MySQL第二讲

主讲人：周景阳（大周）

大纲

- 条件查询
- 联合查询
- 分组与排序
- 子查询、关联查询
- 常用函数

条件查询–基本过滤

基本格式: select 字段名 from 表名 where 过滤条件

*代表所有字段，当查询多个字段时用逗号,进行分隔

举例: select * from score where score > 90

查询分数表中分数大于90的所有字段

常用比较运算符: >(大于)、<(小于)、 =(等于)、!=或<>(不等于)、>=(大于等于)、<=(小于等于)

条件查询–模糊查询

基本格式: select 字段名 from 表名 where 字段名 like “%模糊的关键字%”;

模糊查询的关键字

两端带百分号就是要查询的字段中间需要包含这个关键字

基本格式: select 字段名 from 表名 where 字段名 like “%模糊的关键字”;

两端带百分号就是要查询的字段尾部需要以这个关键字结尾

下边这个代表什么意思?

基本格式: select 字段名 from 表名 where 字段名 like “模糊的关键字%”;

条件查询–多条件查询

逻辑运算符and: select 字段名 from 表名 where 过滤条件1 and 过滤条件2；

逻辑运算符or: select 字段名 from 表名 where 过滤条件1 or 过滤条件2；

条件查询–范围查询

一、基本格式: select 字段名 from 表名 where 字段 in（值1,值2,值3……);

说明: 查询字段的值为值1,值2或值3的数据

举例: select * from student where id in (1,5,10);

二、基本格式: select 字段名 from 表名 where 字段 between（值1,值2);

说明: 查询字段的值为值1到值2的数据

举例: select * from student where id between 5 and 10

条件查询–空的判断

空值: null

空字符串:""

空值判断:is null 和 is not null

举例:

```
select * from school where school_address is null;
select * from school where school_address is not null;
select * from school where school_address = "";
```

条件查询–返回结果的部分数据

关键字: limit

举例:

select * from score limit 10; 返回前查询结果的前10条数据

select * from score limit 10,20; 返回查询结果中跳过前10数据的20条数据

举例:

select * from score where score > 100 limit 10;

select * from score where score > 100 limit 10,20;

联合查询

基本格式:

select 字段名 from 表名 where 筛选条件

union

select 字段名 from 表名 where 筛选条件

注意:

1、【强制要求】多表联合查询时，所查询的字段个数必须相同

2、【潜规则】多表联合查询时，每个表所查询出来的字段所代表的意义要相同，查询结果中列的名字会显示最上边的第一个查询的列名字

分组查询

关键字:group by

基本格式:select * from course group by course_name;

常用使用方法(对分组结果进行统计):

select count(id),course_name from course group by course_name;

select max(id),course_name from course group by course_name;

分组查询

组结果条件筛选格式:group by having

举例:

查询出出现过5次以上的课程

```
select count(id),course_name from course group by course_name having count(id)>5
```

查询出每个学生的总分

```
select sum(score),student_id from score GROUP BY student_id
```

查询出每个学生的最高分

```
select max(score),student_id from score GROUP BY student_id
```

分组查询

针对组结果条件筛选后的结果再筛选格式:group by having with rollup

举例:查询出查询出每个学生的总分的同时，再查出所有学生一共的得分

```
select sum(score),student_id from score GROUP BY student_id with rollup
```

查询出每个学生的最高分的同时再查询出全部学生的最高分

```
select max(score),student_id from score GROUP BY student_id with rollup
```

结果排序

关键字:order by [asc/desc]

举例:

select sum(score) from score group by student_id order by sum(score) [asc]

正序排列时asc关键字加不加都可以

倒序排列:

select sum(score) from score group by student_id order by sum(score) desc

子查询

先学个别名，关键字 as

举例： select student_id as "学生ID", course_id as "课程ID", score as "分数" from score

其实这个as不写也行：

select student_id "学生ID", course_id "课程ID", score "分数" from score

思考一下下边语句什么意思：

select s.student_id "学生ID", s.course_id "课程ID", s.score "分数" from score s;

子查询

子查询的基本格式

select * from 表名 where 筛选条件字段 逻辑运算符 (

select * from 表名 where 筛选条件)

举例：有年级的学校

select * from school where id in (select school_id from grade)

举例：查询出六年级，全体同学的数学分数

select * from score where course_id = (

select id from course where grade_id in (

select id from grade where grade_name = "六年级")

and course_name = "数学")

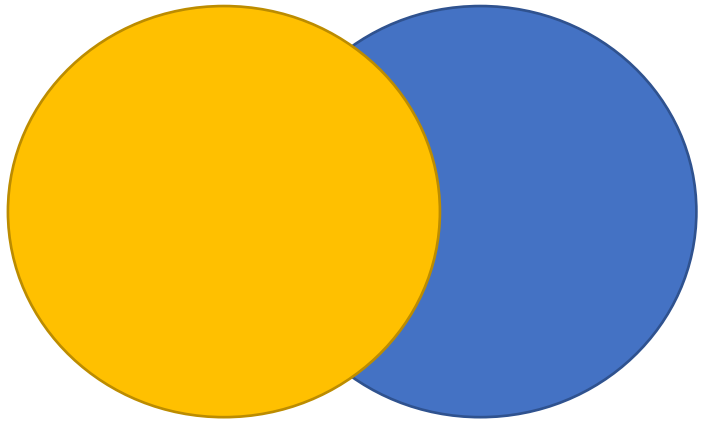
关联查询

左关联:left join on

解释：取左侧的全集和左侧表与右侧的交集

举例:

```
select s.*,g.* from school s
left join grade g
on s.id = g.school_id
```



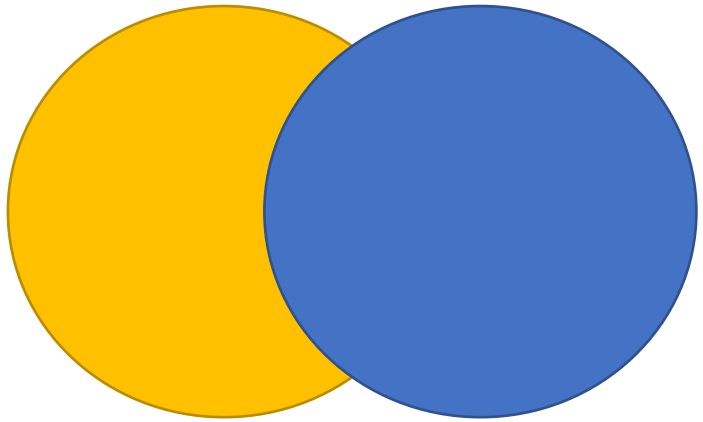
关联查询

右关联:right join on

解释：取右侧的全集和右侧表与左侧的交集

举例:

```
select s.*,g.* from school s
right join grade g
on s.id = g.school_id
```



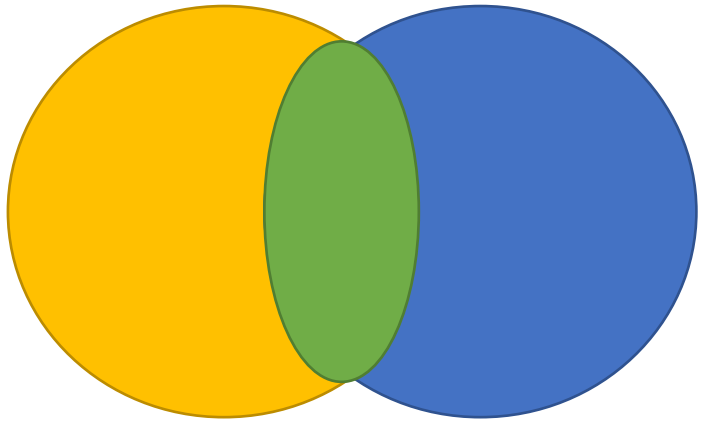
关联查询

全关联(内关联/自连接):[inner] join on

解释：取左侧表和右侧的交集

举例:

```
select s.*,g.* from school s
join grade g
on s.id = g.school_id
```



常用函数

去重 distinct

最大 max

最小 min

平均 avg

求和 sum

MD5加密 md5

绝对值　abs

计数　count

获取时间戳 unix_timestamp 返回1970-01-01到现在秒数

获取时间from_unixtime(“加一个秒的参数”)

时间相减 datediff(a,b)

获取当前时间 current_date