Elasticsearch

Advanced Topics on NoSQL databases

A4 - S8

ESILV
nicolas.travers (at) devinci.fr

Chapter 1

Setting Movies dataset

For this practice work, we will use the elasticsearch indexing software.

1.1 cURL

cURL is a small executable which send/receive HTTP requests in command lines. You can specify headers by adding parameters to your command (XPUT/XGET/XPOST, -H"Content-Type...", –databinary...).

You can download cURL (already installed under Linux and MacOSX) here (better to use the SSL embedded version):
`https://curl.haxx.se/download.html`.

1.2 Elasticsearch & Kibana

The installation steps are little bit straightforward.

- Local installation: `https://www.elastic.co/downloads`
- Docker : name "elasticsearch-kibana" (@nshou v 6.5.4)

1.3 Import dataset with cURL

- Download the `movies_elastic.json.zip` data file[1], and unzip it
- Import it in a **command line** with:

```
curl -XPUT localhost:9200/_bulk -H"Content-Type: application/json" --data-binary @movies_elastic.json
```

[1] `https://devinci-online.brightspace.com/d2l/le/content/15379/viewContent/18377/View`

Chapter 2

DSL queries

For each query, you can write in two formats (when possible):

- With the HTTP GET method, with the q= parameter
- With the HTTP POST method, with the JSON document param "*query.json*" (called DSL queries):

```
curl -XPOST "localhost:9200/movies/movie/_search" -H"Content-Type: application/json"
          -d @query.json -o output.json
```

And then open this JSON document in your Web browser.

- You can also make those queries directly in Kibana in the "Dev Tools"

2.1 Simple queries

2.1.1 Every movies which title matches 'Star Wars' (match query),
Correction : On the direct REST API (Web browser):

```
_search?q=title:Star+Wars
```

Or by POST http request on the REST API with an input document (curl/kibana):

```
{"query":{"match":{"fields.title":"Star Wars"}}}
```

2.1.2 Try with exact match (match_phrase),
Correction :

```
{"query":{"match_phrase":{"fields.title":"Star Wars"}}}
```

2.1.3 Star Wars movies and Directors equal to 'George Lucas' (boolean query),
Correction :

```
{"query":{
  "bool": {
    "should": [
      { "match": { "fields.title": "Star Wars" }},
      { "match": { "fields.directors": "George Lucas" }}
    ]
}}}
```

Elasticsearch is a textual search engine, so results are ranked according to the score of relevance, thanks to "should". You can require to make it mandatory with "must":

```
{"query":{
  "bool": {
   "should": [
     { "match_phrase": { "fields.title": "Star Wars" }},
     { "match_phrase": { "fields.directors": "George Lucas" }}
   ]
}}}
```

2.1.4 Movies were 'Harrison Ford' played,
Correction : `_search?q=fields.actors:Harrison+Ford`

```
{  "query": {"match_phrase": {"fields.actors":"Harrison Ford"}}}
```

2.1.5 Movies were 'Harrison Ford' played with a plot containing 'Jones',
Correction :

```
{"query":{
  "bool": {
    "should": [
      { "match_phrase": { "fields.actors": "Harrison Ford" }},
      { "match": { "fields.plot": "Jones" }}
    ]
}}}
```

2.1.6 Movies were 'Harrison Ford' played with a plot containing 'Jones' but plots without containing 'Nazis'
Correction :

```
{"query":{
  "bool": {
    "must": [
      { "match_phrase": { "fields.actors": "Harrison Ford" }},
      { "match": { "fields.plot": "Jones" }}
    ],
    "must_not" : { "match" : {"fields.plot":"Nazis"}}
}}}
```

2.1.7 Movies of 'James Cameron' which rank is better than 1000 (boolean + range query)
Correction :

```
{"query":{
  "bool": {
    "must": [
      { "match_phrase": { "fields.directors": "James Cameron" }},
      { "range": { "fields.rank": {"lt":1000 }}}
    ]
}}}
```

2.1.8 Movies of 'James Cameron' which rating must be higher than 5
Correction :

```
{"query":{
  "bool": {
    "should": { "match_phrase": { "fields.directors": "James Cameron" }},
    "must": { "range": { "fields.rating": {"gt":5 }}}
}}}
```

2.1.9 Movies of 'James Cameron' which rating must be higher than 5 and which genre must not be 'Action' nor 'Drama'
Correction :

```
{"query":{
  "bool": {
    "should": { "match_phrase": { "fields.directors": "James Cameron" }},
    "must":{ "range": { "fields.rating": {"gt":5 }}},
    "must_not":[
      {"match":{"fields.genres":"Action"}},
      {"match":{"fields.genres":"Drama"}}
    ]
}}}
```

2.1.10 Movies of 'J.J. Abrams' which released date is mandatory between 2010 and 2015 (filtered query)
Correction :

```
{"query": {
  "bool":{
     "must": {"match": {"fields.directors": "J.J. Abrams"}},
     "filter": {"range": {"fields.release_date": {
                                    "from": "2010-01-01",
                                    "to": "2015-12-31"}}}
}}}
```

2.2 Aggregate queries

We wish now to do some aggregate queries on the index in order to extract some statistics.

2.2.1 Complex queries

2.2.1 Give for each year the number of movies,
Correction :

```
{"aggs" : {
  "nb_per_year" : {
    "terms" : {"field" : "fields.year"}
}}}
```

2.2.2 For each category (genres), give the number of movies. To take into account the whole text, you need to use "keyword" after the required field.
Correction : For the mapping, you must send a query on the 'movies' index:

```
{"aggs" : {
  "nb_per_category" : {
    "terms" : {"field" : "fields.genres.keyword"}
}}}
```

2.2.3 Give the average rating of movies,
Correction :

```
{"aggs" : {
  "avg_rating" : {
    "avg" : {"field" : "fields.rating"}
}}}
```

2.2.4 Give the average rating of George Lucas' movies,
Correction :

```
{
  "query" :{
    "match" : {"fields.directors" : "George Lucas"}
  }
  ,"aggs" : {
    "note_moyenne" : {
      "avg" : {"field" : "fields.rating"}
}}}
```

2.2.5 Count the number of movies for the given ranges of rating: 0-1.9, 2-3.9, 4-5.9...),
Correction :

```
{"aggs" : {
  "group_range" : {
    "range" : {
      "field" : "fields.rating",
      "ranges" : [
        {"to" : 1.9},
        {"from" : 2, "to" : 3.9},
        {"from" : 4, "to" : 5.9},
        {"from" : 6, "to" : 7.9},
        {"from" : 8}
      ]
    }
}}}
```

2.2.6 Number of distinct directors in adventures movies,
Correction :

```
{
  "query":{
    "match" : {"fields.genres" : "Adventure"}
  },
  "aggs" : {
    "nb_distinct" : {
      "cardinality" : {"field" : "fields.directors.keyword"}
    }
}}
```

2.2.2 Hard queries

2.2.1 Give the average rating per genre,
Correction :

```
{"aggs" : {
"group_genres" : {
"terms" : {
"field" : "fields.genres.keyword"
},
"aggs" : {
"avg_rating" : {
"avg" : {"field" : "fields.rating"}
}}
}}}
```

2.2.2 Give min, max and average rating for each genre,
Correction :

```
{"aggs" : {
"group_genres" : {
"terms" : {
"field" : "fields.genres.keyword"
},
"aggs" : {
"avg_rating" : {"avg" : {"field" : "fields.rating"}},
"min_rating" : {"min" : {"field" : "fields.rating"}},
"max_rating" : {"max" : {"field" : "fields.rating"}}
}
}}}
```

2.2.3 Give average movie's rank and average movie's rating for each director. Sort the result decreasingly on average rating,
Correction :

```
{"aggs" : {
"group_director" : {
"terms" : {
"field" : "fields.directors.keyword",
"size":10000,
"order" : { "avg_rating" : "desc" }
},
"aggs" : {
"avg_rank" : {
"avg" : {"field" : "fields.rank"}
},
"avg_rating" : {
"avg" : {"field" : "fields.rating"}
}
}
}}}
```

2.2.4 Give the terms occurrences extracted from each movie's title. The text value requires a specific mapping on the dataset stored in elasticsearch, see: `https://www.elastic.co/guide/en/elasticsearch/reference/current/fielddata.html`.
Correction :

```
PUT /movies/movie/_mapping
{ "properties": {
"fields.title": {
"type":     "text",
"fielddata": true
}}}
```

```
{"aggs" : {
"occ_per_term_in_title" : {
"terms" : {"field" : "fields.title"}
}}}
```

The aggregation is made on each occurrence of the ”title” words but Stop words are also diplayed in the result set.

2.2.5 Most significant terms in plots of George Lucas movies,
Correction :

```
{
"query" :{
"match_phrase" : {"fields.directors" : "George Lucas"}
},
"aggs" : {
"terms_significatifs" : {
"significant_terms" : {"field" : "fields.plot"}
}},
      "size":1
}
```

The result with ”match” is interesting to look at.

Chapter 3

Bonus (hard queries)

3.1 Synonyms

We can insert synonyms inside the search engine in order to change the mapping between words. For this, we need to define a new **analyzer**[1]

3.1.1 List of synonyms

First, create a file which contains the synonyms (called here "*SYNONYMS.TXT*"). The structure is detailed here: `https://www.elastic.co/guide/en/elasticsearch/guide/current/synonym-formats.html`

Here is an example:

```
"united states            => usa",
"united states of america => usa"
```

Put them in the following folder: `elasticsearch/config/analysis/SYNONYMS.TXT`

3.1.2 Change the analyzer

- Close your "movies" index:
  `POST movies/_close`
- Add a tokenizer:

```
PUT movies/_settings
{
    "settings": {
        "index" : {
            "analysis" : {
                "analyzer" : {
                    "MY_SYNONYMS" : {
                        "tokenizer" : "whitespace",
                        "filter" : ["graph_synonyms"]
                    }
                },
                "filter" : {
                    "graph_synonyms" : {
                        "type" : "synonym_graph",
                        "synonyms_path" : "analysis/SYNONYMS.TXT"
                    }
                }
            }
        }
    }
}
```

- Open your "movies" index:
  `POST movies/_open`

3.1.3 Query with synonyms

Use the analyzer

[1] `https://www.elastic.co/guide/en/elasticsearch/guide/current/using-synonyms.html`

```
POST movies/_search
{ "_source": {"includes":["..."]},
   "query": {
       "match" : {
           "fields.title": {
               "query" : "United States",
               "analyzer": "MY_SYNONYMS"
           }
       }
   }
}
```

3.2 Extra bonus

Other things you can look at:

- Use stemming and language analyzers:
  `https://www.elastic.co/guide/en/elasticsearch/reference/6.6/analysis-lang-analyzer.html`
- N-Grams tokenizers:
  `https://www.elastic.co/guide/en/elasticsearch/reference/6.6/analysis-ngram-tokenizer.html`
- Scripting with Painless:
  `https://www.elastic.co/guide/en/elasticsearch/painless/6.6/painless-examples.html`
- Add X-Pack plugins to elasticsearch:
  `https://www.elastic.co/guide/en/elasticsearch/reference/current/xpack-api.html`
- Integrate your **dataset** with *Logstash*:
  `https://www.elastic.co/guide/en/logstash/current/getting-started-with-logstash.html`