Réplication mysql



La réplication MySQL permet :

- d'assurer une redondance sur une ou plusieurs bases de données avec des solutions de Haute disponibilité (ipvsadm, corosync, etc) et donc de ne jamais perdre des données.
- de mettre en place de la répartition de charge sur les requêtes, par exemple vous avez des statistiques générés sur un portail tous les X secondes, une réplication master/slave peut servir a déchargé la base principale qui récoltera les données.
  La base esclave nous servira à faire les statistiques, mais on peut aussi assurer ceci avec des solutions de haute disponibilité.

Donc, je vais vous montrer les 2 types de réplication possible sur MySQL master/slave et master/master

Master/slave

Configuration du serveur maître

Nous allons éditer le fichier configuration `my.cnf` qui est dans `/etc/mysql`

```
bind-address = XXX.XXX.XXX.XXX

server-id               = 1
log_bin                 = /var/log/mysql/mysql-bin.log
expire_logs_days        = 1 #on peut mettre plus de jours
max_binlog_size         = 100M
binlog_do_db            = DB_REPLI
```

Configuration de l'esclave

Nous allons éditer le fichier configuration `my.cnf` qui est dans `/etc/mysql`

```
bind-address = XXX.XXX.XXX.XXX

server-id               = 2
log_bin                 = /var/log/mysql/mysql-bin.log
expire_logs_days        = 1 # on peut mettre plus de jours
max_binlog_size         = 100M
replicate_do_db         = DB_REPLI
```

Copie des fichiers de la base de données

Pour simplifier la copie de la base de données, j'ai l'habitude de copier le contenu du répertoire `mysql` dans `/var/lib/`.

Dans un premier temps, j'arrête le service MYSQL sur le maître et l'esclave.

```
/etc/init.d/mysql stop
```

Nous allons installer `rsync` pour synchroniser les fichiers, j'aime bien car cela permet de garder les droits sur les fichiers, etc.

```
aptitude install rsync
```

Puis supprimer le contenu du répertoire `mysql` dans `/var/lib` de notre machine **esclave**

```
rm -rf /var/lib/mysql/*
```

Il ne reste plus qu'à lancer la synchronisation, depuis le maître.

```
rsyn -avz /var/lib/mysql/ <USER>@<IP_SLAVE>:/var/lib/mysql
```



Pour les linux type Debian/Ubuntu penser à copier le fichier `debian.cnf` dans `/etc/mysql/`, ce login sert pour la rotation des journaux et arrêter ou relancer le service MYSQL.

mise en service de la réplication



Ici, quand vous relancez mysql arrangez vous à couper tout ce qui serait susceptible d'écrire dans votre base, ça compliquerai notre affaire avec des erreurs 1062

Vous pouvez relancer Mysql.

```
/etc/init.d/mysql start
```

Sur le master

Nous allons créer un login capable de faire la réplication

```
 mysql -u root -p
 GRANT REPLICATION SLAVE ON *.* TO 'replication_user'@'<IP_SLAVE>'
IDENTIFIED BY 'motdepasse';
 FLUSH PRIVILEGES;
```

Nous pouvons regarder l'état de notre serveur maître.

```
SHOW MASTER STATUS\G
```

```
'*************************** 1. row ***************************'
            File: mysql-bin.000030
        Position: 107
    Binlog_Do_DB: example
Binlog_Ignore_DB:
1 row in set (0.00 sec)
```

voici, un résultat sur une base example

Sur le slave

Nous allons modifier les paramètres que l'esclave utilise pour se connecter et pour communiquer avec le serveur maître.

```
STOP slave;
CHANGE MASTER TO MASTER_HOST='<IP_MASTER>', MASTER_USER='replication_user',
MASTER_PASSWORD='motdepasse';
START slave;
```

Nous pouvons regarder l'état de notre serveur esclave :

```
SHOW SLAVE STATUS\G
```

```
'*************************** 1. row ***************************'
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.2.7
                  Master_User: replica
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000025
          Read_Master_Log_Pos: 107
               Relay_Log_File: mysqld-relay-bin.000082
                Relay_Log_Pos: 313
        Relay_Master_Log_File: mysql-bin.000025
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: example
          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: 107
              Relay_Log_Space: 513
              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
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2
1 row in set (0.00 sec)
```

Voici, le résultat de notre esclave qui arrive bien à joindre notre serveur maître (192.168.2.7) avec le login `replica`, la base répliquée est `example` et surtout pas d'erreur.

Master/master

Pour la réplication **master/master**, c'est très simple surtout si, vous avez réussi la réplication master/slave, il suffit de recommencer l'opération, mais cette fois dans l'autre sens.
Configurer l'esclave en maître et le maître en esclave.

From:
https://www.ksh-linux.info/ - **Know Sharing**

Permanent link:
https://www.ksh-linux.info/systeme/mysql/replication-mysql

Last update: **12/11/2016 20:36**