99 MySQL是如何基于各种规则去优化执行计划的？（下）

今天我们来给大家讲解MySQL里对子查询的执行计划进行优化的一种方式，就是semi join，也就是半连接

这个半连接是什么意思呢，其实就是假设你有一个子查询语句：select * from t1 where x1 in (select x2 from t2 where x3=xxx)，此时其实可能会在底层把他转化为一个半连接，有点类似于下面的样子：

select t1.* from t1 semi join t2 on t1.x1=t2.x2 and t2.x3=xxx

当然，其实并没有提供semi join这种语法，这是MySQL内核里面使用的一种方式，上面就是给大家说那么个意思，其实上面的semi join的语义，是和IN语句+子查询的语义完全一样的，他的意思就是说，对于t1表而言，只要在t2表里有符合t1.x1=t2.x2和t2.x3=xxx两个条件的数据就可以了，就可以把t1表的数据筛选出来了。

其实这个semi join我们这里就是简单提一下概念就行了，但是他还有很多适用场景和不适用场景，我们这里就不再说了，因为也没那个必要，这里先简单了解一下就可以

其实到今天为止，我们就已经把MySQL中各种SQL语句的大致执行原理以及执行计划，都有一个基本的了解了，无论是简单的单表查询，还是多表关联，或者是子查询，大家虽然很多细节还不够熟悉，但是大致的原理基本是都知道了。

而且不同的执行计划到底是如何生成的，可能是根据成本计算选择的，也可能是根据规则优化出来的，然后具体执行计划执行的时候，在底层是如何查询索引的，如何筛选数据，其实大家也都基本清楚了。

到此为止，有了这么多的铺垫，下周开始我们就可以正式进入**执行计划研究环节**了，这是后续能搞定SQL调优的最后一个难题攻关了，只要大家可以看明白各种SQL语句的执行计划以及真实SQL执行过程中各个环节的耗时，找出SQL语句执行慢的原因，那么后续就可以针对性的进行SQL调优。

当然，其实还是要给大家提醒一句，在互联网公司里，我们比较崇尚的是尽量写简单的SQL，复杂的逻辑用Java系统来实现就可以了，SQL能单表查询就不要多表关联，能多表关联就尽量别写子查询，能写几十行SQL就别写几百行的SQL，多考虑用Java代码在内存里实现一些数据就的复杂计算逻辑，而不是都放SQL里做。

其实一般的系统，只要你SQL语句尽量简单，然后建好必要的索引，每条SQL都可以走索引，数据库性能往往不是什么大问题，而接下来要讲的复杂SQL调优，主要是针对那种实在没办法必须写上百行的复杂SQL，然后你还必须得进行调优的情况，当然SQL高级调优也是程序员必须掌握的。

**End**