黑客派

基于 Redis 的实时搜索

作者：changming

原文链接：https://hacpai.com/article/1472362146240

来源网站：黑客派

一直想把前期做的一种完全基于redis的实时搜索功能的实现方式总结一下，但是一直没有合适时间，今天终于可以坐下来把整个思路理一下了。
还是先看下整体效果，如下图所示。

<img title="serach" src="https://static.hacpai.com/images/img-loaing.svg" alt="search" width="1048" height="543" data-src="http://obu3fc5lw.bkt.clouddn.co/05c4d120a06845f7968875a7055d4aab.png">
该搜索是整个瑞知社区的一个功能项，瑞知社区类似于知乎，一个垂直问答社区，为了提高用户验，所以希望在搜索功能上可以体验更好，可以逐字匹配搜索，可以按照拼音(不是拼音首字母)模糊索。由于该项目没有考虑使用关系型数据库，所以决定把所有的索引数据都放在内存中，获取查找目过程完全基于内存，这样速度更快。
下面说说整个过程的实现思路，还有很多不完善的地方，敬请指教！

搜索的过程简单说就是建立索引和按照索引取值的过程，对需要搜索的目标数据进行索引，然后将索引结果放入redis，索引结果的数据结构可以根据实际情况来决定，比如我这里没有用到关系数据库，全部使用的是redi来存储，包括搜索目标数据，所以索引的数据结构是搜索关键词作为key，value中存入的是搜索目标redis中存入时的key值；如果你使用的关系型数据库，比如mysql，那么这里的value可能存的就是对数据库中记录的id值。当用户输入关键词进行搜索的时候，首先会对用户输入的关键词进行分词，再据分词结果查询索引，命中目标后，根据索引获取最终的查询目标数据。

以上就是整个搜的大致流程，下面我们来把他剖开来分析，看看每个过程如何实现。
<blockquote>
建立普通关键词索引
</blockquote>
当用户提出一个问题，或者回答问题时，系统都会实时的建立索引(这里可以考虑使用异步建立引，因为查询新内容不是实时的)，索引的数据结构为Sorted Set，score值全部设为1，这样索引排时就可以按照字母表顺序自然排序。key为分词后的值(使用的分词器是lucene中的IKAnalyzer)，valu保存的是对应搜索目标实体redis中存储的key值；
如果同种类型的多个关键词分词后有重复的词值，就将实体key值存在同一个分词集合中，如，某2个问题进行分词后，都包含【争夺】这个分词，那么这2个问题的实体key值都会作为index:question:[争夺]的value值。如下图所示：
<img title="general" src="https://static.hacpai.com/images/img-oading.svg" alt="general" width="1000" height="448" data-src="http://obu3fc5lw.bkt.cloudd.com/cba88804a5d1452cb59386d539c358d8.png">
<blockquote>
建立逐字匹配索引
</blockquote>
对于逐字匹配搜索的需求，我们的索引在建立时，也需要区别对待，当我们输入关键词时，并不望输入完整的关键词后才能检索出相关内容，而是输入部分关键词时立刻出现相关结果，也即是瞬时应效果。如下图所示：
<img title="prefix" src="https://static.hacpai.com/images/img-loadng.svg" alt="prefix" width="1152" height="612" data-src="http://obu3fc5lw.bkt.clouddn.com57e1234d003c4beca0b6adf03c2668d8.png">
那就要求我们在建立索引时，需要对每个分词后的值再按字或者字母建立索引值，这种索引我们

之为逐字匹配索引，或者前缀匹配索引，用Sorted Set来存储(key为index:prefix:key)，其值为分词字截取后的值，存储时其score值设为相同值1，那么所有的值会按照字母表的顺序进行排列，这样就得逐字匹配索引中相近分词值会集中排列；当你搜索关键词时会搜出一定范围内（该范围的值会直接响性能）的所有相似索引值。前缀匹配索引数据结构如下图所示：
<img title="prefix00" src="https://static.hacpai.com/images/img-oading.svg" alt="prefix00" width="598" height="898" data-src="http://obu3fc5lw.bkt.clouddncom/dd1ca6efadc547869fe8b73c3e2b1d1b.png">
对于中文，前缀匹配索引数据结构如下图所示
<img title="prefixcn" src="https://static.hacpai.com/images/img-lading.svg" alt="prefixcn" width="976" height="796" data-src="http://obu3fc5lw.bkt.clouddn.om/ed897866821c4e64aef96fd6e122acb8.png">
<blockquote>
建立中文拼音索引
</blockquote>
对于中文用户，希望通过汉语拼音直接搜索，那就需要在对中文分词结果索引的同时，进行汉语音的转译，然后将转译后的汉语拼音作为key值，所有同音词的中文分词都作为value，存储在set中这种方式可以确保检索出汉语拼音对应的所有中文分词。数据结构如下图所示：
<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js"></script>
<!-- 黑客派PC帖子内嵌-展示 -->
<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342" data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></in>
<script>
    (adsbygoogle = window.adsbygoogle || []).push({});
</script>
<img title="pinyin" src="https://static.hacpai.com/images/img-loadng.svg" alt="pinyin" width="764" height="498" data-src="http://obu3fc5lw.bkt.clouddn.com/2105063d6ed4df1b703464637a04bc5.png">
从以上我们可以看出，在建立索引数据时，我们对同一个分词进行了三种索引。
<blockquote>
搜索过程
</blockquote>
当用户在输入每个关键词的每个字时，首先对输入的关键词进行分词操作，然后会同时检索前缀中文拼音索引，获取该关键词以及相关类似的关键词，然后除掉重复的关键词，最后根据搜索目标的型，构建【普通关键词索引】的key值，求并集获取结果。
我们以一个例子来说明整个过程，比如我们输入的关键词是“南京”，经过检索前缀和中文拼音引后，返回的关键词如下图所示：
<img title="result00" src="https://static.hacpai.com/images/img-lading.svg" alt="result00" width="740" height="428" data-src="http://obu3fc5lw.bkt.clouddn.om/1e9d450b54204491b722efbfa50200b3.png">
然后根据搜索目标类型，构建的【普通关键词索引】的key值如下图所示：
<img title="result01" src="https://static.hacpai.com/images/img-lading.svg" alt="result01" width="906" height="466" data-src="http://obu3fc5lw.bkt.clouddn.

om/cdfcf2a0b24140b6955d31134170180f.png">
这样，就可以根据question类型的索引key值找到对应的question实体的对应key，求并集，最返回json数据给前端即可。整个流程就是这样。
最终结果如下图所示：
<img title="result" src="https://static.hacpai.com/images/img-loading.svg" alt="result" width="1196" height="725" data-src="http://obu3fc5lw.bkt.clouddn.com/7bb8ac5b17c40aa9840d68916fb1e74.png">
<blockquote>
关于分页
</blockquote>
对于已经排好序的结果集，使用Sorted Set的zrevrange命令即可按照scrore的值逆序排序。</p

<blockquote>
关于性能
</blockquote>
性能方面，普通关键词索引+前缀匹配索引+拼音索引的总和为60万+，而所有问题和回答有100+，在我的mbp(8g,core i7)上面搜索体验还是很流畅的 。
<blockquote>
关于优化
</blockquote>
在前缀匹配时，是根据指定获取一定范围内的相似结果，这个范围值对性能影响很大；
是不是需要对关键词同时建立三种索引；
在搜索的准确性上，需要对分词器进行优化；
灵活设定结果集的排序字段 。