TP2 Exploitation de données

Elasticsearch Query

* Toute requête se fait sur l'API REST, elle doit contenir :

- le nom de l'index

- le nom du type

- Un opérateur de recherche : _count, _search

- un identifiant -> Optionnel

* Avec curl

```
curl -XGET 'http://localhost:9200/tests/test/1'
```

=> Index 'tests', de type 'test', pour le premier document (_id=1)

```
curl -XGET 'http://localhost:9200/tests/test/_count'
```

=> Compte le nombre de documents de 'tests/test'

*Ou aussi avec Kibana DevTools (http://localhost:5601/app/dev_tools), ou Cerebro (http://localhost:9000)

=> Voir si besoin TP1

Requêtes Simples

Le service **_search** permet de faire des requêtes simples, On peut écrire la requête avec trois formats (si possible) :

• Avec la méthode HTTP GET (web brower), avec le paramètre q=

http://localhost:9200/movies/_search?q=fields.title:foo+bar

curl -XGET 'http://localhost:9200/tests/test/_search?q=alien&pretty=true'

• Avec la méthode HTTP POST, avec le paramètre de document JSON « query.json »,pour certaines requêtes complexes en utilisant le DSL (Domain Specific Language), il faut envoyer un document JSON ”requête”:

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

=> consulter ensuite ce document JSON dans un navigateur Web.

• Ou via un admin tool (cerebro ou Kibana devtools)

```
GET /movies/_search
{"query":{
"match":{"fields.title":"foo bar"}}
}
```

Requêtes Simples

Le service _search permet de faire des requêtes simples

- Recherche de mots dans l'ensemble de chaque document

```
curl -XGET 'http://localhost:9200/tests/test/_search?q=alien&pretty=true'
```

- Dans le titre de chaque document

```
curl -XGET 'http://localhost:9200/tests/test/_search?q=title:alien&pretty=true'
```

- Combinaison de critères avec l'espace (%20 pour une URL). Ici, la négation est utilisée ("-")

```
curl -XGET 'http://localhost:9200/tests/test/_search?q=title:alien%20-year:1992&pretty=true'
```

Requêtes Simples: Matching

Pour avoir une requête complexe en utilisant le DSL (Domain Specific Language), il faut envoyer un document JSon ”requête”:

curl -XGET http://localhost:9200/tests/test/_search?pretty -d '

{

"query" : ...

}'

- Requêtes de simple correspondance (matching)

```
{
  "query" : {
    "match" : { "nom_de_l'attribut" : "la\_valeur\_a\_chercher"}
  }
}
```

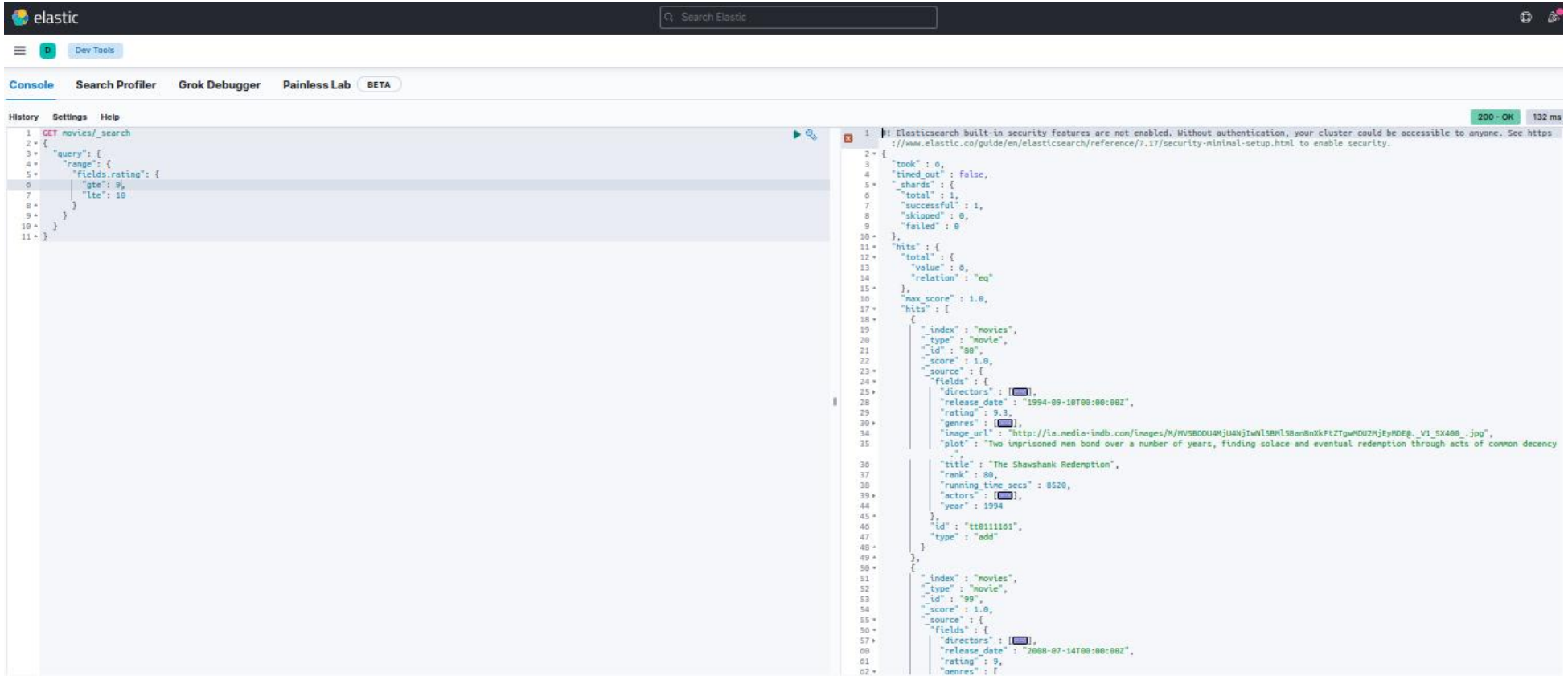
Requêtes Simples: range

- Requêtes d'intervalles (range)

```
{
  "query" : {
    "range" : { "nom_de_l'attribut" : "la\_borne"}
  }
}
```

```
GET movies/_search
{
  "query": {
    "range": {
      "fields.rating": {
        "gte": 9,
        "lte": 10
      }
    }
  }
}
```

Requêtes Simples: range



Requêtes Simples:should & must

- Notions de critère optionnel ’should’ (avec calcul de score de pertinence) et d’obligatoire ’must’ (sans score). Il faut utiliser le critère en imbriquant dans un opérateur ’bool’

```
{"query":{
  "bool": {
    "should": { "match": { "nom_de_l'attribut" : "valeur" }},
    "must": { "range": { "nom_de_l'attribut" : "valeur"}},
    "must_not":[
        {"match":{"nom_de_l'attribut" : "valeur"}},
        {"match":{"nom_de_l'attribut" : "valeur"}},
        ...
    ]
}}}
```

Requêtes Simples:should & must

Occur	Description
`must`	The clause (query) must appear in matching documents and will contribute to the score.
`filter`	The clause (query) must appear in matching documents. However unlike `must` the score of the query will be ignored. Filter clauses are executed in filter context, meaning that scoring is ignored and clauses are considered for caching.
`should`	The clause (query) should appear in the matching document.
`must_not`	The clause (query) must not appear in the matching documents. Clauses are executed in filter context meaning that scoring is ignored and clauses are considered for caching. Because scoring is ignored, a score of `0` for all documents is returned.

The `bool` query takes a *more-matches-is-better* approach, so the score from each matching `must` or `should` clause will be added together to provide the final `_score` for each document.

La REF est ici:
* https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html
* https://www.elastic.co/guide/en/elasticsearch/reference/7.17/query-dsl-bool-query.html

Requêtes Simples:should & must

```
GET movies/_search
{
  "query": {
   "bool": {
     "should": [
       {"match": {
         "fields.rating": 9
       }}
     ],
     "must": [
       {"range": {
         "fields.year": {
           "gte": 1990,
           "lte": 2010
         }
       }}
     ],
     "must_not": [
       {"match": {
         "fields.genres": "Action"
       }}
     ]
   }
  }
}
```


```
#! Elasticsearch built-in security features are not enabled. Without authentication, your cluster could be accessible to anyone. See https
  ://www.elastic.co/guide/en/elasticsearch/reference/7.17/security-minimal-setup.html to enable security.
{
  "took" : 6,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 2069,
      "relation" : "eq"
    },
    "max_score" : 2.0,
    "hits" : [
      {
        "_index" : "movies",
        "_type" : "movie",
        "_id" : "114",
        "_score" : 2.0,
        "_source" : {
          "fields" : {
            "directors" : [
              "Quentin Tarantino"
            ],
            "release_date" : "1994-05-12T00:00:00Z",
            "rating" : 9,
            "genres" : [
              "Crime",
              "Drama",
              "Thriller"
            ],
            "image_url" : "http://ia.media-imdb.com/images/M/MV5BMjE0ODk2NjczOV5BMl5BanBnXkFtZTYwNDQ0NDg4._V1_SX400_.jpg",
            "plot" : "The lives of two mob hit men, a boxer, a gangster's wife, and a pair of diner bandits intertwine in four tales of
              violence and redemption.",
            "title" : "Pulp Fiction",
            "rank" : 114,
            "running_time_secs" : 9240,
            "actors" : [
              "John Travolta",
              "Uma Thurman",
              "Samuel L. Jackson"
            ],
            "year" : 1994
          },
          "id" : "tt0110912",
          "type" : "add"
        }
```

Requêtes Simples:should & must

```
POST _search
{
  "query": {
    "bool" : {
      "must" : {
        "term" : { "user.id" : "kimchy" }
      },
      "filter": {
        "term" : { "tags" : "production" }
      },
      "must_not" : {
        "range" : {
          "age" : { "gte" : 10, "lte" : 20 }
        }
      },
      "should" : [
        { "term" : { "tags" : "env1" } },
        { "term" : { "tags" : "deployed" } }
      ],
      "minimum_should_match" : 1,
      "boost" : 1.0
    }
  }
}
```

Requêtes Simples: filter

- Une requête avec l'opérateur 'filter' ne calcule aucun score pour la requête correspondante (même avec 'should')

```
{
  "query": {
    "filtered": {
      "query": {
        "match": {"nom_de_l'attribut" : "valeur"}
      },
      "filter": {
        "range": {"nom_de_l'attribut" : "valeur"}
}}}}
```

La REF est ici:
* https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html
* https://www.elastic.co/guide/en/elasticsearch/reference/7.17/query-dsl-bool-query.html

Requêtes Simples: filter

- The `title` field contains the word `search`.
- The `content` field contains the word `elasticsearch`.
- The `status` field contains the exact word `published`.
- The `publish_date` field contains a date from 1 Jan 2015 onwards.

```
GET /_search
{
  "query": { (1)
    "bool": { (2)
      "must": [
        { "match": { "title":   "Search"        }},
        { "match": { "content": "Elasticsearch" }}
      ],
      "filter": [ (3)
        { "term":  { "status": "published" }},
        { "range": { "publish_date": { "gte": "2015-01-01" }}}
      ]
    }
  }
}
```

(1) The `query` parameter indicates query context.

(2) The `bool` and two `match` clauses are used in query context, which means that they are used to score how well each document matches.

(3) The `filter` parameter indicates filter context. Its `term` and `range` clauses are used in filter context. They will filter out documents which do not match, but they will not affect the score for matching documents.

https://www.elastic.co/guide/en/elasticsearch/reference/7.17/query-filter-context.html

Requetes Simples

Vous êtes mnt prêt pour le TP2 => voir Notebook