**Mysql5.7 MGR集群搭建**

**1 介绍**

**MySQL Group Replication（简称MGR）是MySQL官方于2016年12月推出的一个全新的高可用与高扩展的解决方案。MySQL组复制提供了高可用、高扩展、高可靠的MySQL集群服务。**

* 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。

**MGR是MySQL数据库未来发展的一个重要方向。**

**MGR基础结构要求：**

**引擎必须为innodb**,因为需事务支持在commit时对各节点进行冲突检查

**每个表必须有主键**，在进行事务冲突检测时需要利用主键值对比

**必须开启binlog且为row格式**

**开启GTID**，且主从状态信息存于表中（--master-info-repository=TABLE 、--relay-log-info-repository=TABLE），--log-slave-updates打开

**一致性检测设置**--transaction-write-set-extraction=XXHASH64

**MGR使用限制**：

RP和普通复制binlog校验不能共存，需设置--binlog-checksum=none

不支持gap lock(间隙锁)，隔离级别需设置为read\_committed

不支持对表进行锁操作（lock /unlock table）,不会发送到其他节点执行 ,影响需要对表进行加锁操作的情况，列入mysqldump全表备份恢复操作

不支持serializable(序列化)隔离级别

DDL语句不支持原子性，不能检测冲突，执行后需自行校验是否一致

不支持外键：多主不支持，单主模式不存在此问题

最多支持9个节点：超过9台server无法加入组

**2 环境准备**

**2.1 数据库服务器规划**

| **序号** | **IP地址** | **主机名** | **数据库** | **端口号** | **Server ID** | **操作系统** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 10.10.10.68 | Xuegod68 | mysql-5.7.20 | 3306 | 100 | CentOS7.4 |
| 2 | 10.10.10.69 | Xuegod69 | mysql-5.7.20 | 3306 | 101 | CentOS7.4 |
| 3 | 10.10.10.70 | Xuegod70 | mysql-5.7.20 | 3306 | 102 | CentOS7.4 |

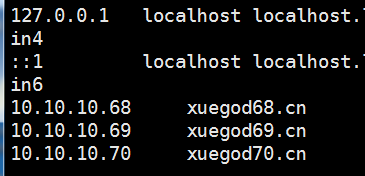
**2.2 安装mysql5.7.20**

安装mysql请参照之前课程

…………………………………………

在这里就不详细介绍。

**2.3 设置hostname和ip映射**

在三台数据库服务器上都设置：   


**3. 创建复制环境**

**3.1 服务器xuegod68.cn**

**3.1.1 配置/etc/my.cnf**

vim /etc/my.cnf

[mysqld]

# Group Replication

server\_id = 100 #服务ID

gtid\_mode = ON #全局事务

enforce\_gtid\_consistency = ON #强制GTID的一致性

master\_info\_repository = TABLE #将master.info元数据保存在系统表中

relay\_log\_info\_repository = TABLE #将relay.info元数据保存在系统表中

binlog\_checksum = NONE #禁用二进制日志事件校验

log\_slave\_updates = ON #级联复制

log\_bin = binlog #开启二进制日志记录

binlog\_format= ROW #以行的格式记录

transaction\_write\_set\_extraction = XXHASH64 #使用哈希算法将其编码为散列

loose-group\_replication\_group\_name = 'ce9be252-2b71-11e6-b8f4-00212844f856' #加入的组名

loose-group\_replication\_start\_on\_boot = off #不自动启用组复制集群

loose-group\_replication\_local\_address = 'xuegod68.cn:33061' #以本机端口33061接受来自组中成员的传入连接

loose-group\_replication\_group\_seeds =' xuegod68.cn:33061, xuegod69.cn:33062, xuegod70.cn:33063' #组中成员访问表

loose-group\_replication\_bootstrap\_group = off #不启用引导组

重启mysql服务

**3.1.2 服务器xuegod68.cn上建立复制账号：**

mysql> set SQL\_LOG\_BIN=0; #停掉日志记录

mysql> grant replication slave on \*.\* to repl@'10.10.10.%' identified by '123456';

mysql> flush privileges;

mysql> set SQL\_LOG\_BIN=1; #开启日志记录

mysql> change master to master\_user='repl',master\_password='123456' for channel 'group\_replication\_recovery'; #构建group replication集群

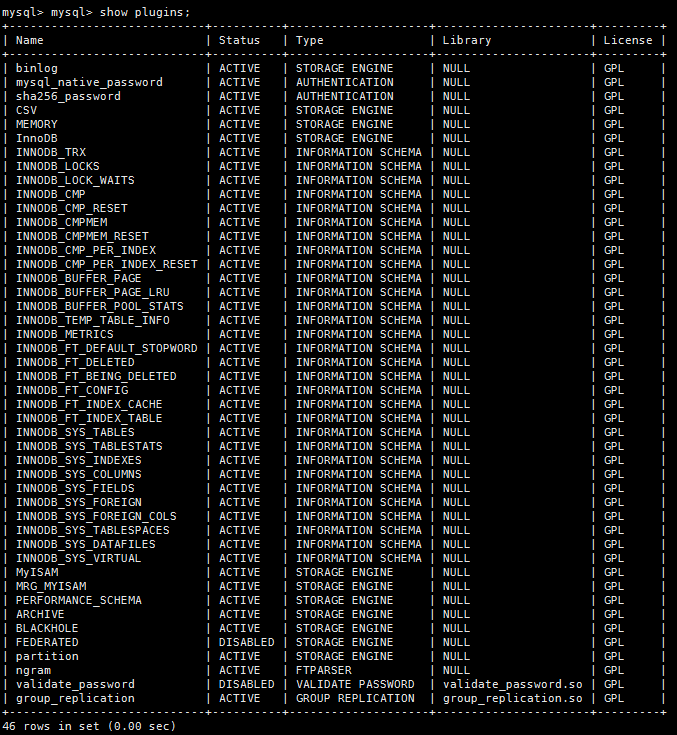
**3.1.3 在mysql服务器xuegod68.cn上安装group replication插件**

-- 安装插件

mysql> install PLUGIN group\_replication SONAME 'group\_replication.so';

-- 查看group replication组件

mysql> show plugins;



**3.1.4 启动服务器xuegod68.cn上mysql的group replication**

-- 设置group\_replication\_bootstrap\_group为ON是为了标示以后加入集群的服务器以这台服务器为基准，以后加入的就不需要设置。

mysql> set global group\_replication\_bootstrap\_group=ON;

-- 作为首个节点启动mgr集群

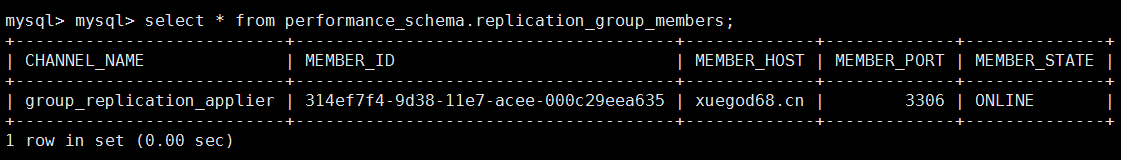
mysql> start group\_replication;

mysql> set global group\_replication\_bootstrap\_group=OFF;

**3.1.5 查看mgr的状态**

-- 查询表performance\_schema.replication\_group\_members

mysql> select \* from performance\_schema.replication\_group\_members;

**3.1.6 测试服务器xuegod68.cn上的mysql**

mysql> create database test;

Query OK, 1 row affected (0.01 sec)

mysql> use test;

Database changed

mysql> create table t1 (id int primary key,name varchar(20)); #注意创建主键

Query OK, 0 rows affected (0.01 sec)

mysql> insert into t1 values (1,'man');

Query OK, 1 row affected (0.01 sec)

mysql> select \* from t1;

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

| id | name |

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

| 1 | man |

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

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 | 181 | 123 | Server ver: 5.7.17-log, Binlog ver: 4 |

| binlog.000001 | 123 | Previous\_gtids | 181 | 150 | |

| binlog.000001 | 150 | Gtid | 181 | 211 | SET @@SESSION.GTID\_NEXT= 'ce9be252-2b71-11e6-b8f4-00212844f856:1' |

| binlog.000001 | 211 | Query | 181 | 270 | BEGIN |

| binlog.000001 | 270 | View\_change | 181 | 369 | view\_id=14916679781649312:1 |

| binlog.000001 | 369 | Query | 181 | 434 | COMMIT |

| binlog.000001 | 434 | Gtid | 181 | 495 | SET @@SESSION.GTID\_NEXT= 'ce9be252-2b71-11e6-b8f4-00212844f856:2' |

| binlog.000001 | 495 | Query | 181 | 585 | create database test |

| binlog.000001 | 585 | Gtid | 181 | 646 | SET @@SESSION.GTID\_NEXT= 'ce9be252-2b71-11e6-b8f4-00212844f856:3' |

| binlog.000001 | 646 | Query | 181 | 769 | use `test`; create table t1(c1 int primary key, c2 text not null) |

| binlog.000001 | 769 | Gtid | 181 | 830 | SET @@SESSION.GTID\_NEXT= 'ce9be252-2b71-11e6-b8f4-00212844f856:4' |

| binlog.000001 | 830 | Query | 181 | 898 | BEGIN |

| binlog.000001 | 898 | Table\_map | 181 | 941 | table\_id: 219 (test.t1) |

| binlog.000001 | 941 | Write\_rows | 181 | 983 | table\_id: 219 flags: STMT\_END\_F |

| binlog.000001 | 983 | Xid | 181 | 1010 | COMMIT /\* xid=40 \*/ |

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

15 rows in set (0.00 sec)

**3.2 复制组添加新实例xuegod69.cn**

**3.2.1 修改/etc/my.cnf 配置文件，方法和之前相同**

[mysqld]

# Group Replication

server\_id = 101 #注意服务ID不一样

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 = binlog

binlog\_format= ROW

transaction\_write\_set\_extraction = XXHASH64

loose-group\_replication\_group\_name = 'ce9be252-2b71-11e6-b8f4-00212844f856'

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

loose-group\_replication\_local\_address = 'xuegod69.cn:33062'

loose-group\_replication\_group\_seeds = 'xuegod68.cn:33061, xuegod69.cn:33062, xuegod70.cn:33063'

loose-group\_replication\_bootstrap\_group = off

重启mysql服务

**3.2.2 用户授权**

mysql> set SQL\_LOG\_BIN=0; #停掉日志记录

mysql> grant replication slave on \*.\* to repl@'10.10.10.%' identified by '123456';

mysql> flush privileges;

mysql> set SQL\_LOG\_BIN=1; #开启日志记录

mysql> change master to master\_user='repl',master\_password='123456' for channel 'group\_replication\_recovery'; #构建group replication集群

-- 安装group replication插件

mysql> install PLUGIN group\_replication SONAME 'group\_replication.so';

Query OK, 0 rows affected (0.00 sec)

**3.2.3 把实例添回到之前的复制组**

mysql> set global group\_replication\_allow\_local\_disjoint\_gtids\_join=ON;

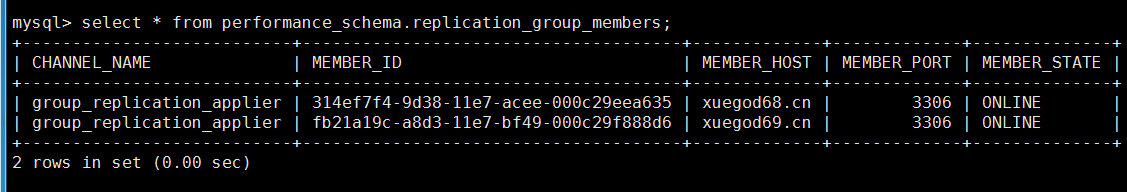
Query OK, 0 rows affected (0.00 sec)

mysql> start group\_replication;

Query OK, 0 rows affected (6.65 sec)

**3.2.4 在xuegod68.cn上查看复制组状态**

mysql> select \* from performance\_schema.replication\_group\_members;

3.2.5 在新回的实例上查看数据库发现test库和t1表已经同步

mysql> show databases;

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

| Database |

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

| information\_schema |

| mysql |

| performance\_schema |

| sys |

| test |

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

5 rows in set (0.00 sec)

mysql> use test;

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> show tables;

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

| Tables\_in\_test |

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

| t1 |

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

1 row in set (0.00 sec)

mysql> select \* from t1;

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

| id | name |

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

| 1 | man |

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

1 row in set (0.00 sec)

**3.3 以同样方法添加xuegod70.cn**

详细步骤请参考3.2, 这里只给出配置文件/etc/my.cnf：

[mysqld]

# Group Replication

server\_id = 102 #注意服务id不一样

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 = binlog

binlog\_format= ROW

transaction\_write\_set\_extraction = XXHASH64

loose-group\_replication\_group\_name = 'ce9be252-2b71-11e6-b8f4-00212844f856'

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

loose-group\_replication\_local\_address = 'xuegod70.cn:33063'

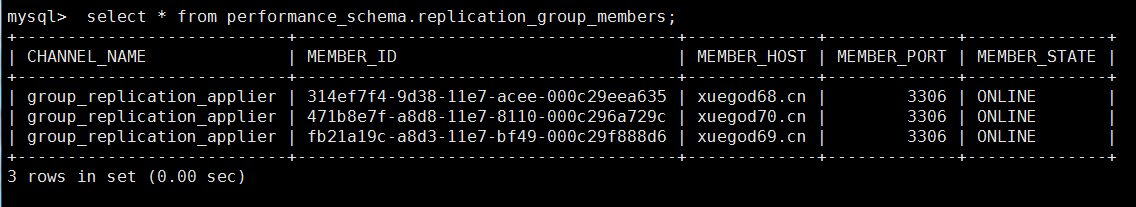
loose-group\_replication\_group\_seeds ='xuegod68.cn:33061, xuegod69.cn:33062, xuegod70.cn:33063'

loose-group\_replication\_bootstrap\_group = off

重启mysql服务 然后参照第二个节点步骤操作加入组复制。

**3.4 查看复制组状态**

mysql> select \* from performance\_schema.replication\_group\_members;



**以上单master节点的集群就搭建完毕!**

**查看集群参数设置列表show variables like 'group\_replication%';**

**multi-primary模式(多主master模式)**：

1、该模式启用需设置两个参数

group\_replication\_single\_primary\_mode=0 #这个参数很好理解，就是关闭单master模式

group\_replication\_enforce\_update\_everywhere\_checks=1 #这个参数设置多主模式下各个节点严格一致性检查

2、 默认启动的都是单master模式，其他节点都设置了read\_only、super\_read\_only这两个参数，需要修改这两个配置

3、 完成上面的配置后就可以执行多点写入了，多点写入会存在冲突检查，这耗损性能挺大的，官方建议采用网络分区功能，在程序端把相同的业务定位到同一节点，尽量减少冲突发生几率。

**由单主模式修改为多主模式方法**

在原来单主模式的主节点执行操作如下：

stop GROUP\_REPLICATION;

set global group\_replication\_single\_primary\_mode=off;

set global group\_replication\_enforce\_update\_everywhere\_checks=ON;

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

START GROUP\_REPLICATION;

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

而对于其他的节点，执行下面的操作即可。

stop GROUP\_REPLICATION;

set global group\_replication\_allow\_local\_disjoint\_gtids\_join=ON;(即使含有组中不存在的事务，也允许当前server加入组)

set global group\_replication\_single\_primary\_mode=off;

set global group\_replication\_enforce\_update\_everywhere\_checks=ON;

start group\_replication;

**直接搭建多主master模式**

my.cnf配置文件：

# Group Replication

server\_id = 100 #服务ID

gtid\_mode = ON #全局事务

enforce\_gtid\_consistency = ON #强制GTID的一致性

master\_info\_repository = TABLE #将master.info元数据保存在系统表中

relay\_log\_info\_repository = TABLE #将relay.info元数据保存在系统表中

binlog\_checksum = NONE #禁用二进制日志事件校验

log\_slave\_updates = ON #级联复制

log\_bin = binlog #开启二进制日志记录

binlog\_format= ROW #以行的格式记录

transaction\_write\_set\_extraction = XXHASH64 #使用哈希算法将其编码为散列

loose-group\_replication\_group\_name = 'ce9be252-2b71-11e6-b8f4-00212844f856' #加入的组名

loose-group\_replication\_start\_on\_boot = off #不自动启用组复制集群

loose-group\_replication\_local\_address = 'xuegod68.cn:33061' #以本机端口33061接受来自组中成员的传入连接

loose-group\_replication\_group\_seeds = 'xuegod68.cn:33061,xuegod69.cn:33062,xuegod70.cn:33063' #组中成员访问表

loose-group\_replication\_bootstrap\_group = off #不启用引导组

group\_replication\_single\_primary\_mode = off #关闭单master模式

group\_replication\_enforce\_update\_everywhere\_checks = ON #多主一致性检查

如果日志不一致(先停止集群 reset master清空日志)