Warsaw R Users

Elastic with R

elasticsearch. RopenSci

@pcejrowski

"...when you've got more data, than you know what to do with..."

Chris Ambler, Principal Software Development Engineer, GoDaddy

...Elasticsearch to the rescue!

Elasticsearch

database and search engine...

...on steroids

Usage:
text search engine, analytics store, auto completer,
spell checker and alerting engine.


interface
shard
Lucene
Apache
elasticsearch
server
search
RESTful
API
Solr
engine
REST
open
real-time
GET
Elastic
JSON
index
data
Java
support
document
scalable
solution
distributed
source

Advantages

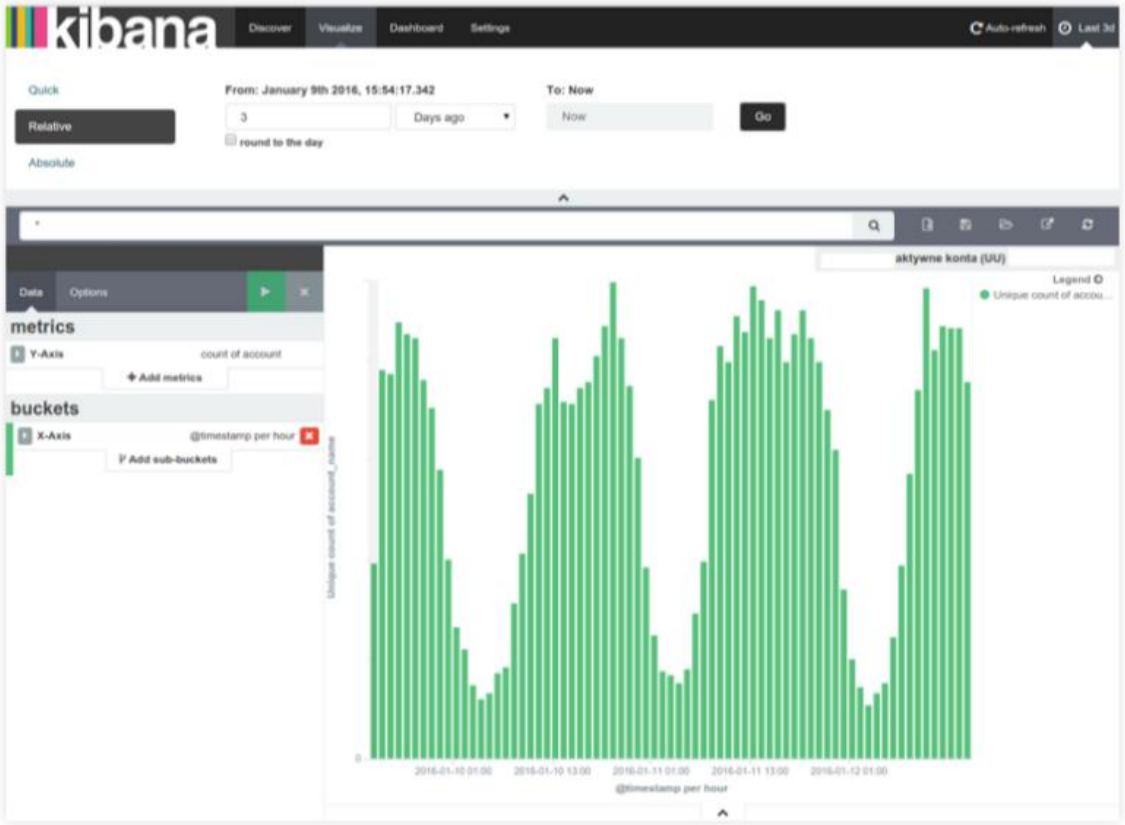
- Scalability
- (Near) real-time searches
- Automatic JSON indexing
- RESTful API

Constraints

- Slow large computations on DB side
- Locking required for transactions
- No native SQL support

ELK


Logstash Search
EVENTS OVER TIME
MAP
REVENUE
22%
VOLUME
9%
TOP DESTINATIONS
TOP SOURCES
GEO PAIRS
ALL EVENTS


kibana
Discover
Visualize
Dashboard
Settings
Auto-refresh
Last 3d
Quick
Relative
Absolute
From: January 9th 2016, 15:54:17.342
3
Days ago
round to the day
To: Now
Now
Go
Data
Options
metrics
Y-Axis
count of account
Add metrics
buckets
X-Axis
@timestamp per hour
Add sub-buckets
aktywne konta (UU)
Legend
Unique count of accou...
Unique count of account_name
0
2016-01-10 01:00
2016-01-10 13:00
2016-01-11 01:00
2016-01-11 13:00
2016-01-12 01:00
@timestamp per hour


Ad Rem

Connection

```
install.packages("elastic")
library('elastic')
connect(url = 'http://127.0.0.1', es_port = 9200)
```

Diagnostics

```
connection()
ping()
cluster_health()
cat_indices()
cat_health()
cat_nodes()
```

Installation

```
#!/bin/sh
set -e

curl -L -O https://download.elasticsearch.org/elasticsearch/release/org/elastic
tar -zxvf elasticsearch-2.1.1.tar.gz

# sudo mv ./elasticsearch-2.1.1 /usr/local
# cd /usr/local
# rm -rf elasticsearch
# sudo ln -s elasticsearch-2.1.1 elasticsearch
```

Data

```
> data(HairEyeColor)
> hec <- as.data.frame(HairEyeColor)
> head(hec)
   Hair  Eye   Sex  Freq
 1 Black Brown Male   32
 2 Brown Brown Male   53
 3   Red Brown Male   10
 4 Blond Brown Male    3
 5 Black  Blue Male   11
 6 Brown  Blue Male   50
> nrow(hec)
 [1] 32
```

Documents

```
docs_bulk(hec, index ="haireyecolor", type = "haireyecolorentity")
alias_create(index = "haireyecolor", "hec")
count(index = "hec")
r0 <- docs_get(index = "hec", type = "haireyecolorentity", id = 1)
r0$`_source`
r1 <- docs_mget(index = "hec", type = "haireyecolorentity",
                        ids = c(1,2,3,4,5,6,7,8,9), raw = TRUE) # JSON
es_parse(r1)
```

Search(bool query)

```
body<-'
  {"query":{
    "bool" : {
      "must" : {
        "term" : { "Sex" : "male" }
      },
      "filter": {
        "term" : { "Eye" : "blue"  }
      },
      "must_not" : {
        "range" : {
          "Freq" : { "from" : 0, "to" : 10 }
        }
      },
      "should" :
      {
        "term" : { "Hair" : "blond" }
      }
```

Google: Apache Lucene Scoring

Aggregations

```
"aggregations" : {
  "_aggregation_name_" : {
    "_aggregation_type_" : {
      _aggregation_body_
    }
    ,"aggregations" : { _sub_aggregation_ }
  }
  "_aggregation_name_2_" : { ... }
}
```

Aggregation types:

- bucket - groups documents based on specified criterion
- metrics - avg, sum, geo bounds

Example

```
agg <- '{
  "aggs": {
    "by_hairs": {
      "terms": {
        "field": "Hair"
      },
      "aggs": {
        "count": {
          "sum": {
            "field": "Freq"
          }
        }
      }
    }
  }
}'
Search(index = "hec", body = agg, asdf = TRUE)
```


'{
"query" : {
"filtered" : {
"filter" : {
"and" : {
"filters" : [{
"terms" : {
"test.raw" : ["test:a"]
}
}, {
"terms" : {
"section" : ["news"]
}
}, {
"range" : {
"@timestamp" : {
"from" : "now-2h",

Fuzzy queries

```
body <- '{"query": {"match" : {"Hair" : "Blond"}}}'
Search(index = "hec", body = body)
```

```
fuzzy_body <- '{
  "query": {
    "fuzzy": {
      "Hair": {
        "value": "Blend",
        "fuzziness": 2
      }
    }
  }
}'
Search(index = "hec", type = "haireyecolorentity", body = fuzzy_body, asdf = TR
```

The fuzziness

0, 1, 2 the maximum allowed Levenshtein Edit Distance (or number of edits)
AUTO distance based on the length of the term.
For lengths:
0..2 must match exactly
3..5 one edit allowed
5+ two edits allowed

Scrolls

```
res <- Search(index = "hec" , q="*", scroll="1m", search_type = "scan")
out <- list()
hits <- 1
while(hits != 0){
  res <- scroll(scroll_id = res$`_scroll_id`)
  hits <- length(res$hits$total)
  if(hits > 0)
    out <- c(out, res$hits$hits)
}
length(out)
```

Percolators

```
percolator_body <- '{
  "query" : {
    "match" : {
      "Sex": "Female"
    }
  }
}'
percolate_register(index = "hec", id = 1, body = percolator_body)
doc <- '{
  "doc": {
    "Hair": "Dark",
    "Sex": "Female"
  }
}'
percolate_match(index = "hec", type = "my", body = doc)
```

Fields referred to in a percolator query must already exist.