



Windows下ElasticSearch的Head安装及基本使用

🕒 发表于 2019-07-22 07:52 👁 阅读: 15085 💬 评论: 2 🌟 推荐: 14

ELASTICSEARCH ELASTICSEARCH HEAD

前段时间，有一朋友咨询我，说es的head插件一直安装失败，为了给朋友解惑，自己百度博文并实践了一番，也的确踩了些坑，但我给爬了起来。今天就来分享下实践心得并跳过的坑。

ElasticSearch 是一个分布式、高扩展、高实时的搜索与数据分析引擎，它能很方便的使大量数据具有搜索、分析和探索的能力，简称es。本文分五部分描述，es的安装，head插件安装，es的基本概念，es的基本使用，问题总结。

📖 目录

es安装

head插件安装

es基本概念

es基本使用

问题总结

📖 一、es安装

安装方式网络上有很多，这里简单说下步骤，具体实践是很简单的

①配置java环境

需要java环境，最好是较新的java环境，java环境的配置就略过了

②安装elasticsearch

下载地址：

<https://www.elastic.co/cn/downloads/elasticsearch>，最新版本已是7.2.0。下载后，解压到任意目录，我的路径是：

D:\elasticsearch-7.2.0

③启动elasticsearch

es的配置文件在config目录下，常用配置在elasticsearch.yml文件。我这里只是做学习目的，所以不修改此文件而直接启动es。在windows环境下启动es方法为命令行进入到bin\目录下，执行elasticsearch.bat，或者双击此文件以启动es。

启动日志如下：

```
figured
[2019-07-21T10:49:16,477][INFO ][o.e.c.c.Coordinator      ] [HONG] cluster UUID [EzMtKZbwQ5eJreWc61Gviw]
[2019-07-21T10:49:16,482][INFO ][o.e.c.c.ClusterBootstrapService] [HONG] no discovery configuration found, w
best-effort cluster bootstrapping after [3s] unless existing master is discovered
[2019-07-21T10:49:16,661][INFO ][o.e.c.s.MasterService      ] [HONG] elected-as-master ([1] nodes joined)[HON
9muQQFI1TRThw] [IXnqlxk1SG-h5NW7RVd8Qg] {127.0.0.1} {127.0.0.1:9300} [ml.machine_memory=8463183872, xpack.instal
,max_open_jobs=20] elect leader, BECOME_MASTER_TASK, FINISH_ELECTION, term: 4, version: 53, reason: mas
nged [previous [], current [HONG] [QNN9_DohQ9muQQFI1TRThw] [IXnqlxk1SG-h5NW7RVd8Qg] {127.0.0.1} {127.0.0.1:9300
,memory=8463183872, xpack.installed=true, ml.max_open_jobs=20}]
[2019-07-21T10:49:16,778][INFO ][o.e.c.s.ClusterApplierService] [HONG] master node changed (previous [], cur
[QNN9_DohQ9muQQFI1TRThw] [IXnqlxk1SG-h5NW7RVd8Qg] {127.0.0.1} {127.0.0.1:9300} [ml.machine_memory=8463183872, xp
d=true, ml.max_open_jobs=20]), term: 4, version: 53, reason: Publication{term=4, version=53}
[2019-07-21T10:49:17,339][INFO ][o.e.l.LicenseService       ] [HONG] license [a130fdd1-4938-4950-8e3d-3395b23e
basic] - valid
[2019-07-21T10:49:17,356][INFO ][o.e.g.GatewayService       ] [HONG] recovered [1] indices into cluster_state
[2019-07-21T10:49:18,022][INFO ][o.e.c.r.a.AllocationService] [HONG] Cluster health status changed from [RED
] (reason: [shards started [[test][1]] ...]).
[2019-07-21T10:49:19,616][INFO ][o.e.h.AbstractHttpServerTransport] [HONG] publish_address {127.0.0.1:9200},
sses {127.0.0.1:9200}, {:::1:9200}
搜狗拼音输入法 全 :9,616][INFO ][o.e.n.Node                ] [HONG] started
```

es的默认端口是9200，在浏览器打开网址：localhost:9200，出现如下截图，则说明es已经安装成功了

```
{
  "name" : "HONG",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "EzMtKZbwQ5eJreWc61Gviw",
  "version" : {
    "number" : "7.2.0",
    "build_flavor" : "default",
    "build_type" : "zip",
    "build_hash" : "508c38a",
    "build_date" : "2019-06-20T15:54:18.811730Z",
    "build_snapshot" : false,
    "lucene_version" : "8.0.0",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

es的安装是不是很简单？接下来来安装head插件

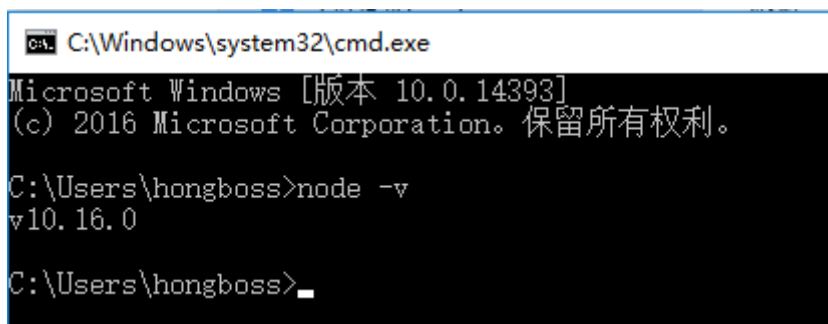
二、head插件安装

1. 安装node

es5以上就需要安装node和grunt，所以安装head插件的前提，是需要把该两项配置好。

node下载地址：<https://nodejs.org/en/download/>，下载对应环境的node版本安装即可，安装步骤略过了。

安装过程结束后，在dos窗口查看是否安装成功，使用命令：node -v，出现如下截图，则说明安装成功。



```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation。保留所有权利。

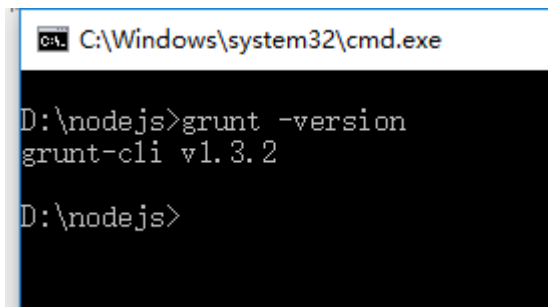
C:\Users\hongboss>node -v
v10.16.0

C:\Users\hongboss>
```

2. 安装grunt

在node安装路径下，使用命令安装：npm install -g grunt-cli 安装grunt。

安装结束后，使用命令grunt -version查看是否安装成功，出现如下截图，说明安装成功。



```
C:\Windows\system32\cmd.exe

D:\nodejs>grunt -version
grunt-cli v1.3.2

D:\nodejs>
```

3. 安装head插件

① 下载head插件

下载地址: <https://github.com/mobz/elasticsearch-head>, 下载zip包

② 解压zip包

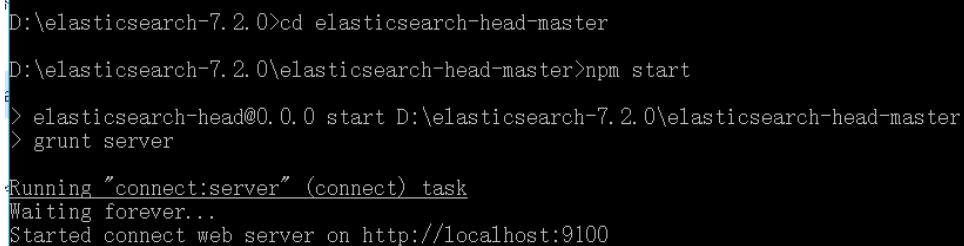
我的解压路径: D:\elasticsearch-7.2.0\elasticsearch-head-master

③ 安装pathomjs

在dos窗口进入到head路径下, 使用命令npm install安装pathomjs

④ 启用服务

使用命令npm start启用服务, 出现如下截图, 则说明服务启动成功



```
D:\elasticsearch-7.2.0>cd elasticsearch-head-master
D:\elasticsearch-7.2.0\elasticsearch-head-master>npm start
> elasticsearch-head@0.0.0 start D:\elasticsearch-7.2.0\elasticsearch-head-master
> grunt server

Running "connect:server" (connect) task
Waiting forever...
Started connect web server on http://localhost:9100
```

4. 浏览器中访问

使用地址: localhost:9100访问, 出现如下截图, 则说明head安装成功, 默认端口是9100



es和head插件都已安装成功，接下来介绍下es的基本概念及基本使用。

三、es的基本概念

① 集群和节点

一个es集群是由一个或多和es节点组成的集合，每一个集群都有一个名字，每个节点都有自己的名字，节点是可以存储数据，参与索引数据的独立服务。

② 索引(类似于数据库里面的database)

索引是含有相同属性的文档集合，索引在es中是通过一个名字来识别的，必须是英文字母小写，且不含中划线。

③ 类型(相当于sql中的table)

一个索引可以定义一个或多个类型，文档必须属于一个类型

④ 文档(相当于sql中的一行记录)

文档是可以被索引的基本数据单位

⑤ 分片

每个索引都有多个分片，每个分片都是一个luncene索引，分片的好处：分摊索引的搜索压力，分片还支持水平的拓展和拆分以及分布式的操作，可以提高搜索和其他处理的效率。

⑥ 备份

拷贝一个分片就完成了分片的备份，备份的好处：当主分片失败或者挂掉，备份就可以代替分片进行操作，进而提高了es的可用性，备份的分片还可以进行搜索操作，以分摊搜索的压力。es在创建索引时，默认创建5

个分片，一份备份，可以修改，分片的数量只能在创建索引的时候指定，索引创建后就不能修改分片的数量了，而备份是可以动态修改的。

? ⑦数据类型

核心类型 字符串类型 string, text, keyword

整数类型 integer, long, short, byte

浮点类型 double, float, half_float, scaled_float

逻辑类型 boolean

日期类型 date

范围类型 range

二进制类型 binary

复合类型 数组类型 array

对象类型 object

嵌套类型 nested

地理类型 地理坐标类型 geo_point

地理地图 geo_shape

特殊类型 IP类型 ip

范围类型 completion

令牌计数类型 token_count

附件类型 attachment

抽取类型 percolator

1/4 四、es基本使用

? 1.es基本格式

es是以RESTful风格来命名API的，其API的基本格式如下：

http://<ip>:<port>/<索引>/<类型>/<文档id>

这里需要注意的是，该格式从es7.0.0开始，移除Type（类型）这个概念，新的基本格式如下：

http://<ip>:<port>/<索引>/_doc/<文档id>

Type（类型）字段那里变为固定值 _doc

es的动作是以http方法来决定的：常用的http方法：

GET/PUT/POST/DELETE

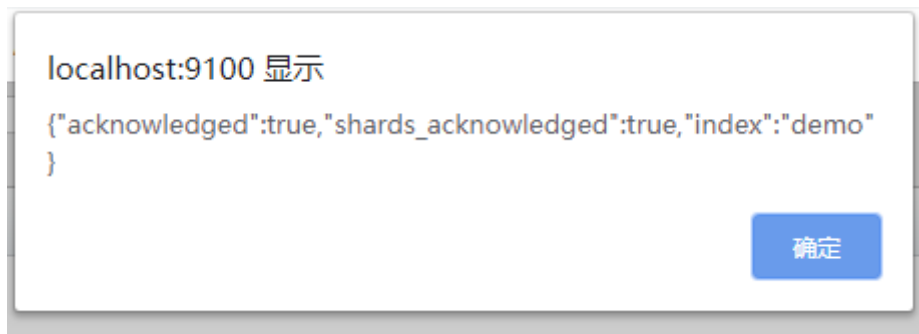
? 2.创建索引

在head插件中创建，操作如下：

点击索引>新建索引



弹出该提示，则说明创建成功

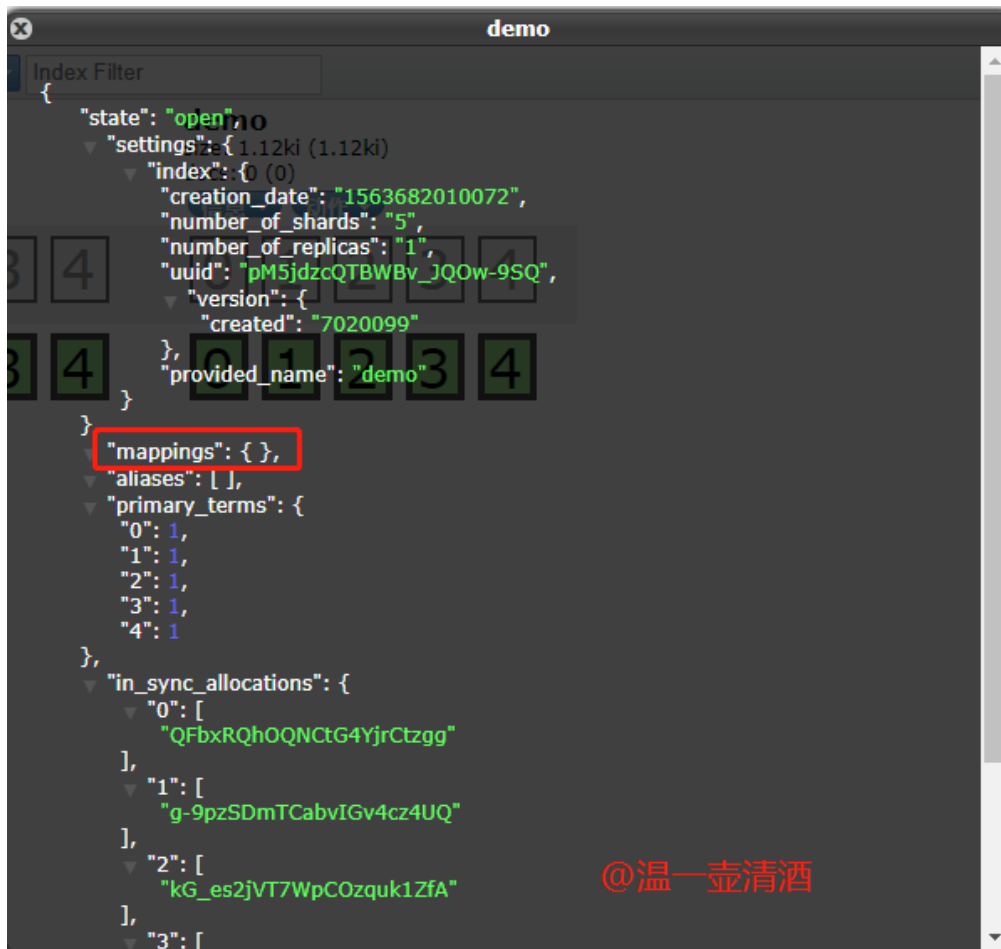


? 3.查看索引信息

点击概览查看创建情况

创建索引分为: **结构化创建与非结构化创建**

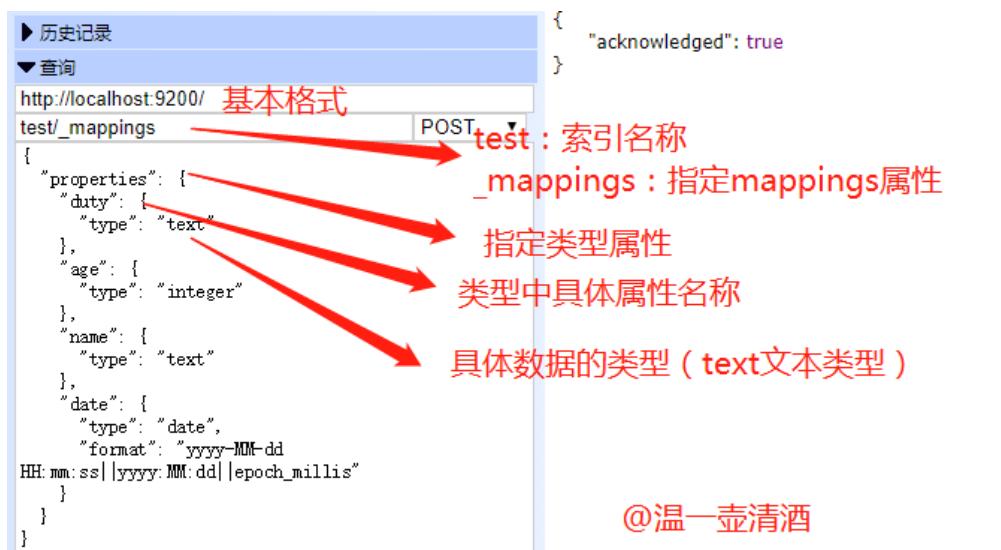
查看索引是否是结构化的方法: 点击刚创建的索引信息, 可查看到如下所示信息:



Mappings是结构化的一个关键词，其后内容是空的，说明这个索引是一个非结构化的索引。

? 4.创建结构化索引

点击head插件的“复合查询”，输入内容如下：

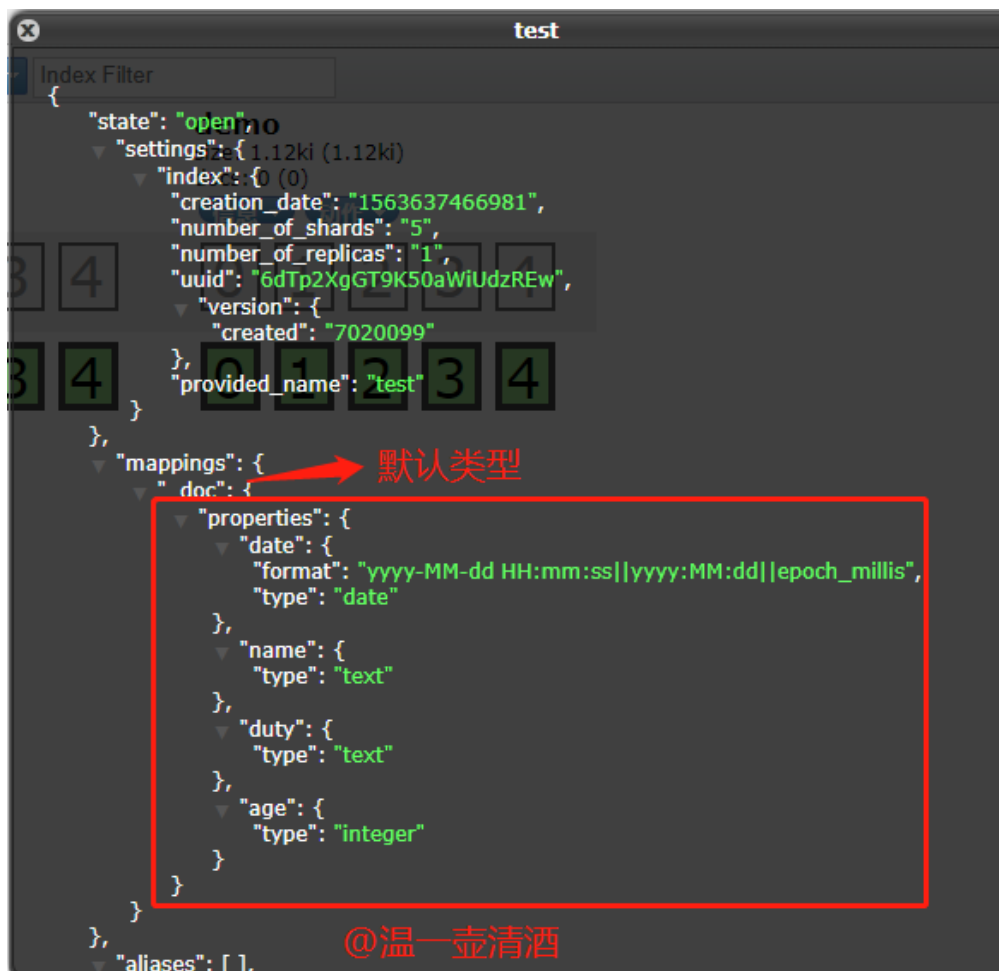


勾选易读，点击验证json，可以检测json格式是否正确

请求方式选择post，点击提交请求，返回如下截图数据，则表示创建成功



创建成功后，可以返回到概览中查看索引信息，如下：



也可以直接在复合查询中，改成get请求方式，提交请求，查看数据：



? 5.数据插入

文档id, 唯一索引值, 指向文档数据

①指定文档id插入

使用http中的**put**方法，插入时输入的ip地址，
`http://localhost:9200/test/_doc/1`

请求参数依次为：索引名称/类型名称/文档id

请求参数

```
{
  "duty": "技术",
  "age": 22,
  "name": "一壶清酒",
  "date": "2019-07-21 11:00:00"
}
```

如下图所示：

历史记录

▼ 查询

http://localhost:9200/

test/_doc/1 PUT

```
{
  "duty": "技术",
  "age": 22,
  "name": "温一壶清酒",
  "date": "2019-07-21 11:00:00"
}
```

@温一壶清酒

```
{
  "_index": "test",
  "_type": "_doc",
  "_id": "1",
  "_version": 4,
  "result": "updated",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 3,
  "_primary_term": 2
}
```

在数据浏览中可查看到该条数据，如下所示：

查询 10 个分片中用的 10 个, 3 命中, 耗时 0.230 秒

_index	_type	_id	_score	duty	age	name	date
test	_doc	0H98EmwBZYMBj4GoiMK1	1	测试	25	温一壶清酒	2019-07-21 11:05:00
test	_doc	2	1	技术小白	27	postman添加	2019-07-21 11:10:00
test	_doc	1	1	技术	22	一壶清酒	2019-07-21 11:00:00

@温一壶清酒

②自动产生文档id插入

使用http中的**post**方法，插入时输入的ip地址：

http://localhost:9200/test/_doc

请求参数

```
{
  "duty": "测试",
  "age": 25,
  "name": "温一壶清酒",
  "date": "2019-07-21 11:05:00"
}
```

Elasticsearch http://localhost:9200/ 连接 elasticsearch

概览 索引 数据浏览 基本查询 [+] 复合查询 [+]

历史记录

▼ 查询

http://localhost:9200/

test/_doc POST

```
{
  "duty": "测试",
  "age": 25,
  "name": "温一壶清酒",
  "date": "2019-07-21 11:05:00"
}
```

@温一壶清酒

```
{
  "_index": "test",
  "_type": "_doc",
  "_id": "0H98EmwBZYMBj4GoiMK1",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 0,
  "_primary_term": 2
}
```

依然到数据浏览中查看数据，如下所示，id为自动生成：

查询 10 个分片中用的 10 个, 3 命中, 耗时 0.230 秒

_index	_type	_id	_score ▲	duty	age	name	date
test	_doc	0H98EmwBZYmbj4GoiMK1	1	测试	25	温一壶清酒	2019-07-21 11:05:00
test	_doc	2	1	技术小白	27	postman添加	2019-07-21 11:10:00
test	_doc	1	1	技术	22	一壶清酒	2019-07-21 11:00:00

@温一壶清酒

③postman插入数据

操作方式一样，只是改成了postman而已，入参如下所示：

POST

http://localhost:9200/test/_doc/2

1 {

2 "duty": "技术小白",

3 "age": "27",

4 "name": "postman添加",

5 "date": "2019-07-21 11:10:00"

6 }

Body

Cookies

Headers (4)

Test Results

Pretty

Raw

Preview

JSON

1 {

2 "_index": "test",

3 "_type": "_doc",

4 "_id": "2",

5 "_version": 1,

6 "result": "created",

7 "_shards": {

8 "total": 2,

9 "successful": 1,

10 "failed": 0

11 },

12 "_seq_no": 0,

13 "_primary_term": 2

14 }

@温一壶清酒

到数据浏览处查看数据，如下：

查询 10 个分片中用的 10 个, 3 命中, 耗时 0.007 秒

_index	_type	_id	_score ▲	duty	age	name	date
test	_doc	0H98EmwBZYmbj4GoiMK1	1	测试	25	温一壶清酒	2019-07-21 11:05:00
test	_doc	2	1	技术小白	27	postman添加	2019-07-21 11:10:00
test	_doc	1	1	技术	27	我是一壶清酒	2019-07-21 11:06:00

@温一壶清酒

postman的操作，就只引用了这一个例子，其他操作都一样，所以就不再赘述。

6.修改文档数据

①直接修改文档

http方法: **post**方法

请求地址:

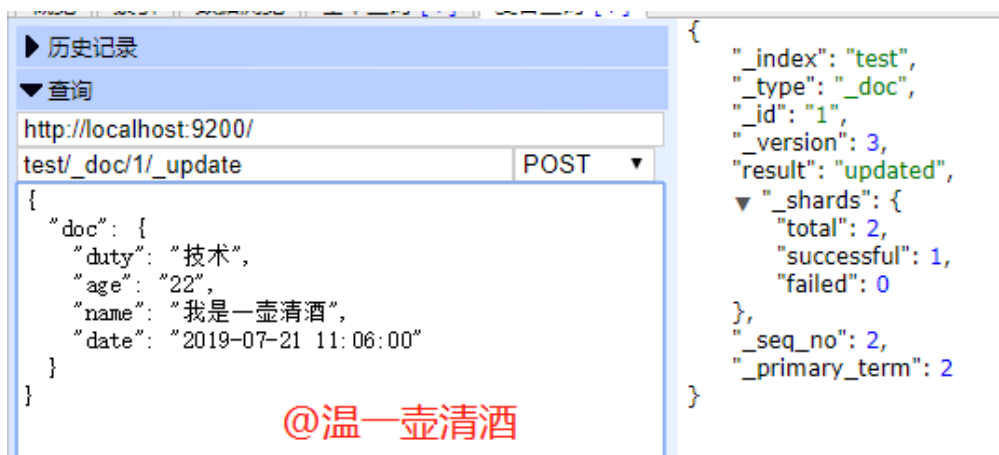
http://localhost:9200/test/_doc/1/_update

请求参数

```
{
  "doc": {
    "duty": "技术",
    "age": 22,
    "name": "我是一壶清酒",
    "date": "2019-07-21 11:06:00"
  }
}
```

关键词: _update, doc

“doc”为关键字，要修改的文档放在doc中，实例修改了type为test索引下_doc中id为1 的name和date属性



到数据浏览处查看修改后的数据，如下：

查询 10 个分片中用的 10 个, 3 命中, 耗时 0.006 秒

_index	_type	_id	_score ▲	duty	age	name	date
test	_doc	0H98EmwBZYMbj4GoiMK1	1	测试	25	温一壶清酒	2019-07-21 11:05:00
test	_doc	2	1	技术小白	27	postman添加	2019-07-21 11:10:00
test	_doc	1	1	技术	22	我是一壶清酒	2019-07-21 11:06:00

@温一壶清酒

②脚本修改文档

通过脚本修改的api格式与直接修改的是一致的

http方法: **post**方法

请求地址:

http://localhost:9200/test/_doc/1/_update

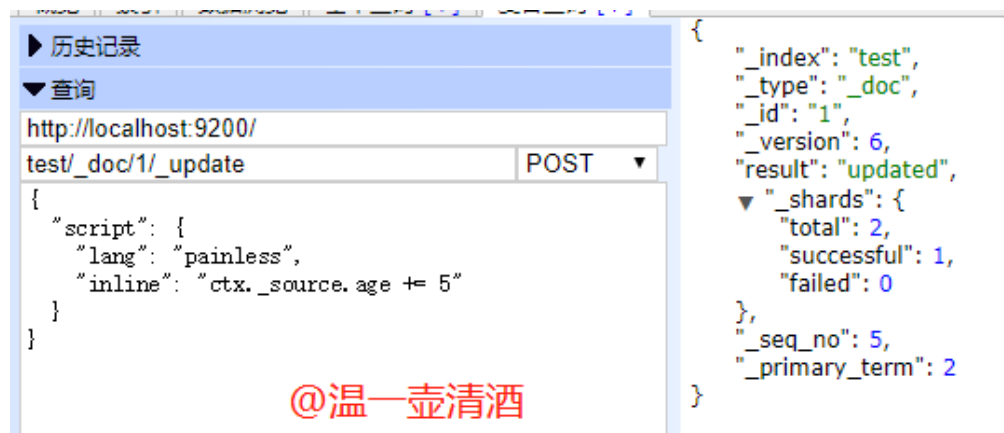
请求参数

```
{
  "script": {
    "lang": "painless",
    "inline": "ctx._source.age += 30"
  }
}
```

关键字“script”: 标志以脚本的方式修改文档

“lang”: 表示以何种脚本语言进行修改, “painless”表示以es内置的脚本语言进行修改。此外es还支持多种脚本语言, 如Python, js等等

“inline”: 指定脚本内容 “ctx”代表es上下文, _source 代表文档



查看数据, age增加了5, 如下所示:

查询 10 个分片中用的 10 个, 3 命中, 耗时 0.007 秒

_index	_type	_id	_score ▲	duty	age	name	date
test	_doc	0H98EmwBZYMBj4GoiMK1	1	测试	25	温一壶清酒	2019-07-21 11:05:00
test	_doc	2	1	技术小白	27	postman添加	2019-07-21 11:10:00
test	_doc	1	1	技术	27	我是一壶清酒	2019-07-21 11:06:00

A red box highlights the 'age' value of 27 for the document with _id 1. A red watermark '@温一壶清酒' is visible in the center.

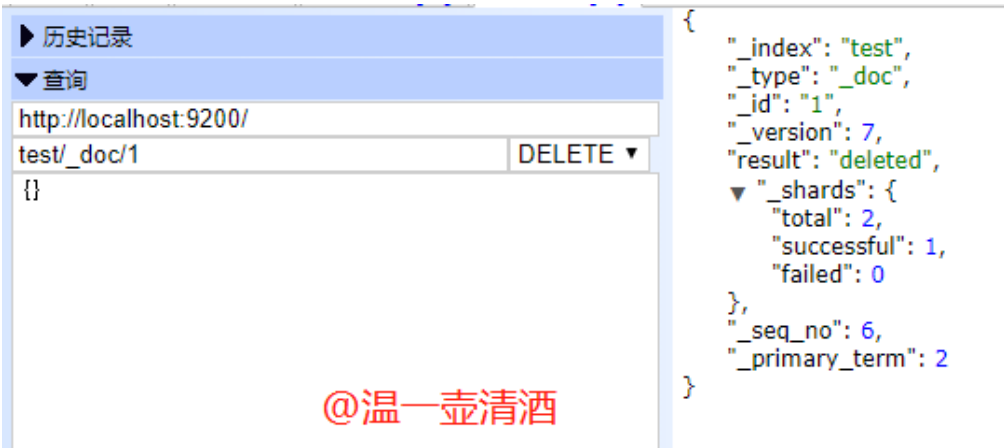
? 7.删除文档

http方法: **delete**

请求地址:

http://localhost:9200/test/_doc/1

操作如下：



到数据浏览处查看数据，已没有id为1的文档了，如下所示：

查询 10 个分片中用的 10 个, 2 命中, 耗时 0.628 秒

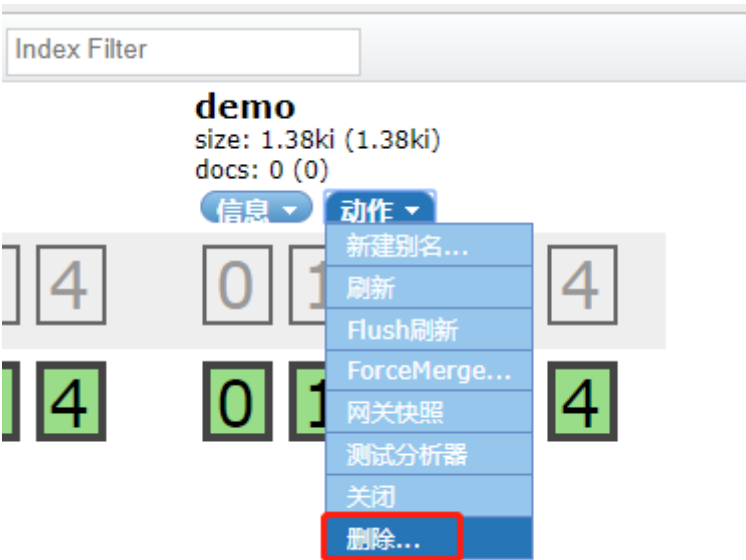
_index	_type	_id	_score ▲	duty	age	name	date
test	_doc	0H98EmwBZYMbj4GoiMK1	1	测试	25	温一壶清酒	2019-07-21 11:05:00
test	_doc	2	1	技术小白	27	postman添加	2019-07-21 11:10:00

@温一壶清酒

8.删除索引

①索引概览中删除

点击已有索引的动作，会有个删除操作，如下：



输入删除，点击确定，该索引就被删除了，会返回一个true的提示框



再次查看，就只有一个索引了



②通过api删除

http方法: **delete**

请求地址:

http://localhost:9200/demo



五、问题总结

1.grunt不是内部或外部命令

使用命令npm start启用服务时，报grunt不是内部或外部命令，如下所示：

```
D:\elasticsearch-7.2.0\elasticsearch-head-master>npm start
> elasticsearch-head@0.0.0 start D:\elasticsearch-7.2.0\elasticsearch-head-master
> grunt server

'grunt' 不是内部或外部命令，也不是可运行的程序
或批处理文件。
npm ERR! code ELIFECYCLE
npm ERR! errno 1
npm ERR! elasticsearch-head@0.0.0 start: `grunt server`
npm ERR! Exit status 1
npm ERR!
npm ERR! Failed at the elasticsearch-head@0.0.0 start script.
npm ERR! This is probably not a problem with npm. There is likely additional logging output above.
npm WARN Local package.json exists, but node_modules missing, did you mean to install?

npm ERR! A complete log of this run can be found in:
npm ERR! C:\Users\hongboss\AppData\Roaming\npm-cache\_logs\2019-07-20T03_13_15_695Z-debug.log
```

解决办法：

将node安装路径，加到环境变量path路径下，重启dos窗口即可。

2.head服务启用失败

使用npm start命令启用服务，报错如下：

```
D:\elasticsearch-7.2.0\elasticsearch-head-master>npm start
> elasticsearch-head@0.0.0 start D:\elasticsearch-7.2.0\elasticsearch-head-master
> grunt server

grunt-cli: The grunt command line interface (v1.3.2)

Fatal error: Unable to find local grunt.

If you're seeing this message, grunt hasn't been installed locally to
your project. For more information about installing and configuring grunt,
please see the Getting Started guide:

https://gruntjs.com/getting-started
npm ERR! code ELIFECYCLE
npm ERR! errno 99
npm ERR! elasticsearch-head@0.0.0 start: `grunt server`
npm ERR! Exit status 99
npm ERR!
npm ERR! Failed at the elasticsearch-head@0.0.0 start script.
npm ERR! This is probably not a problem with npm. There is likely additional logging output above.
npm WARN Local package.json exists, but node_modules missing, did you mean to install?

npm ERR! A complete log of this run can be found in:
npm ERR! C:\Users\hongboss\AppData\Roaming\npm-cache\_logs\2019-07-20T03_39_52_631Z-debug.log
```

@温一壶清酒

解决办法:

在es路径下执行命令 npm install, 因为没有安装pathomjs, 所以导致报错

3.提示未连接

head服务启用成功, 通过localhost:9100访问, 提示集群健康值: 未连接



@温一壶清酒

解决办法:

需要在elasticsearch.yml文件中增加以下配置:

http.cors.enabled: true

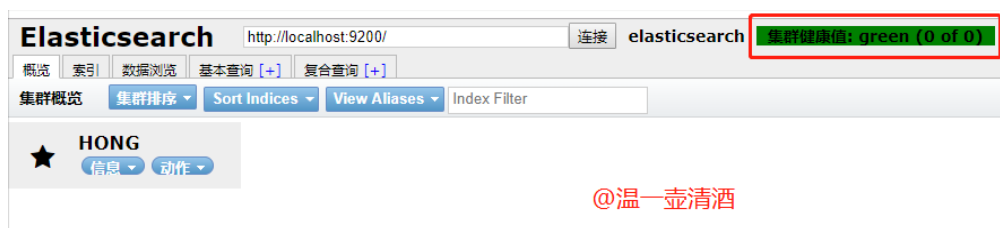
http.cors.allow-origin: ""

说明:

http.cors.enabled:true 如果启用了 HTTP 端口, 那么此属性会指定是否允许跨源 REST 请求。

http.cors.allowed.origin:"" 如果 http.cors.enabled 的值为 true, 那么该属性会指定允许 REST 请求来自何处。

重启服务, 再次访问, 则恢复正常



@温一壶清酒

✎ 小结

全文通读到此，发现ElasticSearch的安装、head插件的安装其实很简单，并不是那么难。es的使用，就需要学习些基本的概念，知识的海洋是渊博的，此次介绍的es的使用，也只是简单的增删改查，属于很浅薄的知识。自己在摸索的时候，也是一点一点学习，百度以前前辈们的博文学习、摸索。只要努力，遇到问题，积极去解决，我相信一定会成功。文中观点，有误之处，欢迎批评指正

This blog has running : 1217 d 8 h 38 m 59 s ㄣゝ∪!) / ♡
Copyright © 2021 温一壶清酒 Powered by .NET 5.0 on Kubernetes
Theme version: v1.3.2 / Loading theme version: v1.3.2