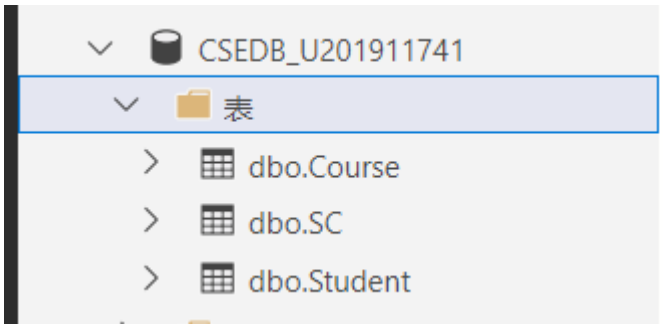
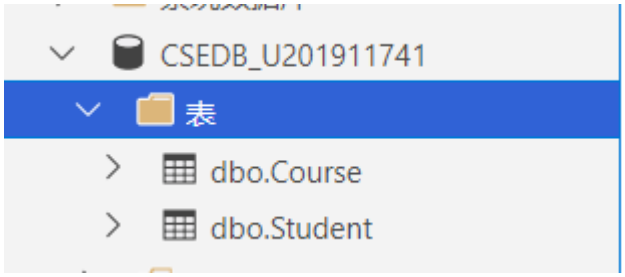
第一次实验：

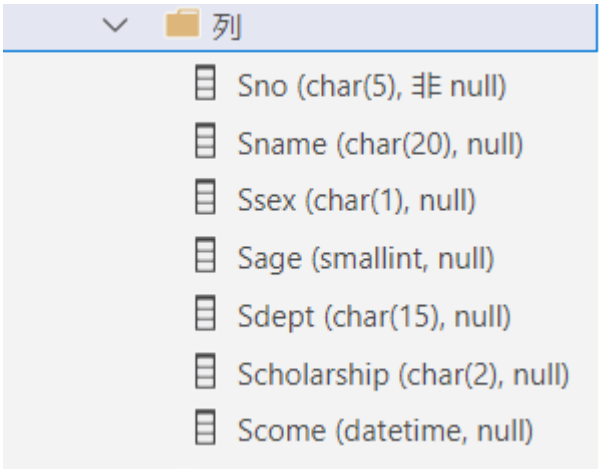
```
Use CSEDB_U201911741;
CREATE TABLE Student
 (
      Sno CHAR(5) NOT NULL UNIQUE,
      Sname CHAR(20) UNIQUE,
      Ssex CHAR(1) ,
      Sage INT,
      Sdept CHAR(15),
      Scholarship CHAR(2)
 )
 CREATE TABLE Course
 (
       Cno CHAR(3),
       Cname CHAR(20),
       Cpno INT,
       Ccredit INT
 )
CREATE TABLE SC(
      Sno CHAR(5),
      Cno CHAR(3),
      Grade int,
      Primary key (Sno, Cno)
);
```



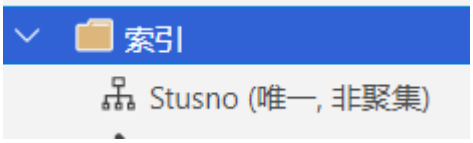

```
DROP TABLE SC;
```



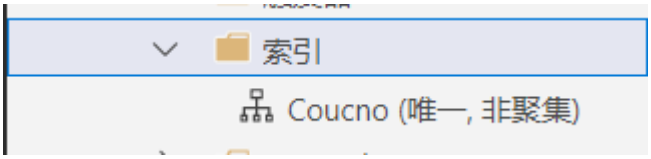
ALTER TABLE Student ADD Scome DATETIME;
ALTER TABLE Student ALTER COLUMN Sage SMALLINT;



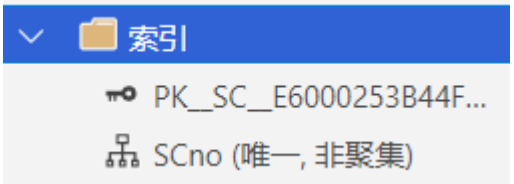
CREATE UNIQUE INDEX Stusno ON Student(Sno);



CREATE UNIQUE INDEX Coucno ON Course(Cno);



CREATE UNIQUE INDEX SCno ON SC(Sno ASC, Cno DESC);



DROP INDEX Stusno on Student;

DROP DATABASE CSEDB_U201911741;

```
/*CREATE DATABASE S_T_U201911741;*/
USE S_T_U201911741;

create table Student
 (Sno CHAR(9) PRIMARY KEY,
```

```
 Sname CHAR(20) UNIQUE,
 Ssex CHAR(2),
 Sage SMALLINT,
 Sdept CHAR(20),
 Scholarship char(2)
 );

/*表 Student 的主码为 Sno，属性列 Sname 取唯一值*/
create table Course
 (Cno CHAR(4) PRIMARY KEY,
 Cname CHAR(40),
 Cpno CHAR(4),
 Ccredit SMALLINT,
 FOREIGN KEY (Cpno) REFERENCES Course(Cno)
 );

/*表 Course 的主码为 Cno，属性列 Cpno(先修课)为外码，被参照表为 Course，被参照列是
Cno*/
create table SC
 (Sno CHAR(9),
 Cno CHAR(4),
 Grade SMALLINT,
 primary key (Sno, Cno),
 FOREIGN KEY (Sno) REFERENCES Student(Sno),
 FOREIGN KEY (Cno) REFERENCES Course(Cno)
 );
```

Name	Schema	Type
Course	dbo	Table
SC	dbo	Table
Student	dbo	Table

```
use S_T_U201911741;
insert into student values('200215121','李勇','男',20,'CS', '否');
insert into student values('200215122','刘晨','女',19,'CS', '否');
insert into student values('200215123','王敏','女',18,'MA', '否');
insert into student values('200215125','张立','男',19,'IS', '否');
insert into course values('1', '数据库', NULL,4);
insert into course values('2', '数学', NULL,2);
insert into course values('3', '信息系统', NULL,4);
insert into course values('4', '操作系统', NULL,3);
insert into course values('5', '数据结构', NULL,4);
insert into course values('6', '数据处理', NULL, 2);
insert into course values('7', 'PASCAL 语言', NULL,4);
```

```
update Course set Cpno = '5' where Cno = '1';
update Course set Cpno = '1' where Cno = '3';
update Course set Cpno = '6' where Cno = '4';
update Course set Cpno = '7' where Cno = '5';
update Course set Cpno = '6' where Cno = '7';

insert into SC values('200215121', '1',92);
insert into SC values('200215121', '2',85);
insert into SC values('200215121', '3',88);
insert into SC values('200215122', '2',90);
insert into SC values('200215122', '3',80);
```

	Sno	Sname	Ssex	Sage	Sdept	Scholarship
1	200215121	李勇 …	男	20	CS …	否
2	200215122	刘晨 …	女	19	CS …	否
3	200215123	王敏 …	女	18	MA …	否
4	200215125	张立 …	男	19	IS …	否

	Sno	Cno	Grade
1	200215121	1	92
2	200215121	2	85
3	200215121	3	88
4	200215122	2	90
5	200215122	3	80

	Cno	Cname	Cpno	Ccredit
1	1	数据库 …	5	4
2	2	数学 …	*NULL*	2
3	3	信息系统 …	1	4
4	4	操作系统 …	6	3
5	5	数据结构 …	7	4
6	6	数据处理 …	*NULL*	2
7	7	PASCAL 语言 …	6	4

use S_T_U201911741;
SELECT Sno,Sname,Ssex,Sage,Sdept FROM Student;

	Sno	Sname	Ssex	Sage	Sdept
1	200215121	李勇	男	20	CS
2	200215122	刘晨	女	19	CS
3	200215123	王敏	女	18	MA
4	200215125	张立	男	19	IS

SELECT Student.Sno, student.Sname
FROM Student, SC
WHERE Student.Sno = SC.Sno AND SC.Cno= ' 2 ' AND SC.Grade > 90;

	Sno	Sname

SELECT Sname,Ssex
FROM Student
WHERE Sdept IN ('IS','MA','CS');

	Sname	Ssex
1	李勇	男
2	刘晨	女
3	王敏	女
4	张立	男

SELECT Sname,Sdept,Sage
FROM Student
WHERE Sage BETWEEN 20 AND 23;

	Sname	Sdept	Sage
1	李勇	CS	20

SELECT Sname,Sno,Ssex
FROM Student
WHERE Sname LIKE '刘%';

	Sname	Sno	Ssex
1	刘晨	200215122	女

```
SELECT Sno,Grade
FROM SC
WHERE Cno= '3'
ORDER BY Grade DESC;
```

	Sno	Grade
1	200215121	88
2	200215122	80

```
SELECT AVG(Grade)
FROM SC
WHERE Cno= '1';
```

	(无列名称)
1	92

```
SELECT Sno
FROM SC
GROUP BY Sno
HAVING COUNT(*) >3;
```

	Sno

```
USE S_T_U201911741;
SELECT Sno, Sname, Sage FROM Student;
```

	Sno	Sname	Sage
1	200215121	李勇	20
2	200215122	刘晨	19
3	200215123	王敏	18
4	200215125	张立	19

```
SELECT * FROM Student WHERE Sdept='CS';
```

	Sno	Sname	Ssex	Sage	Sdept	Scholarship
1	200215121	李勇	男	20	CS	否
2	200215122	刘晨	女	19	CS	否

```
SELECT Sno, Cno, Grade FROM SC WHERE Grade>=90 OR Grade <60;
```

	Sno	Cno	Grade
1	200215121	1	92
2	200215122	2	90

```
SELECT Sname, Ssex, Sage FROM Student WHERE Sage NOT BETWEEN 19 and 20;
```

	Sname	Ssex	Sage
1	王敏	女	18

```
SELECT Sname, Sdept FROM Student WHERE Sdept = 'MA' OR Sdept = 'IS';
```

	Sname	Sdept
1	王敏	MA
2	张立	IS

```
SELECT Cno, Cname, Ccredit FROM Course WHERE Cname LIKE '%数据%';
```

	Cno	Cname	Ccredit
1	1	数据库 …	4
2	5	数据结构 …	4
3	6	数据处理 …	2

```
SELECT DISTINCT Student.Sno, Cno FROM SC, Student WHERE (Student.Sno = SC.Sno AND G
rade IS NULL) OR Student.Sno NOT IN (SELECT Sno From SC);
```

	Sno	Cno
1	200215123	1
2	200215123	2
3	200215123	3
4	200215125	1
5	200215125	2
6	200215125	3

```
SELECT MAX(Grade), MIN(Grade), AVG(Grade) FROM SC WHERE Sno = '200215121';
```

	MAX	MIN	AVG
1	92	85	88

```
SELECT Sno, Grade FROM SC WHERE Cno = '2' ORDER BY Grade ASC;
```

	Sno	Grade
1	200215121	85
2	200215122	90

```
SELECT Sdept, AVG(Sage) FROM Student Group BY Sdept;
```

	Sdept	AVG
1	CS	19
2	IS	19
3	MA	18

```
SELECT AVG(Sage), Sdept FROM Student GROUP BY Sdept HAVING AVG(Sage) <= 19;
```

	AVG	Sdept
1	19	CS
2	19	IS
3	18	MA

第二次实验：

查询每门课程及其被选情况（输出所有课程中每门课的课程号、课程名称、选修该课程的学生学号及成绩--如果没有学生选择该课，则相应的学生学号及成绩为空值）。

```
SELECT Course.Cno, Cname, Sno, Grade FROM Course LEFT OUTER JOIN SC ON (Course.Cno=SC.Cno);
```

	Cno	Cname	Sno	Grade
1	1	数据库 …	200215121	92
2	2	数学 …	200215121	85
3	2	数学 …	200215122	90
4	3	信息系统 …	200215121	88
5	3	信息系统 …	200215122	80
6	4	操作系统 …	NULL	NULL
7	5	数据结构 …	NULL	NULL
8	6	数据处理 …	NULL	NULL
9	7	PASCAL 语言 …	NULL	NULL
1…	8	C语言 …	NULL	NULL

图 3.15 查询结果

查询与“张立”同岁的学生的学号、姓名和年龄。(要求使用至少 3 种方法求解)

```
SELECT Sno, Sname, Sage FROM Student WHERE Sage IN(SELECT Sage FROM Student WHERE S
name='张立');
SELECT Sno, Sname, Sage FROM Student WHERE Sage =(SELECT Sage FROM Student WHERE Sn
ame='张立');
SELECT s1.Sno, s1.Sname, s1.Sage FROM Student s1,Student s2 WHERE s1.Sage=s2.Sage A
ND s2.Sname='张立';
```

	Sno	Sname	Sage
1	200215125	张立 …	21

图 3.16 查询结果

查询选修了 3 号课程而且成绩为良好（80~89 分）的所有学生的学号和姓名。

```
SELECT Student.Sno, Sname FROM Student, SC WHERE Student.Sno=SC.Sno AND SC.Cno='3'
AND SC.Grade BETWEEN 80 AND 90;
```

	Sno	Sname
1	200215121	李勇
2	200215122	刘晨

图 3.17 查询结果

查询学生 200215122 选修的课程号、课程名

```
SELECT SC.Cno,Cname FROM SC,Course WHERE SC.Cno=Course.Cno AND SC.Sno='200215122';
```

	Cno	Cname
1	2	数学 …
2	3	信息系统 …

图 3.18 查询结果

思考：如何查询学生 200215122 选修的课程号、课程名及成绩?

```
SELECT SC.Cno, Cname, Grade FROM SC,Course WHERE SC.Sno='200215122' AND SC.Cno=Cour
se.Cno;
```

	Cno	Cname	Grade
1	2	数学	90
2	3	信息系统	80

图 3.19 查询结果

找出每个学生低于他所选修课程平均成绩 5 分以上的课程号。(输出学号和课程号)

```
/*SELECT x.Sno,x.Cno FROM SC x WHERE x.Grade+5<(SELECT AVG(Grade) FROM SC y WHERE y
.Cno=x.Cno);
```

	Sno	Cno

图 3.20 查询结果

查询比所有男生年龄都小的女生的学号、姓名和年龄。

```
SELECT Sno,Sname,Sage FROM Student WHERE Ssex='女
' AND Sage<ALL(SELECT Sage FROM Student WHERE Ssex='男');
```

	Sno	Sname	Sage
1	200215123	王敏	18

图 3.21 查询结果

查询所有选修了 2 号课程的学生姓名及所在系。

```
SELECT Sname,Sdept FROM Student,SC WHERE SC.Cno='2' AND Student.Sno=SC.Sno;*/
```

	Sname	Sdept
1	李勇	CS
2	刘晨	CS

图 3.22 查询结果

使用 update 语句把成绩为良的学生的年龄增加 2 岁，并查询出来。

```
/*UPDATE Student SET Sage=Sage+2 WHERE Sno IN(SELECT Sno FROM SC WHERE  Grade BETWE
EN 80 AND 90);

SELECT Student.Sno,Sage,Grade FROM Student,SC WHERE Student.Sno=SC.Sno AND Grade BE
TWEEN 80 AND 90;
```

	Sno	Sage	Grade
1	200215121	24	85
2	200215121	24	88
3	200215122	23	90
4	200215122	23	80

图 3.23 查询结果

使用 insert 语句增加两门课程：C 语言和人工智能，并查询出来

```
INSERT INTO Course(Cno,Cname) VALUES('8','C 语言'),('9','人工智能');
SELECT * FROM Course WHERE Cname='C 语言' OR Cname='人工智能';
```

	Cno	Cname	Cpno	Ccredit
1	8	C语言 ...	NULL	NULL
2	9	人工智能 ...	NULL	NULL

图 3.24 查询结果

使用 delete 语句把人工智能课程删除，并查询出来。

```
DELETE FROM Course WHERE Cname='人工智能';
SELECT Cname FROM Course;*/
```

	Cname
1	数据库 ...
2	数学 ...
3	信息系统 ...
4	操作系统 ...
5	数据结构 ...
6	数据处理 ...
7	PASCAL 语言 ...

图 3.25 查询结果

第三次实验：

(1)创建 CS 系的视图 CS_View

```
CREATE VIEW CS_VIEW AS (SELECT * FROM Student WHERE Sdept='CS');
SELECT * FROM CS_VIEW;
```

	Sno	Sname	Ssex	Sage	Sdept	Scholarship
1	200215121	李勇 ...	男	24	CS ...	否
2	200215122	刘晨 ...	女	23	CS ...	否

(2)在视图 CS_View 上查询 CS 系选修了 1 号课程的学生

```
SELECT SC.Sno,CS_VIEW.Sname FROM SC, CS_VIEW WHERE SC.Cno='1' AND SC.Sno=CS_VIEW.Sn
o;
```

	Sno	Sname
1	200215121	李勇

(3)创建 IS 系成绩大于 80 的学生的视图 IS_View

```
CREATE VIEW IS_VIEW
AS
(SELECT Student.Sno,Sname, Ssex, Sage, Sdept, Scholarship, Cno, Grade FROM Student,
 SC WHERE SC.Grade>80 AND SC.Sno=Student.Sno AND Sdept='IS');
```

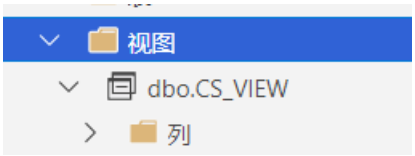
(4)在视图 IS_View 查询 IS 系成绩大于 80 的学生

Sno	Sname	Ssex	Sage	Sdept	Scholarship	Cno	Grade

为空视图

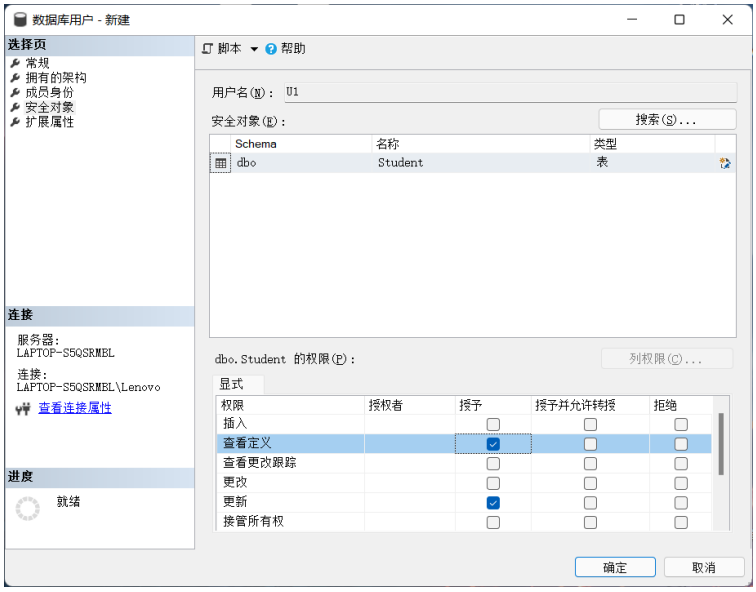
(5)删除视图 IS_View

```
DROP VIEW IS_VIEW;
```

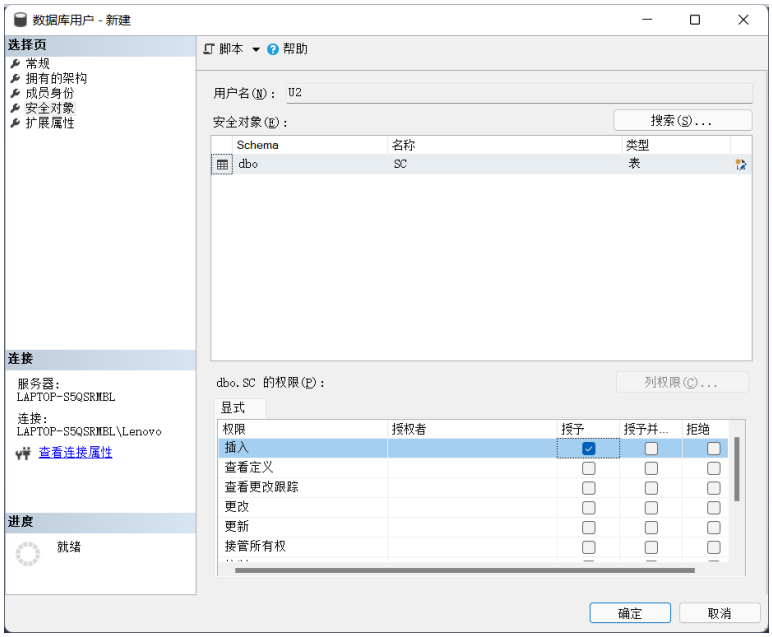


(6) 利用可视化窗口创建 2 个不同的用户 U1 和 U2,利用系统管理员给 U1 授予 Student 表的 查询和更新的权限，给 U2 对 SC 表授予插入的权限。然后用 U1 登录，分别 1）查询学生表 的信息；2）把所有学生的年龄增加 1 岁，然后查询；3）删除 IS 系的学生；4）查询 CS 系 的选课信息。用 U2 登录，分别 1）在 SC 表中插入 1 条记录（'200215122'，'1'，75)；2） 查询 SC 表的信息，3）查询视图 CS_View 的信息。

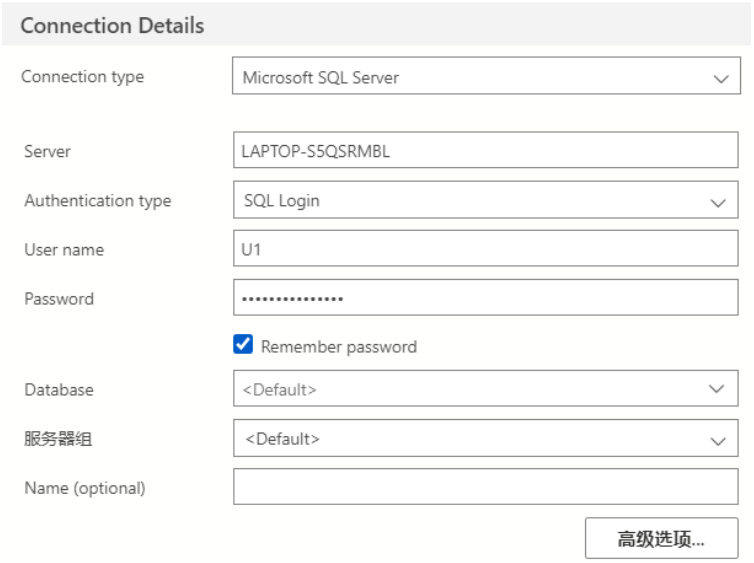
创建 U1：


数据库用户 - 新建
选择页
常规
拥有的架构
成员身份
安全对象
扩展属性
脚本
帮助
用户名(N)： U1
安全对象(E)：
搜索(S)...
Schema
名称
类型
dbo
Student
表
连接
服务器：
LAPTOP-S5QSRMBL
连接：
LAPTOP-S5QSRMBL\Lenovo
查看连接属性
进度
就绪
dbo.Student 的权限(P)：
列权限(C)...
显式
权限
授权者
授予
授予并允许转授
拒绝
插入
查看定义
查看更改跟踪
更改
更新
接管所有权
确定
取消

创建 U2:


数据库用户 - 新建
选择页
常规
拥有的架构
成员身份
安全对象
扩展属性
脚本
帮助
用户名(N)： U2
安全对象(E)：
搜索(S)...
Schema
名称
类型
dbo
SC
表
连接
服务器：
LAPTOP-S5QSRMBL
连接：
LAPTOP-S5QSRMBL\Lenovo
查看连接属性
进度
就绪
dbo.SC 的权限(P)：
列权限(C)...
显式
权限
授权者
授予
授予并...
拒绝
插入
查看定义
查看更改跟踪
更改
更新
接管所有权
确定
取消

用 U1 登录:



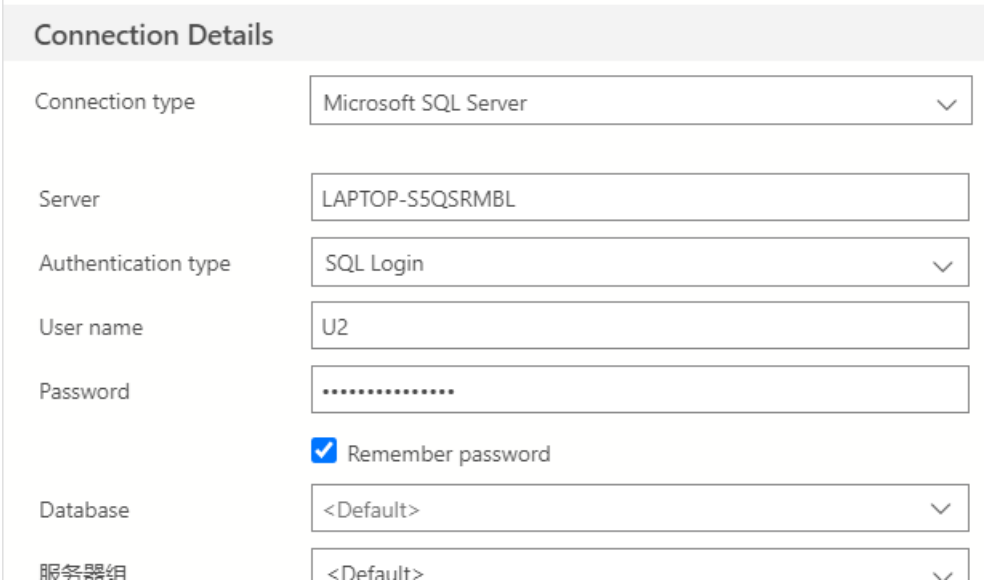
查询学生表信息:

	Sno	Sname	Ssex	Sage	Sdept	Scholarship
1	200215121	李勇	男	24	CS	否
2	200215122	刘晨	女	23	CS	否
3	200215123	王敏	女	18	MA	否
4	200215125	张立	男	21	IS	否

更新后查询:

	Sno	Sname	Ssex	Sage	Sdept	Scholarship
1	200215121	李勇	男	25	CS	否
2	200215122	刘晨	女	24	CS	否
3	200215123	王敏	女	19	MA	否
4	200215125	张立	男	22	IS	否

登录 U2:



插入记录：

```
INSERT INTO SC VALUES('200215122', '1', 75);
```

```
ges
下9:34:54    Started executing query at Line 1
             (1 行受到影响)
             Total execution time: 00:00:00.007
```

查询 CS_VIEW 的结果：（提示没有权限）

```
SELECT * FROM CS_VIEW;
```

```
ges
午9:36:23    Started executing query at Line 1
             Msg 229, Level 14, State 5, Line 29
             拒绝了对对象 'CS_VIEW' (数据库 'S_T_U201911741'，架构 'dbo')的 SELECT 权限。
             Total execution time: 00:00:00.001
```

(7) 用系统管理员登录，收回 U1 的所有权限

```
revoke select,update on student from U1;
```

```
ges
下9:41:37    Started executing query at Line 1
             命令已成功完成。
             Total execution time: 00:00:00.011
```

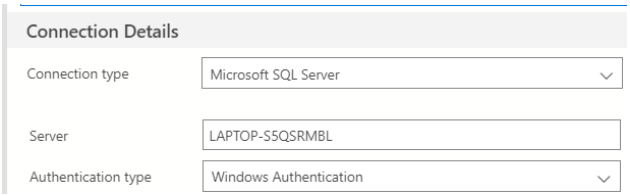
(8) 用 U1 登录，查询学生表的信息

```
SELECT * FROM Student;
```

```
jes
下9:42:26    Started executing query at Line 1
             Msg 229, Level 14, State 5, Line 33
             拒绝了对对象 'Student' (数据库 'S_T_U201911741'，架构 'dbo')的 SELECT 权限。
             Total execution time: 00:00:00
```

提示没有权限

(9) 用系统管理员登录



以本机账户登录

(10) 对 SC 表建立一个更新触发器，当更新了 SC 表的成绩时，如果更新后的成绩大于等于 95，则检查该成绩的学生是否有奖学金，如果奖学金是“否”，则修改为“是”。如果修改后的 成绩小于 95，则检查该学生的其他成绩是不是有大于 95 的，如果都没有，且修改前的成绩 是大于 95 时，则把其奖学金修改为”否”。然后进行成绩修改，并进行验证是否触发器正确 执行。1）首先把某个学生成绩修改为 98，查询其奖学金。2）再把刚才的成绩修改为 80， 再查询其奖学金。

创建触发器:

```
GO
CREATE TRIGGER SC_T
   ON  SC FOR UPDATE
AS
    DECLARE @Grade smallint;
    SELECT @Grade = Grade FROM INSERTED;
BEGIN
    IF(@Grade>=95)
    BEGIN
        UPDATE Student SET Scholarship = ’是’ WHERE Sno in (SELECT Sno FROM
inserted);
        SELECT * FROM Student;
        SELECT * FROM inserted;
    END
    ELSE
    BEGIN
        UPDATE Student SET Scholarship = ’否’ WHERE NOT EXISTS (SELECT * FROM
SC WHERE Sno in (SELECT Sno FROM inserted) AND Grade>=95) AND Sno in (SELECT
Sno FROM inserted);
    END
END;
```

修改，查询:

```
UPDATE SC SET Grade=95 WHERE Sno = ’200215121’ AND Cno = ’1’;
SELECT Sno, Sname, Scholarship FROM Student;
```

	Sno	Sname	Ssex	Sage	Sdept	Scholarship
1	200215121	李勇	男	25	CS	是
2	200215122	刘晨	女	24	CS	否
3	200215123	王敏	女	19	MA	否
4	200215125	张立	男	22	IS	否

	Sno	Cno	Grade
1	200215121	1	95

（11）删除刚定义的触发器

```
DROP TRIGGER SC_T;
```

（12）定义一个存储过程计算 CS 系的课程的平均成绩和最高成绩，在查询分析器或查询编 辑器中执行存储过程，查看结果。

定义存储过程：

```
GO
create procedure getCSAvgMax
as
begin
select sc.Sno,AVG(Grade) AVG,MAX(Grade) MAX from SC,Student where Student.Sno=SC.Sn
o and Sdept='CS' group by SC.Sno;
end;
```

使用 `EXEC getCSAvgMax;` 执行，

执行结果：

	Sno	AVG	MAX
1	200215121	89	95
2	200215122	81	90

（13）定义一个带学号为参数的查看某个学号的所有课程的成绩，查询结果要包含学生姓名。进行验证。

定义存储过程：

```
GO
CREATE PROCEDURE getCourseGrade
@Sno char(9)
AS
BEGIN

SELECT Sname, Cno, Grade FROM Student, SC WHERE Student.Sno=@Sno AND SC.Sno=@Sno;
end;
```

验证：`EXEC getCourseGrade @Sno='200215121';`

	Sname	Cno	Grade
1	李勇	1	95
2	李勇	2	85
3	李勇	3	88

（14）把上一题改成函数。再进行验证。

定义表值函数:

```
GO
CREATE FUNCTION getGrade(@Sno char(9))
RETURNS @test table(sno char(9), name char(20), grade smallint)
AS
BEGIN
    INSERT @test SELECT Sname, Cno, Grade FROM Student, SC WHERE Student.Sno=@Sno
AND SC.Sno=@Sno;
    RETURN
end;
```

调用语句如下:

```
SELECT * from [dbo].getGrade('200215121');
```

	sno	name	grade
1	李勇	1	95
2	李勇	2	85
3	李勇	3	88

（15）在 SC 表上定义一个完整性约束，要求成绩再 0-100 之间。定义约束前，先把某个学 生的成绩修改成 120，进行查询，再修改回来。定义约束后，再把该学生成绩修改为 120，

然后进行查询。

修改查询:

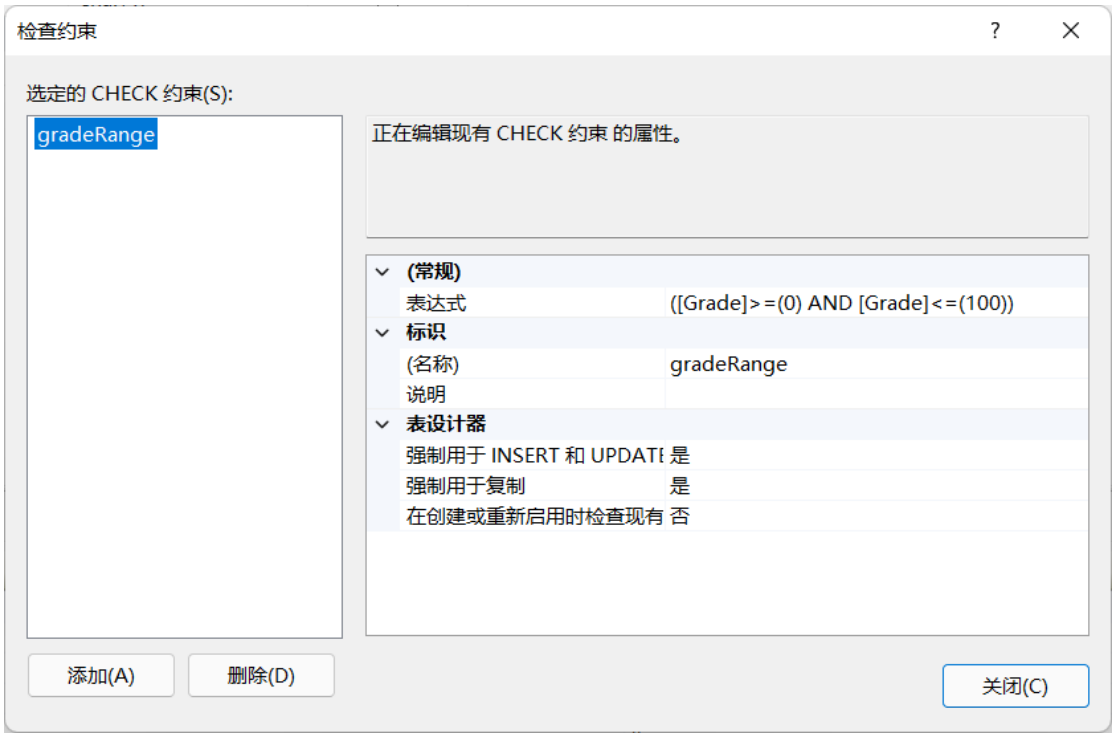
```
UPDATE SC SET Grade=120 WHERE Sno='200215121' AND Cno=1;
SELECT * FROM SC WHERE Sno='200215121' AND Cno=1;
```

	Sno	Cno	Grade
1	200215121	1	120

定义完整性约束:

```
ALTER TABLE SC ADD CONSTRAINT gradeRange CHECK(Grade BETWEEN 0 AND 100);
```

但是表中已有数据，会产生错误，故用可视化界面进行添加



再次修改时，会提示与完整性约束冲突：

```
Started executing query at Line 1
Msg 547, Level 16, State 0, Line 106
UPDATE 语句与 CHECK 约束"gradeRange"冲突。该冲突发生于数据库"S_T_U201911741"，表"dbo.SC", column 'Grade'。
语句已终止。
(1 行受到影响)
```

第四次实验

config.py:

```
from os import execl
import pymssql

db_config = {
    'host': 'LAPTOP-S5QSRMBL',
    'user': 'root',
    'password': 'root',
    'db': 'S_T_U201911741',
    'charset': 'cp936',
    #'cursorclass': pymssql.cursors.DictCursor
}
Student = {
    '11741',
    '鄢涌棚',
    '男',
    '20',
```

```
        'cse',
        '是'
}
Student_list = (
        'Sno',
        'Sname',
        'Ssex',
        'Sage',
        'Sdept',
        'Scholarship'
)
Course_list = (
        'Cno',
        'Cname',
        'Cpno',
        'Ccredit'
)

def printc(str):
        print(str.encode('latin-1').decode('gbk'), end="")

def Add_student_info(cursor,conDB):
        print ("请输入学生信息（学号，姓名，性别，年龄，专业，奖学金获得情况）")
        stu_in = input()
        Student = stu_in.split()
        #print (Student)
        sql_add = "insert into student values('" + Student[0] + "','" + \
                Student[1] + "','" + Student[2] + "','" + Student[3] \
                        + "','" + Student[4] + "','" + Student[5] + "');"
        #print (sql_add)
        cursor.execute(sql_add)
        conDB.commit()
        return

def Update_student_info(cursor,conDB):
        print("请输入要修改的内容（0.学号 1.姓名 2.性别 3.年龄 4.专业 5.奖学金）(先输入
序号，再输入内容)")
        flag = input()
        item = flag.split()
        item[0] = int(item[0])
        if(item[0] != 3):
                item[1] = "'" + item[1] + "'"
        #print (item)
        print("请输入要修改的学生的学号或姓名")
```

```
    opt = input()
    if(opt.isdigit()):
        sno = opt
        sql_update = "update student set " + Student_list[item[0]] + " = " + item[1] +
" where Sno = '" + sno + "';"
    else:
        sname = opt
        sql_update = "update student set " + Student_list[item[0]] + " = " + item[1] +
" where Sname = '" + sname + "';"
    #print (sql_update)
    cursor.execute(sql_update)
    conDB.commit()
    return

def Add_new_course(cursor,conDB):
    print("请输入课程信息（课程号，课程名，先修课程（无则填 NULL），学分）")
    course_in = input()
    course = course_in.split()
    #print(course)
    sql_add = "insert into course values('" + course[0] + "', '" + course[1] + "', " +
course[2] + "," + course[3] + ");"
    #print (sql_add)
    cursor.execute(sql_add)
    conDB.commit()
    return

def Update_course_info(cursor,conDB):
    print("请输入修改课程的课程号或课程名")
    info_in = input()
    print("请输入要修改的内容（0.课程 1.课程名 2.先修课程 3.学分）")
    flag = input()
    item = flag.split()
    item[0] = int(item[0])
    if(item[0] != 3):
        item[1] = "'" + item[1] + "'"
    if(info_in.isdigit()):
        cno = info_in
        sql_update = "update course set " + Course_list[item[0]] + " = " + item[1] + "
where Cno = '" + cno + "';"
    else:
        cname = info_in
        sql_update = "update course set " + Course_list[item[0]] + " = " + item[1] + "
where Cname = '" + cname + "';"
    #print (sql_update)
```

```
    cursor.execute(sql_update)
    conDB.commit()
    return

def Delete_not_select_course(cursor,conDB):
    sql = "delete from course where Cno not in(select distinct Cno from SC);"
    cursor.execute(sql)
    conDB.commit()
    print("Cleanup complete.")
    return

def Add_student_grades(cursor,conDB):
    print("请输入学生学号，课程号，成绩（输入 0 停止输入）")
    while(True):
        a = input()
        if (a == '0'):
            break
        SC = a.split()
        sql_add = "insert into SC values('" + SC[0] + "', '" + SC[1] + "', " + SC[2] + ");"
        #print (sql_add)
        cursor.execute(sql_add)
        conDB.commit()
    return

def Update_student_grades(cursor, conDB):
    print("请输入要修改的学生学号，课程号和成绩")
    SC_in = input()
    SC = SC_in.split()
    sql_update = "update SC set grade = " + SC[2] + " where Sno = '" + SC[0] + "' and
Cno = '" + SC[1] + "';"
    #print(sql_update)
    cursor.execute(sql_update)
    conDB.commit()
    return

def Get_dept_statistics(cursor,conDB):
    sql1 = "select distinct Sdept from student;"
    cursor.execute(sql1)
    dept_get = cursor.fetchall()
    size = len(dept_get)
    for i in range (0,size):
        temp = str(dept_get[i])
        dept = temp[2:4]
        if(temp[4].isalpha()):
```

```
            dept = dept + temp[4]
        print(dept)
        #最大，最小，平均
        sql_max_min_avg = "SELECT MAX(Grade) MAX, MIN(Grade) MIN, AVG(Grade)
AVG FROM SC where Sno in (select Sno from student where Sdept = '"+dept+"');"
        cursor.execute(sql_max_min_avg)
        max_min_avg = cursor.fetchall()

print("max_min_avg:",max_min_avg[0][0],max_min_avg[0][1],max_min_avg[0][2])
        #不及格人数
        sql_failed = "select count(*) from SC, Student where grade<60 and
Sc.Sno=Student.sno and Student.Sdept='"+dept+"';"
        cursor.execute(sql_failed)
        failed = cursor.fetchall()
        print("failed:",failed[0][0])
        #优秀率
        sql2 = "select count(*) from student where Sdept = '"+dept+"';"
        cursor.execute(sql2)
        total = cursor.fetchall()
        print("total:",total[0][0])
        sql3 = "select count(*) from SC, Student where grade >= 80 and
SC.Sno=Student.Sno and Student.Sdept = '"+dept+"';"
        cursor.execute(sql3)
        execelent = cursor.fetchall()
        print("execelent:",execelent[0][0])
        a = float(total[0][0])
        b = float(execelent[0][0])
        print("execelent rate:",b/a)
        print("")
    return

def Get_grade_order(cursor, conDB):
    sql1 = "select distinct Sdept from student;"
    cursor.execute(sql1)
    dept_get = cursor.fetchall()
    size = len(dept_get)
    for i in range (0,size):
        temp = str(dept_get[i])
        dept = temp[2:4]
        if(temp[4].isalpha()):
            dept = dept + temp[4]
        sql2 = "select Sname, Cname, grade from SC, student, Course where SC.Sno =
Student.Sno and SC.Cno = Course.Cno and Sdept = '" + dept + "' order by grade desc;"
        cursor.execute(sql2)
```

```
            ret = cursor.fetchall()
            i = len(ret)
            print(dept)
            if(i != 0):
                for j in range(0,i):
                    printc(ret[j][0])
                    printc(ret[j][1])
                    print(ret[j][2])
            else:
                print("没有学生成绩")
        return

def Get_stu_info(cursor,conDB):
        Sno = input("请输入学生学号:")
        sql1 = "select * from student where Sno = '"+Sno+"';"
        cursor.execute(sql1)
        base_info = cursor.fetchall()
        sql2 = "select SC.Cno, Cname from SC, Course where Sno = '"+Sno+"' and SC.Cno
= Course.Cno;"
        cursor.execute(sql2)
        course_info = cursor.fetchall()
        print("基础信息： ")
        printc(base_info[0][1])
        printc(base_info[0][2])
        print("\t\t",base_info[0][3],"\t\t",end="")
        printc(base_info[0][4])
        printc(base_info[0][5])
        print("")
        if(len(course_info) == 0):
            print("该学生没有选课")
        else:
            print("选课信息")
            print(course_info[0][0], end="")
            printc(course_info[0][1])
            print("\n")
        return

def Show_menu():
        print("""
        功能菜单：(输入对应数字选择)
        1.添加学生信息
        2.修改学生信息
        3.添加新课程
        4.修改课程信息
```

```
5.录入学生成绩
6.修改学生成绩
7.统计学生的平均成绩、最好成绩、最差成绩、优秀率、不及格人数
8.显示学生排名
9.查询学生信息
0.退出
""")
```

main.py:

```
import sys
from config import *
import pymssql
print(sys.getdefaultencoding())

conDB = pymssql.connect(db_config['host'], db_config['user'], db_config['password'], db_config['db'], db_config['charset'])

if conDB:
    print("连接成功")
    Show_menu()
    cursor = conDB.cursor()
    opt = input()
    while(opt != '0'):
        if (opt == '1'):
            Add_student_info(cursor,conDB)
        elif(opt == '2'):
            Update_student_info(cursor,conDB)
        elif(opt == '3'):
            Add_new_course(cursor,conDB)
        elif(opt == '4'):
            Update_course_info(cursor,conDB)
        elif(opt == '5'):
            Add_student_grades(cursor, conDB)
        elif(opt == '6'):
            Update_student_grades(cursor, conDB)
        elif(opt == '7'):
            Get_dept_statistics(cursor, conDB)
        elif(opt == '8'):
            Get_grade_order(cursor, conDB)
        elif(opt == '9'):
            Get_stu_info(cursor,conDB)
        elif(opt == '0'):
```

```
                break
            print("******操作完成******")
            Show_menu()
            opt = input()
else:
    print ("连接失败")
# 连接用完后记得关闭以释放资源
conDB.close()
```