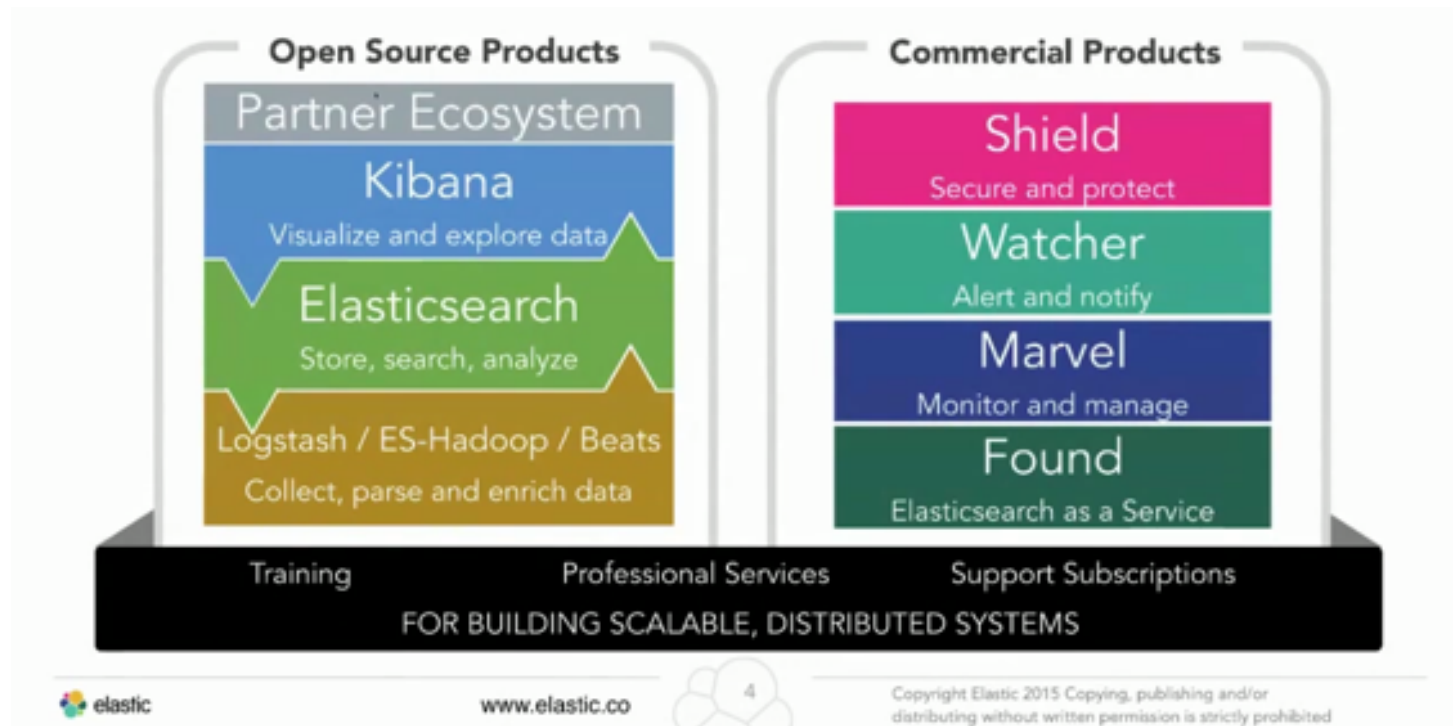




# Elasticsearch



# Elastic Products



# Elasticsearch

- ❑ Highly scalable open-source full-text search and analytics engine
- ❑ Based on Apache Lucene™
- ❑ JSON + RESTful
- ❑ Real-time, Distributed, Schema-Free, Conflict Management

# Who is using ES

- ❏ Wikipedia
- ❏ Stack Overflow
- ❏ GitHub
- ❏ SoundCloud
- ❏ The Guardian
- ❏ Mozilla
- ❏ Foursquare
- ❏ Quora ...

# Basic Concepts

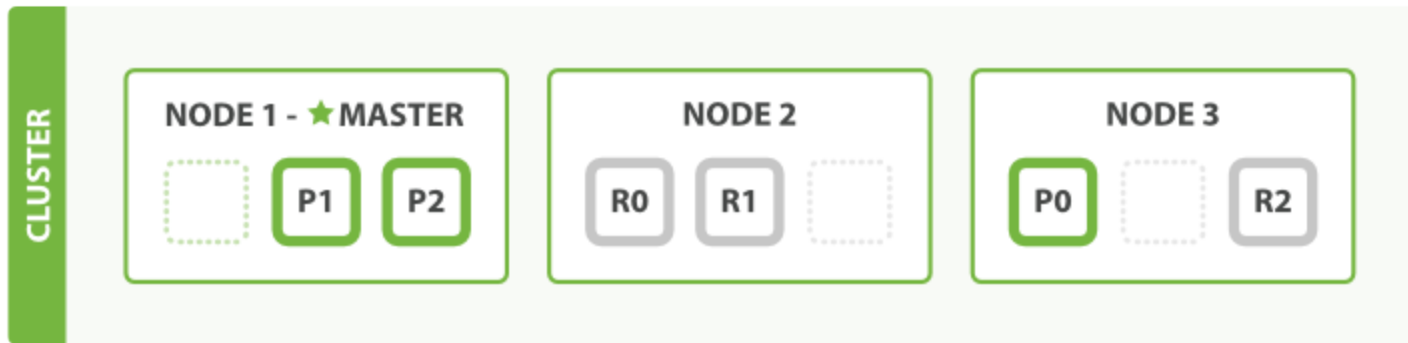
Cluster, Nodes

Index, Type, Document, Field

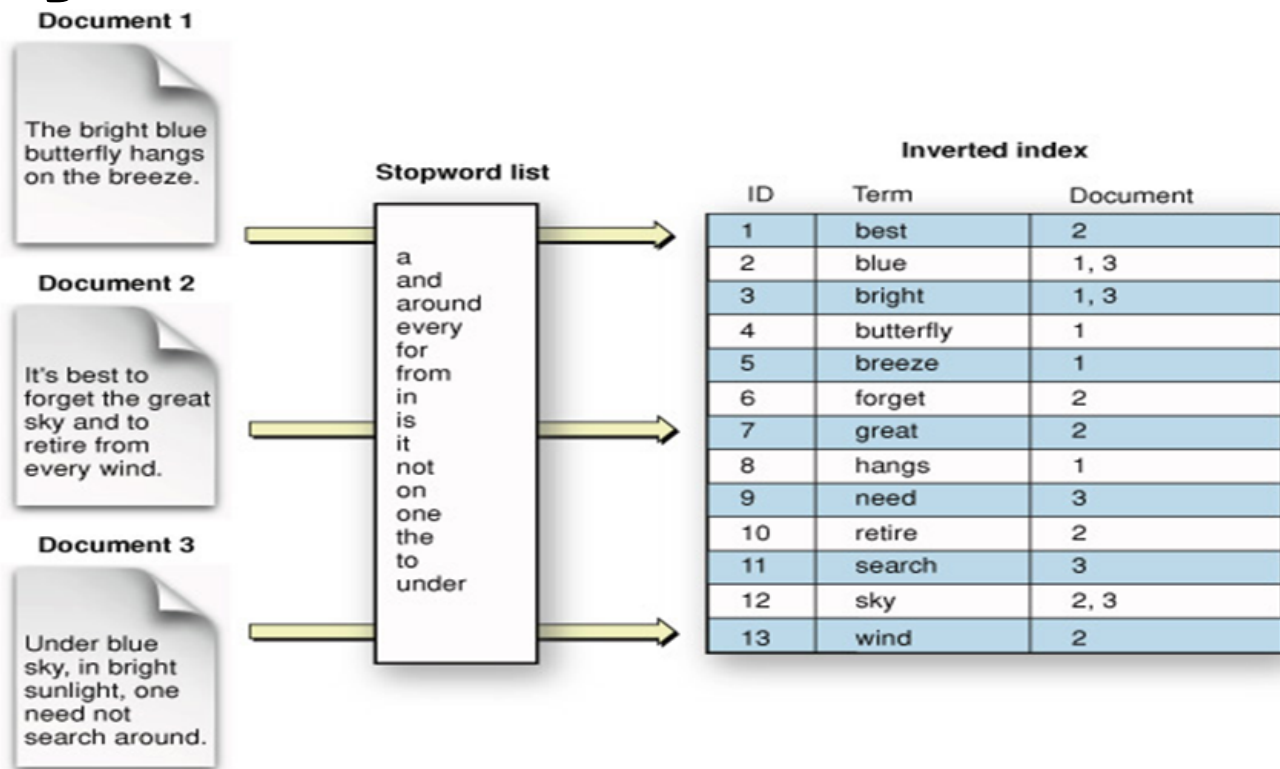
Shards & Replicas

Mapping

# Cluster



# Indexing



# Terminology

Relational DB	ElasticSearch
Database	Index
Table	Type
Row	Document
Column	Field
Schema	Mapping
Index	*Everything
SQL	query DSL



# Installing and running

```
curl -L -O http://download.elasticsearch.org/PATH/TO/VERSION.zip
```

```
unzip elasticsearch-$VERSION.zip
```

```
cd elasticsearch-$VERSION
```

Settings: config/elasticsearch.yml (cluster.name, node.name)

```
./bin/elasticsearch
```

# Info

```
curl 'http://localhost:9200/?pretty'
```

```
{
  "status" : 200,
  "name" : "katarina_dev",
  "cluster_name" : "elasticsearch_kat",
  "version" : {
    "number" : "1.7.1",
    "build_timestamp" : "2015-07-29T09:54:16Z",
    "build_snapshot" : false,
    "lucene_version" : "4.10.4"
  },
  "tagline" : "You Know, for Search"
}
```

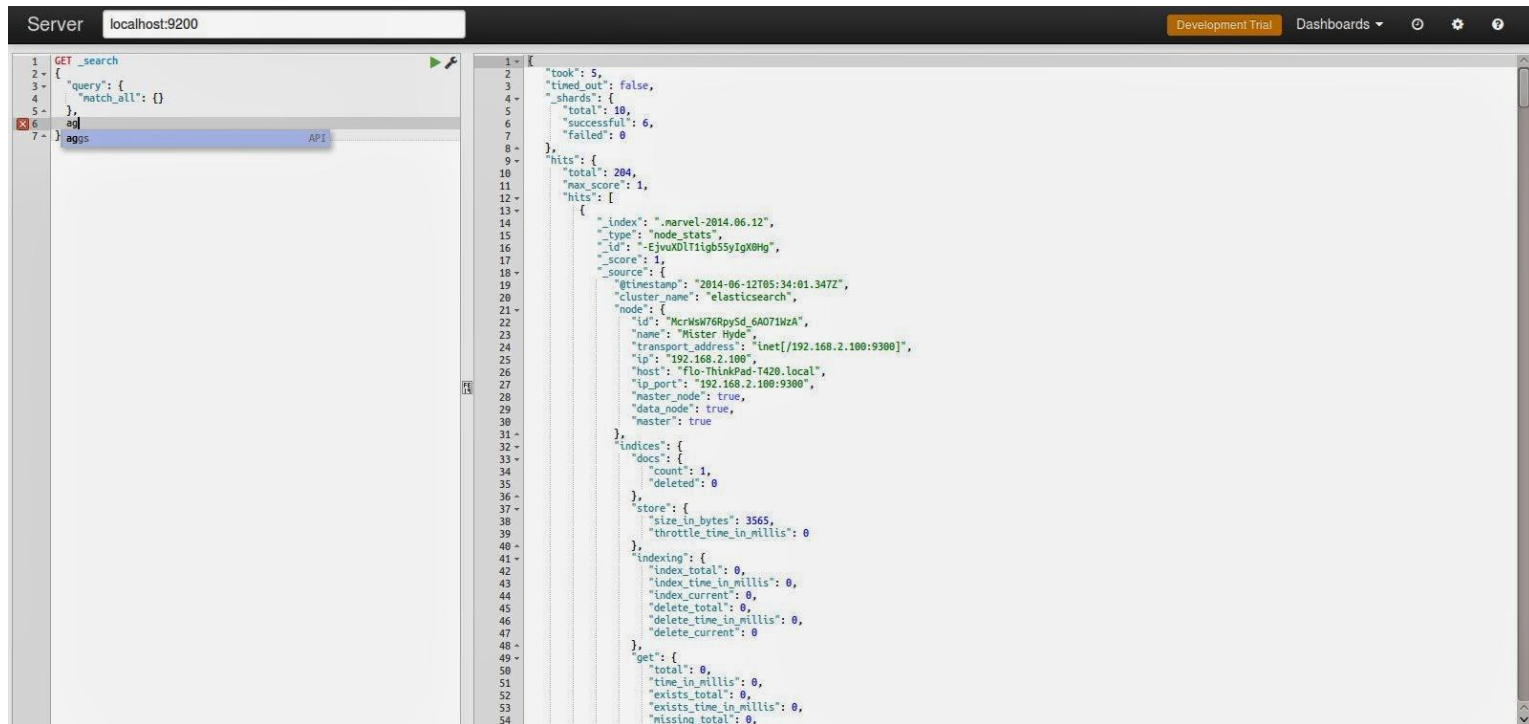
# Cluster health

GET \_cluster/health

```
{  
  "cluster_name": "elasticsearch_kat",  
  "status": "yellow",  
  "timed_out": false,  
  "number_of_nodes": 1,  
  "number_of_data_nodes": 1,  
  "active_primary_shards": 16,  
  "active_shards": 16,  
  "relocating_shards": 0,  
  "initializing_shards": 0,  
  "unassigned_shards": 16,  
  "delayed_unassigned_shards": 0,  
  "number_of_pending_tasks": 0,  
  "number_of_in_flight_fetch": 0  
}
```

# Sense

Chrome extension - a JSON aware developer console to ElasticSearch.



The screenshot displays the Sense Chrome extension interface. At the top, a 'Server' dropdown is set to 'localhost:9200'. The interface is divided into two main panels. The left panel contains a REST client with a 'GET \_search' request. The request body is a JSON object: `{ "query": { "match_all": {} } }`. The right panel shows the corresponding JSON response from ElasticSearch. The response includes metadata such as 'took', 'timed\_out', 'shards', 'total', 'successful', and 'failed'. It also contains a 'hits' array with a single document. This document is a 'node\_stats' object for the index 'marvel-2014-06-12', providing detailed information about the node's health, configuration, and performance metrics.

```
1 GET _search
2 {
3   "query": {
4     "match_all": {}
5   },
6   "aggs":
7 }

[
  {
    "took": 5,
    "timed_out": false,
    "shards": {
      "total": 10,
      "successful": 6,
      "failed": 0
    },
    "hits": {
      "total": 204,
      "max_score": 1,
      "hits": [
        {
          "_index": "marvel-2014-06-12",
          "_type": "node_stats",
          "_id": "-EjvuX0L7i1gb55yIqX8Hg",
          "_score": 1,
          "_source": {
            "@timestamp": "2014-06-12T05:34:01.347Z",
            "cluster_name": "elasticsearch",
            "node": {
              "id": "McRksh76Rpy5d_6A071WZA",
              "name": "Mister Hyde",
              "transport_address": "inet[/192.168.2.100:9300]",
              "ip": "192.168.2.100",
              "host": "Flo-ThinkPad-T420.local",
              "ip_port": "192.168.2.100:9300",
              "master_node": true,
              "data_node": true,
              "master": true
            },
            "indices": {
              "docs": {
                "count": 1,
                "deleted": 0
              },
              "store": {
                "size_in_bytes": 3565,
                "throttle_time_in_millis": 0
              },
              "indexing": {
                "index_total": 0,
                "index_time_in_millis": 0,
                "index_current": 0,
                "delete_total": 0,
                "delete_time_in_millis": 0,
                "delete_current": 0
              },
              "get": {
                "total": 0,
                "time_in_millis": 0,
                "exists_total": 0,
                "exists_time_in_millis": 0,
                "missing_total": 0
              }
            }
          }
        }
      ]
    }
  }
]
```

# Analyzer

```
curl -X POST "localhost:9200/3a2ilati" -d '
```

[illegible]

# Insert data

```
PUT /website/blog/123
```

```
{  
  "title": "My first blog entry",  
  "text": "Just trying this out...",  
  "date": "2015/09/09"  
}
```

```
response:
```

```
{  
  "_index": "website",  
  "_type": "blog",  
  "_id": "123",  
  "_version": 1,  
  "created": true  
}
```

# Get data

GET /website/blog/123?pretty

```
{
  "_index" :   "website",
  "_type" :    "blog",
  "_id" :      "123",
  "_version" : 1,
  "found" :    true,
  "_source" :  {
    "title": "My first blog entry",
    "text":  "Just trying this out...",
    "date":  "2015/09/09"
  }
}
```

part of a document: GET /website/blog/123?\_source=title,text

just source: GET /website/blog/123/\_source

# Multi-index, Multitype

`/_search`

Search all types in all indices

`/gb/_search`

Search all types in the `gb` index

`/gb,us/_search`

Search all types in the `gb` and `us` indices

`/g*,u*/_search`

Search all types in any indices beginning with `g` or beginning with `u`

`/gb/user/_search`

Search type `user` in the `gb` index

`/gb,us/user,tweet/_search`

Search types `user` and `tweet` in the `gb` and `us` indices

`/_all/user,tweet/_search`

Search types `user` and `tweet` in all indices



# Search Lite

*query-string* version that expects all its parameters to be passed in the query string

```
GET /website/blog/_search
```

```
GET /website/blog/_search?q=title:elasticsearch
```

# Search with Query DSL

The domain-specific language (DSL) is specified using a JSON request body

GET /megacorp/employee/\_search

```
{  
  "query" : {  
    "match" : {  
      "last_name" : "Smith"  
    }  
  }  
}
```

# Filters

GET /megacorp/employee/\_search

```
{
  "query": {
    "filtered": {
      "filter": {
        "range": {
          "age": { "gt": 30 }
        }
      },
      "query": {
        "match": {
          "last_name": "smith"
        }
      }
    }
  }
}
```

```
{
  ...
  "hits": {
    "total": 1,
    "max_score": 0.30685282,
    "hits": [
      {
        ...
        "_source": {
          "first_name": "Jane",
          "last_name": "Smith",
          "age": 32,
          "about": "I like to collect rock albums",
          "interests": [ "music" ]
        }
      }
    ]
  }
}
```

# Full-Text Search

GET /megacorp/employee/\_search

```
{
  "query": {
    "match": {
      "about": "rock climbing"
    }
  }
}
```

```
{
  ...
  "hits": {
    "total": 2,
    "max_score": 0.16273327,
    "hits": [
      {
        ...
        "_score": 0.16273327, (1)
        "_source": {
          "first_name": "John",
          "last_name": "Smith",
          "age": 25,
          "about": "I love to go rock climbing",
          "interests": [ "sports", "music" ]
        }
      },
      {
        ...
        "_score": 0.016878016, (1)
        "_source": {
          "first_name": "Jane",
          "last_name": "Smith",
          "age": 32,
          "about": "I like to collect rock albums",
          "interests": [ "music" ]
        }
      }
    ]
  }
}
```

# Phrase Search

```
GET /megacorp/employee/_search
```

```
{
```

```
  "query" : {
```

```
    "match_phrase" : {
```

```
      "about" : "rock climbing"
```

```
    }
```

```
  }
```

```
}
```

# Highlighting

GET /megacorp/employee/\_search

```
{
  "query" : {
    "match_phrase" : {
      "about" : "rock climbing"
    }
  },
  "highlight" : {
    "fields" : {
      "about" : {}
    }
  }
}
```

```
{
  ...
  "hits": {
    "total": 1,
    "max_score": 0.23013961,
    "hits": [
      {
        ...
        "_score": 0.23013961,
        "_source": {
          "first_name": "John",
          "last_name": "Smith",
          "age": 25,
          "about": "I love to go rock climbing",
          "interests": [ "sports", "music" ]
        },
        "highlight": {
          "about": [
            "I love to go <em>rock</em> <em>climbing</em>" (1)
          ]
        }
      }
    ]
  }
}
```

# Bool Queries

Occurrence types:

- `must`
- `should`
- `must_not`

GET `/my_index/my_type/_search`

```
{
  "query": {
    "bool": {
      "should": [
        { "match": { "title": "brown" } }},
        { "match": { "title": "fox" } }},
        { "match": { "title": "dog" } }},
      ],
      "minimum_should_match": 2
    }
  }
}
```

# Combining Queries

GET /my\_index/my\_type/\_search

```
{
  "query": {
    "bool": {
      "must":      { "match": { "title": "quick" }},
      "must_not": { "match": { "title": "lazy"  }},
      "should": [
        { "match": { "title": "brown" }},
        { "match": { "title": "dog"   }}
      ]
    }
  }
}
```

```
{
  "hits": [
    {
      "_id":      "3",
      "_score":   0.70134366,
      "_source": {
        "title": "The quick brown fox jumps over the quick dog"
      }
    },
    {
      "_id":      "1",
      "_score":   0.3312608,
      "_source": {
        "title": "The quick brown fox"
      }
    }
  ]
}
```



# From, Size

GET /megacorp/employee/\_search

```
{
  "from" : 0,
  "size" : 15,
  "query" : {
    "match" : {
      "about" : "rock climbing"
    }
  }
}
```

# Multi-match Query

Run the same match query on multiple fields:

```
"query" :  
{  
  "multi_match" : {  
    "query" : "test",  
    "fields" : [ "subject^3", "message" ]  
  }  
}
```

The `subject` field is three times as important as the `message` field.

# Validate query

GET /website/blog/\_validate/query?explain

```
1 {  
2   "valid": false,  
3   "_shards": {  
4     "total": 1,  
5     "successful": 1,  
6     "failed": 0  
7   },  
8   "explanations": [  
9     {  
10      "index": "shakespeare",  
11      "valid": false,  
12      "error": "org.elasticsearch.index.query.QueryParseException: [shakespeare] request does not support [script_fields]"  
13    }  
14  ]  
15 }
```

# Bulk API

```
{action_and_meta_data}\n{optional_data_source}\n{action_and_meta_data}\n{optional_data_source}\n...
```

## Actions:

POST /\_bulk

```
{ "create": { "_index": "website", "_type": "blog", "_id": "123" }}\n{ "title":    "Cannot create - it already exists" }\n{ "index":   { "_index": "website", "_type": "blog", "_id": "123" }}\n{ "title":    "But we can update it" }
```

- **create** (Create a document only if the document does not already exist)
- **index** (Create a new document or replace an existing document)
- **update** (Do a partial update on a document)
- **delete** (Delete a document)

# Import some data

<https://www.elastic.co/guide/en/kibana/3.0/import-some-data.html>

<https://www.elastic.co/guide/en/kibana/3.0/snippets/shakespeare.json>

```
curl -XPUT localhost:9200/_bulk --data-binary @shakespeare.json
```

# Plugin Elasticsearch - head

Web front end for browsing and interacting with an ElasticSearch cluster. <http://mobz.github.io/elasticsearch-head/>

The screenshot displays the Elasticsearch Head web interface. The top navigation bar includes links for Overview, Indices, Browser, Structured Query, and Any Request. The main header shows the cluster name 'elasticsearch\_kat' with a 'cluster health: yellow (16 of 32)' status. Below this, the 'Cluster Overview' section lists four indices: .kibana, library, shakespeare, and sports. Each index entry includes its size, document count, and a set of five colored boxes (0-4) representing shard status. The 'library' index is selected, and a modal window displays its document structure. The document is a JSON object with fields for index, translog, docs, merges, and refresh.

Index	Size	Docs	Shard 0	Shard 1	Shard 2	Shard 3	Shard 4
.kibana	14.7Ki (14.7ki)	4 (4)	0				
library	9.72Ki (9.72ki)	3 (3)	0	1	2	3	4
shakespeare	18.1Mi (18.1Mi)	111,396 (111,396)	0	1	2	3	4
sports	20.6Ki (20.6ki)	22 (22)	0	1	2	3	4

```
{
  "index": {
    "primary_size_in_bytes": 8962,
    "size_in_bytes": 9952
  },
  "translog": {
    "operations": 0
  },
  "docs": {
    "num_docs": 3,
    "max_docs": 3,
    "deleted_docs": 0
  },
  "merges": {
    "current": 0,
    "current_docs": 0,
    "current_size_in_bytes": 0,
    "total": 0,
    "total_time_in_millis": 0,
    "total_docs": 0,
    "total_size_in_bytes": 0
  },
  "refresh": {
    "total": 5,
    "total_time_in_millis": 0
  }
}
```

<https://github.com/Komakin0/ES-predavanja>