

100 透彻研究通过explain命令得到的SQL执行计划 (1)

今天我们正式进入研究explain命令得到的SQL执行计划的内容了，只要把explain分析得到的SQL执行计划都研究透彻，完全能看懂，知道每个执行计划在底层是怎么执行的，那么后面学习SQL语句的调优就非常容易了。

首先，我们现在应该都知道每条SQL语句，mysql都会经过成本和规则的优化，对这个SQL选择对应的一些访问方法和顺序，包括做一些特殊的改写确保执行效率是最优的，然后优化过后，就会得到一个执行计划。

这个执行计划其实真没那么神秘，如果你把之前的内容都学习的比较透彻的话就会知道，所谓的执行计划，落实到底层，无非就是先访问哪个表，用哪个索引还是全表扫描，拿到数据之后如何去聚簇索引回表，是否要基于临时磁盘文件做分组聚合或者排序，其实这个计划到最后就是这点东西。

平时我们只要用类似于：explain select * from table，这种SQL前面加一个explain命令，就可以轻松拿到这个SQL语句的执行计划。今天我们就先来看看，这个所谓的执行计划里会有哪些东西。

首先，当你执行explain命令之后，拿到的执行计划可能是类似下面这样的东西：

```
id | select_type | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 | SIMPLE | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | NULL | No
tables used |
```

大家看到那所谓的id、select_type、table、partitions、type之类的东西了吗，其实这些就是所谓的执行计划里包含的东西

大致来说，如果是一个简单的单表查询，可能这里就只有一条数据，也就是代表了他打算如何访问这一个表而已。

但是如果你的SQL语句极为的复杂，可能这里会有很多条数据，因为一个复杂的SQL语句的执行是要拆分为很多步骤的，比如先访问表A，接着搞一个排序，然后来一个分组聚合，再访问表B，接着搞一个连接，类似这样子。

好，那么接下来我们就先来研究一下这个所谓的执行计划里包含的各个字段都是什么意思，首先是id这个东西

这个id呢，就是说每个SELECT都会对应一个id，其实说穿了，就是一个复杂的SQL里可能会有很多个SELECT，也可能会包含多条执行计划，每一条执行计划都会有一个唯一的id，这个没啥好说的。

select_type，顾名思义，说的就是这一条执行计划对应的查询是个什么查询类型，table就是表名，意思是要查询哪个表，partitions是表分区概念，这个所谓的分区表我们会在后面给大家讲，这里先不用太关注他。

type，就是比较关键了，针对当前这个表的访问方法，这个之前我们都讲过很多，比如说const、ref、range、index、all之类的，分别代表了使用聚簇索引、二级索引、全表扫描之类的访问方式。

possible_keys，这也很关键，他是跟type结合起来的，意思就是说你type确定访问方式了，那么到底有哪些索引是可供选择，可以使用的呢，这都会放这里。key，就是在possible_keys里实际选择的那个索引，而key_len就是索引的长度。

ref，就是使用某个字段的索引进行等值匹配搜索的时候，跟索引列进行等值匹配的那个目标值的一些信息。rows，是预估通过索引或者别的方式访问这个表的时候，大概可能会读取多少条数据。filtered，就是经过搜索条件过滤之后的剩余数据的百分比。extra是一些额外的信息，不是太重要。

好了，今天就先看到这里，明天我们继续讲解对真实的SQL语句分析得到的执行计划会长什么样子，让大家彻底能看懂执行计划。

End