

RDBMS Analysis for Big Data using ElasticSearch

Taranjyot Singh¹, Geetika Bhatia²

²Assistant professor, Department of computer science

^{1,2}Guru Tegh Bahadur Institute of Technology

Abstract- In today’s era, it is inconceivable to use traditional techniques / RDBMS to analyse the data as it is growing very quickly. Big data offers the solution for analysing large amount of data. Using technique of Elasticsearch, access to data can be made faster. Elasticsearch is a search engine based on Lucene. It is near real time search platform. Elasticsearch uses the concept of indexing to make the search faster. This paper elaborates the search technique of Elasticsearch.

Keywords: Elasticsearch, Lucene

I. INTRODUCTION

RDBMS bolster benefits that have a composition. Unstructured or semi-organized information may should be recorded. It has turned out to be imperative to make another stage to satisfy the interest of association because of the difficulties looked by conventional information. Huge information breaks down developing volumes of organized exchange information, in addition to different types of information that are frequently left undiscovered by traditional business insight (BI) and examination programs [14]. Elasticsearch is a major information innovation which is pattern less. It is an apparatus used to look enormous information. Elasticsearch utilizes the idea of denormalization for hunt. Elasticsearch utilizes the ordering idea. It is an archive arranged device. When the report is included, it very well may be looked inside a one second from now as Elasticsearch is ongoing.

Elasticsearch is utilized for some, utilization cases like examination store, auto completer, spell checker, cautioning motor, and as a broadly useful archive store; Full content pursuit is one of it. It is a hearty web crawler that gives a speedy full content hunt over different archives. It seeks inside full content fields to discover the record and return the most significant outcome first. The importance of reports is great as Elasticsearch utilizes boolean model to discover archive. When a report coordinates an inquiry, Lucene ascertains its score for that question, joining the scores of each coordinating term. The significance of the archive can be determined utilizing viable scoring capacity.

Elasticsearch utilizes the idea of modified file for looking. At the point when the question is terminated, Elasticsearch investigates transformed record table to locate the required information. It will demonstrate the pertinent report where the term is contained. A modified list comprises of a rundown of all the novel words that show up in any report, and for each word, a rundown of the archives where it shows up. [5]. Rearranged list is the most normally utilized strategy for FTS (Full Content Pursuit). The modified file looks through the significant archive by mapping the term to its containing record. In the word reference the terms are arranged which results in brisk inquiry.

The following diagram depicts the working of Elasticsearch.

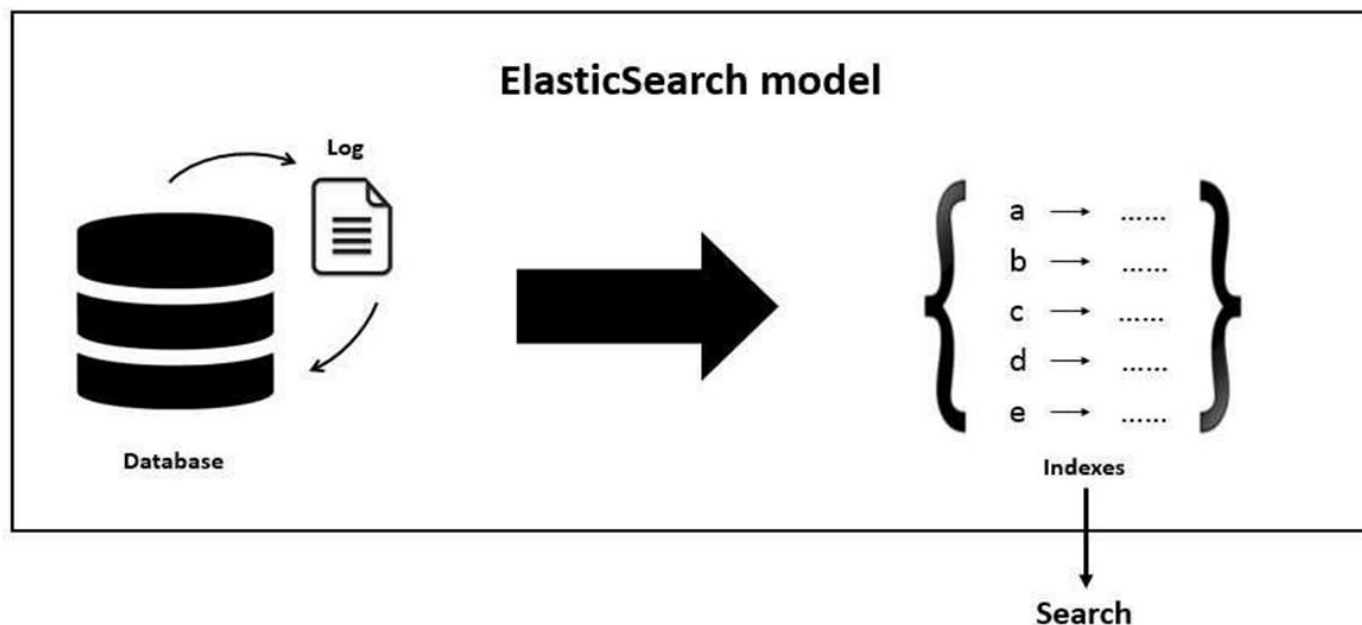


Fig.1: Elasticsearch Model [6]

Elasticsearch database is document oriented. By default, the full document is returned as part of all searches. This is referred to as the source. If the entire source document is not to be returned, then only a few fields from within source can be returned. The Elasticsearch uses the inverted index to search the term. The terms are sorted in ascending order.

II. WORKING OF ELASTICSEARCH

In Elasticsearch, a Report is the unit of inquiry and list. A record comprises of at least one Archives, and a Report comprises of at least one Fields [4]. Elasticsearch can accomplish quick inquiry reactions in light of the fact that,

rather than looking through the content straightforwardly, it looks through a list.

Elasticsearch gives the capacity to subdivide the file into different pieces called shards. At the point when another archive is put away and listed, Elasticsearch server characterizes the shard in charge of that record. At the point when a file is made, client can just characterize the quantity of shards. Every shard is in itself a completely utilitarian and free "list" that can be facilitated on any hub in the group. Any number of archives can be transferred regardless of its sort [2].

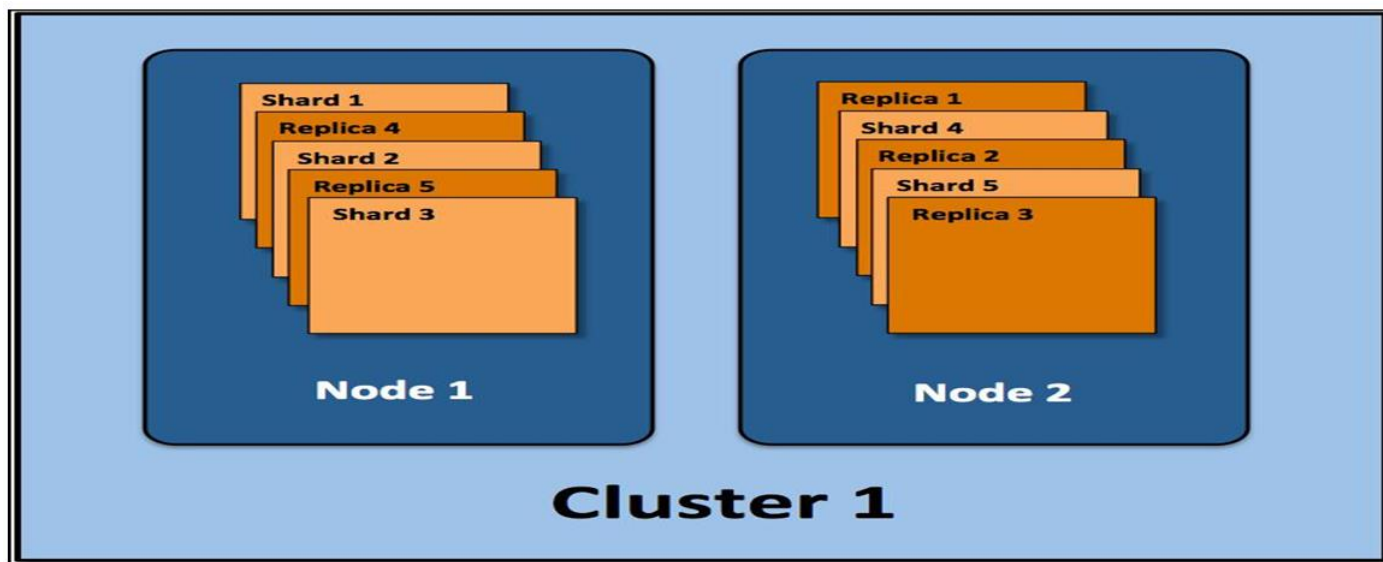


Figure 2: Elasticsearch Cluster [13]

An index is a collection of documents. An index is usually split into elements known as shards that are distributed across multiple nodes. Elasticsearch automatically manages the arrangement of these shards. It also rebalances the shards as necessary, so users need not worry about the details. By default, Elasticsearch creates five primary shards and one replica for each index. This means that each index will consist of five primary shards, and each shard will have one copy. Elasticsearch automatically arranges the five primary shards split across the two nodes. [13]

Elasticsearch indexes all fields of document by default so the field becomes searchable. An inverted index consists of a list of all the unique words that appear in any document, and for each word, a list of the documents in which it appears. Document consists of fields, and each field identified by its name and can contain one or multiple values. Each document may have different set of fields; there is no schema or imposed structure. [1]

3. Explanation of Elasticsearch working through examples

- (i) Creation of inverted index using Elasticsearch
- (ii) Working of Elasticsearch with stopwords list concept

(i) Creation of inverted index using Elasticsearch

If the user wants to find articles that talk about aircraft, you may search using the word "aircraft". If you don't have a special indexing technique, then all the records will be scanned to find a match, which is inefficient technique. "inverted index" offers the solution for this.

The following example elaborates the searching using Elasticsearch. Consider these documents to be indexed with the contents in it.

Document1: "Elasticsearch is fast"

Document2: "Elasticsearch is efficient"

Document3: "Elasticsearch offers high speed"

Now to create the inverted index for the above documents, the contents of each document is split into separate words. After then, the lists of unique words, the document ids in which they are contained and the word frequency list are generated. So the inverted index generated for the above documents would be below:

Terms	Document	Position	Term Frequency
Elasticsearch	Document1,Document2 , Document 3	1,1,1	3
is	Document1,Document2	2,2	2
fast	Document1	3	1
efficient	Document2	3	1
offers	Document3	3	1
High	Document3	4	1
speed	Document3	5	1

Fig.3: Creation of Inverted Index using Elasticsearch

If the query is fired to search the word “offers”, then Elasticsearch will look into its inverted index table. It will find that the word occurs in document 3 and will show that document. So due to the inverted index creation, the search becomes faster.

(ii) Working of Elasticsearch with Stopword list concept
The following diagram elaborates the working of Elasticsearch. It creates an inverted index internally.

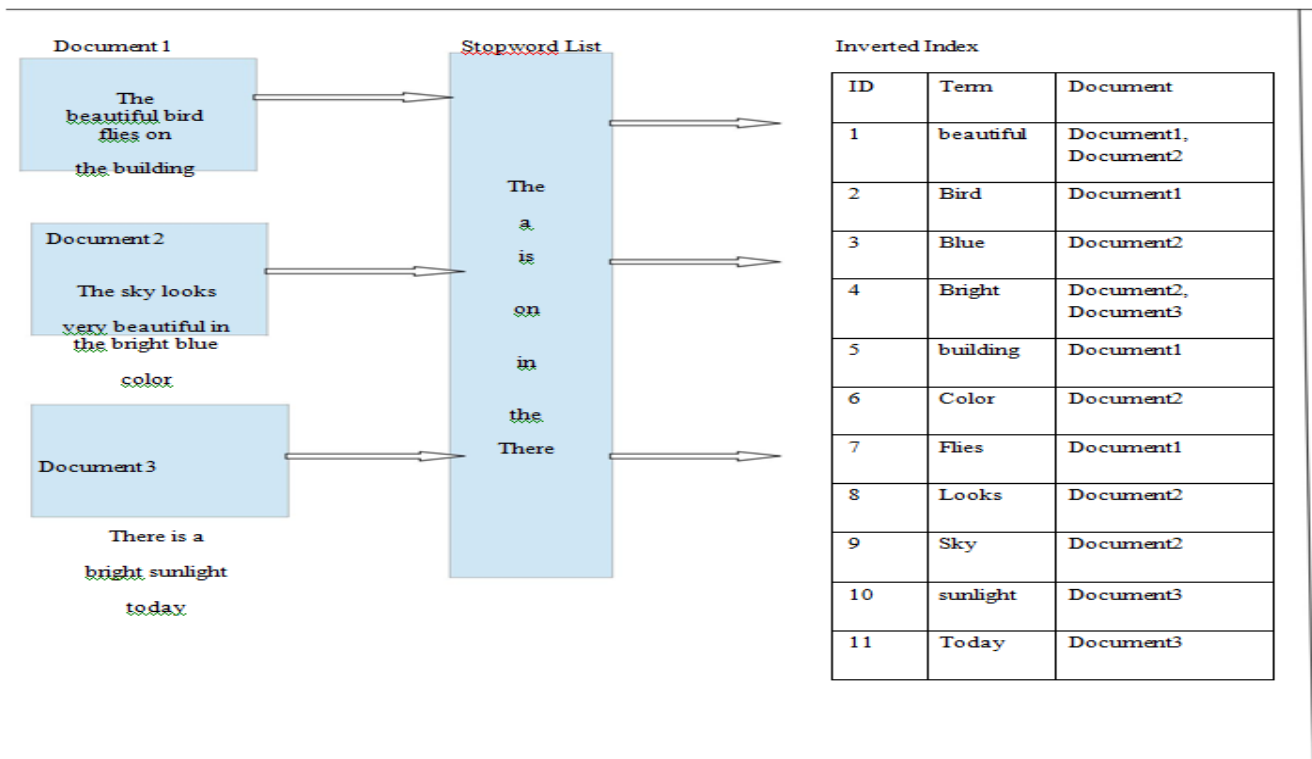


Fig.4: Working of Elasticsearch with Stopword List concept

Some words Seldom add value to a search. As per the above figure, the Elasticsearch uses the concept of Stopword list to fasten the search process. Elasticsearch has its own list of predefined stopwords. Stopwords can usually be filtered out

before indexing. The inverted index is then created over the terms of document to make search faster.

III. FUTURE SCOPE

The stopword list can be improved to fasten the search. The process of tokenization and normalization which is called analysis, can be improved to fasten the search process. The practical scoring function can also be improved for better relevancy of documents.

IV. CONCLUSION

Search is a core part of Elasticsearch. If the analysis process is defined on the index, the retrieval process would be more efficient. Elasticsearch is a good technology of big data for searching operation. It is near real time. Elasticsearch uses inverted index for search. Elasticsearch can search full text fields and most relevant result is returned. The search becomes quicker.

V. REFERENCES

- [1]. Survey Paper on Elastic Search Pragya Gupta, Sreeja Nair International Journal of Science and Research (IJSR)
- [2]. R. Vidhya and G. Vadivu Indian Journal of Science and Technology, Vol 9(37), DOI: 10.17485/ijst/2016/v9i37/102108, September 2016
- [3]. Urvi Thacker ; Manjusha Pandey ; Siddharth S. Rautaray Electrical, Electronics, and Optimization Techniques (ICEEOT), International Conference
- [4]. <http://www.elasticsearchtutorial.com/basic-elasticsearch-concepts.html>
- [5]. <https://www.elastic.co/guide/en/elasticsearch/guide/current/inverted-index.html>
- [6]. https://www.google.co.in/search?q=elasticsearch+architecture+diagram&source=lnms&tbn=isch&sa=X&ved=0ahUKEwiosDhnLTAhXML48KHe16DRgQ_AUIBigB&biw=821&bih=572#tbm=isch&q=elasticsearch+diagram&imgcr=n0
- [7]. PCYMi6gSQVvM[7]https://www.tutorialspoint.com/elasticsearch/elasticsearch_basic_concepts.html
- [8]. Use of Elastic Search for Intelligent Algorithms to Ease the Healthcare Industry C. Bhadane, H. A. Mody, D. U. Shah, P. R. Sheth
- [9]. Full-Text Search on Data with Access Control Ahmad Zaky, Rinaldi Munir, S.T., M.T. School of Electrical Engineering and Informatics Institut Teknologi Bandung
- [10].Elasticsearch: How to Add Full-Text Search to Your Database:<https://medium.com/@MentorMate/elasticsearch-how-to-add-full-text-search-to-your-database-ee2f3ea4d3f3>
- [11].Use Cases for Elasticsearch: Full Text Search:<http://blog.florian-hopf.de/2014/07/use-cases-for-elasticsearch-full-text.html>
- [12].<https://www.elastic.co/guide/en/elasticsearch/guide/current/stopwords.html#stopwords>
- [13].<https://qbox.io/blog/optimizing-elasticsearch-how-many-shards-per-index>
- [14]. <https://www.projectguru.in/publications/difference-traditional-data-big-data/>