搜索API：search API即\_search

空搜索：在所有索引的所有文档中搜索，并返回前10个搜索结果

api/\_search

指定timeout：

api/\_search?timeout=12ms，要带有单位

/\_search

在所有索引的所有类型中搜索

/gb/\_search

在索引gb的所有类型中搜索

/gb,us/\_search

在索引gb和us的所有类型中搜索

/g\*,u\*/\_search

在以g或u开头的索引的所有类型中搜索

/gb/user/\_search

在索引gb的类型user中搜索

/gb,us/user,tweet/\_search

在索引gb和us的类型为user和tweet中搜索

/\_all/user,tweet/\_search

在所有索引的user和tweet中搜索

获取分页数据：

size: 结果数，默认10

from: 跳过开始的结果数，默认0

如果你想每页显示5个结果，页码从1到3，那请求如下：

GET /\_search?size=5

GET /\_search?size=5&from=5

GET /\_search?size=5&from=10

所有类型为tweet并在tweet字段中包含elasticsearch字符的文档：

GET /\_all/tweet/\_search?q=tweet:elasticsearch

查询所有索引所有类型下字段BusinessObjectId包含123的文档

<http://192.168.100.249:9200/_search?q=BusinessObjectId:123>

下一个语句查找name字段中包含"john"和tweet字段包含"mary"的结果。实际的查询只需要：

+name:john +tweet:mary

<http://192.168.100.249:9200/_search?q=+Channel:0+Title:c>

查询所有索引所有类型下字段Channel包含0和Title包含c（不区分大小写）的文档

<http://192.168.100.249:9200/_search?q=-Channel:0+Title:c>

查询所有索引所有类型下字段Channel不包含0和Title包含c的文档

GET /\_search?q=mary

返回所有索引所有类型的所有字段只要包含mary的文档

http://192.168.100.249:9200/\_search?q=两会&from=10&size=1

查询所有包含两会的文档并返回第11个文档，如果索引超过总数则返回空json数组

name字段包含"mary"或"john",或表示用空格隔开

date晚于2014-09-10

\_all字段包含"aggregations"或"geo"

+name:(mary john) +date:>2014-09-10 +(aggregations geo)

编码后的查询字符串变得不太容易阅读：

?q=%2Bname%3A(mary+john)+%2Bdate%3A%3E2014-09-10+%2B(aggregations+geo)

[http://192.168.100.249:9200/\_search?q=BusinessObjectId:(1 2)](http://192.168.100.249:9200/_search?q=BusinessObjectId:(1%202))，空格隔开表示或

查询BusinessObjectId字段包含1或者2的文档

请求体查询：上面为url，下面为请求体

空查询：返回所有文档前10个

GET /\_search

{}

同字符串查询一样，你可以查询一个，多个或\_all索引(indices)或类型(types)：

GET /index\_2014\*/type1,type2/\_search

{}

查询所有索引以index\_2014开头的类型为type1或者type2的所有文档

你可以使用from及size参数进行分页：

GET /\_search

{

"from": 30,

"size": 10

}

结构化查询：使用query字段

空查询：match-\_all的值为空对象

GET /\_search

{

"query": {

"match\_all": {}

}

}

Match匹配查询条件

GET /\_search

{

"query": { //query是规定写法

"match": {

"tweet": "elasticsearch"

}

}

}

查询所有索引所有类型的tweet字段包含elasticsearch的文档

复合语句查询

{

"bool": {

"must": { "match": { "tweet": "elasticsearch" }},

"must\_not": { "match": { "name": "mary" }},

"should": { "match": { "tweet": "full text" }}

}

}

查询过滤语句

Term过滤：精确匹配

{ "term": { "age": 26 }}

{ "term": { "date": "2014-09-01" }}

{ "term": { "public": true }}

{ "term": { "tag": "full\_text" }}

Terms过滤：可以匹配多个值

{

"terms": {

"tag": [ "search", "full\_text", "nosql" ]

}

}

## range 过滤：指定数据范围来过滤

{

"range": {

"age": {

"gte": 20,

"lt": 30

}

}

}

范围操作符包含：

gt :: 大于

gte:: 大于等于

lt :: 小于

lte:: 小于等于

## exists 和 missing 过滤：是否包含字段

{

"exists": {

"field": "title"

}

}

## bool 过滤：相关查询关键词must 、 must\_not 、 should

{

"bool": {

"must": { "term": { "folder": "inbox" }},

"must\_not": { "term": { "tag": "spam" }},

"should": [

{ "term": { "starred": true }},

{ "term": { "unread": true }}

]

}

}

## match\_all 查询

使用match\_all 可以查询到所有文档，是没有查询条件下的默认语句。

{

"match\_all": {}

}

## Match查询

{ "match": { "age": 26 }}

{ "match": { "date": "2014-09-01" }}

## multi\_match 查询

{

"multi\_match": {

"query": "full text search",

"fields": [ "title", "body" ]

}

}

## bool 查询