Introduction elasticsearch

@somkiat

SPRINT3R

Siam Chamnankit Co., Ltd., Odd-e (Thailand) Co., Ltd. and Alliance

Plug-ins

High-Availability

Lucene

Distributed

RESTFul

Scalability

API

JSON

elasticsearch

realtime, search and analytics engine

open-source

documentation

document store

JAVA

History

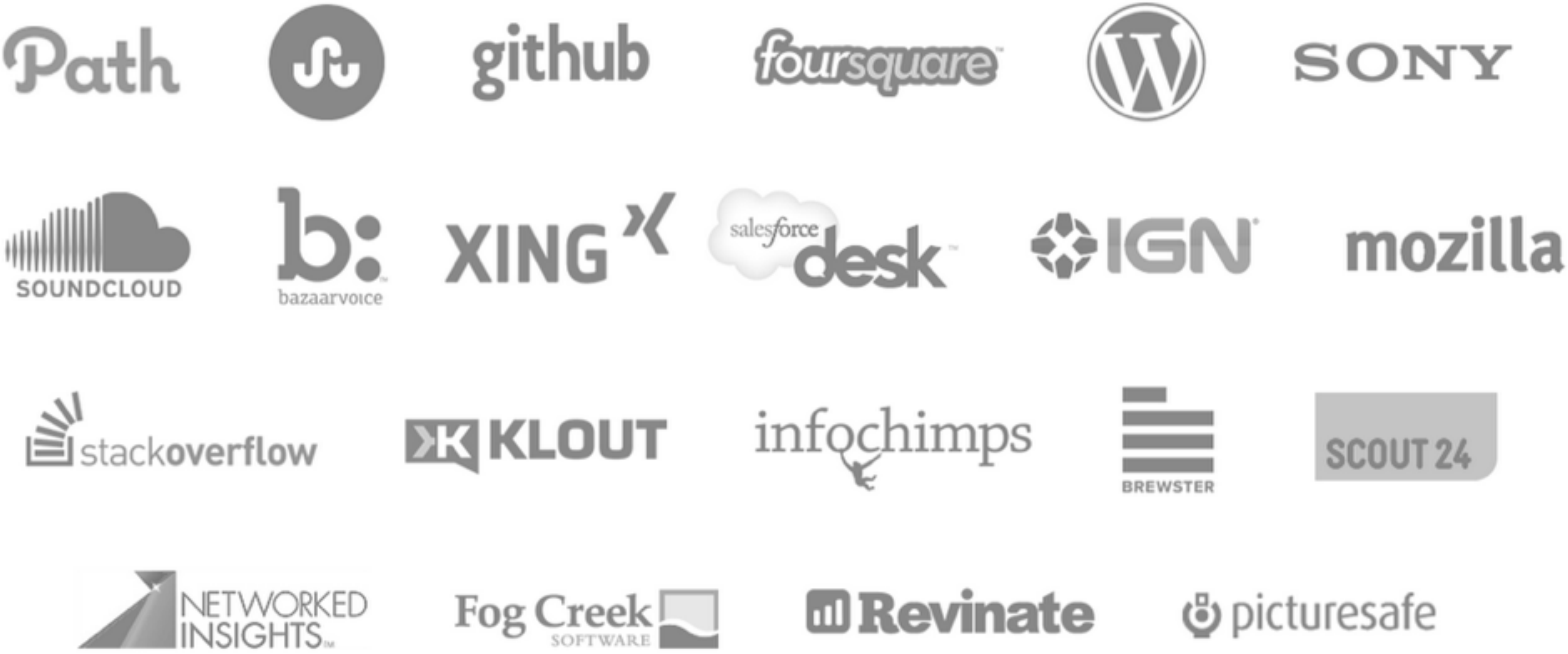
Created by **Shay Banon**

First version 0.4 in Feb 2010

- Rewrite from Compass project
- Add scalability

Current version 1.7.1

Who use ?



https://www.elastic.co/use-cases

- Search repositories, users, issues, pull request
- Search sourcecode 130 พันล้านบรรทัด
- Track alerts, events และ logs

stackoverflow

- ใช้ Full text seach + geolocation
- ใช้ feature More-like-this ในการหาคำถาม และ คำตอบ

Clients

- Java
- PHP
- Ruby
- Python
- JavaScript
- NodeJS
- Go
- Scala
- .Net
- Clojure
- Erlang
- R

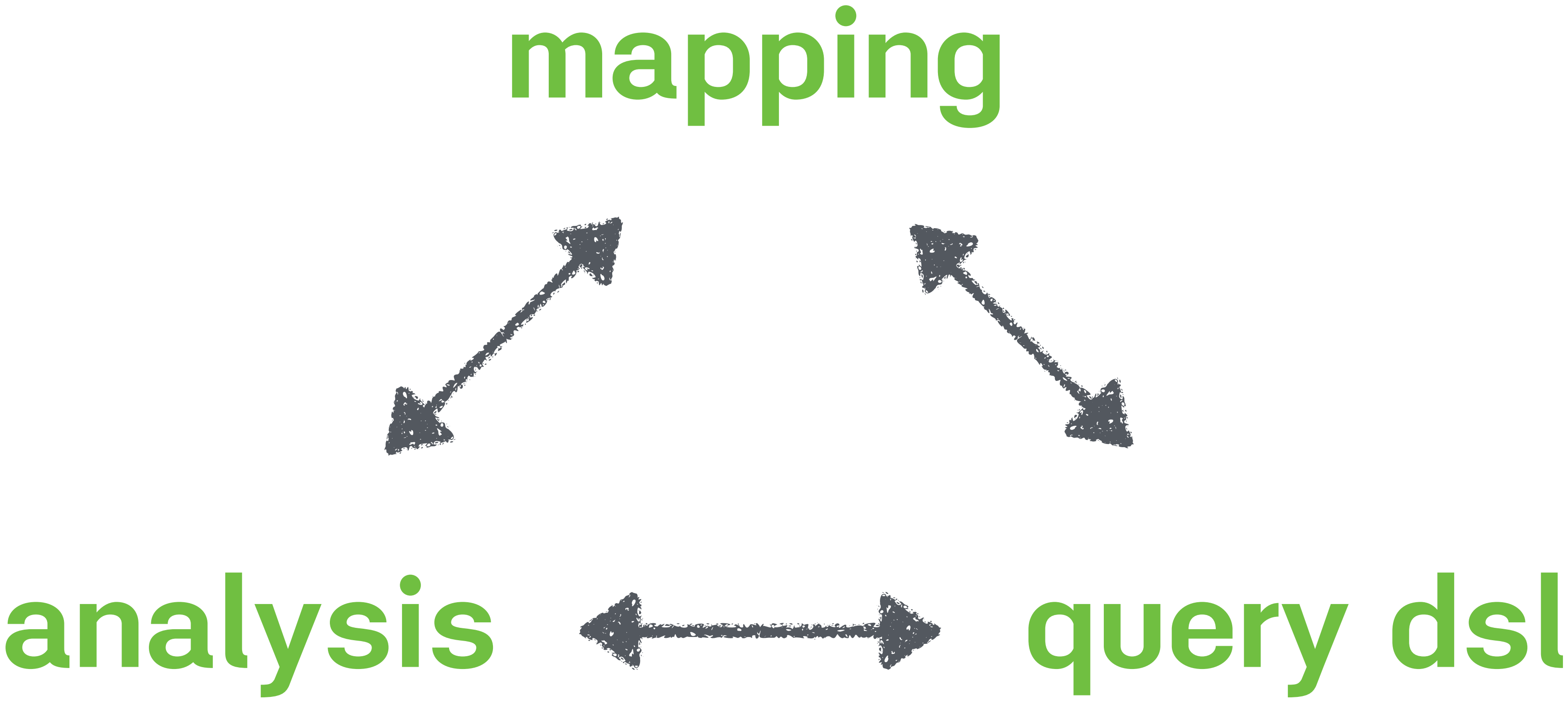
http://www.elasticsearch.org/guide/en/elasticsearch/client/community/current/clients.html https://github.com/github-sprint3r/elasticsearch_workshop

Let’s start

Installation ...

elasticsearch

- real-time
- distributed
- search
- analytics



mapping

analysis

query dsl

query dsl

- flexible
- powerful
- query language

queries

- relevance
- full text
- not cached
- slower

filters

- boolean yes/no
- exact values
- cached
- faster

Filter first

query remaining documents

GET /_search

```
GET /_search
{
    “query” : { . . . }
}
```

```
GET /_search
{
    “query” : { “match” : { “title” : “search” } }
}
```

```
GET /_search
{
    “query” : { “match_all” : { } }
}
```

GET /_search

```
{
    “query” : {
        “filtered” : {
            “query” : { … },
            “filter”  : { … }
        }
    }
}
```

GET /_search

```
{
    “query” : {
        “filtered” : {
            “query” : { “match” : { “title” : “search” } },
            “filter”  : { … }
        }
    }
}
```

GET /_search

```
{
    “query” : {
        “filtered” : {
            “query” : { “match” : { “title” : “search” } },
            “filter”  : { “term” : { “status” : “active” } }
        }
    }
}
```

GET /_search

```
{
    “query” : {
        “filtered” : {
            “query” : { “match_all” : { } },
            “filter”  : { “term” : { “status” : “active” } }
        }
    }
}
```

GET /_search

```
{
    “query” : {
        “filtered” : {
            “query” : { “match_all” : { } },
            “filter”  : { “term” : { “status” : “active” } }
        }
    }
}
```

How data is indexed ?

```
{
  "title":   "Quick brown rabbits",
  "content": "Brown rabbits are commonly seen"
}

{
  "title":   "Keeping pets healthy",
  "content": "My quick brown fox eats rabbits on
              a regular basis"
}
```

where content like

“%brown%fox%”

Analysis

```
{
  "title":   "Quick brown rabbits",
  "content": "Brown rabbits are commonly seen"
}

{
  "title":   "Keeping pets healthy",
  "content": "My quick brown fox eats rabbits on
              a regular basis"
}
```

```
{
  "title":   [quick,brown,rabbits],
  "content": [brown,rabbits,are,commonly,seen]
}

{
  "title":   [keeping,pets,healthy],
  "content": [my,quick,brown,fox,eats,rabbits,on
             ,a,regular,basis]
}
```

field: content

Term	Document 1	Document 2
a		🟧
are	🟧	
basis		🟧
brown	🟧	🟧
commonly	🟧	
eats		🟧
fox		🟧
my		🟧
on		🟧
quick		🟧
rabbits	🟧	🟧
regular		🟧
seen	🟧	

field: content

Term	Document 1	Document 2
a		
are		
basis		
brown	🟧	🟧
commonly		
eats		
fox		🟧
my		
on		
quick		
rabbits		
regular		
seen		

Inverted indexed

where field = “value”

where field contains “value”

where field contains “value”

term filter

```
“term” :  {
      “title” : “brown”
}
```

GET /_search

```
{
    “query” : {
        “filtered” : {
            “query” : { “match_all” : { } },
            “filter”  : { “term” : { “title” : “brown” } }
        }
    }
}
```

field: title

Term	Document 1	Document 2
brown	🟧	
healthy		🟧
keeping		🟧
pets		🟧
quick	🟧	
rabbits	🟧	

field: title

Term	Document 1	Document 2
brown	🟧	
healthy		
keeping		
pets		
quick		
rabbits		

Result

bitset[1, 0]

cache as “title:brown”

where field IN [“value1”, …]

where field IN [“value1”, ...]

terms filter

```
“terms” :  {
      “title” : [“quick”, “pets”]
}
```

field: title

Term	Document 1	Document 2
brown		
healthy		
keeping		
pets		🟧
quick	🟧	
rabbits		

Result

bitset[1, 1]

cache as “title:quick title:pets”

```
where  field  >= “value 1”
and  field < “value 2”
```

where field >= “value 1”
and field < “value 2”

range filter

```
“range” :  {
    “content” : {
        “gte” : “a”,
        “lt” : “m”
    }
}
```

field: content

Term	Document 1	Document 2
a		🟧
are	🟧	
basis		🟧
brown	🟧	🟧
commonly	🟧	
eats		🟧
fox		🟧
my		🟧
on		
quick		
rabbits		
regular		
seen		

Result

bitset[1, 1]

cache as “content:[a TO m]”

AND … OR … NOT

AND … OR … NOT

bool filter

```
“bool” :  {
     “must” :          [ <filters> ],
     “should” :        [ <filters> ],
     “must_not” :  [ <filters> ],
}
```

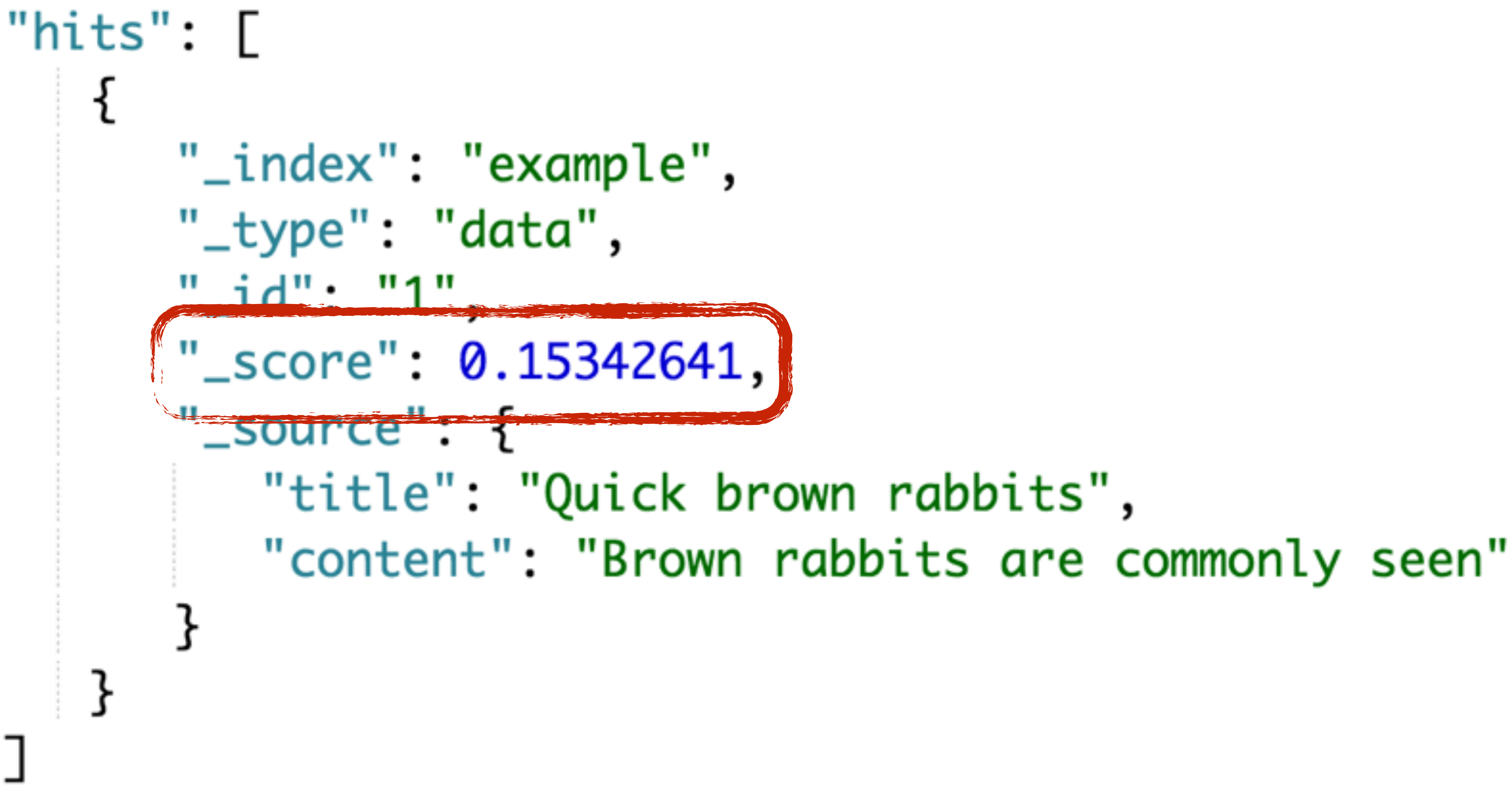
how relevant is this term ?

how relevant is this term ?

term query

```
GET /_search
{
    “query” : {
        “term” : {
            “title”   : “brown”
         }
    }
}
```

how relevant is this document ?



```
"hits": [
  {
    "_index": "example",
    "_type": "data",
    "_id": "1",
    "_score": 0.15342641,
    "_source": {
      "title": "Quick brown rabbits",
      "content": "Brown rabbits are commonly seen"
    }
  }
]
```

Relevant score

Term in this document

more is better

Relevant score

Term in all documents

less is better

Relevant score

How long of this document ?

shorter is better

AND … OR … NOT

bool query

```
“bool” :  {
     “must” :          [ <filters> ],  #AND
     “should” :        [ <filters> ],   ???
     “must_not” :  [ <filters> ],  #NOT
}
```

```
“bool” :  {
      “must” :          [ <filters> ],  #AND
      “should” :       [ <filters> ],   ???
      “must_not” :  [ <filters> ],  #NOT
      “minimum_should_match” : ?
}
```

bool filter -> True | False

bool query -> _score

match query

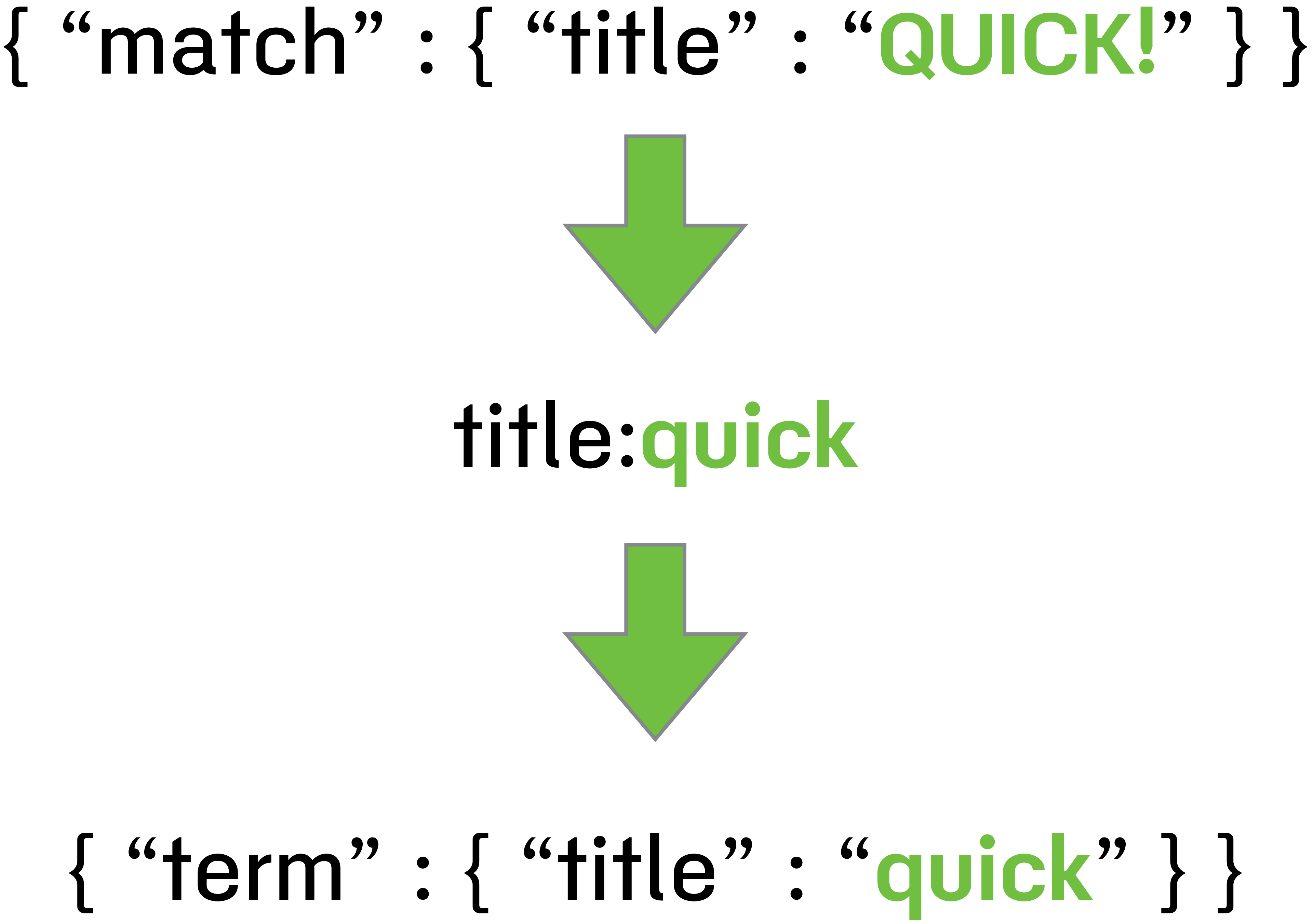
match query

high level query

understand mapping and analysis

match query

1. analyze query string
2. rewrite query



{ “match” : { “title” : “QUICK!” } }

title:quick

{ “term” : { “title” : “quick” } }

{ “match” : { “title” : “QUICK FOX!” } }

↓

title:quick OR title:fox

↓

```
{ “bool” : {
      “should” : [
            { “term” :  { “title” : “quick” } },
            { “term” :  { “title” : “fox” } }
      ]
}}
```

Need all words must match ?

GET /_search

```
{
    “query” : {
        “term” : {
            “title”  : “QUICK FOX!”
        }
    }
}
```

```
GET /_search
{
    “query” : {
        “term” : {
            “title”  : {
                “query” : “QUICK FOX!”,
                “operator” : “and”
            }
        }
    }
}
```

GET /_search

```
{
  "query": {
      "bool": {
          "must": [
                { "term": { "title" : "quick" } },
                { "term": { "title" : "fox" } }
          ]
        }
  }
}
```

```
GET /_search
{
    “query” : {
         “match_pharse” : {
              “title”  : {
                   “query” : “QUICK BROWN”
              }
         }
    }
}
```

```
GET /_search
{
    “query” : {
        “match_pharse” : {
            “title”  : {
                “query” : “QUICK RABBITS”
            }
        }
    }
}
```

```
GET /_search
{
    “query” : {
        “match_pharse” : {
            “title”  : {
                “query” : “BROWN QUICK”
            }
        }
    }
}
```

Limit long tail ?

GET /_search

```
{
    “query” : {
        “term” : {
            “title”  : {
                “query” : “QUICK BROWN FOX!”,
                “minimum_should_match” : “75”%
            }
        }
    }
}
```

GET /_search

```
{
  "query": {
        "bool": {
              "should": [
                      { "term": { "title" : "quick" } },
                      { "term": { "title" : "brown" } },
                      { "term": { "title" : "fox" } }
              ],
              “minimum_should_match” : 2
           }
  }
}
```

Aggregation API

analytic your data
explore your data