](#_Toc503874757)

[2.3.2 数据结构 6](#_Toc503874758)

[2.3.3 数据流 7](#_Toc503874759)

[2.3.4 数据存储 8](#_Toc503874760)

[3 数据库设计 9](#_Toc503874761)

[3.1 概念结构设计 9](#_Toc503874762)

[3.1.1 基本信息分ER图设计 9](#_Toc503874763)

[3.1.2 课表信息分ER图设计 10](#_Toc503874764)

[3.1.3 项目信息分ER图设计 11](#_Toc503874765)

[3.1.4 申请信息分ER图设计 12](#_Toc503874766)

[3.1.5 视图集成 13](#_Toc503874767)

[3.2 逻辑结构设计 14](#_Toc503874768)

[3.3 完整性设计 15](#_Toc503874769)

[3.3.1 查询 15](#_Toc503874770)

[3.3.2 视图操作 15](#_Toc503874771)

[3.3.3 带参存储过程操作 15](#_Toc503874772)

[3.3.4 删除操作 15](#_Toc503874773)

[3.3.5 before触发器 15](#_Toc503874774)

[3.4 物理设计 17](#_Toc503874775)

[3.4.1 索引存取 17](#_Toc503874776)

[3.4.2 聚簇存取 17](#_Toc503874777)

[3.5 安全性设计 17](#_Toc503874778)

[4 应用系统设计及实现 18](#_Toc503874779)

[4.1 注册及登入 18](#_Toc503874780)

[4.2 学生端功能 18](#_Toc503874781)

[4.2.1 选课报名 18](#_Toc503874782)

[4.2.2 班级课表 19](#_Toc503874783)

[4.2.3 创新项目 20](#_Toc503874784)

[4.2.4 课程成绩 21](#_Toc503874785)

[4.2.5 考级功能 22](#_Toc503874786)

[4.3 教师端功能 23](#_Toc503874787)

[4.3.1 课表功能 23](#_Toc503874788)

[4.3.2 教室申请 24](#_Toc503874789)

[4.3.3 项目管理 25](#_Toc503874790)

[4.3.4 课程管理 26](#_Toc503874791)

[5 测试 27](#_Toc503874792)

[5.1 数据库系统 27](#_Toc503874793)

[5.2 应用系统 31](#_Toc503874794)

[6 关键代码 34](#_Toc503874795)

[6.1 建表代码示例 34](#_Toc503874796)

[6.2 建立完整性约束示例 34](#_Toc503874797)

[6.3 建立检索示例 35](#_Toc503874798)

[6.4 建立视图示例 35](#_Toc503874799)

[6.5 建立带参存储过程示例 35](#_Toc503874800)

[6.6 嵌套查询示例 35](#_Toc503874801)

[6.7 简单查询示例 36](#_Toc503874802)

[6.8 多表连接查询示例 36](#_Toc503874803)

[6.9 删除的before触发器示例 36](#_Toc503874804)

[7 个人感想 37](#_Toc503874805)

[8 参考书籍 38](#_Toc503874806)

[9 附录 38](#_Toc503874807)

# 引言

## 课程设计选题

《学生信息数据库管理》

## 课程设计目的

通过数据库系统课程设计，掌握Powerdesigner、MySQL及SQL语言开发关系数据库系统，使用Qt Creator、基于QT、C++语言开发具有图形化、交互性的C/S应用程序。进一步理解巩固并掌握关系数据库中数据结构、数据操作及完整性约束。进一步理解并掌握数据库的开发过程，提高数据库的分析设计能力、数据库系统的开发能力。

## 本选题的设计背景

本次选题选择较为熟悉的学生信息数据库管理，主要针对学生课程信息、大学生创新创业项目信息及相关其它的信息进行管理。

选择这三方面信息主要源于在使用我们的教务系统进行选课、查询课表信息时存在选课信息不同步、课表信息查询不具有个人化，以及缺少相关的申请信息的管理三方面问题，所以决定结合所学知识,一定程度上利用所学知识尝试解决实际的问题。

本次开发选用Powerdesigner、MySQLWorkbench开发工具进行关系数据库开发，源于这两款工具的人性化图形界面、数据管理的简单易用。

本次开发选用Qt Creator开发工具进行应用系统开发，采用面向对象编程和designer设计模式，进一步减轻应用系统图形化设计工作和编程工作，提高代码的可拓展性和修改性。

## 主要工作

本次课设在数据库系统和应用系统两部分中，从设计到实现均有本人完成，并最终进行文档的编写，其中问题的解决途径主要来源于两方面：一为Qt论坛、SQL论坛等线上讨论社区；二为参考相关的书籍。

# 需求分析

## 业务流程分析

学生信息数据库管理旨在通过管理学生在校信息，课表信息及其它信息，提高数据的检索、处理能力，从而满足学生查询课程安排等学习业务、教师查询课程安排等教学业务以及其余业务办理。从而解决信息不同步、信息查询个人化，以及相关申请信息管理三方面问题

以下为整个业务需求分析图：



图 2-1 整体业务概括

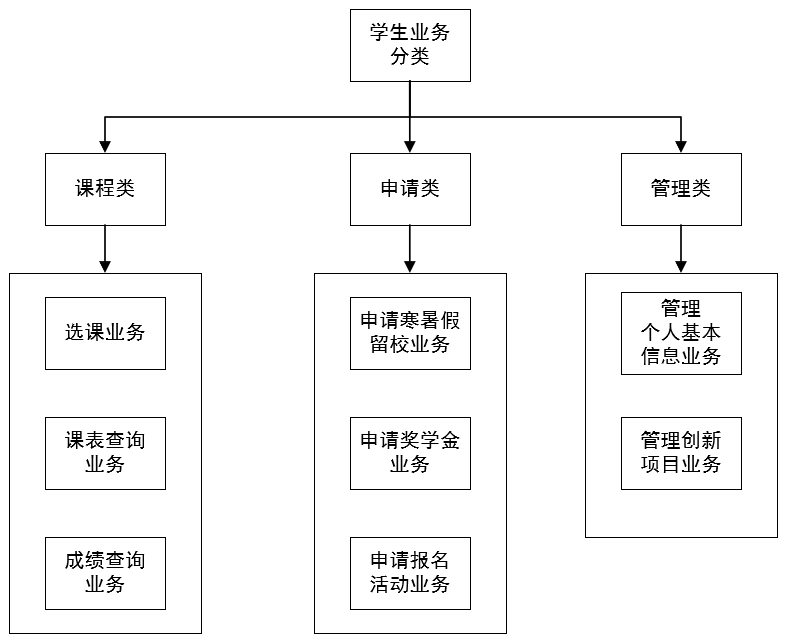


图 2-2学生业务



图 2-3教师业务

由以上业务需求分析图知，整个学生信息数据库管理系统分为学生业务和教师业务两类：学生业务包括选课等课程业务、留校等申请业务、个人信息等管理业务；教师业务包括排课等课程业务、课室等申请业务、个人信息等管理业务。

## 功能分析

结合业务分析，应当具有下列功能：

### 学生端

* 1. **选课**

选课功能提供给学生进行当前学期公共选修课、专业选修课的选择。按照要求每位同学公共选修课每学期至多选修两门，专业选修课按照专业培养计划进行选择。

* 1. **课表查询**

课表查询功能提供给学生进行上课安排的查询。

* 1. **成绩查询**

成绩查询功能提供给学生进行成绩查询。

* 1. **申请报名**

申请报名分为寒暑假留校、奖学金、其余活动三类。

* 1. **管理个人信息**

管理个人信息用于管理学生基本信息。

* 1. **管理创新项目**

管理创新项目用于提供给学生管理各类创新项目的进度。

### 教师端

1. **排课**

排课功能提供给教师进行当前学期教学课程的上课时间安排。系统具有自主排课功能，教师从而可以在此基础上进行课程上课安排调动。从而节省教师排课困难。

1. **课表查询**

课表查询功能提供给教师进行上课安排的查询。

1. **成绩统计**

成绩统计功能提供给教师进行所教课程的成绩评分。

1. **课室申请**

申请主要分为理论课课室申请以及实验课课室申请。系统具有自主安排教师功能，教师从而可以在此基础上进行教师安排的调动，从而节省教师课室申请的困难。

1. **管理个人信息**

管理个人信息用于管理教师基本信息以及教授专业范围。

1. **管理创新项目业务**

管理创新项目用于提供给教师对学生的各类创新项目进行监督及评价。

## 数据需求分析

结合以上功能分析，接下来采用数据字典及数据流图方法阐述数据种类及数据要求。

### 数据项

以下描述相关数据项的设计。一共可以分为基本信息子系统数据项、课程信息子系统数据项、项目信息子系统数据项、申请信息子系统数据项。

表2-1基本信息子系统数据项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项名 | 数据项含义说明 | 别名 | 数据类型 | 取值含义 |
| 院系名 |  | dpt\_name | varchar(20) |  |
| 院系编号 | 院系代码 | dpt\_number | varchar(6) |  |
| 专业名 |  | spe\_name | varchar(20) |  |
| 专业编号 | 专业代码 | spe\_number | varchar(6) |  |
| 班级编号 | 班级代码 | class\_number | varchar(6) |  |
| 学生名 |  | stu\_name | varchar(20) |  |
| 学生编号 | 学生学号 | stu\_number | varchar(10) |  |
| 教师名 |  | t\_name | varchar(20) |  |
| 教师编号 | 教师代码 | t\_ number | varchar(6) |  |
| 年龄 |  | Year | varchar(6) |  |
| 性别 |  | Sex | varchar(6) |  |
| 职称 | 教师职称 | Academic | varchar(20) |  |
| 入学日期 | 学生入学日期 | data | data |  |
| 住址 | 学生家庭地址 | address | varchar(20) |  |

表2-2课程信息子系统数据项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项名 | 数据项含义说明 | 别名 | 数据类型 | 取值含义 |
| 课程名 |  | course\_name | varchar(20) |  |
| 课程编号 | 课程代码 | course\_number | varchar(6) |  |
| 上课节次 | 当天第几课上课 | time | varchar(10) | 第几节课 |
| 上课周次 | 当学期第几周上课 | week | varchar(10) | 第几周 |
| 学时 | 总上课时间 | period | int(11) |  |
| 成绩 | 课程成绩 | grade | varchar(6) |  |
| 学分 | 课程学分 | credit | int(11) |  |
| 课程类别 | 课程类别 | sort | varchar(10) |  |
| 课室编号 |  | Classroom\_number | varchar(10) |  |

表2-3项目信息子系统数据项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项名 | 数据项含义说明 | 别名 | 数据类型 | 取值含义 |
| 项目名 |  | p\_name | varchar(30) |  |
| 项目编号 |  | p\_number | varchar(6) |  |
| 项目进度 |  | p\_state | varchar(20) |  |
| 项目等级 |  | p\_level | varchar(6) |  |

表2-4申请信息子系统数据项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据项名 | 数据项含义说明 | 别名 | 数据类型 | 取值含义 |
| 奖学金名 |  | scholarship\_number | varchar(20) |  |
| 等级 |  | level | varchar(6) |  |
| 奖金 |  | bonus | varchar(6) |  |
| 学期 |  | term | varchar(6) |  |
| 留校时间 |  | time | time |  |
| 留校类型 |  | stay\_sort | varchar(6) |  |
| 留校编号 |  | stay\_number | varchar(6) |  |

### 数据结构

以下描述相关的数据结构。一共可以分为基本信息子系统数据、课程信息子系统数据、项目信息子系统数据、申请信息子系统数据。

表2-5基本信息子系统数据

|  |  |  |
| --- | --- | --- |
| 数据结构名 | 含义说明 | 组成 |
| 院系 | 表示院系 | 院系编号、院系名 |
| 专业 | 表示专业 | 专业编号、专业名 |
| 班级 | 表示班级 | 班级编号、院系编号、专业编号 |
| 学生 | 表示学生基本信息 | 学生编号、学生名、院系编号、  专业编号、班级编号、入学日期、年龄、性别 |
| 教师 | 表示教师基本信息 | 教师编号、教师名、院系编号、学历、年龄、职称 |

表2-6课程信息子系统数据

|  |  |  |
| --- | --- | --- |
| 数据结构名 | 含义说明 | 组成 |
| 课程 | 表示课程基本信息 | 课程编号、课程名、学时、学分、  课程类别 |
| 课室 | 表示课室基本信息 | 课室编号、课室名 |
| 上课课程 | 表示上课课程基本信息 | 课室编号、课室名、上课节次、  上课周次、学时 |
| 教师课程 | 表示教师课程信息 | 课程编号、教师编号、上课课室 |
| 学生课程 | 表示学生课程信息 | 课程编号、学生编号、上课课室 |
| 课程成绩 | 表示学生课程成绩 | 课程编号、学生编号、教师编号、  成绩 |

表2-7项目信息子系统数据

|  |  |  |
| --- | --- | --- |
| 数据结构名 | 含义说明 | 组成 |
| 项目 | 表示项目信息 | 项目编号、项目名 |
| 学生教师项目 | 表示学生教师已开展项目情况 | 学生编号、教师编号、项目编号、  项目进度、项目等级 |

表2-8申请信息子系统数据

|  |  |  |
| --- | --- | --- |
| 数据结构名 | 含义说明 | 组成 |
| 奖学金 | 表示奖学金基本信息 | 奖学金名、奖学金等级、奖金 |
| 学生申请奖学金 | 表示学生申请奖学金 | 学生编号、奖学金名 |
| 留校申请信息 | 表生留校申请信息 | 留校编号、留校类型、学期、留校时间 |
| 学生留校申请 | 表示学生留下申请信息 | 学生编号、留校编号 |

### 数据流

以下描述相关数据的数据流。分为顶层数据流以及下层数据流图。



图 2-4 顶层数据流图



图 2-5基本信息下层数据流图



图 2-6课表信息下层数据流图



图 2-7 项目信息下层数据流图



图 2-8 申请信息下层数据流图

### 数据存储

本次为了提高数据的操作性能，将所有数据存放在云服务器。

通过将整个数据库系统部署到腾讯公有云上，通过互联网实现数据的在线流通，实现应用系统真正的在线使用。

其中部署在腾讯公有云中的数据库系统可以通过远程连接管理的方式，实现管理员管理数据库，并且在腾讯公有云中配置数据库管理系统MySQLWorkbench实现对数据库的可视化管理，提高数据的安全性保护，减轻管理难度。

# 数据库设计

## 概念结构设计

### 基本信息分ER图设计

以下描述学院、专业、班级、学生、教师这五个实体的基本信息关系图，具体如图：



图 3-1 基本信息分E-R图

### 课表信息分ER图设计

以下描述课程、课室、学生课程、教师课程的关系图，具体如图：



图 3-2 课表信息分E-R图

### 项目信息分ER图设计

以下描述项目、学生项目、教师项目关系图，具体如图：



图 3-3 项目信息分E-R图

### 申请信息分ER图设计

以下描述学生、留校、奖学金关系图，具体入图：



图 3-4 申请信息分E-R图

### 视图集成

结合基本信息分ER图、课表信息分ER图、项目信息分ER图、申请信息分ER图，以下为四个分ER图的集成，进一步描述学生、教师、课程、学生课程、教师课程、奖学金、成绩、项目、留校等实体间的关系，具体如图：



图 3-5 合E-R图

## 逻辑结构设计

表3-1逻辑结构设计

|  |  |  |  |
| --- | --- | --- | --- |
| 关系名 | 关系属性 | 主键约束 | 外键约束 |
| 学生 | stu\_number, stu\_name,  data,  year,  sex,  dpt\_number,  spe\_number,  class\_number | s\_number, | dpt\_number,  spe\_number,  class\_number |
| 教师 | t\_number,  t\_name,  dpt\_number,  academic,  year,  title | t\_number | dpt\_number |
| 上课课 程 | s\_number,  classroom\_number,  course\_number | s\_number,  classroom\_number,  course\_number | s\_number,  classroom\_number,  course\_number |
| 课室 | classroom\_number,  time,  week | classroom\_number |  |
| 奖学金 | scholarship\_name,  level,  bonus | scholarship\_name |  |
| 成绩 | s\_number,  t\_number,  course\_number,  grade | s\_number,  t\_number,  course\_number | s\_number,  t\_number,  course\_number, |
| 教师课程 | t\_number,  course\_number,  classroom\_number | t\_number,  classroom\_number | t\_number,  course\_number,  classroom\_number |
| 学生课程 | s\_number,  course\_number,  classroom\_number | s\_number,  classroom\_number | s\_number,  course\_number,  classroom\_number |
| 学生教师项目 | p\_number  p\_name  s\_number  t\_number | p\_number | s\_number  t\_number |
| 班级 | class\_number,  dpt\_number,  spe\_number | class\_number | dpt\_number,  spe\_number |
| 院系 | dpt\_number  dpt\_name | dpt\_number |  |
| 专业 | spe\_number  spe\_name | spe\_number |  |

## 完整性设计

为了确保数据库系统的完整性约束，在设计增删查改的接口时更多的使用以下设计的四种泛型模板。

### 查询

本次设计使用了嵌套查询和单表查询两种查询方式。用于解决固定化数据的查询。其中包括查询某个学生的成绩、查询某个学生的课表、查询某个学生的其它信息。

### 视图操作

对于复杂的查询采用视图操作，减少对数据的管理，提高数据修改的方便些。其中视图操作包括学生选课显示、班级课表显示、申请内容显示等需要显示的数据。

### 带参存储过程操作

对于需要输入的数据，采用带参的存储过程，从而便于更新数据。其中包括成绩的录入、创新项目的评价、创新项目的修改及其它相关申请。

### 删除操作

本次针对删除数据操作，采用级联删除，当删除一个主键，以此为参照的外键相关数据均会被删除。

### before触发器

针对修改数据，采用before触发器，若是修改主键：则先增加若干行待修改的数据，并将主键替换成需要修改的数据，然后修改以此主键为参照的所有外键，再删除该主键。



图 3-6 before触发器流程

## 物理设计

本次在使用Mysql的关系数据库中，由于成绩、课程、项目具有频繁的检索，所以为了提高存取速度，为他们进行以下物理设计。

### 索引存取

* s\_numbet：学生学号索引
* t\_number：教师编号索引
* coursr\_number：课程编号索引
* classroom\_number：课室编号索引

### 聚簇存取

为提高以班级单位的成绩、课表存取速度，建立聚簇索引。

* class\_grade：班级成绩索引，将同一班级的学生学号、姓名、成绩组成一 个聚簇索引。
* class\_course：班级课表索引，将同一班级的学生学号、姓名、课程编号组成一个聚簇索引。

## 安全性设计

为了确保不同用户对不同数据操作的正确和保护数据的安全。对于不同的用户操作数据库时创建不同的用户权限，其中设计如下：

用户表：数据库保存所有用户的身份信息，并对学生用户授予学生权限、教师用户授予教师权限。

学生权限：学生身份连接数据库后获得学生权限，仅可见且操作学生相关的数据，包括学生课表等。

教师权限：教师身份连接数据库后获得教师权限，仅可见且操作教师相关的数据，包括教师课表等。

# 应用系统设计及实现

本次系统设计分为两部分：学生信息管理系统学生客户端和教师客户端设计。由于两个部分较为相似，所以一并介绍。

## 注册及登入

用户如果没有账号，则需要先进行注册，否则登入不进系统。用户注册后，系统会将用户账号密码、账号信息、账号身份存入云端数据库。

当用户登入系统，服务器将用户输入的账号密码上传至云端数据库进行匹配，倘若匹配成功则显示登入成功，反之登入失败，其中失败分为账号不存在、密码错误两类。

## 学生端功能

### 选课报名

选课报名部分主要分为补重修报名和选修报名两部分。学生选择对应功能后，进入相应的补重修界面或选修界面。各个显示界面显示从数据库拉取的数据，并根据用户的操作更新数据库中学生相应的表中数据。

以下是补重修报名逻辑：



图 4-1 选课报名逻辑图

### 班级课表

班级课表用于显示整个班级的学生当前学期修读的全部课程。其中不同的同学可以选择查看自己的课表。自己的课表中仅出现自己选修的课程。其中课表中不仅仅包含上课的课程，还包括上课的时间地点、上课教师信息。

以下是班级课表逻辑：



图 4-2 班级课表逻辑

### 创新项目

创新项目分为项目项目申请和项目修改两部分，学生通过项目申请部分填入基本的项目申请信息，等待老师审批。项目修改部分用于更新开展的项目的最新信息。

以下是创新项目的逻辑：



图 4-3 创新项目逻辑

### 课程成绩

课程成绩查询可用于查询学生本人所修读的所有课程成绩，其中成绩分为平时成绩和考试成绩。 并且该系统具有绩点计算功能、成绩表导出功能，便于学生用于申请各类奖学金。

以下是课程成绩的逻辑：



图 4-4 成绩导出逻辑

### 考级功能

考级成绩分为四六级、计算机等级等各类证书考试成绩查询，并且考级成绩功能还包括各类考级的申请，学生进行考级申请，系统会自动拉取学生填写的基本信息，用于报名，大大减轻工作量。

以下是考级功能的逻辑：



图 4-5 考级功能逻辑

## 教师端功能

### 课表功能

课表功能包括课表查询及课表安排功能。教师可查询自己所教课程的班级、所教课程的上课时间及地点。系统会为教师进行自动排课，减轻教师的排课工作量。当然教师也可以自行进行上课课室的申请。

以下是具体的课表功能逻辑：



图 4-6 课表功能逻辑

### 教室申请

以往教室申请多采用纸质申请方式，教师难以清楚所有教室的使用情况，增设教室申请功能，可以进一步便于管理所有的教室使用资源，并且进一步减轻教师对于教室申请的工作量，大大缩短时间。

教室申请逻辑如下：



图 4-7 教室申请逻辑

### 项目管理

项目管理分为申请项目和修改项目两部分功能。教师对于学生申请的项目进行审核，选择优秀的项目进行开展。对于正在进行的项目，分为前、中、终三个阶段进行阶段化审核和修改。

项目管理逻辑如下：



图 4-8 项目管理逻辑

### 课程管理

课程管理包括成绩管理。教师对于所教学的每一门课程进行成绩的成批化输入，系统自动结算每一位学生的成绩。

课程管理逻辑如下：



图 4-9 课程管理逻辑

# 测试

以下对所实现的学生信息数据库管理系统进行测试截图。

## 数据库系统



图 5-1 数据库基本状态

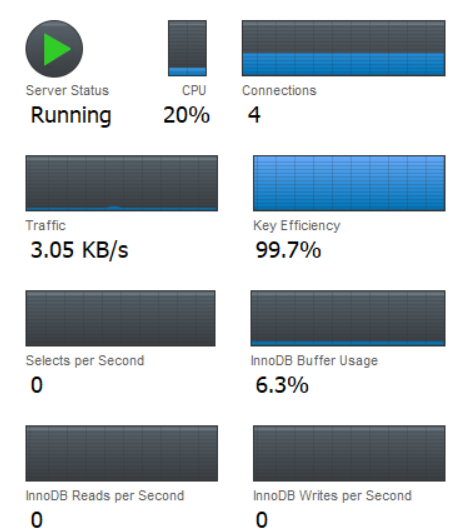


图 5-2 数据库运行信息

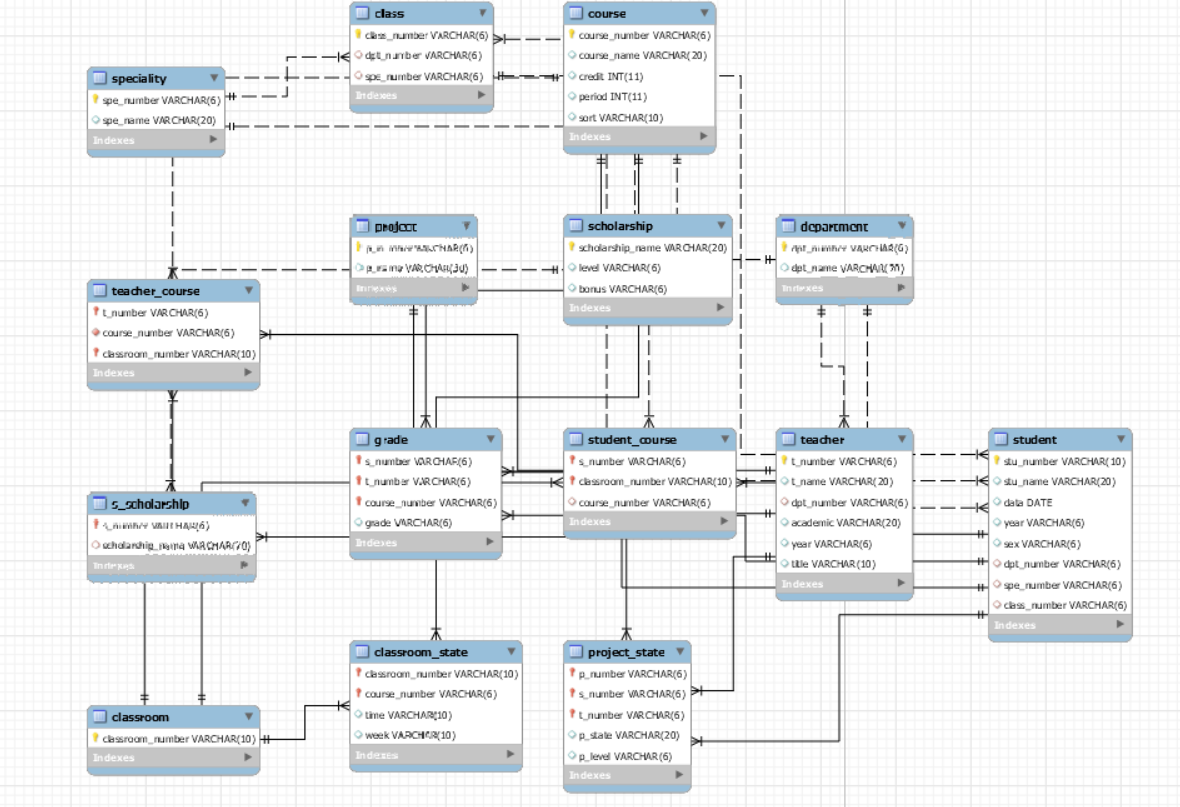


图 5-3 数据库中的表

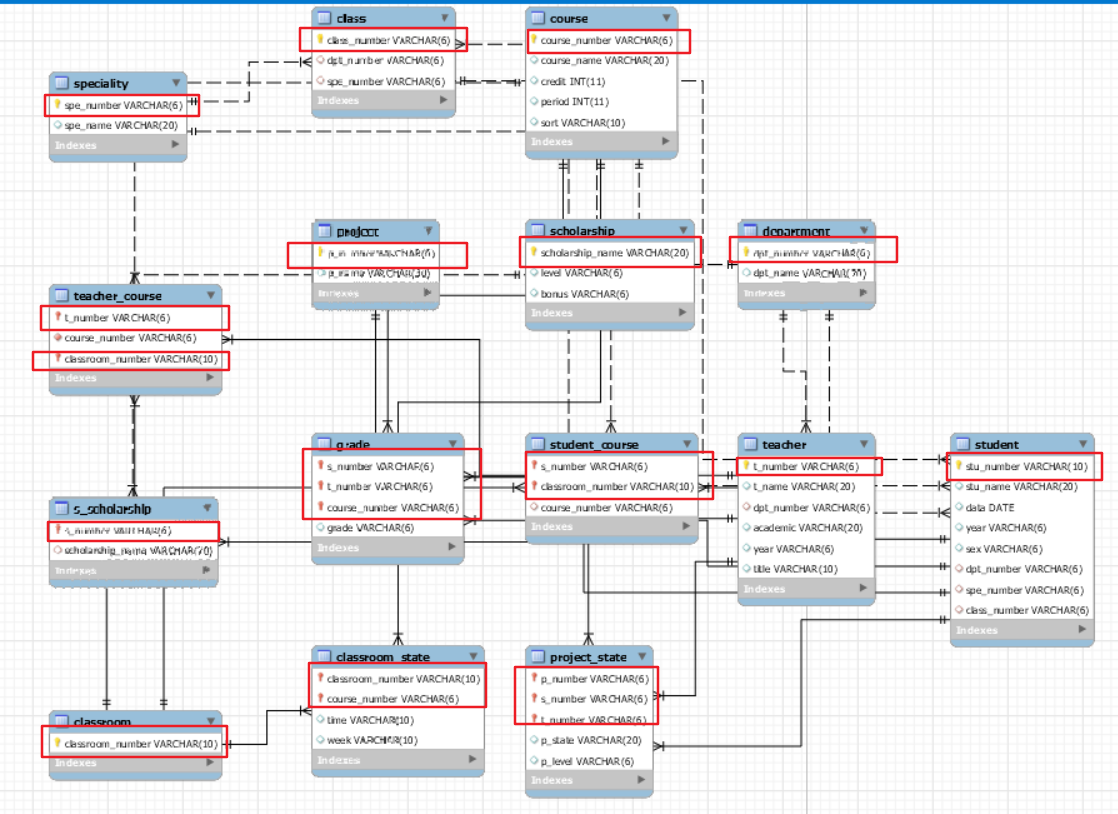


图 5-4 主键约束

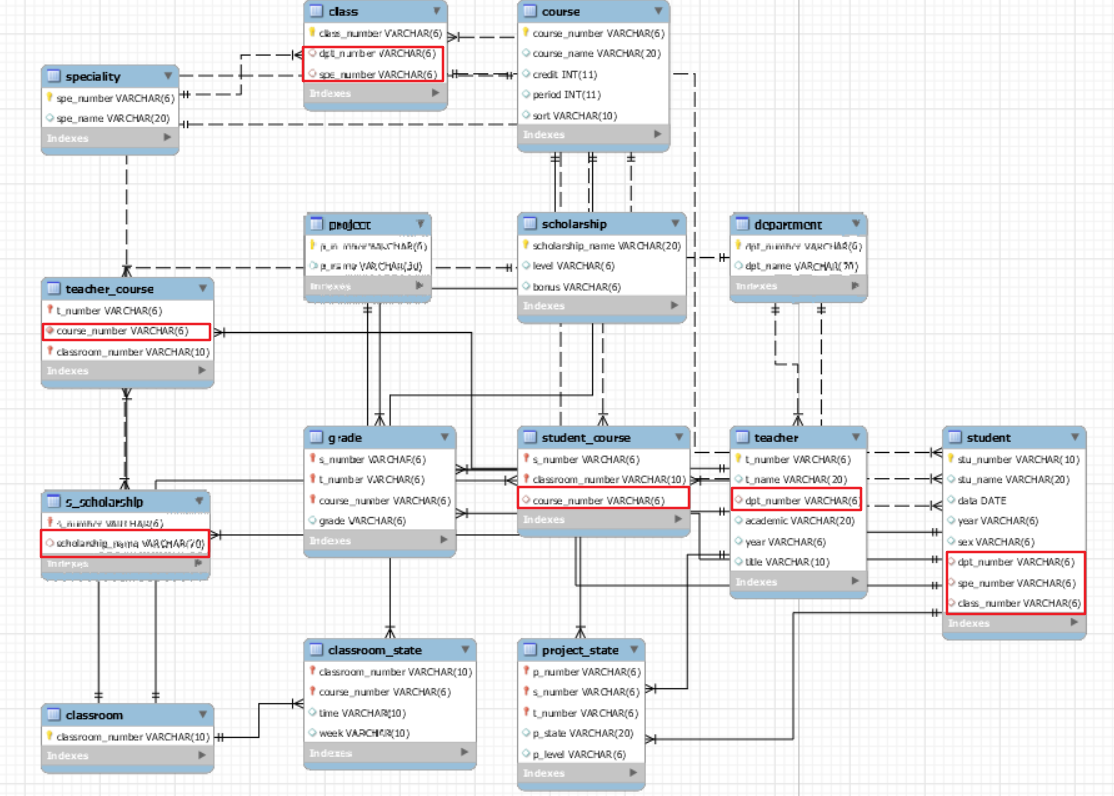


图 5-5 外键约束

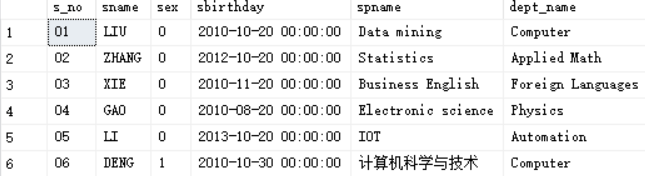


图 5-6 学生元祖

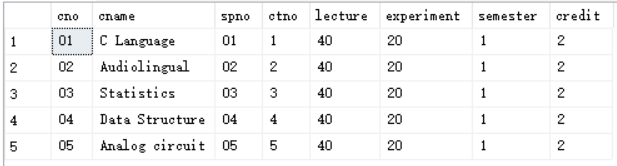


图 5-7 课程元组

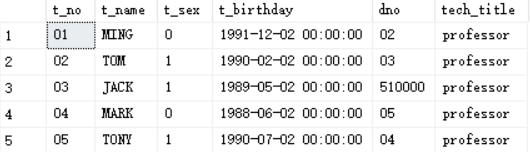


图 5-8 教师元组

## 应用系统

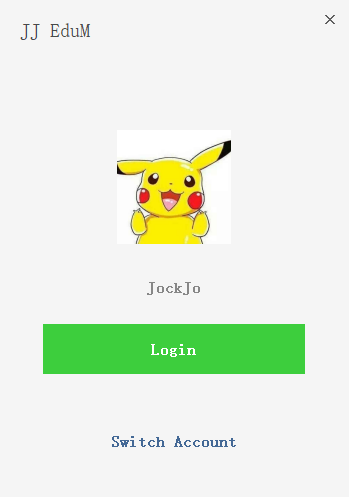


图 5-9 登陆界面



图 5-10 教师界面



图 5-11 教师功能



图 5-12 学生界面



图 5-13 学生功能



图5-14 选课功能

# 关键代码

## 建表代码示例

1. -----------------------------------------------------
2. Table `educatonalmanagement`.`department`
3. -----------------------------------------------------
4. CREATE TABLE IF NOT EXISTS `educatonalmanagement`.`department` (
5. `dpt\_number` VARCHAR(6) NOT NULL,
6. `dpt\_name` VARCHAR(20) NULL DEFAULT NULL,
7. PRIMARY KEY (`dpt\_number`),
8. UNIQUE INDEX `dpt\_number\_UNIQUE` (`dpt\_number` ASC),
9. UNIQUE INDEX `dpt\_name\_UNIQUE` (`dpt\_name` ASC))
10. ENGINE = InnoDB
11. DEFAULT CHARACTER SET = utf8
12. COMMENT = 'department';

## 建立完整性约束示例

1. CONSTRAINT `dpt\_number\_cfk`
2. FOREIGN KEY (`dpt\_number`)
3. REFERENCES `educatonalmanagement`.`department` (`dpt\_number`)
4. ON DELETE NO ACTION
5. ON UPDATE NO ACTION,
6. CONSTRAINT `spe\_number\_cfk`
7. FOREIGN KEY (`spe\_number`)
8. REFERENCES `educatonalmanagement`.`speciality` (`spe\_number`)
9. ON DELETE NO ACTION
10. ON UPDATE NO ACTION)

## 建立检索示例

1. UNIQUE INDEX `class\_number\_UNIQUE` (`class\_number` ASC),
2. INDEX `dep\_number\_idx` (`dpt\_number` ASC),
3. INDEX `spe\_number\_idx` (`spe\_number` ASC),

## 建立视图示例

1. CREATE VIEW ies\_student\_view
2. AS
3. SELECT s\_no,sname,sex,sbirthday,spname
4. FROM student\_3115005141,department\_3115005141,speciality\_3115005141
5. where dept\_name = 'Computer' and student\_3115005141.spno = speciality\_3115005141.spno

## 建立带参存储过程示例

1. if(object\_id('list\_student\_department','P') is not null)
2. drop proc list\_student\_department
3. go
4. create proc list\_student\_department(@dno char(6))
5. as
6. select s\_no,sname,sex,sbirthday,student\_3115005141.dno,spno,class\_no
7. from student\_3115005141
8. where student\_3115005141.dno = @dno ;

## 嵌套查询示例

1. select distinct sname
2. from student\_3115005141,student\_course\_3115005141
3. where student\_3115005141.s\_no not in(select s\_no
4. from student\_course\_3115005141
5. where score < 81) and
6. student\_3115005141.s\_no = student\_course\_3115005141.s\_no

## 简单查询示例

1. select s\_no,sname,sex,convert(varchar(10),sbirthday,121) as sbirthday
2. from student\_3115005141 ;

## 多表连接查询示例

1. select student\_3115005141.s\_no,sname,cname
2. from student\_3115005141,student\_course\_3115005141,Course\_3115005141,teacher\_course\_3115005141
3. where score < 60 and
4. student\_course\_3115005141.s\_no = student\_3115005141.s\_no and
5. teacher\_course\_3115005141.tcid = student\_course\_3115005141.tcid and
6. teacher\_course\_3115005141.cno = Course\_3115005141.cno;

## 删除的before触发器示例

1. create before trigger class\_no\_class
2. on class\_3115005141
3. instead of delete
4. as
5. declare @class\_no char(4);
6. select @class\_no=class\_no from deleted;
7. delete teacher\_course\_3115005141 where class\_no = @class\_no;
8. delete class\_3115005141 where class\_no = @class\_no;

# 个人感想

由于之前自身学习的是嵌入式路线，更侧重于底层开发，接触更多的是面向过程的编码思想。本次尝试进行前后端软件业务开发，着实有点不懂。

世上无难事只怕有心人，只要功夫深，铁杵磨成针。所以经过前期的构思、翻阅书籍，与人交流，渐渐明白前后端开发的思路。明白了前后端的分开设计模式，减轻代码的冗余性，更有利于项目的维护。

通过本次对数据库的设计和应用系统的设计，着实其中还存在许许多多的不足，但是也让我明白对于一个成熟的业务，一个稳健安全的数据库是必不可少的。

其次唯有良好的前后端交互才能给用户带来更好的交互性。

那么通过这次课设，也让我发现自身还有许许多多的不足，剩下的大学时光不多，应当继续夯实基础，丰富自我，技多不压身，多多拓展自身的能力。其次也要技艺精湛，加油！

# 参考书籍

1. SQL入门经典，人民邮电出版社
2. Qt+Creator快速入门
3. 数据库系统概论，高等教育出版社
4. Qt开源社区<http://www.qter.org>
5. CSS参考手册
6. SQL教程 http://www.w3school.com.cn

# 附录

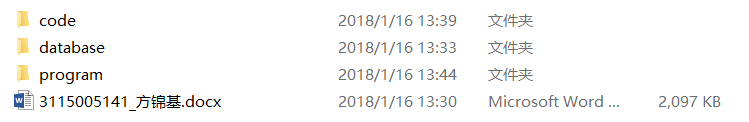


图 9-1 所有文件

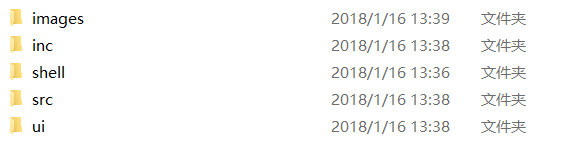


图 9-2 源代码、数据库脚本等



图 9-3数据库日志等

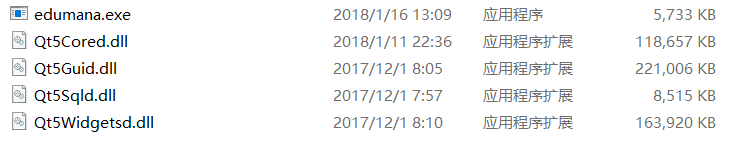


图 9-4 可执行文件