连接查询，又称多表查询：（连接查询一般会为表起别名，为表名起了别名以后，查询的字段只能用别名进行限制不能用原来的名称）

Select \*from 表A,表B

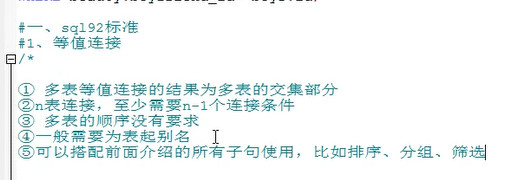
这种查询方式可能出现的问题：这样是使用表A中所有的记录逐行与表B中所有的记录逐行连接。就是直接笛卡尔积，最终结果集m\*n行。

（**笛卡尔积出现的原因是因为没有用有效的连接条件，所以避免笛卡尔积的方法是添加有效的连接条件**）

连接的分类：

按年代分类：

Sql192标准：仅支持内连接



Sql199标准：在MySQL中支持内连接、外连接（左外和右外）、交叉连接

**按功能分类：**

**内连接**

**等值连接**

**非等值连接**

**自连接**

**外连接**

**左外连接**

**右外连接**

**全外连接**

**交叉连接**

## Sql192标准：

内连接：

等值连接（实现的方法：使用前面表的每一行去和后面表的每一行进行连接，但是在连接之前会先进行筛选，筛选的条件便是定义的连接条件，如果符合条件则进行连接，否则就不进行连接）（！！！！！！！！！！！注意，也就是不是说他们两张表先进行笛卡尔积再用where子句进行筛选，如果这样的话效率会及其低下）

等值连接调换表的连接顺序对结果无影响，连接的条件可以有多个用逻辑连接符进行连接。



(group by后面有多个字段怎么执行？

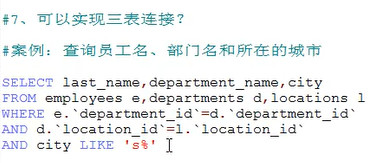
它会将两个字段的值都相同的行划分到一组中

可以看这篇文章中的示例：

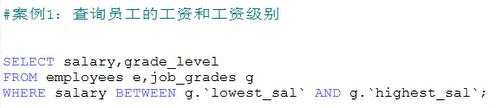
<https://www.cnblogs.com/zhuchenglin/p/9732612.html>

)

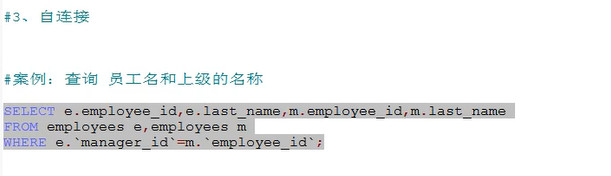




非等值连接：（非等值连接不一定是不等号，而是逻辑运算如<、>、>=）



**自连接:（自己和自己连接，一张表当多张表使用）**



## Sql199标准：

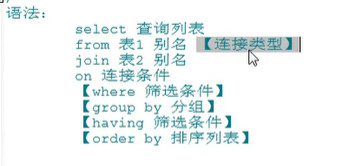
(两者放置相同条件，之所以可能会导致结果集不同，就是因为优先级。on的优先级是高于where的

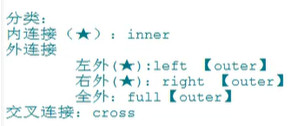
可以看看这篇文章：

<https://blog.csdn.net/cs958903980/article/details/60139792>

***这么一想就发现，其实使用199标准似乎并没有192标准灵活似乎某些情况下还会被迫降低效率，但是实际上192主要支持内连接，而在内连接时使用on和where是没有差别的，当进行外连接时，192仅仅支持部分，所以你说效率上的问题并没有太大意义。***

)

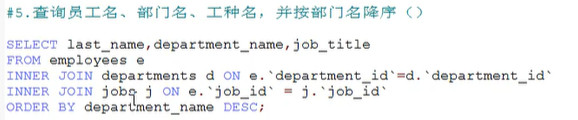




**内连接（inner join）**

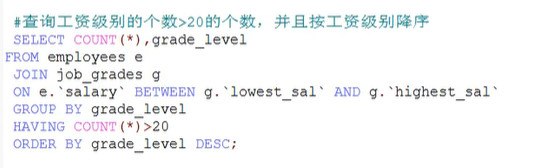
**等值连接**





**（三表连接的顺序问题，因为三表连接是先进行两个连接再和第三个连接，中间表可能出现和第三张表无连接的字段的情况，所以在写时需要进行细致考虑）**

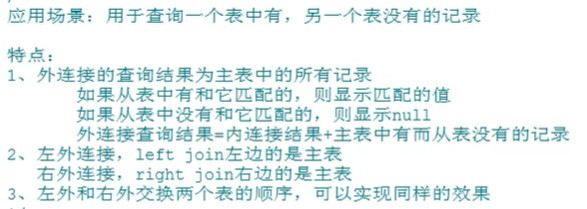
**非等值连接**



**自连接**

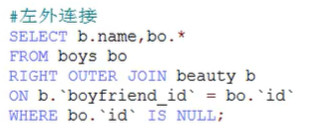
**外连接**

**（左外连接与右外连接的执行过程：将主表中的每一条记录逐行与从表中的每一条记录进行匹配，如果满足连接条件酒进行连接，如果有多行就逐行连接，如果匹配完从表中都没有一个满足条件的行就将主表中的这个行连接部分填充为空）**



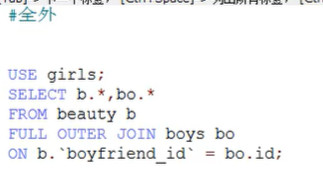
**左外连接left[outer] join**

**右外连接right[outer]join**



**全外连接full[outer]join（MySQL不支持）**





**交叉连接cross join（笛卡尔积）**

