Study of the Master-Slave replication in a distributed database*

Kalonji Kalala Hercule[1]*, Mbuyi Mukendi Eugene[2], Boale Bomolo Paulin[3], Lilongo Bokaletumba Joel[4]

1.Department of Computer Science, Laboratoire d'information de Kinshasa, University of Kinshasa (unikin), Kinshasa, DR Congo.

2.Department of Computer Science, Laboratoire d'information de Kinshasa, University of Kinshasa (unikin), Kinshasa, DR Congo

3.Department of Computer Science, Laboratoire d'information de Kinshasa, University of Kinshasa (unikin), Kinshasa, DR Congo

4.Department of Computer Science, Laboratoire d'information de Kinshasa, University of Kinshasa (unikin), Kinshasa, DR Congo

Abstract

In a distributed database, data replication can be used to increase reliability, and availability of data. Updating a copy should be passed automatically to all its replicas. Generally, an update of data by the peer who has a new version involves not spread to those replicating this data. No form of consistency between replicas is guaranteed [13]. In this paper we propose the study "Replication Master-Slave", which is a way of replicating data used in a distributed database. We will do a brief overview on some principles of distributed databases. Then we will present the different types of replication, the value of using this mode of replication Master exclave. Then we will end up a banking application based on the replication Master-Slave on the Oracle 10g platform.

Keys words: *Distributed Database, replication, fragmentation*

I. Introduction

We consider a model of distributed database system similar to the model in [14-16]. Distributed database is a collection of multiple, logically interrelated database distributed over computer network [8]. In this model, a distributed database system consists of a set of data items residing at various sites. Sites can exchange information via messages transmitted on a communication network, which is assumed to be reliable. The distributed databases are born of the needs of the connected remote databases via a computer network while maintaining transparency in terms of localization, partitioning, and replication relative to the user. The replication of the databases is a well-known technique to improve the performances of the system and to avoid the failures of the sites [1-3-9-19]. Replication was commonly applied to distributed file systems to increase availability and fault tolerance [15]. Replication of databases is by definition the process of copying and maintaining database objects (such as tables, indexes) in databases that form a system of distributed database. Replication allows management of multiple copies that can diverge. Multiple copies may have different values at a given moment, but converging to the same values at another moment. Replication is an important mechanism because it allows organizations to provide their users with access to current data when they need it. It improves performance and increases data availability. Replication is a technique aimed at achieving goals toward both availability and fault-tolerance [22-23].

Replication object

A replication object is an object database such as a relationship, index, view, function or procedure that exists on multiple servers in a distributed database In a replication environment type, any changes to an object of this replication in a site, is reflected in all copies of that object in the other sites [3-16-9-21].

A replication group

In a replication environment, replication objects are managed using replication groups. A replication group is a collection of replication objects logically related [9-16]. The organization database connected to a replication group simplifies management of these objects.

Masters and slaves sites

A replication group can exist in multiple sites of replication. Replication environments support two basic types of sites: [24] masters sites and slaves sites. A replication group can be associated with one or more masters sites and slaves one or more sites. A same site [26] can be both a master site for a specified replication group and a slave site for another replication group, but one site can never be a master site and slave site for the same replication group. A Master Site controls a replication group and objects, maintaining a complete copy of all objects in the replication group and propagates any changes to group all copies located at Slaves sites. A slave site may contain all objects of a replication group or only a subset. [25]The slaves do not host sites, however only a snapshot of a replication group, as, for example, data of a relationship, captured at a given moment. Sites that contain snapshots are usually refreshed periodically to synchronize them with their masters sites. In the case of a replication environment with more than one master site, all masters sites communicate directly with one another to continually propagate data changes that occur in the replication groups.

Propagation strategy

When propagating an update to peer replicas, there are options of either synchronous or asynchronous propagation. In synchronous propagation, an update is sent to peer replicas before the result is returned to the client. In contrast, in asynchronous propagation the update is propagated after the result is returned to the client. Asynchronous propagation gives a better response time to the client. Replication protocols developed in the transactional model are examples of the synchronous propagation; a write operation has to be confirmed to have performed on all replicas before the result is returned. An asynchronous propagation example is demonstrated by the lazy replication architecture, where a set of updates performed at one replica is sent to other replicas in gossip messages from time to time.[17- 9-16 -13]

Asymmetric and symmetric replication

Asymmetrical replication is a technique for managing copies based on primary site only allowed to update and responsible for disseminating updates to the secondary copies. [18-16] The Symmetric replication allows simultaneous updates of all copies of different transactions. All sites can be updated.

Master-Slave property [24][25][26]

Duplicate data asynchronously are the property of a single site, the master or primary site, and may be updated by this site alone. Following the metaphor of the publication and subscription, the master site (the publisher) makes data available sites slaves (subscribers). Sites slaves subscribe to data held by the master site, which means they receive read-only copies on their own local systems. Any site may potentially be the master site data sets do not overlap, but only one site can update the master copy of a particular set of data, so that no dispute can not updated occur among sites. Here are some sample applications that illustrate the potential of this particular type of replication

The distribution and spread of centralized information

Dissemination of data describes an environment where data is updated in a central location and then replicated in read-only sites. For example, information on products, such as the list prices, could be maintained at the headquarters of a company and duplicated the form of read-only copies housed in subsidiaries (Fig 1a) [19-20]. The consolidation of data describes an environment where data is updated locally, and then met in a directory read-only in a specific location. This method gives ownership of data at each site and

gives it certain autonomy. For example, details of properties maintained in each agency are duplicated in a copy of read-only preserved building on the site of the headquarters of the company. (Figure 1 b)

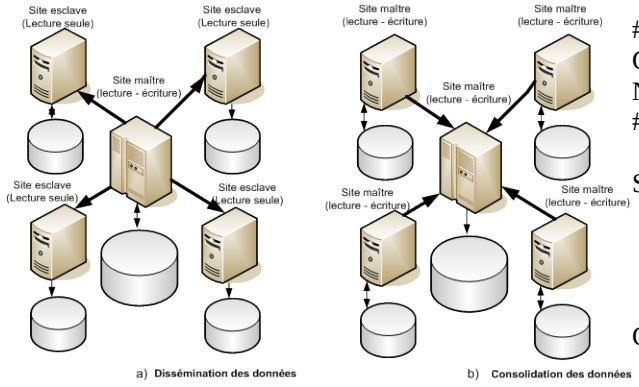

Fig. 1:Master Slave property (a.Data dissemination and b. Data consolidation

II.APPLICATION

The application consists of 3 sites:

- The central site Agency1: Who gathered all the data about customers, employees, accounts, agencies, operations that take place in other agencies (Agency2, Agency3). The central location also allows for creating and managing accounts for clients.

- The Agency2: Provides the ability to create and manage accounts for clients.

- The Agency2: Provides the ability to create and manage accounts for clients.

Representation of the conceptual model of the data

EMPLOYE (NumEmp,NomEmp,PrenomEmp,NumAg)
COMPTE (NumCompte,Numclient,TypeCompte,Somme,NumAg)
CLIENT (NumCli,Nom,Prenom,Age,AdresseAg)
AGENCE (NumAg,NomAg,AdresseAg)
OPERATION (NumOp,NumEmp,NumCompte,Montant,TypeCompt,DateOP)

Configuration

- Configuration du fichier listener.ora

```
# listener.ora Network Configuration File: C:\oracle\product\10.1.0\Db_1\NETWORK\ADMIN\listener.ora
# Generated by Oracle configuration tools.

SID_LIST_LISTENER =
 (SID_LIST =
  (SID_DESC =
   (SID_NAME = PLSExtProc)
   (ORACLE_HOME = C:\oracle\product\10.1.0\Db_1)
   (PROGRAM = extproc)
  )
 )

LISTENER =
 (DESCRIPTION_LIST =
  (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
  )
  (DESCRIPTION =
   (ADDRESS = (PROTOCOL = TCP)(HOST = DJO)(PORT = 1521))
  )
 )
```

- Configuration du fichier tnsnames.ora

```
# tnsnames.ora Network Configuration File: C:\Oracle\product\10.1.0\Client_1\network\admin\tnsnames.ora
# Generated by Oracle configuration tools.

AGENCE3 =
 (DESCRIPTION =
  (ADDRESS_LIST =
   (ADDRESS = (PROTOCOL = TCP)(HOST = DJO)(PORT = 1521))
  )
  (CONNECT_DATA =
   (SERVICE_NAME = AGENCE3)
  )
 )

AGENCE2 =
 (DESCRIPTION =
  (ADDRESS_LIST =
   (ADDRESS = (PROTOCOL = TCP)(HOST = DJO)(PORT = 1521))
  )
  (CONNECT_DATA =
```

```
    (SERVICE_NAME = AGENCE2)
   )
 )

AGENCE1 =
 (DESCRIPTION =
   (ADDRESS_LIST =
    (ADDRESS = (PROTOCOL = TCP)(HOST =
DJO)(PORT = 1521))
   )
   (CONNECT_DATA =
    (SERVICE_NAME = AGENCE1)
   )
 )

DBPROD =
 (DESCRIPTION =
   (ADDRESS_LIST =
    (ADDRESS = (PROTOCOL = TCP)(HOST =
DJO)(PORT = 1521))
   )
   (CONNECT_DATA =
    (SERVICE_NAME = DBPROD)
   )
 )

DBTEST =
 (DESCRIPTION =
   (ADDRESS_LIST =
    (ADDRESS = (PROTOCOL = TCP)(HOST =
DJO)(PORT = 1521))
   )
   (CONNECT_DATA =
    (SERVICE_NAME = DBTEST)
   )
 )
```

Creation des tables

Client table

```
SQL*Plus: Release 10.1.0.2.0 - Production on Jeu. Avr. 29 0

Copyright (c) 1982, 2004, Oracle.  All rights reserved.


Connecté à :
Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 -
With the Partitioning, OLAP and Data Mining options

SQL> create table CLIENT ( Num_Cli number(10) not null,
  2                        Nom char(30) not null,
  3                        Prenom char(30) not null,
  4                        Age number(10) not null,
  5                      Primary key(Num_Cli ));

Table créée.
```

Agence table

```
acle SQL*Plus
Edition  Recherche  Options  Aide
Create table AGENCE ( Num_Ag number(10) not null,
                      Nom_Ag char(30) not null,
                      Adresse char(30) not null,
                    Primary key(Num_Ag ));

e créée.
```

Employé table

```
acle SQL*Plus
Edition  Recherche  Options  Aide
Create table EMPLOYE ( Num_Emp number(10) not null,
                       Nom_Emp char(30) not null,
                       Prenom_Emp char(30) not null,
                       Num_Ag number(10) not null,
                     Primary key(Num_Emp),
                     foreign key(Num_Ag) references AGENCE );

e créée.
```

Creation table Compte

```
cle SQL*Plus
Edition  Recherche  Options  Aide
Create table COMPTE (Num_Compte number(10) not null,
                     Num_Cli number(10) not null,
                     Type_Compte char(30) not null,
                     Somme number(10) not null,
                     Num_Ag number(10) not null,
                   primary key(Num_Compte),
                   foreign key(Num_Ag) references AGENCE,
                   foreign key(Num_Cli) references CLIENT);

créée.
```

Creation table operation

```
cle SQL*Plus
Edition  Recherche  Options  Aide
Create table OPERATION ( Num_Op number(10) not null,
                         Num_Emp number(10) not null,
                         Num_Compte number(10) not null,
                         Montant number(10) not null,
                         Type_Compte char(30) not null,
                         Date_ char(10) not null,
                       primary key(Num_Op),
                       foreign key(Num_Compte) references COMPTE,
                       foreign key(Num_Emp) references EMPLOYE );

créée.
```

Création de des liens et des synonymes

- Liens et synonymes sur AGENCE1

CREATE DATABASE LINK l1
CONNECT TO system
IDENTIFIED BY joe
USING 'AGENCE2';

```
CREATE SYNONYM CLIENT2 FOR CLIENT@l1;
CREATE SYNONYM AGENCE2 FOR AGENCE@l1;
CREATE SYNONYM EMPLOYE2 FOR EMPLOYE@l1;
CREATE SYNONYM COMPTE2 FOR COMPTE@l1;
CREATE SYNONYM OPERATION2 FOR OPERATION@l1;

CREATE DATABASE LINK l3
CONNECT TO system
IDENTIFIED BY joe
USING 'AGENCE3';
CREATE SYNONYM CLIENT3 FOR CLIENT@l3;
CREATE SYNONYM AGENCE3 FOR AGENCE@l3;
CREATE SYNONYM EMPLOYE3 FOR EMPLOYE@l3;
CREATE SYNONYM COMPTE3 FOR COMPTE@l3;
CREATE SYNONYM OPERATION3 FOR OPERATION@l3;
```

- Liens et synonymes sur AGENCE2

```
CREATE DATABASE LINK l2
CONNECT TO system
IDENTIFIED BY joe
USING 'AGENCE1';
CREATE SYNONYM CLIENT1 FOR CLIENT@l2;
CREATE SYNONYM AGENCE1 FOR AGENCE@l2;
CREATE SYNONYM EMPLOYE1 FOR EMPLOYE@l2;
CREATE SYNONYM COMPTE1 FOR COMPTE@l2;
CREATE SYNONYM OPERATION1 FOR OPERATION@l2;

CREATE DATABASE LINK l3
CONNECT TO system
IDENTIFIED BY joe
USING 'AGENCE3';
CREATE SYNONYM CLIENT3 FOR CLIENT@l3;
CREATE SYNONYM AGENCE3 FOR AGENCE@l3;
CREATE SYNONYM EMPLOYE3 FOR EMPLOYE@l3;
CREATE SYNONYM COMPTE3 FOR COMPTE@l3;
CREATE SYNONYM OPERATION3 FOR OPERATION@l3;
```

- Liens et synonymes sur AGENCE3

```
CREATE DATABASE LINK l3
CONNECT TO system
IDENTIFIED BY joe
USING 'AGENCE1';
CREATE SYNONYM CLIENT1 FOR CLIENT@l3;
CREATE SYNONYM AGENCE1 FOR AGENCE@l3;
CREATE SYNONYM EMPLOYE1 FOR EMPLOYE@l3;
CREATE SYNONYM COMPTE1 FOR COMPTE@l3;
CREATE SYNONYM OPERATION1 FOR OPERATION@l3;

CREATE DATABASE LINK l2
CONNECT TO system
IDENTIFIED BY joe
USING 'AGENCE2';
CREATE SYNONYM CLIENT2 FOR CLIENT@l2;
CREATE SYNONYM AGENCE2 FOR AGENCE@l2;
CREATE SYNONYM EMPLOYE2 FOR EMPLOYE@l2;
CREATE SYNONYM COMPTE2 FOR COMPTE@l2;
CREATE SYNONYM OPERATION2 FOR OPERATION@l2;
```

- **Data replication**

1. Using native command of sqlplus ie copy

- AGENCE 2

```
copy from system/joe@Agence1 to
system/joe@Agence2 Create CLIENT_T
(NUM_CLI,NOM,PRENOM,AGE) Using
SELECT * FROM CLIENT;
copy from system/joe@Agence1 to
system/joe@Agence2 Create COMPTE_T
(NUM_COMPTE,NUM_CLI,TYPE_CO
MPTE,SOMME,NUM_AG) Using
SELECT * FROM COMPTE;
copy from system/joe@Agence1 to
system/joe@Agence2 Create
OPERATION_T
(NUM_OP,NUM_EMP,NUM_COMPTE,
MONTANT,TYPE_COMPTE,DATE_)
Using SELECT * FROM OPERATION;
```

- AGENCE3

```
copy from system/joe@Agence1 to
system/joe@Agence3 Create CLIENT_T
```

(NUM_CLI,NOM,PRENOM,AGE) Using SELECT * FROM CLIENT;

copy from system/joe@Agence1 to system/joe@Agence3 Create COMPTE_T (NUM_COMPTE,NUM_CLI,TYPE_COMPTE,SOMME,NUM_AG) Using SELECT * FROM COMPTE;

copy from system/joe@Agence1 to system/joe@Agence3 Create OPERATION_T (NUM_OP,NUM_EMP,NUM_COMPTE,MONTANT,TYPE_COMPTE,DATE_) Using SELECT * FROM OPERATION;

The disadvantage of this method of replication: data can't upgrade but you use replace command to change contain of tables.

2. Using the snapshot

Two types of snapshot can create: a simple or complex. A simple snapshot can't contain a clause in list below : distinct, connect by , group by, join and operation set. The snapshot below extract master data and renew the operation three day after:

```
Create snapshot client_t
Tablespace data2
Storage (initial 100K next 100K PCINCREASE 0)
Refresh fast
Start with sysdate
Next sysdate + 3
As select * from client@link2;
```

Refresh fast use a snapshot log to upgrade a snapshot. This file locates on the same site of master table. In snapshot log, stored all modification intervene on master table. Thus, for each update, only the modifications which are sent, and not the whole of the data. On the other hand, a COMPLETE REFRESH is obligatory for the complex snapshots.

The snapshot log is to be created before the snapshot:

```
Create snapshot log on client
tablespace data
Storage (initial 10k next 10k pctincrease 0)
pctfree 5 pctused 90;
```

A traditional use of the snapshots in update is agencies. All the agencies store data concerning the accounts which they have created during the day. Each night, the data are poured in the national base which centralizes the data of the bank. Let us note that, the snapshots in update can generate conflicts. A release (trigger) saves the updates operated on the snapshot and transmits them to the main site at the time of the cooling of the snapshot.

3. Using materialized view

The materialized views are view whose contents are physically present (contrary to the traditional view which is select with delayed-action). The contents must be regularly refreshed. Several strategies are established.

3.1. Refresh complete

Create materialized view client_t refresh complete as select * from client@link2;

The request will be carried out on the distant table and the result will replace the current contents. Can be very greedy in resource and time for the bulky tables.

3.2. Refresh fast

Create materialized view client refresh fast as select * from client@link2;

The maitre records the operations made on the table; during a refresh of the replica only modifications since the last refresh one will be downloaded and applied locally.

A materialized view can bring several advantages to the level performances. According to the complexity of the request, we can fill it with changes, by means of the log of materialized views (MATERIALIZED VIEW LOG), instead of recreating it.

Contrary to snapshots, the materialized views can be directly used by the optimizer, to modify the path of execution of the requests.

III.CONCLUSION

In this paper, we presented how we can use the Master-Slave replication. We have shown the benefits of this type of replication, how it works, and ways to use depending on whether the updates are immediate or delayed. And we used an application that we designed to implement and demonstrate how this type of replication works. This application, we implemented the Oracle 10g. We also want to point out that this type of replication is suitable for applications in environments from the analysis of a system of decision support, or data from one or more databases are copied and loaded into a separate

system of decision support for read-only analysis, in applications of the distribution and dissemination of information centralized, in consolidating information from remote and finally in applications of mobile computing or itinerant.

IV.REFERENCE

[1] G. Gardarin et O. Gardarin. "The Client-Server", Ed. Eyrolles 1996
[2] R. Bizoi. "Oracle 10g Administration", Tsoft Editeur, Ed. Eyrolles 2005
[3] S. Pelagatti. "Distributed database: Principes and systems", International edition. Copyright 1985.
[4] T. connolly and C. Beqq. "Systèmes de base de données – Approche pratique de la conception. De l'implémentation et de l'administration ",Edition Eyrolles, Reynald Goulet Juillet 2005.
[5] R. katz, e. wong. "Resolving Conflicts in Global Storage Design through Replication ", ACM Transactions on Database Systems, 1983
[6] M. Özsu, P. Valduriez. "Distributed database systems: where are we now?", IEEE Computer, 1991
[7] S. Upadhyaya, S. Lata. "Task allocation in Distributed computing VS distributed database systems : A Comparative study", IJCSNS International Journal of Computer Science and Network Security, Vol.8 N°3, 2008
[8] A. Singh K.S. Kahlon. "Non-replicated Dynamic Data Allocation in Distributed Database Systems", IJCSNS International Journal of Computer Science and Network Security, Vol.9, September 2009.
[9] J.E. Armendariz, J.R. Juarez1, J.R. Gonzaalez de Mendivil, F.D. Munoz "Correctness Criteria for Replicated Database Systems with Snapshot Isolation Replicas" Technical Report ITI-ITE-08/03, 2008.
[10] M. Özsu, P. Valduriez. "Distributed and Parallel Database Systems", 1991
[11] N. Conway, G. Sherry. "A Proposal for a Multi-Master Synchronous Replication" 2006.
[12] H. George, L. Fletcher, J. Bussche, D. Gucht, S. Vansummeren."Towards a Theory of Search Queries "ACM Transactions on Database Systems, 2010.

[13] E. Pacitti. "Réplication Asynchrone des données dans trois contextes". Laboratoire d'Informatique de Nantes Atlantique. Université de Nantes, France, 2008.

[14] J. Wu, D. Manivannan and B. Thuraisingham "Necessary and sufficient conditions for transaction-consistent global checkpoint in a distributed database system", Information Sciences, Vol 179, pp 3659–3672, 2009.
[15] Q. Wenyu, L. Keqiu, B. Jiang, H. Shen, and D. Wu, "Dynamically Selecting Distribution Strategies for Web Documents According to Access Pattern", IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.3A, March 2006.
[16] K. Segun, R. Hurson, V. Desai and A. Spink, "Transaction Management in a Mobile Data Access System", Parallel Computing Technologies Lecture Notes in Computer Science, Springer, Vol, pp.112-127 2127, 2001.
[17] W. Zhou, L. Wang and W. Jia, "An analysis of update ordering in distributed replication systems", Future Generation Computer Systems, Vol 20, pp 565–590, 2004.
[18] M. Masud, I. Kiringa, "Transaction processing in a peer to peer database network", Data & Knowledge Engineering, Vol 70, pp 307–334, 2011.
[19] I. Roussaki, M. Strimpakou, C. Pils, N. Kalatzis and N. Liampotis, "Optimising context data dissemination and storage in distributed pervasive computing systems", Pervasive and Mobile Computing, Vol6, pp,218-238, 2010.
[20] L. Wujuan, B. Veeravalli, "Design and analysis of an adaptive object replication algorithm in distributed network systems", Computer Communications, Vol31, pp 2005–2015, 2008.
[21] P. Beran, W. Mach, E. Schikuta and R. Vigne, "A Multi-Staged Blackboard Query Optimization Framework for World-Spanning Distributed Database Resources", Procedia Computer Science Vol 4, pp156–165, 2011.
[22] W. Zhou, L. Wang and W. Jia, "An analysis of update ordering in distributed replication systems", Future Generation Computer Systems Vol20, pp565–590, 2004.
[23] S. Khan, I. Ahmad, "Comparison and analysis of ten static heuristics-based Internet data replication techniques ", Journal of Parallel and Distributed Computing, Vol 68, pp113 – 136, 2008.
[24] F. Shrouf, M. Eshtay and K. Humaidan, "Performance Optimization for Mobile Agent Message Broadcast Model Using V-Agent", IJCSNS International Journal of Computer Science and Network Security, Vol8, August 2008. (maître Esclave)

[25] S. Gançarsk, H. Naacke, E. Pacitti and P. Valduriez, "The leganet system: Freshness-aware transaction routingin a database cluster", Information Systems, Vol32, pp320–343, 2007.
[26] F. Silva, H. Senger"Improving scalability of Bag-of-Tasks applications running on master–slave platforms", Parallel Computing, Vol35, pp57–71, 2009.

[26] F. Silva, H. Senger"Improving scalability of Bag-of-Tasks applications running on master–slave platforms", Parallel Computing, Vol35, pp57–71, 2009.