

3

[3주차] 4장 엘라스틱서치:검색

4.1 쿼리 컨텍스트와 필터 컨텍스트

- 쿼리 컨텍스트

```

1 GET kibana_sample_data_ecommerce/_search
2 {
3   "query": {
4     "match": {
5       "category": "clothing"
6     }
7   }
8 }

1 {
2   "took": 10,
3   "timed_out": false,
4   "_shards": {
5     "total": 1,
6     "successful": 1,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 3927, 3927개의 도큐먼트를 찾음
13      "relation": "eq"
14    },
15    "max_score": 0.20545526,
16    "hits": [
17      {
18        "_index": "kibana_sample_data_ecommerce",
19        "_type": "_doc",
20        "_id": "oSbEFX8BjJzEbmEHZ_xS",
21        "_score": 0.20545526, 높은 스코어(요청한 검색과 유사도) 순으로 정렬됨
22        "_source": {
23          "category": [
24            "Men's Clothing"
25          ],
26          "currency": "EUR",
27          "customer_first_name": "Eddie",
28          "customer_full_name": "Eddie Underwood",
29          "customer_gender": "MALE",
30          "customer_id": 38,

```

- 질의에 대한 유사도 계산 기준으로 정확한 결과 반환
- 결과값: 연관성에 따른 스코어
- 필터 컨텍스트

```

1 GET kibana_sample_data_ecommerce/_search
2 {
3   "query": {
4     "bool": {
5       "filter": [
6         {
7           "term": {
8             "day_of_week": "Friday"
9           }
10        }
11      ]
12    }
13  }
14 }
15
16 }, hits: 770
17 "max_score" : 0.0,
18 "hits" : [
19   {
20     "_index" : "kibana_sample_data_ecommerce",
21     "_type" : "_doc",
22     "_id" : "tybEFX8BjJzEbmEHZ_xT",
23     "_score" : 0.0, 스코어는 0
24     "_source" : {
25       "category" : [
26         "Women's Shoes",
27         "Women's Clothing"
28       ],
29       "currency" : "EUR",
30       "customer_first_name" : "Mary",
31       "customer_full_name" : "Mary Barber",
32       "customer_gender" : "FEMALE",
33       "customer_id" : 20,
34       "customer_last_name" : "Barber",
35       "customer_phone" : "",
36       "day_of_week" : "Friday",
37       "day_of_week_i" : 4,
38       "email" : "mary@barber-family.zzz",
39       "manufacturer" : [
40         "Goshaevs"
41       ]
42     }
43   }
44 ]

```

- 유사도 계산 X (스코어 계산 X)
 - 속도 향상
 - 캐시
 - 일치 여부에 따른 결과만 반환
 - 결과값: 예/아니요
 - 쿼리/필터 조합으로 사용하는 추세임
- #### 4.2 쿼리 스트링과 쿼리 DSL
- 쿼리 스트링:
 - 한 줄의 간단한 쿼리
 - 복잡해지면 가독성이 떨어질 수 있음

```

1 GET kibana_sample_data_ecommerce/_search?q
  =customer_full_name:Mary
2 {
3   "took" : 7,
4   "timed_out" : false,
5   "_shards" : {
6     "total" : 1,
7     "successful" : 1,
8     "skipped" : 0,
9     "failed" : 0
10  },
11  "hits" : {
12    "total" : {
13      "value" : 154,
14      "relation" : "eq"
15    },
16    "max_score" : 3.4912553,
17    "hits" : [
18      {
19        "_index" : "kibana_sample_data_ecommerce",
20        "_type" : "_doc",
21        "_id" : "oibEFX8BjJzEbmEHZ_xT",
22        "_score" : 3.4912553,
23        "_source" : {
24          "category" : [
25            "Women's Clothing"
26          ],
27          "currency" : "EUR",
28          "customer_first_name" : "Mary",
29          "customer_full_name" : "Mary Bailey"
30        }
31      }
32    ]
33  }
34 }

```

- 쿼리 DSL:

```

1 GET kibana_sample_data_ecommerce/_search
2 {
3   "query": {
4     "match": {
5       "customer_full_name": "Mary"
6     }
7   }
8 }
9 {
10  "took" : 3,
11  "timed_out" : false,
12  "_shards" : {
13    "total" : 1,
14    "successful" : 1,
15    "skipped" : 0,
16    "failed" : 0
17  },
18  "hits" : {
19    "total" : {
20      "value" : 154,
21      "relation" : "eq"
22    },
23    "max_score" : 3.4912553,
24    "hits" : [
25      {
26        "_index" : "kibana_sample_data_ecommerce",
27        "_type" : "_doc",
28        "_id" : "oibEFX8BjJzEbmEHZ_xT",
29        "_score" : 3.4912553,
30        "_source" : {
31          "category" : [
32            "Women's Clothing"
33          ],
34          "currency" : "EUR",
35          "customer_first_name" : "Mary",
36          "customer_full_name" : "Mary Bailey"
37        }
38      }
39    ]
40  }
41 }

```

- 복잡한 쿼리
- 엘라스틱서치에서 제공하는 쿼리 전용 언어
- JSON 기반의 직관적인 언어

4.3 유사도 스코어

- 질의문과 문서의 유사도 표현값

- 높은 스코어 → 높은 연관
- 쿼리에 `explain` 옵션을 사용해 유사도 스코어 알고리즘을 알 수 있다.
- 기본적으로 BM25 알고리즘 사용

BM25 알고리즘

- 검색, 추천에 많이 사용
- TF + IDF + 문서 길이 를 고려한 알고리즘
- IDF 계산
 - 문서 빈도 : 특정 용어가 얼마나 자주 등장했는지 의미
 - 문서 빈도가 높다 → 중요한 용어가 아님 (ex. 관사, 부사)
 - 문서 빈도의 역수(IDF): 도큐먼트 내에 발생 빈도가 적을수록 가중치를 높여주는 것

```
{
  "value" : 7.1974354,
  "description" : "idf, computed as log(1 + (N - n + 0.5) / (n + 0.5)) from:",
  "details" : [
    {
      "value" : 3,
      "description" : "n, number of documents containing term",
      "details" : [ ]
    },
    {
      "value" : 4675,
      "description" : "N, total number of documents with field",
      "details" : [ ]
    }
  ]
}
```

n: 3, N: 4675

- n: 검색했던 용어가 몇 개의 도큐먼트에 있는지
 - N: 인덱스의 전체 도큐먼트 수
- TF 계산
 - 용어 빈도 : 특정 용어가 하나의 도큐먼트에 얼마나 많이 등장했는지 의미
 - 용어 빈도가 높다 → 주제 용어

```
{
  "value" : 0.52217203,
  "description" : "tf, computed as freq / (freq + k1 * (1 - b + b * dl / avgdl)) from:",
  "details" : [
    {
      "value" : 1.0,
      "description" : "freq, occurrences of term within document",
      "details" : [ ]
    },
    {
      "value" : 1.2,
      "description" : "k1, term saturation parameter",
      "details" : [ ]
    },
    {
      "value" : 0.75,
      "description" : "b, length normalization parameter",
      "details" : [ ]
    },
    {
      "value" : 5.0,
      "description" : "dl, length of field",
      "details" : [ ]
    },
    {
      "value" : 7.3161497,
      "description" : "avgdl, average length of field",
      "details" : [ ]
    }
  ]
}
```

- freq : 도큐먼트 내에서 용어가 나온 횟수
- k1, d : 알고리즘을 정규화하기 위한 가중치
- dl : 필드 길이
- avgdl : 전체 도큐먼트에서 평균 필드 길이
- BM25알고리즘 최종 값 = $IDF * TF * boost(2.2)$

```
    "value" : 0.200000,
    "description" : "score(freq=1.0), computed as boost * idf * tf from:",
    "details" : [
      {
        "value" : 2.2,
        "description" : "boost",
        "details" : [ ]
      },
      {

```

- boost: 고정값 = 2.2

4.4 검색 쿼리 search query

- 종류
 - 리프 쿼리 leaf query
 - 특정 필드에서 용어 찾는 용도
 - 매치, 용어, 범위 쿼리
 - 복합 쿼리 compound query
 - 쿼리 조합
 - 논리 쿼리
- 전문 쿼리
 - 전문 검색용
 - 텍스트 타입 매핑
 - 전문 쿼리 과정

```

1 PUT qindex/_doc/1
2 {
3   "contents": "I Love Elastic Stack."
4 }

```

qindex 인덱스 생성
 -> contents 필드를 갖는 도큐먼트 인덱싱
 -> contents 필드는 텍스트 타입으로 매핑
 -> 문자열 분석기로 인해 토큰화

i

love

elastic

stack

```

1 {
2   "_index" : "qindex",
3   "_type" : "_doc",
4   "_id" : "1",
5   "_version" : 1,
6   "result" : "created",
7   "_shards" : {
8     "total" : 2,
9     "successful" : 1,
10    "failed" : 0
11  },
12   "_seq_no" : 0,
13   "_primary_term" : 1
14 }
15

```

일반적인 분석기는 대문자를 소문자로 변경한다.

```

1 GET qindex/_search
2 {
3   "query": {
4     "match": {
5       "contents": "elastic world"
6     }
7   }
8 }

```

elastic
world

토큰화된 검색어와 토큰화된 문서용어들이 매칭
 -> 스코어 계산 -> 검색

```

1 {
2   "took" : 16,
3   "timed_out" : false,
4   "_shards" : {
5     "total" : 1,
6     "successful" : 1,
7     "skipped" : 0,
8     "failed" : 0
9   },
10  "hits" : {
11    "total" : {
12      "value" : 1,
13      "relation" : "eq"
14    },
15    "max_score" : 0.2876821,
16    "hits" : [
17      {
18        "_index" : "qindex",
19        "_type" : "_doc",
20        "_id" : "1",
21        "_score" : 0.2876821,
22        "_source" : {
23          "contents" : "I Love Elastic Stack."
24        }
25      }
26    ]
27  }
28 }

```

- 구글, 네이버 검색 방식
- 종류: 매치 쿼리(match), 매치 프레이즈 쿼리(match phrase), 멀티 매치 쿼리(multi-match), 쿼리 스트링 쿼리(query string)
- 매치 쿼리

```

1 GET kibana_sample_data_ecommerce/_search
2 {
3   "_source": ["customer_first_name"],
4   "query": {
5     "match": {
6       "customer_first_name": {
7         "query": "mary bailey",
8         "operator": "and"
9       }
10    }
11  }
12 }

```

mary
bailey

and

```

1 {
2   "took" : 13,
3   "timed_out" : false,
4   "_shards" : {
5     "total" : 1,
6     "successful" : 1,
7     "skipped" : 0,
8     "failed" : 0
9   },
10  "hits" : {
11    "total" : {
12      "value" : 0,
13      "relation" : "eq"
14    },
15    "max_score" : null,
16    "hits" : [ ]
17  }
18 }
19

```

operator 파라미터를 and 로 변경하면 검색어를 모두 포함한 검색을 한다

■ 전문 쿼리의 가장 기본

- 특정 용어나 용어들 검색용
- 매치 프레이즈 쿼리
 - 2개 이상의 단어가 연결되어 만들어진 단어를 검색 (순서도 맞아햐함)
 - 검색시 많은 리소스를 요구하기 때문에 자주 사용하는 것은 좋지 않다.
- 멀티 매치 쿼리
 - 어떤 필드가 저장되었는지 모를 때 사용
 - 전문 검색 쿼리
 - 1개 이상의 필드에 쿼리 요청 가능
 - 개별 스코어를 구한 후 가장 큰 값이 대표 스코어값이 된다.
 - 검색하려는 필드가 많을 때 *(와일드카드)를 사용해서 유사한 복수 필드를 선택할 수 있다.

```

1 GET kibana_sample_data_ecommerce/_search
2 {
3   "_source": ["customer_first_name",
4               "customer_last_name", "customer_full_name"],
5   "query": {
6     "multi_match": {
7       "query": "mary",
8       "fields": "customer_*_name"
9     }
10  }

```

- 필드에 가중치 두기 = 부스팅 기법
 - 중요한 필드에 가중치를 둘 수 있다.


```
GET kibana_sample_data_ecommerce/_search
{
  "query": {
    "multi_match": {
      "query": "mary",
      "fields": [
        "customer_full_name^2",
        "customer_first_name",
        "customer_last_name"
      ]
    }
  }
}
```

- 필드이름^n ⇒ 필드의 스코어 값을 n 배 해준다.
- 용어 쿼리
 - 정확히 일치하는 용어 검색용
 - 키워드 타입 매핑
 - 용어 쿼리 과정

```
1 PUT qindex/_doc/1
2 {
3   "category": "Tech"
4 }
```

qindex 인덱스 생성
 -> category 필드를 갖는 도큐먼트 인덱싱
 -> category는 키워드타입으로 매핑
 -> 분석기X

```
1 {
2   "_index" : "qindex",
3   "_type" : "_doc",
4   "_id" : "1",
5   "_version" : 1,
6   "result" : "created",
7   "_shards" : {
8     "total" : 2,
9     "successful" : 1,
10    "failed" : 0
11  },
12   "_seq_no" : 0,
13   "_primary_term" : 1
14 }
15
```

<pre> 1 GET qindex/_search 2 { 3 "query": { 4 "term": { 5 "category": { 6 "value": "tech" 7 } 8 } 9 } 10 } </pre>	<pre> 1 { 2 "took" : 5, 3 "timed_out" : false, 4 "_shards" : { 5 "total" : 1, 6 "successful" : 1, 7 "skipped" : 0, 8 "failed" : 0 9 }, 10 "hits" : { 11 "total" : { 12 "value" : 1, 13 "relation" : "eq" 14 }, 15 "max_score" : 0.2876821, 16 "hits" : [17 { 18 "_index" : "qindex", 19 "_type" : "_doc", 20 "_id" : "1", 21 "_score" : 0.2876821, 22 "_source" : { 23 "category" : "Tech" 24 } 25 } 26] 27 } 28 } </pre>
---	---

term -> 용어 검색 쿼리는 분석기를 거치지 않고 그대로 사용한다

"Tech" vs "tech"

분석되지 않은 문서 용어와 분석되지 않은 검색어를 비교

- 키워드, 숫자형, 날짜, 범위형 타입의 필드에서 검색할 때 사용
- 종류: 용어 쿼리(term), 용어들 쿼리(terms), 퍼지 쿼리(fuzzy)
- 용어 쿼리
 - 텍스트 타입 필드에 용어 쿼리사용은 올바르지 않다

```
1 GET kibana_sample_data_ecommerce/_search
2 {
3   "_source": ["customer_full_name"],
4   "query": {
5     "term": {
6       "customer_full_name": "Mary Bailey"
7     }
8   }
9 }
10 }
```

"Mary Baily" vs ["mary", "bailey"]

```
1 {
2   "took" : 0,
3   "timed_out" : false,
4   "_shards" : {
5     "total" : 1,
6     "successful" : 1,
7     "skipped" : 0,
8     "failed" : 0
9   },
10  "hits" : {
11    "total" : {
12      "value" : 0,
13      "relation" : "eq"
14    },
15    "max_score" : null,
16    "hits" : [ ]
17  }
18 }
19 }
```

customer_full_name은 텍스트 타입이다.

- 키워드 타입 필드에 대한 용어 쿼리는 .keyword로 변경하면 된다.

```
1 GET kibana_sample_data_ecommerce/_search
2 {
3   "_source": ["customer_full_name"],
4   "query": {
5     "term": {
6       "customer_full_name.keyword": "Mary Bailey"
7     }
8   }
9 }
```

```
18 {
19   "_index" : "kibana_sample_data_ecommerce",
20   "_type" : "_doc",
21   "_id" : "oibEFX8BjJzEbmEHZ_xT",
22   "_score" : 7.1974354,
23   "_source" : {
24     "customer_full_name" : "Mary Bailey"
25   }
26 },
27 {
28   "_index" : "kibana_sample_data_ecommerce",
29   "_type" : "_doc",
30   "_id" : "PSbEFX8BjJzEbmEHZ_1V",
31   "_score" : 7.1974354,
32   "_source" : {
33     "customer_full_name" : "Mary Bailey"
34   }
35 },
36 {
37   "_index" : "kibana_sample_data_ecommerce",
38   "_type" : "_doc",
39   "_id" : "_CfEFX8BjJzEbmEHdgvA",
40   "_score" : 7.1974354,
41   "_source" : {
42     "customer_full_name" : "Mary Bailey"
43   }
44 }
45 }
```

- 용어들 쿼리

```

1 GET kibana_sample_data_ecommerce/_search
2 {
3   "_source": ["day_of_week"],
4   "query": {
5     "terms": {
6       "day_of_week": ["Monday", "Sunday"]
7     }
8   }
9 }

```

day_of_week -> 범주형 필드

```

81 {
82   "_index": "kibana_sample_data_ecommerce",
83   "_type": "_doc",
84   "_id": "qCbEFX8BjJzEbmEHZ_xT",
85   "_score": 1.0,
86   "_source": {
87     "day_of_week": "Monday"
88   }
89 },
90 {
91   "_index": "kibana_sample_data_ecommerce",
92   "_type": "_doc",
93   "_id": "qSbEFX8BjJzEbmEHZ_xT",
94   "_score": 1.0,
95   "_source": {
96     "day_of_week": "Sunday"
97   }
98 },
99 {
100   "_index": "kibana_sample_data_ecommerce",
101   "_type": "_doc",
102   "_id": "qibEFX8BjJzEbmEHZ_xT",
103   "_score": 1.0,
104   "_source": {
105     "day_of_week": "Monday"
106   }
107 }

```

- 범위 쿼리
 - 특정 날짜, 숫자 범위 검색용
 - 문자형, 키워드 타입의 데이터에서 사용불가
 - 범위 지정 파라미터
 - gte
 - gt
 - lte
 - lt
 - 날짜/시간 데이터 타입 표현식
 - now : 현재시각
 - now+1d: 현재시각 + 1일
 - now+1h+30m+10s: 현재시각 + 1시, 30분, 10초
 - 2021-01-21||+1M : 2021-01-21 +1달
 - 날짜/시간 단위 표기법
 - y (연), d(일), s(초), M(월), H/h(시), w(주), m(분)
 - 범위 데이터 타입 (date_range)

- 날짜/ 시간타임(date)과 다르다.
- `relation` 파라미터로 시작, 끝 범위를 지정한다.
- `relation`에 들어갈 수 있는 값들
 - `intersects(default)`: 쿼리 범위 값이 도큐먼트 범위 데이터를 일부라도 포함하기만 하면 된다.
 - `contains`: 도큐먼트의 범위 데이터가 쿼리 범위 값을 모두 포함해야 한다.
 - `within`: 도큐먼트의 범위 데이터가 쿼리 범위 값 내에 전부 속해야 한다.
- 논리 쿼리
 - 복합 쿼리
 - 논리 쿼리 타입
 - `must`:

쿼리를 실행하여 참인 도큐먼트를 찾는다.

복수 쿼리를 실행하려면 AND 연산을 한다.
 - `must_not`:

쿼리를 실행하여 거짓인 도큐먼트를 찾는다.

다른 타입과 같이 사용할 경우 도큐먼트에서 제외 한다.
 - `should`:

단독으로 사용 시 쿼리를 실행하여 참인 도큐먼트를 찾는다

복수의 쿼리를 실행하면 OR 연산을 한다

다른 타입과 같이 사용할 경우 스코어에만 활용한다.
 - `filter`:

쿼리를 실행하여 예/아니오 형식의 필터 컨텍스트를 수행한다.
 - 패턴 검색
 - 용어 수준 쿼리
 - 1. 와일드카드 쿼리

- 검색어 앞에 *(모든 문자 매칭) 또는 ?(한문자 매칭)을 쓰면 속도가 매우 느려진다.

2. 정규식 쿼리

- +,?,* 같은 기호는 맨 앞에 올 수 없다.
- () 기호는 문자를 그룹핑해 반복되는 문자를 매칭한다
- []기호는 문자를 클래스화해 특정 범위의 문자들을 매칭한다.