ADVANCING UNIVERSITY, SERVING UNIVERSITY

Elasticsearch

신효정

Fast campus

창원문성대학
CHANGWON MOONSUNG UNIVERSITY

Dashboard 만들기 workflow

Fast campus

----------> 작업 시작

ELK 설치

EK 설정 및 실행

E index 구조 잡기

----------> ELK 설치 완료

L 설정 및 실행

----------> index 생성(데이터 수집)

Dashboard 만들기

1) **index 확인(discover)**
2) **dashboard의 목적 정의**
3) **metric 정의**
4) **각 metric을 위한 agg./Visualize type 선정**
5) **metric별 object 생성 후 dashboard 구축**

----------> dashboard 구축 완료

Feed back

1) **수집 데이터 수정**
2) **'Dashboard 만들기' 수정**

데이터 시각화. 전체 Elastic Stack 탐색.

Kibana

더 보기 | 다운로드

Kibana는 데이터를 시각화하고 Elastic Stack의 모든 기능을 구성 및 관리할 수 있는 확장 가능한 UI 도구입니다.

데이터 검색, 분석 및 저장

Elasticsearch

더 보기 | 다운로드

Elasticsearch는 분산 시스템으로서 수평적인 확장성, 최고의 안정성 및 간편한 관리를 위해 설계된 JSON문서 기반의 검색 및 분석 엔진입니다.

자유로운 형식 및 소스로부터의 데이터 수집

Beats

Beats는 단말 장치의 데이터를 Logstash 및 Elasticsearch로 전송하는 경량 수집기용 플랫폼입니다.

더 보기 | 다운로드

Logstash

Logstash는 확장 가능한 플러그인 에코시스템으로 구성된 동적 데이터 수집 파이프라인이며, Elasticsearch와 강력한 시너지를 냅니다.

더 보기 | 다운로드

https://www.elastic.co/kr/products/

Search Internal

Elasticsearch 왜 검색이 빠를까?

'밥' 검색

ID	Content
1	나는 오늘도 밥을 먹는다
2	나는 밥으로 김치와 삼겹살을 먹었다
3	저기에 있는 사람은 밥을 참 빨리 먹더라
4	나무 위에 있는 남자가 급하게 밥을 먹었다
5	저기 있는 빨간색 사과가 먹고 싶다.
6	붉은색은 식욕을 일으켜 밥을 많이 먹게 한다.

Elasticsearch 왜 검색이 빠를까?

Inverted Index

Token	Document Frequency	Postings(Document IDs)
나는	9	12, 24, 32, 55, …
밥을	4	2, 5, 7, 4
먹었다	2	3, 12
빨간색	8	43, 78, 23, 55, …
사과	4	12, 2, 6, 7
…	…	…

Tokenizer : 텍스트를 일정한 토큰으로 나눔

https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-tokenizers.html

Elasticsearch 왜 검색이 빠를까?

ID	Content
1	나는 오늘도 밥을 먹는다
2	나는 밥으로 김치와 삼겹살을 먹었다
3	저기에 있는 사람은 밥을 참 빨리 먹더라
4	나무 위에 있는 남자가 급하게 밥을 먹었다
5	저기 있는 빨간색 사과가 먹고 싶다.
6	붉은색은 식욕을 일으켜 밥을 많이 먹게 한다.

Indexing

Token	IDs
나는	1, 2
오늘도	1
밥을	1, 3, 6
먹는다	1
밥으로	2
김치와	2
삼겹살을	2
먹었다	2, 4
빨리	3
급하게	4
…	…

Inverted Index

Elasticsearch 왜 검색이 빠를까?

Token	IDs
나는	1, 2
오늘도	1
밥을	1, 3, 6
먹는다	1
밥으로	2
김치와	2
삼겹살을	2
먹었다	2, 4
빨리	3
급하게	4
…	…

Analyzer : 검색 성능 향상을 위한 전처리

https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-analyzers.html

Elasticsearch 왜 검색이 빠를까?

Token	IDs
나는	1, 2
오늘도	1
밥을	1, 3, 6
먹는다	1
밥으로	2
김치와	2
삼겹살을	2
먹었다	2, 4
빨리	3
급하게	4
…	…

Stopwords
의미 없는 token
(예: the, a, an, …)

Elasticsearch 왜 검색이 빠를까?

Token	IDs
오늘도	1
밥을	1, 3, 6
먹는다	1
밥으로	2
김치와	2
삼겹살을	2
먹었다	2, 4
빨리	3
급하게	4
…	…

Stopwords
의미 없는 token
(예: the, a, an, …)

Elasticsearch 왜 검색이 빠를까?

Token	IDs
오늘도	1
밥을	1, 3, 6
먹는다	1
밥으로	2
김치와	2
삼겹살을	2
먹었다	2, 4
빨리	3
급하게	4
…	…

Lowercasing
대문자를 소문자로 변경
(예: Cars → cars, Fast → fast)

Elasticsearch 왜 검색이 빠를까?

Token	IDs
오늘도	1
밥을	1, 3, 6
먹는다	1
밥으로	2
김치와	2
삼겹살을	2
먹었다	2, 4
빨리	3
급하게	4
…	…

Grammar(Analyzer, stemming)
형태소분석을 통한 조사제거, 시제변경
(예: 밥을 → 밥, 밥으로 → 밥)

Elasticsearch 왜 검색이 빠를까?

Token	IDs
오늘	1
밥	1, 2, 3, 4, 6
먹다	1, 2, 3, 4, 5, 6
나무	4
김치	2
삼겹살	2
사과	5
빨리	3
급히	4
…	…

Grammar(Analyzer)
형태소분석을 통한 조사제거, 시제변경
(예: 밥을 → 밥, 밥으로 → 밥)

Elasticsearch 왜 검색이 빠를까?

Token	IDs
오늘	1
밥	1, 2, 3, 4, 6
먹다	1, 2, 3, 4, 5, 6
나무	4
김치	2
삼겹살	2
사과	5
빨리	3
급히	4
…	…

Synonyms
문자는 다르지만 의미가 같은 token
(예: 빨리 == 급하게)

Elasticsearch 왜 검색이 빠를까?

Token	IDs
오늘	1
밥	1, 2, 3, 4, 6
먹다	1, 2, 3, 4, 5, 6
나무	4
김치	2
삼겹살	2
사과	5
빨리	3, 4
급히	4, 3
…	…

Synonyms
문자는 다르지만 의미가 같은 token
(예: 빨리 == 급하게)

Elasticsearch 왜 검색이 빠를까?

ID	Content
1	나는 오늘도 밥을 먹는다
2	나는 밥으로 김치와 삼겹살을 먹었다
3	저기에 있는 사람은 밥을 참 빨리 먹더라
4	나무 위에 있는 남자가 급하게 밥을 먹었다
5	저기 있는 빨간색 사과가 먹고 싶다.
6	붉은색은 식욕을 일으켜 밥을 많이 먹게 한다.

Indexing

stopword 삭제
lowercase 처리
analyzer(stemming) 처리
synonym 처리

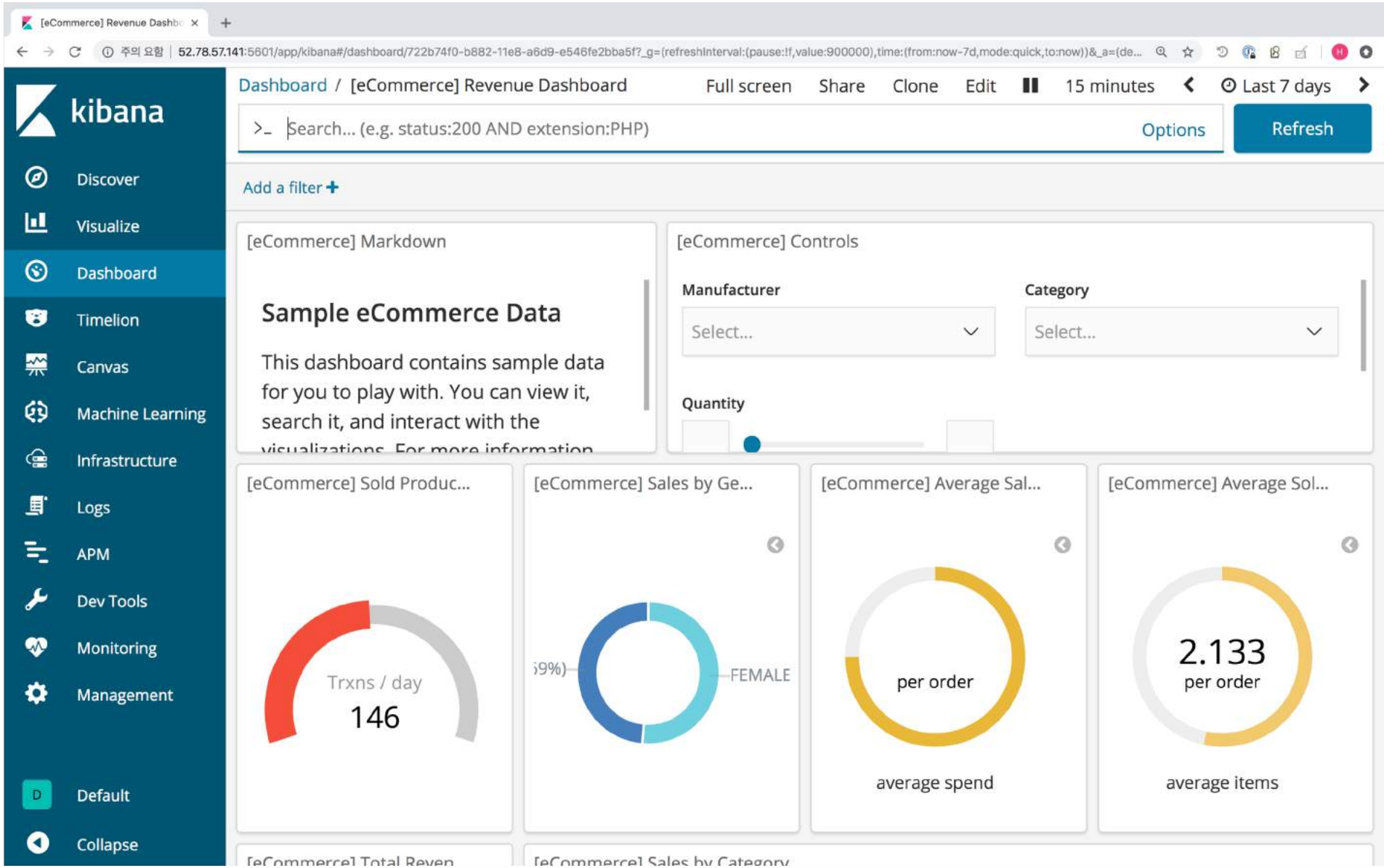
Token	IDs
오늘	1
밥	1, 2, 3, 4, 6
먹다	1, 2, 3, 4, 5, 6
나무	4
김치	2
삼겹살	2
사과	5
빨리	3, 4
급히	4, 3
…	…

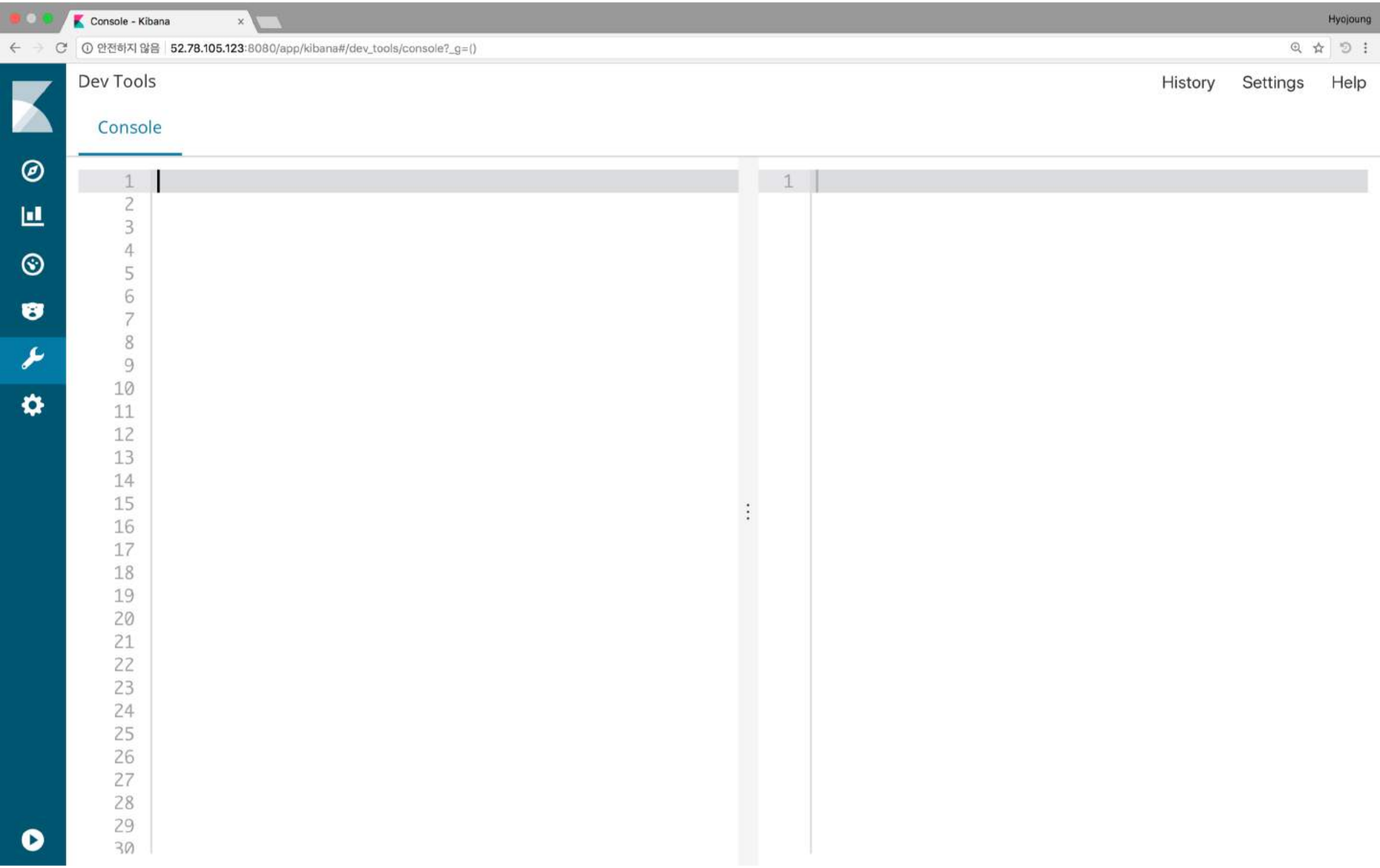
Inverted Index

Document API

URL → 52.78.134.20:5601

Fast campus




Console - Kibana
Hyojoung
안전하지 않음 52.78.105.123:8080/app/kibana#/dev_tools/console?_g=()
Dev Tools
History
Settings
Help
Console

Console - Kibana

Hyojoung

안전하지 않음 52.78.105.123:8080/app/kibana#/dev_tools/console?_g=()

Dev Tools

History Settings Help

Console

click to send request

```
GET _cat/indices
```

Console UI

```
yellow open data                    c20IwkGzQNy_8IjFE23UUQ 5 1
   3 0 12.1kb 12.1kb
yellow open metricbeat-6.2.4-2018.07.17 7f-zWd0wSLSYSF5WdXT3rg 1 1
36790 0  5.7mb  5.7mb
yellow open exam_shape              hYVJnrsLTvW_q8OR6GXmEg 5 1
27438 0  3.7mb  3.7mb
yellow open security-logs           jNqEMZF7Sn6SimVRW9MDwQ 5 1
30874 0  7.3mb  7.3mb
yellow open tophit                  CRJ-GaKqQqKvGQkqzXMT_w 5 1
   5 0 12.4kb 12.4kb
green  open .kibana                 SaVUKoIDQfiP_TAHWBcOdA 1 0
   4 1 20.8kb 20.8kb
yellow open ti-logs                 sEe-mJuMSNOlb4FsjI2qtA 5 1
34106 0 12.6mb 12.6mb
```

Output Pane

Document APIs

- Index API
- Get API
- Delete API
- Delete By Query API
- Update API
- Update By Query API
- Multi Get API
- Bulk API
- Reindex API
- Term Vectors
- Multi termvectors API
- ?refresh

```
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch"
}
```

{request type} {url}
{body}

{request type} : GET, PUT, POST, DELETE, HEAD
{url} : function 마다 정의 되어 있음
{body} : json

Index API : elasticsearch에 document 추가

Fast campus

Console - Kibana

localhost:5601/app/kibana#/dev_tools/console?load_from=https:%2F%2Fwww.elastic.co%2Fguide%2Fen%2Felasticsearch%2Freference%2Fcurrent%2Fsnippets%2Fdocs-index_%2F1.json&_g=()

Dev Tools

History Settings Help

Console

```
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch"
}
```

```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 0,
  "_primary_term": 1
}
```

Console의 API를 Request 하는 방법

1) 요청하고자 하는 API에 커서 이동

2-A) 녹색 플레이 버튼 클릭

2-B) 컨트롤 + 엔터(커맨드 + 엔터)

Index API : elasticsearch에 document 추가

```
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch"
}
```

```
PUT index_name/type_name/document_id
{
    "field_name" : "value"
}
```

Index API : elasticsearch에 document 추가

Fast campus

PUT request 1회 수행


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 0,
  "_primary_term": 1
}
```


PUT request 2회 수행


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 2,
  "result": "updated",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 1,
  "_primary_term": 1
}
```

Index API : elasticsearch에 document 추가

```
POST shj_twitter/tweet/
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch"
}
```

```
POST index_name/type_name
{
    "field_name" : "value"
}
```

Index API : elasticsearch에 document 추가

Fast campus


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "KQ9-9GQBjP0djsH1rvO-",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 0,
  "_primary_term": 1
}
```


랜덤하게 id를 부여하여 Document 추가

Index API : elasticsearch에 document 추가

Fast campus

지정된 document_id에 body insert or update

```
PUT index_name/type_name/document_id
{
    "field_name" : "value"
}
```

Random하게 할당된 document_id에 body insert

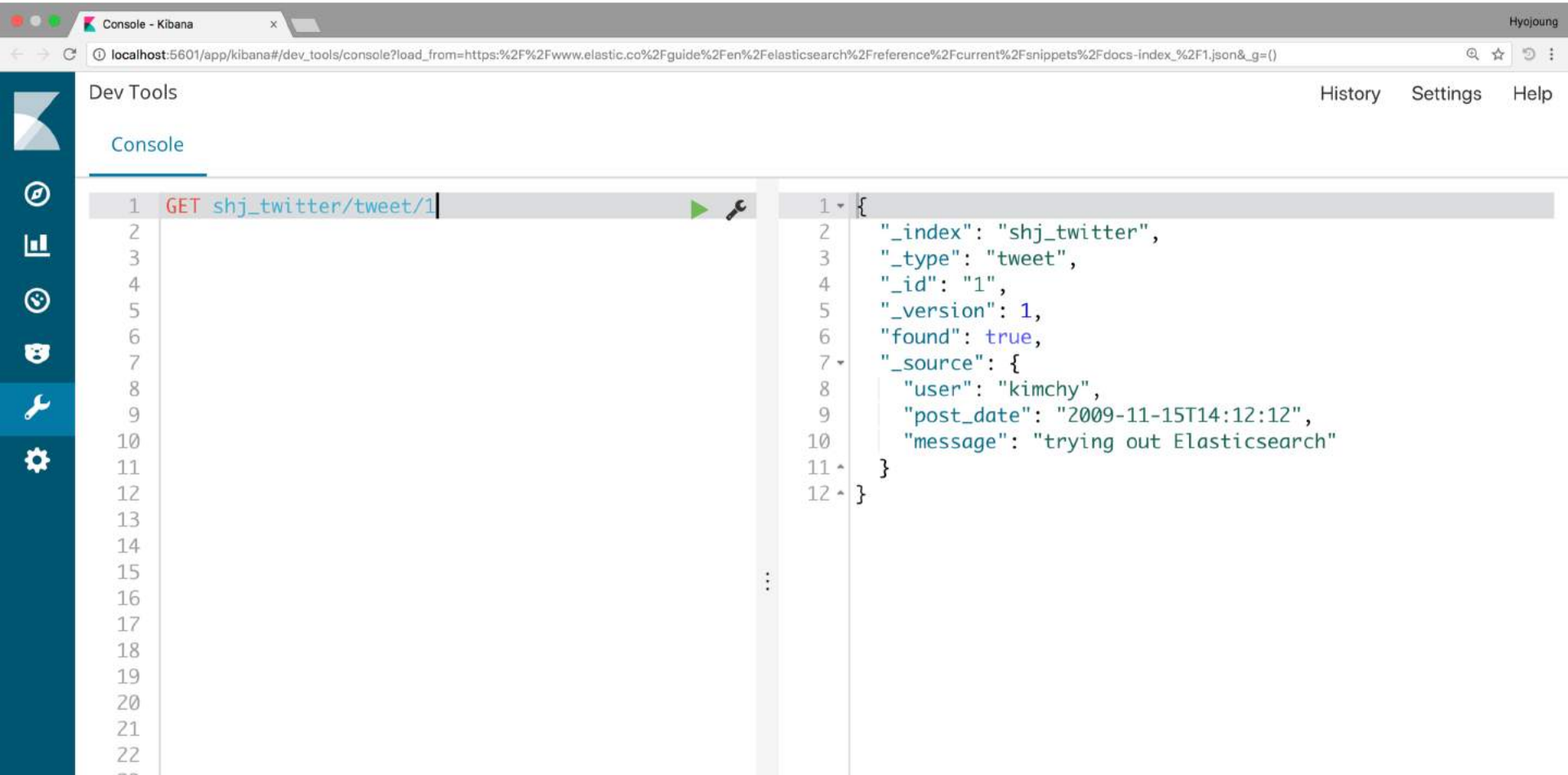
```
POST index_name/type_name
{
    "field_name" : "value"
}
```

GET API : elasticsearch의 내용 확인

Fast campus

GET shj_twitter/tweet/1

GET index_name/type_name/document_id



GET API : elasticsearch의 내용 확인

Fast campus

GET shj_twitter/tweet/1


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 1,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch"
  }
}
```


GET shj_twitter/tweet/2


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "2",
  "found": false
}
```

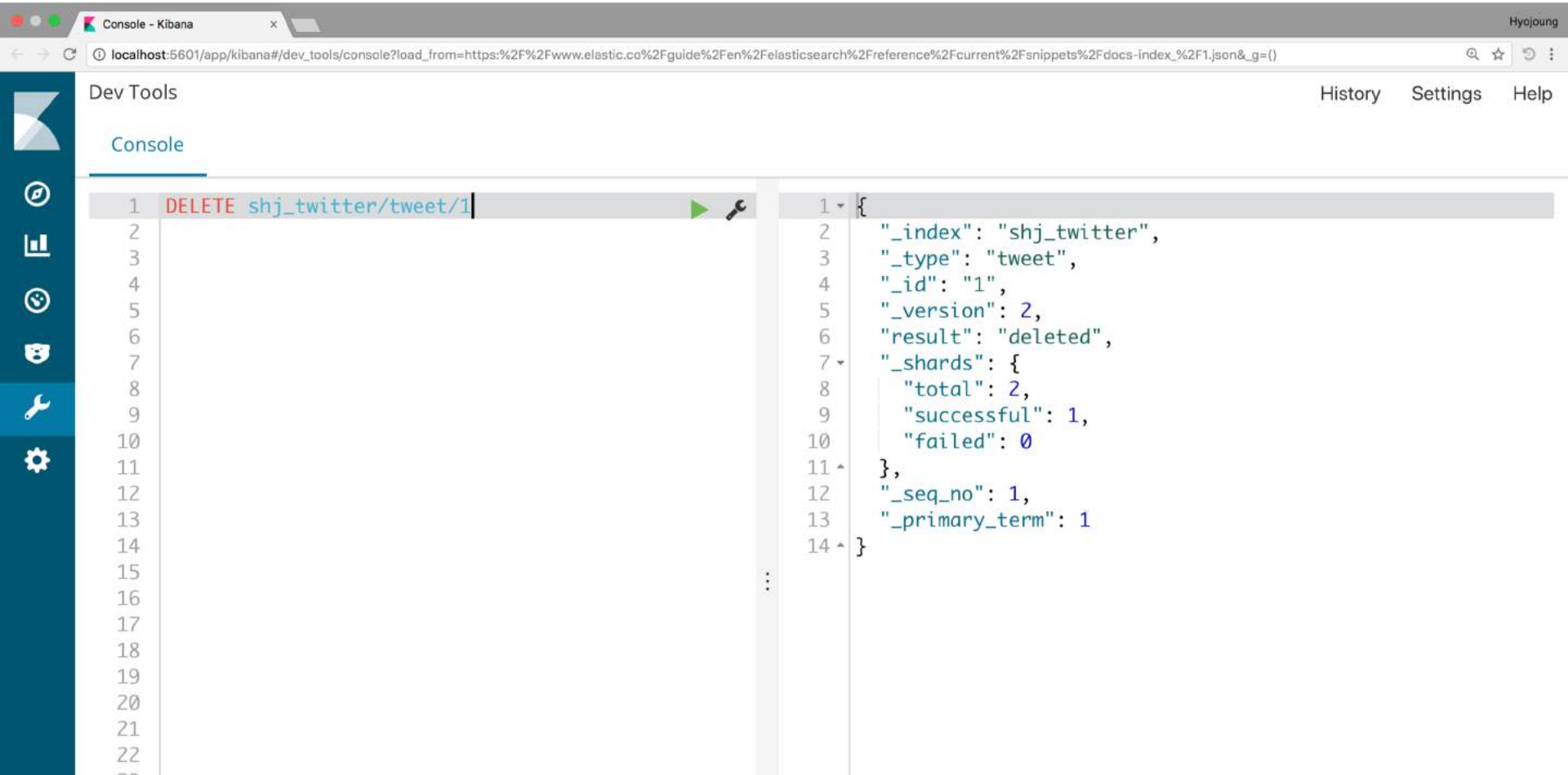

Meta-Fields

`_index`	The index to which the document belongs.
`_uid`	A composite field consisting of the `_type` and the `_id`.
`_type`	The document's mapping type.
`_id`	The document's ID.

DELETE API : elasticsearch의 document 삭제

DELETE shj_twitter/tweet/1

DELETE index_name/type_name/document_id

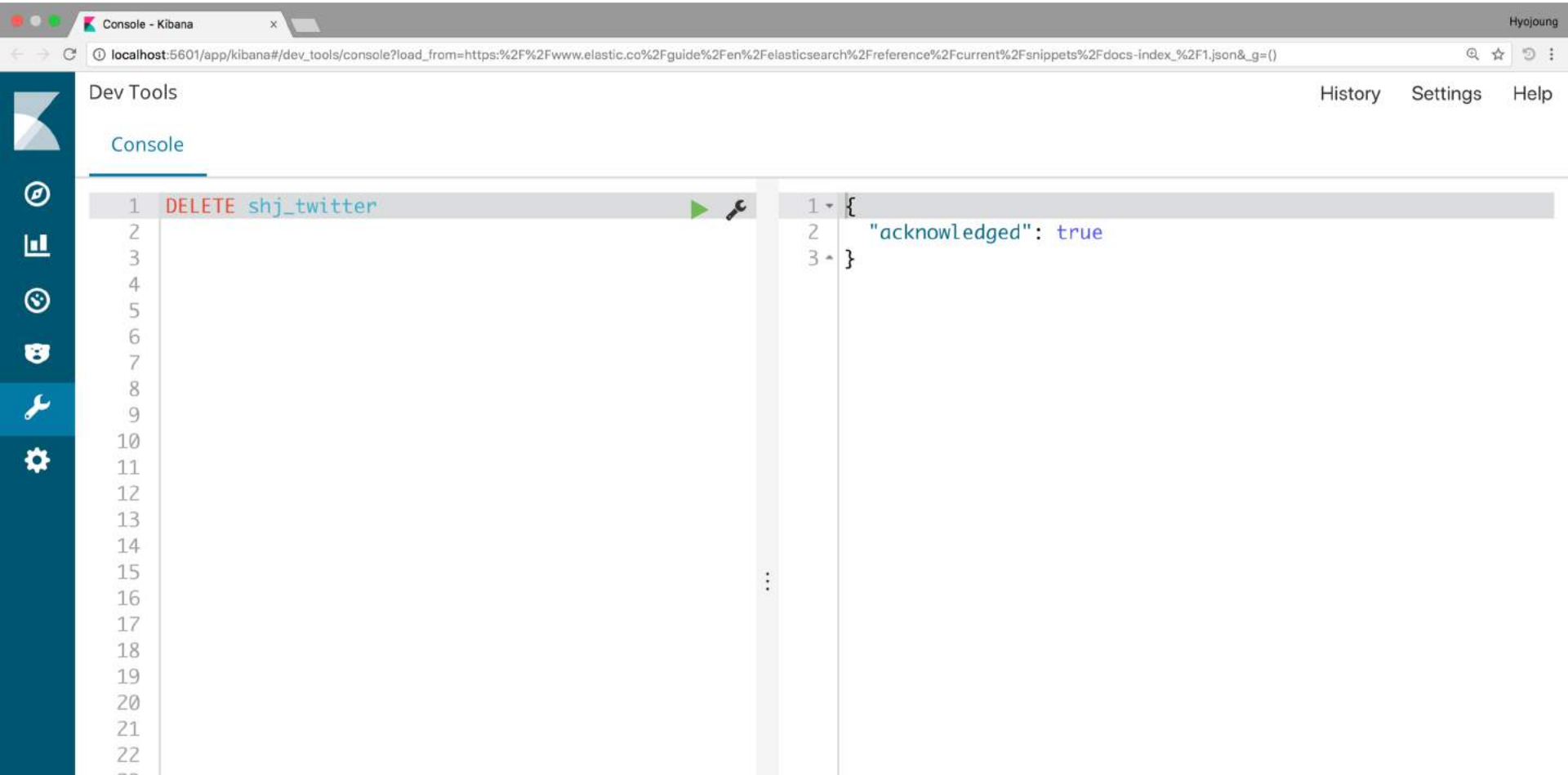


DELETE API : elasticsearch의 index 삭제

Fast campus

DELETE shj_twitter

DELETE **index_name**



UPDATE API : elasticsearch의 document 수정

```
#update api exam1
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch"
}

GET shj_twitter/tweet/1

PUT shj_twitter/tweet/1
{
    "retweet" : 12
}

GET shj_twitter/tweet/1
```

```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 2,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch"
  }
}
```

```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 3,
  "found": true,
  "_source": {
    "retweet": 12
  }
}
```

PUT은 document를 덮어쓰기 함

UPDATE API : elasticsearch의 document 수정

```
#update api exam8
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch",
    "tags" : ["red"]
}

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
    "doc" : {
        "category" : "elastic"
    }
}

GET shj_twitter/tweet/1
```

```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 2,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "tags": [
      "red"
    ],
    "category": "elastic"
  }
}
```

UPDATE API : elasticsearch의 document 수정

```
#update api exam2
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch"
}

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
    "script" : {
        "source": "ctx._source.retweet = 12"
    }
}

GET shj_twitter/tweet/1
```

```
{
  ...
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch"
  }
}
```

```
{
  ...
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "retweet": 12
  }
}
```

_update를 활용하여 document에 field를 추가

UPDATE API : elasticsearch의 document 수정

```
#update api exam3
GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": "ctx._source.retweet += params.count",
      "lang": "painless",
      "params" : {
         "count" : 4
      }
   }
}

GET shj_twitter/tweet/1
```

```
{
  ...
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "retweet": 12
  }
}
```

```
{
  ...
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "retweet": 16
  }
}
```

UPDATE API : elasticsearch의 document 수정

```
#update api exam4
PUT shj_twitter/tweet/1
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch",
   "tags" : ["red"]
}

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": "ctx._source.tags.add(params.tag)",
      "lang": "painless",
      "params" : {
         "tag" : "blue"
      }
   }
}

GET shj_twitter/tweet/1
```

UPDATE API : elasticsearch의 document 수정


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 12,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "tags": [
      "red"
    ]
  }
}
```

```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 13,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "tags": [
      "red",
      "blue"
    ]
  }
}
```

UPDATE API : elasticsearch의 document 수정

```
#update api exam6
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch",
    "tags" : ["red"]
}

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
    "script" : "ctx._source.remove('tags')"
}

GET shj_twitter/tweet/1
```

UPDATE API : elasticsearch의 document 수정


```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 19,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch",
    "tags": [
      "red"
    ]
  }
}
```



```
{
  "_index": "shj_twitter",
  "_type": "tweet",
  "_id": "1",
  "_version": 20,
  "found": true,
  "_source": {
    "user": "kimchy",
    "post_date": "2009-11-15T14:12:12",
    "message": "trying out Elasticsearch"
  }
}
```

UPDATE API : elasticsearch의 document 수정

```
#update api exam7
PUT shj_twitter/tweet/1
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch",
   "tags" : ["red"]
}

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": "if (ctx._source.tags.contains(params.tag)) { ctx.op = 'delete' } else
{ ctx.op = 'none' }",
      "lang": "painless",
      "params" : {
         "tag" : "green"
      }
   }
}

GET shj_twitter/tweet/1
```

UPDATE API : elasticsearch의 document 수정

```
#update api exam7
PUT shj_twitter/tweet/1
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch",
   "tags" : ["red"]
}

POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": """
            if (ctx._source.tags.contains(params.tag)) {
               ctx.op = 'delete'
            } else {
               ctx.op = 'none'
            }
                  """,
      "lang": "painless",
      "params" : {
         "tag" : "green"
      }
   }
}
```

Params의 tag 값이 id 1번 document에 있으면
Delete 수행, 아니면 아무 것도 하지 않음

UPDATE API : elasticsearch의 document 수정

```
#update api exam7
POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": """
            if (ctx._source.tags.contains(params.tag)) {
               ctx.op = 'delete'
            } else {
               ctx.op = 'none'
            }
                  """,
      "lang": "painless",
      "params" : {
         "tag" : "red"
      }
   }
}

GET shj_twitter/tweet/1
```

Params의 tag 값이 id 1번 document에 있으면
Delete 수행, 아니면 아무 것도 하지 않음

UPDATE API : elasticsearch의 document 수정

```
#update api exam9
DELETE shj_twitter/tweet/1

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": "ctx._source.counter += params.count",
      "lang": "painless",
      "params" : {
         "count" : 4
      }
   },
   "upsert" : {
      "counter" : 1
   }
}

GET shj_twitter/tweet/1

POST shj_twitter/tweet/1/_update
{
   "script" : {
      "source": "ctx._source.counter += params.count",
      "lang": "painless",
      "params" : {
         "count" : 4
      }
   },
   "upsert" : {
      "counter" : 1
   }
}

GET shj_twitter/tweet/1
```

```
{
  …
  "_source": {
    "counter": 1
  }
}
```

```
{
  …
  "_source": {
    "counter": 5
  }
}
```

```
{
  …
  "_source": {
    "counter": 9
  }
}
```

id 1번 document가 없으면 upsert, 수행 있으면 script 수행

termvector : field의 token 확인

```
#termvector
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch",
    "tags" : ["red"]
}

GET shj_twitter/tweet/1/_termvectors?fields=message
```

id 1번 document의 message field의 inverted index 확인
이 토큰들로 document가 검색 됨

```
{
  ...
  "term_vectors": {
    "message": {
      "field_statistics": {
        ...
      },
      "terms": {
        "elasticsearch": {
          "term_freq": 1,
          "tokens": [
            {
              "position": 2,
              "start_offset": 11,
              "end_offset": 24
            }
          ]
        },
        "out": {
          ...
        },
        "trying": {
          ...
        }
      }
    }
  }
}
```

* Elasticsearch는 default로 lowercase 적용

Data type

Field

```
{
  "이름" : "백두산",
  "부서" : "영업",
  "사번" : "123",
  "취미" : "낚시, 마라톤",
},
{
  "이름" : "한라산",
  "부서" : "인사",
  "사번" : "124",
  "특기" : "테니스"
},
{
  "이름" : "금강산",
  "부서" : "개발",
  "사번" : "125"
  "취미" : "마라톤"
}
```

- Field datatypes
  - Array datatype
  - Binary datatype
  - Range datatypes
  - Boolean datatype
  - Date datatype
  - Geo-point datatype
  - Geo-Shape datatype
  - IP datatype
  - Keyword datatype
  - Nested datatype
  - Numeric datatypes
  - Object datatype
  - Text datatype
  - Token count datatype
  - Percolator type
  - `join` datatype

Core datatypes

Fast campus

string
text, keyword

Numeric datatypes
long, integer, short, byte, double, float, half_float, scaled_float

Date datatype
date

Boolean datatype
boolean

Binary datatype
binary

Range datatypes
integer_range, float_range, long_range, double_range, date_range

Core datatypes

Fast campus

string
text, keyword

```
PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "message" : "trying out Elasticsearch"
}
```

keyword

Token	Document IDs
kimchy	1
trying out Elasticsearch	1

text

Token	Document IDs
kimchy	1
trying	1
out	1
elasticsearch	1

Range datatypes
integer_range, float_range, long_range, double_range, date_range

```
PUT range_index/doc/1
{
  "expected_attendees" : {
    "gte" : 10,
    "lte" : 20
  },
  "time_frame" : {
    "gte" : "2015-10-31 12:00:00",
    "lte" : "2015-11-01"
  }
}
```

Complex datatypes

Fast campus

Array datatype

```
PUT shj_twitter/tweet/1
{
  "tags": ["red", "blue"]
}
```

Object datatype

```
PUT shj_twitter/tweet/1
{
  "object": { "key1" : "val1", "key2" : "val2" }
}
```

Nested datatype

```
PUT shj_twitter/tweet/1
{
  "user" : "kimchy",
  "follower" : [
    {
      "first" : "John",
      "last" :  "Smith"
    },
    {
      "first" : "Alice",
      "last" :  "White"
    }
  ]
}
```

Geo datatypes

Fast campus

Geo-point datatype

```
PUT shj_twitter/tweet/1
{
  "location": {
    "lat": 41.12,
    "lon": -71.34
  }
}
```

Geo-Shape datatype

```
PUT shj_twitter/tweet/1
{
    "location" : {
        "type" : "polygon",
        "coordinates" : [
            [ [100.0, 0.0], [101.0, 0.0], [101.0, 1.0], [100.0, 1.0], [100.0, 0.0] ]
        ]
    }
}
```

Specialised datatypes

Fast campus

IP datatype

```
PUT shj_twitter/tweet/1
{
  "ip_addr": "192.168.1.1"
}
```

join datatype

```
PUT shj_join/join/1
{
  "text": "This is a question",
  "my_join_field": {
    "name": "question"
  }
}

PUT shj_join/join/2
{
  "text": "This is a another question",
  "my_join_field": {
    "name": "question"
  }
}
```

```
PUT shj_join/join/3
{
  "text": "This is an answer",
  "my_join_field": {
    "name": "answer",
    "parent": "1"
  }
}

PUT shj_join/join/4
{
  "text": "This is another answer",
  "my_join_field": {
    "name": "answer",
    "parent": "1"
  }
}
```

Mapping

```
#mapping
DELETE shj_twitter
PUT shj_twitter/tweet/1
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch",
   "tags" : ["red"],
   "retweet" : 12
}

GET shj_twitter/_mapping
```

```
"shj_twitter": {
 "mappings": {
  "tweet": {
   "properties": {
    "message": {
     "type": "text",
     "fields": {
      "keyword": {
       "type": "keyword",
       "ignore_above": 256
      }
     }
    },
    "post_date": {
     "type": "date"
    },
    "retweet": {
     "type": "long"
    },
    "tags": {
     "type": "text",
     "fields": {
      "keyword": {
       "type": "keyword",
       "ignore_above": 256
      }
     }
    },
    …
```

```
#mapping exam1
DELETE shj_twitter
PUT shj_twitter
{
  "mappings": {
    "tweet" : {
      "properties": {
        "user" : {
          "type": "keyword"
        },
        "post_date" : {
          "type": "date"
        },
        "message" : {
          "type": "text"
        },
        "tags" : {
          "type": "keyword"
        },
        "retweet" : {
          "type": "integer"
        }
      }
    }
  }
}

PUT shj_twitter/tweet/1
{
    "user" : "kimchy",
    "post_date" : "2009-11-15T14:12:12",
    "message" : "trying out Elasticsearch",
    "tags" : ["red"],
    "retweet" : 12
}

GET shj_twitter/_mapping
```

```
{
  "shj_twitter": {
    "mappings": {
      "tweet": {
        "properties": {
          "message": {
            "type": "text"
          },
          "post_date": {
            "type": "date"
          },
          "retweet": {
            "type": "integer"
          },
          "tags": {
            "type": "keyword"
          },
          "user": {
            "type": "keyword"
          }
        }
      }
    }
  }
}
```

```
PUT index_name
{
  "mappings": {
    "type_name" : {
      "properties": {
        "field_name1" : {
          "type": "field_type"
        },
        "field_name2" : {
          "type": "field_type"
        },
        "field_name3" : {
          "type": "field_type"
        },
        "field_name4" : {
          "type": "field_type"
        },
        …
      }
    }
  }
}
```

```
PUT index_name
PUT index_name/_mapping/type_name
{
  "properties": {
    "field_name": {
      "type": "field_type"
    }
  }
}
```

```
#mapping exam2
PUT shj_twitter
PUT shj_twitter/_mapping/tweet
{
  "properties": {
    "email": {
      "type": "keyword"
    }
  }
}
```

* 기존에 생성되어 있는 mapping에 field 추가하기

Dynamic mapping

```
#mapping exam3
PUT shj_dynamic_mapping/doc/1
{
  "text" : "this is a text field",
  "keyword" : "keyword",
  "int" : 10,
  "float" : 10.25
}

GET shj_dynamic_mapping/doc/1

GET shj_dynamic_mapping/_mapping
```

```
"properties": {
  "float": {
    "type": "float"
  },
  "int": {
    "type": "long"
  },
  "keyword": {
    "type": "text",
    "fields": {
      "keyword": {
        "type": "keyword",
        "ignore_above": 256
      }
    }
  },
  "text": {
    "type": "text",
    "fields": {
      "keyword": {
        "type": "keyword",
        "ignore_above": 256
      }
    }
  }
}
```

Search API

Search APIs

Fast campus

- Search
- URI Search
- + Request Body Search
- Search Template
- Multi Search Template
- Search Shards API
- + Suggesters
- Multi Search API
- Count API
- Validate API
- Explain API
- + Profile API
- Field Capabilities API
- Ranking Evaluation API

#search api : search all

```
GET ecommerce/_doc/_search
GET ecommerce/_search
GET ecommerce/_search
{
  "query": {
    "match_all": {}
  }
}
```


```
{
  "took" : 0,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 4675,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "ecommerce",
        "_type" : "_doc",
        "_id" : "KjSWxmcB_tq4r_kT7SG7",
        "_score" : 1.0,
        "_source" : {
          "category" : [
            "Women's Clothing"
          ],
          "currency" : "EUR",
          "customer_first_name" : "Mary",
          "customer_full_name" : "Mary Bailey",
          "customer_gender" : "FEMALE",
          "customer_id" : 20,
          "customer_last_name" : "Bailey",
          "customer_phone" : "",
…
```

#search api : search all from all index
GET /_search

#search api : search all from ecommerce, flights index
GET ecommerce,flights /_search

#search api : search all from index name starts with ec
GET ec*/_search

Request Body Search

Fast campus

검색의 범위 지정

```
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  }
}
```

Request Body Search

```
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "_source": [
    "customer_full_name",
    "category"
  ],
  "from": 0,
  "size": 20,
  "sort": [
    {
      "order_date": {
        "order": "desc"
      }
    }
  ],
  "script_fields": {
    "shj_field": {
      "script": {
        "lang": "painless",
        "source": "doc['customer_full_name.keyword'].value +' : ' + doc['customer_id'].value"
      }
    }
  }
}
```

Request Body Search

```
#search exam1
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  }
}
```

```
{
  …
  "hits": {
    "total": 4675,
    "max_score": 1,
    "hits": [
      {
        …
        "_source": {
          …
        }
      },
      {
        …
      }
}
```

총 10개 document

Request Body Search

```
#search exam2
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "size": 20
}
```

```
{
  …
  "hits": {
    "total": 4675,
    "max_score": 1,
    "hits": [
      {
        …
        "_source": {
          …
        }
      },
      {
        …
      }
}
```

총 20개 document

Request Body Search

```
#search exam3
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "size": 20,
  "from": 0
}

GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "size": 20,
  "from": 1
}

GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "size": 20,
  "from": 2
}
```

Document	0
Document	1
Document	2
Document	3

…

검색 된 document를 기준으로 from index 부터 size 개수만큼의 document만 return

Request Body Search

```
GET index_name/type_name/_search
{
  "query": {
    "match_all": {}
  },
  "sort": [
    {
      " field_name": {
        "order": "acs/desc"
      }
    }
  ]
}
```

```
#search exam4
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "sort": [
    {
      "order_date": {
        "order": "desc"
      }
    }
  ]
}
```

order_date field를 기준으로 내림차순 정렬

Request Body Search

```
GET index_name/type_name/_search
{
  "query": {
    "match_all": {}
  },
  "_source": ["field_name1", "field_name2", …]
}
```

```
#search exam5
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
  "_source": ["customer_full_name",
    "category"]
}
```

Source의 특정 field만 보여줌

```
{
      "_index": "exam_shape",
      "_type": "exam",
      "_id": "3lOFqGQB8H68NvUu2GR2",
      "_score": 1,
      "_source": {
        "customer_full_name" : "Youssef
Jensen",
        "category" : [
          "Men's Shoes",
          "Men's Clothing"
        ]       }
    }
```

Request Body Search

```
GET index_name/type_name/_search
{
  "query": {
    "match_all": {}
  },
  "script_fields": {
    "script_field_name": {
      "script": {
        "lang": "painless",
        "source": "user script"
      }
    }
  }
}
```

Request Body Search

```
#search exam6
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  },
 "script_fields": {
    "shj_field": {
      "script": {
        "lang": "painless",
        "source": "doc['customer_full_name.keyword'].value +' : ' +
doc['customer_id'].value"
      }
    }
  }
}
```

검색된 document에 scripted field 추가

```
{
        "_index": "ecommerce",
        "_type": "_doc",
        "_id": "3IOFqGQB8H68NvUu2GR2",
        "_score": 1,
        "fields" : {
          "shj_field" : [
            "Youssef Jensen : 31"
          ]
        }
  }
```

Query DSL

```
GET ecommerce/_doc/_search
{
  "query": {
    "match_all": {}
  }
}
```

- Query DSL
  - Query and filter context
  - Match All Query
  - + Full text queries
  - + Term level queries
  - + Compound queries
  - + Joining queries
  - + Geo queries
  - + Specialized queries
  - + Span queries
  - Minimum Should Match
  - Multi Term Query Rewrite

Query DSL

```
GET index_name/type_name/_search
{
  "query": {
    "term": {
      "field_name": {
        "value": "query_value"
      }
    }
  }
}
```

```
#Query DSL - term
GET ecommerce/_doc/_search
{
  "query": {
    "term": {
      "day_of_week": {
        "value": "Sunday"
      }
    }
  }
}
```

```
GET ecommerce/_doc/_search
{
  "query": {
    "term": {
      "customer_gender": {
        "value": "FEMALE"
      }
    }
  }
}
```

Keyword type field에서 exact match 수행

Query DSL

```
GET index_name/type_name/_search
{
  "query": {
    "terms": {
      "field_name": {
        "value": ["query_value1", "query_value2"]
      }
    }
  }
}
```

```
#Query DSL - terms
GET ecommerce/_doc/_search
{
  "query": {
    "terms": {
      "day_of_week": ["Sunday", "Monday"]
    }
  }
}
```

Keyword type field에서 여러 term에 대해 exact match 수행

Query DSL

```
GET index_name/type_name/_search
{
   "query": {
      "range" : {
         "field_name" : {
            "gte" : "condition_value",
            "lt" :  "condition_value"
         }
      }
   }
}
```

```
#Query DSL - range query
GET ecommerce/_doc/_search
{
  "query": {
   "range": {
    "day_of_week_i": {
     "gte": "2",
     "lt": "4"
    }
   }
  }
}
```

g(greater), l(less), t(then), e(equal)

Query DSL

```
GET index_name/type_name/_search
{
  "query": {
    "exists": {
      "field": "field_name"
    }
  }
}
```

```
#Query DSL - exists query
GET ecommerce/_doc/_search
{
  "query": {
    "exists": {
      "field": "category"
    }
  }
}
```

Field가 있는 document 검색

Query DSL

```
#Query DSL - pattern
GET ecommerce/_doc/_search
{
  "query": {
    "prefix": {
      "geoip.city_name": "c"
    }
  }
}
```

```
GET ecommerce/_doc/_search
{
  "query": {
    "wildcard": {
      "geoip.city_name": "c*"
    }
  }
}
```

```
GET ecommerce/_doc/_search
{
  "query": {
    "fuzzy": {
      "geoip.city_name.keyword": {
        "value": "Caira",
        "fuzziness": 1
      }
    }
  }
}
```

```
GET ecommerce/_search
{
  "query": {
    "ids": {
      "type": "_doc",
      "values": [
        "TTSWxmcB_tq4r_kT7SG7",
        "fDSWxmcB_tq4r_kT7SG7"
      ]
    }
  }
}
```

Query DSL

```
GET index_name/type_name/_search
{
  "query": {
    "match": {
      "field_name" : "field_value"
    }
  }
}
```

```
#Query DSL - match
GET ecommerce/_doc/_search
{
  "query": {
    "term": {
      "customer_full_name": "Abd Hernandez"
    }
  }
}

GET ecommerce/_doc/_search
{
  "query": {
    "match": {
      "customer_full_name": "Abd Hernandez"
    }
  }
}
```

Match는 field type이 text인 field에서 검색 수행

Query DSL

```
#Query DSL - match with operator
GET ecommerce/_doc/_search
{
  "query": {
    "match": {
      "customer_full_name": {
        "query": "Abd Hernandez"
      }
    }
  }
}

GET ecommerce/_doc/_search
{
  "query": {
    "match": {
      "customer_full_name": {
        "query": "Abd Hernandez",
        "operator": "and"
      }
    }
  }
}

GET ecommerce/_doc/_search
{
  "query": {
    "match": {
      "customer_full_name": {
        "query": "Abd Hernandez",
        "operator": "or"
      }
    }
  }
}
```

Default operator는 or

customer_full_name:Abd or customer_full_name:Hernandez

Query DSL

```
GET index/type/_search
{
  "query": {
    "match_phrase": {
      "field_name": "query”
    }
  }
}
```

```
#Query DSL - match_phrase
GET ecommerce/_doc/_search
{
  "query": {
    "match_phrase": {
      "customer_full_name": "Abd Hernandez"
    }
  }
}

GET ecommerce/_doc/_search
{
  "query": {
    "match_phrase": {
      "customer_full_name": "Hernandez Abd"
    }
  }
}
```

Match_phrase : 구문검색

Query DSL – bool query : 여러 조건

```
GET index/type/_search
{
  "query": {
    "bool" : {
      "must" : {
        "term" : { "field1" : value }
      },
      "filter": {
        "term" : { "field2" : value }
      },
      "must_not" : {
        "range" : {
          "field3" : { "gte" : value, "lte" : value }
        }
      },
      "should" : [
        { "term" : { "field4" : value } },
        { "term" : { "field5" : value} }
      ],
      "minimum_should_match" : 1
    }
  }
}
```

Query DSL

```
#Query DSL - bool exam1
GET ecommerce/_doc/_search
{
  "query": {
    "bool": {
      "filter": {
        "term": {
          "geoip.country_iso_code.keyword": "AE"
        }
      }
    }
  }
}
```

Query DSL

```
#Query DSL - bool exam2
GET ecommerce/_doc/_search
{
  "query": {
    "bool": {
      "filter": {
        "term": {
          "geoip.country_iso_code.keyword": "AE"
        }
      },
      "must" : {
        "term" : { "day_of_week" : "Monday" }
      }
    }
  }
}
```

Query DSL

```
#Query DSL - bool exam3
GET ecommerce/_doc/_search
{
  "query": {
    "bool": {
      "filter": {
        "term": {
          "geoip.country_iso_code.keyword": "AE"
        }
      },
      "must" : {
        "term" : { "day_of_week" : "Monday" }
      },
      "must_not" : {
        "range" : {
          "taxful_total_price" : { "gte" : 100, "lte" : 150 }
        }
      }
    }
  }
}
```

Query DSL

```
#Query DSL - bool exam4
GET ecommerce/_doc/_search
{
  "query": {
    "bool": {
      "filter": {
        "term": {
          "geoip.country_iso_code.keyword": "AE"
        }
      },
      "must" : {
        "term" : { "day_of_week" : "Monday" }
      },
      "must_not" : {
        "range" : {
          "taxful_total_price" : { "gte" : 100, "lte" : 150 }
        }
      } ,
      "should" : [
        { "term" : { "customer_gender" : "MALE" } },
        { "term" : { "manufacturer.keyword" : "Elitelligence" } }
      ],
      "minimum_should_match" : 2
    }
  }
}
```

Aggregation

Aggregations

- Metrics Aggregations
- Bucket Aggregations
- Pipeline Aggregations
- Matrix Aggregations

Caching heavy aggregations

Returning only aggregation results

Aggregation Metadata

Returning the type of the aggregation

- Metrics Aggregations
  - Avg Aggregation
  - Cardinality Aggregation
  - Extended Stats Aggregation
  - Geo Bounds Aggregation
  - Geo Centroid Aggregation
  - Max Aggregation
  - Min Aggregation
  - Percentiles Aggregation
  - Percentile Ranks Aggregation
  - Scripted Metric Aggregation
  - Stats Aggregation
  - Sum Aggregation
  - Top Hits Aggregation
  - Value Count Aggregation

Metric Aggregations

Average

Count

Max

Median

Min

Percentile Ranks

Percentiles

Standard Deviation

Sum

Top Hit

Unique Count

- Bucket Aggregations
  - Adjacency Matrix Aggregation
  - Children Aggregation
  - Composite Aggregation
  - Date Histogram Aggregation
  - Date Range Aggregation
  - Diversified Sampler Aggregation
  - Filter Aggregation
  - Filters Aggregation
  - Geo Distance Aggregation
  - GeoHash grid Aggregation
  - Global Aggregation
  - Histogram Aggregation
  - IP Range Aggregation
  - Missing Aggregation
  - Nested Aggregation
  - Range Aggregation
  - Reverse nested Aggregation
  - Sampler Aggregation
  - Significant Terms Aggregation
  - Significant Text Aggregation
  - Terms Aggregation

Fast campus

Date Histogram

Date Range

Filters

Geohash

Histogram

IPv4 Range

Range

Significant Terms

Terms

- Pipeline Aggregations

Avg Bucket Aggregation

Derivative Aggregation

Max Bucket Aggregation

Min Bucket Aggregation

Sum Bucket Aggregation

Stats Bucket Aggregation

Extended Stats Bucket Aggregation

Percentiles Bucket Aggregation

Moving Average Aggregation

Cumulative Sum Aggregation

Bucket Script Aggregation

Bucket Selector Aggregation

Bucket Sort Aggregation

Serial Differencing Aggregation

Sibling Pipeline Aggregations

Average Bucket

Max Bucket

Min Bucket

Sum Bucket

Parent Pipeline Aggregations

Cumulative Sum

Derivative

Moving Avg

Serial Diff

Fast campus

Matrix Stats

```
#Aggregation - matrix agg.
GET ecommerce/_doc/_search
{
  "aggs": {
    "statistics": {
      "matrix_stats": {
        "fields": [
          "taxful_total_price",
          "total_quantity"
        ]
      }
    }
  },
  "size": 0
}
```

```
{
  "name" : "total_quantity",
  "count" : 4675,
  "mean" : 2.1585026737967916,
  "variance" : 0.35292030781270384,
  "skewness" : 2.667779775115108,
  "kurtosis" : 10.394715779121142,
  "covariance" : {
    "total_quantity" : 0.35292030781270384,
    "taxful_total_price" : 14.699481108831742
  },
  "correlation" : {
    "total_quantity" : 1.0,
    "taxful_total_price" : 0.46868353788892597
  }
}
```

실험적으로 적용 된 기능, 숫자형 Field에만 적용 가능

Matrix Aggregations

Matrix Stats

`count`	Number of per field samples included in the calculation.
`mean`	The average value for each field.
`variance`	Per field Measurement for how spread out the samples are from the mean.
`skewness`	Per field measurement quantifying the asymmetric distribution around the mean.
`kurtosis`	Per field measurement quantifying the shape of the distribution.
`covariance`	A matrix that quantitatively describes how changes in one field are associated with another.
`correlation`	The covariance matrix scaled to a range of -1 to 1, inclusive. Describes the relationship between field distributions.

Agg. : Metric

```
POST index/_search
{
  "aggs": {
    "agg_result_field_name": {
      "function": {
        "field": "field_name"
      }
    }
  }
}
```

function : avg, max, min, sum
cardinality(=unique count)
value_count(=count)
stats, extended_stats
percentiles, percentile_ranks
top_hits
geo_bounds, geo_centroid

Agg. : avg Metric

```
#aggregation - avg metric
POST ecommerce/_doc/_search
{
  "aggs": {
    "avg_taxful_total_price": {
      "avg": {
        "field": "taxful_total_price"
      }
    }
  }
}

POST ecommerce/_doc/_search
{
  "aggs": {
    "avg_taxful_total_price": {
      "avg": {
        "field": "taxful_total_price"
      }
    }
  },
  "size": 0
}
```

```
{
  "took" : 0,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 4675,
    "max_score" : 0.0,
    "hits" : [ ]
  },
  "aggregations" : {
    "avg_taxful_total_price" : {
      "value" : 75.05006301839084
    }
  }
}
```

Agg. : stats Metric

```
#aggregation - stats metric
POST ecommerce/_search
{
  "aggs": {
    "stats_taxful_total_price": {
      "stats": {
        "field": "taxful_total_price"
      }
    }
  },
  "size": 0
}

GET ecommerce/_search
{
  "aggs": {
    "stats_taxful_total_price": {
      "extended_stats": {
        "field": "taxful_total_price"
      }
    }
  },
  "size": 0
}
```

```
{
….
  "aggregations" : {
    "stats_score" : {
      "count" : 4675,
      "min" : 6.989999771118164,
      "max" : 2249.919921875,
      "avg" : 75.05006301839084,
      "sum" : 350859.0446109772,
      "sum_of_squares" : 3.93593610216703E7,
      "variance" : 2786.6026979773383,
      "std_deviation" : 52.788281824447914,
      "std_deviation_bounds" : {
        "upper" : 180.62662666728667,
        "lower" : -30.526500630504984
      }
    }
  }
}
```

Agg. : percentiles, percentile_ranks Metric

Fast campus

```
#aggregation - percentiles, percentile_ranks metric
POST ecommerce/_search
{
  "size": 0,
  "aggs": {
    "price_percentiles": {
      "percentiles": {
        "field": "taxful_total_price",
        "percents": [1,50,99]
      }
    }
  }
}
```

```
"aggregations" : {
  "price_percentiles" : {
    "values" : {
      "1.0" : 21.97999542236328,
      "50.0" : 64.1730991943018,
      "99.0" : 221.9749984741211
    }
  }
}
```

```
POST ecommerce/_search
{
  "size": 0,
  "aggs": {
    "price_percentile_ranks": {
      "percentile_ranks": {
        "field": "taxful_total_price",
        "values": [70, 80, 90]
      }
    }
  }
}
```

```
"aggregations" : {
  "price_percentile_ranks" : {
    "values" : {
      "70.0" : 55.00082750878762,
      "80.0" : 64.78904584201965,
      "90.0" : 72.47202370399248
    }
  }
}
```

Agg. : geo Metric

```
# aggregation - geo metric
PUT museums
{
   "mappings": {
      "doc": {
         "properties": {
            "location": {
               "type": "geo_point"
            }
         }
      }
   }
}

POST /museums/doc/_bulk?refresh
{"index":{"_id":1}}
{"location": "52.374081,4.912350", "name": "NEMO Science Museum"}
{"index":{"_id":2}}
{"location": "52.369219,4.901618", "name": "Museum Het Rembrandthuis"}
{"index":{"_id":3}}
{"location": "52.371667,4.914722", "name": "Nederlands Scheepvaartmuseum"}
{"index":{"_id":4}}
{"location": "51.222900,4.405200", "name": "Letterenhuis"}
{"index":{"_id":5}}
{"location": "48.861111,2.336389", "name": "Musée du Louvre"}
{"index":{"_id":6}}
{"location": "48.860000,2.327000", "name": "Musée d'Orsay"}
```

Agg. : geo Metric

```
# aggregation - geo metric exam1
POST /museums/_search?size=0
{
  "aggs": {
    "viewport": {
      "geo_bounds": {
        "field": "location",
        "wrap_longitude": true
      }
    }
  }
}
```

```
POST /museums/_search?size=0
{
  "query": {
    "match": {
      "name": "musée"
    }
  },
  "aggs": {
    "viewport": {
      "geo_bounds": {
        "field": "location",
        "wrap_longitude": true
      }
    }
  }
}
```

```
"hits": {
  "total": 6,
  ...
},
"aggregations": {
  "viewport": {
    "bounds": {
      "top_left": {
        "lat": 52.374080987647176,
        "lon": 2.3269999679178
      },
      "bottom_right": {
        "lat": 48.85999997612089,
        "lon": 4.91472196765244
      }
    }
  }
}
```

```
"hits": {
  "total": 2,
  ...
},
"aggregations": {
  "viewport": {
    "bounds": {
      "top_left": {
        "lat": 48.86111099738628,
        "lon": 2.3269999679178
      },
      "bottom_right": {
        "lat": 48.85999997612089,
        "lon": 2.3363889567553997
      }
    }
  }
}
```

Agg. : geo Metric

```
# aggregation - geo metric exam2
POST /museums/_search?size=0
{
   "aggs" : {
      "centroid" : {
         "geo_centroid" : {
            "field" : "location"
         }
      }
   }
}
```

```
{
  "took": 0,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 6,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "centroid": {
      "location": {
        "lat": 51.00982963107526,
        "lon": 3.9662130922079086
      },
      "count": 6
    }
  }
}
```

Agg. : Bucket

- **Bucket Aggregations**
  - Adjacency Matrix Aggregation
  - Children Aggregation
  - Composite Aggregation
  - Date Histogram Aggregation
  - Date Range Aggregation
  - Diversified Sampler Aggregation
  - Filter Aggregation
  - Filters Aggregation
  - Geo Distance Aggregation
  - GeoHash grid Aggregation
  - Global Aggregation
  - Histogram Aggregation
  - IP Range Aggregation
  - Missing Aggregation
  - Nested Aggregation
  - Range Aggregation
  - Reverse nested Aggregation
  - Sampler Aggregation
  - Significant Terms Aggregation
  - Significant Text Aggregation
  - Terms Aggregation

Date Histogram

Date Range

Filters

Geohash

Histogram

IPv4 Range

Range

Significant Terms

Terms

Agg. : date_histogram Bucket

```
# aggregation - date_histogram Bucket
POST ecommerce/_search
{
  "aggs": {
    "sales_over_time": {
      "date_histogram": {
        "field": "order_date",
        "interval": "week"
      }
    }
  }
}
```


```
"aggregations" : {
  "sales_over_time" : {
    "buckets" : [
      {
        "key_as_string" : "2018-12-03T00:00:00.000Z",
        "key" : 1543795200000,
        "doc_count" : 582
      },
      {
        "key_as_string" : "2018-12-10T00:00:00.000Z",
        "key" : 1544400000000,
        "doc_count" : 1048
      },
      {
        "key_as_string" : "2018-12-17T00:00:00.000Z",
        "key" : 1545004800000,
        "doc_count" : 1048
      },
      {
        "key_as_string" : "2018-12-24T00:00:00.000Z",
        "key" : 1545609600000,
        "doc_count" : 1073
      },
      {
        "key_as_string" : "2018-12-31T00:00:00.000Z",
        "key" : 1546214400000,
        "doc_count" : 924
      }
    ]
  }
}
```

알아두면 좋은 것

Elasticsearch 공식문서
: https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html
오른쪽 목차 활용

질문
: https://www.facebook.com/groups/elasticsearch.kr/
한글
: https://discuss.elastic.co/
영어

동영상 강좌
: https://www.elastic.co/kr/videos
한글
: https://www.elastic.co/videos
영어

Elasticsearch client
: https://www.elastic.co/guide/en/elasticsearch/client/index.html

감사합니다.

질의

1. Aggregation시 GET과 POST 차이

Fast campus

```
POST /exams/_search?size=0
{
    "aggs" : {
    "avg_grade" : { "avg" : { "field" : "grade" } }
    }
}

GET emails/_search
{
    "size": 0,
    "aggs" : {
        "interactions" : {
           "adjacency_matrix" : {
                "filters" : {
                    "grpA" : { "terms" : { "accounts" : ["hillary", "sidney"] }},
                    "grpB" : { "terms" : { "accounts" : ["donald", "mitt"] }},
                    "grpC" : { "terms" : { "accounts" : ["vladimir", "nigel"] }}
                 }
             }
        }
    }
}
```

xeraa Community Leaders 12m

It's the same thing — no difference. For regular search queries it's the same BTW.

Conceptually `GET` might make more sense, since you are just fetching data and it's an idempotent command. However, `GET` doesn't have a body and some clients (like browser plugins) might simply not send the body if you are doing a `GET`. So `POST` is an alternative, which has a body, but is maybe not the right HTTP verb for fetching data.

Generally we are trying to stick to REST conventions where possible, but we will also pick pragmatic alternatives if required.

https://discuss.elastic.co/t/what-is-difference-between-post-and-get-in-aggregation-request/162432

2. _version으로 rollback 할 수 있을까?

Fast campus

Rollback : 실행 취소(오류로 수정, 삭제 등의 동작을 했을 때 이전으로 돌림

Rollback documents and index

Elasticsearch

nkdev89 Naveen Kumar Oct '17

Hi ,

I'm new to ES , I'm looking for Rollback functionality to rollback new index and a document from another index if any exception occurred during the process. How can i do this ? , Can you please help me with this . Currently I'm using with NEST.

Thanks
Naveen

created	last reply	3	643	3
Oct '17	Nov '17	replies	views	users

warkolm Mark Walkom Elastic Team Member Oct '17

There is no way to do this in Elasticsearch. If you want rollback then you need to manage it externally.

Plugin 설치
: sudo bin/elasticsearch-plugin install analysis-nori

`nori` analyzer

The `nori` analyzer consists of the following tokenizer and token filters:

- `nori_tokenizer`
- `nori_part_of_speech` token filter
- `nori_readingform` token filter
- `lowercase` token filter

Fast campus

Mapping 설정

```
PUT nori_sample
{
  "settings": {
    "index": {
      "analysis": {
        "analyzer": {
          "my_analyzer": {
            "tokenizer": "nori_user_dic",
            "filter": [
              "my_posfilter"
            ]
          }
        },
        "tokenizer": {
          "nori_user_dict": {
            "type": "nori_tokenizer",
            "decompound_mode": "mixed",
            "user_dictionary": "userdict_ko.txt"
          }
        },
        "filter": {
          "my_posfilter": {
            "type": "nori_part_of_speech",
            "stoptags": [
              "NR"
            ]
          }
        }
      }
    }
  },
  "mappings": {
    "doc" : {
      "properties" : {
        "text" : {
          "type" : "text",
          "analyzer" : "my_analyzer"
        }
      }
    }
  }
}
```

- `nori_tokenizer`
- `nori_part_of_speech` token filter
- `nori_readingform` token filter
- `lowercase` token filter

3. Nori(한글 형태소 분석기) 사용 방법

```
"tokenizer": {
  "nori_user_dict": {
    "type": "nori_tokenizer",
    "decompound_mode": "mixed",
    "user_dictionary": "userdict_ko.txt"
  }
},
```

`decompound_mode`

The decompound mode determines how the tokenizer handles compound tokens. It can be set to:

`none`

No decomposition for compounds. Example output:

```
가거도항
가곡역
```

`discard`

Decomposes compounds and discards the original form (**default**). Example output:

```
가곡역 => 가곡, 역
```

`mixed`

Decomposes compounds and keeps the original form. Example output:

```
가곡역 => 가곡역, 가곡, 역
```

3. Nori(한글 형태소 분석기) 사용 방법

```
"tokenizer": {
  "nori_user_dict": {
    "type": "nori_tokenizer",
    "decompound_mode": "mixed",
    "user_dictionary": "userdict_ko.txt"
  }
},
```

`user_dictionary`

The Nori tokenizer uses the mecab-ko-dic dictionary by default. A `user_dictionary` with custom nouns (`NNG`) may be appended to the default dictionary. The dictionary should have the following format:

```
<token> [<token 1> ... <token n>]
```

The first token is mandatory and represents the custom noun that should be added in the dictionary. For compound nouns the custom segmentation can be provided after the first token (`[<token 1> ... <token n>]`). The segmentation of the custom compound nouns is controlled by the `decompound_mode` setting.

As a demonstration of how the user dictionary can be used, save the following dictionary to `$ES_HOME/config/userdict_ko.txt`:

```
c++                ❶
C샤프
세종
세종시 세종 시     ❷
```

❶ A simple noun

❷ A compound noun (`세종시`) followed by its decomposition: `세종` and `시`.

3. Nori(한글 형태소 분석기) 사용 방법

```
"filter": {
  "my_posfilter": {
    "type": "nori_part_of_speech",
    "stoptags": [
      "NR"
    ]
  }
}
```

part_of_speech : 품사
E : Verbal endings
IC : Interjection
J : Ending Particle
MAG : General Adverb
MAJ : Conjunctive adverb
MM : Determiner
…

`stoptags`

An array of part-of-speech tags that should be removed.

and defaults to:

```
"stoptags": [
    "E",
    "IC",
    "J",
    "MAG", "MAJ", "MM",
    "SP", "SSC", "SSO", "SC", "SE",
    "XPN", "XSA", "XSN", "XSV",
    "UNA", "NA", "VSV"
]
```

http://lucene.apache.org/core/7_5_0/analyzers-nori/org/apache/lucene/analysis/ko/POS.Tag.html

3. Nori(한글 형태소 분석기) 사용 방법

Fast campus

```
PUT nori_sample
{
    "settings": {
        "index":{
            "analysis":{
                "analyzer" : {
                    "my_analyzer" : {
                        "tokenizer" : "nori_tokenizer",
                        "filter" : ["nori_readingform"]
                    }
                }
            }
        }
    }
}

GET nori_sample/_analyze
{
  "analyzer": "my_analyzer",
  "text": "鄕歌"      ❶
}
```

COPY AS CURL VIEW IN CONSOLE

❶ A token written in Hanja: Hyangga

Which responds with:

```
{
  "tokens" : [ {
    "token" : "향가",      ❶
    "start_offset" : 0,
    "end_offset" : 2,
    "type" : "word",
    "position" : 0
  }]
}
```

4. PUT vs. POST

```
PUT shj_twitter/tweet/1
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch"
}

POST shj_twitter/tweet/
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch"
}

POST shj_twitter/tweet/1
{
   "user" : "kimchy",
   "post_date" : "2009-11-15T14:12:12",
   "message" : "trying out Elasticsearch"
}
```

Uniform Resource Locator (URL)	POST
Collection, such as `https://api.example.com/resources/`	*Create* a new entry in the collection. The new entry's URI is assigned automatically and is usually returned by the operation.[17]
Element, such as `https://api.example.com/resources/item17`	Not generally used. Treat the addressed member as a collection in its own right and create a new entry within it.[17]

https://en.wikipedia.org/wiki/Representational_state_transfer

5. devTools auto completion


```
PUT users
PUT users/_mapping/doc
{
  "properties": {
    "text" : {
      "type": ""
    }
  }
}
```

binary	API
boolean	API
byte	API
date	API
double	API
float	API
geo_point	API
geo_shape	API

```
PUT users
{
  "settings":
    "index.mapping.single_type": true
```

```
PUT users
PUT users/_mapping/doc
{
  "properties": {
    "text" : {
      "type": "text"
    }
  }
}
```

6. Mapping으로 규정해준 type과 다른 데이터가 Indexing 되는 것 방지

Fast campus

```
PUT my_index
{
  "mappings": {
    "doc": {
      "dynamic": false,
      "properties": {
        "number_one": {
          "type": "integer"
        },
        "number_two": {
          "type": "integer"
        }
      }
    }
  }
}

PUT my_index/doc/1
{
  "number_one": 1
}

PUT my_index/doc/2
{
  "number_two": "foo"
}
```

```
{
  "_index": "my_index",
  "_type": "doc",
  "_id": "1",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "_seq_no": 0,
  "_primary_term": 1
}
```

```
{
  "error": {
    "root_cause": [
      {
        "type": "mapper_parsing_exception",
        "reason": "failed to parse [number_two]"
      }
    ],
    "type": "mapper_parsing_exception",
    "reason": "failed to parse [number_two]",
    "caused_by": {
      "type": "number_format_exception",
      "reason": "For input string: ₩"foo₩""
    }
  },
  "status": 400
}
```

7. Elasticsearch의 디스크 용량 부족 시 처리하는 방법

Fast campus

Curator : curate, or manage, your Elasticsearch indices and snapshots

https://www.elastic.co/guide/en/elasticsearch/client/curator/5.5/about.html

Rollup and Delete raw data(by Curator)

```
actions:
  1:
    action: delete_indices
    options:
      ignore_empty_list: True
     continue_if_exception: False
      disable_action: False
    filters:
    - filtertype: pattern
      kind: prefix
      value: '^[a-z].*-'
      exclude:
    - filtertype: age
      source: creation_date
      direction: older
      unit: months
      unit_count: 1
      exclude:
```

```
PUT _xpack/rollup/job/sensor
{
    "index_pattern": "sensor-*",
    "rollup_index": "sensor_rollup",
    "cron": "*/30 * * * * ?",
    "page_size" :1000,
    "groups" : {
      "date_histogram": {
        "field": "timestamp",
        "interval": "1h",
        "delay": "7d"
      },
      "terms": {
        "fields": ["node"]
      }
    },
    "metrics": [
        {
            "field": "temperature",
            "metrics": ["min", "max", "sum"]
        },
        {
            "field": "voltage",
            "metrics": ["avg"]
        }
    ]
}
```

COPY AS CURL VIEW IN CONSOLE

8. Reindex : index의 mapping을 바꾸고 싶을때

1) 새로운 index(with new mapping 생성
2) Reindex api 수행

```
POST _reindex
{
  "source": {
    "index": "twitter"
  },
  "dest": {
    "index": "new_twitter"
  }
}
```

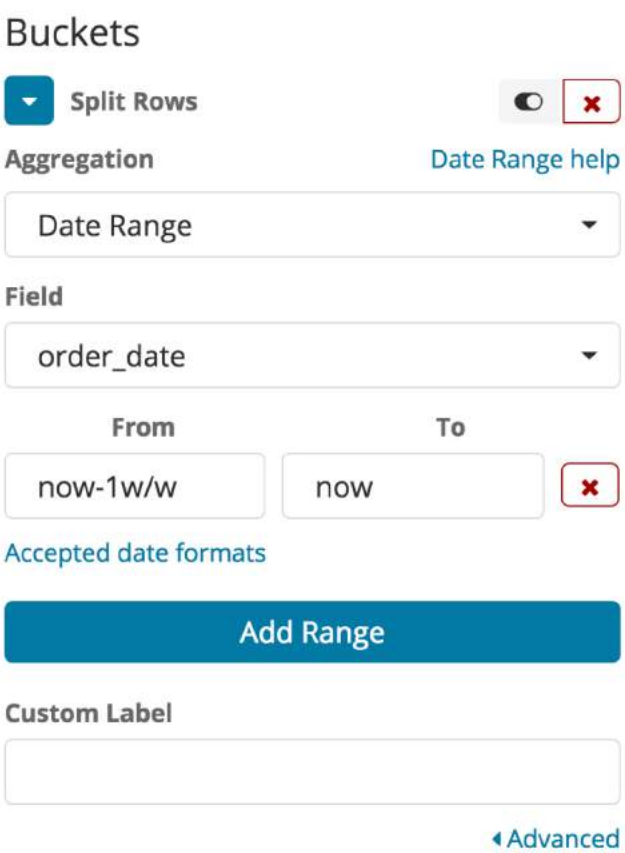
```
POST _reindex
{
  "source": {
    "index": "twitter",
    "type": "_doc",
    "query": {
      "term": {
        "user": "kimchy"
      }
    }
  },
  "dest": {
    "index": "new_twitter"
  }
}
```

```
POST /_reindex
{
  "source": {
    "index": "twitter"
  },
  "dest": {
    "index": "new_twitter",
    "version_type": "external"
  },
  "script": {
    "source": "if (ctx._source.foo == 'bar')
                { ctx._version++; ctx._source.remove('foo')}",
    "lang": "painless"
  }
}
```

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

9. Kibana에서 date range bucket agg. 수행시 오늘이 언제든지 매주 월요일부터 생성되는 date histogram을 만들고 싶음

Fast campus



Date Math

edit

Most parameters which accept a formatted date value — such as `gt` and `lt` in range queries `range` queries, or `from` and `to` in `daterange` aggregations — understand date maths.

The expression starts with an anchor date, which can either be `now`, or a date string ending with `||`. This anchor date can optionally be followed by one or more maths expressions:

- `+1h` - add one hour
- `-1d` - subtract one day
- `/d` - round down to the nearest day

The supported time units differ from those supported by time units for durations. The supported units are:

y	years
M	months
w	weeks
d	days
h	hours
H	hours
m	minutes
s	seconds

```
GET _search
{
    "query": {
        "range" : {
            "age" : {
                "gte" : 10,
                "lte" : 20,
                "boost" : 2.0
            }
        }
    }
}
```

```
GET _search
{
    "query": {
        "range" : {
            "born" : {
                "gte": "01/01/2012",
                "lte": "2013",
                "format": "dd/MM/yyyy||yyyy"
            }
        }
    }
}
```

9. Kibana에서 date range bucket agg. 수행시 오늘이 언제든지 매주 월요일부터 생성되는 date histogram을 만들고 싶음

Fast campus

Kibana

52.78.134.20:5601/app/kibana#/visualize/create?type=table&indexPattern=ff959d40-b880-11e8-a6d9-e546fe2bba5f&_g=(refreshInterval:(pau...

Visualize / New Visualization (unsaved)

Save Share Inspect Refresh Auto-refresh Last 60 days

Search... (e.g. status:200 AND extension:PHP)

Options Refresh

Add a filter +

kibana_sample_data_ecommerce

Data Options

Buckets

Split Rows

Aggregation Date Histogram help

Date Histogram

Field

order_date

Interval

Weekly

Custom Label

Advanced

JSON Input

{"offset": "+1d"}

Add sub-buckets

order_date per week	Count
2018-12-04	667
2018-12-11	1,047
2018-12-18	1,057
2018-12-25	1,074
2019-01-01	206

Export: Raw Formatted

2018.12

Date histogram, weekly interval을 하면 항상 월요일부터
Offset을 이용해 변경 가능(+24h -> 화요일부터)

10. Retention rate 계산하기

	1주차	2주차	3주차	4주차	5주차	6주차
1주차	50	42	35	27	22	18
2주차		40	35	34	34	32
3주차			64	40	54	42
4주차				62	24	36
5주차					55	43
6주차						37


```
{
  "_index" : "retention",
  "_type" : "retention",
  "_id" : "Dl2EC2gBDyTfqrTuxncr",
  "_score" : 1.0,
  "_source" : {
    "uid" : "3",
    "downDate" : "2018-12-03",
    "accessDate" : "2018-12-03"
  }
},
```