http://bit.ly/es_korea_edu

Elasticsearch & Elastic Stack

김종민 – Community Engineer @Elastic
jongmin.kim@elastic.co
2018. 7. 25 @고려대학교

김종민

2015. 6. ~

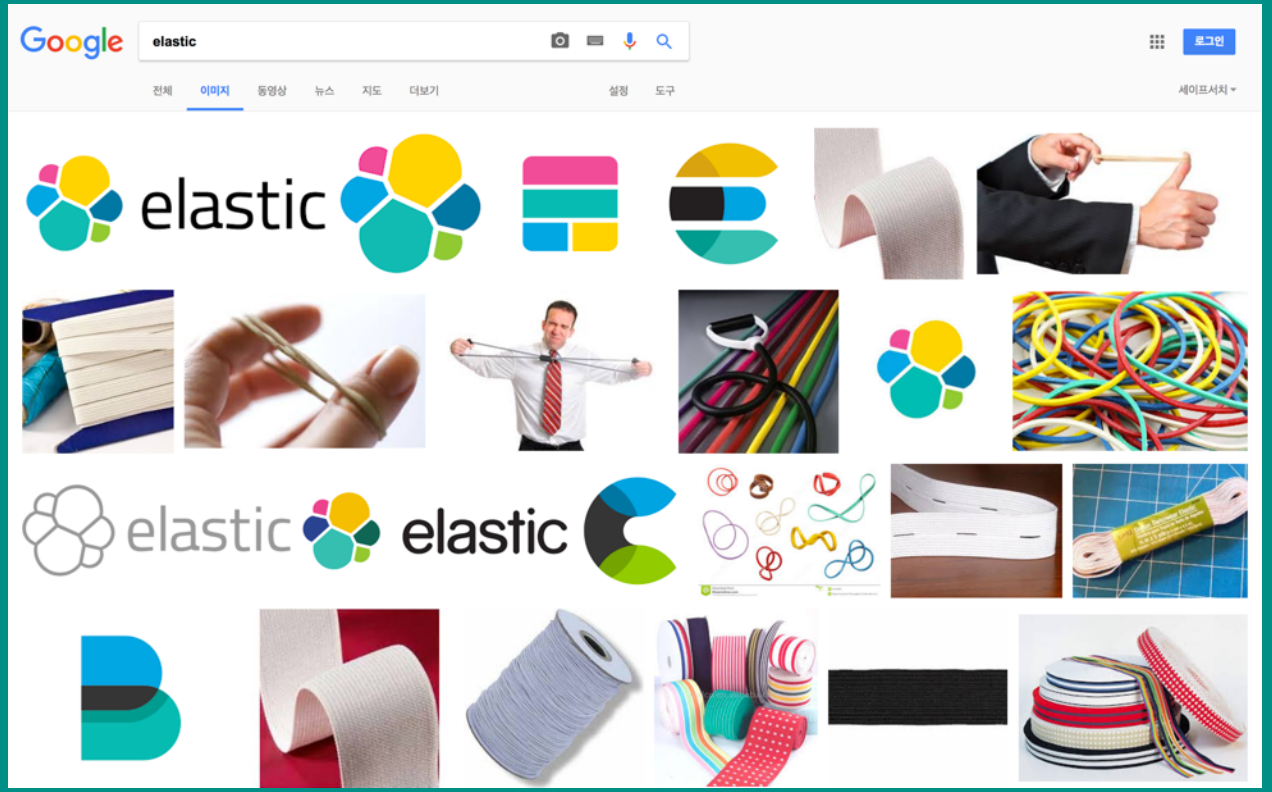
Community Engineer @Elastic

elastic


Korea Elasticsearch User Group
김종민
홈
Korea Elasticsearch User Group
공개 그룹
정보
토론
멤버
이벤트
동영상
사진
파일
그룹 인사이트
그룹 관리
이 그룹 검색
바로가기
작목실(작은목표를실행하는...
D2 COMMUNITY 운영진 ...
Golang Korea
14
Plugit
Korea Elasticsearch Use...
더 보기
Elasticsearch
Beats
Logstash
가입함
알림
공유하기
더 보기
게시물 작성
사진/동영상 추가
라이브 방송
더 보기
글쓰기...
사진/동영상
함께 시청하기
기분/활동
최근 활동
김종민님이 게시물을 공유했습니다.
관리자 · 1시간
지난주 밋업 발표자료 및 녹화 영상 입니다.
김종민님이 링크를 공유했습니다.
1시간
발표 자료 링크 올려드립니다. Roll Up 관련 내용은 제가 만든것이 아니라서 뺐습니다.
멤버 추가
초대 퍼가기
+ 이름이나 이메일 주소를 입력하세요...
멤버
멤버 4,826명
이번 주에 새로운 멤버가 67명 있습니다. 새 멤버를 환영하는 게시물을 작성해보세요.
게시물 작성
추천 멤버
숨기기
친구
정찬용
멤버 추가
정연용
멤버 추가
Youngjo Kim
멤버 추가
더 보기
채팅(64)

stic

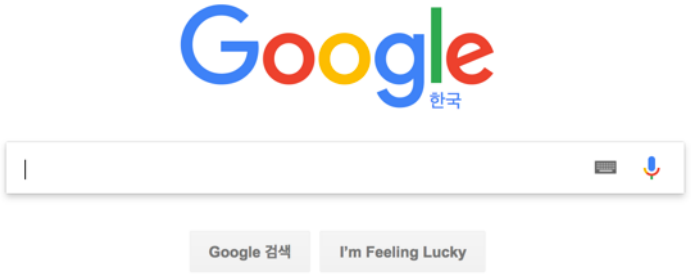
Elastic?

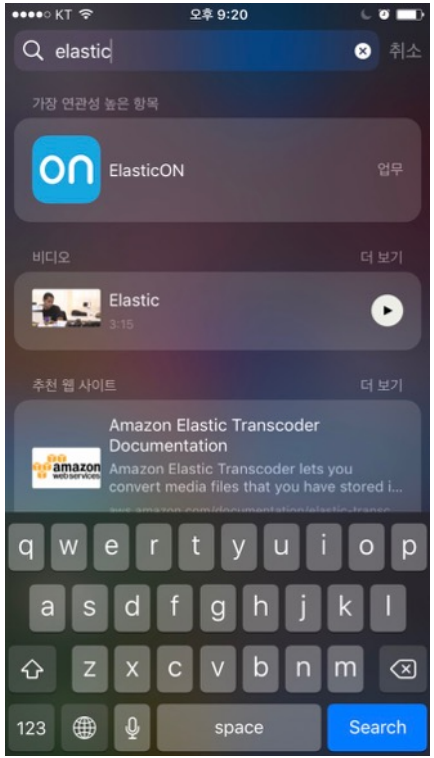


elastic

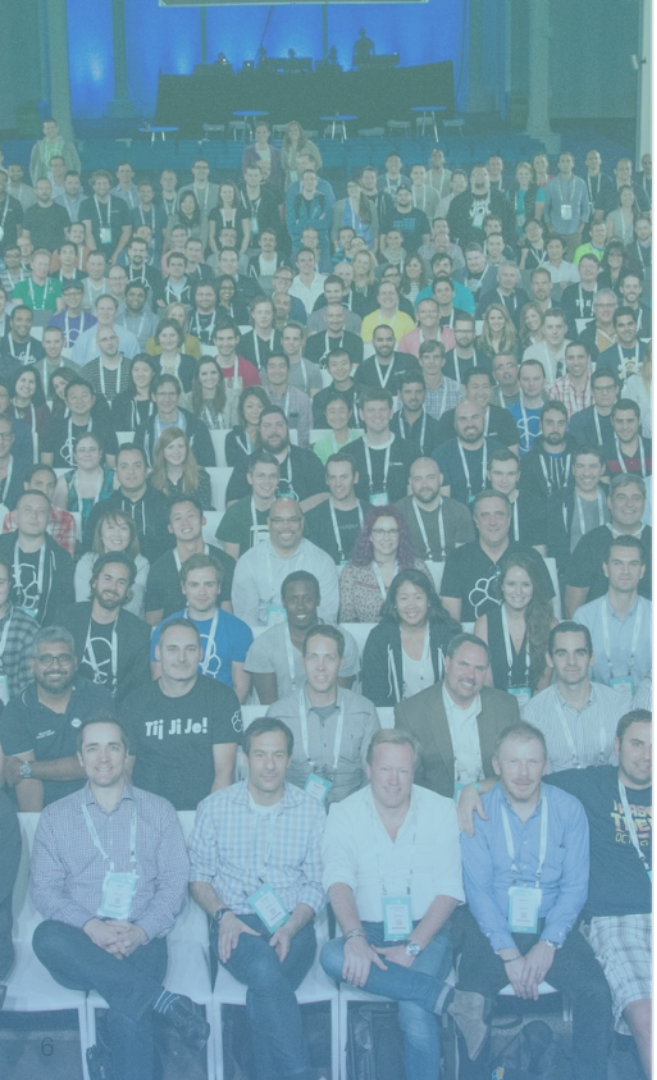
Elastic ?

Elasticsearch 라는 오픈소스 검색엔진을 개발한 회사.



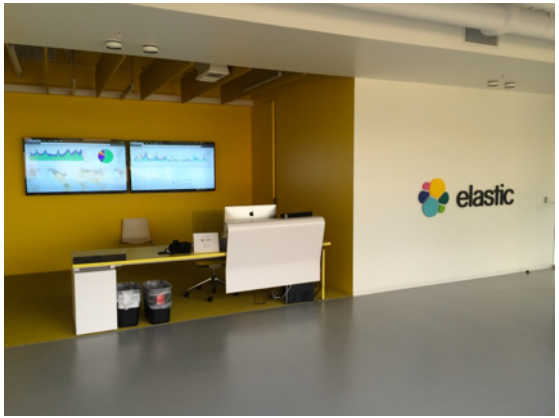


elastic



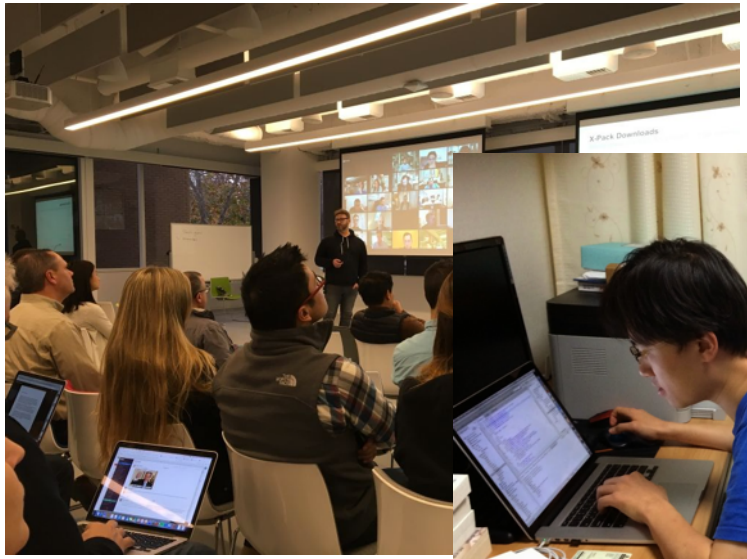
현재 직원은 850명 정도. 한국은 10명.

본사는 네델란드 암스텔담과
캘리포니아 마운틴 뷰에 있습니다.



Remote Working

Elastic에는 40여개 국에 직원들이 있습니다. 매년 1~2 차례 모입니다.



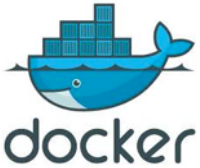
elastic

Tech	Adobe	hp	salesforce	Microsoft	CISCO	DELL
Finance	SwissLife	Scottrade	Goldman Sachs	HSBC	E*TRADE	ZURICH
Telco	verizon	COX COMMUNICATIONS	orange	ROGERS	Alcatel·Lucent	Sprint
Consumer	Audi	UBER	GROUPON	Columbia Sportswear Company	DHL	ebay

Enterprise Customers in Every Industry

elastic

Massive Startup Adoption

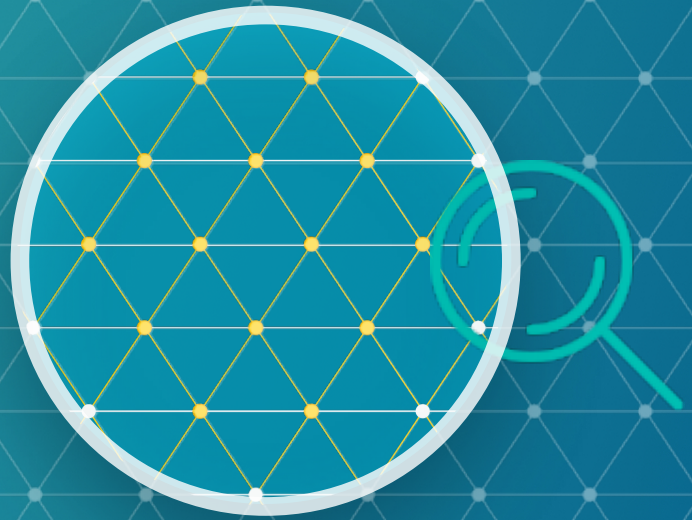


Elastic Subscription Customers in Korea

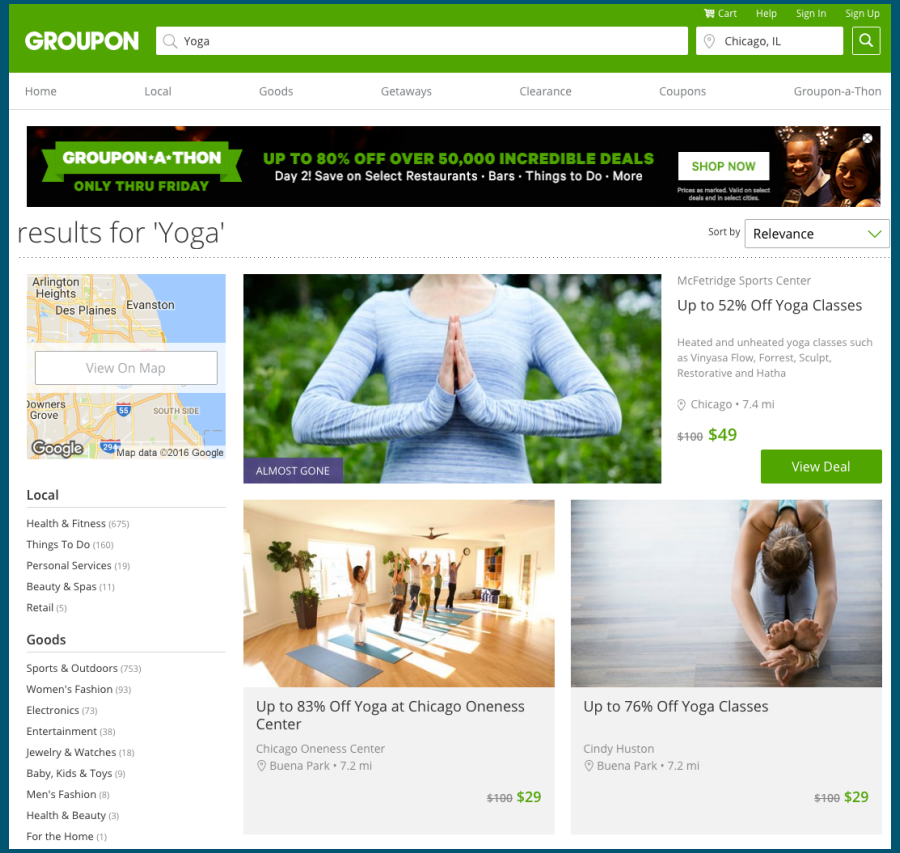
elastic

Search and analytics, it all started here

More than 60% of our customers have a search or analytics use case

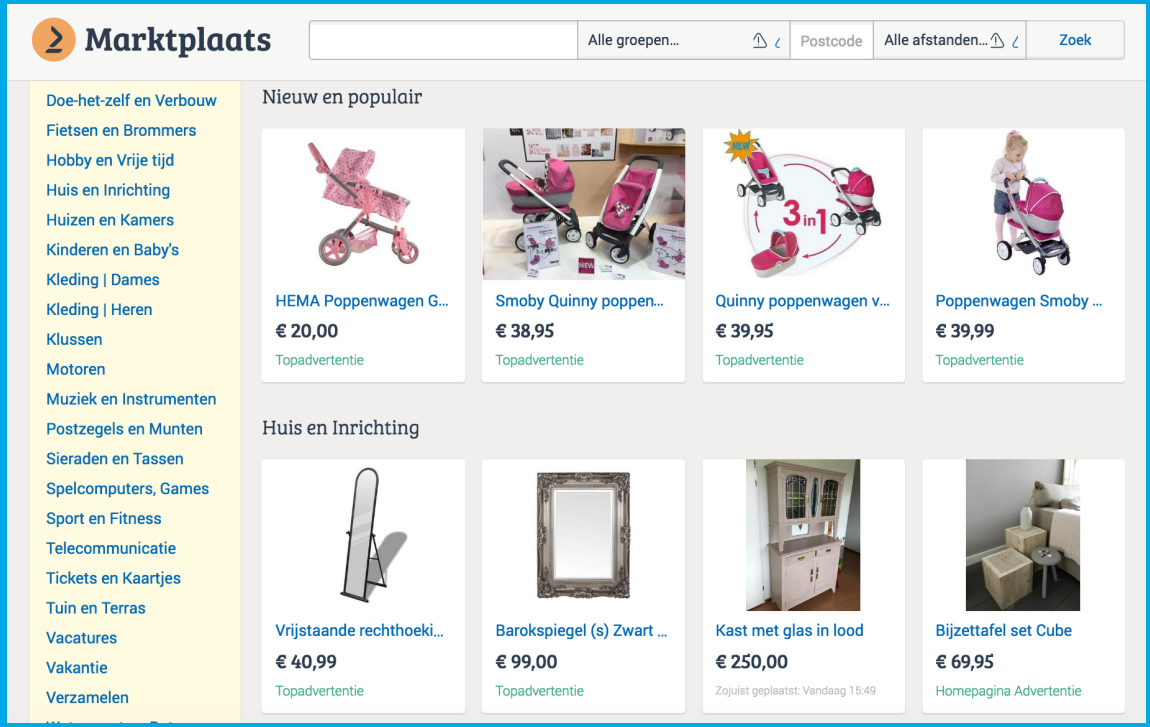


GROUPON

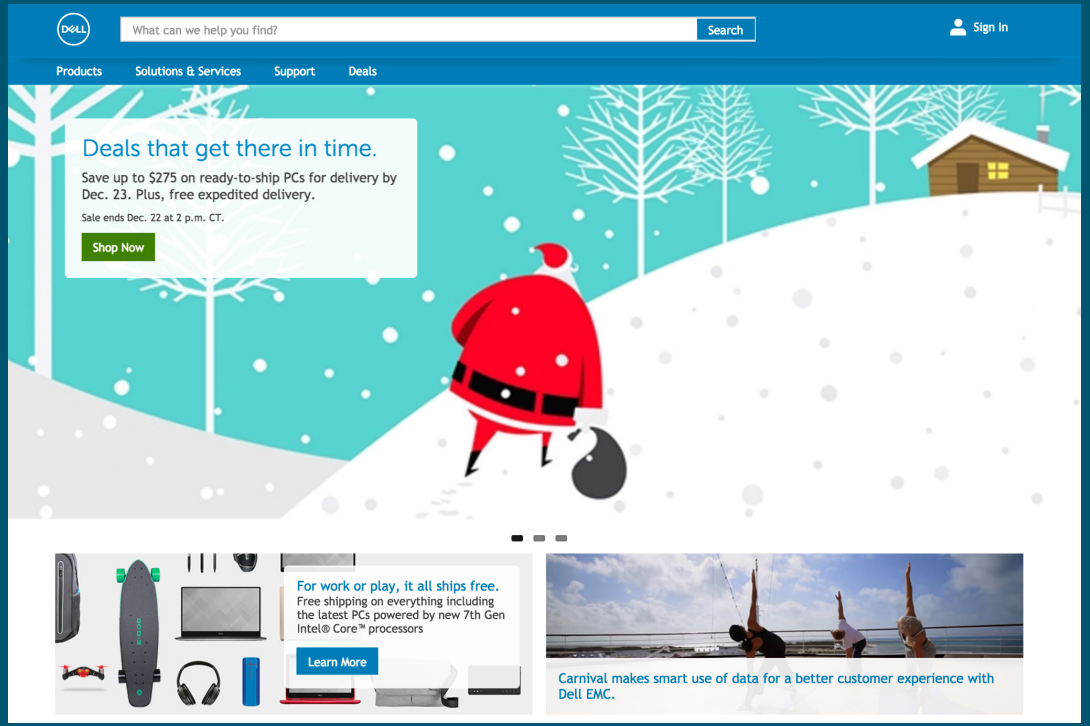


elastic

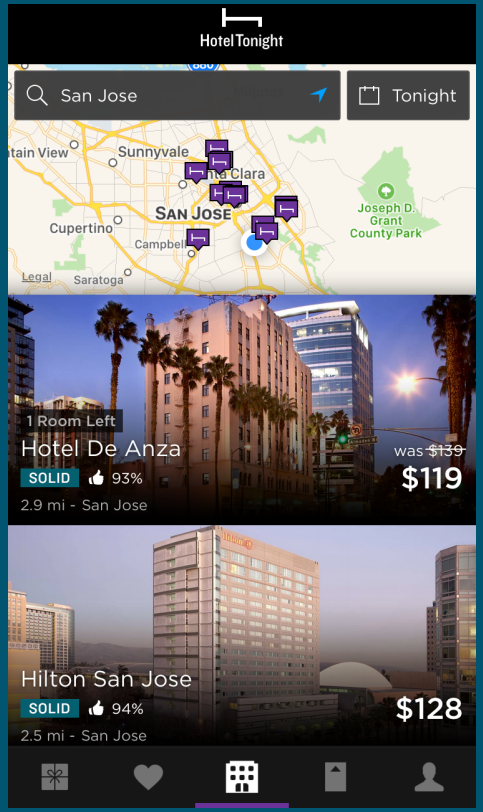
ebay

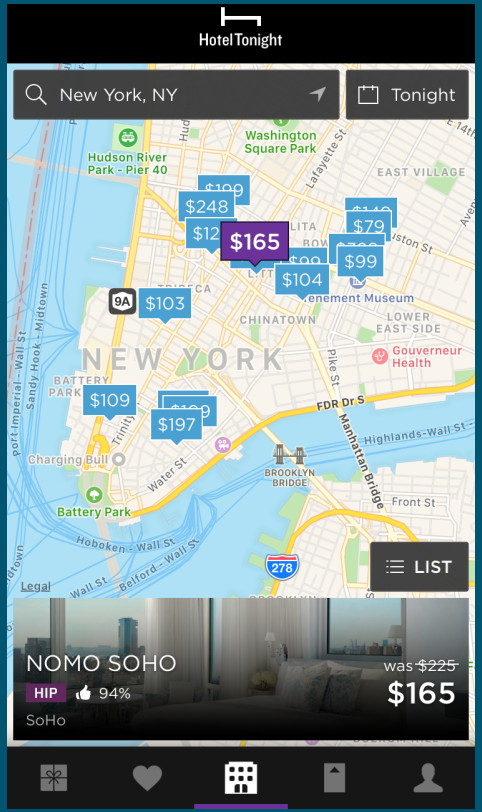


elastic


What can we help you find?
Search
Sign In
Products
Solutions & Services
Support
Deals
Deals that get there in time.
Save up to $275 on ready-to-ship PCs for delivery by Dec. 23. Plus, free expedited delivery.
Sale ends Dec. 22 at 2 p.m. CT.
Shop Now
For work or play, it all ships free.
Free shipping on everything including the latest PCs powered by new 7th Gen Intel® Core™ processors
Learn More
Carnival makes smart use of data for a better customer experience with Dell EMC.

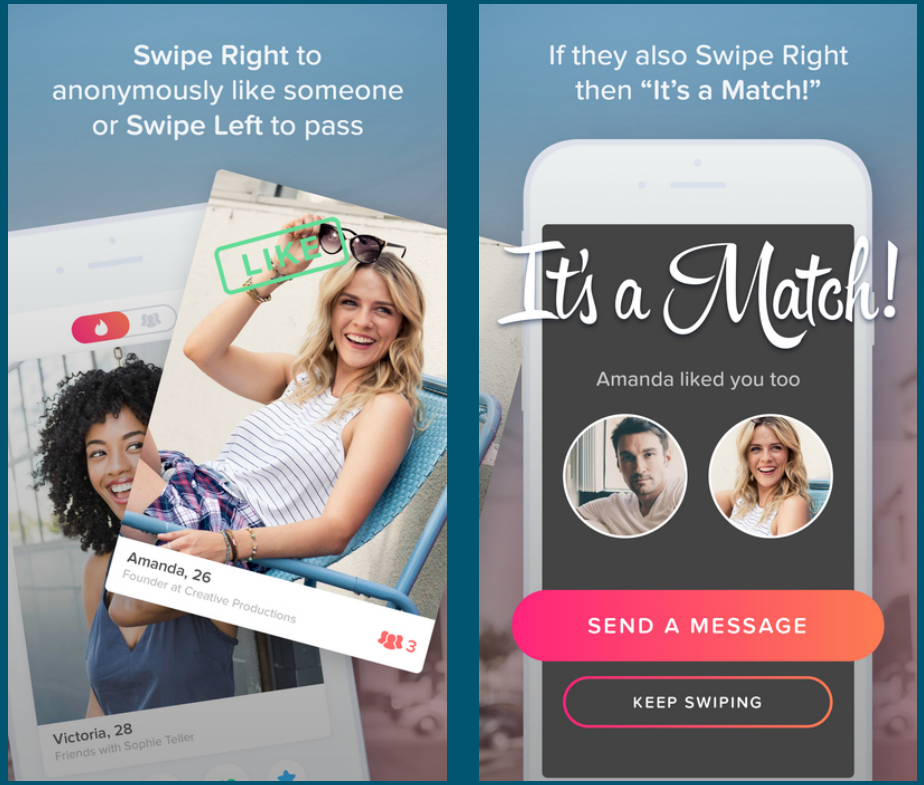
Hotel Tonight





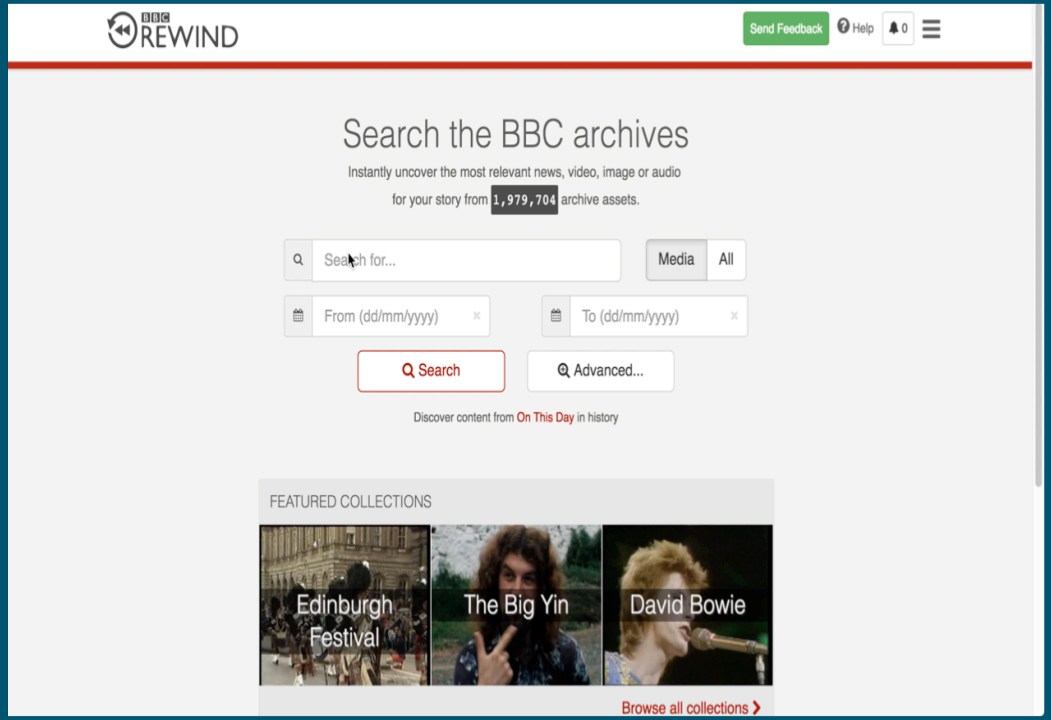
elastic

tinder™



elastic

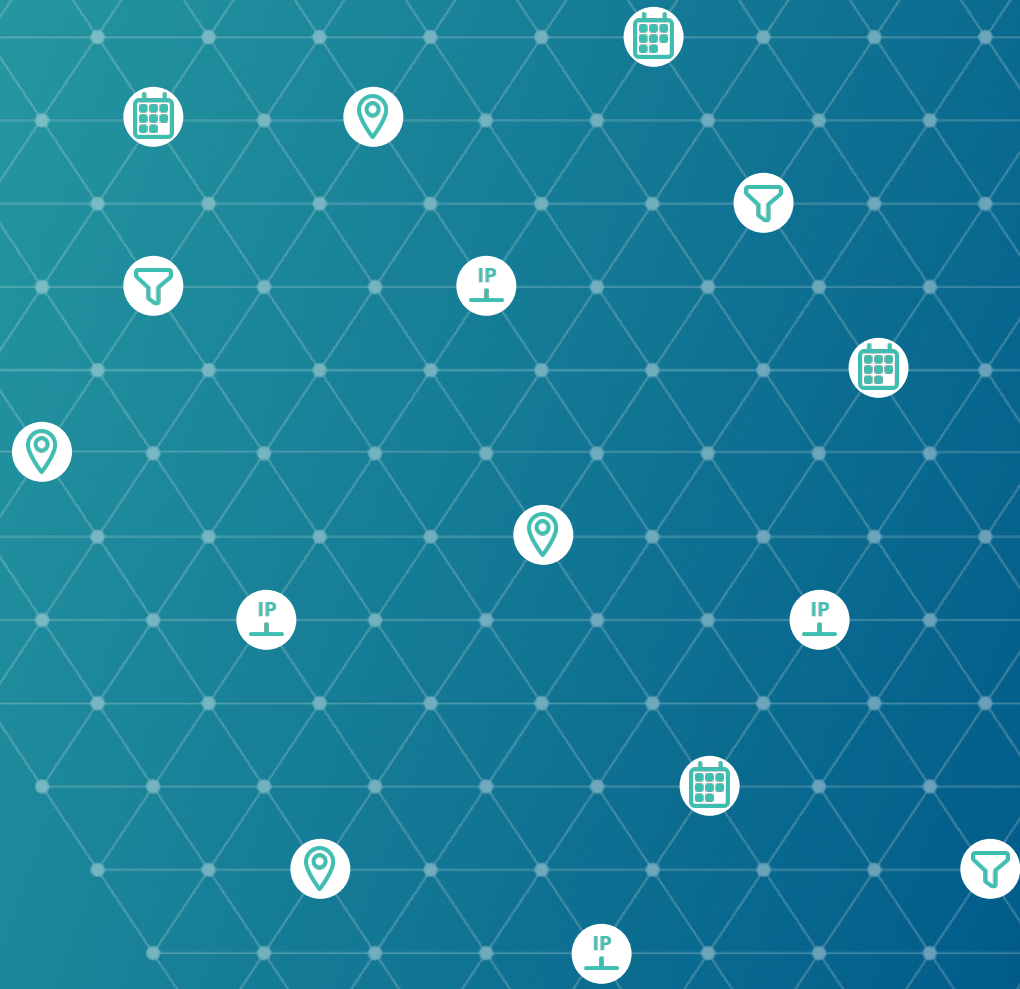
BBC


BBC REWIND
Send Feedback
Help
0
Search the BBC archives
Instantly uncover the most relevant news, video, image or audio
for your story from 1,979,704 archive assets.
Search for...
Media
All
From (dd/mm/yyyy)
To (dd/mm/yyyy)
Search
Advanced...
Discover content from On This Day in history
FEATURED COLLECTIONS
Edinburgh Festival
The Big Yin
David Bowie
Browse all collections

elastic

Logs Logs Logs, many devices, many systems

More than 40% of our customers use our products for operational log analysis



verizon

We collect more than
1.2 TB logs every day from our infrastructure, web servers, and applications.

elastic

We handle more than
3 billion daily events while meeting all of our data security requirements.

elastic

We centralized our common infrastructure logging from 900 servers and 5000 network devices in order to decrease MTTR and wasted labor and capital expenses.

elastic

We instantly decreased our network latency detection from 48 hours to near-real time and shortened our time-to-market for new products and features.

elastic

We get instant analysis of over 20 billion game play, download, and server events per day to ensure optimum player experience and satisfaction.

elastic

The Elastic Stack helped us improve customer satisfaction and retention by allowing us to proactively diagnose and fix website and application errors.

elastic

Sniff sniff sniff, find the bad actors in your data

200% YoY growth in security use cases with our products

“

We mine and analyze
4 billion events every day to detect security hacks and threats.

elastic

We analyze piles of data:

13B AMP queries/day

600B emails/day

16B web requests/day

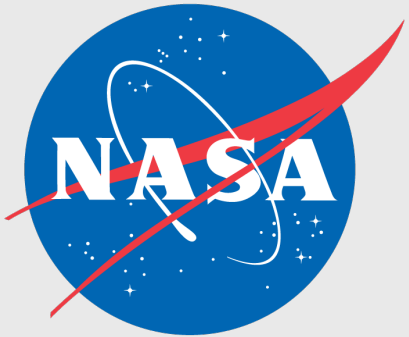
elastic

The Elastic Stack made it possible for us to build Fusion – our centralized cyber security and defense platform – and protect the bank and our customers from real-time threats all over the world.

75% of our customers use our products for multiple use cases

1,000+ developers use the Elastic Stack for use cases from trade tracking to creating new HR and compliance apps.

elastic



“

We send from Mars more than

30K messages

100K documents

4x a day for

operational, telemetry, anomaly resolution, and log analysis.

elastic

DB-Engines Ranking of Search Engines

https://db-engines.com/en/ranking/search+engine

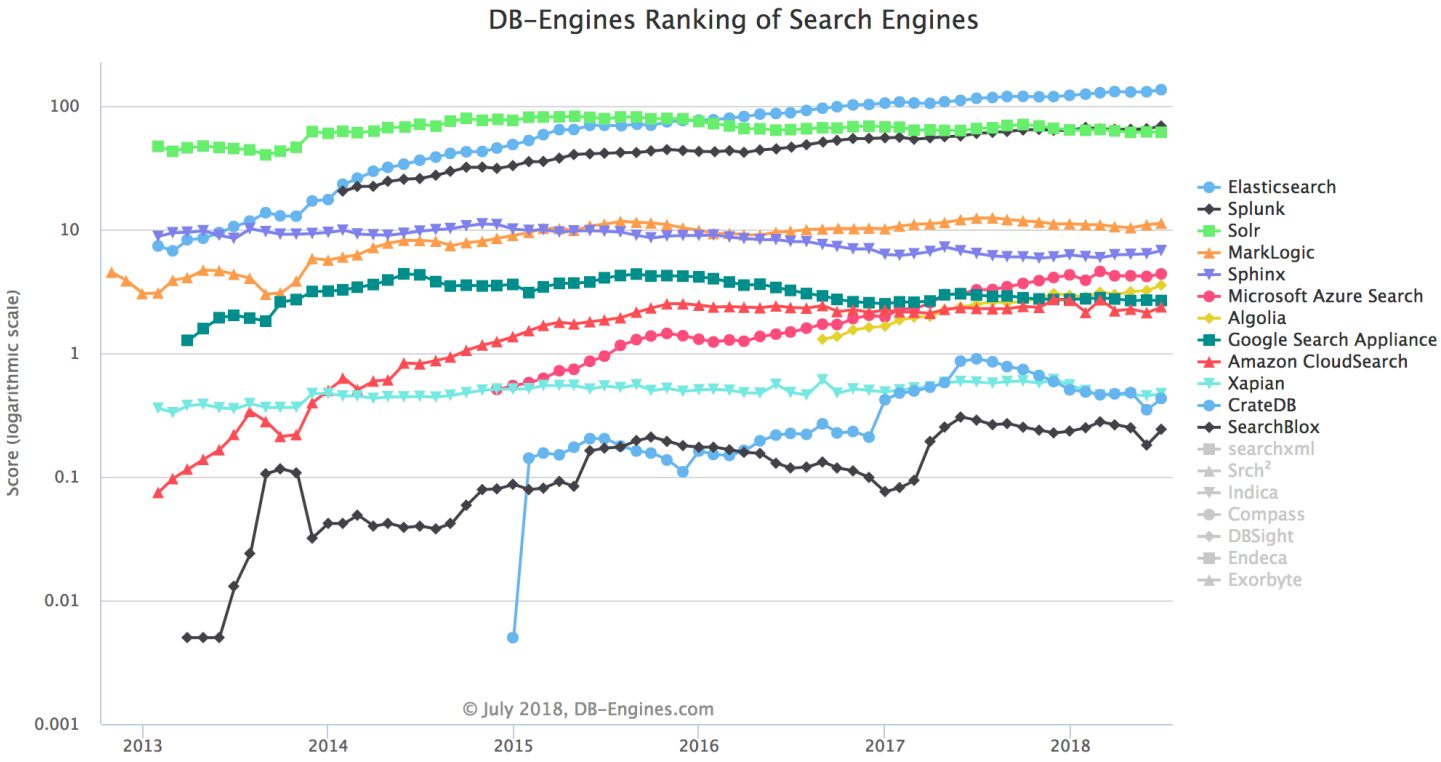
17 systems in ranking, June 2018

Rank			DBMS	Database Model	Score		
Jun 2018	May 2018	Jun 2017			Jun 2018	May 2018	Jun 2017
1.	1.	1.	Elasticsearch	Search engine	131.04	+0.60	+19.48
2.	2.	↑ 3.	Splunk	Search engine	65.78	+0.68	+8.26
3.	3.	↓ 2.	Solr	Search engine	62.06	+0.55	-1.55
4.	4.	4.	MarkLogic	Multi-model	10.97	+0.58	-1.16
5.	5.	5.	Sphinx	Search engine	6.38	+0.06	-0.39
6.	6.	6.	Microsoft Azure Search	Search engine	4.19	-0.05	+1.14
7.	7.	↑ 8.	Algolia	Search engine	3.22	+0.06	+0.84
8.	8.	↓ 7.	Google Search Appliance	Search engine	2.70	+0.02	-0.34
9.	9.	9.	Amazon CloudSearch	Search engine	2.13	-0.15	-0.22
10.	↑ 11.	↑ 11.	Xapian	Search engine	0.45	-0.01	-0.14

Rank			DBMS	Database Model	Score		
Jun 2018	May 2018	Jun 2017					
1.	1.	1.	Oracle				
2.	2.	2.	MySQL				
3.	3.	3.	Microsoft SQL Server				
4.	4.	4.	PostgreSQL				
5.	5.	5.	MongoDB				
6.	6.	6.	DB2	Relational DBMS	185.64	+0.03	-1.86
7.	7.	↑ 9.	Redis	Key-value store	136.30	+0.95	+17.42
8.	↑ 9.	↑ 11.	Elasticsearch	Search engine	131.04	+0.60	+19.48
9.	↓ 8.	↓ 7.	Microsoft Access	Relational DBMS	130.99	-2.12	+4.44
10.	10.	↓ 8.	Cassandra	Wide column store	119.21	+1.38	-4.91

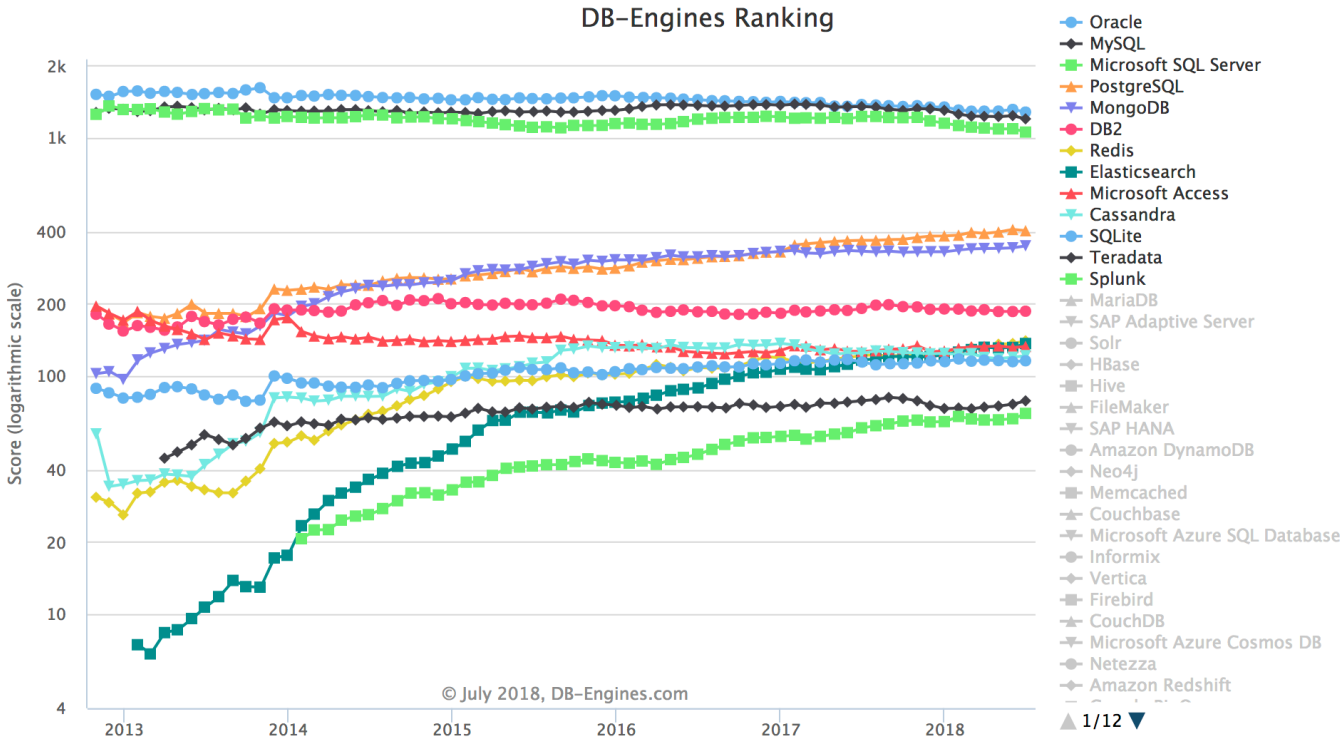
DB-Engines Ranking of Search Engines

https://db-engines.com/en/ranking/search+engine



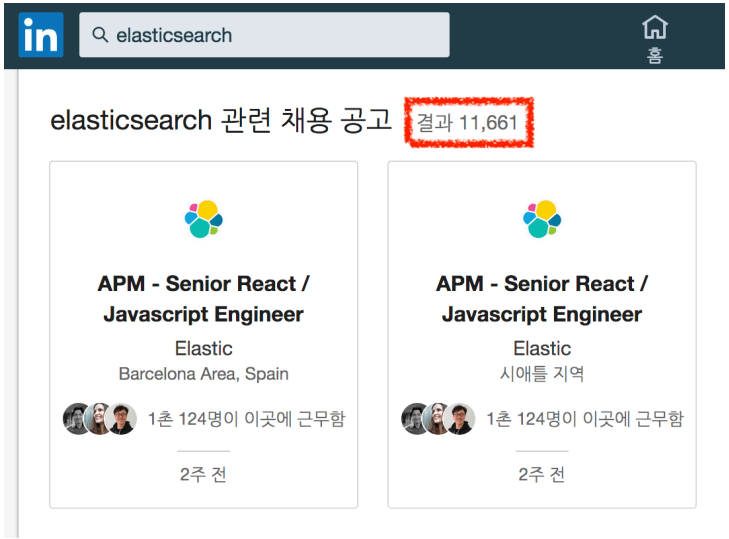
DB-Engines Ranking of Search Engines

https://db-engines.com/en/ranking/search+engine



요즘 elasticsearch engineer 구인도 많습니다

http://www.ciokorea.com/news/37783

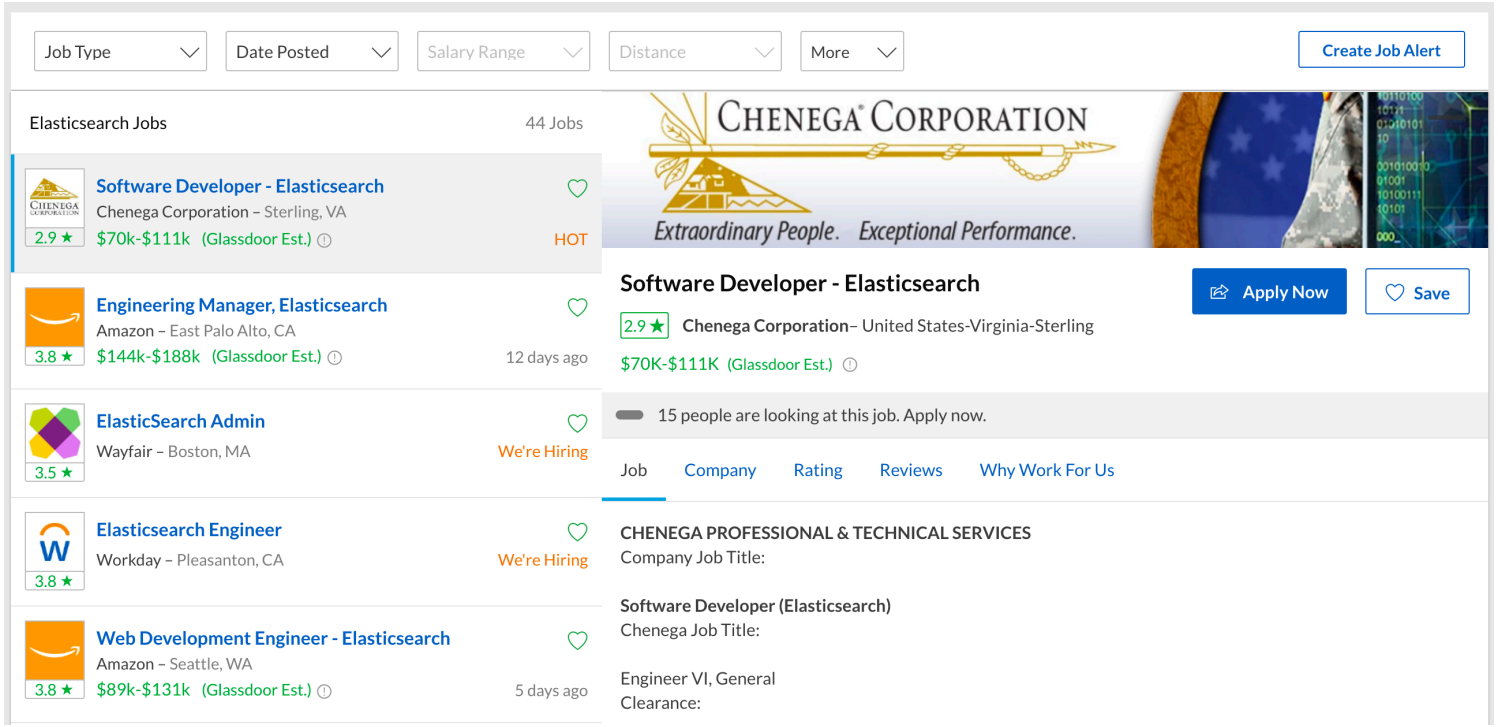


연봉 높은 10대 기술력

elastic

요즘 elasticsearch engineer 구인도 많습니다

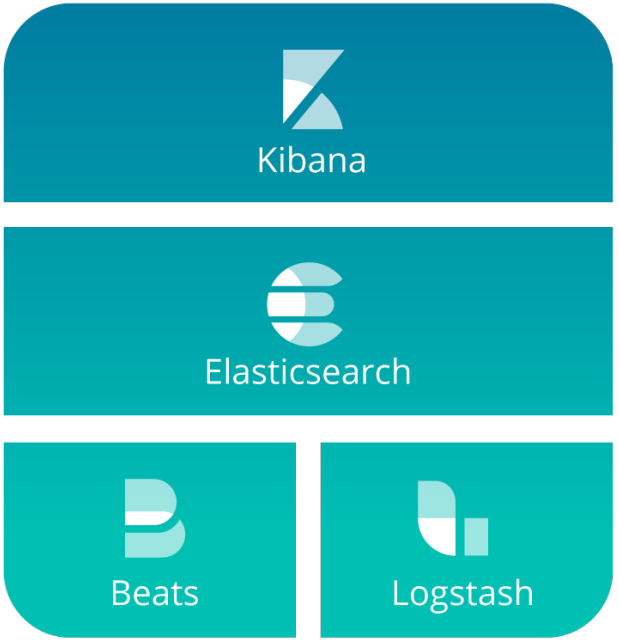
https://www.glassdoor.com



elastic

Elastic Stack

100% open source



elastic

오픈소스로 돈은 어떻게 벌죠?

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

- 기술지원 Subscription
- 유료 사용자 기능
- 교육
- 컨설팅

	무료 사용		GOLD	PLATINUM
	OPEN SOURCE	BASIC		
	무료 다운로드		정보 요청하기	정보 요청하기
기술지원 & 라이센스				
기술지원 범위			Business hours	24/7/365
응답 시간			Critical: 4 hrs L2: 1 day L3: 2 days	Critical: 1 hr L2: 4 hrs L3: 1 day
무제한 기술 문의			✓	✓
기술지원 시스템 계정 제공 수			6	8
원격 시스템을 통한 기술 지원			✓	✓
긴급 패치				✓

elastic

Verizon Wireless 사례

https://www.elastic.co/use-cases/verizon-wireless

- On the whole, the team was able to reduce MTTR from 20-30 minutes to 2-3 minutes on average, a 10x improvement.
  - MTTR : Maintenance time to recover.

elastic

X-Pack

Single install
Extensions for the Elastic Stack
Subscription pricing

Security

Alerting

Monitoring

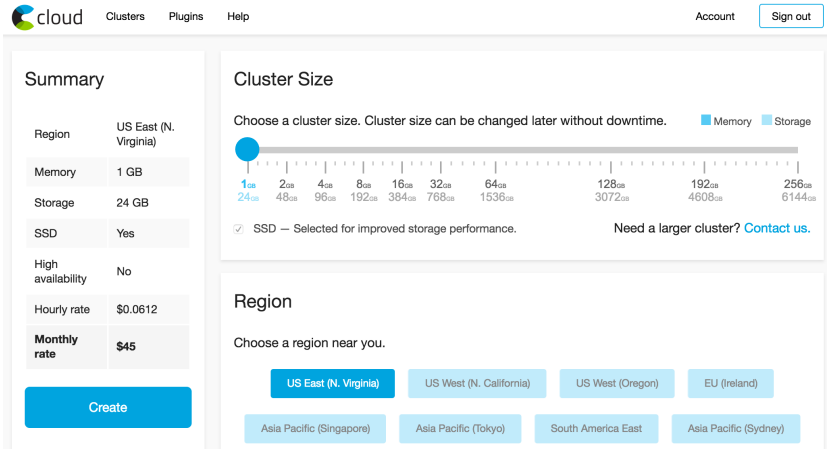
Reporting

Graph

Machine Learning

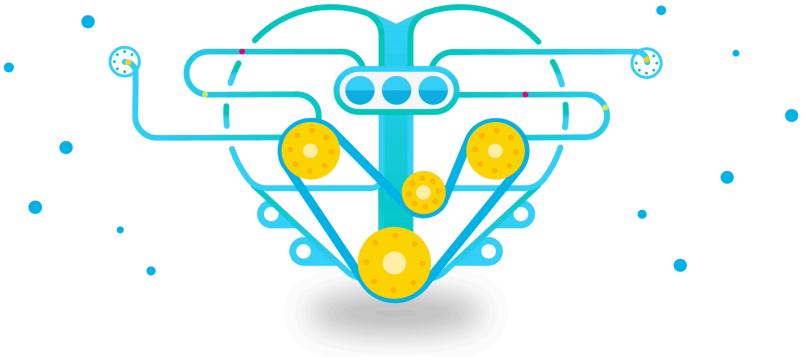
Elastic Cloud

Hosted Elasticsearch & Kibana
Includes X-Pack features
Starts at $45/mo



Available in AWS today
Available in Google Cloud Platform (soon)
Available as a private cloud/on-premise solution
(Elastic Cloud Enterprise)

elastic



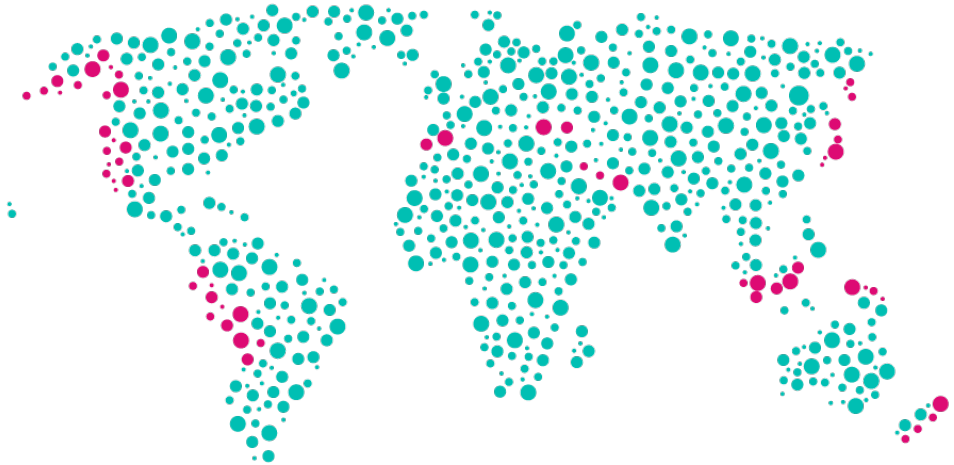
Elasticsearch

Heart of the Elastic Stack

Distributed, Scalable	**High-availability**	**Multi-tenancy**
Developer Friendly	**Real-time, Full-text Search**	**Aggregations**

Kibana

Window into the Elastic Stack



Visualize and analyze	**Geospatial**	**Customize and Share Reports**
Graph Exploration	**UX to secure and manage the Elastic Stack**	**Build Custom Apps**

elastic

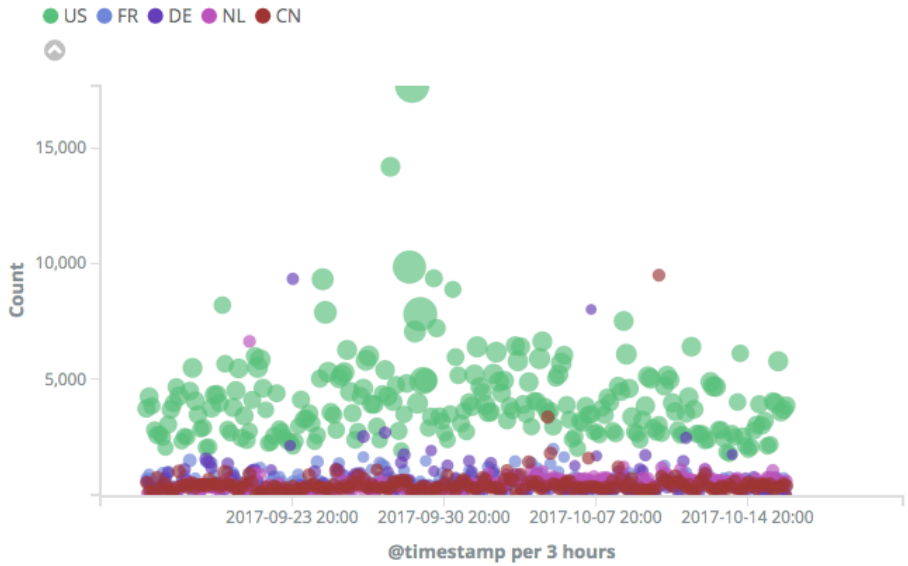
Apache - Total Visitors

2,317,838

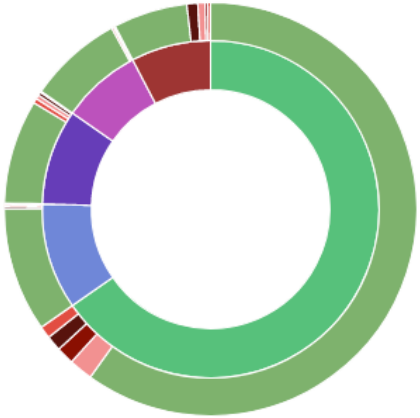
Apache - Unique Visitors

29,740

Apache - Bytes and Count



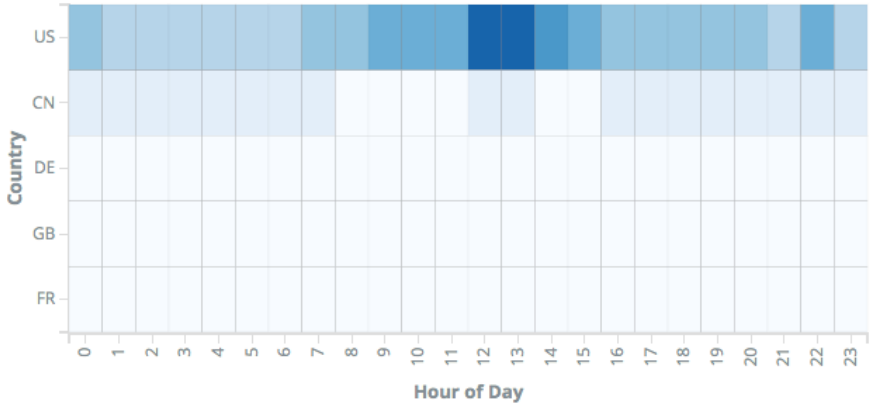
Apache - Country and Status



Apache - Unique Visitor...

City	Count of Unique Clients
Beijing	569
Ashburn	397
Redmond	383
Chicago	379
London	248
Los Angeles	232

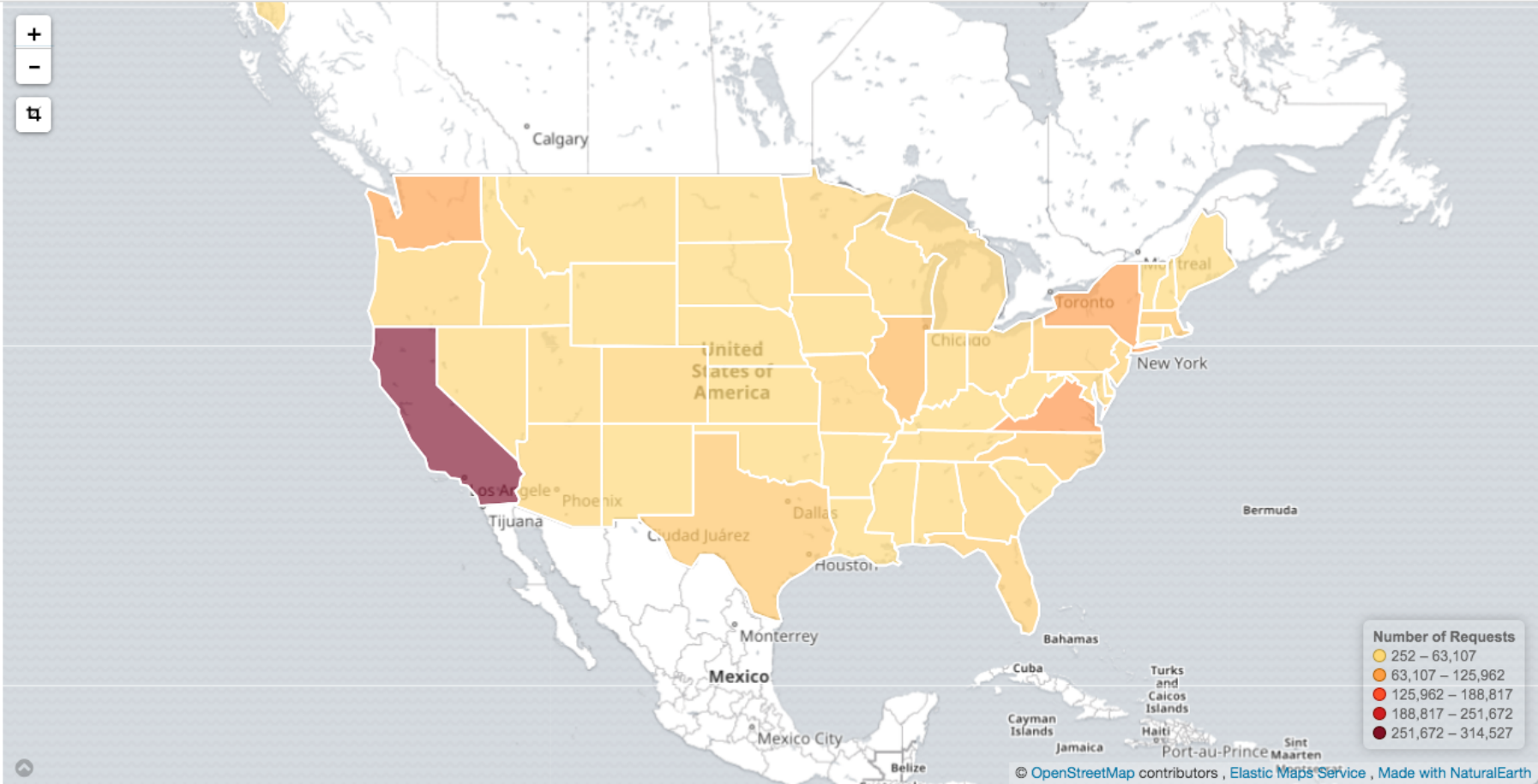
Apache - Country traffic by hour



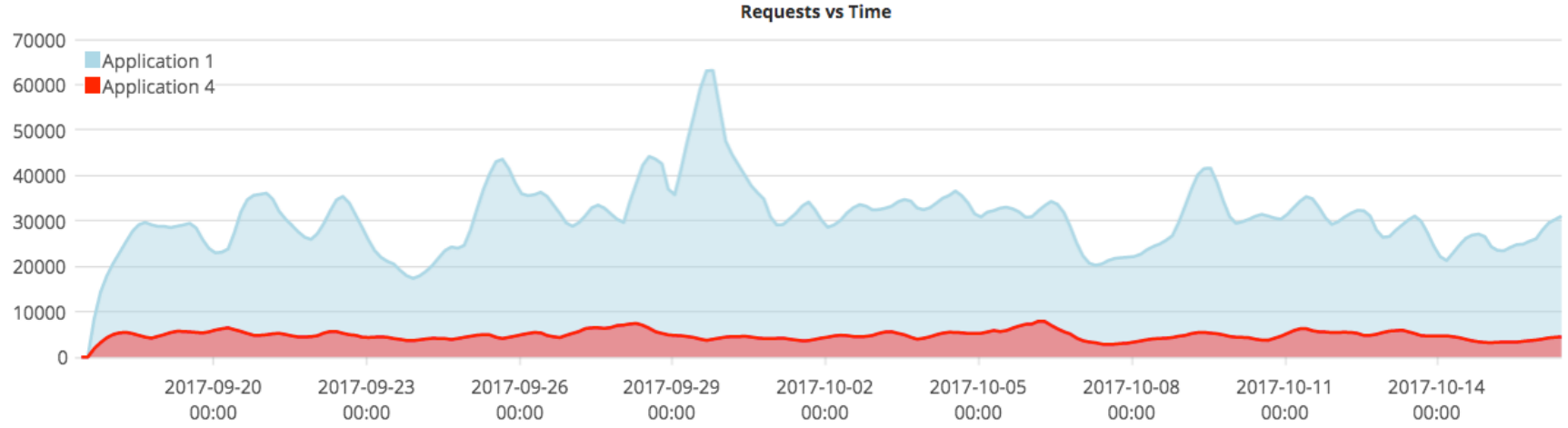
Apache - Top OS small

apache2.access.geoip.country_iso_code:US Uses lucene query syntax

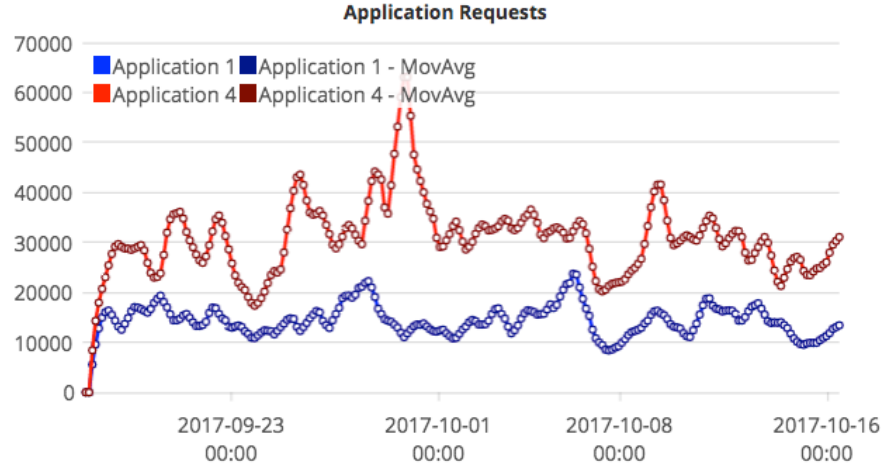
apache2.access.geoip.country_iso_code: "US" Add a filter + Actions



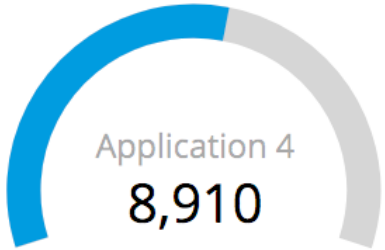
TS - Req v Time



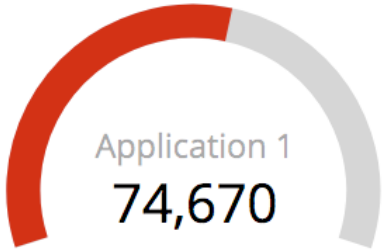
TS - App Requests

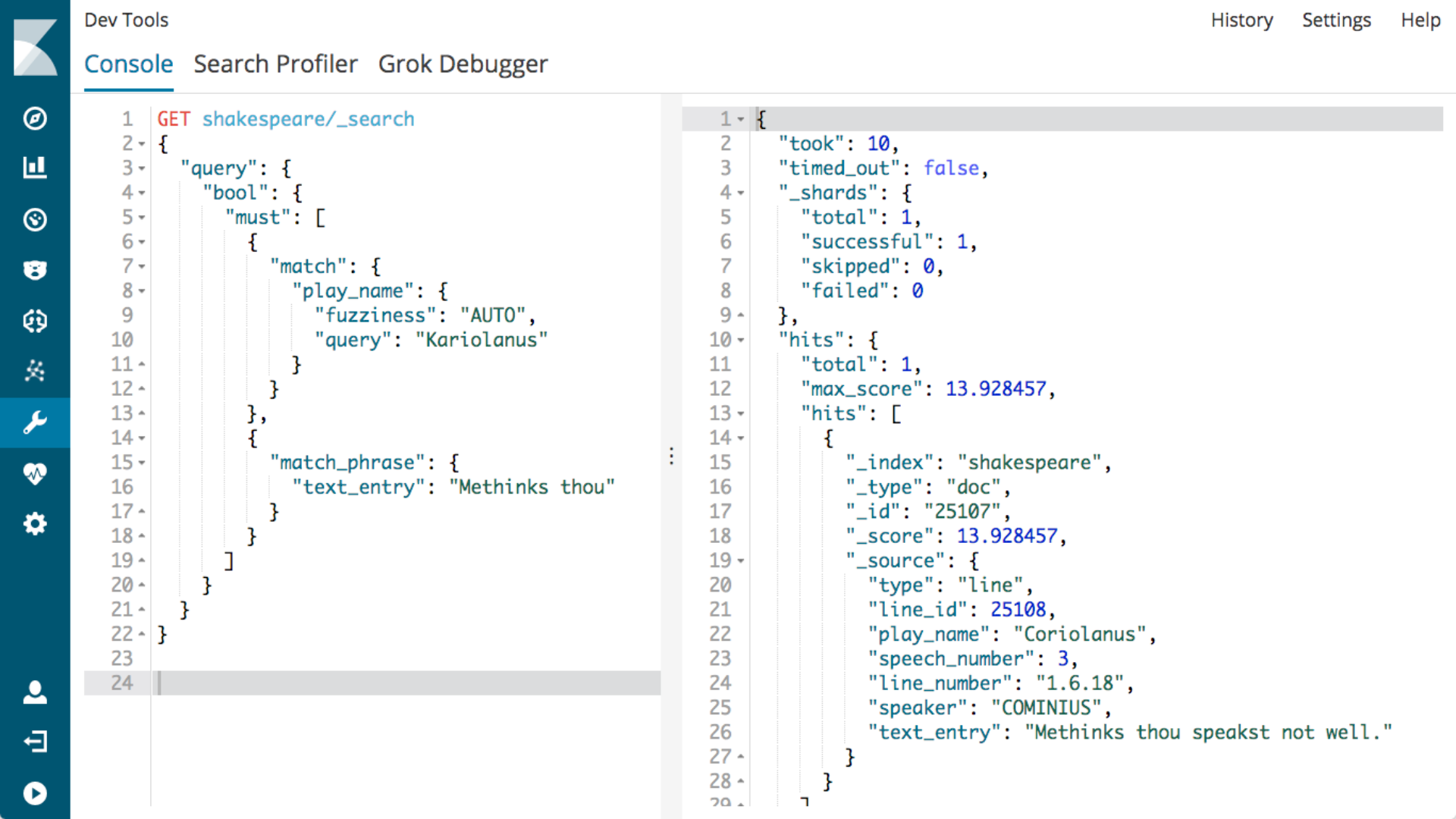


TS - App4



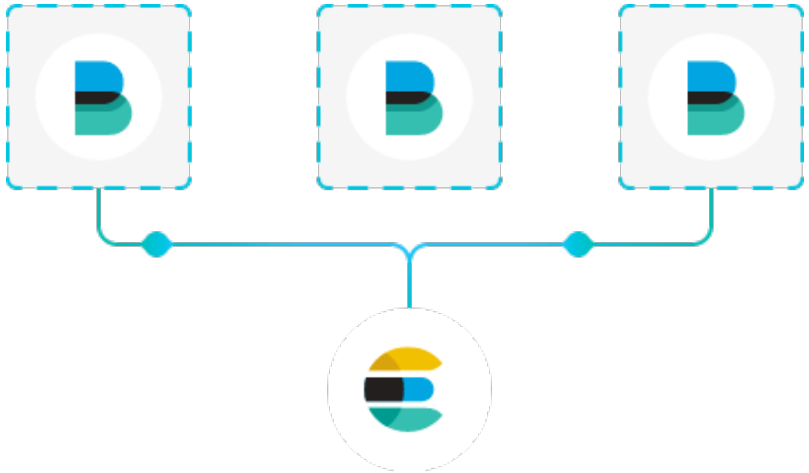
TS - App1



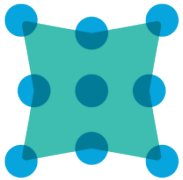

Dev Tools
History
Settings
Help
Console
Search Profiler
Grok Debugger
GET shakespeare/_search
{
"query": {
"bool": {
"must": [
{
"match": {
"play_name": {
"fuzziness": "AUTO",
"query": "Kariolanus"
}
}
},
{
"match_phrase": {
"text_entry": "Methinks thou"
}
}
]
}
}
}
{
"took": 10,
"timed_out": false,
"_shards": {
"total": 1,
"successful": 1,
"skipped": 0,
"failed": 0
},
"hits": {
"total": 1,
"max_score": 13.928457,
"hits": [
{
"_index": "shakespeare",
"_type": "doc",
"_id": "25107",
"_score": 13.928457,
"_source": {
"type": "line",
"line_id": 25108,
"play_name": "Coriolanus",
"speech_number": 3,
"line_number": "1.6.18",
"speaker": "COMINIUS",
"text_entry": "Methinks thou speakst not well."
}
}

Beats

Window into the Elastic Stack



Ship data from the source	**Ship and centralize in Elasticsearch**	**Ship to Logstash for transformation and parsing**
Ship to Elastic Cloud	**Libbeat: API framework to build custom beats**	**30+ community Beats**



PACKETBEAT
Network Data

METRICBEAT
Metrics

WINGLOGBEAT
Window Events

FILEBEAT
Log Files

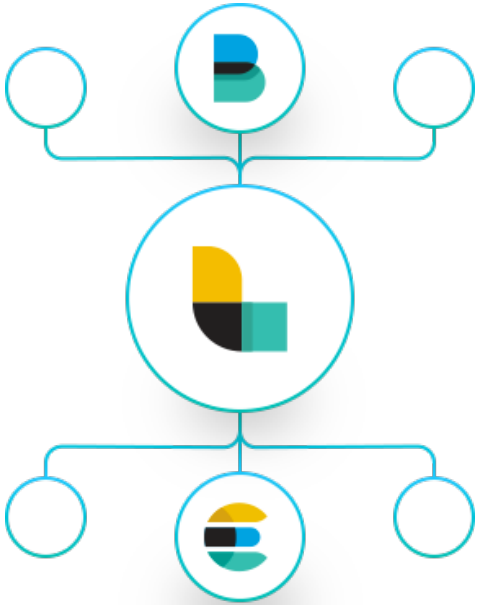
HEARTBEAT
Uptime Monitoring

More than 30 community Beats and growing …

Apachebeat, dockbeat, httpbeat, mysqlbeat, nginxbeat, redis beats, twitterbeat, and more

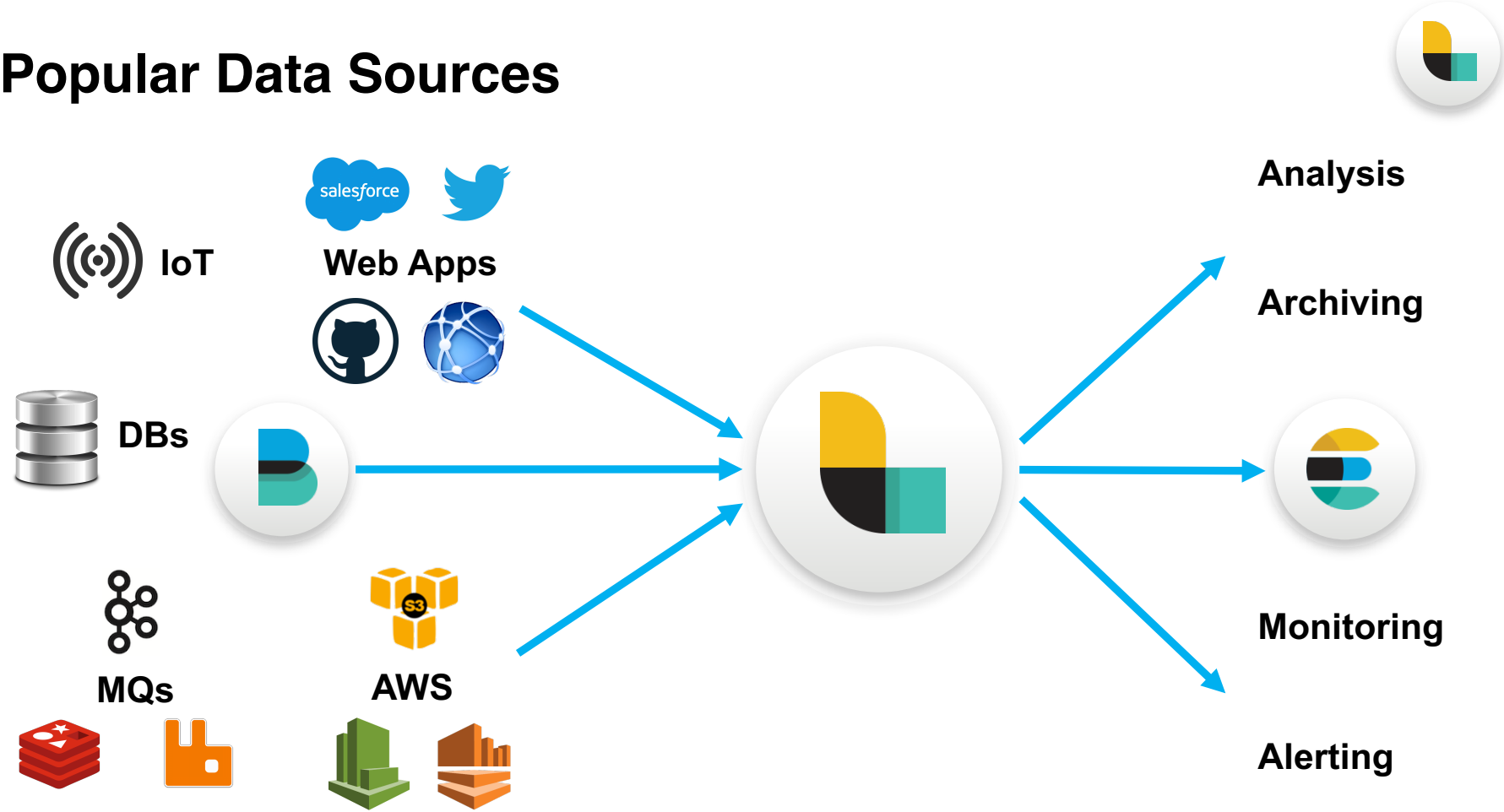
Logstash

Data processing pipeline

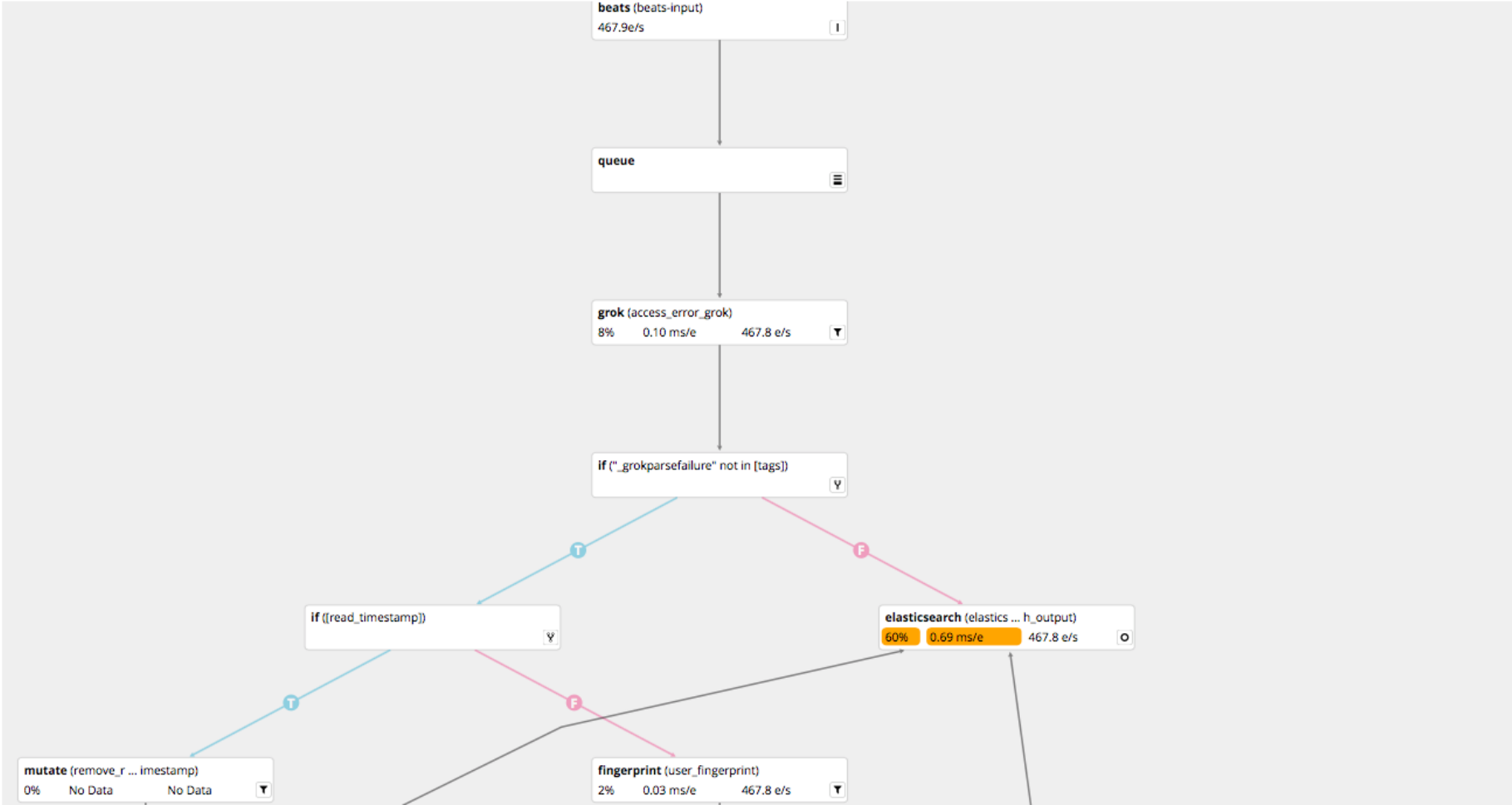


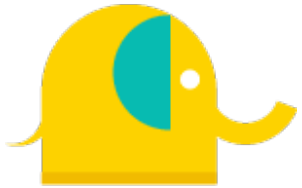
Ingest data of all shapes, sizes, and sources	**Parse and dynamically transform data**	**Transport data to any output**
Secure and encrypt data inputs	**Build your own pipeline**	**More than 200+ plugins**

Popular Data Sources



Version e456ab

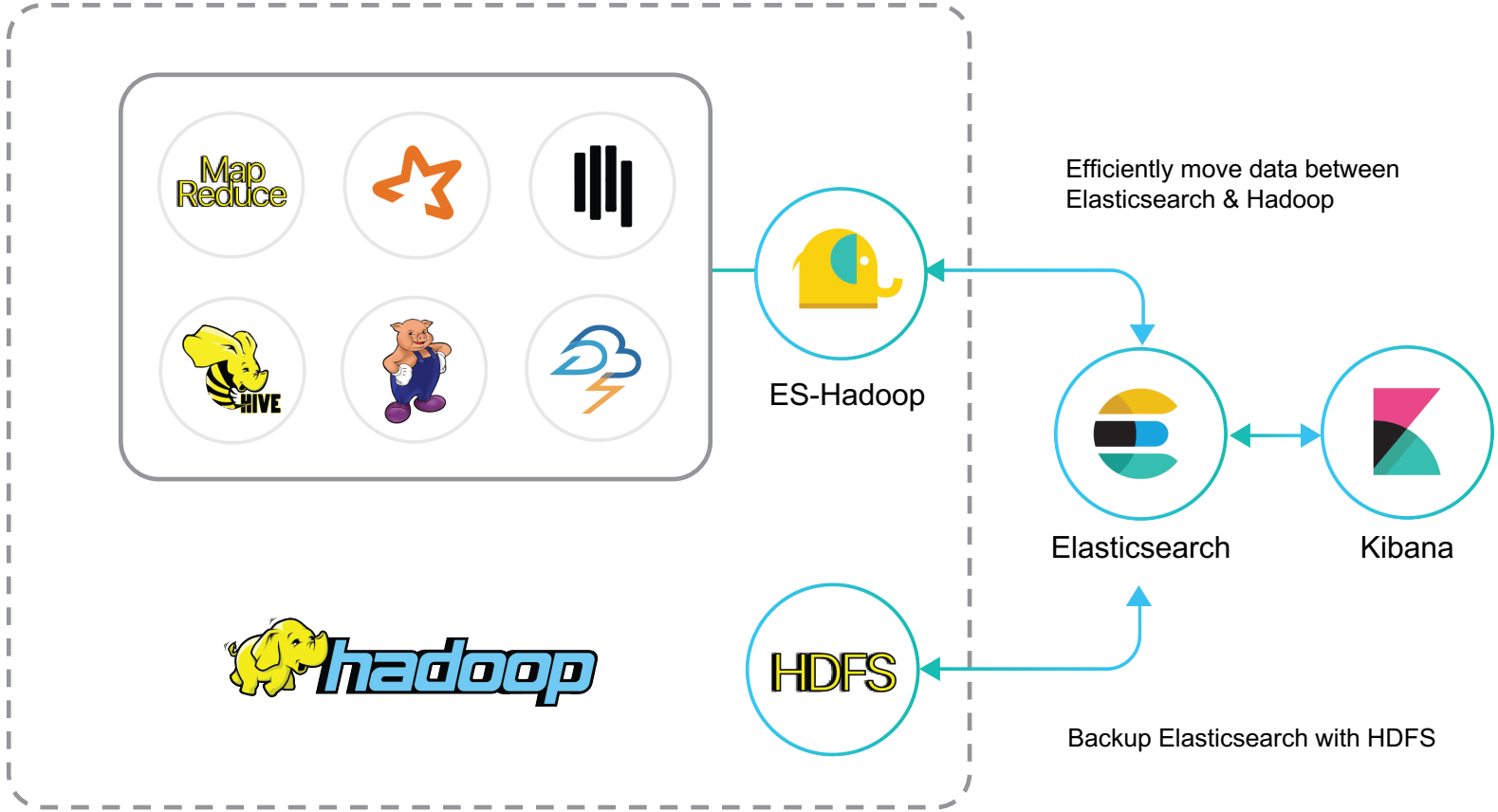




ES-Hadoop

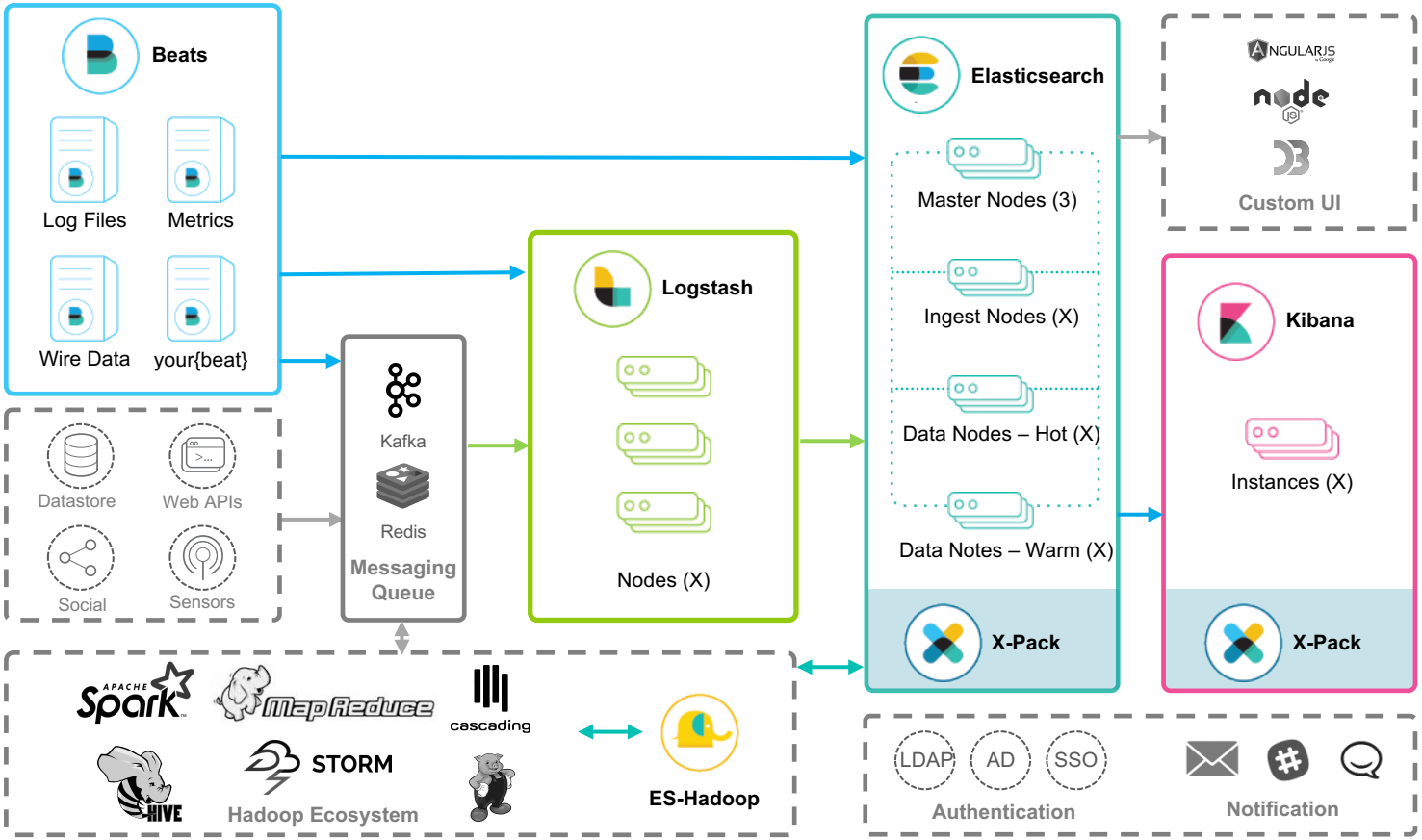
Elasticsearch for Hadoop

Two-way connector	**Index Hadoop data in Elasticsearch**	**Enable real-time search capabilities**
Visualize HDFS data in Kibana	**Snapshot and restore with HDFS**	**Support for Spark, Storm MapReduce, and more**


Map Reduce
HIVE
hadoop
ES-Hadoop
HDFS
Efficiently move data between Elasticsearch & Hadoop
Elasticsearch
Kibana
Backup Elasticsearch with HDFS

elastic

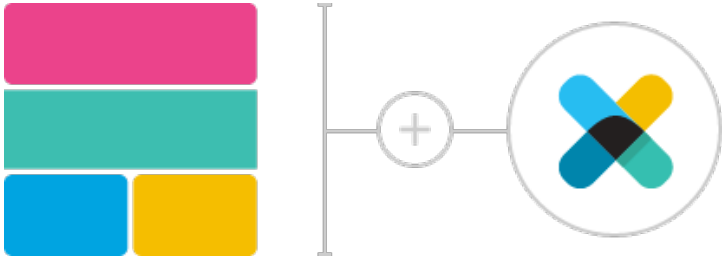
System Architecture



elastic

X-Pack

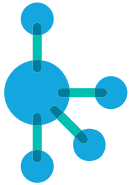
Extensions for the Elastic Stack



Security

Alerting

Monitoring



Graph

Reporting

Machine Learning

X-Pack

Machine Learning

UNSUPERVISED MACHINE LEARNING

- Automatically detect anomalies
- Advanced correlation and categorization
- Identify root cause(s)
- Expose early warning signs

ENABLE NEW USE CASES

- Analyze time series data
- Expand security, IT Ops, fraud, finance, and many more use cases
- Available as beta in the 5.4 release

Chart interval: 1h

Use full server-* data

New job from index pattern server-*

Aggregation

Sum

Field

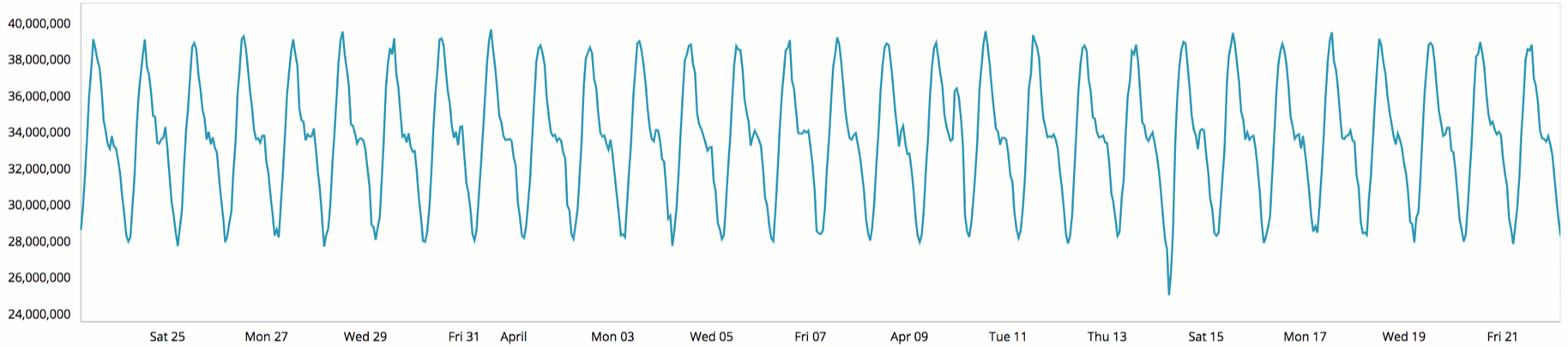
total

Bucket span

30m

Estimate bucket span

Sparse data



Name

sum-total

Description

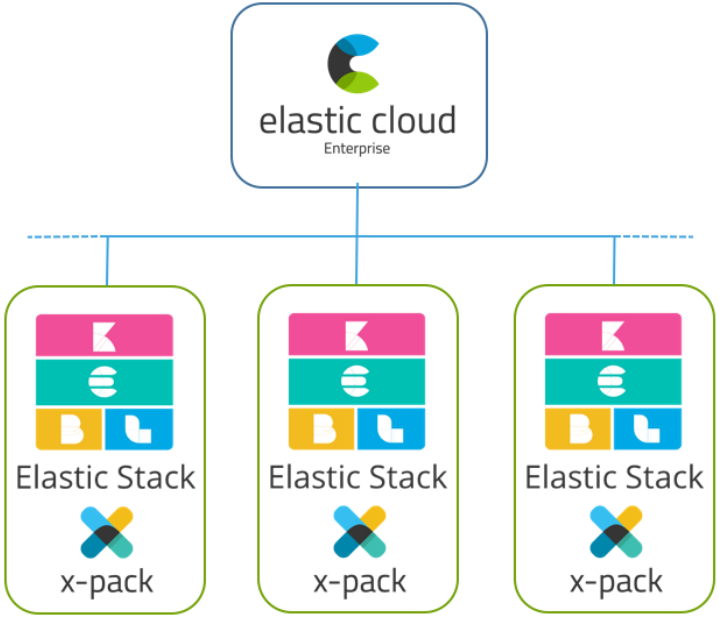
Job description

Advanced

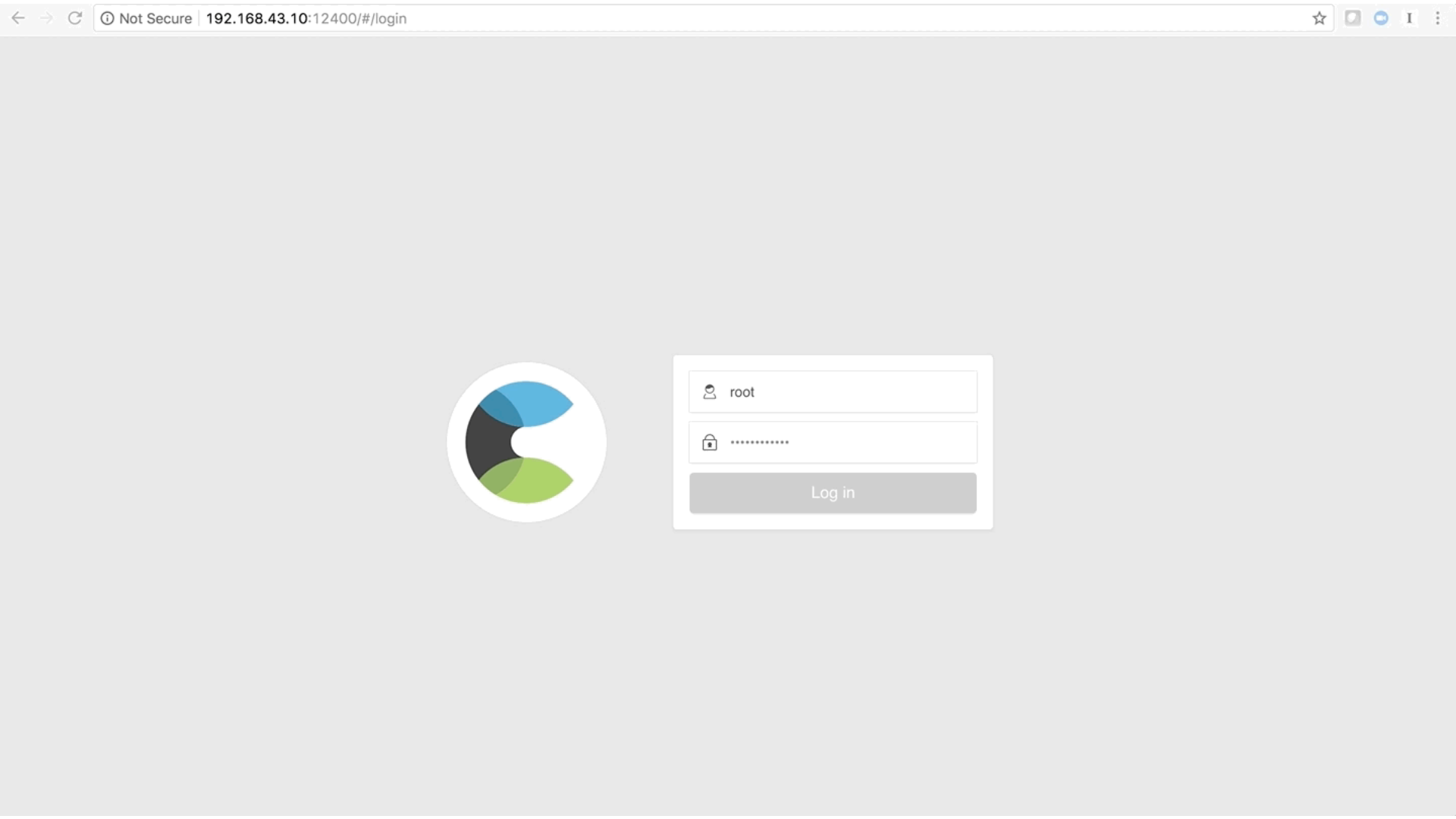
Create Job

Elastic Cloud Enterprise

Provision and manage multiple Elastic Stack environments; Expose logging as a service to your entire organization

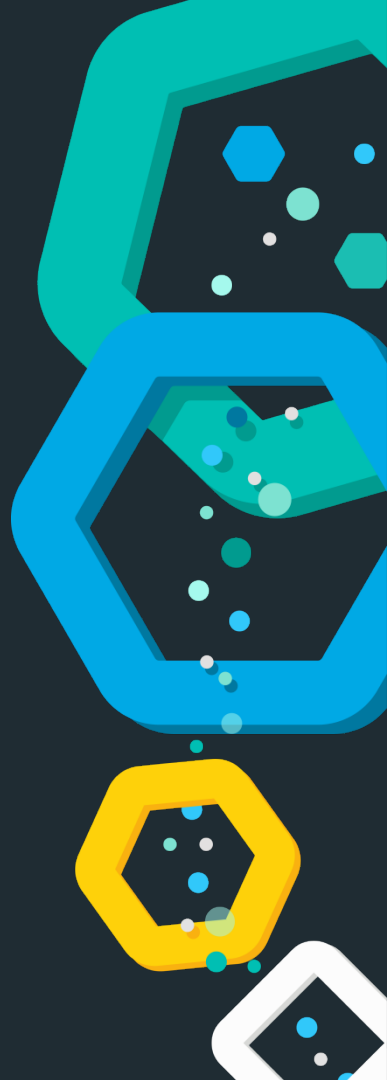


elastic


Not Secure
192.168.43.10:12400/#/login
root
Log in

elastic

Elasticsearch Deep Dive



Elasticsearch is…

*an **open source**, distributed, scalable, highly available, document-oriented, RESTful, full text search engine with real-time search and analytics capabilities*

Apache 2.0 License

https://www.apache.org/licenses/LICENSE-2.0

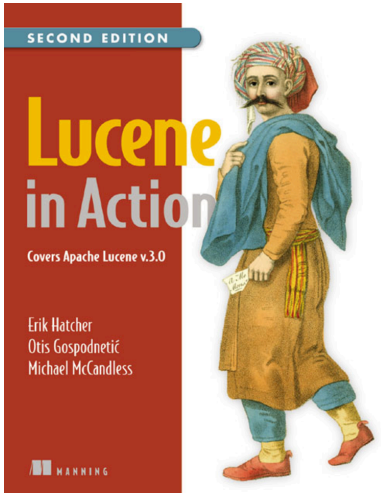
아파치 루씬 (Apache Lucene)

- Created by - Doug Cutting
- Written in - Java
- Apache Solr, Elasticsearch

Welcome Nhat Nguyen as our newest Lucene Committer

Simon Willnauer
to dev@elasticsearch.com, Nhat

I am very happy and proud to announce that the Lucene PMC has voted Nhat Nguyen as a Lucene Committer. Even though the vote has not been publicly announce I want to share this awesome news here. It's not easy to grow new committers into Apache Lucene and Nhat has done an amazing job working on all aspects of the Lucene IndexWriter in the last couple of weeks to deserve the committer bit.



elastic

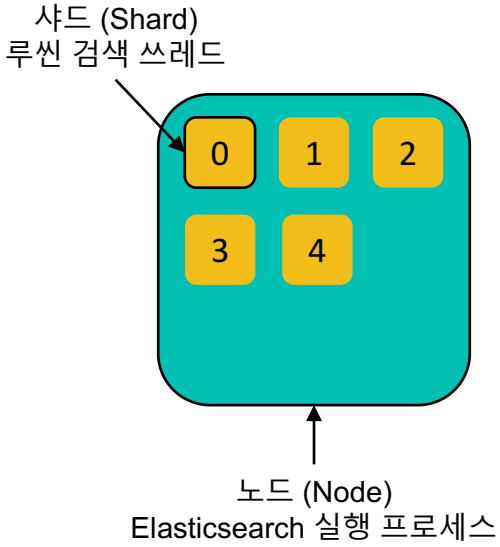
Elasticsearch is…

An open source, ***distributed, scalable, highly available****, document-oriented, RESTful, full text search engine with real-time search and analytics capabilities*

elastic

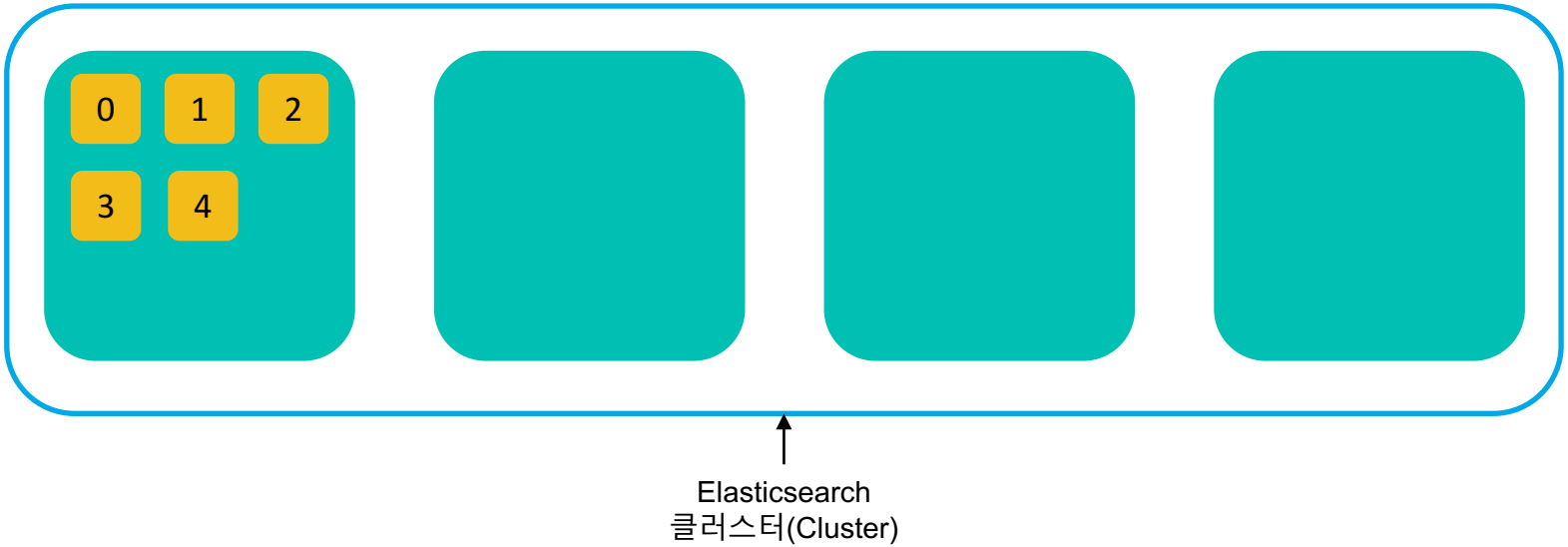
Elasticsearch 클러스터링 과정

대용량 검색을 위해서는 클러스터링이 필요합니다.
Elasticsearch는 데이터를 샤드(Shard) 단위로 분리해서 저장합니다.



Elasticsearch 클러스터링 과정

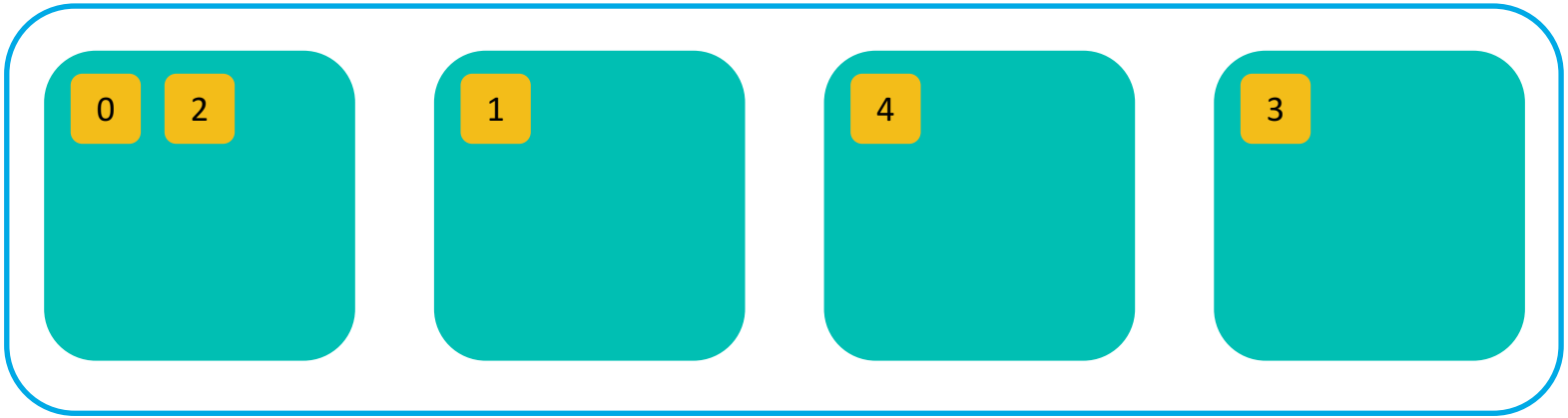
노드를 여러개 실행시키면 같은 클러스터로 묶입니다.



elastic

Elasticsearch 클러스터링 과정

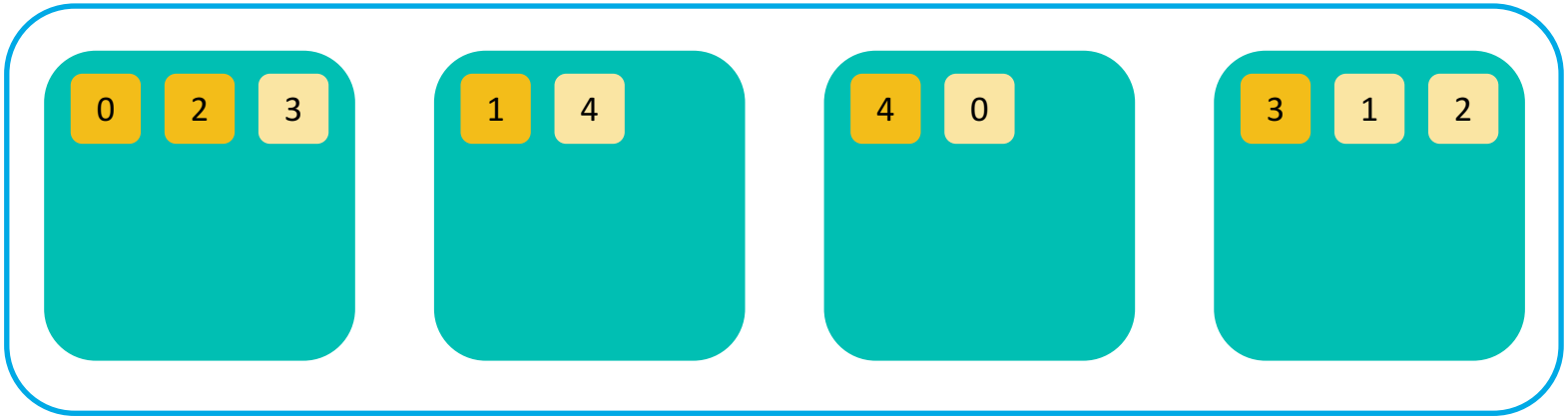
샤드들은 각각의 노드들에 분배되어 저장됩니다.



elastic

Elasticsearch 클러스터링 과정

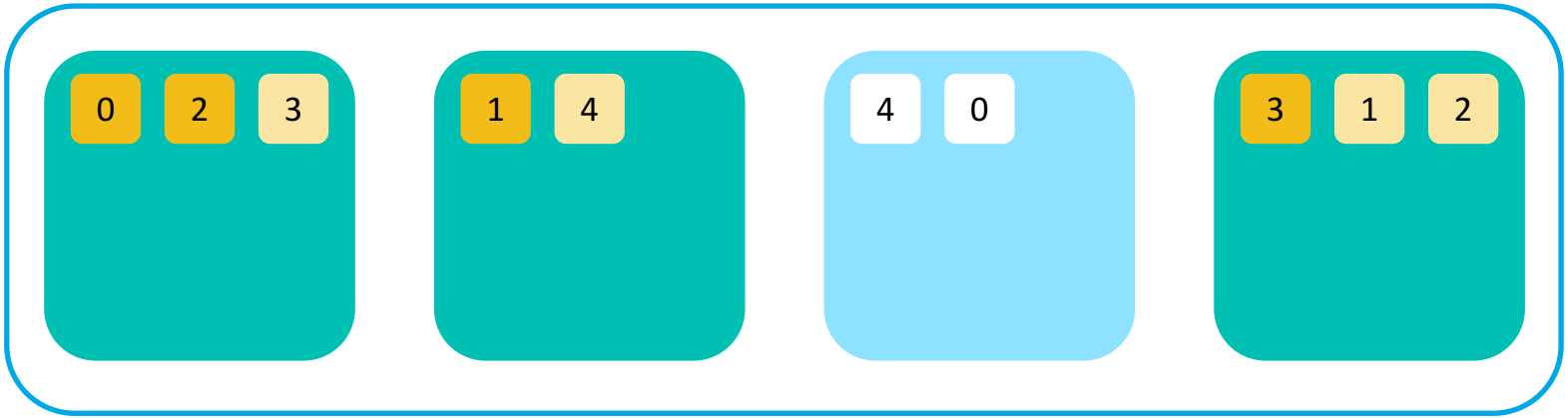
무결성과 가용성을 위해 샤드의 복제본을 만듭니다.
같은 내용의 복제본과 샤드는 서로 다른 노드에 저장됩니다.



elastic

Elasticsearch 클러스터링 과정

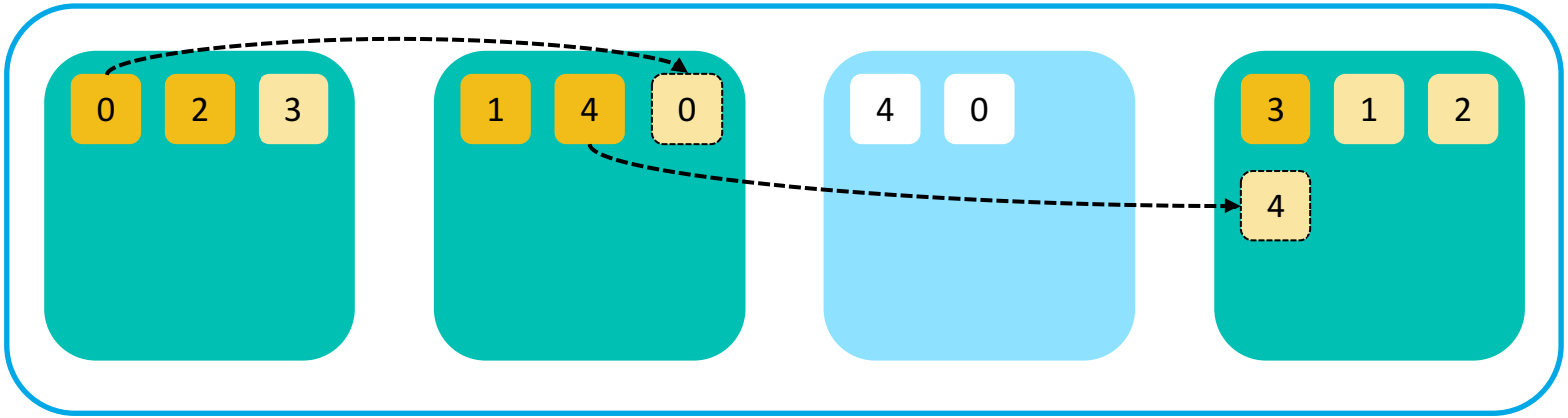
시스템 다운이나 네트워크 단절 등으로 유실된 노드가 생기면



elastic

Elasticsearch 클러스터링 과정

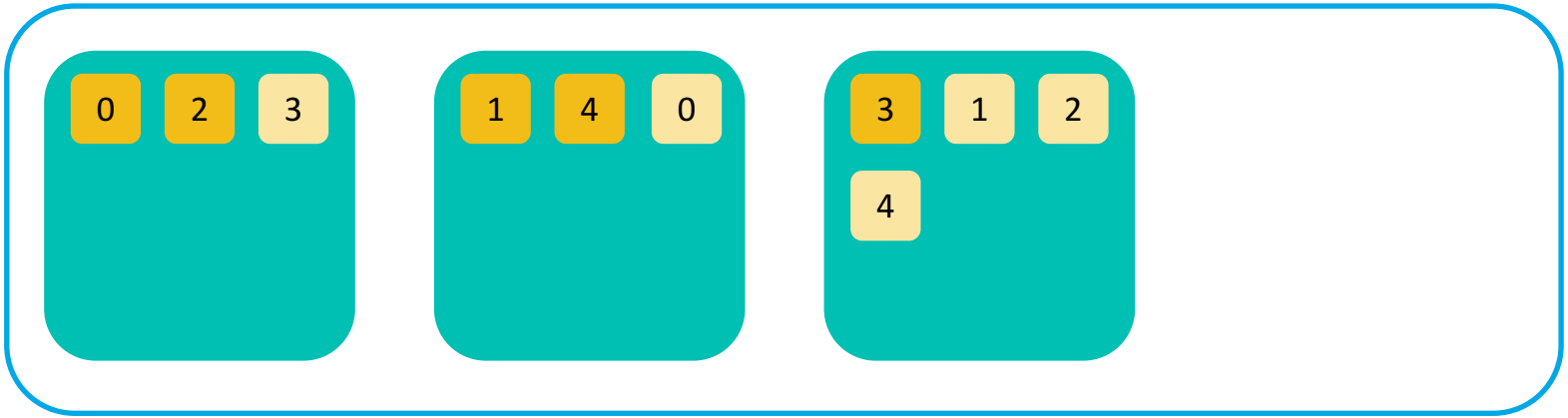
복제본이 없는 샤드들은 다른 살아있는 노드로 복제를 시작합니다.



elastic

Elasticsearch 클러스터링 과정

노드의 수가 줄어들어도 샤드의 수는 변함 없이 무결성을 유지합니다.



elastic

클러스터 (Cluster)

- 엘라스틱서치 시스템의 가장 큰 단위입니다.
- 하나의 클러스터는 다수의 노드로 구성됩니다.
- 하나의 클러스터를 다수의 서버로 바인딩 해서 운영, 또는 역으로 하나의 서버에서 다수의 클러스터의 운용이 가능합니다.

config/elasticsearch.yml

```
cluster.name: elasticsearch
```

```
bin/elasticsearch -E cluster.name=elasticsearch
```

노드 (Node)

- 엘라스틱서치를 구성하는 하나의 단위 프로세스입니다.
- 다수의 샤드로 구성됩니다.
- 같은 클러스터명을 가진 노드들은 자동으로 바인딩 됩니다.

config/elasticsearch.yml

```
node.name: "Node1"
```

```
bin/elasticsearch -E node.name=Node1
```

Master & Data 노드

- 마스터 노드 : 클러스터 상태 정보를 관리합니다.
- 데이터 노드 : 데이터 입/출력, 검색을 수행합니다.

config/elasticsearch.yml

```
node.name: "Node1"
node.master: true
node.data: false
```

```
node.name: "Node2"
node.master: false
node.data: true
```

elastic

샤드(Shard) & 레플리카(Replica)

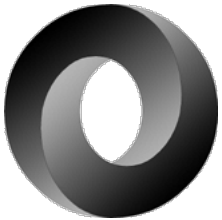
- 샤드 : 루씬 검색 인스턴스입니다.
- 레플리카 : 데이터 무결성 유지를 위한 샤드의 복사본입니다.

```
curl -XPUT "http://localhost:9200/books" -H 'Content-Type: application/json' -d'
{
  "settings": {
    "number_of_shards": 5,
    "number_of_replicas": 1
  }
}'
```

```
curl -XPUT "http://localhost:9200/books/_settings" -H 'Content-Type: application/json' -d'
{
"number_of_replicas": 0
}'
```

Elasticsearch is…

An open source, distributed, scalable, highly available, ***document-oriented****, RESTful, full text search engine with real-time search and analytics capabilities*

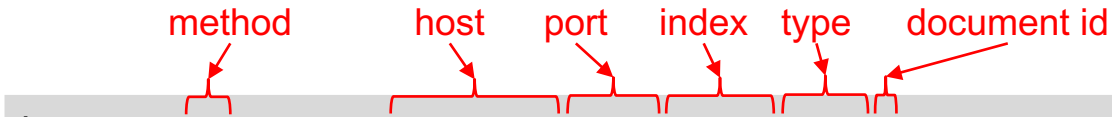


Source: http://json.org

```
{
  "name" : "Craft"
  "geo" : {
    "city" : "Budapest",
    "lat" : 47.49, "lon" : 19.04
  }
}
```

Elasticsearch is…

An open source, distributed, scalable, highly available, document-oriented, ***RESTful****, full text search engine with real-time search and analytics capabilities*




```
$ curl -XPUT http://localhost:9200/books/book/1 -d '
{
  "title" : "Elasticsearch Guide",
  "author" : "Kim",
  "date" : "2014-05-01",
  "pages" : 250
}'
{"_index":"books","_type":"book","_id":"1","_version":1,"created":true}
```


elastic

데이터 입력

PUT 메소드를 이용해서 본문으로 JSON 문서를 입력합니다.

```
curl -XPUT 'http://localhost:9200/books/book/1' -d '
{ "title": "Romeo and Juliet", "author": "William Shakespeare", "category":"Tragedies",
 "written": "1562-12-01T20:40:00", "pages" : 125 }' -H 'Content-Type: application/json'
```

```
curl -XPUT 'http://localhost:9200/books/book/2' -d '
{ "title" : "Hamlet", "author": "William Shakespeare", "category":"Tragedies",
 "written": "1599-06-01T12:34:00", "pages" : 172 }' -H 'Content-Type: application/json'
```

```
curl -XPUT 'http://localhost:9200/books/book/3' -d '
{ "title": "The Prince and the Pauper", "author": "Mark Twain", "category":"Children book",
 "written": "1881-08-01T10:34:00", "pages" : 79}' -H 'Content-Type: application/json'
```

데이터 조회 (접근)

GET 메소드 (생략 가능) 으로 도큐먼트에 접근합니다.
pretty 매개변수를 이용해서 보기 좋게 출력이 가능합니다.

```
curl -XGET 'http://localhost:9200/books/book/1?pretty' -H 'Content-Type: application/json'
```

```
{
  "_index" : "books",
  "_type" : "book",
  "_id" : "1",
  "_version" : 3,
  "found" : true,
  "_source" : {
    "title" : "Romeo and Juliet",
    "author" : "William Shakespeare",
    "category" : "Tragedies",
    "written" : "1562-12-01T20:40:00",
    "pages" : 125
  }
}
```

데이터 삭제

DELETE 메소드를 이용해서 **도큐먼트** 또는 **인덱스** 단위로 삭제합니다.

```
curl -XDELETE localhost:9200/books/book/1 -H 'Content-Type: application/json'
```

```
curl -XDELETE localhost:9200/books -H 'Content-Type: application/json'
```

데이터 배치 입력

_bulk API 를 이용해서 여러개의 문서를 배치로 입력합니다.

```
curl -XPOST "http://localhost:9200/books/book/_bulk" -d '
{"index":{"_id":"1"}}
{"title":"Romeo and Juliet","author":"William Shakespeare","category":"Tragedies","written":"1562-12-01T20:40:00","pages":125}
{"index":{"_id":"2"}}
{"title":"Hamlet","author":"William Shakespeare","category":"Tragedies","written":"1599-06-01T12:34:00","pages":172}
{"index":{"_id":"3"}}
{"title":"The Prince and the Pauper","author":"Mark Twain","category":"Children book","written":"1881-08-01T10:34:00","pages":79}
' -H 'Content-Type: application/json'
```

데이터 검색

_search API를 사용해서 인덱스 단위로 검색합니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty=true"
```

```
{ … 중략 …
},
  "hits" : {
    "total" : 3,
    "max_score" : 1.0,
    "hits" : [ {
… 중략 …
      "_source":
{ "title": "Romeo and Juliet", "author": "William Shakespeare", "category":"Tragedies", "written":
"1562-12-01T20:40:00", "pages" : 125 }
    },
… 중략 …
```


elastic

데이터 검색 (URI)

URI 의 q 매개변수로 검색이 가능합니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty=true"
```

```
curl -XGET "http://localhost:9200/books/_search?q=author:william&pretty=true"
```

```
curl -XGET "http://localhost:9200/books/_search?q=william&df=author&pretty=true"
```

```
curl -XGET "http://localhost:9200/books/_search?q=william%20AND%20romeo&pretty=true"
```

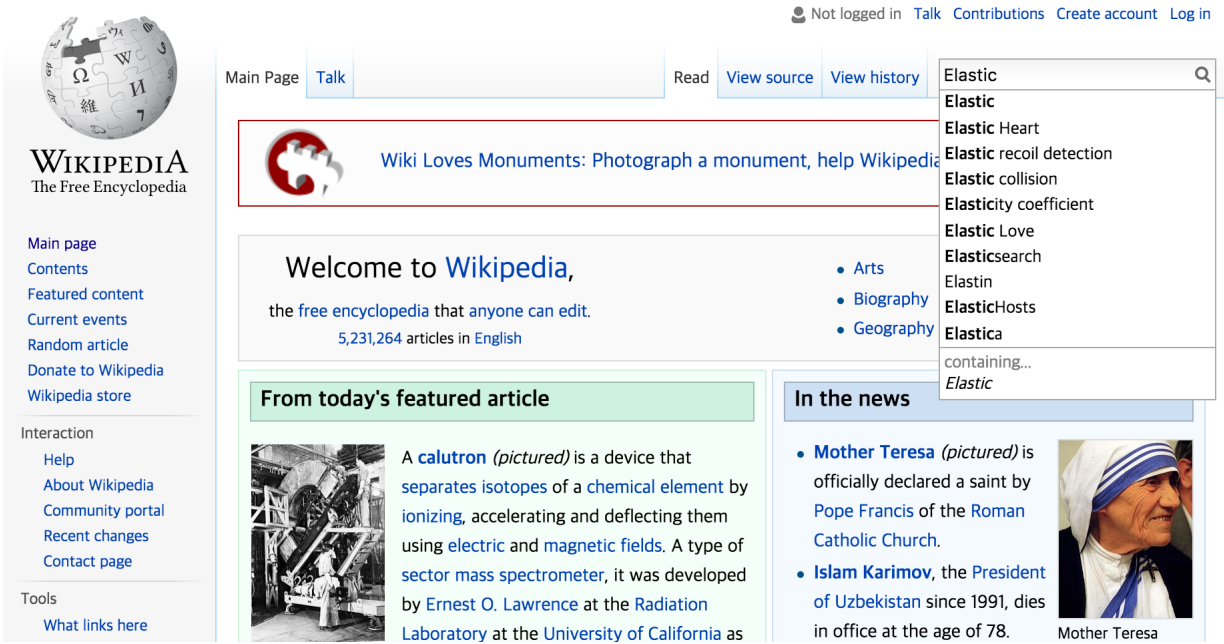
데이터 검색 (Request Body)

http 의 본문에 쿼리를 삽입합니다. 더욱 복잡한 질의를 할 수 있습니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty=true" -d'
{
  "query": {
    "match": {
      "author": "william"
    }
  }
}' -H 'Content-Type: application/json'
```

Elasticsearch is…

An open source, distributed, scalable, highly available, document-oriented, RESTful, ***full text search engine*** *with real-time search and analytics capabilities*



elastic

RDBMS 에서는 데이터를 테이블 형태로 저장합니다.

열을 기준으로 인덱스를 만듭니다.
책의 맨 앞에 있는 제목 리스트와 같습니다.

DOC	TEXT
1	The quick brown fox jumps over the lazy dog
2	Fast jumping rabbits

검색엔진에서는 inverted index 라는 구조로 저장합니다.

RDBMS 와 반대 구조입니다.
텍스트를 다 뜯어서 검색어 사전을 만듭니다. (Term 이라고 합니다)
책의 맨 뒤에 있는 페이지를 가리키는 키워드 같습니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
Fast	2	jumps	1
The	1	lazy	1
brown	1	over	1
dog	1	quick	1
fox	1	rabbits	2
jumping	2	the	1

실제로는 이렇게 저장됩니다.

텍스트를 저장할 때 몇가지 처리 과정을 거칩니다.
이 과정을 텍스트 분석 (Text Analysis) 라고 합니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
brown	1	lazi	1
dog	1	over	1
fast	1 , 2	quick	1 , 2
fox	1	rabbit	2
jump	1 , 2		

텍스트 분석 과정

문장을 분리합니다. 이 과정을 Tokenizing 이라고 합니다.
보통은 **Whitespace Tokenizer** 가 사용됩니다.

TEXT
The quick brown fox jumps over the lazy dog
Fast jumping rabbits

TOKEN (TERM)	TOKEN (TERM)	TOKEN (TERM)	TOKEN (TERM)
Fast	dog	jumps	quick
The	fox	lazy	rabbits
brown	jumping	over	the

텍스트 분석 과정

Tokenized 된 Term 들을 가공합니다. 이 과정을 Token Filtering 이라고 합니다.
먼저 **Lowercase Token Filter**로 대소문자를 변환 합니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
Fast → fast	2	jumps	1
The → the	1	lazy	1
brown	1	over	1
dog	1	quick	1
fox	1	rabbits	2
jumping	2	the	1

텍스트 분석 과정

토큰을 (보통 ascii 순서로) 재 정렬합니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
brown	1	lazy	1
dog	1	over	1
fast	2	quick	1
fox	1	rabbits	2
jumping	2	the	1
jumps	1	the	1

텍스트 분석 과정

불용어(stopwords, 검색어로서의 가치가 없는 단어들) 를 제거합니다.
a, an, are, at, be, but, by, do, for, i, no, the, to … 등등
Stop Token Filter 가 사용됩니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
brown	1	lazy	1
dog	1	over	1
fast	2	quick	1
fox	1	rabbits	2
jumping	2	~~the~~	~~1~~
jumps	1	~~the~~	~~1~~

텍스트 분석 과정

형태소 분석 과정을 거칩니다. 보통 ~s, ~ing 등을 제거하는 과정입니다.
보통 **Snowball Token Filter** 를 사용합니다.
한글은 의미 분석을 해야 해서 좀 더 복잡합니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
brown	1	lazy → lazi	1
dog	1	over	1
fast	2	quick	1
fox	1	rabbits → rabbit	2
jumping → jump	2		
jumps → jump	1		

텍스트 분석 과정

jumping, jumps가 jump 로 똑같이 바뀌었으므로 토큰을 병합 해 줍니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
brown	1	lazi	1
dog	1	over	1
fast	2	quick	1
fox	1	rabbit	2
jump	1 , 2		

텍스트 분석 과정

동의어를 처리합니다.
Synonym Token Filter 를 이용해 동의어 사전을 정의할 수 있습니다.

TOKEN (TERM)	DOC	TOKEN (TERM)	DOC
brown	1	lazi	1
dog	1	over	1
fast	1 , 2	quick	1 , 2
fox	1	rabbit	2
jump	1 , 2		

_analyze API

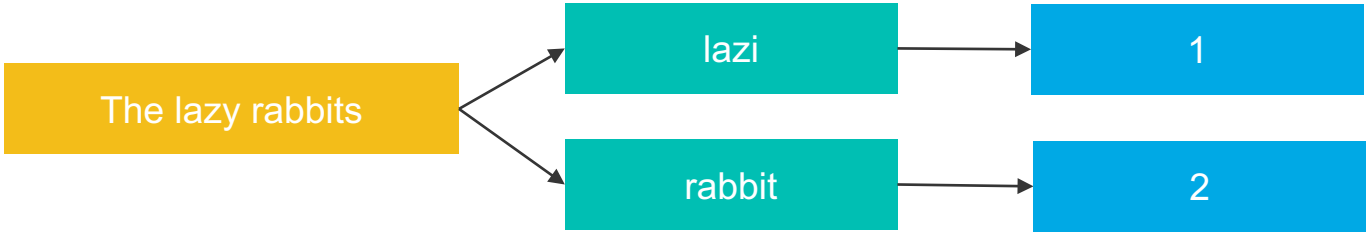
분석된 텍스트를 미리 볼 수 있습니다.
analyzer 는 1개의 tokenizer 와 n개의 token filter로 구성됩니다.

```
curl -XPOST "http://localhost:9200/_analyze?pretty" -d '
{
  "tokenizer": "whitespace",
  "filter": ["lowercase", "stop", "snowball"],
  "text": [
    "The quick brown fox jumps over the lazy dog",
    "Fast jumping rabbits"
  ]
}' -H 'Content-Type: application/json'
```

검색 과정

검색어도 똑같이 텍스트 처리를 합니다.

“The lazy rabbits” 라고 검색하면



DOC	TEXT
1	The quick brown fox jumps over the lazy dog
2	Fast jumping rabbits

검색엔진은

	RDBMS	검색엔진
데이터 저장 방식	정규화	역정규화
전문(Full Text) 검색 속도	느림	빠름
의미 검색	불가능	가능
Join	가능	불가능
수정 / 삭제	빠름	느림

텀(Term) 확인

termvectors 를 이용해서 저장된 데이터의 확인이 가능합니다.

```
curl -XGET "http://localhost:9200/books/book/1/_termvectors?fields=author&pretty"
```

```
"terms": {
  "shakespeare": {
    "term_freq": 1,
    "tokens": [ … ]
  },
  "william": {
    "term_freq": 1,
    "tokens": [ … ]
  }
```

match 검색

검색어도 analyze 과정을 거칩니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty=true" -d'
{
  "query": {
    "match": {
      "author": "William"
    }
  }
}' -H 'Content-Type: application/json'
```

term 검색

검색어의 analysis 과정을 거치지 않습니다.
검색어가 저장된 텀과 정확히 일치해야 결과가 나타납니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty=true" -d'
{
  "query": {
    "term": {
      "author": "William"
    }
  }
}' -H 'Content-Type: application/json'
```

매핑 (Mapping)

- 각 필드별로 데이터를 저장하는 스키마 명세이며 인덱스/타입 별로 구분됩니다.
- 매핑이 없는 경우 데이터를 처음 입력하면 매핑이 자동 생성됩니다.
- 자동 생성된 매핑의 텍스트 필드는 기본적으로 standard analyzer 가 적용되며 **keyword** 타입의 멀티 필드가 생깁니다.

```
curl -XGET "http://localhost:9200/books/_mappings?pretty"
```

```
        "properties" : {
          "author" : {
            "type" : "text",
            "fields" : {
              "keyword" : {
                "type" : "keyword",
                "ignore_above" : 256
              }
```

Keyword

- Keyword 타입은 Analyze 과정을 거치지 않은 해당 필드의 데이터의 원본 데이터 입니다.
- Term 검색으로 검색시 대소문자까지 정확히 일치해야 합니다.

```
curl -XGET "http://localhost:9200/books/book/1/_termvectors?fields=author.keyword&pretty"
```

```
"terms" : {
  "William Shakespeare" : {
    "term_freq" : 1,
    "tokens" : [ … ]
  }
}
```

Keyword

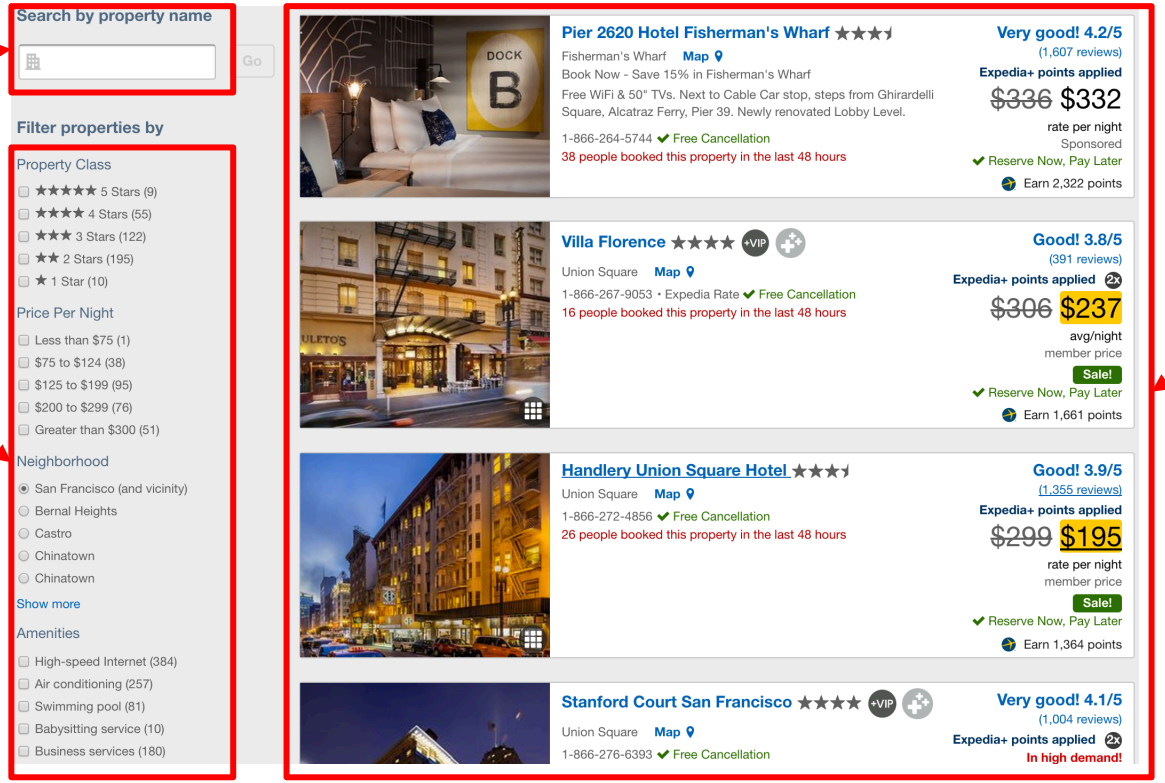
- Term 검색으로 검색시 대소문자까지 정확히 일치해야 합니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty=true" -d'
{
  "query": {
    "term": {
      "author.keyword": "William Shakespeare"
    }
  }
}' -H 'Content-Type: application/json'
```

Aggregation

Search

aggregation



hits

elastic

Aggregation

_search API 에서 query 와 함께 사용이 가능합니다.

```
curl 'localhost:9200/_search' -d '
{
  "query" : {
    // query
  },
  "aggregations" : {    // or "aggs"
    "aggs_name" : {
      // a set of aggregation
    }
  }
}'
```

Aggregation

_search API 에서 query 와 함께 사용이 가능합니다.

```
curl -XGET "http://localhost:9200/books/_search?pretty" -d'
{
  "query": {
    "match_all": {}
  },  "aggs": {
    "authors": {
      "terms": {
        "field": "author.keyword"
      }
    }
  }
}' -H 'Content-Type: application/json'
```

```
…
 "aggregations" : {
    "authors" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "William Shakespeare",
          "doc_count" : 2
        },
        {
          "key" : "Mark Twain",
          "doc_count" : 1
        }
      ]
    }
```

Aggregation

Bucket, Metric, Pipeline

- Bucket
  - 도큐먼트 집단 단위인 버킷을 생성합니다.
  - 또 다른 Bucket 또는 Metric 의 sub-aggregation을 포함합니다.
- Metric
  - 도큐먼트 집단에 대한 하나 또는 그 이상의 계산된 수치를 포함합니다.
- Pipeline
  - agg 결과에 대한 새로운 계산을 실행합니다.

Aggregation

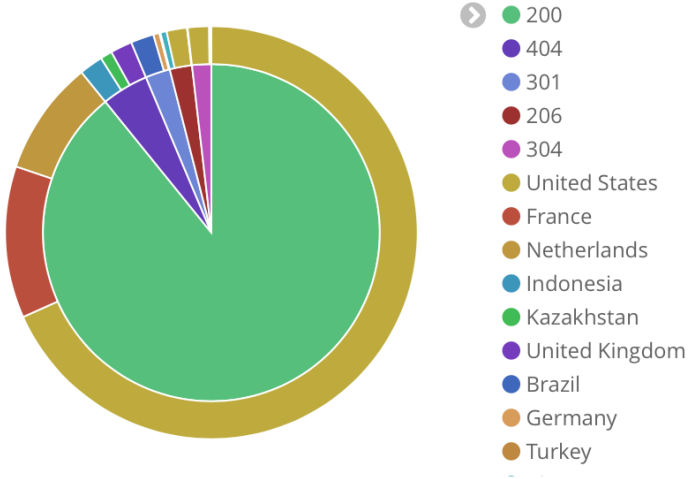
- percentile
- percentile_ranks
- cardinality
- significant_terms
- top hits
- scripted_metric
- filters
- range
- geohash
- terms
- histogram
- date_histogram
- stats
- extended stats
- min / max
- sum
- pipeline aggregations

Agg. Combination

```
curl -XGET "http://localhost:9200/books/_search?pretty" -d'
{
  "query": {
    "match_all": {}
  },  "aggs": {
    "authors": {
      "terms": {
        "field": "author.keyword"
      },
      "aggs": {
        "pages_per_author": {
          "sum": { "field": "pages" }
        }
      }
    }
  }
}' -H 'Content-Type: application/json'
```

```
…
  "aggregations" : {
    "authors" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "William Shakespeare",
          "doc_count" : 2,
          "pages_per_author" : {
            "value" : 297.0
          }
        },
        {
          "key" : "Mark Twain",
          "doc_count" : 1,
          "pages_per_author" : {
            "value" : 79.0
          }
        }
      ]
    }
```

Agg. Combination in Kibana

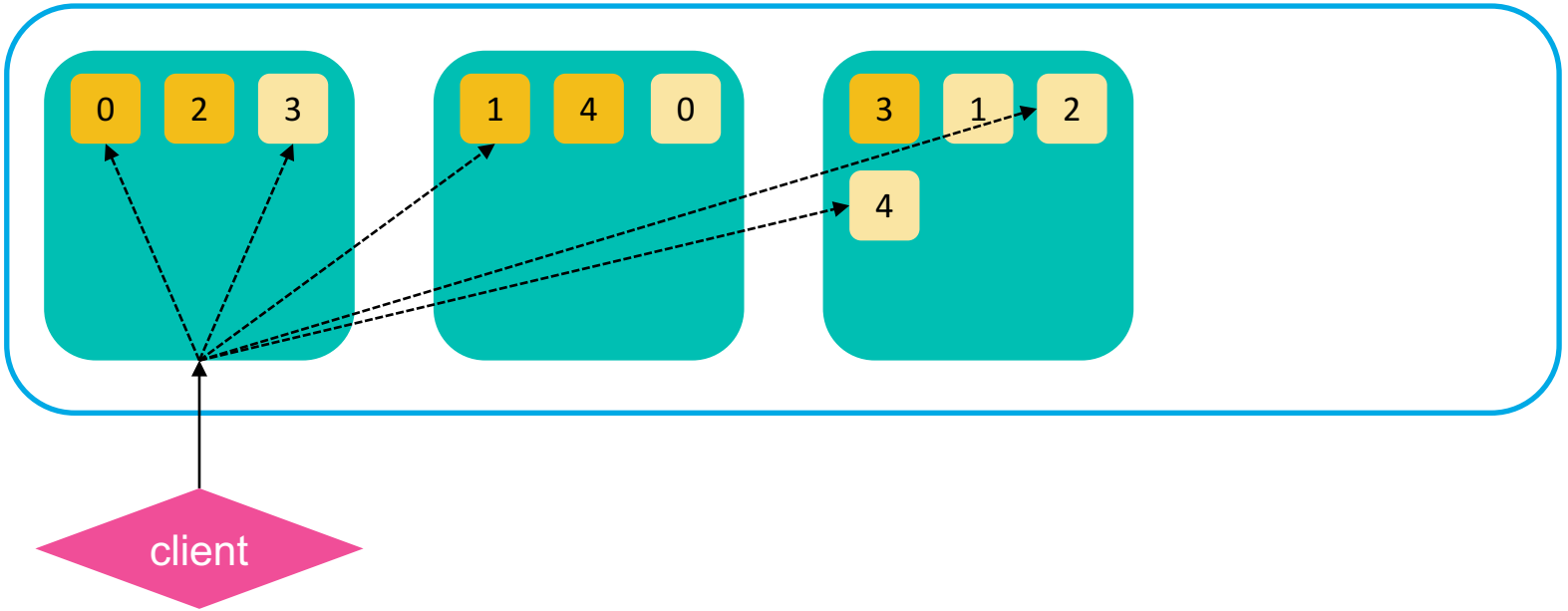



```
"aggregations": {
  "2": {
    "doc_count_error_upper_bound": 0,
    "sum_other_doc_count": 145,
    "buckets": [
      {
        "3": {
          "doc_count": 267343,
          "buckets": [
            {
              "key": "United States",
              "doc_count": 105999,
              "score": 0.010441924596790206,
              "bg_count": 115895
            },
            {
              "key": "France",
              "doc_count": 18450,
              "score": 0.004923687375361934,
              "bg_count": 19325
            },
            {
              "key": "Netherlands",
              "doc_count": 13905,
              "score": 0.004078378127678101,
              "bg_count": 14469
            }
          ]
        },
        "key": 200,
        "doc_count": 267343
      },
      {
        "3": {
```


elastic

검색 과정 – 1. Query Phase

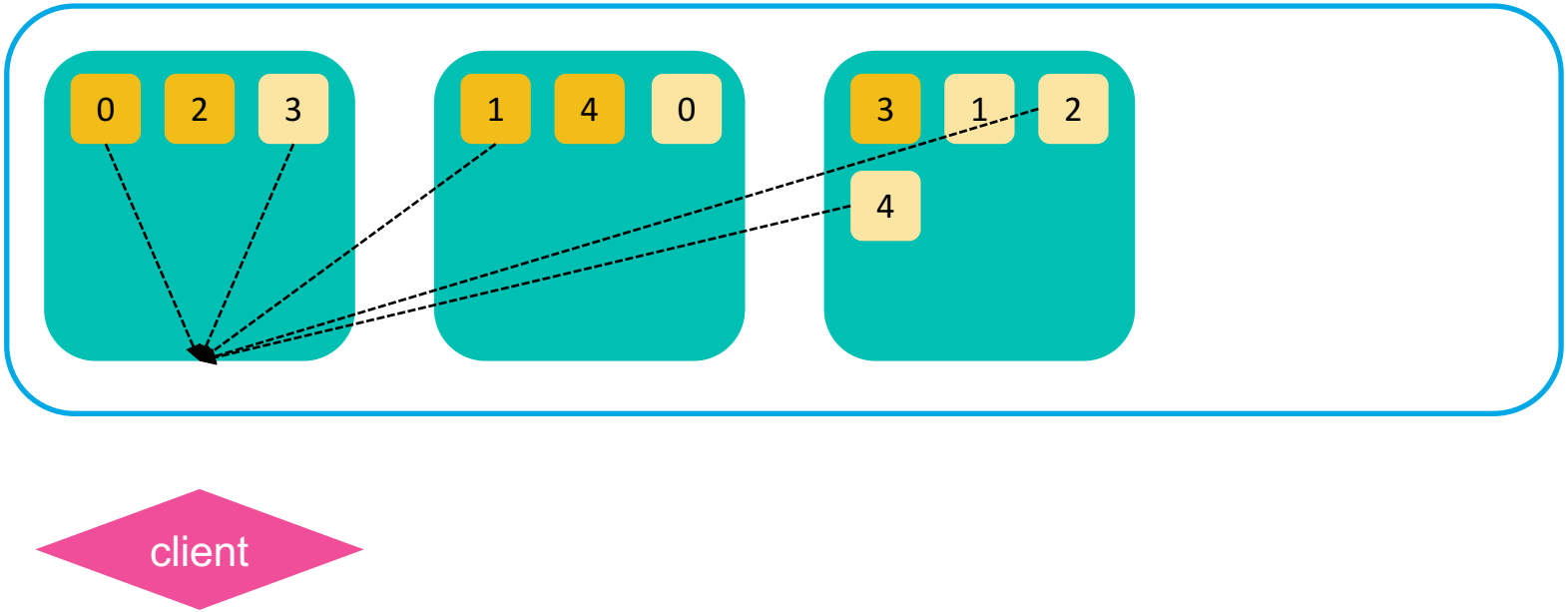
처음 쿼리 수행 명령을 받은 노드는 모든 샤드에게 쿼리를 전달합니다.
1차적으로 모든 샤드(또는 복제본에서) 검색을 실행합니다.



elastic

검색 과정 – 1. Query Phase

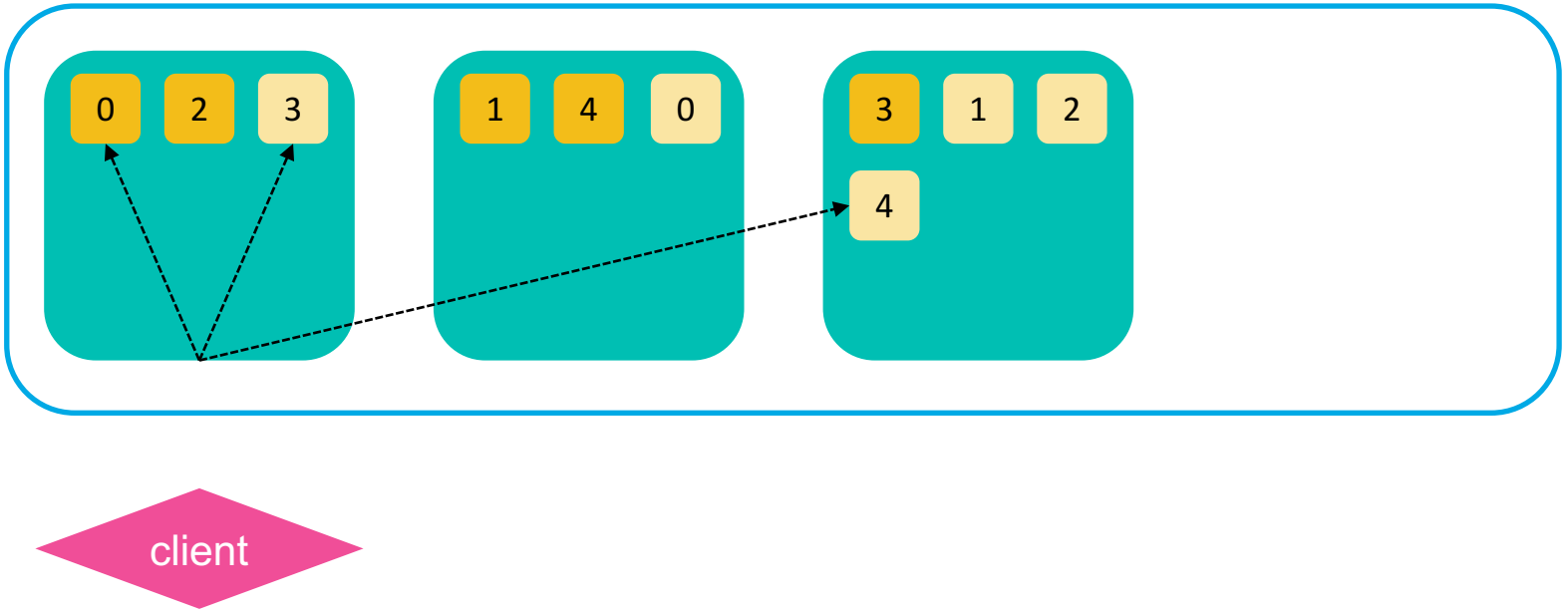
각 샤드들은 요청된 크기만큼의 검색 결과 큐를 노드로 리턴합니다.
리턴된 결과는 루씬 doc id 와 랭킹 점수만 가지고 있습니다.



elastic

검색 과정 – 2. Fetch Phase

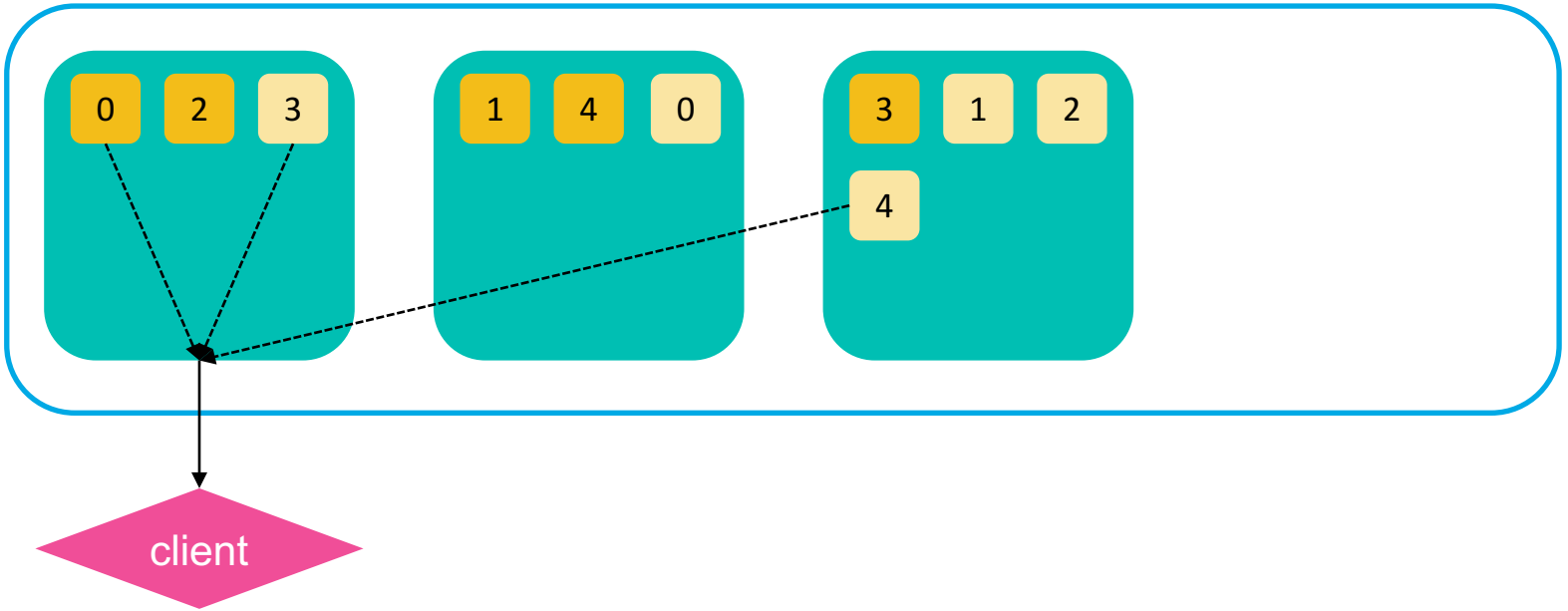
노드는 리턴된 결과들의 랭킹 점수를 기반으로 정렬한 뒤
유효한 샤드들에게 최종 결과들을 다시 요청합니다.



elastic

검색 과정 – 2. Fetch Phase

전체 문서 내용(_source) 등의 정보가 리턴되어 클라이언트로 전달됩니다.



elastic

검색엔진에서는
정확한 검색을 위한
랭킹 알고리즘이
정말 정말 중요합니다.

랭킹 알고리즘

보통은 TF/IDF 를 많이 씁니다.

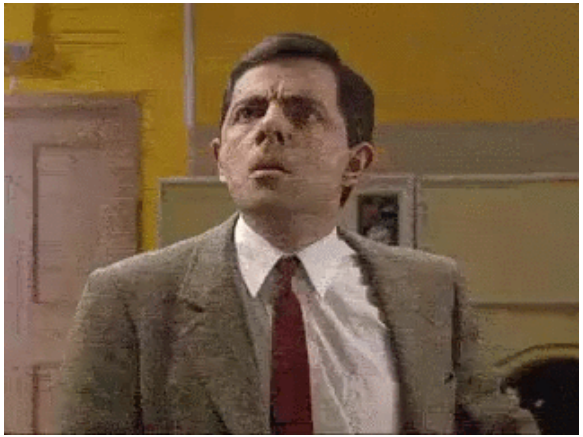
$$\mathrm{tf}(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max\{f_{t',d} : t' \in d\}}$$

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t,D)$$

Elasticsearch 5.0 부터는
BM25 라는 알고리즘을 사용합니다.

$$\mathrm{score}(D,Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i,D) \cdot (k_1+1)}{f(q_i,D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\mathrm{avgdl}}\right)},$$

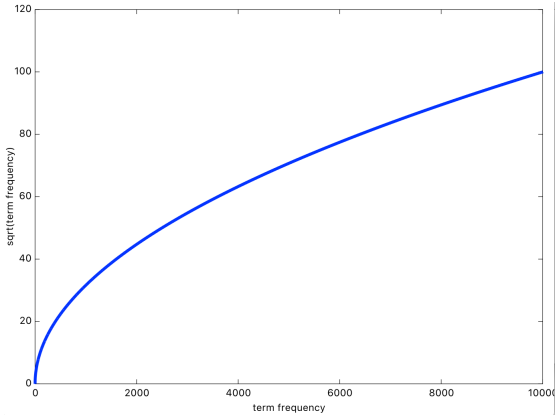


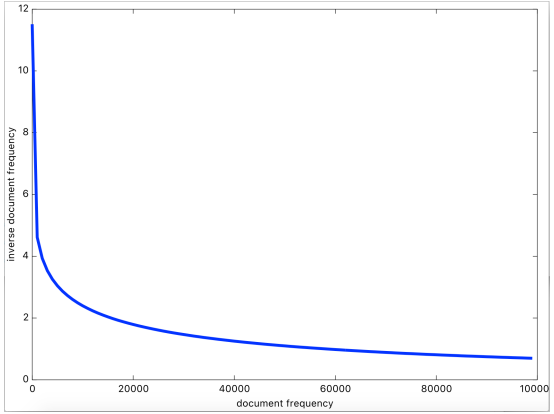
elastic

TF / IDF

Term Frequency / Inverse Document Frequency

- Term Frequency
  - 찾는 검색어가 문서에 많을수록 해당 문서의 정확도가 높습니다.
- Inverse Document Frequency
  - 전체 문서에서 많이 출현한 (흔한) 단어일수록 점수가 낮습니다.





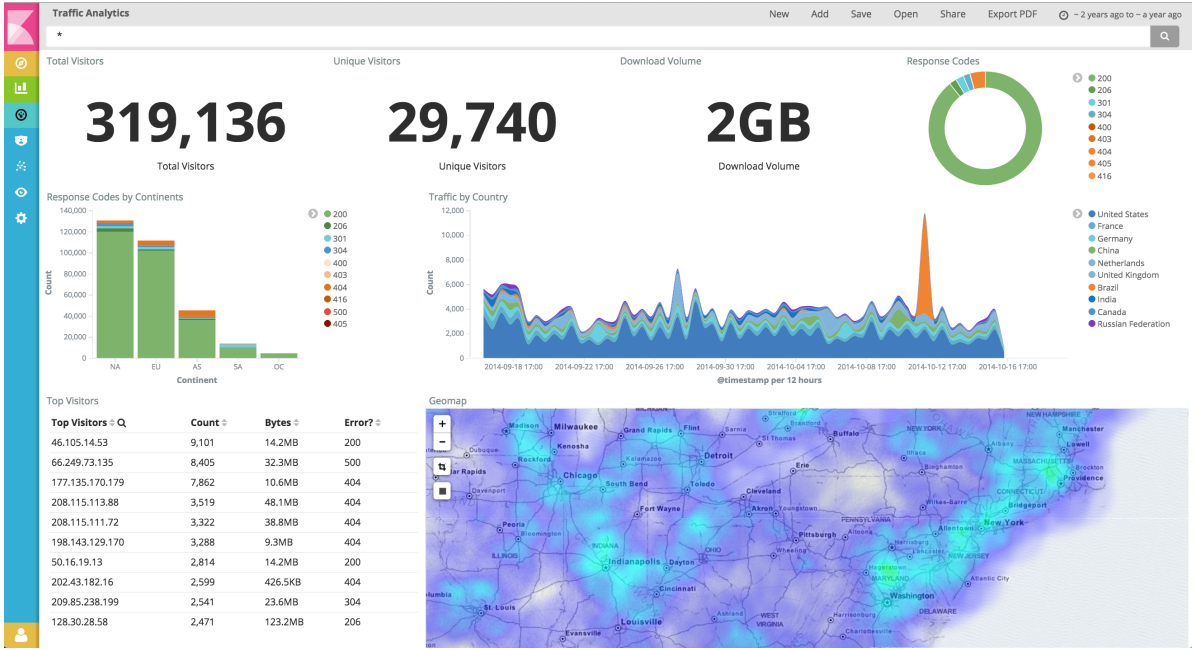
elastic

검색 랭킹이 중요한 이유

- 사용자들은 대부분 처음 나온 결과만 봅니다.
- 결과값이 큰 내용을 fetch 하는 것은 상당히 부하가 큽니다.
- 1~1,000 을 fetch 하는 것이나 990~1,000 을 fetch 하는 것이나 쿼리 작업 규모가 비슷합니다.
- 구글도 랭킹이 중요하긴 마찬가지…

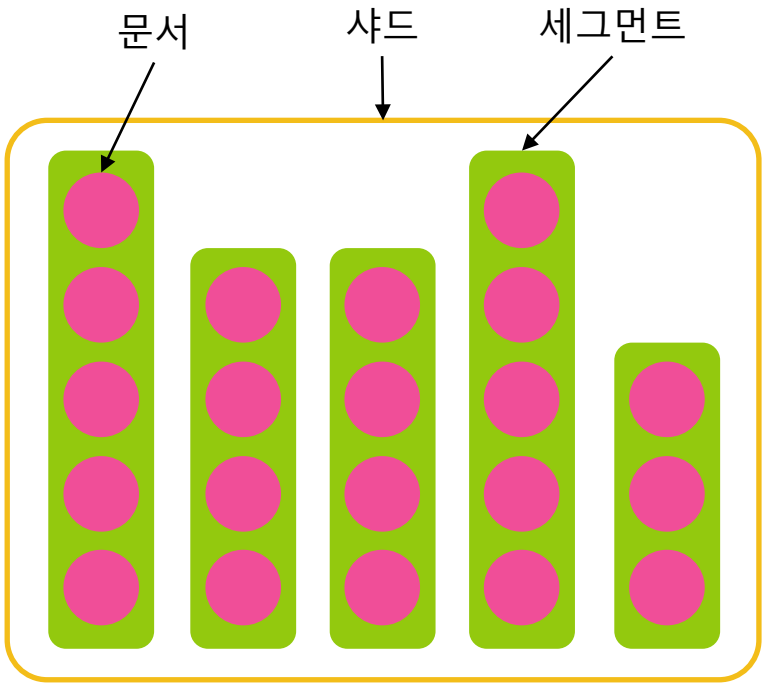
Elasticsearch is…

An open source, distributed, scalable, highly available, document-oriented, ***RESTful****, full text search engine with* ***real-time search and analytics*** *capabilities*



루씬 세그먼트(Segment)

Inverted Index, Doc Value, 원본 문서 등등을 저장하고 있는 단위 파일입니다.



- 루씬은 inverted index를 하나의 거대한 파일이 아니라 여러개의 작은 파일 단위로 저장합니다.
- 입력 버퍼가 가득 차거나, 1초 마다 하나씩 생성됩니다.
- **한번 생성된 세그먼트는 변경되지 않습니다. (immutable)**

elastic

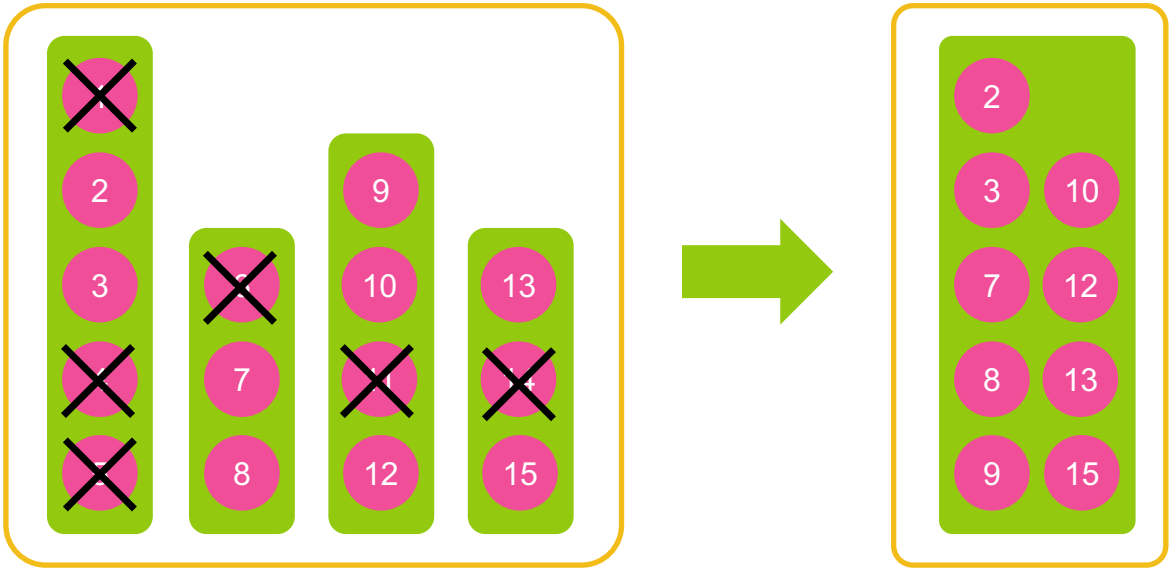
문서 변경/삭제 과정

한먼 생성된 세그먼트는 변경되지 않습니다.

- update는 없습니다. 모두 delete & insert 입니다.
- 문서를 삭제하면 삭제되었다는 상태만 표시하고 검색에서 제외합니다.
- 나중에 세그먼트 병합 과정에서 삭제된 문서를 빼고 나머지 문서들을 모아 새로운 세그먼트를 만듭니다.
  - 그래서 문서를 삭제 하더라도 세그먼트 병합을 하기 전 까지 스토리지 용량은 줄어들지 않습니다.
- 세그먼트 병합은 비용이 큰 동작입니다.
  - 디스크 I/O 작업입니다.
  - 시스템이 느려집니다.
  - 가능하면 사용자들이 적은 시간에 하는것이 좋습니다.

세그먼트 병합(Segment Merge)

http://blog.mikemccandless.com/2011/02/visualizing-lucenes-segment-merges.html

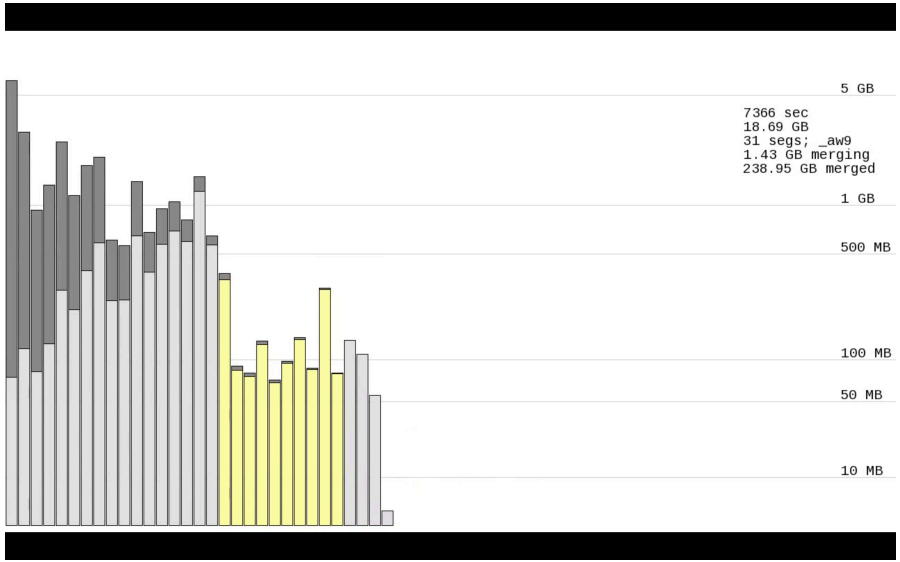


elastic

세그먼트 병합(Segment Merge)

http://blog.mikemccandless.com/2011/02/visualizing-lucenes-segment-merges.html

- 오래된 세그먼트는 크기가 크고 최근 생성된 세그먼트는 상대적으로 크기가 작습니다.
- 오래된 문서를 삭제하는 것은 더욱 비용이 큽니다.
- 날짜별로 저장 영역(인덱스)를 구분하는 것이 바람직합니다.
  - Elasticsearch 에서는 여러 인덱스를 묶어서 검색할 수 있는 멀티테넌시를 지원합니다.



elastic

Elasticsearch 사용하시려면

- 로그는 가능하면 날짜별로 나눠서 저장하세요.
- 원본 데이터는 항상 잘 가지고 계세요. 새로 부어야 하는 경우가 많습니다.
- 세그먼트 병합은 사용하지 않는 시간에.

감사합니다!!

김종민 – Community Engineer @Elastic
jongmin.kim@elastic.co
http://kimjmin.net
https://www.facebook.com/groups/elasticsearch.kr