A Survey Paper On The Comparison Of NOSQL Engines (Mongo dB vs. Cassandra) Using Spark

Mounica.B[1], Kokila.N [2],Butti Pavithra [3], Tejaswini.V [4]
[1]Senior.Asst. Professor, Department of Information Science Engineering, New Horizon College of Engineering
[2,3,4]Student, Dept. Of ISE, New Horizon College of Engineering, Bangalore, India

***Abstract*-**Relational Database has been used for data storage, recovery and management, as there is need for scalability and performance, another system like NoSQL technologies are integrated. While most of the researches are focused on the performance and there has been no focus on the classification for NoSQL databases where databases are compared with each other .To overcome this, we examine and create a brief and up-to-date evaluation of NoSQL engines (Mongo dB vs. Cassandra), recognizing their most valuable use cases and the quality attributes that each of them is mostly suited to.

Keyword-Big data, Mongo dB, Cassandra, Spark, NoSQL.

1. Introduction.

Big data is a term that describes the large volume of data for both structured and unstructured, but it's not concerned about the amount of data, it's about the organizations that do with the data matters [1]. Big data can be analyzed for understandings that lead to better decisions and strategic moves.

Definition of Big data as four V's:

Volume: Organizations contains a collection of data from many sources, such as business transactions, social media and information from sensor or machines.

Velocity: Data streams at high speed and must be distributed within timely manner, for e.g.: the social media messages and credit card transaction details.

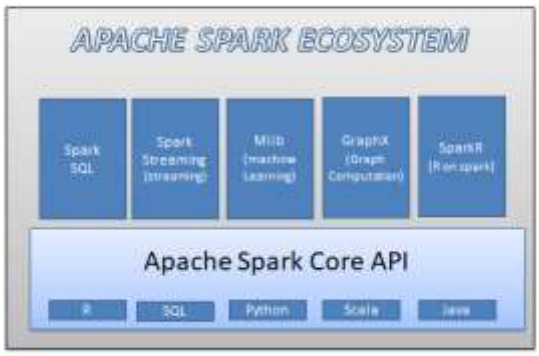
Variety: Data sources are extremely heterogeneous. Data is obtained in all types of formats like structured and unstructured such as documents, email, video, audio and financial transactions. [1, 2]

Hadoop is the foundation for big data applications, and it is the basic platform for big data. Spark is a framework which exploits in-memory abilities to deliver fast (almost 100 times faster than Hadoop). Thus, Spark is mostly used in the world of big data, and mainly for fast processing. Spark is an open source framework that processes big data with speed and simplicity. Spark can be used in Hadoop environment. It is developed at the University of California and then later offered to the Foundation. The purpose of Spark is, it offers the developers with an application framework that works at a centered data structure. Spark is very powerful and has the initial ability to process huge amount of data in a short period of time, which provides excellent performance. Thus, Spark makes it a lot faster than compared to Hadoop [4].

SPARK ECOSYSTEM:

There are 6 components in Apache Spark Ecosystem which are:

- Spark Core
- Spark SQL,
- Spark Streaming,
- Spark MLlib,
- Spark GraphX,
- SparkR.



SPARK CORE:

The functionalities of Apache Spark are built on the top of **Spark Core**. It delivers speedily with the help of Spark. It is responsible for basic I/O functionalities, scheduling and monitoring the jobs on spark clusters. It help in fault recovery, it overcome the problem of Map Reduce by using in-memory computation.

Spark Core is embedded with RDD (Resilient distributed dataset). While when we reuse data in computing systems like Hadoop Map Reduce. It needs to store the data in an intermediate storage. Thus, the speed of computation reduces. Thus, RDD overcomes the limitation with the help of fault tolerant [6].

Spark SQL:

Spark SQL is one of the Spark Ecosystem components. Spark SQL is used when the data is large, to perform structured data analysis. By using the component, we get information about the data structure and their computations. With the help of this information we can perform extra optimization, where an output can be computed on the engine. The extraction and merging of dataset can be performed easily with the help of Spark SQL. So that. It acts as a distributed SQL Query. It can also access structure and semi structure.

Spark Streaming:

It is a light weighted component of Spark Ecosystem. With the help of it developers can perform batch processing and streaming. The real time data can be processed, by using a continuous stream of input data's.

MLIB:

It is one of the most important components of Spark Ecosystem. It's a scalable **machine learning library**; it obtains both High-quality algorithms as well as intense Speed. It supports API's like Java, Scala, and Python [6].

GraphX:

GraphX is used in spark as graphs and graphical computations; Spark has "Graph Computation Engine", called GraphX. It is widely used as graph processing tools.

SparkR:

R is the best for statistical performance. R has already been integrated in Hadoop. It is a package for R language that enables R users to influence the power of Spark from R.

WHY Spark:

Apache Spark has been recognized to overcome Hadoop in various features, which possibly explains why spark is important. One of the main reasons for this is its processing speed. It also offers faster processing for about 100 times than Hadoop for the same amount of data. It also uses expressively few properties as Hadoop, by making it cost-effective. Spark also has the upper hand in terms of compatibility with resource management. It is also known to run with Hadoop, just like Map Reduce. Apache Spark can also work with resource manager like YARN or Mesos.

It has APIs of several languages such as Scala, Java and Python. It is simple to write the user-defined functions.

NOSQL:

NOSQL systems are Non-Relational database systems that are distributed and are understood as Not Only SQL.

NOSQL databases are known to provide easier scalability, storage flexibility, and greater data manipulation and performance improvement.

The types of NOSQL database systems are: Wide-column stores, Graph databases and Document stores are identified most commonly

1. Mongo DB,

2. Cassandra,
3. Dynamo DB and
4. Couch,
5. Neo4j,
6. Riak

are the more popular NOSQL databases used commonly in today's environment. From the above types of NoSQL databases we have chosen Mongo DB and Cassandra [5,8] to check the performance between them both.

Mongo dB:

MongoDB is an open source database which uses document-oriented data model. Instead of using tables and rows like relational database, Mongo DB is developed on the "architecture of collections" and "documents". Here it also holds a set of documents and functions as similar as relational database tables. It also supports "dynamic schema design", It can have a collections of different fields and structures. BSON are used for Document storage and data swapping format, the binary representation of JSON- as documents. "Automatic sharing" the data in a collection to be distributed across multiple systems for horizontal scalabilities data sizes increase.

While inserting the data into mongodb, it is faster, than Cassandra ,but while retrieving the data it take a lot of time ,as the data is been dumped into the mongo dB as a document format[3,5,9].

The characteristics are:

1. It provides Schema-free.
2. It supports for high performance.
3. Replication and fail sat high availability
4. Auto Shading
5. Easy readability
6. Master-slave model
7. CP on CAP

Good for:

1. RDBMS replacement for web applications
2. Semi-structured content management
3. Real-time analytics
4. High-speed logging, caching and high scalability
5. Web 2.0, Media, SAAS, Gaming

Not good for:

1. High transactional systems
2. Applications with traditional database requirements such as foreign key constraints

Usage Case: Craigslist, Foursquare

Cassandra:

Cassandra is also known as a NoSQL database it is extremely scalable and big data prepared. it is also a distributed database i.e. highly fault tolerant without any single point of failure. It is also known as high performance database. Cassandra was basically developed at Face book as a mixture of the Cassandra is mainly used to store a large amount of data very quickly. As many online applications have database requirements that overdo the skills of a relational databases. It is a huge "Open source non-relational database" .which offers a continuous availability and easy data across multiple data centers and cloud, operational simplicity.

Wide-column store based on ideas of Big Table and Dynamo DB

Cassandra has a single-row read performance as long as eventual consistency semantics are enough for the use-case.

If data is stored as columns in Cassandra supports range scans. Cassandra supports secondary indexes on columns where the column name is known.

The Aggregations must be provided by the clients, while they are not accepted by the Cassandra nodes. When the aggregation extends multiple rows, Random Partitioning is very difficult for the client. Thus Storm or Hadoop are recommended for aggregations [3, 5, 9].

Characteristics:

1. High availability

2. Incremental scalability
3. Eventually consistent
4. Trade-offs between consistency and latency
5. Minimal administration
6. No SPF (Single point of failure) – all nodes are the same in Cassandra
7. AP on CAP

Good for:

1. Simple setup, maintenance code
2. Fast random read/write
3. Flexible parsing/wide column requirement
4. No multiple secondary index needed

Not good for:

1. Secondary index
2. Relational data
3. Transactional operations (Rollback, Commit)
4. Primary & Financial record
5. Stringent and authorization needed on data
6. Dynamic queries/searching on column data
7. Low latency

Usage Case: Twitter, Travel portal

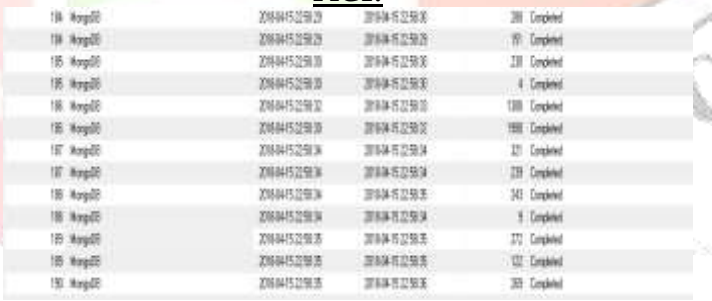
Experimental Framework:

Our comparisons between the two databases include:

1. Insertion of data into the database.
2. Retrieval of the data from the database.
3. Deletion of the data.

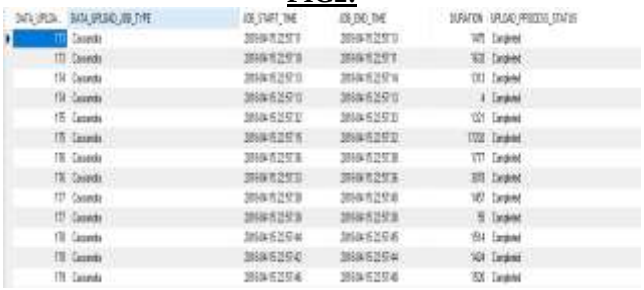
The performance of both the NoSQL Databases can be compared with the help of the duration taken by them to perform the insert and retrieve.

FIG1:



(fig1)The above image shows us the duration of time taken by Mongo dB to insert and retrieve the data.

FIG2:



(fig 2)The above image shows us the duration of time taken by the Cassandra for insert and retrieves of the data. The performance of the retrieval and insert id represented in the graphical form.

Insert to the database:

The above graph is plotted based on insertion of data in MongoDB and Cassandra.

Retrieval of data:

The above graph is plotted based on the retrieval of data in MongoDB and Cassandra.

CONCLUSION:

NoSQL consist of multiple types of databases but for this paper we have chosen MongoDB and Cassandra. Here we check the performance between Mongodb and Cassandra ,as the performance between them is been obtain through a graphical representation, With the help of this paper ,it proves that Cassandra works faster than Mongo DB in clustered memory and it also helps in fast retrieval and insertions of multiple data. Even Mongo DB is fast in retrieval and insertion in single memory. Thus ,this paper make us understand that both the NoSql databases are best in their performance in their own type, as they both have their own advantages as well as disadvantages.

References:

1. Dhole Poonam B, GunjalBaisa L, “Survey Paper on Traditional Hadoop and Pipelined Map Reduce” International Journal of Computational Engineering Research||Vol, 03||Issue, 12||
2. Labrinidis A, Jagadish H. Challenges and opportunities with big data. Proceedings of the VLDB Endowment, 2012, 5(12): 2032–2033Google Scholar
3. A performance comparison of SQL and NoSQL databasesYishan Li and SathiamoorthyManoharan Department of Computer Science University of Auckland New Zealand.
4. B. Tudorica and C. Bucur, “A comparison between several NoSQLdatabases with comments and notes,” in Roedunet International Conference (RoEduNet), 2011 10th, june 2011, pp. 1 –5.M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, I. Stoica, "Spark: cluster computing with working sets", Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, 2010.
5. M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, I. Stoica, "Spark: cluster computing with working sets", Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, 2010B.
6. http://jorditorres.org/spark-ecosystem/
7. https://databricks.com/spark/about
8. L. Bonnet, A. Laurent, M. Sala, B. Laurent, N. Sicard, "Reduce you say: What nosql can do for data aggregation and bi in large repositories", Database and Expert Systems Applications (DEXA) 2011 22nd International Workshop on, pp. 483-488.
9. https://www.linkedin.com/pulse/real-comparison-nosql-databases-hbase-cassandra-mongodb-sahu
10. Big Data Anonymization with Spark Yavuz Department of Computer Engineering Faculty of Engineering, Gazi University Ankara, Turkey Seref Department of Computer Engineering Faculty of Engineering, Gazi University Ankara, Turkey
11. S. Sakr, A. Liu, D. Batista, and M. Alomari, “A survey of large scale
    Data management approaches in cloud environments,” Communications Surveys Tutorials, IEEE, vol. 13, no. 3, pp. 311–336, 2011.
12. Handling Big Data Using NoSQLJagdevBhogal Fac. of Comput., Eng. & the Built Environ. Birmingham City Univ., Birmingham, UK.ImranChoksi, Fac. of Comput., Eng. & the Built Environ, Birmingham City Univ., Birmingham, UK.
13. Clustering of Datasets Using K-Means Algorithm in SPARK, International Journal of Innovative Research in Computer and Communication Engineering, Subiksha N 1, Pallavi R Reddy1, Mounica B2 Vol. 5.
14. Inclusive Analysis of Incomplete Datasets Using IkNN Search International Journal of Innovative Research in Computer and Communication Engineering Vol.5, Baswaraju Swathi1 , Sonia Singh2 , Sujithra K S 3