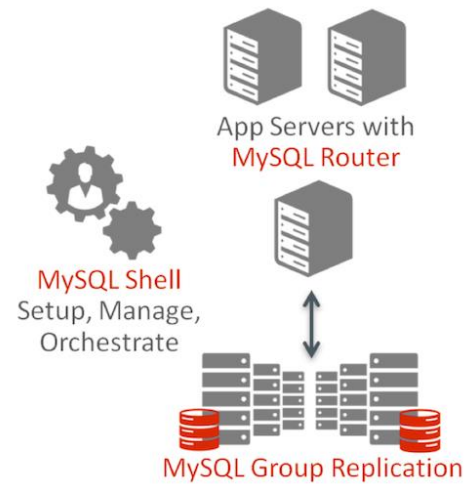


MySQL InnoDB Cluster Step by Step



Chaoyang Han

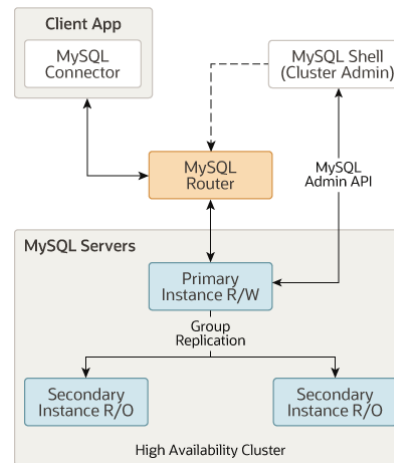
目录

MySQL InnoDB Cluster.....	1
MySQL Group Replication.....	2
通过 MySQL Shell 创建 MGR.....	3
环境信息.....	3
安装 MySQL 软件.....	5
初始化 MySQL.....	6
创建管理用户.....	7
装载样本数据.....	9
创建 MGR 集群.....	9
MGR 主备切换.....	26
配置和使用 MySQL Router.....	29
初始化 MySQL Router.....	30
MySQL Router 配置文件.....	31
启动 MySQL Router.....	34
测试读写分离.....	35
MGR 高可用架构.....	38
双中心架构.....	38
多中心架构.....	40
在单机上手动创建 MGR.....	55
创建用户和目录.....	55
临时 my.cnf 参数.....	55
初始化 MySQL.....	57
合并 my.cnf.....	59

启动多实例.....	62
创建复制用户.....	63
设置复制通道.....	68
设置引导节点.....	70
加入其它节点.....	71
创建测试数据.....	76

MySQL InnoDB Cluster

MySQL InnoDB Cluster 是 MySQL 官方提供的一个完全高可用的数据库解决方案，提供故障转移、故障恢复、读写分离、负载均衡等功能，主要由三个模块构成：



MySQL Server (版本 $\geq 5.7.17$): 支持 Group Replication 功能的，简称 MGR。模块主要功能在于实现了组内通信，故障转移 (Failover，即当活动的服务或应用意外终止时，快速启用冗余或备用的服务器、系统、硬件或者网络接替它们工作)、故障恢复 (Failback，即将系统，组件，服务恢复到故障之前的状态)

MySQL Shell: 可编程的高级客户端，支持标准 SQL 语法、JavaScript 语法、Python 语法，以及 API 接口，可以更方便的管理和使用 MySQL 服务器。实现快速部署，主要提供了一套 AdminAPI，可以自动化配置 Group Replication。

MySQL Router: 轻量级中间件，支持透明路由规则。内置读写分离，负载均衡，可自动根据 MySQL InnoDB Cluster 中的 metadata 自动调整。

MySQL Group Replication

MySQL Group Replication 功能是在 2016 年 MySQL-5.7.17 这个版本上引入的，MGR 基于分布式 paxos 协议，实现组复制，保证数据一致性。内置故障检测和自动选主功能，它实现了 MySQL 各个结点间数据强一致性。但是在 5.7.20 版本之前，MySQL Group Replication 还不完善，存在各种问题，所以搭建 MySQL Group Replication 集群建议采用 5.7.20 之后的版本，最好采用 8.0 版本。本文档采用 8.0.18 来搭建 MGR 集群环境，且采用 MGR 的单主模式。

- 高一致性，基于原生复制及 paxos 协议的组复制技术，并以插件的方式提供，提供一致数据安全保证
- 高容错性，只要不是大多数节点坏掉就可以继续工作，有自动检测机制，当不同节点产生资源争用冲突时，不会出现错误，按照先到者优先原则进行处理，并且内置了自动化脑裂防护机制
- 高扩展性，节点的新增和移除都是自动的，新节点加入后，会自动从其他节点上同步状态，直到新节点和其他节点保持一致，如果某节点被移除了，其他节点自动更新组信息，自动维护新的组信息
- 高灵活性，有单主模式和多主模式，单主模式下，会自动选主，所有更新操作都在主上进行；多主模式下，所有 server 都可以同时处理更新操作

虽然有许多优点，如果要使用 MGR 功能的话，也是有一些限制的：

- 仅用于 InnoDB 存储引擎（事务和行级锁）
- 表必须有主键
- 必须启用 GTID
- 必须开启二进制日志，并且其格式必须为 ROW (binlog_format=row)
- 冲突 DDI、DML 只能在同一成员上执行成功
- 在多主结构中，不完全支持外键
- 不支持 serializable 的事务隔离级别
- 只支持 IPv4，并且需要低延迟，高带宽的网络环境
- MGR 最大支持 9 个成员

- 复制信息元数据必须存在于系统表（master-info-repository=TABLE、relay-log-info-repository=TABLE）
- 不支持复制过滤
- 不支持 savepoint 的使用
- 不支持超大事務

通过 MySQL Shell 创建 MGR

环境信息

我们有三台服务器，用于构建 MGR 集群。三台也是最低的要求。我们在这三台上部署 MySQL 数据库软件和 MySQL Shell 软件。

```
[root@pmydb01 ~]# cat /etc/hosts
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1        localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.10.231 pmydb01
192.168.10.232 pmydb02
192.168.10.233 pmydb03
[root@pmydb01 ~]#
```

	pmydb01	pmydb02	pmydb03
server_id	3306	3307	3308
loose-group_replication_local_address	192.168.10.231:33306	192.168.10.232:33306	192.168.10.233:33306

下面是 my.cnf 的共同部分，除了 server_id 和 loose-group_replication_local_address 不一样，别的都一样。对于每个实例，都需要再加上这两个参数。

```
[mysqld]
basedir=/pgold/mysql/product/8026
log_timestamps=system
```

```
user = mysql
log_error_verbosity = 3
innodb_log_file_size = 500M
innodb_log_files_in_group = 5
binlog-format=row
log_slave_updates=on
binlog_checksum=crc32
character_set_server=utf8mb4

port=3306
socket=/pgold/data/mysql.sock
datadir=/pgold/data
log_bin=/pgold/log/binlog/binlog
relay_log=/pgold/log/relaylog/relaylog
log-error=/pgold/data/error.log

master-info-repository=table
relay-log-info-repository=table
gtid-mode=on
enforce-gtid-consistency=true
binlog_transaction_dependency_tracking=writeset
transaction_write_set_extraction=xxhash64
slave_parallel_type=logical_clock
slave_parallel_workers=128
sql_require_primary_key=1
slave_preserve_commit_order=1
```

```
slave_checkpoint_period=2

loose-plugin_load_add='mysql_clone.so'
loose-plugin_load_add='group_replication.so'
loose-group_replication_group_name="aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa"
loose-group_replication_start_on_boot=off
loose-group_replication_bootstrap_group=off
loose-group_replication_exit_state_action=read_only
loose-group_replication_flow_control_mode = "disabled"
loose-group_replication_single_primary_mode=on
loose-group_replication_group_seeds= "192.168.10.231:33306,192.168.10.232:33306,192.168.10.233:33306"
```

插件相关的参数只能在插件加载之后设置，如果想启动时设置这些参数并且启动后加载插件到配置文件中，可以在参数前加上“loose-”前缀。比如上面组复制的一些参数

安装 MySQL 软件

所需要的软件：

```
mysql-8.0.26-el7-x86_64.tar.gz
mysql-shell-8.0.26-1.el7.x86_64.rpm
```

在所有的数据库实例上，安装 MySQL 数据库和 MySQL Shell 软件

```
rpm -ivh mysql-shell-8.0.26-1.el7.x86_64.rpm
mkdir -p /pgold/mysql/product;cd /pgold;mkdir -p data log/{binlog,relaylog}
```



```
groupadd -g 27 -o -r mysql
useradd -M -N -g mysql -o -r -d /pgold/mysql -s /bin/false -c "MySQL Server" -u 27 mysql
```

```
cd /pgold/mysql/product
tar zxvf /tmp/mysql-8.0.26-el7-x86_64.tar.gz
mv mysql-8.0.26-el7-x86_64 8026
chown -R mysql:mysql /pgold
```

初始化 MySQL

```
/pgold/mysql/product/8026/bin/mysqld --defaults-file=/etc/my.cnf --initialize --user=mysql
```

```
[root@pmydb01 ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/etc/my.cnf --initialize --user=mysql
[root@pmydb01 ~]# grep -i password /pgold/data/error.log
2021-10-08T07:20:02.968448+08:00 0 [Note] [MY-010309] [Server] Auto generated RSA key files through --sha256_password_auto_generate_rsa_keys are placed in data directory.
2021-10-08T07:20:02.971023+08:00 0 [Note] [MY-010308] [Server] Skipping generation of RSA key pair through --caching_sha2_password_auto_generate_rsa_keys as key files are present in data directory.
2021-10-08T07:20:02.987381+08:00 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: M_Xpi6+Nl&52
```

```
[root@pmydb02 ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/etc/my.cnf --initialize --user=mysql
[root@pmydb02 ~]# grep -i password /pgold/data/error.log
2021-10-08T07:20:14.556507+08:00 0 [Note] [MY-010309] [Server] Auto generated RSA key files through --sha256_password_auto_generate_rsa_keys are placed in data directory.
2021-10-08T07:20:14.557861+08:00 0 [Note] [MY-010308] [Server] Skipping generation of RSA key pair through --caching_sha2_password_auto_generate_rsa_keys as key files are present in data directory.
2021-10-08T07:20:14.574245+08:00 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: )Ifb&36xfVF<
```

```
[root@pmydb03 ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/etc/my.cnf --initialize --user=mysql
[root@pmydb03 ~]# grep -i password /pgold/data/error.log
2021-10-08T07:20:20.769598+08:00 0 [Note] [MY-010309] [Server] Auto generated RSA key files through --sha256_password_auto_generate_rsa_keys are placed in data directory.
2021-10-08T07:20:20.771999+08:00 0 [Note] [MY-010308] [Server] Skipping generation of RSA key pair through --caching_sha2_password_auto_generate_rsa_keys as key files are present in data directory.
2021-10-08T07:20:20.790114+08:00 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: Tr5Pqq:Sk/zi
```

```
mysqldadmin shutdown -uroot -pohsdba -S/pgold/data/mysql.sock
```

创建管理用户

```
mysqld_safe --defaults-file=/etc/my.cnf &  
mysql -uroot -S/pgold/data/mysql.sock -p  
alter user current_user() identified by 'ohsdba';
```

```
[root@pmydb01 ~]# mysqld_safe --defaults-file=/etc/my.cnf &  
[1] 3147  
[root@pmydb01 ~]# 2021-10-07T23:24:17.980324Z mysqld_safe Logging to '/pgold/data/error.log'.  
2021-10-07T23:24:18.012957Z mysqld_safe Starting mysqld daemon with databases from /pgold/data  
  
[root@pmydb01 ~]# mysql -uroot -S/pgold/data/mysql.sock -p  
Enter password:  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 8  
Server version: 8.0.26  
  
Copyright (c) 2000, 2021, Oracle and/or its affiliates.  
  
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
mysql> alter user current_user() identified by 'ohsdba';  
Query OK, 0 rows affected (0.08 sec)
```

```
set sql_log_bin=0;  
create user root@%' identified by 'ohsdba';
```

grant select, insert, update, delete, create, drop, reload, shutdown, process, file, references, index, alter, show databases, super, create temporary tables, lock tables, execute, replication slave, replication client, create view, show view, create routine, alter routine, create user, event, trigger, create tablespace, create role, drop role on *.* to 'root'@'%' with grant option;
grant backup_admin, binlog_admin, connection_admin, encryption_key_admin, group_replication_admin, persist_ro_variables_admin, replication_slave_admin, resource_group_admin, resource_group_user, role_admin, set_user_id, system_variables_admin, xa_recover_admin on *.* to 'root'@'%' with grant option;
flush privileges;
set sql_log_bin=1;

```
mysql> set sql_log_bin=0;
create user root@'%' identified by 'ohsdba';
grant select, insert, update, delete, create, drop, reload, shutdown, process, file, references, index, alter, show databases, super, create temporary tables, lock tables, execute, replication slave, replication client, create view, show view, create routine, alter routine, create user, event, trigger, create tablespace, create role, drop role on *.* to 'root'@'%' with grant option;
grant backup_admin, binlog_admin, connection_admin, encryption_key_admin, group_replication_admin, persist_ro_variables_admin, replication_slave_admin, resource_group_admin, resource_group_user, role_admin, set_user_id, system_variables_admin, xa_recover_admin on *.* to 'root'@'%' with grant option;
flush privileges;
set sql_log_bin=1;Query OK, 0 rows affected (0.04 sec)

mysql> create user root@'%' identified by 'ohsdba';
Query OK, 0 rows affected (0.13 sec)

mysql> grant select, insert, update, delete, create, drop, reload, shutdown, process, file, references, index, alter, show databases, super, create temporary tables, lock tables, execute, replication slave, replication client, create view, show view, create routine, alter routine, create user, event, trigger, create tablespace, create role, drop role on *.* to 'root'@'%' with grant option;
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> grant backup_admin, binlog_admin, connection_admin, encryption_key_admin, group_replication_admin, persist_ro_variables_admin, replication_slave_admin, resource_group_admin, resource_group_user, role_admin, set_user_id, system_variables_admin, xa_recover_admin on *.* to 'root'@'%' with grant option;
Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.01 sec)

mysql> set sql_log_bin=1;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

装载样本数据

在第一个实例上创建数据库 ohsdba，并装载 demo 数据。样本数据是通过下面的链接获取的。

<https://wiki.openmrs.org/download/attachments/5047323/large-demo-data-2-2-1.sql.gz>

```
mysql> create database ohsdba;
Query OK, 1 row affected (0.03 sec)

mysql> use ohsdba
Database changed
mysql> source /tmp/mysql-large-demo-data-2-2-1.sql
Query OK, 0 rows affected (0.02 sec)
```

创建 MGR 集群

通过 mysql shell 连接 MySQL 实例，执行检测工作，然后再创建集群

```
mysqlsh --uri root@192.168.10.231:3306
```

```
[root@pmydb01 ~]# mysqlsh --uri root@192.168.10.231:3306
Please provide the password for 'root@192.168.10.231:3306': *****
Save password for 'root@192.168.10.231:3306'? [Y]es/[N]o/[v]er (default No): y
MySQL Shell 8.0.26
```

```
Copyright (c) 2016, 2021, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.
```

```
Type '\help' or '\?' for help; '\quit' to exit.
Creating a session to 'root@192.168.10.231:3306'
Fetching schema names for autocompletion... Press ^C to stop.
Your MySQL connection id is 15
Server version: 8.0.26 MySQL Community Server - GPL
No default schema selected; type \use <schema> to set one.
MySQL 192.168.10.231:3306 ssl JS > \sql select user();
```

```
+-----+
| user() |
+-----+
| root@pmydb01 |
+-----+
```

```
1 row in set (0.0001 sec)
```

```
MySQL 192.168.10.231:3306 ssl JS > \help dba.configureInstance
```

NAME

configureInstance - Validates and configures an instance for MySQL InnoDB Cluster usage.

SYNTAX

dba.configureInstance([instance][, options])

WHERE

instance: An instance definition.
options: Additional options for the operation.

检查实例

检查实例时，发现 root@'%' 的权限不满足。授权后，再检测就正常了。检测命令为

```
dba.configureInstance()
```

```
dba.checkInstanceConfiguration()
```

```
GRANT CLONE_ADMIN, REPLICATION_APPLIER ON *.* TO 'root'@'%' WITH GRANT OPTION;
```

```
MySQL 192.168.10.231:3306 ssl JS >
MySQL 192.168.10.231:3306 ssl JS > dba.configureInstance();
Configuring local MySQL instance listening at port 3306 for use in an InnoDB cluster...
ERROR: The account 'root'@'%' is missing privileges required to manage an InnoDB cluster:
GRANT CLONE_ADMIN, REPLICATION_APPLIER ON *.* TO 'root'@'%' WITH GRANT OPTION;
For more information, see the online documentation.
Dba.configureInstance: The account 'root'@'%' is missing privileges required to manage an InnoDB cluster. (RuntimeError)
MySQL 192.168.10.231:3306 ssl JS > \sql GRANT CLONE_ADMIN, REPLICATION_APPLIER ON *.* TO 'root'@'%' WITH GRANT OPTION;
Query OK, 0 rows affected (0.0440 sec)
MySQL 192.168.10.231:3306 ssl JS >
MySQL 192.168.10.231:3306 ssl JS > dba.configureInstance();
Configuring local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb01:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

applierWorkerThreads will be set to the default value of 4.

The instance 'pmydb01:3306' is valid to be used in an InnoDB cluster.
The instance 'pmydb01:3306' is already ready to be used in an InnoDB cluster.

Successfully enabled parallel appliers.
```

检查配置

我们在第一个实例上导入了 demo 数据。因此，在检测的时候发现，有个表 `ohsdba.room_temperature` 不满足，查询后我们发现这个表是 MyISAM 存储引擎。修改此表为 `innodb` 后再次检测就好了。命令为 `dba.checkInstanceConfiguration()`

```
MySQL 192.168.10.231:3306 ssl JS > dba.checkInstanceConfiguration()
Validating local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb01:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

Checking whether existing tables comply with Group Replication requirements...
ERROR: The following tables use a storage engine that are not supported by Group Replication:
ohsdba.room_temperature

Group Replication requires tables to use InnoDB and have a PRIMARY KEY or PRIMARY KEY Equivalent (non-null unique key). Tables that do not follow these req
uirements will be readable but not updateable when used with Group Replication. If your applications make updates (INSERT, UPDATE or DELETE) to these table
s, ensure they use the InnoDB storage engine and have a PRIMARY KEY or PRIMARY KEY Equivalent.
If you can't change the tables structure to include an extra visible key to be used as PRIMARY KEY, you can make use of the INVISIBLE COLUMN feature availa
ble since 8.0.23: https://dev.mysql.com/doc/refman/8.0/en/invisible-columns.html

Checking instance configuration...

{
  "status": "error"
}
MySQL 192.168.10.231:3306 ssl JS >
```

```
Checking whether existing tables comply with Group Replication requirements...
ERROR: The following tables use a storage engine that are not supported by Group Replication:
ohsdba.room_temperature
```

```

mysql> show create table room_temperature;
+-----+
+-----+
| Table          | Create Table
+-----+-----+
| room_temperature | CREATE TABLE `room_temperature` (
  `room_temperature_id` int NOT NULL AUTO_INCREMENT,
  `time` datetime NOT NULL,
  `temp` int NOT NULL,
  `uuid` varchar(38) NOT NULL,
  PRIMARY KEY (`room_temperature_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-----+-----+
1 row in set (0.00 sec)

mysql> select count(*) from room_temperature;
+-----+
| count(*) |
+-----+
|         0 |
+-----+
1 row in set (0.00 sec)

mysql> alter table room_temperature engine=innodb;
Query OK, 0 rows affected (0.06 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> show create table room_temperature;
+-----+
+-----+
| Table          | Create Table
+-----+-----+
| room_temperature | CREATE TABLE `room_temperature` (
  `room_temperature_id` int NOT NULL AUTO_INCREMENT,
  `time` datetime NOT NULL,
  `temp` int NOT NULL,
  `uuid` varchar(38) NOT NULL,
  PRIMARY KEY (`room_temperature_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-----+-----+
1 row in set (0.00 sec)

```



```
MySQL 192.168.10.231:3306 ssl JS > dba.checkInstanceConfiguration()
Validating local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb01:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

Checking whether existing tables comply with Group Replication requirements...
No incompatible tables detected

Checking instance configuration...
Instance configuration is compatible with InnoDB cluster

The instance 'pmydb01:3306' is valid to be used in an InnoDB cluster.

{
  "status": "ok"
}
MySQL 192.168.10.231:3306 ssl JS >
```

连接第二个实例，该实例之前只做了初始化，修改了 root 密码。

```

MySQL 192.168.10.232:3306 ssl JS >
MySQL 192.168.10.232:3306 ssl JS > dba.configureInstance();
Configuring local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb02:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

applierWorkerThreads will be set to the default value of 4.

The instance 'pmydb02:3306' is valid to be used in an InnoDB cluster.
The instance 'pmydb02:3306' is already ready to be used in an InnoDB cluster.

Successfully enabled parallel appliers.
MySQL 192.168.10.232:3306 ssl JS >
MySQL 192.168.10.232:3306 ssl JS > dba.checkInstanceConfiguration()
Validating local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb02:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

Checking whether existing tables comply with Group Replication requirements...
No incompatible tables detected

Checking instance configuration...
Instance configuration is compatible with InnoDB cluster

The instance 'pmydb02:3306' is valid to be used in an InnoDB cluster.

{
  "status": "ok"
}
MySQL 192.168.10.232:3306 ssl JS >

```

连接第三个实例，该实例之前只做了初始化，修改了 root 密码。

```
MySQL 192.168.10.233:3306 ssl JS > dba.configureInstance();
Configuring local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb03:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

applierWorkerThreads will be set to the default value of 4.

The instance 'pmydb03:3306' is valid to be used in an InnoDB cluster.
The instance 'pmydb03:3306' is already ready to be used in an InnoDB cluster.

Successfully enabled parallel appliers.
MySQL 192.168.10.233:3306 ssl JS >
MySQL 192.168.10.233:3306 ssl JS >
MySQL 192.168.10.233:3306 ssl JS > dba.checkInstanceConfiguration()
Validating local MySQL instance listening at port 3306 for use in an InnoDB cluster...

This instance reports its own address as pmydb03:3306
Clients and other cluster members will communicate with it through this address by default. If this is not correct, the report_host MySQL system variable s
ould be changed.

Checking whether existing tables comply with Group Replication requirements...
No incompatible tables detected

Checking instance configuration...
Instance configuration is compatible with InnoDB cluster

The instance 'pmydb03:3306' is valid to be used in an InnoDB cluster.

{
  "status": "ok"
}
MySQL 192.168.10.233:3306 ssl JS >
MySQL 192.168.10.233:3306 ssl JS > \exit
Bye!
```

创建集群

```
dba.createCluster('ohsCluster')
```

```
MySQL 192.168.10.231:3306 ssl JS > dba.createCluster('ohsCluster')
A new InnoDB cluster will be created on instance '192.168.10.231:3306'.

Validating instance configuration at 192.168.10.231:3306...

This instance reports its own address as pmydb01:3306

Instance configuration is suitable.
NOTE: Group Replication will communicate with other members using 'pmydb01:33061'. Use the localAddress option to override.

Creating InnoDB cluster 'ohsCluster' on 'pmydb01:3306'...

Adding Seed Instance...
Cluster successfully created. Use Cluster.addInstance() to add MySQL instances.
At least 3 instances are needed for the cluster to be able to withstand up to
one server failure.

<Cluster:ohsCluster>
MySQL 192.168.10.231:3306 ssl JS >
```

加入集群

```
var ohsmgr=dba.getCluster()
ohsmgr.addInstance('root@192.168.10.232:3306')
ohsmgr.addInstance('root@192.168.10.233:3306')
```

```

MySQL 192.168.10.231:3306 ssl JS > var ohsmgr=dba.getCluster()
MySQL 192.168.10.231:3306 ssl JS > ohsmgr.addInstance('root@192.168.10.232:3306')

WARNING: A GTID set check of the MySQL instance at 'pmydb02:3306' determined that it contains transactions that do not originate from the cluster, which must be discarded before it can join the cluster.

pmydb02:3306 has the following errant GTIDs that do not exist in the cluster:
1ddb2bf5-27c5-11ec-91eb-080027f7eac0:1-2

WARNING: Discarding these extra GTID events can either be done manually or by completely overwriting the state of pmydb02:3306 with a physical snapshot from an existing cluster member. To use this method by default, set the 'recoveryMethod' option to 'clone'.

Having extra GTID events is not expected, and it is recommended to investigate this further and ensure that the data can be removed prior to choosing the clone recovery method.

Please select a recovery method [C]lone/[A]bort (default Abort): C
Validating instance configuration at 192.168.10.232:3306...

This instance reports its own address as pmydb02:3306

Instance configuration is suitable.
NOTE: Group Replication will communicate with other members using 'pmydb02:33061'. Use the localAddress option to override.

A new instance will be added to the InnoDB cluster. Depending on the amount of data on the cluster this might take from a few seconds to several hours.

Adding instance to the cluster...

Monitoring recovery process of the new cluster member. Press ^C to stop monitoring and let it continue in background.
Clone based state recovery is now in progress.

NOTE: A server restart is expected to happen as part of the clone process. If the server does not support the RESTART command or does not come back after a while, you may need to manually start it back.

* Waiting for clone to finish...
NOTE: pmydb02:3306 is being cloned from pmydb01:3306
** Stage DROP DATA: Completed
** Clone Transfer
  FILE COPY ##### 100% Completed
  PAGE COPY ##### 100% Completed
  REDO COPY  ##### 100% Completed

```

```
NOTE: pmydb02:3306 is shutting down...
* Waiting for server restart... ready
* pmydb02:3306 has restarted, waiting for clone to finish...
** Stage RESTART: Completed
* Clone process has finished: 399.70 MB transferred in 4 sec (99.92 MB/s)

Incremental state recovery is now in progress.

* Waiting for distributed recovery to finish...
NOTE: 'pmydb02:3306' is being recovered from 'pmydb01:3306'
* Distributed recovery has finished

The instance 'pmydb02:3306' was successfully added to the cluster.

MySQL 192.168.10.231:3306 ssl JS >
```

从上面添加的过程看，大概是执行了下面的命令。我们可以看到，本次复制是从主库直接复制过来的。

```
reset master;
```

```
set global clone_valid_donor_list='192.168.10.231:3306';
```

```
clone instance from root@192.168.10.231:3306 identified by 'ohsdba';
```

```

MySQL 192.168.10.231:3306 ssl JS > ohsmgr.addInstance('root@192.168.10.233:3306')

WARNING: A GTID set check of the MySQL instance at 'pmydb03:3306' determined that it contains transactions that do not originate from the cluster, which must be discarded before it can join the cluster.

pmydb03:3306 has the following errant GTIDs that do not exist in the cluster:
21867ccf-27c5-11ec-94b1-080027089a01:1-2

WARNING: Discarding these extra GTID events can either be done manually or by completely overwriting the state of pmydb03:3306 with a physical snapshot from an existing cluster member. To use this method by default, set the 'recoveryMethod' option to 'clone'.

Having extra GTID events is not expected, and it is recommended to investigate this further and ensure that the data can be removed prior to choosing the clone recovery method.

Please select a recovery method [C]lone/[A]bort (default Abort): C
Validating instance configuration at 192.168.10.233:3306...

This instance reports its own address as pmydb03:3306

Instance configuration is suitable.
NOTE: Group Replication will communicate with other members using 'pmydb03:33061'. Use the localAddress option to override.

A new instance will be added to the InnoDB cluster. Depending on the amount of data on the cluster this might take from a few seconds to several hours.

Adding instance to the cluster...

Monitoring recovery process of the new cluster member. Press ^C to stop monitoring and let it continue in background.
Clone based state recovery is now in progress.

NOTE: A server restart is expected to happen as part of the clone process. If the server does not support the RESTART command or does not come back after a while, you may need to manually start it back.

* Waiting for clone to finish...
NOTE: pmydb03:3306 is being cloned from pmydb02:3306
** Stage DROP DATA: Completed
** Clone Transfer
FILE COPY ##### 100% Completed
PAGE COPY ##### 100% Completed
REDO COPY ##### 100% Completed

```

```
NOTE: pmydb03:3306 is shutting down...
* Waiting for server restart... ready
* pmydb03:3306 has restarted, waiting for clone to finish...
** Stage RESTART: Completed
* Clone process has finished: 399.70 MB transferred in 4 sec (99.92 MB/s)

State recovery already finished for 'pmydb03:3306'

The instance 'pmydb03:3306' was successfully added to the cluster.

MySQL 192.168.10.231:3306 ssl JS >
```

从上面添加的过程看，大概是执行了下面的命令。在加入第三个节点的时候，是从 SECONDARY 节点复制的数据，这样就避免了对生产环境的影响。

```
reset master;
set global clone_valid_donor_list='192.168.10.232:3306';
clone instance from root@192.168.10.232:3306 identified by 'ohsdba';
```

查看集群状态

```
var ohsmgr=dba.getCluster()
ohsmgr.status();
```



```

MySQL 192.168.10.231:3306 ssl JS > var ohsmgr=dba.getCluster()
MySQL 192.168.10.231:3306 ssl JS > ohsmgr.status();
{
  "clusterName": "ohsCluster",
  "defaultReplicaSet": {
    "name": "default",
    "primary": "pmydb01:3306",
    "ssl": "REQUIRED",
    "status": "OK",
    "statusText": "Cluster is ONLINE and can tolerate up to ONE failure.",
    "topology": {
      "pmydb01:3306": {
        "address": "pmydb01:3306",
        "memberRole": "PRIMARY",
        "mode": "R/W",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
      },
      "pmydb02:3306": {
        "address": "pmydb02:3306",
        "memberRole": "SECONDARY",
        "mode": "R/O",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
      },
      "pmydb03:3306": {
        "address": "pmydb03:3306",
        "memberRole": "SECONDARY",
        "mode": "R/O",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
      }
    },
    "topologyMode": "Single-Primary"
  },
  "groupInformationSourceMember": "pmydb01:3306"
}
MySQL 192.168.10.231:3306 ssl JS >

```

ohsmgr.describe();

```
MySQL 192.168.10.231:3306 ssl JS > ohsmgr.describe();
{
  "clusterName": "ohsCluster",
  "defaultReplicaSet": {
    "name": "default",
    "topology": [
      {
        "address": "pmydb01:3306",
        "label": "pmydb01:3306",
        "role": "HA"
      },
      {
        "address": "pmydb02:3306",
        "label": "pmydb02:3306",
        "role": "HA"
      },
      {
        "address": "pmydb03:3306",
        "label": "pmydb03:3306",
        "role": "HA"
      }
    ],
    "topologyMode": "Single-Primary"
  }
}
MySQL 192.168.10.231:3306 ssl JS >
MySQL 192.168.10.231:3306 ssl JS > \sql
Switching to SQL mode... Commands end with ;
MySQL 192.168.10.231:3306 ssl SQL > select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME | MEMBER_ID | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 171a636c-27c5-11ec-9cff-08002760df88 | pmydb01 | 3306 | ONLINE | PRIMARY | 8.0.26 |
| group_replication_applier | 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 | pmydb02 | 3306 | ONLINE | SECONDARY | 8.0.26 |
| group_replication_applier | 21867ccf-27c5-11ec-94b1-080027089a01 | pmydb03 | 3306 | ONLINE | SECONDARY | 8.0.26 |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.0016 sec)
MySQL 192.168.10.231:3306 ssl SQL >
```

验证样本数据

我们发现，在第一个节点上创建的 demo 数据，在添加节点的时候，数据库 ohsdba 也同时复制过来了。

```
MySQL JS > \c root@192.168.10.232:3306
Creating a session to 'root@192.168.10.232:3306'
Fetching schema names for autocompletion... Press ^C to stop.
Your MySQL connection id is 56
Server version: 8.0.26 MySQL Community Server - GPL
No default schema selected; type \use <schema> to set one.
MySQL 192.168.10.232:3306 ssl JS > \sql
Switching to SQL mode... Commands end with ;
MySQL 192.168.10.232:3306 ssl SQL > select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 171a636c-27c5-11ec-9cff-08002760df88 | pmydb01     | 3306        | ONLINE       | PRIMARY     | 8.0.26         |
| group_replication_applier | 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 | pmydb02     | 3306        | ONLINE       | SECONDARY   | 8.0.26         |
| group_replication_applier | 21867ccf-27c5-11ec-94b1-080027089a01 | pmydb03     | 3306        | ONLINE       | SECONDARY   | 8.0.26         |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.0167 sec)
MySQL 192.168.10.232:3306 ssl SQL > show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| mysql_innodb_cluster_metadata |
| ohsdba |
| performance_schema |
| sys |
+-----+
6 rows in set (0.0011 sec)
```

添加或删除节点

如果想添加一个新节点，通过以下步骤即可

- 用有管理权限的账号，通过 `mysqlsh` 登陆 MySQL 实例
- 执行 `dba.configureInstance()`
- 执行 `ohsmgr.addInstance()`

样例

```
dba.configureInstance()
var ohsmgr=dba.getCluster()
ohsmgr.addInstance('root@192.168.10.234:3306')
```

如果想删除一个新节点，通过以下步骤即可

- 用有管理权限的账号，通过 `mysqlsh` 登陆 MySQL 实例
- 执行 `ohsmgr.removeInstance()`

样例

```
var ohsmgr=dba.getCluster()
ohsmgr.removeInstance('root@192.168.10.234:3306')
```

如果关闭了 MGR，可以通过下面的方式来启动

```
set global group_replication_bootstrap_group=on;
start group_replication;
set global group_replication_bootstrap_group=off;
```

MGR 主备切换

组复制中的成员之间，角色是可以互换的，可以通过 select 语句或者 MySQL Shell 来进行切换。可以设置成多主模式，建议设置成单主。

```
select group_replication_set_as_primary('1ddb2bf5-27c5-11ec-91eb-080027f7eac0');
```

```
MySQL 192.168.10.232:3306 ssl JS > ohsmgr.setPrimaryInstance('pmydb01:3306');  
Setting instance 'pmydb01:3306' as the primary instance of cluster 'ohsCluster'...  
Instance 'pmydb01:3306' was switched from SECONDARY to PRIMARY.  
Instance 'pmydb02:3306' was switched from PRIMARY to SECONDARY.  
Instance 'pmydb03:3306' remains SECONDARY.  
The instance 'pmydb01:3306' was successfully elected as primary.  
MySQL 192.168.10.232:3306 ssl JS >
```

```

MySQL 192.168.10.232:3306 ssl SQL> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME | MEMBER_ID | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 171a636c-27c5-11ec-9cff-08002760df88 | pmydb01 | 3306 | ONLINE | PRIMARY | 8.0.26 |
| group_replication_applier | 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 | pmydb02 | 3306 | ONLINE | SECONDARY | 8.0.26 |
| group_replication_applier | 21867ccf-27c5-11ec-94b1-080027089a01 | pmydb03 | 3306 | ONLINE | SECONDARY | 8.0.26 |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.0007 sec)
MySQL 192.168.10.232:3306 ssl SQL>
MySQL 192.168.10.232:3306 ssl SQL> select group_replication_set_as_primary('1ddb2bf5-27c5-11ec-91eb-080027f7eac0');
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_set_as_primary('1ddb2bf5-27c5-11ec-91eb-080027f7eac0') |
+-----+-----+-----+-----+-----+-----+-----+
| Primary server switched to: 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (1.1445 sec)
MySQL 192.168.10.232:3306 ssl SQL> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME | MEMBER_ID | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 171a636c-27c5-11ec-9cff-08002760df88 | pmydb01 | 3306 | ONLINE | SECONDARY | 8.0.26 |
| group_replication_applier | 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 | pmydb02 | 3306 | ONLINE | PRIMARY | 8.0.26 |
| group_replication_applier | 21867ccf-27c5-11ec-94b1-080027089a01 | pmydb03 | 3306 | ONLINE | SECONDARY | 8.0.26 |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.0011 sec)

```

```
MySQL 192.168.10.232:3306 ssl JS > ohsmgr.status();
{
  "clusterName": "ohsCluster",
  "defaultReplicaSet": {
    "name": "default",
    "primary": "pmydb02:3306",
    "ssl": "REQUIRED",
    "status": "OK",
    "statusText": "Cluster is ONLINE and can tolerate up to ONE failure.",
    "topology": {
      "pmydb01:3306": {
        "address": "pmydb01:3306",
        "memberRole": "SECONDARY",
        "mode": "R/O",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
      },
      "pmydb02:3306": {
        "address": "pmydb02:3306",
        "memberRole": "PRIMARY",
        "mode": "R/W",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
      },
      "pmydb03:3306": {
        "address": "pmydb03:3306",
        "memberRole": "SECONDARY",
        "mode": "R/O",
        "readReplicas": {},
        "replicationLag": null,
        "role": "HA",
        "status": "ONLINE",
        "version": "8.0.26"
      }
    },
    "topologyMode": "Single-Primary"
  },
}
```

```
MySQL 192.168.10.232:3306 ssl JS > ohsmgr.setPrimaryInstance('pmydb01:3306');
Setting instance 'pmydb01:3306' as the primary instance of cluster 'ohsCluster'...

Instance 'pmydb01:3306' was switched from SECONDARY to PRIMARY.
Instance 'pmydb02:3306' was switched from PRIMARY to SECONDARY.
Instance 'pmydb03:3306' remains SECONDARY.

The instance 'pmydb01:3306' was successfully elected as primary.
MySQL 192.168.10.232:3306 ssl JS >
```

配置和使用 MySQL Router

官方提供的一个用来实现负载的均衡的软件，需要提前自己下载。主要作用如下：将客户端请求重定向到适合的数据库服务器。缓存 MySQL InnoDB Cluster 的元数据负责把客户端的读/写请求路由到当前的主数据库节点，还可以对 client 的请求进行负载均衡，并且在主数据库节点出现故障时，保证 client 的请求被路由到新的主服务器节点。

我们用了一台单独的服务器 pmyapp，上面安装了 MySQL Router 和 MySQL 客户端工具。建议应用和 MySQL Router 部署在同一台服务器上。

```
[root@pmyapp ~]# cat /etc/hosts
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1        localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.10.230 pmyapp
192.168.10.231 pmydb01
192.168.10.232 pmydb02
192.168.10.233 pmydb03
```

```
groupadd -g 27 -o -r mysql
useradd -M -N -g mysql -o -r -d /pgold/mysql -s /bin/false -c "MySQL Server" -u 27 mysql
mkdir /pgold/mysql/router
cd /pgold/mysql/router
tar zxvf /tmp/mysql-router-8.0.26-el7-x86_64.tar.gz
mv mysql-router-8.0.26-el7-x86_64 8026
chown -R mysql:mysql /pgold
/pgold/mysql/router/8026/bin/mysqlrouter
```

```
[root@pmyapp ~]# which mysqlrouter
/pgold/mysql/router/8026/bin/mysqlrouter
[root@pmyapp ~]# █
```

初始化 MySQL Router

```
mysqlrouter --bootstrap root@192.168.10.231:3306 --user=mysql -d /pgold/mysql/my
```

```
[root@pmyapp ~]# mysqlrouter --bootstrap root@192.168.10.231:3306 --user=mysql
Please enter MySQL password for root:
# Bootstrapping system MySQL Router instance...

- Creating account(s) (only those that are needed, if any)
- Verifying account (using it to run SQL queries that would be run by Router)
- Storing account in keyring
- Adjusting permissions of generated files
- Creating configuration /pgold/mysql/router/8026/mysqlrouter.conf

# MySQL Router configured for the InnoDB Cluster 'ohsCluster'

After this MySQL Router has been started with the generated configuration

    $ /etc/init.d/mysqlrouter restart
or
    $ systemctl start mysqlrouter
or
    $ mysqlrouter -c /pgold/mysql/router/8026/mysqlrouter.conf

the cluster 'ohsCluster' can be reached by connecting to:

## MySQL Classic protocol

- Read/Write Connections: localhost:6446
- Read/Only Connections:  localhost:6447

## MySQL X protocol

- Read/Write Connections: localhost:6448
- Read/Only Connections:  localhost:6449

[root@pmyapp ~]#
```

MySQL Router 配置文件

配置文件的名称为 mysqlrouter.conf，该文件是自动生成的

```
[root@pmyapp ~]# cat /pgold/mysql/router/8026/mysqlrouter.conf
# File automatically generated during MySQL Router bootstrap
[DEFAULT]
name=system
user=mysql
keyring_path=/pgold/mysql/router/8026/var/lib/mysqlrouter/keyring
master_key_path=/pgold/mysql/router/8026/mysqlrouter.key
connect_timeout=15
read_timeout=30
dynamic_state=/pgold/mysql/router/8026/bin/./var/lib/mysqlrouter/state.json
client_ssl_cert=/pgold/mysql/router/8026/var/lib/mysqlrouter/router-cert.pem
client_ssl_key=/pgold/mysql/router/8026/var/lib/mysqlrouter/router-key.pem
client_ssl_mode=PREFERRED
server_ssl_mode=AS_CLIENT
server_ssl_verify=DISABLED

[logger]
level = INFO

[metadata_cache:ohsCluster]
cluster_type=gr
router_id=1
```

```
user=mysql_router1_avmkhf6zl1a2
metadata_cluster=ohsCluster
ttl=0.5
auth_cache_ttl=-1
auth_cache_refresh_interval=2
use_gr_notifications=0

[routing:ohsCluster_rw]
bind_address=0.0.0.0
bind_port=6446
destinations=metadata-cache://ohsCluster/?role=PRIMARY
routing_strategy=first-available
protocol=classic

[routing:ohsCluster_ro]
bind_address=0.0.0.0
bind_port=6447
destinations=metadata-cache://ohsCluster/?role=SECONDARY
routing_strategy=round-robin-with-fallback
protocol=classic

[routing:ohsCluster_x_rw]
bind_address=0.0.0.0
bind_port=6448
destinations=metadata-cache://ohsCluster/?role=PRIMARY
routing_strategy=first-available
```

protocol=x

[routing:ohsCluster_x_ro]

bind_address=0.0.0.0

bind_port=6449

destinations=metadata-cache://ohsCluster/?role=SECONDARY

routing_strategy=round-robin-with-fallback

protocol=x

[http_server]

port=8443

ssl=1

ssl_cert=/pgold/mysql/router/8026/var/lib/mysqlrouter/router-cert.pem

ssl_key=/pgold/mysql/router/8026/var/lib/mysqlrouter/router-key.pem

[http_auth_realm:default_auth_realm]

backend=default_auth_backend

method=basic

name=default_realm

[rest_router]

require_realm=default_auth_realm

[rest_api]

[http_auth_backend:default_auth_backend]

```
backend=metadata_cache

[rest_routing]
require_realm=default_auth_realm

[rest_metadata_cache]
require_realm=default_auth_realm

[root@pmyapp ~]#
```

启动 MySQL Router

```
mysqlrouter -c /pgold/mysql/router/8026/mysqlrouter.conf -u=mysql &
```

```
[root@pmyapp ~]# mysqlrouter -c /pgold/mysql/router/8026/mysqlrouter.conf -u=mysql
Error: stat() failed for /pgold/mysql/router/8026/var/lib/mysqlrouter/keyring': Permission denied
[root@pmyapp ~]# ls -l /pgold/mysql/router/8026/var
total 4
drwx-----. 3 root root 4096 Oct  8 14:50 lib
[root@pmyapp ~]# chown -R mysql:mysql /pgold
```

```
[root@pmyapp ~]# mysqlrouter -c /pgold/mysql/router/8026/mysqlrouter.conf -u=mysql &
[1] 16268
[root@pmyapp ~]# logging facility initialized, switching logging to loggers specified in configuration

[root@pmyapp ~]# netstat -nlp|grep mysql
tcp        0      0 0.0.0.0:8443          0.0.0.0:*            LISTEN     16268/mysqlrouter
tcp        0      0 0.0.0.0:6446          0.0.0.0:*            LISTEN     16268/mysqlrouter
tcp        0      0 0.0.0.0:6447          0.0.0.0:*            LISTEN     16268/mysqlrouter
tcp        0      0 0.0.0.0:6448          0.0.0.0:*            LISTEN     16268/mysqlrouter
tcp        0      0 0.0.0.0:6449          0.0.0.0:*            LISTEN     16268/mysqlrouter
[root@pmyapp ~]#
```

测试读写分离

MySQL Innodb Cluster 的读写分离，是通过 MySQL Router 的端口是实现的。默认的写端口是 6446，默认读端口是 6447

```
[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6446 -e "select @@server_uuid;"
Enter password:
+-----+
| @@server_uuid |
+-----+
| 171a636c-27c5-11ec-9cff-08002760df88 |
+-----+
[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6446 -e "select @@server_uuid;"
Enter password:
+-----+
| @@server_uuid |
+-----+
| 171a636c-27c5-11ec-9cff-08002760df88 |
+-----+
```

```

[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6447 -e "select @@hostname;"
Enter password:
+-----+
| @@hostname |
+-----+
| pmydb03    |
+-----+
[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6447 -e "select @@hostname;"
Enter password:
+-----+
| @@hostname |
+-----+
| pmydb02    |
+-----+
[root@pmyapp ~]#
[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6446 -e "select @@server_uuid;"
Enter password:
+-----+
| @@server_uuid |
+-----+
| 171a636c-27c5-11ec-9cff-08002760df88 |
+-----+
[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6446 -e "select @@hostname;"
Enter password:
+-----+
| @@hostname |
+-----+
| pmydb01    |
+-----+
[root@pmyapp ~]# mysql -h127.0.0.1 -uroot -p -P6446 -e "select * from performance_schema.replication_group_members;"
Enter password:
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 171a636c-27c5-11ec-9cff-08002760df88 | pmydb01    | 3306 | ONLINE | PRIMARY | 8.0.26 |
| group_replication_applier | 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 | pmydb02    | 3306 | ONLINE | SECONDARY | 8.0.26 |
| group_replication_applier | 21867ccf-27c5-11ec-94b1-080027089a01 | pmydb03    | 3306 | ONLINE | SECONDARY | 8.0.26 |
+-----+-----+-----+-----+-----+-----+-----+

```

当然也可以在一台服务器上部署多个 MySQL Router，通过 `--directory` 参数

mysqlrouter --bootstrap root@192.168.10.231:3306 --directory /pgold/mysql/router/myrouter --conf-use-sockets --account routerfriend --account-create always --user=mysql

<https://dev.mysql.com/doc/mysql-router/8.0/en/mysql-router-deploying-bootstrapping.html>

```
[root@pmyapp ~]# mysqlrouter --bootstrap root@192.168.10.231:3306 --directory /pgold/mysql/router/myrouter --conf-use-sockets --account routerfriend --account-create always --user=mysql
Please enter MySQL password for root:
# Bootstrapping MySQL Router instance at '/pgold/mysql/router/myrouter'...

Please enter MySQL password for routerfriend:
- Creating account(s)
- Verifying account (using it to run SQL queries that would be run by Router)
- Storing account in keyring
- Adjusting permissions of generated files
- Creating configuration /pgold/mysql/router/myrouter/mysqlrouter.conf

# MySQL Router configured for the InnoDB Cluster 'ohsCluster'

After this MySQL Router has been started with the generated configuration

    $ mysqlrouter -c /pgold/mysql/router/myrouter/mysqlrouter.conf

the cluster 'ohsCluster' can be reached by connecting to:

## MySQL Classic protocol
- Read/Write Connections: localhost:6446, /pgold/mysql/router/myrouter/mysql.sock
- Read/Only Connections: localhost:6447, /pgold/mysql/router/myrouter/mysqlro.sock

## MySQL X protocol
- Read/Write Connections: localhost:6448, /pgold/mysql/router/myrouter/mysqlx.sock
- Read/Only Connections: localhost:6449, /pgold/mysql/router/myrouter/mysqlxro.sock

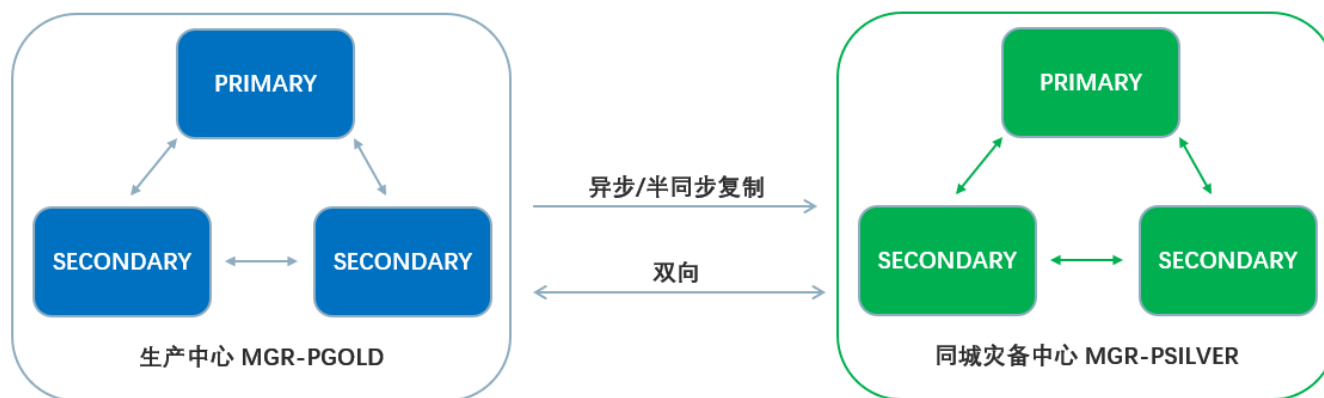
[root@pmyapp ~]#
```


MGR 高可用架构

双中心架构

MGR 解决了单中心的数据库故障。还可以对 MGR 再做半同步复制，实现本地和同城两中心的数据高可用。建议使用下面的方式：

- 确保两套 MGR 集群各节点的 server_uuid 都不重复
- 确保两套 MGR 集群名字 group_replication_group_name 都不重复
- 构建完复制通道后，MGR-PSILVER 里的 Primary 节点最好也设置为只读 super_read_only=1



安装半同步复制 plugin (两套集群都执行)

```
install plugin rpl_semi_sync_master soname 'semisync_master.so';
```

```
install plugin rpl_semi_sync_slave soname 'semisync_slave.so';
```

```
MySQL 192.168.10.231:3306 ssl SQL > install plugin rpl_semi_sync_master soname 'semisync_master.so';
Query OK, 0 rows affected, 1 warning (0.1685 sec)
Note (code 1287): 'rpl_semi_sync_master' is deprecated and will be removed in a future release. Please use rpl_semi_sync_source instead
MySQL 192.168.10.231:3306 ssl SQL > install plugin rpl_semi_sync_slave soname 'semisync_slave.so';
Query OK, 0 rows affected, 1 warning (0.0072 sec)
Note (code 1287): 'rpl_semi_sync_slave' is deprecated and will be removed in a future release. Please use rpl_semi_sync_replica instead
MySQL 192.168.10.231:3306 ssl SQL >
```

修改参数

rpl_semi_sync_master_enabled=on

rpl_semi_sync_slave_enabled=on

rpl_semi_sync_master_timeout=10000

rpl_semi_sync_master_wait_for_slave_count=1

rpl_semi_sync_master_wait_point=AFTER_SYNC

设置复制通道

在 MGR-PSILVER 上执行

change master to

master_host='',master_port=3306,master_user='root',master_password='ohsdba',

master_auto_position=1

for channel 'mgrpgold2mgrpsilver-semirepl';

在 MGR-PGOLD 上执行

change master to

master_host='',master_port=3306,master_user='root',master_password='ohsdba',

master_auto_position=1

for channel 'mgrpsilver2mgrpgold-semirepl';

master_auto_postion 参数的说明:

该参数在 MySQL 5.6.5 版本引入，在设置复制通道(change master to)时使用 master_auto_postion = 1，SALVE 连接 MASTER 将使用基于 GTID 的复制协议，注意首先要开启 GTID_MODE。<https://dev.mysql.com/doc/refman/8.0/en/change-master-to.html>

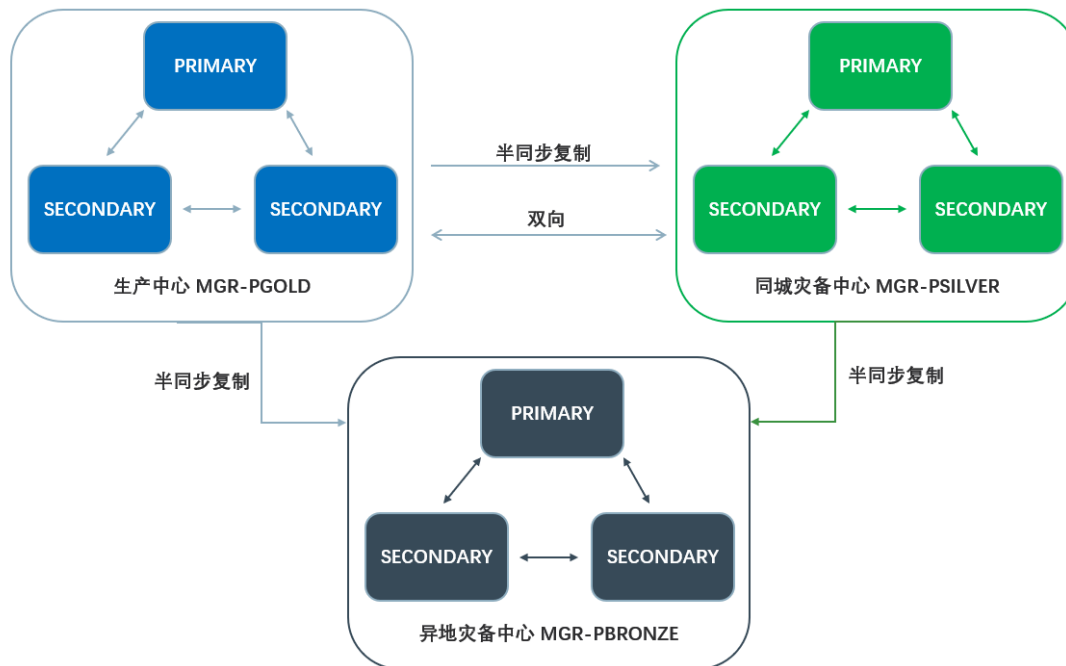
启动 slave (在 MGR-PGOLD 上执行)

```
start replica;
```

```
show replica status\G
```

多中心架构

MGR 两地三中心的高可用，可以在异地建一套。然后选择从生产中心复制，当主生产中心出现问题了，可以从同城灾备中心复制。



我们知道 MGR 可以设置为多主。很多人会有疑问，我们在设置 master 数据源时，只使用了一个地址。如果 MGR 内，主备切换了，或者生产中心或同城

灾备中心瘫痪了，就需要手动修改 master 数据源。在 MySQL 8.0.22 之后，引入了 Async Replication Auto failover 特性，可以为 master 设置多数数据源，还具备了自动 failover 的功能，还可以设置尝试重新连接的次数和时间间隔。

<https://dev.mysql.com/doc/refman/8.0/en/change-master-to.html>

<https://dev.mysql.com/doc/refman/8.0/en/change-replication-source-to.html>

```
CHANGE REPLICATION SOURCE TO option [, option] ... [ channel_option ]
CHANGE MASTER TO option [, option] ... [ channel_option ]

option: {
  SOURCE_BIND = 'interface_name'
| SOURCE_HOST = 'host_name'
| SOURCE_USER = 'user_name'
| SOURCE_PASSWORD = 'password'
| SOURCE_PORT = port_num
| PRIVILEGE_CHECKS_USER = {'account' | NULL}
| REQUIRE_ROW_FORMAT = {0|1}
| REQUIRE_TABLE_PRIMARY_KEY_CHECK = {STREAM | ON | OFF}
| ASSIGN_GTIDS_TO_ANONYMOUS_TRANSACTIONS = {OFF | LOCAL | uuid}
| SOURCE_LOG_FILE = 'source_log_name'
| SOURCE_LOG_POS = source_log_pos
| SOURCE_AUTO_POSITION = {0|1}
| RELAY_LOG_FILE = 'relay_log_name'
| RELAY_LOG_POS = relay_log_pos
| SOURCE_HEARTBEAT_PERIOD = interval
| SOURCE_CONNECT_RETRY = interval
| SOURCE_RETRY_COUNT = count
| SOURCE_CONNECTION_AUTO_FAILOVER = {0|1}

option: {
  MASTER_BIND = 'interface_name'
| MASTER_HOST = 'host_name'
| MASTER_USER = 'user_name'
| MASTER_PASSWORD = 'password'
| MASTER_PORT = port_num
| PRIVILEGE_CHECKS_USER = {'account' | NULL}
| REQUIRE_ROW_FORMAT = {0|1}
| REQUIRE_TABLE_PRIMARY_KEY_CHECK = {STREAM | ON | OFF}
| ASSIGN_GTIDS_TO_ANONYMOUS_TRANSACTIONS = {OFF | LOCAL | uuid}
| MASTER_LOG_FILE = 'source_log_name'
| MASTER_LOG_POS = source_log_pos
| MASTER_AUTO_POSITION = {0|1}
| RELAY_LOG_FILE = 'relay_log_name'
| RELAY_LOG_POS = relay_log_pos
| MASTER_HEARTBEAT_PERIOD = interval
| MASTER_CONNECT_RETRY = interval
| MASTER_RETRY_COUNT = count
| SOURCE_CONNECTION_AUTO_FAILOVER = {0|1}
```

可通过函数设置多个 master 数据源

<https://dev.mysql.com/doc/refman/8.0/en/replication-functions-source-list.html>

`asynchronous_connection_failover_add_source(channel, host, port, network_namespace, weight)`

SOURCE_CONNECTION_AUTO_FAILOVER

Activates the asynchronous connection failover mechanism for a replication channel if one or more alternative replication source servers are available (so when there are multiple MySQL servers or groups of servers that share the replicated data). `SOURCE_CONNECTION_AUTO_FAILOVER` is available as of MySQL 8.0.22. The asynchronous connection failover mechanism takes over after the reconnection attempts controlled by `MASTER_CONNECT_RETRY` and `MASTER_RETRY_COUNT` are exhausted. It reconnects the replica to an alternative source chosen from a specified source list, which you manage using the `asynchronous_connection_failover_add_source` and `asynchronous_connection_failover_delete_source` functions. To add and remove managed groups of servers, use the `asynchronous_connection_failover_add_managed` and `asynchronous_connection_failover_delete_managed` functions instead. For more information, see [Section 17.4.9, “Switching Sources and Replicas with Asynchronous Connection Failover”](#).

Important

1. You can only set `SOURCE_CONNECTION_AUTO_FAILOVER = 1` when GTID auto-positioning is in use (`MASTER_AUTO_POSITION = 1`).
2. When you set `SOURCE_CONNECTION_AUTO_FAILOVER = 1`, set `MASTER_RETRY_COUNT` and `MASTER_CONNECT_RETRY` to minimal numbers that just allow a few retry attempts with the same source in a short time, in case the connection failure is caused by a transient network outage. Otherwise the asynchronous connection failover mechanism cannot be activated promptly. Suitable values are `MASTER_RETRY_COUNT=3` and `MASTER_CONNECT_RETRY=10`, which make the replica retry the connection 3 times with 10-second intervals between.
3. When you set `SOURCE_CONNECTION_AUTO_FAILOVER = 1`, the replication metadata repositories must contain the credentials for a replication user account that can be used to connect to all the servers on the source list for the replication channel. These credentials can be set using the `CHANGE REPLICATION SOURCE TO` statement with the `MASTER_USER` and `MASTER_PASSWORD` options. For more information, see [Section 17.4.9, “Switching Sources and Replicas with Asynchronous Connection Failover”](#).
4. From MySQL 8.0.27, when you set `SOURCE_CONNECTION_AUTO_FAILOVER = 1`, asynchronous connection failover for replicas is automatically activated if this replication channel is on a Group Replication primary in a group in single-primary mode. With this function active, if the primary that is replicating goes offline or into an error state, the new primary starts replication on the same channel when it is elected. If you want to use the function, this replication channel must also be set up on all the secondary servers in the replication group, and on any new joining members. (If the servers are provisioned using MySQL's clone functionality, this all happens automatically.) If you do not want to use the function, disable it by using the `group_replication_disable_member_action` function to disable the Group Replication member action `mysql_start_failover_channels_if_primary`, which is enabled by default. For more information, see [Section 17.4.9.2, “Asynchronous Connection Failover for Replicas”](#).

在生产上启用 master

下面以异地数据中心为例增加一个单向复制通道作为实验。

```
mysql> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 171a636c-27c5-11ec-9cff-08002760df88 | pmydb01     | 3306        | ONLINE       | PRIMARY     | 8.0.26         |
| group_replication_applier | 1ddb2bf5-27c5-11ec-91eb-080027f7eac0 | pmydb02     | 3306        | ONLINE       | SECONDARY   | 8.0.26         |
| group_replication_applier | 21867ccf-27c5-11ec-94b1-080027089a01 | pmydb03     | 3306        | ONLINE       | SECONDARY   | 8.0.26         |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

mysql> show variables like '%semi%';
+-----+-----+
| Variable_name          | Value          |
+-----+-----+
| rpl_semi_sync_master_enabled | OFF           |
| rpl_semi_sync_master_timeout | 10000         |
| rpl_semi_sync_master_trace_level | 32            |
| rpl_semi_sync_master_wait_for_slave_count | 1            |
| rpl_semi_sync_master_wait_no_slave | ON            |
| rpl_semi_sync_master_wait_point | AFTER_SYNC    |
| rpl_semi_sync_slave_enabled | OFF           |
| rpl_semi_sync_slave_trace_level | 32            |
+-----+-----+
8 rows in set (0.04 sec)

mysql> set global rpl_semi_sync_master_enabled=on;
Query OK, 0 rows affected (0.00 sec)

mysql> show variables like '%semi%';
+-----+-----+
| Variable_name          | Value          |
+-----+-----+
| rpl_semi_sync_master_enabled | ON             |
| rpl_semi_sync_master_timeout | 10000         |
| rpl_semi_sync_master_trace_level | 32            |
| rpl_semi_sync_master_wait_for_slave_count | 1            |
| rpl_semi_sync_master_wait_no_slave | ON            |
| rpl_semi_sync_master_wait_point | AFTER_SYNC    |
| rpl_semi_sync_slave_enabled | OFF           |
| rpl_semi_sync_slave_trace_level | 32            |
+-----+-----+
```

在灾备上启用 slave

```
mysql> show variables like '%semi%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| rpl_semi_sync_master_enabled | OFF |
| rpl_semi_sync_master_timeout | 10000 |
| rpl_semi_sync_master_trace_level | 32 |
| rpl_semi_sync_master_wait_for_slave_count | 1 |
| rpl_semi_sync_master_wait_no_slave | ON |
| rpl_semi_sync_master_wait_point | AFTER_SYNC |
| rpl_semi_sync_slave_enabled | OFF |
| rpl_semi_sync_slave_trace_level | 32 |
+-----+-----+
8 rows in set (0.05 sec)

mysql> set global rpl_semi_sync_slave_enabled=on;
Query OK, 0 rows affected (0.00 sec)

mysql> show variables like '%semi%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| rpl_semi_sync_master_enabled | OFF |
| rpl_semi_sync_master_timeout | 10000 |
| rpl_semi_sync_master_trace_level | 32 |
| rpl_semi_sync_master_wait_for_slave_count | 1 |
| rpl_semi_sync_master_wait_no_slave | ON |
| rpl_semi_sync_master_wait_point | AFTER_SYNC |
| rpl_semi_sync_slave_enabled | ON |
| rpl_semi_sync_slave_trace_level | 32 |
+-----+-----+
```

在灾备上初始化数据

MySQL 8.0 clone 插件提供从一个实例克隆数据的功能，非常方便。这里我们就用 clone 插件来克隆数据。

```

mysql> show variables like '%clone%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| clone_autotune_concurrency | ON |
| clone_buffer_size | 4194304 |
| clone_ddl_timeout | 300 |
| clone_donor_timeout_after_network_failure | 5 |
| clone_enable_compression | OFF |
| clone_max_concurrency | 16 |
| clone_max_data_bandwidth | 0 |
| clone_max_network_bandwidth | 0 |
| clone_ssl_ca | |
| clone_ssl_cert | |
| clone_ssl_key | |
| clone_valid_donor_list | |
+-----+-----+
12 rows in set (0.00 sec)

mysql> set global clone_valid_donor_list='192.168.10.231:3306';
Query OK, 0 rows affected (0.01 sec)

mysql> clone instance from root@192.168.10.231:3306 identified by 'ohsdba';
Query OK, 0 rows affected (7.69 sec)

mysql> Restarting mysqld...
2021-10-08T23:51:10.609355Z mysqld_safe Number of processes running now: 0
2021-10-08T23:51:10.616279Z mysqld_safe mysqld restarted

```



```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| mysql_innodb_cluster_metadata |
| ohbdba |
| performance_schema |
| sys |
+-----+
6 rows in set (0.00 sec)

mysql> select binlog_file, binlog_position from performance_schema.clone_status;
+-----+-----+
| binlog_file | binlog_position |
+-----+-----+
| binlog.000003 | 1487 |
+-----+-----+
1 row in set (0.01 sec)

mysql> select @@global.gtid_executed;
+-----+
| @@global.gtid_executed |
+-----+
| 171a636c-27c5-11ec-9cff-08002760df88:1-669,
74c3c2a6-27dd-11ec-bfd3-08002760df88:1-645 |
+-----+
1 row in set (0.00 sec)

mysql>
```

启动 replica 并查看状态

```
mysql> start replica;  
Query OK, 0 rows affected (1.23 sec)
```

```
mysql> show replica status\G  
***** 1. row *****  
      Replica_IO_State: Waiting for source to send event  
      Source_Host: 192.168.10.231  
      Source_User: root  
      Source_Port: 3306  
      Connect_Retry: 60  
      Source_Log_File: binlog.000003  
      Read_Source_Log_Pos: 1487  
      Relay_Log_File: relaylog-mgrpgold2mgrpbronze@002dsemirepl.000003  
      Relay_Log_Pos: 409  
      Relay_Source_Log_File: binlog.000003  
      Replica_IO_Running: Yes  
      Replica_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_Source_Log_Pos: 1487  
      Relay_Log_Space: 823  
      Until_Condition: None  
      Until_Log_File:  
      Until_Log_Pos: 0  
      Source_SSL_Allowed: No  
      Source_SSL_CA_File:  
      Source_SSL_CA_Path:  
      Source_SSL_Cert:  
      Source_SSL_Cipher:  
      Source_SSL_Key:  
      Seconds_Behind_Source: 0  
      Source_SSL_Verify_Server_Cert: No
```

在生产上创建测试数据

```
mysql> use ohsdba
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> desc test;
ERROR 1146 (42S02): Table 'ohsdba.test' doesn't exist
mysql>
mysql> create table test (id int primary key);
Query OK, 0 rows affected (10.08 sec)

mysql> insert into test values(100);
Query OK, 1 row affected (0.01 sec)

mysql>
```

在灾备上查看测试数据

```
Executed_Gtid_Set: 171a636c-27c5-11ec-9cff-08002760df88:1-669,
74c3c2a6-27dd-11ec-bfd3-08002760df88:1-645
  Auto_Position: 1
  Replicate_Rewrite_DB:
    Channel_Name: mgrpgold2mgrpbronze-semirepl
  Source_TLS_Version:
  Source_public_key_path:
  Get_Source_public_key: 0
  Network_Namespace:
1 row in set (0.00 sec)

mysql> use ohsdba
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> desc test;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| id    | int  | NO   | PRI | NULL    |       |
+-----+-----+-----+-----+-----+
1 row in set (0.06 sec)

mysql> select * from test;
+-----+
| id |
+-----+
| 100 |
+-----+
1 row in set (0.02 sec)

mysql>
```

在灾备上为复制通道设置多个数据源

```
select asynchronous_connection_failover_add_source('mgrpgold2mgrpbronze-semirepl','192.168.10.232',3306,null,60);  
select asynchronous_connection_failover_add_source('mgrpgold2mgrpbronze-semirepl','192.168.10.233',3306,null,60);
```

```
mysql>  
mysql> select asynchronous_connection_failover_add_source('mgrpgold2mgrpbronze-semirepl','192.168.10.232',3306,null,60);  
+-----+  
| asynchronous_connection_failover_add_source('mgrpgold2mgrpbronze-semirepl','192.168.10.232',3306,null,60) |  
+-----+  
| The UDF asynchronous_connection_failover_add_source() executed successfully. |  
+-----+  
1 row in set (0.05 sec)  
  
mysql> select asynchronous_connection_failover_add_source('mgrpgold2mgrpbronze-semirepl','192.168.10.233',3306,null,60);  
+-----+  
| asynchronous_connection_failover_add_source('mgrpgold2mgrpbronze-semirepl','192.168.10.233',3306,null,60) |  
+-----+  
| The UDF asynchronous_connection_failover_add_source() executed successfully. |  
+-----+  
1 row in set (0.01 sec)  
  
mysql>
```

在生产 MGR 上做主备切换

```
var ohsmgr=dba.getCluster()  
ohsmgr.setPrimaryInstance('pmydb02:3306');
```

```

MySQL 192.168.10.231:3306 ssl JS > var ohsmgr=dba.getCluster()
MySQL 192.168.10.231:3306 ssl JS > ohsmgr.setPrimaryInstance('pmydb02:3306');
Setting instance 'pmydb02:3306' as the primary instance of cluster 'ohsCluster'...

Instance 'pmydb01:3306' was switched from PRIMARY to SECONDARY.
Instance 'pmydb02:3306' was switched from SECONDARY to PRIMARY.
Instance 'pmydb03:3306' remains SECONDARY.

WARNING: The cluster internal session is not the primary member anymore. For cluster management operations please obtain a fresh cluster handle using dba.getCluster().

The instance 'pmydb02:3306' was successfully elected as primary.
MySQL 192.168.10.231:3306 ssl JS >

```

在生产上插入测试数据

```

mysql> insert into test values(200);
Query OK, 1 row affected (0.01 sec)

mysql> insert into test values(300);
ERROR 1290 (HY000): The MySQL server is running with the --super-read-only option so it cannot execute this statement
mysql> exit
Bye
[root@pmydb01 ~]#
[root@pmydb01 ~]# mysql -h192.168.10.232 -uroot -pohsdba -P3306
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 66
Server version: 8.0.26 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use ohsdba;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> insert into test values(300);
Query OK, 1 row affected (0.00 sec)

mysql>

```

在灾备上查看数据。设置多数据源后，即使主库做主备切换，半同步复制也是不受影响的。

```
mysql> start replica;  
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> select * from test;  
+-----+  
| id |  
+-----+  
| 100 |  
| 200 |  
+-----+  
2 rows in set (0.00 sec)
```

```
mysql> select * from test;  
+-----+  
| id |  
+-----+  
| 100 |  
| 200 |  
| 300 |  
+-----+  
3 rows in set (0.00 sec)
```

```
mysql>
```

在单机上手动创建 MGR

创建用户和目录

```
mkdir -p /pgold/mysql/product;cd /pgold;mkdir -p data/{mgr01,mgr02,mgr03,mgr04} log/{binlog,relaylog} log/binlog/{mgr01,mgr02,mgr03,mgr04}
log/relaylog/{mgr01,mgr02,mgr03,mgr04}
groupadd -g 27 -o -r mysql
useradd -M -N -g mysql -o -r -d /pgold/mysql -s /bin/false -c "MySQL Server" -u 27 mysql
chown -R mysql:mysql /pgold
```

```
[root@pgeneric ~]# mkdir -p /pgold/mysql/product;cd /pgold;mkdir -p data/{mgr01,mgr02,mgr03,mgr04} log/{binlog,relaylog} log/binlog/{mgr01,mgr02,mgr03,mgr04}
log/relaylog/{mgr01,mgr02,mgr03,mgr04}
[root@pgeneric pgold]# groupadd -g 27 -o -r mysql
[root@pgeneric pgold]# useradd -M -N -g mysql -o -r -d /pgold/mysql -s /bin/false -c "MySQL Server" -u 27 mysql
[root@pgeneric pgold]# chown -R mysql:mysql /pgold
```

临时 my.cnf 参数

```
[root@pmydb ~]# cat /tmp/my3306.cnf
[mysqld]
server_id=3306
port=3306
socket=/pgold/data/mgr01/mysql.sock
datadir=/pgold/data/mgr01
log_bin=/pgold/log/binlog/mgr01/mgr01_binlog
```

```
relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin
log-error=/pgold/data/mgr01/error.log
[root@pmydb ~]#
[root@pmydb ~]# cat /tmp/my3307.cnf
[mysqld]
server_id=3307
port=3307
socket=/pgold/data/mgr02/mysql.sock
datadir=/pgold/data/mgr02
log_bin=/pgold/log/binlog/mgr02/mgr02_binlog
relay_log=/pgold/log/relaylog/mgr02/mgr02_relaybin
log-error=/pgold/data/mgr02/error.log
[root@pmydb ~]#
[root@pmydb ~]# cat /tmp/my3308.cnf
[mysqld]
server_id=3308
port=3308
socket=/pgold/data/mgr03/mysql.sock
datadir=/pgold/data/mgr03
log_bin=/pgold/log/binlog/mgr03/mgr03_binlog
relay_log=/pgold/log/relaylog/mgr03/mgr03_relaybin
log-error=/pgold/data/mgr03/error.log
[root@pmydb ~]#
```

```
[root@pmydb ~]# cat /tmp/my3306.cnf
[mysqld]
server_id=3306
port=3306
socket=/pgold/data/mgr01/mysql.sock
datadir=/pgold/data/mgr01
log_bin=/pgold/log/binlog/mgr01/mgr01_binlog
relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin
log-error=/pgold/data/mgr01/error.log
[root@pmydb ~]#
[root@pmydb ~]# cat /tmp/my3307.cnf
[mysqld]
server_id=3307
port=3307
socket=/pgold/data/mgr02/mysql.sock
datadir=/pgold/data/mgr02
log_bin=/pgold/log/binlog/mgr02/mgr02_binlog
relay_log=/pgold/log/relaylog/mgr02/mgr02_relaybin
log-error=/pgold/data/mgr02/error.log
[root@pmydb ~]#
[root@pmydb ~]# cat /tmp/my3308.cnf
[mysqld]
server_id=3308
port=3308
socket=/pgold/data/mgr03/mysql.sock
datadir=/pgold/data/mgr03
log_bin=/pgold/log/binlog/mgr03/mgr03_binlog
relay_log=/pgold/log/relaylog/mgr03/mgr03_relaybin
log-error=/pgold/data/mgr03/error.log
[root@pmydb ~]#
```

初始化 MySQL

```
[root@pmydb ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/tmp/my3306.cnf --initialize --user=mysql
```

```
[root@pmydb ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/tmp/my3307.cnf --initialize --user=mysql
[root@pmydb ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/tmp/my3308.cnf --initialize --user=mysql
[root@pmydb ~]# grep -i password /pgold/data/mgr01/error.log
2021-10-04T15:39:47.742634Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: 0t?#THuyEToD
[root@pmydb ~]#
[root@pmydb ~]# grep -i password /pgold/data/mgr02/error.log
2021-10-04T15:40:03.226877Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: 3GXg4VIfzP.A
[root@pmydb ~]#
[root@pmydb ~]# grep -i password /pgold/data/mgr03/error.log
2021-10-04T15:40:16.832843Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: Mmiq#_y=t5do
[root@pmydb ~]#
```

```
[root@pmydb ~]#
[root@pmydb ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/tmp/my3306.cnf --initialize --user=mysql
[root@pmydb ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/tmp/my3307.cnf --initialize --user=mysql
[root@pmydb ~]# /pgold/mysql/product/8026/bin/mysqld --defaults-file=/tmp/my3308.cnf --initialize --user=mysql
[root@pmydb ~]#
```

```
[root@pmydb ~]# grep -i password /pgold/data/mgr01/error.log
2021-10-04T15:39:47.742634Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: 0t?#THuyEToD
[root@pmydb ~]#
[root@pmydb ~]# grep -i password /pgold/data/mgr02/error.log
2021-10-04T15:40:03.226877Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: 3GXg4VIfzP.A
[root@pmydb ~]#
[root@pmydb ~]# grep -i password /pgold/data/mgr03/error.log
2021-10-04T15:40:16.832843Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: Mmiq#_y=t5do
```

合并 my.cnf

合并非必须，目的是为了通过 mysqld_multi 方便管理。

```
[root@pmydb ~]# cat /etc/my.cnf
[mysqld]
basedir=/pgold/mysql/product/8026
log_timestamps=system
user = mysql
log_error_verbosity = 3
innodb_log_file_size = 500M
innodb_log_files_in_group = 5
binlog-format=row
log_slave_updates=on
binlog_checksum=crc32
character_set_server=utf8mb4

master-info-repository=table
relay-log-info-repository=table
gtid-mode=on
enforce-gtid-consistency=true
binlog_transaction_dependency_tracking=writeset
transaction_write_set_extraction=xxhash64
slave_parallel_type=logical_clock
slave_parallel_workers=128
sql_require_primary_key=1
```

```
slave_preserve_commit_order=1
slave_checkpoint_period=2

loose-plugin_load_add='mysql_clone.so'
loose-plugin_load_add='group_replication.so'
loose-group_replication_group_name="aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa"
loose-group_replication_group_seeds= "127.0.0.1:33306,127.0.0.1:33307,127.0.0.1:33308"
loose-group_replication_start_on_boot=off
loose-group_replication_bootstrap_group=off
loose-group_replication_exit_state_action=read_only
loose-group_replication_flow_control_mode = "disabled"
loose-group_replication_single_primary_mode=on

[mysqld_multi]
mysqld = /pgold/mysql/product/8026/bin/mysqld
log = /pgold/data/mysqld_multi.log
mysqladmin = /pgold/mysql/product/8026/bin/mysqladmin
user=root

[mysqld33306]
server_id=3306
port=3306
socket=/pgold/data/mgr01/mysql.sock
datadir=/pgold/data/mgr01
log_bin=/pgold/log/binlog/mgr01/mgr01_binlog
relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin
```

```
log-error=/pgold/data/mgr01/error.log  
loose-group_replication_local_address= "127.0.0.1:33306"
```

```
[mysqld33307]  
server_id=3307  
port=3307  
socket=/pgold/data/mgr02/mysql.sock  
datadir=/pgold/data/mgr02  
log_bin=/pgold/log/binlog/mgr02/mgr02_binlog  
relay_log=/pgold/log/relaylog/mgr02/mgr02_relaybin  
log-error=/pgold/data/mgr02/error.log  
loose-group_replication_local_address= "127.0.0.1:33307"
```

```
[mysqld33308]  
server_id=3308  
port=3308  
socket=/pgold/data/mgr03/mysql.sock  
datadir=/pgold/data/mgr03  
log_bin=/pgold/log/binlog/mgr03/mgr03_binlog  
relay_log=/pgold/log/relaylog/mgr03/mgr03_relaybin  
log-error=/pgold/data/mgr03/error.log  
loose-group_replication_local_address= "127.0.0.1:33308"  
[root@pmydb ~]#
```


启动多实例

```
[root@pmydb ~]# mysqld_multi start 33306
[root@pmydb ~]# ps -ef|grep mysql
mysql    17717      1 11 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3306 --port=3306 --
socket=/pgold/data/mgr01/mysql.sock --datadir=/pgold/data/mgr01 --log_bin=/pgold/log/binlog/mgr01/mgr01_binlog --
relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin --log-error=/pgold/data/mgr01/error.log --loose-
group_replication_local_address=127.0.0.1:33306
root     17847 11579   0 23:48 pts/0    00:00:00 grep --color=auto mysql
[root@pmydb ~]#
[root@pmydb ~]# mysqld_multi start 33307
[root@pmydb ~]# mysqld_multi start 33308
[root@pmydb ~]# ps -ef|grep mysql
mysql    17717      1  3 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3306 --port=3306 --
socket=/pgold/data/mgr01/mysql.sock --datadir=/pgold/data/mgr01 --log_bin=/pgold/log/binlog/mgr01/mgr01_binlog --
relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin --log-error=/pgold/data/mgr01/error.log --loose-
group_replication_local_address=127.0.0.1:33306
mysql    17955      1  7 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3307 --port=3307 --
socket=/pgold/data/mgr02/mysql.sock --datadir=/pgold/data/mgr02 --log_bin=/pgold/log/binlog/mgr02/mgr02_binlog --
relay_log=/pgold/log/relaylog/mgr02/mgr02_relaybin --log-error=/pgold/data/mgr02/error.log --loose-
group_replication_local_address=127.0.0.1:33307
mysql    18093      1 27 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3308 --port=3308 --
socket=/pgold/data/mgr03/mysql.sock --datadir=/pgold/data/mgr03 --log_bin=/pgold/log/binlog/mgr03/mgr03_binlog --
relay_log=/pgold/log/relaylog/mgr03/mgr03_relaybin --log-error=/pgold/data/mgr03/error.log --loose-
group_replication_local_address=127.0.0.1:33308
```

```
root      18175 11579  0 23:48 pts/0    00:00:00 grep --color=auto mysql
```

```
[root@pmydb ~]#
```

```
[root@pmydb ~]# mysqld_multi start 33306
[root@pmydb ~]# ps -ef|grep mysql
mysql    17717      1 11 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3306 --port=3306 --socket=/pgold/data/mgr01/mysql.sock --d
atadir=/pgold/data/mgr01 --log_bin=/pgold/log/binlog/mgr01/mgr01_binlog --relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin --log-error=/pgold/data/mgr01/
error.log --loose-group_replication_local_address=127.0.0.1:33306
root     17847 11579  0 23:48 pts/0    00:00:00 grep --color=auto mysql
[root@pmydb ~]#
[root@pmydb ~]# mysqld_multi start 33307
[root@pmydb ~]# mysqld_multi start 33308
[root@pmydb ~]# ps -ef|grep mysql
mysql    17717      1 3 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3306 --port=3306 --socket=/pgold/data/mgr01/mysql.sock --d
atadir=/pgold/data/mgr01 --log_bin=/pgold/log/binlog/mgr01/mgr01_binlog --relay_log=/pgold/log/relaylog/mgr01/mgr01_relaybin --log-error=/pgold/data/mgr01/
error.log --loose-group_replication_local_address=127.0.0.1:33306
mysql    17955      1 7 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3307 --port=3307 --socket=/pgold/data/mgr02/mysql.sock --d
atadir=/pgold/data/mgr02 --log_bin=/pgold/log/binlog/mgr02/mgr02_binlog --relay_log=/pgold/log/relaylog/mgr02/mgr02_relaybin --log-error=/pgold/data/mgr02/
error.log --loose-group_replication_local_address=127.0.0.1:33307
mysql    18093      1 27 23:48 pts/0    00:00:01 /pgold/mysql/product/8026/bin/mysqld --server_id=3308 --port=3308 --socket=/pgold/data/mgr03/mysql.sock --d
atadir=/pgold/data/mgr03 --log_bin=/pgold/log/binlog/mgr03/mgr03_binlog --relay_log=/pgold/log/relaylog/mgr03/mgr03_relaybin --log-error=/pgold/data/mgr03/
error.log --loose-group_replication_local_address=127.0.0.1:33308
root     18175 11579  0 23:48 pts/0    00:00:00 grep --color=auto mysql
[root@pmydb ~]#
```

创建复制用户

在每个实例上都执行

```
mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
```

```
mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
```

```
mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
```

```
set sql_log_bin=0;
```

```
alter user current_user() identified by 'root';
```

```
create user repl identified with mysql_native_password by 'ohsdba';  
grant backup_admin,replication slave on *.* to repl;  
set sql_log_bin=1;
```

```
[root@pmydb pgold]# mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
```

```
Enter password:
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
```

```
Your MySQL connection id is 8
```

```
Server version: 8.0.26
```

```
Copyright (c) 2000, 2021, Oracle and/or its affiliates.
```

```
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql> set sql_log_bin=0;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> alter user current_user() identified by 'root';
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> create user repl identified with mysql_native_password by 'ohsdba';
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> grant backup_admin,replication slave on *.* to repl;
Query OK, 0 rows affected (0.01 sec)

mysql> set sql_log_bin=1;
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql>
[1]+  Stopped                  mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
[root@pmydb pgold]#
[root@pmydb pgold]# mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.26

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> set sql_log_bin=0;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> alter user current_user() identified by 'root';
Query OK, 0 rows affected (0.01 sec)

mysql> create user repl identified with mysql_native_password by 'ohsdba';
Query OK, 0 rows affected (0.01 sec)

mysql> grant backup_admin,replication slave on *.* to repl;
Query OK, 0 rows affected (0.00 sec)

mysql> set sql_log_bin=1;
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql>
[2]+  Stopped                  mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
[root@pmydb pgold]# mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.26

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
```

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> set sql_log_bin=0;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> alter user current_user() identified by 'root';
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> create user repl identified with mysql_native_password by 'ohsdba';
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> grant backup_admin,replication slave on *.* to repl;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> set sql_log_bin=1;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql>
```

```
mysql>
```

```
[3]+ Stopped mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
```

```
[root@pmydb pgold]#
```

```
[root@pmydb pgold]# jobs
```

```
[1] Stopped mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
```

```
[2]- Stopped mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
```

```
[3]+ Stopped          mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
[root@pmydb pgold]#
```

```
mysql> set sql_log_bin=0;
Query OK, 0 rows affected (0.00 sec)

mysql> alter user current_user() identified by 'root';
Query OK, 0 rows affected (0.01 sec)

mysql> create user repl identified with mysql_native_password by 'ohsdba';
Query OK, 0 rows affected (0.01 sec)

mysql> grant backup_admin,replication slave on *.* to repl;
Query OK, 0 rows affected (0.01 sec)

mysql> set sql_log_bin=1;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

设置复制通道

在每个实例上执行

```
[root@pmydb pgold]# fg 1
mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
mysql> change master to master_user='repl',master_password='ohsdba' for channel 'group_replication_recovery';
```

```
Query OK, 0 rows affected, 5 warnings (0.02 sec)
```

```
mysql>
```

```
mysql>
```

```
[1]+  Stopped                  mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
```

```
[root@pmydb pgold]# fg 2
```

```
mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
```

```
mysql> change master to master_user='repl',master_password='ohsdba' for channel 'group_replication_recovery';
```

```
Query OK, 0 rows affected, 5 warnings (0.02 sec)
```

```
mysql>
```

```
mysql>
```

```
[2]+  Stopped                  mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
```

```
[root@pmydb pgold]#
```

```
[root@pmydb pgold]# fg 3
```

```
mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
```

```
mysql> change master to master_user='repl',master_password='ohsdba' for channel 'group_replication_recovery';
```

```
Query OK, 0 rows affected, 5 warnings (0.03 sec)
```

```
mysql>
```

```
mysql>
```

```
[3]+  Stopped                  mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
```

```
[root@pmydb pgold]#
```



```
[root@pmydb pgold]# fg 1
mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
mysql> change master to master_user='repl',master_password='ohsdba' for channel 'group_replication_recovery';
Query OK, 0 rows affected, 5 warnings (0.02 sec)

mysql>
mysql>
[1]+  Stopped                  mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
[root@pmydb pgold]# fg 2
mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
mysql> change master to master_user='repl',master_password='ohsdba' for channel 'group_replication_recovery';
Query OK, 0 rows affected, 5 warnings (0.02 sec)

mysql>
mysql>
[2]+  Stopped                  mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
[root@pmydb pgold]#
[root@pmydb pgold]# fg 3
mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
mysql> change master to master_user='repl',master_password='ohsdba' for channel 'group_replication_recovery';
Query OK, 0 rows affected, 5 warnings (0.03 sec)

mysql>
mysql>
[3]+  Stopped                  mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
[root@pmydb pgold]#
```

设置引导节点

在主节点上执行

```
set global group_replication_bootstrap_group=on;
start group_replication;
set global group_replication_bootstrap_group=off;
```

```
select * from performance_schema.replication_group_members;
```

```
mysql -uroot -S/pgold/data/mgr01/mysql.sock -p
mysql> set global group_replication_bootstrap_group=on;
Query OK, 0 rows affected (0.00 sec)

mysql> start group_replication;
Query OK, 0 rows affected (2.48 sec)

mysql> set global group_replication_bootstrap_group=off;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 252145b4-2536-11ec-8795-080027e700aa | pmydb      | 3306        | ONLINE       | PRIMARY     | 8.0.26         |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

加入其它节点

在第二个节点执行

```
start group_replication;
```

```
mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
mysql> start group_replication;
Query OK, 0 rows affected (3.61 sec)

mysql> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 252145b4-2536-11ec-8795-080027e700aa | pmydb      | 3306        | ONLINE       | PRIMARY     | 8.0.26         |
| group_replication_applier | 2c88fc35-2536-11ec-8d14-080027e700aa | pmydb      | 3307        | ONLINE       | SECONDARY   | 8.0.26         |
+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

在第三个节点执行

start group_replication;

```
mysql -uroot -S/pgold/data/mgr03/mysql.sock -p
mysql> start group_replication;
Query OK, 0 rows affected (2.74 sec)

mysql> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 252145b4-2536-11ec-8795-080027e700aa | pmydb       | 3306        | ONLINE       | PRIMARY     | 8.0.26         |
| group_replication_applier | 2c88fc35-2536-11ec-8d14-080027e700aa | pmydb       | 3307        | ONLINE       | SECONDARY   | 8.0.26         |
| group_replication_applier | 338ed6f9-2536-11ec-92fe-080027e700aa | pmydb       | 3308        | ONLINE       | SECONDARY   | 8.0.26         |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

你可能会碰到下面的问题

如果修改 root 密码的时候，没有使用 set sql_log_bin=0;

在第二个节点加入时会出现下面的异常，因为修改 root 密码也是一个事务。通过查看 binlog 确认是修改 root 密码造成的。可以通过 reset master，然后再次启动复制就好。

```
mysql> start group_replication;
ERROR 3092 (HY000): The server is not configured properly to be an active member of the group. Please see more details on error log.
mysql>
```

```
2021-10-05T00:36:56.782890+08:00 0 [Note] [MY-011735] [Repl] Plugin group_replication reported: '[GCS] Group is able to support up to communication protocol version 8.0.16'
```

```
2021-10-05T00:36:56.790233+08:00 0 [ERROR] [MY-011526] [Repl] Plugin group_replication reported: 'This member has more executed transactions than those present in the group. Local transactions: d979b196-252f-11ec-8d8f-080027e700aa:1 > Group transactions: aaaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa:1, d217fba-252f-11ec-8694-080027e700aa:1'
```

```
2021-10-05T00:36:56.790330+08:00 0 [ERROR] [MY-011522] [Repl] Plugin group_replication reported: 'The member contains transactions not present in the group. The member will now exit the group.'
```

```
2021-10-05T00:36:56.790410+08:00 0 [System] [MY-011503] [Repl] Plugin group_replication reported: 'Group membership changed to pmydb:3306, pmydb:3307 on view 16333653736073684:2.'
```

```
2021-10-05T00:36:56.790498+08:00 8 [Note] [MY-011647] [Repl] Plugin group_replication reported: 'Going to wait for view modification'
```

```
2021-10-05T00:36:56.791345+08:00 0 [Note] [MY-011735] [Repl] Plugin group_replication reported: '[GCS] Re-using server node 0 host 127.0.0.1'
```

```
2021-10-05T00:36:56.791370+08:00 0 [Note] [MY-011735] [Repl] Plugin group_replication reported: '[GCS] pid 4023 Installed site start={4317e324 95 1} boot_key={4317e324 84 1} event_horizon=10 node 4294967295 chksum_node_list(&site->nodes) 1981123465'
```

```
2021-10-05T00:37:00.316641+08:00 0 [System] [MY-011504] [Repl] Plugin group_replication reported: 'Group membership changed: This member has left the group.'
```

```
2021-10-05T00:37:00.319611+08:00 13 [Note] [MY-010596] [Repl] Error reading relay log event for channel 'group_replication_applier': slave SQL thread was killed
```

```
2021-10-05T00:37:00.329066+08:00 13 [Note] [MY-010587] [Repl] Slave SQL thread for channel 'group_replication_applier' exiting, replication stopped in log 'FIRST' at position 0
```

```
2021-10-05T00:37:00.331197+08:00 10 [Note] [MY-011444] [Repl] Plugin group_replication reported: 'The group replication applier thread was killed.'
```

```
2021-10-05T00:37:00.331454+08:00 9 [System] [MY-011566] [Repl] Plugin group_replication reported: 'Setting super read only=OFF.'
```

```

[root@pmydb ~]# mysqlbinlog /pgold/log/binlog/mgr02/mgr02_binlog.000002|more
# The proper term is pseudo_replica_mode, but we use this compatibility alias
# to make the statement usable on server versions 8.0.24 and older.
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!50003 SET @OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#211005 0:30:42 server id 3307  end_log_pos 125 CRC32 0x66aa71c3          Start: binlog v 4, server v 8.0.26 created 211005 0:30:42 at startup
# Warning: this binlog is either in use or was not closed properly.
ROLLBACK/*!*/;
BINLOG '
MixbYQ/rDAAaeQAAAH0AAAABAAQAOC4wLjI2AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAyLFthEwANAAGAAAABAAEAAAAYQAEgGgAAAaICAgCAAAACGgOKKioAEjQA
CigBw3GqZg==
'/*!*/;
# at 125
#211005 0:30:42 server id 3307  end_log_pos 156 CRC32 0x49859ab7          Previous-GTIDs
# [empty]
# at 156
#211005 0:32:57 server id 3307  end_log_pos 235 CRC32 0x81fa1f79          GTID      last_committed=0          sequence_number=1          rbr_only=no          original_co
mmitted_timestamp=1633365177637904          immediate_commit_timestamp=1633365177637904          transaction_length=337
# original_commit_timestamp=1633365177637904 (2021-10-05 00:32:57.637904 CST)
# immediate_commit_timestamp=1633365177637904 (2021-10-05 00:32:57.637904 CST)
/*!80001 SET @@session.original_commit_timestamp=1633365177637904/*!*/;
/*!80014 SET @@session.original_server_version=80026/*!*/;
/*!80014 SET @@session.immediate_server_version=80026/*!*/;
SET @@SESSION.GTID_NEXT= 'd979b196-252f-11ec-8d8f-080027e700aa:1'/*!*/;
# at 235
#211005 0:32:57 server id 3307  end_log_pos 493 CRC32 0xc99f300f          Query      thread_id=8          exec_time=0          error_code=0          Xid = 3
SET TIMESTAMP=1633365177.630412/*!*/;
SET @@session.pseudo_thread_id=8/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0, @@session.unique_checks=1, @@session.autocommit=1/*!*/;
SET @@session.sql_mode=1168113696/*!*/;
SET @@session.auto_increment_increment=1, @@session.auto_increment_offset=1/*!*/;
/*!\C utf8mb4 *//*!*/;
SET @@session.character_set_client=255,@@session.collation_connection=255,@@session.collation_server=255/*!*/;
SET @@session.time_zone='SYSTEM'/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
/*!80011 SET @@session.default_collation_for_utf8mb4=255/*!*/;
ALTER USER 'root'@'localhost' IDENTIFIED WITH 'caching_sha2_password' AS '$A$005$1y]J`S}O}D;\Z?C\r&pF7DU4TJYeI3D/1hCn5qPcRiDujMnbs/7TTqpzIKXq0'
/*!*/;
SET @@SESSION.GTID_NEXT= 'AUTOMATIC' /* added by mysqlbinlog */ /*!*/;

```

```
mysql -uroot -S/pgold/data/mgr02/mysql.sock -p (wd: ~)
mysql> start group_replication;
ERROR 3092 (HY000): The server is not configured properly to be an active member of the group. Please see more details on error log.
mysql>
mysql> reset master;
Query OK, 0 rows affected (0.01 sec)

mysql> start group_replication;
Query OK, 0 rows affected (3.61 sec)

mysql> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 4c927abd-2529-11ec-bc5e-080027e700aa | pmydb      | 3306 | ONLINE | PRIMARY | 8.0.26 |
| group_replication_applier | 55cf42dd-2529-11ec-8496-080027e700aa | pmydb      | 3307 | ONLINE | SECONDARY | 8.0.26 |
+-----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.01 sec)
```

```
mysql -uroot -S/pgold/data/mgr03/mysql.sock -p (wd: ~)
mysql>
mysql> start group_replication;
ERROR 3092 (HY000): The server is not configured properly to be an active member of the group. Please see more details on error log.
mysql> reset master;
Query OK, 0 rows affected (0.02 sec)

mysql> start group_replication;
Query OK, 0 rows affected (3.43 sec)

mysql> select * from performance_schema.replication_group_members;
+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME          | MEMBER_ID          | MEMBER_HOST | MEMBER_PORT | MEMBER_STATE | MEMBER_ROLE | MEMBER_VERSION |
+-----+-----+-----+-----+-----+-----+-----+
| group_replication_applier | 4c927abd-2529-11ec-bc5e-080027e700aa | pmydb      | 3306 | ONLINE | PRIMARY | 8.0.26 |
| group_replication_applier | 55cf42dd-2529-11ec-8496-080027e700aa | pmydb      | 3307 | ONLINE | SECONDARY | 8.0.26 |
| group_replication_applier | 5dcd1f49-2529-11ec-8ae5-080027e700aa | pmydb      | 3308 | ONLINE | SECONDARY | 8.0.26 |
+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

创建测试数据

每张需要被组复制的表都必须显式定义一个主键。主键在判断事务是否冲突扮演极其重要的角色，通过主键来准确识别每个事务中修改了表中的哪些行。否则在创建的时候会提示异常。

```
mysql>
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.01 sec)

mysql> create database ohsdba;
Query OK, 1 row affected (0.01 sec)

mysql> use ohsdba;
Database changed
mysql> create table test(a int);
ERROR 3750 (HY000): Unable to create or change a table without a primary key, when the system variable 'sql_require_primary_key' is set. Add a primary key to the table or unset this variable to avoid this message. Note that tables without a primary key can cause performance problems in row-based replication, so please consult your DBA before changing this setting.
mysql>
mysql> create table test(a int not null primary key);
Query OK, 0 rows affected (0.03 sec)

mysql> insert into test values(1);
Query OK, 1 row affected (0.03 sec)

mysql>
```

在只读的节点上查看数据，数据已经同步过来了。

```
[root@pmydb ~]# mysql -uroot -S/pgold/data/mgr02/mysql.sock -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 422
Server version: 8.0.26 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use ohsdba
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from test;
+----+
| a |
+----+
| 1 |
+----+
1 row in set (0.00 sec)

mysql>
mysql> create table test2(a int not null primary key);
ERROR 1290 (HY000): The MySQL server is running with the --super-read-only option so it cannot execute this statement
mysql>
```