실무에 활용하는 Elasticsearch 검색엔진 구축

6일차 : ElasticSearch 2

텍스트 분석 및

검색질의

오늘의 아젠다

엘라스틱서치의 텍스트 분석

엘라스틱서치의 검색질의

1. 엘라스틱서치의 텍스트 분석

분석기의 정의

분석기란?

char filter (전처리)+tokenizer (토큰 분해)+token filter (토큰 변환)= Analyzer

내장 분석기

Standard Analyzer
Simple Analyzer
Whitespace Analyzer
Stop Analyzer
Keyword Analyzer
Pattern Analyzer
Language Analyzers
Fingerprint Analyzer

사용자 정의 분석기

특성

인덱스 단위로 저장

settings 옵션 명령으로 생성

0개 혹은 1개 이상 char_filter

1개 이상 tokenizer

0개 혹은 1개 이상 token filter

사용자 정의 분석기 추가,수정

1. 색인닫기 (POST /index_name/_close)
2. 추가,수정
3. 색인열기 (POST /index_name/_close)

```
curl -XPUT ‘http://localhost:9200/books' -d ‘
{
  "settings": {
    "analysis": {
      "analyzer": {
        "my_custom_analyzer": {
          "type":      "custom",
          "tokenizer": "standard",
          "char_filter": [
            "html_strip"
          ],
          "filter": [
            "lowercase",
            "asciifolding"
          ]
        }
      }
    }
  }
}
```

확인

http://localhost:9200/books/_settings?pretty

http://localhost:9200/books/_analyze?&text={text}

Built in - Character Filter, Tokenizer, Token Filter

캐릭터필터

HTML Strip Character Filter
Mapping Character Filter
Pattern Replace Character Filter

토크나이저

Standard Tokenizer
Letter Tokenizer
Lowercase Tokenizer
Whitespace Tokenizer
UAX URL Email Tokenizer
Classic Tokenizer
Thai Tokenizer
Keyword Tokenizer
Pattern Tokenizer
Path Tokenizer

토큰필터

Reverse Token Filter
Elision Token Filter
Truncate Token Filter
Unique Token Filter
Pattern Capture Token Filter
Pattern Replace Token Filter
Trim Token Filter
Limit Token Count Token Filter
Hunspell Token Filter
Common Grams Token Filter
Normalization Token Filter
CJK Width Token Filter
CJK Bigram Token Filter
Delimited Payload Token Filter
Keep Words Token Filter
Keep Types Token Filter
Classic Token Filter
Apostrophe Token Filter
Decimal Digit Token Filter
Fingerprint Token Filter
Minhash Token Filter
Standard Token Filter
ASCII Folding Token Filter
Flatten Graph Token Filter
Length Token Filter
Lowercase Token Filter
Uppercase Token Filter
NGram Token Filter
Edge NGram Token Filter
Porter Stem Token Filter
Shingle Token Filter
Stop Token Filter
Word Delimiter Token Filter
Word Delimiter Graph Token Filter
Stemmer Token Filter
Stemmer Override Token Filter
Keyword Marker Token Filter
Keyword Repeat Token Filter
KStem Token Filter
Snowball Token Filter
Phonetic Token Filter
Synonym Token Filter
Synonym Graph Token Filter
Compound Word Token Filters

사용자 정의 토크나이저

```
curl -XPUT ‘http://localhost:9200/books' -d ‘
{
  "settings": {
    "analysis": {
      "analyzer": {
        "my_custom_analyzer": {
          "tokenizer": "standard"
        }
      }
    }
  }
}
```

```
curl -XPUT ‘http://localhost:9200/books' -d ‘
{
   "settings": {
      "analysis": {
         "analyzer": {
            "my_custom_analyzer": {
               "tokenizer": "my_custom_tokenizer"
            },
            "tokenizer" : {
               "my_custom_tokenizer" : {
                  "type" : “standard",
                  "max_token_length" : 100
               }
            }
         }
      }
   }
}
```

실습!!
ngram 분석기 만들기

사용자 정의 필터 (동의어 필터)

```
curl -XPUT ‘http://localhost:9200/books' -d ‘
{
   "settings": {
      "index" : {
         "analysis" : {
            "analyzer" : {
               "synonym" : {
                  "tokenizer" : "whitespace",
                  "filter" : [“synonym”]
               }
            },
            "filter" : {
               "synonym" : {
                  "type" : "synonym",
                  "synonyms_path" : "analysis/synonym.txt"
               }
            }
         }
      }
   }
}
```

동의어 필터 옵션

synonyms : 동의어를 인라인으로 정의

synonyms_path : 동의어 사전 위치를 정의 (추천)

format : 기본은 solr 포맷,wordnet 포맷 제공 (http://wordnet.kaist.ac.kr/)

solr 포맷 형식

1) i-pod,i pod => ipod (대체)
2) ipod,i-pod,i pod (확장)
3) foo => foo bar
   foo => baz foo => foo bar, baz (병합)

기타 참고

독음처리 (https://github.com/muik/transliteration)

2. 엘라스틱서치의 검색질의

URI 요청 질의

주로 간단한 테스트 질의에 사용

예) curl –XGET ‘localhost:9200/index1/book/_search?q=title:엘라스틱&pretty=true

주요 파라메터

q (query) : 필드명: 질의어 형태 루씬 기본 파서 사용
df(default field) : 검색할 필드 지정
default_operator : 질의어 사이 공백값에 대한 AND/OR 처리
explain : 스코어 랭킹 계산식 출력
fields : 출력 결과를 표시할 필드 지정
sort : 검색 결과를 출력할 순서 결정 기본값 _score기준 (필드명:desc)
from : 검색결과를 몇번째 값부터 출력할 지 여부 (페이징에 사용)
size : 검색결과를 몇개까지 표시 할것인지 여부 (페이징에 사용)
search_type :

1. query_then_fetch : 전체 샤드검색이 모두 수행된후 출력(default)
2. query_and_fetch : 샤드별로 검색되는 대로 결과를 받아 출력 (size값만큼 각 샤드에서 검색함)
3. dfs_query_then_fetch : 글로벌 문서 빈도를 계산하기위한 사전 쿼리를 수행
4. dfs_query_and_fetch : 글로벌 문서 빈도를 계산하기위한 사전 쿼리를 수행
5. count - 검색된 문서를 배제하고 hit수만 출력
6. scan - scroll과 같이 사용하여 검색 결과를 바로 보여주지 않고 scroll에 저장 후 _scroll_id를 사용하여 나중에 출력

Request Body 검색

QueryDSL 사용

기본 형식

http://localhost:9200/인덱스/타입/_search -d '

```
{
    "_source" : {
        "include" : "c*",
        "exclude" : "*ry"
    },
    "from" : 1,
    "size" : 10,
    "fields" : ["title","category"],
    "sort" : [{"category":{"order":"desc","mode":"min"}},"pages","title"],
    "highlight" : {
        "pre_tags" : ["<strong>"],
        "post_tags" : ["</strong>"],
        "fields" : {"author" : { } }
    },
    "query" : {
        "term" : { "_all" : "time"}
    }
}
```

옵션 영역

질의 영역

mode: 배열로 여러 값을 가진 경우 처리

1. min
2. max
3. avg
4. sum

검색 쿼리의 종류

텀레벨 쿼리	풀텍스트용 쿼리	복합쿼리	조인쿼리	GEO 쿼리	기타
term query **terms query** range query exists query prefix query wildcard query regexp query fuzzy query type query ids query	Match Query Match Phrase Query Match Phrase Prefix Query Multi Match Query Common Terms Query **Query String Query** Simple Query String Query	Constant Score Query Bool Query Dis Max Query Function Score Query Boosting Query Indices Query	Nested Query Has Child Query Has Parent Query Parent Id Query	GeoShape Query Geo Bounding Box Query Geo Distance Query Geo Distance Range Query Geo Polygon Query	More Like This Query Template Query Script Query

TERM 쿼리 ,TERM 필터

Term쿼리

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“term” : {
			“title” : “문재인”
		}
	}
}
```

Term 필터

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“filtered” : {
			“query “ : {
				“match_all” : {}
			}
		},
		“filter” : {
			“term”: {
				“tags”:”elastic”
			}
		}
	}
}
```

Terms 쿼리

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“terms” : {
			“title” : [“문재인”,”대통령”],
			“minimum_should_match”: 2
		}
	}
}
```

RANGE 쿼리,RANGE 필터

Range 쿼리

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“range” : {
			“created_on” : {
				“gt”:“2017-01-01”,
				“lt”:”2017-10-01”
			}
		}
	}
}
```

```
GET _search
{
  "query": {
    "range" : {
      "date" : {
        "gte" : "now-1d/d",
        "lt" :  "now/d"
      }
    }
  }
}
```

Range 필터

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“filtered” : {
			“query “ : {
				“match_all” : {}
			}
		},
		“filter” : {
			“range”: {
				“created_on” : {
					“gt”:“2017-01-01”,
					“lt”:”2017-10-01”
				}
			}
		}
	}
}
```

Prefix 쿼리, Wildcard 쿼리

Prefix 쿼리

```
GET /_search
{ "query": {
    "prefix" : { "user" :  { "value" : "ki", "boost" : 2.0 } }
  }
}
```

Wildcard 쿼리

```
GET /_search
{
    "query": {
        "wildcard" : { "user" : { "value" : "ki*y", "boost" : 2.0 } }
    }
}
```

QUERY_STRING 쿼리

기본 형식

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
   "query": {
      "query_string" : {
         "fields" : ["content", "name"],
         "query" : "this AND that OR thus"
      }
   }
}
```

혹은

```
{
   "query": {
      "query_string": {
         "query": "(content:this OR name:this) AND (content:that OR name:that)"
      }
   }
}
```

루씬이 사용하는 boolean query parser를 그대로 이용함

name:search^ AND (tags:lucene OR tages:”big data”~2) AND description:analytics

MATCH 쿼리,PHRASE 쿼리

Match 쿼리

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“match” : {
			“title” : {
				“query”:“Elastic Search”,
				“operator”:”and”
			}
		}
	}
}
```

Phrase 쿼리

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“match” : {
			“title” : {
				“type”: “phrase”,
				“query”: “elastic search”,
				“slop”: 1
			}
		}
	}
}
```

Multi Match 쿼리

http://localhost:9200/인덱스/타입/_search -d ‘

```
{
	“query” : {
		“multi_match” : {
			“query” : “Elastic Search”,
			“fields”:[”title^3”,”author”],
			 "tie_breaker": 0.3
			}
	}
}
```

tieBreaker

0 (default) : 가장 높은 점수를 기록한 필드의 score만 죄종 score에 반영
1 : 전체 필드의 score를 합산하여 반영

(맥스 필드 score) + (tieBreaker) * (나머지 필드 score의 합)

Bool 쿼리

```
POST _search
{
  "query": {
    "bool" : {
      "must" : {
        "term" : { "user" : "kimchy" }
      },
      "filter": {
        "term" : { "tag" : "tech" }
      },
      "must_not" : {
        "range" : {
          "age" : { "gte" : 10, "lte" : 20 }
        }
      },
      "should" : [
        { "term" : { "tag" : "wow" } },
        { "term" : { "tag" : "elasticsearch" } }
      ],
      "minimum_should_match" : 1,
      "boost" : 1.0
    }
  }
}
```

bool 식이 스코어에 영향을 미치지 않으려면
bool filter

```
GET _search
{
  "query": {
    "bool": {
      "filter": {
        "term": {
          "status": "active"
        }
      }
    }
  }
}
```

Combined 쿼리

Bool 쿼리와 Match 쿼리의 조합을 많이 사용

```
{
  "query": {
    "bool": {
      "must":     { "match": { "title": "quick" }},
      "must_not": { "match": { "title": "lazy"  }},
      "should": [
                  { "match": { "title": "brown" }},
                  { "match": { "title": "dog"   }}
      ]
    }
  }
}
```

```
{
   "query": {
      "bool": {
         "must": {
            "match": {
               "content": {
                  "query":    "full text search",
                  "operator": "and"
               }
            }
         },
         "should": [
            { "match": {
               "content": {
                  "query": "Elasticsearch",
                  "boost": 3
               }
            }},
            { "match": {
               "content": {
                  "query": "Lucene",
                  "boost": 2
               }
            }}
```

GEO Spatial 쿼리

```
GET /example/_search
{
   "query":{
      "bool": {
         "must": {
            "match_all": {}
         },
         "filter": {
            "geo_shape": {
               "location": {
                  "shape": {
                     "type": "envelope",
                     "coordinates" : [[13.0, 53.0], [14.0, 52.0]]
                  },
                  "relation": "within"
               }
            }
         }
      }
   }
}
```

INTERSECTS : 필드가 geometry쿼리와 교차하는 모든 문서
DISJOINT: geometry쿼리 범위밖의 모든 문서
WITHIN: geometry쿼리 범위내의 모든 문서
CONTAINS: geometry쿼리가 포함된 모든 문서

```
GET /my_locations/location/_search
{
   "query": {
      "bool" : {
         "must" : {
            "match_all" : {}
         },
         "filter" : {
            "geo_distance" : {
               "distance" : "12km",
               "pin.location" : [-70, 40]
            }
         }
      }
   }
}
```

```
GET /_search
{
   "query": {
      "bool" : {
         "must" : {
            "match_all" : {}
         },
         "filter" : {
            "geo_polygon" : {
               "person.location" : {
                  "points" : [
                     "40, -70",
                     "30, -80",
                     "20, -90"
                  ]
               }
            }
         }
      }
   }
}
```

NESTED 쿼리(중첩쿼리)

Nested 쿼리 : 중첩 매핑을 실시 할 경우 중첩쿼리가능

```
PUT /my_index
{
   "mappings": {
      "type1" : {
         "properties" : {
            "obj1" : {
               "type" : "nested"
            }
         }
      }
   }
}
```

```
GET /_search
{
   "query": {
      "nested" : {
         "path" : "obj1",
         "score_mode" : "avg",
         "query" : {
            "bool" : {
               "must" : [
               { "match" : {"obj1.name" : "blue"} },
               { "range" : {"obj1.count" : {"gt" : 5}} }
               ]
            }
         }
      }
   }
}
```

PARENT, CHILD 쿼리

parent-child relationship

```
PUT my_index
{
  "mappings": {
    "my_parent": {},
    "my_child": {
      "_parent": {
        "type": "my_parent"
      }
    }
  }
}

PUT my_index/my_parent/1
{
  "text": "This is a parent document"
}

PUT my_index/my_child/2?parent=1
{
  "text": "This is a child document"
}

PUT my_index/my_child/3?parent=1&refresh=true
{
  "text": "This is another child document"
}
```

Has_Child 쿼리

```
GET my_index/my_parent/_search
{
  "query": {
    "has_child": {
      "type": "my_child",
      "query": {
        "match": {
          "text": "child document"
        }
      }
    }
  }
}
```

children aggregation

```
GET my_index/_search
{
  "query": {
    "parent_id": {
      "type": "my_child",
      "id": "1"
    }
  },
  "aggs": {
    "parents": {
      "terms": {
        "field": "_parent",
        "size": 10
      }
    }
  },
  "script_fields": {
    "parent": {
      "script": {
        "inline": "doc['_parent']"
      }
    }
  }
}
```

제약조건

1. 부모와 자식의 타입은 달라야함.
2. _parent.type 설정은 아직 존재하지 않은 타입만가능
   (타입을 만든 이후에는 상위 유형을 만들수 없음)
3. 동일한 샤드에 색인되어야함 (parent id 라우팅에 사용)

SPAN 쿼리 (근접 연산자 쿼리)

span_term query
- span 쿼리의 기본 단위 , term query 와 동일

span_multi query
- term, range, prefix, wildcard, regexp, or fuzzy query의 래핑 쿼리

span_first query
- 필드 시작 부근에서 일치

span_near query
- 질의된 텀들이 서로 가까이 있어야 검색되는 쿼리

span_or query
- 질의된 텀을 OR로 검색

span_not query
- 질의된 텀을 NOT으로 검색

span_containing query
- 두개의 span 쿼리를 묶음

span_within query
- 두개의 span 쿼리를 묶음

SPAN 쿼리 (근접 연산자 쿼리)

span_near query

```
{
   "query": {
      "span_near" : {
         "clauses" : [
            { "span_term" : { "field" : "value1" } },
            { "span_term" : { "field" : "value2" } },
            { "span_term" : { "field" : "value3" } }
         ],
         "slop" : 12,
         "in_order" : false
      }
   }
}
```

span containing query

```
{
   "query": {
      "span_containing" : {
         "A" : {
            "span_term" : { "field1" : "foo" }
         },
         "B" : {
            "span_near" : {
               "clauses" : [
                  { "span_term" : { "field1" : "bar" } },
                  { "span_term" : { "field1" : "baz" } }
               ],
               "slop" : 5,
               "in_order" : true
            }
         }
      }
   }
}
```

A의 일치 항목에 포함된 B의 일치 항목 반환

span_within query

```
{
   "query": {
      "span_within" : {
         "A" : {
            "span_term" : { "field1" : "foo" }
         },
         "B" : {
            "span_near" : {
               "clauses" : [
                  { "span_term" : { "field1" : "bar" } },
                  { "span_term" : { "field1" : "baz" } }
               ],
               "slop" : 5,
               "in_order" : true
            }
         }
      }
   }
}
```

B의 일치 항목에 포함된 A의 일치 항목 반환

Function Score 쿼리

1) function score 쿼리

```
{
   "query": {
      "function_score": {
       "query": { "match_all": {} },
       "boost": "5",
       "functions": [
          {
             "filter": { "match": { "test": "bar" } },
             "random_score": {},
             "weight": 23
          },
          {
             "filter": { "match": { "test": "cat" } },
             "weight": 42
          }
       ],
       "max_boost": 42,
       "score_mode": "max",
       "boost_mode": "multiply",
       "min_score" : 42
      }
   }
```

2) score script 쿼리

```
{
   "query": {
      "function_score": {
         "query": {
            "match": { "message": "elasticsearch" }
         },
         "script_score" : {
            "script" : {
              "inline": "Math.log(2 + doc['likes'].value)"
            }
         }
      }
   }
}
```

3) Field Value Factor

```
{
   "query": {
      "function_score": {
         "field_value_factor": {
            "field": "likes",
            "factor": 1.2,
            "modifier": "sqrt",
            "missing": 1
         }
      }
   }
}
```

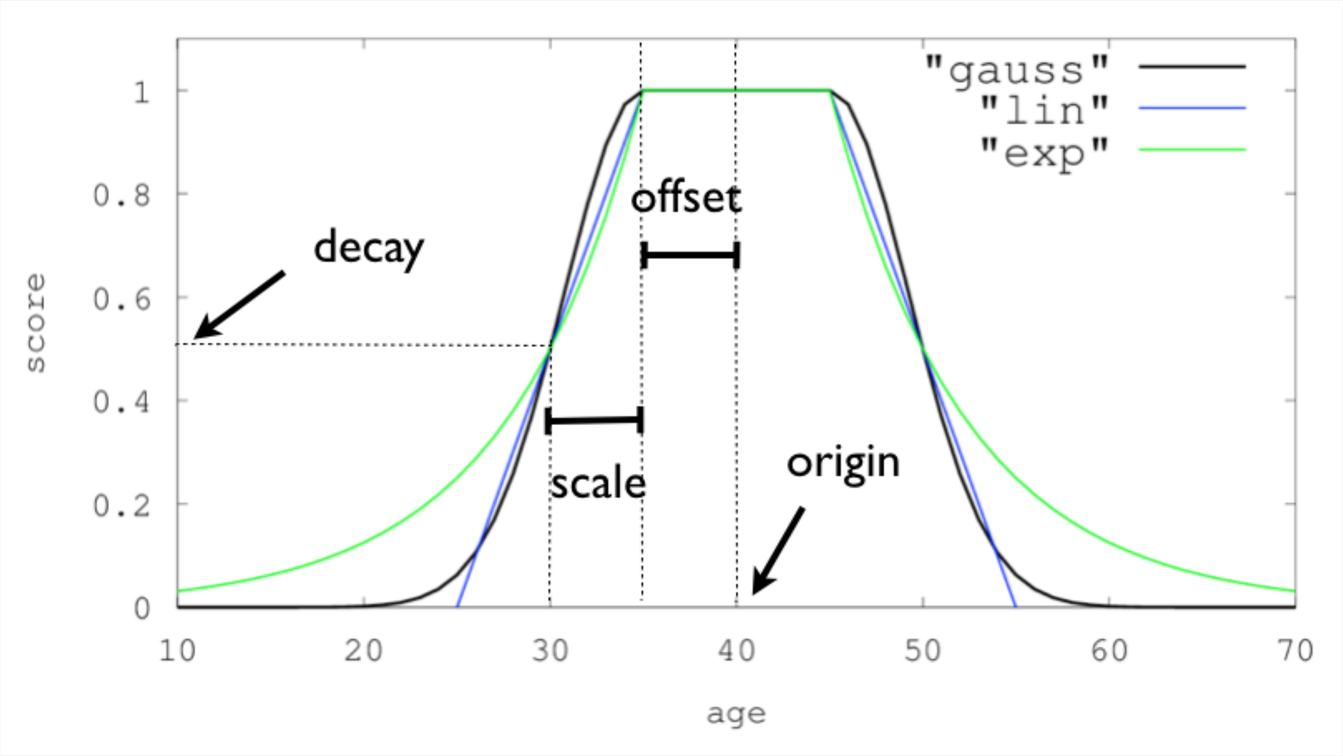

A의 일치 항목에 포함된 B의 일치 항목 반환

B의 일치 항목에 포함된 A의 일치 항목 반환

Decay functions 쿼리

특정 기준점에서 가까워지거나 멀어질때 스코어를 일정 값으로 쇠퇴(Decay) 시키는 함수

```
{
   "query": {
      "function_score": {
         "gauss": {
            "date": {
                  "origin": "2013-09-17",
                  "scale": "10d",
                  "offset": "5d",
                  "decay" : 0.5
            }
         }
      }
   }
}
```



A의 일치 항목에 포함된 B의 일치 항목 반환

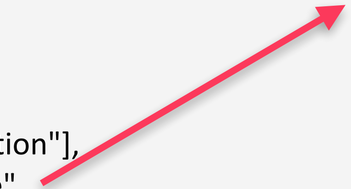
B의 일치 항목에 포함된 A의 일치 항목 반환

MORE LIKE THIS 쿼리

주어진 문서 세트와 가장 유사한 문서를 텀벡터 기반으로 검색

```
PUT /imdb
{
  "mappings": {
    "movies": {
      "properties": {
        "title": {
          "type": "text",
          "term_vector": "yes"
        },
        "description": {
          "type": "text"
        },
        "tags": {
          "type": "text",
          "fields" : {
            "raw": {
              "type" : "text",
              "analyzer": "keyword",
              "term_vector" : "yes"
            }
          }
        }
      }
    }
  }
}
```

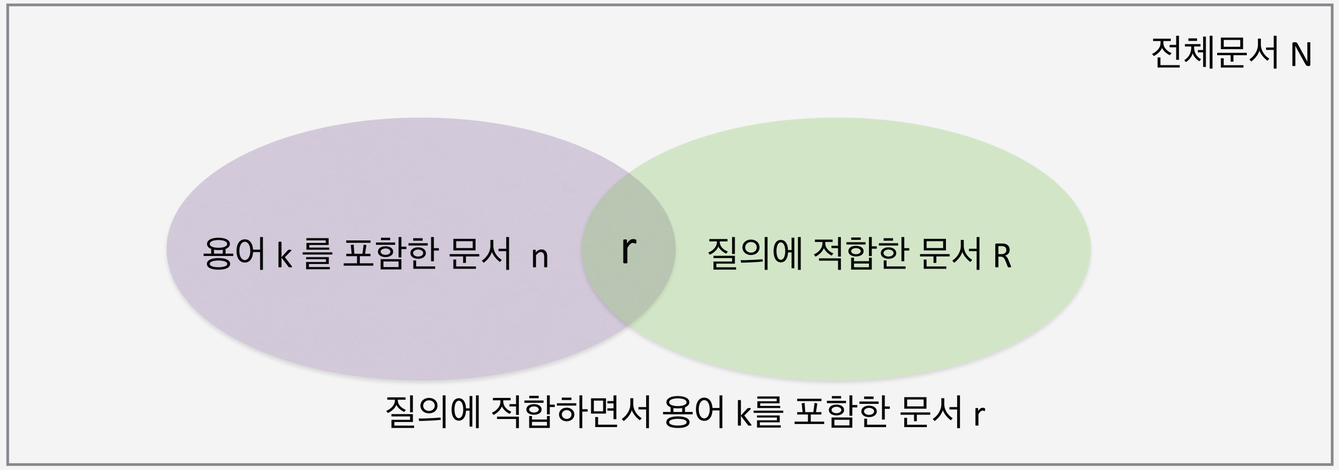
```
GET /_search
{
  "query": {
    "more_like_this" : {
      "fields" : ["title", "description"],
      "like" : "Once upon a time",
      "min_term_freq" : 1,
      "max_query_terms" : 12
    }
  }
}
```



```
"like" : [
    {
      "_index" : "imdb",
      "_type" : "movies",
      "_id" : "1"
    },
    {
      "_index" : "imdb",
      "_type" : "movies",
      "_id" : "2"
    },
    "and potentially some more text here as well"
    ],
    "min_term_freq" : 1,
    "max_query_terms" : 12
```

Okapi BM25

$$\text{bm25}(d) = \sum_{t \in q, f_{t,d} > 0} \log\left(1 + \frac{N - df_t + 0.5}{df_t + 0.5}\right) \cdot \frac{f_{t,d}}{f_{t,d} + k \cdot (1 - b + b\frac{\text{l}(d)}{avgdl})}$$



https://www.slideshare.net/MinsubYim/ir-modeling

k1 : 단어의 빈도가 점수에 얼마나 중요한지 (기본값 1.2)
b : 문서의 길이가 점수에 미치는 영향도 (기본값 0.75)
discount_overlaps : 중복되서 토큰이 발견할 경우 어떻게 길이를 정규화 할지(기본값 true)

Okapi BM25 설정 방법

```
{
  "book" : {
    "properties" : {
      "title" : { "type" : "text", "similarity" : "my_similarity" }
    }
}

"similarity" : {
  "my_similarity" : {
    "type" : "BM25",
    "k1" : 1.2,
    "b" : 0.75,
    "discount_overlaps" : false
  }
}
```

감사합니다.

다음 주제 : 7일차 ElasticSearch Chapter 3
Aggregation
기타 고급 검색 기법