



Kibana 활용

오늘의 학습내용

- Aggregation
- Time Lion
- Dev Tools

◆ Aggregation

■ 집계(Aggregation)

- ❖ 집계 프레임워크는 검색 쿼리를 기반으로 집계된 데이터를 제공
- ❖ 집계는 문서 세트에 대해 분석적인 정보를 구축하는 작업 단위
- ❖ 검색 결과에 다양한 연산 적용
- ❖ RDBMS 의 groupby 와 유사
- ❖ 메트릭 집계, 버킷 집계, 파이프라인 집계, 매트릭스 집계, 누적 커디널리티 집계

◆ Kibana 모니터링

■ 매트릭(metric) 집계

- ❖ 도큐먼트를 계산해서 처리된 값
- ❖ 타입 리스트 : 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

◆ Kibana 모니터링

■ 매트릭(metric) 집계 구조

```
GET /{index_name}/_search?pretty
{
  "size": 0,
  "aggs": { // 집계
    "my_aggs": { // 집계 명 (사용자 정의)
      "{metric_arrgs_type}": { // 집계 타입
        "field": "{file_name}" // 집계 대상 필드
        "order" : { "{field_name}" : "desc" } // 정렬
      }...
    }
  }
```


◆ Kibana 모니터링

■ 버킷(bucket) 집계

- ❖ 조건에 해당하는 도큐먼트를 버킷이라는 저장소 단위로 구분해 담아 새로운 집합을 생성
- ❖ 타입 리스트 : Children Aggregation, Date Histogram Aggregation, Date Range Aggregation, Filter Aggregation, Filters Aggregation, Geo Distance Aggregation, GeoHash grid Aggregation, Global Aggregation, Histogram Aggregation, IPv4 Range Aggregation, Missing Aggregation, Nested Aggregation, Range Aggregation, Reverse nested Aggregation, Sampler Aggregation, Significant Terms Aggregation, Terms Aggregation

◆ Kibana 모니터링

■ 버킷(bucket) 집계 구조

```
GET /{index_name}/_search?pretty
{
  "size": 0,
  "aggs": { // 집계
    "my_aggs": { // 집계 명 (사용자 정의)
      "{bucket_arrgs_type}": { // 집계 타입
        "field": "{file_name}" // 집계 대상 필드
        "order" : { "{field_name}" : "desc" } // 정렬
      } ...
    }
  }
}
```

Kibana Discover

파이프라인(pipeline) 집계

- ❖ 집계 결과 다른 집계 활용
- ❖ 레벨에 따라 부모/자식 집계로 나뉨
- ❖ buckets_path 지정 필수
- ❖ 타입 리스트 : 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, Serial Differencing Aggregation

◆ Kibana Discover

■ 파이프라인(pipeline) 집계 구조 1

```
GET /hotels/_search?pretty
```

```
{  
  "size": 0,  
  "aggs": {  
    "{parent_aggs_name}": { // 부모 집계 이름 (사용자 정의)  
      "{bucket_type_name}": { // 버킷 집계 타입 이름  
        "field": "{field_name}" // 버킷 집계 대상 필드  
      },  
    },  
  },  
}
```

◆ Kibana Visualize

■ 파이프라인(pipeline) 집계 구조 2

GET /hotels/_search?pretty

```
"aggs": {  
  "{aggs_name}": { // 집계 이름 (사용자 정의)  
    "{metric_aggs_type}": { // 매트릭 집계 타입  
      "field": "{field_name}" // 매트릭 집계 대상 필드  
    } ...,  
  "{pipe_arrgs_name}": { // 파이프라인 집계 이름(사용자정의)  
    "{pipe_arrgs_type}": { // 파이프라인 집계 타입  
      "buckets_path": "{parent_aggs_name} > {aggs_name}" // 경로  
    } ...  
  }  
}
```

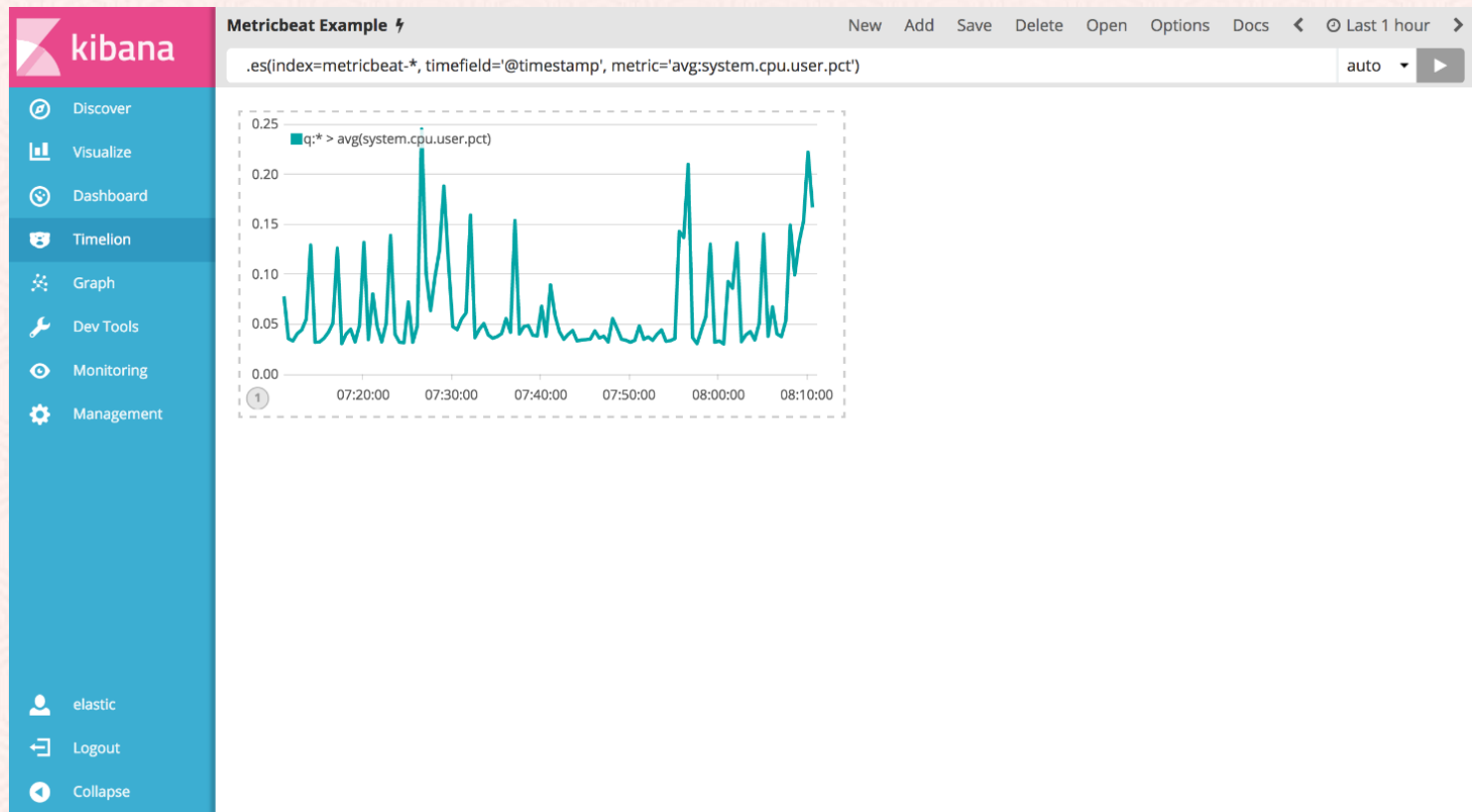
Time Lion

시계열 데이터 분석기 Timelion

- ❖ 서로 다른 데이터를(서로 다른 Index Pattern) 결합하여 하나의 시계열에서 시각화 할 수 있는 도구
- ❖ 완전히 독립적인 데이터 소스를 데이터 검색, 시계열 조합, 변환 및 시각화를 결합하는 간단한 1행 표현식 언어로 구동되는 단일 인터페이스로 통합
- ❖ timelion은 시계열 데이터 분석에 특화되어 있으며, 함수 형식의 표현식을 사용하고 자동완성을 지원
- ❖ 모든 Timelion 표현식은 데이터 소스 함수(예를 들어, `.elasticsearch(*)`) 또는 줄여서 `.es(*)` 로 시작됨

Time Lion

■ Metricbeat 시계열 데이터



Time Lion

대시보드에 추가

kibana

Discover

Visualize

Dashboard

Timelion

Graph

Dev Tools

Monitoring

Management

elastic

Logout

Collapse

Metricbeat Example

New Add Save Delete Open Options Docs Last 1 hour

Save entire Timelion sheet

You want this option if you mostly use Timelion expressions from within the Timelion app and don't need to add Timelion charts to Kibana dashboards. You may also want this if you make use of references to other panels.

Save current expression as Kibana dashboard panel

Need to add a chart to a Kibana dashboard? We can do that! This option will save your currently selected expression as a panel that can be added to Kibana dashboards as you would add anything else. Note, if you use references to other panels you will need to remove the references by copying the referenced expression directly into the expression you are saving. Click a chart to select a different expression to save.

Currently selected expression

`.es(index=metricbeat*, timefield=@timestamp, metric=max:system.network.in.bytes).derivative().divide(1048576).lines(fill=2, width=1).color(green).label("Inbound traffic").title("Network traffic (MB/s)", .es(index=metricbeat*, timefield=@timestamp, metric=max:system.network.out.bytes).derivative().multiply(-1).divide(1048576).lines(fill=2, width=1).color(blue).label("Outbound traffic").legend(columns=2, position=now)`

Save expression as

Network traffic (MB/s)

Save

`.es(index=metricbeat*, timefield=@timestamp, metric=max:system.network.in.bytes).derivative().divide(1048576).lines(fill=2, width=1).color(gre`

auto

CPU usage over time

last hour

current hour

1

Memory consumption over time

max memory

warning

severe

mvavg

3

Network traffic (MB/s)

Inbound traffic

Outbound traffic

2

13/16

◆ Dev Tools

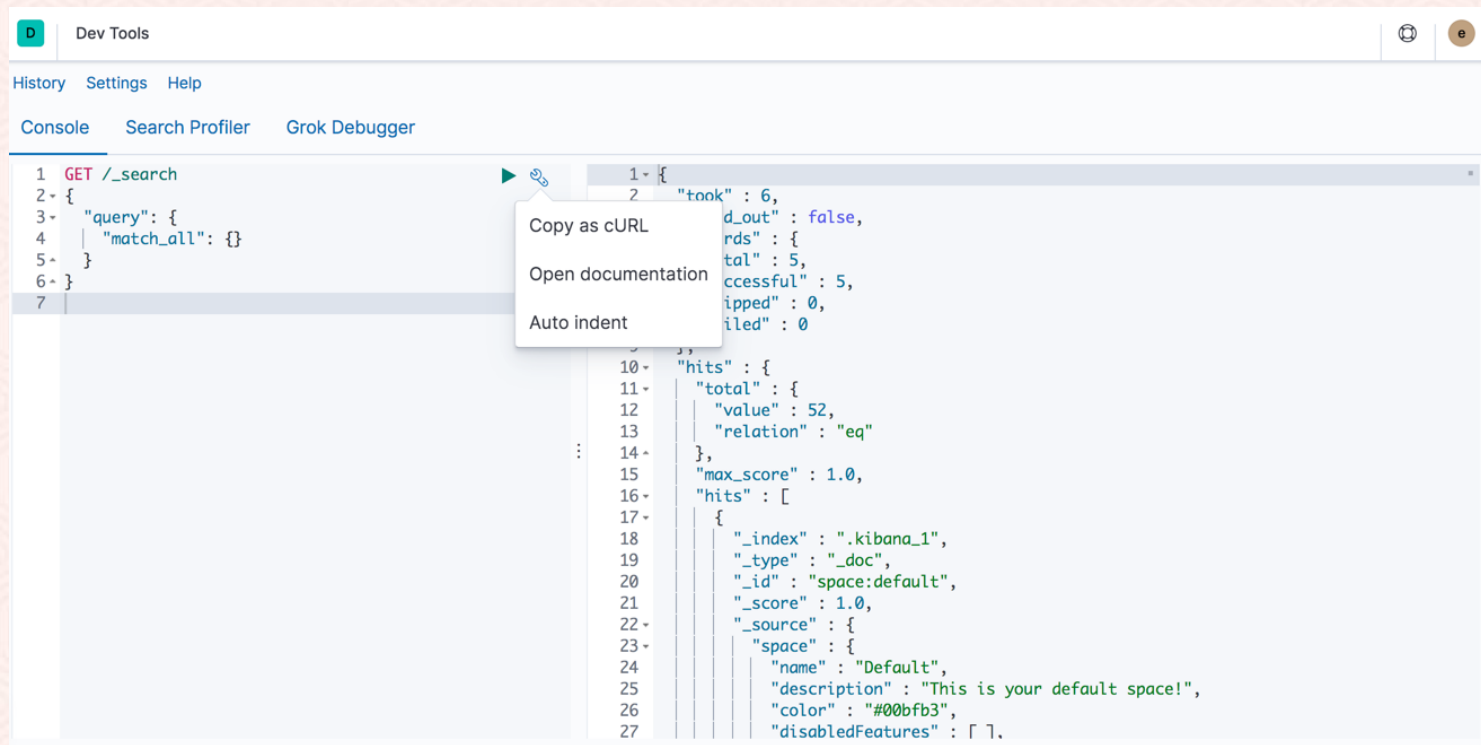
■ RESTFuI 시스템

- ❖ Elasticsearch는 http 프로토콜로 접근이 가능한 REST API를 지원
- ❖ Kibana에서 elasticsearch 의 REST API를 호출할 수 있는 편리한 UI 툴
- ❖ 자원별로 고유 URL로 접근이 가능하며 http 메서드 PUT, POST, GET, DELETE 를 이용해서 자원을 처리
- ❖ RESTFuI 시스템에서의 데이터 처리
 - 입력 : PUT `http://user.com/kim` -d {"name":"kim", "age":38, "gender":"m"}
 - 조회 : GET <http://user.com/kim>
 - 삭제 : DELETE <http://user.com/kim>

Dev Tools

Console

❖ Kibana에서 Elasticsearch 의 REST API 사용 가능



The screenshot shows the Kibana Dev Tools interface. The 'Console' tab is active, displaying a REST API call and its response. The call is a GET request to `/_search` with a query object containing `"match_all": {}`. The response is a JSON object with a `"took"` field of 6, a `"timed_out"` field of false, and a `"hits"` array containing one document. The document has fields for `"_index"`, `"_type"`, `"_id"`, `"_score"`, `"_source"`, and `"disabledFeatures"`. A context menu is open over the response, showing options: 'Copy as cURL', 'Open documentation', and 'Auto indent'.

```
1 GET /_search
2 {
3   "query": {
4     "match_all": {}
5   }
6 }
7

1- {
2   "took" : 6,
3   "timed_out" : false,
4   "total_hits" : 5,
5   "successful_hits" : 5,
6   "failed_hits" : 0,
7   "hits" : [
8     {
9       "_index" : ".kibana_1",
10      "_type" : "_doc",
11      "_id" : "space:default",
12      "_score" : 1.0,
13      "_source" : {
14        "name" : "Default",
15        "description" : "This is your default space!",
16        "color" : "#00bfb3",
17        "disabledFeatures" : [ ]
18      }
19    }
20  ]
21 }
```

Dev Tools

Search Profiler

❖ Kibana에 install X-Pack하면 Search Profiler가 자동으로 활성화

The screenshot displays the Kibana Search Profiler interface. On the left, the 'Index' dropdown is set to '_all' and the 'Type' dropdown is empty. Below these, a JSON query is shown:

```
{
  "query": {
    "match_all": {}
  }
}
```

. The main panel is titled 'Query Profile' and shows performance metrics for four indices: .kibana_1, .kibana_task_manager, .security-7, and test. Each index entry includes a query ID, a bar chart representing cumulative time, and a table of self and total times with percentages. The .kibana_1 entry is expanded, showing a ConstantScoreQuery and a nested DocValuesFieldExistsQuery.

Index	Query	Cumulative Time	Self time	Total time	Percentage
.kibana_1	[Uc_zxmN2TzqiGnKqN8k4FA][0]	0.067ms	0.0ms	0.1ms	100.00%
	ConstantScoreQuery				
	DocValuesFieldExistsQuery [field=_primary_term]		0.0ms	0.0ms	57.14%
.kibana_task_manager	[Uc_zxmN2TzqiGnKqN8k4FA][0]	0.016ms			
.security-7	[Uc_zxmN2TzqiGnKqN8k4FA][0]	0.028ms			
test	[Uc_zxmN2TzqiGnKqN8k4FA][0]	0.014ms			