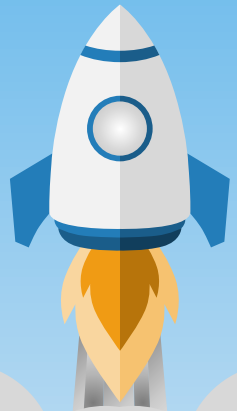


Elastic Search 서버 구축

주식회사 쿼리젯 김지훈

목 차

1 Elastic Search 개요

ElasticSearch 개요

Shay Banon이 Lucene을 바탕으로 개발한 분산 검색엔진

2010년 2월 첫 버전이 공개

Apache2 Licence

설치와 서버 확장이 매우 편리

아파치 루씬 기반

높은 가용성 (HIGH AVAILABILITY)

멀티 테넌시 (MULTY TENANCY)

JSON DOCUMENT / RESTFUL API

실시간 검색 (Near Real Time)

적용 사례

위메프

위메프
- 상품 검색
- 로그 분석
- 인기검색어, 연관검색어 추출

WIKIPEDIA
The Free Encyclopedia

위키피디아 (WIKIPEDIA)
- 전문검색(FULL TEXT SEARCH)를 수행
- 실시간 타이핑 검색
- 추천 검색어 기능

적용 사례

더 가디언 (THE GUARDIAN)
- 방문객의 로그 분석
- SOCIAL 데이터 생성 및 분석으로 실시간 응대
- 기사에 대한 반응 분석

스택 오버플로우 (STACK OVERFLOW)
- 검색내용과 결과를 통합해 유사한 질문과 해답을 연결

적용 사례

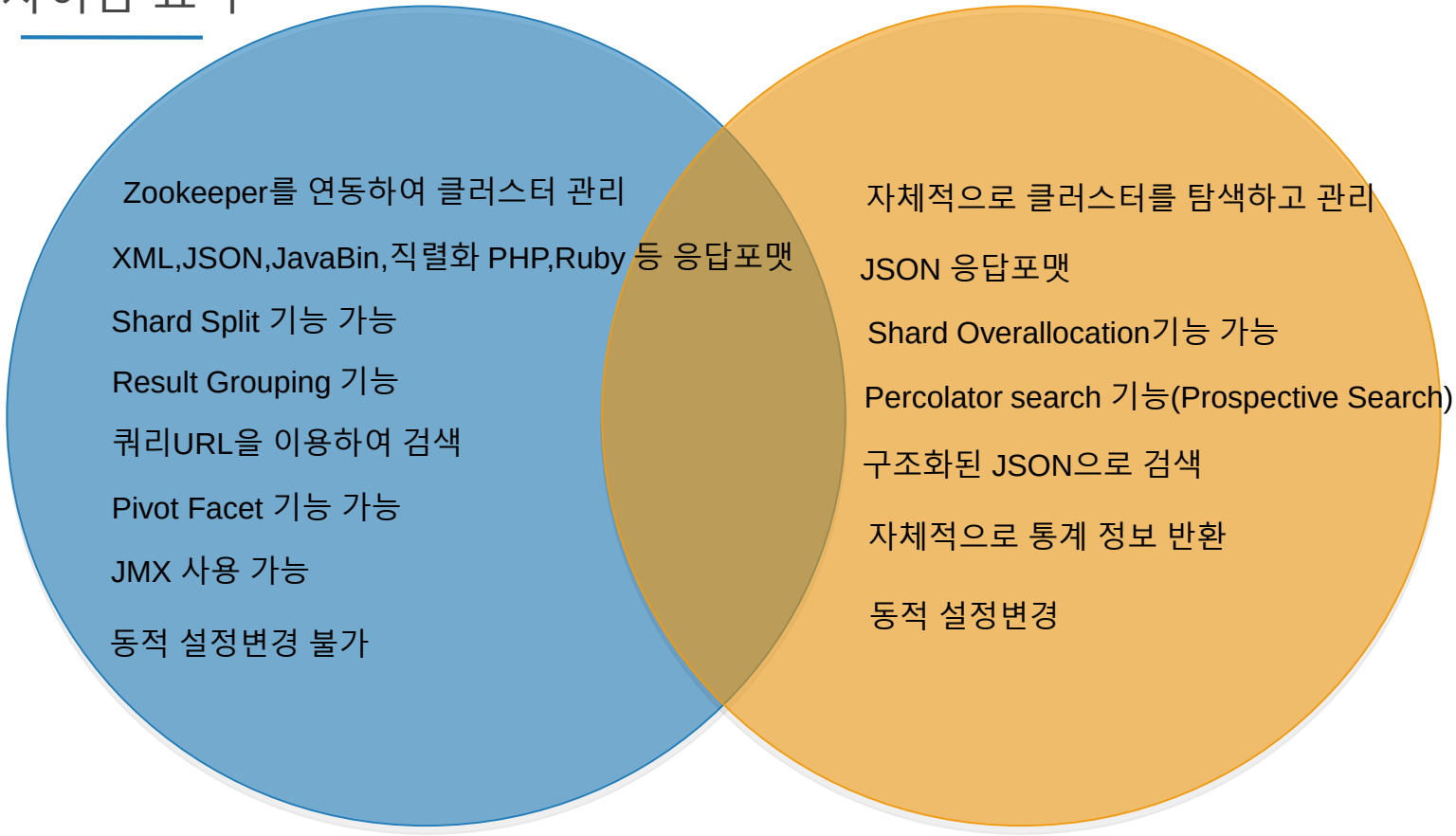
깃허브 (GITHUB)
- 1,300억 줄이 넘는 소스 코드를 검색하는 데 사용

골드만 삭스 (GOLDMAN SACHS)
- 매일 5TB가 넘는 데이터를 저장
- 주식 시장의 변동 분석에 사용

데이터베이스와 개념 비교

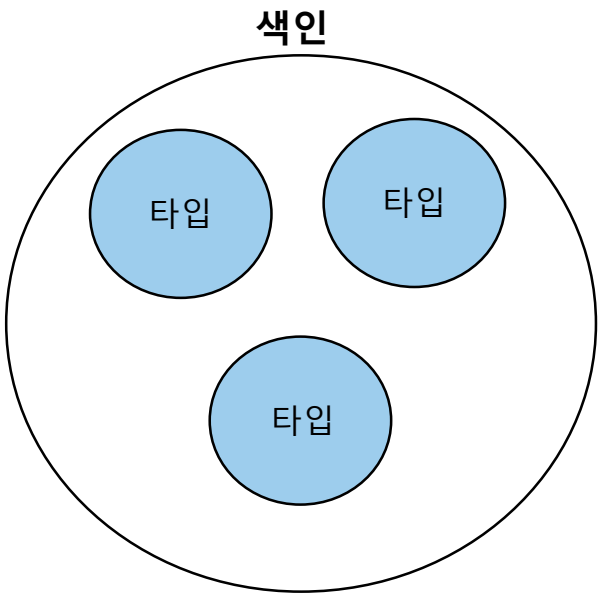
관계형 데이터베이스	elasticsearch	Solr
Database	Index	-
Table	Type	Collection
Row	Document	Document
Column	Field	Field
Schema	Mapping	Schema
Index	Everything is indexed	Everything is indexed
SQL	Query DSL	Query String,XML,JSON

차이점 요약


Zookeeper를 연동하여 클러스터 관리
XML,JSON,JavaBin,직렬화 PHP,Ruby 등 응답포맷
Shard Split 기능 가능
Result Grouping 기능
쿼리URL을 이용하여 검색
Pivot Facet 기능 가능
JMX 사용 가능
동적 설정변경 불가
자체적으로 클러스터를 탐색하고 관리
JSON 응답포맷
Shard Overallocation기능 가능
Percolator search 기능(Prospective Search)
구조화된 JSON으로 검색
자체적으로 통계 정보 반환
동적 설정변경

용어 정리 – 색인(index)이란?

검색엔진의 논리적 저장 단위

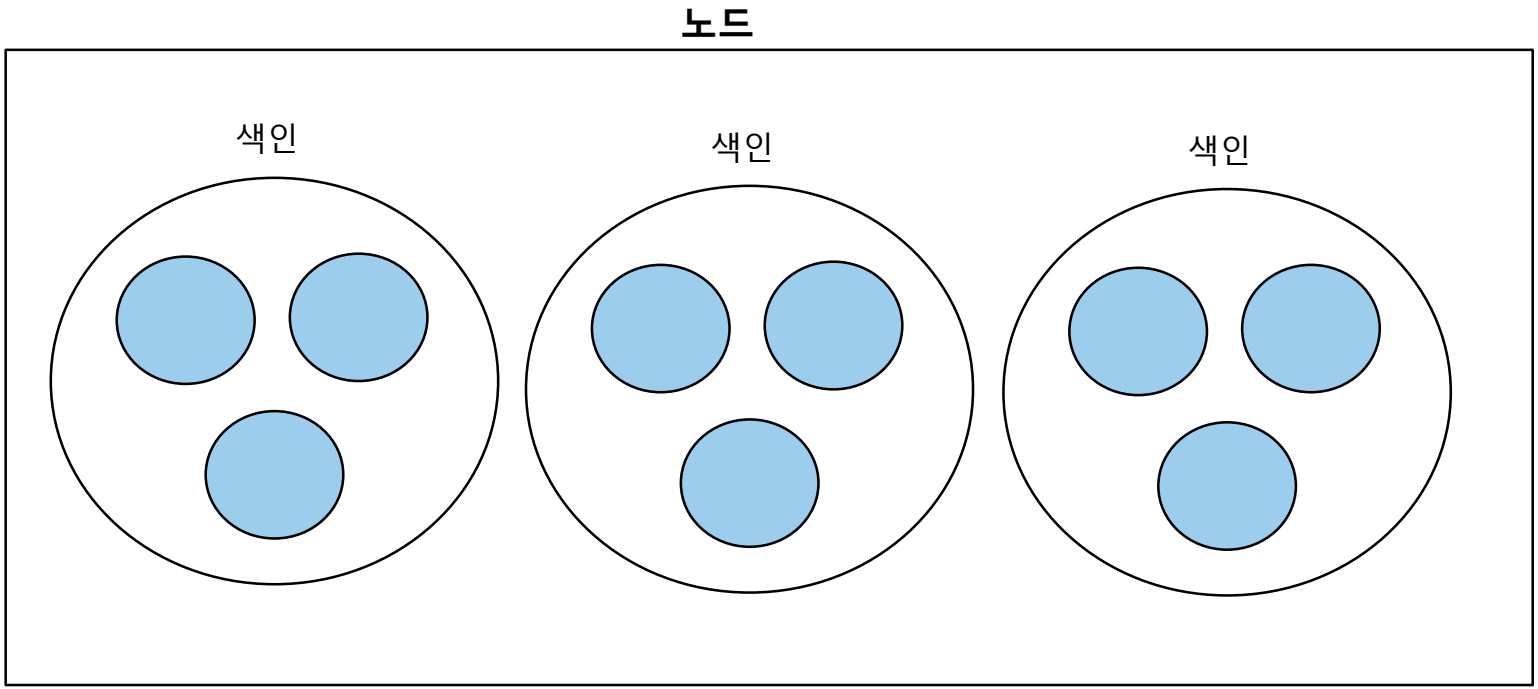



```
   367 2016-03-21 09:58 _0.cfe
391641 2016-03-21 09:58 _0.cfs
   256 2016-03-21 09:58 _0.si
   367 2016-03-21 09:58 _1.cfe
497025 2016-03-21 09:58 _1.cfs
   256 2016-03-21 09:58 _1.si
   367 2016-03-21 09:58 _2.cfe
385642 2016-03-21 09:58 _2.cfs
   256 2016-03-21 09:58 _2.si
   367 2016-03-21 09:59 _3.cfe
395658 2016-03-21 09:59 _3.cfs
   256 2016-03-21 09:59 _3.si
   367 2016-03-21 09:59 _4.cfe
421333 2016-03-21 09:59 _4.cfs
   256 2016-03-21 09:59 _4.si
   367 2016-03-21 09:59 _5.cfe
377152 2016-03-21 09:59 _5.cfs
   256 2016-03-21 09:59 _5.si
   367 2016-03-21 09:59 _6.cfe
399195 2016-03-21 09:59 _6.cfs
   256 2016-03-21 09:59 _6.si
   367 2016-03-21 09:59 _7.cfe
257893 2016-03-21 09:59 _7.cfs
   256 2016-03-21 09:59 _7.si
    36 2016-03-21 10:59 segments.gen
   471 2016-03-21 10:59 segments_3
     0 2016-03-21 09:58 write.lock
```

세그먼트

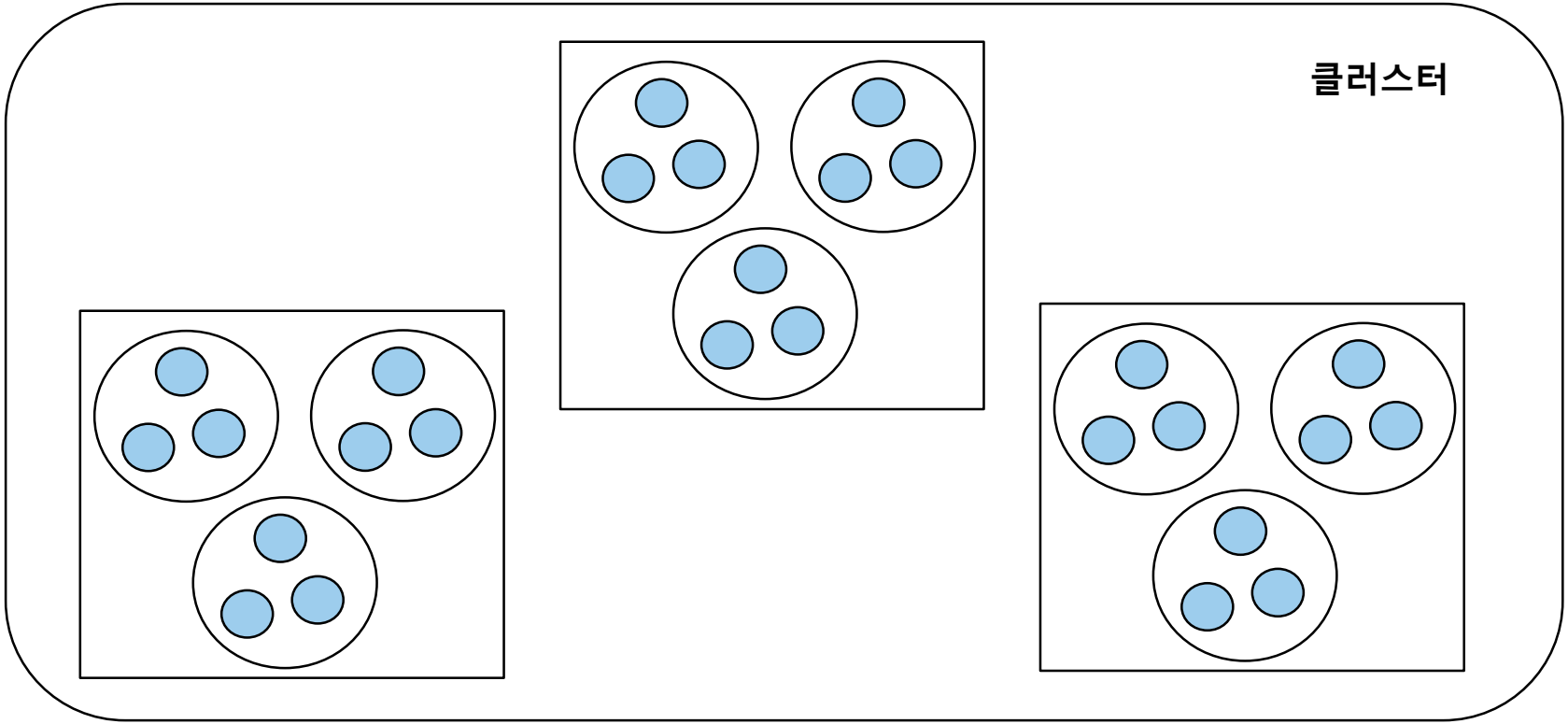
용어정리 – 노드란?

실행중인 엘라스틱 서치 인스턴스 하나를 지칭
통상 서버 1대를 노드로 봄.



용어정리 – 클러스터란?

- 엘라스틱서치의 가장 큰 시스템 단위
- 하나의 클러스터는 여러 개의 노드로 이루어



용어정리 – 샤드와 리플리카

분산 검색엔진에서 shard란 일종의 파티션과 같은 의미이며, 데이터를 저장할 때 나누어진 하나의 조각에 대한 단위

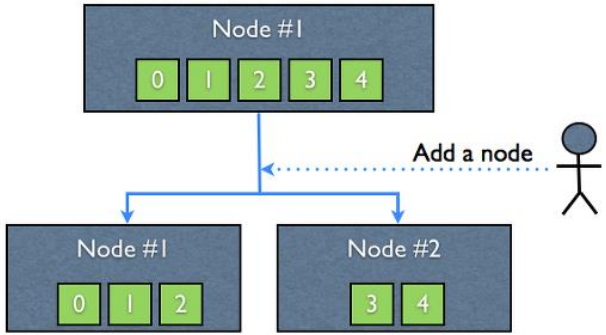
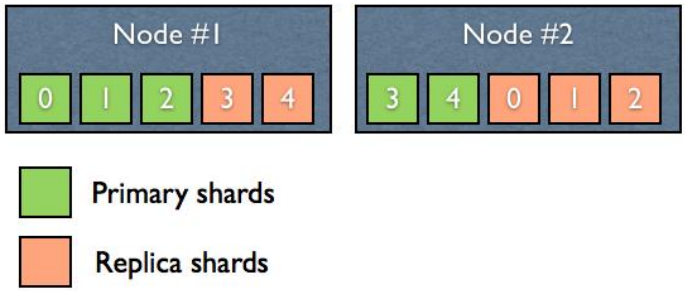


Figure 2. Shard relocating



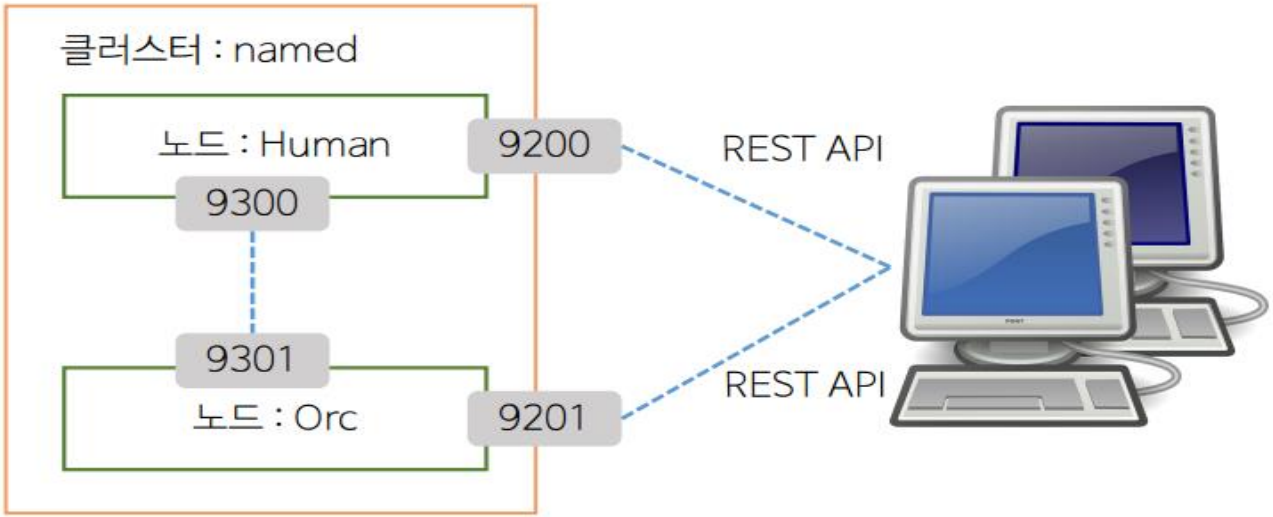
리플리카란 샤드의 복제본

리플리카는 운영중에 그 수를 변경할수 있으나 샤드는 수를 조정 할수 없다.

용어정리 - 노드 바인딩

같은 클러스터 이름을 가지고 실행된 노드는 자동으로 바인딩

- 9200번부터 REST API를 위한 HTTP 통신 포트가 할당
- 9300번 부터 노드간 바인딩을 위한 포트로 할당



용어 정리 - 마스터 노드와 데이터 노드

Data 노드

데이터가 저장되는 노드
인덱싱 및 검색 작업 수행
transport 모듈로 통신하기 때문에 http 모듈은 비활성화

설정 :
node.data : true
node.master : false

Non-Data 노드

마스터 전용 노드 (dedicated master node)
클러스터 - 노드 - 샤드의 맵핑 정보를 담고있는 노드
클러스터 관리 명령 수행하는 노드 (가벼움)
런타임 시 클러스터 멤버 중 마스터 노드로 적합한 노드 자동 선출
한 노드가 데이터 노드와 마스터 노드 역할을 모두 수행하면 클러스터 불안정성 증가
마스터 전용 노드에게는 인덱싱 및 검색 요청을 보내지 않음으로써 클러스터 불안정성 감소

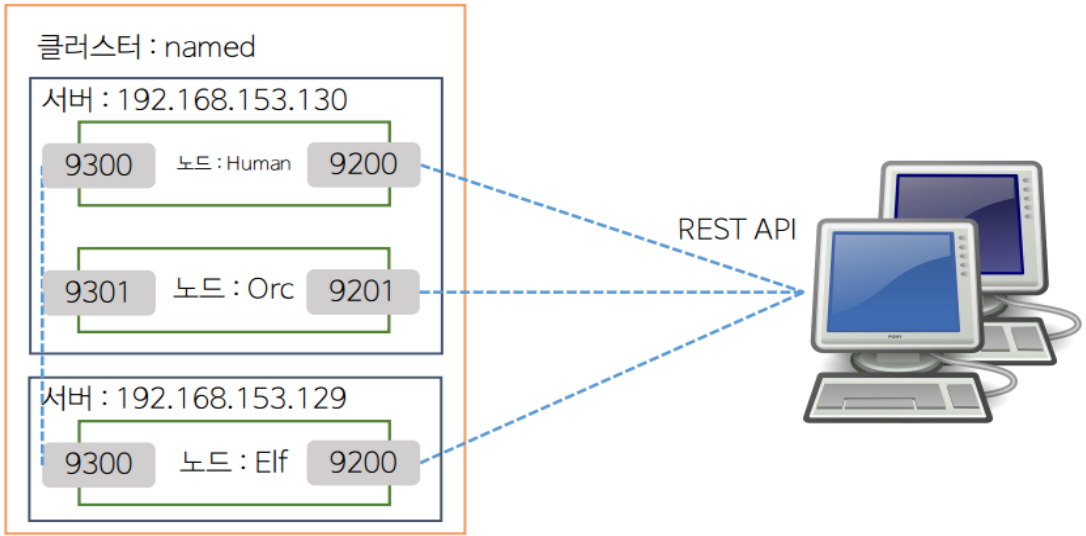
설정 :
node.data : false
node.master : true

클라이언트 노드 (client node)
노드들의 앞단에서 로드 밸런서 역할
HTTP 쿼리 파싱, 검색 요청에 대한 scatter/gatther, 네트워크 부하 담당
데이터 저장, 클러스터 관리 명령은 수행하지 않음
data node는 인덱싱 및 검색 작업에만 집중 가능

설정 :
node.data : false
node.master : false

용어정리 - 네트워크 바인딩

- 효율적인 스케일아웃을 위해 네트워크에 있는 다른 서버의 노드와도 바인딩 가능
- 네트워크 바인딩을 위해 젠 디스커버리(ZEN DISCOVERY) 기능 내장
- 멀티캐스트와 유니캐스트 방식을 모두 지원
- 공식 운영 그룹에서는 유니캐스트의 사용을 권장
- 반드시 두 엘라스틱서치의 버전은 동일해야 함



2 ElasticSearch 설치 및 설정

ElasticSearch 실행

설치하기
Quick Start

elasticsearch는 무설정 설치 가능

$ wget http://download.elasticsearch.org/elasticsearch/elasticsearch/elasticsearch-0.20.4.tar.gz
$ tar xvzf elasticsearch-0.20.4.tar.gz
서버 실행

설치 및 실행 명령
$ bin/elasticsearch -f
$ bin/elasticsearch -d (백그라운드 모드)

플러그인 설치

bin/plugin --install mobz/elasticsearch-head

설치된 플러그인 확인
bin/plugin –list

bin/plugin -install Aconex/elasticsearch-head 필수
bin/plugin -install lukas-vlcek/bigdesk 필수
bin/plugin -install elasticsearch/elasticsearch-transport-thrift/1.4.0
bin/plugin -url https://oss-es-plugins.s3.amazonaws.com/elasticsearch-jetty/elasticsearch-jetty-0.20.1.zip -install elasticsearch-jetty-0.20.1

주요 설정

confing/elasticsearch.yml 파일 (서버 기동전 수정)

1. 클러스터 이름 결정
cluster.name: elasticsearch

2. master 노드 및 data 노드여부 결정
node.master: true
node.data: fals

3. 샤드 및 리플라카 개수 설정
index.number_of_shards: 5
index.number_of_replicas: 1

4. 데이터 경로 설정
5. 필수 플러그인 설정
6. 메모리 설정
7. 네트워크 바인딩 주소 설정
8. 게이트웨이 설정

3 ElasticSearch CRUD

Elastic Search 데이터 처리

- 엘라스틱서치는 주로 REST API를 통하여 데이터를 처리
- HTTP METHOD는 GET, POST, PUT, DELETE, HEAD 등이 있음
- 엘라스틱서치의 REST API를 이용하기 위한 형태는 아래와 같음

curl –X{METHOD} http://host:port/{INDEX}/{TYPE}/{ID} –d '{DATA}'

HTTP METHOD, CRUD, SQL 비교

HTTP METHOD	CRUD	SQL
GET	READ	SELECT
PUT	UPDATE	UPDATE
POST	CREATE	INSERT
DELETE	DELETE	DELETE

REST Ful란?

REST는 웹의 창시자(HTTP) 중의 한 사람인 Roy Fielding의 2000년 논문에 의해서 소개
현재의 아키텍쳐가 웹의 본래 설계의 우수성을 많이 사용하지 못하고 있다고 판단
웹의 장점을 최대한 활용할 수 있는 네트워크 기반의 아키텍쳐를 소개
그것이 바로 Representational state transfer (REST)

REST는 요소로는 크게 리소스,메서드,메세지

HTTP POST , http://myweb/users/
{ Method Resource (URI)
"users":{
"name":"terry"
} Message
}

메서드	의미	Idempotent
POST	Create	No
GET	Select	Yes
PUT	Update	Yes
DELETE	Delete	Yes

CRUD를 메서드로 표현

데이터 입력(색인)

- POST나 PUT 메서드를 이용해 입력
- 처음 입력 시 created는 true
- 도큐먼트 ID를 생략하고 입력 가능
- 임의 ID 입력은 POST 메서드로만 가능

```
$ curl -XPUT http://localhost:9200/books/book/1 -d '
{
  "title": "Elasticsearch Guide",
  "author": "Kim",
  "date": "2015-06-25",
  "pages": 250
}'
```

데이터 삭제

- DELETE 메서드를 이용해 삭제
- 검색을 이용해 선별적 삭제 가능
- 타입과 인덱스 단위로 삭제 가능
- 인덱스 단위로 삭제할 경우 타입, 도큐먼트가 모두 삭제됨
- 도큐먼트는 삭제 후에도 메타 데이터를 그대로 보존

```
$ curl -XDELETE http://localhost:9200/books/book/AU4eDMKk5_2R8Icukymj
```

데이터 갱신

- PUT 메서드를 이용해 갱신
- 갱신 시 created는 false
- _version 필드가 증가
- 기존 도큐먼트는 보관되지 않으며 삭제되고 새로 쓰임
- 이전 버전으로 변경 불가능

```
$ curl -XPOST http://localhost:9200/books/book/ -d '
{
  "title": "Elasticsearch Guide",
  "author": ["Kim", "Lee"],
  "date": "2015-06-25",
  "pages": 300
}'
```

데이터 대량 색인

Solr 의 Dataimport 핸들러와 비슷한 river plugin
-Jdbc를 통해 DB 혹은 기타 데이터 소스에 직접 접근하여 색인

단, elasticsearch 2.0 이후 deprecated 됨. (logstash 이용 권장)

Bulk API

```
curl -XPOST localhost:9200/_bulk -d '
{ "index" : { "_index" : "books", "_type" : "book", "_id" : "1" } }
 { "title" : "Elasticsearch Guide", "author" : "Kim", "pages" : 250 }
 { "index" : { "_index" : "books", "_type" : "book", "_id" : "2" } }
 { "title" : "Elasticsearch Easy Guide", "author" : "Lee", "pages" : 300 } '
```

한글 형태소 분석기 세팅

한글형태소 분석기 플러그인을 설치후(아리랑 혹은 은전한닢)
bin/plugin -install chanil1218/elasticsearch-analysis-korean/1.1.0

```
$ curl -XPUT 'http://localhost:9200/books?pretty' -d '{
  "settings": {
    "index": {
      "analysis": {
        "analyzer": {
          "analysis-korean": {                    #— Analyzer 이름 지정
            "type": "org.elasticsearch.index.analysis.KoreanAnalyzerProvider",
            "tokenizer": "KoreanTokenizer",
            "filter": [ "trim", "lowercase", "KoreanFilter" ]
          }
        }
      }
    }
  }
}'
```

Mapping

특정 인덱스의 명시적 구조를 정의

```
$ curl -XPUT 'http://localhost:9200/books/book/_mapping' -d '
{
    "book" : {
        "properties" : {
            "projectName" : {"type" : "string", "index" : "not_analyzed"},
            “bookName" : {"type" : “string", "index" : "analyzed“,”index_analyzer”:” analysis-korean”},
            “bookContent" : {"type" : "string", "index" : "not_analyzed"},
            “regDate" : {"type" : "date"},
            "host" : {"type" : "string", "index" : "not_analyzed"},
            "body" : {"type" : "string"},
        }
    }
}
'

$ curl -XDELETE 'http://localhost:9200/books/book/_mapping'
```

4 ElasticSearch 검색

데이터 검색

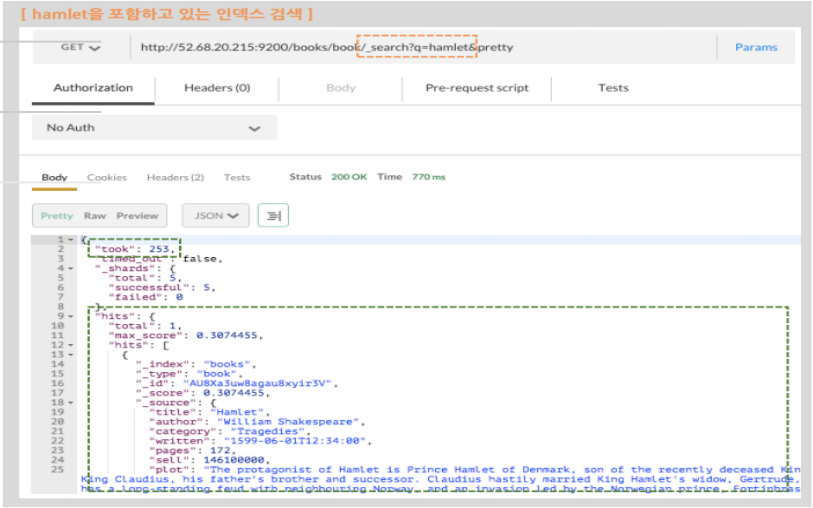
검색 기능은 질의(query) 명령어를 이용해 수행

- 문자열 형식의 매개변수를 이용한 URI 방식과
  HTTP 데이터를 이용한 REQUEST BODY 방식이 있음
- 타입, 인덱스 범위로 질의 가능
- 여러 개의 인덱스를 묶어서 멀티 인덱스 범위로 질의 가능(멀티 테넌시)

URI 요청을 사용한 검색 API

이름	설명
q	쿼리 스트링
default_operator	기본으로 사용할 연산자(AND 혹은 OR). 기본값은 OR.
fields	결과로 가져올 필드. 기본값은 '_source' 필드.
sort	정렬 방법(예: fieldName:asc/fieldName:desc)
timeout	검색 수행 타임아웃 값. 기본값은 무제한.
size	결과 값의 개수. 기본값은 10.

모든 검색 옵션을 제공하지는 않으므로 주로 테스트 용도로 간편하게 사용할 때 유용.

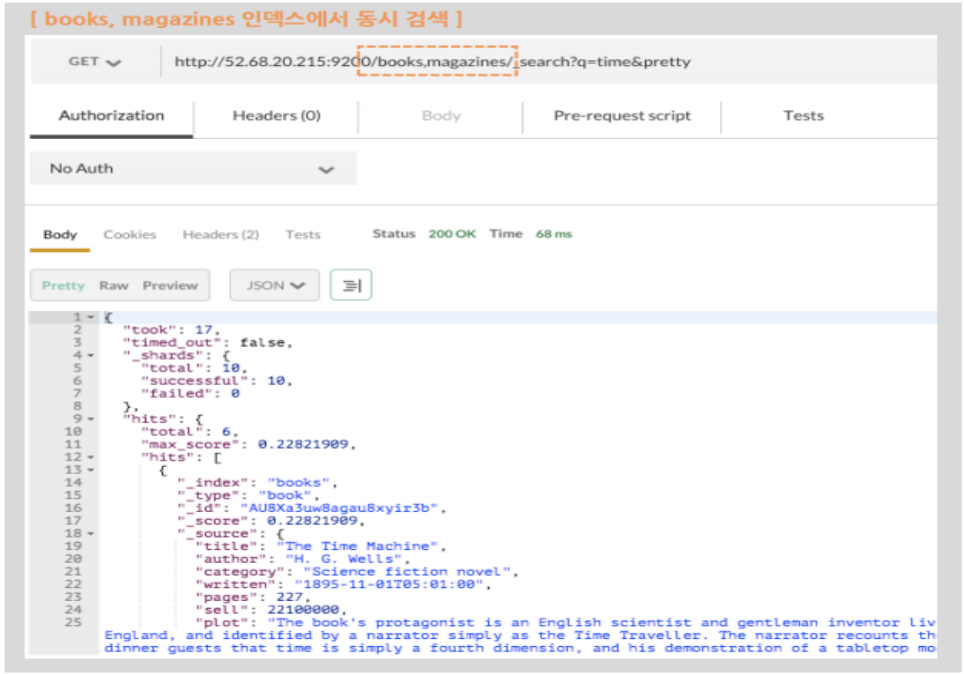


멀티 테넌시

- 여러 인덱스를 동시에 검색
- 인덱스들을 쉼표로 구분하여 입력
- URI와 REQUEST BODY 방식 모두 사용 가능

예)

```
curl -XGET 'http://localhost:9200/books/book1,b
/tomcat,apache/_search?q=host:host2'
```



검색 필드 지정

- 특정 필드만 지정하여 검색 가능
- 필드명 : 질의어 형태로 사용
- df 매개변수를 이용할 수 있음

[title 필드에서 time 검색]

GET http://52.68.20.215:9200/_search?q=title:time q=time&df=title

Authorization | Headers (0) | Body | Pre-request script | Tests

No Auth

Body | Cookies | Headers (2) | Tests | Status 200 OK | Time 101 ms

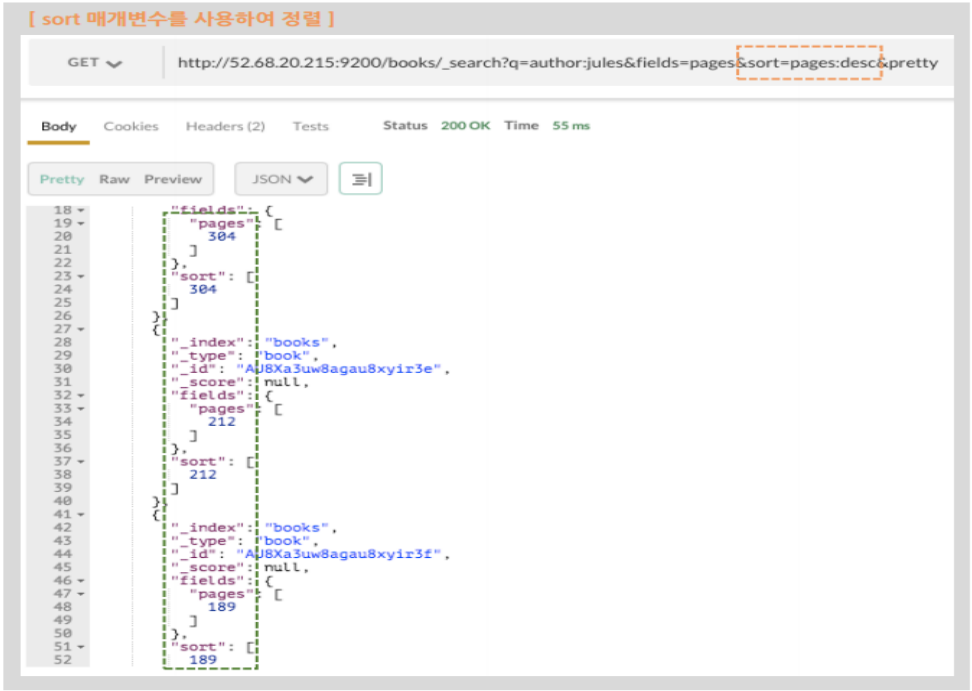
Pretty | Raw | Preview | JSON


```
{
  "took": 7,
  "timed_out": false,
  "_shards": {
    "total": 10,
    "successful": 10,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 0.70273256,
    "hits": [
      {
        "_index": "books",
        "_type": "book",
        "_id": "AU8Xa3uw8agau8xyir3b",
        "_score": 0.70273256,
        "_source": {
          "title": "The Time Machine",
          "author": "H. G. Wells",
          "category": "Science fiction novel",
          "written": "1895-11-01T05:01:00",
          "pages": 227,
          "sell": 22100000,
          "plot": "The book's protagonist is an English scientist and gentleman inventor livi
England, and identified by a narrator simply as the Time Traveller. The narrator recounts the
dinner guests that time is simply a fourth dimension, and his demonstration of a tabletop mod
```

정렬

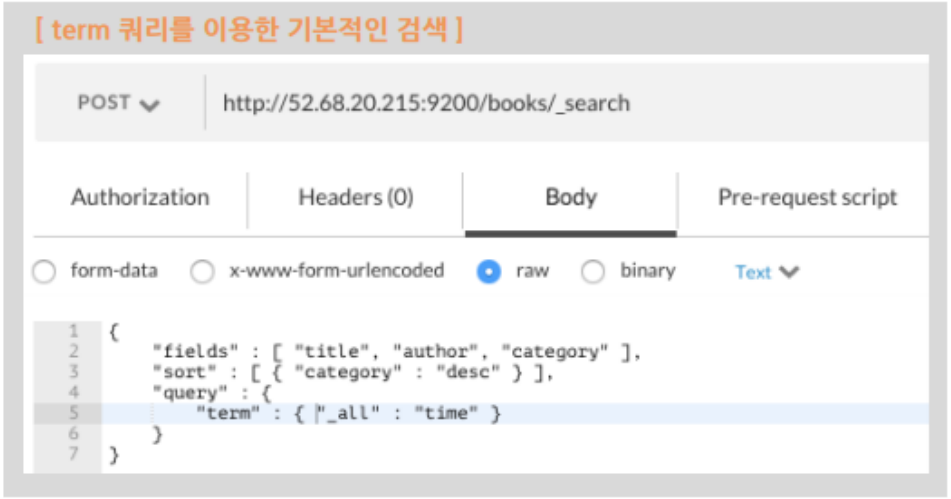
- sort 매개변수를 사용
- 기본적으로 결과는 점수 값을 기준으로 정렬
- 기본적인 정렬 방식은 오름차순
- 검색 정렬 기준을 변경
- sort={필드명}:{정렬방식} 형태로 지정

[sort 매개변수를 사용하여 정렬]



REQUEST BODY 검색

- 검색 조건을 JSON 데이터 형식의 질의로 입력
- URI 검색보다 복잡한 형식으로 검색 가능
- 엘라스틱서치의 질의 언어(QueryDSL) 사용




```
# 요청
$ curl -XPOST
'http://localhost:9200/books/book/_search' -d '{
    "query" : {
        "term" : { "host" : "host2" }
    }
}
```

HIGHLIGHT

- Highlight 옵션을 사용하여 설정
- 기본적으로 *태그를 사용*
- *pre/post_tags 옵션으로 태그 변경 가능*

[문자열 time을 검색하여 highlight 처리]

POST http://52.68.20.215:9200/books/_search

Authorization | Headers (0) | Body | Pre-request script

form-data | x-www-form-urlencoded | raw | binary | Text

```
{
    "fields" : [ "title", "author", "category" ],
    "sort" : [ { "category" : "desc" } ],
    "query" : {
        "term" : { "_all" : "time" }
    },
    "highlight" : {
        "pre_tags" : [ "<strong>" ],
        "post_tags" : [ "</strong>" ],
        "fields" : { "title" : {} }
    }
}
```

```
        "_index": "books",
        "_type": "book",
        "_id": "AU8Xa3uw8agau8xyir3b",
        "_score": null,
        "fields": {
          "author": [
            "H. G. Wells"
          ],
          "category": [
            "Science fiction novel"
          ],
          "title": [
            "The Time Machine"
          ]
        },
        "highlight": {
          "title": [
            "The <strong>Time</strong> Machine"
          ]
        },
        "sort": [
          "science"
        ]
      },
```

루씬의 기본 검색 스코어

최종점수 score(q,d) = coord(q,d) · queryNorm(q)
· ∑ (tf(t in d) · idf(t)^2 · **텀가중치** · lengthNorm(t,d))
+ **사용자정의가중치**

q : query
d : document
t : term(색인어)

lengthNorm(t,d) : 문서의 길이에 대한 가중치 (짧을 수록 유리)
1/sqrt(문서의텀의갯수) * **필드가중치** * **문서가중치**

변수로서 조정 대상이 되는 팩터

1) 텀가중치 : 특정한 텀이 일치 되었을시 가중치 부여 (검색 질의시 결정)
예) 현대^10 카드 (현대라는 단어와 카드라는 단어의 검색어가 있을 경우 현대라는 단어의 일치를 10배 가중치를 부여함)

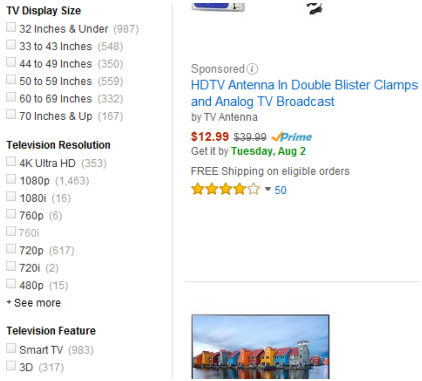
2) 필드가중치 : 특정필드에 일치 되었을시 가중치 부여 (검색 질의시 결정)
예) title:"현대"^10 OR content:"현대" (title필드에서 일치한 문서에 대해서 10배의 가중치를 부여함)

3) 문서가중치 : 특정문서에 대한 중요도(가중치) 부여 (색인 생성시 결정)
문서1.setBoost(10)
문서2.setBoost(5)
문서3.setBoost(0)

문서1에 대해서 10배의 가중치를 부여함.

기타 주요 고급 검색기능

Facet
- Term Facet
- Range Facet
- Date Facet
- Statistical Facet
- geo distance Facet



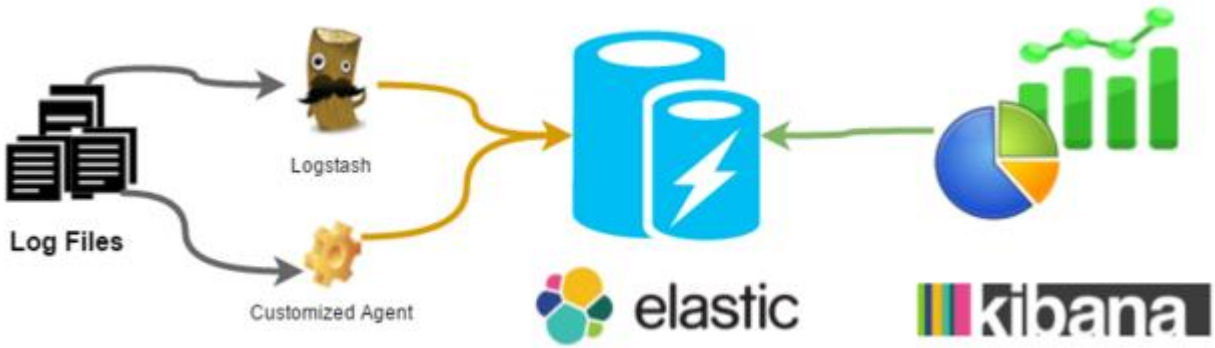
Aggregations
버켓 단위의 도큐먼트 셋에 대해서 지속적인 추적과 연산 수행
-최소(min),최대(max),합(sum),평균(avg),개수(value count)

- Term aggregations
- Range aggregations
- Data aggregations

aggregation은 “facet 의 재 탄생” 입니다.

5 ELK (ElasticSearch & Logstash & Kibana)

ELK란?



실시간 로그 분석 기술 스택

Solr 진영에는 SILK와 BANANA (Solr + Logstash + Kibana)

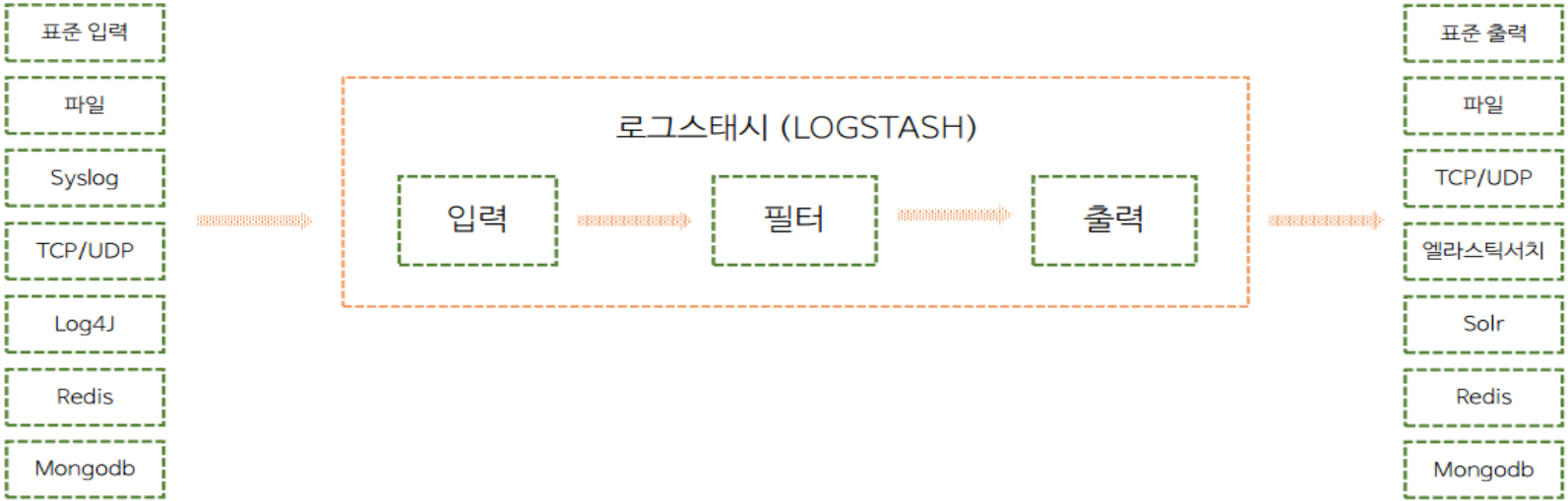
Logstash

- 데이터의 흐름을 관리하기 위해 개발된 오픈 소스 프로젝트
- 엘라스틱서치의 공식 패키지 구성 요소
- 아파치 라이선스 2.0 오픈소스
- JRuby로 작성 (자바 런타임 환경 필수, 1.7 이상)
- 다양한 방식으로 데이터 입/출력 가능

비슷한 기능의

Facebook의 스크라이브(Scribe)
아파치 프로젝트 Flume

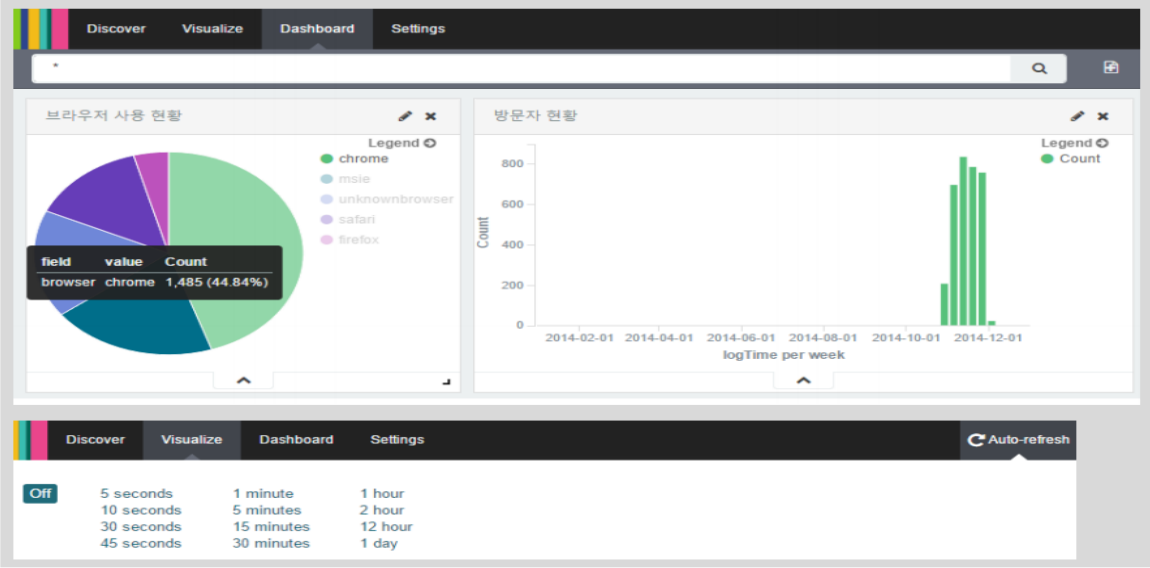
Logstash



입력 : 다양한 경로로 부터 데이터를 읽어오는 작업
- 필터 : 읽어온 데이터를 가공하는 절차
- 출력 : 가공된 데이터를 다른 프로그램이나 채널로 입력

Kibana

- 데이터 분석 시각화 도구
- 엘라스틱서치의 복잡한 질의를 편하게 입력 가능
- 입력된 질의를 간편하게 시각화
- config.js 파일을 수정하여 간편하게 설정
- node.js로 작성
- 일부 설정은 엘라스틱서치 인덱스에 저장



QnA

ElasticSearch 활용 실습

- 실습 데이터 : 뉴스 데이터 10만
- 아파치 로그 데이터
- ElasticSearch 0.20.4
- kibana 4
- 3 Node 구조
- CRUD 작동 확인
- Mapping 설정
- Logstash 를 통한 JDBC 연결
- 데이터 색인
- 검색기능 확인
- kibana 동작 확인

감사합니다.