



MESIIN480323 : No SQL 19/01

PROJECT REPORT

On

ElasticSearch

reuters.json

GABRIEL CHAIX & MALO LEROY & CHARLES TERRIER

ÉCOLE SUPERIEURE D'INGENIEURS LEONARD DE VINCI

ANNEE 2023 / 2024

INDEX

INTRODUCTION AND PROBLEM STATEMENT	2
I. Data Importation.....	3
II. Data cleaning	4
III. Querys	5
A. Simple querys (6)	5
1. List of two first articles	Erreur ! Signet non défini.
2. List of texts titles sorted by date	Erreur ! Signet non défini.
3. Titles, id and dateline of texts where id "120"	Erreur ! Signet non défini.
4. Count of articles whose place is France.	Erreur ! Signet non défini.
5. Count of publications whose "texts" is null.	Erreur ! Signet non défini.
6. Give the number articles written by each author ?.....	8
B. Complex querys (2)	8
1. How much articles have been published per month?	8
2. For each topic, give the average number of people that wrote on. Sort the result decreasingly.....	8
C. Hard query (1)	9
1. Between 1984 and 2024, how long have been the articles for a certain topic for each country? Sort by average body length.	Erreur ! Signet non défini.

INTRODUCTION AND PROBLEM STATEMENT

Reuters is a news agency founded in 1851 in London. It is one of the global and generalist news agencies, an activity that represents a part of its revenue, mainly devoted to financial information. It is one of the largest news agencies in the world. We want to create a MongoDB database with reuters articles and query it.

I. Data Importation

In a first manner, we'll setup the docker container with nshu image and make sure everything is working properly, we try to connect on port 5601.

```
~IT/NoSQL — Gabriel@Alfred — ~IT/NoSQL — zsh
❯ docker ps
CONTAINER ID IMAGE STATUS PORTS NAMES
7ac9ecacd7bb nshou/elasticsearch-kibana:latest "bash entrypoint.sh" 3 hours ago
Up 3 hours 0.0.0.0:5601->5601/tcp, 0.0.0.0:9200->9200/tcp boring_banach
~/IT/NoSQL | batm 12 ➜
```

Now we must import our data using python to write a script. To do this, we run the following code modify our [json file](#) to match the indexing bulk format of elastic search.

```
import json

# Open the original file for reading
with open('reuters.json', 'r') as original_file:
    data = original_file.readlines()

with open('indexed_reuters.json', 'w') as indexed_file:
    for i, line in enumerate(data):
        obj = json.loads(line)
        # Remove the "_id" field
        obj.pop('_id', None)
        # Create the index line
        index_line = '{"index":{"_index":"articles","_id":"' + str(i + 1) + '"}}\n'
        indexed_file.write(index_line)
        # Write the JSON object to the indexed file
        json.dump(obj, indexed_file)
        indexed_file.write('\n')
```

Then here is the formatting script we've wrote



After executing the script, goes to the format we needed.

<pre>1 { "_id" : 1, "date" : "26-FEB-1987 15:01:01.79", "topics" : "coco" 2 { "_id" : 2, "date" : "26-FEB-1987 15:02:20.00", "topics" : "", "seq_n 3 { "_id" : 3, "date" : "26-FEB-1987 15:03:27.51", "topics" : "", "seq_n 4 { "_id" : 4, "date" : "26-FEB-1987 15:07:13.72", "topics" : "", "seq_n 5 { "_id" : 5, "date" : "26-FEB-1987 15:10:44.60", "topics" : "grai 6 { "_id" : 6, "date" : "26-FEB-1987 15:14:36.41", "topics" : "veg-</pre>	<pre>1 {"index":{"_index":"articles","_id":1}} 2 {"date": "26-FEB-1987 15:01:01.79", "topics": "cocoa", 3 {"index":{"_index":"articles","_id":2}} 4 {"date": "26-FEB-1987 15:02:20.00", "topics": "", "seq_n 5 {"index":{"_index":"articles","_id":3}} 6 {"date": "26-FEB-1987 15:03:27.51", "topics": "", "seq_n 7 {"index":{"_index":"articles","_id":4}}</pre>
--	---

We can now import the dataset.

```
curl --insecure -XPOST -u elastic:elastic https://localhost:9200/_bulk -H "Content-Type: application/json" --data-binary @indexed_reuters.json
{"errors":false,"took":25838,"items":[{"index":{"_index":"articles","_id":1,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":1,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":2,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":2,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":3,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":3,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":4,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":4,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":5,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":5,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":6,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":6,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":7,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
o":7,"primary_term":1,"status":201}},{("index":{"_index":"articles","_id":8,"_version":1,"result":{"created":true,"shards":{"total":2,"successful":1,"failed":0}},"_seq_n
```

After importing all the rows, we make sure that we have the count and that each row is correct.

```
curl --insecure -XGET -u elastic:elastic "https://localhost:9200/articles/_count"
{"count":21495,"shards": {"total":1,"successful":1,"skipped":0,"failed":0}}
```

```
1 GET /articles/_doc/1
2 {
3   "_index": "articles",
4   "_id": "1",
5   "_version": 1,
6   "_seq_no": 0,
7   "_primary_term": 1,
8   "found": true,
9   "_source": {
10     "date": "26-FEB-1987 15:01:01.79",
11     "topics": "cocoa",
12     "places": "el-salvador usa uruguay",
13     "people": "",
14     "orgs": "",
15     "exchanges": "",
16     "companies": "",
17     "text": {
18       "title": "BAHIA COCOA REVIEW",
19       "dateline": "SALVADOR, Feb 26 -",
20       "body": "Showers continued throughout
the month in the Bahia region, thou
```

Using Kibana compass, we can see everything looks alright.

II. Data cleaning

The only cleaning we success to do had been to update all date entry to the valid date type. Compared to other noSQL language, we found hard to work with elastic search.

```
1 PUT /articles/_mapping
2 {
3   "properties": {
4     "date": {
5       "type": "date",
6       "format": "dd-MMM-yyyy HH:mm:ss.SSS"
7     }
8   }
9 }
10 POST /articles/_refresh
```

III. Querys

A. Simple querys (6)

1. Every article which title match "financial"

The screenshot shows the Elasticsearch Dev Tools Console interface. The URL is `localhost:5601/app/dev_tools#/console`. The search bar contains the query `Find apps, content, and more.`. The console tab is selected. The history shows a GET request to `/articles/_search` with the following JSON body:

```

1 GET /articles/_search
2 {
3   "query": {
4     "match": {
5       "text.title": "financial"
6     }
7   }
8 }

```

The response status is `200 - OK` and it took `391 ms`. The results show one document with the following fields:

```

{
  "_index": "articles",
  "_id": "5121",
  "_score": 1.0,
  "_source": {
    "date": "19-MAR-1987 14:43:25.91",
    "topics": "acq",
    "places": "usa",
    "people": "",
    "orgs": "",
    "exchanges": "",
    "companies": "",
    "text": {
      "title": "FIRST FINANCIAL CDN",
      "dateline": "ATLANTA, March 19 -",
      "body": "First Financial Management corp said it has offered to acquire Comdata Network Inc for 18 dlr$ per share in cash and stock, or a total of about 342.7 mln dlr$. The company said for each Comdata share it would exchange half a First Data share and enough cash to bring the total value up to 18 dlr$ per share, provided that the market price of First"
    }
  }
}

```

2. Every article which exact match "financial officer"

The screenshot shows the Elasticsearch Dev Tools Console interface. The URL is `localhost:5601/app/dev_tools#/console`. The search bar contains the query `Find apps, content, and more.`. The console tab is selected. The history shows a GET request to `/articles/_search` with the following JSON body:

```

1 GET /articles/_search
2 {
3   "query": {
4     "match_phrase": {
5       "text.title": "financial officer"
6     }
7   }
8 }

```

The response status is `200 - OK` and it took `886 ms`. The results show one document with the following fields:

```

{
  "_index": "articles",
  "_id": "5121",
  "_score": 13.594239,
  "_source": {
    "date": "13-MAR-1987 18:22:03.26",
    "topics": "",
    "places": "usa",
    "people": "",
    "orgs": "",
    "exchanges": "",
    "companies": "",
    "text": {
      "title": "MANVILLE QMAN NAMES FINANCIAL OFFICER",
      "dateline": "DENVER, March 13 -",
      "body": "Manville Corp said it named John"
    }
  }
}

```

3. Articles written by volcker

```

1 GET /articles/_search
2 {
3   "query": {
4     "match": {
5       "people": "volcker"
6     }
7   }
8 }

```

The screenshot shows the Elasticsearch Dev Tools Console interface. A search query is being run against the 'articles' index. The query uses a 'match' query on the 'people' field with the value 'volcker'. The response is displayed in the right pane, showing a single result. The result is a document with the following fields:

- _index: articles
- _id: 330
- _score: 3.0595336
- _source: {
 date: "26-FEB-1987 18:36:39.11",
 topics: "money-supply",
 places: "usa",
 people: "volcker",
 orgs: "",
 exchanges: "",
 companies: "",
 text: {
 title: "FED DATA SUGGEST STABLE U.S. MONETARY POLICY"
 }
 }

4. Article with a body containing "market" but not "the"

```

1 GET /articles/_search
2 {
3   "query": {
4     "bool": {
5       "must": [
6         {
7           "match": {
8             "text.body": "market"
9           }
10        },
11        {
12          "must_not": [
13            {
14              "match": {
15                "text.body": "the"
16              }
17            }
18          ]
19        }
20      }
21 }

```

The screenshot shows the Elasticsearch Dev Tools Console interface. A search query is being run against the 'articles' index. The query uses a 'bool' query with a 'must' clause containing a 'match' query for 'market' and a 'must_not' clause containing a 'match' query for 'the'. The response is displayed in the right pane, showing a single result. The result is a document with the following fields:

- _index: articles
- _id: 19288
- _score: 3.1760411
- _source: {
 date: "19-JUN-1987 13:22:59.41",
 topics: "acc",
 places: "uk",
 people: "",
 orgs: "",
 exchanges: "",
 companies: "",
 text: {
 title: "REED SAYS IT HAS NO COMMENT ON HARCOURT RUMOURS",
 body: "Reed International Plc REED.L said it had no comment to make on U.K. stock market rumors that Harcourt Brace Jovanovich Inc HBJ may make a bid for the company in order to escape unwelcome offers from Robert Maxwell's British Printing and Communication Corp BPCL.L. A spokeswoman for Reed said earlier analysts forecasts that a bid for Reed will have to be about 700 mln stg were totally unrealistic, adding that its current market is about 2.7 billion stg. Reuter"
 }
 }

5. Find all articles that mention a purchase or sale of stocks by individuals or organizations, in particular articles related to agricultural commodities such as pork or livestock

```

1 GET /articles/_search
2 {
3   "query": {
4     "bool": {
5       "must": [
6         {
7           "match": {
8             "text.title": {
9               "query": "purchase OR sell*"
10            }
11          }
12        },
13        {
14          "terms": {
15            "topics": ["carcass", "livestock"]
16          }
17        }
18      ]
19    }
20  }
21 }

22 {
23   "total": 1,
24   "successful": 1,
25   "skipped": 0,
26   "failed": 0
27 },
28 "hits": [
29   {
30     "total": {
31       "value": 2,
32       "relation": "eq"
33     },
34     "max_score": 5.3863955,
35     "hits": [
36       {
37         "_index": "articles",
38         "_id": "16377",
39         "_score": 5.3863955,
40         "_ignored": [
41           "text.body.keyword"
42         ],
43         "_source": {
44           "date": "13-APR-1987 11:40:34.72",
45           "topics": "aca livestock carcass",
46           "places": "usa",
47           "people": "",
48           "orgs": "",
49           "exchanges": "",
50           "companies": "",
51           "text": {
52             "title": "SWIFT TO SELL SOUTH DAKOTA PORK PLANT",
53             "dateline": "DALLAS, April 13 -",
54           }
55         }
56       }
57     ]
58   }
59 }
60 
```

6. Articles whose place is France whose date is after mars-1987

```

1 GET /articles/_search
2 {
3   "query": {
4     "bool": {
5       "must": [
6         {
7           "match": {
8             "places": "france"
9           }
10        },
11        {
12          "range": {
13            "date": {
14              "gte": "1987-04-01T00:00:00Z"
15            }
16          }
17        }
18      ]
19    }
20  }
21 }

22 {
23   "total": 1,
24   "successful": 1,
25   "skipped": 0,
26   "failed": 0
27 },
28 "hits": [
29   {
30     "total": {
31       "value": 1,
32       "relation": "eq"
33     },
34     "max_score": 5.057277,
35     "hits": [
36       {
37         "_index": "articles",
38         "_id": "1674",
39         "_score": 5.057277,
40         "_ignored": [
41           "text.body.keyword"
42         ],
43         "_source": {
44           "date": "2-MAR-1987 18:03:14.58",
45           "topics": "gnp trade",
46           "places": "france",
47           "people": "",
48           "orgs": "oecd",
49           "exchanges": "",
50           "companies": "",
51           "text": {
52             "title": "FRANCE HAS LITTLE ROOM FOR MANOEUVRE, OECD SAYS",
53             "dateline": "PARIS, March 3 -",
54             "body": "French industry is failing to produce the goods its
55             markets need and its loss of competitiveness has left the
56             government little room for manoeuvre to reflate the economy, the
57             Organisation for Economic Cooperation and Development said.
58             With gross domestic product likely to grow only 2.1 pct this year
59             , the same rate as last year, unemployment could climb to 11.5
60             pct of the workforce by mid-1988, from its present 10.9 pct, it
61             said in an annual review of the French economy. The report
62             said the French economy was increasingly ill-adapted to demand
63             selling goods at uncompetitive relative prices on both domestic
64             and export markets. France's poor export performance
65             reflects a geographical bias in favour of markets less dynamic
66             than the average... And...A substantial loss of market share...In
67             "
68           }
69         }
70       }
71     ]
72   }
73 }
74 
```

B. Complex querys (2)

1. Give the number articles written by each author?

```

1 GET /articles/_search
2 {
3   "size": 0,
4   "aggs": {
5     "authors": {
6       "terms": {
7         "field": "people.keyword" // assuming
          people is analyzed and you want exact
          terms
8       }
9     }
10  }
11 }

```

```

25      "doc_count": 20347
26    },
27    {
28      "key": "reagan",
29      "doc_count": 129
30    },
31    {
32      "key": "james-baker",
33      "doc_count": 116
34    },
35    {
36      "key": "miyazawa",
37      "doc_count": 47
38    },
39    {
40      "key": "yeutter",
41      "doc_count": 46
42    },
43    {
44      "key": "lawson",
45      "doc_count": 45
46    },
47    {
48      "key": "volcker",
49      "doc_count": 45
50    },
51    {
52      "key": "nakasone",
53      "doc_count": 39

```

Within the authors aggregation, we specify another aggregation type called terms, which creates a separate bucket for each unique term found within the specified field. In this case, we want to create one bucket per unique author, so we choose the people field.

2. For each topic, give the average number of people that wrote on. Sort the result decreasingly.

```

1 POST /articles/_search
2 {
3   "size": 0,
4   "aggs": {
5     "topic_wise_avg_authors": {
6       "terms": {
7         "field": "topics.keyword"
8       },
9       "aggs": {
10         "average_authors": {
11           "avg": {
12             "field": "entities.people.keyword"
13           }
14         }
15       }
16     }
17   }
18 }

```

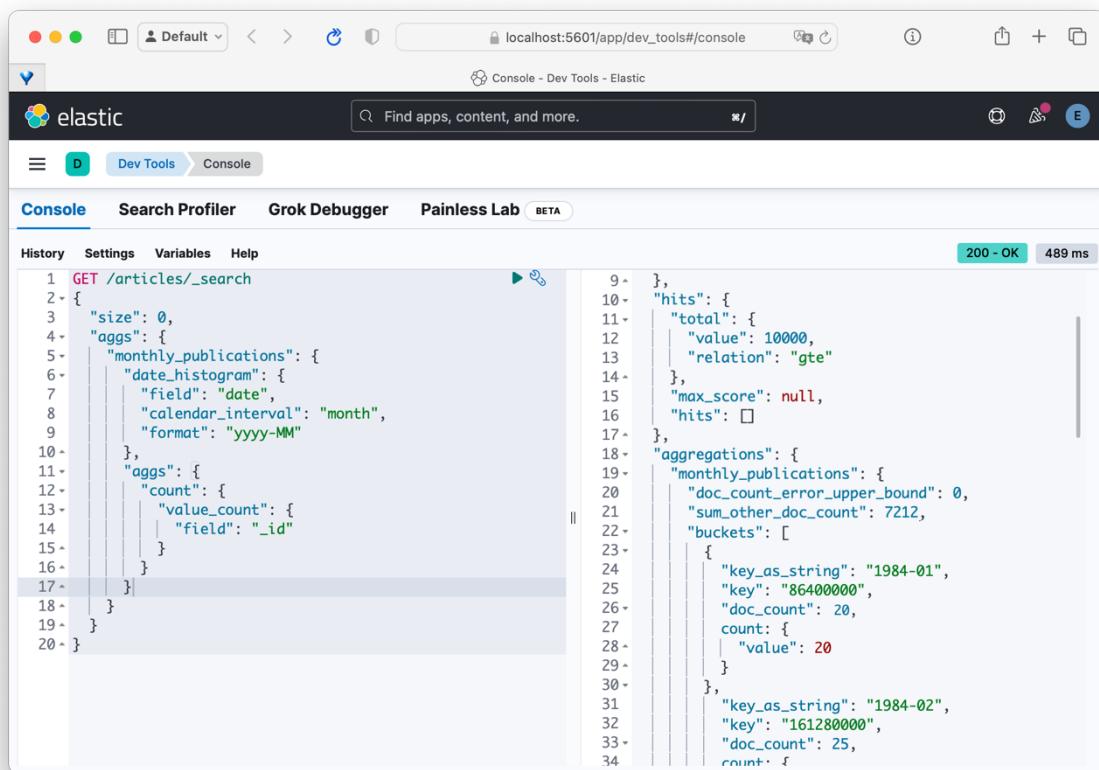
```

50      },
51    {
52      "key": "trade",
53      "doc_count": 360,
54      "average_authors": {
55        "value": 1.4
56      }
57    },
58    {
59      "key": "money-fx",
60      "doc_count": 303,
61      "average_authors": {
62        "value": 1.42857142857142
63      }
64    },
65    {
66      "key": "interest",
67      "doc_count": 285,
68      "average_authors": {
69        "value": 1.5
70      }
71    },
72    {
73      "key": "money-supply",
74      "doc_count": 160,
75      "average_authors": 1.5

```

C.Hard query (1)

1. How much articles have been published per month?



The screenshot shows the Elasticsearch Dev Tools Console interface. The URL is `localhost:5601/app/dev_tools#/console`. The search bar contains "Find apps, content, and more." The main area displays a code editor with a successful API response.

Code (Request):

```
1 GET /articles/_search
2 {
3   "size": 0,
4   "aggs": {
5     "monthly_publications": {
6       "date_histogram": {
7         "field": "date",
8         "calendar_interval": "month",
9         "format": "yyyy-MM"
10      },
11      "aggs": {
12        "count": {
13          "value_count": {
14            "field": "_id"
15          }
16        }
17      }
18    }
19  }
20 }
```

Response (JSON Output):

```
9  },
10  "hits": {
11    "total": {
12      "value": 10000,
13      "relation": "gte"
14    },
15    "max_score": null,
16    "hits": []
17  },
18  "aggregations": {
19    "monthly_publications": {
20      "doc_count_error_upper_bound": 0,
21      "sum_other_doc_count": 7212,
22      "buckets": [
23        {
24          "key_as_string": "1984-01",
25          "key": "86400000",
26          "doc_count": 20,
27          "count": {
28            "value": 20
29          }
30        },
31        {
32          "key_as_string": "1984-02",
33          "key": "161280000",
34          "doc_count": 25,
          "count": {
            "value": 25
          }
        }
      ]
    }
  }
}
```