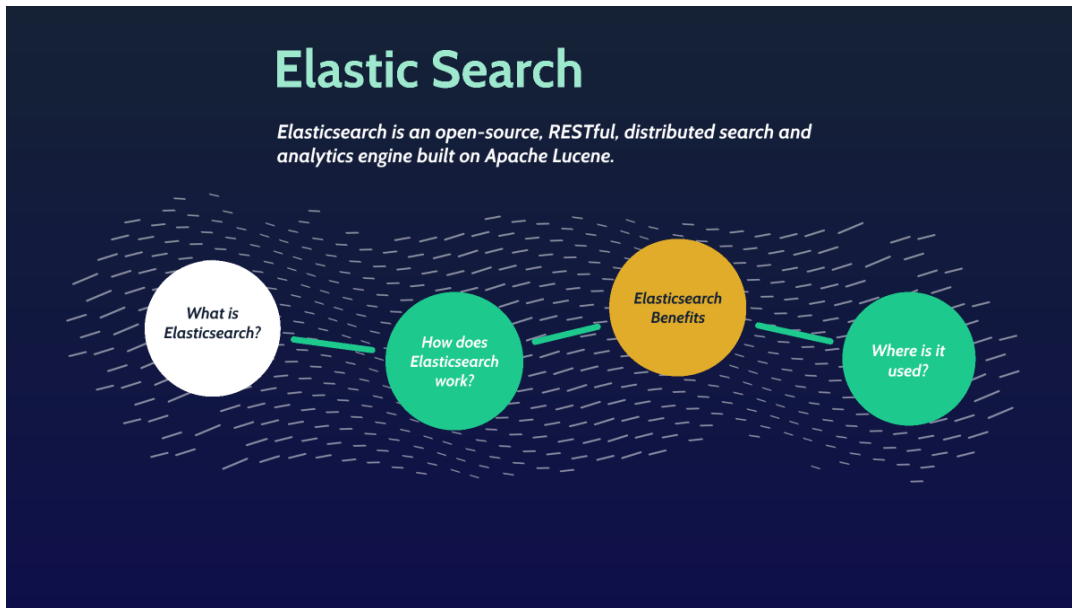


H1 初识 Elasticsearch



2 个目的:

1. 让与会人员都能自己写点 ES 语句, 立竿见影, 得到想要的统计数据
2. 乐于分享

思路: a. 了解基本概念; b. 能读懂文中语句; c. 配以演示加深印象; d. 照猫画虎.

重点: **非关系型思维, 索引配置(DBA), 聚合实例**

Ref: <https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html>

版本间功能变化略大, 建议通读最新官方文档. 若新建, 则建议直接上最新稳定版.

一些特性分享: <https://elasticsearch.cn/slides/>

以下内容基于版本: 6.4.1, 最新版本: 7.10.x, Beta: 8.0

H2 认识

Elasticsearch(ES) 是一个**分布式的实时文档存储和搜索分析引擎**.

当每天有10亿数据时, 需要解决的痛点:

- 高并发写入, 有序存储
- 数据量越来越大, 存储横向扩展问题
- 表数据增加, 分库分表复杂度越来越高
- 查询条件越来越多, 索引越来越多
- 检索数据, 统计结果, 速度变慢, 资源占用越来越高

- 全文检索???

ES 特性:

- NoSQL, 扁平化, 文档型, 类似 MongoDB, JSON
- LSM树, 索引(动词)和搜索快速无锁:
 - 倒排索引不可变性, 无需考虑并发写文件问题
 - 倒排(反向)索引则是通过 value 找 key, 正向索引是通过 key 找 value
 - 一旦读入内核系统缓存就留在那, 大部分请求直接命中内存
 - Ref: <https://www.cnblogs.com/huaweiyun/p/13679175.html>
- 分布式, 集群, 副本备份, 节点竞选
- 分析器, 过滤字符, 分词, 过滤词
- 全文检索, 每个字段都可以被索引与搜索, 高亮结果, 速度快
- 支持上百个节点扩展, 支持 PB 级结构化或非结构化数据
- RESTful API

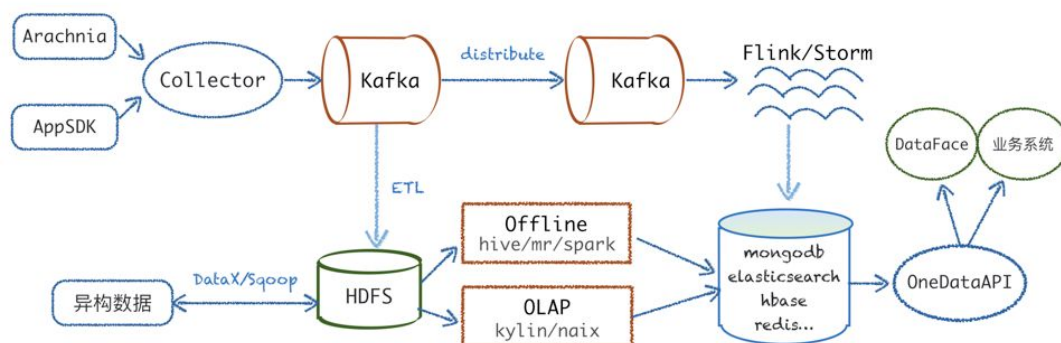
低成本(学习), 高可用, 少运维

实时秒级响应, 若慢:

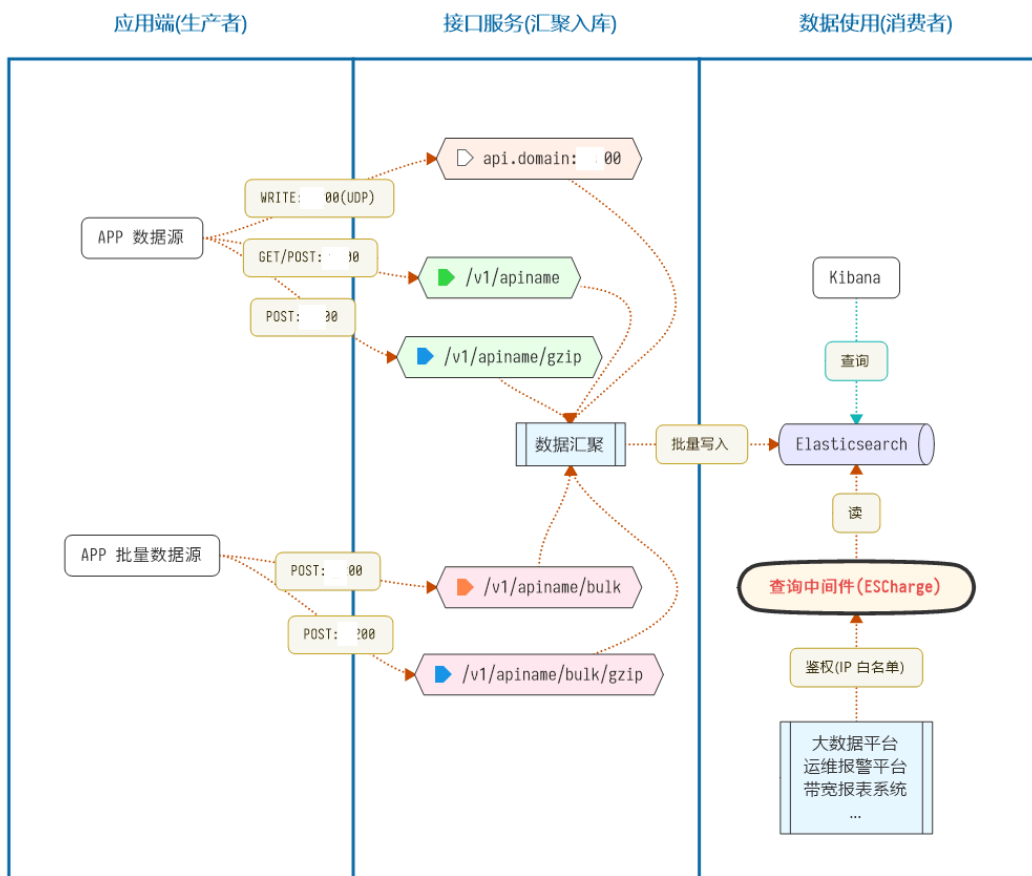
- 数据结构不合理
- 查询语句不优
- 磁盘性能不佳

适用场景:

- 大数据量, 读多写少, 数据几乎无更新的需求
- 复杂场景查询需求, 查询性能有要求, 写入及时性要求不高
- 日志分析, Elastic Stack (ES, Logstash, Kibana, Beats)
- 全文检索首选, 个性化推荐, 快, 高亮 (商品, 新闻推荐, GitHub)
- 事件数据和指标统计
- 数据可视化(Kibana)



Elasticsearch 基础架构



使用前需要知道的:

- 字段类型无法修改
- 写有一定延时, 比如 1 秒
- 吃硬件, 没有 SSD 不要用排序和聚合
- 不支持事务
- 多表关联查询支持弱:
 - 应用层联接, 两次查询 (数据少, 内存码表)
 - 宽表冗余存储 (一对多/多对多)
 - Nested 嵌套类型, 超集合类型 (子文档更新少, 查询多)
 - 父-子关系文档, Join 类型 (一对多, 子文档更新频繁)
 - Ref: <https://www.elastic.co/guide/en/elasticsearch/reference/current/parent-join.html>
- 不支持数据的权限管理
- 深分页场景性能差 (结果最多 10000)

PS:

- 分布式: 一个业务分拆多个子业务, 部署在不同的服务器上
- 集群: 同一个业务, 部署在多个服务器上

H2 概念

- **Cluster** 集群
- **Node** 节点: 每个 ES 实例
- **Shard** 分片: 分布式
- **Replia** 副本: 提高吞吐量, 实现高可用
- **Index** 索引: 类比单个数据库表(一库一表), **名称必须小写**
- **Type** 类型: 弱化了, 之前可类比指数据库中的表. v8.0 取消
- **Document** 文档: 类比一行数据, 相互独立, 文档字段可以不一致:

```
1 {  
2   "Domain": "prod-live-cfentry.playbattlegrounds.com",  
3   "Exception": "",  
4   "IP": "222.111.11.3",  
5   "Mark": {  
6     "arr": [ 1, 2, 3 ],  
7     "date": "2020-11-23T10:55:55+08:00"  
8   },  
9   "Parsesuccess": false,  
10  "Port": 443,  
11  "Readbytes": 0  
12 }
```

- **Field** 字段: 即文档 JSON Key
- **Mapping** 映射: 类比数据表结构定义
 - 定义字段类型
 - 字符串: `string`
 - 整数: `byte`, `short`, `integer`, `long`
 - 浮点数: `float`, `double`
 - 布尔型: `boolean`
 - 日期: `date` 有效的日期字符串
 - 空域: `null`, `[]`, `[null]` 不会被索引
 - `alias`, `nested`, `join`, `range`, `ip`, `version`, `murmur3`, `text`, `geo`, `point`, `shape`, `binary` ...
 - 多层级对象
 - 自动猜测类型
 - Ref: <https://pdf.us/2018/04/16/897.html>
 - 是否索引, 是否存储
 - 分词器: `ik_smart`, `ik_max_word`

ElasticSerach	Mysql
Index	Database

ElasticSerach	Mysql
Type	Table
Document	Row
Field	Column
Mapping	Schema
Everything is indexed	Index(表的索引)
ID	Primary Key
Query DSL	SQL
PUT/POST http://....	insert into
GET http://....	select * from ...
POST http://... (搜索操作)	selcct * from... like ...
PUT http://....	update
DELETE http://....	delete from...

H2 安装

```

1 # 下载
2 curl -L -O
  https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-
  7.10.0-linux-x86_64.tar.gz
3 # 解压
4 tar -xvf elasticsearch-7.10.0-linux-x86_64.tar.gz
5 # 运行
6 elasticsearch-7.10.0/bin/elasticsearch -d

```

依赖 JAVA 环境, 集群配置略.

- 9200 API http 服务端口
- 9300 ES 节点内通信端口

信息查询 (RESTful API):

```

1 # 集群信息
2 GET /
3 curl 192.168.20.103:9200
4 # 健康情况
5 GET _cluster/health
6 curl 192.168.20.103:9200/_cluster/health?pretty

```

kibana

- Discover
- Visualize
- Dashboard
- Timelion
- APM
- Dev Tools
- Monitoring
- Management

Clusters / xunyou-es / Elasticsearch

Overview Nodes Indices

Nodes: 3 Indices: 2103 Memory: 43.8 GB / 71.6 GB Total Shards: 20884

Filter Nodes...

Name ↑	Status	CPU Usage
★ node-0 192.168.20.103:9300	● Online	20% ↓ 41% max 14% min
node-1 192.168.20.104:9300	● Online	15% ↓ 89% max 14% min
node-2 192.168.20.105:9300	● Online	16% ↓ 82% max 10% min

GET http://[redacted]:9200/_cluster/health

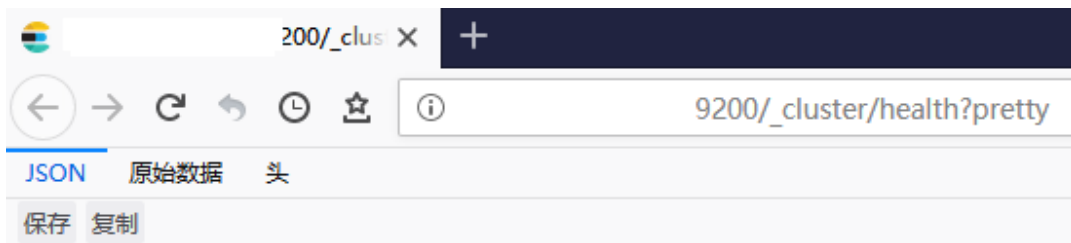
Params Authorization Headers (9) Body ● Pre-request Script Tests Settings

Body Cookies Headers (3) Test Results

Pretty Raw Preview Visualize JSON

```

1  {
2    "cluster_name": "xunyou-es",
3    "status": "green",
4    "timed_out": false,
5    "number_of_nodes": 3,
6    "number_of_data_nodes": 3,
7    "active_primary_shards": 10442,
8    "active_shards": 20884,
9    "relocating_shards": 0,
10   "initializing_shards": 0,
11   "unassigned_shards": 0,
12   "delayed_unassigned_shards": 0,
13   "number_of_pending_tasks": 0,
14   "number_of_in_flight_fetch": 0,
15   "task_max_waiting_in_queue_millis": 0,
16   "active_shards_percent_as_number": 100.0
17 }
```



- green 所有的主分片和副本分片都正常运行
- yellow 所有的主分片都正常运行, 但不是所有的副本分片都正常运行
- red 有主分片没能正常运行

PS: 一个机器可以启动多个实例, 一个实例就是一个节点.

```
1 ./elasticsearch -Epath.data=data2 -Epath.logs=log2
2 ./elasticsearch -Epath.data=data3 -Epath.logs=log3
```

H2 索引

H3 1. 新建索引

PS: 数据量大的索引最好按天新建. 不要用数据量非常大的索引, 影响查询效率.

H4 1.1 主分片和副本分片

主分片与副本分片会分布在不同节点, 7.0 后默认主分片为 1, 副本分片为 0

```
1 PUT /test001
2 {
3   "settings" : {
4     "number_of_shards" : 5,
5     "number_of_replicas" : 1
6   }
7 }
```

```

1 {
2   "acknowledged": true,
3   "shards_acknowledged": true,
4   "index": "test001"
5 }

```

主分片数为 5, 索引创建后无法修改. 每个主分片的复制分片数为 1, 可以随时修改.

```

bw_collect_200922 0 2 3

```

```

bw_collect_200922 0 1 4

```

```

bw_collect_200922 1 2 3 4

```

PS: 增加节点并增加副本分片, 吞吐量更大, 搜索性能更高.

```

1 GET xy_auth_monitor_report_201125/_search
2 {
3   "explain": true,
4   "from": 0,
5   "size": 10,
6   "sort": [
7     {
8       "timestamp": {
9         "order": "desc"
10      }
11    }
12  ]
13 }

```

Ref: <https://www.bilibili.com/video/BV1TJ41ID7ya>

```

1 PUT /test001/_settings
2 {
3   "number_of_replicas": 2
4 }

```

```

1 GET /test001

```

```

1 {
2   "test001": {
3     "aliases": {},
4     "mappings": {},
5     "settings": {
6       "index": {
7         "creation_date": "1606335251589",
8         "number_of_shards": "5",
9         "number_of_replicas": "2",
10        "uuid": "okxJQuyMSiS8xStzDMN-fw",
11        "version": {
12          "created": "6040199"

```



```

13         },
14         "provided_name": "test001"
15     }
16 }
17 }
18 }

```

H4 1.2 索引的别名 (Index Aliases)

类比关系数据库的视图, 将多个索引按一定过滤条件生成别名对外提供查询:

- 周 KPI 变化图 (week_kpi)
- 用户 B 段报警 (地图)

Ref: <https://www.cnblogs.com/Neeo/articles/10897280.html>

H5 1.2.1 索引模板

```

1 PUT /_template/base_settings
2 {
3     "index_patterns": ["*"],
4     "order": 0,
5     "settings": {
6         "number_of_shards": 3,
7         "number_of_replicas": 1
8     }
9 }

```

```

1 PUT /_template/base_mapping
2 {
3     "index_patterns": [
4         "*_20*"
5     ],
6     "order": 1,
7     "mappings": {
8         "_doc": {
9             "properties": {
10                 "_cip": {
11                     "type": "ip"
12                 },
13                 "_ctime": {
14                     "type": "date"
15                 },
16                 "_gtime": {
17                     "type": "date"
18                 },
19                 "timestamp": {
20                     "type": "date"
21                 }
22             }
23         }
24     },
25     "aliases": {
26         "{index}_as": {}
27     }
28 }

```

```
27 | }
28 | }
```

```
1 | PUT /test_201201/_doc/1
2 | {
3 |   "_cip": "1.1.1.2",
4 |   "timestamp": 1606335251589
5 | }
```

H5 1.2.2 Rollover Index

Ref: <https://www.elastic.co/guide/en/elasticsearch/reference/6.4/indices-rollover-index.html>

H4 1.3 映射(Mapping)

```
1 | GET /test001/_mapping
```

```
1 | PUT /test001
2 | {
3 |   "settings": {
4 |     "number_of_shards": 3,
5 |     "number_of_replicas": 1
6 |   },
7 |   "mappings": {
8 |     "test001": {
9 |       "properties": {
10 |        "game_id": {
11 |          "type": "long"
12 |        },
13 |        "game_no": {
14 |          "type": "alias",
15 |          "path": "game_id"
16 |        },
17 |        "game_areaaid": {
18 |          "type": "long"
19 |        },
20 |        "user_name": {
21 |          "type": "keyword"
22 |        },
23 |        "exception": {
24 |          "type": "text"
25 |        },
26 |        "blob": {
27 |          "type": "binary"
28 |        },
29 |        "end_time": {
30 |          "type": "date"
31 |        },
32 |        "ip_addr": {
33 |          "type": "ip"
34 |        }
35 |      }
36 |    }
37 | }
```

```
36 |     }
37 |   }
38 | }
```

H5 1.3.1 举例 IP:

```
1 | PUT /test001/test001/1
2 | {
3 |   "game_id": 123,
4 |   "game_areaid": 666,
5 |   "exception": "This is a test.",
6 |   "end_time": "2020-11-26T12:00:00+08:00",
7 |   "ip_addr": "192.168.1.100"
8 | }
```

```
1 | GET /test001/_search
2 | {
3 |   "query": {
4 |     "term": {
5 |       "ip_addr": "192.168.1.0/24"
6 |     }
7 |   }
8 | }
```

PS: 192.168.1.100, 0.0.0.0/0, 192.168.1.123/16, 192.168.1.0/26

H5 1.3.2 举例 Date:

默认日期格式: strict_date_optional_time||epoch_millis, 自定义日期格式:

```
1 | {
2 |   "mappings": {
3 |     "_doc": {
4 |       "properties": {
5 |         "mydate": {
6 |           "type": "date",
7 |           "format": "yyyy-MM-dd HH:mm:ss||yyyy-MM-dd||epoch_millis"
8 |         }
9 |       }
10 |     }
11 |   }
12 | }
```

```
1 | POST /test001/test001
2 | {
3 |   "end_time": 1606361965000
4 | }
```

```

1 POST /test001/test001
2 {
3   "end_time": "2020-11-26"
4 }

```

```

1 GET /test001/_search
2 {
3   "query": {
4     "range": {
5       "end_time": {
6         "gte": "2020-11-26T11:00:00+08:00",
7         "lte": "2020-11-26T13:00:00+08:00"
8       }
9     }
10  }
11 }

```

H4 1.4 分析器

H5 1.4.1 测试分析器

```

1 POST _analyze
2 {
3   "tokenizer": "standard",
4   "filter": ["lowercase", "asciifolding"],
5   "text": "Is this déjà vu?"
6 }

```

标准分词, 转为小写, 转为等效 ASCII 字符, 得到 4 个词: is, this, déjà, vu

```

1 POST _analyze
2 {
3   "analyzer": "simple",
4   "text": "Is this déjà vu?"
5 }

```

简单分析器无配置, 标准分词后转小写. 常用还有: whitespace, stop, pattern

```

1 PUT /test003
2 {
3   "settings": {
4     "analysis": {
5       "analyzer": {
6         "my_email_analyzer": {
7           "type": "pattern",
8           "pattern": "\\W|_",
9           "lowercase": true
10        }
11      }
12    }
13  }
14 }

```

```

1 POST /test003/_analyze
2 {
3   "analyzer": "my_email_analyzer",
4   "text": "John_Smith@foo-bar.com"
5 }

```

```

1 {
2   "tokens": [
3     {
4       "token": "john",
5       "start_offset": 0,
6       "end_offset": 4,
7       "type": "word",
8       "position": 0
9     },
10    {
11      "token": "smith",
12      "start_offset": 5,
13      "end_offset": 10,
14      "type": "word",
15      "position": 1
16    }
17    ...
18  ]
19 }

```

H5 1.4.2 自定义分析器

```

1 PUT /test002
2 {
3   "settings": {
4     "analysis": {
5       "char_filter": {
6         "&_to_and": {
7           "type": "mapping",
8           "mappings": [
9             "&⇒ and "
10          ]
11        }
12      },
13      "filter": {
14        "my_stopwords": {
15          "type": "stop",
16          "stopwords": [
17            "the",
18            "a"
19          ]
20        }
21      },
22      "analyzer": {
23        "my_analyzer": {
24          "type": "custom",
25          "char_filter": [
26            "html_strip",
27            "&_to_and"

```

```

28         ],
29         "tokenizer": "standard",
30         "filter": [
31             "lowercase",
32             "my_stopwords"
33         ]
34     }
35 }
36 }
37 },
38 "mappings": {
39     "test002": {
40         "properties": {
41             "html": {
42                 "type": "text",
43                 "analyzer": "my_analyzer"
44             }
45         }
46     }
47 }
48 }

```

标准分词结果:

```

1 POST _analyze
2 {
3     "analyzer": "standard",
4     "text": "\"\"<p><img
src=https://image.xunyou.com/images/2019/20190911/pubg/1568627554419.
png\" width=118\" height=136\" alt=AI智能加速\" /></p><p class=tit>AI智
能加速</p><p class=wz>XunYou全区加速</p>\"\"\"
5 }

```

```

1 {
2     "tokens": [
3         {
4             "token": "p",
5             "start_offset": 1,
6             "end_offset": 2,
7             "type": "<ALPHANUM>",
8             "position": 0
9         },
10        {
11            "token": "img",
12            "start_offset": 4,
13            "end_offset": 7,
14            "type": "<ALPHANUM>",
15            "position": 1
16        }
17        ...
18    ]
19 }

```

使用自定义分析结果:

```

1 GET /test002/_analyze
2 {
3   "analyzer": "my_analyzer",
4   "text": "<p><img
src=\"https://image.xunyou.com/images/2019/20190911/pubg/1568627554419
.png\" width=\"118\" height=\"136\" alt=\"AI智能加速\" /></p><p
class=\"tit\">AI智能加速</p><p class=\"wz\">XunYou全区加速</p>"
5 }

```

```

1 {
2   "tokens": [
3     {
4       "token": "ai",
5       "start_offset": 142,
6       "end_offset": 144,
7       "type": "<ALPHANUM>",
8       "position": 0
9     },
10    {
11      "token": "智",
12      "start_offset": 144,
13      "end_offset": 145,
14      "type": "<IDEOGRAPHIC>",
15      "position": 1
16    }
17    ...
18  }

```

PS: 中文分词需要安装扩展: `medcl/elasticsearch-analysis-ik`

H3 2. 删除索引

```

1 DELETE /test001,test002

```

```

1 DELETE /test00*

```

```

1 curl -XDELETE 'http://192.168.20.103:9200/test00*'

```

```

1 {
2   "acknowledged": true
3 }

```

!!! 删除全部索引 !!!: `DELETE /_all`

H3 3. 创建文档

H4 3.1 单条数据

须知:

- 索引(动词)文档(数据)时, 若索引不存在, 则自动创建索引.
- 第一次的数据将自动猜测类型, 往后的数据中相同字段类型必须相同
 - `long` 数字字符串可以与数字字段通用
 - `date` 可以是时间戳(数字或数字字符串)
- 新增文档时字段可以与已定义的 `mapping` 不一致, 索引文档时随意增减
- 文档字段名可以是任何合法字符串, **不能包含英文句号(.)**
- 可以指定文档 `_id`, 也可以自动生成

```

1 PUT /test004/test004/1?timeout=5m
2 {
3   "user": "ff",
4   "post_date": "2009-11-15T14:12:12",
5   "text": "put data to Elasticsearch"
6 }
```

```

1 POST /test004/test004
2 {
3   "uid": 123,
4   "post_date": 1606381535000,
5   "text": "post data to Elasticsearch"
6 }
```

H4 3.2 批量数据

推荐!!!

语法如下:

```

1 action_and_meta_data\n
2 optional_source\n
3 action_and_meta_data\n
4 optional_source\n
5 ... .
6 action_and_meta_data\n
7 optional_source\n
```

```

1 POST _bulk
2 {"index":{"_index":"test001","_type":"test001","_id":"1"}}
3 {"user_name":"fufu","game_id":111}
4 {"delete":{"_index":"test001","_type":"test001","_id":"3"}}
5 {"update":{"_index":"test001","_type":"test001","_id":"1"}}
6 {"doc":{"user_name":"ff","game_areaaid":777}}
7 {"create":{"_index":"test001","_type":"test001","_id":"3"}}
8 {"user_name":"okok"}
9 {"index":{"_index":"test001","_type":"test001"}}
10 {"user_name":"test","ip_addr":"1.1.1.1"}
11
```

```

1 {
```



```

2  "took": 23334,
3  "errors": false,
4  "items": [
5    {
6      "index": {
7        "_index": "test001",
8        "_type": "test001",
9        "_id": "1",
10       "_version": 1,
11       "result": "created",
12       "_shards": {
13         "total": 2,
14         "successful": 2,
15         "failed": 0
16       },
17       "_seq_no": 0,
18       "_primary_term": 1,
19       "status": 201
20     }
21   },
22   {
23     "delete": {
24       "_index": "test001",
25       "_type": "test001",
26       "_id": "3",
27       "_version": 1,
28       "result": "not_found",
29       "_shards": {
30         "total": 2,
31         "successful": 2,
32         "failed": 0
33       },
34       "_seq_no": 0,
35       "_primary_term": 1,
36       "status": 404
37     }
38   },
39   {
40     "update": {
41       "_index": "test001",
42       "_type": "test001",
43       "_id": "1",
44       "_version": 2,
45       "result": "updated",
46       "_shards": {
47         "total": 2,
48         "successful": 2,
49         "failed": 0
50       },
51       "_seq_no": 1,
52       "_primary_term": 1,
53       "status": 200
54     }
55   },
56   ...

```

H3 4. 删除文档

一般不建议删除。

H4 4.1 按 `_id` 删除

```
1 | DELETE /test004/test004/1
```

H4 4.2 筛选后删除

```
1 | POST test004/_delete_by_query
2 | {
3 |   "query": {
4 |     "match": {
5 |       "uid": 123
6 |     }
7 |   }
8 | }
```

H2 查询

H3 1. 按 ID 查询

H4 1.1 单 ID 查询

```
1 | GET /test004/test004/1
```

```
1 | GET /test004/test004/1?_source=user,*_date
```

```
1 | {
2 |   "_index": "test004",
3 |   "_type": "test004",
4 |   "_id": "1",
5 |   "_version": 3,
6 |   "found": true,
7 |   "_source": {
8 |     "post_date": "2009-11-15T14:12:12",
9 |     "user": "ff"
10 |  }
11 | }
```

Ref: <https://www.elastic.co/guide/en/elasticsearch/reference/6.4/search-uri-request.html>

H4 1.2 多 ID 查询

```

1 GET /_mget
2 {
3   "docs": [
4     {
5       "_index": "test004",
6       "_type": "test004",
7       "_id": "1"
8     },
9     {
10      "_index": "test001",
11      "_type": "test001",
12      "_id": "1"
13    }
14  ]
15 }

```

H3 2. 条件查询

H4 2.1 空条件查询

```

1 GET /test004/_search

```

```

1 GET /test004/_search
2 {
3   "query": {
4     "match_all": {}
5   }
6 }

```

H4 2.2 基于模板查询

- 搜索时带模板
- 文件模板
- 预定义模板

```

1 POST _scripts/test_tpl001
2 {
3   "script": {
4     "lang": "mustache",
5     "source": {
6       "query": {
7         "match": {
8           "uid": "{{q_uid}}"
9         }
10      }
11    }
12  }
13 }

```

查看模板:

```
1 GET _scripts/test_tpl001
```

使用模板查询:

```
1 GET /test004/_search/template
2 {
3   "id": "test_tpl001",
4   "params": {
5     "q_uid": 123
6   }
7 }
```

删除模板:

```
1 DELETE _scripts/test_tpl001
```

多模板批量搜索, 略...

H4 2.3 条件查询

```
1 GET /test004/_search?q=uid:123
```

```
1 GET /test004/_search
2 {
3   "query": {
4     "match": {
5       "uid": 123
6     }
7   }
8 }
```

```
1 GET /test004/_search
2 {
3   "query": {
4     "bool": {
5       "filter": {
6         "term": {
7           "uid": 123
8         }
9       }
10    }
11  }
12 }
```

H4 2.4 组合查询

```

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

```

- `must` 所有的语句都 必须(*must*) 匹配, 与 `AND` 等价
- `must_not` 所有的语句都 不能(*must not*) 匹配, 与 `NOT` 等价
- `should` 至少有一个语句要匹配, 与 `OR` 等价

可嵌套:

```

1 SELECT *
2 FROM test001
3 WHERE game_id = 1616
4 OR (game_id = 23646 AND user_name = 'fufu')

```

转为 DSL 语句:

```

1 GET /test001/_search
2 {
3   "query": {
4     "bool": {
5       "should": [
6         { "term": { "game_id": { "value": 1616 }}}},
7         {
8           "bool": {
9             "must": [
10              { "term": { "game_id": { "value": 111 }}}},
11              { "term": { "user_name": { "value": "ff" }}}
12            ]
13          }
14        }
15      ] } } }

```

H4 2.5 查询与过滤

```

1 GET _search
2 {
3   "query": {
4     "bool": {
5       "must": [
6         { "match": { "name": "Jim" }},
7         { "match": { "city": "Guangzhou" }}
8       ],
9       "filter": [
10        { "term": { "weight": "60" }},
11        { "range": { "age": { "gte": "18" }}}
12      ]
13    }
14  }
15 }

```

```
14 }
15 }
```

一些概念:

- 单查询子句: 可单独使用, 对特定字段查询特定值: `term`, `match`, `range`
- 复合查询子句: 组合查询, 逻辑组合: `bool`
- 查询上下文: `query` 查询并计算分值, 按相关度排序, `_score`
- 过滤上下文: `filter` 仅判断是否满足查询条件, `yes` or `no`, 不评分, 也不关心排序, 查询结果可以被缓存, 性能高
- 精确匹配时最好用过滤语句 `filter`, `term`, `terms`
- 全文检索用 `match`, `multi_match`
- 无值 `missing` (`IS_NULL`), 有值 `exists` (`NOT IS_NULL`)
- Ref: https://www.elastic.co/guide/cn/elasticsearch/guide/current/_queries_and_filters.html

更多查询:

- 相关度控制: 权重, 评分
- 近似匹配: 短语匹配, 多值字段, 相关词
- 部分匹配: 前缀查询, 通配符及正则查询

```
1 {
2   "query": {
3     "prefix": {
4       "postcode": "XY"
5     }
6   }
7 }
```

```
1 {
2   "query": {
3     "regexp": {
4       "postcode": "XY[0-9].+"
5     }
6   }
7 }
```

H4 2.6 输出结果控制

`_source`, `stored_fields`, `script_fields` 结果字段

```
1 GET /test001/_search
2 {
3   "query": {
4     "term": {
5       "game_id": {
6         "value": 111
```

```

7     }
8     }
9 },
10 "_source": {
11     "includes": [ "game_*", "ip_addr" ],
12     "excludes": [ "end_time" ]
13 },
14 "script_fields": {
15     "game_id_plus": {
16         "script": {
17             "lang": "painless",
18             "source": "doc['game_id'].value + 10000"
19         }
20     },
21     "game_areaid_test": {
22         "script": {
23             "lang": "painless",
24             "source": "doc['game_areaid'].value + params.prefix",
25             "params": {
26                 "prefix": 20000
27             }
28         }
29     },
30     "game_areaid_alias": {
31         "script": "params['_source']['game_areaid']"
32     }
33 }
34 }

```

```

1 {
2     "_index": "test001",
3     "_type": "test001",
4     "_id": "1",
5     "_score": 1,
6     "_source": {
7         "game_areaid": 777,
8         "game_id": 111
9     },
10    "fields": {
11        "game_id_plus": [
12            10111
13        ],
14        "game_areaid_test": [
15            20777
16        ],
17        "game_areaid_alias": [
18            777
19        ]
20    }
21 }

```

highlight 高亮

```

1 GET /test004/_search
2 {
3   "query" : {
4     "match": { "text": "Elasticsearch" }
5   },
6   "highlight" : {
7     "fields" : {
8       "text" : {}
9     }
10  }
11 }

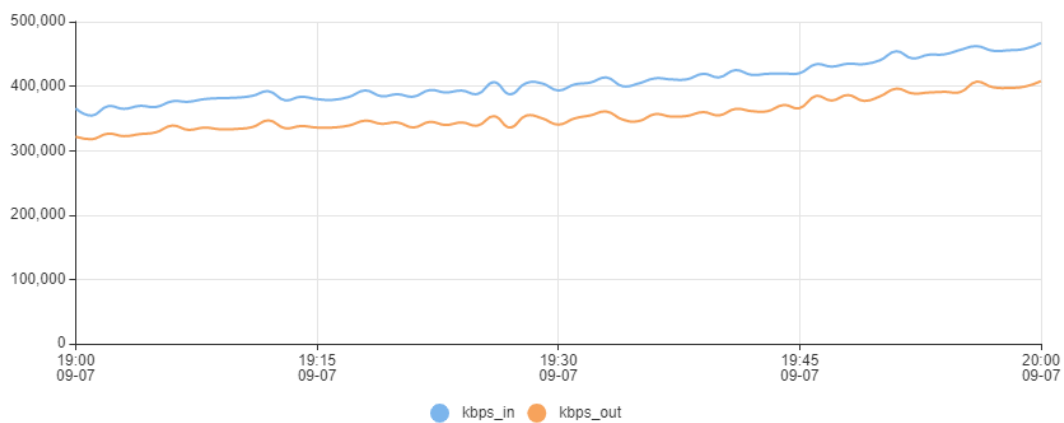
```

```

1 "highlight": {
2   "text": [
3     "put data to <em>Elasticsearch</em>"
4   ]
5 }

```

H4 2.7 查询 18-20 时中欧接口带宽数据



```

1 GET bw_collect_201126/_search
2 {
3   "query": {
4     "bool": {
5       "must": [
6         {
7           "range": {
8             "time": {
9               "gte": "2020-11-26T18:00:00.000+08:00",
10              "lt": "2020-11-26T20:00:00.000+08:00"
11            }
12          }
13        },
14        {
15          "term": {
16            "interface.keyword": {
17              "value": "10GE1/0/111"
18            }
19          }
20        }
21      ]
22    }
23  }
24 }

```



```

23     },
24     "sort": [
25         {
26             "time": {
27                 "order": "asc"
28             }
29         }
30     ],
31     "size": 1000,
32     "_source": ["client_ip", "kbps*", "time"]
33 }

```

SQL 方式查询:

```

1 POST /_xpack/sql?format=txt
2 {
3     "query": "SELECT client_ip, kbps_in, kbps_out, time FROM
bw_collect_201126 WHERE time ≥ '2020-11-26T18:00:00.000+08:00' and
time < '2020-11-26T20:00:00.000+08:00' and
interface.keyword='10GE1/0/111' ORDER BY time ASC"
4 }

```

如果中欧带宽有多个接口呢?

```

1 POST /_xpack/sql?format=txt
2 {
3     "query": "SELECT kbps_in, kbps_out, time, interface FROM
bw_collect_201126 WHERE time ≥ '2020-11-26T18:00:00.000+08:00' and
time < '2020-11-26T20:00:00.000+08:00' and
(interface.keyword='10GE1/0/111' or interface.keyword='Vlanif1777')
ORDER BY time ASC"
4 }

```

SQL 转为 DSL 结构化查询:

```

1 POST /_xpack/sql/translate
2 { ... }

```

```

1 GET bw_collect_201126/_search
2 {
3     "query": {
4         "bool": {
5             "must": [
6                 {
7                     "range": {
8                         "time": {
9                             "from": "2020-11-26T18:00:00.000+08:00",
10                            "to": "2020-11-26T20:00:00.000+08:00"
11                        }
12                    }
13                },
14                {
15                    "bool": {

```

```

16         "should": [
17             {
18                 "term": {
19                     "interface.keyword": {
20                         "value": "10GE1/0/111"
21                     }
22                 }
23             },
24             {
25                 "term": {
26                     "interface.keyword": {
27                         "value": "Vlanif1777"
28                     }
29                 }
30             }
31         ]
32     },
33 }
34 ]
35 }
36 },
37 "sort": [
38     {
39         "time": {
40             "order": "asc"
41         }
42     }
43 ],
44 "size": 1000,
45 "_source": ["client_ip", "kbps_in", "kbps_out", "time"]
46 }

```

SQL 增强

7.2

7.2新的SQL功能:

- 支持地理位置查询
- median absolute deviation 函数
- 支持CASEE/WHEN/ELSE/END

```

SELECT count(*) AS count,
CASE WHEN NVL(languages, 0) = 0 THEN 'zero'
      WHEN languages = 1 THEN 'one'
      WHEN languages = 2 THEN 'bilingual'
      WHEN languages = 3 THEN 'trilingual'
      ELSE 'multilingual'
END as lang_skills
FROM employees
GROUP BY lang_skills
ORDER BY lang_skills;

```

在SQL中可以使用你自己定义的坐标系

```
POST ogc/_bulk
{"index":{"_id": "101"}}
{"ogc_type":"lakes", "fid": 101, "name": "BLUE LAKE", "shore": "POLYGON ((52 18, 66 23, 73 9, 48 6, 52 18), (59 18, 67 18, 67 13, 59 13, 59 18))"}
{"index":{"_id": "102"}}
{"ogc_type":"road_segments", "fid": 102, "name": "Route 5", "num_lanes": 2, "centerline": "LINESTRING (0 18, 10 21, 16 23, 28 26, 44 31)"}
{"index":{"_id": "103"}}
```



```
POST _sql?format=txt
{
  "query": "SELECT address, position, footprint from ogc where ogc_type='buildings'"
}
```



1	2	3	4	5
	address	IST_X(position)	IST_Y(position)	
2				
3	123 Main Street	52.0	130.0	
4	215 Main Street	164.0	133.0	
5				

5



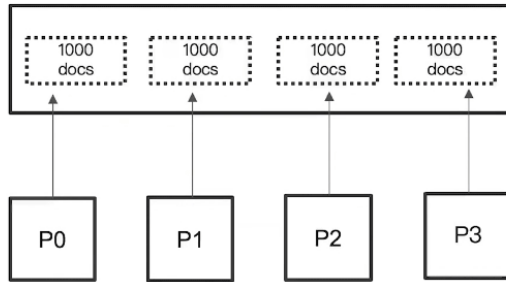
H4 2.8 最近 10 分钟 Nagios 报警

项目Code	节点id	节点名称	总用时(分)	服务器ip	报警信息	最后告警时间
nagios	57	河南测试	16471	218.198.	河南测试:udp	2020-11-27 21:32:02
nagios	113	浙江节点4	1382	122.224.	浙江节点4:udp	2020-11-27 21:32:02
nagios	5	浙江节点	1373	183.129.	浙江节点:udp	2020-11-27 21:32:02

```
1 GET /monitor_alarm_info_201127/_search
2 {
3   "query": {
4     "bool": {
5       "must": [
6         {
7           "term": {
8             "code": {
9               "value": "nagios"
10            }
11          }
12        },
13        {
14          "range": {
15            "time": {
16              "gte": "now-10m"
17            }
18          }
19        }
20      ]
21    }
22  }
23 }
```

H3 3. 分页

分布式系统中深度分页的问题



- ES 天生就是分布式的。查询信息，但是数据分别保存在多个分片，多台机器上，ES 天生就需要满足排序的需要（按照相关性算分）
- 当一个查询：From = 990, Size = 10
 - 会在每个分片上先都获取 1000 个文档。然后，通过 Coordinating Node 聚合所有结果。最后再通过排序选取前 1000 个文档
 - 页数越深，占用内存越多。为了避免深度分页带来的内存开销，ES 有一个设定，默认限定到 10000 个文档

```
1 GET /tcpproxy_201125/_search
2 {
3   "query": {
4     "match": {
5       "IP": "222.111.11.58"
6     }
7   },
8   "from": 0,
9   "size": 5,
10  "sort": [
11    {
12      "StartTime": {
13        "order": "desc"
14      }
15    }
16  ]
17 }
```

size 大小不能超过 `index.max_result_window` 参数的设置, 默认为: 10000

- 实时获取顶部的部分文档, 如最新的订单. 用 `from` `to`
- 需要全部文档, 如导出数据, 用 `Scroll`
- 深度分页, 用 `Search After`

H3 4. 排序

```
1 GET xy_201125,xy_201126/_search
2 {
3   "query": {
4     "term": {
5       "data.game.game_id": {
6         "value": 1616
7       }
8     }
9   },
10  "sort": [
11    {
12      "data.node_id": {
13        "order": "asc"
14      }
15    }
16  ]
17 }
```

```

14     },
15     "data.speed_report_0.delays": {
16         "order": "desc",
17         "mode": "avg"
18     }
19 }
20 ]
21 }

```

若 `order` 有多个值(数组), 取平均值排序.

PS: 数组中数据类型必须一致. [1, 2, 3] 查看 `mapping`:

```

1  "delays": {
2    "type": "long"
3  }

```

多索引: `_all`, `xy_*`, `xy_20*`, `xy_2011*`, `xy_201030`

建议: 去除一切不必要的嵌套, 文档只使用一级 `key-value`

H2 聚合

一些概念:

- `Buckets` 桶, 满足条件的文档集合, 类似于 `GROUP BY`
- `Metrics` 指标, 对桶内文档进行统计计算, 类似于 `COUNT()`, `SUM()`
- `Pipeline` 聚合其他聚合的输出以及相关指标的聚合
- `aggregations` 缩写为 `aggs` 聚合关键字

```

1  "aggregations" : {
2    "<aggregation_name>":
3      "<aggregation_type>":
4        "<aggregation_body>"
5      }
6    [, "meta" : { [<meta_data_body> ] } ]?
7    [, "aggregations" : { [<sub_aggregation>]+ } ]?
8  }
9  [, "<aggregation_name_2>" : { ... } ]*
10 }

```

H3 1. 中欧峰值带宽统计

前提: 每分钟采集每设备每接口的数据到 ES.

思路: 交换机接口 `10GE1/0/111` 的流入流出带宽最大值 `max`, `stats`

```

1  GET bw_collect_201126/_search
2  {
3    "size": 0,

```

```

4   "query": {
5     "match": {
6       "interface.keyword": "10GE1/0/111"
7     }
8   },
9   "aggs": {
10    "max_kbps_in": {
11      "max": {
12        "field": "kbps_out"
13      }
14    }
15  }
16 }

```

```

1  "aggregations": {
2    "max_kbps_in": {
3      "value": 2637288
4    }
5  }

```

也可以按峰值排序取前 5 条数据:

```

1  GET bw_collect_201126/_search
2  {
3    "size": 5,
4    "query": {
5      "match": {
6        "interface.keyword": "10GE1/0/111"
7      }
8    },
9    "sort": [
10     {
11       "kbps_in": {
12         "order": "desc"
13       }
14     }
15   ],
16   "_source": ["client_ip", "kbps_out", "time"]
17 }

```

```

1  {
2    "_index": "bw_collect_201126",
3    "_type": "bw_collect_201126",
4    "_id": "RpxSA3YBtrCG7JwuhwMm",
5    "_score": null,
6    "_source": {
7      "kbps_out": 3677867,
8      "client_ip": "100.119.112.200",
9      "time": "2020-11-26T11:34:00+08:00"
10   },
11   "sort": [
12     696326
13   ]
14 }

```

H3 2. 统计美国用户分布情况

terms, cardinality

```
1 GET userspd_201126/_search
2 {
3   "size": 0,
4   "query": {
5     "bool": {
6       "must": [
7         {
8           "term": {
9             "country_c.keyword": "united states"
10          }
11        }
12      ],
13      "must_not": [
14        {
15          "term": {
16            "prov_c.keyword": "unkown"
17          }
18        }
19      ]
20    }
21  },
22  "aggs": {
23    "by_prov_c": {
24      "terms": {
25        "field": "prov_c.keyword",
26        "size": 100
27      },
28      "aggs": {
29        "by_user_distinct": {
30          "cardinality": {
31            "field": "u_name.keyword"
32          }
33        }
34      }
35    }
36  }
37 }
```

```
1   "aggregations": {
2     "by_prov_c": {
3       "doc_count_error_upper_bound": 0,
4       "sum_other_doc_count": 0,
5       "buckets": [
6         {
7           "key": "california",
8           "doc_count": 1396244,
9           "by_user": {
10            "value": 2728
11          }
12        },
13        {
```

```

14         "key": "new york",
15         "doc_count": 1271922,
16         "by_user": {
17             "value": 3456
18         }
19     },

```

H3 3. 按游戏和区服统计加速人数

terms, cardinality 多重聚合

查询时间	游戏id	区服id	加速人数
2020-11-27 09:39:33	23646	10344	23657
2020-11-27 09:39:33	23646	18379	2146
2020-11-27 09:39:33	23646	836	1592
2020-11-27 09:39:33	23646	3644	658
2020-11-27 09:39:33	23646	21152	724
2020-11-27 09:39:33	23646	3645	216
2020-11-27 09:39:33	23646	21161	19
2020-11-27 09:39:33	1616	25418	21820
2020-11-27 09:39:33	1616	1273	11039
2020-11-27 09:39:33	1616	17091	1127

```

1 GET xy_201127/_search
2 {
3   "size": 0,
4   "_source": false,
5   "query": {
6     "bool": {
7       "filter": [
8         {
9           "term": {
10            "type.keyword": "speed_report"
11          }
12        },
13        {
14          "terms": {
15            "data.game_id": [1616, 23646]
16          }
17        },
18        {
19          "range": {
20            "_ctime": {
21              "gte": "2020-11-27T09:34:33",
22              "lt": "2020-11-27T09:39:33"
23            }
24          }
25        }
26      ]
27    }

```



```

28     },
29     "aggs": {
30         "aggs_hits": {
31             "terms": {
32                 "field": "data.game_id",
33                 "size": 10000
34             },
35             "aggs": {
36                 "aggs_gamearea": {
37                     "terms": {
38                         "field": "data.game_area_id",
39                         "size": 10000
40                     },
41                     "aggs": {
42                         "distinct_user_name": {
43                             "cardinality": {
44                                 "field": "data.user_name.keyword"
45                             }
46                         }
47                     }
48                 }
49             }
50         }
51     }
52 }

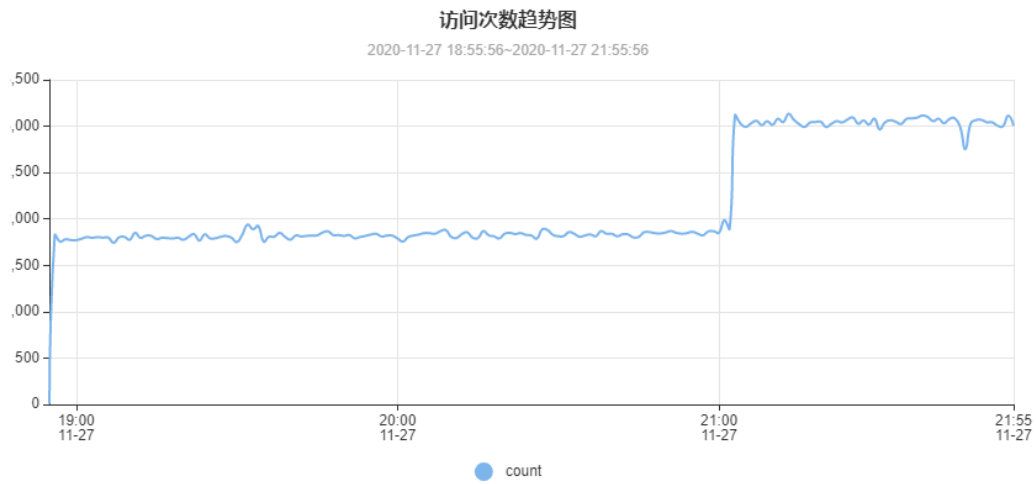
```

```

1     "aggregations": {
2         "aggs_hits": {
3             "doc_count_error_upper_bound": 0,
4             "sum_other_doc_count": 0,
5             "buckets": [
6                 {
7                     "key": 23646,
8                     "doc_count": 117709,
9                     "aggs_gamearea": {
10                         "doc_count_error_upper_bound": 0,
11                         "sum_other_doc_count": 0,
12                         "buckets": [
13                             {
14                                 "key": 10344,
15                                 "doc_count": 116209,
16                                 "distinct_user_name": {
17                                     "value": 23657
18                                 }
19                             },

```

H3 4. 域各单位时间内访问次数趋势图



`date_histogram`, 按时间间隔聚合, 日期直方图

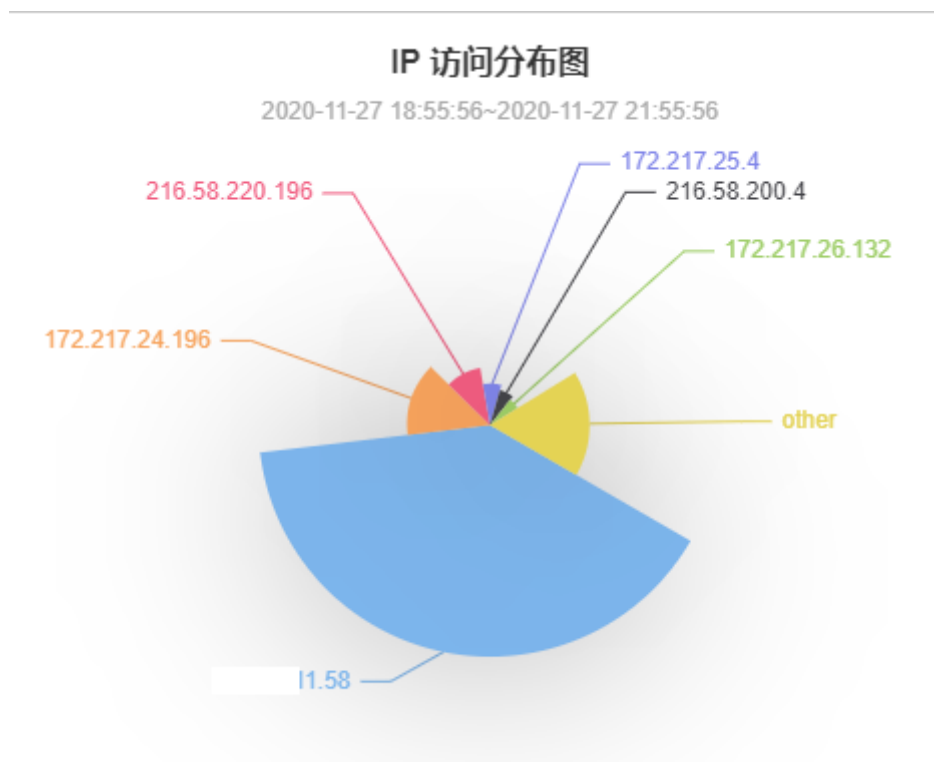
```
1 GET tcpproxy_201127/_search
2 {
3   "query": {
4     "bool": {
5       "filter": [
6         {
7           "range": {
8             "StartTime": {
9               "gte": "2020-11-27T18:55:00+08:00",
10              "lt": "2020-11-27T21:55:00+08:00"
11            }
12          }
13        },
14        {
15          "term": {
16            "Domain.keyword": "www.google.com"
17          }
18        }
19      ],
20      "must_not": [
21        {
22          "term": {
23            "Vppid.keyword": ""
24          }
25        }
26      ]
27    }
28  },
29  "aggs": {
30    "aggs_hits": {
31      "date_histogram": {
32        "field": "StartTime",
33        "interval": "minute"
34      }
35    }
36  },
37  "size": 0
38 }
```

```

1  "aggregations": {
2    "aggs_hits": {
3      "buckets": [
4        {
5          "key_as_string": "2020-11-27T10:55:00.000Z",
6          "key": 1606474500000,
7          "doc_count": 1643
8        },
9        {
10         "key_as_string": "2020-11-27T10:56:00.000Z",
11         "key": 1606474560000,
12         "doc_count": 1684
13       },

```

H3 5. 指定时间范围内域名解析结果 IP 访问分布图



```

1  GET tcpproxy_201127/_search
2  {
3    "query": {
4      "bool": {
5        "filter": [
6          {
7            "range": {
8              "StartTime": {
9                "gte": "2020-11-27T18:55:00+08:00",
10               "lt": "2020-11-27T21:55:00+08:00"
11             }
12           }
13         ],
14         {
15           "term": {
16             "Domain.keyword": "www.google.com"
17           }

```

```

18     }
19   ],
20   "must_not": [
21     {
22       "term": {
23         "Vpppid.keyword": {
24           "value": ""
25         }
26       }
27     }
28   ]
29 }
30 },
31 "aggs": {
32   "aggs_hits": {
33     "terms": {
34       "field": "IP.keyword",
35       "size": 20
36     }
37   }
38 },
39 "size": 0
40 }

```

```

1   "aggregations": {
2     "aggs_hits": {
3       "doc_count_error_upper_bound": 58,
4       "sum_other_doc_count": 11865,
5       "buckets": [
6         {
7           "key": "111.11.11.58",
8           "doc_count": 175187
9         },
10        {
11          "key": "172.22.124.96",
12          "doc_count": 126836
13        },

```

- `doc_count_error_upper_bound` 返回结果之外还有聚合没有返回
- `sum_other_doc_count` 此次聚合没有统计到的文档数

H3 6. 智能DNS服务器解析情况分析

源数据:

```

1  {
2    "conf_records": [
3      "5.5.5.5",
4      "6.6.6.6"
5    ],
6    "dns_check_result": "True",
7    "dns_domain": "monitor.yw.test",
8    "dns_ip": "123.234.111.222",
9    "dns_name": "上海LB",

```

```

10     "local_records": [
11         "5.5.5.5",
12         "6.6.6.6"
13     ]
14 }

```

terms, top_hits

```

1 GET dns_monitor_201126/_search
2 {
3     "size": 0,
4     "aggs": {
5         "dns_check": {
6             "terms": {
7                 "field": "dns_check_result.keyword",
8                 "order": {
9                     "_count": "asc"
10                }
11            },
12            "aggs": {
13                "dns_domain": {
14                    "terms": {
15                        "field": "dns_domain.keyword",
16                        "order": {
17                            "_count": "desc"
18                        }
19                    },
20                    "aggs": {
21                        "dns_name": {
22                            "terms": {
23                                "field": "dns_name.keyword",
24                                "order": {
25                                    "_count": "desc"
26                                }
27                            },
28                            "aggs": {
29                                "top": {
30                                    "top_hits": {
31                                        "size": 1
32                                    }
33                                }
34                            }
35                        }
36                    }
37                }
38            }
39        }
40    }
41 }

```

```

1     "aggregations": {
2         "dns_check": {
3             "doc_count_error_upper_bound": 0,
4             "sum_other_doc_count": 0,
5             "buckets": [
6                 {

```

```

7      "key": "True",
8      "doc_count": 11206,
9      "dns_domain": {
10         "doc_count_error_upper_bound": 0,
11         "sum_other_doc_count": 0,
12         "buckets": [
13             {
14                 "key": "monitor.yw.test",
15                 "doc_count": 8338,
16                 "dns_name": {
17                     "doc_count_error_upper_bound": 0,
18                     "sum_other_doc_count": 0,
19                     "buckets": [
20                         {
21                             "key": "上海DNS1",
22                             "doc_count": 1434,
23                             "top": {
24                                 "hits": {
25                                     "total": 1434,
26                                     "max_score": 1,
27                                     "hits": [
28                                         {
29                                             "_index": "dns_monitor_201126",
30                                             "_type": "dns_monitor_201126",
31                                             "_id": "TyC8_nUBtrCG7JwuGl18",
32                                             "_score": 1,
33                                             "_source": {
34                                                 "_cip": "222.111.222.111",
35                                                 "_ctime": "2020-11-26T00:09:38Z",
36                                                 "_gtime": "2020-11-
26T00:09:38+08:00",
37                                                 "conf_records": [
38                                                     "5.5.5.5",
39                                                     "6.6.6.6"
40                                                 ],
41                                                 "dns_check_result": "True",
42                                                 "dns_domain": "monitor.yw.test",
43                                                 "dns_ip": "123.234.111.222",
44                                                 "dns_name": "上海DNS1",
45                                                 "local_records": [
46                                                     "5.5.5.5",
47                                                     "6.6.6.6"
48                                                 ]
49                                             }
50                                         }
51                                     ]
52                                 }
53                             }
54                         },

```

top_hits Ref: <https://blog.csdn.net/ctwy291314/article/details/82773180>

H3 7. 节点报警次数 TOP 10

top_hits

	节点id	节点名称	节点IP	告警次数
1			183.129.	258
2	113	浙江节点4	122.224.	257
3	396	华南-国际-396	139.159.	49
4	219	华东-国际16	101.132.	41
5	484	华北-国际-484	117.78.3	28

```

1 GET /monitor_alarm_info_201127/_search
2 {
3   "size": 0,
4   "query": {
5     "bool": {
6       "filter": [
7         {
8           "terms": {
9             "code": ["nagios", "smokeping", "system_monitor"]
10          }
11        }
12      ]
13    }
14  },
15  "aggs": {
16    "node_ip": {
17      "terms": {
18        "field": "node_ip.keyword",
19        "order": {
20          "_count": "desc"
21        }
22      },
23      "aggs": {
24        "top": {
25          "top_hits": {
26            "size": 1
27          }
28        }
29      }
30    }
31  }
32 }

```

H3 8. 用户 B 段测速报警



scripted_metric:

- `init_script` 收集任何文件之前执行, 可选
- `map_script` 每个收集的文档执行一次, 必须
- `combine_script` 文档收集完成后, 每个分片执行一次, 必须
- `reduce_script` 所有分片均返回结果后, 在协调节点上执行一次, 必须

```
1 GET userspd_201127/_search
2 {
3   "size": 0,
4   "query": {
5     "bool": {
6       "filter": [
7         {
8           "range": {
9             "rcv_time": {
10              "gte" : "now-1h"
11            }
12          }
13        },
14        {
15          "term": {
16            "country_c": "china"
17          }
18        },
19        {
20          "term": {
21            "country_s": "china"
22          }
23        }
24      ],
```



```

25     "must_not": [
26         {
27             "term": {
28                 "n_avg": 65535
29             }
30         },
31         {
32             "term": {
33                 "prov_c.keyword": "china"
34             }
35         },
36         {
37             "term": {
38                 "prov_s.keyword": "china"
39             }
40         }
41     ]
42 },
43 },
44 "aggs": {
45     "by_prov_c": {
46         "terms": {
47             "field": "prov_c.keyword",
48             "size": 2
49         },
50         "aggs": {
51             "by_prov_s": {
52                 "terms": {
53                     "field": "prov_s.keyword",
54                     "size": 2
55                 },
56                 "aggs": {
57                     "los_percent": {
58                         "scripted_metric": {
59                             "init_script": "params._agg.map = new HashMap();",
60                             "map_script":
61 "if(!params._agg.map.containsKey('total'))params._agg.map.put('total',
62 ,0);if(!params._agg.map.containsKey('los'))params._agg.map.put('los',
63 ,0);
64 params._agg.map.put('total',params._agg.map['total']+1);if(doc['n_los
65 '].value > 0 ) params._agg.map['los'] = params._agg.map['los'] + 1",
66                             "combine_script": "return params._agg.map",
67                             "reduce_script": "int total = 0; int los=0; for (a in
68 params._aggs) { total += a['total'];los+=a['los']} return
69 Math.floor(los*0.1*1000/total);"
70                     }
63                 }
64             }
65         }
66     }
67 }
68 }
69 }
70 }

```

```

1     "aggregations": {
2         "by_prov_c": {
3             "doc_count_error_upper_bound": 844905,

```

```

4      "sum_other_doc_count": 22258607,
5      "buckets": [
6        {
7          "key": "js",
8          "doc_count": 2910301,
9          "by_prov_s": {
10           "doc_count_error_upper_bound": 3797,
11           "sum_other_doc_count": 659311,
12           "buckets": [
13             {
14               "key": "sh",
15               "doc_count": 1411625,
16               "los_percent": {
17                 "value": 2
18               }
19             },
20             {
21               "key": "gd",
22               "doc_count": 839365,
23               "los_percent": {
24                 "value": 4
25               }
26             }
27           ]
28         }
29       },

```

H3 9. 统计 LOL 加速前小于 60ms 用户平均延时

script, stats, v6.5 支持 median_absolute_deviation

```

1 GET xy_201125/_search
2 {
3   "size": 0,
4   "query": {
5     "bool": {
6       "must": [
7         {
8           "match": {
9             "data.game_id": "1616"
10          }
11        },
12        {
13          "match": {
14            "type.keyword": "speed_report"
15          }
16        }
17      ],
18      "filter": [
19        {
20          "script": {
21            "script": {
22              "source": "doc['data.speed_report_0.avg_delay'].value >
23              0",
24              "lang": "painless"
25            }
26          }
27        }
28      ]
29    }
30  }
31 }

```

```

25         }
26     },
27     {
28         "script": {
29             "script": {
30                 "source": "doc['data.speed_report_0.avg_delay'].value <
60",
31                 "lang": "painless"
32             }
33         }
34     }
35 ]
36 }
37 },
38 "aggs": {
39     "stats_delay": {
40         "stats": {
41             "field": "data.speed_report_0.avg_delay"
42         }
43     }
44 }
45 }

```

```

1  "aggregations": {
2      "avg_delay": {
3          "count": 14840620,
4          "min": 1,
5          "max": 59,
6          "avg": 30.474570199685164,
7          "sum": 1147515814
8      }
9  }

```

还有好多招式, Ref:

- <https://pdf.us/2018/05/16/1050.html>
- <https://learnku.com/docs/elasticsearch73/7.3/article-11/6889>
- <https://xiaoxiami.gitbook.io/elasticsearch/ji-chu/36aggregationsju-he-fen-679029>

(网络部.2020.11.28)

fufu