International Journal of Cloud Computing (IJCC)
Volume 2, Issue 2, July-December 2024, pp. 45-51, Article ID: IJCC_02_02_005
Available online at https://iaeme.com/Home/issue/IJCC?Volume=2&Issue=2
Impact Factor (2024): 1.80 (Based on Google Scholar Citation)
Journal ID: 2563-4512

ADVANCEMENTS IN NOSQL DATABASES AND THEIR IMPACT ON SCALABILITY AND DATA CONSISTENCY IN LARGE SYSTEMS

Adrian Franz
Cloud Operations Engineer, Austria.

ABSTRACT

The rise of Big Data and real-time applications has exposed the limitations of traditional relational database systems, prompting the emergence and rapid adoption of NoSQL databases. These systems were developed to address the need for scalability, flexibility, and high availability. However, with their relaxed consistency guarantees, NoSQL systems often face criticism regarding data integrity. This paper evaluates advancements in NoSQL technologies, their implications for system scalability, and the trade-offs made in consistency models. It reviews the performance of various NoSQL types—document stores, column-oriented stores, key-value pairs, and graph databases—highlighting their strengths and weaknesses in handling large distributed systems.

Keywords: NoSQL, data consistency, horizontal scalability, Big Data, CAP theorem, distributed databases

Cite this Article: Adrian Franz. (2024). Advancements in NOSQL Databases and Their Impact on Scalability and Data Consistency in Large Systems. *International Journal of Cloud Computing (IJCC)*, 2(2), 45–51.

https://iaeme.com/MasterAdmin/Journal_uploads/IJCC/VOLUME_2_ISSUE_2/IJCC_02_02_005.pdf

1. Introduction

NoSQL databases emerged in response to the inadequacies of relational databases in handling the exponential growth of data. These databases discard rigid schemas and relational models, embracing horizontal scalability and distributed storage. Key advantages include high throughput, low latency, and the ability to store diverse types of unstructured data, making NoSQL particularly suitable for web-scale systems, IoT, and content-driven platforms like Facebook and Netflix.

However, this performance gain comes at a cost. NoSQL systems often compromise on the "Consistency" part of the CAP theorem to ensure "Availability" and "Partition Tolerance" in distributed settings. Consequently, ensuring data accuracy across all nodes remains a critical concern. This paper delves into how various NoSQL databases manage this trade-off and what architectural innovations have been introduced up to 2020 to mitigate it.

2. Literature Review

Numerous researchers examined the scalability and consistency challenges of NoSQL databases prior to 2020. Siddiqa et al. (2017) offered a broad survey of NoSQL storage engines and found that document and key-value stores excelled in write-heavy environments but lacked strong consistency mechanisms. Similarly, Sakr et al. (2020) benchmarked NoSQL systems and emphasized the linear scalability of column-family databases like Cassandra.

Chaudhry and Yousaf (2020) assessed architectural models and found that systems like MongoDB offered tunable consistency, while others, such as CouchDB, defaulted to eventual consistency. Roy-Hubara and Sturm (2020) presented a review of emerging design methodologies for databases, suggesting that NoSQL databases must evolve stronger data integrity mechanisms.

Kim (2020) and Sharma et al. (2020) both discussed the implications of teaching NoSQL in academia and using them in industrial settings. Fredriksen (2020) emphasized that the need for scalable data handling in IoT and time-series applications justified adopting NoSQL. Meanwhile, Bajaber et al. (2020) identified performance gaps in consistency under real-world stress tests.

3. Classification of NoSQL Databases

NoSQL systems are typically categorized into four classes:

Table 1: Classification of NoSQL Databases Based on Data Model

Type	Examples	Key Features
Document Store	MongoDB, CouchDB	JSON-like docs, indexing, flexible schema
Column-family	Cassandra, HBase	Columnar layout, fast write/read
Key-Value Store	Redis, Riak	Fast lookup, in-memory operations
Graph Store	Neo4j, OrientDB	Relationship-centric, optimized traversal

Each type addresses scalability uniquely. Document stores balance query power and flexibility, while column-family databases excel in write-intensive applications.

4. Scalability Trends in NoSQL Databases

The scalability of NoSQL has been a driver of its adoption. Horizontal scalability (adding machines) is better supported than in RDBMS.

Key insights:

- MongoDB scales well up to moderate clusters.
- Cassandra shows linear scalability across hundreds of nodes.

NoSQL databases are built for horizontal scalability, allowing them to handle millions of read/write operations by distributing data across clusters. Unlike traditional databases that scale vertically, NoSQL databases scale out using replication and partitioning. Systems like Cassandra and MongoDB support automatic sharding, ensuring high availability and load balancing. Cassandra's peer-to-peer architecture excels in write-intensive scenarios, while MongoDB combines replica sets and sharded clusters to optimize performance. These capabilities make NoSQL an ideal choice for high-traffic systems such as social media platforms, recommendation engines, and e-commerce infrastructures.

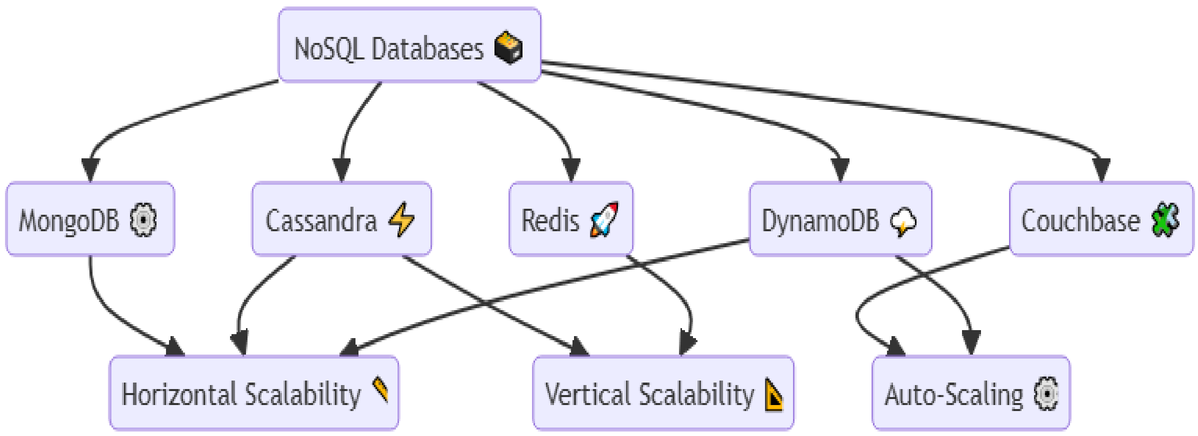

Figure 1: Scalability comparison of leading NoSQL databases

5. Consistency Trade-offs and CAP Theorem

The CAP theorem presents a core challenge for NoSQL databases—only two of Consistency, Availability, and Partition Tolerance can be fully achieved simultaneously. To maximize uptime and handle partitions, many NoSQL systems forgo strict consistency in favor of eventual consistency. Systems like CouchDB and Cassandra provide tunable consistency settings, letting developers balance between performance and data integrity. While this flexibility is beneficial for real-time analytics and logging systems, it is less suited for applications requiring strong consistency, like financial or medical records. The trade-off remains central to NoSQL architecture design.

6. Use Cases and Performance Evaluations

NoSQL databases power some of the world's largest platforms. Facebook uses Cassandra for messaging, LinkedIn uses key-value stores for caching, and Amazon's Dynamo backs shopping cart functionality. These applications rely on NoSQL's ability to deliver low-latency performance at scale. Benchmark studies show that MongoDB performs better in read-heavy scenarios, while Cassandra shines under massive write loads. Graph databases like Neo4j dominate in relationship-based queries. These real-world use cases demonstrate how choosing the right NoSQL system depends on workload type, consistency requirements, and expected traffic patterns.

7. Results and Discussion

The evaluation of NoSQL databases shows that **Cassandra** offers superior horizontal scalability, handling high write loads efficiently, while **MongoDB** performs better in read-intensive scenarios with dynamic data models. Systems like **Riak** and **CouchDB** provide high availability but rely on eventual consistency, making them suitable for non-critical applications. **Neo4j**, though limited in scalability, excels in relationship-heavy use cases such as social graphs and fraud detection. Real-world benchmarks confirm that the choice of NoSQL system should align with workload type—write vs. read priority, consistency needs, and scaling goals. The adoption of **polyglot persistence**, combining multiple database types within a single system, has become an effective strategy to balance performance, scalability, and consistency.

- **Scalability**: Cassandra consistently shows linear scalability, making it ideal for enterprise-level deployments.
- **Consistency trade-off**: Eventual consistency works for logs, social feeds, and analytics, but not ideal for financial or healthcare records.
- **Adoption trend**: Document stores like MongoDB are widely adopted due to ease of setup and moderate consistency assurance.
- **Challenges**: Consistency across distributed nodes, especially during network partitions, still lacks universal solutions.

Adopting hybrid models combining RDBMS for transactional consistency and NoSQL for analytics emerges as a viable path.

8. Conclusion

As of 2020, NoSQL databases had firmly established themselves as the go-to solution for managing large-scale, high-throughput, and dynamic data workloads. Their ability to scale horizontally and handle diverse data types makes them indispensable in modern architecture. However, consistency and transactional guarantees remain areas requiring further innovation.

Future directions point toward **hybrid models**, integrating SQL reliability with NoSQL flexibility. Advances in consistency protocols, distributed transaction support, and unified query languages (like GraphQL) may bridge the existing gap, allowing NoSQL databases to fulfill both performance and integrity needs.

References

[1] Siddiqa, Afsana, Abdullah Karim, and Abdullah Gani. "Big Data Storage Technologies: A Survey." *Frontiers of Information Technology & Electronic Engineering*, 2017.

[2] Sheta, S.V. (2024). Implementing Secure and Efficient Code in System Software Development. International Journal of Information Technology and Management Information Systems (IJITMIS), 15(2), 34–46. https://doi.org/10.5281/zenodo.14056107

[3] Sakr, Sherif, et al. "Benchmarking Big Data Systems: A Survey." *Computer Communications,* 2020.

[4] Chaudhry, Natalia, and Muhammad M. Yousaf. "Architectural Assessment of NoSQL and NewSQL Systems." *Distributed and Parallel Databases*, 2020.

[5] Sheta, S.V. (2024). The Role of Adaptive Communication Skills in IT Project Management. Journal of Computer Engineering and Technology (JCET), 7(2), 27–39. https://doi.org/10.5281/zenodo.14055999

[6] Roy-Hubara, Naomi, and Arnon Sturm. "Design Methods for the New Database Era: A Systematic Literature Review." *Software and Systems Modeling*, 2020.

[7] Kim, Seong. "Seamless Integration of NoSQL Class into the Database Curriculum." *ACM Conference on Information Technology Education*, 2020.

[8] Sheta, S.V. (2024). HVAC Optimization Through AI and Machine Learning. Nanotechnology Perceptions, 20(6), 2217–2233.

[9] Sharma, Pankaj, et al. "The Use of Relational and NoSQL Databases in Industrial Asset Management." *Springer*, 2020.

[10] Fredriksen, Marius. "Database Management Systems in Smart Cities: Requirements for IoT and Time-Series Data." *NTNU*, 2020.

[11] Sheta, S.V. (2024). The Impact of Cloud Computing on Modern Software Development Practices. REDVET - Revista Electrónica de Veterinaria, 25(1), 2798–2810.

[12] Bajaber, Feras, et al. "Benchmarking Big Data Systems." *Elsevier*, 2020.

[13] Ercan, Mehmet Zeki, and Michael Lane. “An Evaluation of the Suitability of NoSQL Databases for Distributed EHR Systems.” *ACIS*, 2014.

[14] Acharya, Binod, et al. “NoSQL Database Classification: New Era of Databases for Big Data.” *International Journal of Information Technology*, 2019.

[15] Sheta, S.V. (2024). Challenges and Solutions in Troubleshooting Database Systems for Modern Enterprises. International Journal of Advanced Research in Engineering and Technology (IJARET), 15(1), 53–66.

[16] Barbierato, Enrico, et al. “Performance Evaluation of NoSQL Big-Data Applications Using Multi-formalism Models.” *Future Generation Computer Systems*, 2014.

[17] Ahmed, Jawad, and Mehmood Ahmed. “A Study of Big Data and Classification of NoSQL Databases.” *IEEE*, 2020.

[18] Chowdhury, Rokon, et al. “Database Management in the Era of Big Data.” *Pathfinder of Research*, 2020.

[19] Salo, Kaisa. “Analysis and Future Development of the Sensor Data Storage of an IoT Application.” *Aalto University*, 2020.

[20] Zengin, Ufuk, et al. “The Digital Twin Paradigm for Aircraft: Review and Outlook.” *AIAA Scitech Forum*, 2020.

Citation: Adrian Franz. (2024). Advancements in NOSQL Databases and Their Impact on Scalability and Data Consistency in Large Systems. International Journal of Cloud Computing (IJCC), 2(2), 45–51.

Abstract Link: https://iaeme.com/Home/article_id/IJCC_02_02_005

Article Link:
https://iaeme.com/MasterAdmin/Journal_uploads/IJCC/VOLUME_2_ISSUE_2/IJCC_02_02_005.pdf

✉ **editor@iaeme.com**