

 免费电子书

学习

Elasticsearch

Free unaffiliated eBook created from
Stack Overflow contributors.

#elasticsearch

ch

.....	1
1: Elasticsearch	2
.....	2
.....	2
Examples.....	2
Ubuntu 14.04Elasticsearch.....	3
.....	3
.....	3
Linux.....	3
WindowsElasticsearch.....	4
.....	4
.....	4
Windows	5
.....	5
.....	5
ID.....	6
.....	6
.....	8
CentOS 7ElasticsearchKibana.....	11
2: Elasticsearch	13
.....	13
.....	13
.....	14
.....	14
Examples.....	14
Elasticsearch.....	14
.....	15
.....	15
.....	16
.....	17
3: Python	19

.....	19
Examples.....	19
.....	19
.....	20
.....	20
.....	20
4:	22
.....	22
Examples.....	22
.....	22
.....	23
5:	26
.....	26
Examples.....	26
.....	26
.....	26
.....	27
.....	27
6:	29
.....	29
Examples.....	29
Curl.....	29
Id.....	29
.....	30
.....	30
.....	30
.....	30
7:	31
.....	31
.....	31
.....	32
.....	32

Examples.....	33
.....	33
.....	33
8: API.....	36
.....	36
Examples.....	36
.....	36
.....	36
.....	36
URI.....	36
9: kibanaElasticsearch.....	38
.....	38
Examples.....	38
Kibana.....	38
.....	39
10:	41
.....	41
Examples.....	41
Cluster Health.....	41
.....	42
.....	42
JSON.....	43
11:	44
.....	44
Examples.....	44
.....	44
.....	44
.....	44
.....	46

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [elasticsearch](#)

It is an unofficial and free Elasticsearch ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Elasticsearch.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: Elasticsearch

ElasticsearchLuceneJava。

HTTP REST API。

5.2.1	2017214
5.2.0	2017131
5.1.2	2017112
5.1.1	2016128
5.0.2	20161129
5.0.1	20161115
5.0.0	20161026
2.4.0	2016831
2.3.0	2016330
2.2.0	201622
2.1.0	20151124
2.0.0	20151028
1.7.0	2015-07-17
1.6.0	2015-06-25
1.5.0	201536
1.4.0	2014115
1.3.0	2014723
1.2.0	2014522
1.1.0	2014325
1.0.0	2014214

Examples

Ubuntu 14.04Elasticsearch

ElasticsearchJava Runtime EnvironmentJRE。 ElasticsearchJava 7Oracle JDK version 1.8.0_73
Oracle JDK version 1.8.0_73。

Oracle Java 8

```
sudo add-apt-repository -y ppa:webupd8team/java
sudo apt-get update
echo "oracle-java8-installer shared/accepted-oracle-license-v1-1 select true" | sudo debconf-
set-selections
sudo apt-get install -y oracle-java8-installer
```

Java

```
java -version
```

1. Elasticsearch。
- 2.

Linux

```
$ bin/elasticsearch
```

apt-get

elasticsearchapt-get。

```
wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
echo "deb https://packages.elastic.co/elasticsearch/2.x/debian stable main" | sudo tee -a
/etc/apt/sources.list.d/elasticsearch-2.x.list
sudo apt-get update && sudo apt-get install elasticsearch
sudo /etc/init.d/elasticsearch start
```

elasticsearch5.x.

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
sudo apt-get install apt-transport-https
echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a
/etc/apt/sources.list.d/elastic-5.x.list
sudo apt-get update && sudo apt-get install elasticsearch
```

Linux

- 。 。 ElasticsearchSysV initsystemd。 。

```
ps -p 1
```

SysV init

```
sudo update-rc.d elasticsearch defaults 95 10
sudo /etc/init.d/elasticsearch start
```

systemd

```
sudo /bin/systemctl daemon-reload
sudo /bin/systemctl enable elasticsearch.service
```

REST^{CURL}Elasticsearch^o

```
curl -X GET http://localhost:9200/
```

WindowsElasticsearch

WindowsElasticsearch <https://www.elastic.co/downloads/elasticsearch> ^o ^o

Windows.ZIP^o “Downloads:^o

“”^o ^o C:\elasticsearch^o

elasticsearch-<version>C:\elasticsearchC:\elasticsearch

ElasticsearchJavaJava Runtime Environment^o Java

```
java -version
```

```
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) Client VM (build 25.91-b14, mixed mode)
```

'java'^o

Java^o Java^o

JAVA_HOME	C:\Program Files\Java\jre
	...; C:\Program Files\Java\jre

^o

Java bin c:\elasticsearch\bin . elasticsearch.bat Elasticsearch . CTRL C

Windows

Elasticsearch

Elasticsearch C:\elasticsearch\config\jvm.options

```
jvm.options 32Windows-Xss320k [...]64Windows-Xss1mjvm.options
```

bin

```
C:\Users\user> cd c:\elasticsearch\bin
```

elasticsearch-service.bat service.bat

```
C:\elasticsearch\bin> elasticsearch-service.bat
```

```
Usage: elasticsearch-service.bat install|remove|start|stop|manager [SERVICE_ID]
```

SERVICE_ID

```
C:\elasticsearch\bin> elasticsearch-service.bat install
```

.

```
C:\elasticsearch\bin> elasticsearch-service.bat start
```

```
C:\elasticsearch\bin> elasticsearch-service.bat stop
```

GUI

```
C:\elasticsearch\bin> elasticsearch-service.bat manager
```

Elastic Service Manager/

Elasticsearch HTTP REST API cURL. JSON. Elasticsearch 9200.

?pretty Elasticsearch JSON .

ID megacorpemployeeID 1

```
curl -XPUT "http://localhost:9200/megacorp/employee/1?pretty" -d'
{
  "first_name" : "John",
  "last_name" : "Smith",
```

```
"age" :      25,
"about" :    "I love to go rock climbing",
"interests": [ "sports", "music" ]
}'
```

```
{
  "_index": "megacorp",
  "_type": "employee",
  "_id": "1",
  "_version": 1,
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "created": true
}
```

PUT.

ID

```
POST /megacorp/employee?pretty
{
  "first_name" : "Jane",
  "last_name"  : "Smith",
  "age" :      32,
  "about" :    "I like to collect rock albums",
  "interests": [ "music" ]
}
```

```
{
  "_index": "megacorp",
  "_type": "employee",
  "_id": "AVYg2mBJYy9ijdngfeGa",
  "_version": 1,
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "created": true
}
```

```
curl -XGET "http://localhost:9200/megacorp/employee/1?pretty"
```

```
{
  "_index": "megacorp",
  "_type": "employee",
  "_id": "1",
  "_version": 1,
```

```
"found": true,
"_source": {
  "first_name": "John",
  "last_name": "Smith",
  "age": 25,
  "about": "I love to go rock climbing",
  "interests": [
    "sports",
    "music"
  ]
}
}
```

employeemegacorp10

```
curl -XGET "http://localhost:9200/megacorp/employee/_search?pretty"
```

```
{
  "took": 2,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 1,
    "hits": [
      {
        "_index": "megacorp",
        "_type": "employee",
        "_id": "1",
        "_score": 1,
        "_source": {
          "first_name": "John",
          "last_name": "Smith",
          "age": 25,
          "about": "I love to go rock climbing",
          "interests": [
            "sports",
            "music"
          ]
        }
      },
      {
        "_index": "megacorp",
        "_type": "employee",
        "_id": "AVYg2mBJYy9ijdngfeGa",
        "_score": 1,
        "_source": {
          "first_name": "Jane",
          "last_name": "Smith",
          "age": 32,
          "about": "I like to collect rock albums",
          "interests": [
            "music"
          ]
        }
      }
    ]
  }
}
```

```
    }
  }
]
}
}
```

match

```
curl -XGET "http://localhost:9200/megacorp/employee/_search" -d'
{
  "query" : {
    "match" : {
      "last_name" : "Smith"
    }
  }
}'
```

```
{
  "took": 2,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "failed": 0
  },
  "hits": {
    "total": 1,
    "max_score": 0.6931472,
    "hits": [
      {
        "_index": "megacorp",
        "_type": "employee",
        "_id": "1",
        "_score": 0.6931472,
        "_source": {
          "first_name": "John",
          "last_name": "Smith",
          "age": 25,
          "about": "I love to go rock climbing",
          "interests": [
            "sports",
            "music"
          ]
        }
      }
    ]
  }
}
```

◦ `_source` ◦ `_source` **false** ◦

account_numberbalance _source

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '
```

```
"query": { "match_all": { } },
"_source": ["account_number", "balance"]
}'
```

`_source` ◦ `_source` `account_number` `balance` ◦

SQLSQL

```
SELECT account_number, balance FROM bank;
```

◦ `match_all` ◦ ◦

`account_number` 20

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '{
  "query": { "match": { "account_number": 20 } }
}'
```

`address` “mill”

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '{
  "query": { "match": { "address": "mill" } }
}'
```

“” `address`

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '{
  "query": { "match": { "address": "mill lane" } }
}'
```

`match` `match_phrase` `match_phrase` `address` [\[1\]](#) ◦

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '{
  "query": { "match_phrase": { "address": "mill lane" } }
}'
```

`boolean` ◦ `bool` ◦

“mill” “lane”

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '{
  "query": {
    "bool": {
      "must": [
        { "match": { "address": "mill" } },
        { "match": { "address": "lane" } }
      ]
    }
  }
}'
```

```
}  
}  
'
```

bool must true

address "mill" "lane"

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '  
{  
  "query": {  
    "bool": {  
      "should": [  
        { "match": { "address": "mill" } },  
        { "match": { "address": "lane" } }  
      ]  
    }  
  }  
}'
```

bool should true

address "mill" "lane"

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '  
{  
  "query": {  
    "bool": {  
      "must_not": [  
        { "match": { "address": "mill" } },  
        { "match": { "address": "lane" } }  
      ]  
    }  
  }  
}'
```

bool must_not true

bool must should must_not bool bool

40_{WA}

```
curl -XPOST 'localhost:9200/bank/_search?pretty' -d '  
{  
  "query": {  
    "bool": {  
      "must": [  
        { "match": { "age": "40" } }  
      ],  
      "must_not": [  
        { "match": { "state": "WA" } }  
      ]  
    }  
  }  
}'
```

CentOS 7ElasticsearchKibana

ElasticsearchJava Runtime EnvironmentJRE。 ElasticsearchJava 7Oracle JDK version 1.8.0_73

Oracle JDK version 1.8.0_73 。

Java。

```
# Install wget with yum
yum -y install wget

# Download the rpm jre-8u60-linux-x64.rpm for 64 bit
wget --no-cookies --no-check-certificate --header "Cookie:
gpw_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie"
"http://download.oracle.com/otn-pub/java/jdk/8u60-b27/jre-8u60-linux-x64.rpm"

# Download the rpm jre-8u101-linux-i586.rpm for 32 bit
wget --no-cookies --no-check-certificate --header "Cookie:
gpw_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie"
"http://download.oracle.com/otn-pub/java/jdk/8u101-b13/jre-8u101-linux-i586.rpm"

# Install jre-*.rpm
rpm -ivh jre-*.rpm
```

JavacentOS。

```
java -version
```

elasticsearch

```
# Download elasticsearch-2.3.5.rpm
wget
https://download.elastic.co/elasticsearch/release/org/elasticsearch/distribution/rpm/elasticsearch/2.3
2.3.5.rpm

# Install elasticsearch-*.rpm
rpm -ivh elasticsearch-*.rpm
```

elasticsearchsystemd

```
sudo systemctl daemon-reload
sudo systemctl enable elasticsearch
sudo systemctl start elasticsearch

# check the current status to ensure everything is okay.
systemctl status elasticsearch
```

Kibana

rpmGPG

```
sudo rpm --import http://packages.elastic.co/GPG-KEY-elasticsearch
```

kibana.repo

```
sudo vi /etc/yum.repos.d/kibana.repo
```

```
[kibana-4.4]
name=Kibana repository for 4.4.x packages
baseurl=http://packages.elastic.co/kibana/4.4/centos
gpgcheck=1
gpgkey=http://packages.elastic.co/GPG-KEY-elasticsearch
enabled=1
```

kibana

```
yum -y install kibana
```

```
systemctl start kibana
```

```
systemctl status kibana
```

o

```
systemctl enable kibana
```

Elasticsearch <https://riptutorial.com/zh-CN/elasticsearch/topic/941/elasticsearch>

2: Elasticsearch

Elasticsearch。

。

Elasticsearch config/elasticsearch.yml。

- cluster.name
 - ◦ ◦
 - elasticsearch。
- node.*
 - node.name
 - ◦ ◦
 - ◦
 - node.master
 - ◦ true。
 - true。
 - node.data
 - ◦ true。
 - true。
- path.*
 - path.data
 - ◦ /。
 - ./data。
 - dataElasticsearch config。
 - path.logs
 - ◦
 - ./logs。
- network.*
 - network.host
 - _local_ localhost。
 -
 - network.bind_host
 - Elasticsearch。
 - ◦
 - network.host。
 - network.publish_host
 - ◦
 - network.bind_host。
 - network.host`。
- discovery.zen.*
 - discovery.zen.minimum_master_nodes
 - ◦ (M / 2) + 1Mnode.master: true。
 -

- 1
 - `discovery.zen.ping.unicast.hosts`
 - ◦
 - ◦
 - `network.publish_host` ◦
 - `localhost` ◦

Elasticsearch

- ◦ ◦
- ◦ ◦
- ◦ ◦
-
- /
 - ◦
 - ◦ ◦ ◦
-
-
-

Elasticsearch◦

-

Elasticsearch 1.x2.xes.Javaes.

```
$ bin/elasticsearch -Des.cluster.name=my_cluster -Des.node.name=`hostname`
```

Elasticsearch 5.xJava -E-Des.

```
$ bin/elasticsearch -Ecluster.name=my_cluster -Enode.name=`hostname`
```

PuppetChefAnsible◦ ◦

- YAML
 -
 -
-

- 1.
- 2.
- 3.
4. YAML

◦

Examples

Elasticsearch

ElasticsearchYAMLYet Another Markup LanguageElasticsearch RPMDEB ◦

config/elasticsearch.yml

```
# Change the cluster name. All nodes in the same cluster must use the same name!
cluster.name: my_cluster_name

# Set the node's name using the hostname, which is an environment variable!
# This is a convenient way to uniquely set it per machine without having to make
# a unique configuration file per node.
node.name: ${HOSTNAME}

# ALL nodes should set this setting, regardless of node type
path.data: /path/to/store/data

# This is a both a master and data node (defaults)
node.master: true
node.data: true

# This tells Elasticsearch to bind all sockets to only be available
# at localhost (default)
network.host: _local_
```

_cluster/settings API_cluster/settings ◦

◦ ◦
◦ ◦ ◦

API◦ index.index.◦ indices.indices.◦

```
POST /_cluster/settings
{
  "persistent": {
    "cluster.routing.allocation.enable": "none"
  }
}
```

Elasticsearch 1.x2.x◦

Elasticsearch 5.xnull

```
POST /_cluster/settings
{
  "persistent": {
    "cluster.routing.allocation.enable": null
  }
}
```

◦
◦ ◦

_cluster/settings API_cluster/settings ◦

◦ ◦ ◦

API◦ index.index.◦ indices.indices.◦

```
POST /_cluster/settings
{
  "transient": {
    "cluster.routing.allocation.enable": "none"
  }
}
```

Elasticsearch 1.x2.x◦

Elasticsearch 5.xnull

```
POST /_cluster/settings
{
  "transient": {
    "cluster.routing.allocation.enable": null
  }
}
```

persistent◦

◦ index.index.◦ number_of_shardsnumber_of_replicas index.number_of_shards

index.number_of_replicas◦

◦ index.number_of_shards◦

```
PUT /my_index
{
  "settings": {
    "index.number_of_shards": 1,
    "index.number_of_replicas": 1
  }
}
```

.

```
PUT /my_index
{
  "settings": {
    "index": {
      "number_of_shards": 1,
      "number_of_replicas": 1
    }
  }
}
```

◦ index_settings◦ [slowlog](#)

```
PUT /my_index/_settings
{
```

```
"index": {
  "indexing.slowlog.threshold.index.warn": "1s",
  "search.slowlog.threshold": {
    "fetch.warn": "500ms",
    "query.warn": "2s"
  }
}
```

Elasticsearch 1.x2.x。 。 Elasticsearch 5.x。 。

Index Settings

```
PUT /*/_settings
{
  "index": {
    "indexing.slowlog.threshold.index.warn": "1s",
    "search.slowlog.threshold": {
      "fetch.warn": "500ms",
      "query.warn": "2s"
    }
  }
}
```

```
PUT /_all/_settings
{
  "index": {
    "indexing.slowlog.threshold.index.warn": "1s",
    "search.slowlog.threshold": {
      "fetch.warn": "500ms",
      "query.warn": "2s"
    }
  }
}
```

```
PUT /_settings
{
  "index": {
    "indexing.slowlog.threshold.index.warn": "1s",
    "search.slowlog.threshold": {
      "fetch.warn": "500ms",
      "query.warn": "2s"
    }
  }
}
```

```
PUT /logstash-*,my_other_index,some-other-*/_settings
{
  "index": {
    "indexing.slowlog.threshold.index.warn": "1s",
    "search.slowlog.threshold": {
      "fetch.warn": "500ms",
      "query.warn": "2s"
    }
  }
}
```


3: Python

	hostport° host'localhost' port9200.[{"host": "ip of es server", "port": 9200}]
sniff_on_start	Booleanelasticsearch
sniff_on_connection_fail	
sniffer_timeout	
sniff_timeout	
retry_on_timeout	Booelanelasticsearch
http_auth	username:passwordhttp

Examples

Python

```
$ pip install elasticsearch
```

ElasticsearchElasticsearch“”。

```
from datetime import datetime
from elasticsearch import Elasticsearch

# Connect to Elasticsearch using default options (localhost:9200)
es = Elasticsearch()

# Define a simple Dictionary object that we'll index to make a document in ES
doc = {
    'author': 'kimchy',
    'text': 'Elasticsearch: cool. bonsai cool.',
    'timestamp': datetime.now(),
}

# Write a document
res = es.index(index="test-index", doc_type='tweet', id=1, body=doc)
print(res['created'])

# Fetch the document
res = es.get(index="test-index", doc_type='tweet', id=1)
print(res['_source'])

# Refresh the specified index (or indices) to guarantee that the document
# is searchable (avoid race conditions with near realtime search)
es.indices.refresh(index="test-index")

# Search for the document
```

```

res = es.search(index="test-index", body={"query": {"match_all": {}}})
print("Got %d Hits:" % res['hits']['total'])

# Show each "hit" or search response (max of 10 by default)
for hit in res['hits']['hits']:
    print("%(timestamp)s %(author)s: %(text)s" % hit["_source"])

```

```

es = Elasticsearch(hosts=hosts, sniff_on_start=True, sniff_on_connection_fail=True,
sniffer_timeout=60, sniff_timeout=10, retry_on_timeout=True)

```

◦

ElasticSearch◦ indexbodyindex◦

```

from elasticsearch import Elasticsearch

# create an Elasticsearch instance
es = Elasticsearch()
# name the index
index_name = "my_index"
# define the mapping
mapping = {
    "mappings": {
        "my_type": {
            "properties": {
                "foo": {'type': 'text'},
                "bar": {'type': 'keyword'}
            }
        }
    }
}

# create an empty index with the defined mapping - no documents added
if not es.indices.exists(index_name):
    res = es.indices.create(
        index=index_name,
        body=mapping
    )
    # check the response of the request
    print(res)
    # check the result of the mapping on the index
    print(es.indices.get_mapping(index_name))

```

id doc_idname “John”◦ ◦

```

doc = {
    "doc": {
        "name": "John"
    }
}
es.update(index='index_name',
          doc_type='doc_name',
          id='doc_id',
          body=doc)

```

name “John”◦


```
q = {
  "script": {
    "inline": "ctx._source.age=23",
    "lang": "painless"
  },
  "query": {
    "match": {
      "name": "John"
    }
  }
}

es.update_by_query (body=q,
                    doc_type='doc_name',
                    index='index_name')
```

Python <https://riptutorial.com/zh-CN/elasticsearch/topic/2068/python>

4:

o o

Examples

Elasticsearch	
/	

elasticsearch.

```

create database test;

use test;

create table product;

create table product (name varchar, id int PRIMARY KEY);

insert into product (id,name) VALUES (1,'Shirt');

insert into product (id,name) VALUES (2,'Red Shirt');

select * from product;

name      | id
-----+-----
Shirt     | 1
Red Shirt | 2

```

Elasticsearch

```

POST test/product
{
  "id" : 1,
  "name" : "Shirt"
}

POST test/product
{
  "id" : 2,
  "name" : "Red Shirt"
}

GET test/product/_search

"hits": [
  {
    =====

```

```

    "_index": "test",           ===> index   |
    "_type": "product",       ===>type    |
    "_id": "AVzglFomaus3G2tXc6sB", |
    "_score": 1,              |
    "_source": {               |====> document
      "id": 2,                 ===>field    |
      "name": "Red Shirt"     ===>field    |
    }                           |
  },                             =====
  {
    "_index": "test",
    "_type": "product",
    "_id": "AVzglD12aus3G2tXc6sA",
    "_score": 1,
    "_source": {
      "id": 1,
      "name": "Shirt"
    }
  }
]

```

- . . . Elasticsearch.

◦

shirtsred ◦ *redshirts* ◦ ◦

```
select * from product where name like '%Red%' or name like '%Shirt%';
```

name	id
Shirt	1
Red Shirt	2

Elasticsearch

```

POST test/product/_search
{
  "query": {
    "match": {
      "name": "Red Shirt"
    }
  }
}

```

```

"hits": [
  {
    "_index": "test",
    "_type": "product",
    "_id": "AVzglFomaus3G2tXc6sB",
    "_score": 1.2422675,           ===> Notice this
    "_source": {
      "id": 2,
      "name": "Red Shirt"
    }
  },

```

```

{
  "_index": "test",
  "_type": "product",
  "_id": "AVzglD12aus3G2tXc6sA",
  "_score": 0.25427115,      ==> Notice this
  "_source": {
    "id": 1,
    "name": "Shirt"
  }
}
]

```

Relational Database Elasticsearch_score

- • • Elasticsearch

-

shirts shrt ◦

```
select * from product where name like '%shrt%';
```

No results found

Elasticsearch

```
POST /test/product/_search
```

```

{
  "query": {
    "match": {
      "name": {
        "query": "shrt",
        "fuzziness": 2,
        "prefix_length": 0
      }
    }
  }
}

```

```

"hits": [
  {
    "_index": "test",
    "_type": "product",
    "_id": "AVzglD12aus3G2tXc6sA",
    "_score": 1,
    "_source": {
      "id": 1,
      "name": "Shirt"
    }
  },
  {
    "_index": "test",
    "_type": "product",

```

```
  "_id": "AVzglFomaus3G2tXc6sB",
  "_score": 0.8784157,
  "_source": {
    "id": 2,
    "name": "Red Shirt"
  }
}
```

Elasticsearch^{fuzzy}

<https://riptutorial.com/zh-CN/elasticsearch/topic/10632/>

5:

- - CharFilters
 - Tokenizer
 - TokenFilters
- Analyzer ◦ ◦

Examples

“analyzer”“”。 “index”“not_analyzed”

```
PUT my_index
{
  "mappings": {
    "user": {
      "properties": {
        "name": {
          "type": "string"
          "analyzer": "my_user_name_analyzer"
        },
        "id": {
          "type": "string",
          "index": "not_analyzed"
        }
      }
    }
  }
}
```

◦ ◦

```
PUT my_index
{
  "mappings": {
    "user": {
      "properties": {
        "name": {
          "type": "string"
          "analyzer": "standard",
          "fields": {
            "special": {
              "type": "string",
              "analyzer": "my_user_name_analyzer"
            },
            "unanalyzed": {
              "type": "string",
              "index": "not_analyzed"
            }
          }
        }
      }
    }
  }
}
```

```
}
}
}
}
```

“user.name.special”“user.name.unanalyzed”“user.name”standard Analyzer。

。


-
-
-

。

1. elasticsearchdr => Doctordoctorrelasticsearchdr。

2. 

3. “”。


resurrectionressurrection

Elasticsearch

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.

。

```
"settings": {
  "analysis": {
    "analyzer": {
      "case_insensitive": {
        "tokenizer": "keyword",
        "filter": ["lowercase"]
      }
    }
  }
}
```

```
}
```

<https://riptutorial.com/zh-CN/elasticsearch/topic/6232/>

6:

- `curl -X <VERB>'<PROTOCOL>// <HOST><PORT> / <PATH><QUERY_STRING>'-d'<BODY>'`
-
- VERBHTTPGETPOSTPUTHEADDELETE
- httphttpsElasticsearchhttps。
- HOSTElasticsearchlocalhost。
- PORTElasticsearch HTTP9200。
- API_count。 _cluster / stats_nodes / stats / jvm
- QUERY_STRINGprettyJSON。
- BODYJSON。
- [ElasticsearchElasticsearch Docs](#)

Examples

Curl

```
curl -XGET 'http://www.example.com:9200/myIndexName/_count?pretty'
```

```
{
  "count" : 90,
  "_shards" : {
    "total" : 6,
    "successful" : 6,
    "failed" : 0
  }
}
```

90。

Id

```
curl -XGET 'http://www.example.com:9200/myIndexName/myTypeName/1'
```

```
{
  "_index" : "myIndexName",
  "_type" : "myTypeName",
  "_id" : "1",
```

```
  "_version" : 1,
  "found": true,
  "_source" : {
    "user" : "mrunal",
    "postDate" : "2016-07-25T15:48:12",
    "message" : "This is test document!"
  }
}
```

```
curl -XPUT 'www.example.com:9200/myIndexName?pretty'
```

```
{
  "acknowledged" : true
}
```

```
curl 'www.example.com:9200/_cat/indices?v'
```

health	status	index	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
green	open	logstash-2016.07.21	5	1	4760	0	4.8mb	2.4mb
green	open	logstash-2016.07.20	5	1	7232	0	7.5mb	3.7mb
green	open	logstash-2016.07.22	5	1	93528	0	103.6mb	52mb
green	open	logstash-2016.07.25	5	1	20683	0	41.5mb	21.1mb

```
curl -XDELETE 'http://www.example.com:9200/myIndexName?pretty'
```

```
{
  "acknowledged" : true
}
```

```
curl -XGET http://www.example.com:9200/myIndexName/_search?pretty=true&q=*:*
```

Search **API**_{myIndexName}◦

<https://riptutorial.com/zh-CN/elasticsearch/topic/3703/>

7:

type SQL index SQL type

Elasticsearch _type my_type my_other_type " _type": "my_type";

o o o o o

o o o

-
-
-
-
-
-
-
- rangerover
-
- ...

o

```
GET /cars/_search
```

```
GET /cars/bmw/_search
```

Elasticsearch

```
GET /cars/_search
{
  "query": {
    "bool": {
      "filter": [
        {
          "term": {
            "_type": "bmw"
          }
        }
      ]
    }
  }
}
```

bmw_type _type

manufacturer make 1/n

Elasticsearch 1.x2.x

```
PUT /cars
```

```

{
  "manufacturer": { <1>
    "properties": {
      "make": { <2>
        "type": "string",
        "index": "not_analyzed"
      }
    }
  }
}

```

1. ◦
2. ◦

Elasticsearch 5.x

```

PUT /cars
{
  "manufacturer": { <1>
    "properties": {
      "make": { <2>
        "type": "keyword"
      }
    }
  }
}

```

1. ◦
2. ◦

◦ ◦

•

◦

◦ - ◦

◦

•

Lucene ◦ Elasticsearch ◦

•

◦ ES 2.x ◦ Lucene ◦

ES 1.x ◦ ◦

• /◦

◦ ◦

◦ ◦

•

- ElasticsearchMarvel1.x2.xX-Pack5.x +Elasticsearch◦ 5 n1◦

Examples

HTTPcURLHTTP◦ [SenseKibana 5.0Console](#)◦

<#>◦

```
PUT /my_index <1>
{
  "mappings": {
    "my_type": { <2>
      "properties": {
        "field1": {
          "type": "long"
        },
        "field2": {
          "type": "integer"
        },
        "object1": {
          "type": "object",
          "properties": {
            "field1" : {
              "type": "float"
            }
          }
        }
      }
    }
  },
  "my_other_type": {
    "properties": {
      "field1": {
        "type": "long" <3>
      },
      "field3": { <4>
        "type": "double"
      }
    }
  }
}
```

1. index◦
2. type ◦
3. indextype s ES 1.x◦ ES 2.x◦
4. type s◦

◦ /◦ 1515◦

◦

HTTPcURLHTTP◦ [SenseKibana 5.0Console](#)◦

<#>◦

```
DELETE /my_index <1>

PUT /my_index/my_type/abc123 <2>
{
  "field1" : 1234, <3>
  "field2" : 456,
  "object1" : {
    "field1" : 7.8 <4>
  }
}
```

1. ◦
2. my_index my_type IDabc123 ◦
 - ◦ ◦
3. ◦ **Elasticsearch** long integers short 1234456 ◦
4. ◦ double float ◦

◦ <3><4>◦

```
PUT /my_index/my_other_type/abc123 <1>
{
  "field1": 91, <2>
  "field3": 4.567
}
```

1. ◦ IDabc123◦
2. field1◦ "field1": "this is some text" "field1": 123.0 ◦

my_index my_other_type◦

Elasticsearch◦

```
GET /my_index/_mappings <1>
{
  "mappings": {
    "my_type": { <2>
      "properties": {
        "field1": {
          "type": "long"
        },
        "field2": {
          "type": "long" <3>
        },
        "object1": {
          "type": "object",
          "properties": {
            "field1" : {
              "type": "double" <4>
            }
          }
        }
      }
    }
  }
}
```

```
    }
  },
  "my_other_type": { <5>
    "properties": {
      "field1": {
        "type": "long"
      },
      "field3": {
        "type": "double"
      }
    }
  }
}
```

1. `_mappings` ◦
2. `my_type` ◦
3. `field2longinteger` ◦ ◦
4. `object1.field1double` **33** ◦
 - `long` ◦ `double` ◦
5. `my_other_type` ◦ `longdouble` ◦
 - `field1 my_type` ◦
 - `field3` ◦

<https://riptutorial.com/zh-CN/elasticsearch/topic/3412/>

8: API

API。

Examples

/。

```
curl -XPOST 'localhost:9200/twitter/tweet?routing=kimchy&pretty' -d'
{
  "user" : "kimchy",
  "postDate" : "2009-11-15T14:12:12",
  "message" : "trying out Elasticsearch"
}'
```

DSL。elasticsearch。DSL。

```
GET /my_index/type/_search
{
  "query" : {
    "term" : { "field_to_search" : "search_item" }
  }
}
```

multi_search。

```
GET /_search
{
  "query": {
    "multi_match" : {
      "query": "text to search",
      "fields": [ "field_1", "field_2" ]
    }
  }
}
```

boost^{*}

```
GET /_search
{
  "query": {
    "multi_match" : {
      "query": "text to search",
      "fields": [ "field_1^2", "field_2*" ]
    }
  }
}
```

URI

URI。 ""。

```
GET Index/type/_search?q=field:value
```

。

```
GET /_search
{
  "query" : {
    "match": { "field": "value" }
  },
  "highlight" : {
    "fields" : {
      "content" : {}
    }
  }
}
```

API <https://riptutorial.com/zh-CN/elasticsearch/topic/8625/api>

9: kibanaElasticsearch

Kibana ◦ kibana ◦ kibana ◦ localhostkibana [https:// localhost5601](https://localhost:5601) kibana ◦

Examples

Kibana

```
<REST Verb> /<Index>/<Type>/<ID>
```

Kibana Console `elasticsearch` ◦

-

```
GET /_cat/health?v
```

-

```
GET /_cat/indices?v
```

- **car**

```
PUT /car?pretty
```

- **id 1 car**

```
PUT /car/external/1?pretty
{
  "name": "Tata Nexon"
}
```

```
{
  "_index": "car",
  "_type": "external",
  "_id": "1",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "created": true
}
```

-

```
GET /car/external/1?pretty
```

-

```
DELETE /car?pretty
```

Elasticsearch

- `/car/external/1`

```
PUT /car/external/1?pretty
{
  "name": "Tata Nexa"
}
```

“Tata Nexon”id 1“Tata Nexa”

- `Id`

```
POST /car/external?pretty
{
  "name": "Jane Doe"
}
```

IdPOSTPUT。IdelasticsearchID。

- `Id`

```
POST /car/external/1/_update?pretty
{
  "doc": { "name": "Tata Nex" }
}
```

-

```
POST /car/external/1/_update?pretty
{
  "doc": { "name": "Tata Nexon", "price": 1000000 }
}
```

-

```
POST /car/external/1/_update?pretty
{
  "script" : "ctx._source.price += 50000"
}
```

ctx._source

-

```
DELETE /car/external/1?pretty
```

Delete by Query API

elasticsearch_bulk API。

- **_bulk API**

```
POST /car/external/_bulk?pretty
{"index":{"_id":"1"}}
{"name": "Tata Nexon" }
{"index":{"_id":"2"}}
{"name": "Tata Nano" }
```

- **_bulk API**

```
POST /car/external/_bulk?pretty
{"update":{"_id":"1"}}
{"doc": { "name": "Tata Nano" } }
{"delete":{"_id":"2"}}
```

API。。

[kibanaElasticsearch](https://riptutorial.com/zh-CN/elasticsearch/topic/10058/kibanaelasticsearch) <https://riptutorial.com/zh-CN/elasticsearch/topic/10058/kibanaelasticsearch>

10:

Cluster Health `15 14`

Elasticsearch `15 14`

status

-
-
-

--

1.

- -
 -
- initializing_shards
- -

2.

- -
- initializing_shards
- -

3.

- relocating_shards
- - Elasticsearch 5.x

Examples

Cluster Health

HTTP `<#>`

`_cat` API

```
GET /_cat/health?v <1>
```

1. `?v`

Elasticsearch 1.x `_cat/health` Elasticsearch 5.x

```

epoch      timestamp cluster      status node.total node.data shards pri relo init unassign
pending_tasks max_task_wait_time active_shards_percent
1469302011 15:26:51 elasticsearch yellow      1      1      45 45  0  0      44
0          -          50.6%

```

HTTP。 <#> 。

_cat API。

```
GET /_cat/health <1>
```

Elasticsearch 1.x `_cat/health` Elasticsearch 5.x

```
1469302245 15:30:45 elasticsearch yellow 1 1 45 45 0 0 44 0 - 50.6%
```

HTTP。 <#> 。

Elasticsearch `_cat API` API。

```
GET /_cat/health?help <1>
```

1. `?help` API。

Elasticsearch 1.x `_cat/health` Elasticsearch 5.x

epoch	t,time	seconds since 1970-01-01
00:00:00		
timestamp	ts,hms,hmmss	time in HH:MM:SS
cluster	cl	cluster name
status	st	health status
node.total	nt,nodeTotal	total number of nodes
node.data	nd,nodeData	number of nodes that can
store data		
shards	t,sh,shards.total,shardsTotal	total number of shards
pri	p,shards.primary,shardsPrimary	number of primary shards
relo	r,shards.relocating,shardsRelocating	number of relocating nodes
init	i,shards.initializing,shardsInitializing	number of initializing
nodes		
unassign	u,shards.unassigned,shardsUnassigned	number of unassigned shards
pending_tasks	pt,pendingTasks	number of pending tasks
max_task_wait_time	mtwt,maxTaskWaitTime	wait time of longest task
pending		
active_shards_percent	asp,activeShardsPercent	active number of shards in
percent		

```
GET /_cat/health?h=timestamp,cl,status&v <1>
```

1. h=...◦
2. v ◦

Elasticsearch 5.x

```
timestamp cl          status
15:38:00  elasticsearch  yellow
```

JSON

HTTP◦ <#>◦

_cat API◦ ◦ JSON API◦

```
GET /_cluster/health
```

Elasticsearch 1.x _cluster/healthElasticsearch 5.x

```
{
  "cluster_name": "elasticsearch",
  "status": "yellow",
  "timed_out": false,
  "number_of_nodes": 1,
  "number_of_data_nodes": 1,
  "active_primary_shards": 45,
  "active_shards": 45,
  "relocating_shards": 0,
  "initializing_shards": 0,
  "unassigned_shards": 44,
  "delayed_unassigned_shards": 0,
  "number_of_pending_tasks": 0,
  "number_of_in_flight_fetch": 0,
  "task_max_waiting_in_queue_millis": 0,
  "active_shards_percent_as_number": 50.56179775280899
}
```

<https://riptutorial.com/zh-CN/elasticsearch/topic/2069/>

11:

- `""{ - "<aggregation_name>"{ - "<aggregation_type>"{ - <aggregation_body> - } - ["meta" [{"meta_data_body}]] - [{"["<sub_aggregation> +]] - } - ["<aggregation_name_2>"{...}] * - }`

Examples

◦

```
POST /index/_search?
{
  "aggs" : {
    "avd_value" : { "avg" : { "field" : "name_of_field" } }
  }
}
```

◦ avg◦

```
{
  ...
  "aggregations": {
    "avg_value": {
      "value": 75.0
    }
  }
}
```

avg_grade◦

◦ ◦

```
POST /index/_search?size=0
{
  "aggs" : {
    "type_count" : {
      "cardinality" : {
        "field" : "type"
      }
    }
  }
}
```

```
{
  ...
  "aggregations" : {
    "type_count" : {
      "value" : 3
    }
  }
}
```


◦ ◦

extended_statssum_of_squaresvariancestd_deviationstd_deviation_bounds◦

```
{
  "aggs" : {
    "stats_values" : { "extended_stats" : { "field" : "field_name" } }
  }
}
```

```
{
  ...

  "aggregations": {
    "stats_values": {
      "count": 9,
      "min": 72,
      "max": 99,
      "avg": 86,
      "sum": 774,
      "sum_of_squares": 67028,
      "variance": 51.55555555555556,
      "std_deviation": 7.180219742846005,
      "std_deviation_bounds": {
        "upper": 100.36043948569201,
        "lower": 71.63956051430799
      }
    }
  }
}
```

<https://riptutorial.com/zh-CN/elasticsearch/topic/10745/>

S. No		Contributors
1	Elasticsearch	Ahsanul Haque , Berto , Community , DJanssens , Dulguun , igo , KartikKannapur , manishrw , mightyteja , noscreenname , Onur , rafa.ferreira , RustyBuckets , sarvajeetsuman , SeinopSys , Shivkumar Mallesappa , Stephan-v , Sahas K , Sumit Kumar , Trilarion
2	Elasticsearch	pickypg
3	Python	aidan.plenert.macdonald , christinabo , KartikKannapur , pickypg , Sumit Kumar
4		Richa
5		Bhushan Gadekar , Sid1199 , Thomas
6		Fawix , Mrunal Pagnis , Mrunal Pagnis
7		pickypg
8	API	aerokite , Sid1199
9	kibanaElasticsearch	sarvajeetsuman
10		Gerardo Rochín , pickypg
11		Sid1199