90 再次重温写出各种SQL语句的时候，会用什么执行计划？（3）

今天我们继续看看写出各种SQL语句的时候，会有什么样的执行计划？其实这些都是MySQL优化的一些基础知识。

如果大家不能把这些理论知识夯的很扎实的话，那么后续的多个MySQL SQL调优实战案例根本不可能会看懂，因为调优的前提，就是彻底搞明白执行计划，也就是彻底搞明白你的一个SQL，现在性能差，他是如何执行的，为什么性能会这么差，应该怎么改写或者设计索引，才能让他的性能变得更好。

之前讲了，有的时候可能会在一个SQL里同时用上多个索引，那么其实如果你在SQL里写了类似x1=xx or x2=xx的语句，也可能会用多个索引，只不过查多个大索引树之后，会取一个并集，而不是交集罢了。

那么现在为止，我们要做一个小小的停顿和总结，就是现在大家已经知道写出来的SQL有哪些执行的方式了。const、ref、range，都是性能最好的方式，说明在底层直接基于某个索引树快速查找了数据了，但有的时候可能你在用了索引之后，还会在回表到聚簇索引里查完整数据，接着根据其他条件来过滤。

然后index方式其实是扫描二级索引的意思，就是说不通过索引树的根节点开始快速查找，而是直接对二级索引的叶子节点遍历和扫描，这种速度还是比较慢的，大家尽量还是别出现这种情况。

当然index方式怎么也比all方式好一些，all就是直接全表扫描了，也就是直接扫描聚簇索引的叶子节点，那是相当的慢，index虽然扫描的是二级索引的叶子节点，但是起码二级索引的叶子节点数据量比较小，相对all要快一些。

然后之前给大家说的可能一个SQL里用多个索引，意思就是可能对多个索引树进行查找，接着用intersection交集、union并集的方式来进行合并，此时可能给你在执行计划里也会看到这些字样，那你起码这里要知道是怎么回事，其实他就是告诉你，他查找了多个索引，做了一些结果集的交集或者是并集，而且这种方式也不一定是会发生的。

好了，到这里为止，大家把一些基本的执行计划里的东西都了解差不多了，这其实都是一些单表查询的执行计划可能包含的内容，下周开始，正式讲解MySQL的多表关联的SQL语句会对应哪些执行计划，讲完多表关联的执行计划原理之后，还会讲解MySQL生成执行计划的原理，包括子查询之类的复杂SQL是如何生成执行计划的。

最后我们会讲多个案例，来给大家用真实复杂的SQL语句，来看MySQL生成的真实执行计划，彻底搞定SQL语句是如何执行的，然后再切入SQL调优实战案例，到时候大家一步一步的进行，就会觉得非常的自然了。

End