SIMPLE SEARCH WITH ELASTIC SEARCH

MARK STORY

@MARK_STORY

WAT?

- Java based
- Lucene powered
- JSON driven
- Document orientated database
- All out super search solution
- Easy to setup, and use

INDEXES AND TYPES

INDEXES

- Similar concept to databases.
- Contain multiple types.

TYPES

- Similar concept to tables.
- Defines datatypes and indexing rules.

DOCUMENT BASED REST API

Simple to use and easy to understand.

CREATE A DOCUMENT

```
curl -XPOST localhost:9200/contacts/people -d '{
  "name": "Mark Story",
  "email": "mark@mark-story.com",
  "twitter": "@mark_story",
  "country": "Canada",
  "tags": ["cakephp", "cakefest", "canada"]
}'

# Response:
{"ok":true,
"_index":"contacts",
"_type":"people",
"_id":"9izMCaSiQBqD1AJW8si57g",
"_version":1}
```

READ IT BACK

```
curl -XGET localhost:9200/contacts/people/$id?pretty=true

# Response:
{"_index":"contacts",
"_type":"people",
"_id":"9izMCaSiQBqD1AJW8si57g",
"_version":1,
"exists":true,
"_source" : {
  "name": "Mark Story",
  "email": "mark@mark-story.com",
  "twitter": "@mark_story",
  "country": "Canada",
  "tags": ["cakephp", "cakefest", "canada"]
}}
```

DELETE IT!

```
curl -XDELETE localhost:9200/contacts/people/$id

# Response:
{"ok":true,
"found":true,
"_index":"contacts",
"_type":"people",
"_id":"9izMCaSiQBqD1AJW8si57g",
"_version":2}
```

SIMPLE SEARCH!

More on search to come

```
curl -XGET localhost:9200/contacts/people/_search?q=Mark&pretty=true
```

```
# Response:
{"took":14,
"timed_out":false,
"_shards":{
  "total":5,
  "successful":5,
  "failed":0
},
"hits":{
  "total":1,
  "max_score":0.11744264,
  "hits":[
    {"_index":"contacts",
      "_type":"people",
      "_id":"slJlyMBTSWaqAMfZUU-lDw",
      "_score":0.11744264,
      "_source" : {
        "name": "Mark Story",
        "email": "mark@mark-story.com",
        "twitter": "@mark_story",
        "country": "Canada",
        "tags": ["cakephp", "cakefest", "canada"]
      }
     }
  ]
}}
```

THIS ALL SOUNDS TOO BADASS TO BE TRUE

DOCUMENT 'DATABASE' IS A BIT LIMITED

- Partial updates are doable but painful
- No joins
- No map reduce
- Cannot replace all other datasources

BUT SEARCH IS AMAZZZING

SEARCH BETWEEN TYPES & INDEXES

Search multiple types

```
curl -XGET localhost:9200/contacts/people,companies/_search?q=name:Mark
```

Search multiple indexes in your cluster

```
curl -XGET localhost:9200/_all/people/_search?q=name:Mark
```

FANCY SEARCH OPTIONS

SEARCH WITH TEXT EXPRESSIONS

```
curl -XGET localhost:9200/contacts/people/_search?pretty=true -d '{
  "query": {
    "query_string": {
      "query": "mark OR weldon"
    }
  }
}'
```

HIGHLIGHT SEARCH KEYWORDS

Wrap search terms in highlighting text/markup/html. Great for larger documents, as you can extract fragments.

```
curl -XGET localhost:9200/contacts/people/_search?pretty=true -d '{
    "query": {
        "text": {
            "email": "mark"
        }
    },
    "highlight": {
        "fields": {
            "email": {},
            "name": {}
        }
    }
}'
```

FACETS

Facets provide aggregated data about a query. You can use this data to create drill down search, or histogram data.

- Term counts.
- Custom script values.
- Ranges - like price ranges.
- Geo distance facets - aggregate results by distance.

```
curl -XGET localhost:9200/contacts/people/_search?pretty=true -d '{
    "query": {
        "query_string": {
            "query": "*.com"
        }
    },
    "facets": {
        "tagged": {"terms": {"field": "tags"} }
    }
}'
```

KNOBS & BUTTONS

MAPPINGS

- Allows fine-grained searching later on, and lets you configure custom mappings.
- Control the data types, and indexing used for JSON document types.
- Disable indexing on specific fields.
- Configure custom analyzers. For example, non-english stemming.

AVAILABLE MAPPING TYPES

- string, integer, float, boolean, null
- object - Standard type for nested objects. Allows
- Arrays are automatically handled as the above. properties to be defined.

- multi_field - Allows a field to be handled multiple ways with different aliases.
- nested - Indexes sub objects, and works with nested filter/queries.
- ip - For ipv4 data.
- geo_point - For lat/lon values. Enables piles of search options.
- attachment - Store a blob. Can index many text based documents like PDF.

CREATE A MAPPING

```
curl -XPUT localhost:9200/contacts/people/_mapping -d '{
  "people": {
    "properties": {
      "name": {"type": "string"},
      "email": {"type": "string"},
      "twitter": {"type": "string"},
      "country": {"type": "string"},
      "tags": {"type": "string"}
    }
  }
}'
```

DEFINE THE ANALYZER USED

- When defining a field you can use `analyzer` `index_analyzer`, and `search_analyzer` to customize the way data is stored, and or searched.
- You can also disable analyzing for specific fields.

DISABLE INDEXING

```
{
  "name": {
    "type": "string",
    "index": "not_analyzed",
  },
  "none": {
    "type": "integer",
    "index": "no"
  }
}
```

SHARDS & REPLICAS

SHARDS

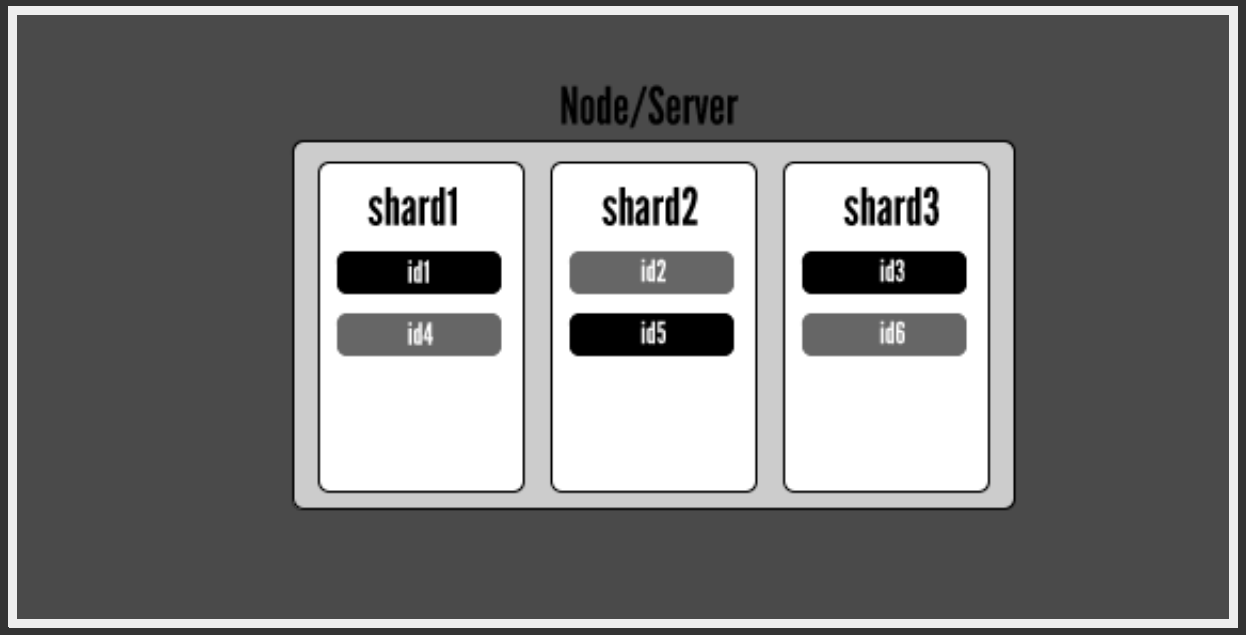
Define how many nodes you want to split your data across.

If a node goes down, you still have some of your data.

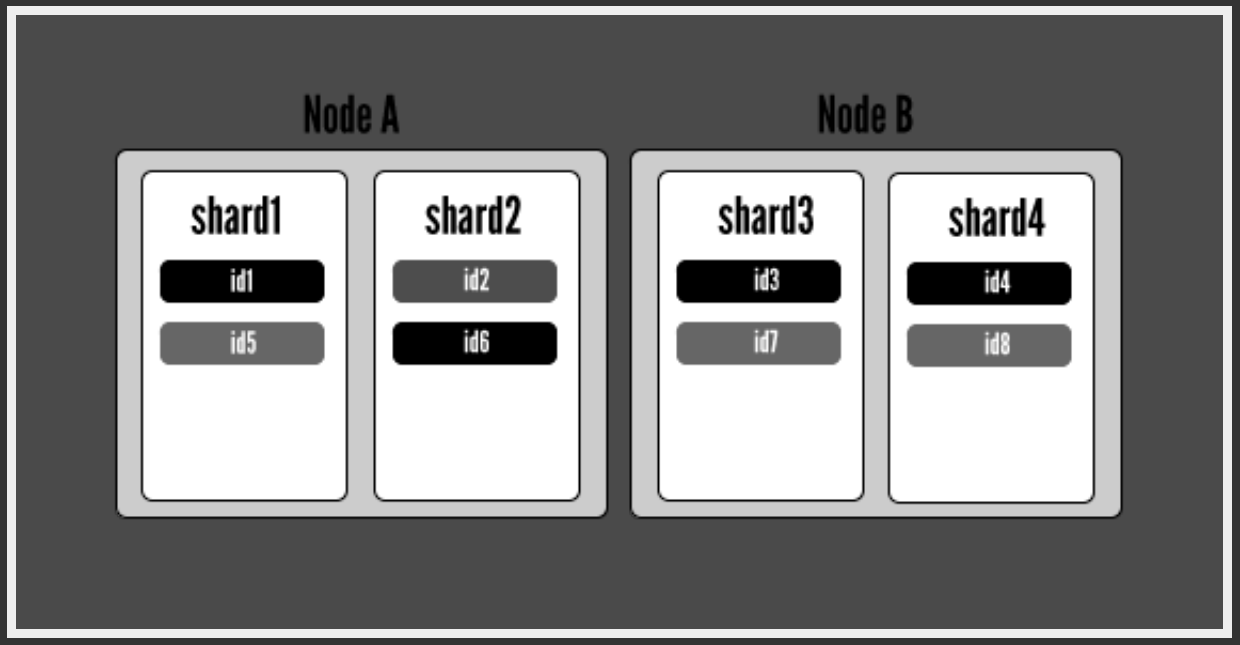
You can use routing to control how data is sharded.

More shards improves indexing performance, as work is distributed.

SIMPLE SHARDING



SHARD OVER MULTIPLE NODES



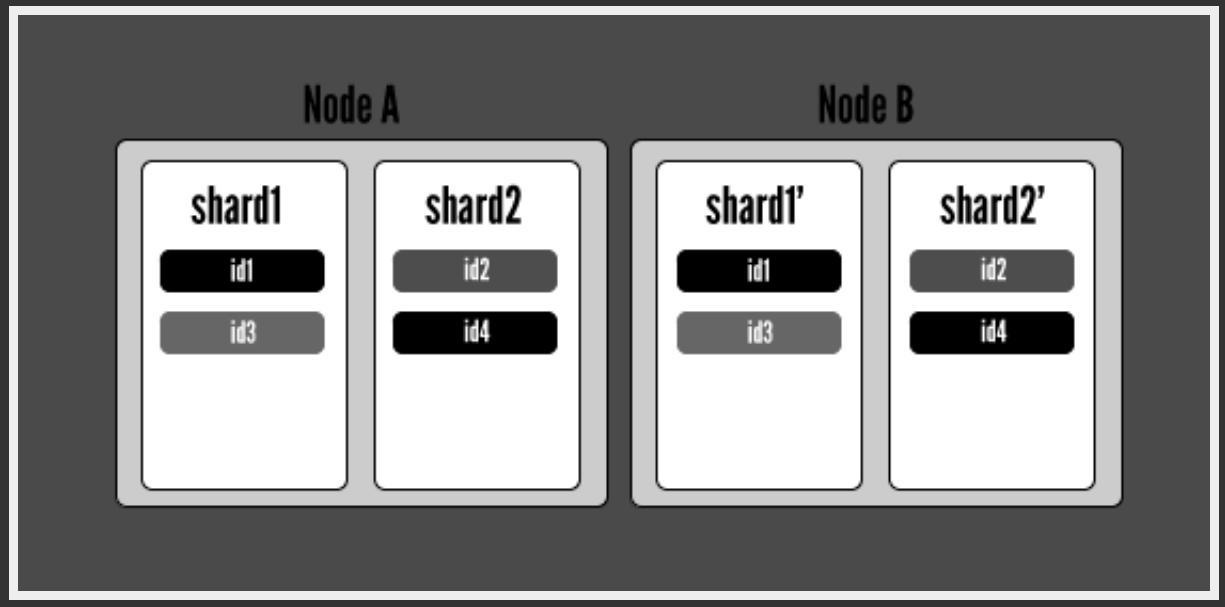
REPLICAS

Define how many copies of your data you want.

If several nodes go down, you might still have all your data.

More replicas improves search performance and cluster availability.

REPLICAS


Node A
Node B
shard1
id1
id3
shard2
id2
id4
shard1'
id1
id3
shard2'
id2
id4

MULTI-TENANCY

Multi-tenancy is a reasonably common requirement, and there are a few ways to do it.

ONE INDEX PER 'TENANT'

- Great for small number of tenants.
- Painful for larger number of tenants. As sharding and replicas can be harder to manage.

```
curl -XGET localhost:9200/mark/contacts/_search?pretty=true -d '{
  "query": {
    "query_string": {
      "query": "weldon OR jose"
    }
  }
}'
```

SPECIAL FILTER CONDITIONS

- More error prone as you have to include a filter condition.
- Easy to shard and setup replicas.
- Easily scales to many tenants. As shards/replicas are shared.
- Make sure tenant id is a non-analyzed value.

```
curl -XGET localhost:9200/accounting/invoices/_search?pretty=true -d '{
  "query": {
    "filtered": {
      "filter": {
        "term": {"accountid": 1}
      },
      "query": {
        "query_string": {
          "query": "purple wifi"
        }
      }
    }
  }
}'
```

OTHER BATTERIES INCLUDED

Routing Define how documents are sharded.

Rivers Pipe data in realtime from sources like RabbitMQ.

Thrift Talk thirft to ElasticSearch.

INTEGRATION WITH CAKEPHP

HTTPSOCKET + JSON_ENCODE()

- Basic, can be hard to use.
- No magic.

ELASTICSEARCH DATASOURCE

(David Kullman)

- Behavior to auto index on aftersave
- Datasource for searching elasticsearch
- Console app to index models

ELASTICSEARCH PLUGIN

(Kevin von Zonneveld)

- Similar features to the previous plugin
- Offers more control on how data is indexed

QUESTIONS?