Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.7660358

A REVIEW AND COMPARISON OF KEY DISTRIBUTED DATABASE CHARACTERISTICS ACROSS SEVERAL NOSQL DISTRIBUTED DATABASES

Kewal Krishan
Research Scholar, Punjabi University Patiala, Punjab
kakkarkewal@gmail.com

Gaurav Gupta
Assistant Professor, Punjabi University Patiala, Punjab
gaurav.shakti@gmail.com

Gurjit Singh Bhathal
Assistant Professor, Punjabi University Patiala, Punjab
gurjit.bhathal@gmail.com

Abstract: These days, the major challenge for database organizations worldwide is efficient and fast data access and processing. The present relational model is good for vertical scaling but is insufficient to meet the big data requirements. So horizontal scalability is required to meet the challenges. This leads to the emergence of non-relational database models commonly known as NoSQL databases. One another major challenge of non-relational is the consistency issues due to the nonfulfillment of transactional support. Thus Consistency is the major challenge of in NoSQL databases that need to be reviewed further. In this paper, the various models of the NoSQL databases are analyzed in terms of key distribution parameters. Comparisons have been made for finding out the best NoSQL types and their models that can be used for particular parameters based on the scenario.
Keywords: NoSQL, RDBMS, Consistency, transaction, CAP.

1. Introduction:

A Database system is a method of storing and managing data instead of a simple file system. The data stored in the database can be reachable only through certain methods of storage and retrieval. Relational database management system (RDBMS) is the dominant technology today. It stores structured data in tables. It means the data is fit into a predefined structure. SQL is used to access the data that is stored in Relational databases. Mostly these Organizations have stored data in a structured way by using relational databases. [1]. When the centralized storage of data is not sufficient, organizations want data should be geographically accessible with fast speed, then parallel and distributed databases came into the existence. In a distributed database, the concurrency (Accessing the same data by multiple users at the same time) is managed by using the concept of serializability. The control is managed by Locking, like 2 phase protocol, Time stamps, and many others. So, relational databases drive for ACID (Atomicity, Consistency, Isolation, and Durability) properties for secure transaction. But by using this technique the organizations are going to negotiate on the availability concerns. Modern web-based applications and Concurrency are the big challenges of Relational Database Management Systems. According to the author, the biggest and most time-consuming issue is ACID (Atomicity, Consistency, Isolation and Durability) and Vertically Scalability. The new era is big data is dealing basically with 3V's i.e. volume, velocity and variety.

The major player Google, Amazon and Facebook are not using the traditional RDBMS. There are many reasons behind this. All major players are using NoSQL databases. NoSQL is better in terms of horizontal scale up [1]. Due to this, we are getting a better option for availability, performance, and Fault Tolerance (In case failure of any node data is available). But there is a big challenge in this distributed scenario. If data is replicated to other nodes, how much time it will take to reach all the nodes, so that a consistent view should be there. For that particular time, the renewed data shall not be available to all the nodes. The traditional databases use a locking mechanism for consistent data but lose in terms of availability. But today the scenario is different because availability is the prime issue. So the question of losing consistency is genuine or not depends upon application [2]. Moreover not all the applications need strong consistency (All Nodes having the same view of the data). It depends on the application and other parameters.

This paper contributes in the following way:

- It provides the complete details of the various NoSQL databases models and the techniques associated with them.
- Comprehensive literature work has been done to find out the consistency related issues and challenges for the NoSQL models.
- It gives insights into the best models and the related applications that follow for effective implementation based on the scenario.

This paper has two main sections .Section 1 gives the introduction of the paper. Section 2 discuss about the models of the NoSQL databases and the associated applications that are following those models. The related work is mentioned in the section 3 of the paper. Comparisons and the analysis of the models have been done in section 4 and the conclusion is mentioned in the last section.

2. NoSQL Models

NoSQL is basically "Not Only SQL".Today big data processing is a major challenge for database organizations and the present models of RDBMS are not appropriate for them. In this section the various models of NoSQL databases are defined and their general properties along with the related applications where they are used are discussed [3].

Key-Value: In this type of model the data is stored by using a key. A hash table is used that contain a unique key and a pointer to a particular data item. One of the most important drawbacks of this poor applicability for cases involving the processing of key ranges. The ordered key-value design overcomes this constraint and significantly improves aggregation capability. Another challenge in this as the data volume increases it becomes difficult to manage different unique keys. Examples are Radis, MemCached, and Dynamo.

Column store: Traditionally data is stored by row, but this model store data by column. Columns can be distributed to different nodes. One of the advantages as compared to key based is the fast search and data aggregation. Examples are Cassandra and PNUTS.
Document-oriented: This model uses documents to store the data. A document consist of field of attributes, for example: name="Kewal", last name ="Kakkar" document of two fields. The documents are in semi-structured formats, JSON, XMLetc. One major advantage of this model is that the metadata is associated with the data in hand. Databases under this category are MongoDB and Couch DB.

Graph-oriented: Here graph is used to store the data. It is characterised by arcs and vertices, with its attribute. One of the main plus point of this model is easy deployment of any application on it. Databases under this category are Neo4j11 and Hyper Graph DB.

When we talk about the consistency in the NoSQL databases then there are majorly two approaches come in the picture i.e. ACID and BASE. The former one means that once the data is written all the rules applied to it for reading and other access where as in the later the once the data is written then it means it is assumed that is available for further course of action. Most of the NoSQL database not follows the ACID properties but some exceptions are there like the MongoDB, Neo4j. But in BASE based NoSQL databases the execution is faster. Hence there is a need for extensive research work for finding the best consistency models.

3. Related Work

This section narrated the work that has been done by the researchers in context to the various models for NoSQL in terms of data and transactional consistency for getting better insights for future endeavors.Han et al. [4] explain the context, the basic features, and the NoSQL data model. In addition, this paper also differentiates the NoSQL databases based on the CAP theorem. In the end, the standard NoSQL databases are discussed in detail based on storage and retrieval methods. Some properties for choosing the NoSQL from a business angle are taken. NoSQL databases have various limitations and their usage in cloud computing has not been done as per the author that still needs to be done. Markis et al. [5] execute the analysis of some key design characteristics of NoSQL systems and use them for various capabilities. Also, it also represents the association between NoSQL systems and cloud structures including their impact with respect to one other. It is determined that there is a trade-off between high availability and the consistency of the data. Most of the NoSQL databases choose two parameters of CAP that is between consistency, availability, partition, or tolerance (CAP theorem). In [6], the author designed a new approach for the performance and scalability requirements of web-based applications which are not taken care of by traditional relational databases. The relational databases use the ACID (Atomicity, Consistency, Isolation, and Durability) properties and he NoSQL databases use BASE (Basically Available, Soft State, Eventual consistency) principles. It covers the BASE feature of some NoSQL databases and acts as a very important step towards the consistency models for NoSQL databases. It is depicted that the resources involved in the development of distributed database systems are stringent, and neither CAP is able to cater at all. Moreover, the addition of consistency in the design while modern approaches considerations is very important as per the author [7].

Corbelling et al. [8] discusses the design and implementations of NoSQL databases for providing the insights of the tools and their usage in the applications of interest. In this author first of all compared the NoSQL databases with traditional RDBMS and figure out some important points like that many NoSQL databases by using this design have the more chances of better execution. Still, there are many applications that consider large correlated data with diverse relationships that need to be taken in future works. A layer-based framework [9] is presented by the author to support the ACID properties through the NoSQL in the middle layer. This middle layer keeps records of all running transactions and interacts with other layers for the better execution of transactions. This leads to more scalability, and throughput. From the results it is inferred that the throughput of the system increases on account of middle layers and it also lead to the more updates for a particular transaction. In [10], the author proposes a model for strong consistency in NoSQL databases that allows for estimation of random wait time of the read request so that record update completion done. Gessert et al. [11] proposed a NoSQL toolbox for effective utilization using the top down approach. Here the authors use the comparative classification model that compares the functional and non- functional requirements to techniques and algorithms employed in NoSQL databases. It is evident that the given proposed approach is very effective when the large amount of NoSQL database system is in hand. Another author [12], proposed a new multi-key transactional model that support with transaction model for accuracy It also has the feature of strong data consistency. The proposed model is suitable where consistency, availability and efficiency can be changes as per the application need. The authors proposed model is validated through a design developed by them of MongoDB. On the similar note one of the author [13] has proposed a transaction processing method for MongoDB but it highlights the problem that the plural data cannot be updated while maintaining the ACID properties. For this they give an approach for processing the plural data as a single transaction for MongoDB. It is a flexible data structure without schema and proven more effective.

Imam et al. [14] gives an effective NoSQL model for the changing requirements of the application in hand like when the parameters like consistency, availability and scalability are to be balanced in accordance with system requirements. The develop model provides a schemas at the initial stage of system development s and CRUD (Create, Read, Update and Delete) operations used in the course of action.

NoSQL databases have been involved as a new tool to fill the missing gaps in relational databases the schema-less design, horizontal scaling, and eventual consistency. This author [15] evaluates and compares the consistency model of popular NoSQL databases: Redis, Cassandra, MongoDB, Neo4j, and Orient DB. It is inferred from the results that MongoDB exhibits good tolerance and having more consistency. But for the availability, Cassandra is the better choice. On the other hand in case of partition intolerance or non-distributed databases are in hand then both Neo4j and Orient DB are the best one to use. Ramzan et al. [16] address the challenges while the storage of NoSQL databases. They came out with a protocol and criteria to evaluate the various models in hand. The need of security in terms of trust and privacy are the main outcomes that need to be achieved in the future.

So from the above literature work it is inferred that there is huge scope for the development of new models and consistency based evaluation of those models with respect to the existing one in the near future. It can be done through the comparative analysis of already existing models on the basis of common factors for the comparison.

4. Comparative Analysis of NoSQL Models

NoSQL databases consist of Database Management Systems which are used for large distributed datasets and are not based on relational models. There are different types of NoSQL operational models for data processing such as Key Value Databases, Columnar Databases, Document Oriented Databases and Graph Databases. The brief outline of each database is given below.

Key Value Databases: These kind of NoSQL models store data in the form of associated arrays in the form of key value pairs. Keys in these models act as unique identifiers and are responsible for the retrieval of associated values. Values in associated arrays are in the form of objects like string, integer and JSON formats. Key value databases store data in the form of single collection without any relation or structure and is in contrast to that of relational databases where data is defined in terms of data structures consisting of rows and columns. Key value databases are very much scalable and efficient in terms of data modelling and most common use cases include queuing of messages, caching and management of sessions. Key Value open source databases are Riak, Redis and Memcached.

Columnar Databases: These kind of databases store data in the form of columns. In these databases, each column is stored as a region or file in terms of storage which is quite different in comparison to relational databases. The data is stored in columnar databases in particular order of records which helps to read columns of interest and not every row in table. Columnar databases are carrying a feature of run length encoding which helps administrators of database to minimize the space required for single columns which will eventually result into higher speed-ups for reading as queries require to cover lesser number of rows. These databases are having importance in applications which require aggregate functions. Columnar open source databases are Cassandra, HBase and Click House.

Document Oriented Databases: These kind of databases store data in documents form. These databases also work in the form of keys and values where each document contains key as its unique identifier and value is treated as document itself. These databases are different than key value databases in a way that each document in these databases contains metadata which provides structure to data. On the other hand, key value databases, data is opaque and it is the job of application to know where data is stored. Document oriented databases contain query language or API for the retrieval of documents out of metadata. These databases also store all data in the form of single document which is quite different in relational databases where the data gets stored across multiple tables and databases. Document oriented databases store data in the form of JSON, XML and PDF documents. Document oriented open source databases are MongoDB, Couch DB and Couch base.

Graph Databases: These databases also store data in the form of documents, however the difference is that they contain one more added layer to documents which provide the relationship existing between the documents. These databases define relationships in documents in terms of graphs. These databases find their use in cases where it is easier to gain outcome based on the relationships existing between individual documents. These databases are most popular in applications of recommendation engines and fraud detections. Graph based open source databases are Neo4j, Orient DB and ArangoDB. The overall summarization of different NoSQL databases is given in Table 1.

Table 1. Summarization of different NoSQL databases

Database Name	**Type of Database**	**Utility of Database**
Key Value Database	Riak	NoSQL distributed database used for storage of large unstructured data with efficient read and writing power based on multi-cluster replication.
	Memcached	Efficient NoSQL database based on distributed memory based caching system which fixes many problems related to data caches.
	Redis	NoSQL database with wide support to data structures and having built in replication. Redis databases are supported with automatic partitioning while dealing with clusters.
Columnar Database	Cassandra	Cassandra database provides high performance and scalability. It provides fault tolerance when data runs on commodity hardware.
	HBase	This database is scalable and distributed database and often used along with Apache Hadoop in the form of Big Tables over the Hadoop Distributed File System.

	ClickHouse	Database which provides support to analytical data which is generated in real time and provides support to queries of SQL as well.
Document Oriented Database	MongoDB	Popular database where data is stored in terms of JSON documents and has powerful querying language which processes documents to any possible nesting level.
	CouchDB	NoSQL document data store where data is stored in terms of JSON and provides high performance with low latency and high throughput.
	Couchbase	NoSQL document data store where data is stored in terms of JSON and queries are handled with JavaScript.
Graph Database	Neo4j	This database is graph based database management system with its storage and processing in terms of graphs.
	OrientDB	This database is multi model NoSQL which combines key value database, document based database and graph based database together with querying in terms of SQL language.
	ArangoDB	This database is multi model NoSQL which combines key value database, document based database and graph based database together with powerful querying language.

The performance of Key-Value Distributed Databases, Columnar Databases, Document Oriented Databases and Graph Databases on the basis of parameters of Scalability, Atomicity, Complexity, Querying, Flexibility and Replication is summed up in Table 2.

Table2. Parameter based comparison of NoSQL databases

Feature/ Parameter	**Key Value Database**	**Columnar Database**	**Document Database**	**Graph Database**
Scalability	Higher Scalability	Higher Scalability	Variable Scalability	Variable Scalability

Atomicity	Atomicity Available	Atomicity Available	Atomicity Available	Atomicity Not Available
Complexity	No Complexity	Low Complexity	Low Complexity	High Complexity
Query Performance	Higher	Higher	Higher	Variable
Flexibility	Higher	Moderate	Higher	Higher
Replication	Yes	No	Yes	Yes

5. Conclusion

In this review article, an attempt has been made to compare various NoSQL databases of Key-Value Databases, Document Oriented Databases, Columnar Databases, and Graph Databases on the basis of parameters of Scalability, Atomicity, Complexity, Querying, Flexibility and Replication. The comparison reveals that in Key Value Databases, the whole value is returned and these databases are unable to filter value fields. Also it is not possible to update part of value and one needs to update it on the whole. Columnar Databases are providing low performance while dealing with transactions and are being slow. Document Oriented databases are quite reliable, however do not provide any support for join operations and take high memory usage. Graph Databases lack consistency in terms of high performance. Therefore here is strong requirement of any novel consistency model which will cover these limitations in terms of performance.

References

1. Nayak, A., Poriya, A., & Poojary, D. (2013). Type of NOSQL databases and its comparison with relational databases. *International Journal of Applied Information Systems*, *5*(4), 16-19.

2. Vanderschaaf, N. R. (2001). *U.S. Patent No. 6,253,213*. Washington, DC: U.S. Patent and Trademark Office.

3. Sharma, S., Tim, U. S., Gadia, S., Wong, J., Shandilya, R., & Peddoju, S. K. (2015). Classification and comparison of NoSQL big data models. *International Journal of Big Data Intelligence*, *2*(3), 201-221.

4. Han, J., Haihong, E., Le, G., & Du, J. (2011, October). Survey on NoSQL database. In *2011 6th international conference on pervasive computing and applications* (pp. 363-366). IEEE.

5. Makris, A., Tserpes, K., Andronikou, V., & Anagnostopoulos, D. (2016). A classification of NoSQL data stores based on key design characteristics. *Procedia Computer Science*, *97*, 94-103.

6. Chandra, D. G. (2015). BASE analysis of NoSQL database. *Future Generation Computer Systems*, *52*, 13-21.

7. Abadi, D. (2012). Consistency tradeoffs in modern distributed database system design: CAP is only part of the story. *Computer*, *45*(2), 37-42.

8. Corbellini, A., Mateos, C., Zunino, A., Godoy, D., & Schiaffino, S. (2017). Persisting big-data: The NoSQL landscape. *Information Systems*, *63*, 1-23.

9. Lotfy, A. E., Saleh, A. I., El-Ghareeb, H. A., & Ali, H. A. (2016). A middle layer solution to support ACID properties for NoSQL databases. *Journal of King Saud University-Computer and Information Sciences*, *28*(1), 133- 145.

10. Burdakov, A., Grigorev, U., Ploutenko, A., & Ttsviashchenko, E. (2016, February). Estimate on models for NoSQL database consistency characteristics. In *2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)* (pp. 35-42). IEEE.

11. Gessert, F., Wingerath, W., Friedrich, S., & Ritter, N. (2017). NoSQL database systems: a survey and decision guidance. *Computer Science-Research and Development*, *32*(3-4), 353-365.

12. Ogunyadeka, A., Younas, M., Zhu, H., & Aldea, A. (2016, March). A multi-key transactions model for NoSQL cloud database systems. In *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)* (pp. 24-27). IEEE.

13. Kudo, T., Ishino, M., Saotome, K., & Kataoka, N. (2016). A proposal of transaction processing method for MongoDB. *Procedia Computer Science*, *96*, 801-810.

14. Imam, A. A., Basri, S., Ahmad, R., Watada, J., & González-Aparicio, M. T. (2018). Automatic schema suggestion model for NoSQL document-stores databases. *Journal of Big Data*, *5*(1), 46.

15. Diogo, M., Cabral, B., & Bernardino, J. (2019). Consistency Models of NoSQL Databases. *Future Internet*, *11*(2), 43.

16. Ramzan, S., Bajwa, I. S., & Kazmi, R. (2019). Challenges in NoSQL-Based Distributed Data Storage: A Systematic Literature Review. *Electronics*, *8*(5), 488.