Analysis of MongoDB on Increasing Dataset Size

[1]Dharmender Kumar, [2]K.L. Bansal
[1]M.Tech. Student, [2]Professor
Department of Computer Science,
Himachal Pradesh University, Shimla, India

Abstract : A database is a collection of some meaningful information that is organized so that it can easily be accessed, managed, and updated. Many types of database management systems have been developed to solve different types of problems such as relational database management system. But today, the amount of data is increasing tremendously which can be structured, semi structured and unstructured. Relational database management system is not sufficient to manage this huge amount of unstructured data. To store unstructured data a new alternative called NoSQL has been evolved. NoSQL databases are distributed and open source. There are many external application programs such as MongoDB, Redis, Couchbase etc. to implement NoSQL. In this paper we have tested the performance of NoSQL databases system namely MongoDB on the basis of six datasets consisting of 10 lakhs, 20 lakhs, 30 lakhs, 40 lakhs, 50 lakhs and 60 lakhs records which shares the same attributes. Different types of queries update, remove, find etc. has been used to evaluate the performance of MongoDB with the increase in number of instances in the same dataset.

***Keywords* - NoSQL, MongoDB, RDBMS, JSON, BASON, BASE, ACID**

I. INTRODUCTION

Today data is generating in diverse forms such as tweets, blog post, social network interaction, voice data, multimedia and log data etc. which generate structured, semi-structured and unstructured data in real time. Relational database management system (RDBMS) that has been being utilized has some issues and limitations in structuring and managing unstructured data. RDBMS has performance and cost problems in processing massive data. As RDBMS is not sufficient to manage and manipulate this huge amount of unstructured data, so another selection is NoSQL databases [1]. NoSQL performs read/write operations faster, which makes NoSQL databases compatible for processing huge amount of data. In contrast to relational database management system (RDBMS), NoSQL can handle unstructured data more easily and sufficiently. Thus, many companies which generate Big data containing structured, semi-structured and unstructured data opted for NoSQL databases. Section II describes NoSQL and MongoDB in detail to get better understanding.

II. OVERVIEWOF NOSQL AND MONGODB

In this digital era, volume of data is growing rapidly and data becomes more complex in nature because of its different formats such as sensor data, multimedia, social network interaction, log data and voice data. This type of data is commonly unstructured or semi-structured. RDBMS looks traditional in front of Big data because it is not sufficient to control, retrieve and analyse big data [3]. The alternative to handle this problem is NoSQL. The term "NoSQL" was first used by Carlo Strozzi in 1998 [2]. NoSQL stands for "Not only SQL". NoSQL databases have many advantages over relational database management system (RDBMS) such as these allow data to store without any fixed schema and ability to scale data horizontally (in linear way). The key features of NoSQL are easy to use in conventional load-balanced clusters, allows schema migration, owns individual query system. NOSQL databases follow BASE which consists of three principles [4]. These Three principles are Basically Available, Soft State and Eventual Consistency. NoSQL databases are divided into four major categories which are Key value data stores, Document data stores, Column-oriented data stores and Graph data stores [5]. In this paper, we will evaluate the performance of MongoDB which comes under Document data store. MongoDB is a NoSQL database and it stores the data in form of key-value pairs. It is an open-source document database and dominant NoSQL database which provides high performance and scalability along with data modelling and data management of huge sets of data in an enterprise application. MongoDB is written in C++. MongoDB is a cross-platform, document-oriented database and it provides high performance, high availability, and easy scalability [6]. MongoDB stores data in a binary JSON-like format called BSON (Binary JSON). MongoDB works on concept of collection and document. In MongoDB, database is a physical container for collections. In MongoDB each database has its own set of files on the file system. The important storage components in document databases like MongoDB are collections, instead of tables in the case of relational databases. These collections in MongoDB consist of similar or different JSON, BSON based documents or sub documents. Collection is a group of MongoDB documents. It is equivalent to RDBMS table [7]. MongoDB has the property of auto sharding in which more replica server nodes are added to the system [8].

III. RELATED WORK

Many researches have been done on the MongoDB after knowing its importance. Dipina Damodaran B and Shirin Salim, et.al. [9] did a study on Performance Evaluation of MySQL and MongoDB Databases and stated that when the number of records is higher, MongoDB takes less amount of time as compared to MySQL. MongoDB can be opted for overall better performance. Min-Gyue Jung, Seon-A Youn, Jayon Bae, et.al. [10] compared MongoDB and PostgreSQL on the basis of data input and output and concluded that using MongoDB with unstructured data model will provide overall performance improvement. However, when the environment requires definite and structured data model, using RDBMS such as PostgreSQL will display higher quality of overall performance.

Mrs. Rupali M. Chopade, Mr. Nikhil S. Dhavase [11] did a study on MongoDB, CouchBase: Performance Comparison for Image Dataset and observed that for image dataset insertion, time required by MongoDB is more as compared to CouchBase through tables and graphs, and for image dataset retrieval case is exactly different. Time required for retrieval is lesser for MongoDB than CouchBase. Mayur M Patil, Akkamahadevi Hanni, et.al. [12] did a qualitative analysis of the performance of MongoDB vs MySQL on the basis of insertion and retrieval operations using a web/android application to explore Load Balancing, Sharding in MongoDB and its advantages and noticed that time taken in an application driven by MongoDB at the backend take lesser time to perform the same action that is being performed by using application with MySQL running at the backend. Miaoran Duan, Gang Chen [13] observed that MongoDB's retrieval performance is better than ArcGIS, and both retrieval performances are very regular. MongoDB takes same time to retrieve same data sample size, on the other hand, ArcGIS takes stably increased time as retrieval quantity increases. Suman Tiwari, M.Akkalakshmi, KrishnaKasyap, et.al. [14] did an analysis of NoSQL databases and three different databases of different categories were compared. Mongodb from Document databases, HBase from Column Oriented database and Neo 4J from Graph database. Some basic operations were performed on all the three databases and outputs in all scenarios were observed. From their observations they have concluded when to use and when not to use any of these databases.

IV. RESULTS AND DISCUSSION

The study emphasis on performance of MongoDB using NoSQL. We have used Studio 3T in to evaluate the performance. The environment in which experiment is performed Windows 10 64-bit with memory 4GB, processor intel core i3-2350M CPU @ 2.30GHz and disk space of 500GB. Further six datasets of different size have been used which contain criminal records. These six datasets are consisting of 10 lakhs, 20 lakhs, 30 lakhs, 40 lakhs, 50 lakhs and 60 lakhs records which shares 22 same attributes. Average run time(ms) is performance evaluation parameter. Five different test sets have been used to analyse the performance of MongoDB. An analysis of dataset has been done by using different queries on every dataset. Which are discussed briefly in table below.

Table 1: Query Table

Query 1	db.getCollection(“collection name”).find({}); db.getCollection(“collection name”).find({“Id”}); db.getCollection(“collection name”).find({“Primary Type” : “ROBBERY”}); db.getCollection(“collection name”).find({“PRIMARY TYPE” : “ROBBERY”, “ARREST” : true, “Domestic” : true }); db.getCollection(“collection name”).find({“PRIMARY TYPE” : “THEFT”, “Location Description” : “STREET”, “ARREST” : true, “Domestic” : true }); db.getCollection(“collection name”).find({“LOCATION DESCRIPTOON” : “BANK”, “Primary Type” : “ROBBERY”}); db.getCollection(“collection name”).find({“LOCATION DESCRIPTOON” : “BANK”, “Primary Type” : “ROBBERY”, “ARREST” : true }); db.getCollection(“collection name”).update({“Primary Type” : “THEFT”}, {$set : { “Primary” Type” : “LOOTING” }} {multi : true}); db.getCollection(“collection name”).update({“Primary Type” : “ROBBERY”}, {$set : { “Primary” Type” : “FRAUD” }} {multi : true}); db.getCollection(“collection name”).remove({“Primary Type” : “ROBBERY”}); db.getCollection(“collection name”).remove({“Primary Type” : “THEFT”});

Test set 1: Import dataset into MongoDB

During the loading phase, each dataset loaded into MongoDB in stepwise increasing order. As shown in figure 1 smallest dataset which is of 10,00,000 records took least amount of time and largest dataset which consists of 60,00,000 records took maximum amount of time for loading into MongoDB. In figure 1, X-axis represents number of records and Y-axis represents loading time in milliseconds. While observing the results, it is possible to see the increasing order of time for loading each dataset into MongoDB.

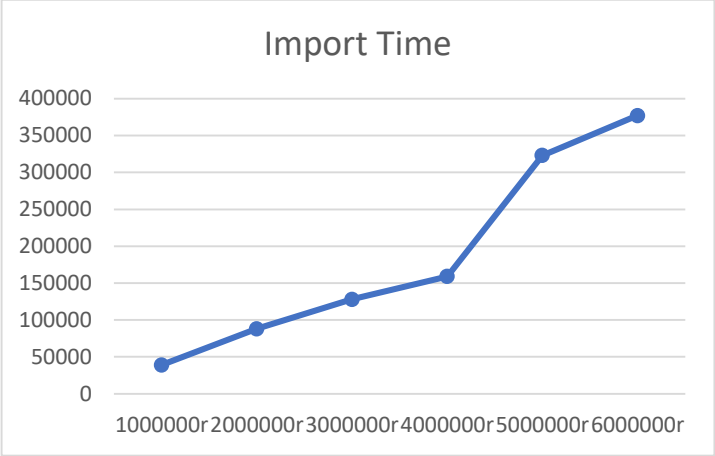

Figure 1: Import time(ms) of all datasets into MongoDB

Test set 2: Find Query

In test set 2, find query is executed in MongoDB to evaluate the execution time of different queries with step wise increase number of records such as 1000000, 2000000, 3000000, 4000000, 5000000, 6000000 as shown in figure 2. In figure 2, X-axis represents number of records and Y-axis represents query execution time in milliseconds. Find query has been used by seven different (Q1, Q2, Q3, Q5, Q6, Q7) types to get better understanding of results. Query 1 is about finding all the documents in the dataset. It shows all the files present in dataset. Query 2 provides a specific record to the user on the basis of unique ID of the documents. Query 3 searches all the criminal records of robbery in the dataset. Query 4 searches specifically for those robbery cases in which arrest has been done. Query 5 finds for those cases in which thievery have been done in street and theft got arrested. Query 6 shows records of all bank robbery cases in dataset. And the last one is Query 7 which shows records of all bank robbery in which criminals got arrested.

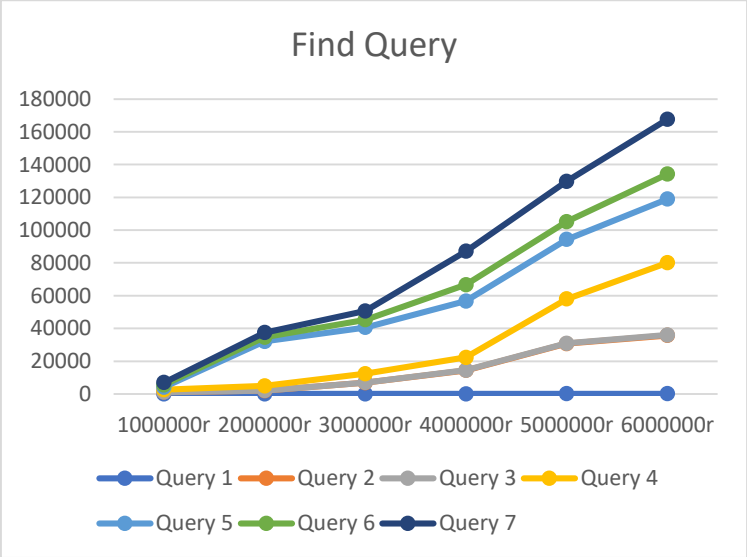

Figure 2: Average run time(ms) to find documents in MongoDB

While observing the results shown in Figure 2, it is possible to see that execution time of queries depends upon the number of records in the dataset and number of searched records. MongoDB is very efficient for both small and large amount of information.

Test Case 3: Update Query

In test case 3, two different queries have been considered to update multiple documents in first one (Query 8) all the records of theft were updated into looting and in second one (Query 9) all records of robbery were updated into fraud. Query 8 updates all the records of "THEFT" into "LOOTING". Query 9 update all the records of "ROBBERY" into "FRAUD".

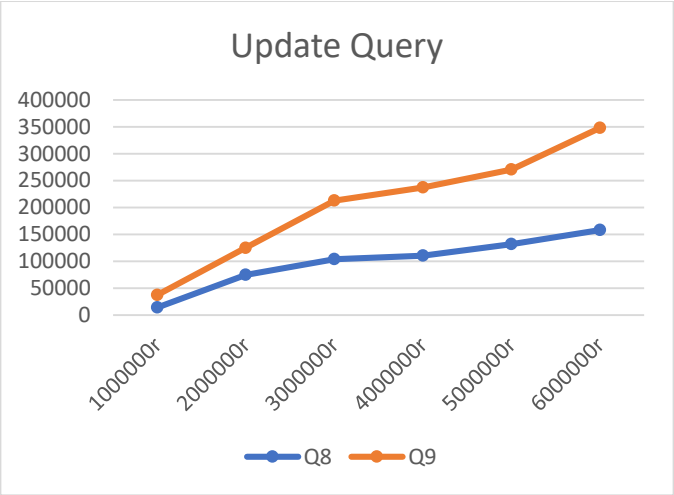

Figure 3: Average run time(ms) for updating multiple documents in MongoDB

From the Figure 3, we noticed that MongoDB is very effective in case of updating multiple documents. In figure 3 number of records is along X-axis and execution time(ms) is along Y-axis. For a large amount of information such as 60,00,000 records it is very efficient. While time taken by MongoDB can vary slightly according to the system configuration.

Test Case 4: Delete Query

Test case 4 is about to evaluate average run time of deleting multiple documents with stepwise increase in number of records such as 1000000, 2000000, 3000000, 4000000, 5000000, 6000000 as shown in figure 4. Test case 4 also has been divided into two cases in first one (Query 10) all records related to robbery were deleted then in second one (Query 11) all records related to theft were deleted.

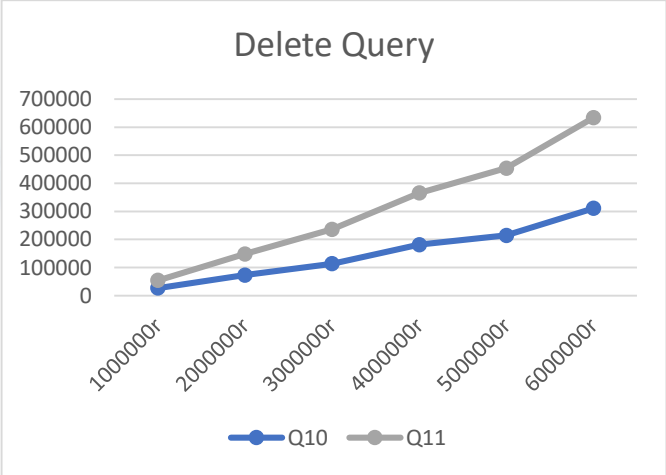

Figure 4: Average run time(ms) for deleting Multiple records in MongoDB

From the figure 4, we can see the amount of time spent by MongoDB to delete multiple documents from a dataset which is very difficult task to do manually. In figure 4, number of records is along X-axis and execution time (in milliseconds) is along Y-axis. While observing the results shown in figure 4, it can be seen that story is same for deleting records, average run time for deleting multiple documents increases when number of records increase. But we can say that MongoDB performs very well in small datasets as well as large datasets. The amount of time it took to delete multiple documents is very less as compared to size of datasets.

Test Case 5: Dataset Export Time

Test case 5 evaluates the export time of different datasets of different sizes from MongoDB. Every dataset exported separately to evaluate the average run time to evaluate the export time.

.

Figure 5: Time taken by MongoDB to export different size of databases.

It is clear from Figure 5 that export time increases as number of records increases in the dataset. In Figure 5, X-axis represents number of records and Y-axis represents export time in milliseconds. For 10,00,000 records time is lowest and for 60,00,000 records it is highest.

V. CONCLUSION AND FUTURE WORK

In this work dataset has been divided into six smaller sets, each dataset consisting of different sizes, containing 10 lakhs, 20 lakhs, 30 lakhs, 40 lakhs, 50 lakhs and 60 lakhs instances. Five test sets with different set of queries and operations have been taken into consideration to analyse the performance of MongoDB. From observing the results, it can be said that the import, export and query processing time of MongoDB increases with the increase in size of dataset used. Although the relationship between the processing time taken by MongoDB and the size of datasets used is not linear but it is directly proportional to the changes. So, it can be concluded that the processing time of MongoDB increases with the increase in size of the dataset used. For future scope, we can implement some other NOSQL tool and compare its performance with MongoDB on a particular application domain.

REFERENCES

[1] Rajat Aghi, Sumeet Mehta, "A comprehensive comparison of SQL and MongoDB databases", Department of Computer Science, Maharaja Surajmal Institute of Technology, Janakpuri, N.delhi 110058, India

[2] Vatika Sharma, Meenu Dave, "SQL and NoSQL Databases", Department of CSE, Jagan Nath University, Jaipur, India, Vol 2, issue 8, Aug 2012

[3] Rakesh Kumar, Bhanu Bhushan Parashar, et.al, "Apache Hadoop, NoSQL and NewSQL Solutions of Big Data", International Journal of Advance Foundation and Research in Science & Engineering (IJAFRSE) Volume 1, Issue 6, October 2014.

[4] Supriya S. Pore1, Swalaya B. Pawar2 "Comparative Study of SQL & NoSQL Databases", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Volume 4 Issue 5, May 2015

[5] Asst. Prof. Raju Sharma, Asst. Prof. Yatendra Kashyap, "A STUDY OF NOSQL DATABASES AND WORKING OVERVIEWS", Department of Computer Science & Engineering Corporate Institute of Science & Technology, Bhopal

[**6]** Cornelia Győrödi, Robert Győrödi, Roxana Sotoc, " A Comparative Study of Relational and NonRelational Database Models in a Web- Based Application" Department of Computer Science and Information Technology, University of Oradea, International Journal of Advanced Computer Science and Applications, Vol. 6, No. 11, 2015

[7] Liberios Vokorokos, Mat´uˇs Uchn´ar, Anton Bal´aˇz, "MongoDB scheme analysis" Department of Computers and Informatics Technical university of Koˇsice Letn´a 9, 042 00 Koˇsice, Slovakia

[8] https://www.mongodb.com

[9] Dipina Damodaran B, Shirin Salim, "PERFORMANCE EVALUATION OF MYSQL AND MONGODB DATABASES", Department of Computer Engineering, M A College of Engineering, Kothamangalam, Kerala, India

[10] Min-Gyue Jung, Seon-A Youn, Jayon Bae, Yong-Lak Choi, "A Study on Data Input and Output Performance Comparison of MongoDB and PostgreSQL in the Big Data Environment" , Graduate School of Software, Soongsil University, South Korea

[11] Mrs. Rupali M. Chopade, Mr. Nikhil S. Dhavase, " MongoDB, CouchBase: Performance Comparison for Image Dataset", Department of Information Technology MMCOE, Karvenagar Pune, India, Vol, Issue 21, December 2017

[12] Mayur M Patil, Akkamahadevi Hanni, C H Tejeshwar, "A qualitative analysis of the performance of MongoDB vs MySQL database based on insertion and retriewal operations using a web/android application to explore load balancing — Sharding in MongoDB and its advantages", B V Bhoomaraddi College of Enginnering and Technology, Hubli, Karnataka, India, Volume 01, Issue October, 2017

[13] Miaoran Duan , Gang Chen, "Assessment of MongoDB's Spatial Retrieval Performance", Jiangsu Provincial Key Laboratory of Geographic Information Science and Technology, Department of Geographic Information Science, Nanjing University, Nanjing, 2lO093

[14] Suman Tiwari, M.Akkalakshmi, KrishnaKasyap Bhagavatula, "Analysis of NoSQL databases: MongoDB, HBASE and Neo4J ", Dept. of CSE GITAM University Hyderabad, India, International Journal of Engineering Trends and Technology (IJETT) – Special Issue – April 2017