Google Elasticsearch

Overview

Elasticsearch is an open source search engine based on Apache Lucene (TM). Lucene is very complicated, and we need to have a deep understanding of the relevant knowledge of the search to understand how it works.

Elasticsearch also uses Java to develop and use Lucene as its core to implement all indexing and search functions, but its purpose is to hide the complexity of Lucene through a simple RESTful API, making full-text search simple.

However, Elasticsearch is more than just Lucene and full-text search, we can describe it like this:

Distributed real-time file storage, each field is indexed and searchable

- Distributed real-time analytics search engine
- Can scale to hundreds of servers to handle petabyte structured or unstructured data
- Moreover, all of these features are integrated into a single service, and your application can interact with it through a simple RESTful API, a client in various languages, or even a command line.

Important concepts

- **Index**: The logical area that Elasticsearch uses to store data, similar to the db concept in relational databases. An index can be on one or more shards, and a shard may have multiple replicas.
- **Document**: Entity data stored in Elasticsearch, similar to a row of data in a table in relational data. The document consists of multiple fields. The fields of the same name in different documents must have the same type. The field inside the document can be repeated, that is, a field will have multiple values, that is, multivalued.
- **Document type**: In order to query, an index may have multiple documents, that is, document type, but it should be noted that the fields of the same name in different documents must be of the same type.
- **Mapping**: Stores the mapping information of the field. Different document types have different mappings.

How I use it on Google cloud:

Install Elasticsearch

wget
https://download.elastic.co/elasticsearch/release/org/elasticsearch/distribution/deb/elasticsearch/2.3.1/elasticsearch-2.3.1.deb
sudo dpkg -i elasticsearch-2.3.1.deb
sudo systemctl enable elasticsearch.service

Edit the configuration file:

sudo vim /etc/elasticsearch/elasticsearch.yml

Start ElasticSearch:

sudo service elasticsearch start

Check the status of ElasticSearch:

sudo service elasticsearch status

RESTful API:

Elasticsearch provides the Restful API, which uses the JSON format, which makes it very easy to interact with the outside world. Although Elasticsearch has a lot of clients, I still easily write a simple client for the project, which confirms Elasticsearch again. It's easy to use your heart.
The Restful interface is very simple. A url represents a specific resource, such as /blog/article/1, which means that an index is blog, type is article and id is 1. And we use the http standard method to operate these resources, POST add, PUT update, GET get, DELETE delete, HEAD to determine whether it exists.

Compared with MySQL and NoSQL

Because MongoDB and ES are both a way of processing Nosql structure, so some comparisons are made here.

Distributed attribute

MongoDB supports clustering of replica sets, while ES also supports distributed architectures.MongoDB supports the JSON method, which is the same as ES. MongoDB also supports Grid mode for storing multimedia documents, which Es does not support.

Transactional

At this point MongoDB and ES are a bit similar, there is no support for strong transactions. Although the MongoDB manual has a two-stage approach to support the operation of multiple documents, it generally does not seem to be used. From the human cognitive system, it is always inclined to connect some unknown and unfamiliar things to the knowledge system that they already have, or to instantiate into a certain scene in real life. We will often see some 'helloworld' programs combined with examples from life to understand, and this is also the case. Here for us to better understand ElasticSearch such a full-text search tool

MySQL	Elastic Search
Database	Index
Table	Type
Row	Document
Column	Field
Schema	Mapping
Index	Everything is indexed
SQL	Query DSL
SELECT * FROM table …	GET http://…

Shard

Each Index (corresponding to Database) contains multiple shards. The default is 5, which is spread across different Nodes, but there will be no two identical shards on a Node, so there is no backup meaning. Shard is a minimal Lucene index unit.
When inserting the document, Elasticsearch hashes the docid to determine which shard it is placed on, and then stores it on the shard.

Shard stands for index fragmentation. es can divide a complete index into multiple fragments. This has the advantage of splitting a large index into multiple nodes and distributing them to different nodes. Form a distributed search. The number of shards can only be specified before the index is created, and cannot be changed after the index is created.

Replica

Replicas are backups. Elasticsearch uses the Push Replication mode. When you index a document onto the master main slice, the fragment will copy the document to all the remaining replica copies. These fragments will also be Index this document.

ES can set up multiple copies of the index. The purpose of the copy is to improve the fault tolerance of the system. When a certain piece of a node is damaged or lost, it can be recovered from the copy. The second is to improve the efficiency of es query, es will automatically load balance the search request.

Gateway

Represents the persistent storage mode of the es index. By default, es stores the index in memory first, and then persists to the hard disk when the memory is full. The index data is read from the gateway when the es cluster is closed and restarted. Es supports multiple types of gateways, including local file system (default), distributed file system, Hadoop HDFS and Amazon's s3 cloud storage service.

Reference:

1. https://en.wikipedia.org/wiki/Elasticsearch
2. http://www.slideshare.net/GauravKukal/elastic-search-indexing-internals
3. https://sematext.com/blog/2013/07/08/elasticsearch-refresh-interval-vs-indexing-performance/