实验3 创建表及维护表

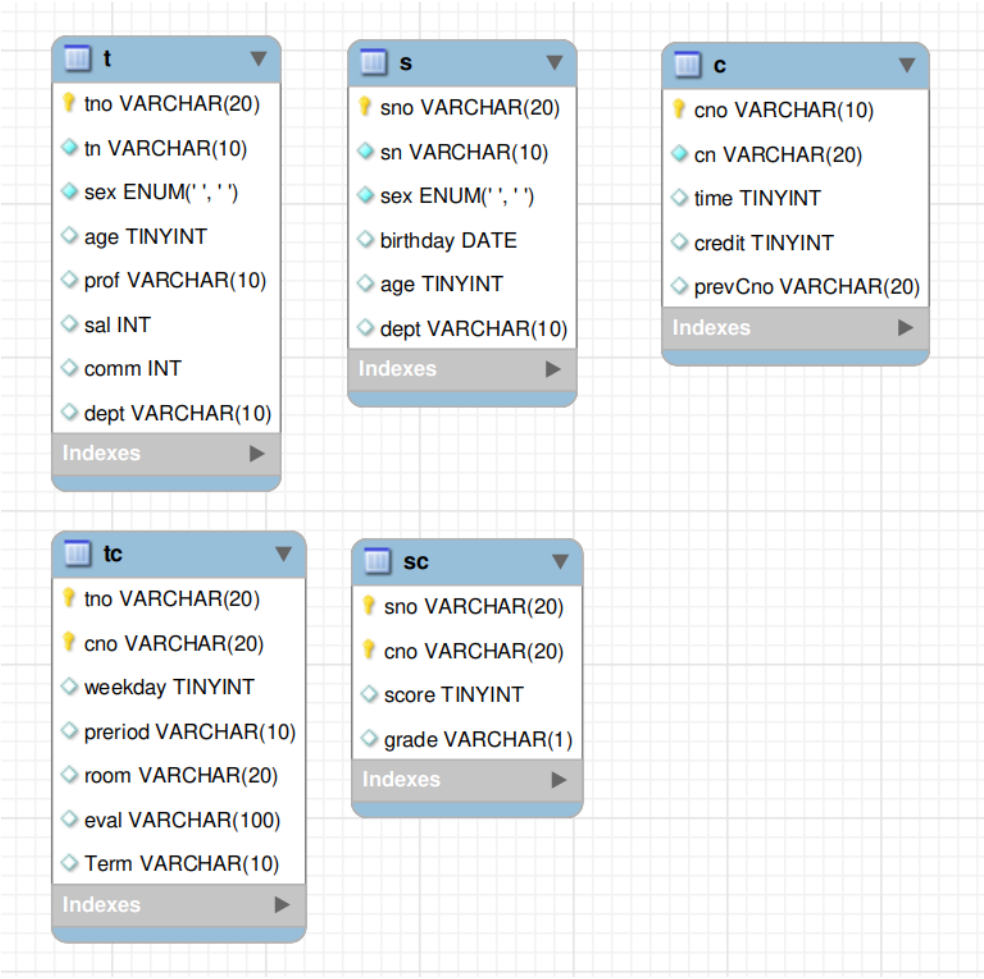
3.1 创表

> 在数据库 jxsk中创建如下数据表
>
> 教师表：t，学生表 s，课程表：c，选课表 sc，授课表：tc。
>
> 各数据表的结构如下：（其中中文为部分字段的解释，字段名一律使用字母，类型由你自定）t（tno教师号，tn姓名，Sex，age，prof职称，sal工资，comm岗位津贴，dept系名）s（sno学号，sn姓名，sex,dirthday,dept院系）c（cno课程号，cn课程名，time课时数，credit学分，prevCno先行课）
> sc（sno，cno，Score成绩）tc（tno，cno，weekday周几，preriod节次，room教室，Eval评价）

SQL语句:

```
USE jxsk;
-- 创建教师表
CREATE TABLE IF NOT EXISTS t(
    tno VARCHAR(20) PRIMARY KEY comment '教师编号 ',
    tn VARCHAR(10) NOT NULL comment '教师名',
    sex TINYINT(1) UNSIGNED NOT NULL comment '性别 0-女 1-男 Tinyint只占1个字节 性能更高 无符号修饰 0-255',
    age TINYINT comment 'tingint为0~255 符合年龄要求',
    prof VARCHAR(10) comment '教师职称',
    sal INT(7) comment '工资',
    comm INT(7) comment '岗位津贴',
    dept VARCHAR(10) comment '所在系名'
);
-- 创建学生表
CREATE TABLE IF NOT EXISTS s(
    sno VARCHAR(20) PRIMARY KEY comment '学号',
    sn VARCHAR(10) NOT NULL comment '学生名',
    sex TINYINT(1) UNSIGNED NOT NULL comment '性别 0-女 1-男 Tinyint只占1个字节 性能更高 无符号修饰 0-255',
    birthday DATE DEFAULT '2000-01-01' comment '出生日期',
    dept VARCHAR(10) comment '所在系名'
);
-- 创建课程表
CREATE TABLE IF NOT EXISTS c(
    cno VARCHAR(20) PRIMARY KEY COMMENT '课程号',
    cn VARCHAR(20) UNIQUE NOT NULL COMMENT '课程名',
    time TINYINT UNSIGNED COMMENT ' 课时数',
    credit TINYINT UNSIGNED COMMENT '学分',
    prevCno VARCHAR(20)
);
-- 创建学生选课表
CREATE TABLE IF NOT EXISTS sc(
    sno VARCHAR(20) COMMENT '学生的学号',
    cno VARCHAR(20) COMMENT '该学生选定课程号',
    score TINYINT UNSIGNED DEFAULT 0 COMMENT '该学生该门课的成绩',
    PRIMARY KEY (sno,cno) COMMENT '组合主键 不允许全部相同'
);
-- 创建教师授课表
CREATE TABLE IF NOT EXISTS tc(
    tno VARCHAR(20) COMMENT '教师编号',
    cno VARCHAR(20) COMMENT '课程编号',
    weekday TINYINT(1) UNSIGNED COMMENT '周几上课',
    preriod TINYINT(1) UNSIGNED COMMENT '节次',
    room VARCHAR(20)  COMMENT '教室',
    eval VARCHAR(100) COMMENT '评价',
    PRIMARY KEY (tno,cno)
);
```

运行后的ER图如下：



3.2 加列

> 向 s表中追加可以存放 20个汉字的学籍（native）列，类型 char(40)不允许为空。

- ### SQL语句:

```
ALTER TABLE s ADD COLUMN native VARCHAR(40) NOT NULL; -- 默认在最后一列添加一列
```

- **拓展**

```
ALTER TABLE s ADD COLUMN dna VARCHAR(40) NOT NULL AFTER sex; -- 设置在sex后添加一列
```

此时学生表 s 的结构为

查表结构

```
SHOW  FULL COLUMNS FROM s; -- 查询s的详细表结构
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

#	Field	Type	Collation	Null	Key	Default	Extra	Privileges	Comment
1	sno	varchar(20)	utf8mb4_0900_ai_ci	NO	PRI	NULL		select,insert,update,references	学号
2	sn	varchar(10)	utf8mb4_0900_ai_ci	NO		NULL		select,insert,update,references	学生名
3	sex	enum('男','女')	utf8mb4_0900_ai_ci	NO		NULL		select,insert,update,references	
4	dna	varchar(40)	utf8mb4_0900_ai_ci	NO		NULL		select,insert,update,references	
5	birthday	date	NULL	YES		2000-01-01		select,insert,update,references	出生日期
6	age	tinyint	NULL	YES		NULL		select,insert,update,references	
7	dept	varchar(10)	utf8mb4_0900_ai_ci	YES		NULL		select,insert,update,references	所在系名
8	native	varchar(40)	utf8mb4_0900_ai_ci	NO		NULL		select,insert,update,references	

3.3 改列

> 将 native列修改为 Unicode字符类型 nchar的列,宽度仍然是存放 20个汉字。

- ### SQL语句

```
ALTER TABLE s CHANGE COLUMN native native CHAR(40);-- 修改列
-- 拓展 修改t表的性别 改为枚举型 要么男 要么女
ALTER TABLE t CHANGE sex sex ENUM('男','女') NOT NULL;
```

3.4 删列

> 删除 native列

- ### SQL语句

```
ALTER TABLE s DROP native, DROP dna; -- 拓展 删除多列
```

此时的 s 表结构为：

Result Grid | Filter Rows: | Export: | Wrap Cell Content

#	Field	Type	Collation	Null	Key	Default
1	sno	varchar(20)	utf8mb4_0900_ai_ci	NO	PRI	NULL
2	sn	varchar(10)	utf8mb4_0900_ai_ci	NO		NULL
3	sex	enum('男','女')	utf8mb4_0900_ai_ci	NO		NULL
4	birthday	date	NULL	YES		2000-01-01
5	age	tinyint	NULL	YES		NULL
6	dept	varchar(10)	utf8mb4_0900_ai_ci	YES		NULL

3.5 插入数据

> 输入下面这些数据，再输入一些自定义数据
>
> 1)交互方式录入一些学生数据： s1，赵亦， 女，1995-01-01，计算机 s2，钱尔， 男，1996-01-10，信息 s3，张小明,男，1995-12-10，信息 s4，李思，男，1995-06-01，自动化 s5，周武， 男，1994-12-01，计算机
>
> 2)用 SQL命令录入一些教师数据： t5，张兰， 女， 39， 副教授,1300，2000， 信息 t4，张雪， 女， 51， 教授， 1600，3000， 自动化 t3，刘伟， 男， 30，讲师， 900， 1200， 计算机 t2，王平， 男， 28， 教授， 1900，2200， 信息 t1，李力， 男， 47， 教授， 1500，3000， 计算机
>
> 3)用命令录入一些课程数据： c1，程序设计，60，3，null c2，微机原理，60，3，c1 c3，数据库， 90，4，c1 c5，高等数学,80，4，null

- ### SQL语句

```
-- 向表s录入学生数据 交互式录入
-- 向表t录取教师数据 SQL
INSERT INTO t VALUES ('t5','张兰','女',39,'副教授',1300,2000,'信息'),
('t4','张雪','女',51,'教授',1600,3000,'自动化'),
('t3','刘伟','男',30,'讲师',900,1200,'计算机'),
('t2','王平','男',28,'教授',1900,2200,'信息'),
('t1','李力','男',47,'教授',1500,3000,'计算机');
-- 向表c录入课程数据 SQL
INSERT INTO c VALUES ('c1','程序设计',60,3,null),
('c2','微机原理',60,3,'c1'),
('c3','数据库',90,4,'c1'),
('c5','高等数学',80,4,null);
```

3.6复制表

> 用 SQL命令再创建一个学生表 stu，结构同 S表

- ### SQL语句

```
-- 复制表结构
SHOW CREATE TABLE s; -- 获取s表的创建语句
CREATE TABLE `stu` ( -- 将获取到的语句 更改表明为复制后的表名 stu
  `sno` varchar(20) NOT NULL COMMENT '学号',
  `sn` varchar(10) NOT NULL COMMENT '学生名',
  `sex` enum('男','女') NOT NULL,
  `birthday` date DEFAULT '2000-01-01' COMMENT '出生日期',
  `dept` varchar(10) DEFAULT NULL COMMENT '所在系名',
  PRIMARY KEY (`sno`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
SHOW CREATE TABLE s;
-- 此时表stu的结构同s一样
```

拓展：完全复制表 `s`

```
-- 1.如上步 先复制表结构位stu
-- 2.将表s的数据插入到stu中即可
INSERT INTO stu (SELECT * FROM s); -- 把s表的数据内容也复制到stu
```

3.7 删除表

> 用 SQL命令删除学生表 stu

- ### SQL语句

```
DROP TABLE stu; -- 删除stu
```

实验4 表及数据操作

4.1 更改表数据

> 用命令实现以下操作：
>
> 1)把学生“周武”的年龄改为 19，系别改为“信息”
>
> 2)将教师“王平”的职称改为“副教授”
>
> 3)删除你自已添加的一些数据行或删除周武和王平两行，注意：删除之前请先备份，以便出错后恢复。

- ### SQL语句

```
-- 4.1 修改表内数据 UPDATE 表名 SET 列名1=新值1，列名2=新值2，.. WHERE 筛选条件
SET SQL_SAFE_UPDATES = 0; -- 关闭安全更新模式 确保关闭 才能进行修改
```

4.1.1 把学生“周武”的年龄改为 19，系别改为“信息”

```
UPDATE s SET birthday = '2002-12-01',dept = '信息' WHERE sn='周武';
```

4.1.2 将教师“王平”的职称改为“副教授"

```
UPDATE t SET prof = '副教授' WHERE tn = '王平';
```

4.1.3 除表中数据 DELECT FROM 表 WHERE 条件

```
-- 一旦删除数据，它就会永远消失。因此，在执行DELETE语句之前，应该先备份数据库，以防万一要找回删除过的数据。
-- zeroac@zeroUbuntu:~$ mysqldump -u exam -p jxsk > ./Desktop/examDatabase/before04.sql
DELETE FROM s WHERE sn = '周武';
DELETE FROM t WHERE tn = '王平';
```

4.2 添加表数据

交互式操作，略，SQL插入数据知识点如下:

```
-- 插单行
INSERT INTO tableName(column1,column2...)
VALUES (value1,value2,...);

-- 插入所有属性 可省略属性名
INSERT INTO tableName VALUES (value1,value2,...);
-- 插入多行
INSERT INTO tableName(column1,column2...)
VALUES (value1,value2,...),
       (value1,value2,...),
...;
```

4.3 数据增删改练习1

> 用 SQL命令实现以下操作：
>
> 1）向表 T中插入一个教师元组（t6，李红， 女， 30， 副教授,1300，2000， 英语）
>
> 2）将“英语”课程的任课教师号修改为“t6”
>
> 3）增加年龄字段 age，并计算并填充所有学生的 age字段（用 datediff(year,birthday,getdate()) 计算并填充 age字段）
>
> 4）再将所有学生的年龄增加 1岁
>
> 5）将“高等数学”课程不及格的成绩修改为 0分
>
> 6）将低于总平均分成绩的女同学的成绩提高 5%
>
> 7）将“张小明”同学的信息分别从基本表 sc和 s中删除（使用两个 DELETE语句）
>
> 8）从基本表 c中删除“张雪”老师的任课信息

4.3.1 向表 T中插入一个教师元组（t6，李红， 女， 30， 副教授,1300，2000， 英语）

```
INSERT INTO t VALUES('t6','李红','女',30,'副教授',1300,2000,'英语');
```

4.3.2 将“英语”课程的任课教师号修改为"t6"

```
UPDATE tc SET tno = 't6' WHERE cno IN (SELECT cno FROM c WHERE cn = '英语');
```

4.3.3 增加年龄字段 age，并计算并填充所有学生的 age字段

```
ALTER TABLE s ADD COLUMN age TINYINT AFTER birthday;
UPDATE s SET age = TIMESTAMPDIFF(YEAR, birthday, CURDATE());
```

4.3.4 再将所有学生的年龄增加 1岁

```
UPDATE s SET age = age + 1 ;
```

4.3.5 将“高等数学”课程不及格的成绩修改为 0分

```
INSERT INTO sc VALUES('s1','c5',59),('s2','c5',89),('s4','c5',22);-- 插入几个不及格的成绩
UPDATE sc SET score = 0 WHERE
(score< 60 AND cno = (SELECT cno FROM c WHERE cn = '高等数学'));
```

4.3.6 将低于总平均分成绩的女同学成绩提高5% ！！！未完成

```
-- 交互式 插入一些女同学 及其成绩
SELECT cno,AVG(score)  FROM sc GROUP BY sc.cno; -- 获取各科的平均成绩
SELECT s.sno,s.sn,s.sex,sc.cno,sc.score FROM s JOIN sc ON s.sno = sc.sno WHERE s.sex = '女'; -- 获取女生的各科成绩

SELECT s.sno, s.sn, s.sex, sc.cno, sc.score, temp.avgsc
FROM s JOIN sc ON s.sno = sc.sno JOIN (SELECT cno,AVG(score) avgsc FROM sc GROUP BY sc.cno) temp ON sc.cno = temp.cno ; -- 获取各科成

SELECT s.sno, s.sn, s.sex, sc.cno, sc.score, temp.avgsc
FROM s JOIN sc ON s.sno = sc.sno JOIN (SELECT cno,AVG(score) avgsc FROM sc GROUP BY sc.cno) temp ON sc.cno = temp.cno
WHERE s.sex = '女' AND sc.score < temp.avgsc; -- 获取低于平均成绩的女生


CREATE VIEW s_avgAS AS (SELECT s.sno, s.sn, s.sex, sc.cno, sc.score, temp.avgsc
FROM s JOIN sc ON s.sno = sc.sno JOIN (SELECT cno,AVG(score) avgsc FROM sc GROUP BY sc.cno) temp ON sc.cno = temp.cno
WHERE s.sex = '女' AND sc.score < temp.avgsc); -- 将低于平均成绩的女生创建视图

SELECT * FROM s_avgAS; -- 使用视图 当表内的数据发生变化时 视图里的内容也会随之变化


SELECT * FROM sc WHERE (sc.sno,sc.cno) IN (SELECT sno,cno FROM s_avgAS); -- 在sc表中找出低于平均成绩的人
UPDATE sc SET score = 1.05*score WHERE sno IN (SELECT sno FROM s_avgAS) AND cno IN (SELECT cno FROM s_avgAS); -- 将低于总平均分成绩的
```

4.3.7 将"张小明"同学的信息分别从基本表 sc和 s中删除

```
DELETE FROM s WHERE sn = '张小明';
DELETE FROM sc WHERE sno = 's3';
```

4.3.8 从基本表 c中删除"张雪"老师的任课信息

```
DELETE FROM c WHERE cno IN (SELECT tc.cno FROM  tc JOIN t USING(tno) WHERE tn = '张雪';
```

4.4 数据增删改练习2

> .用 SQL命令实现以下操作： 1)为 tc表添加"Term"字段，表示此授课是针对哪一级学生的第几学期开课的，如：2018级第 5学期开课，则填写 20185 2)将 term字段统一填写 20181（表示 2018级第 1期）。 3)修改学生选课表 sc，添加 tno，term，grade字段。 4)将 term字段统一填写 20181（表示 2018级第 1学期）。 5 按 score填充 grade,100~90为 A，80以上为 B，70分以上为 C，60分以上为 D，60分以下为 E。

4.4.1 为 tc表添加"Term"字段，表示此授课是针对哪一级学生的第几学期开课的，如：2018级第 5学期开课，则填写 20185

```
ALTER TABLE tc ADD Term VARCHAR(10);

-- 插入时发现数据类型错误 故而把tinyint改为varchar
ALTER TABLE tc CHANGE COLUMN preriod preriod VARCHAR(10);
```

4.4.2 将 term字段统一填写 20181（表示 2018级第 1期）。

```
UPDATE tc SET term = '20181';
```

4.4.3 修改学生选课表 sc，添加 tno，term，grade字段。

```
ALTER TABLE sc ADD tno VARCHAR(20),
ADD term VARCHAR(10),
ADD grade VARCHAR(20);
```

4.4.4 将 term字段统一填写 20181（表示 2018级第 1期）。

```
UPDATE sc SET term = '20181';
```

4.4.5 按 score填充 grade,100~90为 A，80以上为 B，70分以上为 C，60分以上为 D，60分以下为 E

step 1. 创建存储函数，根据分数返回等级

```
-- 按 score填充 grade,100~90为 A，80以上为 B，70分以上为 C，60分以上为 D，60
-- 分以下为 E。

-- 首先定义一个存储函数 来显示输入分数 按照要求输出等级
-- 以下如不设置会出错 具体参考 https://stackoverflow.com/questions/26015160/deterministic-no-sql-or-reads-sql-data-in-its-declaration-a
SET GLOBAL log_bin_trust_function_creators = 1;
DELIMITER $$ -- 设置新的分隔符
CREATE FUNCTION getGrade(score TINYINT UNSIGNED)
RETURNS VARCHAR(1)
BEGIN
    IF score >= 90 THEN RETURN 'A';
    ELSEIF score >= 80 THEN RETURN 'B';
    ELSEIF score >= 70 THEN RETURN 'C';
    ELSEIF score >= 60 THEN RETURN 'D';
    ELSE RETURN 'E';
    END IF;
END$$
DELIMITER ; -- 恢复分隔符

--
```

step 2. 调用存储函数 填充grade

```
UPDATE sc SET grade = getGrade(score);
```

拓展:若想每次插入成绩时 自动填充grade等级 则可以利用触发器机制

实验5 简单查询

> 用 SQL命令实现以下操作：
>
> 1）查询计算机系的所有教师
>
> 2）查询所有女同学的姓名，年龄
>
> 3）查询计算机系教师开设的所有课程的课程号和课程名
>
> 4）查询年龄在 18~20岁（包括 18和 20）之间的所有学生的信息
>
> 5）查询年龄小于 20岁的所有男同学的学号和姓名
>
> 6）查询姓“李”的所有学生的姓名、年龄和性别
>
> 7）查询所有女同学所选课程的课程号
>
> 8）查询至少有一门成绩高于 90分的学生姓名和年龄
>
> 9）查询选修“微机原理”的所有学生的姓名和成绩
>
> 10）试算所有“数据库”成绩统一增加 10%后（超过 100分按 100计算），全班平均分是多少？（注意：请不要修改原始成绩）
>
> 11）试算所有“数据结构”成绩 60分以下的统一增加 10分后，仍有多少人不及格。

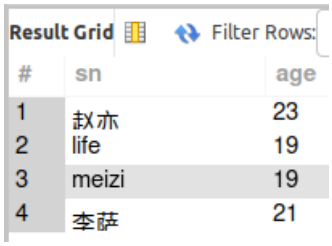
5.1 查询计算机系的所有教师。

```
SELECT * FROM t WHERE dept = '计算机';
```

#	tno	tn	sex	age	prof	sal	comm	dept
1	t1	李力	男	47	教授	1500	3000	计算机
2	t3	刘伟	男	30	讲师	900	1200	计算机
3	t7	王平	男	45	副教授	1600	5000	计算机
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

5.2 查询所有女同学的姓名，年龄

```
SELECT sn,age FROM s WHERE sex = '女';
```



#	sn	age
1	赵亦	23
2	life	19
3	meizi	19
4	李萨	21

5.3 查询计算机系教师开设的所有课程的课程号和课程名

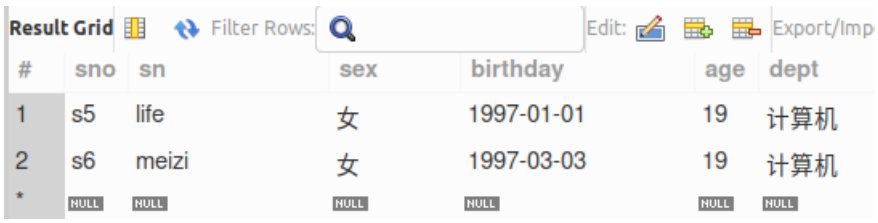
```
-- 1.查计算机系的老师有哪些
SELECT tno FROM t WHERE dept = '计算机';

-- 2.在这些老师中查他们教的课
SELECT tc.tno,tc.cno, c.cn FROM tc JOIN c ON tc.cno = c.cno
WHERE tc.tno IN (SELECT tno FROM t WHERE dept = '计算机');
```

#	tno	cno	cn
1	t1	c1	程序设计
2	t1	c2	微机原理
3	t7	c6	普通物理

5.4 查询年龄在 18~20岁之间的所有学生的信息

```
-- 设置一些年龄
UPDATE s SET age = 19 WHERE sno = 's5' or sno = 's6';
-- 条件查询
SELECT * FROM s WHERE age BETWEEN 18 AND 20;
```



#	sno	sn	sex	birthday	age	dept
1	s5	life	女	1997-01-01	19	计算机
2	s6	meizi	女	1997-03-03	19	计算机
*	NULL	NULL	NULL	NULL	NULL	NULL

5.5 查询年龄小于 20岁的所有男同学的学号和姓名

```
SELECT sno, sn FROM s WHERE age < 20 AND sex = '男';
```

#	sno	sn
1	s8	李胡
*	NULL	NULL

5.6 查询姓“李”的所有学生的姓名、年龄和性别

```
SELECT sn,age,sex FROM s WHERE sn LIKE '李%';
```

Result Grid Filter Rows:

#	sn	age	sex
1	李是	22	男
2	李思	28	男
3	李胡	17	男
4	李萨	21	女

5.7 查询所有女同学所选课程的课程号

```
SELECT s.sn,s.sex,sc.cno FROM s JOIN sc ON sc.sno = s.sno
WHERE s.sex = '女';
```

Result Grid Filter Rows:

#	sn	sex	cno
1	赵亦	女	21
2	赵亦	女	41
3	赵亦	女	c1
4	赵亦	女	c2
5	赵亦	女	c3
6	赵亦	女	c4
7	赵亦	女	c5
8	赵亦	女	c6
9	life	女	c5

5.8 查询至少有一门成绩高于 90分的学生姓名和年龄

```
SELECT s.sn,s.age,MAX(sc.score) FROM s JOIN sc ON s.sno = sc.sno
GROUP BY sc.sno HAVING MAX(sc.score) > 90;
```

Result Grid Filter Rows:

#	sn	age	MAX(sc.score)
1	赵亦	23	105
2	张小明	26	99

5.9 查询选修“微机原理”的所有学生的姓名和成绩

```
SELECT s.sn, sc.score FROM s JOIN sc ON s.sno = sc.sno
WHERE sc.cno = (SELECT cno FROM c WHERE cn = '微机原理');
```

Result Grid Filter Rows:

#	sn	score
1	赵亦	84

5.10 试算所有“数据库”成绩统一增加 10%后（超过 100分按 100计算），全班平均分是多少

```
-- 查找数据库的课程编号
SELECT cno FROM c WHERE cn = '数据库';
-- 数据库的全班平均成绩
SELECT dept,AVG(sc.score*1.1) FROM s JOIN sc ON s.sno = sc.sno WHERE sc.cno = (SELECT cno FROM c WHERE cn = '数据库')
GROUP BY dept;
```

Result Grid | Filter Rows:

#	dept	AVG(sc.score*1.1)
1	计算机	82.50000
2	信息	88.00000
3	自动化	44.00000

5.11 试算所有“数据结构”成绩 60分以下的统一增加 10分后，仍有多少人不及格。

```
SELECT COUNT(*) '仍不及格人数' FROM c JOIN sc ON c.cno = sc.cno
WHERE c.cn = '数据结构' AND sc.score+10 < 60;
```

#	仍不及格人数
1	2

实验六 多表查询

> 用 SQL命令实现以下操作： 1）查询至少选修课程号为“21”和“41”两门课程的学生的学号
>
> 2）查询选修了“高等数学”或“普通物理”的学生姓名
>
> 3）查询选修了王平老师所讲授所有课程的学生的学号和成绩
>
> 4）查询未选修王老师所讲授任意课程的学生的学号和成绩
>
> 5）查询选修了“计算机”系教师所讲授的课程的学生姓名和成绩
>
> 6）查询学号比“张小明”同学大而年龄比他小的学生姓名
>
> 7）查询年龄大于所有女同学年龄的男学生的姓名和年龄
>
> 8）查询未选修“高等数学”的学生的学号和姓名
>
> 9）查询不是计算机系教师所讲授的课程的课程名和课程号
>
> 10)查询未选修“21”号课程的学生的学号和姓名
>
> 11)从学生表和教师表可以了解到哪些院系名称
>
> 12)查询哪些学生所选的课程是由本院系的教师教的，列举学生姓名、课程名和教师名
>
> 13)如果在同一个班上课就认定为同学，请列举所有可能的同学关系，至少包含三列：学生姓名、同学姓名、共同课程名.
>
> 14)由于课程有上下承接关系，请列举课程先后关系，必须先上的在前，后上的在后，无承接关系的不列举

6.1 查询至少选修课程号为“21”和“41”两门课程的学生的学号

```
-- 选修了21和41的人的学号
SELECT s1.sno FROM sc s1 JOIN sc s2 ON s1.sno = s2.sno
WHERE s1.cno = '21' AND s2.cno = '41';
```

Result Grid

#	sno
1	s1
2	s3

6.2 查询选修了“高等数学”或“普通物理”的学生姓名

```
-- 1.先查找“高等数学”和“普通物理”的课程号
SELECT * FROM c WHERE cn = '高等数学' OR cn = '普通物理';

-- 2.根据上述课程号查学生信息
SELECT s.sn,sc.cno FROM sc JOIN s ON sc.sno = s.sno
WHERE sc.cno IN
(SELECT c.cno FROM c WHERE cn = '高等数学' OR cn = '普通物理');
```

Result Grid Filter Rows:

#	sn	cno
1	赵亦	c5
2	赵亦	c6
3	张小明	c5
4	张小明	c6
5	李思	c5
6	life	c5

6.3 查询选修了王平老师所讲授所有课程的学生的学号和成绩

```
-- 查找王平老师所教的课的课程号
SELECT tc.cno FROM t JOIN tc ON t.tno = tc.tno
WHERE t.tn = '王平';

-- 查找选修了该课程号的学生成绩
SELECT sc.sno, sc.cno, sc.score FROM sc WHERE sc.cno IN (SELECT tc.cno FROM t JOIN tc ON t.tno = tc.tno WHERE t.tn = '王平') ;
```

Result Grid Filter Rows:

#	sno	cno	score
1	s1	c6	98
2	s2	c6	77
*	NULL	NULL	NULL

6.4 查询未选修王老师所讲授任意课程的学生的学号和成绩

```
-- 上题取反即可
SELECT sc.sno, sc.cno, sc.score FROM sc WHERE sc.cno NOT IN (SELECT tc.cno FROM t JOIN tc ON t.tno = tc.tno WHERE t.tn = '王平');
```

Result Grid Filter Rows:

#	sno	cno	score
1	s1	21	98
2	s1	41	70
3	s1	c1	105
4	s1	c2	84
5	s1	c3	70
6	s1	c4	90
7	s1	c5	54
8	s10	c3	80
9	s2	21	49
10	s2	c3	80
11	s2	c4	30
12	s2	c5	99
13	s3	21	89
14	s3	41	77
15	s3	c4	45
16	s4	41	87
17	s4	c3	40
18	s4	c4	55
19	s4	c5	20
20	s5	c5	33
*	NULL	NULL	NULL

6.5 查询选修了计算机系教师所讲授的课程的学生姓名和成绩

```
-- 查询计算机系教师所讲授的课程号
SELECT tc.cno FROM t JOIN tc ON t.tno = tc.tno
WHERE t.dept = '计算机';

-- 根据课程号查学生姓名和成绩
SELECT s.sn,sc.cno,sc.score FROM s JOIN sc ON s.sno = sc.sno
WHERE sc.cno IN (SELECT tc.cno FROM t JOIN tc ON t.tno = tc.tno
WHERE t.dept = '计算机');
```

Result Grid Filter Rows:

#	sn	cno	score
1	赵亦	c1	105
2	赵亦	c2	84
3	赵亦	c6	98
4	张小明	c6	77

6.6 查询学号比“张小明”同学大而年龄比他小的学生姓名

```
-- 通过自连接 把张小明和其它同学的信息比较 注意在另外一个表要排除自身
SELECT s1.sn,s1.age,s2.sn,s2.age FROM s s1 JOIN s s2 ON s1.sn = '张小明' AND s2.sn <> '张小明' WHERE s2.sno > s1.sno AND s2.age < s1.
```

Result Grid Filter Rows:

#	sn	cno	score
1	赵亦	c1	105
2	赵亦	c2	84
3	赵亦	c6	98
4	张小明	c6	77

6.7 查询年龄大于所有女同学年龄的男学生的姓名和年龄

```
SELECT * FROM s WHERE sex = '男' AND
age > (SELECT MAX(age) FROM s WHERE sex = '女');
```

Result Grid Filter Rows: Edit: Export/I

#	sno	sn	sex	birthday	age	dept
1	s2	张小明	男	1996-01-10	26	信息
2	s4	李思	男	1995-06-01	28	自动化
3	s7	王伟	男	2003-03-04	26	信息
*	NULL	NULL	NULL	NULL	NULL	NULL

6.8 查询未选修“高等数学”的学生的学号和姓名

```
-- 选修了高等数学的学号
SELECT sno FROM c JOIN sc USING(cno)
WHERE c.cn = '高等数学';
-- 在排除上述学号后的s表中查找
SELECT sno, sn FROM s WHERE
sno NOT IN (SELECT sno FROM c JOIN sc USING(cno)
WHERE c.cn = '高等数学');
```

Result Grid

#	sno	sn
1	s10	李昱
2	s6	meizi
3	s7	干伟
4	s8	李胡
5	s9	李萨
*	NULL	NULL

6.9 查询不是计算机系教师所讲授的课程的课程名和课程号

```
-- 查询计算机系老师讲的课程号
SELECT tc.cno FROM t JOIN tc USING(tno)
WHERE dept = '计算机';

-- 排除上述课程号后查询即可
SELECT cno, cn FROM c WHERE cno NOT IN(SELECT tc.cno FROM t JOIN tc USING(tno)
WHERE dept = '计算机');
```

Result Grid

#	sno	sn
1	s10	李昱
2	s6	meizi
3	s7	干伟
4	s8	李胡
5	s9	李萨
*	NULL	NULL

6.10 查询未选修“21”号课程的学生的学号和姓名

```
-- 选修了21课的学生学号
SELECT sno FROM sc WHERE sc.cno = '21';

-- 排除上述学号即可
SELECT sno,sn FROM s WHERE
sno NOT IN (SELECT sno FROM sc WHERE sc.cno = '21');
```

Result Grid

#	sno	sn
1	s10	李昱
2	s4	李思
3	s5	life
4	s6	meizi
5	s7	干伟
6	s8	李胡
7	s9	李萨
*	NULL	NULL

6.11 从学生表和教师表可以了解到哪些院系名称

```
(SELECT  dept FROM s)
UNION
(SELECT  dept FROM t); -- 两个结果取交集即可 并集为UNION ALL
```

Result Grid

#	dept
1	计算机
2	信息
3	自动化
4	英语

6.12 查询哪些学生所选的课程是由本院系的教师教的，列举学生姓名、课程名和教师名

```
-- 根据学号查学生的系
SELECT sno,sn,dept FROM s;

-- 根据课程号查该课是由哪个系的老师教的
SELECT c.cn,tt.dept,tt.tn FROM c JOIN
(SELECT tc.cno, t.tn, dept FROM t JOIN tc USING(tno)) tt USING(cno);
```

Result Grid Filter Rows:

#	cn	dept	tn
1	程序设计	计算机	李力
2	微机原理	计算机	李力
3	数据库	信息	张兰
4	高等数学	英语	李红
5	普通物理	计算机	王平

```
-- 根据院系相同来连接上述两个表
SELECT a.sn, b.cn, b.tn,dept FROM (SELECT sno,sn,dept FROM s) a
JOIN (SELECT c.cn,tt.dept,tt.tn FROM c JOIN
(SELECT tc.cno, t.tn, dept FROM t JOIN tc USING(tno)) tt USING(cno)) b USING(dept);
```

Result Grid Filter Rows: Export: Wrap Cell Cor

#	sn	cn	tn	dept
1	赵亦	普通物理	王平	计算机
2	赵亦	微机原理	李力	计算机
3	赵亦	程序设计	李力	计算机
4	李昱	普通物理	王平	计算机
5	李昱	微机原理	李力	计算机
6	李昱	程序设计	李力	计算机
7	张小明	数据库	张兰	信息
8	life	普通物理	王平	计算机
9	life	微机原理	李力	计算机
10	life	程序设计	李力	计算机
11	meizi	普通物理	王平	计算机
12	meizi	微机原理	李力	计算机
13	meizi	程序设计	李力	计算机
14	王伟	数据库	张兰	信息
15	李萨	普通物理	王平	计算机
16	李萨	微机原理	李力	计算机
17	李萨	程序设计	李力	计算机

6.13 如果在同一个班上课就认定为同学，请列举所有可能的同学关系，至少包含三列：学生姓名、同学姓名、共同课程名

```
-- 列出每个课都那些人选了 创建为视图
CREATE OR REPLACE VIEW cn_cno AS
(SELECT c.cn,sc.sno FROM sc JOIN c USING(cno) ORDER BY c.cn);
-- 执行视图
SELECT * FROM cn_cno;
```

Result Grid | Filter Rows:

#	cn	sno
1	微机原理	s1
2	数字图像处理	s1
3	数字图像处理	s4
4	数字图像处理	s3
5	数据库	s1
6	数据库	s4
7	数据库	s10
8	数据库	s2
9	数据结构	s2
10	数据结构	s4
11	数据结构	s3
12	数据结构	s1
13	普通物理	s2
14	普通物理	s1
15	现代数字通信	s1
16	现代数字通信	s3
17	现代数字通信	s2
18	程序设计	s1
19	高等数学	s2
20	高等数学	s1
21	高等数学	s4
22	高等数学	s5

```
-- 自连接查询所有同学关系
SELECT cc1.cn,cc1.sno sno1, cc2.sno sno2  FROM cn_cno cc1 JOIN cn_cno cc2 USING(cn)
WHERE cc1.sno <> cc2.sno ORDER BY cn,cc1.sno;
```

因为关系很长 此处只列举一部分

Result Grid | Filter Rows:

#	cn	sno1	sno2
1	数字图像处理	s1	s4
2	数字图像处理	s1	s3
3	数字图像处理	s3	s1
4	数字图像处理	s3	s4
5	数字图像处理	s4	s1
6	数字图像处理	s4	s3
7	数据库	s1	s4
8	数据库	s1	s2
9	数据库	s1	s10
10	数据库	s10	s2
11	数据库	s10	s4
12	数据库	s10	s1
13	数据库	s2	s4
14	数据库	s2	s10
15	数据库	s2	s1
16	数据库	s4	s10

```
-- 根据两个人的学号 显示他们的姓名 需要用自连接 如查s1和s2的姓名
SELECT s1.sno,s1.sn,s2.sno,s2.sn FROM s s1 JOIN s s2 WHERE s1.sno = 's1' AND s2.sno = 's2';
```

Result Grid | Filter Rows:

#	sno	sn	sno	sn
1	s1	赵亦	s2	张小明

```
-- 综合上述查询三表连接即可做出本题
SELECT tt.cn,s1.sno,s1.sn,s2.sno,s2.sn FROM s s1 JOIN s s2 JOIN (SELECT cc1.cn,cc1.sno sno1, cc2.sno sno2  FROM cn_cno cc1 JOIN cn_c
WHERE cc1.sno <> cc2.sno ORDER BY cn,cc1.sno) tt
WHERE s1.sno = tt.sno1 AND s2.sno = tt.sno2 ORDER BY tt.cn,s1.sn,s2.sn;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

#	cn	sno	sn	sno	sn
1	数字图像处理	s4	李思	s1	赵亦
2	数字图像处理	s1	赵亦	s4	李思
3	数据库	s2	张小明	s4	李思
4	数据库	s2	张小明	s10	李昱
5	数据库	s2	张小明	s1	赵亦
6	数据库	s4	李思	s2	张小明
7	数据库	s4	李思	s10	李昱
8	数据库	s4	李思	s1	赵亦
9	数据库	s10	李昱	s2	张小明
10	数据库	s10	李昱	s4	李思
11	数据库	s10	李昱	s1	赵亦
12	数据库	s1	赵亦	s2	张小明
13	数据库	s1	赵亦	s4	李思
14	数据库	s1	赵亦	s10	李昱
15	数据结构	s2	张小明	s4	李思
16	数据结构	s2	张小明	s1	赵亦

6.14 由于课程有上下承接关系，请列举课程先后关系，必须先上的在前，后上的在后，无承接关系的不列举

```
-- 通过赋值优先级 自定义排序即可
SELECT * FROM c ORDER BY (
CASE prevCno
WHEN NULL THEN 0
WHEN 'c6' THEN 3
WHEN 'c5' THEN 2
ELSE 1
END
) ASC;
```

Result Grid | Filter Rows: | Edit:

#	cno	cn	time	credit	prevCn
1	21	现代数字通信	24	1	
2	c1	程序设计	60	3	NULL
3	c2	微机原理	60	3	c1
4	c3	数据库	90	4	c1
5	c4	数据结构	120	5	c1
6	c5	高等数学	80	4	NULL
7	c6	普通物理	64	4	c5
8	41	数字图像处理	22	2	c6
*	NULL	NULL	NULL	NULL	NULL

实验七 统计查询

> 用 SQL命令实现以下操作： 1）查询女同学的人数
>
> 2）查询男同学的平均年龄
>
> 3）查询男、女同学各有多少人
>
> 4）查询年龄大于女同学平均年龄的男学生的姓名和年龄

> 5）查询所有学生选修的课程门数
>
> 6）查询每门课程的学生选修人数（只输出超过 10人的课程），要求输出课程号和课程名及选修人数，查询结果按人数降序排列，若人数相同，按课程号升序排列
>
> 7）查询只选修了一门课程的学生学号和姓名
>
> 8）查询至少选修了两门课程的学生学号
>
> 9）查询至少讲授两门课程的教师姓名和其所在系
>
> 10）查询高等数学课程的平均分
>
> 11)查询每个学生的总分，要求输出学号和分数，并按分数由高到低排列，分数相同时按学号升序排列
>
> 12)查询各科成绩等级分布情况，即看每门课程 A等多少人、B等多少人...
>
> 13)统计各科成绩等级分布情况存入新表 statgrade，即看每门课程 A等多少人、B等多少人...
>
> 14)统计各科课程号、课程名、选课人数、平均分、最高分、最低分，并存入新表 statscore

7.1 查询女同学的人数

```
SELECT COUNT(*) FROM s WHERE sex = '女';
```

Result Grid

#	COUNT(*)
1	4

7.2 查询男同学的平均年龄

```
SELECT AVG(age) FROM s WHERE sex = '男';
```

Result Grid

#	AVG(age)
1	23.8000

7.3 查询男、女同学各有多少人

```
SELECT COUNT(*) FROM s GROUP BY sex;
```

#	COUNT(*)
1	4
2	5

7.4 查询年龄大于女同学平均年龄的男学生的姓名和年龄

```
-- 女同学平均年龄
SELECT AVG(age) FROM s WHERE sex = '女';

-- 在男同学中找大于上述年龄的
SELECT sn,age FROM s WHERE sex = '男' AND age > (SELECT AVG(age) FROM s WHERE sex = '女');
```

Result Grid Filter Rows:

#	sn	age
1	李昱	22
2	张小明	26
3	李思	28
4	王伟	26

7.5 查询所有学生选修的课程门数

```
SELECT sno, COUNT(*) '选修门数' FROM sc GROUP BY sno;
```

Result Grid Filte

#	sno	选修门数
1	s1	8
2	s10	1
3	s2	5
4	s3	3
5	s4	4
6	s5	1

7.6 查询每门课程的学生选修人数（只输出超过2人(10人的数据不好造)的课程），要求输出课程号和课程名及选修人数，查询结果按人数降序排列，若人数相同，按课程号升序排列

```
SELECT cno,cn, COUNT(*) '选修人数' FROM sc JOIN c USING(cno) GROUP BY cno HAVING 选修人数 > 2 ORDER BY 选修人数 DESC,cno ASC;
```

Result Grid Filter Rows:

#	cno	cn	选修人数
1	c3	数据库	4
2	c4	数据结构	4
3	c5	高等数学	4
4	21	现代数字通信	3
5	41	数字图像处理	3

7.7 查询只选修了一门课程的学生学号和姓名

```
SELECT s.sno,s.sn,COUNT(*) '选修门数' FROM s JOIN sc USING(sno)
GROUP BY sno HAVING 选修门数=1;
```

Result Grid Filter Rows:

#	sno	sn	选修门数
1	s10	李昱	1
2	s5	life	1

7.8 查询至少选修了两门课程的学生学号

```
SELECT s.sno,s.sn,COUNT(*) '选修门数' FROM s JOIN sc USING(sno)
GROUP BY sno HAVING 选修门数>=2;
```

Result Grid Filter Rows:

#	sno	sn	选修门数
1	s1	赵亦	8
2	s2	张小明	5
3	s4	李思	4

7.9 查询至少讲授两门课程的教师姓名和其所在系

```
SELECT tno,t.tn,t.dept, COUNT(*) '讲授门数' FROM tc JOIN t USING(tno)
GROUP BY tno HAVING 讲授门数>=2;
```

Result Grid Filter Rows:

#	tno	tn	dept	讲授门数
1	t1	李力	计算机	2

7.10 查询高等数学课程的平均分

```
SELECT cno,c.cn,AVG(score) FROM sc JOIN c USING(cno)
WHERE c.cn = '高等数学' GROUP BY cno;
```

Result Grid | Filter Rows: | Exp

#	cno	cn	AVG(score)
1	c5	高等数学	51.5000

7.11 查询每个学生的总分，要求输出学号和分数，并按分数由高到低排列，分数相同时按学号升序排列

```
SELECT sno,SUM(score) 总分 FROM sc
GROUP BY sno ORDER BY 总分 DESC, sno ASC;
```

Result Grid

#	sno	总分
1	s1	669
2	s2	335
3	s3	211
4	s4	202
5	s10	80
6	s5	33

7.12 查询各科成绩等级分布情况，即看每门课程 A等多少人、B等多少人

```
SELECT cno,grade,COUNT(*) FROM sc GROUP BY cno,grade ORDER BY cno;
```

Result Grid | Filter Rows:

#	cno	grade	COUNT(*)
1	21	A	1
2	21	B	1
3	21	E	1
4	41	B	1
5	41	C	2
6	c1	A	1
7	c2	C	1
8	c3	B	2
9	c3	C	1
10	c3	E	1
11	c4	A	1
12	c4	E	3
13	c5	A	1
14	c5	E	3
15	c6	A	1
16	c6	C	1

7.13 统计各科成绩等级分布情况存入新表 statgrade，即看每门课程 A等多少人、B等多少人

```
CREATE TABLE IF NOT EXISTS statgrade (SELECT cno,grade,COUNT(*) FROM sc GROUP BY cno,grade ORDER BY cno);
SELECT * FROM statgrade;
```

Result Grid | Filter Rows:

#	cno	grade	COUNT(*)
1	21	A	1
2	21	B	1
3	21	E	1
4	41	B	1
5	41	C	2
6	c1	A	1
7	c2	C	1
8	c3	B	2
9	c3	C	1
10	c3	E	1
11	c4	A	1
12	c4	E	3
13	c5	A	1
14	c5	E	3
15	c6	A	1
16	c6	C	1

7.14 统计各科课程号、课程名、选课人数、平均分、最高分、最低分，并存入新表 statscore

```
CREATE TABLE IF NOT EXISTS statscore SELECT sc.cno,c.cn,COUNT(*),AVG(score),MAX(score),MIN(score)
FROM sc JOIN c USING(cno) GROUP BY cno;
SELECT * FROM statscore;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

#	cno	cn	COUNT(*)	AVG(score)	MAX(score)	MIN(score)
1	21	现代数字通信	3	78.6667	98	49
2	41	数字图像处理	3	78.0000	87	70
3	c1	程序设计	1	105.0000	105	105
4	c2	微机原理	1	84.0000	84	84
5	c3	数据库	4	67.5000	80	40
6	c4	数据结构	4	55.0000	90	30
7	c5	高等数学	4	51.5000	99	20
8	c6	普通物理	2	87.5000	98	77