Technology Review: Apache Solr vs Elasticsearch

Ruining Tao

With the development of the web and the emergence of more and more applications. The need for users as well as developers to search and analyze data is getting larger and larger. As a result, more and more search engines are emerging. Google is one of most famous search engines in the world. But beside google there are many other good search engines such as Apache Solr and Elasticsearch. This article is going to introduce these two different search engines and compare their difference.

Before comparing the two different search engines, it is necessary to understand the two search engines themselves. Apache Solr is an open-source search platform which written in java. It is a project organized under Apache software foundation. It has been widely used for enterprise search and analysis use cases. Since it is a well-known open-source project, many developers have formed a community to provide many useful updates to the project. Also, foundation maintains the project on a regular basis. Elasticsearch is also an open-source search engine built on top of Apache Lucene. It reinforced Lucene's features such as indexing and search capabilities using Restful APIs and using index and shards concept to distribute data across multiple servers for archiving. It has good performance on time service and NoSQL database. According to the DB-Engine ranking, Elasticsearch is the most famous enterprise search engine.

The next step is to compare the two search engines in many ways, such as features, indexing and search ability, performance on different platforms. The first thing is the configuration. Before you can use the Apache Solr, you are required to define the index structure, to define fields and their types in the managed-schema file (usually it called schema.xml). Elasticsearch is completely different with Apache Solr on schema definition. Because Elasticsearch is a free-schema search engine. In another words, Elasticsearch do not need to create any type of indexing schema to start sending documents to it to index documents. Elasticsearch will try to guess the field types as opposed to manually creating index mappings, which are not always 100% accurate.

For the search engine, the most important thing is how to get a correct result by the data entered by the user. Hence the way how they get the result is very important. Both of Apache Solr and Elasticsearch are expose their HTTP API. The way for Apache Solr is more traditional. To get search results from it you need to query one of the defined request handlers and pass in the parameters that define your query criteria. Depending on which query parser you choose to use, these parameters will be different, but the method is still the same – an HTTP GET request is sent to Solr to fetch search results. (Kuc, 2022). The HTTP GET in Apache Solr is also supported to return the result in xml, JSON in java bin format and a lot of different formats. The ability to select multiple outputs is very developer friendly. Elasticsearch can also return result from a REST API. Unlike Apache Solr, Elasticsearch's API can have multiple methods, such as HTTP GET, DELETE, POST and PUT methods. Users can use these methods to create indexes, manage them, control analytics, and get all the metrics that describe the current state and configuration of Elasticsearch.

Analysis Engine is another important difference in Apache Solr and Elasticsearch. In Apache Solr it has a function called JSON facets. It is a feature with fast and low memory demanding than regular data slice and dice. Another implementation is a stream-based expression called streaming expression. It can combine data from different source such as SQL, Solr and facets, then engine will decorate them by using various expression. Beside the function that already implemented in Apache Solr, Elasticsearch can also do nest data analysis but this function only support on the top of aggregation results. Elasticsearch has an experimental function that allow user to do matrix aggregation that user can compute the statistics over different fields.

Full text search is an important feature for a search engine. Since both of Apache Solr and Elasticsearch are built on top of Apache Lucene, thus both are leverage the Lucene's capabilities and they can query to match documents after documents have been indexed. But they still have some difference. Apache Solr has a lot of features on full text search. User can modify and do a lot of things on it. This also implied that Solr is more focus on full text search, and it has a good performance on it. Elasticsearch is more focus on filtering and grouping the data. Elasticsearch is more efficient on query it has low CPU usage and memory usage on querying data.

As result, both Apache Solr and Elasticsearch did a really good job on searching data. They have their own advantages and face to different users. For example, if you are aiming to development a search engine on text search, then Solr might fit you very well. If you are aiming to do a lot of data analysis and require a low response time, then Elasticsearch is the one most fit you.

Kuc, Rafal. "Solr vs Elasticsearch: Performance Differences & More [2022]." *Sematext*, Sematext, 19 Oct. 2022, https://sematext.com/blog/solr-vs-elasticsearch-differences/#1-apache-solr-vs-elasticsearch-engine-performance-scalability-benchmark.

Yigal, Asaf. "Solr vs. Elasticsearch: Who's the Leading Open Source Search Engine?" *Logz.io*, Logz.io, 1 Nov. 2021, https://logz.io/blog/solr-vs-elasticsearch/.