

# ElasticSearch의 data저장, 수정 및 Kibana를 이용한 시각화 Tutorial

(이삭엔지니어링 인턴 김형근 [hkkim@isaac-eng.com](mailto:hkkim@isaac-eng.com))

(DIA 클러스터에서 실습. Elasticsearch가 이미 설치 되어있는 환경)

Elasticsearch Kibana 연결 주소 : master1.isaac-eng.com:5601

## Index

1. ElasticSearch Cluster 로 Data Load
2. Kibana를 이용한 시각화

# 1. Elasticsearch Cluster 로 Data Load

1. <https://www.elastic.co/guide/en/kibana/current/tutorial-load-dataset.html> 위의 사이트에서 3가지 파일(Shakespeare.json accounts.zip logs.jsonl.gz)을 다운받는다.

- The complete works of William Shakespeare, suitably parsed into fields. Download this data set by clicking here: [shakespeare.json](#).
- A set of fictitious accounts with randomly generated data. Download this data set by clicking here: [accounts.zip](#)
- A set of randomly generated log files. Download this data set by clicking here: [logs.jsonl.gz](#)

다운로드가 완료되었는지 살펴보면 console창에서

```
드 samadal@samadal-virtual-machine:~/다운로드$ ls -la
합계 33200
drwxr-xr-x  2 samadal samadal    4096  7월  3 14:04 .
drwxr-xr-x 20 samadal samadal    4096  7월  3 13:06 ..
-rw-rw-r--  1 samadal samadal  57700  7월  3 14:03 accounts.zip
-rw-rw-r--  1 samadal samadal 8705693  7월  3 14:03 logs.jsonl.gz
-rw-rw-r--  1 samadal samadal 25216068  7월  3 14:04 shakespeare.json
```

이와 같이 다운이 완료된 것을 확인 할 수 있다.

이때, 압축된 파일을 아래의 명령어로 압축 해제를 해준다.

> unzip 파일이름.zip

> gunzip 파일이름.gz

```
드 samadal@samadal-virtual-machine:~/다운로드$ unzip accounts.zip
Archive:  accounts.zip
  inflating: accounts.json
드 samadal@samadal-virtual-machine:~/다운로드$ gunzip logs.jsonl.gz
```

압축 완료된 화면을 살펴보면

```
합 계 77144
drwxr-xr-x 2 hkkim isaac 4096 1월 23 11:04 .
drwx----- 6 hkkim isaac 4096 1월 22 15:42 ..
-rw-r--r-- 1 hkkim isaac 244848 3월 5 2014 accounts.json
-rw-r--r-- 1 hkkim isaac 57700 1월 22 15:24 accounts.zip
-rw-r--r-- 1 hkkim isaac 53347384 1월 23 11:04 logs.jsonl
-rw-r--r-- 1 hkkim isaac 25327465 1월 23 11:04 shakespeare_6.0.json
```

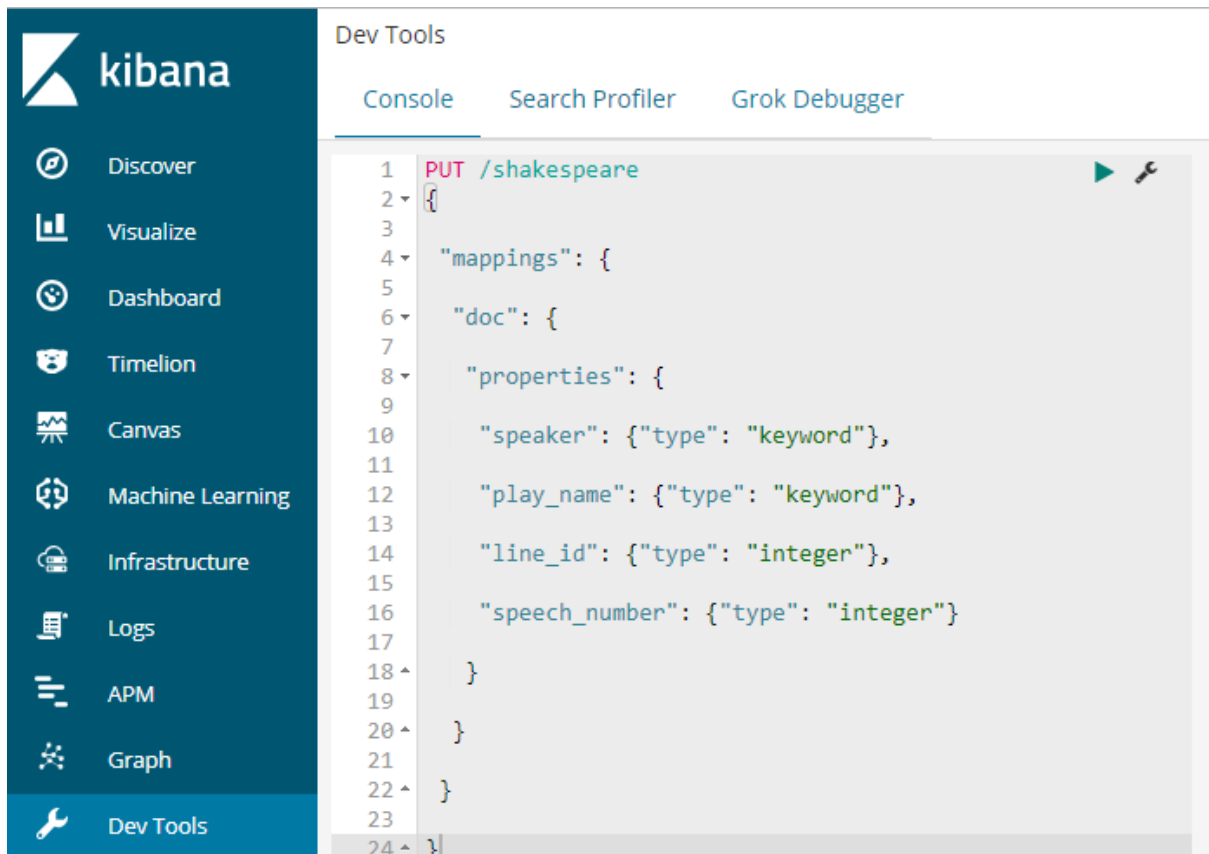
위와 같이 압축이 해제가 완료된 것을 확인 가능하다.

이때, Shakespeare 데이터 집합을 살펴보면 아래와 같은 스키마(구조)로 작성이 되어있는데,

```
{
  "line_id": INT,
  "play_name": "String",
  "speech_number": INT,
  "line_number": "String",
  "speaker": "String",
  "text_entry": "String",
}
```

이를 Elasticsearch Cluster에 데이터를 업로드(Load) 하기 전 먼저 정의된 구조, index 이름 정보를 cluster에 알려준다. 즉 Elasticsearch Cluster가 Shakespeare 데이터 집합이 어떠한 스키마(구조, 이름)로 이루어졌는지 알 수 있게 뼈대를 알려주는 것을 의미한다.

- **NOTE:** mapping이란, data 집합의 스키마를 생성하는 것을 의미한다.



➤ `PUT /`(새로 지을 index이름) { 'mapping' 내용 }

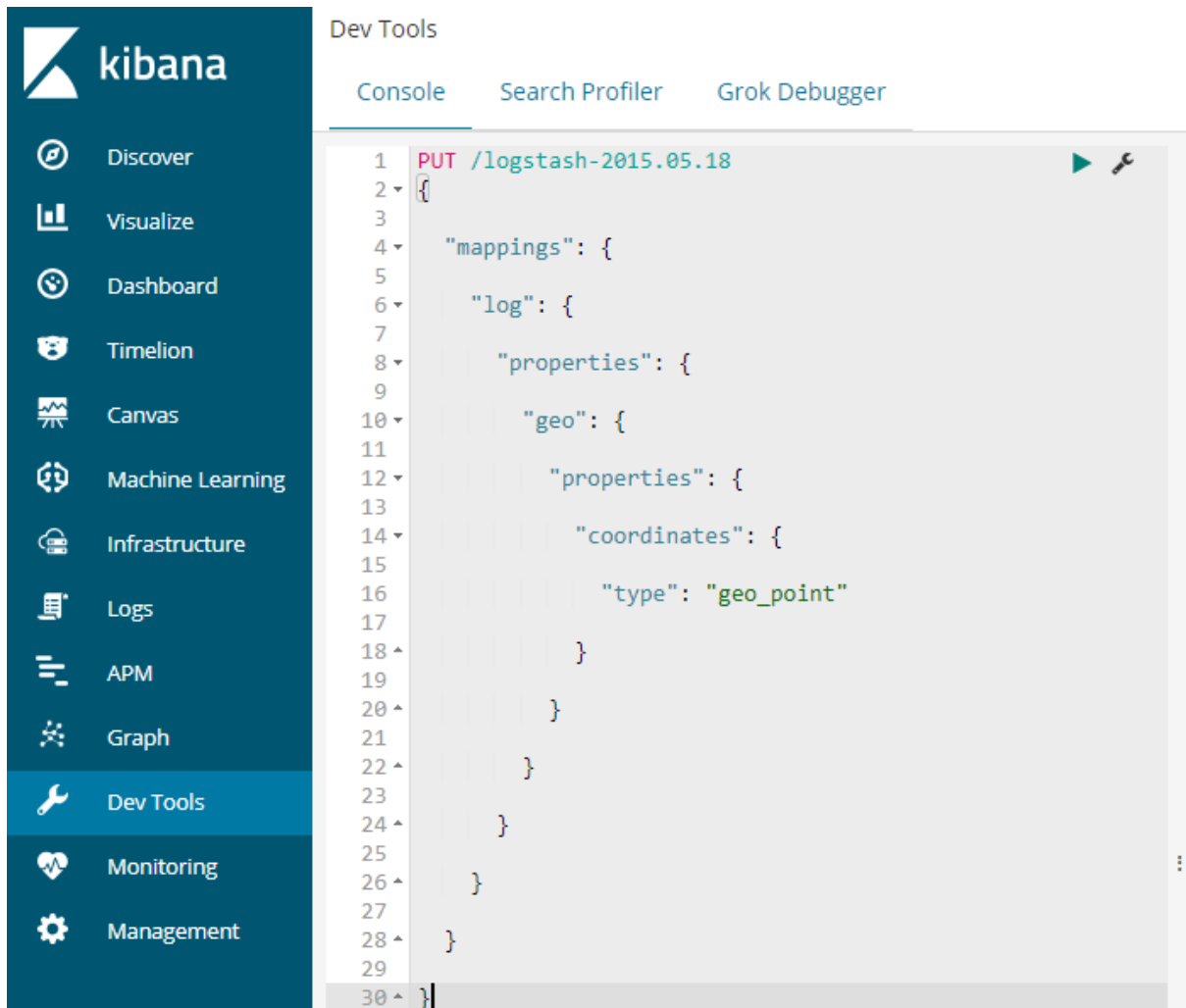
위의 명령어를 살펴보면, end url에 어떠한 index이름을 설정하고, 그 값을 pretty option을 주어 cluster에 등록을 하는 것을 의미한다.

이를 정상적으로 작동 하였을 시에 elasticsearch cluster에서는

```
{
  "acknowledged" : true,
  "shards_acknowledged" : true,
```

이와 같은 Ack(Acknowledgement)를 준다.

마찬가지로 log파일 역시 geo\_point 라는 스키마를 만들기 위해서 아래와 같이 mapping을 실시한 후에 elasticSearch cluster에 그 구조를 알리게 된다.



> PUT /logstash-2015.05.18 { 'mapping' 내용 } // logstash-2015.05.18 ~ 2015.05.20 3개 PUT하기.

```
[hkkim@master1 elasticsearch_hkkim]$ curl -H 'Content-Type: application/x-ndjson' -XPOST '10.100.3.210:9200/shakespeare/doc/_bulk?pretty' --data-binary @shakespeare_6.0.json
```

그리고 ElasticSearch Cluster에 뼈대를 알려주었으므로 이전에 다운받았던 Shakespeare.json 파일을 이용해서 Data들을 업로드(POST)를 시켜준다.

```
>> curl -H 'Content-Type: 파일 형식' -XPOST '포스트를 할 위치/index이름/_bulk?option' -data-binary @파일명
```

```
>> curl -H 'Content-Type:application/x-ndjson' -XPOST master1.isaac-eng.com:9200/shakespeare/doc/_bulk?pretty' --data-binary @shakespeare_6.0.json
```

의 명령어의 형식을 이용한다. Content-Type은 어떠한 형식인지를 정의해 주는 것이고, \_bulk API를 이용하여 option 값을 적용하여 파일을 읽어들이고 ElasticSearch Cluster에 data를 저장시키는 것이다.

마찬가지로 같은 명령어를 이용하여 log.jsonl 파일과, accounts.json 파일 역시 elasticSearch cluster에 Post를 시켜준다.

```
[hkkim@master1 elasticsearch_hkkim]$ curl -H 'Content-Type: application/x-ndjson' -XPOST '10.100.3.210:9200/bank/account/_bulk?pretty' --data-binary @accounts.json
```

```
>>curl -H 'Content-Type:application/x-ndjson' -XPOST 'master1.isaac-eng.com:9200/bank/account/_bulk?pretty' --data-binary @accounts.json
```

```
[hkkim@master1 elasticsearch_hkkim]$ curl -H 'Content-Type: application/x-ndjson' -XPOST '10.100.3.210:9200/_bulk?pretty' --data-binary @logs.jsonl
```

```
>>curl -H 'Content-Type:application/x-ndjson' -XPOST
master1.isaac-eng.com:9200/bulk?pretty' --data-binary @logs.jsonl
```

Post 가 잘되었는지 확인을 위하여

```
>> curl -Xget 주소/_cat/indices?pretty
```

```
>> curl -XGET master1.isaac-eng.com:9200/_cat/indices?v
```

를 사용하여

```
[hkkim@master1 elasticsearch_hkkim]$ curl -XGET 10.100.3.210:9200/_cat/indices?v
```

health	status	index	uuid	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
green	open	shakespeare	U_x9p4B5RI-SSiYg7kaQ1A	5	1	111396	31794	53.9mb	27.8mb
green	open	estest2	0ADehrgDOLqaa6Qwlb_1ZA	5	1	0	0	2.5kb	1.2kb
green	open	hk_shakespeare	1dfjHxqQRHW6Lbso43QE1w	5	1	0	0	2.2kb	1.1kb
green	open	.triggered_watches	x00usNfYRo6rw30-V3ejig	1	1	0	0	14.4mb	7.2mb
green	open	radio	gNt1mdl5RZqI_SU0jW-r0Q	5	1	1	0	7.7kb	3.8kb
green	open	hkkim_shakespeare	-WvsciIBQEqAOP7TACuKKg	5	1	10000	0	4.3mb	2.2mb
green	open	.monitoring-kibana-6-2019.01.19	jUkOHokLQqy4mJgk790Lug	1	1	17272	0	7.4mb	3.7mb
green	open	.monitoring-es-6-2019.01.18	oER3EKUzSyKct2dBQ5wfuQ	1	1	545510	2984	718.2mb	361.6mb
green	open	logstash-2015.05.19	gnwD0FKoT96mOfumINz5dQ	5	1	4624	0	40.7mb	20.3mb
green	open	.monitoring-es-6-2019.01.22	4YwDh41mSeuCXpXA-nEACA	1	1	634760	5680	841.3mb	428.3mb
green	open	dataset_power_consumption	cXKC_XlXTAi5ncFaeYkTow	5	1	2681149	0	390.4mb	195.1mb
green	open	.monitoring-kibana-6-2019.01.23	iwje8HvfQ5SrGFBKxtJ62A	1	1	5968	0	3.1mb	1.5mb
green	open	.watcher-history-9-2019.01.22	Lr5BUBvETjq309GagfNI-g	1	1	8572	0	22.9mb	11.5mb
green	open	production_capacity	RoXD5DtFRUOH5n2dIWSGOQ	5	1	1029	0	200.6kb	100.3kb
green	open	ddd	YAlq6IwiSv2amRtw9b5KCA	5	1	947	0	306.9kb	153.4kb
green	open	ccc	qCMRX1lHs-SH2nVcmkBLXA	5	1	56	0	139.3kb	69.6kb
green	open	.monitoring-es-6-2019.01.17	MVjGVOjvS06XS1NYCmjJ0w	1	1	545811	2978	715.4mb	358.6mb
green	open	.monitoring-kibana-6-2019.01.16	5dJ6A0VUR2aLsv14QoArMQ	1	1	17274	0	8.1mb	4mb
green	open	logstash-2015.05.18	G2SHzZXDRaEueW44Xav06A	5	1	4631	0	40.4mb	20.2mb
green	open	es_bi_twitter_christmas	bjhWZ2ghTACHooiQVaaFvA	5	1	215005	0	116.2mb	57.9mb
green	open	hkkim_logstash-2015.05.18	dbGqF_9ZTHaVDP879zJucQ	5	1	0	0	2.5kb	1.2kb
green	open	.kibana_1	iz7XZGXuTRCLCSdG8y1oQw	1	1	39	0	171.8kb	85.9kb
green	open	.watcher-history-9-2019.01.19	LsEoYax6QPG8Bdi0y1Fg	1	1	8562	0	22mb	11mb

Figure 2. ElasticSearch Cluster data 집합

Figure.2와 같은 값이 출력되는지 확인한다.

## 2. Kibana를 이용한 시각화

The screenshot shows the Kibana Management interface. On the left is a sidebar with navigation links: Discover, Visualize, Dashboard, Timelion, Canvas, Machine Learning, Infrastructure, Logs, APM, Graph, Dev Tools, Monitoring, and Management (selected). The main content area is titled 'Management / Kibana' and includes tabs for Index Patterns, Saved Objects, Spaces, Reporting, and Advanced Settings. The 'Index Patterns' tab is active, showing a list of index patterns under the 'sensor\*' pattern. A 'Create index pattern' button is at the top. Below the list, a table displays fields for the 'sensor\*' index. The table has columns for Name, Type, and Format. Fields listed include O2Dens1 (number), O2Quan1 (number), VoC1 (number), \_id (string), \_index (string), \_score (number), \_source (string), \_type (string), refGas1 (number), and temperatue1 (number). A filter bar and a 'Rows per page' dropdown are also visible.

Name	Type	Format
O2Dens1	number	
O2Quan1	number	
VoC1	number	
_id	string	
_index	string	
_score	number	
_source	string	
_type	string	
refGas1	number	
temperatue1	number	

Kibana Management 메뉴에서는 Index pattern 을 정의 할 수 있습니다.

index란 RDBMS 에서의 table과 같은 역할을 의미하는데, Index pattern 이란 와일드카드(wildcards \*) 와 String을 합친 것으로서 여러 개의 index를 선택 하는 것을 의미한다. 예를 들어서 "ba\*"를 사용하면 "ba"로 시작하는 모든 index를 가르키게 된다..

### Step 1 of 2: Define index pattern

#### Index pattern

sha\*

You can use a \* as a wildcard in your index pattern.  
You can't use spaces or the characters \, /, ?, ", <, >, |.

✓ **Success!** Your index pattern matches **1 index**.

shakespeare


(위와 같이 wildcard를 써서 해당 index pattern이 존재하면 만들 수 있다.)

## Step 1 of 2: Define index pattern

Index pattern

sh\*

You can use a \* as a wildcard in your index pattern.  
You can't use spaces or the characters \, /, ?, ", <, >, |.

 There's already an index pattern called sh\*

(위와 같이 이미 같은 이름을 가진 Index는 만들 수 없다.)

여기서 Logstash 데이터 집합의 경우는 시간 값을 갖고 있으므로

## Step 2 of 2: Configure settings

You've defined **logs\*** as your index pattern. Now you can specify some settings before we create it.

Time Filter field name

[Refresh](#)

@timestamp



The Time Filter will use this field to filter your data by time.  
You can choose not to have a time field, but you will not be able to narrow down your data by a time range.

[> Show advanced options](#)

Time Filter field name을 @timestamp로 설정한다.

작성자 본인은 index pattern을

**shakespeare = sh\*,**

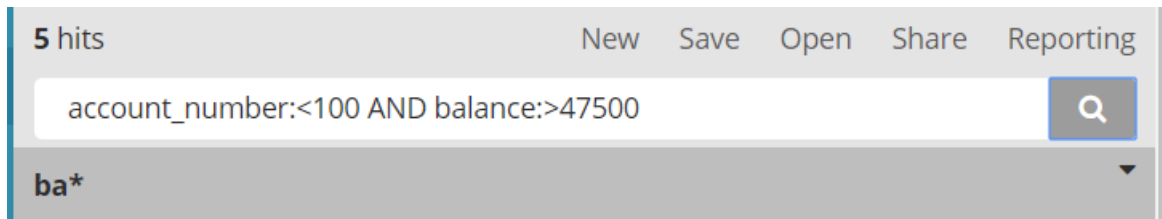
**logstash = log\*,**

**bank = ba\*로**

지정하였음.

이제, Data를 검색하는 방법을 살펴보면 Kibana의 Discovery를 이용한다.





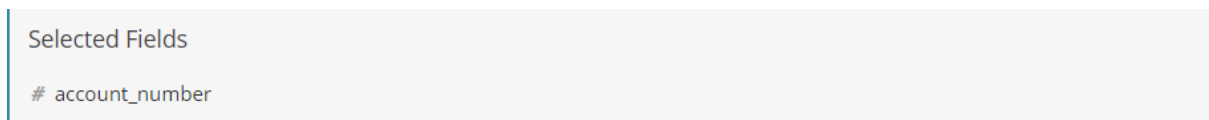
이와 같이 먼저 index pattern 을 지정을 해주어 어떠한 index에서 검색을 할지를 결정을 해 준 후, 질의(query)를 적는다.

이에 대한 결과는

_source	
▶	<pre>account_number: 32 balance: 48,086 firstname: Dillard lastname: Mcpherson age: 34 gender: F address: 702 Quentin Street employer: Quailcom email: dillardmcpherson@quailcom.com city: Veguita state: IN _id: 32 _type: account _index: bank _score: 2</pre>

source에서 볼 수 있다.

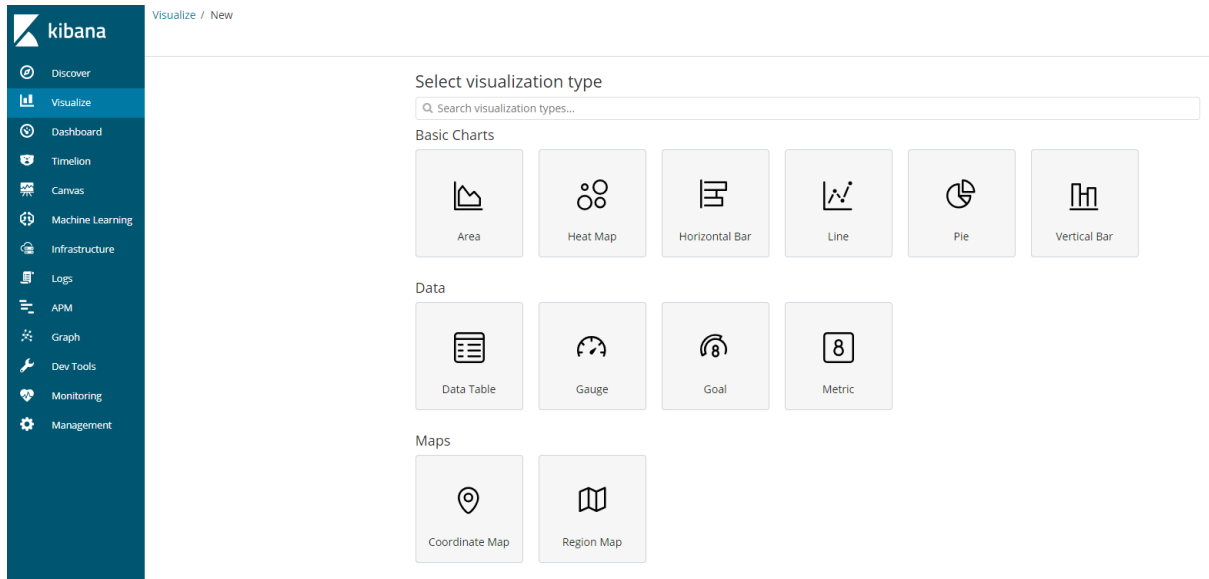
이때, 필요한 Field값만을 출력하기를 원한다면



Selected Fields 부분에 자신이 원하는 Field를 추가한다.

이제 데이터를 시각화 하는 방법이다.

먼저 Visualize 메뉴를 선택한다. 이후 자신이 원하는, 차트를 선택하여 create 시킨다.



## 1. PIE chart ( 사용한 index : ba\* )

**buckets**

☒ Split Slices ⌵

**Aggregation**

Range ⌵

**Field**

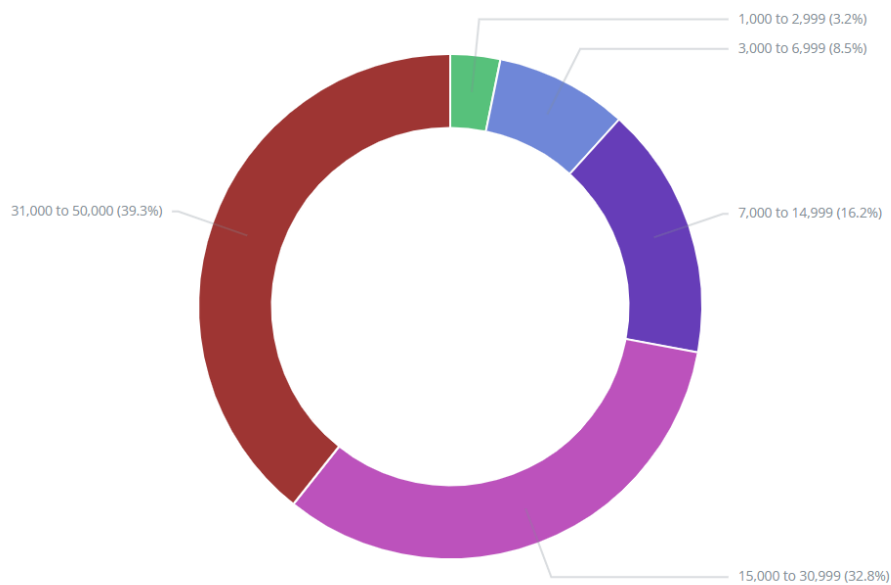
balance ⌵

From	To	
0	999	✕
1000	1999	✕
2000	6999	✕
7000	14999	✕
15000	30999	✕
31000	50000	✕

Add Range

*Custom Label*

이와 같이 어떠한 집합으로 Pie를 나눌지를 선택하고, Field를 이용하여 그 집합이 속하는 Field를 선택한다. 이후, Range를 정하고 마지막으로 화살표를 눌러 update를 누르게 되면



이와 같이 Pie가 나누어지는 것을 확인 가능하다.

Split Slices

Sub Aggregation

Terms

Field

age

Order By

metric: Count

Order

Descending

Size

5

Custom Label

Advanced

새로운 조건을 추가하고 싶다면, "Add sub-buckets"를 누른 후 이와 같은 새로운 조건을 생성해 준다.

결과값을 보면



마찬가지로 X축 역시

▼ X-Axis

Aggregation

Terms ▼

Field

play\_name ▼

Order By

metric: Speaking Parts ▼

Order

Ascending ▼

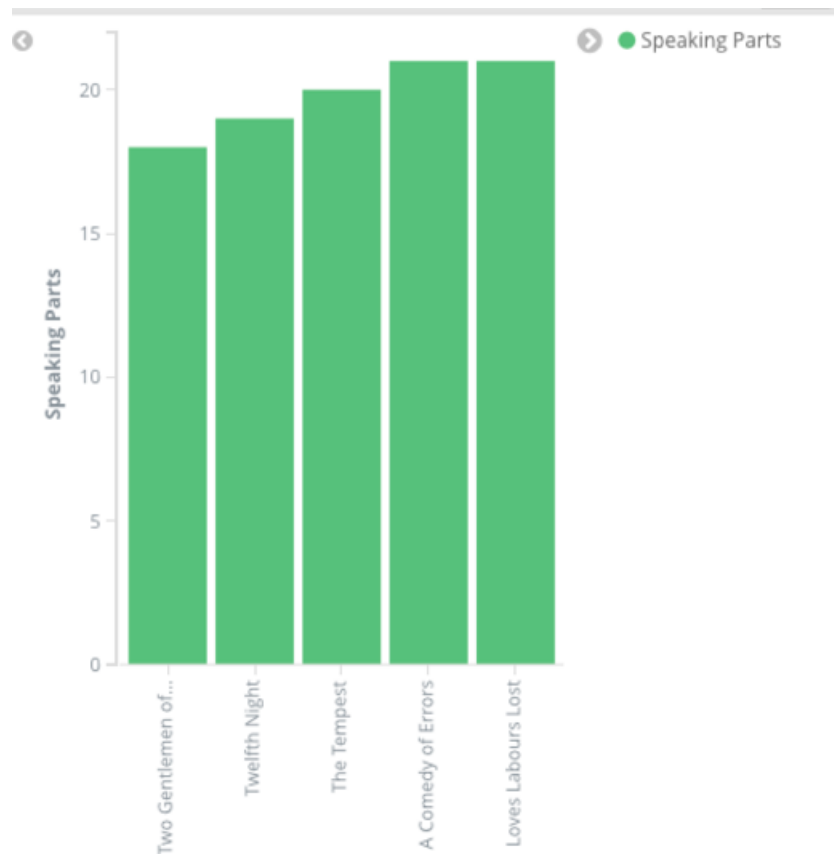
Size

5

Custom Label

Play Name

위와 같이 X축을 지정을 해준다.



위에 대한 결과값으로 Y축은 Speaking Parts, X축은 Play\_Name 으로 나누어진 Chart를 얻을 수 있다.

이때, Shakespeare index를 mapping을 할때, play\_name에 not analyzed 를 사용하여 X축의 PlayName 이 각 단어로 나누어지지 않고, 한 문장을 기본 Unit으로 하여 나누어 지지는 것 확인 할 수 있다.

이때, 같은 Chart에 두가지의 Bar 형태를 담고 싶다면 option Tab 에서 Bar 모드를 grouped로 설정을 해주고 , Y-axis aggregation 을 추가해준다.

Y-Axis

Aggregation

Max

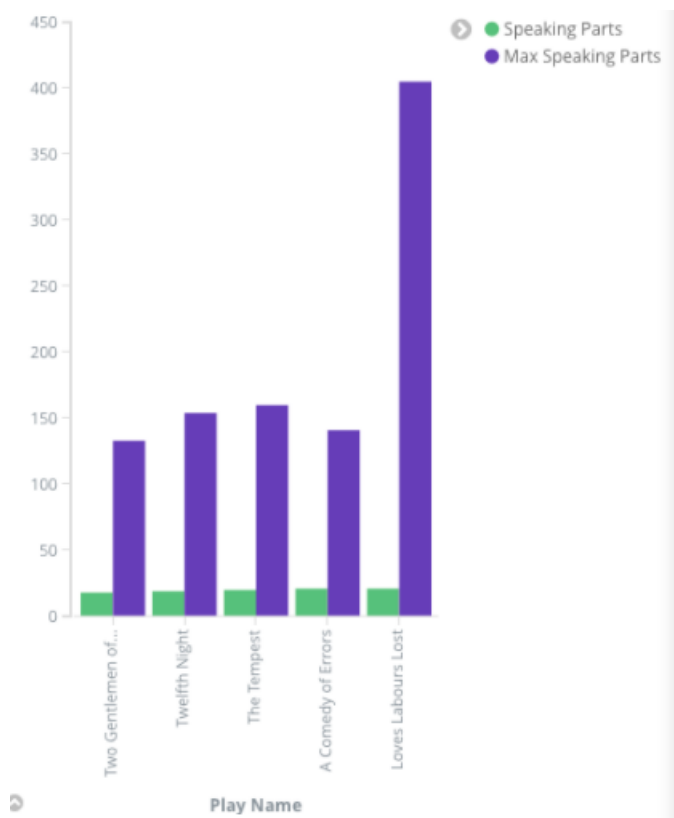
Field

speech number

Custom Label

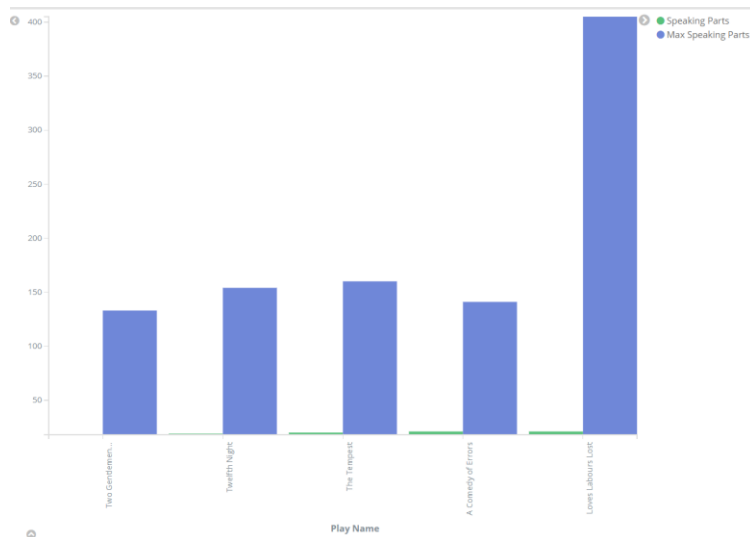
Max Speaking Parts

이와 같이 새로운 Y축을 추가를 해주었을 경우에는



이와 같은 결과 값을 얻을 수 있다.

이때, 두번째 Chart값으로 인하여 첫번째 Data Chart의 차이를 느끼기 어려워 졌는데, 첫번째 chart의 차이가 더 크게 보이게 하기 위하여 "Option" 메뉴의 "Scale Y-Axis to data bounds" 를 check 하여



이와 같이 결과값의 차이가 커 보이는 효과를 넣어줄 수 있다.

### 3. Coordinate map ( 사용한 index : log\* )

Tile map 을 Create를 누른 후, kibana toolbar에 있는 시간(time)을 선택하여 준다. 이후 Absolute를 선택한 후, 원하는 날짜를 선택한다.

**Time Range**

Quick Relative Absolute Recent

**From** Set To Now **To** Set To Now

2015-05-18 00:00:00.000 2015-05-20 23:59:59.999


YYYY-MM-DD HH:mm:ss.SSS YYYY-MM-DD HH:mm:ss.SSS

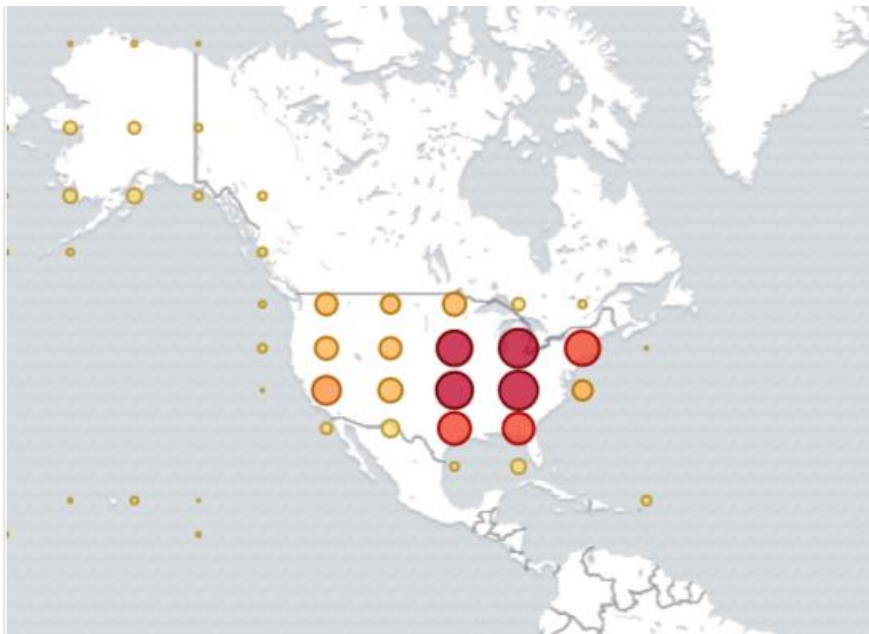
< May 2015 > < May 2015 >

Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat
					01	02						01	02
03	04	05	06	07	08	09	03	04	05	06	07	08	09
10	11	12	13	14	15	16	10	11	12	13	14	15	16
17	18	19	20	21	22	23	17	18	19	20	21	22	23
24	25	26	27	28	29	30	24	25	26	27	28	29	30
31							31						

Go

그리고 "Go" 버튼을 누른다.

이후  버튼을 눌러 갱신을 시켜 주면,



지도에 이와 같이 Circle값이 출력이 되는 것을 확인 할 수 있다.





3가지의 버튼이 존재한다.

첫 번째 버튼인 +, - 는 zoom in, out 을 의미.

두 번째 버튼은 Fit Data Bounds 버튼으로서 모든 점들 중 가장 낮은 레벨로 화면을 zoom시켜주는 역할을 실시.

마지막 세 번째 버튼은 Latitude/Longitude Filter버튼으로서 사각형을 그려 범위를 잘라주는 역할.

마지막으로 Mark down 을 생성한 후 빈 칸에

#### Markdown

```
# This is a tutorial dashboard!
The Markdown widget uses **markdown** syntax.
> Blockquotes in Markdown use the > character.
```

이와 같이 입력을 한 후 update버튼을 누르면

# This is a tutorial dashboard!

The Markdown widget uses **markdown** syntax.

Blockquotes in Markdown use the > character.

이와 같은 결과를 출력 할 수 있다. 이를 저장하고 나면

마지막으로 DashBoard 메뉴에서 지금까지 저장한 visualization chart들을 모두 출력할 수 있다.

## Add Panels

Visualization

Saved Search

Visualizations Filter...

4 of 4

Manage Visualizations

Name ▲

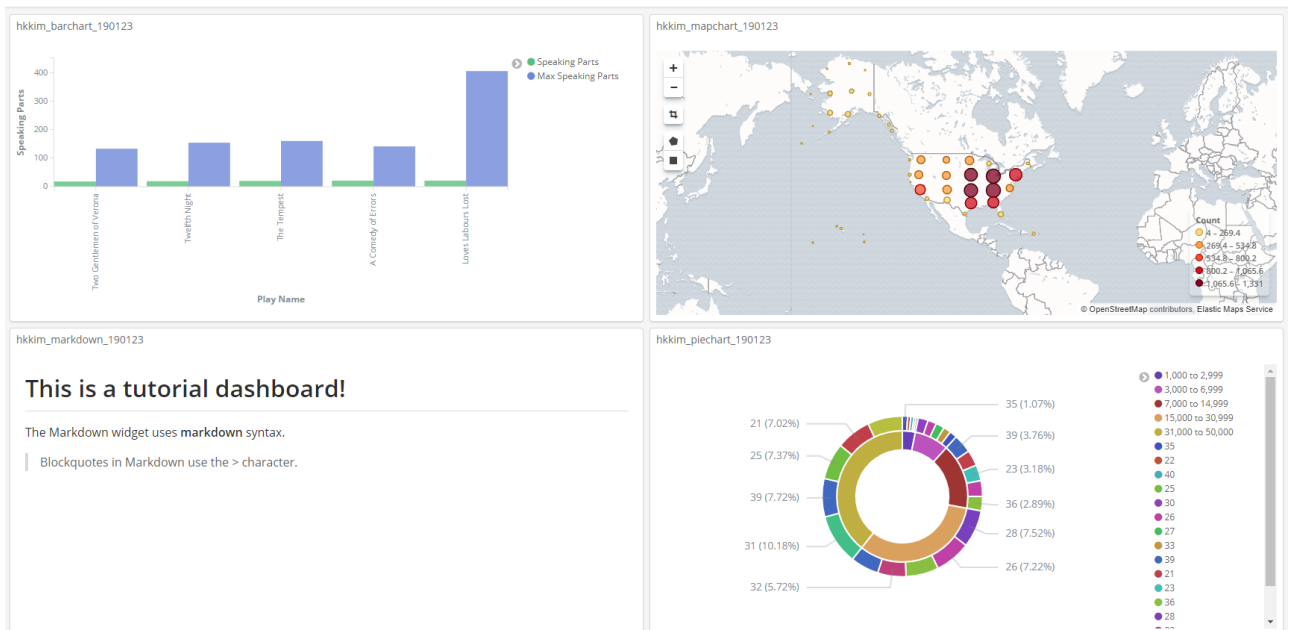
Map Example

Markdown Example

Pie Example

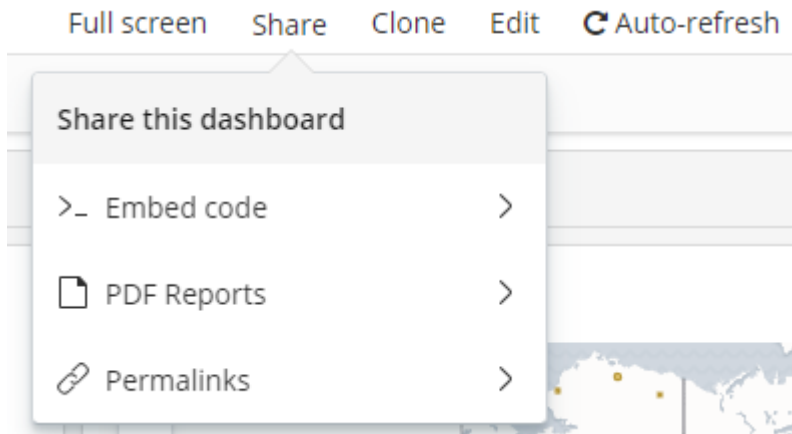
Vertical Bar Chart Example

Add Panel 을 이용하여 지금까지 저장한 모든 Example들을 클릭하면



이와 같이 지금까지 저장한 값들을 한 화면에 출력이 가능하다.

Link를 얻거나, HTML code를 얻기 위해서는 dashboard 를 저장 후, share를 이용합니다.



마지막으로 정리를 하면

- ★ Filtering을 통해서 원하는 정보를 Discover가 가능하다.
- ★ 시각화된 정보를 kibana를 이용하여 얻는다.
- ★ Kibana의 Management 메뉴를 이용하여 object값들을 관리하는 것이 가능하다.
- ★ RESTful 한 request를 이용하여 정보를 Elasticsearch cluster와 연동이 가능하다.

(RESTful 은 GET, PUT, POST, Delete로 명령어가 이루어 지는 것을 의미한다.)

## 참고문헌.

1. <https://www.elastic.co/guide/en/kibana/current/getting-started.html> , Elastic Tutorial