初探 ElasticSearch 聚合

张进军

2018-9-12

## 一、基础概念

Elasticsearch是一个分布式、可扩展、实时的搜索与数据分析引擎。它能全文检索，也支持结构化搜索、地理位置和对象间关联关系等。

ES是面向文档的，具体来说它以JSON作为文档的序列化格式。ES存储并索引文档使之可以被检索。所以在ES中是对文档进行索引、检索、排序和过滤的，而不像关系型数据库对行列数据进行操作。

**索引（index）**：有共同特性被分组到一起的文档集合。可定义别名。

**类型（type）**：在索引中对数据进行逻辑细分，这样的子分区就是一个类型。类型由名称和映射组成。

**映射（mapping）**：就像数据库中的 schema ，描述了文档可能具有的字段或属性、每个字段的数据类型，以及如何索引和存储这些字段的。

一个索引下的类型间的字段有不同，但最好大部分字段能非常相似。实际上，一个索引下所有类型之间的相同字段共享相同的映射，所以不存在两个类型中同名字段的映射不同。

一个文档不仅仅包含它的数据 ，也包含元数据——有关文档的信息。其中有三个是必须的元数据元素：\_index、\_type、\_id，由这三个唯一标识一个文档。

聚合相关：

**桶（bucket）**：满足特定条件的文档的集合。

**指标（metric）**：对桶内的文档进行统计计算。

## 二、数据准备

说明：以下实例基于 ElasticSearch V6.1.1。

创建就诊信息索引test\_medical，类型test\_info，类型字段包括：

|  |  |
| --- | --- |
| event\_no | 就诊事件编号 |
| event\_type | 就诊类型，门诊、住院或体检 |
| event\_date | 就诊日期 |
| org\_area | 地区 |
| org\_code | 机构编码 |
| org\_name | 机构名称 |
| dept\_code | 科室编码 |
| card\_id | 病人身份证号 |
| patient\_name | 病人姓名 |
| patient\_age | 病人年龄 |
| sex | 病人性别编码 |
| sex\_value | 病人性别 |
| cost\_info.type | 费用类型编码 |
| cost\_info.type\_value | 费用类型名称 |
| cost\_info.money | 费用金额 |

索引、类型可先于数据创建，也可以插入数据时通过ES动态映射创建。

准备好的演示数据可通过Kibana 的DevTool插件执行批量接口语法导入：



## 三、实例讲解

### 基本函数统计

求均值、最小值、最大值、总和、总数。

例：

GET test\_medical/test\_info/\_search

{

"size":0,

"aggs": {

"age\_avg": {

"avg": {

"field": "patient\_age"

}

},

"age\_min": {

"min": {

"field": "patient\_age"

}

},

"age\_max": {

"max": {

"field": "patient\_age"

}

},

"patient\_count": {

"value\_count": {

"field": "patient\_name.keyword"

}

},

"cost\_sum": {

"sum": {

"field": "cost\_info.money"

}

},

"cost\_stats": {

"stats": {

"field": "cost\_info.money"

}

}

}

}

### 去重计数

使用cardinality 聚合。

例：

GET test\_medical/test\_info/\_search

{

"size":0,

"aggs":{

"distinct\_count": {

"cardinality": {

"field": "patient\_name.keyword"

}

}

}

}

### 去重查询

使用terms 聚合嵌套 top\_hits 聚合。

例：获取各个病人的就诊信息记录。

GET test\_medical/test\_info/\_search

{

"size": 0,

"aggs": {

"patient\_count": {

"terms": {

"field": "patient\_name.keyword",

"size": 10

},

"aggs": {

"type\_item\_list": {

"top\_hits": {

"from": 0,

"size":10,

"sort":{

"event\_date":{

"order":"desc"

}

}

}

}

}

}

}

}

### 百分比

使用bucket\_script 聚合。

例：统计每个月下，各个机构就诊费用占比

各个机构就诊费用/月就诊费用\*100。

GET test\_medical/test\_info/\_search

{

"size": 0,

"aggs": {

"cost\_per\_month": {

"date\_histogram": {

"field": "event\_date",

"interval": "month",

"format": "yyyy-MM-dd"

},

"aggs": {

"month\_cost\_sum": {

"sum": {

"field": "cost\_info.money"

}

},

"广丰\_count": {

"filter": {

"term": {

"org\_code.keyword": "49230064X"

}

},

"aggs": {

"gf\_cost\_sum": {

"sum": {

"field": "cost\_info.money"

}

}

}

},

"gf\_cost\_sum\_percentage": {

"bucket\_script": {

"buckets\_path": {

"gfCostSum": "广丰\_count>gf\_cost\_sum",

"monthCostSum": "month\_cost\_sum"

},

"script": "params.gfCostSum / params.monthCostSum \* 100"

}

},

"上饶\_count": {

"filter": {

"term": {

"org\_code.keyword": "492190516"

}

},

"aggs": {

"srds\_cost\_sum": {

"sum": {

"field": "cost\_info.money"

}

}

}

},

"srds\_cost\_sum\_percentage": {

"bucket\_script": {

"buckets\_path": {

"srdsCostSum": "上饶\_count>srds\_cost\_sum",

"monthCostSum": "month\_cost\_sum"

},

"script": "params.srdsCostSum / params.monthCostSum \* 100"

}

},

"鄱阳\_count": {

"filter": {

"term": {

"org\_code.keyword": "49236052X"

}

},

"aggs": {

"py\_cost\_sum": {

"sum": {

"field": "cost\_info.money"

}

}

}

},

"py\_cost\_sum\_percentage": {

"bucket\_script": {

"buckets\_path": {

"pyCostSum": "鄱阳\_count>py\_cost\_sum",

"monthCostSum": "month\_cost\_sum"

},

"script": "params.pyCostSum / params.monthCostSum \* 100"

}

}

}

}

}

}

### 关联查询

非规范化数据，即将需要的相关信息设计在一个type中。这也是在ES中最常用的关联方式，也便于数据检索和分析。

例：

GET test\_medical/test\_info/\_search

{

"query": {

"bool": {

"must": [

{

"match": {

"sex\_value.keyword": "女"

}

},

{

"match": {

"cost\_info.type\_value.keyword": "西药"

}

},

{

"range": {

"cost\_info.money": {

"gte": 100,

"lte": 300

}

}

}

]

}

}

}