**Elasticsearch 시작하기**

Elasticsearch 시작하기 (Elasticsearch: Getting Started) 데모에 사용된 코드

비디오 링크 : <https://www.elastic.co/kr/webinars/getting-started-elasticsearch>

#--------------------------------------------------

# 1. library 인덱스 생성

#--------------------------------------------------

PUT library

{

"settings": {

"number\_of\_shards": 1,

"number\_of\_replicas": 0

}

}

#--------------------------------------------------

# 2. Bulk 색인

# 다량의 도큐먼트를 한꺼번에 색인 할 때는 반드시 bulk API를 사용

# 알아보기(Learn) > 문서(Docs) > Elasticsearch Reference > Document APIs > Bulk API

# https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-bulk.html

#--------------------------------------------------

POST library/books/\_bulk

{"index":{"\_id":1}}

{"title":"The quick brow fox","price":5,"colors":["red","green","blue"]}

{"index":{"\_id":2}}

{"title":"The quick brow fox jumps over the lazy dog","price":15,"colors":["blue","yellow"]}

{"index":{"\_id":3}}

{"title":"The quick brow fox jumps over the quick dog","price":8,"colors":["red","blue"]}

{"index":{"\_id":4}}

{"title":"brow fox brown dog","price":2,"colors":["black","yellow","red","blue"]}

{"index":{"\_id":5}}

{"title":"Lazy dog","price":9,"colors":["red","blue","green"]}

#--------------------------------------------------

# 3. 검색 (\_search)

#--------------------------------------------------

#--------------------------------------------------

# 3-1. 전체 도큐먼트 검색

# 옵션을 주지 않으면 기본적으로 인덱스의 \*전체\* 도큐먼트를 검색

#--------------------------------------------------

GET library/\_search

#--------------------------------------------------

# 3-2. fox 가 포함된 도큐먼트 검색

#--------------------------------------------------

GET library/\_search

{

"query": {

"match": {

"title": "fox"

}

}

}

#--------------------------------------------------

# 3-3. fox 또는 dog 가 포함된 도큐먼트 검색

#--------------------------------------------------

GET library/\_search

{

"query": {

"match": {

"title": "quick dog"

}

}

}

#--------------------------------------------------

# 3-4. "quick dog" 구문이 포함된 도큐먼트 검색

#--------------------------------------------------

GET library/\_search

{

"query": {

"match\_phrase": {

"title": "quick dog"

}

}

}

#--------------------------------------------------

# 3-5. 검색 결과에 "relevance" 알고리즘을 이용한 랭킹 적용 (\_score)

# 알아보기(Learn) > 문서(Docs) > Elasticsearch: The Definitive Guide >

# Getting Started > Sorting and Relevance > What Is Relevance

# https://www.elastic.co/guide/en/elasticsearch/guide/current/relevance-intro.html

#--------------------------------------------------

GET library/\_search

{

"query": {

"match": {

"title": "quick"

}

}

}

#--------------------------------------------------

# 4. 복합 쿼리 - bool 쿼리를 이용한 서브쿼리 조합

#--------------------------------------------------

#--------------------------------------------------

# 4-1. must: "quick" 와 "lazy dog" 가 포함된 모든 문서 검색

#--------------------------------------------------

GET /library/\_search

{

"query": {

"bool": {

"must": [

{

"match": {

"title": "quick"

}

},

{

"match\_phrase": {

"title": {

"query": "lazy dog"

}

}

}

]

}

}

}

#--------------------------------------------------

# 4-2. must\_not: "quick" 또는 "lazy dog" 가 포함되지 않은 문서 검색

#--------------------------------------------------

GET /library/\_search

{

"query": {

"bool": {

"must\_not": [

{

"match": {

"title": "lazy"

}

},

{

"match\_phrase": {

"title": {

"query": "quick dog"

}

}

}

]

}

}

}

#--------------------------------------------------

# 4-3. 특정 쿼리에 대한 가중치 조절 (boost)

# 4-3-1. should - 반드시 매칭 될 필요는 없지만, 매칭 되는 경우 더 높은 스코어

#--------------------------------------------------

GET /library/\_search

{

"query": {

"bool": {

"should": [

{

"match\_phrase": {

"title": "quick dog"

}

},

{

"match\_phrase": {

"title": {

"query": "lazy dog",

"boost": 3

}

}

}

]

}

}

}

#--------------------------------------------------

# 4-3-2. must + should

#--------------------------------------------------

GET /library/\_search

{

"query": {

"bool": {

"should": [

{

"match": {

"title": "lazy"

}

}

],

"must": [

{

"match": {

"title": "dog"

}

}

]

}

}

}

#--------------------------------------------------

# 5. highlight - 검색어와 매칭 된 부분을 하이라이트로 표시

# 검색 결과값이 크고 여러 필드를 사용하는 경우 유용함

#--------------------------------------------------

#--------------------------------------------------

# 5-1. highlight

#--------------------------------------------------

GET /library/\_search

{

"query" : {

"bool": {

"should" : [

{

"match\_phrase": {

"title": {

"query" : "quick dog",

"boost": 2

}

}

},

{

"match\_phrase": {

"title": {

"query" : "lazy dog"

}

}

}

]

}

},

"highlight" : {

"fields" : {

"title": { }

}

}

}

#--------------------------------------------------

# 6. filter - 검색 결과의 sub-set 도출

# 스코어를 계산하지 않고 캐싱되어 쿼리보다 대부분 빠름

#--------------------------------------------------

#--------------------------------------------------

# 6-1. (bool) must + filter 사용

#--------------------------------------------------

GET /library/\_search

{

"query": {

"bool": {

"must": [

{

"match": {

"title": "dog"

}

}

],

"filter": {

"range": {

"price": {

"gte": 5,

"lte": 10

}

}

}

}

}

}

#--------------------------------------------------

# 6-2. 스코어가 필요 없는 경우 filter 만 사용

# 알아보기(Learn) > 문서(Docs) > Elasticsearch: The Definitive Guide >

# Search in Depth > Structured Search

# https://www.elastic.co/guide/en/elasticsearch/guide/current/structured-search.html

#--------------------------------------------------

GET /library/\_search

{

"query": {

"bool": {

"filter": {

"range": {

"price": {

"gt": 5

}

}

}

}

}

}

#--------------------------------------------------

# 7. 분석 - Analysis (\_analyze)

#--------------------------------------------------

#--------------------------------------------------

# 7-1 Tokenizer 을 통해 문장을 검색어 텀(term)으로 쪼갬

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "standard",

"text": "Brown fox brown dog"

}

#--------------------------------------------------

# 7-2 Filter(토큰필터) 를 통해 쪼개진 텀들을 가공

# 7-2-1. lowercase - 소문자로 변경

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "standard",

"filter": [

"lowercase"

],

"text": "Brown fox brown dog"

}

#--------------------------------------------------

# 7-2-2. unique - 중복 텀 제거

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "standard",

"filter": [

"lowercase",

"unique"

],

"text": "Brown brown brown fox brown dog"

}

#--------------------------------------------------

# 7-3. (Tokenizer + Filter) 대신 Analyzer 사용

#--------------------------------------------------

GET library/\_analyze

{

"analyzer": "standard",

"text": "Brown fox brown dog"

}

#--------------------------------------------------

# 8. 분석 과정 이해하기

#--------------------------------------------------

#--------------------------------------------------

# 8-1. 복합적인 문장 분석 - T:standard, F:lowercase

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "standard",

"filter": [

"lowercase"

],

"text": "THE quick.brown\_FOx jumped! $19.95 @ 3.0"

}

#--------------------------------------------------

# 8-2. 복합적인 문장 분석 - T:letter, F:lowercase

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "letter",

"filter": [

"lowercase"

],

"text": "THE quick.brown\_FOx jumped! $19.95 @ 3.0"

}

#--------------------------------------------------

# 8-3. Email, URL 분석 - T:standard

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "standard",

"text": "elastic@example.com website: https://www.elastic.co"

}

#--------------------------------------------------

# 8-4. Email, URL 분석 - T:uax\_url\_email

#--------------------------------------------------

GET library/\_analyze

{

"tokenizer": "uax\_url\_email",

"text": "elastic@example.com website: https://www.elastic.co"

}

#--------------------------------------------------

# 알아보기(Learn) > 문서(Docs) > Elasticsearch: The Definitive Guide >

# Search in Depth > Full-Text Search > Controlling Analysis

# https://www.elastic.co/guide/en/elasticsearch/guide/master/\_controlling\_analysis.html

#--------------------------------------------------

#--------------------------------------------------

# 9. 애그리게이션 - 집계 (Aggregation)

#--------------------------------------------------

#--------------------------------------------------

# 9-1. terms aggs 를 이용한 colors.keyword 필드 값 집계

#--------------------------------------------------

GET library/\_search

{

"size": 0,

"aggs": {

"popular-colors": {

"terms": {

"field": "colors.keyword"

}

}

}

}

#--------------------------------------------------

# 9-2. 검색(query)과 애그리게이션(aggs) 동시에 사용

#--------------------------------------------------

GET library/\_search

{

"query": {

"match": {

"title": "dog"

}

},

"aggs": {

"popular-colors": {

"terms": {

"field": "colors.keyword"

}

}

}

}

#--------------------------------------------------

# 9-3. 여러개의 애그리게이션, sub-aggs 사용

#--------------------------------------------------

GET library/\_search

{

"size": 0,

"aggs": {

"price-statistics": {

"stats": {

"field": "price"

}

},

"popular-colors": {

"terms": {

"field": "colors.keyword"

},

"aggs": {

"avg-price-per-color": {

"avg": {

"field": "price"

}

}

}

}

}

}

#--------------------------------------------------

# 10. 도큐먼트 업데이트

# 동일한 URL에 데이터 입력시 기존 데이터 대체됨

#--------------------------------------------------

GET library/books/1

#--------------------------------------------------

# 10-1. POST 메소드 이용

#--------------------------------------------------

POST library/books/1

{

"title": "The quick brow fox",

"price": 10,

"colors": ["red","green","blue"]

}

#--------------------------------------------------

# 10-1. \_update API 이용

#--------------------------------------------------

POST library/books/1/\_update

{

"doc": {

"title": "The quick fantastic fox"

}

}

#--------------------------------------------------

# 11. 매핑 (Mapping)

# 데이터가 색인될 때 Elasticsearch 스스로 매핑을 정의함

#--------------------------------------------------

#--------------------------------------------------

# 11-1. 매핑 확인

#--------------------------------------------------

GET library/\_mapping

#--------------------------------------------------

# 11-2. 직접 매핑을 설정한 인덱스 생성

#--------------------------------------------------

PUT famous-librarians

{

"settings": {

"number\_of\_shards": 2,

"number\_of\_replicas": 0,

"analysis": {

"analyzer": {

"my-desc-analyzer": {

"type": "custom",

"tokenizer": "uax\_url\_email",

"filter": [

"lowercase"

]

}

}

}

},

"mappings": {

"librarian": {

"properties": {

"name": {

"type": "text"

},

"favourite-colors": {

"type": "keyword"

},

"birth-date": {

"type": "date",

"format": "year\_month\_day"

},

"hometown": {

"type": "geo\_point"

},

"description": {

"type": "text",

"analyzer": "my-desc-analyzer"

}

}

}

}

}

#--------------------------------------------------

# 11-3-1. 예제 데이터(1)

#--------------------------------------------------

PUT famous-librarians/librarian/1

{

"name": "Sarah Byrd Askew",

"favourite-colors": [

"Yellow",

"light-grey"

],

"birth-date": "1877-02-15",

"hometown": {

"lat": 32.349722,

"lon": -87.641111

},

"description": "An American public librarian who pioneered the establishment of county libraries in the United States - https://en.wikipedia.org/wiki/Sarah\_Byrd\_Askew"

}

#--------------------------------------------------

# 11-3-2. 예제 데이터(2)

#--------------------------------------------------

PUT famous-librarians/librarian/2

{

"name": "John J. Beckley",

"favourite-colors": [

"Red",

"off-white"

],

"birth-date": "1757-08-07",

"hometown": {

"lat": 51.507222,

"lon": -0.1275

},

"description": "An American political campaign manager and the first Librarian of the United States Congress, - https://en.wikipedia.org/wiki/John\_J.\_Beckley"

}

#--------------------------------------------------

# 11-4-1. query\_string - keyword 필드 확인

# yellow - X

# Yellow - O

#--------------------------------------------------

GET famous-librarians/\_search

{

"query": {

"query\_string": {

"fields": [

"favourite-colors"

],

"query": "yellow OR off-white"

}

}

}

#--------------------------------------------------

# 11-4-2. range - 날짜 범위 검색

#--------------------------------------------------

GET famous-librarians/\_search

{

"query": {

"bool": {

"must": [

{

"match\_all": {}

}

],

"filter": {

"range": {

"birth-date": {

"gte": "now-200y",

"lte": "2000-01-01"

}

}

}

}

}

}

#--------------------------------------------------

# 11-4-3. geo\_distance - 특정 지점에서 반경 100km 거리 검색

#--------------------------------------------------

GET famous-librarians/\_search

{

"query": {

"bool": {

"filter": {

"geo\_distance": {

"distance": "100km",

"hometown": {

"lat": 32.41,

"lon": -86.92

}

}

}

}

}

}