链滴

知乎问答：MySQL 对于千万级的大表要怎么优化?

作者：seanlee

原文链接：https://ld246.com/article/1497318811539

来源网站：链滴

<blockquote>
该文章摘自于知乎问答

MySQL 对于千万级的大表要怎么优化？ </san>
回答者：zhuqz
</blockquote>

很多人第一反应是各种切分；我给的顺序是:</dv>
第一优化你的sql和索引；

第二加缓存，memcached,redis；

第三以上都做了后，还是慢，就做主从复制或主主复制读写分离，可以在应用层做，效率高，也可以用三方工具，第三方工具推荐360的atlas,其它的要么效不高，要么没人维护；

第四如果以上都做了还是慢，不要想着去做切分，mysq自带分区表，先试试这个，对你的应用是透明的，无需更改代码,但是sql语句是需要针对分区表做优的，sql条件中要带上分区条件的列，从而使查询定位到少量的分区上，否则就会扫描全部分区，另分区表还有一些坑，在这里就不多说了；

第五如果以上都做了，那就先做垂直拆分，其实就是根你模块的耦合度，将一个大的系统分为多个小的系统，也就是分布式系统；

第六才是水平切分，针对数据量大的表，这一步最麻烦最能考验技术水平，要选择一个合理的sharding key,为了有好的查询效率，表结构也要改动，做一定冗余，应用也要改，sql中尽量带sharding key，将数据定位到限定的表上去查，而不是扫描全部的；

mysql数据库一般都是按照这个步骤去演化的，成本也由低到高；

有人也许要说第一步优化sql和索引这还用说吗？的确，家都知道，但是很多情况下，这一步做的并不到位，甚至有的只做了根据sql去建索引，根本没对sql化（中枪了没？），除了最简单的增删改查外，想实现一个查询，可以写出很多种查询语句，不同的句，根据你选择的引擎、表中数据的分布情况、索引情况、数据库优化策略、查询中的锁策略等因素最终查询的效率相差很大；优化要从整体去考虑，有时你优化一条语句后，其它查询反而效率被降低，所以要取一个平衡点；即使精通mysql的话，除了纯技术面优化，还要根据业务面去优化sql语句，样才能达到最优效果；你敢说你的sql和索引已经是最优了吗?

再说一下不同引擎的优化，myisam读的效果好，写的率差，这和它数据存储格式，索引的指针和锁的策略有关的，它的数据是顺序存储的（innodb数据储方式是聚簇索引），他的索引btree上的节点是一个指向数据物理位置的指针，所以查找起来很快（innodb索引节点存的则是数据的主键，所以需要根据主键二次查找）；myisam锁是表锁，只有读之间是并发的，写写之间和读写之间（读和插入之间是可以并发的，去设置concurrent_insert参数，期执行表优化操作，更新操作就没有办法了）是串行的，所以写起来慢，并且默认的写优先级比读优级高，高到写操作来了后，可以马上插入到读操作前面去，如果批量写，会导致读请求饿死，所以要置读写优先级或设置多少写操作后执行读操作的策略;myisam不要使用查询时间太长的sql，如果策略用不当，也会导致写饿死，所以尽量去拆分查询效率低的sql,

innodb一般都是行锁，这个一般指的是sql用到索引的候，行锁是加在索引上的，不是加在数据记录上的，如果sql没有用到索引，仍然会锁定表,mysql的写之间是可以并发的，普通的select是不需要锁的，当查询的记录遇到锁时，用的是一致性的非锁定照读，也就是根据数据库隔离级别策略，会去读被锁定行的快照，其它更新或加锁读语句用的是当前，读取原始行；因为普通读与写不冲突，所以innodb不会出现读写饿死的情况，又因为在使用索引时候用的是行锁，锁的粒度小，竞争相同锁的情况就少，就增加了并发处理，所以并发读写的效率还很优秀的，问题在于索引查询后的根据主键的二次查找导致效率低；

ps:很奇怪，为什innodb的索引叶子节点存的是主键而是像mysism一样存数据的物理地址指针吗？如果存的是物理地址指针不就不需要二次查找了吗，这是我开始的疑惑，根据mysism和innodb数据存储方式的差异去想，你就会明白了，我就不费口舌了

所以innodb为了避免二次查找可以使用索引覆盖技术，法使用索引覆盖的，再延伸一下就是基于索引覆盖实现延迟关联；不知道什么是索引覆盖的，建议你论如何都要弄清楚它是怎么回事！

尽你所能去优化你的sql吧！说它成本低，却又是一项费费力的活，需要在技术与业务都熟悉的情况下，用心去优化才能做到最优，优化后的效果也是立竿见的！