

Mise en place d'une réplication MariaDB Master ↔ Master

✨ Introduction

Cette procédure vise à configurer une **réplication bidirectionnelle** (master ↔ master) entre deux serveurs MariaDB. L'objectif est que **chacun des deux serveurs réplique automatiquement les modifications** apportées sur l'autre. Cette solution permet de :

- Maintenir une **haute disponibilité**
- Répartir les **charges de lecture/écriture**
- Assurer une **redondance en cas de panne**

Architecture utilisée

Rôle	IP	Nom
Serveur MariaDB 1	192.168.1.20	db-master1
Serveur MariaDB 2	192.168.1.25	db-master2

🔧 Étapes de configuration

🔧 Etape 1 – Configuration de **db-master1**

1. Modifier la configuration MariaDB

Fichier à modifier : `/etc/mysql/mariadb.conf.d/50-server.cnf`

Rechercher et adapter ou ajouter les lignes suivantes :

```
server-id = 1
log_bin = /var/log/mysql/mysql-bin
bind-address = 0.0.0.0
```

- `server-id` : identifiant unique du serveur pour la réplication.
- `log_bin` : chemin du fichier de journal binaire (nécessaire à la réplication).
- `bind-address` : permet d'écouter sur toutes les interfaces réseau (pour accepter les connexions distantes).

2. Redémarrer MariaDB

```
sudo systemctl restart mariadb
```

Cela applique les modifications de configuration.

3. Se connecter à la base de données MariaDB

```
mysql -u root -p
```

- `u root` : utilisateur MariaDB avec les privilèges d'administration.
- `p` : invite à entrer le mot de passe de l'utilisateur root.

4. Créer un utilisateur de réplication

```
CREATE USER 'replicator'@'192.168.1.25' IDENTIFIED BY 'replicaPass!';
GRANT REPLICATION SLAVE ON *.* TO 'replicator'@'192.168.1.25';
FLUSH PRIVILEGES;
```

- `CREATE USER` : crée l'utilisateur dédié à la réplication.
- `GRANT REPLICATION SLAVE` : autorise cet utilisateur à lire les journaux binaires.
- `FLUSH PRIVILEGES` : recharge les permissions immédiatement.

5. Obtenir les informations du master

```
SHOW MASTER STATUS;
```

Ce qui retourne :

- `File` : nom du fichier binaire courant (ex : `mysql-bin.000001`)
- `Position` : position dans le fichier à partir de laquelle commencer la réplication (ex : `1234`)

Notez soigneusement ces deux valeurs, elles seront utilisées dans la configuration du serveur secondaire.

Etape 2 – Configuration de **db-master2**

1. Modifier le fichier de configuration

Fichier : `/etc/mysql/mariadb.conf.d/50-server.cnf`

```
server-id = 2
log_bin = /var/log/mysql/mysql-bin
bind-address = 0.0.0.0
```

 Le server-id doit être différent de celui de db-master1.

2. Redémarrer MariaDB

```
sudo systemctl restart mariadb
```

3. Connexion à la base MariaDB

```
mysql -u root -p
```

4. Créer l'utilisateur de réplication pour db-master1

```
CREATE USER 'replicator'@'192.168.1.20' IDENTIFIED BY 'replicaPass!';
GRANT REPLICATION SLAVE ON *.* TO 'replicator'@'192.168.1.20';
FLUSH PRIVILEGES;
```

5. Configurer db-master2 en tant qu'esclave de db-master1

 À ce stade, vous devez remplacer les valeurs MASTER_LOG_FILE et MASTER_LOG_POS par les informations récupérées via la commande SHOW MASTER

STATUS; sur le serveur db-master1. Ces valeurs correspondent :

- à **File** : le nom du fichier binaire actuellement utilisé par le master
- à **Position** : la position exacte de lecture à partir de laquelle commencer la réplication

Exemple : si **SHOW MASTER STATUS;** retourne **mysql-bin.000001** et **Position: 1234** , vous devrez utiliser :

```
MASTER_LOG_FILE='mysql-bin.000001',  
MASTER_LOG_POS=1234;
```

❗ Si ces informations sont incorrectes ou désynchronisées, la réplication échouera avec des erreurs binlog.

```
STOP SLAVE;
```

```
CHANGE MASTER TO  
MASTER_HOST='192.168.1.20',  
MASTER_USER='replicator',  
MASTER_PASSWORD='replicaPass!',  
MASTER_LOG_FILE='mysql-bin.000001',  
MASTER_LOG_POS=1234;
```

```
START SLAVE;
```

- **STOP SLAVE** : arrête temporairement la réplication si active
- **CHANGE MASTER TO** : configure le serveur master distant
- **START SLAVE** : démarre la réplication

6. Vérifier le statut de la réplication

```
SHOW SLAVE STATUS\G
```

Les lignes suivantes doivent indiquer :

- `Slave_IO_Running: Yes`
- `Slave_SQL_Running: Yes`

Etape 3 – Activer la réplication dans l'autre sens

1. Sur `db-master2` , récupérer l'état master

```
SHOW MASTER STATUS;
```

Notez le `File` et `Position` retournés.

2. Sur `db-master1` , configurer l'esclavage inverse

 À ce stade, vous devez remplacer les valeurs `MASTER_LOG_FILE` et `MASTER_LOG_POS` par les informations récupérées via la commande `SHOW MASTER STATUS;` sur le serveur `db-master2`. Ces valeurs correspondent :

- à `File` : le nom du fichier binaire actuellement utilisé par le master
- à `Position` : la position exacte de lecture à partir de laquelle commencer la réplication

Exemple : si `SHOW MASTER STATUS;` retourne `mysql-bin.000003` et `Position: 742` , vous devrez remplacer :

```
MASTER_LOG_FILE='mysql-bin.000003',  
MASTER_LOG_POS=742;
```

! Si ces informations sont incorrectes ou désynchronisées, la réplication échouera avec des erreurs binlog. À ce stade, vous devez remplacer les valeurs `MASTER_LOG_FILE` et `MASTER_LOG_POS` par celles obtenues lors du `SHOW MASTER STATUS` exécuté sur `db-master2`. Ces paramètres correspondent au nom du fichier binaire courant et à la position exacte à partir de laquelle la réplication doit reprendre sur le serveur secondaire. Une mauvaise valeur ici empêchera la synchronisation ou pourra provoquer des erreurs de lecture binlog.

```
STOP SLAVE;

CHANGE MASTER TO
  MASTER_HOST='192.168.1.25',
  MASTER_USER='replicator',
  MASTER_PASSWORD='replicaPass!',
  MASTER_LOG_FILE='mysql-bin.000002',
  MASTER_LOG_POS=456;

START SLAVE;
```

3. Vérifier le statut

```
SHOW SLAVE STATUS\G
```

Assurez-vous d'avoir `Slave_IO_Running: Yes` et `Slave_SQL_Running: Yes` sur **les deux serveurs**.

Résultat attendu

Serveur	Rôle	Réplication
db-master1	Master & Slave	IO: Yes / SQL: Yes 

db-master2	Master & Slave	IO: Yes / SQL: Yes 
------------	----------------	--

Test de validation

1. Sur **db-master1** :

```
USE testdb;  
INSERT INTO users (name) VALUES ('test_master1');
```

2. Sur **db-master2** :

```
USE testdb;  
SELECT * FROM users WHERE name = 'test_master1';
```

 Si les données apparaissent, la réplication fonctionne 