International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Recent Advances in Computer Applications and Information Technology
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

SQL and NOSQL Database: A Comparative Study

Lukesh Kawde, Piyush Paulzagade

G H Raisoni University, Amravati, Maharashtra, India

Abstract

As the volume of data generated by different applications has experienced tremendous and unprecedented growth across different categories of applications in today's society, the need for high-performance database management systems has also increased significantly. Two of the most commonly used database types are SQL databases and NoSQL databases. SQL-based databases operate on the relational model, which is based on a predefined schema (the way the data is organised) and guarantees the consistency of data through strict adherence to the ACID properties (the rules governing the treatment of data)[1]. SQL databases are well suited for applications that require structured data and complex queries. In contrast, NoSQL databases operate with a dynamic schema (flexible ways to organise data), so they can support large volumes of either unstructured or semi-structured data. Successful implementation of NoSQL has also proven able to provide extreme horizontal scalability (expanding the power of a computer by adding more computers), and have proven reliable, yielding high availability, which makes NoSQL a great choice for applications requiring both big data and real-time processing needs [2]. This paper will evaluate and compare the respective benefits and weaknesses of both SQL databases and NoSQL databases, in terms of their data models, scaling characteristics, consistency, performance, and real-world use cases. Each of the two database types will be studied holistically. The study will result in recommendations regarding how organisations can make the best use of either SQL databases or NoSQL databases [3].

KEYWORDS: *SQL Database, NoSQL Database, Relational Database, Non-Relational Database, Data Models, Scalability, ACID, BASE, Big Data, Database Management Systems.*

1. Introduction

Due to the rapid increase in digital data and internet-based applications, organizations have faced significant issues with effective storage and management of these large amounts of data. Most software systems utilize one or more databases as their primary source of structured storage, retrieval and manipulation of data. As such, almost all software systems depend on databases to store data. For this reason, since the emergence of SQL (Structured Query Language) databases, they have been, historically speaking, the predominant choice of organizations and software developers for data storage and management. SQL databases have traditionally provided organizations with reliable data storage solutions, a structured schema for their data, strong transaction support, and no single point of failure. On the other side are the many large-scale Internet applications being developed today as well as big data, cloud computing and real-time web applications have all generated a demand for NoSQL (Not Only SQL) databases as a viable alternative to the SQL paradigm for data management[5].

Due to the rapid increase in digital data and internet-based applications, organizations have faced significant issues with effective storage and management of these large amounts of data. Most software systems utilize one or more databases as their primary source of structured storage, retrieval and manipulation of data. As such, almost all software systems depend on databases to store data. For this reason, since the emergence of SQL (Structured Query Language) databases, they have been, historically speaking, the predominant choice of organizations and software developers for data storage and management.[6] SQL databases have traditionally provided organizations with reliable data storage solutions, a structured schema for their data, strong transaction support, and no single point of failure. On the other side are the many large-scale Internet applications being developed today as well as big data, cloud computing and real-time web applications have all generated a demand for NoSQL (Not Only SQL) databases as a viable alternative to the SQL paradigm for data management. In order to solve 0these limitations, the concept of NoSQL (not SQL, i.e. not just a structure) databases was developed as a different approach to managing data. In contrast to traditional SQL databases that rely on rigid or structured data models, NoSQL databases are based on flexible or unstructured data models and are able to horizontally scale/out-source/extend the storage of data across multiple servers. By design, NoSQL databases are intended to provide high performance, high availability, and fault tolerant ("failure resistant") functionality to support large scale and cloud computing systems. Examples of the types of NoSQL databases that exist today are MongoDB, Cassandra, Redis, Neo4j, and others -- all of which support various types of data models including document, key-value, column family an SQL databases use the relational database model to store data using tables where there are pre-defined schemas, and the relationships between tables are identified by foreign keys.[7]

Data maintained in SQL databases is subject to the rules and criteria for maintaining the integrity of the data (i.e., together with maintaining the ACID properties of a SQL database). This model makes SQL database technology a standard methodology for applications that involve complex querying and require highly consistent data, such as banking applications, enterprise resource planning (ERP) applications, and inventory management applications. Examples of popular SQL databases include: MySQL, PostgreSQL, Oracle, and Microsoft SQL Server. Unlike SQL databases, NoSQL databases are design to accommodate the rapidly evolving requirements for storing, managing and processing large-scale, distributed and unstructured data. NoSQL databases allow for more flexible schema designs than relational databases and the resulting architecture can be horizontally SQL (Structured Query Language) database management systems have long been the traditional model for data management. SQL databases are based on a

relational data model that uses rows and columns to store data in tables that have defined relationships with each widely used for supporting applications such as banking systems, financial transactions, airline reservations, and enterprise other. They enforce the same schema and have ACID properties (atomicity, consistency, isolation, durability), which result in high consistency and reliability of data. Because SQL databases provide these benefits, they are resource planning (ERP) systems. Additionally, NoSQL databases are increasing in popularity due to the rise in cloud computing and microservice architectures. Database management system have a long with each other database. multiple servers. By design, NoSQL databases are intended to provide high performance, high availability, and fault tolerant ("failure resistant") functionality to support large scale and cloud computing systems. Examples of the types of NoSQL databases that exist today are MongoDB, Cassandra, Redis, Neo4j, and others -- all of which support various types of data models including document. fot maintaining the of the data banking application that involve complex highly consisits data such example of popular SQL database consistency, isolation, durability), which result in high consistency and reliability of data. Because SQL databases Query Language) databases, they have been, historically speaking, the predominant choice of organizations and relational databases not being suited for handling large-scale data storage and horizontal scaling. As explained by

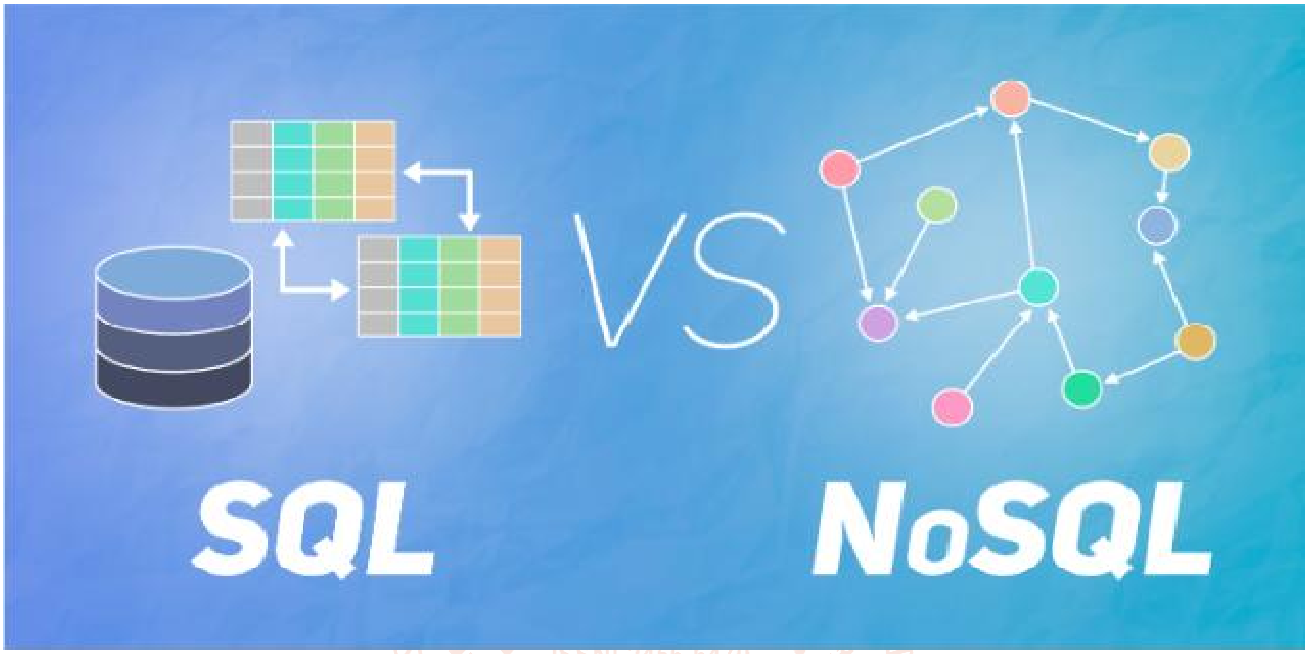

Fig 1. Comparative Architecture of SQL and NoSQL Databases

2. Literature Review

In recent years, the rapid rise in use of data-heavy applications has limited the amount of research performed on Database Management Systems (DBMSs). Most of this research has focused on comparing SQL to NoSQL databases from various angles such as Scalability, Consistency, Performance, Data Modeling, and Real World Applicability. Date (2004) looked at the theoretical basis for Relational Database Systems (RDBMSs) and made them aware of the importance of normalizing data, having referential integrity, and using a structured schema to design data. Date's conclusion stated that RDBMSs provide a high level of trust for storing structured data and as a result are an excellent choice for use in a transaction-oriented system, thus making an RDBMS an ideal choice for any mission critical business application. Stonebraker et al. (2010) provided a discussion regarding the limitations to using RDBMSs in distributed environments on a large scale. The authors argue that traditional RDBMSs are not designed to be optimized for horizontal scale and fault tolerance, which are important requirements for web scale applications. This research provided much of the background for driving the development of NoSQL databases. Leavitt (2010) provided insight into the development of NoSQL databases as a response to the limitations of RDBMSs. This publication provided an explanation for how NoSQL technologies trade off the ability to provide strict consistency guarantees for improved availability and performance, based on the principles of the CAP theorem. Leavitt also illustrated the fact that many large companies had been early users of NoSQL technology. According to Stonebraker et al. (2010), traditional SQL databases cannot be used in large-scale distributed environments because they were designed to manage only small amounts of data. The authors state that performance problems have arisen with many modern web applications due to the original design of relational databases not being suited for handling large-scale data storage and horizontal scaling. As explained by Leavitt (2010), issues of scalability and performance are the main reasons organizations have adopted NoSQL databases. He noted that organizations are willing to trade-off having strict consistency for greater availability and system responsiveness [8].Cattell (2011) provided a detailed classification of different types of NoSQL databases, discussing how each has been built as well as how they work. He cited schema flexibility as one of the major advantages of many NoSQL databases when it comes to developing applications that require managing rapidly changing data structures. Additionally, he indicated that there are challenges associated with query complexity and the fact that NoSQL databases are not standardized.

According to a survey of NoSQL databases. Han et al. (2011) compared NoSQL (Not Only SQL) databases and SQL databases based on their data models, ability to scale, and

the consistency of their transaction processing. The results of their study concluded that SQL is more efficient at executing complex database queries involving multiple related tables, while NoSQL is better able to handle very large amounts of distributed data. Using the example of base versus acid (the BASE model versus the ACID model), Pritchett (2008) described how "eventual" consistency allows for distributed systems' scalability and represents an important principle that was utilized in designing NoSQL databases. Brewer (2008, 2012) updated the CAP theorem (i.e., the CAP theorem describes the fundamental trade-offs between consistency, availability, and partition tolerance) to illustrate how such trade-offs can affect the design of real-world computer systems and applications that utilize both SQL and NoSQL databases[9]In their paper, Abadi (2012) examined the consistency trade-off within modern distributed databases while emphasizing that application requirements will dictate the level of consistency. This research helped close the gap between theoretical models for databases and their application in real life. Polyglot persistence was introduced as a concept in Sadalage and Fowler (2012). It shows how modern applications often require multiple database technologies to work together. The authors continued on from this to reinforce the point that NoSQL technology compliments rather than replaces traditional SQL database technology. The authors of Hecht and Jablonski (2011) evaluated NoSQL database technologies based on possible use cases, and concluded that a database should be selected based upon the requirements of the workload and application, rather than following a trend from an information technology perspective. More recent literature indicates that cloud computing has also played an important part in the evolution of databases. There are numerous studies that show how the adoption of NoSQL databases through cloud platforms is encouraged because of their ability to scale easily and how cost-effective these types of databases are to use [10]. On the other hand, cloud based SQL databases have also evolved over time to address the scaling limitations found in traditional SQL databases. He noted that organizations are willing to trade-off having strict consistency for greater availability and system responsiveness [8].Cattell (2011) provided a detailed classification of different types of NoSQL databases, discussing how each has been built as well as how they work. He cited schema flexibility as one of the major advantages of many NoSQL databases when it comes to developing applications that require managing rapidly changing data structures. Additionally, he indicated that there are challenges associated with query complexity and the fact that NoSQL databases are not standardized.

3. Research Methodology

This study will analyze and evaluate SQL and NoSQL databases, using a descriptive and comparative research methodology. The method compares both SQL and NoSQL-type databases by creating predetermined parameters based on both previous literature (and studies) and practical observations. This research will be qualitative in nature using only secondary data. I will use a comparative analysis framework to evaluate the data from my secondary sources based on the characteristics, advantages, and disadvantages of SQL and NoSQL databases. I will be comparing SQL and NoSQL databases in this study by looking at the conceptual differences rather than doing a qualitative/quantitative experiments testing on my research by using SQL and NoSQL databases [11]. The data in this research will come from my secondary sources, including the following: Research articles published in leading journals, Conference proceedings, Technical books/whitepapers ,Authoritative documentation of popular SQL and NoSQL databases, Repudiated online resources and case studies Each source will provide both a theoretical view of technology database options along with actual examples of database technology in use. Evaluating SQL versus NoSQL databases requires some degrees of structure and impartiality; so we have compared them according to eight parameters: 1) Data Model (relational /non-relational), 2) Schema Flexibility, 3) Scalability (vertical/horizontal), 4) Performance, 5) Consistency & Transaction Support, 6) Query Language, 7) Security Features, and 8) Use Cases & Applications[12].Each of these parameters will be discussed individually to compare how / where SQL and NoSQL databases are alike and different. Analyzing gathered information through comparative analysis will allow examining/synthesizing information from multiple sources to find out what their common strengths, weaknesses, and general trends are. This analysis presents the findings through the use of tables with descriptive wording to help an audience consume the information clearly. No specific software tools or experimental platforms were used in this research, as it principally utilized data from previously published research-based articles. Examples of the database systems used to collate data and carry out equivalent prior studies include: Microsoft SQL Server, IBM, MySQL, PostgreSQL, MongoDB, Cassandra, and Redis. Findings are validated by cross-referencing information from multiple sources. Conflicting viewpoints are critically analyzed to ensure balanced and unbiased conclusions [13].This research is limited to secondary data sources and does not include primary experimental testing or real-time performance benchmarking of SQL and NoSQL databases. As a result, the findings are dependent on previously. This research is limited to conceptual and literature-based analysis. Performance results are based on existing studies rather than real-time experiments. Additionally, rapid advancements in database technologies may affect the long-term applicability of some findings To determine how well SQL databases and NoSQL databases can manage growth in terms of data size and the number of users, scalability assessment is performed. The comparison between vertical scaling (scale-up) and horizontal scaling (scale-out) will demonstrate their potential influences on cost, efficiency and complexity of the underlying architecture [15].

This research focuses on how transactions are handled, how data is consistent, and how systems are tolerant to faults in both types of databases. As such, particular care is taken to evaluate SQL database's ACID characteristics and NoSQL database's eventual consistency model for reliability. The analysis of various security factors including authentication, authorization, encryption, and access control is performed using materials from valid research publications as well as government documents; with understanding of the positive and negative characteristics of SQL and NoSQL systems regarding data protection is arrived at[14]. This study adopts a descriptive and comparative research methodology to analyze and evaluate SQL and NoSQL database systems. The research does not involve experimental implementation or performance benchmarking; rather, it relies entirely on qualitative analysis of existing literature and documented case studies. The purpose of this methodology is to systematically compare the conceptual foundations, architectural characteristics, advantages, and limitations of

SQL and NoSQL databases. Scalability assessment examines how effectively database systems handle growth in1.Data volume2.Concurrent users3.Transaction throughput Examples of database systems frequently referenced in prior studies include relational systems such as Microsoft SQL Server, MySQL, PostgreSQL, and IBM Db2, as well as NoSQL systems such as MongoDB, Apache Cassandra, and Redis.These sources provide both theoretical insights and practical examples of database technologies deployed in real-world environments such as cloud computing platforms, large-scale web applications, and distributed systems. The comparative analysis is conceptual in nature, meaning that the focus is on theoretical foundations, architectural differences, and documented real-world applications rather than controlled experiments or quantitative testing

The data for this research is collected from reputable secondary sources, including peer-reviewed journal articles, conference proceedings, scholarly textbooks, technical whitepapers, and official documentation from major database vendors. Authoritative documentation and case studies provide insight into real-world implementations of relational systems such as Microsoft SQL Server, MySQL, PostgreSQL, and IBM Db2, as well as NoSQL systems such as MongoDB, Apache Cassandra, and Redis. These sources provide both theoretical perspectives and practical examples of database technologies deployed in large-scale web applications, cloud computing environments, and distributed systems. The comparative analysis is based on predetermined evaluation parameters designed to ensure structure, consistency, and impartiality. The study examines the fundamental data models used by each paradigm, contrasting the relational model of SQL databases with the non-relational models adopted by NoSQL systems. It evaluates schema design approaches, comparing fixed, predefined schemas with flexible or dynamic schema structures. Scalability is analyzed by examining vertical scaling strategies traditionally associated with relational databases and horizontal scaling mechanisms commonly implemented in distributed NoSQL architectures. The assessment considers how these scaling approaches affect system cost, architectural complexity, and performance efficiency. Transaction management and data consistency form a central component of the analysis. SQL databases are evaluated with respect to their support for ACID properties—atomicity, consistency, isolation, and durability—while NoSQL databases are examined in the context of distributed system principles such as eventual consistency and trade-offs described by the CAP theorem. The research critically explores how these differences influence reliability, fault tolerance, and system behavior under high-load or distributed conditions. To maintain validity and reduce bias, findings are verified through cross-referencing multiple independent sources. Conflicting viewpoints identified in the literature are critically analyzed to ensure that conclusions remain balanced and evidence-based. The comparative synthesis of information enables identification of recurring strengths, weaknesses, and overall trends in database technology adoption. Security analysis is conducted by reviewing Thus, database performance is not something that can be evaluated independently of the requirements of the application. The results of this investigation indicate that consistency, availability, and partition tolerance trade-offs are key factors when selecting a database

Fig 2 Research Methodology Framework Diagramz

4. Result

Fig 3 SQL and NoSQL Databases – Comparative Study

5. Conclusion

The findings from this comparative study of SQL and NoSQL databases provide detailed information regarding how current database technologies meet varying needs regarding data management practices. SQL databases have been shown to be reliable options that offer an established and mature form of structured-data storage, as well offer sufficient capabilities for carrying out complex-query-processing needs. Furthermore, SQL technology adheres to the relational model's structure and also conforms to the ACID attributes which support mission-critical applications (where accuracy and therapeutic efficacy must be maintained).The increasing significance of NoSQL databases in dealing with the issues caused by big data, distributed systems and cloud applications is evident in the findings of the research. The horizontal scalability and loose schema design of NoSQL databases facilitates the effective management of very large amounts of unstructured and semi structured types of data. Thus, NoSQL databases are particularly well suited for uses where high availability, low latency and real-time need for processing data is required[16]. Another key result of this study shows that both performance and scalability are extremely dependent upon the workload characteristics being applied. SQL databases perform best in environments that have a large number of transactions while NoSQL databases excel at high read and high write workloads. Thus, database performance is not something that can be evaluated independently of the requirements of the application. The results of this investigation indicate that consistency, availability, and partition tolerance trade-offs are key factors when selecting a database.

While SQL databases focus on maintaining a high level of consistency and reliability; NoSQL databases have typically relaxed some consistency constraints in order to achieve greater scalability and fault tolerance. This understanding of how these trade-off factors work together, allow

organizations to make better choices for which database solution is best suited for their system[17]. Concerning database design and selection, the study shows that trade-offs as articulated in the CAP theorem with respect to the design of database systems affect trade-offs of database systems, including consistency, availability, and fault tolerance. The design of SQL databases prioritizes strong consistency, while the design of NoSQL databases typically follow an eventual consistency model to support the need for scalability and resiliency. It is important to understand these design trade-offs to effective create a database system. Security analysis shows that SQL databases have tended to provide more mature and standardized security mechanisms than NoSQL databases; however, NoSQL databases can achieve similar levels of security if configured and managed properly. However, as new security features are being developed for NoSQL databases, the differences in security between SQL and NoSQL databases are narrowing. In summary, this research supports that there is not one superior type of database; rather, SQL and NoSQL databases are typically complementary in function when developing applications today, which is why there has been an increase in the use of hybrid or polyglot persistence architectures by organizations to maximize the strengths of both sorts of databases. Future research could expand on this work by implementing experimental benchmarking and cost and security assessments, in cloud-native and distributed environments.

Reference

[1] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks (1970)," *Communications of the* ACM, vol. 13, no. 6, pp. 377–387.

[2] R. Cattell, "Scalable SQL and NoSQL Data Stores (2011)," *ACM SIGMOD Record*, vol. 39, no. 4, pp. 12–27.

[3] M. Stonebraker, "SQL Databases v. NoSQL Databases (2010)," *Communications of the ACM*, vol. 53, no. 4, pp. 10–11.

[4] A. Silberschatz, H. F. Korth, and S. Sudarshan, "Database System Concepts (2019)," 7th ed., McGraw-Hill Education.

[5] D. Sullivan, "NoSQL for Mere Mortals (2015)," Addison-Wesley.

[6] C. J. Date, "An Introduction to Database Systems (2004)," Pearson Education.

[7] H. Garcia-Molina, J. D. Ullman, and J. Widom, "Database Systems: The Complete Book (2008)," Pearson.

[8] R. Ramakrishnan and J. Gehrke, "Database Management Systems (2003)," McGraw-Hill.

[9] R. Elmasri and S. B. Navathe, "Fundamentals of Database Systems (2016)," Pearson.

[10] T. Connolly and C. Begg, "Database Systems: A Practical Approach to Design, Implementation, and Management (2014)," Pearson.

[11] C. Coronel and S. Morris, "Database Systems: Design, Implementation, and Management (2015)," Cengage Learning.

[12] P. DuBois, "MySQL (2013)," Addison-Wesley.

[13] Oracle Corporation, "Oracle Database SQL Language Reference (2023)," Oracle Documentation.

[14] Microsoft, "SQL Server Documentation (2023)," Microsoft Documentation.

[15] PostgreSQL Global Development Group, "PostgreSQL Documentation (2023)," PostgreSQL Documentation.

[16] International Organization for Standardization, "ISO/IEC 9075: Information Technology — Database Languages — SQL (2016)," ISO Standard.

[17] E. Brewer, "CAP Twelve Years Later: How the 'Rules' Have Changed (2012)," *Computer*, vol. 45, no. 2, pp. 23–29.

[18] S. Gilbert and N. Lynch, "Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services (2002)," *ACM SIGACT News*, vol. 33, no. 2, pp. 51–59.

[19] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking Cloud Serving Systems with YCSB (2010)," in *Proceedings of the 1st ACM Symposium on Cloud Computing (SoCC)*, pp. 143–154.

[20] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: A Distributed Storage System for Structured Data (2006)," in *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 205–218.