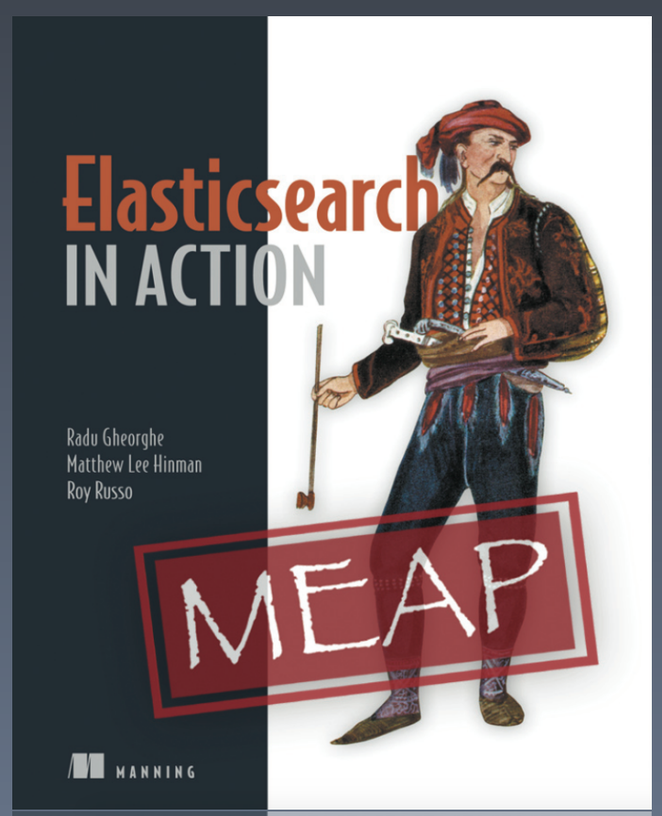
Introduction To ElasticSearch

Real-Time Search and Analytics

Roy Russo, DevNexus, 2015

Who Am I

- Roy Russo
- VP Engineering, Predikto
- Co-Author - Elasticsearch in Action
  - Due ~April 2015.
- ElasticHQ.org
- Other (*)



* Silverpop, JBoss, AltiSource Labs

Why Am I Here?

- What is Search
- What is Elasticsearch
- Real-World Use
- Scale Out
- Interacting with Elasticsearch

Search is about filtering information and determining relevance.

How does a Search Engine

Select * FROM make WHERE name LIKE ‘%Tesla%’

Search Engines use Magic

It's FM!

Where Magic == Inverted Index

Inverted Index

- Take some documents
- Tokenize them
- Find unique tokens
- Map tokens to documents

	apple	oranges	peach
Document 1			
Document 2			
Document 3			
Document 4			
Document 5			
Document 6			

Inverted Index

Search for “apple peach”

	apples	oranges	peach
Document 1			
Document 2			
Document 3			
Document 4			
Document 5			
Document 6			

Relevance

- How many tokens per document?
- How many tokens relative to the number of total tokens in the document?
- What is the frequency of token across all documents?

Relevance in Elasticsearch

- At Search Time
- At Index Time
- Term Frequency
  - Term / Document
- Inverse Document Frequency (IDF)
  - Term / All Documents in the collection
- Field-Length Norm

What is Elasticsearch?

Elasticsearch is…

- Search and Analytics engine
- Document Store
  - Every field is indexed/searchable
- Distributed

What Elasticsearch is not

- Key-Value Store
  - Redis, Riak
- Column Family Store
  - C*, HBase
- Graph Database
  - Neo4J

ElasticSearch in a Nutshell

- Based on Apache Lucene
- Distributed
- Document-Oriented
- Schema free
- HTTP + JSON
- (Near) Real-time search
- Ecosystem
  - Hosting, Monitoring Apps, Clients (SDK)

Where can I get it?

- Free and Open Source
- https://www.elastic.co/
- https://github.com/elastic/elasticsearch
- Backed by a Company, Elastic
  - Training
  - Support
  - Auth/AuthZ
  - Marvel for Monitoring

How do I run it?

- Download it
  - https://www.elastic.co/downloads
- bin/elasticsearch
- http://localhost:9200


```
{
  status: 200,
  name: "Tesla",
  cluster_name: "elasticsearch_royrusso",
  version: {
    number: "1.4.2",
    build_hash: "927caff6f05403e936c20bf4529f144f0c89fd8c",
    build_timestamp: "2014-12-16T14:11:12Z",
    build_snapshot: false,
    lucene_version: "4.10.2"
  },
  tagline: "You Know, for Search"
}
```

Elasticsearch requires Java. *

* You have 5 seconds to whine about it and then shutup.

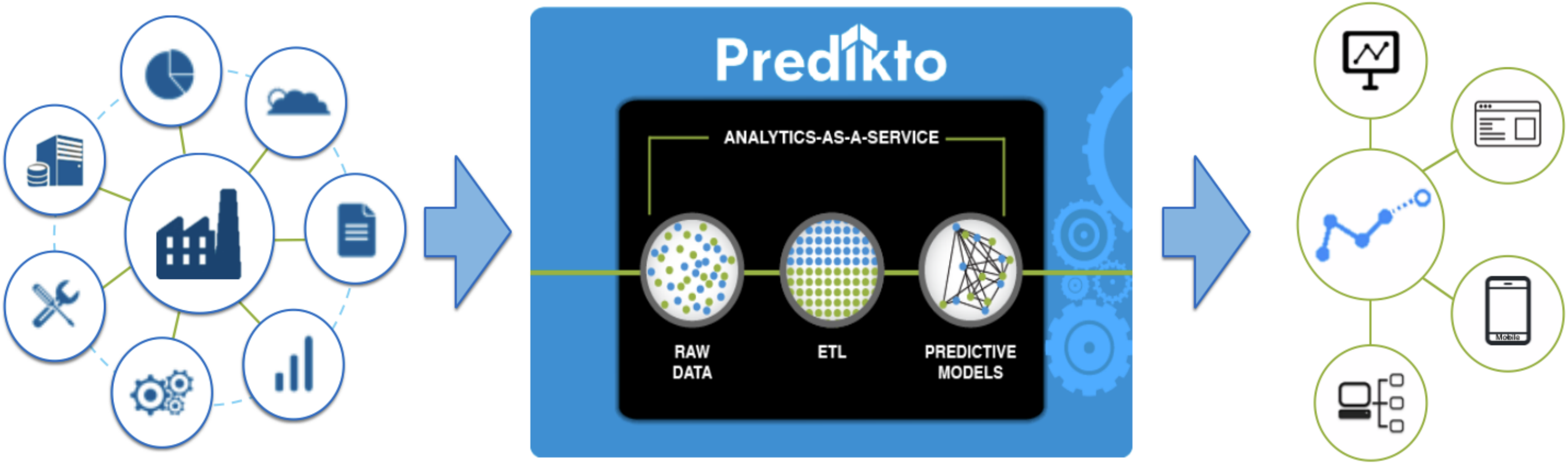
Some Use-Cases

ElasticSearch for Centralized

- Logstash + ElasticSearch + Kibana (ELK)
- Well… and then there's Loggly

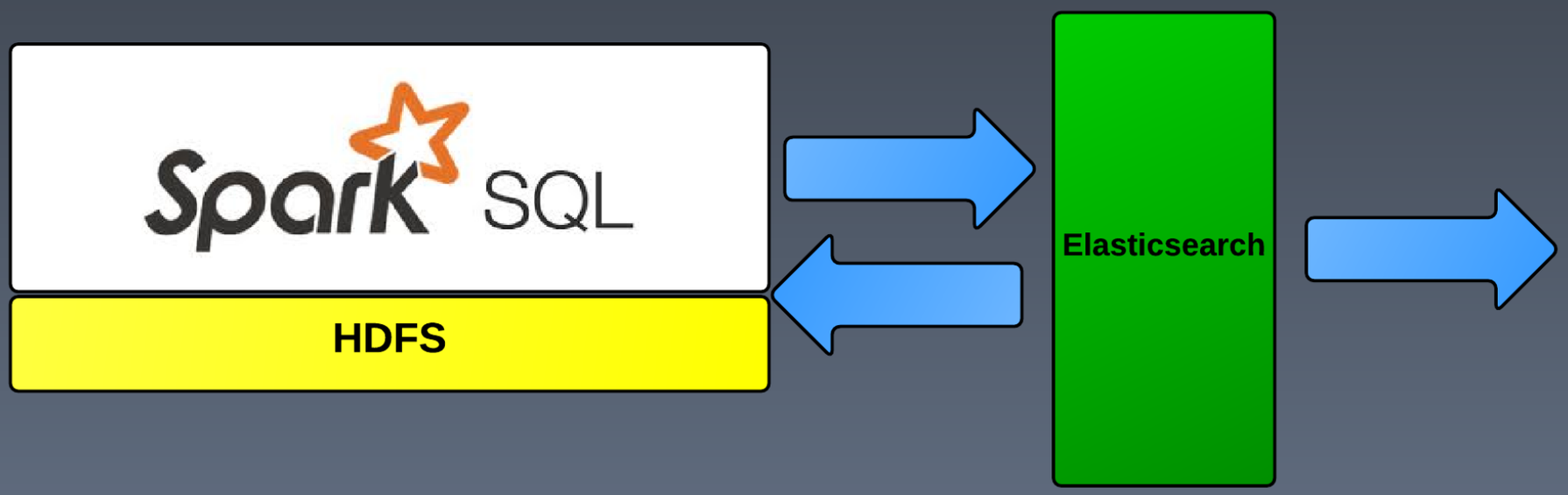
"Netflix is a Log generating company that happens to stream movies"

Elasticsearch at Predikto



Elasticsearch at Predikto

- Write From Spark to Elasticsearch
- Query from Spark to Elasticsearch
- Visualize



Widely Used

LinkedIn

NETFLIX

stackoverflow

TARGET

OpenTable

Wikipedia
The Free Encyclopedia

Atlassian

Based on Apache Lucene

- Free and Open Source
- Started in 1999
- Created by Doug Cutting
- What's it do?
  - Tokenizing
  - Locations
  - Relevance scoring
  - Filtering
  - Text search
  - Date Parsing

Elasticsearch is a Document Store

Document Store

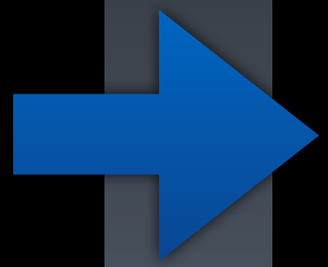
- Like MongoDB and CouchDB
- Document DBs:
  - JSON documents
  - Collections of key-value collections
  - Nesting
  - Versioned

What is a document?

```
{
    "genre": "Crime",
    “language": "English",
    "country": "USA",
    "runtime": 170,
    "title": "Scarface",
    "year": 1983
}
```

Modeled in JSON

```
{
    "genre": "Crime",
    "language": "English",
    "country": "USA",
    "runtime": 170,
    "title": "Scarface",
    "year": 1983
}
```



```
{
    "_index": "imdb",
    "_type": "movie",
    "_id": "u17o8zy9RcKg6SjQZqQ4Ow",
    "_version": 1,
    "_source": {
        "genre": "Crime",
        "language": "English",
        "country": "USA",
        "runtime": 170,
        "title": "Scarface",
        "year": 1983
    }
}
```

Schema-Free

- Dynamic Mapping
  - Elasticsearch guesses the data-types (string, int, float…)

```
"imdb": {
    "movie": {
        "properties": {
            "country": {
                "type": "string“,
                “store”:true,
                “index”:false
            },
            "genre": {
                "type": "string“,
                "null_value" : "na“,
                “store”:false,
                “index:true
            },
            "year": {
                "type": "long"
            }
        }
    }
}
```

Elasticsearch is Distributed

Terminology

MySQL	Elasticsearch
Database	Index
Table	Type
Row	Document
Column	Field
Schema	Mapping
Index	(Everything is indexed)
SQL	Query DSL

- Cluster: 1..N Nodes w/ same Cluster Name
- Node: One ElasticSearch instance (1 java proc)
- Shard = One Lucene instance
  - 0 or more replicas

High Availability

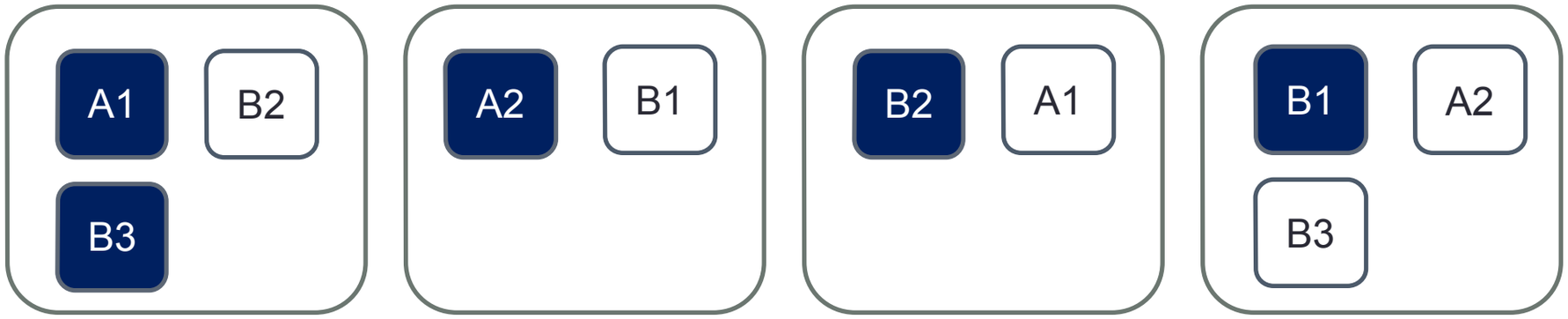
- No need for load balancer
- Different Node Types
- Indices are Sharded
- Replica shards on different Nodes
- Automatic Master election & failover

About Indices / Shards

```
$ curl -XPUT 'http://localhost:9200/twitter/' -d '{
"settings" : {
    "index" : {
        "number_of_shards" : 3,
        "number_of_replicas" : 2
    }
}
}'
```

Cluster Topology

4 Node Cluster



Index A: 2 Shards & 1 Replica
Index B: 3 Shards & 1 Replica

Discovery

- Nodes discover each other using multicast.
  - Unicast is an option
- Each cluster has an elected master node
  - Beware of split-brain

```
discovery.zen.ping.multicast.enabled: false
discovery.zen.ping.unicast.hosts: ["host1", "host2:port", "host3"]
```

Nodes

- Master node handles cluster-wide (Meta-API) events:
  - Node participation
  - New indices create/delete
  - Re-Allocation of shards
- Data Nodes
  - Indexing / Searching operations
- Client Nodes
  - REST calls
  - Light-weight load balancers

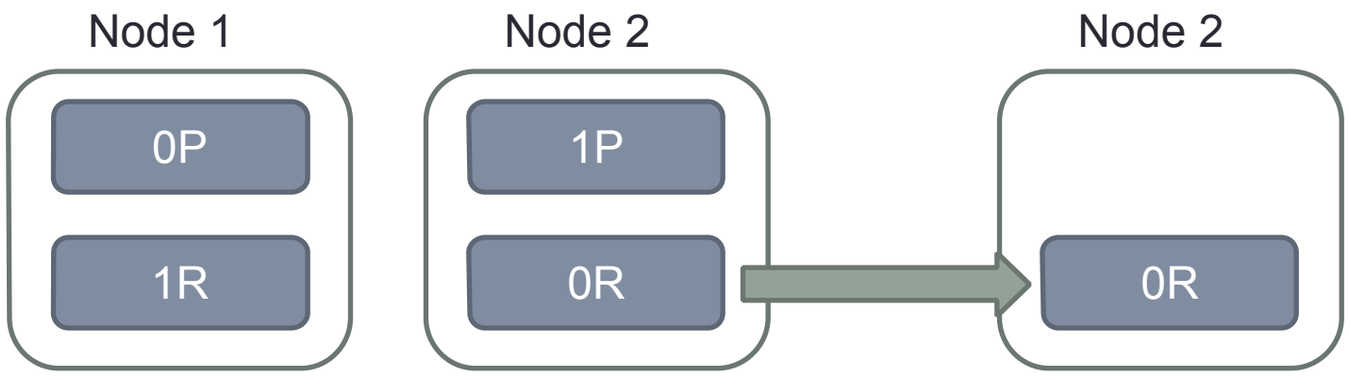
node.data | node.master

The Basics - Shards

- Primary Shard:
  - First time Indexing
  - Index has 1..N primary shards (default: 5)
  - # Not changeable once index created
- Replica Shard:
  - Copy of the primary shard
  - # Can be changed later
  - Each primary has 0..N replicas
  - HA:
    - Promoted to primary if primary fails

Shard Auto-Allocation

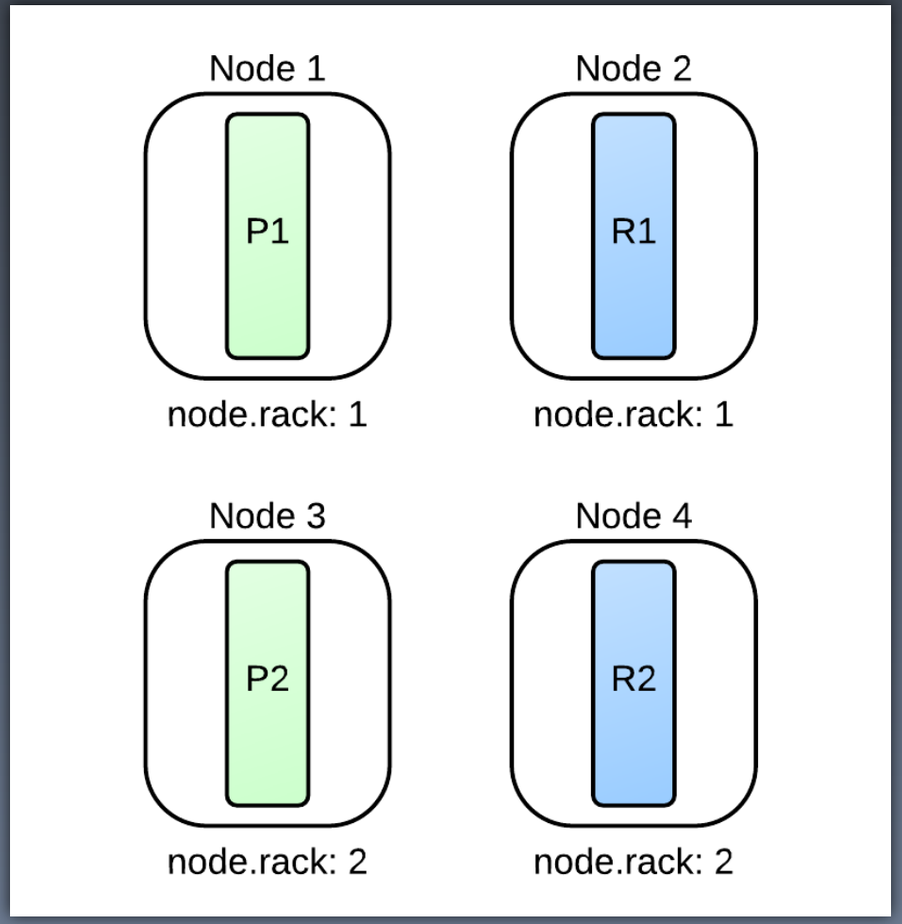
Add a Node: Shards Relocate



- Shard Phases:
  - Unassigned
  - Initializing
  - Started
  - Relocating

Allocation Awareness

- Shard Allocation Awareness
  - cluster.routing.allocation.awareness.attributes: rack
  - Shards RELOCATE to even distribution
  - Primary & Replica will NOT be on the same rack



Cluster State

- Cluster State
  - Node Membership
  - Indices Settings
  - Shard Allocation Table

```
cURL -XGET http://localhost:9200/_cluster/state?pretty=1'
{
    "cluster_name" : "elasticsearch_royrusso",
    "version" : 27,
    "master_node" : "s3fpXfPKSFeUqo1MYZxSng",
    "blocks" : { },
    "nodes" : {
        "s3fpXfPKSFeUqo1MYZxSng" : {
            "name" : "Bulldozer",
            "transport_address" : "inet[localhost/127.0.0.1:9300]",
            "attributes" : { }
        }
    },
    "metadata" : {
        "templates" : {
            "logging_index_all" : {
                "template" : "logstash-09-*",
                "order" : 1,
                "settings" : {
                    "index" : {
                        "number_of_shards" : "2",
                        "number_of_replicas" : "1"
                    }
                },
                "mappings" : {
                    "date" : {
                        "store" : false
                    }
                }
            },
            "logging_index" : {
                "template" : "logstash-*",
                "order" : 0,
                "settings" : {
                    "index" : {
                        "number_of_shards" : "2",
                        ""number_of_replicas"" : "1"
                    }
                },
                "mappings" : {
                    ""date"" : {
                        ""store"" : true
                    }
```

Talking to Elasticsearch

REST

- HTTP Verbs: GET, POST, PUT, DELETE
- JSON
- _cat API

```
% curl '192.168.56.10:9200/_cat/health?v&ts=0'
cluster status nodeTotal nodeData shards pri relo init unassign
foo     green          3        3      3   3    0    0        0
```

The API

- Document
- Cluster
  - Node
- Index
- Search

Create a Document

```
curl -XPOST 'http://127.0.0.1:9200/imdb/movie' -d '
{
    "genre": "Crime",
    "language": "English",
    "country": "USA",
    "runtime": 170,
    "title": "Scarface",
    "year": 1983
}';
```

```
{
    "_index":"imdb",
    "_type":"movie",
    "_id":"AUwGeWib1u4mCngDYT7y",
    "_version":1,
    "created":true
}
```

Of note…

```
curl -XPOST 'http://127.0.0.1:9200/imdb/movie' -d '
{
    "genre": "Crime",
    "language": "English",
    "country": "USA",
    "runtime": 170,
    "title": "Scarface",
    "year": 1983
}';
```

Auto-creates Index & Type

Auto-Gen ID

Auto-Version

```
{
    "_index":"imdb",
    "_type":"movie",
    "_id":"AUwGeWib1u4mCngDYT7y",
    "_version":1,
    "created":true
}
```

Get a Document

```
curl -XGET 'http://127.0.0.1:9200/imdb/movie/AUwGeWib1u4mCngDYT7y';
```


```
{
    "_index":"imdb",
    "_type":"movie",
    "_id":"AUwGeWib1u4mCngDYT7y",
    "_version":1,
    "found":true,
    "_source":
    {
        "genre": "Crime",
        "language": "English",
        "country": "USA",
        "runtime": 170,
        "title": "Scarface",
        "year": 1983
    }
}
```

Update a Document

```
curl -XPUT 'http://127.0.0.1:9200/imdb/movie/AUwGeWib1u4mCngDYT7y' -d '
{
    "genre": "Crime",
    "language": "English",
    "country": "USA",
    "runtime": 180,
    "title": "Scarface",
    "year": 1983
}';
```

```
{
    "_index":"imdb",
    "_type":"movie",
    "_id":"AUwGeWib1u4mCngDYT7y",
    "_version":2,
    "created":false
}
```

* More like an Upsert.

Delete a Document

```
curl -XDELETE 'http://127.0.0.1:9200/imdb/movie/AUwGeWib1u4mCngDYT7y'
```

```
{
  “found":true,
  “_index":"imdb",
  “_type":"movie",
  “_id":"AUwGeWib1u4mCngDYT7y",
  “_version":2
  }
```

You can also…

- Partial document updating
- Specify Version
- Specify ID
- Multi-Get API
- Exists API
- Bulk API

How Searching Works

- How it works:
  - Search request hits a node
  - Node broadcasts to every shard in the index
  - Each shard performs query
  - Each shard returns metadata about results
  - Node merges results and scores them
  - Node requests documents from shards
  - Results merged, sorted, and returned to client.

REST API - Search

- Free Text Search
  - URL Request
- Complex Query

```
http://localhost:9200/imdb/movie/_search?q=scar*

http://localhost:9200/imdb/movie/_search?q=scarface+OR+star

http://localhost:9200/imdb/movie/_search?q=(scarface+OR+star)+AND+year:[1981+TO+1984]
```

REST API – Query DSL

```
curl -XPOST 'localhost:9200/_search?pretty' -d '{
"query" : {
    "bool" : {
        "must" : [
                  {
                  "query_string" : {
                    "query" : "scarface or star"
                  }
                },
                {
                  "range" : {
                  "year" : { "gte" : 1931 }
                  }
                }
        ]
    }
}
}'
```

REST API – Query DSL

- Boolean Query

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

REST API – Query DSL

- Range Query
  - Numeric / Date Types
- Prefix/Wildcard Query
  - Match on partial terms
- RegExp Query
- Geo_bbox
  - Bounding box filter
- Geo_distance
  - Geo_distance_range

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

Filters

- Filters recommended over Queries
  - Better cache support

```
curl -XGET 'http://localhost:9200/my_index/events/_search?pretty=1' -d '
{
    "from" : 0,
    "size" : 0,
    "query" : {
        "terms" : {
            "message" : [ "apples"],
            "minimum_should_match" : "1"
        }
    },
    "post_filter" : {
        "terms" : {
            "userId" : [ "25476c6788ce", "g20d5470d7b4" ],
            "execution" : "or"
        }
    },
    "sort": { "eventDate": { "order": "desc" }},
    "explain" : false
}';
```

Analyzers / Tokenizers

```
curl -XPUT ‘http://localhost:9200/my_index/' -d '
{
    "settings" :
    {
       "analysis" : {
            "analyzer" : {
                "str_search_analyzer" : {
                    "tokenizer" : "keyword",
                    "filter": ["lowercase"]
                },
                "str_index_analyzer" : {
                    "tokenizer" : "substring",
                    "filter" : ["lowercase", "stop"]
                }
            },
            "tokenizer" : {
                "substring" : {
                    "type" : "edgeNgram",
                    "min_gram" : "3",
                    "max_gram" : "42",
                    "token_chars" : ["letter", "digit"]
                }
            }
```

```
curl -XPUT 'http://localhost:9200/my_index/events/_mapping' -d '
{
    "events" : {
       "properties" :
        {
            "eventId" : {"type" : "string", "store" : true, "index" : "not_analyzed" },
            "userId" : {"type" : "string", "store" : false, "index" : "not_analyzed" },
            "message" : {
                "type" : "string", "store" : false,
                "search_analyzer" : "str_search_analyzer",
                "index_analyzer" : "str_index_analyzer"
                }
       }
    }
}';
```

Tokenizers

- Whitespace
- NGram
- Edge NGram
- Letter
  - @ non-letters

Clients

- Client list: http://www.elasticsearch.org/guide/clients/
  - Java (Node) Client, JS, PHP, Perl, Python, Ruby
- Spring Data:
  - Uses TransportClient
  - Implementation of ElasticsearchRepository aligns with generic Repository interfaces

Monitoring

- BigDesk
- Kopf
- Head
- ElasticHQ
- Marvel
- Sematext SPM

Questions?