

**学校教务系统的设计与实现**

|  |  |
| --- | --- |
| **二级学院** | **医药信息工程学院** |
| **专 业** | **计算机科学与技术** |
| **班 级** | **外包特色班19（2）** |
| **学生姓名** | **钟学炫** |
| **学 号** | **1920502254** |
| **指导教师** | **李闵** |
|  |  |
| 2021年5月 | |

目录

[1需求分析 1](#_Toc12675)

[1.1设计背景 1](#_Toc26505)

[1.1.1系统开发背景  1](#_Toc32736)

[1.1.2系统分析 1](#_Toc788)

[1.1.3用户需求分析 1](#_Toc25216)

[1.2 数据流图（或功能模块图） 2](#_Toc24593)

[1.3 数据字典 3](#_Toc20239)

[1.3.1数据结构 3](#_Toc22900)

[1.3.2数据项定义 4](#_Toc18653)

[1.4 安全性和完整性要求 8](#_Toc20682)

[2 概念结构设计 8](#_Toc28855)

[2.1 实体的设计 8](#_Toc31554)

[2.2 联系的设计 8](#_Toc10339)

[2.3 概念模型（ER图） 9](#_Toc22439)

[3 数据库物理设计 10](#_Toc3858)

[3.1 表的设计 10](#_Toc7582)

[3.2 索引的设计 14](#_Toc1604)

[3.3 视图的设计 18](#_Toc14181)

[3.4 存储过程的设计 24](#_Toc10996)

[3.5 触发器的设计 28](#_Toc8927)

[3.6 角色和用户的设计 34](#_Toc19721)

[3.7 约束的设计 38](#_Toc15286)

[4 数据库实施 41](#_Toc32604)

[4.1 代码的生成 41](#_Toc16344)

[4.2 相关内容的改进 70](#_Toc27853)

[4.3 数据测试 70](#_Toc10399)

[4.3.1视图的测试 70](#_Toc17224)

[4.3.2存储过程测试 73](#_Toc1627)

[4.3.3触发器测试 75](#_Toc19369)

[4.3.4约束测试 81](#_Toc14623)

[5 School数据库逆向生成ER图 89](#_Toc15822)

# 1需求分析

## 1.1设计背景

### 1.1.1系统开发背景

随着计算机科学与技术的迅速发展，全世界几乎所有的公司都离不开互联网和计算机。信息化席卷全世界，也伴随用户数量的急剧增长，这引起了这些公司对数据存储的成本，格式，安全，存放位置等等的考虑。

在教育方面也迫切需要数据管理系统，以便于对每一届的学生进行信息管理。数据管理系统需要合理的分析，精确的设计，重复的测试，才得以在实践中发挥其作用。

为此我设计了一个学生信息管理系统，以便于老师，管理员，学生三者对所有相关信息进行妥善的管理和利用。

### 1.1.2系统分析

学生信息管理系统的目的是对学生的综合信息进行管理。建立该管理系统的主要目标是为该校提供全面的学生信息管理的解决方案，以解决传统的人工管理方式效率低，保密性差，不利于查找，更新和维护的问题，极大地提高学校学生综合测评成绩管理的效率，节约教育经费，以适应学校信息化以及全球信息化建设的发展趋势。

系统的数据输入信息包括系别、班级、课程、学生、成绩、老师、请假信息、学籍调动信息等等。系统的数据输出信息包括计算机汇总输出信息，如各类查询结果、各类报表等。

本系统主要能实现以下功能：

1、能够浏览、查询学生成绩、学籍异地调动、请假记录等所有信息；

2、能够对学生的信息注册登记管理；

3、能够查询教师请假信息、教师评价信息等所有信息；

4、管理人员能对所有信息进行维护；

5、能对不同用户设置管理权限。

6、能进行学生成绩查询、添加、删除和修改；

7、能进行教师信息查询、添加、删除和修改；

8、能进行课程信息查询、添加、删除和修改；

9、能进行系别信息查询、添加、删除和修改；

### 1.1.3用户需求分析

（1）管理员类型 管理员可以成功登陆系统，系统管理员有权限进行如下操作：1.修改教师个人信息 2.修改学生个人信息 3.修改学生成绩 4.修改系别信息 5.审核和管理课程信息 6.维护学生请假或者教师请假记录 7.学生异地调动记录等等。

（2）教师类型 教师可以成功登录系统，教师有权限进行如下操作：1.录入学生课程成绩 2.查询学生的成绩 3.修改个人信息4.选择班级代表 5.所教课程 6.教师请假 7.查询学生异地调动等等

（3）学生类型 学生可以成功登录系统，学生有权限进行如下操作：1.查询成绩2.查询和选择所选修课程3.修改个人信息 4.学生请假 5.教师评价等等。

## 1.2 数据流图（或功能模块图）

用户

登录

选择用户类型

学生

老师

管理员

教师请假以及其他

查询或批准学生请假

查询班级或学生信息

修改个人密码以及个人版信息

查询或申请教师课程

录入和查询学生成绩

教师评价

查询

考试成绩

查询

或选修课程

更新个人信息及密码信息

学生请假以及其他

更新教师相关信息

更新学生

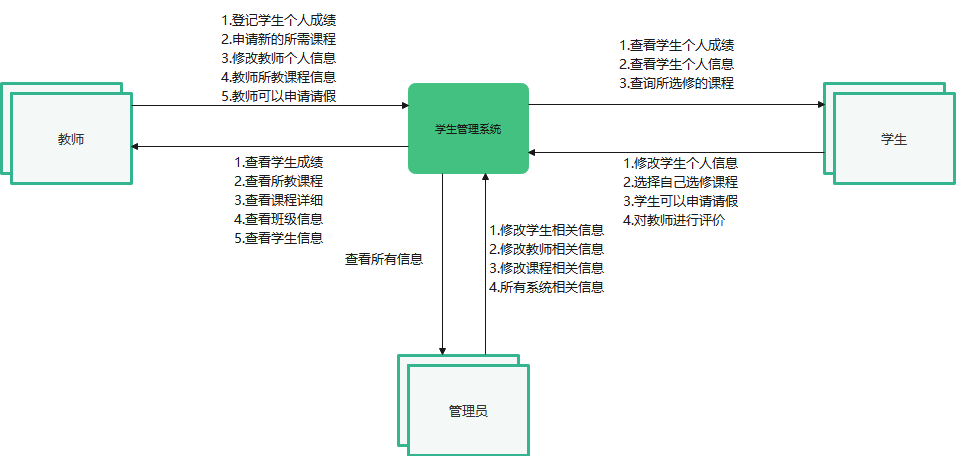
相关信息

查询系统信息

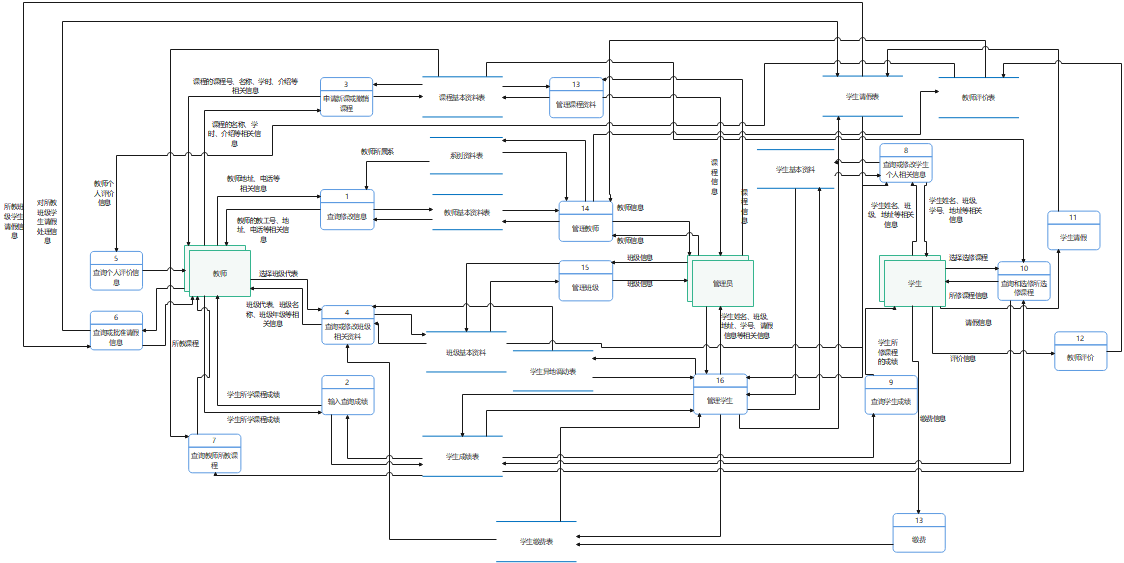
更新成绩相关信息

更新课程相关信息

图一 整体设计流程



图二 数据流程顶层图



图三 数据流程第二层图

## 1.3 数据字典

### 1.3.1数据结构

|  |  |  |
| --- | --- | --- |
| **数据结构** | | |
| **编号** | 数据结构名 | 组成 |
| **1** | 学生信息 | 学号、姓名、性别、出生年月、联系电话、家庭地址、所在班级编号 |
| **2** | 课程信息 | 课程编号、课程名称、课时数、课程介绍 |
| **3** | 系别信息 | 系编号、系名称、联系电话、系教师人数、系学生人数 |
| **4** | 班级信息 | 班级编号、所属系编号、班级名称、班长、辅导员、入学年、班级类型、班级学制、班级人数 |
| **5** | 学生请假信息 | 请假日期、请假原因、请假课程号、教师号、学号 |
| **6** | 成绩信息 | 学号、课程号、成绩 |
| **7** | 教师任课信息 | 课程号、班级号、任课老师编号、任教年度、任教学期 |
| **8** | 学籍异地调动信息 | 调动开始时间、调动结束时间、调动原因 |
| **9** | 教师请假信息 | 请假日期、请假原因、请假课程编号、教师编号、班级号 |
| **10** | 学生缴费信息 | 应缴费用、缴费年度、是否缴费 |
| **11** | 教师评价信息 | 学号、课程号、教师号、评分、评价年度、评价学期 |
| **12** | 教师信息 | 教师号、姓名、性别、出生日期、级别 |

### 1.3.2数据项定义

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **学生表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **学号** | Varchar(9) | 否 |  | 主键，聚集 |
| **姓名** | varchar(8) | 否 |  |  |
| **性别** | varchar(2) | 否 | ‘男’ |  |
| **出生日期** | DateTime |  |  |  |
| **联系电话** | varchar(15) |  |  |  |
| **家庭地址** | varchar(100) |  |  |  |
| **所在班级编号** | Varchar(6) | 否 |  | 外键 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **课程表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **课程号** | varchar(4) | 否 |  | 主键，聚集 |
| **课程名** | varchar(20) | 否 |  |  |
| **课时数** | smallint | 否 | 60 |  |
| **课程介绍** | Varchar(200) |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **系别表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **系编号** | varchar(2) | 否 |  | 主键，聚集 |
| **系名称** | varchar(50) | 否 |  |  |
| **联系电话** | varchar(15) |  |  |  |
| **系教师人数** | int |  |  |  |
| **系学生人数** | int |  |  |  |
| **系主任** | varchar(4) |  |  | 外键 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **教师表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **教师号** | varchar(4) | 否 |  | 主键，聚集 |
| **姓名** | varchar(8) | 否 |  |  |
| **性别** | varchar(2) |  |  |  |
| **出生日期** | datetime |  |  |  |
| **级别** | varchar(10) |  | ‘讲师’ |  |
| **所在系别编号** | varchar(2) |  |  | 外键 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **班级表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **班级编号** | varchar(6) | 否 |  | 主键，聚集 |
| **班级名称** | varchar(20) | 否 |  |  |
| **辅导员** | varchar(4) |  |  | 外键 |
| **班长** | varchar(9) |  |  | 外键 |
| **所在系别编号** | varchar(2) |  |  | 外键 |
| **班级类型** | varchar(10) | 否 |  |  |
| **班级学制** | smallint | 否 |  |  |
| **入学年份** | datetime | 否 |  |  |
| **班级人数** | int |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **选修表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **学号** | varchar(9) | 否 |  | 组合主键，外键 |
| **课程号** | varchar(4) | 否 |  | 组合主键，外键 |
| **选修年度** | varchar(10) | 否 |  | 组合主键 |
| **选修学期** | int | 否 |  | 组合主键 |
| **成绩** | Smallint |  | 0 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **任教表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **课程编号** | varchar(4) | 否 |  | 组合主键，外键 |
| **班级编号** | varchar(4) | 否 |  | 组合主键，外键 |
| **任课教师编号** | varchar(4) | 否 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **教师请假表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **教师工号** | varchar(4) | 否 |  | 组合主键，外键 |
| **课程编号** | varchar(6) | 否 |  | 组合主键，外键 |
| **班级编号** | varchar(4) | 否 |  | 组合主键，外键 |
| **请假日期** | datetime | 否 |  |  |
| **请假缘由** | varchar（100） | 否 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **学籍异地调动表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **调动开始时间** | datetime | 否 |  | 组合主键 |
| **调动结束时间** | datetime | 否 |  | 组合主键 |
| **调动原因** | varchar（100） | 否 |  |  |
| **学号** | Varchar(9) | 否 |  | 组合主键，外键 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **学生课程请假表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **学号** | varchar(9) | 否 |  | 主键，外键 |
| **课程编号** | varchar(4) | 否 |  | 主键，外键 |
| **教师工号** | varvhar(4) | 否 |  | 主键，外键 |
| **请假日期** | datetime | 否 |  | 主键 |
| **请假缘由** | varchar(100) | 否 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **教师评价表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **学号** | varchar(9) | 否 |  | 组合主键，外键 |
| **课程编号** | varchar(4) | 否 |  | 组合主键，外键 |
| **教师工号** | varvhar(4) | 否 |  | 组合主键，外键 |
| **评分** | int |  | 0 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **学生缴费表数据项定义** | | | | |
| **列名** | 数据类型 | 空 | 默认值 | 键/索引 |
| **班级编号** | varchar(6) | 否 |  | 组合主键，外键 |
| **学号** | varchar(9) | 否 |  | 组合主键，外键 |
| **缴费年度** | int | 否 |  | 组合主键 |
| **应缴费用** | int | 否 |  |  |
| **是否已缴费** | bit |  | 0 |  |

## 1.4 安全性和完整性要求

系统应满足实体完整性、参照完整性和用户自定义的完整性规则。对不同的用户赋予不同的权限，每个用户学生可以成功登录系统，学生有权限进行如下操作：1.查询成绩（课程成绩或者综合测评的各项成绩）2.输入综合测评的相关加分项目及分值3.修改个人密码以及个人基本信息4.教师请假。 每个用户教师可以成功登录系统，教师有权限进行如下操作：1.查询成绩（课程成绩或者综合测评的各项成绩）2.查询自身的学生评价分数3.修改个人密码以及个人基本信息4.教师请假。 超级管理员可以成功登录系统，管理有权限进行如下操作：查询和修改、更新、维护一切系统的信息。

# 2 概念结构设计

## 2.1 实体的设计

1、教师实体，其属性有教师编号、系别编号、姓名、性别、出生日期、级别。

2、系别实体，其属性有系编号、系名称、系主任、联系电话、系教师人数、系学生人数。

3、班级实体，其属性有班级编号、班级名称、辅导员、班长、班级人数、入学年份、班级类型、班级学制、系别编号。

4、学生实体，其属性有学号、姓名、性别、出生日期、联系电话、地址、班级编号。

5、课程实体，其属性有课程编号、课程名称、课程介绍、课时数。

6、学籍异地调动实体（弱实体：依赖于学生实体），其属性有学号，调动开始时间，调动结束时间，调动原因。

## 2.2 联系的设计

二元联系：

一对一联系：

1.班级和班长之间的担任联系:一个班级只有一个班长，一个班长只能有一个班级

一对多联系：

1、老师和系别之间的属于联系：一个系有多个老师，一个老师只能属于一个系。

2、学生和系别之间的属于联系：一个系有多个学生，一个学生只能属于一个系。

3、班级和学生之间的属于联系：一个班级有多个学生，一个学生只能属于一个班级。

4、老师和班级之间的担任联系：一个老师能担任多个班级辅导员，但是一个班只有一个老师当辅导员。

5、老师和系之间的担任系主任关系:一个老师能担任多个系主任，一个系主任一位老师担任。

6、学生和学籍异地调动的调动关系关系:一个学生可以有多次调动记录，一个调动记录只能被一个学生持有。

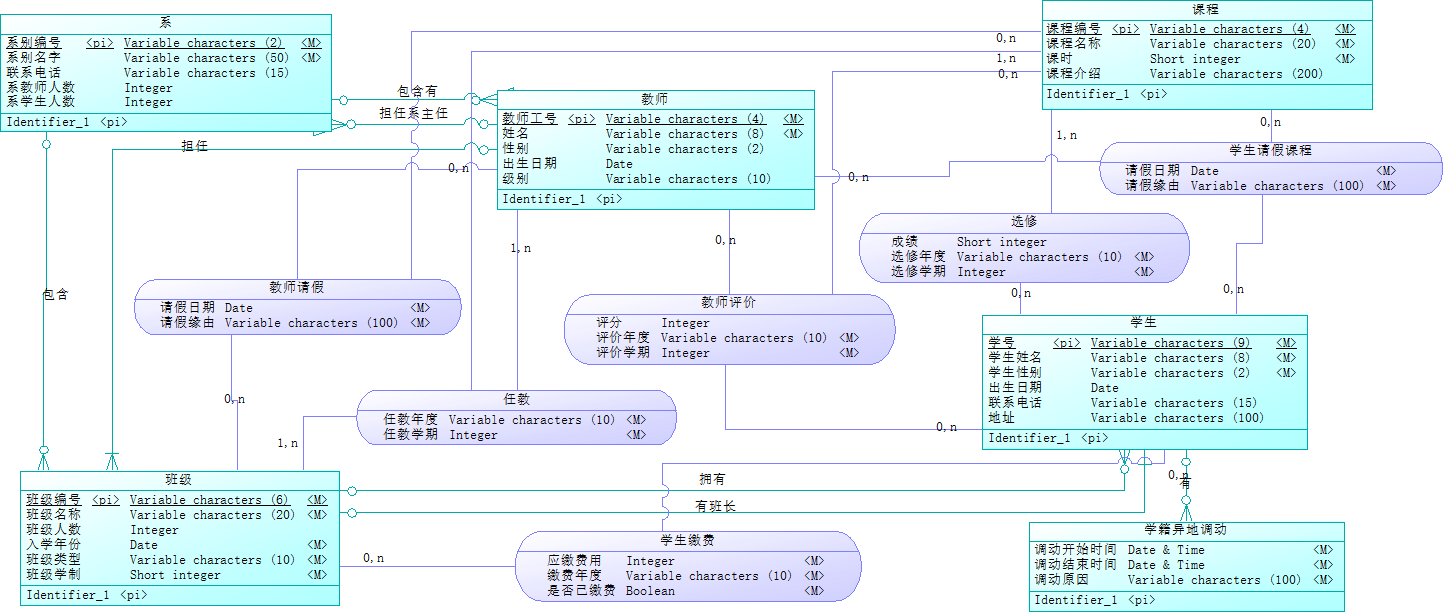
多对多联系:

1. 学生和课程之间的学习联系:一位学生可以学习一门或多门课程，一门课程也可有多位学生学习。
2. 学生和班级之间的缴费关系:一位学生可以缴费多个班级（双学位）,一个班级可以有多个学生缴费。

三元联系:

1. 老师、课程、班级的任教联系:一个班级的一门课程由一位老师任教，一位老师所教的一门课程可归属多个班级，一门课程可在不同班级中由多位老师任教。
2. 课程、教师、学生的请假关系：一个学生可请假多位老师教的多门课程，一个老师的多门课程可以被多位学生请假，一堂课可以被多位教师所教的多位学生请假。
3. 班级、课程、教师的请假关系：一个班级的一门课程可以被一位教师请假，一位教师在某个班级可以请假多门课程，一门课程可以在多个班级被老师请假。
4. 课程、教师、学生的评价关系：一个学生可评价多门课程的老师，一个老师所教的课程可以被多位学生评价，一门课程可以被多位所教老师的学生评价。

2.3 概念模型（ER图）

图四 学生综合测评系统E-R图

说明：图四是根据需求分析和数据流图设计出的E-R图，该概念模型有6个实体（1个弱实体，5个强实体），二元联系当中有6个一对多联系 ，1个一对一联系，2个多对多联系；4个三元联系。

# 3 数据库物理设计

3.1 表的设计

1. 学生表：

（1） 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，学生表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学号是主码，非主属性都完全依赖于学号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该学生表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。

（2） 满足三类完整性规则：

实体完整性：主码学号不为空并且唯一。

参照完整性：学生表有外键班级编号，且班级编号不为空。

用户自定义完整性：设计属性性别为默认值为“男”。

2. 班级表：

（1） 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，班级表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，班级编号是主码，非主属性都完全依赖于班级编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该班级表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性班级名对码的传递依赖，所以符合BC范式。

（2） 满足三类完整性规则：

实体完整性：主码班级编号不为空并且唯一。

参照完整性：有外键系编号，且系编号不为空。

用户自定义完整性：班级名称不能为空。

1. 教师表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，教师表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，教工号是主码，非主属性都完全依赖于教工号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该教师表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。
3. 满足三类完整性规则：

实体完整性：主码教工编号不为空并且唯一。

参照完整性：有外键系编号，且系编号不为空。

用户自定义完整性：教师名称不能为空，设计属性性别为默认值为“男”。

1. 系表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，系表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，系编号是主码，非主属性都完全依赖于系编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该系表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的传递函数依赖，所以满足BC范式。
3. 满足两类完整性规则：

实体完整性：主码系编号不为空并且唯一。

用户自定义完整性：系名称不能为空。

1. 课程表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，课程表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，课程编号是主码，非主属性都完全依赖于课程编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该课程表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的传递函数依赖，所以满足BC范式。
3. 满足两类完整性规则：

实体完整性：课程编号不为空并且唯一。

用户自定义完整性：课程名称不能为空。

1. 选修表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，选修表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学号和课程编号是主码，非主属性都完全依赖于学号和课程编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该选修表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。
3. 满足三类完整性规则：

实体完整性：学号和课程编号不为空并且唯一。

参照完整性：有外键学号和课程编号，且学号和课程编号不为空。

用户自定义完整性：成绩默认值为0

1. 任教表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，任教表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学号，课程和班级编号是主码，非主属性都完全依赖于学号，课程和班级编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该任教表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。
3. 满足两类完整性规则：

实体完整性：学号，课程和班级编号不为空并且唯一。

参照完整性：有外键学号，课程和班级编号，且学号，课程和班级编号不为空。

用户自定义完整性：请假缘由不能为空。

1. 学生课程请假表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，学生课程请假表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学号，课程号和教工编号是主码，非主属性都完全依赖于学号，课程号和教工编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该学生课程请假表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。
3. 满足三类完整性规则：

实体完整性：学号，课程号和教工编号不为空并且唯一。

参照完整性：有外键学号，课程和教工编号，且学号，课程号和教工编号不为空。

1. 教师评价表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，教师评价表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学号，课程号和教工编号是主码，非主属性都完全依赖于学号，课程和教工编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该教师评价表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。

（2） 满足两类完整性规则：

实体完整性：学号，课程号和教工编号不为空并且唯一。

参照完整性：有外键学号，课程号和教工编号，且学号，课程号和教工编号不为空。

1. 教师请假表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，教师请假表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，班级编号、教师工号、请假日期和课程编号是主码，非主属性都完全依赖于班级编号、教师工号、请假日期和课程编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该教师请假表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。
3. 满足两类完整性规则：

实体完整性：班级编号、教师工号、请假日期和课程编号不为空并且唯一。

参照完整性：有外键班级编号、教师工号和课程编号，且班级编号、教师工号和课程编号不为空。

1. 学生请假表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，学生请假表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学生编号、教师工号、请假日期和课程编号是主码，非主属性都完全依赖于学生编号、教师工号、请假日期和课程编号，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该学生请假表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分和传递函数依赖，所以满足BC范式。
3. 满足两类完整性规则：

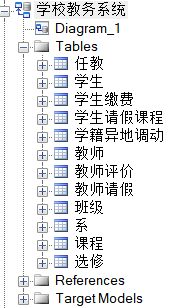
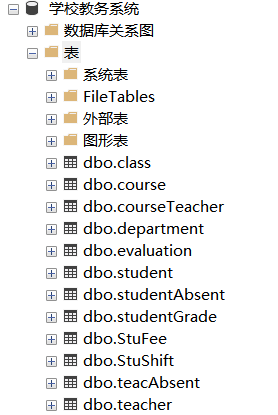
实体完整性：学生编号、教师工号、请假日期和课程编号不为空并且唯一。

参照完整性：有外键学生编号、教师工号和课程编号，且班级编号、学生编号和课程编号不为空。

1. 学籍异地调动表：
2. 满足BC范式：第一范式满足的条件是元组中的每一个分量都必须是不可分割的数据项，学生异地调动表中没有出现重复项，满足第一范式；第二范式要求在第一范式的基础上，要求所有的非主属性完全依赖于其主码，学生编号、调动开始时间、调动结束时间是主码，非主属性都完全依赖于学生编号、调动开始时间、调动结束时间，故满足第二范式；第三范式在第二范式的基础上，任何一个非主属性都不传递依赖于任何主关键字，该学籍异地调动表中的非主属性都不传递依赖于主关键字，故满足第三范式；不存在主属性对码的部分依赖和函数依赖，所以满足BC范式。
3. 满足两类完整性规则：

实体完整性：学生编号、调动开始时间、调动结束时间不为空并且唯一。

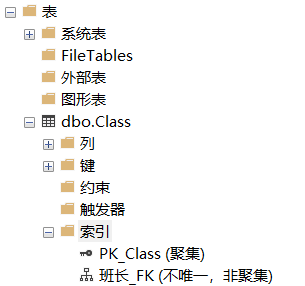
参照完整性：有外键学生编号，且学生编号不为空。

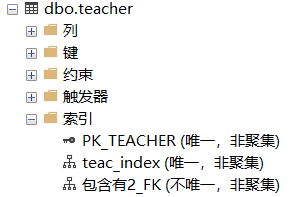
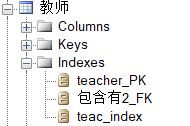
 

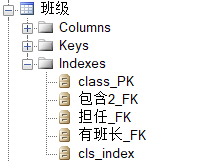
图五 PowerDesigner、SQL中的表

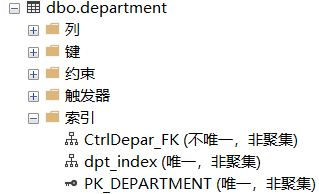
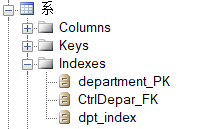
## 3.2 索引的设计

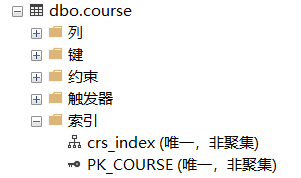
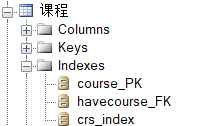
|  |  |
| --- | --- |
| **索引名称** | **索引选择** |
| **课程索引** | 选用了课程编号和课程名称当唯一非聚簇索引 |
| **学生索引** | 选用了学号、班级编号和学生姓名当唯一非聚簇索引 |
| **教师索引** | 选用了教师工号、系别编号和教师姓名当唯一非聚簇索引 |
| **系别索引** | 选用了系别编号和系别名字当唯一非聚簇索引 |
| **班级索引** | 选用了班级编号、班级名称、系别编号当唯一非聚簇索引 |









图六PowerDesigner、SQL中显示的非聚集索引

|  |
| --- |
| **课程索引代码** |
| if exists (select 1  from sysindexes  where id = object\_id('course')  and name = 'crs\_index'  and indid > 0  and indid < 255)  drop index course.crs\_index  go  /\*==============================================================\*/  /\* Index: crs\_index \*/  /\*==============================================================\*/  create unique index crs\_index on course (  course\_id ASC,  course\_name ASC  )  go |

|  |
| --- |
| **学生索引代码** |
| if exists (select 1  from sysindexes  where id = object\_id('student')  and name = 'stu\_index'  and indid > 0  and indid < 255)  drop index student.stu\_index  go  /\*==============================================================\*/  /\* Index: stu\_index \*/  /\*==============================================================\*/  create unique index stu\_index on student (  stu\_id ASC,  class\_id ASC,  stu\_name ASC  )  go |

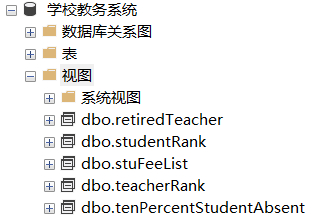
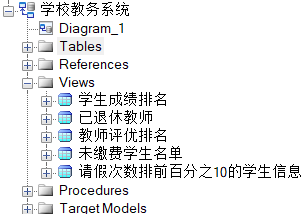
|  |
| --- |
| **教师索引代码** |
| if exists (select 1  from sysindexes  where id = object\_id('teacher')  and name = 'teac\_index'  and indid > 0  and indid < 255)  drop index teacher.teac\_index  go  /\*==============================================================\*/  /\* Index: teac\_index \*/  /\*==============================================================\*/  create unique index teac\_index on teacher (  teac\_id ASC,  depar\_id ASC,  teac\_name ASC  )  go |

|  |
| --- |
| **系别索引代码** |
| if exists (select 1  from sysindexes  where id = object\_id('department')  and name = 'dpt\_index'  and indid > 0  and indid < 255)  drop index department.dpt\_index  go  /\*==============================================================\*/  /\* Index: dpt\_index \*/  /\*==============================================================\*/  create unique index dpt\_index on department (  depar\_id ASC,  depar\_name ASC  )  go |

|  |
| --- |
| **班级索引代码** |
| if exists (select 1  from sysindexes  where id = object\_id('class')  and name = 'cls\_index'  and indid > 0  and indid < 255)  drop index class.cls\_index  go  /\*==============================================================\*/  /\* Index: cls\_index \*/  /\*==============================================================\*/  create unique index cls\_index on class (  class\_id ASC,  depar\_id ASC,  class\_name ASC  )  go |

## 3.3 视图的设计

|  |  |
| --- | --- |
| **视图名称** | **视图功能** |
| **学生成绩排名视图** | 显示全部学生的成绩排名情况(班级,学号,姓名,成绩,排名) |
| **教师评优排名视图** | 显示全部教师的评优排名情况(课程名,姓名,教工号,课程编号,分数,排名) |
| **请假次数排前百分之十学生信息视图** | 显示请假次数排前百分之十的学生，并且输出其请假次数（学生姓名,学号,请假次数） |
| **未交费学生视图（包括是否学籍调动）** | 显示学生的学号，班级号，学生姓名，班级名，应缴费用，是否已缴费，缴费年度，是否学籍调动 |
| **已退休教师视图** | 显示已退休教师的姓名，教工编号（本视图分男女不同退休标准） |



图七 PowerDesigner、SQL中显示的视图

|  |
| --- |
| **学生成绩排名视图代码** |
| if exists (select 1  from sysobjects  where id = object\_id('studentRank')  and type = 'V')  drop view studentRank  go  /\*==============================================================\*/  /\* View: studentRank \*/  /\*==============================================================\*/  create view studentRank (course\_id, class\_id, stu\_name, stu\_id, 成绩, "'排名'", "'年度'", "'学期'") as  select  studentGrade.course\_id,  student.class\_id,  student.stu\_name,  studentGrade.stu\_id,  studentGrade.grade as '分数',  dense\_rank () OVER (PARTITION BY studentGrade.course\_id,selyear,selterm ORDER BY studentGrade.grade desc) as '排名',  selyear as '年度',  selterm as '学期'  from  student,  studentGrade  where  student.stu\_id = studentGrade.stu\_id  go |

|  |
| --- |
| **教师评优排名视图代码** |
| if exists (select 1  from sysobjects  where id = object\_id('teacherRank')  and type = 'V')  drop view teacherRank  go  /\*==============================================================\*/  /\* View: teacherRank \*/  /\*==============================================================\*/  create view teacherRank (course\_name, teac\_name, teac\_id, course\_id, 分数, "'排名'", "'年度'", "'学期'") as  select  course.course\_name,  teacher.teac\_name,  evaluation.teac\_id,  evaluation.course\_id,  avg (evaluation.score) as '分数',  dense\_rank () OVER (Partition by evalyear,evalterm ORDER BY (avg (evaluation.score)) desc) as '排名',  evalyear as '年度',  evalterm as '学期'  from  course,  student,  teacher,  evaluation  where  course.course\_id=evaluation.course\_id  and  student.stu\_id=evaluation.stu\_id  and  teacher.teac\_id=evaluation.teac\_id  Group By  evaluation.teac\_id,  evaluation.course\_id,  course.course\_name,  teacher.teac\_name,  evalyear,  evalterm  go |

|  |
| --- |
| **请假次数排前百分之十学生信息视图代码** |
| if exists (select 1  from sysobjects  where id = object\_id('tenPercentStudentAbsent')  and type = 'V')  drop view tenPercentStudentAbsent  go  /\*==============================================================\*/  /\* View: tenPercentStudentAbsent \*/  /\*==============================================================\*/  create view tenPercentStudentAbsent (stu\_name, stu\_id, count) as  select  top 10 percent  student.stu\_name,  studentAbsent.stu\_id,  count(\*) as '请假次数'  from  student,  studentAbsent  Where  student.stu\_id=studentAbsent.stu\_id  Group By  student.stu\_name,  studentAbsent.stu\_id  Order By count(\*) desc  go |

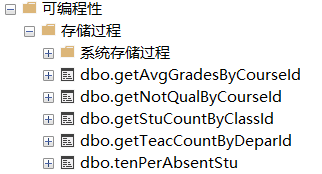
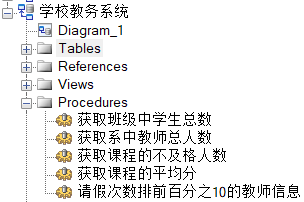
|  |
| --- |
| **未交费学生名单（包括是否学籍调动）视图代码** |
| if exists (select 1  from sysobjects  where id = object\_id('stuFeeList')  and type = 'V')  drop view stuFeeList  go  /\*==============================================================\*/  /\* View: stuFeeList \*/  /\*==============================================================\*/  create view stuFeeList as  select  student.stu\_id,  student.class\_id,  student.stu\_name,  StuFee.stuFee,  StuFee.isTrue,  StuFee.feeYear,  class.class\_name,  dbo.isNotEmpty(StuShift.stu\_id) as '学籍是否调动'  from  student  Join  stufee  On  student.stu\_id=stufee.stu\_id  Join  class  on  class.class\_id=stufee.class\_id  Right outer join  StuShift  On  stuShift.stu\_id=stufee.stu\_id  go |

|  |
| --- |
| **已退休教师视图代码** |
| if exists (select 1  from sysobjects  where id = object\_id('retiredTeacher')  and type = 'V')  drop view retiredTeacher  go  /\*==============================================================\*/  /\* View: retiredTeacher \*/  /\*==============================================================\*/  create view retiredTeacher as  select  teacher.teac\_id,  teacher.teac\_name  from  teacher  where  year(getdate())-year(teacher.birthday) > 55  And teac\_sex = '女'  union  select  teacher.teac\_id,  teacher.teac\_name  from  teacher  where  year(getdate())-year(teacher.birthday) > 60  And teac\_sex = '男'  go |

## 3.4 存储过程的设计

学生综合测评系统中设计了4个存储过程，名称、功能、代码如下表所示。

|  |  |
| --- | --- |
| **存储过程名称** | **存储过程功能** |
| **获取课程的平均分** | 输入所查课程的课程号，即可返回该课程的平均分 |
| **获取系中教师总数** | 输入所查系的编号，即可返回该系的教师总人数 |
| **获取班级中学生总数** | 输入所查班级的编号，即可返回该班级的学生总人数 |
| **获取课程不及格人数** | 输入所查课程的编号，即可返回该课程的不及格人数 |
| **请假次数排前百分之10的教师信息** | 直接可以查询到请假次数排前百分之10的教师信息 |



图八 PowerDesigner、SQL中显示的存储过程

|  |
| --- |
| **获取课程的平均分的存储过程代码** |
| if exists (select 1  from sysobjects  where id = object\_id('getAvgGradesByCourseId')  and type in ('P','PC'))  drop procedure getAvgGradesByCourseId  go  create procedure getAvgGradesByCourseId  @CourseId int  as  begin  select Avg(grade) as '成绩',studentGrade.selyear as '年度' ,studentGrade.selterm as '学期'  from studentGrade  where studentGrade.course\_id=@CourseId  Group By course\_id,selyear,selterm;  end  go |

|  |
| --- |
| **获取班级中学生总数的存储过程代码** |
| if exists (select 1  from sysobjects  where id = object\_id('getStuCountByClassId')  and type in ('P','PC'))  drop procedure getStuCountByClassId  go  create procedure getStuCountByClassId  @classId varchar(6),  @classNum int output  as  begin  select @classNum=count(\*) From Student Where class\_id=@classId  end  go |

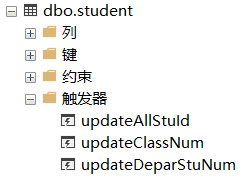
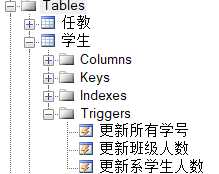
|  |
| --- |
| **获取系中教师总数的存储过程代码** |
| if exists (select 1  from sysobjects  where id = object\_id('getTeacCountByDeparId')  and type in ('P','PC'))  drop procedure getTeacCountByDeparId  go  create procedure getTeacCountByDeparId  @deparId varchar(2),  @teacNum int output  as  begin  select @teacNum=count(\*) From Teacher Where depar\_id=@deparId  end  go |

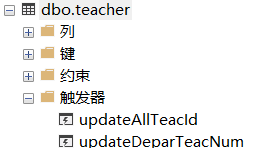
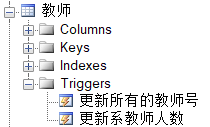
|  |
| --- |
| **获取课程不及格人数的存储过程代码** |
| if exists (select 1  from sysobjects  where id = object\_id('getNotQualByCourseId')  and type in ('P','PC'))  drop procedure getNotQualByCourseId  go  create procedure getNotQualByCourseId  @CourseId int  as  begin  select count(\*) as '人数' ,selyear as '年度' ,selterm as '学期'  from studentGrade  where studentGrade.course\_id=@CourseId  Group By course\_id,selyear,selterm;  end  go |

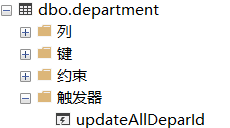
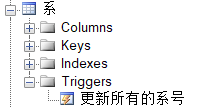
|  |
| --- |
| **请假次数前百分之10的教师信息存储过程代码** |
| if exists (select 1  from sysobjects  where id = object\_id('tenPerAbsentStu')  and type in ('P','PC'))  drop procedure tenPerAbsentStu  go  create procedure tenPerAbsentStu  As  begin  select  top 10 percent  teacher.teac\_name,  teacAbsent.teac\_id,  count(\*) as '请假次数'  from  teacher,  teacAbsent  Where  teacher.teac\_id=teacAbsent.teac\_id  Group By  teacher.teac\_name,  teacAbsent.teac\_id  Order By count(\*) desc  end  go |

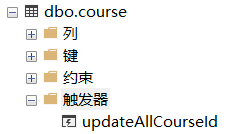
## 3.5 触发器的设计

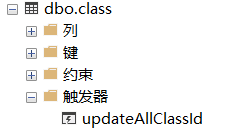
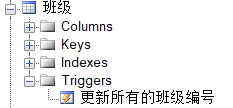
|  |  |
| --- | --- |
| **触发器名称** | **触发器功能** |
| **更新所有的学号** | 若学生表一个学号修改，则更新所有数据表关联的学号 |
| **更新班级的人数** | 若学生表插入一个学生，则更新班级表的总人数 |
| **更新所有的教师号** | 若教师表一个教师号修改，则更新所有数据表关联的教师号 |
| **更新教师的人数** | 若教师表插入一个教师，则更新系表的总人数 |
| **更新系中的学生人数** | 若学生表插入一个学生，则更新系表的总人数 |











图九 PowerDesigner、SQL中显示的触发器

为了防止触发一系列触发器，造成触发器链，所以在触发器内使用判断。

|  |
| --- |
| **更新所有的学号** |
| Create trigger updateAllStuId On student  instead of update  As  Begin  if update(stu\_id)  begin  Alter Table evaluation drop constraint FK\_EVALUATI\_EVALUATIO\_STUDENT;  Alter Table stuShift drop constraint FK\_STUSHIFT\_HAVESHIFT\_STUDENT;  Alter Table stuFee drop constraint FK\_STUFEE\_STUFEE2\_STUDENT;  Alter Table studentAbsent drop constraint FK\_STUDENTA\_STUDENTAB\_STUDENT;  Alter Table studentGrade drop constraint FK\_STUDENTG\_STUDENTGR\_STUDENT;  Alter Table class drop constraint FK\_CLASS\_有班长\_STUDENT;  update student Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);  update stuShift Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);  update stuFee Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);  Update Class Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);  Update evaluation Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);  Update StudentAbsent Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);  Update StudentGrade Set stu\_id=(Select stu\_id From inserted)  Where stu\_id=(Select stu\_id From deleted);    Alter Table evaluation add constraint FK\_EVALUATI\_EVALUATIO\_STUDENT Foreign key(stu\_id) References student(stu\_id);  Alter Table stuShift add constraint FK\_STUSHIFT\_HAVESHIFT\_STUDENT Foreign key(stu\_id) References student(stu\_id);  Alter Table stuFee add constraint FK\_STUFEE\_STUFEE2\_STUDENT Foreign key(stu\_id) References student(stu\_id);  Alter Table studentAbsent add constraint FK\_STUDENTA\_STUDENTAB\_STUDENT Foreign key(stu\_id) References student(stu\_id);  Alter Table studentGrade add constraint FK\_STUDENTG\_STUDENTGR\_STUDENT Foreign key(stu\_id) References student(stu\_id);  Alter Table class add constraint FK\_CLASS\_有班长\_STUDENT Foreign key(stu\_id) References student(stu\_id);  end  if update(stu\_name) update student set stu\_name=(select stu\_name From inserted Where stu\_id=inserted.stu\_id);  if update(stu\_sex) update student set stu\_sex=(select stu\_sex From inserted Where stu\_id=inserted.stu\_id);  if update(birthday) update student set birthday=(select birthday From inserted Where stu\_id=inserted.stu\_id);  if update(phone) update student set phone=(select phone From inserted Where stu\_id=inserted.stu\_id);  if update(address) update student set address=(select address From inserted Where stu\_id=inserted.stu\_id);  end  Go |

|  |
| --- |
| **更新班级的人数** |
| Create trigger updateClassNum On student  After insert,delete  As  Begin  update Class set class\_num=(select count(\*) From student where class\_id=(select class\_id From inserted)) where class\_id=(select class\_id From inserted);  End;  Go |

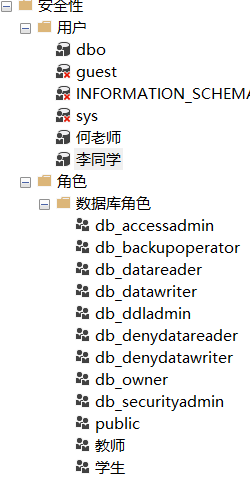
|  |
| --- |
| **更新所有的教师号** |
| Create trigger updateAllTeacId On teacher  instead of update  As  Begin  if update(teac\_id)  begin  Alter Table teacAbsent drop constraint FK\_TEACABSE\_TEACABSEN\_TEACHER;  Alter Table class drop constraint FK\_CLASS\_担任\_TEACHER;  Alter Table evaluation drop constraint FK\_EVALUATI\_EVALUATIO\_TEACHER;  Alter Table StudentAbsent drop constraint FK\_STUDENTA\_STUDENTAB\_TEACHER;  Alter Table courseTeacher drop constraint FK\_COURSETE\_COURSETEA\_TEACHER;  update teacher set teac\_id=(select teac\_id From inserted)  Where teac\_id=(select teac\_id From deleted);  update teacAbsent Set teac\_id=(Select teac\_id From inserted)  Where teac\_id=(Select teac\_id From deleted);  Update class Set teac\_id=(Select teac\_id From inserted)  Where teac\_id=(Select teac\_id From deleted);  Update evaluation Set teac\_id=(Select teac\_id From inserted)  Where teac\_id=(Select teac\_id From deleted);  Update StudentAbsent Set teac\_id=(Select teac\_id From inserted)  Where teac\_id=(Select teac\_id From deleted);  Update courseTeacher Set teac\_id=(Select teac\_id From inserted)  Where teac\_id=(Select teac\_id From deleted);    Alter Table teacAbsent add constraint FK\_TEACABSE\_TEACABSEN\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);  Alter Table class add constraint FK\_CLASS\_担任\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);  Alter Table evaluation add constraint FK\_EVALUATI\_EVALUATIO\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);  Alter Table StudentAbsent add constraint FK\_STUDENTA\_STUDENTAB\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);  Alter Table courseTeacher add constraint FK\_COURSETE\_COURSETEA\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);  end  if update(teac\_name)  update teacher set teac\_name=(select teac\_name From inserted Where teac\_id=teacher.teac\_id);  if update(depar\_id)  update teacher set depar\_id=(select depar\_id From inserted Where teac\_id=teacher.teac\_id);  if update(teac\_sex)  update teacher set teac\_sex=(select teac\_sex From inserted Where teac\_id=teacher.teac\_id);  if update(birthday)  update teacher set birthday=(select birthday From inserted Where teac\_id=teacher.teac\_id);  if update(techpost)  update teacher set techpost=(select techpost From inserted Where teac\_id=teacher.teac\_id);  End;  Go |

|  |
| --- |
| **更新教师的人数** |
| Create trigger updateDeparTeacNum On Department  After insert,delete  As  Begin  update Department set teac\_num=(select count(\*) From Teacher where depar\_id=(select depar\_id From inserted)) where depar\_id=(select depar\_id From inserted);  End;  Go |

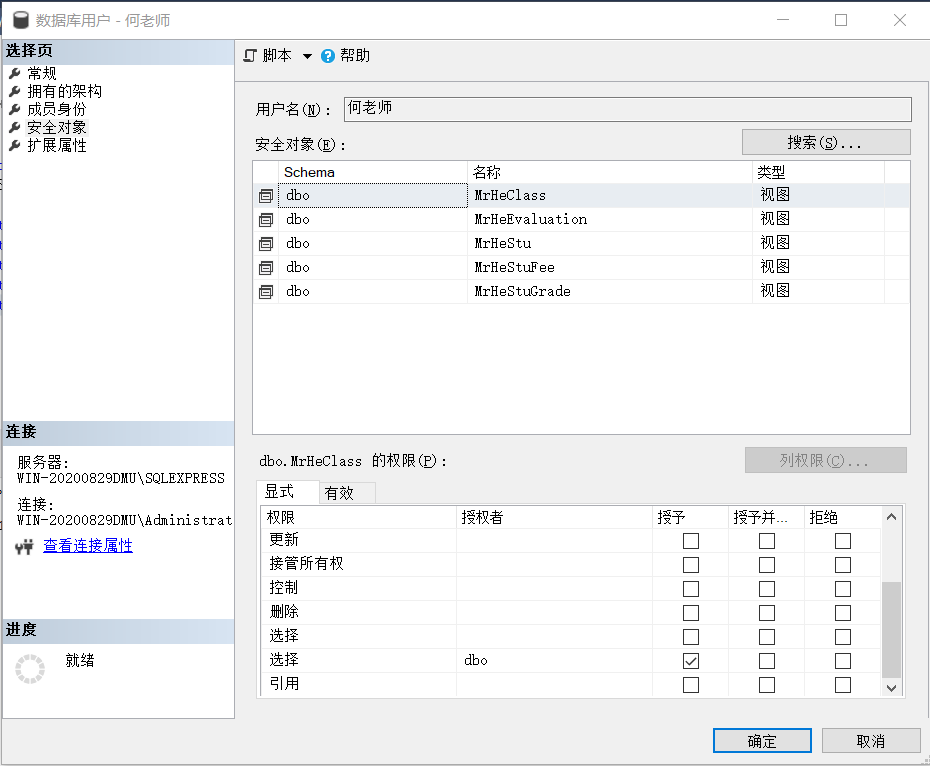
|  |
| --- |
| **更新系学生的人数** |
| Create trigger updateDeparStuNum On student  After insert,delete  As  Begin  update Department set stu\_num=(select count(\*) From student where depar\_id=(select depar\_id From class Where class\_id=(select class\_id From inserted))) where depar\_id=(select depar\_id From class Where class\_id=(select class\_id From inserted));  End;  Go |

## 3.6 角色和用户的设计

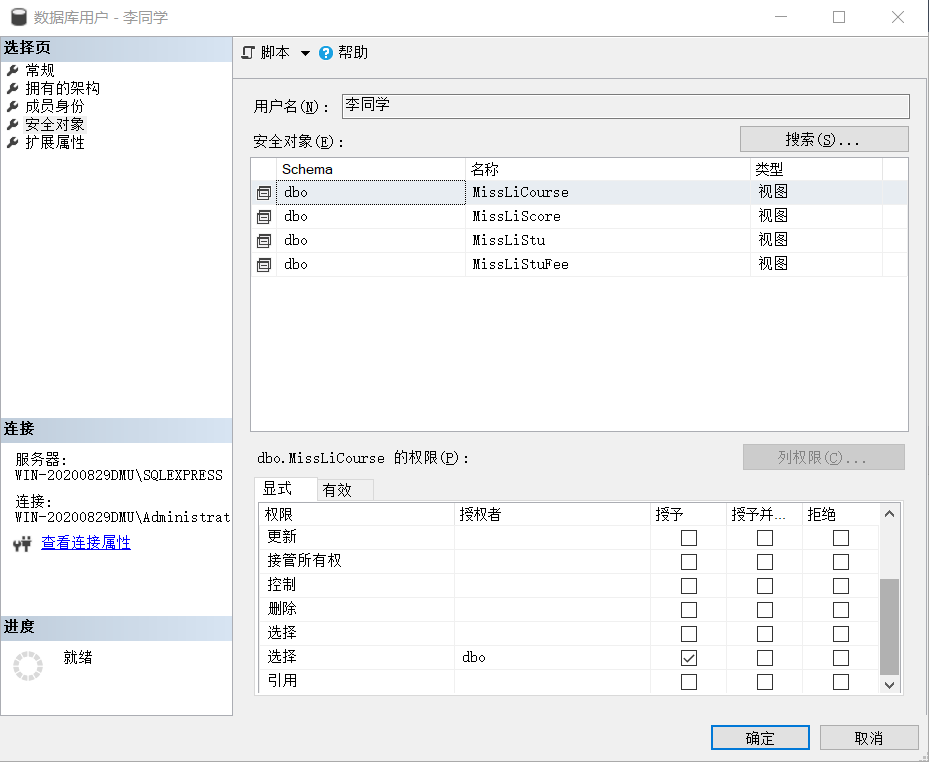
|  |  |
| --- | --- |
| **角色名称** | **用户以及相关功能** |
| **教师** | 其下有何老师的用户，可以查看学生成绩，学生学费，自身评优等等，也可以修改自身相关资料等等 |
| **学生** | 其下有李同学的用户，可以查看自身所教学费，老师授课信息，自身成绩，也可以修改自身相关资料等等 |



图十 SQL中的用户和角色



图十一 查看何老师用户的权限



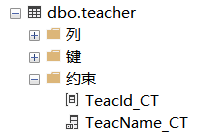
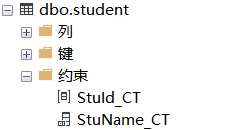
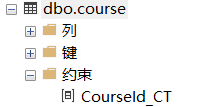
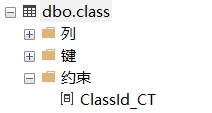
图十二 查看李同学用户的权限

|  |
| --- |
| 何老师的权限代码 |
| 先建视图  Create view MrHeStu  as  select \* From student where class\_id in (select class\_id From class Where teac\_id='0111');  go  Create view MrHeClass  as  Select \* From class where teac\_id='0111';  go  Create view MrHeEvaluation  as  Select evaluation.teac\_id,teac\_name ,evaluation.course\_id,course\_name,evalterm,avg(score) as score  From evaluation,course,teacher  where evaluation.teac\_id='0111' and teacher.teac\_id=evaluation.teac\_id and course.course\_id=evaluation.course\_id  group by evaluation.teac\_id,teac\_name ,evaluation.course\_id,course\_name,evalterm;  Go  Create view MrHeStuGrade  as  select student.stu\_id,stu\_name,course.course\_id,course.course\_name,grade,selyear,selTerm  From student,studentGrade,course  where student.stu\_id=studentGrade.stu\_id and studentGrade.course\_id=course.course\_id and student.class\_id  in (select class\_id From courseTeacher where teac\_id='0111') and studentGrade.course\_id  in (Select course\_id From courseTeacher Where teac\_id='0111');  Go  Create view MrHeTeachCourse  as  select \* From courseTeacher where teac\_id='0111'  Go  Create view MrHeTeacher  as  select \* From Teacher where teac\_id='0111'  Go  Create view MrHeStuFee  as  select \* From stuFee Where class\_id in (Select class\_id From class Where teac\_id='0111');  Go  后赋予权限  Grant select on MrHeEvaluation To 何老师;  Grant select,update(grade) on MrHeStuGrade To 何老师;  Grant select on MrHeStu To 何老师;  Grant select on MrHeClass To 何老师;  Grant select on MrHeStuFee To 何老师;  Grant select,update(teac\_name,teac\_sex,birthday) on MrHeTeacher To 何老师;  Grant select on MrHeTeachCourse To 何老师; |

|  |
| --- |
| 李同学的权限代码 |
| 先建视图  Create View MissLiStu  as  Select \* From student Where stu\_id='190101001'  Go  Create View MissLiScore  as  Select \* From studentGrade Where stu\_id='190101001'  Go  Create View MissLiCourse  as  select \* From course  GO  Create View MissLiStuFee  as  select \* From stuFee Where stu\_id='190101001'  Go  后赋予权限  Grant select,update(stu\_name,stu\_sex,birthday,phone,address) On MissLiStu To 李同学  Grant select On MissLiStuFee To 李同学  Grant select On MissLiCourse To 李同学  Grant select On MissLiScore To 李同学 |

## 3.7 约束的设计

|  |  |
| --- | --- |
| **约束表** | **约束作用说明** |
| **系表** | 系别编号为“01”到“99” |
| **教师表** | 教师编号由所属系别编号和序列号组成，序列号为“01”到“99” |
| **班级表** | 班级编号由入学年份后两位与所属系别编号与序列号组成，序列号为“01”到“99” |
| **课程表** | 课程编号由所属系别编号与序列号组成，序列号为“01”到“99” |
| **学生表** | 学生编号由入学年份的后两位与所属班编号后四位与序列号组成，序列号为“001”到“999” |
| **学生表，教师表** | 学生姓名、教师姓名在一定汉字范围内随机生成 |



图十三 相关约束图

|  |
| --- |
| 系表约束代码 |
| Alter table department with nocheck add constraint DeparId\_CT check(Depar\_id like '[0-9][0-9]' and convert(int,depar\_id) between 1 and 99) ; |

|  |
| --- |
| 教师表约束代码 |
| Alter table teacher with nocheck add constraint TeacId\_CT check(Teac\_id like depar\_id+'[0-9][0-9]' and convert(int,subString(teac\_id,3,2)) between 1 and 99) ; |

|  |
| --- |
| 班级表约束代码 |
| Alter table class with nocheck add constraint ClassId\_CT check(class\_id like substring(convert(varchar(4),datepart(yy,enroll\_year)),3,2)+depar\_id+'[0-9][0-9]'and convert(int,subString(class\_id,5,2)) between 1 and 99) ; |

|  |
| --- |
| 课程表约束代码 |
| Alter table course with nocheck add constraint CourseId\_CT check(Course\_id like depar\_id+'[0-9][0-9]' and convert(int,subString(course\_id,3,2)) between 1 and 99) ; |

|  |
| --- |
| 学生表约束代码 |
| Alter table student with nocheck add constraint StuId\_CT check(Stu\_id like class\_id+'[0-9][0-9][0-9]' and convert(int,subString(stu\_id,7,3)) between 1 and 999) ; |

|  |
| --- |
| 学生,教师表随机名约束代码以及总结 |
| 第一步建姓名随机表  Create table firstName(  chinese\_id int identity primary key,  chinese\_str char(4)  );  Create table lastName(  chinese1\_id int identity primary key,  chinese1\_str char(4)  );  Insert into firstName values('陈'),('李'),('钟'),('许'),('谢'),('朱'),('吴'),('冯'),('蒋'),('王'),('何'),('吕'),('秦'),('潘'),('余');  Insert into lastName values('学'),('炫'),('俊'),('霖'),('伟'),('峰'),('志'),('桥'),('篮'),('子'),('继'),('杰'),('嘉'),('骏'),('浩'),('然');  第二步建随机数视图  Create View MidRand  As  select Round(Rand()\*(15-1)+1,0) AS Num;  Go  Create View MidRand1  As  select Round(Rand()\*(16-1)+1,0) AS Num;  Go  第三步建随机获取姓名函数  Create Function getName()  Returns varchar(6)  As  Begin  Declare @one varchar(2);  Declare @two varchar(2);  Declare @three varchar(2);  select @one=chinese\_str from firstName Where chinese\_id=(Select num From MidRand);  select @two=chinese1\_str from lastName Where chinese1\_id=(Select num From MidRand1);  select @three=chinese1\_str from lastName Where chinese1\_id=(Select num From MidRand1);  return @one+@two+@three;  End  GO  第四步建约束  Alter Table student Add constraint StuName\_CT default dbo.getName() for stu\_name;  Alter Table teacher Add constraint TeacName\_CT default dbo.getName() for teac\_name;  总结  为何建函数？  因为在约束中default里面不能使用查询表达句，只能使用标量函数。  为何建视图？  因为在函数中不允许使用rand随机数的函数，所以只能建视图，用视图当中转站。 |

# 4 数据库实施

## 4.1 代码的生成

1.课程表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('course') and o.name = 'FK\_COURSE\_HAVECOURS\_DEPARTME')

alter table course

drop constraint FK\_COURSE\_HAVECOURS\_DEPARTME

go

if exists (select 1

from sysindexes

where id = object\_id('course')

and name = 'crs\_index'

and indid > 0

and indid < 255)

drop index course.crs\_index

go

if exists (select 1

from sysindexes

where id = object\_id('course')

and name = 'havecourse\_FK'

and indid > 0

and indid < 255)

drop index course.havecourse\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('course')

and type = 'U')

drop table course

go

/\*==============================================================\*/

/\* Table: course \*/

/\*==============================================================\*/

create table course (

course\_id varchar(4) not null,

depar\_id varchar(2) null,

course\_name varchar(20) not null,

course\_hour smallint not null,

intoduce varchar(200) null,

constraint PK\_COURSE primary key nonclustered (course\_id)

)

go

/\*==============================================================\*/

/\* Index: havecourse\_FK \*/

/\*==============================================================\*/

create index havecourse\_FK on course (

depar\_id ASC

)

go

/\*==============================================================\*/

/\* Index: crs\_index \*/

/\*==============================================================\*/

create unique index crs\_index on course (

course\_id ASC,

course\_name ASC

)

go

alter table course

add constraint FK\_COURSE\_HAVECOURS\_DEPARTME foreign key (depar\_id)

references department (depar\_id)

go

1. 学生表

if exists (select 1

from sysobjects

where id = object\_id('updateAllStuId')

and type = 'TR')

drop trigger updateAllStuId

go

if exists (select 1

from sysobjects

where id = object\_id('updateClassNum')

and type = 'TR')

drop trigger updateClassNum

go

if exists (select 1

from sysobjects

where id = object\_id('updateDeparStuNum')

and type = 'TR')

drop trigger updateDeparStuNum

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('student') and o.name = 'FK\_STUDENT\_拥有\_CLASS')

alter table student

drop constraint FK\_STUDENT\_拥有\_CLASS

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('student') and o.name = 'FK\_STUDENT\_有班长2\_CLASS')

alter table student

drop constraint FK\_STUDENT\_有班长2\_CLASS

go

if exists (select 1

from sysindexes

where id = object\_id('student')

and name = 'stu\_index'

and indid > 0

and indid < 255)

drop index student.stu\_index

go

if exists (select 1

from sysindexes

where id = object\_id('student')

and name = '有班长2\_FK'

and indid > 0

and indid < 255)

drop index student.有班长2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('student')

and name = '拥有\_FK'

and indid > 0

and indid < 255)

drop index student.拥有\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('student')

and type = 'U')

drop table student

go

/\*==============================================================\*/

/\* Table: student \*/

/\*==============================================================\*/

create table student (

stu\_id varchar(9) not null,

class\_id varchar(6) null,

stu\_name varchar(8) not null,

stu\_sex varchar(2) not null,

birthday date null,

phone varchar(15) null,

address varchar(100) null,

constraint PK\_STUDENT primary key nonclustered (stu\_id)

)

go

/\*==============================================================\*/

/\* Index: 拥有\_FK \*/

/\*==============================================================\*/

create index 拥有\_FK on student (

class\_id ASC

)

go

/\*==============================================================\*/

/\* Index: stu\_index \*/

/\*==============================================================\*/

create unique index stu\_index on student (

stu\_id ASC,

class\_id ASC,

stu\_name ASC

)

go

alter table student

add constraint FK\_STUDENT\_拥有\_CLASS foreign key (class\_id)

references class (class\_id)

go

Create trigger updateAllStuId On student

instead of update

as

if update(stu\_id)

begin

Alter Table evaluation drop constraint FK\_EVALUATI\_EVALUATIO\_STUDENT;

Alter Table stuShift drop constraint FK\_STUSHIFT\_HAVESHIFT\_STUDENT;

Alter Table stuFee drop constraint FK\_STUFEE\_STUFEE2\_STUDENT;

Alter Table studentAbsent drop constraint FK\_STUDENTA\_STUDENTAB\_STUDENT;

Alter Table studentGrade drop constraint FK\_STUDENTG\_STUDENTGR\_STUDENT;

Alter Table class drop constraint FK\_CLASS\_有班长\_STUDENT;

update student Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

update stuShift Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

update stuFee Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

Update Class Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

Update evaluation Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

Update StudentAbsent Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

Update StudentGrade Set stu\_id=(Select stu\_id From inserted)

Where stu\_id=(Select stu\_id From deleted);

Alter Table evaluation add constraint FK\_EVALUATI\_EVALUATIO\_STUDENT Foreign key(stu\_id) References student(stu\_id);

Alter Table stuShift add constraint FK\_STUSHIFT\_HAVESHIFT\_STUDENT Foreign key(stu\_id) References student(stu\_id);

Alter Table stuFee add constraint FK\_STUFEE\_STUFEE2\_STUDENT Foreign key(stu\_id) References student(stu\_id);

Alter Table studentAbsent add constraint FK\_STUDENTA\_STUDENTAB\_STUDENT Foreign key(stu\_id) References student(stu\_id);

Alter Table studentGrade add constraint FK\_STUDENTG\_STUDENTGR\_STUDENT Foreign key(stu\_id) References student(stu\_id);

Alter Table class add constraint FK\_CLASS\_有班长\_STUDENT Foreign key(stu\_id) References student(stu\_id);

end

if update(stu\_name) update student set stu\_name=(select stu\_name From inserted Where stu\_id=inserted.stu\_id);

if update(stu\_sex) update student set stu\_sex=(select stu\_sex From inserted Where stu\_id=inserted.stu\_id);

if update(birthday) update student set birthday=(select birthday From inserted Where stu\_id=inserted.stu\_id);

if update(phone) update student set phone=(select phone From inserted Where stu\_id=inserted.stu\_id);

if update(address) update student set address=(select address From inserted Where stu\_id=inserted.stu\_id);

go

Create trigger updateClassNum On student

For insert,delete

As

Begin

update Class set class\_num=(select count(\*) From student where class\_id=(select class\_id From inserted)) where class\_id=(select class\_id From inserted);

End;

go

Create trigger updateDeparStuNum On student

After insert,delete

As

Begin

update Department set stu\_num=(select count(\*) From student where depar\_id=(select depar\_id From class Where class\_id=(select class\_id From inserted))) where depar\_id=(select depar\_id From class Where class\_id=(select class\_id From inserted));

End;

go

1. 教师表

if exists (select 1

from sysobjects

where id = object\_id('updateAllTeacId')

and type = 'TR')

drop trigger updateAllTeacId

go

if exists (select 1

from sysobjects

where id = object\_id('updateDeparTeacNum')

and type = 'TR')

drop trigger updateDeparTeacNum

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('teacher') and o.name = 'FK\_TEACHER\_包含有2\_DEPARTME')

alter table teacher

drop constraint FK\_TEACHER\_包含有2\_DEPARTME

go

if exists (select 1

from sysindexes

where id = object\_id('teacher')

and name = 'teac\_index'

and indid > 0

and indid < 255)

drop index teacher.teac\_index

go

if exists (select 1

from sysindexes

where id = object\_id('teacher')

and name = '包含有2\_FK'

and indid > 0

and indid < 255)

drop index teacher.包含有2\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('teacher')

and type = 'U')

drop table teacher

go

/\*==============================================================\*/

/\* Table: teacher \*/

/\*==============================================================\*/

create table teacher (

teac\_id varchar(4) not null,

depar\_id varchar(2) null,

teac\_name varchar(8) not null,

teac\_sex varchar(2) null,

birthday datetime null,

techPost varchar(10) null,

constraint PK\_TEACHER primary key nonclustered (teac\_id)

)

go

/\*==============================================================\*/

/\* Index: 包含有2\_FK \*/

/\*==============================================================\*/

create index 包含有2\_FK on teacher (

depar\_id ASC

)

go

/\*==============================================================\*/

/\* Index: teac\_index \*/

/\*==============================================================\*/

create unique index teac\_index on teacher (

teac\_id ASC,

depar\_id ASC,

teac\_name ASC

)

go

alter table teacher

add constraint FK\_TEACHER\_包含有2\_DEPARTME foreign key (depar\_id)

references department (depar\_id)

go

Create trigger updateAllTeacId On teacher

instead of update

As

Begin

if update(teac\_id)

begin

Alter Table teacAbsent drop constraint FK\_TEACABSE\_TEACABSEN\_TEACHER;

Alter Table class drop constraint FK\_CLASS\_担任\_TEACHER;

Alter Table evaluation drop constraint FK\_EVALUATI\_EVALUATIO\_TEACHER;

Alter Table StudentAbsent drop constraint FK\_STUDENTA\_STUDENTAB\_TEACHER;

Alter Table courseTeacher drop constraint FK\_COURSETE\_COURSETEA\_TEACHER;

update teacher set teac\_id=(select teac\_id From inserted)

Where teac\_id=(select teac\_id From deleted);

update teacAbsent Set teac\_id=(Select teac\_id From inserted)

Where teac\_id=(Select teac\_id From deleted);

Update class Set teac\_id=(Select teac\_id From inserted)

Where teac\_id=(Select teac\_id From deleted);

Update evaluation Set teac\_id=(Select teac\_id From inserted)

Where teac\_id=(Select teac\_id From deleted);

Update StudentAbsent Set teac\_id=(Select teac\_id From inserted)

Where teac\_id=(Select teac\_id From deleted);

Update courseTeacher Set teac\_id=(Select teac\_id From inserted)

Where teac\_id=(Select teac\_id From deleted);

Alter Table teacAbsent add constraint FK\_TEACABSE\_TEACABSEN\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);

Alter Table class add constraint FK\_CLASS\_担任\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);

Alter Table evaluation add constraint FK\_EVALUATI\_EVALUATIO\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);

Alter Table StudentAbsent add constraint FK\_STUDENTA\_STUDENTAB\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);

Alter Table courseTeacher add constraint FK\_COURSETE\_COURSETEA\_TEACHER Foreign key(teac\_id) References teacher(teac\_id);

end

if update(teac\_name)

update teacher set teac\_name=(select teac\_name From inserted Where teac\_id=teacher.teac\_id);

if update(depar\_id)

update teacher set depar\_id=(select depar\_id From inserted Where teac\_id=teacher.teac\_id);

if update(teac\_sex)

update teacher set teac\_sex=(select teac\_sex From inserted Where teac\_id=teacher.teac\_id);

if update(birthday)

update teacher set birthday=(select birthday From inserted Where teac\_id=teacher.teac\_id);

if update(techpost)

update teacher set techpost=(select techpost From inserted Where teac\_id=teacher.teac\_id);

End;

go

Create trigger updateDeparTeacNum On Teacher

After insert,delete

As

Begin

update Department set teac\_num=(select count(\*) From Teacher where depar\_id=(select depar\_id From inserted)) where depar\_id=(select depar\_id From inserted);

End;

go

1. 系表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('department') and o.name = 'FK\_DEPARTME\_CTRLDEPAR\_TEACHER')

alter table department

drop constraint FK\_DEPARTME\_CTRLDEPAR\_TEACHER

go

if exists (select 1

from sysindexes

where id = object\_id('department')

and name = 'dpt\_index'

and indid > 0

and indid < 255)

drop index department.dpt\_index

go

if exists (select 1

from sysindexes

where id = object\_id('department')

and name = 'CtrlDepar\_FK'

and indid > 0

and indid < 255)

drop index department.CtrlDepar\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('department')

and type = 'U')

drop table department

go

/\*==============================================================\*/

/\* Table: department \*/

/\*==============================================================\*/

create table department (

depar\_id varchar(2) not null,

teac\_id varchar(4) null,

depar\_name varchar(50) not null,

phone varchar(15) null,

teac\_num int null,

stu\_num int null,

constraint PK\_DEPARTMENT primary key nonclustered (depar\_id)

)

go

/\*==============================================================\*/

/\* Index: CtrlDepar\_FK \*/

/\*==============================================================\*/

create index CtrlDepar\_FK on department (

teac\_id ASC

)

go

/\*==============================================================\*/

/\* Index: dpt\_index \*/

/\*==============================================================\*/

create unique index dpt\_index on department (

depar\_id ASC,

depar\_name ASC

)

go

alter table department

add constraint FK\_DEPARTME\_CTRLDEPAR\_TEACHER foreign key (teac\_id)

references teacher (teac\_id)

go

1. 班级表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('class') and o.name = 'FK\_CLASS\_包含2\_DEPARTME')

alter table class

drop constraint FK\_CLASS\_包含2\_DEPARTME

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('class') and o.name = 'FK\_CLASS\_担任\_TEACHER')

alter table class

drop constraint FK\_CLASS\_担任\_TEACHER

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('class') and o.name = 'FK\_CLASS\_有班长\_STUDENT')

alter table class

drop constraint FK\_CLASS\_有班长\_STUDENT

go

if exists (select 1

from sysindexes

where id = object\_id('class')

and name = 'cls\_index'

and indid > 0

and indid < 255)

drop index class.cls\_index

go

if exists (select 1

from sysindexes

where id = object\_id('class')

and name = '有班长\_FK'

and indid > 0

and indid < 255)

drop index class.有班长\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('class')

and name = '担任\_FK'

and indid > 0

and indid < 255)

drop index class.担任\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('class')

and name = '包含2\_FK'

and indid > 0

and indid < 255)

drop index class.包含2\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('class')

and type = 'U')

drop table class

go

/\*==============================================================\*/

/\* Table: class \*/

/\*==============================================================\*/

create table class (

class\_id varchar(6) not null,

teac\_id varchar(4) null,

stu\_id varchar(9) null,

depar\_id varchar(2) null,

class\_name varchar(20) not null,

class\_num int null,

enroll\_year datetime not null,

class\_genre varchar(10) not null,

class\_dur smallint not null,

constraint PK\_CLASS primary key nonclustered (class\_id)

)

go

/\*==============================================================\*/

/\* Index: 包含2\_FK \*/

/\*==============================================================\*/

create index 包含2\_FK on class (

depar\_id ASC

)

go

/\*==============================================================\*/

/\* Index: 担任\_FK \*/

/\*==============================================================\*/

create index 担任\_FK on class (

teac\_id ASC

)

go

/\*==============================================================\*/

/\* Index: 有班长\_FK \*/

/\*==============================================================\*/

create index 有班长\_FK on class (

stu\_id ASC

)

go

/\*==============================================================\*/

/\* Index: cls\_index \*/

/\*==============================================================\*/

create unique index cls\_index on class (

class\_id ASC,

depar\_id ASC,

class\_name ASC

)

go

alter table class

add constraint FK\_CLASS\_包含2\_DEPARTME foreign key (depar\_id)

references department (depar\_id)

go

alter table class

add constraint FK\_CLASS\_担任\_TEACHER foreign key (teac\_id)

references teacher (teac\_id)

go

alter table class

add constraint FK\_CLASS\_有班长\_STUDENT foreign key (stu\_id)

references student (stu\_id)

go

1. 老师请假表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('teacAbsent') and o.name = 'FK\_TEACABSE\_TEACABSEN\_COURSE')

alter table teacAbsent

drop constraint FK\_TEACABSE\_TEACABSEN\_COURSE

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('teacAbsent') and o.name = 'FK\_TEACABSE\_TEACABSEN\_CLASS')

alter table teacAbsent

drop constraint FK\_TEACABSE\_TEACABSEN\_CLASS

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('teacAbsent') and o.name = 'FK\_TEACABSE\_TEACABSEN\_TEACHER')

alter table teacAbsent

drop constraint FK\_TEACABSE\_TEACABSEN\_TEACHER

go

if exists (select 1

from sysindexes

where id = object\_id('teacAbsent')

and name = 'teacAbsent3\_FK'

and indid > 0

and indid < 255)

drop index teacAbsent.teacAbsent3\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('teacAbsent')

and name = 'teacAbsent2\_FK'

and indid > 0

and indid < 255)

drop index teacAbsent.teacAbsent2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('teacAbsent')

and name = 'teacAbsent\_FK'

and indid > 0

and indid < 255)

drop index teacAbsent.teacAbsent\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('teacAbsent')

and type = 'U')

drop table teacAbsent

go

/\*==============================================================\*/

/\* Table: teacAbsent \*/

/\*==============================================================\*/

create table teacAbsent (

course\_id varchar(4) not null,

class\_id varchar(6) not null,

teac\_id varchar(4) not null,

absentDate datetime not null,

reason varchar(100) not null,

constraint PK\_TEACABSENT primary key (course\_id, class\_id, teac\_id, absentDate)

)

go

/\*==============================================================\*/

/\* Index: teacAbsent\_FK \*/

/\*==============================================================\*/

create index teacAbsent\_FK on teacAbsent (

course\_id ASC

)

go

/\*==============================================================\*/

/\* Index: teacAbsent2\_FK \*/

/\*==============================================================\*/

create index teacAbsent2\_FK on teacAbsent (

class\_id ASC

)

go

/\*==============================================================\*/

/\* Index: teacAbsent3\_FK \*/

/\*==============================================================\*/

create index teacAbsent3\_FK on teacAbsent (

teac\_id ASC

)

go

alter table teacAbsent

add constraint FK\_TEACABSE\_TEACABSEN\_COURSE foreign key (course\_id)

references course (course\_id)

go

alter table teacAbsent

add constraint FK\_TEACABSE\_TEACABSEN\_CLASS foreign key (class\_id)

references class (class\_id)

go

alter table teacAbsent

add constraint FK\_TEACABSE\_TEACABSEN\_TEACHER foreign key (teac\_id)

references teacher (teac\_id)

go

1. 学生请假表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('studentAbsent') and o.name = 'FK\_STUDENTA\_STUDENTAB\_STUDENT')

alter table studentAbsent

drop constraint FK\_STUDENTA\_STUDENTAB\_STUDENT

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('studentAbsent') and o.name = 'FK\_STUDENTA\_STUDENTAB\_COURSE')

alter table studentAbsent

drop constraint FK\_STUDENTA\_STUDENTAB\_COURSE

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('studentAbsent') and o.name = 'FK\_STUDENTA\_STUDENTAB\_TEACHER')

alter table studentAbsent

drop constraint FK\_STUDENTA\_STUDENTAB\_TEACHER

go

if exists (select 1

from sysindexes

where id = object\_id('studentAbsent')

and name = 'studentAbsent3\_FK'

and indid > 0

and indid < 255)

drop index studentAbsent.studentAbsent3\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('studentAbsent')

and name = 'studentAbsent2\_FK'

and indid > 0

and indid < 255)

drop index studentAbsent.studentAbsent2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('studentAbsent')

and name = 'studentAbsent\_FK'

and indid > 0

and indid < 255)

drop index studentAbsent.studentAbsent\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('studentAbsent')

and type = 'U')

drop table studentAbsent

go

/\*==============================================================\*/

/\* Table: studentAbsent \*/

/\*==============================================================\*/

create table studentAbsent (

stu\_id varchar(9) not null,

course\_id varchar(4) not null,

teac\_id varchar(4) not null,

absentDate datetime not null,

reason varchar(100) not null,

constraint PK\_STUDENTABSENT primary key (stu\_id, course\_id, teac\_id, absentDate)

)

go

/\*==============================================================\*/

/\* Index: studentAbsent\_FK \*/

/\*==============================================================\*/

create index studentAbsent\_FK on studentAbsent (

stu\_id ASC

)

go

/\*==============================================================\*/

/\* Index: studentAbsent2\_FK \*/

/\*==============================================================\*/

create index studentAbsent2\_FK on studentAbsent (

course\_id ASC

)

go

/\*==============================================================\*/

/\* Index: studentAbsent3\_FK \*/

/\*==============================================================\*/

create index studentAbsent3\_FK on studentAbsent (

teac\_id ASC

)

go

alter table studentAbsent

add constraint FK\_STUDENTA\_STUDENTAB\_STUDENT foreign key (stu\_id)

references student (stu\_id)

go

alter table studentAbsent

add constraint FK\_STUDENTA\_STUDENTAB\_COURSE foreign key (course\_id)

references course (course\_id)

go

alter table studentAbsent

add constraint FK\_STUDENTA\_STUDENTAB\_TEACHER foreign key (teac\_id)

references teacher (teac\_id)

go

1. 选修表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('studentGrade') and o.name = 'FK\_STUDENTG\_STUDENTGR\_STUDENT')

alter table studentGrade

drop constraint FK\_STUDENTG\_STUDENTGR\_STUDENT

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('studentGrade') and o.name = 'FK\_STUDENTG\_STUDENTGR\_COURSE')

alter table studentGrade

drop constraint FK\_STUDENTG\_STUDENTGR\_COURSE

go

if exists (select 1

from sysindexes

where id = object\_id('studentGrade')

and name = 'studentGrade2\_FK'

and indid > 0

and indid < 255)

drop index studentGrade.studentGrade2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('studentGrade')

and name = 'studentGrade\_FK'

and indid > 0

and indid < 255)

drop index studentGrade.studentGrade\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('studentGrade')

and type = 'U')

drop table studentGrade

go

/\*==============================================================\*/

/\* Table: studentGrade \*/

/\*==============================================================\*/

create table studentGrade (

stu\_id varchar(9) not null,

course\_id varchar(4) not null,

grade smallint null,

selYear varchar(10) not null,

selTerm int not null,

constraint PK\_STUDENTGRADE primary key (stu\_id, course\_id, selYear, selTerm)

)

go

/\*==============================================================\*/

/\* Index: studentGrade\_FK \*/

/\*==============================================================\*/

create index studentGrade\_FK on studentGrade (

stu\_id ASC

)

go

/\*==============================================================\*/

/\* Index: studentGrade2\_FK \*/

/\*==============================================================\*/

create index studentGrade2\_FK on studentGrade (

course\_id ASC

)

go

alter table studentGrade

add constraint FK\_STUDENTG\_STUDENTGR\_STUDENT foreign key (stu\_id)

references student (stu\_id)

go

alter table studentGrade

add constraint FK\_STUDENTG\_STUDENTGR\_COURSE foreign key (course\_id)

references course (course\_id)

go

1. 学籍调动表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('StuShift') and o.name = 'FK\_STUSHIFT\_HAVESHIFT\_STUDENT')

alter table StuShift

drop constraint FK\_STUSHIFT\_HAVESHIFT\_STUDENT

go

if exists (select 1

from sysindexes

where id = object\_id('StuShift')

and name = 'haveshifting\_FK'

and indid > 0

and indid < 255)

drop index StuShift.haveshifting\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('StuShift')

and type = 'U')

drop table StuShift

go

/\*==============================================================\*/

/\* Table: StuShift \*/

/\*==============================================================\*/

create table StuShift (

shift\_start datetime not null,

shift\_end datetime not null,

stu\_id varchar(9) not null,

shift\_reason varchar(100) not null,

constraint PK\_STUSHIFT primary key (shift\_start, shift\_end)

)

go

/\*==============================================================\*/

/\* Index: haveshifting\_FK \*/

/\*==============================================================\*/

create index haveshifting\_FK on StuShift (

stu\_id ASC

)

go

alter table StuShift

add constraint FK\_STUSHIFT\_HAVESHIFT\_STUDENT foreign key (stu\_id)

references student (stu\_id)

go

1. 学生缴费表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('StuFee') and o.name = 'FK\_STUFEE\_STUFEE\_CLASS')

alter table StuFee

drop constraint FK\_STUFEE\_STUFEE\_CLASS

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('StuFee') and o.name = 'FK\_STUFEE\_STUFEE2\_STUDENT')

alter table StuFee

drop constraint FK\_STUFEE\_STUFEE2\_STUDENT

go

if exists (select 1

from sysindexes

where id = object\_id('StuFee')

and name = 'StuFee2\_FK'

and indid > 0

and indid < 255)

drop index StuFee.StuFee2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('StuFee')

and name = 'StuFee\_FK'

and indid > 0

and indid < 255)

drop index StuFee.StuFee\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('StuFee')

and type = 'U')

drop table StuFee

go

/\*==============================================================\*/

/\* Table: StuFee \*/

/\*==============================================================\*/

create table StuFee (

class\_id varchar(6) not null,

stu\_id varchar(9) not null,

stuFee int not null,

feeYear varchar(10) not null,

isTrue bit not null,

constraint PK\_STUFEE primary key (class\_id, stu\_id, feeYear)

)

go

/\*==============================================================\*/

/\* Index: StuFee\_FK \*/

/\*==============================================================\*/

create index StuFee\_FK on StuFee (

class\_id ASC

)

go

/\*==============================================================\*/

/\* Index: StuFee2\_FK \*/

/\*==============================================================\*/

create index StuFee2\_FK on StuFee (

stu\_id ASC

)

go

alter table StuFee

add constraint FK\_STUFEE\_STUFEE\_CLASS foreign key (class\_id)

references class (class\_id)

go

alter table StuFee

add constraint FK\_STUFEE\_STUFEE2\_STUDENT foreign key (stu\_id)

references student (stu\_id)

go

1. 任教表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('courseTeacher') and o.name = 'FK\_COURSETE\_COURSETEA\_TEACHER')

alter table courseTeacher

drop constraint FK\_COURSETE\_COURSETEA\_TEACHER

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('courseTeacher') and o.name = 'FK\_COURSETE\_COURSETEA\_COURSE')

alter table courseTeacher

drop constraint FK\_COURSETE\_COURSETEA\_COURSE

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('courseTeacher') and o.name = 'FK\_COURSETE\_COURSETEA\_CLASS')

alter table courseTeacher

drop constraint FK\_COURSETE\_COURSETEA\_CLASS

go

if exists (select 1

from sysindexes

where id = object\_id('courseTeacher')

and name = 'courseTeacher3\_FK'

and indid > 0

and indid < 255)

drop index courseTeacher.courseTeacher3\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('courseTeacher')

and name = 'courseTeacher2\_FK'

and indid > 0

and indid < 255)

drop index courseTeacher.courseTeacher2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('courseTeacher')

and name = 'courseTeacher\_FK'

and indid > 0

and indid < 255)

drop index courseTeacher.courseTeacher\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('courseTeacher')

and type = 'U')

drop table courseTeacher

go

/\*==============================================================\*/

/\* Table: courseTeacher \*/

/\*==============================================================\*/

create table courseTeacher (

teac\_id varchar(4) not null,

course\_id varchar(4) not null,

class\_id varchar(6) not null,

TeacYear varchar(10) not null,

TeacTerm int not null,

constraint PK\_COURSETEACHER primary key (teac\_id, course\_id, class\_id, TeacYear)

)

go

/\*==============================================================\*/

/\* Index: courseTeacher\_FK \*/

/\*==============================================================\*/

create index courseTeacher\_FK on courseTeacher (

teac\_id ASC

)

go

/\*==============================================================\*/

/\* Index: courseTeacher2\_FK \*/

/\*==============================================================\*/

create index courseTeacher2\_FK on courseTeacher (

course\_id ASC

)

go

/\*==============================================================\*/

/\* Index: courseTeacher3\_FK \*/

/\*==============================================================\*/

create index courseTeacher3\_FK on courseTeacher (

class\_id ASC

)

go

alter table courseTeacher

add constraint FK\_COURSETE\_COURSETEA\_TEACHER foreign key (teac\_id)

references teacher (teac\_id)

go

alter table courseTeacher

add constraint FK\_COURSETE\_COURSETEA\_COURSE foreign key (course\_id)

references course (course\_id)

go

alter table courseTeacher

add constraint FK\_COURSETE\_COURSETEA\_CLASS foreign key (class\_id)

references class (class\_id)

go

1. 教师评价表

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('evaluation') and o.name = 'FK\_EVALUATI\_EVALUATIO\_STUDENT')

alter table evaluation

drop constraint FK\_EVALUATI\_EVALUATIO\_STUDENT

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('evaluation') and o.name = 'FK\_EVALUATI\_EVALUATIO\_TEACHER')

alter table evaluation

drop constraint FK\_EVALUATI\_EVALUATIO\_TEACHER

go

if exists (select 1

from sys.sysreferences r join sys.sysobjects o on (o.id = r.constid and o.type = 'F')

where r.fkeyid = object\_id('evaluation') and o.name = 'FK\_EVALUATI\_EVALUATIO\_COURSE')

alter table evaluation

drop constraint FK\_EVALUATI\_EVALUATIO\_COURSE

go

if exists (select 1

from sysindexes

where id = object\_id('evaluation')

and name = 'evaluation3\_FK'

and indid > 0

and indid < 255)

drop index evaluation.evaluation3\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('evaluation')

and name = 'evaluation2\_FK'

and indid > 0

and indid < 255)

drop index evaluation.evaluation2\_FK

go

if exists (select 1

from sysindexes

where id = object\_id('evaluation')

and name = 'evaluation\_FK'

and indid > 0

and indid < 255)

drop index evaluation.evaluation\_FK

go

if exists (select 1

from sysobjects

where id = object\_id('evaluation')

and type = 'U')

drop table evaluation

go

/\*==============================================================\*/

/\* Table: evaluation \*/

/\*==============================================================\*/

create table evaluation (

stu\_id varchar(9) not null,

teac\_id varchar(4) not null,

course\_id varchar(4) not null,

score int null,

evalYear varchar(10) not null,

evalTerm int not null,

constraint PK\_EVALUATION primary key (stu\_id, teac\_id, course\_id, evalYear, evalTerm)

)

go

/\*==============================================================\*/

/\* Index: evaluation\_FK \*/

/\*==============================================================\*/

create index evaluation\_FK on evaluation (

stu\_id ASC

)

go

/\*==============================================================\*/

/\* Index: evaluation2\_FK \*/

/\*==============================================================\*/

create index evaluation2\_FK on evaluation (

teac\_id ASC

)

go

/\*==============================================================\*/

/\* Index: evaluation3\_FK \*/

/\*==============================================================\*/

create index evaluation3\_FK on evaluation (

course\_id ASC

)

go

alter table evaluation

add constraint FK\_EVALUATI\_EVALUATIO\_STUDENT foreign key (stu\_id)

references student (stu\_id)

go

alter table evaluation

add constraint FK\_EVALUATI\_EVALUATIO\_TEACHER foreign key (teac\_id)

references teacher (teac\_id)

go

alter table evaluation

add constraint FK\_EVALUATI\_EVALUATIO\_COURSE foreign key (course\_id)

references course (course\_id)

go

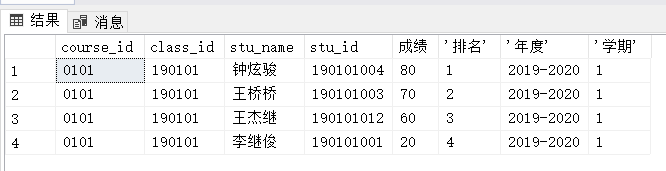
## 4.2 相关内容的改进

1.对表中相关的数据类型长度进行修改，例如学号改为varchar(9)，这是由于最后的约束的限制，所以不得不修改。

2.添加了一些比较符合常规的表，增添该系统的多样性。

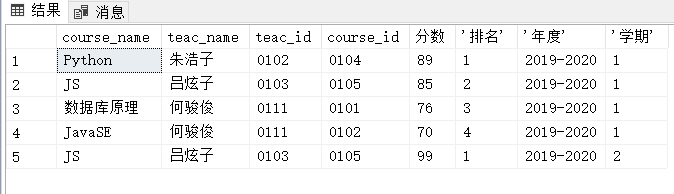
## 4.3 数据测试

### 4.3.1视图的测试



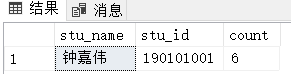
图十四 某一课程的学生成绩排名

|  |
| --- |
| 测试的视图名 |
| **学生成绩排名视图** |
| 测试代码 |
| select \* From studentRank Where course\_id='0101'; |
| 说明 |
| 该视图是某一课程的学生成绩排名的视图，该视图对course\_id分组排名（支持并列排名），输出的是course\_id,class\_id,stu\_name,stu\_id,成绩,排名,年度，学期，这样通过输入course\_id即可查询到该门课程某学期的学生的成绩排名（按学年学期分组排名，学年学期不同不能混排） |



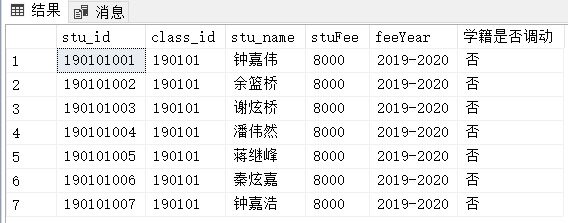
图十五 每门课程的教师评优排名

|  |
| --- |
| 测试的视图名 |
| **教师评优排名视图** |
| 测试代码 |
| select \* From teacherRank; |
| 说明 |
| 该视图是每门课程的教师评优排名，教师评优排名是通过学生对其评价的分数的平均进行排名，最后输出course\_name,teac\_name,teac\_id,course\_id,分数,排名，排名，年度，学期。（按学年学期分组排名，学年学期不同不能混排） |



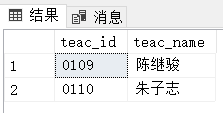
图十六 请假次数排前百分之十的学生信息

|  |
| --- |
| 测试的视图名 |
| **请假次数排前百分之十学生信息视图** |
| 测试代码 |
| select \* From tenPercentStudentAbsent; |
| 说明 |
| 该视图是请假次数排前百分之10的学生信息视图，该视图通过对学生请假次数进行排名，取前百分之10的学生，最后输出stu\_name,stu\_id,count,这里count就是他们的请假次数。 |



图十七 未交费学生视图信息

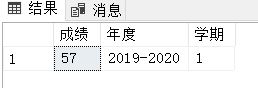
|  |
| --- |
| 测试的视图名 |
| **未交费学生视图（包括是否学籍调动，学籍调动是否过期）** |
| 测试代码 |
| select \* From stuFeeList; |
| 说明 |
| 该视图是未交费学生视图，该视图输出学生未交费的学生，其中有一点就是学籍调动的时候，是有日期的，有的是永久的，有的是暂时的，那么就要判断他是否过了这个最后期限，如果过了，那么它这一次就应该交费，否则就不用交费。最后输出的是stu\_id,class\_id,stu\_name.stuFee,feeYear,学籍是否调动（有期限）。 |



图十八 已退休教师信息

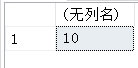
|  |
| --- |
| 测试的视图名 |
| **已退休教师视图** |
| 测试代码 |
| select \* From retiredTeacher; |
| 说明 |
| 该视图是已退休教师视图，该视图主要就是输出已退休的教师视图，其中我国政策规定男的达到60岁退休，女的达到55岁即可退休，所以该视图也按照该政策进行输出相关符合规定的退休教师，输出teac\_id和teac\_name |

### 4.3.2存储过程测试



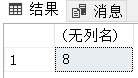
图十九 某一课程的学生成绩排名

|  |
| --- |
| 测试的存储过程名 |
| **获取课程的平均分** |
| 测试代码 |
| exec getAvgGradesByCourseId '0101' |
| 说明 |
| 该存储过程是查询某一课程的平均成绩，通过输入参数course\_id即可输出查询到的平均成绩，以及学年度，学期 |



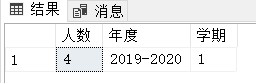
图二十 某系中的教师总数

|  |
| --- |
| 测试的存储过程名 |
| **获取系中教师总数** |
| 测试代码 |
| declare @ret1 int  exec getAvgGradesByCourseId '0101',@ret1 output  select @ret1 |
| 说明 |
| 该存储过程是查询某一系中的教师总数，通过输入参数depar\_id即可输出查询该系教师总数 |



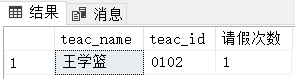
图二十一 某班级中学生总数

|  |
| --- |
| 测试的存储过程名 |
| **获取班级中学生总数** |
| 测试代码 |
| declare @ret3 int  exec getStuCountByClassId '190101',@ret3 output  select @ret3 |
| 说明 |
| 该存储过程是获取班级中学生总数，通过输入参数class\_id即可输出查询该班级学生总数 |



图二十二 某课程中不及格人数

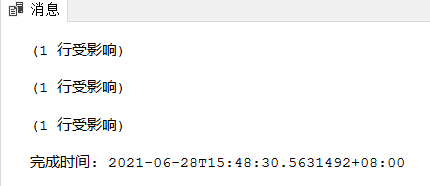
|  |
| --- |
| 测试的存储过程名 |
| **获取课程不及格人数** |
| 测试代码 |
| exec getNotQualByCourseId '0101' |
| 说明 |
| 该存储过程是获取课程不及格人数,通过输入参数course\_id即可输出查询该课程不及格人数，以及学年度和学期 |



图二十三 请假次数排前百分之10的教师信息

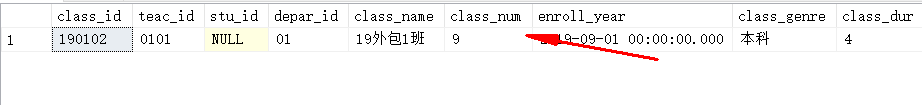
|  |
| --- |
| 测试的存储过程名 |
| **请假次数排前百分之10的教师信息** |
| 测试代码 |
| exec tenPerAbsentTeac |
| 说明 |
| 该存储过程是获取请假次数排前百分之10的教师信息，直接存储过程内部输出结果集。 |

### 4.3.3触发器测试



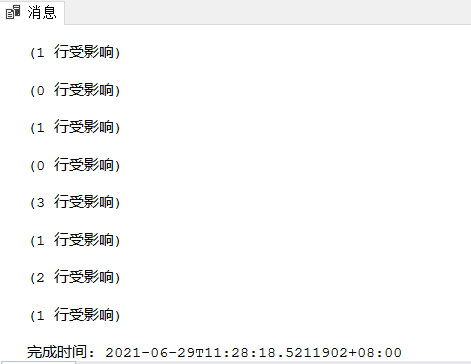
图二十四 插入学生后返回插入消息

图二十五 查询该班级学生的返回结果

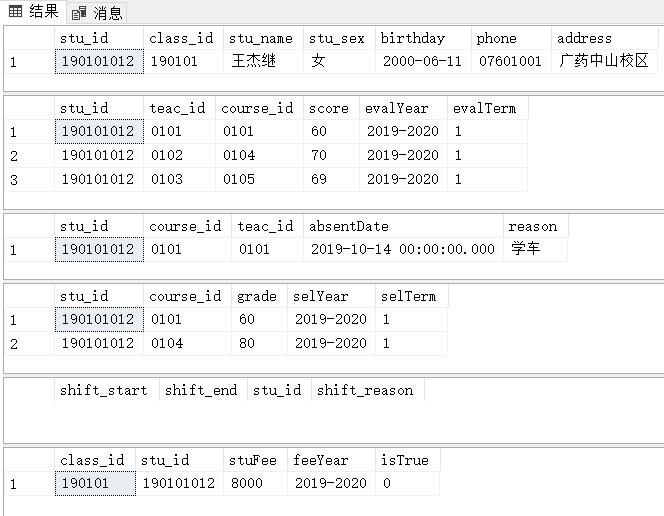


图二十六 查询班级的返回结果

|  |
| --- |
| 测试的触发器名称 |
| **更新班级的人数** |
| 测试代码 |
| insert into student(stu\_id,stu\_name,class\_id,stu\_sex,birthday,phone,address) values('190102009','柏拉图','190102','女','2000-06-10','07601011','广药中山校区');  select \* From student where class\_id=190102;  select \* From class where class\_id=190102; |
| 说明 |
| 该触发器作用是刷新班级人数，在定义时规定了在插入和删除student表的时候，将class表中的class的class\_num重新计算。 |

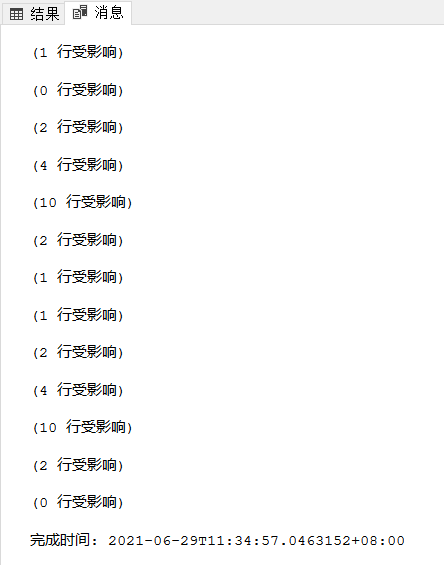


图二十七 执行更新修改学生学号的返回信息

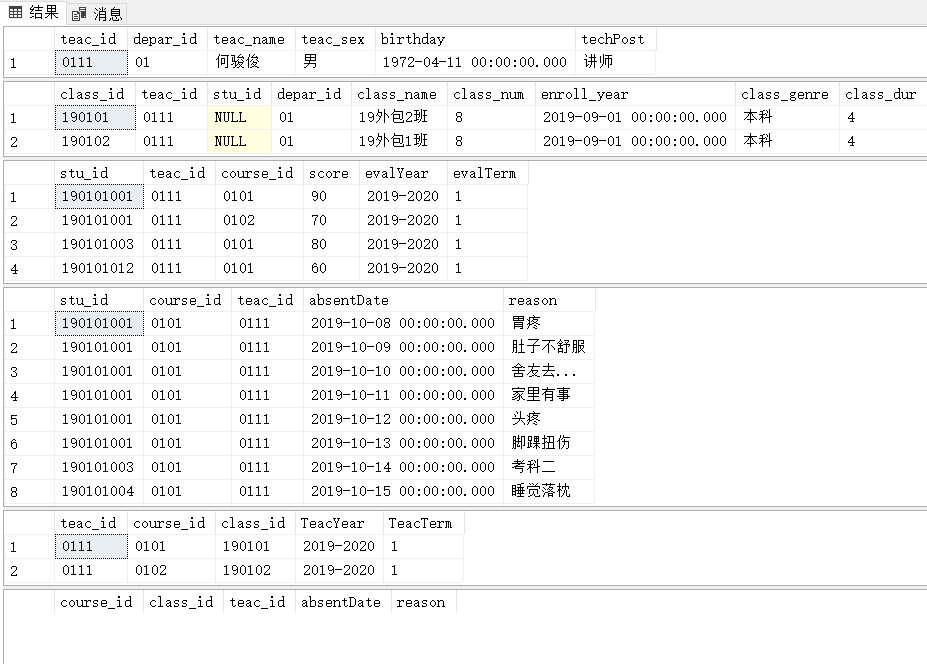


图二十八 执行查询修改后各个表的学生的返回结果

|  |
| --- |
| 测试的触发器名称 |
| **更新所有的学号** |
| 测试代码 |
| update student set stu\_id='190101012' where stu\_id='190101002';  select \* From student Where stu\_id='190101012';  select \* From evaluation Where stu\_id='190101012';  select \* From studentAbsent where stu\_id='190101012';  select \* From studentGrade where stu\_id='190101012';  select \* From stuShift where stu\_id='190101012';  select \* From stuFee where stu\_id='190101012';  select \* From class where stu\_id='190101012'; |
| 说明 |
| 该触发器作用是修改全体的学号，由于表定义了外键，所以不能直接update所有的外键，应当先删后再改，改后再增，其次由于不支持before触发器，所以该触发器采用insteadof触发器 |

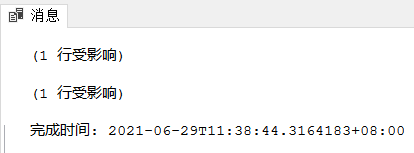


图二十九 执行修改教师工号后的返回消息

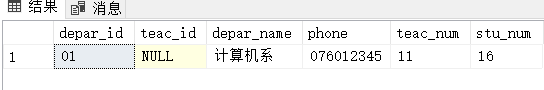


图三十 执行查询各个表中的相关教师信息

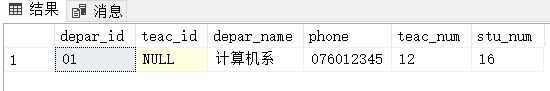
|  |
| --- |
| 测试的触发器名称 |
| **更新所有的教师号** |
| 测试代码 |
| update teacher set teac\_id='0111' where teac\_id='0101';  select \* From teacher where teac\_id='0111';  select \* From class where teac\_id='0111';  select \* From evaluation where teac\_id='0111';  select \* From studentAbsent where teac\_id='0111';  select \* From courseTeacher where teac\_id='0111';  select \* From teacAbsent where teac\_id='0111'; |
| 说明 |
| 该触发器作用是更新所有的教师号，教师号的修改方法跟修改学号的方法类似，这里不再赘述。 |



图三十一 执行插入教师后的返回消息

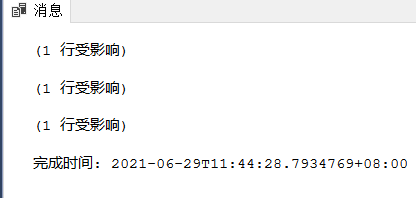


图三十二 执行插入教师前的系表

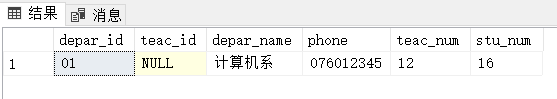


图三十三 执行插入教师后的系表

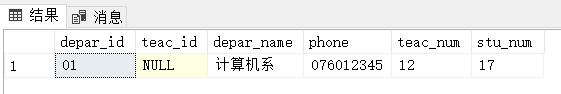
|  |
| --- |
| 测试的触发器名称 |
| **更新系中教师的人数** |
| 测试代码 |
| insert into teacher(teac\_id,depar\_id,teac\_sex,birthday,techPost) values('0120','01','男','1972-04-11 00:00:00.000','讲师');  select \* From department where depar\_id='01'; |
| 说明 |
| 该触发器作用是更新系中的教师人数，该修改方法跟修改班级中的人数类似，这里不再赘述。 |



图三十四 执行修改系号后的返回消息



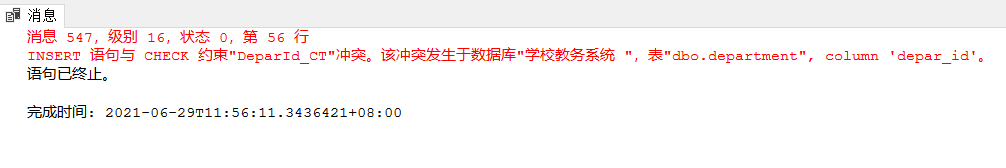
图三十五 执行插入学生前的系信息



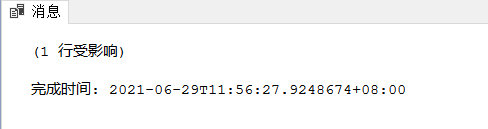
图三十六 执行插入学生前的系信息

|  |
| --- |
| 测试的触发器名称 |
| **修改系中的学生人数** |
| 测试代码 |
| insert into student(stu\_id,class\_id,stu\_sex,birthday,phone,address) values('190101020','190101','女','2000-06-10','07601000','广药中山校区');  select \* From department where depar\_id='01'; |
| 说明 |
| 该触发器作用是修改系中的学生人数，该修改方法跟修改系中教师人数的方法类似，这里不再赘述。 |

### 4.3.4约束测试



图三十七 执行插入第一个系后的返回消息

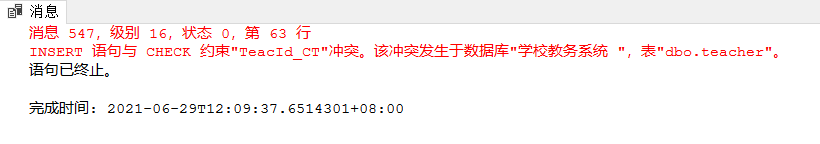


图三十八 执行插入第二个系后的返回信息

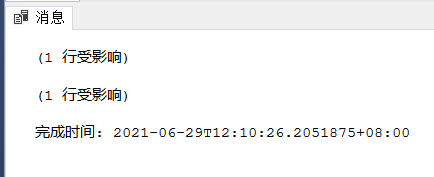


图三十九 执行查询系的返回的结果

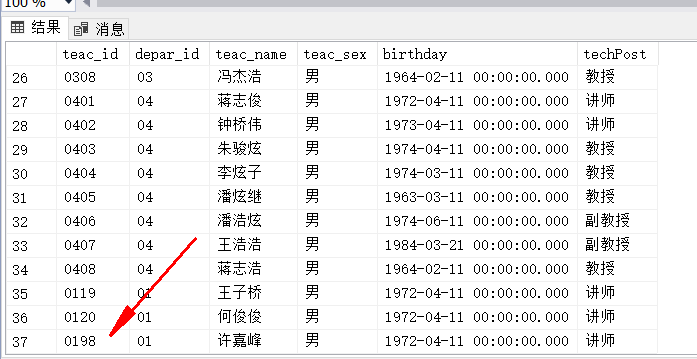
|  |
| --- |
| 测试的约束内容 |
| **系别编号为“01”到“99”** |
| 测试代码 |
| insert into department(depar\_id,teac\_id,depar\_name,phone) values('00',null,'计算机系','076012345');  insert into department(depar\_id,teac\_id,depar\_name,phone) values('19',null,'计算机系','076012345');  select \* From department; |
| 说明 |
| 该约束是系别编号在01到99之间，代码使用了like判断和强制转换类型的函数 |



图四十 执行插入第一个教师后的返回消息



图四十一 执行插入第二个教师后的返回信息

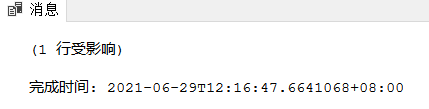


图四十二 执行查询教师的返回的结果

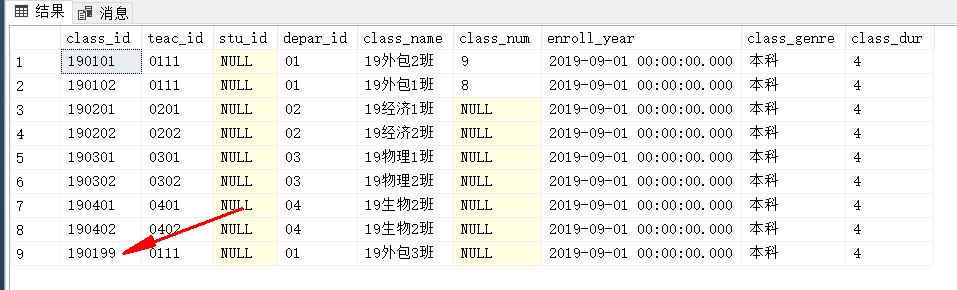
|  |
| --- |
| 测试的约束内容 |
| **教师编号由所属系别编号和序列号组成，序列号为“01”到“99”** |
| 测试代码 |
| insert into teacher(teac\_id,depar\_id,teac\_sex,birthday,techPost) values('0100','01','男','1972-04-11 00:00:00.000','讲师');  insert into teacher(teac\_id,depar\_id,teac\_sex,birthday,techPost) values('0198','01','男','1972-04-11 00:00:00.000','讲师');  select \* From teacher; |
| 说明 |
| 该约束是教师编号由所属系别编号和序列号组成，序列号为“01”到“99”，代码使用了like判断和强制转换类型和取子串的函数 |



图四十三 执行插入第一个班级后的返回消息

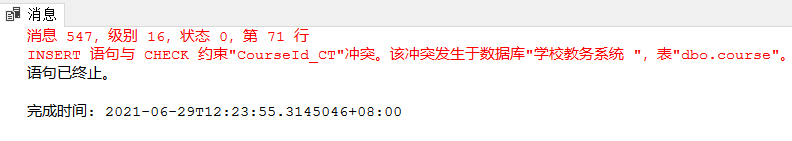


图四十四 执行插入第二个班级后的返回信息

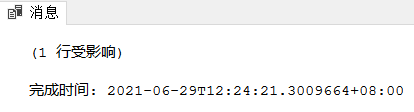


图四十五 执行查询教师的返回的结果

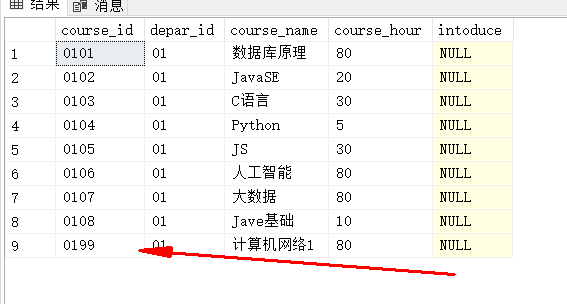
|  |
| --- |
| 测试的约束内容 |
| **班级编号由入学年份后两位与所属系别编号与序列号组成，序列号为“01”到“99”** |
| 测试代码 |
| insert into class(class\_id,teac\_id,depar\_id,class\_name,enroll\_year,class\_genre,class\_dur) values('190100','0111','01','19外包3班','2019-09-01','本科','4');  insert into class(class\_id,teac\_id,depar\_id,class\_name,enroll\_year,class\_genre,class\_dur) values('190199','0111','01','19外包3班','2019-09-01','本科','4');  select \* From class; |
| 说明 |
| 该约束是班级编号由入学年份后两位与所属系别编号与序列号组成，序列号为“01”到“99”，代码使用了like判断，强制转换类型和取子串的函数 |



图四十六 执行插入第一个课程后的返回消息



图四十七 执行插入第二个课程后的返回信息

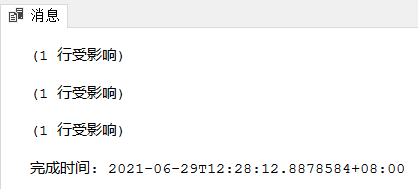


图四十八 执行查询课程的返回的结果

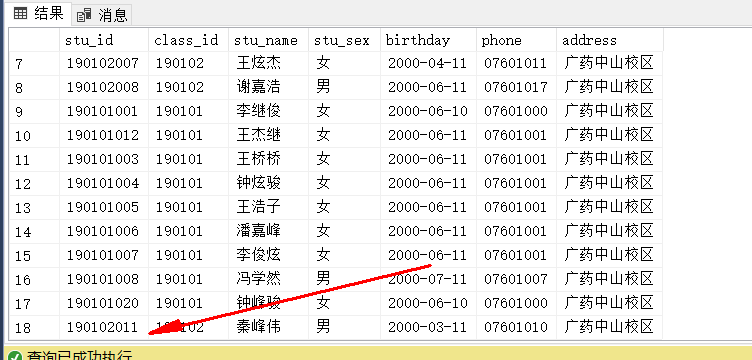
|  |
| --- |
| 测试的约束内容 |
| **课程编号由所属系别编号与序列号组成，序列号为“01”到“99”** |
| 测试代码 |
| insert into course(course\_id,depar\_id,course\_name,course\_hour) values('0100','01','计算机网络1','80');  insert into course(course\_id,depar\_id,course\_name,course\_hour) values('0199','01','计算机网络1','80');  select \* From course; |
| 说明 |
| 该约束是课程编号由所属系别编号与序列号组成，序列号为“01”到“99”代码使用了like判断，强制转换类型和取子串的函数 |



图四十九 执行插入第一个学生后的返回消息

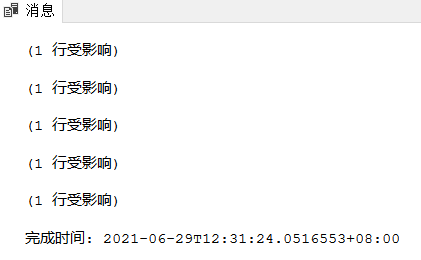


图五十 执行插入第二个学生后的返回信息

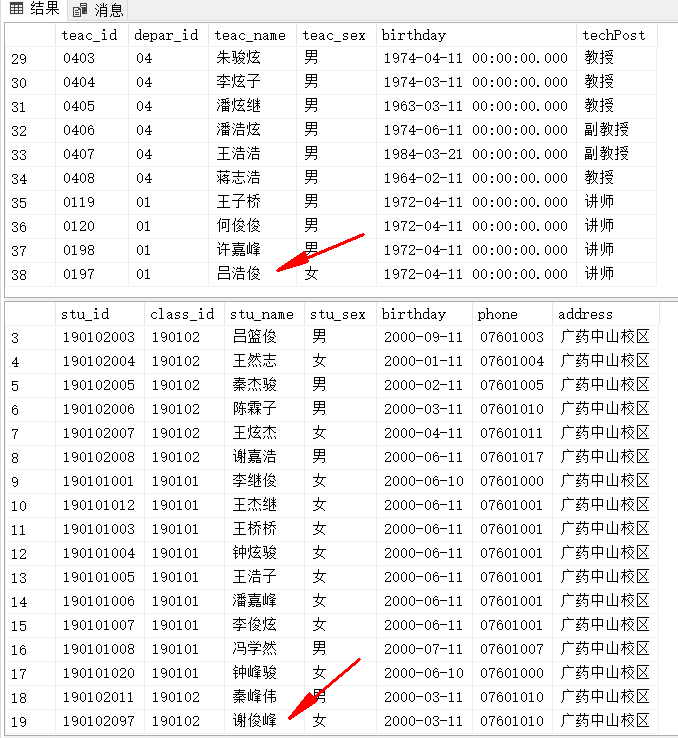


图五十一 执行查询学生的返回的结果

|  |
| --- |
| 测试的约束内容 |
| **学生编号由入学年份的后两位与所属班编号后四位与序列号组成，序列号为“001”到“999”** |
| 测试代码 |
| insert into student(stu\_id,class\_id,stu\_sex,birthday,phone,address) values('190102000','190102','男','2000-03-11','07601010','广药中山校区');  insert into student(stu\_id,class\_id,stu\_sex,birthday,phone,address) values('190102011','190102','男','2000-03-11','07601010','广药中山校区');  select \* From student; |
| 说明 |
| 该约束是学生编号由入学年份的后两位与所属班编号后四位与序列号组成，序列号为“001”到“999”代码使用了like判断，强制转换类型和取子串的函数 |



图五十二 执行插入学生和教师后的返回消息

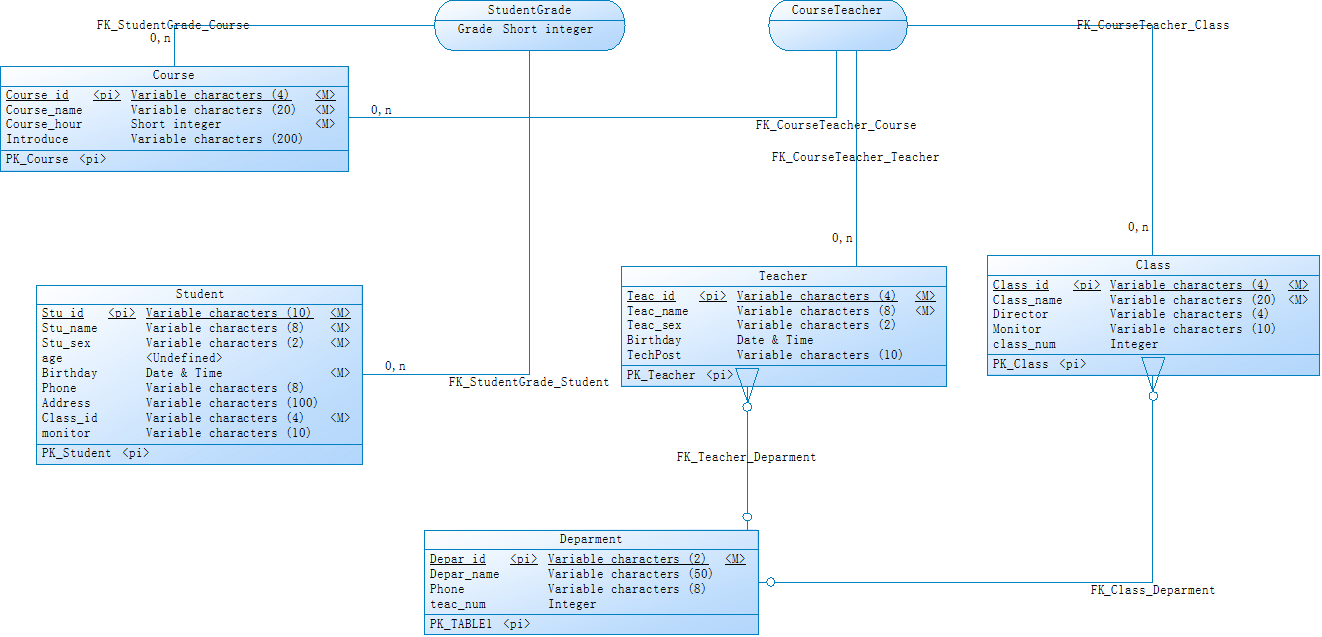


图五十三 执行查询教师、学生的返回的结果

|  |
| --- |
| 测试的约束内容 |
| **学生姓名、教师姓名在一定汉字范围内随机生成** |
| 测试代码 |
| insert into teacher(teac\_id,depar\_id,teac\_sex,birthday,techPost) values('0197','01','女','1972-04-11 00:00:00.000','讲师');  insert into student(stu\_id,class\_id,stu\_sex,birthday,phone,address) values('190102097','190102','女','2000-03-11','07601010','广药中山校区');  select \* From teacher;  select \* From student; |
| 说明 |
| 该约束是学生姓名、教师姓名在一定汉字范围内随机生成，首先建两个表，表中分别放一些可以取姓和名的汉字，然后在约束中加入即可（通过视图和函数进行转换） |

# 5 School数据库逆向生成ER图

**School逆向生成概念模型：**



**School物理模型:**

