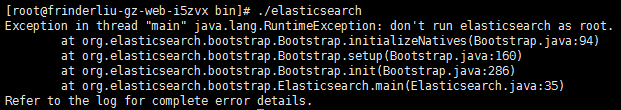
## 安装

* 安装：elasticsearch-2.4.1.zip；
  + 使用命令：unzip elasticsearch-2.4.1.zip –d /opt 中，直接解压即可；
  + 当使用 root 用户启动时，报错：



在网上查是因为安全等的问题，使用非root用户启动即可解决；

* 在线安装插件：elasticsearch-head
  + 使用命令：./plugin -install mobz/elasticsearch-head，报错：



* + 使用命令：./plugin install mobz/elasticsearch-head 即可；
* 离线安装 elasticsearch-head
  + 下载：elasticsearch-head-master.zip，并解压为：elasticsearch-head-master；
  + cp elasticsearch-head-master /opt/elasticsearch/plugins；
  + 修改名称为head：mv elasticsearch-head-master head；
  + 也可以使用命令安装：
    - ./plugin install file:/home/soft/elasticsearch-head-master.zip
  + 访问：http://10.199.195.228:9200/\_plugin/head/
* 安装license/ marvel-agent
  + 使用 ./plugin install file:/xx.zip的方式安装
* 安装Kibana
  + 解压即可
* 安装sense，使用Kibana安装
  + 切换到kibana/bin目录，执行命令：
    - ./kibana plugin -i sense -u file:/home/soft/sense-2.0.0-beta7.tar.gz
  + 访问Kibana
    - http://10.199.195.228:5601/app/sense

## 启动与集群搭建

* 启动： /opt/elasticsearch/bin/elasticsearch –d，

默认使用：/opt/elasticsearch/config/elasticsearch.yml配置文件；

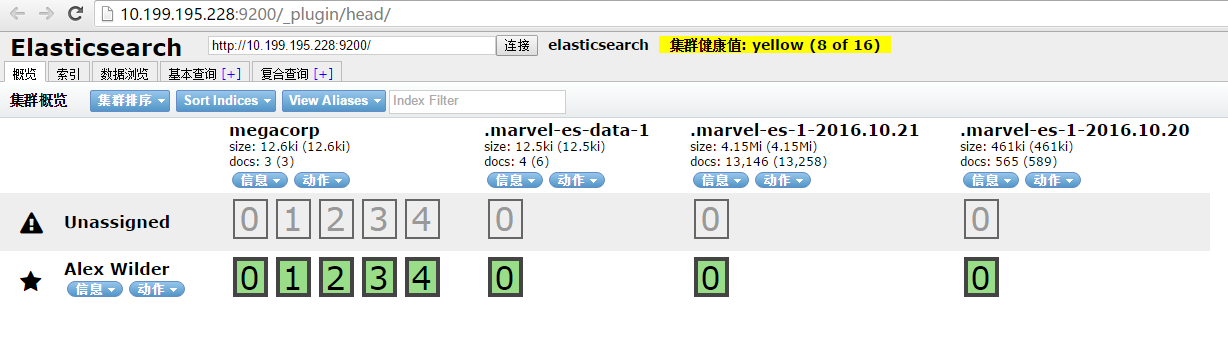
* 增加节点：
  + 复制 cp /opt/elasticsearch/config/elasticsearch.yml elasticsearch2.yml
  + 修改：http.port: 9200 为 http.port: 9201
  + 启动节点：./elasticsearch --../config/elasticsearch2.yml -d

## 概念性比较



## 简单使用

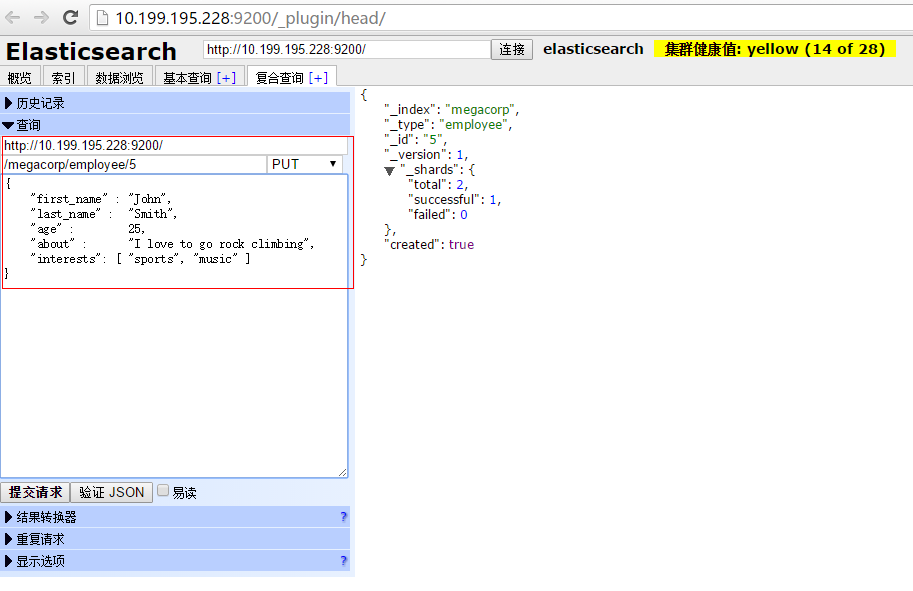
* + 1. 插入一条记录
* 访问：<http://196.168.0.1:9200/_plugin/head/>
* 页面如下：



* 切换到复合查询模板，如图：



* 切换到“PUT”命令方式，如图：



命令：

PUT /megacorp/employee/1

{

"first\_name" : "John",

"last\_name" : "Smith",

"age" : 25,

"about" : "I love to go rock climbing",

"interests": [ "sports", "music" ]

}

命令解释：

| **名字** | **说明** |
| --- | --- |
| megacorp | 索引名 |
| employee | 类型名 |
| 1 | 这个员工的ID |

* 检索文档：

GET /megacorp/employee/1

* 简单搜索：

GET /megacorp/employee/\_search

**注：默认情况下搜索会返回前10个结果**

查询名字为Fir的记录：

GET /megacorp/employee/\_search?q=last\_name:Fir

* 基于DSL查询：

GET /megacorp/employee/\_search

{

"query" : {

"match" : {

"last\_name" : "Fir"

}

}

}

* 复杂查询：

GET /megacorp/employee/\_search

{

"query" : {

"filtered" : {

"filter" : {

"range" : {

"age" : { "gt" : 30 } <1>

}

},

"query" : {

"match" : {

"last\_name" : "smith" <2>

}

}

}

}

}

* 全文检索（相关评分性查询）：

GET /megacorp/employee/\_search

{

"query" : {

"match" : {

"about" : "rock climbing"

}

}

}

查询结果：

{

...

"hits": {

"total": 2,

"max\_score": 0.16273327,

"hits": [

{

...

"\_score": 0.16273327, <1>

"\_source": {

"first\_name": "John",

"last\_name": "Smith",

"age": 25,

"about": "I love to go rock climbing",

"interests": [ "sports", "music" ]

}

},

{

...

"\_score": 0.016878016, <2>

"\_source": {

"first\_name": "Jane",

"last\_name": "Smith",

"age": 32,

"about": "I like to collect rock albums",

"interests": [ "music" ]

}

}

]

}

}

**注：<1><2> 结果相关性评分。**

**默认情况下，Elasticsearch根据结果相关性评分来对结果集进行排序，所谓的「结果相关性评分」就是文档与查询条件的匹配程度。很显然，排名第一的John Smith的about字段明确的写到“rock climbing”。**

**但是为什么Jane Smith也会出现在结果里呢？原因是“rock”在她的abuot字段中被提及了。因为只有“rock”被提及而“climbing”没有，所以她的\_score要低于John。**

**这个例子很好的解释了Elasticsearch如何在各种文本字段中进行全文搜索，并且返回相关性最大的结果集。相关性(relevance)的概念在Elasticsearch中非常重要，而这个概念在传统关系型数据库中是不可想象的，因为传统数据库对记录的查询只有匹配或者不匹配。**

* 短语检索：

GET /megacorp/employee/\_search

{

"query" : {

"match\_phrase" : {

"about" : "rock climbing"

}

}

}

* 高度检索结果：

GET /megacorp/employee/\_search

{

"query" : {

"match\_phrase" : {

"about" : "rock climbing"

}

},

"highlight": {

"fields" : {

"about" : {}

}

}

}

* 新增一条文档记录：
  + PUT
    - 提供id
  + POST
    - 系统生成id，自动生成的ID有22个字符长，URL-safe, Base64-encoded string universally unique identifiers, 或者叫[UUIDs](http://en.wikipedia.org/wiki/Uuid)，如：**wM0OSFhDQXGZAWDf0-drSA**