

Elasticsearch

Jakub Podeszwik

Yameo

28.03.2018

- ① Lucene
- ② Elasticsearch
- ③ Basic data structures
- ④ Searching, Indexing
- ⑤ Full text search
- ⑥ Analyzers
- ⑦ Cluster

- ① Full text search library written in Java
- ② initially released in 1999
- ③ in 2010 Apache Solr joined Lucene as subproject

- ① Open source Full text search Engine written on top of lucene
- ② 02.2010 - version 0.4 released
- ③ 02.2014 - version 1.0 released
- ④ scalable, near realtime, highly available, restful

Inverted index

1. Tom has a cat

2. Kate has a dog

3. Mike has an owl

Term	Documents
------	-----------

a	1, 2
---	------

an	3
----	---

cat	1
-----	---

dog	2
-----	---

has	1, 2, 3
-----	---------

kate	2
------	---

mike	3
------	---

owl	3
-----	---

tom	1
-----	---

Immutable, stored on disk data structure consisting of:

- 1 inverted index
- 2 fielddata cache / doc_values
- 3 __source
- 4 live documents bitset (this one is not immutable)
- 5 ...

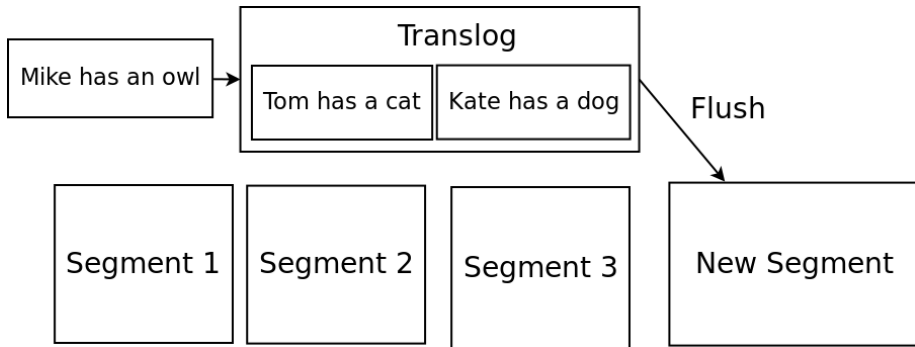
Segment 1

a	1, 2
cat	1
dog	2
has	1, 2
kate	2
tom	1

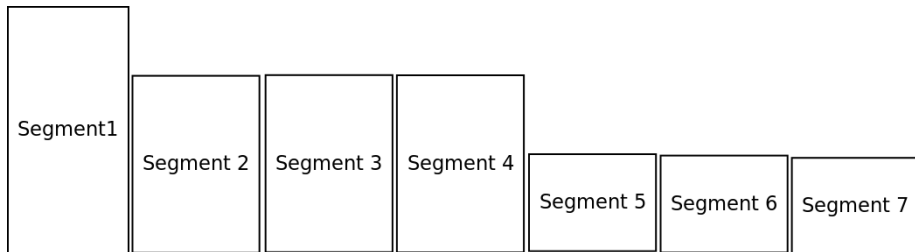
Segment 2

an	3
has	3
mike	3
owl	3

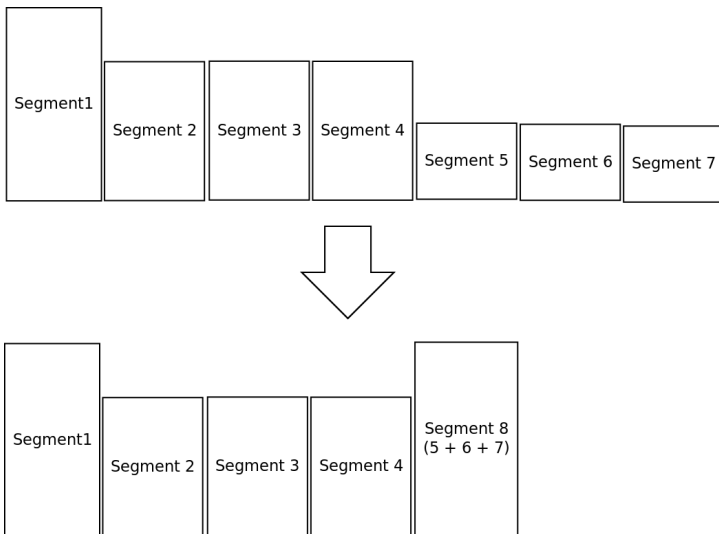
Translog



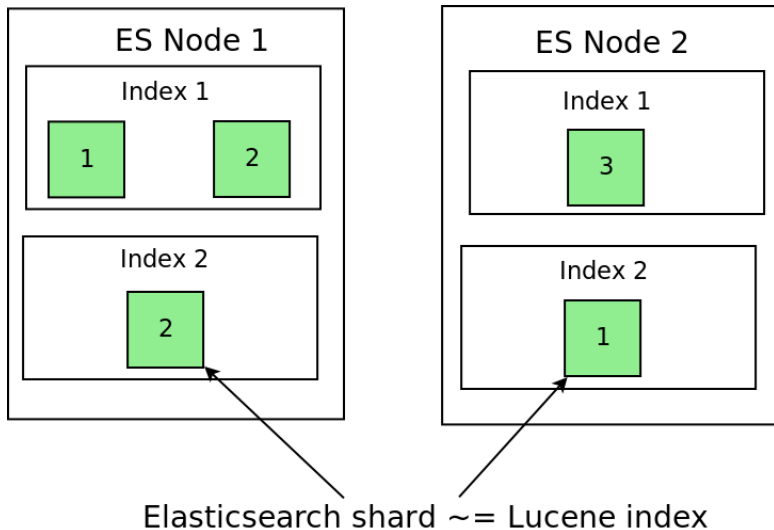
Segments merging



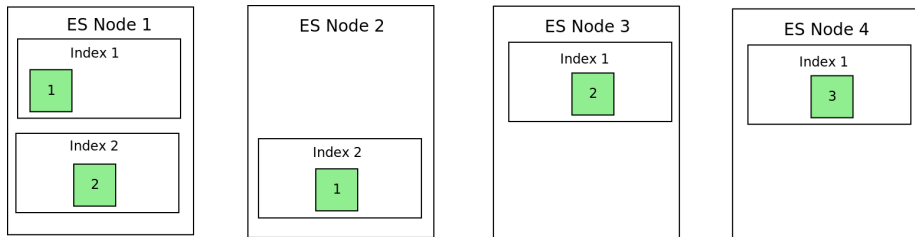
Segments merging



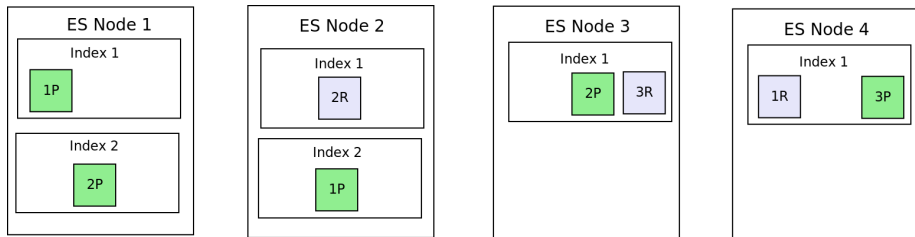
Elasticsearch index



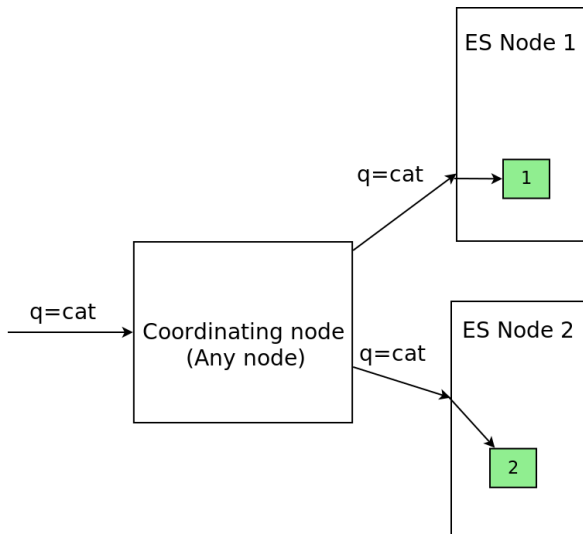
Elasticsearch index

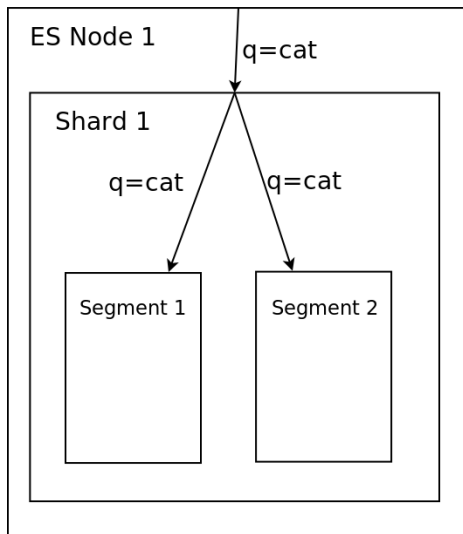


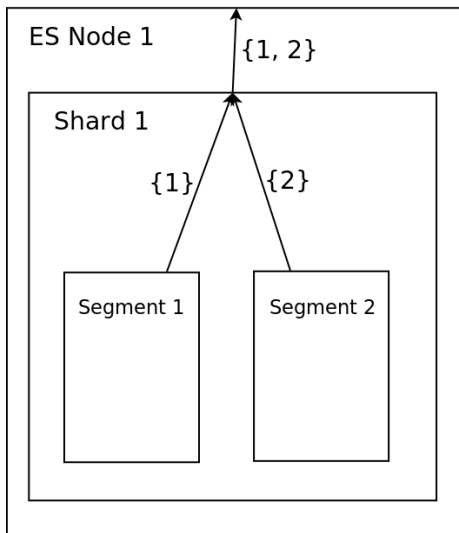
Elasticsearch shard replicas

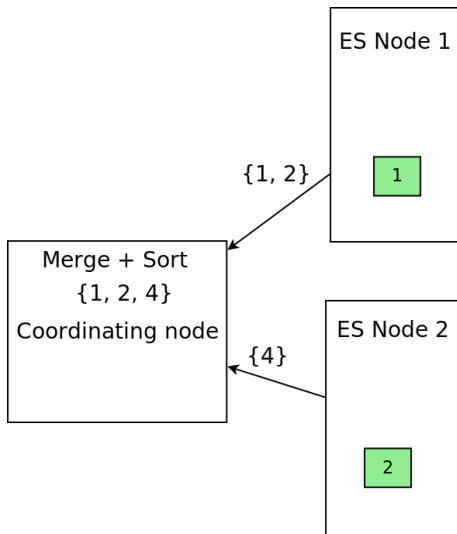


Search

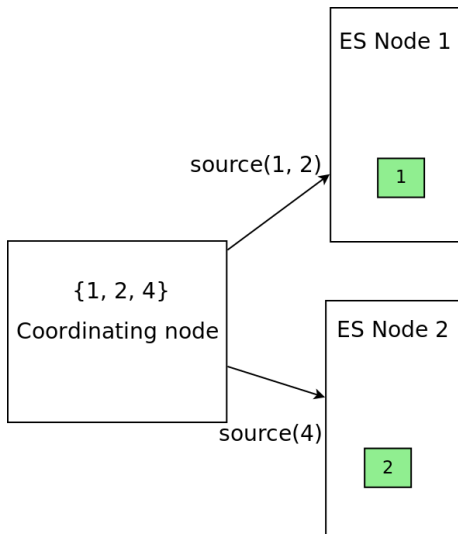




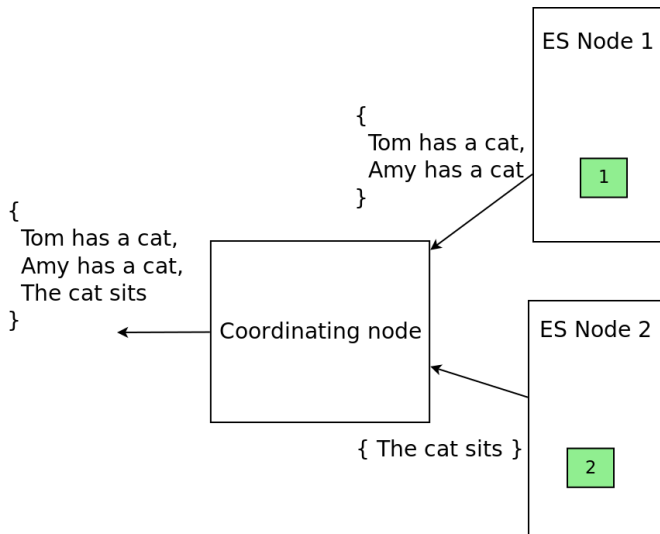




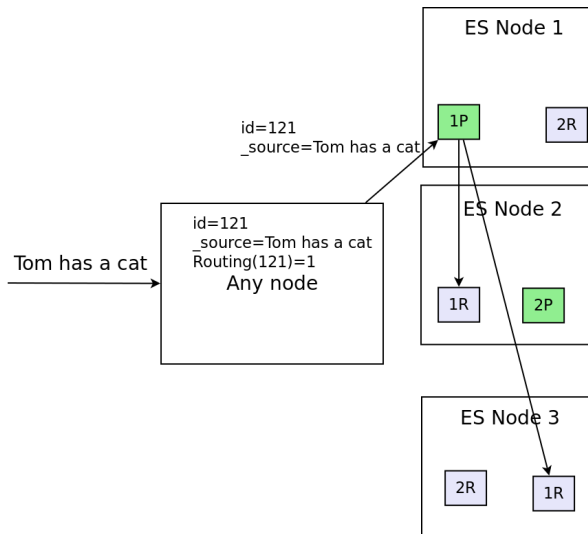
Search



Search



Indexing



- ① indices
- ② shards
- ③ nodes
- ④ master
- ⑤ ?v - legend

Definition of how documents in elasticsearch should be stored / indexed / searched.

Elasticsearch will try to guess mapping from incoming documents if some of the fields are not specified.

- 1 term
- 2 bool
- 3 prefix
- 4 wildcard
- 5 fuzzy
- 6 match, match_all
- 7 query_string
- 8 filter vs query

Analazers split input string into stream of terms. Examples:

- ① whitespace
- ② standard - default analyzer for text fields.
- ③ keyword - noop

Custom analyzer

- 1 char_filters - removes / changes / adds characters before text goes to tokenizer
- 2 tokenizer - splits text into tokens
- 3 filters - removes / changes / adds tokens

Custom analyzer

Tom has a cat, and Kate has two dogs.

Strip punctuation char filter

Tom has a cat and Kate has two dogs

Whitespace tokenizer

[Tom, has, a, cat, and, Kate, has, two, dogs]

Lowercase filter

[tom, has, a, cat, and, kate, has, two, dogs]

English stopwords filter

[tom, cat, kate, two, dogs]

English stem filter

[tom, cat, kate, two, dog]

$$\text{score}(q, d) = \text{queryNorm}(q) * \text{coord}(q, d) * \sum (tf(t \text{ in } d) * idf(t)^2 * t.\text{getBoost()} * \text{norm}(t, d))(t \text{ in } q)$$

Scoring

- 1 $\text{score}(q,d)$ - the relevance score of document d for query q
- 2 $\text{queryNorm}(q)$ - query normalization factor
- 3 $\text{coord}(q,d)$ is the coordination factor (rewards for higher percentage of query terms contained in document)
- 4 $\text{tf}(t \text{ in } d)$ - term frequency for term t in document d ,
- 5 $\text{idf}(t)$ - inverse document frequency for term t ,
- 6 $t.\text{getBoost}()$ - boost that has been applied to the query,
- 7 $\text{norm}(t,d)$ - field-length norm, combined with the index-time field-level boost

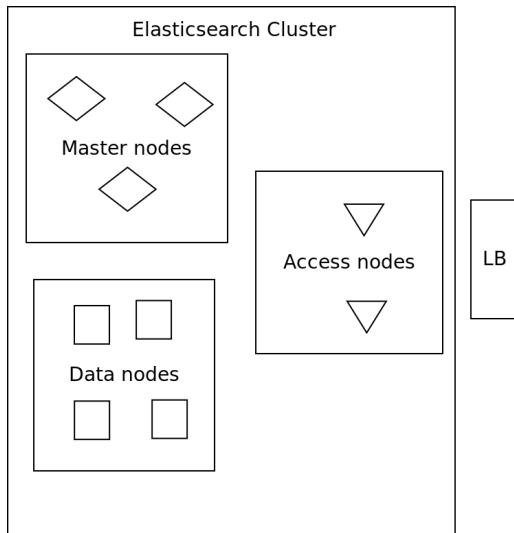
(<https://www.elastic.co/guide/en/elasticsearch/guide/current/practical-scoring-function.html>)

- 1 Master node - keeping, updating, broadcasting state of the cluster (list of nodes, mappings)
- 2 Data node - storing data and executing searches on shard
- 3 Access node - forwarding requests to data nodes and merging results
- 4 Ingest node - preprocess documents before indexing them

Leader election

- 1 zen discovery
- 2 leader election
- 3 quorum

Cluster



Q&A