Journal of Computer and Communications, 2020, 8, 59-71
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Comparative Study of Unstructured Data with SQL and NO-SQL Database Management Systems

Ahsan Malik*, Aqil Burney, Fawad Ahmed

Department of Computer Science CCSIS, Iobm, Korangi Creek Karachi, Pakistan
Email: *std_19231@iobm.edu.pk, std_18916@iobm.edu.pk, aqil.burney@iobm.edu.pk

How to cite this paper: Malik, A., Burney, A. and Ahmed, F. (2020) A Comparative Study of Unstructured Data with SQL and NO-SQL Database Management Systems. *Journal of Computer and Communications*, **8**, 59-71.
https://doi.org/10.4236/jcc.2020.84005

Received: February 3, 2020
Accepted: April 12, 2020
Published: April 15, 2020

Abstract

This paper aims to establish a relative study between a relational Microsoft SQL Server database and a non-relational MongoDB database within the unstructured representation of data in JSON format. There is a great amount of work done regarding comparison of multiple database management applications on the basis of their performances, security etc., but we have limited information available where these databases are assessed on the basis of provided data. This study will mainly focus on looking at all the possibilities that both these database types offer us when handling data in JSON. We will accomplish this by implementing a series of experiments while taking into consideration that the subjected data does not require to be normalized; and therefore, evaluate the outcome to conclude the result.

Keywords

SQL, NoSql, MongoDB, Databases, Unstructured Data, JSON

1. Introduction

In today's age, concurrent support for millions of users is an essential factor for the applications. These applications are also required to handle the massive amount of data. RDBMS has not been able to fulfill this requirement efficiently, which has resulted in the arrival of non-relational databases, or NoSQL, as they are publicly known. The RDBMS model features an inflexible schema which suggests that a "schema" should be designed before data is stored within, where all the specifications that define the characteristics of the schema, or the attributes, are uniform for all elements. Besides, they should allow for missing values to replace null values. RDBMS is generally known for their utility for

data that requires transactional reliability and/or can be normalized [1]. Non-relational databases do not require tables to store data. They employ a straightforward and simple data model, their schema dynamic; besides being able to handle unstructured data (documents, multimedia, e-mails, social media, etc.) proficiently. The unstructured data is usually populated in JSON or XML format, in which normalization is generally not required. It is noteworthy to know that with this type of data, when and why we should use an RDBMS model database application opposed to NoSQL or document-oriented database (e.g. SQL Server instead of MongoDB), as well as what will be the advantages and disadvantages [2].

Problem Statement

The problem we are presented with is to determine the more effective out of these two database types (SQL or NO SQL) while dealing with JSON, and what case were they considered to be more effective than the other. The aim is to present the results of the experiments; as well as the comparative study along with the scenarios in which these results have a greater bearing. It is essential to conduct a careful analysis and take into account the main factors amongst the pliability of schema, the amount of data, the number of resultant transactions and the budget once opting for the data model for the application.

2. Databases

A database is an assortment of data, information and knowledge stored in an organized manner so that it may be accessed, managed and updated. As existing data has new data added to it, deleted or expanded, this existing data gets modified. Database systems work by querying the information or data present, and then executing relevant applications against it (Creating and updating themselves) [3]. Computer databases typically contain combinations of data records or files, consisting of sales transactions, inventories and product catalogs. Generally, a DBMS manager provides users the authority to generate reports, control read/write operations, as well conduct an analysis of usage. As such, there are also databases that follow the ACID compliance (Atomicity, Consistency, Isolation, and Durability) to ensure that the transactions are complete and the data is compatible [4].

2.1. SQL (Structured Query Language)

SQL: Structured Query Language is a standard programming language employed in relational databases management and executing essential operations on their data. Originally formed in the 70s, SQL is the database administrators' most regular tool. It is also widely utilized for analytical queries; and writing scripts for data integration by data analysts and developers respectively [5]. SQL usage exemplify modifying index structures and database tables; which include adding, deleting and updating data as well as information retrieval or subsets of data

from a database for processing transactions as well as for analytics applications. SQL operations and queries are generally in command forms, written as statements. Insert, select, created, update, add, delete, truncate and alter are the frequently used SQL statements [6]. SQL (Structured Query Language) emerged in the later part of the 1970s and early 1980s and soon became the stock programming language for relational databases [7]. Relational database systems (also called SQL databases) consist of tables sets that contain data within columns and rows. Every individual table column represents a data categories (e.g. name, age, phone number, etc.), whereas each row comprises of the intersecting columns matching value of the data [8] (**Figure 1**).

The SQL structure is used to build both the types of the Relational Database Management Systems, proprietary and open source. These databases are broadly used by organizations. Some of the widely used are MS SQL Server, MySQL (owned by Oracle), Oracle Database, SAP Adaptive Server, Integrated Business Machines DB2, Systems Applications and Products (SAP) HANA, PostgreSQL, etc. [9]. Many of the database management systems are SQL centric with a specific extension to the programming and other relative functions standard language.

2.2. NoSQL (Not Only SQL Database)

This is an addition to database design. NoSQL deals with a wide assortment of data models that include graph, columnar, document, and key-value formats. "Not Only SQL", abbreviated to NoSQL, can be akin to a substitute to the traditional relational database approach. The traditional approach is using tables for storing data, and meticulously designing the schema. Then only can data be inserted within the database. When working with huge sets of distributed data, NoSQL database is the superlative platform [8]. A few databases that preceded the relational database management system are also to be considered as NoSQL, but this term usually refers to the database systems built in the earlier part of the 21st century to handle comprehensive handling of the clustering of the database in web and cloud applications. The requirements for performance and scalability overtake the SQL's instantaneous, inflexible data consistency [10] (**Figure 2**).

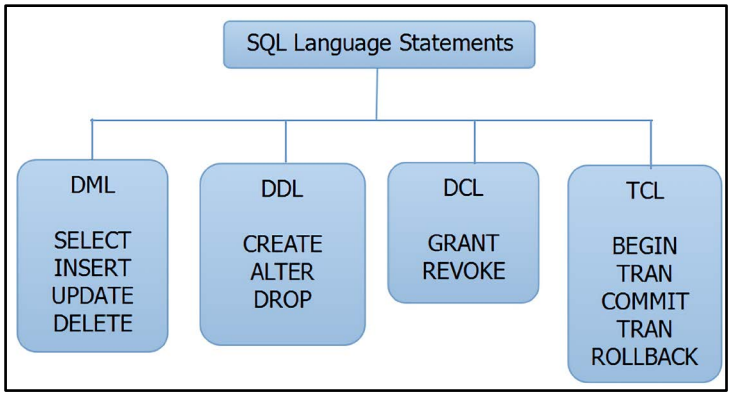


Figure 1. SQL breakup.

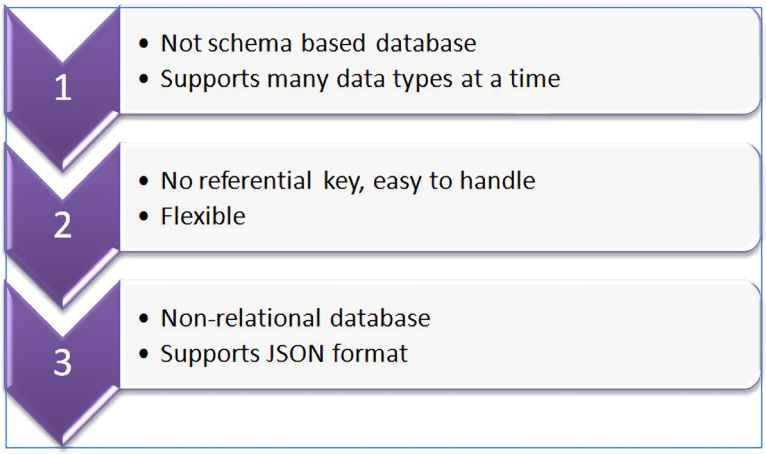


Figure 2. Properties of NoSQL databases.

The NoSQL, Not only SQL database system was not designed to adhere to relational schema rules. Google, Amazon and other large-scale web organizations like them used NoSQL databases for channeling their emphasis on small operating goals and assimilate relational databases of significant data consistency [11]. The earlier versions of NoSQL databases put their focus on specific characteristics of data management for web and cloud applications. The tendency to handle a significant amount of data's distribution swiftly across computing clusters was required in web and cloud design. To better enable fast changes on applications that were continuously updating, developers also went for designing flexible schema or no schema at all [12].

3. Database Management System (DBMS)

A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manipulate data [13]. A DBMS enables its end users to create, update, read and delete data in a database. The DBMS actually acts as an interface between the end users and database to ensure that data is consistently organized and remains easily accessible (**Figure 3**).

The DBMS manages three essential things: the data, the database engine that allows data to be obtained, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, standardized administration procedures, and data integrity [9]. The DBMS supports standard database administration tasks such as backup and recovery, change management and performance monitoring/tuning recovery. Many DBMS are effectively reliable for automatic restarts, rollbacks, and recovery, in addition to activity logging and auditing.

3.1. Structured vs. Unstructured Data

The "Structured Data" consist of plainly described types of data, which have a pattern that makes the search easy, like in text. On the other hand "Unstructured

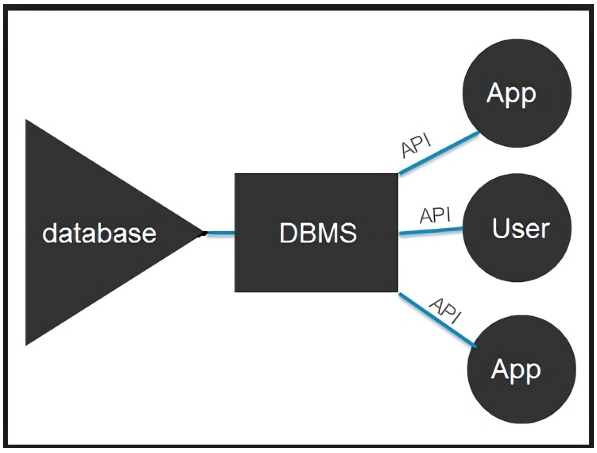


Figure 3. DBMS.

Data", is the type of data which usually cannot be searched easily, like in videos, audios and social media (Facebook, Twitter etc.) posts [14].

Customers choose the data types not based on their structure of data, but rather based on the software/applications that uses these data. Relational databases are usually referred for structured data and the remaining other type of application usually go for the unstructured data.

Nevertheless, a rising debate is going on, comparing the comfort of examination the data of structured format to the complex examination on the data of unstructured format. Analysis of data of structured format is an established course with proven technology. Whereas, analysis of unstructured data, on the other hand, is an evolving industry in its initial stages having a requirement of venture in R & D and is not considered a stabilized technology.

Table 1 outlines the individualities of each, as well as its possible usages, along with a few other features.

Position of Semi-Structured Data among Structured and Unstructured Data

Data form of semi-structured consists of built-in tags and markings that split data values, enabling grouping and hierarchies of information. Databases and documents both can be of semi structured data type. Data of this type only stand for about 5% to 10% of the structured, semi-structured or unstructured data ocean, but it does have significant commercial use [15].

An excellent case of a semi-structured data type can be an email. Though superior technology tools are required for analysis of thread tracking and concept searching; email's metadata provides classification and keyword searching without the need for supplementary apparatuses at all.

3.2. Extensible Markup Language: XML

XML, "Extensible Markup Language" is cast-off for denoting information in a specified format. The XML standard is a versatile way to create data formats and share structured data electronically via the Internet or corporate networks.

Table 1. Structured vs. unstructured data.

	Structured Data	Unstructured data
Characteristics	Pre-defined data models	No Pre-defined-data model
	Usually text only	Maybe text, image, sound, video or Other formats
	Easy to search	Difficult to search
Resides in	Relational databases	Applications
	Data warehouses	NoSQL databases
		Data lakes
		Data warehouses
Generatedby	Humans or machines	Humans or machines
Typical applications	Airline reservation systems	Word processing
	Inventory control	Presentation software
	ERP systems	Email clients
	CRM systems	Tools for viewing or editing media
Examples	Dates	Text files
	Phone numbers	Reports
	Social security numbers	Email messages
	Credit card numbers	Audio files
	Customer names	Video files
	Addresses	Images
	Product names and numbers	Surveillance imagery
	Transaction Information	

EXTENSIBLE MARKUP LANGUAGE code which is formally recommended by W3C World Wide Web Consortium, is like Hypertext Markup Language (HTML). Both EXTENSIBLE MARKUP LANGUAGE and HTML use markup symbols to narrate a page or file contents. HTML code denotes Web page content; mainly text and graphic images, only concerning how it is to be viewed and interacted with [14].

3.2.1. Difference between JSON and XML

Both these languages JSON and XML can obtain information from the web server. The below example for JSON and XML defines a worker's object, consisting of an array of 3 workers: [16]

JSON Example:

```
{"Workers":[
{"firstName":"Ahsan", "lastName":"Malik"},
{"firstName":"Charles", "lastName":"Crasto"},
{"firstName":"Fawad", "lastName":"Ahmed"},
]}
```

XML Example:

```
<workers>
<worker>
<firstName>Ahsan</firstName><lastName>Malik</lastName>
</worker>
<worker>
<firstName>Charles</firstName><lastName>Crasto</lastName>
</worker>
<worker>
<firstName>Fawad</firstName><lastName>Ahmed</lastName>
</worker>
</workers>
```

3.2.2. JSON Resembles XML on the Points

- JSON and XML both are human understandable hence called self-describing.
- Both follow hierarchical structure (values defined within values)
- Both provide the luxury be easily parsed and are used in a variety of coding languages
- Both can be called using the request "XMLHttpRequest" [17].

3.2.3. JSON Does Not Resemble XML as

- It doesn't make use of the end tag like XML.
- It has shorter syntax as compared to XML.
- Arrays can be used in JSON.
- It is much faster in writing and reading the data than XML [18].

And the most significant difference is that XML can only be parsed with help of an "XML parser" while JSON is parse-able with standardized JavaScript function.

4. MongoDB

MongoDB is a document-oriented data model, and is an "open source database" creation in the NoSQL universe. MongoDB is one of the many types of databases to appear after the 20th century [19]. What sets MongoDB apart is that rather than following the earlier tables-rows traditional concept that were the hallmark of relational databases, MongoDB model comprises of documents and collections. The documents, which are the basic unit of data in MongoDB database, are key-value set of pairs. Collections are the sets of functions and documents; and are considered as the equivalent to that of relational database tables [20] (**Figure 4**).

In line with the practice set by all NoSQL databases, MongoDB follows an extremely dynamic schema design. MongoDB lets the documents in a collection to hold different structures and fields.

5. Results of Experiments Conducted

To get a better picture in assessing and evaluating how both these databases handle data in JSON, we conducted a series of experiments.

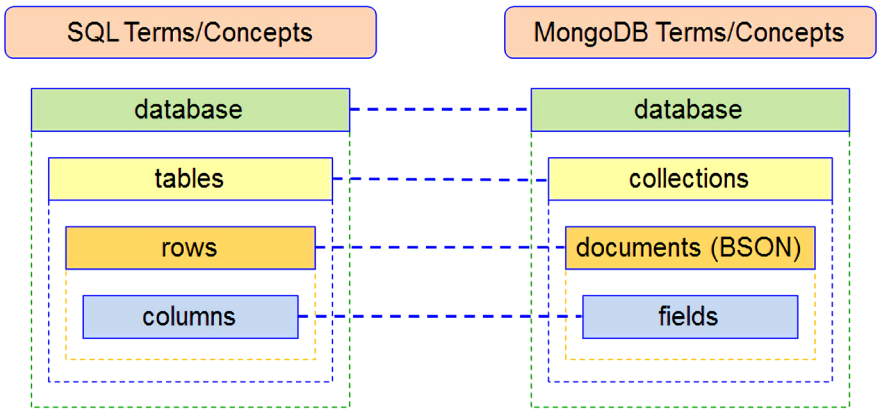


Figure 4. SQL vs. MongoDB terms.

Please note that whilst conducting these experiments, we took into consideration the fact that the subjected data did not require to be normalized.

Data was loaded in both databases for analysis, upon which we executed multiple types of experiments.

Data Collection used in the experiments

Data collection was done through an organization working on an evidence management system for a governmental security department. Hence, the data will not be publicly available due to its sensitivity and will only be utilized for research and experimentation purposes in the organization's premises.

Specifications of the computerused for the Data Analysis

The analysis of the data was performed by using Microsoft SQL Server 2014 and Mongo DB 3.6. The specifications of the computer used to run the experiments are shown in Table 2.

Experiment 1

In the first experiment we inserted 100,000 entries within both the RDBMS (SQL) and NoSQL (Mongo) database.

We used a similar insertion method in both databases for consistency, as different implementation methods could result in garnering an unfair advantage for either of the two.

The experiment was carried out ten times, Figure 5 presents the results of the analysis:

It can be easily deducted that out of the two databases used in the experiment, the fastest one is MongoDB, generally faster by fifteen seconds more than the SQL Server.

Out of the ten times we conducted the experiment, and even though the results of the 5th and 7th experiments were pretty even, but MongoDB emerged the fastest than SQL Server.

Experiment 2

For the second experiment, we decided to search for a random string ten times within both databases and compare the outcome of every iteration with each other. Figure 6 shows the startling results we obtained.

Table 2. Computer specifications.

Operating System	Windows 10 Pro
Processor	AMD Quad-Core A8-5545M APU@ 1.7 GHz
Installed memory (RAM)	16.00 GB
Disk	1 TB 5400 rpm SATA

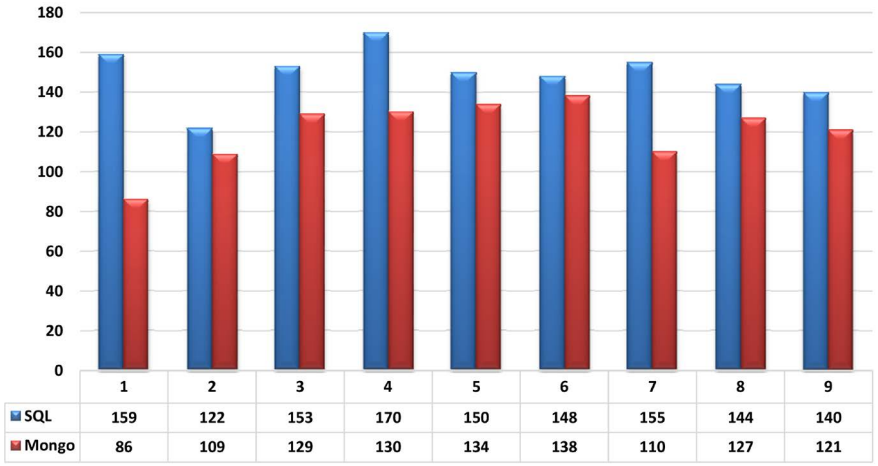


Figure 5. Result of experiment 1.

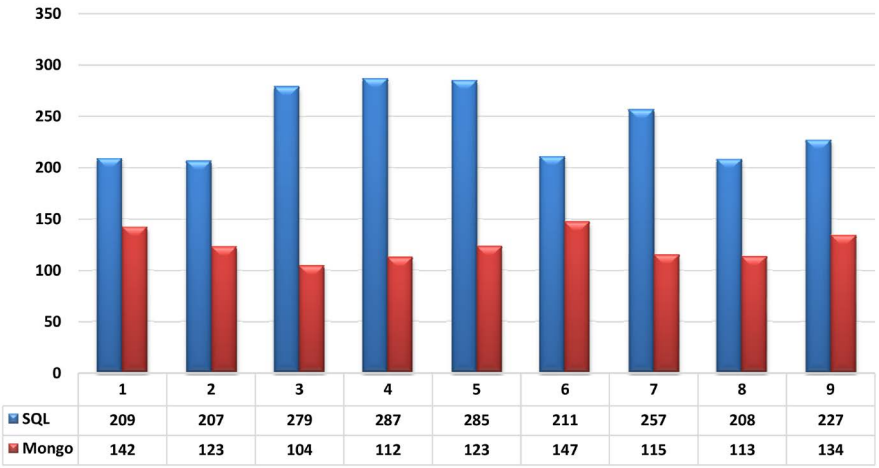


Figure 6. Result of experiment 2.

The outcome of these experiments was quite intense, to say the least. It can be seen that there is a huge difference between both the types of database.

We would go far out to say that searching for a particular occurrence of a string is just very efficient and extremely convenient to execute within MongoDB.

Experiment 3

How do these two databases fare whilst searching by a randomly generated ID?

This was the question behind our third experiment. Again, as in previous experiments, we executed the search 10 times on each to find our answer.

Primary Key is the SQL Server ID, whereas the default index in MongoDB is in the _id field (Figure 7).

Here the SQL Server proved to be extremely quick and efficient tool when searching by the ID field, which has a primary key. This was a superior result in favor of the SQL, as compared to that of the MongoDB.

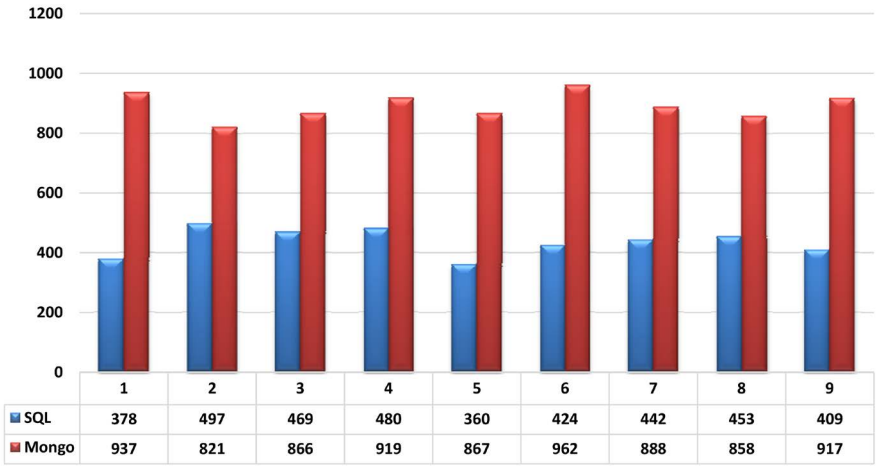
Experiment 4

The fourth experiment consisted of updating a randomly generated ID field. As was the pattern with the above three experiments, this exercise was conducted 10 times (Figure 8).

Again, we find that as has been the case in most of the experiments, the difference is tremendous and favorable in MongoDB when compared with SQL Server.

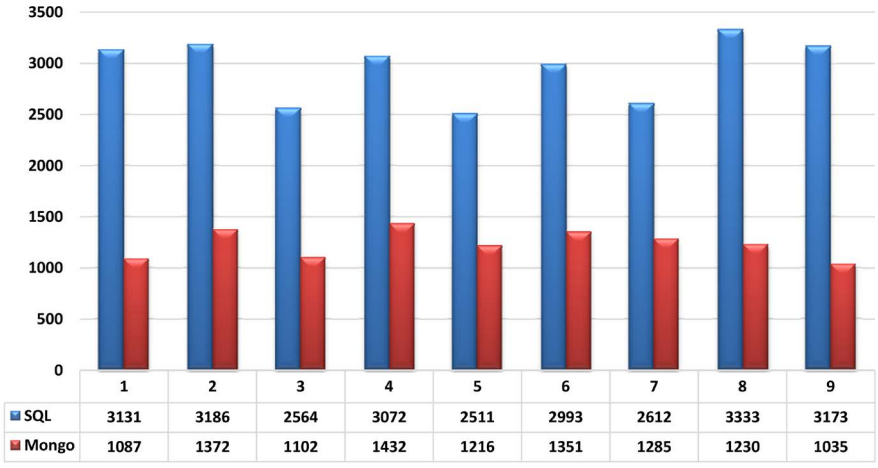
6. Conclusions

Whilst presenting all the data, and conducting the experiments, we were in fact striving to find and reach an answer to a simple, yet oft thrown around question.



	1	2	3	4	5	6	7	8	9
SQL	378	497	469	480	360	424	442	453	409
Mongo	937	821	866	919	867	962	888	858	917

Figure 7. Result of experiment 3.



	1	2	3	4	5	6	7	8	9
SQL	3131	3186	2564	3072	2511	2993	2612	3333	3173
Mongo	1087	1372	1102	1432	1216	1351	1285	1230	1035

Figure 8. Result of experiment 4.

How do we know which database type is best suited for our environment, and when should we prefer a relational database to NoSQL database, like MongoDB; and vise-versa?

So, even though the question we posed above may sound tedious and complicated, as one would imagine, the fact is that it really is not!!

The experiments conducted clearly reveal that the NoSQL should be an obvious choice of database. These databases do not place a limit on the types of data that can be stored together. They offer better scalability and are immensely advantageous for cloud storage and computing. Moreover, it is a simple way to keep the data comprehensible and clear.

During normalization, we often need many tables to store the data, and then we require several indexes so that we do not lose on the performance. The MongoDB database can be modeled on the same kind of structure, but better yet, we do it in such a manner that we can completely not require the need to use JOIN type operations. Doing so will not only radically and significantly increase performance but will also be considerably quicker than any other relational database.

If we look at this particular scenario, when we possess data where normalization is not an option, then it is a simple conclusion that when processing a substantial amount of data, the best answer is always MongoDB. We can go a step further and as far as a model of a single document data. This will still always prove to be effective and efficient than utilizing SQL Server database for unstructured data storage

7. Discussion Points

The inability of the RDBMS model to handle massive amounts of data efficiently, stemming from an inflexible feature of firstly designing a schema before data could be stored within it, gave way for the emergence of non-relational databases. But, this is a proven technology.

On the other hand, the simple characteristics of non-relational databases to handle unstructured data in a proficient manner, its low cost and easy scalability is making it a popular choice exceedingly, even though it is in its initial stages.

We need to conduct a thorough and proper analysis of the operations we will execute on our database. We will also need to study the data that we are going to be working with. As observed in the above experiments, Microsoft SQL Server brings us speed as well as transactional integrity whilst conducting JOIN operations. Whereas, the read and update speeds within MongoDB are exceptional.

Therefore, we must ask ourselves some questions.

Does our data have a rigid schema?

Will our data structure go through rigorous modifications? If it does indeed suffer severe modifications, then how flexible can we be if this indeed does happen?

Furthermore, in the event that our data cannot be normalized, we have to

question ourselves again, "Does the existing data within our relational database anyway relate to the data that will not be normalized?" If our answer is "Yes" then another question arises: Will the hybrid model be enough?

Therefore, it is extremely essential that we always identify the requirements of each application, as this is an important and vital cog.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Sanobar, K. and Vanita, M. (2013) SQL Support over MongoDB Using Metadata. *International Journal of Scientific and Research Publications*, **3**, 1-5. https://technet.microsoft.com/enus/library/bb522562(v=sql.105).aspx

[2] MongoDB for Giant Ideas. "Database References". https://docs.mongodb.org/manual/reference/database-references

[3] A Comparative Study of Databases with Different Methods of Internal Data Management https://pdfs.semanticscholar.org/75b1/c2b5bd593b7a47786c6cf3cb8593554aa746.pdf

[4] Jing, H., Haihong, E., Guan, L. and Jian, D. (2011) Survey on NoSQL Database. 2011 6*th International Conference on Pervasive Computing and Applications* (*ICPCA*), Port Elizabeth, South Africa, 26-28 October 2011, 363-366. https://doi.org/10.1109/ICPCA.2011.6106531

[5] Han, J., Song, M. and Song, J. (2011) A Novel Solution of Distributed Memory NoSQL Database for Cloud Computing. *Proceedings of the* 2011 10*th IEEE/ACIS International Conference on Computer and Information Science*, Sanya, 16-18 May 2011, 351-355. https://doi.org/10.1109/ICIS.2011.61

[6] Tudorica, B.G. and Bucur, C. (2011) A Comparison between Several NoSQL Databases with Comments and Notes. 2011 10*th Roedunet International Conference* (*RoEduNet*), Iași, Romania, 23-25 June 2011, 1-5. https://doi.org/10.1109/RoEduNet.2011.5993686

[7] Okman, L., Gal-Oz, N., Gonen, Y., Gudes, E. and Abramov, J. (2011) Security Issues in NoSQL Databases. 2011 *IEEE* 10*th International Conference on Trust, Security and Privacy in Computing and Communications* (*TrustCom*), 16-18 November 2011, 541-547. https://doi.org/10.1109/TrustCom.2011.70

[8] Wei-Ping, Z. and Ming-Xin, L. (2011) Using MongoDB to Implement Textbook Management System instead of MySQL. 3*rd IEEE International Conference on Communication Software and Networks* (*ICCSN*), Xi'an, China, 27-29 May 2011. https://doi.org/10.1109/ICCSN.2011.6013720

[9] Tauro, C.J.M., Aravindh, S. and Shreeharsha, A.B. (2012) Comparative Study of the New Generation, Agile, Scalable, High Performance NOSQL Databases. *International Journal of Computer Applications*, **48**, No. 20. https://doi.org/10.5120/7461-0336

[10] Edlich, S. (2011) List of NoSQL Databases. http://nosqldatabase.org

[11] Leavitt, N. Will NoSQL Databases Live Up to Their Promise?

[12] Strauch, C. NoSQL Databases. http://www.christofstrauch.de/nosqldbs.pdf

[13] Bhat, U. and Jadhav, S. (2010) Moving Towards Non-Relational Databases. *International Journal of Computer Applications*, **1**, No. 13. https://doi.org/10.5120/284-446

[14] Jatana, N., Puri, S., Ahuja, M., Kathuria, I. and Gosain, D. (2012) A Survey and Comparison of Relational and Non-Relational Databases. *International Journal of Engineering Research &Technology*, **1**, 1-5. https://doi.org/10.1142/9781848168701_0002

[15] MongoDB for Giant Ideas. "Database References". https://docs.mongodb.org/manual/reference/database-references

[16] Bulos, R.D., Bonsol, J., Diaz, R., Lazaro, A. and Serra, V. (2013) Comparative Analysis of Relational and Non-Relational Database Models for Simple Queries in a Web-Based Application. *Research Congress* 2013, de la Salle University Manila, 7-9 March 2013.

[17] XML Data Type and Columns (SQL Server). https://msdn.microsoft.com/en-us/library/hh403385.aspx

[18] Kriha, P.W. NoSQL Databases. https://www.christofstrauch.de/nosqldbs.pdf

[19] NoSQL Databases. https://nosqldatabase.org

[20] Raghu Ramakrishnan, J.G. (2002) Database Management Systems. McGraw-Hill, New York.