Pourquoi elasticsearch?

Ainsi, vous construire un site web ou un système en ligne et que vous souhaitez ajouter de recherche pour le. Mais alors, il vous frappe: obtenir de recherche au travail est dur. Vous voulez la solution de recherche pour être rapide, indolore pour configurer et à l'échelle. Vous voulez être en mesure d'indexer les données en utilisant simplement JSON via HTTP sans avoir à pré-définir des schémas de l'indice.Vous voulez que le serveur de recherche de toujours être disponible et à créer de petites mais potentiellement à grande échelle - Données Big grande. Vous voulez créer des indices autant que bon vous semble, qui sera en charge un ensemble diversifié de types de documents. Comme vous recherche, vous voulez comme près de recherche en temps réel que possible. Ah oui ... et ce serait formidable si cette solution de recherche est construit pour le nuage.

*"Ce devrait être plus facile ... et cool, bonsaï cool!"*

**NOUS AVONS DÉCLARÉ**

*elasticsearch* a été créé comme une solution pour ces exigences et plus encore. Il s'agit d'un [Apache 2](http://www.apache.org/licenses/LICENSE-2.0.html)source sous licence libre, distribué et RESTful moteur de recherche, construit au sommet d' [Apache Lucene](http://lucene.apache.org/" \t "_blank) .

Schéma gratuit et orientation du document

Le modèle de données d'un moteur de recherche des racines aux bases de données de schéma de documents libres orientés, et comme le montre le mouvement # nosql, ce modèle s'avère très efficace pour construire des applications. *elasticsearch* modèle de l 'est JSON, qui émerge lentement comme de facto standard pour représenter les données de ces jours. Plus plus, avec JSON, il est simple de fournir des données semi-structurées avec des entités complexes, ainsi que langage de programmation étant naturelles avec des analyseurs de premier niveau.

Assignation de schéma

*elasticsearch* est du schéma moins, juste jeter un document dactylographié JSON et il sera automatiquement l'index. Les types tels que les numéros et dates sont automatiquement détectés et traités en conséquence. Cela dit, comme nous le savons tous, les moteurs de recherche sont assez sophistiqués. Les champs dans les documents peuvent avoir des niveaux d'appoint qui affectent la notation, les analyseurs peuvent être utilisés pour contrôler la façon dont le texte se tokenizés en termes, certains domaines ne doivent pas être analysés à tous, et ainsi de suite .... *elasticsearch* vous permet de contrôler complètement la façon dont un document JSON est mappé dans le moteur de recherche sur un type par et par niveau de l'indice. [Lire la suite ...](http://www.elasticsearch.org/guide/reference/mapping)

Apprendre quelques mots de données

L'indexation des données est toujours effectuée à l'aide d'un identifiant unique (au niveau du type). Ceci est très pratique car de nombreuses fois que nous souhaitons mettre à jour ou supprimer les données réelles indexées, ou juste l'obtenir. Obtenir des données ne pouvait pas être plus simple et tout ce qui est nécessaire est le nom de l'index, le type et l'identifiant. Ce que nous obtenons est le véritable document JSON utilisé pour indexer les données spécifiques, de manière efficace se comporter comme un distribués clé / valeur magasin pour les documents structurés. [Lire la suite ...](http://www.elasticsearch.org/guide/reference/api/get.html)

Rechercher

Quoi que tout se résume à à la fin, est d'être capable de chercher, et avec *elasticsearch* il ne pouvait pas être plus simple. Délivrance des requêtes d'un simple appel, se cachant le sophistiqué basé distribués soutien à la recherche *elasticsearch* fournit. La recherche peut être exécutée soit en utilisant une simple chaîne de requête fondée sur Lucene ou un vaste JSON basé sur les requêtes de recherche DSL.

Recherche ne se termine pas avec des requêtes seulement. Facettes, surlignage, scripts personnalisés, et bien plus encore sont tous là pour être utilisé en cas de besoin. [Lire la suite ...](http://www.elasticsearch.org/guide/reference/api/search)

Location multi

Un seul indice est déjà un grand pas en avant, mais ce qui arrive quand nous avons besoin de plus d'un index. Dans de nombreux cas, les indices multiples sont nécessaires. Un exemple peut être stocker un indice par semaine de fichiers journaux, ou même ayant des indices de différents avec des paramètres différents, dont une avec mémoire de stockage, et l'autre avec le système de stockage de fichiers).

*elasticsearch* permet facilement la création d'autant d'indices selon les besoins, permettant des requêtes croisées index doit être exécuté et le regroupement index en utilisant la fonctionnalité aliasing avancé.

Paramètres

La possibilité de configurer est une épée à double tranchant. Nous voulons la possibilité de commencer à travailler avec le système aussi vite que possible, sans aucune configuration, et être encore capable de contrôler presque tous les aspects de l'application si nécessaire.

*elasticsearch* est construit avec cette notion à l'esprit. Presque tout est configurable et peut être branché.De plus, chaque indice peut avoir ses propres paramètres qui peuvent l'emporter sur les paramètres de base. Par exemple, un indice peut être configuré avec mémoire de stockage et dispose de 10 éclats avec 1 réplique chacun, et un autre indice peut avoir de stockage de fichiers basé sur avec 1 tesson et 10 répliques. Tous les réglages de niveau d'index peut être contrôlé lors de la création d'un index, soit en utilisant un ou YAML format JSON.

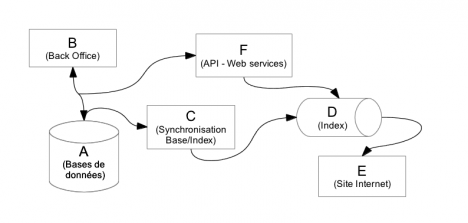
Distribué

Une des principales caractéristiques de *elasticsearch* est sa nature distribuée. Les indices sont décomposés en fragments, chaque tesson avec 0 ou plusieurs répliques. Chaque nœud de données au sein du cluster héberge un ou plusieurs éclats, et agit comme un coordinateur de déléguer les opérations à la bonne fragment (s). Le rééquilibrage et de routage sont fait automatiquement et dans les coulisses.[Voir dans l'action ...](http://www.youtube.com/watch?v=l4ReamjCxHo&feature=youtu.be)

Nuage

*elasticsearch* a été volontairement construit avec le nuage à l'esprit. À partir avec des options comme la détection automatique des nœuds lors de l'exécution dans AWS EC2, d'être un système adaptatif distribué qui gère automatiquement les machines qui vont et viennent, et s'adapte à un environnement dynamique.

Dans le cas de sites e-commerce par exemple, les accès au moteur de recherche ou à une fiche produit peuvent engendrer un grand nombre de requêtes SQL. Si les données sont modifiées régulièrement en "back office", il est difficile de mettre en place une stratégie de cache efficace et l’exploitation des plate-formes d’hébergement est fastidieuse. Pour régler ce problème, une solution désormais répandue consiste à exposer un index. Cet index stocke les informations nécessaires au "front" et autorise des temps d’accès réduits pour les clients du site.

Dans cet article, nous allons mettre en place la solution d’indexation [Elasticsearch](http://www.elasticsearch.org/). Il s’agit d’une solution open source en Java qui expose une API JSON au dessus d’[Apache Lucene](https://lucene.apache.org/).

Par rapport au modèle relationnel, le schéma d’Elasticsearch est ouvert, orienté document. Un "mapping" des données (lien entre la définition des tables SQL et la définition des données contenues dans l’index) permet de conserver la cohérence des données. De nouvelles fonctionnalités de recherche sont accessibles (facettes, optimisation, ...). Le paramétrage de la solution est simple et il sera facile de déployer de nouveaux "cluster" si les performances venaient à se dégrader.

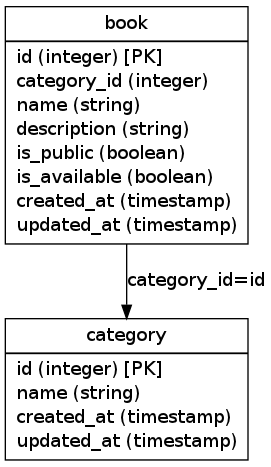
L’initialisation se déroule en trois étapes :

1. création de l’index ;
2. mapping des données ;
3. insertion des données.

Elasticsearch est une solution très riche. La [documentation](http://www.elasticsearch.org/guide) est assez bien faite. Même si le format court des chapitres peut être un peu déroutant au début, elle permet d’explorer les différentes fonctionnalités et de trouver des réponses à des questions sur une implémentation particulière (objets imbriqués, percolation, ...).

A première vue, les développeurs familiers des bases de données relationnelles seront peut-être déçus ne de pas disposer d’équivalent à l’instruction SQL "GROUP BY" pour manipuler les données de l’index Elasticsearch. [Ce problème a déjà été soulevé](https://github.com/elasticsearch/elasticsearch/issues/256) et peut être contourné facilement. L’implémentation de cette fonctionnalité est très attendue par les utilisateurs ...

Parfois la richesse de l’API d’Elasticsearch est un peu déroutante mais l’outil possède de nombreux avantages : open source, schéma de définition des données libre et ouvert (NoSQL), API JSON, paramétrage de base simple, scalabilité, ergonomie de recherche ... Un projet en devenir à utiliser sans attendre.



SF1 Action:

$this**->**form **=** **new** searchForm();

        $this**->**params **=** $request**->**getGetParameter('search', **array**());

        $q **=** **new** Elastica\_Query();

        $q**->**setSort(**array**('name' **=>** **array**('order' **=>** 'asc')));

        $facet **=** **new** Elastica\_Facet\_Terms('category');

        $facet**->**setField('category')**->**setSize(100)**->**setGlobal()**->**setOrder('term');

        $q**->**addFacet($facet);

        $client **=** **new** Elastica\_Client(**array**('host' **=>** 'localhost', 'port' **=>** 9200));

        $search **=** **new** Elastica\_Search($client);

        $search**->**addIndex($client**->**getIndex('books'));

        $q**->**setParam('size', 20);

        $q**->**setFields(**array**('name', 'description', 'category'));

        $this**->**books **=** **new** sfOutputEscaperSafe($search**->**search($q));

**<?php** use\_helper('search'); **?>**

<div class="span-6">

**<?php** search\_display\_facet($books**->**getFacets()); **?>**

</div>

<div class="span-18 last">

**<?php** **foreach** ($books **as** $book)**:** **?>**

**<?php** $data **=** $book**->**getData(); **?>**

<li>**<?php** **echo** sprintf('%s %s %s', $data['name'], $data['description'], $data['category']); **?>**</li>

**<?php** **endforeach**; **?>**

</div>