Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

A consolidated study on transformation of Traditional Data to Big Data

Swara Joshi
Department of Information and Communication Technology, Pandit Deendayal Energy University, Gujarat
Email: swara.jict20@sot.pdpu.ac.in
Aakarsh Garg
Department of Information and Communication Technology, Pandit Deendayal Energy University, Gujarat
Email: aakarsh.gict20@sot.pdpu.ac.in
Madhu Shukla
Department of Computer Engineering - AI & BDA, Marwadi University, Gujarat
Email: madhu.shukla@marwadieducation.edu.in
Dhara Joshi
Department of Computer Engineering, Marwadi University, Gujarat
Email: dhara.joshi@marwadieducation.edu.in
Santushti Betgeri
Department of Computer Engineering, Marwadi University, Gujarat
Email: santushti.betgeri@marwadieducation.edu.in
Pooja Shah
Department of Computer Science Engineering, Pandit Deendayal Energy University, Gujarat
Email: dr.pooja.research@gmail.com

----------ABSTRACT----------

This paper entails in detail concept of SQL's foundational, and the journey to the evolution of Big Data with the different methodologies. From its naïve beginning as a query language for relational database management systems (RDBMS), to its development into non-relational (NoSQL) databases, and finally into Big Data and Analytics data has played huge role in making technology transient. We've also made an effort to show how to use MongoDB's query language and its representation of data. We also discuss the reasons why and how you should switch from a traditional DBMS to a NoSQL one. The ACID properties of SQL are discussed in alignment of the BASE and CAP qualities that NoSQL adheres to. The purpose of this paper is to provide an objective evaluation of the performance of key• value storage in both NoSQL and SQL databases. Schema transformation, query translation, and query optimization are all methods for transforming databases that focus on problems that have yet to be fully resolved. We have an OLAP workload in a column-oriented NoSQL environment. Improved query response can be achieved in both RDBMS and NoSQL systems by properly utilizing the schema. Since data normalization isn't strictly necessary in the OLTP setting, that's where this research is focused. Data management is examined as it evolves from traditional relational databases to NoSQL-based systems in this research. Data volumes continue to rise, and Big Data has put a strain on existing infrastructure that is unable to keep up. As a result, horizontally scalable frameworks for huge data storage, such as MongoDB, are gaining in importance. The document database MongoDB is used to build large, flexible online applications.

Keywords - **Big Data, MongoDB, NoSql, Optimization, Query Translation**

Date of Submission:23-09-23 Date of Acceptance:15-11-23

I. INTRODUCTION

In reality, the continuously generated, inherently unstructured data of the present day makes relational data look inadequate. Since the need to manage and analyze more data has grown, relational databases must be adapted to NoSQL to accommodate Big Data. Big companies like Google and Facebook are among those aiming to move their data into NoSQL's Big Tables. Redesigning industry- standard schemas is a massive undertaking. Software like Apache Sqoop facilitates the migration of data from relational database management systems (RDBMS) to NoSQL databases employing preset NoSQL structures by placing an emphasis on transforming relational schemas to data representations that may be accepted by NoSQL systems on the fly. The transition from RDBMS to NoSQL raises three major concerns: data modeling, space occupancy, and efficiency. The storage and management of enormous volumes of data is getting increasingly challenging, despite the fact that relational database management remains the most effective.

Unix-based systems are often considered to be the natural home for NoSQL applications. Anybody can compile, update, and examine the code for these database management systems because they are open source and free. XML is the primary format for storing and retrieving data in RDBMS. When it comes to efficiency and scalability, non-relational databases like NoSQL shine. In addition to the tabular format employed by relational databases, the non-

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

relational (NoSQL) variety allow for the storage and retrieval of data that is structured as unstructured. OLAP (Online Analytical Processing) systems that prioritise read-intensive processes work well with NoSQL, and this is especially true when dealing with large amounts of data and real-time applications where some degree of data duplication is tolerated. In [I] NoSQL solutions may be categorized into four groups:

• Key Value data store: It is a relational database in terms of having rows and columns, but it simplifies the structure by containing only two columns: the key and the corresponding value. The values linked to each key may be accessed using its own key.

• Column oriented data store: It is fundamentally very similar to the key value data store; the only significant difference is that it stores the data in columnar format. The column family refers to the file that contains the linked data column. The column family includes the row key, which is composed of super columns. A column that doesn't contain any other columns is referred to as a super column.

• Document data stores: Data is stored in document databases as documents that are encoded with JSON, BSON, or XML. The file is kept alongside the collections.

• Graph data store: Graph databases use graph topologies such as nodes and edges to store and display the data. It is an unindexed adjacent store system. Each node in the sequence is directly linked to the adjacent node, eliminating the need for any lookup operation.

MongoDB stores data in flexible, JSON-like documents called BSON (Binary JSON), which allows for nested structures and dynamic schemas. BSON, being a document-based format, allows for the representation of complex hierarchical relationships using a single record. It supports embedded documents, which means one can nest documents within other documents. This allows for the creation of more complex data structures and facilitates the representation of relationships between different pieces of data. In addition to embedded documents, BSON also supports arrays, which means one can store multiple values within a single field. This is useful when dealing with lists or collections of related data. It gives developers the flexibility to work with developing data models. Data can be moved from one engine to another with the help of MongoDB's single deployment engine.

In this below example, the MongoDB document is represented in BSON format. We can see how the document allows for nested structures with the "address" field, which itself contains nested key-value pairs. Additionally, the "hobbies" field is an example of how MongoDB supports arrays, where multiple values are stored within a single field.

MongoDB Document (BSON)

```
{
  "_id": ObjectId("5f7f8859cc97f457f51a9ab2"),
  "name": "John Doe",
  "age": 30,
  "address": {
    "street": "123 Main St",
    "city": "Exampleville",
    "zipcode": "12345"
  },
  "hobbies": ["Reading", "Hiking", "Cooking"]
}
```

II. LITERATURE REVIEW

In [1], delve into a comprehensive exploration of SQL and NoSQL databases. The study meticulously investigates the distinct features, advantages, and limitations of both database paradigms, contributing substantially to the scholarly discourse on database management systems. Through a thorough comparative analysis, the authors provide insights into the optimal use cases for SQL and NoSQL databases, offering a nuanced understanding of their applicability in diverse computing environments. This work serves as a valuable resource for researchers, practitioners, and industry professionals navigating the evolving landscape of database technologies.

ln NoSQL, four categories can be identified to store data, they are:

1. **Key Value Databases**
2. **Document Stores Databases**
3. **Columnar Databases**
4. **Graph databases**

In [2], Sitalakshmi Venkatraman, Kiran Fahd, Samuel Kaspi, Ramanathan Venkatraman the objective of the research was to enhance understanding of Big Data analytics and NoSQL databases within their respective contexts. The study focused on investigating four main database models: Key• Value Store Databases, Column- Oriented (or Wide-Column) Store Databases, Document Store Databases, and Graph Store Databases. The need for performance improvements is the primary factor behind the transition from relational databases to NoSQL. NoSQL databases have a number of benefits over SQL databases, including simple scaling, flexible schema, lower cost, and great efficiency and performance but also have several disadvantages. The benchmark testing revealed that for interactive database applications, Couchbase achieved the lowest latencies. Couch base demonstrates superior performance compared to MongoDB and Cassandra in terms of handling an augmented operation frequency, while maintaining a lower time delay for both data reading and writing. MongoDB has a quicker writing speed than Cassandra, but both have comparable reading speeds. Furthermore, it is said that each NoSQL database is appropriate for particular application contexts and cannot be viewed as a comprehensive solution for all workloads and use cases. Despite the fact that high• performance platforms like IBM Netezza AMPP could handle Big Data,

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

choices like Hadoop have grown globally as a result of economic considerations, leading to the increase of NoSQL database usage that can interact with Hadoop conveniently. Overall, the need for new generation data analytics tools has been encouraged by Big Data, and it is reasonable to assume that both SQL and NoSQL databases will coexist simultaneously in the future.

In their study published in [3], Divya Chauhan and K. L. Bansal explores the representation and querying of data in MongoDB, a NoSQL document database that is open source. MongoDB maintains records in a binary JSON format known as BSON. One notable characteristic of MongoDB is its schema-less nature, which grants users the freedom to insert new fields or modify the structure of existing documents. JavaScript Object Notation (JSON) is a lightweight and language-agnostic data interchange format. It is widely used for structuring and representing data in a readable and portable manner. It is easy to understand and utilize. The JSON format represents data from a document as a combination of name• value pairs. A user's relevant data is encompassed within a single document, while a collection stores multiple interconnected documents. Additionally, each document has the capability to include one or more independent embedded documents. MongoDB offers a range of features that contribute to its popularity, including flexibility, a robust query.

Fast data processing is required in cloud platforms that support SQL databases to permit effective elasticity and Big Data analytics that incorporate both historical and real-time data as well as expectations for the future. However, as the requirement to store and handle huge datasets for business analytics has recently increased, NoSQL databases offer a solution to these problems. NoSQL databases provide a schema-less data store and transactions that relieve companies from the organized necessity of prioritizing the definition of the schema, which is a key restriction in SQL databases. Language, sharding capabilities, user-friendly interface, high performance, and support for multiple storage engines. In terms of replication, MongoDB facilitates master-slave replication, wherein the slave nodes function as read-only replicas of the master nodes and serve as backup nodes. The MongoDB cluster consists of three vital constituents:

1. **Shard Node:** A shard node encompasses one or more shards. Each master shard links to either one or several slave shards, which store actual data copies used in case of failures.

2. **Configuration Server:** The configuration server plays a vital role in directing requests to the appropriate shard and storing routing data.

3. **Mongos:** Mongos acts as an intermediary between the client and the database. It collects data from various shards and merges it before delivering it to the client.

Dynamic data can be conveniently stored in document data stores, making them user-friendly and straightforward. The inherent autonomy of individual documents in such stores enhances effectiveness and mitigates the adverse effects of concurrency. MongoDB is a prominent example of an open• source database technology that embodies these characteristics.

In paper [4] by Alexandru Adrian TOLE, "Big data" refers to the categorization of large and complex datasets that present challenges in terms of their management, processing, and analysis using traditional data processing tools and methodologies. Depending on the context and the specific domain in which it has been collected, big data may exist in a variety of sizes and forms

In this paper [5], Jiao Dai discussed the significance of database migration in the context of big data processing, capturing the attention of industry enterprises amid continuous advancements in data storage and management technology. Focusing on the migration from traditional RDBMS to big data platforms, the paper addresses the dilemma of balancing normalization and the unique benefits of big data. It emphasizes the potential risks associated with inappropriate denormalization, emphasizing the need for a carefully designed data migration system framework. The paper conducts an in-depth analysis of various aspects of the problem, outlining existing solutions' advantages and disadvantages in different scenarios. Additionally, it underscores the importance of optimizing multiple queries, pointing to this as a key consideration for future work in this domain.

In [6], they present a crucial contribution to the evolving landscape of database management systems. Database migration, particularly from SQL to NoSQL, is a pertinent challenge in the era of big data processing and technological advancements. The paper addresses this challenge by proposing a schema conversion model, offering a systematic approach to navigate the complexities of transitioning from traditional relational database management systems to more flexible NoSQL alternatives. The authors delve into the intricacies of schema conversion, acknowledging the delicate balance required to preserve the advantages of both SQL and NoSQL paradigms. Through a comprehensive literature review, the paper synthesizes existing knowledge, highlights gaps in the current understanding, and lays the groundwork for a structured framework that can guide practitioners and researchers in the effective conversion of database schemas. This work is valuable for its practical insights and theoretical underpinnings, contributing significantly to the ongoing discourse on optimal strategies for SQL to NoSQL migration. Apache Sqoop and DataX are the two tools

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

which are used for transferring data. The job of Sqoop is to import the data into Hadoop. Apache Sqoop is used for Big Data transfer between Hadoop and relational databases are both utilized for structured data storage, but they differ in their approaches and use cases. It can convert from RDBMS to HDFS and back again. DataX is used for data exchange between sources. It provides a framework for scheduling processes and exchanging data, known as the DataX engine. Converting schema to NoSQL database: Jack Hare is a framework which is used to translate query from SQL to NoSQL using Map Reduce.

In [7], Benymol Jose and Sajimon Abraham emphasized the use of the MongoDB database. MongoDB is a schema-adaptive approach that allows for the construction of broad, easily accessible, and durable frameworks. lt provides a flexible data model that can accommodate changing data structures and evolving requirements. When alter table instructions are used to change the schema, there is no database blockage as there is with RDBMS. MongoDB's features include adaptability, scalability, usability, high performance, readiness, support for multiple storage engines, with the WiredTiger storage engine being one of the most commonly used options. Additionally, these databases typically provide native support for JSON, which is a data format based on the JavaScript programming language. It provides the simplicity of use of MongoDB and is written in a human readable language. All data types, including numbers, texts, Booleans, arrays, and hashes, are supported.Widely used commands for database used in json db.collectionname.insert ({namel :value1,name2 :value2}) db.collection_name.re move ({condition}) db.collection_name({} ,{condition}) db.createCollection('testing') show.collections. db.testing.insert({'name':'jim','class': '2'}) db.testing.find () db.testing.find.pretty()

db.testing.update({'name': 'jim'},{$set:{'class': '50'}})
db.testing.remove

In [8], Parker et. al analyzed the comparison between SQL and MongoDB, the study reveals that MongoDB excels in runtime performance for inserts, updates, and simple queries, making it particularly suitable for larger datasets with dynamic schemas. MongoDB's schema flexibility is advantageous for scenarios like document management systems with frequently changing structures. On the other hand, SQL outperforms MongoDB in updating and querying non-key attributes, as well as in aggregate queries. Despite MongoDB's strengths, it faces challenges in aggregate functions and querying based on non-key values. The choice between the two depends on the specific needs of users; MongoDB is favored for its adaptability to dynamic structures, while SQL remains the industry standard with broader support. Notably, MongoDB's implementation requires additional effort and strategic decisions that impact performance, reinforcing SQL's position as a widely supported and industry-established database solution. Ultimately, the paper underscores the nuanced considerations involved in choosing between SQL and MongoDB, emphasizing the trade-offs in performance and structural rigidity based on the nature of the data and query requirements.

In [9], Yishan Li and Sathiamoorthy Manoharan selected NoSQL databases which are MongoDB, Hyper table, Apache CouchDB, RavenDB, Apache Cassandra, and Couchbase for comparison. For the selected databases, experiments evaluate how well these five operations perform. Authors do the selected operation (such Read or Write) five times for each of these databases, then average the times. The dataset is automatically generated in the form (kN, vN), where N is a sequence number.

The first experiment measures how long it takes to create a database bucket; the fastest database bucket creation times have been found to be offered by RavenDB, HyperTable, and MongoDB. The creation of buckets is among the slowest using CouchDB, Couchbase, and SQL Express.

The second experiment evaluates the time to read data from a bucket that matches certain keys; The read performance of SQL Express was shown to be faster for some NoSQL databases, but not all of them.

The third experiment counts the number of seconds it takes to compose a function to store key-value pairs in the designated bucket; Compared to SQL Express, RavenDB and CouchDB perform poorly while writing data. However, SQL express is outperformed by other NoSQL databases.

In the fourth experiment, the duration for removing key• value pairs from the buckets is measured. SQL Express outperforms all other NoSQL databases, except for Couchbase and MongoDB, in terms of delete performance

Fig 1. Evolution of Data

Table 1. Relationship of Database tool with data type

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

Database Tool	Support for Bucketing	Comments
MongoDB	Yes	MongoDB supports sharding, where data is divided into chunks or "shards," effectively a form of bucketing for horizontal scaling.
Couchbase	Yes	Couchbase supports sharding and allows the distribution of data across nodes, achieving similar functionality to bucketing.
Apache Cassandra	Yes	Cassandra uses the concept of keyspaces, which can be considered analogous to buckets, providing logical separation of data.
Amazon DynamoDB	Yes (through Partition Keys)	DynamoDB uses partition keys to distribute data, allowing users to effectively "bucket" data for scalability and performance.
HBase	Yes	HBase uses the notion of column families, which can be seen as a way to group and organize data, akin to bucketing.
Redis	No (Key-Value Storage)	Redis does not have built-in support for bucketing, as it is primarily a key-value store with a simpler data structure model.
MySQL	No (Traditional RDBMS)	Traditional relational databases like MySQL typically do not have native support for bucketing, as they follow a structured table-based model.
PostgreSQL	No (Traditional RDBMS)	Similar to MySQL, PostgreSQL is a traditional RDBMS and doesn't inherently support the concept of bucketing.

In the final experiment, the time required to retrieve each key from the bucket is measured. all databases, with the exception of CouchDB, were quick to get the keys. The quickest of them was SQL Express. Since Couchbase's API does not permit retrieving all the keys, it was not used in the experiment. RavenDB and CouchDB, two NoSQL databases, struggle in read, write, and delete operations. Although Casandra performs read operations slowly, it performs write and delete operations fairly well. The two databases with the fastest read, write, and delete speeds are Couchbase and MongoDB.

The research paper [10] authored by Khan et al. conducts a systematic literature review to comprehensively analyze and assess the performance of SQL and NoSQL database software architectures. Published in the journal Big Data and Cognitive Computing, the paper likely provides a detailed examination of the existing body of literature on the subject. The authors are likely to explore key aspects such as the architectural considerations, performance metrics, and comparative evaluations of SQL and NoSQL databases. Through this systematic review, the paper likely aims to identify trends, challenges, and emerging patterns in the performance analysis of database software architectures. By synthesizing information from various studies, the authors contribute to a holistic understanding of the strengths and limitations of both SQL and NoSQL databases, offering valuable insights for researchers, practitioners, and decision-makers involved in database technology selection and optimization for diverse applications.

The research paper [11] authored by de Oliveira et al. investigates the role of both SQL and NoSQL databases within the framework of Industry 4.0. The authors explore the integration of these database technologies in the context of the fourth industrial revolution, focusing on their implications for the efficient management and processing of data in smart manufacturing environments. The paper delves into the specific applications and advantages offered by SQL and NoSQL databases in the Industry 4.0 landscape. It likely discusses how SQL databases, with their structured query language, and NoSQL databases, known for their flexibility with unstructured data, contribute to handling the diverse and voluminous data generated in smart manufacturing processes. The authors likely analyze real-world use cases and provide insights into the considerations and challenges associated with choosing between SQL and NoSQL databases in the context of Industry 4.0. Overall, the paper contributes valuable insights into database technology choices in the rapidly evolving landscape of smart manufacturing and Industry 4.0.

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

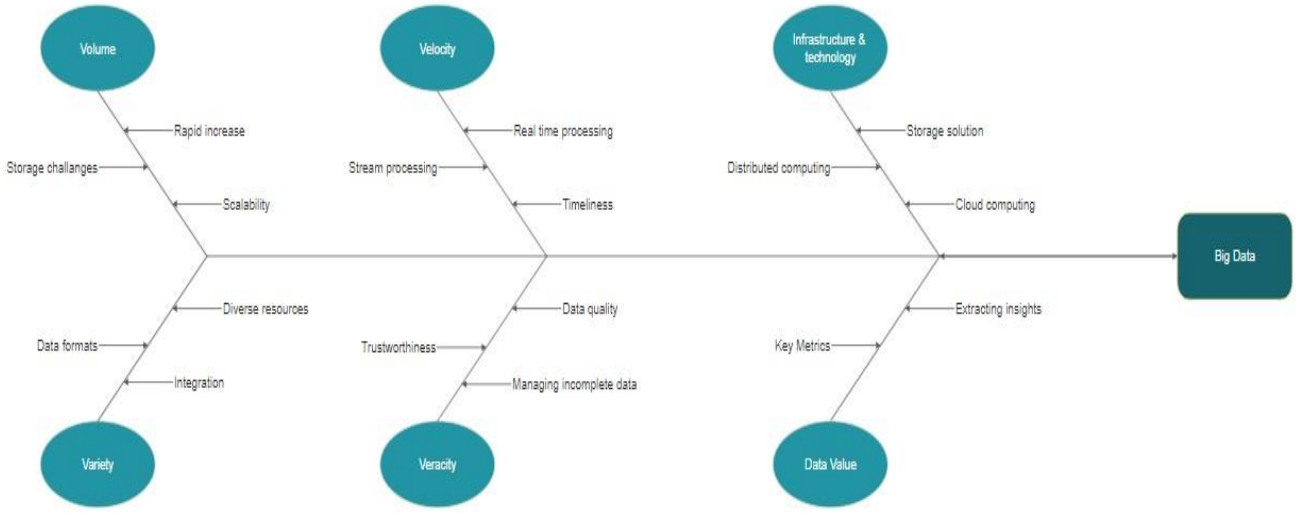

Fig 2. Evolution of Data Details.

The research paper[12] authored by Arshad et al. delves into the evolving landscape of big data analytics with a specific focus on NoSQL databases and their future trajectory in comparison to traditional relational database management systems (RDBMS). The authors highlight the distinctive characteristics of NoSQL databases that position them as potential alternatives to RDBMS in handling large-scale data. The paper explores various types of NoSQL databases and their applications, offering insights into their strengths and weaknesses.

Through a comprehensive comparison, the research evaluates the scalability and performance aspects of both NoSQL and RDBMS in the context of big data. The challenges associated with implementing NoSQL databases are discussed, and the paper concludes with reflections on current trends and potential future directions in the dynamic field of database management systems for big data analytics. The work, presented in the context of the impact of information technology on business and marketing intelligence systems, contributes to the understanding of the role NoSQL databases may play in shaping the future of big data analytics.

Summarization of few Models with Tools:

- Traditional Databases (e.g., MySQL, Oracle)
- NoSQL Databases (e.g., MongoDB, Cassandra)
- Distributed Processing Frameworks (e.g., Apache Hadoop)
- Big Data Platforms (e.g., Apache Spark, Apache Flink)

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

Table 2: Aspect comparison of different Data Models

Aspect / Database	CouchBase	MongoDB	Cassendra	NoSQL Benefits	NoSQL Limitations
Latency	Lowest latencies for interactive applications	Comparable reading speeds	Comparable reading speeds	- Transition from relational databases to NoSQL driven by performance improvements - Simple scaling - Flexible schema - Lower cost - Efficiency and performance	- No comprehensive solution for all workloads and use cases - Appropriate for specific application contexts
Performance	Superior performance in handling augmented operation frequency Low time delay for data reading and writing	Quicker writing speed than Cassandra Comparable reading speeds	Comparable reading speeds	-	-
Database Models	Key-Value, Column-Oriented, Document Store, Graph Store	Document Store	Column - Oriented	-	-
Coexistence with SQL	Coexistence with SQL databases	Coexistence with SQL databases	Coexistence with SQL databases	- Coexistence of SQL and NoSQL databases in the future - Fast data processing needed in cloud platforms for SQL databases - NoSQL solutions handle huge datasets for business analytic	- NoSQL provides schemaless data store and transactions - Relieves companies from schema definition prioritization
Connection with Hadoop	Can interact with Hadoop conveniently	-	-		

Below is a tabular comparison of SQL (relational databases), NoSQL databases, and Big Data technologies. This is a general overview, and specific features can vary among different database systems within each category.

We have compared all the tangible and intangible details of traditional database querying systems to NoSQL reaching put to Big Data Technologies.

Table 3. Comparison of Different Database types

Feature	SQL (Relational Databases)	NoSQL Databases	Big Data Technologies
Data Structure	Structured data with a predefined schema	Dynamic schema with support for semi-structured data	Support for structured, semi-structured, and unstructured data
Scalability	Vertical scaling (adding more power to existing server)	Horizontal scaling (adding more servers to the database)	Horizontal scaling with distributed processing
Schema	Strict schema adhered to by all records	Flexible schema; can evolve with application needs	Schema-on-write (structured) and Schema-on-read (flexible)
Query Language	SQL (Structured Query Language)	Query languages specific to each NoSQL database	Query languages (e.g., HiveQL for Hive in Hadoop)
Consistency	ACID properties (Atomicity, Consistency, Isolation, Durability)	Eventual consistency, not strictly adhering to ACID properties	Consistency models can vary (e.g., eventual consistency, strong consistency)
Use Cases	Well-suited for structured, transactional data	Suited for unstructured or semi-structured data	Ideal for large-scale data processing and analytics
Examples	MySQL, PostgreSQL, Oracle	MongoDB, Cassandra, Redis, Couchbase	Apache Hadoop, Apache Spark, Apache Flink
Storage Model	Table-based storage with rows and columns	Document-oriented, Key-Value, Column-family, Graph	Distributed file systems (e.g., HDFS) and NoSQL databases
Scaling Limitations	Limited scalability due to vertical scaling	Highly scalable horizontally across commodity hardware	Designed for horizontal scalability with distributed computing
Transaction Support	Strong support for transactions	Varies by NoSQL type and implementation	Transaction support can vary, with focus on batch processing
Flexibility	Rigid schema requires predefined structure	Flexible schema allows for easier adaptation to changes	Flexible schema and support for diverse data types
Community Support	Well-established with extensive community support	Growing communities with a variety of resources	Active communities with a focus on open-source development
Example Use Cases	Financial applications, ERP systems	Content management systems, real-time big data apps	Large-scale analytics, machine learning, and data processing[14]

This table provides a high-level comparison, and it's essential to consider the specific requirements of your application or use case when choosing between SQL, NoSQL, or Big Data technologies. Depending on the nature of the data and the application's needs, a combination of these technologies might also be employed in a modern data architecture.

However, the "3Vs" of big data; volume, velocity, and variety, typically serve as its defining characteristics.

Volume: Big data's sheer volume, or the enormous amount of data it involves, is one of its defining characteristics. Terabytes (JQAJ2 bytes), petabytes (JQAJ5 bytes), and even Exabyte (I QA l 8 bytes) of data can fall into this category. Social media posts, sensor data, financial transactions, and data from scientific study are a few examples of huge data volume.

Velocity: The influx of big data often occurs rapidly, requiring real-time or near real-time processing to effectively handle and analyze the data in a timely manner and analysis. For instance, a constant stream of data created by social media platforms, stock market transactions, or Internet of Things (IoT) devices must be quickly processed and analyzed.

Variety: Big data encompasses a diverse array of data types and formats, encompassing a wide range of

information sources. Structured data (such as those found in conventional databases), unstructured data (such as text documents, emails, and social media postings), semi-structured data (such as XML, JSON), multimedia data (such as photographs and videos), geographical data (such as GPS coordinates), and other types of data are all included. Integration, storage, and analysis are made more challenging by the variety of data types.

In addition to the 3Vs, big data can also exhibit two additional characteristics As shown in Figure 1:

Veracity: Veracity, which refers to the data's quality and accuracy, is frequently used to describe big data. Ensuring data quality and reliability becomes increasingly important as data sources grow and data is gathered from multiple sources.

Value: If properly analyzed, big data has great value and offers potential insights. Big data can be used to extract valuable information and insights that can be used to make better decisions, identify patterns and correlations, forecast trends, and create new goods and services.

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

Industry, application, and technological developments in data gathering and storage can all affect the size and type of big data.

Some common applications of Big Data include utilizing it for business analytics, applying it in healthcare settings, optimizing supply chain management, implementing it in smart city initiatives, and leveraging it in Internet of Things (IoT) deployments ‘

Big Data[13] systems often face challenges that involve demanding processing power and intricate network setups, which necessitate the involvement of specialists. Furthermore, the cost of software solutions can be significant, particularly if companies choose proprietary options instead of open-source alternatives. Even with open-source software, expert configuration and maintenance are still required. Although open-source software may lack direct maintenance and support, there are online communities and forums where users can seek assistance. However, in numerous instances, external maintenance teams are necessary to ensure the proper functioning of Big Data solutions.

Big data's characteristics go beyond the traditional 3Vs, encompassing additional aspects like veracity, value, variability, validity, and volatility. These characteristics collectively present both opportunities and challenges for organizations seeking to harness the potential of big data for strategic decision-making and innovation.

III. CONCLUSION

In conclusion, this study provides a comprehensive examination of the evolving database landscape, with a specific focus on the interplay between SQL and NoSQL databases within the context of Big Data. It underscores key insights and findings that shed light on the strengths and limitations of each database category. One notable highlight is the superior performance exhibited by NoSQL databases, exemplified by MongoDB, in certain scenarios. Their streamlined schema structure, reduced overhead, and simplified query mechanisms make them particularly well-suited for specific data management tasks, especially in the face of rapidly growing and diverse datasets. However, it's important to emphasize that not all NoSQL databases universally outperform SQL counterparts. The impact of database operations on performance varies across different database systems, with some excelling in particular use cases while others may lag behind. This performance variability underscores the importance of selecting the right database solution tailored to specific requirements. It also highlights the diverse traits and subcategories within the NoSQL realm, particularly focusing on graph and key-value stores. These databases shine when dealing with data featuring intricate relationship patterns or straightforward data structures, making them valuable assets in the realm of Big Data analytics. Furthermore, the adaptability of NoSQL's data modeling emerges as a critical asset, offering scalability and increased efficiency for Big Data analytics. This adaptability may pave the way for innovative data structures coexisting with traditional SQL databases. However, it's important to recognize that transitioning from SQL to NoSQL databases in response to the Big Data surge poses its own set of challenges. Data migration carries the risk of data loss, and careful management of normalization processes is essential to prevent information tampering.

In the ever-expanding landscape of Big Data, this research underscores the paramount importance of making informed decisions regarding database management systems. As organizations grapple with vast and complex datasets, the choice between SQL and NoSQL databases will remain a pivotal decision. The adaptability of NoSQL databases may serve as a catalyst for innovations in data management and analytics, further shaping the data-driven landscape of the future.

REFERENCES

Journal Papers:
[1] Sharma, Vatika, and Meenu Dave. "Sql and NoSql databases." International Journal of Advanced Research in Computer Science and Software Engineering 2, no. 8 (2012).

[2] Venkatraman, Sitalakshmi, Kiran Fahd, Samuel Kaspi, and Ramanathan Venkatraman. "SQL versus NoSql movement with Big Data analytics." International Journal of Information Technology and Computer Science 8, no. 12 (2016): 59-66.

[3] Chauhan, Divya, and K. L. Bansal. "Using the advantages of NOSQL: a case study on MongoDB." international Journal on Recent and Innovation Trends in Computing and Communication 5, no. 2 (20 I 7): 90-93.

[4] Tole, Alexandru Adrian. "Big Data challenges." Database systems journal 4, no. 3 (2013): 31-40.

.

Proceedings Papers:
[5] Dai, Jiao. "SQL to NoSql: What to do and How." In !OP Conference Series: Earth and Environmental Science, vol. 234, no. 1, p. 012080. IOP Publishing, 2019.

[6] Zhao, Gansen, Qiaoying Lin, Libo Li, and Zijing Li. "Schema conversion model of SQL database to NoSql." In 2014 Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, pp. 355-362. IEEE, 2014.

[7] Jose, Benymol, and Sajimon Abraham. "Exploring the merits of NoSql: A study based on MongoDB." In 2017 International Conference on Networks & Advances in

Int. J. Advanced Networking and Applications
Volume: 15 Issue: 04 Pages:100-200 (2023) ISSN: 0975-0290 (Special Issue – ASCIS 2023)

Computational Technologies (NetACT), pp. 266-271. IEEE, 2017.

[8] Parker, Zachary, Scott Poe, and Susan V. Vrbsky. "Comparing NoSql MongoDB to an sq! db." In Proceedings of the 51st ACM Southeast Conference, pp. 1-6.2013.

[9] Li, Yishan, and Sathiamoorthy Manoharan. "A performance comparison of SQL and NoSql databases." In 2013 IEEE Pacific Rim conference on communications, computers and signal processing (PACRIM), pp. 15-19. IEEE, 2013.

[10] Khan, Wisal, Teerath Kumar, Cheng Zhang, Kislay Raj, Arunabha M. Roy, and Bin Luo. "SQL and NoSQL Database Software Architecture Performance Analysis and Assessments-A Systematic Literature Review." Big Data and Cognitive Computing 7, no. 2 (2023): 97

[11] de Oliveira, Vitor Furlan, Marcosiris Amorim de Oliveira Pessoa, Fabricio Junqueira, and Paulo Eigi Miyagi. "SQL and NoSQL Databases in the Context of lndustry 4.0." Machines 10, no. 1 (2022):20.

[12] Arshad, Muhammad, M. Nawaz Brohi, Tariq Rahim Soomro, Taher M. Ghazal, Haitham M. Alzoubi, and Muhammad Alshurideh. "NoSQL: Future of BigData Analytics Characteristics and Comparison with RDBMS." In The Effect of lnformation Technology on Business and Marketing Intelligence Systems, pp. 1927-1951. Cham: Springer International Publishing, 2023.

[13] Swain, D., Satapathy, S., Acharya, B., Shukla, M., Gerogiannis, V. C., Kanavos, A., & Giakovis, D. (2022). Deep Learning Models for Yoga Pose Monitoring. Algorithms, 15(11), 403.

[14]Zala, K., Thakkar, H. K., Jadeja, R., Singh, P., Kotecha, K., & Shukla, M. (2022). PRMS: Design and Development of Patients' E-Healthcare Records Management System for Privacy Preservation in Third Party Cloud Platforms. IEEE Access, 10, 85777-85791.