



Elasticsearch.

Le moteur de recherche
élastique pour tous

David Pilato, IDEO Technologies, Paris



Qui ?



```
$ curl http://localhost:9200/talk/speaker/dpilato
{
  "nom" : "David Pilato",
  "jobs" : [
    { "boite" : "SRA Europe (SSII)", "mission" : "bon à tout faire", "duree" : 3 },
    { "boite" : "SFR", "mission" : "touche à tout", "duree" : 3 },
    { "boite" : "e-Brands / Vivendi", "mission" : "chef de projets", "duree" : 4 },
    { "boite" : "DGDDI (douane)", "mission" : "mouton à 5 pattes", "duree" : 8 },
    { "boite" : "IDEO Technologies", "mission" : "directeur technique", "duree" : 0 } ],
  "passions" : [ "famille", "job", "deejay" ],
  "blog" : "http://dev.david.pilato.fr/",
  "twitter" : [ "@dadoonet", "@elasticsearchfr", "@scrutmydocs" ],
  "email" : "david@pilato.fr"
}
```


ScrutMyDocs.org

ScrutMyDocs

UploadSettings



3 documents found in 78 milliseconds

[NOTICE.txt](#)

by The **Apache** Software Foundation (<http://www.apache.org/>).

 [Apache.pdf](#)

Configurer **Apache** mars 2007 i Ce dossier est public et libre au sens de la licence GNU-FDL : Permission serveur Web **Apache** [10] : ins- tallation et configuration de base, notion de Directive, cre´ation d'un re´pertoire serveur Web **Apache** est un logiciel Open source [6] et multiplate-forme : on peut l'installer aussi bien sur serveur **Apache** doit eˆtre installe´ en tant que ((service)) ; – Si vous faites cohabiter ce serveur avec serveur **Apache** doit alors eˆcouter les requetes HTTP [5] sur un autre port, conventionnellement le port

[LICENSE.txt](#)

Apache License Version 2.0, January 2004 <http://www>
apply the **Apache** License to your work. To apply the **Apache** License to your work, attach the following under the **Apache** License, Version 2.0 (the "License"); you may not use this file except in compliance

Pour la démo

Faites du bruit sur Twitter

avec le hashtag

#elasticsearch

SQL Classique

Cherche moi un document
de **décembre 2011** portant sur la **france**
et contenant **produit** et **david**

En SQL :

```
SELECT
  doc.*, pays.*
FROM
  doc, pays
WHERE
  doc.pays_code = pays.code AND
  doc.date_doc > to_date('2011-12', 'yyyy-mm') AND
  doc.date_doc < to_date('2012-01', 'yyyy-mm') AND
  lower(pays.libelle) = 'france' AND
  lower(doc.commentaire) LIKE '%produit%' AND
  lower(doc.commentaire) LIKE '%david%';
```


Les limites de la recherche SQL

- Performances désastreuses sur du like % sur des millions de ligne
- Plombe les performances de l'insertion
- Pas de tolérance aux fotes de frappe
- En général, on se limite aux champs figés ou codifiés
- Recherche « google » impossible !

Au final, on obtient

Power Search:

ID Number	
Web Title	
Url	
Category	Select v
Web Description	
Keywords	
Contact Name	
Contact Email	
Featured Links 	Select v
Cool Links 	Select v
Bold Links	Select v
Icon	☐  ☐  ☐  ☐  ☐  ☐ 
Rating Average 	Select v
Number of Votes	between and
Total Hits	between and
Hits Today	between and
IP Address	
Submission Software Name	

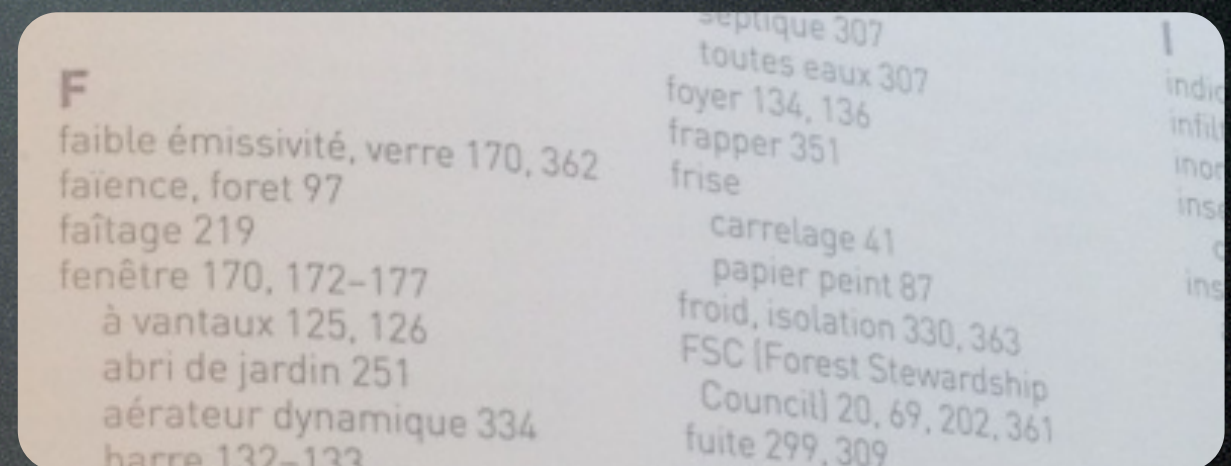
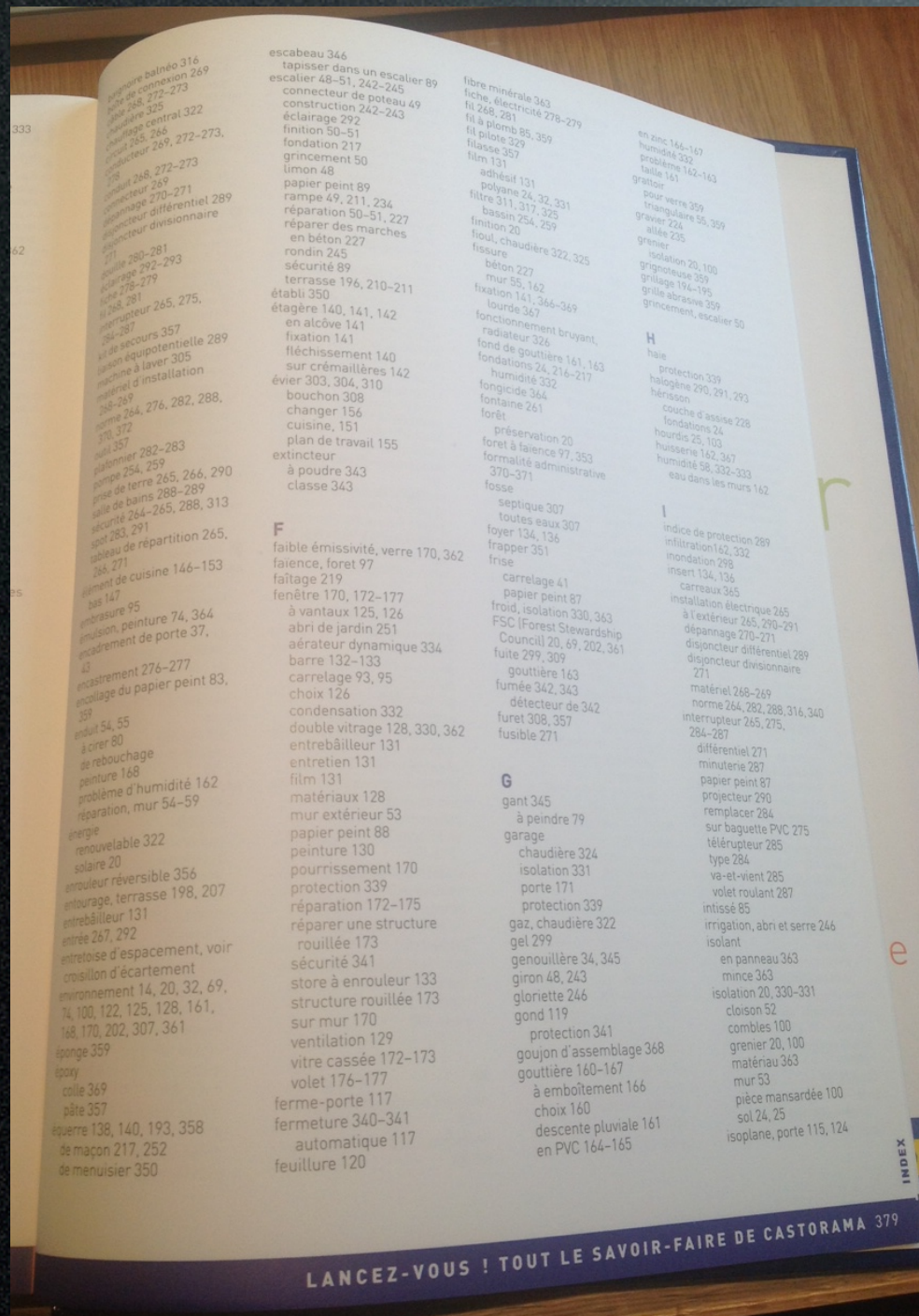
- Autrement dit :
tu as intérêt à
savoir ce que
tu cherches !
- Fouiller est
interdit !

Un moteur de recherche

- Un moteur de recherche est composé de :
 - un moteur d'**indexation de documents**
 - un moteur de **recherche sur les index**
- De fait, un moteur de recherche est énormément plus rapide qu'une base de données pour faire des recherches :

c'est son métier !

L'indexation, c'est quoi en fait ?





Elasticsearch

- Moteur de recherche pour la génération NoSQL
- Basé sur le standard Apache Lucene
- Masque la complexité Java/Lucene à l'aide de services standards HTTP / RESTful / JSON
- Utilisable à partir de n'importe quelle technologie
- Ajoute la couche cloud manquante à Lucene
- C'est un **moteur**, pas une **interface graphique** !

Points clés

- **Simple !** En quelques minutes (Zero Conf), on dispose d'un moteur complet prêt à recevoir nos documents à indexer et à faire des recherches.
- **Efficace !** Il suffit de démarrer des nœuds Elasticsearch pour bénéficier immédiatement de la réplication, de l'équilibrage de charge.
- **Puissant !** Basé sur Lucene, il en parallélise les traitements pour donner des temps de réponse acceptables (en général inférieurs à 100ms)
- **Complet !** Beaucoup de fonctionnalités : analyse et facettes, percolation, rivières, plugins, ...

Ranger ses données

- **Document** : Un objet représentant les données (au sens NoSQL).
Penser "recherche", c'est oublier le SGBDR et penser "Documents"

```
{
  "text": "Bienvenue à la conférence #elasticsearch pour #JUG",
  "created_at": "2012-04-06T20:45:36.000Z",
  "source": "Twitter for iPad",
  "truncated": false,
  "retweet_count": 0,
  "hashtag": [ { "text": "elasticsearch", "start": 27, "end": 40 },
               { "text": "JUG", "start": 47, "end": 55 } ],
  "user": { "id": 51172224, "name": "David Pilato",
            "screen_name": "dadoonet", "location": "France",
            "description": "Soft Architect, Project Manager, Senior Developer.\r\nAt this time, enjoying NoSQL world : CouchDB, ElasticSearch.\r\nDeeJay 4 times a year, just for fun !" }
}
```

- **Type** : Regroupe des documents de même type
- **Index** : Espace logique de stockage des documents dont les types sont fonctionnellement communs

Interagir avec Elasticsearch

- API REST : `http://host:port/[index]/[type]/[_action/id]`
Méthodes HTTP : **GET**, **POST**, **PUT**, **DELETE**
- Documents
 - `curl -XPUT http://localhost:9200/twitter/tweet/1`
 - `curl -XGET http://localhost:9200/twitter/tweet/1`
 - `curl -XDELETE http://localhost:9200/twitter/tweet/1`
- Recherche
 - `curl -XPOST http://localhost:9200/twitter/tweet/_search`
 - `curl -XPOST http://localhost:9200/twitter/_search`
 - `curl -XPOST http://localhost:9200/_search`
- Meta-données
 - `curl -XGET http://localhost:9200/twitter/_status`
 - `curl -XPOST http://localhost:9200/_shutdown`

Indexer

```
$ curl -XPUT localhost:9200/twitter/tweet/1 -d '{
  "text": "Bienvenue à la conférence #elasticsearch pour #JUG",
  "created_at": "2012-04-06T20:45:36.000Z",
  "source": "Twitter for iPad",
  "truncated": false,
  "retweet_count": 0,
  "hashtag": [ { "text": "elasticsearch", "start": 27, "end": 40 },
                { "text": "JUG", "start": 47, "end": 55 } ],
  "user": { "id": 51172224, "name": "David Pilato",
            "screen_name": "dadoonet", "location": "France",
            "description": "Soft Architect, Project Manager, Senior Developer.\r\nAt this time, enjoying NoSQL world : CouchDB, ElasticSearch.\r\nDeeJay 4 times a year, just for fun !" }
}'
```



```
{
  "ok":true,
  "_index":"twitter",
  "_type":"tweet",
  "_id":"1"
}
```


Chercher

```
$ curl localhost:9200/twitter/tweet/_search?q=elasticsearch
```



```
{
  "took" : 24,
  "timed_out" : false,
  "_shards" : { "total" : 5, "successful" : 5, "failed" : 0 },
  "hits" : {
    "total" : 1,
    "max_score" : 0.227,
    "hits" : [ {
      "_index" : "twitter",
      "_type" : "tweet",
      "_id" : "1",
      "_score" : 0.227, "_source" : {
        "text": "Bienvenue à la conférence #elasticsearch pour #JUG",
        "created_at": "2012-04-06T20:45:36.000Z",
        "source": "Twitter for iPad",
        [...]
      ]
    } ]
  }
}
```

Nb de documents

Coordonnées

Document source

Pertinence

Les résultats

- Elasticsearch renvoie les 10 premiers résultats. Il fournit l'API permettant de naviguer de page en page (from, size)
- Par défaut, le tri est réalisé sur le score de pertinence

Query DSL

- Requêtes précises : plutôt que de chercher « à la google », on peut utiliser des critères précis :

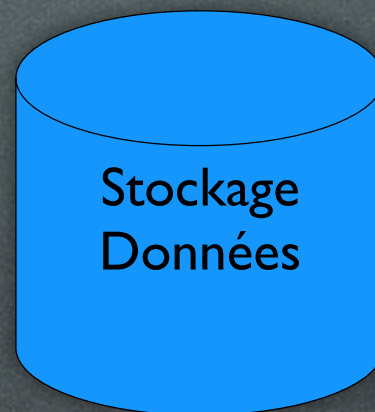
```
$ curl -XPOST localhost:9200/twitter/tweet/_search -d '{
  "bool" : {
    "must" : {
      "term" : { "user" : "kimchy" }
    },
    "must_not" : {
      "range" : {
        "age" : { "from" : 10, "to" : 20 }
      }
    },
    "should" : [
      {
        "term" : { "tag" : "wow" }
      }, {
        "match" : { "tag" : "elasticsearch is cool" }
      }
    ]
  }
}'
```



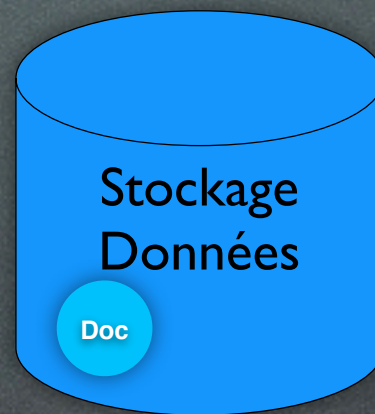

Injecter les données

Et au milieu coule une rivière

La collecte



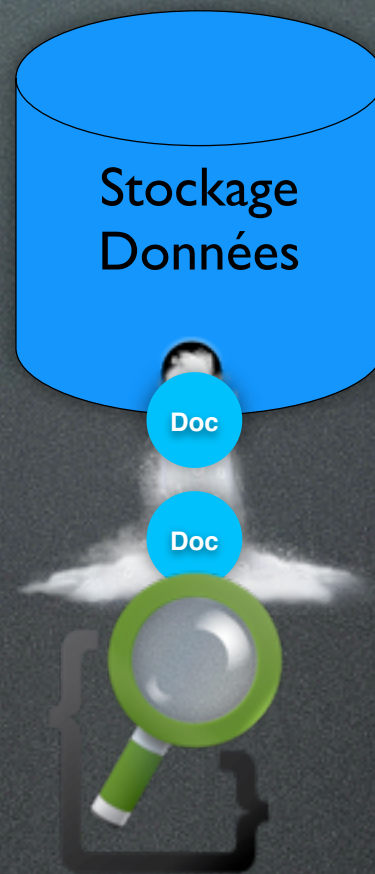
La collecte



La collecte



La collecte



La collecte



La collecte



Quelques Rivers...

- CouchDB River
- CouchBase River
- MongoDB River
- JDBC River
- Wikipedia River
- Twitter River
- RabbitMQ River
- ActiveMQ River
- RSS River
- LDAP River
- FS River
- Dropbox River
- Dick Rivers



Analyser

La puissance des facettes !

Faites parler vos données en les regardant sous différentes facettes !
(Et en temps quasi réel, s'il vous plait !)

Des tweets

ID	Username	Date	Hashtag
1	dadoonet	2012-04-18	1
2	ideo	2012-04-18	5
3	elasticsearch	2012-04-18	2
4	dadoonet	2012-04-18	2
5	ideo	2012-04-18	6
6	elasticsearch	2012-04-19	3
7	dadoonet	2012-04-19	3
8	ideo	2012-04-19	7
9	elasticsearch	2012-04-20	4

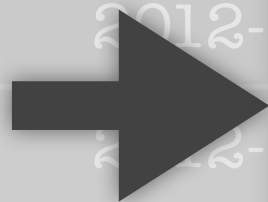
Term Facet

	Username	Date	Hashtag
0	dadoonet	2012-04-18	1
1	ideo	2012-04-18	5
2	elasticsearch	2012-04-18	3
3	dadoonet	2012-04-19	3
4	ideo	2012-04-19	7
5	elasticsearch	2012-04-20	4
6	dadoonet	2012-04-19	3
7	ideo	2012-04-19	7
8	elasticsearch	2012-04-20	4

Username	Count
dadoonet	3
ideo	3
elasticsearch	3

Term Facet

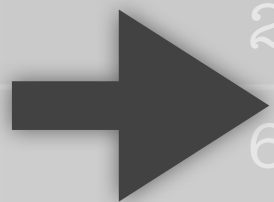
Username	Date	Hashtag
dadoonet	2012-04-20	
ideo	2012-04-20	
elasticsearch	2012-04-20	
dadoonet	2012-04-20	
ideo	2012-04-20	
elasticsearch	2012-04-20	
dadoonet	2012-04-20	
ideo	2012-04-20	
elasticsearch	2012-04-20	4



```
"facets" : {  
  "users" : { "terms" : { "field" : "username" } }  
}  
  
"facets" : {  
  "users" : {  
    "_type" : "terms",  
    "missing" : 0,  
    "total": 9,  
    "other": 0,  
    "terms" : [  
      { "term" : "dadoonet", "count" : 3 },  
      { "term" : "ideo", "count" : 3 },  
      { "term" : "elasticsearch", "count" : 3 }  
    ]  
  }  
}
```

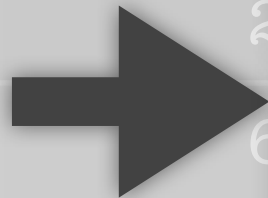

Date Histogram Facet

Date	Hashtag	Par mois	
2012-04-18	1	Date	Count
2012-04-18	5		
2012-04-18	2	2012-04	9
2012-04-18	2	Par jour	
2012-04-18	6		
2012-04-19	3	Date	Count
2012-04-19	3	2012-04-18	5
2012-04-19	7	2012-04-19	3
2012-04-20	4	2012-04-20	1



Date Histogram Facet

Date	Hashtag
2012-04-18	1
2012-04-18	5
2012-04-18	2
2012-04-18	2
2012-04-18	6
2012-04-19	3
2012-04-19	3
2012-04-19	7
2012-04-20	4



```
"facets" : {  
  "perday" : {  
    "date_histogram" : {  
      "field" : "date",  
      "interval" : "day"  
    }  
  }  
}
```

```
"facets" : {  
  "perday" : {  
    "_type" : "date_histogram",  
    "entries": [  
      { "time": 1334700000000, "count": 5 },  
      { "time": 1334786400000, "count": 3 },  
      { "time": 1334872800000, "count": 1 }  
    ]  
  }  
}
```


Range Facet

Hashtag
1
5
2
2
6
3
3
7
4



Hashtag	Count	Min	Max	Moy	Total
$x < 3$	3	1	2	1.667	5
$3 \leq x < 5$	3	3	4	3.333	10
$x \geq 5$	3	5	7	6	18

Range Facet

Hashtag
1
5
2
2
6
3
3
7
4



```
"facets" : { "hashtags" : {  
  "range" : { "field" : "hashtag",  
    "ranges" : [  
      { "to" : 3 },  
      { "from" : 3, "to" : 5 },  
      { "from" : 5 }  
    ] } } }
```

```
"facets" : {  
  "hashtags" : {  
    "_type" : "range",  
    "ranges" : [ {  
      "to": 3,  
      "count": 3,  
      "min": 1, "max": 2,  
      "total": 5, "mean": 1.667  
    }, {  
      "from": 3, "to" : 5,  
      "count": 3,  
      "min": 3, "max": 4,  
      "total": 10, "mean": 3.333  
    }, {  
      "from": 5,  
      "count": 3,  
      "min": 5, "max": 7,  
      "total": 18, "mean": 6  
    } ] } }
```


Site marchand

The image shows a sidebar for filtering products on a marketplace website. The sidebar is divided into several sections, each with a dropdown arrow and a title. The main content area shows a grid of product thumbnails.

Annotations:

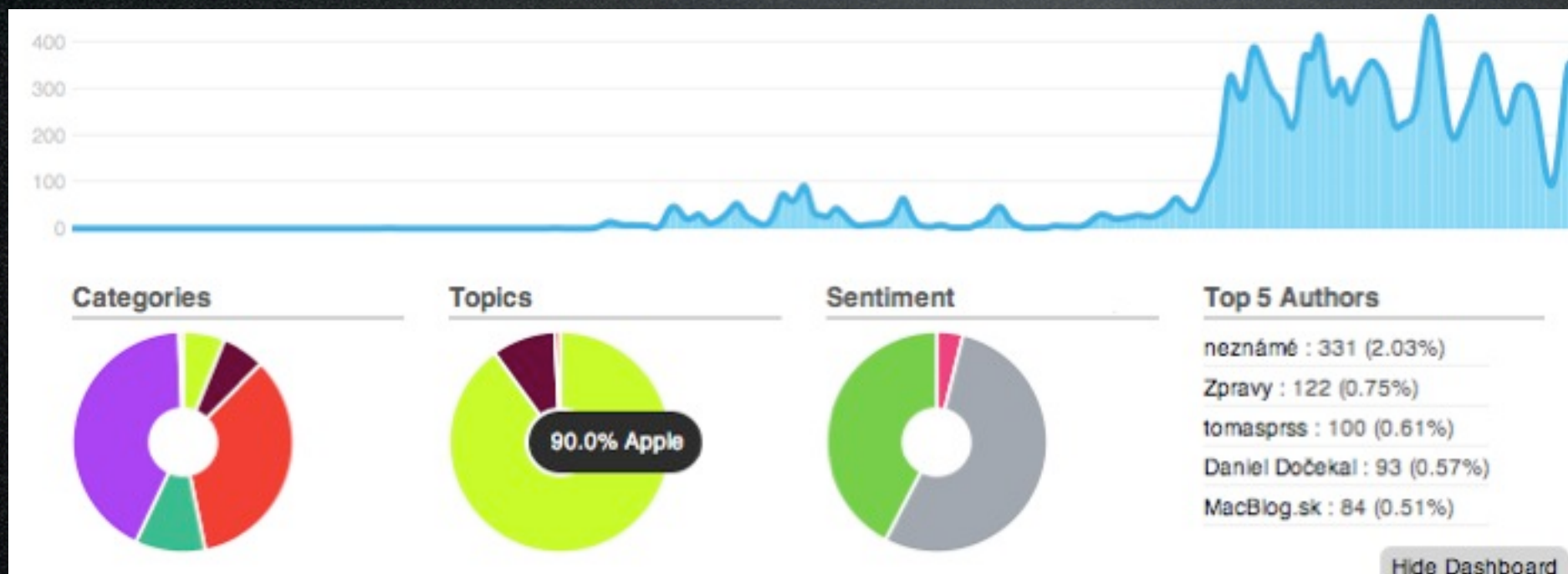
- Ranges:** A red arrow points from the 'Ancienneté' (Age) section to the 'Ranges' label.
- Term:** A red arrow points from the 'Type' section to the 'Term' label.
- Term:** A red arrow points from the 'Orientation' section to the 'Term' label.
- Ranges:** A red arrow points from the 'Tailles' (Sizes) section to the 'Ranges' label.

Filtering Options:

- Ancienneté (Age):**
 - ☒ Tout
 - ☐ ≤ 2 ans (~ 38 600)
 - ☐ ≤ 1 an (~ 21 000)
 - ☐ ≤ 6 mois (~ 10 100)
 - ☐ ≤ 1 mois (~ 1 900)
 - ☐ ≤ 1 semaine (~ 140)
- Collection:**
 - ☐ Collection Standard
 - ☐ Collection Infinite
 - ☐ Collection abonnement
 - ☒ Toutes les collections
- Type:**
 - ☐ Photos (~ 30 000)
 - ☐ Illustrations (~ 18 000)
 - ☐ Vecteurs (~ 16 800)
 - ☐ Vidéos (~ 950)
 - ☒ Tous types
- Orientation:**
 - ☐ Paysage (~ 35 400)
 - ☐ Portrait (~ 13 500)
 - ☐ Carré (~ 16 800)
 - ☒ Tous
- Options:**
- Tailles (Sizes):**
 - ☐ L (~ 64 700)
 - ☐ XL (~ 56 300)
 - ☐ XXL (~ 34 800)
 - ☐ XXXL (~ 10 000)

Analyse temps-réel

- Faire un matchAll sur l'ensemble des données
- Actualiser toutes les x secondes
- Indexer en même temps les nouvelles données



← **Date histogram**

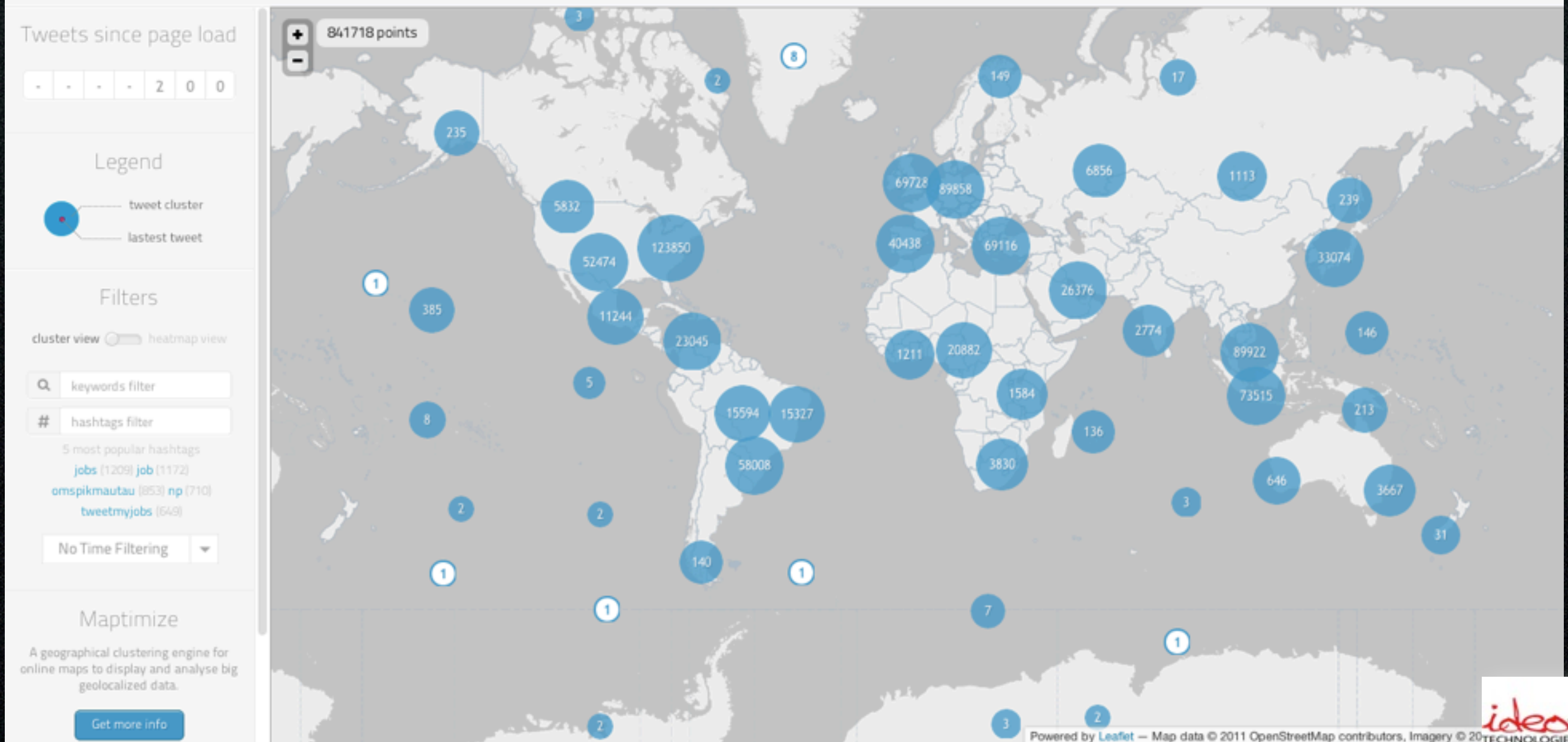
← **Term**

Facettes

Cartographiques

The one million tweet map

powered by **maptimize**



Reprenons notre formulaire

Power Search:

ID Number

Web Title

Url

Category

Web Description

Keywords

Contact Name

Contact Email

Featured Links

Cool Links

Bold Links

Icon ☐  ☐  ☐ 

☐  ☐  ☐ 

Rating Average

Number of Votes between and

Total Hits between and

Hits Today between and

IP Address

Submission Software Name

Recherche Full Text



Reprenons notre formulaire

Power Search:

ID Number

Web Title

Url

Category

Web Description

Keywords

Contact Name

Contact Email

Featured Links

Cool Links

Bold Links

Icon

Rating Average

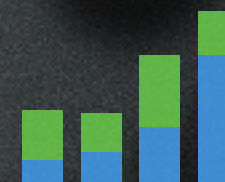
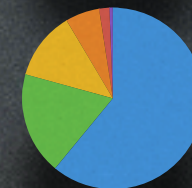
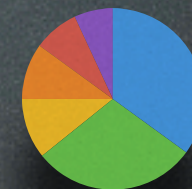
Number of Votes

Total Hits

Hits Today

IP Address

Submission Software Name



[ElasticSearch - Big Data Search and Analytics Made Easy](http://www.elasticsearch.com/)

www.elasticsearch.com/ - Traduire cette page

We're the company behind **elasticsearch**, the open source distributed search engine. We develop, support and provide professional services. Discover and ...

[Améliorer les performances des applications web avec Elasticsearch](http://www.clever-age.com/Veille/Notre_blog_de_veille)

www.clever-age.com/Veille/Notre_blog_de_veille

3 juil. 2012 - Le modèle de données relationnel de certaines applications web est parfois si complexe que les performances se dégradent rapidement en ...

[ElasticSearch - Wikipédia](http://fr.wikipedia.org/wiki/ElasticSearch)

fr.wikipedia.org/wiki/ElasticSearch

ElasticSearch est un moteur de recherche libre (open source) basé sur Apache Lucene. Il est : distribué (architecture cloud); utilise une base de données noSQL ...

[ElasticSearch - Wikipedia, the free encyclopedia](http://en.wikipedia.org/wiki/ElasticSearch)

en.wikipedia.org/wiki/ElasticSearch - Traduire cette page

ElasticSearch is a distributed, RESTful, free/open source search server based on Apache Lucene. It is developed by Shay Banon and is released under the ...

[ElasticSearch : moteur de recherche taillé pour le Cloud](http://soat.developpez.com/articles/elasticsearch/)

soat.developpez.com/articles/elasticsearch/

4 mai 2012 - Cette rétrospective de la présentation de David Pilato lors de Devcoxx France 2012 dont le titre était : « **ElasticSearch** : Moteur de recherche ...

[ElasticSearch - Twitter](https://twitter.com/search?q=%23ElasticSearch)

twitter.com/search?q=%23ElasticSearch - Traduire cette page

La description de ce résultat n'est pas accessible à cause du fichier robots.txt de ce site. En savoir plus

Démonstration

Avez-vous fait du bruit ?

Architecture



```
$ curl -XPUT localhost:9200/_river/twitter/_meta -d '{
  "type" : "twitter",
  "twitter" : {
    "user" : "twitter_user",
    "password" : "twitter_password",
    "filter" : { "tracks" : ["elasticsearch"] }
  }
}
```


Démonstration

<http://onemilliontweetmap.com/>

Architecture

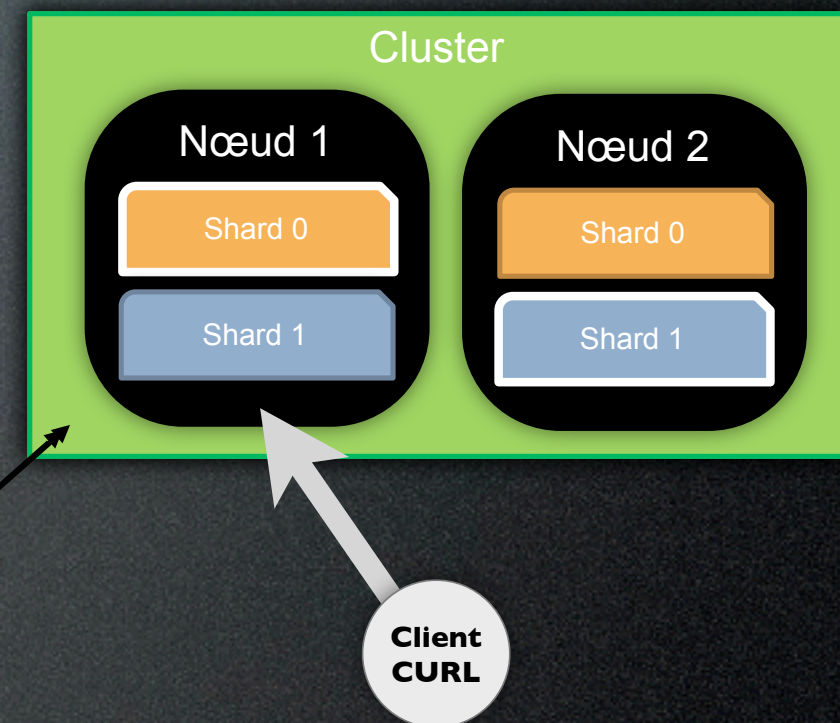
Un peu plus de technique : partitions / répliquions / scalabilité

Lexique

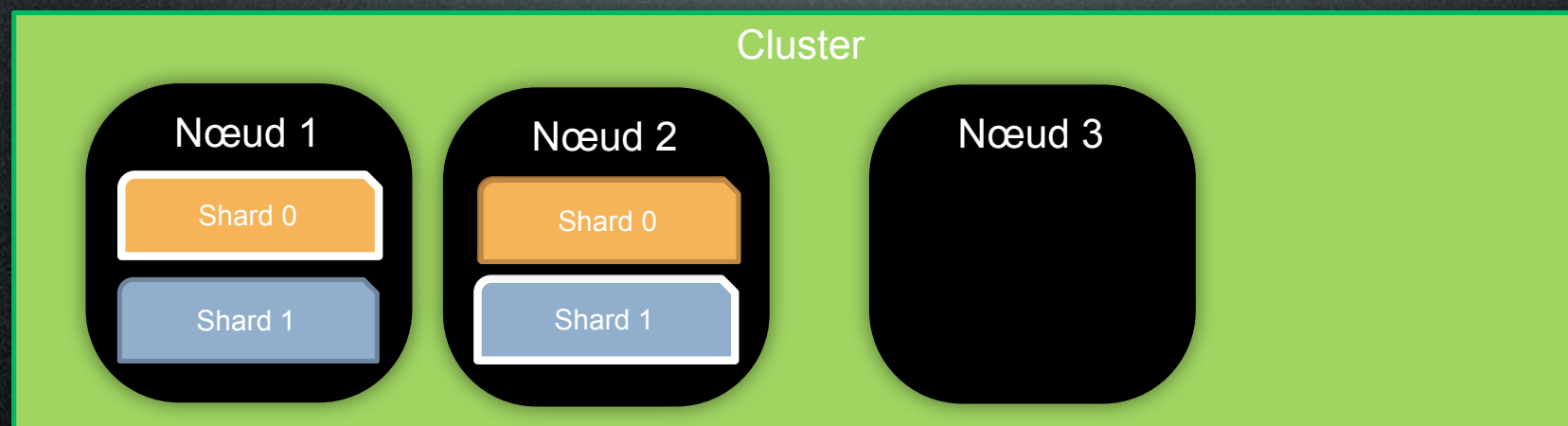
- **Nœud (node)** : Une instance d'Elasticsearch (~ machine ?)
- **Cluster** : Un ensemble de nœuds
- **Partition (shard)** : permet de découper un index en plusieurs parties pour y distribuer les documents
- **Réplication (replica)** : recopie d'une partition en une ou plusieurs copies dans l'ensemble du cluster
- **Partition primaire (primary shard)** : partition élue "principale" dans l'ensemble du cluster. C'est là que se fait l'indexation par Lucene. Il n'y en a qu'une seule par shard dans l'ensemble du cluster.
- **Partition secondaire (secondary shard)** : partitions secondaires stockant les replicas des partitions primaires.

Créons un index

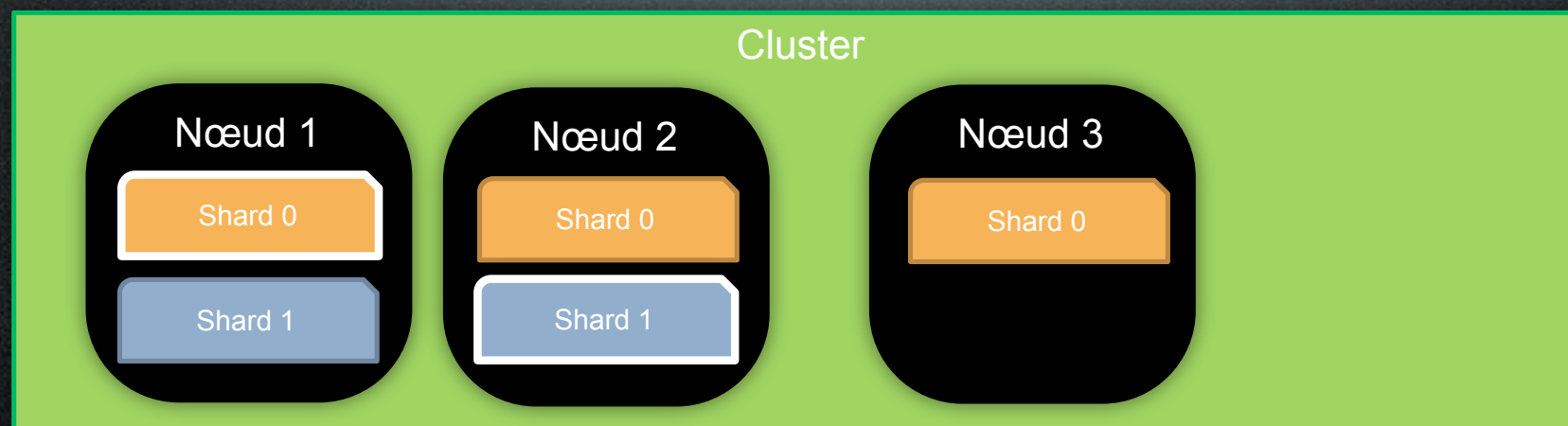
```
$ curl -XPUT localhost:9200/twitter -d '{  
  "index" : {  
    "number_of_shards" : 2,  
    "number_of_replicas" : 1  
  }  
'
```



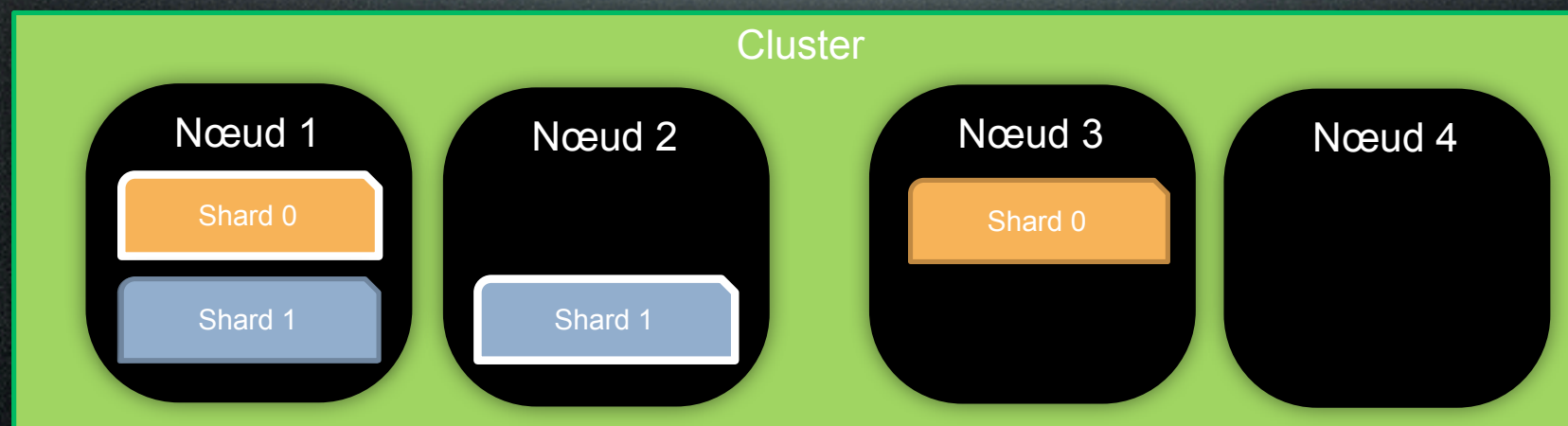
Réallocation dynamique



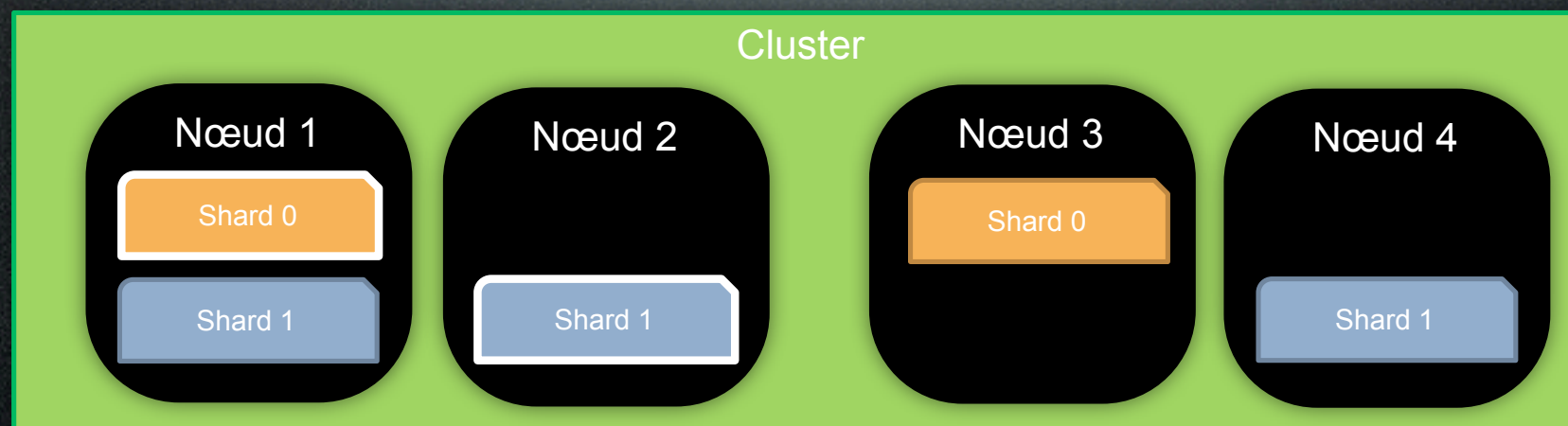
Réallocation dynamique



Réallocation dynamique

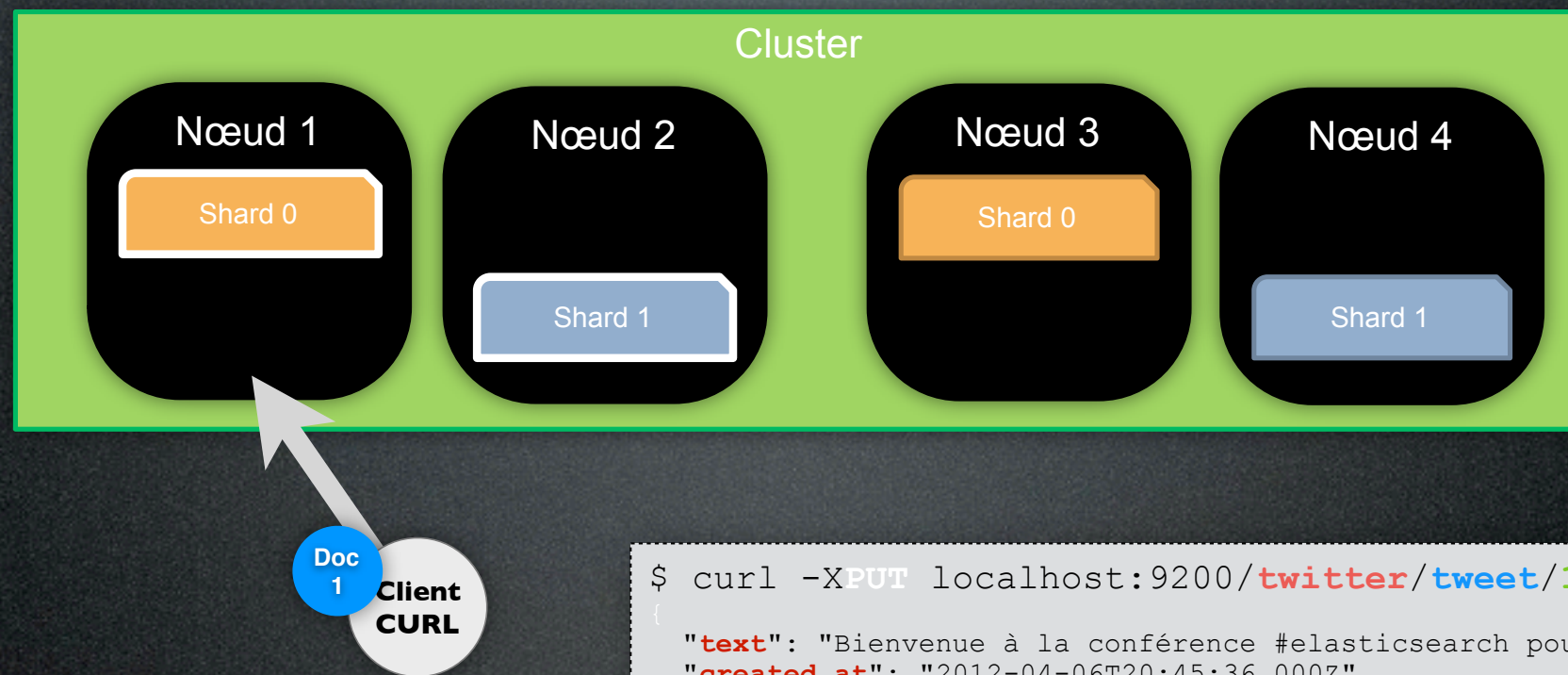


Réallocation dynamique



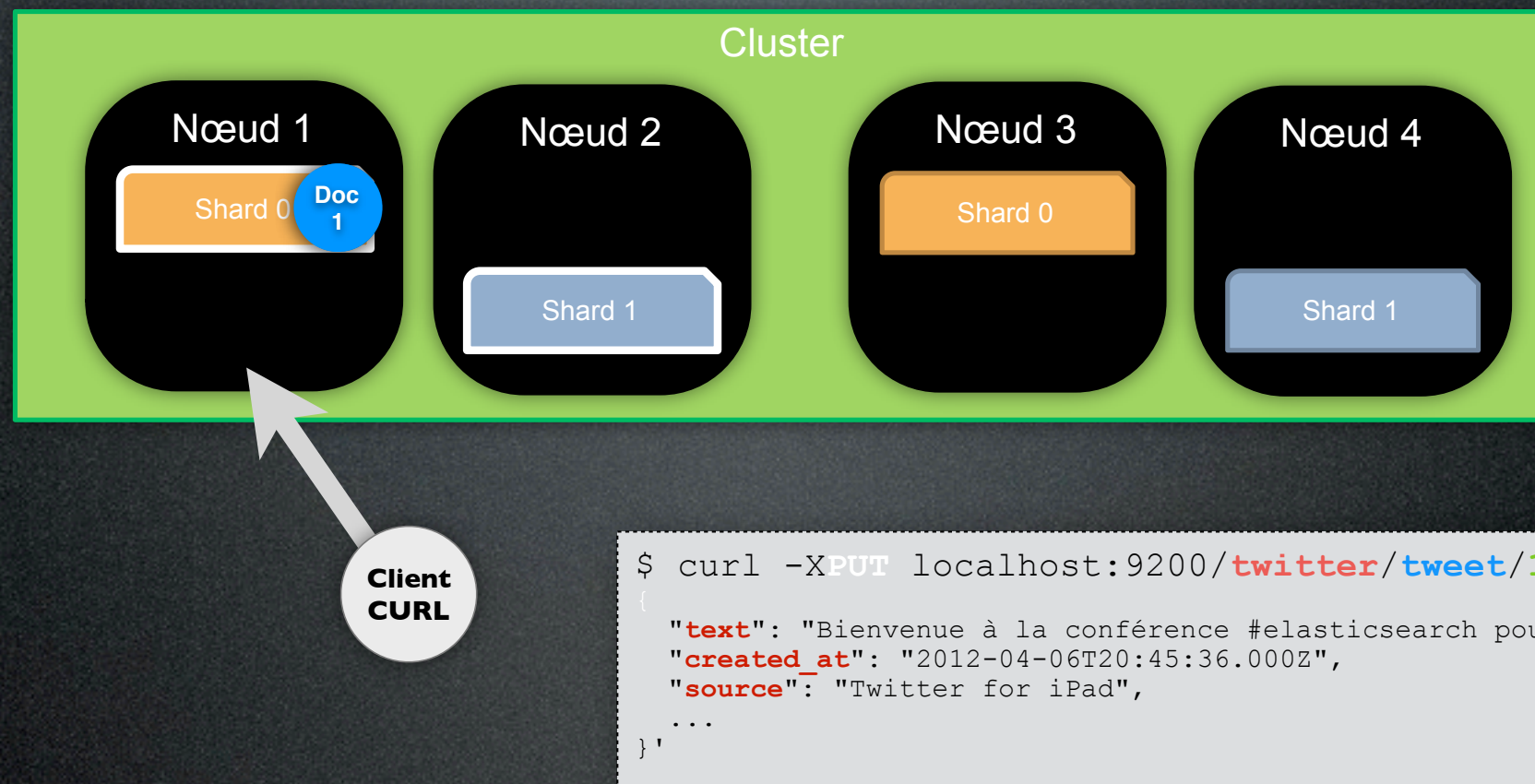
Le **tuning**, c'est trouver le bon équilibre entre le nombre de **nodes**, **shards** et **replicas** !

Indexons un document

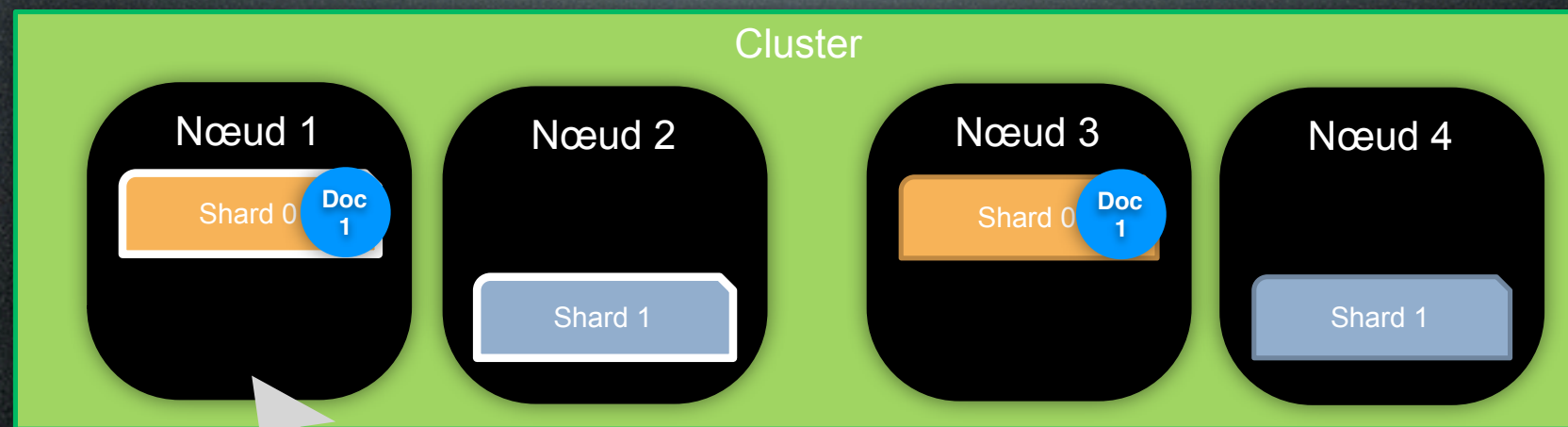


```
$ curl -XPUT localhost:9200/twitter/tweet/1 -d '{
  "text": "Bienvenue à la conférence #elasticsearch pour #JUG",
  "created_at": "2012-04-06T20:45:36.000Z",
  "source": "Twitter for iPad",
  ...
}'
```


Indexons un document



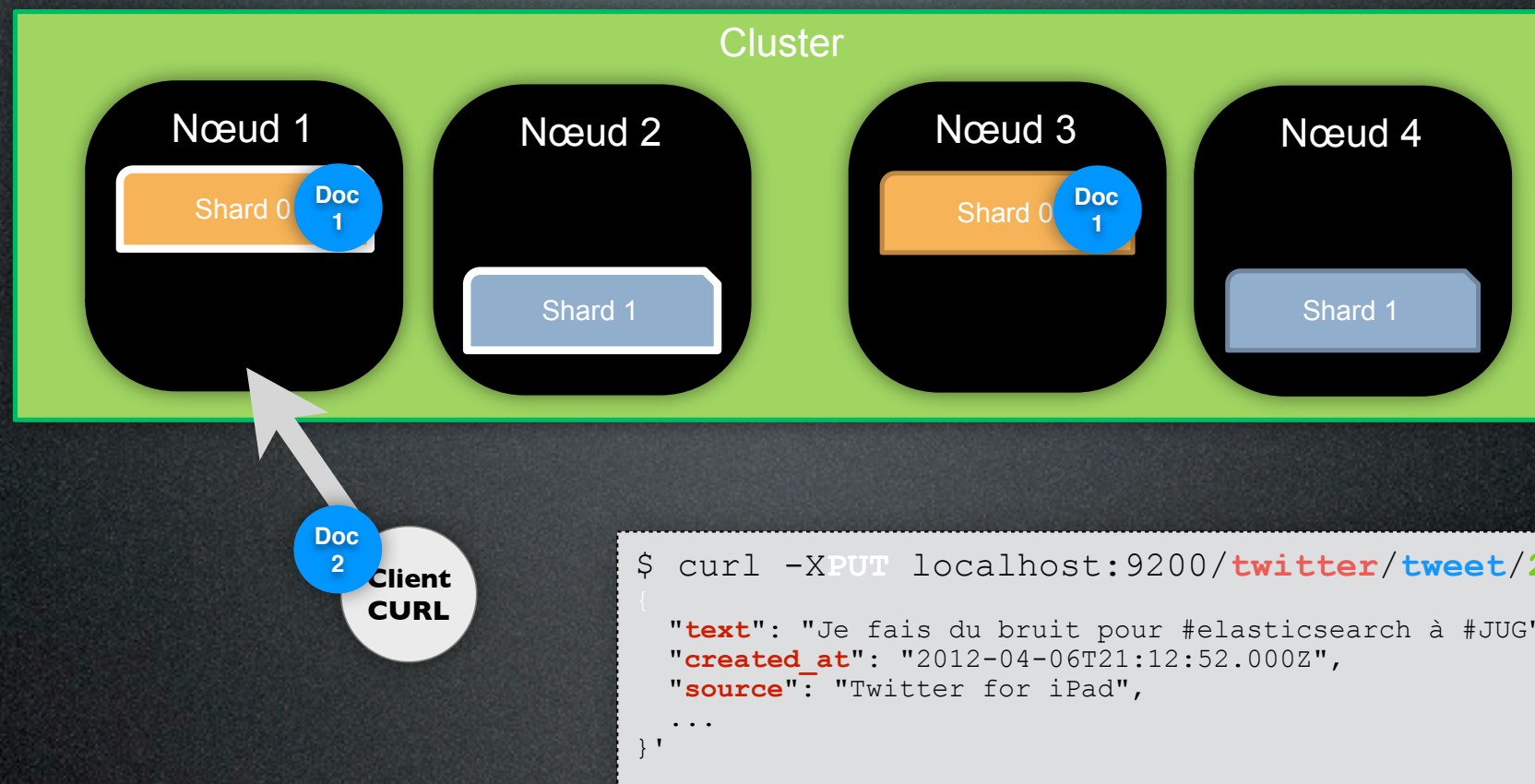
Indexons un document



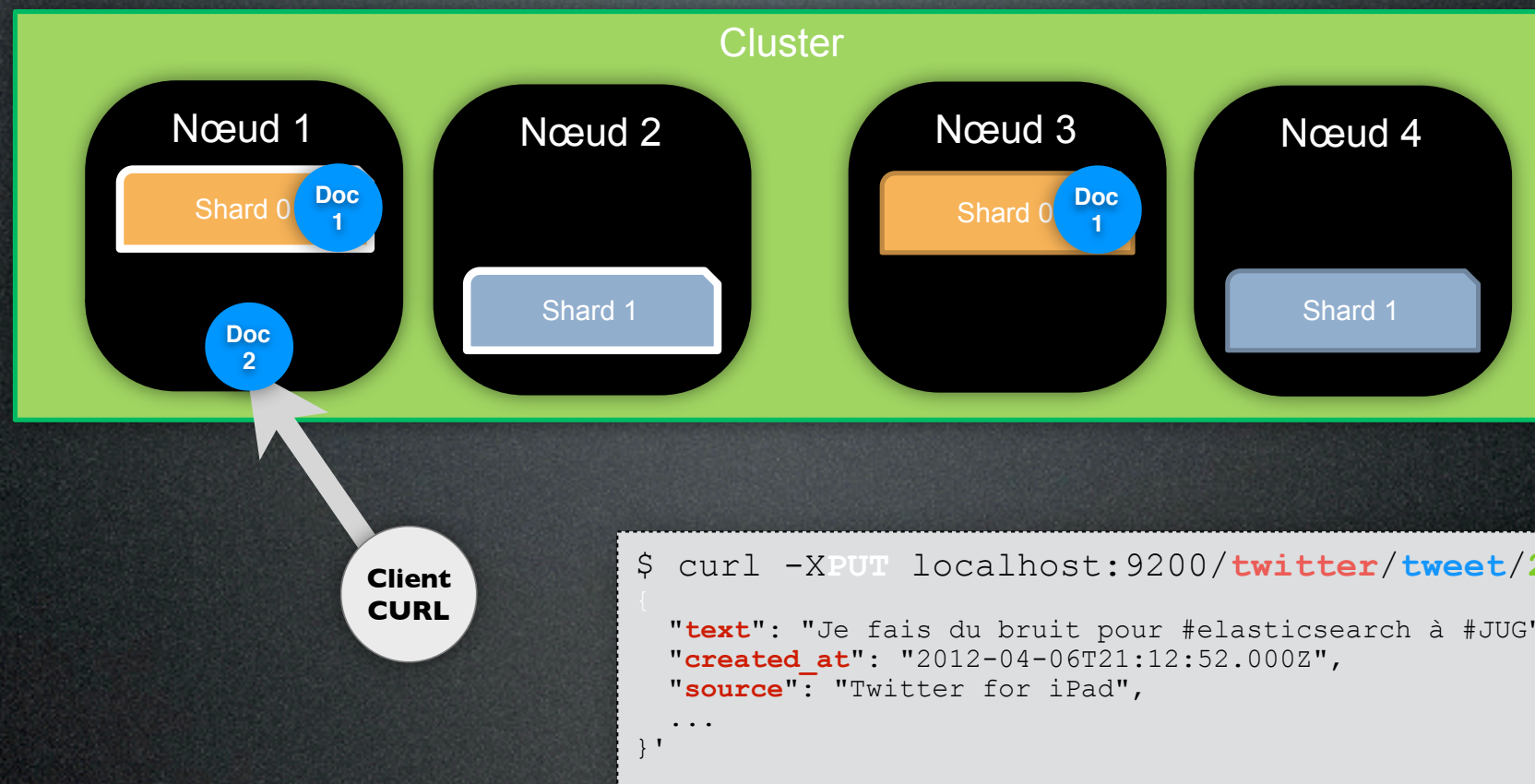
Client
CURL

```
$ curl -XPUT localhost:9200/twitter/tweet/1 -d '{
  "text": "Bienvenue à la conférence #elasticsearch pour #JUG",
  "created_at": "2012-04-06T20:45:36.000Z",
  "source": "Twitter for iPad",
  ...
}'
```

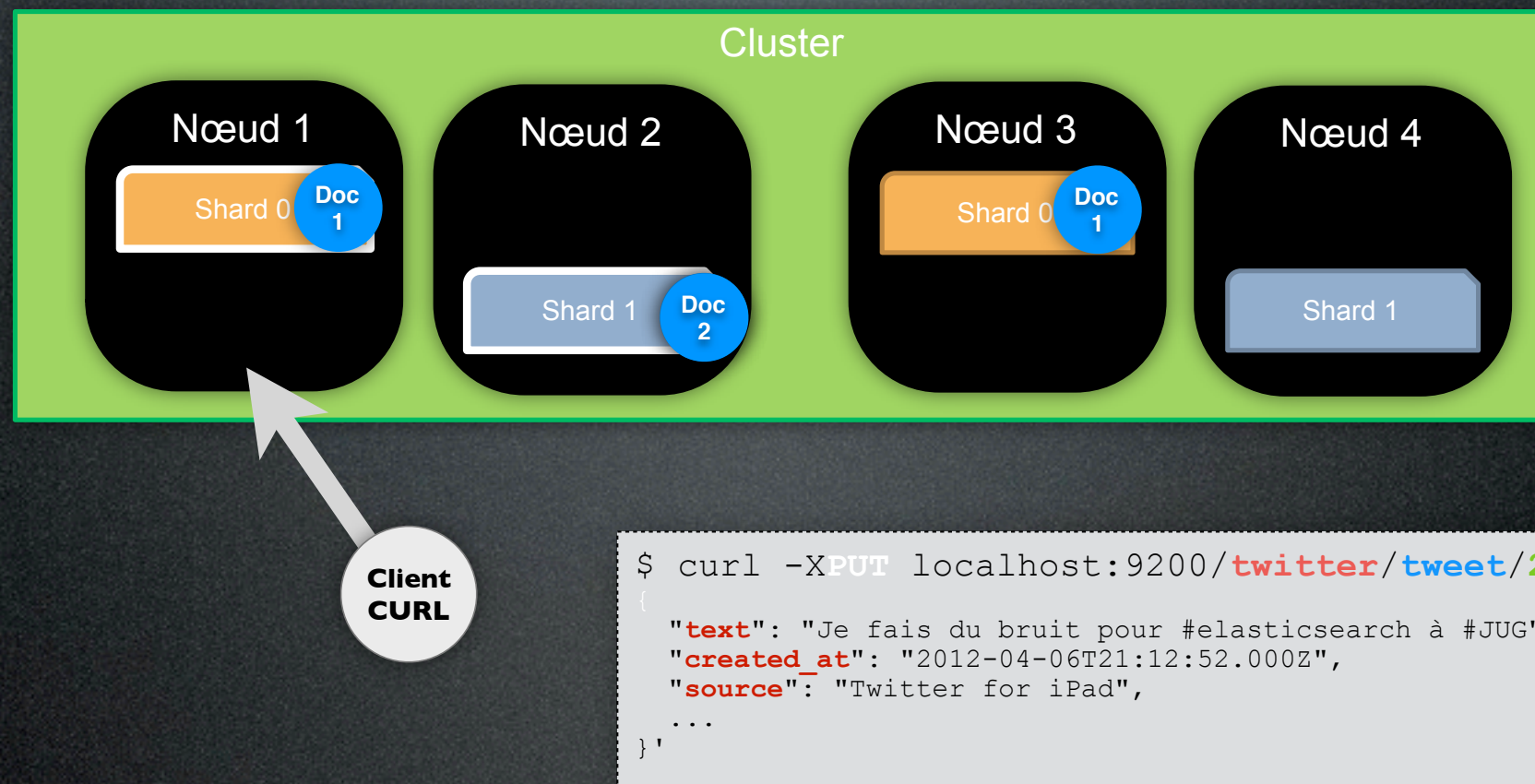

Indexons un 2^{ème} document



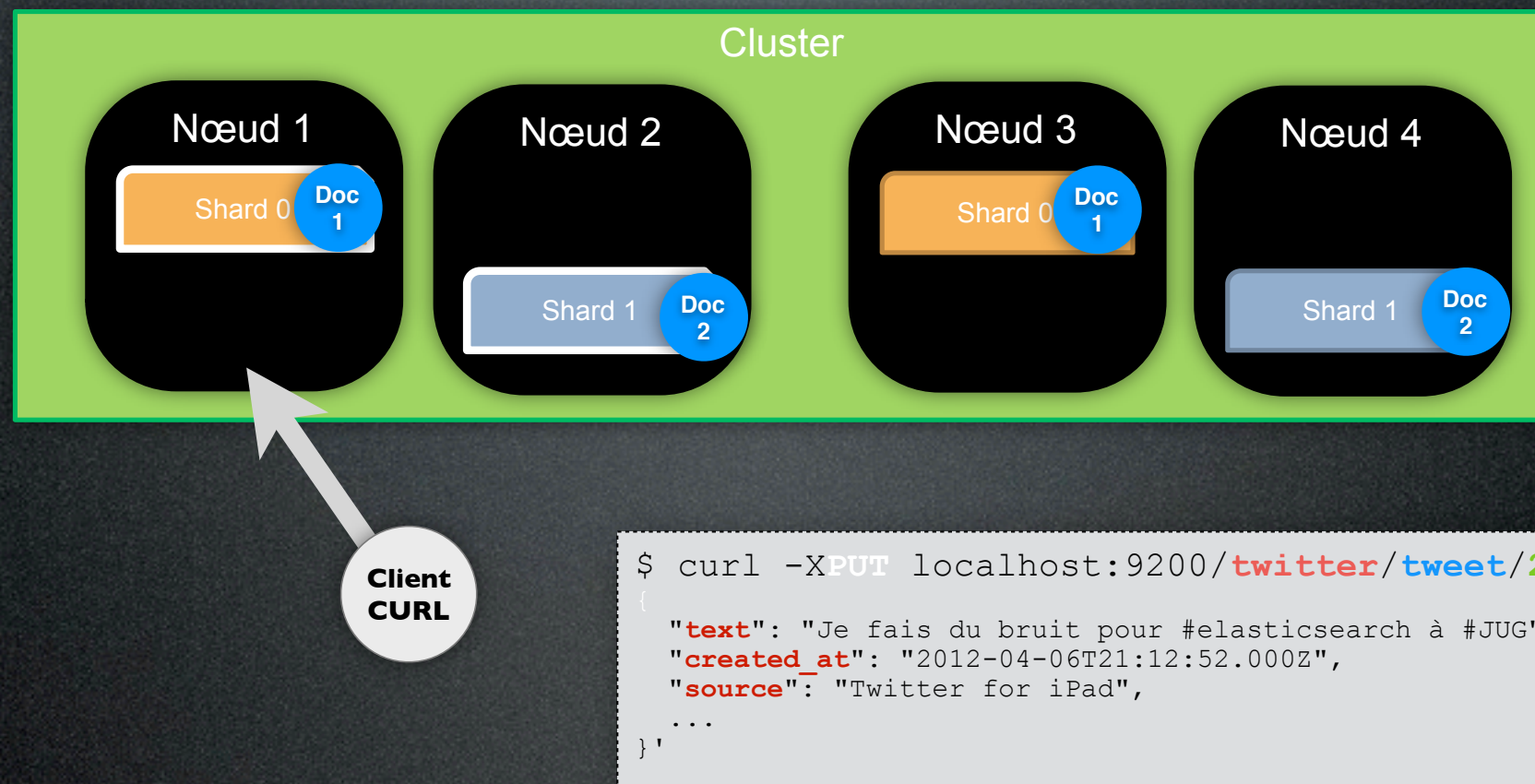
Indexons un 2^{ème} document



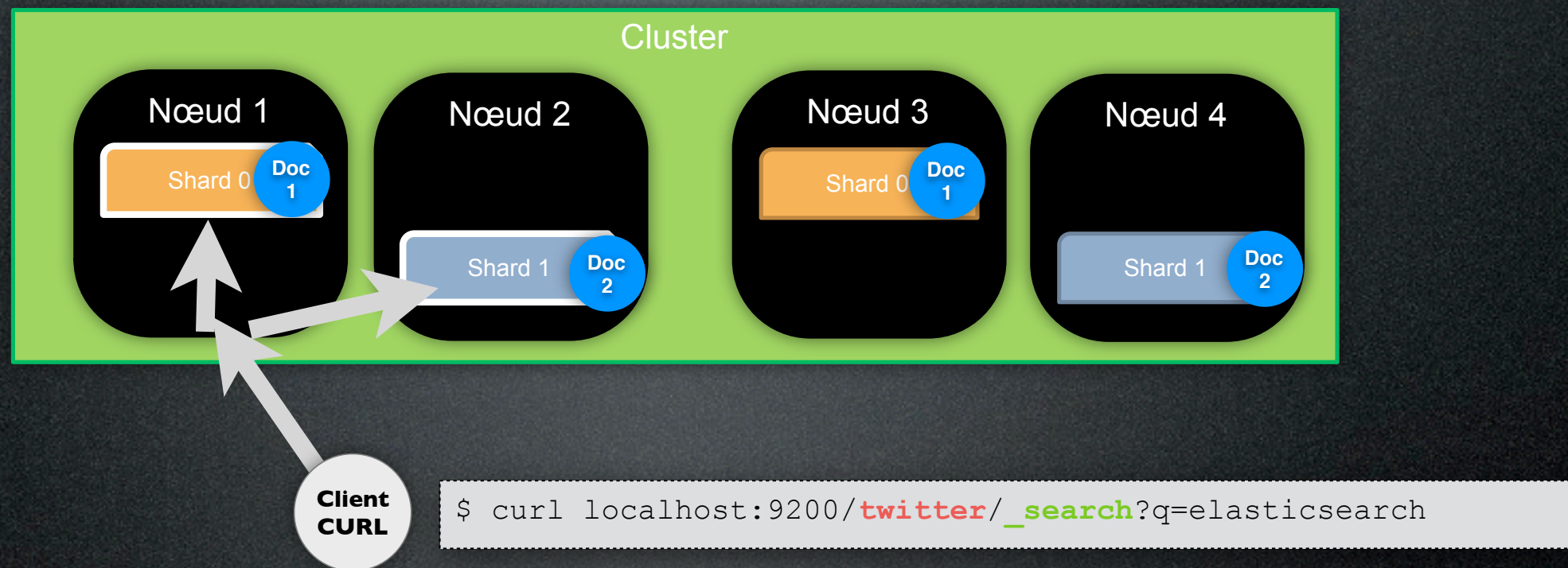
Indexons un 2^{ème} document



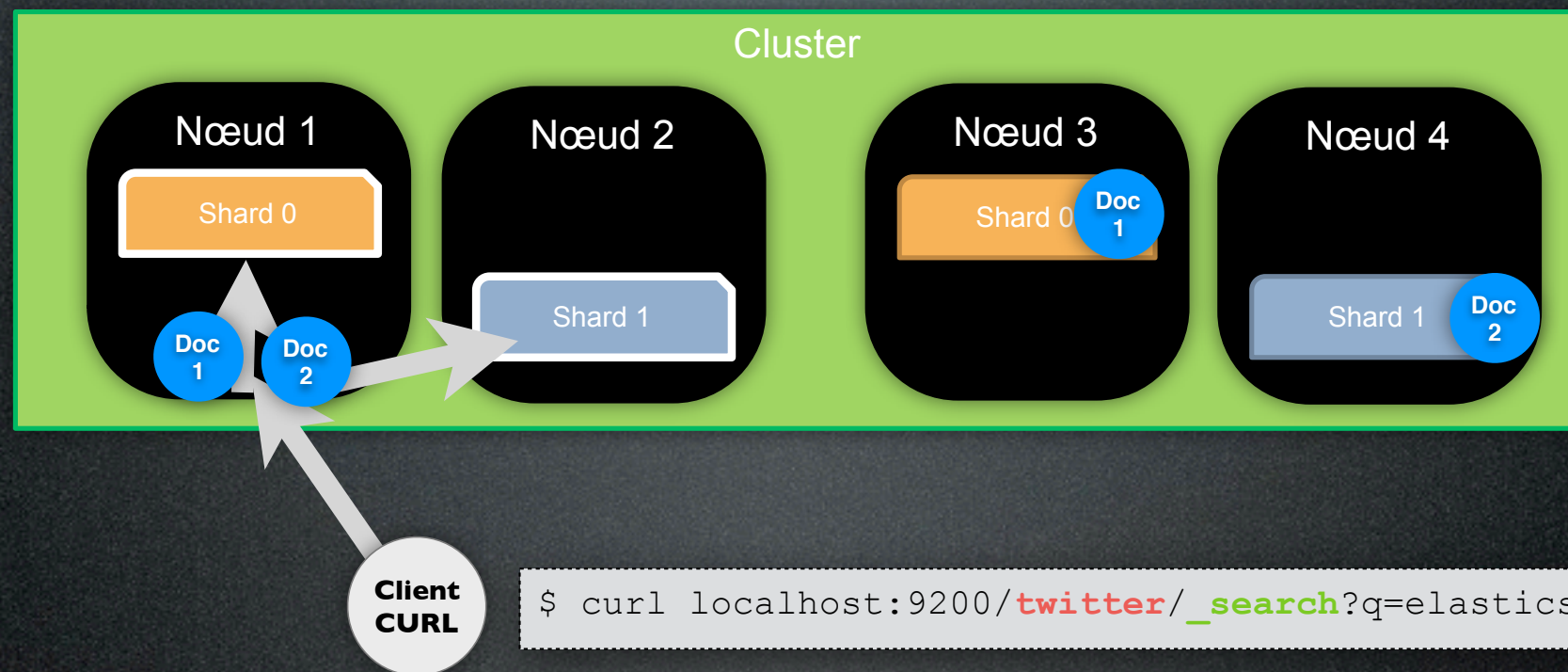
Indexons un 2^{ème} document



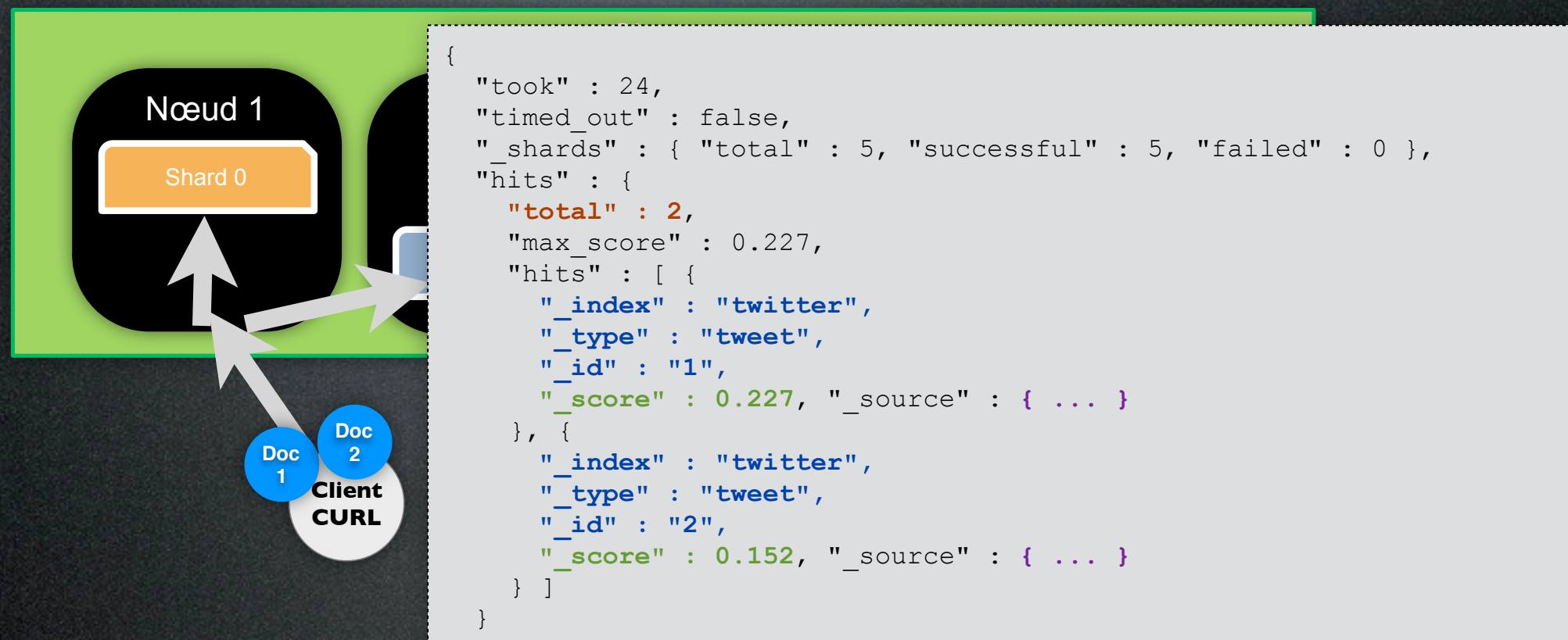
Cherchons !



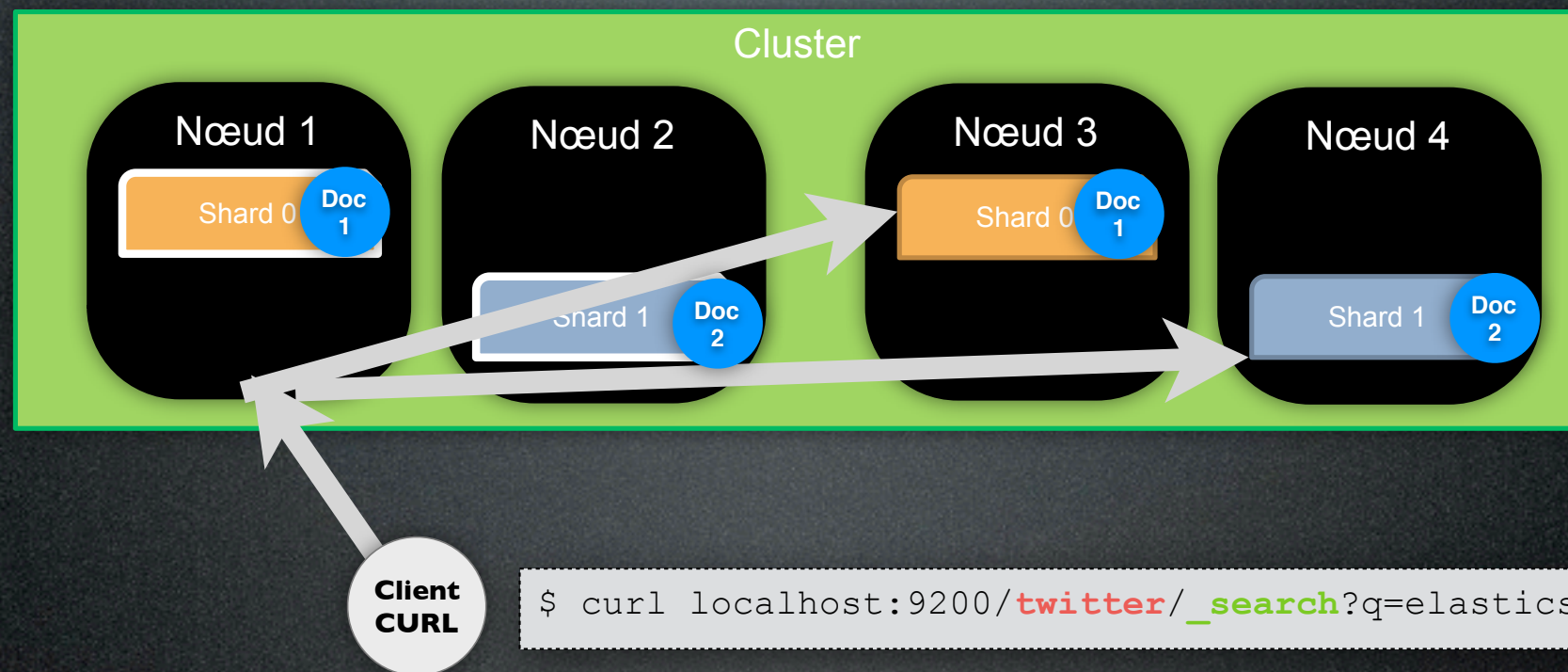
Cherchons !



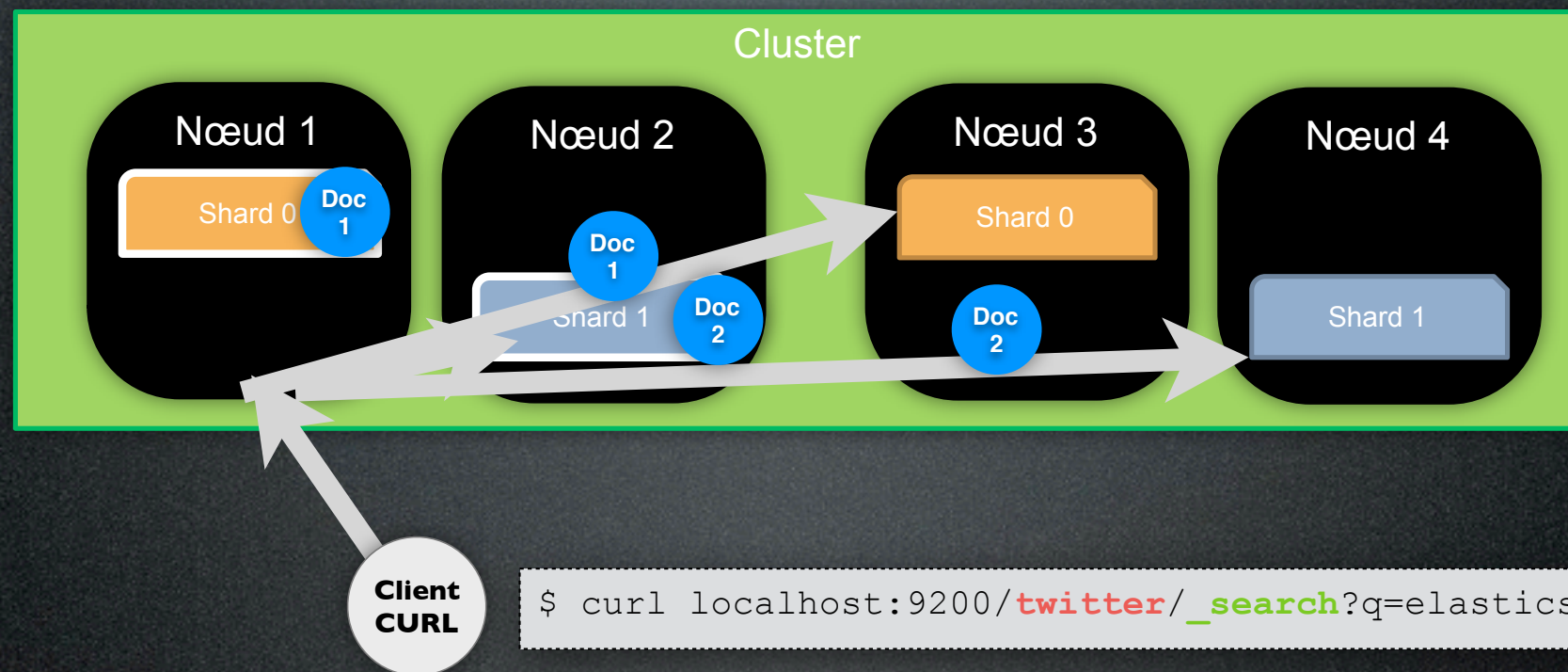
Cherchons !



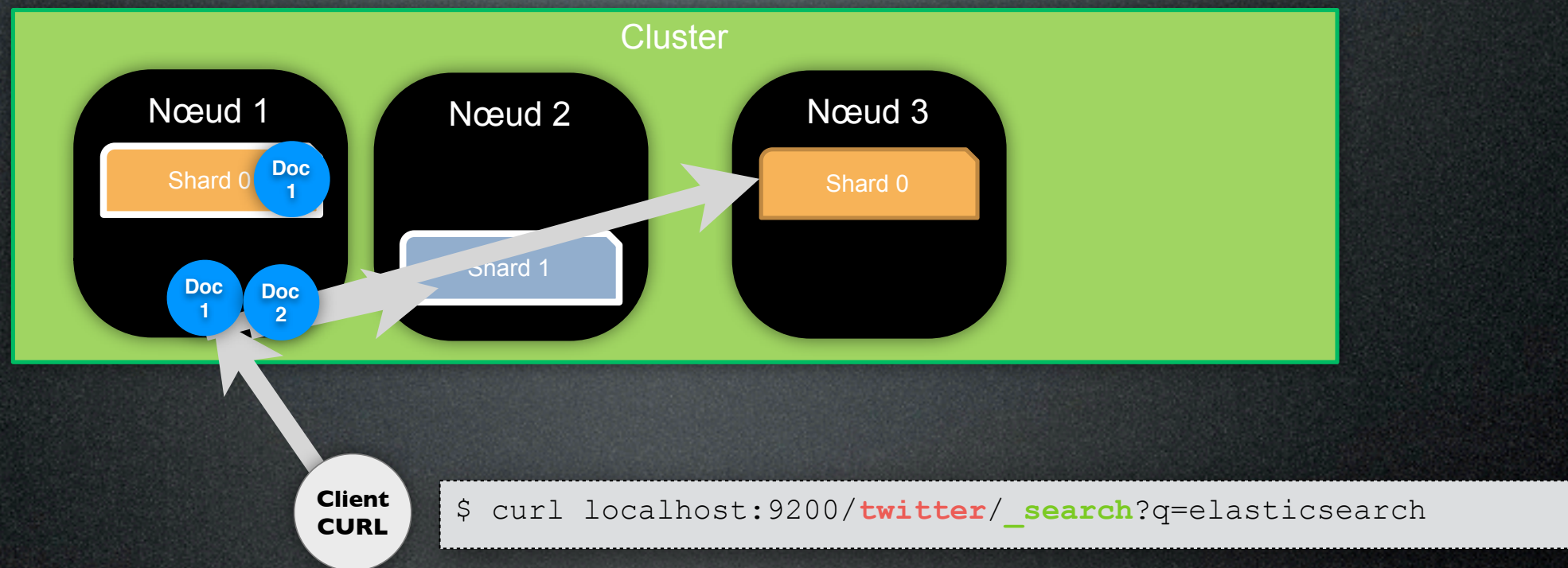
Cherchons encore !



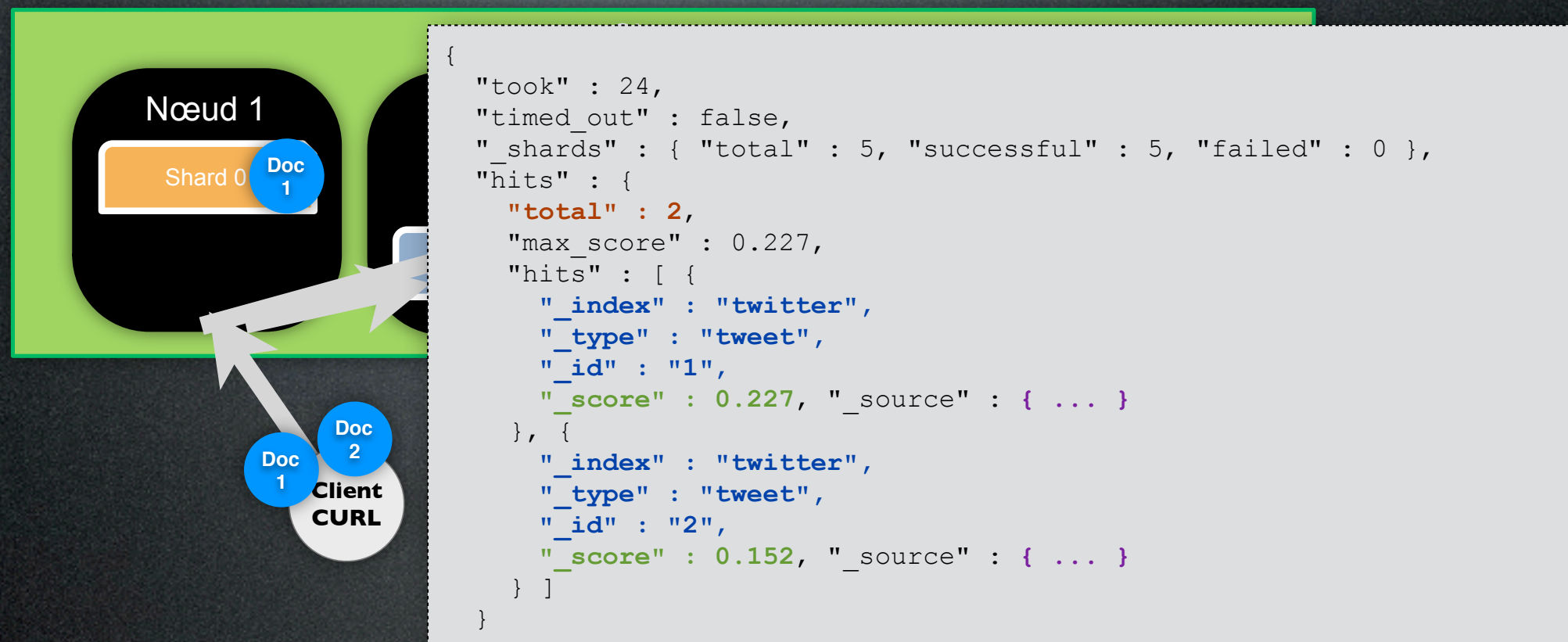
Cherchons encore !



Cherchons encore !



Cherchons encore !



La percolation

Ou la recherche inversée

Usage courant d'un moteur de recherche

- J'indexe un document
- Je cherche de temps en temps si un document m'intéresse
- **Avec de la chance**, il sera bien placé au niveau pertinence dans les résultats. Sinon, il passe **inaperçu** !

La recherche inversée

- Enregistrer ses critères de recherche
- A chaque document indexé, on récupère la liste des recherches qui correspondent
- On a un « **listener** » sur le moteur d'indexation : le **percolator**

Usage du percolator

```
$ curl -XPOST localhost:9200/_percolator/twitter/dadoonet -d '{
  "query" : { "term" : { "user.screen_name" : "dadoonet" } }
}'

$ curl -XPOST localhost:9200/_percolator/twitter/elasticsearch -d '{
  "query" : { "match" : { "hashtag.text" : "elasticsearch" } }
}'

$ curl -XPOST localhost:9200/_percolator/twitter/mycomplexquery -d '{
  "query" : {
    "bool" : {
      "must" : {
        "term" : { "user" : "kimchy" }
      },
      "must_not" : {
        "range" : {
          "age" : { "from" : 10, "to" : 20 }
        }
      },
      "should" : [
        {
          "term" : { "tag" : "wow" }
        }, {
          "match" : { "tag" : "elasticsearch is cool" }
        }
      ]
    }
  }
}'
```


Usage du percolator

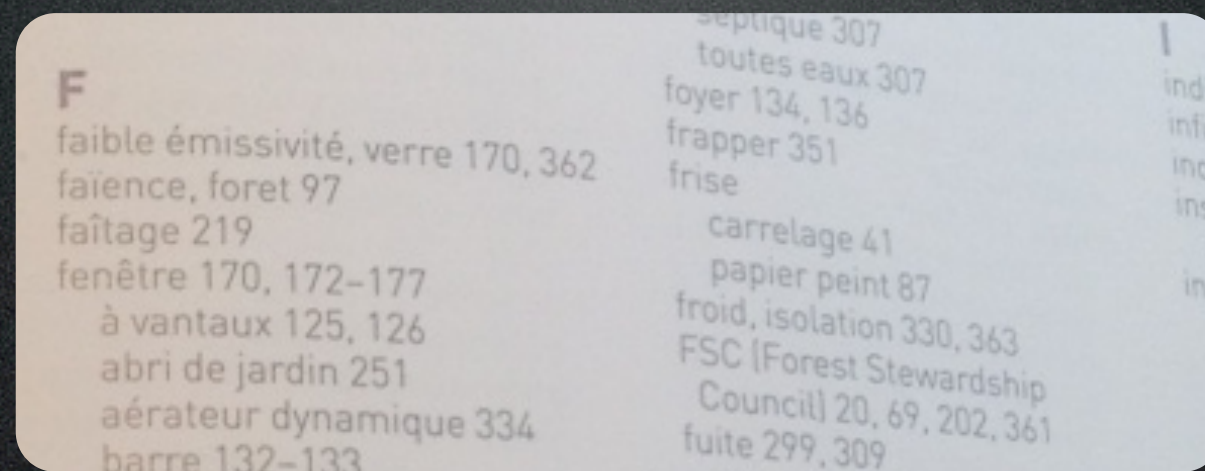
```
$ curl -XPUT localhost:9200/twitter/tweet/1&percolate=* -d '{
  "text": "Bienvenue à la conférence #elasticsearch pour #JUG",
  "created_at": "2012-04-06T20:45:36.000Z",
  "source": "Twitter for iPad",
  "truncated": false,
  "retweet_count": 0,
  "hashtag": [ { "text": "elasticsearch", "start": 27, "end": 40 },
                { "text": "JUG", "start": 47, "end": 55 } ],
  "user": { "id": 51172224, "name": "David Pilato",
            "screen_name": "dadoonet", "location": "France",
            "description": "Soft Architect, Project Manager, Senior
Developer.\r\nAt this time, enjoying NoSQL world : CouchDB, ElasticSearch.\r
\r\nDeeJay 4 times a year, just for fun !" }
}'
```



```
{
  "ok": true,
  "_index": "twitter",
  "_type": "tweet",
  "_id": "1",
  "matches": [
    "dadoonet",
    "elasticsearch"
  ]
}
```


Tout doit être indexé ?

Analyse et mapping



The lazy dog...

The quick brown fox
jumped over the lazy Dog

The quick brown fox
jumped over the lazy dog

Analyseur standard

```
$ curl -XPOST 'localhost:9200/test/_analyze?analyzer=standard&pretty=1' -d  
'The quick brown fox jumped over the lazy Dog'
```



```
{  
  "tokens" : [ {  
    "token" : "quick",  
    "start_offset": 4, "end_offset": 9, "type": "<ALPHANUM>", "position": 2  
  }, {  
    "token" : "brown",  
    "start_offset": 10, "end_offset": 15, "type": "<ALPHANUM>", "position": 3  
  }, {  
    "token" : "fox",  
    "start_offset": 16, "end_offset": 19, "type": "<ALPHANUM>", "position": 4  
  }, {  
    "token": "jumped",  
    "start_offset": 20, "end_offset": 26, "type": "<ALPHANUM>", "position": 5  
  }, {  
    "token": "over",  
    "start_offset": 27, "end_offset": 31, "type": "<ALPHANUM>", "position": 6  
  }, {  
    "token" : "lazy",  
    "start_offset": 36, "end_offset": 40, "type": "<ALPHANUM>", "position": 8  
  }, {  
    "token" : "dog",  
    "start_offset": 41, "end_offset": 44, "type": "<ALPHANUM>", "position": 9  
  } ] }
```


Analyseur whitespace

```
$ curl -XPOST 'localhost:9200/test/_analyze?analyzer=whitespace&pretty=1' -d  
'The quick brown fox jumped over the lazy Dog'
```



```
{  
  "tokens" : [ {  
    "token" : "The", ...  
  }, {  
    "token" : "quick", ...  
  }, {  
    "token" : "brown", ...  
  }, {  
    "token" : "fox", ...  
  }, {  
    "token" : "jumped", ...  
  }, {  
    "token" : "over", ...  
  }, {  
    "token" : "the", ...  
  }, {  
    "token" : "lazy", ...  
  }, {  
    "token" : "Dog", ...  
  } ] }
```


Un analyseur

Un ensemble
de tokenizers et
de filtres

Un tokenizer

- Découpe une chaîne en « mots » et transforme :
- whitespace tokenizer :

"the dog!" -> "the", "dog!"

- standard tokenizer :

"the dog!" -> "the", "dog"

Un filtre

- Supprime ou transforme un token :

- asciifolding filter :

éléphant -> elephant

- stemmer filter (french) :

elephants -> "eleph"

cheval -> "cheval"

chevaux -> "cheval"

- phonetic (plugin) :

quick -> "Q200"

quik -> "Q200"

Analyzer

```
"analysis":{
  "analyzer":{
    "francais":{
      "type":"custom",
      "tokenizer":"standard",
      "filter":["lowercase", "stop_francais", "fr_stemmer", "asciifolding", "elision"]
    }
  },
  "filter":{
    "stop_francais":{
      "type":"stop",
      "stopwords":["_french_", "twitter"]
    },
    "fr_stemmer" : {
      "type" : "stemmer",
      "name" : "french"
    },
    "elision" : {
      "type" : "elision",
      "articles" : ["l", "m", "t", "qu", "n", "s", "j", "d"]
    }
  }
}
```


Mapping

```
"type1" : {  
  "properties" : {  
    "text1" : { "type" : "string", "analyzer" : "simple" },  
    "text2" : { "type" : "string", "index_analyzer" : "simple",  
               "search_analyzer" : "standard"  
    },  
    "text3" : {  
      "type" : "multi_field",  
      "fields" : {  
        "text3" : {  
          "type" : "string",  
          "analyzer" : "standard"  
        },  
        "ngram" : {  
          "type" : "string",  
          "analyzer" : "ngram"  
        },  
        "soundex" : {  
          "type" : "string",  
          "analyzer" : "soundex"  
        }  
      }  
    }  
  }  
}
```


Les types

- string
- integer / long
- float / double
- boolean
- null
- array
- objects
- multi_field
- ip
- geo_point
- geo_shape
- binary
- attachment (plugin)

Champs spéciaux

- `_all` (et `include_in_all`)
- `_source`
- `_ttl`
- `parent / child`
- `nested`

Autres fonctionnalités

- highlighting
- scoring
- sort
- explain
- multi get / multi search
- bulk

Démonstrations

www.scrutmydocs.org

CURL est ton ami !

JAVA est aussi ton ami !



La communauté



elasticsearch.

bluekiwi
enterprise social software



lusini.de



mozilla
FOUNDATION

presspeople



foursquare

fotopedia

salesforce
desk

Promovacances.com

KLOUT BETA



cuestamenos.com

infochimps



Cadremploi.fr



atlas
the global video to audio index

b:
bazaarvoice



JEMPLAZIK.COM

sonian
Cloud Powered Archiving and Search



ataxo

fotolia

LesFurets.com
Comparez futé

InSTEDD

maptimize

JBoss
Community

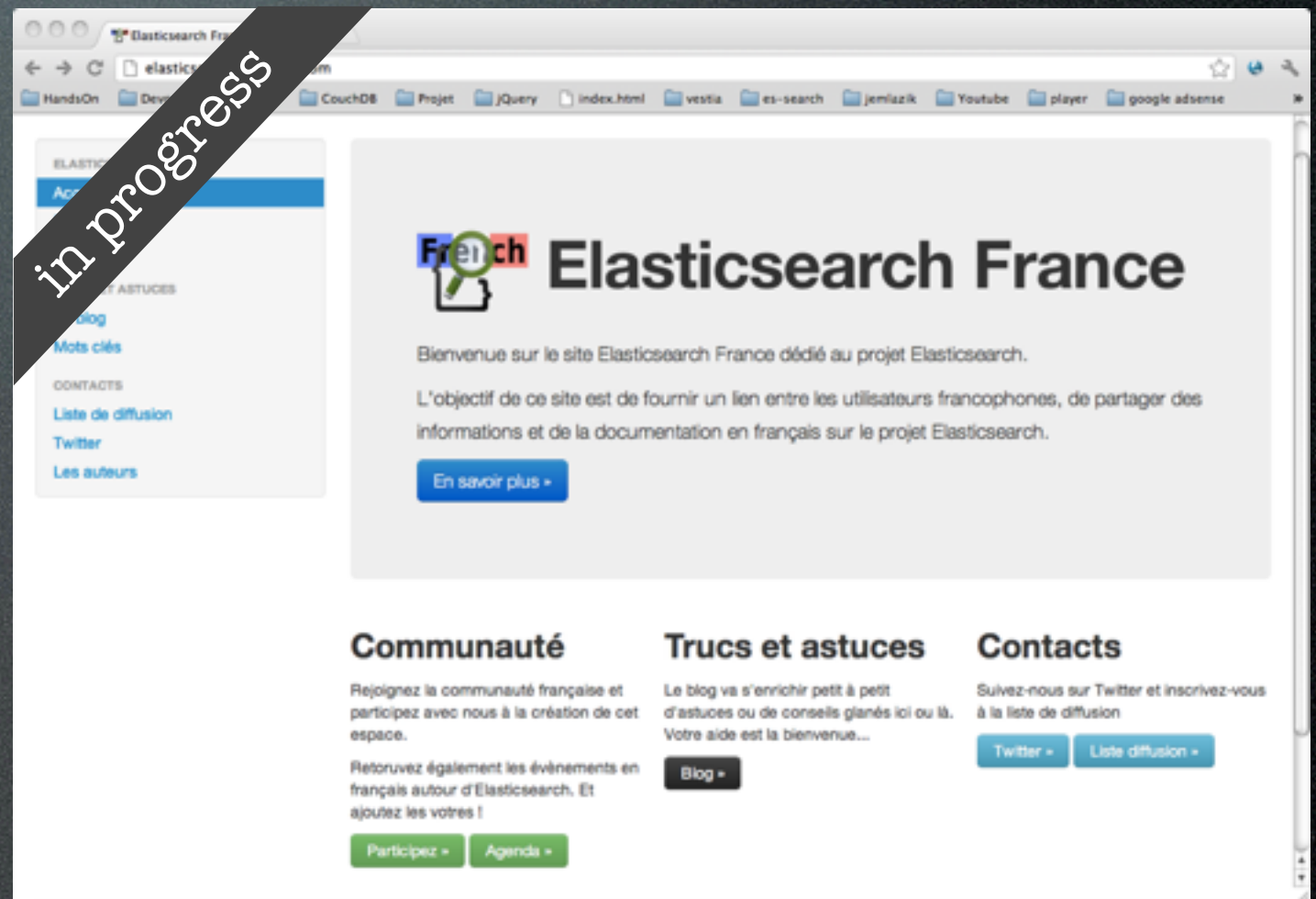
~ 100 contributeurs directs au projet (+ de 2800 watchers et + de 470 forks)

Rejoignez le mouvement !

@ElasticsearchFR



www.elasticsearch.fr



Questions ?

Posez aussi vos questions sur elasticsearch-fr@googlegroups.com



Slides sur <http://fr.slideshare.net/dadoonet>

Sources sur <https://github.com/dadoonet/talks>

