实验报告

实验（二）

姓　　名	熊恪峥
学　　号	22920202204622
日　　期	2023年4月3日
学　　院	信息学院
课程名称	数据库

实验（二）

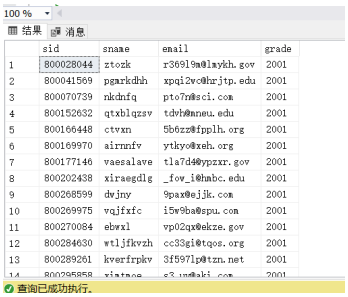
目录

1 实验2.1

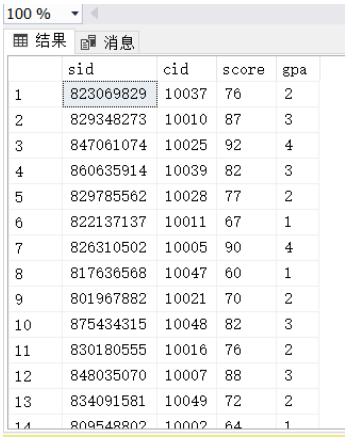
1. 查询年级为2001的所有学生的名称并按编号升序排列。

```
select * from STUDENTS where grade=2001 order by sid asc
```

图 1: 运行结果



	sid	sname	email	grade
1	800028044	ztozk	r36919m@lmykh.gov	2001
2	800041569	pgarkdhh	xpqi2vc@hrjtp.edu	2001
3	800070739	nkdnfq	pto7n@sci.com	2001
4	800152632	qtxblqzsv	tdvh@mneu.edu	2001
5	800166448	ctvxn	5b6zz@fpplh.org	2001
6	800169970	airnnfv	ytkyo@xeh.org	2001
7	800177146	vaesalave	tla7d4@ypzxr.gov	2001
8	800202438	xirasegdlg	_fov_i@hmbc.edu	2001
9	800268599	dvjny	9pax@ejjk.com	2001
10	800269975	vqjfxfc	i5w9ba@spu.com	2001
11	800270084	ebwxl	vp02qx@ekze.gov	2001
12	800284630	wtljfkvzh	cc33gi@tqos.org	2001
13	800289261	kverfrpkv	3f597lp@tzn.net	2001

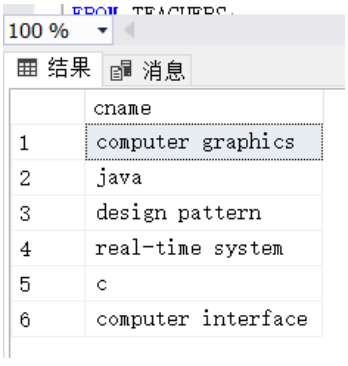


	sid	cid	score	gpa
1	823069829	10037	76	2
2	829348273	10010	87	3
3	847061074	10025	92	4
4	860635914	10039	82	3
5	829785562	10028	77	2
6	822137137	10011	67	1
7	826310502	10005	90	4
8	817636568	10047	60	1
9	801967882	10021	70	2
10	875434315	10048	82	3
11	830180555	10016	76	2
12	848035070	10007	88	3
13	834091581	10049	72	2

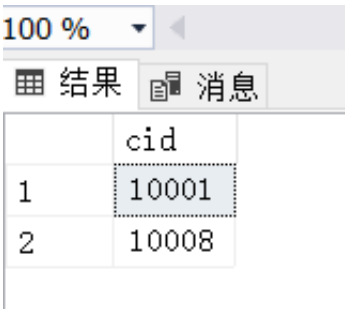
2. 查询学生的选课成绩合格的课程成绩，并把成绩换算为积点（60分对应积点为1，每增加1分，积点增加0.1）。

```
SELECT sid, cid, score, CASE WHEN score >= 60 THEN (score - 50) / 10 ELSE 0 END
AS gpa FROM CHOICES WHERE score >= 60;
```

图 2: 运行结果



	cname
1	computer graphics
2	java
3	design pattern
4	real-time system
5	c
6	computer interface



	cid
1	10001
2	10008

3. 查询课时是48或64的课程的名称。

```
SELECT cname FROM COURSES WHERE hour IN (48, 64);
```

4. 查询所有课程名称中含有data的课程编号。

```
SELECT cid FROM COURSES WHERE cname LIKE ’
```

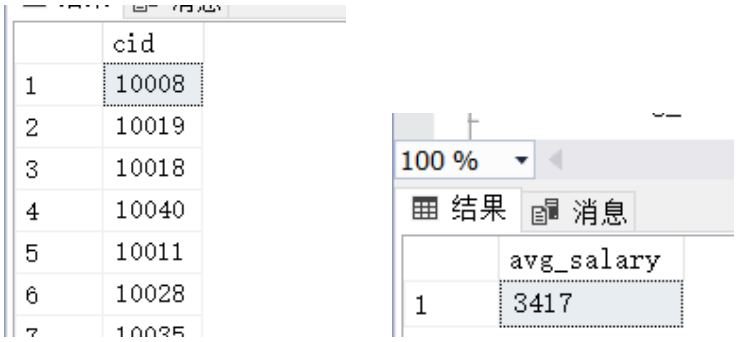
5. 查询所有选课记录的课程号（不重复显示）。

```
SELECT DISTINCT cid FROM CHOICES;
```

6. 统计所有教师的平均工资。

```
SELECT AVG(salary) AS avg_salary FROM TEACHERS;
```

图 3: 运行结果

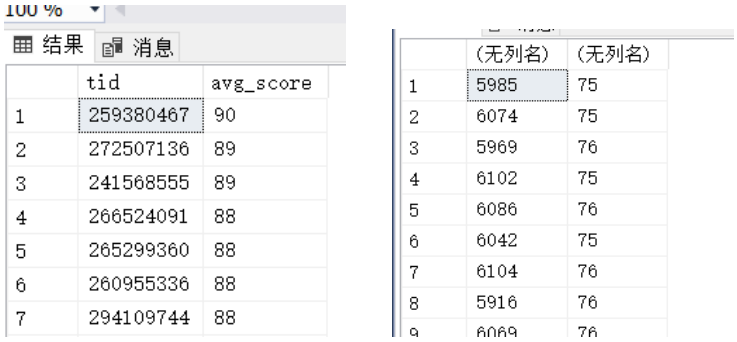


	cid
1	10008
2	10019
3	10018
4	10040
5	10011
6	10028

100 % 结果 消息

	avg_salary
1	3417

图 4: 运行结果



100 % 结果 消息

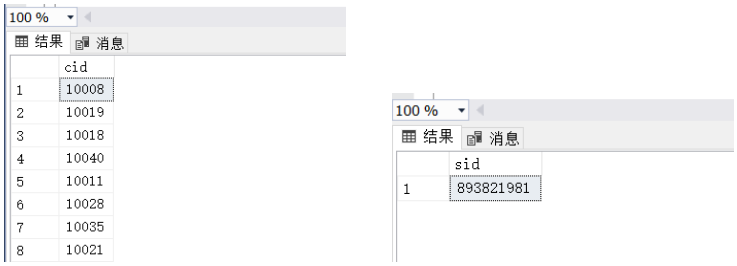
	tid	avg_score
1	259380467	90
2	272507136	89
3	241568555	89
4	266524091	88
5	265299360	88
6	260955336	88
7	294109744	88

	(无列名)	(无列名)
1	5985	75
2	6074	75
3	5969	76
4	6102	75
5	6086	76
6	6042	75
7	6104	76
8	5916	76

7. 查询所有教师的编号及选修其课程的学生的平均成绩，按平均成绩降序排列。

```
SELECT TEACHERS.tid, AVG(CHOICES.score) AS avg_score FROM TEACHERS JOIN CHOICES
ON TEACHERS.tid = CHOICES.tid JOIN STUDENTS ON STUDENTS.sid = CHOICES.sid GROUP
BY TEACHERS.tid ORDER BY avg_score desc
```

图 5: 运行结果



100 % 结果 消息

	cid
1	10008
2	10019
3	10018
4	10040
5	10011
6	10028
7	10035
8	10021

100 % 结果 消息

	sid
1	893821981

8. 统计各个课程的选课人数和平均成绩。

```
SELECT COUNT(*),AVG(CHOICES.score) from CHOICES GROUP BY cid
```

9. 查询至少选修了三门课程的学生编号。

```
SELECT sid from CHOICES GROUP BY sid HAVING COUNT(*)>=3 select * from CHOICES
where sid=812917218
```

10. 查询编号800009026的学生所选的全部课程的课程名和成绩。

```
SELECT cname,score FROM CHOICES JOIN COURSES ON CHOICES.cid = COURSES.cid where
sid = 800009026
```

11. 查询所有选修了database的学生的编号。

图 6: 运行结果

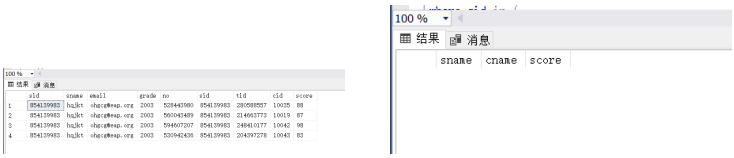
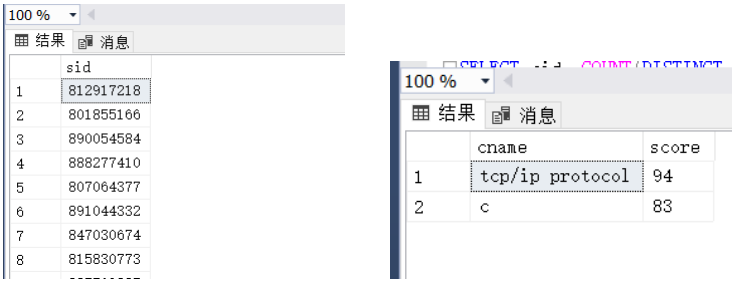
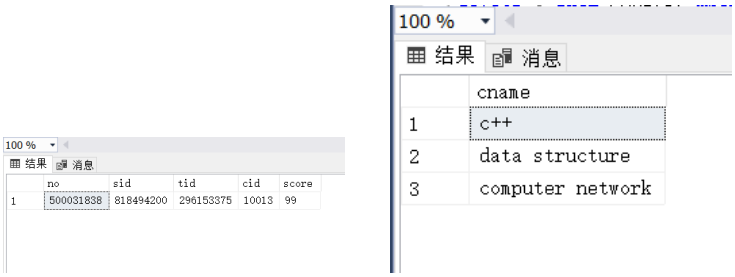


图 7: 运行结果




```
SELECT sid FROM CHOICES where cid in (select cid from COURSES where cname like
'database') select * from CHOICES where sid=870899566 select * from COURSES
```

图 8: 运行结果



12. 求出选择了同一个课程的学生数。

```
SELECT cid, COUNT(DISTINCT sid) AS num_of_students FROM CHOICES GROUP BY cid;
```

```
SELECT COUNT(DISTINCT c1.sid) AS num_of_students, c1.cid FROM CHOICES c1 INNER
JOIN CHOICES c2 ON c1.cid = c2.cid AND c1.sid <> c2.sid GROUP BY c1.cid;
```

13. 求出至少被两名学生选修的课程编号。

```
SELECT cid FROM CHOICES GROUP BY cid HAVING COUNT(*)>=2 select * from CHOICES
where cid=10008
```

14. 查询选修了编号800009026的学生所选的某个课程的学生编号。

```
select top(1) sid from CHOICES where cid in ( select cid from CHOICES where sid
= 800009026 ) order by NEWID()
```

15. 查询学生的基本信息及选修课程编号和成绩。

```
select * from STUDENTS JOIN CHOICES
ON STUDENTS.sid = CHOICES.sid where STUDENTS.sid=854139983
```

16. 查询学号850955252的学生的姓名和选修的课程名及成绩。

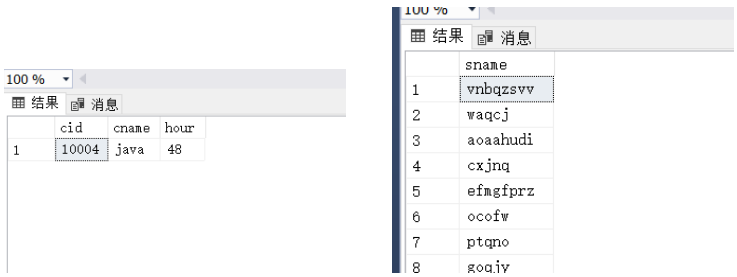
图 9: 运行结果


```
SELECT s.sname, c.cname, ch.score FROM STUDENTS s, CHOICES ch, COURSES c WHERE
s.sid = ch.sid AND ch.cid = c.cid AND s.sid = '850955252';
```

17. 查询与学号850955252的学生同年级的所有学生资料。

```
SELECT * FROM STUDENTS where grade in (SELECT grade from STUDENTS where sid='850955252')
```

图 10: 运行结果



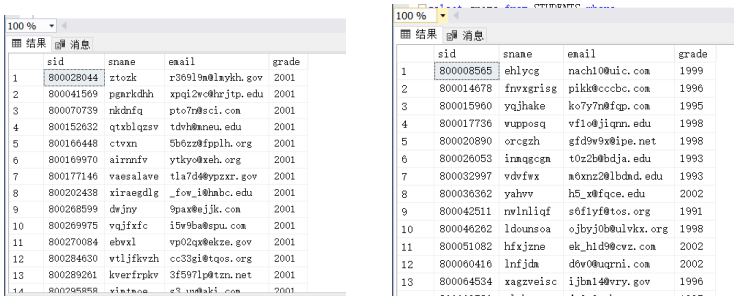
18. 查询所有有选课的学生的详细信息。

```
select * from STUDENTS where sid in ( select distinct sid from CHOICES )
```

19. 查询没有学生选的课程的编号。

```
select * from COURSES where cid not in ( select distinct cid from CHOICES )
```

图 11: 运行结果



20. 查询选修了与C++的课时一样课程的学生名称。

```
select sname from STUDENTS where sid in ( select distinct sid from CHOICES where
cid in ( select distinct cid from COURSES where hour in ( select hour from COURSES
where cname like 'c++' ) ) )
```

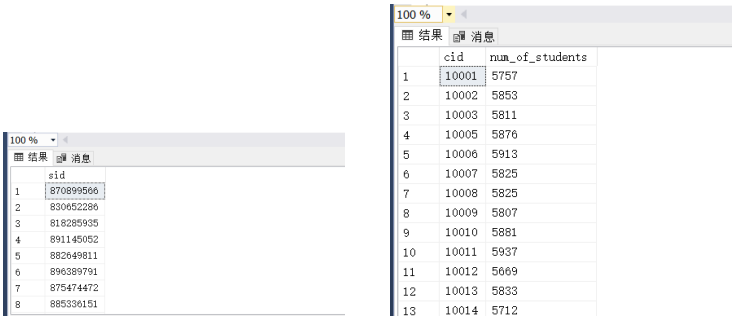
21. 找出选修课程成绩最好的选课记录。

```
select top (1) * from CHOICES order by score DESC
```

22. 找出和课程UML或课程C++的课时一样课程名称。

```
select cname from COURSES where hour in ( select hour from COURSES where cname
like 'C++' or cname like 'UML' )
```

图 12: 运行结果



	sid
1	870899566
2	830652286
3	818285935
4	891145052
5	882649811
6	896389791
7	875474472
8	885336151

	cid	num_of_students
1	10001	5757
2	10002	5853
3	10003	5811
4	10005	5876
5	10006	5913
6	10007	5825
7	10008	5825
8	10009	5807
9	10010	5881
10	10011	5937
11	10012	5669
12	10013	5833
13	10014	5712

23. 查询所有选修编号10001的课程的学生的姓名。

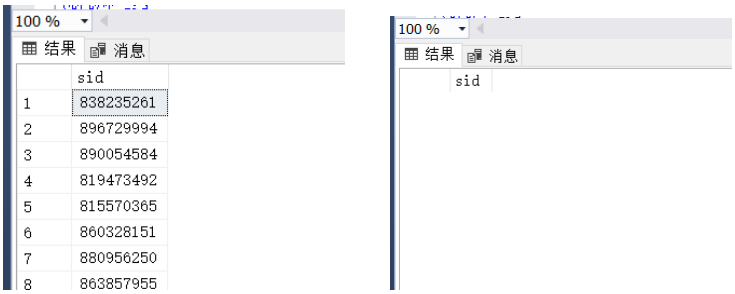
```
select sname from STUDENTS where sid in( select distinct sid from CHOICES where
cid = 10001 )
```

24. 查询选修了所有课程的学生姓名。

```
SELECT sid from CHOICES where not exists( select * from COURSES where cid not
in (select CHOICES.cid from CHOICES where CHOICES.sid = sid) )
```

```
SELECT sname FROM STUDENTS WHERE sid IN ( SELECT sid FROM CHOICES GROUP BY sid
HAVING COUNT(DISTINCT cid) = ( SELECT COUNT(*) FROM COURSES ) );
```

图 13: 运行结果



	sid
1	838235261
2	896729994
3	890054584
4	819473492
5	815570365
6	860328151
7	880956250
8	863857955

sid

25. 利用集合运算，查询选修课程C++或选修课程Java的学生的编号。

```
SELECT sid FROM CHOICES WHERE cid = (SELECT cid FROM COURSES WHERE cname = 'C++')
UNION SELECT sid FROM CHOICES WHERE cid = (SELECT cid FROM COURSES WHERE cname
= 'Java');
```

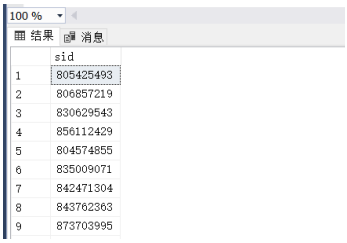
26. 实现集合交运算，查询既选修课程C++又选修课程Java的学生的编号。

```
SELECT sid FROM CHOICES WHERE cid = (SELECT cid FROM COURSES WHERE cname = 'C++')
INTERSECT SELECT sid FROM CHOICES WHERE cid = (SELECT cid FROM COURSES WHERE cname
= 'Java');
```

27. 实现集合减运算，查询选修课程C++而没有选修课程Java的学生的编号。

```
SELECT sid FROM CHOICES WHERE cid = (SELECT cid FROM COURSES WHERE cname = 'C++')
EXCEPT SELECT sid FROM CHOICES WHERE cid = (SELECT cid FROM COURSES WHERE cname
= 'Java');
```

图 14: 运行结果



2 实验2.2

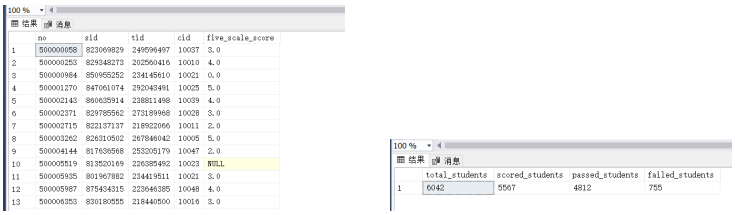
1. 查询所有选课记录的成绩并将它换算为五分制（满分5分，合格3分），注意SCORE取NULL值的情况。

```
SELECT no, sid, tid, cid, CASE WHEN score IS NULL THEN NULL WHEN score >= 60 THEN
(score-60)/10+2 ELSE 0.0 END AS gpa FROM CHOICES;
```

2. 通过查询选修编号10028的课程的学生的人数，其中成绩合格的学生人数，不合格的学生人数，讨论NULL值的特殊含义。

```
SELECT COUNT(*) AS total_students, SUM(CASE WHEN score IS NULL THEN 0 ELSE 1 END)
AS scored_students, SUM(CASE WHEN score >= 60 THEN 1 ELSE 0 END) AS passed_students,
SUM(CASE WHEN score >= 60 THEN 0 WHEN score IS NULL THEN 0 ELSE 1 END) AS failed_students
FROM CHOICES WHERE cid = '10028';
```

图 15: 运行结果



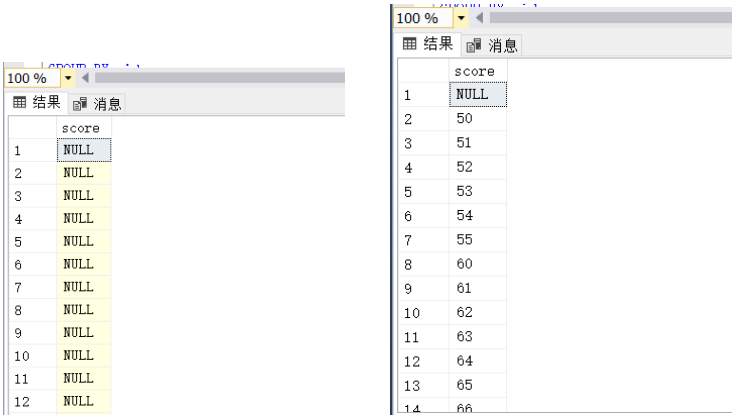
3. 通过实验检验在使用ORDER BY进行排序时，取NULL的项是否出现在结果中？如果有，在什么位置？

```
SELECT score FROM CHOICES ORDER BY score ASC;
```

4. 在上面的查询过程中如果加上保留字DISTINCT会有什么效果？

```
SELECT DISTINCT score FROM CHOICES ORDER BY score ASC;
```

图 16: 运行结果



5. 通过实验说明使用分组GROUP BY对取值为NULL的项的处理。

```
SELECT score, COUNT(*) FROM CHOICES GROUP BY score;
```

6. 结合分组，使用集合函数求每个同学的平均分、总的选课记录数、最高成绩、最低成绩和总成绩。

```
SELECT sid, AVG(score) AS avg_score, COUNT(*) AS total_records, MAX(score) AS
max_score, MIN(score) AS min_score, SUM(score) AS sum_score FROM CHOICES GROUP BY
sid;
```

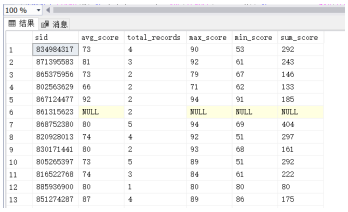
```
SELECT sid, AVG(ISNULL(score, 0)) AS avg_score, COUNT(*) AS total_records, MAX(ISNULL(score, 0)) AS max_score, MIN(ISNULL(score, 0)) AS min_score, SUM(ISNULL(score, 0)) AS sum_score FROM CHOICES GROUP BY sid;
```

图 17: 运行结果

100 %

结果 消息

	score	(无列名)
23	90	5898
24	96	5907
25	65	5912
26	79	5734
27	73	5771
28	85	5847
29	62	5765
30	99	5839
31	76	5877
32	NULL	23441
33	82	5915
34	88	5848
35	53	5788
36	71	5881



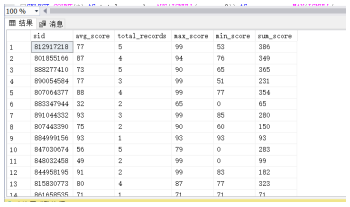
7. 查询成绩小于60的选课记录，统计总数、平均分、最大值和最小值。

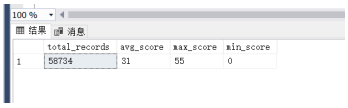
```
SELECT COUNT(*) AS total_records, AVG(ISNULL(score, 0)) AS avg_score, MAX(ISNULL(score, 0)) AS max_score, MIN(ISNULL(score, 0)) AS min_score FROM CHOICES WHERE ISNULL(score, 0) < 60;
```

8. 采用嵌套查询的方式，利用比较运算符和谓词ALL的结合来查询表COURSES中最少的课时。假设数据库中只有一个记录的时候，使用前面的方法会得到什么结果，为什么？

```
SELECT MIN(hour) FROM COURSES WHERE hour <= ALL (SELECT hour FROM COURSES WHERE hour > 0);
```

图 18: 运行结果





9. 创建一个学生表S(NO, SID, SNAME), 教师表T(NO, TID, TNAME)作为实验用的表。其中NO分别是这两个表的主键, 其他键允许为空。向S插入元组(1, 0129871001, 王小明)、(2, 0129871002, 李兰)、(3, 0129871005, NULL)、(4, 0129871004, 关红)；向T插入元组(1, 100189, 王小明)、(2, 100180, 李小)、(3, 100121, NULL)、(4, 100128, NULL)。对这两个表作对姓名的等值连接运算, 找出既是老师又是学生的人员的学生编号和老师编号。

```
CREATE TABLE S ( NO INT PRIMARY KEY, SID CHAR(10), SNAME VARCHAR(20) );
CREATE TABLE T ( NO INT PRIMARY KEY, TID CHAR(10), TNAME VARCHAR(20) );

INSERT INTO S(NO, SID, SNAME) VALUES (1, '0129871001', '王小明'), (2, '0129871002', '李兰'), (3, '0129871005', NULL), (4, '0129871004', '关红');
INSERT INTO T(NO, TID, TNAME) VALUES (1, '100189', '王小明'), (2, '100180', '李
```

```
小'), (3, '100121', NULL), (4, '100128', NULL);
SELECT * FROM S
SELECT * FROM T

 SELECT S.SID, T.TID FROM S JOIN T ON S.SNAME = T.TNAME WHERE S.SNAME IS NOT NULL
AND T.TNAME IS NOT NULL;
```

图 19: 运行结果