SQL查询

SQL语句中对于程序员来讲最重要的就是DML，而在DML语句当中最常用的语句就是增删改查，而其中增删改基本上是千篇一律，唯独查询千变万化

查询语句的语法规范：

```
select 查询列 from 查询表

where 查询条件
group by 分组依据 having 分组条件
order by 排序列
limit 控制查询结果的数量
```

举例：创建实验用表

```
create table student(
    sid int primary key auto_increment,
    sname varchar(10) not null,
    cno varchar(20) not null,
    phone char(11) unique key not null,
    score float not null
);

insert into student(sname,cno,phone,score) values
('李四','2','13111111111',78.7),
('lily','2','13111111112',81.2),
('james','1','13111111113',64.1),
('lucy','2','13111111114',91)
```

1、基础查询

```
#查询所有数据
select * from student;
#查询指定列
select sname,score from student;
#查询所有学生姓名
select sname from student;
#查询学生姓名、手机号（前缀拼接）
select sname,concat('+86-',phone) from student
#同级数据的总行数
select count(*) from student;
```

```
#查询最高分
select max(score) from student;
#查询最低分
select min(score) from student;
#查询平均分]
select avg(score) from student;
#求所有学生分数总和
select sum(score) from student;
```

2、条件查询

```
#查询所有分数在80分以上的学生信息
select * from student where score >= 80;
#查询区间值（查询分数在70-90之间的学生信息）
select * from student where score >= 70 and score <= 90;
select * from student where score between 70 and 90;
#查询james、lily、lucy三名学生的成绩和姓名
select sname,score from student where sname='james' or
sname='lily' or sname='lucy';
select sname,score from student where sname='james' ||
sname='lily' || sname='lucy';
select sname,score from student where sname
in('james','lily','lucy');
#查询所有分数不等于80的学生信息
select * from student where score != 80;
select * from student where score <> 80;
#查询所有名字以‘l’开头的学生信息
select * from student where sname like 'l%';
#查询所有名字以‘j’开头同时后面跟了4个字符的学生信息
select * from student where sname like 'j____';
#查询所有名字中包含‘e’的学生信息
select * from student where sname like '%e%';
#查询名字不是‘j’开头的学生信息
select * from student where sname not like 'j%';
```

> 模糊查询：
>
> % 是一个占位符，标识任意字符长度
>
> _ 是一个占位符，只标识一个字符长度

3、分组、排序、分页

```
/* 分组 */
#查询每个班的平均成绩和班级编号
select cno,avg(score) from student group by cno;
#查询每个班的总人数
select count(cno) '人数',cno from student group by cno;
#查询总人数超过4个人的班级的总人数
select count(cno) '人数',cno from student group by cno having
count(cno) > 4;

/* 排序 */
#查询所有的学生成绩，并升序显示(默认情况下就是升序，由低到高，asc)
select * from student order by score;
#查询所有的学生成绩，降序排列
select * from student order by score desc;
#查询每个班的平均分并降序排列
select cno,avg(score) from student group by cno order by
avg(score) desc;

/* 分页 */
#查询前3条数据
#查询分数排名前三的学生信息
#查询第二页数据，每页显示3条
```

4、去除重复行

在select语句中使用关键distinct去除重复行

```
select distinct cno from student;
```

> 注意事项：
>
> distinct 需要放在所有列的前面，如果夹在中间会报错

5、空值参与运算

所有运算符与列值运算时如果遇到null，运算结果都是null

> 这里一定注意：
>
> 在mysql里面，空值是不等于空字符串，一个空字符串的长度是0，而一个空值的长度是空，所以在mysql当中空值是占空间的

6、着重号

```
create table `table`(
)
```

我们需要保证表中的字段名，表名没有和保留字关键字、系统中常用方法名冲突，如果真的相同，在sql语句中可以使用 `` 着重号框起来

7、列的别名

```
select sname as '学生姓名' from student;
```

重命名一个列，方便计算，使用as关键字，可以省略不写

8、多表查询

多表查询也叫关联查询，一般指的是两张或两张以上的多张表一起完成查询操作

前提条件：

要实现多表查询，表与表之间需要形成关系，他们之间有关联的字段，这个关联字段可以建立外键，比如，员工表和部门表，这两个表依靠部门编号进行关联

```
#案例：查询员工姓名和部门名称
select ename,dname from emp,dept;
```

> 结果分析：
>
> 我们发现返回的结果并不是我们想要的一个结果，简单来说这里做了一个两列查询，结果返回了一个多重组合，这种情况我们称之为**笛卡尔积的错误**

9、笛卡尔积（交叉链接）的理解

笛卡尔积是一个数学运算，假设我们现在有两个集合x和y，那么x和y的笛卡尔积就是x和y的所有可能组合，组合的个数即为两个集合种元素个数的乘积

笛卡尔积的错误会在以下条件种产生：

- 省略了多个表的关联条件
- 关联条件无效

- 所有表种的所有行相互关联

为了避免笛卡尔积，可以在where后面加入有效的关联条件

语法：

```
select table1.column,table2.column
from table1,table2
where table1.column = table2.cloumn
```

> 说明：
>
> table1和table2表示你需要查询的表

把上面的例子做修改

```
SELECT ename,dname
FROM    emp,dept
WHERE   emp.deptno = dept.deptno;
```

还可以简化用别名

```
SELECT ename,dname
FROM    emp e,dept d
WHERE   e.deptno = d.deptno;
```