Bases de données documentaires et distribuées, http://b3d.bdpedia.fr

Indexation avec Elasticsearch

Schéma de l'index

Le schéma donne la liste de tous les champs d'un document

Nombreuses options :

- type (numérique, entier)
- possibilité de calcul de la valeur du champ à partir d'un autre
- traitements divers sur les valeurs du champ

Reprenons un document bien connu


```
{
  "title": "Vertigo",
  "year": 1958,
  "genre": "drama",
  "summary": "Scottie Ferguson, ancien inspecteur de police... ",
  "director":         {
    "_id": "artist:3",
    "last_name": "Hitchcock",
    "first_name": "Alfred",
    "birth_date": "1899"
  },
  "actors": [
     {
       "_id": "artist:15",
       "first_name": "James",
       "last_name": "Stewart",
       "birth_date": "1908",
       "role": "John Ferguson"
     }
  ]
}
```

Son schéma

```
{ "mappings": {
    "movie": {
      "_all":       { "enabled": true  },
      "properties": {
        "title":    { "type": "string"  },
        "year":     { "type":   "date", "format": "yyyy" },
        "genre":      { "type": "string" },
        "summary": { "type": "string"},
        "director":     {
          "properties": {
            "_id": { "type":"string" },
            "last_name": { "type":"string"},
            "first_name": { "type":"string"},
            "birth_date": { "type": "date", "format":"yyyy"}
          }},
        "actors": {
          "type": "nested",
          "properties": {"..."}
        }
      }
    }
}
```

Définition des types et de la clé

Chaque champ utilisé dans un document doit apparaître dans un des éléments. Sinon, ElasticSearch propose un "Dynamic Mapping"

ElasticSearch fournit tout un ensemble de types pré-définis pour les besoins courants ;

Les options indiquent d'éventuels traitements à appliquer à chaque valeur du type avant son insertion dans l'index

Ex : type **string** (remplacé par **text** depuis 5.0)

- on lui définit un "analyzer" "standard"
- traite le texte pour une recherche plein-texte
- retenir : permet d'indexer chaque mot, et donc de faire des recherches sur toutes les combinaisons de mots

Définition des champs

Le **nom** et le **type** sont les informations de base

```
{
  "genre": {
                "type": "string",
                "index": "no",
                "store": "true",
                "index_options": "offsets"
    }
}
```

Ensuite, divers attributs (souvent optionnels) :

- **index** indique simplement si le champ peut être utilisé dans une recherche ;
- **store** indique si la **valeur** du champ est **stockée** dans l'index,

L'option **store** permet de considérer ElasticSearch comme une base de données indexées.

Définition des champs

Les attributs **index** et **store** sont très importants

- **index=true**, **store=false** : on pourra interroger le champ, mais il faudra accéder au document principal dans la base documentaire si on veut sa valeur ;
- **index=true**, **store=true** : on pourra interroger le champ, **et** accéder à sa valeur dans l'index ;
- **index=false**, **store=true** : on ne peut pas interroger le champ, mais on peut récupérer sa valeur dans l'index.

index=false, **store=false** : n'a pas de sens à priori.

Définition des champs (suite)

Comment peut-on indexer un champ sans le stocker ?

- c'est notamment le cas pour les textes qui sont décomposés en **termes** : chaque terme est indexé indépendamment
- très difficile pour l'index de reconstituer le texte

Doit-on stocker les documents dans l'index ? Question de compromis :

- (-) **stocker** une valeur prend plus d'espace que **l'indexer** ; À l'extrême, la base documentaire est dupliquée dans l'index
- (+) Peut considérablement simplifier l'architecture

Attention aux autres facteurs, comme la latence pour les insertions et mises à jour.

Champ par défaut, document original

Le champ **_source** contient le document, mais n'est pas indexé. Il est possible ainsi de récupérer la totalité du document passé à l'indexation.

Le champ **_all** contient la concaténation de tous les autres champs ; il est indexé, pas stocké.

Permet de retrouver les documents contenant une valeur, sans savoir dans quel champ elle se trouve.

Configuration des pré-traitements

Analyseur = une chaîne de traitement constituée de tokeniseurs et de filtres

```
curl -XPUT '<url>/monindex' -H 'Content-Type: application/json' -d @settings.json
```

Contenu de `settings.json` :

```
{"settings": {
    "analysis": {
      "analyzer": {
        "custom_lowercase_stemmed": {
          "tokenizer": "standard",
          "filter": [
            "lowercase",
            "custom_english_stemmer"
          ]
        }
      }
    }
}
```

Test d'un analyseur

On peut tester un analyseur en lui soumettant un texte-test.

```
curl -XPUT '<url>/monindex/_analyze' -H 'Content-Type: application/json' -d @test.json
```

Contenu de `test.json` :

```
{
        "analyzer": "custom_lowercase_stemmed",
        "text":"Finiras-tu ces analyses demain ?"
}
```

Résultat d'un test


```
{
  "tokens" : [ {
    "token" : "finira",
    "start_offset" : 0,
    "end_offset" : 7,
    "type" : "<ALPHANUM>",
    "position" : 0
  }, {
    "token" : "tu",
    "start_offset" : 8,
    "end_offset" : 10,
    "type" : "<ALPHANUM>",
    "position" : 1
  },
  "...": "..."
}
```

Résumé

La configuration d'un moteur de recherche, c'est :

- Définir le schéma des champs indexés (stockage / indexation / les deux)
- Définir les chaînes de traitement.

C'est **aussi** beaucoup d'autres choses.

- Définir les synonymes
- Définir les mots protégés (acronymes)
- Définir les mots vides
- Acquérir et exploiter les actions utilisateurs

À pratiquer et approfondir (projet).