elasticsearch

Shifa Khan
github/shifakhan
@findshifa

Why do we need search options?

We need it to make our lives easier

Find stuff that is relevant to us

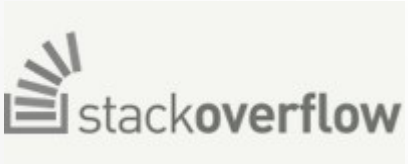
Find it faster

How do we add seach features to our projects?

Options available:

- Standard Database Query
- Lucene libraries
- Solr
- Sphinx
- Elasticsearch

elasticsearch



. . .

Topics for today!

Full Text Searching

RESTful ES

Elasticsearch in Rails

Advanced features of ES

Note: ES = ElasticSearch

What is elasticsearch?

- Database server
- Implemented with RESTful HTTP/JSON
- Easily scalable (hence the name *elastic*search)
- Based on Lucene

Features of elasticsearch

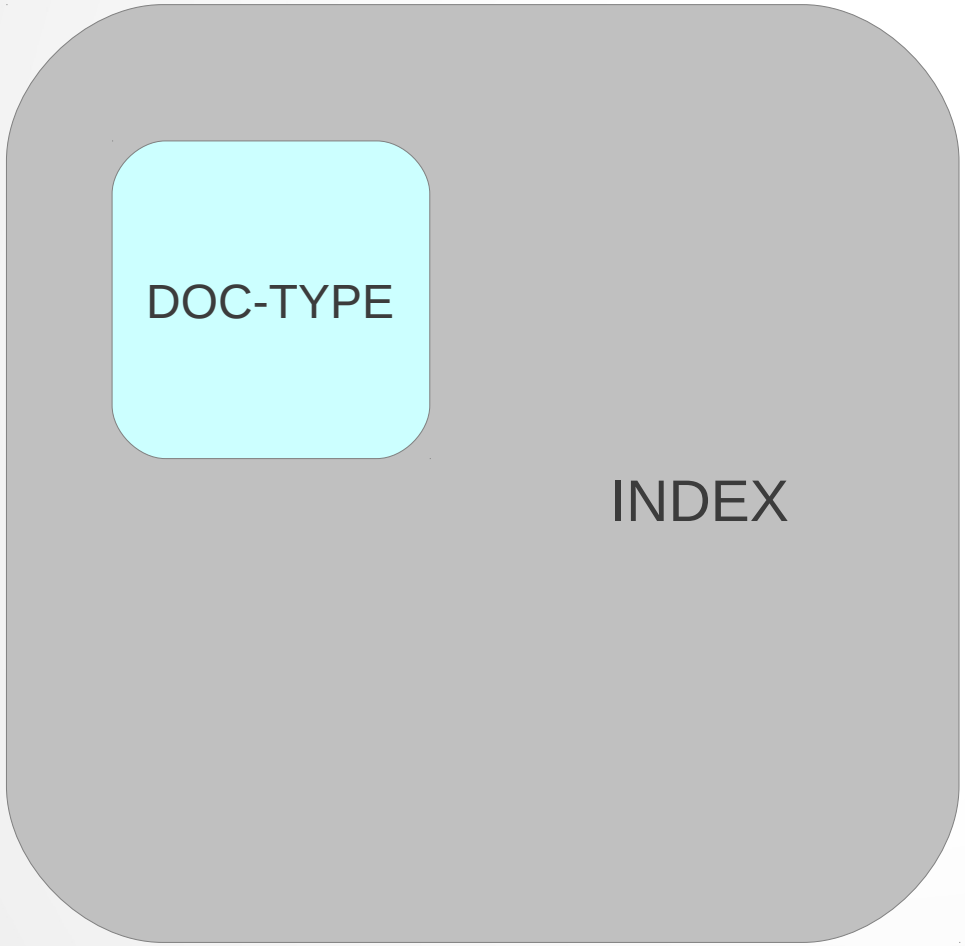
- Schema-free
- Real-time
- Easy to extend with a plugin system for new functionality
- Automatic discovery of peers in a cluster
- Failover and replication
- Community support: Multiple clients available in various languages for easy integration

  Tire gem- ruby client available for ActiveModel integration

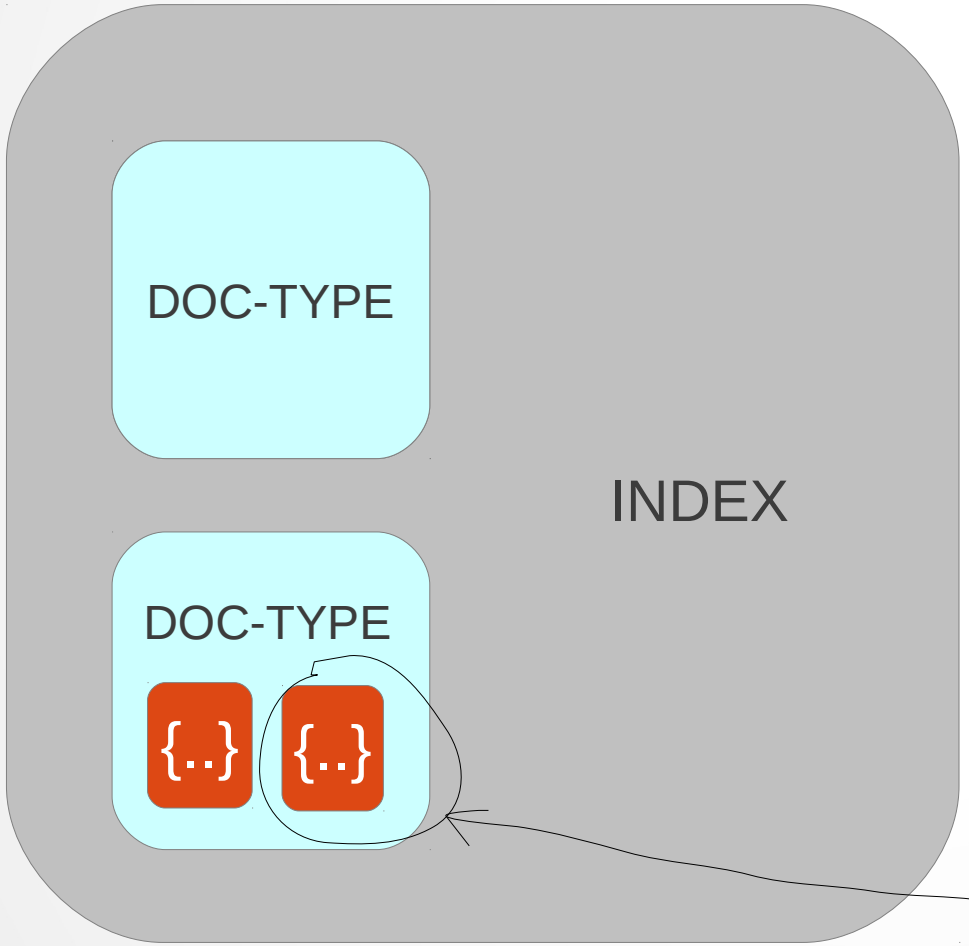
Terminology

Relational database		Elasticsearch
Database	⟷	Index
Table	⟷	Type
Row	⟷	Document
Column	⟷	Field
Schema	⟷	Mapping

Inside ES



Data Structure
of
Elasticsearch

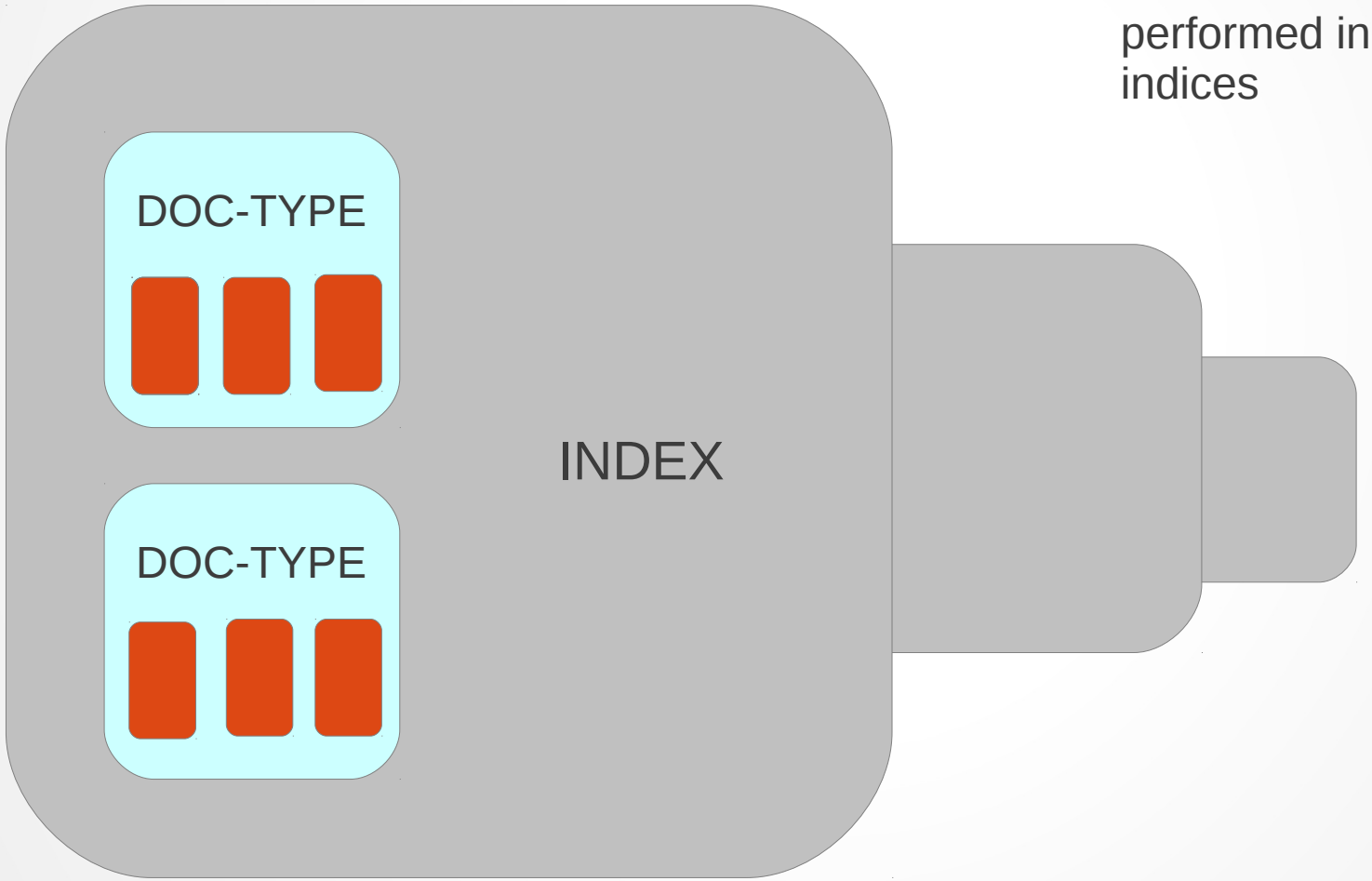


An Index can have multiple types.

Each type can have multiple documents.

Each document contains data in **JSON** format

Searches can also be performed in multiple indices



How does elasticsearch work?

Full text searching

- Inverted indexing
- Analysis

FULL TEXT SEARCH

FULL TEXT SEARCH

stars shine sky

stars stars sky

cloudy sky

WORD	LOCATION
stars	0,1
shine	0
sky	0,1,2
cloudy	2

FULL TEXT SEARCH

stars shine sky

stars stars sky

cloudy sky

WORD	LOCATION
stars	0,1
shine	0
sky	0,1,2
cloudy	2

FULL TEXT SEARCH

stars shine sky

stars stars sky

cloudy sky

WORD	LOCATION	POSITION
stars	0	0
	1	0,1
shine	0	1
sky	0	2
	1	2
	2	1
cloudy	2	0

FULL TEXT SEARCH

stars shine sky

stars stars sky

cloudy sky

WORD	LOCATION	POSITION
stars	0	0
	1	0,1
shine	0	1
sky	0	2
	1	2
	2	1
cloudy	2	0

Are these positions in the document are consecutive ?

FULL TEXT SEARCH

stars shine sky

stars stars sky

cloudy sky

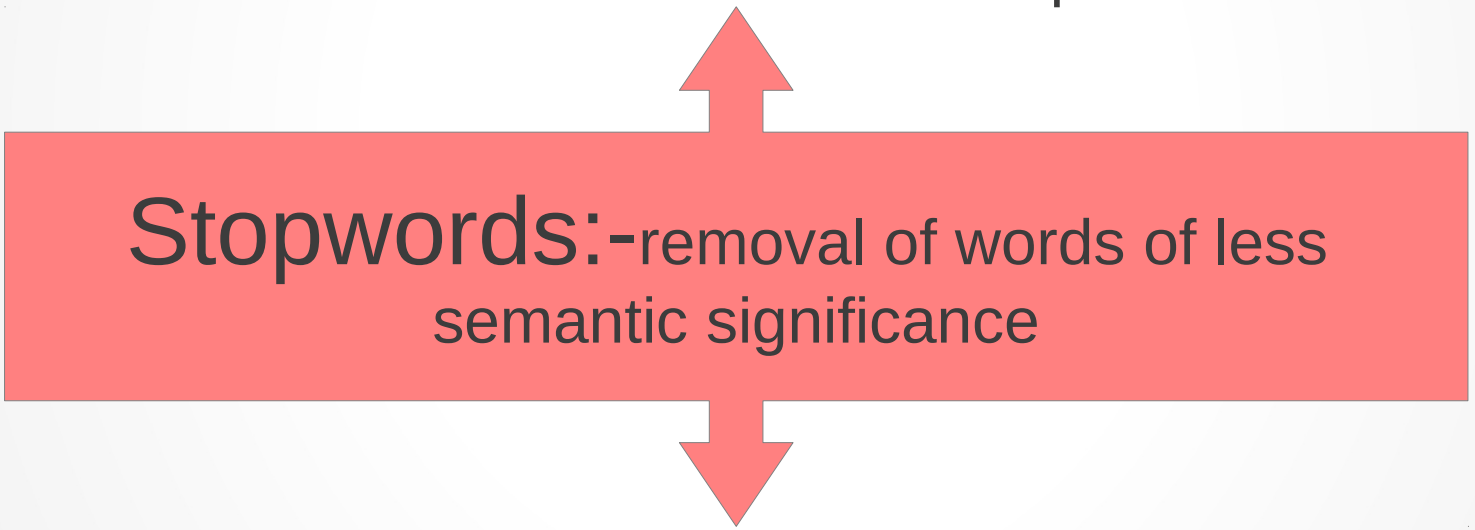
WORD	LOCATION	POSITION
stars	0	0
	1	0,1
shine	0	1
sky	0	2
	1	2
	2	1
cloudy	2	0

We find the words in consecutive positions in Document 1

ANALYSIS

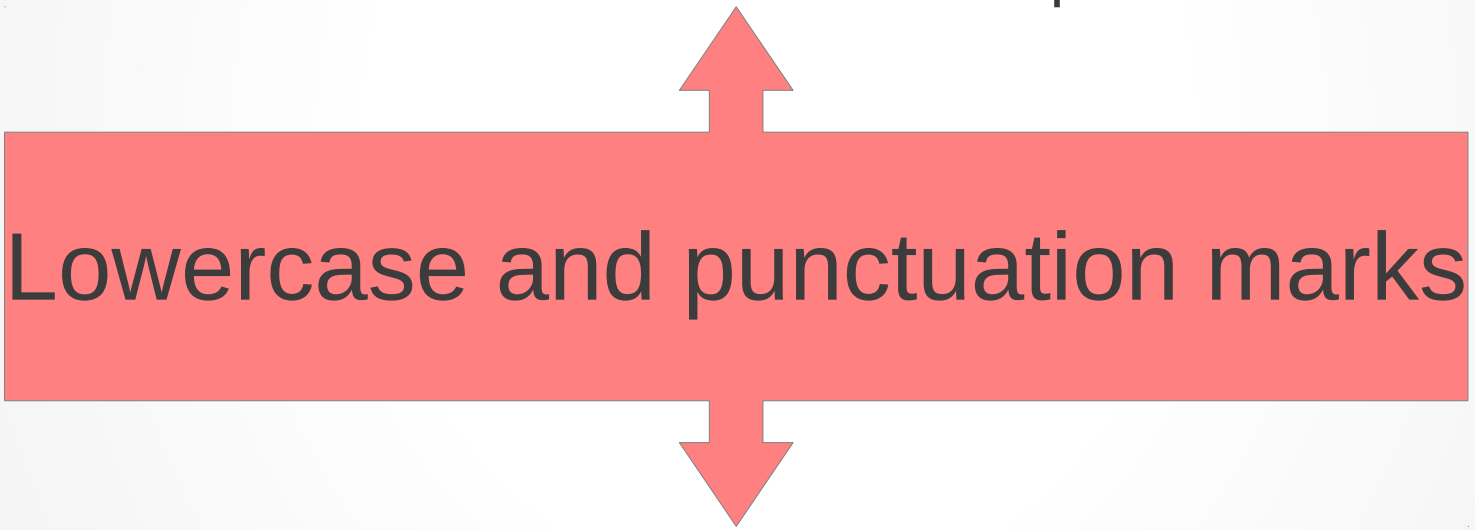
Analyzing is extracting “terms” from given text. Processing natural language to make it computer searchable.

Ruby is a dynamic, reflective, general-purpose OOP language that combines syntax inspired by Perl with Smalltalk-like features. Ruby was first designed and developed in the mid-1990s by Yukihiro "Matz" Matsumoto in Japan.



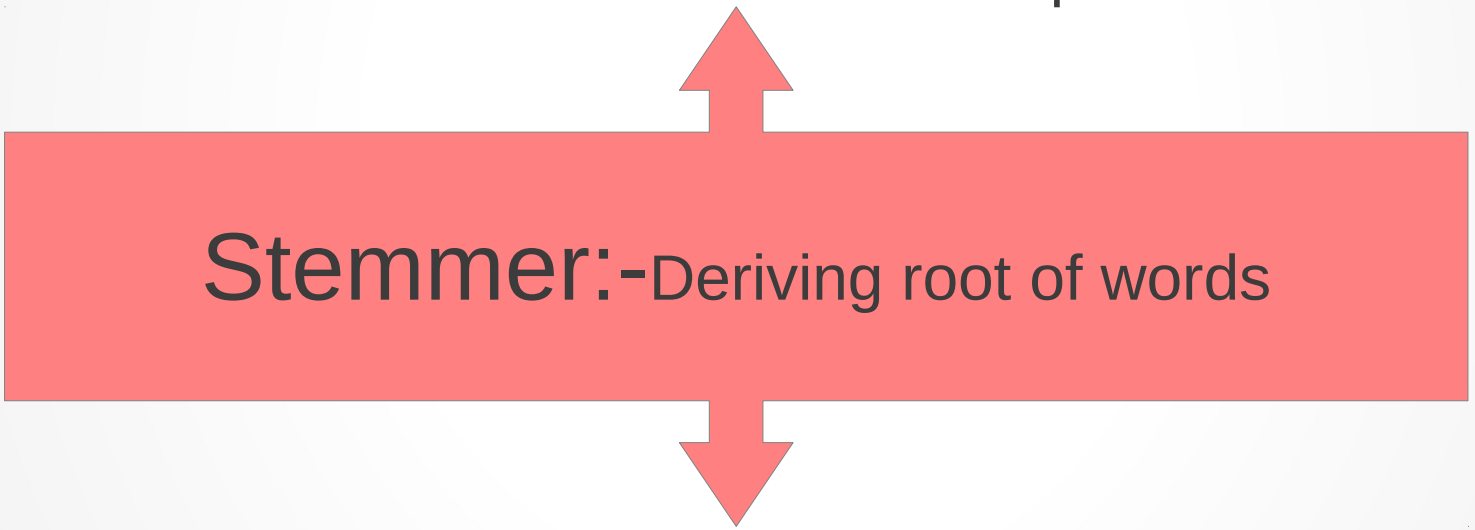
ruby dynamic reflective general purpose oop language combine syntax inspire perl smalltalk feature first design develop mid 1990 yukihiro matz matsumoto japan

Ruby is a dynamic, reflective, general-purpose OOP language that combines syntax inspired by Perl with Smalltalk-like features. Ruby was first designed and developed in the mid-1990s by Yukihiro "Matz" Matsumoto in Japan.



ruby dynamic reflective general purpose oop language combine syntax inspire perl smalltalk feature first design develop mid 1990 yukihiro matz matsumoto japan

Ruby is a dynamic, reflective, general-purpose OOP language that combines syntax inspired by Perl with Smalltalk-like features. Ruby was first designed and developed in the mid-1990s by Yukihiro "Matz" Matsumoto in Japan.



ruby dynamic reflective general purpose oop language combine syntax inspire perl smalltalk feature first design develop mid 1990 yukihiro matz matsumoto japan

ES Analyzer

- Analyzer:
  Consists of one tokenizer and multiple token filters
  eg: Whitespace, Snowball,etc

- Tokenizer:
  It tokenizes all words. Splits sentences into individual 'terms' . Ngram and EdgeNgram highly useful for autocomplete feature. Path hierarchy tokenizers.

- Token filter:
  Actions on tokenized words, basic lowercase to phonetic filters and stemmers (available in many languages)

Popular Analyzers

ES is easy to use. Readymade analyzers for general usage.

- Snowball – excellent for natural language

  Standard tokenizer, with standard filter, lowercase filter, stop filter, and snowball filter

- Some fancy analyzers: Pattern analyzers

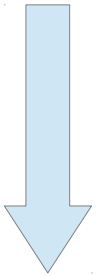
  For all the Regular expressions guys out there!

  "type": "pattern",

  "pattern":"\\s+"

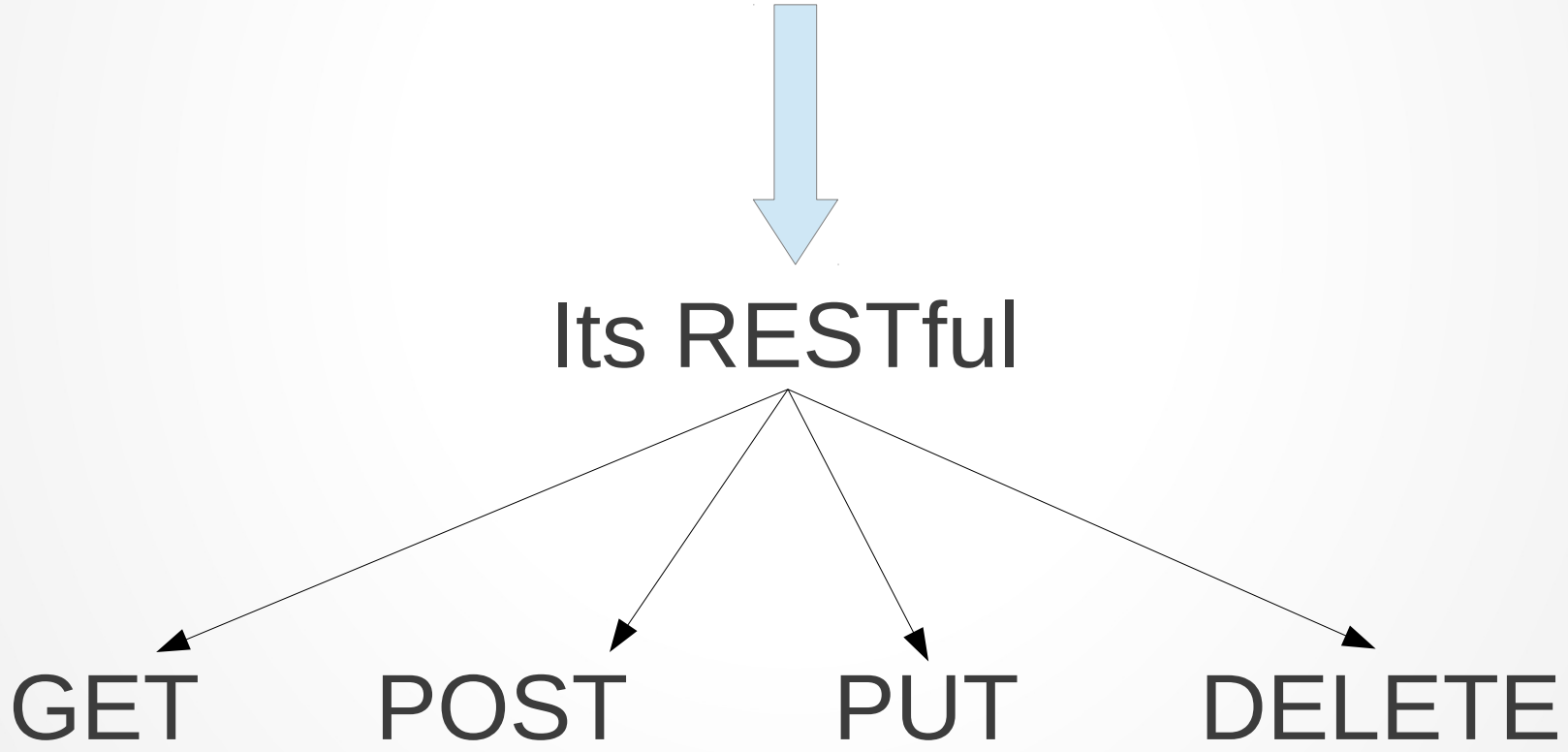
How do we access Elasticsearch?

How do we access Elasticsearch?

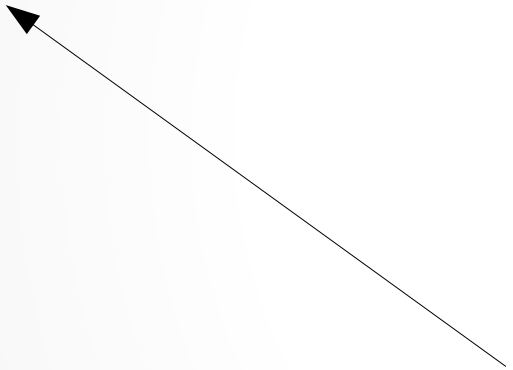


Its RESTful

How do we access Elasticsearch?

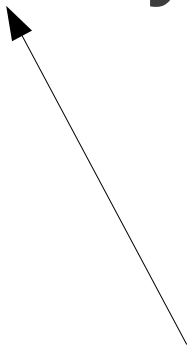


PUT /index/type/id



action?

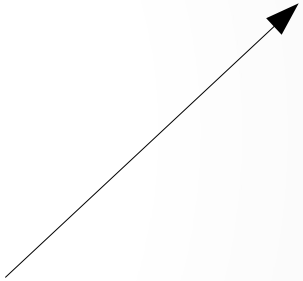
PUT /index/type/id



where?

PUT /twitter_development/type/id

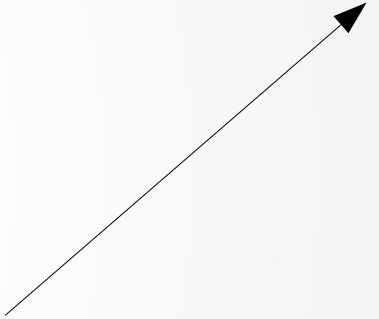
PUT /twitter_development/type/id



what?

PUT /twitter_development/tweet/id

PUT /twitter_development/tweet/id



which?

PUT /twitter_development/tweet/1

```
curl -XPUT 'localhost:9200/twitter_development/tweet/1' -d
    '{
        "tweet" : " Elasticsearch is cool! ",
        "name" : "Mr Developer"
    }'
```

```
curl -XPUT 'localhost:9200/twitter_development/tweet/1' -d
    '{
        "tweet" : " Elasticsearch is cool! ",
        "name" : "Mr Developer"
    }'
```

```
{
    "_index":"twitter_development",
    "_type":"tweet",
    "_id":"1",
    "_version": 1,
    "ok":true
}
```

Similarly,

```
curl -XGET 'http://localhost:9200/twitter/tweet/1'
```

Similarly,

```
curl -XGET 'http://localhost:9200/twitter/tweet/1'
```

```
{
    "_index" : "twitter",
    "_type" : "tweet",
    "_id" : "1",
     "_source" : {
         "tweet" : " Elasticsearch is cool! ",
         "name" : "Mr Developer"
    }
}
```

Also

GET /index/_search

GET /index1,index2/_search

GET /ind*/_search

GET /index/type/_search

GET /index/type1,type2/_search

GET /index/type*/_search

GET /_all/type*/_search

Search all types (columns)

Search multiple or all indices (databases)

Similarly,

DELETE

Delete document

```
curl -XDELETE 'http://localhost:9200/twitter/tweet/1'
```

```
curl -XDELETE 'http://localhost:9200/twitter/'
```

Delete index

```
curl -XDELETE 'http://localhost:9200/twitter/tweet/_query'-d
'{
    "term" : { "name" : "developer" }
}'
```

Elasticsearch in Rails

ES in your RAILS app

1. Install ES from website. Start service.
2. In your browser check 'http://localhost:9200' for confirmation.
3. Add 'tire' to Gemfile.
4. In your model file (say 'chwink.rb') add-

   include Tire::Model::Search

   include Tire::Model::Callbacks
5. Run rake environment tire:import CLASS=Chwink
6. In rails console type

   Chwink.search(“world”) or Chwink.tire.search(“world”)
7. Results!

Request for Model.search

```
curl -XGET
'http://localhost:9200/chwink_development/chwink/_search?
-d
'{
  "query": {
    "query_string": {
      "query": "world"
    }
  }
}'
```

Autocomplete feature implementation
and
Some examples of readymade and custom analyzers

Querying

More types of queries and use cases:

- Faceting : Allowing multiple filters
- Pagination : Limiting per page results
- Sort : Sorting results by relevance and scoring (using Elasticsearch's scoring algorithm)

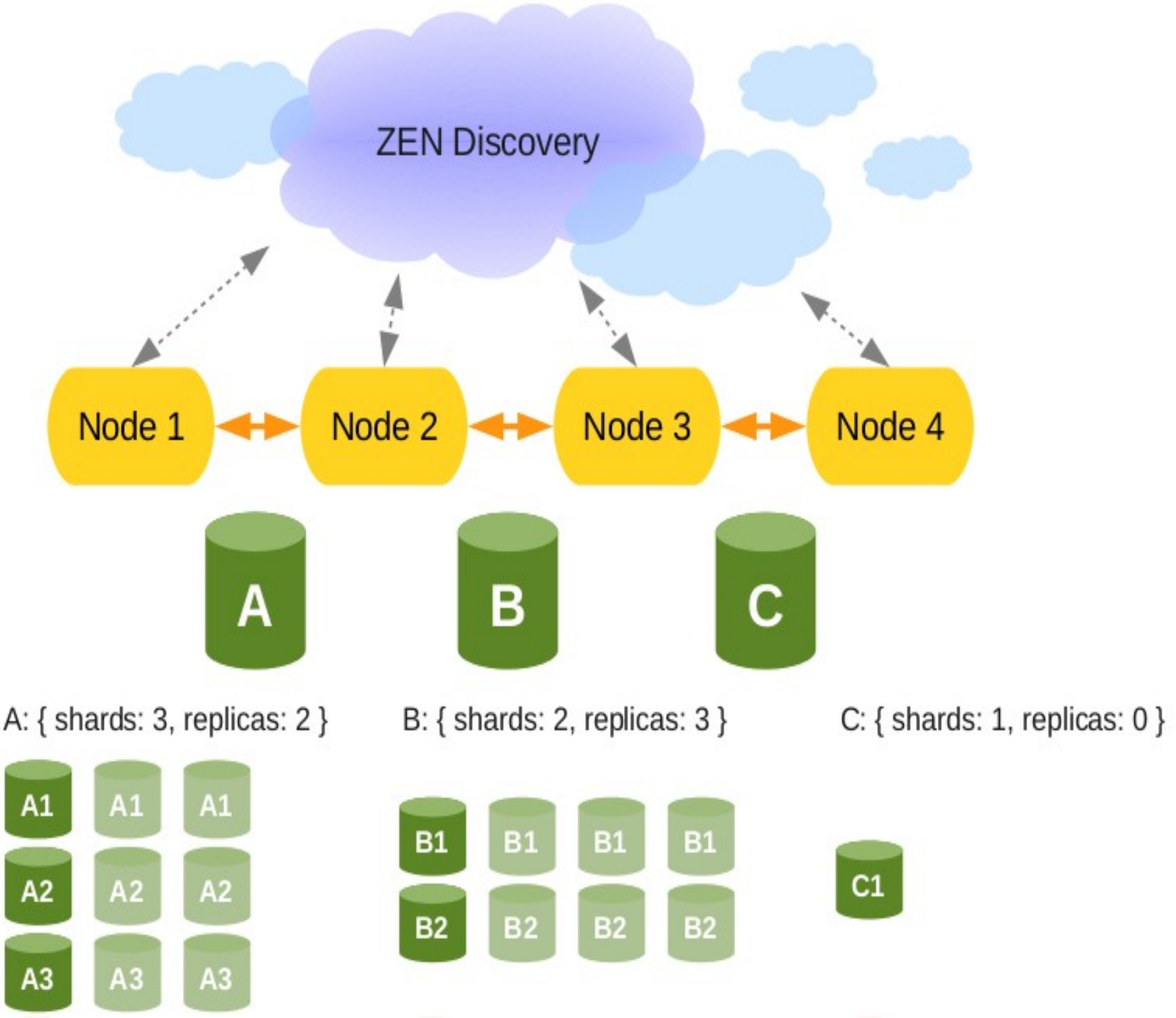
Advanced features of Elasticsearch

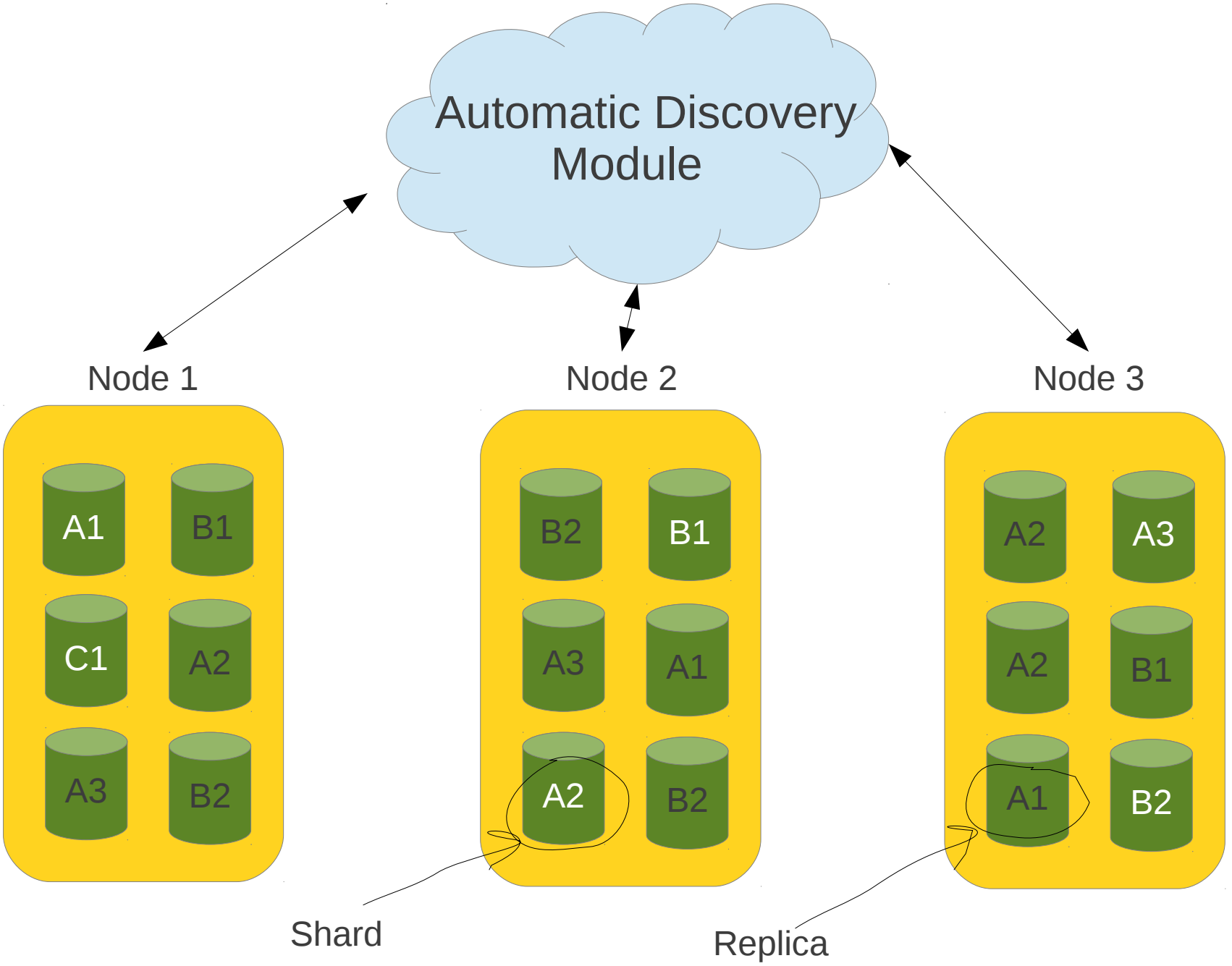
USP of Elasticsearch

BIG! Data Analysis in Real Time

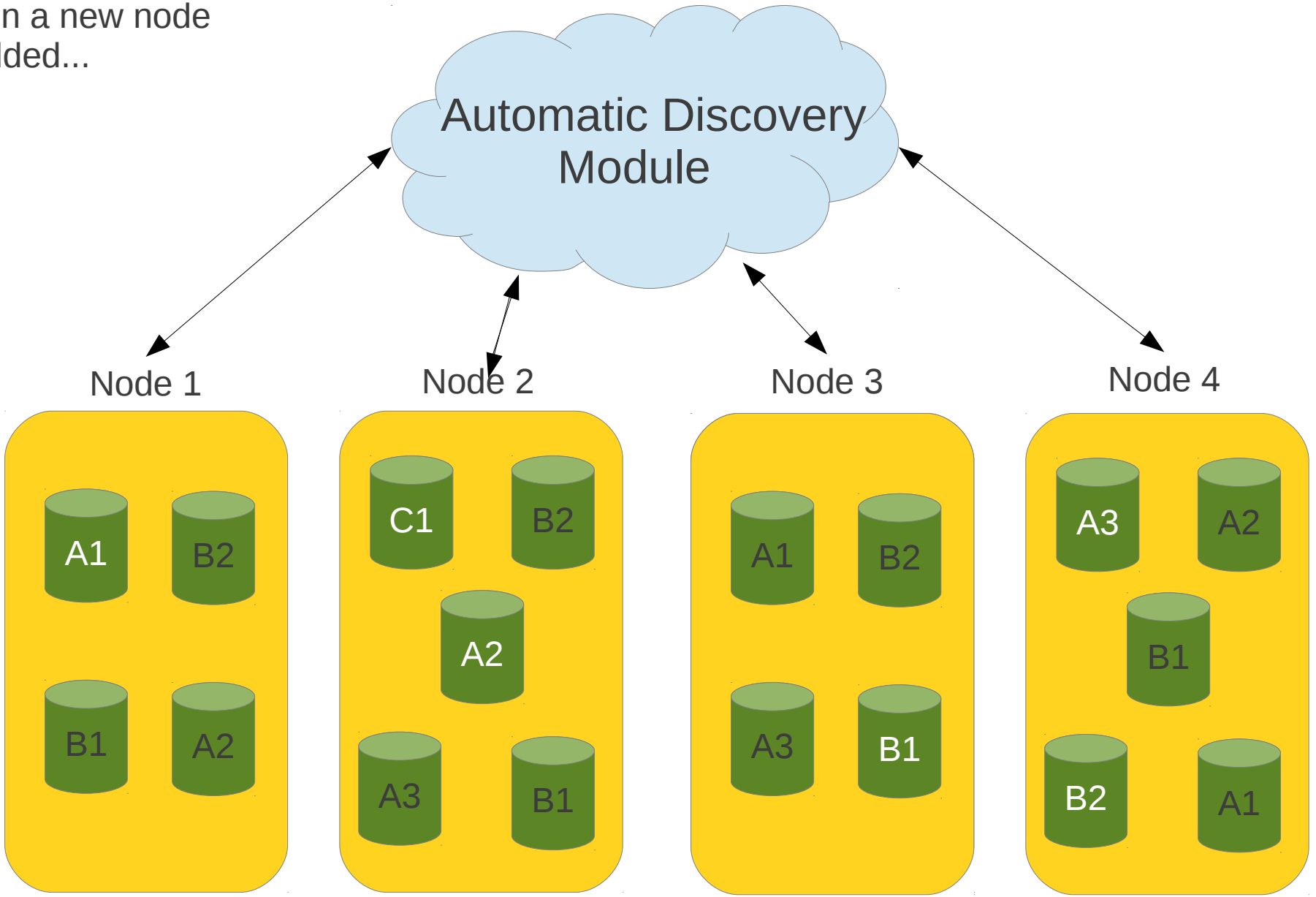
Automatic discovery of peers in a cluster

Failover and replication


ZEN Discovery
Node 1
Node 2
Node 3
Node 4
A
B
C
A: { shards: 3, replicas: 2 }
B: { shards: 2, replicas: 3 }
C: { shards: 1, replicas: 0 }
A1
A1
A1
A2
A2
A2
A3
A3
A3
B1
B1
B1
B1
B2
B2
B2
B2
C1


Automatic Discovery Module
Node 1
Node 2
Node 3
A1
B1
C1
A2
A3
B2
B2
B1
A3
A1
A2
B2
A2
A3
A2
B1
A1
B2
Shard
Replica

When a new node is added...



High Availability

For each index you can specify:

- Number of shards
  - Each index has fixed number of shards
  - Shards improve indexing performance
- Number of replicas
  - Each shard can have 0-many replicas, can be changed dynamically
  - Replicas improve search performance

Tips

- Make sure you make separate indices for test and development environment
  Eg: Add index_name "Chwink_#{Rails.env}" to your model file

- During tests whenever you save an object make sure you add : Chwink.tire.index.refresh after each test case.

- Make sure you delete and recreate the index after tests.
  Eg: You can add this to your Rspec configuration file

```
config.after(:each) do
    Chwink.tire.Chwink_test.delete
    Chwink.tire.create_elasticsearch_index
end
```

- Debug using log files :
  Add to Tire config file: logger "tire_#{Rails.env}.log"

Room for exploration...

Plugins!

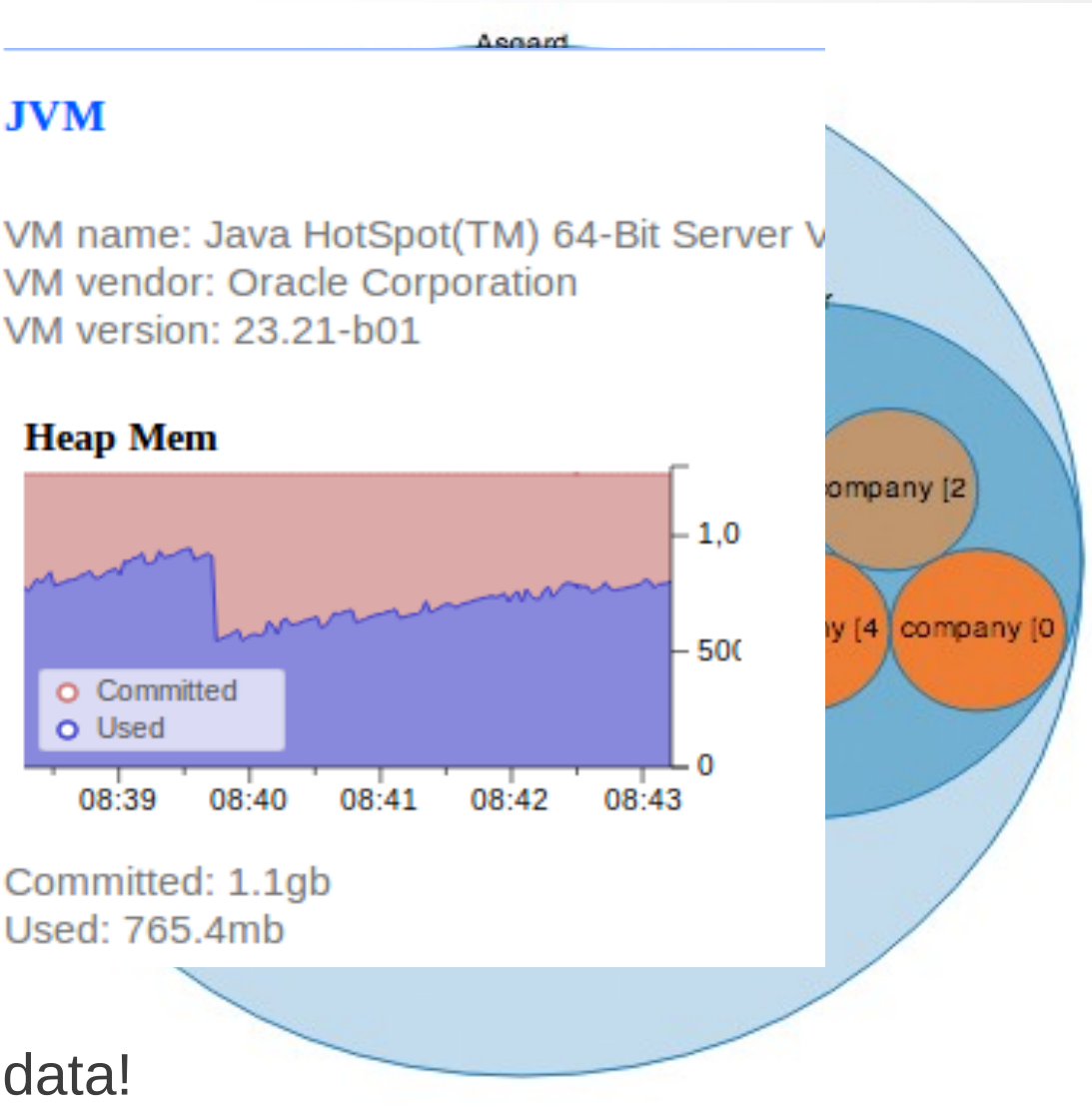
Big Desk plugin!

Plug-in for analysis of your search data!

Room for exploration...

Plugins!

Big Desk plugin!



Plug-in for analysis of your search data!

Plugin

Big Desk

Plug-in for analys

bigdesk

bigdesk.org/v/master/#nodes/1P8QmepBTlqcQWcSZQRinQ

Reader

Transport: 72
HTTP: 0
HTTP total opened: na

Series: weighted avg
Rx: 1.6gb, #2508691
Tx: 690.2mb, #2497097

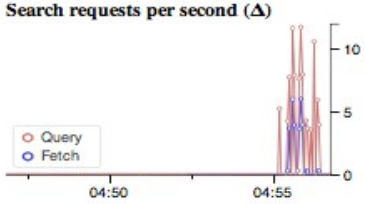
Indices

Docs count: 106820
Docs deleted: 20149

Flush: 5124, 13s
Refresh: 861, 1m

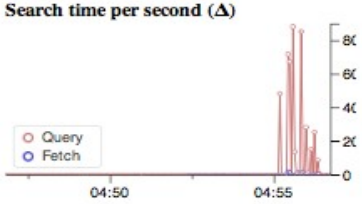
Size: 2.9gb
Filters cached: 439

Search requests per second (Δ)



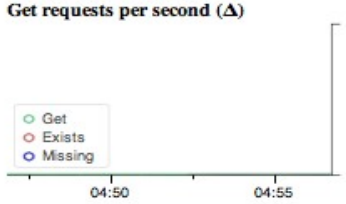
Query: 1842
Fetch: 113

Search time per second (Δ)



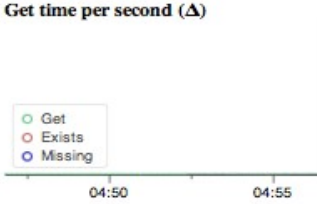
Query: 24.1s
Fetch: 792ms

Get requests per second (Δ)



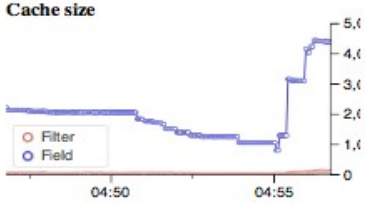
Get: 0
Exists: 0
Missing: 0

Get time per second (Δ)



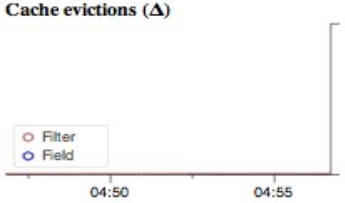
Get: 0s
Exists: 0s
Missing: 0s

Cache size



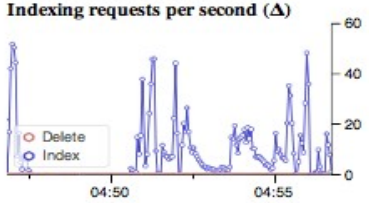
Filter: 102.6kb
Field: 4.1mb

Cache evictions (Δ)



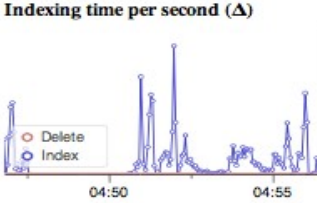
Filter: 0
Field: 0

Indexing requests per second (Δ)



Delete: 0
Index: 197605

Indexing time per second (Δ)

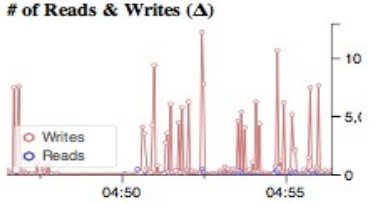


Delete: 0s
Index: 21.2m

File system

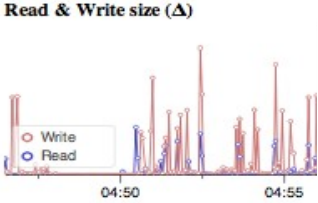
Device: /dev/mapper/system-root
Mount: /
Path: /var/lib/elasticsearch/search-stg.jboss.org/nodes/0
Free: 11.7gb
Available: 10.9gb
Total: 16.4gb

# of Reads & Writes (Δ)



Writes: 3749613
Reads: 64661

Read & Write size (Δ)



Write: 14.3gb
Read: 1.3gb

Now you can go ahead and start exploring Elasticsearch for yourself!!

Thank you!

Questions are welcome!