Ready

浅谈索引及SQL查询优化

主讲人：蔡文豪

什么是索引？

索引是对数据库表中一列或多列的值进行排序的一种结构，使用索引可快速访问数据库表中的特定信息。

解释：生活中最经典的例子就是字典了，字典它有检索目录，如果我们不看偏旁、部首等目录检索页，直接在字典里面找的话，速度很慢很慢，这就相当于对数据表进行全表扫描，数据越多，扫描的时间越长，但是如果使用了检索目录，我们通常可以很快的查找到要找的字，这里的检索目录就是我要说的索引了。

索引的类型

1、BTREE

适用范围：适用于全键值、键值范围或前缀查找，其中键前缀查找只适用于根据最左前缀的查找，例如全值匹配、匹配最左前缀、匹配列前缀、匹配范围值等。

限制：

（1）如果不是按照索引的最左列开始查找，则无法使用索引。

（2）不能跳过索引中的列。

（3）如果查询中有某个列的范围查询，则其右边所有列都无法适用索引优化查找。

PS：索引列的顺序很重要

2、HASH

基于哈希表实现，只有精确匹配索引所有列的查询才有效。索引自身只需存储对应的哈希值，结构十分紧凑，这让哈希索引查找的速度非常快。

适用范围：

限制：（1）哈希索引只包含哈希值和行指针，而不存储字段值，索引不能使用索引中的值来避免读取行；（2）哈希索引数据并不是按照索引值顺序存储的，索引无法用于排序；（3）哈希索引不支持部分索引列匹配查找，因为它是使用索引列的全部内容来计算哈希值。（4）哈希索引只支持等值比较查询，包括=、IN()，不支持范围查询，例如WHERE price > 100。（5）访问哈希索引的数据非常快，除非有很多哈希冲突（不同的索引列值却有相同的哈希值）。

3、空间数据索引

MYISAM引擎支持空间索引，用作地理数据存储。

4、全文索引

特殊索引，查找文本中的关键词，不是直接比较索引中的值。

怎么建索引

（1）create index index_name on table_name (column1)
（2）alter table table_name add index index_name (column1,column2)
// 如果要指定索引类型，可以在语句后面加上 USING index_type，例如USING BTREE

建索引的原则

（1）最左前缀匹配原则，非常重要的原则，mysql会一直向右匹配直到遇到范围查询(>、<、between、like)就停止匹配，比如a = 1 and b = 2 and c > 3 and d = 4，如果建立(a,b,c,d)顺序的索引，d是用不到索引的，如果建立(a,b,d,c)的索引则，都可以用到，a,b,d的顺序可以任意调整。
（2）=和in可以乱序，比如a = 1 and b = 2 and c = 3 建立(a,b,c)索引可以任意顺序，mysql的查询优化器会帮你优化成索引可以识别的形式
（3）尽量选择区分度高的列作为索引,区分度的公式是count(distinct col)/count(*)，表示字段不重复的比例，比例越大我们扫描的记录数越少，唯一键的区分度是1，而一些状态、性别字段可能在大数据面前区分度就是0。一般在join的字段我要求是0.8以上，即平均1条扫描1.25条记录。
（4）索引列不能参与计算，保持列“干净”，比如from_unixtime(create_time) = ’2014-05-29’就不能使用到索引，原因很简单，b+树中存的都是数据表中的字段值，但进行检索时，需要把所有元素都应用函数才能比较，显然成本太大。所以语句应该写成create_time = unix_timestamp(’2014-05-29’);
（5）尽量的扩展索引，不要新建索引。比如表中已经有a的索引，现在要加(a,b)的索引，那么只需要修改原来的索引即可

不适合建索引

（1）字段值重复太多的列（例如性别），即使建了索引，但是每个索引覆盖的记录条数过多，对查询也没有多大提升，反而会因为维护索引而降低数据库性能
（2）有空值的列，索引NULL值需要多一个字节，而且遇到索引值为NULL时会使用全表扫描

建了索引不工作的情况

（1）在索引列上使用了函数或者运算
例如： select `id` from tb_test where length(name) > 1
（2）使用了LIKE语句通配符%匹配首字符，类似like '%%'
例如：select `id` from tb_test where name like '%张三%'
（3）字段编码或类型不一致
（4）IS NULL或者IS NOT NULL
（5）使用的索引列不是索引的前缀（一般我们都是建复合索引，即在多个列建）
例如，我们在表tb_test上新建了如下索引：create index idx_test on tb_test(id,name,addr)
以上索引IDX_TEST相当于建立了index(id)、index(id,name)、index(id,name,addr) 这3个索引。在SQL语句的where条件中单独使用name或addr时不会使用到该索引，必须使用id时才会使用到该索引。

优化

（1）有时候为了提高查询效率，我们往往牺牲范式（增加冗余字段）。
（2）避免使用NULL字段：一是NULL字段很难查询优化；二是NULL字段的索引需要额外空间；三是NULL字段的复合索引无效。
（3）避免使用select *或者count(*)来查询或统计数据，只查询需要的字段，只涉及到索引列的查询时最快的。