前言

我是邢开春，有过几年搜索引擎开发的经验，现在搞了一个网站，一是交流分享，二是希望利用自己的工作经验赚取一些外快。现在已将自己多年搜索引擎的经验做成了一套视频（包含 lucene、solr、elasticsearch），这套视频从入门使用到深入原理，面面俱到，可以帮助初学者快速入门，帮助使用者务实基础。本人的课程(博客文章、视频教程等)力求在知识上直击本质。

什么时候应该考虑使用搜索引擎?

第一点：有搜索功能的需求

想为 App、网站添加类似”百度搜索的服务”搜索自家资源；想为用户提供类似”百度搜索的服务”搜索自家资源。

第二点: 对搜索速度有要求

有时会遇到 mysql 数据库搜索的速度过慢。举一个例子，假设你有一亿篇文章，使用数据库提供的 like ‘%关键词%’，将会耗时几小时才能查找出结果。而搜索引擎则能做到几十毫秒查询到结果。

【mysql 的 like 查询为什么慢】

mysql 数据库的 like 查询会顺序扫描每一个数据记录，所谓顺序扫描，就是一个记录一个记录的查询扫描，对于每一个记录，从头看到尾，一个字符一个字符的检测是否包含被查询字符串。假设每一万记录耗时一秒，扫描一亿条记录，约耗时一万秒，约三小时。

第三点：对搜索结果的质量有要求

假设我们是一电商平台，用户搜索关键词”华为手机”搜出 5000 个商品。因为商品数据太多，可以用分页技术把数据展示给用户，但是一般用户也就查看前三页，总计几十个的商品。这说明只有极小一部分商品数据才能被用户看到，而 5000 个商品可能只有一小部分可能被用户感兴趣，如何尽量让用户可能感兴趣的商品排在前面，从而让用户看到他们可能感兴趣的商品？如果排在前面展示的商品都是用户不感兴趣的，而用户感兴趣的商品都展示在页码靠后的页面，那用户体验太糟糕了。

[说句题外话，对于不同用户，由于身份，阅历不同，对待同一商品的满意程度是不一致的。当然，这篇文章不讨论这个问题]。

Lucene 会通过评分模型为关键词与每一个被搜索出来商品一一计算一个相关度，相关度越大说明关键词与商品之间的关系越大。然后商品按照与关键词的相关度排序，和搜索关键词关系深的排在前面，越靠前的商品与搜索关键词相关度越高，

关系越大，而关键词又是用户输入的，自然靠前的商品越有可能是用户感兴趣的商品。

mysql、oracle 等数据库则不能很好地完成这个功能。

lucene 是什么

Lucene 是一款开源的、高性能的、可拓展的信息检索 Java 工具库(也就是说 lucene 是提供搜索功能的 Java jar 包)。具体是下图的东西，也可以下载下来看一眼是什么。(下载链接)。将 lucene 引入 Java 项目中，你可以利用它做出类似百度、google 提供的搜索服务。

lucene 单纯的是一个 jar 包。它仅仅只提供搜索最核心的的开发接口【后面有讲:创建索引与利用索引进行搜索】，说直白点，它只是一个 java 语言的开发工具 jar 包。它没有用户操作界面，只支持 Java 编程语言开发，缺少其它编程语言的 sdk[链接]，不提供分布式搜索服务。

lucene、mysql、oracle 的区别？

lucene 是搜索引擎，mysql、oracle 是关系型数据库，问 lucene 与 mysql 的区别，不如问搜索引擎与关系型数据库的区别

	搜索引擎	关系型数据库
搜索速度	亿级数据量，可以保持在几十毫秒内查询出结果	亿级数据量，不能很好支持查询 like 查询
搜索质量	搜索质量高。有评分模型，使得搜索结果中与搜索词相关度较高的结果排前面返回	搜索质量较差。数据库一般不关注搜索质量。数据库着重于对数据资源的管理。具体来说是增删改查，ACID。一般，数据库提供按照数字大小，字符编码进行排序的功能。
维护数据与数据之间的关系的能力	弱	专业

Solr 是什么？ElasticSearch 是什么？

solr 是什么？(链接)elasticsearch 是什么？(链接)(视频)

之所以把 solr 与 elasticsearch 放在一起说，是因为它们的底层都是 lucene，再 lucene 提供的功能上进行了二次开发，附加了众多的功能，是搜索引擎的一个完整解决方案，属于企业级[我理解的企业级，开箱即用]的搜索引擎平台。它们都提供用户操作控制台；提供 RESTful 风格编程接口，也就是说它可以被任何编程语言使用；提供分布式搜索功能，不用再担心数据增多。

另外说一句，它们的关系好比是 oracle 与 mysql(它们都是关系型数据库)。

lucene 与 solr(elasticsearch)的区别？

	Lucene	Solr/Elasticsearch
是 Java jar 包	是	不是
是 Web 应用	不是	是
提供用户操作控制台	不提供	提供
提供 RESTful 风格编程接口	不提供	提供
提供分布式搜索功能	不提供	提供

既然如上所说，solr/Elasticsearch 功能比 lucene 强大太多，我还需要学习 lucene 吗？

这里可以给肯定的答复，如果只满足于简单使用搜索引擎这一初级阶段，跳过 lucene，直接学习使用 Solr/ElasticSearch 也是可以的。但是想要得心应手，必定要深入学习 lucene，lucene 是高楼大厦的地基，地基打的好，楼房才能又高又稳。

另外，也有 lucene 适用，Solr/ElasticSearch 反而不适用的场景。

例如:为手机文件管理 app 提供，根据搜索词，搜索内容包含搜索词的文件列表。这里用内嵌的 jar 的方式开发反而更为恰当。如果用 Solr、ElasticSearch 开发，你需要搭建 Solr、ElasticSearch 服务，我们不能在手机用户手机上搭建服务，需要在服务器上搭建，用户的数据需要上传到服务器端，才能为用户提供服务，特别繁杂。

我的意见是一定要深入学习 lucene，但在使用选择上尽量使用 Solr/ElasticSearch，能不用 Lucene 就别单独使用 Lucene。单说一点，随着数据量上升，单机已经承受不住性能，你需要为 lucene 做二次开发，使其支持分布式。而 Solr/ElasticSearch 作为成熟的企业级搜索引擎解决方案，早已经提供了分布式。

solr 与 elasticsearch 的区别？

既然 solr 与 elasticsearch 都是企业级的搜索平台，它们的相同点有很多：它们的底层都是 lucene，它们都是企业级搜索平台，它们都提供分布式搜索功能，都能轻易处理亿级的数据量，都提供实时搜索功能.....

那它们有什么区别呢？据我的观察，它们最大的区别是生态。solr 是 apache 开源的软件，它开源，完全免费的企业级搜索平台。而 elasticsearc 是公司企业在维护推广，它部分产品收费，同时企业围绕着 elasticsearch 推出了一套完善的数据分析框架，也就是大名鼎鼎的 ELK。

	Solr	ElasticSearch
企业级搜索平台	是	是
是 Web 应用	是	是
提供用户操作控制台	提供	提供
提供 RESTful 风格编程接口	提供	提供
提供分布式搜索功能	提供	提供
提供安全认证	提供	提供
组织	大名鼎鼎的 apache	elasticsearc 公司
开源	完全开源	部分开源
产品收费	完全免费	部分收费
生态	Lucene/Solr	ELK 等

lucene 原理是什么？

这个问题问的不够准确，一般人其实想问的问题包含两个:

一是 lucene 快速检索的原理是什么？

二是 lucene 的搜索效果为什么好？

第一个问题: lucene 快速检索的原理是什么？

索引的概念

假设，我们有一个问题，如何在一亿个从小到大排序好的有序数据集中查找一个特定的数存不存在？

基本原理是：首先在有序的数字中找到中值(最中间的那个数的值)，将要查找的目标与中值进行比较，如果目标等于中值，则数据集中存在要查找的数；如果目标小于中值，若目标存在，则一定在前半部分数据中(这样，我们待查找的数据

范围缩小了一半，在一亿条数据中查找和在五千万条数据中查找，差别还是蛮大的)，如果目标大于中值，则在后半部分查找；

如何在剩下的五千万条中查找呢？

同样的方法，先找这五千万条数据中的中值，如果目标等于中值，则数据集中存在要查找的数；如果目标小于中值，则在前半部分找；如果目标大于中值，则在后半部分找。

如何在剩下的二千五百万条数据中查找呢？

同样的方法，先找中值，如果目标等于中值，则数据集中存在要查找的数；如果目标小于中值，则在前半部分找；如果目标大于中值，则在后半部分找。

如何在剩下的一千二百五十万条数据中查找呢？

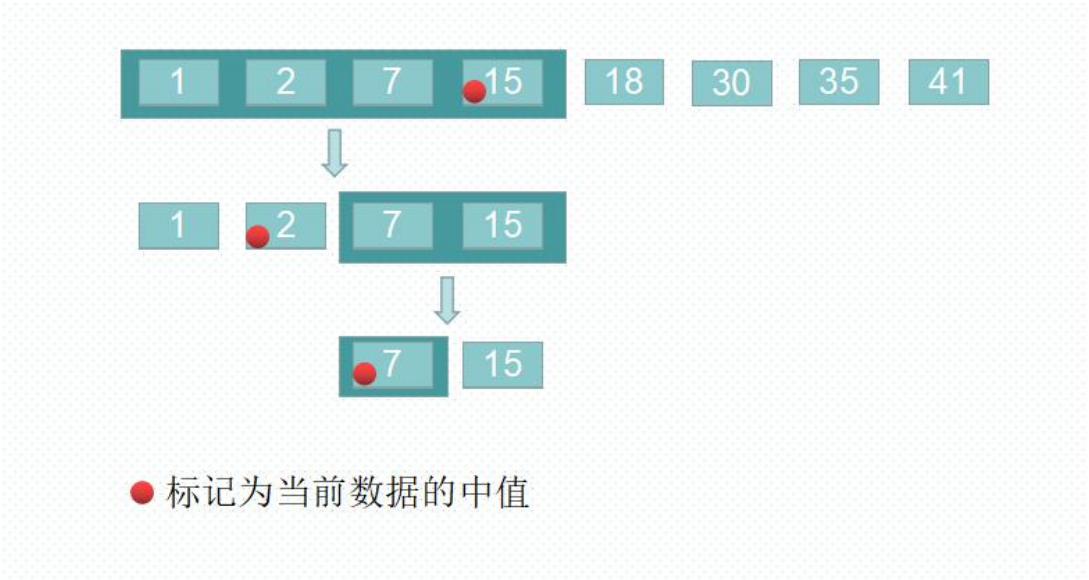
以此类推，直到找到目标为止。

二分查找大大降低了比较次数，二分查找的时间复杂度为：$O(log_2 n)$，即$logn$。

用二分查找，至多查询 27 次，就可确定一亿个有序的数中是否包含某个数。

具体案例

假设我们要在 1，2，7，15，18，30，35，41 中查找 7，上图所示，则查找步骤为：



第一次查询:在 1，2，7，15，18，30，35，41 中查找 7。

首先找到中值：中值为 15（下标= (0+7)/2），将 7 与 15 进行比较，发现 7 比 15 小，继续在前面部分[1，2，7，15]找；

第二次查询: 在[1，2，7，15]中查找 7。

首先找到中值：中值为 2（下标= (0+3)/2），将 7 与 2 进行比较，发现 7 比 2 大，继续在后面部分[7，15]找；

第二次查询: 在[7，15]中查找 7。

首先找到中值：中值为 7（下标= (0+1)/2），将 7 与 7 进行比较，发现相等，代表我们已经查找到 7 了。查询结束。

假设，我们有一个问题，如何在一亿个无序的数据集中查找一个特定的数存不存在？

因为是无序、毫无规则，最容易想到，顺序的拿数据集中的每一个数字与待查询的数字比较。虽然可以解决这个问题，但是这最差的情况下可能需要比较一亿次。

换一种思路:如果我把无序的数据集整理成有序的数据集，则最多 27 次就可以查询到结果了。

新的做法:我们先用无序的数字集生成一份有序的数据集[这份有序的数据集，有个专业术语叫做索引]。然后我们在索引上寻找待查询的数字。

【补充一下:专业术语】

索引作为名词时: 索引是一种拥有特定数据结构【上面的例子，指的是”有序的”这种结构】的数据。

”有序的”这种结构是【索引类型】的一种。另外，我们熟知的索引类型还有 B Tree 索引、红黑树索引、B+ Tree 索引等

把无序的数字集整理成有序的数据集的动作被称为[创建索引、生成索引]。

索引还有一种意思，作为动词:代表用...生成索引[名词]的意思。

把无序的数字集索引一下。【含义，用[无序的数字集]生成了一份索引[名词]】

索引文档。【含义，用[文档]生成了一份索引[名词]】

由于无序的数据查找数字效率太低，我们创建了一份可以进行快速查找数据的有着特定数据结构的数据【索引】，在进行数字查找的时候，我们是在索引上进行查找的，所以查找速度提升了成千上万倍。

同时我们可以归纳出索引的使用方式:

①创建索引

②利用索引进行检索

索引的优缺点:

相比直接在原始数据做查询，在索引上做查询速度更快。如果我们需要实时查询，直接在原始数据上操作已经不能够满足实时要求，则需要创建适合的索引。

而且创建索引是一次操作，而无论多少次查询都可以使用同一份索引。

虽然建立这个索引的过程比较耗资源，但是如果我的查询次数比较多，这个前置的消耗可以被认为是值得的。

另外原始数据有改变【增加或是减少】，也需要维护相对应的索引保持一致，所以维护索引的成本也比较高。

假设，我们有一个问题，如何在一亿篇文章中查找包含'lucene'的文章？

顺序扫描

一种传统的做法叫做顺序扫描法：所谓顺序扫描，比如要找【内容包含某一个字符串】的文件，就是一个文档一个文档的看，对于每一个文档，从头看到尾，如果此文档包含此字符串，则此文档为我们要找的文件，接着看下一个文件，直到扫描完所有的文件，最终找出所有【内容包含某一个字符串】的文件。

对于小批量的文件，这种方法还是很直接，很方便的。但是对于大批量的文件，这种方法就很慢了。

使用索引

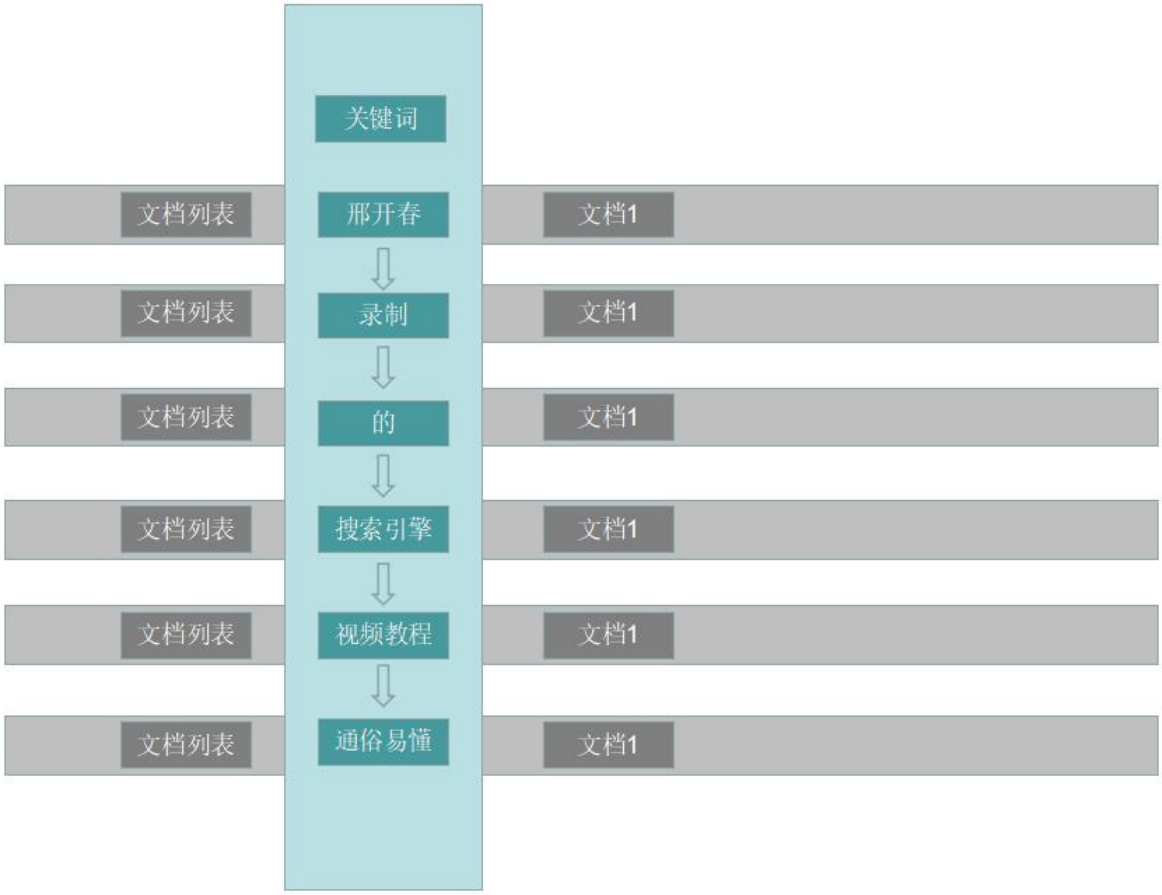
我们能不能仿照【查找无序数字集中是否存在某个数】时的思维，先创建一种【拥有方便查找"关键词"被哪些文章包含的数据结构的数据】索引，然后利用索引进行搜索，从而加快搜索速度。

我们的目标是创造一种适合查找"关键词"被哪些文章包含的数据结构。那好，我就直接创建一种**关键词到文件 ID 列表的映射**的索引，这样给出关键词，我就能随即找到文件 ID 列表，不就完美解决问题了吗?

具体案例

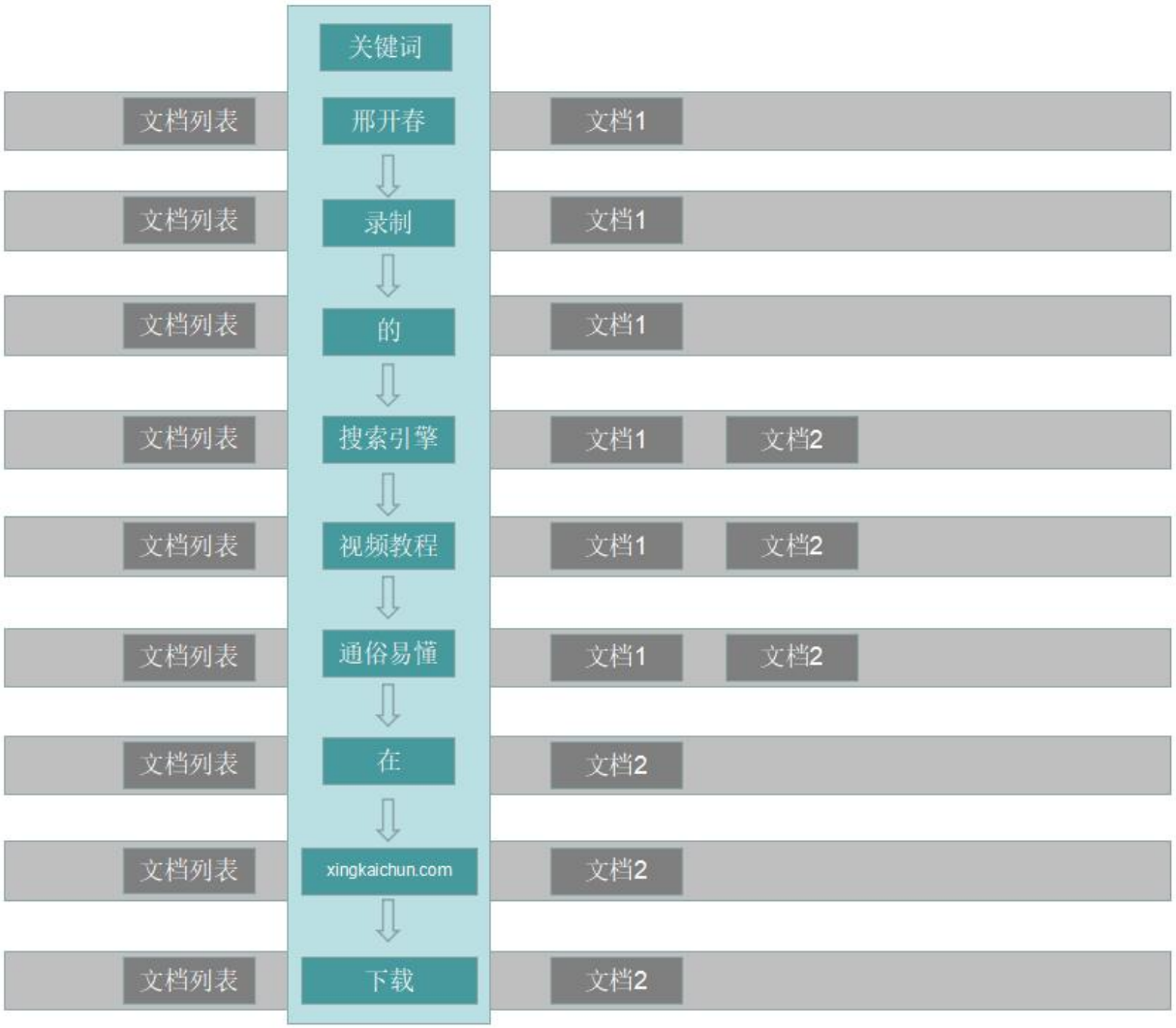
文档 1:邢开春录制的搜索引擎视频教程通俗易懂

文档 1 可以划分出关键词(/邢开春/录制/的/搜索引擎/视频教程/通俗易懂/)



文档 2:搜索引擎视频教程在 www.xingkaichun.com 下载

文档 2 可以划分出关键词(/搜索引擎/视频教程/在/www.xingkaichun.com/下载/)



假设查找’视频教程’出现在哪些文档中。

关键字’视频教程’的那一行的文档列表就是’视频教程’出现的所有文档。

可以发现在这种关键词到文档列表的结构基础上，轻而易举就可以查到某个关键词被哪些文档包含。

以上所说的【关键词到文件列表 ID 映射】是一种特别的索引类型，专业术语叫做倒排索引【倒着排列的索引】或全文索引。

Lucene 之所以搜索快速最重要的原因就是使用了倒排索引。除此之外，众所周知，CPU 速度远远大于磁盘 IO 速度，lucene 里大量使用压缩技术(磁盘压缩，内存压缩)，减少磁盘 IO 开销，充分利用 CPU 的性能以提高程序性能。【视频里讲解作者的理解】

这里为了充分理解倒排索引，将补充一些概念，既然有倒排索引？试想，是否存在与之相对应的正排索引？

全文索引，又称倒排索引、反向索引、inverted index，与之相对应的是正排索引、正向索引、forward index。

正排索引是什么：无论课本，杂志，还是报纸，它们都拥有一个目录。假如我们想看某篇文章，通过目录，我们可以看到这篇文章所在的页面，而不是笨拙的一页页的去看是否是这篇文章。这里，目录就是一个索引【思考：它是什么内容的索引？这样结构的索引为了什么？】。

其实，书籍可以看做两部分组成，一部分是目录索引，另一部分是去掉目录、剩余的实际的内容。

目录有文章标题到文章位置(页码)映射的特殊数据结构，可以帮助我们去了解书本中所有文章，如果喜欢文章，通过文章关联的页码直接定位文章所在书本的位置。

书籍的目录即是相对书籍中的文章的一份索引信息。像目录这种索引，因为通过文章名称(文章名称相当于文章的唯一标识)去找文章，是一个很自然的操作，所以目录这种索引，被称为正向索引[正方向的索引]、正排索引[正方向排列的索引]。

倒排索引是什么：与正排索引相对立，如果我想通过文章中的一部分内容去找这篇文章（例如："邢开春"这个姓名出现在哪些文章中？），就是一个反向操作了[由部分内容找整体]，不那么自然了，而且查找速度会特别慢（一本几百页的书籍，如果不做任何处理的话，想找到这个网址在哪里,你需要一行行去查找匹配，查找效率自然很慢）。好在我们可以针对书籍先做处理，找出所有出现"邢开春"的文章，建立一个从"邢开春"到文章的映射，通过映射，还是可以快速检索"邢开春"在哪篇文章之中出现，这种不太自然的映射结构，被称为倒排索引[倒着排列的索引]、全文索引、反向索引[反方向的索引]。

倒排索引应该如何使用？

上面我们归纳了索引的使用方式:

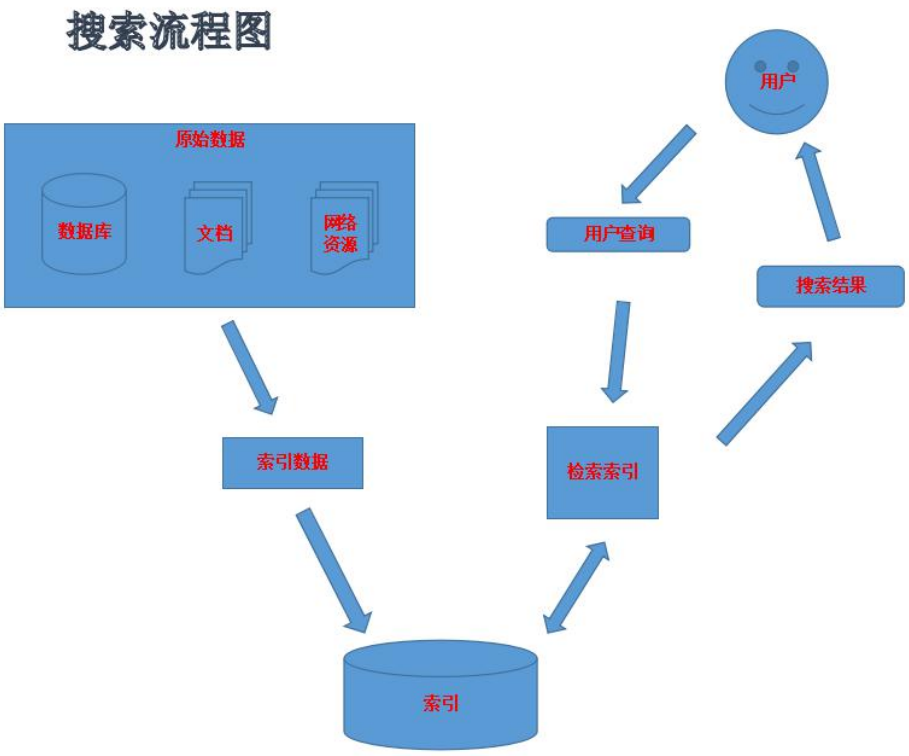
①创建索引

②利用索引进行检索

同理，倒排索引是索引的一种类型，当然它的使用方式也应当是

①建立索引:为原文档创建索引。

②检索：利用索引进行检索。



[搜索引擎相关概念]

文档 1:邢开春录制的搜索引擎视频教程通俗易懂

文档 1 可以划分出关键词(/邢开春/录制/的/搜索引擎/视频教程/通俗易懂/)

同学们有没有注意到这里我把文章 1 切分了很多关键词，这个动作的专业术语叫做分词[把一句话划分为一个个独立的有意义的词汇单元]。负责分词的类在 lucene 源码里被称为 Analyzer[分词器]。而这些个独立的有意义的词汇单元在 lucene 源码里被称为 Term。以后我们用 Term 代称被分词后的一个个词汇单元。

分词是搜索引擎里特别重要的技术。分词出来的词语是倒排索引里的词汇单元。

Term 匹配时，必须完全相等。

而搜索就是利用的倒排索引。所以如果 Term 没有被分出来，就会搜索不到。

用文档 1 的倒排索引(/邢开春/录制/的/搜索引擎/视频教程/通俗易懂/)搜索'教程'这两个字就搜索不到结果，因为倒排索引里没有'教程'这个 Term。

注意 Term 匹配的时候，字符串完全相等才能算匹配的上。特别注意大小写、空格等匹配，这些也需要完全相等，不然匹配不上。

停用词

进一步说，在中文文章里，几乎每一篇文章都会出现'的'、'了'等 Term，思考一下，我们真的需要为这些 Term 创建倒排表吗？'的'、'了'，这些 term 本身也没什么特殊含义，另外每一篇文章都会出现这些 Term，也就相当于用'的'、'了'搜索时，几乎每一篇文章都会匹配，既然所有文章都被匹配了，还搜它干什么。我们希望搜索获取有用信息，而不是大量信息。分词器在分词的时候可不是单单的划

分 Term，它还可以对分词出来的 Term 进一步处理，如过滤掉一部分词语。这里希望被分词器过滤的 Term 被称为停用词。

还有一些违法犯罪的 Term[词汇有罪哈，例如一些历史事件、一些不可描述的词汇可能出现网站就被河蟹掉了]，我们希望屏蔽搜索这些词，一种方法就是不为这些 Term 创建倒排表。分词器在分词的时候可不是单单的划分 Term，它还可以对分词出来的 Term 进一步处理，如过滤掉一部分词语。

文档 1:邢开春录制的搜索引擎视频教程通俗易懂

‘的’是一个停用词，分词器经过分词分析处理后的分词结果如下。

文档 1 可以划分出关键词(/邢开春/录制/搜索引擎/视频教程/通俗易懂/)

同义词

还有同义词，只要在倒排索引里创建同义词的倒排表。搜索的时候，就可以搜索到同义词。[这只是做同义词功能的一种方式]。分词器在分词的时候可不是单单的划分 Term，它还可以对分词出来的 Term 进一步处理，如新增一部分 Term。

文档 1:邢开春录制的搜索引擎视频教程通俗易懂

简单明了时通俗易懂的近义词，分词器经过分词分析处理后的分词结果如下。

文档 1 可以划分出关键词(/邢开春/录制/搜索引擎/视频教程/通俗易懂/简单明了/)

简繁搜索

/邢开春/邢開春/录制/錄製/搜索引擎/搜尋引擎/视频教程/視頻教程/通俗易懂/通俗易懂/

拼音搜索

/邢开春/xingkaichun/录制/luzhi/搜索引擎/sousuoyinqing/视频教程/shipinjiaocheng/通俗易懂/tongsuyidong/

.....

以上可以看到分词器十分重要，众多搜索功能的实现都与它息息相关。做搜索引擎的同学，最好要理解分词器，还需要看分词组件的源码，才好彻底掌握，才好对分词器的使用得心应手。视频会对常用分词方式、分词质量的评估、当前分词的困难点、分词组件源码进行讲解，让读者对分词有一个全面的认识。

第二个问题: lucene 的搜索效果为什么好？

用户搜索的关键词可以看做一篇文章 A，搜索匹配出来文章 B、C、D。

计算 A 与 B、A 与 C、A 与 D 文档的相似度。相似度高的优先返回，就能有很好的效果。

困难点在于如何计算两篇文章的相似度。Lucene 提供了空间向量模型、BM25 算法可以用来评估文章的相似度。我录的视频里包含了算法的推导、源码解读、多年搜索引擎经验评分总结。

这里简单说一下两篇文章的相似度与哪些因素相关。

lucene 快速使用

lucene Maven 地址

创建索引

利用索引进行搜索

lucene 搜索为什么快？

lucene 调优？

lucene 索引管理？

lucene 打分算法

lucene 源码解读