添加商品

```
PUT /ecommerce/product/1
{
    "name" : "gaolujie yagao",
    "desc" :  "gaoxiao meibai",
    "price" :  30,
    "producer" :      "gaolujie producer",
    "tags": [ "meibai", "fangzhu" ]
}

PUT /ecommerce/product/2
{
    "name" : "jiajieshi yagao",
    "desc" :  "youxiao fangzhu",
    "price" :  25,
    "producer" :      "jiajieshi producer",
    "tags": [ "fangzhu" ]
}

PUT /ecommerce/product/3
{
    "name" : "zhonghua yagao",
    "desc" :  "caoben zhiwu",
    "price" :  40,
    "producer" :      "zhonghua producer",
    "tags": [ "qingxin" ]
}

PUT /ecommerce/product/4
{
    "name" : "special yagao",
    "desc" :  "special meibai",
    "price" :  44,
    "producer" : "special producer",
    "tags": [ "meibai", "fangzhu" ]
}
```

修改索引

```
PUT /ecommerce/product/1
{
    "name" : "jiaqiangban gaolujie yagao",
    "desc" :  "gaoxiao meibai",
    "price" :  30,
    "producer" :      "gaolujie producer",
    "tags": [ "meibai", "fangzhu" ]
}
```

返回响应:

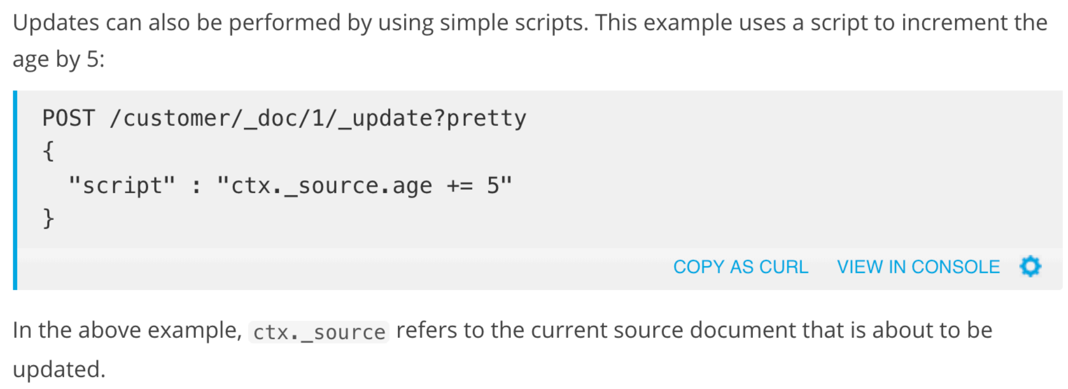
```
{
  "_index": "ecommerce",
  "_type": "product",
  "_id": "1",
  "_version": 2,
  "result": "updated",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "created": false
}
```

需要将所有的字段信息field填入去完成修改

单独修改无需所有字段:

将产品价格改为24,并添加新的字段

```
POST /ecommerce/product/1/_update
{
  "doc": {
   "price": 24,
      "flag":"true"
 }
}
```



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

```
POST /customer/_doc/1/_update?pretty
{
  "script" : "ctx._source.age += 5"
}
```

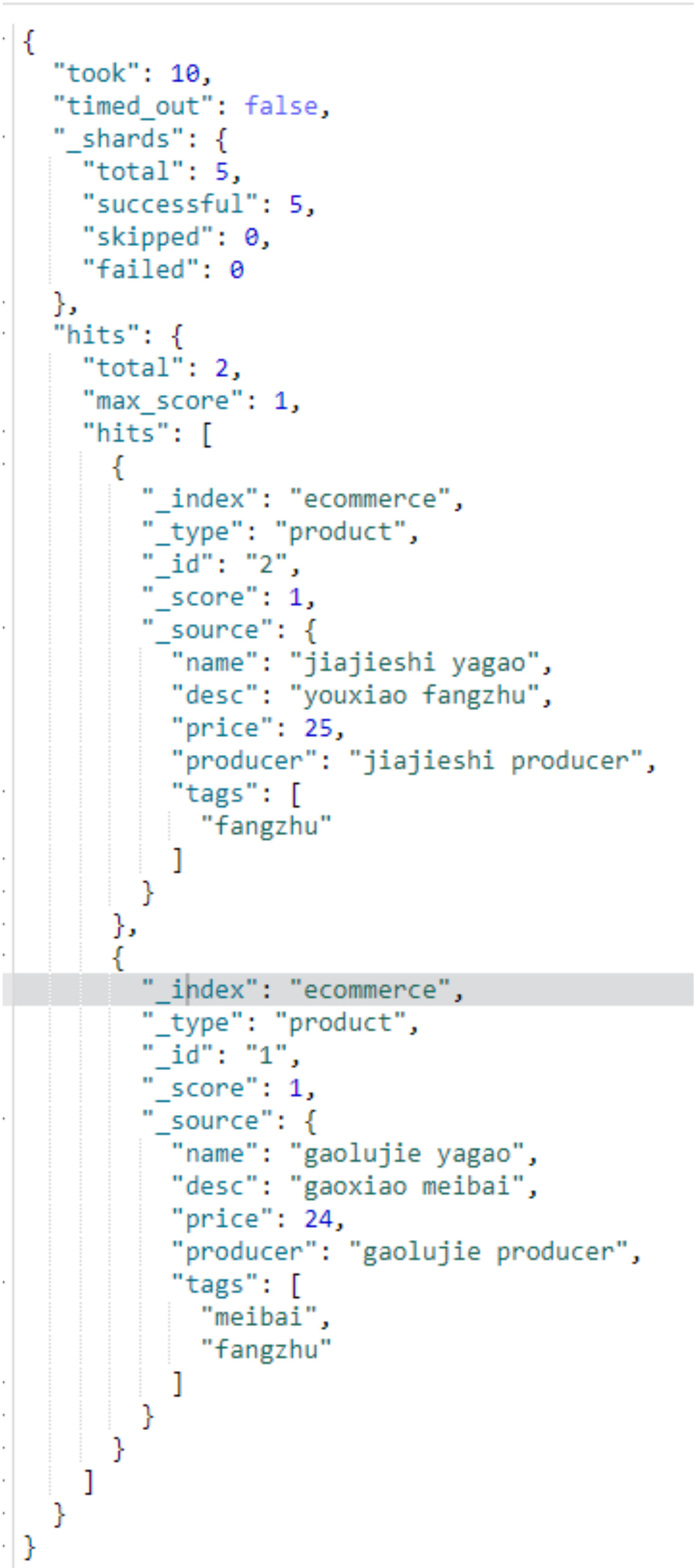
In the above example, `ctx._source` refers to the current source document that is about to be updated.

查询索引信息:

GET /ecommerce/product/1 //查询id为1的商品

1. **查询索引Query String Search**

GET /ecommerce/product/_search 搜索全部



```
{
  "took": 10,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 1,
    "hits": [
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "2",
        "_score": 1,
        "_source": {
          "name": "jiajieshi yagao",
          "desc": "youxiao fangzhu",
          "price": 25,
          "producer": "jiajieshi producer",
          "tags": [
            "fangzhu"
          ]
        }
      },
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "1",
        "_score": 1,
        "_source": {
          "name": "gaolujie yagao",
          "desc": "gaoxiao meibai",
          "price": 24,
          "producer": "gaolujie producer",
          "tags": [
            "meibai",
            "fangzhu"
          ]
        }
      }
    ]
  }
}
```

查询商品名包含牙膏且价格降序排序

```
GET /ecommerce/product/_search?q=name:yagao&sort=price:desc
```

2. **Query Domain specified Language,特定领域语法**

查询索引的产品信息:

```
GET /ecommerce/product/_search
{
  "query":{"match_all": {}}
}
```

查询商品名包含牙膏且价格降序排序

```
GET /ecommerce/product/_search
{
  "query": {
    "match": {
      "name": "yagao"
    }
  }
  , "sort": [
    {
      "price": {
        "order": "desc"
      }
    }
  ]
}
```

分页显示数据

```
GET /ecommerce/product/_search
{
  "query": {
    "match": {
      "name": "yagao"
    }
  },
  "sort": [
    {
      "price": {
        "order": "desc"
      }
    }
  ],
  "from": 0
  , "size": 1
}
```

查询只有名字和价格的信息

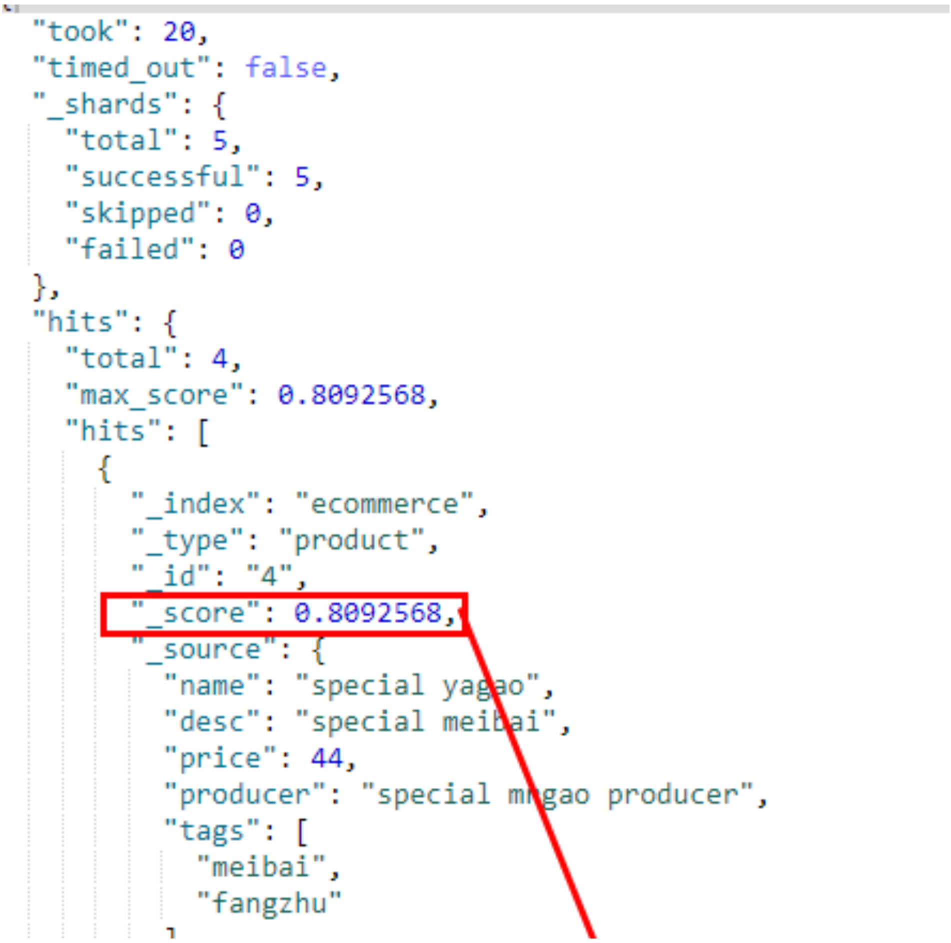
```
GET /ecommerce/product/_search
{
  "query": {
    "match_all": {}
  },
  "_source": ["name","price"]
}
```

搜索商品名称包含yagao且价格大于20小于30的商品:

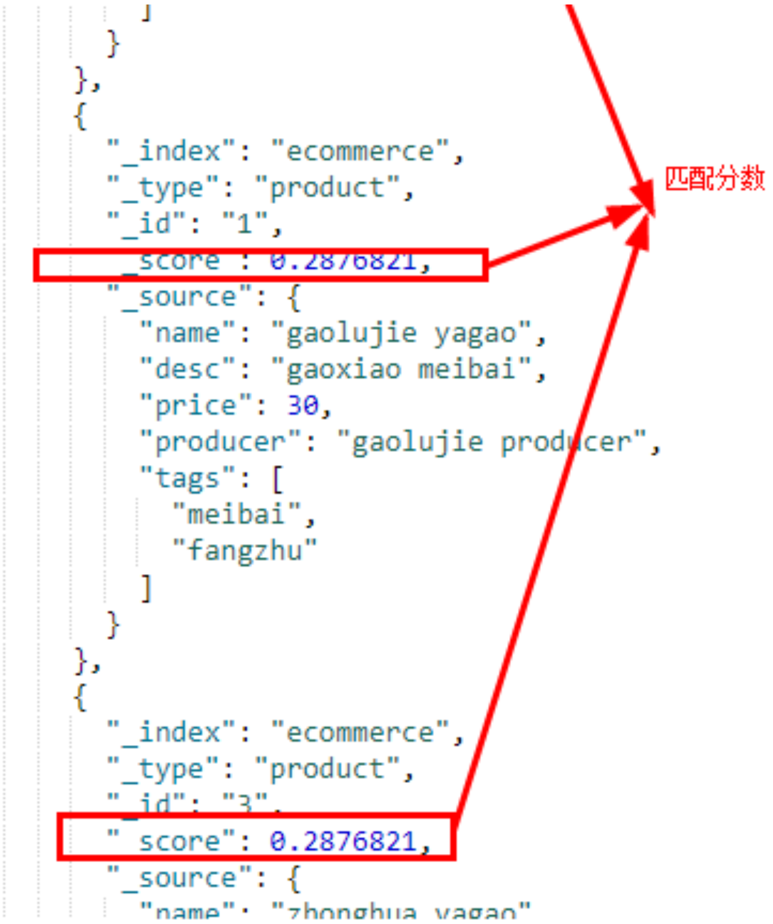
```
GET /ecommerce/product/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "match": {
            "name": "yagao"
          }
        }
      ],
      "filter": {
        "range": {
          "price": {
            "gte": 20,
            "lte": 30
        }
      }
    }
  }
 }
}
```

全文检索

```
GET ecommerce/product/_search
{
  "query": {
    "match": {
      "producer": "special producer"
    }
  }
}
```



```
  "took": 20,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 4,
    "max_score": 0.8092568,
    "hits": [
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "4",
        "_score": 0.8092568,
        "_source": {
          "name": "special yagao",
          "desc": "special meibai",
          "price": 44,
          "producer": "special mengao producer",
          "tags": [
            "meibai",
            "fangzhu"
          ]
```



```
        ]
      }
    },
    {
      "_index": "ecommerce",
      "_type": "product",
      "_id": "1",
      "_score": 0.2876821,
      "_source": {
        "name": "gaolujie yagao",
        "desc": "gaoxiao meibai",
        "price": 30,
        "producer": "gaolujie producer",
        "tags": [
          "meibai",
          "fangzhu"
        ]
      }
    },
    {
      "_index": "ecommerce",
      "_type": "product",
      "_id": "3",
      "_score": 0.2876821,
      "_source": {
        "name": "zhonghua yagao"
```

这里全文检索使用倒排索引机制:

首先会对producer字段进行拆解,拆解信息如下,数字为文档ID

special	4
mrgao	4
producer	1234
gaolujie	1
zhonghua	3
jiejieshi	2

当全文检索查询special,producer.会找到special 4 与producer 1234

短语搜索:与全文检索不同,不会将输入的字符串拆解,去倒排索引去查,只要匹配任意拆解后的单词成功即可返回结果,phrase search 要求必须按输入的字符串搜索,不能拆解查询

下面这个有查询结果

```
GET ecommerce/product/_search
{
  "query": {
    "match_phrase": {
      "producer": "special mrgao producer"
    }
  }
}
```

而这个查询无结果

```
GET ecommerce/product/_search
{
  "query": {
    "match_phrase": {
      "producer": "special producer"
    }
  }
}
```

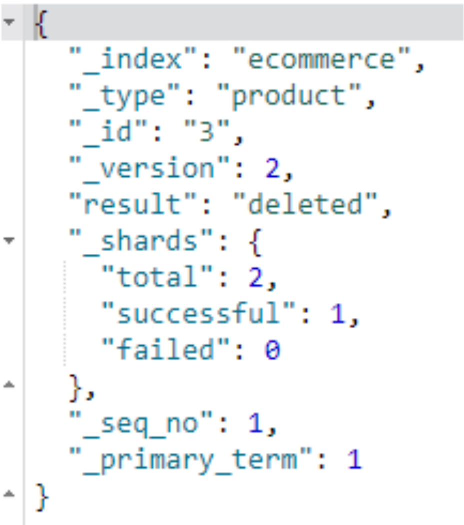
查询高亮显示:

```
GET ecommerce/product/_search
{
  "query": {
    "match": {
      "producer": "special"
    }
  },
  "highlight": {
   "fields": {
     "producer": {}
   }
  }
}
```

- 删除**索引信息:**

DELETE /ecommerce/product/3

返回信息:



```
{
  "_index": "ecommerce",
  "_type": "product",
  "_id": "3",
  "_version": 2,
  "result": "deleted",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 1,
  "_primary_term": 1
}
```

- 批处理添加，修改以及删除

As a quick example, the following call indexes two documents (ID 1 - John Doe and ID 2 - Jane Doe) in one bulk operation:

```
POST /customer/_doc/_bulk?pretty
{"index":{"_id":"1"}}
{"name": "John Doe" }
{"index":{"_id":"2"}}
{"name": "Jane Doe" }
```

This example updates the first document (ID of 1) and then deletes the second document (ID of 2) in one bulk operation:

```
POST /customer/_doc/_bulk?pretty
{"update":{"_id":"1"}}
{"doc": { "name": "John Doe becomes Jane Doe" } }
{"delete":{"_id":"2"}}
```

批处理不会因为其中一个失败而造成整个处理过程失败，仍会处理其他文档操作。并会返回一个操作的状态。