International Journal of Grid Distribution Computing
Vol.7, no.3 (2014), pp.13-22
http://dx.doi.org/10.14257/ijgdc.2014.7.3.02

Research on Multi-tenant Replication Consistency Based on Quorum NRW System

Kong Lanju, Li Qingzhong, Li Lin and Sang Chengliang

School of Computer Science and Technology, Shandong University, Jinan 250101, China
klj@sdu.edu.cn, corresponding author *lqz@sdu.edu.cn, ducklilin@126.com,scll@163.com*

Abstract

Replication technology, such as Quorum NRW system, is the most effective technology to improve availability of SaaS. However, the variety of SaaS transaction will result in application unbalance. In this paper, we construct a new Quorum NRW system for multi-tenant database which is based on the method of sharing database and schema model. It can provide different R and W values for each tenant according to application transaction features. The paper summarizes the left change and the right change modes according to the differences changes, and proposes Multi-tenant Quorum NRW system which transitive left change method and transitive right change method online. Multi-tenant Quorum NRW system can ensure strong consistency in condition of R + W> N. We design an experimental evaluation of multi-tenant Quorum NRW system, experimental data shows that Quorum NRW system enhances the transaction throughput by 5%-25%.

Keywords: *Multi-tenant Database, Replication, Quorum NRW*

1. Introduction

With the developing of SaaS and cloud computing, multi-tenant data management technology has also made significant progress. To ensure the high availability of the multi-tenant database, distributed technology and database replication technology are inevitable.

Among most replication technologies of improving availability and performance, Quorum [1] achieves consistency and availability tradeoffs by configuring the reasonable value of NRW. In the case of R + W> N, it can provide strong consistency. The value of RW is generally different configured according to requirement, such as read performance, write performance or the need of balance between them. However, in the multi-tenant database, multiple tenants share the same database, and each tenant has different needs. It's not a good idea to configure RW setting statically, and the RW configuration needs to vary according to each tenant.

This paper proposes a tenant-oriented dynamic Quorum system. The multi-tenant Quorum system explores difference between each tenant, and then assigns to each tenant different RW in multi-tenant application. It provides two adjust methods, named transitive left change and transitive right change, which can change RW value online. Multi-tenant Quorum NRW system could not only guarantee the strong consistency of system, but also could improve the availability and performance.

This paper is organized as follows. In Section 2 describes the relevant research foundation. In Section 3 proposes multi-tenant Quorum NRW system. In Section 4 proposes NRW computation model of multi-tenant Quorum NRW System. In Section 5 proposes

ISSN: 2005-4262 IJGDC

maintenance of multi-tenant Quorum NRW System, followed by the experimental evaluations in Section 6. The conclusion is given in Section 7.

2. Related Work

In traditional data management, data replication technology provides a guarantee for the high availability and fault tolerance to database system. For ACID transactional applications, multiple replications of which have been maintained strong consistency is desired selection.

Among all the algorithms for keeping consistency of replications, the Quorum algorithm has multiple ways to be realized. It includes level Quorum [1], plane Quorum [2] and tree-based Quorum [3]. Avaishai Wool [4] analyzed several Quorum achieved and pointed out that with the continuous development of hardware and applications, quorum did not achieve the expected demand replication performance and fault tolerance, to achieve load balancing and other aspects, we need further improvements. R. Jiménez-Peris (2003) [5] did a detailed analysis to several of them in scalability, availability, communication cost and performance, which showed that in most cases, the use of read one write all is a relatively better choice.

In 2000, Brewer [6] proposed the famous theory of CAP. The theory was proved correct by Gilbert and Lynch [7]. In a distributed system, there exists the inherent trade-offs among data consistency, system availability and partition according to the theory. Yet the system can only provide two of these three features. The existing replication technique can be divided into synchronous replication and asynchronous replication. The synchronous replication technique can keep consistency and partition fault-tolerance, while the asynchronous replication technique can provide high availability and partition fault-tolerance.

Emmanuel Cecchet (2008) [8] discussed the differences between the theory and practice of database replication technique. Besides that, they explained profound theories in simple language from various points of view, such as relational database, SQL, copy middleware, and system management. Bettina Kemme (2010) [9] combined distributed system technique and database technique to present a technology which is similar to the asynchronous replication but can keep strong system consistency in PostgreR database.

Meanwhile, for cloud computing, in recent years, many new databases and data models have been proposed to support its new features. Dynamo [10] is a key-value storage system with high availability features, and it designed replication technology, using sloppy quorum and distributed a copy of the synchronization protocol. However, the availability was dumping after weighing the availability and the consistency. Megastore [11] is a cloud database realized by Google. It uses improved paxos algorithm to realize strong consistency. And Megastore designed Entity to ensure the strong consistency. In another cloud database, Sales force [12] uses a wide table which has 500 columns to store data, which takes a model of sharing data to support multi-tenants.

The paper builds up multi-tenant Quorum NRW system to ensure strong consistency and enhance the system's performance, scalability. In the full replication multi-tenant Quorm NRW system, in order to ensure the strong consistency and improve the database's performance, it can dynamically change the value of W and R when the requirements of tenants' transactional properties changed.

3. Multi-tenant Quorum NRW System

Quorum system uses replication redundancy to implement a fault-tolerant and high availability. The value of N means replications of the system, R is the minimum number of nodes in Quorum replication system which is involved in the transaction read, W is minimum

number of nodes which are involved in the synchronous write operations. It can provide strong consistency guarantees by setting a reasonable value of RW and the value of N.

Quorum NRW multi-tenant system uses the ideas of Quorum systems in a multi-tenant database; it extracts system characteristics and load characteristics of different services.

Assumed that N = {N <1>, N <2> ..... N <n>} is a completely replication system, and then N <i> = {DT <1>, DT <2> ..... DT <m>}($1 \leq i \leq n$), in which DT <i> means tenants data of T <i>. From the perspective of multi-tenant database, the transactions of each tenant show a certain rule. We need to set reasonable values of R and W separately for DT <i> to ensure consistency.

We could get a complete copy of the system if placing the same replica in every node. Such as N = {{DT <1>, DT <2> ..... DT <m>}, {DT <1>, DT <2> ..... DT <m>} ..... {DT <1>, DT <2> ..... DT <m>}}. And then, the logical unit is set based on the count of DT <j> (($1 \leq j \leq m$) in N <i> ($1 \leq i \leq n$). Quorum NRW multi-tenant system is shown in Figure 2.

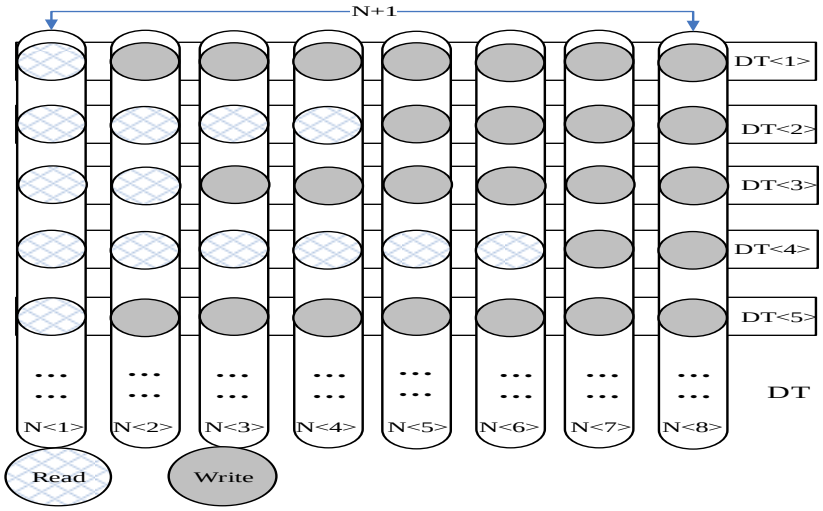

Figure 2. Multi-tenant Quorum System

We can see from the diagram, N is set to 7 and R+W=N+1. Each portrait can be seen as a single data node N, which is composed of tenants' data DT. The tenant's DT <i> is copied on many data nodes, and gray represents the write operation, a read operation is indicated by squares. For different tenants, it dynamically allocates R and W to improve the value of overall system performance. For the case, the tenant1 uses read 1 write 7, and tenants2 uses read 4 write 4, while other tenants have their own reading and writing strategy. In the view of SaaS applications, multi-tenant Quorum NRW will extend to the data for each tenant level, further speaking, just pay as you go.

4. NRW Computation Model of Multi-tenant Quorum NRW System

From the above definition of multi-tenant Quorum NRW system, the system is a read-write Quorum system, which is a full replica of the system. The probability of N <i> ($1 \leq i \leq n$) participating in the read operation for each node is R / N, the probability of participating in the write operation is (W / N).

We assume that the capabilities of the system is A, and g is the system workload percent generating from write operation. Then gA is used for processing the writing process, while (1-

g) A is used for processing the read operation. The processing capacity of a single node at a time is B. In consideration of the whole system, there is a relationship between B with A:

$$B= (W/N)gA+(R/N)(1-g)A \quad (1)$$

Based on the relationship described in Equation 1, the expression of Sca which is the scalable of systems can be derived as shown in Equation 2

$$Sca=A/B=N/(gW+(1-g) R) \quad (2)$$

As can be seen from the formula (2), the system scalability is directly related to the value of N and WR. Rational allocation of valid WR values can improve system scalability. It can be seen from the formula, if g is too large, the smaller configuration of W can be reasonable. While corresponding to the smaller (1-g) and larger R, this can increase the scalability.

In multi-tenant database based on sharing model, it can be expanded to extend performance too. The value of unified W may be suitable for partial tenants, and other tenants acquired inappropriate values of W and R. Then we let f <i> be the proportion of processing and storage capacity for each tenant according to the occupied workload in the system. There is a corresponding g <i>, W <i> and R <i> for each tenant T <i>, as shown in Equation 3.

$$MSca = \sum_{i=1}^{n} (f < i > N /(g < i > W < i > +(1 - g < i >)R < i >)) \quad (3)$$

As Equation 3 shows, we can allocate W and R values for different tenants instead of using unified W and R values to access higher system scalability. The different W <i> can give different values of g <i>. And then the multi-tenant Quorum NRW system can get the value to improve the system scalability.

5. RWN Modification

According to different tenants' transaction and workload, the multi-tenant Quorum NRW system can dynamically distribute the value of R and W for different tenants to improve the system's performance. However, we can't directly change the value of RW. When R and W changed, one of them will be smaller than before while executing the transaction. And the system can't provide strong consistency.

Given that the system's final configuration is R+W=N+1, we assume that the RW was on a scale changing line. R was placed on the left and W was placed on the right. We summarized two changes model. The left change model is that when R decreases, W becomes larger and the RW scale moves left in the line. The right change model is contrary to the left change model. To show the behavior of RW change more clearly, we use R<b>,W<b> to name the original RW and call the changed RW R<a>,W<a>.After that, we can use the formula R<b>+W<b>=N+1=R<a>+W<a>.

In the left change model, we can find that R decreases, W becomes larger, in other words, R<a> is less than R<b> and W<a> is greater than W<b>. If we analyze the changing process, we could get a conclusion that R's reducing is the main reason for the system's inconsistency and W does not work. For Quorum NRW system, there exists W<a>+R<b> is greater than N+1, while R<a>+W<b> is less than N+1. Similarly, in the right change model, it exists a contrary condition that W<a>+R<b> is less than N+1, while R<a>+W<b> is greater than N+1.

Based on this, we proposed two transition methods to accomplish Quorum NRW system's RW dynamic changes. In this method, system's RW has three states, including the original state (R<b>,W<b>),the changed state (R<a>,W<a>) and the transitional state (R<m>,W<m>).This method is aiming at every tenant instead of the whole system.

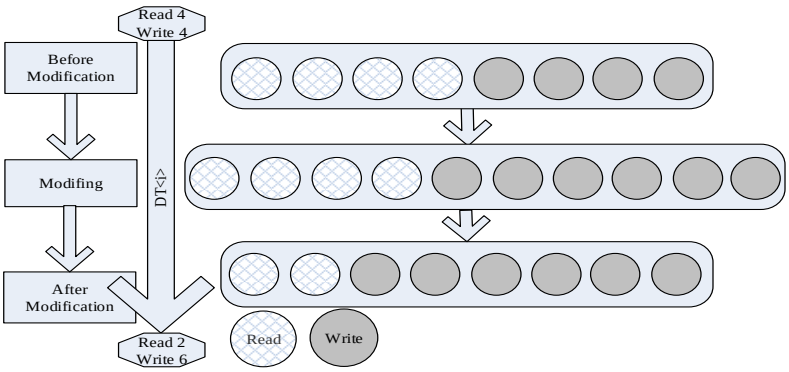

Figure 3. Transition Method's Left Change Mode

In the left change model, the system remains original state before receives change order. The RW value is (R<b>,W<b>).After the system receiving change order, the RW becomes (R<a> , W<a>).When the system turns into changed model, the RW becomes (R<m>,W<m>). And R<m>=R<b>,W<m>=W<a>. At this time, R<m>+W<m>=R<b>+W<a>, which is greater than N+1.The system will stay in transitional state for some time. During this period, the system ensures all read-write operation to be accomplished before it comes into the transitional state. The time setting depends on the tenant's concrete status, we will not discuss it in detail.

After the transitional time, the system turns into changed stare. The RW becomes (R<a>,W<a>).When system running from transactional state to changed state, the W does not change, the R decreases, but R<a>+W<a>=N+1.Because all the read-write operation has been accomplished before the transitional state, the system can keep strong consistency in the process of change. The left change model is shown in Figure 1.

The right change model is symmetrically similar to the left change model. In the transitional state, R<m>=R<a>,W<m>=W<b>.When the system turns into changed state, W<m> will reduce to W<a>.The right change model is shown in Figure 2.

In transitional state, no matter in the left change model or the right change model, there exists status of (R<m>+W<m>) >N+1.This will bring to extra operation. For every tenant, the transitional time last for a short time and it impacts little on the whole system. However, the transitional method can ensure strong consistency in the transitional process and the whole system can be accomplished online.

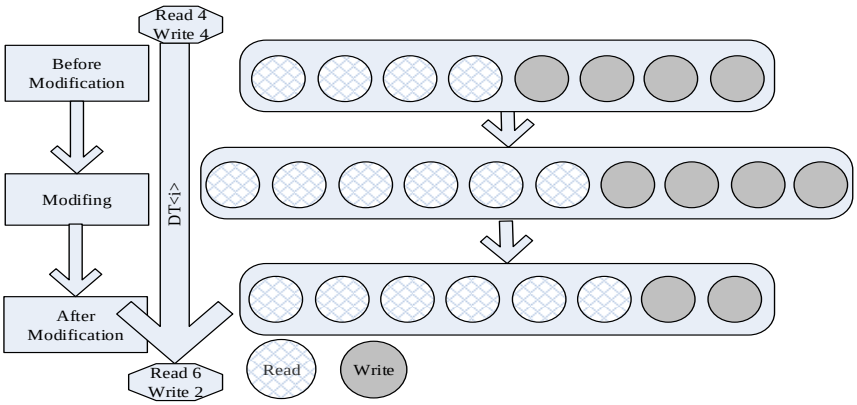

Figure 4. Transition Method's Right Change Mode

6. Experiment

The database servers are all IBM X3650M3, which is configured with Intel (R) Xeon (R) CPU X5620 * 16 of 2.4GHZ, 8G main memory, and 500G * 2 of hard disk. The operating system is Red Hat Enterprise Linux Server release 5.5 X 32. MySQL has been installed on the database node.

Application servers use the same hardware configuration database server node, and the operating system is Microsoft Windows Server 2003 R2 Enterprise Edition Service Pack 2. Tomcat 6.0.18 has been installed. Tenants' data are stored into the base table plus extension table. Client nodes are ordinary PC, Dell Optiplex 380, in which cpu is Intel (R) Core (TM) 2 Duo CPU E7500@2.93GHz, main memory 4G, hard disk space 320G, the operating system is Microsoft Windows XP Professional Edition Service Pack3.

To ensure enough characteristics of multi-tenant transaction, more than a dozen applications with different characteristics are involved, such as agency management services, payment management services, expert management services, examination management services, certification management services and training management services and so on. Each application is rented by assorted tenants (*e.g.,* examination management services may be leased by tenants in different sectors such as health and transport *etc.*,).

Depending on the application size and characteristics of the experimental data, we configure the system N is 7, and using the multi-thread technique to simulate application transaction requests.

At first number of tenants is fixed at 15, and we changes the number of users for each tenant to simulate changes of requests. Each tenant sends many kinds of requests. Set a different R and W value for each tenant, then the whole system begins to run. After several rounds of repeated tests, we get the average results of the individual. Throughput results TPS = (Com_Commit + Com_Rollback) / Seconds is shown in Figure 5.

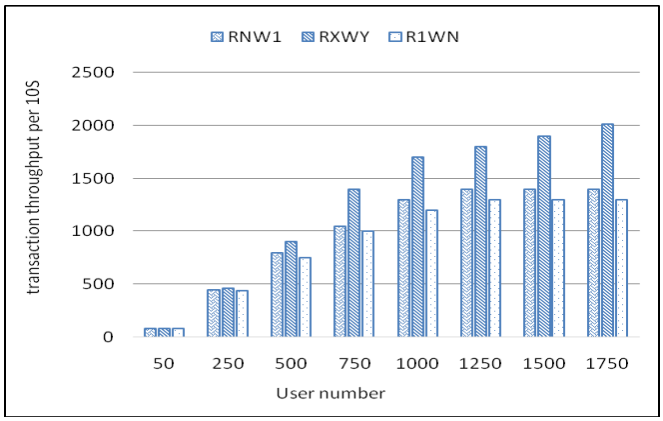

Figure 5. Transaction Throughputs Different Request

In the view of overall point, the use of reading 1 write N (R1WN) has an outstanding performance than read N 1 write 1 (RNW1), but the multi-tenant Quorum NRW performance even better than R1WN. Multi-tenant Quorum NRW can follow the individual tenant needs to configure the value of R and W, therefore, it must be able to bring a better overall performance results.

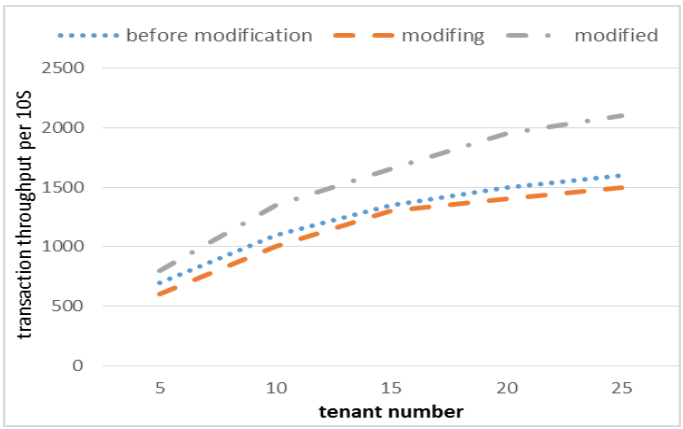

Figure 6. Transaction Throughput During Modification

In order to test the influence of modification, we design another experiment. We changed the number of tenants and number of users of each tenant to imitate the difference of SaaS application. The results of this experiment are shown in Figure 6.

As can be seen from the Figure 6, the transaction throughput will be a little decrease during the modification. But after modification, it will be more lager than before. And with the increase of tenant numbers, the advantage of multi-tenant Quorum NRW is more obvious. Experimental data shows that Quorum NRW systems enhance the transaction throughput by 5%-25%.

7. Conclusion

This article is faced to multi-tenant database. We define multi-tenant Quorum NRW system. And then we do scientific and rational analysis for the scalability, performance of the system. With the consideration of each tenant's applying transaction peculiarity, requirements, and changing distinctions, two changing ways are provided, left change and right change. Correspondingly, two on-line transitional changing ways are offered to provide reasonable R, W values for each tenant. It ensures the high consistency and bringing better performance to the tenants and the whole system. Finally, we design experiments to verify the multi-tenant Quorum NRW system. Experimental Result proves that it improves the availability and performance of the multi-tenant database.

Yet, there are still many problems which need to be further studied. We will do a further research in the relationship between individual features and the tenant's application transaction between RW settings. Of course, with the commencement of these studies, there may be new problems and research points. We will continue to focus on these issues.

Acknowledgements

This work is supported by National Key Technologies R&D Program No.2012BAH54F01; National Natural Science Foundation of China under Grant No.No.61272241,No.61303085; Science and Technology Development Plan Project of Shandong Province No. 2012GGX10134; Independent Innovation Foundation of Shandong University under Grant No.2012TS075, No.2012TS074; Shandong Province Independent Innovation Major Special Project No.2013CXC30201.

References

[1] A. Kumar, "Hierarchical quorum consensus: A new algorithm for managing replicated data", IEEE Trans. Comput, vol. 40, no. 9, **(1991)**.
[2] R. A. Bazzi, "Planar quorums", In Proc. 10'th Inter. Workshop on Dist. Algorithms (WDAG), **(1996)** October, pp. 251-268, Bologna, Italy.
[3] D. Agrewal and A. E. Abbadi, "The generalized tree quorum protocol: An efficient approach for managing replicated data", ACM Trans. Datab. Syst., vol. 17, no. 4, **(1992)**.
[4] A. Wool, "Quorum Systems in Replicated Databases: Science or Fiction", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, vol. 21, **(1998)**.
[5] R. Jiménez-Peris, M. Patiño-Martínez, G. Alonso and B. Kemme, "Are quorums an alternative for data replication? ACM Trans. Database Syst., vol. 28, no. 3, **(2003)**.
[6] E. A. Brewer, "Towards robust distributed systems", In Proceedings of the 9th Annual ACM Symposium on Principles of Distributed Computing, **(2000)** July 7, Portland, USA.
[7] S. Gifford and N. Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services", ACM SIGACT News, vol. 33, no. 2, **(2002)**.
[8] E. Cecchet, A. Ailamaki and G. Candea, "Middleware based database replication: The Gaps Between Theory and Practice", SIGMOD, **(2008)** June, pp. 739-752, Vancouver, Canada.
[9] B. Kemme and G. Alonso, "Database Replication: A Tale of Research across Communities", Proceedings of the VLDB Endowment, vol. 3, no. 1, **(2010)**.
[10] 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 SOSP, **(2007)** October, pp. 205-220, Stevenson, WA, USA.
[11] J. Baker, C. Bond, J. C. Corbett, J. J. Furman, A. Khorlin, J. Larson, J.-M. Leon, Y. Li, A. Lloyd and V. Yushprakh, "Megastore Providing Scalable, Highly Available: Storage for Interactive Services", CIDR, **(2011)**, pp. 223-234.
[12] C. D. Weissman and S. Bobrowski, "The design of the force.com multi tenant internet application development platform", In SIGMOD, **(2009)** June, pp. 889-896, Providence, Rhode Island, USA.

Authors

Lanju Kong, she born in 1978, Ph.D. She is a lecturer in Shandong University and Shandong Provincial Key Laboratory of Software Engineering. Her main research interests include computer software and theory, software and data engineering, XML query and access.

Li Qingzhong, he born in 1965, professor, Ph.D. supervisor. His research interests include large-scale network data management and web data integration.

Li lin, she born in 1980, master. Her research interests include database.

Sang Chengliang, he born in 1988, master. His research interests include database.