Journal of Engineering and Applied Sciences 14 (19): 7218-7222, 2019
ISSN: 1816-949X


# Performance Evaluation for MongoDB and Redis: A Comparative Study

[1]Safaa Alraddadi and [2]Sultan Almotairi
[1]Department of Computer Science, King Abdulaziz University (KAU), Jeddah, Saudi Arabia
[2]Department of Natural and Applied Sciences, Community College, Majmaah University,
Al Majmaah, Saudi Arabia, almotairi@mu.edu.sa

**Abstract:** NoSQL databases have been emerging as promising technology as well as the customary platform for big data application. These databases become an alternative approach for relational databases based on what it characterized as fast access data, limited query and efficient horizontal scalability. This study aims to investigate the performance for MongoDB and Redis databases in light of basic operations. The experimental result measure execution time for each operation which in turn helps to determine suitable NoSQL Model for a specific application because there is a little correlation between performance and data model. The results show Redis is better when the dataset is large and two databases have the same performance in insert operation when dataset is small. In case complex query MongoDB is the best choice.

**Key words:** SQL, NoSQL, MongoDB, Redis, performance, data model

## INTRODUCTION

Databases store and process increasingly large data sets generated from various online and offline sources (Reniers *et al*., 2017). The relational database offers great features and works well with structure data where it is the best solution for the critical transaction. Companies that not deals with a critical transaction and still use relational database they will face some problems in scaling as well as cost and complexity (Leavitt, 2010; Nyati *et al*., 2013).

Now a days with continued development of the internet and growing used of social network demand increased for databases that able to store, processing and retrieve a huge amount of different data in efficient ways also to overcome the limitation of relational database (Han *et al*., 2011). In light of this, some databases (non-relational) have been rise in the last few years known as NoSQL databases which offer high performance and high availability (Li and Manoharan, 2013). NoSQL dissipative data model handles unstructured data with a different format, large scale and parallel data processing (Nyati *et al*., 2013). Although, NoSQL is an umbrella for wide various of databases, these databases can be categories according to the support data model and four common type of subcategories are defined as (Kanade *et al*., 2014):

- Document based databases: data store as documents that used a different format such as XML or JSON This database flexible each document can have different fields
- Key-value based databases: this database organized as a pair of the key and value. Key is used to access and retrieved value, this means each key is a unique (Kaur and Rani, 2013)
- Column-based databases: structure similar to there lational database. Data store sets of columns in a contiguous way (Zafar *et al*., 2016)
- Graph based databases: represented data as the set of entities associated with linked. This database most used in social network

NoSQL database that adopted in this study MongoDB and Redis. MongoDB is classified as a document oriented databases use BSON and binary JSON format, to store data in documents. Features in MongoDB and relational are most the same both have rich query language, support conditional operator, regular expression, query an embedded document and array and complicated aggregate operations can be implemented by utility functions. Collection in MongoDB corresponding to the table in MySQL and document corresponding to the record. It is the best alternative for MySQL when application deals with a large and complicated query (Gu *et al*., 2015). Used in LinkedIn.

Redis is based on key-value storage does not have the table and predefine scheme like classical relational databases. It classified as in memory data where entire data load and operate in memory. It does not allow store complex multilevel documents. Redis handle many data type include string, hash, set list (Das, 2015). Used in Twitter, Stack Overflow and GitHub (Ramesh *et al*., 2016).

**Corresponding Author:** Sultan Almotairi, Department of Natural and Applied Sciences, Community College, Majmaah University, Al Majmaah, Saudi Arabia, almotairi@mu.edu.sa

This study contributed to evaluate MongoDB and Redis performance in insert, delete, select and update operations to help determine the appropriate database for specific requirements.

**Literature review:** Many research initiatives evaluate a performance of NoSQL as well as SQL databases to highlight choose of an appropriate database for each specific use case.

Contributed given by Hecht and Jablonski (2011), provide use case oriented to evaluation NoSQL. Paper represent important criteria to choose suitable databases for the specific requirement. They used performance for reading and write, data model, query possibility and concurrence to compared fourteen databases include MongoDB, Redis and Casandra.

Cornelia and others Gyorodi *et al*. (2015), made the comparative study between relational and non-relational databases. They choose MongoDB for implemented NoSQL and Oracle for implemented MySQL. A comparative study based on a performance of the basic operation (insert, delete, select and update) on dynamic structure forum. The results show MongoDB have efficient execution time than MySQL in all operations, if data and query are massive.

Yassien and Desouky (2016), evaluate operations performance, latency and throughput on a different databases system. MongoDB for NoSQL, HBase for Hadoop and MySQL for relational by conducted YCSB benchmark. They report HBase suitable in case application have a high update and insert operations. MySQL suited well with the application that requires most read operation. MongoDB typical use for the application that requires convenient read and write performance.

Another contributed by Gustavo and Anderson (Martins *et al*., 2015), the study carried out to evaluate Casandra and MongoDB influence of virtualization overhead generated. They implement full virtualization and paravirtualization techniques on different VMM. To conduct performance analysis in the virtual and physical machine by YCSB benchmark. They conclude MongoDB have better performance in select, update and load operations for both physical and virtual machine.

Qi (2014), aimed to analyze performance in read and update operations conduct by Amazon EC2 cloud benchmark. MongoDB and Rika were selected to be used in the experiment. MongoDB deployment where master node used for benchmark application, a configuration server and for mongo process other nodes have one mongo process to execute. Rika deployment where master node used for benchmark application and remaining nodes have one Rika process to execute. Investigation shows Rika better when dataset is large and MongoDB outperforms when dataset is small.

Alexandru and others, Gu *et al*. (2015), present differences between Oracle and MongoDB. Comparison criteria based on thermotical differences, features, integrity, system requirements, query and insertion execution time. They conclude MongoDB for all operations more efficient when dealing with a large amount of data while Oracle provides better performance with a little amount of data.

This study contributed measure performance of basic operations in Redis and MongoDB databases. As we can see from related research, the issue is difficult to determine database that suitable for the specific application. In light of this, need to identify performance for two databases and kind of application that typical use with.

## MATERIALS AND METHODS

**Experimental test:** The test result depends on the computer on which the test was performed as well as the version of databases. All experiments conducted in this study where carried out on the computer that holds the characteristics as well as databases version that indicated in Table 1.

Script wrote in both databases with PHP. Dataset consisted of company staff information with the different size of records (1000,100000,1000000) and 12 fields. The test consists of four basic operations that can be performed in any databases: insert, select, update and delete.

The test is beginning with creation of databases with null contents. The structure of the two databases is different one dealing with documents and the other dealing with a keys but they have the same number of keys and documents.

First, insert staff information in both databases. In MongoDB used PHP function insert many to insert many staff records at the same time. While function that responsible to insert data in Redis is Hmset.

Table 1: Computer characteristics and setting

| Setting | Values |
|---|---|
| Operating system | Windows 8 |
| Processors | 3.4 GHz |
| RAM | 8 GB DDR3 |
| Core | Intel Core i7 |
| Hard disk | 20 GB |
| MongoDB | Version 3.4.4 |
| Redis | Version 3.2.9 |

For update operation three different types of update used. The first type involves updating the staff name based on the age that matches chosen age (27). The second type update balance based on the date of registered in the company. The third type update company name for each staff that matches 27 age and the first name Robert. For delete operation in MongoDB used function delete many to delete all staff that matches female gender and eyecolor is blue. In Redis delete data using function HDEL.

Test two different select operations. Select divide into two categories simple and complex query. The first type of select retrieved all staff information by staff age. The second type retrieved staff ID by balance. The first and second type classified as a simple query which selected data of only one object type. The third type is contained nested query and multiple object type and classified that as a complex query.

## RESULTS AND DISCUSSION

**Performance analysis and discussion:** For the best comparison between the two database engines, run some tests and measure performance of execution time for each operation by microtime function which calculates the time from the beginning of the operation until its completion. The results of all experiments appear in Fig. 1-7. Each figure compares the average time, in milliseconds, of MongoDB implementation versus Redis implementation of the given job. Run each operation five-times to reduce the skewing effect on the average time.

Thereover, after running MongoDB script and Redis script, Fig. 1 shows execution time when inserted 1000 staff records into both databases in 0.124 msec time where added 100000 staff records into MongoDB in 14.25 msec time while in Redis, time was just 7.35 msec time. Execution time for the MongoDB when inserted 1000000 staff records is 50.12 msec time where Redis take 23.50 msec time to execute. Redis is faster than MongoDB when a dataset is large while both databases have same performance when the dataset is small. As noted earlier, inserts data objects in Redis using a hash which in turn not require additional data to insert.

Figure 2-4 illustrates execution time for three different cases in update operation. For the first case in Fig. 2, update staff age based on the first name. Among different tested dataset, Redis exhibited a performance of 0.0003 msec time when dataset 1000 records while taking 0.04 msec time and 1.05 msec time when dataset 100000 and 1000000 records, respectively. The MongoDB shows inferior performance comparing with Redis.

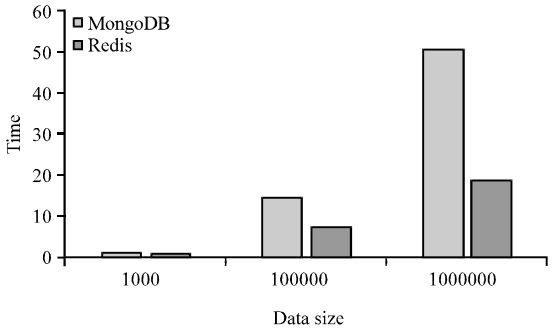


Fig. 1: Inserted performance

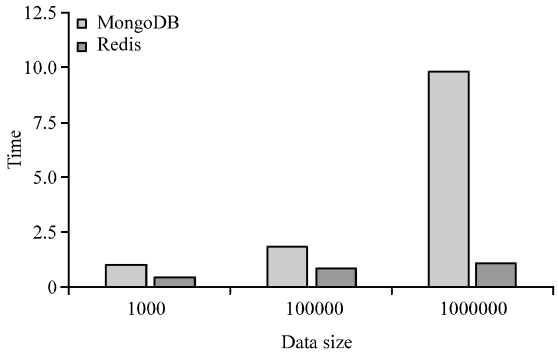


Fig. 2: Update staff name

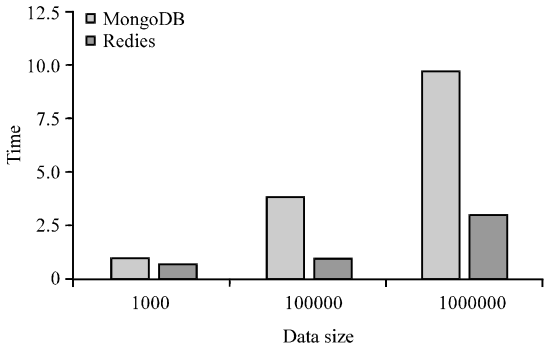


Fig. 3: Update balance

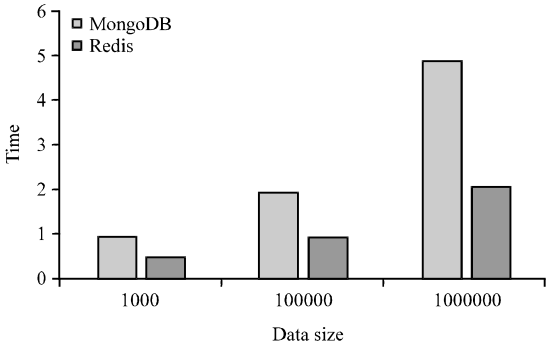


Fig. 4: Update company name

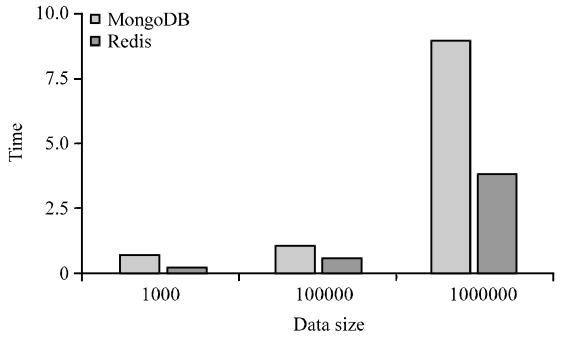


Fig. 5: Deleted performance

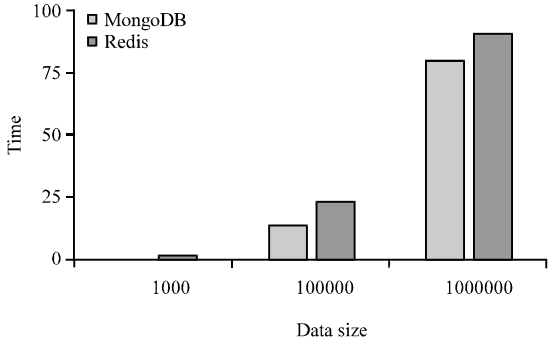


Fig. 6: Retrieved staff information

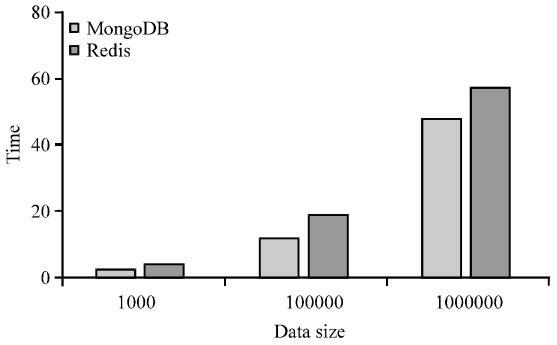


Fig. 7: Retrieved staff ID

Redis outperformed MongoDB in the second and third case as shown in Fig. 3 and 4. Analyzing the result of execution time for three different types of update operation a good performance achieved by Redis database in both large and small dataset. This database based on volatility memory for data storage and retravel which allow Redis to expend lower execution time than MongoDB.

Execution time increase as dataset increased in delete operation as illustrated in Fig. 5. Moreover, MongoDB shows a relatively slow increased in delete operation compare to Redis. As for Redis, starting from 1000 to 1000000 datasets, delete shows tremendous performance.

During the first query which retrieved staff information based on age and second query retrieved company name by staff ID, the time taken to select the record increased directly in Redis while it was a gradual increase in MongoDB with an increasing number of records (1000, 100000, 1000000) on the same computer configuration. Hence, MongoDB shows better result during query phase over Redis. The results for select records from both MongoDB and Redis databases are shown in Fig. 6 and 7.

## CONCLUSION

The continued development of the web and growing use of the social network needed databases able to store and process a huge amount of data effectively, the relational database is facing many new challenges to respond to the massive read and write without any noticeable latency. NoSQL databases emerge and become popular used in many productions. In this study, we analyzed and evaluated a performance of insert, delete, update and select for two of the NoSQL databases: MongoDB and Redis to determine which database most suitable for a certain use case. In the experiments, test the execution time according to different dataset size. Redis outperformed MongoDB in insert operation when data size is large. While both databases have the same performance with a small amount of data. Analyzing update operation showes Redis is better than MongoDB in both small and large size of data. MongoDB shows a relatively slow increased in delete operation while Redis shows an immense performance. Overall analysis conclude Redis shows best results for most seniors but it is not the best choice for the complex and nested query.

## REFERENCES

Das, V., 2015. Learning Redis. PACKT Publishing, Birmingham, England, UK., ISBN:978-1-78398-012-3, Pages: 293.

Gu, Y., S. Shen, J., Wang and J.U. Kim, 2015. Application of nosql database mongoDB. Proceedings of the 2015 IEEE International Conference on Consumer Electronics-Taiwan, June 6-8, 2015, IEEE, Taipei, Taiwan, pp: 158-159.

Gyorodi, C., R. Gyorodi, G. Pecherle and A. Olah, 2015. A comparative study: MongoDB vs. MySQL. Proceedings of the 2015 13th International Conference on Engineering of Modern Electric Systems (EMES), June 11-12, 2015, IEEE, Oradea, Romania, pp: 1-6.

Han, J., E. Haihong, G. Le and J. Du, 2011. Survey on NoSQL database. Proceedings of the 2011 6th International Conference on Pervasive Computing and Applications (ICPCA), October 26-28, 2011, IEEE, Port Elizabeth, South Africa, ISBN:978-1-4577-0208-2, pp: 363-366.

Hecht, R. and S. Jablonski, 2011. NoSQL evaluation: A use case oriented survey. Proceedings of the 2011 International Conference on Cloud and Service Computing, December 12-14, 2011, IEEE, Hong Kong, China, pp: 336-341.

Kanade, A., A. Gopal and S. Kanade, 2014. A study of normalization and embedding in MongoDB. Proceedings of the 2014 IEEE International conference on Advance Computing Conference (IACC), February 21-22, 2014, IEEE, Gurgaon, India, pp: 416-421.

Kaur, K. and R. Rani, 2013. Modeling and querying data in NoSQL databases. Proceedings of the 2013 IEEE International Conference on Big Data, October 6-9, 2013, IEEE, Silicon Valley, California, pp: 1-7.

Leavitt, N., 2010. Will NoSQL databases live up to their promise? Computer, 43: 12-14.

Li, Y. and S. Manoharan, 2013. A performance comparison of SQL and NoSQL databases. Proceedings of the 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), August 27-29, 2013, IEEE, Victoria, Canada, pp: 15-19.

Martins, G., P. Bezerra, R. Gomes, F. Albuquerque and A. Costa, 2015. Evaluating performance degradation in NoSQL databases generated by virtualization. Proceedings of the 2015 Latin American Conference on Network Operations and Management Symposium (LANOMS), October 1-3, 2015, IEEE, Joao Pessoa, Brazil, pp: 84-91.

Nyati, S.S., S. Pawar and R. Ingle, 2013. Performance evaluation of unstructured NoSQL data over distributed framework. Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), August 22-25, 2013, IEEE, Mysore, India, pp: 1623-1627.

Qi, M., 2014. Digital forensics and NoSQL databases. Proceedings of the 2014 11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), August 19-21, 2014, IEEE, Xiamen, China, pp: 734-739.

Ramesh, D., A. Sinha and S. Singh, 2016. Data modelling for discrete time series data using Cassandra and MongoDB. Proceedings of the 2016 3rd International Conference on Recent Advances in Information Technology (RAIT), March 3-5, 2016, IEEE, Dhanbad, India, ISBN:978-1-4799-8580-7, pp: 598-601.

Reniers, V., D.V. Landuyt, A. Rafique and W. Joosen, 2017. On the state of NoSQL benchmarks. Proceedings of the 8th ACM/SPEC International Conference on Performance Engineering Companion ICPE'17 Companion, April 22-26, 2017, ACM, L'Aquila, Italy, pp. 107-112.

Yassien, A.W. and A.F. Desouky, 2016. RDBMS, NoSQL, Hadoop: A performance-based empirical analysis. Proceedings of the 2nd Africa and Middle East Conference on Software Engineering, May 28-29, 2016, ACM, Cairo, Egypt, pp: 52-59.

Zafar, R., E. Yafi, M.F. Zuhairi and H. Dao, 2016. Big data: The NoSQL and RDBMS review. Proceedings of the 2016 International Conference on Information and Communication Technology (ICICTM), May 16-17, 2016, IEEE, Kuala Lumpur, Malaysia, pp: 120-126.