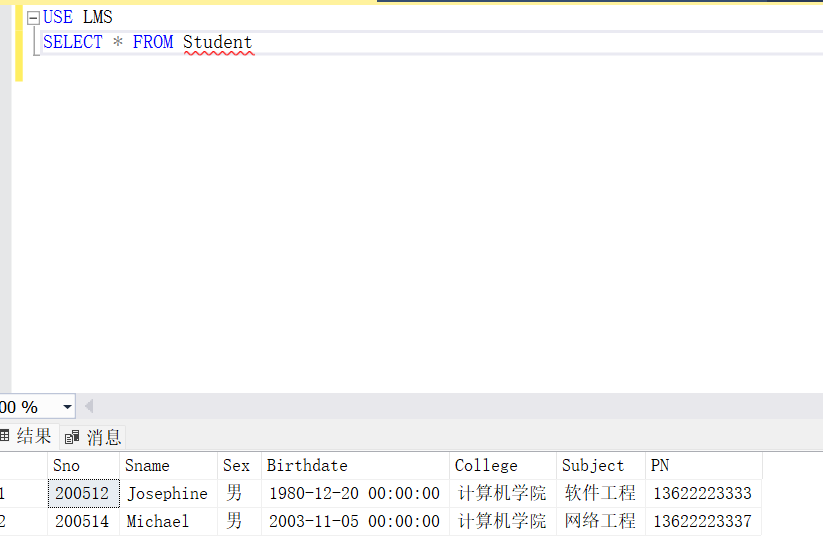
数据库作业：

实验四：

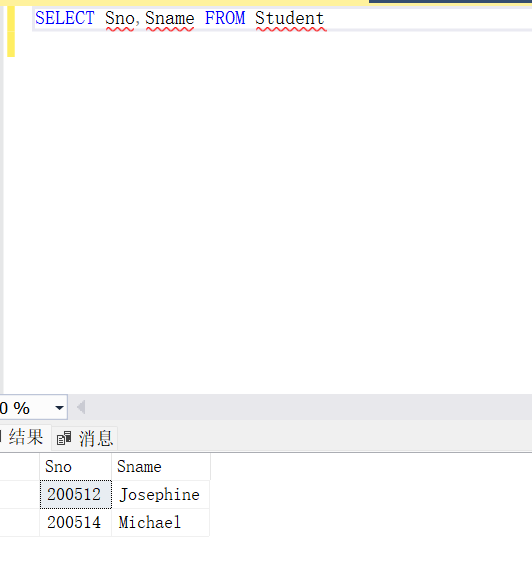
验证性实验：

1）无条件查询全部数据

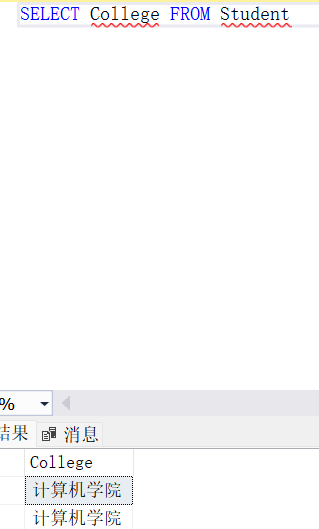


2）在SELECT关键字后指明要检索的列名

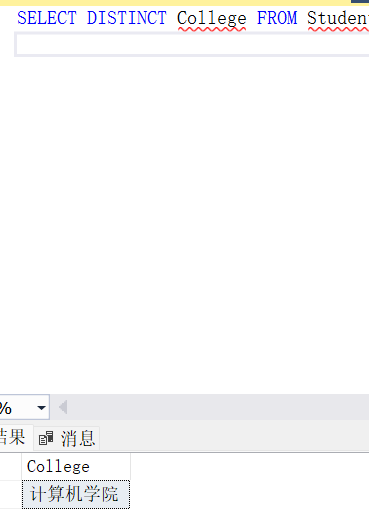
* 1. 查询student表的学生学号和姓名



* 1. 查询student表中的学院信息

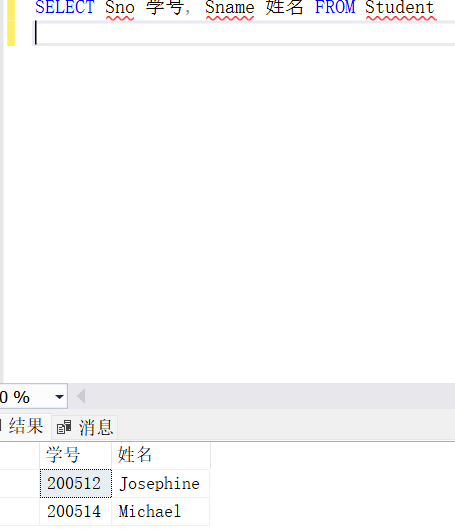


* 1. 查询student表中不重复的学院信息

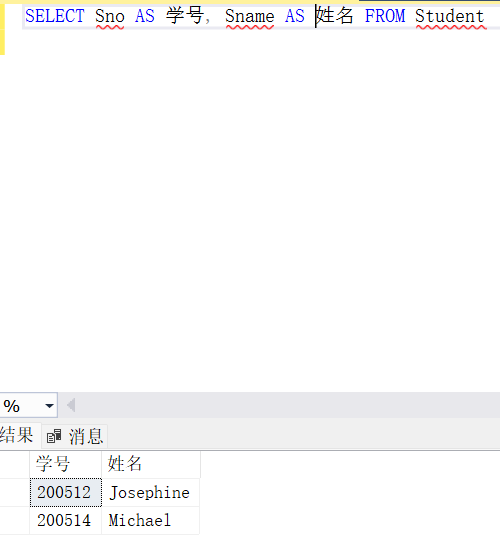


3）改变列标题的检索

①使用 空格形式： 列名 新标题

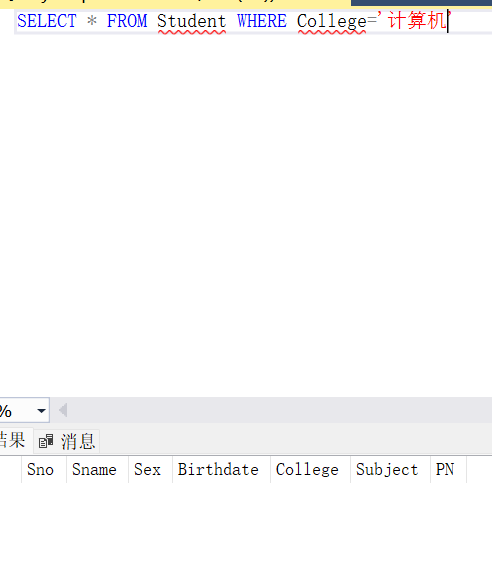


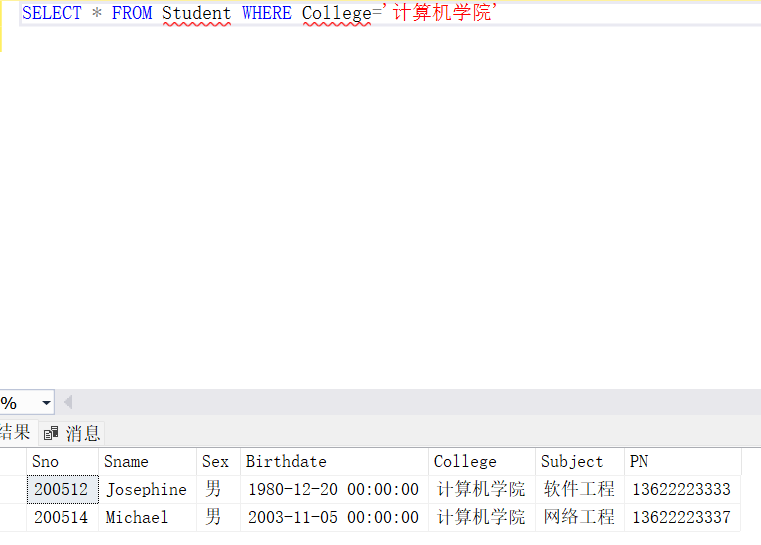
②使用“AS”形式，列名 AS 新标题



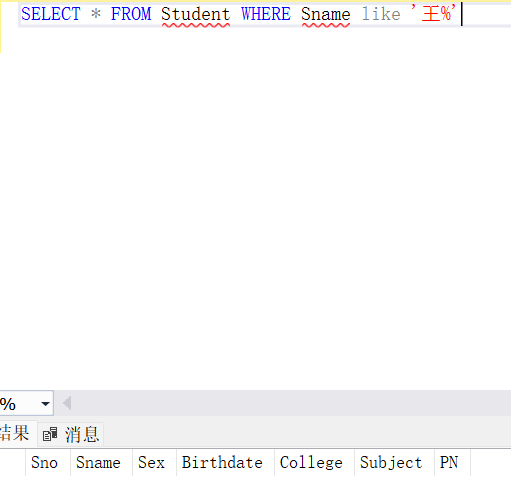
4） 有条件选择的查询

①在student表检索“计算机”学院的学生信息

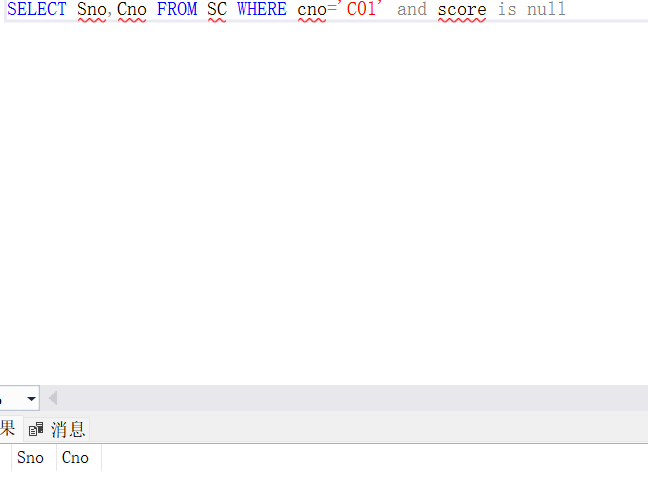




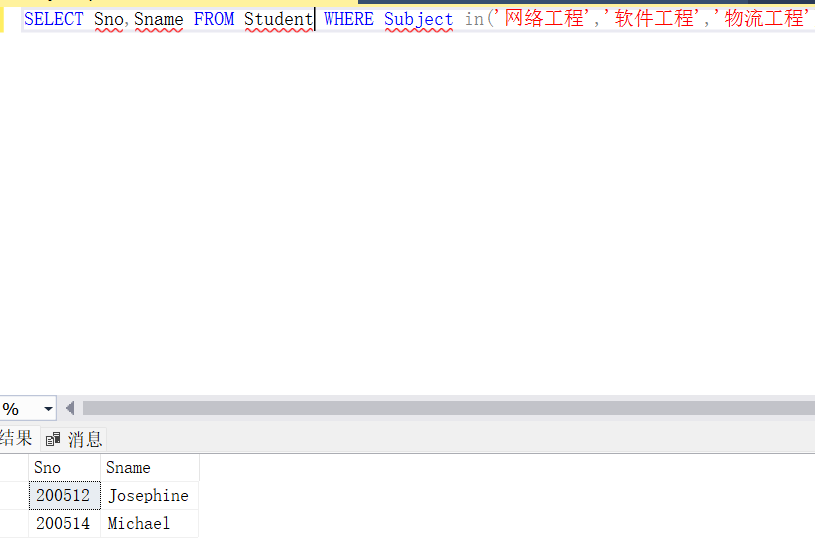
②在student表中检索姓“王”的学生信息



③在SC表检索’C01’选修课成绩为空的选课记录



④检索专业为网络工程，软件工程, 物流工程的学生学号、姓名

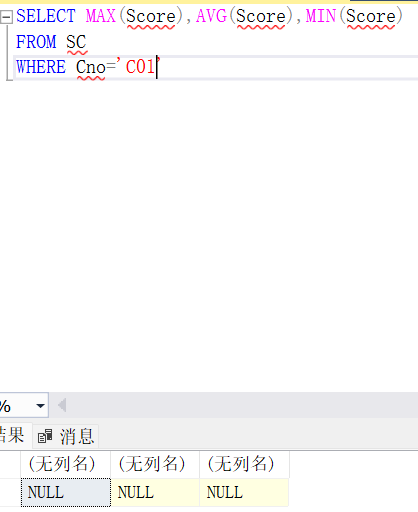


5）使用聚集函数

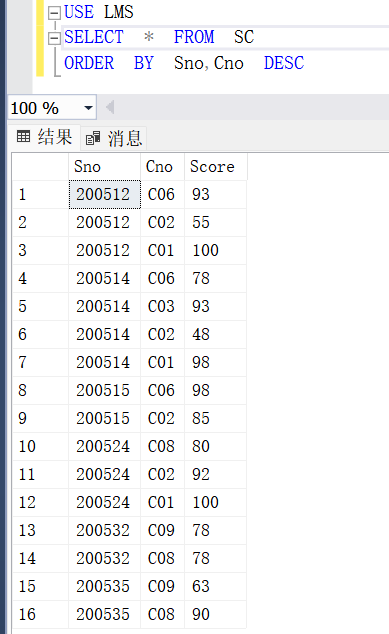
①查询选课表中最高分、平均分、最低分



②查询“C01”课程的最高分、平均分和最小成绩。

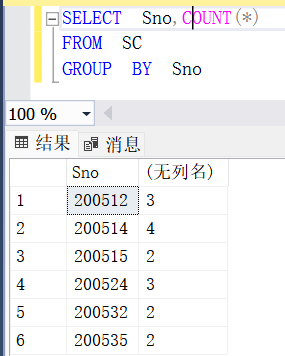


1. 对检索结果进行排序

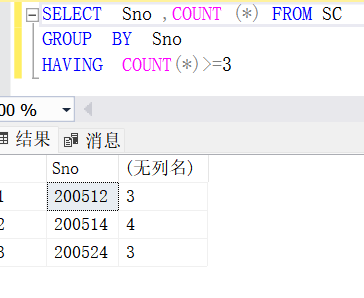


7）进行分组统计：

①查询各学生的选课数

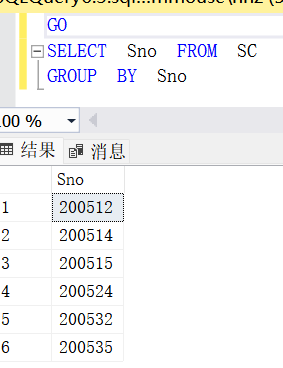
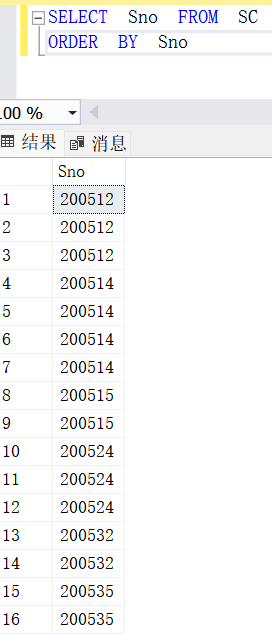


②使用HAVING ：“选课表”中查询选修了3 门以上课程的学生学号。



③“选课表”中按学号分组汇总学生的平均分，并按平均分降序排列。



④分析下面两个SELECT语句执行结果，说明有什么不同？

第二个有GROUP分组，将相同的Sno合在一起，从而比结果1少了重复的Sno数据。

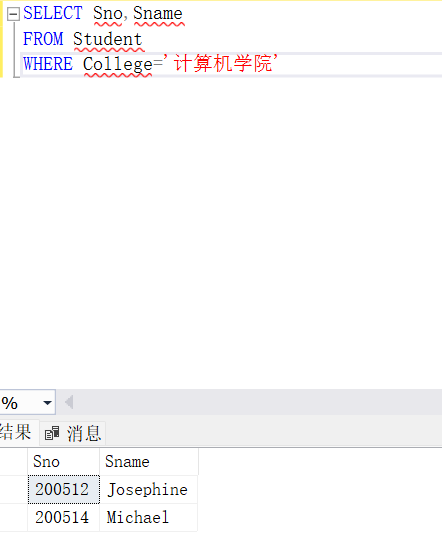
设计性实验：

（1）查询计算机学院学生的学号和姓名。

SELECT Sno,Sname

FROM Student

WHERE College='计算机学院'



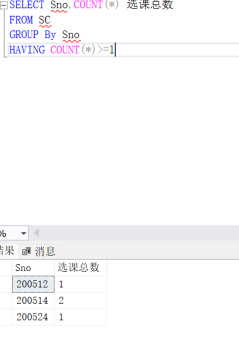
（2）查询选修了课程的学生学号。

SELECT Sno,COUNT(\*) 选课总数

FROM SC

GROUP BY Sno

HAVING COUNT(\*)>=1



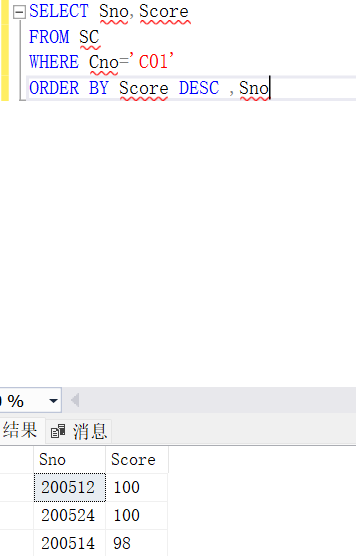
（3）查询选修C01 课程的学生学号和成绩，并要求对查询结果按成绩的降序排列，如果成绩相同则按学号的升序排列。

SELECT Sno,Score

FROM SC

WHERE Cno='C01'

ORDER BY Score DESC,Sno

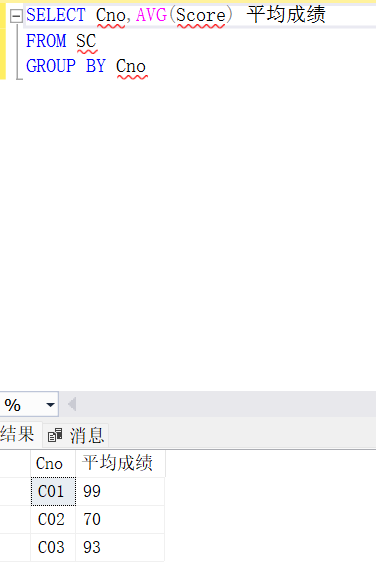


（4）查询每门课程的平均分。

SELECT Cno,AVG(Score) 平均成绩

FROM SC

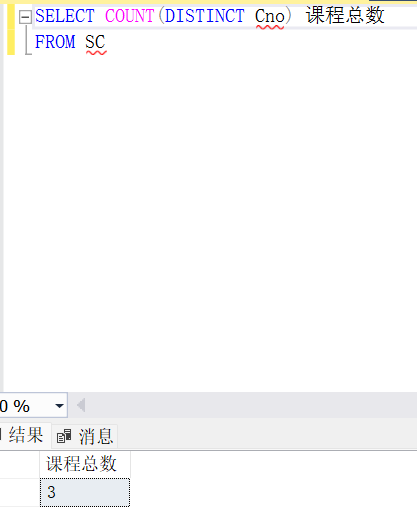
GROUP BY Cno



（5）查询学校开设的课程总数。

SELECT COUNT(DISTINCT Cno) 课程总数

FROM SC



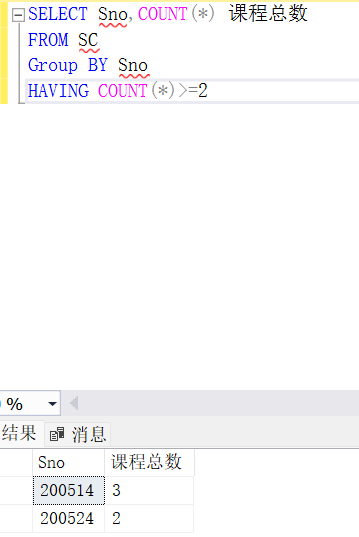
（6）查询选修两门及两门以上课程的学生学号。

SELECT Sno,COUNT(\*)课程总数

FROM SC

GROUP BY Sno

HAVING COUNT(\*)>=2

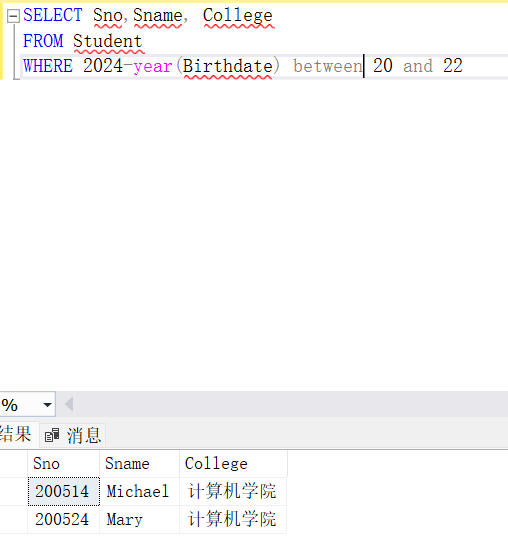


（7）查询年龄在20-22之间的男生的学号、姓名、学院。（注意表中没有年龄字段）

SELECT Sno,Sname,College

FROM Student

WHERE 2024-YEAR(Birthdate) BETWEEN 20 AND 22

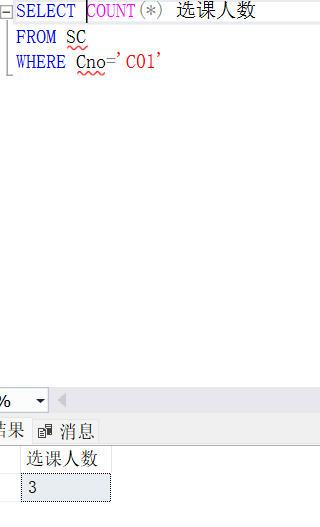


（8）查询选修’C01’课程的学生人数。

SELECT COUNT(\*) 选课人数

FROM SC

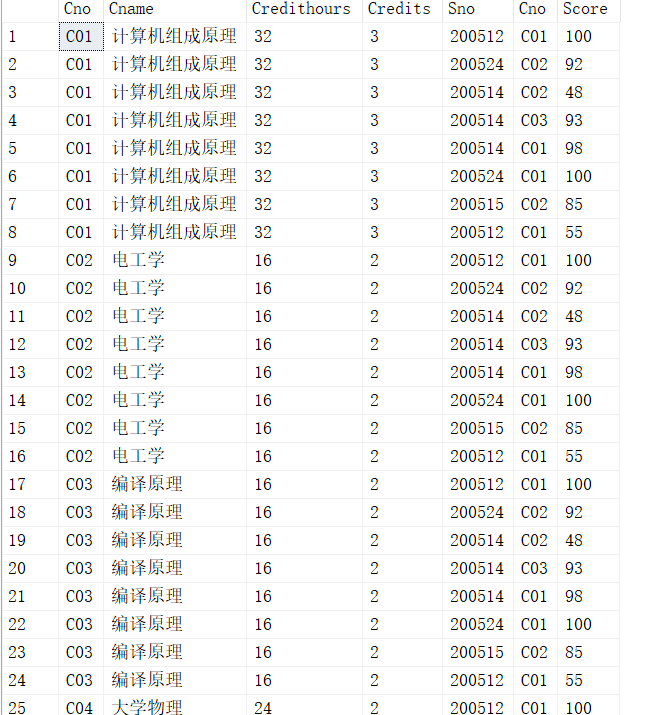
WHERE Cno='C01'



实验5：笛卡尔成绩，自身连接

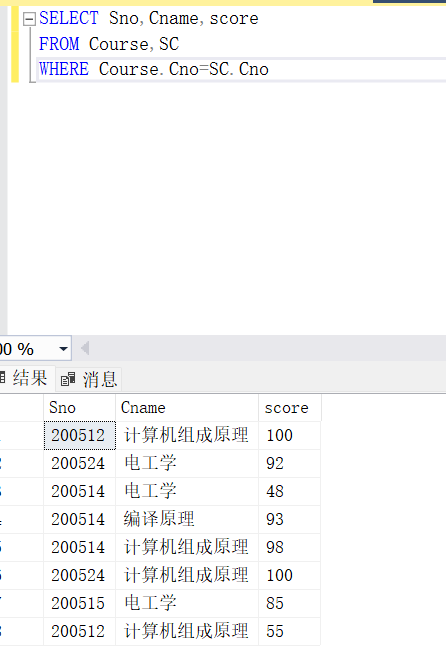
验证性实验：

1）多表连接查询：

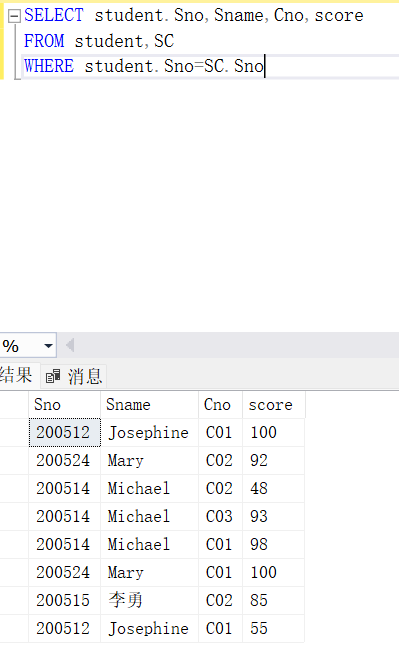


1. 表的等值连接查询：

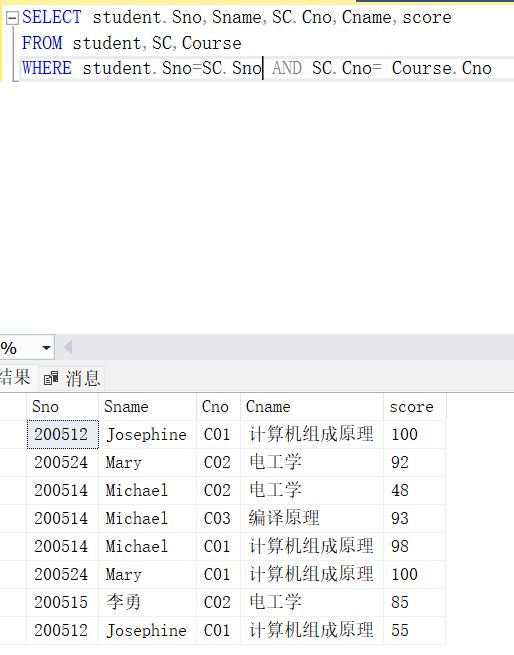
①查询各学生的选课信息（包括学号、课程名、成绩）



②查询学生的选课记录，显示学生的学号、姓名、课程号、成绩）

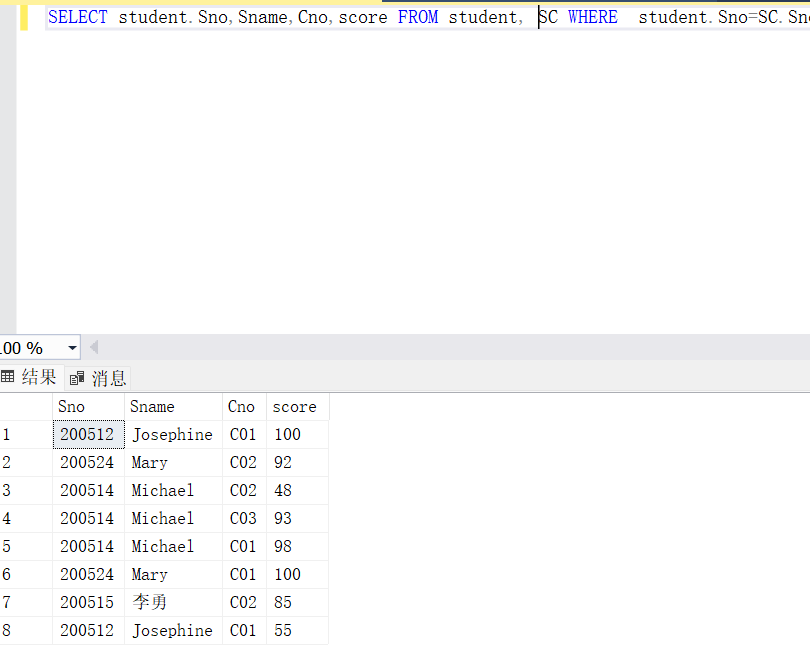
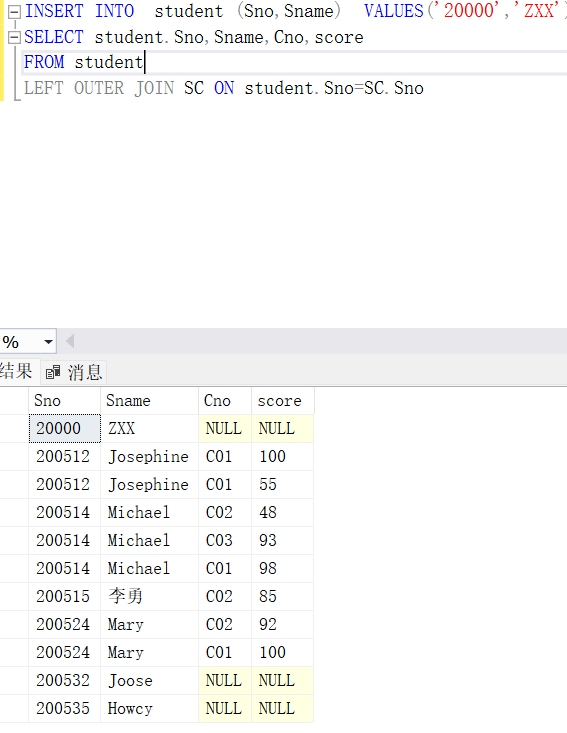


③查询学生的选课记录，显示学生的学号、姓名、课程名、成绩



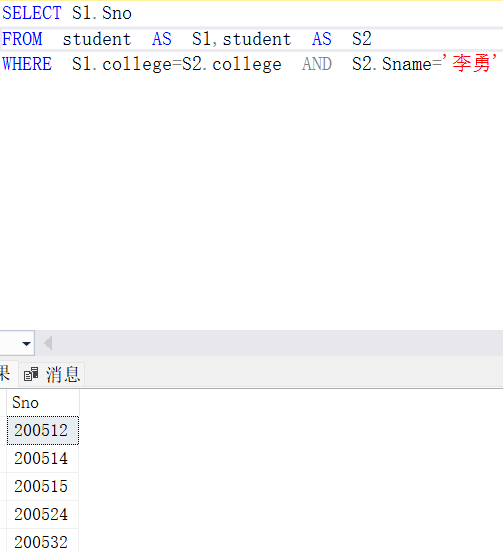
④左外连接查询：当希望左表（第一张表）中所有记录全部显示出来时，需用左外连接操作。

第一个操作：

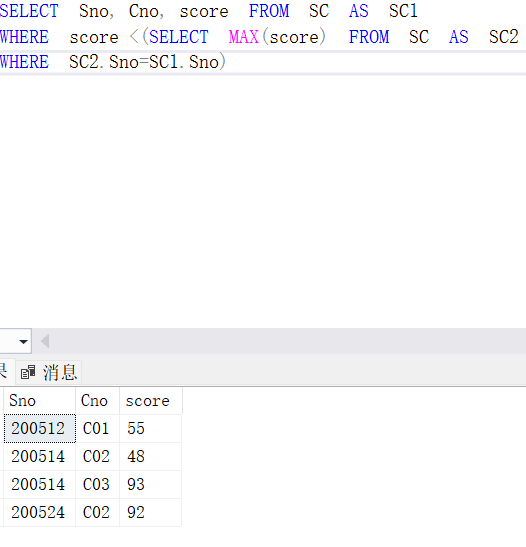


结果不同：因为不是左外连接，所以操作二会把空值省略。

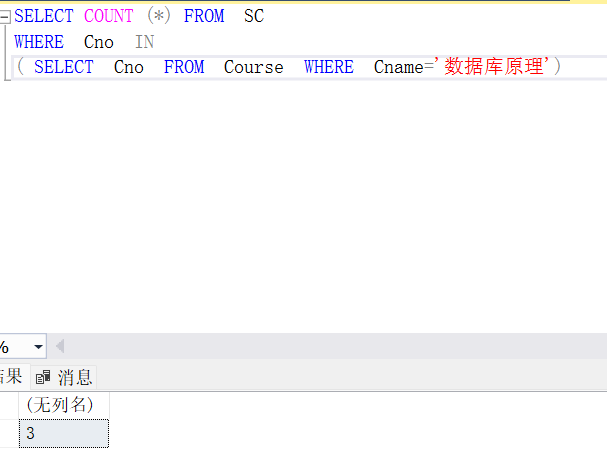
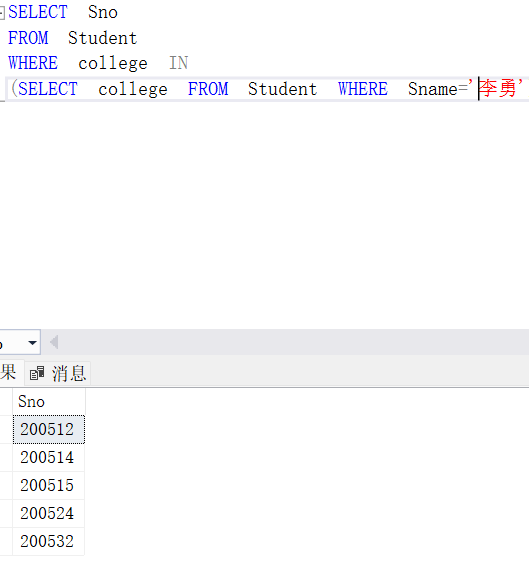
1. 多表连接查询：
2. 查询与’李勇’同系的学生学号



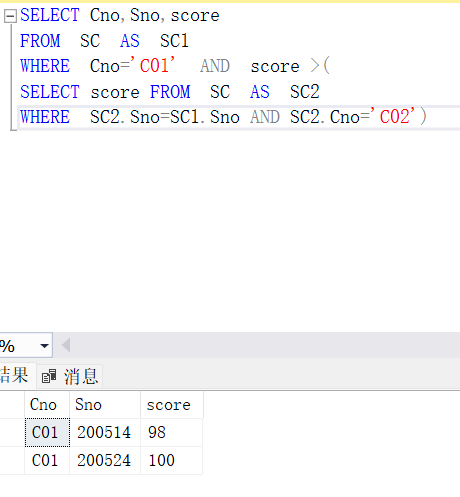
1. 显示每个学生的非最高分成绩（学生自己的选课成绩中，不是最高分的选课记录显示出来）



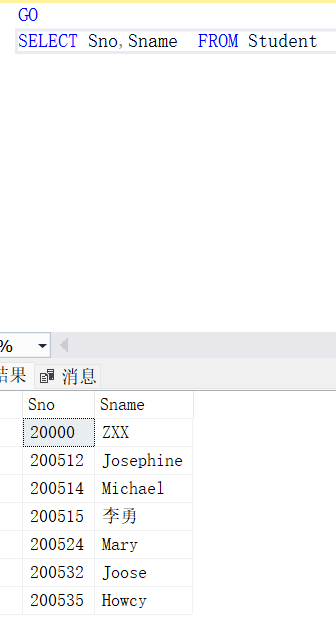
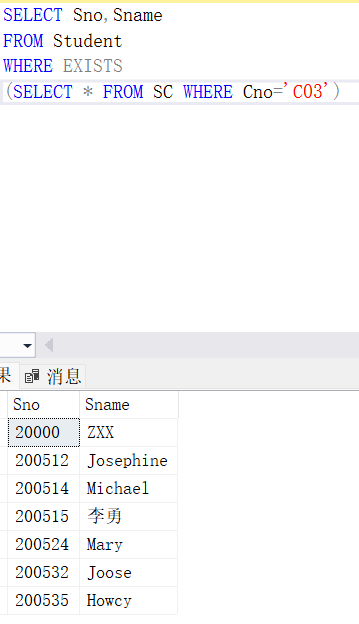
1. IN嵌套子查询：
2. 不相关的IN 子查询：子查询可以单独执行，与被嵌套的查询无关。



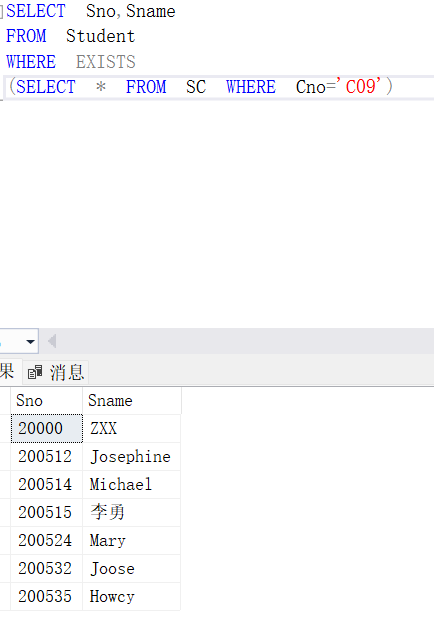
1. 相关的子查询：子查询中要用到父查询表的信息，子查询不能独立执行。



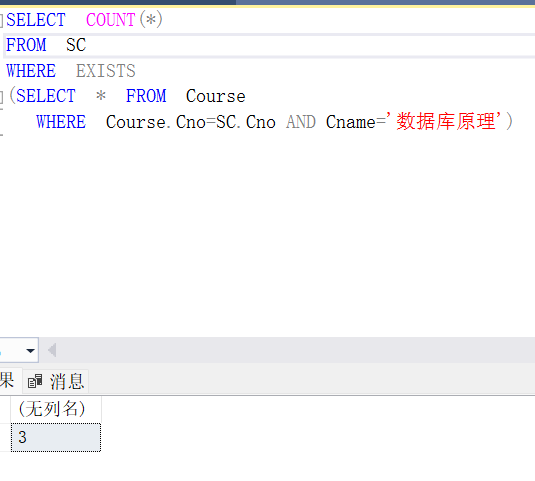
1. EXISTS嵌套子查询
2. 执行以下语句，观察显示的两个查询结果



1. 执行以下语句，观察显示的两个查询结果



1. 查询’数据库原理’课程的选课人数



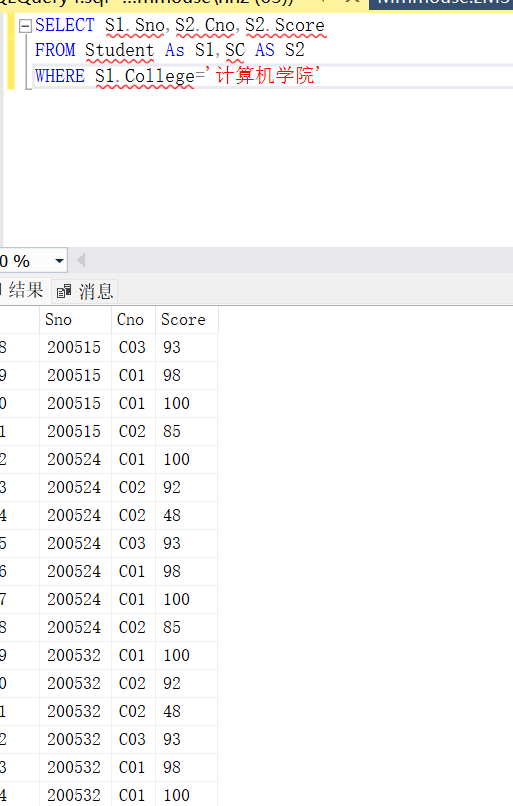
实验5设计性作业：

1. 查询“计算机学院“学生所教课程的成绩表。

SELECT S1 Sno, S2.Cno, S2. Score

FROM Student As S1,SC AS S2

WHERE S1.College ’计算机学院’



（2）查询成绩比该课程平均成绩低的同学的成绩表。

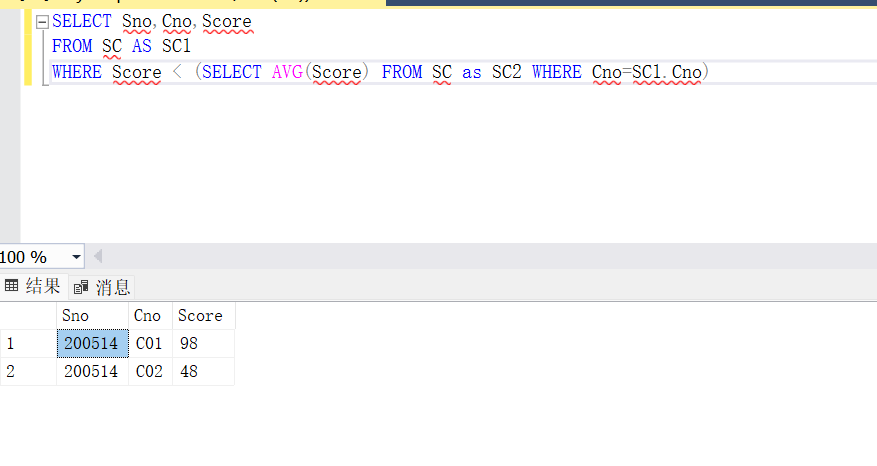
SELECT Sno,Cno,Score

FROM SC AS SC1

WHERE Score < (SELECT AVG(Score)

FROM SC AS SC2

WHERE Cno=SC1. Cno)



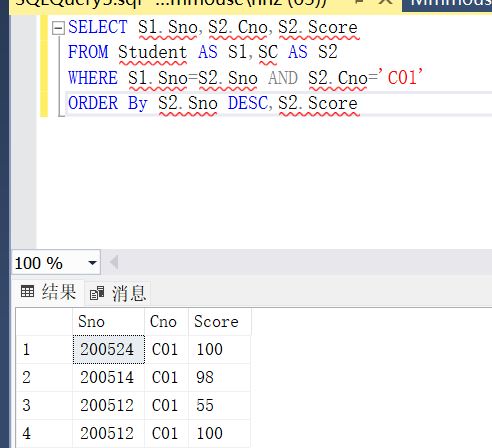
（3）查询选修C01 课程的学生学号、课程名、成绩，并要求对查询结果按学号的降序排列，如果学号相同则按成绩的升序排列。

SELECT S1.Sno, S2.Cno, S2. Score

FROM Student As S1,SC AS S2

WHERE S1.Sno = S2.Sno AND S2.Cno = ’C01’

ORDER By S2.Sno DESC, S2. Score



（4）查询选修两门及两门以上课程的学生学号及姓名。

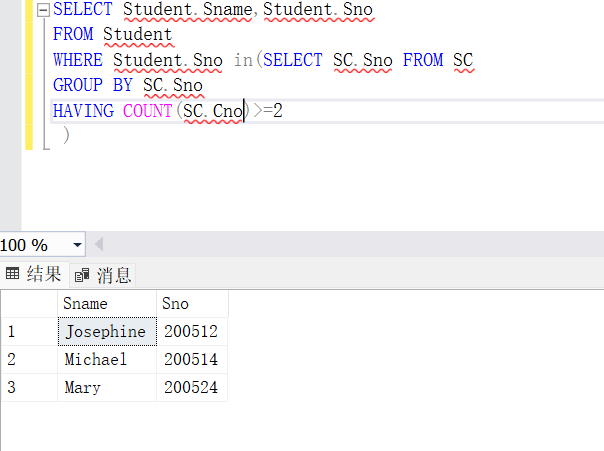
SELECT Student.sname Student.Sno

FROM Student

WHERE Student.Sno in (SELECT SC.Sno FROM SC

GROUP BY SC.Sno

HAVING COUNT (SC. Cno) >=2）



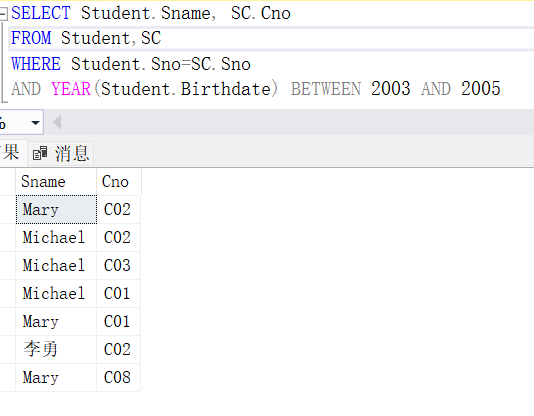
（5）查询出生年份在2003年-2005年之间的男生的选修的课程号。

SELECT Student.Sname, SC.Cno

FROM Student,SC

WHERE Student.Sno=SC.Sno

AND YEAR(Student.Birthdate) BETWEEN 2003 AND 2005

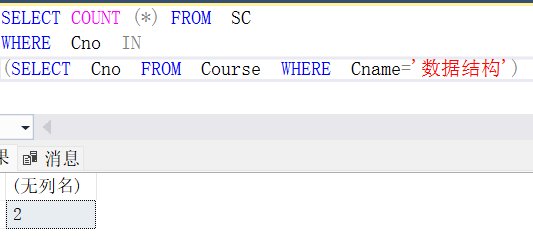


（6）查询选修’数据结构’课程的学生人数。

SELECT COUNT (\*) FROM SC

WHERE Cno IN

(SELECT Cno FROM Course WHERE Cname='数据结构')



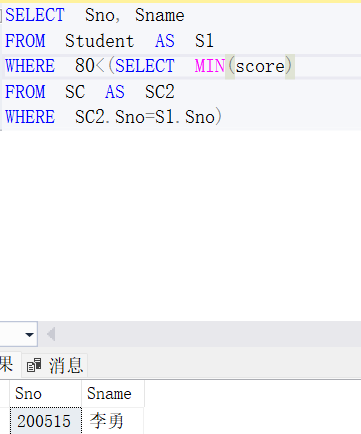
（7）查询每门选课成绩在80分以上的学生学号、姓名。

SELECT Sno, Sname

FROM Student AS S1

WHERE 80<(SELECT MIN(score)

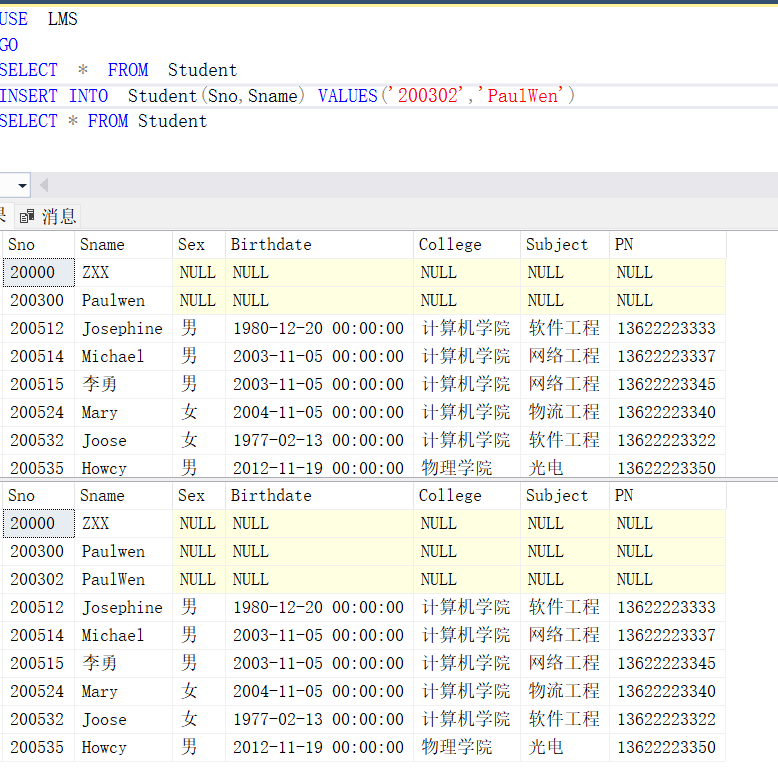
FROM SC AS SC2 WHERE SC2.Sno=S1.Sno)



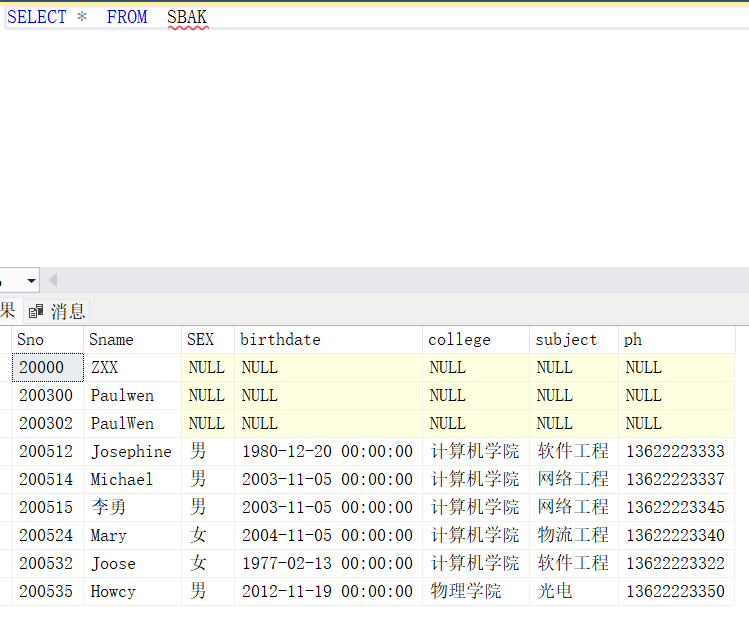
实验6：

验证性实验：

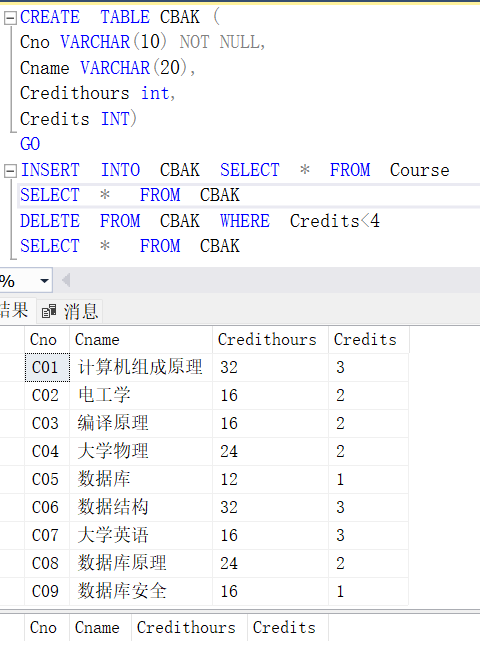
1. 数据的插入
2. 单记录的插入



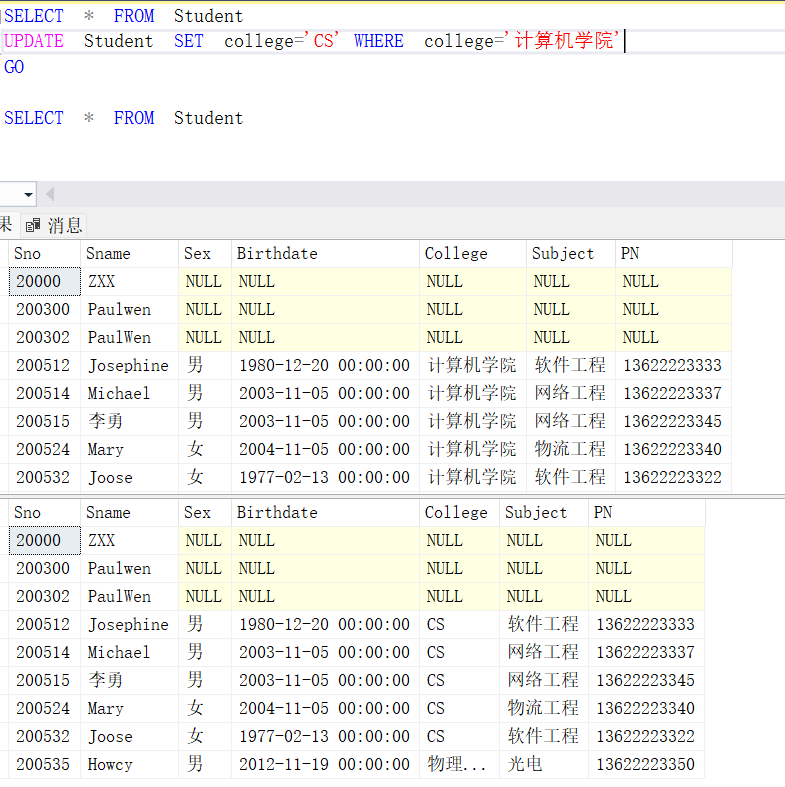
1. 多记录的插入



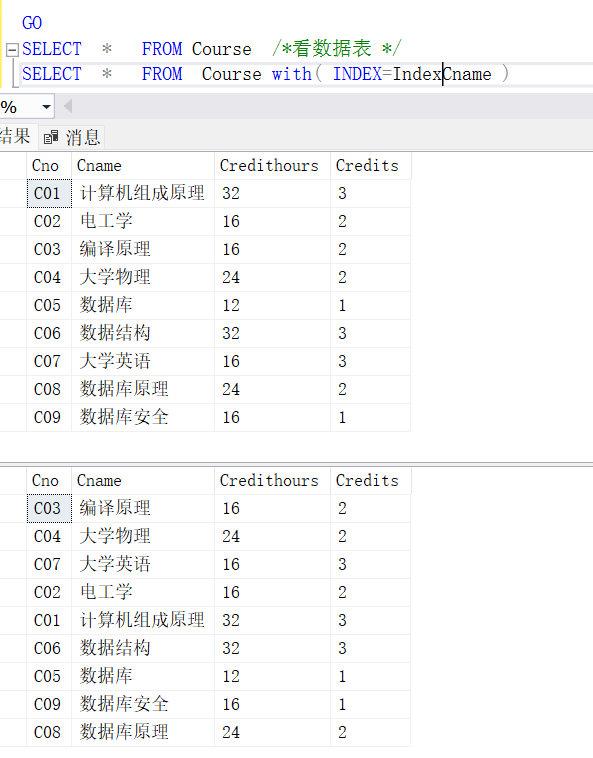
1. 数据的删除



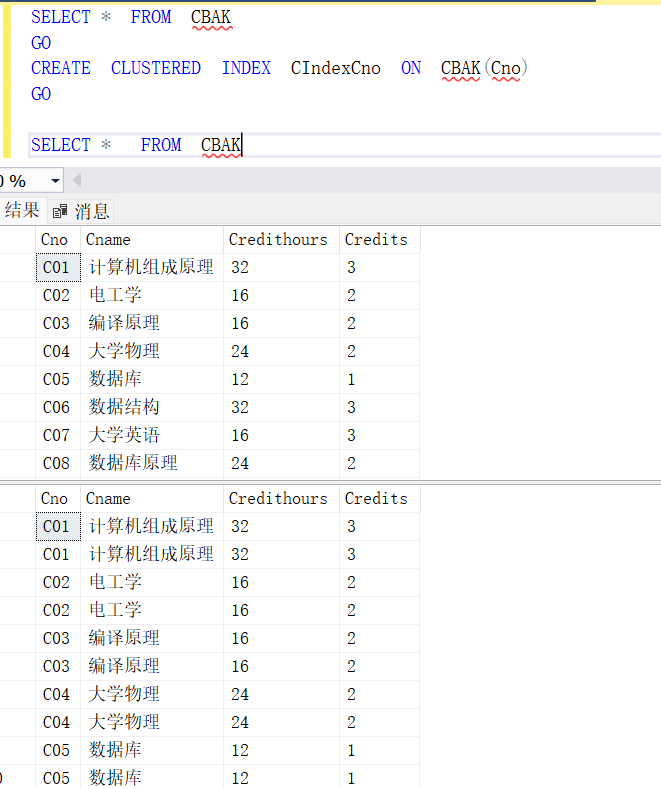
1. 数据的修改



1. 索引的建立
2. 建立非聚族索引



1. 建立聚族索引



1. 建立唯一索引

CREATE UNIQUE INDEX UUIndexSno ON SC （Sno）

执行后有什么结果？为什么会出现这个结果？

因为发现对象名称 'dbo.SC' 和索引名称 'UUIndexSno' 有重复的键，所以 CREATE UNIQUE INDEX 语句终止。重复的键值为 (200512)。

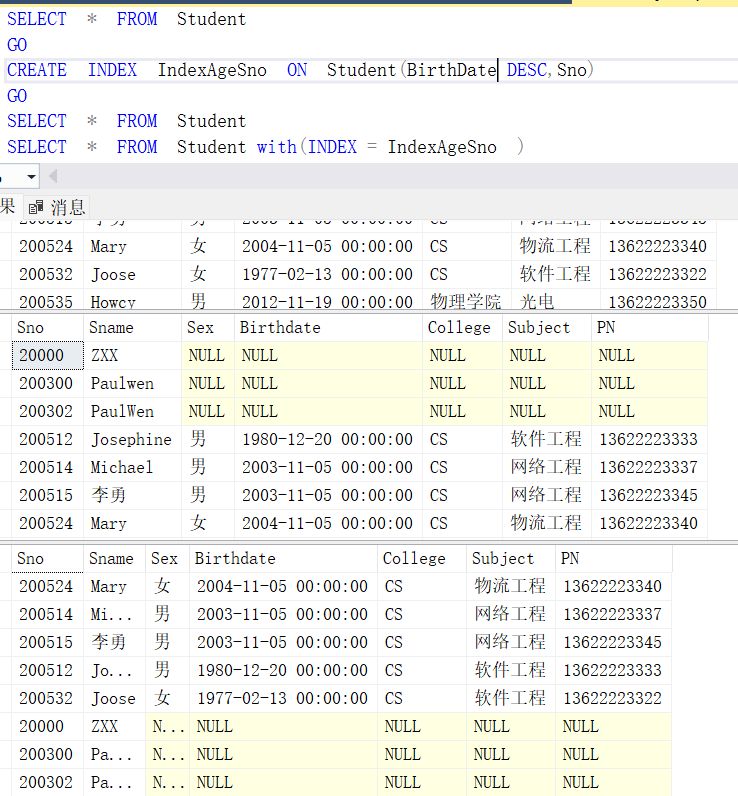
CREATE UNIQUE INDEX UISno ON SC （Sno， Cno DESC）

执行后有什么结果？

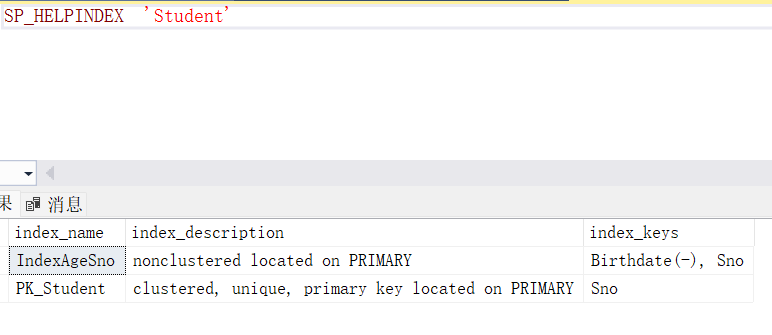
命令已成功完成。

这是因为SC表中只有Sno和Cno一起才能唯一标识一个个体，才能建立唯一索引。

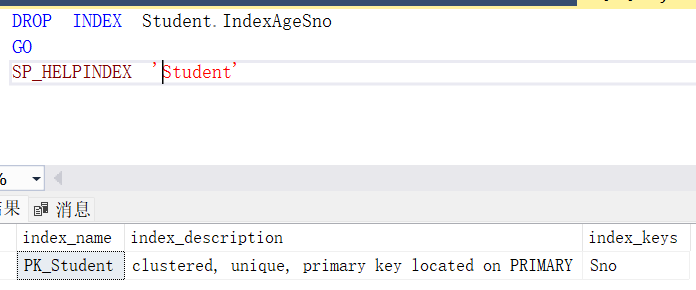
1. 建立复合索引



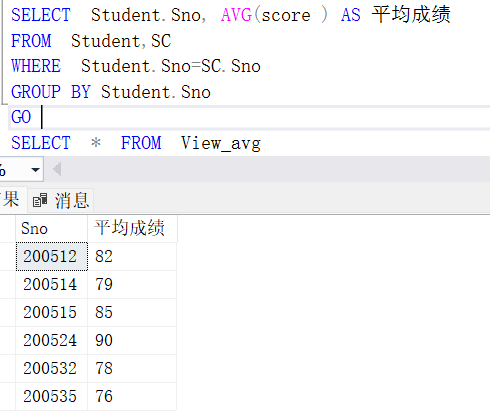
1. 查看表中的索引



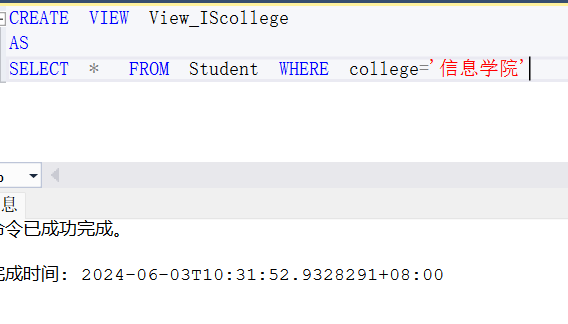
1. 删除表中的索引



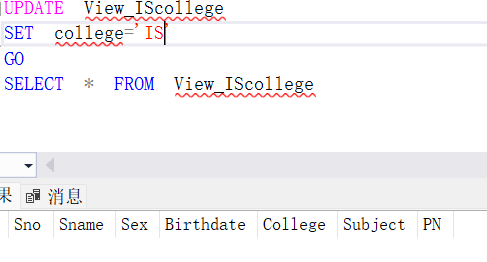
1. 视图及其操作
2. 视图的建立



创建信息系学生的视图

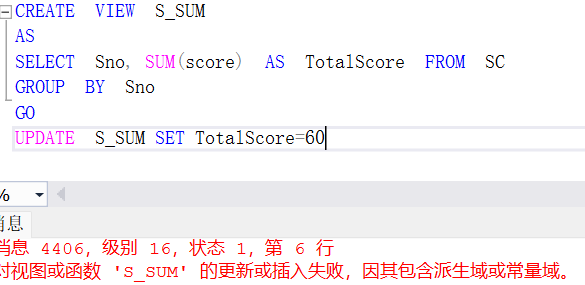


1. 在视图上修改数据



视图中还有数据吗？为什么会这样？

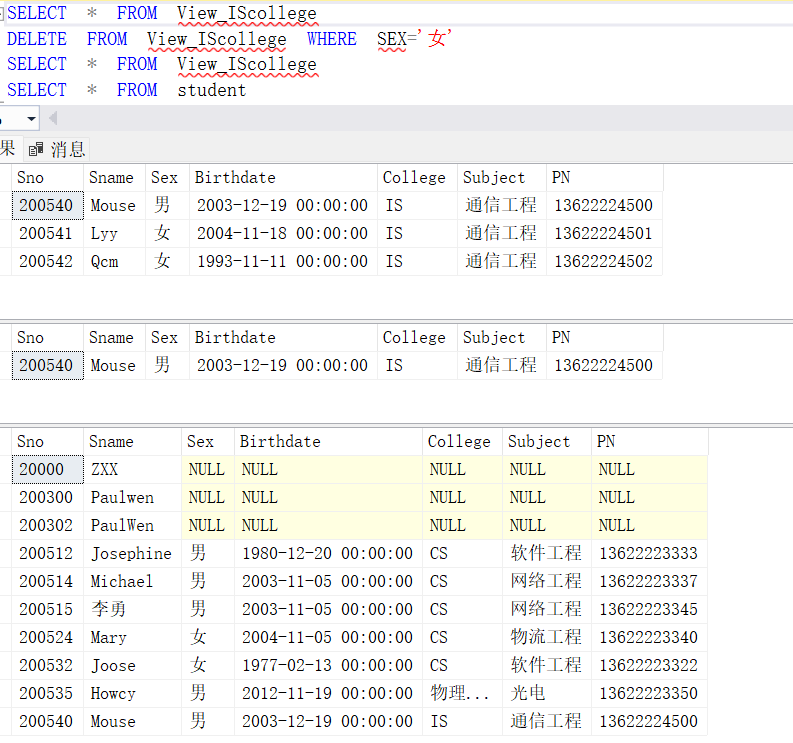
没有了，因为college变成了IS以后表中没有“IS”学院的学生。



执行语句，出现什么结果？分析为什么？

更新失败，这是因为视图或函数中的派生字段(total\_score)是根据其他表格的数据计算得出的，而非实际存储在数据库中的字段。SQL不允许直接更新视图或函数中的派生字段，因为这些字段的值不是直接存储在表格中的。

1. 在视图上删除数据



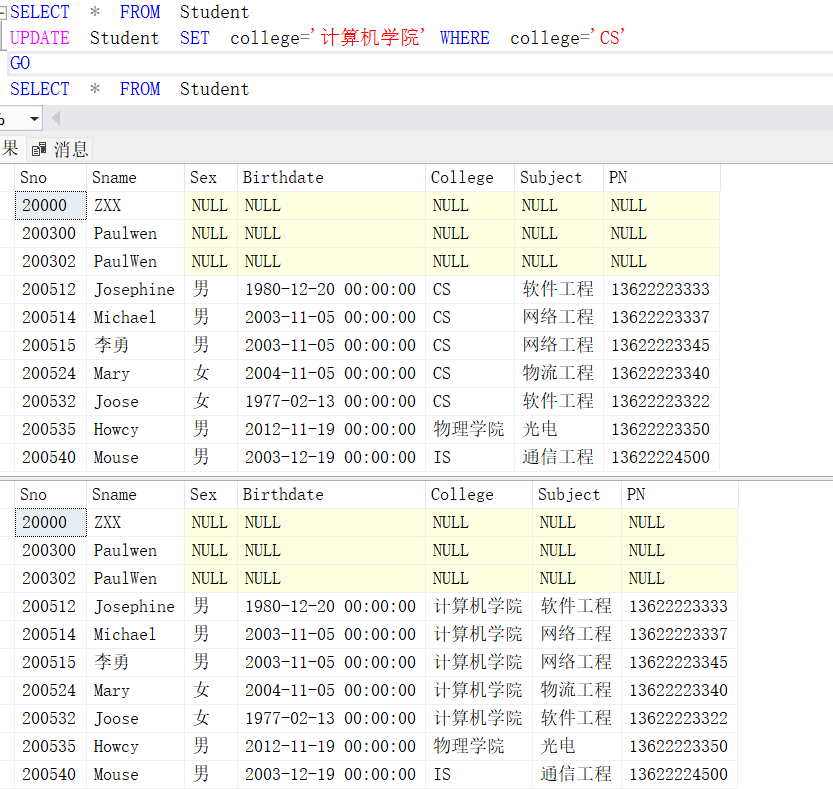
设计性实验：

1）将student表学院为“计算机学院”学生对应的改为’CS’。

SELECT \* FROM Student

UPDATE Student SET college='CS' WHERE college='计算机学院'

SELECT \* FROM Student（图中表应相反）



2）创建一个与SC表相同新数据表SCNEW，查询SC表中成绩为空的记录并将其复制到SCNEW表中。

CREATE TABLE SCNEW(

Sno VARCHAR (10) NOT NULL,

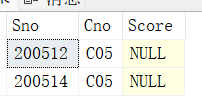
Cno VARCHAR (3) NOT NULL,

Score Tinyint )

SELECT \* FROM SCNEW

INSERT INTO SCNEW SELECT \* FROM SC WHERE Score IS NULL

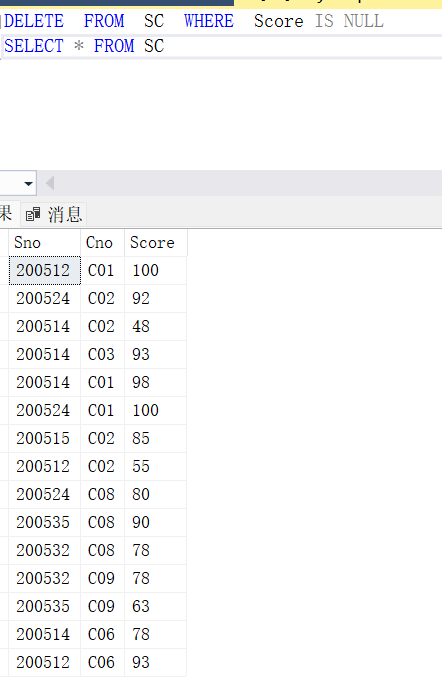
SELECT \* FROM SCNEW



3）删除SC表中成绩为空的选课记录。

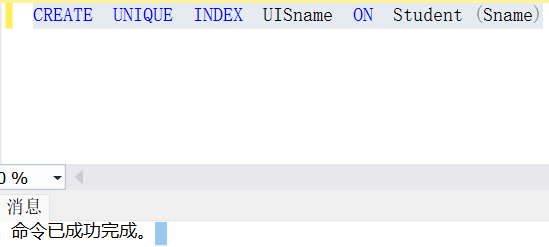
DELETE FROM SC WHERE Score IS NULL

SELECT \* FROM SC



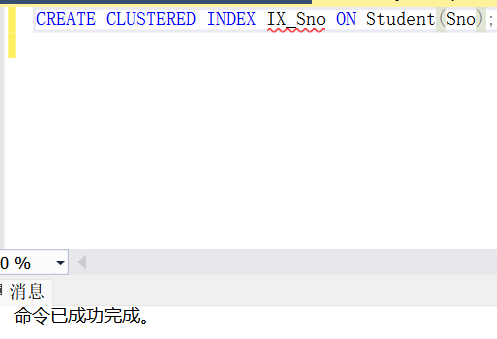
4）根据student表中姓名字段建立唯一性索引。

CREATE UNIQUE INDEX UISname ON Student (Sname)



5）建立student表上Sno的聚族索引。

CREATE CLUSTERED INDEX IX\_Sno ON Student(Sno);



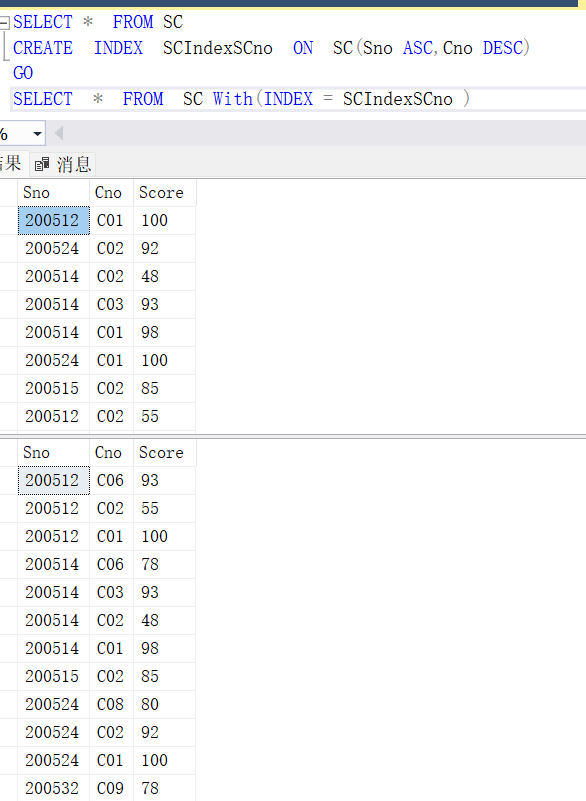
6）建立SC表上根据Sno升序，Cno降序的复合索引。

SELECT \* FROM SC

CREATE INDEX SCIndexSCno ON SC(Sno ASC,Cno DESC)

GO

SELECT \* FROM SC With(INDEX = SCIndexSCno )



7）建立每门课程的课程号、课程名、选课人数、平均分的视图。

CREATE VIEW View\_Cavg

AS

SELECT Course.Cno, AVG(score) AS 平均成绩 , Count(\*) AS 选课人数

FROM Course,SC

WHERE Course.Cno=SC.Cno

GROUP BY Course.Cno

GO

SELECT \* FROM View\_Cavg



实验7：

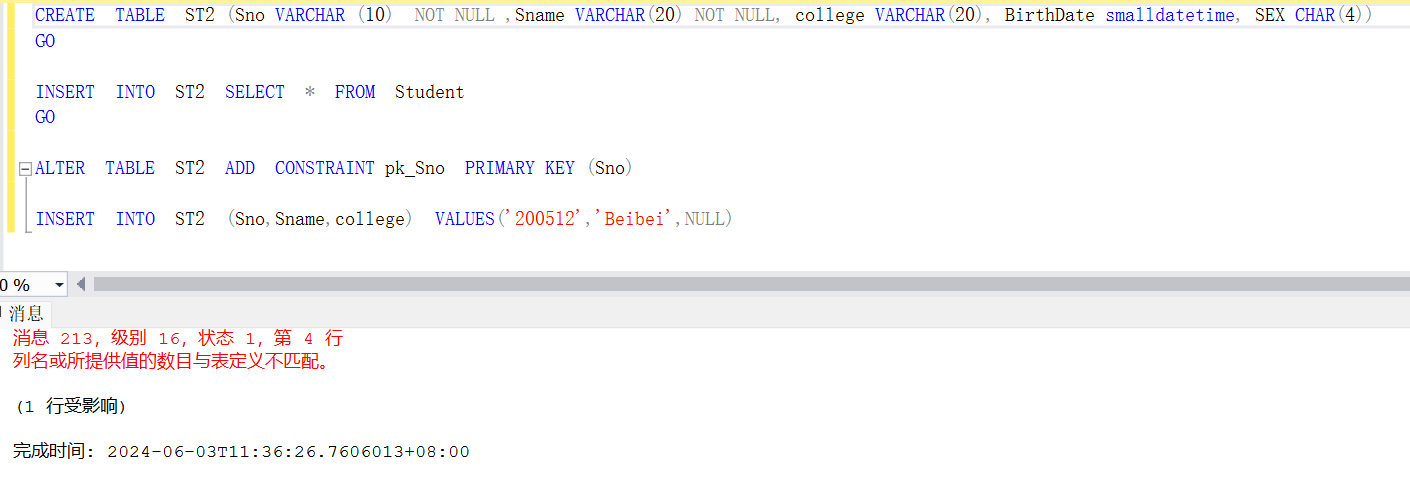
验证性实验：

（1）PRIMARY KEY主键约束的建立

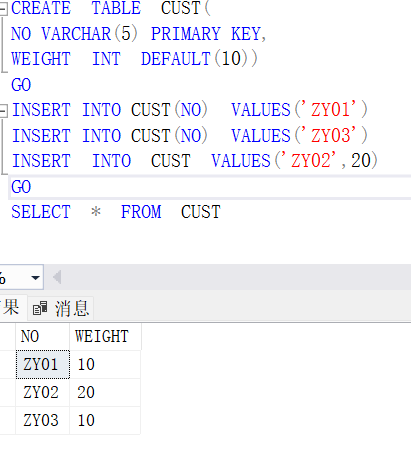
1. 建立表时加主键约束



2)在已有的表上添加约束



（2）DEFAULT约束



（3）CHECK 约束



（4）UNIQUE 唯一性约束的建立



（5）FOREIGN KEY外键约束的建立



设计性实验：

1）在S表中添加完整性约束：Sno设置为主键，SEX 的却省值为’女’,AGE 的有效值为16-25。

ALTER TABLE S ADD PRIMARY KEY(Sno),DEFAULT '女' FOR SEX,CHECK(AGE BETWEEN 16 AND 25)

1. 创建1张与S相同的表S1，在创建的同时将Sno设置为主键，SEX 的却省值为’女’,AGE 的有效值为16-25 ，并将S表中的数据插入到S1中，插入不同的记录来观察设置的完整性。

SELECT \* INTO S1 FROM S WHERE 1=0;

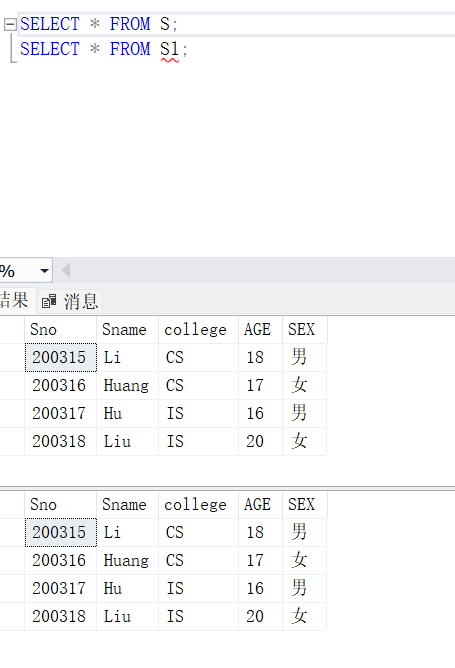
ALTER TABLE S1

ADD PRIMARY KEY(Sno),

CHECK (AGE >= 16 AND AGE <= 25),

DEFAULT '女' FOR SEX;

INSERT INTO S1 SELECT \* FROM S;



3）创建1张与SC相同的表SC1，将（Sno，Cno）设置为主键，Sno和Cno设置为外键，并将SC表中的数据复制到SC1中，插入不同的记录来观察设置的完整性。

SELECT \* INTO SC2 FROM SC WHERE 1=0;

ALTER TABLE SC2

ADD CONSTRAINT PK\_SC1 PRIMARY KEY (Sno, Cno),

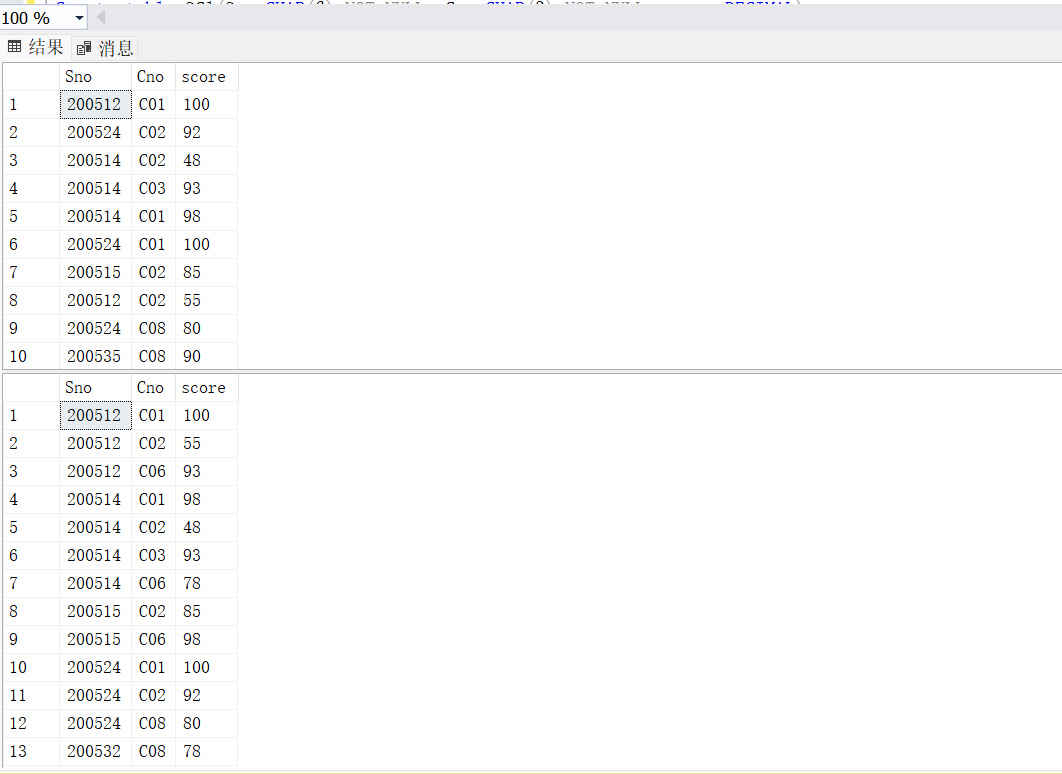
CONSTRAINT FK\_Sno2 FOREIGN KEY (Sno) REFERENCES S(Sno),

CONSTRAINT FK\_Cno2 FOREIGN KEY (Cno) REFERENCES C(Cno);

INSERT INTO SC2 SELECT \* FROM SC;

SELECT \* FROM SC;

SELECT \* FROM SC2;



4）创建1张与C表相同的表C1，将C中数据插入到C1后，在C1上添加完整性约束：将（Cno）设置为主键，CPNO 引用 Cno，CREDIT值为1-6。

CREATE TABLE C1 (

CNO VARCHAR,

CPNO VARCHAR,

CREDIT INT,

PRIMARY KEY (CNO),

);

INSERT INTO C1 (CNO,CREDIT)

SELECT Cno,credit

FROM C;

ALTER TABLE C1

ADD CONSTRAINT FK\_CPNO\_CNO1 FOREIGN KEY (CPNO) REFERENCES C1(CNO);

ALTER TABLE C1

ADD CONSTRAINT CHK\_CREDIT1 CHECK (CREDIT BETWEEN 1 AND 6);

SELECT \* FROM C;

SELECT \* FROM C1;

