NoSQL Databases

Polytech Nice-Sophia, SI4

Pierre Monnin

pierre.monnin@inria.fr

Special thanks to **Dr. Sabeur Aridhi**, **Dr. Jörn Franke** and **Dr. Nicolas Travers** for providing their materials

Chapter 3:
Document Databases
ElasticSearch & Kibana

Source: https://perso.univ-rennes1.fr/pierre.nerzic/Hadoop/semaine6.pdf

ElasticSearch

- Base de données documents
- Orientée vers l'indexation et la recherche rapide

- Notamment recherche floue sur du texte

➔ Retrouver des documents à partir de mots-clefs, regex, fragments de textes

  - curl -XGET 'http://localhost:9200/_search?q=genration~'
  - REGEX
  - text similarity

- Utilise le moteur Apache Lucene en interne

https://lucene.apache.org/core/

  - API Java libre permettant d'indexer des documents.
  - Index qu'elle produit ~20-30 % de la taille des documents indexés
  - Toutes sortes de recherches sont possibles, et les résultats sont classés par pertinence (score)

- Offre une API REST (plutôt que de programmer en Java)

- ES Scalable et installable sur un cluster de machines

- Associé à des outils pour exploiter diverses données : graphiques, ML, etc.

ElasticSearch: interface REST

Syntaxe générale d'une URL ElasticSearch : http://<host>:<port>/[index]/[type]/[_action/id]

Pour indexer une donnée

```
curl -X POST "localhost:9200/magasin/_doc/1" -H 'Content-Type: application/json' -d'
{
  "nom": "poire",
  "type": "FRUIT"
}
'
```

Pour récupérer une donnée

```
curl -X GET http://localhost:9200/magasin/_doc/1
```

Pour supprimer une donnée

```
curl -X DELETE http://localhost:9200/magasin/_doc/1
```

Pour rechercher une donnée

```
curl -X POST "http://localhost:9200/magasin/_search" -H 'Content-Type: application/json' -d'{
  "query": {
    "query_string": {
      "query": "p*"
    }
  }
}'
```

ElasticSearch: documents à indexer ?

Pour indexer une donnée

```
curl -X POST "localhost:9200/magasin/_doc/1" -H 'Content-Type: application/json' -d'
{
  "nom": "poire",
  "type": "FRUIT"
}
'
```

Documents

- Peuvent être du texte simple
- Des documents JSON

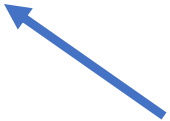
Index ES pour stocker des documents
Double sens : table d'une BDD et index d'une BDD

- Chaque propriété est indexée
- Appelés index inversés dans ES (BDD : index sur une colonne)
- Liste de tous les mots présents

ElasticSearch: schéma d'un index ?

Pour indexer une donnée

```
curl -X POST "localhost:9200/magasin/_doc/1" -H 'Content-Type: application/json' -d'
{
  "nom": "poire",
  "type": "FRUIT"
}
'
```



Index ES pour stocker des documents
Double sens : table d'une BDD et index d'une BDD

- Ajout du premier document ➔ ES déduit les types des champs automatiquement

/!\ pas nécessairement correct!

- Vocabulaire ES : *mapping* (~schema d'un index en BDD relationnelle)
- Nombreux types de champs
  - Texte : *text, keyword*
  - Numérique : *long, integer, double, float, …*
  - Divers : *boolean, date, geo_point, ip*
  - Tableau, objet JSON, …
- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)
- **Possible de fournir le mapping**

ElasticSearch: schéma d'un index ?

Pour indexer une donnée

```
curl -X POST "localhost:9200/magasin/_doc/1" -H 'Content-Type: application/json' -d'
{
  "nom": "poire",
  "type": "FRUIT"
}
'
```

Index ES pour stocker des documents
Double sens : table d'une BDD et index d'une BDD

- Ajout du premier document ➔ ES déduit les types des champs automatiquement

/!\ pas nécessairement correct!

- Vocabulaire ES : *mapping* (~schema d'un index en BDD relationnelle)
- Nombreux types de champs
  - Texte : *text, keyword*
  - Numérique : *long, integer, double, float, …*
  - Divers : *boolean, date, geo_point, ip*
  - Tableau, objet JSON, …
- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)
- **Possible de fournir le mapping**

text: texte quelconque, découpé en mots, chacun indexé, recherches possibles sur des extraits

keyword: texte mémorisé en tant qu'entité unique, recherche uniquement possible sur entité entière, et non extraits

ElasticSearch: schéma d'un index ?

Pour indexer une donnée

```
curl -X POST "localhost:9200/magasin/_doc/1" -H 'Content-Type: application/json' -d'
{
  "nom": "poire",
  "type": "FRUIT"
}
'
```

Index ES pour stocker des documents
Double sens : table d'une BDD et index d'une BDD

- Ajout du premier document ➔ ES déduit les types des champs automatiquement

/!\ pas nécessairement correct!

- Vocabulaire ES : *mapping* (~schema d'un index en BDD relationnelle)
- Nombreux types de champs
  - Texte : *text, keyword*
  - Numérique : *long, integer, double, float, …*
  - Divers : *boolean, date, geo_point, ip*
  - Tableau, objet JSON, …
- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)
- **Possible de fournir le mapping**

text: texte quelconque, découpé en mots, chacun indexé, recherches possibles sur des extraits

keyword: texte mémorisé en tant qu'entité unique, recherche uniquement possible sur entité entière, et non extraits

ElasticSearch: schéma d'un index ?

Pour indexer une donnée

```
curl -X POST "localhost:9200/magasin/_doc/1" -H 'Content-Type: application/json' -d'
{
  "nom": "poire",
  "type": "FRUIT"
}
'
```

Index ES pour stocker des documents
Double sens : table d'une BDD et index d'une BDD

- Ajout du premier document ➔ ES déduit les types des champs automatiquement

/!\ pas nécessairement correct!

- Vocabulaire ES : *mapping* (~schema d'un index en BDD relationnelle)
- Nombreux types de champs
  - Texte : *text, keyword*
  - Numérique : *long, integer, double, float, …*
  - Divers : *boolean, date, geo_point, ip*
  - Tableau, objet JSON, …
- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)
- **Possible de fournir le mapping**

Texte quelconque

text: texte quelconque, découpé en mots, chacun indexé, recherches possibles sur des extraits

keyword: texte mémorisé en tant qu'entité unique, recherche uniquement possible sur entité entière, et non extraits

Elements Ids (mail, tel, etc.)

ElasticSearch: indexing du texte

- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)

Texte quelconque

text: texte quelconque, découpé en mots, chacun indexé, recherches possibles sur des extraits

keyword: texte mémorisé en tant qu'entité unique, recherche uniquement possible sur entité entière, et non extraits

Elements Ids (mail, tel, etc.)

Types de films :
« Action américaine »
« Drama américaine »

Quels choix ?

- *keyword*: pas possible de chercher .*américain.*
- *text:* rechercher une chaine exacte va quand même mener ES à proposer des chaines proches « Action sud-américaine »

(avec un score de pertinence moindre)

ElasticSearch: indexing du texte

- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)

Texte quelconque

text: texte quelconque, découpé en mots, chacun indexé, recherches possibles sur des extraits

keyword: texte mémorisé en tant qu'entité unique, recherche uniquement possible sur entité entière, et non extraits

Elements Ids (mail, tel, etc.)

Types de films :
« Action américaine »
« Drama américaine »

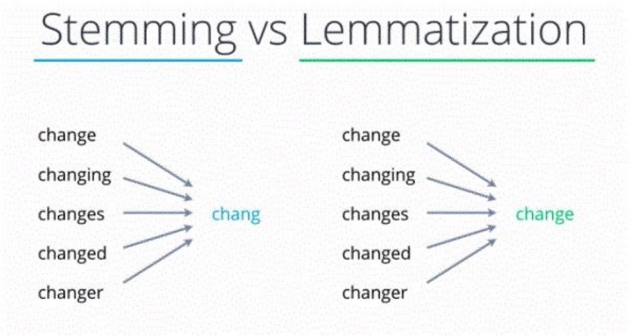
Quels choix ?

- *keyword*: pas possible de chercher .*américain.*
- *text:* rechercher une chaine exacte va quand même mener ES à proposer des chaines proches « Action sud-américaine »

(avec un score de pertinence moindre)

➔ Possible d'associer plusieurs types au même champ (voir la doc)

ElasticSearch: indexing du texte



- **Les types influent sur les capacités d'indexage et de requêtes** (voir TP)

Texte quelconque

text: texte quelconque, découpé en mots, chacun indexé, recherches possibles sur des extraits

keyword: texte mémorisé en tant qu'entité unique, recherche uniquement possible sur entité entière, et non extraits

Elements Ids (mail, tel, etc.)

Types de films :
« Action américaine »
« Drama américaine »

Quels choix ?

- *keyword*: pas possible de chercher .*américain.*
- *text:* rechercher une chaine exacte va quand même mener ES à proposer des chaines proches « Action sud-américaine »

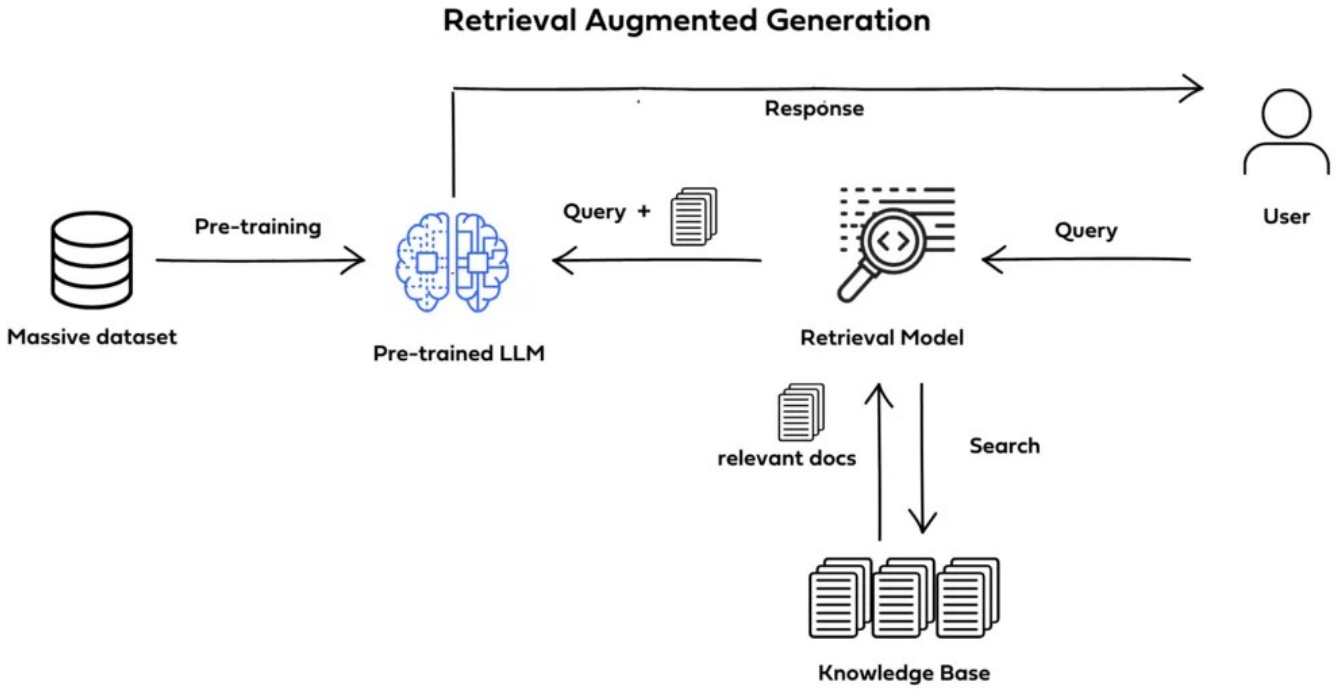
(avec un score de pertinence moindre)

➔ Possible d'associer plusieurs types au même champ (voir la doc)

Indexing du text avec differents analyzers : découpage des mots, stopwords, stemming, etc.
Possibilité de définir les analysers et de donner la langue du texte pour aider ES à choisir les outils adéquats

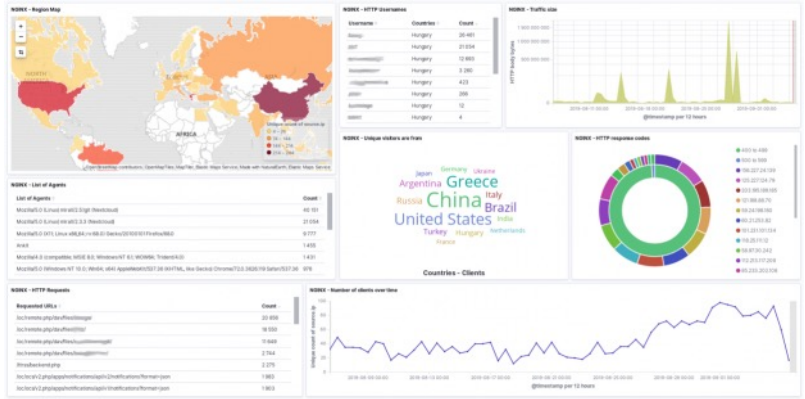
Applications

- Search engines (+ other techniques)
- RAG: Retrieval Augmented Generation

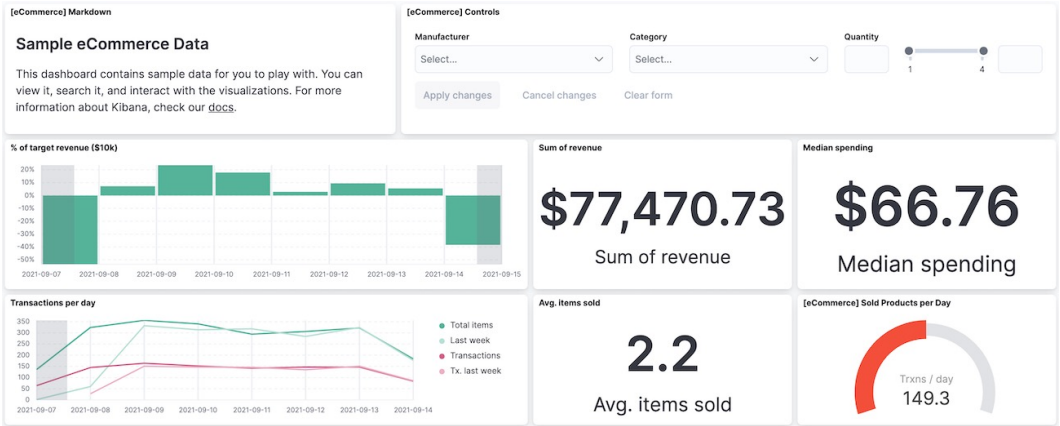


Kibana

- Interface Web permettant de réaliser tous les traitements vus plus facilement qu'avec cURL
- Outils de visualisation
- Nombreux plugins et intégrations supplémentaires



Source: https://balagetech.com/visualizing-nginx-access-logs-kibana/



Source: https://www.elastic.co/guide/en/kibana/7.17/dashboard.html