Comparative Study MongoDB vs. VoltDB

[1]Amany M. Al Luhaybi, [2*]Fahad AlQurashi

King Abdulaziz University, Jeddah, Saudi Arabia *Corr. Author: fahad@kau.edu.sa

Abstract: **New alternatives for relational database models have been introduced and highly recommended for big data applications, which are NoSQL and NewSQL database management systems. This paper contains a comparative study of MongoDB as an implementation of NoSQL database and VoltDB as an implementation of NewSQL database. The comparison was based on the execution time of the four basic operations: INSERT, DELETE, UPDATE and Query (select) operation. The Results show that MongoDB and VoltDB have a high performance in handling big data, but in this experiment VoltDB perform much faster than MongoDB for most of the operations.**

Keywords: **Comparison, MongoDB, NewSQL, NoSQL, VoltDB.**

1. INTRODUCTION

Nowadays, with the massive amount of data like the social media applications that their users alone could reach to millions, the demands to have an appropriate database that can accommodate big data is rapidly increasing. When dealing with data that is not as complex as today's, relational databases is a good option and have been used widely, but its performance is inefficient with the rapid increase with volumes, variety, and velocity of big data.

To overcome the limitation of the traditional relational database model, a new era of database model is introduced by Eric Evans which called NoSQL and stands for "Not Only SQL", Eric later described the motivation that led to NoSQL as "the whole point of seeking alternatives is that you need to solve a problem that relational databases are a bad fit for" [1]. It has many features which make it an efficient database model such as handling various types of unstructured data like documents and social media, reading and writing data in high speed, supporting mass storage, easy to expand and low cost [2]. NoSQL databases are classified based on the data model type each database is using as showing in Figure 1, which can be document, key-value, column family, and graph databases [3].

Another alternative for dealing with the massive data was developed in 2011, and called NewSQL which basically bring the relational data model into the world of NoSQL, in another words, the NewSQL support the relational model and use SQL as their query language but the support varies between the different types of the database belongs to this model [4]. The most suitable applications of this database model are those with large number of transactions in them [5].

Recently, with the existing of many databases determining the appropriate database is highly important, therefore, there is a need for performance evaluation and comparative study between different databases to determine which database will have a good performance, thus, they can be perfect for some applications. Here is a performance evaluation and a comparison between two databases one from NoSQL and another one from NewSQL. As for NoSQL MongoDB [6] was chosen as it is the most popular database in NoSQL, and it should powerful performance in many research papers as it will be discussed in the related work section later on. It is using the document data model, it does not a have the concept of a row, and the format these documents are stored in is called BSON, which offers us a binary representation of the JSON documents [7]. And for NewSQL the VoltDB [8] has been chosen, it offers fast response time but it has some restrictions like it is not allowing to use the *having* clause and not possible to join tables.

The organization of the paper is as follow: section 2. presents the related work that has been done, section 3. contains the evaluation method that is used to for this research, Section 4. is the experiment results and the comparison between the MongoDB and VoltDB, section 5 is the conclusion to conclude the research results and the future work.

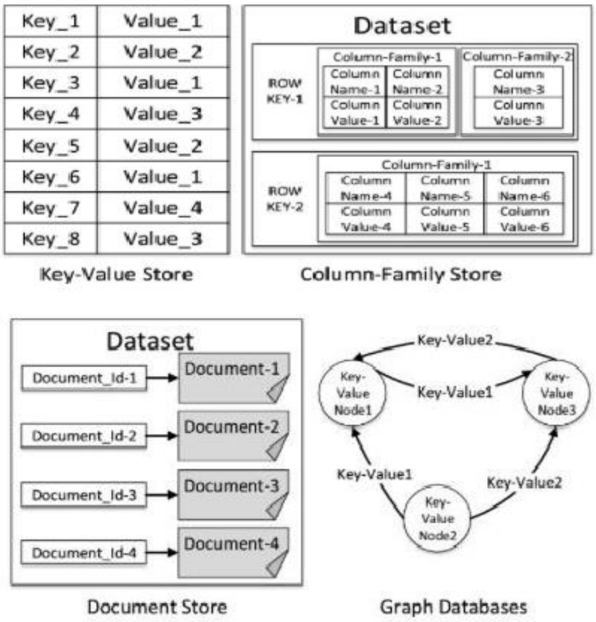


Figure 1: NoSQL data models types [5]

2. RELATED WORKS

All of the databases performance comparative studies that have been conducted was aiming to determine which database has a better performance and based on that for which kind of applications are these types most suitable. According to the previous related researches the MongoDB was compared to either another NoSQL database or to relational databases, what this research is aiming is to test MongoDB performance on different data sizes compared to a NewSQL database which is the VoltDB, and it is like MongoDB can handle the huge amount of data. In this section details information is given of the related researches and their results.

2.1 PostgreSQL and MongoDB

In [9] a comparison between PostgreSQL and MongoDB has according to the execution time for the four basic operations which are insert, select, update, and delete operations. These operations have been running for various data size like 30000, 90000, 150000, 210000 and 300000 data cases. The final result showed that MongoDB performed much better and faster than PostgreSQL.

2.2 MongoDB and MySQL

In [10] it shows a comparison between MongoDB and MySQL, the operations that been running were Insert, select, update, and delete, and the data size was 10,000. Therefore, MongoDB had less execution time in all operations, which is was expected as for large data volume non-relational database will perform better than relational database.

2.3 MongoDB vs. Cassandra

MongoDB and Cassandra have been compared based on the workloads in [11], by running tests and then measuring the time execution. The research findings are MongoDB had a low performance whenever the data size got increased. On the other hand, Cassandra performance was high whenever the data size got increased. To test the performance of read/update and run various workloads, the final result shoed that in update operation Cassandra had low execution time which make it performed much faster than MongoDB no matter what the data size was in this test, to sum the results up Cassandra had high performance compared to MongoDB in almost all test cases.

2.4 Cassandra, MongoDB and Couchbase

In [12] a performance comparison between three databases is done which are Cassandra, MongoDB and Couchbase, the result of this study showed that Couchbase had the best performance in reading then MongoDB comes next and then at last Cassandra. Moreover, for the writing processing Cassandra and Couchbase performed much better than MongoDB.

2.5 MongoDB to Oracle DB

The results in [13] showed that when comparing the performance of MongoDB and Oracle DB that if the data size is huge then as the previous studies shows, MongoDB has a better performance in retrieving the data, but when aggregation function is used in the testing Oracle DB performed better and had a faster speed than MongoDB.

3. THE EVALUATION METHOD

The primary goal of this paper is to make a comparison between the MongoDB which follows the NoSQL model and VoltDB an implementation of NewSQL database, based on the execution time of the basic operation INSERT, DELETE, UPDATE and query operations basically two select operations. To evaluate the performance of both databases, these operations were performed by using java programming language for both VoltDB and MongoDB to prevent different programming languages to affect the results of the experiment. To measure the execution time System.currentTimeMillis() function is used which measure the time from the beginning of running operation until its completed. The dataset that is used to conduct this research is a random data that was generated from a website in [14] this data has been created to fulfill the major requirement of this research which is basically a huge data volume. For VoltDB there are two tables one for users and the other one for orders, the schema is differ according to the size of the data. A sample schema of this data is shown below:

-User table schema:

CREATE TABLE USERS100 (USERID VARCHAR(10) NOT NULL, NAME VARCHAR(100),

EMAIL VARCHAR(200), CITY VARCHAR(50), PRIMARY KEY (USERID));

PARTITION TABLE USERS100 ON COLUMN USERID;

-Orders table schema:

CREATE TABLE Orders100 (

OrderID VARCHAR(10) NOT NULL, OrderNumber VARCHAR(10) NOT NULL,

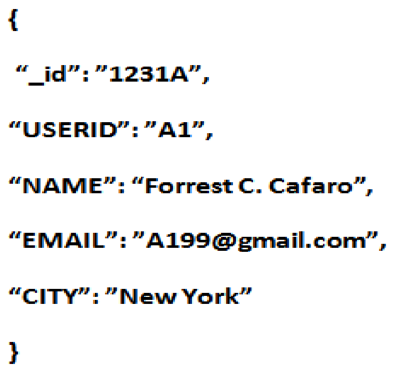
PRICE INT,

USERID VARCHAR(10) NOT NULL, PRIMARY KEY (OrderID),

FOREIGN KEY (USERID) REFERENCES USERS100(USERID));

As for MongoDB two collections were created one for users and another one for orders, then documents with various data sizes were inserted The data in MongoDB is in JSON format as it shown in Figure 2. for one document in the collection.

To evaluate MongoDB, the operations were tested by using MongoDB java Driver [15], imported in eclipse IDE [16].



```
{
 “_id”:”1231A”,
“USERID”:”A1”,
“NAME”:“Forrest C. Cafaro”,
“EMAIL”:”A199@gmail.com”,
“CITY”:”New York”
}
```

Figure 2: One document in MongoDB database.

For each of the basic operations and the query operations, various file sizes is tested which contain 100, 1000, 1000000, 2000000 data, and running them for five times for each operation in MongoDB or VoltDB then the average of these runs is computed and that will be illustrated in the figures that show the results of the experiments later on.

4. THE EXPERIMENTS RESULTS

The results might differ according to the characteristics of the computer the tests are performed on therefore, it should be highlighted that the following results were gained from computer with MacOS Sierra,

2.5 GHZ Intel Core i5 and 4 GB memory.

In the following subsections there are details of the operations and the results are demonstrated in each one of them.

4.1 Insert Operation

Insert operation was done on different data sizes 100, 1000, 1000000 and 2000000, for both VoltDB and MongoDB. The insertion in users collection (table in VoltDB) showed less execution time for VoltDB in all sizes as Figure 3. shows. For example, for the inserting of 2000000 the execution time in VoltDB was about 4.579 minutes and in MongoDB it was 9.339 minutes. However, the results differ for the insertion in orders collection (table in VoltDB) as it showed that MongoDB has less execution time for example in 2000000 MongoDB completed this in 6.413 minutes and the VoltDB in 17.672 minutes as it shown in Figure 4.

In general, in both databases we can see how the execution time got increased by the rapid increased in the data volume and both databases had high performance in handling this amount of data.

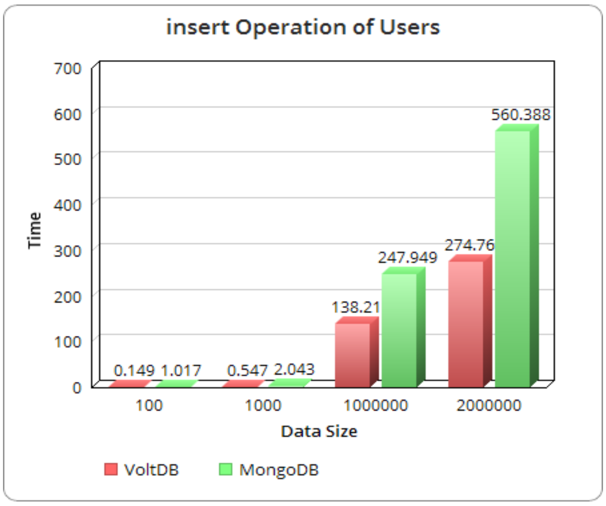


Figure 3: Comparing the performance of inserting in users at varies sizes (Time in seconds).

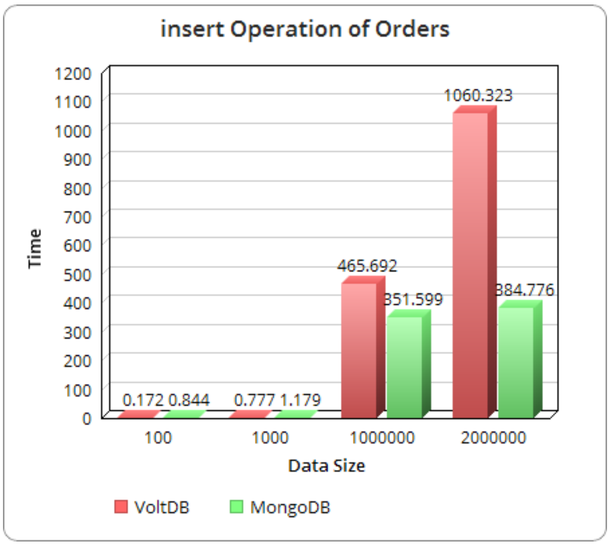


Figure 4: Comparing the performance of inserting in orders at varies sizes (time in seconds).

4.2 Update Operation

The update operation was done as well on all sizes 100,1000,1000000,2000000 for both databases, and it has the following same query to be tested:

-Change the email of the user Sharolyn R. Beuchert to sharolyn@hotmail.com

As Figure 5. Shows VoltDB has less execution time compared to MongoDB, as for 2000000 data size the updating happened in 0.456 seconds in VoltDB while in MongoDB was completed in 3.219 seconds. So for this operation VoltDB outperforms MongoDB.

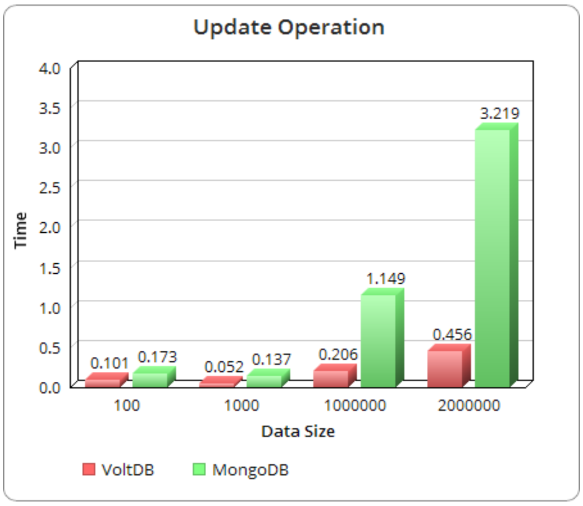


Figure 5: Comparing the performance of update operation at varies sizes (time in seconds)

4.3 Delete Operation

The delete operation was done as well on all sizes 100,1000,1000000,2000000 , and it has the following same query was done one both databases:

-Delete the user with name “Margert G. Wartman”. VoltDB has faster execution time compared with

MongoDB, as it is illustrated in Figure 6. for example, for 2000000 data size the deletion completed in VoltDB in 2.969 seconds while in MongoDB it is 5.609 minutes. This means VoltDB has high performance in Delete operation as well compared to MongoDB.

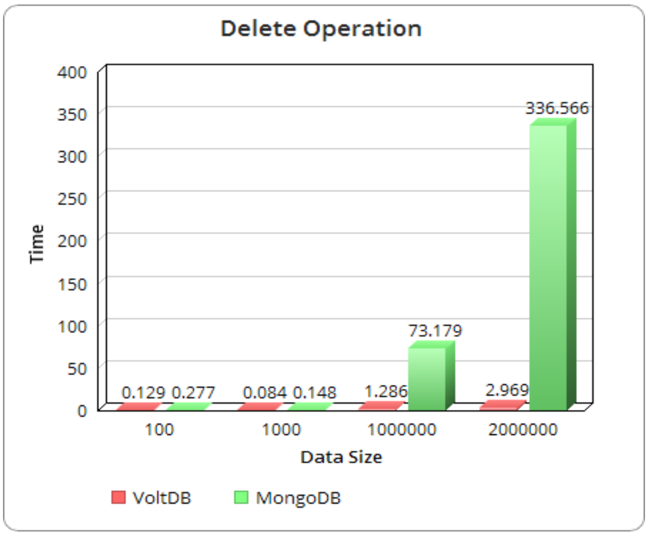


Figure 6: Comparing the performance of delete operation at varies sizes (time in seconds).

4.4 Query Operations

There were two queries that have been tested in both databases for all sizes 100, 1000, 1000000, 2000000. These two queries are as following:

-Query 1: List the number of users in each city

-Query 2: List all orders where the price is greater than 180 and less than 200.

The results illustrated in Figure 7. Show that VoltDB has faster execution time compared with MongoDB for query 1 , for example, for 2000000 data size the operation completed in VoltDB in 0.459 seconds while in MongoDB it is done in 2.293 seconds. This means VoltDB has high performance for query 1. However, for query 2 MongoDB has a clear superiority over VoltDB as it is shown in Figure 8. Looking at the results for 2000000 data size VoltDB completed the operation in 0.144 seconds, while MongoDB it was completed in 0.003 seconds.

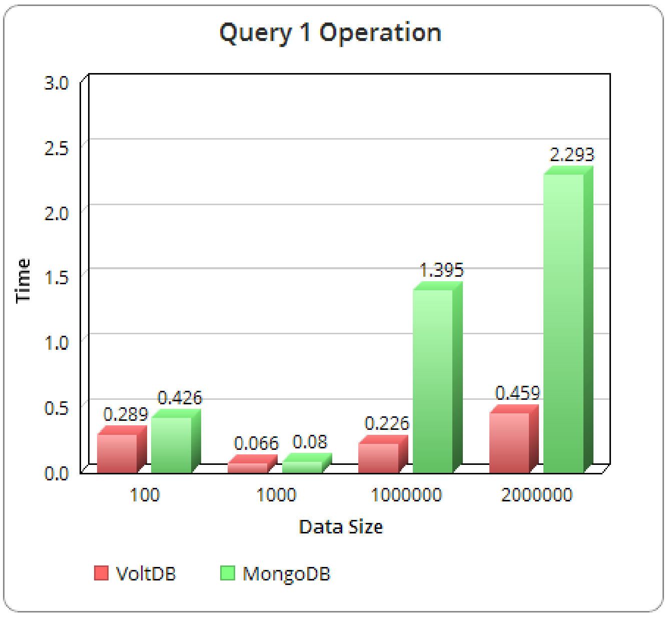


Figure 7: Comparing the performance of Query 1 operation at varies sizes (time in seconds).

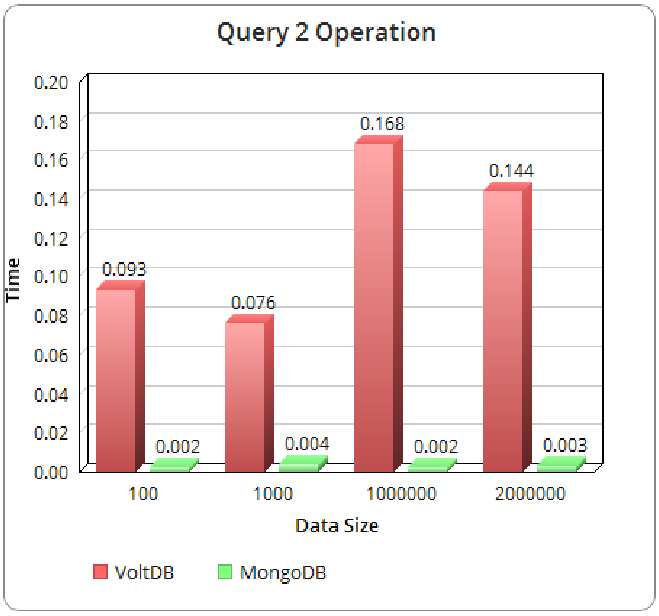


Figure 8: Comparing the performance of Query 1 operation at varies sizes (time in seconds).

5. CONCLUSION

Choosing the suitable database implementation of an application can be very challenging task, it highly depending of the requirements that needed to be fulfilled. However, the performance of the application is playing an important role in gaining the acceptance of it among its users.

NewSQL mainly uses the common language SQL for application interaction. And it is a good choice if the application has online transaction processing (OLTP).

NoSQL provides simple schema or schema-free which makes it flexible data store.

To conclude the results VoltDB showed in general faster execution time compared to MongoDB, However, MongoDB as well has high performance as well. Both of the databases showed that they can handle different operations on huge data volume in fast execution time.

In Future work, a real world application will be used, with meaningful database to be able to make a performance comparison between these two databases when a real transactions and real life queries is established.

REFERENCES

[1] Strauch, Christof, Ultra-Large Scale Sites, and Walter Kriha. "NoSQL databases." *Lecture Notes, Stuttgart Media University* (2011).

[2] Han, Jing, et al. "Survey on NoSQL database." *Pervasive computing and applications (ICPCA), 2011 6th international conference on*. IEEE, 2011.

[3] Leavitt, Neal. "Will NoSQL databases live up to their promise?." *Computer* 43.2 (2010).

[4] Grolinger, Katarina, et al. "Data management in cloud environments: NoSQL and NewSQL data stores." *Journal of Cloud Computing: Advances, Systems and Applications* 2.1 (2013): 22.

[5] Moniruzzaman, A. B. M. "Newsql: towards next- generation scalable rdbms for online transaction processing (oltp) for big data management." *arXiv preprint arXiv:1411.7343* (2014).

[6] MongoDB http://www.mongodb.org/ . Accessed 25 Mar 2017

[7] Győrödi, Cornelia, et al. "A Comparative Study of Databases with Different Methods of Internal Data Management."

[8] VoltDB Inc: VoltDB Technical Overview. 1–4. 2013.http://voltdb.com/downloads/datasheets_collateral/technical_overview.pdf . Accessed 25 Mar 2017

[9] Jung, Min-Gyue, et al. "A Study on Data Input and Output Performance Comparison of MongoDB and PostgreSQL in the Big Data Environment." *Database Theory and Application (DTA), 2015 8th International Conference on*. IEEE, 2015.

[10] Győrödi, Cornelia, et al. "A comparative study: MongoDB vs. MySQL." *Engineering of Modern Electric Systems (EMES), 2015 13th International Conference on*. IEEE, 2015.

[11] Abramova, Veronika, and Jorge Bernardino. "NoSQL databases: MongoDB vs cassandra." *Proceedings of the international C* conference on computer science and software engineering*. ACM, 2013.

[12] Băzăr, Cristina, and Cosmin Sebastian Iosif. "The transition from rdbms to nosql. a comparative analysis of three popular non-relational solutions: Cassandra, mongodb and couchbase." *Database Syst J* 5.2 (2014): 49-59.

[13] Faraj, Azhi, Bilal Rashid, and Twana Shareef. "Comparative study of relational and non-relations database performances using Oracle and MongoDB systems." *Int J Comput Eng Technol* 5 (2014): 11-22.

[14] City & Town Name http://www.mithrilandmages.com/utilities/CityNames.php Accessed 26 Mar 2017

[15] MongoDB Java Driver http://mongodb.github.io/mongo-java- driver/?_ga=2.97432024.617207923.1494729863-238380260.1491216126 . /. Accessed 26 Mar 2017

[16] Eclipse https://eclipse.org/ide/. Accessed 26 Mar 2017