Documentation : Installation et Configuration d'un Cluster MySQL avec Docker

Ecrit par Mohamed Trabelsi

1️⃣ Introduction

Cette documentation décrit l'installation et la configuration d'un **cluster MySQL** avec un nœud primaire et deux réplicas, ainsi que l'intégration avec **Joget** via MySQL Router. L'architecture comprend :

- **VM1** : MySQL primaire
- **VM2** : Réplica 1
- **VM3** : MySQL Router
- **VM4** : Réplica 2

2️⃣ Configuration des VMs

Ajoutez ces entrées dans `/etc/hosts` sur chaque VM :

```
192.168.75.111 mysql-replica1
192.168.75.152 mysql-primary
192.168.75.199 mysql-router
192.168.75.203 mysql-replica2
```

3️⃣ Déploiement des Conteneurs MySQL avec Docker Compose

📌 Docker Compose pour MySQL Primaire (VM1)

Créez un fichier `docker-compose.yml` :

```
version: '3.8'
#ubuntu 3
services:
  # MySQL Primary Node
  mysql-primary:
    image: mysql/mysql-server:8.0
    container_name: mysql-primary
    hostname: mysql-primary
    network_mode: "host"
    environment:
      MYSQL_ROOT_PASSWORD: fb
      MYSQL_DATABASE: jwdb
      MYSQL_USER: fb
      MYSQL_PASSWORD: fb
    volumes:
      - mysql-primary-data:/var/lib/mysql
    ports:
      - "3306:3306"
      - "33061:33061"
    command: >
      --server-id=1
      --log-bin=mysql-bin
      --gtid-mode=ON
      --enforce-gtid-consistency=ON
      --binlog-format=ROW
      --log-slave-updates=ON
      --transaction-write-set-extraction=XXHASH64
      --innodb-flush-log-at-trx-commit=1
      --innodb-flush-method=O_DIRECT
      --innodb-file-per-table=1
      --innodb-buffer-pool-size=1G
      --innodb-log-file-size=500M
      --innodb-log-buffer-size=64M
      --innodb-flush-log-at-timeout=1
      --innodb-monitor-enable=all
```

```
      --innodb-print-all-deadlocks=1
      --innodb-status-file=1
      --innodb-buffer-pool-dump-at-shutdown=1
      --innodb-buffer-pool-load-at-startup=1
      --innodb-buffer-pool-filename=ib_buffer_pool
      --innodb-buffer-pool-dump-pct=25
      --innodb-buffer-pool-chunk-size=128M
      --innodb-buffer-pool-instances=8
      --innodb-log-files-in-group=2
      --innodb-log-file-size=500M
      --innodb-log-buffer-size=64M
      --innodb-flush-log-at-timeout=1
      --innodb-monitor-enable=all
      --innodb-print-all-deadlocks=1
      --innodb-status-file=1
      --innodb-buffer-pool-dump-at-shutdown=1
      --innodb-buffer-pool-load-at-startup=1
      --innodb-buffer-pool-filename=ib_buffer_pool
      --innodb-buffer-pool-dump-pct=25
      --innodb-buffer-pool-chunk-size=128M
      --innodb-buffer-pool-instances=8
      --innodb-log-files-in-group=2

# Docker Volumes
volumes:
  mysql-primary-data:
```

📌 Docker Compose pour Réplica 1 (VM2)

```
version: '3.8'

services:
  mysql-replica1:
    image: mysql/mysql-server:8.0
    container_name: mysql-replica1
```

```
    hostname: mysql-replica1
    network_mode: "host"
    environment:
      MYSQL_ROOT_PASSWORD: fb
      MYSQL_DATABASE: jwdb
      MYSQL_USER: fb
      MYSQL_PASSWORD: fb
    volumes:
      - mysql-replica1-data:/var/lib/mysql
    ports:
      - "3306:3306"
      - "33061:33061"
    command: >
      --server-id=2
      --log-bin=mysql-bin
      --gtid-mode=ON
      --enforce-gtid-consistency=ON
      --binlog-format=ROW
      --log-slave-updates=ON
      --transaction-write-set-extraction=XXHASH64
      --innodb-flush-log-at-trx-commit=1
      --innodb-flush-method=O_DIRECT
      --innodb-file-per-table=1
      --innodb-buffer-pool-size=1G
      --innodb-log-file-size=500M
      --innodb-log-buffer-size=64M
      --innodb-flush-log-at-timeout=1
      --innodb-monitor-enable=all
      --innodb-print-all-deadlocks=1
      --innodb-status-file=1
      --innodb-buffer-pool-dump-at-shutdown=1
      --innodb-buffer-pool-load-at-startup=1
      --innodb-buffer-pool-filename=ib_buffer_pool
      --innodb-buffer-pool-dump-pct=25
      --innodb-buffer-pool-chunk-size=128M
      --innodb-buffer-pool-instances=8
```

```
    --innodb-log-files-in-group=2
# Docker Volumes
volumes:
  mysql-replica1-data:
```

📌 Docker Compose pour Réplica 2 (VM3 - Container Docker)

```
version: '3.8'

services:
  mysql-replica2:
    image: mysql/mysql-server:8.0
    container_name: mysql-replica2
    hostname: mysql-replica2
    network_mode: "host"
    environment:
      MYSQL_ROOT_PASSWORD: fb
      MYSQL_DATABASE: jwdb
      MYSQL_USER: fb
      MYSQL_PASSWORD: fb
    volumes:
      - mysql-replica2-data:/var/lib/mysql
    ports:
      - "3306:3306"
      - "33061:33061"
    command: >
      --server-id=3
      --log-bin=mysql-bin
      --gtid-mode=ON
      --enforce-gtid-consistency=ON
      --binlog-format=ROW
      --log-slave-updates=ON
      --transaction-write-set-extraction=XXHASH64
      --innodb-flush-log-at-trx-commit=1
      --innodb-flush-method=O_DIRECT
```

```
      --innodb-file-per-table=1
      --innodb-buffer-pool-size=1G
      --innodb-log-file-size=500M
      --innodb-log-buffer-size=64M
      --innodb-flush-log-at-timeout=1
      --innodb-monitor-enable=all
      --innodb-print-all-deadlocks=1
      --innodb-status-file=1
      --innodb-buffer-pool-dump-at-shutdown=1
      --innodb-buffer-pool-load-at-startup=1
      --innodb-buffer-pool-filename=ib_buffer_pool
      --innodb-buffer-pool-dump-pct=25
      --innodb-buffer-pool-chunk-size=128M
      --innodb-buffer-pool-instances=8
      --innodb-log-files-in-group=2
# Docker Volumes
volumes:
  mysql-replica2-data:
```

📌 Docker Compose pour MySQL Router (VM3)

```
version: '3'
services:
  mysql-router:
    image: mysql/mysql-router
    container_name: mysql-router
    network_mode: host
    environment:
      MYSQL_HOST: mysql-primary
      MYSQL_PORT: "3306"
      MYSQL_USER: admin
      MYSQL_PASSWORD: admin
      MYSQL_INNODB_CLUSTER_MEMBERS: "3"
    restart: always
```

4️⃣ Déploiement et Configuration

1️⃣ Démarrer les services

Exécutez la commande suivante sur chaque VM :

```
docker-compose up -d
```

2️⃣ Créer l'utilisateur admin dans chaque conteneur

```
docker exec -it mysql-primary mysql -uroot -pfb -e "CREATE USER 'admi
n'@'%' IDENTIFIED BY 'admin'; GRANT ALL PRIVILEGES ON *.* TO 'admi
n'@'%' WITH GRANT OPTION; FLUSH PRIVILEGES; RESET MASTER;"
```

Faites de même pour `mysql-replica1` et `mysql-replica2`.

3️⃣ Configurer les instances pour le cluster

Dans chaque conteneur, exécutez :

```
docker exec -it mysql-primary mysqlsh -uroot -pfb -e "dba.configureInstance
('admin@mysql-primary:3306')"
docker exec -it mysql-replica1 mysqlsh -uroot -pfb -e "dba.configureInstance
('admin@mysql-replica1:3306')"
docker exec -it mysql-replica2 mysqlsh -uroot -pfb -e "dba.configureInstance
('admin@mysql-replica2:3306')"
```

4️⃣ Créer le cluster depuis MySQL Primary

```
docker exec -it mysql-primary mysqlsh -uroot -pfb
```

Puis, dans MySQL Shell :

```
var cls = dba.createCluster("myCluster");
cls.addInstance("admin@mysql-replica1:3306");
```

```
cls.addInstance("admin@mysql-replica2:3306");
cls.status();
```

5️⃣ Vérifier le cluster

```
docker exec -it mysql-primary mysqlsh -uroot -pfb -e "var cls = dba.getClust
er(); cls.status();"
```

5️⃣ Intégration avec Joget

📌 Configuration de la base de données dans Joget

Ajoutez ces paramètres dans `wflow-datasource.properties` :

```
workflowDriver=com.mysql.cj.jdbc.Driver
workflowUrl=jdbc:mysql://192.168.75.199:6446/jwdb?characterEncoding=UTF
-8&useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC
workflowUser=admin
workflowPassword=admin
```

Vérifiez que **Joget** peut se connecter à MySQL Router et au cluster.

🎯 Remarques Importantes

1. *Les identifiants de la base de données doivent être stockés dans un ** `.env` pour plus de sécurité.
2. **L'initialisation de la base de données (`init.sql`) doit être intégrée dans Docker Compose** pour automatiser la création des tables et utilisateurs.
3. **Joget doit se connecter au Router et non aux nœuds directement**, afin d'assurer la haute disponibilité.

✅ Conclusion

Cette configuration permet de déployer un **cluster MySQL hautement disponible** avec **Joget** via **MySQL Router**. Vérifiez que les conteneurs fonctionnent

correctement avec `docker ps` et que Joget peut établir la connexion.

🚀 **Cluster MySQL prêt à l'emploi !**