

# NSD DBA2 DAY01

1. [案例1：MySQL一主一从](#)
2. [案例2：配置主从从同步结构](#)
3. [配置半同步复制模式](#)

## 1 案例1：MySQL一主一从

### 1.1 问题

- 构建 主-->从 复制结构
- 其中主机192.168.4.10作为主库
- 主机192.168.4.20作为从库

1.

### 1.2 方案

使用2台RHEL 7虚拟机，如图-1所示。其中192.168.4.10是MySQL主服务器，负责提供同步源；另一台192.168.4.20作为MySQL从服务器，通过调取主服务器上的binlog日志，在本地重做对应的库、表，实现与主服务器的AB复制（同步）。

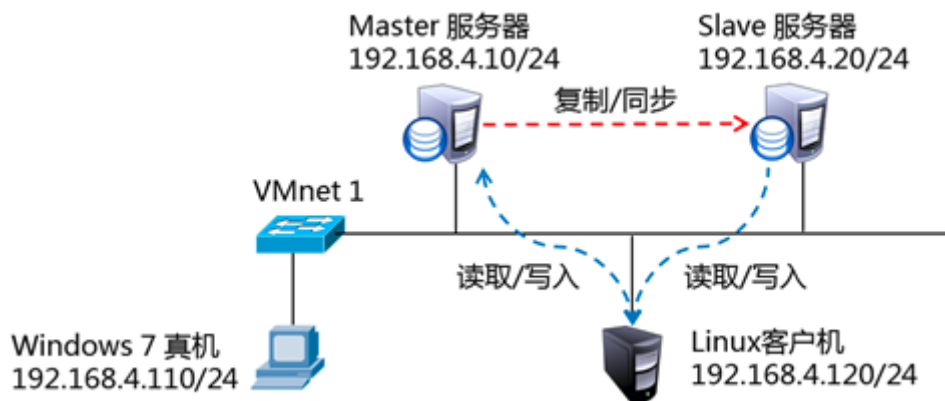


图 - 1

提前为两台MySQL服务器安装好MySQL-server、MySQL-Client软件包，并为数据库用户root修改密码；Linux客户机上则只需安装MySQL-Client软件包即可。

### 1.3 步骤

实现此案例需要按照如下步骤进行。

#### 步骤一：初始化现有库

为了在启用binlog日志及同步之前保持主、从库的一致性，建议进行初始化——备份主服务器上现有的库，然后导入到从服务器上。

当现有库、表都采用MyISAM引擎时，可执行离线备份、恢复，这样更有效率；否则，可通过mysqldump等工具来实现库的导出、导入。

1) 备份MySQL Master ( 192.168.4.10 ) 上现有的库

[Top](#)

如果服务器已经启用binlog，建议对日志做一次重置，否则可忽略：

```

01. [root@dbsvr1 ~] # mysql -u root -p
02. Enter password: //以数据库用户root登入
03. ...
04. mysql> RESET MASTER; //重置binlog日志
05. Query OK, 0 rows affected ( 0.06 sec)
06. mysql> quit //退出mysql> 环境
07. Bye

```

以备份mysql库、sys库为例，导出操作如下：

```

01. [root@dbsvr1 ~] # mysqldump -u root -p -- all- databases > /root/mytest.sql
02. Enter password: //验证口令
03. [root@dbsvr1 ~] # ls -lh /root/mytest.sql //确认备份结果
04. -rw-r--r-- . 1 root root 777172 4月 23 12:21 /root/mytest.sql

```

## 2) 在MySQL Slave ( 192.168.4.20 ) 上导入备份的库

先清理目标库，避免导入时冲突。主要是采用InnoDB引擎的库，授权库mysql多采用MyISAM引擎，可不作清理。

```

01. [root@dbsvr2 ~] # mysql -u root -p
02. Enter password: //以数据库用户root登入
03. ...
04. mysql> DROP DATABASE test; //删除test库
05. Query OK, 0 rows affected ( 0.03 sec)
06. mysql> quit //退出mysql> 环境
07. Bye

```

使用scp工具下载备份文件：

```

01. [root@dbsvr2 ~] # scp /root/mytest.sql root@192.168.4.20: /
02. root@dbsvr1's password: //验证对方系统用户root的口令
03. mytest.sql 100% 759KB 759.0KB/s 00:00
04. [root@dbsvr2 ~] # ls -lh mytest.sql //确认下载结果
05. -rw-r--r-- . 1 root root 759K 4月 23 12:22 /mytest.sql

```

[Top](#)

执行导入操作：

```
01. [ root@dbsvr2 ~] # mysql - u root - p < /my test.sql
02. Enter password: //验证口令
```

导入成功后，可重新登入 mysql> 环境，确认清理的目标库已恢复：

```
01. mysql> show databases;
02. +-----+
03. | Database |
04. +-----+
05. | information_schema |
06. | mysql |
07. | performance_schema |
08. | sys |
09. +-----+
10. 4 rows in set (0.00 sec)
```

## 步骤二：配置MySQL Master（主服务器，192.168.4.10）

1）修改/etc/my.cnf配置，重新启动MySQL服务程序

指定服务器ID号、允许日志同步：

```
01. [ root@dbsvr1 mysql] # vim /etc/my.cnf
02. [ mysqld]
03. log_bin=dbsvr1 bin //启用binlog日志，并指定文件名前缀
04. server_id = 10 //指定服务器ID号
05. ....
```

重启mysql服务：

```
01. [ root@dbsvr1 ~] # systemctl restart mysqld.service
```

2）新建一个备份用户，授予复制权限

需要的权限为REPLICATION SLAVE，允许其从Slave服务器访问：

```
01. mysql> GRANT REPLICATION SLAVE ON *.* TO 'replicater'@'192.168.4.%' IDENTIFIED BY '123456';
02. Query OK, 0 rows affected, 1 warning (0.09 sec)
```

### 3) 检查Master服务器的同步状态

在已经初始化现有库的情况下，查看MASTER状态，记录下当前的日志文件名、偏移的位置（下面SLAVE发起复制时需要用到）：

```
01.  my sql> SHOW MASTER STATUS\G
02.  ***** 1 row *****
03.      File: dbsvr1- bin.000001          //记住当前的日志文件名
04.      Position: 154                    //记住当前的位置
05.      Binlog_Do_DB:
06.      Binlog_Ignore_DB:
07.      Executed_Gtid_Set:
08.  1 row in set ( 0.00 sec)
```

### 步骤三：配置MySQL Slave（从服务器，192.168.4.20）

#### 1) 修改/etc/my.cnf配置，重新启动MySQL服务程序

指定服务器ID号、允许日志同步：

```
01.  [ root@dbsvr2 ~] # vim /etc/my.cnf
02.  [ my sql]
03.  log_bin=dbsvr2- bin          //启动SQL日志，并指定文件名前缀
04.  server_id = 20              //指定服务器ID号，不要与Master的相同
05.  ... ..
```

在生产环境中，还可以根据需要设置更详细的同步选项。比如，指定当主、从网络中断时的重试超时时间（slave-net-timeout=60）等，具体可参考MySQL手册。

配置完成后，重启mysql服务：

```
01.  [ root@dbsvr2 ~] # systemctl restart mysqld.service
```

通过CHANGE MASTER语句指定MASTER服务器的IP地址、同步用户名/密码、起始日志文件、偏移位置（参考MASTER上的状态输出）：

```
01.  my sql> CHANGE MASTER TO MASTER_HOST='192.168.4.10',
02.      -> MASTER_USER='replicater',
03.      -> MASTER_PASSWORD='pwd123',
04.      -> MASTER_LOG_FILE='dbsvr1- bin.000002', //对应Master的日志文件
```

[Top](#)

05. - > MASTER\_LOG\_POS=334; //对应Master的日志偏移位置
06. Query OK, 0 rows affected, 2 warnings ( 0.12 sec)

然后执行START SLAVE ( 较早版本中为SLAVE START ) 启动复制：

01. my sql> START SLAVE; //启动复制
02. Query OK, 0 rows affected ( 0.00 sec)

**注意：一旦启用SLAVE复制，当需要修改MASTER信息时，应先执行STOP SLAVE停止复制，然后重新修改、启动复制。**

通过上述连接操作，MASTER服务器的设置信息自动存为master.info文件，以后每次MySQL服务程序时会自动调用并更新，无需重复设置。查看master.info文件的开头部分内容，可验证相关设置：

01. [ root@dbsvr2 ~] # ls -lh /var/lib/mysql/master.info
02. -rw-r----- . 1 mysql mysql 132 4月 23 12:06 /var/lib/mysql/master.info
03. [ root@dbsvr2 ~] # head /var/lib/mysql/master.info
04. 25
05. dbsvr1-bin.000001
06. 154
07. 192.168.4.10
08. replicater
09. pwd123
10. 3306
11. 60
12. 0

## 2 ) 检查Slave服务器的同步状态

通过SHOW SLAVE STATUS语句可查看从服务器状态，确认其中的IO线程、SQL线程正常运行，才能成功同步：

01. my sql> SHOW SLAVE STATUS\G
02. Slave\_IO\_State: Waiting for master to send event
03. Master\_Host: 192.168.4.1
04. Master\_User: replicater
05. Master\_Port: 3306
06. Connect\_Retry: 60
07. Master\_Log\_File: dbsvr1-bin.000001

[Top](#)

```

08.      Read_Master_Log_Pos: 154
09.      Relay_Log_File: db2-relay-bin.000003
10.      Relay_Log_Pos: 321
11.      Relay_Master_Log_File: db2-relay-bin.000001
12.      Slave_IO_Running: Yes      //IO线程应该已运行
13.      Slave_SQL_Running: Yes    //SQL线程应该已运行
14.      Replicate_Do_DB:
15.      Replicate_Ignore_DB:
16.      Replicate_Do_Table:
17.      Replicate_Ignore_Table:
18.      Replicate_Wild_Do_Table:
19.      Replicate_Wild_Ignore_Table:
20.      Last_Errno: 0
21.      Last_Error:
22.      Skip_Counter: 0
23.      Exec_Master_Log_Pos: 154
24.      Relay_Log_Space: 2490
25.      Until_Condition: None
26.      Until_Log_File:
27.      Until_Log_Pos: 0
28.      Master_SSL_Allowed: No
29.      Master_SSL_CA_File:
30.      Master_SSL_CA_Path:
31.      Master_SSL_Cert:
32.      Master_SSL_Cipher:
33.      Master_SSL_Key:
34.      Seconds_Behind_Master: 0
35.      Master_SSL_Verify_Server_Cert: No
36.      Last_IO_Errno: 0
37.      Last_IO_Error:
38.      Last_SQL_Errno: 0
39.      Last_SQL_Error:
40.      Replicate_Ignore_Server_Ids:
41.      Master_Server_Id: 10
42.      Master_UUID: 2d4d8a11-27b7-11e7-ae78-52540055c180
43.      Master_Info_File: /var/lib/mysql/master.info
44.      SQL_Delay: 0
45.      SQL_Remaining_Delay: NULL
46.      Slave_SQL_Running_State: Slave has read all relay log; waiting for more updates
47.      Master_Retry_Count: 86400
48.      Master_Bind:

```

[Top](#)

```

49.      Last_IO_Error_Timestamp:
50.      Last_SQL_Error_Timestamp:
51.      Master_SSL_Crl:
52.      Master_SSL_Crlpath:
53.      Retrieved_Gtid_Set:
54.      Executed_Gtid_Set:
55.      Auto_Position: 0
56.      Replicate_Rewrite_DB:
57.      Channel_Name:
58.      Master_TLS_Version:
59.  1 row in set ( 0.00 sec)

```

若START SLAVE直接报错失败，请检查CHANGE MASTER相关设置是否有误，纠正后再重试；若IO线程或SQL线程有一个为“No”，则应检查服务器的错误日志，分析并排除故障后重启主从复制。

#### 步骤四：测试主从同步效果

1) 在Master上操作数据库、表、表记录

新建newdb库、newtable表，随意插入几条表记录：

```

01.  my sql> CREATE DATABASE newdb;                //新建库newdb
02.  Query OK, 1 row affected ( 0.17 sec)
03.
04.  my sql> USE newdb;                            //切换到newdb库
05.  Database changed
06.
07.  my sql> CREATE TABLE newtable( id int( 4) );    //新建newtable表
08.  Query OK, 0 rows affected ( 0.46 sec)
09.
10.  my sql> INSERT INTO newtable VALUES( 1234) ,( 5678); //插入2条表记录
11.  Query OK, 2 rows affected ( 0.24 sec)
12.  Records: 2 Duplicates: 0 Warnings: 0
13.  my sql> SELECT * FROM newtable;                //确认表数据
14.  +-----+
15.  | id |
16.  +-----+
17.  | 1234 |
18.  | 5678 |
19.  +-----+
20.  2 rows in set ( 0.00 sec)

```

[Top](#)

## 2) 在Slave上确认自动同步的结果

直接切换到newdb库，并查询newtable表的记录，应该与Master上的一样，这才说明主从同步已经成功生效：

```

01.  my sql> USE newdb;                                //直接切换到newdb库
02.  Reading table information for completion of table and column names
03.  You can turn off this feature to get a quicker startup with - A
04.
05.  Database changed
06.
07.  my sql> SELECT * FROM newtable;                    //输出表记录
08.  +-----+
09.  | id |
10.  +-----+
11.  | 1234 |
12.  | 5678 |
13.  +-----+
14.  2 rows in set (0.02 sec)

```

## 3) 在Master服务器上可查看Slave主机的信息

```

01.  my sql> SHOW SLAVE HOSTS;
02.  +-----+-----+-----+-----+-----+
03.  | Server_id | Host | Port | Master_id | Slave_UUID |
04.  +-----+-----+-----+-----+-----+
05.  | 2 | 3306 | 10 | 512cf7c1-27c4-11e7-8f4b-5254007b030b |
06.  +-----+-----+-----+-----+-----+
07.  1 row in set (0.00 sec)

```

## 2 案例2：配置主从从同步结构

### 2.1 问题

- 具体要求如下：
- 配置主机192.168.4.51为主数据库服务器
- 配置主机192.168.4.52为51主机的从库服务器
- 配置主机192.168.4.53为52主机的从库服务器
- 客户端连接主数据库服务器51主机创建的数据，连接52和53主机时，也可以访问到库、表、记录。

1.



2.

## 2.2 方案

使用3台RHEL 7虚拟机，如图-2所示。其中192.168.4.51是MySQL主服务器，负责提供同步源；另一台192.168.4.52作为192.168.4.51从服务器，最后一台192.168.4.53作为192.168.4.52从服务器，通过调取主服务器上的binlog日志，客户端访问主库51时创建库表记录在52和53数据库服务器都可以看到



图 - 2

## 2.3 步骤

实现此案例需要按照如下步骤进行。

### 步骤一：环境准备

为了在启用binlog日志及同步之前保持主、从库的一致性，主从同步未配置之前，要保证从库上要有主库上的数据，禁用selinux，关闭防火墙服务，保证物理连接正常

```

01. [ root@db51 ~ ] # systemctl stop firewalld
02. [ root@db51 ~ ] # setenforce 0
03. [ root@db51 ~ ] # ping -c 2 192.168.4.51
04. PING 192.168.4.51 ( 192.168.4.51 ) 56( 84 ) bytes of data.
05. 64 bytes from 192.168.4.51: icmp_seq=1 ttl=64 time=0.059 ms
06. 64 bytes from 192.168.4.51: icmp_seq=2 ttl=64 time=0.063 ms
07.
08. --- 192.168.4.51 ping statistics ---
09. 2 packets transmitted, 2 received, 0% packet loss, time 999ms
10. rtt min/avg/max/mdev = 0.059/0.061/0.063/0.002 ms
11. [ root@db51 ~ ] # ping -c 2 192.168.4.52
12. PING 192.168.4.52 ( 192.168.4.52 ) 56( 84 ) bytes of data.
13. 64 bytes from 192.168.4.52: icmp_seq=1 ttl=64 time=0.698 ms
14. 64 bytes from 192.168.4.52: icmp_seq=2 ttl=64 time=0.365 ms
15.
16. --- 192.168.4.52 ping statistics ---
17. 2 packets transmitted, 2 received, 0% packet loss, time 1000ms
18. rtt min/avg/max/mdev = 0.365/0.531/0.698/0.168 ms
  
```

[Top](#)

## 步骤二：配置主服务器192.168.4.51

### 1) 对yaya用户进行授权

```
01. [root@db51 ~]# mysql -uroot -p123456
02. mysql> grant replication slave on *.* to yaya@"%" identified by "123456";
03. Query OK, 0 rows affected, 1 warning (0.03 sec)
```

### 2) 启用binlog日志，修改/etc/my.cnf配置，重新启动MySQL服务程序

指定服务器ID号、允许日志同步：

```
01. [root@db51 ~]# vim /etc/my.cnf
02. [mysqld]
03. log_bin=db51           //启用binlog日志，并指定文件名前缀
04. server_id=51           //指定服务器ID号
05. binlog_format="mixed"  //指定binlog日志格式
```

重启mysql服务：

```
01. [root@db51 ~]# systemctl restart mysqld
```

确保/var/lib/mysql下面有两个文件：

```
01. [root@db51 ~]# ls /var/lib/mysql/db51 *
02. /var/lib/mysql/db51.000001 /var/lib/mysql/db51.index
```

查看主服务正在使用的日志信息

查看主服务器状态，记录下当前的日志文件名、偏移的位置（下面SLAVE发起复制时需要用到）：

```
01. mysql> show master status;
02. +-----+-----+-----+-----+-----+
03. | File      | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
04. +-----+-----+-----+-----+-----+ Top
05. | db51.000002 | 437 | | | |
06. +-----+-----+-----+-----+-----+
```

07. 1 row in set ( 0.00 sec)

### 步骤三：配置从服务器192.168.4.52

1) 在服务器192.168.4.52上对user53用户进行授权

```
01. [ root@db52 ~] # mysql -u root -p123456
02. mysql> grant replication slave on *.* to user53@"192.168.4.53" identified by "6543"
03. Query OK, 0 rows affected, 1 warning ( 0.00 sec)
```

2) 修改/etc/my.cnf配置，启用binlog日志，指定server\_id 和 允许级联复制

```
01. [ root@db52 ~] # vim /etc/my.cnf
02. [ mysql]
03. server_id=52
04. log_bin=db52
05. binlog_format="mixed"
06. log_slave_updates           //允许级联复制
```

3) 配置完成后，重启mysql服务：

```
01. [ root@db52 ~] # systemctl restart mysqld
```

4) 确保/var/lib/mysql下面有两个文件：

```
01. [ root@db52 ~] # ls /var/lib/mysql/db52.*
02. /var/lib/mysql/db52.000001 /var/lib/mysql/db52.index
```

5) 查看正在使用的日志信息

```
01. [ root@db52 ~] # mysql -u root -p123456
02. mysql> show master status;
03. +-----+-----+-----+-----+-----+
04. | File      | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
```

[Top](#)

```

05.  +-----+-----+-----+-----+-----+
06.  | db52.000001 |   154 |           |           |           |
07.  +-----+-----+-----+-----+-----+
08.  1 row in set ( 0.00 sec)  //查看日志文件名、偏移的位置

```

6 ) 验证主库的授权用户

```
[root@db52 ~]# mysql -h192.168.4.51 -uyaya -p123456
```

**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 4**

**Server version: 5.7.17-log MySQL Community Server (GPL)**

**Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.**

**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.**

```
01.  my sql>                                     //验证成功
```

7 ) 通过change master语句指定master服务器的IP地址、同步用户名/密码、起始日志文件、偏移位置 ( 参考master上的状态输出 ) :

```
[root@db52 ~]# mysql -uroot -p123456
```

```
mysql> change master to
```

```
-> master_host="192.168.4.51",
```

```
-> master_user="yaya",
```

```
-> master_password="123456",
```

```
-> master_log_file="db51.000002",
```

```
-> master_log_pos=437;
```

**Query OK, 0 rows affected, 2 warnings (0.43 sec)**

8 ) 启动slave进程

```
mysql> start slave;
```

**Query OK, 0 rows affected (0.03 sec)**

9 ) 查看进程状态信息，通过show slave status语句可查看从服务器状态，确认其中的IO线程、SQL线程正常运行，才能成功同步，IO线程和SQL线程必须是Yes

```
mysql> show slave status \G;
```

```
***** 1. row *****
```

**Slave\_IO\_State: Waiting for master to send event**

[Top](#)

**Master\_Host: 192.168.4.51**  
**Master\_User: yaya**  
**Master\_Port: 3306**  
**Connect\_Retry: 60**  
**Master\_Log\_File: db51.000002**  
**Read\_Master\_Log\_Pos: 437**  
**Relay\_Log\_File: db52-relay-bin.000002**  
**Relay\_Log\_Pos: 315**  
**Relay\_Master\_Log\_File: db51.000002**  
**Slave\_IO\_Running: Yes**  
**Slave\_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\_Master\_Log\_Pos: 437**  
**Relay\_Log\_Space: 521**  
**Until\_Condition: None**  
**Until\_Log\_File:**  
**Until\_Log\_Pos: 0**  
**Master\_SSL\_Allowed: No**  
**Master\_SSL\_CA\_File:**  
**Master\_SSL\_CA\_Path:**  
**Master\_SSL\_Cert:**  
**Master\_SSL\_Cipher:**  
**Master\_SSL\_Key:**  
**Seconds\_Behind\_Master: 0**  
**Master\_SSL\_Verify\_Server\_Cert: No**  
**Last\_IO\_Errno: 0**  
**Last\_IO\_Error:**  
**Last\_SQL\_Errno: 0**  
**Last\_SQL\_Error:**  
**Replicate\_Ignore\_Server\_Ids:**  
**Master\_Server\_Id: 51**  
**Master\_UUID: 81a13101-aa66-11e8-ad11-525400019e62**

[Top](#)

**Master\_Info\_File: /var/lib/mysql/master.info**

**SQL\_Delay: 0**

**SQL\_Remaining\_Delay: NULL**

**Slave\_SQL\_Running\_State: Slave has read all relay log; waiting for more updates**

**Master\_Retry\_Count: 86400**

**Master\_Bind:**

**Last\_IO\_Error\_Timestamp:**

**Last\_SQL\_Error\_Timestamp:**

**Master\_SSL\_Crl:**

**Master\_SSL\_Crlpath:**

**Retrieved\_Gtid\_Set:**

**Executed\_Gtid\_Set:**

**Auto\_Position: 0**

**Replicate\_Rewrite\_DB:**

**Channel\_Name:**

**Master\_TLS\_Version:**

**1 row in set (0.00 sec)**

**步骤四：配置从服务器192.168.4.53**

**1) 验证主库的授权用户**

```

01. [root@db53 ~] # mysql -h192.168.4.52 -uuser53 -p654321
02. mysql: [Warning] Using a password on the command line interface can be insecure.
03. Welcome to the MySQL monitor. Commands end with ; or \g.
04. Your MySQL connection id is 7
05. Server version: 5.7.17-log MySQL Community Server (GPL)
06.
07. Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.
08.
09. Oracle is a registered trademark of Oracle Corporation and/or its
10. affiliates. Other names may be trademarks of their respective
11. owners.
12.
13. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
14.
15. mysql> //验证成功

```

**2) 指定server\_id**

[Top](#)

01. [ my sql d]
02. validate\_password\_policy=0
03. validate\_password\_length=6
04. server\_id=53

### 3 ) 重新启动服务

01. [ root@db53 ~] # systemctl restart mysqld

### 4 ) 管理员登录指定主库信息

01. [ root@db53 ~] # mysql -uroot -p123456
02. mysql> change master to
03. -> master\_host="192.168.4.52",
04. -> master\_user="user53",
05. -> master\_password="654321",
06. -> master\_log\_file="db52.000001",
07. -> master\_log\_pos=154;
08. Query OK, 0 rows affected, 2 warnings ( 0.37 sec)

### 5 ) 启动slave进程

01. mysql> start slave;
02. Query OK, 0 rows affected ( 0.04 sec)

### 6 ) 查看进程状态信息

01. mysql> show slave status\G
02. \*\*\*\*\* 1 row \*\*\*\*\*
03. Slave\_IO\_State: Waiting for master to send event
04. Master\_Host: 192.168.4.52
05. Master\_User: user53
06. Master\_Port: 3306
07. Connect\_Retry: 60
08. Master\_Log\_File: db52.000001

[Top](#)

- 09. Read\_Master\_Log\_Pos: 154
- 10. Relay\_Log\_File: db53-relay-bin.000003
- 11. Relay\_Log\_Pos: 315
- 12. Relay\_Master\_Log\_File: db52.000001
- 13. Slave\_IO\_Running: Yes
- 14. Slave\_SQL\_Running: Yes

## 步骤五：客户端验证配置

### 1) 在主服务器上在主库上授权访问gamedb库的用户

- 01. [root@db51 ~]# mysql -uroot -p123456
- 02. mysql> grant all on gamedb.\* to dada@"%" identified by "123456";
- 03. Query OK, 0 rows affected, 1 warning ( 0.03 sec)

### 2) 客户端使用授权用户连接主库，建库、表、插入记录

- 01. [root@room9pc01 ~]# mysql -h192.168.4.51 -udada -p123456
- 02. Welcome to the MariaDB monitor. Commands end with ; or \g.
- 03. Your MySQL connection id is 7
- 04. Server version: 5.7.17-log MySQL Community Server (GPL)
- 05.
- 06. Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.
- 07.
- 08. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
- 09.
- 10. MySQL [(none)]> //验证成功
- 11. MySQL [(none)]> create database gamedb; //创建测试库
- 12. Query OK, 1 row affected ( 0.04 sec)
- 13. MySQL [(none)]> create table gamedb.t1(id int); //在gamedb下创建t1表
- 14. Query OK, 0 rows affected ( 0.17 sec)
- 15. MySQL [(none)]> insert into gamedb.t1 values( 8888); //在t1表中插入数值
- 16. Query OK, 1 row affected ( 0.22 sec)

### 3) 客户端使用授权用户连接2台从库时，也可以看到主库上新的库表记录

- 01. [root@room9pc01 ~]# mysql -h192.168.4.52 -udada -p123456 //验证52主机的状态
- 02. Welcome to the MariaDB monitor. Commands end with ; or \g.

[Top](#)



```

03. Your My SQL connection id is 10
04. Server version: 5.7.17-log My SQL Community Server ( GPL)
05.
06. Copyright ( c ) 2000, 2017, Oracle, MariaDB Corporation Ab and others.
07.
08. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
09. My SQL [ ( none ) ] > select * from gamedb.t1; //查询插入的表格
10. +-----+
11. | id |
12. +-----+
13. | 8888 |
14. +-----+
15. 1 row in set ( 0.00 sec)
16. My SQL [ ( none ) ] > exit
17. [ root@room9pc01 ~ ] # mysql -h192.168.4.53 -uudada -p123456 //验证53主机的状态
18. Welcome to the MariaDB monitor. Commands end with ; or \g.
19. Your My SQL connection id is 6
20. Server version: 5.7.17 My SQL Community Server ( GPL)
21.
22. Copyright ( c ) 2000, 2017, Oracle, MariaDB Corporation Ab and others.
23.
24. Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
25.
26. My SQL [ ( none ) ] > select * from gamedb.t1;
27. +-----+
28. | id |
29. +-----+
30. | 8888 |
31. +-----+
32. 1 row in set ( 0.00 sec)

```

## 3 配置半同步复制模式

### 3.1 问题

- 开启案例1 主库192.168.4.51 半同步复制模式
- 开启案例1 从库192.168.4.52 半同步复制模式
- 开启案例1 从库192.168.4.53 半同步复制模式
- 查看半同步复制模式是否开启

### 3.2 步骤

[Top](#)

实现此案例需要按照如下步骤进行。

**步骤一：查看是否允许动态加载模块，****1) 查看是否允许动态加载模块默认允许**

```

01.  mysql> show variables like 'have_dynamic_loading';
02.  +-----+-----+
03.  | Variable_name | Value |
04.  +-----+-----+
05.  | have_dynamic_loading | YES |
06.  +-----+-----+
07.  1 row in set (0.01 sec)

```

**2) 命令行加载插件，用户需有SUPER权限**

主库上面操作：

```
01.  mysql> INSTALL PLUGIN rpl_semi_sync_master SONAME 'semisync_master.so';
```

从库上面操作：

```
01.  mysql> INSTALL PLUGIN rpl_semi_sync_slave SONAME 'semisync_slave.so';
```

查看系统库下的表，模块是否安装成功：

```

01.  mysql> SELECT PLUGIN_NAME, PLUGIN_STATUS FROM INFORMATION_SCHEMA.PLUGINS W
02.  +-----+-----+
03.  | PLUGIN_NAME | PLUGIN_STATUS |
04.  +-----+-----+
05.  | rpl_semi_sync_master | ACTIVE |
06.  | rpl_semi_sync_slave | ACTIVE |
07.  +-----+-----+
08.  2 rows in set (0.00 sec)

```

**3) 启用半同步复制，在安装完插件后，半同步复制默认是关闭的**

主库上面执行：

[Top](#)

01. mysql> SET GLOBAL rpl\_semi\_sync\_master\_enabled = 1;
02. Query OK, 0 rows affected ( 0.00 sec)

从库上面执行：

01. mysql> SET GLOBAL rpl\_semi\_sync\_slave\_enabled = 1;
02. Query OK, 0 rows affected ( 0.00 sec)

查看半同步复制模式是否启用：

01. mysql> show variables like "rpl\_semi\_sync\_%\_enabled";
02. +-----+-----+
03. | Variable\_name | Value |
04. +-----+-----+
05. | rpl\_semi\_sync\_master\_enabled | ON |
06. | rpl\_semi\_sync\_slave\_enabled | ON |
07. +-----+-----+
08. 2 rows in set ( 0.00 sec)

#### 4) 永久启用半同步复制

主库配置

01. [ root@master51 ~] # vim /etc/my.cnf
02. [ mysql]
03. plugin\_load=rpl\_semi\_sync\_master=semisync\_master.so
04. rpl\_semi\_sync\_master\_enabled=1

从库配置

01. [ root@slave52 ~] # vim /etc/my.cnf
02. [ mysql]
03. plugin\_load=rpl\_semi\_sync\_slave=semisync\_slave.so
04. rpl\_semi\_sync\_slave\_enabled=1

[Top](#)

在高可用架构下，master和slave需同时启动，以便在切换后能继续使用半同步复制

01. [ root@master51 ~] # vim /etc/my.cnf
02. [ mysqld]
03. plugin-load \
04. ="rpl\_semi\_sync\_master=semisync\_master.so;rpl\_semi\_sync\_slave=semisync\_slave.so"
05. rpl\_semi\_sync\_master\_enabled = 1
06. rpl\_semi\_sync\_slave\_enabled = 1

[Top](#)