3、表的约束+CURD进阶

笔记本:	Java17&18班上课笔记		
创建时间:	2021/3/10 17:55	**更新时间:**	2021/3/10 22:08
作者:	1262913815@qq.com		

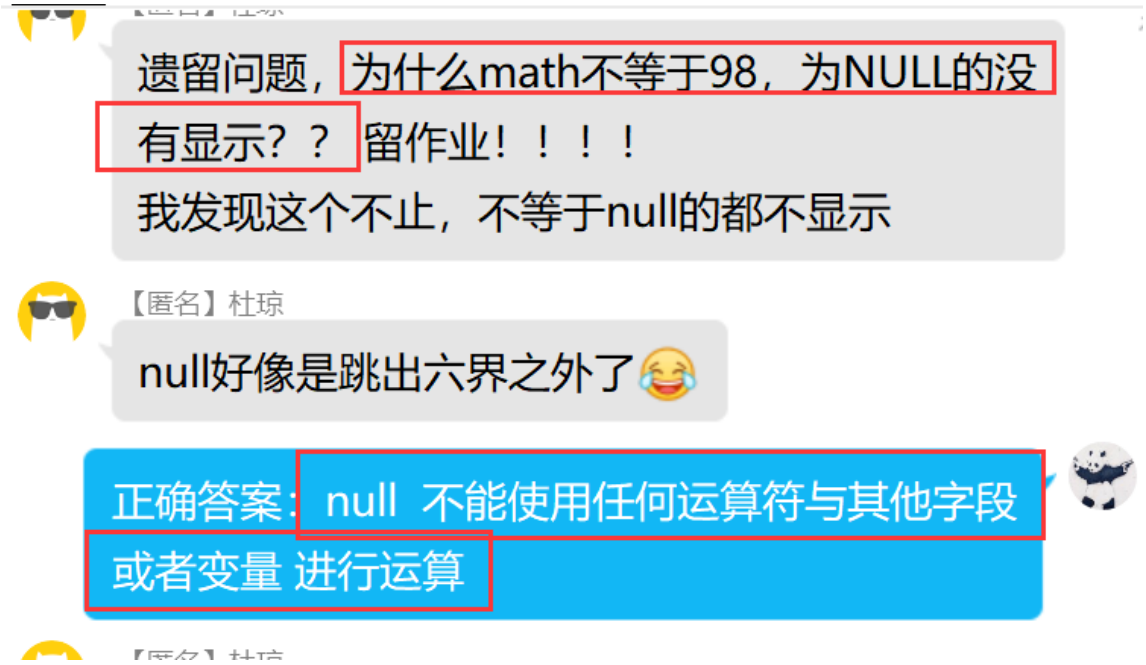
CURD进阶

回顾:

1、插入:
-- 全列+单行插入
insert into user value(1,'bit');

-- 指定列+多行插入:
insert into user(id,name) values(1,'bit1'),(2,'bit3');

2、查询:



- 删除

-- 删除孙悟空同学的考试成绩

```
mysql> select * from exam_result;
```

```
+------+--------+---------+-------+---------+
| id   | name   | chinese | math  | english |
+------+--------+---------+-------+---------+
|    1 | 唐三藏 |    67.0 |  98.0 |    56.0 |
|    2 | 孙悟空 |    87.5 |  78.0 |    77.0 |
|    3 | 猪悟能 |    88.0 |  98.0 |    90.0 |
|    4 | 曹孟德 |    70.0 |  60.0 |    67.0 |
|    5 | 刘玄德 |    55.5 |  25.0 |    45.0 |
|    6 | 孙权   |    70.0 |  99.0 |    78.5 |
|    7 | 宋公明 |    75.0 |   5.0 |    30.0 |
|    8 | 唐三藏 |    67.0 |  NULL |    80.0 |
|    9 | 唐三藏 |    67.0 | -31.0 |    80.0 |
|   10 | 张三丰 |    67.0 |  41.0 |    80.0 |
+------+--------+---------+-------+---------+
10 rows in set (0.01 sec)
```

```
mysql> delete from exam_result where name = '孙悟空';
Query OK, 1 row affected (0.01 sec)

mysql> select * from exam_result;
+------+--------+---------+-------+---------+
| id   | name   | chinese | math  | english |
+------+--------+---------+-------+---------+
|    1 | 唐三藏 |    67.0 |  98.0 |    56.0 |
|    3 | 猪悟能 |    88.0 |  98.0 |    90.0 |
|    4 | 曹孟德 |    70.0 |  60.0 |    67.0 |
|    5 | 刘玄德 |    55.5 |  25.0 |    45.0 |
|    6 | 孙权   |    70.0 |  99.0 |    78.5 |
|    7 | 宋公明 |    75.0 |   5.0 |    30.0 |
|    8 | 唐三藏 |    67.0 |  NULL |    80.0 |
|    9 | 唐三藏 |    67.0 | -31.0 |    80.0 |
|   10 | 张三丰 |    67.0 |  41.0 |    80.0 |
+------+--------+---------+-------+---------+
9 rows in set (0.00 sec)
```

只要是姓名符合，都将被删除。

```
mysql> delete from exam_result where name = '唐三藏';
Query OK, 3 rows affected (0.00 sec)

mysql> select * from exam_result;
+------+--------+---------+------+---------+
| id   | name   | chinese | math | english |
+------+--------+---------+------+---------+
|    3 | 猪悟能 |    88.0 | 98.0 |    90.0 |
|    4 | 曹孟德 |    70.0 | 60.0 |    67.0 |
```

```
|    5 | 刘玄德 |    55.5 | 25.0 |    45.0 |
|    6 | 孙权   |    70.0 | 99.0 |    78.5 |
|    7 | 宋公明 |    75.0 |  5.0 |    30.0 |
|   10 | 张三丰 |    67.0 | 41.0 |    80.0 |
+------+--------+---------+------+---------+
6 rows in set (0.00 sec)
```

直接删除表：会删除表内的所有的数据，但是，这张表还是存在的，只不过里面没有数据了！！

```
mysql> delete from exam_result;
Query OK, 6 rows affected (0.01 sec)

mysql> show tables;
+-------------------------+
| Tables_in_testjava17_18 |
+-------------------------+
| exam_result             |
| stu_test                |
+-------------------------+
2 rows in set (0.00 sec)

mysql> select * from exam_result;
Empty set (0.00 sec)
```

删除表本身：

```
mysql> drop table exam_result;
Query OK, 0 rows affected (0.06 sec)

mysql> show tables;
+-------------------------+
| Tables_in_testjava17_18 |
+-------------------------+
| stu_test                |
+-------------------------+
1 row in set (0.00 sec)
```

- 数据库的约束（表中的字段）

1、NULL约束

```
drop table if exists student;
create table student(
    id int,
    sn int,
    name varchar(20) NOT NULL,
```

```
    qq_mail varchar(20)
);
```

insert into student(id,sn,name,qq_mail)
values(1,101,NULL,'123@qq.com');

```
mysql> insert into student(id,sn,name,qq_mail)
values(1,101,NULL,'123@qq.com');
ERROR 1048 (23000): Column 'name' cannot be null
mysql> desc student;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| id      | int(11)     | YES  |     | NULL    |       |
| sn      | int(11)     | YES  |     | NULL    |       |
| name    | varchar(20) | NO   |     | NULL    |       |
| qq_mail | varchar(20) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

2、UNIQUE 唯一约束

```
drop table if exists student;
create table student(
    id int,
    sn int unique,
    name varchar(20) NOT NULL,
    qq_mail varchar(20)
);
```

```
mysql> insert into student(id,sn,name,qq_mail)
values(1,101,'bit','123@qq.com');
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+------+------+------+------------+
| id   | sn   | name | qq_mail    |
+------+------+------+------------+
|    1 |  101 | bit  | 123@qq.com |
+------+------+------+------------+
1 row in set (0.00 sec)

mysql> insert into student(id,sn,name,qq_mail)
values(2,101,'bit2','1232@qq.com');
ERROR 1062 (23000): Duplicate entry '101' for key 'sn'
```

3、DEFAULT 默认值约束：

```
drop table if exists student;
create table student(
    id int,
    sn int unique,
    name varchar(20) NOT NULL,
    qq_mail varchar(20) DEFAULT '110@qq.com'
);
```

进行插入：

```
mysql> insert into student(id,sn,name,qq_mail)
values(1,101,'bit','123@qq.com');
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+------+------+------+------------+
| id   | sn   | name | qq_mail    |
+------+------+------+------------+
|    1 |  101 | bit  | 123@qq.com |
+------+------+------+------------+
1 row in set (0.00 sec)

mysql> insert into student(id,sn,name,qq_mail)
values(2,102,'bit2',NULL);
Query OK, 1 row affected (0.00 sec)

mysql> select * from student;
+------+------+------+------------+
| id   | sn   | name | qq_mail    |
+------+------+------+------------+
|    1 |  101 | bit  | 123@qq.com |
|    2 |  102 | bit2 | NULL       |
+------+------+------+------------+
2 rows in set (0.00 sec)
```

到底怎么体现默认值呢?

```
mysql> insert into student(id,sn,name)
values(3,103,'bit3');  在此时只要不指定列，不给qq_mail任何值。
Query OK, 1 row affected (0.00 sec)

mysql> select * from student;
+------+------+------+------------+
| id   | sn   | name | qq_mail    |
```

```
+------+------+------+------------+
|    1 |  101 | bit  | 123@qq.com |
|    2 |  102 | bit2 | NULL       |
|    3 |  103 | bit3 | 110@qq.com |
+------+------+------+------------+
3 rows in set (0.00 sec)
```

4、primary key 主键约束：她是NOT NULL 和 UNIQUE的结合。也就是说，当一个字段被PRIMARY KEY 修饰后，那么这个字段就是不能为空且是独一无二的！！！
因为是独一无二的，所以，一般搭配：auto_increment;

创建表：

```
drop table if exists student;
create table student(
    id int PRIMARY KEY AUTO_INCREMENT,
    sn int unique,
    name varchar(20) NOT NULL,
    qq_mail varchar(20) DEFAULT '110@qq.com'
);
```

当创建好表之后，表中没有任何的数据，当第一次执行插入的时候，当前主键，也就是ID，会自动从1开始。

```
mysql> insert into student(sn,name,qq_mail)
values(101,'bit','123@qq.com');
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+----+------+------+------------+
| id | sn   | name | qq_mail    |
+----+------+------+------------+
|  1 |  101 | bit  | 123@qq.com |
+----+------+------+------------+
1 row in set (0.01 sec)
```

当我将刚刚插入的数据删除后，再次进行插入的时候，就会在原来的基础，也就是上一次最后插入的语句的ID上开始加1；

```
mysql> delete from student where id = 1;
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
```

```
Empty set (0.00 sec)

mysql> insert into student(sn,name,qq_mail)
values(101,'bit','123@qq.com');
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+----+------+------+------------+
| id | sn   | name | qq_mail    |
+----+------+------+------------+
|  2 |  101 | bit  | 123@qq.com |
+----+------+------+------------+
1 row in set (0.00 sec)
```

只有当你把整个表都删除了，那么你再次插入的时候，才是从1开始的！！！

问题：

id int

id long

alter 可以修改字段的类型

但是一般在工作当中，创建项目的时候，立项的时候。这些东西都要考虑到。

5、 FOREIGN KEY 外键约束

外键用于关联其他表的主键或唯一键，语法：foreign key (字段名) references 主表(列)

创建表：

```
drop table if exists classes;
create table classes(
    id int PRIMARY KEY AUTO_INCREMENT,
    name varchar(20),
    `desc` varchar(30)
);

drop table if exists student;
create table student(
    id int PRIMARY KEY AUTO_INCREMENT,
    sn int unique,
    name varchar(20) NOT NULL,
```

```
    qq_mail varchar(20) DEFAULT '110@qq.com'
    classes_id int,
    FOREIGN KEY (classes_id) REFERENCES classes(id)
);
```

classes_id就是外键，关联的是classes这张表的id字段
这样做：安全！！！

现在我们有2张表，一张班级表，一张学生表。
0、建表：
先创建主表！！

```
mysql> create table student(
    -> id int PRIMARY KEY AUTO_INCREMENT,
    -> sn int unique,
    -> name varchar(20) NOT NULL,
    -> qq_mail varchar(20) DEFAULT '110@qq.com',
    -> classes_id int,
    -> FOREIGN KEY (classes_id) REFERENCES classes(id)
    -> );
ERROR 1215 (HY000): Cannot add foreign key constraint
```

1、插入数据：
问题：请问应该先插入哪个表？？？

```
mysql> insert into
student(sn,name,qq_mail,classes_id)values(101,'bit','123@qq.com',1);
ERROR 1452 (23000): Cannot add or update a child row: a
foreign key constraint fails (`testjava17_18`.`student`,
CONSTRAINT `student_ibfk_1` FOREIGN KEY (`classes_id`)
REFERENCES `classes` (`id`))
```

答案是：先插入主表！！
mysql> insert into classes(name,`desc`) values ('Java17班','今天上SQL')
  -> ;
Query OK, 1 row affected (0.01 sec)

mysql> select * from classes;
+----+---------+----------+
| id | name | desc |
+----+---------+----------+
| 1 | Java17班 | 今天上SQL |
+----+---------+----------+
1 row in set (0.00 sec)

mysql> insert into student(sn,name,qq_mail,classes_id)values(101,'bit','123@qq.com',1);
Query OK, 1 row affected (0.01 sec)

2、删除情况：
先删除哪张表的数据？？？ 删除的是子表是可以的。但是，删除主表也是可以的，前提是主表当中的这个id,,没有被关联。
如果被关联了，那么就会失败！！

子表数据：

```
mysql> select * from student;
+----+------+------+------------+------------+
| id | sn   | name | qq_mail    | classes_id |
+----+------+------+------------+------------+
|  3 |  101 | bit  | 123@qq.com |          1 |
+----+------+------+------------+------------+
```

主表数据：

```
mysql> select * from classes;
+----+----------+------------+
| id | name     | desc       |
+----+----------+------------+
|  1 | Java17班 | 今天上SQL  |
|  3 | Java17班 | 今天上SQL  |
|  4 | Java18班 | 今天上SQL2 |
+----+----------+------------+
```

删除：

```
mysql> delete from classes where id = 3;
Query OK, 1 row affected (0.01 sec)
```

原因是，id为3的这个班级，没有被关联。

```
mysql> delete from classes where id = 1;
ERROR 1451 (23000): Cannot delete or update a parent row:
a foreign key constraint fails
(`testjava17_18`.`student`, CONSTRAINT `student_ibfk_1`
FOREIGN KEY (`classes_id`) REFERENCES `classes` (`id`))
mysql>
```

因为id=1的这个班级是被关联的，所以不可以删除。

什么时候，才能删除id为1的这个班级呢？？只有当学生表当中的数据，没有人关联这个id

```
mysql> select * from student;
+----+------+------+------------+------------+
| id | sn   | name | qq_mail    | classes_id |
+----+------+------+------------+------------+
|  3 |  101 | bit  | 123@qq.com |          1 |
+----+------+------+------------+------------+
1 row in set (0.00 sec)

mysql> delete from student where id = 3; -- 此时没有人关联classes表当中的id 为1的班级了
Query OK, 1 row affected (0.01 sec)

mysql> delete from classes where id = 1; -- 就可以删除了
Query OK, 1 row affected (0.01 sec)
```

6、CHECK约束

```
drop table if exists test_user;
create table test_user (
    id int,
    name varchar(20),
    sex varchar(1),
    check (sex ='男' or sex='女')
);
```

将来在插入数据的时候，只能插入sex为男或者女。

```
mysql> insert into test_user values(1,'bit','哈');
Query OK, 1 row affected (0.01 sec)
```

此时MySQL并不会报错。

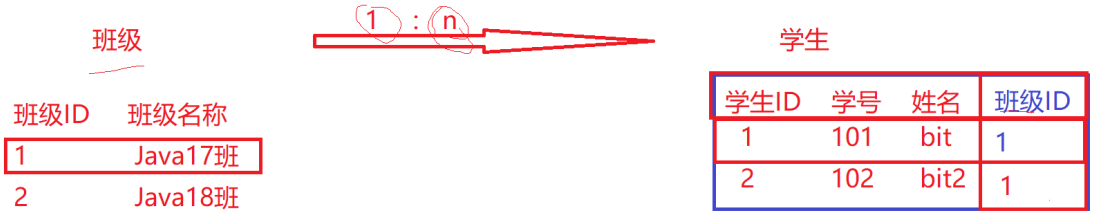
三大范式：规则。了解一下。

以后设计表的时候，会有以下3种情况：

1、一对一

name	身份证号

2、一对多



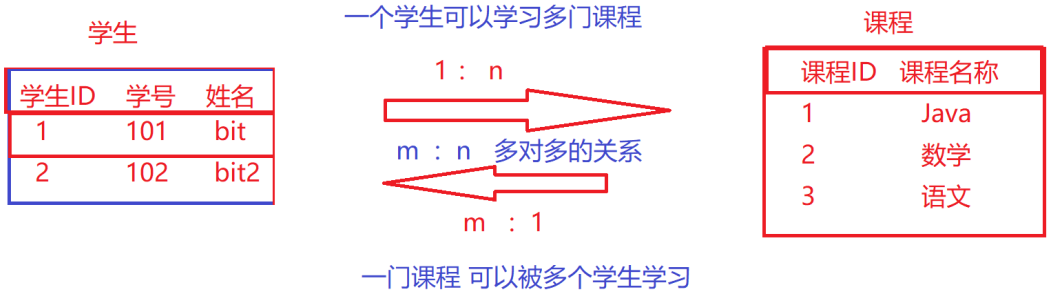
多的那边 存入1的这边的ID。

3、多对多

学生 和 课程的关系：

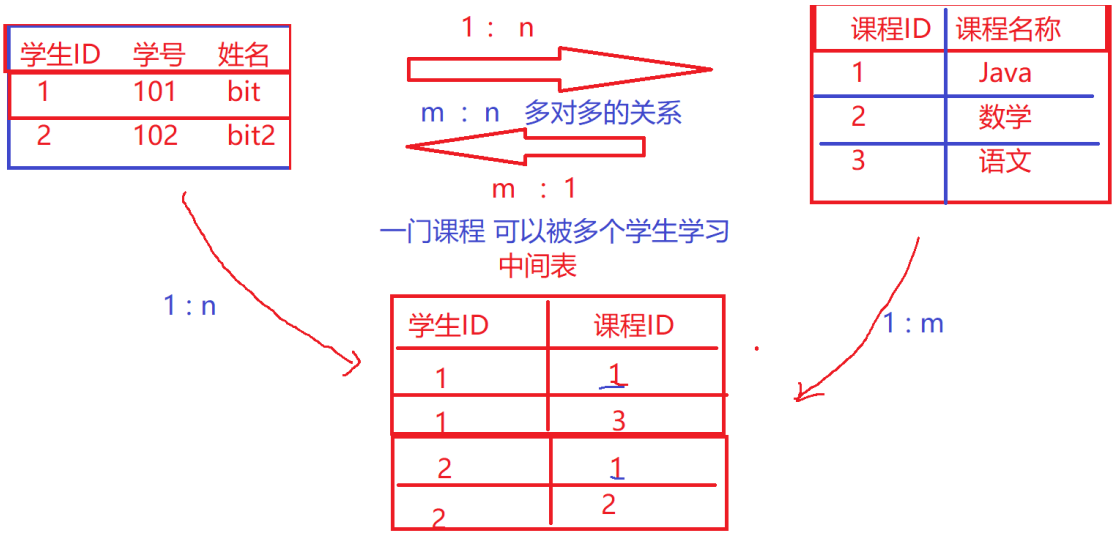
1个学生 可以学习多门课程。

1个课程 可以被多个学生学习。



有一个问题：不管在那张表里面写另一张表的ID 都是不合适的。

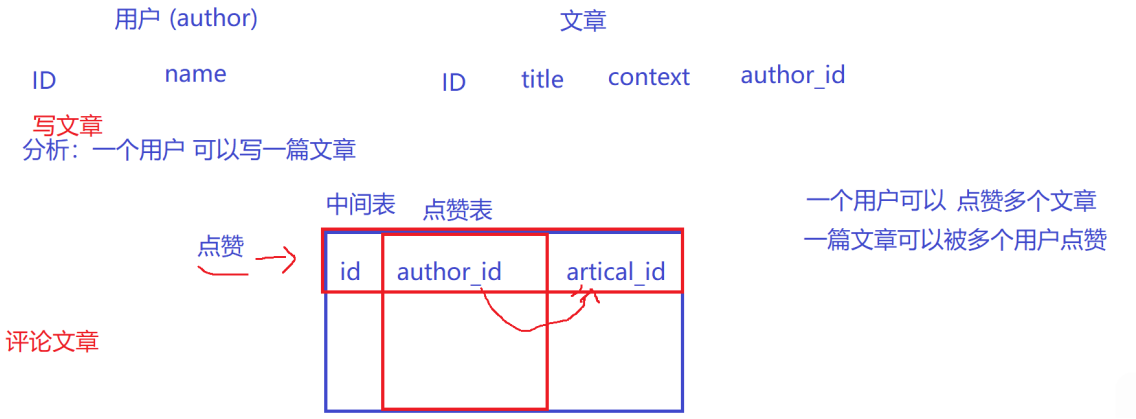
所以有了中间表。来处理这个**对应关系。**



设计一个博客系统：

1、写文章

2、点赞文章



- 进阶查询：

准备数据：

```
mysql> select * from exam_result;
+------+--------+---------+------+---------+
| id   | name   | chinese | math | english |
+------+--------+---------+------+---------+
|    1 | 唐三藏 |    67.0 | 98.0 |    56.0 |
|    2 | 孙悟空 |    87.5 | 78.0 |    77.0 |
|    3 | 猪悟能 |    88.0 | 98.0 |    90.0 |
|    4 | 曹孟德 |    82.0 | 84.0 |    67.0 |
|    5 | 刘玄德 |    55.5 | 85.0 |    45.0 |
|    6 | 孙权   |    70.0 | 73.0 |    78.5 |
|    7 | 宋公明 |    75.0 | 65.0 |    30.0 |
+------+--------+---------+------+---------+
7 rows in set (0.01 sec)
```

1、聚合函数

```
mysql> select count(*) from exam_result;
+----------+
| count(*) |
+----------+
|        7 |
+----------+
1 row in set (0.00 sec)

mysql> select count(0) from exam_result;
+----------+
| count(0) |
+----------+
|        7 |
+----------+
1 row in set (0.00 sec)

mysql> select count(1) from exam_result;
+----------+
```

```
| count(1) |
+----------+
|        7 |
+----------+
1 row in set (0.00 sec)
```

可以对计数的列进行去重：

```
mysql> select * from exam_result;
+------+--------+---------+------+---------+
| id   | name   | chinese | math | english |
+------+--------+---------+------+---------+
|    1 | 唐三藏 |    67.0 | 98.0 |    56.0 |
|    2 | 孙悟空 |    87.5 | 78.0 |    77.0 |
|    3 | 猪悟能 |    88.0 | 98.0 |    90.0 |
|    4 | 曹孟德 |    82.0 | 84.0 |    67.0 |
|    5 | 刘玄德 |    55.5 | 85.0 |    45.0 |
|    6 | 孙权   |    70.0 | 73.0 |    78.5 |
|    7 | 宋公明 |    75.0 | 65.0 |    30.0 |
|    1 | 唐三藏 |    67.0 | 98.0 |    56.0 |
+------+--------+---------+------+---------+
8 rows in set (0.00 sec)
```

```
mysql> select count(distinct id) from exam_result;
+---------------------+
| count(distinct id) |
+---------------------+
|                   7 |
+---------------------+
1 row in set (0.01 sec)
```

SUM:

```
mysql> select sum(math) from exam_result;
+-----------+
| sum(math) |
+-----------+
|     679.0 |
+-----------+
1 row in set (0.00 sec)

mysql> select sum(math) from exam_result where math < 70;
+-----------+
| sum(math) |
+-----------+
|      65.0 |
```

```
+-----------+
1 row in set (0.00 sec)

mysql> select sum(math) from exam_result where math < 80;
+-----------+
| sum(math) |
+-----------+
|     216.0 |
+-----------+
1 row in set (0.00 sec)
```

AVG:

```
mysql> select AVG(math) from exam_result;
+-----------+
| AVG(math) |
+-----------+
|  84.87500 |
+-----------+
1 row in set (0.00 sec)
```

MAX:

```
mysql> select MAX(math) from exam_result;
+-----------+
| MAX(math) |
+-----------+
|      98.0 |
+-----------+
1 row in set (0.00 sec)
```

注意点:**在where后面，不要出现聚合函数。**

```
mysql> select id,name from exam_result where MAX(math) >
80 ;
ERROR 1111 (HY000): Invalid use of group function
mysql>
```

MIN: 查找数学成绩大于70分的，最低分。

```
mysql> select MIN(math) from exam_result where math > 70;
+-----------+
| MIN(math) |
+-----------+
|      73.0 |
```

```
+-----------+
1 row in set (0.00 sec)
```