

06 | 如何用Elasticsearch构建商品搜索系统?

2020-03-10 李玥

后端存储实战课

[进入课程 >](#)



讲述：李玥

时长 13:28 大小 12.34M



你好，我是李玥。

搜索这个特性可以说是无处不在，现在很少有网站或者系统不提供搜索功能了，所以，即使你不是一个专业做搜索的程序员，也难免会遇到一些搜索相关的需求。搜索这个东西，表面上看功能很简单，就是一个搜索框，输入关键字，然后搜出来想要的内容就好了。

搜索背后的实现，可以非常简单，简单到什么程度呢？我们就用一个 SQL，LIKE 一下就能实现；也可以很复杂，复杂到什么程度呢？不说百度谷歌这种专业做搜索的公司，其业务做搜索的互联网大厂，搜索团队大多是千人规模，这里面不仅有程序员，还有算法工程师、业务专家等等。二者的区别也仅仅是，搜索速度的快慢以及搜出来的内容好坏而已。

今天这节课，我们就以电商中的商品搜索作为例子，来讲一下，如何用 ES(Elasticsearch) 来快速、低成本地构建一个体验还不错的搜索系统。

理解倒排索引机制

刚刚我们说了，既然我们的数据大多都是存在数据库里，用 SQL 的 LIKE 也能实现匹配，也能搜出结果，为什么还要专门做一套搜索系统呢？我先来和你分析一下，为什么数据库不适合做搜索。

搜索的核心需求是全文匹配，对于全文匹配，数据库的索引是根本派不上用场的，那只能全表扫描。全表扫描已经非常慢了，这还不算，还需要在每条记录上做全文匹配，也就是一个字一个字的比对，这个速度就更慢了。所以，使用数据库来做搜索，性能上完全没法满足要求。

那 ES 是怎么来解决搜索问题的呢？我们来举个例子说明一下，假设我们有这样两个商品，一个是烟台红富士苹果，一个是苹果手机 iPhone XS Max。

DOCID	SKUID	标题
666	100002860826	烟台红富士苹果 5kg 一级铂金大果 单果230g以上 新鲜水果
888	100000177760	苹果 Apple iPhone XS Max (A2104) 256GB 金色 移动联通电信4G手机 双卡双待

这个表里面的 DOCID 就是唯一标识一条记录的 ID，和数据库里面的主键是类似的。

为了能够支持快速地全文搜索，ES 中对于文本采用了一种特殊的索引：倒排索引 (Inverted Index) 。那我们看一下在 ES 中，这两条商品数据倒排索引长什么样？请看下面这个表。

TERM	DOCID
烟台	666
红富士	666
苹果	666,888
5kg	666
一级	666
铂金	666
大果	666
Apple	888
iphone	888
XS	888
Max	888
手机	888
...	...

可以看到，这个倒排索引的表，它是以单词作为索引的 Key，然后每个单词的倒排索引的值是一个列表，这个列表的元素就是含有这个单词的商品记录的 DOCID。

这个倒排索引怎么构建的呢？当我们往 ES 写入商品记录的时候，ES 会先对需要搜索的字段，也就是商品标题进行**分词**。分词就是把一段连续的文本按照语义拆分成多个单词。然后 ES 按照单词来给商品记录做索引，就形成了上面那个表一样的倒排索引。

当我们搜索关键字“苹果手机”的时候，ES 会对关键字也进行分词，比如说，“苹果手机”被分为“苹果”和“手机”。然后，ES 会在倒排索引中去搜索我们输入的每个关键字分词，搜索结果应该是：

TERM	DOCID
苹果	666,888
手机	888

666 和 888 这两条记录都能匹配上搜索的关键词，但是 888 这个商品比 666 这个商品匹配度更高，因为它两个单词都能匹配上，所以按照匹配度把结果做一个排序，最终返回的搜索结果就是：

苹果Apple iPhone XS Max (A2104) 256GB 金色 移动联通电信 4G手机双卡双待

烟台红富士苹果5kg 一级铂金大果 单果 230g 以上 新鲜水果

看起来搜索的效果还是不错的。

为什么倒排索引可以做到快速搜索？我和你一起来分析一下上面这个例子的查找性能。

这个搜索过程，其实就是对上面的倒排索引做了二次查找，一次找“苹果”，一次找“手机”。**注意，整个搜索过程中，我们没有做过任何文本的模糊匹配。**ES 的存储引擎存储倒排索引时，肯定不是像我们上面表格中展示那样存成一个二维表，实际上它的物理存储结构和 MySQL 的 InnoDB 的索引是差不多的，都是一颗查找树。

对倒排索引做两次查找，也就是对树进行二次查找，它的时间复杂度，类似于 MySQL 中的二次命中索引的查找。显然，这个查找速度，比用 MySQL 全表扫描加上模糊匹配的方式，要快好几个数量级。

如何在 ES 中构建商品的索引？

理解了倒排索引的原理之后，我们一起用 ES 构建一个商品索引，简单实现一个商品搜索系统。虽然 ES 是为搜索而生的，但本质上，它仍然是一个存储系统。ES 里面的一些概念，基本上都可以在关系数据库中找到对应的名词，为了便于你快速理解这些概念，我把这些概念的对应关系列出来，你可以对照理解。

ElasticSearch	RDBMS
INDEX	表
DOCUMENT	行
FIELD	列
MAPPING	表结构


在 ES 里面，数据的逻辑结构类似于 MongoDB，每条数据称为一个 **DOCUMENT**，简称 DOC。DOC 就是一个 JSON 对象，DOC 中的每个 JSON 字段，在 ES 中称为 **FIELD**，把一组具有相同字段的 DOC 存放在一起，存放它们的逻辑容器叫 **INDEX**，这些 DOC 的

JSON 结构称为 **MAPPING**。这里面最不好理解的就是这个 INDEX，它实际上类似于 MySQL 中表的概念，而不是我们通常理解的用于查找数据的索引。

ES 是一个用 Java 开发的服务端程序，除了 Java 以外就没有什么外部依赖了，安装部署都非常简单，具体你可以参照它的 [官方文档](#) 先把 ES 安装好。我们这个示例中，使用的 ES 版本是目前的最新版本 7.6。


另外，为了能让 ES 支持中文分词，需要给 ES 安装一个中文的分词插件 [IK Analysis for Elasticsearch](#)，这个插件的作用就是告诉 ES 怎么对中文文本进行分词。

你可以直接执行下面的命令自动下载并安装：

 复制代码

```
1 $elasticsearch-plugin install https://github.com/medcl/elasticsearch-analysis-
```

安装完成后，需要重启 ES，验证一下是否安装成功：

 复制代码

```
1 curl -X POST "localhost:9200/_analyze?pretty" -H 'Content-Type: application/js
2 {
3   "tokens" : [
4     {
5       "token" : "极",
6       "start_offset" : 0,
7       "end_offset" : 1,
8       "type" : "CN_CHAR",
9       "position" : 0
10    },
11    {
12      "token" : "客",
13      "start_offset" : 1,
14      "end_offset" : 2,
15      "type" : "CN_CHAR",
16      "position" : 1
17    },
18    {
19      "token" : "时间",
20      "start_offset" : 2,
21      "end_offset" : 4,
22      "type" : "CN_WORD",
23      "position" : 2
24    }
25  ]
26 }
```

```
25   ]
26 }
```


可以看到，这个分词器把“极客时间”分成了“极”、“客”和“时间”，没认出来“极客”这个词，还是有改进空间的。

为了能够实现商品搜索，我们需要先把商品信息存放到 ES 中，首先我们先定义存放在 ES 中商品的数据结构，也就是 MAPPING。

Field	Datatype	说明
sku_id	long	商品ID
title	text	商品标题

我们这个 MAPPING 只要两个字段就够了，sku_id 就是商品 ID，title 保存商品的标题，当用户在搜索商品的时候，我们在 ES 中来匹配商品标题，返回符合条件商品的 sku_id 列表。ES 默认提供了标准的 RESTful 接口，不需要客户端，直接使用 HTTP 协议就可以访问，这里我们使用 [curl](#) 通过命令行来操作 ES。

接下来我们使用上面这个 MAPPING 创建 INDEX，类似于 MySQL 中创建一个表。

 复制代码

```
1 curl -X PUT "localhost:9200/sku" -H 'Content-Type: application/json' -d '{
2     "mappings": {
3         "properties": {
4             "sku_id": {
5                 "type": "long"
6             },
7             "title": {
8                 "type": "text",
9                 "analyzer": "ik_max_word",
10                "search_analyzer": "ik_max_word"
11            }
12        }
13    }
14 }'
15 {"acknowledged":true,"shards_acknowledged":true,"index":"sku"}
```

这里面，使用 PUT 方法创建一个 INDEX，INDEX 的名称是 “sku”，直接写在请求的 URL 中。请求的 BODY 是一个 JSON 对象，内容就是我们上面定义的 MAPPING，也就是数据结构。这里面需要注意一下，由于我们要在 title 这个字段上进行全文搜索，所以我们把数据类型定义为 text，并指定使用我们刚刚安装的中文分词插件 IK 作为这个字段的分词器。

创建好 INDEX 之后，就可以往 INDEX 中写入商品数据，插入数据需要使用 HTTP POST 方法：

 复制代码

```
1 curl -X POST "localhost:9200/sku/_doc/" -H 'Content-Type: application/json' -d
2     "sku_id": 100002860826,
3     "title": "烟台红富士苹果 5kg 一级铂金大果 单果230g以上 新鲜水果"
4 }'
5 {"_index":"sku","_type":"_doc","_id":"yxQVSHABiy2kuAJG8ilW","_version":1,"resu`
6
7 curl -X POST "localhost:9200/sku/_doc/" -H 'Content-Type: application/json' -d
8     "sku_id": 100000177760,
9     "title": "苹果 Apple iPhone XS Max (A2104) 256GB 金色 移动联通电信4G手机 双
10 }'
11 {"_index":"sku","_type":"_doc","_id":"zBQWSHABiy2kuAJGgim1","_version":1,"resu`
```

这里面我们插入了两条商品数据，一个烟台红富士，一个 iPhone 手机。然后就可以直接进行商品搜索了，搜索使用 HTTP GET 方法。

 复制代码

```
1 curl -X GET 'localhost:9200/sku/_search?pretty' -H 'Content-Type: application/;
2     "query" : { "match" : { "title" : "苹果手机" }}
3 }'
4 {
5     "took" : 23,
6     "timed_out" : false,
7     "_shards" : {
8         "total" : 1,
9         "successful" : 1,
10        "skipped" : 0,
11        "failed" : 0
12    },
13    "hits" : {
14        "total" : {
15            "value" : 2,
16            "relation" : "eq"
17        },
```

```

18     "max_score" : 0.8594865,
19     "hits" : [
20         {
21             "_index" : "sku",
22             "_type" : "_doc",
23             "_id" : "zBQWSHABiy2kuAJGgim1",
24             "_score" : 0.8594865,
25             "_source" : {
26                 "sku_id" : 100000177760,
27                 "title" : "苹果 Apple iPhone XS Max (A2104) 256GB 金色 移动联通电信4G手机"
28             }
29         },
30         {
31             "_index" : "sku",
32             "_type" : "_doc",
33             "_id" : "yxQVSHABiy2kuAJG8ilW",
34             "_score" : 0.18577608,
35             "_source" : {
36                 "sku_id" : 100002860826,
37                 "title" : "烟台红富士苹果 5kg 一级铂金大果 单果230g以上 新鲜水果"
38             }
39         }
40     ]
41 }
42 }

```

我们先看一下请求中的 URL，其中的 “sku” 代表要在 sku 这个 INDEX 内进行查找，“_search” 是一个关键字，表示要进行搜索，参数 pretty 表示格式化返回的 JSON，这样方便阅读。再看一下请求 BODY 的 JSON，query 中的 match 表示要进行全文匹配，匹配的字段就是 title，关键字是 “苹果手机”。

可以看到，在返回结果中，匹配到了 2 条商品记录，和我们在前面讲解倒排索引时，预期返回的结果是一致的。

我们来回顾一下使用 ES 构建商品搜索服务的这个过程：首先安装 ES 并启动服务，然后创建一个 INDEX，定义 MAPPING，写入数据后，执行查询并返回查询结果，其实，这个过程和我们使用数据库时，先建表、插入数据然后查询的过程，就是一样的。所以，你就把 ES 当做一个支持全文搜索的数据库来使用就行了。

小结

ES 本质上是一个支持全文搜索的分布式内存数据库，特别适合用于构建搜索系统。ES 之所以能有非常好的全文搜索性能，最重要的原因就是采用了倒排索引。倒排索引是一种特别为搜索而设计的索引结构，倒排索引先对需要索引的字段进行分词，然后以分词为索引组成一个查找树，这样就把一个全文匹配的查找转换成了对树的查找，这是倒排索引能够快速进行搜索的根本原因。

但是，倒排索引相比于一般数据库采用的 B 树索引，它的写入和更新性能都比较差，因此倒排索引也只是适合全文搜索，不适合更新频繁的交易类数据。

思考题

我们在电商的搜索框中搜索商品时，它都有一个搜索提示的功能，比如我输入“苹果”还没有点击搜索按钮的时候，搜索框下面会提示“苹果手机”、“苹果 11、苹果电脑”这些建议的搜索关键字，请你课后看一下 ES 的文档，想一下，如何用 ES 快速地实现这个搜索提示功能？

欢迎你在留言区与我讨论，如果你觉得今天的内容对你有帮助，也欢迎把它分享给你的朋友。

后端存储实战课

类电商平台存储技术应用指南

李玥

京东零售计算存储平台部资深架构师



新版升级：点击「👤请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

精选留言 (7)

写留言



李玥 置顶

2020-03-10

Hi, 我是李玥。

还是在这里回顾一下上节课的思考题：

2PC也有一些改进版本，比如3PC、TCC这些，它们大体的思想和2PC是差不多的，解决...
展开



Geek_c76e2d

2020-03-10

因为用户每输入一个字都可能会发请求查询搜索框中的搜索推荐。所以搜索推荐的请求量远高于搜索框中的搜索。es针对这种情况提供了suggestion api，并提供的专门的数据结构应对搜索推荐，性能高于match，但它应用起来也有局限性，就是只能做前缀匹配。再结合pinyin分词器可以做到输入拼音字母就提示中文。如果想做非前缀匹配，可以考虑Ngram。不过Ngram有些复杂，需要开发者自定义分析器。比如有个网址www.geekbang.c...
展开

作者回复: 谢谢支持



8



每天晒白牙

2020-03-10

老师，请教个问题，比如一个商品搜索系统，给一些商品打标签，然后支持根据商品信息和标签搜索商品，有啥方案吗？

展开

作者回复: 这个需求用ES很合适啊



1



每天晒白牙



2020-03-10

应该输入的时候就会实时去 ES 中检索吧

展开 ▾



hello

2020-03-10

老师，能否来一篇加餐，讲讲ES、MySQL、MongoDB、RocketMQ/Kafka、newSQL这些存储的对比，底层是基于什么原理擅长干哪些事，不擅长干哪些事？

展开 ▾



墨雨

2020-03-10

那一般什么时候来更新索引呢？是建立一个定时任务来更新么

作者回复: 写入数据的时候。



ELLIOT

2020-03-10

使用match query对商品名进行匹配，然后按score进行sort排序，选择前n项

