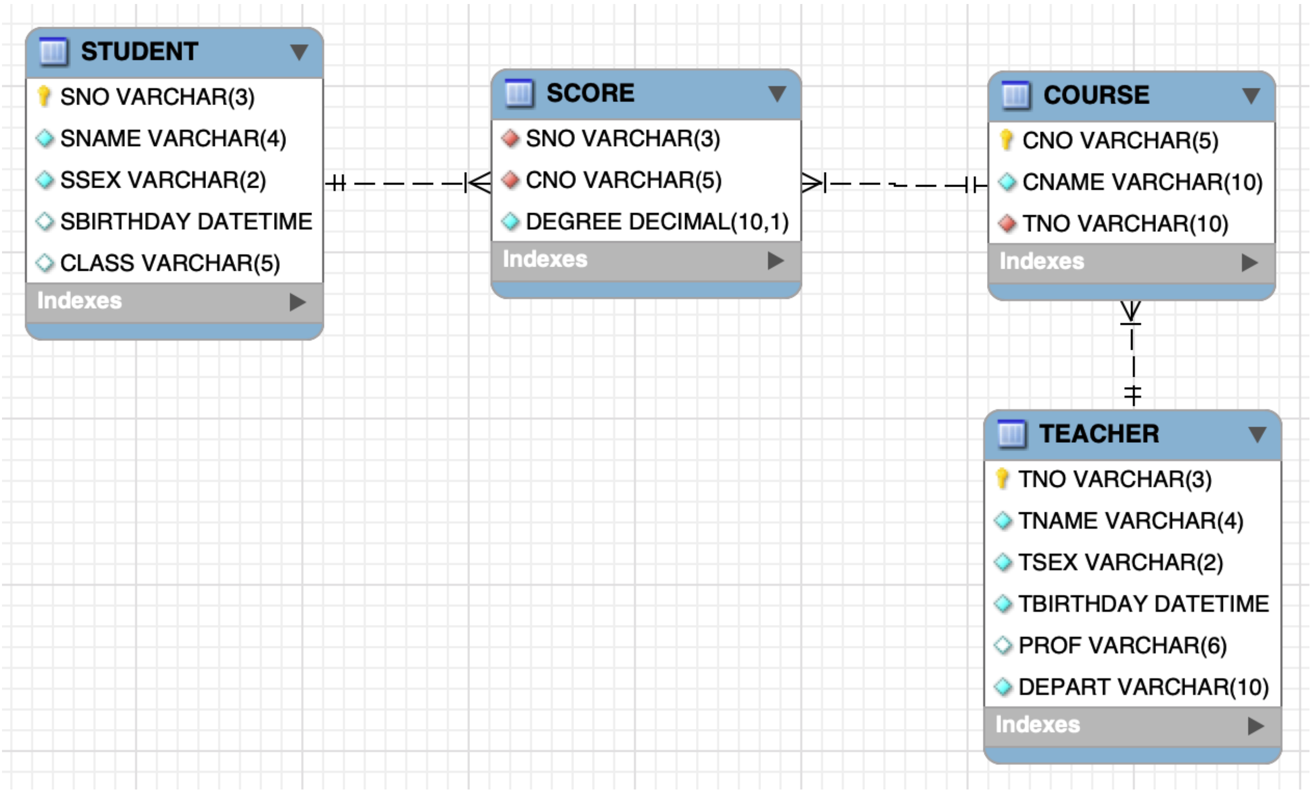
SQL语言 - SQL语句练习

构建如下表结构

还有一个Grade表，在如下的练习中体现



插入数据

> 下面表*SQL*和相关测试数据是我*Dump*出来的

```
-- MySQL dump 10.13  Distrib 5.7.17, for macos10.12 (x86_64)
--
-- Host: localhost    Database: learn_sql_pdai_tech
-- ------------------------------------------------------
-- Server version   5.7.28

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
```

```
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO'
*/;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `COURSE`
--

DROP TABLE IF EXISTS `COURSE`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `COURSE` (
  `CNO` varchar(5) NOT NULL,
  `CNAME` varchar(10) NOT NULL,
  `TNO` varchar(10) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `COURSE`
--

LOCK TABLES `COURSE` WRITE;
/*!40000 ALTER TABLE `COURSE` DISABLE KEYS */;
INSERT INTO `COURSE` VALUES ('3-105','计算机导论','825'),('3-245','操作系
统','804'),('6-166','数据电路','856'),('9-888','高等数学','100');
/*!40000 ALTER TABLE `COURSE` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `SCORE`
--

DROP TABLE IF EXISTS `SCORE`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `SCORE` (
  `SNO` varchar(3) NOT NULL,
  `CNO` varchar(5) NOT NULL,
  `DEGREE` decimal(10,1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
```

```
-- Dumping data for table `SCORE`
--

LOCK TABLES `SCORE` WRITE;
/*!40000 ALTER TABLE `SCORE` DISABLE KEYS */;
INSERT INTO `SCORE` VALUES ('103','3-245',86.0),('105','3-245',75.0),
('109','3-245',68.0),('103','3-105',92.0),('105','3-105',88.0),
('109','3-105',76.0),('101','3-105',64.0),('107','3-105',91.0),
('101','6-166',85.0),('107','6-106',79.0),('108','3-105',78.0),
('108','6-166',81.0);
/*!40000 ALTER TABLE `SCORE` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `STUDENT`
--

DROP TABLE IF EXISTS `STUDENT`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `STUDENT` (
  `SNO` varchar(3) NOT NULL,
  `SNAME` varchar(4) NOT NULL,
  `SSEX` varchar(2) NOT NULL,
  `SBIRTHDAY` datetime DEFAULT NULL,
  `CLASS` varchar(5) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `STUDENT`
--

LOCK TABLES `STUDENT` WRITE;
/*!40000 ALTER TABLE `STUDENT` DISABLE KEYS */;
INSERT INTO `STUDENT` VALUES ('108','曾华','男','1977-09-01
00:00:00','95033'),('105','匡明','男','1975-10-02 00:00:00','95031'),
('107','王丽','女','1976-01-23 00:00:00','95033'),('101','李
军','男','1976-02-20 00:00:00','95033'),('109','王芳','女','1975-02-10
00:00:00','95031'),('103','陆君','男','1974-06-03 00:00:00','95031');
/*!40000 ALTER TABLE `STUDENT` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `TEACHER`
--

DROP TABLE IF EXISTS `TEACHER`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
```

```
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `TEACHER` (
  `TNO` varchar(3) NOT NULL,
  `TNAME` varchar(4) NOT NULL,
  `TSEX` varchar(2) NOT NULL,
  `TBIRTHDAY` datetime NOT NULL,
  `PROF` varchar(6) DEFAULT NULL,
  `DEPART` varchar(10) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `TEACHER`
--

LOCK TABLES `TEACHER` WRITE;
/*!40000 ALTER TABLE `TEACHER` DISABLE KEYS */;
INSERT INTO `TEACHER` VALUES ('804','李诚','男','1958-12-02 00:00:00','副教授','计算机系'),('856','张旭','男','1969-03-12 00:00:00','讲师','电子工程系'),('825','王萍','女','1972-05-05 00:00:00','助教','计算机系'),('831','刘冰','女','1977-08-14 00:00:00','助教','电子工程系');
/*!40000 ALTER TABLE `TEACHER` ENABLE KEYS */;
UNLOCK TABLES;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2020-02-06 18:18:25
```

相关练习

- 1、 查询Student表中的所有记录的Sname、Ssex和Class列。

```
select SNAME, SSEX, CLASS from STUDENT;
```

- 2、 查询教师所有的单位即不重复的Depart列。

```
select distinct DEPART from TEACHER;
```

- 3、 查询Student表的所有记录。

```
select * from STUDENT;
```

- 4、 查询Score表中成绩在60到80之间的所有记录。

```
select *
from SCORE
where DEGREE > 60 and DEGREE < 80;
```

- 5、 查询Score表中成绩为85，86或88的记录。

```
select *
from SCORE
where DEGREE = 85 or DEGREE = 86 or DEGREE = 88;
```

- 6、 查询Student表中“95031”班或性别为“女”的同学记录。

```
select *
from STUDENT
where CLASS = '95031' or SSEX = '女';
```

- 7、 以Class降序查询Student表的所有记录。

```
select *
from STUDENT
order by CLASS desc;
```

- 8、 以Cno升序、Degree降序查询Score表的所有记录。

```
select *
from SCORE
order by CNO asc, DEGREE desc;
```

- 9、 查询“95031”班的学生人数。

```
select count(*)
from STUDENT
where CLASS = '95031';
```

- 10、查询Score表中的最高分的学生学号和课程号。

```
select
  sno,
  CNO
from SCORE
where DEGREE = (
  select max(DEGREE)
  from SCORE
);
```

- 11、查询‘3-105’号课程的平均分。

```
select avg(DEGREE)
from SCORE
where CNO = '3-105';
```

- 12、查询Score表中至少有5名学生选修的并以3开头的课程的平均分数。

```
select
  avg(DEGREE),
  CNO
from SCORE
where cno like '3%'
group by CNO
having count(*) > 5;
```

- 13、查询最低分大于70，最高分小于90的Sno列。

```
select SNO
from SCORE
group by SNO
having min(DEGREE) > 70 and max(DEGREE) < 90;
```

- 14、查询所有学生的Sname、Cno和Degree列。

```
select
  SNAME,
  CNO,
  DEGREE
from STUDENT, SCORE
where STUDENT.SNO = SCORE.SNO;
```

- 15、查询所有学生的Sno、Cname和Degree列。

```
select
  SCORE.SNO,
  CNO,
  DEGREE
from STUDENT, SCORE
where STUDENT.SNO = SCORE.SNO;
```

- 16、查询所有学生的Sname、Cname和Degree列。

```
SELECT
  A.SNAME,
  B.CNAME,
  C.DEGREE
FROM STUDENT A
  JOIN (COURSE B, SCORE C)
    ON A.SNO = C.SNO AND B.CNO = C.CNO;
```

- 17、查询“95033”班所选课程的平均分。

```
select avg(DEGREE)
from SCORE
where sno in (select SNO
              from STUDENT
              where CLASS = '95033');
```

- 18、假设使用如下命令建立了一个grade表:

```
create table grade (
  low  numeric(3, 0),
  upp  numeric(3),
  rank char(1)
);
insert into grade values (90, 100, 'A');
insert into grade values (80, 89, 'B');
insert into grade values (70, 79, 'C');
insert into grade values (60, 69, 'D');
insert into grade values (0, 59, 'E');
```

- 现查询所有同学的Sno、Cno和rank列。

```
SELECT
  A.SNO,
  A.CNO,
  B.RANK
FROM SCORE A, grade B
WHERE A.DEGREE BETWEEN B.LOW AND B.UPP
ORDER BY RANK;
```

- 19、查询选修“3-105”课程的成绩高于“109”号同学成绩的所有同学的记录。

```
select *
from SCORE
where CNO = '3-105' and DEGREE > ALL (
  select DEGREE
  from SCORE
  where SNO = '109'
);
```

- 20、查询score中选学一门以上课程的同学中分数为非最高分成绩的学生记录

```
select * from STUDENT where SNO
  in (select SNO
  from SCORE
  where DEGREE < (select MAX(DEGREE) from SCORE)
  group by SNO
  having count(*) > 1);
```

- 21、查询成绩高于学号为“109”、课程号为“3-105”的成绩的所有记录。

```
select *
from SCORE
where CNO = '3-105' and DEGREE > ALL (
  select DEGREE
  from SCORE
  where SNO = '109'
);
```

- 22、查询和学号为108的同学同年出生的所有学生的Sno、Sname和Sbirthday列。

```
select
  SNO,
  SNAME,
  SBIRTHDAY
from STUDENT
where year(SBIRTHDAY) = (
  select year(SBIRTHDAY)
  from STUDENT
  where SNO = '108'
);
```

- 23、查询“张旭“教师任课的学生成绩。

```
select *
from SCORE
where cno = (
  select CNO
  from COURSE
    inner join TEACHER on COURSE.TNO = TEACHER.TNO and TNAME = '张旭'
);
```

- 24、查询选修某课程的同学人数多于5人的教师姓名。

```
select TNAME
from TEACHER
where TNO = (
  select TNO
  from COURSE
  where CNO = (select CNO
               from SCORE
               group by CNO
               having count(SNO) > 5)
);
```

- 25、查询95033班和95031班全体学生的记录。

```
select *
from STUDENT
where CLASS in ('95033', '95031');
```

- 26、查询存在有85分以上成绩的课程Cno.

```
select cno
from SCORE
group by CNO
having MAX(DEGREE) > 85;
```

- 27、查询出"计算机系"教师所教课程的成绩表。

```
select *
from SCORE
where CNO in (select CNO
              from TEACHER, COURSE
              where DEPART = '计算机系' and COURSE.TNO = TEACHER.TNO);
```

- 28、查询"计算机系"与"电子工程系"不同职称的教师的Tname和Prof

```
select
  tname,
  prof
from TEACHER
where depart = '计算机系' and prof not in (
  select prof
  from TEACHER
  where depart = '电子工程系'
);
```

- 29、查询选修编号为“3-105“课程且成绩至少高于选修编号为“3-245”的同学的Cno、Sno和Degree,并按Degree从高到低次序排序。

```
select
  CNO,
  SNO,
  DEGREE
from SCORE
where CNO = '3-105' and DEGREE > any (
  select DEGREE
  from SCORE
  where CNO = '3-245'
)
order by DEGREE desc;
```

- 30、查询选修编号为“3-105”且成绩高于选修编号为“3-245”课程的同学的Cno、Sno和Degree.

```
SELECT *
FROM SCORE
WHERE DEGREE > ALL (
  SELECT DEGREE
  FROM SCORE
  WHERE CNO = '3-245'
)
ORDER by DEGREE desc;
```

- 31、查询所有教师和同学的name、sex和birthday.

```
select
  TNAME     name,
  TSEX      sex,
  TBIRTHDAY birthday
from TEACHER
union
select
  sname     name,
  SSEX      sex,
  SBIRTHDAY birthday
from STUDENT;
```

- 32、查询所有“女”教师和“女”同学的name、sex和birthday.

```
select
  TNAME     name,
  TSEX      sex,
  TBIRTHDAY birthday
from TEACHER
where TSEX = '女'
union
select
  sname     name,
  SSEX      sex,
  SBIRTHDAY birthday
from STUDENT
where SSEX = '女';
```

- 33、查询成绩比该课程平均成绩低的同学的成绩表。

```
SELECT A.*
FROM SCORE A
WHERE DEGREE < (SELECT AVG(DEGREE)
                FROM SCORE B
                WHERE A.CNO = B.CNO);
```

- 34、查询所有任课教师的Tname和Depart.

```
select
  TNAME,
  DEPART
from TEACHER a
where exists(select *
             from COURSE b
             where a.TNO = b.TNO);
```

- 35、查询所有未讲课的教师的Tname和Depart.

```
select
  TNAME,
  DEPART
from TEACHER a
where tno not in (select tno
                  from COURSE);
```

- 36、查询至少有2名男生的班号。

```
select CLASS
from STUDENT
where SSEX = '男'
group by CLASS
having count(SSEX) > 1;
```

- 37、查询Student表中不姓“王”的同学记录。

```
select *
from STUDENT
where SNAME not like "王%";
```

- 38、查询Student表中每个学生的姓名和年龄。

```
select
  SNAME,
  year(now()) - year(SBIRTHDAY)
from STUDENT;
```

- 39、查询Student表中最大和最小的Sbirthday日期值。

```
select min(SBIRTHDAY) birthday
from STUDENT
union
select max(SBIRTHDAY) birthday
from STUDENT;
```

- 40、以班号和年龄从大到小的顺序查询Student表中的全部记录。

```
select *
from STUDENT
order by CLASS desc, year(now()) - year(SBIRTHDAY) desc;
```

- 41、查询“男”教师及其所上的课程。

```
select *
from TEACHER, COURSE
where TSEX = '男' and COURSE.TNO = TEACHER.TNO;
```

- 42、查询最高分同学的Sno、Cno和Degree列。

```
select
  sno,
  CNO,
  DEGREE
from SCORE
where DEGREE = (select max(DEGREE)
                from SCORE);
```

- 43、查询和“李军”同性别的所有同学的Sname.

```
select sname
from STUDENT
where SSEX = (select SSEX
              from STUDENT
              where SNAME = '李军');
```

- 44、查询和“李军”同性别并同班的同学Sname.

```
select sname
from STUDENT
where (SSEX, CLASS) = (select
                         SSEX,
                         CLASS
                       from STUDENT
                       where SNAME = '李军');
```

- 45、查询所有选修“计算机导论”课程的“男”同学的成绩表

```
select *
from SCORE, STUDENT
where SCORE.SNO = STUDENT.SNO and SSEX = '男' and CNO = (
  select CNO
  from COURSE
  where CNAME = '计算机导论');
```

- 46、使用游标方式来同时查询每位同学的名字，他所选课程及成绩。

```
declare
 cursor student_cursor is
  select S.SNO,S.SNAME,C.CNAME,SC.DEGREE as DEGREE
  from STUDENT S, COURSE C, SCORE SC
  where S.SNO=SC.SNO
  and SC.CNO=C.CNO;

  student_row student_cursor%ROWTYPE;

begin
  open student_cursor;
   loop
    fetch student_cursor INTO student_row;
    exit when student_cursor%NOTFOUND;
     dbms_output.put_line( student_row.SNO || '' ||

student_row.SNAME|| '' || student_row.CNAME || '' ||

student_row.DEGREE);
   end loop;
  close student_cursor;
END;
/
```

- 47、 声明触发器指令，每当有同学转换班级时执行触发器显示当前和之前所在班级。

```
CREATE OR REPLACE TRIGGER display_class_changes
AFTER DELETE OR INSERT OR UPDATE ON student
FOR EACH ROW
WHEN (NEW.sno > 0)

BEGIN

   dbms_output.put_line('Old class: ' || :OLD.class);
   dbms_output.put_line('New class: ' || :NEW.class);
END;
/

Update student
set class=95031
where sno=109;
```

- 48、 删除已设置的触发器指令

```
DROP TRIGGER display_class_changes;
```