Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Volume 21, No. 1, ISSN 1726-9679
ISBN 978-3-901509-73-5, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2010
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals of DAAAM International

DISTRIBUTED DATABASES REPLICATION METHODS FOR MYSQL

BOICEA, A[lexandru]; SERBAN, A[lexandru]; NICULA, A[drian] - I[onut] & RADULESCU, F[lorin]

Abstract: *A distributed database is a collection of several, logically interrelated database located at multiple locations of a computer network. Replication is the process where distributed databases synchronize with each other. In this process one database updates it's own data with respect to another or with reference to certain criteria for updates.*
In this paper we have looked at the the principles of database replication in MySQL, basics of replication topologies, replication agents and the effect of replication on transactional consistency.
Key words: *replication, topologie, agent, clustering*

1. INTRODUCTION

Replication is actually quite straightforward. At its core, it merely involves an administrator picking a server to act as the master, and then registering one or more slave servers to receive updates from the master. Each slave server is responsible for contacting the master server. This master server records all data manipulation statements in a binary log, which is then fed in a stream to any slave(s) that contact the master. The slave computers then play back these statements locally, thus updating their own data copies accordingly.
In addition, a slave can, in turn, act as a master to other servers. This lets you construct sophisticated chains of replication servers.
Obviously, there are many steps to follow to correctly configure and use replication, but the preceding discussion describes it accurately at a high level..

2. BENEFITS OF REPLICATION

Replication is recommended if any of the following are met:
• high availability - the data stored on your MySQL server needs to be accessible 24 x 7.
• frequent backups - to protect against data loss, you often back up your databases.
• mixed processing profiles - your MySQL database server must field requests from online transaction process (OLTP) and decision support system (DSS) users.
• abundant, low-performance computers - your organization might not have the fastest computers, but they have lots of them.
• widely dispersed users - your MySQL users are spread among multiple locations.
• modular application code - your MySQL-based applications can be easily altered to read data from the slave servers while writing data to the master.

3. MYSQL DATABASES REPLICATION

There are some topologies that are known for replication. MySQL is supporting the following ones:
• *Central publisher*
• *Central subscriber*
• *Central publisher with remote distributor*
• *Central distributor*
• *Publishing subscriber*
An example of central publisher with remote distributor topologie is showing in Fig.1.

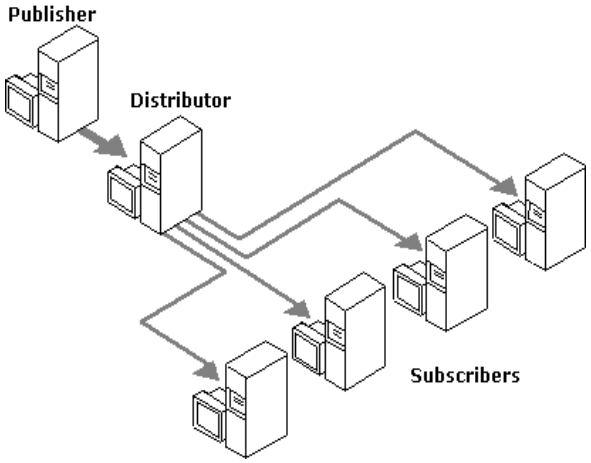

Fig.1. Central publisher with remote distributor

Also for replication there are defined the following types that are supported also be MySQL. Because we are talking about MySQL in this paper, we will say that the types of replications that are supported by it are: Snapshot, Transactional, Merge. For further information about those types see http://msdn.microsoft.com/en-us/library/Aa198189.
There are also available a lot of tools for replication in MySQL, called agents, for instance Microsoft SQL Server 7.0/2000 supports the following replication agents:
• *Snapshot Agent*
• *Log Reader Agent*
• *Distribution Agent*
• *Merge Agent*
The *Snapshot Agent* is a replication agent that makes snapshot files, stores the snapshot on the Distributor, and records information about the synchronization status in the distribution database.
The *Log Reader Agent* is a replication agent that moves transactions marked for replication from the transaction log on the Publisher to the distribution database.
The *Distribution Agent* is a replication agent that moves the snapshot jobs from the distribution database to Subscribers, and moves all transactions waiting to be distributed to Subscribers. The Distribution Agent is used in Snapshot and Transactional replications and can be administered by using SQL Server Enterprise Manager.
The *Merge Agent* is a replication agent that applies initial snapshot jobs from the publication database tables to Subscribers, and merges incremental data changes that have

occurred since the initial snapshot was created. The Merge Agent is used only in Merge replication.

4. REPLICATION AND TRANSACTIONAL CONSISTENCY

Research in distributed DBMS has focused on the problem of access to distributed data (the problem of translating and decomposing a database update or query into local updates or retrievals at a set of communicating sites), but has not paid much attention to administration of distributed data. In a distributed system, it is desirable to consider distribution and replication as physical details of an object's representation. Replication transparency means that the user is unaware that a logical object may be represented by multiple physical copies.
In case of access control with site autonomy, sites can control a user's ability to access individual copies of data. This violates transparency because a user may be able to access some, but not all, copies of a data item, and the behavior of a query may vary depending on the copy selected. Furthermore, the DBMS cannot guarantee mutual consistency of replicated data if it cannot update all copies in an identical fashion(Kenneth et al.,2008).
In Fig.2 is showing how the replication would violate transactional consistency for different types of replications.

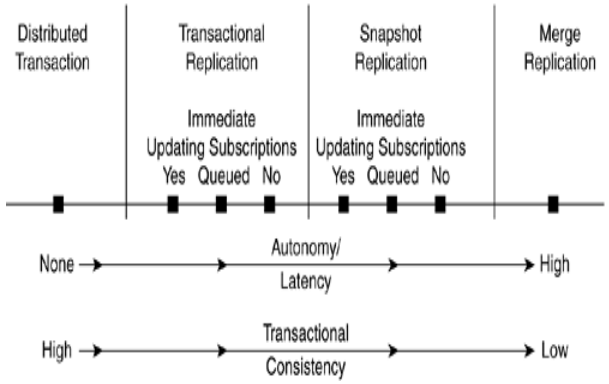

Fig. 2. Replication vs transactional consistency

5. REPLICATION OVER THE INTERNET

Synchronous multi-master replication, which is called clustering in MySQL is supported using distributed Transactions. In a way to create a multi master replication schema is proposed, but without conflict resolution(Maxia, 2006). This is basically done by circularly defining master-slave relationships, and avoiding to replicate data, which originated from the same server.

In MySQL it not possible to have transparency in replication for distributed environments.
That's why it's a common practice to use the DBMS over the Internet. In this case there are some pre-requisites:
• there has to be a MySQL database available for querying over the Internet;
• non-updating queries are in a higher number than updates;
• only a restricted set of users are performing updates via Intranet application;
• reads are made by the use of an Internet site;
• the Internet interface of the database is an application that resides on a HTTP server
When a replication group uses synchronous replication, all of the sites in the group need to be up and running in order for any of the sites to be updatable. This is because with synchronous replication, each transaction needs to be applied immediately to all of the sites in the group, or else the entire transaction will be rolled back(Keating, 2001).

6. HIGH-AVAILABILITY IN CASE OF FAILOVER

There are some cases where we need to have high availability and great protection against failover and for that we can talk about some actions to provide that. First, we are talking about shared sorage, where multiple MySQL servers are ready to access the storage and begin to send requests. It is also possible to have replicated storage between two MySQL servers. Most often two standard servers with local storage use DRBD to replicate the MySQL file system. In a failover situation, the standby server will mount the replica file system before starting up MySQL, just the same as in the SAN-based example above. This configuration also makes sense in other configurations, where the storage back-end also supports replication, to avoid being dependent upon a single storage device. You can even create two groups of clusters using creative storage replication instead of MySQL-level replication.
Standby failover with MySQL works extremely well, and data integrity is guaranteed as far as the last successful transaction before the primary server stopped responding. In that sense, failover is safer than replication.
Replica storage, replica MySQL servers, clusters failing over; With every replication, failover, and clustering technology used, the fear that data inconsistency will afflict you becomes greater, so be sure that it's really necessary. In the event of a crash, even greater care must be paid to recovery procedures, because a mistake can quickly spell disaster.

7. CONCLUSIONS

Replication is designed as a data distribution mechanism. At the most basic level, changes made to one database are distributed to one or more targets. The core replication engine is designed for flexible implementation, but the core architecture can be leveraged to provide availability for a database because a redundant copy of data is maintained in synchronization with a master copy.
Multimaster replication in distributed databases is a very powerful and flexible feature. However, with power and flexibility comes complexity.
It is possible to create multiple replication groups within a single database. Using multiple replication groups allows a database's objects to be "partitioned" between different replication groups.

8. REFERENCES

Maxia, G. (2006). *Advanced MySQL Replication*. Retrieved 2010, from

Keating, B. (2001). *Challenges Involved in Multimaster Replication*. Retrieved 2010, from

Kenneth ,R. ;Abbott. D.& McCarthy R.(2008). *Proceedings of the 14th VLDB Conference*, Los Angeles, California

***(2008)http://msdn.microsoft.com/en-us/library/Aa198189, *Accessed on: 2010-06-12*

*** www.dbspecialists.com/files/presentations/mm_replication.html

*** www.onlamp.com/pub/a/onlamp/2006/04/20/advanced-mysql-replication.html