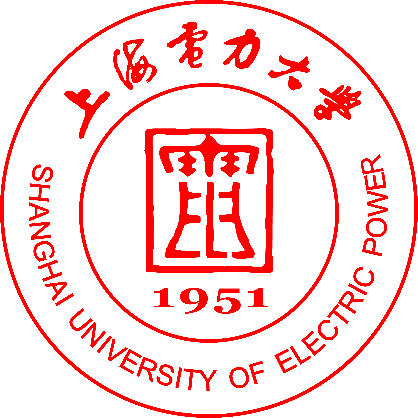
上海电力大学

数据库应用课程设计



**题　　目:**

**学生姓名：**

**学 号： 班 级：**

**院　　系：　　 计算机科学与技术学院**

**专业班级：**

**年　　月　 　日**

目录

[一、需求分析 3](#_Toc138109818)

[（1）问题描述 3](#_Toc138109819)

[（2）系统功能描述 3](#_Toc138109820)

[（3）安全性与完整性方面要求 3](#_Toc138109821)

[二、概念结构设计 5](#_Toc138109822)

[三、逻辑结构设计 6](#_Toc138109823)

[**（**1）模式设计 6](#_Toc138109824)

[（2）子模式设计 9](#_Toc138109825)

[四、数据库的物理设计 15](#_Toc138109826)

[五、数据库设计实现及运行 15](#_Toc138109827)

[（1）数据库的创建 15](#_Toc138109828)

[（2）数据表的创建 15](#_Toc138109829)

[（3）视图的创建 15](#_Toc138109830)

[（4）存储过程的定义实现 17](#_Toc138109831)

[（5）触发器的定义实现 17](#_Toc138109832)

[（6）自行设计各模块中所涉及的操作语句 22](#_Toc138109833)

[（7）原子性事务显示定义：无 22](#_Toc138109834)

[六、系统详细设计及实现 22](#_Toc138109835)

[（1）系统功能模块说明 22](#_Toc138109836)

[（2）每个模块的关键语句及关键技术说明 22](#_Toc138109837)

[七、总结 23](#_Toc138109838)

一、需求分析

（1）问题描述

系统功能大致有：

教师端：授课列表查询，对应课程赋分。

学生端：信心查询，已选课列表；选课；查询成绩。

管理员端：对学院、专业、班级、教师、学生信息的维护；对教师授课情况的维护；对学生可选课程的维护；对教师学生信息的查询；添加教务管理人员账号。

连接方法：通过登陆界面区分不同用户，分别打开各自的界面（区分选项是id），根据权限大小分别对应不同的界面。

（2）系统功能描述

对自己所负责的功能模块，可画出所设计模块的操作流程；并分析所需要存储的数据信息。



（3）安全性与完整性方面要求

（一）没有定义显示定义原子事务

（二）没有定义存储过程

（三）定义了以下功能的触发器

1（trg\_UpdateEntranceDate\_Class）当Class\_20211576表中加入新数据时，会在添加数据后自动将入学时间Entrance\_date更改为同年9/1；

2（check\_records\_CourseAvail\_Course\_CourseTea）当向Course\_Available\_20211576添加数据时，先判断其对应的组合在Course\_20211576,Course\_Teacher\_202111576中是否存在，若不存在，添加失败并报错；

3（trg\_insert\_from\_course\_chosen\_to\_SC）当向Course\_chosen\_20211576中插入数据时，会在插入成功后在SC\_20211576中插入对应信息；

4（update\_stu\_chosen）当向Course\_chosen\_20211576中插入数据时，会更新在Course\_Available\_20211576中的对应行的Stu\_chosen(Stu\_chosen+=1);

5（add\_teacher\_password）当向Teacher\_202111576中插入数据时，会先判断这条信息在Password\_20211576中是否存在，若存在，则在Password\_20211576中插入对应数据，并设置默认密码为123456；

6（delete\_Teacher\_Password）当要删除Teacher\_20211576中的信息时，执行instead of delete操作，先分别删除Course\_chosen\_20211576,Password\_20211576,Course\_Available\_20211576,Course\_Teacher\_20211576这几张表中对应的信息，最后再删除Teacher\_20211576中的信息；

7（trg\_NewStudent）当向Teacher\_202111576中插入数据时，在Password\_20211576中插入对应数据，并设置默认密码为123456；

8（delete\_Student\_Password）当要删除Teacher\_20211576中的信息时，执行instead of delete操作，先分别删除Course\_chosen\_20211576,Password\_20211576,SC\_20211576这几张表中对应的信息，最后再删除Student\_20211576中的信息；

9（trg\_delete\_info）当删除Academy\_20211576中信息时，先删除Password\_20211576中属于Academy\_20211576中对应删除学院的专业的教师和学生的信息，再删除Class\_20211576中对应删除学院对应专业名下的班级，再删除SC\_20211576中对应删除学院的对应专业的考试成绩信息，再删除Course\_chosen\_20211576中对应删除学院的对应专业的已选课信息，再删除Student\_20211576对应删除学院的专业的学生信息，再删除Teacher\_20211576中对应删除学院的专业的教师信息，再删除Sdept\_20211576中对应删除学院所属的专业信息，再删除Course\_Available\_20211576中对应删除学院的对应专业的可选课程，再删除Course\_Teacher\_20211576中对应删除学院的对应课程及对应的教师授课信息，最后在删除Course\_20211576中对应删除学院开设的课程信息。

10（delete\_course\_trigger）首先在SC\_20211576表中删除与该课程相关的成绩信息，在Course\_chosen\_20211576表中删除与该课程相关的选课信息，在Course\_Available\_20211576表中删除该课程的开课信息，在Course\_Teacher\_20211576表中删除该课程的任课信息，最后在Course\_20211576表中删除该课程的基本信息

11（delete\_course\_teacher\_trigger）首先删除SC\_20211576中与该课程相关的成绩信息，然后删除Course\_chosen\_20211576中与该课程相关的选课信息，然后删除Course\_Available\_20211576中该课程的开课信息，最后删除Course\_Teacher\_20211576中该课程的任课信息。

12（delete\_course\_available\_trigger）首先删除SC\_20211576中与该课程相关的成绩信息，然后删除Course\_chosen\_20211576中与该课程相关的选课信息，最后删除Course\_Available\_20211576中该课程的开课信息。

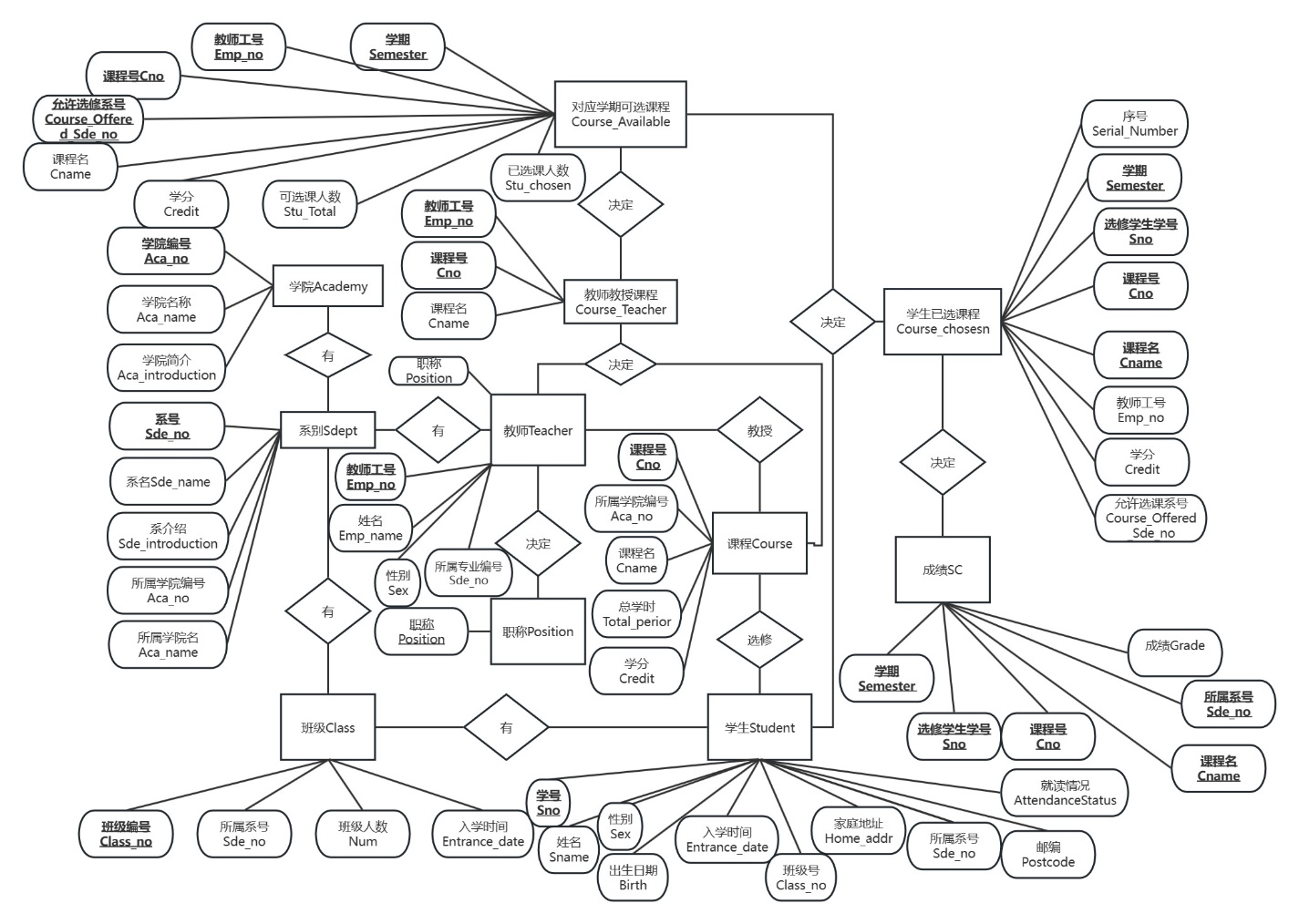
13（delete\_course\_chosen\_trigger）首先删除SC\_20211576中对应的成绩信息，再删除Course\_chosen\_20211576中的已选课信息。

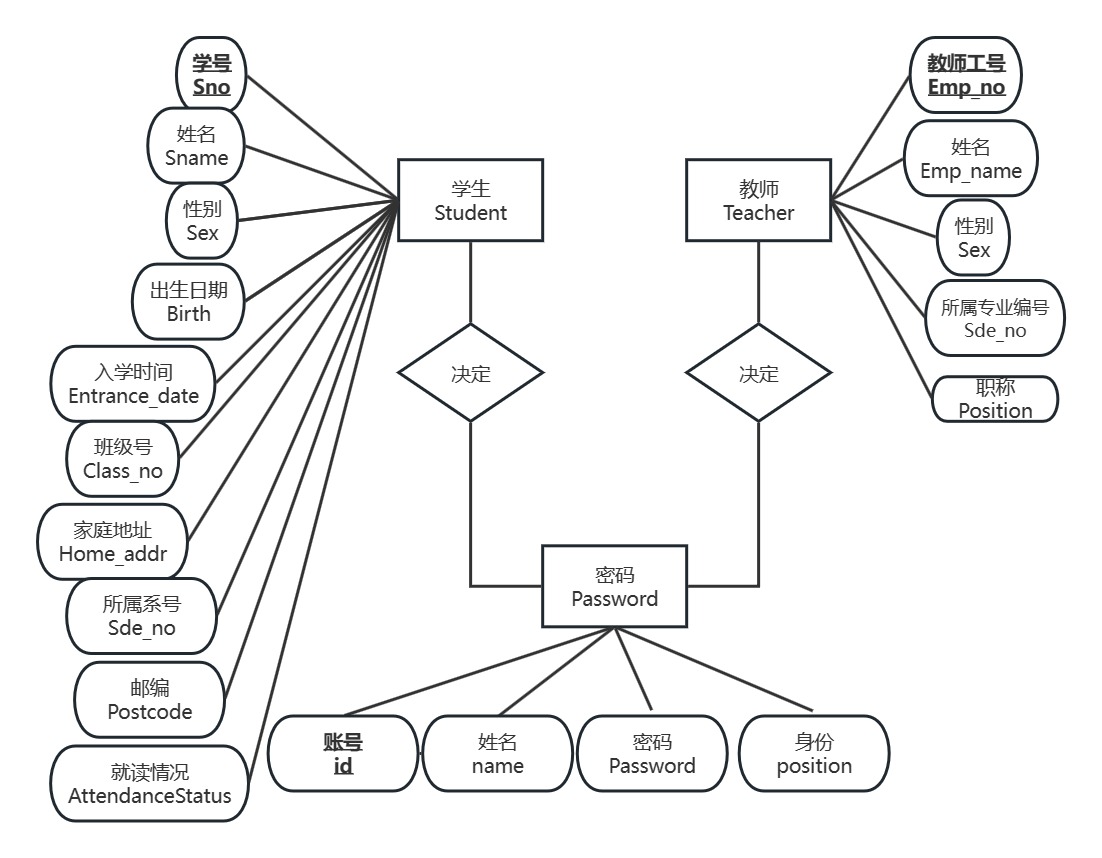
（四）对不同用户采取登陆时鉴别身份的操作

对管理员用户给予全部操作权限，对学生仅提供视图读取权限，对教师，在对学科进行赋分时给予更新权限，当输入成绩后，对已赋分部分仅有视图读取权限，对除赋分以外的其他操作仅有视图读取权限。

二、概念结构设计

画出系统整体的E-R模型；并对模型中所出现的实体及属性等信息加以说明。





三、逻辑结构设计

**（**1）模式设计

按系统整体E-R模型，写出关系模式；并利用数据字典加以描述。

1、Academy\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 |
| Aca\_no | char | 3 | 否 |  | 主键约束 | 是 |
| Aca\_name | varchar | 20 | 否 |  |  |  |
| Aca\_introduction | char | 100 | 是 | 无 |  |  |

2、Sdept\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Sde\_no | char | 3 | 否 |  | 主键约束 | 是 |  |
| Sde\_name | Char | 20 | 否 |  |  |  |  |
| Sde\_introduction | Char | 100 | 是 | 无 |  |  |  |
| Aca\_no | Char | 3 | 否 |  | 外键约束（Academy\_20211576.Aca\_no） |  | 是 |
| Aca\_name | char | 20 | 是 |  |  |  |  |

3、Class\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Class\_no | char | 3 | 否 |  | 主键约束 | 是 |  |
| Sde\_no | char | 3 | 否 |  | 外键约束(Sdept\_20211576.Sde\_no) |  | 是 |
| Entrance\_date | smalldatetime | 100 | 否 |  |  |  |  |
| Num | int |  | 是 |  |  |  |  |

4、Student\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Sno | char | 9 | 否 |  | 主键约束 | 是 |  |
| Sname | Varchar | 8 | 否 |  |  |  |  |
| Sex | Char | 2 | 否 | 男 |  |  |  |
| Birth | Smalldatetime |  | 是 |  |  |  |  |
| Entrance\_date | Smalldatetime |  | 是 |  |  |  |  |
| Class\_no | Char | 7 | 是 |  | 外键约束（Class\_20211576.Class\_no） |  |  |
| Home\_addr | Varchar | 40 | 是 |  |  |  |  |
| Sde\_no | Char | 3 | 否 |  | 外键约束（Sdept\_20211576.Sde\_no） |  | 是 |
| Postcode | Char | 6 | 是 |  | 检查约束（Like’[0-9] [0-9] [0-9] [0-9] [0-9] [0-9]’） |  |  |
| AttendanceStatus | char | 20 | 是 |  |  |  |  |

5、Course\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Cno | char | 10 | 否 |  | 主键约束 | 是 |  |
| Aca\_no | Char | 3 | 否 |  | 外键约束（Academy\_20211576.Aca\_no） |  | 是 |
| Cname | Varchar | 40 | 否 |  |  |  |  |
| Total\_perior | Smallint |  | 是 |  |  |  |  |
| Credit | Float |  | 是 |  |  |  |  |

6、Teacher\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Emp\_no | char | 9 | 否 |  | 主键约束 | 是 |  |
| Emp\_name | varchar | 8 | 否 |  |  |  |  |
| Sex | Char | 2 | 是 | 男 |  |  |  |
| Sde\_no | Char | 10 | 否 |  | 外键约束（Sdept\_20211576.Sde\_no） |  |  |
| Position | char | 10 | 否 |  | 外键约束（Position\_20211576.Position） |  | 是 |

7、Position\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Position | char | 20 | 否 |  | 主键约束 | 是 |  |

8、Course\_Teacher\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Emp\_no | char | 9 | 否 |  | 外键约束（Teacher\_20211576.Emp\_no） | 是 | 是 |
| Cno | Char | 10 | 否 |  | 外键约束（Course\_20211576.Cno） | 是 | 是 |
| Cname | Varchar | 20 | 是 |  |  |  |  |

主键约束：（Emp\_no,Cno）

9、Course\_Available\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Semester | char | 20 | 否 |  |  | 是 | 是 |
| Emp\_no | Char | 9 | 否 |  |  | 是 | 是 |
| Cno | Char | 10 | 否 |  | 外键约束（Course\_20211576.Cno） | 是 | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 | 是 |
| Cname | Varchar | 20 | 是 |  |  |  |  |
| Credit | Tinyint |  | 是 |  |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |  |
| Stu\_chosen | tinyint |  | 否 |  |  |  |  |

主键约束：（Semester,Emp\_no,Cno,Course\_Offered\_Sde\_no）

Emp\_no,Cno外键约束:

|  |  |
| --- | --- |
| 主键表 | 外键表 |
| Course\_Teacher\_20211576 | Course\_Available\_20211576 |
| Emp\_no | Emp\_no |
| Cno | Cno |

10、Course\_chosen\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Semester | char | 20 | 否 |  |  | 是 | 是 |
| Emp\_no | Char | 9 | 否 |  |  | 是 | 是 |
| Sno | char | 9 | 否 |  | 外键约束（Student\_20211576.Sno） | 是 | 是 |
| Cno | Char | 10 | 否 |  |  | 是 | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 | 是 |
| Cname | Varchar | 20 | 是 |  |  |  |  |
| Credit | Tinyint |  | 是 |  |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |  |
| Stu\_chosen | tinyint |  | 否 |  |  |  |  |

Semester,Cno,Emp\_no,Course\_Offered\_Sde\_no外键约束：

|  |  |
| --- | --- |
| 主键表 | 外键表 |
| Course\_Available\_20211576 | Course\_chosen\_20211576 |
| Semester | Semester |
| Cno | Cno |
| Emp\_no | Emp\_no |
| Course\_Offered\_Sde\_no | Course\_Offered\_Sde\_no |

主键约束：（Semester,Emp\_no,Sno,Cno,Course\_Offered\_Sde\_no）

11、SC\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| Semester | char | 20 | 否 |  |  | 是 | 是 |
| Sno | Char | 9 | 否 |  |  | 是 | 是 |
| Cno | char | 10 | 否 |  |  | 是 | 是 |
| Emp\_no | Char | 9 | 否 |  |  | 是 | 是 |
| Sde\_no | Char | 10 | 否 |  |  | 是 | 是 |
| Grade | tinyint |  | 是 |  |  |  |  |

主键约束：（Semester,Emp\_no,Sno,Cno,Course\_Offered\_Sde\_no）

Semester,Cno,Emp\_no,Course\_Offered\_Sde\_no外键约束：

|  |  |
| --- | --- |
| 主键表 | 外键表 |
| Course\_chosen\_20211576 | SC\_20211576 |
| Semester | Semester |
| Sno | Sno |
| Cno | Cno |
| Emp\_no | Emp\_no |
| Course\_Offered\_Sde\_no | Sde\_no |

12、Password\_20211576

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 | 外键 |
| id | char | 9 | 否 |  |  | 是 |  |
| name | Varchar | 8 | 否 |  |  |  |  |
| position | Char | 20 | 否 |  | 外键约束（Position\_20211576.Position） |  | 是 |
| password | char | 20 | 是 |  |  |  |  |

（2）子模式设计

针对自己的设计模块的应用需定义什么样的视图表结构；并利用数据字典加以描述。

（一）登录功能：Manager\_Password\_view

create view Manager\_Password\_view as

select id,name,position,password

from Password\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| id | char | 9 | 否 |  |  | 是 |
| name | Varchar | 8 | 否 |  |  |  |
| position | Char | 20 | 否 |  |  |  |
| password | char | 20 | 是 |  |  |  |

（二）登陆后教师界面

1))Teacher\_view (展示教师信息)

create view Teacher\_view as

select Emp\_no,Emp\_name,Sex,Teacher\_20211576.Sde\_no,Sde\_name,Position

from Teacher\_20211576,Sdept\_20211576

where Teacher\_20211576.Sde\_no=Sdept\_20211576.Sde\_no

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Emp\_no | char | 9 | 否 |  |  | 是 |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 是 | 男 |  |  |
| Sde\_no | Char | 10 | 否 |  |  |  |
| Sde\_name | char | 20 | 否 |  |  |  |
| Position | char | 10 | 否 |  |  |  |

2))Manager\_Course\_Teacher\_manage\_view(查询教授课程)

CREATE VIEW Manager\_Course\_Teacher\_manage\_view AS

SELECT Semester,Course\_Offered\_Sde\_no,Sde\_name,Cno,Cname,Emp\_name,Credit,Stu\_total

FROM Course\_Available\_20211576,Teacher\_20211576,Sdept\_20211576

where Course\_Available\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and Sdept\_20211576.Sde\_no=Course\_Available\_20211576.Course\_Offered\_Sde\_no

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 |
| Sde\_name | char | 20 | 否 |  |  | 是 |
| Cno | Char | 10 | 否 |  |  | 是 |
| Cname | Varchar | 20 | 否 |  |  |  |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Credit | float |  | 是 |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |

Teacher\_Manage\_SC\_view(赋分)：

create view Teacher\_Manage\_SC\_view as

select Semester,Sno,SC\_20211576.Cno,Cname,Emp\_name,SC\_20211576.Sde\_no,Grade

from Teacher\_20211576,SC\_20211576,Course\_20211576

where SC\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and SC\_20211576.Cno=Course\_20211576.Cno

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Sno | Char | 9 | 否 |  |  | 是 |
| Cno | char | 10 | 否 |  |  | 是 |
| Cname | varchar | 8 | 否 |  |  |  |
| Emp\_no | Char | 9 | 否 |  |  | 是 |
| Sde\_no | Char | 10 | 否 |  |  | 是 |
| Grade | tinyint |  | 是 |  |  |  |

（三）登陆后学生界面：

（（一））学生信息+已选课程查询：

1））Student\_Enquiry\_view(已选课程查询)

create view Student\_Enquiry\_view as

SELECT Semester,Sno,Cno,Cname,Emp\_name,Credit

FROM Course\_chosen\_20211576,Teacher\_20211576

where Course\_chosen\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Sno | Char | 9 | 否 |  |  | 是 |
| Cno | char | 10 | 否 |  |  | 是 |
| Cname | varchar | 8 | 否 |  |  |  |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Credit | float |  | 是 |  |  |  |

2））Student\_view(学生信息)

CREATE VIEW Manager\_Student\_view AS

SELECT Sno , Sname , Sex , Birth , Entrance\_date , Class\_no , Home\_addr , Sde\_no , Postcode ,

CASE WHEN (DATEDIFF(year,Entrance\_date,getdate())>=4) THEN '本科毕业' ELSE '本科在读' END AS AttendanceStatus

FROM Student\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Sno | char | 9 | 否 |  |  | 是 |
| Sname | Varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 否 | 男 |  |  |
| Birth | Smalldatetime |  | 是 |  |  |  |
| Entrance\_date | Smalldatetime |  | 是 |  |  |  |
| Class\_no | Char | 7 | 是 |  |  |  |
| Home\_addr | Varchar | 40 | 是 |  |  |  |
| Sde\_no | Char | 3 | 否 |  |  |  |
| Postcode | Char | 6 | 是 |  |  |  |
| AttendanceStatus | char | 20 | 是 |  |  |  |

（（二））学生选课：

Manager\_Course\_Teacher\_manage\_view (选课)

CREATE VIEW Manager\_Course\_Teacher\_manage\_view AS

SELECT Semester,Course\_Offered\_Sde\_no,Sde\_name,Cno,Cname,Emp\_name,Credit,Stu\_total

FROM Course\_Available\_20211576,Teacher\_20211576,Sdept\_20211576

where Course\_Available\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and Sdept\_20211576.Sde\_no=Course\_Available\_20211576.Course\_Offered\_Sde\_no

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 |
| Sde\_name | char | 20 | 否 |  |  | 是 |
| Cno | Char | 10 | 否 |  |  | 是 |
| Cname | Varchar | 20 | 否 |  |  |  |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Credit | float |  | 是 |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |

（四）登陆后管理员界面

（（一））各部分信息维护

1))Manager\_Academy\_view (添加学院信息)

CREATE VIEW Manager\_Academy\_view AS

SELECT Aca\_no,Aca\_name,Aca\_introduction

FROM Academy\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Aca\_no | char | 3 | 否 |  |  | 是 |
| Aca\_name | varchar | 20 | 否 |  |  |  |
| Aca\_introduction | char | 100 | 是 | 无 |  |  |

2)) Manager\_Sdept\_view(专业信息维护)

CREATE VIEW Manager\_Sdept\_view AS

SELECT Sde\_no,Sde\_name,Sde\_introduction,Aca\_no,Aca\_name

FROM Sdept\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Sde\_no | char | 3 | 否 |  |  | 是 |
| Sde\_name | Char | 20 | 否 |  |  |  |
| Sde\_introduction | Char | 100 | 是 | 无 |  |  |
| Aca\_no | Char | 3 | 否 |  |  |  |
| Aca\_name | char | 20 | 是 |  |  |  |

3)) Manager\_Class\_view(班级信息维护,入学时间取出年份/月份/日/yy-mm-dd)

CREATE VIEW Manager\_Class\_view AS

SELECT Class\_no,Sde\_no,substring(CONVERT(varchar, Entrance\_date, 120),0,11) 'Entrance\_date',Num

FROM Class\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Class\_no | char | 3 | 否 |  |  | 是 |
| Sde\_no | char | 3 | 否 |  |  |  |
| Entrance\_date | smalldatetime | 100 | 否 |  |  |  |
| Num | int |  | 是 |  |  |  |

4)) Manager\_Teacher\_view(教师信息维护)

CREATE VIEW Manager\_Teacher\_view AS

SELECT Emp\_no,Emp\_name,Sex,Sde\_no,Position

FROM Teacher\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主键 |
| Emp\_no | char | 9 | 否 |  |  | 是 |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 是 | 男 |  |  |
| Sde\_no | Char | 10 | 否 |  |  |  |
| Position | char | 10 | 否 |  |  |  |

5)) Manager\_Student\_view(学生信息维护，在最后一栏增加该生是否毕业的项)

CREATE VIEW Manager\_Student\_view AS

SELECT Sno , Sname , Sex , Birth , Entrance\_date , Class\_no , Home\_addr , Sde\_no , Postcode ,

CASE WHEN (DATEDIFF(year,Entrance\_date,getdate())>=4) THEN '本科毕业' ELSE '本科在读' END AS AttendanceStatus

FROM Student\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Sno | char | 9 | 否 |  |  | 是 |
| Sname | Varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 否 | 男 |  |  |
| Birth | Smalldatetime |  | 是 |  |  |  |
| Entrance\_date | Smalldatetime |  | 是 |  |  |  |
| Class\_no | Char | 7 | 是 |  |  |  |
| Home\_addr | Varchar | 40 | 是 |  |  |  |
| Sde\_no | Char | 3 | 否 |  |  |  |
| Postcode | Char | 6 | 是 |  |  |  |
| AttendanceStatus | char | 20 | 是 |  |  |  |

（（二））教师授课管理

1))Manager\_Course\_Teacher\_manage\_view(查询教师授课信息)

CREATE VIEW Manager\_Course\_Teacher\_manage\_view AS

SELECT Semester,Course\_Offered\_Sde\_no,Sde\_name,Cno,Cname,Emp\_name,Credit,Stu\_total

FROM Course\_Available\_20211576,Teacher\_20211576,Sdept\_20211576

where Course\_Available\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and Sdept\_20211576.Sde\_no=Course\_Available\_20211576.Course\_Offered\_Sde\_no

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 |
| Sde\_name | char | 20 | 否 |  |  | 是 |
| Cno | Char | 10 | 否 |  |  | 是 |
| Cname | Varchar | 20 | 否 |  |  |  |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Credit | float |  | 是 |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |

2))Course\_Available\_20211576(增加教师授课项)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Emp\_no | Char | 9 | 否 |  |  | 是 |
| Cno | Char | 10 | 否 |  |  | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 |
| Cname | Varchar | 20 | 是 |  |  |  |
| Credit | Tinyint |  | 是 |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |
| Stu\_chosen | tinyint |  | 否 |  |  |  |

（（三））学生可选课程信息维护

Manager\_Course\_Teacher\_manage\_view(查询对应系学生可选课程)

CREATE VIEW Manager\_Course\_Teacher\_manage\_view AS

SELECT Semester,Course\_Offered\_Sde\_no,Sde\_name,Cno,Cname,Emp\_name,Credit,Stu\_total

FROM Course\_Available\_20211576,Teacher\_20211576,Sdept\_20211576

where Course\_Available\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and Sdept\_20211576.Sde\_no=Course\_Available\_20211576.Course\_Offered\_Sde\_no

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Semester | char | 20 | 否 |  |  | 是 |
| Course\_Offered\_Sde\_no | Char | 10 | 否 |  |  | 是 |
| Sde\_name | char | 20 | 否 |  |  | 是 |
| Cno | Char | 10 | 否 |  |  | 是 |
| Cname | Varchar | 20 | 否 |  |  |  |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Credit | float |  | 是 |  |  |  |
| Stu\_Total | Tinyint |  | 是 |  |  |  |

（（四））教师学生信息查询系统

1））Manager\_Teacher\_view(教师信息查询)

CREATE VIEW Manager\_Teacher\_view AS

SELECT Emp\_no,Emp\_name,Sex,Sde\_no,Position

FROM Teacher\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Emp\_no | char | 9 | 否 |  |  | 是 |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 是 | 男 |  |  |
| Sde\_no | Char | 10 | 否 |  |  |  |
| Position | char | 10 | 否 |  |  |  |

2)) Manager\_Student\_view(学生信息查询，在最后一栏增加该生是否毕业的项)

CREATE VIEW Manager\_Student\_view AS

SELECT Sno , Sname , Sex , Birth , Entrance\_date , Class\_no , Home\_addr , Sde\_no , Postcode ,

CASE WHEN (DATEDIFF(year,Entrance\_date,getdate())>=4) THEN '本科毕业' ELSE '本科在读' END AS AttendanceStatus

FROM Student\_20211576

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Sno | char | 9 | 否 |  |  | 是 |
| Sname | Varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 否 | 男 |  |  |
| Birth | Smalldatetime |  | 是 |  |  |  |
| Entrance\_date | Smalldatetime |  | 是 |  |  |  |
| Class\_no | Char | 7 | 是 |  |  |  |
| Home\_addr | Varchar | 40 | 是 |  |  |  |
| Sde\_no | Char | 3 | 否 |  |  |  |
| Postcode | Char | 6 | 是 |  |  |  |
| AttendanceStatus | char | 20 | 是 |  |  |  |

（（五））添加教务管理系统管理员用户

Teacher\_20211576(插入管理员教师信息，Position为”教务管理人员”)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名称 | 类型 | 宽度 | 允许空值 | 缺省值 | 约束 | 主码 |
| Emp\_no | char | 9 | 否 |  |  | 是 |
| Emp\_name | varchar | 8 | 否 |  |  |  |
| Sex | Char | 2 | 是 | 男 |  |  |
| Sde\_no | Char | 10 | 否 |  |  |  |
| Position | char | 10 | 否 |  |  |  |

四、数据库的物理设计

为了提高在表中搜索元组的速度，在实际实现的时候应该基于某些属性建立索引。给出所建立的索引。（包括按哪些表中哪些属性按升序还是降序来创建索引及相应的T-SQL的创建语句）。

1.CREATE CLUSTERED INDEX Academy\_clustered\_index

ON Academy\_20211576(Aca\_no ASC);

2.CREATE CLUSTERED INDEX Class\_clustered\_index

ON Class\_20211576(Class\_no ASC);

3.CREATE CLUSTERED INDEX Course\_clustered\_index

ON Course\_20211576(Cno ASC);

4.CREATE CLUSTERED INDEX Course\_Available\_clustered\_index

ON Course\_Available\_20211576(Semester ASC,Emp\_no ASC,Cno ASC,Course\_Offered\_Sde\_no ASC);

5.CREATE CLUSTERED INDEX Course\_chosen\_clustered\_index

ON Course\_chosen\_20211576(Semester ASC,Sno ASC,Cno ASC,Emp\_no ASC,Course\_Offered\_Sde\_no ASC);

6.CREATE CLUSTERED INDEX Course\_Teacher\_clustered\_index

ON Course\_Teacher\_20211576(Emp\_no ASC,Cno ASC);

7.CREATE CLUSTERED INDEX Password\_clustered\_index

ON Password\_20211576(id ASC);

8.CREATE CLUSTERED INDEX Position\_clustered\_index

ON Position\_20211576(Position ASC);

9.CREATE CLUSTERED INDEX SC\_clustered\_index

ON SC\_20211576(Semester ASC,Sno ASC,Cno ASC,Emp\_no ASC,Sde\_no ASC);

10.CREATE CLUSTERED INDEX Sdept\_clustered\_index

ON Sdept\_202111576(Sde\_no ASC);

11.CREATE CLUSTERED INDEX Student\_clustered\_index

ON Student\_20211576(Sno ASC);

12.CREATE CLUSTERED INDEX Teacher\_clustered\_index

ON Teacher\_20211576(Emp\_no ASC);

五、数据库设计实现及运行

（1）数据库的创建

T-SQL语句：

Create database Academic\_Affairs\_Management\_System\_20211576

Use Academic\_Affairs\_Management\_System\_20211576

（2）数据表的创建

T-SQL语句：

以下是各个表的CREATE语句：

1.Academy\_20211576表：

CREATE TABLE Academy\_20211576 (

Aca\_no char(3) NOT NULL PRIMARY KEY,

Aca\_name varchar(20) NOT NULL,

Aca\_introduction char(100) NULL);

2.Sdept\_20211576表：

CREATE TABLE Sdept\_20211576 (

Sde\_no char(3) NOT NULL PRIMARY KEY,

Sde\_name char(20) NOT NULL,

Sde\_introduction char(100) NULL,

Aca\_no char(3) NOT NULL,

Aca\_name char(20) NULL,

FOREIGN KEY (Aca\_no) REFERENCES Academy\_20211576(Aca\_no));

3.Class\_20211576表：

CREATE TABLE Class\_20211576 (

Class\_no char(3) NOT NULL PRIMARY KEY,

Sde\_no char(3) NOT NULL,

Entrance\_date smalldatetime NOT NULL,

Num int NULL,

FOREIGN KEY (Sde\_no) REFERENCES Sdept\_20211576(Sde\_no));

4.Student\_20211576表：

CREATE TABLE Student\_20211576 (

Sno char(9) NOT NULL PRIMARY KEY,

Sname varchar(8) NOT NULL,

Sex char(2) NOT NULL DEFAULT '男',

Birth smalldatetime NULL,

Entrance\_date smalldatetime NULL,

Class\_no char(7) NULL,

Home\_addr varchar(40) NULL,

Sde\_no char(3) NOT NULL,

Postcode char(6) NULL,

AttendanceStatus char(20) NULL,

FOREIGN KEY (Sde\_no) REFERENCES Sdept\_20211576(Sde\_no),

FOREIGN KEY (Class\_no) REFERENCES Class\_20211576(Class\_no));

5.Course\_20211576表：

CREATE TABLE Course\_20211576 (

Cno char(10) NOT NULL PRIMARY KEY,

Aca\_no char(3) NOT NULL,

Cname varchar(40) NOT NULL,

Total\_perior smallint NULL,

Credit float NULL,

FOREIGN KEY (Aca\_no) REFERENCES Academy\_20211576(Aca\_no));

6.Teacher\_20211576表：

CREATE TABLE Teacher\_20211576 (

Emp\_no char(9) NOT NULL PRIMARY KEY,

Emp\_name varchar(8) NOT NULL,

Sex char(2) NULL DEFAULT '男',

Sde\_no char(10) NOT NULL,

Position char(20) NOT NULL,

FOREIGN KEY (Sde\_no) REFERENCES Sdept\_20211576(Sde\_no),

FOREIGN KEY (Position) REFERENCES Position\_20211576(Position));

7.Position\_20211576表：

CREATE TABLE Position\_20211576 (

Position char(20) NOT NULL PRIMARY KEY);

8.Course\_Teacher\_20211576表：

CREATE TABLE Course\_Teacher\_20211576 (

Emp\_no char(9) NOT NULL,

Cno char(10) NOT NULL,

Cname varchar(20) NULL,

PRIMARY KEY (Emp\_no, Cno),

FOREIGN KEY (Emp\_no) REFERENCES Teacher\_20211576(Emp\_no),

FOREIGN KEY (Cno) REFERENCES Course\_20211576(Cno));

9.Course\_Available\_20211576表：

CREATE TABLE Course\_Available\_20211576 (

Semester char(20) NOT NULL,

Emp\_no char(9) NOT NULL,

Cno char(10) NOT NULL,

Course\_Offered\_Sde\_no char(10) NOT NULL,

Cname varchar(20) NULL,

Credit tinyint NULL,

Stu\_Total tinyint NULL,

Stu\_chosen tinyint NOT NULL,

PRIMARY KEY (Semester, Emp\_no, Cno, Course\_Offered\_Sde\_no),

FOREIGN KEY (Emp\_no,Cno) REFERENCES Course\_Teacher\_20211576(Emp\_no,Cno),

FOREIGN KEY (Cno) REFERENCES Course\_20211576(Cno));

10.Course\_chosen\_20211576表：

CREATE TABLE Course\_chosen\_20211576 (

Serial\_Number int NULL,

Semester char(20) NOT NULL,

Emp\_no char(9) NOT NULL,

Sno char(9) NOT NULL,

Cno char(10) NOT NULL,

Course\_Offered\_Sde\_no char(10) NOT NULL,

Cname varchar(20) NULL,

Credit tinyint NULL,

Stu\_Total tinyint NULL,

Stu\_chosen tinyint NOT NULL,

PRIMARY KEY (Semester, Emp\_no, Sno, Cno, Course\_Offered\_Sde\_no),

FOREIGN KEY (Semester, Emp\_no, Sno, Cno, Course\_Offered\_Sde\_no) REFERENCES Course\_Available\_20211576(Semester, Emp\_no, Sno, Cno, Course\_Offered\_Sde\_no),

FOREIGN KEY (Sno) REFERENCES Student\_20211576(Sno));

11.SC\_20211576表：

CREATE TABLE SC\_20211576 (

Semester char(20) NOT NULL,

Sno char(9) NOT NULL,

Cno char(10) NOT NULL,

Emp\_no char(9) NOT NULL,

Sde\_no char(10) NOT NULL,

Grade tinyint NULL,

PRIMARY KEY (Semester, Sno, Cno, Emp\_no, Sde\_no),

FOREIGN KEY (Semester, Sno, Cno, Emp\_no, Sde\_no) REFERENCES Course\_chosen\_20211576(Semester, Sno, Cno, Emp\_no, Sde\_no));

12.Password\_20211576表：

CREATE TABLE Password\_20211576 (

id char(9) NOT NULL PRIMARY KEY,

name varchar(8) NOT NULL,

position char(20) NOT NULL,

password char(20) NULL,

FOREIGN KEY (position) REFERENCES Position\_20211576(Position));

说明：若有完整性约束的一方面可以创建时同时实现。按需求及自行设计模块中所实现的情况，在该部分对完整性约束加以实现。

（3）视图的创建

T-SQL语句：

1、Manager\_Student\_view：

CREATE VIEW Manager\_Student\_view AS

SELECT Sno , Sname , Sex , Birth , Entrance\_date , Class\_no , Home\_addr , Sde\_no , Postcode ,

CASE WHEN (DATEDIFF(year,Entrance\_date,getdate())>=4) THEN '本科毕业' ELSE '本科在读' END AS AttendanceStatus

FROM Student\_20211576

2、Manager\_Teacher\_view：

CREATE VIEW Manager\_Teacher\_view AS

SELECT Emp\_no,Emp\_name,Sex,Sde\_no,Position

FROM Teacher\_20211576

3、Manager\_Class\_view：

CREATE VIEW Manager\_Class\_view AS

SELECT Class\_no,Sde\_no,substring(CONVERT(varchar, Entrance\_date, 120),0,11) 'Entrance\_date',Num

FROM Class\_20211576

4、Manager\_Academy\_view：

CREATE VIEW Manager\_Academy\_view AS

SELECT Aca\_no,Aca\_name,Aca\_introduction

FROM Academy\_20211576

5、Manager\_Sdept\_view：

CREATE VIEW Manager\_Sdept\_view AS

SELECT Sde\_no,Sde\_name,Sde\_introduction,Aca\_no,Aca\_name

FROM Sdept\_20211576

6、Course\_teacher\_view:

create view Course\_teacher\_view as

select Emp\_no,Cno,Cname

from Course\_Teacher\_20211576

7、Manager\_Course\_Teacher\_manage\_view：

CREATE VIEW Manager\_Course\_Teacher\_manage\_view AS

SELECT Semester,Course\_Offered\_Sde\_no,Sde\_name,Cno,Cname,Emp\_name,Credit,Stu\_total

FROM Course\_Available\_20211576,Teacher\_20211576,Sdept\_20211576

where Course\_Available\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and Sdept\_20211576.Sde\_no=Course\_Available\_20211576.Course\_Offered\_Sde\_no

8、Student\_Enquiry\_view:

create view Student\_Enquiry\_view as

SELECT Semester,Sno,Cno,Cname,Emp\_name,Credit

FROM Course\_chosen\_20211576,Teacher\_20211576

where Course\_chosen\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

9、Student\_view:

create view Student\_view as

select Sno,Sname,Sex,substring(CONVERT(varchar, Birth, 120),0,11) 'Birth',substring(CONVERT(varchar, Entrance\_date, 120),0,11) 'Entrance\_date',Class\_no,Home\_addr,Student\_20211576.Sde\_no,Sde\_name,Postcode,

CASE WHEN (DATEDIFF(year,Entrance\_date,getdate())>=4) THEN '本科毕业' ELSE '本科在读' END AS AttendanceStatus

from Student\_20211576,Sdept\_20211576

where Student\_20211576.Sde\_no=Sdept\_20211576.Sde\_no

10、Teacher\_view:

create view Teacher\_view as

select Emp\_no,Emp\_name,Sex,Teacher\_20211576.Sde\_no,Sde\_name,Position

from Teacher\_20211576,Sdept\_20211576

where Teacher\_20211576.Sde\_no=Sdept\_20211576.Sde\_no

11. Teacher\_Score\_Manage\_view:查看已经被选的课程的课程信息

create view Teacher\_Score\_Manage\_view as

select Semester,Sno,Cno,Cname,Emp\_no,Credit,Course\_Offered\_Sde\_no

from Course\_chosen\_20211576

12、Teacher\_Manage\_SC\_view:给学生赋分

create view Teacher\_Manage\_SC\_view as

select Semester,Sno,SC\_20211576.Cno,Cname,Emp\_name,SC\_20211576.Sde\_no,Grade

from Teacher\_20211576,SC\_20211576,Course\_20211576

where SC\_20211576.Emp\_no=Teacher\_20211576.Emp\_no

and SC\_20211576.Cno=Course\_20211576.Cno

13、Student\_Score\_Enquiry\_view:

create view Student\_Score\_Enquiry\_view as

select Semester,SC\_20211576.Sno,Sname,SC\_20211576.Cno,Cname,SC\_20211576.Emp\_no,Emp\_name,Grade

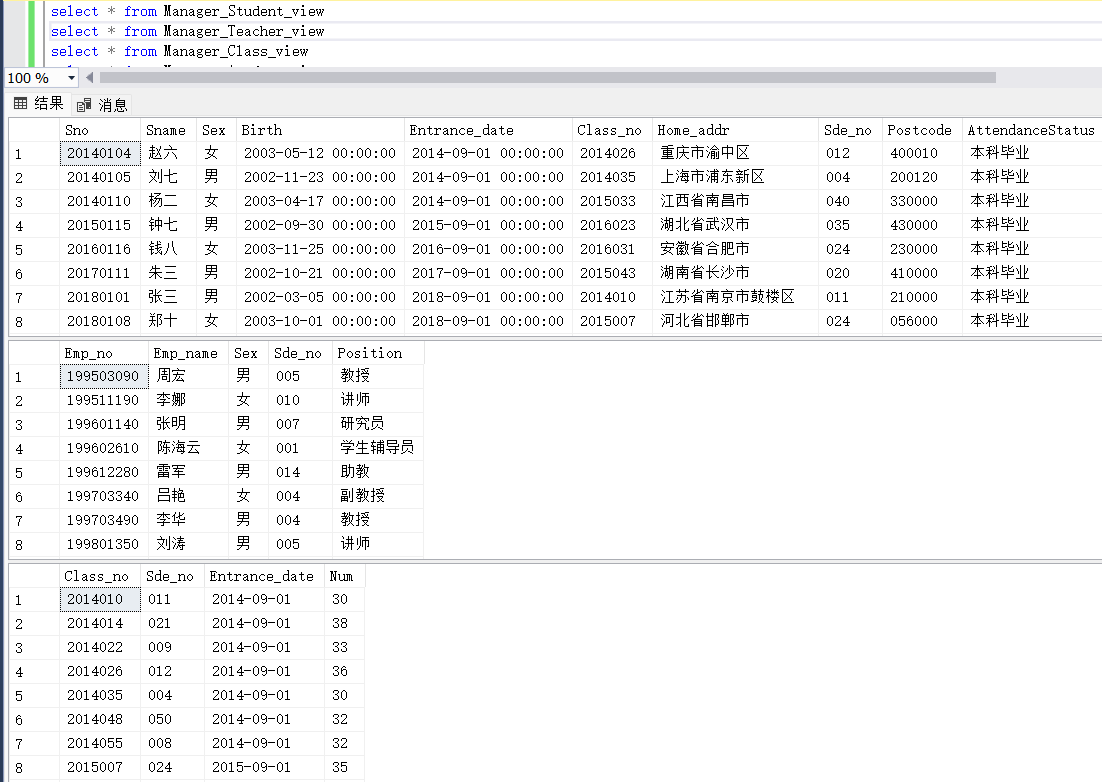
from SC\_20211576,Student\_20211576,Course\_20211576,Teacher\_20211576

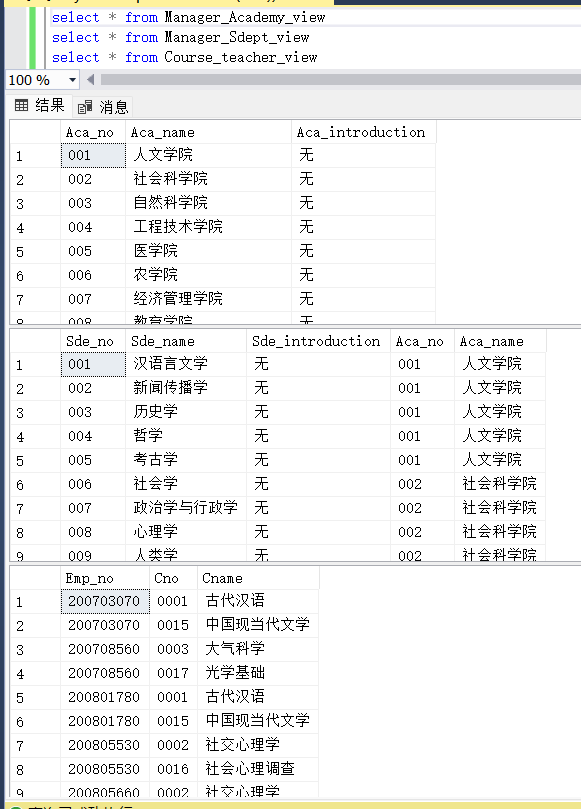
where SC\_20211576.Sno=Student\_20211576.Sno

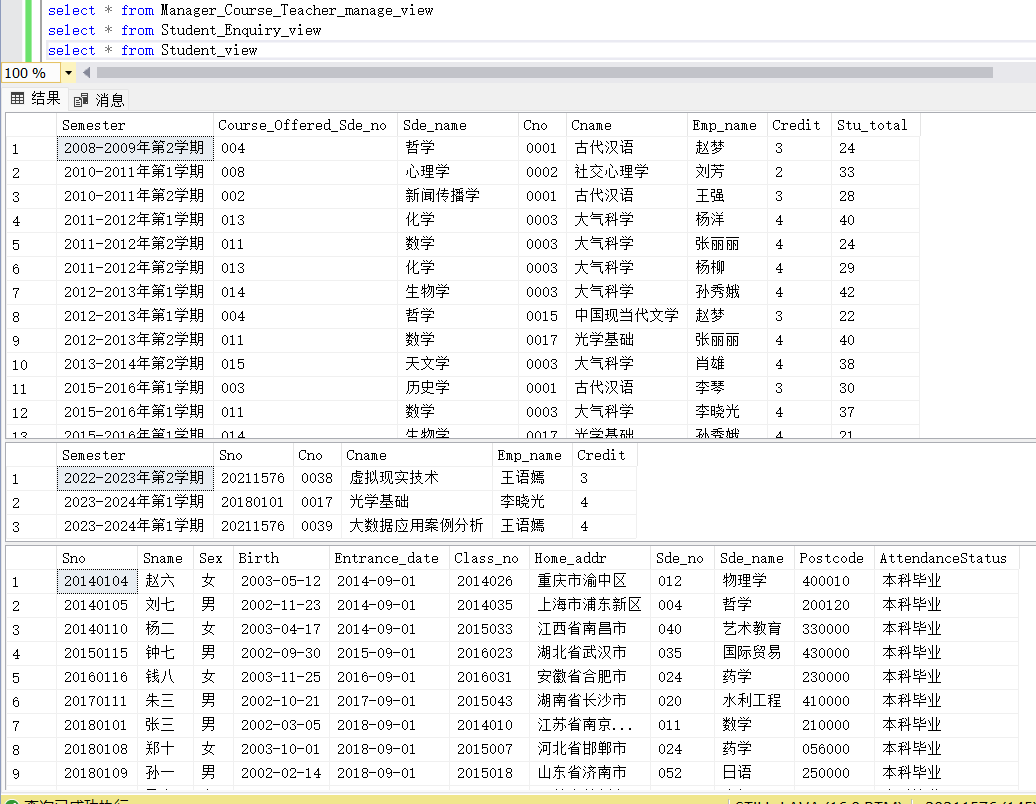
and Course\_20211576.Cno=SC\_20211576.Cno

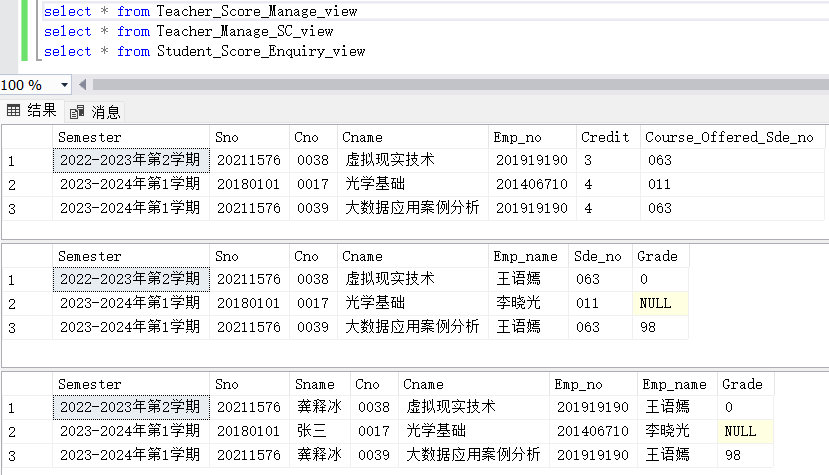
and Teacher\_20211576.Emp\_no=SC\_20211576.Emp\_no

测试结果粘贴：









（4）存储过程的定义实现

T-SQL语句：

无

在宿主语言中调用存储过程代码段（粘贴）：

无

测试结果粘贴：

无

（5）触发器的定义实现

T-SQL语句：

1）trg\_UpdateEntranceDate\_Class：

CREATE TRIGGER trg\_UpdateEntranceDate\_Class

ON Class\_20211576

AFTER INSERT, UPDATE

AS

BEGIN

UPDATE S

SET Entrance\_date = SUBSTRING(S.Class\_no, 1, 4) + '/09/01'

FROM Class\_20211576 AS S

INNER JOIN (

SELECT DISTINCT Class\_no FROM inserted

) AS T ON S.Class\_no = T.Class\_no

END

2)check\_records\_CourseAvail\_Course\_CourseTea

CREATE TRIGGER check\_records\_CourseAvail\_Course\_CourseTea

ON Course\_Available\_20211576

AFTER INSERT

AS

BEGIN

IF NOT EXISTS (

SELECT \*

FROM inserted AS i

WHERE NOT EXISTS (

SELECT \*

FROM Course\_20211576

WHERE Cno = i.Cno AND Credit = i.Credit

) OR NOT EXISTS (

SELECT \*

FROM Course\_Teacher\_20211576

WHERE Cno = i.Cno AND Emp\_no = i.Emp\_no

)

)

BEGIN

RAISERROR ('插入失败！Course\_20211576或Course\_Teacher\_20211576不存在相同数据！', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

END

3)trg\_insert\_from\_course\_chosen\_to\_SC：

CREATE TRIGGER trg\_insert\_from\_course\_chosen\_to\_SC

ON Course\_chosen\_20211576

AFTER INSERT

AS

BEGIN

INSERT INTO SC\_20211576(Semester, Sno, Cno, Emp\_no, Sde\_no, Grade)

SELECT i.Semester, i.Sno, i.Cno, i.Emp\_no, i.Course\_Offered\_Sde\_no,0

FROM inserted i;

END

1. update\_stu\_chosen：

CREATE TRIGGER update\_stu\_chosen

ON Course\_Chosen\_20211576

AFTER INSERT

AS

BEGIN

UPDATE Course\_Available\_20211576

SET Stu\_chosen = Stu\_chosen + 1

WHERE Semester IN (SELECT Semester FROM inserted)

AND Emp\_no IN (SELECT Emp\_no FROM inserted)

AND Cno IN (SELECT Cno FROM inserted);

END

4)add\_teacher\_password：

CREATE TRIGGER add\_teacher\_password

ON Teacher\_20211576

AFTER INSERT

AS

BEGIN

INSERT INTO Password\_20211576 (id, name, position, password)

SELECT i.Emp\_no, i.Emp\_name, i.Position, '123456'

FROM inserted i

WHERE NOT EXISTS (

SELECT \* FROM Password\_20211576 WHERE id = i.Emp\_no AND position = i.Position

);

END

5)delete\_Teacher\_Password：

CREATE TRIGGER delete\_Teacher\_Password

ON Teacher\_20211576

INSTEAD OF DELETE

AS

BEGIN

DECLARE @DeletedEmp\_no TABLE (Emp\_no char(9));

INSERT INTO @DeletedEmp\_no (Emp\_no)

SELECT Emp\_no FROM DELETED;

DELETE FROM Course\_chosen\_20211576 WHERE Cno IN (SELECT Cno FROM Course\_Available\_20211576 WHERE Emp\_no IN (SELECT Emp\_no FROM @DeletedEmp\_no));

DELETE FROM Password\_20211576 WHERE id IN (SELECT Emp\_no FROM @DeletedEmp\_no);

DELETE FROM Course\_Available\_20211576 WHERE Emp\_no IN (SELECT Emp\_no FROM @DeletedEmp\_no);

DELETE FROM Course\_Teacher\_20211576 WHERE Emp\_no IN (SELECT Emp\_no FROM @DeletedEmp\_no);

DELETE FROM Teacher\_20211576 WHERE Emp\_no IN (SELECT Emp\_no FROM @DeletedEmp\_no);

END

6)trg\_NewStudent：

CREATE TRIGGER trg\_NewStudent

ON Student\_20211576

AFTER INSERT

AS

BEGIN

INSERT INTO Password\_20211576 (id, name, position, password)

SELECT i.Sno, i.Sname, '学生', '123456'

FROM inserted i

END

7)delete\_Student\_Password：

CREATE TRIGGER delete\_Student\_Password

ON Student\_20211576

AFTER DELETE

AS

BEGIN

DELETE FROM Course\_chosen\_20211576 WHERE Sno IN (

SELECT Sno FROM deleted

);

DELETE FROM Password\_20211576 WHERE id IN (

SELECT Sno FROM deleted

);

DELETE FROM SC\_20211576 WHERE Sno IN (

SELECT Sno FROM deleted

);

UPDATE Course\_Available\_20211576 SET Stu\_chosen = Stu\_chosen - 1 WHERE Cno IN (

SELECT DISTINCT Cno FROM SC\_20211576 WHERE Sno IN (

SELECT Sno FROM deleted

)

);

END;

8)trg\_delete\_info：

CREATE TRIGGER trg\_delete\_info

ON Academy\_20211576

INSTEAD OF DELETE

AS

BEGIN

DELETE FROM Password\_20211576 WHERE id IN (SELECT Sno FROM Student\_20211576 WHERE Sde\_no IN (SELECT Sde\_no FROM Sdept\_20211576 JOIN deleted d ON Sdept\_20211576.Aca\_no = d.Aca\_no));

DELETE FROM Password\_20211576 WHERE id IN (SELECT Emp\_no FROM Teacher\_20211576 WHERE Sde\_no IN (SELECT Sde\_no FROM Sdept\_20211576 JOIN deleted d ON Sdept\_20211576.Aca\_no = d.Aca\_no));

DELETE c FROM Class\_20211576 c JOIN deleted d ON c.Sde\_no IN (SELECT s.Sde\_no FROM Sdept\_20211576 s WHERE s.Aca\_no = d.Aca\_no);

DELETE sc FROM SC\_20211576 sc WHERE sc.Cno IN (SELECT co.Cno FROM Course\_20211576 co JOIN deleted d ON co.Aca\_no = d.Aca\_no);

DELETE ccs FROM Course\_chosen\_20211576 ccs WHERE ccs.Cno IN (SELECT co.Cno FROM Course\_20211576 co JOIN deleted d ON co.Aca\_no = d.Aca\_no);

DELETE st FROM Student\_20211576 st JOIN deleted d ON st.Sde\_no IN (SELECT s.Sde\_no FROM Sdept\_20211576 s WHERE s.Aca\_no = d.Aca\_no);

DELETE t FROM Teacher\_20211576 t JOIN deleted d ON t.Sde\_no IN (SELECT s.Sde\_no FROM Sdept\_20211576 s WHERE s.Aca\_no = d.Aca\_no);

DELETE s FROM Sdept\_20211576 s JOIN deleted d ON s.Aca\_no = d.Aca\_no;

DELETE cas FROM Course\_Available\_20211576 cas WHERE cas.Cno IN (SELECT co.Cno FROM Course\_20211576 co JOIN deleted d ON co.Aca\_no = d.Aca\_no);

DELETE cts FROM Course\_Teacher\_20211576 cts WHERE cts.Cno IN (SELECT co.Cno FROM Course\_20211576 co JOIN deleted d ON co.Aca\_no = d.Aca\_no);

DELETE co FROM Course\_20211576 co JOIN deleted d ON co.Aca\_no = d.Aca\_no;

END

9)delete\_course\_trigger：

CREATE TRIGGER delete\_course\_trigger

ON Course\_20211576

INSTEAD OF DELETE

AS

BEGIN

BEGIN TRANSACTION;

DELETE FROM SC\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_chosen\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_Available\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_Teacher\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

COMMIT TRANSACTION;

END

10)delete\_course\_teacher\_trigger：

CREATE TRIGGER delete\_course\_teacher\_trigger

ON Course\_Teacher\_20211576

INSTEAD OF DELETE AS

BEGIN

DELETE FROM SC\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_chosen\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_Available\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_Teacher\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

END;

11)delete\_course\_available\_trigger：

CREATE TRIGGER delete\_course\_available\_trigger

ON Course\_Available\_20211576

INSTEAD OF DELETE AS

BEGIN

DELETE FROM SC\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_chosen\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_Available\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

END;

12)delete\_course\_chosen\_trigger：

CREATE TRIGGER delete\_course\_chosen\_trigger

ON Course\_chosen\_20211576

INSTEAD OF DELETE AS

BEGIN

DELETE FROM SC\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

DELETE FROM Course\_chosen\_20211576 WHERE Cno IN (SELECT Cno FROM deleted);

END;

测试结果粘贴：

（6）自行设计各模块中所涉及的操作语句

1）插入数据操作

功能模块1：教务管理系统学生界面-学生选课界面选课

功能界面：



描述：点击选课按钮前需要先选中表格中一项，若未选会弹出报错窗口，点击选课后，若该课已经被该同学选过，则会弹窗提示该课程已选，若未选过，则会弹窗提示选课成功。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）；

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btnChoose) {

if(sqlInsert.equals("")) {

JOptionPane.showMessageDialog(null, "请先选择课程！", "提示", JOptionPane.ERROR\_MESSAGE);

}else {

String Driver2 = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

String url = "jdbc:sqlserver://localhost:1433;DatabaseName=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";

String userName = "s20211576"; // 默认用户名

String userPwd = "s20211576"; // 密码

Connection dbConn = null;

try {

Class.forName(Driver2);

dbConn = DriverManager.getConnection(url, userName, userPwd);

} catch (ClassNotFoundException ec) {

throw new RuntimeException(ec);

} catch (SQLException ex) {

throw new RuntimeException(ex);

}

try {

Statement stmt = dbConn.createStatement();

System.out.println("sqlInsert:"+sqlInsert);

if(checkInsert(sqlCheck) == true){//checkInsert:检查该生是否已选过这门课

JOptionPane.showMessageDialog(null, "该课程已选！", "提示", JOptionPane.ERROR\_MESSAGE);

dbConn.close();

return;

}

else{

JOptionPane.showMessageDialog(null, "选课成功！", "提示", JOptionPane.INFORMATION\_MESSAGE);

}

ResultSet rs = stmt.executeQuery(sqlInsert);

} catch (SQLException ee) {

throw new RuntimeException(ee);

} finally {

try {

dbConn.close();

} catch (SQLException exc) {

throw new RuntimeException(exc);

}

}

}

}

// 添加监听器，监听表格的行选中事件

table.getSelectionModel().addListSelectionListener(new ListSelectionListener() {

public void valueChanged(ListSelectionEvent event) {

// 获取当前选中行的索引

int row = table.getSelectedRow();

if (row >= 0) {

// 获取选中行的各列的数据

String insertSemester = table.getValueAt(row, 0).toString();

String insertSno = loginID.toString();

String insertCourseOfferedSdeNo = table.getValueAt(row, 1).toString();

String insertCno = table.getValueAt(row, 3).toString();

String insertCname = table.getValueAt(row, 4).toString();

//中间变量insertEmpName

String insertEmpName = table.getValueAt(row, 5).toString();

String insertEmpNo = getEmpNo(insertEmpName);

String insertCredit = table.getValueAt(row, 6).toString();

String insertStuTotal = table.getValueAt(row, 7).toString();

sqlInsert = "INSERT INTO Course\_chosen\_20211576(Semester,Sno,Cno,Cname,Emp\_no,Credit,Course\_Offered\_Sde\_no) VALUES ('" + insertSemester.trim() + "','" + insertSno.trim() + "','" + insertCno.trim() + "','" + insertCname.trim() + "','" + insertEmpNo.trim() + "'," + insertCredit + ",'" + insertCourseOfferedSdeNo.trim() + "')";

sqlCheck = "SELECT \* FROM Course\_chosen\_20211576 where Sno='" + insertSno.trim() + "' and Cno='" + insertCno.trim() + "' and Semester='" + insertSemester.trim() + "' and Course\_Offered\_Sde\_no='" + insertCourseOfferedSdeNo.trim() + "' and Emp\_no='" + insertEmpNo.trim() + "'";

}

}

});

}

private boolean checkInsert(String sqlCheck) {

String Driver2 = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

String url = "jdbc:sqlserver://localhost:1433;DatabaseName=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";

String userName = "s20211576"; // 默认用户名

String userPwd = "s20211576"; // 密码

boolean flag = false;

Connection dbConn = null;

try {

Class.forName(Driver2);

dbConn = DriverManager.getConnection(url, userName, userPwd);

} catch (ClassNotFoundException ex) {

throw new RuntimeException(ex);

} catch (SQLException exc) {

throw new RuntimeException(exc);

}

int columnCount;

try {

Statement stmt = dbConn.createStatement();

ResultSet rs = stmt.executeQuery(sqlCheck);

ResultSetMetaData rsmd = rs.getMetaData();

columnCount = 0;

while (rs.next()) {

flag = true;

break;

}

} catch (SQLException e) {

throw new RuntimeException(e);

} finally {

try {

dbConn.close();

} catch (SQLException e) {

throw new RuntimeException(e);

}

}

return flag;

}

测试结果粘贴：



每个功能模块描述方法如下：

功能界面（粘贴）；

功能界面简单描述；

T-SQL语句与宿主语言嵌套使用代码段（粘贴）；

测试结果粘贴；

2）删除数据操作

每个功能模块描述方法如下：

功能界面（粘贴）；

功能界面简单描述；

T-SQL语句与宿主语言嵌套使用代码段（粘贴）；

测试结果粘贴：

3）修改数据操作

每个功能模块描述方法如下：

功能界面（粘贴）；

功能界面简单描述；

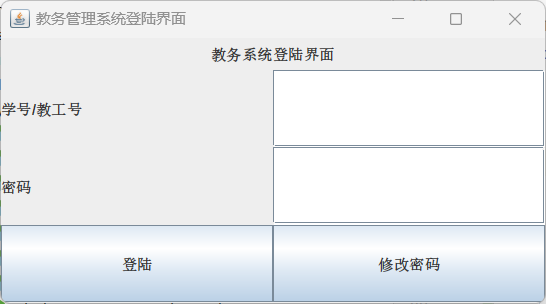
T-SQL语句与宿主语言嵌套使用代码段（粘贴）；

测试结果粘贴；

4）查询数据操作：

功能模块1：登陆界面查询操作

功能界面：



描述：包含用户名和密码输入框、登录按钮和注册按钮，用户输入用户名和密码后，若密码正确，则点击登录按钮即可进入主界面，教务管理人员对应的账号进入管理员界面，学生对应的账号进入学生界面，其余密码与用户名匹配的均进入教师界面。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == buttonRegister) {

if (txtLabelNo.getText().trim().equals("")) {

JOptionPane.showMessageDialog(null, "请输入学号/教工号", "提示", JOptionPane.ERROR\_MESSAGE);

} else if (txtLabelPassword.getText().trim().equals("")) {

JOptionPane.showMessageDialog(null, "密码不能为空", "提示", JOptionPane.ERROR\_MESSAGE);

} else {

String userInput = null;

if(txtLabelNo.getText().length()<7){

userInput = "'" + txtLabelNo.getText() + "'";

}

else{

userInput = txtLabelNo.getText();

}

String passwordInput = txtLabelPassword.getText().toString();

String Driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

String url = "jdbc:sqlserver://localhost:1433;database=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";

String sql="use Academic\_Affairs\_Management\_System\_20211576 select Position from Password\_20211576 where id = " + userInput;

String sql1="use Academic\_Affairs\_Management\_System\_20211576 select password from Password\_20211576 where id = " + userInput;

String userName = "s20211576"; // 默认用户名

String userPwd = "s20211576"; // 密码

Connection dbConn = null;

try {

Class.forName(Driver);

dbConn = DriverManager.getConnection(url, userName, userPwd);

} catch (Exception em) {

em.printStackTrace();

}

Statement stml;

ResultSet rs,rs1;

try {

stml = dbConn.createStatement();

rs = stml.executeQuery(sql);//SQL查询语句

while (rs.next()) {

String passwordCorrect;

String position = rs.getString("Position");

rs1 = stml.executeQuery(sql1);

while (rs1.next()) {

passwordCorrect = rs1.getString("password");

if(passwordInput.equals(passwordCorrect.toString().trim())) {

JOptionPane.showMessageDialog(null, "登陆成功");

//关闭界面，打开新界面

if (position.toString().trim().equals("学生")){

loginID = userInput;

loginName = getStudentName(loginID);

System.out.println(loginName);

System.out.println(loginID);

WindowStudent student = new WindowStudent();

}else if (position.trim().equals("教务管理人员")){

WindowManager manager = new WindowManager();

}else{

loginID = userInput;

loginName = getTeacherName(loginID);

WindowTeacher teacher = new WindowTeacher();

}

frame1.dispose();

}

else{

JOptionPane.showMessageDialog(null, "密码错误", "提示", JOptionPane.ERROR\_MESSAGE);

}

}

}

dbConn.close();

} catch (SQLException ex) {

System.out.println(ex);

}

}

}

else if (e.getSource() == buttonForget) {

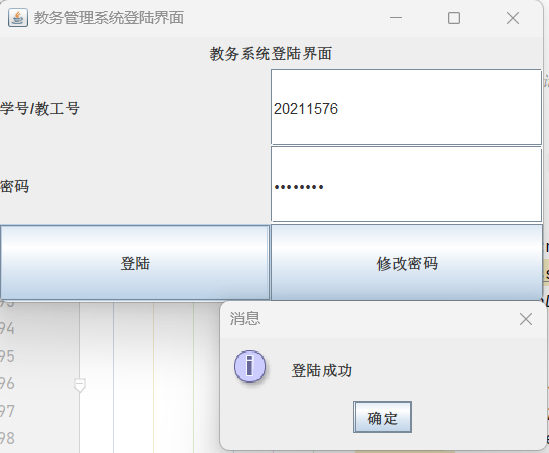
WindowForget forget = new WindowForget();

}

}

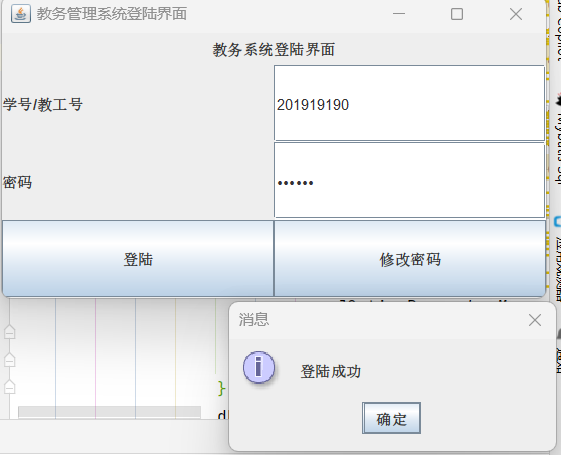
测试结果粘贴；





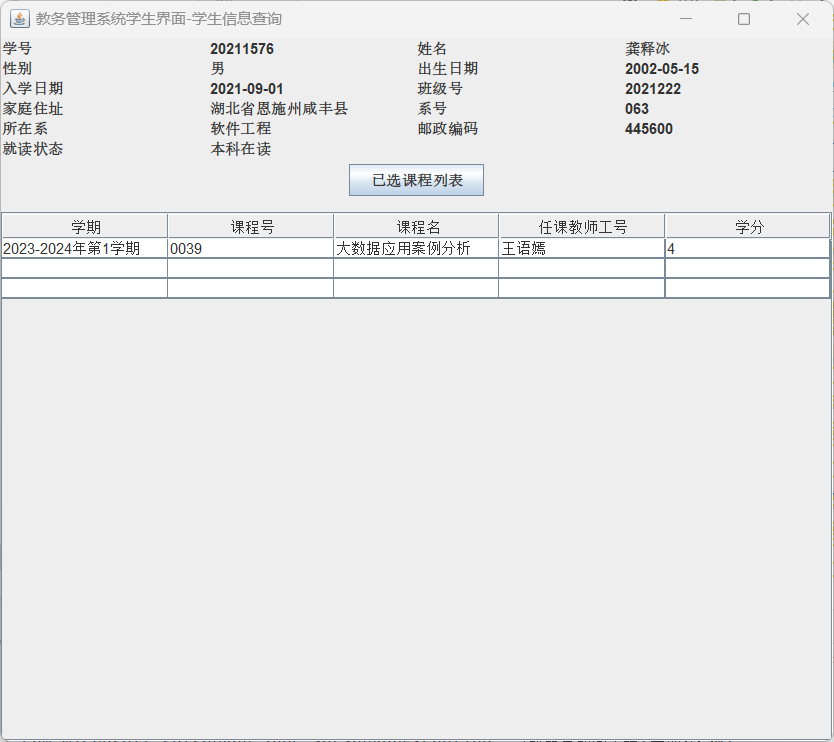






功能模块2：教务管理系统学生界面-

功能界面：



描述：点击学生界面-个人信息+已选课列表按钮，打开如图界面，点击已选课程列表按钮，通过学号查询并输出该学生已选修的所有课程到表中，学号是从登陆界面通过static string变量传过来的。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btnEnquiry) {

String sqlEnquiryStudent = "use Academic\_Affairs\_Management\_System\_20211576 select \* from Student\_Enquiry\_view where Sno = '" + loginID.toString() + "'";

btnEnquiry(sqlEnquiryStudent);

}

}

public void btnEnquiry(String sql){

String Driver2 = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

String url = "jdbc:sqlserver://localhost:1433;DatabaseName=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";

String userName = "s20211576"; // 默认用户名

String userPwd = "s20211576"; // 密码

Connection dbConn = null;

try {

Class.forName(Driver2);

dbConn = DriverManager.getConnection(url, userName, userPwd);

} catch (ClassNotFoundException ea) {

throw new RuntimeException(ea);

} catch (SQLException eb) {

throw new RuntimeException(eb);

}

int columnCount = 0;

try {

Statement stmt = dbConn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

data[0][0] = rs.getString("Semester");

data[0][1] = rs.getString("Cno");

data[0][2] = rs.getString("Cname");

data[0][3] = rs.getString("Emp\_name");

data[0][4] = rs.getString("Credit");

}

} catch (SQLException ex) {

throw new RuntimeException(ex);

} finally {

try {

dbConn.close();

} catch (SQLException exc) {

throw new RuntimeException(exc);

}

}

DefaultTableModel model = new DefaultTableModel(data, columnNames);

table.setModel(model);

}

功能模块3：教务管理系统学生界面-学生选课界面查询

功能界面：



描述：点击学生界面-选课按钮，打开如图界面，点击当前学期内按钮、课程号查询按钮、课程名查询按钮、授课教师查询按钮、学分对应课程按钮，对应查找符合条件的课程，暂时只支持单项查询，复合查询（多个条件一起查询）还没做，系名是通过查询Student\_view中学好对应的系名，学号是从登陆界面通过static string变量传过来的。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btnSemestersEnquire) {

String sql = "SELECT \* FROM Manager\_Course\_Teacher\_manage\_view where Semester='"+comSemester.getSelectedItem()+"' and Sde\_name='"+comSdept.getText().trim()+"' and Semester='"+comSemester.getSelectedItem()+"'";

Connect(sql);

}else if (e.getSource() == btnCnoEnquire) {

String sql = "SELECT \* FROM Manager\_Course\_Teacher\_manage\_view where Cno='" + txtCno.getText().trim() + "'and Sde\_name='"+comSdept.getText().trim()+"' and Semester='"+comSemester.getSelectedItem()+"'";

Connect(sql);

} else if (e.getSource() == btnCnameEnquire) {

String sql = "SELECT \* FROM Manager\_Course\_Teacher\_manage\_view where Cname='" + txtCname.getText().trim() + "'and Sde\_name='"+comSdept.getText().trim()+"' and Semester='"+comSemester.getSelectedItem()+"'";

Connect(sql);

} else if (e.getSource() == btnTeacherEnquire) {

String sql = "SELECT \* FROM Manager\_Course\_Teacher\_manage\_view where Emp\_Name='" + txtEmp\_Name.getText().trim() + "'and Sde\_name='"+comSdept.getText().trim()+"' and Semester='"+comSemester.getSelectedItem()+"'";

Connect(sql);

} else if (e.getSource() == btnCreditEnquire) {

String sql = "SELECT \* FROM Manager\_Course\_Teacher\_manage\_view where Credit='" + txtCredit.getText().trim() + "'and Sde\_name='"+comSdept.getText().trim()+"' and Semester='"+comSemester.getSelectedItem()+"'";

Connect(sql);

}

}

public void Connect(String sql){  
 for (int i = 0; i < getColumns(); i++) {  
 for (int j = 0; j < 8; j++) {  
 data[i][j] = "";  
 }  
 }  
 String Driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";  
 String url = "jdbc:sqlserver://localhost:1433;DatabaseName=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";  
 String userName = "s20211576"; *// 默认用户名* String userPwd = "s20211576"; *// 密码* Connection dbConn = null;  
 try {  
 Class.*forName*(Driver);  
 dbConn = DriverManager.*getConnection*(url, userName, userPwd);  
 Statement stmt = dbConn.createStatement();  
 System.*out*.println(sql);  
 ResultSet rs = stmt.executeQuery(sql);  
 ResultSetMetaData rsmd = rs.getMetaData();  
 int numColumns = rsmd.getColumnCount();  
 int row = 0;  
 while (rs.next()) {  
 for (int col = 0; col < numColumns; col++) {  
 data[row][col] = rs.getString(col + 1);  
 }  
 row++;  
 }  
 rs.close();  
 stmt.close();  
 dbConn.close();  
 } catch (Exception em) {  
 em.printStackTrace();  
 } finally {  
 try {  
 } catch (Exception ec) {  
 ec.printStackTrace();  
 }  
 }  
 DefaultTableModel model = new DefaultTableModel(data, columnNames);  
 table.setModel(model);  
}

测试结果粘贴：











功能模块4：教务管理系统学生界面-学生成绩查询界面

功能界面：



描述：点击学生界面-成绩查询按钮，打开如图界面，点击成绩查询按钮，查找该生选修的所有课程的成绩（通过Sno在Student\_Score\_Enquiry\_view中查询），学号是从登陆界面通过static string变量传过来的。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btnEnquiry) {

String Driver2 = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

String url = "jdbc:sqlserver://localhost:1433;DatabaseName=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";

String userName = "s20211576"; // 默认用户名

String userPwd = "s20211576"; // 密码

String sqlCourseEnquiry = "select \* from Student\_Score\_Enquiry\_view where Sno = '" + loginID + "' and Sname = '" + loginName + "' order by Semester asc";

System.out.println(sqlCourseEnquiry);

Connection dbConn = null;

try {

Class.forName(Driver2);

dbConn = DriverManager.getConnection(url, userName, userPwd);

} catch (ClassNotFoundException e1) {

throw new RuntimeException(e1);

} catch (SQLException e2) {

throw new RuntimeException(e2);

}

try {

Statement stmt = dbConn.createStatement();

ResultSet rs = stmt.executeQuery(sqlCourseEnquiry);

ResultSetMetaData rsmd = rs.getMetaData();

int count = 0;

while (rs.next()&&count<getColumns()) {

data[count][0] = rs.getString("Semester");

data[count][1] = rs.getString("Cno");

data[count][2] = rs.getString("Cname");

data[count][3] = rs.getString("Emp\_name");

data[count][4] = rs.getString("Grade");

count++;

}

} catch (SQLException e3) {

throw new RuntimeException(e3);

} finally {

try {

dbConn.close();

} catch (SQLException e4) {

throw new RuntimeException(e4);

}

}

DefaultTableModel model = new DefaultTableModel(data, columnNames);

table.setModel(model);

}

}

测试结果粘贴：



功能模块5：教务管理系统教师界面授课列表查询

功能界面：



描述：点击教务管理系统登陆界面，打开如图界面，点击授课列表按钮，查找该教师所授的所有课程的信息（通过Emp\_name在Manager\_Course\_Teacher\_manage\_view中查询），教师姓名是从登陆界面通过static string变量loginName传过来的。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btnEnquiry) {

Enquiry();

}

}

public void Enquiry(){

String Driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

String url = "jdbc:sqlserver://localhost:1433;DatabaseName=Academic\_Affairs\_Management\_System\_20211576;encrypt=false";

String sql = "use Academic\_Affairs\_Management\_System\_20211576 select \* from Manager\_Course\_Teacher\_manage\_view where Emp\_name = '" + loginName.trim() + "'";

System.out.println(sql);

String userName = "s20211576"; // 默认用户名

String userPwd = "s20211576"; // 密码

Connection dbConn = null;

try {

Class.forName(Driver);

dbConn = DriverManager.getConnection(url, userName, userPwd);

} catch (ClassNotFoundException ea) {

throw new RuntimeException(ea);

} catch (SQLException eb) {

throw new RuntimeException(eb);

}

int columnCount = 0;

try {

Statement stmt = dbConn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

data[0][0] = rs.getString("Semester");

data[0][1] = rs.getString("Course\_Offered\_Sde\_no");

data[0][2] = rs.getString("Sde\_name");

data[0][3] = rs.getString("Cno");

data[0][4] = rs.getString("Cname");

data[0][5] = rs.getString("Credit");

data[0][6] = rs.getString("Stu\_total");

}

} catch (SQLException ex) {

throw new RuntimeException(ex);

} finally {

try {

dbConn.close();

} catch (SQLException exc) {

throw new RuntimeException(exc);

}

}

DefaultTableModel model = new DefaultTableModel(data, columnNames);

table.setModel(model);

}

测试结果粘贴：



功能模块6：教务管理系统教师界面-对应课程成绩管理界面

功能界面：



描述：点击教务管理系统登陆界面，打开如图界面，选中一列后，点击成绩管理按钮，查找该教师所授的所有课程的成绩（通过Emp\_name在Manager\_Course\_Teacher\_manage\_view中查询）并打开新界面，教师姓名是从登陆界面通过static string变量loginName传过来的，当在教务管理系统教师界面未选中课程直接点击成绩管理时，会弹窗报错。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btnScoreManage){

//如果没有选中行，则弹出提示框

if (table.getSelectedRow() == -1) {

JOptionPane.showMessageDialog(null, "请先选中课程再进行此操作！", "提示", JOptionPane.ERROR\_MESSAGE);

return;

}

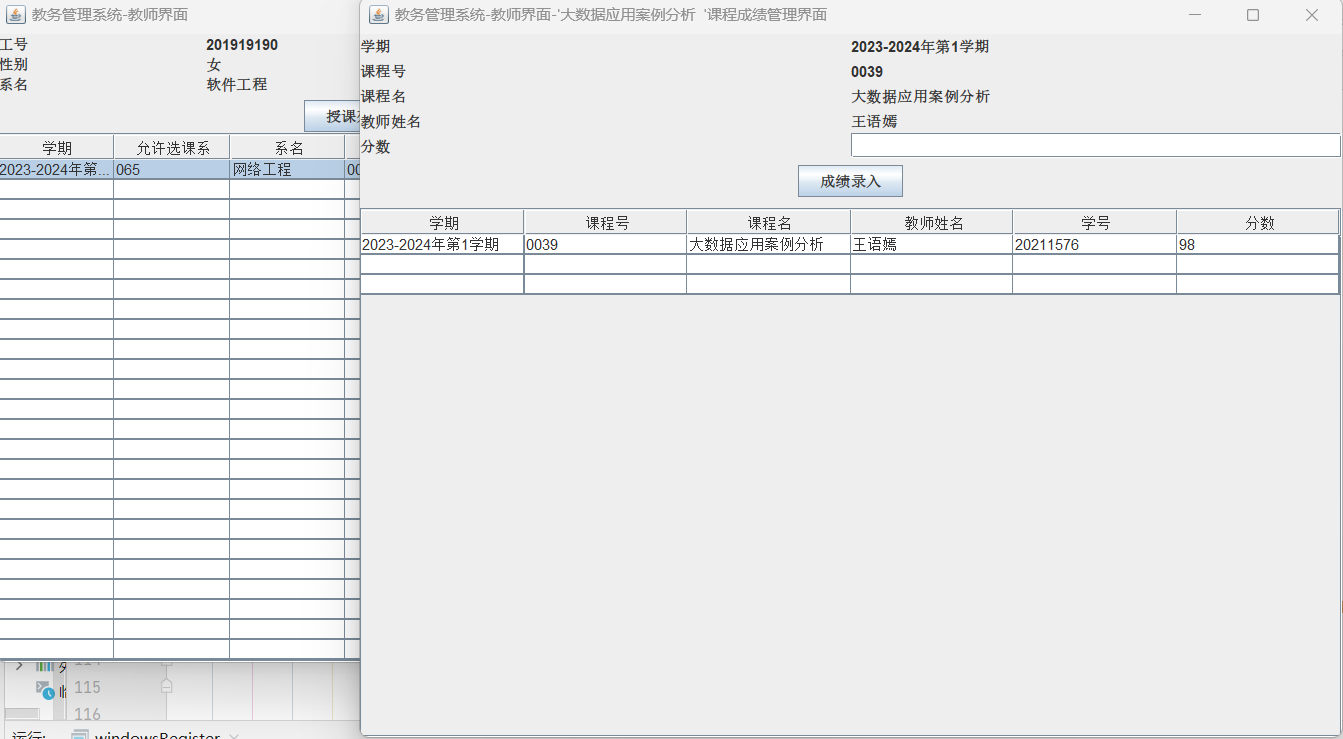
WindowTeacherScoreManage windowTeacherScoreManage = new WindowTeacherScoreManage();

}

}

测试结果截图：





功能模块6：教务管理系统教师界面-对应课程成绩管理界面

功能界面：

描述：点击教务管理系统登陆界面，打开如图界面，选中一列后，点击成绩管理按钮，查找该教师所授的所有课程的成绩（通过Emp\_name在Manager\_Course\_Teacher\_manage\_view中查询）并打开新界面，教师姓名是从登陆界面通过static string变量loginName传过来的，当在教务管理系统教师界面未选中课程直接点击成绩管理时，会弹窗报错。

T-SQL语句与宿主语言嵌套使用代码段（粘贴）：

（7）原子性事务显示定义：无

六、系统详细设计及实现

（1）系统功能模块说明

（2）每个模块的关键语句及关键技术说明

七、总结