[mysql主从复制配置](http://blog.chinaunix.net/uid-26610882-id-4083396.html) 2014-01-17 11:14:12

分类： Mysql/postgreSQL

**一.mysql主从备份（复制）的基本原理**  
mysql支持单向、异步复制，复制过程中一个服务器充当主服务器，而一个或多个其它服务器充当从服务器。mysql复制基于主服务器在二进制日志中跟踪所有对数据库的更改(更新、删除等等)。因此，要进行复制，必须在主服务器上启用二进制日志。每个从服务器从主服务器接收主服务器已经记录到其二进制日志的保存的更新。当一个从服务器连接主服务器时，它通知主服务器从服务器在日志中读取的最后一次成功更新的位置。从服务器接收从那时起发生的任何更新，并在本机上执行相同的更新。然后封锁并等待主服务器通知新的更新。从服务器执行备份不会干扰主服务器，在备份过程中主服务器可以继续处理更新。  
　　  
**二.mysql主从备份配置方法**  
在进行mysql主从备份时，最好确保主从服务器的版本兼容。从服务器至少与主服务器版本相同或更高。  
　　  
主机（master）配置：  
1.修改mysql配置文件my.cnf  
在[mysqld]标签下添加以下几行

**[plain]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. log-bin #开启二进制日志
2. server-id=id #主服务器id号
3. binlog-do-db=db\_nameA #指定对db\_nameA记录二进制日志
4. binlog-ignore-db=db\_namB #指定不对db\_namB记录二进制日志

注意：  
log-bin，server-id是配置文件中必须添加的内容。此时主服务器默认对所有数据库进行备份。如果需要特殊指明只对某个数据库进行备份或不备份，则可以加入binlog-do-db和binlog-ignore-db选项。有关（log-bin的详细说明见附录1）  
  
在测试主机上，我们实际添加入如下内容：

**[plain]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. log-bin
2. server-id=1
3. binlog-do-db=test

2.为从服务器添加mysql账户并配置权限  
在主服务器上，必须为从服务器创建一个用来连接主服务器的用户，并设置replication slave权限。所用具体命令如下：

**[sql]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. **grant** replication slave
2. **on** \*.\*
3. **to** '帐号' @ '从服务器IP' identified **by** '密码';

在测试机上，我们实际执行：

**[sql]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. **grant** replication slave
2. **on** \*.\*
3. **to** 'replication'@'%' identified **by** 'sosotest';

这时在mysql库的user表中使用

**[sql]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. **select** \* **from** user **where** user = 'replication' \G;

可以看到Repl\_slave\_priv项对就的值为Y。  
3.导出主服务器数据  
导出主服务器数据，以备之后将其导入从服务器，使主从服务器的初始状态保持一致。  
  
在测试机上，我们实际执行：

**[sql]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. mysqldump test > test.bak

4.记录File 及Position 项的值  
　　重启mysql，使用show master status\G;查看主服务器状态，记录File 及Position 项的值，以便之后对从服务器进行配置。  
　　  
在测试机上，我们实际执行时看到的结果如下：  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
File: simba-bin.000008  
Position: 79  
Binlog\_Do\_DB: test  
Binlog\_Ignore\_DB:   
1 row in set (0.00 sec)  
  
其中File为imba-bin.000008；Position为79。  
  
至此主服务器配置完毕  
  
从机（slave）配置：  
1.修改mysql配置文件my.cnf  
在[mysqld]标签下添加以下面一行：

**[plain]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. server-id=id #从机id

实际中，我们添加：

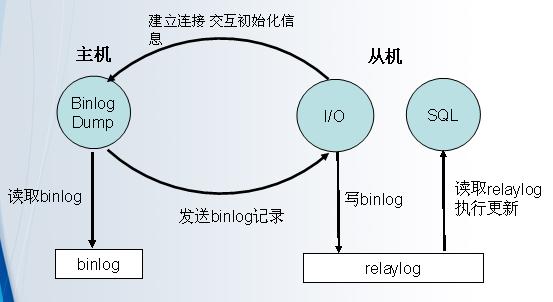
**[plain]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. server-id=2

2.导入主机数据库  
在测试机上，我们实际执行：  
mysql test < test.bak  
  
3.重启mysql数据库并设置相关参数

**[sql]** [view plaincopy](http://blog.csdn.net/qmhball/article/details/8233769)

1. change master **to**
2. master\_host = '10.1.146.133',
3. master\_user = 'replication',
4. master\_password = 'sosotest',
5. master\_log\_file = 'darkstar-bin.000008',
6. master\_log\_pos = 79;
7. slave start;

4.检查数据库查看相关参数  
　　使用show slave status \G;查看相关参数是否正确。  
　　在实际执行时，我们在从机上看到的结果如下（只摘取了部分输出）：  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Slave\_IO\_State: Waiting for master to send event  
Master\_Host: 10.1.146.133  
Master\_User: replication  
Master\_Port: 3306  
Connect\_Retry: 60  
Master\_Log\_File: darkstar-bin.000006  
Read\_Master\_Log\_Pos: 79  
Relay\_Log\_File: simba-relay-bin.000003  
Relay\_Log\_Pos: 171  
Relay\_Master\_Log\_File: darkstar-bin.000006  
Slave\_IO\_Running: Yes  
Slave\_SQL\_Running: Yes  
……  
  
对部分参数的解释：  
Slave\_IO\_State: Waiting for master to send event   
Slave\_IO\_Running: Yes  
Slave\_SQL\_Running: Yes  
以上参数说听slave正常运行，正等待主服务器发来的消息。此时若用netstate命令可以看到从机与主机间已经建立了一条边接。  
  
特别需要注意的两个参数是：  
Master\_Log\_File和Read\_Master\_Log\_Pos。Master\_Log\_File代表主机上用于主备同步的日志文件名，Read\_Master\_Log\_Pos代表上一次成功同步到的日志文件中的位置。  
如果这两项与先前在主服务器上看到的File及Position的值不相符，则无法正确进行同步。  
  
  
**三.主从备份的实现细节**  
mysql使用3个线程来执行复制功能(其中1个在主服务器上，另两个在从服务器上）。当发出start slave时，从服务器创建一个I/O线程，以连接主服务器并让它发送记录在其二进制日志中的语句。主服务器创建一个线程将二进制日志中的内容发送到从服务器。该线程可以即为主服务器上show processlist输出中的Binlog Dump线程。从服务器I/O线程读取主服务器Binlog Dump线程发送的内容并将该数据拷贝到从服务器数据目录中的本地文件中，即中继日志。第3个线程是sql线程，由从服务器创建，用于读取中继日志并执行日志中包含的更新。在从服务器上，读取和执行更新语句被分成两个独立的任务。当从服务器启动时，其I/O线程可以很快地从主服务器索取所有二进制日志内容。  
  
  
1.复制线程状态  
　　通过show slave status\G和show master status可以查看复制线程状态。常见的线程状态有：  
（1）主服务器Binlog Dump线程  
Has sent all binlog to slave; waiting for binlog to be updated  
线程已经从二进制日志读取所有主要的更新并已经发送到了从服务器。线程现在正空闲，等待由主服务器上新的更新导致的出现在二进制日志中的新事件。  
  
（2）从服务器I/O线程状态  
Waiting for master to send event  
线程已经连接上主服务器，正等待二进制日志事件到达。如果主服务器正空闲，会持续较长的时间。如果等待持续slave\_read\_timeout秒，则发生超时。此时，线程认为连接被中断并企图重新连接。  
  
（3）从服务器SQL线程状态  
Reading event from the relay log  
线程已经从中继日志读取一个事件，可以对事件进行处理了。  
Has read all relay log; waiting for the slave I/O thread to update it  
线程已经处理了中继日志文件中的所有事件，现在正等待I/O线程将新事件写入中继日志。  
  
2.复制过程中使用的传递和状态文件  
　　默认情况，中继日志使用host\_name-relay-bin.nnnnnn形式的文件名，其中host\_name是从服务器主机名，nnnnnn是序列号。中继日志与二进制日志的格式相同，并且可以用mysqlbinlog读取。  
　　从服务器在data目录中另外创建两个小文件。这些状态文件默认名为主master.info和relay-log.info。状态文件保存在硬盘上，从服务器关闭时不会丢失。下次从服务器启动时，读取这些文件以确定它已经从主服务器读取了多少二进制日志，以及处理自己的中继日志的程度。  
  
  
**附录1**  
关于log\_bin日志   
　　my.conf 文件中的[mysqld]标签下的log\_bin指定日志文件，如果不提供文件名，mysql将自己产生缺省文件名。mysql会在文件名后面自动添加数字引，每次启动服务时，都会重新生成一个新的二进制文件。此外，使用log-bin-index可以指定索引文件；使用binlog-do-db可以指定记录的数据库；使用binlog-ignore-db可以指定不记录的数据库。注意的是：binlog-do-db和binlog-ignore-db一次只指定一个数据库，指定多个数据库需要多个语句。而且，MySQL会将所有的数据库名称改成小写，在指定数据库时必须全部使用小写名字，否则不会起作用。  
　　以后对数据库每做的一次操作，都会在binlog中有所记录。  
  
  
**附录2**  
试验环境：  
主机  myslq版本 4.1.22  
从机 myslq版本 4.0.23a