9999-0-es倒排索引技术原理

Created by 杨超, last modified on 2018 Aug 13

参考文档

https://www.cnblogs.com/jiu0821/p/7688628.html

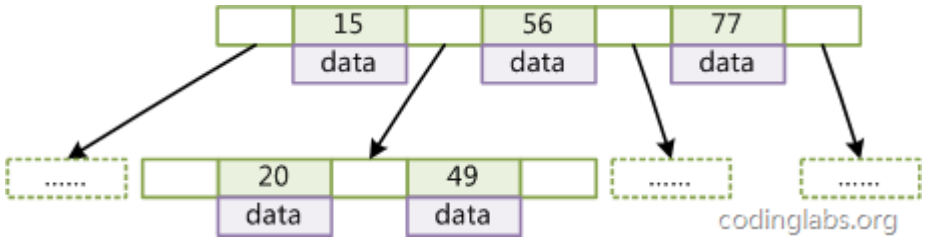
http://www.cnblogs.com/jiu0821/p/7688669.html

原理综述

1-倒排表

搜索为何快于普通的关系型数据库，本质上 是搜索的结构是 倒排表 而 db是用的 B+树

B+树 示例 – 比较基础，不赘述



倒排表方案 – 存储表

ID	Name	Age	Sex
1	Kate	24	Female
2	John	24	Male
3	Bill	29	Male

倒排表--索引，假设每个字段都需要索引，那么就会有3个倒排表，see?

Name:

Term	Posting List
Kate	1
John	2
Bill	3

Age:

Term	Posting List
24	[1,2]
29	3

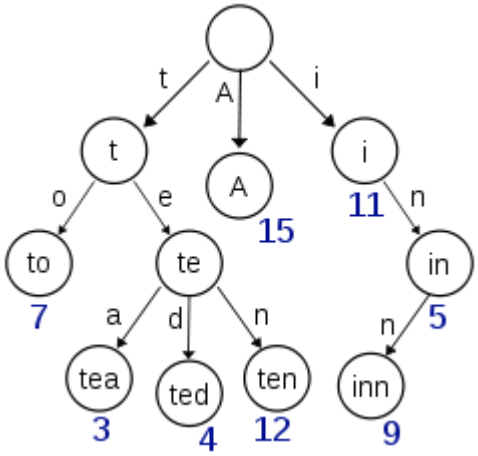
Sex:

Term	Posting List
Female	1
Male	[2,3]

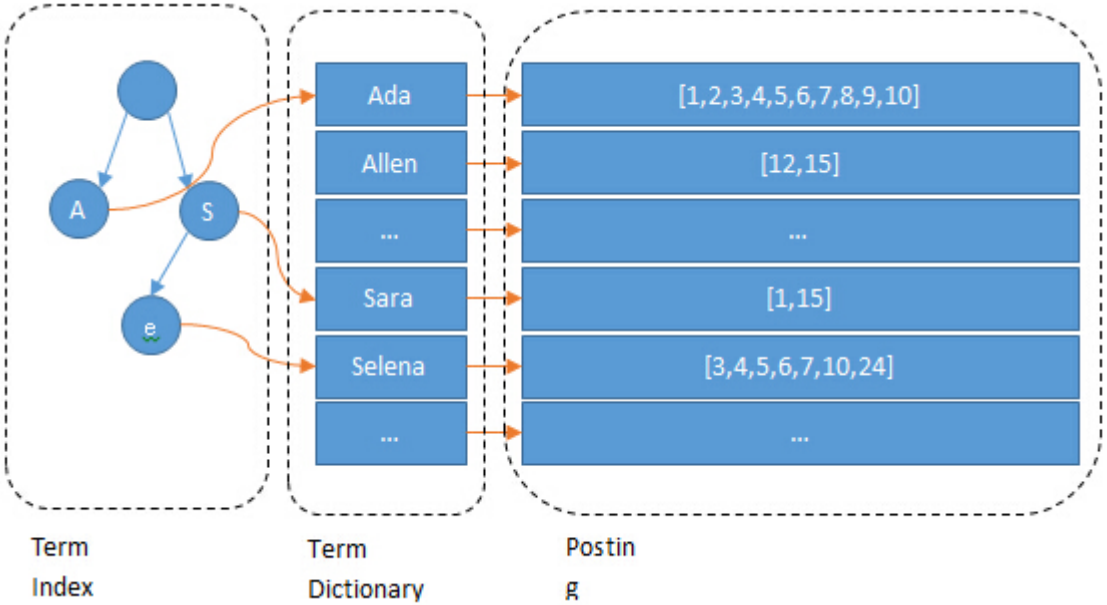
2-字典树

Term Index

B-Tree通过减少磁盘寻道次数来提高查询性能，Elasticsearch也是采用同样的思路，直接通过内存查找term，不读磁盘，但是如果term太多，term dictionary也会很大，放内存不现实，于是有了**Term Index**，就像字典里的索引页一样，A开头的有哪些term，分别在哪页，可以理解term index是一颗树：



这棵树不会包含所有的term，它包含的是term的一些前缀。通过term index可以快速地定位到term dictionary的某个offset，然后从这个位置再往后顺序查找。



所以term index不需要存下所有的term，而仅仅是他们的一些前缀与Term Dictionary的block之间的映射关系，再结合FST(Finite State Transducers)的压缩技术，可以使term index缓存到内存中。从term index查到对应的term dictionary的block位置之后，再去磁盘上找term，大大减少了磁盘随机读的次数。

走到这里，基本上就知道了es的索引结构，后续的FST 也就是有穷状态机 只是辅助的一种压缩技术，但是term index会这样存在

3-FST技术

这个不影响对倒排的理解，我单开一章，有兴趣可以看看 9999-1-FST

总结和思考

es的索引思路

> 将磁盘里的东西尽量搬进内存，减少磁盘随机读取次数(同时也利用磁盘顺序读特性)，结合各种奇技淫巧的压缩算法，用无所不用其极的态度使用内存。