# 数据库实验

howlinggggg

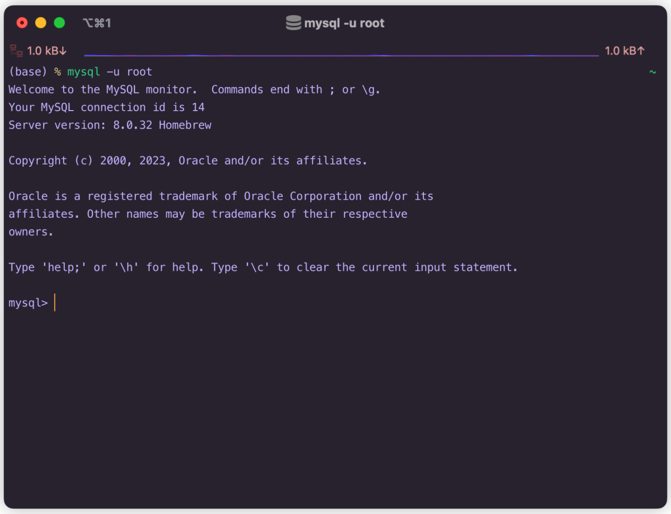
2023 年 4 月 15 日

## 1　实验目的

设有如下关系模式：

- Student(SNO, NAME, GENDER, BIRTHDAY, DEPART)
  SNO 为学生学号，DEPART 为系别
- Teacher(TNO,NAME,GENDER,BIRTHDAY,POSITION,DEPART)
  TNO 为教师工号，POSITION 为职称，DEPART 为系别。
- Course(CNO, NAME, TNO)
  CNO 为课程编号，TNO 为教师工号。
- Score(SNO, CNO, DEGREE)
  SNO 为学生学号，CNO 为课程编号，DEGREE 为成绩。

## 2　实验过程



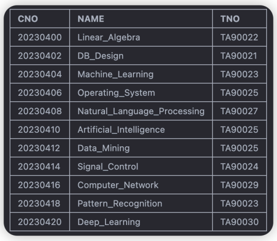
```
(base) % mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 8.0.32 Homebrew

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```

### 2.1　任务一

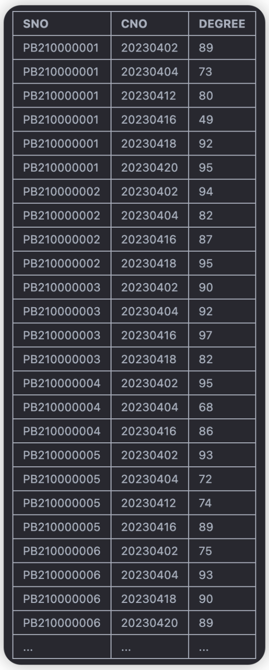
根据所给 Student.csv、Teacher.csv、Course.csv、Score.csv 表中的数据信息，在数据库中创建对应的关系表并将数据录入到数据库中。可能涉及到的数据类型：varchar, char, int, float ,datetime。（也需要截图展示！）

```
create table if not exists course(
CNO char(8),
NAME varchar(50),
TNO char(7),
primary key(CNO)
);

create table if not exists score(
SNO char(12),
CNO char(8) ,
DEGREE int unsigned,
primary key(SNO, CNO)
);

create table if not exists student(
SNO char(12),
NAME varchar(4),
GENDER varchar(6),
BIRTHDAY datetime,
DEPART int unsigned,
primary key(SNO)
);

create table if not exists teacher(
TNO char(12),
NAME varchar(4),
GENDER varchar(6),
BIRTHDAY datetime,
POSITION varchar(20),
DEPART int unsigned,
primary key(TNO)
);
```

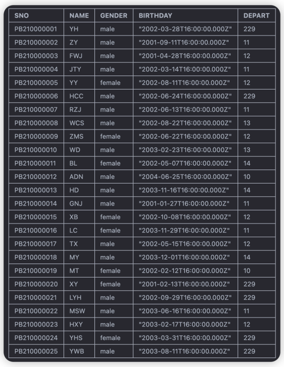
```
load data infile './data/Course.csv'
into table course
FIELDS TERMINATED BY ','
ENCLOSED BY ''''
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;

load data infile './data/Score.csv'
into table score
FIELDS TERMINATED BY ','
ENCLOSED BY ''''
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;

load data infile './data/Student.csv'
into table student
FIELDS TERMINATED BY ','
ENCLOSED BY ''''
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;

load data infile './data/Teacher.csv'
into table teacher
FIELDS TERMINATED BY ','
ENCLOSED BY ''''
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```



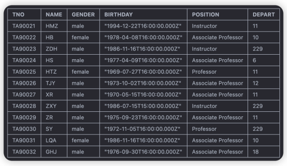
| CNO | NAME | TNO |
|---|---|---|
| 20230400 | Linear_Algebra | TA90022 |
| 20230402 | DB_Design | TA90021 |
| 20230404 | Machine_Learning | TA90023 |
| 20230406 | Operating_System | TA90025 |
| 20230408 | Natural_Language_Processing | TA90027 |
| 20230410 | Artificial_Intelligence | TA90025 |
| 20230412 | Data_Mining | TA90025 |
| 20230414 | Signal_Control | TA90024 |
| 20230416 | Computer_Network | TA90029 |
| 20230418 | Pattern_Recognition | TA90023 |
| 20230420 | Deep_Learning | TA90030 |



| SNO | CNO | DEGREE |
|---|---|---|
| PB210000001 | 20230402 | 89 |
| PB210000001 | 20230404 | 73 |
| PB210000001 | 20230412 | 80 |
| PB210000001 | 20230416 | 49 |
| PB210000001 | 20230418 | 92 |
| PB210000001 | 20230420 | 95 |
| PB210000002 | 20230402 | 94 |
| PB210000002 | 20230404 | 82 |
| PB210000002 | 20230416 | 87 |
| PB210000002 | 20230418 | 95 |
| PB210000003 | 20230402 | 90 |
| PB210000003 | 20230404 | 92 |
| PB210000003 | 20230416 | 97 |
| PB210000003 | 20230418 | 82 |
| PB210000004 | 20230402 | 95 |
| PB210000004 | 20230404 | 68 |
| PB210000004 | 20230416 | 86 |
| PB210000005 | 20230402 | 93 |
| PB210000005 | 20230404 | 72 |
| PB210000005 | 20230412 | 74 |
| PB210000005 | 20230416 | 89 |
| PB210000006 | 20230402 | 75 |
| PB210000006 | 20230404 | 93 |
| PB210000006 | 20230418 | 90 |
| PB210000006 | 20230420 | 89 |
| ... | ... | ... |



| SNO | NAME | GENDER | BIRTHDAY | DEPART |
|---|---|---|---|---|
| PB210000001 | YH | male | "2002-03-28T16:00:00.000Z" | 229 |
| PB210000002 | ZY | male | "2001-09-11T16:00:00.000Z" | 11 |
| PB210000003 | FWJ | male | "2001-04-28T16:00:00.000Z" | 12 |
| PB210000004 | JTY | male | "2002-03-14T16:00:00.000Z" | 11 |
| PB210000005 | YY | female | "2002-08-11T16:00:00.000Z" | 12 |
| PB210000006 | HCC | male | "2002-06-24T16:00:00.000Z" | 229 |
| PB210000007 | RZJ | male | "2002-06-13T16:00:00.000Z" | 11 |
| PB210000008 | WCS | male | "2002-08-22T16:00:00.000Z" | 13 |
| PB210000009 | ZMS | female | "2002-06-22T16:00:00.000Z" | 12 |
| PB210000010 | WD | male | "2003-02-23T16:00:00.000Z" | 13 |
| PB210000011 | BL | female | "2002-05-07T16:00:00.000Z" | 14 |
| PB210000012 | ADN | male | "2004-06-25T16:00:00.000Z" | 10 |
| PB210000013 | HD | male | "2003-11-16T16:00:00.000Z" | 14 |
| PB210000014 | GNJ | male | "2001-01-27T16:00:00.000Z" | 11 |
| PB210000015 | XB | female | "2002-10-08T16:00:00.000Z" | 12 |
| PB210000016 | LC | female | "2003-11-29T16:00:00.000Z" | 11 |
| PB210000017 | TX | male | "2002-05-15T16:00:00.000Z" | 12 |
| PB210000018 | MY | male | "2003-12-01T16:00:00.000Z" | 14 |
| PB210000019 | MT | female | "2002-02-12T16:00:00.000Z" | 10 |
| PB210000020 | XY | female | "2001-02-13T16:00:00.000Z" | 229 |
| PB210000021 | LYH | male | "2002-09-29T16:00:00.000Z" | 229 |
| PB210000022 | MSW | male | "2003-06-16T16:00:00.000Z" | 11 |
| PB210000023 | HXY | male | "2003-02-17T16:00:00.000Z" | 12 |
| PB210000024 | YHS | female | "2003-03-31T16:00:00.000Z" | 229 |
| PB210000025 | YWB | male | "2003-08-11T16:00:00.000Z" | 229 |



| TNO | NAME | GENDER | BIRTHDAY | POSITION | DEPART |
|---|---|---|---|---|---|
| TA90021 | HMZ | male | "1994-12-22T16:00:00.000Z" | Instructor | 11 |
| TA90022 | HB | female | "1978-04-08T16:00:00.000Z" | Associate Professor | 10 |
| TA90023 | ZDH | male | "1986-11-16T16:00:00.000Z" | Instructor | 229 |
| TA90024 | HS | male | "1977-04-09T16:00:00.000Z" | Associate Professor | 6 |
| TA90025 | HTZ | female | "1969-07-27T16:00:00.000Z" | Professor | 11 |
| TA90026 | TJY | male | "1973-10-02T16:00:00.000Z" | Associate Professor | 12 |
| TA90027 | XR | male | "1970-05-15T16:00:00.000Z" | Associate Professor | 11 |
| TA90028 | ZXY | male | "1986-07-15T15:00:00.000Z" | Instructor | 229 |
| TA90029 | ZR | male | "1975-09-23T16:00:00.000Z" | Associate Professor | 11 |
| TA90030 | SY | male | "1972-11-05T16:00:00.000Z" | Professor | 229 |
| TA90031 | LQA | female | "1986-11-16T16:00:00.000Z" | Associate Professor | 10 |
| TA90032 | GHJ | male | "1976-09-30T16:00:00.000Z" | Associate Professor | 18 |

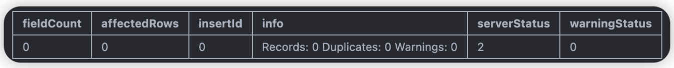
图 1: 任务一

## 2.2 任务二

依顺序写出实现以下各题功能的 SQL 语句 (部分题目的结果受前面题目影响).

1、在学生表 student 中增加一个新的属性列 AGE(年龄)，类型为 int。

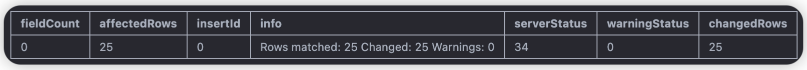
```sql
alter table student add age int;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |

2、计算每个学生的年龄 (AGE)（简单用 2023 减去出生年份即可）。注意，此操作可能需要关闭安全更新模式；提示，可使用 MySQL 的 YEAR 函数。

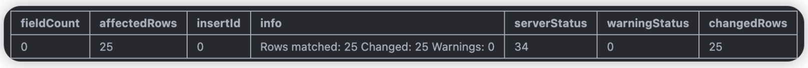
```sql
update student set age = 2023 - year(BIRTHDAY);
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus | changedRows |
|---|---|---|---|---|---|---|
| 0 | 25 | 0 | Rows matched: 25 Changed: 25 Warnings: 0 | 34 | 0 | 25 |

3、为每个学生的年龄加 2。

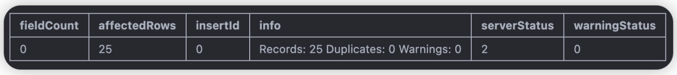
```sql
update student set age = age + 2;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus | changedRows |
|---|---|---|---|---|---|---|
| 0 | 25 | 0 | Rows matched: 25 Changed: 25 Warnings: 0 | 34 | 0 | 25 |

4、将 AGE（年龄）的数据类型由 int 改为 char。

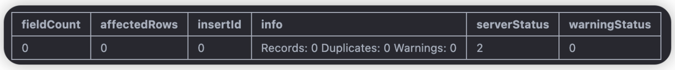
```
alter table student modify age char(2);
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 25 | 0 | Records: 25 Duplicates: 0 Warnings: 0 | 2 | 0 |

5、删除属性列 AGE。

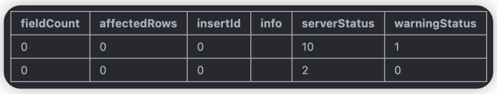
```
alter table student drop age;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |

6、创建一个教师课程数量表：teacher_course(TNO,NUM_COURSE)，两个属性分别表示授课教师工号，课程数量，类型自定义 (注意，这里 TNO 还不是主键)。

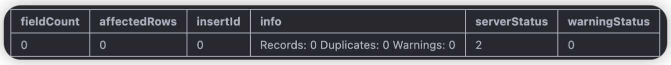
```
drop table if exists teacher_course;
create table if not exists teacher_course(
TNO char(12),
NUM_COURSE int unsigned
);
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 |  | 10 | 1 |
| 0 | 0 | 0 |  | 2 | 0 |

7、为表 teacher_course 添加主键 (TNO)。

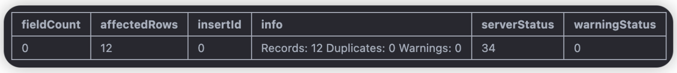
```sql
alter table teacher_course add primary key(TNO);
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |

8、用一条语句，结合表 course 记录，为表 teacher 中所有教师，在表 teacher_course 添加对应记录（若是表 course 中未出现的教师，则课程数量记为 NULL）。

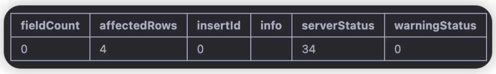
```sql
insert into teacher_course (TNO, NUM_COURSE)
select t.TNO, if(count(c.CNO), count(c.CNO), null)
from teacher t
left outer join course c on c.TNO = t.TNO
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 12 | 0 | Records: 12 Duplicates: 0 Warnings: 0 | 34 | 0 |

9、删除表 teacher_course 中含有 NULL 的记录。

```sql
delete from teacher_course where NUM_COURSE is null;
```



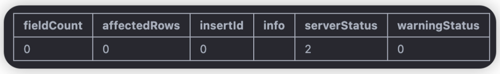
| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 4 | 0 |  | 34 | 0 |

10、删除表 teacher_course。

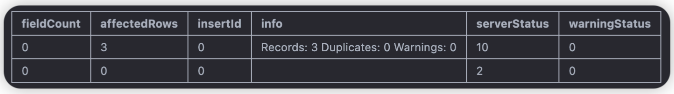
```sql
drop table if exists teacher_course;
```

11、在学生表 student 、成绩表 score 中分别插入一些数据

```sql
insert into student(SNO, NAME, GENDER, BIRTHDAY, DEPART)
values
('PB18061443', 'JHL', 'MALE', '2000-05-23 0:00', 229),
('PB20061376', 'GSB', 'MALE', '2002-07-02 0:00', 229),
('PB18061444', 'LGM', 'MALE', '2000-04-02 0:00', 229);

insert into score(SNO, CNO, DEGREE)
values
('PB18061443', '20230402', 97),
('PB18061443', '20230410', 98),
('PB18061443', '20230412', 99);
```



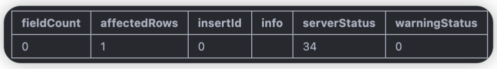
| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | | 2 | 0 |



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 3 | 0 | Records: 3 Duplicates: 0 Warnings: 0 | 10 | 0 |
| 0 | 0 | 0 | | 2 | 0 |

12、在 score 表中删除你所选的课程中成绩最低的一门课程的记录（可能会用到子查询）。

```
DELETE FROM score
WHERE (SNO, DEGREE) = (
  SELECT SNO, min_degree
  FROM (
    SELECT SNO, MIN(DEGREE) as min_degree
    FROM score
    WHERE SNO = 'PB18061443'
  ) AS temp
  WHERE SNO = 'PB18061443'
)
LIMIT 1;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 1 | 0 | | 34 | 0 |

## 2.3 索引

13、用 create 语句在 course 表的名称 NAME 上建立普通索引 NAME_INDEX。

```
create index NAME_INDEX on course(NAME);
```

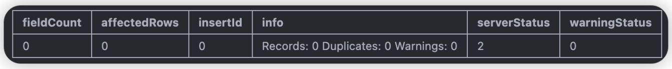
14、用 create 语句在 teacher 表的工号 TNO 上建立唯一索引 TNO_INDEX。

```
create unique index TNO_INDEX on teacher(TNO);
```

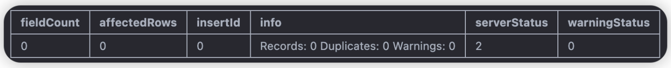
15、用 create 语句在 score 表上的学号 SNO 、成绩 DEGREE 上建立复合索引 RECORD_INDEX，要求学号为降序，学号相同时成绩为升序。

```
create index RECORD_INDEX on score(sno desc, degree asc);
```

16、用一条语句查询表 score 的索引。



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |

```
show index from score;
```

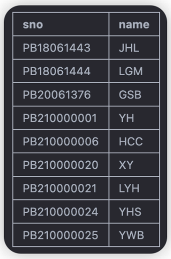
17、删除 teacher 表字段 TNO 上的索引 TNO_INDEX

```
alter table teacher drop index TNO_INDEX;
```

## 2.4 查询

18、查询和你属于同一个系的学生学号和姓名 (包括你本人)。

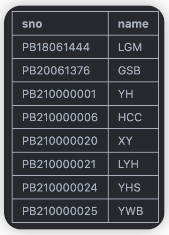
```
select sno, name
from student
where depart = (
    select depart
    from student
    where name = 'JHL'
);
```



| sno | name |
|---|---|
| PB18061443 | JHL |
| PB18061444 | LGM |
| PB20061376 | GSB |
| PB210000001 | YH |
| PB210000006 | HCC |
| PB210000020 | XY |
| PB210000021 | LYH |
| PB210000024 | YHS |
| PB210000025 | YWB |

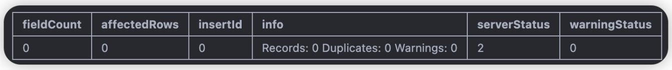
19、查询和你属于同一个系的学生学号和姓名 (不包括你本人)。

```
select sno, name
from student
where depart = (
    select depart
    from student
    where sno = 'PB18061443'
)
and sno != 'PB18061443';
```

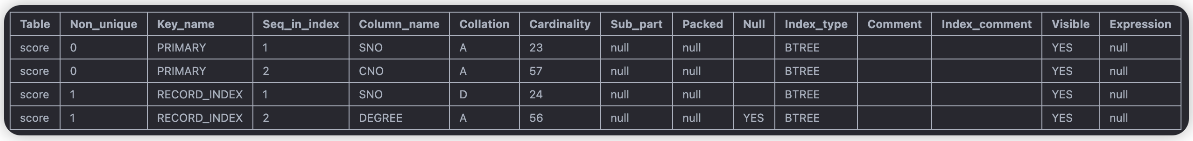


| sno | name |
|---|---|
| PB18061444 | LGM |
| PB20061376 | GSB |
| PB210000001 | YH |
| PB210000006 | HCC |
| PB210000020 | XY |
| PB210000021 | LYH |
| PB210000024 | YHS |
| PB210000025 | YWB |

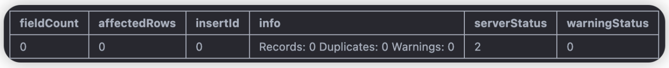
20、查询和你的某个好友属于同一个系的学生学号和姓名（11 题插入的某个好友)。



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |

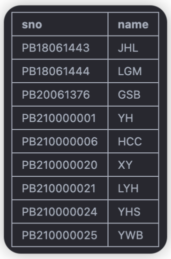


| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| score | 0 | PRIMARY | 1 | SNO | A | 23 | null | null |  | BTREE |  |  | YES | null |
| score | 0 | PRIMARY | 2 | CNO | A | 57 | null | null |  | BTREE |  |  | YES | null |
| score | 1 | RECORD_INDEX | 1 | SNO | D | 24 | null | null |  | BTREE |  |  | YES | null |
| score | 1 | RECORD_INDEX | 2 | DEGREE | A | 56 | null | null | YES | BTREE |  |  | YES | null |



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Records: 0 Duplicates: 0 Warnings: 0 | 2 | 0 |

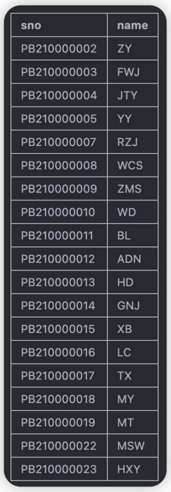
```sql
select sno, name
from student
where depart = (
    select depart
    from student
    where name = 'LGM'
);
```



| sno | name |
|---|---|
| PB18061443 | JHL |
| PB18061444 | LGM |
| PB20061376 | GSB |
| PB210000001 | YH |
| PB210000006 | HCC |
| PB210000020 | XY |
| PB210000021 | LYH |
| PB210000024 | YHS |
| PB210000025 | YWB |

21、查询和你的两个好友都不在一个系的学生学号和姓名（11 题插入的两个好友）。

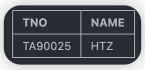
```sql
select sno, name
from student
where depart not in (
    select depart
    from student
    where name in ('LGM', 'GSB')
);
```



| sno | name |
|---|---|
| PB210000002 | ZY |
| PB210000003 | FWJ |
| PB210000004 | JTY |
| PB210000005 | YY |
| PB210000007 | RZJ |
| PB210000008 | WCS |
| PB210000009 | ZMS |
| PB210000010 | WD |
| PB210000011 | BL |
| PB210000012 | ADN |
| PB210000013 | HD |
| PB210000014 | GNJ |
| PB210000015 | XB |
| PB210000016 | LC |
| PB210000017 | TX |
| PB210000018 | MY |
| PB210000019 | MT |
| PB210000022 | MSW |
| PB210000023 | HXY |

22、查询教过你的所有老师的工号和姓名。

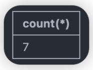
```sql
select distinct t.TNO, t.NAME
from teacher t
join course c on c.TNO = t.TNO
join score s on s.CNO = c.CNO
where s.SNO = 'PB18061443';
```



| TNO | NAME |
|---|---|
| TA90025 | HTZ |

23、查询 11 系和 229 系教师的总人数。

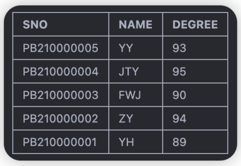
```sql
select count(*)
from teacher
where depart in (11, 229)
```



| count(*) |
|---|
| 7 |

24、查询选修 DB_Design 课程且成绩在 89 分以上（包括 89）的学生的学号、姓名和分数。

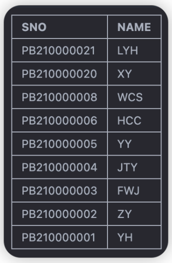
```sql
select stu.SNO, stu.NAME, s.DEGREE
from student stu
join score s on stu.SNO = s.SNO
join course c on c.CNO = s.CNO
where c.NAME = 'DB_Design' and s.DEGREE >= 89
```



| SNO | NAME | DEGREE |
|---|---|---|
| PB210000005 | YY | 93 |
| PB210000004 | JTY | 95 |
| PB210000003 | FWJ | 90 |
| PB210000002 | ZY | 94 |
| PB210000001 | YH | 89 |

25、查询选修过“ZDH”老师课程的学生学号和姓名（去掉重复行）。

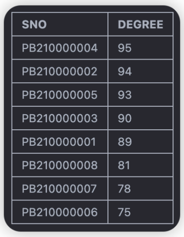
```sql
select distinct stu.SNO, stu.NAME
from student stu
join score sc on sc.SNO = stu.SNO
join course c on c.CNO = sc.CNO
join teacher t on t.TNO = c.TNO
where t.NAME = 'ZDH';
```



| SNO | NAME |
|---|---|
| PB210000021 | LYH |
| PB210000020 | XY |
| PB210000008 | WCS |
| PB210000006 | HCC |
| PB210000005 | YY |
| PB210000004 | JTY |
| PB210000003 | FWJ |
| PB210000002 | ZY |
| PB210000001 | YH |

26、查询选过某课程的学生学号和分数，并按分数降序展示。(某课程是指 course 表中的某一课程名 NAME，你自行选择)。

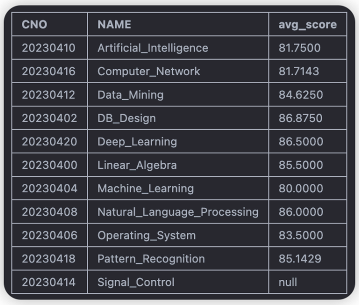
```sql
select s.SNO, s.DEGREE
from score s
join course c on c.CNO = s.CNO
where c.NAME = 'DB_Design'
order by s.DEGREE desc;
```



| SNO | DEGREE |
|---|---|
| PB210000004 | 95 |
| PB210000002 | 94 |
| PB210000005 | 93 |
| PB210000003 | 90 |
| PB210000001 | 89 |
| PB210000008 | 81 |
| PB210000007 | 78 |
| PB210000006 | 75 |

27、查询每门课的平均成绩，其中每行包含课程号、课程名和平均成绩（包括平均成绩为 NULL，即该课没有成绩)。

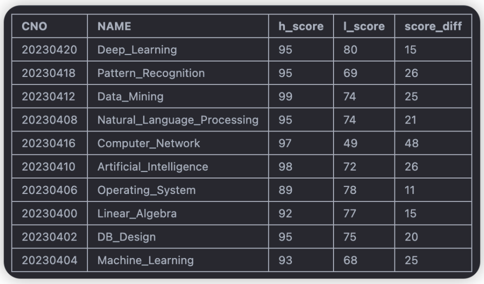
```sql
select c.CNO, c.NAME, avg(s.DEGREE) as avg_score
from course c
left join score s on s.CNO = c.CNO
group by c.CNO, c.NAME;
```



| CNO | NAME | avg_score |
|---|---|---|
| 20230410 | Artificial_Intelligence | 81.7500 |
| 20230416 | Computer_Network | 81.7143 |
| 20230412 | Data_Mining | 84.6250 |
| 20230402 | DB_Design | 86.8750 |
| 20230420 | Deep_Learning | 86.5000 |
| 20230400 | Linear_Algebra | 85.5000 |
| 20230404 | Machine_Learning | 80.0000 |
| 20230408 | Natural_Language_Processing | 86.0000 |
| 20230406 | Operating_System | 83.5000 |
| 20230418 | Pattern_Recognition | 85.1429 |
| 20230414 | Signal_Control | null |

28、查询每门课程的最高分和最低分，并计算其分数差。其中每行包含课程号、课程名和最高分、最低分和分数差。(课程无成绩的不用包括)。

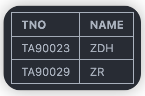
```sql
select c.CNO, c.NAME, max(s.DEGREE) as h_score,
    min(s.DEGREE) as l_score, (max(s.DEGREE) -
    min(s.DEGREE)) as score_diff
from course c
join score s on s.CNO = c.CNO
group by c.CNO, c.NAME
having count(s.DEGREE) > 0;
```



| CNO | NAME | h_score | l_score | score_diff |
|---|---|---|---|---|
| 20230420 | Deep_Learning | 95 | 80 | 15 |
| 20230418 | Pattern_Recognition | 95 | 69 | 26 |
| 20230412 | Data_Mining | 99 | 74 | 25 |
| 20230408 | Natural_Language_Processing | 95 | 74 | 21 |
| 20230416 | Computer_Network | 97 | 49 | 48 |
| 20230410 | Artificial_Intelligence | 98 | 72 | 26 |
| 20230406 | Operating_System | 89 | 78 | 11 |
| 20230400 | Linear_Algebra | 92 | 77 | 15 |
| 20230402 | DB_Design | 95 | 75 | 20 |
| 20230404 | Machine_Learning | 93 | 68 | 25 |

29、查询所教过的课程中有学生考试成绩低于 72 分的教师的工号和姓名（去掉重复行)。

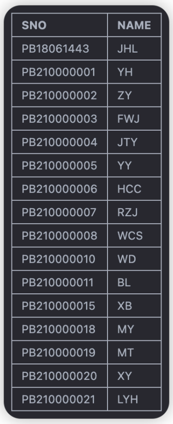
```sql
select distinct t.TNO, t.NAME
from teacher t
join course c on t.TNO = c.TNO
join score s on s.CNO = c.CNO
where s.DEGREE < 72
```



| TNO | NAME |
|---|---|
| TA90023 | ZDH |
| TA90029 | ZR |

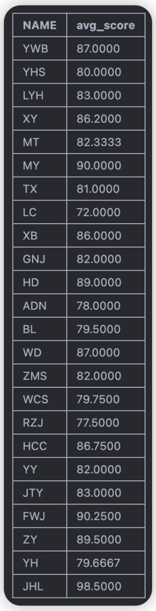
30、查询选修了 2 门课程及以上的学生的学号、姓名。

```sql
select stu.SNO, stu.NAME
from student stu
join score sc on sc.SNO = stu.SNO
group by stu.SNO, stu.NAME
having count(distinct sc.CNO) >= 2;
```



| SNO | NAME |
|---|---|
| PB18061443 | JHL |
| PB210000001 | YH |
| PB210000002 | ZY |
| PB210000003 | FWJ |
| PB210000004 | JTY |
| PB210000005 | YY |
| PB210000006 | HCC |
| PB210000007 | RZJ |
| PB210000008 | WCS |
| PB210000010 | WD |
| PB210000011 | BL |
| PB210000015 | XB |
| PB210000018 | MY |
| PB210000019 | MT |
| PB210000020 | XY |
| PB210000021 | LYH |

31、查询 student 表中各个学生姓名与相应的平均成绩（没有选课的学生平均成绩为 NULL）。

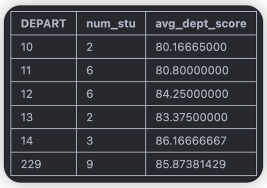


| NAME | avg_score |
|---|---|
| YWB | 87.0000 |
| YHS | 80.0000 |
| LYH | 83.0000 |
| XY | 86.2000 |
| MT | 82.3333 |
| MY | 90.0000 |
| TX | 81.0000 |
| LC | 72.0000 |
| XB | 86.0000 |
| GNJ | 82.0000 |
| HD | 89.0000 |
| ADN | 78.0000 |
| BL | 79.5000 |
| WD | 87.0000 |
| ZMS | 82.0000 |
| WCS | 79.7500 |
| RZJ | 77.5000 |
| HCC | 86.7500 |
| YY | 82.0000 |
| JTY | 83.0000 |
| FWJ | 90.2500 |
| ZY | 89.5000 |
| YH | 79.6667 |
| JHL | 98.5000 |

```sql
select stu.NAME, avg(sc.DEGREE) as avg_score
from student stu
join score sc on sc.SNO = stu.SNO
group by stu.NAME;
```

32、查询每个系的学生人数和每个系的平均分，其中每行包含系号、系的人数和平均成绩。这里平均成绩是指每个学生的所有课程的平均成绩计算后，与同一个系的其他同学再次计算平均值。

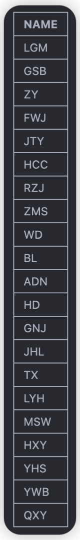
```sql
select stu.DEPART, count(distinct stu.SNO) as num_stu,
    avg(avg_score) as avg_dept_score
from student stu
left join(
    select s.SNO, avg(S.DEGREE) AS avg_score
    from score s
    group by s.SNO
)as avg_dept_score
on stu.SNO = avg_dept_score.SNO
group by stu.DEPART;
```



| DEPART | num_stu | avg_dept_score |
|---|---|---|
| 10 | 2 | 80.16665000 |
| 11 | 6 | 80.80000000 |
| 12 | 6 | 84.25000000 |
| 13 | 2 | 83.37500000 |
| 14 | 3 | 86.16666667 |
| 229 | 9 | 85.87381429 |

33、查询所有未选修 Data_Mining 课程的学生姓名（去掉重复行）。

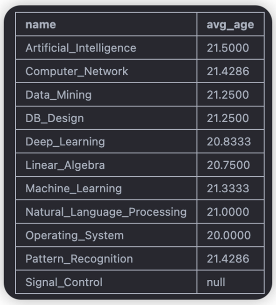
```sql
select distinct student.NAME
from student
where student.sno not in(
    select distinct score.sno
    from score
    where score.cno = (
        select course.cno
        from course
        where course.name = 'Data_Mining'
    )
);
```



| NAME |
|---|
| LGM |
| GSB |
| ZY |
| FWJ |
| JTY |
| HCC |
| RZJ |
| ZMS |
| WD |
| BL |
| ADN |
| HD |
| GNJ |
| JHL |
| TX |
| LYH |
| MSW |
| HXY |
| YHS |
| YWB |
| QXY |

34、查询各个课程的课程名及选该课的学生的平均年龄。（包括没有人选的课）

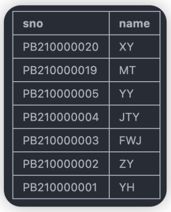
```sql
select course.name, avg(2023 - year(student.birthday)) as
    avg_age
from course
left join score on course.cno = score.cno
left join student on student.sno = score.sno
group by course.name;
```



| name | avg_age |
|---|---|
| Artificial_Intelligence | 21.5000 |
| Computer_Network | 21.4286 |
| Data_Mining | 21.2500 |
| DB_Design | 21.2500 |
| Deep_Learning | 20.8333 |
| Linear_Algebra | 20.7500 |
| Machine_Learning | 21.3333 |
| Natural_Language_Processing | 21.0000 |
| Operating_System | 20.0000 |
| Pattern_Recognition | 21.4286 |
| Signal_Control | null |

35、查询选修了课程名中包含”Computer”课程的学生的学号和姓名。

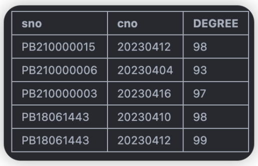
```sql
select student.sno, student.name
from student
where student.sno in (
    select score.sno
    from score
    where score.cno =(
        select course.cno
        from course
        where course.name like '%computer%'
    )
);
```



| sno | name |
|---|---|
| PB210000020 | XY |
| PB210000019 | MT |
| PB210000005 | YY |
| PB210000004 | JTY |
| PB210000003 | FWJ |
| PB210000002 | ZY |
| PB210000001 | YH |

36、查询成绩比该课程平均成绩高 12 分以上的同学的成绩表，即包含 SNO、CNO、DEGREE。

```sql
select score.sno, score.cno, score.DEGREE
from score
inner join (
    select cno, avg(degree) as avg_degree
    from score
    group by cno
) as avgDegree on avgDegree.cno = score.cno
where score.degree > avgDegree.avg_degree + 12;
```

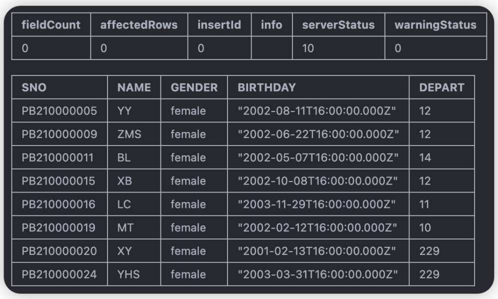


| sno | cno | DEGREE |
|---|---|---|
| PB210000015 | 20230412 | 98 |
| PB210000006 | 20230404 | 93 |
| PB210000003 | 20230416 | 97 |
| PB18061443 | 20230410 | 98 |
| PB18061443 | 20230412 | 99 |

## 2.5 视图

37、建立女学生的学生视图（db_female_student），属性与 student 表一样，并要求对该视图进行修改和插入操作时仍需保证该视图只有女学生。

```sql
select score.sno, score.cno, score.DEGREE
from score
inner join (
    select cno, avg(degree) as avg_degree
    from score
    group by cno
) as avgDegree on avgDegree.cno = score.cno
where score.degree > avgDegree.avg_degree + 12;
```

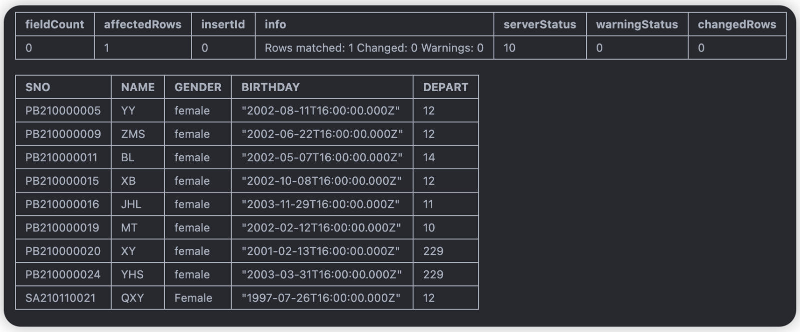


| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 |  | 10 | 0 |

| SNO | NAME | GENDER | BIRTHDAY | DEPART |
|---|---|---|---|---|
| PB210000005 | YY | female | "2002-08-11T16:00:00.000Z" | 12 |
| PB210000009 | ZMS | female | "2002-06-22T16:00:00.000Z" | 12 |
| PB210000011 | BL | female | "2002-05-07T16:00:00.000Z" | 14 |
| PB210000015 | XB | female | "2002-10-08T16:00:00.000Z" | 12 |
| PB210000016 | LC | female | "2003-11-29T16:00:00.000Z" | 11 |
| PB210000019 | MT | female | "2002-02-12T16:00:00.000Z" | 10 |
| PB210000020 | XY | female | "2001-02-13T16:00:00.000Z" | 229 |
| PB210000024 | YHS | female | "2003-03-31T16:00:00.000Z" | 229 |

38、将女学生视图(db_female_student)中学号为“PB210000016”的学生姓名改为 {你的姓名(英文首字母)}。

```sql
update db_female_student
set name = 'JHL'
where sno = 'PB210000016';
select * from db_female_student;
```

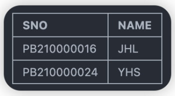


| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus | changedRows |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | Rows matched: 1 Changed: 0 Warnings: 0 | 10 | 0 | 0 |

| SNO | NAME | GENDER | BIRTHDAY | DEPART |
|---|---|---|---|---|
| PB210000005 | YY | female | "2002-08-11T16:00:00.000Z" | 12 |
| PB210000009 | ZMS | female | "2002-06-22T16:00:00.000Z" | 12 |
| PB210000011 | BL | female | "2002-05-07T16:00:00.000Z" | 14 |
| PB210000015 | XB | female | "2002-10-08T16:00:00.000Z" | 12 |
| PB210000016 | JHL | female | "2003-11-29T16:00:00.000Z" | 11 |
| PB210000019 | MT | female | "2002-02-12T16:00:00.000Z" | 10 |
| PB210000020 | XY | female | "2001-02-13T16:00:00.000Z" | 229 |
| PB210000024 | YHS | female | "2003-03-31T16:00:00.000Z" | 229 |
| SA210110021 | QXY | Female | "1997-07-26T16:00:00.000Z" | 12 |

39、在女学生视图（db_female_student）中找出年龄小于 21 岁的学生，包含 SNO、NAME。

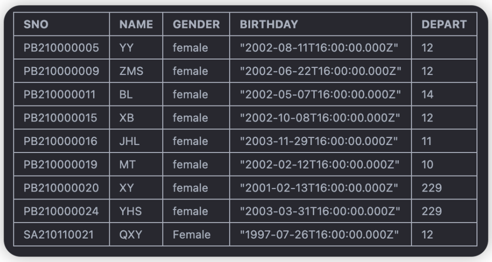
```
select sno, name
from db_female_student
where 2023 - year(birthday) < 21;
```



| SNO | NAME |
|---|---|
| PB210000016 | JHL |
| PB210000024 | YHS |

40、向 student 表中插入一名“学号 SA210110021，姓名 QXY，性别女，生日 1997/7/27，12 系”的学生。然后查询视图 db_female_student 的所有学生，验证其是否更新。

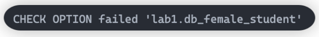
```
insert into student(sno, name, gender, birthday, depart)
value ('SA210110021', 'QXY', 'Female', '1997/7/27', '12');
```



| SNO | NAME | GENDER | BIRTHDAY | DEPART |
|---|---|---|---|---|
| PB210000005 | YY | female | "2002-08-11T16:00:00.000Z" | 12 |
| PB210000009 | ZMS | female | "2002-06-22T16:00:00.000Z" | 12 |
| PB210000011 | BL | female | "2002-05-07T16:00:00.000Z" | 14 |
| PB210000015 | XB | female | "2002-10-08T16:00:00.000Z" | 12 |
| PB210000016 | JHL | female | "2003-11-29T16:00:00.000Z" | 11 |
| PB210000019 | MT | female | "2002-02-12T16:00:00.000Z" | 10 |
| PB210000020 | XY | female | "2001-02-13T16:00:00.000Z" | 229 |
| PB210000024 | YHS | female | "2003-03-31T16:00:00.000Z" | 229 |
| SA210110021 | QXY | Female | "1997-07-26T16:00:00.000Z" | 12 |

41、向视图 db_female_student 中插入一名“学号 SA210110023，姓名 DPC，性别男，生日 1997/4/27，11 系”的学生，观察到了什么现象？

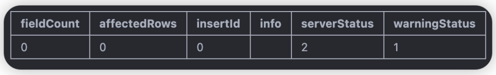
```
insert into db_female_student(sno, name, gender, birthday,
    depart)
value ('SA210110023', 'DPC', 'Male', '1997/4/27', '11');
```



```
CHECK OPTION failed 'lab1.db_female_student'
```

42、删除视图 db_female_student。

```
drop view if exists db_female_student;
```

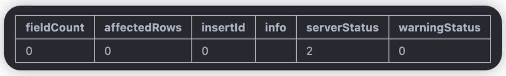


| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 |  | 2 | 1 |

## 2.6 触发器

43、创建关系表：teacher_salary(TNO, SAL)，其中 TNO 是教师工号（主键），SAL 是教师工资（类型 float）。

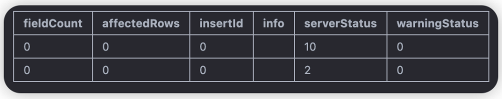
```
create table if not exists teacher_salary(
    tno char(8) primary key,
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 |  | 2 | 0 |

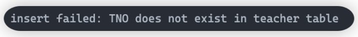
44、定义一个 BEFORE 行级触发器，为关系表 teacher_salary 定义完整性规则：“表中出现的工号必须也出现在 teacher 表中，否则报错”。注：该规则实际上就是外键约束；MySQL 中可使用 SIGNAL 抛出错误；需要为 INSERT 和 UPDATE 分别定义触发器。请展示出成功创建触发器和测试抛出错误信息的截图。

```sql
create trigger if not exists before_teacher_salary_insert
before insert on teacher_salary
for each row
begin
    declare tno_count INT;
    select count(*) into tno_count from teacher where tno
    = new.tno;
    if (tno_count = 0) THEN
        SIGNAL SQLSTATE '45000' set MESSAGE_TEXT = 'insert
    failed: TNO does not exist in teacher table';
    end if;
end;

create trigger if not exists before_teacher_salary_update
before update on teacher_salary
for each row
begin
    declare tno_count INT;
    select count(*) into tno_count from teacher where tno
    = new.tno;
    if (tno_count = 0) THEN
        SIGNAL SQLSTATE '45001' set MESSAGE_TEXT = 'update
    failed: TNO does not exist in teacher table';
    end if;
end;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 |  | 10 | 0 |
| 0 | 0 | 0 |  | 2 | 0 |

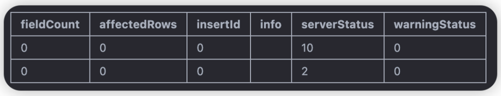
```sql
insert into teacher_salary (tno, sal)
value ('TA11', 1000);
```



```
insert failed: TNO does not exist in teacher table
```

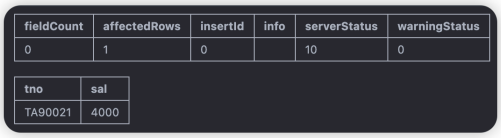
45 、定义一个 BEFORE 行级触发器，为关系表 teacher_salary 定义完整性规则：“Instructor/Associate Professor/Professor 的工资不能低于 4000/5000/6000，如果低于，则改为 4000/5000/6000”。注：需要为 INSERT 和 UPDATE 分别定义触发器。并检验触发器是否工作：为 teacher_salary 构造 INSERT 和 UPDATE 语句并激活所定义过的触发器，将过程截图展示。

```
create trigger before_teacher_salary_insert_sal
before insert on teacher_salary
for each row
begin
    declare posit varchar(30);
    select position into posit from teacher where tno =
    new.tno;
    if (posit = 'instructor' and new.sal < 4000) then
        set new.sal = 4000;
    elseif (posit = 'Associate Professor' and new.sal <
    5000) then
        set new.sal = 5000;
    elseif (posit = 'Professor' and new.sal < 6000) then
        set new.sal = 6000;
    end if;
end;

create trigger before_teacher_salary_update_sal
before update on teacher_salary
for each row
begin
    declare posit varchar(30);
    select position into posit from teacher where tno =
    new.tno;
    if (posit = 'instructor' and new.sal < 4000) then
        set new.sal = 4000;
    elseif (posit = 'Associate Professor' and new.sal <
    5000) then
        set new.sal = 5000;
    elseif (posit = 'Professor' and new.sal < 6000) then
        set new.sal = 6000;
    end if;
end;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 |  | 10 | 0 |
| 0 | 0 | 0 |  | 2 | 0 |

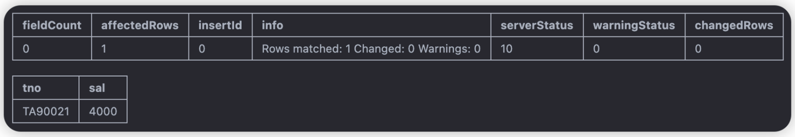
```
insert into teacher_salary (tno, sal)
value ('TA90021', 100);
select * from teacher_salary;
```



| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 1 | 0 |  | 10 | 0 |

| tno | sal |
|---|---|
| TA90021 | 4000 |

```
update teacher_salary
set sal = 100
where TNO = 'TA90021';
```

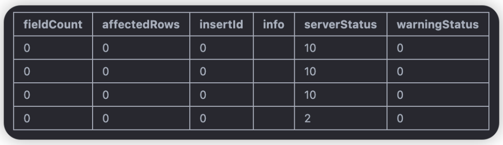


| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus | changedRows |
|---|---|---|---|---|---|---|
| 0 | 1 | 0 | Rows matched: 1 Changed: 0 Warnings: 0 | 10 | 0 | 0 |

| tno | sal |
|---|---|
| TA90021 | 4000 |

46、删除刚刚创建的所有触发器。

```
drop trigger if exists before_teacher_salary_insert;
drop trigger if exists before_teacher_salary_update;
drop trigger if exists before_teacher_salary_insert_sal;
drop trigger if exists before_teacher_salary_update_sal;
```

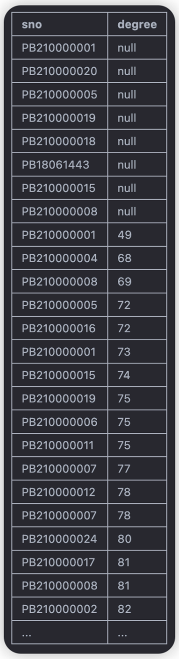


| fieldCount | affectedRows | insertId | info | serverStatus | warningStatus |
|---|---|---|---|---|---|
| 0 | 0 | 0 | | 10 | 0 |
| 0 | 0 | 0 | | 10 | 0 |
| 0 | 0 | 0 | | 10 | 0 |
| 0 | 0 | 0 | | 2 | 0 |

## 2.7 空值

47、将 score 表中的 Data_Mining 课程成绩设为空值，然后在 score 表查询学生学号和分数，并按分数升序展示。观察 NULL 在 MySQL 中的大小是怎样的?

```
update score
set degree = null
where CNO = (
    select cno
    from course
    where NAME = 'data_mining'
);

select sno, degree
from score
order by degree asc;
```

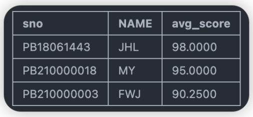


| sno | degree |
|---|---|
| PB210000001 | null |
| PB210000020 | null |
| PB210000005 | null |
| PB210000019 | null |
| PB210000018 | null |
| PB18061443 | null |
| PB210000015 | null |
| PB210000008 | null |
| PB210000001 | 49 |
| PB210000004 | 68 |
| PB210000008 | 69 |
| PB210000005 | 72 |
| PB210000016 | 72 |
| PB210000001 | 73 |
| PB210000015 | 74 |
| PB210000019 | 75 |
| PB210000006 | 75 |
| PB210000011 | 75 |
| PB210000007 | 77 |
| PB210000012 | 78 |
| PB210000007 | 78 |
| PB210000024 | 80 |
| PB210000017 | 81 |
| PB210000008 | 81 |
| PB210000002 | 82 |
| ... | ... |

## 2.8 开放题

48、查询选修了两门及以上课程的学生中，选课平均成绩最高的前三名学生的学号、姓名和平均成绩。

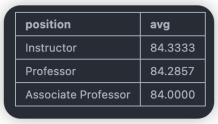
```
select student.sno, student.NAME, avg(score.degree) as
    avg_score
from student
join score on student.sno = score.sno
where student.sno in (
    select sno
    from score
    group by sno
    having count(*) >= 2
)
group by student.sno
order by avg_score desc
limit 3;
```



| sno | NAME | avg_score |
|---|---|---|
| PB18061443 | JHL | 98.0000 |
| PB210000018 | MY | 95.0000 |
| PB210000003 | FWJ | 90.2500 |

49、查询每个职位给的平均成绩。

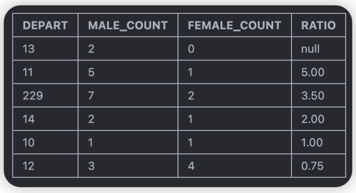
```
select teacher.position, avg(score.degree) as avg
from teacher
join course on teacher.tno = course.tno
join score on score.cno = course.cno
group by teacher.position
order by avg desc;
```



| position | avg |
|---|---|
| Instructor | 84.3333 |
| Professor | 84.2857 |
| Associate Professor | 84.0000 |

50、查询每个系的男女比，并降序排列。

```
SELECT DEPART,
    SUM(CASE WHEN GENDER = 'male' THEN 1 ELSE 0 END) AS
    MALE_COUNT,
    SUM(CASE WHEN GENDER = 'female' THEN 1 ELSE 0 END) AS
    FEMALE_COUNT,
    ROUND(SUM(CASE WHEN GENDER = 'male' THEN 1 ELSE 0 END)
    / SUM(CASE WHEN GENDER = 'female' THEN 1 ELSE 0 END),
    2) AS RATIO
FROM student
GROUP BY DEPART
ORDER BY RATIO IS NULL DESC, RATIO DESC;
```



| DEPART | MALE_COUNT | FEMALE_COUNT | RATIO |
|---|---|---|---|
| 13 | 2 | 0 | null |
| 11 | 5 | 1 | 5.00 |
| 229 | 7 | 2 | 3.50 |
| 14 | 2 | 1 | 2.00 |
| 10 | 1 | 1 | 1.00 |
| 12 | 3 | 4 | 0.75 |