

Formation ElasticSearch

Ihab ABADI / UTOPIOS

SOMMAIRE

- 1 – Présentation de ElasticSearch et l'écosystème ELK.
- 2 – Lucene comme base.
- 3 – Concurrent et comparaison.
- 4 – Installation et Configuration.
- 5 – Eléments et opérations de base d'ElasticSearch.
- 6 – Tours de l'API Rest
- 7 – Utilisation des Mappings
- 8 – Utilisation des Templates
- 9 – Mécanisme de recherche avec DSL
- 10 – Mécanisme d'agrégations.
- 11 – Pertinence des recherches et action sur le score
- 12 – Utilisation des Analyzers lors de l'indexation
- 13 – Utilisation des suggesters
- 14 – Les Highligthing
- 15 – Les Percolators
- 16 – Utilisation des scripts pour les filtres et le scoring
- 17 – Introduction à la visualisation des données avec Kibana

Présentation d'ElasticSearch Et ELK

- ElasticSearch est une base de données non relationnelles
- ElasticSearch est orienté document
- ElasticSearch est distribué
- ElasticSearch est openSource
- ElasticSearch est sans schéma
- ElasticSearch est basé sur Le moteur de recherche Lucene
- ElasticSearch fait parti de un écosystème ELK pour intégration, visualisation et recherche sur des documents (ElasticSearch, Logstash, Kibana)

Présentation d'ElasticSearch Et ELK

- Logstash est une application qui permet la lecture et la transformation le transfert de toutes types de données vers ElasticSearch
- Logstash utilise un mécanisme de pipeline :
 - Entrée
 - Filtre
 - Sortie
- Ingérer des données dans ElasticSearch peut se faire également par
 - Elastic Beats
 - Clients Langage

Présentation d'ElasticSearch Et ELK

- Kibana est l'outils de visualisation FrontEnd des données dans la suite ELK
- Kibana offre la possibilité d'effectuer des recherches et personnaliser l'affichage à l'aide de tableaux de bord et de diagrammes
- Kibana est gratuit
- Kibana peut être étendu avec des plugins (Développement en javascript et nodejs)

ElasticSearch et Lucene

- Lucene est une API Open Source développée en JAVA
- Lucene permet l'indexation et la recherche sur tous types de documents
- Lucene utilise un mécanisme Near Real Time
- Lucene est utilisée un mécanisme de score sur 3 axes

Concurrent et Comparaison

- ElasticSearch est une base de données assez jeunes
- D'autre base de données basées sur Lucene existe
- Concurrent le plus connu est Solr, qui est plus ancien
- Il existe beaucoup de points commun et de différence entre solr et ElasticSearch
- Solr VS ElasticSearch

Quelques cas d'utilisation de ElasticSearch

- Recherche dans une application
- Recherche sur site web
- APM
- Monitoring d'infrastructure
- Analyse de sécurité
- ...

ElasticSearch Installation et configuration

- ElasticSearch peut être installé en cluster
- La mise en place d'un cluster ElasticSearch est assez simple comme tâche
- ElasticSearch a besoin d'installation des composants sur chaque nœud du cluster
- ElasticSearch a besoin de JAVA
- La configuration d'elasticsearch se fait par un fichier `elasticsearch.yml`
- La configuration de la JVM pour répondre au prérequis d'`elasticsearch` se fait par `jvm.options`

ElasticSearch Installation et configuration

- Pour les besoins de la formation :
- On utilisera des images de docker (elasticsearch et kibana)
- On utilisera docker-compose pour démarrer les services
- Un dépôt git pour échanger les sources

Eléments et Opérations de base ElasticSearch

- Rappel terminologie
- Cluster : Définition d'un ensemble d'index répartis sur un ou plusieurs nodes
- Node : Instance d'elasticsearch, un node par machine
- Shared : Instance de Lucene, les shareds sont distribués sur les différents nodes
- Replica : Les copies des shareds
- Index : Conteneur de données
- Type : catégorie d'éléments à indexer
- Document : une donnée
- Field : Paire clé-valeur

Eléments et Opérations de base ElasticSearch

- ElasticSearch expose une Api Rest pour exécuter les différentes actions
- Les instructions ainsi que les réponses sous format json.

Eléments et Opérations de base ElasticSearch

- API Pour la gestion d'index.
- La création et la mise à jour d'index se fait par le verb Put et comme paramètre principal le nom de l'index
- La création accepte plusieurs paramètres de configuration :
- Quelque exemple:
- Nombre de shared
- Nombre de replicas
- Alias
- Mapping
- Templates
-

Eléments et Opérations de base ElasticSearch

- Exemple de création index sans configuration
- Exemple de création index avec configuration
- (Alias, shared, replicas..)
- Exemple de suppression index
- Exemple de mise à jour de configuration d'index
- Référence Api Index : <https://www.elastic.co/guide/en/elasticsearch/reference/current/indices.html>

Eléments et Opérations de base ElasticSearch

- Pour ajouter des données à l'aide de l'API On peut utiliser :
- Le verb POST sur notre index, avec le paramètre _doc pour ajouter un document.
- Le verb PUT sur notre index, avec le paramètre _bulk pour ajouter plusieurs documents.
- Exemple d'ajout d'un ou plusieurs documents

Eléments et Opérations de base ElasticSearch

- Pour rechercher des données, on utilise L'API search.
- Cette API accepte une multitude de paramètres pour la recherche.
- Exemple de base de recherche

API ElasticSearch

- ElasticSearch propose une multitude de EndPoints pour exécuter toute sorte de fonctionnalités.
- API pour ingestion de données par pipeline
- API pour gérer le cluster
- API pour la sécurité
-

Mapping en ElasticSearch

- Le mapping est le mécanisme qui permet de définir les documents d'un index.
- Mapping définit les différents Fieds, (Type de données, façon d'être index, leur pertinence lors des recherches...)
- Le mapping peut être dynamique.
- Le mapping peut être explicite
- Chaque index possède son propre Mapping

Mapping en ElasticSearch (Dynamique)

- Le mapping dynamique est le mécanisme qui permet d'ajouter un field au mapping de l'index automatiquement.
- A l'ajout d'un document, elasticSearch détecte un nouveau field et essaye de trouver le bon type de données.
- Avantage : rapidité d'utilisation si on commence à utiliser ElasticSearch.
- Inconvénient : On peut rapidement se trouver avec un nombre de Fields élevé pour notre Index, qui peut causer des problèmes de mémoire.
- Exemple de mapping dynamique.

Mapping en ElasticSearch (Dynamique)

- Quelque exemple de type que ElasticSearch détecte automatiquement:
- Boolean, float, long, object, date, text.
- ElasticSearch propose un mécanisme d'Auto parse pour les dates et les valeurs numérique à partir d'une chaîne de caractères.

Mapping en ElasticSearch (Dynamique)

- ElasticSearch nous donne la possibilité de définir un template de Mapping dynamique.
- Le template de Mapping dynamique est défini à partir d'un type détecté automatique par ElasticSearch (La propriété `match_mapping_type`)
- Une correspondance ou non avec le nom du field (propriétés `match` et `unmatch`).
- Le type de donnée souhaité.
- On peut également utiliser des regex pour `match` ou `unmatch` avec le nom du field
- Exemple

Mapping en ElasticSearch (Explicite)

- Le mapping explicite nous délègue la tâche de définition de chaque type de field.
- Nous avons la possibilité d'ajouter un type de field par la suite à l'aide de l'API mapping.
- Chaque field peut accéder une multitude paramètre en fonction des conditions utilisations.
- Exemple.

TP 1

- En utilisant le fichier big_movies_elastic.json
- Ajouter les données movies avec mapping dynamique
- Ajouter les données movies avec mapping dynamique avec templates.
- Ajouter les données movies avec mapping statique.

Template Index

- ElasticSearch permet la création des templates Index
- Les templates index est un macénisme qui permet de définir un modèle qui sera appliquer à la création de l'index.
- Ce modèle peut contenir des mappings ou des settings
- Les templates index sont associés au Index à l'aide de l'index pattern.
- Exemple

Suite TP 1

- Création d'un template index pour toute index qui commence par movie

DSL ElasticSearch

- Elasticsearch DSL est une bibliothèque de haut niveau dont le but est d'aider à écrire et à exécuter des requêtes sur Elasticsearch.
- La DSL offre la possibilité d'écrire deux types de requêtes de recherches.
- Term-level
- Full-text

DSL ElasticSearch (Term-level)

- Le term-level permet une recherche exact d'une expression dans l'index inversé sans être analysée
- Exemple

DSL ElasticSearch (Full-text)

- Dans le cadre d'une recherche full-text, la requête est analysée et tokénisée en tableau.
- La recherche dans l'index inversé se fait pour chaque token en fonction de la pertinence.
- DSL offre une multitude de paramètres pour effectuer la recherche.
 - Match
 - Multi match
 - Match boolean prefix
 - Match phrase
 - Match phrase prefix
 - Common terms
 - Query string
 - Simple query string
 - Match all
 - Match none

DSL ElasticSearch

- DSL ElasticSearch offre d'autre façon d'effectuer des recherches:
- boolean
- Range
- Prefix
- Exists
- Wildcards
- Regex

DSL ElasticSearch (Template de recherches)

- DSL ElasticSearch offre la possibilité de créer de modèle de recherche.
- Les modèles de recherches peuvent avoir des paramètres pour les rendre plus dynamique.
- Un modèle de recherche possède une requête et des paramètres
- Exemple

Suite TP 1

- Réaliser des requêtes pour :
- Films “Star Wars” dont le réalisateur (directors) est “George Lucas”
- Films dans lesquels “Harrison Ford” a joué
- Films dans lesquels “Harrison Ford” a joué dont le résumé (plot) contient “Jones”
- Films de “James Cameron” dont le rang devrait être inférieur à 1000
- Films de “James Cameron” dont le rang **doit** être inférieur à 400
- Films de “J.J. Abrams” sortis (released) entre 2010 et 2015

ElasticSearch (Mécanisme d'agrégations)

- Les agrégations nous permettent d'exploiter le puissant moteur d'analyse d'Elasticsearch pour analyser vos données et en extraire des statistiques.
- Les cas d'utilisation des agrégations varient de l'analyse de données en temps réel pour prendre des mesures à l'utilisation de Kibana pour créer un tableau de bord de visualisation.
- Elasticsearch peut effectuer des agrégations sur des ensembles de données massifs en quelques millisecondes. Par rapport aux requêtes, les agrégations consomment plus de cycles CPU et de mémoire.
- L'agrégation n'est pas supporté sur des types text.
- Exemple d'agrégation simple

ElasticSearch (Mécanisme d'agrégations)

- Les agrégations nous permettent d'exploiter le puissant moteur d'analyse d'Elasticsearch pour analyser vos données et en extraire des statistiques.
- Les cas d'utilisation des agrégations varient de l'analyse de données en temps réel pour prendre des mesures à l'utilisation de Kibana pour créer un tableau de bord de visualisation.
- Elasticsearch peut effectuer des agrégations sur des ensembles de données massifs en quelques millisecondes. Par rapport aux requêtes, les agrégations consomment plus de cycles CPU et de mémoire.
- L'agrégation n'est pas supporté sur des types text.
- Exemple d'agrégation simple

ElasticSearch (Mécanisme d'agrégations)

- ElasticSearch propose 3 types de mécanisme d'agrégations:
- Metric aggregations : calculez des métriques telles que la somme, le min, le max et la moyenne sur des champs numériques.
- Bucket aggregations : Triez les résultats de la requête en groupes en fonction de certains critères.
- Pipeline aggregations : Canalisez la sortie d'une agrégation en tant qu'entrée vers une autre

ElasticSearch (Mécanisme d'agrégations)

- Une agrégation est la combinaison d'un *bucket* (au moins) et d'une *metric* (au moins).
- On peut, pour des requêtes complexes, imbriquer des *buckets* dans d'autres *buckets*.
- La syntaxe est très modulaire.

Suite TP

- Donner la note (rating) moyenne des films.
- Donner la note (rating) moyenne, et le rang moyen des films de George Lucas
- Donnez la note (rating) moyenne des films par année.
- Donner la note (rating) minimum, maximum et moyenne des films par année.
- Donner le rang (rank) moyen des films par année et trier par ordre décroissant.
- Compter le nombre de films par tranche de note (0-1.9, 2-3.9, 4-5.9...).
- Donner le nombre d'occurrences de chaque genre de film