排序的语法为: order by 字段名 [asc][desc]，默认是升序asc，如果是升序，则可以忽略排序规则的书写，但是降序必须书写desc

聚合函数

max(字段) 求最大值

min(字段) 求最小值

sum(字段) 求和

avg(字段) 求均值

count(字段) 求记录条数

聚合函数的返回结果都是单行单列的数据，也就是单数据

如果字段的值为null，则不计数

count(\*)在统计时，只要所有字段中有一个不为null，即可计数

除非所有字段都是null，才不会计数

****分组查询****

**语法格式：group by 字段名，按照此字段进行分组**

**分组查询的结果都是通过聚合函数来得到的**

**分组查询与聚合函数是密不可分**

**什么时候使用分组查询呢？？？**

**当需求中出现每个、每组之类的关键字时，就需要使用分组查询**

查询的关键字顺序

**select 查询的内容(字段、常量、运算结果、聚合函数)**

**from 表名**

**where**

**1、条件(只能以字段作为条件，不能直接书写聚合函数！)**

**2、where后面的条件是不需要分组就可使用的条件**

**group by 字段 分组**

**having 条件**

**1、条件(不能直接以字段作为条件，只能以聚合函数作为条件)**

**2、having后面的条件只能是分组后才知道的条件**

**order by 字段 排序**

**limit 分页**

where 与 having的区别？？？

**where**

**只能以字段作为条件，不能直接跟聚合函数**

**where后面的条件是不分组即可得到的条件**

**where出现在group by之前**

having

只能以聚合函数作为条件，不能直接跟字段

having后面的条件只有分组之后才可使用

having出现在group by之后

分页查询

**limit 参数1, 参数2**

**参数1 代表查询的起始位置，下标从0开始**

**参数2 代表查询的数据条数**

**limit是MySQL的方言，limit这个关键字仅限在MySQL中使用**

**特殊情况：**

**如果想要从第一条开始查询，可以省略参数1的书写，直接写查询条数**

**比如limit 4,就是从第一条开始查，查四条**

约束

**约束的分类：**

**\*主键约束，主键的作用是用来唯一标识表中的一条记录的。主键有个特点：它不具备含义！！！仅仅用来标识唯一记录的！非空且唯一！**

**\*唯一约束，字段值不能重复**

**\*非空约束，字段值不能为空null**

**\*默认约束，在插入数据时，可以省略默认约束字段的赋值，那么就会选取默认值**

**\*外键约束，是实现表关系设计的前提**

**# auto\_increment的作用：自增长，在插入数据时，自增长的主键可以传入null，自动有值**

**# auto\_increment修饰主键的类型必须是int类型**！！！

## 外键约束

**外键是表与表关系设计的核心，它的存在让表与程序的设计更加合理，也更加明确**

**外键的注意事项：**

**1、外键必须是另一张表的主键！！！**

**2、一张表中可以存在多个外键**

**3、外键的值可以重复**

**4、外键的值可以为null**

**5、外键的名称"必须"与关联表主键名称一致！**

**constraint 声明外键约束的关键字**

**fk\_student\_teacher 外键的描述**

**foreign key(tid) 当前表中的tid字段为外键**

**references 关联**

**teacher(tid) teacher表里的tid字段**

表与表的关系

一对多关系的设计原则：

一对多关系中，一方的表称为主表，多方的表称为从表。在从表中添加外键，用来关联主表的主键，这样就可以实现一对多关系！一对多关系的使用场景是最多的！

一对一关系的设计原则：

两张表A、B，分别添加主键，在其中任意一张表中设置外键关联另一张表的主键，并且外键添加在当前表的主键上。一对一关系使用的场景并不多！

多对多关系的设计原则：

两张表A、B，分别添加主键，创建C表作为中间表，在C表中设置外键，用来分别关联A、B表的主键！场景也居多！

连接查询(多表查询)

什么场景下需要使用连接查询？？？

**查询的结果集中的内容包含多张表的数据时，就需要使用连接查询。**

1. **内连接：只能找到去除笛卡尔积之后多张表中公共的数据，取交集**
2. **外连接：可以依赖于某一张表进行查询，此表中的数据都会得到，另一张表中不满足连接条件的全部为null**

内连接查询

**select 字段名,... from 表1, 表2 where 连接条件**

在使用连接查询时，出现大量错误信息的原因是没有给定连接条件

这种现象称为笛卡尔积现象

emp 15 department 4 15 \* 4

所以在使用连接查询时一定要书写表连接的条件

正因为有连接条件，才说明表之间存在关系

我们以后不会同时得到两张或者多张毫无关系的表中的数据

**去除笛卡尔积：**

**内连接查询 99标准版**

**select 字段,... from 表1 inner join 表2 on 连接条件**

**自然内连接**

select 字段,... from 表1 natural join 表2;

优点：

不需要书写连接条件，自动去除笛卡尔积！

缺点：

**1、两张表中必须存在同名字段！！！此同名字段会自动作为去除笛卡尔积的条件**

**解释了为什么外键的名称要与关联表主键的名称一致！！！**

**2、为什么表中的姓名或者名称不以name命名，而要加表名的首字母呢？**

**因为如果两张表中存在不止一个同名字段，那么在去除笛卡尔积上**

**需要两个字段值同时相等才可以**

**有且只有一个同名字段，才能使用自然内连接！！！**

外连接

外连接：左外连接、右外连接

左外：左外、自然左外

右外：右外、自然右外

\*/

/\*

**左外连接**

select 字段,... from 表1 left join 表2 on 去除笛卡尔积的条件

**左外连接以左表为准，左表中的数据都可以获取**

**如果右表中的数据不满足连接条件，全部为null**

\*/

/\*

**左外自然连接**

select 字段,... from 表1 natural left join 表2;

**会自动去除笛卡尔积，但是要保证两张表中有且只有一个同名字段**

**而这个同名字段就是我们的外键**

\*/

**右连接反之**

****子查询****

子查询是一个结果集

一个查询语句中包含另外一个查询语句

一个查询语句的结果集有几种表现形式呢？？？

1. 单行单列 聚合函数

select max(sal) from emp;

1. 多行多列

select \* from emp;

1. 单行多列 通过主键或者唯一字段进行查找

select \* from emp where empno = 1001;

1. 单列多行 查询一个字段的值

select ename from emp;

**子查询中，子查询的结果集为单行单列的场景是最多的！**

#查询薪资大于平均薪资的员工信息

SELECT AVG(sal) FROM emp;

SELECT \* FROM emp

WHERE sal > (SELECT AVG(sal) FROM emp);

#查询大于30部门所有员工薪资的员工信息

/\*

1、大于30部门最高薪资

2、大于30部门任意一个员工的薪资

\*/

SELECT MAX(sal) FROM emp WHERE deptno = 30;

SELECT \* FROM emp

WHERE sal > (SELECT MAX(sal) FROM emp WHERE deptno = 30);

SELECT sal FROM emp WHERE deptno = 30;#30部门所有薪资 单列多行

SELECT \* FROM emp

WHERE sal > ALL(SELECT sal FROM emp WHERE deptno = 30);

#查询高于任意一位经理工资的员工信息

/\*

1、大于经理中工资最低的

2、大于任意一位经理

\*/

SELECT MIN(sal) FROM emp WHERE job = "经理";

SELECT \* FROM emp

WHERE sal > (SELECT MIN(sal) FROM emp WHERE job = "经理")

AND job != "经理";

SELECT sal FROM emp WHERE job = "经理";

SELECT \* FROM emp

WHERE sal > ANY(SELECT sal FROM emp WHERE job = "经理")

AND NOT(job = "经理");

#查询出与殷天正岗位及薪资相同的员工信息

SELECT job, sal FROM emp WHERE ename = "殷天正";#多行多列

SELECT \* FROM emp

/\*

job sal

销售员 12500

分析师 50000\*/

WHERE (job, sal) IN(SELECT job, sal FROM emp WHERE ename = "殷天正")

AND ename != "殷天正";