《高性能MySQL》--高性能的索引策略

1. 独立的列：查询条件中的列，应该是单独的，例如：where id + 1 = 4，无法使用到索引；
2. 前缀索引和索引的选择性：前缀索引是数据库类型如blob，text，很长的varchar，这些无法直接使用索引的列（b-tree索引不允许），取出前若干字符，作为索引；索引的选择性是指不重复的索引值与所有的存储行数#T的比值，1/#T到1就是索引选择性的范围，1的时候效率最高；所以前缀的取值要适中，太短索引选择性太小，没效果，太长不利于节省空间；
3. 多列索引:建立多列索引时，如or操作会使优化器联合这些索引，多列索引中如果explain解释的语句中有联合索引，说明表结构或者查询语句需要优化；
4. 选择合适的索引列顺序，一般的索引列会将使用最频繁的列放在最左侧，这是因为索引中列的使用是从左开始的，但是除了这点也要考虑group by,order by的分组和排序，值的分布，索引的选择性等；
5. 聚簇索引：聚簇索引是将整行数据都存储在叶子页上，节点页存储索引，这样会导致①主键的查询效率增强，因为可以直接找到数据，二级索引（非聚簇索引）需要两次查询，多次IO，因为需要通过索引找到主键，然后通过主键去查找数据②如果聚簇索引不是按照顺序插入，会导致页分裂，从而使插入效率降低，因为更新的行要移动位置③密集型的记录效率更高，记录比较稀疏的，数据不连续的，全表扫描变慢；
6. 覆盖索引：将需要查询的列，用索引覆盖，这样就可以不需要回表查询；①索引的空间占用远小于记录，如果只需要索引查询，效率回提高；②索引是按顺序排列的，对于密集型IO，可以直接按顺序查询，这要比随机IO读取一条数据节省很多资源；③对于InnoDB这更加重要，因为它本身需要进行二次查询，在如果将二级主键覆盖在内，则不需要了；有些情况会导致覆盖索引无法使用，比如覆盖索引只能覆盖部分，这会导致无法使用，我们这时候可以使用嵌套查询先查出覆盖索引锁覆盖内容；
7. 使用索引扫描来做排序：排序有两种情况①排序操作②使用索引扫描做排序（标志：type列的值为index）；使用的条件：①索引覆盖查询；②索引顺序与order by一致；③前列是常量或者符合最左前缀原则；
8. 压缩（前缀）索引：myisam将前缀压缩节省空间，以来与前一个值的相同前缀，如果是倒序或者 随机扫描都会减少很多效率；
9. 冗余索引，重复索引和未使用的索引需要避免；
10. 索引和锁：索引可以使InnoDB的行级锁锁定更少的行。