International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 6 | ISSN : 2456-3307

Survey on Elasticsearch

Umesh Taware*, Nuzhat Shaikh

Department of Computer Engineering,Savitribai Phule University, Pune, Maharashtra, India

ABSTRACT

Elasticsearch is distributed, real-time, scalable text based search engine.It is product Apache software foundation. It enables you to search, analyze and explore the data stored in it .It is used for full-text search, analytics, structured search. Elasticsearch gives way to easily access your data and gives performance enhancement over traditional SQL database when querying a data.

Keywords: Elasticsearch, NoSQL Database, Data Querying, Search Engine, Full-text Search

I. INTRODUCTION

Elasticsearch is full-text search-engine library. It is built on Apache Lucene .It is a open-source library.It present data in JSON like form also called BSON data format.

One of the problem with Lucene was you needed to integrate it with java to make it work with your application.It does not expose its services directly to it's client.

Elasticsearch does uses Lucene at the back end for indexing and storing its data at the same time it is also written in Java, Elasticsearch makes it easier to access the stored data by expsoing it by the means of RESTful web services.

This paper aims at exploring various aspects of Elasticsearch and take a brief look towards its functionality.

II. BASIC CONCEPTS

Elasticsearch gives more than just being full-text based search engine . It can be seen as combination of following:

- It offers distributed search enviroment in which data stored is searchable at the delay of 1 second.
- Elasticsearch gives real-time analytics for the data stored.
- One of the most exciting feature of Elasticsearch is its ability to upscale.It has got ability to expand as and when need without affecting existing architecture.It gives access to its data using RESTful webservices and also URL based querying for data can also be done.

Following are the basic components of Elasticsearch:

A. Index

An index can be seen as 'database' in a relational database.Mapping in Index defines multiple types present in it. An index is a logical namespace which maps to one or more primary shards .An Index can have zero or more replica shards.This replication can be increased in order to increase availability of data.

B. Document

Document is top level element in Elaticsearch which is serialized into JSON object which is mapped to unique id.Fields name can be any valid string but periods in the field name are not allowed.

C. Type and Mappings

Every document in Elasticsearch has its own Type associated with it. Where as Mapping acts as database schema which desribes properties that document of particular type might hold.

Types can be useful abstractions for partitioning similar-but-not-identical data. But due to how Lucene operates they come with some restrictions.

D. Node

A node acts as single entity in Elasticsearch.There can be multiple nodes on in single cluster.Each node is given default name which is given at the time of start up by default called as Universally Unique Identifier(UUID).

A node can be specifically assigned to or joined to cluster by providing a cluster name ,if it is not specified then it will be joined to default cluster name elasticsearch.

E. Cluster

A cluster contains one or more nodes.A cluster holdes large amount of data by having it distributed over multiple nodes enclosed in it.

There might be one or more clusters present but each cluster must have unique id to it.By default each cluster is given a name at creation which can be changed as per the user preference .

F. Shards and Replicas

Elasticsearch spreads data in clusters over several physical Lucene disks called shards.Index is divided into shards.Elasticsearch automatically divides data into shards and binds in the form of Index.

III. WORKING OF ELASTICSEARCH

Following section explains key working aspects of Elasticsearch.

A. Start-up process

When ElasticSearch node starts, it uses the discovery module to find the other nodes on the same cluster and connect to them. By default the multicast request is broadcast to the network to find the other ElasticSearch nodes with same cluster name.In the preceding figure, the cluster, one of the nodes that is master eligible is elected as master node. This node is responsible for the managing the cluster state and the process of assigning shards to node in reaction to changes in cluster topology.

The master node reads the cluster state and, if necessary, goes into the recovery process. During this state, it checks which shards are available and decides which shards will be the primary shards. After this the whole cluster enters into a yellow state.

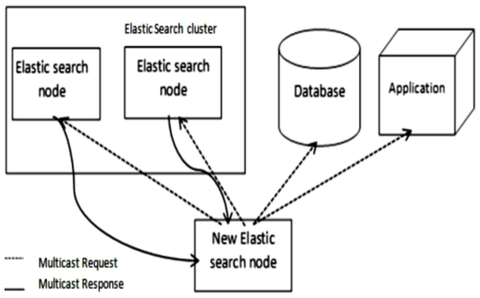

Figure 1. Start up Process

B. *Querying Data*

The Query API is a big part of Elastic Search API.The Query process can be divided into two phase: the scatter phase and the gather phase. The scatter phase is about querying all the relevant shards of the index. The gather phase is about gathering the results from the relevant shards, combining them, sorting, processing and returning to the client.

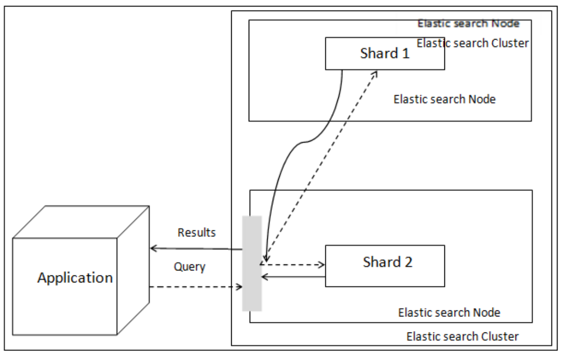

Figure 2. Quering Data

C. *Indexing Data*

There are few ways to send data to Elasticsearch. The easiest way is using the index API, which allows sending a single document to particular index. The second way allows sending many documents using the bulk API and the UDP.
The difference between these two methods is the connection type. Common bulk command sends documents by HTTP protocol and UDP bulk sends this using connection less datagram protocol. This is faster but not so reliable.

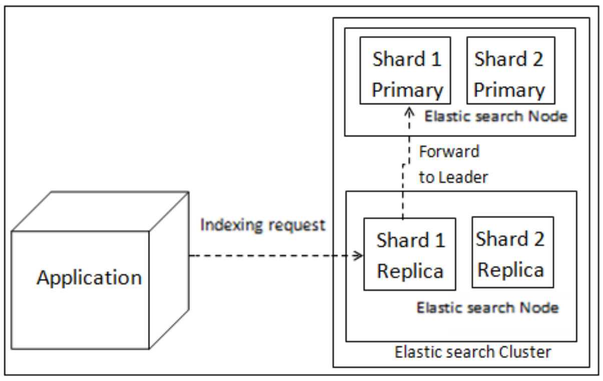

Figure 3. Indexing in Elasticsearch

IV. SQL VS ELASTICSEARCH

There are significant differences in traditional databases and ElasticSerach based database engine.These database engines use different approach in saving and querying the data and also vary in the way they are implemented. Following table does basic comparison of features between Elasticsearch and Micorsof SQL:

TABLE I
SQL VS TRADITIONAL DATABASE

Name	**Elasticsearch**	**Microsoft SQL**
Database Model	Search Engine	Relational DBMS
DB Engine Ranking	Score 119.78	Score 1172.48
Cloud Based	no	no
Impl. Language	Java	C++
Data Scheme	Schema Free	Yes
XML Support	No	Yes
SQL Support	No	Yes
In memory Capabilities	No	Yes
Transaction Concepts	No	ACID

Following graph shows DB-Engines rating for Elasticsearch vs Microsoft SQL vs Solr DB engines.

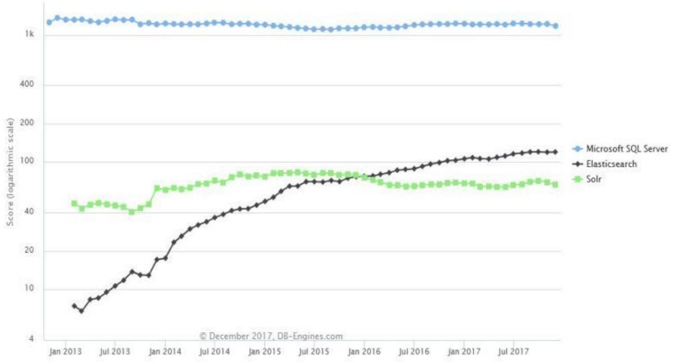

Figure 4. DB-Engine Ranking for ElasticSearch vs Microsoft SQL vs Solr

V. CONCLUSION

Elasticsearch due to high scalabilty and feature rich APIs can be used as underlying technology/engine that powers applications that have complex search features and requirements. It can be used to either build sophisticated search applications or to mine intelligence from your data.

VI.REFERENCES

[1] Mayuri Sadaphule and Nuzhat Shaikh, “An Application of Database Query Translation into Spreadsheets” IEEE International Conference on Electrical, Electronics and Optimization Techniques , 2016, 2nd-4th March 2016.

[2] Mayuri Sadaphule and Nuzhat F. Shaikh, “A Survey: A Tool for Database Query Translation into Spreadsheets” International Journal of Engineering Science and Technology (IJEST), Vol. 7 No.11, pp 401-406 ", ISSN : 0975-5462, Nov 2015.

[3] DB-Engines - https://db-engines.com

[4] Elasticsearch - https://www.elastic.com

[5] Dzone - https://www.dzone.com

[6] **Olga Baysal and Reid Holmes, “Mining modern repositories with elasticsearch”,** 11th Working Conference on Mining Software Repositories(2014), ISBN: 978-1-4503-2863-0,May 2014.

[7] Yeu yu and Tao Wang ,”Detecting Duplicate Pull-requests in GitHub”,9th Asia-Pacific Symposium on Internetware,ISBN: 978-1-4503-5313-7,September 23 - 23, 2017.

[8] Stephen Mock,”Design safe:Using Elasticsearch to share and Search Data on Science Web Portal”,Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact,ISBN: 978-1-4503-5272-7,July 13,2017.

[9] Joaquin Delgado,”**Scalable Recommender Systems: Where Machine Learning Meets Search**”,9th ACM Conference on Recommender Systems,ISBN: 978-1-4503-3692-5,September 20 ,2015.

[10] Anton Firsov ,”Traditional IR meets Ontology Enginnering in search for Data”,40th International ACM SIGIR Conference on Research and Development in Information Retrieval,ISBN: 978-1-4503-5022-8,August 2017.

[11] Andreas Both, ”**A service-oriented search framework for full text, geospatial and semantic search”,**10th International Conference on Semantic Systems,ISBN: 978-1-4503-2927-9,September 2014.

[12] Shelly Garion,”Big Data Analysis of cloud storage logs using spark”, 10th ACM International Systems and Storage Conference,ISBN: 978-1-4503-5035-8,May 2017.

[13] Bela Gipp, ”**Citolytics: A Link-based Recommender System for Wikipedia”,11th** ACM Conference on Recommender Systems,ISBN: 978-1-4503-4652-8,August 2017.