IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 25, Issue 3, Ser. I (May. – June. 2023), PP 01-12
www.iosrjournals.org

A Study of Performance Evaluation and Comparison of NOSQL Databases Choosing for Big Data: HBase and Cassandra Using YCSB

Nasrullah[1], Nijad Ahmad[2], Dr. Neeta Sharma[3], Mohammad Saber Niazy[4]
[1] Dean (Faculty of Computer Science at Khurasan University, Nangarhar, Jalalabad Afghanistan)
[2] Vice Chancellor (Khurasan University, Afghanistan)
[3] Associate: Prof (Dept. of CSE, Noida International University Grater Noida, U.P)
[4] Assistant: prof (Faculty of Computer Science at Khurasan University, Nangarhar, Jalalabad Afghanistan)

Abstract: *For years' data has been a critical part of the technology and data has been perceived with the growth in technology and the population. This data is often referred to as NoSQL and unstructured data. Over the period of time, it is growing in complexity for traditional database management systems to manage some enormous databases in a virtual cloud environment. Nowadays different cloud services are offered like NoSQL databases to manage such Non-relational data which is dressing different requirements like availability, reliability, performance, safety as well as security. Hence there is a need to evaluate processes and find results in the performance and behavior of different NoSQL databases HBase and Cassandra using YCSB. The scale of big data will come from different sources it will be generated and processed from the TB level, PB level, and ZB level this range of big data processing Platforms have various rules policies guide line tools, and techniques a different level. Some common revolutions in data management systems have occurred recently, like Big Data analytics, data visualization, and NoSQL databases. They will be evaluated for different purposes, their independent developments complement each other in the given criteria. Their convergence would benefit businesses tremendously in making real-time decisions using volumes of complex data sets that could be structured or unstructured and semi-structured. several software as services solutions have emerged in supporting Big Data analytics, on the other hand, many NoSQL database packages have arrived in the market nowadays cloud offering anything as services. However, they lack independent benchmarking compression and comparative evaluation in every solution. The aim of this paper is to provide an understanding of their contexts of HBase and Cassandra performance for Big Data Analytics and an in-depth study to compare the features of two main NoSQL data models that have evolved using YCSB.*

Keywords*: Big Data, Hadoop, NoSQL Databases, Cassandra, HBase, YCSB Application.*

Date of Submission: 28-04-2023 Date of Acceptance: 07-05-2023

I. INTRODUCTION

Data is growing in complexity with the rise in data over the period of time 2.5 billion Bytes of data is generated every day and it is the biggest concern for every organization to handle this huge amount of data. A large amount of data is being generated from different sources like Facebook, WhatsApp, and YouTube which are the corners of the internet. This exponential data growth day to day is represented by big data. It is helping different use cases in the present-day in a data-driven environment in the virtual cloud environment and there is a need to manage it to velocity, volume, variety, and values. The traditional way of managing the databases using relational database management systems could not handle this huge amount of data because of the volume, and processing power and they are capable of storing internal data which is schema-based and only in a number of predefined formats for different data sets. The Big Data paradigm is gradually changing the present data storing techniques, processing, administration the methods of analysis in a virtual cloud environment [1]. This led to new developments and deployment in the design architecture of databases to handle big data bud due to the data processing rate and visualization rate being very low The NoSQL data do not have a fixed and free define structure or schema and it is enormous in volume. They allow storing of multiple forms of data which are structured, unstructured, or even semi-structured data for storage [2]. They store data in the form of column rows families, key-value data fairs, and document data stores. Hence NoSQL databases are designed to replace the traditional SQL DBMS. Since these databases are non-relational, the query language support is subjective. Different types of NoSQL database management systems are being used in present-day applications as they are not dependent entirely on queries for processing storing and data management. [3][4][5] These databases are extensively used in environments where data do not rely on a relational model and structure data. There are different NoSQL databases, with every category depending on the type of data stored. They are categorized into document-based, key value-based, column-based, etc. Each type of database serves user-specific data storage requirements in its own perspective. Two such databases are Apache HBase and Apache Cassandra. Both databases are NoSQL databases and are popularly used for present-day non-relational database management. With the rise in cloud technologies, virtualization has become one of the widely adapted technologies. Open source offerings such as OpenStack are providing different platforms to execute the workloads on every virtual machine Though the virtual environments are scalable and highly performing as per the requirement, there are certain challenges when it comes to latency, performance, and bandwidth allocation. As a solution, offerings such as Amazon's Cloud Front are providing edge locations to replicate and store the data to the closest possible availability reliability all zones. However, the performance of the databases in very virtual environments and clouds has remained a question to the research community in every platform. The types of NoSQL databases are given in Figure 2 [8][9]. This paper aims to evaluate the performance of NoSQL Databases HBase and Cassandra in a virtual cloud environment, HBase and Cassandra performance that are deployed over every single virtual machine in OpenStack and YCSB. The later sections of the paper, give an understanding of the key characteristics and the architectures of both the databases and the differences using some graphs. As a part of the study, the later sections of the paper also cover the performance evaluation techniques implemented in previous research concluding with the evaluation and the results.

II. BIG DATA

Big data is a large dataset that cannot be handled by traditional computing. Big data is not merely data, but today it becomes a complete subject in computer science and Engineering fields, which involves various producers, tools and techniques, frameworks, and materials. It is so big and large size that traditional or conventional data processing applications are inadequate in handling these techniques.

III. HADOOP ECOSYSTEM

The Hadoop Ecosystem is used for a top-down structure of Hadoop 1. X in this system only has two units. Ex. HDFS and Map Reduce. The Map Reduce process the whole data and the result automatically get stored in the Hadoop distributed files system. [13]. In the second version of Hadoop 2. x there are some new different compounds added to the ecosystem i.e. YARN (Yet Another Resource Negotiator). The YARN is the same as Map Reduce but the way and structure of processing is a little bit different. YARN processes the data in a container manner. The containers are the logical units that consist of the resource and task. With the presence of YARN, when a deadlock situation occurs which usually use to appear in 1. x is minimized in 2. x. In the Hadoop ecosystem every data breakdown into data chunks also known as the data blocks every chunks and blocks stored different records.

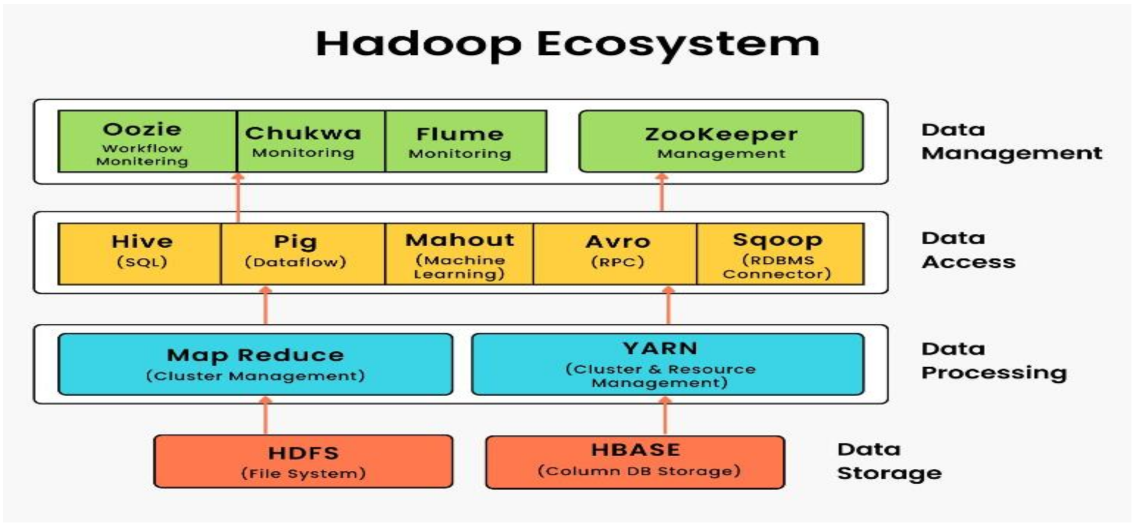


Figure 1: Turing.com/kb/Hadoop-Ecosystem

3.1 NOSQL Database Architectures Apache HBase

HBase is a Database designed for Hadoop Distributed File systems and built on the Map Reduce framework. The difference between HBase and Hadoop is that HDFS is a file system for storing large datasets in every node or file and unlike a normal file system, it fails to provide instant record lookups and updates [5]. But HBase stores all the record and data in the form of indexed store files which is stored in HDFS. This allows to achieve fast lookups of the files and records. The architecture of the HBase all the master nodes and several slave nodes. Figure 2 Explains all the architectural and very components of HBase. It consists of Master and slave nodes. A single master node is present in the HBase architecture which assigns the regions and load balancing called Master. These region servers are the computers in a Hadoop every cluster node that serves different regions with other clusters. Each region can only be handled can control by one region server in every server. When a written request is sent by the client, it is received by the HBase Master and it is further sent to the specified region server. Zookeeper is used to monitor all the systems and servers.

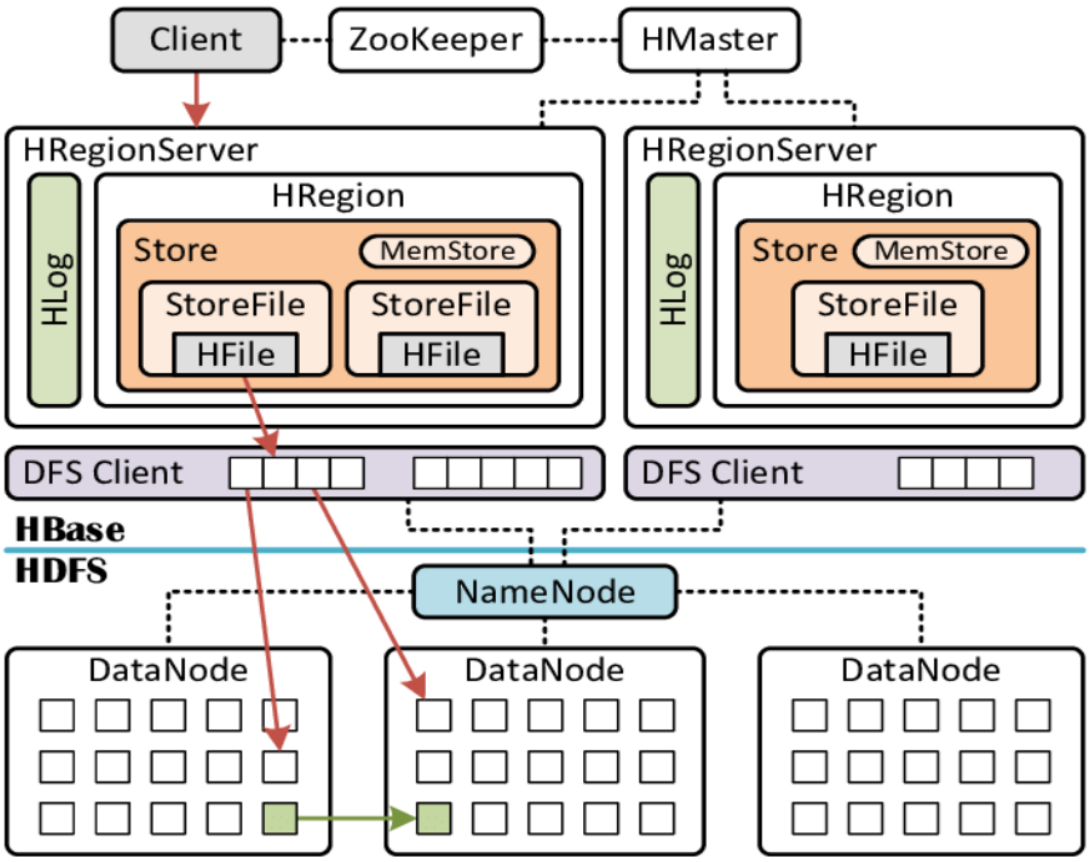


Figure 2: Hadoop NoSQL Database HBase Realizing Cloud-Based Big Data Infrastructures

IV. NoSQL DATABASE ARCHITECTURE: CASSANDRA

4.1 NOSQL Database:

In NOSQL Database different workloads in Big Data handling without a single point of failure in a virtual cloud environment is one of the main motivations overdue the design and development of the Apache Cassandra platform. If in a cluster running with Cassandra, all nodes are interconnected peer-to-peer fashion, and the distribution of all types of data is achieved over all the nodes of the cluster. Each node in the cluster is free and separated by itself and communicates with other nodes and all the nodes are given some similar roles.

4.1.1 Cassandra Platform Ring Design with Token Nodes:

The scalability, reliability, avaliblaity, performance, and continuing to bring availability to the platform are the three key features of the Cassandra database and its architecture that contributes every ring to do it. So ring type architecture in which every master node is absent in the region, is often referred to as a masterless ring. [12] This ring-type architecture makes Cassandra less complex to install and the main domain difficult to maintain. Due to the scalable property of the architecture, Cassandra is capable of managing huge data sets that is divided from master to slave. It also allows several nodes. of operations that can be performed in multiple data centers. For example, when a workload is a specified node, that node is not entirely responsible for overall operations but the workload is spread across all the nodes in every cluster to hence and contribute to the operations and the behavior of the cluster during the processing environment [7].

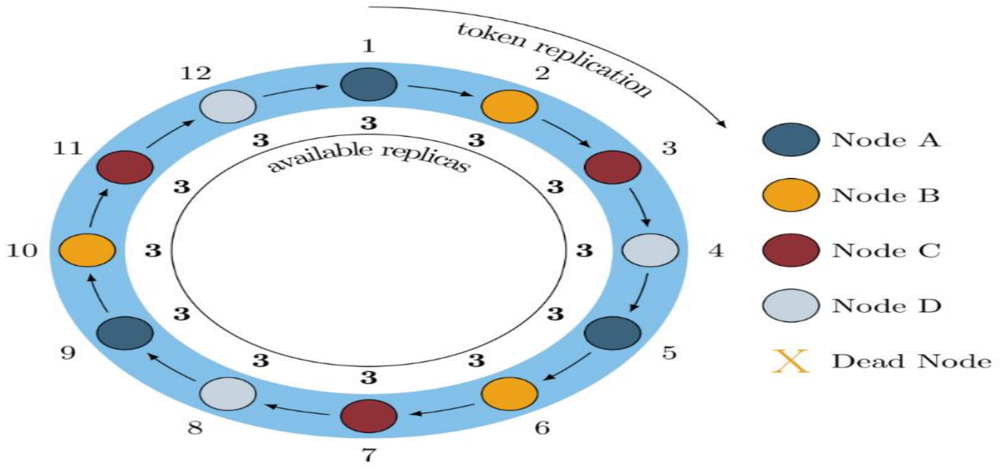


Figure 3: Three layers of Cassandra Architecture [3] From engineeringblog.yelp.com.html

V. HBASE AND CASSANDRA

HBase and Cassandra are two different NoSQL Databases which is licensed by Apache Foundation. both the Databases are non-relational databases they share some identical features and furcation and some similarities such as being wide-column structure NoSQL Database stores based on Big table are prominent. As HBase runs on top of the Hadoop Ecosystem, it does not support query languages but it works with HBase shell environment which is based on Ruby languages and can also include features such as Drill and Hive and distributed files system. Cassandra supports its own Cassandra query language. Both of these databases offer different security policies and tools and also different transaction management system.

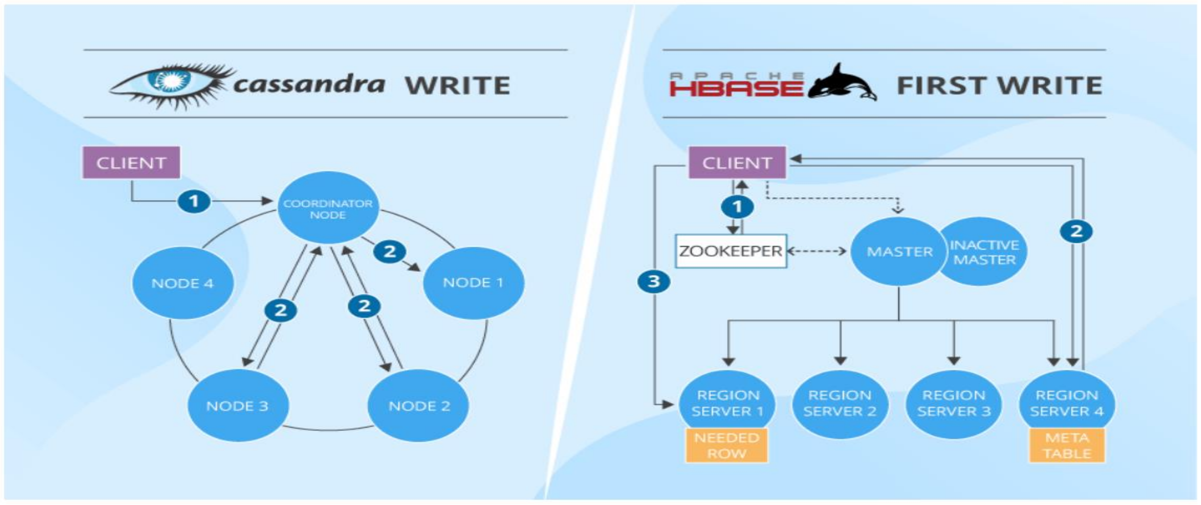


Figure 4.scnsoft.com/blog/Cassandra-vs-HBase

5.1. Security Privacy and Trust

Cassandra and HBase have their security privacy and trust challenges and workarounds. The first issue with high security in these Databases can be loss of performance during the huge data sets processing. But both of them offer unique features and functions to address these security issues in the cloud. They ensure the security of data by authentication and authorization in HBase and Cassandra. But, Unlike HBase Cassandra has more rigid security features such as internode or connected in every region and client-to-node data encryption on one side. But HBase makes use of other technologies to secure the communication medium between the client and the cluster.

HBase offers cell-level security features. It offers the following options;

1. Role-based Challenging Security Level
2. Authentication and authorization
3. Data Security, Logging records

HBase offers authentication and authorization for both server-side and client-side security and high availability. As it maintains the user credentials it also provides some secure storage for both side's credentials. It uses different kinds of protocols and policies to authenticate the traffic into the different databases. As HBase also works in distributed fission, the database servers can authenticate themselves from the source of communication with each other to secure and provide trustable communication in the different clusters. A credential store is provided to securely store the data about the group of users' credentials rules. It is often stored in an external file. HBase also ensures to provide security by providing different kinds of roles. It is a secure approach to authorizing the user access to the contents of the database in a virtual cloud environment. It allows users to create their own roles and also provides default roles. Moreover, it is essential to define the scope base policies of each role to ensure finer granularity, especially for highly sensitive data. Logging is one of the most authentic security features of HBase which is helpful in maintaining master and slave nodes in the data database availability and security. HBase achieves database security availability and high trust by encrypting procedure Database nodes partitions.

5.2. Security: Cassandra

Cassandra provides two different security functionalities to the database.

1. Client Authentication and Authorization
2. Every Inter-node with master client communication encryption with TLS/SSL
   However, these features come disabled by the initial platform as Cassandra structure which is easily found by the members of the cluster nodes. Proper configuration in every layer with given security features. can help in ensuring the security of the cluster. Communication security is achieved by enabling SSL / TLS/FIPS encryption for inter-node communication and between client machines and Databases.

5.2.1 Cassandra: Implementation

Cassandra architecture also implements different roles in the Database which can be applied from a single user to a group of users. These roles can define permissions, policies, and guidelines and authenticate the group of users.

5.2.1.1 Cassandra: Authentication and Authorization

Cassandra Authentication and authorization also maintain security in the database. The authentication is provided by an authenticator class in Cassandra during the configurations in the open stack cloud. It allows the authenticator which is responsible check the authentication without any credentials required during the login, and the Password Authenticator which takes the user credentials such as passwords, and stores them in a mic table. Authorization is handled by an authorized user or group of user configurations Allow authorizer which does not require any checking permissions to the user roles. It also comes with the default Cassandra authorizer which handles permission management policies and stores the user data in a system table.

VI. LITERATURE REVIEW

we have described the NoSQL database in Big Data technologies and how to process and visualize the data. We categorized different NoSQL databases, particularly the CAP theorem and then we discussed the Big Data life cycle. Moreover, we have discussed different types of strengths, weaknesses, comparisons, and evaluations of various NoSQL databases in a virtual cloud environment. The pointing and output from our discussion would be helpful to the business leader and their organizations in selecting different types of data for processing using a NoSQL database in order how to manage and store Big Data [1]. HBase describes the setup of a single-node standalone HBase and its Master Nodes A standalone instance has all HBase daemons and all Master, and Region Servers in a virtual cloud environment, and Zookeeper running in a single JVM persisting to the local file in the local system s. [2] It is our most basic deployment profile in Cassandra and HBase performance analysis. We will show you how to process organized and create a table in the HBase shell CLI Command line, insert rows into the table using unstructured and semi-structure data, operations against different tables, and start and stop HBase [3]. DBMS and NoSQL are non-relational and unstructured data they offer more functionalities to store processes and analyzed different kinds of data. From the performance evaluation conducted over HBase and Cassandra using YCSB, several variations in the database behavior have been observed as per the requirement and presented in the evaluations. H Base being backed files up with HDFS in Map Reduce, has a r larger ecosystem than Cassandra. Both databases have performance exceptions at higher workloads for big data Visualization. At workload A, Cassandra has to fluctuate latency A during read and over-the-write different types of operations but HBase presented significantly less in their ecosystem latency during reads operations. with the rest of the operations as insert, searches, and updates, but Cassandra has increased latency per operation. Runtime for each operation can also be considered as per schedule in every file system but to know the performance of the virtual every node in the machine running both databases other metrics had to be considered. however, from the observations, Cassandra has higher runtimes processing results than HBase. [4]. There has been some kind of research that was carried out in the field of every HBase ecosystem of performance evaluation of DBMS Applications. Several techniques and methodologies were proposed to benchmark the comparative study for the performance of NoSQL databases. [5]. They proposed a framework to evaluate different results in a virtual cloud environment, and analyze and predict the behavior of Cassandra with other databases. This architecture helped in forming the challenges and risk analysis that are faced in evaluating NoSQL databases. And it was observed that the behavior of every task in the evaluation framework depends on the database characterization and the unit testing system. However, their model is inclined toward nodes in machine learning and prediction of every database's behavioral patterns. [6] gave insights for the evaluation entire memory of every database. The study emphasized the available variations of NoSQL databases HBase and Their study drew evaluations as compared to MongoDB, Me cached, Radis, and Cassandra. Software based on Java has been used to draw different evaluations over metrics such as the execution time, read, and load time per operation. And their tool was based on the studies conducted with different results [6]. suggests that NoSQL databases are generally characterized by the properties such as no-schema data models, horizontal scalability, and simple cloud deployment. The study also suggests that there is a need to identify the correct system requirements before deployment to avoid overprovisioning. The benchmarking was done between MongoDB, HBase, and Cassandra databases deployed on Amazon EC2 and Amazon EC3. YCSB was used as a benchmarking tool. They have tested each of the databases with specific workloads in different requirements and deployed virtual instances offered by Amazon EC3. The proposed modeling approach suggested complex modeling and tools of replication to accurately depict the performance of a replica A. [7] Brian Cooper emphasized on two-tier benchmarking sub-system in which one focuses on the performance and reliability, availability of the database while the other focuses on the impact on performance due to the scalable feature of the database. Their benchmarking system measures metrics such the inserts, delete searches, updates, reads, and scans. And they have defined certain workloads read and write to choose from depending on the targeting metric.

[7] compared the performance and evaluation working of SQL databases with NoSQL databases. The comparison is done on a specific dataset as per organizational scale. Their study suggested the implementation of both types of databases. The transactions were tested over storing the digital media with respect to social media platforms the and final result will be simulated in the social network environments to test different types of workloads. Their experiment results suggested that NoSQL databases surpass form SQL operations and their rules. when it comes to transactions and transforming data for storing digital media. Similar evaluations were conducted as per the requirement. [8] Between, Cassandra, MySQL, and HBase on the write-heavy and load operations. Their experimental implementation for both was executed with the help of a web-based REST application. The study also emphasized the CAP properties of the databases which suggest the trade-offs between each database management system of their own policy and security rules. They made use of a Java application as JVM that is executed with the help of all files loading of Representational State data per unit of time transfers. It puts the data in the database using HTTP and POST request methods. The standard metric for the number of throughput selected transactions per second and the application was hosted on a Tomcat server and also in an open stack cloud. Their test results suggest that HBase speeds up twice as fast as MySQL database which is a relational database. It is also observed that Cassandra gives significantly fast and writes even in a write-heavy application. [9] NoSQL databases have gained prominence in recent years because of their ability to scale easily and provide support for large amounts of structured as well as unstructured data. User applications that need to perform heavy write operations have started making a paradigm shift to NoSQL databases with the overall intention to enhance performance and reliability. Eventually, various types of NoSQL databases have come up that have several features that suit different user-specific needs. In this paper, the authors evaluate the performance of MySQL, Cassandra, and HBase for heavy write operations by a Web-based REST application. [10]. In this work, we studied the distributed management of massive remote-sensing image data based on the NoSQL database and pyramid map. We presented a storage method to divide remote sensing image data into blocks based on a pyramid

map and store the data blocks in three different database models Cassandra, HBase, and MongoDB. The feasibility of the presented storage model for massive remote sensing image data has been verified. Finally, we came out with the Cassandra as the most suitable database [9] [11]. analyzing the results from the three NoSQL databases, MongoDB 4.4 as a document store, Cassandra 4.0.3 as column store, and Redis 6.2.6 as a key-value store, and after executing six workloads made up of 100000, 250000, 500000, 750000, and 1000000 operations, we came to the conclusion that the numerous optimizations used by the designers of NoSQL solutions to improve performance, such as good cache memory operation, have a direct impact on the execution time. [12]. [13] [14] we have proposed transformation rules that ensure the successful translation from conceptual DW schema to two logical NoSQL models (column-oriented and document-oriented). We proposed two possible transformations namely: simple and hierarchical transformations. The first one stores the fact and dimensions into one column-family/collection. [15] [16] [17] Increasing requirements for scalability and elasticity of data storage for web applications have made Not Structured Query Language NoSQL databases more invaluable to web developers. One such NoSQL Database solution is Radis. A budding alternative to the Redis database is the SSDB database, which is also a key-value store but is disk-based. This research work aims to benchmark both databases using the Yahoo Cloud Serving Benchmark YCSB a platform that has been used to compare and benchmark similar NoSQL database systems. Both databases were given variable workloads to identify the throughput of all given operations. The results obtained show that SSDB gives better throughput. [18] for the majority of operations to Redis's performance.

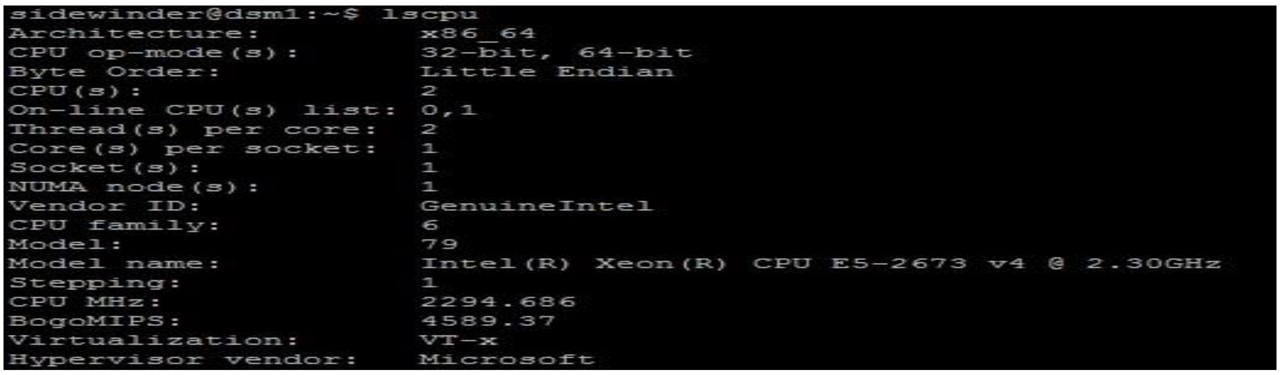
VII. PERFORMANCE TEST PLAN

Performance of HBase and Cassandra, a Java-based tool YCSB has been taken into consideration. The test follows several iterations to test both databases in different workloads and processing times. A test harness file is used to bind the databases with YCSB. The configurations and implementation files with specifications required for the test environment are like this.

7.1. System Specifications

Native Machine:

1. Processor type: 4CPUs 2.0 GHz,
2. AMD A12-89410 APU with AMD Radeon R7
3. Graphics Installed Memory: 16 GB
4. Operating System: Windows 10



```
sidewinder@dsm1:~$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                2
On-line CPU(s) list:   0,1
Thread(s) per core:    2
Core(s) per socket:    1
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 79
Model name:            Intel(R) Xeon(R) CPU E5-2673 v4 @ 2.30GHz
Stepping:              1
CPU MHz:               2294.686
BogoMIPS:              4589.37
Virtualization:        VT-x
Hypervisor vendor:     Microsoft
```

Figure 5 lSCPUs Console Processing Power of Virtual Instance

7.2 Virtual Instance: Open Stack

1. Processor: Intel Xeon CPU E5-2673 v4 2.30 GHz
2. VCPU: 8
3. Memory: 16 GB per system
4. Hadoop 4.1.2 for HBase Ecosystem
5. Apache HBase 2.4.0
6. Apache Cassandra 5.11.4
7. Per System Load Balancing
8. Casandra Core Base Processing
9. YCSB-0.14.0 Yahoo!
10. Cassandra performs
11. HBase Performance
12. Cloud Servicing Benchmark

Figure 5 shows the other system specification of the virtual instance launched on OpenStack. This is achieved by executing the LSCPU command.

7.3 Workloads Selected:

YCSB [7] is an open-source platform providing a benchmarking tool that offers different workloads to test different reads and writes and executed commands in a distributed environment. five different workloads. these workloads vary in nature.

7.3.1 The Following Are the Selected Workloads:

1. Workload A: Update heavy with 95-50 reads and updates
2. Workload D: Read the latest with 98-10 read-inserts

7.4 **No. of Accounts Defined:**
The performance tests are executed with a different number of operational counts.

1. 100000
2. 125000
3. 185000
4. 300000
5. 250000
6. 350000

The performance test strategy for every system is to install and run standalone HBase and Cassandra databases in a distributed fashion on a virtual machine deployed on an OpenStack cloud environment. To achieve VM was launched with the next virtual machine configuration. Hadoop was installed to support both types of deploying underlying HDFS systems of HBase. Later HBase and Cassandra were installed and tested. YCSB was installed and configured to test both databases. A set of configuration files called test harness over the first VM configuration to use to define the test database and the workloads.

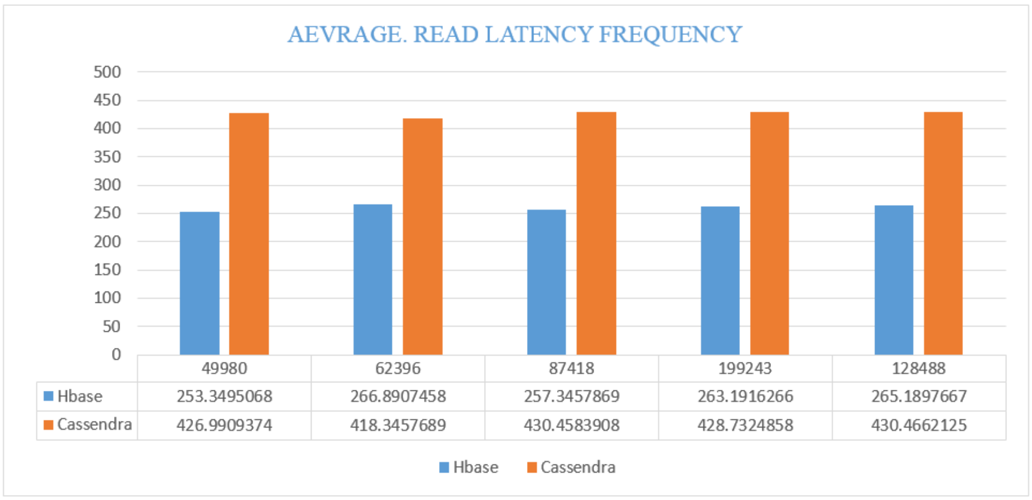
VIII. EVALUATION OF HBASE AND CASSANDRA

To evaluate the performance and final result of HBase and Cassandra, Performance With YCSB - Volt Active Data YCSB-0.14.0 was used. As suggested evaluation studies over NoSQL databases and the performance and processing evaluation of distributed virtual cloud environments, Yahoo Cloud Servicing Benchmark A and D is chosen to draw different analysis in its daily operations. From point to point workloads offered by YCSB, workload A, and workload D are selected in YCSB-0.14.0. Both of these workloads have different specifications as per user requirement. The evaluation of both databases is executed on a single virtual node with 2 VCPUs and 16GB of memory. After each test run, YCSB generates different metrics of the operations involved in the workload. Metrics such as per unit runtime, evaluation time, scavenge time, sweep time etc. Since the operations involved in the workloads present and past as the metrics for all selected data for final result evaluation, the average latency is taken over the operations performed in every table.

8.1. Workload in Each System

The YCSB client's side operation workload A is updated and heavy. It uses read () and update () to compare methods to evaluate each input in different systems. These methods represent the standard create, read, update, and delete operations in every database using YCSB.

8.1.1 Avg. Read Latency Over Read Operations Frequency

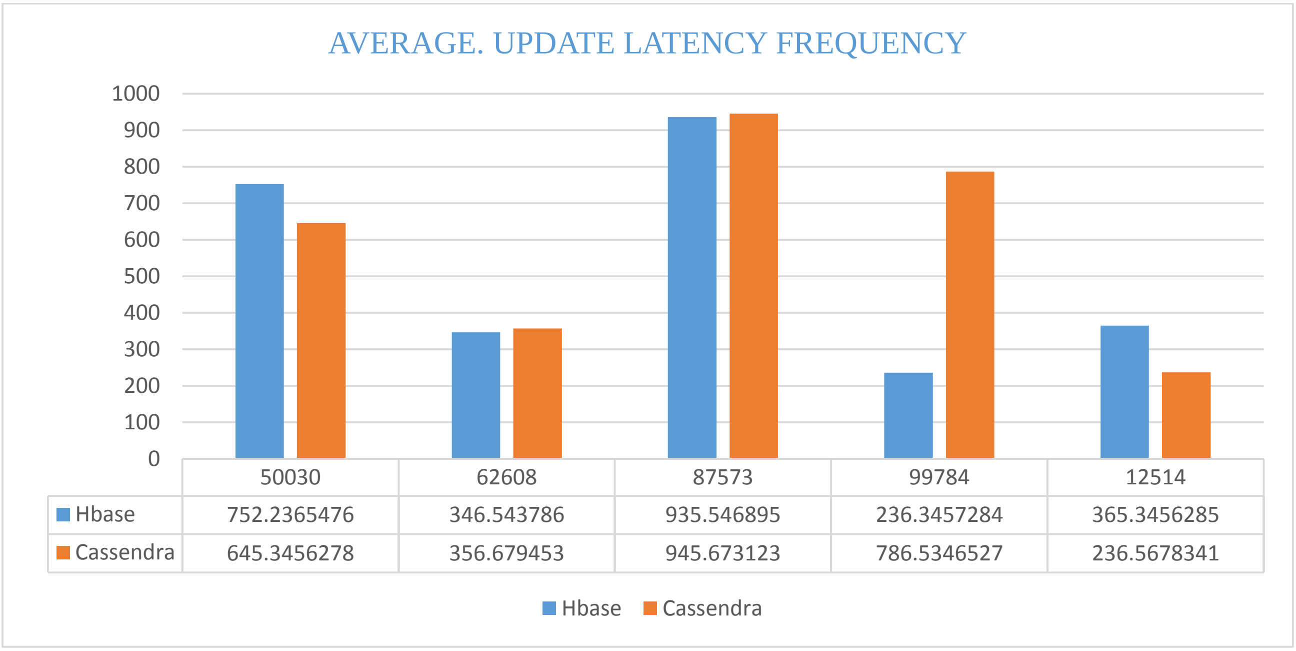


	49980	62396	87418	199243	128488
■ Hbase	253.3495068	266.8907458	257.3457869	263.1916266	265.1897667
■ Cassendra	426.9909374	418.3457689	430.4583908	428.7324858	430.4662125

Figure 5 No. of Read Operations Against Average Read Latency

In the above given diagram data performance and evaluation and its representation of variations in workload A. The graph in Figure 5 is plotted between the operations' start and end average read latency. The test is executed in five different operation counts in HBase and Cassandra. It can be observed that HBase but Cassandra has less latency in read operations and the latency stayed almost the same except for both operations of more than 120000. and the latency tends to decrease when it occurs in the middle and increases in reads.

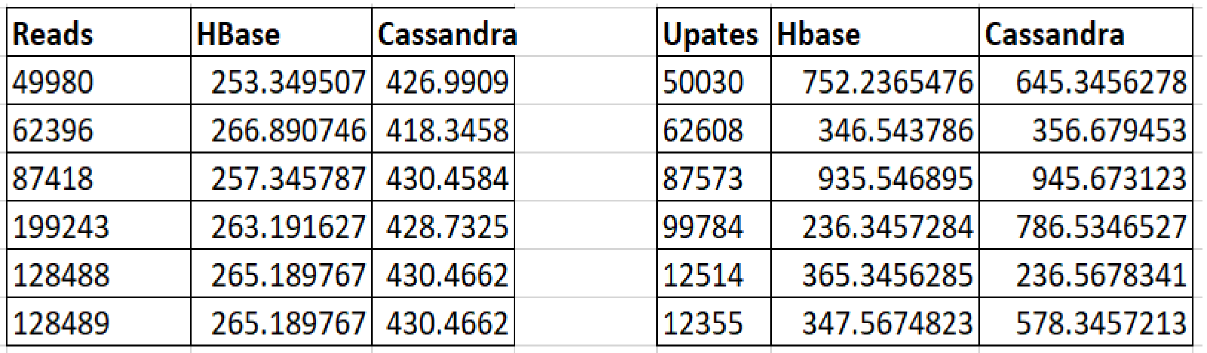
8.1.1 Average Update Latency Operations



	50030	62608	87573	99784	12514
🟦 Hbase	752.2365476	346.543786	935.546895	236.3457284	365.3456285
🟧 Cassandra	645.3456278	356.679453	945.673123	786.5346527	236.5678341

Figure 6 No. of Update operations against average Update latency

Figure 6 shows that from the number of given operational counts units, the Update latency in Cassandra increased with the no of inputs. of updates performed evaluation results. But HBase has fluctuating latency over different No. of updates per unit of operations.

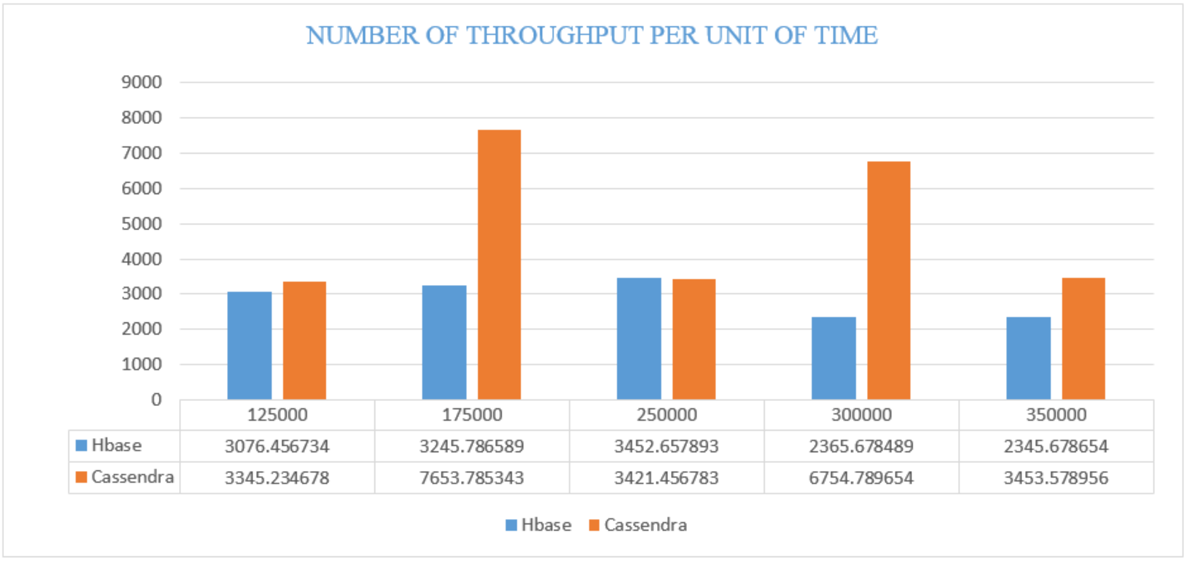


Reads	HBase	Cassandra
49980	253.349507	426.9909
62396	266.890746	418.3458
87418	257.345787	430.4584
199243	263.191627	428.7325
128488	265.189767	430.4662
128489	265.189767	430.4662

Upates	Hbase	Cassandra
50030	752.2365476	645.3456278
62608	346.543786	356.679453
87573	935.546895	945.673123
99784	236.3457284	786.5346527
12514	365.3456285	236.5678341
12355	347.5674823	578.3457213

Table 1 Average Update and Read Latency in HBase and Cassandra

Above all given data in both tables represents the no. of Read and Update operations per unit of time performed and overall average recorded, average latency HBase, and Cassandra operations.
As YCSB presented different metrics for performance evaluation and the number of inputs, the throughput performance is also considered per unit of time.



	125000	175000	250000	300000	350000
■ Hbase	3076.456734	3245.786589	3452.657893	2365.678489	2345.678654
■ Cassendra	3345.234678	7653.785343	3421.456783	6754.789654	3453.578956

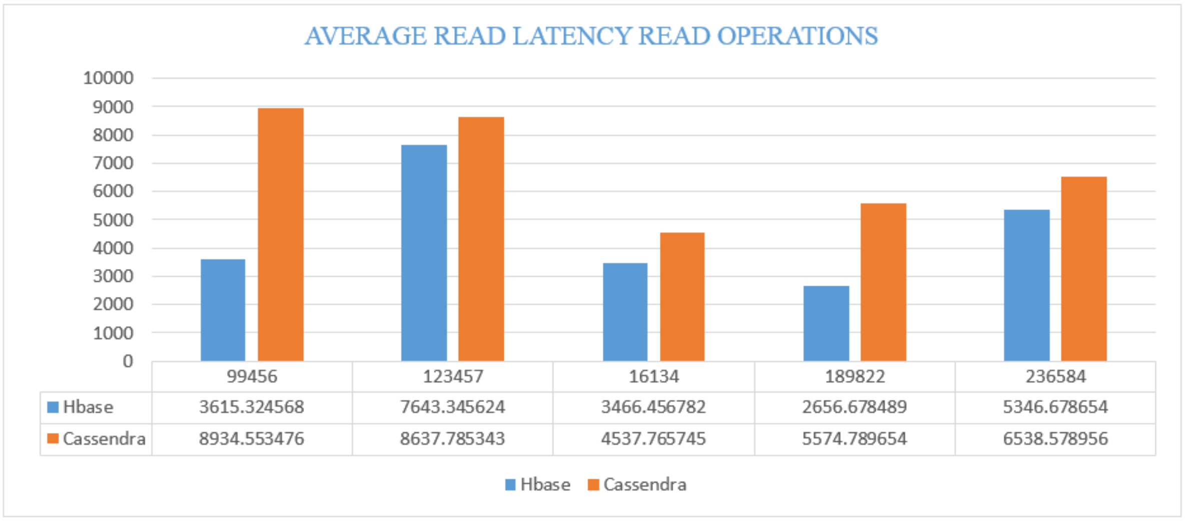
Figure 7 Throughput over No. of Operations in HBase and Cassandra

HBase and Cassandra are presented in Figure 7. It is observed that Cassandra has a rise over the period of time in throughput with an increase in different workloads. whereas HBase showed equivalent throughput but higher throughput is recorded per number of inputs at the highest operational count i.e. 350000.

8.2. Workload D in Lead Latency Operations

Workload D offered in YCSB is a read latest workload which has 95 percentiles of read different types of operations and 10 percentile of insert operations. These operations are executed in several iterations similar to workload A.

8.2.1 Average Read Latency Against No. of Reading Operations:



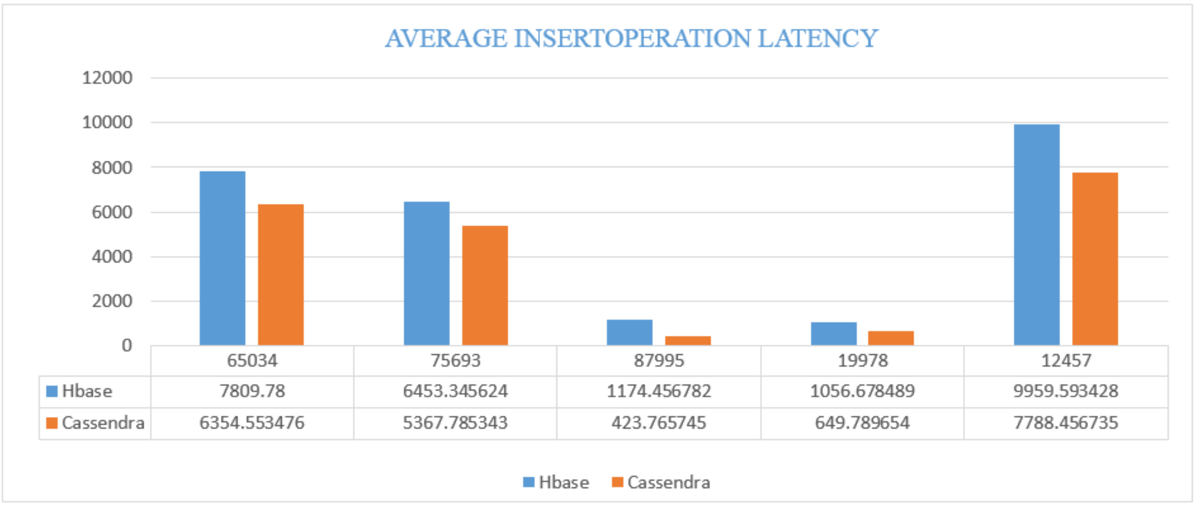
	99456	123457	16134	189822	236584
■ Hbase	3615.324568	7643.345624	3466.456782	2656.678489	5346.678654
■ Cassendra	8934.553476	8637.785343	4537.765745	5574.789654	6538.578956

Figure 8 Average Read Latency Over No of Read

Figure 8 shows the differences in average latency in read against the number of read operations in the database. It can be seen that Cassandra has higher average read latency than HBase but all the operation counts are relatively equivalent.

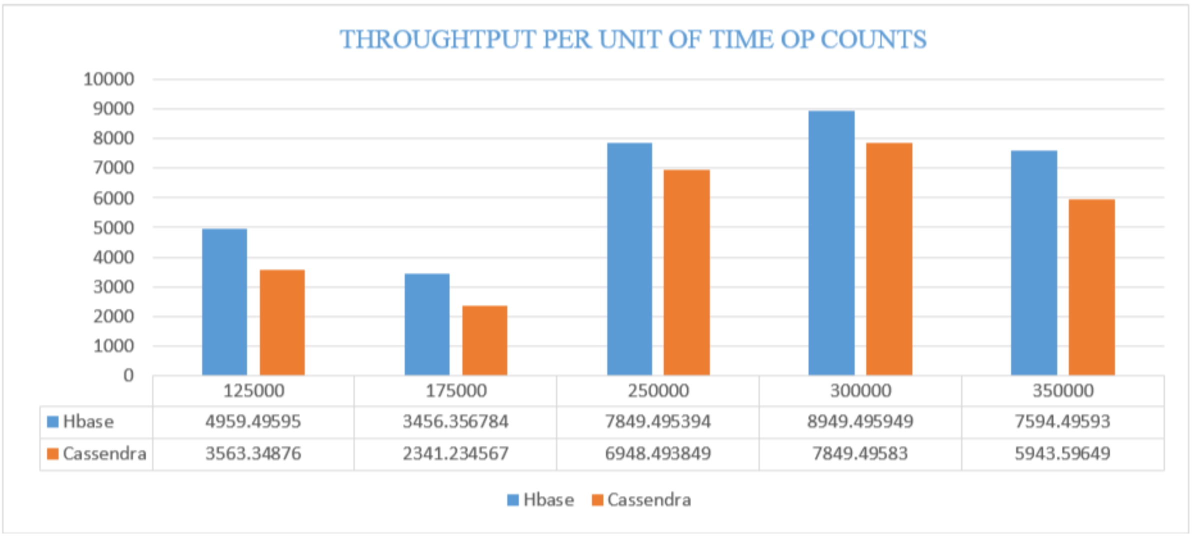
8.2.2 Average Insert Latency Against No of Insert Operations:

It can be seen in Figure 9 that the average insert operations in Cassandra are equivalent to all the different counts of inserts but HBase showed significant peaks in workload D at 8866 and 10078 inserts. As workload D is a read latest workload, fewer inserts are observed. Higher insert operations are performed with a higher number of operational counts. The graph in Figure 10 represents the throughput of each operation for a different number of operations. From the recorded observations, it can be seen that Cassandra has a slight increase in throughput with an increase in operational counts. On HBase, fluctuations in throughput are observed. at the beginning of the workload, the throughput was high and the highest was recorded at 250000 op counts.



	65034	75693	87995	19978	12457
■ Hbase	7809.78	6453.345624	1174.456782	1056.678489	9959.593428
■ Cassendra	6354.553476	5367.785343	423.765745	649.789654	7788.456735

Figure 9 Average Insert Latency Over Insert Operations



	125000	175000	250000	300000	350000
■ Hbase	4959.49595	3456.356784	7849.495394	8949.495949	7594.49593
■ Cassendra	3563.34876	2341.234567	6948.493849	7849.49583	5943.59649

Figure 10 Throughput of Databases Over OP Accounts

Below Table 2 represents over all the data which is recorded in workload D. As the workload is read latest both read latency per unit of time operations and the insert latency per unit of time operations are considered for both HBase and Cassandra databases. Throughput is also given to determine the best performing database in both databases in below table.

Read Latency			Thtoughput			Insert Latency		
Read	**Hbase**	**Cassandra**	**Opcounts**	**Hbase**	**Cassandra**	**Insert**	**Hbase**	**Cassandra**
94967	214.0992	378.92822	100000	3908.235	2388.402	5033	609.70256	444.6485
118789	207.7972	375.27442	125000	3558.719	2453.097	6211	564.40444	423.3444
166134	216.6994	374.87698	175000	3564.3	2505.046	8866	1172.7045	412.9791
189922	210.0735	356.77994	200000	3789.673	2640.473	10078	1028.5734	400.2724
237636	202.8797	353.58521	250000	4350.474	2688.374	12364	577.89882	390.1287

Table 2 Workload D Data

IX. CONCLUSION

HBase and Cassandra NoSQL databases are gaining attention with the rise over the period of time in distributed computing in a virtual cloud environment, guided media, unguided media as well as digital media, etc. Unlike the conventional and traditional DBMS, NoSQL is non-relational and they offer more functionalities to store different kinds of data in distributed fission. From the performance evaluation conducted over HBase and Cassandra in YCSB, several variations in the database behavior have been observed and presented in the observation and evaluations. HBase being backed up with the Hadoop Distributed files system has a relatively huge ecosystem as compared to Cassandra. Both Hbase and Cassandra databases have high-performance exceptions at higher workloads from A to D. At workload A, Cassandra has changeable latency throughput during read operations but HBase presented significantly less latency during reads latency. with the rest of the operations as Insert, updates, and HBase showed different results with the number of inputs the final result is fluctuating final records but Cassandra has increased read and update latency. Runtime and per unit of time for each operation can also be considered for both databases but to know the performance of every virtual machine which is running both Hbase and Cassandra databases some other metrics had to be considered as per the requirement. however, from the observations, and evaluation, Cassandra has higher runtimes than HBase. It can be assumed that it is because of the Hadoop map-reduce framework. In this research paper YCSB helped in both evaluation of performance with different metrics as per the requirement, multiple databases cannot be evaluated the performance in a single execution of the tools. However, features such as high scalability availability reliability, safety and security and the fast writes speeds of Cassandra can help when executing higher workload as observed. Our Research Paper Represents The properties and characteristics of HBase such as SQL type execution and run-time properties shared with Big Table in Figure 2. Which can make the database suitable for writing heavy applications in a virtual cloud environment. And significant performance increase can be observed on a native deployment as the network latency issues can be mitigated. Though YCSB is successful and complete in presenting overall metrics, new metrics can have included and tested in the given diagrams. There is a scope in this research paper which towards forming some new kinds of metrics and simplified the performance overall evaluation tools.

REFERENCES

[1] Ahmed, M. Razu, M. Arifa Khatun, A. Ali, and Kenneth Sundaraj. "A literature review on NoSQL database for big data processing." Int. J. Eng. Technol 7, no. 2 (2018): 902-906. “Apache HBase Architecture Overview.” https://hbase.apache.org/book.html Accessed on: 04-08-2021.

[2] Jakkula, Prashanth. "HBase or Cassandra? A Comparative Study of NoSQL Database Performance." International Journal of Scientific and Research Publications 10, no. 3 (2020): 808-820. “Apache Tephra.” https://tephra.apache.org/ Accessed on: 04-08-2023.

[3] Jakkula, Prashanth. "HBase or Cassandra? A Comparative Study of NoSQL Database Performance." International Journal of Scientific and Research Publications 10, no. 3 (2020): 808-820.

[4] Khattab, AbdAlhamid, Alsayed Algergawy, and Amany Sarhan. "MAG: A performance evaluation framework for database systems." Knowledge-Based Systems 85 (2015): 245-255.

[5] Kabakus, Abdullah Talha, and Resul Kara. "A performance evaluation of in-memory databases." Journal of King Saud University-Computer and Information Sciences 29, no. 4 (2017): 520-525.

[6] Jogi, Vishal Dilipbhai, and Ashay Sinha. "Performance evaluation of MySQL, Cassandra and HBase for heavy write operation." In 2016 3rd International Conference on Recent Advances in Information Technology (RAIT), pp. 586-590. IEEE, 2019.

[7] Hajjaji, Yosra, and Imed Riadh Farah. "Performance investigation of selected NoSQL databases for massive remote sensing image data storage." In 2018 4th international conference on advanced technologies for signal and image processing (ATSIP), pp. 1-6. IEEE, 2018.

[8] Abu Kausar, M., M. Nasar, and A. Soosaimanickam. "A Study of Performance and Comparison of NoSQL Databases: MongoDB, Cassandra, and Redis Using YCSB." Indian J. Sci. Technol 15 (2022): 1532-1540.

[9] Yangui, Rania, Ahlem Nabli, and Faiez Gargouri. "Automatic transformation of data warehouse schema to NoSQL data base: comparative study." Procedia Computer Science 96 (2016): 255-264.

[10] Seghier, Nadia Ben, and Okba Kazar. "Performance benchmarking and comparison of NoSQL databases: Redis vs mongodb vs Cassandra using YCSB tool." In 2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI), pp. 1-6. IEEE, 2021.

[11] Matallah, Houcine, Ghalem Belalem, and Karim Bouamrane. "Comparative study between the MySQL relational database and the MongoDB NoSQL database." International Journal of Software Science and Computational Intelligence (IJSSCI) 13, no. 3 (2021): 38-63.

[12] Pandey, Rachit. Performance benchmarking and comparison of cloud-based databases MongoDB (NoSQL) vs MySQL (Relational) using YCSB. Technical Report. https://doi. org/10.13140/RG. 2.2. 10789.32484, 2020.

[13] Singh, Pradeep Kumar, Yashwant Singh, Maheshkumar H. Kolekar, Arpan Kumar Kar, Jitender Kumar Chhabra, and Abhijit Sena, eds. Recent Innovations in Computing: Proceedings of ICRIC 2020. Springer, 2021.

[14] Kaur, Rupali, and Jaspreet Kaur Sahiwal. "A review of comparison between NoSQL databases: MongoDB and couchDB." International Journal of Recent Technology and Engineering 7 (2019): 892-898.

[15] Nasrullah and Ms. Akanksha Bana,” hadoop configuration implementation and deployment model in virtual cloud environment” international journal of Research and analytical reviews (ijrar.org) Volume.6, Issue 1, Page No pp.983-989, February-2019

[16] Seghier, Nadia Ben, and Okba Kazar. "Performance benchmarking and comparison of NoSQL databases: Redis vs Mongodb vs Cassandra using YCSB tool." In 2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI), pp. 1-6. IEEE, 2021.

[17] Nasrullah and Ms. Akanksha Bana, "Review paper on Hadoop configuration and implementation in virtual cloud environment", Vol.5, Issue 10, page no.530-535, October-2018