Document Oriented NoSQL Databases: An Empirical Study

Omji Mishra(✉), Pooja Lodhi, and Shikha Mehta

Department of Computer Science, Jaypee Institute of Information Technology, Noida, India
omji.mishra@yahoo.com, poolodhi@gmail.com,
shikha.mehta@jiit.ac.in

Abstract. In today's era, organizations are developing applications which continuously and rapidly generates large amount of data. In this world of big data, NoSQL databases are rapidly becoming popular for storing information among organizations. Therefore, it is essential for an organization to choose a database which is compatible and efficient for their applications. To choose a correct database, it is essential to examine the performance of various databases under diverse workload conditions. In this paper, we are examining different document oriented databases on various workloads, so that we can categorize them according to application need. The evaluation has been performed on four NoSQL document oriented databases: MongoDB, ArangoDB, Elastic search and OrientDB with the help of Yahoo Cloud Service Benchmark (YCSB), which is a popular benchmark tool. Comparison is done in two parts: In the first part, all the databases are compared for single thread on the basis of throughput and runtime. Here, Mongo DB shows better results with highest throughput and lowest runtime among all the databases. In the second part, a thorough analysis is done in which MongoDB and ArangoDB are compared for some threads on different workloads. Here also, Mongo DB outperforms ArangoDB with a high percentage.

Keywords: Big data · MongoDB · ArangoDB · OrientDB · Elastic search
Document oriented databases · YCSB

1 Introduction

1.1 A Subsection Sample

Big data is a term used for large and complex datasets that cannot be processed with the traditional data processing software. Analysis of these large datasets is relevant in finding patterns, spotting trends, preventing crime and enhancing user experience. To manage this big data NoSQL databases are designed. NoSQL databases are those databases which provides the mechanism of storing data in key-value, wide column, graph, document rather than storing data in tabular form as in relational databases. It is also called "Not only SQL". It uses data structures that are faster in terms of relational databases. Examples of NoSQL databases are: Riak, Couchbase, MongoDB, Cassandra, HBase, Neo4j, OrientDB, etc.

B. Panda et al. (Eds.): REDSET 2017, CCIS 799, pp. 126–136, 2018.
https://doi.org/10.1007/978-981-10-8527-7_12

Document databases are subset of NoSQL databases. It is used for storing and managing semi-structured data. In document databases, the term "document" refers to a block of XML or JSON. Collections of documents are used to store data [6]. These databases rely on the internal architecture in the document in order to extract metadata for further optimization.

The Yahoo! Cloud Serving Benchmark (YCSB) is open source software used for evaluating the capabilities of computer program. The goal of YCSB is to provide a platform for performance comparison of NoSQL databases. YCSB framework consists of a workload generating client and a collection of standard workloads that cover interesting aspects of performance space [1, 8]. An important feature of YCSB is its ability to define new workload types.

Today there are various types of NoSQL databases based on different architecture. Databases under different categories preform differently. But it must be understood that various database under same category also perform differently. In this research work, we are doing performance comparison of different NoSQL document-oriented databases like MongoDB, OrientDB, ArangoDB and Elastic Search. Yahoo! Cloud Serving Benchmark framework is used for comparisons of these databases. We have selected workload a, b, c, d and f to compare databases. The workload 'a' contains large number of update operation therefore it is also called update heavy workload whereas; workload 'b' is read heavy. Workload c is only contains read operation while workload 'd' is read latest. Workload 'f' can be used to simulate read modify and updates operation in a group. The detailed explanation of different workloads is provided in Table 1. We have done deep analysis of MongoDB and ArangoDB by comparing their throughput and runtime for different workloads for 1 million records. We have also performed comparison between other databases on different workloads for 1 thousand records.

The paper consists of different sections. In Sect. 1, introduction is provided which gives an overview of main tools and databases used in paper. Section 2, consist of the literature survey. In Sect. 3, we have done background study related to the research work. Section 4 contains experimental setup and the results of all databases with comparison and analysis. In Sect. 5, we have concluded our research work. In our last section that is Sect. 6 we have discussed some of the limitation that we found during the experiment.

2 Literature Survey

Various Studies are conducted to understand how different databases behave under different conditions [2–5, 7, 9–11]. A study to assess the performance difference between RDBMS and NoSQL databases systems was performed in [3]. It also discussed about the optimal design for enhanced functionality when NoSQL databases are used. They created their own dataset and used CRUD operation to perform comparison between NoSQL and RDBMS databases. To perform all these operations, they developed their own program.

In [5], three open source NoSQL databases: MongoDB, Cassandra and HBase, were analyzed in detail. An analysis on the performance compares the NoSQL databases on their ability to scale under different workload conditions and with different dataset sizes.

In order to do the analysis, they generated records of 10 and 20 fields. They also used zipfian strategy to distribute the data. At the end, they provided a benchmark report about which database should be used in what condition.

Literature [4], briefly explained some of the NoSQL unstructured databases and presented performance analysis of MongoDB. It compared the time required for insertion into different databases as well as searching for different number of threads in database with different number of entries. The paper also presented the importance of Sharding and Configuration of the cluster for MongoDB.

The author developed near real-time Twitter data warehouse by NoSQL database, Cassandra, and compare its storing and querying performance [11]. The results show that Cassandra performs better in storing data, whereas it is slower in querying.

From the above research papers, we observe that although abundance of work has been done to compare Relational and NoSQL databases, different types of NoSQL databases and to measure performance of a particular database. But, there has been limited attempt to measure the performance of similar databases. Performance evaluation of similar databases needed attention because it is essential to choose a correct database from a set of similar databases. So, we have evaluated different document-oriented NoSQL databases. In order to do so we have analyzed MongoDB, ArangoDB, OrientDB and Elastic Search.

3 Background

3.1 NoSQL Databases

The databases which provide the different mechanism of storing data other than tabular form are called NoSQL databases. It uses data structures that are faster in terms of relational databases. There are majorly four different types of NoSQL databases.

Key-Value Store is based on the concept of key-value pair and is implemented using big hash tables. This database is well-suited for applications in which small reads and writes are performed frequently.
Document-Based Store uses the central concept of document. This database has the property of being flexible. Therefore they are appropriate solution for application in which data keep varying in terms of attributes.
Column-Based Store data is stored in cells and these cells are grouped in columns. It is different from the traditional way of storing data as rows. Columns are further combined to form column families. These databases are well-suited for data which is distributed to various sites.
Graph-Based Database data is represented in graphical from. The problems that are well represented with networks of connected entities for such scenarios graph databases are most suitable.

3.2 Document Oriented Databases

Document oriented database uses the central concept of document. While every document oriented database implements document in different forms, they all assume document encapsulation and data encoding in some standard formats. Documents can be compared to object concept of programming language. Different encoding used in document oriented databases are XML, YAML, JSON, BSON, etc. The core operations of document-oriented databases are Creation (Insertion), Retrieval (Search or Query), Update (or Edit) and Delete.

Mongo dB is a free open source document oriented database. It uses JSON-like document format.
Arango dB is a NoSQL multi-model open source database. It has been designed specifically to allow key/value, document and graph data to be stored together and queried with a common language. It also uses JSON as a default storage format.
Elastic Search It is an open source search engine based on Lucene. It is developed in JAVA.
Orient dB is a NoSQL database written in JAVA. It is a multi-model database so it is sometimes also considered as a graph database instead of document-oriented database.

3.3 YCSB Workload

Workloads in the benchmark tool are defined by assigning different distributions to two main parameters: which operation we perform and which record is read or write. While doing analysis, loading a database takes longer time than running a database. All the YCSB core package workloads uses the same dataset so that it is possible to load the database only once and then run all the required workloads. Since workload d and e insert records so if database writes are likely to impact the operations of other

Table 1. YCSB workloads [1].

Workload	Operations	Application example
A-update heavy	Read: 50% Update: 50%	Session store where recent actions are recorded in a session
B-read heavy	Read: 95% Update: 5%	Photo tagging
C-read only	Read: 100%	User profile cache, where profiles are constructed elsewhere
D-read latest	Read: 95% Insert: 5%	User status updates; people always want to read the latest statuses of other people
E-short ranges	Scan: 95% Insert: 5%	Threaded conversations
F-read-modify-write	100% Read Modify Write	User database or user profile updates where records are read and modified or to record activity performed by user

workloads then it may be necessary to re-load the database [1]. YCSB has two phases, load and run. In load phase, data is inserted into the source database, which represents the initial state of the database. Then, in the run phase, different workload performs different operation like modify previously uploaded data, insert new data or read the data [10].

4 Experiments and Results

4.1 System Configuration

The experiment was performed on a single machine consisting of Dual core AMD A4-3330MX APU CPU at 2.2 GHz. The RAM of the system was 6 GB and the main memory size was 500 GB. OS of system was Ubuntu 12.04 LTS.

4.2 Testing Process

For the testing process, YCSB tool, Mongo dB, Arango dB, Orient dB and Elastic Search databases were installed on the system. The testing process was divided into two parts. Firstly, we have run the YCSB tool on different databases for different workloads with record count of thousand. We have taken workloads a, b, c, d and f (see Table 1) for evaluation as we want to analyze how large amount of read and write operation affect the databases. Workload e was a short ranges workload which was not much compatible for our analysis as it requires deleting database each time before executing the run command. Secondly, we have conducted the deep analysis of MongoDB and ArangoDB with a dataset having record count of 1 million. Steps for testing are as follows:

1. For each experiment, workload was loaded into the database through terminal.
2. Then the corresponding workload was run for different number of client threads (1, 3, 6, 10, 20, 32, 50, 64 and 128).

   Afterwards, the results were interpreted with the help of graphs.

4.3 Results

In this section we evaluate the results of the benchmark testing of different NoSQL databases. First we evaluated various above mentioned document oriented databases on different workloads, for a single thread. Then we compared performance of MongoDB with ArangoDB on different workloads, by varying the thread count.

Performance of Document Oriented Databases. MongoDB, ArangoDB, Elastic search and OrientDB are compared on parameter of runtime and throughput to understand their behavior when different types of operation are performed on them.

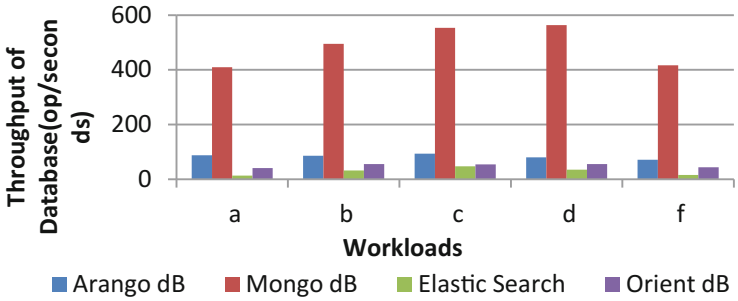

Fig. 1. Workload vs throughput for various databases.

Figure 1 shows that among all the document oriented databases, Mongo dB has highest throughput value for each workload. Elastic Search, ArangoDB and OrientDB has very low throughput compared to MongoDB which show that they cannot handle large number operation of any kind per second.

Figure 2 clearly shows that, Elastic search has highest runtime for all workloads. This implies that Elastic search is not an appropriate solution for any application where time is an important factor. This also shows that more time is required to perform update operation in Elastic search. OrientDB and ArangoDB show similar behavior for all the workloads, while MongoDB perform best with least runtime.

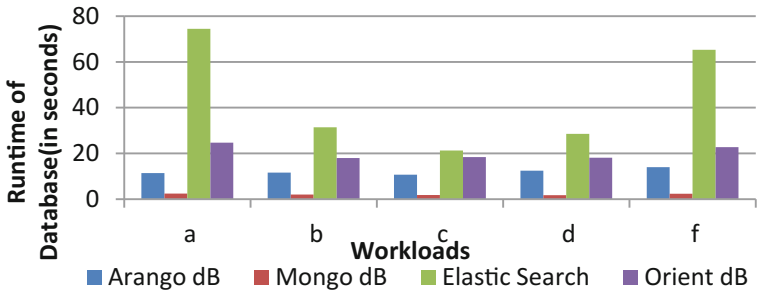

Fig. 2. Workload vs runtime for various databases.

MongoDB vs ArangoDB. Here we analyze how MongoDB and ArangoDB perform when the thread are increased for different type of workloads.

Update heavy workload. The graphs for the execution time of 50% Read and 50% write are given in the above Fig. 3(a). From fig. we observe that MongoDB has low runtime value for every thread as compared to ArangoDB. Low runtime implies that MongoDB is a suitable database to use where there is need of heavy update operation. Graph also shows that thread 1 runtime value for ArangoDB is quite high as compared to its other threads, which means that for heavy update operation for single thread ArangoDB is not a good option.

Figure 3(b) depicts that throughput value of MongoDB is better enough as compared to that of ArangoDB which implies that for heavy update operation MongoDB is a better choice. Heavy update operation for thread 6 and 10 has highest throughput value i.e., it performs highest no. of operation per sec for heavy update operation. So, for applications like session store where recent action are recorded MongoDB is better document oriented database to use.

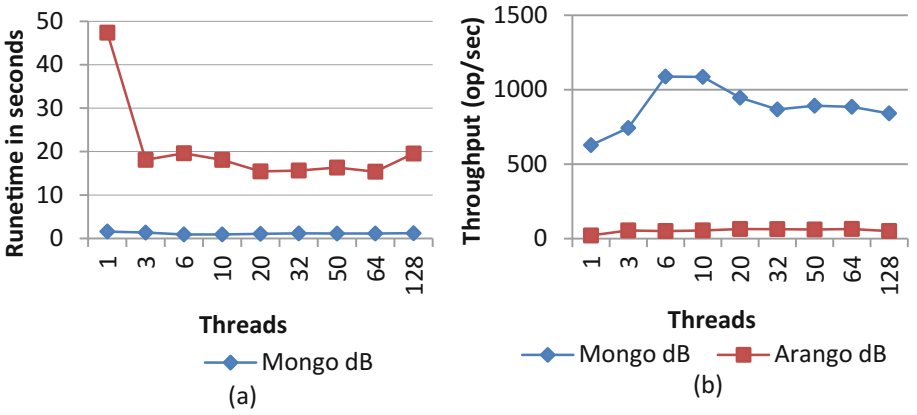

Fig. 3. (a) Threads vs runtime for workload A. (b) Threads vs throughput for workload A

Read mostly workload. Figure 4(a) shows graph of runtime v/s threads for workload b. It is observed that MongoDB has low runtime value for every thread as compared to ArangoDB. This means that MongoDB is more efficient than ArangoDB where read operation happens more frequently. We can also observe that thread 128 runtime value for ArangoDB is quite high as compared to its other threads, which means that for heavy read operation for large number of threads, ArangoDB is not a good match.

From Fig. 4(b), we see that the throughput value of MongoDB is better enough as compared to that of ArangoDB implying that for heavy read operation MongoDB is a good choice. On the other hand when number of threads increases, value of throughput starts decreasing after reaching peak value. Thus MongoDB can be used for photo tagging application like Facebook or investment forecasting where large amount of data is read and analyzed to extract some knowledge.

Read only workload. Given the situation where database execute only read operations like user profile cache, MongoDB has lowest value (see Fig. 5(a)) implying that MongoDB is an efficient database for an application where there is need of only read operation. Thread 1 runtime is quite high for ArangoDB as compared to its other threads, which means that for read only operations on single thread it is not a good database. Figure 5(b) depicts that throughput value of MongoDB is better as to that of ArangoDB.

Read latest workload. From Fig. 6(a) we can see that as usual MongoDB has low runtime value for every thread as compared to ArangoDB. Thus MongoDB is a suitable database to use where there is need of read latest operation. On the other hand, runtime value for ArangoDB is quite high making it not desirable a database for reading latest record. Figure 6(b) depicts that operation performed per second in MongoDB is more, so it is better choice for application where reading latest data is done like, inbox messages in a mailbox or latest posts on Facebook or latest tweets on twitter.

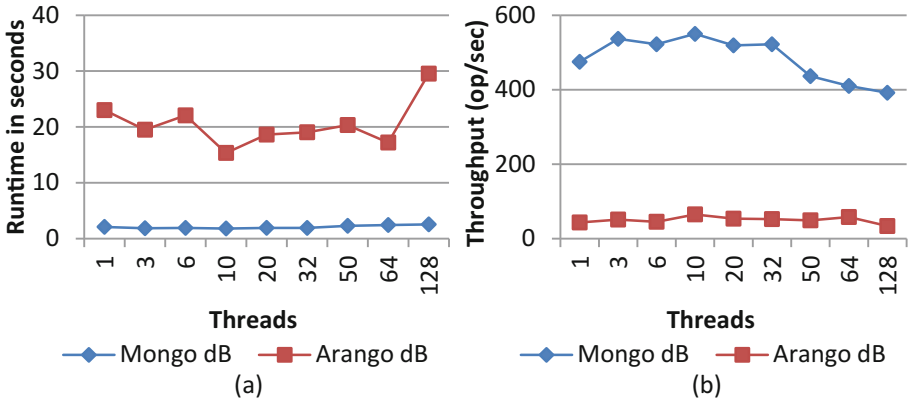

Fig. 4. (a) Threads vs runtime for workload B. (b) Threads vs throughput for workload B

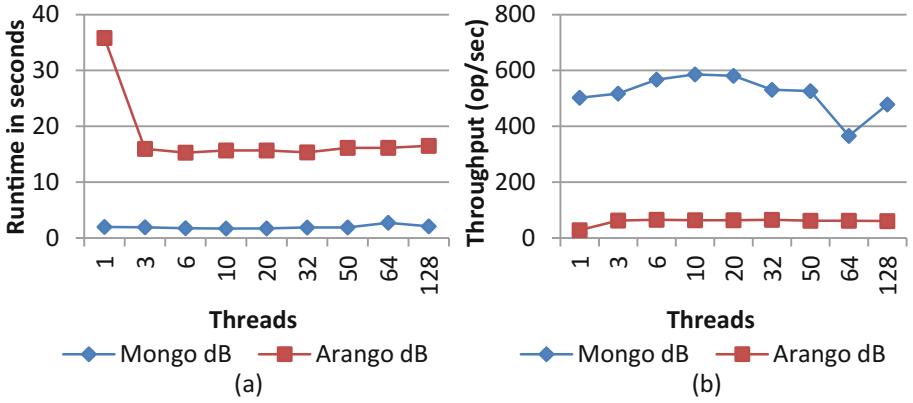

Fig. 5. (a) Threads vs runtime for workload C. (b) Threads vs throughput for workload C.

Read modify write workload. Figure 7(a) clearly shows that ArangoDB has highest runtime for all threads. Therefore it is not an appropriate solution for any application where low running time is preferred. It also shows that more time is required to perform read, modify and write operation in ArangoDB for all threads, while MongoDB requires less running time.

The performance behavior exhibited by workload 'f' is given in Fig. 7(b). It shows that among MongoDB and ArangoDB, MongoDB has highest throughput value. ArangoDB has very low throughput as compared to MongoDB which shows that it cannot handle large number operation of read, modify and write per second. Thus, MongoDB is good option to use in scenario such as money transaction where balance in a bank account is read then modified and then finally gets updated in the databases.

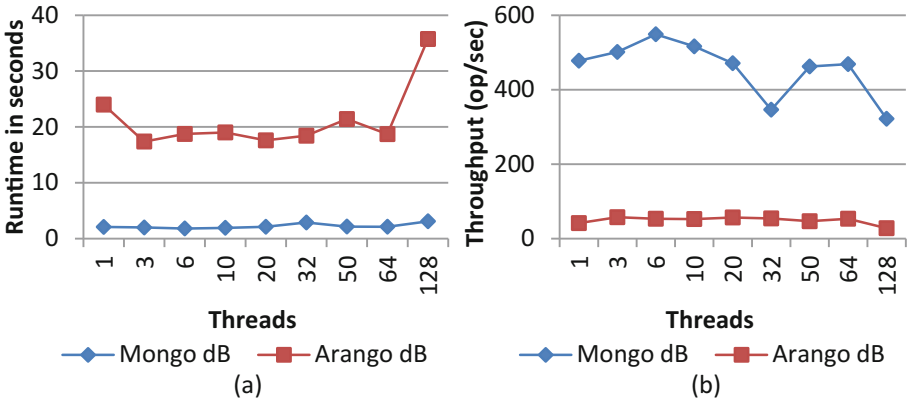

Fig. 6. (a) Threads vs runtime for workload D. (b) Threads vs throughput for workload D.

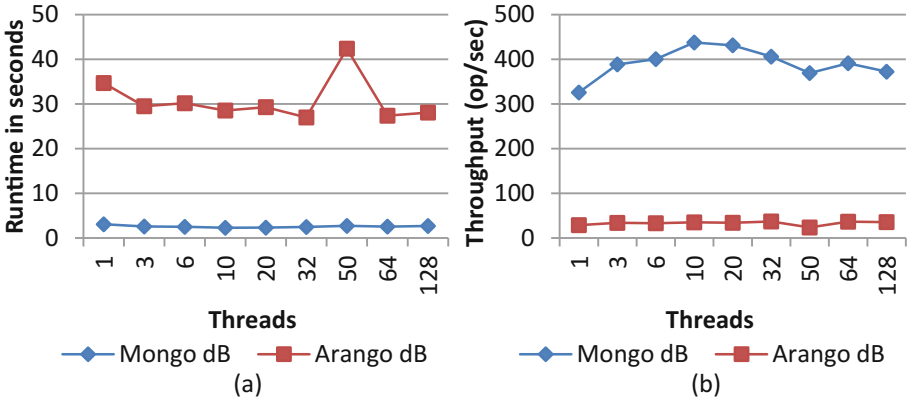

Fig. 7. (a) Threads vs runtime for workload F. (b) Threads vs throughput for workload F

Overall experimental analysis and research concluded that MongoDB outperform other databases because it's a single model database which is a specialized solution for documents. Its performance increases for single reads and writes. This is because of document model due to which data get more localized and this in turn reduces the need to join separate results. The feature of automatic sharding and replication leads to strong consistency and availability which result in higher performance and scalability of the database.

5 Conclusion

Today large number of NoSQL databases exist which are beneficial to certain types of operations only. So it is necessary to evaluate the behavior of different databases before using them for required operations. In this era large amount of information is generated over web and most of it is in form of documents. So in this research work, different document oriented NoSQL databases like: MongoDB, ArangoDB, Elastic Search and OrientDB were analyzed in detail. An analysis was done on the basis of throughput and runtime of the databases on different number of threads. MongoDB and ArangoDB were deeply analyzed by running them on 1 million records for every thread. The analysis helped us to realize that MongoDB is the best suited document oriented database for every type of workload. This is because of the fact that its runtime is low and operation performed per seconds is high when run for different number of threads.

6 Limitation

Some of the limitations were found while performing the experiment in the tools and the databases. Firstly, MongoDB and ArangoDB run perfectly for 1 million records for any number of thread in given system specification, while Elastic Search and OrientDB were limited to single thread only. Secondly, OrientDB and Elastic search cannot load a large amount of data while being used with YCSB tool. It throws Direct Memory Buffer Error implying that the system memory is not enough to run large number of records. Thirdly, when MongoDB was run for workload f, it was required to load the database for each thread. This was because of the existence of duplicate key while insertion operation performed by workload.

References

1. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with YCSB. In: Proceedings of 1st ACM Symposium on Cloud Computing (SoCC 2010), pp. 143–154. ACM, New York (2010)
2. Comparing NoSQL Databases with YCSB Standard Benchmark report by AVALON (2016)
3. Jung, M.G., Youn, S.A., Bae, J., Choi, Y.L.: A study on data input and output performance comparison of MongoDB and PostgreSQL in the big data environment. In: 2015 8th International Conference on Database Theory and Application (DTA), pp. 14–17 (2015)
4. Truică, C.O., Boicea, A., Rădulescu, F., Bucur, I.: Performance evaluation for CRUD operations in asynchronously replicated document oriented database. In: International Conference on Control Systems and Computer Science, pp. 191–196 (2015)
5. Swaminathan S.N., Elmasri R.,: Quantitative analysis of scalable NoSQL databases. In: IEEE International Congress on Big Data, pp. 323–326 (2016)
6. Scofield, B.: NoSQL, death to relational databases. In: Presentation at CodeMash Conference, Sandusky, Ohio (2010)
7. Klein, J., et al.: Performance evaluation of NoSQL databases: a case study. Paper presented at 1st Workshop on Performance Analysis of Big Data Systems, Austin, Texas, USA (2015)

8. Barata, M., Bernardino, J., Furtado, P.: YCSB and TPC-H: big data and decision support benchmarks. In: IEEE BigData Congress, pp. 800–801 (2014)
9. Nyati, S.S., Pawar, S., Ingle, R.: Performance evaluation of unstructured NoSQL data over distributed framework. In: International Conference on Advances in Computing, Communications and Informatics, pp. 1623–1627 (2014)
10. Schmidt, F.M., Geyer, C., Schaeffer-Filho, A., et al.: Change data capture in NoSQL databases: a functional and performance comparison. In: IEEE Symposium on Computers and Communication (2015)
11. Murazza, M.R., Nurwidyantoro, A.: Cassandra and SQL database comparison for near real-time Twitter data warehouse. In: International Seminar on Intelligent Technology and Its Applications (2016)