National College of Ireland

Data Storage and Management Project

On

Performance Analysis of MongoDB and HBase

By:

Ranu Parate

X17161452

MSc in Data Analytics – 2018/19

Table of Contents

1. Abstract:

In this report the comparison of two NoSQL databases namely; MongoDB and HBase are performed. This comparison is made on the basis of performance of databases at its security level. Also, the key features and data base architecture of MongoDB and HBase is discussed in this report. Detailed explanation of database architecture of both the database is shown. Using YCSB benchmarking tool the performance evaluation and results of read and write operations for MongoDB and HBase are analyzed and compared considering average latency in workloads A and F. The description of the machine is also discussed on which the test is performed. That is the configuration of the physical machine and virtual machine is discussed. The overall throughput is analyzed for MongoDB and HBase in workload A and F. In this report literature review is done on YCSB benchmark tool.

2. Introduction:

Fast growing of technology has transformed the way of reading, writing, gathering information, implementation of different projects with huge amount of data and many more. Here comes the term BIG DATA. "Big data is a term used to refer to data sets that are too large or complex for traditional data-processing application software to adequately deal with"(Wikipedia 2018). It is basically enormous data. The transformation due to rapid growth of technology originates through a common result of massive amount of data; this data is around 2 to 3 Exabyte, which means this much of data gets produce per day across the world. This produced data need to store as every single data has its own importance. Storing this data and handling this massive data play an important role. Dealing the huge data with the normal methods is not efficient. It may affect the availability, scalability, reliability and security of the data. To successfully deal with the factors which are important for the massive data there are many storage methods available like MongoDB, HBase, Cassandra, MySQL. Also the HDFS, YCSB are available to handle the big amount of data. In this report the key characteristics, database architecture of MongoDB and HBase are discussed. Also, the performance at different workloads using YCSB is analyzed and compared.

3. MongoDB:

"MongoDB is an open-source, document database designed for ease of development and scaling" (Docs.mongodb.com 2008). It is built by MongoDB Inc. It is a document oriented database which provides high availability, performance and scalability. MongoDB works on concept of collection and document where collection is a set of MongoDB documents and document consists of key-value pairs. The set of these pairs are present in the document.

3.1 Features of MongoDB:

According to the documentation of MongoDB (2008) Key features of MongoDB are:

1. High Availability: The servers of MongoDB maintain the replicas of data set, to provide the redundancy due to which it increases the high availability of data.
2. High Performance: MongoDB delivers high performance of data perseverance. Due to high performance the I/O activities performed on database system get reduced. Also, the faster queries get supported by indexes.

3. Horizontal Scalability: The horizontal scalability is considered as a part of its core functionality. In this the data is distributed across the machine cluster using Sharding.
4. Rich Query Language: The rich query language is supported by MongoDB for supporting read-write operations and Data Aggregation functions.
5. Multiple Storage Engines: MongoDB supports for Multiple Storage Engines: they are WiredTiger Storage Engine, In-Memory Storage Engine. Additionally, the API of pluggable storage engine is also provided by MongoDB which allows the development of storage engines required for MongoDB.

4. Hbase:

Wikipedia (2018), HBase is an open-source, non-relational, horizontally scalable, column-oriented database built on top of the Hadoop file system which is molded to Google's Bigtable. It offers arbitrary real-time read/write entree to data in the Hadoop File System (HDFS). HBase gives quick lookups to the tables having large amount of data.

4.1 Features of HBase:

Kumar and Shindgikar (2018, p. 136) describes the key Characteristics of HBase as:

1. Control data sharding: In HBase table data is strongly impacted by data sharding. Table row-key, column-key and column families are used for sorting the data present in the table in ascending order. To ensure whether the data is equally circulated across the Hadoop cluster a solid row-key is used.
2. Sorted rowkeys: Three basic operations are used to process the data in HBase. These three operations are get, put and scan. For accessing the data all the three operations are used by making use of rowkeys. A scan is defined here using sorted rowkeys. By this all the data which are required can be achieved by calling the database one time.
3. Flexibility: HBase is good in flexibility. It supports any type from; structured, semi-structured and unstructured.
4. Strong consistency: The consistency is preferred by HBase compared to availability.
5. Reliability: The data present in the tables are replicated many more times just to avoid the data loss. And the redundancy is maintained. Fault tolerance is also supported by HBase.
6. Supports scalability.
7. Low Latency processing: Random generated read and writes for all the data which is stored are supported by HBase.

5. Database Architecture of MongoDB and HBase:

5.1 Architecture of MongoDB:

The architecture of MongoDB is explained as: (Sankaraoandi et al., 2015). The architecture image is taken from the article. (Sankaraoandi et al., 2015)

Architecture of MongoDB consists of three main components. They are:

- Shard nodes
- Configuration servers
- Routing services or Mongos

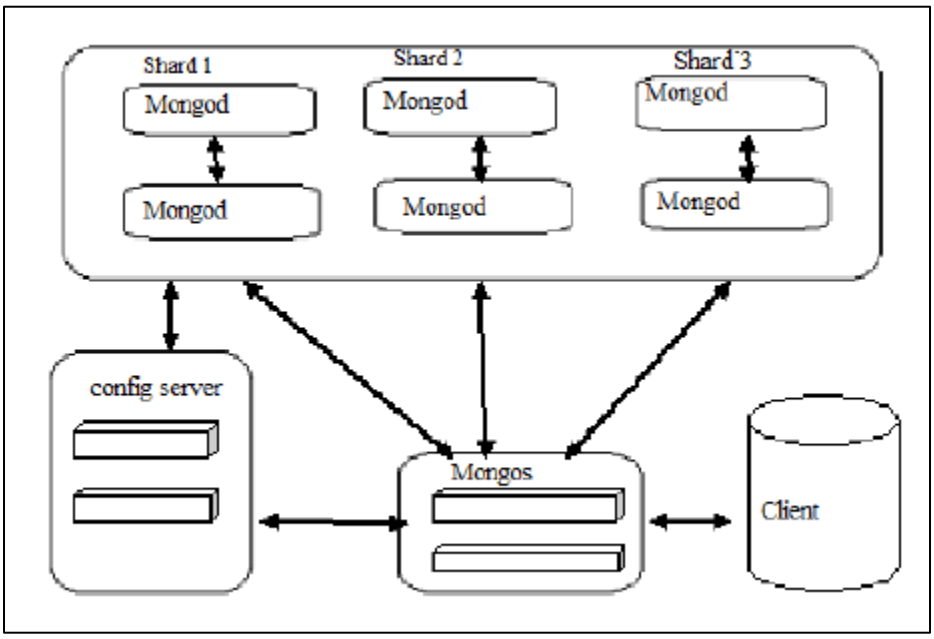

Figure 1: MongoDB Architecture (Sankaraoandi et al., 2015)

Shard nodes: The architecture of MongoDB consists of one or more shards. These shard nodes are used for processing huge number of data and for operations with greater or higher throughput. The set of shard nodes is connected to configuration servers and mongos. Each node of the shard is responsible for loading the data in the database, and it contains the replicated node or one node which has data of that shard. Every shard node organizes with set of replicas. And these replicas sustain the high accessibility, redundancy and availability.

Configuration servers: Docs.mongodb.com (2008) explains the configuration servers as. The metadata from the shard nodes are stored in the configuration servers. The metadata contain the information about state of all data, organization and components of shard data of each shard node. The authentication configuration information is also stored in configuration server.

Routing services or Mongos: Docs.mongodb.com (2008) explains the mongos as. The routing service here is also known as Mongos. This is connected to configuration servers, shard nodes and client.

The data from configuration servers are cache here in mongo instance and it routes the read and write operations to the respective shard node.

5.2 Architecture of HBase:

Architecture of HBase consists of four main components. They are:

- HMaster
- HRegionserver
- HRegions
- Zookeeper

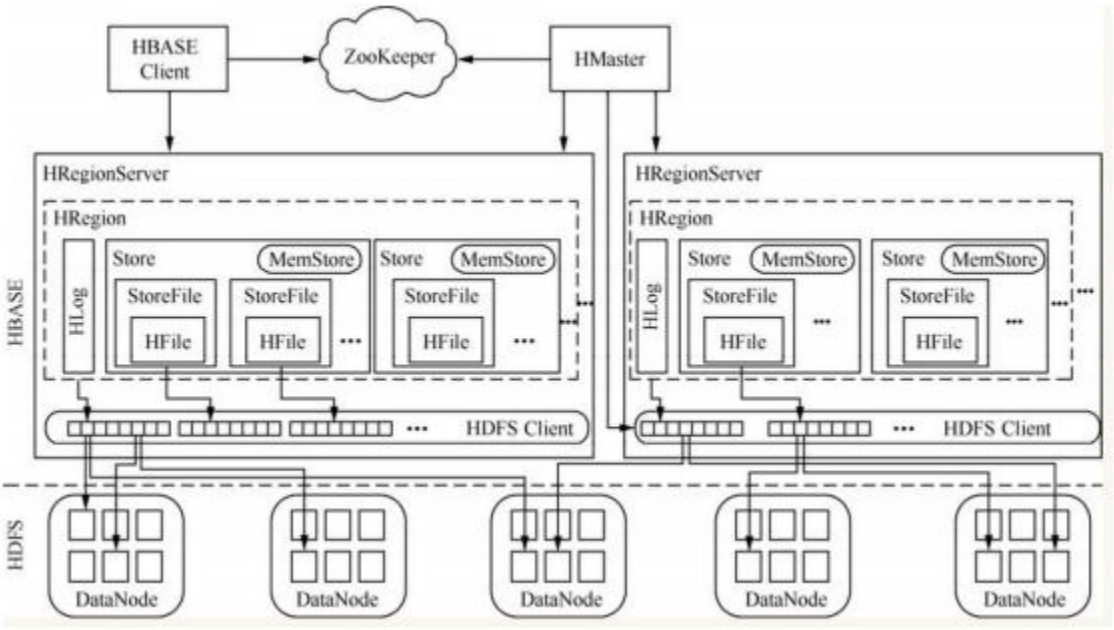

Figure 2: HBase Architecture (Liu, 2013)

HMaster: HMaster is the usage of Master server in HBase architecture. It acts like checking agent to manage all Region Server instances which exists in the cluster. HMaster plays an important role in execution and upholding all the nodes in the cluster. It gives administrator execution and allocates services to region servers. It also allocates regions to region servers. Some features of HMaster are: monitoring load balancing and failover to deal with the load in existing cluster for the respective node.

HRegionserver: HRegionserver consists of regions. These regions make communication with the client and take care of data-related operations. It also handles all the received read and writes requests. By the size of region threshold, it decides the size of the region. All the data present in the distributed cluster is served and managed by HRegionserver. It runs on DataNodes which is present in the cluster of Hadoop. The requests from the client for read and write operations received on region server are assigned to respective regions. Region server can be removed or added as per the requirement.

HRegions: HRegions are the important elements for building the cluster of HBase, which consist of column families and tables. It also comprises of multiple stores, which are for each column family one store is compromised. Two main components of HRegions are Memstore and Hfile.

Zookeeper: Zookeeper is centralized open source framework monitoring servers. It manages and maintains the information of configuration, naming and also distributed synchronization is provided by zookeeper. Distributed synchronization is used for accessing the running applications which are distributed across the cluster. The communication of the client with the region severs is done in zookeeper. Taken care of zookeeper is done by HBase itself in the standalone and pseudo node.

6. Comparison of MongoDB and Hbase: Security Level

Security is the most important part considered related to data. As the huge volume of data is produced every day, maintaining this data and securing it to the maximum level is very important to avoid loss of any data. As this data is handled and transferred over the networks that are not secure the most stimulating part here in terms of security is to secure the data which is spread over various servers. Thus, the security of big data in NoSQL database such as; MongoDB and HBase play vital role.

- Following are the criteria discussed for security of NoSQL database: (Zahid, Shibli, and Masood, 2014)

  - Authentication: "The process of verifying a user's identity who wants to access the resources, data or applications of an organization is known as authentication" (Zahid, Shibli, and Masood, 2014, p.2).
    In MongoDB, for authentication of the user, the Plain MongoDB-CR is supported. Also, it makes use of SSL with X.509 certificates to build a communication which need to be secure between the user and MongoDB cluster. In HBase, user authentication is supported. And for map reduced tasks HBase supports token based authentication. By using SASL (Simple authentication and security layer) user authentication is done with Kerberos on per connection basis" (Zahid, Shibli, and Masood, 2014, p.5).
    The metric value in terms of Authentication for both MongoDB and HBase are considered to be Medium which means both can support single type of authentication for example SSO, SAML either with the client or within the shared cluster. (Zahid, Shibli, and Masood, 2014, p.4)
  - Access Controls: Access control is one of the criteria through which it can be ensured that the person who is authorized can access the system resources. (Zahid, Shibli, and Masood, 2014)
    In MongoDB, within a shared cluster, the access control is kept in enable the mode making use of custom defined and the system defined roles. In HBase, Access Control List (ACL) manages the authorization.
    The metric value in terms of Access control for MongoDB is high as compared to HBase. The Access control for MongoDB is high: Dynamic access control rules following key of minimum benefits, partition of obligations with exceptionally characterized access control strategies, for example, UCON (Usage Control Mode) and so on. (Zahid, Shibli, and Masood, 2014) For HBase the metric values are medium: "Support for RBAC, DAC, MAC, ABAC or FGAC" (Zahid, Shibli, and Masood, 2014, p.4)

- Secure configuration: A security breach may effortlessly happen because of misconfigurations at the OS, database or application layer (Zahid, Shibli, and Masood, 2014). Securing the patches and updates, protocols, services, file and directories, etc. are some of the important categories considered for secure configuration.
  In MongoDB securing the configuration of the database is maintained at actual low level and it is dependent on administrator of the database. In HBase the monitoring feature of security configuration is not present.
  The metric values in terms of secure configuration are low for HBase and medium for MongoDB. That means HBase uses default configuration and MongoDB maintenances less number of protocol for communication (Zahid, Shibli, and Masood, 2014).

- Some security features of NoSQL database are discussed by (Dadapeer et al., 2016):

  - MongoDB:
    1. MongoDB does not have support for authorization, as the authentication is not supported by MongoDB when the process is running in Shared mode.
    2. No facilities are provided by MongoDB for reviewing activities performed in the database.

  - HBase:
    1. HBase supports user authentication which is done by using SASL with Kerberos on basis of each connection and also for map-reduce tasks it supports authentication based on tokens.
    2. In HBase monitoring features and high level auditing are not present.
    3. The logging support is provided by HBase up to data node level.

- (Sahafizadeh,E. and Nematbaksh, A., 2015) has discussed some security issues related to NoSQL database. They are:

  - MongoDB:
    All information related to the data is stored in the form of plain content and no encryption system is available to encode information documents. It implies that any noxious user who doesn't have any rights to use the data can access the records or the data and can extract the data from the database which is present. Hence, data encryption is not done by MongoDB.
    In MongoDB the algorithm used for encryption of the password is MD5. This MD5 algorithm is not secure. Authentication is not supported by MongoDB.
    Authorization is not supported in MongoDB. Auditing is not available in the MongoDB.
    When the data is at rest it is not in encrypted form.

  - HBase:
    Hbase is dependent on SSH for inter-node communication (Sahafizadeh,E. and Nematbaksh, A., 2015,p.70). Using SASL the user authentication is supported in HBase and using ACL the authorization is supported. HBase does not support data encryption.

7. Literature Survey:

There are immense numbers of NoSQL databases with different kind of functionality, features and methods of optimization. For comparing the performance evaluation of NoSQL databases the YCSB (Yahoo! Cloud Serving Benchmark) tool is used. “YCSB tool allows benchmarking multiple systems and comparing them by creating “workloads” (Gaikwad, R. and Goje, A. 2015, p.37).

(Gaikwad, R. and Goje, A. 2015) has discussed the four types of operations within workloads. They are:

- Insert: The Insert is used to insert or add the new record.
- Update: Used to update the record by changing or substituting the value of the particular field in the data.
- Read: Used to read the data from all the files or from the file which has been selected randomly.
- Scan: Used to scan records all together, beginning at a haphazardly chosen record key. The quantity of records to scan is likewise chosen arbitrarily somewhere in the range of 1 and 100.

(Gaikwad, R. and Goje, A. 2015, p.38) Has discussed about the seven workloads:

- Workload A: Updated heavily .
- Workload B: Read mostly.
- Workload C: Read only.
- Workload D: Read latest.
- Workload E: Scan short range.
- Workload F: Read-modify-write.
- Workload G: Write heavily.

(Abubakar, Y., Adeyi, T. and Auta, I. 2014) The author here has provided the framework of YCSB and also the performance comparison is done using YCSB tool of NoSQL databases. The resources were limited in the environment where comparisons were made. The authors have performed the experiment by using single PC. They have defined a core set of the benchmark, and the output was generated for implementation of MongoDB, ElasticSearch, Redis, and OrientDB.

(Kumar, A. and Ganpati, A. 2016) In this paper, the author has calculated and estimated the average run time of MongoDB for read intensive cases with increasing the number of operations and records using YCSB benchmark tool. They have also evaluated the average run time of MongoDB considering different cases this is also done by using YCSB benchmark tool.

(Patil et al., 2011) Here in this paper, the author has described the extensions to the YCSB, which is YCSB++. The description of YCSB++ is done to show the understanding of performance improvements. They have customized the existing cluster monitoring tool to enhance the performance and debugging and to collect the internal statistics, table stores and to offer easy post-test correlation.

8. Performance Test Plan:

Executions or performance likewise rely upon various machines. The set-up used here for performing the test is as follows:

Configuration of Physical Machine:

- Machine name: Dell.
- Processor used: Intel (R) Core (TM) i3 4th generation
- System type: 64-bit Operating System.
- Machine RAM: 8GB
- Operating System: Window 10.

Configuration of Virtual Machine:

- Open stack instance: Ubuntu 18.04.1 LTS (64 bit)
- RAM: 4.0 GB
- JAVA 1.8
- YCSB 0.11.0

Database used:

- HBase-1.4.8 (Hadoop and Hadoop Distributed File System)
- MongoDB-3.2.10

Workloads: YCSB tool provides six default workloads they are:

1. Workload A: 50% Reads, 50% Updates
2. Workload B: 95% Reads, 5% Updates
3. Workload C: 100% Reads
4. Workload D: 95% Reads, 5% Insert
5. Workload E: 95% Scan, 5% Insert
6. Workload F: 50% Reads, 50% Read-modify-write

Workloads used for performing the test are: Workload A and Workload F

Operation counts:

- 200000
- 250000
- 300000
- 350000
- 400000

9. Evaluation and Results: The program was executed three times, and the readings were noted. The readings shown in below tables are the average of all the readings noted after execution of the program.

Workload A:

Workload A has 50 % Read and 50 % Update operations. On the basis of Average Latency and read operations, the comparisons of two databases HBase and MongoDB are carried out in Workload A.

Read Operations:

Here the read operations are considered. The comparisons of Average latency of both the databases HBase and MongoDB are made with the read operations which are recorded. Table 1 shows the readings of Read Operation and Average Latency of both the databases.

	HBase		MongoDB	
Operational Counts	**Read Operation**	**Average Latency**	**Read Operation**	**Average Latency**
200000	99978	363.11	99970.33	190.05
250000	125139	393.84	125079	154.28
300000	149923.67	524.89	149881.33	180.04
350000	175313.33	489.78	175021.67	155.71
400000	200024.33	559.91	199705.33	193.16

Table 1: Shows the readings of Read Operations and Average Latency (us) [Workload A]

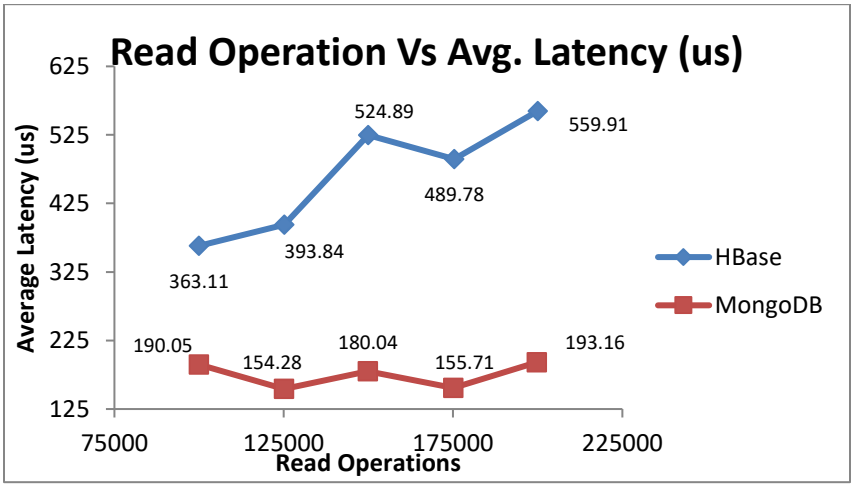

Figure 3: Read Operations Vs Average Latency (us) [Workload A]

From the Fig 3 it is clearly signifies that the average latency of MongoDB is less as compared to HBase. Also when Read operations are increases, the average latency of HBase is also comparatively increases this is not seen in case of MongoDB database. For example, for 400000 operational counts the average latency of MongoDB (193.16) is less as compared to HBase (559.91) for same operational counts. From this it can be concluded that MongoDB is good for read operations compared to HBase.

Update Operations: (Workload A):

Here the update operations are considered. The comparisons of Average latency of both the databases HBase and MongoDB are made for the update operations which are recorded. Fig 4 below shows the line graph of update operations verses average latency for Workload A. As the operational counts increases there is decrease in average latency of MongoDB. Also for HBase the increase in operational counts results in increase of average latency. When the comparisons are made with update operation and average latency the graph shows that the increase in update operation will decrease the average latency for MongoDB and in case of HBase the increase in update operations results with the increase of average latency. From this it can be concluded that MongoDB is good for update operation as it results with decreasing the average latency with increasing of operational counts and update operations.

	HBase		MongoDB	
Operational Counts	**Update Operations**	**Average Latency**	**Update Operations**	**Average Latency**
200000	100022	555.79	100029.67	271.72
250000	124861	546.79	124921	311.9
300000	150076.33	536.01	150118.67	324.37
350000	174686.67	531.47	174978.33	299.5
400000	199975.67	538.76	200294.67	290.23

Table 2: Shows the readings of Update Operations and Average Latency (us) [Workload A]

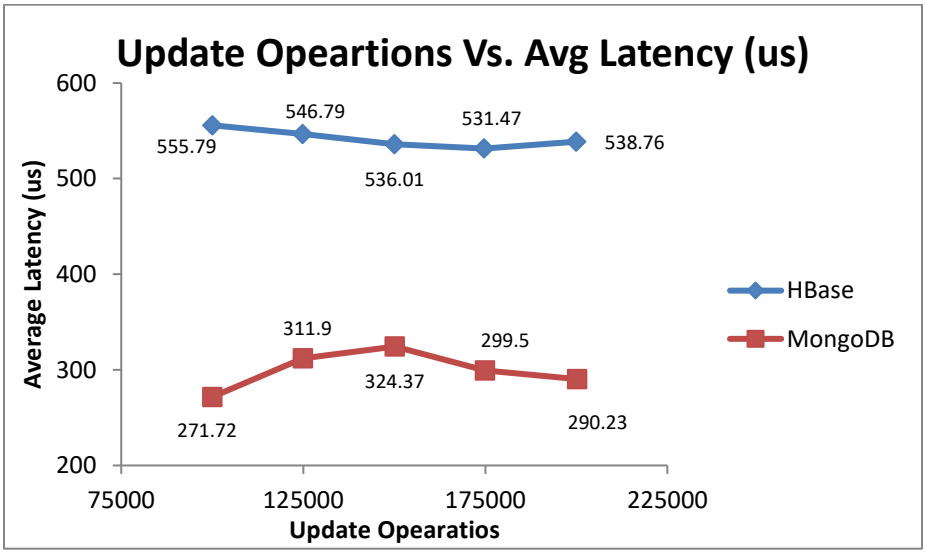

Figure 4: Update Operations Vs Average Latency (us) [Workload A]

Workload F:

Workload F has 50 % Read and 50 % Read-modify-write operations. On the basis of Average Latency and read operations, the comparisons of two databases HBase and MongoDB are carried out in Workload F.

Read Operations:

Here the read operations are considered. The comparisons of Average latency (us) of both the databases HBase and MongoDB are made with the read operations which are recorded. Table 3 shows the readings of Read Operation and Average Latency of both the databases for workload F.

	HBase		MongoDB	
Operational Counts	**Read Operation**	**Average Latency**	**Read Operation**	**Average Latency**
200000	200000	362.35	200000	139.95
250000	250000	320.08	250000	166.83
300000	300000	451.15	300000	136.39
350000	350000	378.75	350000	149.41
400000	400000	437.09	400000	171.27

Table 3: Shows the readings of Read Operations and Average Latency (us) [Workload F]

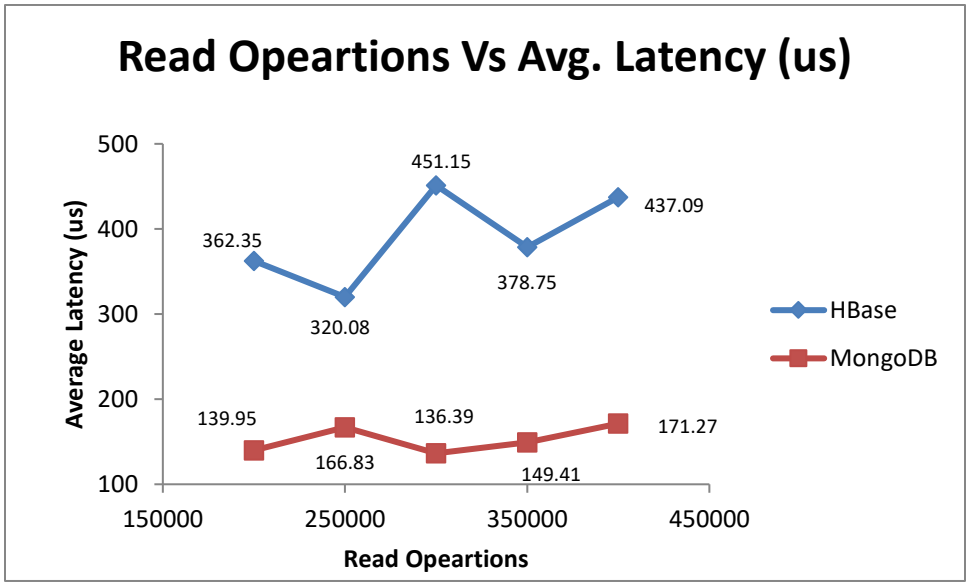

Figure 5: Read Operations Vs Average Latency (us) [Workload F]

From fig 5 and table 3 it can be seen that the average latency of MongoDB is less as compared to average latency of HBase. As the read operations increases the average latency is comparatively less for MongoDB. Hence, the conclusion can be made that MongoDB is good for Read Operations in the workload F.

Update Operations: (Workload F)

Here the update operations are considered for the workload F. The comparisons of Average latency of both the databases HBase and MongoDB is made for the update operations which are recorded.

Table 4 shows the readings of update operations and average latency of both the databases HBase and MongoDB for workload F. Fig 6 shows the line graph of update operations verses average latency for Workload F. For increasing operational counts there is comparatively decrease in average latency of MongoDB with comparisons to HBase. The line graph shows that the increase in update operations there is decrease in average latency of MongoDB. The average latency of MongoDB is less compared to HBase. From this it can be concluded that the MongoDB is good for update operations for the workload F.

	HBase		MongoDB	
Operational Counts	**Update Operations**	**Average Latency**	**Update Operations**	**Average Latency**
200000	99953	556.54	100150.67	281.66
250000	124783	540.96	125058	244.8
300000	150083.33	541.73	150035.67	265.82
350000	175083	548.7	175097	274.95
400000	200091	560.84	199966	263.39

Table 4: Shows the readings of Update Operations and Average Latency (us) [Workload F]

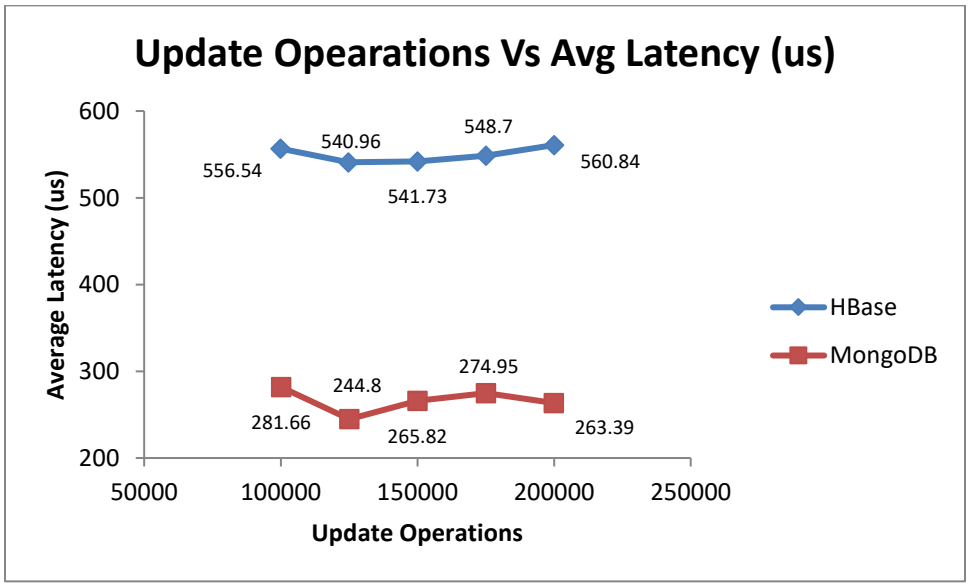

Figure 6: Update Operations Vs Average Latency (us) [Workload F]

Overall Throughput: Throughput is calculated as number of operations per second. The larger the throughput value the good is the database.

Workload A:

Workload A has 50 % Read and 50 % Update operations. The table 5 below shows the overall throughput for each operational count in the workload A for both the databases. Fig 7 below shows the line graph of overall throughput in operations per second for HBase and MongoDB. The overall throughput of MongoDB is increasing as operational counts are increasing. From the line graph it can be concluded that the overall throughput of MongoDB is greater than the overall throughput of HBase. Hence, MongoDB is better than HBase considering overall throughput in the workload A.

	HBase	MongoDB
Operational Counts	**Overall Throughput**	
200000	2125.127	4243.16
250000	2065.09	4201.71
300000	1861.78	3937.893
350000	1928.743	4333.04
400000	1796.257	4141.463

Table 5: Shows the readings of OverAll Throughput (ops/sec) [Workload A]

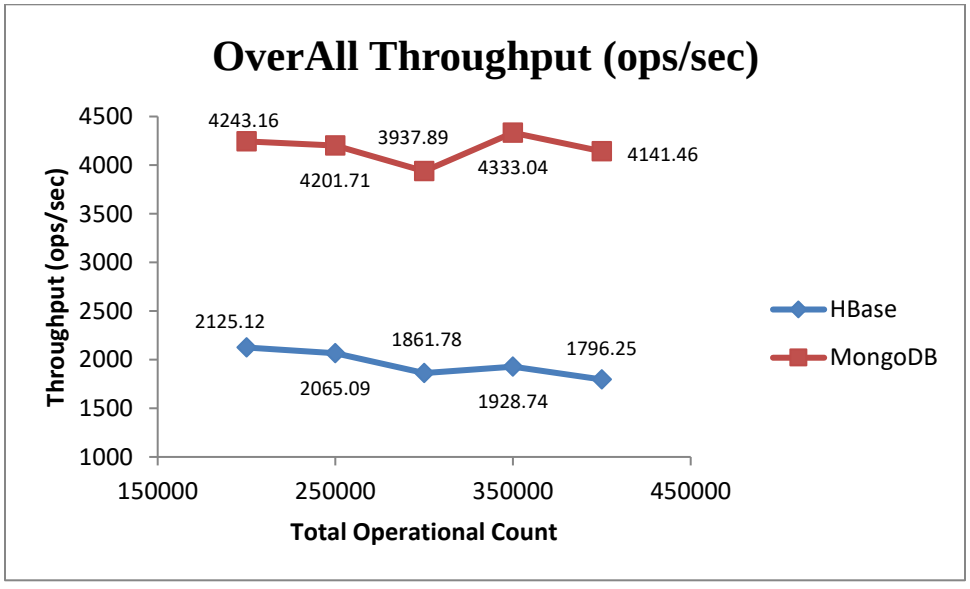

Figure 7: OverAll Throughput (ops/sec) [Workload A]

Workload F:

Workload F has 50 % Read and 50 % Read-modify-write operations. The table 6 shows the overall throughput for each operational count in the workload F for both the databases. Fig 8 shows the line graph of overall throughput in operations per second for HBase and MongoDB. On increasing the operational count the overall throughput of MongoDB is increasing. It can be concluded from the line graph that the overall throughput of MongoDB is larger than the overall throughput of HBase. Hence considering overall throughput MongoDB is good than HBase in workload F.

	HBase	MongoDB
Operational Counts	**Overall Throughput**	
200000	1526.997	3453.917
250000	1654.89	3375.493
300000	1345.53	3629.123
350000	1516.633	3408.983
400000	1371.603	3266.123

Table 6: Shows the readings of OverAll Throughput (ops/sec) [Workload F]

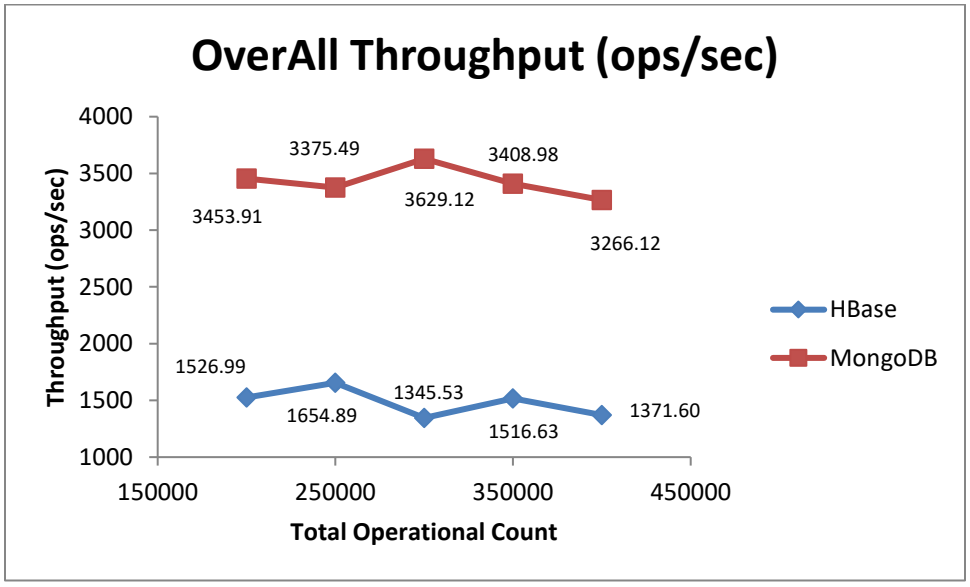

Figure 8: OverAll Throughput (ops/sec) [Workload F]

10. Conclusion and Discussion:

In this report the analysis, key features, evaluation of the two NoSQL databases which are MongoDB and HBase is compared and shown. This comparison is made considering factors such as at security level. Using YCSB tool, on basis of certain operational counts and two workloads, the result is compared and discussed. The analysis is made for overall throughput, read operations and update operations against average latency. From this analysis it is concluded that the MongoDB NoSQL database is good for read and update operations for both the workload A and F. That means the average latency of MongoDB is less compared to HBase in workload A and F for read and update operations. Considering overall throughput, MongoDB is having greater throughput for both the workload A and F. Hence the conclusion is that MongoDB is good than HBase for both the workload A and F.

11. References:

1. Wikipedia (2018) *Big Data.* Available at: https://en.wikipedia.org/wiki/Big_data [Accessed 9 December 2018]

2. Docs.mongodb.com (2008) *The mongoDB 4.0 manual — mongoDB manual*. Available at: https://docs.mongodb.com/manual/ [Accessed 9 December 2018]

3. Tutorialspoint (2018) *MongoDB-Overview.* Available at: https://www.tutorialspoint.com/mongodb/mongodb_overview.htm [Accessed 9 December 2018]

4. Wikipedia (2018) *MongoDB.* Available at: https://en.wikipedia.org/wiki/MongoDB [Accessed 9 December 2018]

5. Docs.mongodb.com (2008) *Introduction to mongoDB.* Available at: https://docs.mongodb.com/manual/introduction/ [Accessed 9 December 2018]

6. Wikipedia (2018) *Apache HBase.* Available at: https://en.wikipedia.org/wiki/Apache_HBase [Accessed 9 December 2018]

7. Tutorialspoint (2018) *HBase – Overview.* Available at: https://www.tutorialspoint.com/hbase/hbase_overview.htm [Accessed 9 December 2018]

8. Kumar, V. and Shindgikar, P. (2018) *Modern big data processing with Hadoop*. Google Books. Available at: https://books.google.ie/books?id=55lUDwAAQBAJ&printsec=frontcover&dq=modern+big+data+processing+with+hadoop+by+Prashant+Shindgikar,+V.+Naresh+Kumar&hl=en&sa=X&ved=0ahUKEwiiraf_4qXfAhUaThUIHXthACMQ6AEIKjAA#v=onepage&q=modern%20big%20data%20processing%20with%20hadoop%20by%20Prashant%20Shindgikar%2C%20V.%20Naresh%20Kumar&f=false [Accessed 9 December 2018]

9. Sankaraoandi,S.,SaiBaba,M.,Jayanthi,S. and Soundararajan,E.(2015) 'Storing of Unstructured data into MongoDB using Consistent Hashing Algorithm', *International Journal of Emerging Technologies in Engineering Research*, 3(3),pp.79-80. Available at: https://www.researchgate.net/publication/289519350_Storing_of_Unstructured_data_into_MongoDB_using_Consistent_Hashing_Algorithm [Accessed 10 December 2018]

10. Docs.mongodb.com (2008) Config servers. Available at: https://docs.mongodb.com/manual/core/sharded-cluster-config-servers/ [Accessed 10 December 2018]

11. Docs.mongodb.com (2008) Mongos. Available at: https://docs.mongodb.com/manual/core/sharded-cluster-query-router/ [Accessed 10 December 2018]

12. Liu, J. (2013) *The Big Data Processing.* Beijing: People Posts& Telecom Press Publisher

13. Guru99.com *HBase Architecture, Data Flow, and Use cases.* Available at: https://www.guru99.com/hbase-architecture-data-flow-usecases.html[Accessed 10 December 2018]

14. Zahid,A.,Shibli,A.and Masood,R.(2014)'Security of Sharded NoSQL Databases: A Comparative Analysis', in *Conference on Information Assurance and Cyber Security*. Pakistan, June 2014, pp. ,ResearchGate.doi:10.1109/CIACS.2014.6861323 Available at: https://www.researchgate.net/publication/269293971_Security_of_sharded_NoSQL_databases_A_comparative_analysis [Accessed 10 December 2018]

15. Dadapeer,Indravasan,N. and Adarsh.G (2016) *A Survey on Security of NoSQL Databases*. Available at: http://www.ijircce.com/upload/2016/april/194_39_A%20survey.pdf [Accessed 11 December 2018]

16. Sahafizadeh,E. and Nematbaksh, A. (2015) 'A Survey on security issues in big data and nosql',Advances in Computer Science: an International Journal,4(4),pp.69-71,Academia.doi:ISSN:2322-5157. Available at: https://www.academia.edu/14758127/A_Survey_on_Security_Issues_in_Big_Data_and_NoSQL?auto=download [Accessed 11 December 2018]

17. Gaikwad, R. and Goje, A. (2015) 'A Study of YCSB –tool for measuring a performance of NOSQL databases', *International Journal of Engineering Technology and Computer Research,* 3(6), pp.37-40, IJETCR. Available at: https://ijetcr.org/index.php/ijetcr/article/view/275/272 [12 December 2018].

18. Abubakar, Y., Adeyi, T. and Auta, I. (2014) 'Performance Evaluation of NoSQL Systems Using YCSB in a resource Austere Environment', *International Journal of Applied Information Systems,* 7(8), pp. 23-27, IJAIS. doi: ISSN:2249-0868 Available: at: https://research.ijais.org/volume7/number8/ijais14-451229.pdf [Accessed 12 December 2018].

19. Kumar, A. and Ganpati, A. (2016) 'MongoDB for Read Intensive Cases Using YCSB', *International Journal Of Engineering And Computer Science*, 5(10), pp. 18564-18567, IJECS. doi:ISSN: 2319-7242 Available at: http://www.ijecs.in/index.php/ijecs/article/view/2684/2481 [12 December 2018].

20. Patil, S., Polte, M., Ren, K., Tantisiriroj, W., Xiao, L., López, J., Gibson, G., Fuchs, A. and Rinaldi, B.(2011) 'YCSB++: Benchmarking and Performance Debugging Advanced Features in Scalable Table Stores', ResearchGate. doi: 10.1145/2038916.2038925 Available: at: https://www.researchgate.net/publication/239761095_YCSB_Benchmarking_and_Performance_Debugging_Advanced_Features_in_Scalable_Table_Stores [13 December 2018]