05 | 倒排索引：如何从海量数据中查询同时带有“极”和“客”的唐诗?

2020-04-01 陈东

讲述：陈东
时长 14:12 大小 13.01M

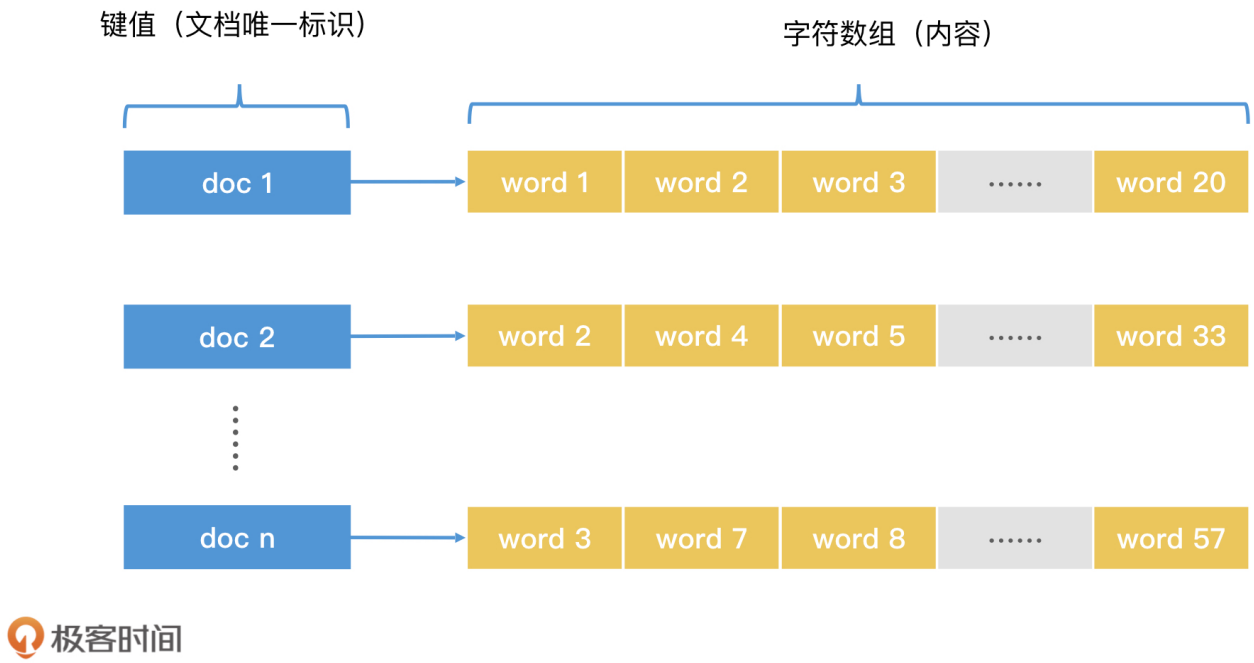
你好，我是陈东。

试想这样一个场景：假设你已经熟读唐诗 300 首了。这个时候，如果我给你一首诗的题目，你可以马上背出这首诗的内容吗？相信你一定可以的。但是如果我问你，有哪些诗中同时包含了“极”字和“客”字？你就不见得能立刻回答出来了。你需要在头脑中一首诗一首诗地回忆，并判断每一首诗的内容是否同时包含了“极”字和“客”字。很显然，第二个问题的难度比第一个问题大得多。

那从程序设计的角度来看，这两个问题对应的检索过程又有什么不同呢？今天，我们就一起来聊一聊，两个非常常见又非常重要的检索技术：正排索引和倒排索引。

什么是倒排索引？

我们先来看比较简单的那个问题：给出一首诗的题目，马上背出内容。这其实就是一个典型的键值查询场景。针对这个场景，我们可以给每首诗一个唯一的编号作为 ID，然后使用哈希表将诗的 ID 作为键（Key），把诗的内容作为键对应的值（Value）。这样，我们就能在 O(1) 的时间代价内，完成对指定 key 的检索。这样一个以对象的唯一 ID 为 key 的哈希索引结构，叫作**正排索引**（Forward Index）。

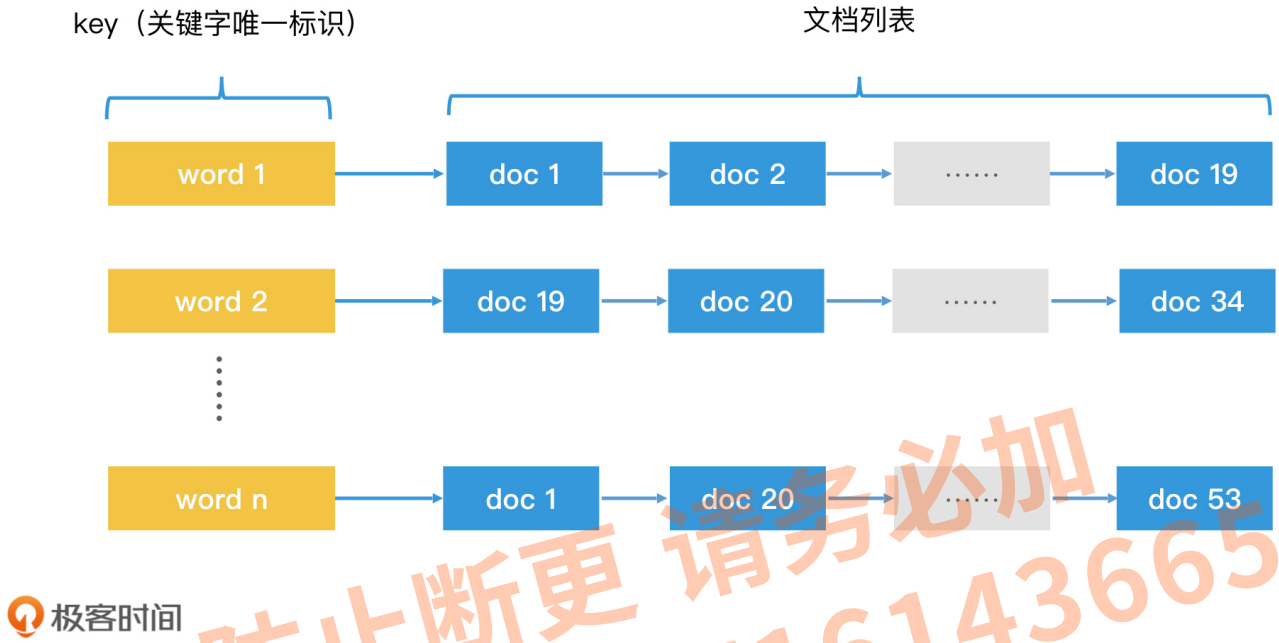


哈希表存储所有诗

一般来说，我们会遍历哈希表，遍历的时间代价是 O(n)。在遍历过程中，对于遇到的每一个元素也就是每一首诗，我们需要遍历这首诗中的每一个字符，才能判断是否包含“极”字和“客”字。假设每首诗的平均长度是 k，那遍历一首诗的时间代价就是 O(k)。从这个分析中我们可以发现，这个检索过程全部都是遍历，因此时间代价非常高。对此，有什么优化方法吗？

我们先来分析一下这两个场景。我们会发现，“根据题目查找内容”和“根据关键字查找题目”，这两个问题其实是完全相反的。既然完全相反，那我们能否“反着”建立一个哈希表来帮助我们查找呢？也就是说，如果我们以关键字作为 key 建立哈希表，是不是问题就解决了呢？接下来，我们就试着操作一下。

我们将每个关键字当作 key，将包含了这个关键字的诗的列表当作存储的内容。这样，我们就建立了一个哈希表，根据关键字来查询这个哈希表，在 O(1) 的时间内，我们就能得到包含该关键字的文档列表。这种根据具体内容或属性反过来索引文档标题的结构，我们就叫它**倒排索引**（Inverted Index）。在倒排索引中，key 的集合叫作**字典**（Dictionary），一个 key 后面对应的记录集合叫作**记录列表**（Posting List）。



倒排索引

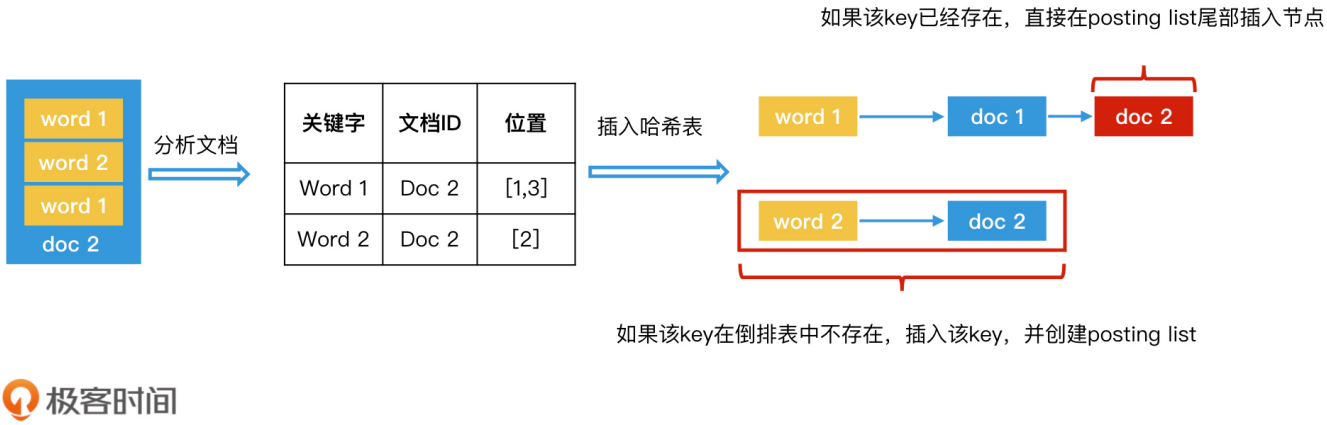
如何创建倒排索索引？

前面我们介绍了倒排索引的概念，那创建一个倒排索引的过程究竟是怎样的呢？我把这个过程总结成了以下步骤。

1. 给每个文档编号，作为其唯一的标识，并且排好序，然后开始遍历文档（为什么要先排序，然后再遍历文档呢？你可以先想一下，后面我们会解释）。
2. 解析当前文档中的每个关键字，生成 < 关键字，文档 ID，关键字位置 > 这样的数据对。为什么要记录关键字位置这个信息呢？因为在许多检索场景中，都需要显示关键字前后的内容，比如，在组合查询时，我们要判断多个关键字之间是否足够近。所以我们需要记录位置信息，以方便提取相应关键字的位置。
3. 将关键字作为 key 插入哈希表。如果哈希表中已经有这个 key 了，我们就在对应的 posting list 后面追加节点，记录该文档 ID（关键字的位置信息如果需要，也可以一并

记录在节点中）；如果哈希表中还没有这个 key，我们就直接插入该 key，并创建 posting list 和对应节点。

4. 重复第 2 步和第 3 步，处理完所有文档，完成倒排索引的创建。



将一个文档解析并加入倒排索引

如何查询同时含有“极”字和“客”字两个 key 的文档？

如果只是查询包含“极”或者“客”这样单个字的文档，我们直接以查询的字作为 key 去倒排索引表中检索，得到的 posting list 就是结果了。但是，如果我们的目的是要查询同时包含“极”和“客”这两个字的文档，那我们该如何操作呢？

我们可以先分别用两个 key 去倒排索引中检索，这样会得到两个不同的 posting list：A 和 B。A 中的文档都包含了“极”字，B 中文档都包含了“客”字。那么，如果一个文档既出现在 A 中，又出现在 B 中，它是不是就同时包含了这两个字呢？按照这个思路，我们只需查找出 A 和 B 的公共元素即可。

那么问题来了，我们该如何在 A 和 B 这两个链表中查找出公共元素呢？如果 A 和 B 都是无序链表，那我们只能将 A 链表和 B 链表中的每个元素分别比对一次，这个时间代价是 O(m*n)。但是，如果两个链表都是有序的，我们就可以用归并排序的方法来遍历 A 和 B 两个链表，时间代价会降低为 O(m + n)，其中 m 是链表 A 的长度，n 是链表 B 的长度。

我把链表归并的过程总结成了 3 个步骤，你可以结合我在图片中给出的例子来理解。

第 1 步，使用指针 p1 和 p2 分别指向有序链表 A 和 B 的第一个元素。

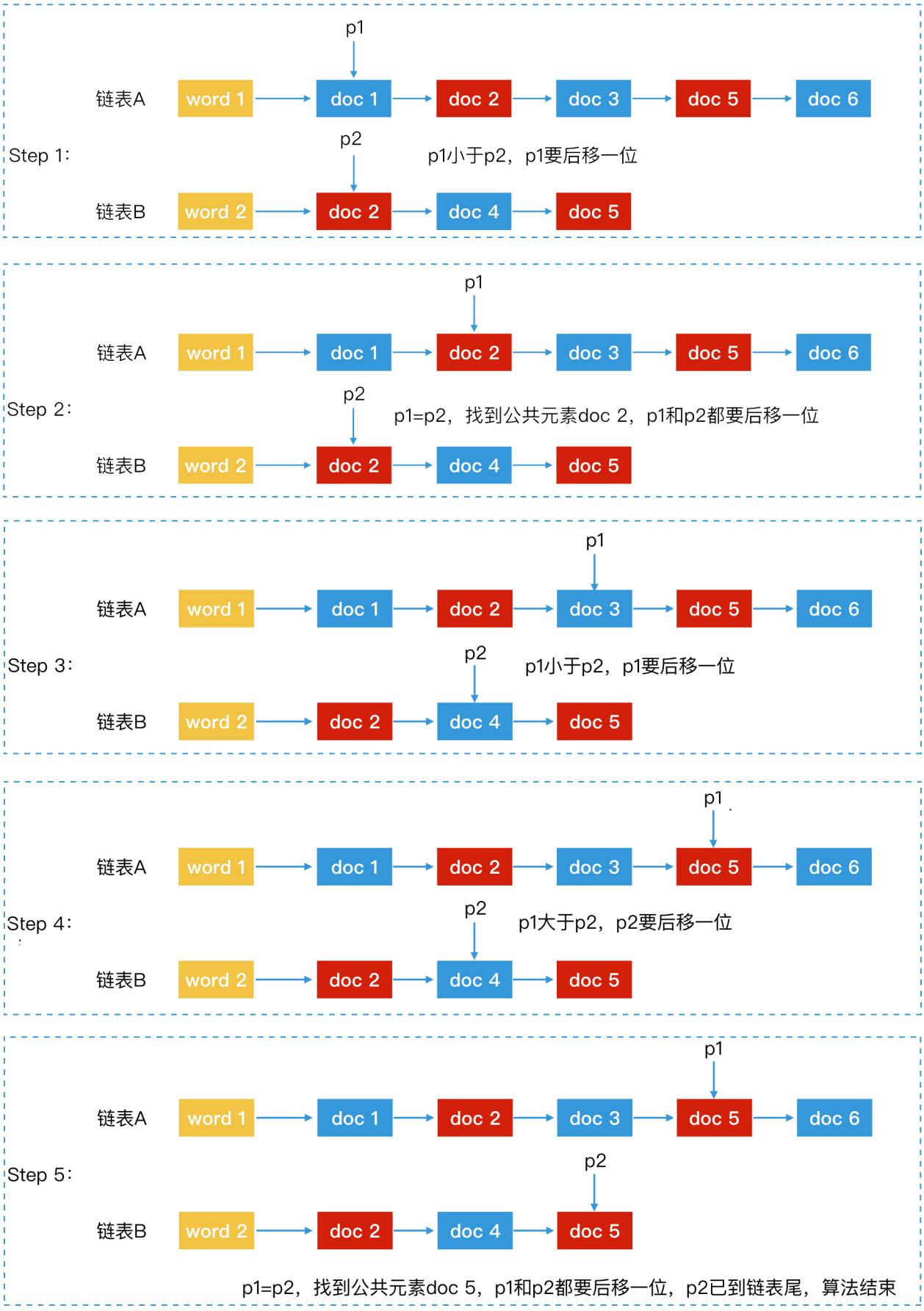
第 2 步，对比 p1 和 p2 指向的节点是否相同，这时会出现 3 种情况：

两者的 id 相同，说明该节点为公共元素，直接将该节点加入归并结果。然后，p1 和 p2 要同时后移，指向下一个元素；

p1 元素的 id 小于 p2 元素的 id，p1 后移，指向 A 链表中下一个元素；

p1 元素的 id 大于 p2 元素的 id，p2 后移，指向 B 链表中下一个元素。

第 3 步，重复第 2 步，直到 p1 或 p2 移动到链表尾为止。

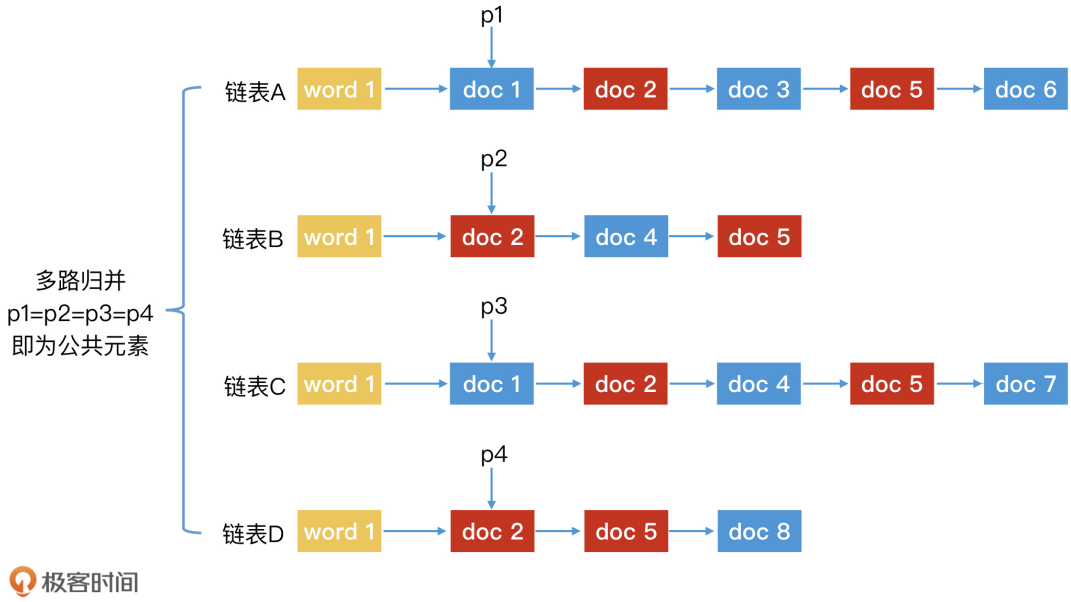


极客时间

链表归并提取公共元素例子

那对于**两个 key** 的联合查询来说，除了有“同时存在”这样的场景以外，其实还有很多联合查询的实际例子。比如说，我们可以查询包含“极”**或**“客”字的诗，也可以查询包含“极”**且不包含**“客”的诗。这些场景分别对应着集合合并中的交集、并集和差集问题。它们的具体实现方法和“同时存在”的实现方法差不多，也是通过遍历链表对比的方式来完成的。如果感兴趣的话，你可以自己来实现看看，这里我就不再多做阐述了。

此外，在实际应用中，我们可能还需要对**多个 key** 进行联合查询。比如说，要查询同时包含“极”“客”“时”“间”四个字的诗。这个时候，我们利用多路归并的方法，同时遍历这四个关键词对应的 posting list 即可。实现过程如下图所示。



多路归并

重点回顾

好了，今天的内容就先讲到这里。你会发现，倒排索引的核心其实并不复杂，它的具体实现其实是哈希表，只是它不是将文档 ID 或者题目作为 key，而是反过来，通过将内容或者属性作为 key 来存储对应的文档列表，使得我们能在 O(1) 的时间代价内完成查询。

尽管原理并不复杂，但是倒排索引是许多检索引擎的核心。比如说，数据库的全文索引功能、搜索引擎的索引、广告引擎和推荐引擎，都使用了倒排索引技术来实现检索功能。因此，这一讲的内容我也希望你能好好理解消化，打好扎实的基础。

课堂讨论

今天的内容实践性比较强，你可以结合下面这道课堂讨论题，动手试一试，加深理解。

对于一个检索系统而言，除了根据关键字查询文档，还可能有其他的查询需求。比如说，我们希望查询李白都写了哪些诗。也就是说，如何在“根据内容查询”的基础上，同时支持“根据作者查询”，我们该怎么做呢？

欢迎在留言区畅所欲言，说出你的思考过程和最终答案。如果有收获，也欢迎把这篇文章分享给你的朋友。

上一篇　04 | 状态检索：如何快速判断一个用户是否存在？

精选留言 (12)

写留言

范闲
2020-04-01

拉链的是倒排索引在数据量不大的情况下应该很好？如果数量上去了，要改成跳表了吧？如果跳表也支撑不下去了呢？

展开

作者回复: 你的问题我理解有两点:

1.检索效率问题。

2.存储空间问题。

关于检索效率问题，马上会出来两篇热腾腾的加餐，和你聊聊怎么进行倒排索引的检索加速。

关于存储空间问题，我接下来在进阶篇会开始介绍解决方案。

1 1

刘凯

2020-04-01

我能想到的方案是在数据库中建两张表，一张保存唯一索引的分词表，一张保存对应分词的文章id，这样是不是弱爆了。这样变成关系型数据库的查找了，而且外键表行数会暴增。脱离了老师说的hash. 老师你说的hash加链表我也看懂了，但持久话如何实现呢

展开

作者回复: 其实你已经渐渐掌握了核心原理了！如果你能用数据库那就简单了，因为数据库中的全文索引功能，就是倒排索引的具体实现！

你只需要建立一张表，一列是文章id，一列是文章内容，然后指定对文章内容这一列建立全文索引，那接下来就可以用sql语句直接检索了。

明翼

2020-04-01

老师对于邮件中敏感词检测适不适合用倒排索引那，用的话可能每个邮件都只要检测一次，不用直接搜索可能又找不到近义词

展开

作者回复: 就像你说的，邮件只需要检测一次，因此对邮件做倒排索引并不适用。而且倒排索引也解决不了近义词问题。

邮件敏感词检测一般是这样的思路:

1.准备一个敏感词字典。

2.遍历邮件，提取关键词，去敏感词字典中查找，找到了就说明邮件有敏感词。

这里的核心问题是如何提取关键词和如何在敏感词字典中查询。

一种方式是用哈希表存敏感词字典，然后用分词工具从邮件中提取关键字，然后去字典中查。

另一种方式是trie树来实现敏感词字典，然后逐字扫描邮件，用当前字符在trie树中查找。

不过，这两种方式都无法解决近义词，或者各种刻意替换字符的场景。要想解决这种问题，要么提供近义词字典，要么得使用大量数据进行训练和学习，用机器学习进行打分，将可疑的高分词找出来。

其实这种近义词处理方案，和搜索引擎解决近义词和查询纠错的过程很像。我在搜索引擎那篇里面会介绍。

一步
2020-04-01

倒排索引的核心就是关键词的提取，也就是如何合理的对内容进行分词

作者回复: 分词是第一步，这样就有了倒排索引的key。至于有了倒排索引以后，如何提高检索效率，马上会有加餐为你揭晓

李恒达
2020-04-01

老师，我有个疑问，为了实现根据关键词获取数据的功能，是不是需要在正常的表存储的基础上，再额外维护这样一个倒排索引？那这种在关键词不明确的情况下是不是就不会有这个东西了？

作者回复: 你的思考很好！的确是这样的。你可以回到开头背唐诗的场景。如果只要求给题目背内容，那么是只需要正排索引就好。不需要倒排索引。
倒排索引是用在需要根据部分信息或者属性去反查出数据主体的场景中。搜索引擎就是典型的应用场景，因为我们只知道我们想找什么关键字，而不知道哪些网页有这些关键字，因此需要倒排索引。数据库也一样，很多时候，我们去数据库中查找，也不是直接找id，而是用where去限定一些属性和字段。因此，你会发现，根据我们关心的属性去寻找主体，这种需求其实很常见，这些场景就可以用倒排索引了。

刘凯
2020-04-01

老师，这样的倒排索引能介绍一些语言的工具名称吗？我见过的有. net中使用的盘古分词，其他语言有没有

作者回复: “倒排索引”本身不是一个工具，它更多是一种检索思想和技术。因此许多编程语言中没有现成的“倒排索引”，不过我们可以自己实现它:用一个哈希表进行key的查找，然后哈希表中的value存一个数组或链表作为posting list，这样你就可以用任何语言实现倒排索引了。
然后，你提到的“盘古分词”，那个是分词工具，类似的分词工具还有许多，比如说jieba，pkus

eg，thulac等。
你完全可以找一个分词器，然后利用我文章中介绍的构建倒排索引的思路，自己做出来一个倒排索引。

努力努力再努力Xmn
2020-04-01

老师在文章中提到了在构建倒排索引过程中要记录位置信息，我想可不可以同时检索 李 字和 白 字，然后判断二者的位置是否相邻？希望老师解答。

展开

作者回复: 很好，你关注到了“李白”的分词问题了！
对于如何确定一个词，常见的做法是使用分词技术，将“李白”作为一个整体处理。这样检索性能也最好。
而你提的这个方案，是在分词技术无效的情况下，搜索引擎会采用的方案，它会根据位置信息进行短语查询，查出来的“李”和“白”是有序相邻的，优先级最高，位置越远的，优先级越低。
通过描述，你也能体会到这样的效率的确没有直接处理一个整体高。因此，分词也是很重要的技术。

范闲
2020-04-01

posting list里面可以增加一个author的信息，word,author,pos,id。这样查完以后只需要做个集合检查即可。

作者回复: 很正确！posting list是可以根据需要放更多复杂的信息的，从而帮助我们解决更多复杂的需求。
不过也要注意一个度，如果把所有信息都放posting list中，依赖于集合检查，那就变成了遍历查找的效率了。因此，在合适的场景下，有时候也可以为author独立建一个新索引。

pedro
2020-04-01

回答一下问题，支持作者查询其实就和查询内容一样，但是觉得新建哈希表比较合适，这样内容和作者查询在不同的posting list中，分别进行归并，查询完毕后，再根据作者和内容的权重进行打分，将分数最高放在结果首位。

我自己也有个问题，如果倒排索引非常大，内存不可能全部载入所有索引，那么如何边取部分索引再归并了？😊

展开

作者回复: 1.新建一个倒排索引是不错的方案。除了新建索引，还可以在posting list的记录中加域区分。比如用两个比特位，第一个表示是否是作者，第二个表示是否是内容。你所提到的按作者和内容权重进行打分的方案，用这个方法更容易实现。
2.内存放不下，有三个思路:1.通过压缩，全塞进内存；2.放磁盘上，用b+树或分层跳表处理；3.分布式，分片后全放内存。进阶篇中我会介绍这些方案。

峰
2020-04-01

感觉老师这个问题有点水到渠成的感觉，既然讲了倒排，一个很明显的答案就是把作者也当检索内容一样处理构建索引，对这种类似kv m 查的问题，这应该是最优的一个手段，具体的方案应该考虑如何压缩表示的问题，不明显的答案想不到哈哈哈。

既然提到索引给大家分享一个观点，索引是什么，从机器学习的角度上看，它其实是检...

展开

作者回复: 哈哈，用“口音牛逼”作为我的属性，就可以通过这个标签将我检索出来了。😁
明显的答案水到渠成，那我们来加入一点细节吧。许多诗的内容里都有李白的名字，比如杜甫就写了许多思念李白的诗。那我们用作者“李白”构建倒排索引，和用内容中的“李白”做倒排索引，会不会有冲突?会不会直接把杜甫的诗给召回了?
ps:机器学习构建索引就是我们现在在做的事情

1

Kǎfкā²⁰²⁰
2020-04-01

在关键字为key所在文档为posting list的基础上，再加以作者名为key，posting list为作者诗集的索引

作者回复: 很好，你提到了“以关键字为key”和“以作者名为key”，我们是可以用两个不同的key来区分作者“李白”和内容中的“李白”。
这样就能解决“明明想搜的是李白写的诗，结果出来了杜甫写李白的诗”这样的问题

李跃爱学习

2020-04-01

作者看做是文档的一个属性，建立属性倒排索引

展开

作者回复: 很好。对于属性建立倒排索引是正确的。

你可以再思考一些细节:如果有一些诗的内容里也有“李白”这个关键词，比如杜甫的诗。

那么作者“李白”对应的posting list，和内容中的“李白”对应的posting list是否会冲突？可以怎么处理？