www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issue 10 Oct. 2016, Page No. 18564-18567

MongoDB for Read Intensive Cases Using YCSB

Adarsh Kumar[1], Anita Ganpati[2]
[1]Department of Computer Science, Himachal Pradesh University,
Summer Hill, Shimla-5
adarshsharma008@gmail.com
[2]Department of Computer Science, Himachal Pradesh University,
Summer Hill, Shimla-5
anitaganpati@gmail.com

Abstract: ***With the emergence of cloud computing and continuously decreasing cost of the storage; it has become very easy to store unstructured data generated from the social media post, multimedia etc. To store unstructured data a new mechanism called NoSQL has evolved, which can store the data naturally and logically with lose restrictions on the database schema. This paper attempts to use NoSQL database system namely MongoDB for read intensive type cases, implementation scheme, and finally average run time compared of different cases with increase number of records and operations using YCSB.***

Keywords: BASE, Big data, MongoDB, NoSQL, YCSB

1. Introduction

The relational database systems is the traditional technology used widely that enables storage, management and retrieval of varied data schemas. These systems have limitations to deal with scalability for large amount of data for store and execute. With the beginning of digital revolution since 1991, the Internet or WWW and HTTP protocol has become standard for information sharing [4]. Data is generating in diverse forms such as blog post, tweets, social network interaction and log data etc. A study of International Data Corporation in 2014 estimates that digital universe will be large by 2020 containing nearly as many digital bits as there are stars in the universe. It is doubling in size every two years. By 2020 the data will be creating and copying 44 zettabytes or 44 trillon gigabytes per annum [12]. To handle this type of emerging data, NoSQL systems are evolved. In this paper, evaluate the average run time of MongoDB for read intensive cases with increase number of records and number of operations using YCSB (yahoo cloud storage benchmark).
section V, evaluate the average run time of MongoDB for different cases using YCSB (yahoo cloud storage benchmark). Finally Section VI presents the conclusion and future scope of the paper.

2. Overview of NoSQL, MongoDB and YCSB

The large volume of data is generating at a much faster velocity in varieties of formats such as multimedia, log data and voice data. This data is not easily fit into a column and row database systems. To handle this type of emerging data, a term "NoSQL" was coined by Carlo Strozzi in 1998 [3]. The NoSQL stands for "Not only SQL" or "Not Relational". These systems are inspired from Google's BigTable, memcached and Amazon's Dynamo. The Key features of these systems are shared nothing, horizontal scaling and replication and partition of data over many servers. The NoSQL based on Eric Brewers's CAP Theorem [5]: A distributed storage system must choose to sacrifice either consistency or availability while having partition tolerance. The term "BASE: Basically Available, Soft state, Eventually consistent" coined by Eric brewer [5] for these systems to handle the needs of internet and cloud based models of storage which abandon the ACID properties as a tradeoff for their increased performance and Scalability. These systems are grouped according to their strengths and CAP theorem compromise by Big Data Working Group [10] are relational, document –oriented, key-value, BigTable-inspired, Dynamo-inspired, graph and NewSQL.
MongoDB [11] is a GPL open source, document-oriented, written in C++ and supported by 10gen. It supports automatic sharding, distributing documents over servers. MongoDB stores data in a binary JSON-like format called BSON (Binary JSON). Its documents tend to have all data for a given record in a single document. It supports drivers of all programming languages and frameworks are Java, .NET, Ruby, PHP, JavaScript, node.js, Python, Perl, Scala and others to make development natural, support query types are key value, range, geospatial, text search, aggregation and MapReduce. It also supports a GridFS specification for large binary objects like images and videos. It supports native replication with automatic failover and recovery. It supports high availability across racks and data center, and multi-center scalability. Replication is asynchronous for higher performance. It is use by Expedia, Forbes, MetLife, OTTO, CRITTERCISM and BOSCH. It is the only database that harnesses the innovations of NoSQL and maintaining the foundation of relational databases with its Nexus Architecture. Yahoo Cloud Service Benchmark [1] (YCSB), an open source, extensible, Java based client satisfied the requirements and was the ideal choice for benchmarking NoSQL databases. The primary focus is creating a benchmarking tool that simplifies the process of benchmarking different system that can be deployed on single machine or cloud platform. Each YCSB release contains code to interface with various database systems such as HBase, Redis and MongoDB. There are steps to run a workload are set up the database system, choose the appropriate workload, choose the appropriate parameter, load the data and execute the data [13].

3. Literature Survey

Alexandru Boiceaet. al. [2] compared MongoDB and Oracle database. A comparison criterion includes theoretical

differences, features, restrictions, integrity, distribution, system requirements, architecture, query and insertion times. In their study for a best comparison between the two database engines run some tests and compute the time that took each engine to take some actions on the database are insert, update and delete. MongoDB perform these operations efficiently than oracle system for large number for records. MongoDB is a more rapid database management system.

D. Nelubin and B. Engber [8] in their study on durability and performance tradeoff of NoSQL systems. In this paper, analyzes performance characteristics of four key-value database are Cassandra, Couchbase, Aerospike, and MongoDB using YCSB [1] from Yahoo. The performance characteristics are throughput test and step-wise load test for both read-heavy and balanced read-write workloads in RAM and disk for 4-node cluster. The most striking result was the raw throughput number Aerospike was able to achieve even while committing to disk across multiple nodes with maintaining a speed of 200 thousand operations per second and normally not associate with ACID semantics. When the entire data set fit into RAM and the durability guarantees are weakened, the results showed both Couchbase and Aerospike in a near-tie in terms of performance. Couchbase slightly outperformed Aerospike for the balanced read-write workload, and Aerospike somewhat more significantly outperformed Couchbase for the read-heavy workload. Both Cassandra and MongoDB lagged far behind the others, but both offer a significantly larger feature set than Aerospike and Couchbase 1.8.

Min-Gyue Jung et al. [7] have done a study on data input and output performance comparison of MongoDB and PostgreSQL in the Big Data Environment. In this study, PostgreSQL and MongoDB have been selected to represent RDBMS and NoSQL respectively for comparative analysis. Due to advancement of social network and popularization of mobile devices, the existing relational database management system (RDBMS) processing of massive data has become an issue. NoSQL is a database management system which makes processing of massive and/or unstructured data easier. Converting the RDBMS of current systems to NoSQL has become a trend. In their study comparison parameters are insert, select, update, and delete operations on PostgreSQL and MongoDB. Insert, select, update and delete operation speed of MongoDB is faster than that of PostgreSQL in general and designing with unstructured data model seems to be better than designing with relational data model for performance improvement.

Cornelia Gyorodi et al. [6] compared MongoDB and MySQL.In their study, the advantages of using the non-relational Database MongoDB compared to the relational database MySQL and various operations were performed on the two databases. These operations are the four basic operations that can be performed on any database are insert, select (query), update and delete. MongoDB provided lower execution times than MySQL in all four basic operations. The comparison tests proves that for large amounts of data MongoDB has a good performance and it is preferred than MySQL.

The study has performed by [9] Nico Rutishauser in which TPC-H queries was implemented in MongoDB to see the performance difference with the open source RDBMS PostgreSQL. The performance of the MongoDB was observed very poor as compared to the PostgreSQL.

4. Objective and scope of study

In this paper, an evaluation of MongoDB for read intensive cases using YCSB. The evaluation is done on average run time with increase number of records and operations using YCSB from yahoo.

5. Research Methodology

The research methodology follows theoretical approach that comprises literature survey, articles, books, research papers and internet. An experimental approach followed which evaluate the average run time of MongoDB the basis of number of records and number of operations using YCSB.

6. Experimental Scheme

The Experimental Scheme of MongoDB based on a Simple Model. According to this model, first specify the requirements, and then choose a NoSQL database system. The third step is to design the case and execute the test cases. Finally, analyzes the reports/ results for suitability of NoSQL database system for specific requirement as shown in Figure 1. In this paper, different types of read intensive cases are taken for MongoDB and evaluated using YCSB. The environment in which experiment is performed is Ubantu 16.04 LTS 64-bit with memory 4GB, processor Intel core i5 CPU 650 @3.20 GHz X 4 and disk 320 GB.

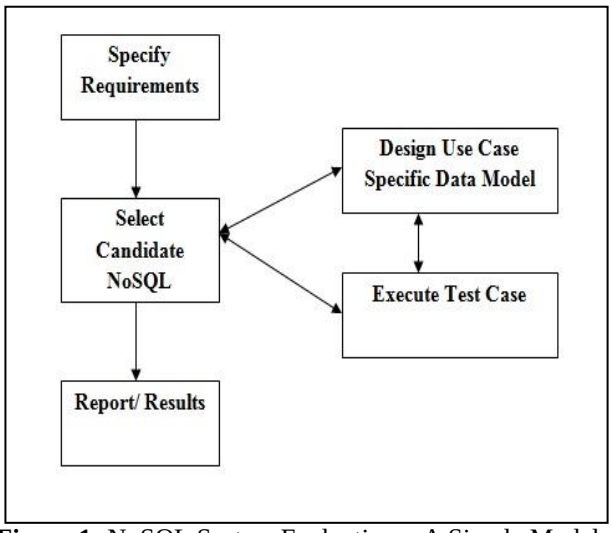

Figure 1: NoSQL System Evaluation – A Simple Model

7. Results and Analysis

The study emphasizes on the performance evaluation of MongoDB using YCSB benchmark from yahoo. The performance evaluation parameter is average run time and average throughput with increase of record count and operations for read mostly, read only and read latest workload cases. The MongoDB-Read Mostly case contains 95 % read and 5% update. The MongoDB-Read only case contains 100% read. The MongoDB-Read Latest case contains 95 % read and 5% insert.

In Loading Phase

During the loading phase, records of each 1Kbyte loads into MongoDB in stepwise in increasing order such as 1000, 10000, 40000, 80000,120000, 160000 and 200000.

DOI: 10.18535/ijecs/v5i10.39

Table 1: Average Run Time (ms) for Loading Phas

Workload Type		MongoDB-Read Mostly	MongoDB-Read Only	MongoDB-Read Latest
No. of Records	1000	276	295	290
	10000	773	674	784
	40000	3151	2949	3006
	80000	6016	5730	5803
	120000	8566	8464	8787
	160000	11306	11153	11331
	200000	14139	14386	14425

As shown in Table 1, average run time in milliseconds with increase number of records for read intensive cases of MongoDB.

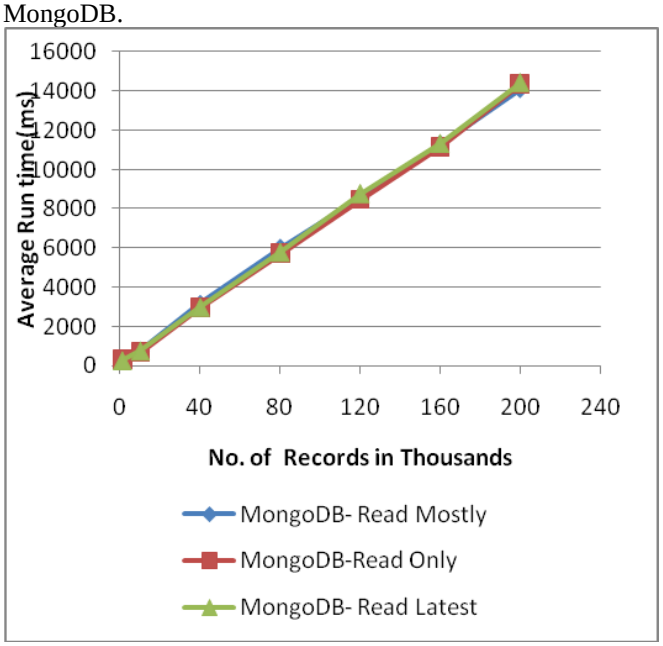

Figure 2: Loading of Read Intensive cases into MongoDB

To evaluate the loading run time of different read intensive cases with step-wise increase of number of records such as 1000, 10000, 40000, 80000, 120000, 160000 and 200000 as shown in Figure 2. While observing results, it is possible to see that there is no significant difference among MongoDB-Read Mostly, MongoDB-Read Only and MongoDB-Read Latest. When the number of records increase, the average run time increases in similar proportion for all MongoDB cases.

In Executing Phase

During the executing phase, the number of records loaded into MongoDB is 1000 each of 1 Kbyte and number of operation performed by MongoDB is in stepwise increasing order such as 1000, 10000, 40000, 80000, 120000, 160000 and 200000.

Table 2: Average Run Time (ms) for Executing Phase

Workload Type		MongoDB-Read Mostly	MongoDB-Read Only	MongoDB-Read Latest
No. of Operations	1000	383	409	396
	10000	1423	1386	1491
	40000	4249	4164	4190
	80000	7373	6900	7336
	120000	10383	10132	10349
	160000	12900	12501	13499
	200000	16243	15341	16142

As shown in Table 2, average run time in milliseconds with increase number of operations for read intensive cases of MongoDB.

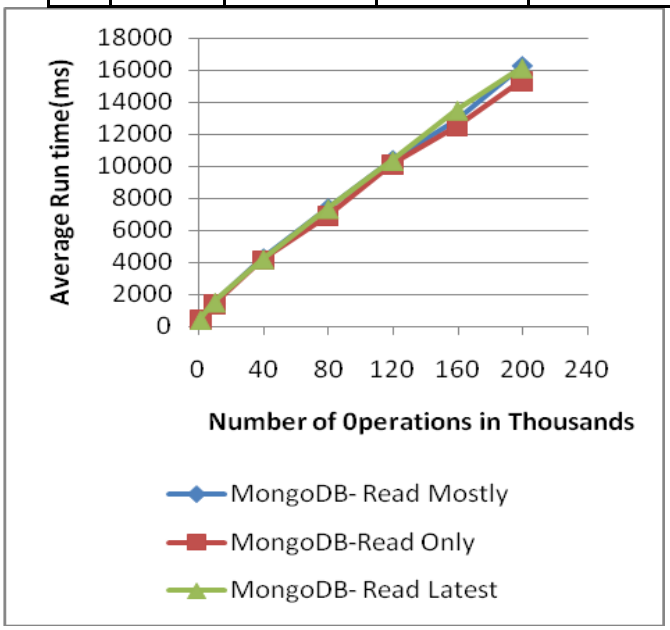

Figure 3: Execution of Read Intensive Cases in MongoDB

To evaluate the loading run time of different read intensive cases with step-wise increase of number of operations such as 1000, 10000, 40000, 80000, 120000, 160000 and 200000 as shown in Figure 3. While observing results, it is possible to see that there is no significant difference among MongoDB-Read Mostly, MongoDB-Read Only and MongoDB-Read Latest. When the number of operations increase, the average run time increases in similar proportion for all MongoDB cases.

8. Conclusion and Future Scope

With the development of web, there is need of store and process big data effectively and demand for high performance when reading and writing. So, the relational database systems are facing many new challenges. To deal with these challenges NoSQL databases are evolved and have been successful in many production systems. In this paper, evaluate average run time of MongoDB for read intensive with increase number of records in loading phase and operations in executing phase using YSCB. For all the read intensive cases, it is observed that average run time increases in similar proportion with increase number of records and operations. In Future, there will be scope of doing empirical analysis based on throughput, number of node it handle by MongoDB and on various algorithms used for Concurrency control, failure detection and transaction mechanism.

References

[1] B. F. Cooper, A. Silberstein, E. Tam, R. Rama ksrihnan, and R. Sears, Benchmarking cloud serving systems with YCSB," Proceedings of the 1st ACM symposium on Cloud computing - SoCC '10, pp. 143-154, 2010.

[2] Boicea, Alexandru, Florin Radulescu, and Laura IoanaAgapin. "MongoDBvs Oracle-Database Comparison." 2012 Third International Conference on Emerging Intelligent Data and Web Technologies. IEEE, 2012.

[3] Cattell, Rick. "Scalable SQL and NoSQL data stores." ACM SIGMOD Record39.4 (2011): 12-27

[4] Dean, Jared. Big data, data mining, and machine learning: value creation for business leaders and practitioners. John Wiley & Sons, 2014.Cattell, Rick. "Scalable SQL and NoSQL data stores." ACM SIGMOD Record39.4 (2011): 12-27

[5] Gilbert, Seth, and Nancy Lynch. "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services." ACM SIGACT News33.2 (2002): 51-59.

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

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

[8] Nelubin, D., and B. Engber. "Ultra-high performance nosql benchmarking: Analyzing durability and performance tradeoffs." Thumbtack Technology, Inc., White Paper (2013).

[9] Nico Rutishauser, "TPC-H applied to University of Zuric, "MongoDB: How a NoSQL database performs", A Report submitted to , (2012):pp. 1-29

[10] Big Data Taxonomy, https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Big_Data_Taxonomy.pdf, accessed on 3rd April, 2016 on 10:30 AM

[11] MongoDB, https://www.mongodb.com , accessed on 1st July,2016 on 01:00 PM

[12] Digital Universe, http://india.emc.com/leadership/digital_universe/2014iview/executive-summary.htm, accessed on 3rd May, 2016 on 11:30 AM

[13] YCSB, https://github.com/brianfrankcooper/YCSB/wiki/Getting-Started, accessed on 22th May, 2016 at 11:10 AM