# 概述

## 简介

## 1.2 倒序索引

## 1.3 DSL

分页/排序

1. {
2. "from": 0,
3. "size": 10,
4. "sort": [{ "age": "desc" }],
5. }

### Index

#### 查看mapping

[http://10.65.3.17:9200/event\_\*/\_mapping](http://10.65.3.17:9200/event_*/_mapping)

#### create

URL: http://ip:port/index-name

Method: PUT

Body

1. {
2. "settings" : {
3. "index" : {
4. "number\_of\_shards" : 3,
5. "number\_of\_replicas" : 1
6. }
7. },
8. "mappings":{
9. "test\_index\_type":{
10. "properties":{
11. "appId":{"type":"keyword"},
12. "name":{"type":"text"},
13. "age":{"type":"integer"},
14. "address":{"type":"keyword"},
15. "time":{"type":"date"},
16. "detail":{"type":"text"}
17. }
18. }
19. }
20. }

#### delete

URL: http://ip:port/index-name

Method: DELETE

### hits

1. "hits": {
2. "total": 47158,
3. "max\_score": 6.8834624,
4. "hits": [
5. {
6. "\_index": "event\_20190725",
7. "\_type": "event",
8. "\_id": "AWwn6RtM4U5VpdVg4\_f-",
9. "\_score": 6.8834624,
10. "\_source": {
11. "intoTime": 1564037678643,
12. "action": "LOGIN",
13. "category": "APP\_MANAGER"
14. }
15. }
16. ]
17. }

total

max\_score

## Query String&Simple Query String

q

**URL**: http://10.65.3.17:9200/event\_\*/\_search

Query Params

**Key**:q

**Value**:action:LOGIN

## 1.4 Mapping

### Dynamic Mapping

### 字段类型

[Field datatypes](https://www.elastic.co/guide/en/elasticsearch/reference/6.4/mapping-types.html)

* string：5.x开始不再支持string，由text和keyword类型替代。

text在索引前analyzer将string分解成terms. 不用于排序，很少用于聚合

keyword相当于ES 2.X的定义mapping,{"name":{"type":"string","index":"not\_analyzed"}}

* Numeric datatypes：long, integer, short, byte, double, float, half\_float, scaled\_float
* 复杂数据类型：数组Array，对象Object，Nested类型([Reference](https://mp.weixin.qq.com/s?__biz=MzI1NTI3MzEwMg==&mid=2247485737&idx=1&sn=afd0defffea50fc160203a0989537365&chksm=ea3938c8dd4eb1defbc07398368e7efac84fefbe2af587ba63e491d67ccba7738e313a60c631&mpshare=1&scene=1&srcid=&sharer_sharetime=1566479720981&sharer_shareid=ff601b700721a407cdee60a9c63c1b87&key=dd5051400a9fb58fc80fb5d2c36e2d09f45f8ce685491002d723aa0c0d004c8e2d5b955c17b95edb4fc75de8698d8773470775adba7457862113ee1b1dfb22003398acc69ca6762a23359acf23e5c143&ascene=1&uin=Mjc3ODQ1MTk0MA%3D%3D&devicetype=Windows+10&version=62060841&lang=zh_CN&pass_ticket=S%2FRQHZpgLG1%2FHlyzPFg%2BuVsb5pU%2BCJJsVcRMr2dZKO5KcY2WEuI5pP59xal4spTP))

Index

http://10.65.3.17:9200/pageview\_20190626/\_search

http://10.65.3.17:9200/\_cat/indices/log\_my\_index?v

http://10.65.3.17:9200/log\_my\_index/\_search

## 1.5 Index Template&Dynamic Template

# 第二章 搜索

### 查询和过滤

"filter" 表示不评分、只过滤的查询，从 es2.0 开始，过滤（filters）已从技术上被排除，查询（queries）拥有不评分查询的能力。

## 2.1 Structured search

结构化搜索

### 精确值查找

不会计算相关度（直接跳过了整个评分阶段）且很容易被缓存。

url: [http://10.65.3.17:9200/event\_\*/\_search](http://10.65.3.17:9200/event_*/_search)

method: GET

1. {
2. "query" : {
3. .....json....
4. }
5. }

#### term

用处理数字（numbers）、布尔值（Booleans）、日期（dates）以及文本（text）。

term vs match

match会对搜索关键字进行分词

1. {
2. "query": {"term": {"detail" : "here"} } ,
3. "sort": [{ "age": "desc" }],
4. "size":  100
5. }

terms

一个字段多个值

1. {
2. "query" : {
3. "terms" : {"action" : ["OPERATE\_LOG","OPEN\_APP"]}
4. }
5. }

### 组合过滤器

#### bool

它接受多个其他过滤器作为参数，并这些过滤器结合成各式各样的布尔（逻辑）组合，构造复杂逻辑条件，

1. {
2. "bool" : {
3. "must" :     [],
4. "should" :   [],
5. "must\_not" : [],
6. }
7. }

must-and,must\_not-not,should-OR

**注意must和should之前是and的关系**

##### must/must\_not/should

1. "bool": {
2. "must":{
3. "term" : {"pttStatus" : "1"}
4. },
5. "should": [
6. { "match": { "action": "LOGIN" }}
7. ]
8. }

must有多个条件时可用 "must":[{ "term" : {"pttStatus" : "1"} ]

组合

1. "bool": {
2. "must": {
3. "term" : {"action" : "LOGIN"}
4. },
5. "filter": {
6. "range" : { "status" : { "gt" : 0 }}
7. }
8. }

##### term

1. {
2. "query" : {
3. "bool": {
4. "must":[
5. {"term":{"category": "APP\_MANAGER"}}
6. ]
7. }
8. }
9. }

##### filter

1. {
2. "query" : {
3. "bool": {
4. "filter": [
5. { "term":  { "action": "LOGIN" }},
6. {"range" : {"age" : { "gt" : 0 } }}
7. ]
8. }
9. }
10. }

##### exists

1. {
2. "query" : {
3. "bool": {
4. "must":[
5. {"term":{"category": "APP\_MANAGER"}},
6. {"exists" : {"field" : "userCode"}}
7. ],
8. "must\_not":[
9. {"exists": {"field": "value.userName"}}
10. ]
11. }
12. }
13. }

### range

1. {
2. "query": {
3. "range" : { "age" : {"from":"3","to":10"} },
4. "range" : { "code" : { "gt" : 0 } }
5. }
6. }

## document

### delete

http://ip:port/index-name/\_delete\_by\_query

method:POST

可以加过滤参数，通query

1. {
2. "query": {
3. "term": {"\_id" : "AWvWAGMOJ7AmwS\_GxknK"}
4. }
5. }

### Create

http://ip:port/index-name/index-type

method:POST

Param

1. {
2. "name": "first",
3. "clientTime": 1567733117271,
4. "address":"苏州",
5. "age": 18,
6. "detail":"this is fine day"
7. }

## 2.2 full-text search

全文搜索

### match

1. {
2. "query" : {
3. "match" : {"detail" : "here1 detail1"}
4. }
5. }

multi\_match

1. {
2. "query" : {
3. "multi\_match": {
4. "query":    "0",
5. "fields":   [ "userCode", "intoTimeStr" ]
6. }
7. }
8. }

### fuzzy

f1

1. {
2. "query": {
3. "fuzzy" : { "detail" : "heba" }
4. }
5. }

f2

1. {
2. "query": {
3. "fuzzy" : {
4. "detail" : {
5. "value" :         "detail",
6. "boost" :         1.0,
7. "fuzziness" :     2,
8. "prefix\_length" : 0,
9. "max\_expansions": 100
10. }
11. }
12. }
13. }

分词 vs 模糊查找

## 2.3 部分匹配

### prefix

1. {
2. "query": { "prefix": {"name": "a"}}
3. }

#### wildcard

范围查找

1. {
2. "query": {"wildcard": {"name": "dong\*"}}
3. }

支持模糊匹配

## 2.4 控制相关度

# 第三章 聚合

高阶概念

桶（Buckets）：满足特定条件的文档的集合

指标（Metrics）：对桶内的文档进行统计计算

aggs返回结构

1. {
2. "aggregations": {
3. "terms\_app\_id": {
4. "buckets": [
5. {
6. "key": "P8OMfaCYNMJEnuJuTbJI",
7. "doc\_count": 1
8. }...
9. ]
10. }
11. }

## Aggs

1. {
2. "aggs": {
3. "terms\_app\_id": {
4. "terms": {
5. "field": "value.appId",
6. "size": 10,
7. "order" : { "\_count" : "desc" }
8. }
9. }
10. }
11. }

## metrics

### stats

1. "aggs": {
2. "age\_stats": {"stats": {"field": "age"}}
3. }

结果包含 count/min/max/avg/sum

### cardinality

cardinality相当于sql的distinct

1. {
2. "size": 0,
3. "aggs": {
4. "app\_count": {"cardinality": { "field": "appId"}}
5. }
6. }

## 性能分析

1. {
2. "took": 77,
3. "timed\_out": false,
4. "\_shards": {
5. "total": 400,
6. "successful": 400,
7. "skipped": 0,
8. "failed": 0
9. },
10. "hits": {...}
11. }

# 第四章 分布式

## 4.1 集群

## 4.2 cluster info

查看dev es

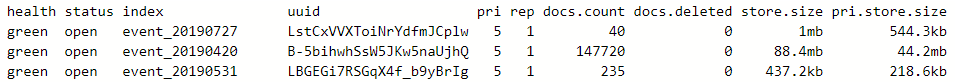
url: <http://10.65.3.17:9200> GET

1. {
2. "name": "elasticsearch-2-dev",
3. "cluster\_name": "es-dev",
4. "version": {
5. "number": "5.6.6",
6. "build\_date": "2018-01-09T23:55:47.880Z",
7. "lucene\_version": "6.6.1"
8. }
9. }

### indices

<http://10.65.3.17:9200/_cat/indices?v>

[http://10.65.3.17:9200/\_cat/indices/event\_\*?v](http://10.65.3.17:9200/_cat/indices/event_*?v)



# 第五章 ELK大数据分析

# 附件 1 java demo

## setPostFilter

SearchRequestBuilder.setPostFilter(QueryBuilder postFilter).

Sets a filter that will be executed after the query has been executed and only has affect on the search hits(not aggregations).

This filter is always executed as last filtering mechanism.

query影响hits和aggs

{

"size": 10,

"query": {"match":{"\_id":"AWuiIP2jL\_yXvRf0iFHS"}},

"aggs": {

"avg\_age": { "avg": {"field": "age"}},

"max\_age": {"max": {"field": "age" }}

}

}

filter只影响aggs

{

"aggs": {

"age\_terms": {

"filter": {"match":{"\_id":"AWuiIP2jL\_yXvRf0iFHS"}},

"aggs": {

"max\_age": {

"max": {

"field": "age"

}

}

}

}

}

}