# MySQL5.6主从复制的配置

**MySQL主从复制的方式:**

MySQL5.6开始主从复制有两种方式：基于日志（binlog）、基于GTID（全局事务标示符）。

**MySQL主从复制（也称A/B复制）的原理:**

(1) Master将数据改变记录到二进制日志(binary log)中，也就是配置文件log-bin指定的文件，这些记录叫做二进制日志事件(binary log events)；

(2) Slave通过I/O线程读取Master中的binary log events并写入到它的中继日志(relay log)；

(3) Slave重做中继日志中的事件，把中继日志中的事件信息一条一条的在本地执行一次，完成数据在本地的存储，从而实现将改变反映到它自己的数据(数据重放)。

**主从配置需要注意的点:**

(1)主从服务器操作系统版本和位数一致；

(2) Master和Slave[**数据库**](http://lib.csdn.net/base/mysql)的版本要一致；

(3) Master和Slave数据库中的数据要一致；

(4) Master开启二进制日志，Master和Slave的server\_id在局域网内必须唯一；

**主从配置的简要步骤:**

1、Master上的配置

(1) 安装数据库；

(2) 修改数据库配置文件，指明server\_id，开启二进制日志(log-bin)；

(3) 启动数据库，查看当前是哪个日志，position号是多少；

(4) 登录数据库，授权数据复制用户（IP地址为从机IP地址，如果是双向主从，这里的还需要授权本机的IP地址，此时自己的IP地址就是从IP地址)；

(5) 备份数据库（记得加锁和解锁）；

(6) 传送备份数据到Slave上；

(7) 启动数据库；

以下步骤，为单向主从搭建成功，想搭建双向主从需要的步骤：

(1) 登录数据库，指定Master的地址、用户、密码等信息（此步仅双向主从时需要）；

(2) 开启同步，查看状态；

2、Slave上的配置

(1) 安装数据库；

(2) 修改数据库配置文件，指明server\_id（如果是搭建双向主从的话，也要开启二进制日志log-bin）；

(3) 启动数据库，还原备份；

(4) 查看当前是哪个日志，position号是多少（单向主从此步不需要，双向主从需要）；

(5) 指定Master的地址、用户、密码等信息；

(6) 开启同步，查看状态。

单向主从环境（也称MySQL A/B复制）的搭建

1、Master（192.168.1.205）和Slave（192.168.1.206）上都安装了相同版本的数据库（mysql-5.6.26.tar.gz），参考《高可用架构篇--第13节--MySQL源码编译安装（CentOS6.6+MySQL5.6）》。

注意：两台数据库服务器的的selinux都要disable（永久关闭selinux，请修改/etc/selinux/config，将SELINUX改为disabled）

2、修改Master的配置文件/etc/my.cnf

[root@edu-mysql-01 ~]# vi /etc/my.cnf

## 在 [mysqld] 中增加以下配置项

## 设置server\_id，一般设置为IP

server\_id=205

## 复制过滤：需要备份的数据库，输出binlog

#binlog-do-db=roncoo

## 复制过滤：不需要备份的数据库，不输出（mysql库一般不同步）

binlog-ignore-db=mysql

## 开启二进制日志功能，可以随便取，最好有含义

log-bin=edu-mysql-bin

## 为每个session 分配的内存，在事务过程中用来存储二进制日志的缓存

binlog\_cache\_size=1M

## 主从复制的格式（mixed,statement,row，默认格式是statement）

binlog\_format=mixed

## 二进制日志自动删除/过期的天数。默认值为0，表示不自动删除。

expire\_logs\_days=7

## 跳过主从复制中遇到的所有错误或指定类型的错误，避免slave端复制中断。

## 如：1062错误是指一些主键重复，1032错误是因为主从数据库数据不一致

slave\_skip\_errors=1062

## 如果需要同步函数或者存储过程

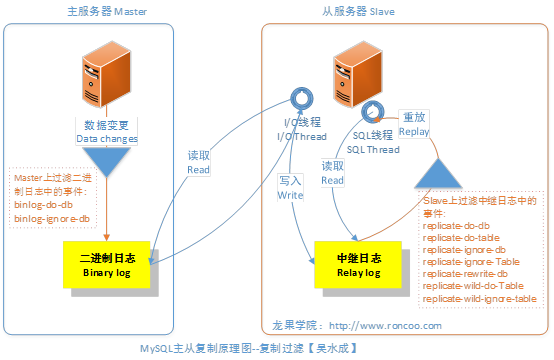
log\_bin\_trust\_function\_creators=true

（如想了解以上参数的更多详细解析，大家可以直接百度参数名）

2.1 复制过滤可以让你只复制服务器中的一部分数据，有两种复制过滤：

(1)     在Master上过滤二进制日志中的事件；

(2)     在Slave上过滤中继日志中的事件。如下：

[](http://static.oschina.net/uploads/space/2016/0405/142244_ZD3J_57136.png)

2.2 MySQL对于二进制日志 (binlog)的复制类型

(1) 基于语句的复制：在Master上执行的SQL语句，在Slave上执行同样的语句。MySQL默认采用基于语句的复制，效率比较高。一旦发现没法精确复制时，会自动选着基于行的复制。

(2) 基于行的复制：把改变的内容复制到Slave，而不是把命令在Slave上执行一遍。从MySQL5.0开始支持。

(3) 混合类型的复制：默认采用基于语句的复制，一旦发现基于语句的无法精确的复制时，就会采用基于行的复制。

3、启动/重启Master数据库服务，登录数据库，创建数据同步用户，并授予相应的权限

[root@edu-mysql-01 ~]# service mysql restart

Shutting down MySQL..[  OK  ]

Starting MySQL..[  OK  ]

[root@edu-mysql-01 ~]# mysql -uroot -p

Enter password:

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

Your MySQL connection id is 1

Server version: 5.6.26-log Source distribution

Copyright (c) 2000, 2015, [**Oracle**](http://lib.csdn.net/base/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.

##创建数据同步用户，并授予相应的权限

mysql> grant replication slave, replication client on \*.\* to 'repl'@'192.168.1.206' identified by 'roncoo.123';

Query OK, 0 rows affected (0.00 sec)

## 刷新授权表信息

mysql> flush privileges;

Query OK, 0 rows affected (0.00 sec)

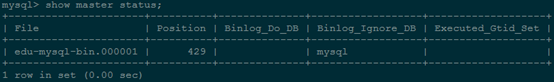
## 查看MySQL现在有哪些用户及对应的IP权限

mysql> select user,host from mysql.user;

[](http://static.oschina.net/uploads/space/2016/0405/142341_vh3m_57136.png)

## 查看position号，记下position号（从机上需要用到这个position号和现在的日志文件)

mysql> show master status;

[](http://static.oschina.net/uploads/space/2016/0405/142427_08rV_57136.png)

Master重启后会修改mysql-bin文件名（序号加1）

5、为保证Master和Slave的数据一致，我们采用主备份，从还原来实现初始数据一致

## 先临时锁表

mysql> flush tables with read lock;

Query OK, 0 rows affected (0.00 sec)

## 这里我们实行全库备份，在实际中，我们可能只同步某一个库，那也可以只备份一个库

 [root@edu-mysql-01 ~]# mysqldump -p3306 -uroot -p --add-drop-table roncoo > /tmp/edu-master-roncoo.sql;

Warning: Using a password on the command line interface can be insecure.

Enter password:

[root@edu-mysql-01 ~]# cd /tmp

[root@edu-mysql-01 tmp]# ll

total 644

-rw-r--r--  1 root root 644266 Dec 20 04:10 edu-master-roncoo.sql

## 注意：实际生产环境中[**大数据**](http://lib.csdn.net/base/hadoop)量（超2G数据）的备份，建议不要使用mysqldump进行比分，因为会非常慢。此时推荐使用 XtraBackup 进行备份。

## 解锁表

mysql> unlock tables;

Query OK, 0 rows affected (0.00 sec)

将Master上备份的数据远程传送到Slave上，以用于Slave配置时恢复数据

[root@edu-mysql-01 ~]# scp /tmp/edu-master-roncoo.sql root@192.168.1.206:/tmp/

root@192.168.1.206's password:

edu-master-roncoo.sql                        100%  629KB 629.2KB/s   00:00

[root@edu-mysql-01 ~]#

6、接下来处理Slave（192.168.1.206），配置文件只需修改一项，其余配置用命令来操作

[root@edu-mysql-02 ~]# vi /etc/my.cnf

## 在 [mysqld] 中增加以下配置项

## 设置server\_id，一般设置为IP

server\_id=206

## 复制过滤：需要备份的数据库，输出binlog

#binlog-do-db=roncoo

##复制过滤：不需要备份的数据库，不输出（mysql库一般不同步）

binlog-ignore-db=mysql

## 开启二进制日志，以备Slave作为其它Slave的Master时使用

log-bin=edu-mysql-slave1-bin

## 为每个session 分配的内存，在事务过程中用来存储二进制日志的缓存

binlog\_cache\_size = 1M

## 主从复制的格式（mixed,statement,row，默认格式是statement）

binlog\_format=mixed

## 二进制日志自动删除/过期的天数。默认值为0，表示不自动删除。

expire\_logs\_days=7

## 跳过主从复制中遇到的所有错误或指定类型的错误，避免slave端复制中断。

## 如：1062错误是指一些主键重复，1032错误是因为主从数据库数据不一致

slave\_skip\_errors=1062

## relay\_log配置中继日志

relay\_log=edu-mysql-relay-bin

## log\_slave\_updates表示slave将复制事件写进自己的二进制日志

log\_slave\_updates=1

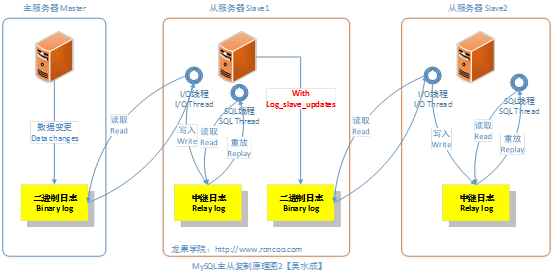
## 防止改变数据(除了特殊的线程)

#read\_only=1

如果Slave为其它Slave的Master时，必须设置bin\_log。在这里，我们开启了二进制日志，而且显式的命名(默认名称为hostname，但是，如果hostname改变则会出现问题)。

relay\_log配置中继日志，log\_slave\_updates表示slave将复制事件写进自己的二进制日志。

当设置log\_slave\_updates时，你可以让slave扮演其它slave的master。此时，slave把SQL线程执行的事件写进行自己的二进制日志(binary log)，然后，它的slave可以获取这些事件并执行它。如下图所示（发送复制事件到其它Slave）：

[](http://static.oschina.net/uploads/space/2016/0405/142519_aLoA_57136.png)

7、保存后重启MySQL服务，还原备份数据

[root@edu-mysql-02 ~]# service mysql restart

Shutting down MySQL..[  OK  ]

Starting MySQL..[  OK  ]

Slave上创建相同库：

create database if not exists roncoo default charset utf8 collate utf8\_general\_ci;

use roncoo;

导入数据

[root@edu-mysql-02 ~]# mysql -uroot -p roncoo < /tmp/edu-master-roncoo.sql

Enter password:

[root@edu-mysql-02 ~]#

8、登录Slave数据库，添加相关参数

（Master的IP、端口、同步用户、密码、position号、读取哪个日志文件)

[root@edu-mysql-02 ~]# mysql -uroot -p

Enter password:

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

Your MySQL connection id is 3

Server version: 5.6.26-log Source distribution

Copyright (c) 2000, 2015, 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.

mysql> change master to master\_host='192.168.1.205', master\_user='repl', master\_password='roncoo.123', master\_port=3306, master\_log\_file='edu-mysql-bin.000001', master\_log\_pos=429, master\_connect\_retry=30;

Query OK, 0 rows affected, 2 warnings (0.01 sec)

上面执行的命令的解释：

master\_host='192.168.1.205'                  ## Master的IP地址

master\_user='repl'                             ## 用于同步数据的用户（在Master中授权的用户）

master\_password='roncoo.123'     ## 同步数据用户的密码

master\_port=3306                              ## Master数据库服务的端口

master\_log\_file='edu-mysql-bin.000001'        ##指定Slave从哪个日志文件开始读复制数据（可在Master上使用show master status查看到日志文件名）

master\_log\_pos=429                         ## 从哪个POSITION号开始读

master\_connect\_retry=30  ##当重新建立主从连接时，如果连接建立失败，间隔多久后重试。单位为秒，默认设置为60秒，同步延迟调优参数。

## 查看主从同步状态

mysql> show slave status\G;

可看到Slave\_IO\_State为空， Slave\_IO\_Running和Slave\_SQL\_Running是No，表明Slave还没有开始复制过程。

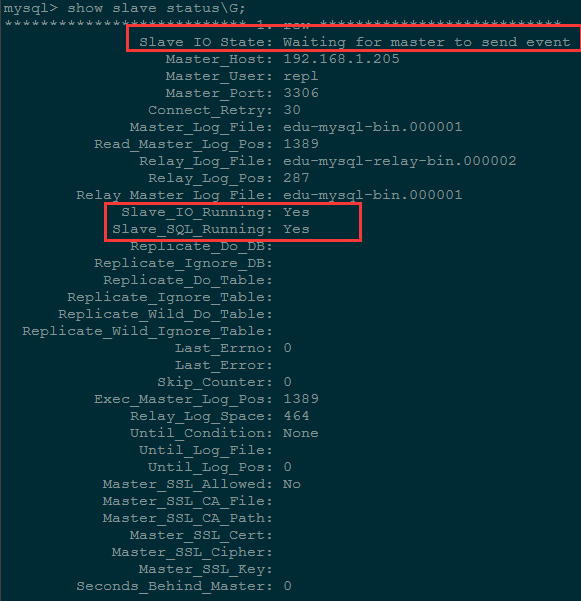
## 开启主从同步

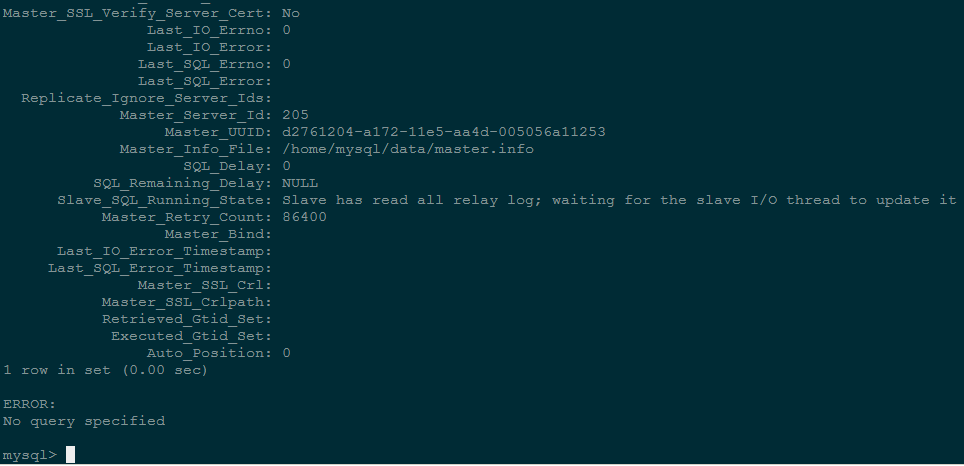
mysql> start slave;

Query OK, 0 rows affected (0.00 sec)

## 再查看主从同步状态

mysql> show slave status\G;

[](http://static.oschina.net/uploads/space/2016/0405/142643_juYb_57136.png)

[](http://static.oschina.net/uploads/space/2016/0405/142653_9UyG_57136.png)

主要看以下两个参数，这两个参数如果是Yes就表示主从同步正常

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

由截图中的主从同步状态信息可以看出，我们配置的主从同步是正常的。

可查看master和slave上线程的状态。在master上，可以看到slave的I/O线程创建的连接：

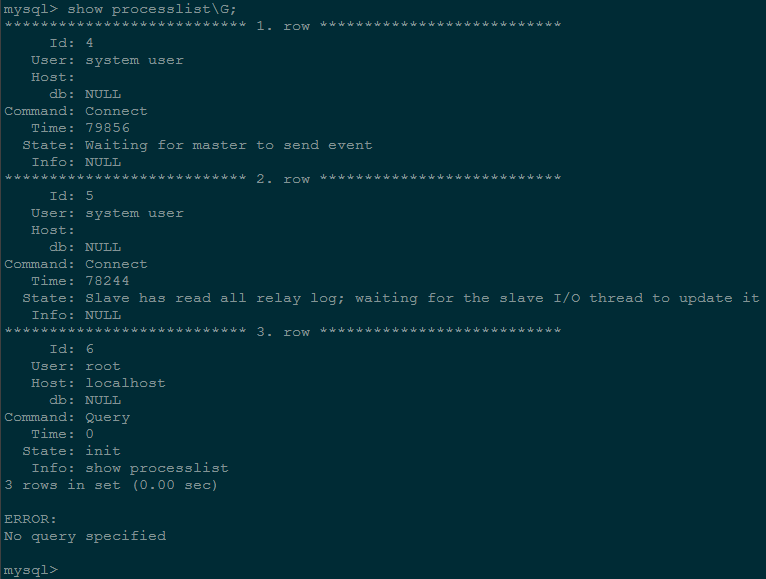
Master : mysql> show processlist\G;

1.row 为处理slave的I/O线程的连接。

 2.row 为处理MySQL客户端连接线程。

 3.row 为处理本地命令行的线程。

Slave : mysql> show processlist\G;

[](http://static.oschina.net/uploads/space/2016/0405/142844_wFo7_57136.png)

1.row 为I/O线程状态。

2.row 为SQL线程状态。

3.row 为处理本地命令行的线程。

9、主从数据复制同步[**测试**](http://lib.csdn.net/base/softwaretest)

(1) 在Master中的roncoo库上变更数据的同步测试；

mysql> INSERT INTO `edu\_user` VALUES (4,'同步测试1','123456'),(5,'同步测试2','123456');

Master中添加完之后，登录Slave中查看数据是否已同步。

(2) 在Master上新建一个ron库

mysql> create database if not exists ron default charset utf8 collate utf8\_general\_ci;

在Slave中查看数据库

mysql> show databases;

最终的测试结果是，在Master中的操作，都成功同步到了Slave。

10、测试过程中，如果遇到同步出错，可在Slave上重置主从复制设置（选操作）：

(1) mysql> reset slave;

(2) mysql> change master to master\_host='192.168.1.205',

master\_user='repl',

master\_password='roncoo.123',

master\_port=3306,

master\_log\_file='edu-mysql-bin.00000x',

master\_log\_pos=xx,

master\_connect\_retry=30;

(此时，master\_log\_file和master\_log\_pos要在Master中用show master status 命令查看)

注意：如果在Slave没做只读控制的情况下，千万不要在Slave中手动插入数据，那样数据就会不一致，主从就会断开，就需要重新配置了。

11、上面所搭建的是单向复制的主从，也是用的比较多的，而双向主从其实就是Master和Slave都开启日志功能，然后在Master执行授权用户（这里授权的是自己作为从服务器，也就是这里的IP地址是Master的IP地址），然后再在Master上进行chang master操作。

MySQL主从数据同步延迟问题的调优

基于局域网的Master/Slave机制在通常情况下已经可以满足“实时”备份的要求了。如果延迟比较大，可以从以下几个因素进行排查：

(1) 网络延迟；

(2) Master负载过高；

(3) Slave负载过高；

一般的做法是使用多台Slave来分摊读请求，再单独配置一台Slave只作为备份用，不进行其他任何操作，就能相对最大限度地达到“实时”的要求了。

两个可以减少主从复制延迟的参数（按需配置）：

MySQL可以指定3个参数，用于复制线程重连主库：--master-retry-count，--master-connect-retry，--slave-net-timeout 。其中 master-connect-retry 和 master-retry-count 需要在 Change Master 搭建主备复制时指定，而 slave-net-timeout 是一个全局变量，可以在 MySQL 运行时在线设置。具体的重试策略为：备库过了 slave-net-timeout 秒还没有收到主库来的数据，它就会开始第一次重试。然后每过 master-connect-retry 秒，备库会再次尝试重连主库。直到重试了 master-retry-count 次，它才会放弃重试。如果重试的过程中，连上了主库，那么它认为当前主库是好的，又会开始 slave-net-timeout 秒的等待。slave-net-timeout 的默认值是 3600 秒，master-connect-retry 默认为 60 秒，master-retry-count 默认为 86400 次。也就是说，如果主库一个小时都没有任何数据变更发送过来，备库才会尝试重连主库。这就是为什么在我们模拟的场景下，一个小时后，备库才会重连主库，继续同步数据变更的原因。

这样的话，如果你的主库上变更比较频繁，可以考虑将 slave-net-timeout 设置的小一点，避免主库 Binlog dump 线程终止了，无法将最新的更新推送过来。当然 slave-net-timeout 设置的过小也有问题，这样会导致如果主库的变更确实比较少的时候，备库频繁的重新连接主库，造成资源浪费。

slave-net-timeout=seconds

参数说明：当Slave从Master数据库读取log数据失败后，等待多久重新建立连接并获取数据，单位为秒，默认设置为3600秒。

在做MySQL Slave的时候经常会遇到很多错误，需要根据具体原因跨过错误继续同步，但有时候是因为网络不稳定、网络闪断造成同步不正常，如果Slave机器非常多的情况下，一个一个登录服务器去stop slave、start slave变得无聊而且重复。从MySQL5.1开始支持的解决方案配置：

master-connect-retry=seconds

参数说明：在主服务器宕机或连接丢失的情况下，从服务器线程重新尝试连接主服务器之前睡眠的秒数。如果主服务器.info文件中的值可以读取则优先使用。如果未设置，默认值为60。

通常配置以上2个参数可以减少网络问题导致的主从数据同步延迟。

一般网络问题的错误是：

[ERROR] Error reading packet from server: Lost connection to MySQL server during query (server\_errno=xxxx)

[ERROR] Slave I/O thread: Failed reading log event, reconnecting to retry, log ‘edu-mysql-bin.000256’  position 23456