链滴

Lucene 的范围查询详解

作者：llh

原文链接：https://ld246.com/article/1493725053430

来源网站：链滴

<h2 id="Lucene的范围查询是怎样实现的呢">Lucene 的范围查询是怎样实现的呢</h2>
粗略来说，是两种方式：

根据 docId 获取 field 的值，和设定的范围进行比较过滤，得到满足范围条件的 docList。

根据范围条件从 term 列表过滤出满足条件的 term，把 term 组成 BooleanQuery，查询倒排表，得到满足范围条件的 docList。

第一种方式，从 docId 获取 value，需要用到 fieldCache，比较占内存，如果候选的 doc 数量常大（满足其他查询条件的 doc 非常多或没有其他查询条件），则要过滤的计算比较多，性能不会好。
第二种方式，如果过滤的范围比较大，则过滤出来的 term 非常多，要查的 term 很多，性能会差。
从源代码看，Lucene 的实现是这两种方式的组合，但第二种方式经过了优化，过滤出来的 term 的数量会非常少，性能非常好。
如果 field 没有索引，且有 docValues，则采取第一种方式（没有索引，是无法用第二种方式的因为没有对应的 term)
如果 field 有索引，则采取第二种方式,TrieField 的范围过滤
<h2 id="TrieField实现原理">TrieField 实现原理</h2>
数值类型 TrieXXXField(XXX 可为 Long、Int、Float、Double、Date 等)，如果需要范围查询一般要设置一个参数 precisionStep。这个参数的用途是，在索引阶段，会把一个数值，根据 precisionStep 进行精度截取，分为多个不同精度的 term 来存储。我们以一个 int 为例，一个 int 共 32 位，果 precisionStep 为 8，则每根据不同的精度，可以得到 4 个 term
term&11111111111111111111111111111111
term&11111111111111111111111100000000
term&11111111111111110000000000000000
term&11111111000000000000000000000000
这样，每个 term 都会变为 4 个 term，存储会增加很多，低精度的 term 重复率比较高，因此主要是倒排列表占用的空间会多很多。
查询时，范围的上界和下界也按照这种规则，划分为 4 段，间隔的两个段之间有 256 个 term，/p>
这样范围内的 term，就先取高低精度的，再取高精度的。最多一共有
256+2562+2562+256*2=1280 个 term，这些 term 再用 ConstantScoreQuery 来查询，比较打分计算。
下面举个例子来说明一个范围内的 term 是怎样得到的。
比如，范围[232420561,1399563675]，
232420561 的二进制是 00001101110110100111010011010001
1399563675 的二进制是 01010011011010111010010110011011
命中的 term 为
下界高精度
00001101110110100111010011010001
到
00001101110110100111010011111111
共 48 个 term
下界去掉 8 位精度
00001101110110100111010100000000
到
00001101110110101111111100000000
共 139 个 term
下界去掉 16 位精度
00001101110110110000000000000000

到
00001101111111110000000000000000
共 37 个 term
去掉 24 位精度
00001110000000000000000000000000
到
01010010000000000000000000000000
共 69 个 term
上界去掉 16 位精度
01010011000000000000000000000000
到
01010011011010100000000000000000
共 107 个 term
上界去掉 8 位精度
01010011011010110000000000000000
到
01010011011010111010010000000000
共 165 个 term
上界最高精度
01010011011010111010010100000000
到
01010011011010111010010110011011
共 156 个 term
总共要查询 721 个 term，使用 ConstantScoreQuery 来查询还是很快的。