**https://www.cnblogs.com/skins/bj2008/images/fire.gif**[**MYSQL查询操作 详细**](https://www.cnblogs.com/liuguangbiao/p/9259889.html)

**学习目标**

1 掌握select查询所有字段、指定字段的数据

2 掌握消除重复行命令distinct

3 掌握as给字段、表起别名

4 掌握条件查询where后跟比较运算符、逻辑运算符的用法

5 掌握条件查询中的模糊查询like和范围查询in

6 掌握空判断is null和非空判断is not null

7 掌握查询中使用order by排序

8 掌握常用到的聚合函数count、max、min、sum、avg

9 掌握分组查询group by + group\_concat(字段名)/聚合函数/having

10 掌握分页查询获取部分行的命令limit

11 掌握连接查询inner/left/right join

12 掌握子查询的方法

ps：本文章很长请收藏以便学习。

**基本查询**

**1. 创建数据库、数据表**

-- 创建数据库

create database python\_test\_1 charset=utf8;

-- 使用数据库

use python\_test\_1;

-- students表

[复制代码](javascript:void(0);)

create table students(

id int unsigned primary key auto\_increment not null,

name varchar(20) default '',

age tinyint unsigned default 0,

height decimal(5,2),

gender enum('男','女','中性','保密') default '保密',

cls\_id int unsigned default 0,

is\_delete bit default 0

);

[复制代码](javascript:void(0);)

-- classes表

create table classes (

id int unsigned auto\_increment primary key not null,

name varchar(30) not null

);

**2. 准备数据**

-- 向students表中插入数据

[复制代码](javascript:void(0);)

insert into students values

(0,'小明',18,180.00,2,1,0),

(0,'小月月',18,180.00,2,2,1),

(0,'彭于晏',29,185.00,1,1,0),

(0,'刘德华',59,175.00,1,2,1),

(0,'黄蓉',38,160.00,2,1,0),

(0,'凤姐',28,150.00,4,2,1),

(0,'王祖贤',18,172.00,2,1,1),

(0,'周杰伦',36,NULL,1,1,0),

(0,'程坤',27,181.00,1,2,0),

(0,'刘亦菲',25,166.00,2,2,0),

(0,'金星',33,162.00,3,3,1),

(0,'静香',12,180.00,2,4,0),

(0,'郭靖',12,170.00,1,4,0),

(0,'周杰',34,176.00,2,5,0);

[复制代码](javascript:void(0);)

-- 向classes表中插入数据

insert into classes values (0, "python\_01期"), (0, "python\_02期");

**3. 查询练习**

* 查询所有字段

select \* from 表名;

例：

select \* from students;

* 查询指定字段

select 列1,列2,... from 表名;

例:

select name from students;

* 使用 as 给字段起别名

select id as 序号, name as 名字, gender as 性别 from students;

* 使用 as 给表起别名

-- 如果是单表查询 可以省略表明

select id, name, gender from students;

-- 表名.字段名

select students.id,students.name,students.gender from students;

-- 可以通过 as 给表起别名

select s.id,s.name,s.gender from students as s;

* 消除重复行

在select后面列前使用distinct可以消除重复的行

select distinct 列1,... from 表名;

例：

select distinct gender from students;

**条件查询**

使用where子句对表中的数据筛选，结果为true的行会出现在结果集中

* 语法如下：

select \* from 表名 where 条件;

例：

select \* from students where id=1;

* where后面支持多种运算符，进行条件的处理
  + 比较运算符
  + 逻辑运算符
  + 模糊查询
  + 范围查询
  + 空判断

**1. 比较运算符**

* 等于: =
* 大于: >
* 大于等于: >=
* 小于: <
* 小于等于: <=
* 不等于: != 或 <>

例1：查询编号大于3的学生

select \* from students where id > 3;

例2：查询编号不大于4的学生

select \* from students where id <= 4;

例3：查询姓名不是“黄蓉”的学生

select \* from students where name != '黄蓉';

例4：查询没被删除的学生

select \* from students where is\_delete=0;

**2. 逻辑运算符**

* and
* or
* not

例5：查询编号大于3的女同学

select \* from students where id > 3 and gender=0;

例6：查询编号小于4或没被删除的学生

select \* from students where id < 4 or is\_delete=0;

**3. 模糊查询**

* like

%表示0到多个任意字符 , \_表示一个任意字符

例7：查询姓黄的学生

select \* from students where name like '黄%';

例8：查询姓黄并且“名”是一个字的学生

select \* from students where name like '黄\_';

例9：查询姓黄或叫靖的学生

select \* from students where name like '黄%' or name like '%靖';

**4. 范围查询**

* in (表示在一个非连续的范围内)

例10：查询编号是1或3或8的学生

select \* from students where id in(1,3,8);

* between ... and ...表示在一个连续的范围内

例11：查询编号为3至8的学生

select \* from students where id between 3 and 8;

例12：查询编号是3至8的男生

select \* from students where (id between 3 and 8) and gender=1;

**5. 空判断**

* is null (判断是否为空)

注意：null与''是不同的

例13：查询没有填写身高的学生

select \* from students where height is null;

* is not null (判非空)

例14：查询填写了身高的学生

select \* from students where height is not null;

例15：查询填写了身高的男生

select \* from students where height is not null and gender=1;

**优先级**

* 优先级由高到低的顺序为：小括号比较运算符，逻辑运算符
* not>and>or，如果同时出现并希望先算or，需要结合()使用

select \* from students where not age>=18 or height>=180 and gender="男";

**排序**

为了方便查看数据，可以对数据进行排序

**语法：**

select \* from 表名 [where ...] order by 列1 asc|desc [,列2 asc|desc,...]

**说明**

* 将行数据按照列1进行排序，如果某些行列1的值相同时，则按照列2排序，以此类推
* 默认按照列值从小到大排列（asc）
* asc从小到大排列，即升序
* desc从大到小排序，即降序

例1：查询未删除学生的信息，按名称升序

select \* from students where is\_delete=0 order by name;

例2：查询年龄在18到34岁之间的女性，身高从高到矮排序, 如果身高相同的情况下按照年龄从小到大排序

select \* from students where (age between 18 and 34) and gender = "女" order by height desc,age asc;

例3：显示所有的学生信息，先按照年龄从大-->小排序，当年龄相同时 按照身高从高-->矮排序

select \* from students order by age desc,height desc;

**聚合函数**

为了快速得到统计数据，经常会用到如下5个聚合函数

**1.求总数**

* count(\*)表示计算表的总行数
* count(\列)表示计算某列的总行数，假如该列某个值为null则不会统计

例1：查询学生总数

select count(\*) from students;

例2：统计学生表中，height 字段 有数据的总人数

select count(height) from students;

**2.求最大值**

* max(列)表示求此列的最大值

例3：求年龄最大的女生

select max(age) from students where gender="女";

**3.求最小值**

* min(列)表示求此列的最小值

例3：查询未删除的学生最小编号

select min(id) from students where is\_delete=0;

**4.求和**

* sum(列)表示求此列的和

例4：查询男生的总年龄

select sum(age) from students where gender=1;

-- 平均年龄

select sum(age)/count(\*) from students where gender=1;

**5.求平均值**

* avg(列)表示求此列的平均值

例5：查询未删除女生的编号平均值

select avg(id) from students where is\_delete=0 and gender="女";

**分组**

在实际业务中，经常会对数据进行汇总分析，通过 group by 分组查询可解决该需求

**1. group by**

1. group by的含义:将查询结果按照1个或多个字段进行分组，字段值相同的为一组
2. group by可用于单个字段分组，也可用于多个字段分组

[复制代码](javascript:void(0);)

select \* from students;

+----+-----------+------+--------+--------+--------+-----------+

| id | name | age | height | gender | cls\_id | is\_delete |

+----+-----------+------+--------+--------+--------+-----------+

| 1 | 小明 | 18 | 180.00 | 女 | 1 | |

| 2 | 小月月 | 18 | 180.00 | 女 | 2 | |

| 3 | 彭于晏 | 29 | 185.00 | 男 | 1 | |

| 4 | 刘德华 | 59 | 175.00 | 男 | 2 | |

| 5 | 黄蓉 | 38 | 160.00 | 女 | 1 | |

| 6 | 凤姐 | 28 | 150.00 | 保密 | 2 | |

| 7 | 王祖贤 | 18 | 172.00 | 女 | 1 | |

| 8 | 周杰伦 | 36 | NULL | 男 | 1 | |

| 9 | 程坤 | 27 | 181.00 | 男 | 2 | |

| 10 | 刘亦菲 | 25 | 166.00 | 女 | 2 | |

| 11 | 金星 | 33 | 162.00 | 中性 | 3 | |

| 12 | 静香 | 12 | 180.00 | 女 | 4 | |

| 13 | 周杰 | 34 | 176.00 | 女 | 5 | |

| 14 | 郭靖 | 12 | 170.00 | 男 | 4 | |

+----+-----------+------+--------+--------+--------+-----------+

select gender from students group by gender;

+--------+

| gender |

+--------+

| 男 |

| 女 |

| 中性 |

| 保密 |

+--------+

[复制代码](javascript:void(0);)

根据gender字段来分组，gender字段的全部值有4个'男','女','中性','保密'，所以分为了4组 当group by单独使用时，只显示出每组的第一条记录。 假如要显示表中的列，则只能显示分组中的列

**2. group by + group\_concat()**

1. group\_concat(字段名)可以作为一个输出字段来使用
2. 表示分组之后，根据分组结果，使用group\_concat()来放置每一组的某字段的值的集合

例1：统计出同种性别的学生姓名

[复制代码](javascript:void(0);)

select gender from students group by gender;

+--------+

| gender |

+--------+

| 男 |

| 女 |

| 中性 |

| 保密 |

+--------+

select gender,group\_concat(name) from students group by gender;

+--------+-----------------------------------------------------------+

| gender | group\_concat(name) |

+--------+-----------------------------------------------------------+

| 男 | 彭于晏,刘德华,周杰伦,程坤,郭靖 |

| 女 | 小明,小月月,黄蓉,王祖贤,刘亦菲,静香,周杰 |

| 中性 | 金星 |

| 保密 | 凤姐 |

+--------+-----------------------------------------------------------+

[复制代码](javascript:void(0);)

例2：统计出同种性别的学生id

[复制代码](javascript:void(0);)

select gender,group\_concat(id) from students group by gender;

+--------+------------------+

| gender | group\_concat(id) |

+--------+------------------+

| 男 | 3,4,8,9,14 |

| 女 | 1,2,5,7,10,12,13 |

| 中性 | 11 |

| 保密 | 6 |

+--------+------------------+

[复制代码](javascript:void(0);)

**3. group by + 聚合函数**

1. 通过group\_concat()的启发，我们既然可以统计出每个分组的某字段的值的集合，那么我们也可以通过聚合函数来对这个值的集合做一些操作

[复制代码](javascript:void(0);)

select gender,group\_concat(age) from students group by gender;

+--------+----------------------+

| gender | group\_concat(age) |

+--------+----------------------+

| 男 | 29,59,36,27,12 |

| 女 | 18,18,38,18,25,12,34 |

| 中性 | 33 |

| 保密 | 28 |

+--------+----------------------+

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

例3：按性别分组，统计出每组的平均年龄

select gender,avg(age) from students group by gender;

+--------+----------+

| gender | avg(age) |

+--------+----------+

| 男 | 32.6000 |

| 女 | 23.2857 |

| 中性 | 33.0000 |

| 保密 | 28.0000 |

+--------+----------+

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

例4：按性别分组，统计出每组的人数

select gender,count(\*) from students group by gender;

+--------+----------+

| gender | count(\*) |

+--------+----------+

| 男 | 5 |

| 女 | 7 |

| 中性 | 1 |

| 保密 | 1 |

+--------+----------+

[复制代码](javascript:void(0);)

**4. group by + having**

1. having 条件表达式：分组查询完后，再指定一些条件对分组后的查询结果进行过滤
2. having作用和where相似，但having只能用于group by，且having可以使用聚合函数

例5：按性别分组，分别统计出平均年龄超过30岁的组的总人数

[复制代码](javascript:void(0);)

select gender ,count(\*) from students group by gender having avg(age)>30;

+--------+----------+

| gender | count(\*) |

+--------+----------+

| 男 | 5 |

| 保密 | 1 |

+--------+----------+

[复制代码](javascript:void(0);)

**5. group by + with rollup**

1. with rollup的作用是：在最后新增一行，来记录当前列里所有记录的总和

[复制代码](javascript:void(0);)

select gender,count(\*) from students group by gender with rollup;

+--------+----------+

| gender | count(\*) |

+--------+----------+

| 男 | 5 |

| 女 | 7 |

| 中性 | 1 |

| 保密 | 1 |

| NULL | 14 |

+--------+----------+

select gender,group\_concat(age) from students group by gender with rollup;

+--------+-------------------------------------------+

| gender | group\_concat(age) |

+--------+-------------------------------------------+

| 男 | 29,59,36,27,12 |

| 女 | 18,18,38,18,25,12,34 |

| 中性 | 33 |

| 保密 | 28 |

| NULL | 29,59,36,27,12,18,18,38,18,25,12,34,33,28 |

+--------+-------------------------------------------+

[复制代码](javascript:void(0);)

**获取部分行**

当数据量过大时，通过分批、分页加载数据既能提升加载速度，也可更好显示查询结果

**语法**

select \* from 表名 limit start,count

**说明**

* 从start位置开始，获取count条数据
* limit 必须放在查询的最后面

例1：查询前3行男生信息

select \* from students where gender=1 limit 3;

select \* from students where gender=1 limit 0,3;

**示例：分页**

* 已知：每页显示m条数据，当前显示第n页
* 求第n页的数据

select \* from students where is\_delete=0 limit (n-1)\*m,m

**连接查询**

当查询结果的列来源于多张表时，需要将多张表连接成一个大的数据集，再选择合适的列返回

mysql支持三种类型的连接查询，分别为：

* 内连接查询：查询的结果为两个表匹配到的数据
* 左连接查询：查询的结果为两个表匹配到的数据，左表特有的数据，对于右表中不存在的数据使用null填充
* 右连接查询：查询的结果为两个表匹配到的数据，右表特有的数据，对于左表中不存在的数据使用null填充

**语法**

select \* from 表1 inner或left或right join 表2 on 表1.列 = 表2.列

例1：使用内连接查询班级表与学生表

select \* from students inner join classes on students.cls\_id = classes.id;

例2：使用左连接查询班级表与学生表

* 此处使用了as为表起别名，目的是编写简单

select \* from students as s left join classes as c on s.cls\_id = c.id;

例3：使用右连接查询班级表与学生表

select \* from students as s right join classes as c on s.cls\_id = c.id;

例4：查询学生姓名及班级名称

select s.name,c.name from students as s inner join classes as c on s.cls\_id = c.id;

**自关联**

* 设计省信息的表结构provinces
  + id
  + title
  + proid
* 设计市信息的表结构citys
  + id
  + title
  + proid
* 设计区县信息表结构areas
  + id
  + title
  + proid
* title表示名称，proid表示所属上一级的id值，比如citys.proid 是对应所属省份的id，areas.proid对应的是所属城市的id值

**问题：**

能不能将三个表合成一张表呢？

**思考：**

观察三张表发现，表的结构都是一样的，存储的都是地区信息，而且每种信息的数据量有限

**意义：**

假如合成一张表，即可减少创建表的开销，也用不着多个表关联查询

**答案：**

定义表areas，结构如下

* id
* atitle
* pid

**说明:**

* 因为省没有所属的省份，所以可以填写为null
* 城市所属的省份pid，填写省所对应的编号id
* 区县所属的城市pid，填写城市所对应的编号id
* 这就是自关联，表中的某一列，关联了这个表中的另外一列，但是它们的业务逻辑含义是不一样的，城市信息的pid引用的是省信息的id

**创建areas表的语句如下：**

create table areas(

aid int primary key,

atitle varchar(20),

pid int

);

* 从sql文件中导入数据

source areas.sql;

* 查询一共有多少个省

select count(\*) from areas where pid is null;

* 例1：查询省的名称为“山西省”的所有城市

select city.\* from areas as city

inner join areas as province on city.pid=province.aid

where province.atitle='山西省';

* 例2：查询市的名称为“广州市”的所有区县

select dis.\* from areas as dis

inner join areas as city on city.aid=dis.pid

where city.atitle='广州市';

**子查询**

**子查询**

在一个 select 语句中,嵌入了另外一个 select 语句, 那么被嵌入的 select 语句称之为子查询语句

**主查询**

主要查询的对象,第一条 select 语句

**主查询和子查询的关系**

* 子查询是嵌入到主查询中
* 子查询是辅助主查询的,要么充当条件,要么充当数据源
* 子查询是可以独立存在的语句,是一条完整的 select 语句

**子查询分类**

* 标量子查询: 子查询返回的结果是一个值(一行一列)
* 列子查询: 返回的结果是一列(一列多行)
* 行子查询: 返回的结果是一行(一行多列)
* 表子查询: 返回的结果是一个临时表(多行多列)

**标量子查询**

* 例1: 查询大于平均年龄的学生

select \* from students where age > (select avg(age) from students);

**列级子查询**

* 例2: 查询有班级名称的学生信息
  1. 找出班级表中的班级 id
  2. 找出学生表中有对应班级的名字

select name from students where cls\_id in (select id from classes);

**行级子查询(了解)**

* 例3: 查找班级年龄最大,且身高最高的学生

select \* from students where (height,age) = (select max(height),max(age) from students);

**表子查询(了解)**

* 例4: 查询编号小于6的男性同学的姓名

select name from (select gender,name from students where id<6) as s where gender="男";

**子查询中特定关键字使用**

* in 范围
* 格式: 主查询 where 条件 in (列子查询)

**总结**

**查询的完整格式 ^\_*^ 不要被吓到 其实很简单 !*\_ !**

SELECT select\_expr [,select\_expr,...] [

FROM tb\_name

[WHERE 条件判断]

[GROUP BY {col\_name | postion} [ASC | DESC], ...]

[HAVING WHERE 条件判断]

[ORDER BY {col\_name|expr|postion} [ASC | DESC], ...]

[ LIMIT {[offset,]rowcount | row\_count OFFSET offset}]

]

* 完整的select语句

select distinct \*

from 表名

where ....

group by ... having ...

order by ...

limit start,count

* 执行顺序为：
  + from 表名
  + where ....
  + group by ...
  + select distinct \*
  + having ...
  + order by ...
  + limit start,count
* 实际使用中，只是语句中某些部分的组合，而不是全部

标签: [mysql](https://www.cnblogs.com/liuguangbiao/tag/mysql/), [mysql查询详细](https://www.cnblogs.com/liuguangbiao/tag/mysql%E6%9F%A5%E8%AF%A2%E8%AF%A6%E7%BB%86/)