MongoDB et ElasticSearch
L'analyse de la polarité de tweets

Nicolas Dugué
nicolas.dugue@univ-orleans.fr

M2 MIAGE
Systèmes d'information répartis
14 octobre 2014

Plan

Des tweets


```
{
   "date":"2009-04-20T06:40:11Z",
   "query":"NO_QUERY",
   "user":"Tom_1994",
   "tweet":"Noooooooooooooo!!!!!! School
   today. But the worst part is that I
   wont be able to tweet troughout the
   day"
}
```


API Streaming Twitter

- Obtenir de grandes quantités de tweets rapidement ;
- Des tweets répondant à une requête particulière ;
- De nombreuses librairies : Twitter4j en Java.

L'analyse de sentiment

Être capable d'analyser à partir d'un corpus de textes les avis, sentiments exprimés dans ces textes.

Un outil puissant

Synthesio, Trendybuzz ou AMI Software
$\longrightarrow$ Comprendre l'image d'une marque, d'un produit à travers l'avis des clients, usagers.

L'analyse de polarité

La base de l'analyse de sentiment

Être capable d'analyser à partir d'un corpus de textes si les avis, sentiments exprimés sont positifs, négatifs ou neutres.

Etiqueter le tweet


```
{
   "date":"2009-04-20T06:40:11Z",
   "query":"NO_QUERY",
   "user":"Tom_1994",
   "tweet":"Noooooooooooooo!!!!!! School
   today. But the worst part is that I
   wont be able to tweet troughout the
   day",
   "sentiment":0
}
```

Objectifs du TP

Stocker les tweets

- Des tweets retournés au format JSON par les APIs ;
- Un schéma qui peut évoluer ;
- Aucun besoin de cohérence, mais un besoin de rapidité ;
- Stocker et requêter efficacement de grands volumes ;
- Mettre à jour et manipuler les tweets.

$\longrightarrow$ MongoDB, BDD NOSQL orientée Document

Objectifs du TP

Fouiller les tweets

- Interroger les tweets ;
- Rechercher dans le contenu texte des tweets ;
- Indexer et requêter efficacement de grands volumes ;
- Présenter visuellement les résultats.

$\longrightarrow$ ElasticSearch, moteur de recherche basé sur Apache Lucene, librairie d'indexation et de recherche d'information
$\longrightarrow$ Kibana, création de Dashboard pour documents indexés dans ES

Plan

Pourquoi MongoDB ?

Une BDD matûre

- Yandex, Ebay, McAfee, Adobe, Craiglist, ... ;
- Une doc très fournie et une grande communauté ;

Pourquoi MongoDB ?

Une BDD matûre

- Yandex, Ebay, McAfee, Adobe, Craiglist, ... ;
- Une doc très fournie et une grande communauté ;

Adaptation facile

- Orientée document - format JSON ;
- SQL $\rightarrow$ MongoDB : Easyyy ;
- Utilise Javascript ;

Pourquoi MongoDB ?

Une BDD matûre

- Yandex, Ebay, McAfee, Adobe, Craiglist, ... ;
- Une doc très fournie et une grande communauté ;

Adaptation facile

- Orientée document - format JSON ;
- SQL $\rightarrow$ MongoDB : Easyyy ;
- Utilise Javascript ;

Scalable

- MapReduce en natif ou avec le connecteur ;
- Réplication, sharding automatique ;

Surement d'autres trucs cools !

Plan

MongoDB - Les bases

Le vocabulaire

table = collection
entrée, ligne, tuple = document

Exemple - Création, insertion

> use db_name
> db.createCollection(”collection_name”)
> db.collection_name.insert({field1 : ”value_string”, field2 : value_int, ..., fieldN: [arrayV1, ..., arrayVn]})

MongoDB - Les bases

Exemple - Modification

```
> db.collection_name.update( { _id: 1 },
{
$inc: { field1: 5 },
$set: { field2: "ABC123" }
} )
```

Exemple - Suppression

```
> db.collection_name.remove({ }) //Delete the collection
> db.collection_name.remove({field1 : value}) //Delete doc that match
```

MongoDB - Les requêtes

Exemple - Select, Projection

```
> db.collection_name.find( { }, { field1: 1, field2: 0 } )
```

Exemple - Where

```
> db.collection_name.find( { field: { $gt: v1, $lt: v2 } } ); // where v1 < field < v2
> db.collection_name.find( { field: value } ); // where field == value
```

MongoDB - Les requêtes

Exemple - Order By équivalent

> db.collection_name.find().sort({ field1: 1, field2: -1 }) // Order by field1 ASC, field2 DESC

Exemple - Limit

> db.collection_name.find().limit(1) // Un doc

MongoDB - Les requêtes

Exemple - Group By

> db.collection_name.aggregate([{ $group : { _id : "$field"} }]) // Group by field

> db.collection_name.aggregate([{ $group : { _id : "$user", count: { $sum: 1 } } } , { $match : {count : { $gte : 10 } } } }]) // Group by field having count > 10

> db.collection_name.aggregate([{ $group : { _id : "$user", count: { $sum: 1 } } } , { $match : {count : { $gte : 10 } } }, {$sort : {count : −1 } } }])

MongoDB - Les requêtes

Exemple - Cursor

```
var cursor = db.collection_name.find()
while (cursor.hasNext()) {
var doc = cursor.next();
db.collection_name.update({ _id : doc._id}, {$set : {field : value} })
}
```

Plan

MongoDB - Monitoring

Commande `mongostat`
Commande `mongotop`
Interface HTTP `http://localhost:28017`

MongoDB - Moteur d'indexation

Indexer les données

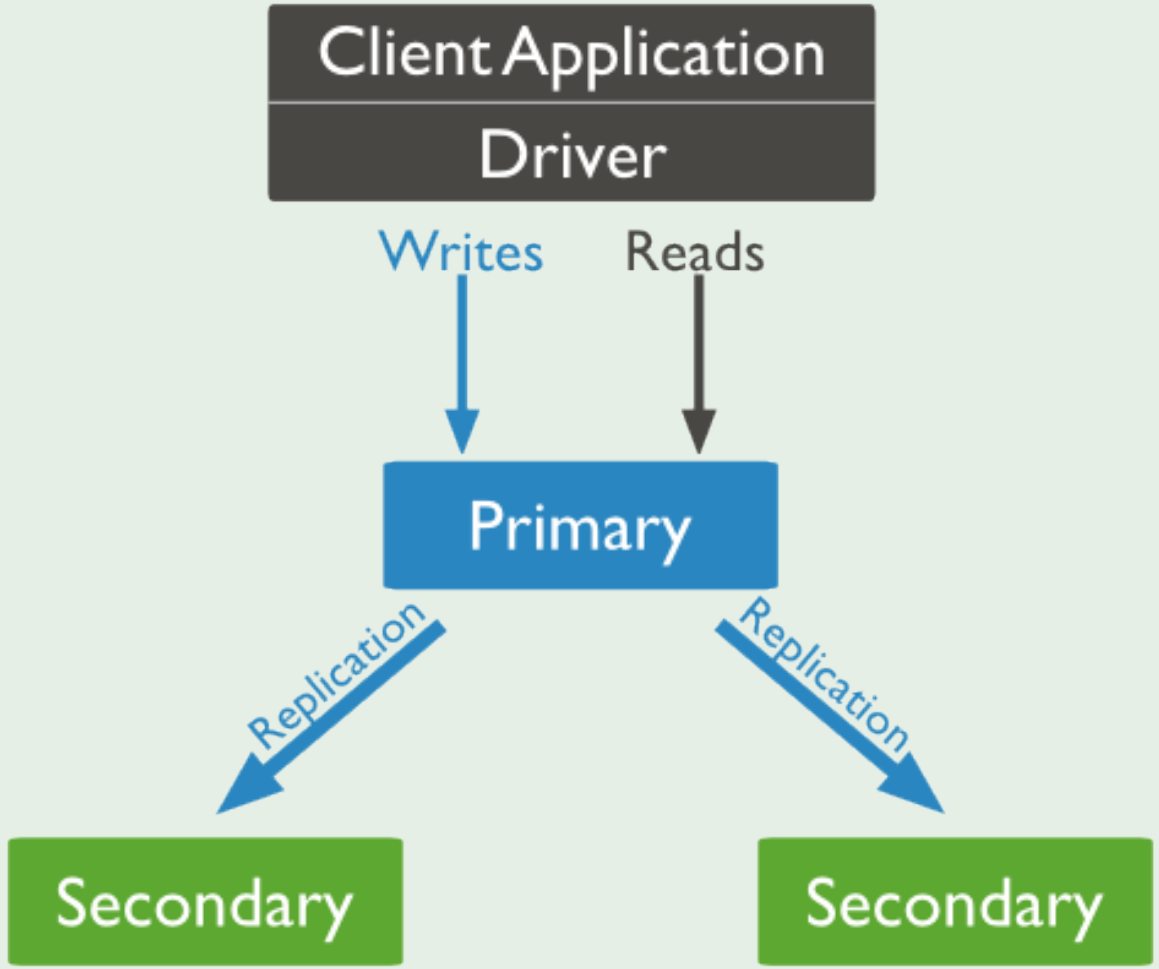
db.collection_name.ensureIndex({ field: 1}, {background: boolean})

Impact sur les performances

Augmente l'efficacité de la recherche
Ralentit l'insertion
Augmente l'espace requis

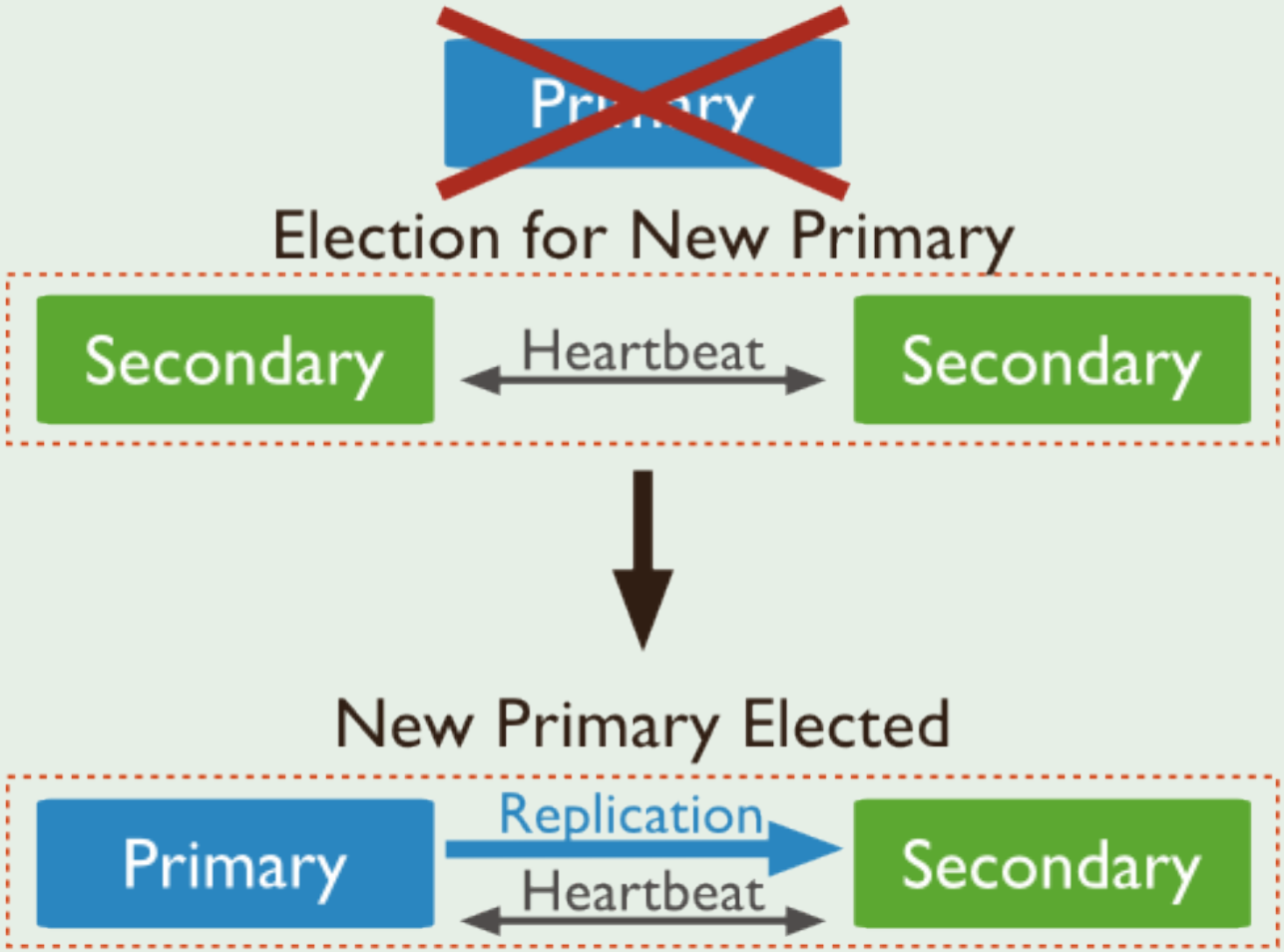
MongoDB - Réplication

Par défaut



MongoDB - Failover

Par défaut



MongoDB - Réplication

Read preferences

- primary - Default mode - All operations read from the current replica set primary.
- primaryPreferred - In most situations, operations read from the primary but if it is unavailable, operations read from secondary members.
- nearest - Operations read from member of the replica set with the least network latency, irrespective of the member's type.

MongoDB - Cohérence des données

Unacknowledged

On ne sait pas si le requête d'écriture est reçue

MongoDB - Cohérence des données

Unacknowledged

On ne sait pas si le requête d'écriture est reçue

Acknowledged

Mode **par défaut** :
On ne sait que la requête d'écriture est reçue, on ne sait pas si elle est persistée

MongoDB - Cohérence des données

Journaled

Le requête est reçue et persistée dans le journal de la DB, elle est donc persistée, mais pas sur tout le cluster

MongoDB - Cohérence des données

Journaled

Le requête est reçue et persistée dans le journal de la DB, elle est donc persistée, mais pas sur tout le cluster

Replica Acknowledged

La requête d'écriture est forcément propagée aux noeuds du cluster répliqué

Plan

ElasticSearch (ES)

- Stocker et indexer les documents
- API REST : PUT, POST, GET, DELETE, HEAD
- Moteur de recherche

ES - Création d'un index (BDD)

```
curl -XPUT "localhost:9200/_river/tw_sentiment/_meta"
{
    "type": "mongodb",
    "mongodb": {
        "servers": [
            { "host": "127.0.0.1", "port": 27017 }
        ],
        "options": { "secondary_read_preference": true
        "db": "infra_prod", //Nom de la db mongo
        "collection": "tweets_sentiment" //Nom de la c
    },
    "index": {
        "name": "tw_sentiment", //Nom de l?index
        "type": "tweet" //Type des documents
    }
}?
```

Plan

ES - Les recherches

```
curl -XGET 'http://localhost:9200/twitter/_search'
curl -XGET 'http://localhost:9200/twitter/_search?q=field:value'
```

ES - Les aggrégations

```
{
    "aggs" : {
        "aggs_name" : {
            "terms" : { "field" : "field_name" }
        }
    }
}
```

Plan

ES - Le mapping

```
{
    "tw_sentiment": {
        "mappings": {
            "tweet": {
                "properties": {
                    "date": {
                        "type": "date",
                        "format": "dateOptionalTime"
                    },
                    "sentiment": {
                        "type": "long"
                    },
                    "tweet": {
                        "type": "string"
                    },
                    "tweet_id": {
```

ES - Un analyzer pour la langue anglaise

```
PUT /english_docs {
    "settings": {
        "analysis": {
            "analyzer": {
                "es_std": {
                    "type":      "standard",
                    "stopwords": "_english_"
                }
            }
        }
    }
}
```

ES - Un analyzer de mots négatifs

```
PUT /classifieur_negs_stopwords {
"settings" : {
        "analysis" : {
            "char_filter" : {
              "negs" : {
                  "type" : "pattern_replace",
                  "pattern" : "\\b((?i:never|no)\\b",
                  "replacement" : "NEG"  } },
            "analyzer" : {
              "negcon_tagger" : {
                  "type" : "custom",
                  "tokenizer" : "whitespace",
                  "filter" : ["lowercase", "kstem"],
                  "char_filter" : ["negs"] }
            }   } } }
```

Plan

Conclusion

- MongoDB pour stocker les tweets et les mettre à jour
- Cluster vertical ES qui contient la réplique de MongoDB
- ES + Kibana pour présenter les résultats