TWO-WAYS DATABASE SYNCHRONIZATION IN HOMOGENOUS DATABASE MANAGEMENT SYSTEM WITH BINARY LOG APPROACH

[1]GEDE HADI SURYA, [2]I MADE SUKARSA, [3]I GUSTI MADE ARYA SASMITA
[1]Student, Departement of Information Technology, Udayana University, Bali, Indonesia
[2,3]Lecturer, Departement of Information Technology, Udayana University, Bali , Indonesia
E-mail: [1]hadisurya.1991@yahoo.com, [2]e_arsa@yahoo.com, [3]aryasasmita@yahoo.com

ABSTRACT

Nowadays, there are several researches conducted about design from Database for the real-time application. A real-time Database system is usually defined as a Database system in which the transaction is correlated with the deadline, right when their transactions are done. Along with the developing of the Database system these days, there are many theories emerged which combines the real-time system with the distributed Database and it is called Distributed Real-time Database System (DRTDBS). One of the foremost utilization of DRTDBS is using Replication. With Replication, all of the data will be sent from the original place to the destination place in real-time. On the other hand, Replication has many shortcomings, either from the consistency of the network or when the network is not working properly. This paper will discuss about how a DRTDBS application is built with the Client-Server concept by utilizing Replication and Binary Log from MySQL. This approach will contribute a new alternative and new concept on the process of synchronization Database in real-time with Binary Log file processing. It is also expected to give contribution in developing DRTDBS in the future.

Keywords: *Database Synchronization, Real-Time Database Systems, Replication, Binary Log*

1. INTRODUCTION

Real-time Database system is usually correlated with the transaction which the deadline is right when the transaction is done and all accessed item of the data must be valid until the commit time. If it is not, it will impact the result seriously [1]. The development of Database system which leads to Distributed System makes it possible to apply Replication for making Database which is located at various places, thus it can exchange information and synchronize each other in real-time. It clearly contributes to the efficiency in processing the data if we take data with very wide scope and various different geographic circumstances. Considering with that condition, the coherency and equilibrium of data is very important, it is necessary to maintain the consistency of data which belongs to those various places. Through synchronization, we can achieve that. Data Synchronization is a process to maintain the consistency of the data which exists at a Server and another data which exists at another Server either periodically or in real-time [2]. In the process of synchronization, there are copying and distributing processes, also Database objects from one Database to another Database, until it is formed a system, called Distributed Database System (DDBS) [3]. Since the organization tends to be influenced by geographic, a DDBS is suitable for the organization structure and is better than Centralized Database. Each location will have local data and ability to acquire necessary data from the other locations through communication network [4].

The different approach is attempted to innovate a new alternative way in synchronizing real-time data on the distributed Database by utilizing Replication of MySQL. The Replication process on MySQL Database will produce Binary Log file which is used by Server Master to save every commands which will be delivered to Slave. Binary Log is query note that is written whilst the Replication is activated on certain Database [5]. This approach of Binary Log does not completely use the concept of Replication just as previously explained. The application merely utilizes Binary Log file that is made by MySQL in order to read and take the newest command that occurs at a Database. That command will be delivered through network in the message form which is encrypted via application that is developed in programming socket. Socket enables the users to do exchanging data among process/program through the network based on TCP/IP taken from every branch and sent to the destination branch according to the configuration previously made [6].

Synchronization is not without challenges. Many problems occur when the synchronization process is ongoing, one of them is concurrency. Concurrency occurs when many users access and do processing activities (update and delete) at the same data and at the same time [7]. In the term of real-time system, this case would often happen and can cause problems in consistency data. To overcome the problems in concurrency data, some approaches can be done, such as Blocking (Wait transaction) and Rollback [8]. Blocking will have a transaction wait, and then the transaction will be finished until the mistakes that make the data unsynchronized can be solved. It is different from Rollback, it is conflict transaction by make a comparation between the data conflict and cancelling contradict transaction which has smaller timestamp that recoreded before.

The security of the message also becomes an important issue. When there is a data with high confidentiality level, so that, there must be proper security method to secure that data. Advanced Encryption Standard (AES) can be used as the solutions for the anxiety of that data security. A comparative study among AES and other encryption methods results that AES is better than the other encryption methods [9].

The scope of this paper is to discover a new application that is able to do Database synchronization, either in one way or two ways, by using Binary Log approach at homogeneous DBMS. It is considered to be effective, efficient, convenient to use and secured from outside attacks.

In the next section we will see about proposed scheme about this system. That scheme contains general architecture that we used for synchronization, server architecture, and client architecture.

2. PROPOSED SCHEME

This scheme of Database synchronization application have some components involved in a system, they are client component and server component. Those components will be fused with some Database that helps the synchronization process. There will be some Database at the server, it will be used as the hub of client authentication data and the hub of message queue. The client component is connected to Database existing and client Database. The function of client Database is to collect client Database, synchronization configuration and branch list.

This architecture will be a real-time system. Each new data or data changes occur in existing Database that obtained by the application, the data is sent to the destination place based on synchronization configuration that has been made to that transaction. Figure 1 shows the scheme proposed in the form of synchronization application.

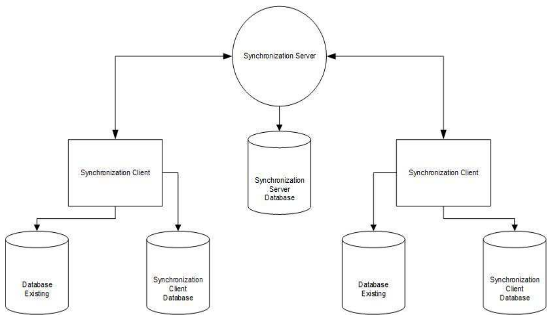

Figure 1: Proposed scheme on synchronization architecture

Server Synchronization Components in the process of synchronizing and integrating data has the function as the media of communication among the branches (Synchronization Client). Synchronization Server component is responsible to receive the whole query message from the original branch and send it to the destination branch. There is simply a process of message routing from the original place to the destination place of the data. However, Synchronization Client component is responsible to reading Binary Log on Database Existing related to the alternation whilst transaction, furthermore, it is to send and receive query data of the alternation data from one branch to the others through Synchronization Server.

2.1 Synchronization Server Architecture

Synchronization Server component has the function as data routing or direct the data from the original to determined destination. The process can be clearly illustrated as figure 2 below.

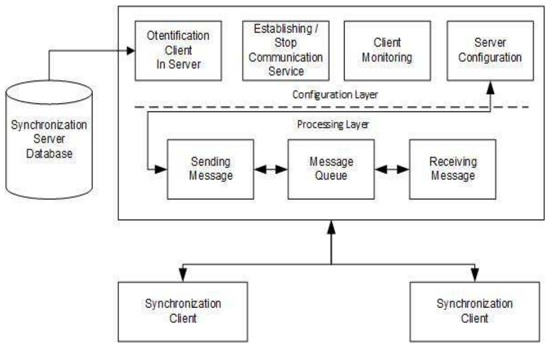

Figure 2: Synchronization Server Architecture

Synchronization Server is started by the authorized user to do synchronization configuration. Next, server synchronization does server management, it is the setting of Server's IP Address and Port which are going to be used as the destination server of the Synchronization Client. Besides addressing, server also does Database management which means connecting to server Database in order to read the user list (Synchronization Client) registered on the server. Layer processing will do sending-and-receiving process from the original to the destination data and do message queue, only if there is a failure in sending data.

2.2 Synchronization Client Architecture

Synchronization Client component is a component functions as Client from Synchronization Server in each branch, in which Synchronization Client will process Binary Log and read the alteration data, and then send that query message through Synchronization Server, finally it is forwarded to the destination place. Synchronization Client will do the process below as figure 3.

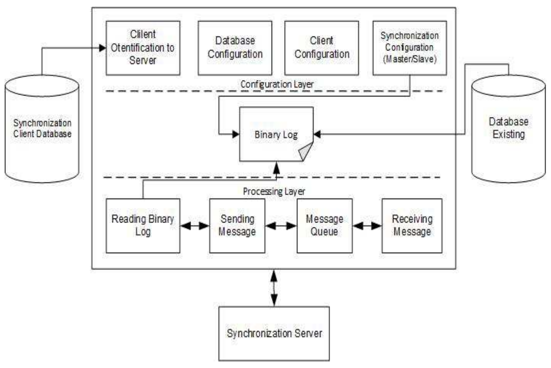

Figure 3: Synchronization Client Architecture

Synchronization Client will do Database configuration, in which the probable selected Database is Synchronization Client Database for the necessity to log in to the server synchronization. After logging in to Synchronization Server, there will be identity setting of the other branch where the Synchronization Client is positioned to join in the synchronization surroundings. This is important to do, because we will communicate with the other branch at certain address, therefore that address must be registered. After that, Synchronization Client application will do Replication configuration process (Master/Slave) and that configuration will be saved in Binary Log. On Processing Layer, the application will read Binary Log regarding to the alteration on Database existing, its purpose is to acquire newest data from Database Existing in Query form. That Query will indeed be sent to server in real-time, it is the forwarded to the destination branch which is previously registered.

Synchronization Client has two abilities to do, they are Primary Copy and Update Everywhere [10]. Primary has one primary data which is going to be distributed to the other branch, when there is an alteration on primary data, so that data will be sent to the other branch. Meanwhile, the alteration on the branch will not influence the primary data. It is reasonably called One-Way Replication. In contrast, Update Everywhere enables to do update data everywhere in the synchronization network. It means it may be parallel update, for instance, 2 different Primary Copy from similar data item (it actually cannot dependently occur in Primary Copy). It is reasonably called Two-Ways Replication.

3. READING THE BINARY LOG

The most important aspect of this synchronization process is how we manage Binary Log file and take the benefit to be data source of synchronization. Binary Log has a structure that must be utilized properly for the query accuracy which will be taken by the application [11].

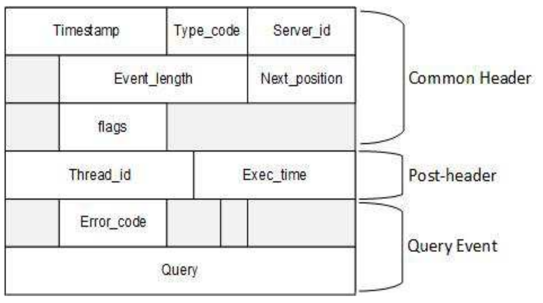

Figure 4: Binary Log Structure

According to the Binary Log structure seen on Fig.4, there is next position at the part of common header. Next position is the information of the last query position recorded on Binary Log. This information is important to take, after that it is used to check the query alteration. Furthermore, we also take Query in the part of Query event. This is clearly an absolute thing to obtain. This Query contains the last Query information at certain position, therefore by getting that last position, we have already obtained the last Query executed at Database.

Generally, we make new algorithm, so that the application can read Binary Log accurately. We combine the algorithm to process index file of Binary Log and the Binary Log. We need to read both of them go get the correct information like find the last file Binary Log in Binary Log index, and then process the Binary Log to get the last query executed at Database, after get the query application have to send that information to destination server. Figure 5 shows the flowchart that used to process the Binary Log.

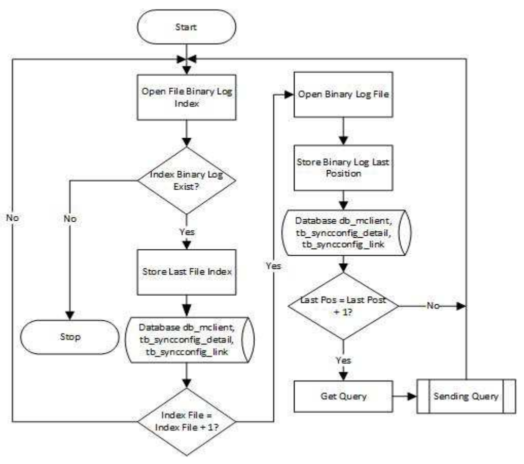
Start
Open File Binary Log Index
Index Binary Log Exist?
No
No
Yes
Store Last File Index
Database db_mclient, tb_syncconfig_detail, tb_syncconfig_link
Stop
Index File = Index File + 1?
Yes
Open Binary Log File
Store Binary Log Last Position
Database db_mclient, tb_syncconfig_detail, tb_syncconfig_link
Last Pos = Last Pos + 1?
No
Yes
Get Query
Sending Query

Figure 5: Flowchart Reading The Binary Log

This flowchart will be applied continuously according to position of alteration occurred at Binary Log. Therefore, any additional query recorded on Binary Log is readable for taking that query, thus query delivery will occur in real-time while that alteration data occurs.

4. SYNCHRONIZATION ISSUES

During synchronization process, we found out many conflicts and problems which must be done appropriately. In this case, the consistency of data must be maintained, thus it will not emerge wider conflict effect towards the synchronization process. Followings are conflicts and problems possibly occur during the synchronization process.

4.1 When Synchronization Client Fails to Connect with Server

This problem occurs when client synchronization fails to connect with server, in order to forward the message to the destination place.

Our solution: To overcome this problem, it is used a tb_outbox table in Synchronization Client Database which has function to store message temporarily while waiting Synchronization Client connected with Synchronization Server again. After checking the connectivity once again to Synchronization Server, and then Synchronization Client is connected back, thus that message will be forwarded to server automatically. In this case, the message will consistently stay as the original and never change at all.

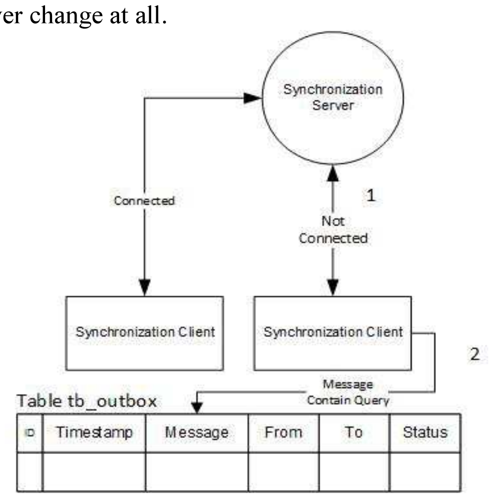

Figure 6: Solution When The Synchronization Client Is Not Connected With Server Synchronization

4.2 When the Destination of Client Synchronization is Offline

This problem occurs when Synchronization Server cannot communicate with Synchronization Client, only if server is ready to forward message which has been sent from the original place.

Our solution: Synchronization Server will check whether destination client is connected or not. If it is not, the message cannot be forwarded to destination place and going to be stored temporarily in tb_msg_queue table of server synchronization. If Synchronization Server is finally connected, data in tb_msg_queue table will be sent to Synchronization Client.

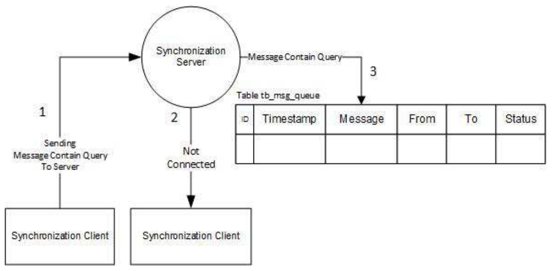

Figure 7: Solution When The Synchronization Server Is Not Connected With Client Synchronization

4.3 Data Reconciliation

During synchronization process, it is obviously possible for some additional or alternation data becomes two Replications. It is dangerous and potentially leads to become conflict [12]. Reconciliation data can be done if there is similar ID with different data during two ways Database synchronization process. The application must be able to manage this problem so that there will be no double ID.

Our solution: The easiest way to overcome this problem is to do recording that data into a new table which contains ID information, contents of the message and anywhere the message comes from. Synchronization Client involved in this conflict will communicate to determine which data will be used and which one will be eliminated. In a nutshell, this process needs aid to take the decision as deciding which data is valid.

This problem actually can be prevented or minimized. It is by ensuring if the early initial state before doing synchronization of each Database is entirely similar. It is early anticipation which can be done if we desire to do Database synchronization. If the initial state Database is not similar from the beginning, then this problem immediately emerges. That is why, import Database facility is also available in this application.

4.4 Double Recording at Binary Log File

This problem occurs when the synchronization is done in two ways. In this case, Binary Log will record query which is executed at the original place, and sent at the destination place. Next, at the destination place, that query will be directly executed by Synchronization Client and get into the table. The alteration of that table will be recorded on Binary Log file and will be taken and sent back to the original place, therefore the exchanged data is twice similar data or in continuous. It is supposed to be no reading back Binary Log with similar data, the process should be stopped at the step of execution and right when it gets into the table.

Our solution: To overcome this problem, we make a tb_jumper table which has function to be a marker in doing a jump when reading Binary Log. Synchronization data in the form of original Binary Log file name, last position and that original data, all of them will be stored in this table. The application will check over tb_jumper to every single input data. If that data is from the other host, then the reading of Binary Log will be heading to the last position of the original host, therefore the application will merely read query which is from the other host.

4.5 Message Security

During delivering or receiving message, we have to be aware to anticipate any outsider activities which can steal message passing through the network.

Our solution: To overcome this problem, the message is encrypted by AES when it is sent, and it is also decrypted by client synchronization that receives the message, thus the message security at the network is guaranteed. Synchronization Server will certainly decrypt message to find out the contents of the message and recognize where to forward this message. After that, the message will be encrypted again and sent to destination client synchronization, the purpose is to be decrypted and executed.

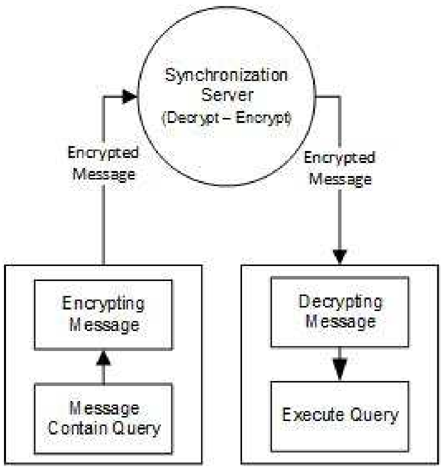

Figure 8: Message Security Solution with AES

Based on the message processing above, it will take time longer about only a fraction of second. However, of that fraction of second, at least we can make anticipation to diminish disadvantages tens hours in the next day, since the data sent is not safe and easy to hijack.

5. NEXT WORKS

Unfortunately, this does not mean that Synchronization works have been fully accomplished. There are many things to develop in order to create a better DRTDBS. Several general problems and solutions have been proposed. The next work is to make this synchronization application useful in wider field, it means that it can be utilized in wider scale and maximum configuration.

On the proposed architecture, we make an approach with two machines which can substitute as client synchronization or server synchronization. In nearly future, we have to make this scheme compatible with 10 or more synchronization machines which are responsible as client or server, moreover those whole machines can be Synchronization Server. This absolutely needs extra work. This is how we manage and process one Binary Log file at each server and determined to be used at many servers at once. This is the biggest challenge of synchronization with the approach of Binary Log, it is then should be developed to be more perfect and create a better synchronization concept.

6. CONCLUSION

Two-ways Database Synchronization In Homogenous Database Management System With Binary Log Approach create an client and server application that can do synchronization according to configuration and make the users conevenient in doing Replication configuration. With the user interface previously made and Binary Log processing, it will complement the Replication weakness at MySQL. The most important aspect is how we do Binary Log processing to be utilized by application in synchronization. Binary Log can be accessed through direct query command and combined with algorithm that is made to do so. We only need to investigate the position to obtain the last query recorded on Binary Log and select Binary Log structure to obtain that query and send it to the destination branch. With this approach we know how to get a benefit with Binary Log file such as new concept of Real Time in Database synchronization.

Besides, there is not only a few problem in synchronization. Such problems include network error, data conflict, double ID processing and data security, those should be managed with appropriate handling, therefore the errors in synchronization can be minimized. This application is able to solve those problems by using approach in the form of new table. This table collects every data which fails to send, therefore the transaction must wait until finally it sends again after the transaction is available [8].

REFERENCES:

[1] Kam-Yiu Lam, Tei-Wei Kuo. "Real-Time Database Systems: Architecture and Techniques". *The International Series in Engineering and Computer Science Volume 593*, 2002, pp. 3-8.

[2] Kao, B. and Garcia-Molina, H. "An Overview of Real-Time Database Systems." *In proceedings of NATO Advanced Study Institute on Real-Time Computing*. St. Maarten, Netherlands Antilles, October 9, 1992, Springer-Verlag, pp. 9.

[3] Mazilu, Marius Christian. "Database Replication". *Database Systems Journal vol. I, no. 2/2010*, pp. 33-38.

[4] Ashish Srivastava, Udai Shankar, Sanjay Kumar Tiwari. "Transaction Management in Homogenous Distributed Real-Time Replicated Database Systems". *International Journal of Advanced Research in Computer Science and Software Engineering,* Volume 2, Issue 6, June 2012, pp. 190-196.

[5] Jeremy D. Zawodny, Derek J. Balling. "Optimization, Backups, Replication, Load Balancing & More, First Edition". *O'Reilly Media, Inc.. America,* 2004, pp. 150.

[6] Ming Xue, Changjun Zhu. "The Socket Programming and Software Design for Communication Based on Client/Server". *Circuits, Communications and Systems, 2009. PACCS '09. Pacific-Asia Conference on*, 16-17 May 2009, pp.775-777.

[7] Arun Kumar Yadav, Dr. Ajay Agarwal. "A Distributed Architecture for Transactions Synchronization in Distributed Database Systems". *International Journal on Computer Science and Engineering (IJCSE)*. Vol. 02, No. 06, 2010, pp. 1984-1991.

[8] K. M. Prakash Lingam. "Analysis of Real-Time Multi version Concurrency Control Algorithms using Serialisability Graphs Blocking". *International Journal of Computer Applications (0975 - 8887)*, Volume 1 – No. 21, 2010, pp. 57-62.

[9] Hamdan.O.Alanazi, B.B.Zaidan, A.A.Zaidan, Hamid A.Jalab, M.Shabbir and Y. Al-Nabhani. "New Comparative Study Between DES, 3DES and AES within Nine Factors". *Journal Of Computing*, Volume 2, Issue 3, March 2010, pp. 152-157.

[10] Matthias Wiesmann, Fernando Pedone, Andre Schiper, Bettina Kemme, Gustavo Alonso. "Database Replication Techniques: a Three Parameter Classification". *Proceedings 19th IEEE Symposium on Realible Distributed Systems (SRD2000)*, Nurnberg, Germany, October 2000, pp. 206-215.

[11] Mats Kindahl & Lars Thalmann. *An API for Reading the MySQL Binary Log.* Oracle.

[12] J. Foster. N., Michael B. Greenwald, C. Kirkegaard, Benjamin C. Pierce, and Alan Schmitt. "Exploiting Schemas in Data Synchronization". *Journal of Computer and System Sciences*, Volume 73, Issue 4, June 2007, pp. 669-689.