MySQL - Réplication

Réplication MySQL

MySQL supporte la réplication unidirectionnelle interne. Un serveur sert de maître, et les autres servent d'esclaves. Le serveur entretient des logs binaires, ainsi qu'un fichier d'index des logs binaires.

Chaque esclave, après connexion réussie au serveur maître, indique le point qu'il avait atteint depuis la fin de la dernière réplication, puis récupère les dernière modifications, puis se met en attente des prochains évènements en provenance du maître.

Un esclave peut aussi servir de maître à son tour, pour réaliser une chaîne de réplication. toute modification des tables sont répliquées, et doivent intervenir sur le serveur maître.

On peut utiliser la commande **LOAD DATA FROM MASTER** pour configurer une esclave. Mais ne fonctionne que si toutes les tables du maître sont du type **MyISAM**, et qu'il est possible d'obtenir un verrou de lecture global, pour qu'aucune lecture ne se fasse durant le transfert des tables depuis le maître.

Cette limitation est temporaire et sera supprimée une fois le système de sauvegarde programmé. On estime une vitesse de 1Mo par seconde.

Si le maître est indisponible ou que la connexion est perdue, l'esclave va tenter une reconnexion tous les **master-connect-retry** secondes.

Les capacités de réplications de MySQL sont implémentées à l'aide de 3 threads : 1 sur le maitre et 2 sur l'esclave. Lorsque la commande **START SLAVE** est envoyée, l'esclave crée un thread d'I/O qui se connecte au maître et lit les logs binaires. le maître crée un thread pour envoyer les logs binaires. Ce thread est le Binlog Dump dans le résultat de **SHOW PROCESSLIST**. Le 3eme thread lit les commandes et les exécute. pour obtenir des info sur les esclaves : **SHOW SLAVE STATUS**.

Fichiers de relais et de statut de la réplication

Par défaut ils sont nommés sous la forme **host_name-relay-bin.nnn** où **host_name** est le nom de l'esclave. l'esclave garde la trace des logs dans un fichier d'index **host_name-relay-bin.index**. Les noms par défaut se remplacent avec les options : **–relay-log** et **–relay-log-index**.

ces logs ont le même format que les logs binaires et peuvent être lus avec mysqlbinlog.

Un log est crée à chaque démarrage du serveur, par la commande **FLUSH LOGS** ou une taille supérieure à **max_relay_log_size**.

Un esclave crée 2 autres fichiers appelés **master.info** et **relay-log.info** qui contiennent des informations comme celle affichées par la commande **SHOW SLAVE STATUS**.

Si vous perdez les logs, mais que **relay-log.info** existe, on peut l'étudier pour déterminer ce que le thread SQL a traité, puis utiliser **CHANGE MASTER TO** avec les options **MASTER_RELAY_LOG** et **MASTER_RELAY_POS** pour lui faire relire les logs depuis ce point.

Mise en place de la réplication

Il est nécessaire d'éteindre brièvement le serveur principal.

Créer un utilisateur Mysql spécial pour la réplication, avec les droits de **REPLICATION SLAVE**. Il faut lui donner les droits de réplication depuis tous les esclaves. Le nom d'hôte doit être tel que chaque serveur esclave peut l'utiliser pour se connecter au maître :

GRANT REPLICATION SLAVE ON generator.php TO replication@'%s' identified by 'password'

Si on envisage d'utiliser **LOAD TABLE FROM MASTER** ou **LOAD DATA FROM MASTER** sur l'esclave, on doit donner des droits supplémentaires : **SUPER** et **RELOAD**, ainsi que les droits **SELECT** sur les tables à charger.

Si on utilise des tables MyISAM, décharger toutes les tables et blocs en utilisant **FLUSH TABLES WITH READ LOCK ;** puis faire une sauvegarde des données du maître.

Éventuellement compresser la sauvegarde
tar -cvf /tmp/mysql-snapshot.tar ./cette_base
puis copier l'archive sur l'esclave au même emplacement et la décompresser
tar -xvf /tmp/mysql-snapchot.tar
Il n'est pas besoin de répliquer la base mysql
Lorsque le verrou de lecture a été posé par
FLUSH TABLES WITH READ LOCK
lire les valeurs courantes du fichier de log et de son offset sur le maitre
SHOW MASTER STATUS ;
la colonne file montre le nom du fichier de log et la colonne position affiche l'offset. Pour enlever le verrou
UNLOCK TABLES ;
pour faire une sauvegarde rapide d'une base InnoDB le mieux est d'arrêter le serveur, puis de copier les fichiers de données InnoDB, leur logs et leur fichier de définition (.frm). Puis utiliser
SHOW MASTER STATUS
Pour noter l'offset courant. Puis éteindre le serveur sans déverrouiller le serveur et vérifier par
mysqladmin -root shutdown
Une alternative, valable pour les 2 types MyISAM et InnoDB est de prendre un export SQL du maître mais c'est plus long
mysqldump -master-data
sur le maitre, puis exécuter les commandes SQL sur les esclaves

- Si le maître fonctionne sans l'option **-log-bin**, le nom du fichier de log et l'offset seront vides, dans ce cas utiliser la chaîne vide (") comme nom de fichier de log et la valeur 4 comme offset.
- Dans my.cnf, ajouter les options **-log-bin** et **server-id=unique_number** dans le groupe **[mysqld]**. Chaque serveur doit avoir un identifiant différent.
- Copier la sauvegarde des données dans les esclaves, s'assurer que les droits sur ces données sont corrects. L'utilisateur qui fait fonctionner MySQL doit avoir les droits d'écriture et de lecture sur ces bases, comme le maître.
- Redémarrer les esclaves. lancer l'esclave avec l'option **–skip-slave-start** s'il était déjà configuré pour la réplication. Utiliser **–log-warnings** pour plus de détails.
- Si vous avez fait une sauvegarde du maître avec l'utilitaire mysqldump, charger l'export avec **mysql -u root -p < dump_file.sql**

puis sur l'esclave en utilisant les valeurs lu sur le maître
mysql> CHANGE MASTER TO
-> MASTER_HOST='<master host name>',
-> MASTER_USER='<replication user name>',
-> MASTER_PASSWORD='<replication password>',
-> MASTER_LOG_FILE='<recorded log file name>',
-> MASTER_LOG_POS=<recorded log offset> ;
Lancer les thread esclaves
mysql > START SLAVE ;
l'esclave devrait alors se connecter au maître pour rattraper les modifications.

Une fois que l'esclave a initié la réplication, vous trouverez 2 fichiers : master.info et relay.info

Options pour les esclaves, normalement il y'en a pas besoin ils sont dans master.info

–master-host

–master-user

–master-password

–master-port

–master-connect-retry

–master-ssl

–master-ssl-ca

–master-ssl-capath

–master-ssl-cert

–master-ssl-cipher

–master-ssl-key

Ces commandes sont utilisés lors du lancement de l'esclave la première fois ou après un **RESET SLAVE** et arrêté puis relancé le serveur. On peut préférer spécifier ces options avec **CHANGE MASTER TO**.

Exemple de configuration

```
[mysqld]
server-id=2
master-host=db-master.mycompany.com
master-port=3306
master-user=pertinax
master-password=freitag
master-connect-retry=60
report-host=db-slave.mycompany.com
```

Options de contrôle de la réplication

–log-slave-updates Dit à l'esclave d'enregistrer les modifications effectuées par son thread SQL dans son propre log binaire. Par défaut, cette option est à Off. Pour que cette option ait un effet, l'esclave doit être lancé avec le log binaire activé.
–log-slave-updates sert lorsque vous voulez faire une chaîne de serveur de réplication. Par exemple : **A -> B -> C**. C'est-à-dire, A sert de maître à l'esclave B, et B sert de maître à l'esclave C. Pour que cela fonctionne, avec B qui sert d'esclave et de maître simultanément, vous devez lancer B avec l'option **–log-slave-updates**. A et B doivent être lancés avec le log binaire activé.

–log-warnings Fait que l'esclave affiche plus de message sur ses activités Cette option n'est pas limitée à la réplication.

–master-connect-retry=seconds Le nombre de secondes qu'un esclave attend avant de tenter de se re-connecter au maître. Par défaut, elle vaut 60.

–master-host=host Spécifie l'hôte ou l'IP du maître de réplication. Si cette option n'est pas fournie, le thread esclave ne sera pas lancé.

–master-info-file=file_name Le nom à utiliser pour le fichier dans lequel l'esclave stocke les informations sur le maître. Par défaut, c'est **mysql.info**

–master-password=password Le mot de passe que l'esclave utilise lors de l'identification auprès du maître

–master-port=port_number Le port du maître que l'esclave utilise lors de l'identification auprès du maître. Si le port n'est pas configuré, la valeur de la variable **MYSQL_PORT** est utilisée. Si vous n'y avez pas touché lors de la compilation avec configure, ce doit être 3306

–master-ssl

–master-ssl-ca=file_name

–master-ssl-capath=directory_name

–master-ssl-cert=file_name

–master-ssl-cipher=cipher_list

–master-ssl-key=file_name Ces options servent à configurer la réplication chiffrée, lorsque la connexion avec le maître utilise SSL. Leurs significations respectives est la même que les options **–ssl, –ssl-ca, –ssl-capath, –ssl-cert, –ssl-cipher, –ssl-key**

–master-user=username Le nom d'utilisateur que l'esclave utilise lors de l'identification auprès du maître. Le compte doit avoir les droits de **REPLICATION SLAVE**. Si l'utilisateur maître n'est pas configuré, l'utilisateur test est utilisé.

–max-relay-log-size=# Pour faire la rotation automatique des logs.

–read-only Cette option fait que le serveur n'autorise aucune modification, hormis celles du thread esclave, ou celle des utilisateurs ayant les droits de **SUPER**. Cela peut être utile si vous voulez vous assurer que l'esclave ne reçoit aucune modification des clients.

–relay-log=filename Pour spécifier la localisation et le nom qui doivent être utilisés pour les logs de relais. Les noms par défaut sont de la forme **host_name-relay-bin.nnn**, où host_name est le nom du serveur esclave et nnn indique le numéro de séquence du log de relais. Vous pouvez utiliser ces options pour avoir des noms de fichier de log de relais indépendants du nom d'hôte, ou si vos logs ont tendances à devenir très grands (et que vous ne voulez pas réduire la valeur de max_relay_log_size) et que vous devez les mettre dans un autre dossier, ou simplement pour accélérer la vitesse d'équilibrage entre deux disques.

–relay-log-index=filename Pour spécifier la localisation et le nom qui doivent être utilisés pour le fichier d'index du log de relais. Le nom par défaut est **host_name-relay-bin.index**, où host_name est le nom du serveur esclave.

–relay-log-info-file=filename Pour donner au fichier **relay-log.info** un autre nom ou pour le placer dans un autre dossier. Le nom par défaut est **relay-log.info** dans le dossier de données.

–relay-log-purge=0|1 Active ou désactive la vidange automatique des logs de relais, dès qu'ils ne sont plus utiles. C'est une variable globale, qui peut être dynamiquement modifiée avec **SET GLOBAL RELAY_LOG_PURGE=0|1**. Sa valeur par défaut est 1.

–relay-log-space-limit=# Limite la taille maximale de tous les fichiers de logs de relais sur l'esclave (une valeur de 0 signifie "sans limite"). Lorsque la limite est atteinte, le thread d'I/O fait une pause : il ne lit plus rien dans le log binaire du maître, jusqu'à ce que le thread SQL ait avancé, et effacé des fichiers de logs. Notez que cette limite n'est pas absolue. Avec **–relay-log-space-limit**, il ne faut pas utiliser de valeur inférieure à deux fois la taille de **–max-relay-log-size** (ou **–max-binlog-size** si **–max-relay-log-size** vaut 0) car dans ce cas, il y a des chances que le thread d'I/O attende de l'espace libre parce que **–relay-log-space-limit** est dépassée, mais que le thread SQL n'ait pas de logs à effacer, et ne peut donc libérer le thread d'I/O, forcant le thread d'I/O à ignorer temporairement **–relay-log-space-limit**.

–replicate-do-db=db_name Indique à l'esclave qu'il doit restreindre la réplication aux commandes qui utilisent la base de données **db_name** par défaut (c'est à dire celle qui est sélectionnée avec la commande **USE**). Pour spécifier plusieurs base de données, utilisez cette option aussi souvent que nécessaire. Noter que cela ne va pas autoriser les commandes multi-bases, comme **UPDATE some_db.some_table SET foo='bar'** si une base de données différente ou qu'aucune base de données n'est sélectionnée. Si vous avez besoin que les commandes multi-bases fonctionnent, utilisez **–replicate-wild-do-table=db_name.%**

Un exemple qui pourrait ne pas fonctionner comme vous l'attendez : si l'esclave est lancé avec –replicate-do-db=sales et que vous émettez une commande sur le maître, la commande UPDATE suivante ne sera pas répliquée :

USE prices ;

UPDATE sales.january SET amount=amount+1000 ;

Si vous avez besoin de répliquer des commandes multi-bases, utilisez l'option –replicate-wild-do-table=db_name.% à la place.

La raison principale de ce comportement "vérifie juste la base par défaut" est qu'il est difficile de savoir si une requête doit être répliquée, uniquement à partir de la requête. Par exemple, si vous utilisez une requête multi-tables DELETE ou multi-tables UPDATE, qui a des conséquences dans d'autres bases. La vérification de la base courante est aussi très rapide.

–replicate-do-table=db_name.table_name Dit à l'esclave qu'il doit restreindre la réplication à une table spécifiée. Pour spécifier plusieurs tables, il faut utiliser cette directive plusieurs fois, une fois par table. Cela fonctionnera pour les mises à jours multi-bases, au contraire de **–replicate-do-db**

–replicate-ignore-db=db_name Indique à l'esclave qu'il doit ne doit pas assurer la réplication avec les commandes qui utilisent la base de données **db_name** par défaut (c'est à dire celle qui est sélectionnée avec la commande USE). Pour spécifier plusieurs base de données, utilisez cette option aussi souvent que nécessaire. Note que cela ne va pas autoriser les commandes multi-bases, comme **UPDATE some_db.some_table SET foo='bar'** si une base de données différente ou qu'aucune base de données n'est sélectionnée. Si vous avez besoin que les commandes multi-bases fonctionnent, assurez vous que vous avez MySQL 3.23.28 ou plus récent, et utilisez **–replicate-wild-do-table=db_name.%**

Un exemple qui pourrait ne pas fonctionner comme vous l'attendez : si l'esclave est lancé avec –replicate-ignore-db=sales et que vous émettez une commande sur le maître, la commande UPDATE suivante ne sera pas répliquée

USE prices ;

UPDATE sales.january SET amount=amount+1000 ;

Si vous avez besoin de répliquer des commandes multi-bases, utilisez l'option –replicate-wild-ignore-table=db_name.% à la place

–replicate-ignore-table=db_name.table_name Dit à l'esclave qu'il ne doit pas répliquer les commandes qui touchent à la table spécifiée, même si d'autres tables sont modifiées dans la même commande. Pour spécifier plusieurs tables, il faut utiliser cette directive plusieurs fois, une fois par table. Cela fonctionnera pour les mises à jours multi-bases, au contraire de **–replicate-ignore-db**

Notes sur cette liste d'options

–replicate-wild-do-table=db_name.table_name Dit à l'esclave qu'il doit restreindre la réplication aux tables dont le nom vérifie le masque spécifié. Le masque peut contenir les caractères '%' et '_', qui ont la même signification que dans les expressions régulières de la clause LIKE. Pour spécifier plusieurs tables, il faut utiliser cette directive plusieurs fois, une fois par table. Cela fonctionnera pour les mises à jours multi-bases, au contraire de **–replicate-do-db**

Exemple :
–replicate-wild-do-table=foo%.bar%
va répliquer les mises à jour qui surviennent sur toutes les tables de toutes les bases qui commencent par foo, et dont le nom de table commence par bar.
Notez que si vous utilisez **–replicate-wild-do-table=foo%.%**, alors la règle sera propagée à **CREATE DATABASE** et **DROP DATABASE**, c'est à dire que ces deux commandes seront répliquées si le nom de la base correspond au masque (foo% ici) (la magie est ici déclenchée par % comme masque de table.).

Si le masque de noms de tables est **%**, il accepte tous les noms de tables et les options s'appliquent aux commandes de niveau base de données (comme **CREATE DATABASE**, **DROP DATABASE** et **ALTER DATABASE**). Par exemple, si vous utilisez **–replicate-wild-do-table=foo%.%**, les commandes de niveau de base de données seront répliquées si le nom de la base de données est accepté par le masque foo%.

Si vous voulez faire la réplication des tables du type **ma_petite%base** (ceci est le nom exact de la base), mais que vous ne voulez pas répliquer la base **ma1petiteAABCbase**, vous devez protéger les caractères '_' et '**%**' : il faut utiliser une syntaxe équivalent à : **replicate-wild-do-table=my_own\%db**. Et si vous spécifiez cette option en ligne de commande, suivant votre système, vous devrez protéger aussi le caractère **** (par exemple, en Shell bash, vous devez émettre une option sous la forme **–replicate-wild-do-table=my_own\\%db**).

–replicate-wild-ignore-table=db_name.table_name Dit à l'esclave qu'il ne doit pas répliquer les tables dont le nom vérifie le masque spécifié. Pour spécifier plusieurs tables, il faut utiliser cette directive plusieurs fois, une fois par table. Cela fonctionnera pour les mises à jours multi-bases, au contraire de **–replicate-do-db**.

Exemple
–replicate-wild-ignore-table=foo%.bar%
n'autorisera pas de modifications dans les tables des bases dont le nom commence par foo et dont le nom de table commence par bar. Pour des informations sur le fonctionnement du filtre, voyez l'option –replicate-wild-ignore-table. La règle pour inclure des caractères littéraux est la même que pour –replicate-wild-ignore-table.

–replicate-rewrite-db=from_name->to_name Dit à l'esclave de remplacer la base courante (celle qui est sélectionnée avec USE) par **to_name** si elle était **from_name** sur le maître. Seules les commandes impliquant des tables peuvent être affectées. (**CREATE DATABASE, DROP DATABASE** ne le seront pas), et uniquement si from_name était la base de données courante sur le maître. Cela ne fonctionnera pas pour les commandes multi-bases de données. Notez que la traduction est faite avant que les règles **–replicate-*** ne soient testées.

Si vous utilisez cette option en ligne de commande, et que vous utilisez le caractère '**>**', qui peut être spécial pour votre interpréteur Shell, protégez-le comme ceci :
shell> mysqld –replicate-rewrite-db="olddb->newdb"

–replicate-same-server-id À utiliser sur les serveurs esclaves. Généralement, vous pouvez spécifier la valeur 0 pour éviter les réplications infinies. Si cette option vaut 1, l'esclave n'ignorera pas les événements de réplication, même s'ils portent son propre numéro d'identification. Normalement, cela n'est utile que pour de très rares configurations. Vous ne pouvez pas mettre cette option à 1 si **–log-slave-updates** est utilisé.

–report-host=host Le nom d'hôte ou l'adresse IP de l'esclave, qui doit être indiquée lors de l'enregistrement de l'esclave chez le maître. Cela apparaîtra dans l'affichage de la commande **SHOW SLAVE HOSTS**. Laissez cette option vide pour que l'esclave ne s'enregistre pas sur le maître. Notez qu'il n'est pas suffisant pour que le maître lise l'adresse IP de l'esclave sur le socket, une fois que l'esclave se connecte. À cause du NAT et des problèmes de routages, cette IP peut être invalide pour se connecter au maître depuis l'hôte ou les autres esclaves.

–report-port=port_number Le port de connexion indiqué par l'esclave lors de son enregistrement chez le maître. Configurez cette option si l'esclave utilise un port autre que le port par défaut, ou si vous avez installé un tunnel spécial pour le maître ou les autres esclaves. Dans le doute, laissez cette option vide.

–skip-slave-start Dit à l'esclave de ne pas lancer les threads esclaves au démarrage du serveur. L'utilisateur pourra les lancer manuellement, avec **START SLAVE**.

–slave_compressed_protocol=# Si cette option vaut 1, alors le protocole client/serveur compressé sera utilisé, si l'esclave et le maître le supportent.

–slave-load-tmpdir=filename Cette option vaut par défaut la variable **tmpdir**. Lorsque le thread SQL réplique des commandes **LOAD DATA INFILE**, il extrait les fichiers à charger du log de relais dans un fichier temporaire, puis charge ce fichier dans la table. Si le fichier chargé sur le maître est immense, le fichier temporaire sera aussi grand. Il faudra donc dire à l'esclave de placer ces fichiers temporaires sur un grand disque, qui sera différent de **tmpdir** : utilisez cette option. Dans ce cas, vous pouvez aussi utiliser l'option **–relay-log**, car les fichiers de log de relais seront aussi grands. **–slave-load-tmpdir** doit pointer sur un système de fichier basés sur un disque, et non pas sur une portion de mémoire : l'esclave doit pouvoir accéder à ce fichier pour répliquer la commande **LOAD DATA INFILE**, même après un redémarrage.

–slave-net-timeout=# Le nombre de secondes à attendre des données du maître, avant d'annuler la lecture en considérant que la connexion est rompue, et de tenter de se reconnecter. La première reconnexion intervient immédiatement après l'expiration du délai. L'intervalle entre deux tentatives de connexion est contrôlé par l'option **–master-connect-retry**.

–slave-skip-errors= [err_code1,err_code2,... | all] Normalement, la réplication s'arrête lorsqu'une erreur survient, ce qui vous donne l'opportunité de résoudre les incohérences manuellement. Cette option Indique au thread SQL les erreurs qu'il doit ignorer durant la réplication. N'utilisez pas cette option si vous ne connaissez pas la raison des erreurs que vous rencontrez. S'il n'y a pas de bugs dans votre réplication, et qu'il n'y a pas de bug dans MySQL, vous ne devriez pas rencontrer d'erreurs, ni utiliser cette option. L'utilisation abusive de cette option conduit irrémédiablement l'esclave à être désynchronisé avec le maître sans que vous ne sachiez d'où vient l'erreur.

Pour les codes d'erreur, il faut utiliser les numéros d'erreurs fournis par l'esclave dans le log d'erreur, et dans le résultat de **SHOW SLAVE STATUS**. La liste complète des messages d'erreurs est disponible dans la distribution source, dans le fichier **Docs/mysqld_error.txt**.

Exemple de réplication

Mysql permet de multiples configurations de réplication.

Sur le maître

Dans **/etc/mysql/my.cnf** ajouter :

log-bin les log-bin doivent être activés, c'est sur ces log que se base la réplication

id-server=1 ce numéro doit être unique à chaque serveur

on crée l'utilisateur pour la réplication
GRANT REPLICATION SLAVE ON generator.php TO replication@'server-B' identified by 'password'
puis on lock les tables
FLUSH TABLES WITH READ LOCK ;
on note le fichier de log et sa position ; attention la valeur n'est valide que si log-bin est déjà activé, sinon utiliser les valeur par défaut
SHOW MASTER STATUS ;
on copie les tables que l'on souhaite répliquer pour les envoyer sur l'esclave. et on enlève le verrou
UNLOCK TABLES ;
on peut relancer le serveur mysql
/etc/init.d/mysql restart
Sur le serveur esclave :

dans **/etc/mysql/my.cnf**, dans la section **[mysqld]** ajouter :

log-bin

id-server=2

skip-slave-start cette options sert pour éviter de prendre en compte les options de réplication puisque l'on va la lancer en ligne.

#master-host=server-B on laisse ces lignes commentée pour le premier démarrage.

#master-port=3306

#master-user=replication

#master-password=password

#master-connect-retry=60

#report-host=db-slave.uubu

#log-slave-updates n'est utile qu'en cas d'un modèle de réplication A > B > C. B joue alors le rôle de relay

on crée l'utilisateur pour la réplication
GRANT REPLICATION SLAVE ON generator.php TO replication@'server-A' identified by 'password'
on stop le serveur mysql
/etc/init.d/mysql stop
on copie les bases du master, bien vérifier les droits ! On relance le serveur mysql
/etc/init.d/mysql start
on lance la réplication
mysql> CHANGE MASTER TO
-> MASTER_HOST='server-A',
-> MASTER_USER='replication',
-> MASTER_PASSWORD='password',
le fichier de log qu'on a noté via SHOW MASTER STATUS
-> MASTER_LOG_FILE='<recorded log file name>',
idem pour la position
-> MASTER_LOG_POS=<recorded log offset>;
puis lancement en tant qu'esclave
START SLAVE
ouvrir de nouveau /etc/mysql/my.cnf
on peut supprimer **skip-slave-start** puisque la réplication est en place, et décommenter les lignes qu'on avait ajouté en commentaire.
Relancer le serveur
/etc/init.d/mysql restart

Explication de log-slave-updates

Le serveur **B** se base sur le **log-bin** du serveur **A** pour répliquer les données. Par défaut, le serveur **B** ne log pas dans ses **log-bin** les données de réplication. Dans le cas où on souhaiterai ajouter un serveur **C** qui se réplique sur **B**, il est nécessaire d'ajouter cette options, afin que les données répliquées soient bien écrite dans les **log-bin**.

Autres exemple de réplication

Le système est très souple, ici j'explique une réplication de type **A > B +** le principe pour une réplication en chaîne de type **A > B > C**.

Il est bien sûr possible que **B** et **C** se répliquent tous 2 sur **A** directement.

Il est même possible de créer une réplication croisée de type **A<>B**. La manip n'est pas très différente, mais un peu plus délicate.

Dans un premier temps il faut suivre l'exemple ci-dessus pour créer une réplication de **A** vers **B**. Ensuite reprendre l'exemple, locker les tables sur **B**, noter le **log-bin** et sa position, puis suivre l'exemple de **B** ci-dessus, et l'appliquer sur **A** : renseigner **/etc/mysql/my.cnf**, relancer le serveur, puis lancer la réplication.