# MySQL Group Replication Single-Primary Mode（单主）搭建部署过程

# MySQL Group Replication Single-Primary Mode（单主）搭建部署过程

## 环境介绍：

3台linux服务器

IP地址：192.168.1.20（节点1）、192.168.4.20（节点2）、192.168.4.21（节点3）

部署前准备：

检查所有机器是否安装mysql

ps -ef |grep mysql

卸载mysql

yum erase Percona\*

安装mysql5.7.18（因为组复制是MySQL Server 5.7.17及更高版本提供的内置MySQL插件）

yum install Percona-Server-server-57

## 参数说明

Group Replication 相关设置  官方文档：<https://dev.mysql.com/doc/refman/5.7/en/group-replication-configuring-instances.html>

**Group Replication Settings**

report\_host = 192.168.1.20

transaction\_write\_set\_extraction = XXHASH64 #主键采集功能

loose-group\_replication\_group\_name = "ec2ff1f6-893a-11e7-b476-288023af50f8" #组的名字,需要一致新成员添加不进来

loose-group\_replication\_local\_address = "192.168.1.20:33061" #成员本地地址

loose-group\_replication\_group\_seeds = "192.168.4.20:33061,192.168.4.21:33061" #设置种子成员地址 通信需要知道至少一个当前组内成员的地址，即其他成员的local\_address

loose-group\_replication\_bootstrap\_group = off

loose-group\_replication\_start\_on\_boot = off

loose-group\_replication\_ip\_whitelist = '192.168.1.20,192.168.4.20,192.168.4.21'

loose-group\_replication\_auto\_increment\_increment = 1

参数说明补充：

**参数补充说明**

loose-group\_replication\_bootstrap\_group：#初始化第一个成员时使用，初始化后需要将其关闭，如果您多次引导组，例如，当多个服务器实例设置此选项时，它们可能会创建一个人造裂脑情景，其中存在两个具有相同名称的不同组。

loose-group\_replication\_start\_on\_boot：指示插件在服务器启动时不会自动启动操作。

loose-group\_replication\_ip\_whitelist ： 设置成员IP白名单，通过白名单来控制哪些IP地址的MySQL服务器可以加入到组里来。其值可以是地址列表，具体IP，网段。地址之间用 “ , ” 逗号隔开。不设置其会自动识别本机网口上配置的私网地址和私网网段，127.0.0.1的连接一直被允许。配置时需要关闭group\_replication。

loose-group\_replication\_auto\_increment\_increment ： 设置自增步长，单主模式下该值默认为7，offset默认为server\_id。

以下设置根据MySQL组复制要求配置复制：

**The following settings configure replication according to the MySQL Group Replication requirements.**

server\_id=3232235796

gtid\_mode=ON

enforce\_gtid\_consistency=ON

master\_info\_repository=TABLE

relay\_log\_info\_repository=TABLE

binlog\_checksum=NONE

log\_slave\_updates=ON

log\_bin=mysql-bin.log

binlog\_format=ROW

## 部署过程：

1.my.cnf的配置文件

my.cnf for 192.168.1.20

**my.cnf for 192.168.1.20**  Expand source

[mysqld\_safe]

pid-file =/data/mysqldata/mysql.pid

[mysqld]

# GENERAL

#basedir = /usr/local/mysql

datadir = /data/mysqldata

tmpdir = /tmp

socket = /data/mysqldata/mysql.sock

pid\_file = /data/mysqldata/mysql.pid

user = mysql

port = 3306

character-set-server = utf8

bind-address = 0.0.0.0

server-id = 1490858127

skip-name-resolve = 1

# INNODB

# This changes how |InnoDB| autoincrement locks are managed and is a requirement for Galera

default\_storage\_engine = InnoDB

innodb\_buffer\_pool\_size = 50G

innodb\_autoinc\_lock\_mode = 2

innodb\_buffer\_pool\_instances = 8

innodb\_thread\_concurrency = 16

innodb\_log\_buffer\_size = 32M

innodb\_log\_file\_size = 1024M

innodb\_online\_alter\_log\_max\_size = 512M

innodb\_open\_files = 1024

innodb\_purge\_threads = 2

innodb\_data\_home\_dir = /data/mysqldata

innodb\_data\_file\_path = ibdata1:256M:autoextend

innodb\_read\_io\_threads = 8

innodb\_write\_io\_threads = 8

innodb\_file\_per\_table = 1

innodb\_flush\_method = O\_DIRECT

innodb\_flush\_log\_at\_trx\_commit = 2

innodb\_max\_dirty\_pages\_pct = 90

innodb\_file\_format = Barracuda

innodb\_file\_format\_max = Barracuda

innodb\_buffer\_pool\_dump\_at\_shutdown = ON

innodb\_buffer\_pool\_load\_at\_startup = ON

innodb\_undo\_log\_truncate = ON

innodb\_undo\_tablespaces = 4

innodb\_strict\_mode = OFF

# MyISAM

key\_buffer\_size = 32M

# LOGS

#general\_log = 1

#general\_log\_file = /data/logs/mysql/mysql\_general.log

log\_timestamps = system

log\_warnings = 2

log\_error = /data/logs/mysql/mysql\_error.log

slow\_query\_log = ON

slow\_query\_log\_file = /data/logs/mysql/mysql\_slow.log

log\_queries\_not\_using\_indexes = 0

long\_query\_time = 1

expire\_logs\_days = 15

log-bin = mysql-bin.log

innodb\_print\_all\_deadlocks = 1

relay-log = relay-log

relay-log-index = relay-log

# BINLOG

# In order for Galera to work correctly binlog format should be ROW

binlog\_format = ROW

binlog\_cache\_size = 16M

max\_binlog\_size = 512M

# OTHER

tmp\_table\_size = 32M

max\_heap\_table\_size = 128M

query\_cache\_type = 0

query\_cache\_size = 0M

max\_connections = 1024

thread\_cache\_size = 200

open\_files\_limit = 65535

optimizer-switch = "mrr=on,mrr\_cost\_based=off,batched\_key\_access=on" #开启mmr功能

auto\_increment\_offset = 1

join\_buffer\_size = 5M

sort\_buffer\_size = 5M

sql\_mode = STRICT\_TRANS\_TABLES,NO\_ENGINE\_SUBSTITUTION

performance\_schema = ON

default\_password\_lifetime = 0

# Replication

log\_slave\_updates = 1

gtid-mode = on

enforce\_gtid\_consistency = true

binlog\_checksum = NONE #disable binary log event checksums

slave\_allow\_batching = 1

master\_verify\_checksum = 1

slave\_sql\_verify\_checksum = 1

master\_info\_repository = TABLE

relay\_log\_info\_repository = TABLE

slave\_parallel\_type = LOGICAL\_CLOCK

slave\_parallel\_workers = 16

relay\_log\_recovery = ON

slave\_preserve\_commit\_order = ON

# Group Replication

report\_host = 192.168.1.20

transaction\_write\_set\_extraction = XXHASH64 #主键采集功能

loose-group\_replication\_group\_name = "ec2ff1f6-893a-11e7-b476-288023af50f8" #组的名字

loose-group\_replication\_local\_address = "192.168.1.20:33061" #成员本地地址

loose-group\_replication\_group\_seeds = "192.168.4.20:33061,192.168.4.21:33061" #官方推荐33061端口

loose-group\_replication\_bootstrap\_group = off

loose-group\_replication\_start\_on\_boot = off

loose-group\_replication\_ip\_whitelist = '192.168.1.20,192.168.4.20,192.168.4.21'

loose-group\_replication\_auto\_increment\_increment = 1

[client]

socket = /data/mysqldata/mysql.sock

port = 3306

[mysql]

default-character-set = utf8

prompt ="\\u@\\h : \\d \\R:\\m:\\s>"

no-auto-rehash

初始化并启动mysql：

mysqld --initialize

service mysql start

3.在每台机器上安装Group Replication插件：

INSTALL PLUGIN group\_replication SONAME 'group\_replication.so';

查看group replication插件，ACTIVE代表添加成功：

**group replication active**  Expand source

root@localhost : (none) 14:06:22>show plugins;

+-----------------------------+----------+--------------------+----------------------+---------+

| Name | Status | Type | Library | License |

+-----------------------------+----------+--------------------+----------------------+---------+

| binlog | ACTIVE | STORAGE ENGINE | NULL | GPL |

| mysql\_native\_password | ACTIVE | AUTHENTICATION | NULL | GPL |

| sha256\_password | ACTIVE | AUTHENTICATION | NULL | GPL |

| PERFORMANCE\_SCHEMA | ACTIVE | STORAGE ENGINE | NULL | GPL |

| MEMORY | ACTIVE | STORAGE ENGINE | NULL | GPL |

| MRG\_MYISAM | ACTIVE | STORAGE ENGINE | NULL | GPL |

| MyISAM | ACTIVE | STORAGE ENGINE | NULL | GPL |

| CSV | ACTIVE | STORAGE ENGINE | NULL | GPL |

| InnoDB | ACTIVE | STORAGE ENGINE | NULL | GPL |

| XTRADB\_READ\_VIEW | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| XTRADB\_INTERNAL\_HASH\_TABLES | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| XTRADB\_RSEG | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| XTRADB\_ZIP\_DICT | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| XTRADB\_ZIP\_DICT\_COLS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_TRX | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_LOCKS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_LOCK\_WAITS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CMP | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CMP\_RESET | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CMPMEM | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CMPMEM\_RESET | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CMP\_PER\_INDEX | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CMP\_PER\_INDEX\_RESET | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_BUFFER\_PAGE | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_BUFFER\_PAGE\_LRU | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_BUFFER\_POOL\_STATS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_TEMP\_TABLE\_INFO | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_METRICS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_FT\_DEFAULT\_STOPWORD | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_FT\_DELETED | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_FT\_BEING\_DELETED | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_FT\_CONFIG | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_FT\_INDEX\_CACHE | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_FT\_INDEX\_TABLE | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_TABLES | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_TABLESTATS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_INDEXES | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_COLUMNS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_FIELDS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_FOREIGN | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_FOREIGN\_COLS | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_TABLESPACES | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_DATAFILES | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_CHANGED\_PAGES | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| INNODB\_SYS\_VIRTUAL | ACTIVE | INFORMATION SCHEMA | NULL | GPL |

| partition | ACTIVE | STORAGE ENGINE | NULL | GPL |

| FEDERATED | DISABLED | STORAGE ENGINE | NULL | GPL |

| ARCHIVE | ACTIVE | STORAGE ENGINE | NULL | GPL |

| BLACKHOLE | ACTIVE | STORAGE ENGINE | NULL | GPL |

| ngram | ACTIVE | FTPARSER | NULL | GPL |

| group\_replication | ACTIVE | GROUP REPLICATION | group\_replication.so | GPL |

+-----------------------------+----------+--------------------+----------------------+---------+

51 rows in set (0.00 sec)

4.在每台机器上都创建复制用户并添加复制凭证，我们也可以继续使用现有的复制用户：

**Create group\_replication User**

SET SQL\_LOG\_BIN=0;

grant replication slave on \*.\* to 'replication'@'%' identified by '&UJM8ik,';

SET SQL\_LOG\_BIN=1;

CHANGE MASTER TO MASTER\_USER='replication', MASTER\_PASSWORD='&UJM8ik,' for channel 'group\_replication\_recovery';

5.启动MGR：

组初始化：192.168.1.20

**初始化组**

SET GLOBAL group\_replication\_bootstrap\_group=ON;

START GROUP\_REPLICATION;

SET GLOBAL group\_replication\_bootstrap\_group=OFF;

START GROUP\_REPLICATION语句返回后，组已启动。您可以检查该组是否已创建，并且其中有一个成员：

**online**

root@localhost : (none) 14:16:36>SELECT \* FROM performance\_schema.replication\_group\_members ;

+---------------------------+--------------------------------------+--------------+-------------+--------------+

| CHANNEL\_NAME | MEMBER\_ID | MEMBER\_HOST | MEMBER\_PORT | MEMBER\_STATE |

+---------------------------+--------------------------------------+--------------+-------------+--------------+

| group\_replication\_applier | 0098bd1f-8c9e-11e7-a58a-ecb1d7897858 | 192.168.1.20 | 3306 | ONLINE |

+---------------------------+--------------------------------------+--------------+-------------+--------------+

为了表明服务器确实在一个组中，并且能够处理加载，创建一个表并向其添加一些内容，并查看：

**create test date**

CREATE DATABASE test;

USE test;

CREATE TABLE t1 (c1 INT PRIMARY KEY, c2 TEXT NOT NULL);

INSERT INTO t1 VALUES (1, 'Luis');

mysql> SELECT \* FROM t1;

+----+------+

| c1 | c2 |

+----+------+

| 1 | Luis |

+----+------+

192.168.4.21。

1.配置配置文件需要修改以下几项即可：

server-id、report\_host、loose-group\_replication\_local\_address、loose-group\_replication\_group\_seeds

2.初始化并启动服务：

**mysql server start**

mysqld --initialize

service mysql start

3.配置group\_replication\_recovery通道的恢复凭据。

**通道配置**

SET SQL\_LOG\_BIN=0;

grant replication slave on \*.\* to 'replication'@'%' identified by '&UJM8ik,';

SET SQL\_LOG\_BIN=1;

CHANGE MASTER TO MASTER\_USER='replication', MASTER\_PASSWORD='&UJM8ik,' for channel 'group\_replication\_recovery';

4.安装组复制插件并启动它：

INSTALL PLUGIN group\_replication SONAME 'group\_replication.so';

START GROUP\_REPLICATION;

再次检查performance\_schema.replication\_group\_members表显示组中现在有两个ONLINE服务器：

root@localhost : (none) 15:04:29>SELECT \* FROM performance\_schema.replication\_group\_members ;

+---------------------------+--------------------------------------+--------------+-------------+--------------+

| CHANNEL\_NAME | MEMBER\_ID | MEMBER\_HOST | MEMBER\_PORT | MEMBER\_STATE |

+---------------------------+--------------------------------------+--------------+-------------+--------------+

| group\_replication\_applier | 0098bd1f-8c9e-11e7-a58a-ecb1d7897858 | 192.168.1.20 | 3306 | ONLINE |

| group\_replication\_applier | 0219385c-8ca3-11e7-bae0-288023af52c8 | 192.168.4.21 | 3306 | ONLINE |

+---------------------------+--------------------------------------+--------------+-------------+--------------+

验证复制状态：

 Expand source

mysql> SHOW DATABASES LIKE 'test';

+-----------------+

| Database (test) |

+-----------------+

| test |

+-----------------+

1 row in set (0,00 sec)

mysql> SELECT \* FROM test.t1;

+----+------+

| c1 | c2 |

+----+------+

| 1 | Luis |

+----+------+

1 row in set (0,00 sec)

mysql> SHOW BINLOG EVENTS;

+---------------+------+----------------+-----------+-------------+--------------------------------------------------------------------+

| Log\_name | Pos | Event\_type | Server\_id | End\_log\_pos | Info |

+---------------+------+----------------+-----------+-------------+--------------------------------------------------------------------+

| binlog.000001 | 4 | Format\_desc | 2 | 123 | Server ver: 5.7.17-log, Binlog ver: 4 |

| binlog.000001 | 123 | Previous\_gtids | 2 | 150 | |

| binlog.000001 | 150 | Gtid | 1 | 211 | SET @@SESSION.GTID\_NEXT= 'aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa:1' |

| binlog.000001 | 211 | Query | 1 | 270 | BEGIN |

| binlog.000001 | 270 | View\_change | 1 | 369 | view\_id=14724832985483517:1 |

| binlog.000001 | 369 | Query | 1 | 434 | COMMIT |

| binlog.000001 | 434 | Gtid | 1 | 495 | SET @@SESSION.GTID\_NEXT= 'aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa:2' |

| binlog.000001 | 495 | Query | 1 | 585 | CREATE DATABASE test |

| binlog.000001 | 585 | Gtid | 1 | 646 | SET @@SESSION.GTID\_NEXT= 'aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa:3' |

| binlog.000001 | 646 | Query | 1 | 770 | use `test`; CREATE TABLE t1 (c1 INT PRIMARY KEY, c2 TEXT NOT NULL) |

| binlog.000001 | 770 | Gtid | 1 | 831 | SET @@SESSION.GTID\_NEXT= 'aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa:4' |

| binlog.000001 | 831 | Query | 1 | 890 | BEGIN |

| binlog.000001 | 890 | Table\_map | 1 | 933 | table\_id: 108 (test.t1) |

| binlog.000001 | 933 | Write\_rows | 1 | 975 | table\_id: 108 flags: STMT\_END\_F |

| binlog.000001 | 975 | Xid | 1 | 1002 | COMMIT /\* xid=30 \*/ |

| binlog.000001 | 1002 | Gtid | 1 | 1063 | SET @@SESSION.GTID\_NEXT= 'aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa:5' |

| binlog.000001 | 1063 | Query | 1 | 1122 | BEGIN |

| binlog.000001 | 1122 | View\_change | 1 | 1261 | view\_id=14724832985483517:2 |

| binlog.000001 | 1261 | Query | 1 | 1326 | COMMIT |

+---------------+------+----------------+-----------+-------------+--------------------------------------------------------------------+

19 rows in set (0,00 sec)

## 添加其他实例

向组中添加其他实例基本上与添加第二个服务器相同的步骤顺序，只是必须更改配置。

## 验证单主模式

1.查看当前主节点的uuid，配合replication\_group\_members中的信息确定主节点的IP：

 root@localhost : (none) 17:43:13>select \* from performance\_schema.global\_status where variable\_name = 'group\_replication\_primary\_member';

+----------------------------------+--------------------------------------+

| VARIABLE\_NAME | VARIABLE\_VALUE |

+----------------------------------+--------------------------------------+

| group\_replication\_primary\_member | 0098bd1f-8c9e-11e7-a58a-ecb1d7897858 |

+----------------------------------+--------------------------------------+

1 row in set (0.00 sec)

root@localhost : (none) 17:43:15>SELECT \* FROM performance\_schema.replication\_group\_members;

+---------------------------+--------------------------------------+--------------+-------------+--------------+

| CHANNEL\_NAME | MEMBER\_ID | MEMBER\_HOST | MEMBER\_PORT | MEMBER\_STATE |

+---------------------------+--------------------------------------+--------------+-------------+--------------+

| group\_replication\_applier | 0098bd1f-8c9e-11e7-a58a-ecb1d7897858 | 192.168.1.20 | 3306 | ONLINE |

| group\_replication\_applier | 0219385c-8ca3-11e7-bae0-288023af52c8 | 192.168.4.21 | 3306 | ONLINE |

| group\_replication\_applier | 30cbdc50-8ca1-11e7-a221-288023af50f8 | 192.168.4.20 | 3306 | ONLINE |

+---------------------------+--------------------------------------+--------------+-------------+--------------+

3 rows in set (0.00 sec)

2.在各个节点上执行DDL语句，并观察其返回结果，没有执行过去的可以查看一下参数来观察其原因：

**Execute on 192.168.4.20/21**

root@localhost : (none) 17:49:42>create database yooli;

ERROR 1290 (HY000): The MySQL server is running with the --super-read-only option so it cannot execute this statement

root@localhost : (none) 17:50:03>show variables like '%super\_read\_only%';

+-----------------+-------+

| Variable\_name | Value |

+-----------------+-------+

| super\_read\_only | ON |

+-----------------+-------+

1 row in set (0.01 sec)

**Execute on 192.168.1.20**

root@localhost : (none) 17:52:58>create database yooli;

Query OK, 1 row affected (0.01 sec)

root@localhost : (none) 17:53:05>show variables like '%super\_read\_only%';

+-----------------+-------+

| Variable\_name | Value |

+-----------------+-------+

| super\_read\_only | OFF |

+-----------------+-------+

1 row in set (0.00 sec)

root@localhost : (none) 17:53:12>show databases;

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

| sys |

| test |

| yooli |

+--------------------+

6 rows in set (0.00 sec)

root@localhost : (none) 17:54:54>