

文档: 1、ElasticSearch快速入门与集群高可用...

链接: [http://note.youdao.com/noteshare?](http://note.youdao.com/noteshare?id=c9f03451f751beb5f913bbfbd9818ed1&sub=0A07974608094B04B16632B407F4AEC0)

[id=c9f03451f751beb5f913bbfbd9818ed1&sub=0A07974608094B04B16632B407F4AEC0](http://note.youdao.com/noteshare?id=c9f03451f751beb5f913bbfbd9818ed1&sub=0A07974608094B04B16632B407F4AEC0)

# 一. ElasticSearch简介

## 1、ElasticSearch（简称ES）

Elasticsearch是用Java开发并且是当前最流行的开源的企业级搜索引擎。

能够达到近实时搜索，稳定，可靠，快速，安装使用方便。

客户端支持Java、.NET（C#）、PHP、Python、Ruby等多种语言。

官方网站: <https://www.elastic.co/>

下载地址: <https://www.elastic.co/cn/start>

创始人: Shay Banon（谢巴农）



## 2、应用场景

## 长沙有哪些好玩的地方推荐? - 知乎



2020年12月22日 空中悬浮体验 ★★★☆☆3.西湖公园散步,58小镇(我家旁边) ★★★☆☆4.大学城各种小吃,爬岳麓山观景,体力不行的可以坐缆车上山,玩滑道下山,鸟语林、欣赏湖...

知乎 百度快照

## 长沙最好玩的几个地方

名称: 长沙青天寨 服务特色: 项目丰富 游玩类型: 农家乐 现价: 90元起

长沙好玩的地方,2020年本人亲身游玩体验写下此篇旅游攻略,纯玩的经验分享,省心又省钱,多种玩法推荐,不走回头路,各种注意事项,自行游需看,交通便利,环境优美,项目丰富有趣..

长沙青天寨生态农业股份 2021-01 广告 保障

## 长沙有什么好玩的? - 知乎



2016年12月5日 国内出游必玩指南 1,914 人赞同了该回答 长沙并不是一个旅游城市,更多时候长沙被作为革命教育的地方,代表着人民对毛泽东的追思,对长沙人来说长沙有的也是情怀,而...

知乎 百度快照

## 长沙好玩的地方-2021长沙有什么好玩的地方推荐-去哪儿攻略



去哪儿攻略为您推荐长沙好玩的地方、长沙适合年轻人好玩的地方,2021长沙有什么好玩的地方快来看去哪儿攻略的推荐吧。

travel.qunar.com/p-cs300022-ch... 保障 百度快照

## 长沙 有哪些好玩的地方 - 百度知道



18个回答 · 回答时间: 2020年11月11日

最佳答案: 1、岳麓山 橘子洲和岳麓山,是长沙最值得去的两大景点,因为疫情期间岳麓书院没有开,我还小纠结了一会,要不要在这次行...

search.jd.com/Search?keyword=华为手机&enc=utf-8&wq=华为手机&pvid=e593f2f271264a45a859a6f1067e8443

上传

学院 资料 白起老师 百度一下,你就知道

|   |  |   |   |  |
|---|--|---|---|--|
| <p>畅享20 Pro 5G手机 12期免息可选 华为直供 现货速发 全国联保/店铺延保1年</p> <p>¥2599.00</p> <p>华为畅享20 Pro 5G手机【现货速发 12期免息可选】下单尊享豪礼 星河银 全1万+条评价</p> <p>久汇通手机旗舰店</p> | <p>畅享20 SE手机 6.67英寸全高清大屏 5000mAh大电池 幻夜黑 (8GB+128GB)</p> <p>¥1899.00</p> <p>1000+条评价</p> <p>聚捷联盟手机旗舰店</p> | <p>荣耀play4T 全网通手机 幻夜黑 (6G+128G) 标准版【现货速发,内含】</p> <p>¥1499.00</p> <p>3000+条评价</p> <p>聚捷联盟手机旗舰店</p> | <p>华为nova5i 手机 影音蓝 (6GB+128GB)【现货速发】京仓发货, 2400W后置AI四</p> <p>¥2099.00</p> <p>3万+条评价</p> <p>聚捷联盟手机旗舰店</p> | <p>荣耀Play3e 智能老人老年备用手机 3020mAh长续航1300万大光圈相机 幻夜</p> <p>¥958.00</p> <p>1万+条评价</p> <p>伟德手机专营店</p> |
|---|--|---|---|--|

## 3、ElasticSearch与Lucene的关系

Lucene可以被认为迄今为止最先进、性能最好的、功能最全的搜索引擎库（框架）

但是想要使用Lucene，必须使用Java来作为开发语言并将其直接集成到你的应用中，并且Lucene的配置及使用非常复杂，你需要深入了解检索的相关知识来理解它是如何工作的。

Lucene缺点：

- 1) 只能在Java项目中使用, 并且要以jar包的方式直接集成项目中.
- 2) 使用非常复杂-创建索引和搜索索引代码繁杂
- 3) 不支持集群环境-索引数据不同步（不支持大型项目）
- 4) 索引数据如果太多就不行，索引库和应用所在同一个服务器, 共同占用硬盘. 共用空间少.

上述Lucene框架中的缺点, ES全部都能解决.

## 4、哪些公司在使用Elasticsearch

- 1 1. 京东
- 2 2. 携程
- 3 3. 去哪儿
- 4 4. 58同城
- 5 5. 滴滴
- 6 6. 今日头条
- 7 7. 小米
- 8 8. 哔哩哔哩
- 9 9. 联想
- 10 10. GitHub
- 11 11. 微软
- 12 12. Facebook
- 13 等等...

在国内，阿里巴巴、腾讯、滴滴、今日头条、饿了么、360安全、小米，vivo 等诸多知名公司都在使用Elasticsearch，多到不能穷举，主要用于日志搜集分析、用于APP综合搜索、订单系统搜索、企业级网站搜索等方面。

如果你还在使用Mysql进行海量数据检索,动不动遇到性能瓶颈以及极差的搜索体验时，那么你应该考虑使用Elasticsearch了

## 搜索引擎技术排名

1 <https://db-engines.com/en/ranking/search+engine>

☐ include secondary database models

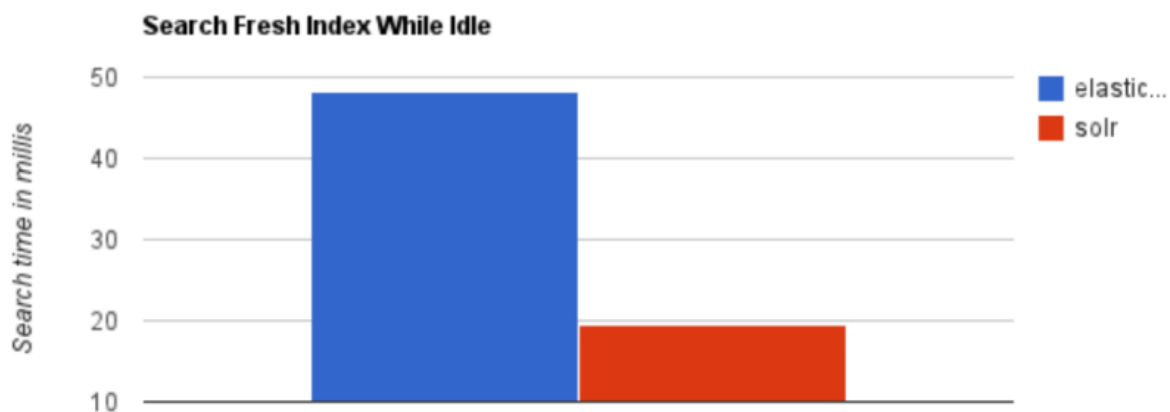
21 systems in ranking, June 2021

| Rank     |          |          | DBMS                      | Database Model             | Score    |          |          |
|----------|----------|----------|---------------------------|----------------------------|----------|----------|----------|
| Jun 2021 | May 2021 | Jun 2020 |                           |                            | Jun 2021 | May 2021 | Jun 2020 |
| 1.       | 1.       | 1.       | Elasticsearch             | Search engine, Multi-model | 154.71   | -0.65    | +5.02    |
| 2.       | 2.       | 2.       | Splunk                    | Search engine              | 90.27    | -1.84    | +2.19    |
| 3.       | 3.       | 3.       | Solr                      | Search engine, Multi-model | 52.10    | +0.91    | +0.84    |
| 4.       | 4.       | 4.       | MarkLogic                 | Multi-model                | 9.38     | -0.15    | -1.87    |
| 5.       | 6.       | 5.       | Sphinx                    | Search engine              | 8.11     | +0.53    | +1.76    |
| 6.       | 5.       | 8.       | Algolia                   | Search engine              | 7.66     | -0.05    | +2.88    |
| 7.       | 7.       | 6.       | Microsoft Azure Search    | Search engine              | 6.55     | +0.50    | +0.25    |
| 8.       | 8.       | 7.       | ArangoDB                  | Multi-model                | 4.92     | +0.53    | -0.47    |
| 9.       | 9.       | 10.      | Virtuoso                  | Multi-model                | 3.69     | +0.25    | +1.41    |
| 10.      | 10.      | 9.       | Amazon CloudSearch        | Search engine              | 2.14     | -0.05    | -0.39    |
| 11.      | 11.      | 11.      | Xapian                    | Search engine              | 0.89     | +0.01    | +0.16    |
| 12.      | 12.      | 12.      | CrateDB                   | Multi-model                | 0.80     | +0.03    | +0.07    |
| 13.      | 13.      | 14.      | Alibaba Cloud Log Service | Search engine              | 0.47     | +0.04    | +0.19    |
| 14.      | 14.      | 13.      | SearchBlox                | Search engine              | 0.38     | +0.01    | +0.08    |
| 15.      | 15.      | 16.      | Weaviate                  | Search engine              | 0.14     | +0.08    | +0.08    |
| 16.      | 16.      | 18.      | Exorbyte                  | Search engine              | 0.06     | +0.01    | +0.03    |
| 17.      | 17.      | 15.      | Manticore Search          | Search engine              | 0.03     | -0.01    | -0.06    |
| 18.      | 18.      | 19.      | FinchDB                   | Multi-model                | 0.03     | 0.00     | +0.00    |

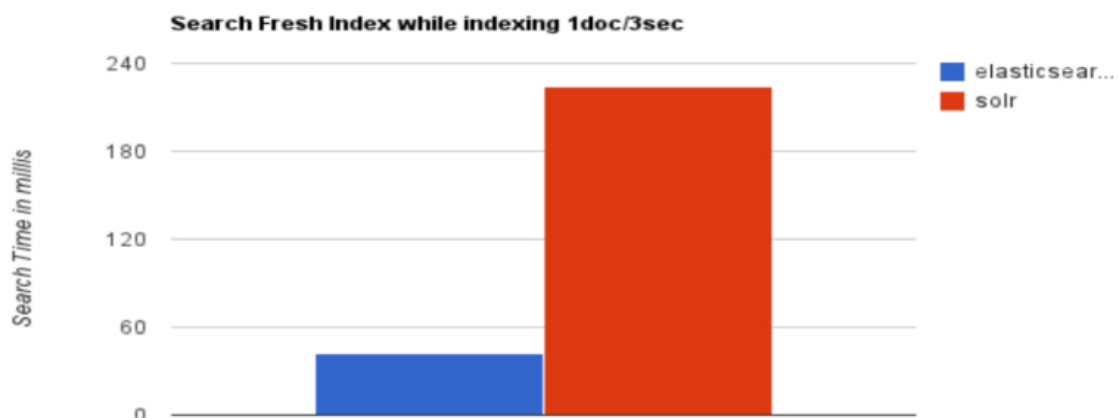
## 5、ES vs Solr比较

### 5.1、ES vs Solr 检索速度

当单纯的对已有数据进行搜索时，Solr更快。



当实时建立索引时，Solr会产生io阻塞，查询性能较差，Elasticsearch具有明显的优势。



大型互联网公司，实际生产环境测试，将搜索引擎从Solr转到 Elasticsearch以后的平均查询速度有了50倍的提升。



## 总结：

二者安装都很简单。

- 1、Solr 利用 Zookeeper 进行分布式管理，而Elasticsearch 自身带有分布式协调管理功能。
- 2、Solr 支持更多格式的数据，比如JSON、XML、CSV，而 Elasticsearch 仅支持 json文件格式。
- 3、Solr 在传统的搜索应用中表现好于 Elasticsearch，但在处理实时搜索应用时效率明显低于 Elasticsearch。
- 4、Solr 是传统搜索应用的有力解决方案，但 Elasticsearch更适用于新兴的实时搜索应用。

## 5.2、ES vs 关系型数据库

|               |                   |              |                  |               |
|---------------|-------------------|--------------|------------------|---------------|
| 关系型数据库        | Database<br>(数据库) | Table<br>(表) | Row<br>(行)       | Column<br>(列) |
| ElasticSearch | Index<br>(索引库)    | Type<br>(类型) | Document<br>(文档) | Field<br>(字段) |

## 二、Lucene全文检索框架

### 1、什么是全文检索

全文检索是指：

- 通过一个程序扫描文本中的每一个单词，针对单词建立索引，并保存该单词在文本中的位置、以及出现的次数
- 用户查询时，通过之前建立好的索引来查询，将索引中单词对应的文本位置、出现的次数返回给用户，因为有了具体文本的位置，所以就可以将具体内容读取出来了

### 2、分词原理之倒排索引



倒排索引总结：

索引就类似于目录，平时我们使用的都是索引，都是通过主键定位到某条数据，那么倒排索引呢，刚好相反，数据对应到主键。这里以一个博客文章的内容为例：

1. 索引

| 文章ID | 文章标题        | 文章内容                          |
|------|-------------|-------------------------------|
| 1    | 浅析JAVA设计模式  | JAVA设计模式是每一个JAVA程序员都应该掌握的进阶知识 |
| 2    | JAVA多线程设计模式 | JAVA多线程与设计模式结合                |

2. 倒排索引

假如，我们有一个站内搜索的功能，通过某个关键词来搜索相关的文章，那么这个关键词可能出现在标题中，也可能出现在文章内容中，那我们将会在创建或修改文章的时候，建立一个关键词与文章的对应关系表，这种，我们可以称之为倒排索引，因此倒排索引，也可称之为反向索引。如：

| 关键词  | 文章ID |
|------|------|
| JAVA | 1    |
| 设计模式 | 1, 2 |
| 多线程  | 2    |

注：这里涉及中文分词的问题

三、Elasticsearch中的核心概念

1、索引 index

一个索引就是一个拥有几分相似特征的文档的集合。比如说，可以有一个客户数据的索引，另一个产品目录的索引，还有一个订单数据的索引

一个索引由一个名字来标识（必须全部是小写字母的），并且当我们要对对应于这个索引中的文档进行索引、搜索、更新和删除的时候，都要使用到这个名字

2、映射 mapping

ElasticSearch中的映射（Mapping）用来定义一个文档

mapping是处理数据的方式和规则方面做一些限制，如某个字段的数据类型、默认值、分词器、是否被索引等等，这些都是映射里面可以设置的

### 3、字段Field

相当于是数据表的字段|列

### 4、字段类型 Type

每一个字段都应该有一个对应的类型，例如：Text、Keyword、Byte等

### 5、文档 document

一个文档是一个可被索引的基础信息单元，类似一条记录。文档以JSON（Javascript Object Notation）格式来表示；

### 6、集群 cluster

一个集群就是由一个或多个节点组织在一起，它们共同持有整个的数据，并一起提供索引和搜索功能

### 7、节点 node

一个节点是集群中的一个服务器，作为集群的一部分，它存储数据，参与集群的索引和搜索功能

一个节点可以通过配置集群名称的方式来加入一个指定的集群。默认情况下，每个节点都会被安排加入到一个叫做“elasticsearch”的集群中

这意味着，如果在网络中启动了若干个节点，并假定它们能够相互发现彼此，它们将会自动地形成并加入到一个叫做“elasticsearch”的集群中

在一个集群里，可以拥有任意多个节点。而且，如果当前网络中没有运行任何Elasticsearch节点，这时启动一个节点，会默认创建并加入一个叫做“elasticsearch”的集群。



## 8、分片和副本 shards&replicas

### 8.1、分片

- 一个索引可以存储超出单个结点硬件限制的大量数据。比如，一个具有10亿文档的索引占据1TB的磁盘空间，而任一节点都没有这样大的磁盘空间；或者单个节点处理搜索请求，响应太慢
- 为了解决这个问题，Elasticsearch提供了将索引划分成多份的能力，这些份就叫做分片
- 当创建一个索引的时候，可以指定你想要的分片的数量
- 每个分片本身也是一个功能完善并且独立的“索引”，这个“索引”可以被放置到集群中的任何节点上
- 分片很重要，主要有两方面的原因
  - 允许水平分割/扩展你的内容容量
  - 允许在分片之上进行分布式的、并行的操作，进而提高性能/吞吐量
- 至于一个分片怎样分布，它的文档怎样聚合回搜索请求，是完全由Elasticsearch管理的，对于作为用户来说，这些都是透明的

### 8.2、副本

- 在一个网络/云的环境里，失败随时都可能发生，在某个分片/节点不知怎么的就处于离线状态，或者由于任何原因消失了，这种情况下，有一个故障转移机制是非常有用并且是强烈推荐的。为此目的，Elasticsearch允许你创建分片的一份或多份拷贝，这些拷贝叫做副本分片，或者直接叫副本
- 副本之所以重要，有两个主要原因
  - 1) 在分片/节点失败的情况下，提供了高可用性。  
注意到复制分片从不与原/主要（original/primary）分片置于同一节点上是非常重要的
  - 2) 扩展搜索量/吞吐量，因为搜索可以在所有的副本上并行运行

每个索引可以被分成多个分片。一个索引有0个或者多个副本  
一旦设置了副本，每个索引就有了主分片和副本分片，分片和副本的数量可以在索引

创建的时候指定

在索引创建之后，可以在任何时候动态地改变副本的数量，但是不能改变分片的数量

## 四、安装Elasticsearch

### 1、安装Elasticsearch

#### 1.1、创建普通用户

**ES不能使用root用户来启动，必须使用普通用户来安装启动。**这里我们创建一个普通用户以及定义一些常规目录用于存放我们的数据文件以及安装包等。

创建一个es专门的用户（**必须**）

# 使用root用户在服务器执行以下命令

```
1 先创建组，再创建用户：
2 1) 创建 elasticsearch 用户组
3 [root@localhost ~]# groupadd elasticsearch
4
5 2) 创建用户 tlbaiqi 并设置密码
6 [root@localhost ~]# useradd tlbaiqi
7 [root@localhost ~]# passwd tlbaiqi
8
9 3) # 创建es文件夹，
10 并修改owner为baiqi用户
11 mkdir -p /usr/local/es
12
13 4) 用户es 添加到 elasticsearch 用户组
14 [root@localhost ~]# usermod -G elasticsearch tlbaiqi
15 [root@localhost ~]# chown -R tlbaiqi /usr/local/es/elasticsearch-7.6.1
16
17 5) 设置sudo权限
18 #为了让普通用户有更大的操作权限，我们一般都会给普通用户设置sudo权限，方便普通用户的操作
19 #三台机器使用root用户执行visudo命令然后为es用户添加权限
20 [root@localhost ~]# visudo
21
22 #在root ALL=(ALL) ALL 一行下面
```

```
23 #添加tlbaiqi用户 如下:
24 tlbaiqi ALL=(ALL) ALL
25
26 #添加成功保存后切换到tlbaiqi用户操作
27
28 [root@localhost ~]# su tlbaiqi
29 [tlbaiqi@localhost root]$
30
```

## 1.2、上传压缩包并解压

将es的安装包下载并上传到服务器的/user/local/es路径下，然后进行解压

使用tlbaiqi用户来执行以下操作，将es安装包上传到指定服务器，并使用es用户执行以下命令解压。

```
1 # 解压Elasticsearch
2 su tlbaiqi
3 cd /user/local/
4 tar -zxvf elasticsearch-7.6.1-linux-x86_64.tar.gz -C /usr/local/es/
```

## 1.3、修改配置文件

### 1.3.1、修改elasticsearch.yml

进入服务器使用**baiqi**用户来修改配置文件

```
1
2 mkdir -p /usr/local/es/elasticsearch-7.6.1/log
3 mkdir -p /usr/local/es/elasticsearch-7.6.1/data
4
5
6 cd /usr/local/es/elasticsearch-7.6.1/config
7 vim elasticsearch.yml
8 cluster.name: my-application
9 node.name: node-1
10 path.data: /usr/local/es/elasticsearch-7.6.1/data
11 path.logs: /usr/local/es/elasticsearch-7.6.1/log
12 network.host: 0.0.0.0
13 http.port: 9200
14 discovery.seed_hosts: ["服务器IP"]
15 cluster.initial_master_nodes: ["node-1"]
16 bootstrap.system_call_filter: false
17 bootstrap.memory_lock: false
```

```
18 http.cors.enabled: true
19 http.cors.allow-origin: "*"

```

### 1.3.2、修改jvm.option

修改jvm.option配置文件，调整jvm堆内存大小

**node1.baiqi.cn使用baiqi用户**执行以下命令调整jvm堆内存大小，每个人根据自己的服务器的内存大小来进行调整。

```
1 cd /usr/local/es/elasticsearch-7.6.1/config
2 vi jvm.options
3 -Xms2g
4 -Xmx2g

```

## 2、修改系统配置，解决启动时候的问题

由于现在使用普通用户来安装es服务，且es服务对服务器的资源要求比较多，包括内存大小，线程数等。所以我们需要给普通用户解开资源的束缚

### 2.1、普通用户打开文件的最大数限制

**问题错误信息描述：**

max file descriptors [4096] for elasticsearch process likely too low,  
increase to at least [65536]

ES因为需要大量的创建索引文件，需要大量的打开系统的文件，所以我们需要解除linux系统当中打开文件最大数目的限制，不然ES启动就会抛错

**三台机器使用baiqi用户执行以下命令解除打开文件数据的限制**

`sudo vi /etc/security/limits.conf`

添加如下内容：注意\*不要去掉了

```
1 * soft nfile 65536
2 * hard nfile 131072
3 * soft nproc 4096
4 * hard nproc 4096

```

**此文件修改后需要重新登录用户，才会生效**

### 2.2、普通用户启动线程数限制

**问题错误信息描述**

max number of threads [1024] for user [es] likely too low, increase to at least [4096]

修改普通用户可以创建的最大线程数

max number of threads [1024] for user [es] likely too low, increase to at least [4096]原因：无法创建本地线程问题, 用户最大可创建线程数太小解决方案：修改90-nproc.conf 配置文件。

三台机器使用baiqi用户执行以下命令修改配置文件

```
1 Centos6
2 sudo vi /etc/security/limits.d/90-nproc.conf
3 Centos7
4 sudo vi /etc/security/limits.d/20-nproc.conf
```

找到如下内容：

```
1 * soft nproc 1024#修改为
2 * soft nproc 4096
```

## 2.3、普通用户调大虚拟内存

错误信息描述：

max virtual memory areas vm.max\_map\_count [65530] likely too low, increase to at least [262144]

调大系统的虚拟内存

原因：最大虚拟内存太小

每次启动机器都手动执行下。

三台机器执行以下命令

```
1 vi /etc/sysctl.conf, 追加以下内容: vm.max_map_count=655360 保存后, 执行: sysctl -p
```

备注：以上三个问题解决完成之后，重新连接secureCRT或者重新连接xshell生效

## 3、启动ES服务

三台机器使用baiqi用户执行以下命令启动es服务

```
cd /usr/local/es/elasticsearch-7.6.1/bin/
```

后台启动ES

```
./elasticsearch -d
```

启动成功之后jsp即可看到es的服务进程，并且访问页面

`http://192.168.21.128:9200/?pretty`

能够看到es启动之后的一些信息

注意：如果哪一台机器服务启动失败，那么就到哪一台机器的

`/usr/local/es/elasticsearch-7.6.1/log`

这个路径下面去查看错误日志

```
1 关闭Linux防火墙
2 永久性生效，重启后不会复原
3 开启: chkconfig iptables on
4 关闭: chkconfig iptables off
5 即时生效，重启后复原
6 开启: service iptables start
7 关闭: service iptables stop
8 centos7关闭防火墙 systemctl stop firewalld.service
```

注意:启动ES的时候出现 **Permission denied**

原因: 当前的用户没有对XX文件或目录的操作权限

## 五、客户端Kibana安装

1、客户端可以分为图形界面客户端, 和代码客户端.

2、ES主流客户端Kibana，开放9200端口与图形界面客户端交互

1) 下载Kibana放之`/usr/local/es`目录中

2) 解压文件: `tar -zxvf kibana-X.X.X-linux-x86_64.tar.gz`

3) 进入`/usr/local/es/kibana-X.X.X-linux-x86_64/config`目录

4) 使用vi编辑器: `vi kibana.yml`

```
1  server.port: 5601
2  server.host: "服务器IP"
3  elasticsearch.hosts: ["http://IP:9200"] #这里是elasticsearch的访问地址
```

5) 启动Kibana

启动kibana

```
1 cd /usr/local/es/kibana-7.6.1-linux-x86_64/bin
```

`./kibana`

## 后台启动kibana

```
1 nohup ./kibana &
```

## 6) 访问Kibana

http://ip:5601/app/kibana

## 六、安装IK分词器

我们后续也需要使用Elasticsearch来进行中文分词，所以需要单独给Elasticsearch安装IK分词器插件。以下为具体安装步骤：

### 1、下载Elasticsearch IK分词器

<https://github.com/medcl/elasticsearch-analysis-ik/releases>

### 2、切换到baiqi用户，并在es的安装目录下/plugins创建ik

```
1 mkdir -p /usr/local/es/elasticsearch-7.6.1/plugins/ik
```

### 3、将下载的ik分词器上传并解压到该目录

```
1 cd /usr/local/es/elasticsearch-7.6.1/plugins/ik
2 解压 elasticsearch-analysis-ik-7.6.1.zip
```

### 4、重启Elasticsearch

### 5、测试分词效果

```
1 POST _analyze
2 {
3   "analyzer": "standard",
4   "text": "我爱你中国"
5 }
```

```
1 POST _analyze
2 {
3   "analyzer": "ik_smart",
4   "text": "中华人民共和国"
5 }
6 #ik_smart:会做最粗粒度的拆分
```

```
1 POST _analyze
2 {
3   "analyzer":"ik_max_word",
4   "text":"中华人民共和国"
5 }
6 #ik_max_word:会将文本做最细粒度的拆分
```

## 七、指定IK分词器作为默认分词器

ES的默认分词设置是standard，这个在中文分词时就比较尴尬了，会单字拆分，比如我搜索关键词“清华大学”，这时候会按“清”，“华”，“大”，“学”去分词，然后搜出来的都是些“清清的河水”，“中华儿女”，“地大物博”，“学而不思则罔”之类的莫名其妙的结果，这里我们就想把这个分词方式修改一下，于是呢，就想到了ik分词器，有两种ik\_smart和ik\_max\_word。

ik\_smart会将“清华大学”整个分为一个词，而ik\_max\_word会将“清华大学”分为“清华大学”，“清华”和“大学”，按需选其中之一就可以了。

修改默认分词方法(这里修改school\_index索引的默认分词为：ik\_max\_word)：

```
1 PUT /school_index
2 {
3   "settings" : {
4     "index" : {
5       "analysis.analyzer.default.type": "ik_max_word"
6     }
7   }
8 }
```

## 八、ES数据管理

### 1、ES数据管理概述

ES是面向文档(document oriented)的，这意味着它可以存储整个对象或文档(document)。

然而它不仅仅是存储，还会索引(index)每个文档的内容使之可以被搜索。

在ES中，你可以对文档（而非成行成列的数据）进行索引、搜索、排序、过滤。

ES使用JSON作为文档序列化格式。

JSON现在已经被大多语言所支持，而且已经成为NoSQL领域的标准格式。



ES存储的一个员工文档的格式示例：

```
1 {  
2   "email": "584614151@qq.com",  
3   "name": "张三",  
4   "age": 30,  
5   "interests": [ "篮球", "健身" ]  
6  
7 }
```

## 2、基本操作

### 1) 创建索引

格式：PUT /索引名称

```
1 举例：PUT /es_db
```

### 2) 查询索引

格式：GET /索引名称

```
1 举例：GET /es_db
```

### 3) 删除索引

格式：DELETE /索引名称

```
1 举例：DELETE /es_db
```

### 4) 添加文档

格式：PUT /索引名称/类型/id

```
1 举例：  
2 PUT /es_db/_doc/1  
3 {  
4   "name": "张三",  
5   "sex": 1,  
6   "age": 25,  
7   "address": "广州天河公园"  
8 }  
9  
10 PUT /es_db/_doc/2  
11 {  
12   "name": "李四",  
13   "sex": 1,
```

```
14  "age": 28,
15  "address": "广州荔湾大厦"
16  }
17
18  PUT /es_db/_doc/3
19  {
20  "name": "王五",
21  "sex": 0,
22  "age": 26,
23  "address": "广州白云山公园"
24  }
25
26  PUT /es_db/_doc/4
27  {
28  "name": "admin",
29  "sex": 0,
30  "age": 22,
31  "address": "长沙橘子洲头"
32  }
33
34  PUT /es_db/_doc/5
35  {
36  "name": "小明",
37  "sex": 0,
38  "age": 19,
39  "address": "长沙岳麓山"
40  }
```

## 5) 修改文档

```
1  格式: PUT /索引名称/类型/id
2  举例:
3  PUT /es_db/_doc/1
4  {
5  "name": "白起老师",
6  "sex": 1,
7  "age": 25,
8  "address": "张家界森林公园"
9  }
```

## 6) 查询文档

```
1  格式: GET /索引名称/类型/id
```

2 举例: `GET /es_db/_doc/1`

## 7) 删除文档

1 格式: `DELETE /索引名称/类型/id`

2 举例: `DELETE /es_db/_doc/1`

## 九、查询操作

### 1、查询当前类型中的所有文档 `_search`

1 格式: `GET /索引名称/类型/_search`

2 举例: `GET /es_db/_doc/_search`

3 `SQL`: `select * from student`

### 2、条件查询, 如要查询age等于28岁的 `_search?q=*:***`

1 格式: `GET /索引名称/类型/_search?q=*:***`

2 举例: `GET /es_db/_doc/_search?q=age:28`

3 `SQL`: `select * from student where age = 28`

### 3、范围查询, 如要查询age在25至26岁之间的 `_search?q=***[** T0 **]` 注意: `T0` 必须为大写

1 格式: `GET /索引名称/类型/_search?q=***[25 T0 26]`

2 举例: `GET /es_db/_doc/_search?q=age[25 T0 26]`

3 `SQL`: `select * from student where age between 25 and 26`

### 4、根据多个ID进行批量查询 `_mget`

1 格式: `GET /索引名称/类型/_mget`

2 举例: `GET /es_db/_doc/_mget`

```
3 {
4   "ids":["1","2"]
5 }
```

6 `SQL`: `select * from student where id in (1,2)`

### 5、查询年龄小于等于28岁的 `:<=`

1 格式: `GET /索引名称/类型/_search?q=age:<=**`

2 举例: `GET /es_db/_doc/_search?q=age:<=28`

3 `SQL`: `select * from student where age <= 28`

### 6、查询年龄大于28前的 `:>`

1 格式: `GET /索引名称/类型/_search?q=age:>**`

```
2 举例: GET /es_db/_doc/_search?q=age:>28
3 SQL:  select * from student where age > 28
```

## 7、分页查询 from=&size=\*

```
1 格式: GET /索引名称/类型/_search?q=age[25 TO 26]&from=0&size=1
2 举例: GET /es_db/_doc/_search?q=age[25 TO 26]&from=0&size=1
3 SQL:  select * from student where age between 25 and 26 limit 0, 1
```

## 8、对查询结果只输出某些字段 \_source=字段, 字段

```
1 格式: GET /索引名称/类型/_search?_source=字段, 字段
2 举例: GET /es_db/_doc/_search?_source=name, age
3 SQL:  select name, age from student
```

## 9、对查询结果排序 sort=字段:desc/asc

```
1 格式: GET /索引名称/类型/_search?sort=字段 desc
2 举例: GET /es_db/_doc/_search?sort=age:desc
3 SQL:  select * from student order by age desc
```

# 十、DSL语言高级查询

ES提供了强大的查询语言（DSL），它可以允许我们进行更加强大、复杂的查询，Elasticsearch DSL中有Query与Filter两种

1、Query方式查询，会在ES中索引的数据都会存储一个\_score分值，分值越高就代表越匹配。另外关于某个搜索的分值计算还是很复杂的，因此也需要一定的时间。

- 根据名称精确查询姓名 term，term查询不会对字段进行分词查询，会采用精确匹配

注意：采用term精确查询，查询字段映射类型属于为keyword.

举例：

```
1 POST /es_db/_doc/_search
2 {
3   "query": {
4     "term": {
5       "name": "admin"
```

```
6  }
7  }
8  }
```

```
1 SQL: select * from student where name = 'admin'
```

- 根据备注信息模糊查询 match, match会根据该字段的分词器, 进行分词查询

举例:

```
1 POST /es_db/_doc/_search
2 {
3   "from": 0,
4   "size": 2,
5   "query": {
6     "match": {
7       "address": "广州"
8     }
9   }
10 }
```

```
1 SQL: select * from user where address like '%广州%' limit 0, 2
```

- 多字段模糊匹配查询与精准查询 multi\_match

```
1 POST /es_db/_doc/_search
2 {
3   "query":{
4     "multi_match":{
5       "query":"张三",
6       "fields":["address","name"]
7     }
8   }
9 }
```

```
1 SQL: select * from student where name like '%张三%' or address like '%张三%'
```

- 范围查询

注: json请求字符串中部分字段的含义

range: 范围关键字

gte 大于等于

lte 小于等于

gt 大于

lt 小于

now 当前时间

```
1 POST /es_db/_doc/_search
2 {
3   "query" : {
4     "range" : {
5       "age" : {
6         "gte":25,
7         "lte":28
8       }
9     }
10  }
11 }
```

```
1 SQL: select * from user where age between 25 and 28
```

- 分页、输出字段、排序综合查询

```
1 POST /es_db/_doc/_search
2 {
3   "query" : {
4     "range" : {
5       "age" : {
6         "gte":25,
7         "lte":28
8       }
9     }
10  },
11   "from": 0,
12   "size": 2,
13   "_source": ["name", "age", "book"],
14   "sort": {"age":"desc"}
15 }
```

2、Filter过滤器方式查询，它的查询不会计算相关性分值，也不会对结果进行排序，因此效率会高一点，查询的结果可以被缓存。

Filter Context 对数据进行过滤

```
1 POST /es_db/_doc/_search
2 {
3   "query" : {
4     "bool" : {
5       "filter" : {
6         "term":{
7           "age":25
8         }
9       }
10    }
11  }
12 }
```

十一、文档映射

1、ES中映射可以分为动态映射和静态映射

动态映射：

在关系数据库中，需要事先创建数据库，然后在该数据库下创建数据表，并创建表字段、类型、长度、主键等，最后才能基于表插入数据。而Elasticsearch中不需要定义Mapping映射（即关系型数据库的表、字段等），在文档写入Elasticsearch时，会根据文档字段自动识别类型，这种机制称之为动态映射。

动态映射规则如下：

| JSON数据     | 自动推测的类型     |
|------------|-------------|
| null       | 没有字段被添加     |
| true或false | boolean型    |
| 小数         | float型      |
| 数字         | long型       |
| 日期         | date或text   |
| 字符串        | text        |
| 数组         | 由数组第一个非空值决定 |
| JSON对象     | object类型    |

静态映射：

静态映射是在Elasticsearch中也可以事先定义好映射，包含文档的各字段类型、分词器等，这种方式称之为静态映射。

## 2、动态映射

### 2.1、删除原创建的索引

```
1 DELETE /es_db
```

### 2.2、创建索引

```
1 PUT /es_db
```

### 2.3、创建文档(ES根据数据类型，会自动创建映射)

```
1 PUT /es_db/_doc/1
2 {
3   "name": "Jack",
4   "sex": 1,
5   "age": 25,
6   "book": "java入门至精通",
7   "address": "广州小蛮腰"
8 }
```

### 2.4、获取文档映射

```
1 GET /es_db/_mapping
```

## 3、静态映射

### 3.1、删除原创建的索引

```
1 DELETE /es_db
```

### 3.2、创建索引

```
1 PUT /es_db
```

### 3.3、设置文档映射

```
1 PUT /es_db
2 {
3   "mappings":{
4     "properties":{
5       "name":{"type":"keyword","index":true,"store":true},
```



```
6 "sex":{"type":"integer","index":true,"store":true},
7 "age":{"type":"integer","index":true,"store":true},
8 "book":{"type":"text","index":true,"store":true},
9 "address":{"type":"text","index":true,"store":true}
10 }
11 }
12 }
```

### 3.4、根据静态映射创建文档

```
1 PUT /es_db/_doc/1
2 {
3   "name": "Jack",
4   "sex": 1,
5   "age": 25,
6   "book": "elasticSearch入门至精通",
7   "address": "广州车陂"
8 }
```

### 3.5、获取文档映射

```
1 GET /es_db/_mapping
```

## 十二、keyword 与 text 映射类型的区别

将 book 字段设置为 keyword 映射（只能精准查询，不能分词查询，能聚合、排序）

```
1 POST /es_db/_doc/_search
2 {
3   "query": {
4     "term": {
5       "book": "elasticSearch入门至精通"
6     }
7   }
8 }
```

将 book 字段设置为 text 映射（能模糊查询，能分词查询，不能聚合、排序）

```
1 POST /es_db/_doc/_search
```

```
2 {
3   "query": {
4     "match": {
5       "book": "elasticsearch入门至精通"
6     }
7   }
8 }
```

### 十三、创建静态映射时指定text类型的ik分词器

#### 1、设置ik分词器的文档映射

先删除之前的es\_db

再创建新的es\_db

定义ik\_smart的映射

```
1 PUT /es_db
2 {
3   "mappings":{
4     "properties":{
5       "name":{"type":"keyword","index":true,"store":true},
6       "sex":{"type":"integer","index":true,"store":true},
7       "age":{"type":"integer","index":true,"store":true},
8       "book":{"type":"text","index":true,"store":true,
9         "analyzer":"ik_smart",
10        "search_analyzer":"ik_smart"},
11       "address":{"type":"text","index":true,"store":true}
12     }
13   }
14 }
```

#### \* 分词查询

```
1 POST /es_db/_doc/_search
2 {
3   "query": {
4     "match": {
5       "address": "广东"
6     }
7   }
8 }
9
```

```
10 POST /es_db/_doc/_search
11 {
12   "query": {
13     "match": {
14       "address": "广州"
15     }
16   }
17 }
18
```

## 十四、ES集群环境搭建

### 1、修改elasticsearch.yml

node1.baiqi.cn 服务器使用baiqi用户来修改配置文件

```
1 cd /usr/local/es/elasticsearch-7.6.1/config
2 mkdir -p /usr/local/es/elasticsearch-7.6.1/log
3 mkdir -p /usr/local/es/elasticsearch-7.6.1/data
4
5
6 vim elasticsearch.yml
7 cluster.name: baiqi-es
8 node.name: node1.baiqi.cn
9 path.data: /usr/local/es/elasticsearch-7.6.1/data
10 path.logs: /usr/local/es/elasticsearch-7.6.1/log
11 network.host: node1.baiqi.cn
12 http.port: 9200
13 discovery.seed_hosts: ["IP1", "IP2", "IP3"]
14 cluster.initial_master_nodes: ["节点1名称", "节点2名称", "节点3名称"]
15 bootstrap.system_call_filter: false
16 bootstrap.memory_lock: false
17 http.cors.enabled: true
18 http.cors.allow-origin: "*"

```

### 2、修改jvm.option

修改jvm.option配置文件，调整jvm堆内存大小

node1.baiqi.cn使用baiqi用户执行以下命令调整jvm堆内存大小，每个人根据自己服务器的内存大小来进行调整。

```
1 cd /usr/local/es/elasticsearch-7.6.1/config
2 vim jvm.options

```

```
3 -Xms2g
4 -Xmx2g
```

### 3、将安装包分发到其他服务器上面

### 4、node2与node3修改es配置文件

node2. baiqi. cn与node3. baiqi. cn也需要修改es配置文件

node2. baiqi. cn使用baiqi用户执行以下命令修改es配置文件

```
1 cd /usr/local/es/elasticsearch-7.6.1/config
2 mkdir -p /usr/local/es/elasticsearch-7.6.1/log
3 mkdir -p /usr/local/es/elasticsearch-7.6.1/data
4
5 vim elasticsearch.yml
6 cluster.name: baiqi-es
7 node.name: node2.baiqi.cn
8 path.data: /usr/local/es/elasticsearch-7.6.1/data
9 path.logs: /usr/local/es/elasticsearch-7.6.1/log
10 network.host: node2.baiqi.cn
11 http.port: 9200
12 discovery.seed_hosts: ["IP1", "IP2", "IP3"]
13 cluster.initial_master_nodes: ["节点1名称", "节点2名称", "节点3名称"]
14 bootstrap.system_call_filter: false
15 bootstrap.memory_lock: false
16 http.cors.enabled: true
17 http.cors.allow-origin: "*"

```

node3. baiqi. cn使用baiqi用户执行以下命令修改配置文件

```
1 cd /usr/local/es/elasticsearch-7.6.1/config
2 mkdir -p /usr/local/es/elasticsearch-7.6.1/log
3 mkdir -p /usr/local/es/elasticsearch-7.6.1/data
4
5 vim elasticsearch.yml
6 cluster.name: baiqi-es
7 node.name: node3.baiqi.cn
8 path.data: /usr/local/es/elasticsearch-7.6.1/data
9 path.logs: /usr/local/es/elasticsearch-7.6.1/log
10 network.host: node3.baiqi.cn
11 http.port: 9200

```

```
12 discovery.seed_hosts: ["IP1", "IP2", "IP3"]
13 cluster.initial_master_nodes: ["节点1名称", "节点2名称", "节点3名称"]
14 bootstrap.system_call_filter: false
15 bootstrap.memory_lock: false
16 http.cors.enabled: true
17 http.cors.allow-origin: "*"

```

## 十五、Elasticsearch-head插件

由于es服务启动之后，访问界面比较丑陋，为了更好的查看索引库当中的信息，我们可以通过安装elasticsearch-head这个插件来实现，这个插件可以更方便快捷的看到es的管理界面

elasticsearch-head这个插件是es提供的一个用于图形化界面查看的一个插件工具，可以安装上这个插件之后，通过这个插件来实现我们通过浏览器查看es当中的数据

安装elasticsearch-head这个插件这里提供两种方式进行安装，第一种方式就是自己下载源码包进行编译，耗时比较长，网络较差的情况下，基本上不可能安装成功。

第二种方式就是直接使用我已经编译好的安装包，进行修改配置即可

要安装elasticsearch-head插件，需要先安装Node.js

### 1、安装nodejs

Node.js是一个基于 Chrome V8 引擎的 JavaScript 运行环境。

Node.js是一个Javascript运行环境(runtime environment)，发布于2009年5月，由Ryan Dahl开发，实质是对Chrome V8引擎进行了封装。Node.js 不是一个JavaScript 框架，不同于CakePHP、Django、Rails。Node.js 更不是浏览器端的库，不能与 jQuery、ExtJS 相提并论。Node.js 是一个让 JavaScript 运行在服务端的开发平台，它让 JavaScript 成为与PHP、Python、Perl、Ruby 等服务端语言平起平坐的脚本语言。

#### 1.1、下载安装包

node1.baiqi.cn机器执行以下命令下载安装包，然后进行解压

```
1 cd /usr/local/es

```

```
2 wget https://npm.taobao.org/mirrors/node/v8.1.0/node-v8.1.0-linux-x64.tar.gz
3 tar -zxvf node-v8.1.0-linux-x64.tar.gz -C /usr/local/es/
```

## 1.2、创建软连接

node1.baiqi.cn执行以下命令创建软连接

```
1 sudo ln -s /usr/local/es/node-v8.1.0-linux-x64/lib/node_modules/npm/bin/npm-
cli.js /usr/local/bin/npm
2 sudo ln -s /usr/local/es/node-v8.1.0-linux-x64/bin/node /usr/local/bin/node
```

## 1.3、修改环境变量

node1.baiqi.cn服务器添加环境变量

```
1 vi /etc/profile
2 export NODE_HOME=/usr/local/es/node-v8.1.0-linux-x64
3 export PATH=$PATH:$NODE_HOME/bin
```

修改完环境变量使用source生效

```
1 source /etc/profile
```

## 1.4、验证安装成功

node1.baiqi.cn执行以下命令验证安装生效

```
1 node -v
2 npm -v
```

## 2、本地安装

### 2.1、上传压缩包到/usr/local/es路径下去

将我们的压缩包 elasticsearch-head-compile-after.tar.gz 上传到服务器的/usr/local/es 路径下面去

### 2.2、解压安装包

在服务器中执行以下命令解压安装包

```
1 cd /usr/local/es/
2 tar -zxvf elasticsearch-head-compile-after.tar.gz -C /usr/local/es/
```

### 2.3、node1.baiqi.cn机器修改Gruntfile.js

## 修改Gruntfile.js这个文件

```
1 cd /usr/local/es/elasticsearch-head
2 vim Gruntfile.js
```

找到代码中的93行: hostname: '192.168.100.100', 修改为: node1.baiqi.cn

```
1 connect: {
2   server: {
3     options: {
4       hostname: 'node1.baiqi.cn',
5       port: 9100,
6       base: '.',
7       keepalive: true
8     }
9   }
10 }
```

## 2.4、node1机器修改app.js

### 第一台机器修改app.js

```
1 cd /usr/local/es/elasticsearch-head/_site
2 vim app.js
```

```
1 在Vim中输入「:4354」, 定位到第4354行, 修改 http://localhost:9200为
http://node1.baiqi.cn:9200
```

## 2.5、启动head服务

### node1.baiqi.cn启动elasticsearch-head插件

```
1 cd /usr/local/es/elasticsearch-head/node_modules/grunt/bin/
```

### 进程前台启动命令

```
1 ./grunt server
```

### 进程后台启动命令

```
1 nohup ./grunt server >/dev/null 2>&1 &
```

```
1 Running "connect:server" (connect) taskWaiting forever...Started connect web
server on http://192.168.52.100:9100
2 如何停止: elasticsearch-head进程
3 执行以下命令找到elasticsearch-head的插件进程, 然后使用kill -9 杀死进程即可
```

```
4 netstat -nltp | grep 9100
5 kill -9 8328
```

```
[es@node01 bin]$ netstat -nltp | grep 9100
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 192.168.52.100:9100      0.0.0.0:*               LISTEN      8328/grunt
[es@node01 bin]$ kill -9 8328
```

## 2.6、访问elasticsearch-head界面

打开Google Chrome访问

<http://ip:9100/>

注意：搭建es集群，启动三个es节点，访问elasticsearch-head时只显示一个master

解决方案：进到节点2、3的/elasticsearch-7.6.1/data/目录下删除nodes文件，之后重启节点2、3的es进程即可

文档：1、ElasticSearch快速入门与集群高可用...

链接：[http://note.youdao.com/noteshare?](http://note.youdao.com/noteshare?id=c9f03451f751beb5f913bbfbd9818ed1&sub=0A07974608094B04B16632B407F4AEC0)

[id=c9f03451f751beb5f913bbfbd9818ed1&sub=0A07974608094B04B16632B407F4AEC0](http://note.youdao.com/noteshare?id=c9f03451f751beb5f913bbfbd9818ed1&sub=0A07974608094B04B16632B407F4AEC0)