增加一条记录

PUT /case/zhangyuan/1?pretty

{

"name": "John Doe"

}

验证查询

GET /case/zhangyuan/1

替换同一ID

PUT case/zhangyuan/1?pretty

{

"name": "Jane Doe"

}

系统随机生成ID

POST case/zhangyuan?pretty

{

"name": "Jane Doe"

}

Updating Documents[edit](https://github.com/elastic/elasticsearch/edit/6.3/docs/reference/getting-started.asciidoc" \o "Edit this page on GitHub)

In addition to being able to index and replace documents, we can also update documents. Note though that Elasticsearch does not actually do in-place updates under the hood. Whenever we do an update, Elasticsearch deletes the old document and then indexes a new document with the update applied to it in one shot.

POST /case/zhangyuan/1/\_update?pretty

{

"doc": { "name": "Jane Doe zhangyuan" }

}

增加一个字段

POST /case/zhangyuan/1/\_update?pretty

{

"doc": { "name": "Jane Doe", "age": 20 }

}

结果：

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "1",

"\_version": 3,

"found": true,

"\_source": {

"name": "Jane Doe",

"age": 20

}

}

Updates can also be performed by using simple scripts. This example uses a script to increment the age by 5:

POST /case/zhangyuan/1/\_update?pretty

{

"script" : "ctx.\_source.age += 5"

}

In the above example, ctx.\_source refers to the current source document that is about to be updated.

<https://www.elastic.co/guide/en/elasticsearch/reference/6.3/docs-update.html>

API DOC模式 增加一个字段：

POST case/zhangyuan/1/\_update

{

"doc" : {

"sex" : "M"

}

}

Script 模式 增加字段

POST case/zhangyuan/1/\_update

{

"script" : "ctx.\_source.project = 'tour'"

}

删除字段

POST case/zhangyuan/1/\_update

{

"script" : "ctx.\_source.remove('project')"

}

增加数组字段

POST case/zhangyuan/1/\_update

{

"doc" : {

"project" :[ "tour"]

}

}

数组里增加元素 （如果不存在project属性，会创建）

POST case/zhangyuan/1/\_update

{

"script" : {

"source": "ctx.\_source.project.add(params.channel)",

"lang": "painless",

"params" : {

"channel" : "med"

}

}

}

**删除数组中的元素**

POST case/zhangyuan/1/\_update

{

"script" : {

"source": "ctx.\_source.project.remove(ctx.\_source.project.indexOf(params.channel))",

"lang": "painless",

"params" : {

"channel" : "med"

}

}

}

**删除一条记录（根据数组下标）**

POST case/zhangyuan/1/\_update

{

"script" : {

"source": "ctx.\_source.project.remove(1)",

"lang": "painless"

}

}

**新增或修改 saveOrUpdate**

If the document does not already exist, the contents of the upsert element will be inserted as a new document. If the document does exist, then the script will be executed instead:

POST case/zhangyuan/2/\_update

{

"script" : {

"source": "ctx.\_source.age += params.count",

"lang": "painless",

"params" : {

"count" : 4

}

},

"upsert" : {

"age" : 20,

"name": "ebaosvc"

}

}

**运行二遍上面的语句**。结果如下：

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "2",

"\_version": 2,

"found": true,

"\_source": {

**"age": 24,**

"name": "ebaosvc"

}

}

#### doc\_as\_upsert

Instead of sending a partial doc plus an upsert doc, setting doc\_as\_upsert to true will use the contents of doc as the upsert value:

POST case/zhangyuan/2/\_update

{

"doc" : {

"name" : "ebao-svc"

},

"doc\_as\_upsert" : true

}

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "2",

"\_version": 3,

"found": true,

"\_source": {

"age": 24,

"name": "ebao-svc"

}

}

没有id = 3的记录。

POST case/zhangyuan/3/\_update

{

"doc" : {

"name" : "ebao-svc"

},

"doc\_as\_upsert" : true

}

GET /case/zhangyuan/3

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "3",

"\_version": 1,

"found": true,

"\_source": {

"name": "ebao-svc"

}

}

小知识点noop 如果没有改动。

POST case/zhangyuan/3/\_update

{

"doc" : {

"name" : "ebao-svc"

}

}

返回值

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "3",

"\_version": 2,

"result": "noop",

"\_shards": {

"total": 0,

"successful": 0,

"failed": 0

}

}

## 查询统计

### 过滤结果集 filter

查询

GET case/zhangyuan/\_search

{

"query" : {

"match\_all": {}

},

"script\_fields" : {

"test1" : {

"script" : "params['\_source']['name']"

}

}

}

知识点：

 \_source keyword here to navigate the json-like model

It’s important to understand the difference between doc['my\_field'].value and params['\_source']['my\_field']. The first, using the doc keyword, will cause the terms for that field to be loaded to memory (cached), which will result in faster execution, but more memory consumption. However, using doc is still the recommended way to access values from the document, if at all possible, because \_source must be loaded and parsed every time it’s used. Using \_source is very slow.

返回值

{

"took": 67,

"timed\_out": false,

"\_shards": {

"total": 5,

"successful": 5,

"skipped": 0,

"failed": 0

},

"hits": {

"total": 3,

"max\_score": 1,

"hits": [

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "2",

"\_score": 1,

"fields": {

"test1": [

"ebao-svc"

]

}

},

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "1",

"\_score": 1,

"fields": {

"test1": [

"Jane Doe"

]

}

},

Script field

Allows to return a [script evaluation](https://www.elastic.co/guide/en/elasticsearch/reference/6.3/modules-scripting.html" \o "Scripting) (based on different fields) for each hit

GET case/zhangyuan/\_search

{

"query" : {

"match\_all": {}

},

"script\_fields" : {

"test1" : {

"script" : {

"lang": "painless",

"source": "doc['age'].value \* 2"

}

},

"test2" : {

"script" : {

"lang": "painless",

"source": "doc['age'].value \* params.factor",

"params" : {

"factor" : 2.0

}

}

}

}

}

结果：

{

"took": 67,

"timed\_out": false,

"\_shards": {

"total": 5,

"successful": 5,

"skipped": 0,

"failed": 0

},

"hits": {

"total": 3,

"max\_score": 1,

"hits": [

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "2",

"\_score": 1,

"fields": {

"test1": [

48

],

"test2": [

48

]

}

},

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "1",

"\_score": 1,

"fields": {

"test1": [

50

],

"test2": [

50

]

}

},

### 过滤结果集 Fields （未通）

Allows to selectively load specific stored fields for each document represented by a search hit.

The stored\_fields parameter is about fields that are explicitly marked as stored in the mapping, which is off by default and generally not recommended. Use [source filtering](https://www.elastic.co/guide/en/elasticsearch/reference/6.3/search-request-source-filtering.html" \o "Source filtering) instead to select subsets of the original source document to be returned.

GET case/zhangyuan/\_search

{

"stored\_fields" : ["name","age"],

"query" : {

"match\_all": {}

}

}

结果：

{

"took": 0,

"timed\_out": false,

"\_shards": {

"total": 5,

"successful": 5,

"skipped": 0,

"failed": 0

},

"hits": {

"total": 3,

"max\_score": 1,

"hits": [

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "2",

"\_score": 1

},

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "1",

"\_score": 1

},

{

"\_index": "case",

"\_type": "zhangyuan",

"\_id": "3",

"\_score": 1

}

]

}

}

**删除一条记录（根据特定条件）**

POST case/zhangyuan/1/\_update

{

"script" : {

"source": "if (ctx.\_source.project.contains(params.tag)) { ctx.op = 'delete' } else { ctx.op = 'none' }",

"lang": "painless",

"params" : {

"tag" : "AA"

}

}

}