**REST API用途**

ES提供了很多全面的API，大致可以分成如下几种：

1 检查集群、节点、索引的健康情况

2 管理集群、节点，索引数据、元数据

3 执行CRUD，创建、读取、更新、删除 以及 查询

4 执行高级的查询操作，比如分页、排序、脚本、聚合等

**查看集群状态**

可以通过CURL命令发送REST命令，查询集群的健康状态：

curl 'localhost:9200/\_cat/health?v'

Localhost是主机的地址，9200是监听的端口号，ES默认监听的端口号就是9200.

这里需要注意的是，windows下安装的CURL有可能不支持单引号，如果有报错，还请改成双引号，内部使用转义字符转义。

得到的相应结果。

可以看到集群的名字是默认的"elasticsearch"，集群的状态时"green"。这个颜色之前也有说过：

1 绿色，最健康的状态，代表所有的分片包括备份都可用

2 黄色，基本的分片可用，但是备份不可用（也可能是没有备份）

3 红色，部分的分片可用，表明分片有一部分损坏。此时执行查询部分数据仍然可以查到，遇到这种情况，还是赶快解决比较好。

上面的结果还可以看到，目前有一个节点，但是没有分片，这是因为我们的ES中还没有数据，一次也就没有分片。

**当使用elasticsearch作为集群名字时，会使用单播，查询本机上是否还运行着其他的节点。如果有，则组成一个集群。**

通过下面的命令，可以查询节点的列表：

curl 'localhost:9200/\_cat/nodes?v'

**查看所有的索引**

在ES中索引有两个意思：

1 动词的索引，表示把数据存储到ES中，提供搜索的过程；这期间可能正在执行一个创建搜索的过程。

2 名字的索引，它是ES中的一个存储类型，与数据库类似，内部包含type字段，type中包含各种文档。

通过下面的命令可以查看所有的索引：

curl 'localhost:9200/\_cat/indices?v'

得到的结果如下：

curl 'localhost:9200/\_cat/indices?v'

health index pri rep docs.count docs.deleted store.size pri.store.size

由于集群中没有任何的数据，上面的结果中也就只包含列的信息了。

**创建索引**

下面是创建索引，以及查询索引的例子：

复制代码

curl -XPUT 'localhost:9200/customer?pretty'

{

"acknowledged" : true

}

curl 'localhost:9200/\_cat/indices?v'

health index pri rep docs.count docs.deleted store.size pri.store.size

yellow customer 5 1 0 0 495b 495b

复制代码

上面的结果中，customer索引的状态是yellow,这是因为此时虽然有5个主分片和一个备份。但是由于只是单个节点，我们的分片还在运行中，无法动态的修改。因此当有其他的节点加入到集群中，备份的节点会被拷贝到另一个节点中，状态就会变成green。

**索引和搜索文档**

之前说过，索引里面还有类型的概念，在索引文档之前要先设置类型type。

执行的命令如下：

curl -XPUT 'localhost:9200/customer/external/1?pretty' -d '

{

"name": "John Doe"

}'

执行成功后会得到如下的信息：

复制代码

{

"\_index" : "customer",

"\_type" : "external",

"\_id" : "1",

"\_version" : 1,

"created" : true

}

复制代码

注意2.0版本的ES在同一索引下，不同的类型，相同的字段名字，是不允许字段类型不一致的。

上面的例子中，为我们创建了一个文档，并且id自动设置为1.

ES不需要再索引文档前，不需要明确的创建索引，如果执行上面的命令，索引不存在，也会自动的创建索引。

执行下面的命令查询，返回信息也如下：

复制代码

curl -XGET 'localhost:9200/customer/external/1?pretty'

{

"\_index" : "customer",

"\_type" : "external",

"\_id" : "1",

"\_version" : 1,

"found" : true, "\_source" : { "name": "John Doe" }

}

复制代码

这里会新增两个字段：

1 found 描述了请求信息

2 \_source 为之前索引时的数据

**删除索引**

执行下面的命令就可以删除索引：

curl -XDELETE 'localhost:9200/customer?pretty'

返回结果：

{

"acknowledged": true

}

**增删改查**

可用于索引、visuazation、dashboard等资源的增删改查。

**数据写入**

ES 的一大特点，就是全 RESTful 接口处理 JSON 请求。所以，数据写入非常简单：

# curl -XPOST http://127.0.0.1:9200/logstash-2015.06.21/testlog -d '{

"date" : "1434966686000",

"user" : "chenlin7",

"mesg" : "first message into Elasticsearch"

}'

命令返回响应结果为：

{"\_index":"logstash-2015.06.21","\_type":"testlog","\_id":"AU4ew3h2nBE6n0qcyVJK","\_version":1,"created":true}

**数据获取**

可以看到，在数据写入的时候，会返回该数据的 \_id。这就是后续用来获取数据的关键：

# curl -XGET http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK

命令返回响应结果为：

{"\_index":"logstash-2015.06.21","\_type":"testlog","\_id":"AU4ew3h2nBE6n0qcyVJK","\_version":1,"found":true,"\_source":{

"date" : "1434966686000",

"user" : "chenlin7",

"mesg" : "first message into Elasticsearch"

}}

这个 \_source 里的内容，正是之前写入的数据。

如果觉得这个返回看起来有点太过麻烦，可以使用 curl -XGET http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK/\_source 来指明只获取源数据部分。

更进一步的，如果你只想看数据中的一部分字段内容，可以使用 curl -XGET http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK?fields=user,mesg 来指明获取字段，结果如下：

{"\_index":"logstash-2015.06.21","\_type":"testlog","\_id":"AU4ew3h2nBE6n0qcyVJK","\_version":1,"found":true,"fields":{"user":["chenlin7"],"mesg":["first message into Elasticsearch"]}}

**数据删除**

要删除数据，修改发送的 HTTP 请求方法为 DELETE 即可：

# curl -XDELETE http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK

删除不单针对单条数据，还可以删除整个整个索引。甚至可以用通配符。

# curl -XDELETE http://127.0.0.1:9200/logstash-2015.06.0\*

在 Elasticsearch 2.x 之前，可以通过查询语句删除，也可以删除某个 \_type 内的数据。现在都已经不再内置支持，改为 Delete by Query 插件。因为这种方式本身对性能影响较大！

**数据更新**

已经写过的数据，同样还是可以修改的。有两种办法，一种是全量提交，即指明 \_id 再发送一次写入请求。

# curl -XPOST http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK -d '{

"date" : "1434966686000",

"user" : "chenlin7",

"mesg" " "first message into Elasticsearch but version 2"

}'

另一种是局部更新，使用 /\_update 接口：

# curl -XPOST 'http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK/\_update' -d '{

"doc" : {

"user" : "someone"

}

}'

或者

# curl -XPOST 'http://127.0.0.1:9200/logstash-2015.06.21/testlog/AU4ew3h2nBE6n0qcyVJK/\_update' -d '{

"script" : "ctx.\_source.user = \"someone\""

}'