A comprehensive comparison of SQL and MongoDB databases

Rajat Aghi, Sumeet Mehta, Rahul Chauhan, Siddhant Chaudhary and Navdeep Bohra

Department of Computer Science, Maharaja Surajmal Institute of Technology, Janakpuri, N.delhi 110058, India

Abstract- Since time immemorial, one of the most heavily needed and relied upon functionality of computers has been the memory. Although the technical aspects and the implementation methods may vary, most computers these days have the necessary hardware to process information and safe-keep it to be used in future as and when required. Database Management Systems (DBMS) are higher-level software programs that work with lower-level application programming interfaces (APIs) that take care of these operations. To help solving different kinds of problems, new kinds of DBMSs have been developed (e.g. Relational, NoSQL, etc.) along with applications programs implementing them (e.g. MySQL, PostgreSQL, MongoDB, Redis, etc). In this paper we aim at comparing both the database options for various CRUD operations namely Create, Read, Update and Delete for small and large datasets and then use this analysis to decide which database to use for any given set of data.

Index Terms- comparison between SQL and NoSQL, comparison between SQL and MongoDB, performance comparison of SQL and MongoDB, SQL or NoSQL.

I. INTRODUCTION

Database Management System is an umbrella term that refers to all sorts of completely different tools (i.e. computer programs or embedded libraries), with different and unique ways of working. These applications handle, or assist in handling and dealing with multiple sets of information at the same time. Since information (or data) can exist in various shapes and sizes, multiple DBMS' have been developed, along with numerous database applications, since the latter half of the 21st century to help in meeting different programming and computerisation needs. Database management systems are based on **database models**: structures defined for handling the information or data. Every new DBMS, along with the applications created to implement their methods, work in different ways with regards to definitions and storage and retrieval operations of the said information.

Although there are numerous solutions that implement different DBMS', each period in history has seen small amount of choices rapidly become extremely popular and stay in use for a long time, with probably the most predominant choice over the past couple of decades being the **Relational Database Management Systems** (RDBMS).

The remainder of the paper is organised as follows:

Section 2: An overview of the existing database models

Section 3: Comparative analysis between SQL and MongoDB for various data sets

Section 4: Conclusion based on the performed analysis

II. OVERVIEW OF THE EXISTING DATABASE MODELS

Each database system implements a different database model to logically structure the data that is being managed. These models are the first step and the biggest determiner of how a database application will work and handle the information it deals with. There are quite a few different types of database models which clearly and strictly provide the means of structuring the data, with most popular probably being the Relational Model. Although the relational model and relational databases are extremely powerful and flexible - when the programmer knows how to use them, for many, there have been several issues or features that these solutions never really offered. Recently, a series of different systems and applications called NoSQL databases started to gain popularity, expeditiously, with their promise of solving these problems and offering some very interesting additional functionality.

2.1 *The relational model*

Introduced in 1970s, the relational model offers a very mathematically-adapt way of structuring, keeping, and using the data. It expands the earlier designs of flat model, network model, et cetera by introducing means of relations. Relations bring the benefits of group-keeping the data as constrained collections whereby data-tables, containing the information in a structured way (e.g. a Person's name and address), relates all the input by assigning values to attributes (e.g. a Person's ID number).

Thanks to decades of research and development, database systems that implement the relational model work extremely efficiently and reliably. Combined with the long experience of programmers and database administrators working with these tools, using relational database applications has become the choice of mission-critical applications which cannot afford loss of any information. Despite their strict nature of forming and handling data, relational databases can become extremely flexible and offer a lot, granted with a little bit of effort.

2.2 *The model-less (NoSQL) approach*

The NoSQL way of structuring the data consists of getting rid of the constraints imposed by the relational model, hence liberating the means of keeping, querying, and using information. NoSQL databases, by using an unstructured (or structured-on-the-go) kind of approach, aim to eliminate the limitations of strict

relations, and offer many different types of ways to keep and work with the data for specific use cases efficiently (e.g. full-text document storage). By eradicating the strictly structured data keeping style defined within the relational model, these DB systems work by offering a much more freely shaped way of working with information, thus providing a great deal of flexibility and ease -- despite the fact that they come with their own problems, some serious considering the important and indispensable nature of data.

III. COMPARITIVE ANALYSIS BETWEEN SQL AND MONGODB FOR VARIOUS DATA SETS

The project was coded in PHP and using Wamp Server, it was simulated on Windows machine to be able to use Apache server and MySQL. For MongoDB, NodeJS was used to be able to use MongoDB on the Windows machine. The data was initially inserted manually and later large datasets were taken from free data sources thereby saving time in creating huge data for analysis. Later the time in seconds were retrieved for each operation and duly noted. The specifications for the three datasets are:

a) *Small Dataset*: 10 rows and 2 columns
b) *Medium Dataset*: 400 rows and 35 columns
c) *Large Dataset*: 2000 rows and 20 columns

3.1 Insertion Time

The table below depicts the insertion time of various data into the two databases.

Table 1- Insertion Time

Data Size	Time in MySQL (in sec)	Time in MongoDB (in sec)
Small	0.000668045	0.000203108
Medium	0.000612954	0.00023527
Large	0.000612974	0.000242857

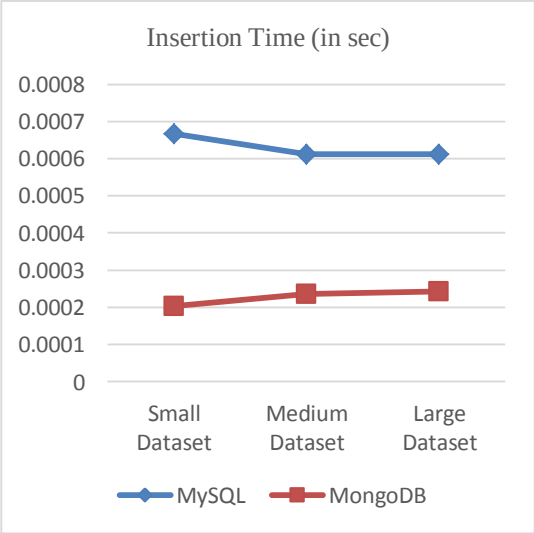

Figure 1- Chart Depicting Insertion Time

3.2 Join Time

The table below depicts the query time of various data into the two databases

Table 2- Join Time

Data Size	Time in MySQL (sec)	Time in MongoDB (sec)
Small	0.23942999	0 (No Joins Required)
Medium	10.37390231	0 (No Joins Required)
Large	27.24617983	0 (No Joins Required)

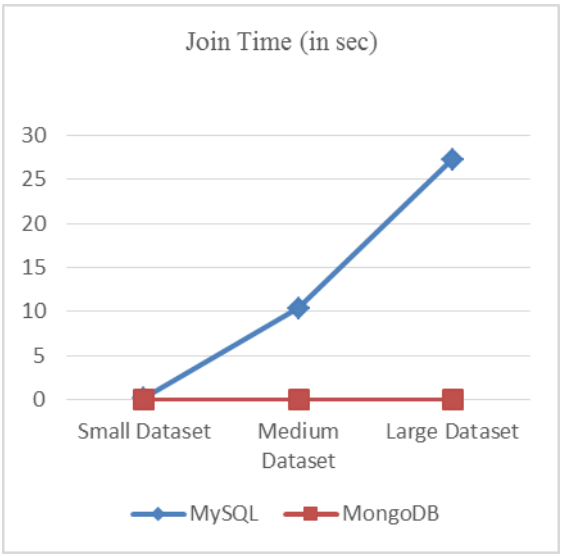

Figure 2- Chart Depicting Join Time

3.3 Retrieval Time

The table below depicts the insertion time of various data into the two databases.

Table 3 – Retrieval Time

Data Size	Time in MySQL (sec)	Time in MongoDB (sec)
Small	0.000838741	0.002098083
Medium	0.001597839	0.005849808
Large	5.270938449	6.2942504

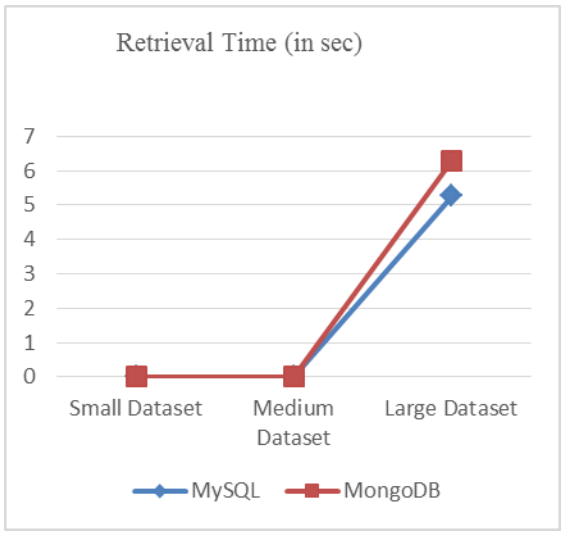

Figure 3 - Chart Depicting Retrieval Time

IV. CONCLUSION BASED ON THE PERFORMED ANALYSIS

Based on the above analysis, the following conclusions can be drawn:

1. For small datasets, MySQL performed better in retrieval time.
2. For large datasets, there was minor difference between MySQL and MongoDB.
3. Insertion time was always better for MongoDB.
4. For relational data where the data could be arranged in row and column format, MySQL performed better.
5. As the data grew non-relational or there were significant gaps in the data, MongoDB performed better.
6. For complex queries involving multiple joins, MongoDB performed significantly better than MySQL due to its data structure allowing it to accommodate any type of data.

If it is required to use medium data without complex queries and normal day to day functioning, then MySQL is a better but if the data is non-relational and may involve complex queries and joins if used in SQL, then MongoDB gives better performance for basic CRUD operations.

REFERENCES

[1] B. Tudorica and C. Bucur, "A comparison between several NoSQL databases with comments and notes," in Roedunet International Conference (RoEduNet), 2011 10th, june 2011, pp. 1 –5.

[2] J. Han, E. Haihong, G. Le, and J. Du, "Survey on NoSQL database," in Pervasive Computing and Applications (ICPCA), 2011 6th International Conference on, oct. 2011, pp. 363 –366.

[3] K. Chodorow and M. Dirolf, MongoDB: The Definitive Guide. O'Reilly Media, September 2010.

[4] N. Leavitt, "Will NoSQL databases live up to their promise?" Computer, vol. 43, no. 2, pp. 12 –14, feb. 2010.

[5] D. Bartholomew, "SQL vs. NoSQL," Linux Journal, no. 195, July 2010.

AUTHORS

First Author – Rajat Aghi, B.Tech(cse), Maharaja Surajmal Institute of Technology and rajataghi@gmail.com.
Second Author – Sumeet Mehta, B.Tech(cse), Maharaja Surajmal Institute of Technology and sumeetrmehta@gmail.com.
Third Author – Rahul chauhan, B.Tech(cse), Maharaja Surajmal Institute of Technology and Rahulchauhan_20@yahoo.co.in.
Fourth Author – Siddhant Chaudhary, B.Tech(cse), Maharaja Surajmal Institute of Technology and siddhant31593@gmail.com.
Fifth Author – Navdeep Bohra, B.Tech, Maharaja Surajmal Institute of Technology and navdeepbohra@gmail.com.