iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 Issue: IV Month of publication: April 2024
DOI: https://doi.org/10.22214/ijraset.2024.60920

www.ijraset.com
Call: 08813907089 | E-mail ID: ijraset@gmail.com

A Performance Analysis of CassandraDB and ScyllaDB

Sushma Kumari[1], Mrs. Anita Ganpati[2]
Department of Computer Science, Himachal Pradesh University, India

Abstract: Big Data encompasses vast amounts of data, reaching exabytes or even zettabytes. In the current landscape, large databases play a crucial role, particularly in generating substantial data for daily analyses of social media and multimedia content. The enormity of Big Data poses challenges, given its extensive volume of structured, semi-structured, and unstructured data, making traditional database systems and software techniques insufficient. Big Data is frequently defined by its 9 V's: velocity, variety, volume, veracity, validity, variability, volatility, visualization, and value. This complexity highlights the need for a simple information management strategy that integrates various new data types alongside traditional data. The significance of Big Data databases is emphasized by the daily generation of millions of terabytes of data from sources like social media posts and multimedia. This study aims to evaluate the performance of two Wide-Column Store Big Data database systems, Apache CassandraDB and ScyllaDB, using the CassandraStress Benchmarking Tool. Key metrics such as total operation time, operation rate, partition rate, row rate, and maximum latency will be assessed as the number of records and operations increase. The development of these databases, motivated by diverse industry requirements, emphasizes their adaptation to specific needs. The research methodology outlines the tool used to compare these WideColumn Store databases on various parameters, contributing valuable insights into their performance in real-world scenarios.

Keywords: Big Data Database, ACID, BASE, CAP, NoSQL Databases.

I. INTRODUCTION

Data management has been recast by the advent of Big Data [30], which has pushed the boundaries of conventional database systems. Traditional relational databases, characterized by SQL-based query languages, struggle to handle the vast and unstructured datasets generated by modern applications, IoT devices, social media, and more. This gave rise to the boost of NoSQL databases, which offer a scalable and flexible solution for processing and managing large datasets.

Big Data technology is a big deal in many areas [20]. It helps handle lots of complicated information and can be used to make predictions and improve how things work. It has changed the way we do things and has a lot of potential for different organizations. For example, it has transformed the healthcare industry by enabling predictive analytics, disease surveillance, and precision medicine. It also helps businesses optimize their consumption of energy, guide citizens to more sustainable means of transportation, and improve efficiency in various sectors. Overall Big Data has emerged as a game-changer in the digital age, transforming industries and the way we live, work, and make decisions.[26].

A. Evolution of Big Data Databases

Historically, databases evolved from flat files to network models and hierarchical before settling on the relational model. However, the curb of relational databases became evident as data sizes exploded in the digital age. The emergence of Big Data databases marked a paradigm shift, with distributed processing frameworks and storage like Apache Hadoop gaining prominence. These systems have incorporated scalability and fault tolerance, which have made it possible to process large and complex datasets efficiently.

B. Characteristics of Big Data Databases

Big Data databases exhibit distinct characteristics that set them apart from their traditional counterparts. Horizontal scalability, flexibility in handling diverse data types, and the ability to operate in a distributed environment are key features. Additionally, these databases often prioritize performance and fault tolerance, allowing Organizations to maximize the value of their data assets by effectively utilizing them.

1) Big Data CPIVW framework categorizes characteristics into five categories

The various characteristics of big data are related to each other and can be grouped into different categories accordingly. We have identified five categories - Collecting Data, Processing Data, Integrity Data, Visualization Data, and Worth of Data - by clustering the characteristics of big data into groups based on their relationships. These categories have been extracted from the nine characteristics of big data. [17, 25].

Figure 1 demonstrates the Big Data CPIVW framework categories and characteristics.

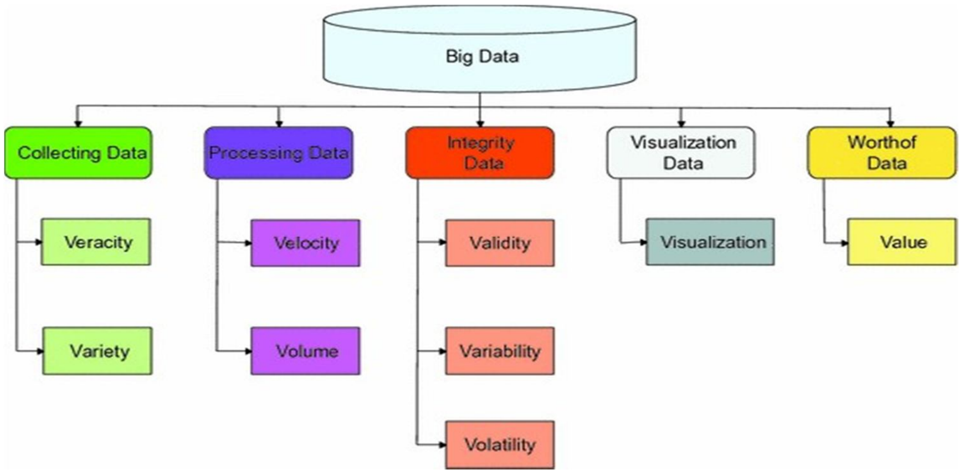


Fig.1 Five Big Data CPIVW framework categories with their 9 V's Characteristics
Source: http://surl.li/ozona

The following discussion outlines the five categories of CPIVW: collecting data, processing data, maintaining data integrity, visualizing data, and determining the worth of data.

a) Collection of Data(C)

Data is gathered from various sources in different forms, such as structured, unstructured, and semi-structured data, to create Big Data. Due to the diverse characteristics and varied accuracy of this data, it is grouped under the category of collected data [25].

b) Processing of Data (P)

The term "Processing Data" refers to the combination of two main characteristics of Big Data, namely velocity and volume. These two characteristics are closely related to each other. The processing data category is concerned with the speed of processing the data in relation to the size of Big Data. The data that falls under the processing data category goes through various processes and is processed on demand [25].

c) Integrity of Data(I)

The accuracy and consistency of data stored in Big Data is referred to as data integrity. It is categorized into three Vs: validity, variability, and volatility. Data integrity ensures that the stored information is truthful, correct, and has not been manipulated. It also guarantees the quality of data in Big Data[25].

d) Visualization of Data(V)

Data visualization is a technique that helps to explore and comprehend data in a way that is similar to how the human brain processes information. The Visualization Data category includes only the visual aspect of Big Data, making it easier to read and analyze from complex graphs. When data is visualized in Big Data, it helps users understand the meaning of different data values more quickly when they are displayed in charts and graphs rather than reading about them in reports [25].

e) Worth of Data(W)

The value of a data category in Big Data is determined by its characteristic value. This value signifies the cost and management of data in Big Data.

Small data can have a higher value than a corresponding Big Data collection, but the latter has a higher economic value. During the data collection process, there may be some noise that needs to be filtered out to ensure accurate analysis and decision-making by the user [25].
Volume is a crucial component of Big Data. It can be said that without volume, Big Data would not be considered as big. In fact, it would be a small data set that could easily be managed using traditional storage and processing systems.

"All V's of BigData -Volume != Big Data"

Nevertheless, numerous characteristics exist within the Big Data paradigm[26].
The primary aim of collecting data is not merely due to the ability to capture large amounts of data quickly and from various sources. Instead, the collection of data is aimed at finding solutions for research or business problems. The goal is to find actionable intelligence that can assist decision-makers. Pure data-driven analysis may not add much value to the decision-making process. [31].
The management of data has evolved over the years, starting with files in the 1960s and moving on to databases. In the 1980s, relational databases came into existence, followed by object-oriented databases in the 1990s. Since the 2010s, a new category of databases called non-relational databases or NoSQL (Not only SQL) databases have become increasingly popular [24].

II. NOSQL (NOT ONLY SQL) DATABASE

A. History

NoSQL [23] stands for "Not Only SQL" and is pronounced as "**no-sequel**". Carlo Strozzi first used the term "NoSQL" in 1998 as the name of the file he was developing for his database. NoSQL is a type of data storage that differs from traditional databases that were used in the past [14]. Strozzi argues that the current NoSQL movement departs from the relational model entirely, so it would have been more appropriate to call it "NoREL". However, Eric Evans (who was then working at Rackspace) reintroduced the term NoSQL in early 2009. This was when Johan Oskarsson of Last.fm wanted to organize an event to discuss open-source distributed databases. The name NoSQL was intended to label the emergence of a growing number of non-relational, distributed data stores that often did not aim to provide atomicity, consistency, isolation, and durability guarantees that are fundamental attributes of traditional relational database systems [16]. Computer programmers classify database management systems based on the type of DBMS they support.
Relational databases, originally described in the late 1970s, are collections of tables containing rows and columns. Many of them use SQL to input and retrieve information [18].

B. Advantages and Disadvantages of Relational Database

The Benefits and Drawbacks of relational databases are illustrated in Figure 2.

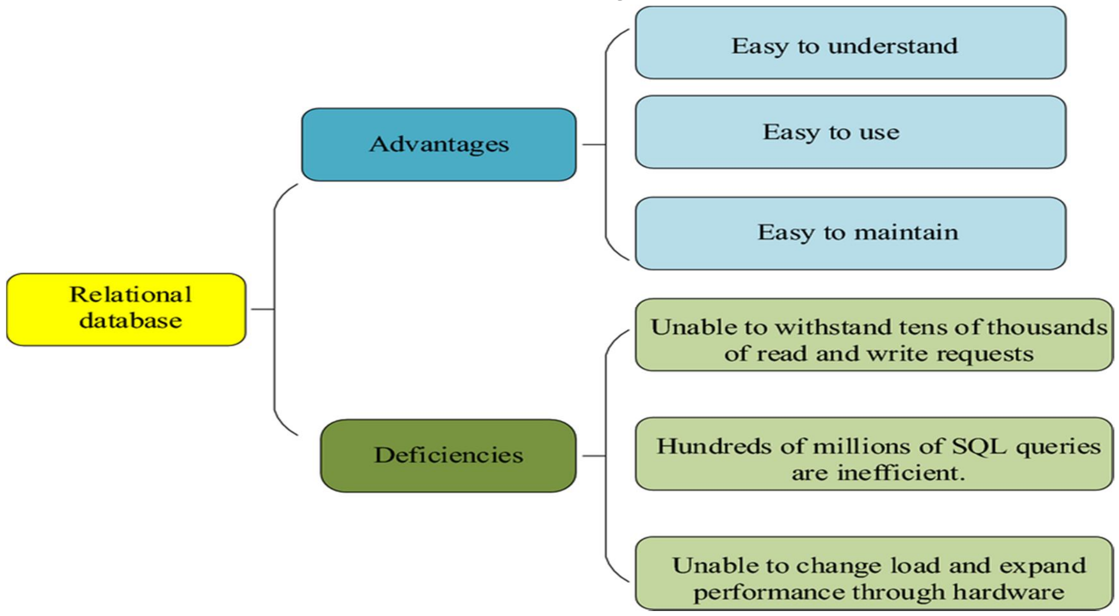

Relational database
Advantages
Easy to understand
Easy to use
Easy to maintain
Deficiencies
Unable to withstand tens of thousands of read and write requests
Hundreds of millions of SQL queries are inefficient.
Unable to change load and expand performance through hardware

Fig.2 Benefits and Drawbacks of Relational-Database
Source: https://rb.gy/5uhmbh

In RDBs, merging large amounts of data from multiple tables can be slow and cause businesses to lose their competitive edge. Switching to a NoSQL database can help enterprises remain competitive by better handling large amounts of data [12].
Non-relational databases, known as NoSQL, became popular in the late 2000s [9], due to their unique query language and capability to handle large-scale, unstructured data efficiently. In today's world of constantly-evolving Big Data Processing, NoSQL databases have established a significant role and are used to power a wide range of applications, from social media platforms like Facebook to expansive online games [6].
It is commonly accepted that BigTable,[10] developed by Google, was the first NoSQL database [21].
NoSQL was not a popular term until a meeting held in San Francisco in 2009. Since then, it has become a buzzword [5].

C. ACID and BASE

Structured databases use normalization techniques to make sure data stays accurate and avoids unnecessary repetition. This process involves organizing data to keep things straightforward and reduce the risk of errors. By following the ACID principles [19] (Atomicity, Consistency, Isolation, Durability), relational databases ensure that transactions happen reliably and securely.
Instead of using ACID semantics, No-SQL databases utilize BASE semantics. These semantics are described below:

1) Basically Available (BA)

This means that an application is always ready to accept read/write requests..

2) Soft state (S)

The results obtained from an application may not always be based on consistent data, and there is no guarantee of consistency.

3) Eventually consistency (E)

The system assures that data will eventually become consistent at some later point, although there is no guarantee of when that will happen [28].

D. Features of NoSQL DB are below:

Schema-less Structure
Horizontal Scalable
High Performance
Distributed Architecture Automatic Sharding

E. CAP Theory [11]

1) Consistency

Consistency implies that every client should observe identical data simultaneously, regardless of their connected node. Achieving this involves promptly forwarding or replicating any data written to a single node to all other nodes in the system before considering the write operation as successful.

2) Availability

Availability means that any client requesting data should receive a response, even if one or more nodes are down. In other words, all working nodes in the distributed system should return a valid response for any request, without any exception.

3) Partition tolerance

A partition is a communication break within a distributed system, such as a lost or temporarily delayed connection between two nodes. Partition tolerance means that the cluster must continue to work despite any number of communication breakdowns between nodes in the system.

F. Overview of NOSQLDB

1) Key-Value Stores

Simple and Fast, Like a Digital Dictionary. Imagine a giant digital dictionary where you can instantly retrieve anything using a unique key.

That is the essence of a key-value store. It excels in fast data access and simple operations, making it perfect for session management, caching, and real-time applications. Redis and Memcached are popular examples.

2) Document Databases

Flexible and Familiar, Like JSON on Steroids. Think of documents you store on your computer but with superpowers. Document databases store data in JSON-like structures, allowing for flexible schema and complex relationships. They're ideal for storing user profiles, product catalogs, and content management systems. MongoDB and Couchbase are prominent players in this arena.

3) Column-Oriented Databases

Wide and Scalable, Like Super Spreadsheets. These databases break down data into columns, enabling efficient storage and retrieval of specific data slices. Imagine a massive spreadsheet where you can analyze specific columns without loading the entire thing. This makes them perfect for analytics, logs, and time-series data. Cassandra, ScyllaDB and HBase are giants in this domain.

4) Graph Databases

Connecting the Dots, Like a Social Network for Data. Ever wondered how social media platforms recommend friends or suggest related products? Graph databases hold the key. They store data as nodes (entities) and relationships (edges) between them, revealing hidden connections and patterns. Neo4j and OrientDB are leading the graph revolution.

5) Time Series Databases

Capturing the flow like a Movie Reel for Data.Imagine storing every frame of a movie, not just the final product. Time series databases do just that, capturing data points at specific timestamps. This makes them ideal for sensor data, financial transactions, and IoT applications. InfluxDB and TimescaleDB are masters of this temporal dance.

III. WIDE-CULUMN DATABASES- CASSANDRADB AND SCYLLADB

A. CassandraDB

Apache Cassandra[29] is a cool and free-to-use distributed NoSQL database system that's awesome at handling massive amounts of data across a bunch of regular servers. It was initially developed at Facebook and later made opensource in 2008. The Apache Software Foundation got in on the action in 2009, taking over the development and turning it into this superstar database that people love. Now, lots of big-scale applications that need to grab loads of data real quick are jumping on the Cassandra train. It's like the rockstar of databases!

Cassandra is like a tech marvel! Its setup is inspired by Dynamo, a replication model that's super smart and doesn't have any weak spots, so it's super reliable. It organizes data in columns, just like Google's Big Table, and spreads it out across lots of computers. This way, even if one computer takes a break, everything keeps running smoothly. This design makes it lightning-fast to grab or add data, which is perfect for things like quick analytics and massive internet apps. It's like having a speedy superhero to handle all your data needs!

The history of Cassandra can be traced back to 2008 when it originated at Facebook [7] and became open-source later that year. The Apache Software Foundation took over its development in 2009. Since then, it has become one of the most popular NoSQL databases, used by companies like Amazon, Netflix, Spotify, and Comcast. eBay was one of the early adopters of CassandraDB.

Cassandra is a popular choice for high-performance and scalable applications that require fast access to large data sets. Cassandra stores data in a column-oriented approach called a wide-column store [15], which allows for fast read and write operations. The database is designed to be distributed, storing data evenly across a cluster of nodes to ensure reliability and fault tolerance. CQL shell is a command-line shell for interacting with CassandraDB through CQL [8].

B. ScyllaDB

ScyllaDB[27] is like an upgraded edition of Apache Cassandra [22]. It is a distributed NoSQL database meant to integrate smoothly with Cassandra while providing a substantial performance boost [1]. The journey began in 2014 when a startup named Cloudius Systems, later rebranded as ScyllaDB Inc., brought it to life. They generously shared it with the world in September 2015, following the AGPL (Affero General Public License) license.

What sets ScyllaDB apart in terms of speed is its smart shard-per-core setup, running on the highly asynchronous Seastar. All the shards collaborate simultaneously, providing a substantial performance enhancement.

It aligns well with Cassandra’s protocols (CQL and Thrift) and file formats (SSTable). Whether you prefer it on-premises, on major cloud platforms, or as a service (ScyllaDB Cloud), it’s all set to deliver top-notch performance. ScyllaDB is a disk-oriented DBMS, that stores data in SSTable[3] files. It also supports inmemory tables, which reduces read latency for mostly read workloads. Scylla uses a shared-nothing model, with nodes in the cluster organized in a decentralized consistent hashing ring. The database uses a shard-per-core architecture, where each thread for a shard executes on its CPU core. ScyllaDB is designed to run on Linux and is written in C++. It utilizes a modern NoSQL database architecture that is specifically designed for high performance and availability, while also providing API compatibility with Cassandra. [2].

IV. RESULTS AND ANALYSIS

In this section, experiments were conducted on a comparative analysis of two prominent NoSQL databases, namely CassandraDB and ScyllaDB, both widely adopted by multinational corporations in contemporary enterprise environments. The objective is to discern the optimal use cases for each database and provide insights into their respective strengths and weaknesses. The evaluation employs the Cassandra-Stress Benchmarking Tool [13] to facilitate a comprehensive and rigorous assessment of performance metrics. The parameters considered in the study are operation rate, partition rate, row rate, latency rate, and total operation time with an increase of records (which are 50000,100000,150000,175000).

These are the configurations that were used during the experiment: A client is running 19 threads and a small database cluster with limited workloads. The hardware specifications for the experiments include a client machine with an Intel processor (8 logical processors, 4 cores, 8 GB of RAM, and 1.60 GHZ). Additionally, Docker version 24.0.6 was employed to provide an isolated and reproducible environment for benchmarking ScyllaDB. Docker [4] simplifies application creation, deployment, and management using containers.

TABLE I

TABLE I: PERFORMANCE METRICS FOR READ

DATABASE	CASSANDRADB					SCYLLADB				
RECORD	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time
50000	7900	7900	7900	75.7	00:00:06	1011	1011	1011	702.5	00:00:49
100000	8414	8414	8414	133.3	00:00:11	1016	1016	1016	2095.1	00:01:38
150000	8860	8860	8860	129.6	00:00:16	836	836	836	7264.5	00:02:59
175000	9384	9384	9384	157.4	00:00:18	2752	2752	2752	717.2	00:01:03

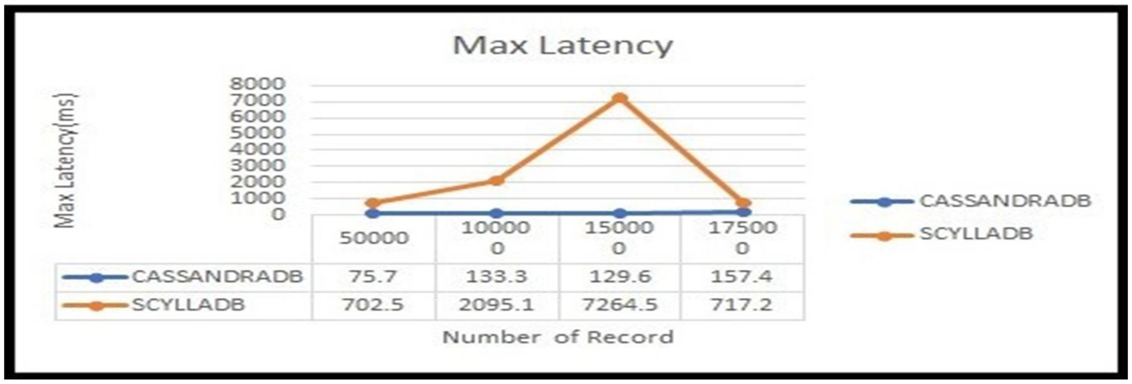


Fig.3 Maximum Latency rate for read

TABLE II

TABLE II: PERFORMANCE METRICS FOR WRITE

DATABASE	CASSANDRADB					SCYLLADB				
RECORD	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time
50000	9094	9094	9094	126.7	00:00:05	6140	6140	6140	50.7	00:00:08
100000	10592	10592	10592	111.9	00:00:09	5503	5503	5503	179.2	00:00:18
150000	10369	10369	10369	87.4	00:00:14	3479	3479	3479	4676.6	00:00:43
175000	10481	10481	10481	125.2	00:00:16	4080	4080	4080	237.1	00:00:42

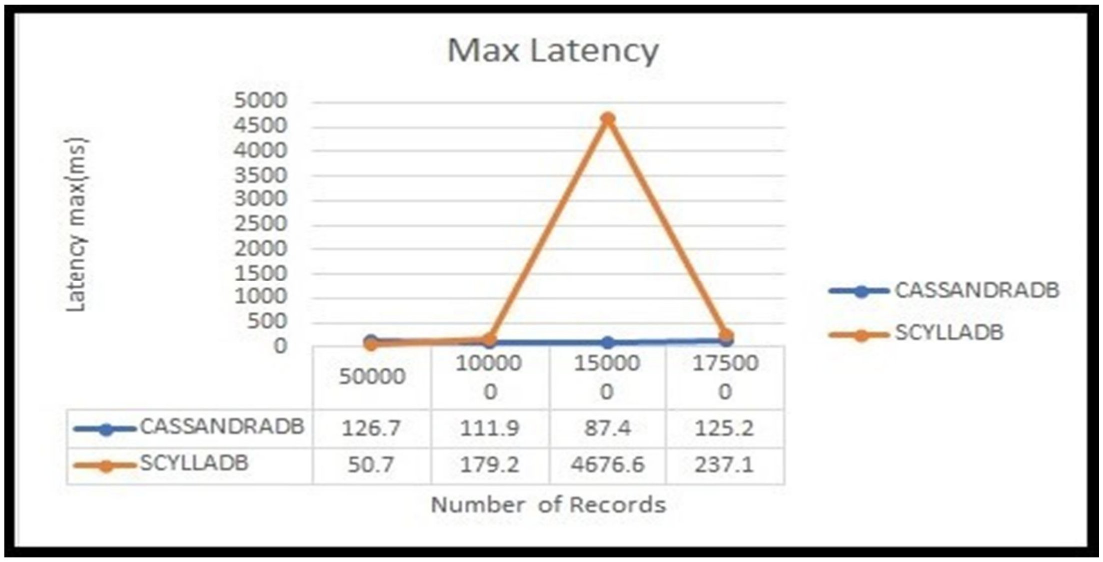


Fig.4 Maximum Latency rate for write

The mixed experiments aimed to compare Cassandra and ScyllaDB's performance in various configurations. The stress-testing tool "Cassandra-stress" was used to conduct all the experiments at the client end. The client executed a 50:50 mix of read and write operations in the experimental setup.

TABLE III
TABLE III: PERFORMANCE METRICS FOR READS/RATES (50:50 MIX)

DATABASE	CASSANDRADB					SCYLLADB				
RECORD	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time
50000	8921	8921	8921	86.4	00:00:05	4320	4320	4320	109.3	00:00:11
100000	8149	8149	8149	131.2	00:00:12	4419	4419	4419	240.5	00:00:22
150000	8222	8222	8222	155.5	00:00:18	3852	3852	3852	373.0	00:00:38
175000	7925	7925	7925	127.5	00:00:22	3136	3136	3136	554.4	00:00:55

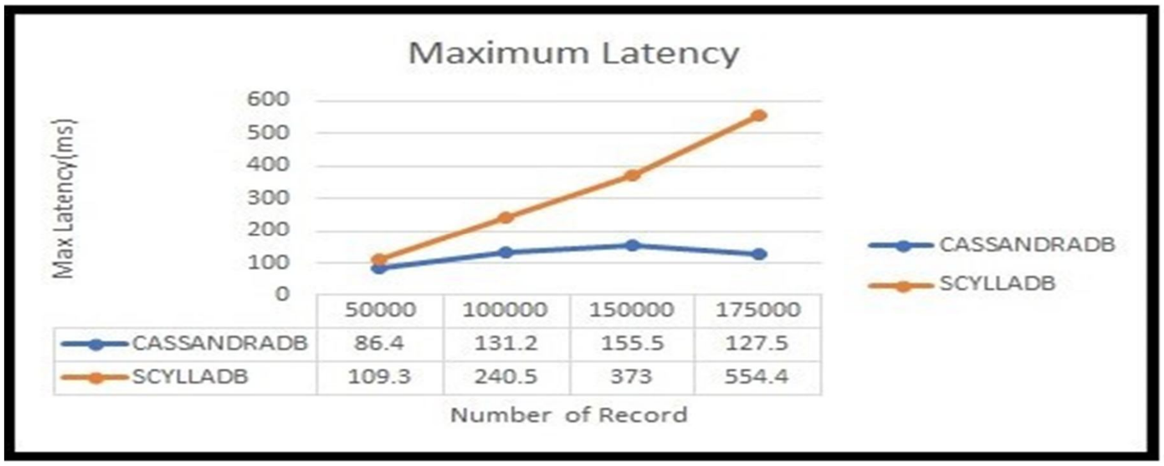
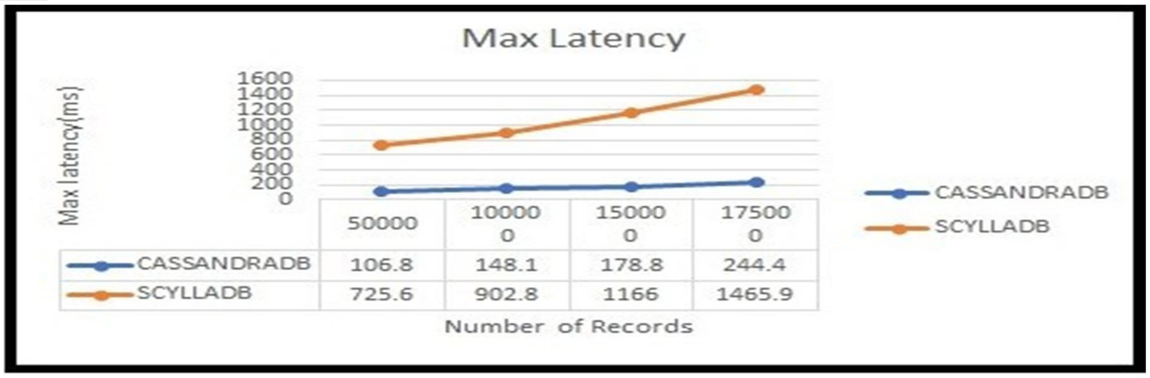


Fig.5 Maximum latency rate for r/w mixed

TABLE IV
TABLE IV: PERFORMANCE METRICS FOR COUNTER-WRITES

DATABASE	CASSANDRADB					SCYLLADB				
RECORD	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time	OP RATE (OP/Sec)	Partitioning rate (pk/sec)	Row rate (row/sec)	Latency max (ms)	Total operation time
50000	7684	7684	7684	106.8	00:00:06	384	384	384	725.6	00:02:10
100000	7728	7728	7728	148.1	00:00:12	513	513	513	902.8	00:03:15
150000	7875	7875	7875	178.8	00:00:19	798	798	798	1166.0	00:03:07
175000	6591	6591	6591	244.4	00:00:26	1518	1518	1518	1465.9	00:01:55



	50000	100000	150000	175000
CASSANDRADB	106.8	148.1	178.8	244.4
SCYLLADB	725.6	902.8	1166	1465.9

Fig.6 Maximum Latency rate for counter-writes

As indicated in Tables 2,3,4, and 5 the overall maximum latency of Cassandra DB is lower than that of ScyllaDB. Furthermore, the total operation time to complete the task is faster for Cassandra DB as the number of records increases. The row rate indicates the number of rows written or updated per second. For applications with frequent data updates, a higher row rate is desirable higher partition rate signifies the ability to handle more write operations across different data segments. For write-intensive workloads, a higher partition rate is crucial. CassandraDB generally boasts a higher partition rate compared to ScyllaDB.

As shown in Figures 3, 4, 5, and 6 ScyllaDB exhibits higher latency compared to CassandraDB. Lower latency is preferred, indicating quicker data travel and faster application response times. This is crucial for real-time applications, ensuring a smoother and more responsive user experience, such as in video conferencing and streaming. Lower latency enhances efficiency in task completion for applications and systems.

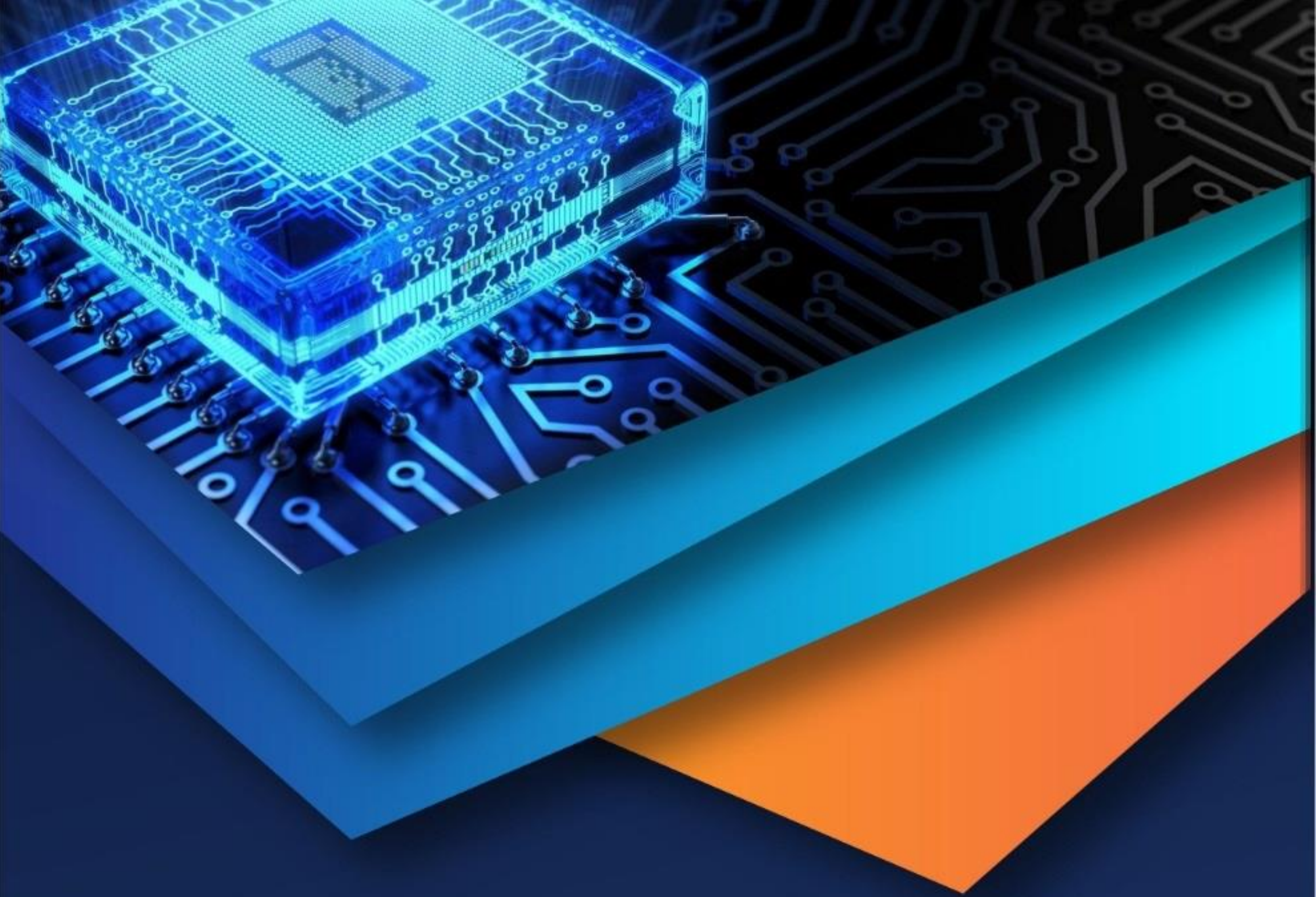
V. CONCLUSION AND FUTURE SCOPE

1) In this study, key parameters, including operation rate, partition rate, row rate, latency rate, and total operation time, were examined to compare the performance of CassandraDB and ScyllaDB. The results consistently favored CassandraDB, demonstrating its superiority over ScyllaDB across these metrics. CassandraDB performed faster operations, more efficient data division, quicker responses, and completed tasks in less time, demonstrating it is better for demanding workloads.
2) The study indicates that CassandraDB performs well in the parameters considered. In the future, ScyllaDB could excel when examining different measures and increased workloads. However, there are interesting areas for future research that can provide a better understanding:
3) Exploring extra measures: It would be beneficial to check databases in different situations, examining factors like the load on the computer, and how much memory is used. This will give us a better overall picture of how well they perform in different scenarios.
4) Examining delays: Examining delays, and understanding the reasons behind delays in each database is crucial. This insight can contribute to improving their performance by identifying ways to enhance efficiency.
5) Instead of just examining speed, it's crucial to compare various features and unique capabilities of each database. Evaluating how they handle problems, like computer malfunctions or internet disruptions, is essential. Especially for tasks where errors are not acceptable.

REFERENCES

[1] Cassandra and Scylladb: Similarities and differences. Accessed on January 10, 2024, at 6:11 PM.
[2] Cassandra vs Scylladb: A comparative analysis for your next database solution. Accessed on January 10, 2024 at 6:13 PM.
[3] Sstable. Accessed on January 10, 2024 at 6:09 PM.
[4] Download center - get started. https://www.scylladb.com/download/, June 05 2023. Accessed on January 8, 2024, at 4:49 PM.
[5] Veronika Abramova and Jorge Bernardino. Nosql databases: Mongodb vs cassandra. In Proceedings of the international C* conference on computer science and software engineering, pages 14–22, 2013.
[6] Adam AE Alflahi, Mohammed AY Mohammed, and Abdallah Alsammani. Enhancement of database access performance by improving data consistency in a non-relational database system (nosql). arXiv preprint arXiv:2308.13921, 2023.
[7] Musbah J Aqel, Aya Al-Sakran, and Mohammad Hunaity. A comparative study of nosql databases. Bioscience Biotechnology Research Communications, 12:17–26, 2019.

[8] Mal gorzata Bach and Aleksandra Werner. Standardization of nosql database languages. In Beyond Databases, Architectures, and Structures: 10th International Conference, BDAS 2014, Ustron, Poland, May 27-30, 2014. Proceedings 10, pages 50–60. Springer, 2014.
[9] Roman Cereˇsnˇ´ak and Michal Kvet.ˇ Comparison of query performance in relational a non-relation databases. Transportation Research Procedia, 40:170–177, 2019.
[10] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS), 26(2):1–26, 2008.
[11] Jyoti Chaudhary, Vaibhav Vyas, and CK Jha. Qualitative analysis of sql and nosql database with an emphasis on performance. In IOT with Smart Systems: Proceedings of ICTIS 2022, Volume 2, pages 155–165. Springer, 2022.
[12] Jeang-Kuo Chen and Wei-Zhe Lee. The transformation of rdb to nosql db. In 2021 International Conference on Technologies and Applications of Artificial Intelligence (TAAI), pages 184–187. IEEE, 2021.
[13] DataStax Enterprise. The cassandra-stress tool. https://docs.datastax. com/en/dse/5.1/docs/tooling/cassandra-stress-tool.html, 2023. Accessed on January 6, 2024 at 6:06 PM.
[14] M Dave. International journal of advanced research in sql and nosql databases. Int. J. Adv. Res. Comput. Sci. Softw. Eng. Res, 2(8), 2016.
[15] Andrea Gandini, Marco Gribaudo, William J Knottenbelt, Rasha Osman, and Pietro Piazzolla. Performance evaluation of nosql databases. In Computer Performance Engineering: 11th European Workshop, EPEW 2014, Florence, Italy, September 11-12, 2014. Proceedings 11, pages 16–29. Springer, 2014.
[16] S George. Nosql–not only sql. International Journal of Enterprise Computing and Business Systems, 2(2), 2013.
[17] Gayatri Kapil, Alka Agrawal, and RA Khan. A study of big data characteristics. In 2016 International Conference on Communication and Electronics Systems (ICCES), pages 1–4. IEEE, 2016.
[18] Muhammad Zohaib Khan, Fahim Uz Zaman, Muhammad Adnan, Aisha Imroz, Mahira Abdul Rauf, and Zuhaib Phul. Comparative case study: An evaluation of performance computation between sql and nosql database. Journal of Software Engineering, 1(2):14–23, 2023.
[19] David Lauzon. Introduction to big data. 2012.
[20] David Lazer and Jason Radford. Data ex machina: introduction to big data. Annual Review of Sociology, 43:19–39, 2017.
[21] Yishan Li and Sathiamoorthy Manoharan. A performance comparison of sql and nosql databases. In 2013 IEEE Pacific Rim conference on communications, computers and signal processing (PACRIM), pages 15–19. IEEE, 2013.
[22] Ashraf Mahgoub, Sachandhan Ganesh, Folker Meyer, Ananth Grama, and Somali Chaterji. Suitability of nosql systems—cassandra and scylladb—for iot workloads. In 2017 9th International Conference on Communication Systems and Networks (COMSNETS), pages 476–479. IEEE, 2017.
[23] Sourav Mukherjee. The battle between nosql databases and rdbms. Available at SSRN 3393986, 2019.
[24] Simona-Vasilica Oprea, Adela Bˆara, and Niculae Oprea. Big data management and nosql databases. Ovidius University Annals, Economic Sciences Series, 23(1):466–475, 2023.
[25] Suhail Sami Owais and Nada Sael Hussein. Extract five categories cpivw from the 9v's characteristics of the big data. International Journal of Advanced Computer Science and Applications, 7(3), 2016.
[26] Ripon Patgiri and Arif Ahmed. Big data: The v's of the game changer paradigm. In 2016 IEEE 18th international conference on high performance computing and communications; IEEE 14th international conference on smart city; IEEE 2nd international conference on data science and systems (HPCC/SmartCity/DSS), pages 17–24. IEEE, 2016.
[27] ScyllaDB. Welcome to scylladb documentation. https://docs.scylladb. com/stable/, 2024. Accessed on January 5, 2024 at 3:10 PM.
[28] Atul Thakare, Omprakash W Tembhurne, Abhijeet R Thakare, and Soora Narasimha Reddy. Nosql databases: Modern data systems for big data analytics-features, categorization and comparison. International journal of electrical and computer engineering systems, 14(2):207–216, 2023.
[29] The Apache Software Foundation. Welcome to apache cassandra's documentation! https://cassandra.apache.org/doc/4.1/index.html, 2024. Accessed on January 5, 2024 at 3:05 PM.
[30] Richard L Villars, Carl W Olofson, and Matthew Eastwood. Big data: What it is and why you should care. White paper, IDC, 14:1–14, 2011.
[31] Caesar Wu, Rajkumar Buyya, and Kotagiri Ramamohanarao. Big data analytics= machine learning+ cloud computing. arXiv preprint arXiv:1601.03115, 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)