NoSQL databases: Critical analysis and comparison

Conference Paper · October 2017

DOI: 10.1109/IC3TSN.2017.8284494

Shelly Sachdeva
National Institute of Technology Delhi

NoSQL Databases: Critical Analysis and Comparison

Adity Gupta, Swati Tyagi, Nupur Panwar, Shelly Sachdeva
Jaypee Institute of Information Technology, India
adityguptа4@gmail.com, tswati934@gmail.com,
nupurpanwar42@gmail.com, shelly.sachdeva@jiit.ac.in

Upaang Saxena
Minjar Cloud Services Private Limited, India
upaang.saxena@minjar.com

***Abstract*— The current research explores and differentiates between various forms in which NoSQL databases exist. It examines the need of NoSQL and how they have become an important alternative to relational databases. NoSQL databases can be categorized into four major classifications which are: key value stores, graph databases, wide column stores, and document stores. These categories are compared on the basis of functional features and non-functional features. The non-functional features include performance, scalability, flexibility, structure and complexity. The functional features include de-normalization, joins, atomicity, aggregation and keys. Then for further analysis, one database is selected from each of these categories that is, MongoDB (document stores), Cassandra (wide column stores), Redis (key value stores), and Neo4j (graph databases). Selected databases are compared on their data model, CAP theorem, distributive properties and other factors. By performing the comparison on non-functional features, it has been found that a document store can be used if high performance, flexibility and scalability are required and if we have represented the data in JSON format. Column store can be used for semi structured data which requires high performance and scalability. Redis is anin-memory store and performs exceptionally fast in the case of single shard operation. Graph databases can be used when it comes to highly interconnected data and continuously evolving data models. The comparison between MongoDB, Cassandra, Redis and Neo4j concluded that all of them follow horizontal scaling and are schema free. Except Neo4j, others don't have complete ACID properties. Write and delete operations are fast for databases MongoDB, Redis and Cassandra, whereas read operation is comparatively slow in Cassandra. In case of Neo4j, REST performance is similar to MongoDB, whereas embedded is comparatively slow. We also discuss how these databases work in a distributed environment.**

Keywords—database; NoSQL; comparison; database systems;

I. Introduction

The recent advancements in distributed web applications and cloud computing have generated large volumes of data which cannot be managed by single nodes systems. Thus, distributed storage offers the solutions that provide high availability and scalability are needed. Examples of distributed (non-relational storage) are Dynamo by Amazon and Google's Big Table.

A. Relational Database

Initially, every record was maintained manually, but the advent of technology has led to drastic changes over the years. To make maintaining data easier databases were created. A database varies from a simple text document to much more complex databases. These databases have to be refined periodically to remove any kind of redundant, inconsistent or dirty data so as to perform effectively. The most common, well-known conception to store this data is through relational model. Structured Query Language (SQL) extracts relevant data from the pool of database.

Relational databases are the most common type of database because of its simplicity. In an RDBMS [26] data is broken into several tables which can be accessed as per the requirements without actually making changes in the table. Operations like join, aggregation, addition, creation, retrieval and deletion are easily performed in relational databases and it is also very easy to extend or modify existing tables. Examples: SQL Server [23], Oracle Database [24] and MySQL [25].

B. Why NoSQL Databases are used?

The major challenge with the growing data is its non-uniformity. Due to this problem, in recent years, a non-relational database is needed to scale the growing need of industry and at the same time, must be highly efficient. This gave rise to NoSQL databases which are highly scalable, efficient and can store large amount of data.

Though RDBMS are able to manage all three kinds of data i.e., structured, semi-structured and unstructured, but labor and compromises are required to achieve efficient storage of unstructured and semi-structured data. RDBMS stores structured data as it is because they are already in required form. But, storing the semi-structured data involves a few complexities. The semi-structured data needs to be converted in relational data before storage. Also, in case of unstructured data the data is saved as a blob object and is not stored directly.

Hence, to satisfy this non-uniformity of data a fresh thought was given to the storage of data, leading to the creation of NoSQL (Not only SQL) Databases.

NoSQL databases have emerged as an important substitute to relational databases and we choose them according to features like scalability, availability, and fault tolerance. They do not follow the general table/row/column approach which is practiced by all RDBMSs. NoSQLs are primarily called distributed or non-relational database. They support horizontal scalability, so to scale number of servers are increased rather than upgrading hardware of the system which happens in RDBMS where vertical scalability is performed.

C. Importance of NoSQL

NoSQL [16][39]databases are geared towards management of large, varied and continuously changing data sets. They are often used in distributed systems or cloud databases. In

978-1-5386-0627-8/17/$31.00 ©2017 IEEE

NoSQL databases rigid schemes and many other limitations are avoided. They were initially introduced as databases to provide an alternative to the long existing relational databases. For these NoSQL databases scalability, fault tolerance and availability are the most important deciding factors. They do not follow the strict schema approach of RDBMSs [26].
NoSQLs have a certain edge over relational databases as they are able to tackle big data efficiently, provide high velocity, and can handle variety of data with varied complexities. As they are horizontally scalable, managing them is also simpler, which can be done by the addition of a new node to the cluster which will handle load efficiently. To avoid failure as the data is distributed amongst several servers, so even if one fails others are still in working condition, and hence can easily continue the work of the faulty node. This guarantees that single point of failure doesn't exist in the database and also depicts true fault tolerance of a NoSQL database. In case of data and function, it also enjoys the ability of built-in redundancy.

There are four general types of NoSQL databases where every database has its own properties:

- Graph database: The basis of this type of databases is graph theory. Examples: Neo4j [27] and Titan [28].
- Key-Value store: In this database, we store the data in two parts, namely key and value. Examples: Redis [29], DyanmoDB [30], Riak [31].
- Column store: Here, data is stored in the form of sections of columns of data. Examples: HBase[32], BigTable[18][20] and Cassandra [33].
- Document database: This database is higher version of key-value stores. Here values are saved as documents which are data in the form of complex structures (like JSON). Examples: MongoDB [34] and CouchDB [35].

CAP [19] theorem explains the limitation posed on all databases. It states that anyone can pick only any two out of the three features abbreviated as CAP in which C stands for Consistency, A for Availability, and P stands for Partition tolerance. The main statement of Brewer's theorem says that for any shared-data system, a maximum of two properties can be exist from these properties [36].

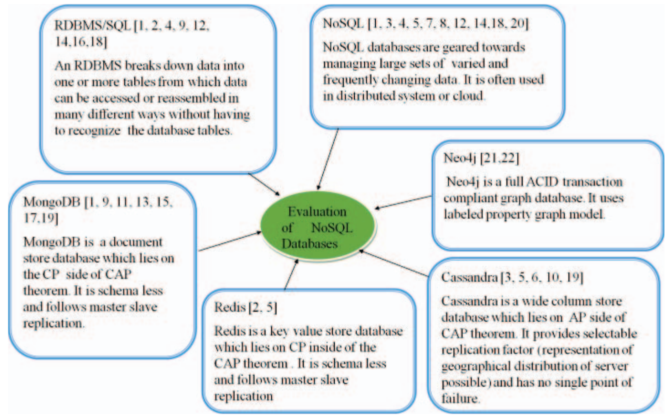
II. Literature Survey

The detailed summary of related papers has been presented in appendix (Table 5).
Figure 1 depicts that we surveyed papers from different sources such as, VLDB, IEEE, ACM and SIGMOD etc. These papers have been categorized into six groups (NoSQL, SQL/RDBMS, Redis, MongoDB, Cassandra and Neo4j) and critical analysis of each group has been performed.

- NoSQL:

NoSQL, Not Only SQL, is distributive data model that does not follow relational database guidelines. It supports huge data storage, horizontal scaling and massive- parallel data processing [9]. NoSQL also supports data which cannot be easily expressed in terms of SQL [17].

Figure 1: Classification of relevant papers



Hence NoSQL databases have been adopted as a widespread substitute to conventional SQL databases, especially in the scenario where we are managing extremely large scale of data [13]. NoSQL was developed to overcome the disadvantages of relational databases. Therefore, many companies invested into researching the field of these databases [9]. Nowadays ACID properties can be achieved by NoSQL databases also with the help of middleware [3]. NoSQL databases rely on the services and capabilities of the underlying storage systems [8].

- RDBMS/SQL

Relational databases are the most common type of database because of its simplicity. In an RDBMS [26] data is divided into multiple tables which are usually in their normalized form for more efficiency. While accessing data it can be reassembled as per the requirements of the user. Structured Query Language (SQL) consists of four types of queries that are data definition language (DDL), data control language (DCL) and data manipulation language (DML). Each one has its own set of queries which are executed to define data i.e. create table, alter table etc. , to manipulate existing data as per the requirements using update, insert etc. and define the control of transaction using queries like roll back, commit etc. respectively. A detailed comparison of NoSQL versus RDBMS on features such as data validity, query language, data type, data storage, schema, flexibility, scalability and ACID compliancy is presented [40] Generally in NOSQL, only single record transactions and an eventual consistency replica system are supported, where it is assumed that transactions are commutative. Thus, ACID transactions are compromised for performance [41].

- Document store (MongoDB):

MongoDB resides on the CP side of CAP theorem. MongoDB supports format BSON [37] which is JSON [38] like document with dynamic schemas which make data integration easier and faster. Some of the common features of MongoDB are that it has a document-oriented storage layer and for replication of data between servers it uses asynchronous replication [15]. In MongoDB and other NoSQL databases additional implementation decisions are made which were not required in SQL databases. These decisions have an effect on the performance of databases[12]. Other advantages of using MongoDB include easy replication, MapReduce, and clustering [11].

- Column store (Cassandra):

Cassandra resides on the AP side of CAP theorem. It provides its users scalability as it's linearly scalable and availability without compromising with its performance. Cassandra is easily capable of managing heaps of data across number of commodity servers while maintaining high availability without any single point of failure. [6][10]

- Key value store (Redis):

Redis resides on the CP side of CAP theorem. Redis key-value data store with a choice for data durability. It is an in-memory NoSQL database, Redis supports various data structure servers like strings, lists, sets, hashes and sorted sets. It can be replicated using relax master slave architecture.

- Graph database (Neo4j):

Neo4j utilizes labeled property graph model. In Neo4j nodes and edges can have properties associated with them. Nodes can be further associated with labels which categorize different them according to their roles. Neo4j is a full ACID transaction compliant graph database. It can be used as both standalone server (REST interface) or in embedded form [22].

III. COMPARITIVE ANALYSIS OF NOSQL DATABASES

A. *Comparison of NoSQL databases on the basis of functional and non functional requirements.*

Table1: Different NoSQL databases on basis of Non-functional features

Data model	Performance of queries	Scalability of data	Flexibility of schema	Structure of database	Complexity of values
Key-value store	High	High	High	Primary key with some value	None
Column Store	High	High	Moderate	row consisting multiple columns	Low
Document Store	High	Variable (High)	High	JSON in form of tree	Low
Graph Database	Variable	Variable	High	Graph – entities and relation	High

Table 1 compares key-value store, column store, document store and graph database based on their non-functional features such as, Performance of queries, Scalability of data, Flexibility of schema, Structure of database and Complexity of values.Table1 depicts that for a simple data that can be represented as a key-value pair form easily; key value store may be chosen as it will provide high performance, scalability and flexibility. If the value can be represented in column from and is semi structured, then column store is the appropriate database as it will provide high performance and scalability. If data can be represented in JSON format, then document store should be preferred as it has high performance, flexibility and usually high scalability. If we need to store data which can be represented using graph theory or if the data is strongly inter-related, then we use graph store model which provides high stability, but performance and scalability is variable.

Table 2 compares the four categories of NoSQL databases on the basis of functional features, such as, De-normalization, Single aggregate (adding multiple composite keys to a single key), Atomicity, Unordered Keys, Derived Table (a table can be created on the basis of master class this helps in sorting according to multi-dimensional indices), Composite Key, Composite Aggregation, Aggregation, Aggregation and Group by, Adjacency Lists (each node is designed as an individualistic record that accommodates arrays of immediate ancestors or descendants), Nested Sets and Joins.

Key value store should be avoided if we want to use composite key, joins or derived table operations on the database.

Document Store should be avoided if we want to use de-normalization, unordered key, composite key, composite aggregation, joins or derived table operations on the database.

Wide Column store should be avoided if we want to use unordered keys, aggregation and group by, adjacency lists, nested sets or joins operations on the database.

Graph Store should be used if we want to perform just de-normalization.

Table 2: Different NoSQL databases on basis of functional features

S. No.	Features	Key Value Store	Document Store	Wide Column store	Graph Store
1.	**Denormalization**	Applicable	Not Applicable	Applicable	Applicable
2.	**Single Aggregate**	Applicable	Applicable	Applicable	Not Applicable
3.	**Atomicity**	Applicable	Applicable	Applicable	Not Applicable
4.	**Unordered Keys**	Applicable	Not Applicable	Not Applicable	Not Applicable
5.	**Derived Table**	Not Applicable	Not Applicable	Applicable	Not Applicable
6.	**Composite Key**	Not Applicable	Not Applicable	Applicable	Not Applicable
7.	**Composite Aggregation**	Applicable(ordered)	Not Applicable	Applicable	Not Applicable
8.	**Aggregation**	Applicable	Applicable	Applicable	Not Applicable
9.	**Aggregation and Group by**	Applicable	Applicable	Not Applicable	Not Applicable
10.	**Adjacency Lists**	Applicable	Applicable	Not Applicable	Not Applicable
11.	**Nested Sets**	Applicable	Applicable	Not Applicable	Not Applicable
12.	**Joins**	Not Applicable	Not Applicable	Not Applicable	Not Applicable

Table 3: Differentiation of Cassandra, MongoDB, Redis, Neo4J and MySQL.

S. No.	Feature	Wide Column Store (Cassandra)	Document Store (MongoDB)	Key Value Store (Redis)	SQL (MySQL)	Graph Database (Neo4j)
1	**Database Model**	Wide column store	Document Store	Key-Value Store	Relational DBMS	Graph database
2	**Description**	It is one of the most popular wide column store database. It is based on the concept of BigTable	It is one of the well-known document store database	It is an in-memory data structure store and an important key value store	Widely used open source RDBMS	Open source graph database
3	**DB**	Key space	Database	Database	Database	Graphs
4	**Table**	Column Family	Collection	Hash set, List, Set, Sorted set and String	Relation	Label
5	**Value**	Rows	Documents	Key value pair	Rows	Node and edges
6	**Read Operations**	Slow[4]	Fast[4]	Fast[5]	Slow (Join dependent)	Data dependent
7	**Write Operations**	Fast[4]	Fast[4]	Fast[5]	Slow	Data dependent
8	**Delete Operations**	Fast [4]	Fast[4]	Fast[5]	Slow	Data dependent
9	**Language**	Java	C++	C[14]	C and C++	Java, Scala
10	**License**	Open Source	Open Source	Open Source	Open Source	Open Source
11	**Data scheme**	Schema-free	No particular schema is followed but usually contents of same documents as a convention have similar structures though it is not mandatory	Schema-free	Yes	Schema-free
12	**Predefined types**	Yes; ASCII, int, blob, counter, decimal, double, list, map, set, text, timestamp, varchar	Yes; Boolean, date, object_id, String, Integer, double.	Partial; data types supported for value are strings, Bit arrays, hyper logs, hashes, lists, sets, sorted sets, and geospatial indexes	Yes; int, float, double, date, time, bit, char, enum, binary, blob, Boolean	Yes; Boolean, byte, short, int long, float, double, char, string
13	**Server side scripts**	No	JavaScript	Lua	Yes	Yes
14	**Triggers**	Yes	No	No	Yes	Yes
15	**Partitioning methods**	Sharding (In this very large databases are divided or partitioned into much smaller and manageable units called shards)	Sharding with no individual point of failure	Sharding	Horizontal partitioning, sharding with MySQL Cluster or MySQL Fabric	Partitioning should be avoided in Neo4j
16	**Foreign Keys**	No	Usually, not used, however equivalent operation with DBRef can be done	No	Yes	Yes
17	**Transaction Concepts**	Atomicity and Isolation are supported for single operations	Atomic operations can be performed within single document	Optimistic locking, atomic execution of command blocks and scripts	ACID	ACID
18	**User Concepts**	Access rights for users can be defined per object	Access rights for users and roles	Simple password – based access control.	Users with fine grained authorization concepts; no user groups or roles	Users, roles and permissions
19	**Website**	cassandra.apache.org	www.mongodb.org	redis.io	www.mysql.com	www.neo4j.com
20	**Developer**	Apache Software Foundation	MongoDB, Inc.	Salvatore Sanfilippo	Oracle	Neo Technology
21	**Initial Release**	2008	2009	2009	1995	2007

B. Comparison on the basis of categories of NoSQL databases.

In Table 3, comparison is made using features such as, Database Model, Description, Database, Table, Value, Read Operations, Write Operations, Delete Operations, Language License, Data scheme, Predefined types, Server side scripts, Triggers, Partitioning methods, Replication Methods, Scaling, Foreign Keys, Transaction Concepts, User Concepts, Website, Developer, Initial Release and Current Release.

The current research has taken databases from each category of NoSQL databases that is Cassandra, MongoDB, Redis) and Neo4j. Table 3 presents the differentiation of various NOSQL databases with an example from each category. SQL is a Relational DBMS; Cassandra falls under the category of Wide Column stores which are based on the ideas of BigTable and DynamoDB. MongoDB is a Document Store, whereas Redis follows the concepts of a Key-Value Store.

Cassandra has a keyspace analogous to a database in SQL and a column family instead of a table. MongoDB makes a collection, while Redis has options of hashes, lists, sets and sorted sets instead of a table.

Read operations are slower in Cassandra and SQL compared to the other two. For both write and delete operations, SQL falls short in comparison to all NoSQL databases. In case of Neo4j, even though the embedded version is slow REST's performance is roughly similar to MongoDB[21]. For partitioning methods, SQL is the only one to use Horizontal Partitioning, while the rest use sharding. Also SQL is the only one which uses the concept of foreign keys. Coming to the transaction concepts, SQL and Neo4j follow the ACID properties. For single operations, atomicity and isolation are supported in Cassandra. Atomic operations are possible inside a long document in MongoDB, while Redis supports optimistic locking and atomic execution of command blocks and scripts.

MongoDB supports access rights for different types of users. For Cassandra, access rights can be established per object. Redis supports uncomplicated password based access control [44]. In MongoDB, authorization and authentication are disabled by default. Here, the authorization is provided by following a role-based approach on a per-database level. Provision for authentication on a per-database level has been made available in basic MongoDB where the users subsist particularly for a single logical database [42]. Authorization and authentication is enabled by default in Neo4j [45].

C. Comparison on the basis of distributive properties

Table 4 explains how the four databases work when database is spread on multiple computers which may or may not be in same physical location. In case of MongoDB auto sharding is used to partition data amongst multiple nodes in order preserving manner. MongoDB supports horizontal scaling which enables it to scale data across multiple nodes. The load is distributed equally across nodes an if balance is disrupted it automatically redistributes the load equally.

Table 4: Analysis of NOSQL databases based on distributive properties.

Feature	Wide Column Store (Cassandra)	Document Store (MongoDB)	Key Value pair Store (Redis)	Graph Database (Neo4j)
Sharding and Partitioning	Auto sharding and order preserving	Built in and order preserving	Auto sharding and no order	Supports sharding but should be avoided
Scaling	Horizontal	Horizontal	Horizontal	Horizontal
Replication	Selectable Replication Factor	Master slave	Relaxed Master slave	Causal Clustering using Raft protocol (master slave)

In Cassandra vast quantity of data is divided across many nodes which imparts user with very high availability and without failure. It also supports horizontal scaling, selectable replication factor and cross data center replication.

Redis is designed for in-memory data using master-slave architecture. Categorically, Redis supports less strict practice of master-slave replication, wherein information from any master is easily replicated to whatever number of slaves, whereas a slave itself can act as a master to other slaves. This database doesn't partition data across nodes in an order preserving manner.

Subject to Neo4J scalability package is noted as high availability. It does not support partitioning and complete dataset is replicated across whole cluster.

IV. Conclusions

SQL databases are scale vertically (hardware) while the NoSQL databases are horizontally scalable (server). This paper has the aim of giving a thorough overview and introduction of NoSQLs, which have recently emerged in the market as an alternative to predominant relational database management systems. The first half discusses the motives and rationales behind the development and usage of non-relational management systems, while the next half categorizes NoSQLs into types, namely, Document stores, Key-value stores and Column based stores, and then elucidate on their models and workings. Each database performs and behaves in a different manner and all of them are constantly evolving. The current research has taken databases from each category that is Cassandra (wide column store), Neo4j (Graph database) Redis (Key value pair store) and MongoDB (Document store) and compared them on the basis of data models, distributive properties and other features. The research compares them on their non-functional features. It has been found that for a simple data that can be represented in the form of key value easily, a key value store should be chosen as it will provide high performance, scalability and flexibility. If the value can be represented in column form, and is semi structured, then column store is the appropriate database as it will provide high performance and scalability. If data can be represented in JSON format, document store should be preferred as it has high performance, flexibility and usually high scalability. If the graph theory represents the data then we use graph store model which provides us high stability, but performance and scalability is variable. Following this, the comparison is made on the basis of functional features. It has been concluded that Key value store ought to be avoided if one needs to use composite key, joins or derived table operations on the database. Document Store ought to be avoided if one needs to use de-normalization, unordered key, composite key, composite aggregation, joins or derived table operations on the database. Wide column store should be avoided if we want to use unordered keys, aggregation and group by, adjacency lists, nested sets or joins operations on the database. Graph Store should be used if we want to perform just de-normalization. Redis is not optimized for maximum security [43] but for maximum performance and simplicity. Stonebraker [41] considered various performance arguments in support of NOSQL databases and observed them insufficient. Thus, these systems have various limitations also.

V. REFERENCES

[1] AvriliaFloratou, Nikhil Teletia , David J. DeWitt, Jignesh M. Patel, Donghui Zhang, "Can the Elephants Handle the NoSQL Onslaught?", Proceedings of the VLDB Endowment, 2012

[2] Tilmann Rabl, Mohammad Sadoghi, Hans-Arno Jacobsen, Sergio G´omez Villamor, Victor Munt´es Mulero and Serge Mankovskii, "Solving Big Data Challenges for Enterprise Application Performance Management", The 38th International Conference on Very Large Data Bases, August 27th - 31st 2012, Istanbul, Turkey. Proceedings of the VLDB Endowment,2012

[3] Bogdan George Tudorica, Cristian Bucur, "A comparison between several NoSQL databases with comments and notes", RoEduNet International Conference 10th Edition: Networking in Education and Research, 2011

[4] Yishan Li and Sathiamoorthy, "A performance comparison of SQL and NoSQL databases.", IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, 2013

[5] Laurie Butgereit, "Four NoSQLs in Four Fun Fortnights: Exploring NoSQLs in a Corporate in a Corporate IT Environment", SAICSIT '16 Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists, 2016

[6] Avinash Lakshman, Prashant Malik, "Cassandra - A Decentralized Structured Storage System", Operating Systems Review, 2010

[7] John Klein, Ian Gorton, Neil Ernst, Patrick Donohoe, Kim Pham, Chriisjan Master, "A comparison between several NoSQL databases with comments and notes", PABS '15 Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems, 2015

[8] Jiri Schindler, "I/O performance of NoSQL Databases(VLDB)", SIGMETRICS '13 Proceedings of the ACM; SIGMETRICS/international conference on Measurement and modelling of computer systems, 2013

[9] Amal W. Yassien, Amr F. Desouky, "RDBMS, NoSQL, Hadoop: A Performance-Based Empirical Analysis", AMECSE '16 Proceedings of the 2nd Africa and Middle East Conference on Software Engineering, 2016

[10] Hua Fan, Aditya Ramaraju, Marlon McKenzie, Wojciech Golab, Bernard Wong, "Understanding the Causes of Consistency Anomalies in Apache Cassandra", Proceedings of the VLDB Endowment, 2015

[11] Sumitkumar Kanoje, Varsha Powar, Debajyoti Mukhopadhyay,"Using MongoDB for Social Networking Website: Deciphering the Pros and Cons", IEEE Sponsored 2nd International Conference on Innovations in Information Embedded and Communication Systems, 2015

[12] Zachary Parker, Scott Poe, Susan V. Vrbsky, "Comparing NoSQLMongoDB to an SQL DB", ACMSE '13 Proceedings of the 51st ACM Southeast Conference, 2013

[13] Gansen Zhao, Weichai Huang, Shunlin Liang, Yong Tang,"Modelling MongoDB with Relational Model", Fourth International Conference on Emerging Intelligent Data and Web Technologies, 2013

[14] Rick Cattell, "Scalable SQL and NoSQL Data Stores", SIGMOD Record, December 2010 (Vol. 39, No. 4), 2010

[15] Lanjun Wang, Shuo Zhang, Juwei Shi, Limei Jiao,Oktie Hassanzadeh, Jia Zou, Chen Wangz, "Schema Management for Document Stores", Proceedings of the VLDB Endowment, 2015

[16] http://www.planetcassandra.org/what-is-nosql/

[17] Concurrent Programming for Scalable Web Architectures (2012),Benjamin Erb Research Assistant, University of Ulm Distributed Systems (Cited by 14):http://berb.github.io/diploma-thesis/original/061_challenge.html

[18] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber, "Bigtable: A Distributed Storage System for Structured Data", ACM Transactions on Computer Systems,Volume 26, issue 2,Article 4, June 2008

[19] Seth Gilbert, Nancy Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services", ACM SIGACT NewsVolume 33 Issue 2, June 2002 ,Pages 51-59 http://dl.acm.org/citation.cfm?id=564601

[20] "Google Bigtable, Compression, Zippy and BMDiff". *2008-10-12*. *Archived from* the original *on 1 May 2013*. Retrieved 14 April 2015..

[21] Santosh S. Ravi, Kalyanaraman santhanam , "Performance of Neo4j versus MongoDB for social actions"; 2014

[22] Chad Vicknair, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen, Dawn Wilkins, "A Comparison of a Graph Database and a Relational", ACM SE '10 Proceedings of the 48th Annual Southeast Regional Conference; 2010

[23] DatabaseSQL Server by Microsoft: https://www.microsoft.com/en-us/sql-server/sql-server-2016

[24] Oracle Official website: https://www.oracle.com/database/index.html

[25] MySQL Documentation: https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html

[26] Definition of RDBMS: http://searchsqlserver.techtarget.com/definition/relational-database-management-system

[27] Neo4j Official Website: https://neo4j.com/developer/graph-database/

[28] TITAN's (a distributed graph database) Official website: http://titan.thinkaurelius.com/

[29] Redis Documentation: https://redis.io/documentation

[30] Amazon DynamoDB Official Page: http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html

[31] Riak Official Website: http://basho.com/products/

[32] Apache HBASE official website: https://hbase.apache.org/

[33] Cassandra Documentation by Datastax Academy: https://academy.datastax.com/resources/brief-introduction-apache-cassandra

[34] MongoDB Official Website: https://www.mongodb.com/what-is-mongodb

[35] Apache CouchDB Official Website: http://couchdb.apache.org/

[36] Brewer's CAP Theorem: http://www.royans.net/wp/2010/02/14/brewers-cap-theorem-on-distributed-systems/

[37] BSON Official Website and Documentation: http://bsonspec.org/

[38] JSON Official Website and Documentation: http://www.json.org/

[39] NoSQL Explanation by Datastax Academy:http://www.planetcassandra.org/what-is-nosql/

[40] Antonios Makrisa, Konstantinos Tserpesa, Vassiliki Andronikoub, Dimosthenis Anagnostopoulosa, "A classification of NoSQL data stores based on key design characteristics", Cloud Futures: From Distributed to Complete Computing, CF2016, 18-20 October 2016.

[41] Michael Stonebraker. 2010. SQL databases v. NoSQL databases. *Commun. ACM* 53, 4 (April 2010), 10-11. DOI: https://doi.org/10.1145/1721654.1721659.

[42] "MongoDB Security Guide Release3.2.1", MongoDB,Inc. February09, 2016,©MongoDB,Inc.2008–2015.

[43] IBM systems power solution for redis: https://www-304.ibm.com/partnerworld/wps/servlet/download/DownloadServlet?id=6sXLmWUMhJJiPCA$cnt&attachmentName=ibm_power_systems_solution_for_redis.pdf&token=MTQ1NTc4NTMxMTgzMA==&locale=en_ALL_ZZ

[44] Security: http://redis.io/topics/security/

[45] Neo4j Security, https://neo4j.com/docs/operations-manual/current/security/checklist/

VI. APPENDIX

Table 5: Summary of Related Work

S.No	Paper Title	Authors	Publication Details	Description
1	**Can the Elephants Handle the NoSQL Onslaught?**	Avrilia Floratou, Nikhil Teletia , David J. DeWitt, Jignesh M. Patel, Donghui Zhang	2012;Proceedings of the VLDB Endowment	Comparative study between SQL and NoSQL is done on interactive data serving and decision support analysis. With the help of this evaluation conclusions are drawn and future trends are speculated.
2	**Solving Big Data Challenges for Enterprise Application Performance Management**	Tilmann Rabl, Mohammad Sadoghi, Hans-Arno Jacobsen, Sergio G´omez Villamor, Victor Munt´es Mulero and Serge Mankovskii	2012; The 38th International Conference on Very Large Data Bases, August 27th - 31st 2012, Istanbul, Turkey. Proceedings of the VLDB Endowment	Various benchmark programs are run on DBMSs i.e. Cassandra, HBase, Voldemort, MySQL and Redis and the throughput and the latencies of operations like read write etc and create graphs for these comparisons
3	**A comparison between several NoSQL databases with comments and notes**	Bogdan George Tudorica, Cristian Bucur	2011; RoEduNet International Conference 10th Edition: Networking in Education and Research.	The paper explains the relevance of SQL and NoSQL databases in different environments. NoSQL database models are those databases in which fixed schemas are not required, scaling in done horizontally, joins are avoided and SQL interface isn't exposed.
4	**A performance comparison of SQL and NoSQL databases.**	Yishan Li and Sathiamoorthy	2013; IEEE Pacific Rim Conference on Communications , Computers and Signal Processing	This paper compares the performance on the basis of key-value store implementations on NoSQL and SQL databases. Various (CRUD) operations are performed on various NoSQL and SQL databases and drastic variations are recorded as a result, even among the NoSQL databases
5	**Four NoSQLs in Four Fun Fortnights: Exploring NoSQLs in a Corporate in a Corporate IT Environment**	Laurie Butgereit	2016; SAICSIT '16 Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists.	NoSQL databases are been divided into groups: key-value stores, columnar databases, graph databases, and document databases. One database has been picked from each category: Redis, Cassandra, Nep4j and MongoDB
6	**Cassandra - A Decentralized Structured Storage System**	Avinash Lakshman, Prashant Malik	2010; Operating Systems Review	This paper explains how Cassandra works. Cassandra is a distributed storage system which is used to manage a large amount of structured data
7	**A comparison between several NoSQL databases with comments and notes**	John Klein, Ian Gorton, Neil Ernst, Patrick Donohoe, Kim Pham, Chriisjan Master	2015 ;PABS '15 Proceedings of the 1st Workshop on Performance Analysis of Big Data Systems.	This paper describes methods and the corresponding results of a study that is conducted for the three selected NoSQL databases when we consider the scenario of the data of a large healthcare organization.
8	**I/O performance of NoSQL Databases**	Jiri Schindler	2013; SIGMETRICS '13 Proceedings of the ACM; SIGMETRICS/international conference on Measurement and modelling of computer systems.	The paper aims to evaluate the architecture of a few selected NoSQL databases to lay the foundations for understanding the functioning of the new emerging database systems. They base their work on an electronic healthcare record (HER) System where they were given use cases by a customer that he primarily wanted them to work on.
9	**RDBMS, NoSQL, Hadoop: A Performance-Based Empirical Analysis**	Amal W. Yassien, Amr F. Desouky	2016; AMECSE '16 Proceedings of the 2nd Africa and Middle East Conference on Software Engineering	This paper aims to provide an insight into choosingthe suitable data model by conducting a benchmark usingYahoo! Cloud Serving Benchmark (YCSB) on three types of database systems i.e., MySQL for RDBMS, MongoDB for NoSQL data model, and HBase for Hadoop framework.
10	**Understanding the Causes of Consistency Anomalies in Apache Cassandra**	Hua Fan, Aditya Ramaraju, Marlon McKenzie, WojciechGolab, Bernard Wong	2015;Proceedings of the VLDB Endowment	This paper studies the staleness of values returned by read operations applied to Cassandra which supports eventual consistency using quorum-based replication.
11	**Using MongoDB for Social Networking Website: Deciphering the Pros and Cons**	Sumit kumar Kanoje, Varsha Powar, Debajyoti Mukhopadhyay	2015; IEEE Sponsored 2nd International Conference on Innovations in Information Embedded and Communication Systems	This purpose focuses on the drawbacks and advantages offered by MongoDB, so that the developer can make a wise decision while choosing a database for a social networking website.
12	**Comparing NoSQLMongoDB to an SQL DB**	Zachary Parker, Scott Poe, Susan V. Vrbsky	2013; ACMSE '13 Proceedings of the 51st ACM Southeast Conference	A comparison has been made between one of the NoSQL solutions, MongoDB, to the standard SQL relational database, SQL Server. The performance, in terms of runtime, of these two databases for a modest-sized structured database has been compared
13	**Modelling MongoDB with Relational Model**	Gansen Zhao, Weichai Huang, Shunlin Liang, Yong Tang	2013; Fourth International Conference on Emerging Intelligent Data and Web Technologies	This paper explores the problems by modelling of MongoDB, with relational algebra. The dissimilarities of semantic expression powers have been highlighted between RDBMS and MongoDB.
14	**Scalable SQL and NoSQL Data Stores**	Rick Cattell	2010; SIGMOD Record, December 2010 (Vol. 39, No. 4)	This paper compares many SQL data stores against the various NoSQL data stores which are meant to scale simple DLTO-style application loads. The new data models sacrifice some dimensions like consistency, availability, high scalability etc.
15	**Schema Management of Document Stores**	Lanjun Wang , Oktie Hassanzadeh, Shou Zhang, Juwei Shi, Limei Jiao, Jia Zou, Chen Wang	2015; Proceedings of the VLDB Endowment	A schema management framework has been presented in this paper for document stores. The simplicity offered by JSON document stores can cause snags in certain database management tasks.
16	**Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services**	Seth Gilbert, Nancy Lynch	June 2002, ACM SIGACT News Volume 33 Issue 2	CAP theorem has been explained in this paper.
17	**MongoDB vs Oracle - database comparison**	Alexandru Boicea, Florin Radulescu, Laura Ioana Agapin	2012, Emerging Intelligent Data and Web Technologies (EIDWT), Third International Conference	This paper describes the distinctions we come across while comparing an SQL database i.e. Oracle database and NoSQL form of database i.e. document store database.
18	**Comparative Study of SQL & NoSQL Databases**	Supriya S. Pore, Swalaya B. Pawar	5, May 2015, International Journal of Advanced Research in Computer Engineering & Technology	This research paper aims to evaluate and compare these two SQL & NOSQL databases and examines which of these is better when it comes to performance and scalability. Index Terms— RDB
19	**NoSQL Databases: MongoDB vs Cassandra**	Veronika Abramova, Jorge Bernardino	2013, C3S2E '13 Proceedings of the International C* Conference on Computer Science and Software Engineering	This research paper revolves around NoSQL databases, their features and operational principles. It compares and evaluates two NoSQL databases i.e. MongoDB and Cassandra.
20	**Have Your Data and Query It Too: From Key-Value Caching to Big Data Management**	Dipti Borkar, Ravi Mayuram, Gerald Sangudi, Michael Carey	2016, SIGMOD'16 Proceedings of the 2016 International Conference on Management of Data	This paper explains the architectural alterations that are vital to be made to tackle the requirements of future generation applications which employ databases.