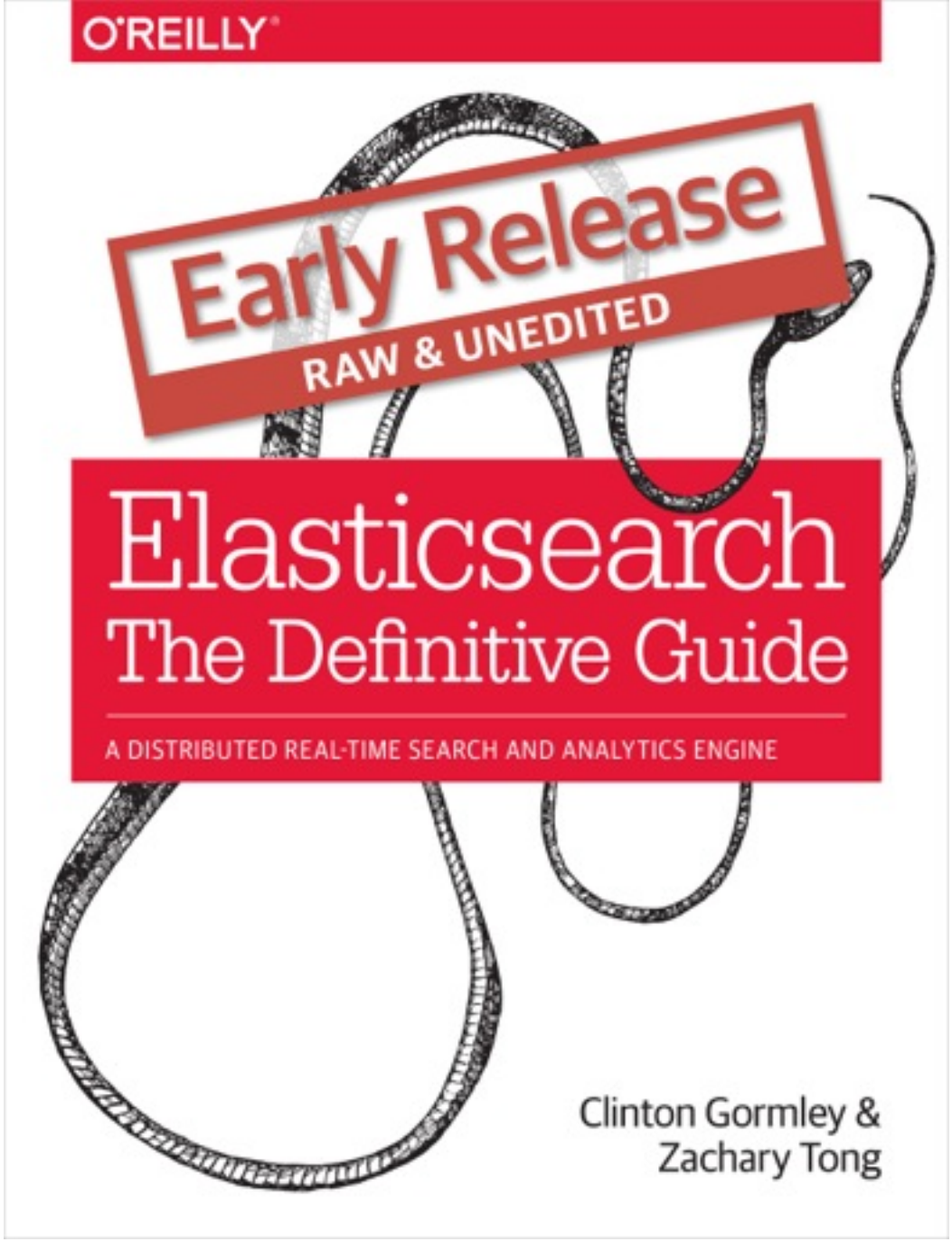
Elasticsearch Query DSL

… not just for wizards

Clinton Gormley

@clintongormley

elasticsearch.



elasticsearch.

elasticsearch.org/guide

elasticsearch.

elasticsearch

elasticsearch

- real-time

elasticsearch

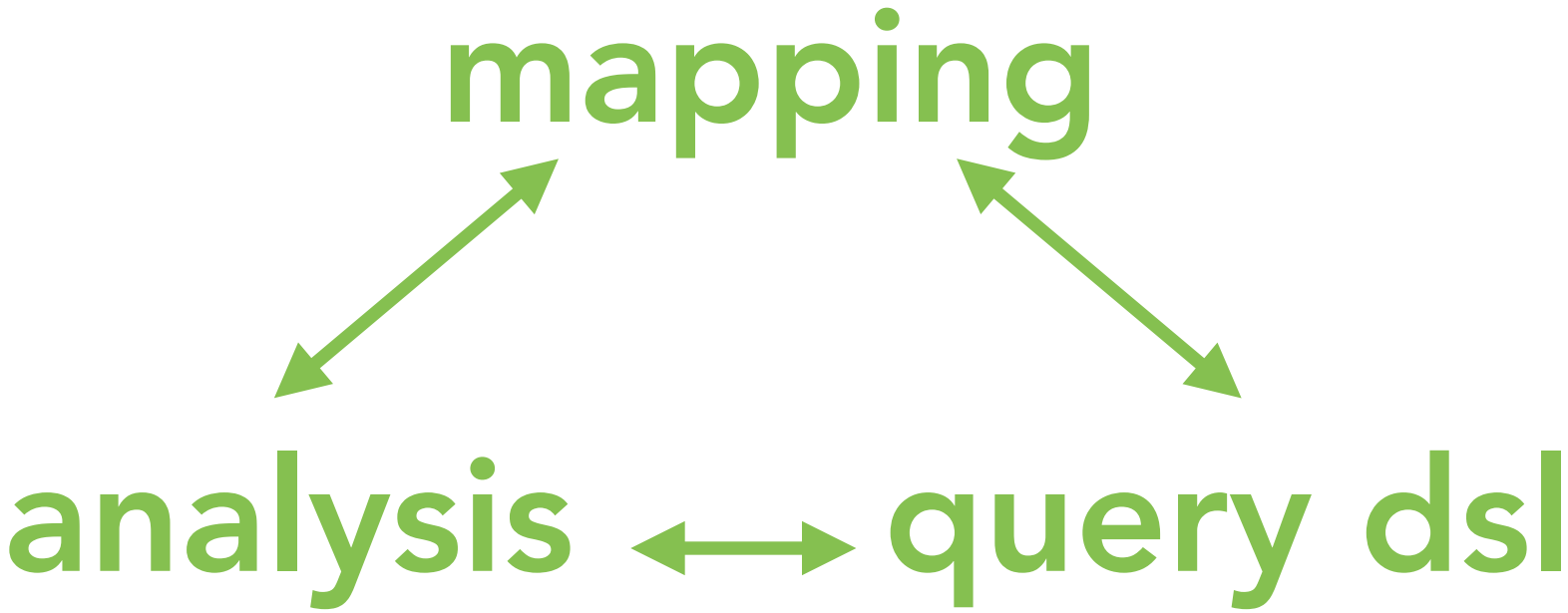
- real-time
- distributed

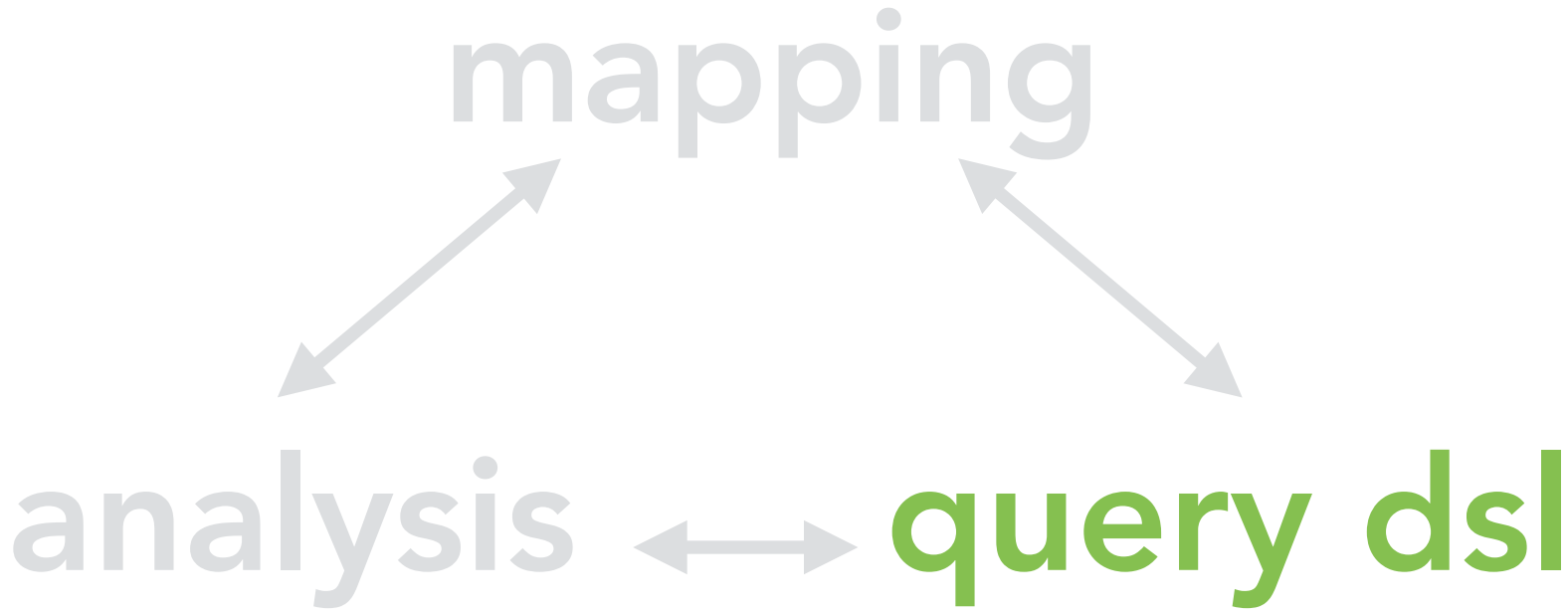
elasticsearch

- real-time
- distributed
- search

elasticsearch

- real-time
- distributed
- search
- analytics





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, then query remaining docs

GET /_search

GET /_search

```
{
  "query": {...}
}
```

GET /_search

```
{
  "query": { "match": { "title": "search" }}
}
```

GET /_search

```
{
  "query": { match_all: {}}
}
```

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":   "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"
}
```

where content like "%brown%fox%"

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

slow & inflexible

```
{
  "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"
}
```

"analysis"

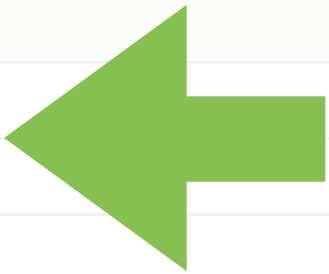
```
{
  "title":   "Keeping pets healthy",
  "content": "My quick brown fox eats rabbits on a
              regular basis"
}
```

```
{
  "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"
}
```

```
{
  "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	Doc 1	Doc 2
a		
are		
basis		
brown		
commonly		
eats		
fox		
my		
on		
quick		
rabbits		
regular		
seen		



sorted list of unique terms

field: content

Term	Doc 1	Doc 2

where they occur

field: content

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

field: content

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

inverted index

inverted index

not just for text

inverted index

numbers, dates, bools, enums

geopoints, geoshapes, etc

```
WHERE field = "value"
```

```
WHERE field CONTAINS "value"
```

```
WHERE field CONTAINS "value"
```

term filter

WHERE field CONTAINS "value"

```
"term":  {
    "title": "brown"
}
```

GET /_search

```
{
  "query": {




  }
}
```

GET /_search

```
{
  "query": {
    "filtered": {


    }
  }
}
```

GET /_search

```
{
  "query": {
    "filtered": {
      "query":  {                 ...                 },
      "filter": {                 ...                 }
    }
  }
}
```

GET /_search

```
{
  "query": {
    "filtered": {
      "query":  { "match_all": {}                  },
      "filter": {                 ...              }
    }
  }
}
```

GET /_search

```
{
  "query": {
    "filtered": {
      "query":  { "match_all": {}                    },
      "filter": {                  ...               }
    }
  }
}
```

GET /_search

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

field: title

Term	Doc 1	Doc 2
brown		
healthy		
keeping		
pets		
quick		
rabbits		

field: title

Term	Doc 1	Doc 2
brown		
healthy		
keeping		
pets		
quick		
rabbits		

```
"term": { "title": "brown" }
```

➔ result: bitset[1, 0]

➔ cache as: "title:brown"

```
WHERE field IN ["val",…]
```

```
WHERE field IN ["val",…]
```

terms filter

WHERE field IN ["val",…]

```
"terms":  {
    "title": ["quick", "pets"]
}
```

field: title

Term	Doc 1	Doc 2
brown		
healthy		
keeping		
pets		
quick		
rabbits		

```
"terms": { "title": ["quick","pets"] }
```

→ result: bitset[1, 1]

→ cache as: "title:quick title:pets"

```
WHERE field >= "val1"
  AND field <  "val2"
```

```
WHERE field >= "val1"
  AND field <  "val2"
```

range filter

```
WHERE field >= "val1"
  AND field <  "val2"
```

```
"range":  {
    "content":{
      "gte": "a",
      "lt":  "m"
    }
}
```

field: content

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

```
"range":  {
   "content":{ "gte": "a", "lt":  "m" }
}
```

→ result: bitset[1, 1]

→ cache as: "content:[a TO m}"

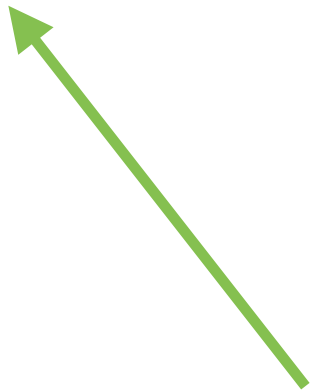
```
"range":  {
    "date":{
      "gte": "2014-01-01",
      "lt":  "2041-02-01"
    }
 }
```

"range": {

numeric/date fields
optimised
for range filters

}

```
"range":  {
    "date":{
      "gte": "now - 1h"
    }
 }
```



not cached

```
"range":  {
    "date":{
      "gte": "now - 1h / h"
    }
 }
```

cached

WHERE field IS NOT NULL

WHERE field has any term

`WHERE field has any term`

exists filter

WHERE field has any term

```
"exists":  {
  "field": "title"
}
```

```
WHERE field has no term
```

missing filter

WHERE field has no term

```
"missing":  {
  "field": "title"
}
```

AND … OR … NOT

AND … OR … NOT

bool filter

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

```
"bool":  {
  "must":     [ <filters> ], # AND
  "should":   [ <filters> ],
  "must_not": [ <filters> ]
}
```

```
"bool":  {
   "must":     [ <filters> ],
   "should":   [ <filters> ], # OR
   "must_not": [ <filters> ]
}
```

```
"bool":  {
  "must":     [ <filters> ],
  "should":   [ <filters> ],
  "must_not": [ <filters> ]  # NOT
}
```

```
"bool":  {
  "must": { "term": { "title": "rabbits" }},




}
```

```
"bool":  {
  "must": { "term": { "title": "rabbits" }},
  "should": [
    { "term": { "title":   "quick" }},
    { "term": { "content": "quick" }}
  ],

}
```

```
"bool":  {
  "must": { "term": { "title": "rabbits" }},
  "should": [
    { "term": { "title":   "quick" }},
    { "term": { "content": "quick" }}
  ],
  "must_not": { "term": { "content": "fox" }}
}
```

```
"bool":  {
  "must": [
    { "bool":  {
      "should": [
        { "term": { "title":   "rabbits" }},
        { "term": { "content": "rabbits" }}
    ]}},
    { "bool":  {
      "should": [
        { "term": { "title":   "quick" }},
        { "term": { "content": "quick" }}
    ]}}
  ],
  "must_not": { "term": { "content": "fox" }}
}
```

```
"bool": {
  "must": [
    { "bool": {
      "should": [
        { "term": { "title":   "rabbits" }},
        { "term": { "content": "rabbits" }}
    ]}},
    { "bool":  {
      "should": [
        { "term": { "title":   "quick" }},
        { "term": { "content": "quick" }}
    ]}}
  ],
  "must_not": { "term": { "content": "fox" }}
}
```

not cached

```
"bool": {
  "must": [
    { "bool": {
      "should": [
        { "term": { "title":   "rabbits" }},
        { "term": { "content": "rabbits" }}
    ]}},
    { "bool": {
      "should": [
        { "term": { "title":   "quick" }},
        { "term": { "content": "quick" }}
    ]}}
  ],
  "must_not": { "term": { "content": "fox" }}
}
```

cached

```
result bitset =

    (title:rabbits OR content:rabbits)
  AND
    (title:quick   OR content:quick)
  AND
    NOT content:fox
```

filters

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

Filter first, then query

how relevant is this term?

how relevant is this term?

term query

how relevant is this term?

term query

≈ term filter + relevance

how relevant is this term?

```
"term":  {
    "title": "brown"
}
```

GET /_search

```
{
    "query": {


    }
}
```

GET /_search

```
{
    "query": {
        "term": {
            "title": "brown"
        }
    }
}
```

how relevant is this doc?

```
{
    "_index": "myindex",
    "_type":  "mytype",
    "_id":    "1",
    "_score": 0.5,
    "_source": {
       "title":   "Quick brown rabbits",
       "content": "Brown rabbits are commonly seen"
    }
}
```

relevance score

How common is the term in this doc?

→ more is better

relevance score

How common is the term in this doc?

→ more is better

How common is the term in ALL docs?

→ less is better

relevance score

How common is the term in this doc?

→ more is better

How common is the term in ALL docs?

→ less is better

How long is this doc?

→ shorter is better

lucene similarity

How common is the term in this doc?

→ more is better

How common is the term in ALL docs?

→ less is better

How long is this doc?

→ shorter is better

lucene similarity

How common is the term in this doc?

→ more is better

How common is the term in ALL docs?

→ less is better

How long is this doc?

→ shorter is better

lucene similarity

Term frequency

How common is the term in ALL docs?

→ less is better

How long is this doc?

→ shorter is better

lucene similarity

Term frequency

How common is the term in ALL docs?

→ less is better

How long is this doc?

→ shorter is better

lucene similarity

Term frequency

Inverse document frequency

How long is this doc?

→ shorter is better

lucene similarity

Term frequency

Inverse document frequency

How long is this doc?

→ shorter is better

lucene similarity

Term frequency

Inverse document frequency

Length norm

AND … OR … NOT

AND … OR … NOT

bool query

AND … OR … NOT

bool query

like bool filter, but different...

```
"bool":  {
  "must":     [ <queries> ], # AND
  "should":   [ <queries> ],
  "must_not": [ <queries> ]  # NOT
}
```

```
"bool":  {
  "must":     [ <queries> ], # AND
  "should":   [ <queries> ], # Hmmm
  "must_not": [ <queries> ]  # NOT
}
```

```
"bool":  {
  "must":     [ <queries> ], # AND
  "should":   [ <queries> ], # Hmmm
  "must_not": [ <queries> ], # NOT
  "minimum_should_match": ?  # Hmmm
}
```

no "must" queries

```
"bool":  {
 "should":   [
    { "term": { "title": "quick" }},
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ]
}
```

at least one must match

no "must" queries

```
"bool":  {
 "should":   [
    { "term": { "title": "quick" }},
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ]
}
```

minimum_should_match = 1

with "must" queries

```
"bool":  {
  "must": { "term": { "title": "quick"}},
  "should":   [
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ]
}
```

all are optional!

with "must" queries

```
"bool":  {
  "must": { "term": { "title": "quick"}},
  "should":   [
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ]
}
```

minimum_should_match = 0

bool filter → T/F

```
bool filter → T/F

bool query  → _score
```

_score of bool query =

```
_score of bool query =
    sum( _score of each query)
```

```
_score of bool query =
    sum( _score of each query)
    * num of matching queries
```

```
_score of bool query =
    sum( _score of each query)
    * num of matching queries
    / num of queries
```

more matching should queries

more matching should queries

==

better relevance score

trim the long tail

```
"bool":  {
 "should":   [
    { "term": { "title": "quick" }},
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ]
}
```

trim the long tail

```
"bool":  {
 "should":   [
    { "term": { "title": "quick" }},
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ],
  "minimum_should_match": "75%"
}
```

trim the long tail

```
"bool":  {
 "should":   [
    { "term": { "title": "quick" }},
    { "term": { "title": "brown" }},
    { "term": { "title": "rabbits" }}
  ],
  "minimum_should_match": "75%" # 2 of 3
}
```

match query

match query

high level query

match query

high level query

understands mapping & analysis

match query

→ analyze query string

→ rewrite query

one word query

```
{ "match":  { "title": "QUICK!" }}
```

```
title:quick
```

```
{ "term":  { "title": "quick" }}
```

multi word query

```
{ "match":  { "title": "QUICK FOX!" }}
```

```
title:quick OR title:fox
```

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

all words must match

```
{
  "match":  {
    "title": "QUICK FOX!"
  }
}
```

```
{
  "match":  {
    "title": {
      "query":    "QUICK FOX!",

    }
  }
}
```

```
{
  "match":  {
    "title": {
      "query":    "QUICK FOX!",
      "operator": "and"
    }
  }
}
```

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

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

trim long tail

```
{
  "match":  {
    "title": {
      "query": "QUICK BROWN FOX!",
      "minimum_should_match": "75%"
    }
  }
}
```

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

fuzzy queries

levenshtein edit distance

insertion

bron → brown

deletion

bron → brown

foxs → fox

substitution

bron → brown

foxs → fox

kiuck → qiuck

transposition

bron → brown

foxs → fox

kiuck → qiuck → quick

```
{
  "match":  {
    "title": {
      "query": "KIUCK BRON FOXS!",
      "fuzziness": "AUTO"
    }
}
```

phrase / proximity

```
{
  "match_phrase":  {
    "title":"QUICK BROWN FOX!"
  }
}
```

```
{
  "match_phrase":  {
    "title": {
      "query": "BROWN QUICK FOX!",
      "slop":  "10"
    }
  }
}
```

combine queries

```
"bool": {
   "must": { <min_should_match> },
   "should": [
             {      <fuzzy>          },
             {    <proximity>        }
   ]
}
```

```
"bool": {
   "must": { <min_should_match> },
   "should": [
             {      <fuzzy>        },
             {     <proximity>     }
   ]
}
```

```
"match": {
  "title": {
    "query": "<words>",
    "minimum_should_match": "75%"
  }
}
```

```
"bool": {
   "must": { <min_should_match> },
   "should": [
            {      <fuzzy>          },
            {     <proximity>       }
   ]
}
```

```
"match": {
  "title": {
    "query": "<words>",
    "fuzziness": "AUTO"
  }
}
```

```
"bool": {
   "must": { <min_should_match> },
   "should": [
             {      <fuzzy>        },
             {     <proximity>     }
   ]
}
```

```
"match_phrase": {
    "title": {
      "query": "<words>",
      "slop": "10"
    }
 }
```

```
"bool": {
   "must": { <min_should_match> },
   "should": [
            {      <fuzzy>        },
            {     <proximity>     }
   ]
}
```

multi-field queries

Advanced Search

First name	Reginald
Middle name	Kenneth
Last name	Dwight
Department	
City	

Advanced Search

First name: Reginald

Middle name:

Last name:

Department:

City:

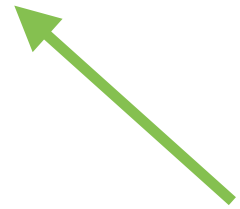
easy!

```
"bool": {
  "should": [
    { "match": {"first":  "Reginald" }},
    { "match": {"middle": "Kenneth"  }},
    { "match": {"last":   "Dwight"   }}
    ],
    "minimum_should_match": "75%"
}
```

```
"bool": {
  "should": [
    { "match": {
        "title":  "quick brown fox"
    }},
    { "match": {
        "content": "quick brown fox"
    }}
  ]
}
```

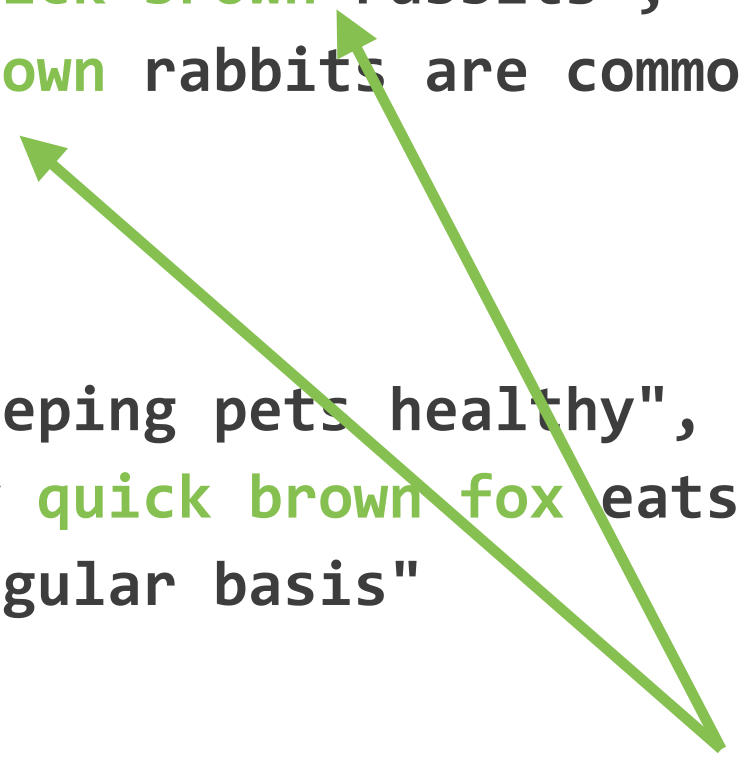
```
{
  "title":   "Quick brown rabbits",
  "content": "Brown rabbits are commonly seen"
}

{
  "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"
}
```



better match

```
{
  "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"
}
```



But 2 matches wins

dis_max query

all docs which match any query

dis_max query

all docs which match any query

`_score = best matching query`

```
"bool": {
  "should": [
    { "match": {
        "title":  "quick brown fox"
    }},
    { "match": {
        "content": "quick brown fox"
    }}
  ]
}
```

```
"dis_max": {
  "queries": [
    { "match": {
        "title":  "quick brown fox"
    }},
    { "match": {
        "content": "quick brown fox"
    }}
  ]
}
```

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

```
"dis_max": {
  "queries": [
    { "match": {
        "title":  "quick brown fox"
    }},
    { "match": {
        "content": "quick brown fox"
    }}
  ],
  "tie_breaker": 0.2
}
```

dis_max query

all docs which match any query

```
_score = best matching query
       + tie_breaker * others
```

multi_match query

multi_match query

match query on multiple fields

```
"dis_max": {
  "queries": [
    { "match": {
        "title":  "quick brown fox"
    }},
    { "match": {
        "content": "quick brown fox"
    }}
  ],
  "tie_breaker": 0.2
}
```

```
"multi_match": {
  "query":       "quick brown fox",
  "fields":   [ "title", "content" ]
  "tie_breaker": 0.2
}
```

```
"multi_match": {
  "query":       "quick brown fox",
  "fields":    [ "title", "content" ]
  "tie_breaker": 0.2,
# "type":        "best_fields"
}
```

best_fields

find whole "concept" in one field

best_fields

"quick brown fox" in title or content

best_fields

"quick brown fox" in title or content

dis_max

```
"title": {
   "type":              "string"
}
```

```
"title": {
  "type":          "string",
  "fields": {

}}
```

```
"title": {
  "type":              "string",
  "fields": {
    "stemmed": {
      "type":     "string",
      "analyzer": "english"
    }


}}
```

```
"title": {
  "type":            "string",
  "fields": {
    "stemmed": {
      "type":     "string",
      "analyzer": "english"
    },
    "autocomplete": {
      "type":     "string",
      "analyzer": "edge_ngrams"
}}}
```

```
title:
   [ brown, fox, jumped ]
```

```
title:
   [ brown, fox, jumped ]

title.stemmed:
   [ brown, fox, jump ]
```

```
title:
  [ brown, fox, jumped ]

title.stemmed:
  [ brown, fox, jump ]

title.autocomplete
  [ b, br, bro, brow, brown,
    f, fo, fox,
    j, ju, jum, jump, jumpe, jumped
  ]
```

```
"multi_match": {
  "query":       "quick brown fox",
  "fields":   [
                 "title",
                 "title.stemmed",
                 "title.autocomplete"
              ]
  "type":        "most_fields"
}
```

most_fields

match same text
analyzed in different ways

most_fields

more matching fields = better

most_fields

more matching fields = better

bool

```
{
  "first":  "Reginald",
  "middle": "Kenneth"
  "last":   "Dwight"
}
```

```
"multi_match": {
  "query": "Reginald Kenneth Dwight",
  "fields":   [
                "first",
                "middle",
                "last"
              ]

}
```

```
"multi_match": {
  "query": "Reginald Kenneth Dwight",
  "fields":   [
                "first",
                "middle",
                "last"
              ]
  "type":       "????"
}
```

```
"multi_match": {
  "query": "Reginald Kenneth Dwight",
  "fields":  [
               "first",
               "middle",
               "last"
             ]
  "type":    "most_fields"
}
```

problem

field centric

```
(     first: Reginald
  OR  first: Kenneth
  OR  first: Dwight
) OR (
      middle:Reginald
  OR  middle:Kenneth
  OR  middle:Dwight
) OR (
      last:  Reginald
  OR  last:  Kenneth
  OR  last:  Dwight
)
```

```
(     first: Reginald
  OR  first: Kenneth
  OR  first: Dwight
) OR (
      middle:Reginald
  OR  middle:Kenneth
  OR  middle:Dwight
) OR (
      last:  Reginald
  OR  last:  Kenneth
  OR  last:  Dwight
)
```

problem

operator: and

```
(     first: Reginald
  AND first: Kenneth
  AND first: Dwight
) OR (
      middle:Reginald
  AND middle:Kenneth
  AND middle:Dwight
) OR (
      last:  Reginald
  AND last:  Kenneth
  AND last:  Dwight
)
```

```
(     first: Reginald
  AND first: Kenneth
  AND first: Dwight
) OR (
      middle:Reginald
  AND middle:Kenneth
  AND middle:Dwight
) OR (
      last:  Reginald
  AND last:  Kenneth
  AND last:  Dwight
)
```

problem

term frequencies

```
first:dwight  → common
last: dwight  → uncommon
```

solution

index time solution

single "fullname" field

```
"first": {
    "type": "string"
},
"middle": {
    "type": "string"
},
"last": {
    "type": "string"
}
```

```
"first": {
    "type": "string"
},
"middle": {
    "type": "string"
},
"last": {
    "type": "string"
},
"full": {
    "type": "string"
}
```

```
"first": {
    "type": "string", "copy_to": "full"
},
"middle": {
    "type": "string", "copy_to": "full"
},
"last": {
    "type": "string", "copy_to": "full"
},
"full": {
    "type": "string"
}
```

```
"match": {
  "full": "Reginald Kenneth Dwight",
  "minimum_should_match": "75%"
}
```

query time solution

term-centric query

```
(     first: Reginald
  AND first: Kenneth
  AND first: Dwight
) OR (
      middle:Reginald
  AND middle:Kenneth
  AND middle:Dwight
) OR (
      last:  Reginald
  AND last:  Kenneth
  AND last:  Dwight
)
```

```
(     first: Reginald
  OR  middle:Reginald
  OR  last:  Reginald
) AND (
      first: Kenneth
  OR  middle:Kenneth
  OR  last:  Kenneth
) AND (
      first: Dwight
  OR  middle:Dwight
  OR  last:  Dwight
)
```

```
    blend(first,middle,last):Reginald
AND blend(first,middle,last):Kenneth
AND blend(first,middle,last):Dwight
```

blends term frequencies

cross_fields

query multiple fields
as if they were one

```
"multi_match": {
  "query": "Reginald Kenneth Dwight",
  "fields":   [
                "first",
                "middle",
                "last"
              ]
  "type":       "cross_fields"
}
```

```
"multi_match": {
  "query": "Reginald Kenneth Dwight",
  "fields":   [
                "first",
                "middle",
                "last"
              ]
  "type":         "cross_fields",
  "minimum_should_match": "75%"
}
```

best_fields:

whole concept in single field

most_fields:

same text, different analyzers

cross_fields:

treat multiple fields as one

understand the building blocks

the rest is details

thank you

@clintongormley

elasticsearch.org/downloads

elasticsearch.com/support

elasticsearch.com/jobs