122 如何为MySQL搭建一套主从复制架构？（2）

上回已经给大家讲解了如何为MySQL搭建一套主从复制架构，其实搭建一点都不难，相信大家自己照着之前讲解的步骤做，基本都能搭建出来一套主从复制的架构，只要你搭建出来主从复制架构，就可以实现读写分离了。

比如可以用mycat或者sharding-sphere之类的中间件，就可以实现你的系统写入主库，从从库去读取了。

但是现在搭建出来的主从复制架构有一个问题，那就是之前那种搭建方式他默认是一种异步的复制方式，也就是说，主库把日志写入binlog文件，接着自己就提交事务返回了，他也不管从库到底收到日志没有。

那万一此时要是主库的binlog还没同步到从库，结果主库宕机了，此时数据不就丢失了么？即使你做了高可用自动切换，一下子把从库切换为主库，但是里面是没有刚才写入的数据的，所以这种方式是有问题的。

因此一般来说搭建主从复制，都是采取半同步的复制方式的，这个半同步的意思，就是说，你主库写入数据，日志进入binlog之后，起码得确保 binlog日志复制到从库了，你再告诉客户端说本次写入事务成本了是不是？

这样起码你主库突然崩了，他之前写入成功的数据的binlog日志都是到从库了，从库切换为主库，数据也不会丢的，这就是所谓的半同步的意思。

这个半同步复制，有两种方式，第一种叫做AFTER_COMMIT方式，他不是默认的，他的意思是说，主库写入日志到binlog，等待binlog复制到从库了，主库就提交自己的本地事务，接着等待从库返回给自己一个成功的响应，然后主库返回提交事务成功的响应给客户端。

另外一种是现在MySQL 5.7默认的方式，主库把日志写入binlog，并且复制给从库，然后开始等待从库的响应，从库返回说成功给主库了，主库再提交事务，接着返回提交事务成功的响应给客户端。

总而言之，这种方式可以保证你每个事务提交成功之前，binlog日志一定都复制到从库了，所以只要事务提交成功，就可以认为数据在从库也有一份了，那么主库崩溃，已经提交的事务的数据绝对不会丢失的。

搭建半同步复制也很简单，在之前搭建好异步复制的基础之上，安装一下半同步复制插件就可以了，先在主库中安装半同步复制插件，同时还得开启半同步复制功能：

install plugin rpl_semi_sync_master soname 'semisync_master.so';

set global rpl_semi_sync_master_enabled=on;

show plugins;

可以看到你安装了这个插件，那就ok了。

接着在从库也是安装这个插件以及开启半同步复制功能：

install plugin rpl_semi_sync_slave soname 'semisync_slave.so';

set global rpl_semi_sync_slave_enabled=on;

show plugins;

接着要重启从库的IO线程：stop slave io_thread; start slave io_thread;

然后在主库上检查一下半同步复制是否正常运行：show global status like '%semi%';，如果看到了Rpl_semi_sync_master_status的状态是ON，那么就可以了。

到此半同步复制就开启成功了，其实一般来说主从复制都建议做成半同步复制，因为这样配合高可用切换机制，就可以保证数据库有一个在线的从库热备份主库的数据了，而且主要主库宕机，从库立马切换为主库，数据不丢失，数据库还高可用。

End