|  |
| --- |
| **《数据库系统SSD7》**  **实验报告**  专业班级 软件工程2306班  学 号 8209230614  姓 名 李兴  **实验成绩：**  **批阅教师：**  2025年 4月25日 |

# 实验2《数据表查询与更新》

实验学时： 2 实验地点： 信息楼404 实验日期： 2025/4/25

**一、实验目的**

1. 熟悉和掌握对数据表中数据的查询操作和SQL命令的使用，学会灵活熟练的使用SQL 语句的各种形式，加深理解关系运算的各种操作（尤其是关系的选择，投影，连接和除运算）；

2. 熟悉和掌握数据表中数据的插入、修改、删除操作和命令的使用（熟悉使用UPDATE/INSERT/DELETE语句进行表操作）；加深理解表的定义对数据更新的作用。

**二、实验内容**

（一）在表S，C，SC上完成以下查询：

1． 查询学生的基本信息；

2． 查询“CS”系学生的基本信息；

3． 查询“CS”系学生年龄不在19到21之间的学生的学号、姓名；

4． 找出“CS”系年龄最大的学生，显示其学号、姓名；

5． 找出各系年龄最大的学生，显示其学号、姓名；

6． 统计“CS”系学生的人数；

7． 统计各系学生的人数，结果按升序排列；

8． 按系统计各系学生的平均年龄，结果按降序排列；

9． 查询无先修课的课程的课程名和学分数；

10．统计每位学生选修课程的门数、学分及其平均成绩；

11．统计选修每门课程的学生人数及各门课程的平均成绩；

12．找出平均成绩在85分以上的学生，结果按系分组，并按平均成绩的升序排列；

13．查询选修了“1”或“2”号课程的学生学号和姓名；

14．查询选修了课程名为“数据库系统”且成绩在60分以下的学生的学号、姓名和成绩；

15．查询每位学生选修了课程的学生信息（显示：学号，姓名，课程号，课程名，成绩）；

16．查询没有选修课程的学生的基本信息；

17．查询选修了3门以上课程的学生学号；

18．查询选修课程成绩至少有一门在80分以上的学生学号；

19．查询选修课程成绩均在80分以上的学生学号；

（二）在表S、C、SC中完成下列更新：

1．  将数据分别插入表S、C、SC；

2．  将表S、C、SC中的数据保存在磁盘上。

3．  在表S、C、SC上练习数据的插入、修改、删除操作。（比较在表上定义/未定义主码（Primary Key）或外码（Foreign Key）时的情况）

4．  将表S、C、SC中的数据全部删除，再利用磁盘上备份的数据来恢复数据。

5．  如果要在表SC中插入某个学生的选课信息（如：学号为“2007001005”，课程号为“c123”，成绩待定），应如何进行？

6．  求各系学生的平均成绩，并把结果存入数据库；

7．  将“CS”系全体学生的成绩置零；

8．  删除“CS”系全体学生的选课记录；

9．  删除学号为“S1”的相关信息；

10．将学号为“S1”的学生的学号修改为“S001”；

11．把平均成绩大于80分的男同学的学号和平均成绩存入另一个表S——GRADE（SNO，AVG\_GRADE）；

12． 把选修了课程名为“数据结构”的学生的成绩提高10%；

13． 把选修了“C2”号课程，且成绩低于该门课程的平均成绩的学生成绩删除掉。

1. **实验方法与实验步骤**

**实验方法：**

1.查询操作实验方法：

基础查询方法

聚合查询方法

连接查询方法

嵌套查询方法

集合运算方法

2.数据更新操作实验方法：

数据插入方法

数据修改方法

数据删除方法

完整性约束验证方法

数据备份与恢复方法

3.实验对比分析方法：

约束条件对比

操作效率对比

结果验证方法

异常处理方法

**实验步骤：**

**查询：**

1.在表Ｓ、Ｃ、ＳＣ上进行简单查询、连接查询、嵌套查询；

2.使用聚合函数的查询、对数据分组查询、对数据的排序查询。

**插入：**

1.用ＳＱＬ命令将数据插入当前数据库的表Ｓ、Ｃ、ＳＣ中；

2.用ＳＱＬ命令形式修改表Ｓ、Ｃ、ＳＣ中的数据；

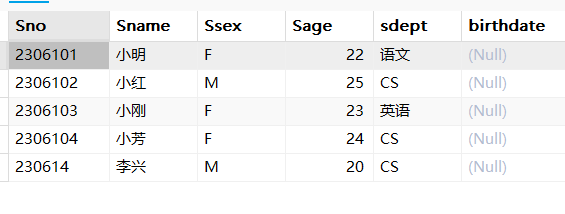
3.用ＳＱＬ命令形式  删除表Ｓ、Ｃ、ＳＣ中的数据。

1. **实验结果**

**以sql语句结合截图的形式展示：**

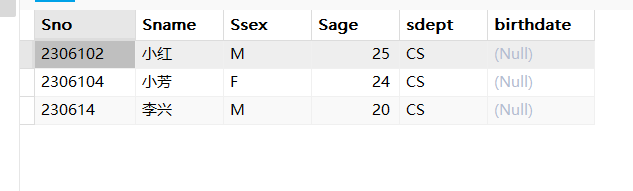
1.查询学生的基本信息：

SELECT \* FROM S;



2.查询“CS”系学生的基本信息：

SELECT \* FROM S WHERE sdept = 'CS';



3.查询“CS”系学生年龄不在19到21之间的学生的学号、姓名：

SELECT Sno, Sname FROM S WHERE sdept = 'CS' AND (Sage < 19 OR Sage > 21);



4.找出“CS”系年龄最大的学生，显示其学号、姓名：

SELECT Sno, Sname FROM S

WHERE sdept = 'CS' AND Sage = (

SELECT MAX(Sage) FROM S WHERE sdept = 'CS'

);



5.找出各系年龄最大的学生，显示其学号、姓名：

SELECT s1.sdept,s1.Sno, s1.Sname

FROM S s1

WHERE s1.Sage = (

SELECT MAX(s2.Sage)

FROM S s2

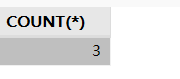
WHERE s2.sdept = s1.sdept

);



6.统计“CS”系学生的人数：

SELECT COUNT(\*) FROM S WHERE sdept = 'CS';



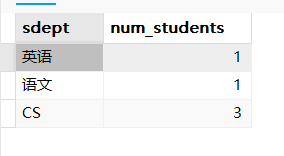
7.统计各系学生的人数，结果按升序排列：

SELECT sdept, COUNT(\*) AS num\_students

FROM S

GROUP BY sdept

ORDER BY num\_students ASC;



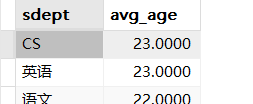
8.按系统计各系学生的平均年龄，结果按降序排列：

SELECT sdept, AVG(Sage) AS avg\_age

FROM S

GROUP BY sdept

ORDER BY avg\_age DESC;



9.查询无先修课的课程的课程名和学分数：

SELECT Cname, ccredit

FROM C

WHERE Cpno IS NULL;



10.统计每位学生选修课程的门数、学分及其平均成绩：

SELECT S.Sno, Sname, COUNT(SC.Cno) AS course\_count,

SUM(C.ccredit) AS total\_credits,

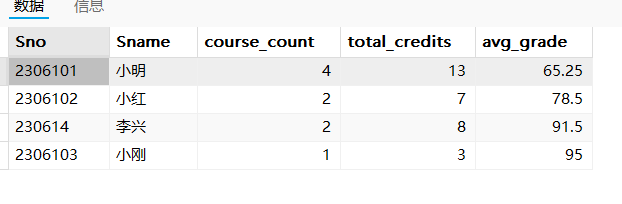
AVG(SC.grade) AS avg\_grade

FROM S

JOIN SC ON S.Sno = SC.Sno

JOIN C ON SC.Cno = C.Cno

GROUP BY S.Sno, Sname;

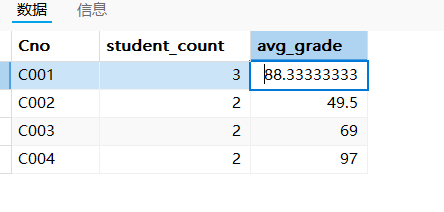


11.统计选修每门课程的学生人数及各门课程的平均成绩：

SELECT Cno, COUNT(Sno) AS student\_count, AVG(grade) AS avg\_grade

FROM SC

GROUP BY Cno;



12.找出平均成绩在85分以上的学生，结果按系分组，并按平均成绩的升序排列：

SELECT S.sdept, S.Sno, Sname, AVG(SC.grade) AS avg\_grade

FROM S

JOIN SC ON S.Sno = SC.Sno

GROUP BY S.sdept, S.Sno, Sname

HAVING AVG(SC.grade) > 85

ORDER BY S.sdept, avg\_grade ASC;



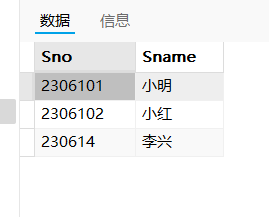
13.查询选修了“1”或“2”号课程的学生学号和姓名：

SELECT DISTINCT S.Sno, Sname

FROM S

JOIN SC ON S.Sno = SC.Sno

WHERE SC.Cno IN ('C001', 'C002');



14.查询选修了课程名为“数据库系统”且成绩在60分以下的学生的学号、姓名和成绩：

SELECT S.Sno, Sname, SC.grade

FROM S

JOIN SC ON S.Sno = SC.Sno

JOIN C ON SC.Cno = C.Cno

WHERE C.Cname = '数据库系统' AND SC.grade < 60;



15.查询每位学生选修了课程的学生信息（显示：学号，姓名，课程号，课程名，成绩）

SELECT S.Sno, Sname, C.Cno, Cname, grade

FROM S

JOIN SC ON S.Sno = SC.Sno

JOIN C ON SC.Cno = C.Cno;

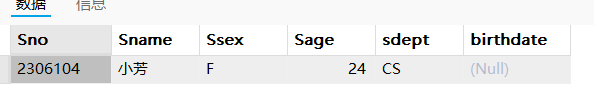


16.查询没有选修课程的学生的基本信息

SELECT \*

FROM S

WHERE Sno NOT IN (SELECT DISTINCT Sno FROM SC);



17.查询选修了3门以上课程的学生学号

SELECT Sno

FROM SC

GROUP BY Sno

HAVING COUNT(Cno) > 3;

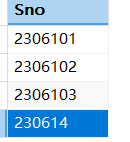


18.查询选修课程成绩至少有一门在80分以上的学生学号

SELECT DISTINCT Sno

FROM SC

WHERE grade >= 80;



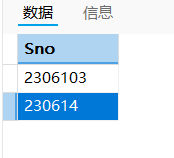
19.查询选修课程成绩均在80分以上的学生学号

SELECT Sno

FROM SC

GROUP BY Sno

HAVING MIN(grade) >= 80;



（二）在表S、C、SC中完成下列更新：

1.将数据分别插入表S、C、SC

我先删除原来的表中数据，再插入合理的数据

INSERT INTO S (Sno, Sname, Ssex, Sage, sdept) VALUES

('S001', '张三', 'M', 22, 'CS'),

('S002', '李四', 'F', 20, 'CS'),

('S003', '王五', 'M', 18, 'CS'),

('S004', '赵六', 'F', 21, 'CS'),

('S005', '刘七', 'M', 23, 'Math'),

('S006', '周八', 'M', 19, 'Math'),

('S007', '孙九', 'F', 22, 'Physics'),

('S008', '钱十', 'M', 20, 'Physics'),

('S009', '胡十一', 'F', 20, 'English');

INSERT INTO C (Cno, Cname, Cpno, ccredit) VALUES

('C001', '高等数学', NULL, 5),

('C002', '大学英语', NULL, 2),

('C003', '数据结构', 'C001', 3),

('C004', '操作系统', 'C003', 3),

('C005', '数据库系统', 'C003', 3),

('C006', '人工智能', NULL, 4),

('1', '网络原理', NULL, 2),

('2', '电路分析', NULL, 3);

INSERT INTO SC (Sno, Cno, grade) VALUES

('S001', 'C001', 90),

('S001', 'C003', 88),

('S001', 'C005', 92),

('S001', '1', 89),

('S002', 'C005', 55),

('S002', 'C002', 75),

('S003', 'C001', 82),

('S003', 'C003', 84),

('S003', 'C004', 85),

('S004', 'C005', 59),

('S004', 'C006', 95),

('S004', '2', 65),

('S005', 'C001', 75),

('S005', 'C004', 80),

('S006', 'C002', 68),

('S006', 'C003', 70),

('S007', 'C005', 58),

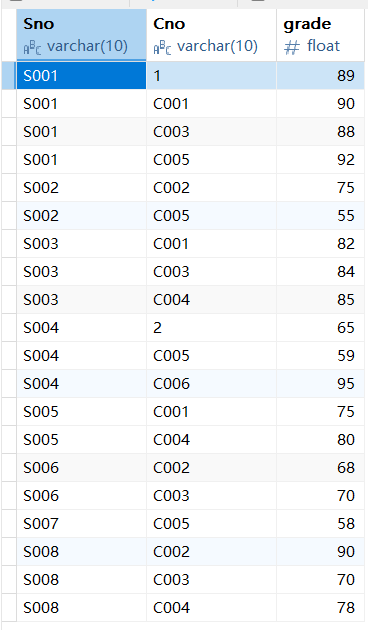
('S008', 'C002', 90),

('S008', 'C004', 78),

('S008', 'C003', 70);







1. 将表S、C、SC中的数据保存在磁盘上

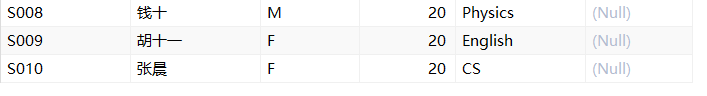
这里依旧使用Navicat Premium中的GUI实现将数据保存到磁盘上

1. 在表S、C、SC上练习数据的插入、修改、删除操作。（比较在表上定义/未定义主码（Primary Key）或外码（Foreign Key）时的情况）

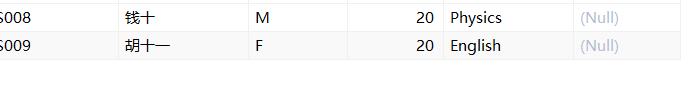
INSERT INTO S (Sno, Sname, Ssex, Sage, sdept) VALUES ('S010', '陈晨', 'F', 20, 'CS');



UPDATE S SET Sname = '张晨' WHERE Sno = 'S010';



DELETE FROM S WHERE Sno = 'S010';



1. 将表S、C、SC中的数据全部删除，再利用磁盘上备份的数据来恢复数据

DELETE FROM SC;

DELETE FROM C;

DELETE FROM S;

这里依旧使用Navicat Premium中的GUI实现利用磁盘上备份的数据来恢复数 据

1. 如果要在表SC中插入某个学生的选课信息（如：学号为“2007001005”，课程号为“c123”，成绩待定），应如何进行？

先插入学生，再插入课程，最后插入选课信息

INSERT INTO S (Sno, Sname, Ssex, Sage, sdept)

VALUES ('2007001005', '测试学生', 'M', 20, 'CS');

INSERT INTO C (Cno, Cname, Cpno, ccredit)

VALUES ('C123', '测试课程', NULL, 2);

INSERT INTO SC (Sno, Cno, grade)

VALUES ('2007001005', 'C123', NULL);



1. 求各系学生的平均成绩，并把结果存入数据库

CREATE TABLE Dept\_AvgGrade (

sdept VARCHAR(20),

avg\_grade FLOAT

);

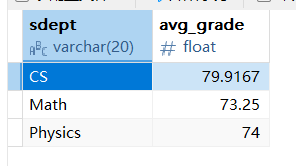
INSERT INTO Dept\_AvgGrade (sdept, avg\_grade)

SELECT S.sdept, AVG(SC.grade)

FROM S

JOIN SC ON S.Sno = SC.Sno

GROUP BY S.sdept;



1. 将“CS”系全体学生的成绩置零

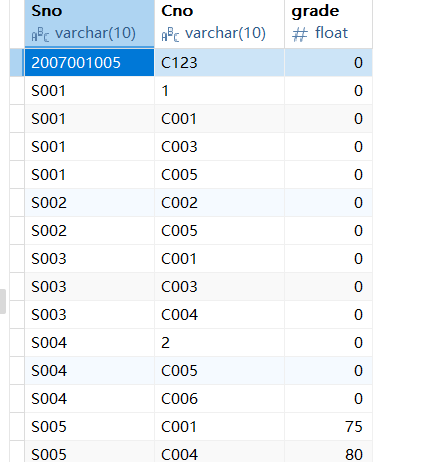
UPDATE SC

SET grade = 0

WHERE Sno IN (

SELECT Sno FROM S WHERE sdept = 'CS'

);



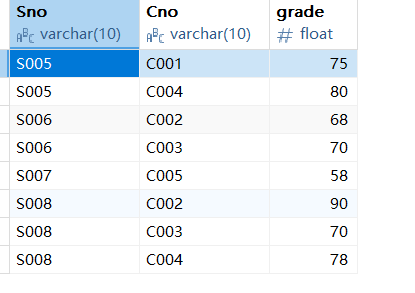
1. 删除“CS”系全体学生的选课记录

DELETE FROM SC

WHERE Sno IN (

SELECT Sno FROM S WHERE sdept = 'CS'

);



1. 删除学号为“S1”的相关信息

DELETE FROM SC WHERE Sno = 'S1';

DELETE FROM S WHERE Sno = 'S1';

1. 将学号为“S1”的学生的学号修改为“S001”

UPDATE S

SET Sno = 'S001'

WHERE Sno = 'S1';

1. 把平均成绩大于80分的男同学的学号和平均成绩存入另一个表S——GRADE（SNO，AVG\_GRADE）

CREATE TABLE S\_GRADE (

Sno VARCHAR(20),

AVG\_GRADE FLOAT

);

INSERT INTO S\_GRADE (Sno, AVG\_GRADE)

SELECT S.Sno, AVG(SC.grade) AS avg\_grade

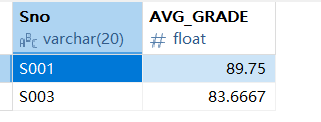
FROM S

JOIN SC ON S.Sno = SC.Sno

WHERE S.Ssex = 'M'

GROUP BY S.Sno

HAVING avg\_grade > 80;



1. 把选修了课程名为“数据结构”的学生的成绩提高10%

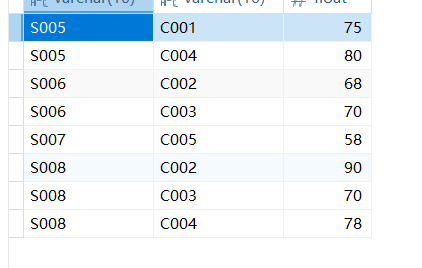
UPDATE SC

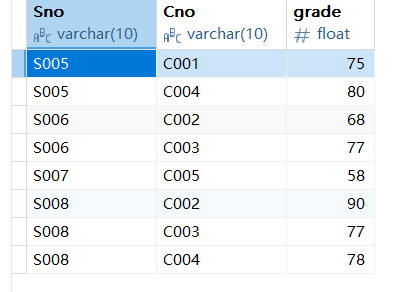
SET grade = grade \* 1.1

WHERE Cno = (

SELECT Cno FROM C WHERE Cname = '数据结构'

);





1. 把选修了“C2”号课程，且成绩低于该门课程的平均成绩的学生成绩删除掉

DELETE FROM SC

WHERE Cno = 'C002'

AND grade < (

SELECT avg\_grade

FROM (

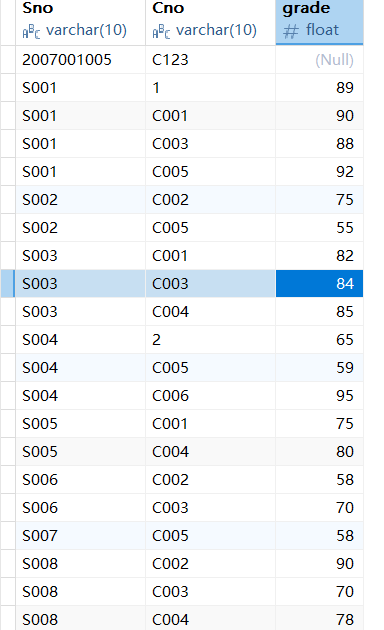
SELECT AVG(grade) AS avg\_grade

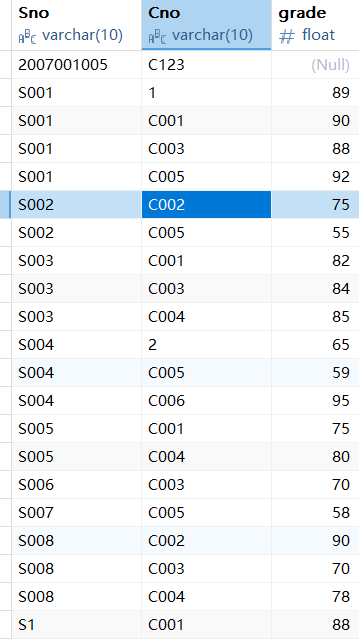
FROM SC

WHERE Cno = 'C002'

) AS temp

);





**五、实验小结**

通过本次学生选课系统的数据库实验，我对 SQL 语言的使用有了更深入的理解，尤其是在查询（SELECT）、插入（INSERT）、更新（UPDATE）、删除（DELETE）等基本操作方面得到了系统的练习。掌握了常见的聚合函数（如 COUNT()、AVG()、MAX() 等）以及分组（GROUP BY）、排序（ORDER BY）、连接查询（JOIN）等复杂语句的写法，并能根据实际需求灵活组合使用。

实验中遇到的问题：

问题：插入数据时出现了外键约束错误

解决：通过仔细检查表之间的主外键关系，确保先插入父表数据，后插入子表数据，最终成功解决

问题：删除数据时遇到了 MySQL 报错

解决：通过引入子查询，即在 DELETE 语句中套一层 SELECT 子查询并起别名，有效解决了该问题

问题：插入和查询过程中数据的一致性和完整性问题

解决：保证 Sno、Cno 在各表中的存在性，避免因数据缺失导致查询结果为空。

通过这些问题的解决，我进一步理解了数据库完整性约束的重要性，同时也提升了独立分析和解决问题的能力。当然我也存在很多不足之处，比如在设计初期对数据内容覆盖的考虑还不够周全，导致部分查询出现无结果的情况，今后在插入数据阶段应该更加细致规划，确保各项测试查询都能有实际数据支撑。

总体来说，本次实验不仅加深了我对数据库基本操作的理解，还提高了我的SQL编程能力和排错能力，为后续学习更加复杂的数据库应用打下了坚实基础。