

Introduction to Elasticsearch

'20 kidevelop & ivy

SPARCS new-ara team

new-ara and elasticsearch: why need it?

8월 15일자 에브리타임 게시글.

“검색이 너무 느려”

자유게시판



익명

08/15 17:59

쪽지 신고

근데

지금 이라 디자인 그렇게 이상한건 아닌것같은데.
깔끔하고 괜찮아 보이는데 제가 이상한건가

새내기라서 아라 한번도 들가본적 없고
“아리”글 사진만 봤어요

0 0 0 3 ☆ 0



익명1

대댓글 공감 쪽지 신고

메블메 깔립만한데 접속시간이 너무 오래걸리고 불안정해서 성능때문에 더 구려보일수도 있음

08/15 18:03 🔗 1

공감 쪽지 신고



익명4

성능 0 ⚡ 검색이 너무 느려

08/15 18:04

대댓글 공감 쪽지 신고



익명5

들어가봐

08/15 18:49 🔗 2

대댓글 공감 쪽지 신고

댓글을 입력하세요.

익명



글 목록

new-ara and elasticsearch: why need it?

9월 3일자 밤부 게시글.

“검색에 한세월 걸리는거부터”

잡담 아라는 사이트가 안돌아가네요;;

익명 | 6일 전 최근 코멘트로 넘어가기 +0 / -0 / 조회 175

아라는 사이트 개편 의지가 없는걸까요
사이트가 안돌아감.. 페이지가 안바뀜ㅠㅠ
아라에도 종종 유용한 경보가 올라오던거같던데..아쉽네

1. 익명1 | 6일 전 +0 / -0

너무 오래 방치되어어서 ㅋㅋ 밤부도 슬슬 그럴긴 한데.. 익명의 특성으로 어떻게든 굴러가는듯

2. 익명2 | 6일 전 +3 / -0

아라는 검색에 한세월 걸리는거부터 사용의지 뚝떨어짐

new-ara and elasticsearch: why need it?

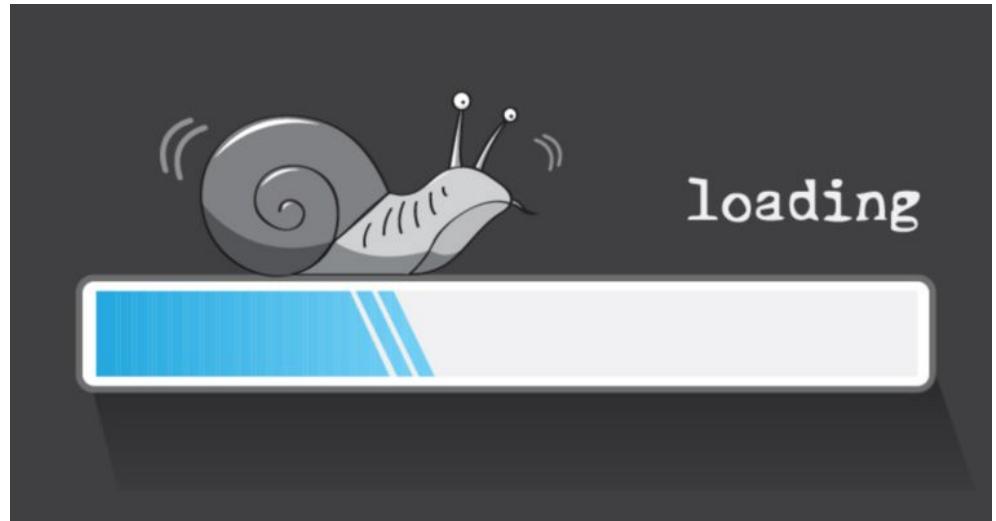
“legacy ara를 사용하지 않는 이유”에
대해,

“속도가 느리다”

“검색하는 데 하루종일 걸린다”

“사람이 없다”

등의 반응이 많습니다.



elasticsearch: a super-fast search db

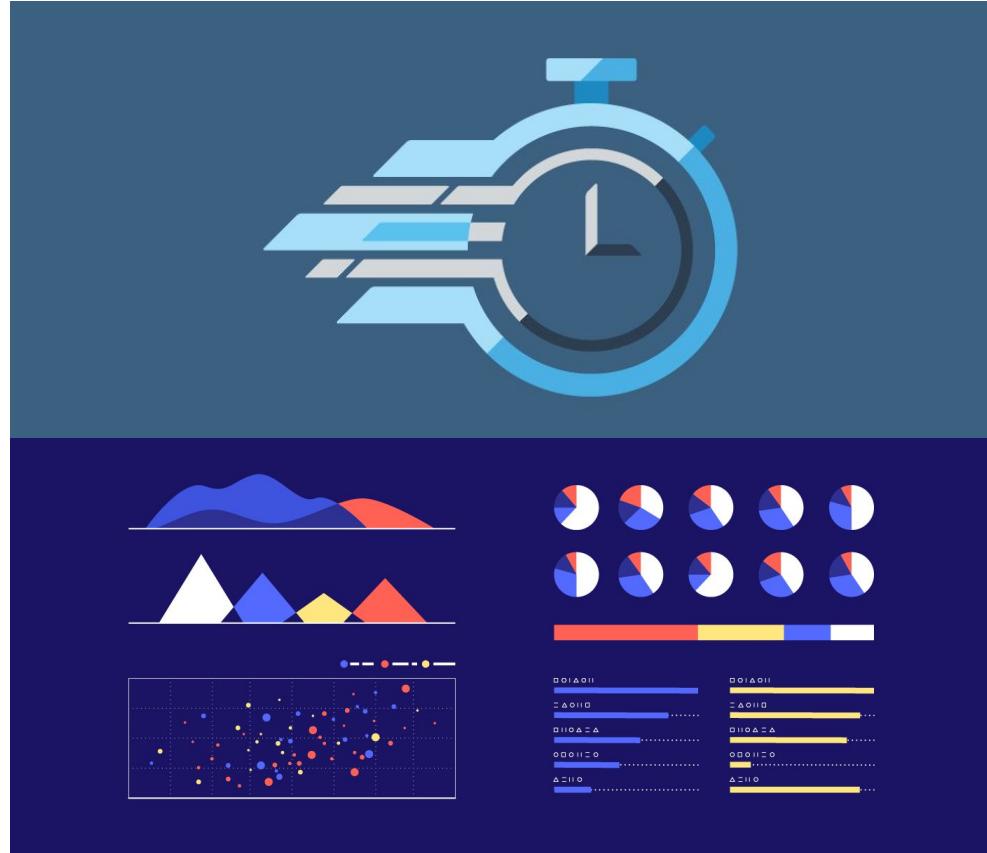
elasticsearch는 검색 기능을 제공하는
별도의 DB로, 많은 양의 데이터에 대한
매우 빠른 검색을 지원합니다.



new-era and elasticsearch: expected effect

하나, 검색 속도 향상

둘, 서버 log 검색과 데이터 시각화



elasticsearch: NoSQL db

elasticsearch는 검색에 최적화된 NoSQL 데이터베이스이며, JVM 위에서 작동합니다.

오른쪽과 같이 관계형 DB와는 다른 용어를 사용합니다.

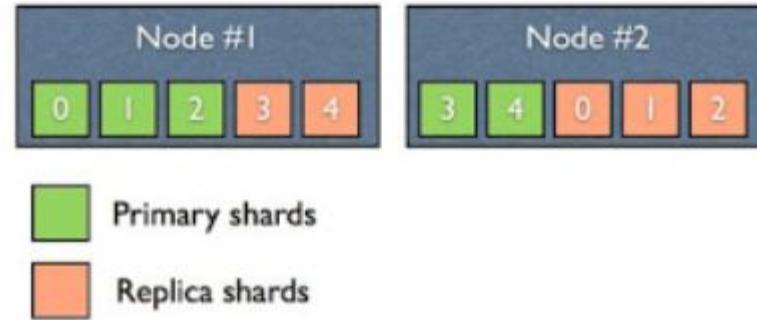
특히, 여러 Index에서 동시에 검색하는 multi-tenancy를 지원합니다.

관계형 데이터베이스	elasticsearch
Database	Index
Table	Type
Row	Document
Column	Field
Schema	Mapping
Index	Everything is indexed
SQL	Query DSL

elasticsearch: distributed processing

elasticsearch는 여러 node 위에서 동작할 수 있습니다. node는 각각의 서버에서 돌아가는 elasticsearch 프로세스를 말합니다.

한편 shard는 데이터가 저장되는 파티션입니다. primary shard는 원본이고, replica shard는 다른 node에 있는 데이터 복제본입니다.



elasticsearch: distributed processing

shard들은 자동적으로 node들 사이에
재분산되어, 검색 성능을 유지하는 동시에
한 node가 장애를 일으킨 경우에도 검색이
가능하도록 해 줍니다.

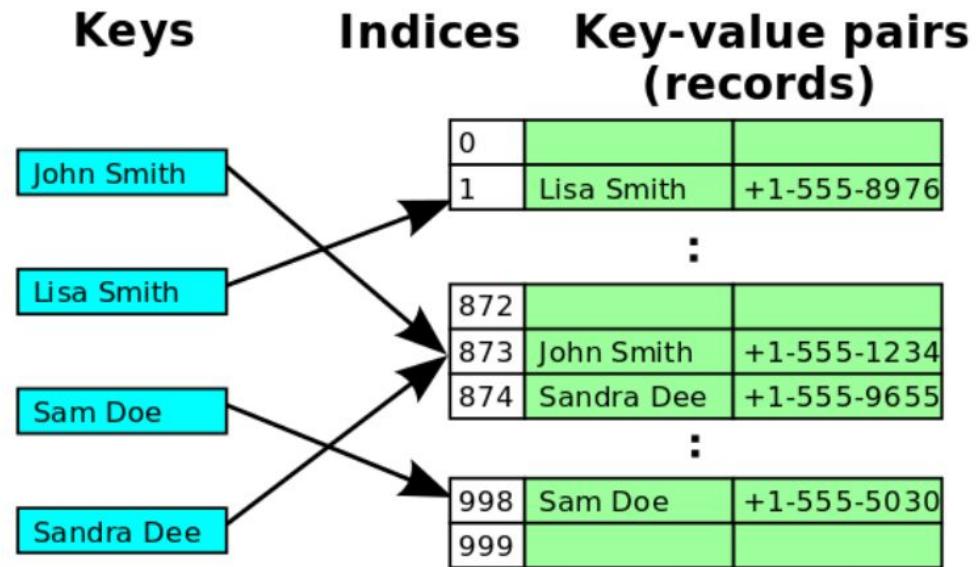


elasticsearch: super-fast search

elasticsearch에서 매우 빠른 검색이 가능한 이유는,

새 Document가 들어올 때마다 각 단어를 분리, hashing하여 단어가 어디서 등장했는지에 대한 hash table을 만들기 때문입니다.

검색 시에는 단어를 hash table에서 조회하는 것으로 O(1) 검색이 가능합니다.



elasticsearch: RESTful API

elasticsearch는 자체 REST API를 제공합니다.

<https://d2.naver.com/helloworld/273788>

The diagram illustrates a PUT request to Elasticsearch. The URL is `PUT /customer/external/1?pretty`. Red annotations point to specific parts of the URL and the JSON payload:

- `/customer/external` is labeled `index`.
- `/1` is labeled `type`.
- `?pretty` is labeled `_id`.
- The JSON object `{"name": "John Doe"}` is labeled `document`.

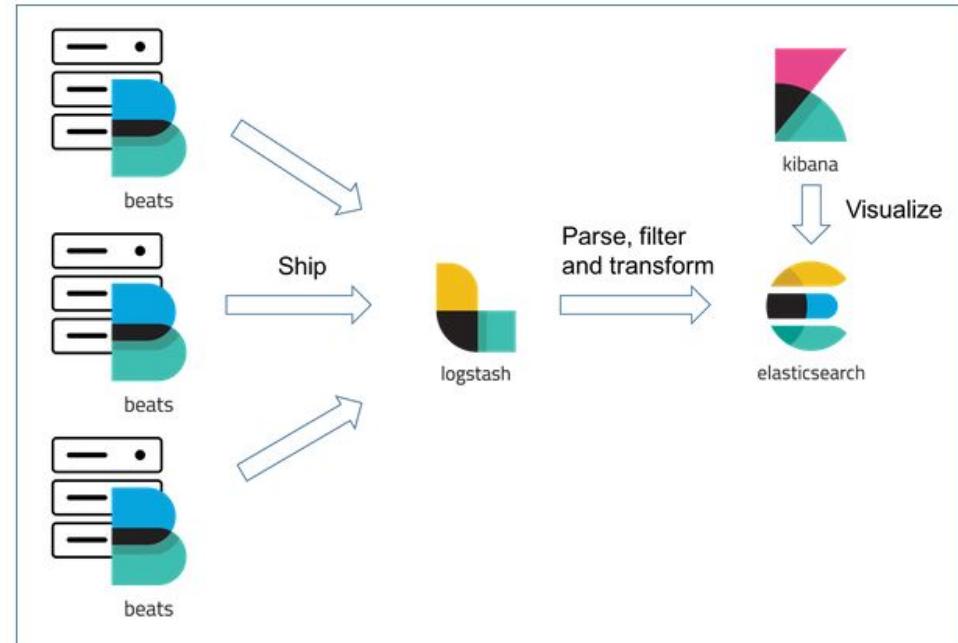
```
PUT /customer/external/1?pretty
{
  "name": "John Doe"
}
```

the elastic stack: gather, search, visualize

Elastic stack (a.k.a. ELK stack)은

수집하는 beats, logstash
처리하고 검색하는 elasticsearch
시각화하는 kibana

로 이루어져 정보를 다루는 스택입니다.



elasticsearch: with Python

elasticsearch를 Python에서 이용하기
위한 방법으로는 크게,

elasticsearch-py (저수준)
elasticsearch-dsl (고수준)

이 존재합니다.



The screenshot shows the PyPI project page for 'elasticsearch' version 7.9.1. At the top right, there's a green button with a checkmark labeled 'Latest version' and a note 'Released: Aug 21, 2020'. Below the header, it says 'Python client for Elasticsearch'. On the left, there's a 'Navigation' sidebar with 'Project description' (which is currently selected and highlighted in blue), 'Release history', and 'Download files'. On the right, under 'Project description', it says: 'Official low-level client for Elasticsearch. Its goal is to provide common ground for all Elasticsearch-related code in Python; because of this it tries to be opinion-free and very extendable.' There's also a 'Installation' section.



The screenshot shows the PyPI project page for 'elasticsearch-dsl' version 7.2.1. At the top right, there's a green button with a checkmark labeled 'Latest version' and a note 'Released: Jun 3, 2020'. Below the header, it says 'Python client for Elasticsearch'. On the left, there's a 'Navigation' sidebar with 'Project description' (selected), 'Release history', and 'Download files'. On the right, under 'Project description', it says: 'Elasticsearch DSL is a high-level library whose aim is to help with writing and running queries against Elasticsearch. It is built on top of the official low-level client ([elasticsearch-py](#)).'. It continues: 'It provides a more convenient and idiomatic way to write and manipulate queries. It stays close to the Elasticsearch JSON DSL, mirroring its terminology and structure. It exposes the whole range of the DSL from Python either directly using defined classes or a queryset-like expressions.' There's also a 'Project links' section with a 'Homepage' link.

elasticsearch: resource usage

elasticsearch는 hash table을 메모리에
계속 올리려고 하기 때문에,

RAM 크기 및 swap memory 속도를 많이
요구하게 됩니다.

즉, 사양을 많이 탑니다.

```
...
1 [||||| 79.9%] 5 [||||| 79.2%] 9 [||||| 76.4%] 13 [||||| 77.6%]
2 [||||| 17.0%] 6 [||||| 15.9%] 10 [||||| 15.6%] 14 [||||| 13.1%]
3 [||||| 78.5%] 7 [||||| 78.5%] 11 [||||| 77.9%] 15 [||||| 77.9%]
4 [||||| 15.9%] 8 [||||| 15.5%] 12 [||||| 14.2%] 16 [||||| 13.1%]
Mem[||||| 15.7G 32.0%] Tasks: 558, 2535 thr; 5 running
Swap[||||| 58.0M 1.00%] Load average: 6.88 4.25 3.30
Uptime: 2 days, 03:51:35

PID USER PRI NI VIRT RES S CPU% MEM% TIME+ Command
87382 kiddevelop 17 5 9309M 1492M ? 45.7 4.6 0:58.01 /Users/kidevelop/Elastic/elasticsearch-7.9.1/jdk.app/Contents/Home/bin/java -Xshare:auto
659 kiddevelop 16 9 11.8G 1442M ? 0.2 4.4 24:04.21 /Applications/Kite.app/Contents/MacOS/Kite --system-boot
85396 kiddevelop 24 0 11.3G 958M ? 29.0 2.9 15:20.25 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
66372 kiddevelop 8 0 8049M 923M ? 59.9 2.8 56:53.50 /Applications/Google Chrome.app/Contents/MacOS/Google Chrome
87543 kiddevelop 17 0 11.3G 798M ? 111. 2.4 1:18.20 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
83867 kiddevelop 17 0 11.2G 751M ? 0.0 2.3 0:50.52 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
82522 kiddevelop 24 0 11.1G 615M ? 0.0 1.9 0:50.78 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
86332 kiddevelop 24 0 11.8G 502M ? 0.1 1.5 1:54.18 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
776 kiddevelop 24 0 8197M 479M ? 0.1 1.5 38:57.74 /Applications/KakaoTalk.app/Contents/MacOS/KakaoTalk
66385 kiddevelop 17 0 8041M 451M ? 5.5 1.4 1h17:57 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
66408 kiddevelop 17 0 10.7G 422M ? 0.0 1.3 2:19.94 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
68745 kiddevelop 17 0 9329M 390M ? 0.0 1.2 9:55.37 /Applications/Keynote.app/Contents/MacOS/Keynote
83858 kiddevelop 17 0 14.8G 375M ? 0.0 1.1 0:21.28 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
92255 kiddevelop 17 0 18.9G 366M ? 30.0 1.1 0:23.13 /Applications/Notion.app/Contents/Frameworks/Notion Helper (Renderer).app/Contents/MacOS/Notion Helper (Renderer)
83771 kiddevelop 17 0 7527M 351M ? 0.9 0.1 0:41.71 /Applications/Notion.app/Contents/Frameworks/Notion Helper (Renderer).app/Contents/MacOS/Notion Helper (Renderer)
64805 kiddevelop 24 0 10.7G 342M ? 13.3 1.0 7:59.75 /Applications/Adobe Photoshop 2020.app/Contents/MacOS/Adobe Photoshop 2020
94103 kiddevelop 24 0 18.4G 326M ? 89.5 1.0 0:32.14 /Applications/Visual Studio Code.app/Contents/Frameworks/Code Helper (Renderer).app/Contents/MacOS/Code Helper (Renderer)
62468 kiddevelop 17 0 11.8G 311M ? 0.1 0.9 19:47.19 /Applications/Slack.app/Contents/Frameworks/Slack Helper (Renderer).app/Contents/MacOS/Slack Helper (Renderer)
6196 kiddevelop 17 0 6932M 298M ? 0.3 0.9 10:27.26 /usr/local/Cellar/mysql/8.0.21_1/bin/mysqld --basedir=/usr/local/Cellar/mysql/8.0.21_1
66386 kiddevelop 17 0 6790M 227M ? 40.4 0.7 13:07.01 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
83038 kiddevelop 17 0 14.9G 219M ? 0.0 0.7 0:09.17 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
463 kiddevelop 17 0 7404M 211M ? 0.1 0.6 3:34.87 /System/Library/CoreServices/Finder.app/Contents/MacOS/Finder
86949 kiddevelop 17 0 7088M 210M ? 5.5 0.6 0:07.39 /Applications/Hyper.app/Contents/Frameworks/Hyper Helper.app/Contents/MacOS/Hyper Helper
545 kiddevelop 17 0 11.2G 201M ? 0.1 0.6 6:11.91 com.mapcav.CleanMyMac4.Menu
1080 kiddevelop 17 0 7600M 198M ? 1.1 0.6 14:48.95 /Applications/Visual Studio Code.app/Contents/MacOS/Electron
64823 kiddevelop 17 0 6813M 188M ? 0.0 0.6 0:08.01 /Applications/Adobe Photoshop 2020.app/Contents/MacOS/CEPHtmlEngi
64821 kiddevelop 17 0 6875M 186M ? 0.0 0.6 0:09.96 /Applications/Adobe Photoshop 2020.app/Contents/MacOS/CEPHtmlEngi
87541 kiddevelop 17 0 10.9G 184M ? 0.0 0.6 0:05.02 /Applications/Google Chrome.app/Contents/Frameworks/Google Chrome Framework.framework/
File Help F3Setup F3Search F4Filter F5Tree F6SortByF7Nice F8Nice F9Kill F10Quit

kidevelop@kidevelop-MacBookPro.local ~ 218.148.54.99 12663MB / 32768MB 100% 46.54 ↑ 0kB/s 0kB/s Not running
```

Macbook Pro 16' CTO (i9-9880H, 32GB DDR4)

“아라” 검색 5000회

demo: ara with elasticsearch

legacy ara DB의 title, content_text를
elasticsearch에 올려 검색엔진을
구현하였습니다.

contain, full text index, elastic search
3개의 검색 메소드를 구현하였습니다.

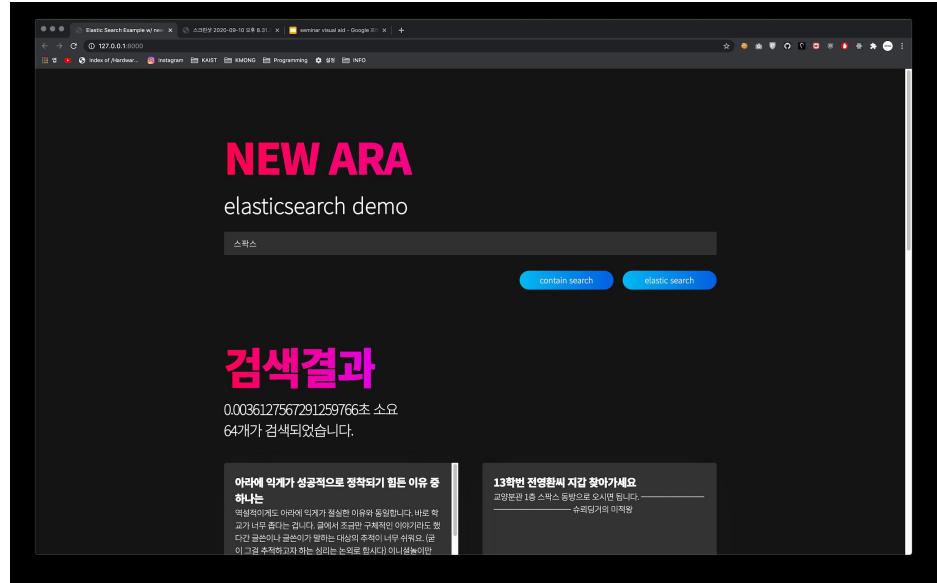
github.com/hjh010501/sparcs-seminar-elasticsearch

```
>>>
>>>
>>> search('나는 야 ')
<Response: [<Hit(newara-index/417926): {'title': '[Secret Bar] 나는 야
>>> search('스팍스 ')
<Response: [<Hit(newara-index/313207): {'title': '스팍스 화이팅 ', 'co
스 배너 ', 'content': "http://ara.kaist.ac.kr/b..."}], <Hit(newara-index/67426): {'title': '교분에서 자주 보이시고 스팍스 의심해 갈던데...
요~', 'content': '\r\n\r\n...'}], <Hit(newara-index/117964): {'title': 'e': '스팍스 자보를 보다가 문득 궁금해서요.', 'content': '거의 모든 자
>>> search('아라 ')
<Response: [<Hit(newara-index/434246): {'title': '[를 ] 아라 분위기가
도 데체 아라 아이디를 어떻게 만드는거죠 ', 'content': '오늘 푸드...'}],
...}, <Hit(newara-index/442694): {'title': '아라 메인 페이지가 바꼈
인증 건의 ', 'content': '안녕하세요 \r\n\r\n...'}, <Hit(newara-index/630): {'title': '아라 홈페이지 건의 여기다 해도 되나요?', 'content':
nt': '○ 모니터링 ○ \r\n...'}], <Hit(newara-index/470044): {'title': '아라 서버 임시 점검 예고 (오늘 오후 2시~4시경)', 'content': '...'}]>
>>> search('함종현 ')
<Response: {}>
```

demo: ara with elasticsearch

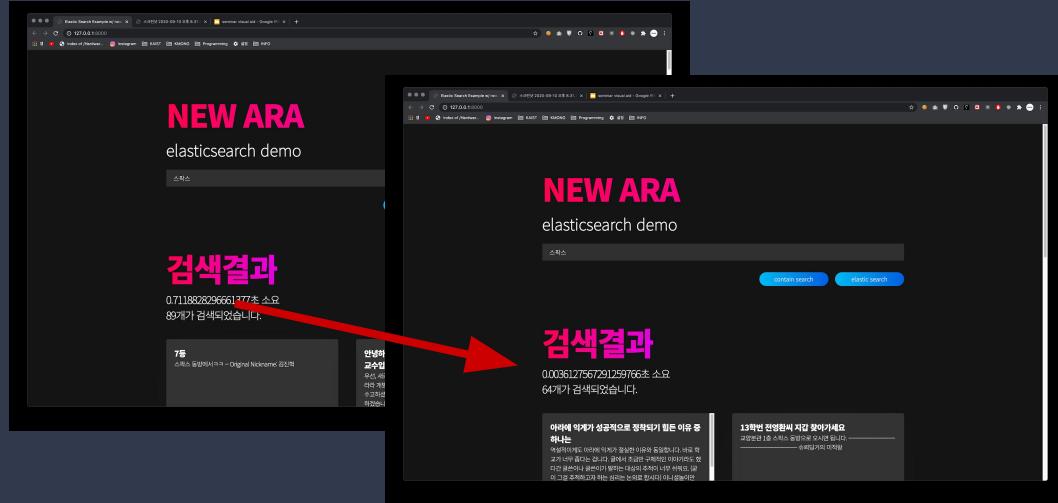
Django template으로 단순한 웹사이트를
만들었고, axios 통신으로 검색 기능을
구현하였습니다.

contain, full-text-index, elastic 각각의
검색을 구현했습니다.



20x speed

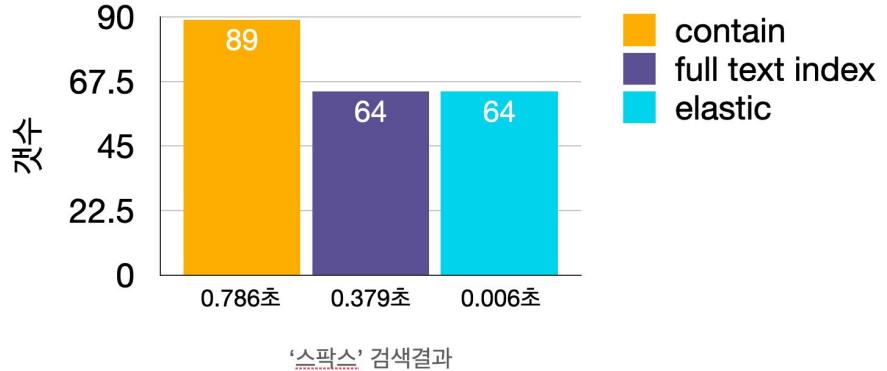
elasticsearch가
django query보다
수십 배 빠른 것을
확인하실 수 있습니다.



demo: benchmark

데모의 자세한 비교 데이터입니다.

elasticsearch 쪽이 검색 결과가 조금 적지만, 훨씬 더 빠르게 응답하고 있습니다.



demo: benchmark

하지만 ‘ara’ 같은 경우, 검색 갯수 차이가
컸습니다. 검색 결과를 비교해본 결과
contain은 a, r, a 있으면 모두 가져오는
것이었습니다.

full-text-index와 elasticsearch의 차이는
웹 url의 ‘ara’를 가져오는지 여부로
생각됩니다.



elasticsearch: precautions

nori 형태소 분석기는 일반적인 단어를 기준으로 검색용 hash를 만들기 때문에,

카이스트 내에서만 쓰이는 단어,
신조어 등은 잘 검색되지 않습니다.

이를 위해 ‘사용자 사전’에 미리 적절한 단어를 입력해 줄 필요성이 있습니다.
(데모에서는 시간상 구현하지 않음)

사용자 사전은 다음 경로에 있습니다.

```
/etc/elasticsearch/userdict_ko.txt
```

사용자 사전의 용도는 인덱싱을 위해 잘라지지 않는 단어들에 대한 분리를 정의 할 수 있습니다. 전체 단어와 그 단어가 잘라진 모양을 기록해두면 됩니다.

고양이톱밥 고양이 톱밥

한글형태소분석기 한글 형태소 분석기

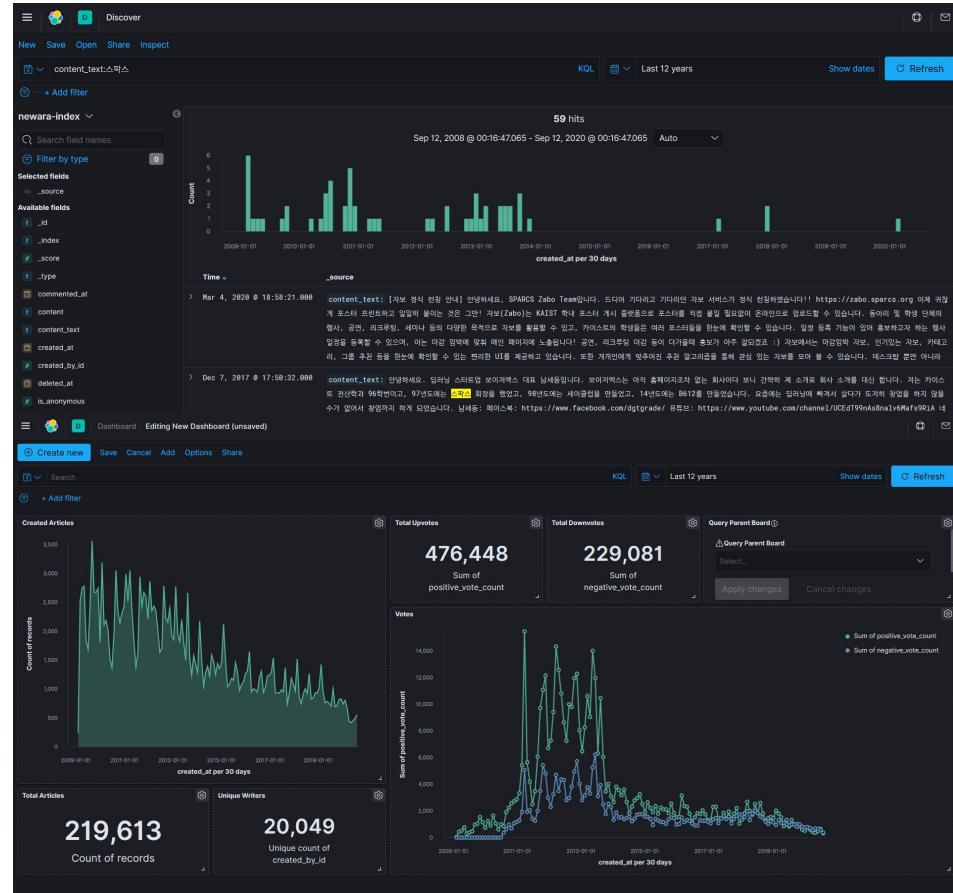
대한민국임시경부수립기념일 대한민국 임시정부 수립 기념일
우리동네 우리 동네

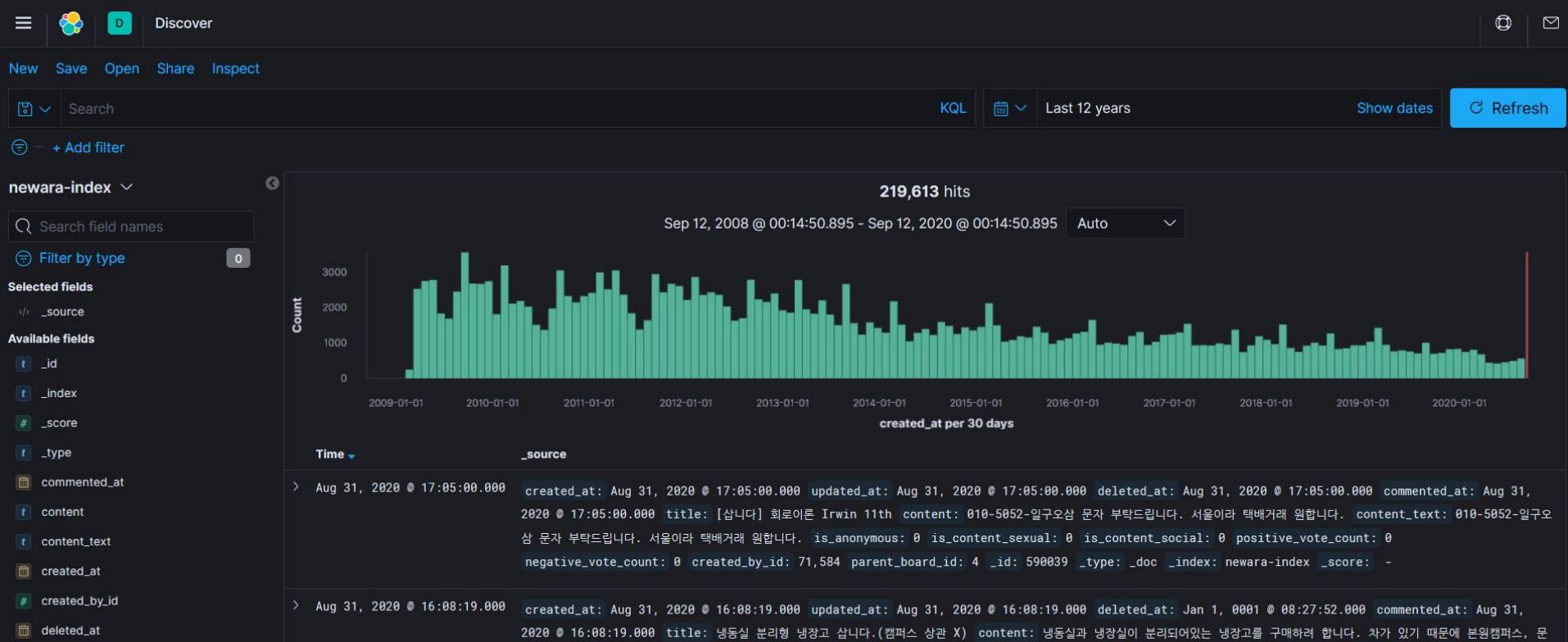
..... 등등 단어의 확장으로 ...

도스르다 무슨 일을 하려고 벌려서 마음을 가다듬다(x 이건안됨)

demo: elastic stack kibana

kibana를 이용해 elasticsearch 서버에 접속한 모습을 보시겠습니다.

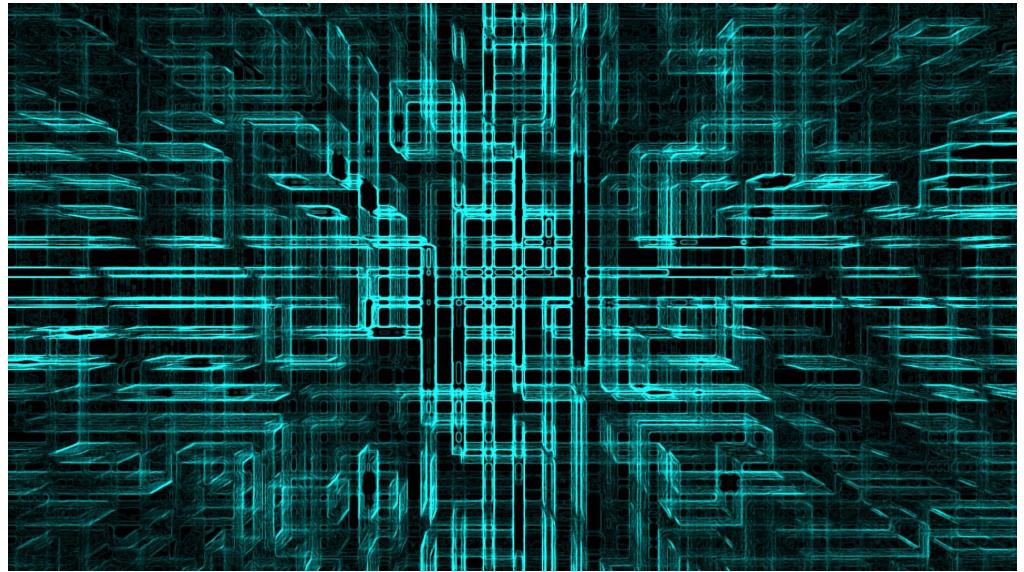




demo: kibana

new-ara and elasticsearch: future plans

elasticsearch를 이용해 new-ara 및 포탈
공지 등의 빠른 검색을 구현하고자 합니다.



Introduction to Elasticsearch

Q&A

SPARCS new-ara team