**武汉轻工大学**

**数学与计算机学院**

《数据库系统原理》

期末大作业

**专 业： 人工智能**

**班 级： 2201**

**学 号： 2205150039**

**姓 名： 黄斯杰**

**指导老师： 吴巍**

**2024年 11 月 29日**

## 一、 设计目的

数据库系统原理课程设计是专业的集中实践性环节之一，是学习完《数据库系统原理》课程后进行的一次全面的综合练习。其目的在于加深对数据库基础理论和基本知识的理解，掌握使用数据库设计与使用的基本方法，提高运用数据库解决实际问题的能力。

## 二、设计内容

## 2.1 阅读资料，完成读书笔记

SQL（结构化查询语言）是用于管理数据库的语言，具有简洁、功能强大的特点。它独立于平台，具有良好的开放性、可移植性和扩展性，使用户在数据查询、插入、删除、更新等应用和维护中，一种操作只需要一种操作符。SQL语言采用面向集合的操作方式，不仅查询结果是数据元组的集合，非关系型数据库的查询是面向记录的操作方式，用户必须说明循环结构、指明路径，逐条处理数据记录。SQL查询命令主要包括SELECT、INSERT、UPDATE、DELETE等，这些命令用于从数据库中检索、插入、更新和删除数据。

SQL查询命令，尤其是SELECT语句，是数据库管理系统中最基本的功能之一，用于实现数据的查询操作。SQL查询命令在不同的数据库系统中有广泛的应用。例如，在Visual FoxPro数据库中，SQL查询命令用于设置筛选记录条件；在ASP.NET环境下，SQL查询用于实现动态网站的数据库交互功能；而在Access数据库中，通过设计选择查询、交叉表查询等，充分利用了SQL-SELECT查询语句中的联合查询、传递查询等功能。

SQL查询命令在特定领域中的应用展示了其强大的功能和灵活性。例如，在医疗统计中，SQLServer查询语句被用于快速全方位地观察、监测、分析医院的各项工作；在报表统计中，利用SQL查询技术生成各类统计报表的方法和技巧被总结出来；而在班组考勤系统中，通过构造SQL查询语句一次性选择出所有指定记录，展示了SQL查询命令在实际应用中的高效性。

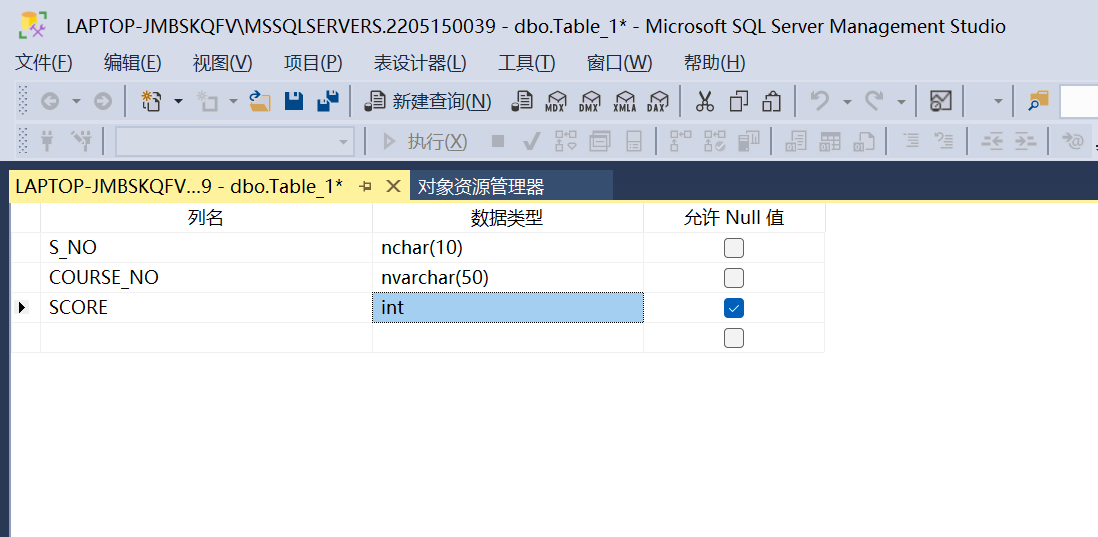
SQL在查询程序方面的应用，联合查询将来自一个或多个表的字段组合成一个结果集。例如，科研信息管理系统中的论文、著作、项目等数据分别存于表中，通过联合查询，实现用户所需数据的统计和分析。传递查询直接向数据库发送命令，实现对数据记录的级联更新。数据定义查询可以根据用户需要创建或更新数据库对象。子查询可以在查询设计网格的字段行输入语句定义新字段，或在准则行定义字段的准则，用于测试子查询的某种结果是否存在。

SQL是一个功能强大且灵活的工具，适用于各种规模和类型的数据库应用。通过学习和掌握SQL，用户可以有效地管理和操作数据，提高工作效率和数据处理能力。

## 2.2 数据库设计

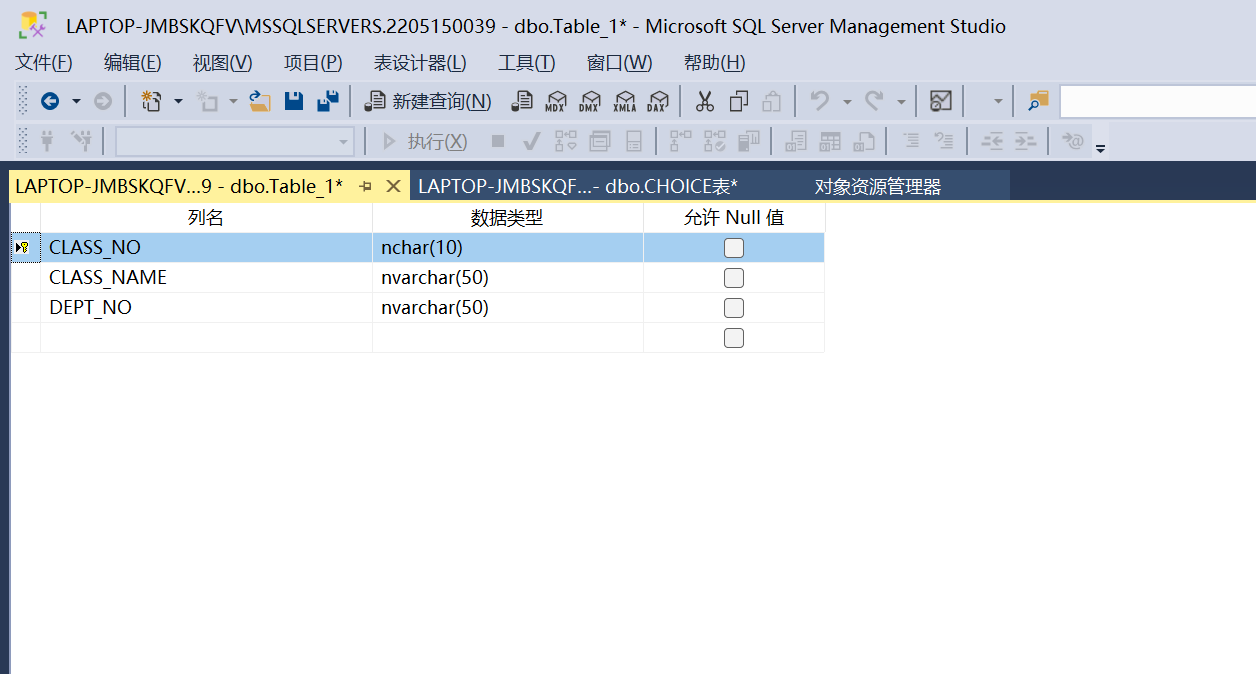
## 1. 创建下列关系表：

**（1）CHOICE表（选课表）**



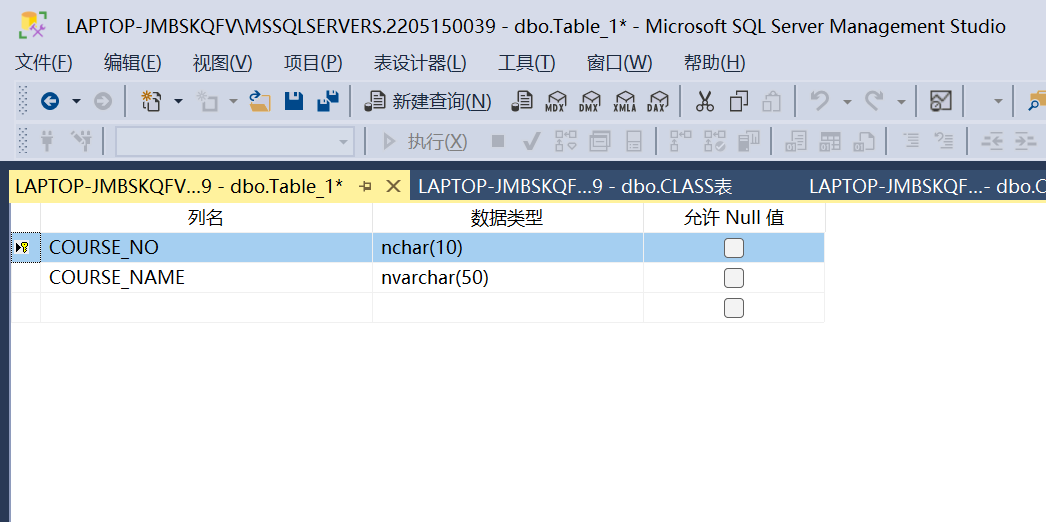
主键：(S\_No, Course\_no)

**（2）CLASS表（班级表）**



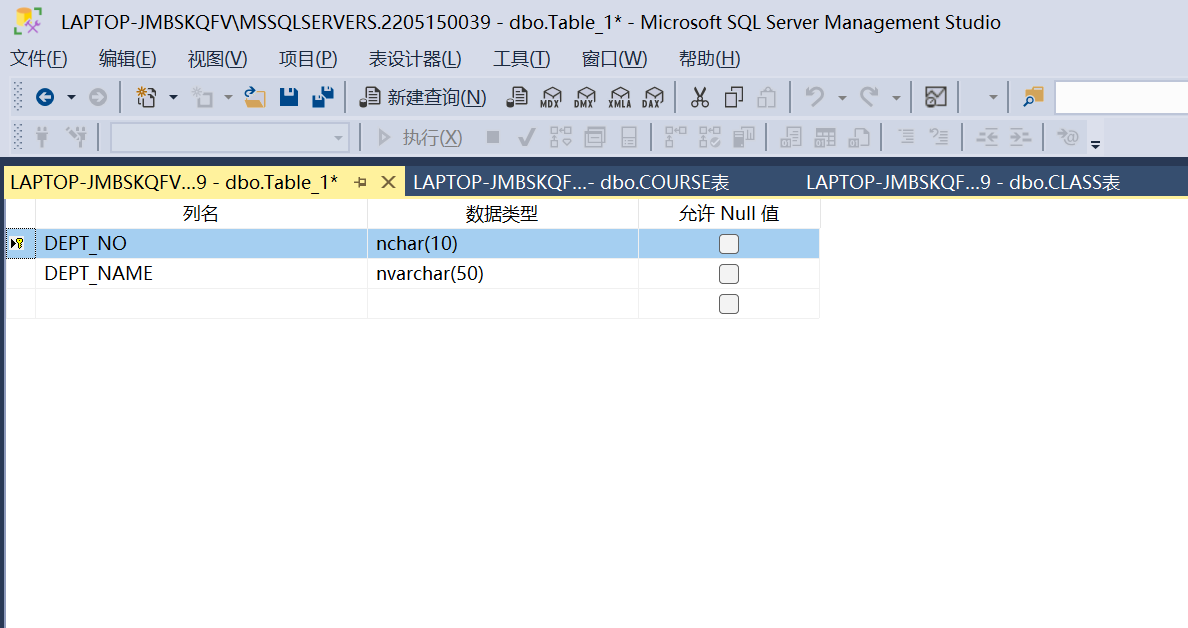
主键：CLASS\_NO

**（3）COURSE表（课程表）**



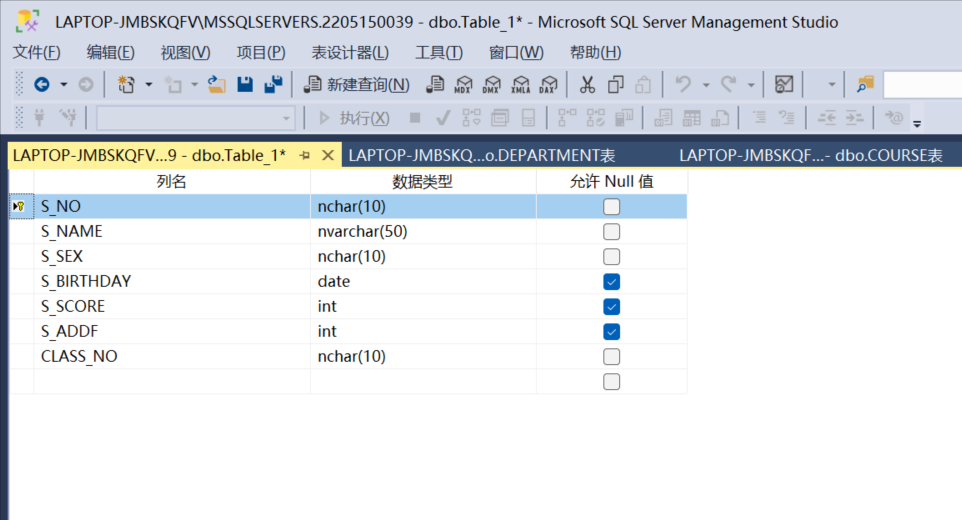
主键：COURSE\_NO

**（4）DEPARTMENT表（系部表）**



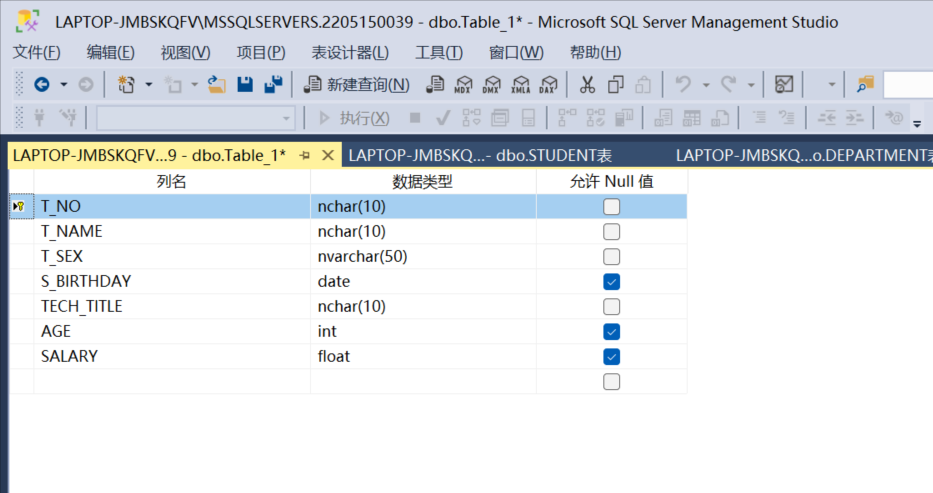
主键：DEPT\_NO

**（5）STUDENT表（学生表）**



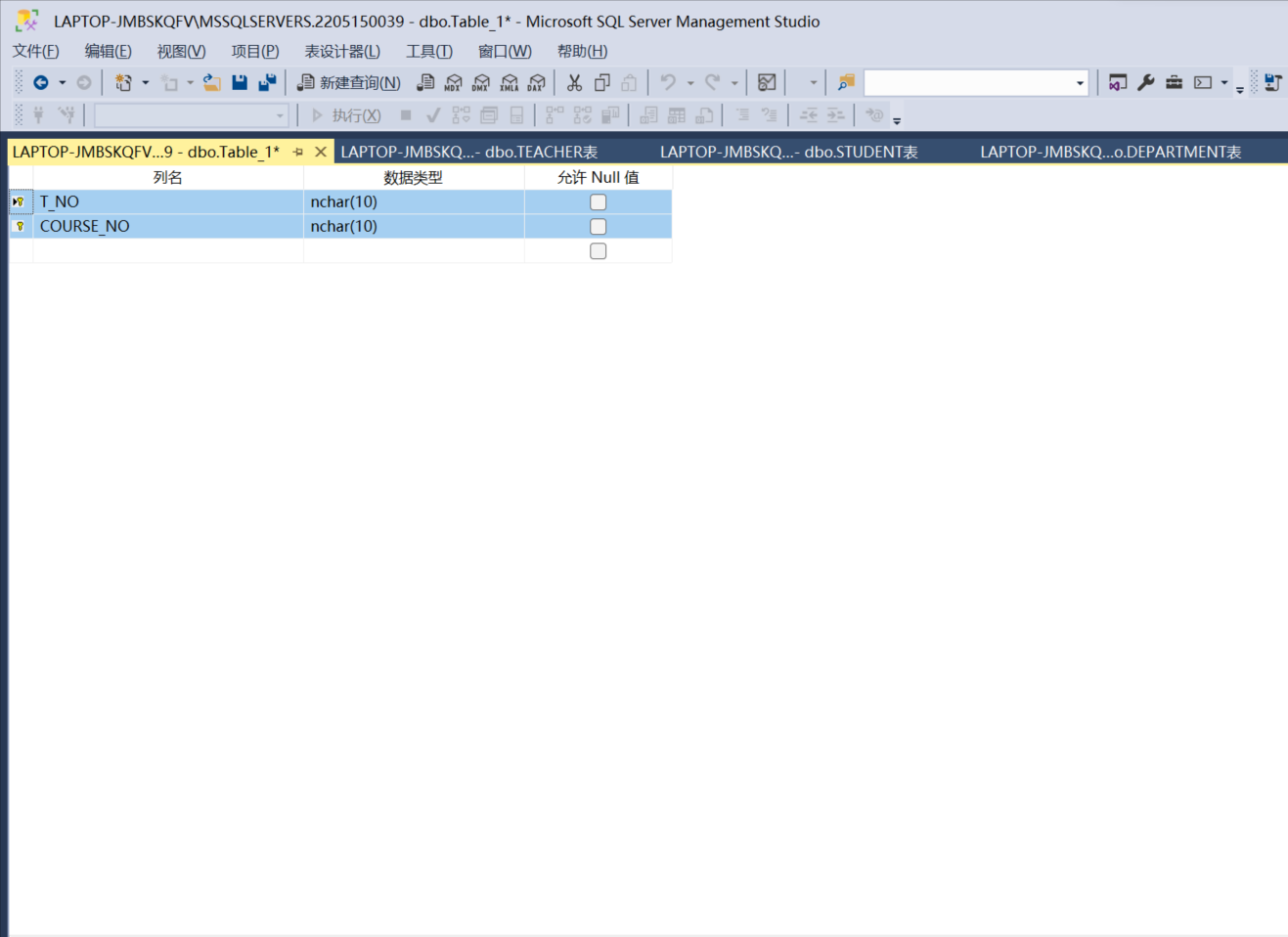
主键：S\_NO

**（6）TEACHER表（教师表）**



**主键：**T\_NO

**（7）TEACHING表（教师授课表）**



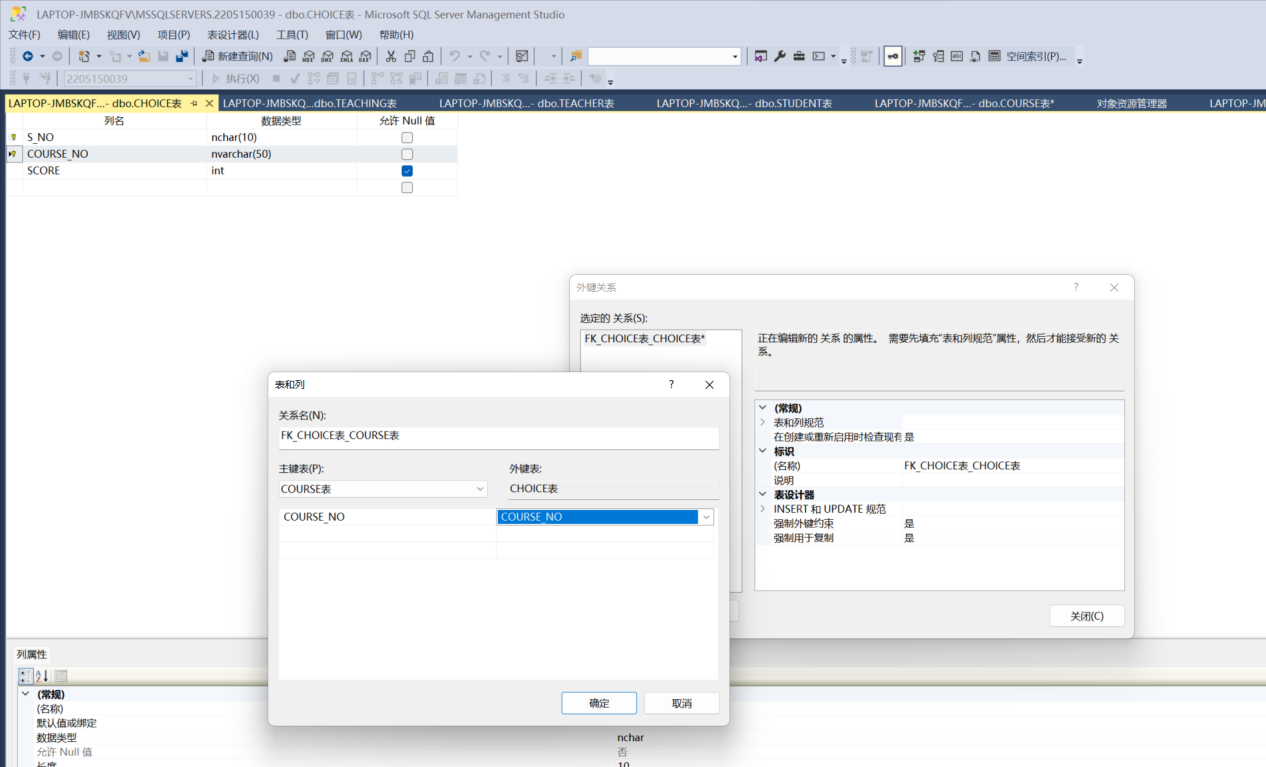
主键：（T\_NO，COURSE\_NO）

其中，约束条件如下：

外键约束：

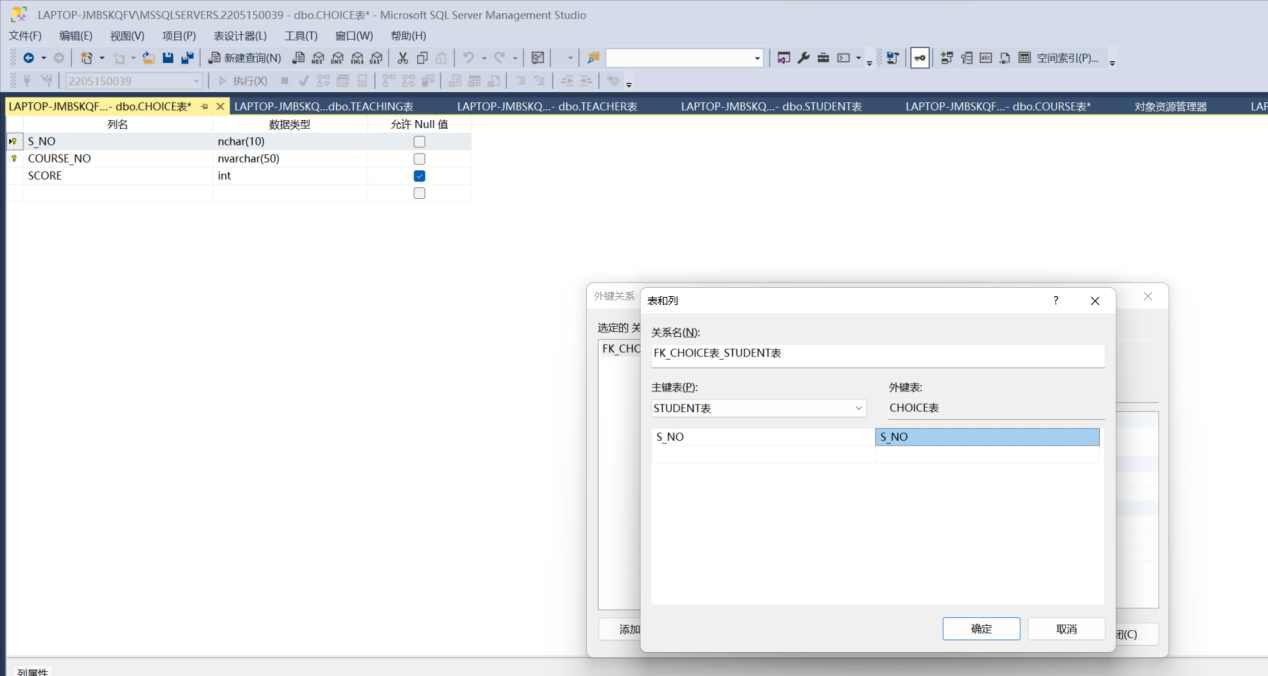
1. CHOICE——COURSE关系

COURSE表中COURSE\_NO为主键，CHOICE表中COURSE\_NO为外键



1. CHOICE——STUDENT关系

STUDENT表中S\_NO为主键，CHOICE表中S\_NO为外键



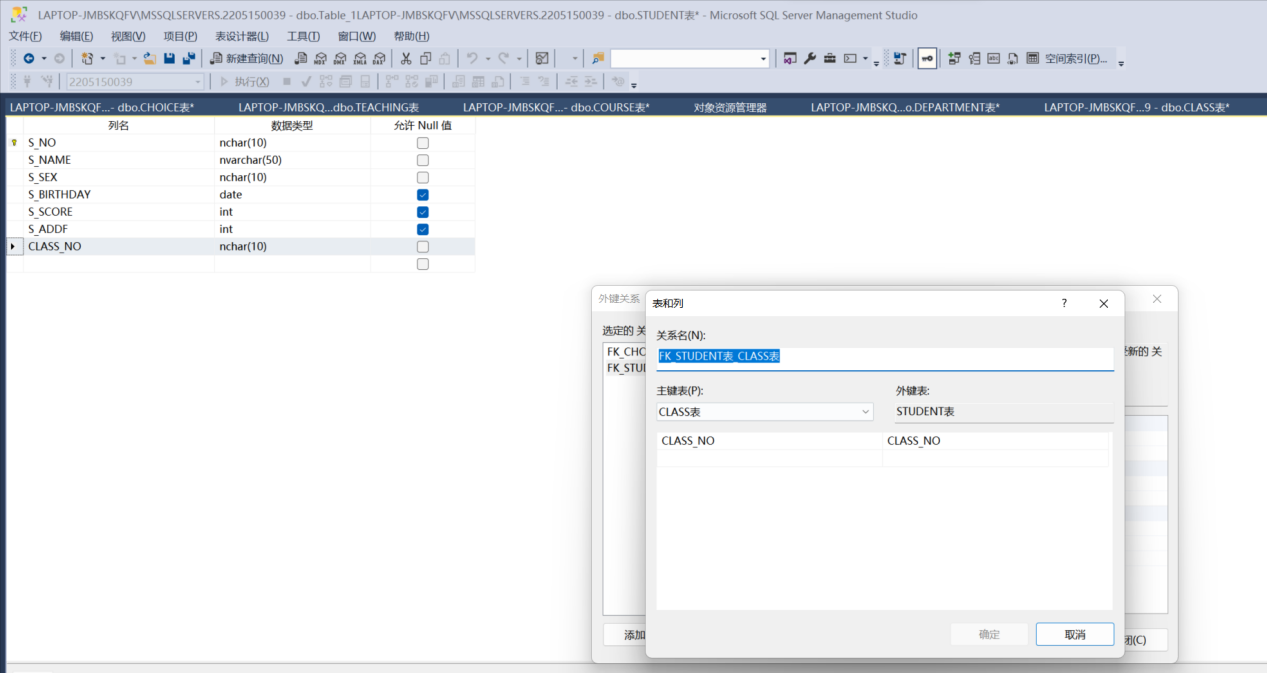
1. CLASS—— DEPARTMENT关系

DEPARTMENT表中DEPT\_NO为主键，CLASS表中DEPT\_NO为外键



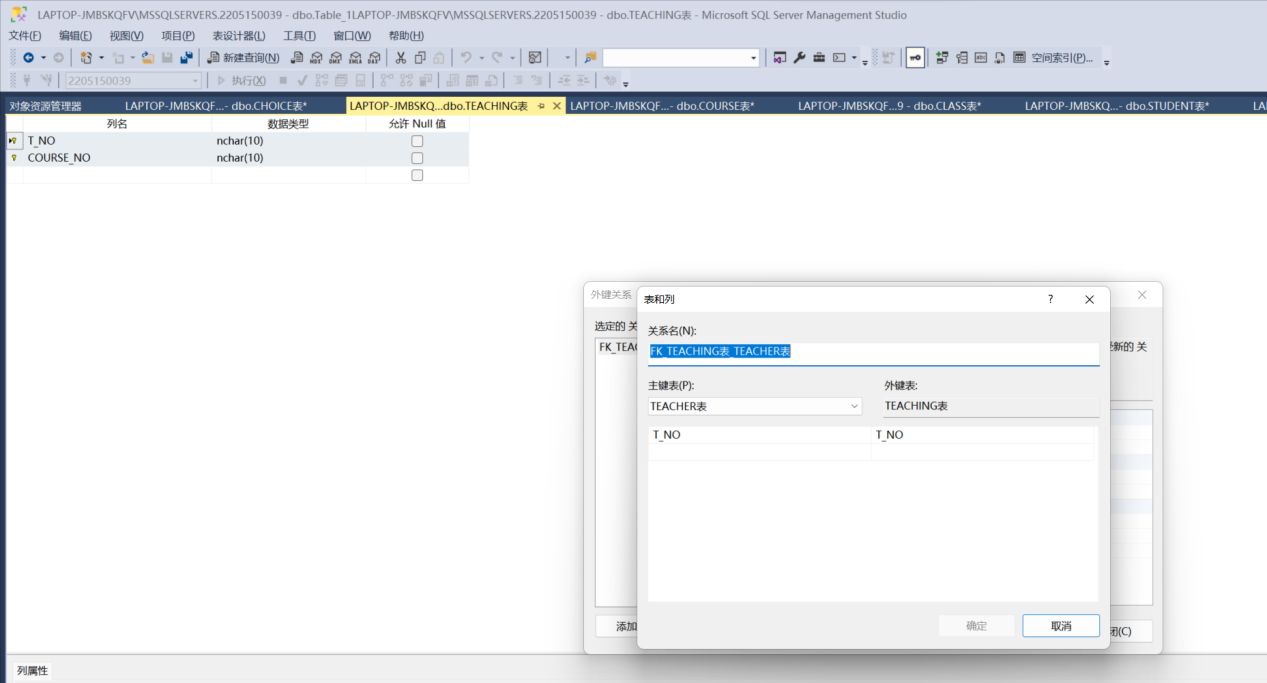
1. STUDENT——CLASS关系

CLASS表中CLASS\_NO为主键，STUDENT表中CLASS\_NO为外键



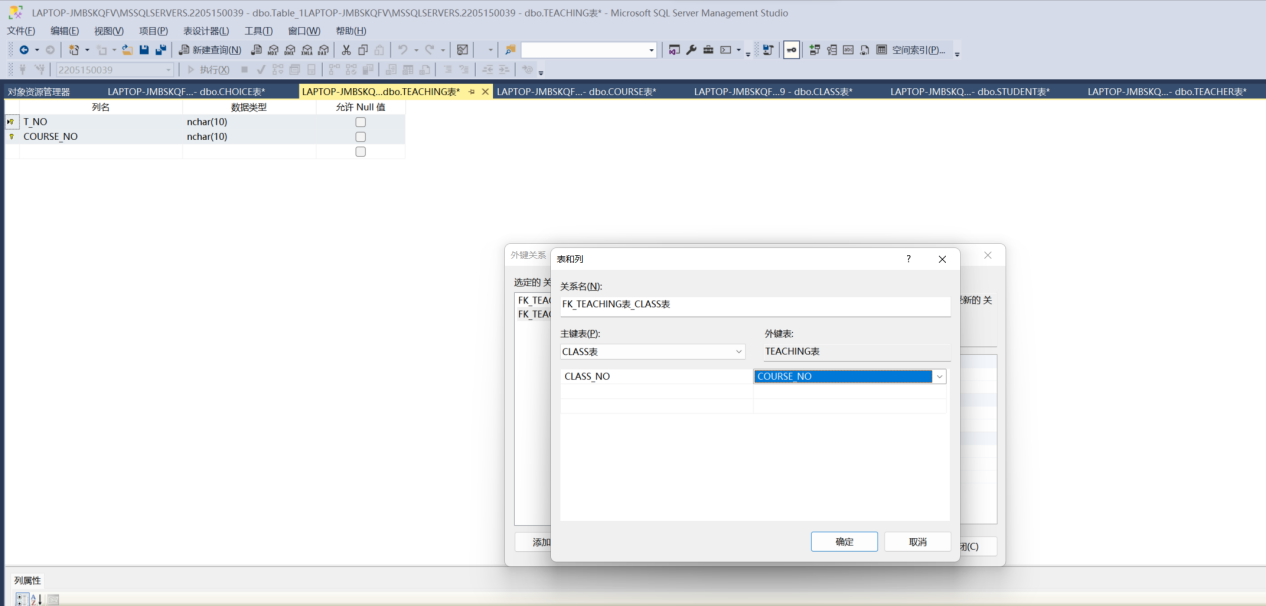
（5）TEARCHING——TEARCHER关系

TEARCHER表中T\_NO为主键，TEARCHING表中T\_NO为外键



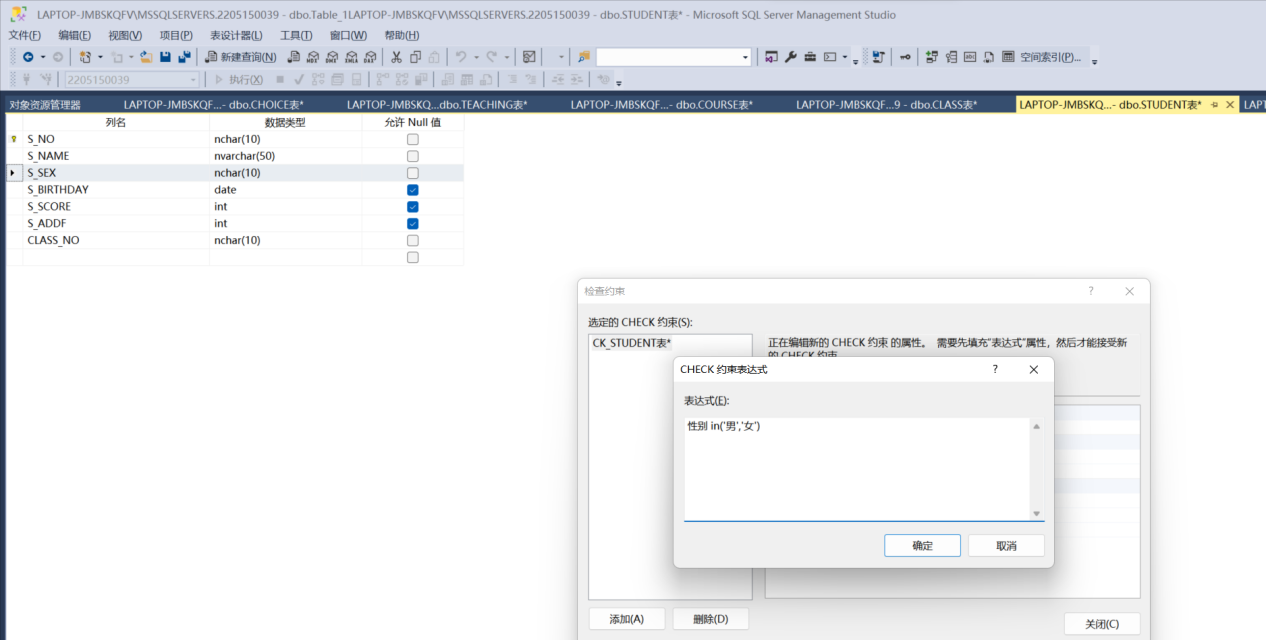
1. TEARCHING——COURSE关系

COURSE表中COURSE\_NO为主键，TEARCHING表中COURSE\_NO为外键



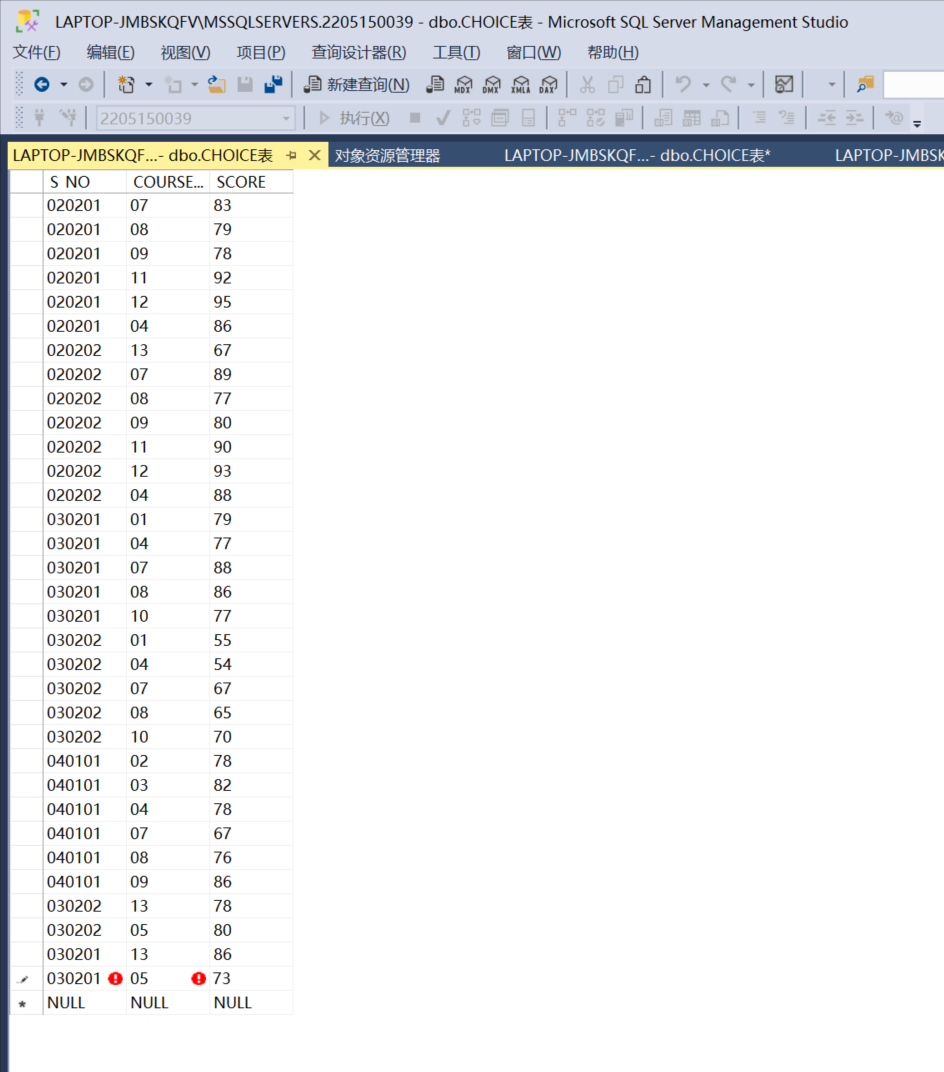
检查性约束：

STUDENT表的S\_SEX字段和TEA CHER表的T\_SEX字段取值均为’男’或’女’。

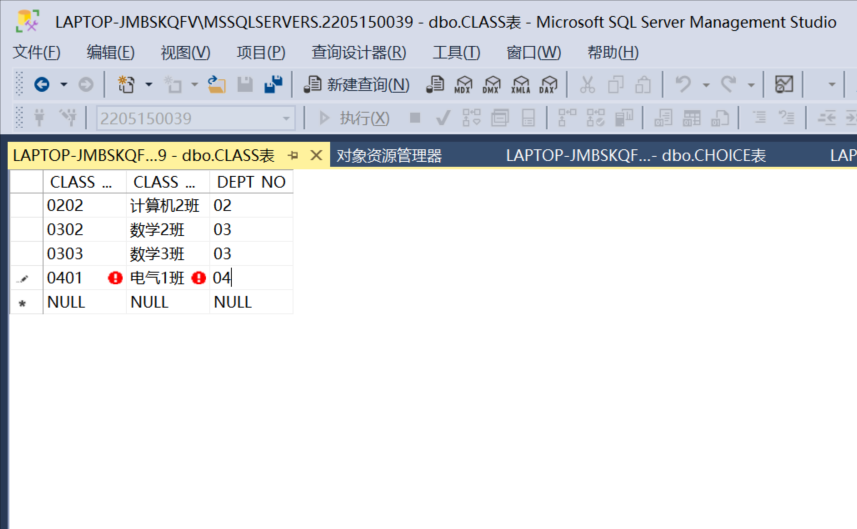


## 插入下列所示的数据

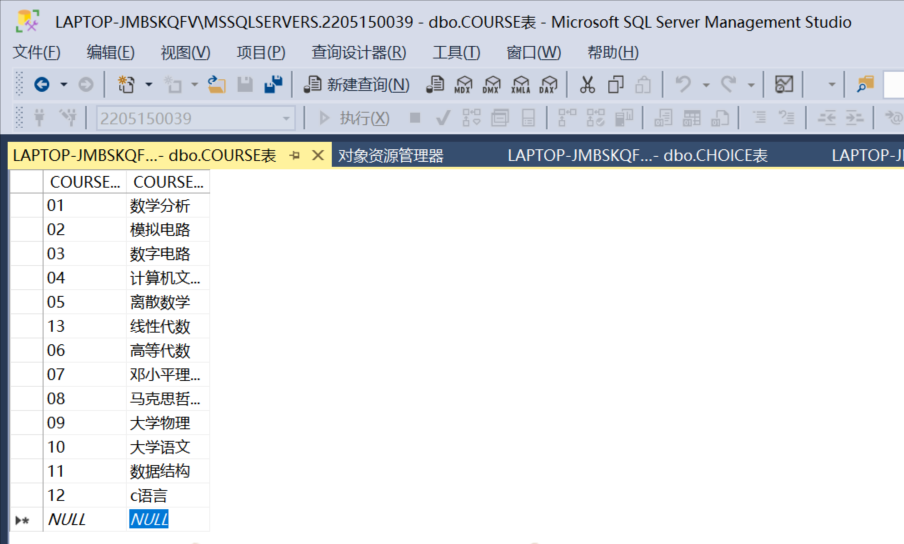
（1）CHOICE表



（2）CLASS表



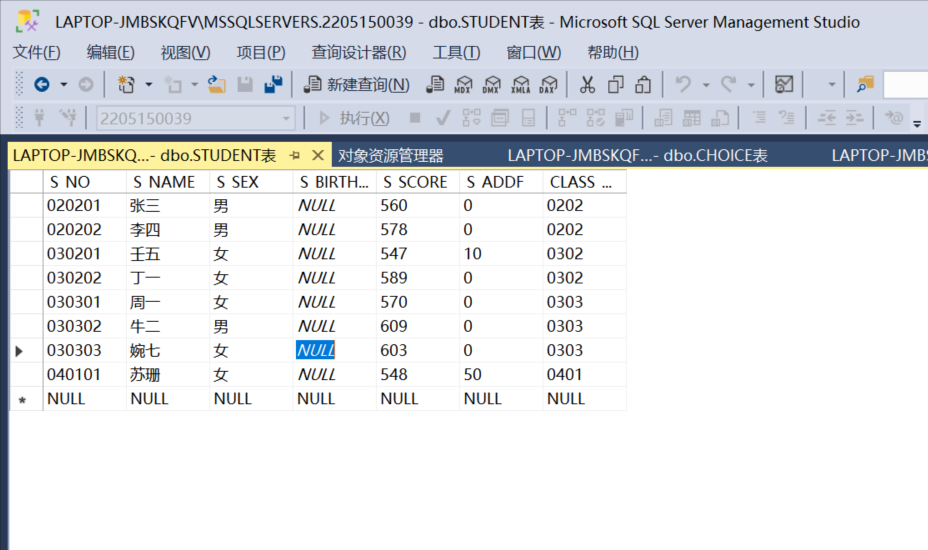
（3）COURSE表



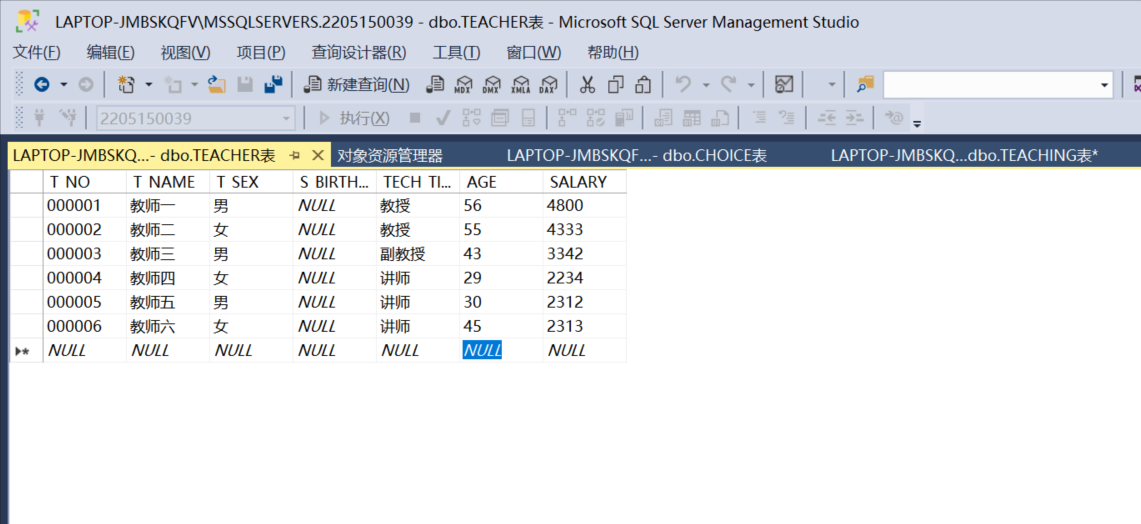
（4）DEPARTMET表



（5）STUDENT表



1. TEACHER表



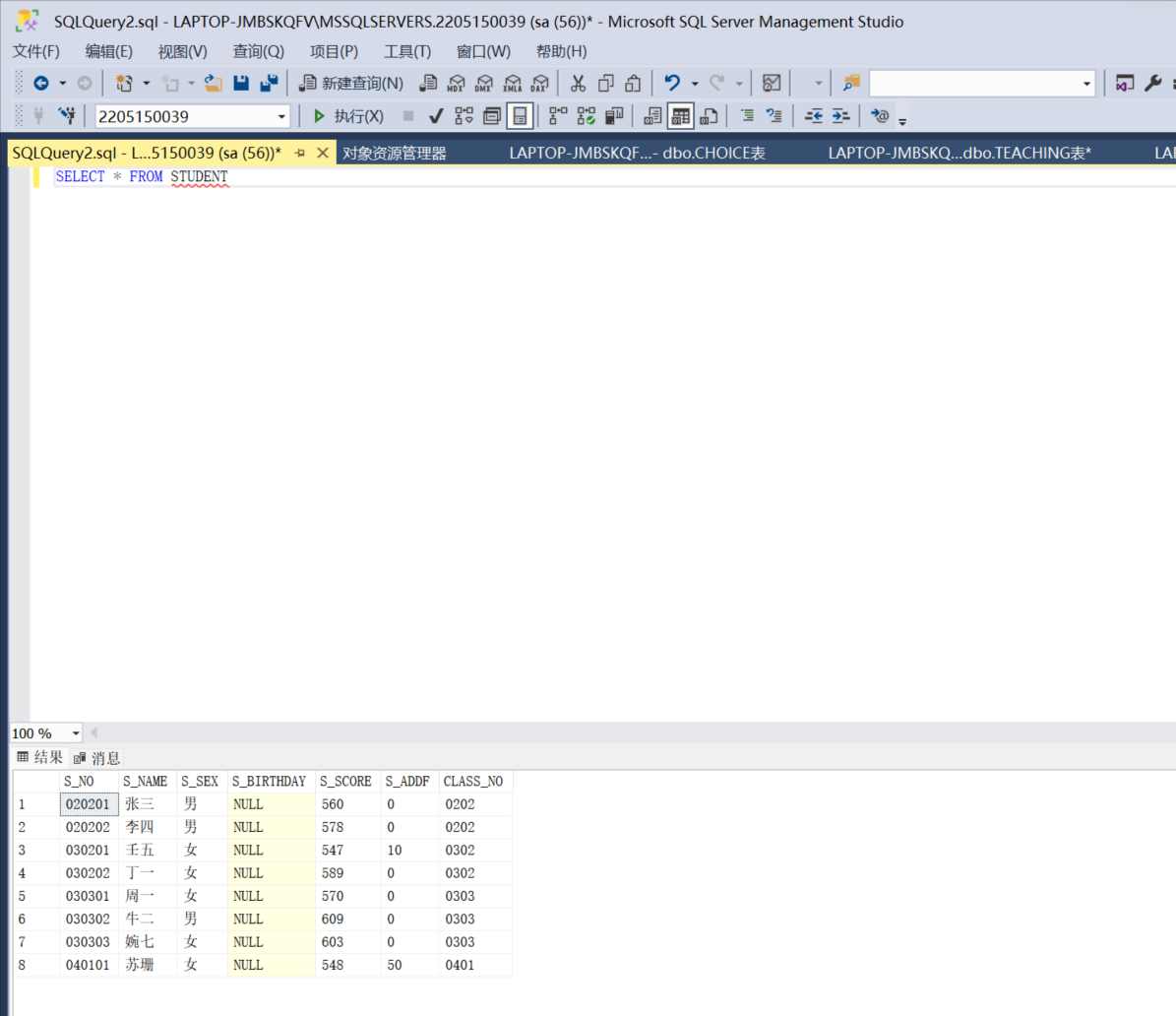
1. TEACHING表



## 3. 完成以下要求，给出实现该要求的SQL语句

（1）查询学生表中所有学生的全部信息

SELECT \*FROM STUDENT

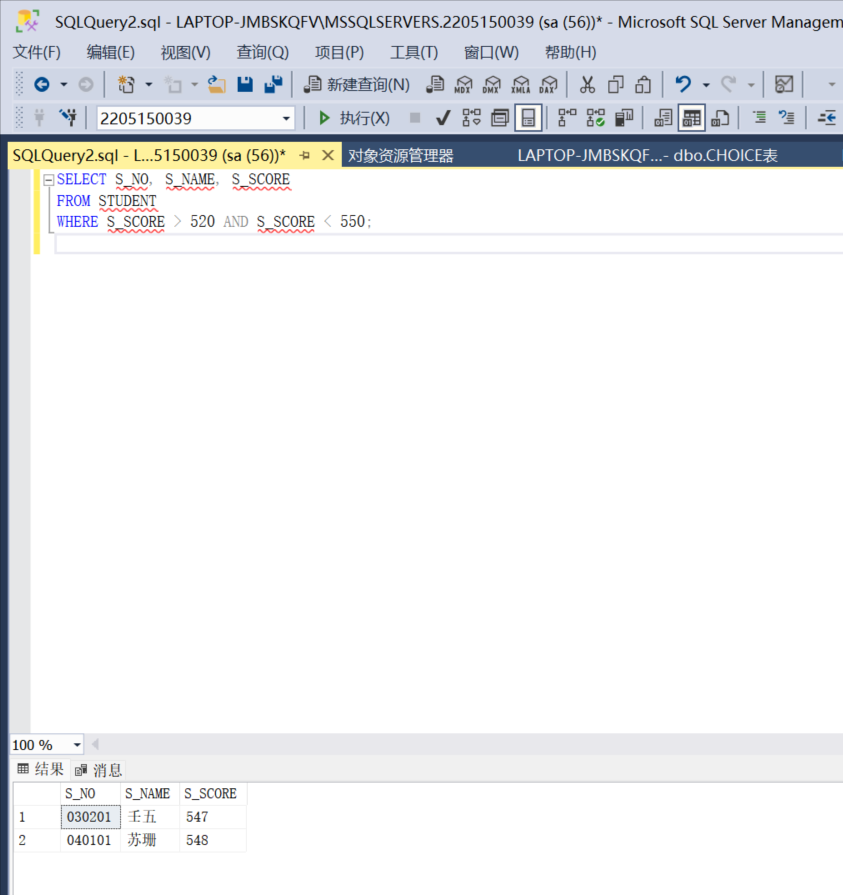


1. 查询student表中入学总分在520和550之间的学生的学号、姓名、入学总分信息；

SEECT S\_NO,S\_NAME,S\_SCORE

FROM STUDENT

WHERE S\_SCORE>520 AND S\_SCORE<550;

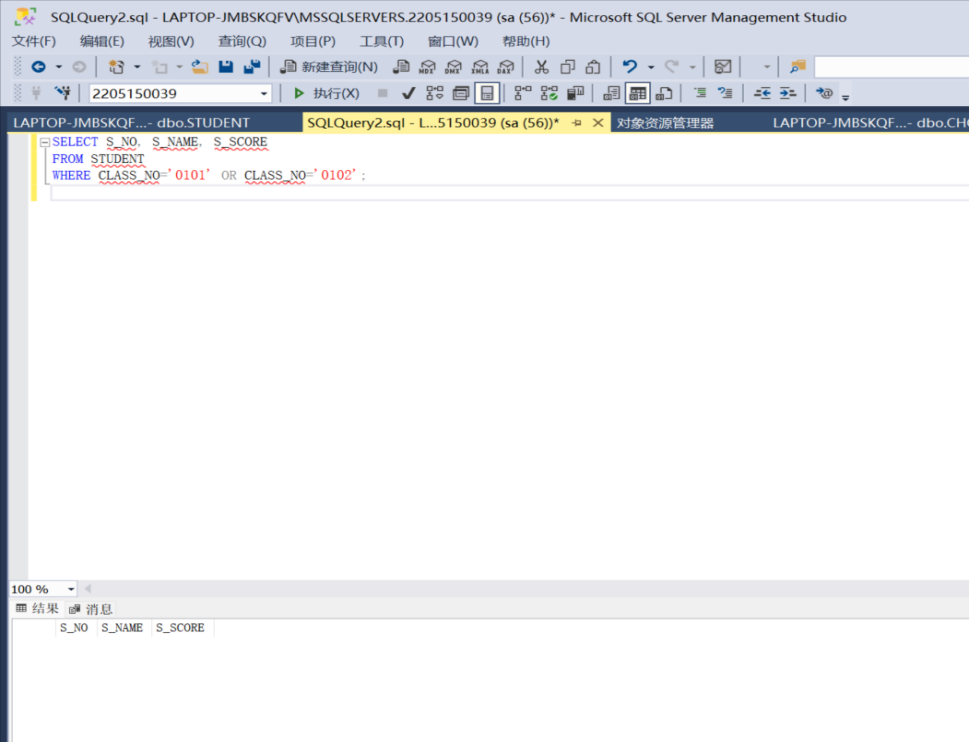


1. 查询student表中班级编码为“0101”和“0201”的学生的学号、姓名和入学总分信息；

SELECT S\_NO,S\_NAME,S\_SCORE

FROM STUDENT

WHERE CLASS\_NO=’0101’ OR CLASS\_NO=’0201’



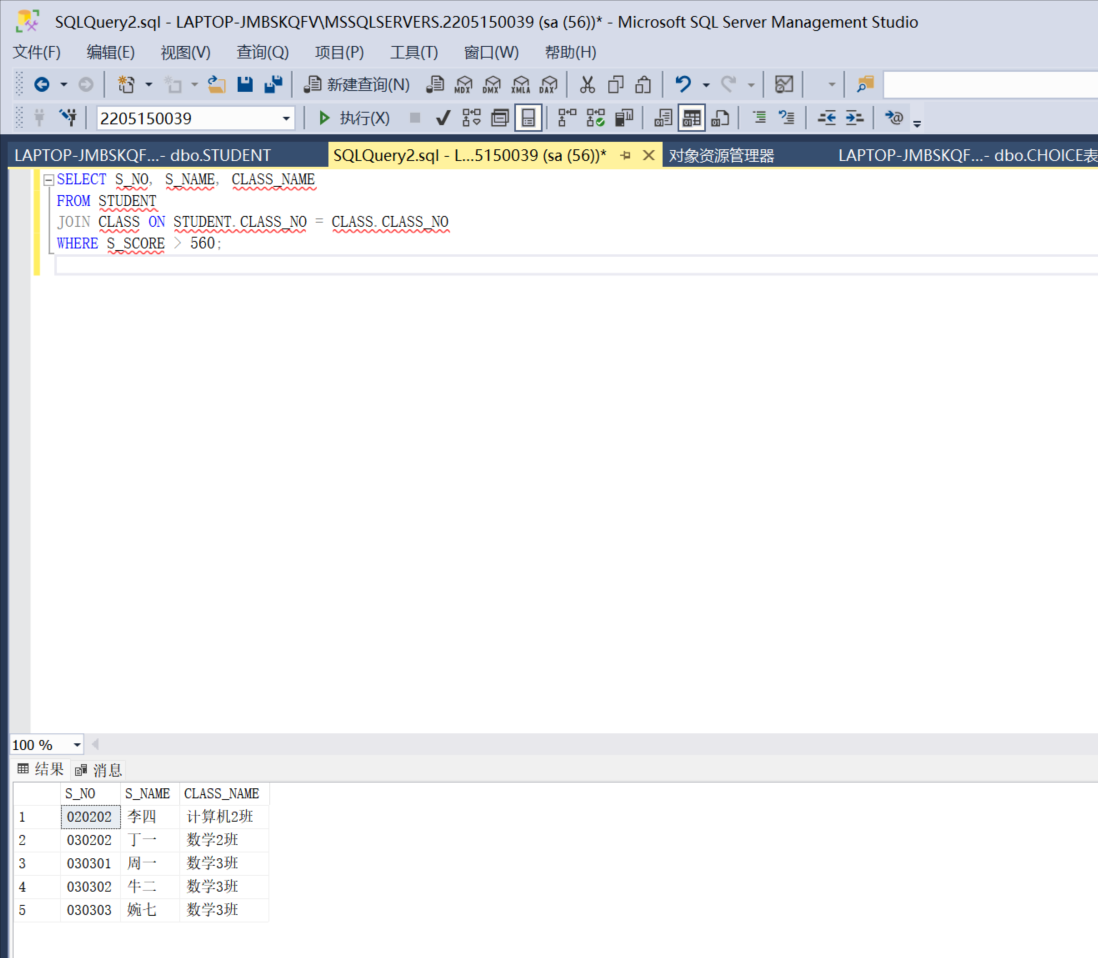
1. 查询入学总分大于560的学生的学号、姓名、总分和班级名称等信息；

SELECT S\_NO,S\_NAME,CLASS\_NAME

FROM STUDENT

JOIN CLASS ON STUDENT.CLASS\_NO= CLASS.CLASS\_NO

WHERE S\_SCORE>560



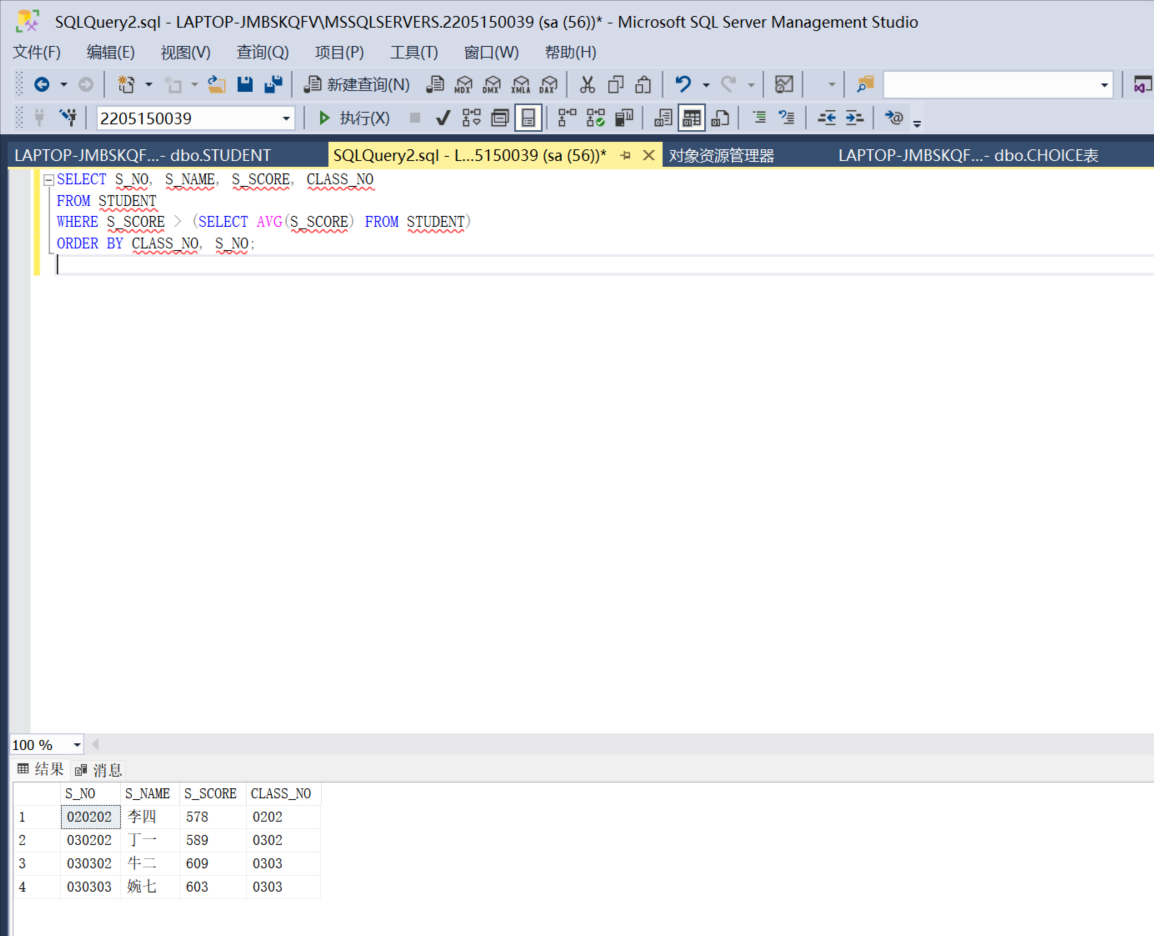
（5）查询入学总分高于平均总分的学生的学号、姓名、总分、班级编号信息，并将显示结果按班级编号、学号排列；

SELECT S\_NO,S\_NAMEL,S\_SCORE,CLASS\_NO

FROM STUDENT

WHERE S\_SCORE>(SELECT AVG(S\_SCORE FROM STUDENT)

ORDER BY CLASS\_NO,S\_NO;



（6）查询入学附加分不为0的学生的学号、姓名、入学总分、入学附加分、班级编号、班级名信息，并将显示结果按学号排序；

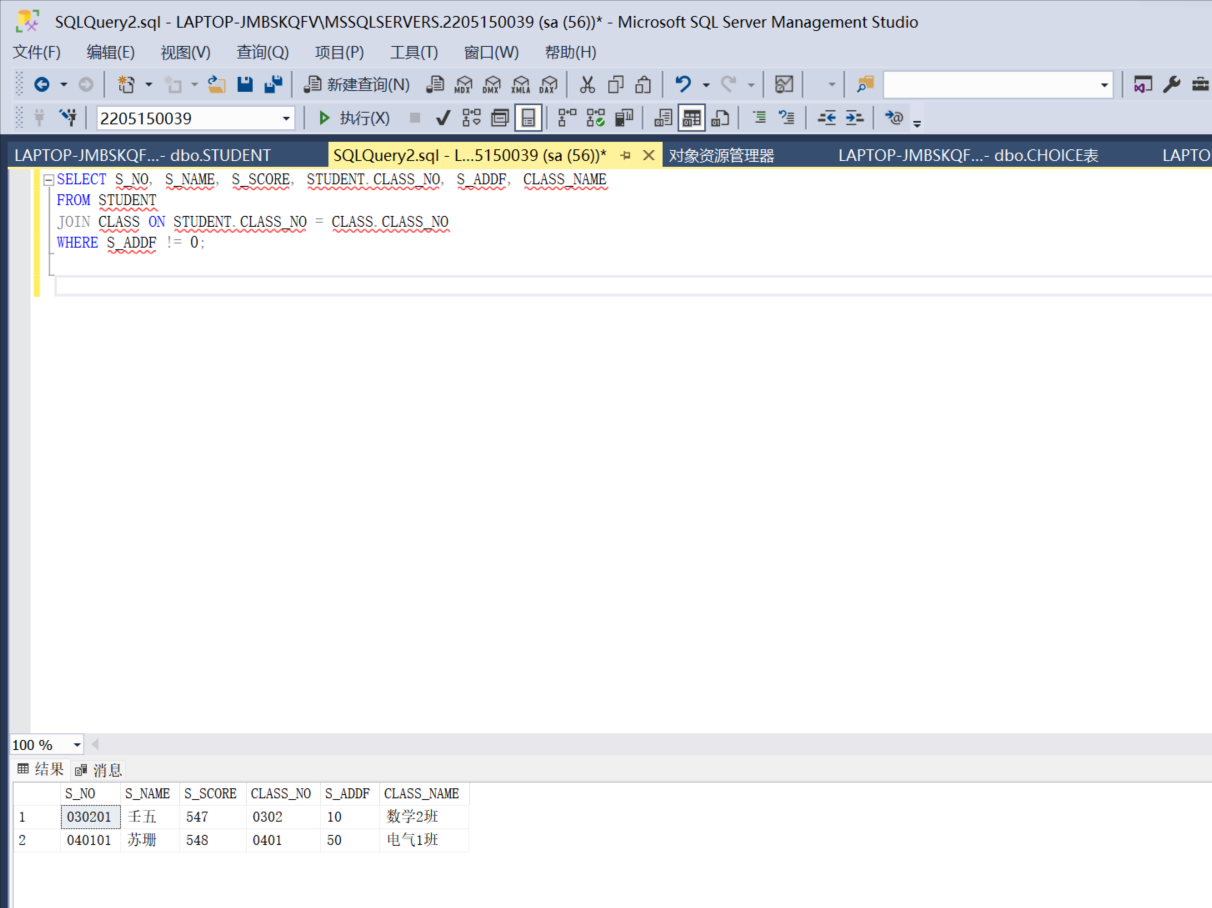
SELECT S\_NO, S\_NAME, S\_SCORE, STUDENT.CLASS\_NO, S\_ADDF, CLASS.CLASS\_NAME

FROM STUDENT

JOIN CLASS ON STUDENT.CLASS\_NO = CLASS.CLASS\_NO

WHERE S\_ADDF != 0

ORDER BY S\_NO;



（7）查询“马克思哲学”课程的JOIN分数高于计算机2班的“马克思哲学”课程平均分的学生的学号和姓名；

SELECT STUDENT.S\_NO, S\_NAME

FROM STUDENT

JOIN CHOICE ON STUDENT.S\_NO=CHOICE.S\_NO

JOIN COURSE ON CHOICE.COURSE\_NO=COURSE.COURSE\_NO

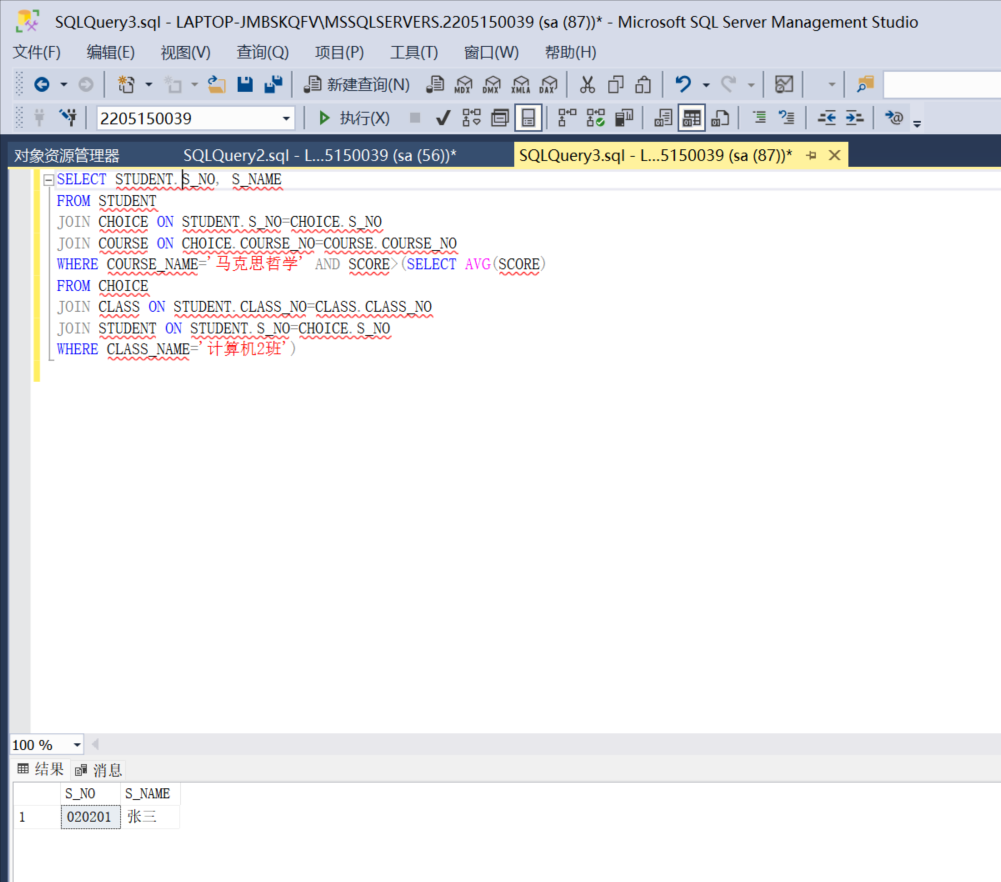
WHERE COURSE\_NAME='马克思哲学' AND SCORE>(SELECT AVG(SCORE)

FROM CHOICE

JOIN CLASS ON STUDENT.CLASS\_NO=CLASS.CLASS\_NO

JOIN STUDENT ON STUDENT.S\_NO=CHOICE.S\_NO

WHERE CLASS\_NAME='计算机2班')



（8）查询每个教师所担任课程的课程编号，课程名以及该课程的平均分；

SELECT TEACHER.T\_NAME,TEACHER.T\_NO,COURSE\_NAME,AVG(CHOICE.SCO RE) AS AVG\_SCORE

FROM TEACHER

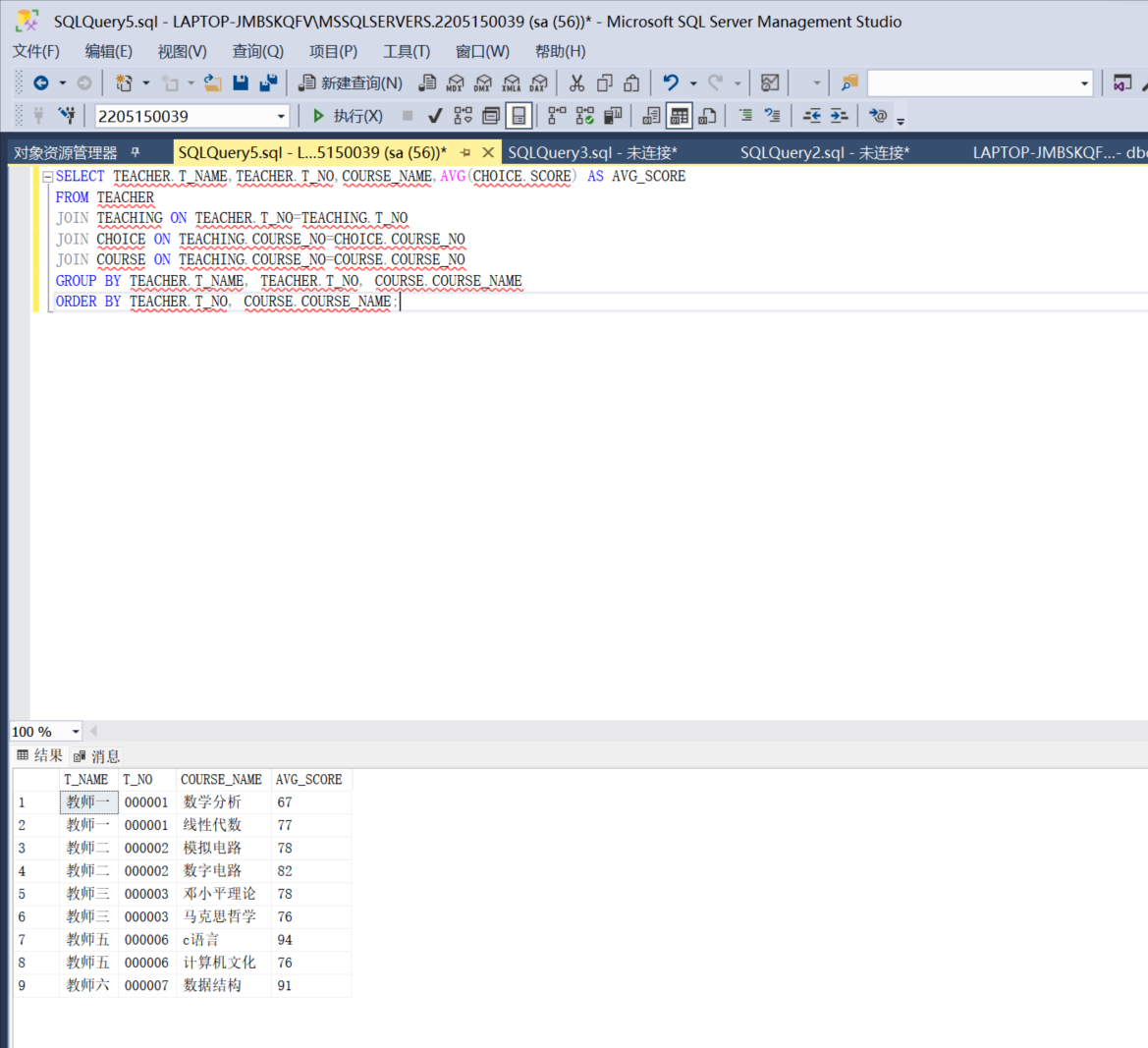
JOIN TEACHING ON TEACHER.T\_NO=TEACHING.T\_NO

JOIN CHOICE ON TEACHING.COURSE\_NO=CHOICE.COURSE\_NO

JOIN COURSE ON TEACHING.COURSE\_NO=COURSE.COURSE\_NO

GROUP BY TEACHER.T\_NAME, TEACHER.T\_NO, COURSE.COURSE\_NAME

ORDER BY TEACHER.T\_NO, COURSE.COURSE\_NAME;



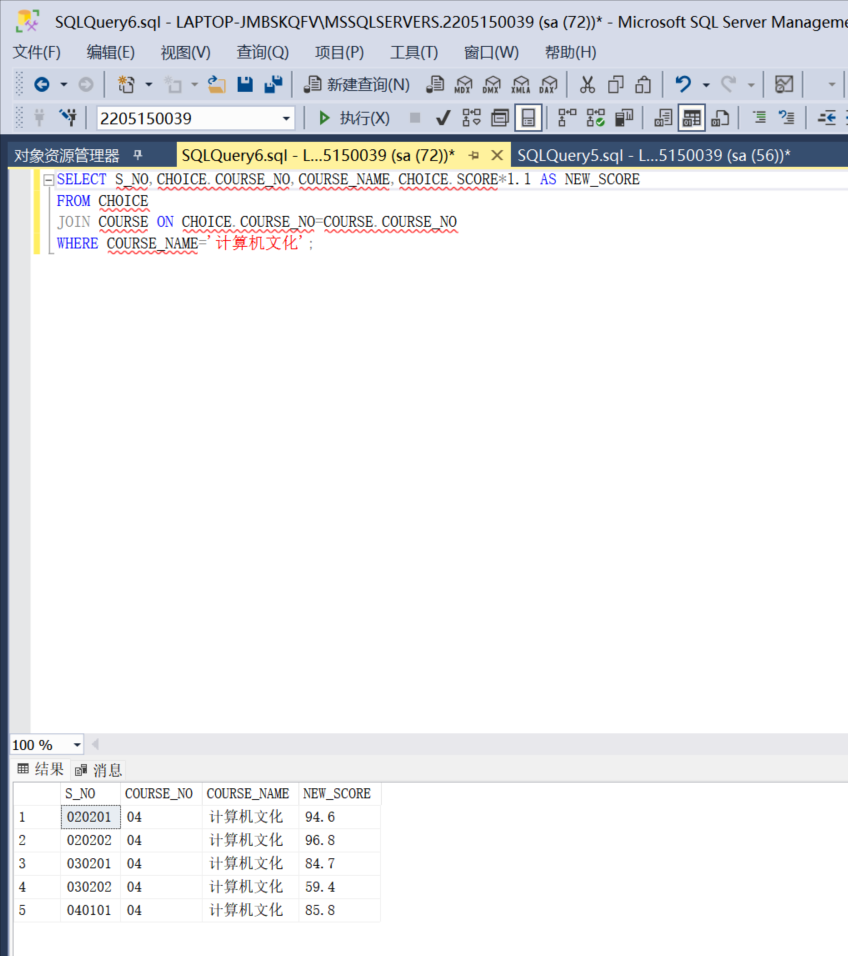
（9）把所有学生的“计算机文化”课程的成绩提高10%；

SELECT S\_NO,CHOICE.COURSE\_NO,COURSE\_NAME,CHOICE.SCORE\*1.1 AS NEW\_SCORE

FROM CHOICE

JOIN COURSE ON CHOICE.COURSE\_NO=COURSE.COURSE\_NO

WHERE COURSE\_NAME='计算机文化';



（10）删除“丁一”的所有数据；

DELETE STUDENT

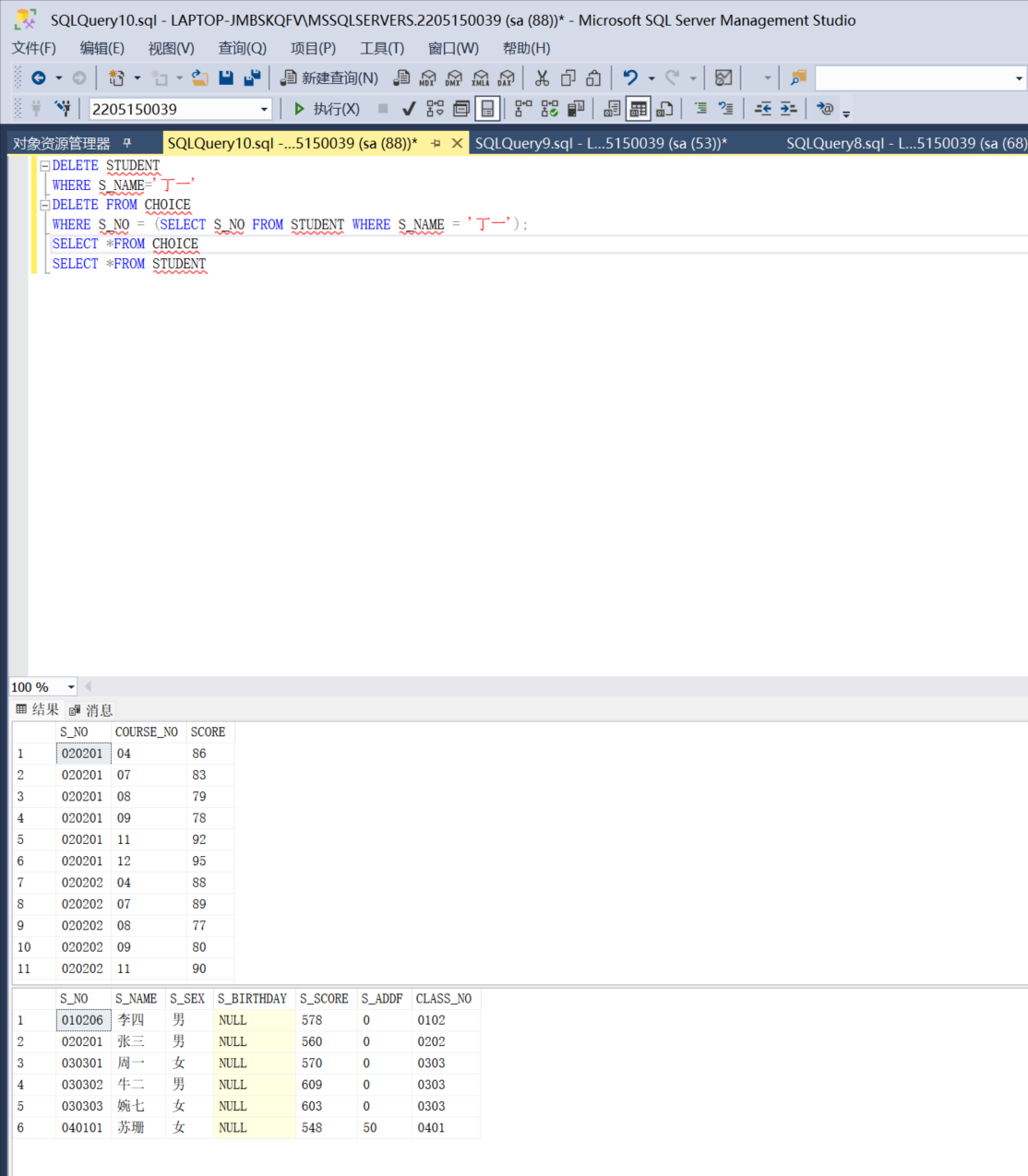
WHERE S\_NAME='丁一'

DELETE FROM CHOICE

WHERE S\_NO = (SELECT S\_NO FROM STUDENT WHERE S\_NAME = '丁 一');

SELECT \*FROM CHOICE

SELECT \*FROM STUDENT



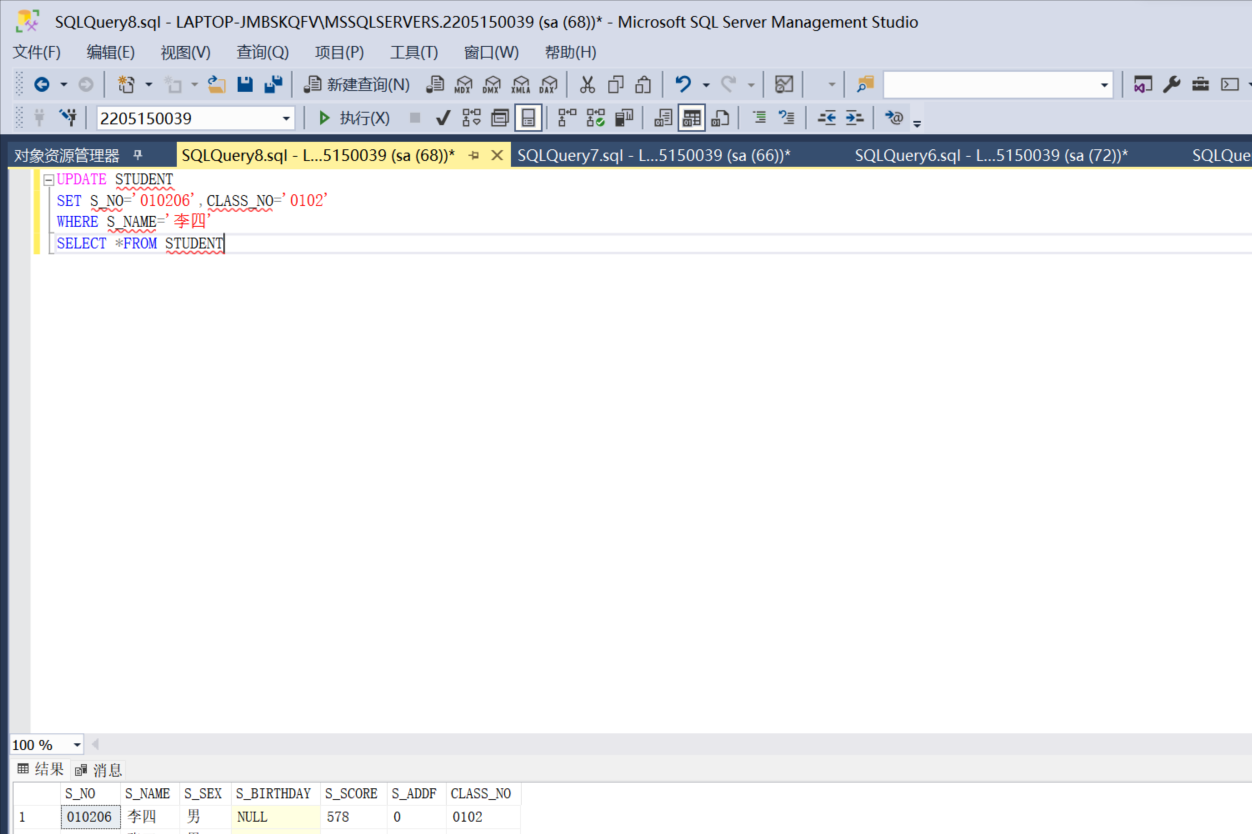
1. 将student表中李四的学生编号改为“010206”，班级编号改为“0102”；

UPDATE STUDENT

SET S\_NO='010206',CLASS\_NO='0102'

WHERE S\_NAME='李四'

SELECT \*FROM STUDENT



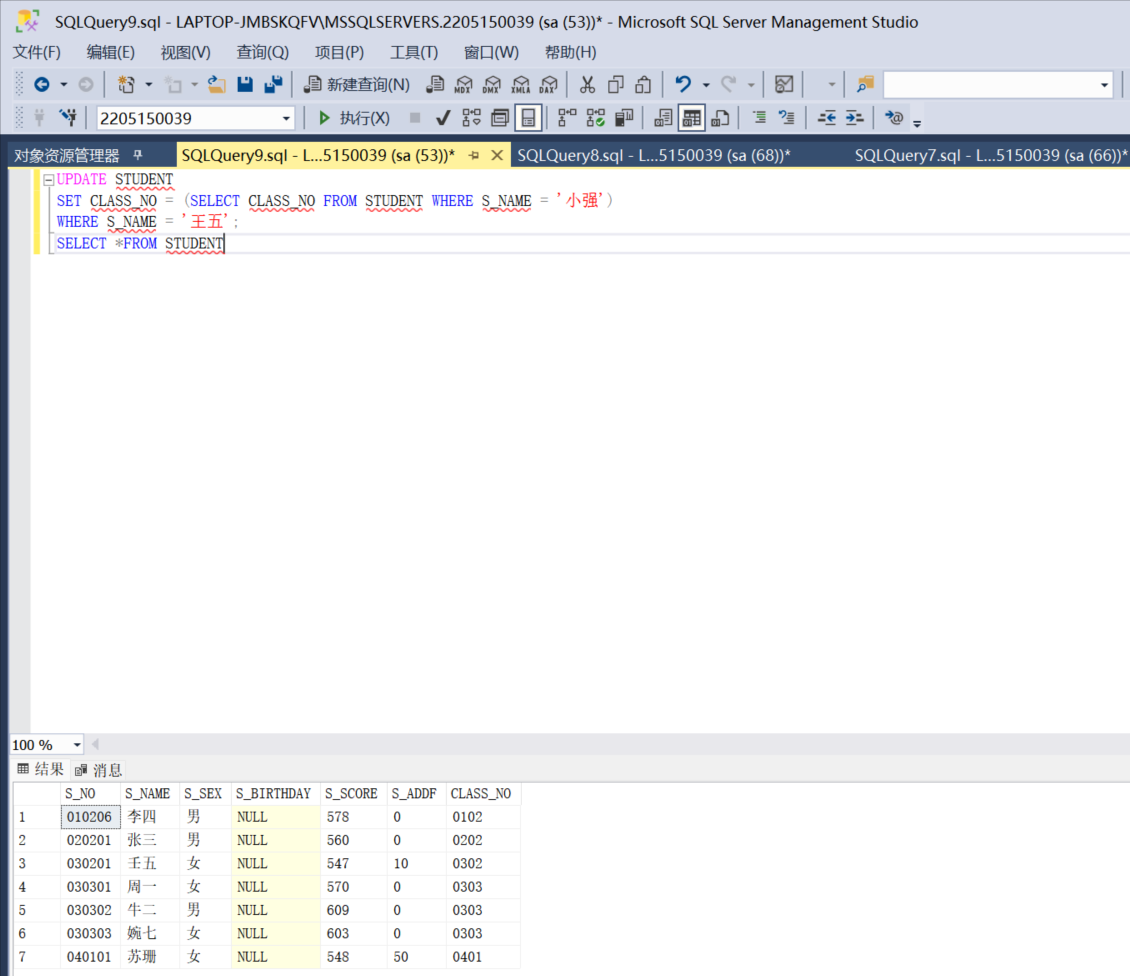
（12）将student表中王五的班级编号改为小强的班级编号；

UPDATE STUDENT

SET CLASS\_NO = (SELECT CLASS\_NO FROM STUDENT WHERE S\_NAME = '小强')

WHERE S\_NAME = '王五';

SELECT \*FROM STUDENT

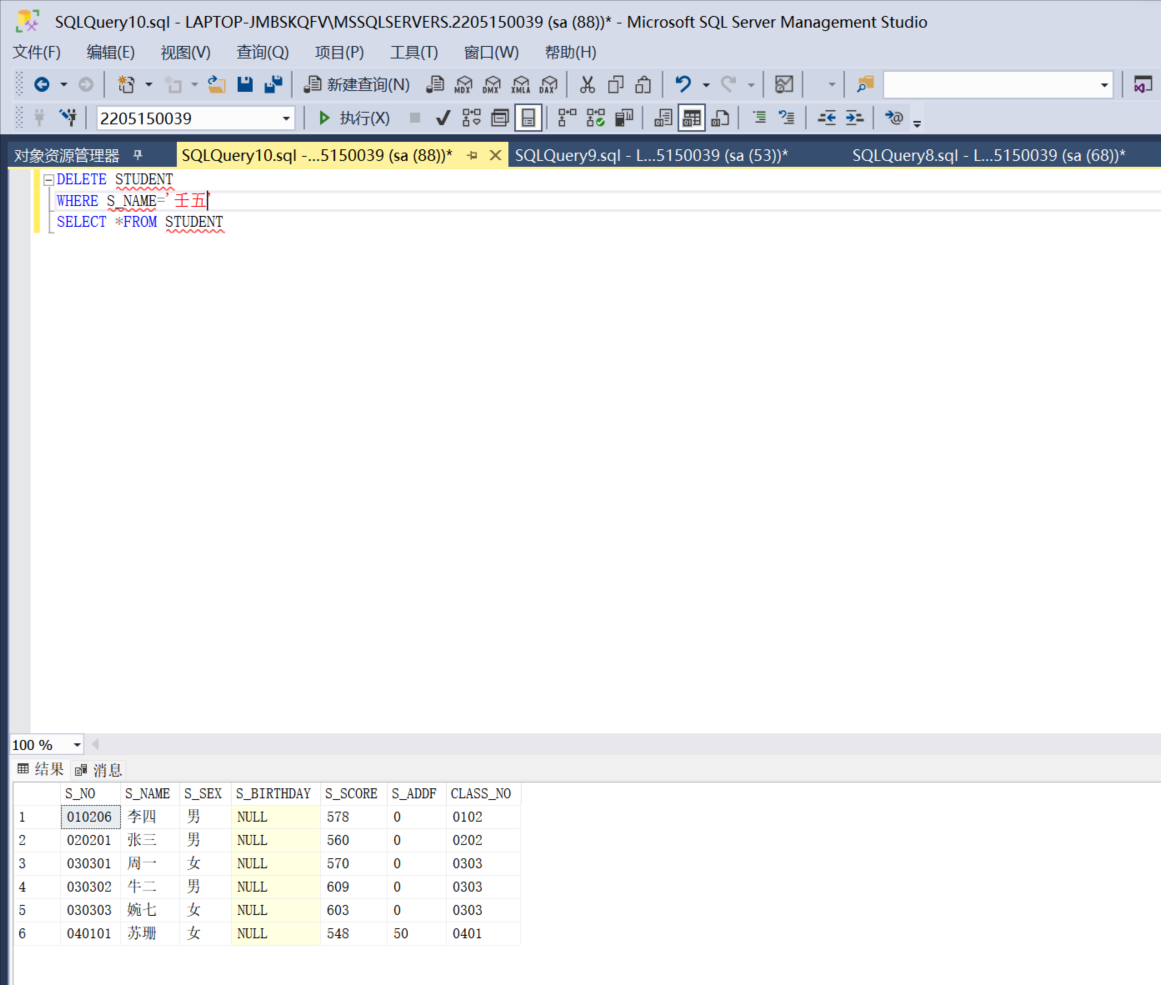


（13）将student表中王五的记录删除；

DELETE STUDENT

WHERE S\_NAME=’壬五’

SELECT \*FROM STUDENT



（14）创建一个视图，数学3班学生的选课记录；

CREATE VIEW V\_MATH3\_CHOICE AS

SELECT

STUDENT.S\_NO,

STUDENT.S\_NAME,

CLASS.CLASS\_NAME,

COURSE.COURSE\_NO,

COURSE.COURSE\_NAME,

CHOICE.SCORE

FROM CHOICE

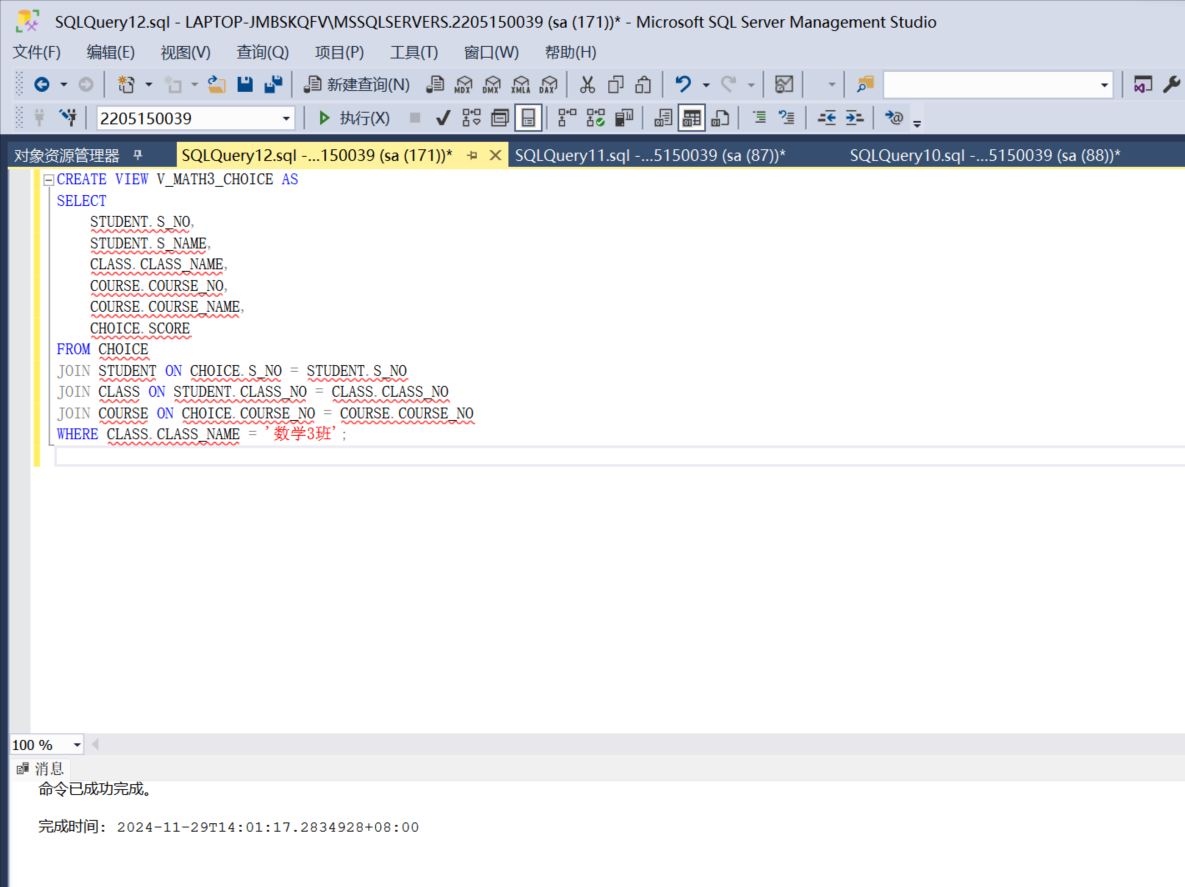
JOIN STUDENT ON CHOICE.S\_NO = STUDENT.S\_NO

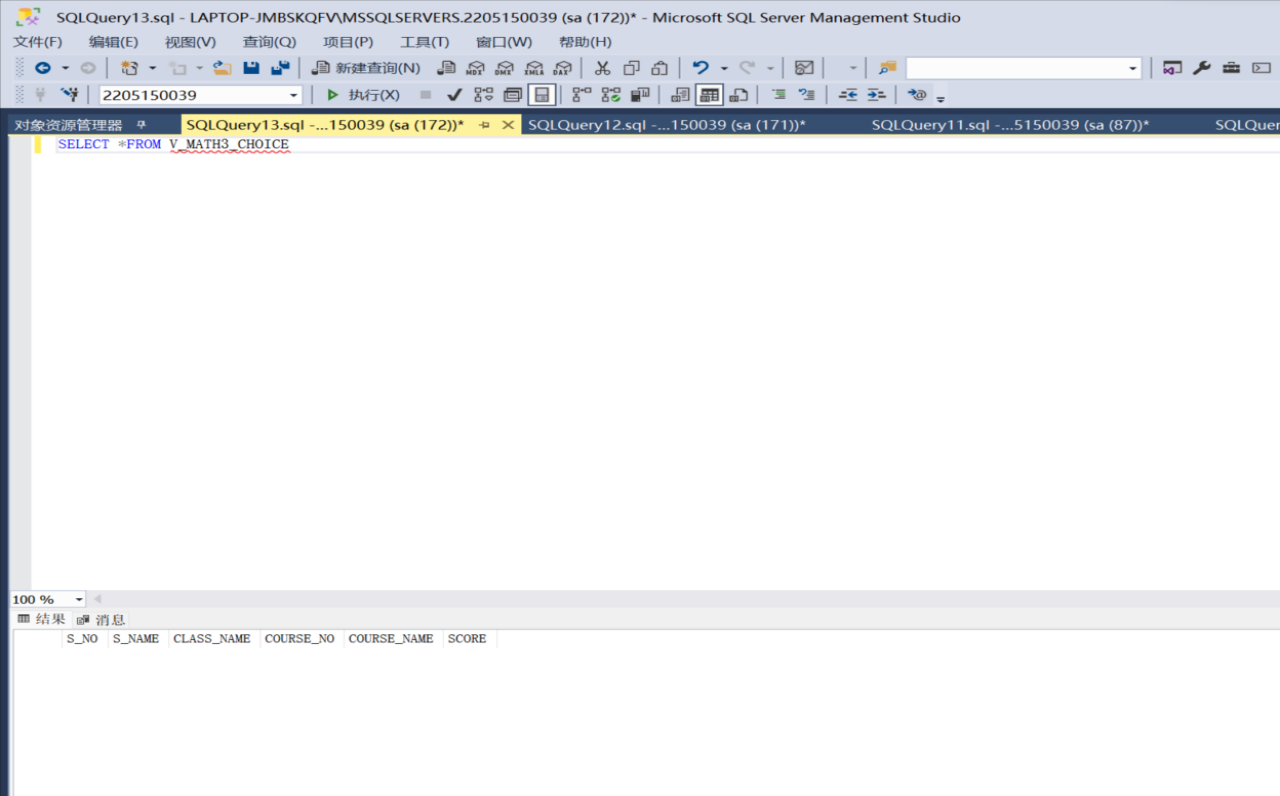
JOIN CLASS ON STUDENT.CLASS\_NO = CLASS.CLASS\_NO

JOIN COURSE ON CHOICE.COURSE\_NO = COURSE.COURSE\_NO

WHERE CLASS.CLASS\_NAME = '数学3班';

SELECT \*FROM V\_MATH3\_CHOICE





1. 创建一个视图，选修c语言的学生记录。

CREATE VIEW V\_C\_LANGUAGE\_CHOICE AS

SELECT

STUDENT.S\_NO,

STUDENT.S\_NAME,

CLASS.CLASS\_NAME,

COURSE.COURSE\_NO,

COURSE.COURSE\_NAME,

CHOICE.SCORE

FROM CHOICE

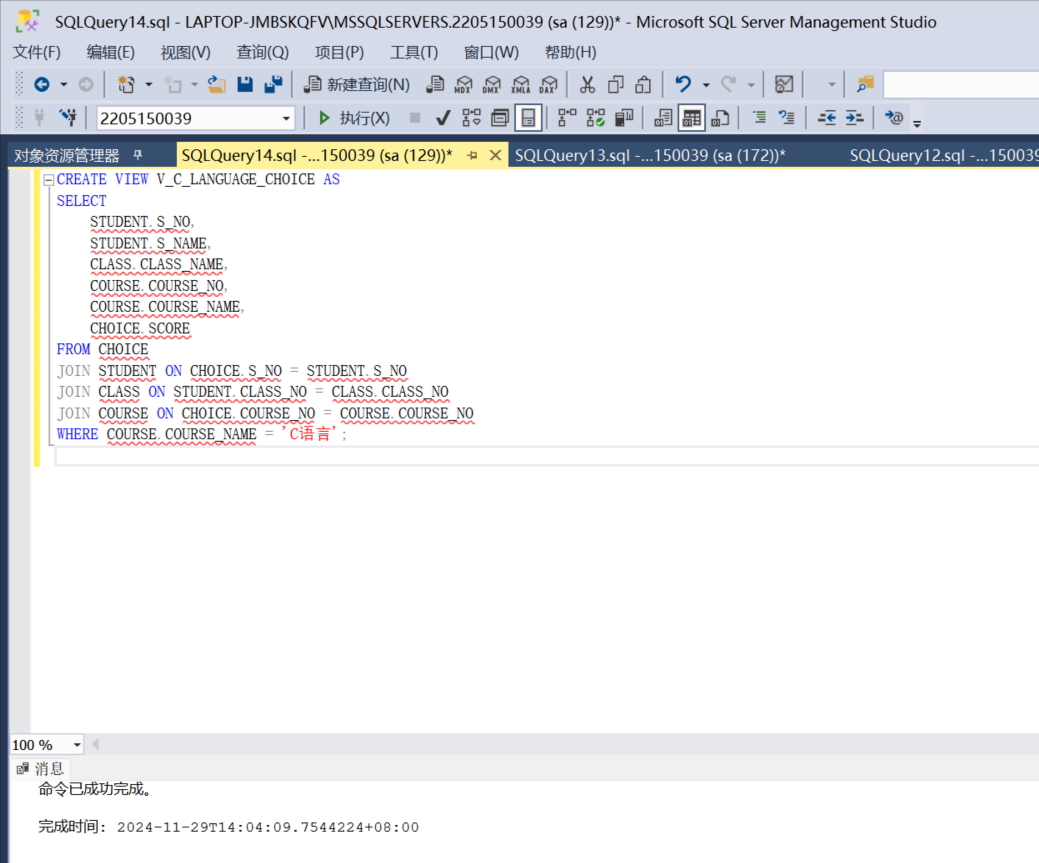
JOIN STUDENT ON CHOICE.S\_NO = STUDENT.S\_NO

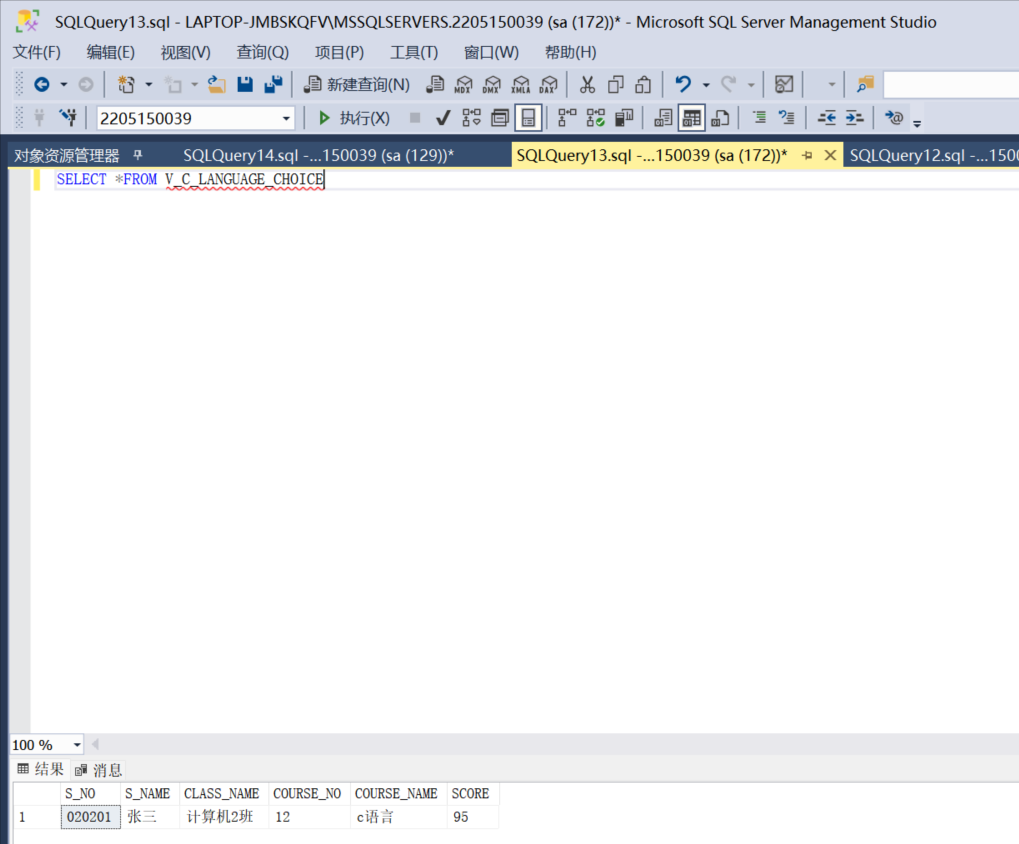
JOIN CLASS ON STUDENT.CLASS\_NO = CLASS.CLASS\_NO

JOIN COURSE ON CHOICE.COURSE\_NO = COURSE.COURSE\_NO

WHERE COURSE.COURSE\_NAME = 'C语言';

SELECT \*FROM V\_C\_LANGUAGE\_CHOICE





## 总结

本次实验，完成了数据库表的设计与创建，根据给定的需求设计并创建了多个数据库表，包括学生表、教师表、课程表、选课表等。每个表都设置了适当的字段类型和约束（如主键、外键等），确保数据的一致性和完整性。向数据库表中插入了模拟数据，并进行了数据更新操作，如修改学生的班级、删除指定学生的数据等。使用 SELECT 语句进行数据查询，结合 WHERE、ORDER BY 和 GROUP BY 等子句来满足不同的查询需求。进行了如查询班级学生信息、查询课程的平均成绩等操作。创建了多个视图，如查询“数学3班”学生的选课记录、查询选修“C语言”的学生等。视图简化了复杂查询，提供了对特定数据的定制化展示。使用了 JOIN 连接多张表，结合多条件查询，解决了多个表之间的数据关联问题。例如，查询每个教师所授课程的课程编号及其平均分，查询学生成绩高于某班级平均成绩的学生等。

实验中遇到的问题与解决方法，在创建视图时，遇到了一些列名重复或不明确的问题。解决方法是确保视图中的每个字段来源明确，并且所有字段都正确地连接和指定。在使用 JOIN 连接多张表时，出现了连接条件错误的情况，导致查询结果不正确。通过检查各个表之间的连接条件，确保连接字段正确，解决了这个问题。有时候查询返回空结果集，这可能是因为查询条件过于严格，导致没有符合条件的记录。

通过实验，我深入理解了 SQL 语句的基本结构和语法，能够熟练地进行数的增删改查操作。通过实际操作，我提升了使用 JOIN、GROUP BY 和子查询等复杂 SQL 查询的能力，能够处理多表关联查询和数据聚合分析。通过本次实验，我对 SQL 数据库的基本操作有了更加深入的了解，并能独立完成数据库的表结构设计、数据操作和查询任务。