

# 目录

1、需求分析 .....	1
1.1 数据需求描述.....	1
1.2 系统功能需求.....	10
1.3 其他性能需求.....	10
1.4 其他需求.....	10
2、概念结构设计 .....	11
2.1 局部 E-R 图.....	11
2.2 全局 E-R 图.....	13
2.3 优化 E-R 图 .....	13
3、逻辑结构设计 .....	16
3.1 关系模式设计.....	16
3.2 数据类型定义.....	16
3.3 关系模式的优化.....	17
4、物理结构设计 .....	18
4.1 聚簇设计.....	18
4.2 索引设计.....	18
4.3 分区设计.....	18
5、数据库实施 .....	19
5.1 基本表的建立.....	19
5.2 视图的建立.....	24
5.3 索引的建立.....	29
5.4 触发器建立.....	30
5.5 建存储过程.....	32
5.6 数据加载.....	33
6、应用系统开发与试运行 .....	36
6.1 开发平台和开发环境介绍.....	36
6.2 前台界面与后台数据库连接说明、代码实现.....	36
6.3 系统各功能设计和运行界面截图.....	36
6.4 数据库的备份.....	47
7、实验总结 .....	48
7.1 遇到的问题和解决的办法.....	48
7.2 系统设计的不足.....	49
7.3 进一步改进思路和体会.....	49

# 高校成绩管理数据库系统的设计与实现

## 1、需求分析

数据库技术是计算机数据管理的最新技术，是计算机科学的重要分支。今天，越来越多的新应用领域都采用数据库来存储和处理它们的信息资源。高校成绩管理数据库系统记录了高校师生、课程的基本信息以及学生的成绩情况，它的出现使得这些信息的查询、更新更加简单、高效。相比手工管理，高校成绩管理数据库系统具有信息存储更新及时、查询方便快捷、可靠性强、存储量大、安全性好等优点，可大大减少高校成绩管理的时间、人力、金钱成本，是高校成绩管理信息化的必由之路。

本系统以 SQL SERVER 为平台，通过 Python 设计可视化操作界面，适合在装有 win10 系统的个人电脑上运行。操作和维护人员应懂得 SQL 语言。

通过分析基本数据和业务处理需求，我们认为基于现有的数据库及编程知识，能够实现该系统，满足技术可行性；该系统开发完成后将为高校成绩管理带来较大的便利，节省大量的时间、人力、金钱成本，满足经济可行性；系统可视化的操作方式便于工作人员管理成绩，利用简单易懂的 SQL 和 Python 进行实现也便于其进行维护、调试和扩展，满足操作可行性。

具体的需求分析如下。

### 1.1 数据需求描述

#### 数据流图

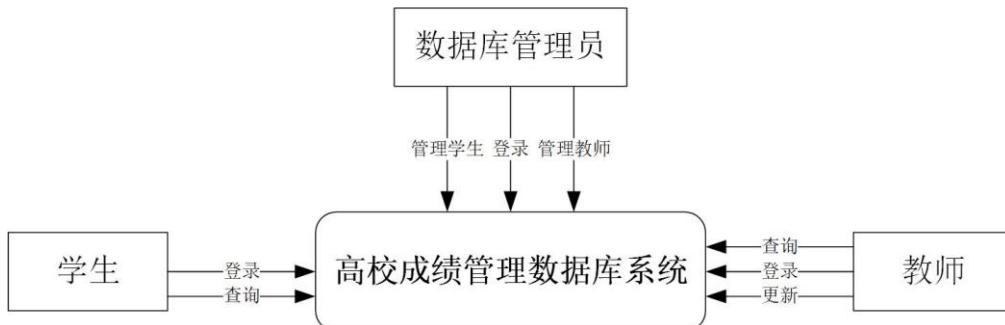


图 1-1 顶层数据流图

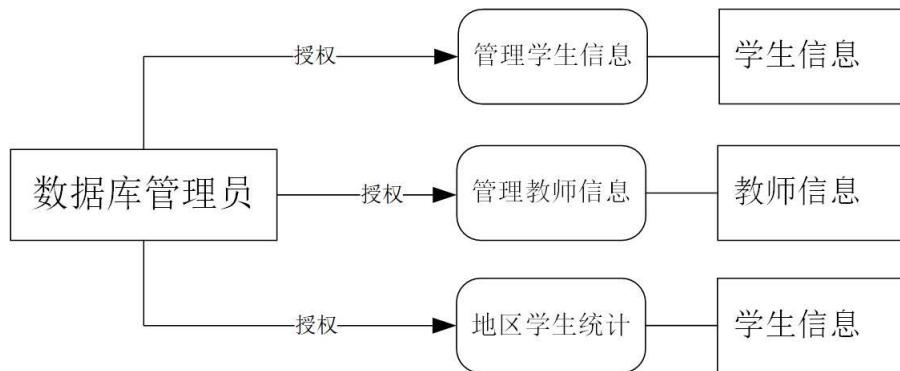


图 1-2 管理员界面数据流图

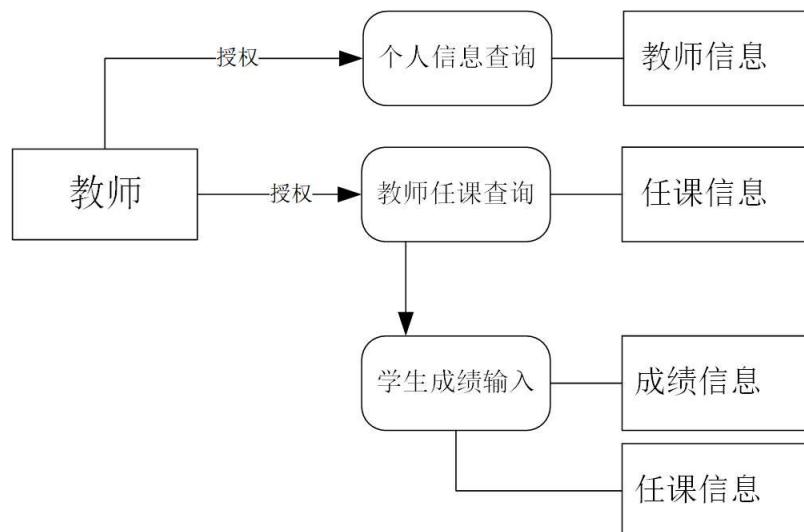


图 1-3 教师界面数据流图

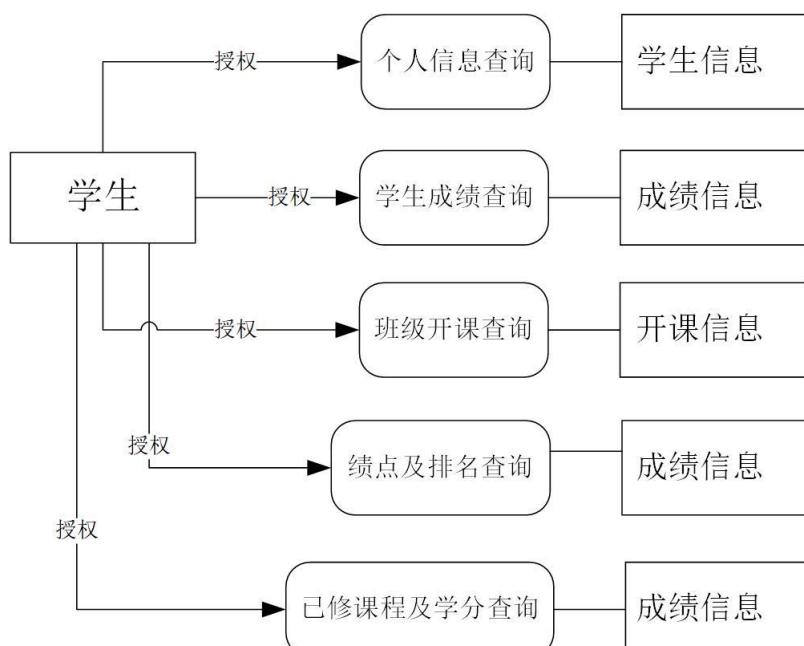


图 1-4 学生界面数据流图

## 数据字典

### ➤ 数据项卡片

数据项名:	学号
含义说明:	唯一标识每个学生
数据类型:	CHAR
长 度:	12
取值范围:	000000000000 至 999999999999
取值含义:	前 4 位标识入学年级, 中 4 位标识专业, 后 4 位标识班级和学号

数据项名:	学生姓名
含义说明:	标识每个学生的姓名

数据类型:	CHAR
长 度:	20

数据项名:	学生性别
含义说明:	表示每个学生的性别
数据类型:	CHAR
长 度:	2
取值范围:	“男”或者“女”
取值含义:	只有男或女两种选择

数据项名:	学生年龄
含义说明:	表示每个学生的年龄
数据类型:	INT
长 度:	4
取值范围:	0 至 99
取值含义:	不存在年龄小于 0 或年龄大于 99 的学生

数据项名:	生源所在省
含义说明:	表示每个学生入学前的来源地（省）
别 名:	籍贯（省）
数据类型:	CHAR
长 度:	20

数据项名:	生源所在市
含义说明:	表示每个学生入学前的来源地（市）
别 名:	籍贯（市）
数据类型:	CHAR
长 度:	20

数据项名: 已修学分总数  
含义说明: 学生已修课程的学分总数  
数据类型: INT  
长 度: 4  
取值范围: >0  
取值含义: 学分都是正数  
与其他数据项的逻辑关系: 等于该学生已修的所有课程的“学分”数据项之和

数据项名: 专业编号  
含义说明: 唯一标识每个专业  
数据类型: CHAR  
长 度: 4  
取值范围: 0000 至 9999

数据项名: 专业名称  
含义说明: 表示每个专业的名称  
数据类型: CHAR  
长 度: 20

数据项名: 班级编号  
含义说明: 唯一标识每个专业中的每个班级  
数据类型: CHAR  
长 度: 4  
取值范围: 0000 至 9999  
取值含义: 前 2 位标识年级, 后 2 位标识班级

数据项名: 教师编号  
含义说明: 唯一标识每个教师  
数据类型: CHAR  
长 度: 12  
取值范围: 000000000000 至 999999999999  
取值含义: 顺序编号

数据项名: 教师姓名  
含义说明: 表示每个教师的姓名  
数据类型: CHAR  
长 度: 20

数据项名: 教师性别  
含义说明: 表示每个教师的性别  
数据类型: CHAR  
长 度: 2  
取值范围: “男”或者“女”  
取值含义: 只有男或女两种选择

数据项名: 教师年龄  
含义说明: 表示每个教师的年龄  
数据类型: INT  
长 度: 4  
取值范围: 0 至 99  
取值含义: 不存在年龄小于 0 或年龄大于 99 的教师

数据项名: 职称  
含义说明: 表示每个教师的职称等级  
数据类型: CHAR  
长 度: 8  
取值范围: “助教”、“讲师”、“副教授”、“教授”  
取值含义: 只有以上四种选择

数据项名: 联系电话  
含义说明: 表示每个教师的联系电话  
数据类型: CHAR  
长 度: 12  
取值范围: 0000000000 至 9999999999

数据项名: 课程编号  
含义说明: 唯一标识每个课程  
数据类型: CHAR  
长 度: 12  
取值范围: 000000000000 至 999999999999  
取值含义: 顺序编号

数据项名: 课程名称  
含义说明: 表示每个课程的名称  
数据类型: CHAR  
长 度: 20

数据项名: 开课学年  
含义说明: 表示每个课程开设的学年  
数据类型: CHAR  
长 度: 5

数据项名: 开课学期  
含义说明: 表示每个课程开设的学期  
数据类型: CHAR  
长 度: 5  
取值范围: 1 或 2

数据项名: 学分  
含义说明: 表示每门课程的学分数  
数据类型: INT  
长 度: 4  
取值范围: >0  
取值含义: 取值越大计算绩点时权重越大

数据项名: 学时  
含义说明: 表示每门课程的学时数  
数据类型: INT  
长 度: 4  
取值范围: >0, 一般取 16 的倍数  
取值含义: 教学周一般是 16 周

数据项名: 考试或考查  
含义说明: 表示课程的考核方式  
别 名: 考核方式  
数据类型: CHAR  
长 度: 4  
取值范围: “考试”或“考查  
取值含义: 只有以上两种选择

数据项名: 成绩  
含义说明: 表示每个学生每门课程的成绩  
数据类型: INT  
长 度: 4  
取值范围: 0 至 100  
取值含义: 越大越好, 一般 100 满分, 60 以下不及格

### ➤ 数据结构卡片

数据结构名:	学生
含义说明:	定义了一个学生的有关信息
组 成:	学号, 学生姓名, 学生性别, 学生年龄, 生源所在地, 已修学分总数

数据结构名:	班级
含义说明:	定义了一个班级的有关信息
组 成:	班级编号

数据结构名:	专业
含义说明:	定义了一个专业的有关信息
组 成:	专业编号, 专业名称

数据结构名:	课程
含义说明:	定义了一个课程的有关信息
组 成:	课程编号, 课程名称, 开课学期, 学分, 学时, 考试或考查

数据结构名:	教师
含义说明:	定义了一个教师的有关信息
组 成:	教师编号, 教师姓名, 教师性别, 职称, 教师年龄, 联系电话

### ➤ 数据流卡片

数据流名:	输入登录信息
含义说明:	用户登录系统时需要输入的信息
数据流来源:	用户
数据流去向:	用户账户表
组 成:	用户账号, 用户密码, 用户身份

数据流名:	查询、增加、修改、删除学生信息
含义说明:	用户对学生信息进行操作时需要输入的信息
数据流来源:	用户
数据流去向:	学生信息表
组 成:	学号, 学生姓名, 学生性别, 学生年龄, 生源所在地, 班级, 专业

数据流名:	查询、增加、修改学生成绩信息
含义说明:	用户对学生选课成绩信息进行操作时需要输入的信息
数据流来源:	用户
数据流去向:	学生成绩表
组 成:	学号, 学期, 课程名称, 成绩, 任课老师

数据流名:	查询、增加、修改、删除教师信息
含义说明:	用户对教师信息进行操作时需要输入的信息
数据流来源:	用户
数据流去向:	教师信息表
组 成:	教师编号, 教师姓名, 教师性别, 教师年龄, 职称, 联系电话

数据流名:	查询、增加、修改、删除开课信息
含义说明:	用户对开课信息进行操作时需要输入的信息
数据流来源:	用户
数据流去向:	开课信息表
组 成:	班级, 课程编号, 教师编号, 学期

## ➤ 数据存储

数据存储名:	用户账号信息
说 明:	记录用户账号信息
输入数据流:	管理员输入账号信息操作以及相关信息
输出数据流:	用户账号表
组 成:	用户账号, 用户密码, 用户身份
存取方式:	随机存取

数据存储名:	学生信息
说 明:	记录学生基本信息
输入数据流:	管理员输入学生信息更新操作以及相关信息
输出数据流:	学生信息表
组 成:	学号, 姓名, 性别, 年龄, 生源所在地, 已修学分总数, 班级
存取方式:	随机存取

数据存储:	课程信息
说 明:	记录课程基本信息
输入数据流:	管理员输入课程信息更新操作以及相关信息
输出数据流:	课程信息表
组 成:	课程编号, 课程名称, 学时, 考核方式, 学分, 开设学期
存取方式:	随机存取

数据存储名:	教师信息
说 明:	记录教师基本信息
输入数据流:	管理员输入教师信息更新操作以及相关信息
输出数据流:	教师信息表
组 成:	教师编号, 姓名, 性别, 年龄, 职称, 联系电话
存取方式:	随机存取

数据存储名: 教师任课信息  
说 明: 记录课程基本信息  
输入数据流: 管理员输入任课信息更新操作以及相关信息  
输出数据流: 课程开设信息表  
组 成: 班级, 课程编号, 教师编号, 学期  
存取方式: 随机存取

数据存储名: 教师任课信息  
说 明: 记录课程基本信息  
输入数据流: 管理员输入任课信息更新操作以及相关信息  
输出数据流: 课程开设信息表  
组 成: 班级, 课程编号, 教师编号, 学期  
存取方式: 随机存取

数据存储名: 学生成绩信息  
说 明: 记录学生成绩信息  
输入数据流: 教师输入成绩信息更新操作以及相关信息  
输出数据流: 成绩信息表  
组 成: 学号, 课程编号, 教师编号, 学号, 成绩, 开设学期  
存取方式: 随机存取

### ➤ 处理过程

处理过程: 登录  
说 明: 核对用户信息  
输入: 用户账户, 用户密码, 用户身份  
输出: 登录反馈信息  
处 理: 接受用户登录时的输入信息将其与用户信息表中的信息核对, 若存在相同记录项, 则用户登录成功, 否则反馈用户登录失败

处理过程: 权限检查  
说 明: 检查操作用户权限  
输入: 用户身份, 用户操作  
输出: 反馈权限信息  
处 理: 用户进行某一操作时会自动进行该用户身份对该操作的权限检查, 如果该用户身份

处理过程: 用户操作  
说 明: 将用户在平台上的操作与后台数据库连接  
输入: 用户操作信息  
输出: 用户操作结果  
处 理: 用户选择某一操作并将该操作相关的信息输入到系统, 系统在后台转化为数据库处

## 1.2 系统功能需求

各个系统功能模块的功能需求如下。

### 1. 管理员（教务处）

- a) 添加教师名单；
- b) 删除教师名单；
- c) 查询教师信息；
- d) 修改教师信息；
- e) 添加学生名单；
- f) 删除学生名单；
- g) 查询学生信息；
- h) 修改学生信息；
- i) 统计生源地信息。

### 2. 教师

- a) 查看个人信息；
- b) 查看任课信息；
- c) 查看课程学生成绩；
- d) 查看课程平均成绩；
- e) 修改学生成绩。

### 3. 学生

- a) 查看个人信息；
- b) 查看学生课程成绩；
- c) 查看班级开课情况；
- d) 查看个人绩点及排名；
- e) 查看已修课程及学分。

## 1.3 其他性能需求

1. 当出现不规范输入或错误输入时会有错误提示；
2. 用户需要账号密码才能登录系统；
3. 不同用户可使用的功能不同；
4. 创建触发器，保证数据的一致性；
5. 创建存储过程，方便某些功能的实现。

其他性能分析，如并发用户数、响应时间和存储需求描述等。

## 1.4 其他需求

1. 学校设置了各专业，在专业下开设班级，每个班级包含若干学生，学生信息至少需要包含学号、姓名、性别、年龄、生源所在地、已修学分总数等数据项；
2. 课程信息表至少需要包含课程编号、课程名称、任课教师、开课学期、学时、考试或考查、学分等数据项，课程根据班级开设；
3. 教师信息表至少需要包含教师编号、姓名、性别、年龄、职称、联系电话等数据项；
4. 学生成绩至少需要包含学号、学期、课程名称、成绩、任课老师等数据项；
5. 操作界面简洁、美观、人性化、易操作。

## 2、概念结构设计

### 2.1 局部 E-R 图

根据数据库提供的服务种类进行划分，可以确定四个局部 E-R 模式，其范围分别为学生信息查询、学生成绩查询、教师任课查询和班级课程开设查询。局部 E-R 模式如下。

#### 学生信息查询的局部 E-R 图

根据工作人员的实际工作需要，学生信息至少需要包含学号、姓名、性别、年龄、生源所在地、已修学分总数等数据项。考虑到课程根据班级开设，因此为方便选课将专业、班级信息也划入学生信息范围。得到的局部 E-R 模式中，实体集有：

- 专业（专业编号，专业名称）
- 班级（班级编号）
- 学生（学号，姓名，性别，年龄，生源所在地，已修学分总数）

其中，因为学校在各专业下开设若干班级，每个班级包含若干学生，因此实体集“专业”和“班级”间有 1:n 联系“开设”，“班级”和“学生”间有 1:n 联系“包含”。学生信息查询的局部 E-R 图如下。

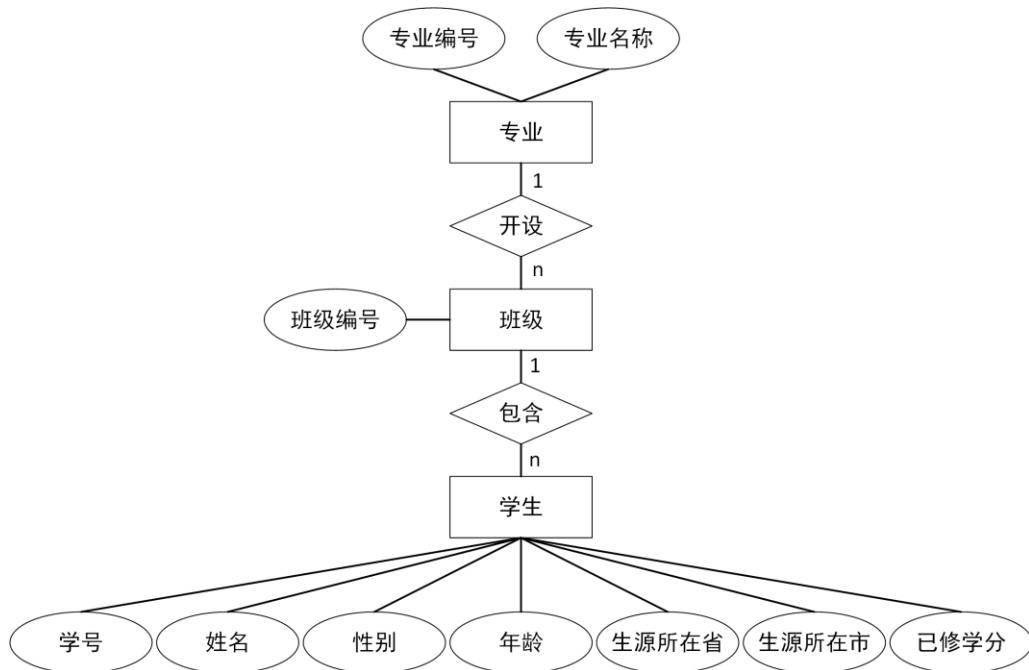


图 2-1 学生信息查询的局部 E-R 图

#### 学生成绩查询的局部 E-R 图

根据需求，课程信息至少需要包含课程编号、课程名称、任课教师、开课学期、学时、考试或考查、学分等数据项。得到的局部 E-R 模式中，实体集有：

- 学生（学号，姓名，性别，年龄，生源所在地，已修学分总数）
- 教师（教师编号，姓名，性别，年龄，职称，联系电话）

- 课程（课程编号，课程名称，开课学期，学时，考试或考查，学分）
- 其中，因为一名学生的一门课只有一名教师教授，一名教师的一门课有若干名学生学习，一名学生可能会上一名教师教授的若干门课，故“教师”、“学生”和“课程”间有  $1:m:n$  联系“上课”，并且该联系包含一个属性“成绩”。学生成绩查询的局部 E-R 图如下。

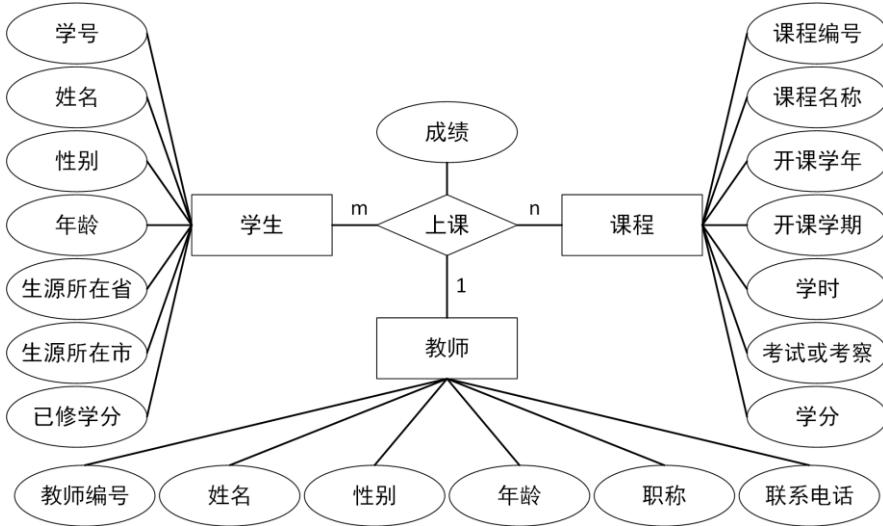


图 2-2 学生成绩查询的局部 E-R 图

### 教师任课查询的局部 E-R 图

根据需求，教师信息至少需要包含教师编号、姓名、性别、年龄、职称、联系电话等数据项。得到的局部 E-R 模式中，实体集有：

- 教师（教师编号，姓名，性别，年龄，职称，联系电话）
- 课程（课程编号，开课学期，教师编号，课程名称，学时，考试或考查，学分）

其中，因为一名教师可能任教若干门课，一门课有若干名教师任教，故“课程”和“教师”间有  $m:n$  联系“任课”。教师任课查询的局部 E-R 图如下。

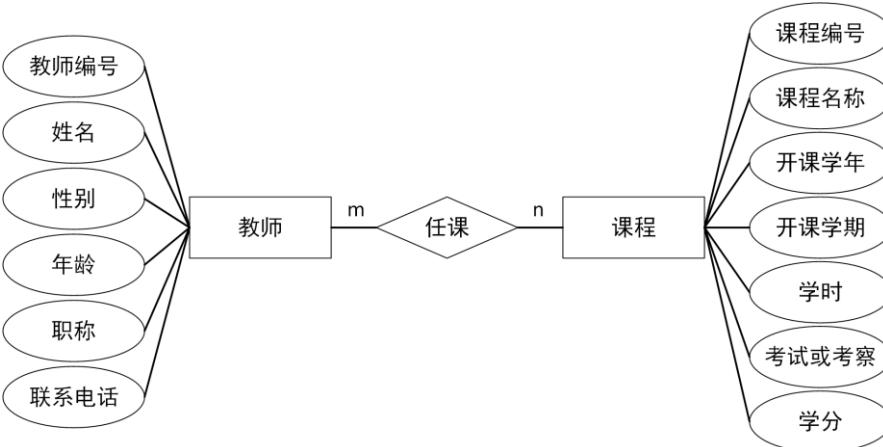


图 2-3 教师任课查询的局部 E-R 图

### 班级课程开设查询的局部 E-R 图

由于课程根据班级开设，因此在班级课程开设查询的局部 E-R 模式中有实体集：

- 专业（专业编号，专业名称）

- 班级（班级编号）
  - 课程（课程编号，开课学期，课程名称，学时，考试或考查，学分）
- 其中，“专业”和“班级”间有  $1:n$  联系“开设”，“教师”、“班级”和“课程”间有  $1:m:n$  联系“开设”。班级课程开设查询的局部 E-R 图如下。

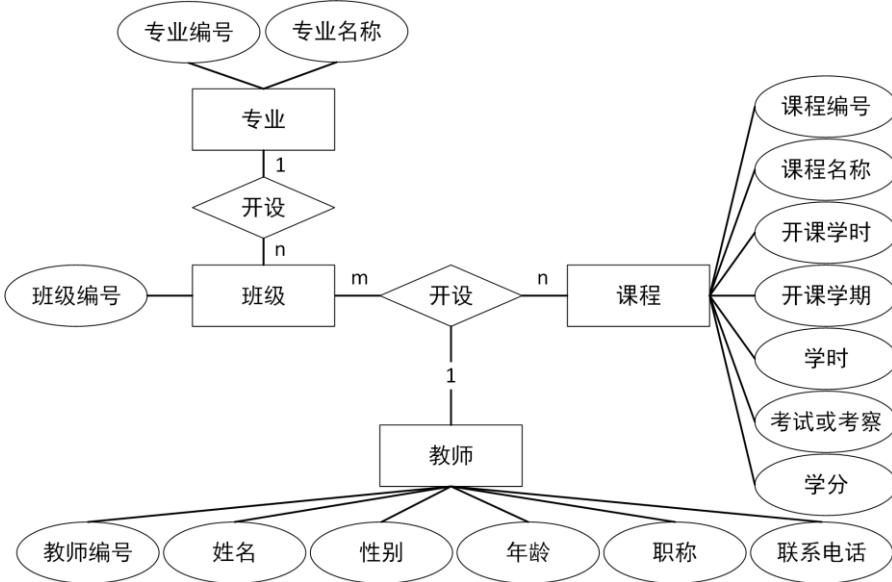


图 2-4 班级课程开设查询的局部 E-R 图

## 2.2 全局 E-R 图

由于“专业”与“班级”之间的联系“开设”和“班级”与“课程”之间的联系“开设”存在命名冲突，因此将“专业”与“班级”之间的联系名改为“属于”。消除冲突后，得到全局 E-R 图如图 2-5。

## 2.3 优化 E-R 图

首先考察实体性是否合并的问题。由于不同专业的班级可能具有同样的班级编号，因此要真正确定一个班级，总需要将“班级”和“专业”两个实体集相连接，这会在处理班级相关信息时造成较大的开销；同时，“班级”实体集中仅包含一个属性，和“专业”合并后不会产生大量的空值，也不会产生较大的存储代价，权衡后决定将“班级”和“专业”两实体集合并成一个实体集，名为“专业班级”。

在用户需求中，发现课程信息表需要包含任课教师数据项，按照现有 E-R 模式，在查询时需要对 3 张基本表进行连接，由于课程信息在实际工作中可能需要经常查询，频繁的连接操作将降低系统的效率，但如果将教师编号和姓名同时加入“课程”的属性，又将造成存储空间的浪费，因此在权衡之后，决定只将教师编号加入“课程”的属性，形成新实体“教师课程”，从而将需要连接的基本表减少到 2 张。由于“课程”实体发生了改变，因此原来的联系也将受到调整，新的实体及其联系图如图 2-6。

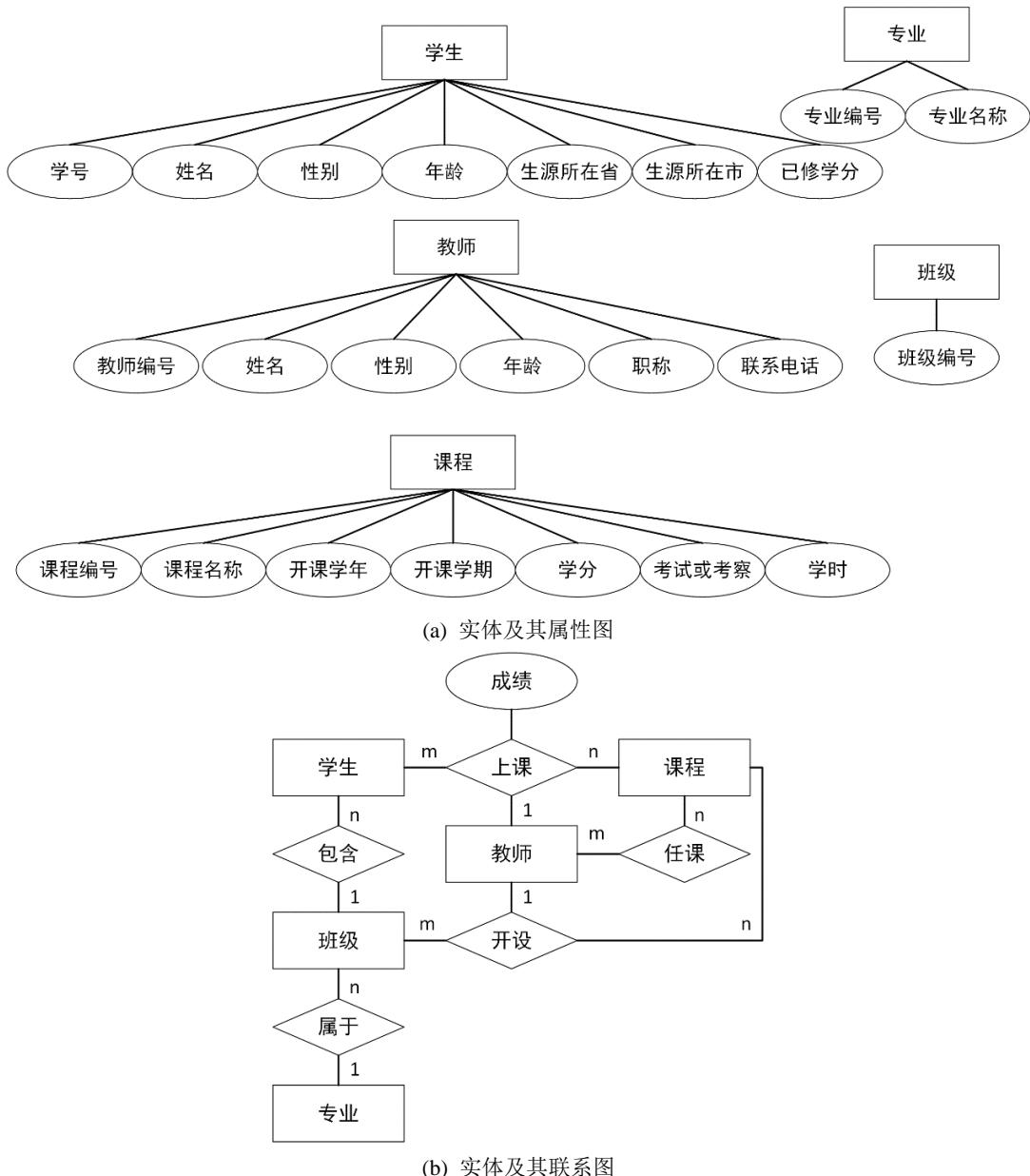


图 2-5 全局 E-R 图 (分离法表示)

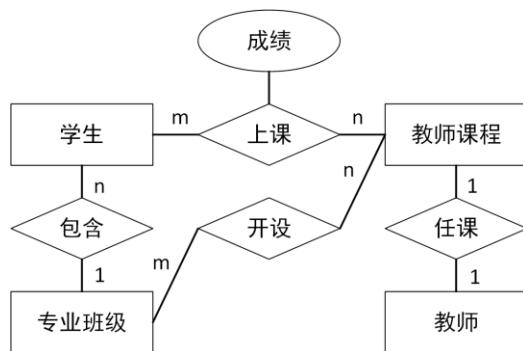
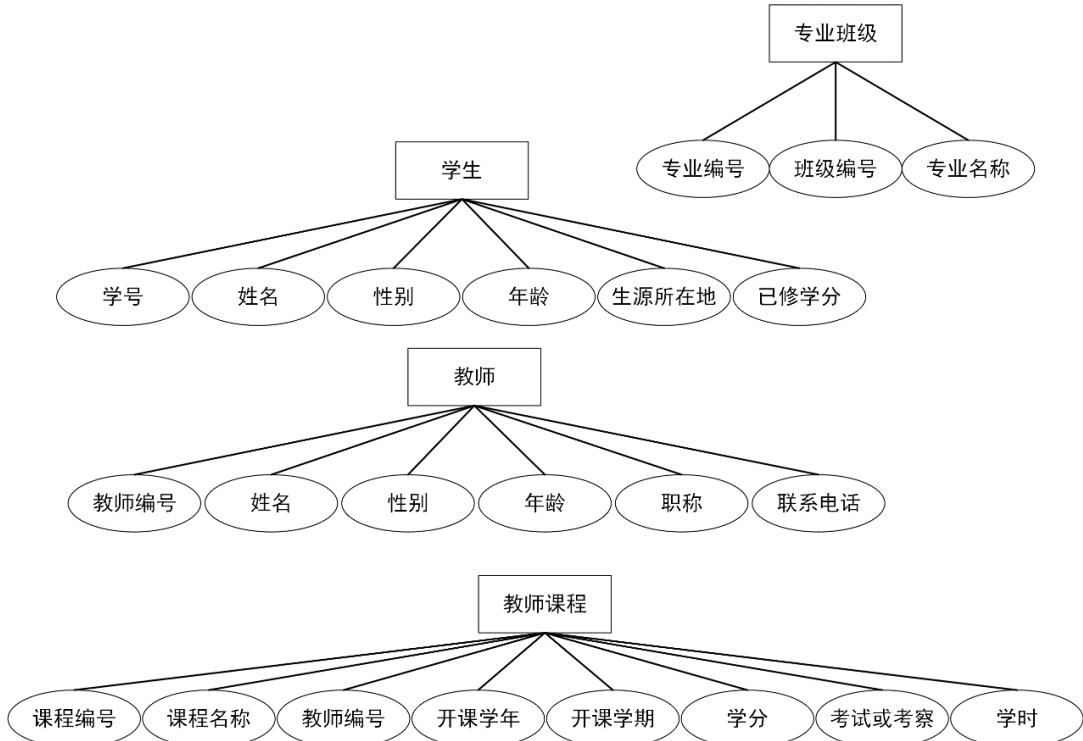
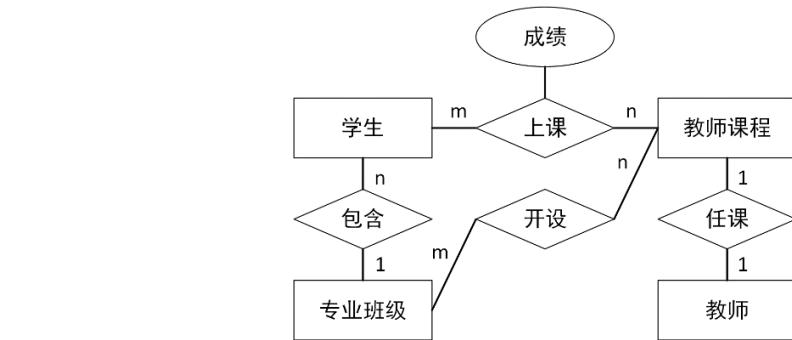


图 2-6 实体及其联系图

由于未发现冗余属性，因此直接考虑冗余联系是否消除的问题。由上图可知，“班级”与“教师课程”间的“开设”联系可以由“学生”和“教师课程”间的“上课”联系、“学生”和“班级”间的“包含”联系推导出来，因此属于冗余属性，可以消除。但是根据需求分析，管理学生成绩的工作人员实际工作中可能需要经常查询班级课程开设情况，为了提高效率，应当允许该冗余联系的存在。优化后的全局 E-R 图如下。



(a) 实体及其属性图



(b) 实体及其联系图

图 2-7 优化全局 E-R 图（分离法表示）

## 3、逻辑结构设计

### 3.1 关系模式设计

Chenhr\_专业班级 01(chr\_专业编号 01, chr\_班级编号 01, chr\_专业名称 01)

Chenhr\_学生 01(chr\_学号 01, chr\_专业编号 01, chr\_班级编号 01, chr\_学生姓名 01,  
chr\_学生性别 01, chr\_学生年龄 01, chr\_生源所在省 01, chr\_生源所在市 01, chr\_已修学分总数 01)

Chenhr\_教师课程 01(chr\_课程编号 01, chr\_开课学年 01, chr\_开课学期 01, chr\_教师编号 01, chr\_课程名称 01, chr\_学分 01, chr\_考试或考查 01, chr\_学时 01)

Chenhr\_教师 01(chr\_教师编号 01, chr\_教师姓名 01, chr\_教师性别 01, chr\_教师年龄 01, chr\_职称 01, chr\_联系电话 01)

Chenhr\_上课 01(chr\_学号 01, chr\_课程编号 01, chr\_教师编号 01, chr\_开课学年 01, chr\_开课学期 01, chr\_成绩 01)

Chenhr\_开课 01(chr\_专业编号 01, chr\_班级编号 01, chr\_课程编号 01, chr\_教师编号 01, chr\_开课学年 01, chr\_开课学期 01)

### 3.2 数据类型定义

表 3-1 Chenhr\_专业班级 01

字段名	数据类型	长度	完整性约束	描述
chr_专业编号 01	CHAR	4	主键	
chr_班级编号 01	CHAR	4	主键	不同专业可能班级编号相同
chr_专业名称 01	CHAR	20		

表 3-2 Chenhr\_学生 01

字段名	数据类型	长度	完整性约束	描述
chr_学号 01	CHAR	12	主键, 唯一	
chr_专业编号 01	CHAR	4	外键	
chr_班级编号 01	CHAR	4	外键	
chr_学生姓名 01	CHAR	20	非空	可能重名
chr_学生性别 01	CHAR	2	“男”或“女”	
chr_学生年龄 01	INT	4	0 至 99	
chr_生源所在省 01	CHAR	20		
chr_生源所在市 01	CHAR	20		
chr_已修学分总数 01	INT	4	大于等于 0	

表 3-3 Chenhr\_教师课程 01

字段名	数据类型	长度	完整性约束	描述
chr_课程编号 01	CHAR	12	主键	
chr_开课学年 01	CHAR	5	主键	同一门课每年都会开

表 3-4 Chenhr\_教师课程 01 (续)

字段名	数据类型	长度	完整性约束	描述
chr_开课学期 01	CHAR	5	主键	
chr_教师编号 01	CHAR	12	主键, 外键	
chr_课程名称 01	CHAR	20		
chr_学分 01	INT	4	大于 0	
chr_考试或考查 01	CHAR	4	“考试”或“考查”	
chr_学时 01	INT	4	大于 0	

表 3-5 Chenhr\_教师 01

字段名	数据类型	长度	完整性约束	描述
chr_教师编号 01	CHAR	12	主键, 唯一, 非空	
chr_教师姓名 01	CHAR	20	非空	可能重名
chr_教师性别 01	CHAR	2	“男”或“女”	
chr_教师年龄 01	INT	4	0 至 99	
chr_职称 01	CHAR	8	“无职称”或“助教”或“副教授”或“教授”	
chr_联系电话 01	CHAR	12		

表 3-6 Chenhr\_上课 01

字段名	数据类型	长度	完整性约束	描述
chr_学号 01	CHAR	12	主键, 外键	
chr_课程编号 01	CHAR	12	主键, 外键	
chr_开课学年 01	CHAR	5	主键, 外键	同一门课每年都会开
chr_开课学期 01	CHAR	5	主键, 外键	考虑重修, 即不同学期上同一门课
chr_教师编号 01	CHAR	12	主键, 外键	
chr_成绩 01	INT	4	0 至 100	

表 3-7 Chenhr\_开课 01

字段名	数据类型	长度	完整性约束	描述
chr_专业编号 01	CHAR	4	主键, 外键	
chr_班级编号 01	CHAR	4	主键, 外键	
chr_课程编号 01	CHAR	12	主键, 外键	
chr_开课学年 01	CHAR	5	主键, 外键	同一门课每年都会开
chr_开课学期 01	CHAR	5	主键, 外键	
chr_教师编号 01	CHAR	12	主键, 外键	

### 3.3 关系模式的优化

考察关系模式 “Chenhr\_教师课程 01”, 可知其中包含函数依赖 “{chr\_课程

编号 01, chr\_开课学期 01, chr\_教师编号 01}→chr\_课程名称 01”、“{chr\_课程编号 01, chr\_开课学期 01}→chr\_课程名称 01”，因此关系模式存在非主属性对候选键的部分函数依赖，是 1NF。同样地，考察关系模式“Chenhr\_专业班级 01”，可知其中包含函数依赖“{chr\_专业编号 01, chr\_班级编号 01}→chr\_专业名称 01”、“chr\_专业编号 01→chr\_专业名称 01”，因此关系模式存在非主属性对候选键的部分函数依赖，是 1NF。然而，根据 2.3 中所述，为提高效率不对其进行分解。其余关系模式均为 3NF，故不做优化。

## 4、物理结构设计

### 4.1 聚簇设计

聚簇是将有关的数据元组集中存放于一个物理块内或若干相邻物理块内或同一柱面内，以提高查询效率的数据存储结构。

由于建立聚簇后对非聚簇属性列的查询效果不如聚簇属性列，且数据库建立和维护聚簇的开销很大，因此仅对以下一些特定情况才考虑建立聚簇：

1. 当对一个关系的某些属性列的访问时该关系的主要应用，而对其他属性的访问很少或是次要应用时，可考虑对该关系在这些属性列上建立聚簇；
2. 若一关系在某些属性列上的值重复率很高，则可考虑对该关系在这些组属性列上建立聚簇；
3. 若一关系一旦装入数据，某些属性列的值很少修改，也很少增加或删除元组，则可考虑对该关系在这些组属性列上建立聚簇。

由上所述，考虑对每个基本表的主键其建立聚簇。

### 4.2 索引设计

对关系选择有效的索引对提高数据库访问效率很有帮助。对于一个确定的关系，通常在下列情况可以考虑建立索引：

1. 在主键属性列和外键属性列上可以分别建立索引，不仅有助于唯一性检查和完整性检查，而且可以加快连接查询的速度；
2. 以查询为主的关系可建立尽可能多的索引；
3. 对等值连接，但满足条件较少的查询可考虑建立索引；
4. 若查询可从索引直接得到结果而不必访问关系，则对此种查询建立索引。

由上述条件，在每个关系模式的主键和外键属性列上，都分别建立索引；此外，由于“Chenhr\_上课 01”中“chr\_成绩 01”属性的 AVG 函数值和“Chenhr\_教师课程 01”中“chr\_学分 01”属性的 SUM 函数值需要被查询，因此对两者也建立索引。

### 4.3 分区设计

磁盘分区设计的本质是确定数据库数据的存放位置，其目的是提高系统性能。其一般原则是：

1. 减少访问冲突，提高 I/O 并行性；
2. 分散热点数据，均衡 I/O 负担；
3. 保证关键数据快速访问，缓解系统“瓶颈”。

具体到实际操作，由于本高校成绩管理数据库系统规模较小，仅需在 PC 上运行，故不进行分区设计。

## 5、数据库实施

### 5.1 基本表的建立

#### 创建数据库

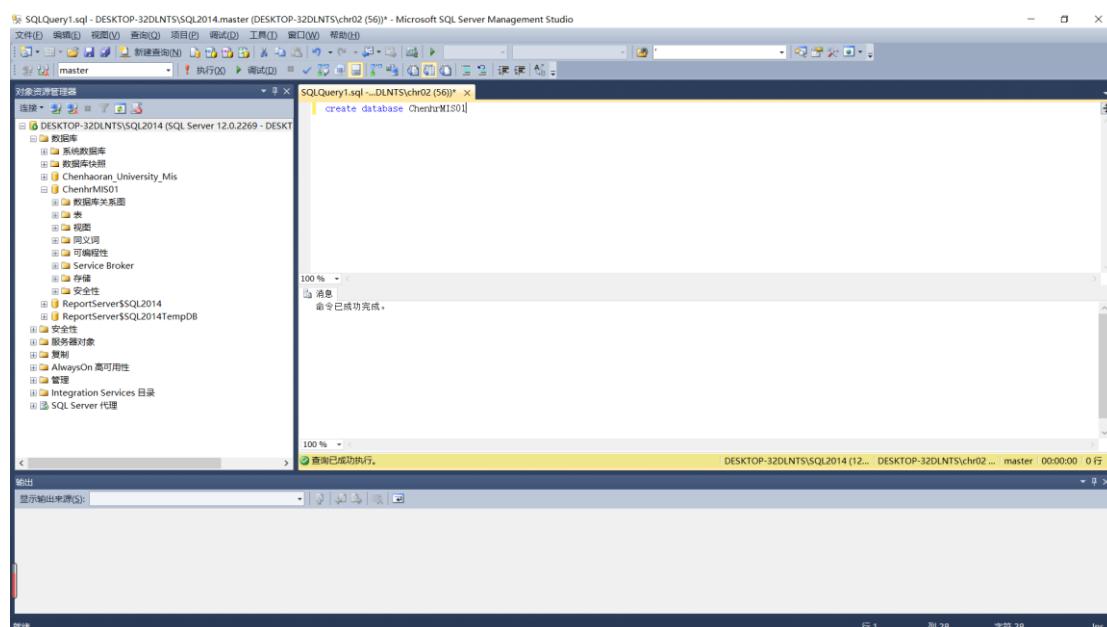


图 5-1 创建数据库

#### 创建专业班级信息表

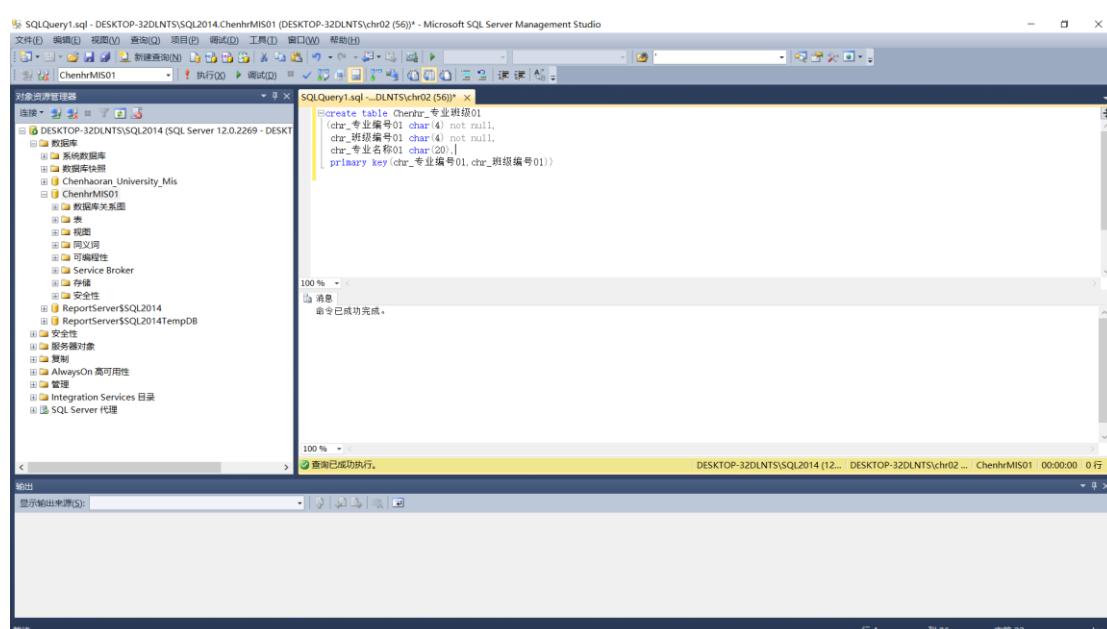
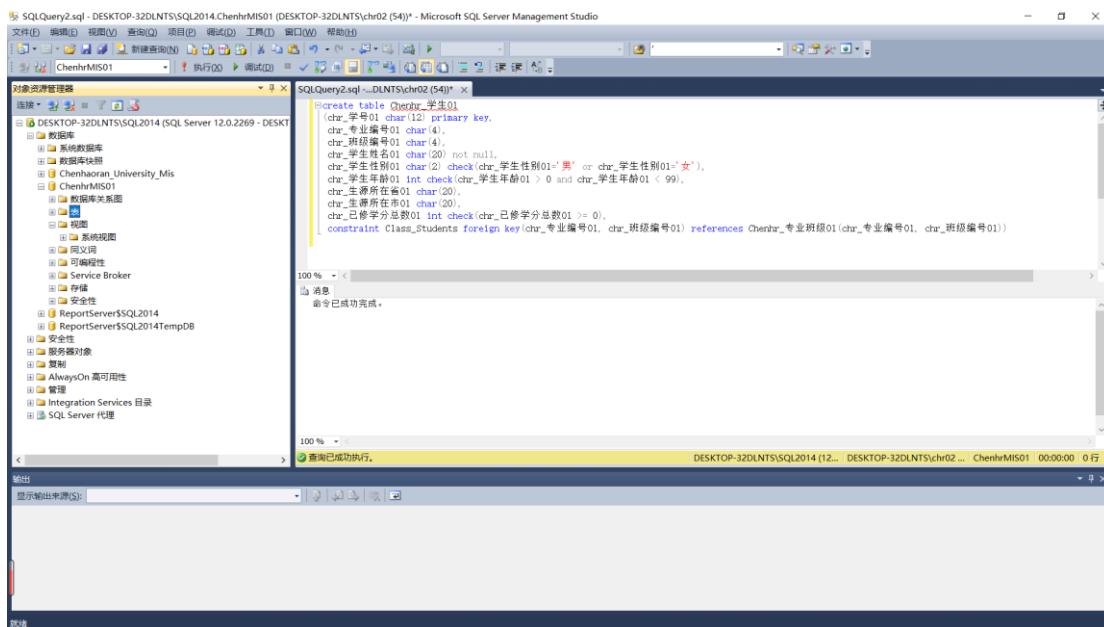


图 5-2 创建专业班级信息表

## 创建学生信息表



```

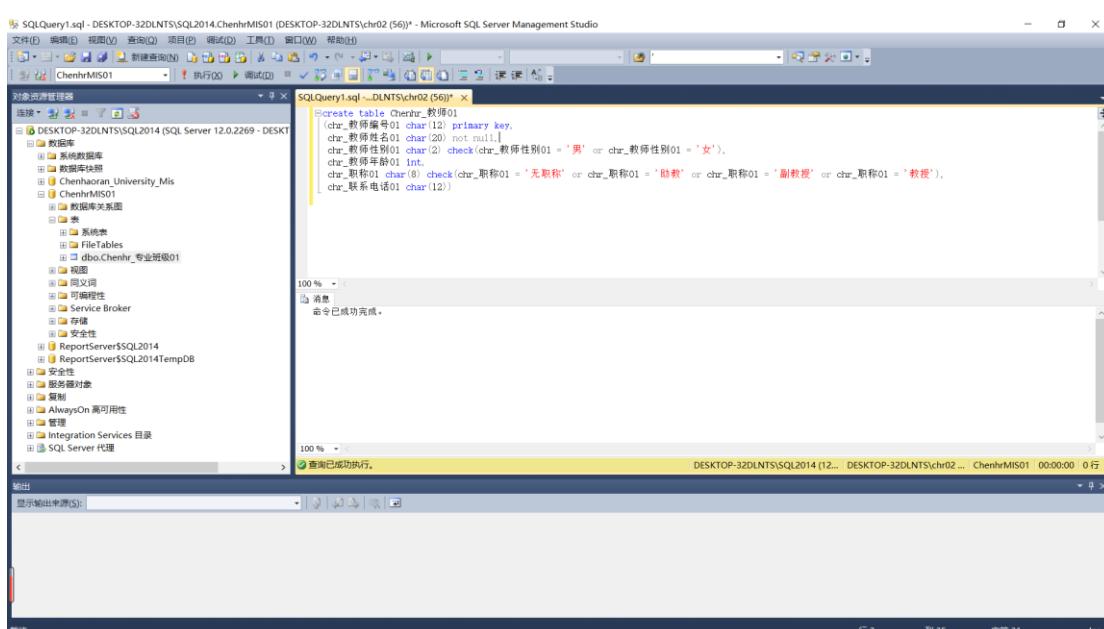
CREATE TABLE Chenhr_学生成绩01
(
    学号01 CHAR(12) PRIMARY KEY,
    专业编号01 CHAR(4),
    班级编号01 CHAR(4),
    学生性别01 CHAR(2) NOT NULL,
    CHECK (studentGender01 IN ('男', '女')),
    学生年龄01 INT CHECK (studentAge01 > 0 AND studentAge01 < 99),
    住处所在班级01 CHAR(20),
    已修学分总数01 INT CHECK (totalCredits01 >= 0),
    CONSTRAINT Class_Students FOREIGN KEY (专业编号01, 班级编号01) REFERENCES Chenhr_专业班级01 (专业编号01, 班级编号01)
)

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database 'ChenhrMIS01' selected. The right pane shows the 'SQLQuery2.sql' query window containing the SQL code for creating the 'Chenhr\_学生成绩' table. The table has columns for student ID (学号), major number (专业编号), class number (班级编号), gender (性别), age (年龄), residence location (住处所在班级), total credits (已修学分总数), and a foreign key constraint linking to the 'Chenhr\_专业班级' table. A message at the bottom of the query window indicates the command was successfully completed.

图 5-3 创建学生信息表

## 创建教师信息表



```

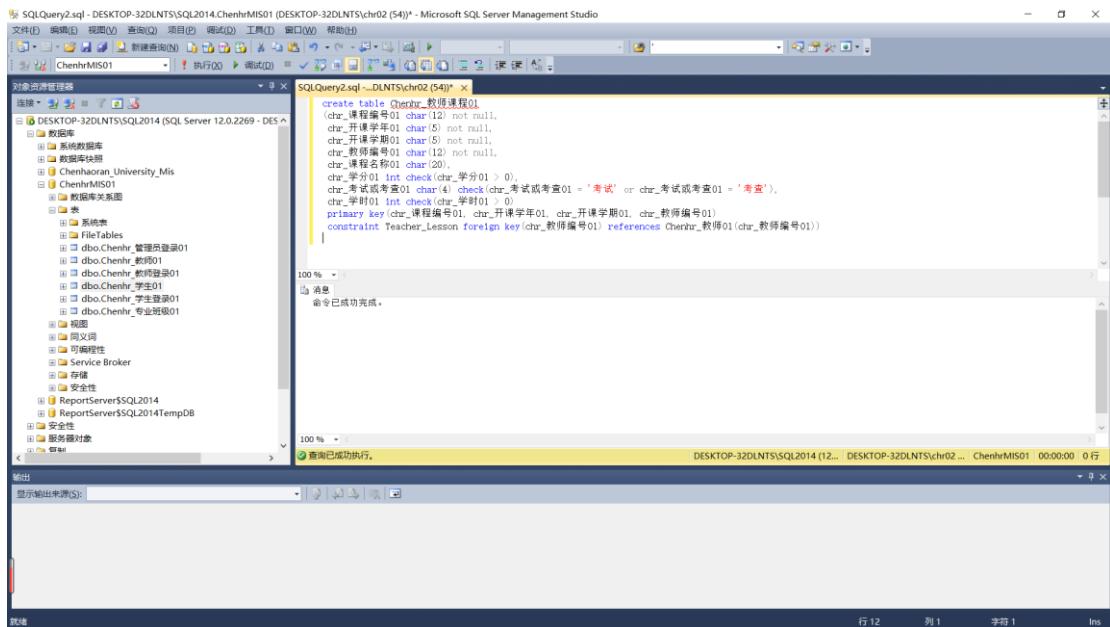
CREATE TABLE Chenhr_教师01
(
    教师编号01 CHAR(12) PRIMARY KEY,
    教师姓名01 CHAR(20) NOT NULL,
    教师性别01 CHAR(2) CHECK (teacherGender01 IN ('男', '女')),
    教师年龄01 INT CHECK (teacherAge01 > 0 AND teacherAge01 < 99),
    教师职称01 CHAR(8) CHECK (teacherTitle01 IN ('无职称', '助教', '讲师', '副教授', '教授')),
    联系电话01 CHAR(12)
)

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database 'ChenhrMIS01' selected. The right pane shows the 'SQLQuery1.sql' query window containing the SQL code for creating the 'Chenhr\_教师' table. The table has columns for teacher ID (教师编号), name (教师姓名), gender (教师性别), age (教师年龄), title (教师职称), and contact phone number (联系电话). A message at the bottom of the query window indicates the command was successfully completed.

图 5-4 创建教师信息表

## 创建教师课程信息表



```

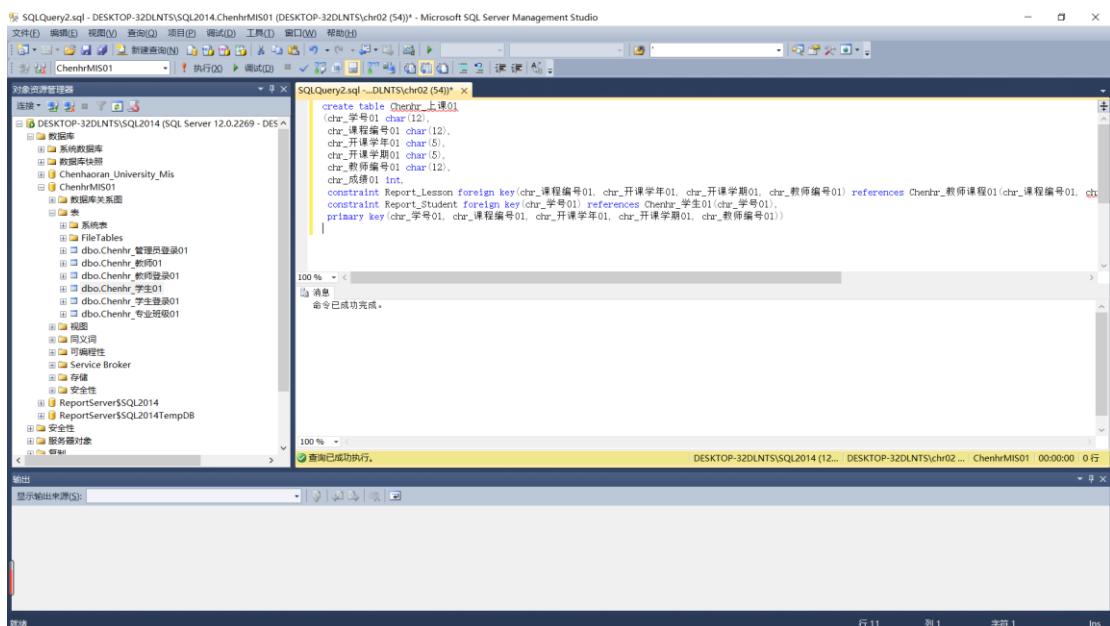
CREATE TABLE [Chenhr].[教师课程表01]
(
    [chr_课程编号01] CHAR(12) NOT NULL,
    [chr_开课学年01] CHAR(5) NOT NULL,
    [chr_开课学期01] CHAR(5) NOT NULL,
    [chr_教师编号01] CHAR(12) NOT NULL,
    [chr_学时01] INT CHECK([chr_学时01] > 0),
    [chr_考试或者考查01] CHAR(4) CHECK([chr_考试或者考查01] = '考试' OR [chr_考试或者考查01] = '考查'),
    [chr_学时01] INT CHECK([chr_学时01] > 0)
)
PRIMARY KEY ([chr_课程编号01], [chr_开课学年01], [chr_开课学期01], [chr_教师编号01])
CONSTRAINT Teacher_Lesson FOREIGN KEY ([chr_教师编号01]) REFERENCES Chenhr.[教师表01]([chr_教师编号01])

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database 'ChenhrMIS01' selected. The right pane shows the 'SQLQuery2.sql' query window containing the SQL code for creating the 'Teacher\_Lesson' table. The message '命令已成功完成。' (Command completed successfully.) is visible at the bottom of the results pane.

图 5-5 创建教师课程信息表

## 创建学生成绩信息表



```

CREATE TABLE [Chenhr].[上课表01]
(
    [chr_学号01] CHAR(12),
    [chr_课程编号01] CHAR(12),
    [chr_成绩01] CHAR(5),
    [chr_开课学年01] CHAR(5),
    [chr_开课学期01] CHAR(5),
    [chr_教师编号01] CHAR(12),
    [chr_成绩01] INT
)
CONSTRAINT Report_Lesson FOREIGN KEY ([chr_课程编号01], [chr_开课学年01], [chr_开课学期01], [chr_教师编号01]) REFERENCES Chenhr.[教师课程表01]([chr_课程编号01], [chr_开课学年01], [chr_开课学期01], [chr_教师编号01])
CONSTRAINT Report_Student FOREIGN KEY ([chr_学号01]) REFERENCES Chenhr.[学生表01]([chr_学号01])
PRIMARY KEY ([chr_学号01], [chr_课程编号01], [chr_开课学年01], [chr_开课学期01], [chr_教师编号01])

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database 'ChenhrMIS01' selected. The right pane shows the 'SQLQuery2.sql' query window containing the SQL code for creating the 'Report\_Lesson' table. The message '命令已成功完成。' (Command completed successfully.) is visible at the bottom of the results pane.

图 5-6 创建学生成绩信息表

## 创建课程开设信息表

```

create table Chenzh_开课01
(
    chr_专业编号01 char(4),
    chr_班级编号01 char(4),
    chr_课程编号01 char(12),
    chr_学年01 char(5),
    chr_学期01 char(5),
    chr_教师编号01 char(12)
);
constraint Open_Lesson foreign key(chr_课程编号01, chr_开课学年01, chr_开课学期01, chr_教师编号01) references Chenzh_教师课程01(chr_课程编号01, chr_教师编号01);
constraint Open_Class foreign key(chr_专业编号01, chr_班级编号01) references Chenzh_专业班级01(chr_专业编号01, chr_班级编号01);
primary key(chr_专业编号01, chr_班级编号01, chr_课程编号01, chr_开课学年01, chr_开课学期01, chr_教师编号01);

```

图 5-7 创建课程开设信息表

```

alter table Chenzh_上课01 drop constraint Report_Lesson;
alter table Chenzh_上课01 add constraint Report_Lesson foreign key(chr_课程编号01, chr_开课学年01, chr_开课学期01, chr_教师编号01) on update cascade on delete cascade
references Chenzh_教师课程01(chr_课程编号01, chr_开课学年01, chr_开课学期01, chr_教师编号01);

alter table Chenzh_上课01 drop constraint Report_Lesson;
alter table Chenzh_上课01 add constraint Report_Lesson foreign key(chr_学号01)
references Chenzh_学生01(chr_学号01) on update cascade on delete cascade

alter table Chenzh_上课01 drop constraint Open_Lesson;
alter table Chenzh_上课01 add constraint Open_Lesson foreign key(chr_课程编号01, chr_开课学年01, chr_开课学期01, chr_教师编号01) on update cascade on delete cascade
references Chenzh_教师课程01(chr_课程编号01, chr_开课学年01, chr_开课学期01, chr_教师编号01);

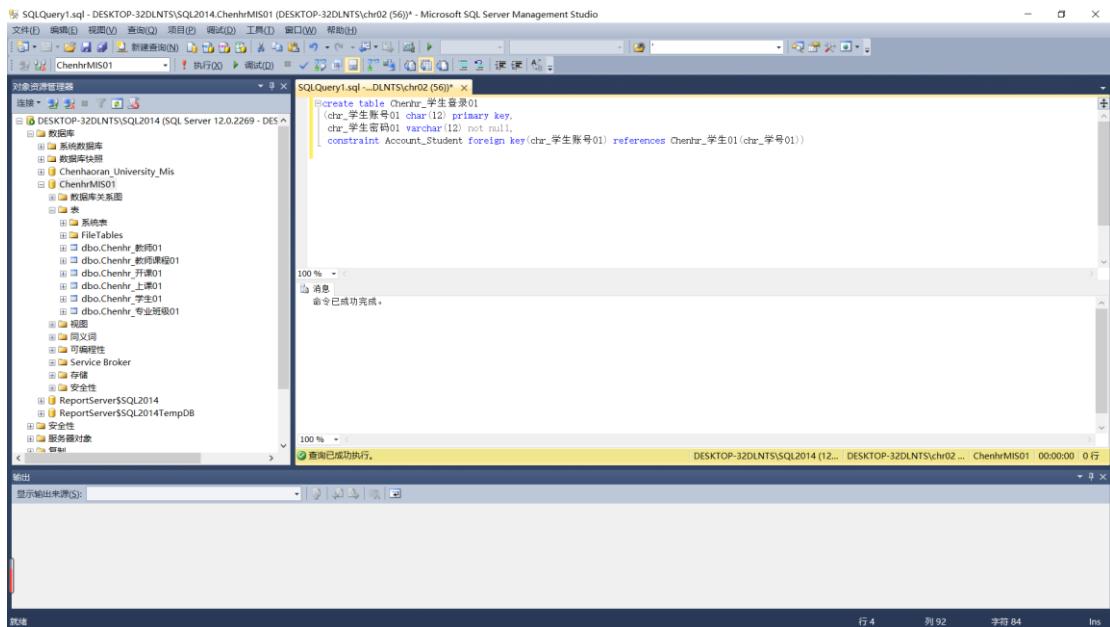
alter table Chenzh_学生01 drop constraint Account_Student;
alter table Chenzh_学生01 add constraint Account_Student foreign key(chr_学号01)
references Chenzh_学生01(chr_学号01) on update cascade on delete cascade

alter table Chenzh_教师01 drop constraint Account_Teacher;
alter table Chenzh_教师01 add constraint Account_Teacher foreign key(chr_教师编号01)
references Chenzh_教师01(chr_教师编号01) on update cascade on delete cascade

```

图 5-8 修改参照完整性约束

## 创建学生用户账号信息表

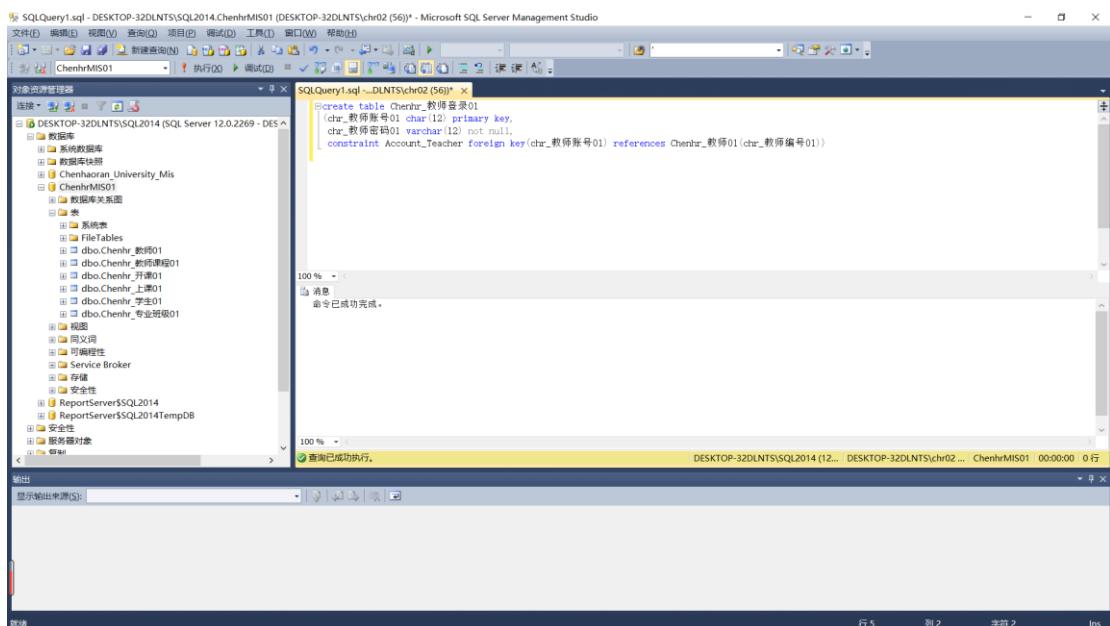


The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database 'ChenhrMIS01' selected. The right pane shows the results of a query in the 'SQLQuery1.sql' window. The query creates a table named 'Chenhr\_学生成录01' with two columns: 'chr\_学生账号01' (char(12) primary key) and 'chr\_学生密码01' (varchar(12) not null). A foreign key constraint 'Account\_Student' is defined, linking 'chr\_学生账号01' to 'chr\_账号01' in the 'Chenhr\_学生01' table. The message pane at the bottom indicates that the command was successfully completed.

```
CREATE TABLE Chenhr_学生成录01
(
    chr_学生账号01 char(12) primary key,
    chr_学生密码01 varchar(12) not null,
    constraint Account_Student foreign key (chr_学生账号01) references Chenhr_学生01 (chr_账号01)
)
```

图 5-9 创建学生用户账号信息表

## 创建教师用户账号信息表

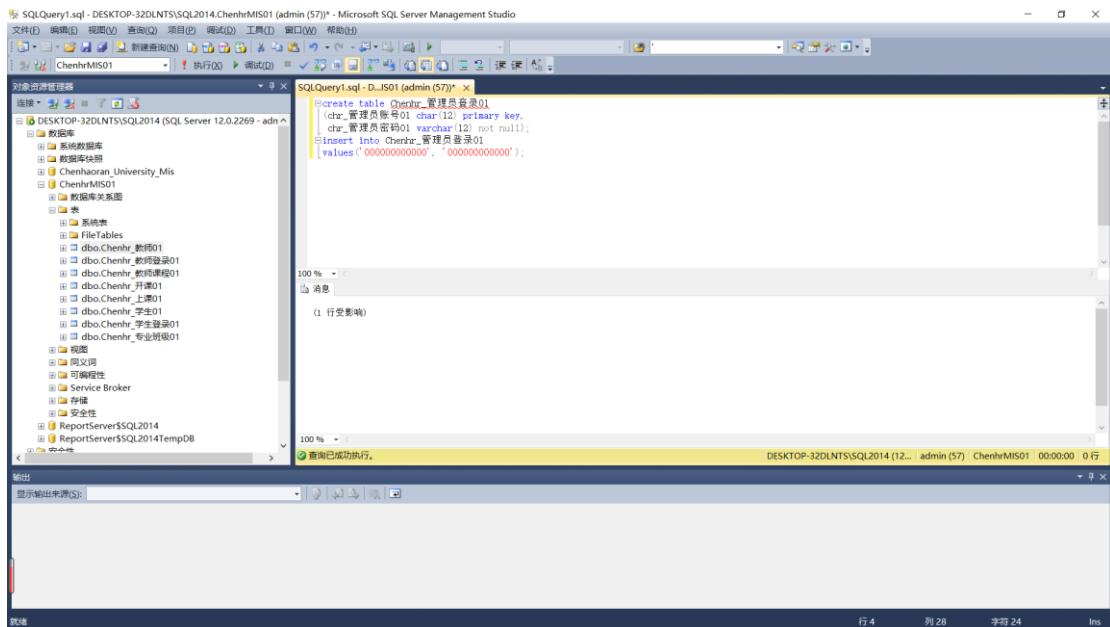


The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the database 'ChenhrMIS01' selected. The right pane shows the results of a query in the 'SQLQuery1.sql' window. The query creates a table named 'Chenhr\_教师登录01' with two columns: 'chr\_教师账号01' (char(12) primary key) and 'chr\_教师密码01' (varchar(12) not null). A foreign key constraint 'Account\_Teacher' is defined, linking 'chr\_教师账号01' to 'chr\_教师编号01' in the 'Chenhr\_教师01' table. The message pane at the bottom indicates that the command was successfully completed.

```
CREATE TABLE Chenhr_教师登录01
(
    chr_教师账号01 char(12) primary key,
    chr_教师密码01 varchar(12) not null,
    constraint Account_Teacher foreign key (chr_教师账号01) references Chenhr_教师01 (chr_教师编号01)
)
```

图 5-10 创建教师用户账号信息表

## 创建管理员账号信息表并插入数据



The screenshot shows the Microsoft SQL Server Management Studio interface. A query window titled 'SQLQuery1.sql - DESKTOP-32DLNTS\SQL2014.ChenhrMIS01 (admin (57))' is open. The code entered is:

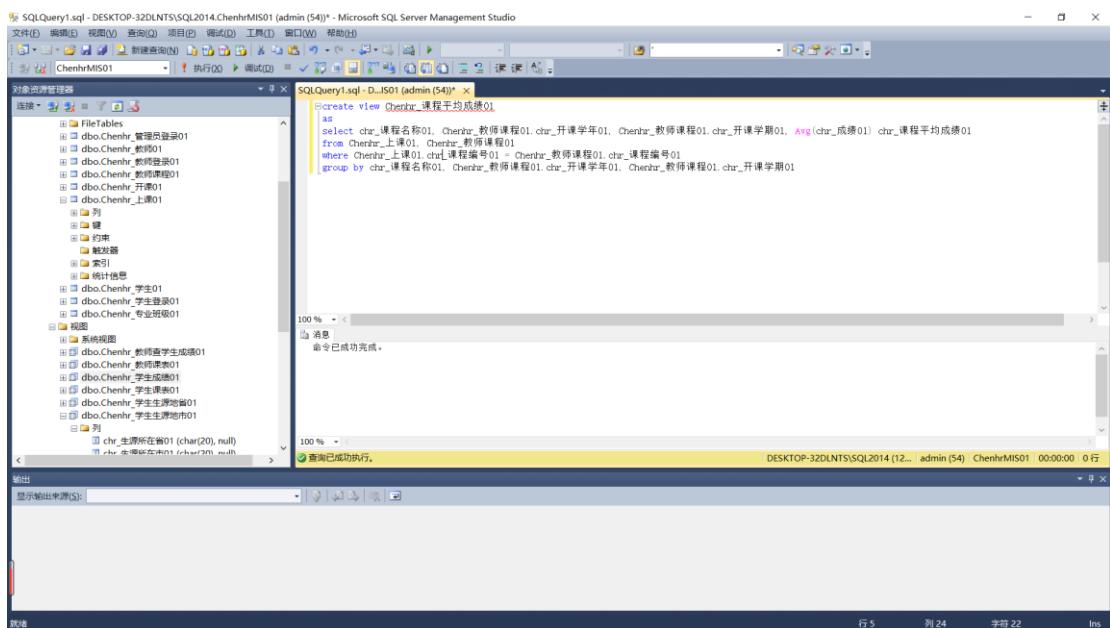
```
CREATE TABLE Chenhr_管理员登录01
(
    chr_管理员账号01 char(12) primary key,
    chr_管理员密码01 varchar(12) not null
);
Insert into Chenhr_管理员登录01
values('000000000000', '000000000000');
```

The status bar at the bottom right indicates '行 4 列 28 字符 24 Ins'.

图 5-11 创建管理员账号信息表并插入数据

## 5.2 视图的建立

### 建立课程平均成绩视图



The screenshot shows the Microsoft SQL Server Management Studio interface. A query window titled 'SQLQuery1.sql - DESKTOP-32DLNTS\SQL2014.ChenhrMIS01 (admin (54))' is open. The code entered is:

```
Create view Chenhr_课程平均成绩01
as
select chr_课程名称01, Chenhr_教师课程01.chr_开课学年01, Chenhr_教师课程01.chr_开课学期01, Avg(chr_成绩01) chr_课程平均成绩01
from Chenhr_上课01, Chenhr_教师课程01
where Chenhr_上课01.chr_课程编号01 = Chenhr_教师课程01.chr_课程编号01
group by chr_课程名称01, Chenhr_教师课程01.chr_开课学年01, Chenhr_教师课程01.chr_开课学期01;
```

The status bar at the bottom right indicates '行 5 列 24 字符 22 Ins'.

图 5-12 建立课程平均成绩视图

## 建立学生生源地省视图

```

Create view Chenhr_学生生源地省01
as
select chr_生源所在省01, count(chr_学号01) chr_学生数量01
from Chenhr_学生01
group by chr_生源所在省01;

```

图 5-13 建立学生生源地省视图

## 建立学生生源地市视图

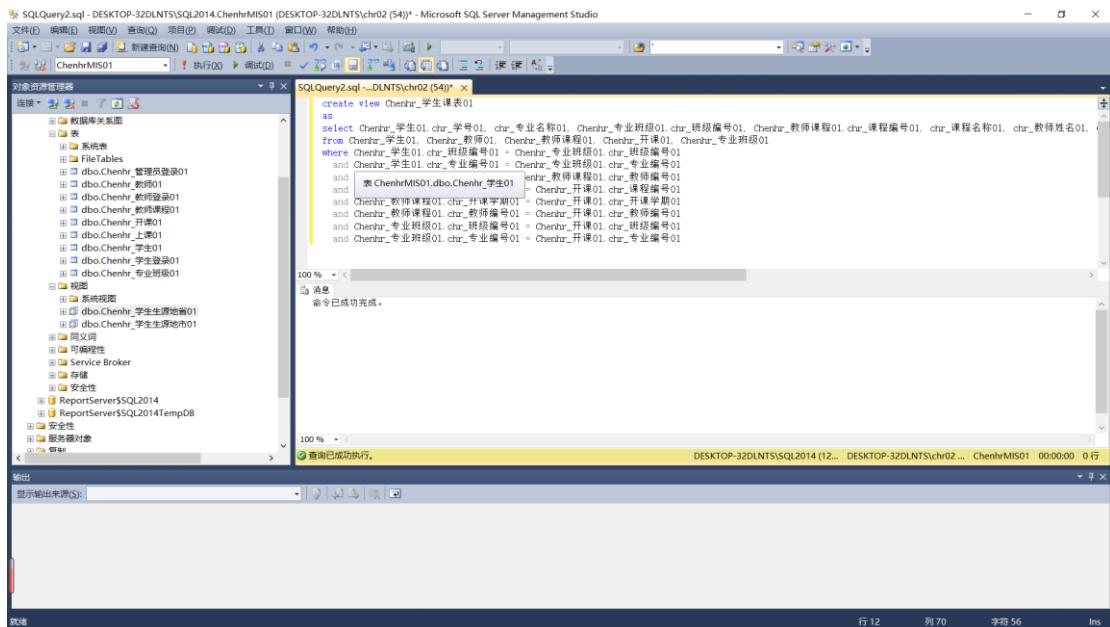
```

Create view Chenhr_学生生源地市01
as
select chr_生源所在省01, chr_生源所在市01, count(chr_学号01) chr_学生数量01
from Chenhr_学生01
group by chr_生源所在省01;

```

图 5-14 建立学生生源地市视图

## 建立学生课表视图



The screenshot shows the Microsoft SQL Server Management Studio interface. In the center pane, a query window titled 'SQLQuery2.sql - DESKTOP-32DLNTS\SQL2014.ChenhrMIS01 (DESKTOP-32DLNTS\chr02 (54))' displays the following T-SQL code:

```

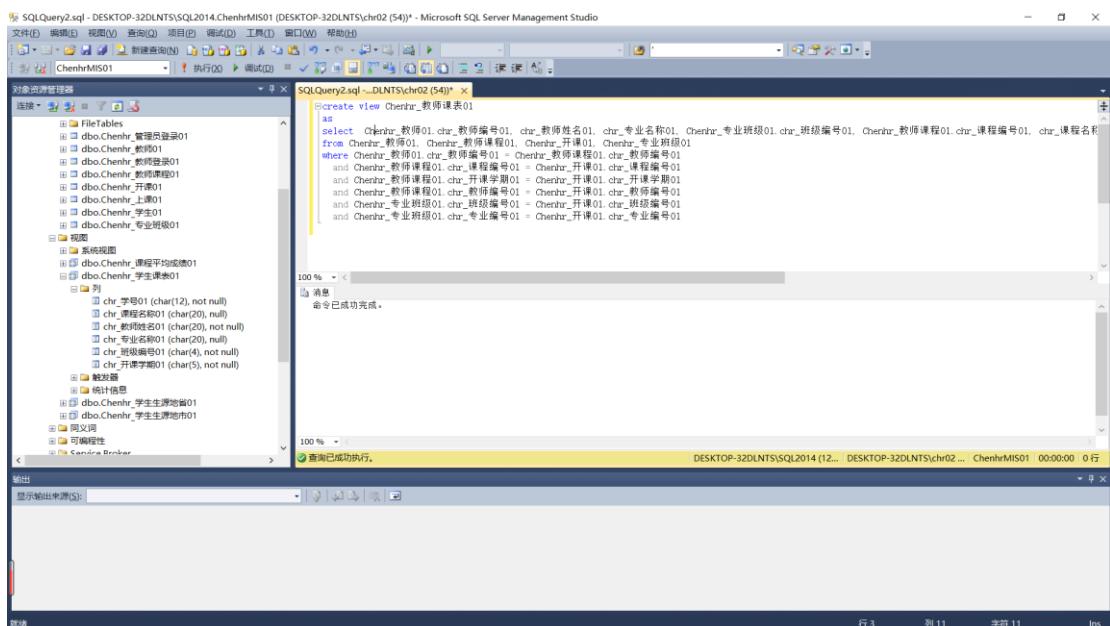
Create view Chenhr_学生课表01
as
select Chenhr_学生01.chr_学号01, chr_专业名称01, Chenhr_专业班级01.chr_班级编号01, Chenhr_教师课程01.chr_课程编号01, chr_课程名称01, chr_教师姓名01
from Chenhr_学生01,Chenhr_教师01,Chenhr_教师课程01,Chenhr_开课01,Chenhr_专业班级01
where Chenhr_学生01.chr_班级编号01 = Chenhr_教师课程01.chr_班级编号01
and Chenhr_学生01.chr_专业编号01 = Chenhr_教师课程01.chr_专业编号01
and Chenhr_教师课程01.chr_教师编号01 = Chenhr_开课01.chr_教师编号01
and Chenhr_教师课程01.chr_课程编号01 = Chenhr_开课01.chr_课程编号01
and Chenhr_教师课程01.chr_开课学期01 = Chenhr_开课01.chr_开课学期01
and Chenhr_专业班级01.chr_班级编号01 = Chenhr_开课01.chr_班级编号01
and Chenhr_专业班级01.chr_专业编号01 = Chenhr_开课01.chr_专业编号01

```

The status bar at the bottom right indicates '行 12 列 70 字符 56 Ins'.

图 5-15 建立学生课表视图

## 建立教师课表视图



The screenshot shows the Microsoft SQL Server Management Studio interface. In the center pane, a query window titled 'SQLQuery2.sql - DESKTOP-32DLNTS\SQL2014.ChenhrMIS01 (DESKTOP-32DLNTS\chr02 (54))' displays the following T-SQL code:

```

Create view Chenhr_教师课表01
as
select Chenhr_教师01.chr_教师编号01, chr_教师姓名01, chr_专业名称01, Chenhr_专业班级01.chr_班级编号01, Chenhr_教师课程01.chr_课程编号01, chr_课程名称01
from Chenhr_教师01,Chenhr_教师课程01,Chenhr_开课01,Chenhr_专业班级01
where Chenhr_教师01.chr_教师编号01 = Chenhr_教师课程01.chr_教师编号01
and Chenhr_教师课程01.chr_课程编号01 = Chenhr_开课01.chr_课程编号01
and Chenhr_教师课程01.chr_开课学期01 = Chenhr_开课01.chr_开课学期01
and Chenhr_教师课程01.chr_教师编号01 = Chenhr_开课01.chr_教师编号01
and Chenhr_专业班级01.chr_班级编号01 = Chenhr_开课01.chr_班级编号01
and Chenhr_专业班级01.chr_专业编号01 = Chenhr_开课01.chr_专业编号01

```

The status bar at the bottom right indicates '行 3 列 11 字符 11 Ins'.

图 5-16 建立教师课表视图

## 建立学生成绩视图

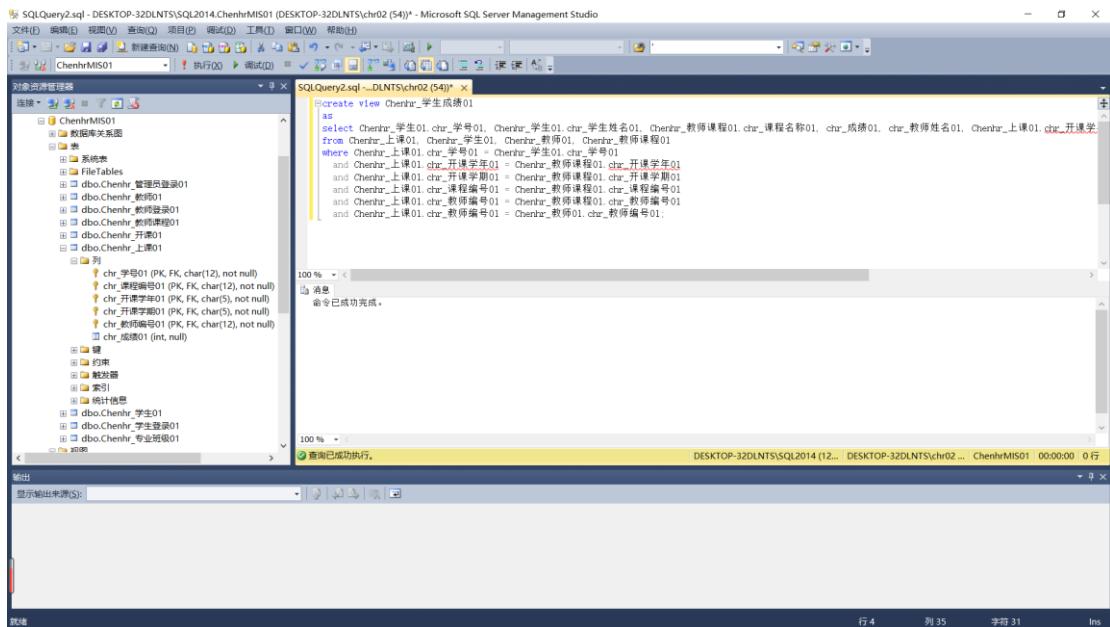


图 5-17 建立学生成绩视图

## 建立教师查学生成绩视图

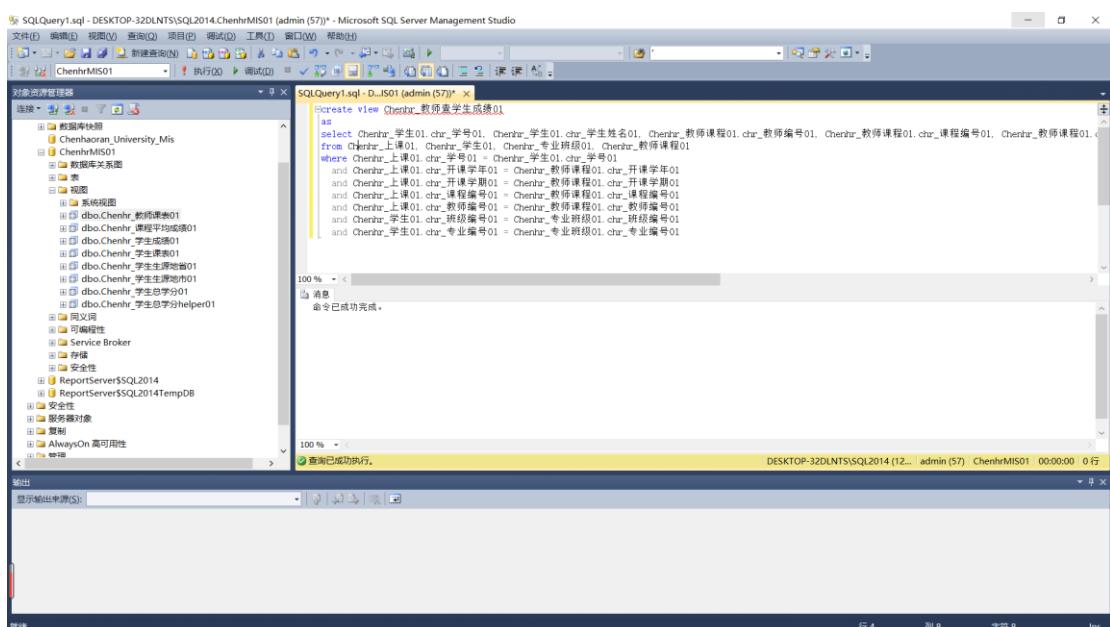


图 5-18 建立教师查学生成绩视图

## 建立学生学分视图

```
CREATE VIEW Chenhr_学生总学分helper01
AS
SELECT chr_学号01, Chenhr_上课01.chr_开课学年01, Chenhr_上课01.chr_开课学期01, SUM(chr_学分01) chr_学生总学分01
FROM Chenhr_上课01, Chenhr_教师课程01
WHERE Chenhr_教师课程01.chr_课程编号01 = Chenhr_上课01.chr_课程编号01
AND Chenhr_教师课程01.chr_开课学年01 = Chenhr_上课01.chr_开课学年01
AND Chenhr_教师课程01.chr_教师编号01 = Chenhr_上课01.chr_教师编号01
AND Chenhr_上课01.chr_成绩01 >= 60
GROUP BY chr_学号01, Chenhr_上课01.chr_开课学年01, Chenhr_上课01.chr_开课学期01
```

图 5-19 建立学生学分辅助视图

```
CREATE VIEW Chenhr_学生总学分01
AS
SELECT Openng_学生01.chr_学号01, chr_学生姓名01, Chenhr_学生总学分helper01.chr_开课学期01, chr_学生总学分01
FROM Chenhr_学生01, Chenhr_学生总学分helper01
WHERE Chenhr_学生01.chr_学号01 = Chenhr_学生总学分helper01.chr_学号01
```

图 5-20 建立学生学分视图

## 建立绩点排名视图

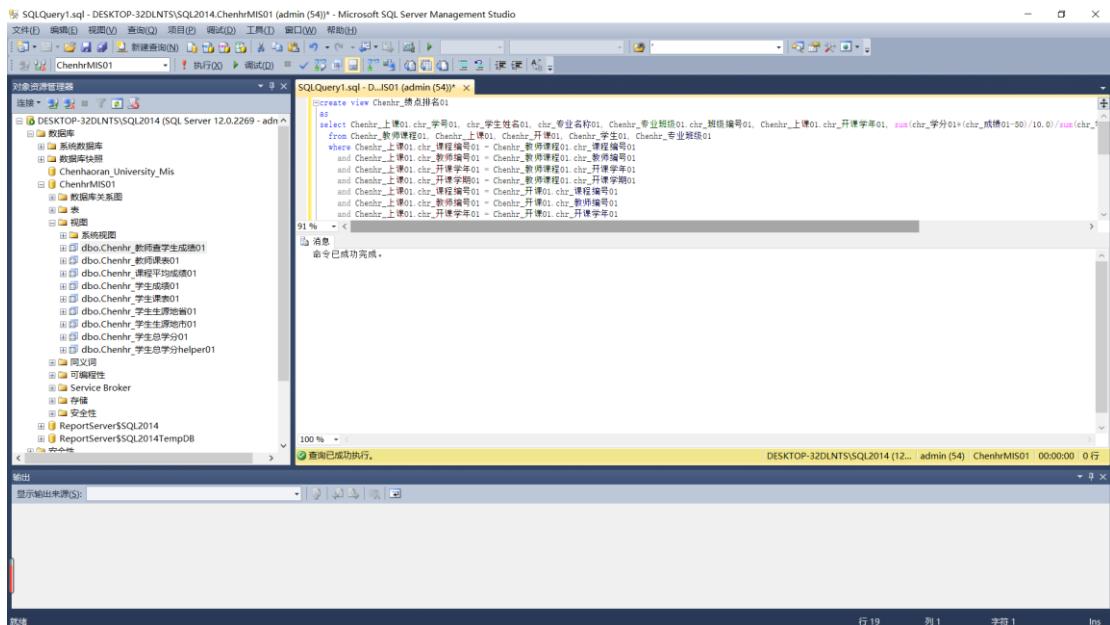


图 5-21 建立绩点排名视图

## 5.3 索引的建立

由于在定义主键时已经建立了主键列的聚集索引，因此不必再额外建立，其他属性列的索引建立操作如下。

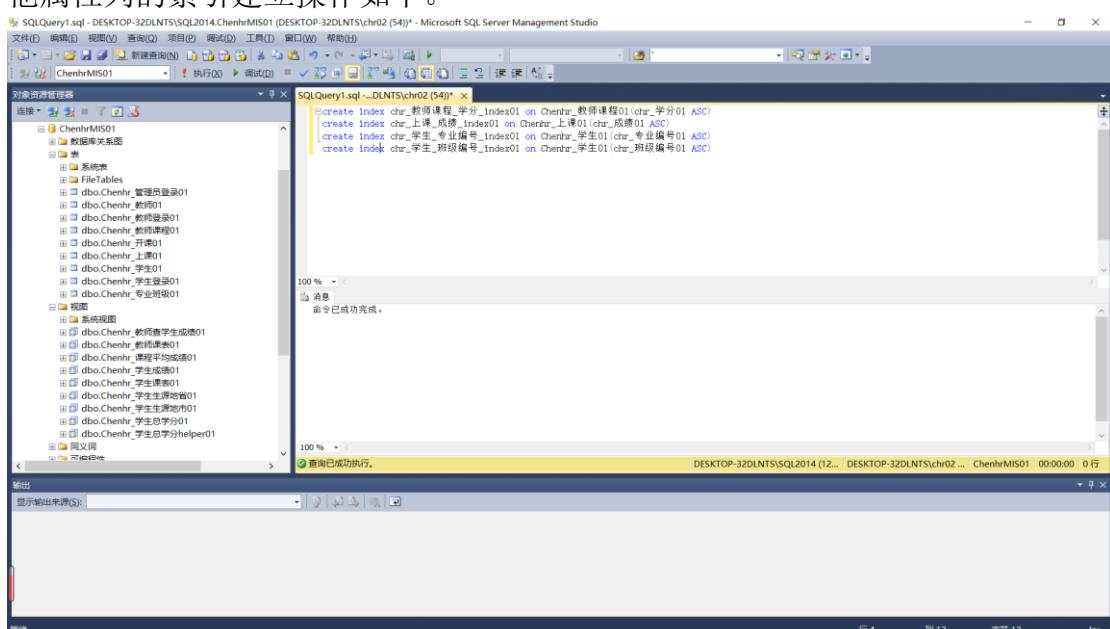
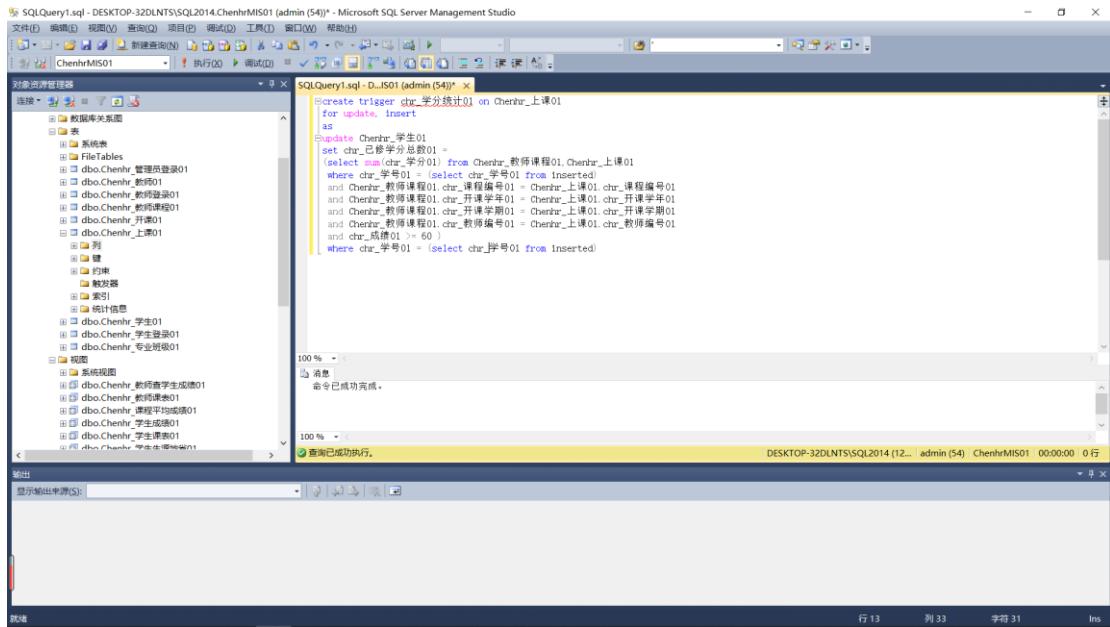


图 5-22 索引的建立

## 5.4 触发器建立

- 当用户更新或插入记录到“Chenhr\_上课 01”时，要求自动更新“Chenhr\_学生 01”中“chr\_已修学分总数 01”的值，因此在“Chenhr\_上课 01”建立触发器。

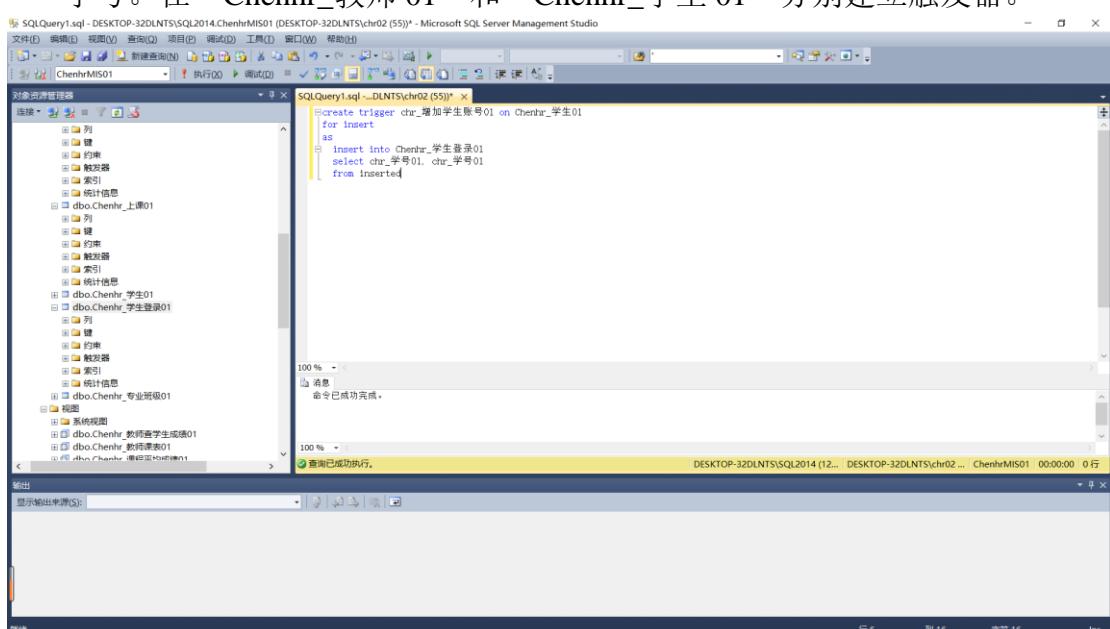


```

CREATE TRIGGER chr_学分统计01 ON Chenhr_上课01
FOR UPDATE, INSERT
AS
    UPDATE Chenhr_学生01
    SET chr_已修学分总数01 =
        (SELECT sum(chr_学分01) FROM Chenhr_教师课程01 Chenhr_上课01
        WHERE chr_学号01 = (SELECT chr_学号01 FROM Inserted)
        AND Chenhr_教师课程01.chr_课程编号01 = Chenhr_上课01.chr_课程编号01
        AND Chenhr_教师课程01.chr_开课学年01 = Chenhr_上课01.chr_开课学年01
        AND Chenhr_教师课程01.chr_开课学期01 = Chenhr_上课01.chr_开课学期01
        AND Chenhr_教师课程01.chr_教师编号01 = Chenhr_上课01.chr_教师编号01
        AND chr_成绩01 >= 60)
    WHERE chr_学号01 = (SELECT chr_学号01 FROM Inserted)
  
```

图 5-23 学分统计

- 当增加教师或学生时，其账号也应该同时增加，账号密码默认为教师编号或学号。在“Chenhr\_教师 01”和“Chenhr\_学生 01”分别建立触发器。



```

CREATE TRIGGER chr_增加学生账号01 ON Chenhr_学生01
FOR INSERT
AS
    INSERT INTO Chenhr_学生登录01
    SELECT chr_学号01, chr_学号01
    FROM Inserted
  
```

图 5-24 增加学生账号

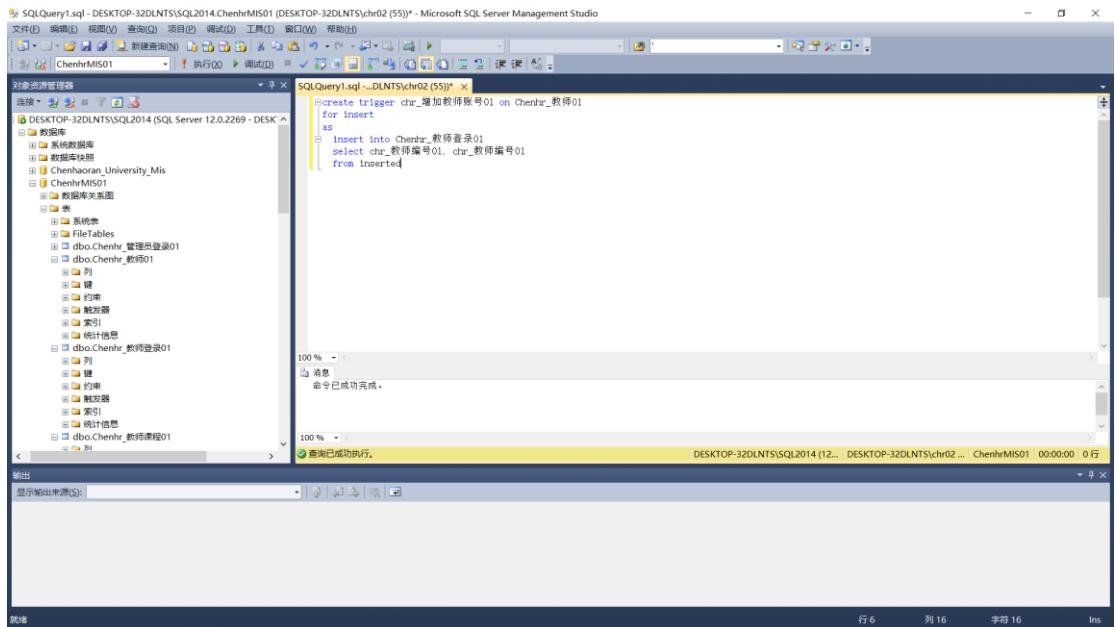


图 5-25 增加教师账号

3. 新开设课程后，管理系统应当自动对学生选课情况做出相应的增加，因此在“Chenhr\_开课 01”建立触发器。

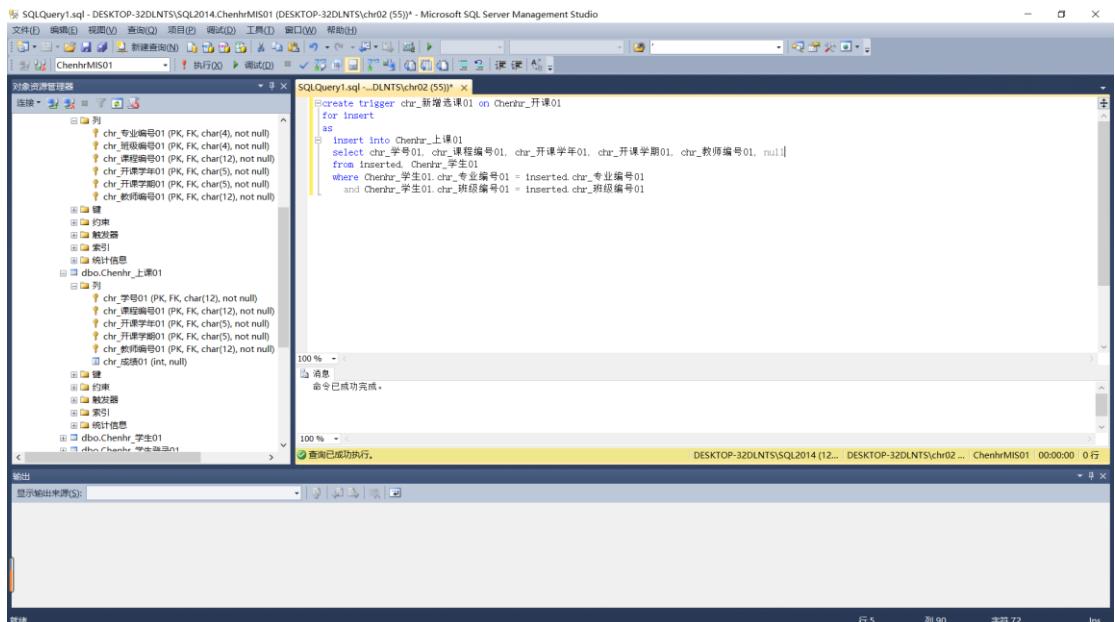


图 5-26 新增选课

## 5.5 建存储过程

- 建立存储过程，输入学号和学年，输出该学号同学对应专业的学年绩点和排名情况。

```

CREATE PROCEDURE chr_学年绩点排名sp01
    @chr_学号01 char,
    @chr_学年01 char
AS
BEGIN
    SELECT Chenhr_上课01.chr_学号01, chr_学生姓名01, chr_专业名称01, Chenhr_专业班级01, Chenhr_上课01.chr_开课学年01, SUM(chr_学分01*(chr_成绩01/chr_满分为01))
    FROM Chenhr_教师授课01, Chenhr_上课01, Chenhr_课程01, Chenhr_学生01, Chenhr_专业班级01
    WHERE Chenhr_上课01.chr_课程编号01 = Chenhr_教师授课01.chr_课程编号01
    AND Chenhr_上课01.chr_教师编号01 = Chenhr_教师授课01.chr_教师编号01
    AND Chenhr_上课01.chr_开课学年01 = Chenhr_教师授课01.chr_开课学年01
    AND Chenhr_上课01.chr_课程编号01 = Chenhr_课程01.chr_课程编号01
    AND Chenhr_上课01.chr_教师编号01 = Chenhr_教师授课01.chr_教师编号01
    AND Chenhr_上课01.chr_开课学年01 = Chenhr_教师授课01.chr_开课学年01
    AND Chenhr_上课01.chr_开课学期01 = Chenhr_教师授课01.chr_开课学期01
    AND Chenhr_上课01.chr_专业编号01 = Chenhr_教师授课01.chr_专业编号01
    AND Chenhr_学生01.chr_学号01 = Chenhr_上课01.chr_学号01
    GROUP BY Chenhr_上课01.chr_学号01, chr_学生姓名01, chr_专业名称01, Chenhr_专业班级01, Chenhr_上课01.chr_开课学年01
    ORDER BY chr_绩点01 DESC
END

```

图 5-27 学年绩点排名

- 建立存储过程，输入最低学分要求，删除学分达标（即毕业）的学生。

```

CREATE PROCEDURE chr_毕业sp01
    @chr_最低学分01 int
AS
BEGIN
    declare @chr_sno char(30)
    declare chr_cur cursor for
        select chr_学号01 from Chenhr_学生01 where chr_已修学分总数01 >= @chr_最低学分01
    open chr_cur
    fetch chr_cur into @chr_sno
    declare @PFILED_STATUS = 0
    begin
        delete from Chenhr_学生01 where chr_学号01 = @chr_sno
    end
    close chr_cur
    deallocate chr_cus
END

```

图 5-28 删除毕业学生

## 5.6 数据加载

### 导入专业班级信息表

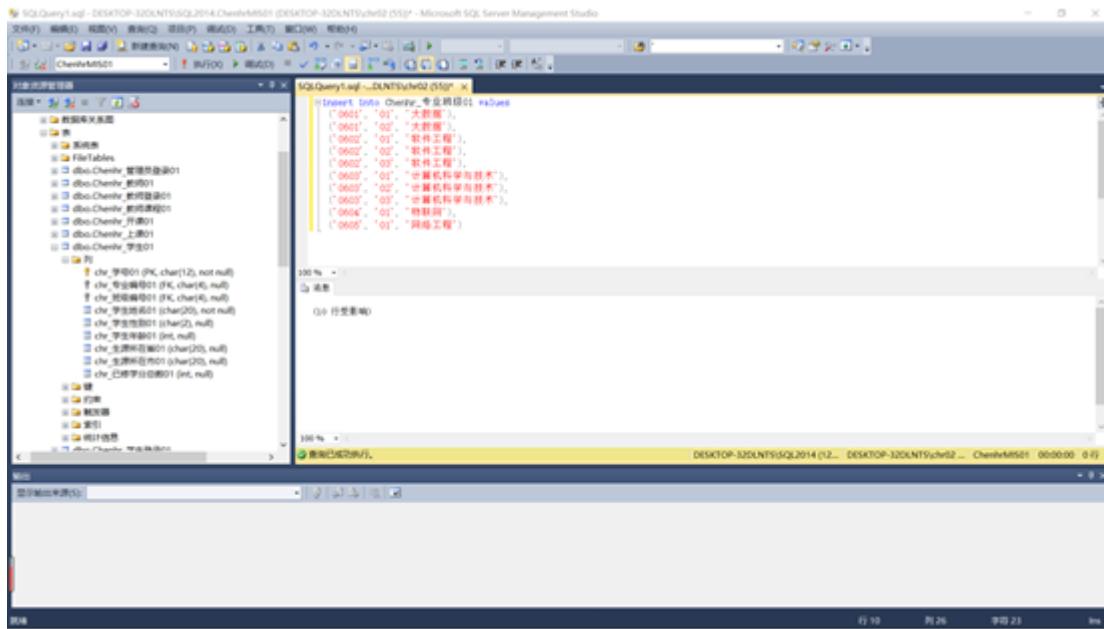


图 5-29 导入专业班级信息表

### 导入学生信息表

此时学生账号表也会同时导入，账号密码均为学号。

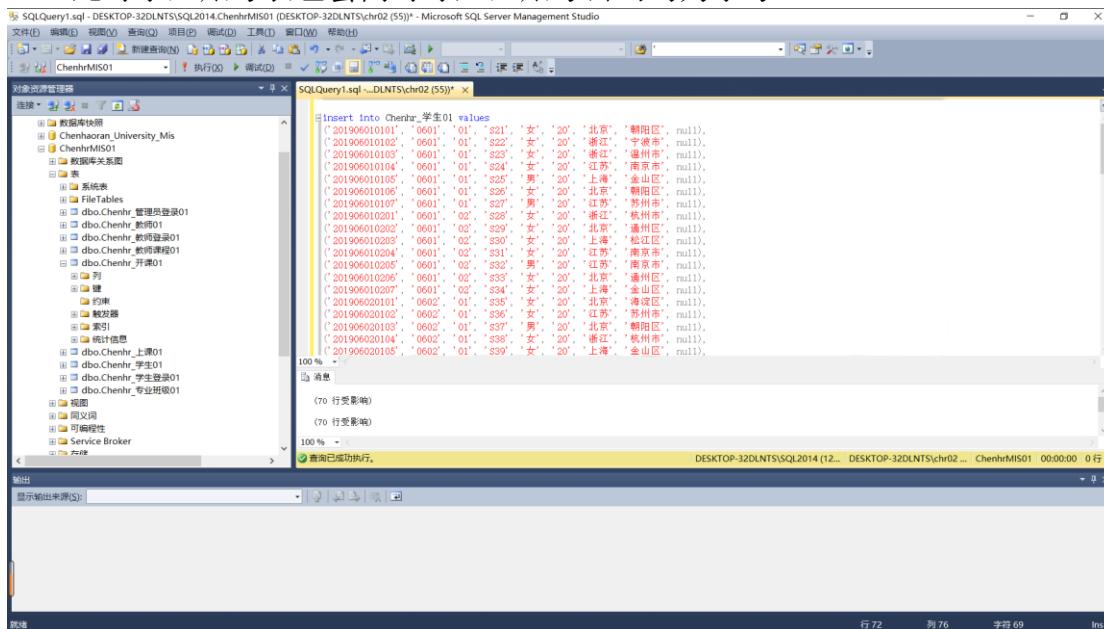


图 5-30 导入学生信息表

## 导入教师信息表

此时教师账号表也会同时导入，账号密码均为教师编号。

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left lists the database structure for 'ChenhrMIS01'. The 'Chenhr\_教师' schema contains tables like '课程', '课程成绩', '学生', and '教师'. A query editor window titled 'SQLQuery1.sql ~DLNTS\chr02 (55)' displays an 'Insert Into' statement for the '教师01' table:

```
Insert Into [教师01].values  
(  
    '000000000001', '701', '女', '56', '助教', '51419081114',  
    '000000000002', '702', '男', '63', '教授', '1415914811',  
    '000000000003', 'T03', '男', '44', '副教授', '14191851491',  
    '000000000004', 'T04', '女', '58', '教授', '98145119011',  
    '000000000005', 'T05', '女', '39', '副教授', '19045181911',  
    '000000000006', 'T06', '男', '50', '无职称', '95140891114',  
    '000000000007', 'T07', '男', '55', '助教', '14510941911')
```

The status bar at the bottom right indicates the command was successfully executed at 00:00:00 on DESKTOP-32DLNTS\SQL2014 (12...).

图 5-31 导入教师信息表

## 导入课程信息表

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left lists database objects like tables, stored procedures, and triggers. A central query window displays an 'Insert Into' statement for a table named '教师\_课程01'. The status bar at the bottom indicates the connection details: DESKTOP-32DLNTS\SQL2014...\_ChenhrMIS01, the execution time: 00:00:00.00, and the number of rows affected: 0 行 (0 rows).

图 5-32 导入课程信息表

## 导入开课信息表

此时学生选课信息表也会同时导入，成绩默认为空。

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left displays the database structure for 'ChenhrMIS01'. The 'Tables' node under 'dbo' contains several tables: '专业课程表', '教师表', '学生表', '课程表', '开课表', and '教师课程表'. A context menu is open over the '开课表' table, with options like '插入(I)' (Insert), '更新(U)...', '删除(D)...', '选择(S)...', '转换(T)...', '另存为(S...)', '设计(E)...', '快照(S...)', and '属性(P)...'. The 'FileTables' node also lists '开课表'. The 'Script' node under '开课表' has a submenu with '插入(I)' selected, which is reflected in the central query window.

The central query window shows the following T-SQL script:

```
Insert Into 开课表 values
('0601', '01', 'C01', '2020', '2', '0000000002'),
('0601', '01', 'C02', '2020', '2', '0000000002'),
('0601', '01', 'C03', '2020', '2', '0000000005'),
('0601', '01', 'C05', '2020', '1', '0000000001'),
('0601', '01', 'C06', '2020', '1', '0000000005'),
('0601', '01', 'C07', '2020', '1', '0000000007'),
('0601', '02', 'C02', '2020', '2', '0000000002'),
('0601', '02', 'C04', '2020', '2', '0000000005'),
('0601', '02', 'C05', '2020', '1', '0000000007'),
('0601', '02', 'C06', '2020', '1', '0000000002'),
('0602', '01', 'C01', '2020', '2', '0000000001'),
('0602', '01', 'C02', '2020', '2', '0000000004'),
('0602', '02', 'C03', '2020', '2', '0000000007'),
('0602', '01', 'C06', '2020', '1', '0000000009'),
('0602', '02', 'C01', '2020', '2', '0000000001'),
('0602', '02', 'C02', '2020', '2', '0000000009'),
('0602', '02', 'C03', '2020', '2', '0000000005'),
('0602', '02', 'C06', '2020', '1', '0000000002'),
('0602', '03', 'C01', '2020', '2', '0000000001'),
('0602', '03', 'C02', '2020', '2', '0000000002'),
```

The status bar at the bottom right indicates 'DESKTOP-32DLNTS\SQL2014(12...' and 'ChenhrMIS01 00:00:00 0 行'.

图 5-33 导入开课信息表

## 导入成绩信息

填写成绩信息时学生的学分、绩点、排名会同时改变。

图 5-34 导入成绩信息

## 6、应用系统开发与试运行

### 6.1 开发平台和开发环境介绍

开发平台: SQL SERVER2014 + Python3.9

开发环境: window10

### 6.2 前台界面与后台数据库连接说明、代码实现

使用了 Python 的 pymssql 库连接数据库和对数据进行操作，使用 tkinter 库开发图形交互界面。

在 pymssql 的 connect 函数中输入服务器名称、登录名、密码、要连接的数据库名，即可连接。连接数据库代码如图。

```
try:
    db = pymssql.connect('DESKTOP-32DLNTS\SQL2014', 'admin', 'admin',
                         'ChenhrMIS01')
except Exception:
    tk.messagebox.showerror(title='Error', message='连接数据库失败!')
```

图 6-1 连接数据库

### 6.3 系统各功能设计和运行界面截图

系统登录界面如图。登录界面有一个下拉框，2 个输入文本框，以及一个登录按钮，在下拉框选择身份后输入帐号密码，点击登录即可登录系统。



图 6-2 登录界面

### 6.3.1 学生界面

进入学生界面后初始界面如图所示。

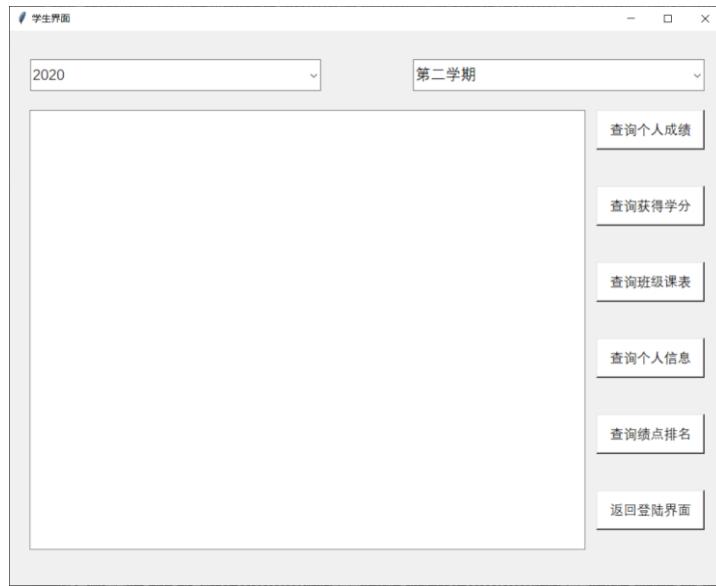


图 6-3 学生界面

学生账号可进行的操作有：

- 查询个人成绩
- 查询获得学分
- 查询班级课表
- 查询个人信息
- 查询绩点排名

#### 查询个人成绩

在上方下拉框选择学年和学期，点击“查询个人成绩”，中间表格框中便会显示该学生本学期各科成绩，如图所示。

学号	姓名	课程	得分	任课老师
201906010101	S21	数据库原理及其应用	54	T02
201906010101	S21	算法设计与分析	72	T02
201906010101	S21	数据挖掘	97	T05

图 6-4 查询个人成绩

## 查询获得学分

在上方下拉框选择学年和学期，点击“查询获得学分”，中间表格框中便会显示该学生本学期获得学分总数，如图所示。

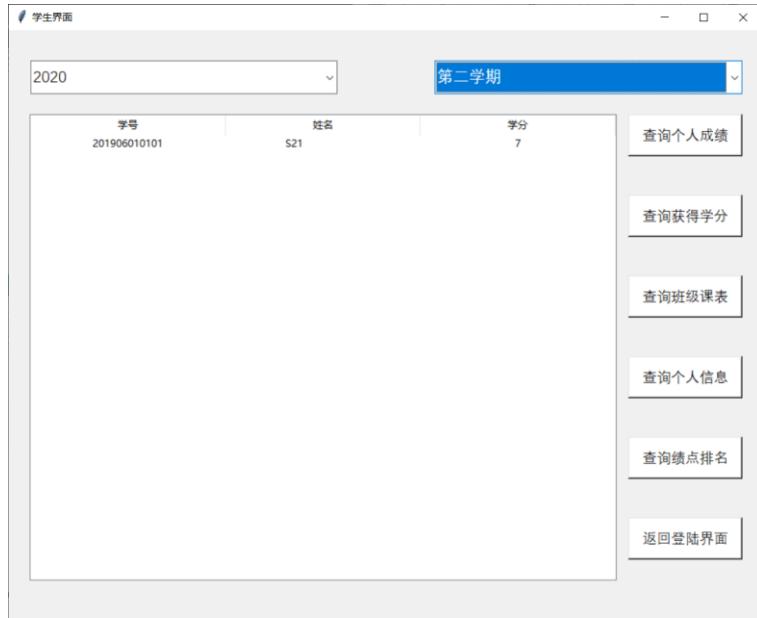


图 6-5 查询获得学分

## 查询班级课表

在上方下拉框选择学年和学期，点击“查询班级课表”，中间表格框中便会显示该学生所在班级本学期课表，如图所示。

The screenshot shows the same '学生界面' window. The '学期' dropdown is now set to '第二学期'. The central area displays a table of course schedules:

班级	课程编号	课程名称	任课老师	学时	考核方式	学分
大数据01	C01	数据库原理及其应用	T02	48	考试	3
大数据01	C02	算法设计与分析	T02	48	考试	3
大数据01	C04	数据挖掘	T05	64	考试	4

To the right of the table are the same vertical buttons as in Figure 6-5: '查询个人成绩', '查询获得学分', '查询班级课表' (highlighted in blue), '查询个人信息', '查询绩点排名', and '返回登陆界面'.

图 6-6 查询班级课表

### 查询个人信息

点击“查询个人信息”，中间表格框中便会显示该学生的个人信息及已修学分总数，如图所示。



图 6-7 查询个人信息

### 查询绩点排名

在上方下拉框选择学年，点击“查询绩点排名”，中间表格框中便会显示本学年该学生所在专业的绩点情况，并按绩点由高到低排序，如图所示。

The screenshot shows a window titled "学生界面". At the top, there are two dropdown menus: "2020" and "第二学期". Below them is a table with the following data:

学号	姓名	班级	学年	绩点
201906010104	S24	大数据01	2020	4.270588
201906010203	S30	大数据02	2020	4.005882
201906010107	S27	大数据01	2020	3.288235
201906010201	S28	大数据02	2020	3.152941
201906010206	S33	大数据02	2020	2.888235
201906010204	S31	大数据02	2020	2.876470
201906010205	S32	大数据02	2020	2.741176
201906010105	S25	大数据01	2020	2.523529
201906010101	S21	大数据01	2020	2.317647
201906010207	S34	大数据02	2020	2.264705
201906010102	S22	大数据01	2020	1.647058
201906010202	S29	大数据02	2020	1.076470

To the right of the table is a vertical column of buttons:

- 查询个人成绩
- 查询获得学分
- 查询班级课表
- 查询个人信息
- 查询绩点排名** (This button is highlighted)
- 返回登陆界面

图 6-8 查询绩点排名

### 6.3.2 教师界面

进入教师界面后初始界面如图所示。



图 6-9 教师界面

教师账号可进行的操作有：

- 学生成绩管理
- 个人信息查询
- 教师授课查询
- 课程平均成绩

#### 学生成绩管理

点击“学生成绩管理”，中间表格框中便会显示学习该教师所授课程的学生的信息及成绩，如图所示。

学号	姓名	课程编号	课程名称	专业	开课时间	成绩
201906010101	S21	C05	计算机网络原理	大数据	2020(1)	50
201906020306	S54	C01	数据库原理及其应用	软件工程	2020(2)	50
201906030302	S71	C05	计算机网络原理	计算机科学	2020(1)	51
201906040101	S77	C05	计算机网络原理	物联网	2020(1)	51
201906050107	S90	C01	数据库原理及其应用	网络工程	2020(2)	51
201906050107	S90	C05	计算机网络原理	网络工程	2020(1)	52
201906302021	S63	C01	数据库原理及其应用	计算机科学	2020(2)	52
201906201016	S40	C01	数据库原理及其应用	软件工程	2020(2)	52
201906030307	S76	C01	数据库原理及其应用	计算机科学	2020(2)	53
201906040102	S78	C01	数据库原理及其应用	物联网	2020(2)	54
201906020202	S43	C01	数据库原理及其应用	软件工程	2020(2)	54
201906050104	S87	C01	数据库原理及其应用	网络工程	2020(2)	54
201906050102	S85	C05	计算机网络原理	网络工程	2020(1)	55
201906301016	S61	C05	计算机网络原理	计算机科学	2020(1)	55
201906203037	S55	C01	数据库原理及其应用	软件工程	2020(2)	56
201906020104	S38	C01	数据库原理及其应用	软件工程	2020(2)	56
201906040104	S80	C05	计算机网络原理	物联网	2020(1)	56
201906050105	S88	C01	数据库原理及其应用	网络工程	2020(2)	56
201906050101	S84	C05	计算机网络原理	网络工程	2020(1)	57
201906303033	S72	C05	计算机网络原理	计算机科学	2020(1)	57
201906301013	S58	C05	计算机网络原理	计算机科学	2020(1)	57
201906302023	S65	C01	数据库原理及其应用	计算机科学	2020(2)	58
201906202027	S48	C01	数据库原理及其应用	软件工程	2020(2)	60
201906030304	S73	C01	数据库原理及其应用	计算机科学	2020(2)	60
201906050103	S86	C01	数据库原理及其应用	网络工程	2020(2)	60
201906030206	S68	C01	数据库原理及其应用	计算机科学	2020(2)	61

图 6-10 学生成绩管理

点击其中某个学生的成绩，可以在右侧文本框中对其进行修改，同时会有触发器修改该名学生的学分，当成绩大于 60 分时，该课学分将加入到该学生的已修学分中。修改过程如图所示。

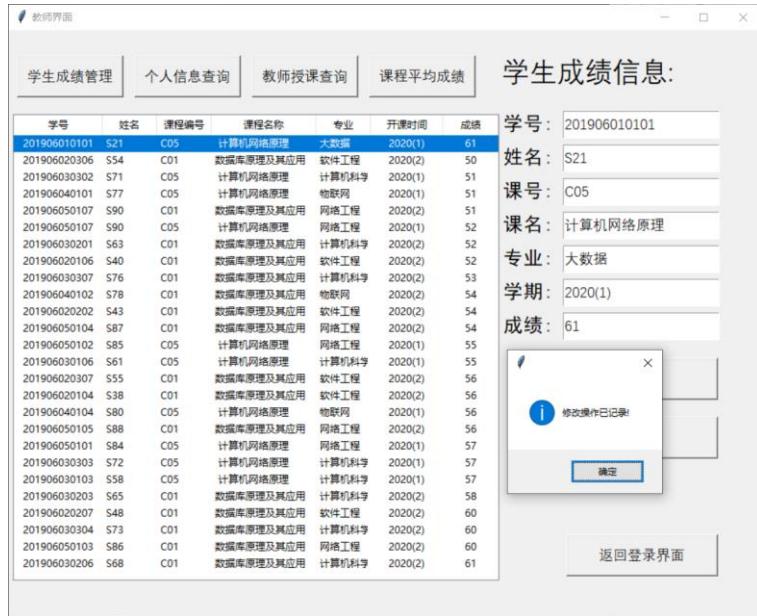


图 6-11 修改学生成绩

## 个人信息查询

点击“个人信息查询”，中间表格框中便会显示该教师的信息，如图所示。



图 6-12 个人信息查询

## 教师授课查询

点击“教师授课查询”，中间表格框中便会显示该教师所授课程的班级、名称、编号、开课时间、学时、考核方式、学分等信息，如图所示。



图 6-13 教师授课查询

## 课程平均成绩

点击“课程平均成绩”，中间表格框中便会显示所有课程的平均成绩，如图所示。



图 6-14 课程平均成绩

### 6.3.3 教务处界面

进入教务处界面后初始界面如图所示。



图 6-15 教务处界面

教务处管理员账号可进行的操作有：

- 学生信息管理，包括增加、修改、删除学生信息，以及删除毕业学生
- 教师信息管理，包括增加、修改、删除教师信息
- 地区信息统计

#### 学生信息管理

点击“学生信息管理”，下方表格框中便会显示所有学生的信息，如图所示。

学号	专业编号	班级编号	姓名	性别	年龄	生源所在省	生源所在市	已修学分总数
201906010101	0601	01	S21	女	20	北京	海淀区	11
201906010102	0601	01	S22	女	20	浙江	宁波市	10
201906010104	0601	01	S24	女	20	江苏	南京市	17
201906010105	0601	01	S25	男	20	上海	金山区	17
201906010107	0601	01	S27	男	20	江苏	苏州市	17
201906010201	0601	02	S28	女	20	浙江	杭州市	17
201906010202	0601	02	S29	女	20	北京	朝阳区	11
201906010203	0601	02	S30	女	20	上海	宝山区	17
201906010204	0601	02	S31	女	20	江苏	南京市	17
201906010205	0601	02	S32	男	20	江苏	南京市	17
201906010206	0601	02	S33	女	20	北京	朝阳区	17

图 6-16 学生信息管理

点击其中某个学生的信息，可以在上方文本框中对其进行修改，再点击“修改学生信息”保存修改。也可以在文本框中输入学生信息后，点击“新建学生信息”，进行学生的插入，此时会有触发器新建学生的登录账号，默认账号密码都是学号。点击“删除学生信息”，即可删除选中的学生。在“已修学分总数”文本框中输入毕业最低学分，点击“删除毕业学生”，即可删除大于该学分的所有毕业学生。

### 教师信息管理

点击“教师信息管理”，下方表格框中便会显示所有教师的信息，如图所示。

The screenshot shows a Windows application window titled "教师信息管理". On the left, there is a form labeled "教师信息:" with input fields for "编号", "姓名", "性别", "年龄", "职称", and "电话". On the right, there is a section labeled "操作选择:" with three buttons: "新建教师信息", "修改教师信息", and "删除教师信息". Below these sections is a table displaying teacher data:

编号	姓名	性别	年龄	职称	联系电话
000000000001	T01	女	46	助教	51419081114
000000000002	T02	男	63	教授	14915914811
000000000003	T03	男	44	副教授	14191851491
000000000004	T04	女	58	教授	98145119012
000000000005	T05	女	39	副教授	19045181911
000000000006	T06	男	50	无职称	95114089114
000000000007	T07	男	55	助教	145109419114

At the bottom center is a "返回" (Return) button.

图 6-17 教师信息管理

点击其中某个教师的信息，可以在上方文本框中对其进行修改，再点击“修改教师信息”保存修改。也可以在文本框中输入教师信息后，点击“新建教师信息”，进行教师的插入，此时会有触发器新建教师的登录账号，默认账号密码都是教师编号。点击“删除教师信息”，即可删除教师的信息。

## 地区信息统计

点击“教师信息管理”，便会进入相应界面，如图所示。



图 6-18 地区信息统计

若选择省份不限，点击查询，则表格框中会显示所有省份的学生人数，如图所示。

省	人数
北京	17
江苏	15
上海	15
浙江	21

A screenshot of the same application window as Figure 6-18. The dropdown menu now shows '不限'. The central area contains a table with four rows. The first row has headers '省' (Province) and '人数' (Number of Students). The subsequent three rows list Beijing (17), Jiangsu (15), Shanghai (15), and Zhejiang (21). To the right of the table are the 'Query' and 'Return' buttons.

图 6-19 省份不限

选择身份后，将出现市区信息下拉框，若选择不限，点击查询，则表格框中会显示所有市区的学生人数，如图所示。

The screenshot shows a window titled '地区信息统计' (Region Information Statistics) with a green header bar containing the title. Below the header are two dropdown menus: the first is set to '北京' (Beijing), and the second is set to '不限' (Unlimited). A large table below the dropdowns displays student counts for three districts: Chaoyang District (5 students), Haidian District (5 students), and Tongzhou District (7 students). To the right of the table are two buttons: '查询' (Query) and '返回' (Return).

市	人数
朝阳区	5
海淀区	5
通州区	7

图 6-20 市区不限

选择市区后，则表格框中显示该市区的学生人数，如图所示。

This screenshot shows the same window as Figure 6-20, but with a different selection in the second dropdown menu: '朝阳区' (Chaoyang District). The table now only displays data for the Chaoyang District, showing 5 students. The '查询' (Query) and '返回' (Return) buttons are also visible.

市	人数
朝阳区	5

图 6-21 市区

## 6.4 数据库的备份

### 6.4.1 数据库的故障和备份概述

故障的种类有事务故障、系统故障、介质故障、病毒破坏等。大到自然灾害，小到病毒感染、电源故障乃至操作员失误等，都会影响数据库系统的正常运行，可能引起数据库的破坏甚至系统的瘫痪。

如果不进行数据备份，一旦发生故障，则将无法恢复丢失的数据，学生的成绩信息只能重新录入，这样不但会花费大量的精力和时间，甚至可能造成学生各项信息的丢失，影响学生评奖评优乃至毕业。因此，进行数据库的备份至关重要。

### 6.4.2 备份策略

数据转储时数据库恢复中经常采用的基本技术，是最简单的确保能恢复大部分信息的方法。数据转储是指由 DBA 定期地将整个数据库复制到磁带或另一个磁盘上保存起来的过程。当数据库遭到破坏时，便可用保存的后备副本把数据库恢复至某个一致性的状态。本高校成绩管理数据库系统的备份策略包括：

1. 采用静态转储。转储期间不允许对数据库有任何操作活动，以保证后备副本数据的一致性。
2. 采用海量转储。由于数据库系统规模较小，增量转储的效率提高效果不明显，因此采用较简单的海量转储。
3. 备份周期采用每两月一次。由于数据转储十分耗费时间和资源，且高校成绩管理数据库系统本身更新频率较小，因此可以适当降低转储频率。鉴于本系统一般仅在学期初和学期末存在大量更新，而一学期一般在 16 周（约 4 个月），因此采用每两月更新一次的策略是相对合理的。

### 6.4.3 备份的实施

现在商品化的 RDBMS 都为 DBA 提供了数据库转储与恢复的工具与命令。本数据库系统基于的 SQL Server 提供了“分离/附加”数据库、“备份/还原数据库”、复制数据库等多种数据库备份和恢复的方法。这里我们使用常用的“分离/附加”方法。具体操作如下：

1. 分离数据库即把本数据库从 SQL Server 数据库列表中删除，使其不再被管理和使用，但该数据库的文件 (.MDF) 和对应的日志文件 (.LDF) 完好无损。分离成功后，就可把该数据库文件和对应的日志文件复制到其他盘中作为备份保存。分离数据库操作如图 6-22。
2. 附加数据库即把备份磁盘中的数据库文件和对应的日志文件复制到需要的计算机，并将其添加到某个 SQL Server 数据库服务器中，由该服务器来管理和使用这个数据库。附加数据库操作如图 6-23。

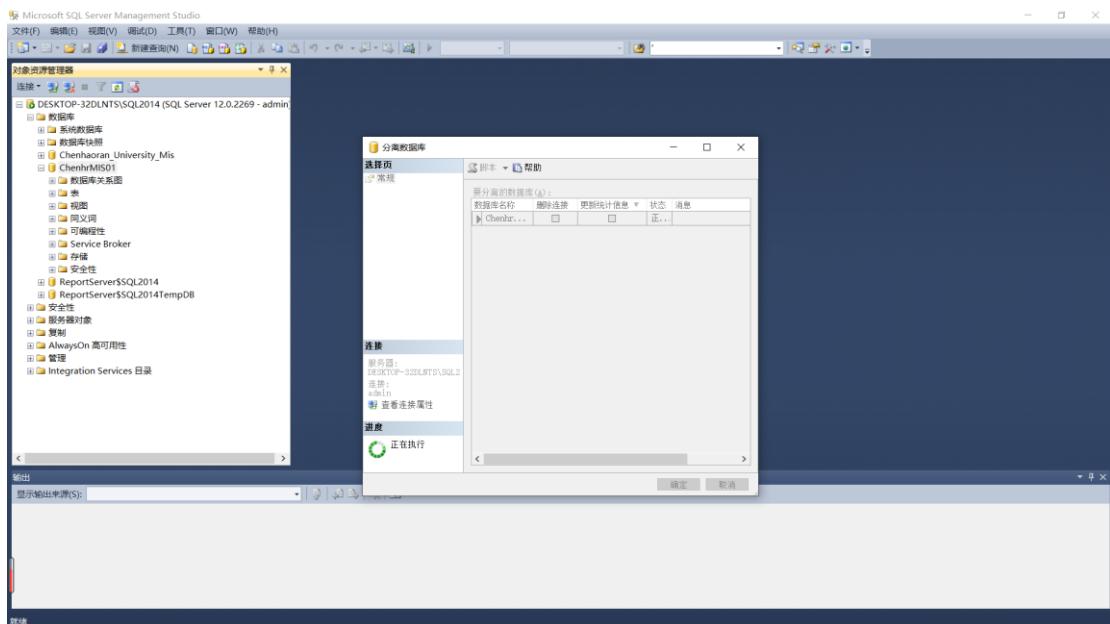


图 6-22 分离数据库

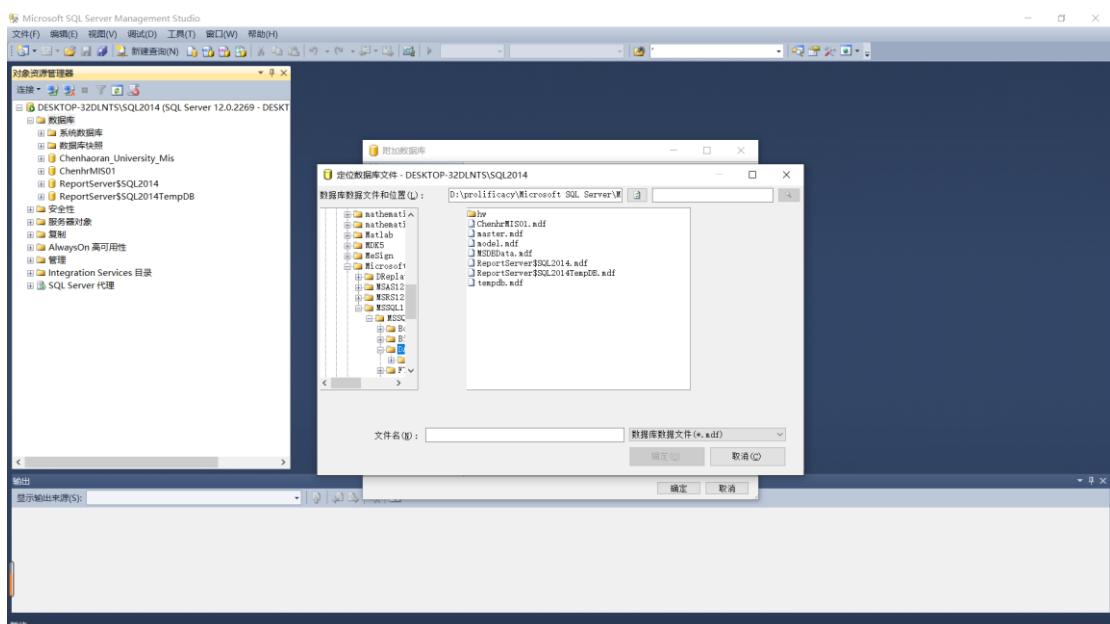


图 6-23 附加数据库

## 7、实验总结

### 7.1 遇到的问题和解决的办法

- 报错“选择列表中的列 ‘.....’ 无效，因为该列没有包含在聚合函数或 GROUP BY 子句中。”

解决方法：选择列表中任何非聚合表达式内的每个属性名都应包含在 GROUP BY 列表中，因此只需将选择列表中未加入 GROUP BY 列表的属性名补上即可。

## 2. 图形界面显示乱码的问题。

编号	姓名	性别	年龄	职称	联系电话
000000000001	T01	男	46	高级工程师	51419081114
000000000002	T02	女	63	研究员	14915914811
000000000003	T03	男	44	博士生导师	14191851491
000000000004	T04	男	58	教授	98145119011
000000000005	T05	女	39	讲师	19045181911
000000000006	T06	男	50	助教	95114089114
000000000007	T07	女	55	高级工程师	14510941911

图 7-1 图形界面显示乱码

解决方法: SQL SERVER 的默认编码是 Latin-1, 不支持中文, 因此出现乱码。要支持中午需要把数据库的默认编码修改为 gbk, 故只需在输出前先用 encode('latin-1')进行编码, 再用 decode('gbk')解码即可。

## 7.2 系统设计的不足

1. 表格的显示比较杂乱, 查看表格时缺少筛选、排序等实用按键;
2. 下拉框的值直接在程序中规定好, 而不是通过数据库中的值确定;
3. 没有真正实现不同用户在数据库中的权限不同, 而是用不同用户看到的界面不同代替;
4. 系统界面不够美观。

## 7.3 进一步改进思路和体会

1. 可以考虑加入背景图片, 使界面更美观;
2. 加入更多的实用的控件;
3. 实现更多的查询统计功能;
4. 真正实现不同用户在数据库中的权限不同;
5. 可以实现通过数据库中的值来决定下拉框的值。

在本次实验中, 我们切身经历了数据库规划、需求分析、概念结构设计、逻辑结构设计、物理结构设计以及数据库的实施和维护的整个流程, 设计的学生成绩管理数据库系统规模较小, 功能也有待改进, 但在这不到一周的时间内, 要完成数据库的开发和完成前端界面设计, 以及报告的书写, 时间还是相当紧迫的, 同时也使我收获颇丰。