Core Elasticsearch Training

Lital Kolog - 28-Apr-2015

elastic

Agenda

day 1

- introduction
- data in / data out
- elasticsearch & lucene
- text analysis
- mappings
- _search
- _suggest
- relevancy & boosting

day 2

- aggregations
- document relations
- geo location
- .percolator
- distributed model
- index management
- data management
- moving to production

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Introduction

Lital Kolog - 28-Apr-2015

elastic

What is elasticsearch?

- Document oriented search engine
  - JSON based, Apache Lucene
- Schema Free
  - Yet enables control of it when needed
- Distributed
  - Scales Up+Out, Highly Available
- Multi-tenancy
  - Dynamically create/delete indices
- API centric & RESTful
  - Most functionality is exposed through an API

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

What can it do?

- Unstructured search
  - Find all companies in the ‘search’ domain
- Structured search
  - Find all companies founded between 2010 and 2012
- Analytics
  - Return the average annual revenue for all companies
- Combine
  - Return the average annual revenue for all companies founded between 2010 and 2012 in the ‘search’ domain
- All in (near) realtime

Lital Kolog - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Basic glossary

cluster a cluster consists of one or more nodes which share the same cluster name. Each cluster has a single master node which is chosen automatically by the cluster and which can be replaced automatically if the current master node fails.

node a node is a running instance of elasticsearch which belongs to a cluster. Multiple nodes can be started on a single server for testing purposes, but usually you should have one node per server.

at startup, a node will use multicast (or unicast, if specified) to discover an existing cluster with the same cluster name and will try to join that cluster.

Lital Kolog 28-Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

What's in a distribution?

Executable scripts

Node config files

Data storage

Internal libs

Log files

```
.
├── LICENSE.txt
├── NOTICE.txt
├── README.textile
├── bin
│   ├── elasticsearch
│   ├── elasticsearch.in.sh
│   └── plugin
├── config
│   ├── elasticsearch.yml
│   └── logging.yml
├── data
│   └── elasticsearch
├── lib
│   ├── elasticsearch-x.y.z.jar
│   ├── ...
│   └
└── logs
    ├── elasticsearch.log
    └── elasticsearch_index_search_slowlog.log
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Starting a Node

- Execute

```
# run in the foreground
bin/elasticsearch

# run as a daemon
bin/elasticsearch -d

#log the pid
bin/elasticsearch -p /var/run/elasticsearch.pid
```

- Important env. variables

Lital Kolog - 28-Apr-2015

ES_HOME	The “home” path for the installation
ES_HEAP_SIZE	The JVM heap size
ES_JAVA_OPTS	Additional JVM options

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Configuration

- Node configuration in *config/elasticsearch.yml*

```
cluster.name: test_cluster
network.host: localhost
discovery.zen.ping.multicast.enabled: false
```

- Can be configured with -- parameters when executing the process

```
bin/elasticsearch --cluster.name=test_cluster --network.host=localhost
```

Lital Kolodny - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Paths

path.conf	path to the config directory
path.data	path to the data directory
path.logs	path to the logs directory
path.plugins	path to the plugins directory

Litan's blog - 28-Apr-2015

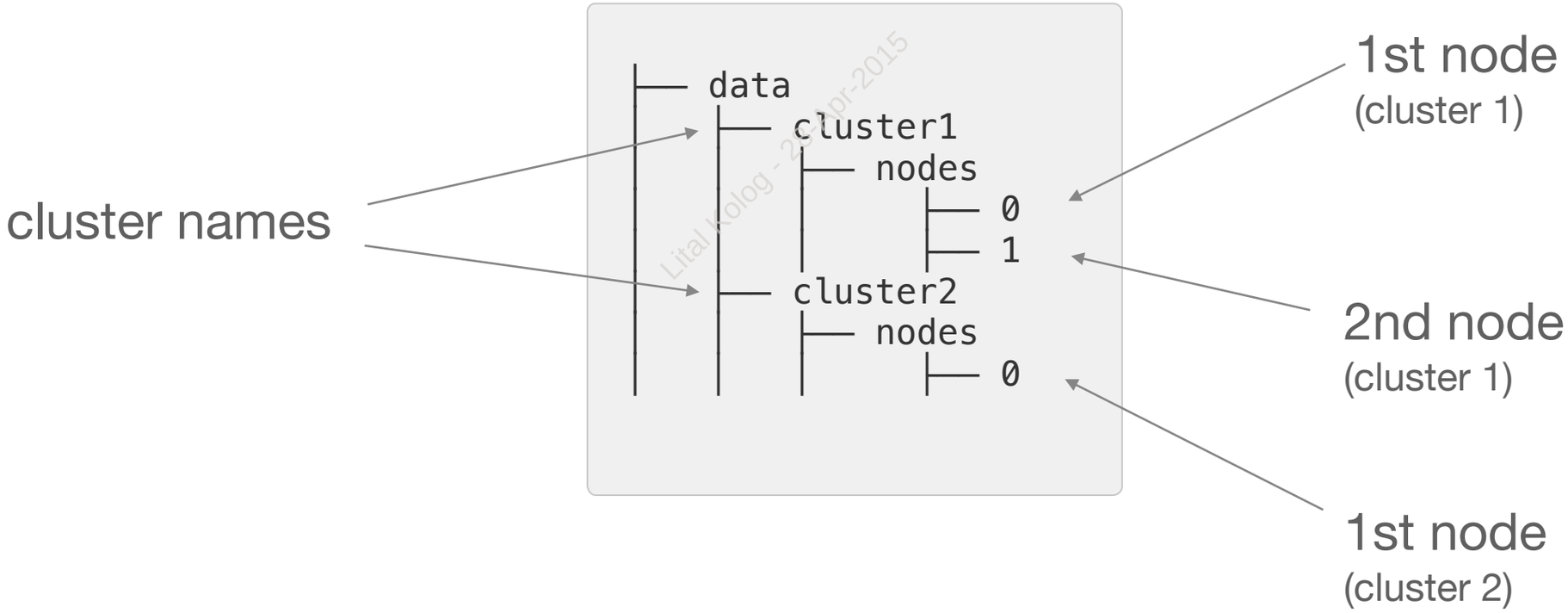
- Sample process execution with custom paths

```
bin/elasticsearch --path.data=/var/lib/elasticsearch \
--path.logs=/var/log/elasticsearch
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Data Path

- Aims at easily running multiple node processes using the same data location
- For example, running 3 nodes with different cluster names



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Logging

- All logging is configured in *config/logging.yml*
  - simple properties based configuration using log4j
  - all logging components map nicely to settings configuration (e.g. *indices.recovery.xxx* setting maps to *indices.recovery* logging component)

tal Kolog - 28-Apr-2015

```
rootLogger: INFO, console, file
logger:
  # log action execution errors for easier debugging
  action: DEBUG

  # custom DEBUG logging for recovery
  indices.recovery: DEBUG
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Important Settings

- Discovery
  - nodes discover each other using *multicast* (default) or *unicast*
  - multicast can be disabled and unicast requires a "seed" list of nodes to connect to

```
discovery.zen.ping.multicast.enabled: false
discovery.zen.ping.unicast.hosts: ["host1", "host2"]
```

Lukáš Kolog - 28-Apr-2015

- Network
  - generic network settings that apply to all network components
  - *network.host* controls both the bind address, and publish address (the address other nodes will talk to the node post discovery)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Important Settings

- HTTP
  - The HTTP transport, by default on ports *[9200-9300)* (it will automatically try and find a free port within the range)
- Transport
  - The internal node to node transport communication, by default on ports *[9300-9400)* (it will automatically try and find a free port within the range).

Vital Kolos - 28-Apr-2015

```
http.port: 9200
transport.tcp.port: 9300
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Plugin Management

- Install
  - from direct URL or file system path

```
bin/plugin -install -url http://your.website.com/path.to.plugin.zip
```

  - from GitHub

```
bin/plugin -install elasticsearch-marvel
```

Lea Kolog - 28-Apr-2015

    » also from *maven* repo & *download.elasticsearch.org*

- Remove

```
bin/plugin -remove elasticsearch-marvel
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Plugins

- Plugins are installed into the plugins location
  - installation simply extracts the zip files under the plugins directory under a "plugin name" directory
- A plugin can optionally have code
  - as libraries that automatically get included under the main plugin directory
- A plugin can optionally have a "*_site*"
  - a set of files that can be served through the HTTP interface
  - during the installation process, if a plugin is not detected to have code associated with it, it is automatically treated as *_site* plugin

Lukáš Kolog - 23-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Basic glossary

index An index can be seen as a named collection of documents. It is a logical namespace which maps to one or more primary shards and can have zero or more replica shards.

shard A shard is a single Apache Lucene instance. It is a low-level "worker" unit which is managed automatically.

Shards are distributed across all nodes in the cluster, and can move automatically from one node to another in the case of node failure, or the addition of new nodes.

There are two types of shards: *primary* and *replica*.

Vital Blog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Basic glossary

Primary shard

An index can have one or more primary shards (defaults to 5) and it is not possible to change this number after index creation. When you index a document, it is first indexed on the primary shard, then on all replicas of this shard.

Replica shard

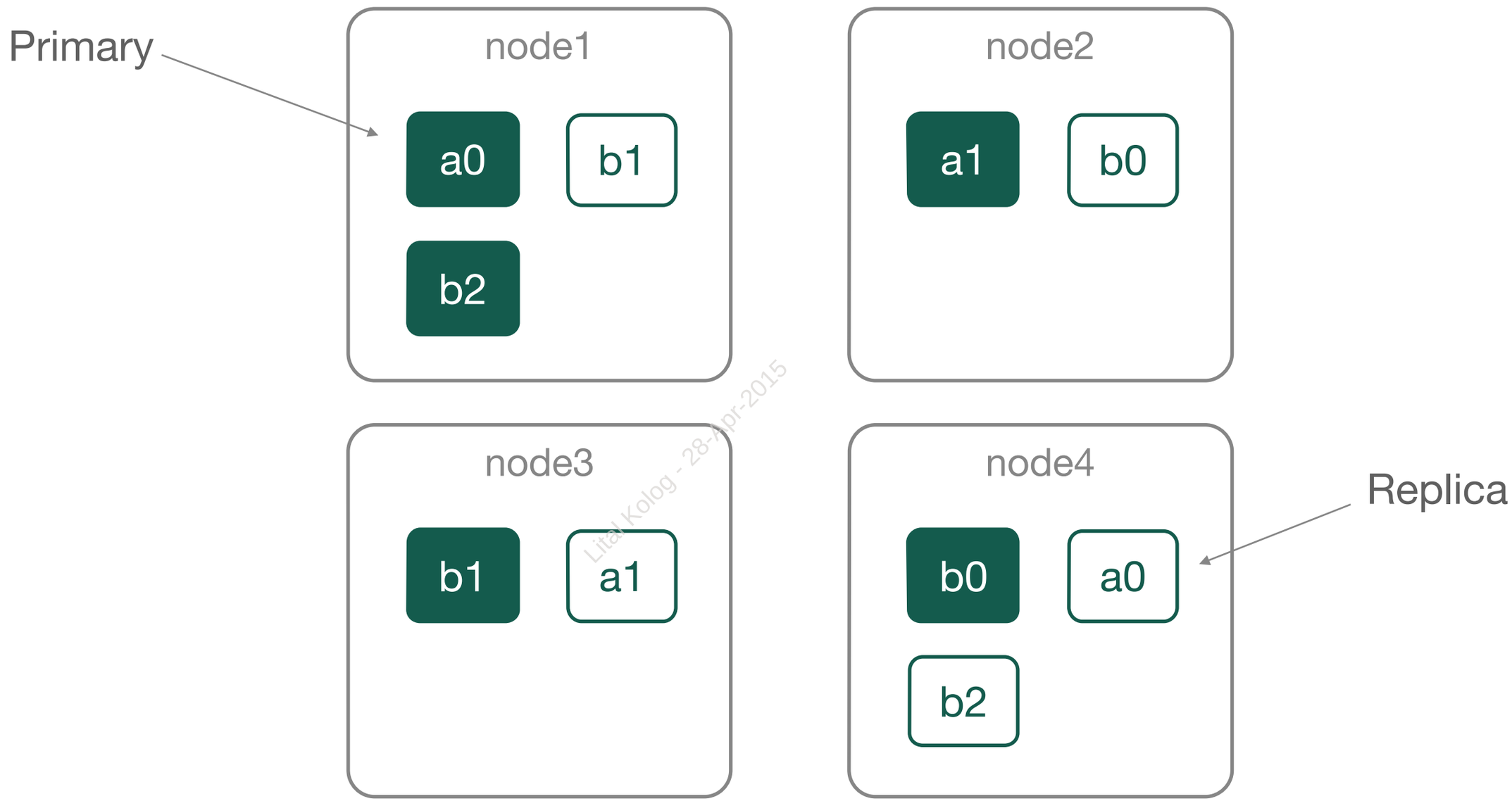
Each primary shard can have zero or more replicas (defaults to 1). A replica is a copy of the primary shard, and serves two purposes:

- Increase high availability - a replica is another copy of the data and will be promoted to a primary shard if the primary fails
- Increase read throughput - get and search requests can be handled by primary or replica shards

Gal Kobi 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

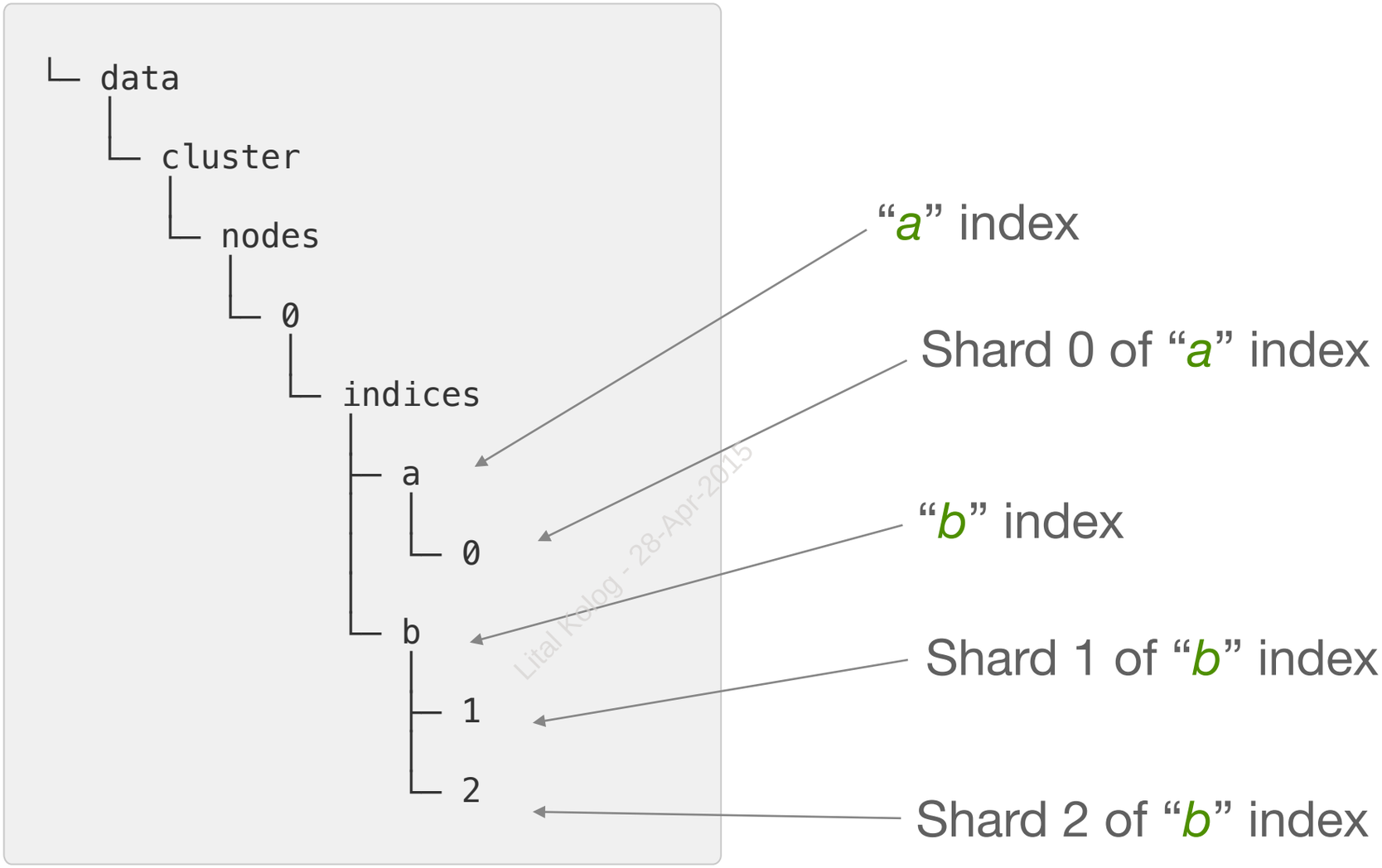
Node/Shard Allocation



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Data Directory Structure



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Data in

Data out

Lital Kolog - 28-Apr-2015

elastic

Data structure

- Documents & Fields
- A document is modeled as a JSON object

```
{
   "id" : "elasticsearch",
   "name" : "elasticsearch",
   "website" : "http://www.elasticsearch.com",
   "category" : "software",
   "founded_year" : 2012,
   "overview" : "The company behind the elasticsearch open source project",
    "tags" : ["search", “datastore", "analytics", "realtime", "scalability"],
   "location" : {
      "city" : "Amsterdam",
      "country_code" : "NL",
         "geo" : {
         "lat" : 52.370176,
         "lon" : 4.895008
      }
   }
}
```

Lital Kolodi 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Core document metadata fields

name	default	description
_id		The id of the document
_type		The document type
_source	enabled	Stores the original document that was indexed
_all	enabled	Indexes all values of all document fields
_timestamp	disabled	timestamp associated with the document
_ttl	disabled	optionally defines an expiration time
_size	disabled	indexes the size of the uncompressed *_source* content

Lital Kolog 28-Apr-201

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Field data types

- Core data types
  - string, number, boolean, datetime, binary (base64)
- Complex types
  - Array, Object
- Other types
  - geo_point, geo_shape, ip, multi-field (see *Mappings*)

Lital Kolog - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Create Index API

HTTP REST operation

target index name

```
curl -XPUT 'localhost:9200/crunchbase' -d '{
   "settings" : {
      ...
   },
   "mappings" : {
      ...
   }
}'
```

Lital Kolog - 28-Apr-2015

optional index configuration

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index API

HTTP REST operation

target index name

document type

```
curl -XPUT ‘localhost:9200/crunchbase/person/1’ -d '{
   "first_name" : "Tony",
   "last_name"  : "Stark"
}'
```

Lital Kolog - 28-Apr-2015

document JSON source

document id

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index API

- Operation response
  - *201 (CREATED)* if it's a newly created doc
  - *200 (OK)* if the doc was updated (replaced/reindexed)

index name

document type

document id

document version

```
{
    "_index" : "crunchbase",
    "_type" : "person",
    "_id" : "1",
    "_version" : 1,
     "created" : true
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index API

- The *_id* part of the url is optional
  - Alternatively can be extracted from a field in the document, or can be auto generated (*POST* vs. *PUT*)
- Index will automatically be created if needed
  - Default settings & dynamic type mappings will be applied and can be disabled at node level configuration
    - If not needed, should probably be disabled in production

Lital Kones - 28-Apr-2015

```
action.auto_create_index: false
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index API

- The index operation is also used for reindexing
  - Based on the id of the document, if a document with the same id already exists, it will be replaced with the new one. Re-indexing is actually delete & index operation.
- explicit _create operation
  - Acts as *index-if-absent* operation. If a document with the same id already exists, the index operation will fail (*409* status code)

Vital Kolos 28-Apr-2015

```
curl -XPUT 'localhost:9200/crunchbase/person/1/_create' -d '{
   "first_name" : "Tony",
   "last_name"  : "Stark"
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index API

- Timestamp
  - It is possible to associate a timestamp with the document by using the timestamp query string parameter

```
curl -XPUT 'localhost:9200/crunchbase/person/1?timestamp=2012-10-15T00:00:00'
```

Lital Kolog - 28-Apr-2015

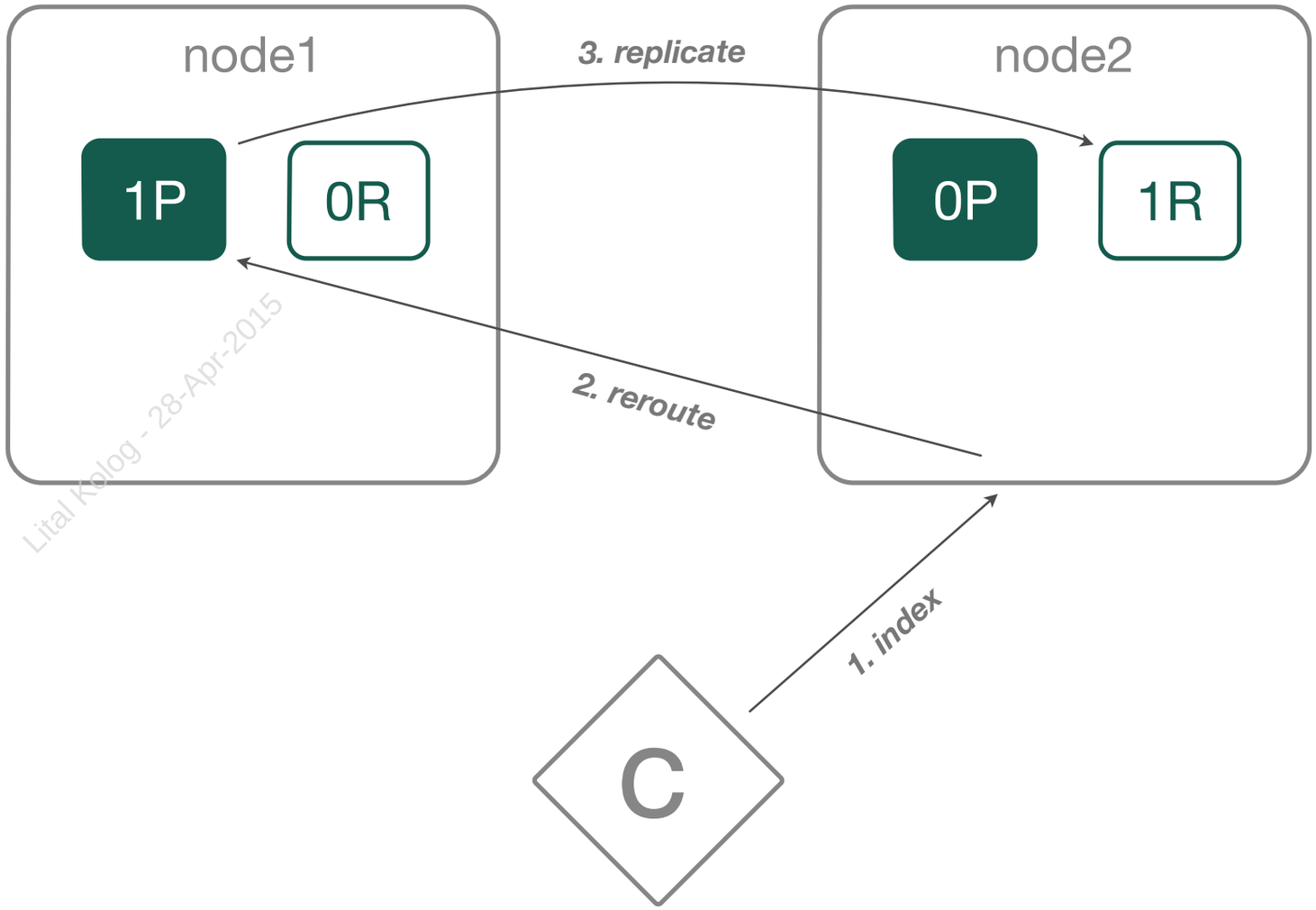
- TTL
  - A document can have an expiration time, which will cause it to be deleted once exceeded. Specified using the *ttl* query string parameter (as date math expression)

```
curl -XPUT ‘localhost:9200/crunchbase/person/1?ttl=1d’
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index API

1. Issue index request
2. Hash the routing value (defaults to *_id*), and pick the matching shard (shard 1). Find the primary shard location for it, and execute / redirect the request to it
3. Once successfully executed on the primary shard, replicate to all replicas



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Get API

- It is possible to retrieve a specific document from the index using its *_type* and *_id*

HTTP REST operation

target index name

document type

document id

```
curl -XGET 'localhost:9200/crunchbase/person/1'
```

Litan's blog - 28-Apr-2015

- The *GET* operation is realtime
  - Meaning, once a document is indexed, it is immediately available to be retrieved using the *GET API*

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Get API

- Operation response
  - *200 (OK)* if exists
  - *404 (NOT FOUND)* if doesn't exist

index name

document type

document id

document version

document source

```
{
   "_index" : "crunchbase",
   "_type" : "person",
   "_id" : "1",
   "_version" : 1,
   "found" : true,
   "_source" : {
      "first_name" : "Tony",
      "last_name" : "Stark"
   }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Get API

- It is also possible to request specific fields using the *_source* parameter

```
curl -XGET 'localhost:9200/crunchbase/person/1?_source=first_name'
```

```
{
   "_index" : "crunchbase",
   "_type" : "person",
   "_id" : "1",
   "_version" : 1,
   "exists" : true,
   "_source" : {
      "first_name" : "Tony"
   }
}
```

Lital Kolog - 28-Apr-2015

Endpoint to return the json document (the source) without the additional metadata

```
curl -XGET 'localhost:9200/crunchbase/person/1/_source'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Get API

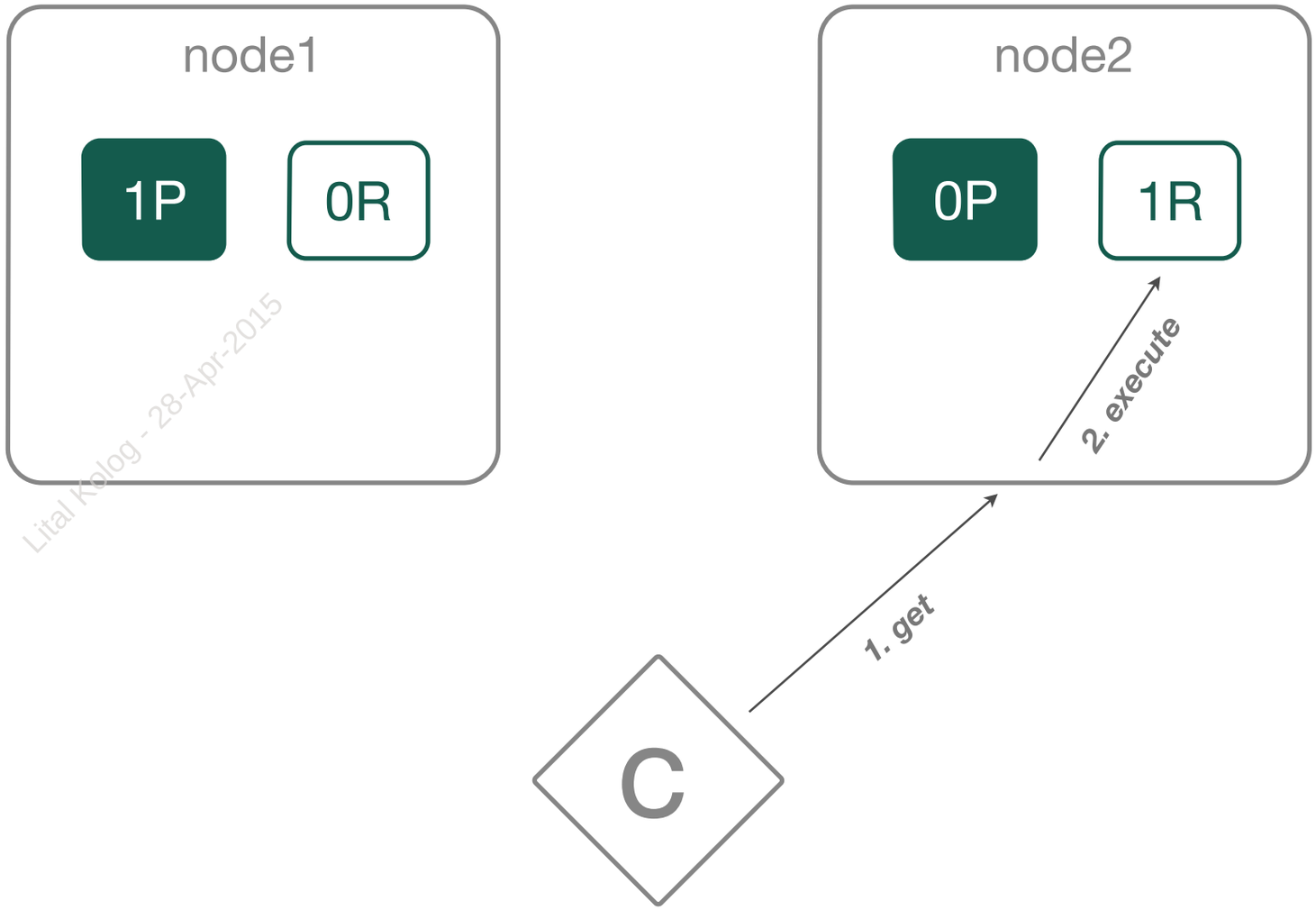
1. Issue get request

2. Hash the routing value (defaults to *_id*), and pick the matching shard (shard 1).

   By default, will round robin between copies of the shards, (primary+replicas) possibly redirecting to another node

   ?preference=_local: prefers to execute locally on a node

   ?preference=_primary: prefers to execute on the primary shard



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Exists API

- Check if a document is in the index
  - Without the overhead of loading it

```
curl -XHEAD -i 'localhost:9200/crunchbase/person/1'
```

- The response is based on HTTP status code
  - *200 (OK)* if exists
  - *404 (NOT FOUND)* if doesn't exist

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Delete API

- Deleting a specific document by _id

```
curl -XDELETE 'localhost:9200/crunchbase/person/1'
```

- Response
  - *200 (OK)* if deleted
  - *404* if not found

Lital Kolog - 28-Apr-2015

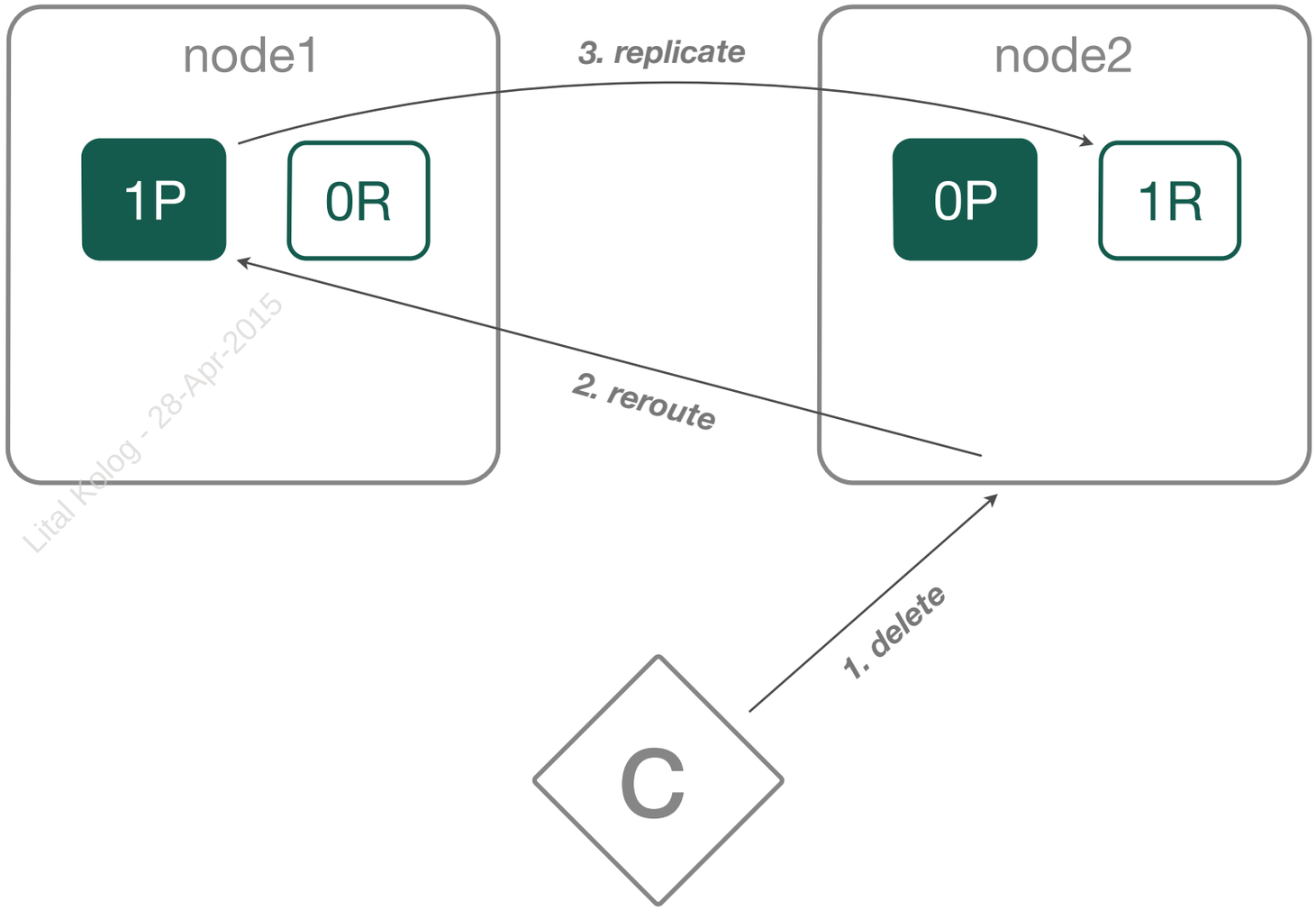
Indication if it was actually found

```
{
   "found" : true,
   "_index" : "test",
   "_type" : "person",
   "_id" : "1",
   "_version" : 3
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Delete API

1. Issue delete request
2. Hash the routing value (defaults to *_id*), and pick the matching shard (shard 1). Find the primary shard location for it, and execute / redirect the request to it.
3. Once successfully executed on the primary shard, replicate to all replicas.



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Document Versioning

- Every document is versioned
  - Enables optimistic concurrency control on a single document level which supports use cases such as transactional *read-then-write* operations.
- A version is assigned to a document on creation
  - By default, the initial version of a document is 1, but it is possible to explicitly specify a version number when creating
- Each write operation increments the version
  - Including: *reindex*, *update* & *delete* operations
- Version numbers can also come from external systems

Lital Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Document Versioning

- Each write operation returns a version
  - This is the version of the document, after the operation execution
- Each write operation can accept a version
  - This is the document version number which the operation is expected to execute on. If the specified version and the current document version don't match, the operation fails.

Lital Kolodny 28-Apr-2015

```
curl -XPUT localhost:9200/crunchbase/person/1?version=1 -d '{...}'
```

```
curl -XDELETE localhost:9200/crunchbase/person/1?version=1
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Update API

- Updates a document using partial data or script
  - Internal *get-then-reindex* operation, therefore minimizes version conflicts and network roundtrips
  - *retry_on_conflict* specifies number of times to retry an idempotent update
  - The update will be retried if there is a version conflict between the *get* and the *update* (another operation "sneaked" in)
  - All index related parameters are supported (*ttl*, *timestamp*, *routing*, ...)
  - Only works if *_source* is stored (the default)

Lital Kolodny 28-Apr-2016

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Update API

- Update by partial data

```
curl -XPOST 'localhost:9200/crunchbase/person/1/_update' -d '{
    "doc" : {
        "first_name" : "Tony"
    }
}'
```

  - Partial doc is merged with existing doc

ital Konas 28-Apr-2015

Non-object properties with the same key are replaced.
Object properties are recursively merged

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Update API

- Update by script

```
curl -XPOST localhost:9200/crunchbase/person/1/_update -d '{
   "script" : "ctx._source.intelligence_level = 7"
}'
```

  – Supports named parameters

Lital Kuntz 28-Apr-2015

```
curl -XPOST localhost:9200/crunchbase/person/1/_update -d '{
   "script" : "ctx._source.intelligence_level = level",
   "params" : {
      "level" : 7
   }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Update API

- Fails if document does not exists
- *upsert* used to create a document if it doesn't already exist

```
curl -XPOST localhost:9200/crunchbase/person/1/_update -d '{
    "script" : "ctx._source.counter += 1",
    "upsert" : {
        "counter" : 1
    }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

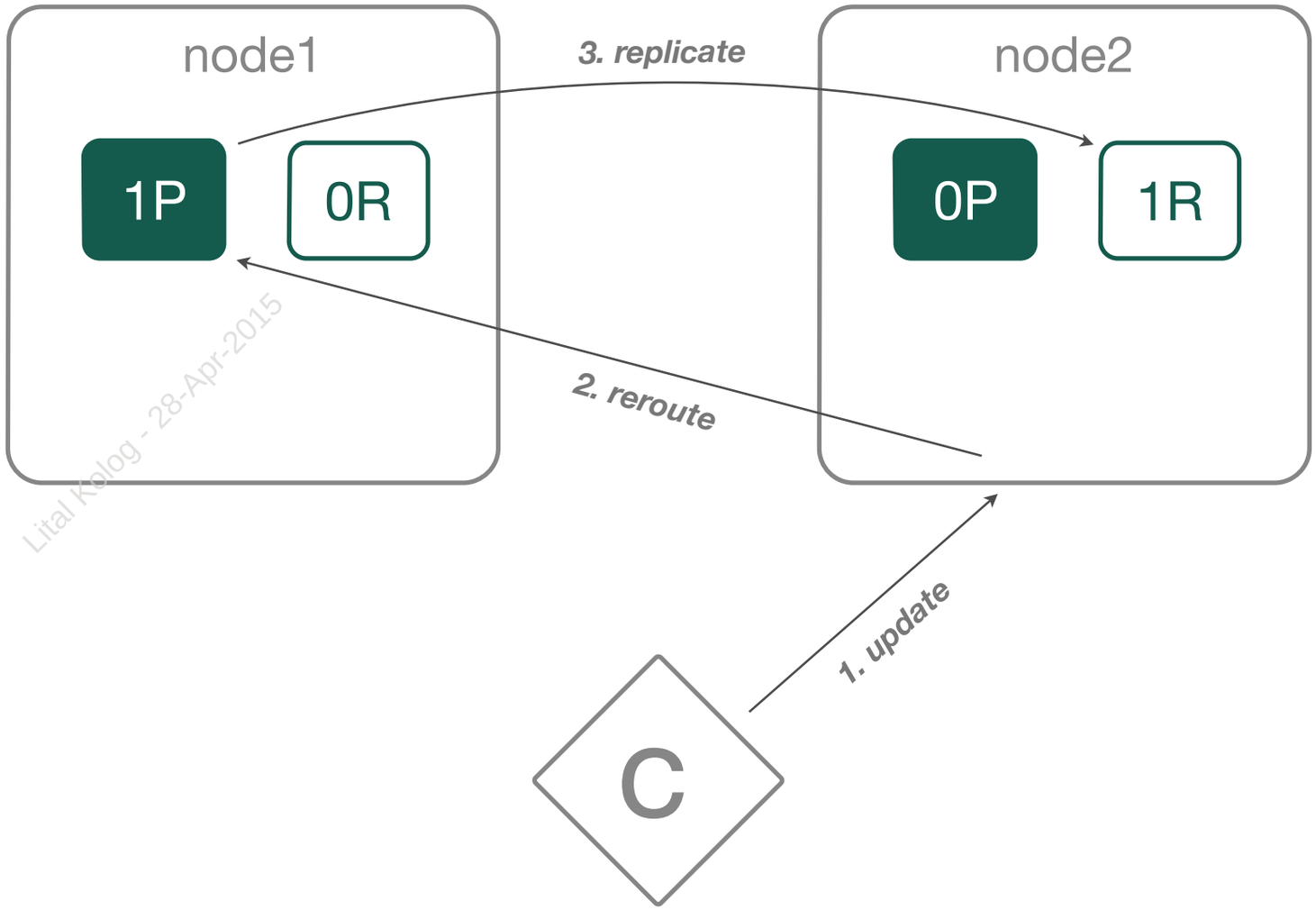
Update API

1. Issue update request

2. Hash the routing value (defaults to *_id*), and pick the matching shard (shard 1). Find the primary shard location for it, and execute / redirect the request to it.

   Get the document (*_source*) from the primary shard, update it, index it back, handling versioning conflicts

3. Once successfully executed on the primary, replicate to all replicas.



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multi Get API

- Sometimes you'd like to get multiple documents in one go
  - Avoid round trips when using the Get API

index name

document type

document id

Optionally specify what fields should be returned

```
curl 'localhost:9200/_mget' -d '{
    "docs" : [
        {
            "_index" : "crunchbase",
            "_type" : "person",
            "_id" : "1"
        },
        {
            "_index" : "marvels",
            "_type" : "hero",
            "_id" : "2"
          "_source" : [ "first_name" ]
        }
    ]
}'
```

Lital Kolog - 28-Apr-2016

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multi Get API

- Also possible to target a specific index

```
curl 'localhost:9200/crunchbase/_mget' -d '{
   "docs" : [
      {
         "_type" : "person",
         "_id" : "1"
      },
      ...
   ]
}'
```

Lital Kolog - 28-Apr-2015

- Or specific type

```
curl 'localhost:9200/crunchbase/person/_mget' -d '{
    "ids" : ["1", "2"]
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bulk API

- Minimizes round trips when performing bulk *index*/*delete*/*update* operations
- The format of a bulk request is as follows

action metadata

```
{ "delete" : { "_index" : "crunchbase", "_type" : "person", "_id" : "2" } }\n
{ "index" : { "_index" : "crunchbase", "_type" : "person", "_id" : "1" } }\n
{ "first_name" : "Tony", "last_name" : "Stark" }\n
.
.
.
{ "create" : { "_index" : "crunchbase", "_type" : "person", "_id" : "3" } }\n
{ "first_name" : "Thor", "last_name" : "Odinson" }\n
```

optional action body

each line must end with a line break (incl. the last line)

Lital Kohen 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bulk API

- What's up with the line break?
  - Enables sending multiple JSON objects at once which allows elasticsearch to only parse the actions metadata and defer the document parsing to the target shards
- The target URL and endpoint

```
curl -XPUT 'localhost:9200/_bulk'
```

Lital Kolog - 28-Apr-2015

```
curl -XPUT 'localhost:9200/crunchbase/_bulk'
```

```
curl -XPUT 'localhost:9200/crunchbase/person/_bulk'
```

When the index and/or doc type are specified in the url, they serve as default fallbacks

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bulk API

- Response structure
  - Contains individual response for each individual request action
  - Order is guaranteed, i.e. position 5 in response will be the output for bulk action in position 5 in the request
  - Failure of a single bulk item action will not affect other actions!
  - Each response item includes the *_index*, *_type*, and *_id*, and either the action response, or an *error* element

ital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bulk API

name	description
_version	The expected doc version of the index/delete action
_timestamp	To be used as the timestamp of the indexed document
_ttl	Expiration date/time for the indexed document
_parent	(will be covered later in *document relations*)
_routing	(will be covered later in *index management*)

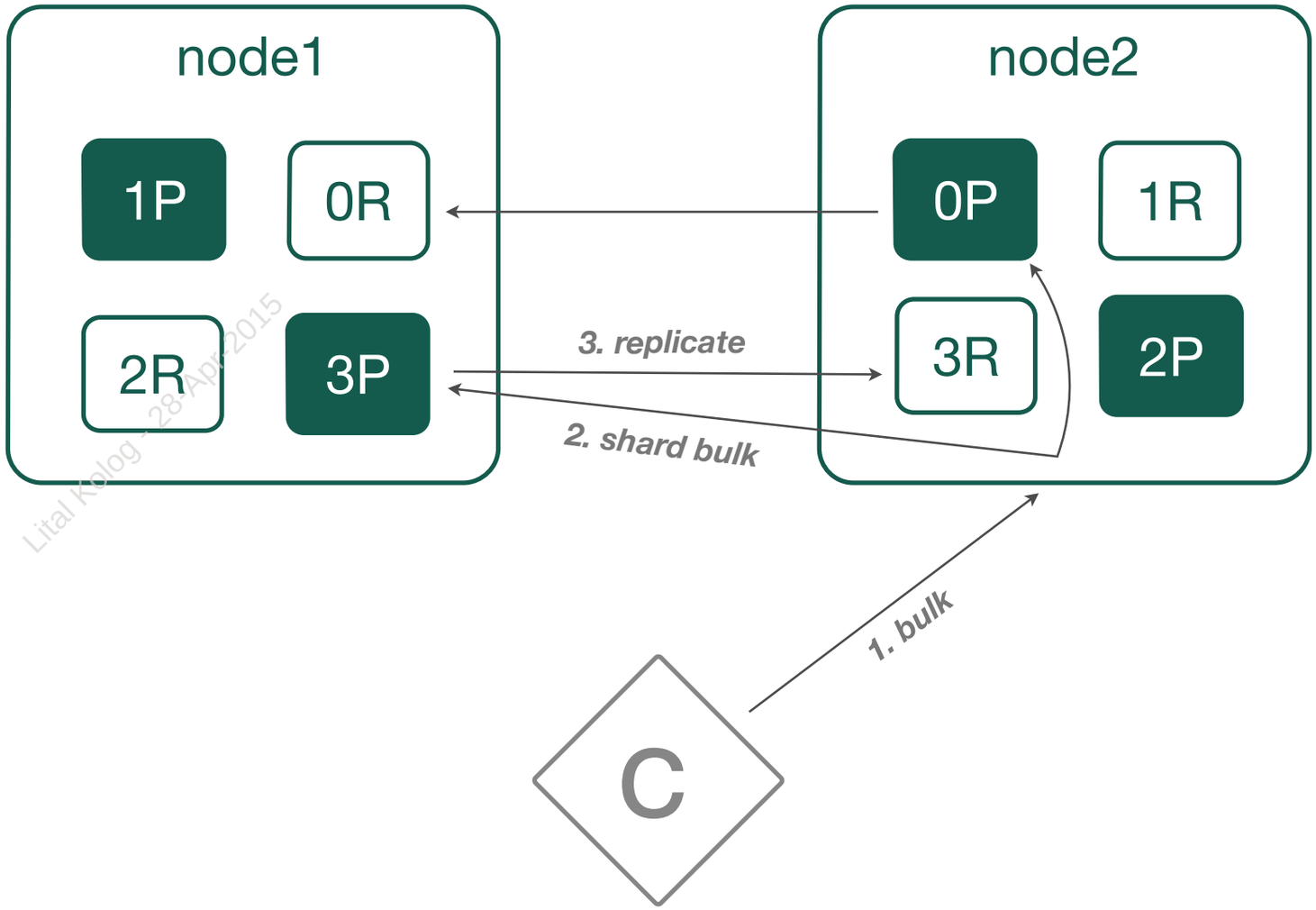
Lital Kologlu 13-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bulk API

1. Issue bulk request
2. Hash the routing of each item, building shard level bulk requests

   Issue shard level bulks on applicable shards
3. Replicate to all replica shards only the successful items executed on the primary shards



Lital Kolog - 28 Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search API

- Finding document based on free text search

```
curl -XGET 'localhost:9200/marvels/_search?q=tony'
```

- Response

Header

query time

did the search time out?

How many shards successfully replied

```
{
  "took" : 5,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "failed" : 0
  },
  ...
}
```

Paul Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search API

Hits

Total matches

All hits of the first “page”

The document source

```
{
   ...

   "hits" : {
      "total" : 1,
         "max_score" : 0.3125,
         "hits" : [
         {
               "_index" : "marvels",
               "_type" : "hero",
               "_id" : "ironman",
               "_score" : 0.3125,
               "_source" : {
               "first_name" : "Tony",
               "last_name" : "Stark",
               }
         }
         ]
   }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search API - Query DSL

```
curl -XGET 'localhost:9200/_search?pretty' -d '{
  "query" : {
    "bool" : {
      "must" : [
        {
          "match" : {
            "name" : "tony thor"
          }
        },
        {
          "range" : {
            "intelligence" : { "gte" : 6 }
          }
        }
      ]
    }
  }
}'
```

Lital Kolod 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search API

- Other supported features
  - execution modes (search types)
  - sorting & pagination
  - extensive query types
  - cacheable filters
  - highlighting
  - (all will be covered later in the _*search* section)

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search API

- Control your search context
  - Search within a specific type

```
curl -XPOST 'localhost:9200/index1/type1/_search' -d {...}
```

  - Search across multiple indices (wildcards supported)

```
curl -XPOST 'localhost:9200/index1,index2/type1/_search' -d {...}
```

  - Search within multiple types

```
curl -XPOST 'localhost:9200/index1/type1,type2/_search' -d {...}
```

  - Search across all indices

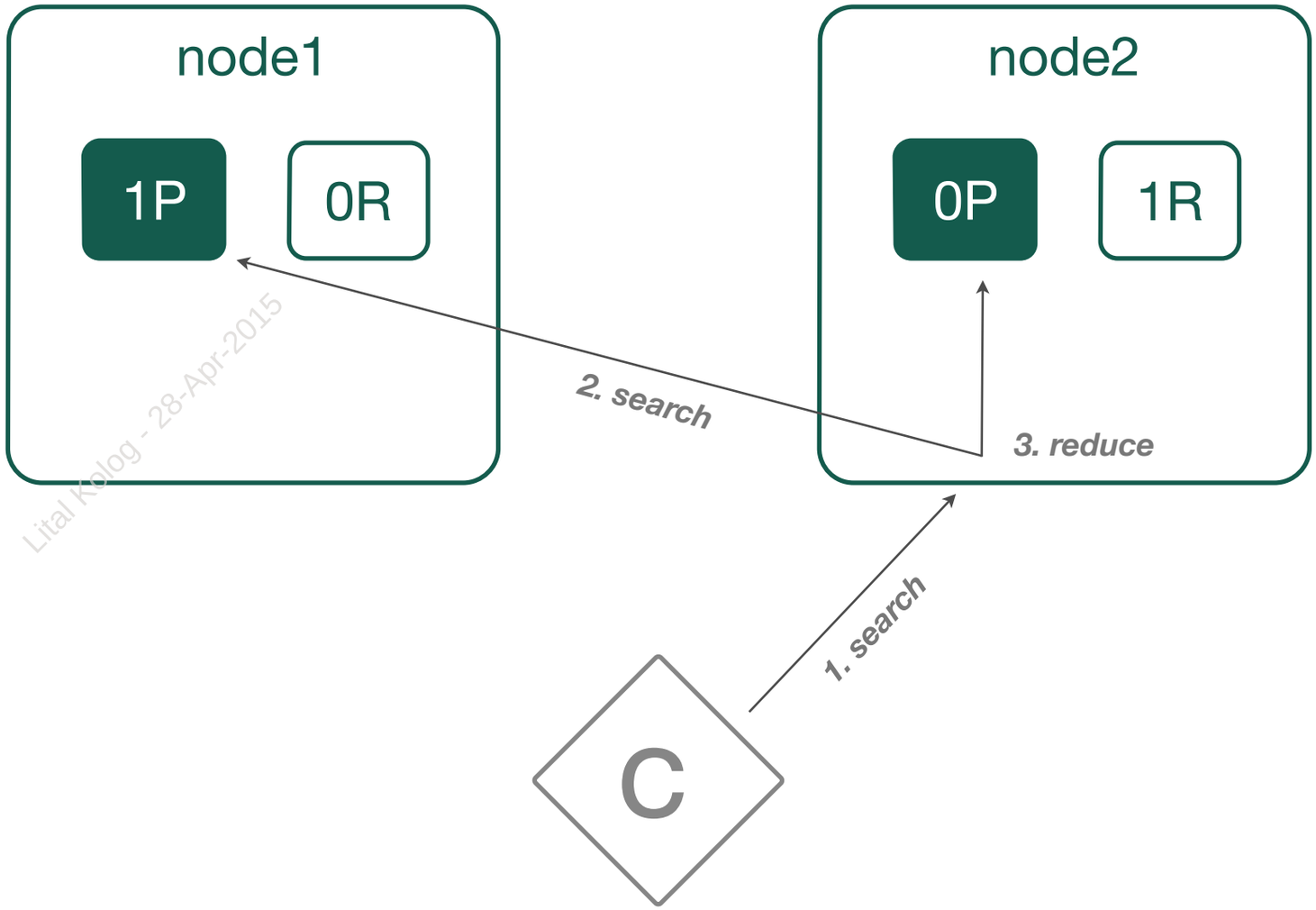
```
curl -XPOST 'localhost:9200/_all/type1/_search' -d {...}
```

Lital Kozlovski - 28 April 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search API

1. Issue search request
2. Find the relevant shards matching the request, and execute shard level search requests on them
3. Reduce the results of the search requests, possibly doing another round trip to relevant shards to fetch additional data



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Elasticsearch & Lucene

Lital Kolog - 28-Apr-2015

elastic

Grep!

- Which plays contain the word "Brutus" in the complete works of Shakespeare?
  - Grep it!
  - Go over each play, word by word, and mark the play that contains it
- Linear to the number of words
- Fails at large scale

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Inverted Index

- Inverting Shakespeare
  - Take all the plays and break them down word by word
  - For each word, store the ids of the documents that contain it
  - Sort all tokens (words)
- Search
  - First look for the relevant word (fast as words are sorted), if found, iterate over the document ids that are associated it

Lital Kozlov - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Inverted Index

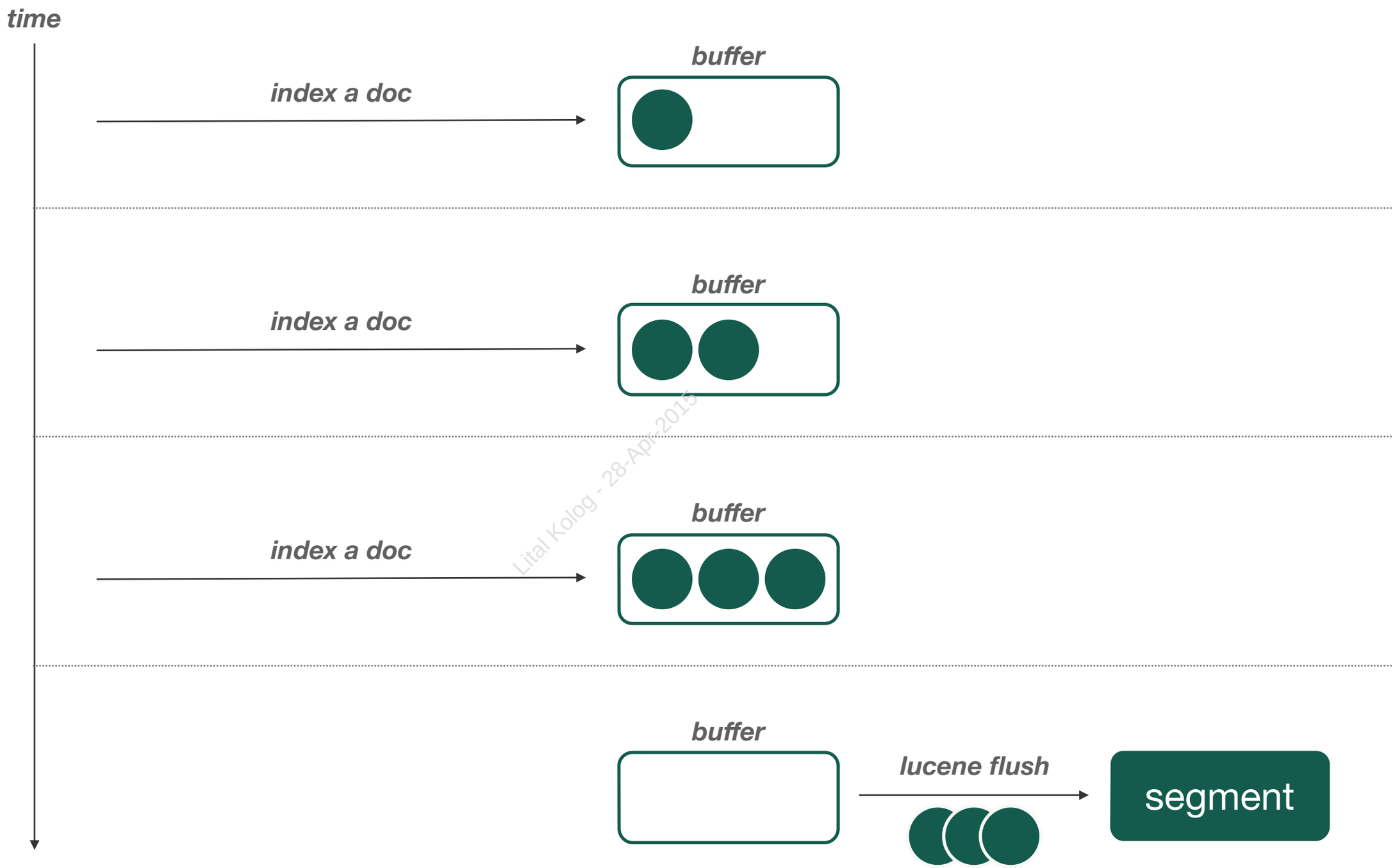
- Allows for boolean retrieval model
  - Relatively simple and extremely efficient (think intersections)
- Each token contains metadata
  - Plays an important role later in different scoring models

Lital Kolog 23-Apr-2015

token	doc freq.	postings (doc ids)
Anthony	2	1, 2
Brutus	1	5
Caesar	2	2, 3
Calpurnia	2	4, 5

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

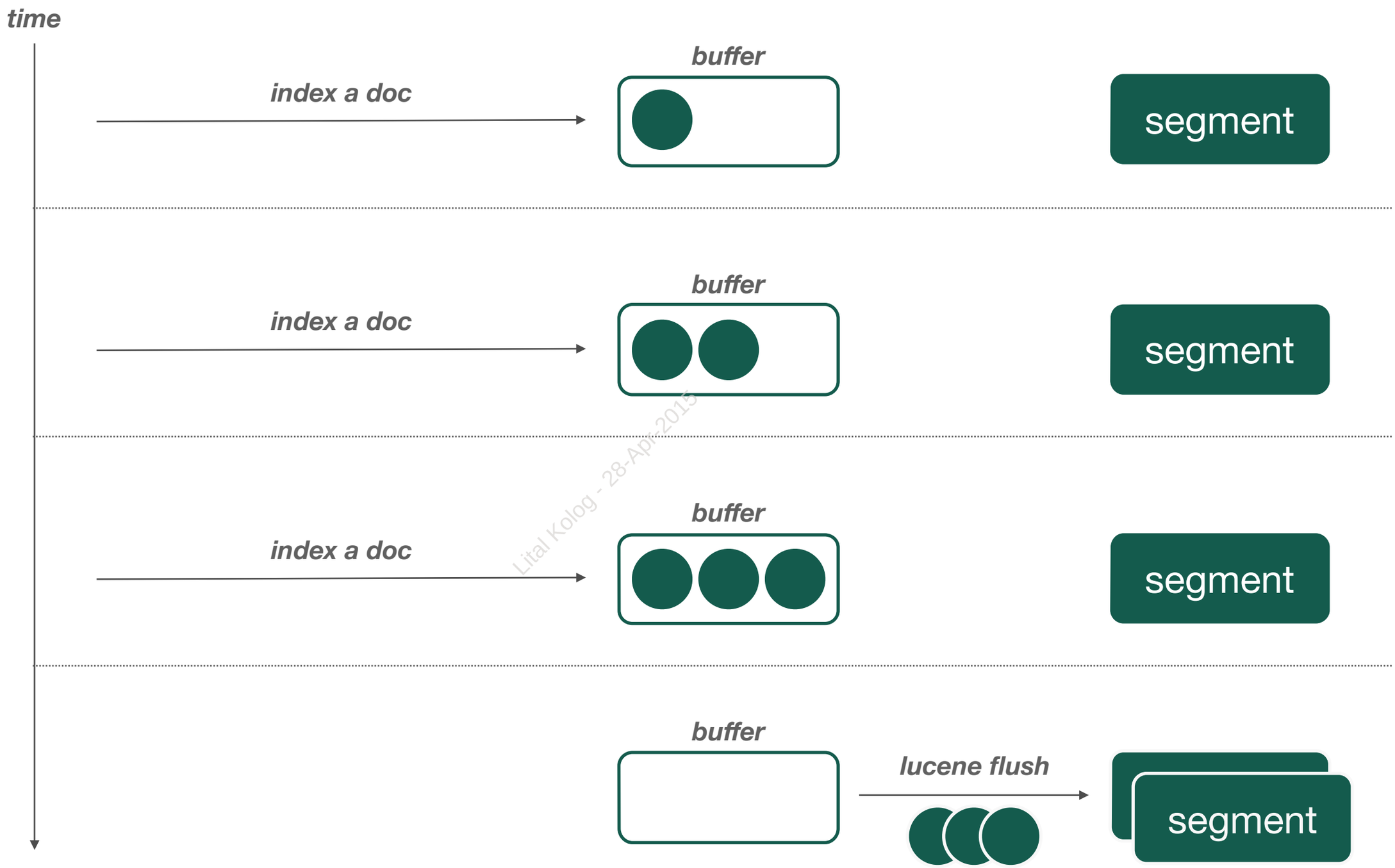
Lucene - Indexing



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Lucene - Indexing



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Lucene indexing

- Each indexed document is analyzed and kept in an in memory buffer
- A buffer is *lucene* flushed to a segment:
  - When the memory buffer is full
  - When new changes needs to be searched ("lucene reopen"/*elasticsearch* refresh)
  - On a *lucene* commit to fully persist the changes (including other non committed segments)
- A segment is a self sufficient immutable inverted index
  - All files belonging to a segment are written once and never rewritten again (no write-in-place)

Vital Kokin 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

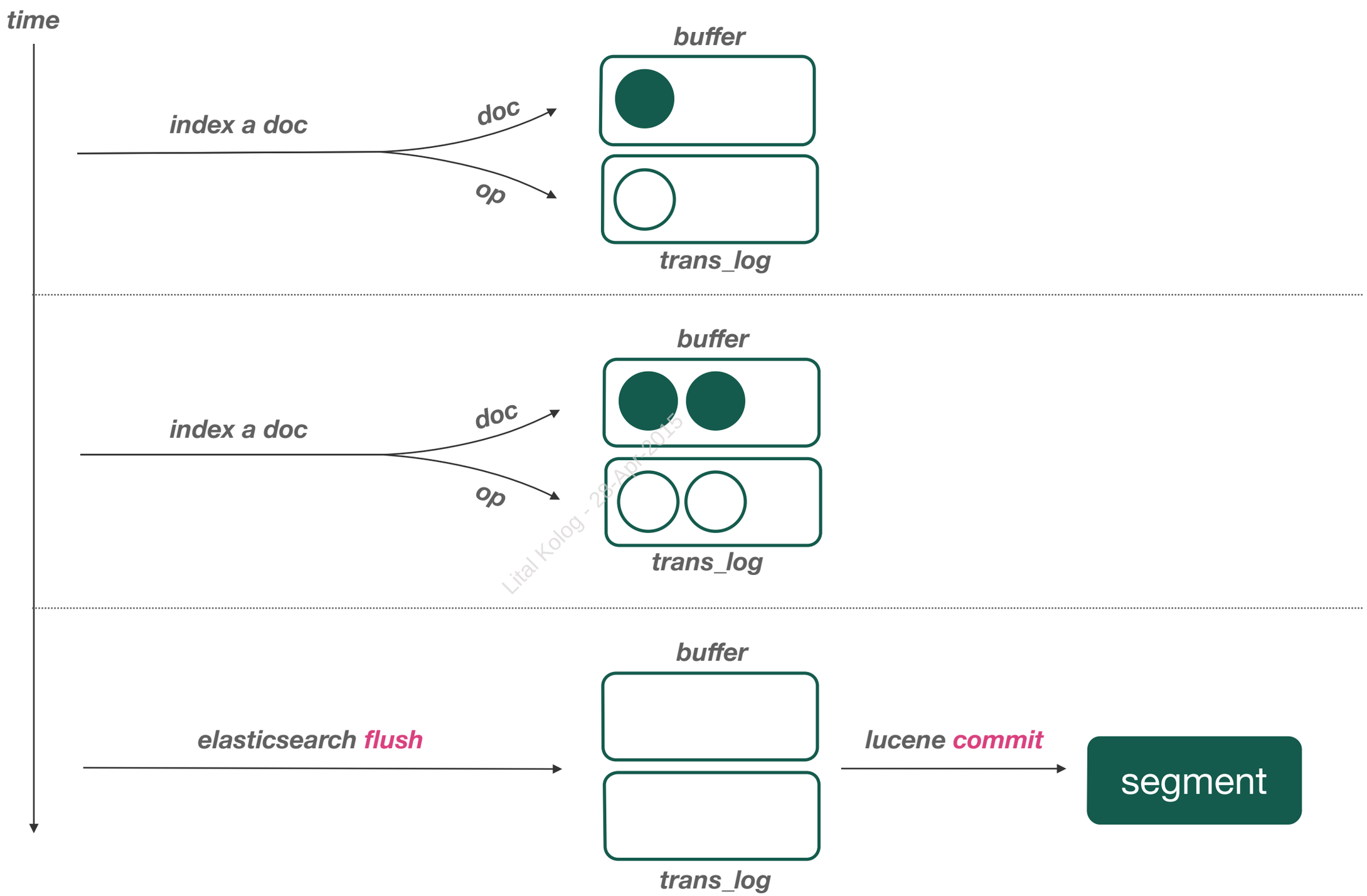
The Need for a Transaction Log

- Changes are persisted in *Lucene* on a commit
  - Expensive! requires flushing all in memory changes and fsync'ing all the segments files to disk
- Data not committed can be lost on a crash
  - Or even simple process "exit"
- Can't *lucene* commit on each operation
  - Did we mention its expensive?

Lital Kotler - 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Transaction Log



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Transaction Log

- Added by elasticsearch
- Each operation is recorded in a transaction log
- An elasticsearch "flush" will
  - Commit the lucene index
  - Clear the transaction log
- The transaction log guarantees durability
  - Operations are automatically replayed when a shard is reopened
- It also simplifies shard relocation/recovery
  - when moving a shard from one node to another
  - replay changes while transferring committed segments

Lital Kesing - 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Flush - elasticsearch

- The process of
  - Issuing a *Lucene* commit & clearing the translog
- Needed to keep the translog "at bay"
  - Too big, and replaying it will take a long time...
- Automatically managed by *elasticsearch*
  - Based on the number of operations in the translog, its size, and the period since it was last flushed
  - All have sensible defaults, and changing these should be considered as expert settings!
- Exposed through an API

```
curl -XPOST 'localhost:9200/crunchbase/_flush'
```

Lital Kobay - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Refresh - elasticsearch

- Make latest changes visible for searches
- New lucene segments may be created (out of the buffer)
  - But not committed!, more light weight
- Exposed through an API

```
curl -XPOST 'localhost:9200/crunchbase/_refresh'
```

- Controlled by index.refresh_interval setting (defaults to 1s)
  - can be changed dynamically on a live index/indices using the update settings API

Lital Kozlovski - 28-Apr-2015

```
curl -XPOST 'localhost:9200/crunchbase/_settings?index.refresh_interval=5s'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

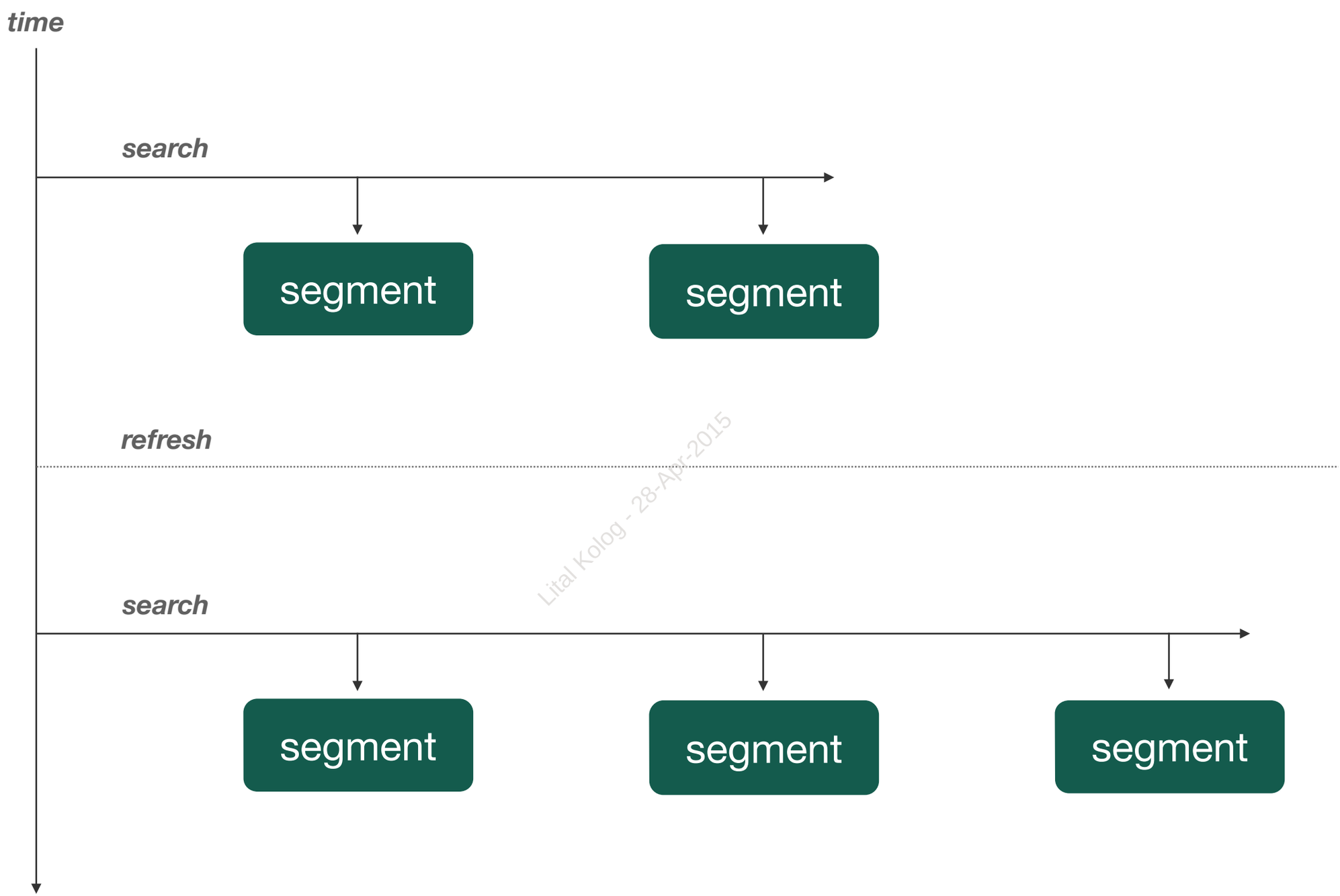
Segments & Search

- A *Lucene* index is composed of N segments
- Each segment is immutable
  - Great concurrency model (no locks) and also accounts for optimal utilization of the file system cache
- Each segment is a fully functional inverted index
- Search is executed on a point-in-time view of the index
  - This view is largely defined by the searchable segments and on every refresh operation the point-in-time view is refreshed with the latest segments
- Search is executed sequentially, one segment at a time

Vital Kuala 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Segments & Search



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Deletes

- *Lucene* does not do update-in-place
  - An update translates to an atomic "delete + add" operation
- Deletes are "soft"
  - When a document is deleted it is only "marked as deleted" and it doesn't go away immediately. *Lucene* maintains a list of deleted documents per segment. Using generational deleted docs files, the segments maintain their immutable nature

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Merge - Keeping Segments at Bay

- Segments keep on being created
- Can't have infinite number of segments
  - Each segment comes with its resources cost (file handles, caches/memory, etc…) and remember that search is sequential across segments
- Let's merge segments
  - Take several segments and merge them into a new (immutable) segment, and while at it, clean up the "garbage" (remove deleted docs)

Vital Korea - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Merge Process

- Segments are merged periodically
- Merge Policy
  - Controls which segments ends up being merged, and how
  - Can be controlled on index creation
  - Several types, defaults to tiered merge policy
  - Merge policy settings can be set at index creation time and changed dynamically on a live index, expert settings!

Lital Kuzog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Merge Throttling

- Merges can be expensive
  - Elasticsearch allows to throttle them on the node level (which is agnostic to number of shards allocated on the node). Configure it by setting:
    - *indices.store.throttle.type* to merge
    - *indices.store.throttle.max_bytes_per_sec* to *5mb*
      - defaults to *20mb*
- Note, throttling affects both IO and CPU cost of a merge
  - Slower merges do less IO, and also less CPU

ital Konig - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Optimize API - Explicit Merge

- One can explicitly request for an explicit merge of an index
  - The *optimize* API merges all segments down to a designated maximum

```
curl -XPOST 'localhost:9200/crunchbase/_optimize?max_num_segments=2'
```

- Handy when
  - Completed heavy index of initial data
  - An index will stop receiving (frequent) changes (e.g. time based index)
- Be careful
  - Avoid triggering explicit merges (optimizing) on frequently changing indices - the merge policy does a great job at balancing active segments

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Optimize API - Explicit Merge

- Be careful
  - Avoid triggering explicit merges (optimizing) on frequently changing indices - the merge policy does a great job at balancing active segments

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Segments API

```
curl -XGET 'localhost:9200/crunchbase/_segments'
```

```
{
  "ok" : true,
  "indices" : {
    "crunchbase" : {
      "shards" : {
        "0" : [{
          "routing" : { ... },
          "num_committed_segments" : 18,
          "num_search_segments" : 21,
          "segments" : {
            "_ye" : {
              "generation" : 1238,
              "num_docs" : 1218810,
              "deleted_docs" : 58863,
              "size" : "561.7mb",
              "size_in_bytes" : 589074752,
              "committed" : true,
              "search" : true
            },
            ...
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- All writes (segments and translog) are sequential
  - a.k.a append only
- All files created are immutable
- Perfect for file system cache
  - Less cache churn concurrency and many operations are "disk free" as they are read directly from the file system cache

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- Compared
  - Most popular data storage engines rewrite modified data in place: *MySQL* (*InnoDB*), *PostgresSQL*, *Oracle*.
    - B-Tree based databases.
  - Most perform buffering of writes before writing to disk
  - ... but flushes are still random

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- Spinning disks
  - Cheap (*$0.08/GB*)
  - Seek time is limited by the time it takes for drive to rotate

```
IOPS = Bytes per Second / Operation Data Size
```

Lital Kolog 28-Apr-2015

  - *7,200 RPM = ~80 IOPS*
  - *10,000 RPM = ~120 IOPS*
  - *15,000 RPM* = up to *~180 IOPS* (the max for decades)
  - Sequential operations are best:
    - *125MB/sec* for modern SATA drives

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD
  - Affordable: *~$1.75/GB*
  - Massive IOPS: *40,000/sec* read, *23,000/sec* write
  - Latency is less than *100μs*
  - Good sequential throughput: *270MB/sec* read, *205MB/sec* write
  - Way cheaper per IOPS: *$0.02* vs *$1.25*

Lital Kolos 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

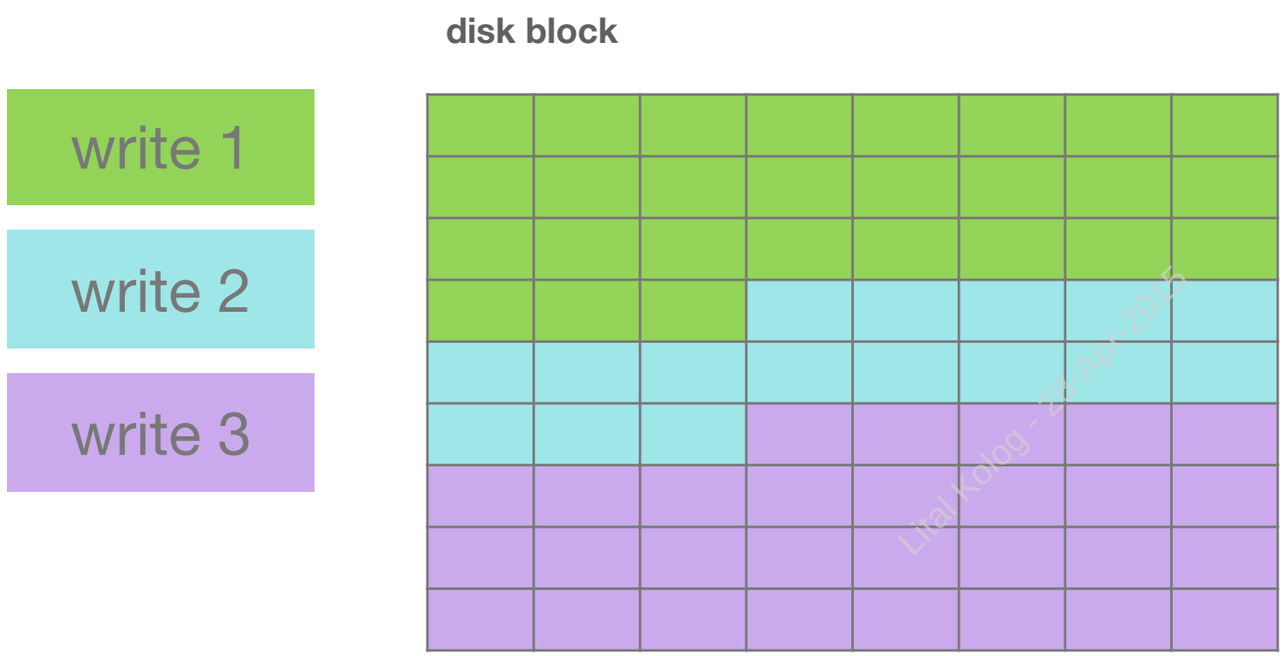
- SSD … BUT
  - Cannot overwrite directly: must erase first, then write
  - Can write in small pages (*4kb*), but only erase in *~512KB* blocks
  - Latency: write is *~100μs*, erase is *~2ms*
  - Limited durability: *~5000* cycles for each erased block

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD Wear Leveling
  - A disk page with 3 writes

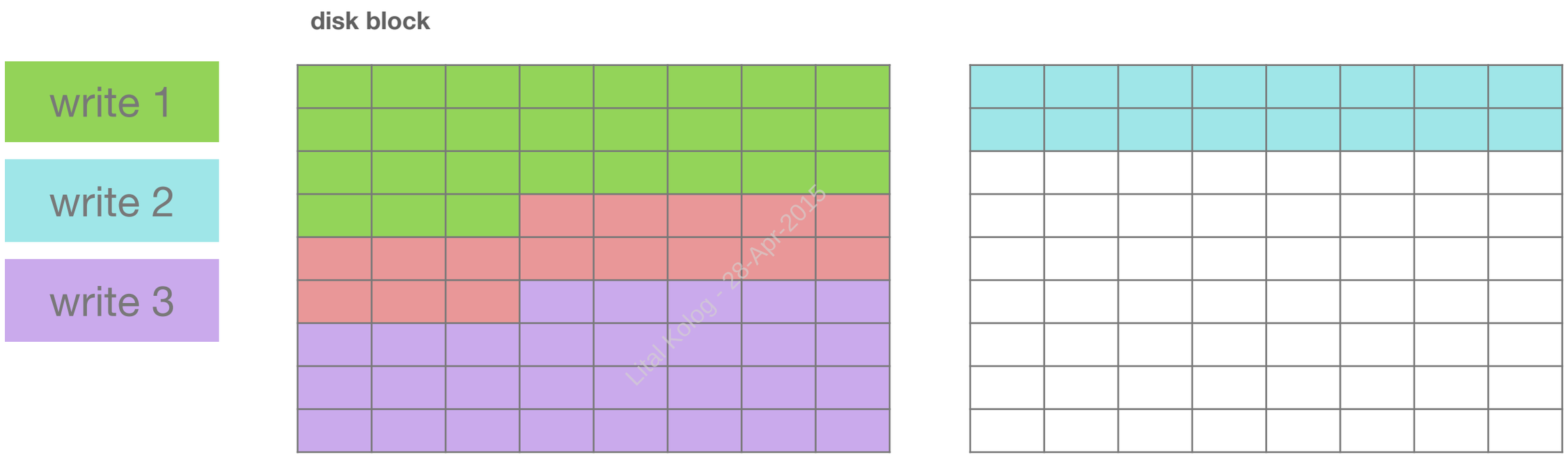


Lital Kolog - 29 Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD Wear Leveling
  - Modify data from write 2, mark as garbage, write to new block

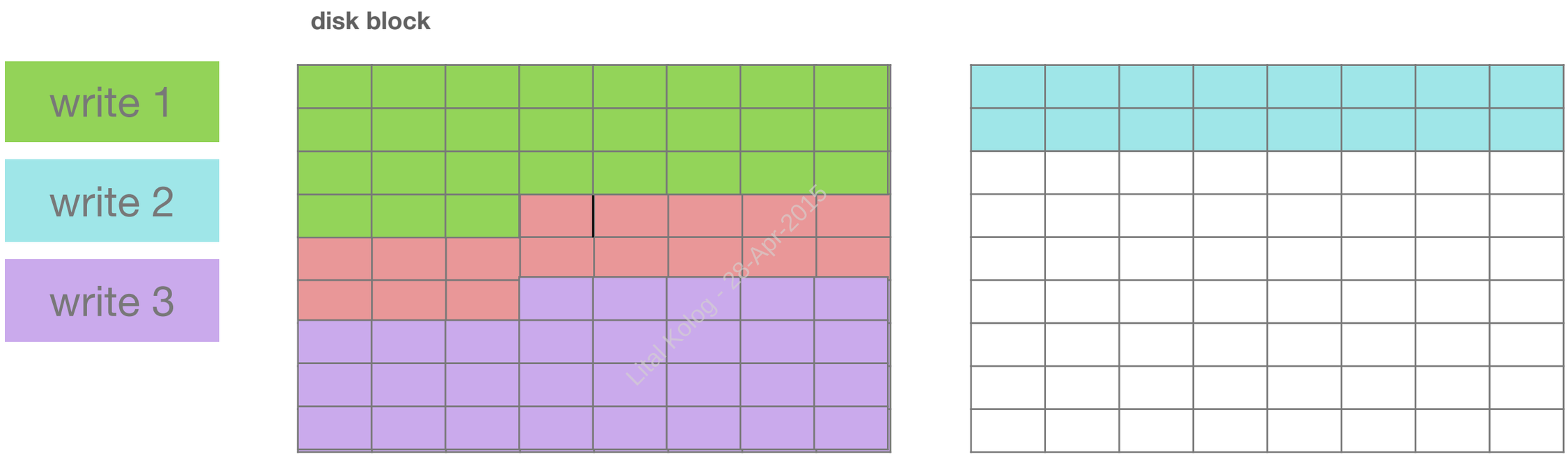


Lilial Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD Wear Leveling
  - Garbage? That means, garbage collection....



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD: Write Amplification
  - When only a few kilobytes are written, but fragmentation causes a whole block to be rewritten
  - The smaller & more random writes, the worse it gets
  - Modern "mark & sweep" GC reduces it, but can't eliminate it
  - *TRIM* allows the OS to actively tell the drive when a region of disk is no longer used and by that reduce the number of GCs (but not eliminate them completely)

Lital Kolko - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD: Write Amplification
  - *AnandTech* estimates that modern SSDs only last about *1.5* years under heavy *MySQL* load, which causes around *10x* write amplification

Intel SSD 320 - 120GB - SSD Longevity		
	MS SQL Server (Main Site DB)	MySQL Server (Forums/Ads DB)
Total Bytes Written (9 days)	576GB	1030GB
Estimated Write Amplification	10x	10x
Drive Capacity	120GB	120GB
Cycles Used	48	85.8
Cycles Used per Day	5.33	9.53
Worst Case P/E Cycles Available	5000	5000
Estimated Lifespan (Days)	937.5	524.3
Estimated Life (Years)	2.57 years	1.44 years

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Segments, Translog and Disk

- SSD
  - Remember this: In *Elasticsearch/Lucene*, all writes are sequential
  - "For sequential write workload, write amplification is equal to *1*, i.e., there is no write amplification"

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Text Analysis

Lital Kolog - 28-Apr-2015

elastic

The need for text analysis

id	content
1	The quick brown fox jumps over the lazy dog
2	Fast jumping spiders

- Break into tokens
- Index them by doc id

Lital Kdog - 28-Apr-2015

token	postings
Fast	2
The	1
brown	1
dog	1
fox	1
jumping	2
jumps	1
lazy	1
over	1
quick	1
spiders	2
the	1

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	***The*** quick brown fox jumps over ***the*** lazy dog
2	Fast jumping spiders

Stopwords

- Search for “the” doesn’t bring much value and makes little sense
- Do we really need to index it?

Lital Kozlovski - 28-Apr-2015

token	postings
Fast	2
The	1
brown	1
dog	1
fox	1
jumping	2
jumps	1
lazy	1
over	1
quick	1
spiders	2
the	1

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	***The*** quick brown fox jumps over ***the*** lazy dog
2	Fast jumping spiders

Stopwords

- Search for “the” doesn’t bring much value and makes little sense
- Do we really need to index it?
- Lets drop them

Lital Kozlovski - 28-Apr-2015

token	postings
Fast	2
brown	1
dog	1
fox	1
jumping	2
jumps	1
lazy	1
over	1
quick	1
spiders	2

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The quick brown fox jumps over the lazy dog
2	***Fast*** jumping spiders

Lowercasing

- Don't we want to have case insensitive search?
- Searching for "fast" and "Fast" should yield same results

Lital Kozlov - 28-Apr-2015

token	postings
Fast	2
brown	1
dog	1
fox	1
jumping	2
jumps	1
lazy	1
over	1
quick	1
spiders	2

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The quick brown fox jumps over the lazy dog
2	***Fast*** jumping spiders

Lowercasing

- Don't we want to have case insensitive search?
- Searching for "fast" and "Fast" should yield same results
- Lets lowercase all tokens

Lital Kozlov - 28-Apr-2015

token	postings
brown	1
dog	1
fast	2
fox	1
jumping	2
jumps	1
lazy	1
over	1
quick	1
spiders	2

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The quick brown fox ***jumps*** over the lazy dog
2	Fast ***jumping*** spiders

Grammar

- ***jumps*** and ***jumping*** share the same stem - ***jump***
- Wouldn't it be better if searching for any word stemming from ***jump*** would hit both documents?

Lital Kozlovski - 28-Apr-2015

token	postings
brown	1
dog	1
fast	2
fox	1
jumping	2
jumps	1
lazy	1
over	1
quick	2
spiders	1

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The quick brown fox ***jumps*** over the lazy dog
2	Fast ***jumping*** spiders

Grammar

- ***jumps*** and ***jumping*** share the same stem - ***jump***
- Wouldn't it be better if searching for any word stemming from ***jump*** would hit both documents?
- Lets apply ***stemming*** and only index the stem

token	postings
brown	1
dog	1
fast	2
fox	1
jump	1, 2
lazy	1
over	1
quick	1
spiders	2

Lita Blog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The ***quick*** brown fox jumps over the lazy dog
2	***Fast*** jumping spiders

Synonyms

- Are ***quick*** and ***fast*** interchangeable?
- If we search for ***quick*** do we want to match documents containing ***fast***?

token	postings
brown	1
dog	1
fast	2
fox	1
jump	1, 2
lazy	1
over	1
quick	1
spiders	2

ital Kong - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The ***quick*** brown fox jumps over the lazy dog
2	***Fast*** jumping spiders

Synonyms

- Are ***quick*** and ***fast*** interchangeable?
- If we search for ***quick*** do we want to match documents containing ***fast***?
- If so, lets have the index reflect that

token	postings
brown	1
dog	1
fast	1, 2
fox	1
jump	1, 2
lazy	1
over	1
quick	1, 2
spiders	2

Vital Berg - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The need for text analysis

id	content
1	The quick brown fox jumps over the lazy dog
2	Fast jumping spiders

token	postings
brown	1
dog	1
fast	1, 2
fox	1
jump	1, 2
lazy	1
over	1
quick	1, 2
spiders	2

Process before indexing

- Drop stopwords
- Lowercase all tokens
- Reduce tokens to their stems
- Consider synonyms

Lital Kolog - 28-Apr-2015

But also

- Process the query prior to searching (e.g. turn “Jumped” to “jump”)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Anatomy of an Analyzer

- Tokenizer
  - Breaks the text into tokens and produces a token stream
  - Example: keyword, whitespace, regex, etc...
- Token Filter
  - Acts on the token stream - can drop and modify existing tokens, or add new ones.
  - Example: lowercase, stopword, ngram, etc..
  - Operate in a chain - each filter does its processing and propagates the tokens to the next one in the chain

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

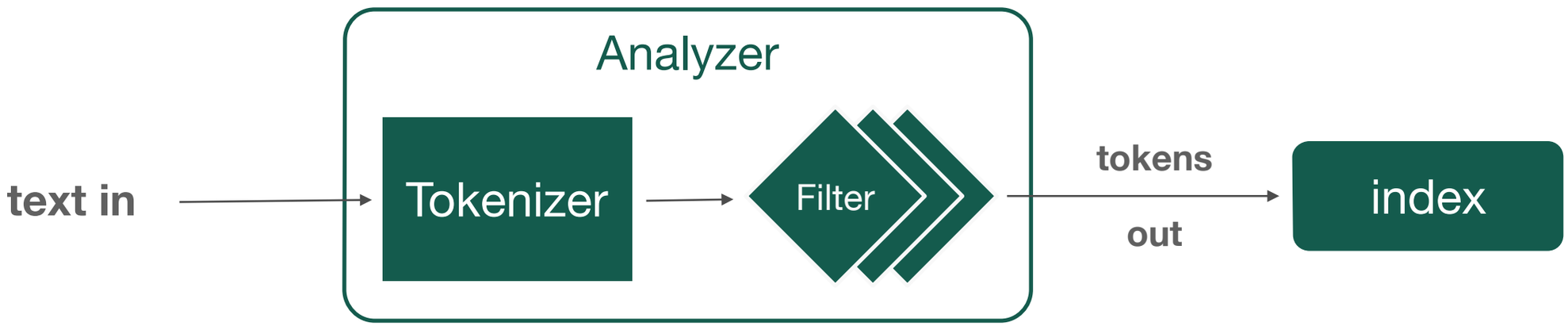
Anatomy of an Analyzer

position increment	The position of the token relative to the prev. one
start/end offset	The offsets of the token in the original text

Lital Kolog - 28-Apr-2015

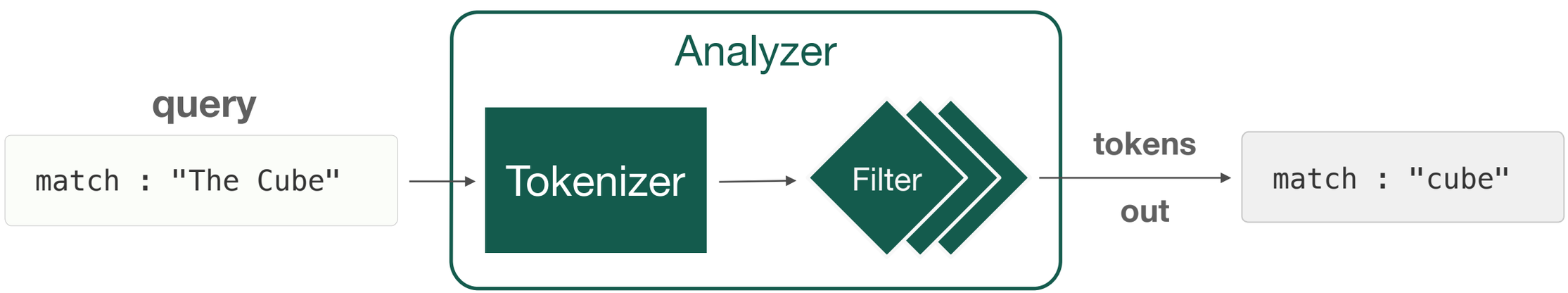
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Anatomy of an analyzer



Lital Kolog - 28-Apr-2015

Search analysis process



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Anatomy of an analyzer

- Build custom analyzers
  - There are many types of tokenizers and token filters to pick & choose from that can serve as the building blocks for custom analyzers
- Or use any of the pre-built analyzers
  - standard
  - simple
  - whitespace
  - stop
  - keyword
  - language
  - etc…

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Analyze API

```
curl -XPOST 'localhost:9200/_analyze?tokenizer=keyword' -d 'New York'
```

```
{
  "tokens" : [
    {
      "token" : "New York",
      "start_offset" : 0,
      "end_offset" : 8,
      "type" : "word",
      "position" : 1
    }
  ]
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Analyze API

```
curl -XPOST 'localhost:9200/_analyze?tokenizer=keyword&filter=lowercase' -d 'New York'
```

```
{
  "tokens" : [
    {
      "token" : "New York",
      "start_offset" : 0,
      "end_offset" : 8,
      "type" : "word",
      "position" : 1
    }
  ]
}
```

Lital Kolog - 28-Apr-2015

```
{
  "tokens" : [
    {
      "token" : "new york",
      "start_offset" : 0,
      "end_offset" : 8,
      "type" : "word",
      "position" : 1
    }
  ]
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Analyze API

```
curl -XPOST 'localhost:9200/_analyze?tokenizer=whitespace' -d '
   The Quick Fox Jumped
'
```

```
{
  "tokens" : [
    {
      "token" : "The",
      "position" : 1
    },
    {
      "token" : "Quick",
      "position" : 2
    },
    {
      "token" : "Fox",
      "position" : 3
    },
    {
      "token" : "Jumped",
      "position" : 4
    }
  ]
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Analyze API

```
curl -XPOST 'localhost:9200/_analyze?tokenizer=whitespace&filters=lowercase' -d '
   The Quick Fox Jumped
'
```

```
{
  "tokens" : [
    {
      "token" : "The",
      "position" : 1
    },
    {
      "token" : "Quick",
      "position" : 2
    },
    {
      "token" : "Fox",
      "position" : 3
    },
    {
      "token" : "Jumped",
      "position" : 4
    }
  ]
}
```

```
{
  "tokens" : [
    {
      "token" : "the",
      "position" : 1
    },
    {
      "token" : "quick",
      "position" : 2
    },
    {
      "token" : "fox",
      "position" : 3
    },
    {
      "token" : "jumped",
      "position" : 4
    }
  ]
}
```

Lital Kolog - 23 Jan 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Analyze API

```
curl -XPOST 'localhost:9200/_analyze?tokenizer=whitespace&filters=lowercase,stop' -d '
   The Quick Fox Jumped
'
```

```
{
  "tokens" : [
    {
      "token" : "The",
      "position" : 1
    },
    {
      "token" : "Quick",
      "position" : 2
    },
    {
      "token" : "Fox",
      "position" : 3
    },
    {
      "token" : "Jumped",
      "position" : 4
    }
  ]
}
```

```
{
  "tokens" : [
    {
      "token" : "the",
      "position" : 1
    },
    {
      "token" : "quick",
      "position" : 2
    },
    {
      "token" : "fox",
      "position" : 3
    },
    {
      "token" : "jumped",
      "position" : 4
    }
  ]
}
```

```
{
  "tokens" : [
    {
      "token" : "quick",
      "position" : 2
    },
    {
      "token" : "fox",
      "position" : 3
    },
    {
      "token" : "jumped",
      "position" : 4
    }
  ]
}
```

Lital Kolog - 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Configuration

- Analysis can be configured per index
  - Custom tokenizers, filters and analyzers

```
curl -XPUT 'localhost:9200/crunchbase' -d '{
  "settings" : {
    "analysis" : {
      "analyzer" : {
        "my_analyzer" : {
          "tokenizer" : "whitespace",
          "filter" : [ "lowercase", "stop", "snowball" ]
        }
      }
    }
  }
}'
```

Order is important

Lital Kolog - 28-Apr-2015

- Global analysis can also be configured on a node level

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Configuration

```
curl -XGET 'localhost:9200/crunchbase/_analyze?analyzer=my_analyzer' -d '
   The quick fox jumped
'
```

```
{
  "tokens" : [
    {
      "token" : "quick",
      "position" : 2
    },
    {
      "token" : "fox",
      "position" : 3
    },
    {
      "token" : "jump",
      "position" : 4
    }
  ]
}
```

Note the impact of the *snowball* filter

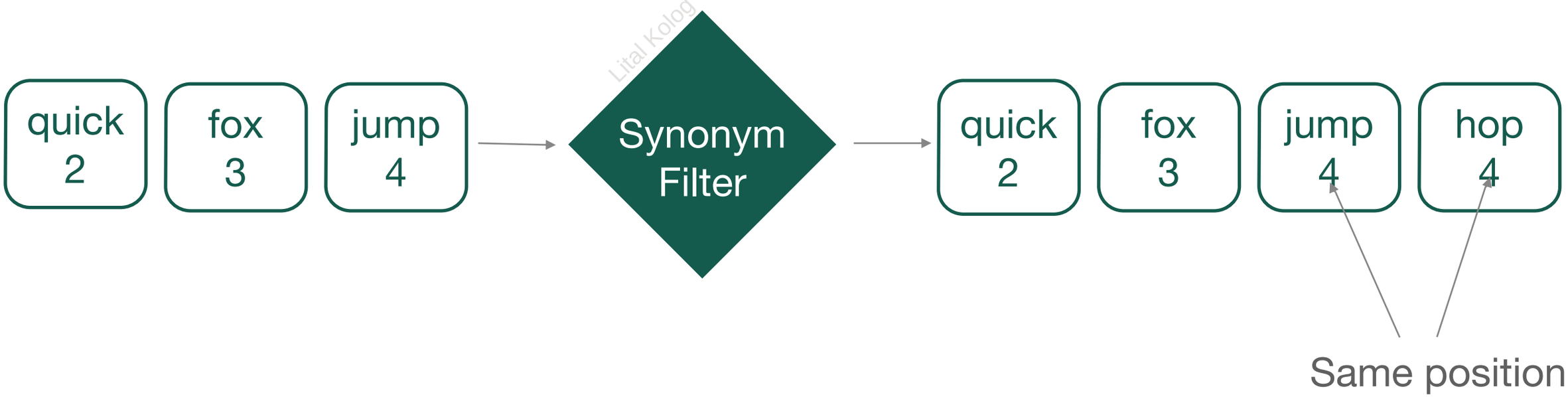
Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Synonyms

- A dedicated token filter for synonym support
  - "Injects" synonyms for tokens into the token stream
  - The injected synonym token will have the same position as its related token
- Say "jump" and "hop" are considered synonyms

Lital Kolog - 28-Apr-2015



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Synonyms

```
curl -XPUT 'localhost:9200/crunchbase' -d '{
  "settings" : {
    "analysis" : {
      "analyzer" : {
        "my_analyzer" : {
          "tokenizer" : "whitespace",
          "filter" : [ "lowercase", "stop", "snowball", "my_domain_synonyms" ]
        }
      },
      "filter" : {
        "my_domain_synonyms" : {
          "type" : "synonym",
          "synonyms" : [
            "jump, hop"
          ]
        }
      }
    }
  }
}'
```

The synonyms definition

Added to the filter chain

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

ICU Tokenizer & Filters

- Available as a plugin
- Provides support for the latest unicode

icu_tokenizer	Extensive multi-lingual support for split character based tokenization (Chinese, Japanese, arabic, and more...)
icu_collation filter	Generates language specific sortable token streams
icu_normalizer filter	‣ Handles composed characters correctly (ä = ä) ‣ Handles ambiguous lowercasing
ICUTransformFilter	Language transformation, such as traditional to simplified Chinese and Serbian Cyrillic to Latin.
icu_folding	Folding characters according to *UTR#30*

Lital Kolog - 23-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Tokenizers

standard	Grammar based tokenization (works well for european languages)
keyword	Creates a single token out of the text (ids, enums)
whitespace	Breaks the text at whitespaces
pattern	Tokenizes based on regex pattern (can be used to break the text based on paragraphs or line breaks)
UAX URL Email	Like the standard tokenizer, but identifies emails and urls and keeps them intact
custom	Create your own tokenizer (Java) and plug it in

Lita Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Token Filters

lowercase	For case insensitive search
nGrams / edgeNGrams	Breaks the tokens into ngrams chunks. Helps with multi-lingual documents and for auto-suggest functionality
shingle	“NGrams for words” - builds combinations of adjacent tokens as single tokens
stemmer / snowball	Algorithmic stemming filters
hunspell	Dictionary based stemming filters
custom	Create your own token filter (Java) and plug it in

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Mappings

Lital Kolog - 28-Apr-2015

elastic

What is mapping?

- The indexed data is based on document and fields
- Mapping defines how these documents should be handled
  - how should the documents be indexed?
  - what are the data types of the document fields?
  - how to treat object typed fields?
  - what are the relations between different types of docs?
  - how to handle document metadata?
  - define boosts per fields / document type

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Dynamic Mapping

- Elasticsearch is "schema-less" by nature
  - "Configure only when you really need to"
- It's not required to define mappings for an index
- When indexing never before "seen" document, a mapping is dynamically created for it
  - Called Dynamic Mapping
- Extracting as much information from the JSON document (e.g. data types) and applying sensible defaults

Lital Kozlov - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

When should you create mapping?

- Defaults are nice, but they're not domain specific
  - Use domain knowledge to fine tune how documents & fields are handled
- Use analyzers to adapt search behavior to user behavior
  - E.g. multi-lingual, domain terminology, etc..
- Certain functionalities simply require it
  - Aggregations
  - Sorting on strings (so the text will not be analyzed)
  - Highlighting

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Configuration

- There are several ways to configure mapping
  - when creating an index
  - by using the dedicated Update Mapping API
  - using index templates
  - node level configuration
  - redefine the default mappings configuration
- It is strongly suggested to prefer the APIs over static configurations

Lital Kolker - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Mapping API

Mapping for a specific document type

```
curl -XPUT 'localhost:9200/crunchbase' -d '{
  "settings" : {
    ...
  },
  "mappings" : {
    "company" : {
      "properties" : {
        "name" : {
          "type"     : "string",
          "analyzer" : "standard"
        }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Get the current mappings of a specific type

```
curl -XGET ‘localhost:9200/crunchbase/_mapping/company'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Mapping API

- Dynamically update the mapping

```
curl -XPUT ‘localhost:9200/crunchbase/_mapping/company' -d '{
  "properties" : {
    "founded_year" : {
      "type" : "integer",
      "boost" : "1.5"
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

- Caution
  - Updating mappings cannot break existing ones (e.g. changing an analyzer is not allowed, as it would not be compatible with the already indexed data)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Basic Mapping

Document type
(root object)

The expected fields
of the document

```
curl -X POST localhost:9200/crunchbase -d '{
  "mappings" : {
    "company" : {
      "properties" : {
        "name"   : {
            "type" : "string",
            "analyzer" : "standard",
            "boost" : 10.0
        },
        "founded_on" : {
            "type" : "date",
            "format" : "YYYY-MM-dd"
        }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Type mapping

- Every document field is of a specific type
  - Each type has a set of attributes that can be configured
  - Field mapping holds for both single and multiple values (arrays)

type	description
string, byte, short, integer, long, float, double, date,	The core data types
root-object / object	Represents a document or an inner-object
nested	An “embedded” document
ip	An IPv4 address
geo_point / geo_shape	Geo types (see *geo location*)

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Text fields

attr. name	description
index	Defines whether this field should be searchable. • *analyzed* - searchable and processed with an analyzer • *not_analyzed* - searchable, but is not processed with an analyzer • *no* - not searchable
analyzer, *index_analyzer,* *search_analyzer*	Defines the analyzer that should be used when analyzing the field at index and/or search time (only applicable when index is set to analyzed)
term_vector	Defines whether term vectors should be stored for the field (and if so, how)

Lisa Kreisig - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Numeric fields

- To index numbers in different precisions
  - float (32bit), double (64bit), integer (32bit), long (64bit), short (16bit) or byte (8bit)
    - Depending on the possible values, always strive to assign the smallest possible numeric type

attr. name	description
ignore_malformed	Whether to ignore malformed values, defaults to *false*
coerce	Whether type casting is supported (*string* to *number*, *double* to *integer*, etc…), defaults to *true*

Ankel Kologi - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Date fields

attr. name	description
format	Defines the expected format of the date field. There's a set of pre-defined named formats that can be used, but custom formats are also supported (the default is *dateOptionalTime*, which is *ISO8601* based format) Also supports multiple formats, such as: `yyyy/MM/dd HH:mm:ss||yyyy/MM/dd`
ignore_malformed	Whether to ignore malformed values (defaults to *false*)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Boolean fields

- The boolean type uses JSON *true* or *false* values.
  - Internally in the index, they are stored as "*T*" and "*F*"

```
{ "is_public" : false }
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Common attributes

attr. name	description
include_in_all	Indicates whether the value of the field should be included in the *_all* field
null_value	Defines a default value for the field
store	Whether to store the field's original value (apart from indexing it)
index_name	Defines the name by which the field is indexed

Digital Blog - 28 August 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Object fields

```
curl -XPUT localhost:9200/crunchbase/company/_mapping -d '{
  "company" : {
    "properties" : {
      "name" : { "type" : "string", "analyzer" : "snowball" },
      "ceo" : {
        "type" : "object",
        "properties" : {
          "since" : { "type" : "date" },
          "name" : {
            "properties" : {
              "first" : { "type" : "string", "analyzer" : "standard" },
              "last" : { "type" : "string", "analyzer" : "snowball" }
            }
          }
        }
      }
    }
  }
}'
```

Two levels of object fields

Alex Kolog 08-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Object fields

- Referencing an inner-object field is done by its full path

```
ceo.name.first
```

  - All search elements can refer to inner-object fields by their full path
  - It is also possible to include the document type as part of the path. This will provide added value in the context of its usage, for example:
    - Using `company.ceo.name.first` in a query will cause it to be expanded with a filter on the company type

ital Kafka, 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Back to dynamic types

- By default, mappings have a dynamic nature
  - New fields/objects introduced will automatically be indexed and the mapping will be updated
- It is possible to restrict this dynamic behavior
  - setting *dynamic* to *false* - causes unmapped data to be ignored
  - setting *dynamic* to *strict* - causes unmapped data to return an error

Lital Kolos 28-Apr-2015

```
"company" : {
  "dynamic" : false,
  "properties" : {
    ...
  }
}
```

```
"company" : {
  "dynamic" : “strict”,
  "properties" : {
    ...
  }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multi-fields

- Enables indexing a single document field in multiple ways
  - A typical use case for that is the need to have a string field analyzed in multiple ways
  - Handy for supporting aggregations or sorting along with queries

Use for search

Use for aggs/sorting

```
"company" : {
  "properties" : {
    "title" : {
      "type" : “string”,
      "analyzer" : “standard”,
      "fields" : {
        "raw" : { "type" : "string", "index" : "not_analyzed" }
      }
    }
  }
}
```

Lital Kalug - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multi-fields

- Multi field access
  - Any additional "sub"-fields that are defined under the fields object can be access using the dot-notation, where the main field serves as a namespace
    - to access the main field, use the field name `title`
    - to access the sub field, use the dot-notation `title.raw`
- Sub-fields can be dynamically added using put mappings

Lital Kotler 28-Apr-2015

```
curl -XPUT localhost:9200/crunchbase/company/_mapping '{
   "title" : {
      "properties" : {
         "title" : {
            "type" : "string",
            "fields" : {
               "sort" : { "type" : "string", "analyzer" : "alphasort" }
            }
      ...
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metadata fields

attr. name	description
_id	The id of the document
_type	The type of the document
_source	Stores the original document that was indexed
_all	Indexes all values of all document fields
_timestamp	Timestamp associated with the document
_ttl	Optionally defines an expiration time
_size	Indexes the size of the uncompressed *_source* content
_analyzer	Custom analyzer for the document
_routing	Assign a routing value per document

Vital Kolodyazhnyy 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metadata fields

- *_type* is indexed by default, can be disabled

```
"_type" : { "index" : "no" }
```

- *_all* is an indexed "catch all" field
  - Can hold values of other fields, thus enables simple queries across multiples fields
    - Has its own analyzer
    - Can also be disabled

Lital Kolog - 28-Apr-2015

```
"_all" : { "enabled" : false }
```

    - When disabled, don't forget to reset the *index.query.default_field* which is used in queries that are not associated with a specific field (e.g. *query_string*)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metadata fields

- *_analyzer* enables setting a default analyzer per document
  - Specifies a field in the document from which the analyzer name will be extracted (defaults to "*_analyzer*")
  - When specified, the analyzer will be used for all fields that don't already have an analyzer explicitly configured

Define the path to the field in the document which specifies the analyzer to be used

```
"_analyzer" : {
  "path" : "meta._analyzer"
}
```

Lital Kolog - 5-Mar-2015

then, when indexing:

```
{
  "name" : "Tony Stark",
  "meta" : { "_analyzer" : "english" }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metadata fields

- *_timestamp* enables associating timestamps with documents
  - It is disabled by default. When enabled it will automatically be generated if not explicitly provided
  - Can be specified either as a request parameter or as a field within the document itself
    - The mapping allows to specify the field in the document from which it will be extracted

Lital Kalog - 28-10-2015

```
"mappings" : {
  "company" : {
    "_timestamp" : {
      "enabled" : true,
      "path" : "created_at"
    }
  }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metadata fields

- *_ttl* enables assigning an expiration time to the document
  - When expired, a document is deleted from the index
  - It is disabled by default, but can be enabled in the mapping

```
"mappings" : {
  "company" : {
    "_ttl" : {
      "enabled" : true,
      "default" : "1d"
    }
  }
}
```

Lital Kolog - 28-Apr-2015

A default value can also be defined for documents that don't explicitly specify *_ttl*

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Custom metadata

- It is possible to associate meta data with a document
  - This is a custom meta-data that will be saved in the mappings. Can be used for example to:
    - Store deserialization information for automatic unmarshalling of a document to its appropriate object type in a client library
    - Assign versions to the mappings

Lital Kolog 28-Apr-2015

```
"mappings" : {
  "company" : {
    "_meta" : {
      "version" : “1.0.1”,
      "class_name" : "MyApplication::Models::Company"
    }
  }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Default mappings

- It is possible to redefine the default mappings in an index
  - Will be applied if a field wasn't explicitly mapped

```
curl -XPUT 'localhost:9200/test' -d '{
  "settings" : {
    "number_of_shards" : 3,
    "number_of_replicas" : 0
  },
  "mappings" : {
    "_default_" : {
      "_all" : { "enabled" : false }
    },
    "type1" : {
      ...
    }
  }
}'
```

Disable the *_all* field for all mapping types in this index

Lital Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Dynamic field mapping

- It is also possible to define dynamic mappings rules for field
  - Will match unmapped fields based on name/type

```
curl -XPUT 'localhost:9200/test' -d '{
  "mappings" : {
    "type1" : {
      "dynamic_templates" : [
         {
           "integers" : {
              "match" : "i_*",
              "mapping" : { "type" : "integer" }
           }
         },
         {
            "multi_strings" : {
               "match_mapping_type" : "string",
               "mapping" : {
                  "type" : "string", "index" : "analyzed",
                  "fields" : {
                     "orig" : { "type" : "string", "index" : "not_analyzed"}
                  }
               }
            }
         }
         ...
```

Every unmapped field starting with *i_** will be mapped as integer by default

Every unmapped *string* field will be mapped as multi-field by default

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

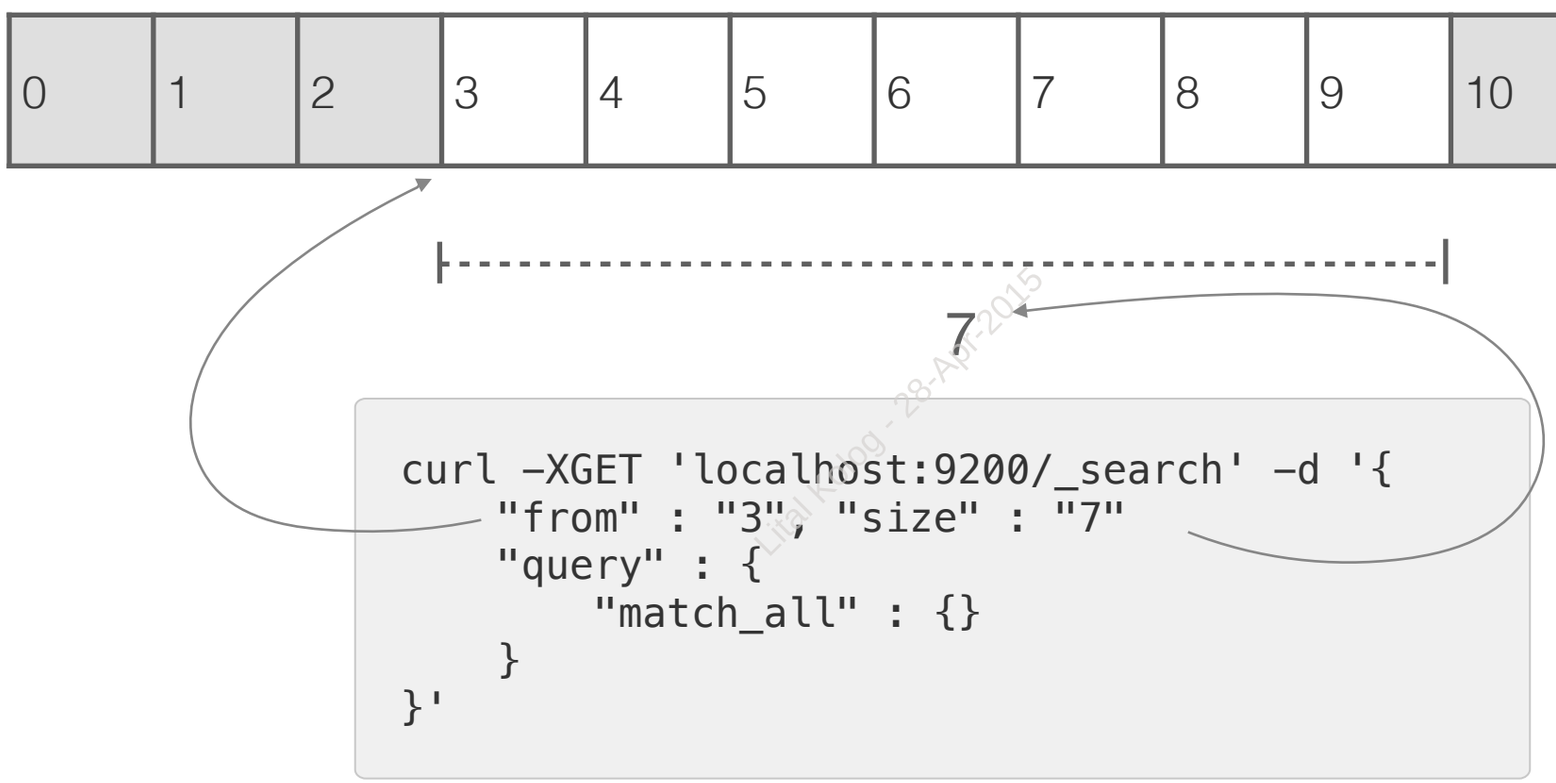
_search

Lital Kolog - 28-Apr-2015

elastic

Pagination

- A search request is always within a "page" context
  - A window into the list of documents matching the query



  - The order of the list is determined by the sorting context

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Sorting

- Determines the order by which the hits will be returned
  - Relevancy - sorts by best/worse matching document
  - Field - sorts by field value
- Use multi-level sorting for tie breaking

```
curl -XPOST 'localhost:9200/_search' -d '{
  "sort" : [
    { "intelligence" : { "order" : "desc" } },
    { "strength" : { "order" : "asc" } },
    "_score"
  ],
  "query" : {
    "match_all" : {}
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Sorting - fielddata

- When sorting by field, all values of the field in all documents will be loaded into memory
  - Required for fast sorting
- Fielddata
  - A segment level data structure to hold field values for documents
  - Elasticsearch automatically chooses the most optimal structure based on the nature of the field
    - data type
    - single valued vs. multi-valued
    - number of unique values
    - maximum value (for numeric types)

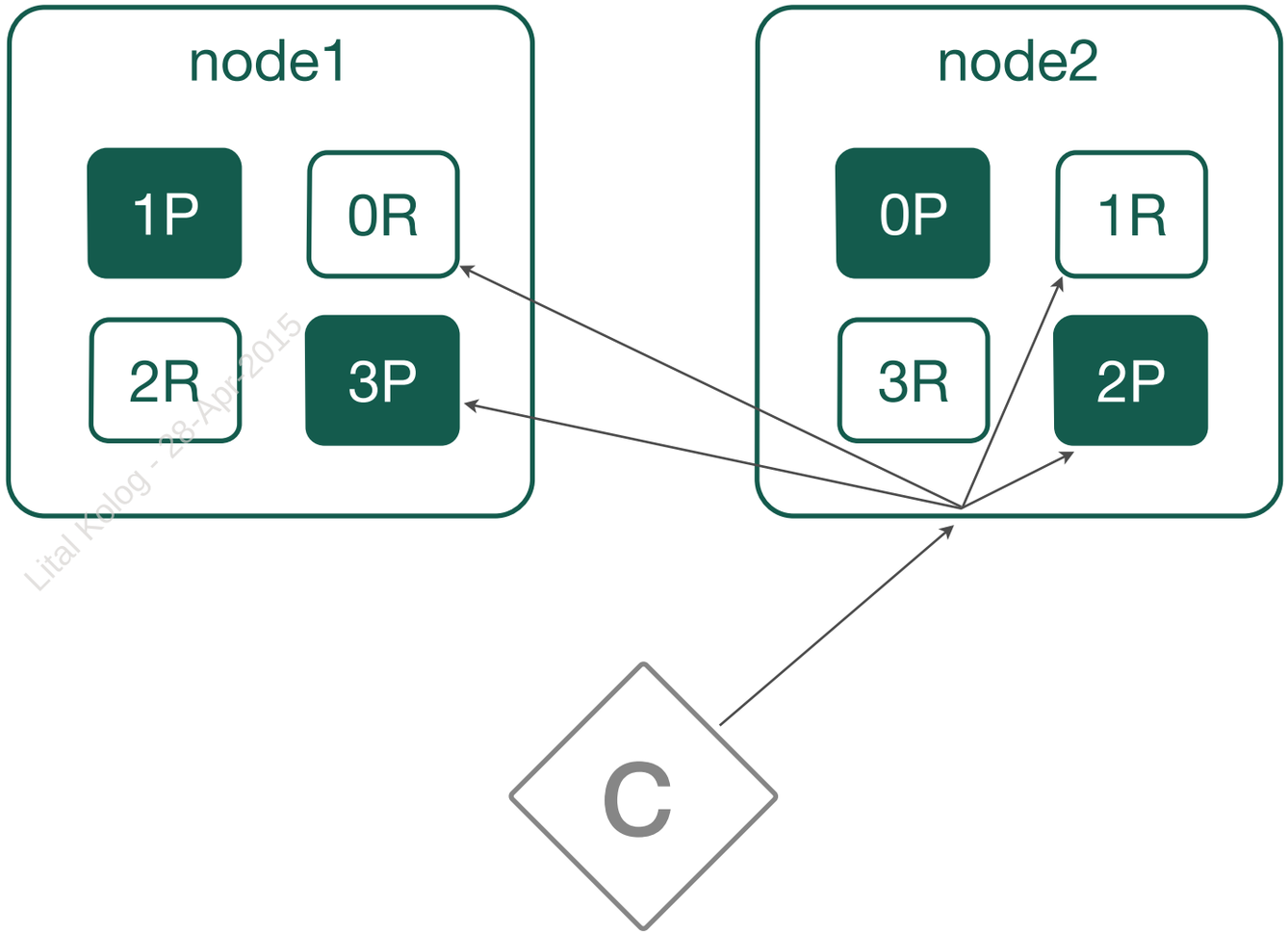
Lital Kozlov - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- First phase is *QUERY*, executed on all relevant shards of the search request
- *QUERY* phase includes execution of the actual query on each shard

  Builds a sorted priority queue on each shard (length is “from” + “size”) and fills it up
- Also includes aggregations execution if required



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

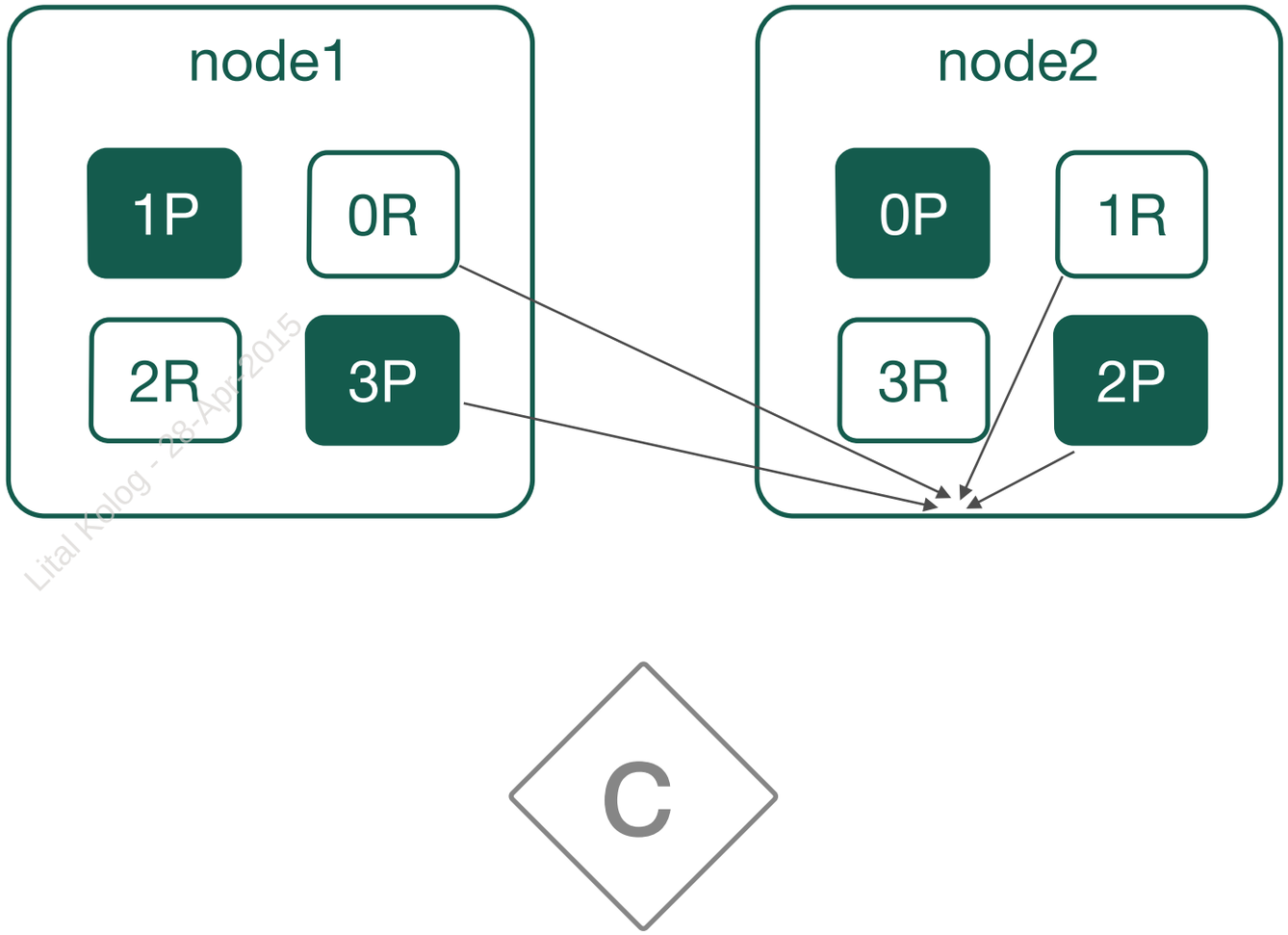
Search Types

- Results of the query execution are sent back to the originating node

  Include doc ids (Lucene numeric ones) and all sorting values per doc (by default, score)

  Also includes faceting results

- Results are reduced and resorted / aggregated



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

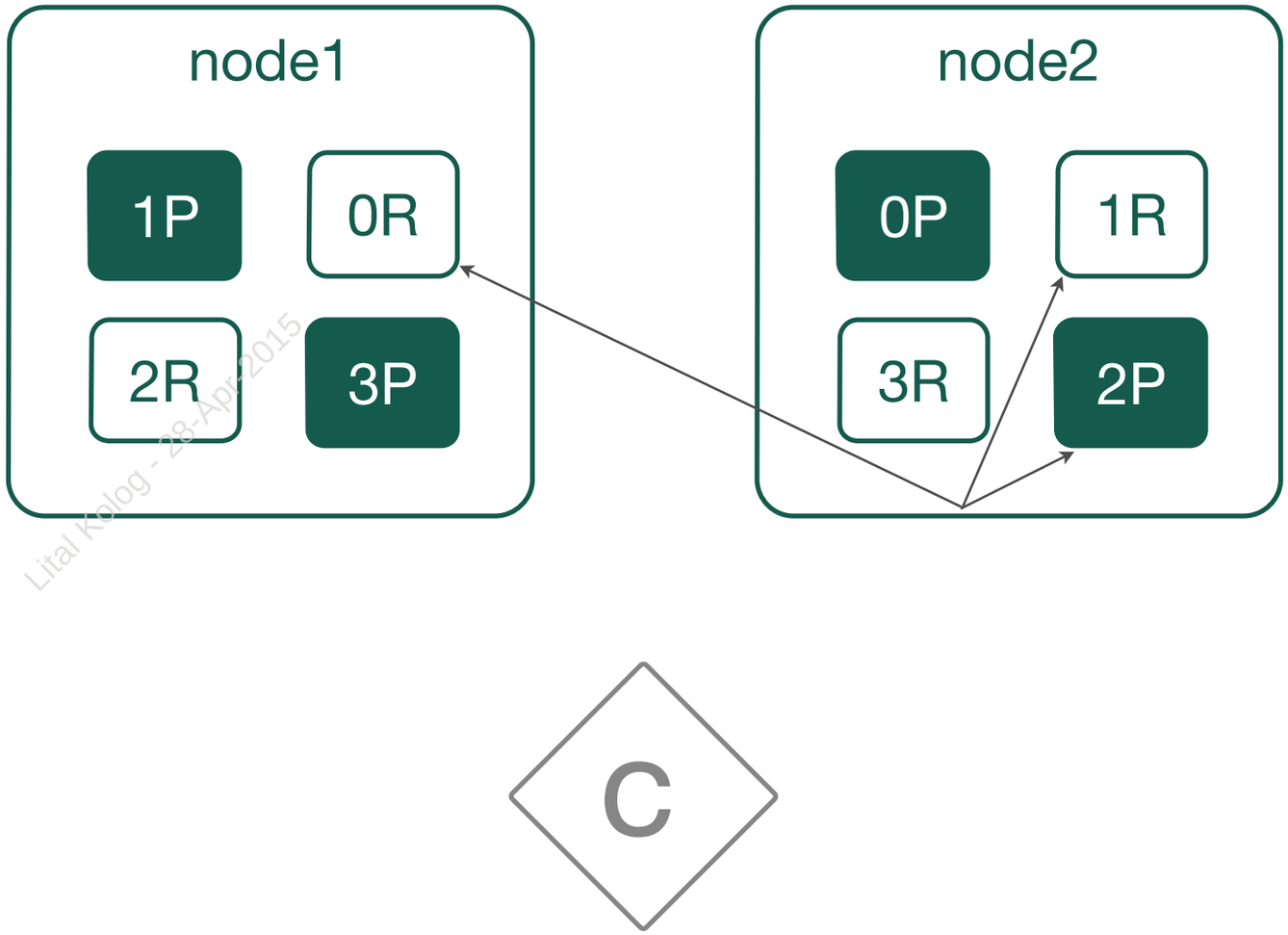
Search Types

- *FETCH* phase is executed, choosing the shards where the top size documents need to be fetched from

  (in our case, shard 3 held no relevant hits)

- *FETCH* includes fetching the actual *_id*, *_type*, and *_source*, including any custom fields

- Also includes highlighting, explain, and any “hit” level information

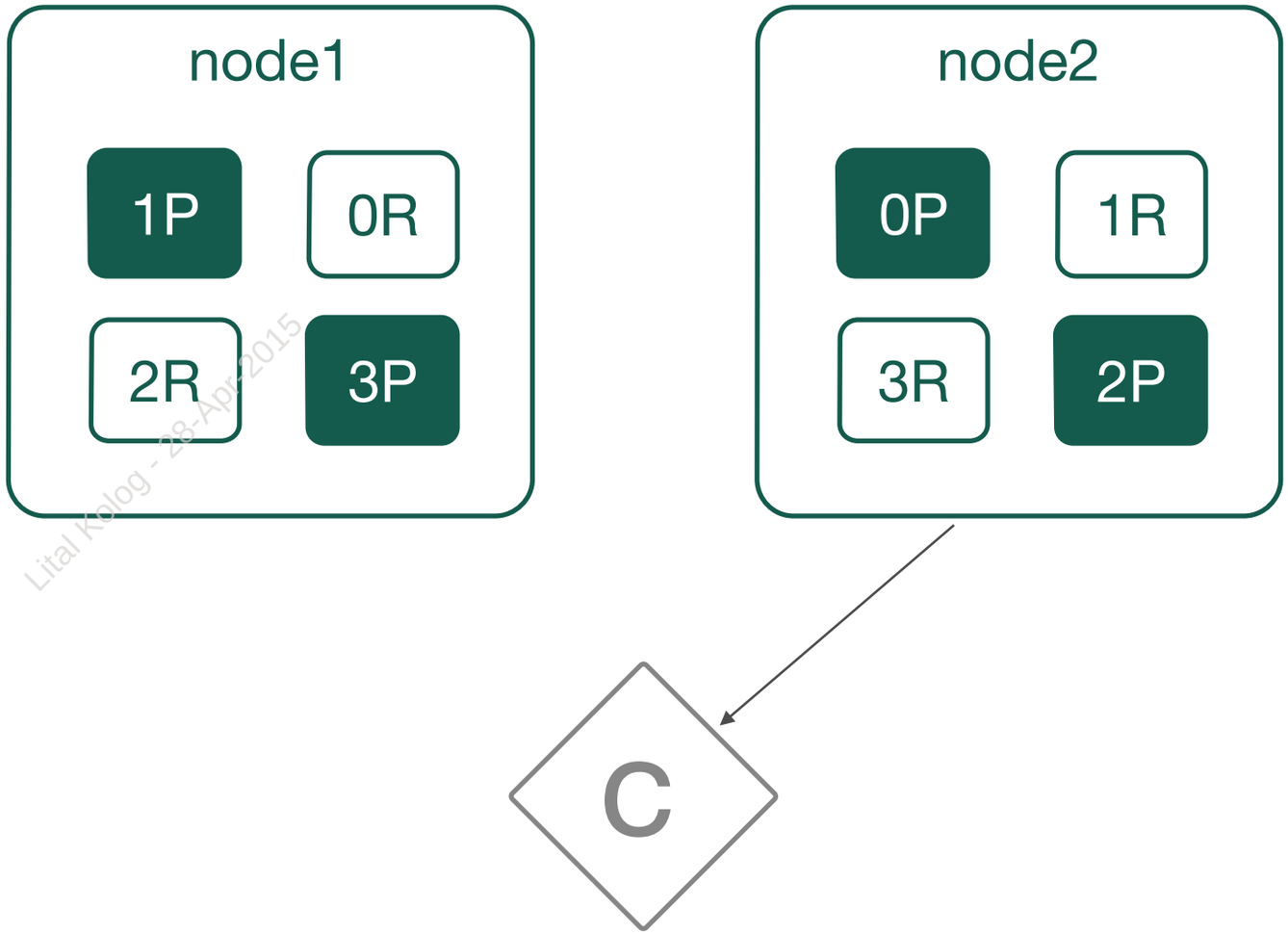


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- Hits are enriched and returned to the user



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- QUERY_THEN_FETCH
  - If the request resolves to be executed on a single shard, *QUERY* and *FETCH* are executed in a single roundtrip automatically
  - Excellent for deep pagination, since only numeric document ids and sorting values for "from + size" hits are sent after the *QUERY* phase

Lital Kolog 28 April 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- DFS_QUERY_THEN_FETCH
  - By default, frequencies are computed on the shard level which can cause unexpected scoring behaviour
    - Note, this tends to even out with "enough data" in each shard
- A pre-QUERY phase that is executed on the relevant shards
  - Sending back the relevant statistics (document frequencies) for the query which are then aggregated, and sent to each shard on the *QUERY* phase
    - The query itself is executed using the aggregated statistics

Vital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- COUNT
  - Returns no hits, only total count matching the query, thus executes in a single round trip to the shards
    - No need to maintain any internal priority queue for sorting documents
    - No need for a FETCH phase
- Great for
    - Quick counts
    - When only *aggregation* results are needed

Lital Kolog - 28-Apr-2018

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- SCAN
  - Allows to iterate over large amounts of data, giving up on sorting
    - With normal search, sorting gets more expensive the deeper you paginate
    - With scan there is no sorting, thus no need to maintain expensive consistent distributed sorting behaviour
- Highly optimized on the Lucene level
  - A segment that has been scanned already is skipped in the next search execution

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- SCAN

```
curl -XGET 'localhost:9200/_search?search_type=scan&scroll=10m&size=50' -d '{
    "query" : {
        "match_all" : {}
    }
}'
```

  - pick as small number as possible
  - *size* controls the number of documents **per shard**
  - returns a *scroll_id* that can be used to initiate the scrolling process

Lital Kabas - 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Search Types

- SCAN - scrolling
  - Subsequent requests use the *scroll_id* returned from the previous response

```
curl -XGET 'localhost:9200/_search/scroll?scroll=10m' -d
'c2NhbjsxOjBLMzdpWEtqU2IyZHlmVURPeFJOZnc7MzowSzM3aVhLalNiMmR5ZlVET3hSTmZ3OzU6MEszN2lYS
2pTYjJkeWZVRE94Uk5mdzsyOjBLMzdpWEtqU2IyZHlmVURPeFJOZnc7NDowSzM3aVhLalNiMmR5ZlVET3hSTmZ
3Ow==='
```

Lital Vashg - 28-Apr-2015

- When finished scanning, explicitly clear the scroll
  - Also useful for aborting the scan

```
curl -XDELETE 'localhost:9200/_search/scroll/c2NhbjsxOjBLMzdpWEtqU2IyZHlmVURPeFJ
OZnc7MzowSzM3aVhLalNiMmR5ZlVET3hSTmZ3OzU6MEszN2lYS2pTYjJkeWZVRE94Uk5mdzsyOjBLMzdpWEtqU
2IyZHlmVURPeFJOZnc7NDowSzM3aVhLalNiMmR5ZlVET3hSTmZ3Ow==='
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL

- The query DSL (domain specific language) allows to express complex definitions of how to slice & dice data
- Two main types:
  - Queries
    - Typically contribute to scoring a document
    - No caching. Fast!, thanks to the inverted index structure
  - Filters
    - No scoring
    - Typically cacheable, contributing to its execution speed

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL

- Represented as an AST (abstract syntax tree)
- Queries
  - Core - executed directly against one or more fields ("leaf nodes")
  - Compound - composed of one or more queries
- Filters
  - Core - execute directly against one or more field ("leaf nodes")
  - Compound - composed of one or more filters

Lital Kozlovski - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL

- Queries and Filters are standalone elements
  - Can be used by across many available APIs in potentially multiple places within an API request
    - Example: a filter can be placed within a *filtered* query, as well as within a *filter* aggregation

Typically described as:

```
# term filter
{
  "term" : { "color" : “blue" }
}
```

And used as:

```
GET /_search '{
  "query" : {
    "filtered" : {
      "query" : { "match_all" : {} },
      "filter" : {
        "term" : { "color” : "blue" }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: What's in a Logic?

```
"if it was so, it might be; and if it were so, it would be; but as it isn't, it ain't.
That' s logic"
```

- Elasticsearch has a special boolean logic
  - A super set of the typical logic we all know that is based on unary operators (*must* & *should*) rather than binary operators (*and* & *or*)
- For a document to match a boolean query:

Yael Kolog - 28-Apr-2015

must(Q)	document must match query *Q*
must_not(Q)	document must not match query *Q*
should(Q)	document doesn't have to match *Q*, but if it matches, it will get a higher score

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: What's in a Logic?

- A boolean expression can have any number of *must*, *should*, and *must_not* clauses
- Note, if the expression is only made of *should* clauses, at least one of these clauses must match
  - Otherwise, all documents would always match the query
- Elasticsearch's boolean logic is a super set over the typical boolean logic

Lital Kolog - 28-April-2015

`Q1 AND Q2`	=	`must(Q1), must(Q2)`	=	`+Q1 +Q2`
`Q1 OR Q2`	=	`should(Q1), should(Q2)`	=	`Q1 Q2`
`NOT Q1`	=	`must_not(Q1)`	=	`-Q1`

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: match

- A one stop shop for matching on a field
- Does the right thing based on the field type and mapping
  - Works well for numeric fields, not_analyzed fields, and analysis chain associated with a string field

```
{
  "match" : { "title" : "Fox" }
}
```

title is analyzed, so the input text will be analyzed as well

Lital Kolog - 28-Apr-2015

```
{
  "match" : { "age" : 25 }
}
```

age is numeric, will build the proper numeric query

```
{
  "match" : { "color" : "blue" }
}
```

color is not_analyzed, will not analyze the incoming text

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: match

- When executed on string type
  - The *analyzer* setting enables specifying a different analyzer that will be used to analyze the text in the query (in place of the mapped search analyzer)

```
{
  "match" : {
    "title" : {
      "query" : "quick fox",
      "analyzer" : "whitespace"
    }
  }
}
```

Lital Kolog - 28-Apr-2015

Note, the structure changed to allow for additional parameters to be provided

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: match

- When executed on string type
  - With *type: boolean*, the query text is analyzed and all produced tokens are added to a boolean query

```
{
  "match" : {
    "title" : {
      "type” : "boolean",
      "query” : "quick fox jumped",
      "minimum_should_match" : "30%"
    }
  }
}
```

Lital Kolog - 28-Apr-2020

attr. name	default	description
operator	*OR*	Add as a should clause (OR), or as a must clause (AND)
minimum_should_match	*1*	How many should clauses are required for a match

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: match

- When executed on string type
  - With type: phrase, a phrase query is constructed out of the analyzed terms, with an optional slop

```
{
  "match" : {
    "title" : {
      "type” : "phrase",
      "query” : "quick fox",
      "slop" : 1
    }
  }
}
```

Lital Kolog - 28-Apr-2015

attr. name	default	description
slop	*0*	The number of other terms allowed between query terms in a document

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: match

- When executed on string type
  - The *type: phrase_prefix* is the same as *type: phrase* except the last analyzed term will be matched as a prefix

```
{
  "match" : {
    "title" : {
      "type” : “phrase_prefix",
      "query” : "quick fo",
      “max_expansion" : 100
    }
  }
}
```

Lital Kolog - 28-Apr-2015

attr. name	default	description
slop	*0*	The number of other terms allowed between query terms in a document
max_expansion	*unbounded*	The maximum number of terms the last prefix will expand to

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: multi_match

- Same as *match* query, with all the mentioned options, but executed on multiple fields
  - Same as using *bool*/*dis_max* with match queries, only simplified syntax

```
{
  "multi_match” : {
    "fields" : [ "title", "body" ],
    "query" : "quick fox"
  }
}
```

Executed on two fields, wrapping each field query as a should clause in a bool query

Lital Kolog - 28-Apr-2015

```
{
  "multi_match” : {
    "fields" : [ “title^10", "body" ],
    "query" : "quick fox"
  }
}
```

Boost any subject hits by 10

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: multi_match

```
{
  "multi_match” : {
    "fields" : [ "title", “name.*” ],
    "query" : "quick fox"
  }
}
```

Automatically expand *name.** to all fields that exists within the name object structure

```
{
  "multi_match” : {
    "fields" : [ "title", "body" ],
    "query" : "quick fox",
    "type" : “best_fields"
  }
}
```

Will score documents based on the fields that match best, can be controlled by *tie_breaker* (default)

Lital Kolog - 28-Apr-2015

```
{
  "multi_match” : {
    "fields" : [ "title", "body" ],
    "query" : "quick fox",
    "type" : "most_fields"
  }
}
```

Will score documents by summing the scores of the matched queries

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: bool

- A compound query, composing other queries in a boolean logic
- The three unary operators are supported
  - *must*, *must_not*, *should*

```
{
    "bool" : {
        "must" : [
            { "match" : { "color" : "blue" } },
            { "match" : { "title" : "shirt" } }
        ],
        "must_not" : [
            { "match" : { "size" : "xxl" } }
        ],
        "should" : [
            { "match" : { "textile" : "cotton" } }
        ]
    }
}
```

Lita Kong - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: bool

attr. name	default	description
minimum_should_match	*0/1*	The minimum *should* clauses that are required to match the document - *0* if there is a *must* or a *must_not* clause - *1* if there are only *should* clauses Support percent format, i.e. *20%*

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: range

- Matches document field that is within a specified range
  - Numeric/date types
  - String uses lexicographical range
- Usually (specifically on numeric/date types), a range filter should be used
  - Caching can be disabled on the range filter level, so filter applies also for cases where the range matching should not be cached

Vital Kolos - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: range

- Ranges are notated with
  - gt / gte (greater, greater or equal)
  - lt / lte (less, less or equal)

```
{
  "range" : {
    "founded_year" : {
      "gte" : 1990,
      "lt" : 2000
    }
  }
}
```

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: match_all

- A query that matches all documents
  - Used when “counting” or in combination with filters

```
{
  “match_all" : {}
}
```

Fast execution of matching on all docs, with score matching the query boost

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: query_string

- Specialized query syntax support as input
  - Based on the Lucene query syntax
- Performs query time analysis of text
  - But not completely, since it needs to parse the text for query operators
- Query syntax is very strict

Lital Kolog - 28-Apr-2015

```
{
  "query_string" : {
    "query" : "(tony OR thor) AND intelligence:[6 TO *]"
  }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: query_string

attr. name	default	description
query	*NONE*	The query text to parse
default_field	*_all*	The field to use when no field is specified
fields	*NONE*	Allows to specify multiple fields to automatically use for non named field syntax, support wildcarding on field names (similar to *multi_match*)
default_operator	*OR*	The operator to use when combining analyzed terms
analyze_wildcard	*FALSE*	Best effort to analyze wildcard syntax text
lenient	*FALSE*	Ignore format based failures

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: simple_query_string

- A simplified query syntax support as input
  - Based on the Lucene simple query syntax
- A lenient query syntax
  - Will gracefully handle syntax errors and will try to do the "right" thing
- Somewhat restricted query power
  - E.g. no ranges, only trailing wildcards, no field syntax

Lital Kuras - 28-Apr-2015

```
{
  "simple_query_string" : {
    "query" : "(tony | ironman) + thor"
  }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: simple_query_string

attr. name	default	description
query	*NONE*	The query text to parse
field	*_all*	The field to use when no field is specified
fields	*NONE*	Allows to specify multiple fields to automatically use for non named field syntax, support wildcarding on field names (similar to *multi_match*)
default_operator	*OR*	The operator to use when combining analyzed terms
flags	*ALL*	Controls the features of the query syntax. Available flags: *ALL*, *NONE*, *AND*, *OR*, *PREFIX*, *PHRASE*, *PRECEDENCE*, *ESCAPE*, and *WHITESPACE*.

Multiple flags can be separated by `|`

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Other Queries

- *term*: matches on the field content **without** analyzing the provided text
  - Should generally not be used, *match* "does the right thing" when it comes to analysis
- *prefix*/*wildcard*: matching on partial term, does not analyze provided text
- *fuzzy*: matches similar looking text based on edit distance
  - Based on Levenshtein algorithm
  - Text not analyzed, use *match* with *fuzziness*
- *regexp* - matches on regular expression

Vital Kolos 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL: Filters

- Many types of filters
  - *term*, *bool*, *and*, *or*, *not*, *range*, *geo_distance*, ...
- Filters can be used:
  - In a *filtered* query
  - In a *constant_score* query (turns a filter to a query)
  - As a post filter element in a search request (used along with aggregations)
  - As a filter aggregation in search request
  - ...

Lital Kobo - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL: Filters

```
{
  "term" : { "color" : "blue" }
}
```

Filters all docs with color blue, note, not analyzed

```
{
  "term" : { "age" : 25 }
}
```

Automatically works for core types, like numerics and dates

```
{
  "range" : { "age" : { "gte" : 25, "lt" : 30 }}
}
```

Automatically works for core types, like numerics and dates

```
{
  "terms" : { "color" : ["blue", “red"] }
}
```

Filters all docs with color blue, note, not analyzed

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL: Filters

```
{
  "and" : [
    { "term" : { "age" : 25 }},
    { "term" : { "color" : "blue" }}
  ]
}
```

Compound filter, "and-ing" all the inner filters

```
{
  "or" : [
    { "term" : { "age" : 25 }},
    { "term" : { "color" : "blue" }}
  ]
}
```

Compound filter, “or-ing” all the inner filters

Lital Kolog - 28-Apr-2015

```
{
  "not" : { "term" : { "age" : 25 }}
}
```

Negate the result of the inner filter

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query DSL: Filters

- Mostly prefer *bool* filter over *and*/*or*/*not*
  - specifically when composing core filters (BitSet based)
- Prefer *and*/*or*/*not* when composing non BitSet filters
  - e.g. geo_*, scripts, etc...
- Combination at times helps (e.g. bool within and)

Lital Kolog - 28-Apr-2015

```
{
  "bool" : {
    "must" : [
      { "term" : { "color" : "blue" }}
    ],
    "should" : [
      { "term" : { "age" : 25 }},
      { "term" : { "priority" : 3 }}
    ]
  }
}
```

Compound Filter

- *must* clauses are required (and)
- *should* clauses are optional (or)
- *must_not* clauses are required not to match (not)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: BitSets, Filters, and Lucene

- BitSet is:
  - Array data structure that compactly stores bits (on/off)
- Since:
  - Search is per Lucene segment and internal doc ids are sequential integer values on the segment level
- Then
  - Per segment, we can easily use a bitset to represent matching documents, with doc ids as the index into the bitset, and the bit representing match (1), or no match (0)

Lital Kolker - 28-Apr-2015

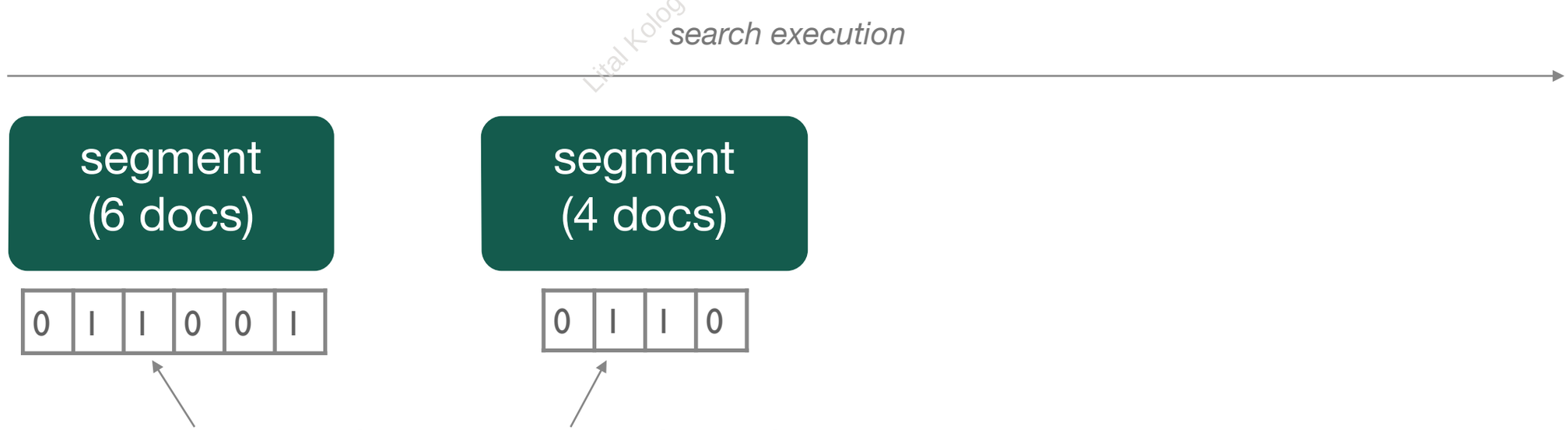
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: BitSets, Filters, and Lucene

```
{
  "filtered" : {
    "query" : {
      "match" : { "text" : "quick fox" }
    },
    "filter" : {
      { "term" : { "age" : 25 }}
    }
  }
}
```

Execute a free text query including a specific age filter

Lital Kolog - 28-Apr-2015



BitSets per segment for *"age" : 25*, filled from the index

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

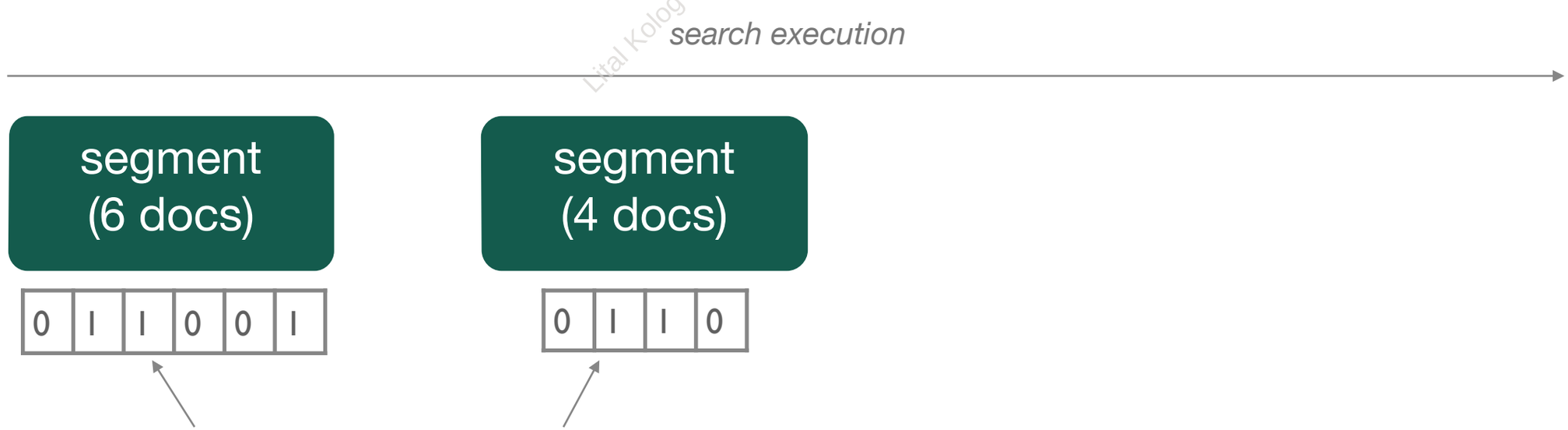
Detour: BitSets, Filters, and Lucene

```
{
  "filtered" : {
    "query" : {
      "match" : { "text" : "slow bear" }
    },
    "filter" : {
      { "term" : { "age" : 25 }}
    }
  }
}
```

Execute a different free text query including the same age filter

Oh…, that filter BitSet didn't really change from the previous execution

Lital Kolog - 28-Apr-2015



BitSets per segment for *"age" : 25*, filled from the index

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

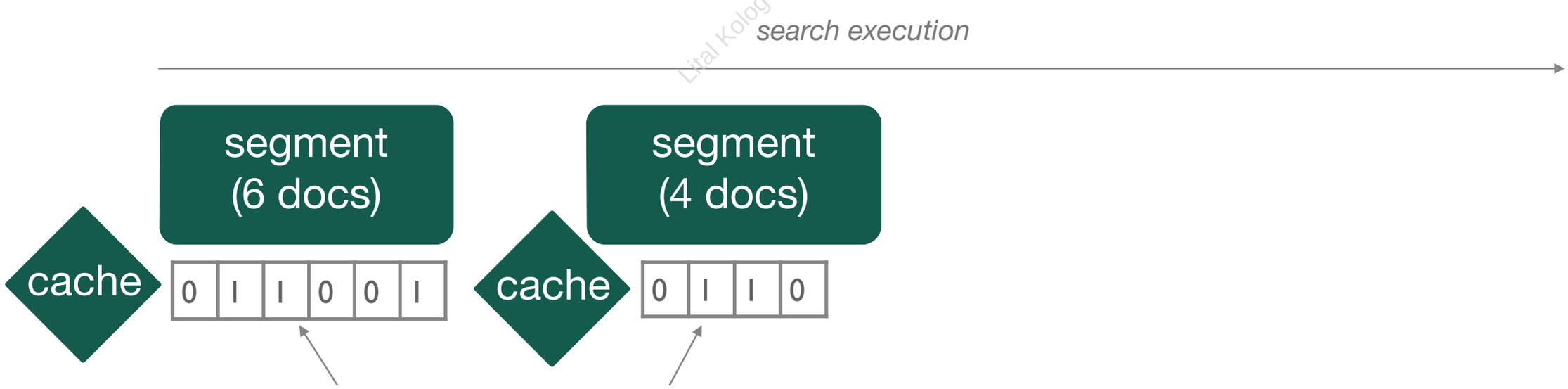
Detour: BitSets, Filters, and Lucene

```
{
  "filtered" : {
    "query" : {
      "match" : { "text" : "quick fox" }
    },
    "filter" : {
      { "term" : { "age" : 25 }}
    }
  }
}
```

Let's try again…

But now, lets cache the *"age" : 25* term filter

Lital Kolog - 28-Apr-2015



BitSets per segment for *"age" : 25*, filled from the index, **and cached**

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

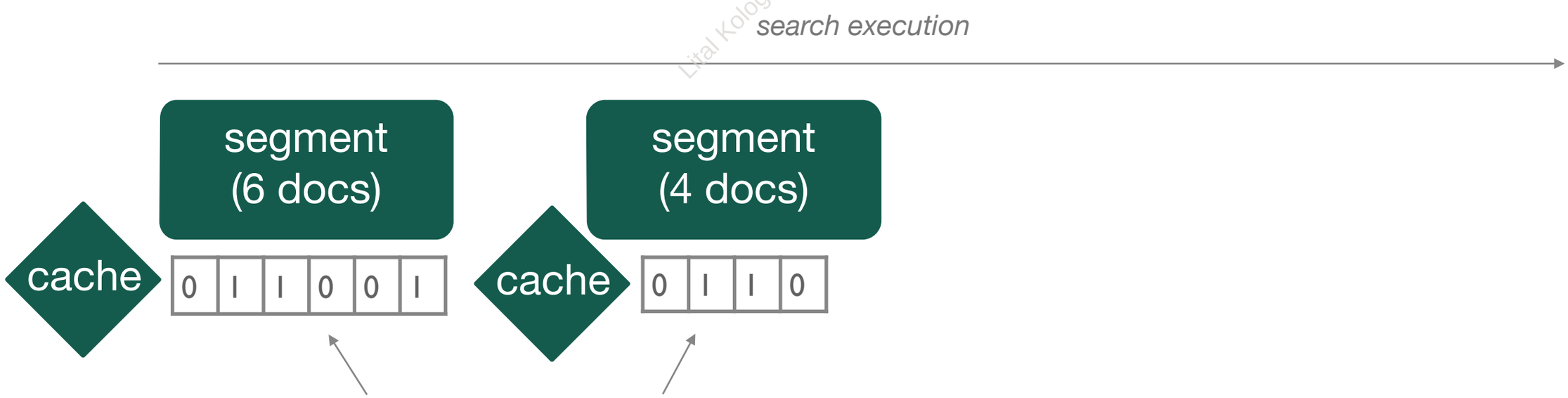
Detour: BitSets, Filters, and Lucene

```
{
  "filtered" : {
    "query" : {
      "match" : { "text" : "slow bear" }
    },
    "filter" : {
      { "term" : { "age" : 25 }}
    }
  }
}
```

Let's try again…

Query changed, but the filter did not, so we can load it from the cache without needing to re-execute it

Lital Kolog - 28-Apr-2015



BitSets per segment for *"age" : 25*, fetched from the cache

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: BitSets, Filters, and Lucene

- The beauty of filter caching
  - A filter can be cached on its "own", regardless of where it exists within a complex query, or a search request, if it's cached, it will be used
  - Memory footprint per filter entry cache is very low
    - Cache entry is: "term filter on field age with value 25"
  - Most core filters implementations already use a BitSet as their internal representation, caching them is cheap
    - Those include *term*, *terms*, *range*, ...
  - Filter that does not match anything on a segment can shortcut any future search execution on that segment

Vital Kolle - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Filter Cache

- A node level filter cache used by all shards
  - The *indices.cache.filter.size* controls the size (memory) of the filter cache
  - Accepts either a percent value, e.g. *30%*, which is computed out of the total heap (*ES_HEAP_SIZE*), or an exact size value, e.g. *2gb*
    - Defaults to 15%
  - Eviction is LRU based with some weighting involved (prefer to evict "small" sized cache entries, i.e. small segments of a shard)
  - Can add expiration: *indices.cache.filter.expire*, e.g. *10m*.

ital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Caching Filters

- All filters can have a *_cache* flag that controls if they should be cached or not
- The default *_cache* value depends on the filter
  - Core filters that automatically generate a BitSet are cached
    - *term*, *terms*, *range*
  - Filters that work on field values loaded to memory (field data) are not cached
    - *geo_distance*, *geo_bbox*, *script*
  - Compound filters are not cached
    - *bool*, *and*, *or*, *not*

ital Kolle 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Caching Filters

- All filters can have a custom filter *_cache_key*
  - The value is used as the unique representation of the filter instance (type + parameters)
  - Handy to reduce memory footprint
    - A terms filter with 1000 values needs to use those 1000 values as part of the unique identification of the filter
    - Application level knowledge can assign application key to it that reduce the memory footprint
    - Under some acceptable cases, can, for example, SHA it

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Caching Filters

- Date filters using *now* expression are never cached
- Default caching behaviour might not be acceptable
  - Mainly for filters that are by default cached, but do not tend to repeat
    - For example, a range filter on a date field with millisecond level resolution that does not repeat
- Consider turning on caching
    - Compound filters that tend to repeat using the exact same filter clauses
    - *geo_bbox* filters with a bounding box that represents a "hot" area

Lital Kotler 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Warmer API

- Segment level caches (e.g. fielddata & filter) are populated lazily on first use when required
- They are also bound to the lifetime of the segment
- Thus
  - They will need to be populated on refresh (as new segments are added to the search)
  - They are invalidated on merge (when multiple old segments are merged into a bigger one)
- Remember, this can happen while users are searching (thus, hinder performance and user experience)

Vital Kolos - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Warmer API

- Warmer API is a mechanism that enables elasticsearch to automatically "warm" (populate) the caches prior to user first use
- Works by registering "warming" queries
  - The queries will run & populate the caches as part of the refresh process, before the new "point in time" view of the index is exposed to the user for search
  - Minimizes the chances for the user to hit an empty cache
- There are several mechanism of registering queries

Michal Kologrzy 08-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Warmer API

At index creation

```
curl -XPUT localhost:9200/crunchbase -d '{
    "warmers" : {
        "warmer_1" : {
            "types" : [ 'company' ],
            "source" : {
                "query" : {
                    ...
                },
                "aggregations" : {
                    ...
                }
            }
        }
    }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Warmer API

- Registering queries to an existing index

```
curl -XPUT 'localhost:9200/crunchbase/company/_warmer/warmer_1' -d '{
    "query" : {
        ...
    },
    "aggregations" : {
        ...
    }
}'
```

Lital Kolog - 28-Apr-2015

- Deleting

```
curl -XDELETE 'localhost:9200/crunchbase/_warmer/warmer_1'
```

- Getting

```
curl -XGET 'localhost:9200/crunchbase/_warmer/warmer_1'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Warmer API

- Note
  - It is important to carefully choose the warming queries
    - Make sure they warm up the field data cache (mainly used for sorting & aggregations)
    - Specify frequently used filters
  - But don't go overboard
    - The more expensive the queries and the more queries you register, the longer it takes for the refresh to complete

Vital Kozlov - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Highlighting

- Showing snippets of text matching the query
  - Per hit
- Text analysis plays an important role here
- Original text needs to be available
  - Automatically extracted from *_source*, which is stored by default

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Highlighting

```
curl -XGET 'localhost:9200/crunchbase/company/_search' -d '{
   "query" : {
      "match" : {
         "overview" : "social network"
      }
   },
   "highlight" : {
      "fields" : {
         "overview" : {}
      }
   }
}'
```

```
{
   ...
   "highlight":{
      "overview" : [
         "<p>GitHub is a <em>social</em> <em>network</em> for
 programmers. Git is a distributed software management
 program", ...
      ]
   }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Highlighting - highlighters

- *highlighter*
  - Requires re-analysis of the text to be highlighted
  - No need for extra information during indexing
  - Does not support all query types, and fails on "complex" queries
- *FVH: fast-vector-highlighter*
  - Requires storing term vectors during indexing
    - Considerably increases the size of the index
  - No need for re-analysis of the text
  - Better support for complex query types

Lital Kovea - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Highlighting - highlighters

- *postings*
  - Requires *index_options* to be set to *offsets* in the mappings
  - Sentence based
  - Doesn't require re-analysis of the text to be highlighted
  - Doesn't require term vectors to be enabled
  - Not as accurate as the other highlighters
    - E.g. when executing phrase queries, all terms in the phrase will be highlighted regardless of their position

Lital Kohen 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Highlighting

- Highlighter type automatically chosen
  - Based on the mapping of the highlighted field
    - The *highlighter* highlighter is used by default
    - If term vectors are enabled, the *FVH* highlighter will be used
    - If *index_options* is set to *offsets*, the *postings* highlighter will be used
  - Options can be specified per field or globally

Lital Kolog - 28-Apr-2015

attr. name	default	description
number_of_fragments	*5*	The number of fragments to show per field. When set to 0 the whole field with with highlight markers will be returned.
pre_tags/post_tags	**	The text to place before and after the snippet
fragment_size	*100*	The size of each document fragment in bytes/ characters

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

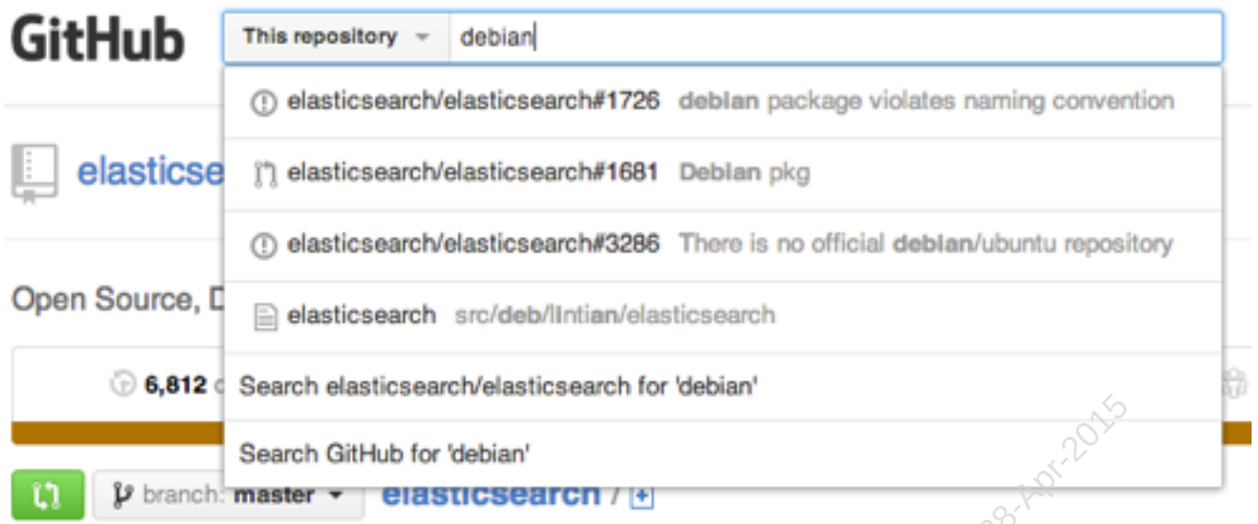
_suggest

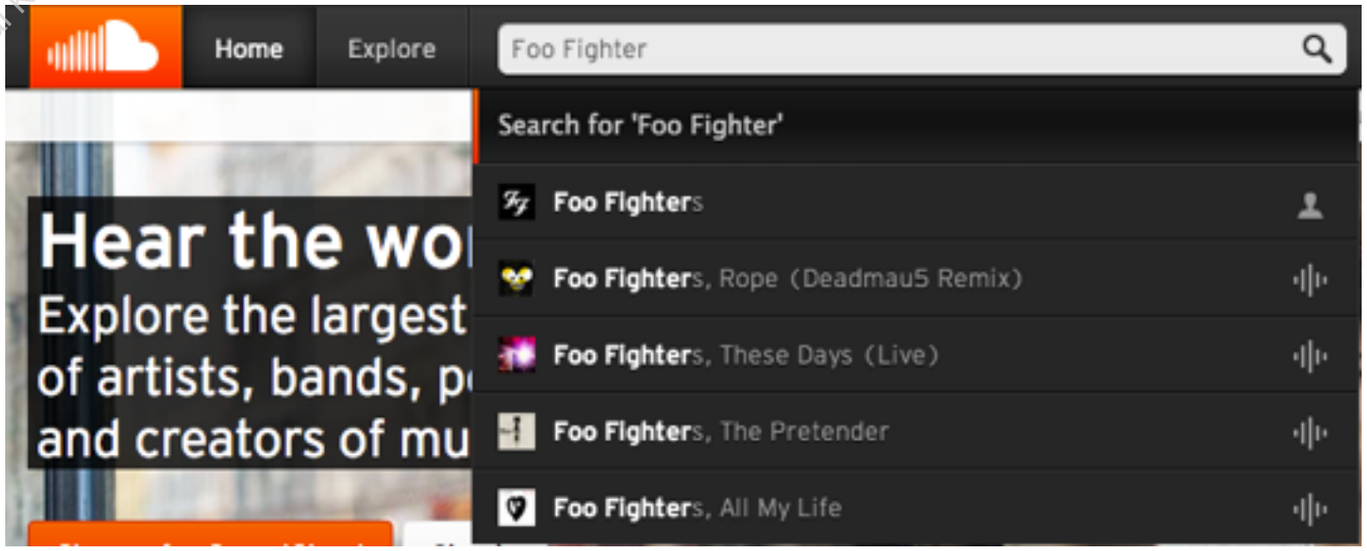
Lital Kolog - 28-Apr-2015

elastic

Copyright Elasticsearch 2013. Copying, publishing and/or distributing without written permission is strictly prohibited

Suggestions





Lital Kolog - 28-Apr-2015

Copyright Elastic 2013. Copying, publishing and/or distributing without written permission is strictly prohibited
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Why suggestions?

- Can often be solved with queries, but
  - Speed is a key factor (requires to run faster than queries)
  - Hit based response data structure is not optimal for extracting suggestions, needs client side parsing
- Can be used in addition to a search
  - Useful if no hits are returned, but a fuzzy query is too verbose
- Suggestions work on an index, not on a type!
- Different Suggesters: *term*, *phrase*, *completion*

Lital Kobi 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query structure

Search API

```
curl 'localhost:9200/crunchbase/company/_search' -d '{
  "query" : {
    // query
  },
  "suggest" : {
    "<ID>" : {
      "text" : "elasti",
      "<TYPE>" : {
        // type specific settings
      }
    }
  }
}'
```

Suggest API

```
curl 'localhost:9200/crunchbase/_suggest' -d '{
 "<ID>" : {
   "text" : "elasti",
   "<TYPE>" : {
     // type specific settings
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Response structure

```
{
  "suggest": {
    "<ID>": [
      {
        "text" : "elasti",
        "offset": 4,
        "length": 9,
        "options": [
          {
            "text": "elasticsearch",
            "score": 34
          }
        ]
      },
    ],
    "<ID>" : [
      ...
    ]
  }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2013. Copying, publishing and/or distributing without written permission is strictly prohibited
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Term

- Based on edit distance
- Takes text as an input, and analyzes it
  - breaking it down to individual tokens in the process
- Suggestions are returned per analyzed token
  - Sorted by terms frequency or by score

Kolog - 28-Apr-2015

```
curl 'localhost:9200/crunchbase/company/_search' -d '{
  "suggest" : {
    “title_suggestions" : {
      "text" : "elasti",
      "term" : {
        "field" : "title"
      }
    }
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Term

- Supports *suggest_mode* parameter
  - *missing* - only suggest if the term doesn't exist in the index (default)
  - *always* - always try to find similar suggestions
  - *popular* - find only suggestions occurring more often than the input term

Limit to two suggestions per term

```
curl -XGET 'localhost:9200/stackoverflow/_suggest' -d '{
  "suggest" : {
    “title_suggestions" : {
      "text" : "rubi on rail",
      "term" : {
        "field" : "title",
        "suggest_mode" : "always"
        "size" : 2
      }
    }
  }
}'
```


elastic
www.elastic.co
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Term

Term: *rubi*

```
{
  "text": "rubi",
  "offset": 0,
  "length": 4,
  "options": [
    {
      "text": "ruby",
      "score": 0.75,
      "freq": 1931
    },
    {
      "text": "rubs",
      "score": 0.75,
      "freq": 4
    }
  ]
}
```

Term: *rail*

```
{
  "text": "rail",
  "offset": 8,
  "length": 5,
  "options": [
    {
      "text": "rails",
      "score": 0.75,
      "freq": 811
    },
    {
      "text": "rais",
      "score": 0.75,
      "freq": 202
    }
  ]
}
```

Lital Kolog - 28-Apr-2015

Note, in this example, the term *on* was dropped in the analysis process

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Phrase

- Selects entire corrected phrases instead of individual terms
  - Relieves the client from merging individual terms
- Weight based on ngram-language models
- Many options: separator, highlighting, error percentage
- Smoothing: Used to balance weight between infrequent/ frequent grams

Lital Kolog - 28-Apr-2015

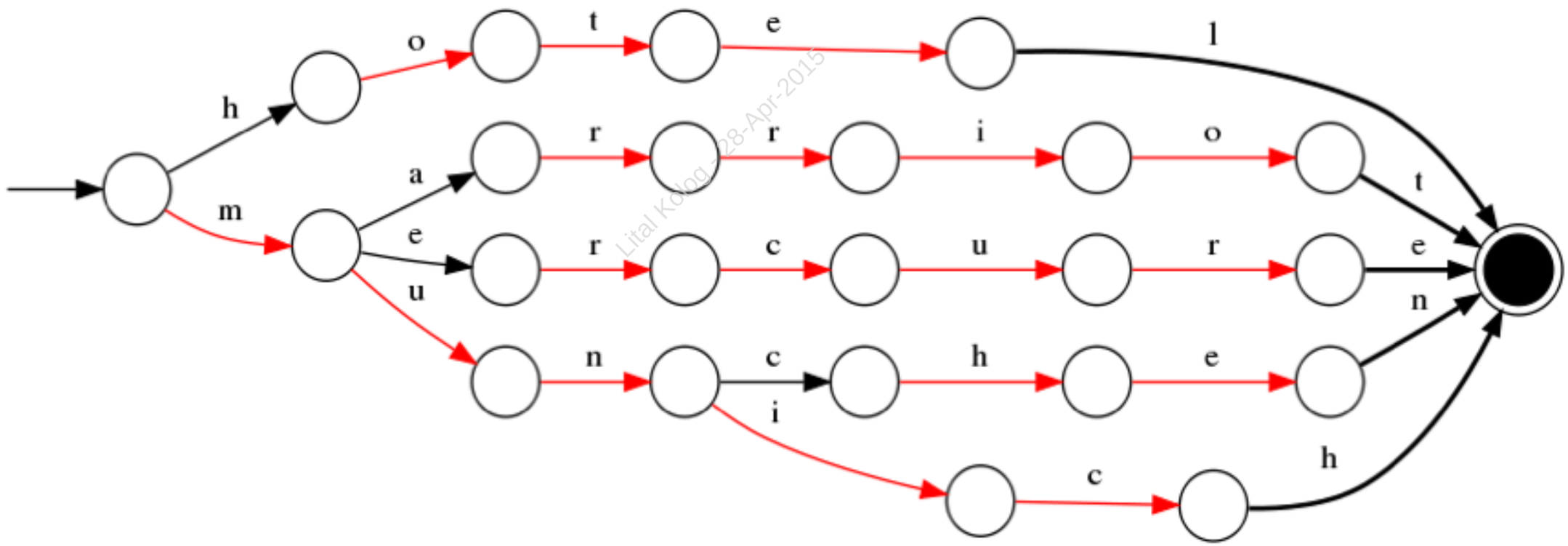
Input `Rubi on Rail`

Suggestion `Ruby on Rails`

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Completion

- Auto-complete, search-as-you-type
- Prefix suggester using FSTs
- Stores FST data structure on disk, which can easily be loaded into memory, going sub millisecond



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Completion

- Requires custom mapping

```
curl -X PUT localhost:9200/music/song/_mapping -d '{
  "song" : {
    "properties" : {
      "name" : { "type" : "string" },
      "suggest" : {
        "type" : "completion",
        "analyzer" : "simple",
        "payloads" : true
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Completion

- Optional: weight, payload & output
- Optional: Multiple inputs
- Using the output to merge entries (weight wins)

```
curl -X PUT 'localhost:9200/music/song/1' -d '{
    "name" : "Nevermind",
    "suggest" : {
        "input": [ "Nevermind", "Nirvana" ],
        "output": "Nirvana - Nevermind",
        "payload" : { "artistId" : 2321 },
        "weight" : 34
    }
}'
```

Search terms

The actual suggestion that will be returned

score

Payload - will be returned along with the suggestion

Paul Kolog - 22-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Completion

```
curl -X POST 'localhost:9200/music/_suggest' -d '{
    "song-suggest" : {
        "text" : "n",
        "completion" : {
            "field" : "suggest"
        }
    }
}'
```

```
{
  ...
  "song-suggest" : [
    {
      "text" : "n",
      "offset" : 0,
      "length" : 1,
      "options" : [ {
        "text" : "Nirvana - Nevermind",
        "score" : 34.0,
        "payload" : { “artistId":2321 }
      ]
    }
  ]
}
```

Output

Score

The payload

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggesters - Fuzzy

- Allows for typos, will still match on *Nirvana*
- Using *edit_distance=1* will not match

```
curl -X POST 'localhost:9200/music/_suggest?pretty' -d '{
  "song-suggest" : {
    "text" : "nurvun",
    "completion" : {
      "field" : "suggest",
      "fuzzy" : {
        "edit_distance" : 2
      }
    }
  }
}'
```

Lilia Kologi 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Suggestions for Suggestions

- Suggestion strategy: Index, monitor, adapt, repeat
  - Analyze & change inputs, weights, output
- Full analysis chain at work (synonyms, analyzer)
- Avoid huge payloads
- Monitor your FST sizes (per index)

Karel Kolos 28-Apr-2015

```
curl 'localhost:9200/_stats/completion' -d '{
 ...
   "completion" : {
    "size_in_bytes" : 163691700,
    "size": "156.1mb"
    }
...
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Relevancy & Boosting

Lital Kolog - 28-Apr-2015

elastic

Relevancy

- How well does a document match a query?
- Lets use a simple example:

step	query	d1	d2
The text	*brown fox*	*The quick brown fox likes brown nuts*	*The red fox*
The terms	*(brown, fox)*	*(brown, brown, fox, likes, nuts, quick)*	*(red, fox)*
A frequency vector	*(1, 1)*	*(2, 1)*	*(0, 1)*
Relevancy	-	*2?*	*1?*

Lital Kolog - 28-Apr-2015

Note, weights (how important is a term, for example) here are all 1

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

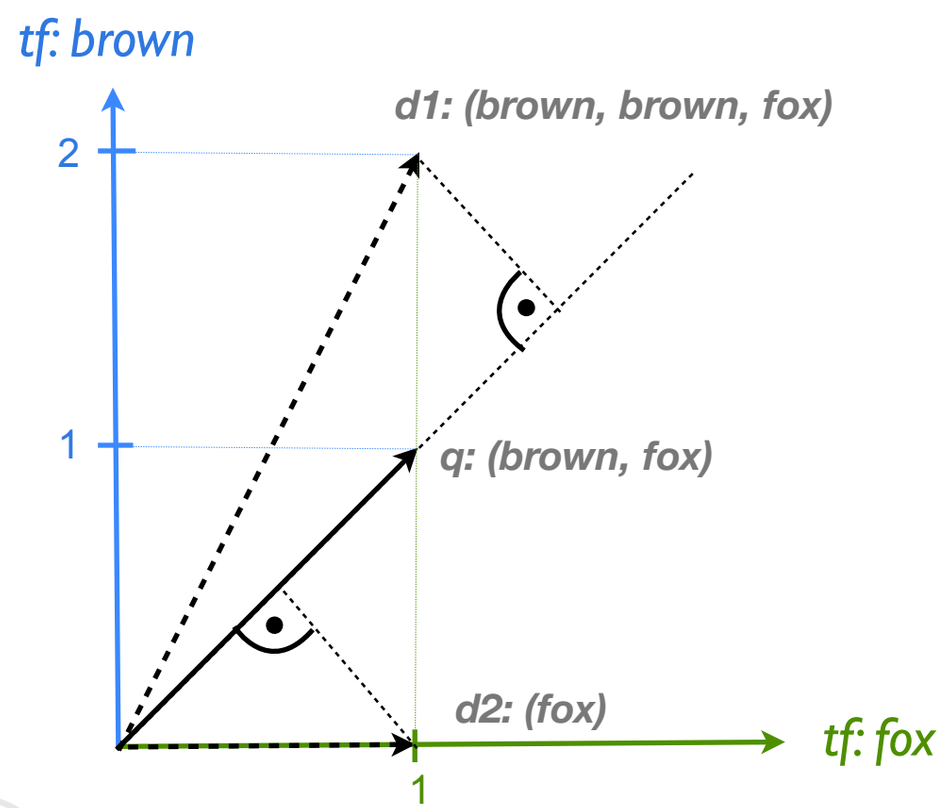
Relevancy - Vector Space Model

- How well *q* matches *d1* and *d2*?
  - The coordinates in the vector represent weights per term
  - The simple *(1, 0)* vector we discussed defines these weights based on the frequency of each term
  - But to generalize:

$$d_j = (w_{1,j}, w_{2,j}, ..., w_{t,j})$$

$$q = (w_{1,q}, w_{2,q}, ..., w_{t,q})$$

Lital Kolog - 28-Apr-2015



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Relevancy TF-IDF

- A way to compute the weight of a token in a document
  - We can do better than the previous 0/1
    - Term frequency / Inverse Document Frequency
- It increases proportionally to the number of times a token appears in the doc
  - The more a token appears in a doc, the more important it is
- But is offset by the frequency of the token in the document collection
  - The more documents containing the term, the less important it is
    - Think very common words in the collection ("money" in financial data)

Lital Kolog - 28-Apr-201

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Relevancy TF-IDF

$$w_{t,d} = \mathrm{tf}_{t,d} \cdot \log \frac{|D|}{|\{d' \in D \mid t \in d'\}|}$$

Lital Kolog - 28-Apr-20

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Lucene Ranking Formula

Score of a document for a given query

Normalized doc length, shorter docs are more likely to be relevant than longer docs

$$score_{q,d} = norm(q) \times \sum_{t \text{ in } q} \underbrace{\sqrt{tf_{t,d}} \times idf_t^2}_{\text{Core TF/IDF weight}} \times norm(d, field) \times boost(t)$$

Lital Kolog - 28-Apr-2015

Can be ignored (was an attempt to make query scores comparable across indices, it's there for backward compatibility)

Boost of query term t

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Explaining - Search API

```
curl -XGET 'localhost:9200/crunchbase/company/_search?pretty' -d '{
  "explain" : true,
  "query" : {
    "match" : { "overview" : "social network" }
  }
}'
```

```
"value" : 0.1058217,
"description" : "fieldWeight(overview:network in 0), product of:",
"details" : [
    {
      "value" : 1.0,
      "description" : "tf(termFreq(overview:network)=1)"
    },
    {
      "value" : 1.6931472,
      "description" : "idf(docFreq=4, maxDocs=10)"
    },
    {
      "value" : 0.0625,
      "description" : "fieldNorm(field=overview, doc=0)"
    }
]
```

Lital Kolog - 22-Apr-2015

Note, explanation is an expensive process

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Explaining - Explain API

- Allows to explain a query on a specific document
  - Also help to understand why a document does not match a query

index type id

```
curl -XGET 'localhost:9200/crunchbase/company/1/_explain' -d '{
  "query" : {
    "match" : { "overview" : "social network" }
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Influencing Relevancy

- Boosting
  - Query time, on a per query basis
- Custom Scoring
  - Based on per document values
  - Or specific criteria that a document match on

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Boosting - query time

```
"bool" : {
   "should" : [
      {
         "match" : {
              "name" : {
                  "query" : "microsoft"
              }
         }
      },
      {
         "match" : {
              "overview" : {
                  "query" : "apple",
                  "boost" : 2
              }
         }
      }
   ]
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Combining score with numerical fields

- Relevance of a document can depend on numerical values
  - Typically derived from numeric fields
    - Distance from a location
    - Time passed since document creation and query
- Can be combined with query
  - Find documents matching a specific criteria and compute relevance using numerical values

Lital Kuzey, 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: function_score

- Score is recomputed for docs matching specific filters
- Scores can be combined in different ways
  - multiply, add,...
- Function score can be combined with query score
  - multiplication, addition, replace...

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: function_score

```
"function_score": {
    "(query|filter)": {},
    "boost": 2,
    "functions": [
        {
            "filter": {},
            "<FUNCTION>": {}
        },
            ...
    ],
    "max_boost": 10.0,
    "score_mode": "(mult|max|...)",
    "boost_mode": "(mult|replace|...)"
}
```

Compute score only for docs matching the filter

Apply a function on the matching docs

Limit the final boost

How to combine scores from multiple functions

How to merge the query score with the function score

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Query: function_score - functions

- *boost_factor*
  - Apply boost that is not normalized
- Decay functions
  - Boost depending on distance of numerical value from a user given point of reference
    - available functions: *gauss*, *exp*, *linear*
- *script_score*
  - Define your own script for flexible/complex logic
- *random_score*
  - Compute a random score, useful for shuffling results

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

boost_factor example 1

- The function applies a boost_factor to a score
  - The boost is not normalized

Check if ingredients contains the term “apple”

```
"function_score" : {
    "query": { "term" : { "recipe" : "cake" }},
    "functions": [
        {
            "filter" : { "terms" : { "ingredients" : ["apple"] }},
            "boost_factor" : 2
        }
    ]
}
```

If so: Score with 2

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

boost_factor example 2

```
"function_score" : {
  "query" : { ... },
  "functions" : [
    {
      "filter" : { "range" : { "age" : { "lt" : 10 } } },
      "boost_factor" : "3"
    },
    {
      "filter" : { "range" : { "age" : { "gte" : 5 } } },
      "boost_factor" : "2"
    }
  ],
  "score_mode" : "first"
}
```

Lital Kolog - 28-Apr-2015

score_mode controls how boost computation is done (which is later combined with the query score). Supported: *first* (to only use the first filter that matched), and *min*/*max*/*avg*/*multiply* aggregation styles.

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Decay functions

- Diminishes score depending on distance of a reference point
- Example: Search for...
  - ... a hotel near a specific location
  - ... a hotel with low prices
  - ... tweets that were tweeted around a certain date
- Decreases score with a factor that depends on distance
- Works for *numeric*, *date* and *geo_point* fields

Lital Kovas - 28-Apr-2015

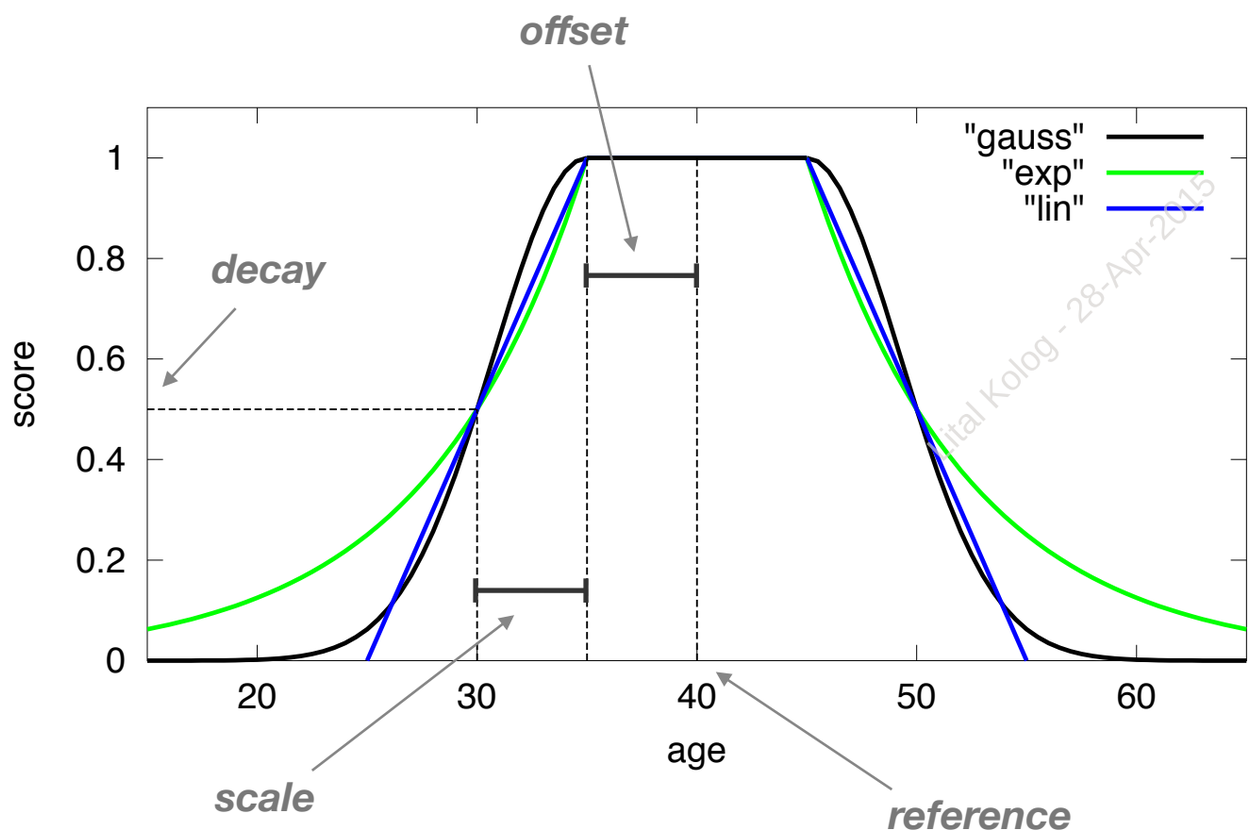
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Decay functions

Shape of decay function,
options: *gauss*, *exp*, *linear*

Field name

```
"gauss": {
    "age": {
        "reference": 40,
        "scale": 5,
        "decay": 0.5,
        "offset": 5
    }
}
```



https://www.desmos.com/calculator/yw0rlo9zmc

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

function script_score

```
"script_score" : {
   "script" : "_score * log10(doc['popularity'].value + 1)"
}
```

The score produced
by the wrapped query

Causes the popularity field values
to be loaded to memory
for faster access

Lital Kolog - 28-Apr-2015

```
"script_score" : {
  "params" : { "mean" : 2, "sigma" : 3 },
  "script" : "_score * (log10(doc['popularity'].value + 1)-mean) / sigma"
}
```

Scripts are compiled and cached for faster execution,
make sure to reuse scripts with parameters

Copyright Elastic 2015 Copying, publishing and/or
distributing without written permission is strictly prohibited

function random_score

```
"query" : {
   "function_score" : {
      "query" : { "match_all" : {}}
   },
   "functions" : [
      { "random_score" : { "seed" : 1376773391128418000 }}
   ]
}
```

Lital Kolog - 28-Apr-2015

Use a seed for consistent scoring across multiple requests

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

function_score: final example

Searching for hotels

```
"function_score": {
    "query": {"term": {"category": "budget"}},
    "functions": [
        {
            "filter": {"terms": {"facilities": ["balcony"]}},
            "boost_factor": 2
        },{
            "gauss": {
                "field": "location",
                "scale": "1km",
                "reference": [51,0]
            }
        },{
            "script": “log10(doc['popularity'].value + 1)"
        },{
            "random_score": {}
        }
    ],
    "max_boost": 10,
    "score_mode": "sum",
    "boost_mode": "replace"
}
```

Hotels with balcony get +2

Close to London central

Factor in popularity

Shuffle a little, to make sure that rarely booked or new hotels get a chance

Cut off boost at 10

Sum up result of the functions

Ignore query score

Lior Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Scripts

- Elasticsearch has an abstraction of a script engine
- Different engines are supported
  - Groovy - Java oriented (default)
  - Other dynamic languages
    - javascript, MVEL, Lucene expressions (official es embedded/plugins)
    - python, clojure, scala (community plugins)
- Scripts are used in many places across the APIs
    - Update API
    - Sorting & Boosting
    - Aggregations

Lital Kabel - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Scripts

- There are several ways to access field values
  - Use the appropriate one based on the nature of the API

`doc['popularity']`	Loads the field value from the *fielddata* cache (thus, the value is loaded from the index). Very fast at the cost of memory consumption
`_source.popularity`	Loads the field value from the *_source*. (the source must be stored). Slow & heavy operation
`_field.popularity`	Loads the field value from the *stored field*. Might be lighter than *_source*, yet still very heavy & slow compared with doc notation

Gal Kobi - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregations

(a.k.a. aggs)

Lital Kolog - 28-Apr-2015

elastic

Introduction to aggregations

- While executing queries dramatically improve the "findability" of information, as data scales increase, so does the challenge
  - Due to the number of document matching a query
  - And often due to the "ignorance" of the user with regard to what the data holds (i.e. the user don't know what to search for)
- Faceted navigation allows the refinement of the search queries by providing insight into both the search results and the data as a whole
  - i.e. further restrict the search to certain category, price or date range.

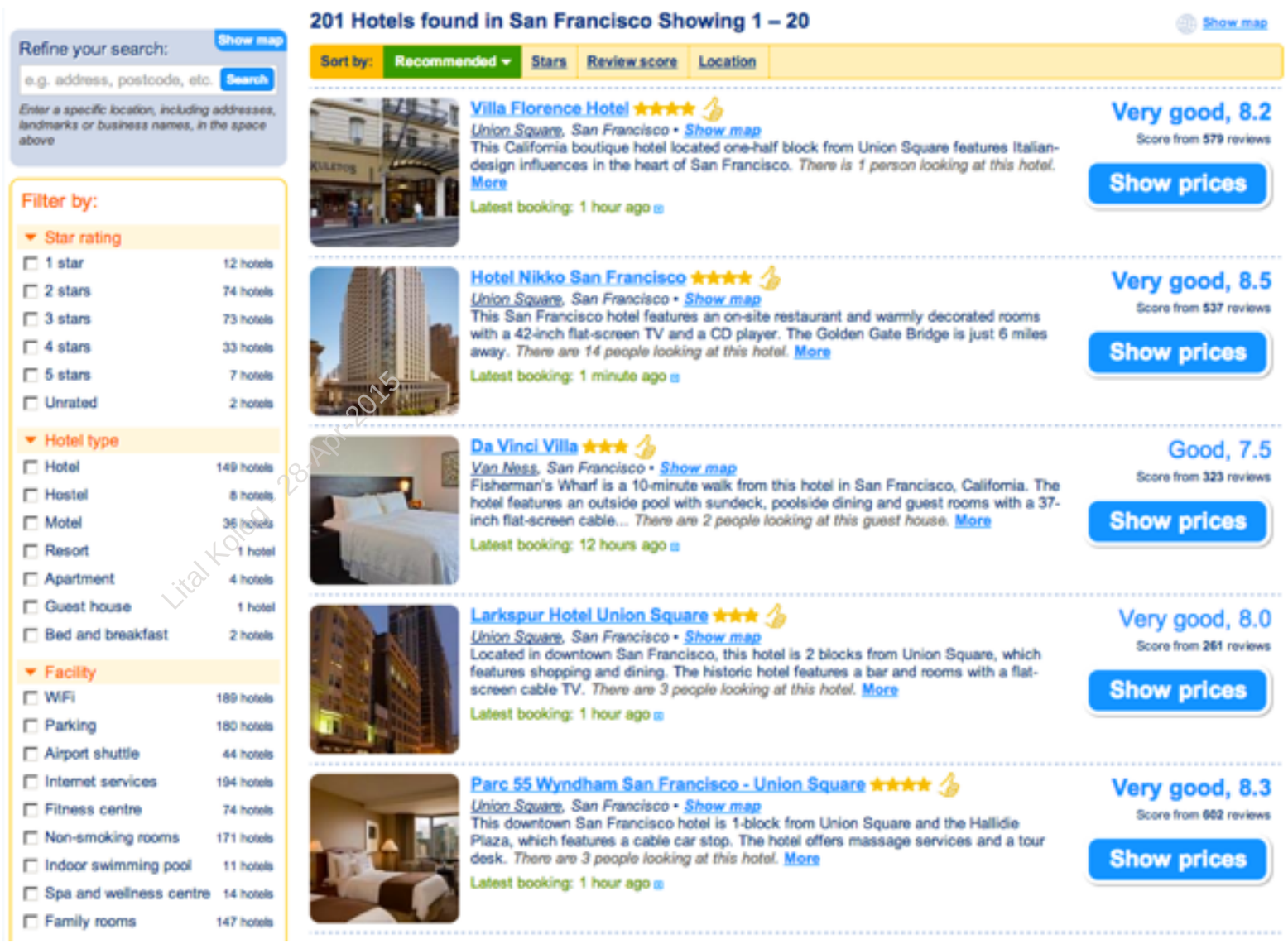
Josef Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Introduction to aggregations

Free text search

Facets



Lital Kolog 28 Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

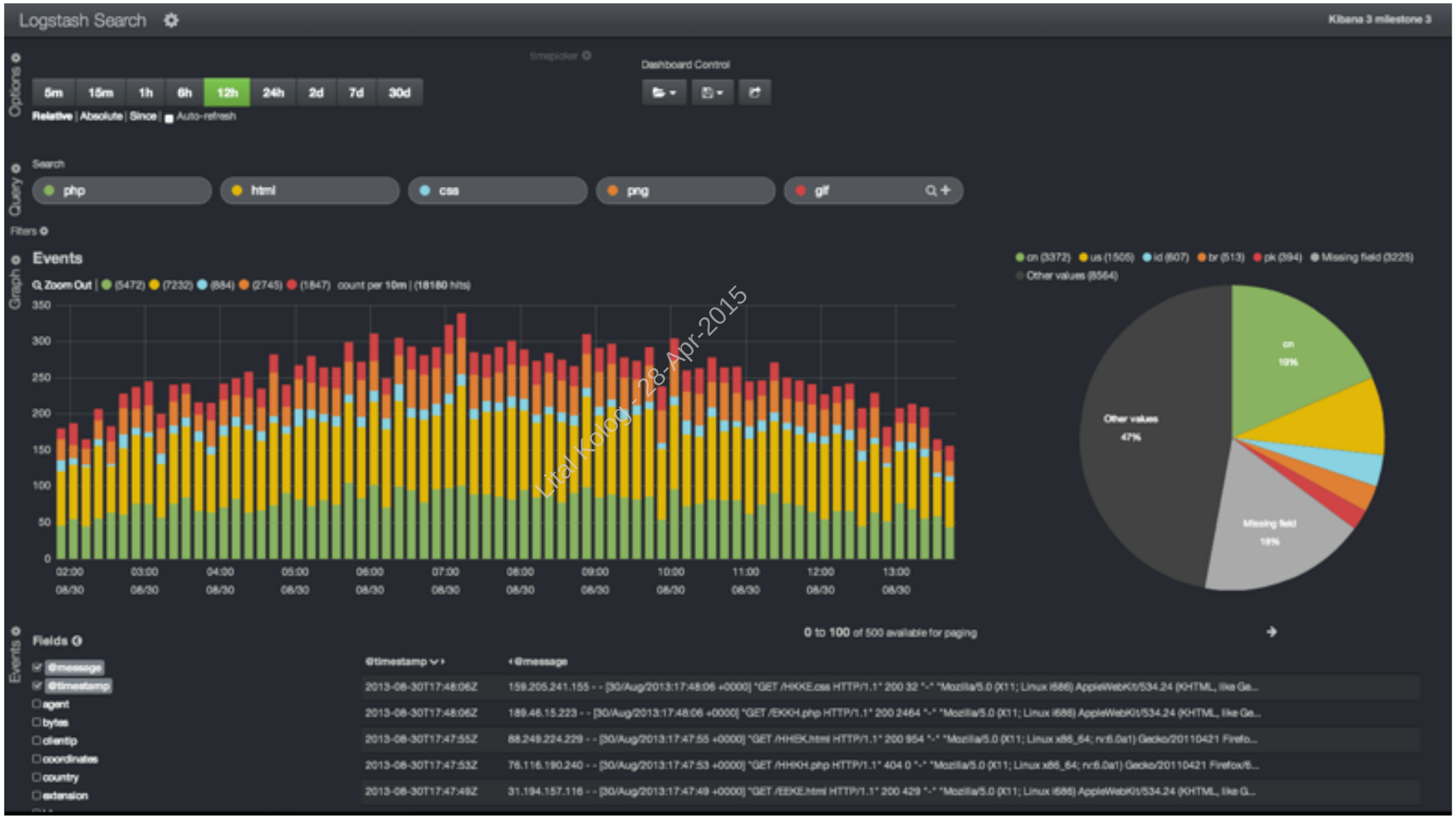
Introduction to aggregations

- We've seen how the inverted index at the core of the search engine accounts for the fast evaluation of queries with fine level of control over ranking
- The same inverted index can also be used for real-time data analytics, that can be computed on demand along side the executed search queries
- Aggregations enable slicing & dicing the data within the different (dynamic) search contexts
- Effectively turning *elasticsearch* to a powerful real-time analytics engine

Paul Kologrivov 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Introduction to aggregations



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

But wait… what about facets?

- Yes, elasticsearch had exposed data analytics functionality for quite some time now - facets
- And yes, facets are/were super powerful and essentially turned elasticsearch to a data analytics engine
- But…
  - When facets were first introduced, we didn't have enough knowledge as for how far users will push it. The requirements & features were introduced by the **users** over time and turned *facets* into the power tool they are today.
  - We learned a lot about how users approach data and how they like to view/analyze it, and we wanted elasticsearch to reflect it - build the next generation data analytics tool - *Aggregations*

Eyal Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

"Just" another search tool

- Aggregations are another search tool
  - Remember, for us, search = exploring information and extracting knowledge
- As all other search tools, they are
  - Near realtime in their nature
  - Executed as part of the *_search* API
  - Relative to the search context (which essentially defines a doc set)

```
curl 'localhost:9200/crunchbase/company/_search' -d '{
  "query" : {
    // query
  },
  "aggregations" : {
    // a set of aggregation to compute
  }
}'
```

Vital Kolesov - 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

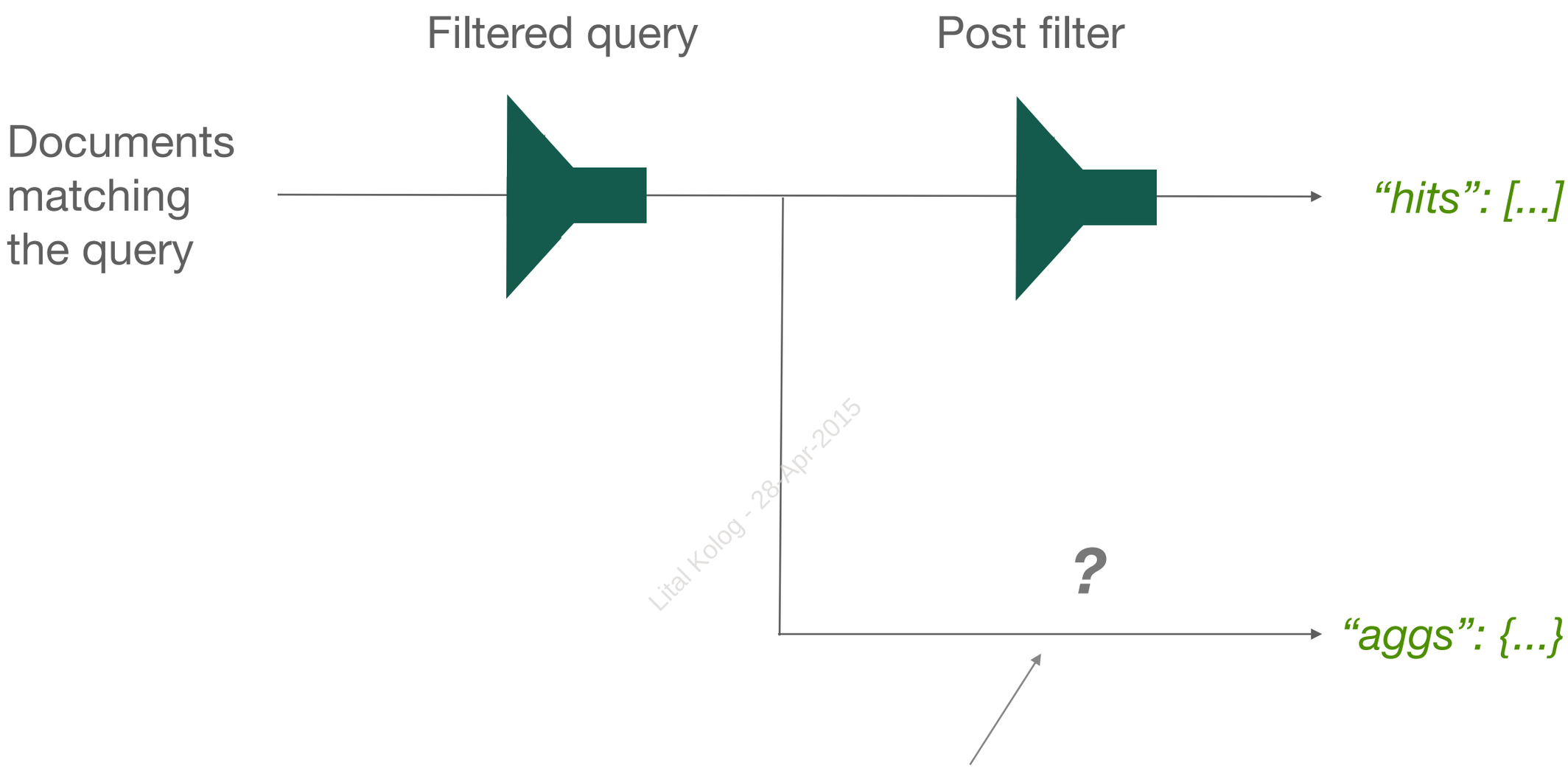
Aggregations scope

- Aggregations scope refers to the document set on which they are computed
- The default scope is the query scope (a.k.a *main*)
  - I.e. the set of documents that matched the query
- The scope is influenced by *Filters*, two types of them:
  - A *filtered_query* is defined **in** the query part, hence impacting the *main* scope, and in turn filtering the document set on which the aggregations are executed on
  - A *post_filter* is a special filter that is defined **outside** of the query part, thus not impacting the *main* scope, that is, not impacting the document set for the aggregations (it only impacts the returned *hits*)

Lital Vashdi - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregations scope



Wouldn't it be nice to have another filtering option here? We'll get to that later…

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregation scope

The aggregation will be calculated over all “John”s between the ages of 20 to 50

```
curl 'localhost:9200/address-book/_search' -d '{
  "query" : {
    "filtered" : {
      "query" : { "match" : { "name" : "John" }}
      "filter" : {
        "range" : { "age" : { "from" : 20, "to" : 50 }}
      }
    },
  "aggregations" : {
    "employer" : {
      "terms" : {
        "field" : "company"
      }
    }
  }
}'
```

Lital Volcani - 25-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregation scope

The search hits will only include the “John”s between the the ages of 20 and 50

The aggregations will be calculated over all “John”s, regardless of their age

```
curl 'localhost:9200/address-book/_search' -d '{
  "query" : {
    "match" : { "name" : "John" }
  },
  “post_filter" : {
    "range" : { "age" : { "from" : 20, "to" : 50 }}
  },
  "aggregations" : {
    "employer" : {
      "terms" : {
        "field" : "company"
      }
    }
  }
}'
```

Lital Kozlovski - AU 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Buckets & Metrics

- Two categories of aggregations - *Buckets* & *Metrics*
- Buckets
  - Aggregations that build buckets. Each bucket is associated with some criteria over documents. During query execution, each document is evaluated against the created buckets and each bucket keeps track of what documents "fall" in it. Each bucket effectively defines a set of documents derived from the document set within the aggregations scope.
- Metrics
  - Aggregations that given a set of documents, produce a single/multiple scalar/s. Typically, metrics aggregations generate numeric stats that are computed over a specific document set

Lital Kolog - 28-Apr-2025

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Buckets & Metrics

- Bucket aggregations can contain sub-aggregations (which can be either *bucket* or *metrics* aggregations)
- When nesting an aggregation *A2* **under** a bucket aggregation *A1*, each bucket generated by *A1* will define the document set for *A2* to operate on
  - Meaning, *A2* will be computed for each of the buckets created by *A1*
- Enables the construction of complex aggregations hierarchies
- When a new aggregation is introduced, all other bucket aggregations are implicitly "extended"

Joel Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregation Anatomy

Grammar

```
aggs       := ("aggregations" | "aggs") : { [ agg [,agg]* ]? }
agg        := "<agg_name>" : { (bucket_agg | metric_agg) }
metric_agg := "<metric_agg_type>" : { <metric_agg_settings> }
bucket_agg := "<bucket_agg_type>" : { <bucket_agg_settings> }[, aggs]?
```

Example

```
{
  "aggs" : {
    "countries" : {
      "terms" : { "field" : "country" },
      "aggs" : {
        "subjects" : {
          "terms" : { field" : "subject" },
          "aggs" : {
            “avg_score" : {
              "avg" : { "field" : "score" }
            }
          }
        }
      }
    }
  }
}
```

Bucket aggs

Aggregations names

Metrics agg

Lukas Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregation Anatomy

Aggregations names

```
{
  "hits" : { ... },
  "aggregations" : {
    "countries" : {
      "buckets" : [
        {
          "key" : "USA",
          "doc_count" : 5
          "aggregations" : {
            "subjects" :
              "buckets" : [
                {
                  "key" : "Mathematics",
                  "doc_count" : 3,
                  "aggregations" : {
                    "avg_score" : {
                      "value" : 87.5
                    }
                  }
                },
                ...
              ]}},
        ...
      ]}
  }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metrics - extended_stats

- Given a set of documents, enables extracting values from each document and compute statistics over those values across all documents in the set
  - *min*, *max*, *sum*, *avg*, *value_count*, *sum_of_squares*, *std_deviation*, *variance*
- Most commonly, the values will be extracted from a specific field in the document
- Scripts are supported as well
  - Scripts that can operate on the document level and generate values
  - Scripts that can operate on the values of a specific field and transform them to other values

Lal Kolog - 28-Apr-2017

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metrics - extended_stats

```
{
  "subject": "Mathematics",
  "state": "CA",
  "age": 8,
  "grade": 69,
  "male": true
}
```

Sample document

Request

```
{
  "aggs" : {
    "grades" : {
      "extended_stats" : {
        "field" : "grade"
      }
    }
  }
}
```

Response

```
"aggregations": {
  "grades": {
     "count": 4375,
     "min": 65,
     "max": 99,
     "avg": 82.14765714285714,
     "sum": 359396,
     "sum_of_squares": 29970052,
     "variance": 102.06002593959144,
     "std_deviation": 10.102476228113158
  }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metrics - extended_stats

```
"aggs" : {
  "grades" : {
    "extended_stats" : {
      "script" : "doc['grade'].value * factor",
      "params" : { "factor" : 1.2 }
    }
  }
}
```

Factorizing the grades by *1.2*

Value script support

Lital Kolog - 28-Apr-2015

```
"aggs" : {
  "grades" : {
    "extended_stats" : {
      "field" : "grade",
      "script" : "_value * factor",
      "params" : { "factor" : 1.2 }
    }
  }
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Metrics - others

metrics type	description
stats	Trimmed down version of extended_stats (excludes *sum_of_squares*, *variance* and *std_deviation*)
min	Returns the min value out of all values in the doc set
max	Returns the max value out of all values in the doc set
sum	Cumulates and returns all values in the doc set
avg	Computes and returns the average value in the doc set
value_count	Returns the number of values in the doc set

Lise Kolog - 28-Avr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - global

- Defines a single bucket that contains all documents within the search context (regardless of the executed query)
  - The search context is defined by the searched indices & types

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "query" : {
    "match" : { "subject" : "mathematics" }
  },
  "aggs" : {
    "all_grades" : {
      "global" : {
      }
    }
  }
}'
```

The hits will only return the math grades

Creates a single bucket with all the grades in the *grades* index

Response

```
{
  "hits" : { ... },
  "aggregations": {
    "all_grades": {
      "doc_count": 4375
    }
  }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - global

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "query" : {
    "match" : { "subject" : "mathematics" }
  },
  "aggs" : {
    "all_grades" : {
      "global" : {},
      "aggs" : {
        "avg" : {
          "avg" : { "field" : "grade" }
        }
      }
    }
  }
}'
```

Response

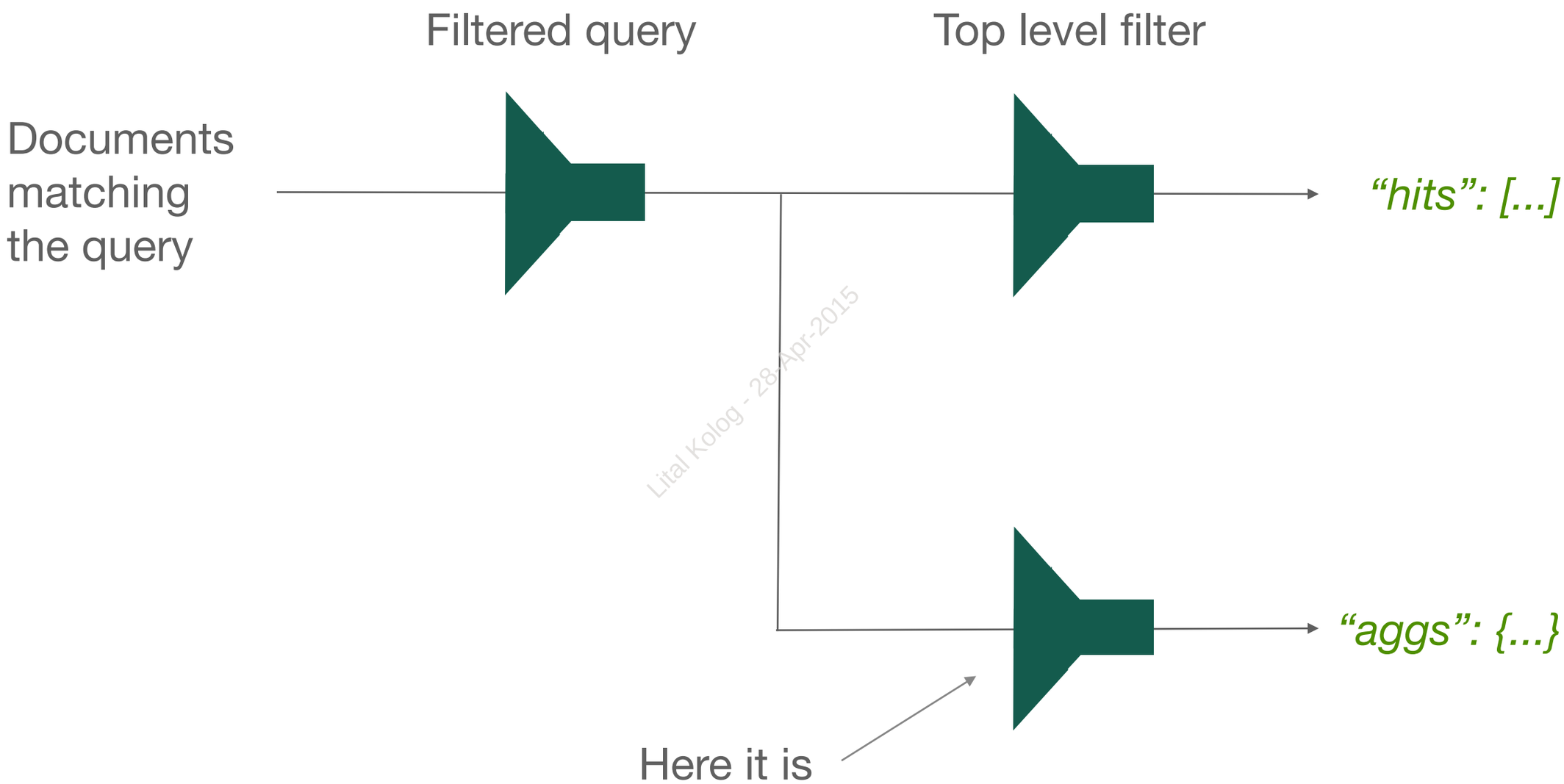
```
{
   "hits": { ... },
   "aggregations": {
      "all_grades": {
         "doc_count": 4375,
         "avg": {
            "value": 82.14765714285714
         }
      }
   }
}
```

The average grade over **all** grades (not just the math ones)

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - filter



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - filter

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
    "aggs" : {
      "math_grades" : {
        "filter" : { "term" : { "subject.raw" : "Mathematics" }}
      }
    }
  }
}'
```

Response

```
{
   "hits": { ... },
   "aggregations": {
      "math_grades": {
         "doc_count": 875
      }
   }
}
```

Creates a bucket of all the math grades

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - filter

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "math_grades" : {
      "filter" : { "term" : { "subject.raw" : "Mathematics" }},
      "aggs" : {
        "max" : {
          "max" : { "field" : "grade" }
        }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Response

```
{
   "hits": { ... },
   "aggregations": {
      "math_grades": {
         "doc_count": 875,
         "max": {
            "value": 99
         }
      }
   }
}
```

The maximum grade out of all the math grades

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - missing

- Defines a single bucket of all documents that **don't** have a specific field

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "no_grades" : {
      "missing" : {
        "field" : "grade"
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

```
{
   "hits": { ... },
   "aggregations": {
      "no_grade": {
         "doc_count": 2
      }
   }
}
```

2 exams were missed…
which ones?

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - missing

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "no_grades" : {
      "missing" : { "field" : "grade" },
      "aggs" : {
        "math" : {
          "filter" : {
            "term": {
              "subject.raw": "Mathematics"
            }
          }
        }
      }
    }
  }
}'
```

```
{
   "hits": { ... },
   "aggregations": {
      "no_grades": {
         "doc_count": 2,
         "math": {
            "doc_count": 1
         }
      }
   }
}
```

Now we see that one of the missed exams is in math. But now do we need to get the subject of the other missed exam?

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - terms

- Extracts values of the documents in the scope and dynamically builds buckets - one per value (*term*)
  - Mostly used to extract the terms from a specific field, but scripts are supported as well

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "states" : {
      "terms" : { "field" : "state" }
    }
  }
}'
```

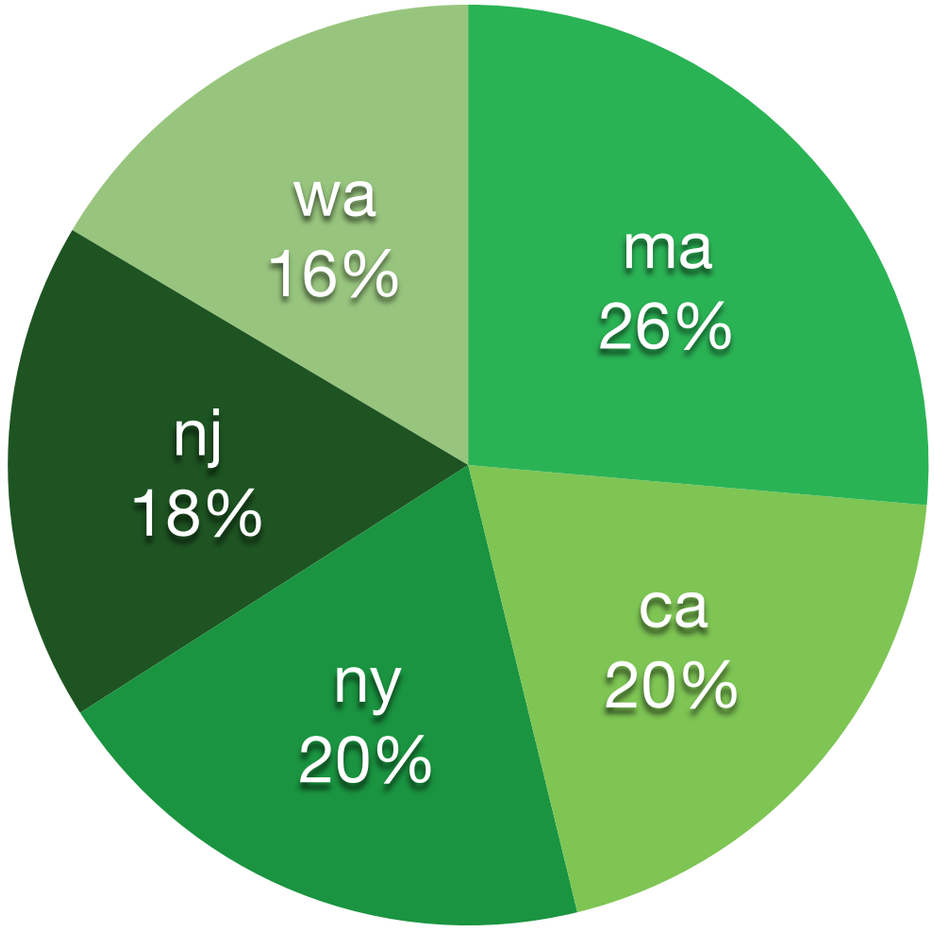
LitoKolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - terms

Response

```
"aggregations": {
  "states": {
    "buckets": [
      {
        "key": "ma",
        "doc_count": 841
      },
      {
        "key": "ca",
        "doc_count": 631
      },
      ...
        "key": "ny",
        "doc_count": 630
      },
      {
        "key": "nj",
        "doc_count": 560
      },
      {
        "key": "wa",
        "doc_count": 525
      }
    ]
  }
}
```



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - terms

- Often with *terms* aggregation, you'd want to get the original (unanalyzed) form of the term
  - Can be achieved with multi-field mapping

```
"mappings" : {
  "grade" : {
    "properties" : {
      "state" : {
        "type" : "string",
        "fields" : {
          "raw" : { "type" : "string", "index" : "not_analyzed" }
        }
      },
      ...
    }
  }
}
```

Perform terms aggregations on *state.raw*

Lital Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - terms

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "states" : {
      "terms" : {
        "field" : "male",
        "script" : "_value == 'T' ? 'Male' : 'Female'"
      }
    }
}'
```

```
{
  "hits": { ... },
  "aggregations": {
    "states": {
      "buckets": [
        {
          "key": "Female",
          "doc_count": 1596
        },
        {
          "key": "Male",
          "doc_count": 1591
        }
      ]
    }
  }
}
```

The document has a boolean field indicating whether or not the student is a male. The script will then generate two terms based on this value - *Male* and *Female*

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - terms

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "no_grades" : {
      "missing" : { "field" : "grade" },
      "aggs" : {
        "subjects" : {
          "terms" : { "field" : "subject.raw" }
        }
      }
    }
  }
}'
```

```
"aggregations": {
  "no_grades": {
     "doc_count": 2,
     "subjects": {
        "buckets": [
           {
              "key": "English",
              "doc_count": 1
           },
           {
              "key": "Mathematics",
              "doc_count": 1
           }
        ]
     }
  }
}
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - range

- Extracts numeric values from the documents and matches them against a specified set of ranges. Each bucket is associated with a range - a document will “fall” in a bucket only if its values matches the bucket’s range.

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "age_groups" : {
      "range" : {
        "field" : "age",
        "ranges" : [
          { "to" : 5 },
          { "from" : 5, "to" : 10 },
          { "from" : 10 }
        ]
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

```
"age_groups": {
  "buckets": [
    { "to": 5, "doc_count": 0 },
    { "from": 5, "to": 10, "doc_count": 911 },
    { "from": 10, "doc_count": 2276 }
  ]
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - range

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "age_groups" : {
      "range" : {
        "field" : "age",
        "ranges" : [
          { "from" : 5, "to" : 10 },
          { "from" : 10 }
        ]
      },
      "aggs" : {
        "avg_grade" : {
          "avg" : {
            "field" : "grade"
          }
        }
      }
    }
  }
}'
```

```
"age_groups": {
  "buckets": [
    {
      "from": 5,
      "to": 10,
      "doc_count": 911,
      "avg_grade": {
        "value": 81.6032967032967
      }
    },
    {
      "from": 10,
      "doc_count": 2276,
      "avg_grade": {
        "value": 82.35736263736264
      }
    }
  ]
}
```

Average grade per age group

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - *_range

- *date_range*
  - A dedicated range aggregations that works on *date* fields. Ranges can be defined as date math expressions

```
{ "from" : "now-1M", "to" : "now" }
```

- *ip_range*
  - A dedicated range aggregation that works on *ip* fields. Ranges can be defined as ipv4 strings

Lita Kolog 28-Apr-2015

```
{ "from" : "10.0.0.0", "to" : "10.0.0.128" }
```

  - or CIDR masks

```
{ "mask" : "10.0.0.0/25" }
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - histogram

- Extract numeric values from the documents and distributes them into fixed-size interval buckets
  - Each value is "rounded" into a bucket
    - A value of *135* and a bucket interval of 10 will "fall" into bucket 130

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "grades_distribution" : {
      "histogram" : {
        "field" : "grade",
        "interval" : 10
      }
    }
  }
}'
```

Lital Kolog - 28-Aug-2015

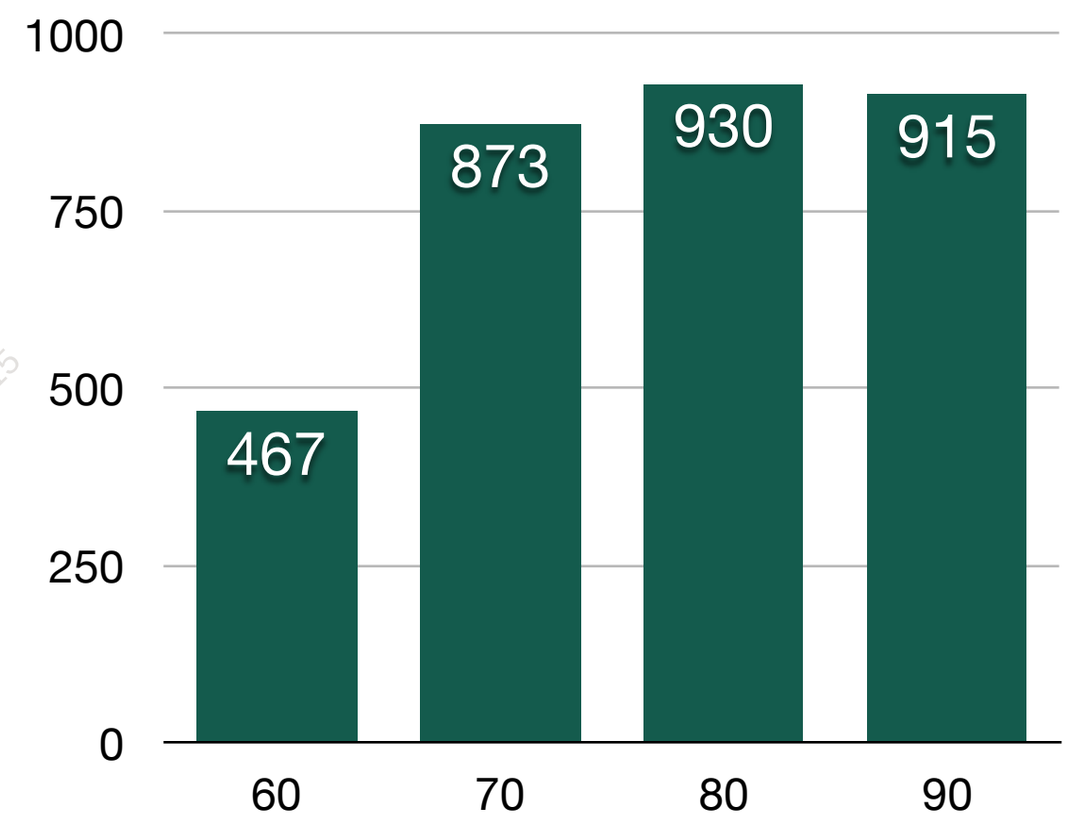
Will define "virtual" buckets of
0, 10, 20, 30, 40,…, 70, 80, 90

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - histogram

Response

```
"aggregations": {
  "grades_distribution": {
   "buckets": [
      {
        "key": 60,
        "doc_count": 467
      },
      {
        "key": 70,
        "doc_count": 873
      },
      {
        "key": 80,
        "doc_count": 930
      },
      {
        "key": 90,
        "doc_count": 915
      }
    ]
  }
}
```



Lital Kolog - 28-Apr-2015

By default, only non-empty buckets will be returned

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - histogram

Computes the distribution of grades across all states

```
curl -XGET 'localhost:9200/grades/grade/_search' -d '{
  "aggs" : {
    "grades_distribution" : {
      "histogram" : {
        "field" : "grade",
        "interval" : 10
      },
      "aggs" : {
        "states" : {
          "terms" : {
            "field" : "state"
          }
        }
      }
    }
  }
}'
```

Lital Vaknin - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Bucket - date_histogram

- A special histogram aggregation that can be applied on date values
  - Supports date/time oriented intervals (“day”, “month”, “1.5w”, etc…)
  - Support pre/post time zone conversion
  - Support scale factorization (conversion from seconds to milliseconds)
- Ideal for analyzing trends over time
  - Great for analyzing time-series data (e.g. logs, social event streams)
  - Think correlation between different stats over same time series

tal Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregations - Example

```
{
  "aggs" : {
    "states" : {
      "terms" : {
        "field" : "state"
      },
      "aggs" : {
        "age_groups" : {
          "histogram" : {
            "field" : "age",
            "interval" : 5
          },
          "aggs" : {
            "grades" : {
              "stats" : { "field" : "grade" }
            },
            "gender" : {
              "terms" : {
                "field" : "male",
                "script" : "_value == 'T' ? 'Male' : 'Female'"
              },
              "aggs" : {
                "grades" : {
                  "stats" : { "field" : "grade" }
                }
              }
          ...
```

Analyze the grades per *state*

Analyze per *age_group*

Stats per *state* & *age_group*

Stats per *state*, *age_group* & *gender*

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregations - Example

```
{
  "aggs" : {
    "states" : {
      "terms" : {
        "field" : "state"
      },
      "aggs" : {
        "age_groups" : {
          "histogram" : {
            "field" : "age",
            "interval" : 5
          },
          "aggs" : {
            "grades" : {
              "stats" : { "field" : "grade" }
            },
            "gender" : {
              "terms" : {
                "field" : "male",
                "script" : "_value == 'T' ? 'Male' : 'Female'"
              },
              "aggs" : {
                "grades" : {
                  "stats" : { "field" : "grade" }
                }
              }
          ...
```

All in a single request!

Lital Kologi 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Aggregations

- percentile
- percentile_ranks
- cardinality
- significant_terms
- top hits
- scripted metric
- filters

- nested
- reverse nested
- children
- geohash grid
- geo_bounds

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Document Relations

Lital Kolog - 28-Apr-2015

elastic

The Impedance Mismatch

- The world is not flat
  - Objects expressed in JSON can have complex hierarchies
  - In the real world, there are relations between different objects which often influences the structure of the data
    - Order and order items
    - Products and SKUs
- Search engines model the world with flat documents
  - Due to the nature of the inverted index backing them
- How can we bridge the two worlds?

Ital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

The Impedance Mismatch

- Approach 1 - Flatten your data
  - Creates overhead of duplicates
  - Heavy updates
- Approach 2 - Manage relations yourself
  - Index several types of documents
  - Query for one type, and then query for the others
    - Overhead of roundtrips and load on the system (N+1)
- Or...
  - Nested objects & Parent/Child support

Lital Kotlar - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Inner Objects

- But wait... can't inner objects help?
  - Yes, they can... but limited
- Works well for one-to-one mapping
  - e.g. a single address can only be related with a company
  - Represented as single valued inner-object fields
- The challenge lays with one-to-many mapping
  - Can be represented as multi-valued inner-object fields
  - Or parent/child relationship between different objects

Lital Kobler - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Inner Objects

- Suppose we have two companies in the index

```
{
  "name" : "Stark Enterprises",
  "employee" : [
    { "first_name" : "Tony", "last_name" : "Stark" },
    { "first_name" : "Virginia", "last_name" : "Potts" }
  ]
}
```

```
{
  "name" : "NBC Universal",
  "employee" : [
    { "first_name" : "Tony", "last_name" : "Potts" }
  ]
}
```

Lital Kolog - 28-Apr-201

- Now we search for
  - Both documents will be returned, which is not what you might expect. Why is that?

```
first_name:tony AND last_name:potts
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Inner Objects

Original doc

```
{
  "name" : "Stark Enterprises",
  "employee" : [
    { "first_name" : "Tony", "last_name" : "Stark" },
    { "first_name" : "Virginia", "last_name" : "Potts" }
  ]
}
```

Lital Kolog - 28-Apr-2015

Lucene doc

```
{
  "name" : "Stark Enterprises",
  "employee.first_name" : [ "Tony", "Virginia" ],
  "employee.last_name" : [ "Stark", "Potts" ]
}
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Inner Objects - Revisited

- What you really want is
  - Queries will execute on the inner-objects, treating them as distinct object
  - A hit on the inner-object yields a hit on the root-object
- Nested type provide exactly that
  - Each nested object is in fact a special (hidden) Lucene document which is marked as "related" to its root object
  - Querying is executed against these hidden documents and matches resolve the root-object as a hit
    - Execution is still fast, as Lucene indexes nested objects and root-objects in the same "block"
  - Nested documents are only accessible via nested queries

Lita's blog - 28-Apr-2016

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nested Type

Inner objects marked as nested

```
curl -XPUT 'localhost:9200/crunchbase' -d '{
  "mappings" : {
    "company" : {
      "properties" : {
        "milestones" : {
          "type" : "nested"
        },
        "acquisitions" : {
          "type" : "nested",
          "properties" : {
            "description" : { "type" : "string" }
          }
        }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nested Type

Specify the path to the nested object

Score mode

Query requires the full path of the nested fields

```
curl -XGET 'localhost:9200/crunchbase/company/_search'
-d '{
   "query" : {
      "nested" : {
         "path" : "acquisitions",
         "score_mode" : "avg",
         "query" : {
            "match" : {
               "acquisitions.description" : {
                  "query" : "media startup"
               }
            }
         }
      }
   }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nested Query

modes	description
avg	The score average of matching child documents are pushed to the parent.
max	The highest child score is pushed to the parent.
sum	All the scores of the matching children documents are summed and pushed to the parent.

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nested Filter

```
curl -XGET 'localhost:9200/crunchbase/company/_search' -d '{
  "filter" : {
    "nested" : {
      "path" : "acquisitions",
      "filter" : {
        "bool" : {
          "must" : [
            { "range" : { "acquisitions.price_amount" : { "gte" : 1000000 }}},
            { "range" : { "acquisitions.acquired_year" : { "lte" : 2008 }}}
          ]
        }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2016

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nested Aggregations

- It is possible execute aggregations on nested objects
  - In fact, in some scenarios, aggregations without nested objects don't make much sense

```
{
  ...
  "aggs" : {
    "acquisitions" : {
      "nested" : { "path" : "acquisitions" },
      "aggs" : {
        "annual" : {
          "terms" : { "field" : "acquisitions.acquired_year" },
          "aggs" : {
            "total_cost" : {
              "sum" : { "field" : "acquisitions.price_amount" }
            }
          }
        }
      }
    }
  }
}
```

Litan Polog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Parent / Child

- Nested objects are great for some scenarios
  - The "same block" indexing makes it very fast and memory efficient to execute
- But it also introduces a limitation
  - A a nested object update requires a complete reindex of its root object (and all other nested objects)
- It is sometimes required to have a complete separation between the two related object
  - So updating one will not affect the other
- This can be done with parent/child relationship

tal Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Parent / Child

- Enables you to associate one type of document with another
  - While the two are essentially completely separate and standard documents in the index
- The *_parent* settings in the mapping creates this association
  - Specifies the type of the parent document

A company document is a parent of an employee document

```
curl -XPUT 'localhost:9200/crunchbase' -d '{
   "mappings" : {
      "employee" : {
         "_parent" : { "type" : "company" }
      }
   }
}'
```

Michal Kolosy 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Parent / Child

- When indexing a child document, the parent _id must be specified to create the association
- The parent id also serves as the routing value for the child document
  - The parent and child documents are routed to the same shard so joining them when searching will be an in memory join

Lital Kolog - 28-Apr-2015

```
curl -XPOST 'localhost:9200/crunchbase/employee?parent=amazon' -d '{
  "first_name" : "Jeff",
  "last_name" : "Bezos"
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

has_child query & filter

- Enables querying/filtering parent documents based on whether their children match a query

```
curl -XGET 'localhost:9200/crunchbase/company/_search' -d '{
   "query" : {
      "has_child" : {
         "type" : "employee",
         "score_mode" : "avg",
        "query" : {
            "match" : {
               "last_name" : {
                  "query" : "gates"
               }
            }
         }
      }
   }
}'
```

Lital Kolog - 28-Ap 2015

- *score_mode* can be one of the following:
  - *avg*, *max*, *sum*, or *none*

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

has_parent query & filter

- Enables querying/filtering child documents based on whether their parent match a query

```
curl -XGET 'localhost:9200/crunchbase/employee/_search' -d '{
   "query" : {
       "has_parent" : {
          "type" : "company",
          "score_mode" : "score",
         "query" : {
             "match" : {
                "name" : {
                   "query" : "microsoft"
                }
             }
          }
       }
   }
}'
```

Lital Kolog - 28-Apr-2015

- score_mode can be either “score” or “none”

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nested vs. Parent / Child

- Nested type
  - Fast and memory efficient, but not flexible
  - A change requires re-indexing the root document

- Parent / Child
  - Reasonably fast, but require more memory
    - Document ids need to be loaded into memory.
  - Flexible - parent and child documents can be updated independently.

Lital Kobig - 28-Dec-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Geo Location

Lital Kolog - 28-Apr-2015

elastic

The Need for Geo

- Geo related features are becoming more and more important in today apps
  - Think Yelp, Foursquare, Uber, Google Maps, etc…
- Being able to combine free text search with geo location is the crucial next step for those applications
  - Find all restaurants that serve french food within 20 kilometers from my current location

Lital Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Mapping Geo Points

- Mappings need to be used to mark which part(s) in the document represent geo points
  - Note, needs to be set before introducing the document

```
curl -XPUT 'localhost:9200/restaurant/desc/_mapping' -d '{
  "desc" : {
    "properties" : {
      "location" : {
        "type" : "geo_point"
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Indexing Geo Points

As an object
(the “safest”format)

```
curl -XPUT 'localhost:9200/restaurant/desc/1' -d '{
  "desc" : {
    "location" : { "lat" : 41.12, "lon" : -71.34 }
  }
}'
```

As a *“lat,lon”* string
(the de facto convention)

```
curl -XPUT 'localhost:9200/restaurant/desc/1' -d '{
  "desc" : {
    "location" : "41.12,-71.34"
  }
}'
```

Lita Kolog 13 Apr 2015

As a *[lon, lat] GeoJSON* array
(the “official” standard)

```
curl -XPUT 'localhost:9200/restaurant/desc/1' -d '{
  "desc" : {
    "location" : [-71.34, 41.12]
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Searching Using Geo Points

- *geo_bbox* filter
  - A bounding box filter matching on documents that only exists within the bounding box

```
curl localhost:9200/restaurant/desc/_search -d '{
  "query" : {
    "filtered" : {
      "query" : { "match_all" : {} },
      "filter" : {
        "geo_bbox" : {
          "location" : {
            "top_left" : { "lat" : 40.73, "lon" : -74.1 },
            "bottom_right" : { "lat" : 40.717, "lon" : -73.99 }
          }
        }
      }
    }
  }
}'
```

Little Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Searching Using Geo Points

- *geo_distance* filter
  - Match on documents that are within a distance from a specific geo point

```
curl localhost:9200/restaurant/desc/_search -d '{
  "query" : {
    "filtered" : {
      "query" : { "match_all" : {} },
      "filter" : {
        "geo_distance" : {
          "distance" : "400km"
          "location" : { "lat" : 40.73, "lon" : -74.1 }
        }
      }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Searching Using Geo Points

- *geo_distance* filter
  - Computing distance is a relatively heavy computation
    - Easily scalable as we add more shards / nodes
- *optimize_bbox*
  - By default, a bounding box is built "around" the circle the distance forms which enables us to first filter all those documents that don't "fall" within the box, thus minimize the number of distance computations
- *distance_type* - the nature of the distance computation
    - *plane* - "flattens" the world and computes simple geometric distance
    - *arc* - computes the distance considering the spherical shape of earth
    - *sloppy_arc* - like *arc*, but cheaper at the cost of accuracy (near poles)

Lital Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Searching Using Geo Points

- *geo_distance_range* filter
  - Similar to geo_distance with the ability to customize the from and to distances

```
curl localhost:9200/restaurant/desc/_search -d '{
  "query" : {
    "filtered" : {
      "query" : { "match_all" : {} },
      "filter" : {
        "geo_distance_range" : {
          "from" : "10km",
          "to" : "50km"
          "location" : { "lat" : 40.73, "lon" : -74.1 }
        }
      }
    }
  }
}'
```

Lilia Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Geo Point Aggregations

- *geo_distance*
  - A bucketing aggregation that buckets documents within the same distance range from a given point of reference

```
curl 'localhost:9200/restaurant/desc/_search' -d '{
    "aggs" : {
        "sf_rings" : {
            "geo_distance" : {
                "field" : "location",
                "origin": "37.75, -122.39",
                "unit" : "meters",
                "ranges" : [
                    { "to" : 4000},
                    { "from" : 4000, "to" : 5000},
                    { "from" : 5000}
                ]
            }
        }
    }
}'
```

Litanyology - 28 Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Geo Point Aggregations

- *geo_distance*
  - The result contains a bucket per range which is associated with the document count in the bucket.
  - As all bucketing aggregations, one can further nest other aggregations under it.

```
"aggregations": {
      "sf_rings": [
         {
            "key": "*-4000.0",
            "from": 0,
            "to": 4000,
            "doc_count": 52
         },
         {
            "key": "4000.0-5000.0",
            "from": 4000,
            "to": 5000,
            "doc_count": 1
         },
         {
            "key": "5000.0-*",
            "from": 5000,
            "doc_count": 287
         }
      ]
   }
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Geo Point Implementation

- How is geo_point implemented?
  - All the values for the geo points are loaded to memory (*lat* & *lon* values) in a compressed manner
  - Allows for fast distance calculations as documents matched are traversed
- Important Optimizations
  - Don't use geo filters in a *bool* filter, use *and*/*or*/*not* instead
  - When using *and*/*or* prefer to place it as the last filter

Lital Kobel 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Geo Shapes

- A geo shape can be a:
  - Envelope (box)
  - Lines
  - Polygon, Multi-Polygons, Polygon with holes
  - Point
- Documents can be associated with geo-shapes
- It is possible to filter based on geo-shapes
  - geo-shape filter is associated with a geo-shape, and matches only those documents who's associated shapes intersect with the filter one
- Support for intersect, contains & disjoint relations

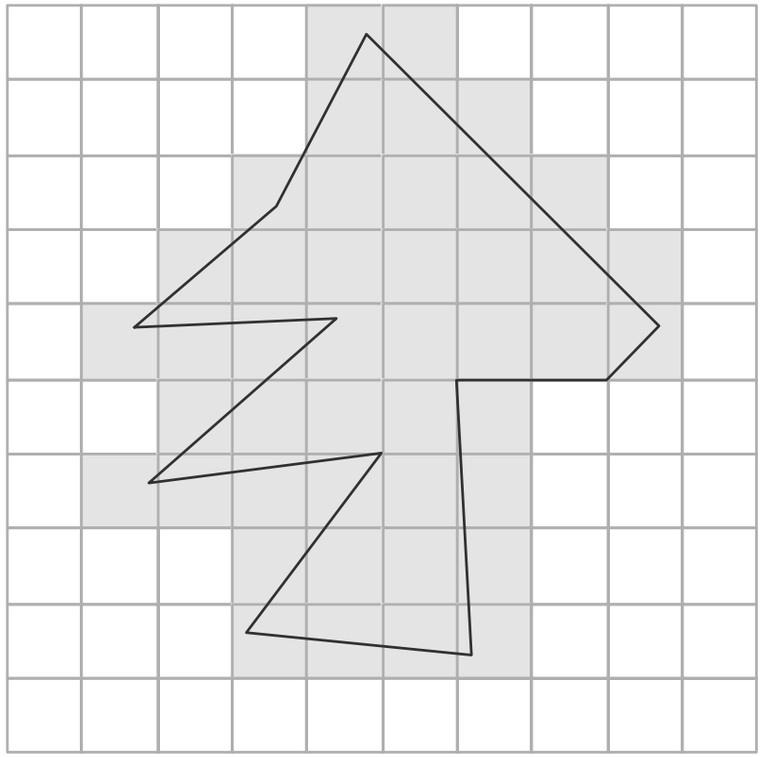
Lital Kesing - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Geo Shape Internal Representation

- Challenge
  - Find a way to translate a shape to terms which can be indexed and queried

- Solution
  - Divide the world into regions, each encoded with unique (textual) code
  - Represent shapes by the regions they "touch"
  - The finer (more) regions you have, the more accurate the shape is represented



Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

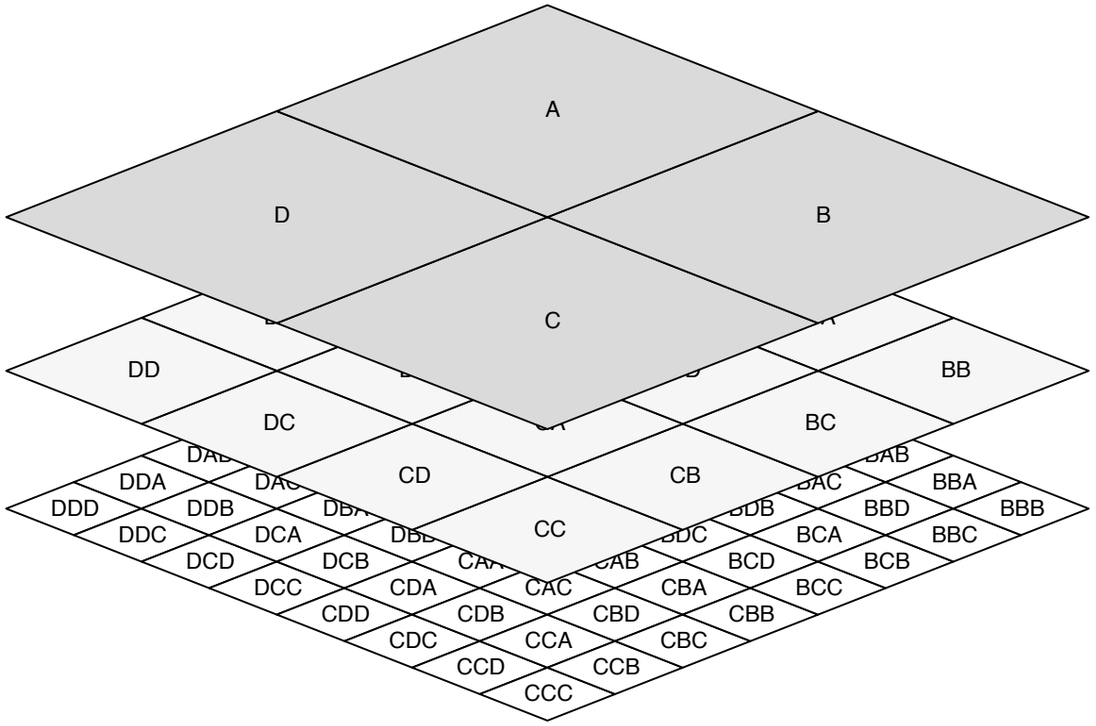
Geo Shape Internal Representation

- There are two ways to create such encoding
  - *geohashes*
  - *quad trees*

- Both are similar

  Mainly differing by the encoding strategy

- Quad trees
  - The more levels in the tree, the finer the regions & the higher the accuracy



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Mapping Geo Shapes

- Supported by the *geo_shape* types
  - Define the encoding
  - Precision is defined in distance (internally translated to tree levels)

```
curl -XPUT 'localhost:9200/amsterdam' -d '{
  "mappings" : {
    "region" : {
      "properties" : {
        "area" : {
          "type" : "geo_shape",
          "tree" : "quadtree",
          "precision" : "1km",
        }
      }
    }
  }
}'
```

Terms will be generated by a 16 levels quad tree

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Indexing Geo Shapes

```
curl -XPUT 'localhost:9200/europe/poi/1' -d '{
    "city" : "Amsterdam",
    "name" : "Dam Square",
    "area" : {
        "type" : "polygon",
        "coordinates" : [[
            [52.37356, 4.89218],
            [52.37276, 4.89205],
            [52.37274, 4.89301],
            [52.37250, 4.89392],
            [52.37287, 4.89431],
            [52.37346, 4.89331],
            [52.37326, 4.89305],
            [52.37356, 4.89218]
        ]]
    }
}'
```

Start & end points must be identical

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Searching Using Geo Shapes

```
curl -XGET 'localhost:9200/europe/poi/_search' -d '{
    "filter": {
        "geo_shape": {
            "area": {
                "shape": {
                    "type": "polygon",
                    "coordinates" : [[
                        [52.3861, 4.883],
                        [52.3716, 4.873],
                        [52.3613, 4.883],
                        [52.3582, 4.898],
                        [52.3622, 4.921],
                        [52.3668, 4.925],
                        [52.3861, 4.883]
                    ]]
                }
            }
        }
    }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Searching Using Geo Shapes

```
curl -XPUT 'localhost:9200/european-regions/city/amsterdam-center' -d '{
    "shape" : {
        "type": "polygon",
        "coordinates" : [[
            [52.3861, 4.883], [52.3716, 4.873],  [52.3613, 4.883],
            [52.3582, 4.898], [52.3622, 4.921], [52.3668, 4.925], [52.3861, 4.883]
        ]]
    }
}'
```

Indexing shapes

```
curl -XGET 'localhost:9200/poi/_search' -d '{
    "filter": {
        "geo_shape": {
            "area": {
                "indexed_shape": {
                    "id": "amsterdam-center",
                    "type": "city",
                    "index": "european-regions"
                }
            }
        }
    }
}'
```

Referring to the indexed shapes

Lars Kerg - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

.percolator

Lital Kolog - 28-Apr-2015

elastic

Introduction to Percolator

- Percolator tells what queries match a given document
  - While search tells what documents match a given query
  - (A.k.a "reversed search")
- Percolator has been around for a long time.
  - Since version 0.15.0
- Percolator has been redesigned in version 1.0
  - Scale out with the number of queries
  - Additional features were added as well

Lital Kosig - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Introduction to Percolator

- Percolator queries can be registered in elasticsearch
  - The *QueryDSL* is just a json document, therefore it can be index as any other json document. Internally these queries are indexed as an internal (reserved) *.percolator* document type
- The *_percolate* API enables sending documents, which will then be matched against the registered queries.
  - The response will return the IDs of all queries that matched
- This API is mostly realtime
  - As soon as a query is registered, it immediately participates in the percolation process
    - Some features are near realtime

Lisa Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Registering a query

- Documents & Fields
- A document is modeled as a JSON object

Any index can contain percolator queries

The reserved percolator type

The *id* of the query

```
$ curl -XPUT 'localhost:9200/news/.percolator/breaking-news' -d '{
    "query" : {
      "match" : {
         "level" : "breaking"
      }
    }
}'
```

Lital Kolog - 28-Apr-2015

The actual query that gets parsed and loaded into memory

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

_percolate API

- Documents & Fields
- A document is modeled as a JSON object

The index containing registered percolator queries

The type of the document being percolated. The mapping is used to parse the document.

Percolate endpoint

```
$ curl -XGET “localhost:9200/news/item/_percolate" -d '{
    "doc" : {
      "title" : "Stark Industries caught up in waste scandal",
      "message" : "Reporters have found evidence that link Stark …",
      "level" : "breaking"
   }
}'
```

The document that is going to be percolated

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

_percolate API - Response

- Each match always includes the index and id of the matching query.
  - Also additional features may include extra information per match

```
{
   "took" : 19,
   "_shards" : {
      "total" : 1,
      "successful" : 1,
      "failed" : 0
   },
   "count" : 1,
   "matches" : [
      {
         "_index" : "news",
         "_id" : “breaking-news"
      }
   ]
}
```

Response header

Total number of matching queries

A query match

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolator - Under The Hood

- Each shard has its own set of percolation queries
  - When percolation queries are indexed and registered, the top level *query* part is parsed into a native Lucene query and is kept in memory
- A node that received the percolate request scatters the request to all the shards (depends on the index/ices and/or routing)
- Percolation execution on the shard:
  - When percolating a document, that document is parsed and indexed into an dedicated in-memory Lucene index
  - All percolator queries are executed one-by-one on this in-memory index and if the query matches its *id* is collected

Lindy Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Routing

- Routing enables to limit the number of shards the *_percolate* request is executed on
  - A single routing value will execute the percolate request only on a single shard and that will reduce the percolation time and load on the cluster

Index time

```
curl -XPUT “localhost:9200/news/.percolator/breaking-news?routing=stark" -d '{
    "query" : {
       ...
   }
}'
```

Lital Kolog - 28-Apr-2015

Search time

```
curl -XGET "localhost:9200/news/item/_percolate?routing=stark" -d '{
    "doc" : {
       ...
   }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Filtering

- Filtering enables including/excluding specific queries based on associated metadata
  - Reduces the number of queries being evaluated and can be useful to for application logic filtering.
    - Remember, filters are cached (same as in other APIs).

Lital Kolog 28-Apr-2015

```
$ curl -XPUT 'localhost:9200/news/.percolator/breaking-news' -d '{
    "query" : {
       ...
   },
   "company" : "stark"
}'
```

Associating metadata with the registered query

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Filtering

The document will only be percolated through the queries that match this filter

```
curl -XGET 'localhost:9200/news/item/_percolate' -d '{
    "doc" : {
      ...
   },
   "filter" : {
      "term" : {
         "company" : “stark"
      }
   }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolating Existing Documents

- Percolate a document that has already been indexed
  - by specifying its *id* and optionally *type* and *index*.
- Internally a *get* fetches the specified document and that document will then be percolated

LitaLKorea - 28-Apr-2015

```
curl -XGET 'localhost:9200/news/item/_percolate/1'
```

The same index used for both fetching the document and percolation

The id of the document that will be percolated

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolating Existing Document

name	description
percolate_index	The index to percolate the fetched document in
percolate_type	The type used for percolating
percolate_routing	Percolate routing
percolate_preference	Allows to select certain shards (like *_primary* or *_local*)
version	The version the fetched document must have for in order to percolate

Lital's blog - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Scoring & Sorting

- The *_percolator* also supports a query option in the body
  - Does the same as *filter*, but also is able to compute a score that is based on how well the percolation query matches

```
curl -XGET 'localhost:9200/news/item/_percolate' -d '{
    "doc" : {
       ...
    },
    "query" : {
       "term" : {
          "company" : "stark"
       }
    }
}'
```

Lital Kozlovski - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Scoring & Sorting

- The score will only be computed if *sort* or *track_scores* are specified
  - At the moment only sorting by score is supported.

```
curl -XGET 'localhost:9200/news/item/_percolate' -d '{
    "doc" : {
       ...
    },
    "query" : {
       ...
    },
    "sort" : [
       "_score"
     ],
    "track_scores" : true

}'
```

Required for scoring

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Scoring & Sorting

- The score will only be computed if *sort* or *track_scores* are specified

  At the moment only sorting by score is supported.

```
curl -XGET "localhost:9200/news/item/_percolate" -d '{
    "doc" : {
       ...
   },
   "query" : {
        "function_score" : {
            "query" : { "match_all": {}},
            "functions" : [
                {
                    "exp" : {
                        "create_date" : {
                            "reference" : "2014/01/01",
                            "scale" : "1000d"
                        }
                    }
                }
            ]
        }
    },
   "size" : 10,
   "track_scores" : true,
   "sort" : [
       "_score"
   ]
}'
```

Using *function_score* query with exponential function to sort the matches by creation date

If *sort* is specified then *size* is a required option

Lital Kobo 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Scoring & Sorting

Example

```
{
   "took": 2,
   "_shards": {
      ...
   },
   "total": 2,
   "matches": [
         {
         "_index": "news",
         "_id": “breaking-news",
         "_score": 0.4002574
      },
      ...
   ]
}
```

The score is added to the query match.

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Highlighting

- Highlights what part of the document a matched query matches with
- So if we index the following query:

```
curl -XPUT 'localhost:9200/news/.percolator/brown-foxes' -d '{
    "query" : {
      "match" : {
         "body" : "brown fox"
      }
   }
}'
```

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Highlighting

- And execute the following percolate request:

```
curl -XGET “localhost:9200/news/item/_percolate” -d ‘{
   "doc" : {
         "body" : "The quick brown fox jumps over the lazy dog"
    },
    "highlight" : {
        "fields" : {
            "body" : {}
        }
    },
    "size" : 5
}'
```

Same as in the search api

size is a required when highlighting

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Highlighting

- Then we get the following response:

```
{
   "took": 18,
   "_shards": {
      "total": 5,
      "successful": 5,
      "failed": 0
   },
   "total": 1,
   "matches": [
      {
         "_index": "news",
         "_id": “brown-foxes”,
         "highlight": {
            "body": [
               "The quick <em>brown</em> <em>fox</em> jumps over the lazy dog"
            ]
         }
      }
   ]
}
```

Lital Kolog - 28-Apr-2015

Highlight snippets are included in the match

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multi _percolate API

- Combines multiple percolation requests in a single request
  - Similar to how *_bulk*, *_mget* and *_msearch* APIs work

```
curl -XGET 'localhost:9200/twitter/tweet/_mpercolate' --data-binary @requests.txt;
```

Request item header

New line is required

```
{"percolate" : {"index" : “twitter", "type" : “tweet“}}\n
{"doc" : {"message" : "some text”}}\n
{"percolate" : "index" : "twitter", "type" : "tweet"}\n
{"doc" : {"message" : “some other text”}}\n
{"percolate" : "index" : "twitter", "type" : "tweet", "id" : “1"}\n
{}\n
```

Percolating an existing document

itaf.olog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multi percolate api

The response items have the same order as the request items have been specified

```
{
    "items" : [
        {
            "took" : 24,
            "_shards" : {
                ...
            },
            "total" : 3,
            "matches" : [ ... ]
        },
        {
            "took" : 12,
            "_shards" : {
                ...
            },
            "total" : 3,
            "matches" : [ … ]
        },
        {
            "error" : "[user][3]document missing"
        }
    ]
}
```

The response for percolating existing document, but the document didn't exist

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation - Aggregations

- Aggregations can also be used in the *_percolate* api
  - Applied on the percolation queries metadata

```
curl -XGET ‘localhost:9200/news/item/_percolate' -d '{
   "doc" : {
         ...
    },
   "aggs" : {
      “saved_at_monthly_trend" : {
            "date_histogram" : {
                "field" : “saved_at",
                "interval" : "month"
            }
        }
   }
}'
```

Lilian Kolosov 28 Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Percolation Storing Strategies

- Let the queries and the data co-exist in the same index
  - Percolation queries aren't included by default in search result
    - Only if specifically searching in *.percolator* type
  - A viable solution when the number of registered queries is small
- Dedicated percolator index
  - A better solution when there are many registered queries
  - The dedicated percolator index can have different sharding options than the index containing the data
  - Consider putting percolator indices on different nodes than those indices containing the data
    - We'll cover how this can be done when we discuss *allocation filtering* in the *distributed model* section

Lital Vashlog - 28-Apr-2016

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Distributed Model

Lital Kolog - 28-Apr-2015

elastic

Finding Other Nodes

- The process of "pinging" to find other nodes in a cluster
- Two types:
  - *multicast*: Uses multicast to publish a message on a multicast address to find other nodes
  - *unicast*: A list of "seed" nodes to use in order to discover other nodes in the cluster

Lital Kolog - 28-Apr-2015

```
# disable multicast
discovery.zen.ping.multicast.enabled: false

# set the list of hosts for unicast discovery
discovery.zen.ping.unicast.hosts: ["host1", "host2", "host3"]

# how long to wait for responses to a ping request (default 3s)
discovery.zen.ping.timeout: 5s
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Elected Master Node

- Each cluster has an elected master node
- Needed to handle cluster wide events
  - Nodes joining and leaving
  - Cluster meta changes
    - New indices created and deleted
    - Mappings being updated
  - (Re)allocation of shards on nodes

Lital Kozlo - 28-Apr-2015

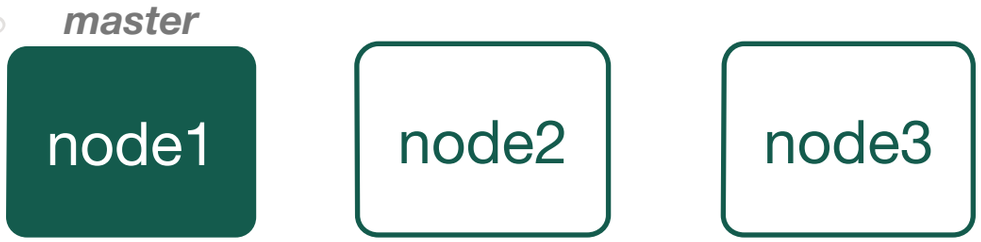
Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Master Election

time

node1 starts, elects itself as the master

node2 & *node3* join the cluster



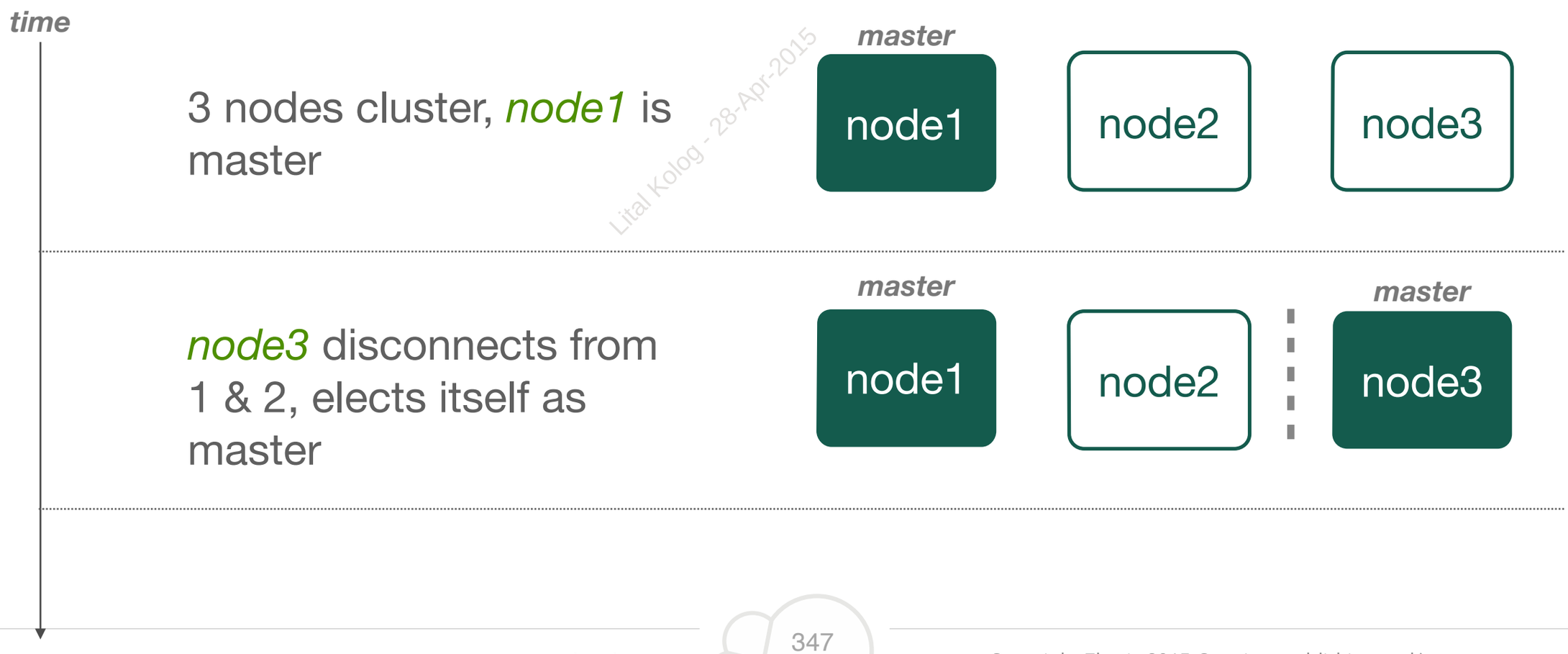
Lital Kolog - 28-Apr-2015

node1 fails, *node3* elected as master

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Minimum Master Nodes

- A cluster can potentially split in two
- Resulting in two sub clusters
  - Bad! resolving it requires explicitly killing one of clusters



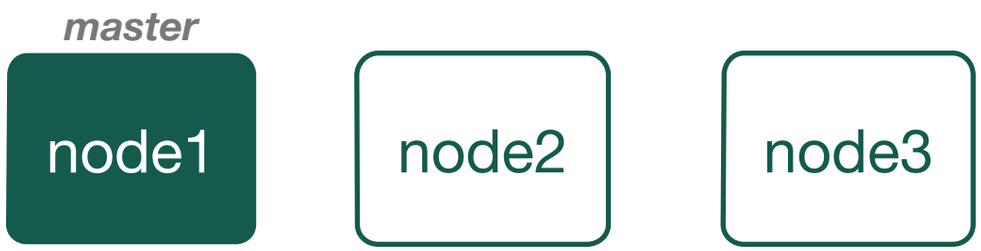
Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

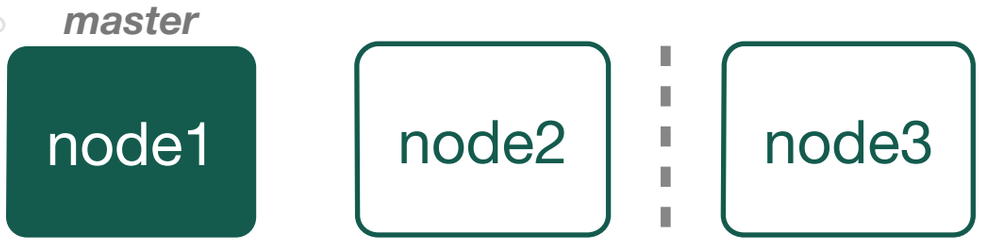
Minimum Master Nodes

time

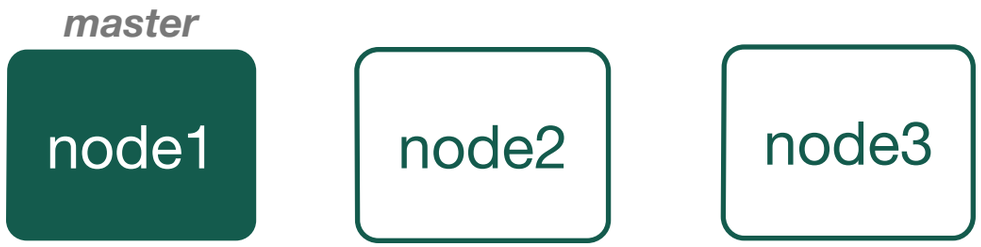
3 nodes cluster,
minimum_master_nodes
is set to 2



node3 disconnects, with not enough master nodes, goes back to trying to form a cluster



Disconnection resolved, *node3* joins back to the cluster

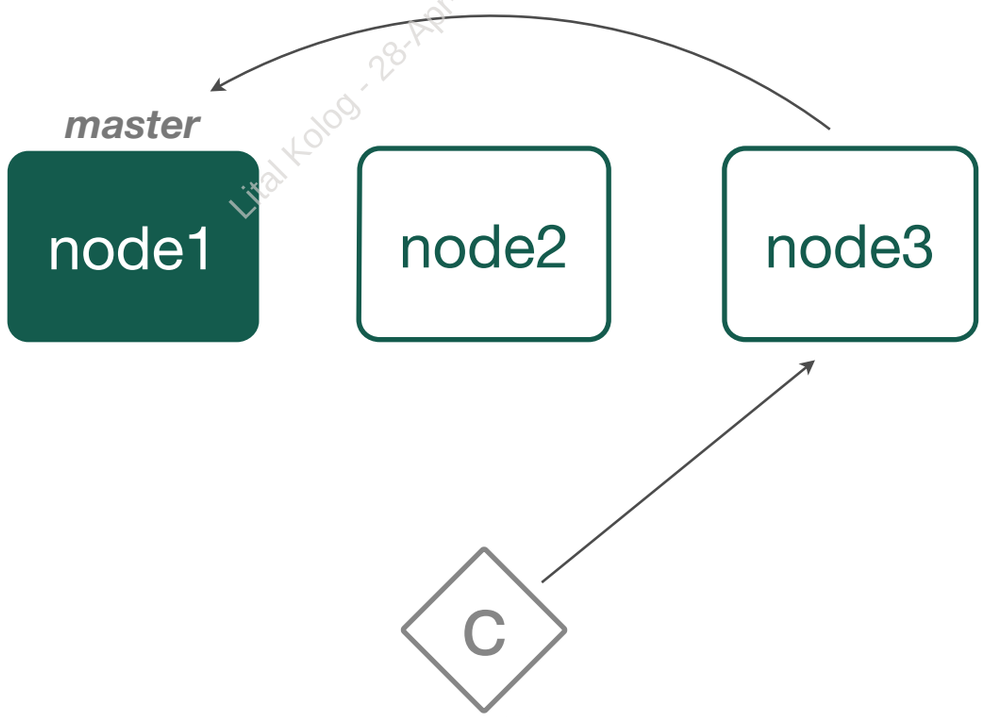


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Cluster Meta APIs

- All cluster meta APIs are executed on the master node to ensure consistency
  - For example, fail a create index API if the index already exists
- All nodes can automatically reroute APIs to the master node



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

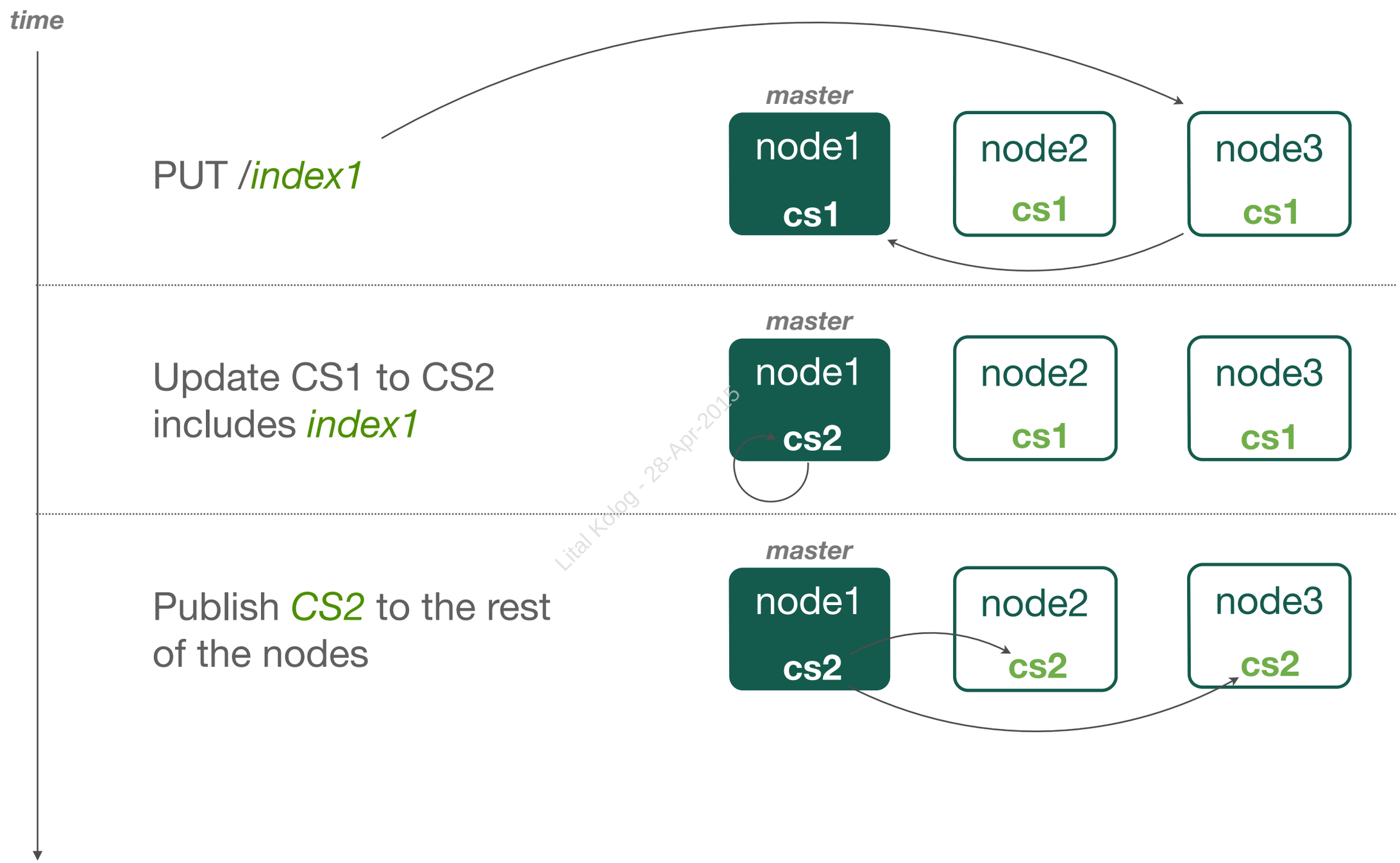
Cluster State

- The global meta state of the cluster, including:
  - Nodes that are currently part of the cluster
  - The indices, with their respective settings and mappings
  - Shards allocation table (which shard exists on which node)
  - Shards state (unassigned, initializing, started, relocating)
- The elected master node is the only node that changes the cluster state
  - Once changed, it gets published to the rest of the cluster

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Cluster State



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Cluster State API

- Returns the cluster state with all the relevant information
  - E.g. metadata, routing table (a lot of data potentially, can be filtered)
- Two main modes:
  - *local* flag set to *false* (default), redirect the request to the master node, and returns its state
  - *local* flag set to *true*, return the cluster state that the node being "hit" holds

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation

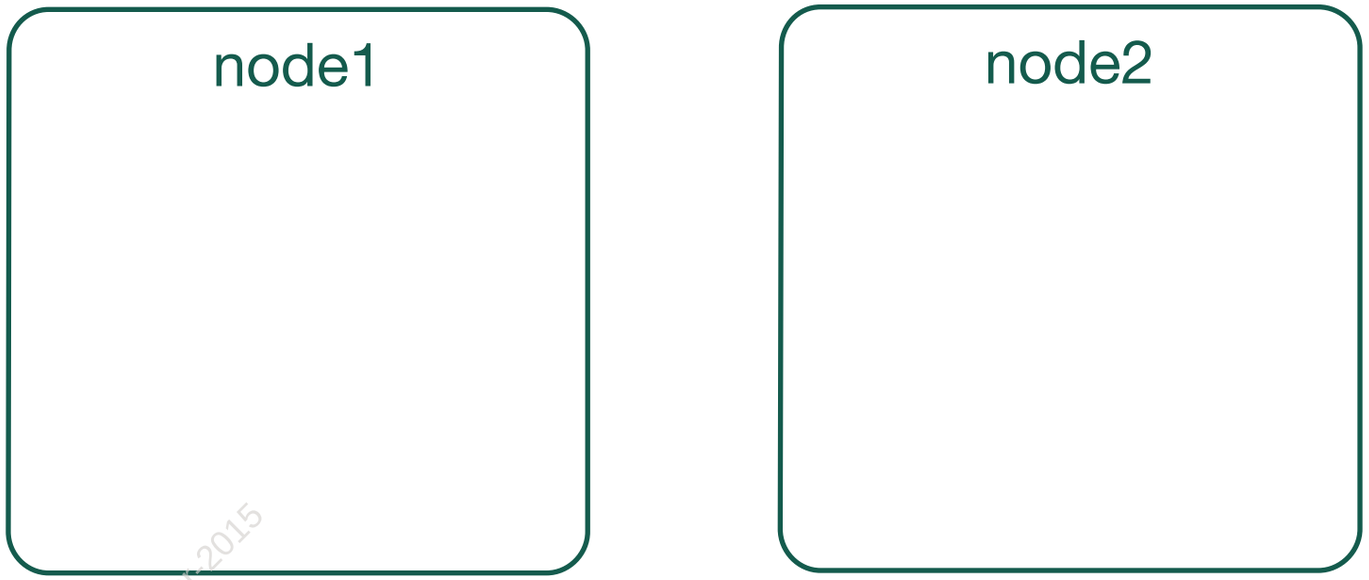
- The process of allocating shards to nodes
- Shards go through several states
  - *UNASSIGNED*, *INITIALIZING*, *STARTED*, *RELOCATING*
- Shards allocation is stored in the cluster state
  - Under the *routing_table*
    - Viewed from the "indices" perspective
  - Can also be represented in *routing_nodes*
    - Viewed from "nodes" and whats allocated on them
    - Built automatically out of *routing_table*
- Nodes act on their relevant allocation in the state

Lital Kotlar - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Create Index

- PUT *index1*
- 2 shards, each with 1 replica (2 copies of the data)
- Total of 4 shards
- All shards in *UNASSIGNED* state

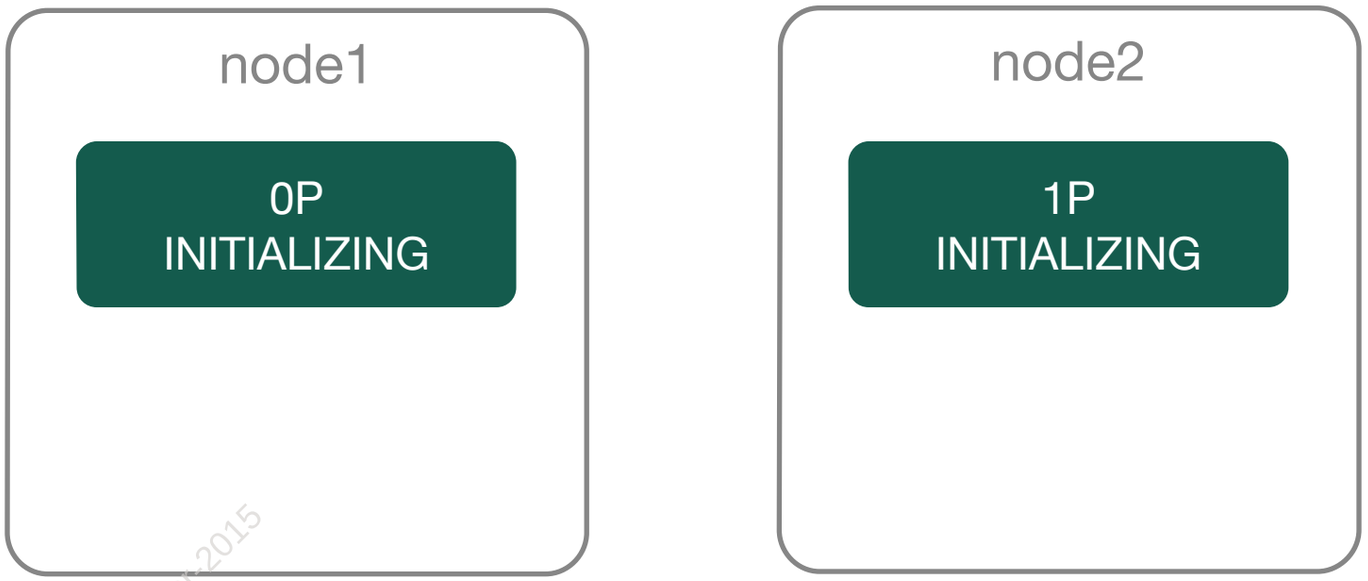


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Create Index

- All the primaries are allocated first
- They are all in *INITIALIZING* state

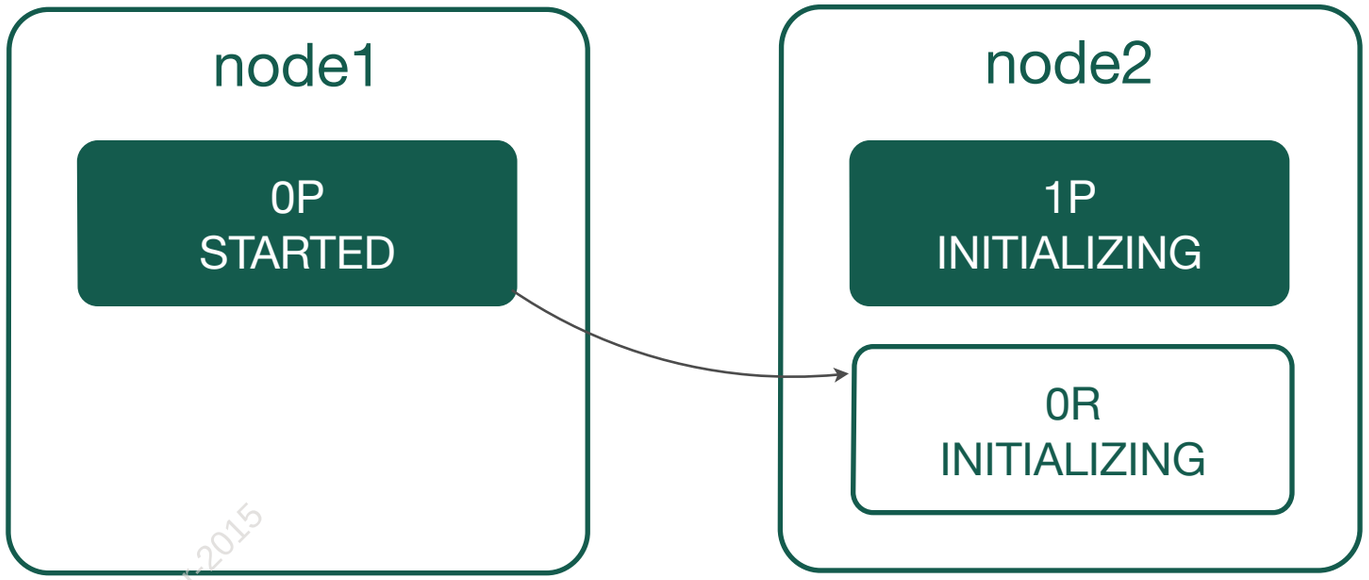


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Create Index

- *0P* done *INITIALIZING*, moves to *STARTED*
- 0R can now be allocated, in INITIALIZING state, recovering from *0P*
- Recovery can be done while operations are executed (*index*, delete, search, ...)

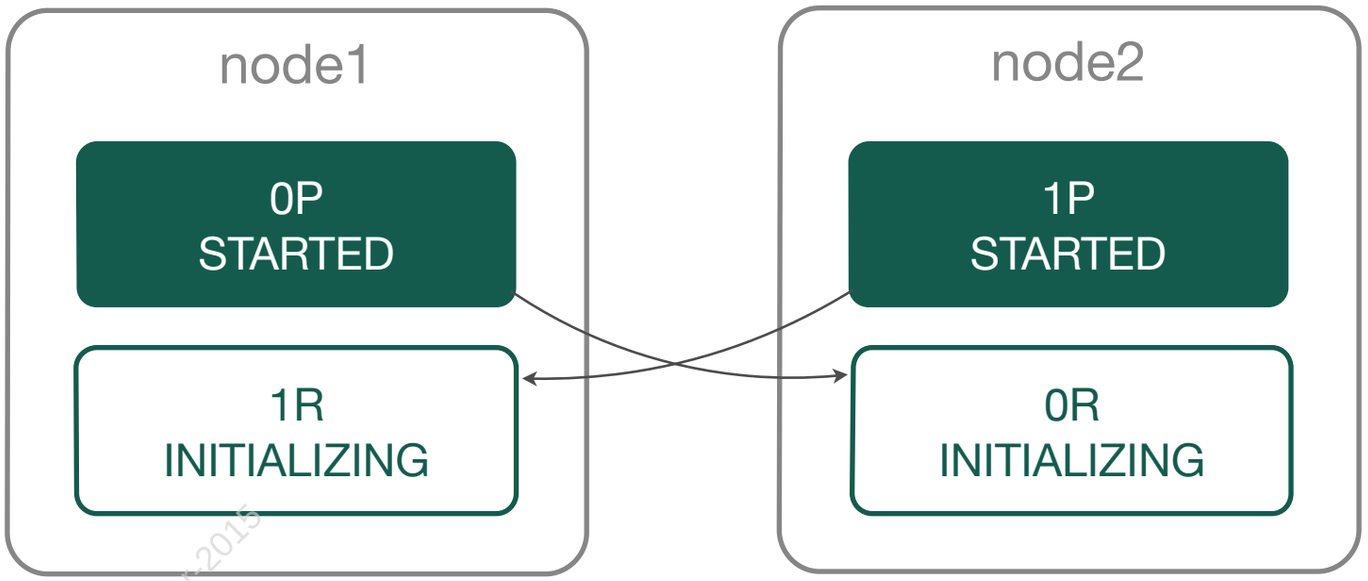


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Create Index

- *1P* done *INITIALIZING*, moves to *STARTED*
- *1R* can now be allocated in *INITIALIZING* state, recovering from *1P*

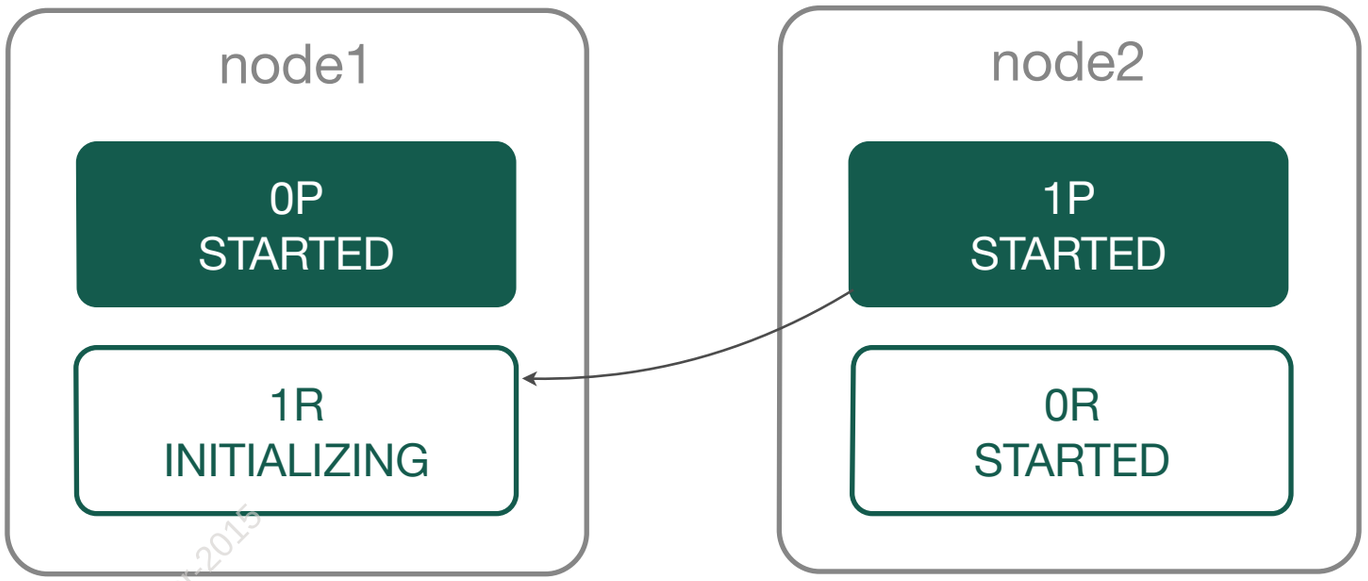


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Create Index

- *0R* done *INITIALIZING* from *0P*, moves to *STARTED*

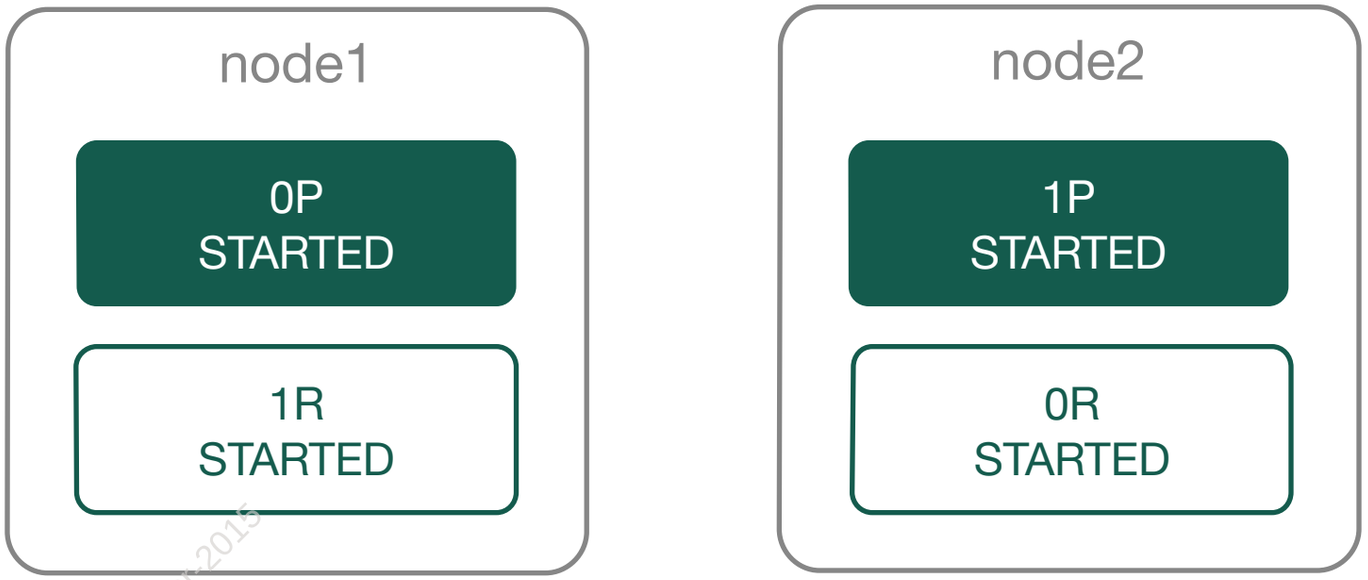


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Create Index

- *1R* done *INITIALIZING* from *1P*, moves to *STARTED*



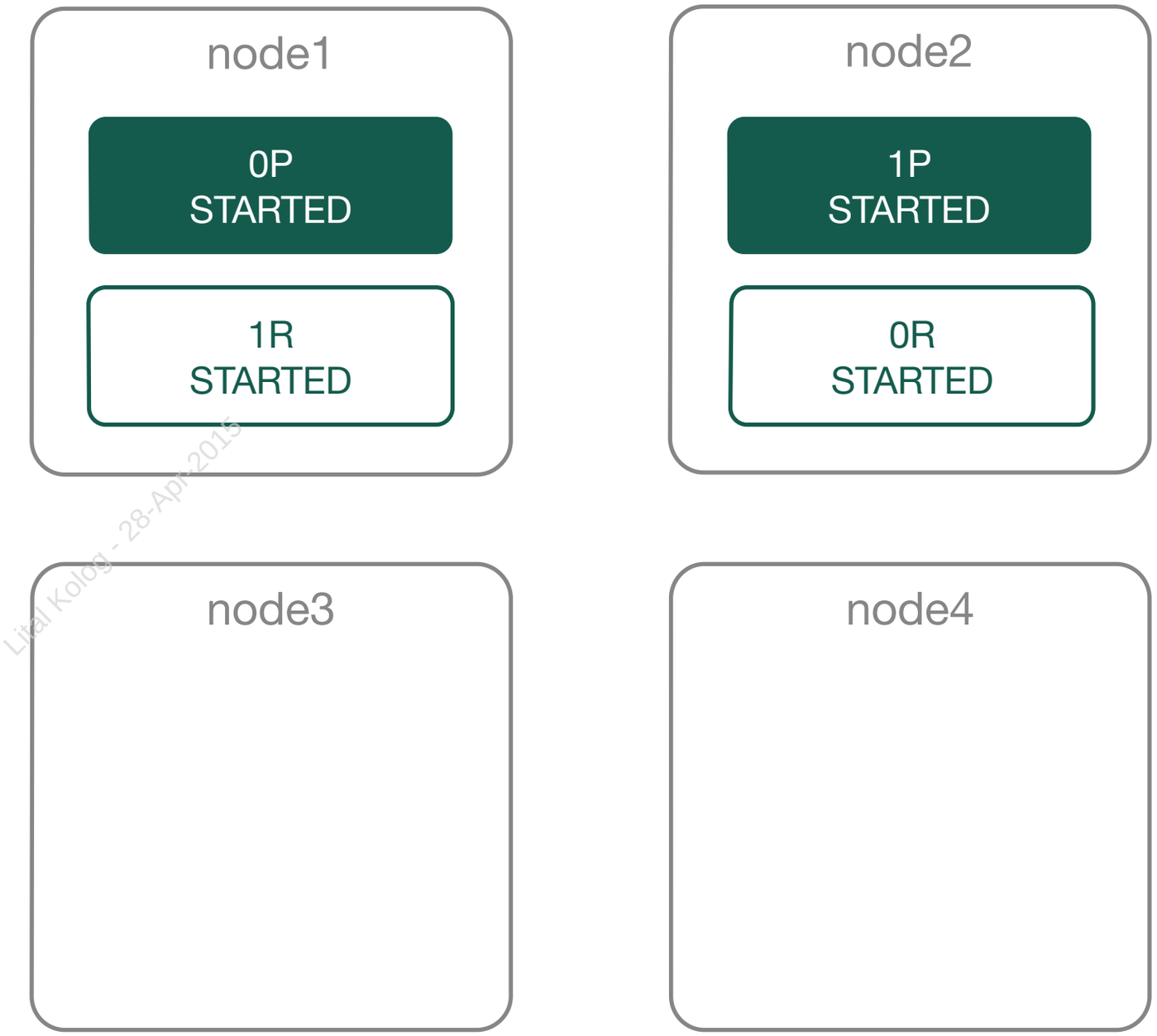
Lital Kolog - 28-Apr-2015

The index is now fully allocated and distributed on the cluster (what we like to refer to as a *green* state)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Adding Nodes

- Having *node3* and *node4* join the cluster

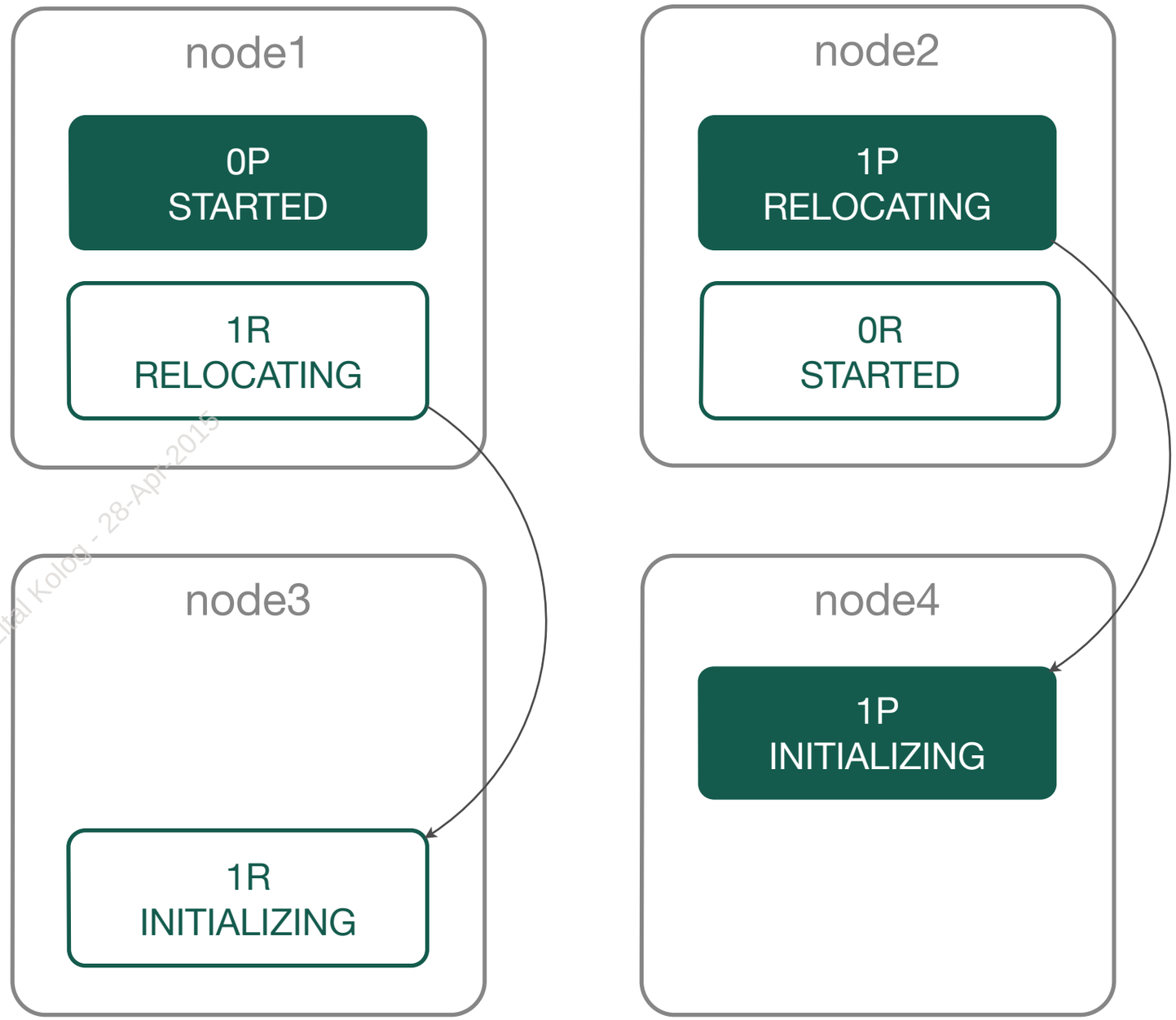


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Adding Nodes

- *1R* shard relocates from *node1* to *node3*
- *1P* shard relocates from *node2* to *node4*
- Relocation moves original shard state to *RELOCATING*
- Creates a “duplicate” target shard that is in *INITIALIZING* state
- Relocation can be done while operations execute on the shard (index, delete, search, ...)

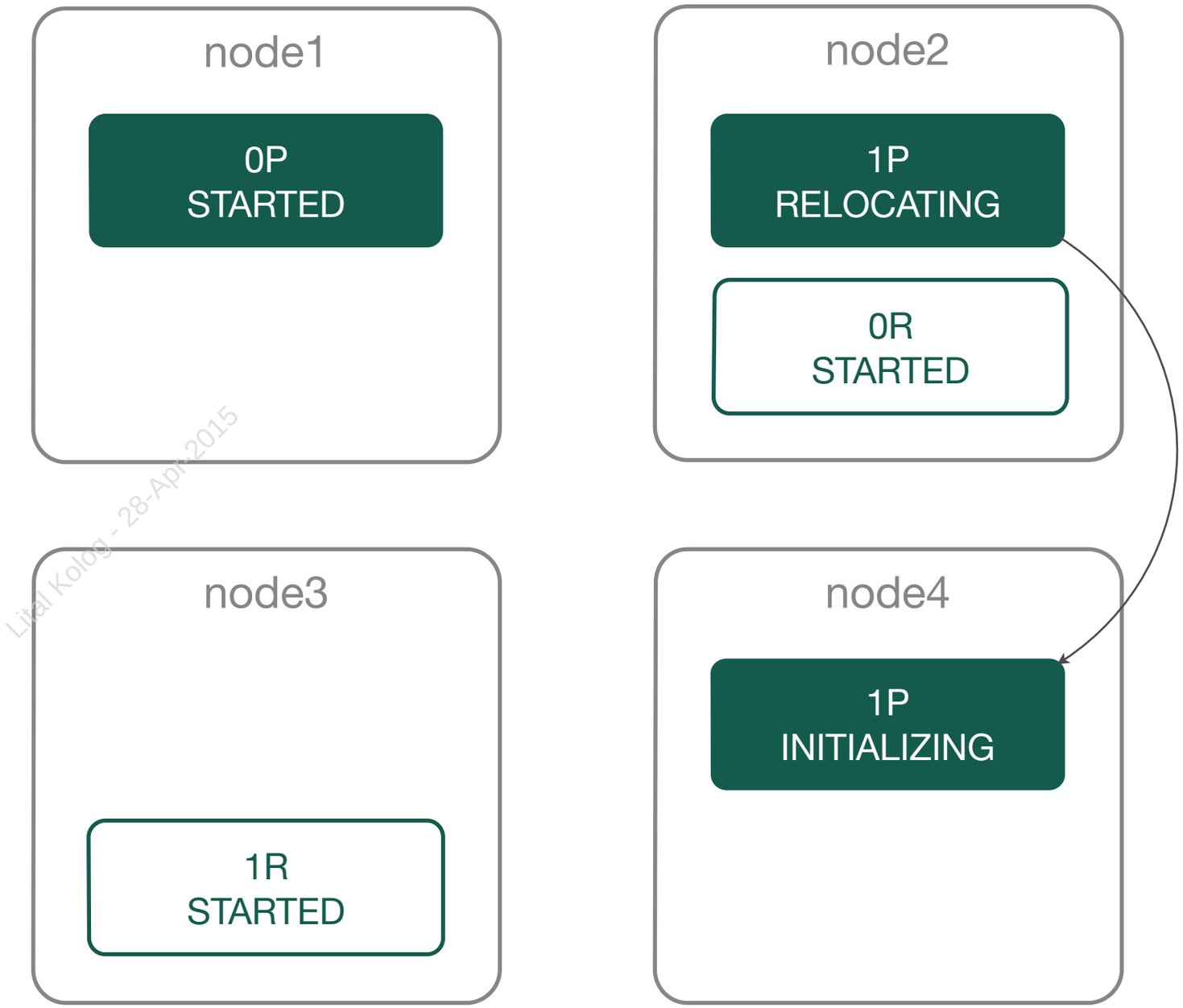


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Adding Nodes

- *1R* relocation is done
- Marking *1R* on *node3* as *STARTED*
- Removing *1R* from *node1*

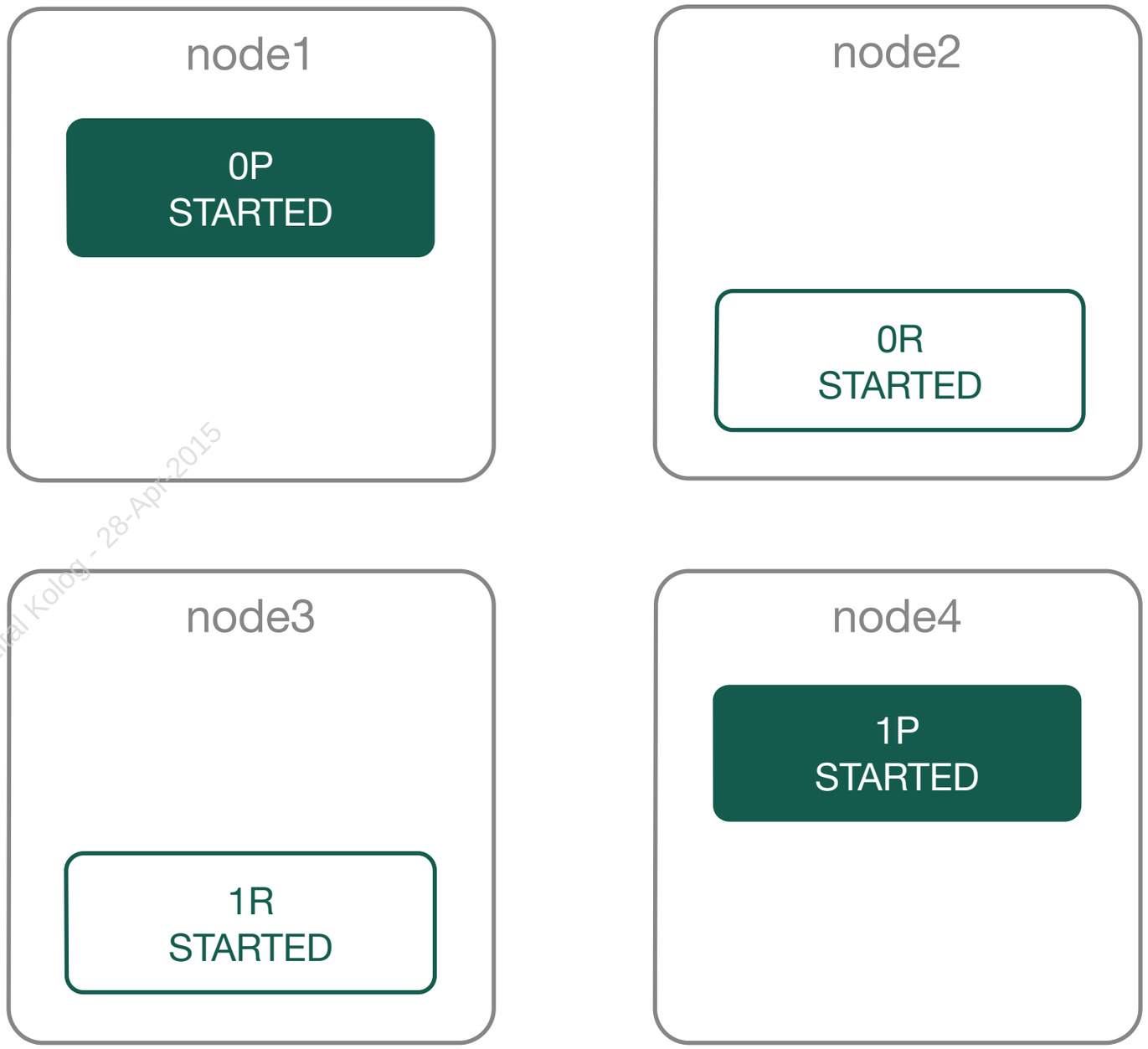


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Adding Nodes

- *1P* relocation is done
- Marking *1P* on *node4* as *STARTED*
- Removing *1P* on *node2*

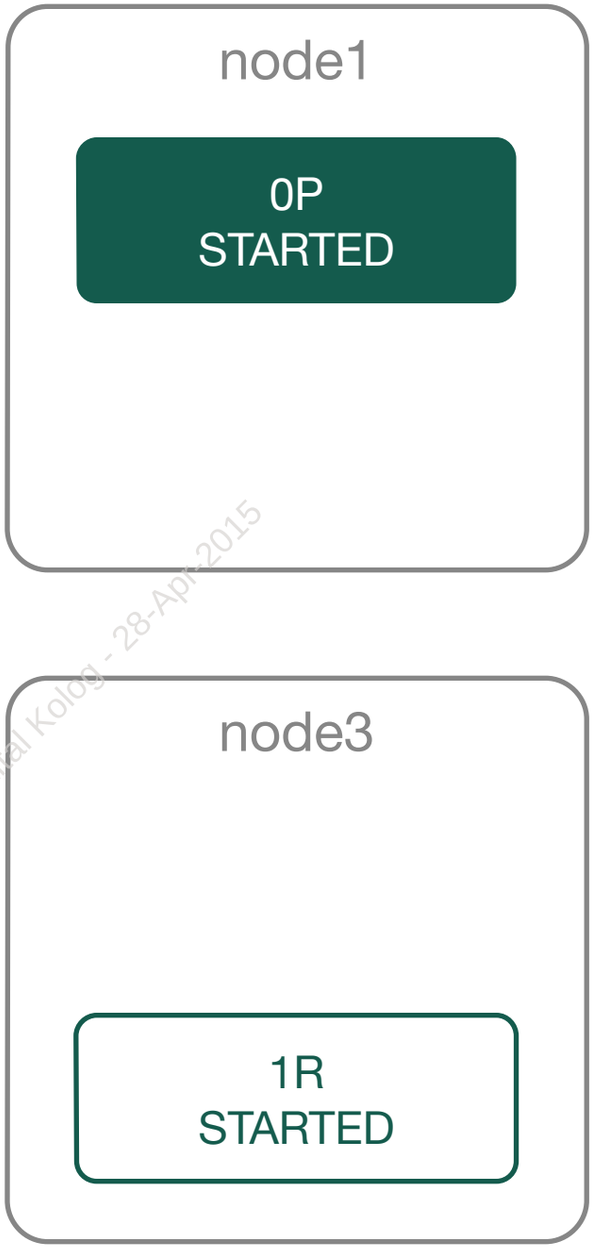


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Removing Nodes

- Remove *node2* and *node4*
- Master detects they were removed

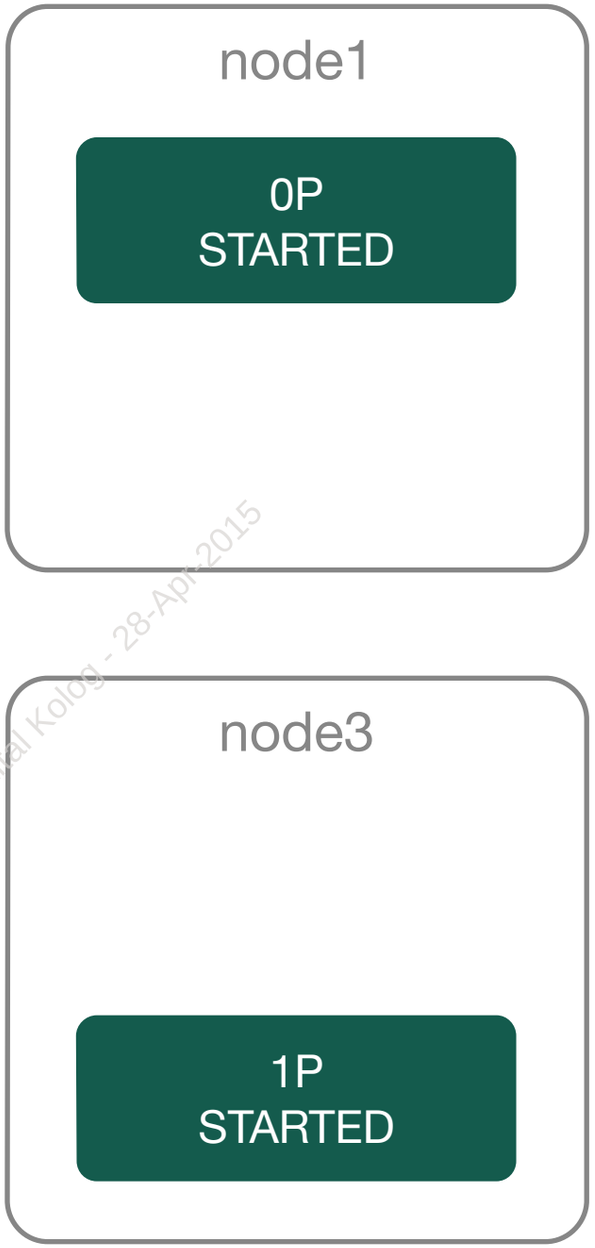


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Removing Nodes

- *1R* shard is promoted to *PRIMARY*, becomes *1P*

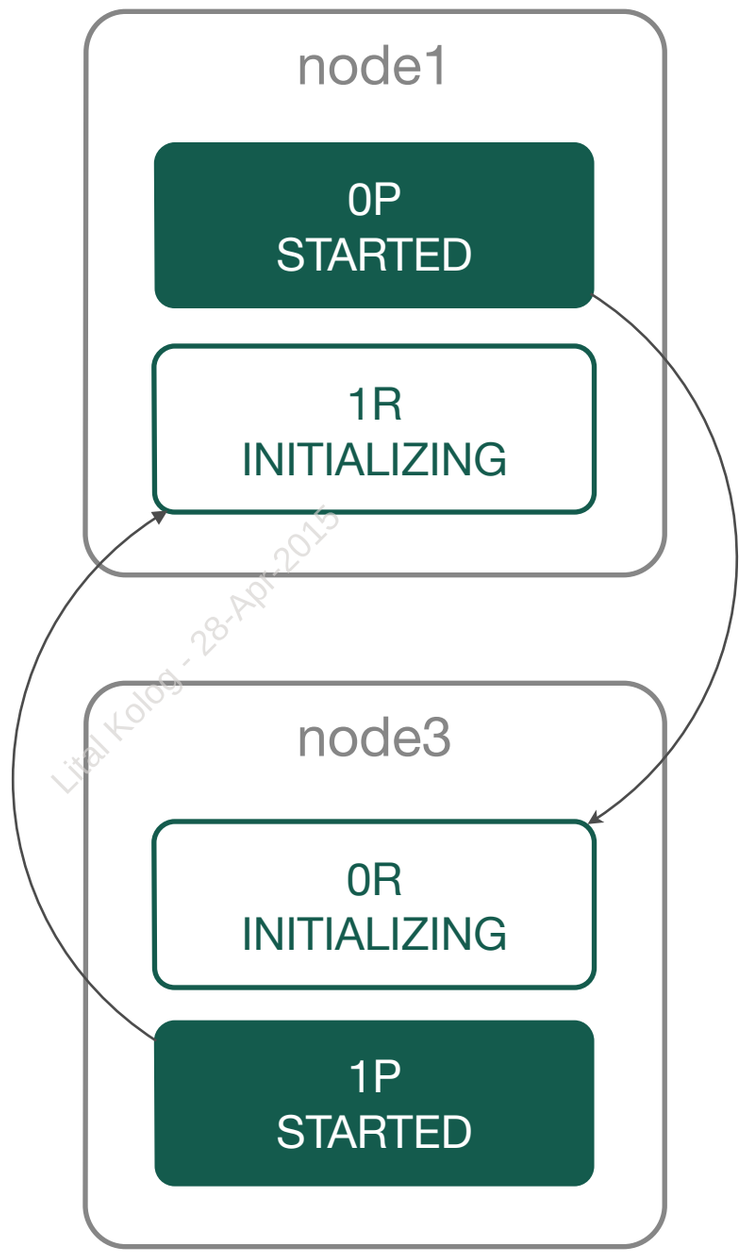


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Removing Nodes

- *0R* is allocated, in *INITIALIZING* state, recovering from *0P*
- *1R* is allocated, in *INITIALIZING* state, recovering from *1P*

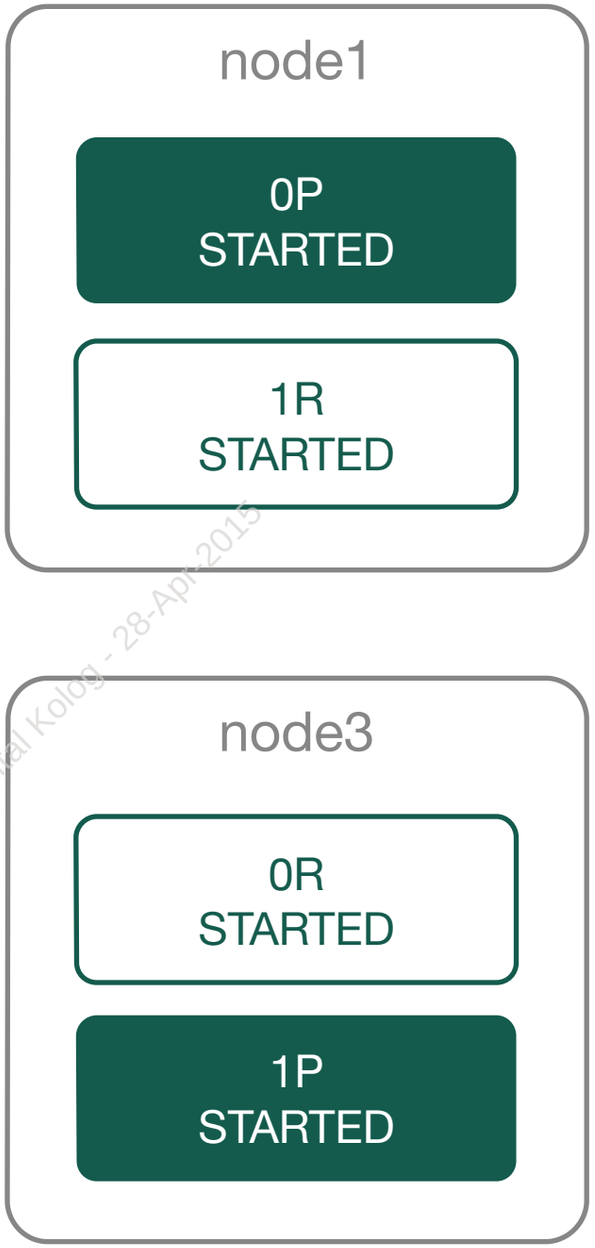


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shards Allocation - Removing Nodes

- *0R* is done recovering from *0P*, move to *STARTED*
- *1R* is done recovering from *1P*, move to *STARTED*



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Filtering

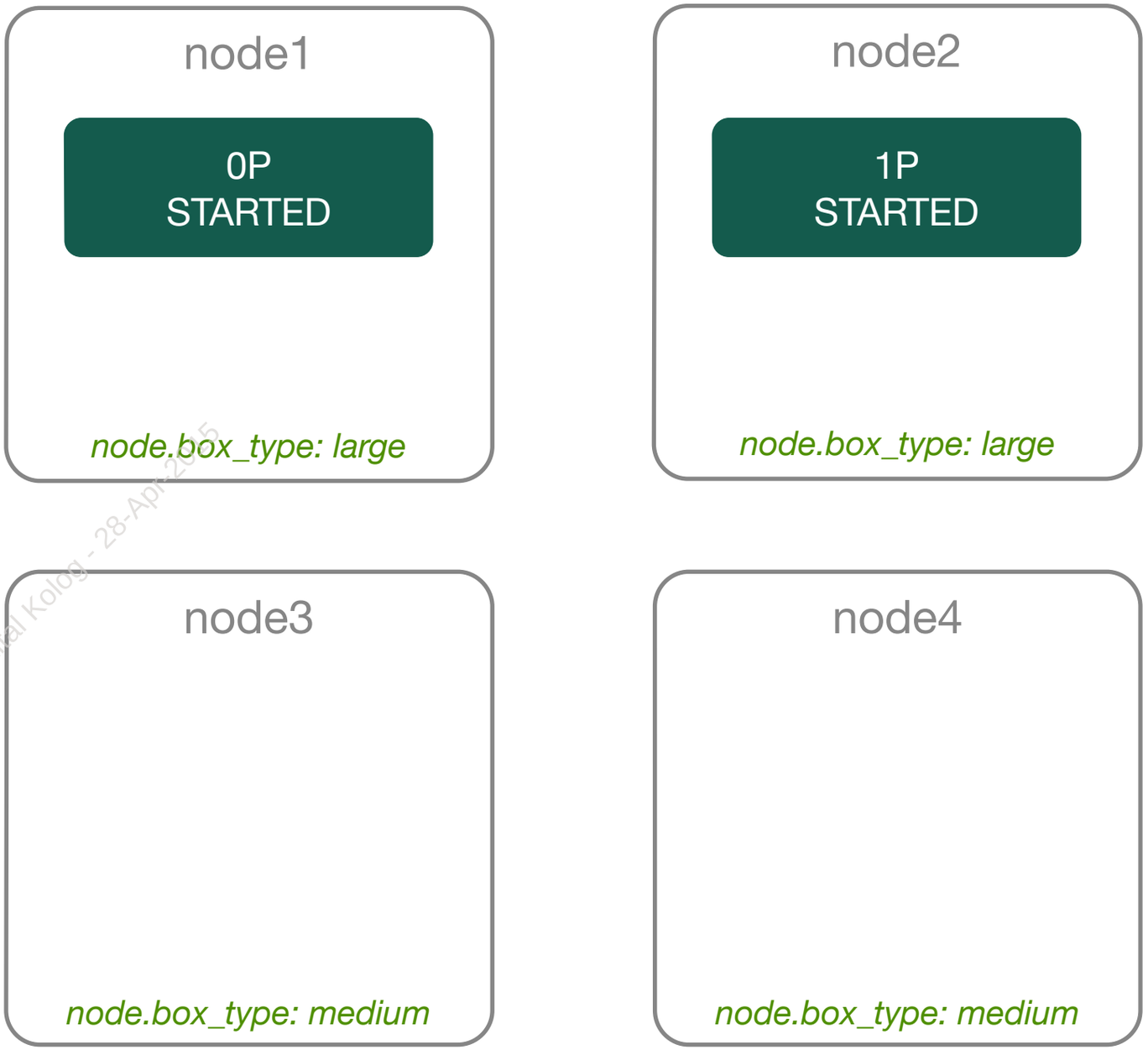
- The ability to control which nodes an index may be allocated to
- Each node can be associated with key value pairs that can later be used with filtering
- Index settings filtering allow to control allocation
  - Allowed to be updated in real time using index update settings API
    - *index.routing.allocation.include.[key]: [value]*
    - *index.routing.allocation.exclude.[key]: [value]*

Digital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Filtering

- Create an index with the *i.r.a.include.box_type* set to *large*
- The shards of this index will only be allocated on the matching nodes

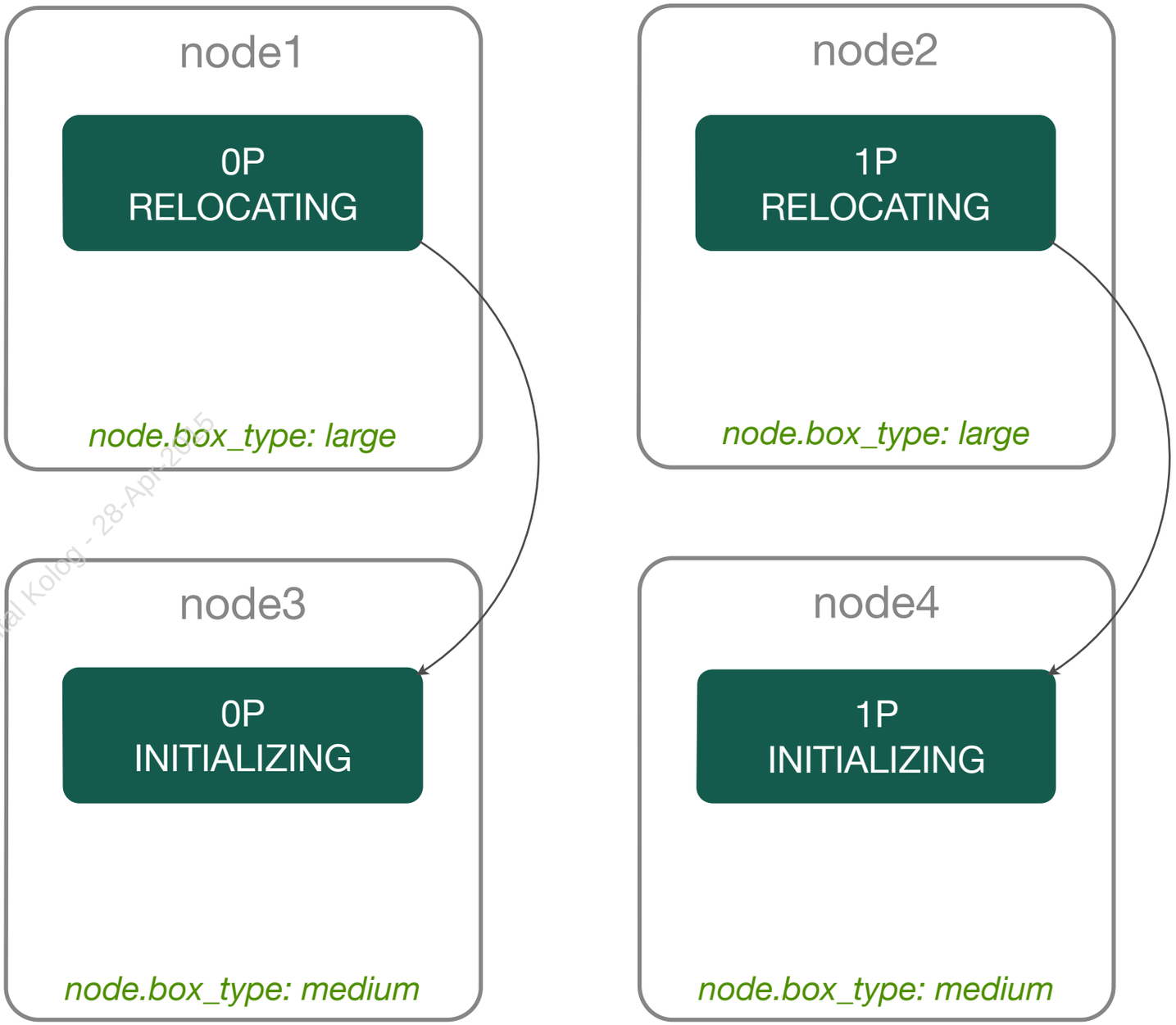


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Filtering

- Update setting *i.r.a.include.box_type* to *medium*
- Shards automatically relocate to conform to new filtering rules

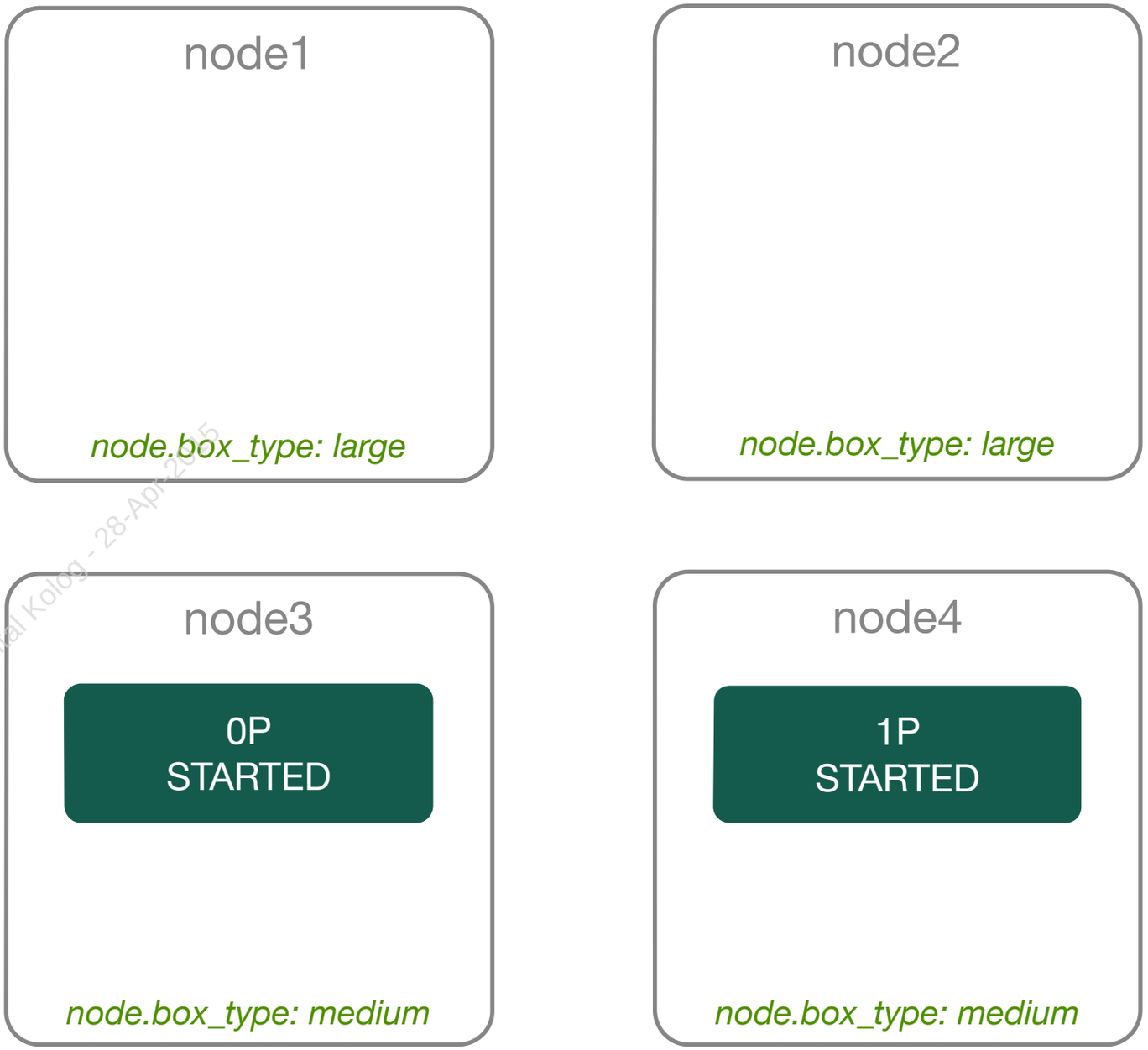


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Filtering

- Setting *i.r.a.include.box_type* still set to *medium*
- Relocation done, cluster state now conforms to new filtering rules



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Awareness

- Awareness of where to place copies of data is important
  - More resiliency to failure, for example, making sure copies of the data are even out among physical racks
  - Potentially faster read operations, for example, preferring to execute search on nodes within the same rack due to faster networking
- Uses nodes metadata (key value pairs)
- Each unique value for a key represents a "group"
  - For example, a key called *rack_id*, setting on all nodes

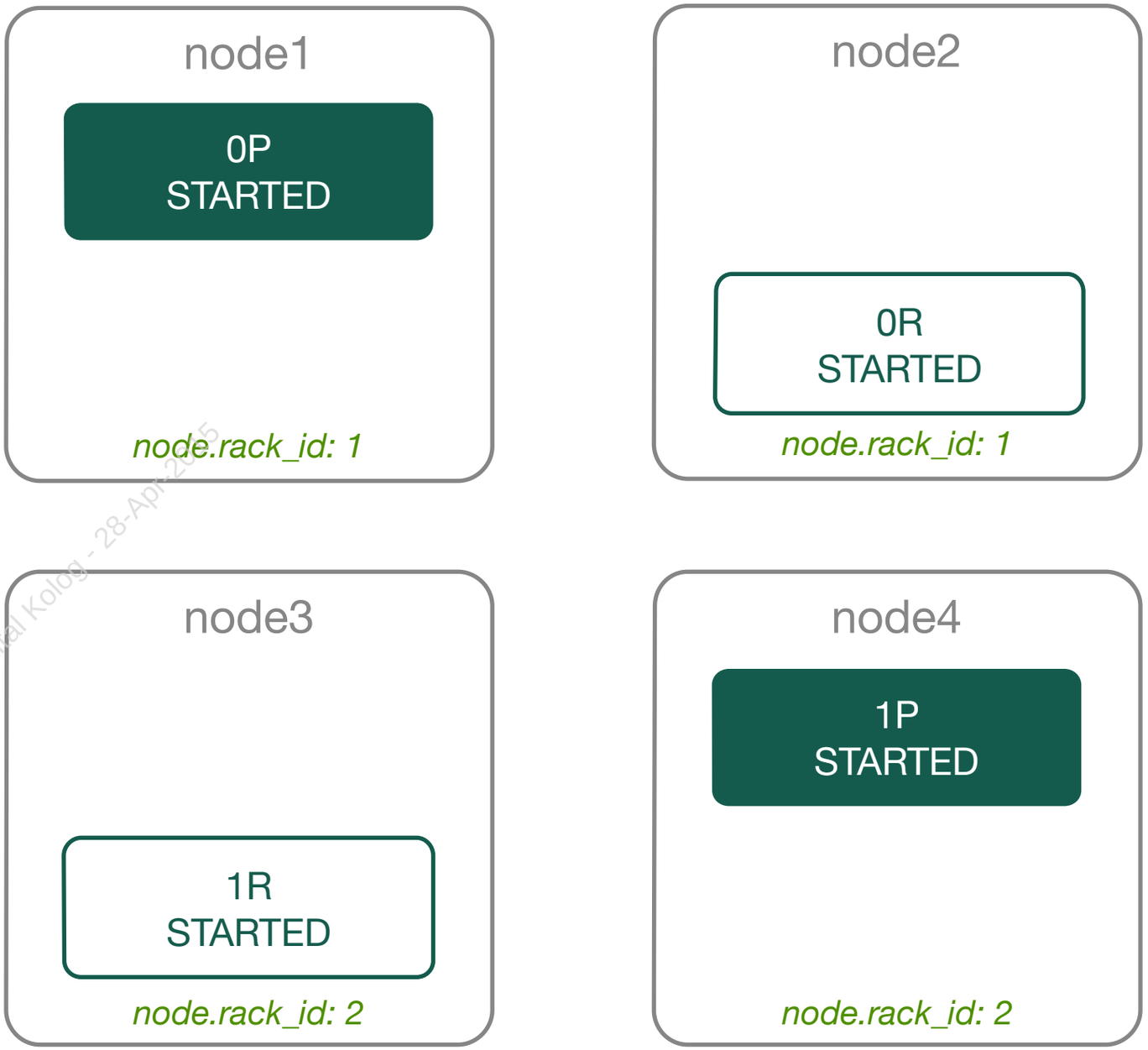
```
cluster.routing.allocation.awareness.attributes: rack_id
```

Lital Kolodny 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Awareness

- **No awareness**
- Copies of the data exists on the same “rack” (*0P* and *0R* are both on rack *1*)
- If one of the racks disconnects (e.g. switch failure), all copies of a shard are going with it

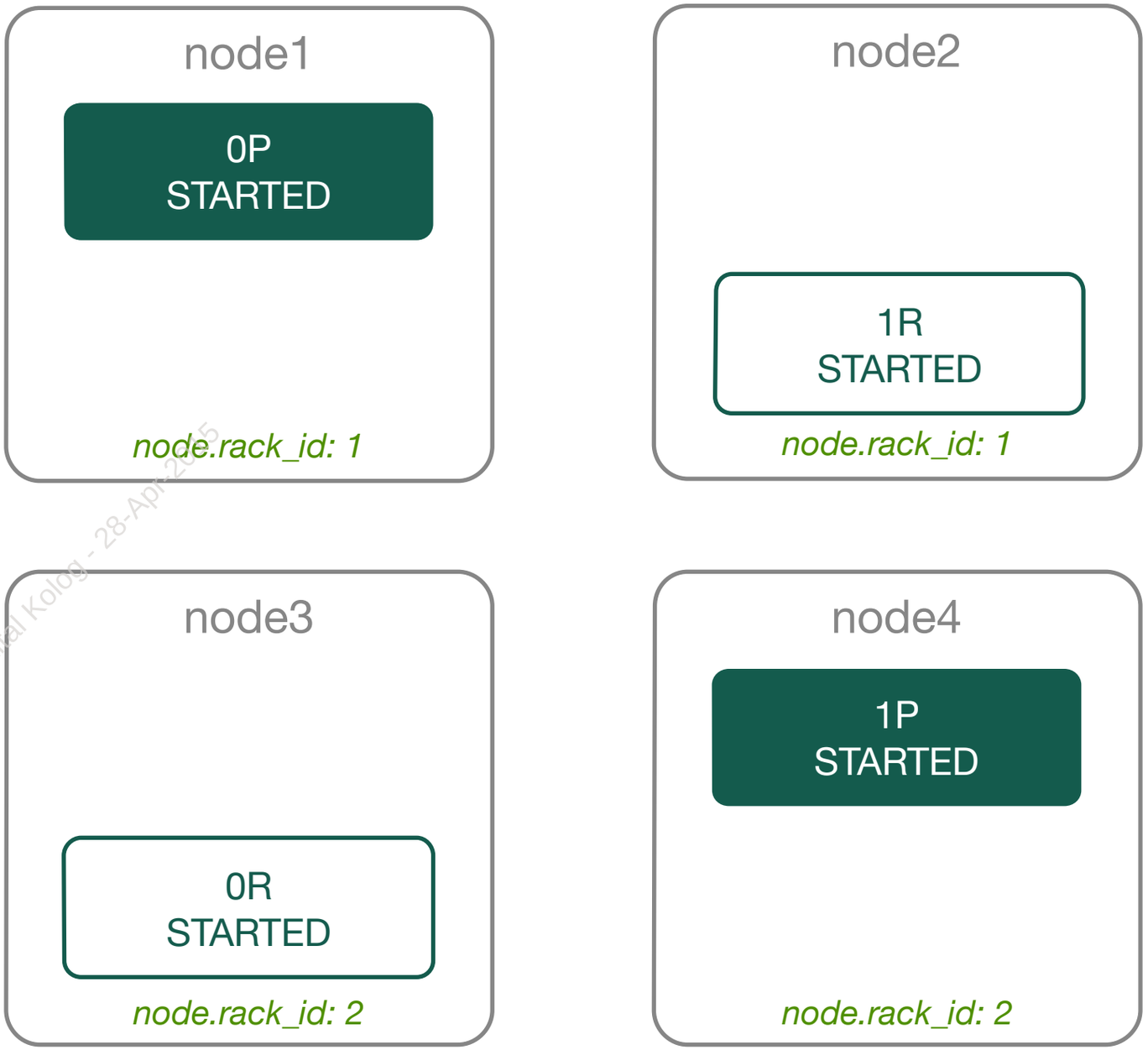


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Awareness

- **With awareness (based on *rack_id***
- Copies of the data will even out across existing *rack_id* values
- If a rack disconnects (e.g. switch failure), we still have copies of the shards on the other rack



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Forced Awareness

- Applicable to data centers or "AWS zones"
- Explicitly list the group values
  - Allowing to compute in advance how many copies should exist in each group
    - Never over-allocate copies from other groups
- Similar configuration to regular awareness
  - With an additional setting to list the forced group values

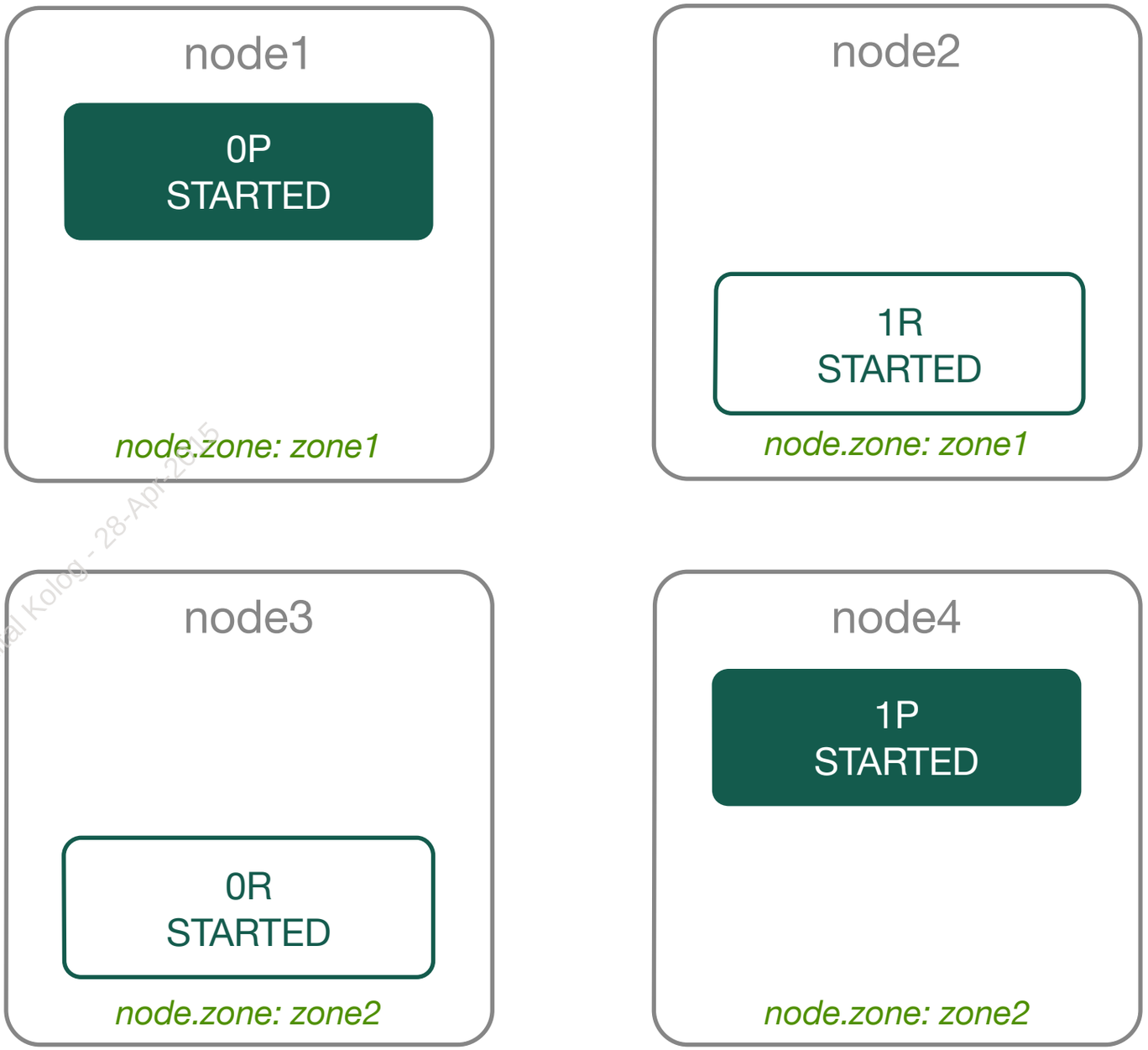
Lital Kobler 28-Aug-2015

```
cluster.routing.allocation.awareness.attributes: zone
cluster.routing.allocation.awareness.force.zone.values: zone1,zone2
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Forced Awareness

- **With (regular) awareness Using *zone***
- Copies of the data will even out across existing zone values

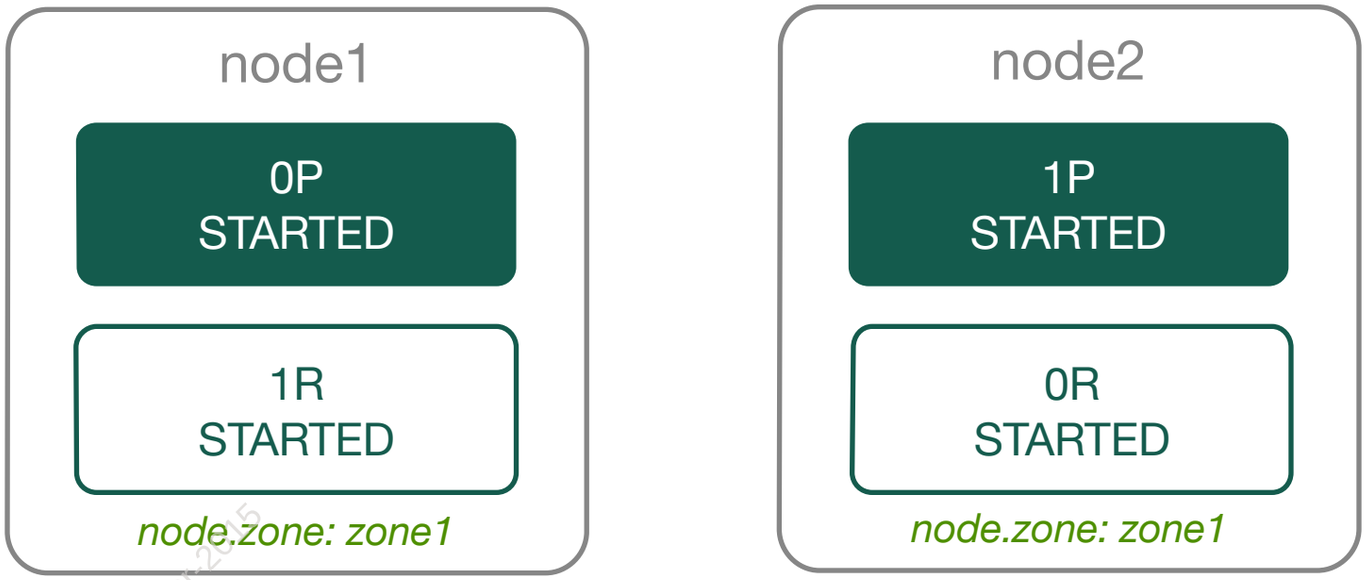


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Forced Awareness

- **With (regular) awareness Using *zone***
- If a zone goes down, copies will over-allocate on the rest of the zones
- Might cause problems since the zone might not have enough capacity to hold all those copies of the data

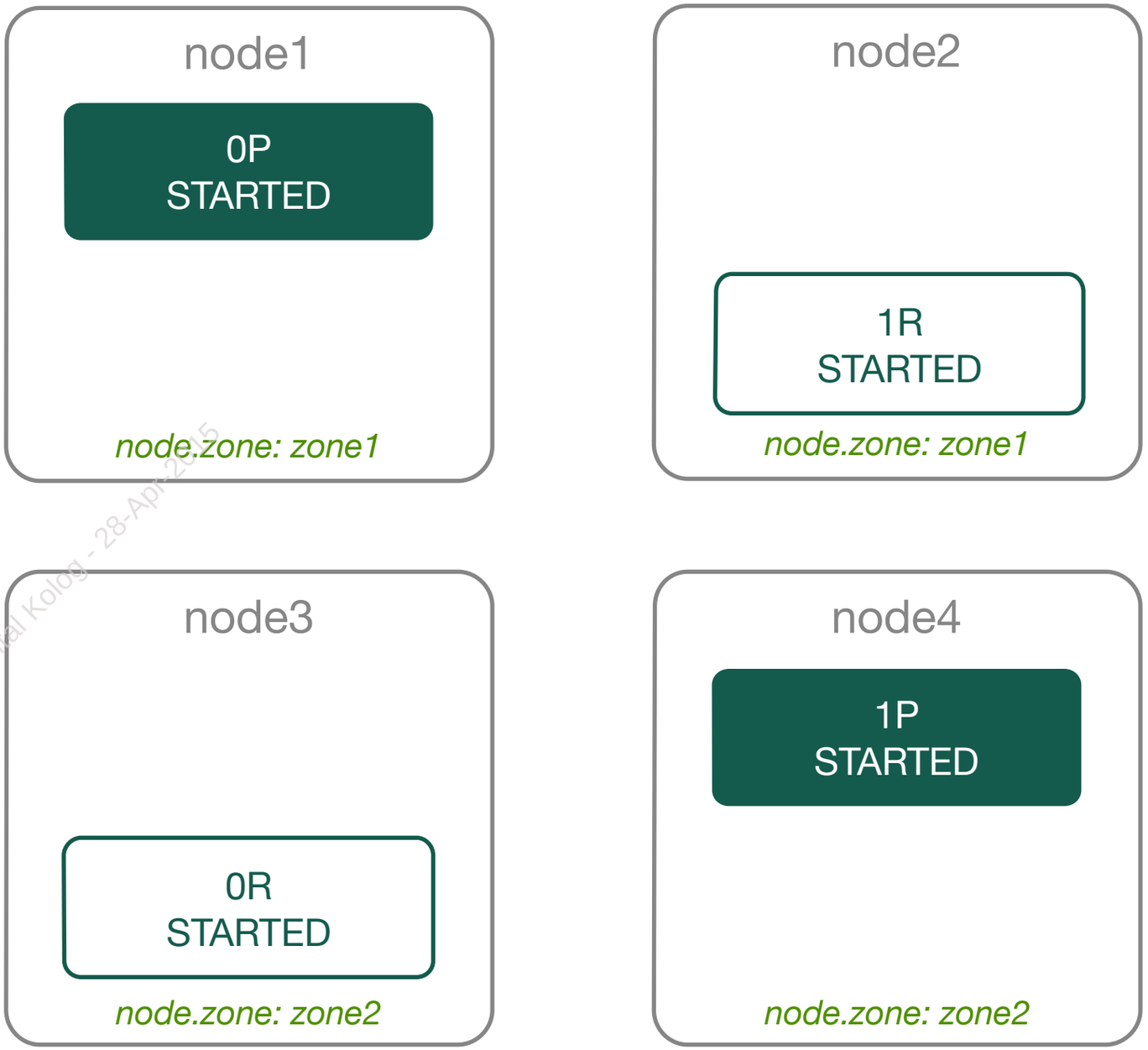


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Forced Awareness

- **With forced awareness Using *zone***
- Copies of the data will even out across existing zone values

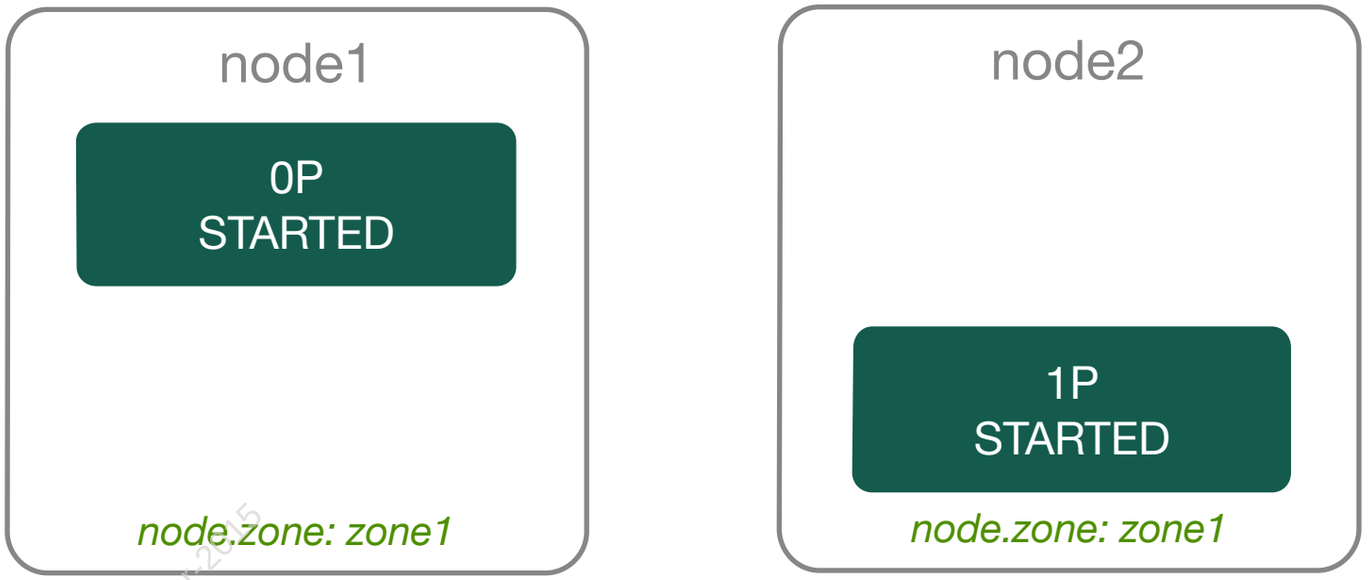


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Shard Allocation - Forced Awareness

- **With forced awareness Using *zone***
- If a zone goes down, copies will **not** be over-allocated on the other zones
- We can do that because we know what zones ("values") are expected form our cluster
- NOTE: If needed, replicas will still be promoted to primaries!



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Node Types

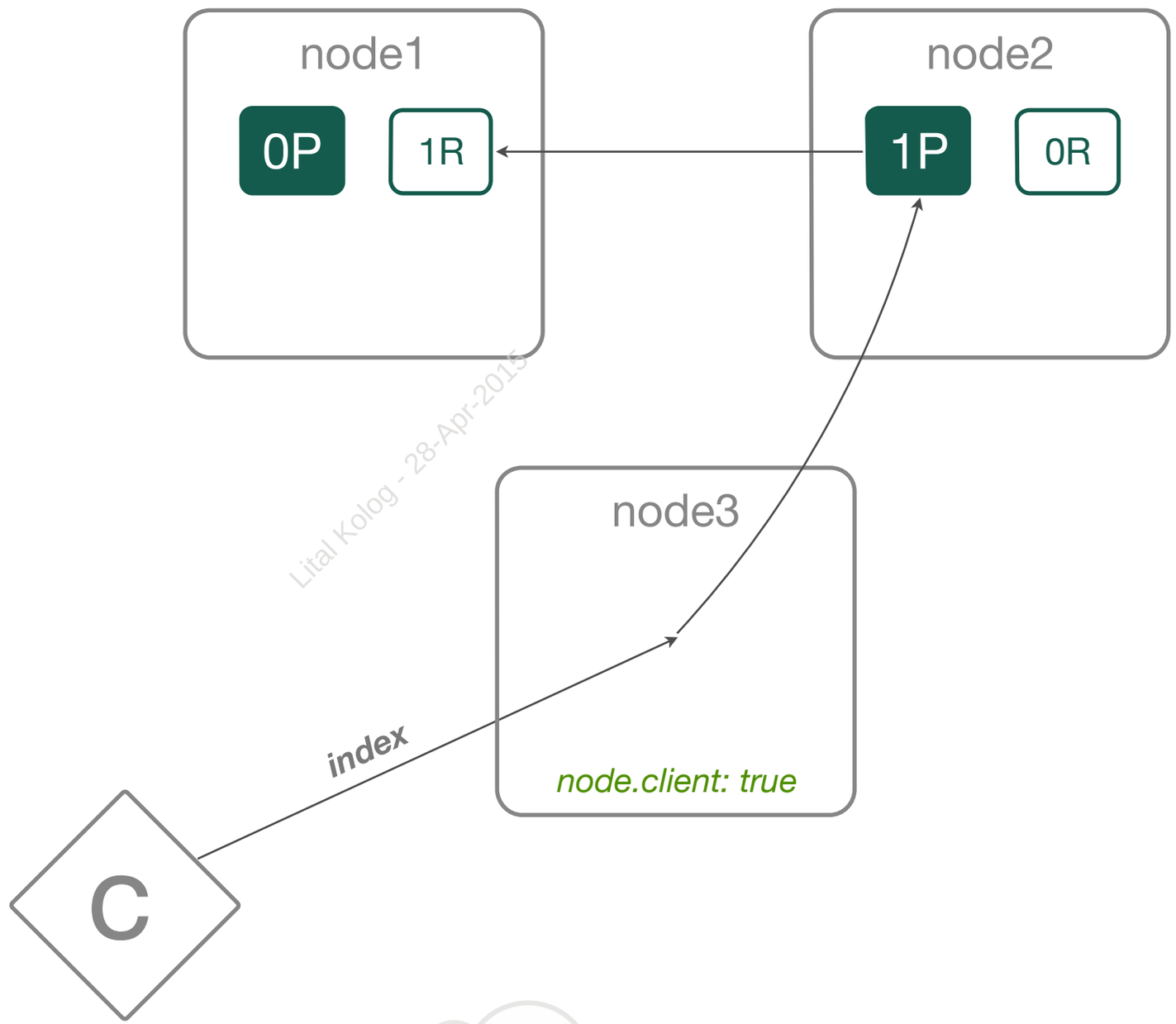
- Data Node (*node.data*)
  - A data node is a node that shards (data) is allowed to be allocated on it
- Master Node (*node.master*)
  - A master node is a node that is allowed to be elected as master
- Client Node (*node.client*)
  - A node that does not allow to have data allocated on it, and is not allowed to become master
- Tribe Node (*tribe.**)
  - A special client node that can communicate with multiple clusters

By default, a node is a data+master node

Lita Kopog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

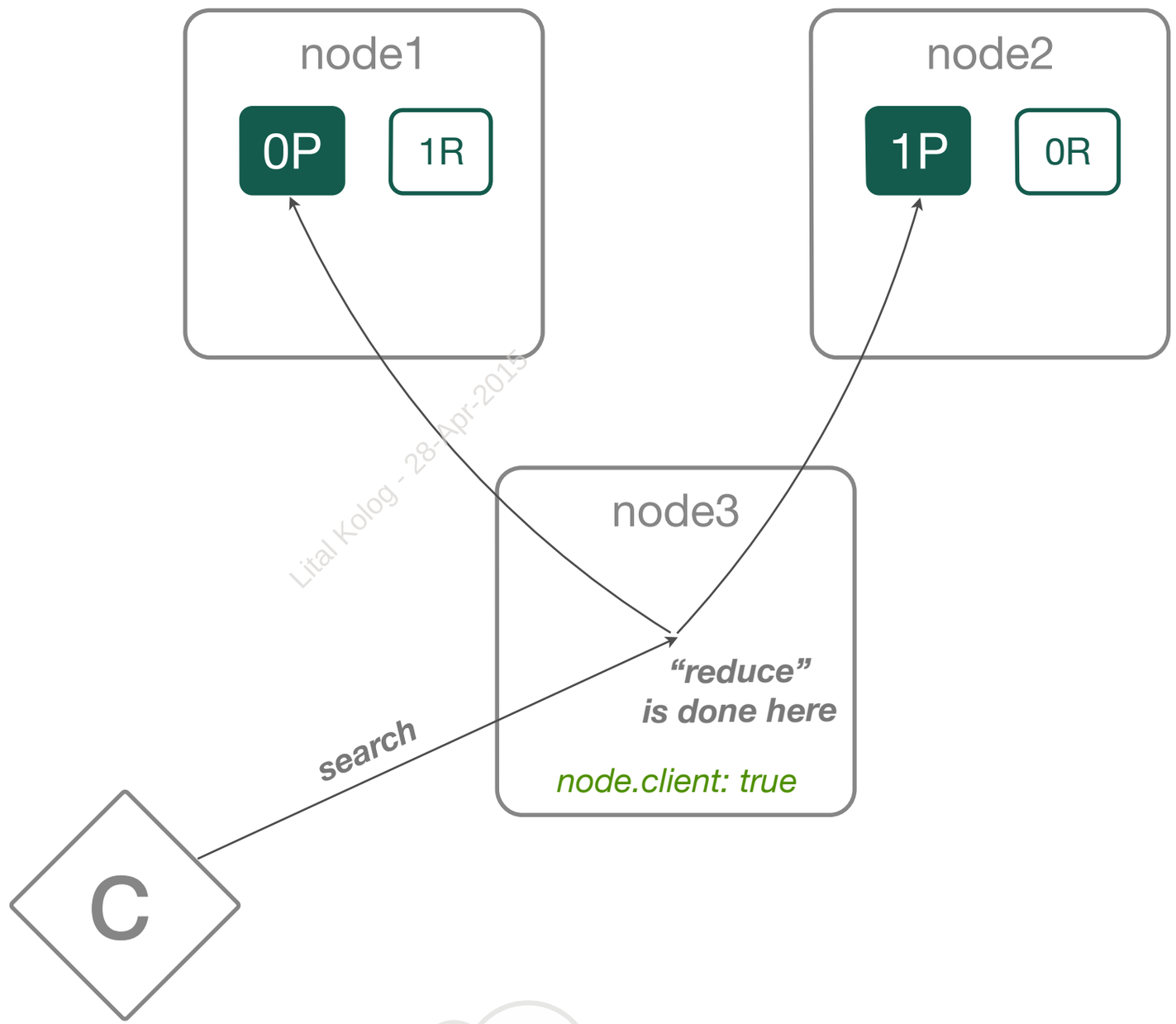
Node Types



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Node Types



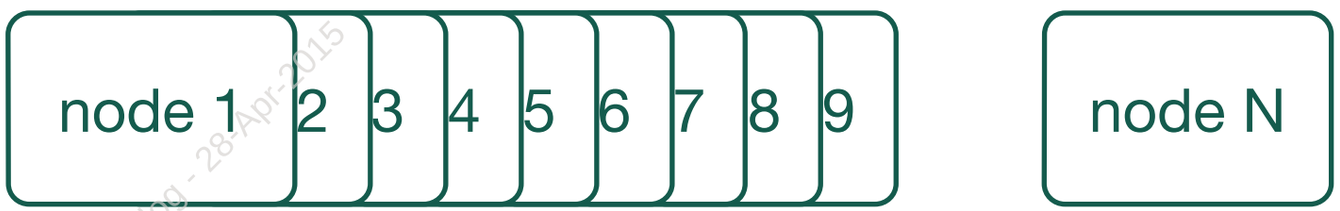
Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Node Types

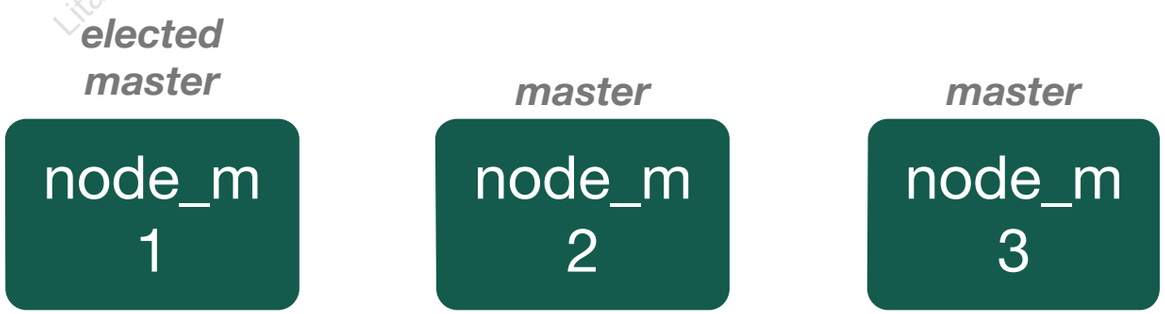
- Dedicated master nodes
  - Dedicated master nodes don't "suffer" from resource constraints data can cause
    - Master node is very lightweight

node.master: false
node.data: true



Lital Kolog - 28-Apr-2015

node.master: true
node.data: false



Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

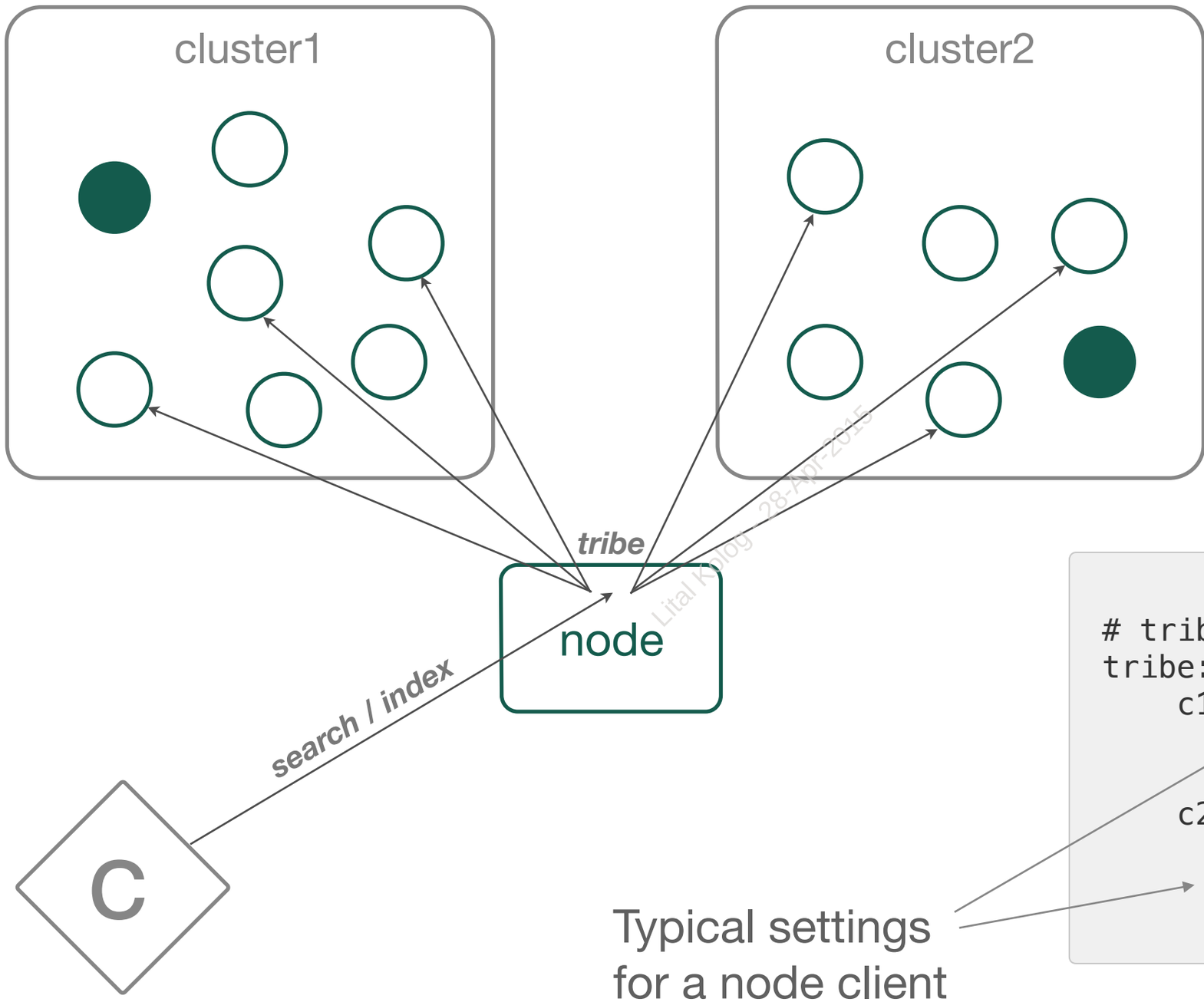
Node Types - Tribe

- Serves as a client to multiple cluster
  - "federated search"
- Works by running multiple internal client nodes and merging to a global (cross-cluster) state
  - A client node per cluster
- Support all data operations
  - Searching, indexing, percolation, aggregations, etc…
- Limited support for cluster management operations

Lital Kozlo - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Node Types - Tribe




```
# tribe node configuration
tribe:
    c1:
        cluster.name: cluster1
        ...
    c2:
        cluster.name: cluster2
        ...
```

Typical settings
for a node client

Lital Kplog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Networking

- All networking operations in elasticsearch are "evented"
  - non blocking
- Meaning no threads blocking to receive a response
  - Very important for real time, large, distributed systems
- For example:
  - A search on a cluster with 120 nodes across all of them will not have 120 threads blocking to receive the response from each node
  - 100 concurrent indexing requests will not have 100 network threads blocking waiting for the index operation to complete

Lital Kobar - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Detour: Networking

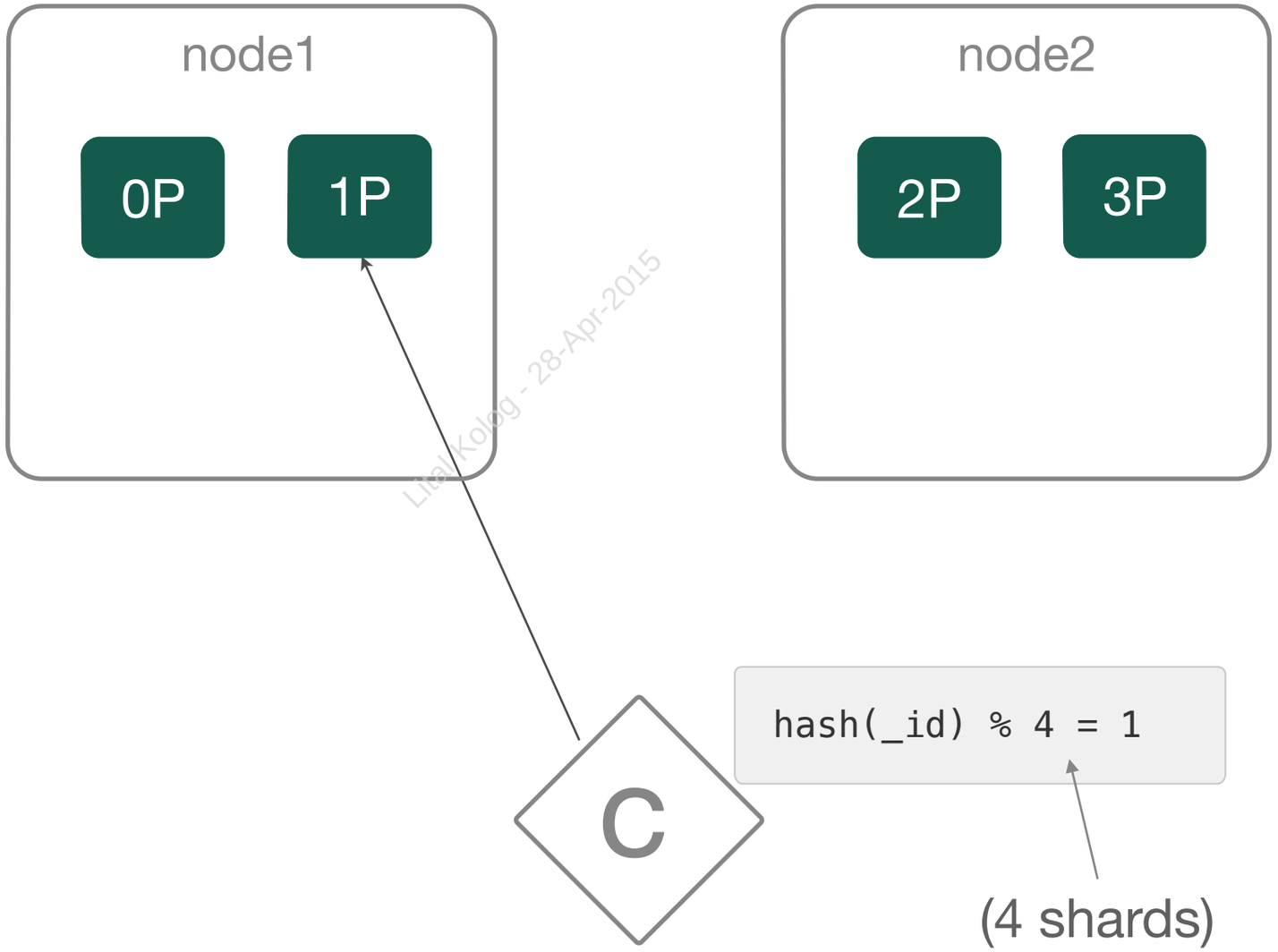
- Allows to reduce thread context switching
  - Operation level thread pools are important
    - 20 threads to do indexing, ...
  - No need for "networking" thread pool, worker network threads either
    - Pass the operation to be executed on the relevant operation thread pool
    - Pass the operation to another node over the networking layer

Lital Kolus – 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Index time
  - The process of controlling which shard a document will be allocated to



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Index time
  - The *_id* value is used by default to control which shard a document will be allocated on
  - A custom routing value can be provided, in which case, it will be used to control routing instead of the *_id*

```
curl -XPOST localhost:9200/crunchbase/person/1?routing=xerox -d '{
   ...
}'
```

Lital Kolog 22-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Index time
  - Can be marked as required in mapping, causing a failure if not provided

```
curl -XPUT 'localhost:9200/crunchrebase/' -d '{
  "mappings" : {
    "person" : {
      "_routing" : {
        "required" : true,
      },
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Index time
  - Can also be automatically extracted from the document
    - Comes with the cost of having to parse the document twice (to figure out the routing before the API is directed to the relevant shard)

```
curl -XPUT 'localhost:9200/crunchrebase/' -d '{
  "mappings" : {
    "person" : {
      "_routing" : {
        "required" : true,
        "path" : “group”
      }
    }
  }
}'
```

```
curl -XPUT localhost:9200/crunchbase/person/1 -d '{
  "group"   : "xerox",
  ...
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Get time
  - Specify which shard to execute the get / search APIs
  - Watch out: if routing is used at index time, it should be used with single doc APIs otherwise the *_id* will be used to pick the shard

```
curl -XGET localhost:9200/crunchbase/person/1?routing=xerox
```

Lital Kologrom 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Search time
  - Routing can be used to constrain the shards being searched on, otherwise, all shards will be hit

```
curl -XPOST localhost:9200/crunchbase/_search?routing=xerox -d '{
  "query" :  {
    "filtered" : {
      "query" : { ... },
      "filter" : { "term" : { "group" : "xerox"} }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

We still need the filter to make sure we only find the “xerox” group docs

If we don’t specify routing, the search still works correctly, but it will simply be executed on all the shards

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Search time
  - Multi routing values are supported

Multiple routing values will automatically resolve to the relevant shard(s) for the search to execute on

```
curl -XPOST localhost:9200/crunchbase/_search?routing=xerox,apple -d '{
  "query" :  {
    "filtered" : {
      "query" : { ... },
      "filter" : { "terms" : { "group" [ "xerox", "apple"] } }
    }
  }
}'
```

Lital Kolog - 28-Apr-2015

The filter now is either “xerox” or “apple”

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Routing

- Aliases can simplify routing
  - By associating a routing value with an alias to an index
  - And even a filter
  - By working well when multiple aliases are searched upon
  - We will learn more about aliases in the *index management* section, and more about how you can utilize routing and aliases in the *data management* section

Vital Kalogi - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replication

- Each shard can be replicated multiple times
  - *index.number_of_replicas* controls the number of additional copies of the data
- Required for:
  - Fault tolerance
  - Balancing read requests across more nodes
- Replication is document based
  - A document is indexed on the primary shard, and then replicated and indexed on all the replicas

Lital Kozlovski - 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replication

- By default, replication is done in a sync manner
  - Meaning the API call will not return until the docs have been indexed on all replicas
    - Still, multiple replicas are indexed in parallel
- Can be changed to async replication
  - Note, async replication will return immediately after the document was indexed on the primary shard (will not wait for replicas response).

```
curl -XDELETE localhost:9200/crunchbase/person/1?replication=async
```

Vital Korea 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replication

- Write Consistency
  - The minimum active shard copies (primary+replicas) - *N* - required in order for a write operation to be permissible
  - Supports:
    - *quorum*: *N/2 + 1* available shard copies
    - *one*: at least one available shard copy
    - *all*: all shard copies must be available

```
curl -XDELETE localhost:9200/crunchbase/person/1?consistency=all
```

Lital Kobo 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Management

Lital Kolog - 28-Apr-2015

elastic

Index Management

- There is a breadth of APIs exposed that allow to manage indices in a cluster
  - Including creating an index, deleting it, updating its settings, and execution management operations on it
- In this section we will cover those APIs
  - Excluding already covered APIs such as *refresh* and *optimize*

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Creating an Index

- Create Index API
  - An index can be explicitly created with its own set of settings

```
curl -XPUT 'localhost:9200/test' -d '{
  "settings" : {
    "number_of_shards" : 3,
    "number_of_replicas" : 0
  }
}'
```

Lisa Kolog - 28-Apr-2015

  - Note, an index is automatically created if it doesn't exists when indexing a document
    - Disable it by setting *action.auto_create_index* to *false*
  - As we've seen in the *mappings* section, the mappings can also be provided in the body

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Update Index Settings

- Some index level settings can be updated on a live index
  - For example, the *index.refresh_interval* or the *index.number_of_replicas*
- Use the update settings API to do so

```
curl -XPUT 'localhost:9200/my_index/_settings' -d '{
    "index.number_of_replicas" : 2
}'
```

Lital Kolog - 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Deleting an Index

- Delete Index API

```
curl -XDELETE localhost:9200/test
```

- Supports multiple indices

```
curl -XDELETE localhost:9200/test1,test2
```

- Or all indices

```
curl -XDELETE localhost:9200/_all
```

considered a destructive action

- Destructive operations can be controlled using the *action.destructive_requires_name* setting

Lise Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Templates

- Many times, multiple indices share the same mappings
- Index templates allow to create templates for multiple indexes matching on their name with settings and mappings

```
curl -XPUT localhost:9200/_template/te_prefix -d '{
    "template" : "te*",
    "settings" : {
        "number_of_shards" : 1
    },
    "mappings" : {
        "_default_" : {
            "_all" : { "enabled" : false }
        }
    }
}'
```

The name of the template

The wildcard expression, that when match on an index name, will automatically apply this template

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Templates

- Deleting a template

```
curl -XDELETE localhost:9200/_template/te_prefix
```

- Getting a template

```
curl -XGET localhost:9200/_template/te_prefix
```

  - Or all templates

```
curl -XGET localhost:9200/_template
```

Lital Kolog - 23-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Templates

- Multiple template matching is supported
  - All the templates are "merged" and apply to the index
  - An order flag be provided to control the order of matching
- Index created always have a "first" order precedence, overriding templates configuration parameters

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Open / Close Index

- Closing an index
  - Removes all shard allocations from the cluster
  - But keeps the index data around
  - Helps reduce the resources used on the cluster
- Opening an index
  - Allows to open a closed index

```
curl -XPOST localhost:9200/test/_close
curl -XPOST localhost:9200/test/_open
```

- Note, those are not “milliseconds” time operation, opening an index can take a few seconds to a couple of minutes
  - Flushing before closing will reduce the opening time

Vital Kolos - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore

- An incremental backup mechanism for indices
  - Only available in 1.0 and above
- Remember, replication != backup

Replicas

- fail-safe mechanism for hardware failures
- doesn't span multiple geographic regions
- doesn't protect from destructive operations
- immediate failover

Snapshot & Restore

- backup mechanism
- can span geographic regions
- requires time to restore data
- possibility of "rolling back" to previously stored data

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Repositories

- Storage system for backup data
- Multiple repositories per cluster
  - Each may contain multiple snapshots of data
- Different repository types are supported
  - File system
  - URL (read-only)
  - S3
  - Azure
  - HDFS

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Register repository

Snapshot/Restore - Repositories

Repository name

Repository type

```
curl -XPUT 'localhost:9200/_snapshot/main_backup' -d '{
    "type" : "fs",
    "settings" : {
        "location" : "/mnt/es-main-repo"
    }
}'
```

Lital Kolog - 28-Apr-2015

Repository settings
(changes per type)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Repositories

- File system
  - Snapshots the indices to local/shared file system

name	default	description
location		Shared location of repository
compress	*false*	Compress metadata
chunk_size	*unlimited*	Maximum allowed files size in repository

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Snapshots

- Point-in-time copy of data
- Incremental
  - Only files that changed since the last snapshot are copied
- Each snapshot is identified by unique name given by the user

Lital Blog - 28-Apr-2015

Repository name

Snapshot name

```
curl -XPUT 'localhost:9200/_snapshot/main_backup/snap1' -d '{
    "indices" : "+test_*,-test_4"
}'
```

List of indices

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Snapshots

name	default	description
indices	*all indices*	List of indices to be included into the snapshot
ignore_unavailable	*true*	Ignore unavailable
include_global_state	*true*	Include global state metadata (persistent settings, templates, etc)

Vital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Snapshots

- Wait for snapshot completion

```
curl -XPUT 'localhost:9200/_snapshot/main_backup/snap1?wait_for_completion=true'
```

- Get Snapshot information

```
curl -XGET 'localhost:9200/_snapshot/main_backup/snap1'
```

- Get information about all snapshots in repository

```
curl -XGET 'localhost:9200/_snapshot/main_backup/_all'
```

- Delete snapshot

```
curl -XDELETE 'localhost:9200/_snapshot/main_backup/snap1'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Restore

- Documents & Fields
- A document is modeled as a JSON object

Repository name

Snapshot name

```
curl -XPOST 'localhost:9200/_snapshot/main_backup/snap1/_restore' -d '{
    "indices":"+test_*,-test_4",
    "ignore_unavailable": true,
    "include_global_state": false,
    "rename_pattern": "test_(.+)",
    "rename_replacement": "restored_index_$1"
}'
```

Lital Kalos - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Snapshot/Restore - Restore

name	default	description
indices	*all*	List of indices to be restored
ignore_unavailable	*true*	Ignore unavailable
include_global_state	*true*	Restore global state metadata
rename_pattern		Rename pattern
rename_replacement		Rename replacement

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Aliases

- An alias allows to have a name aliasing a single/several concrete index/indices
- Adding an alias:

```
curl -XPOST 'localhost:9200/_aliases' -d '{
    "actions" : [
        { "add" : { "index" : "employees", "alias" : "current" } }
    ]
}'
```

- Removing an alias:

Lital Kolog - 28-Apr-2015

```
curl -XPOST 'localhost:9200/_aliases' -d '{
    "actions" : [
        { "remove" : { "index" : "employees", "alias" : "current" } }
    ]
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Aliases

- Renaming an alias
  - Multiple actions are executed in an atomic manner

```
curl -XPOST 'localhost:9200/_aliases' -d '{
    "actions" : [
        { "remove" : { "index" : "employees1", "alias" : "current" } },
        { "add" :    { "index" : "employees2", "alias" : "current" } }
    ]
}'
```

Lital Kozlovski - 28-Apr-2015

- Associating an alias with multiple indices

```
curl -XPOST 'localhost:9200/_aliases' -d '{
    "actions" : [
        { "add" : { "index" : "employees", "alias" : "contacts" } },
        { "add" : { "index" : "clients", "alias" : "contacts" } }
    ]
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Aliases

- Filters can be associated with an alias
  - Any search executed against the alias will automatically have the filter applied to it

```
curl -XPOST 'localhost:9200/_aliases' -d '{
    "actions" : [
        {
            "add" : {
                "index" : "employees",
                "alias" : "sales",
                "filter" : { "term" : { "department" : "sales" } }
            }
        }
    ]
}'
```

Lital Kolomer 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Index Aliases

- Routing values as well:
  - All operations executed against the alias will have the routing value used to route them
    - index will use it to pick the shard to index the doc
    - search will use it to pick the shard to search on

Lital Kolog 28-Apr-2015

```
curl -XPOST 'localhost:9200/_aliases' -d '{
    "actions" : [
        {
            "add" : {
                 "index" : "employees",
                 "alias" : "usa",
                 "routing" : "usa"
            }
        }
    ]
}'
```

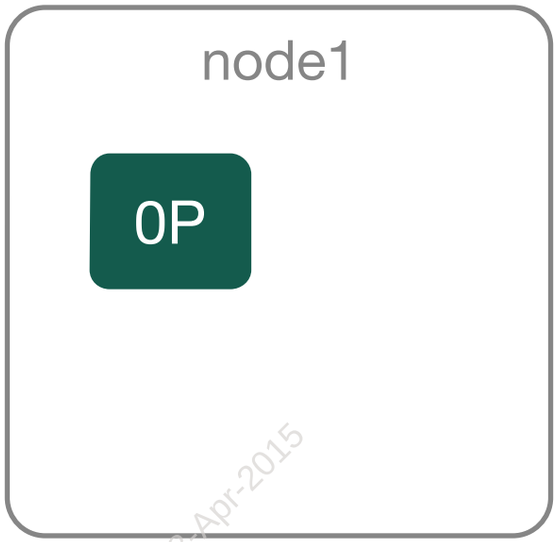

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Data Management

Lital Kolog - 28-Apr-2015

elastic

Overallocation



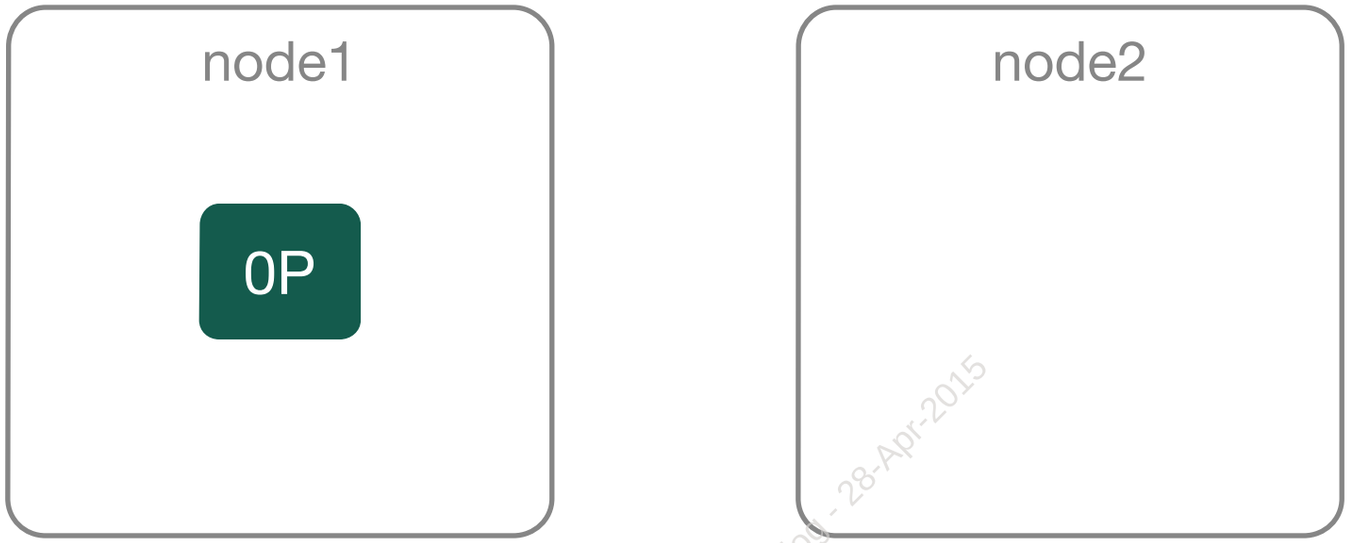
Lital Kolog - 28-Apr-2015

```
curl -XPUT localhost:9200/test -d '{
  "settings" : {
    "number_of_shards" : 1,
    "number_of_replicas" : 0
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Overallocation

- No scaling factor
- Potentially can split the shard in two



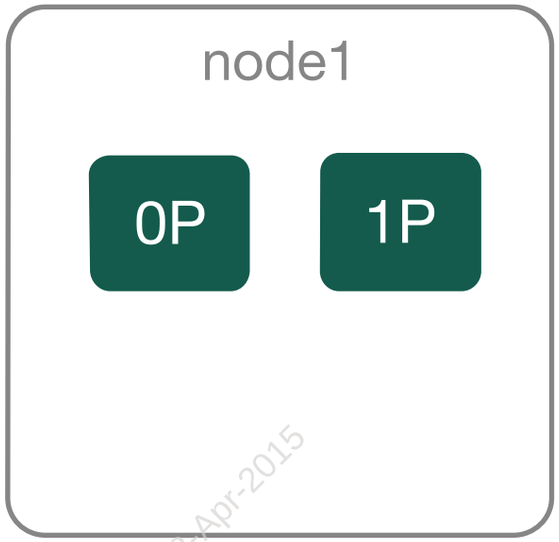
Lital Kolog - 28-Apr-2015

- No scaling factor
- Potentially can split the shard in two

  Expensive!, and each future split doubles the number of shards (assuming applied to all shards)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Overallocation



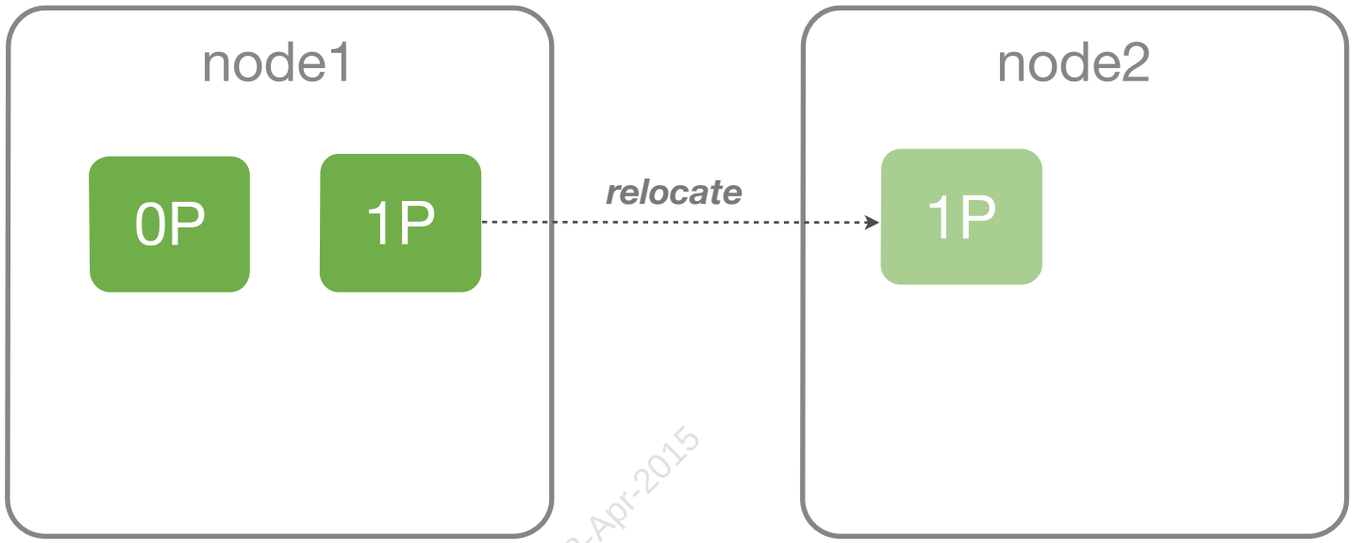
Lital Kolog - 28-Apr-2015

```
curl -XPUT localhost:9200/test -d '{
  "settings" : {
    "number_of_shards" : 2,
    "number_of_replicas" : 0
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Overallocation

- No scaling factor
- Potentially can split the shard in two



Lital Kolog - 28-Apr-2015

- Now, we can move shards around

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Overallocation

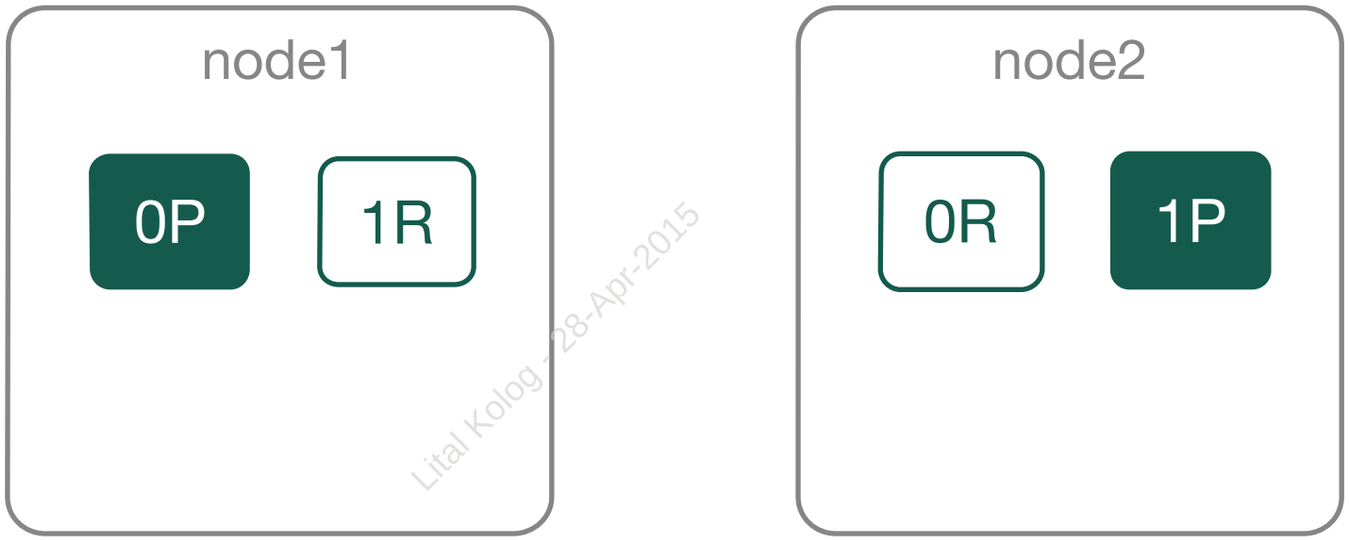
- *#shards > #nodes*
  - Common architecture where the number of shards (partitions) serves as your scaling unit
    - Other systems refer to them as "virtual nodes"
  - Moving shards is faster, and requires less resources, compared to splitting them

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replicas

- Multiple copies of the data for HA (High Availability)
- Also serves reads, allow to scale search



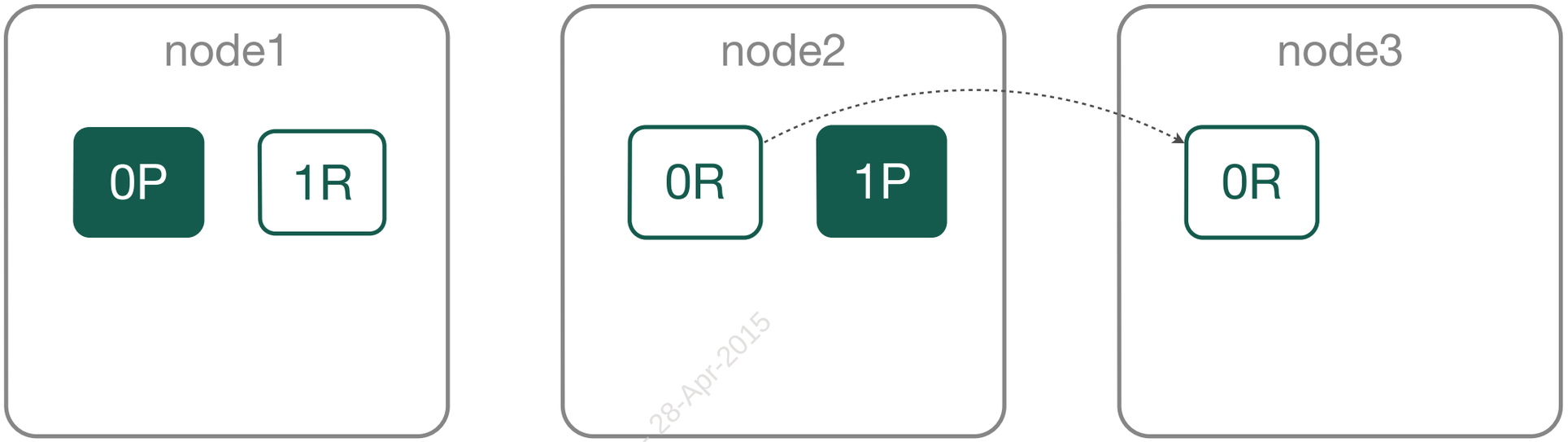
Lital Kolog - 28-Apr-2015

```
# add a replica
curl -XPUT localhost:9200/test/_settings -d '{
    "number_of_replicas" : 1
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replicas

- Shards move around to make use of the new node

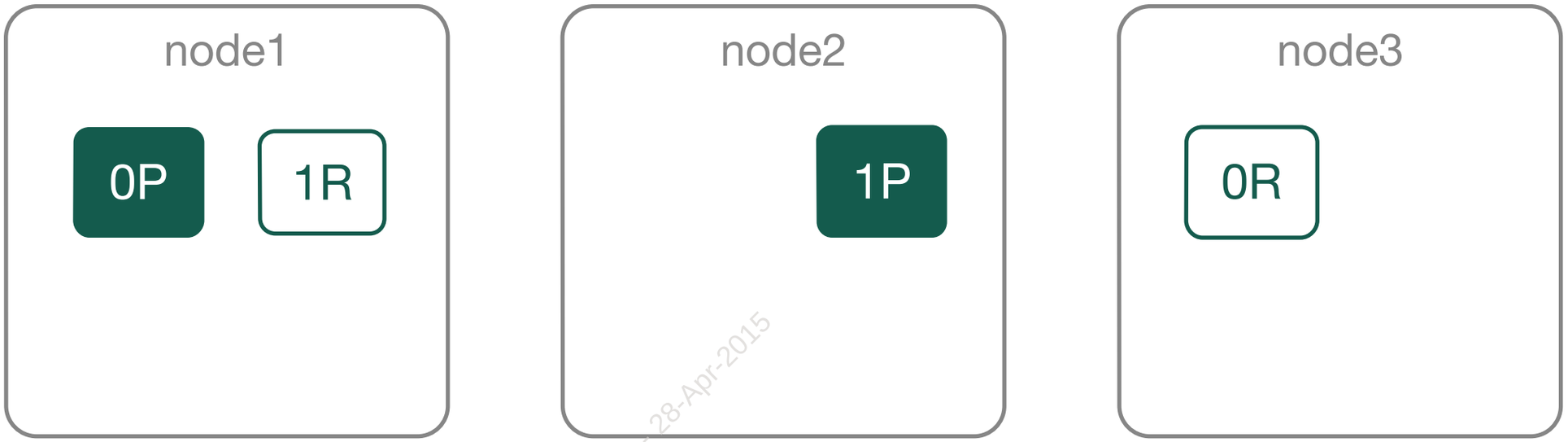


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replicas

- Shards move around to make use of the new node

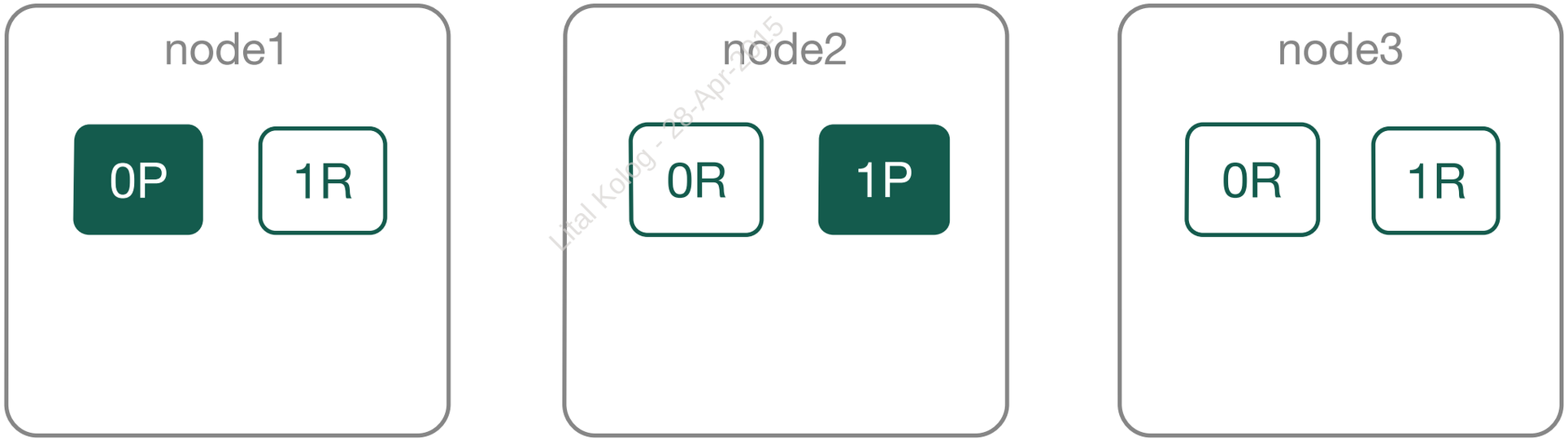


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Replicas

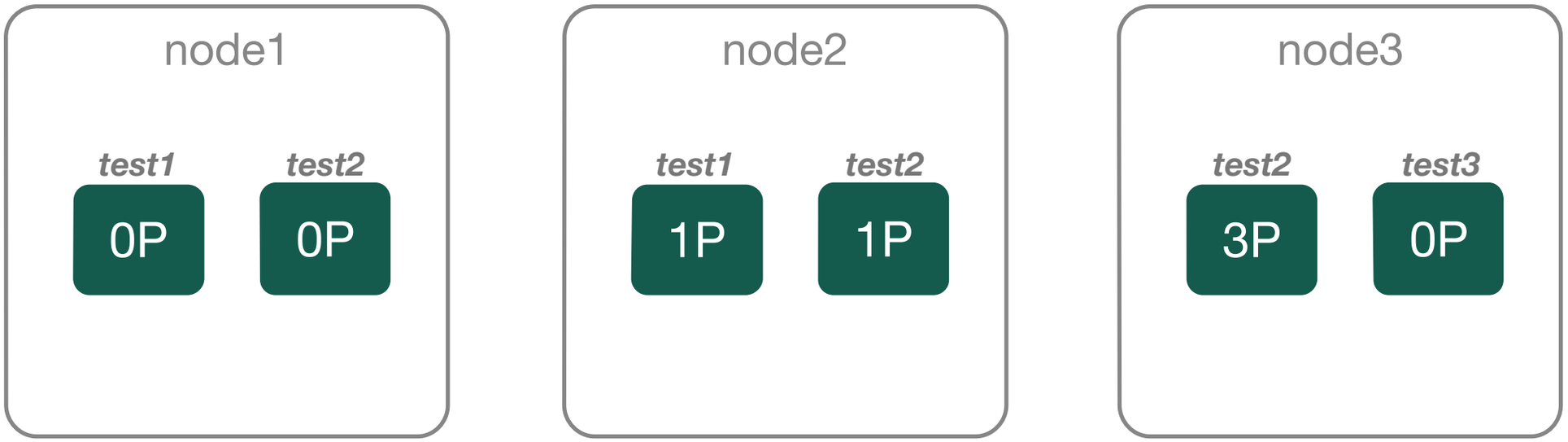
- Better HA, more fault tolerant
- Does not help search performance though
  - A single shard is plenty concurrent, and there aren't enough nodes to make use of the extra copy



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multiple Indices




```
curl -XPUT 'localhost:9200/test1' -d '{
  "settings" : {
    "number_of_shards" : 2,
    "number_of_replicas" : 0
  }
}'
```

```
curl -XPUT 'localhost:9200/test3' -d '{
  "settings" : {
    "number_of_shards" : 1,
    "number_of_replicas" : 0
  }
}'
```

```
curl -XPUT 'localhost:9200/test2' -d '{
  "settings" : {
    "number_of_shards" : 3,
    "number_of_replicas" : 0
  }
}'
```

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Multiple Indices

- Easily search across indices (*test1,test2,test3/_search*)
- Each index can have a different distributed characteristics
- Searching an index with 50 shards is the same as searching on 50 indices with 1 shard each

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Capacity Planning

- There is a "maximum" shard capacity
- No simple formula to it, depends on usage
- Many factors involved
  - hardware, number of docs, how data is indexed, size of data per doc, and type of searches (sort / aggregations), and more…
- The maximum shard size is easy to evaluate
  - Create a single index with one shard (no replicas) and start loading it
  - When overloaded, record the doc count for it

Lital Kotz - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Capacity Planning

- There is a "maximum" index capacity
  - Remember, number of shards is fixed per index
  - *num_of_shards * max_shard_size* (from previous slide)

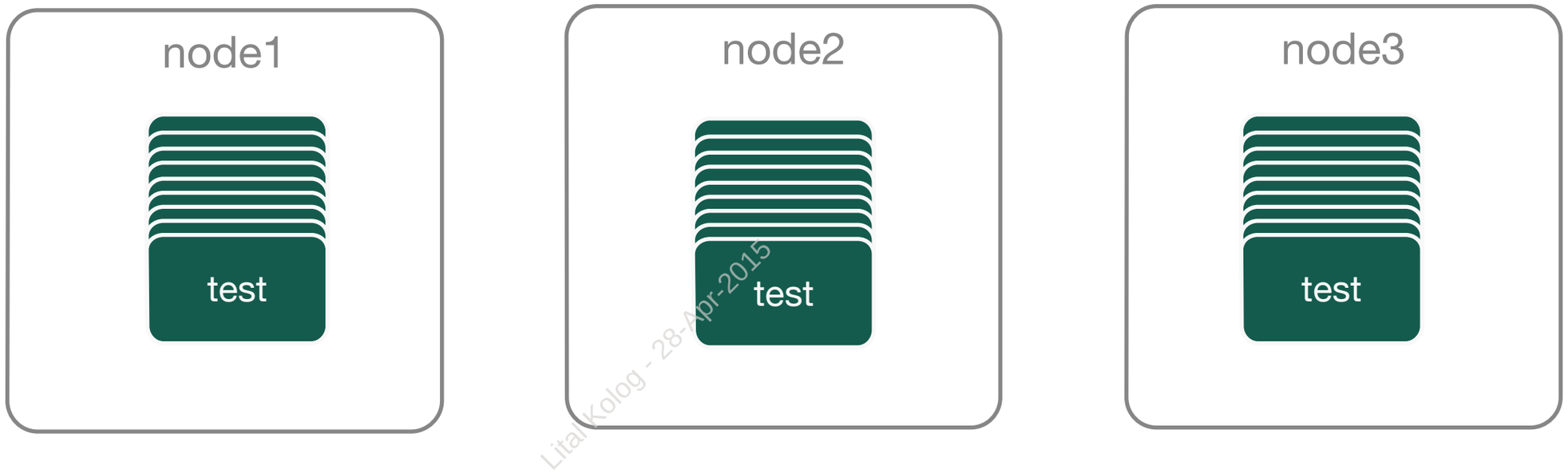
- Replicas play no part here, they are only additional copies of data
  - Need to be taken into account with cluster sizing

Lital Kozol - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Capacity? Ha!

Create a “kagillion” shards…



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Capacity? Ha!

- The "kagillion" shards problem
  - Each shard comes at a cost
    - Lucene under the covers
    - File descriptors
    - Memory requirements
    - Less compression on the total index size
  - Search requires to span all shards
    - Might be ok for key/value stores or small range lookups, but problematic for distributed search

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Sample Data Flows

- When designing a system, its important to understand how its data "flows"
- We will cover two common use cases
  - But note, this is for "special" cases
  - Typically the 5 shards index can take you a loooong way
    - And, it's fast!

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- Each "user" has its own set of documents
- Possibly big variance
  - Most users have small number of docs
  - Some have very large number of docs
- Most (if not all) searches are per user

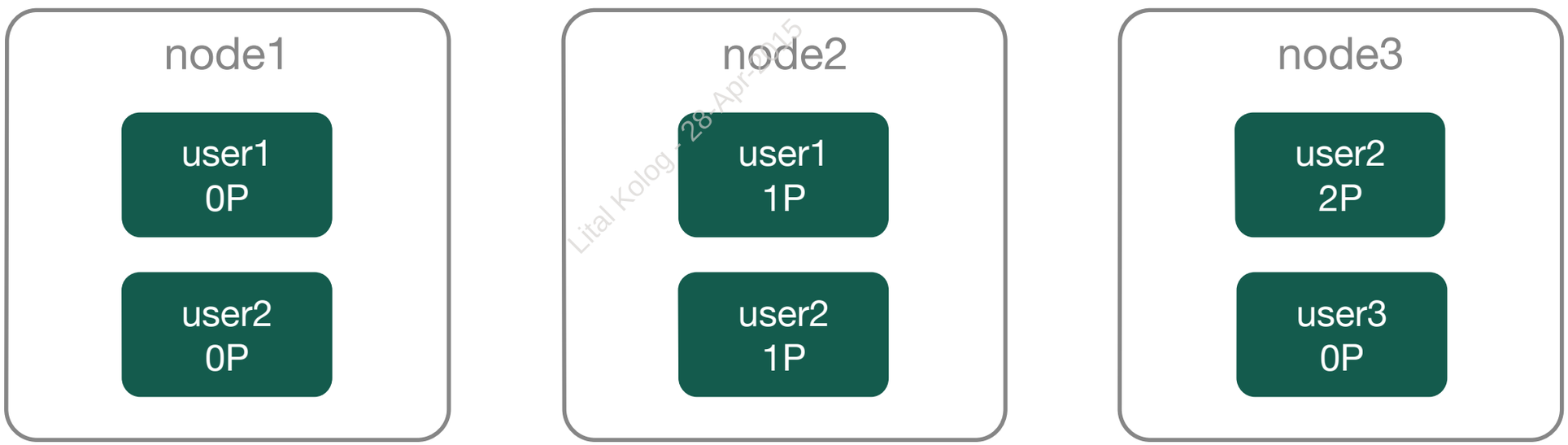
Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- Each user has its own index
  - Extremely simple to manage and use

- Possibly with different sharding



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

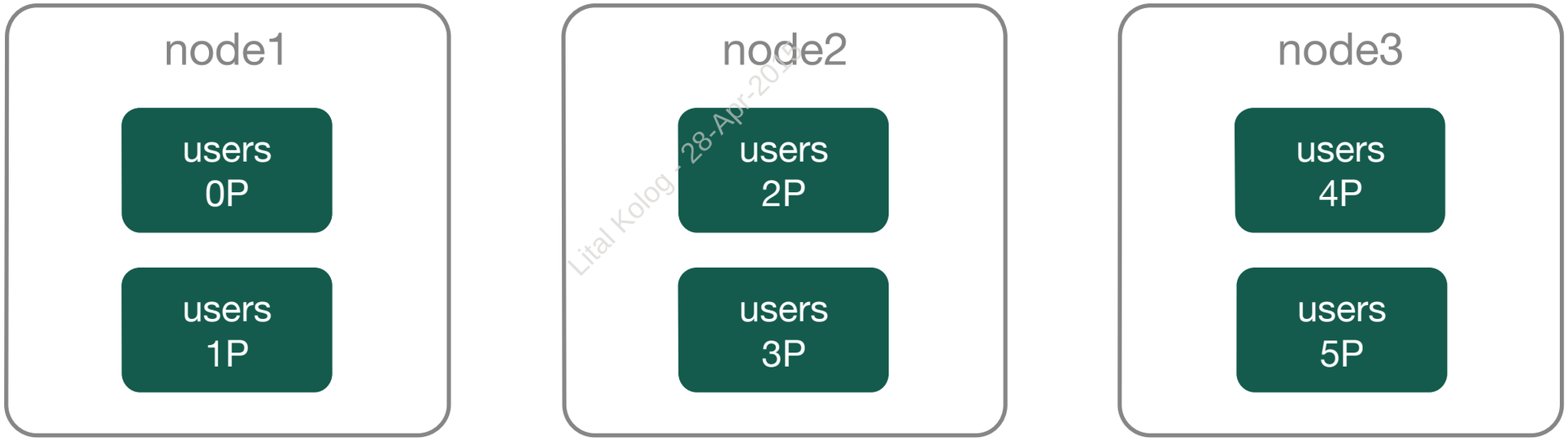
- An index per user
  - A single shard can hold a substantial amount of data (docs / size)
  - Small users become “resource expensive” as they require at least 1 shard
  - More nodes needed to be able to support so many shards
    - Its fine, but can have higher Total Cost of Ownership compared to overloading users on the same index

Vital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- All users are stored on the same index
- Overallocation of shards to support future growth
  - Just add nodes and shards will balance



Lital Kolog - 28-Apr-2021

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- Single index
  - Search is executed and filtered by a user_id
    - Filter caching makes this a snap
  - Search goes to all shards...
    - Expensive to do overallocation with not enough nodes

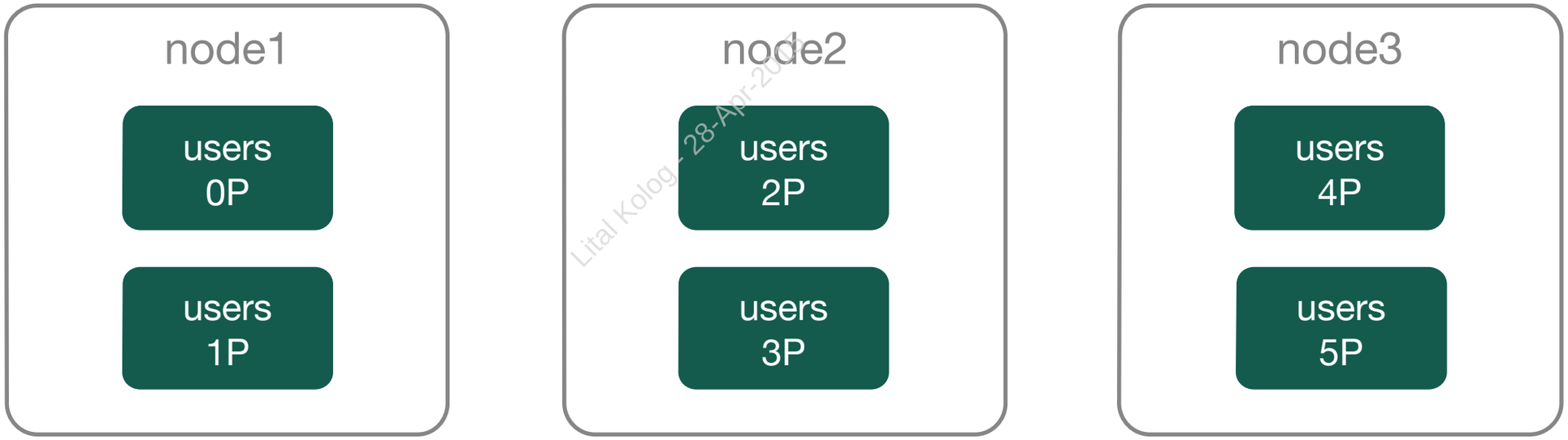
Lital Kozlo - 28-Apr-2015

```
curl -XGET localhost:9200/users/_search -d '{
  "query" : {
    "filtered" : {
      "query" : { ... },
       "filter" : { "term" : { "user_id" : "1" } }
    }
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- All data for a specific user is routed to the same shard when indexed
- Search uses routing + filtering, hits 1 shard
- Allows for excessive overallocation



Lital Kologi - 28-Apr-2026

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- Single index + routing
  - Index with routing

```
curl -XPUT 'localhost:9200/users/docs/1?routing=user1' -d '{
   ...
}'
```

  - Search with routing + filter

Kolog - 28-Apr-2015

```
curl -XGET 'localhost:9200/users/_search?routing=user1' -d '{
  "query" : {
    "filtered" : {
      "query" : { ... },
      "filter" : { "term" : { "user_id" : "user1" } }
    }
  }
}'
```


Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- Single index + routing
  - Aliases makes this a snap

```
curl -XPOST 'localhost:9200/_aliases' -d '{
  "actions" : [
    {
      "add" : {
        "index" : "users",
        "alias" : "user1",
        "filter" : { "term" : { "user" : "user1" } },
        "routing" : "user1"
      }
    }
  ]
}'
```

Lital Kolog - 28-Apr-2016

  - Index and search executes against the alias instead of the index name
  - Multi alias search possible (*/user1,user2/_search*)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Users Data Flow

- Single index + routing
  - Very large users can expand beyond a single shard limit
    - They can be migrated to their own index
    - With their own distributed aspects
  - Flipping the switch is a matter of flipping an alias
    - The new alias on the new index will not require routing or filtering

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Time Data Flow

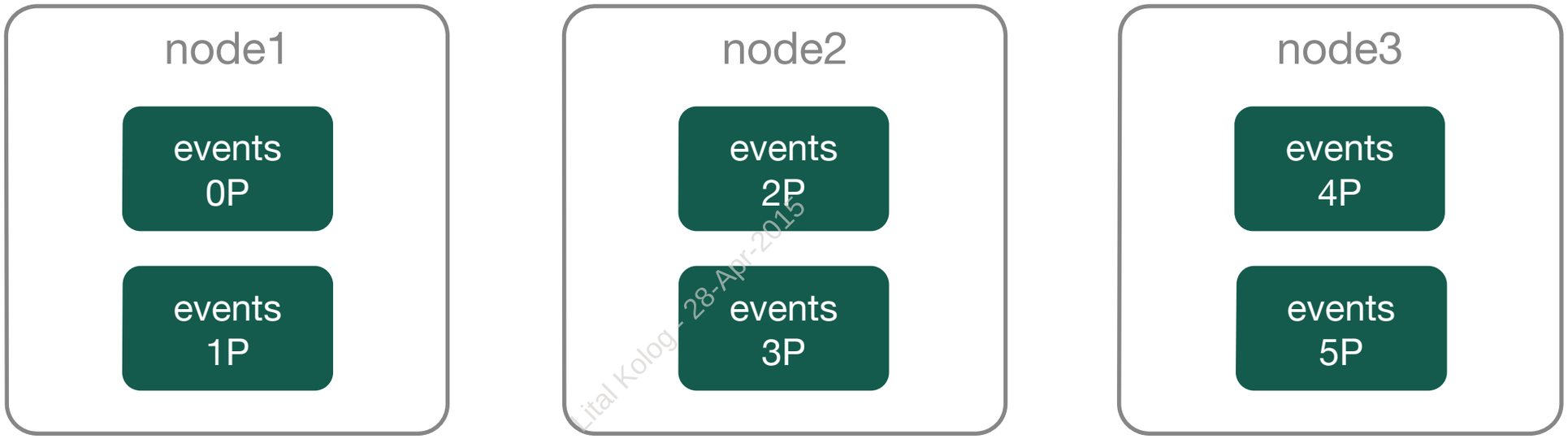
- Time based data (specifically, range enabled)
  - logs, social stream, time events
- Docs have a “timestamp”, don’t change (much)
- Search is usually ranged on the “timestamp”

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Time Data Flow

- Data keeps on flowing, eventually growing out of the shard count of the index

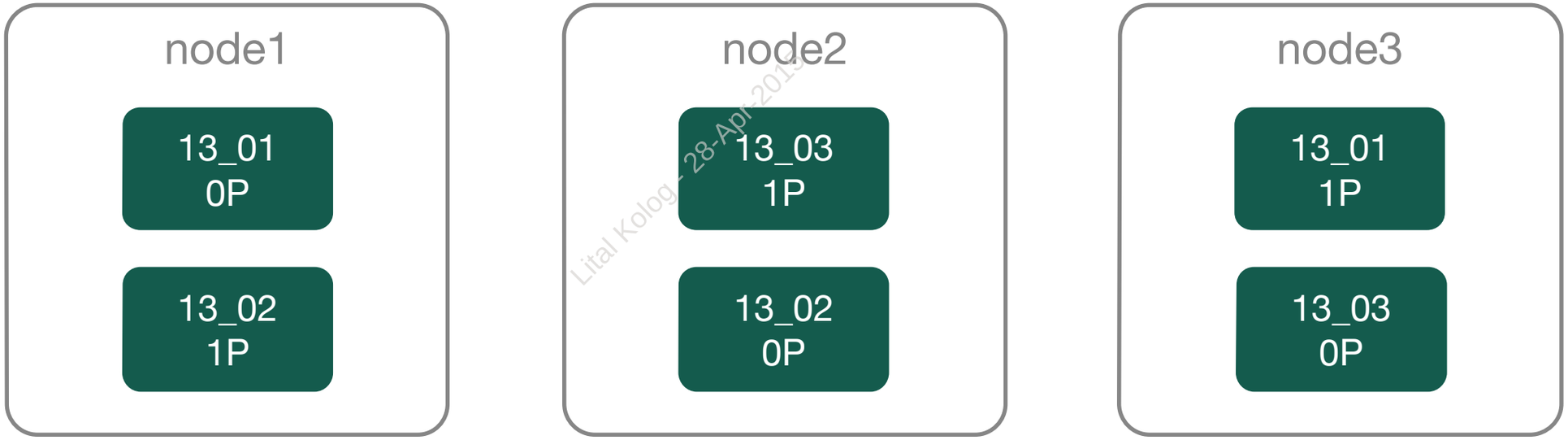


Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Time Data Flow

- An index per month/week/day
- Can adapt, change time range or number of shards for "future" indices
- Search on specific indices (*/13_01,13_02/_search*)



Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Time Data Flow

- Index per timeframe
  - Aliases simplifies it

```
curl -XPOST 'localhost:9200/_aliases' -d '{
  "actions" : [
    {
      "add" : {
        "index" : "2013_03",
        "alias" : "last_2_months",
      },
      “remove” : {
        "index" : "2013_01",
        "alias" : "last_2_months"
      }
    }
  ]
}'
```

Lita Cho - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Time Data Flow

- Index per timeframe
  - “Old” indices can be optimized

```
curl -XPOST 'localhost:9200/12_01/_optimize?max_num_segments=1'
```

  - “Old” indices can be moved

```
curl -XPOST ‘localhost:9200/2012_01/_settings’ -d ‘{
    “index.routing.allocation.include.tag” : “not_so_strong_box”
}’
```

  - Really old indices can be closed (or removed)

```
curl -XDELETE ‘localhost:9200/12_01’
```

    - Much more lightweight compared to deleting docs from an existing index, its just deleting a bunch of files compared to merging out deleted docs

Yaal Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Moving To Production

Lital Kolog - 28-Apr-2015

elastic

Installation

- Manual
  - Download, extract and set the relevant configuration
- deb & rpm Package
  - Complete package, including: user, init script, proper file system layout, etc.
- Chef
  - Cookbooks for Chef are available, which allows for cluster wide synchronized configuration, convenient settings of options, HTTP proxy installation, etc.
- Puppet
  - Modules for puppet are available, centralized config

Lital Kobiy - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Configuration

- Almost all configuration options have sensible defaults.
- There are some that needs to be reviewed and set when going into production

cluster.name	Give the cluster a unique name, simpler to uniquely identify a cluster deployment
node.name	Nodes get memorable names, but it might make sense to give them application related names
*path.**	Configure paths location, specifically path.data (to not exits “within” the installation)

LinuxKolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Configuration

discovery.zen.minimum_master_nodes	Set to a higher value, in general, you want *N/2 + 1* where *N* is the number of eligible master nodes
gateway.recover_after_nodes *recover_after_time*	Starts the recovery post full cluster restart when enough nodes are available in the cluster. Increases reuse of "local" available data, and reduces shards movement post full restart
gateway.expected_nodes	When *recover_after_time* is set, if the number of expected nodes is reached, recovery will start regardless of the *recover_after_time* setting
discovery.zen.ping.multicast.enabled *unicast.hosts*	Prefer using unicast discovery (more production resiliency for multicast "noise" from "external" nodes) Don't forget to disable multicast in this case

Digital Nomad - 28-April-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Memory

- Make sure to set *ES_HEAP_SIZE*
  - Sets the JVM memory
  - Recommended setting is ~1/2 of the machine memory
  - The rest of the memory is not going to waste!
    - The OS will automatically use it for file system cache
  - Prefer not to allocate more than *30gb*
    - In that case, the JVM can compress pointers from 64bit to 32bit
- Make sure the process does not swap
  - Setting *bootstrap.mlockall* to true in the config will cause it not to swap
    - Use along with *ulimit -l unlimited*

Lital Kolog - 28-Apr-2016

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Memory

- Big (JVM) memory "users"
  - Field Data
    - Field values are loaded to memory when sorting, faceting, or using scripting with "doc" prefix
    - Needed to be loaded to provide fast execution times
    - Make sure not to facet on a field that is analyzed, has many possibly values
    - Can be monitored using node stats API
    - Parent / Child
      - In order to implement it, ids are loaded to memory to provide fast joins

Lital Kones 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Memory

- Big (JVM) memory users
  - Lucene index, though highly optimized, still loads part of the inverted index to memory
  - Usually only applicable when there are many shards on a node
  - Bloom filters can have significant memory footprint, can be disabled (disabled by default since 1.4.0)
- OutOfMemoryError
  - Thrown when the JVM runs out of memory
  - Monitor it in the logs, if it happens, there are memory problems and action must be taken
    - Increase *ES_HEAP_SIZE*, but consider the *1/2* and less than *30gb* rule, better to add more nodes and not break those

Lital Kelog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

File Descriptors

- Increase the default file descriptors allocated to the process (default is almost always too low!)
  - File descriptors are both open files and sockets
- Set it to *64,000*, if constrained, *32,000*
- Tricky to set it (actually being applied)
  - On linux, watch out for */etc/security/limits.conf*

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

File Descriptors

```
# all nodes process info
curl ‘localhost:9200/_nodes/process’
```

```
{
  "ok" : true,
  "cluster_name" : "elasticsearch",
  "nodes" : {
    "3Kyh4VD3TpiShxL_YT_6Tg" : {
      ...
      "process" : {
        "refresh_interval" : 1000,
        "id" : 12208,
        "max_file_descriptors" : 64000
      }
    }
  }
}
```

Vital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

File Descriptors

```
# all nodes process stats
curl ‘localhost:9200/_nodes/stats/process'
```

```
{
  "cluster_name" : "elasticsearch",
  "nodes" : {
    "3Kyh4VD3TpiShxL_YT_6Tg" : {
      "timestamp" : 1349727139898,
      ...
      "process" : {
        "timestamp" : 1349727139899,
        "open_file_descriptors" : 261,
        ...
      }
    }
  }
}
```

Karl Kolog - 21 Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Hardware

- Prefer "medium" and/or "large" boxes
  - Small boxes usually end up not being economical
- As much memory as economically feasible
  - You can never have too much file system cache
  - Might be required for things like faceting/sorting
  - Can use more for filter cache
- CPUs and Cores
  - A single node is highly concurrent, single instance will easily make use of all the cores in the machine
  - Fast CPUs will mainly help with indexing, and search (filters, faceting, and sorting)

Lital Kolog - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Hardware

- Disk
  - Fast disks makes a difference when searching and indexing
    - Consider SSDs, might actually be economically attractive
  - Consider RAIDing mainly as a performance improvement, use replicas for fault tolerance
- Network
  - As fast as possible
  - Certain network heavy operations can be throttled (recovery, merging - with network drives)

Lital Kolodny 23-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Hardware

- AWS
  - Prefer large instances
    - Minimum recommended type is *m1.xlarge*
    - Suffers less from noisy neighbors
  - When using *EBS*
    - Prefer instances with *provisioned IOPS*

Lital Kolog - 28 Apr 2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Hardware

- Virtualization
  - When running multiple virtualized instances on the same box use shard allocation to make sure copies are not allocated on the same node

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring

- Cluster Health
  - A simple API to gauge the cluster health with 3 values:
    - green: all shards are allocated
    - yellow: all primary shards are allocated, some replicas are not allocated or are being allocated
    - red: not all primary shards are allocated
  - Note, under “red” health cluster is still functional
    - Might have 100 indices, but only 1 is “red”, use level set to indices to see health value per index
    - Even a red index still serves search requests, resulting in partial results (not including the missing shards) including an indication in the response

Lilia Blog - 26-03-201

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring

```
curl 'localhost:9200/_cluster/health'
```

```
{
  "cluster_name" : "elasticsearch",
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 3,
  "number_of_data_nodes" : 3,
  "active_primary_shards" : 5,
  "active_shards" : 10,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0
}
```

Digital Kology - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nodes Info / Stats

- Two types of node monitoring APIs:
  - Info: Static information that does not change
  - Stats: Dynamic values that change over time
  - For example, *max_file_descriptors* in info, while *open_file_descriptors* in stats
- Several types:
  - *os*, *process*, *jvm*, *thread_pool*, *network*, *http*
  - Use
    - */_nodes*
    - */_nodes/process*
    - */_nodes/process,jvm*

Will return all types

Lital Kozlo - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Nodes Info / Stats

- When no nodes is provided, returns for all nodes
  - */_nodes/process*

- Can specify one or more nodes, identified by:
  - id: */_nodes/3Kyh4VD3TpiShxL_YT_6Tg/process*
  - name: */_nodes/node_name1,node_name2/process*
  - IP/host: */_nodes/10.0.0.3,10.0.0.4/process*
  - Node Attributes: */_nodes/rack:2*
  - All values support wildcards

Lital Kobig - 28-Aug-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring JVM

- A quick detour to JVM heap ergonomics

Heap	Young	Survivor	Old

- Heap is broken down to:
  - Young generation
    - New objects are allocated here, assumes many new objects are short lived can can be quickly collected
    - Parallel garbage collector, runs frequently
  - Old generation
    - Old objects are promoted from young generation
    - Concurrent Mark & Sweep collection, has stop the world phases in it

Litan Ilodog - 28-Apr-2013

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring JVM

- elasticsearch works hard to work nicely with the JVM garbage collector
  - Simple default settings (see *elasticsearch.in.sh*)
  - Rarely need to change them, except for *ES_HEAP_SIZE*
  - Working hard includes
    - Reduce the amount of young garbage generated
      - For example, slice and dice network buffers
    - Old caches are being reused so the old generation garbage collector will not need to collect them

Lital Kuller - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring JVM

```
curl 'localhost:9200/_nodes/jvm'
```

```
{
  "nodes" : {
    "ZyWiUo9nSO21KXt_l8_p7A" : {
      ...
      "jvm" : {
        ...
        "version" : "1.7.0_55",
        "mem" : {
          "heap_init" : "256mb",
          "heap_init_in_bytes" : 268435456,
          "heap_max" : "1015.6mb",
          "heap_max_in_bytes" : 1065025536,
          ...
          "direct_max" : "1015.6mb",
          "direct_max_in_bytes" : 1065025536
        }
      }
    }
  }
}
```

The java version, make sure you are running a recent version java, many bugs are fixed constantly!

The maximum heap size, should match *ES_HEAP_SIZE*

Advanced: the direct memory maximum memory (used for native memory network buffers for example)

Lital Kolog - 22-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring JVM

```
curl ‘localhost:9200/_nodes/stats/jvm'
```

```
{
  "nodes": {
    "ItkvHYXsRVKyQnfMbUZxHQ": {
      "timestamp": 1391318893829,
      "jvm": {
        "timestamp": 1391318893829,
        "uptime_in_millis": 1270510,
        "mem": {
          "heap_used_in_bytes": 164154776,
          "heap_used_percent": 15,
          "heap_committed_in_bytes": 266272768,
          "heap_max_in_bytes": 1065025536,
          "non_heap_used_in_bytes": 50566840,
          "non_heap_committed_in_bytes": 81063936,
```

Lal Kallog - 28-Apr-2015

How much heap is currently used (note, includes “garbage”) a value close to heap max over time indicates memory pressure

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring JVM

```
"pools": {
  "young": {
    "used_in_bytes": 14615120,
    "max_in_bytes": 69795840,
    "peak_used_in_bytes": 17432576,
    "peak_max_in_bytes": 69795840
  },
  "survivor": {
    "used_in_bytes": 344208,
    "max_in_bytes": 8716288,
    "peak_used_in_bytes": 2162688,
    "peak_max_in_bytes": 8716288
  },
  "old": {
    "used_in_bytes": 51362600,
    "max_in_bytes": 986513408,
    "peak_used_in_bytes": 51362600,
    "peak_max_in_bytes": 986513408
  }
}
```

How much memory is used by young generation

How much memory is used by survivor generation

How much memory is used by old generation

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring JVM

```
"threads": {
  "count": 57,
  "peak_count": 61
}
```

How many threads are currently active

What is the peak number of threads created

```
"gc": {
  "collectors": {
    "young": {
      "collection_count": 1205,
      "collection_time_in_millis": 3730
    },
    "old": {
      "collection_count": 3,
      "collection_time_in_millis": 13
    }
  }
}
```

Lital Kolog - 28-Apr-2015

Young generation garbage collection count and time (expect high numbers here, its ok!)

Old generation garbage collection count and time (frequent collection here mean memory pressure)

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Thread Pools

- Threads are used to handle actions on the cluster
  - For example, *index*, *search*, *get*, *bulk*, ...
  - But also handle background jobs like refreshing the index and merging
- Thread pools are broken down into types, important ones include:
  - index: Handles *index*/*delete*/*update* requests on the shard level
  - search: Handles search requests on the shard level
  - bulk: Handles bulk execution on the shard level

Lital Kohen 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Thread Pools

- There are several types of thread pools, though we recommend using fixed
  - A fixed set of threads, requests are served if a thread is available, if not, they are queued
    - *size*: Sets the number of threads in the pool
    - *queue_size*: The pending work queue size, if set, any incoming requests are rejected (*503* status code)

Lital Kolos 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Thread Pools

- By default, all action thread pool types are fixed
  - Consider setting the index, bulk, search thread pools to fixed, with a specific queue size
    - Setting size should be a factor of the number of cores, but not 1, since there is IO involved as well
    - Setting the *queue_size* to an acceptable level to not fail requests immediately when there aren't available threads, but still not have the backlog unbounded
- It's also possible to bound the thread pools by specifying the number of processors on the box (using the *processors* setting)
  - Some OS's don't report the right number of processors, but also, huge boxes with too many processors might cause an "explosion" of threads

Lital Kotler - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Thread Pools

```
threadpool:
  index:
    type: fixed
    size: 8
    queue_size: 200
  search:
    type: fixed
    size: 24
    queue_size: 1000
  bulk:
    type: fixed
    size: 8
    queue_size: 50
```

Single *index*/*delete*/*update* operations

Remember, single shard bulk executes multiple requests (*index*/*delete*/*update*)

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Thread Pools

```
curl 'localhost:9200/_nodes/thread_pool'
```

```
{
   "nodes": {
      "ItkvHYXsRVKyQnfMbUZxHQ": {
         "thread_pool": {
            "index": {
               "type": "fixed",
               "min": 4,
               "max": 4,
               "queue_size": "200"
            },
            ...
         }
      }
   }
}
```

Name of the thread pool

Thread pool configuration

Lukolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Thread Pools

```
curl ‘localhost:9200/_nodes/stats/thread_pool'
```

```
{
   "nodes": {
      "ItkvHYXsRVKyQnfMbUZxHQ": {
         "thread_pool": {
            "index": {
               "threads": 4,
               "queue": 0,
               "active": 0,
               "rejected": 0,
               "largest": 4,
               "completed": 941
            },
            ...
         }
      }
   }
}
```

Name of the thread pool

Total number of threads

Current pending queue size

Current active threads

Number of rejected executions
(for example, due to queue_size limit breached)

Lital Kolog 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Field Data

- Field Data can potentially use a lot of memory
  - Used when sorting or faceting on a field, or using it in a script (with doc notation)
- Can't really get around it
  - The values are needed in memory to provide acceptable response times
  - If they are not available in memory and are requested, they are loaded from the index
- Important to monitor it to see its usage pattern
  - Good indication that new nodes need to be added to the cluster

Vital Kolosov, 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Field Data

```
curl ‘localhost:9200/_nodes/stats'
```

```
{
  "nodes" : {
    "LmiwPxIZTy2pZitlONkltg" : {
      ...
      "indices" : {
         ...
         "fielddata": {
            "memory_size": "150mb",
            "memory_size_in_bytes": 157286400,
            "evictions": 0
         }
      }
    }
  }
}
```

field data size

Stackology - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Field Data

```
curl 'localhost:9200/_nodes/stats?fields=created_on'
```

```
{
  "nodes" : {
    "LmiwPxIZTy2pZitlONkltg" : {
      ...
      "indices" : {
         ...
         "fielddata": {
          "memory_size": "150mb",
          "memory_size_in_bytes": 157286400,
          "evictions": 0,
          "fields": {
            "created_on": {
              "memory_size": "150mb",
              "memory_size_in_bytes": 157286400
            }
          }
         }
      }
    }
  }
}
```

Field name

Lital Kolog - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring Store Size

```
{
  "nodes" : {
    "LmiwPxIZTy2pZitlONkltg" : {
      ....
      "indices" : {
        "store" : {
          "size" : "5.3mb",
          "size_in_bytes" : 55574528,
          "throttle_time" : "0s",
          "throttle_time_in_millis" : 0
        },
        "docs" : {
          "count" : 48721,
          "deleted" : 0
        },
        ...
```

Total size used on the node by all the shards allocated on it

Total number of docs on the node

Lifa Kolbg 22-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring HTTP

- Strongly prefer long live HTTP connections from clients
  - Short live connections can cause excessive load needing to handle opening and closing sockets
  - Can run out of available sockets (require low level configuration of the OS)
  - If not possible, consider a dedicated proxy/load-balancer to maintain long HTTP lived connections to the cluster, while handling the short lived connections
    - *nginx*, for example, makes this simple (configuration can be found online)

Uzal Kolos 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Monitoring HTTP

```
curl ‘localhost:9200/_nodes/stats/http'
```

```
{
  "cluster_name" : "elasticsearch",
  "nodes" : {
    "ZyWiUo9nSO21KXt_l8_p7A" : {
      "timestamp" : 1349740794114,
      ...
      "http" : {
        "current_open" : 50,
        "total_opened" : 19456
      }
    }
  }
}
```

Lital Kolog - 28-Apr-2015

If this keeps on increasing, persistent connections are probably not used

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Indices Stats

- Indices level statistics
  - Up until now, we saw info/stats on the node level
- Index level statistics provide
  - Information on the index level, regardless of node allocation
  - Optional breakdown per shard by providing a level parameter with shards value
- Several types
  - *docs*, *store*, *indexing*, *search*, *get*, *merge*, *refresh*, *flush*, *warmer*

Lital Kolko - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited

Indices Info & Stats

```
# by default, docs & store are returned
curl localhost:9200/index1/_stats

# can be executed on multiple indices
curl localhost:9200/index1,index2/_stats

# return *just* search statistics
curl 'localhost:9200/index1/_stats/search

# return *just* the search & indexing statistics
curl 'localhost:9200/index1/_stats/search,indexing
```

Lital Kologi - 28-Apr-2015

Copyright Elastic 2015 Copying, publishing and/or distributing without written permission is strictly prohibited