Evaluating the Performance of Some NoSQL Databases in the Storage and Management of Big Data

[1]Ogheneovo, E.E. & [2]Egwu, C.E.
Department of Computer Science
, University of Port Harcourt,
Port Harcourt, Nigeria
E-mail: edward.ogheneovo@uniport.edu.ng
Phone: +2348038591480

ABSTRACT

This paper discusses the performance evaluation of some NoSQL databases in the storage and management of big data with specific emphasis on MongoDB, Apache Casandra, HBase and CouchDB in terms of their flexibility and scalability as well as ability to handle both structured and unstructured data. The paper considered the time and speed performance between various NoSQL databases. Although, these might not be the only considerations in choice of database, they played significant roles in decision making process for large organization with very large records. The time recorded in each table represents the best time from the test conducted. From the tables below, when MongoDB is tested against Apache Cassandra, while the comparison in performance might be closely tied, Apache Cassandra out performed MongoDB for reads and MongoDB out performing Cassandra during write operations. One of the major advantages of Cassandra over MongoDB is the use of multiple masers nodes by Cassandra while MongoDB uses a single master node is control of slave nodes designed to replace the master node in the event of failure. Similar to MongoDB and Cassandra, HBase shows similarity when compared to Couchbase, though Couchbase shows slight advantage over HBase when making single reads, the advantage is shared among both databases when carrying out multiple reads respectively. The key/value data access of Couchbase offers a major advantage for the retrieval and access of data while HBase ability to easily scale to handle large data presents an advantage over Couchbase.

Keywords: NoSQL, RDBMS, Big Data, MongoDB, Apache Casandra, CouchDB, HBase

23rd iSTEAMS Conference Proceedings Reference Format

Ogheneovo, E.E. & Egwu, C.E. (2020): Evaluating the Performance of Some NoSQL Databases in the Storage and Management of Big Data. Proceedings of the 23rd iSTEAMS Conference, American University of Nigeria, Yola. April, 2020. Pp 97-108 www.isteams.net/yola2020

1. INTRODUCTION

The concept of big data is a common discussion among computer scientists and other computer users all over the world today. In fact, the term big data has become a buzzword used frequently when discussing information and communication technology these days due to the advent of the Internet technology arising from the development of Web 2.0 [1] [2]. There is hardly a time people will discuss information technology without mentioning big data. Rapid computing, simple access to Internet-based devices, steady growth in mobile networks, cloud-based infrastructure and new technologies have created an incomprehensibly big data world, described as "Big Data" [3]. Big data is a phrase used to mean a massive volume of both structures and unstructured data that is so large to the extent that they cannot be processed by traditional relational database management systems (RDBMS).

Big data are large data sets that can be analyzed computationally to reveal patterns, trends, associations, especially those that relate to human behavior and interactions. They can be structured, semi-structured and unstructured information being constantly updated in real time, making it difficult to manage or use standard data management instruments efficiently: e.g., relational database management systems (RDBMS) [4] [5]. In the simplest form, big data is employed in defining information that are semi-structured and unstructured in size and structure. These forms of data are larger than information deemed appropriate for periodic database handling scheme for storage or processing. This information are not organized to suite relational databases which consists of rows and columns [6] [7]. Big data is a filed that deals with the way data and information are systematically extracted from large datasets, analyzed, and used by researchers and data scientists. The massive increment in data offers likelihood for great scientific innovations, better business models, idea and better procedure for handling banking sector, food production, chain supply, merchandizing, health information management systems, etc. [8]. Big data has the potential to help organizations improve operations, make faster and intelligent decisions that can help organizations improve their revenue base [9].

NoSQL databases are based on the different models and each implementation within a category may have its own specifies [10]. One common feature of NoSQL databases is that the denormalized aspect of NoSQL database make query performance highly dependent on access paths. Also, there is no common query language for all NoSQL databases. They also have heterogenous data management systems, e.g., it is not all NoSQL databases that supports indexing. There are currently two JSON data types with PSQL, JSON and JSONB. They accept almost identical sets of values as input. The most significant difference between both is more of efficiency [11]. While the JSON data type stores an exact copy of the input text, JSONB stores a decomposed binary format. The trade-off of the binary format comes with the need to reparse on each execution, making it slower for each input due to the need for constant conversion, but significantly faster to process since the reparsing is not needed [12]

In this paper, we considered the time and speed performance between various NoSQL databases. Although, these might not be the only considerations in choice of database, they played significant roles in decision making process for large organization with very large records. The time recorded in each table represents the best time from the test conducted. From the tables below, when MongoDB is tested against Apache Cassandra, while the comparison in performance might be closely tied, Apache Cassandra out performed MongoDB for reads and MongoDB out performing Cassandra during write operations. One of the major advantages of Cassandra over MongoDB is the use of multiple masers nodes by Cassandra while MongoDB uses a single master node is control of slave nodes designed to replace the master node in the event of failure. Similar to MongoDB and Cassandra, HBase shows similarity when compared to Couchbase, though Couchbase shows slight advantage over HBase when making single reads, the advantage is shared among both databases when carrying out multiple reads respectively. The key/value data access of Couchbase offers a major advantage for the retrieval and access of data while HBase ability to easily scale to handle large data presents an advantage over Couchbase.

2. NOSQL DATABASES AND TYPES

NoSQL, or Not Only Sequel Query Language describes an extensive and progressively recognizable category of databases considered significantly different from relational databases. in that databases are not modelled on relations and tables and the use Sequel Query Language is excluded [13]. NoSQL databases are object-oriented databases intended to tackle processing problems caused by increasing information quantities and variety, especially in large information applications.

NoSQL databases are seen as very necessary in situations where amount of information is well beyond what relational database can handle and also structure of information is in manner that it is not stored in interactive database. No Sequel Query Language may aid Sequel-like query language therefore; it is Not Only SQL. Different types of NoSQL databases exist [14]. They are grouped into five categories. They are:

Key-Value Stores: Key-value stores s used for distributed index for the storage of unstructured data. These data or objects are typically not interpreted by the system. They are stored and handed back to the application as Binary Large Objects (BLOBs), such as memcached system. Another feature of the key-value stores is that it replicates data, which helps in recovery process whenever node is bad or malfunctioning. partitioning of the data over many machines, and object persistence. Examples are: memcached, Redis, Riak, ArangoDB, InfinityDB, Oracle NoSQL Database, OrientDB, Scalaris, and Project Voldemart [14]

Document Stores: Document stores is a NoSQL database system that stores unstructured and semi-structure data and objects or documents. It provides a simple query mechanism to search collections for objects with particular attribute values. document stores can also partition the data over many machines, replicate data for automatic recovery and persistent the data thus helping in recovery process whenever the need arises especially when one or more nodes in the network is down or malfunctioning. Document store is an extension of key-value databases where the value consists of a structured document.by organizing data in a similar way as Extensible Markup Language (XML), JavaScript object notation (JSON), and Binary JSON (BSON), which ensures one-to-one and one-to-many relationships in a single document. Thus a complex document can be retrieved or stored without using joint. Field indexes and advanced query features can also be defined in document store databases. Examples are: CouchDB, MongoDB, SimpleDB, and DynamoDB [15].

Graph Databases: Graph databases are used to store graphical data consisting of data interconnected with a finite number of relations between them [16]. These data could be social relations, links road maps, network topologies, etc. Examples include Neo4j, OrientDB, AllegroGraph, FlockDB, InfiniteGraph, Sqrrl Enterprise, etc. Examples include Neo4j, …. Graph databases are very useful for consumer and campaign surveys, fraud detection, and social network applications. Using graph databases, it is feasible to identify the group of products often bought together by the same customers.

Column-Family Databases: Column family databases [17] stores data in a persistent, sparse, distributed hash maps. This way, arbitrary keys (rows) are applied to arbitrary key value pairs (columns). These columns can be extended further with arbitrary key value pairs since these key-value pairs lists can be organized into column families and keyspaces. Also, column family stores can appear in a very similar shape to relational databases on the surface. Examples include HBase and Cassandra both influenced by Google BigTable. Others are Accumulo, Vertica, etc.

3. METHODOLOGY

Multiple tests are carried out, with the best results recorded in the tables below, the test involved multiple automated reads and writes on the various database shown below.

MongoDB

MongoDB is a cross platform MongoDB is a cross-platform, open – source, document – oriented database program that uses JSON – like documents with schema. MongoDB was developed by 10gen in 2007 [18]. It is written in C++ programming language. It supports APTs (drivers) in many computer languages e.g. JavaScript,

Python, Ruby, Perl, Java, Java Scala, C#, C++, HasKell, Erlang, etc. NOSQL Database used by many modern mode – based web applications to persist data. It is used for high volume of data storage each document can be different with a varying number of fields, size and contents [19]. Documents in MongoDB need not have schema, i.e., they are schema less since the field can be created any time MongoDB allows data to be represented hierarchically to store arrays and other more complex structure. MongoDB is very scalable (for the storage of document objects). It is very flexible and is adaptable to real business world situation and requirements. It stores data in document form as opposed to relational from in RDBMS. It allows the creation of indexing to improve the performance of searches within MongoDB. Any field in MongoDB can be indexed [20]. MongoDB can provide high availability of replica sets.

A replica set consists of two or more mongo DB instances. Each replica set member may act as primary secondary replica at any time. The primary replica is the main a server which interacts with the client and performs all the read/write operations while the secondary replicas maintain a copy of the data of the primary using built – in replication. When the primary replica fails, the replica set automatically switches over to the secondary and then it becomes the primary server. MongoDB uses shading to scale horizontally by splitting data across multiple MongoDB instances. MongoDB can run over multiple servers, balancing the load and/or duplicating data to keep the system up and running in case of hardware failure. Other features present in MongoDB include auto-sharding, support querying, has Map/Reduce functionality, etc. [21] [22]

Apache Casandra

Apache Cassandra [23] is a highly scalable, high-performance distributed database designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. It is a type of NoSQL database. Apache Casandra is a free and open – source distributed, wide column store, Apache Casandra is a NOSQL database management system designed to handle large amounts of data across many commodities serves, providing high availability with no single point of failure [24]. It was developed by Apache Software Foundation It was initially developed by Avinash La Kshman and Prashant Malik at Facebook in 2008 to power the Facebook inbox search feature. It is distributed, supports replication and multi data centre replication, scalability designed to have read and write throughput with no downtime or interruption to applications and supports fault – tolerant through automatic replication of data across multiple nodes for fault – tolerance.

Cassandra is wide – column store a hybrid between a key – value and a tabular database mgt system. Its date model is a partitioned row store with tunable consistency with row organized into tables. In Cassandra, tables can be created, dropped and even altered at run – time without blocking updates and queries. Column families contain rows and column which make it similar to RDBMS. Each row is unequal identified by a row key. Each row has multiple columns, each of which has a name, value, and a time step. However, unlike a table in an RDBMS, different rows in the same set of columns, and a column may be added to one or multiple rows at any time [25]. Each key in Cassandra corresponds to a value which is an object. Each key has values as columns, and columns are grouped together into sets called column families. Hence, each key identifies a row of a variable number of elements. These column families can be considered as tables. A table in Cassandra is a distributed multi-dimensional map indexed by a key. Also, applications can specify the sort order of columns within a super column or simple column family. Cassandra is Java – based. Cassandra is a distributed database management system designed for handling a high volume of structure data across commodity servers. Cassandra handles the huge amount of data with its distributed architecture. Data is placed on different machines with more than one replication factor that provides high availability and no single point of failure.

Cassandra is an open source, masterless architecture highly available, fault tolerance, scalable high performance, and affordable and low-cost database management system. Apache Cassandra is built on Amazori's Dynamo and Google's BigTable [25]. Apache Cassandra offers capabilities that relational databases and other NOSQL database cannot match such as continuous availability, linear scale performance, operational simplicity and easy data distribution across multiple data centres and cloud availability zones.

Apache Cassandra architecture is responsible for its ability to scale, perform, and offer continuous uptime. Rather than using a legacy master-slave or a manual and difficult – to – maintain sharded architecture. Apache Cassandra is scalable and is capable of handling large amounts of data and thousands of concurrent users or operations per second – ever across multiple data centres – just as it can also manage small amounts of data and user traffic. Therefore, since it is masterless or sharded it has no single point of failure. Thus it is capable of offering true continuous availability and uptime simple by adding new nodes to an existing cluster without having to take it down. It offers great performance and scalability without sacrificing availability [26] [27].

Apache HBase

Hbase is an open source, scalable, distributed, NoSQL big data store that runs on a Hadoop cluster database modeled after Google's BigTable. HBase is a column-oriented non-relational database management system that runs on top of Hadoop Distribute File System (HDFS), providing BigTable-like capabilities for Hadoop. Apache HBase can host very large tables – billions of rows, millions of columns – and can provide real-time, random read/write access to Hadoop data [28]. HBase is a wide-column data store. Each table is defined with rows and columns and have a primary key which provides access to each table in the database. HBase provides high-performance coordination in the database management systems using a query engine for processing of big data, to enable fault-tolerant big data application thereby ensuring that data are replicated across multiple nodes and data recovery in case of node failure or one or more nodes malfunctioning.

HBase allows many attributes to be grouped together into column families such that the elements of a column family are all stored together with predefine schemas and specify the column families. Also, new columns can be added to families at any time, making the schema flexible and able to adapt to changing application requirements. The database interface to the proprietary Google File System. HBase provides a fault-tolerant way of storing sparse data sets, which are common in many big data use cases. It is well suited for real-time data processing or random read/write access to large volumes of data [29].

CouchBase

CouchBase server is an open – source distributed, multi – model, NOSQL document – oriented database software package that is optimized for interactive applications. It is a decentralized database system and has a shareless architecture [30]. The applications in CouchBase may serve many concurrent users by creating, storing, retrieving, aggregating, manipulating and presenting data. CouchBase server is designed to provide easy – to – scale key – value or JSON document access with low latency and high sustained throughput. CouchBase is designed to be clustered from a single machine to very large – scale deployments spanning many machines. CouchBase server is client protocol compatibility, persistence, data replication, live cluster recognition, rebalancing and multi latency with data partitioning.

CouchBase is a NOSQL document database for interactive web applications. It has a flexible data model, easily scalable provides consistent high performance and is capable of serving application data with 100% uptime. CouchBase is the merge of two popular NOSQL technologies. MemBase and CouchDB. Membase provides persistence, replication, sharding to the high performance memcached technology which CouchBase provides the document – oriented capability based on JSON [31]. Basically, CouchBase is very flexible since it has the JSON documents for easy flexible with no downtime. It is also easily scalable both within a cluster of servers and between clusters at different data centres.

It is possible to add additional instances of CouchBase server to address additional user and growth in application data without any interruptions or changes in application code. Clusters of CouchBase is very efficient in handling additional workload and keep data evenly distributed. It also provides automatic sharding of data and rebalancing at run time for easy resizing of server cluster on demand [32].

CouchBase provides easy developer integration with different programming languages. It also provides a query mechanism for retrieving data where the client provides a query and the view through indexing. CouchBase also provide a key-based update mechanism where the client sends an updated document with the key. CouchBase also provide consistent high performance for massively concurrent data use and consistent high through in micro – second response time which help ensure a good experience for application users. Through consistent and high data throughput, CouchBase server allow more users with fewer servers to be connected. It also helps to balance load automatically across all servers to ensure consistent performance.

Table 1 provides a summary of the of the main features of MongoDB, Apache Cassandra, HBase and CouchBase.

Table 1: Summarizes the basic features of the NoSQL databases.

	MongoDB	Apache Cassandra	HBase	CouchBase
Language	C++	Java	Java	Erlang
License	AGPL	Apache	Apache	Apache
Database Model	Document-oriented	Column-oriented	Column-oriented	Document-oriented
Protocol	BSON	TCP/Thrift	HTTP/REST or TPC/Thrift	HTTP//REST
Storage	Memory Mapped B-Trees	MemTable/	HDFS	COW B-Tree
Data Model	JSON	Google BigTable	BigTable	Dynamo
Schemaless	Yes	Yes	Yes	Yes
MapReduce	Yes	Yes	Yes	
Indexing	Yes	Yes	Yes	Yes
Replication	Yes	Yes	Yes	Yes
High Availability	Yes	Yes	Yes	Yes
Auto-Sharding	Yes	No	Yes	Yes
Fast In–Place Updates	Yes	Yes	Yes	Yes
Supports Ad hoc Querying	Yes	No	**Yes**	Yes
Supports Complex Joins and Sub-queries	No	No	No	Yes
Scalability	Yes	Yes	Yes	Yes
Distributed	Yes	Yes	Yes	Yes
Supports denormalization	No	Yes	Yes	Yes
Column Partitioning	No	Yes	Yes	Yes
Secured	Yes	Yes	Yes	Yes
Automatic Fail-over	Yes	Yes	Yes	Yes
Capped Collection	Yes	No	Yes	No
Support Multiple Storage Engine	Yes	Yes	Yes	Yes
File Storage and Load Balancing	Yes	Yes	Yes	Yes
Multiple Server	Yes	Yes	Yes	No
Transaction Support	Yes	Yes	Yes	No
Tunable Consistency	No	Yes	Yes	Yes
Sharding	Yes	Yes	Yes	Yes
Procedural	No	Yes	No	Yes
Duplication of Data	Yes	Yes	Yes	Yes
Aggregation Framework	Yes	No	Yes	Yes
Peer-to-Peer Architecture	No	Yes	No	Yes
Open Source	Yes	Yes	Yes	Yes
Master-Slave Model	Yes	No	Yes	Yes
Durability	Yes	Yes	Yes	Yes
Primary Key	Yes	Yes	No	Yes
Foreign Key	No	No	No	No

4. RESULTS AND DISCUSSION

Table 2 presents time-based performance comparison between various NoSQL databases. While speed might not be the only consideration in choice of database, it plays a crucial role in decision making for large organization having to sort through thousands or millions or records if not billions of records in a single instance. The time recorded in each table represents the best time from the test conducted. MongoDB, Apache Cassandra, HBase and CouchBase are tested and compared based on time of data interaction among these NoSQL databases. The results provide the best time for reading and writing both structured and non-structured data. From the results it is possible to observe the speed at which data is being read and written on the databases compared below. From the tables you will also discover that more time is spent writing to disk than reading from disk.

While the read and write speed difference is negligible to most regular users, a few microseconds can serve as the deciding factor for enterprise industries interacting with big data. Analysis is very important as a phase in system development life cycle where real data are collected, understood and processed to identify problems and recommend solutions in order to improve the functionality of the system. This is carried out with the aim of producing new ideas and sharing knowledge to the development of new concepts and encourage new approach to data manipulation. Multiple unit testing tools were used and the best result recorded, NoSQL Unit, Mockup and DBWatch Database Control were used to monitor and test performances of the different database system, even among database not listed her.

While the table shows results for multiple reads and writes of all databases again structured and unstructured data and it can clearly be seen that there is a close performance tie among all four databases. However, other factors should be put into consideration when selecting best database to store big data. Most of the currently trending programming languages are fully supported by all four databases, MongoDB still offers a vast array of programming languages when compared to the other three NoSQL databases, with HBase offering the least support for available programing languages.

Cassandra and HBase offer wide column store model while Couchbase and MongoDB make use of the document store model. Both offer various advantages, document-based database offers a flexible structure that is not strictly bound by the data, allowing the database to be rearranged. Wide column store also known as the two-dimensional key-value store offer the ability to store data separately in various column reducing the load for data retrieval.

Table 2: Comparison of time of data interactions between

TIME OF DATA INTERACTION	MongoDB (seconds)	Apache Cassandra (seconds)	HBase (seconds)	CouchBase (seconds)
Structured data (Single Read) 1000	5.2	5.12	9.22	8.12
Unstructured data (Single Read) 1000	6.1	5.72	10.21	7.72
Structured data (Multiple Read) 2000	19.6	13.92	19.6	23.92
Unstructured data (Multiple Read) 2000	22.3	18.83	22.3	28.83
Structured data (Single Write) 1000	10.42	19.79	19.42	15.79
Unstructured data (Single Write) 1000	19.21	15.38	19.21	15.38
Structured data (Multiple Write) 2000	27.31	24.22	21.31	19.22
Unstructured data (Multiple Write) 2000	27.22	27.44	25.22	24.44

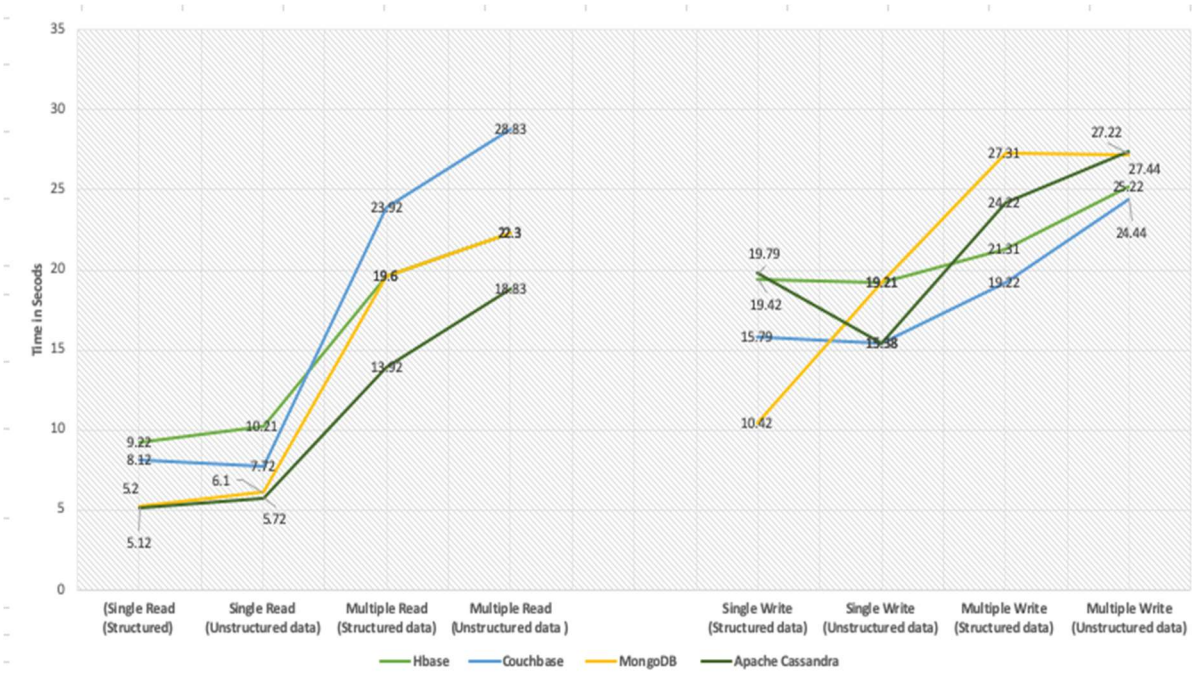

Fig. 2: Graph showing comparison of time of data interactions between MongoDB, Cassandra, HBase and CouchBase

The results from table 2 are plotted as shown in the graph in figure 2. The results form the graph shows that MongoDB is the faster and more scalable than the other NoSQL databases.

5. CONCLUSION

This study discusses the option of achieving NoSQL database flexibility and scalability as well as the stability and transactional components of a relational database, while maintaining a single database management system. The study provides an enhanced approach to storing and managing data using a geodatabase, contributing to further research into alternative way of handling big data. NoSQL, or Not Only Sequel Query Language describes an extensive and progressively recognizable category of databases considered significantly different from relational databases in that databases are not modelled on relations and tables and the use Sequel Query Language is excluded. NoSQL databases are object-oriented databases intended to tackle processing problems caused by increasing information quantities and variety, especially in large information applications. NoSQL databases are seen as very necessary in situations where amount of information is well beyond what relational database can handle and also structure of information is in manner that it is not stored in interactive database.

REFERENCES

1. Messaoudi, C., R. Fissoune and H. Badir (2018). A Performance Evaluation of NoSQL Databases to Mananage Proteomis Data. Int'l Jourrnal of Data Mining and Bioinformatics, Vol. 2, No. 1, pp. 70-89.
2. Zhang, L., Z. Sang, X.-Y. Ye, Y.-H. Shen, J. Tan and M.-Z. Bai (2016). Efficient Approach to Store Big Proteonic Data Based on NoSQL Database. In Proceedings of the Int'l Conference on Compute Science and Technology (CST'16). World Scientific, pp. 641-649.
3. Shao, B. and T. Courad (2015). Are NoSQL Data Stores Useful for Bioinformatics Researchers? Int'l Journal on Recent and Innovation Trends in Computing and Communication, Vol. 3, No. 3, pp. 1704-1708.
4. Zhang, Y., R. Zhang, Q. Chen, X. Gao, R. Hu, Y. Zhang and G. Liu (2012). A Hadoop-Based Massive Molecular Data Storage Solution for Virtual Screening. In Proceedings of the 7th China Grid Annual Conference (ChinaGrig), 2012, IEEE, pp. 142-147.
5. Abramova, V., J. Bemandino and P. Furtado (2014). Which NoSQL Databases? A Performance Overview. Open Journal of Database, Vol. 1, No. 2, pp. 17-24.
5. Mpinda, S.A.T., L. C. Ferreira, M. X. Riberio and M. T. P. Santos (2015). Evaluation of Graph Databases Performance Through Indexing Techniques. Int'l Journal of Artificial Intelligence & Applications, Vol. 6, No. 5, pp. 87-98.
6. Köhler, J., J. Baumbach, J. Tanbert, M. Specht, A. Skusa, A. Rüegg, C. Rawlings, P. Vèrrier and S. Philipi (2006). Graph-Based Analysis and Visualization of Experimental Results with Ondex, Bioinformatics, Vol. 22, No. 11, pp. 1383-1390.
7. Ameri, P., U. Grabowski, J. Meyer and A. Strait (2014). On the Application and Performance of MongoDB for Climate Satellite Data Trust, Security and Privacy. In Proceedings of 2014 IEEE 13th Int'l Conference on Data Trust in Computing and Communications (TrustCon),
8. Moniruzzaman, A. and H. S. Alkhter (2013). NoSQL Database: New Era of Databases for Big Data Analytics – Classification, Characteristics, and Comparison. Int'l Journal of Database Theory and Application, Vol. 6, No. 4, pp. 1-14.

9. Atzeni, P., C. S. Jensen, G. Orsi, S. Ra, L. Tanca and R. Torlone (2013). The Relational Model is Dead, SQL is Dead, and I Don't Feel So Good Myself. ACM SIGMOD Record, Vol. 42, No. 2, pp. 64-68.
10. Wang, G. and T. Jianfeng (2012). The NoSQL Principle and Basic Application of Cassandra Model. In Proceedings of Int'l Conference on Computer Science and Service System, 11-13 August, 2012, pp. 1323-1335.
11. Barmpas, K. and D. S. Kolovos (2014). Evaluation of Contemporary Graph Databases for Efficient Persistence of Large-Scale Models. Journal of Object Technology, Vol. 3, No. 3, pp. 3: 1-26.
12. Kahlenkamp, J. M. Klems and O. Ross (2014). Benchmarking Scalability and Elasticity of Distributed Database Systems. In Proceedings of the 40th Int'l Conference on Very Large Databases Endowment, September 1-5, 2014, Hangzhou, China, Vol. 7, No. 13, pp. 1219-1230.
13. Elif, D., S. Bedri and K. Pinar (2013). An Evaluation of Cassandra for Hadoop. In Proceedings of 2012 Int'l Conference on Cloud Computing, 28 June – 3July 2013, Santa Clara, California, USA.
14. Ogheneovo, E. E. (2019). NoSQL Databases: A Paradigm Shift in the Storage and Management of Data in Databases. In Proceedings of the 20th iSTEAMS Multidisciplinary Trans-Atlantic GoingGlobal Conference, KEAN University, New Jersey, USA, pp. 191-206
15. Gopinath, M. P., G. S. Tamilzharas, S. L. Aarthy and R. Mohenasandra (2017). An Analysis and Performance Evaluation of NoSQL Databases for Efficient Data Management in E-Health Clouds. Int'l Journal of Pure ad Applied Mathematics, Vol. 117, No. 2, pp. 177-197.
16. Olivera, F. R. and L. del Val Cura (2016). Performance Evaluation of NoSQL Multi-Model Data Stores in Polyglot Persistence Applications. In Proceedings of the 29th Int'l Journal of Database Engineering & Appliations Syposium, ACM, pp. 230-235.
17. Joulili, S. and V, Vansteanbarghe (2013). An Empirical Comparison of Graph Databases. Social Computing (SocialCom), 2013. Int'l Conference on IEEE, pp. 708-715.
18. An Introduction to MongoDB, https://www.sitepoint.com/an-introduction-to-mongodb/. Retrieved 27/02/2020.
19. GURU99 What is MongoDB? Introduction, Architecture, Features & Examples. https://www.guru99.com/what-is-mongodb.html. Retrieved 27/02/2020.
20. Durant, K. Introduction to NoSQL and MongoDB, Lesson 20 CS 3200, Northeastern University, http://www.ccs.neu.edu/home/kathleen/classes/cs3200/20-NoSQLMongoDB.pdf.
21. Chodorow, K. (2013). MongoDB: The Definitive Guide, O'Reilly, 2nd edition, http://usuaris.tinet.cat/bertolin/pdfs/mongodb_%20the%20definitive%20guide%20-%20kristina%20chodorow_1401.pdf.
22. MongoDB Release Documentation 3.0.4, MongoDB Documentation Project (2015), https://web.cs.wpi.edu/~cs585/s17/Books/Books-PDF/MongoDB-manual.pdf.
23. Wikipedia, Apache Casandra, https://en.wikipedia.org/wiki/Apache_Cassandra. Retrieved 28/02/2020
24. Guru99, Cassandra Tutorial for Beginners: Learn in 3 Days. https://www.guru99.com/cassandra-tutorial.html. Retrieved 28/02/2020.
25. Fedak, V. (2018). Top 5 Reasons to Use the Apache Cassandra Database, https://towardsdatascience.com/top-5-reasons-to-use-the-apache-cassandra-database-d541c6448557. Retrieved 28/02/2020.
26. DataStax Academy. What is Apache Cassandra? https://www.datastax.com/resources/definitions/what-is-apache-cassandra. Retrieved 28/02/2020.

27. Matloka, M. (2019). 7 Mistakes When Using Apache Cassandra. https://blog.softwaremill.com/7-mistakes-when-using-apache-cassandra-51d2cf6df519. Retrieved 28/02/2020.
28. Wikipedia, CouchBase Server, https://en.wikipedia.org/wiki/Couchbase_Server. Retrieved 29/02/2020.
29. Objeelia, A. Introduction to Couchbase – NoSQL Document Database, https://www.todaysoftmag.com/article/1506/introduction-to-couchbase-nosql-document-database. Retrieved 29/02/2020.
30. Mishra, S. (2018). Introduction to Couchbase: The Engagement Database, https://developer.rackspace.com/blog/introduction-to-couchbase-the-engagement-database/. Retrieved 29/02/2020.
31. Wikipedia, Apache HBase, https://en.wikipedia.org/wiki/Apache_HBase. Retrieved 2/3/2020.
32. GURU99, HBase Tutorial for Beginners: Learn in 3 Days, https://www.guru99.com/hbase-tutorials.html. Retrieved 2/3/2020.