J Inf Process Syst, Vol.10, No.3, pp.355~364, September 2014
http://dx.doi.org/10.3745/JIPS.04.0006

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

Improving Database System Performance by Applying NoSQL

Yong-Lak Choi*, Woo-Seong Jeon** , and Seok-Hwan Yoon***

Abstract—Internet accessibility has been growing due to the diffusion of smartphones in today's society. Therefore, people can generate data anywhere and are confronted with the challenge that they should process a large amount of data. Since the appearance of relational database management system (RDBMS), most of the recent information systems are built by utilizing it. RDBMS uses foreign-keys to avoid data duplication. The transactions in the database use attributes, such as atomicity, consistency, isolation, durability (ACID), which ensures that data integrity and processing results are stably managed. The characteristic of RDBMS is that there is high data reliability. However, this results in performance degradation. Meanwhile, from among these information systems, some systems only require high-performance rather than high reliability. In this case, if we only consider performance, the use of NoSQL provides many advantages. It is possible to reduce the maintenance cost of the information system that continues to increase in the use of open source software based NoSQL. And has a huge advantage that is easy to use NoSQL. Therefore, in this study, we prove that the leverage of NoSQL will ensure high performance than RDBMS by applying NoSQL to database systems that implement RDBMS.

Keywords—Database System Performance, NoSQL, NoSQL Practical Use

1. Introduction

Relational database management system (RDBMS) ensures data integrity using foreign-keys and data consistency to support transactions. Also, it has very high reliability and portability because it supports standard structured query language (SQL) [1]. However, there are some constraints in the usability of the database system that require high performance or distributed processing database systems. For these reasons, when RDBMS emerged, it was used in a limited manner because of performance degradation. However, the spread of the RDBMS is used in conjunction with improved performance of hardware, so most recent information systems had to build a RDBMS.

NoSQL is the general name for the collection of databases that do not use SQL or a relational data model [2]. NoSQL is a useful database for application development productivity increase and for dealing with large amounts of data. In particular, it is used for rapid data storage, increase of data concurrency and easy integration of the data.

According to the report by Digital Universe, the amount of digital information is expected to

※ This paper was supported by the Semyung University Research Grant of 2014.
Manuscript received August 29, 2013; accepted March 20, 2014.
Corresponding Author: Seok-Hwan Yoon (ysh1213@semyung.ac.kr)
* Graduate School of Software, Soongsil University, Seoul, 156-743, Korea (ylchoi58@ssu.ac.kr)
** POSCO ICT Inc., Seoul, 135-777, Korea (jeonwoosung@gmail.com)
*** Department of Computer Sciences, Semyung University, Chungbuk, 390-711, Korea (ysh1213@semyung.ac.kr)

more than double every two years. From this prospective, it is predicted that the most frequently used RDBMS will reach a limit of processing rapidly growing data [3]. The NoSQL data processing technique emerged as a solution. The schema of NoSQL is not fixed, and NoSQL is used as a method of storing data using key-value. Also, it does not generally guarantee the integrity of the database. In addition, it ensures higher performance than RDBMS by allowing data duplication and the high throughput of database systems by allowing data distributed processing to occur through horizontal expansion.

An information system should utilize accurate data to provide correct information for processing mission critical tasks. However, data consistency can be abandoned to improve performance and user convenience. Accordingly, the utilization of NoSQL can reduce the performance time of transactions in RDBMS and improve the entire performance in an information system.

This research presents a system operated by RDBMS and a model that utilizes NoSQL. We then compare the performance with a system built utilizing existing RDBMS. By doing this, we proved that the system with NoSQL has higher performance than an information system that uses RDBMS.

2. RELATED RESEARCH

An RDBMS transaction should be processed with both atomicity and consistency. Thus, database transactions should not violate any constraint conditions. Even if one transaction does not have a problem, another transaction interference can cause an erroneous result. Faulty transaction can result from update anomalies, incomplete dependent problems, and inconsistent analysis problems [4].

RDBMS solves these problems through locking, which is a simultaneous operation control method. Locking is the method for when it needs to be ensured that a tuple will not changed in unpredictable way while one transaction is performed, other transactions cannot access a corresponding tuple by obtaining the lock before work for corresponding tuple. This technique is used to control the transaction as it can ensure the consistency of the database, but it can also rapidly degrade the performance of the database systems.

Until recently, research on NoSQL has been led by large Internet companies, which cannot process services by using existing RDBMS. Main NoSQL solutions are Google's BigTable, Amazon's Dynamo, Facebook's Cassandra, and so on. BigTable is Google's database for handling petabytes of data. It is used in Gmail, YouTube, Google Analytics, Google Finance, Orkut, Personalized Search, and Google Earth [5]. Dynamo assures the continuous increase of data, high availability of the data utilization, horizontal scalability of large data and high performance [6]. Cassandra supports high scalability, high availability, partition tolerance, consistency, etc. [7].

NoSQL makes use of the Sharding method, which stores divided data into other servers. The unit of data is a shard, which is divided in Sharding, and dispersing and storing each shard that is split horizontally into multiple servers can enable to data store and process more data. Sharding techniques are Feature-based Shard, Key-based Sharding, the Lookup table, and so on [2].

NoSQL cannot guarantee all three (Consistency, Availability, Partition tolerance) characteristics under the influence of the CAP theory. The CAP theory was published at the

Principles of Distributed Computing of ACM Symposium by Eric Brewer in 2000, and later was proved by Seth Gilbert and Nancy Lynch of Massachusetts Institute of Technology (MIT). The availability and consistency is used as the RDBMS to NoSQL. In the case of NoSQL it is divided into products that ensure consistency and partition tolerance and products that ensure availability and partition tolerance [8]. There are products that ensure consistency and partition tolerance, such as Dynamo, Cassandra, CouchDB, etc., and products that ensure availability and partition tolerance, such as BigTable, MongoDB, MemcacheDB, HBase, and so on.

Until now, most of the research that has been conducted has been done to implement features required by companies. Although there is a growing interest in NoSQL, studies on its effectiveness do not advanced yet.

3. Performance Comparison

3.1 Test Methods and Selection of the Target System

Test the current RDBMS on specific features of the system in operation and compare throughput between system with RDBMS and system applied NoSQL Currently the most popular NoSQL solutions are HBase, Cassandra, Mongolia DB (MongoDB), etc. HBase involves complex setup and difficult maintenance management, while Cassandra has severely degraded performance when performing GC (Garbage Collection). Recently, Mongolian DB is the most used NoSQL solution because it is easy to install and expand, and is most beneficial when you configure it on a single-node information system. In this paper, the test environment was conducted by using the DM Mongolia because it is not a distributed test system. We used a SUN information system in this study, as it is the internal blog system of many companies, such as IBM. It was based on an open-source project of the Apache Foundation, Roller.

3.2 Data Structure of the Target System

Fig. 1 is the entity relationship diagram outline of the main entities of target systems. The 'weblogentry' entity generates the large number of transactions in this system. Thus, this research compares the performance between two systems that operates in one RDBMS and that in another system utilizes NoSQL for only 'weblogentry' entity operation.

3.3 Environment and Configuration of the Test System

Fig. 2 shows the physical configuration of the system that we used in our test. In order to distribute the load between the several servers, server separated Web application server (WAS) server, and database server and use special client to generate the data loading.

The WAS server was constructed by using CPU Inter Core i5 M560, 4 GB memory, and Windows Server 2003. WAS used the Tomcat 6.0. DB server, including CPU AMD Athlon 64 X2 Dual Core 5200+, 2 GB memory, and Windows Server 2003.

For the RDBMS in our test, we used Oracle 10g XE, and for NoSQL we used MongoDB 2.0. The terminal for generating a load was built with CPU Intel Core 2 DUO CPU E7500, 2 GB memory, and built-in Windows XP. The Apache JMeter was used for generating loads and the network we used was 100 Mbps LAN.

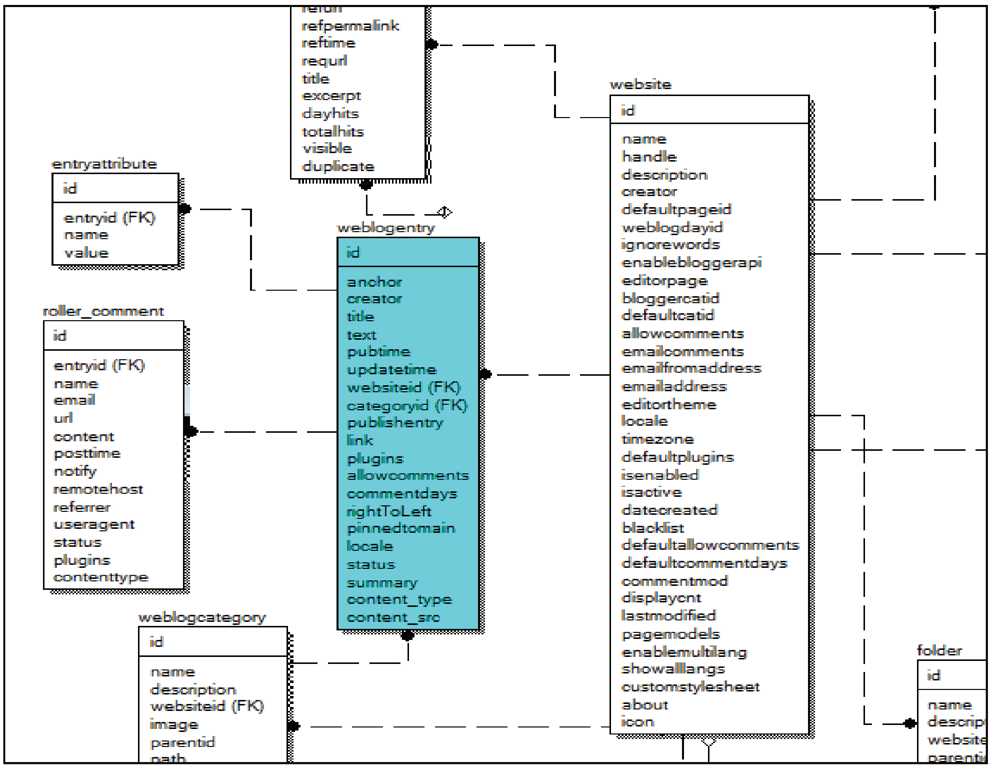


Fig. 1. The entity relationship diagram of the main entities of target systems.

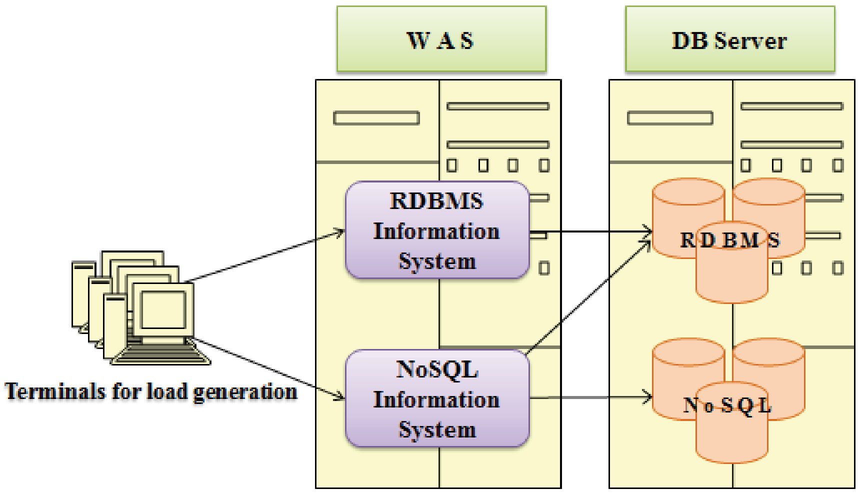


Fig. 2. The system environment used for testing. WAS=Web application server, RDBMS= relational database management system.

The WAS server takes advantage of system WAS applying RDBMS and NoSQL, while the DB server drove the RDBMS server and NoSQL used in our test. The RDBMS systems and the

NoSQL system used the RDBMS. The same database in the DB server had to use a separate schema.

3.4 Test and Results Analysis

In order to performance measure, the load test was performed on the querying function and store function of notice, which is the most frequently used feature in implemented blogging systems using RDBMS and NoSQL. The store function used JMeter in an implemented system utilizing RDBMS and NoSQL and was caused to generate a total of 1,000 posts forming 25 threads 40 times.

Figs. 3 and 4 indicate the post registration performance measurement results of the NoSQL system and RDBMS system. The legend is as follows: the dot of background in the figure represents the processing time for requests and a triangle represents the change of throughput. A circle denotes a change in the average; a square stands for a change of median; and a diamond represents deviation.

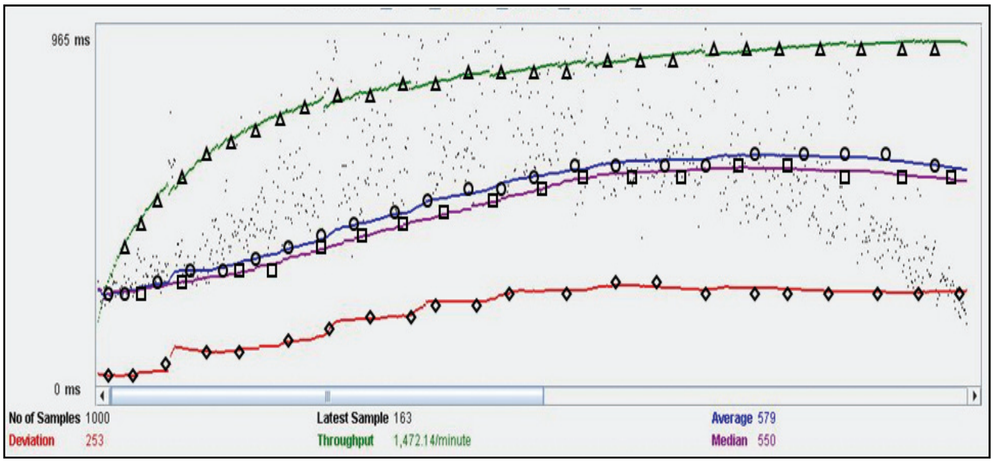


Fig. 3. Post registration experiment result of the NoSQL system.

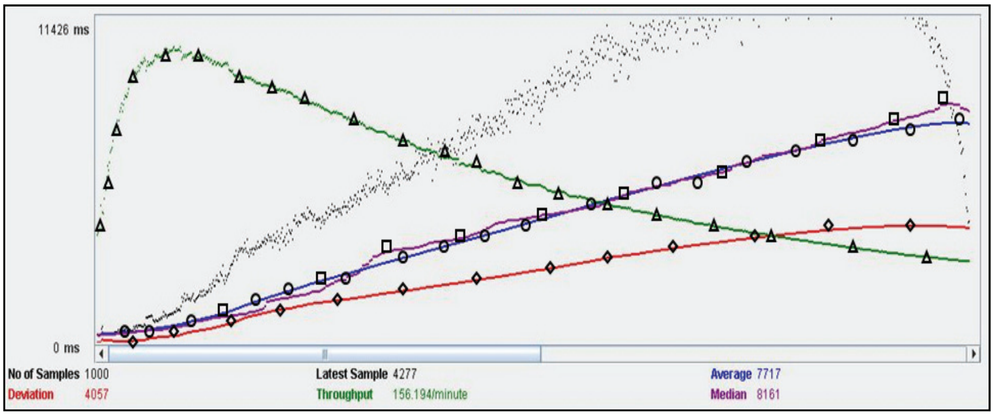


Fig. 4. Post registration experiment result of the relational database management system (RDBMS).

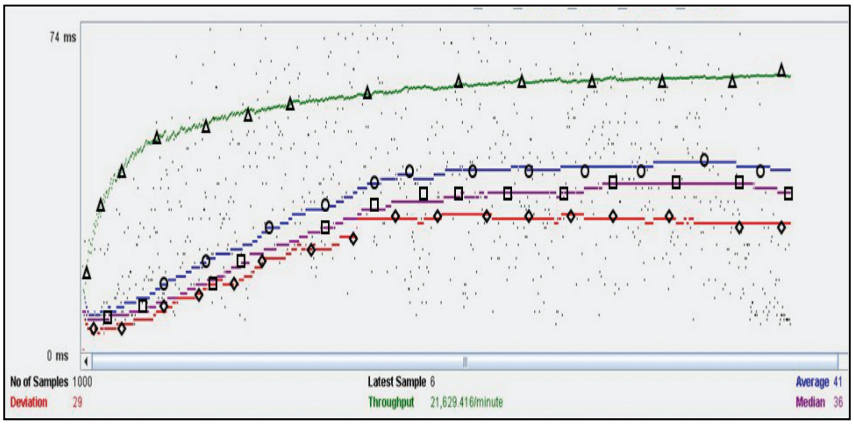
According to the test results, the throughput of RDBMS system increased to the extent that the system can accommodate then was sharply reduced.

The reason of this result is that when there is a large amount of simultaneous data processing, interference between data and the waiting that takes place during the process of obtaining resources between concurrency controls leads to performance degradation. As a result, the NoSQL system improves the performance by 850% as compared to RDBMS system.

For list of query functions, the test was performed by dividing two cases into simple query lists and storage loads in query lists. The JMeter was used to test simple query lists in order to take advantage of the system that used RDBMS and NoSQL. In regards to the implementation of the system, 25 threads to query lists were performed 40 times.

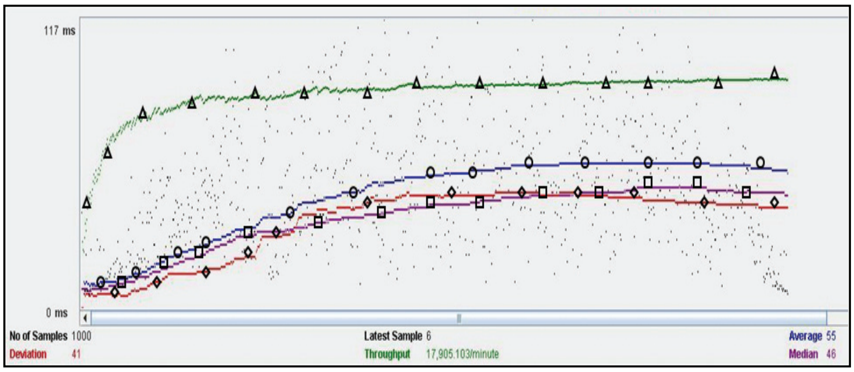
Figs. 5 and 6 indicate the post simple query performance measurement results of the NoSQL system and RDBMS system.

According to the results, the throughput of the system using NoSQL was 20% higher than the one using RDBMS. The RDBMS system had a lower performance as compared to NoSQL system. However, both systems operated stably because the RDBMS system had a standard deviation for response time that was less than 100%.



No of Samples 1000
Deviation 29
Latest Sample 6
Throughput 21,629.416/minute
Average 41
Median 36

Fig. 5. Post query load test result of the NoSQL system.



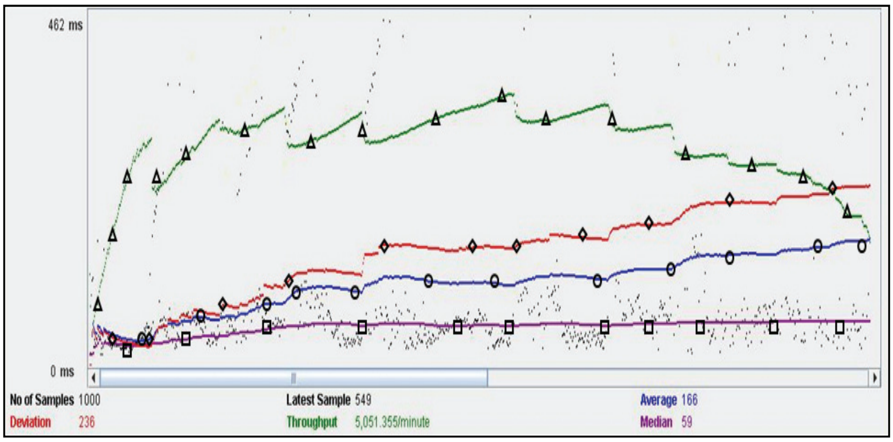
No of Samples 1000
Deviation 41
Latest Sample 6
Throughput 17,905.103/minute
Average 55
Median 46

Fig. 6. Post query load test result of the relational database management system (RDBMS).

Finally, the inquiry time performance of the blog system using RDBMS and the blog system using NoSQL was tested. For each test system, 25 threads were performed 40 times for post-registration, and 25 threads were performed 40 times for the inquiry list.

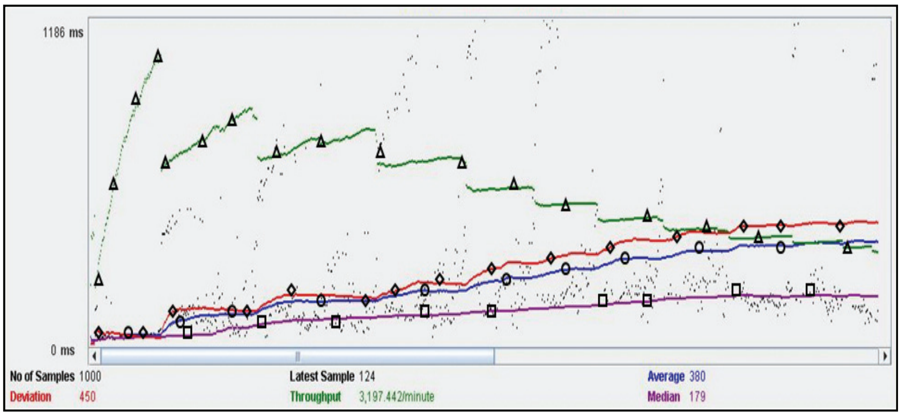
Figs. 7 and 8 indicate the post simple query performance measurement result of the NoSQL system and RDBMS system.

As a result, the throughput of blog system using NoSQL was lower than the throughput of the blog system using RDBMS by 58%. However, the inquiry time was increased by approximately 38%. The cause of this difference is that the load was added to the written posts. Therefore, as the load increases, the difference will be even greater.



No of Samples 1000 Latest Sample 549 Average 166
Deviation 236 Throughput 5,051.355/minute Median 59

Fig. 7. Query load test result for registering the posts of the NoSQL system.



No of Samples 1000 Latest Sample 124 Average 380
Deviation 450 Throughput 3,197.442/minute Median 179

Fig. 8. Query load test result for registering the posts of relational database management system (RDBMS).

Table 1 is a summary of the performance test results and Fig. 9 is a graph of the performance test results.

Table 1. Summary of the performance test results

Test item		Test results entry			
		Standard deviation	Throughput (TPM)	Average (ms)	Median (ms)
Post registration	NoSQL	253	1472.14	579	550
	RDBMS	4,057	156.194	7,717	8,161
Inquiry post	NoSQL	29	21,629.42	41	36
	RDBMS	41	17,905.10	55	46
Inquiry post during registration	NoSQL	236	5,051.33	166	59
	RDBMS	450	3,197.44	380	179

RDBMS=relational database management system, TPM=transactions per minute.

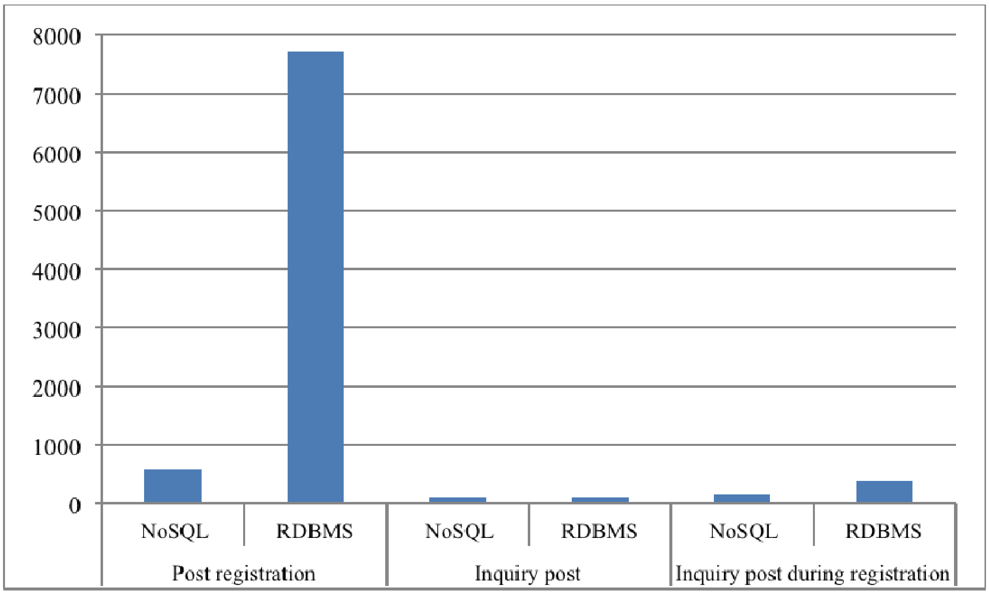


Fig. 9. Graph of performance test results. RDBMS=relational database management system.

As seen from the results shown in the table, when the main entity is switched from the operating system using RDBMS to NoSQL and a load occurs, the system utilizing NoSQL ensures higher throughput than the system using RDBMS by 850% higher in the post storage, 20% higher in the inquiry list, and approximately 58% higher in the inquiry list post.

For the standard deviation of measurable throughput for system load stability, compared to the system utilizing RDBMS, the system using the standard deviation NoSQL was 94% lower in posts storage, 30% lower in the inquiry list, and was about 55% lower for the post inquiry list.

Therefore, NoSQL was verified to be more stable on load than the RDBMS system.

4. Conclusion

Recently, with the rapid increase of data, performance degradation of existing RDBMS has become a problem. In order to ensure the consistency of data in RDBMS, a new solution for the horizontal partitioning of data and limited performance is necessary. This has lead to the advent of NoSQL. In this study, when a load occurs in an operating system that runs on RDBMS, some functions switch to NoSQL and the system utilizing NoSQL and stores 850% higher in post storage, 20% higher in inquiry list, and provides higher performance by about 58% higher in inquiry list posts, as compared to the system only using RDBMS. Therefore, for the systems requiring performance rather than consistency, applying NoSQL could bring about significant performance improvements.

To store data in NoSQL, data is partitioned and shared on another server by using Sharding techniques to save the data processing time. With the Sharding technique, a split data unit is called a shard. The data is divided horizontally, and each shard is dispersed and stored in multiple servers. Therefore, numerous amounts of data can be stored and processed. In this study, a part of distributed processing was excluded and the test was performed. However, if the experiment was conducted in a distributed processing environment with more data and a higher load context, the performance difference between RDBMS and NoSQL would be greater than the result of this study.

In a system utilizing existing RDBMS, SQL is used to access to database. The SQL has a high portability between RDBMS because of American National Standards Institute (ANSI) standard management. However, the portability in the case of NoSQL is quite low because most of the APIs used are vendor-dependent. To ensure portability between vendors, it is urgent to raise the standard for NoSQL.

REFERENCES

[1] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377-387, Jun. 1970.

[2] E. Hewitt, *Cassandra: The Definitive Guide*. Beijing: O'Reilly, 2011.

[3] J. Gantz and D. Reinsel (2011, June). "Extracting value from chaos," *IDC iView* [Online]. Available: http://www.emc.com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf.

[4] C. J. Date, *An Introduction to Database Systems*, 8th ed. Boston, MA: Pearson/Addison Wesley, 2004, pp. 466-499.

[5] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: a distributed storage system for structured data," in *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation*, Seattle, WA, November 6-8, 2006, p. 15.

[6] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: Amazon's highly available key-value store," in *Proceedings of 21st ACM SIGOPS Symposium on Operating Systems Principles*, Stevenson, WA, October 14-17, 2007, pp. 205-220.

[7] A. Lakshman and P. Malik, "Cassandra: a decentralized structured storage system," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 2, pp. 35-40, Apr. 2010.

[8] S. Gilbert and N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services," *SIGACT News*, vol. 33, no. 2, pp. 51-59, Jun. 2002.

Yong-Lak Choi

He received the Ph.D. degree in Computer Science from Graduate School of Soongsil University in 2002. He is currently a professor of Graduate School of Software of Soongsil University. His research and main interest areas are databases, data modeling, big data, software engineering, information strategy planning, and open source architecture.

Woo-Seung Jeon

He received the M.S. degree in Information Science from Graduate School of Soongsil University in 2011. During 2000-2014, he stayed in POSCO ICT Inc. to software develop and customize the enterprise resource planning solutions. His research and main interest areas are databases, data modeling, and open source architecture.

Seok-Hwan Yoon

He received the Ph.D. Degree in Department of Industrial Engineering from Ajou University, South Korea in 1996. During 1986-1997, he stayed in Electronics and Telecommunications Research Institute (ETRI) as a representative member of technical staff, and during 1998-2004, he was the team leader of Institute for Information Technology Advancement (IITA), and during 2005-2012, he was the CEO in IT company. And now he is a professor in Department of Computer Sciences at Semyung University. His research interests are in the areas of cloud computing, algorithm, business process reengineering/information strategy planning, development schema.