International Journal of Applied Mathematical Research, 8 (1) (2019) 27-31

International Journal of Applied Mathematical Research

Website: www.sciencepubco.com/index.php/IJAMR

Research paper

A comparison study of NoSQL document-oriented database system

Sugimiyanto Suma [1] *, Fahad Alqurashi [2]

[1] Department of Computer Science, FCIT, King Abdulaziz University, Saudi Arabia
[2] Information Technology for Infrastructure, King Abdulaziz University, Saudi Arabia
**Corresponding author E-mail: fahad@kau.edu.sa*

Abstract

By increasing data generation at these day, requirement for a sufficient storage system are strongly needed by stakeholders to store and access huge number of data in efficient way for fast analysis and decision. While RDBMS cannot deal with this challenge, NoSQL has emerged as a solution to address this challenge. There have been plenty of NoSQL database engine with their categories and characteristics, especially for document-oriented database. However, it makes a confusion for the system developer to choose the appropriate NoSQL database for their system. This paper is our preliminary report to provide a comparison of NoSQL databases. The comparison is based on performance of execution time which is measured by building a simple program. This experiment was done in our local cluster by exploiting around 1 million datasets. The result shows that RDB has better performance than CDB in terms of execution time.

Keywords: *Comparison; Document-Oriented; Execution-Time; Huge Data; Nosql.*

1. Introduction

The growing of internet and cloud computing has affected to increasing data generation which lead to huge number of data. When this data needs to be stored and accessed in efficient way for fast analysis, RDBMS not able to answer this requirement. NoSQL arises to address this challenge. NoSQL which refers to not-only SQL or non-relational, take cares the data with different manner of what RDBMS does. There have been numbers of NoSQL currently exist, and it is classified to several categories based on the way their store the data, the place, and the special purpose of data storing such as graph. By the numbers of existing NoSQL, it needs to be compared to give a view about some databases engine for developers. It will give them idea when selecting database system for their application based on the requirement.

In this paper, we did performance comparison against selected document-oriented databases in terms of query execution. The selected databases are RethinkDB (RDB) these two document databases are they are in the top 10 of document oriented databases [1], which are the most popular one among users at these days.

The next section is divided into four phase. Section 2 will describe the background of this research and related work. Then, Section 3 explanation about methodology. After that, Section 4 describes result and analysis. Finally, we conclude in Section 5.

2. Background and related work

NoSQL aka not-only SQL or non-relational database system. It is state-of-the-art of database storage system which the emergence as a solution to address demand of storing and processing big data with high performance effectively [2]. While relational database cannot handle this requirement. Instead of storing data with row-column relational based, data storing in NoSQL has their own manners. There are three main categories of NoSQL databases: (1). Key-Value database; it means storing data by a pair of key and value, while a value corresponds to a key. This structure is simple and has higher query speed than relational database. (2) Column-oriented database; using table as the data model, however does not support table relation. It stores data by column and stored separately for each column. The advantage is more suitable application on aggregation function and data warehouse. (3). Document-oriented database; this model is similar to Key-value based in structure, however the value of document-oriented database is semantic, as well as it is stored in XML or JSON format.

There are numerous researches regarding NoSQL database and its comparisons. Tudorica and Bucur performed comparison between several NoSQL databases in general [3], while there are several categories of NoSQL exists at these days. Oguducu et al. conducted comparison against several NoSQL databases, particularly in-memory database system [4]. Less number of comparative study which evaluates various in-memory databases became their motivation. They selected Timesten, Altibase, solidDB, and SQLite to be compared in terms of performance, and with different operating system. They measured the query response time of databases system to queries different amount of data and diverse level of query complexity. The result shows that Altibase gives the best results running in Linux, and TimesTen performs much better compared with others under the same conditions.

Wang et al. studied performance survey of in-memory database (IMDB) [5]. They focused on the data structure, architecture, concurrency, volume, scalability, and availability. As IMDB is on demand to address stakeholder requirement while emerging big data term and fast performance processing. They proposed V3 performance model to evaluate big data characteristics (volume, velocity, and variety) of 19 IMDB systems. The experiment results show that NewSQL is better at dealing with high-frequency trading models.

Kolomicenko et al. conducted research on contrasting NoSQL graph databases system, and revealing possibilities and limitation of graph database system [6]. They formulated the requirements of a universal graph database benchmark, as well as they developed an extensible benchmarking tool called BlueBench. Whereas, Lourenco et al. performed survey comparison against NoSQL engines in different manner than what researchers have been done [7]. Their comparison is mainly focus on identifying the most beneficial use case scenarios from the software engineer point of view, as well as the suitability for quality attributes on the design of enterprise systems.

Klein et al. evaluated performance of three NoSQL databases with different category [8]. The three databases are MongoDB as document-oriented database, Cassandra as column oriented database, and Riak as key-value database. The performance comparison was done by executing a case study for an electronic healthcare system. They also revealed the challenges during experiment such as creating the test environment, and validating quantitative criteria.

Upeksha et al. contrasted several database systems from graph databases, relational databases, indexed file systems, and column store databases [9]. This paper aims to give a view about data storage system for developer when they are going to select a data storage system for their environment. The comparison steps are inserting data into corpus, and fetching data from it. As the result, trying to suggest an optimal data storage architecture for a language corpus. While, the speed of data insertion and information retrieval are the most important facts to be considered when selecting data storage system.

Plechawska-Wojcik and Rykowski compared different database system such as relational, document oriented, and graph databases in the context of web application development [10]. The comparison consists of data model analysis which discuss requirements meeting and difficulties of different implementation, and performance tests which present execution result of five different tasks. The solutions show that each database system has their own strengths and weaknesses in some cases.

Other researches which performing comparison on different particular aspect such as Schindler [11], which profiled and analyzed the I/O performance of NoSQL database compared with RDBMS, or Naheman and Wei [12], which reviewed NoSQL databases and tested the performance on top of HBase. A comparative study is essentially needed since there are plenty of NoSQL database system nowadays which leads to confusing for developer, particularly document oriented based in terms of performance query execution. According to above literature review, there have been plenty of works comparing NoSQL database system. Nonetheless, most of them comparing others category of NoSQL such as in-memory or graph database, or if exists comparing document database, it is not in terms of performance query execution.

3. Methodology

The goal of this paper is to reveal the different performance of RDB and CDB in terms of query execution time. The performance comparison is done by constructing scenario testing of query execution with different level of complexity and number of dataset, this query execution is limited by select query statement. Since both of RDB and CDB have their own caching mechanism to speedup query processing, thus, to make it fair we set the cache size of both database are similar, that is 4GB.

3.1. Query tester

Before we do the performance comparison test, we created several queries that will be executed on top of each database. The query consists of three queries with different level of complexity. First, simple query (Q1), intermediate query (Q2), and complex query (Q3). Q1 contains a simple select query, Q2 comprises intermediate query with select and equi join statements, and Q3 contains a complex query with select, equi join, group by, order by query statements. For Q2 and Q3, the datasets will be joined with another dataset for joining purpose which contains 10.000 objects.

3.2. Dataset

We generated the dataset to be executed by query tester. The dataset is adopted from [13], which contains JSON document in one file with thousands of object. For our experiment purpose, since our goal is to compare the performance in terms of query execution, thus it does not consider about context of data validity or redundancy. Therefore, to increase the data size, we duplicated the data and separate it into three datasets with different size. First, D1 which contains around 10.000 objects, D2 contains 100.000 objects, and D3 contains 1.000.000 objects.

3.3. Execution time measurement

In order to measure the execution time taken by database engine to execute given query, we create a java- maven based program to do it. The program will connect to database engine through library of maven repository, then execute the given queries. The execution time is measured by placing java function System.currentTimeMillis(), before and after line code of query execution as shown in Figure 1, then calculate the difference between them to get the time in millisecond. To get the pure execution time, the select query is only line code to execute the query without printing the result.

```
. . .
Timer.start;
Call.query("SELECT * FROM <TABLE>");
Timer.stop;
. . .
```

Fig. 1: Illustration of Execution Time Measurement.

The code in line 3 is the query tester (Q1, Q2, Q3). The code in line 1 and 5 are other necessary codes such as opening connection to database, configuration, or closing the connection.

3.4. Testing scenario

In testing phase, we need a defined scenario to test, and to reduce the external factor which influences the database engine performance during query execution, we close all unnecessary application either which are running in background or in window, except several applications such as IDE window to write the program, Couchbase and RethinkDB service, and TeamViewer as this is done remotely and everything is installed on top of my local cluster. Dataset which is mentioned before is in form of JSON document file. Thus, it needs to be imported first to RDB and CDB engines before move to testing phase. The scenario to test the performance is described in

Table 1: Testing Scenario

Start
Step 1: close all unnecessary application Step 2: open IDE window
Step 3: select particular class (RDB | CDB)
Step 4: define the dataset to be used (D1 | D2 | D3)
Step 5: define the query level to be executed (Q1 | Q2 | Q3)
Step 6: run the program
Step 7: record the time execution
Step 8: back to step 3 until exhausted End

In step 4 and step 5, we did the test by increasing the dataset size gradually. Thus, first, we took D1 then executed it with Q1, Q2, Q3 respectively. After that, we took D2, and D3 with the same step. All of these steps were done 5 times repeatedly to get the average time.

3.5. Experiment environment

NoSQL is mainly for handling large data with distributed system. Our experiment was done in local cluster with 3 nodes, the environment detail is shown in Table 2. Almost all the nodes including master node have equal specification, just to make is simple representation, I took the average.

Table 2: Environment System of Experiment

Operating System	Linux Ubuntu 16.04 TLS x64
Processor	Intel Core i7
RAM	16 GB
Network	Intranet (Ethernet)

Both RDB and CDB are installed and ran on top of that cluster environment, and both have the same configuration of master node and slave node. The same dataset is imported into both database system. As well as, to make it fair comparison, we disabled their caching feature.

4. Result and analysis

According to testing scenario which we have discussed in previous phase. The experiments were done 5 times repeatedly to get the average time for each dataset (D1, D2, D3) and query (Q1 Q2, Q3). The difference testing result achieved by RDB and CDB of varied query with dataset D1 can be figured out as shown in Figure 2.

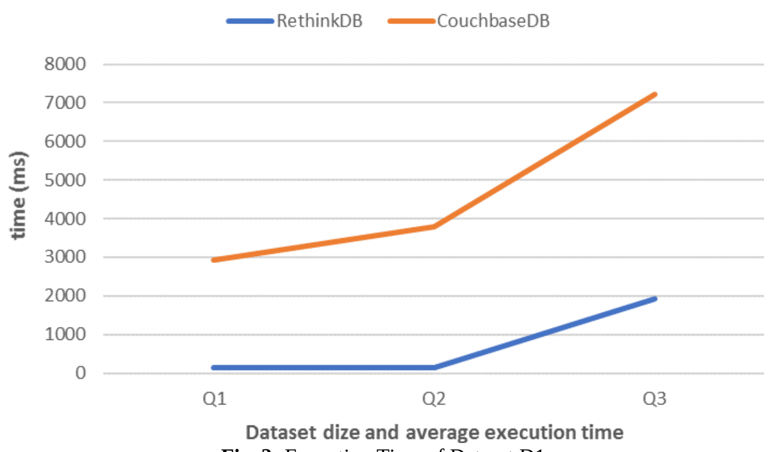

Fig. 2: Execution Time of Dataset D1.

The number of execution time of dataset D1 achieved by RDB and CDB with the various queries in detail can be looked at Table 3.

Table 3: Testing Result Executing Dataset D1

Query	Average of execution time (millisecond)	
	RDB	CDB
Q1	129	2929
Q2	151	3785
Q3	1935	7216

The second testing result, is the difference execution time achieved by RDB and CDB for dataset D2 can be figured out in Figure 3.

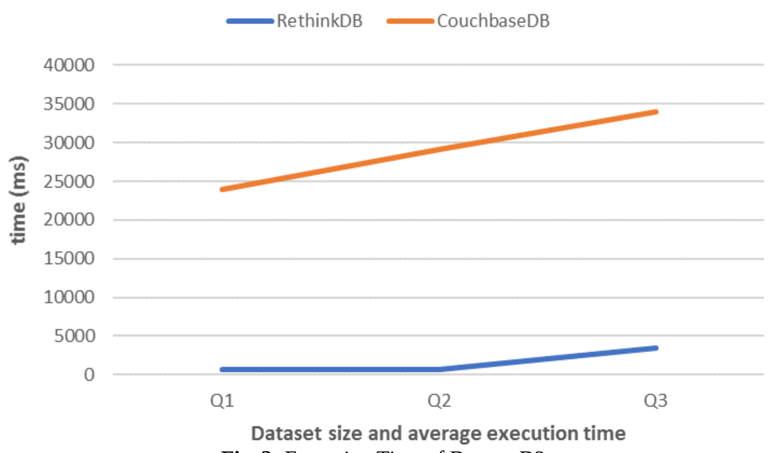

Fig. 3: Execution Time of Dataset D2.

The number of execution time achieved for dataset D2 in detail can be looked at Table 4.

Table 4: Testing Result Executing Dataset D2

Query	Average of execution time (millisecond)	
	RDB	CDB
Q1	610	23902
Q2	625	29098
Q3	3522	33978

The last result of achieved execution time for dataset D3 is shown in Figure 4.

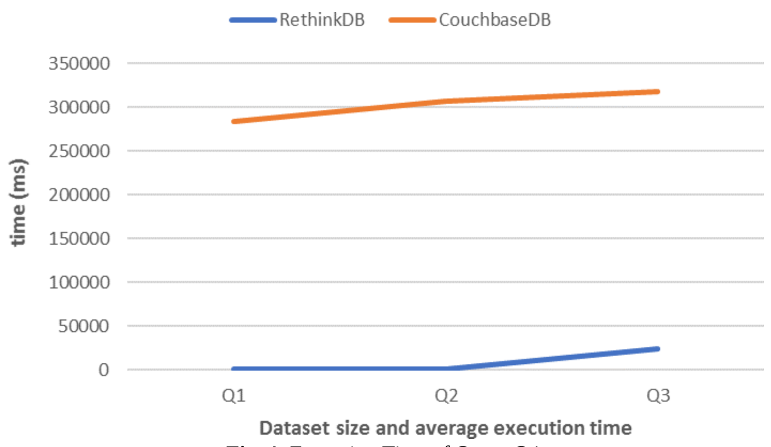

Fig. 4: Execution Time of Query Q4.

The number of execution time achieved for dataset D3 in detail can be looked at Table 5.

Table 5: Testing Result Executing Dataset D3

Query	Average of execution time (millisecond)	
	RDB	CDB
Q1	633	283131
Q2	643	306402
Q3	23235	317584

According to the testing experiment of executing the datasets with varied queries, we can observe the difference performance achieved by RDB and CDB. The difference is significant, while RDB has faster performance in terms of execution time, although the cache size set similarly. During learning and experiment, we noticed that they have different mechanism to manage the data, this thing causes the performance differentiation between them.

5. Conclusion

NoSQL has emerged as a solution to address the storing of huge data at these days. There have been a lot of NoSQL database engine have developed, in their categories and characteristics, particularly for document- oriented database. Meanwhile, this number of NoSQL makes confusion for the developer to select the appropriate NoSQL database to support their system. This paper is our preliminary work to provide a comparison of document oriented databases. The comparison is based on performance of execution time which was measured by a simple program. This experiment was done on our local cluster by exploiting around 1 million datasets. The result shows that RDB has better performance than CDB in terms of execution time. We hope, this will give at least a brief introduction about RDB and CDB difference in terms of execution time.

The future work will cover comparison of more document-oriented database system, as well as adding additional features to be compared beside execution time, such as reliability of scaling system, simplicity, and others.

Acknowledgement

The experiments in this paper were supported by High Performance Computing (HPC) centre at King Abdulaziz University.

References

[1] S. IT, "DB-Engines Ranking of Document Stores." [Online]. Available: https://db- engines.com/en/ranking/document+store. [Accessed: 17-May-2017].
[2] J. Han, H. E, G. Le, and J. Du, "Survey on NoSQL Database," 6th Int. Conf. Pervasive Comput. Appl., vol. 2, no. 2, pp. 50-54, 2011.
[3] B. G. Tudorica and C. Bucur, "A comparison between several NoSQL databases with comments and notes," Proc. - RoEduNet IEEE Int. Conf., 2011. https://doi.org/10.1109/RoEduNet.2011.5993686.
[4] S. Oguducu, M. Gayberi, and H. Kutluay, "A study for Performance Comparison of Different In- Memory Databases," 2013 7th Int. Conf. Appl. Inf. Commun. Technol., pp. 6-10, 2013.
[5] Y. Wang et al., "The Performance Survey of In Memory Database," IEEE 21st Int. Conf. Parallel Distrib. Syst., pp. 1-6, 2015.
[6] V. Kolomi?enko, M. Svoboda, and I. H. Mlýnková, "Experimental Comparison of Graph Databases," Proc. Int. Conf. Inf. Integr. Web-based Appl. Serv. - IIWAS '13, pp. 115-124, 2013. https://doi.org/10.1145/2539150.2539155.
[7] J. R. Lourenço, B. Cabral, P. Carreiro, M. Vieira, and J. Bernardino, "Choosing the right NoSQL database for the job: a quality attribute evaluation," J. Big Data, vol. 2, no. 1, p. 18, 2015. https://doi.org/10.1186/s40537-015-0025-0.
[8] J. Klein, I. Gorton, N. Ernst, P. Donohoe, K. Pham, and C. Matser, "Performance evaluation of NoSQL databases: A case study," PABS 2015 - Proc. 1st ACM/SPEC Int. Work. Perform. Anal. Big Data Syst., pp. 5-10, 2015. https://doi.org/10.1109/BigDataCongress.2015.83.
[9] D. Upeksha et al., "Comparison Between Performance of Various Database Systems for Implementing a Language Corpus," Commun. Comput. Inf. Sci., vol. 521, pp. 82-91, 2015. https://doi.org/10.1007/978-3-319-18422-7_7.
[10] M. Plechawska-Wojcik and D. Rykowski, "Comparison of Relational, Document and Graph Databases in the Context of the Web Application Development," Adv. Intell. Syst. Comput., vol. 429, pp. 79-88, 2016.
[11] J. Schindler, "Profiling and analyzing the I/O performance of NoSQL DBs," ACM SIGMETRICS Perform. Eval. Rev., vol. 41, no. 1, pp. 389-390, 2013. https://doi.org/10.1145/2494232.2479782.
[12] W. Naheman and J. Wei, "Review ofNoSQL Databases and Performance Testing on HBase," 2013 Int. Conf. Mechatron. Sci. Electr. Eng. Comput., pp. 2304-2309, 2013. https://doi.org/10.1109/MEC.2013.6885425.
[13] MongoDB, "Import Example Dataset." [Online]. Available: https://docs.mongodb.com/getting- started/shell/import-data/. [Accessed: 20-May-2017].