（先从，from开始进行笛卡尔积）

今天遇到一个问题就是mysql中insert into 和update以及delete语句中能使用as别名吗？目前还在查看，但是在查阅资料时发现了一些有益的知识，给大家分享一下，就是关于sql以及MySQL语句执行顺序：

sql和mysql执行顺序，发现内部机制是一样的。最大区别是在别名的引用上。

一、sql执行顺序

from

join

on

where

group by(开始使用select中的别名，后面的语句中都可以使用)

 avg,sum....

having

select

distinct

order by

limit

从这个顺序中我们不难发现，所有的 查询语句都是从from开始执行的，在执行过程中，每个步骤都会为下一个步骤生成一个虚拟表，这个虚拟表将作为下一个执行步骤的输入。

第一步：首先对from子句中的前两个表执行一个笛卡尔乘积，此时生成虚拟表 vt1（选择相对小的表做基础表）。

第二步：接下来便是应用on筛选器，on 中的逻辑表达式将应用到 vt1 中的各个行，筛选出满足on逻辑表达式的行，生成虚拟表 vt2 。

第三步：如果是outer join 那么这一步就将添加外部行，left outer jion 就把左表在第二步中过滤的添加进来，如果是right outer join 那么就将右表在第二步中过滤掉的行添加进来，这样生成虚拟表 vt3 。

第四步：如果 from 子句中的表数目多余两个表，那么就将vt3和第三个表连接从而计算笛卡尔乘积，生成虚拟表，该过程就是一个重复1-3的步骤，最终得到一个新的虚拟表 vt3。

第五步：应用where筛选器，对上一步生产的虚拟表引用where筛选器，生成虚拟表vt4，在这有个比较重要的细节不得不说一下，对于包含outer join子句的查询，就有一个让人感到困惑的问题，到底在on筛选器还是用where筛选器指定逻辑表达式呢？on和where的最大区别在于，如果在on应用逻辑表达式那么在第三步outer join中还可以把移除的行再次添加回来，而where的移除的最终的。举个简单的例子，有一个学生表（班级,姓名）和一个成绩表(姓名,成绩)，我现在需要返回一个x班级的全体同学的成绩，但是这个班级有几个学生缺考，也就是说在成绩表中没有记录。为了得到我们预期的结果我们就需要在on子句指定学生和成绩表的关系（学生.姓名=成绩.姓名）那么我们是否发现在执行第二步的时候，对于没有参加考试的学生记录就不会出现在vt2中，因为他们被on的逻辑表达式过滤掉了,但是我们用left outer join就可以把左表（学生）中没有参加考试的学生找回来，因为我们想返回的是x班级的所有学生，如果在on中应用学生.班级='x'的话，left outer join会把x班级的所有学生记录找回（感谢网友康钦谋\_\_康钦苗的指正），所以只能在where筛选器中应用学生.班级='x' 因为它的过滤是最终的。

第六步：group by 子句将中的唯一的值组合成为一组，得到虚拟表vt5。如果应用了group by，那么后面的所有步骤都只能得到的vt5的列或者是聚合函数（count、sum、avg等）。原因在于最终的结果集中只为每个组包含一行。这一点请牢记。

第七步：应用cube或者rollup选项，为vt5生成超组，生成vt6.

第八步：应用having筛选器，生成vt7。having筛选器是第一个也是为唯一一个应用到已分组数据的筛选器。

第九步：处理select子句。将vt7中的在select中出现的列筛选出来。生成vt8.

第十步：应用distinct子句，vt8中移除相同的行，生成vt9。事实上如果应用了group by子句那么distinct是多余的，原因同样在于，分组的时候是将列中唯一的值分成一组，同时只为每一组返回一行记录，那么所以的记录都将是不相同的。

第十一步：应用order by子句。按照order\_by\_condition排序vt9，此时返回的一个游标，而不是虚拟表。sql是基于集合的理论的，集合不会预先对他的行排序，它只是成员的逻辑集合，成员的顺序是无关紧要的。对表进行排序的查询可以返回一个对象，这个对象包含特定的物理顺序的逻辑组织。这个对象就叫游标。正因为返回值是游标，那么使用order by 子句查询不能应用于表表达式。排序是很需要成本的，除非你必须要排序，否则最好不要指定order by，最后，在这一步中是第一个也是唯一一个可以使用select列表中别名的步骤。

第十二步：应用top选项。此时才返回结果给请求者即用户。

二、mysql的执行顺序

1、SELECT语句定义

一个完成的SELECT语句包含可选的几个子句。SELECT语句的定义如下：

SQL代码

<SELECT clause> [<FROM clause>] [<WHERE clause>] [<GROUP BY clause>] [<HAVING clause>] [<ORDER BY clause>] [<LIMIT clause>]

SELECT子句是必选的，其它子句如WHERE子句、GROUP BY子句等是可选的。

一个SELECT语句中，子句的顺序是固定的。例如GROUP BY子句不会位于WHERE子句的前面。

2、SELECT语句执行顺序

SELECT语句中子句的执行顺序与SELECT语句中子句的输入顺序是不一样的，所以并不是从SELECT子句开始执行的，而是按照下面的顺序执行：

开始->FROM子句->WHERE子句->GROUP BY子句->HAVING子句->ORDER BY子句->SELECT子句->LIMIT子句->最终结果

每个子句执行后都会产生一个中间结果，供接下来的子句使用，如果不存在某个子句，就跳过

对比了一下，mysql和sql执行顺序基本是一样的, 标准顺序的 SQL 语句为:

select 考生姓名, max(总成绩) as max总成绩

from tb\_Grade

where 考生姓名 is not null

group by 考生姓名

having max(总成绩) > 600

order by max总成绩

 在上面的示例中 SQL 语句的执行顺序如下:

　　 (1). 首先执行 FROM 子句, 从 tb\_Grade 表组装数据源的数据

　　 (2). 执行 WHERE 子句, 筛选 tb\_Grade 表中所有数据不为 NULL 的数据

　　 (3). 执行 GROUP BY 子句, 把 tb\_Grade 表按 "学生姓名" 列进行分组(注：这一步开始才可以使用select中的别名，他返回的是一个游标，而不是一个表，所以在where中不可以使用select中的别名，而having却可以使用，感谢网友  zyt1369  提出这个问题)

　　 (4). 计算 max() 聚集函数, 按 "总成绩" 求出总成绩中最大的一些数值

　　 (5). 执行 HAVING 子句, 筛选课程的总成绩大于 600 分的.

　　 (7). 执行 ORDER BY 子句, 把最后的结果按 "Max 成绩" 进行排序.