Synchronisation Mysql (Replication)

[**Petit avertissement :** Bon, après relecture, je constate que c'est l'un des plus mauvais document que j'ai écrit. Mais bon, il est quand même utile ce torchon.]
Nous allons dans ce petit document décrire comment nous faisons de la synchronisation de deux serveurs de bases de données mysql. En fait, nous avons un serveur qui ne fera que recopier les données du premier serveur ; c'est de la réplication de données.

A- Principes :

- Deux serveurs ; l'un est maître et possède les données à jour ; l'autre est esclave du maître et se met à jour à partir de celui-ci.
- Le serveur esclave démarre avec une copie des données du serveur maître (données les plus à jour possible).
- Le serveur maitre log les modifications des données à son niveau dans un fichier log et notifie le ou les esclaves.
- Le serveur esclave lit les logs du maître qui sont des requêtes qu'il exécute pour avoir les mêmes données que le maître.

B- Implémentation Mysql :

B.1-Au niveau du maitre :

Ajoutons les options suivantes dans la section [mysqld] du fichier de config de mysql « /etc/mysql/my.cnf » :

```
server-id            = 1
log-bin              = /var/log/mysql/mysql-bin.log
```

On active les logs binaires et on spécifie l'id du serveur qui est un entier et qui doit être différent des id des esclaves.

On peut également préciser les bases de données à synchroniser et celles à exclure de la synchronisation avec ces options :

```
binlog-do-db         = BD_synchroniser
binlog-ignore-db     = mysql
```

Après toutes ces modifs dans le fichier de conf, il faut évidement redémarrer le serveur mysql. Il ne faut surtout pas avoir peur d'un message de ce genre lorsque mysqld redémarre : « Checking for corrupt, not cleanly closed and upgrade needing tables.. »

Ensuite :
mysql -u root -p : pour passer quelques commandes MySQL.

On va créer un utilisateur ***sync*** qui aura les droits pour faire la replication. C'est avec cet utilisateur

que l'esclave se connectera au maître.

mysql> *grant replication slave on *.* to sync@% identified by 'lemotdepassedesync';*

mysql> *grant super, reload, select on *.* to sync@% identified by 'lemotdepassedesync';*

L'utilisateur sync étant créé, nous allons passer une commande qui va nous permettre d'avoir la position du serveur MYSQL dans ses logs binaires. Ce sont les logs binaires qui permettent aux esclaves de se resynchroniser, mais ceux-ci se resynchronisent à partir d'une certaine position (pas forcément à partir de 0). Après, on donne à l'esclave une copie de la base de données jusqu'à cette position, et il se mettra à jour. Voilà !

mysql> *show master status ;*

```
+--------------------+----------+--------------------+----------------------+
| File               | Position | Binlog_Do_DB       | Binlog_Ignore_DB     |
+--------------------+----------+--------------------+----------------------+
| mysql-bin.000788   |      815 | BD_synchroniser    | mysql                |
+--------------------+----------+--------------------+----------------------+
```

Il faut donc noter sur un papier le nom du fichier **mysql-bin.000788** et la position **815**. On constate aussi au passage que la base qui sera synchronisée est bien **BD_synchroniser**.

Ensuite, il faut prendre une copie de la base de données actuelle et la charger sur le client. Plusieurs techniques existent. Nous on fait juste une copie tar.gz du dossier de données de mysql /var/lib/mysql/ qu'on transfère sur l'esclave avec scp :
cd /var/lib/mysql
tar -czvf BD_synchroniser.tgz BD_synchroniser
scp BD_synchroniser.tgz user@slave_server:/var/lib/mysql

(Notons qu'il peut être parfois intéressant de locker la base de données en lecture seule avant de prendre le snapshot).

Y'a plus rien à faire sur le maître, on passe au slave.

On peut tout de même vérifier que les logs bianires sont bien renseignés avant de passer à l'esclave. Selon notre config, ces logs devraient se trouver dans « /var/log/mysql/ » et être du genre mysql-bin.000788. Pour tester, il suffit de faire des mises à jour de données SQL (update, insert, delete) et de voir si ces fichiers changent (juste en regardant la date de modification). Si ces fichiers ne changent pas, c'est qu'il y'a des erreurs de config au niveau du maître. Régler cela avant de passer au slave. Nous, on passe au slave.

B.2 - Au niveau de l'esclave :

L'esclave n'a pas besoin de produire de logs binaires. Il s'agit donc de spécifier son id qui doit être différent de celui du maître (et des autres esclaves).

Modifications du fichier de config :

```
server-id            = 2
#log_bin             = /var/log/mysql/mysql-bin.log
report_host          = <nomduserveuresclave>
```

Le paramètre **report_host** permet à l'esclave d'indiquer son nom au maître. Et donc avec ce paramètre, si on tape la commande ***show slave hosts*** au niveau du maître, ce nom s'affiche. En gros, ça permet de lister les esclaves au niveau du maître.

Ensuite, il faut restaurer la sauvegarde prise sur le serveur.

```
# cd /var/lib/mysql
# tar -xzvf BD_synchroniser.tgz
```

Puis redémarrage de mysqld.

Enfin, allons y taper quelques commandes MySQL pour préciser qui est le maître et démarrer l'esclave:

```
# mysql -u root -p
mysql> CHANGE MASTER TO
    -> MASTER_HOST='<IP_Seveur_mâitre>',
    -> MASTER_USER='sync',
    -> MASTER_PASSWORD='motdepassedesync',
    -> MASTER_LOG_FILE='mysql-bin.000788',
    -> MASTER_LOG_POS=815;
```

On précise ainsi le serveur, l'utilisateur, et son mot de passe créé sur le serveur ainsi que la position que nous avions noté.

```
mysql> start slave;
```

Puis, on démarre le slave

```
mysql> load data from master;
```

On charge les données du maître. A partir de là, les 2 bases de données sont synchronisées, et toutes modifications sur le maître se répercute sur l'esclave.
A noter que l'esclave n'a pas forcement besoin d'être en ligne, il peut se connecter de temps en temps et se mettre à jour.

On lance une commande qui permet de voir les process en cours et savoir si notre slave tourne bien.

```
mysql> show processlist\G ;
*************************** 1. row ***************************
     Id: 12
   User: root
   Host: localhost
     db: NULL
Command: Query
   Time: 0
  State: NULL
   Info: show processlist
*************************** 2. row ***************************
     Id: 15
   User: system user
```

```
   Host:
     db: NULL
Command: Connect
   Time: 6
  State: Waiting for master to send event
   Info: NULL
*************************** 3. row ***************************
     Id: 16
   User: system user
   Host:
     db: NULL
Command: Connect
   Time: 6
  State: Has read all relay log; waiting for the slave I/O thread to update it
   Info: NULL
3 rows in set (0.00 sec)
```

Les 2 derniers process nous montrent que l'esclave fonctionne bien : un process qui est connecté au maître et attend les évènements de ce dernier, et l'autre process qui applique les modifications reçues du maître.

Une autre commande intéressante :
mysql> *show slave status\G;*

```
*************************** 1. row ***************************
             Slave_IO_State: Waiting for master to send event
                Master_Host: <IP_Serveur_maître>
                Master_User: sync
                Master_Port: 3306
              Connect_Retry: 60
            Master_Log_File: mysql-bin.000788
        Read_Master_Log_Pos: 815
             Relay_Log_File: <serveur_maitre>-relay-bin.000002
              Relay_Log_Pos: 235
      Relay_Master_Log_File: mysql-bin.000788
           Slave_IO_Running: Yes
          Slave_SQL_Running: Yes
            Replicate_Do_DB:
        Replicate_Ignore_DB:
         Replicate_Do_Table:
     Replicate_Ignore_Table:
    Replicate_Wild_Do_Table:
Replicate_Wild_Ignore_Table:
                 Last_Errno: 0
                 Last_Error:
               Skip_Counter: 0
        Exec_Master_Log_Pos: 815
            Relay_Log_Space: 235
            Until_Condition: None
             Until_Log_File:
              Until_Log_Pos: 0
```

```
Master_SSL_Allowed: No
Master_SSL_CA_File:
Master_SSL_CA_Path:
Master_SSL_Cert:
Master_SSL_Cipher:
Master_SSL_Key:
Seconds_Behind_Master: 0
```

La dernière ligne nous indique que l'esclave est bien synchronisé avec le serveur puisqu'il est à 0 seconde derrière le serveur.
Et si on regarde dans le répertoire des données de mysql, on devrait voir des fichiers de ce genre : **relay-log.info, master.info** (qui contient les infos de connexion au serveur), **<nomduserveurmaitre>-relay-bin.index**, et aucun de ces fichiers n'est à modifier manuellement. Tout ceci indique que le slave tourne bien. Donc se poser des questions le jour où ces fichiers ne sont plus là.

C - Quelques commandes supplémentaires :

C.1 - Pour gérer le slave :

mysql> *LOAD DATA FROM MASTER ;*
Charger les données du maître.
mysql> *LOAD TABLE tbl_name FROM MASTER ;*
Charger une table à partir du maître
mysql> *START SLAVE ;*
Démarre l'esclave
mysql> *STOP SLAVE ;*
Arrêt de l'esclave
mysql> *RESET SLAVE ;*
Commande à faire seulement quand l'esclave est arrêté. Tue l'esclave en quelques sorte, parce que ça demande à l'esclave d'oublier son point de synchronisation avec le serveur. Donc pour pouvoir relancer le slave, il faudra lui repréciser toutes ces informations.
mysql> *SHOW SLAVE STATUS ;*
mysql> *SHOW PROCESSLIST ;*

C.2 - Pour gérer le maitre :

mysql> *PURGE MASTER LOGS TO '<mysql-bin.xyz>' ;*
Efface les logs binaires plus vieux que mysql-bin.xyz
mysql> *PURGE MASTER LOGS BEFORE '2008-10-24 20:10:20';*
mysql> *SHOW SLAVE HOSTS ;*
Affiche les esclaves qui ont l'option report_host activée
mysql> *SHOW MASTER STATUS ;*
Le status du maître quoi... Concerne en fait les logs binaires : informations utilisées pour synchrniser l'esclave.
mysql> *SHOW MASTER LOGS ;*
Liste les logs binaires disponibles sur le maître
mysql> *SHOW BINLOG EVENTS IN '<binlog_file>';*
Affiche les requêtes SQL de ce fichier. C'est pareil que d'éditer ce fichier avec la commande

mysqlbinlog.
mysql> *FLUSH TABLES WITH READ LOCK ;*
Ferme toutes les tables et verrouille, vide le cache, et verrouille les BD en lecture. Utile pour faire un snapshot des BD.
mysql> *UNLOCK TABLES ;*
Déverrouiller les bases de données.

Sources :

http://dev.mysql.com/doc/refman/5.0/fr/replication-howto.html
http://dev.mysql.com/doc/refman/5.0/en/replication.html