Elastic Stack
데이터 검색 및 집계

Kim Hye Kyung
topickim@naver.com

검색

검색 API

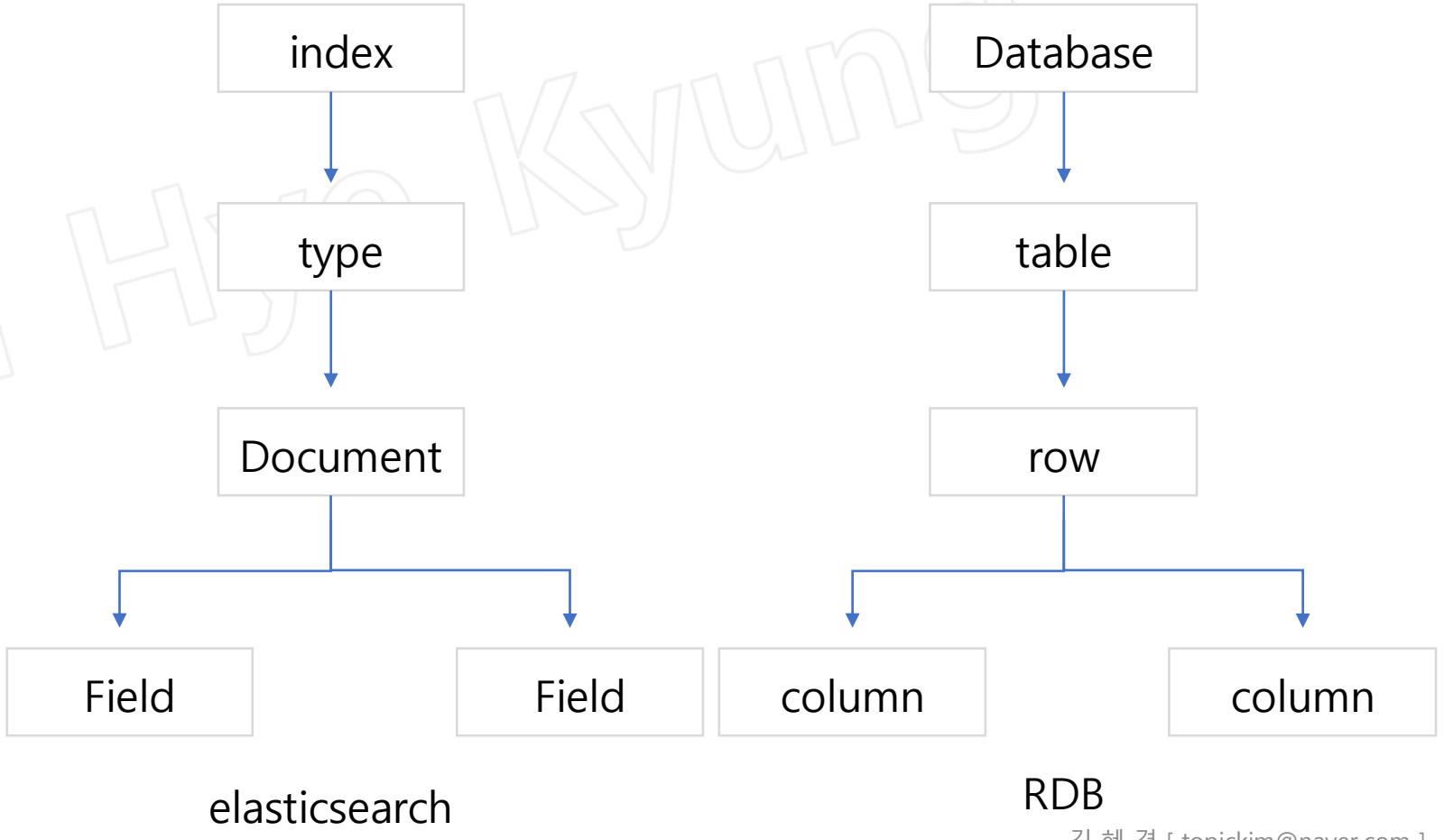
Query DSL

Query DSL의 주요 쿼리

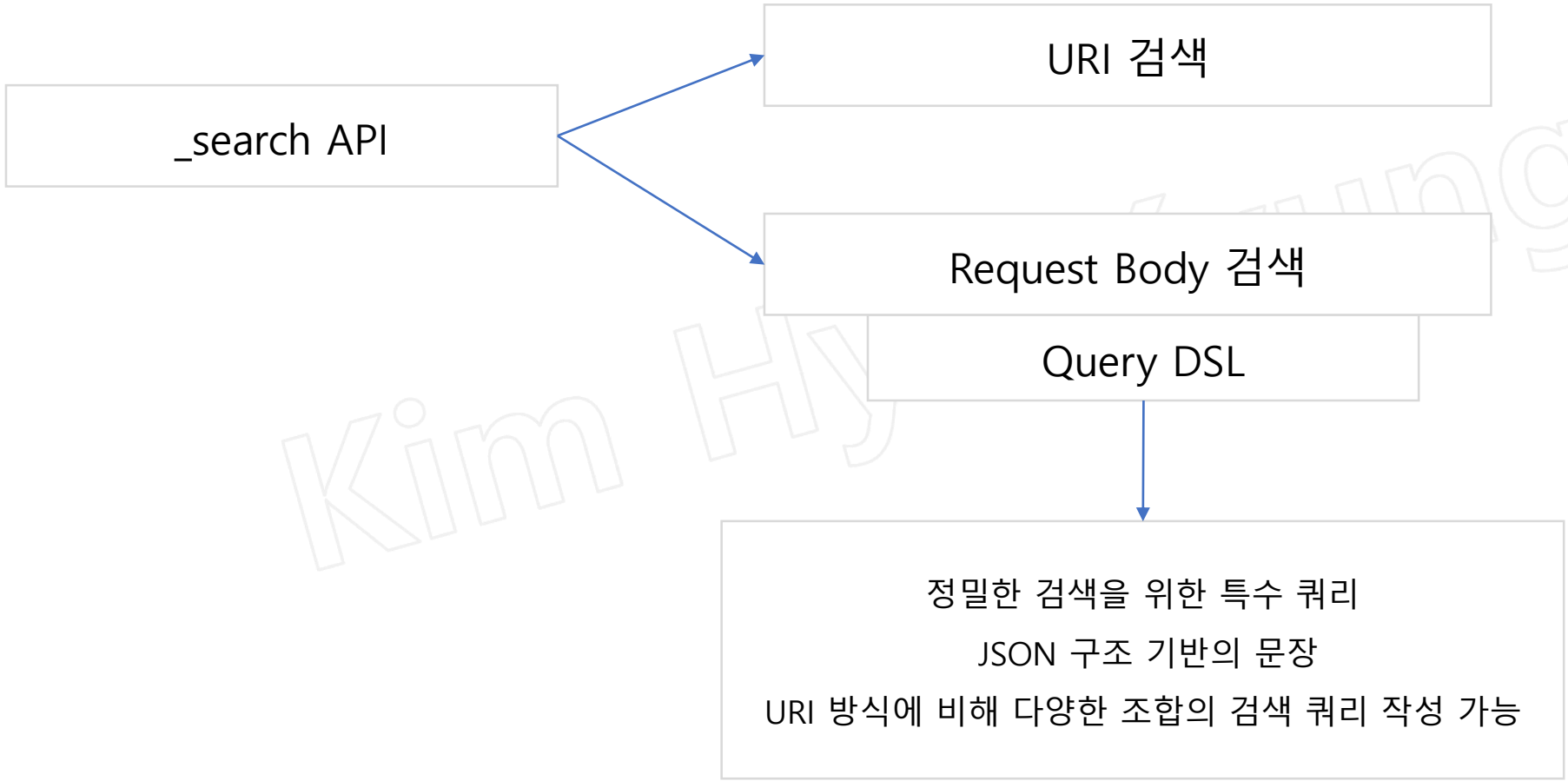
부가적인 검색 API

Elasticsearch 구성 이해하기

- 문서를 색인화 한다는 의미
  - Rest API를 통해 index에 document를 추가
- RDB와의 비교



Elasticsearch 검색 방식



Elasticsearch 검색 방식 : _search API

- 검색 API 방식 1 & 2

URI 검색	HTTP URI(Uniform Resource Identifier) 형태의 파라미터를 URI에 추가해 검색

```
GET twitter/_search?q=user:kimchy
```

Request Body 검색	RESTful API 방식인 QueryDSL을 사용해 요청 본문(request body)에 질의 내용을 추가해 검색

기본 구문

```
POST /{index명}/_search
{
   JSON 쿼리 구문
}
```

```
GET /_search
{
  "query": { 1
    "bool": { 2
      "must": [
        { "match": { "title":   "Search"        }}, 3
        { "match": { "content": "Elasticsearch" }}  4
      ],
      "filter": [ 5
        { "term":  { "status": "published" }}, 6
        { "range": { "publish_date": { "gte": "2015-01-01" }}} 7
      ]
    }
  }
}
```

Elasticsearch 검색 방식 : _search API

- 검색 API 방식 1 : URI 검색

HTTP URI(Uniform Resource Identifier) 형태의 파라미터를 URI에 추가해 검색

```
GET twitter/_search?q=user:kimchy
```

  - 문서 ID인 _id 값을 사용해 문서를 조회하는 방식
  - url에 parameter를 붙여 조회하는 방식
    - q

Elasticsearch 검색 방식 : _search API

- 검색 API 방식 2 - Request Body 검색

RESTful API 방식인 QueryDSL을 사용해 요청 본문(request body)에 질의 내용을 추가해 검색

기본 구문

```
POST /{index명}/_search
{
   JSON 쿼리 구문
}
```

```
GET /_search
{
  "query": { 1
    "bool": { 2
      "must": [
        { "match": { "title":   "Search"        }}, 3
        { "match": { "content": "Elasticsearch" }}  4
      ],
      "filter": [ 5
        { "term":  { "status": "published" }}, 6
        { "range": { "publish_date": { "gte": "2015-01-01" }}} 7
      ]
    }
  }
}
```

- 현업에서 더 선호 하는 방식
- JSON 형식으로 다양한 표현이 가능
- body에 검색할 칼럼과 검색어를 JSON 형태로 표현해서 전달
- 쿼리의 조건이 복잡하고 길어질 경우에 적합

Elasticsearch 검색 방식 : _search API

- URI 검색과 Request Body 검색 비교

URI 검색	GET user/_search?q=name:유재석

Request Body	POST user
	{
	"query" : {
	"term" : { "name" : "유재석"}
	}
	}

Elasticsearch 검색 방식 : _search API
: Request Body의 Response 분석 - 예시

```
GET /twitter/_search
{
    "query" : {
        "term" : { "user" : "kimchy" }
    }
}
```

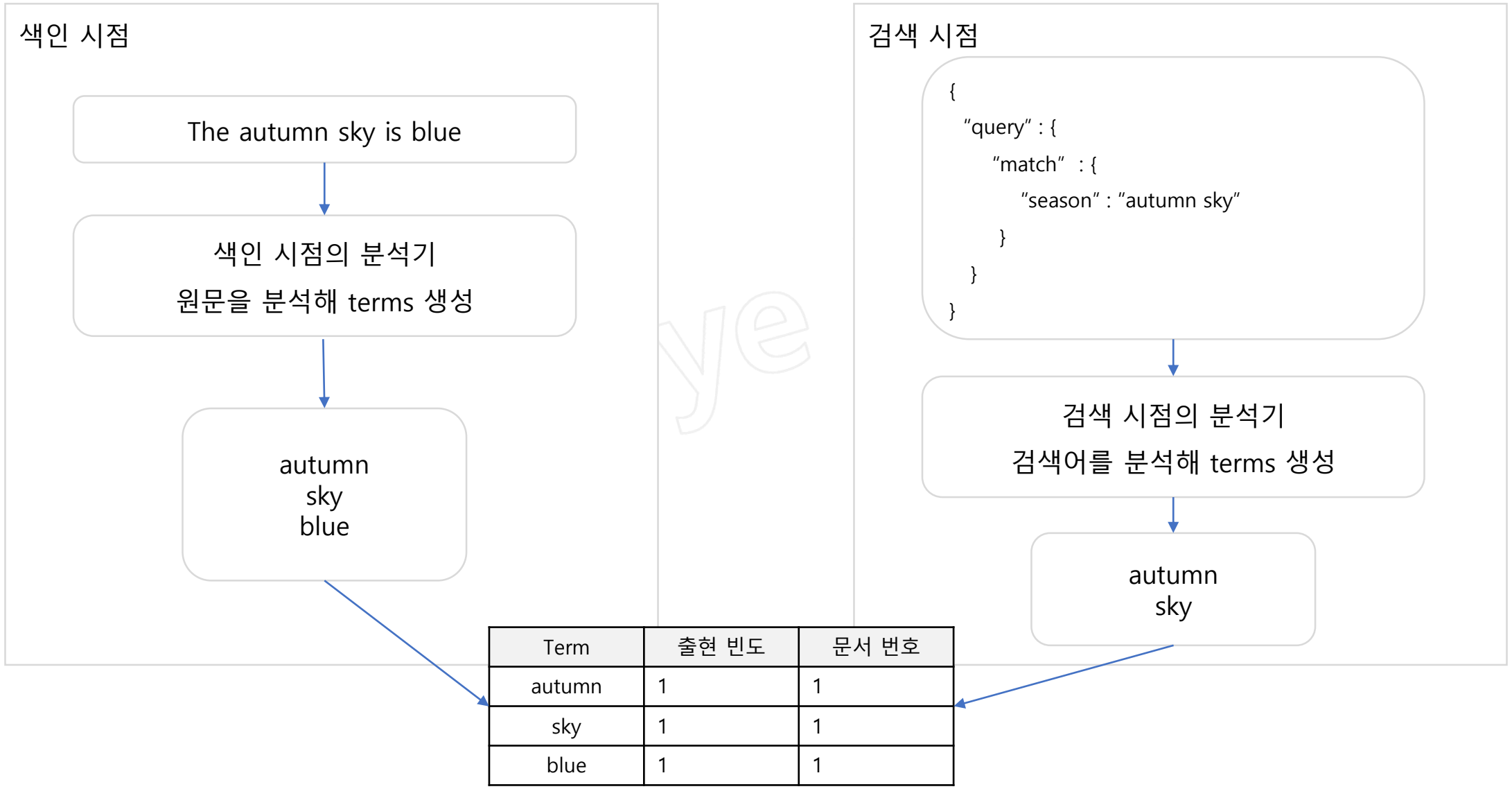

```
{
  "took" : 3,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : 0.105360515,
    "hits" : [
      {
        "_index" : "twitter",
        "_type" : "_doc",
        "_id" : "NyzsAG0BrvdlXMEXgSPp",
        "_score" : 0.105360515,
        "_routing" : "kimchy",
        "_source" : {
          "user" : "kimchy",
          "post_date" : "2009-11-15T14:12:12",
          "message" : "trying out Elasticsearch"
        }
      },
```


- took : 검색(쿼리 실행) 소요 시간(단위 : ms)
- timed_out : 검색시 시간 초과 여부
- _shards : 검색한 shard 수 및 검색에 성공 또는 실패한 shard 수
- hits : 검색 결과
  - total : 검색 조건과 일치하는 문서의 총 개수
  - max_score : 검색 조건과 일치하는 문서의 스코어 값중 가장 높은 값
  - hits : 검색 결과에 해당하는 실제 데이터들 (기본 값으로 10개가 설정되며, size를 통해 조절 가능)
    - _score :
      - 해당 document가 지정된 검색 쿼리와 얼마나 일치하는지를 상대적으로 나타내는 숫자 값
      - 높을수록 관련성이 높음

Elasticsearch 검색 방식 : _search API

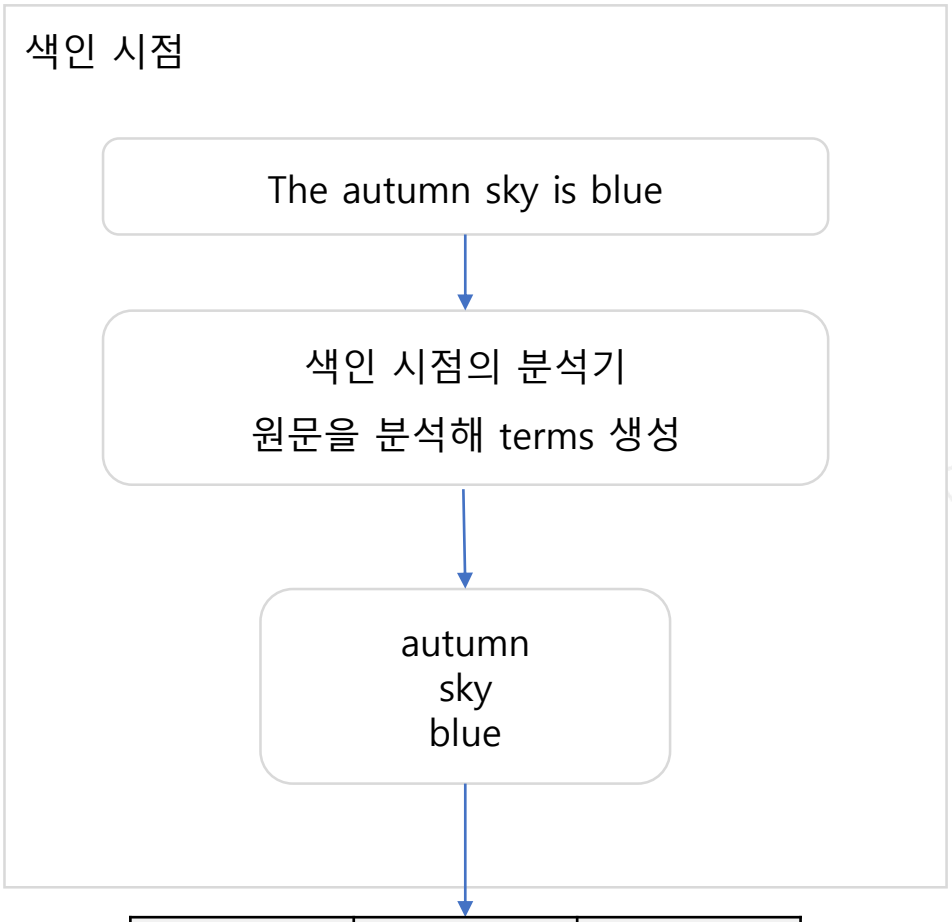
- 색인 시점
  - analyze를 통해서 분석된 terms을 terms, 출현빈도, 문서번호와 같이 역색인 구조로 만들어 내부적으로 저장
  - 예시
    - 색인 데이터 : The autumn sky is blue
    - 검색 데이터 : autumn sky

Elasticsearch 검색 방식 : _search API



Term	출현 빈도	문서 번호
autumn	1	1
sky	1	1
blue	1	1

Elasticsearch 검색 방식 : _search API

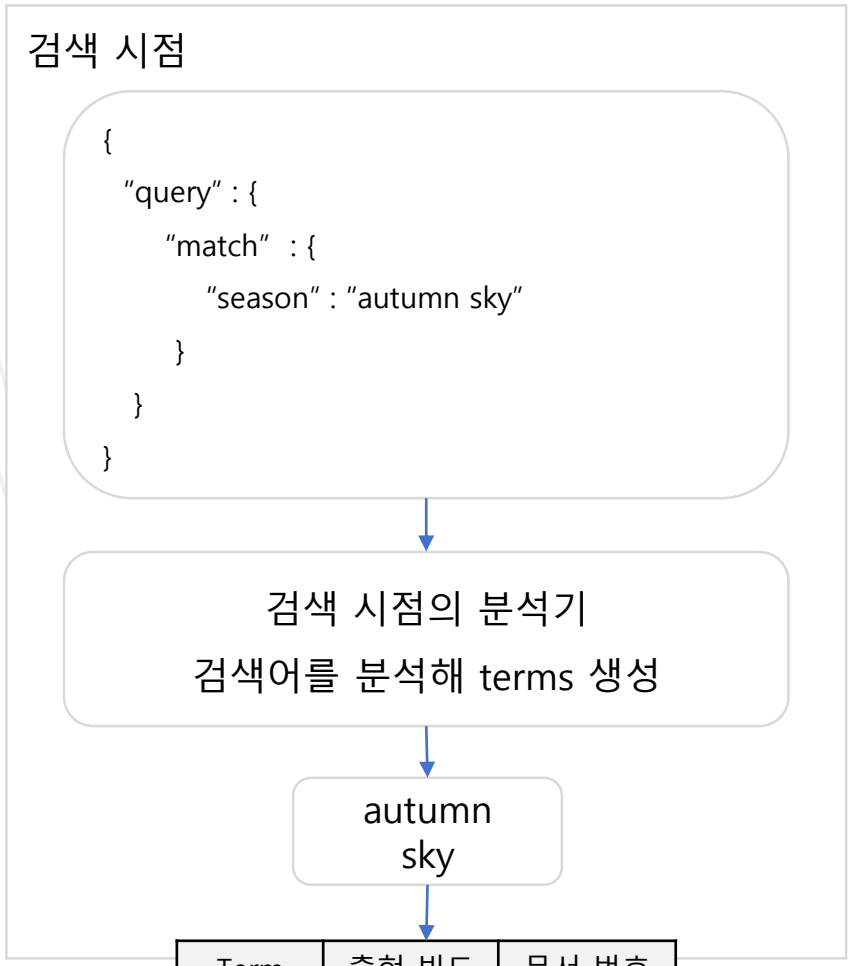


Term	출현 빈도	문서 번호
autumn	1	1
sky	1	1
blue	1	1

- 색인 시점 진행 단계
  - analyze를 통해 분석된 term을 Term, 출현빈도, 문서 번호와 같이 역색인 구조로 만들어 내부적으로 저장

Elasticsearch 검색 방식 : _search API

- 검색 시점 진행 단계
  - Keyword 타입과 같은 분석이 불가능한 데이터와 Text 타입과 같은 분석이 가능한 데이터를 구분해서 분석이 가능할 경우 분석기를 이용해 분석 수행
  - term 획득
  - 획득한 term으로 역색인 구조를 이용해 문서를 찾고 이를 통해 스코어 계산해서 결과값 제공



Term	출현 빈도	문서 번호
autumn	1	1
sky	1	1
blue	1	1

Elasticsearch 검색 방식

- Request Body 방식에 다양한 조합의 검색 쿼리 작성 할 수 있게 지원
  - 여러 개의 질의를 조합하거나 질의 결과에 대해 다시 검색을 수행하는 등 기존의 URI 검색보다 JSON 기반의 강력한 검색 기능 제공
- 다양한 parameter 옵션 제공

기본 구문

```
POST /{index명}/_search
{
   JSON 쿼리 구문
}
```

```
GET /_search
{
  "query": { 1
    "bool": { 2
      "must": [
        { "match": { "title":   "Search"        }}, 3
        { "match": { "content": "Elasticsearch" }}  4
      ],
      "filter": [ 5
        { "term":  { "status": "published" }}, 6
        { "range": { "publish_date": { "gte": "2015-01-01" }}} 7
      ]
    }
  }
}
```

Query DSL 이해하기 : Term Query

- Term Query
  - text 형태의 값을 검색하기 위한 두가지 매핑 유형
  - 문자형 데이터 타입

데이터 타입	설명
Text	필드에 데이터가 저장되기 전에 데이터가 분석되어 역색인 구조로 저장됨
Keyword	데이터가 분석되지 않고 그대로 필드에 저장됨

Query DSL 이해하기 : Term Query

- match query 과 term query 비교

match query

- 쿼리를 수행하기 전에 먼저 분석기를 통해 text를 분석한 후 검색 수행

term query

- 별도의 분석 작업을 수행하지 않고 입력된 text가 존재하는 문서 검색
- 결론
  - keyword 데이터 타입을 사용하는 field 검색시 사용해야 함
  - keyword 데이터 타입을 대상으로 하기 때문에 일반적으로 숫자, keyword, 날짜 데이터를 쿼리하는데 사용

Query DSL 이해하기 : bool Query

- RDB의 where절과 흡사

bool query 기본 문법

```
{
  "query": {
    "bool": {
      "must": [
        {}
      ],
      "must_not": [
        {}
      ],
      "should": [
        {}
      ],
      "filter": {}
    }
  }
}
```

Elasticsearch	SQL	설명
must	and column = 조건	반드시 조건에 만족하는 문서만 검색
must_not	and column != 조건	조건을 만족하지 않는 문서가 검색
should	or column = 조건	여러 조건중 하나 이상을 만족하는 문서가 검색
filter	column in (조건)	조건을 포함하고 있는 문서 검색

집계

집계 구문의 구조

Structuring Aggregations

The following snippet captures the basic structure of aggregations:

```
"aggregations" : {
    "<aggregation_name>" : {
        "<aggregation_type>" : {
            <aggregation_body>
        }
        [,"meta" : {  [<meta_data_body>] } ]?
        [,"aggregations" : { [<sub_aggregation>]+ } ]?
    }
    [,"<aggregation_name_2>" : { ... } ]*
}
```

- aggregations 단어 명시
  - aggs로 표현 가능
- aggregation_name
  - 하위 집계의 이름
  - 집계의 결과 출력에 사용
- aggregation_type
  - 집계의 유형
  - 어떤 집계를 먼저 수행할 것인가?
-