**一.需求问题**

假设目前有两台 MySQL 数据库服务器，如何实现这两台机器的数据同步问题？即在一台机器上修改数据库后，另一台机器会同步更新所修改的信息。

**二.解决方案**

查资料发现 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，而另一个或多个其他服务器充当从服务器。

原理是这样的：

主服务器将更新写入二进制日志文件，并维护文件的一个索引来跟踪日志循环。这些日志可以记录发送到从服务器的更新。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。从服务器接受从那时起发生的任何更新，然后封锁并等待主服务器通知新的更新。

**2.1 测试环境**

1. Master : 192.168.7.67 (CentOS 5.5  x86\_64 )   MySQL Version  :  5.0.77
2. Slave: 192.168.56.103 (CentOS 5.3 i386)  MySQL  Version : 5.0.45

备注：

Master 和 slave 端的 MySQL 版本最好要一样的，或者 Master 端的版本高于 Slave 端

**2.2 配置过程**

**2.2.1 Master 端设置**

开启 MySQL 服务并新建一个测试数据库 abc:

1. root@camlit ~: /etc/init.d/mysqld start
2. jian.ma@camlit  ~: mysql -u root -p
3. Enter **password**: xxxx
4. Welcome **to** the MySQL monitor.  Commands **end** **with** ; or \g.
5. Your MySQL **connection** id **is** 3
6. Server version: 5.0.77 Source distribution
8. Type 'help;' or '\h' **for** help. Type '\c' **to** clear the buffer.
10. mysql> **create** **database** abc;
11. Query OK, 1 row affected (0.31 sec)
13. ###创建一个用来同步的用户，指定只能在 192.168.56.103 登录
14. ###REPLICATION SLAVE: Enable replication slaves **to** **read** **binary** log events **from** the master
16. mysql> **grant** replication slave **on** \*.\* **to** 'test1'@'192.168.56.103' identified **by** 'test1';
17. Query OK, 0 **rows** affected (0.16 sec)

修改配置文件：

1. root@camlit ~: vi /etc/my.cnf

备注：在修改配置文件之前做好该文件的备份工作。

1. [mysqld]
2. datadir=/var/lib/mysql
3. socket=/var/lib/mysql/mysql.sock
4. user=mysql
5. old\_passwords=1
7. ##增加下面内容
8. server\_id=1###1 表示 master, 2 表示 slave   binlog-do-db=abc ###需要同步的数据库，如果有多个数据库，每个数据库一行  binlog-**ignore**-db=mysql###不需要同步的数据库 log-bin=mysql-bin
10. [mysqld\_safe]
11. log-error=/var/log/mysqld.log
12. pid-file=/var/run/mysqld/mysqld.pid

重启服务：

1. root@camlit ~: /etc/init.d/mysqld restart

**2.2.2  Slave 端设置**

和 master 端一样创建一个相同的数据库: abc

1. Enter **password**:
2. Welcome **to** the MySQL monitor.  Commands **end** **with** ; or \g.
3. Your MySQL **connection** id **is** 5
4. Server version: 5.0.45-log Source distribution
6. Type 'help;' or '\h' **for** help. Type '\c' **to** clear the buffer.
8. mysql> **create** **database** abc;
9. Query OK, 1 row affected (0.31 sec)

修改配置文件：

1. root@test2 ~: vi /etc/my.cnf
2. [mysqld]
3. datadir=/var/lib/mysql
4. socket=/var/lib/mysql/mysql.sock
5. user=mysql
6. old\_passwords=1
8. ###增加下面内容
9. server\_id=2   log-bin=mysql-bin   master-host=192.168.7.67   master-user=test1   master-**password**=test1   master-port=3306   master-**connect**-retry=10  ###连接次数  replicate-do-db=abc   ###接受的数据库名  replicate-**ignore**-db=mysql  ###不要接受的数据库
11. [mysqld\_safe]
12. log-error=/var/log/mysqld.log
13. pid-file=/var/run/mysqld/mysqld.pid

重启服务：

1. root@test2~: /etc/init.d/mysqld restart

备注：

配置成功 后会在 mysql 目录(/var/lib/mysql/)下生成 master.info 文件，如果要更改 slave 设置，要先将该文件删除才会起作用。

进入 mysql，输入下面命令：

1. root@test2~: mysql -u root -p
2. Enter **password**:
3. Welcome **to** the MySQL monitor.  Commands **end** **with** ; or \g.
4. Your MySQL **connection** id **is** 4
5. Server version: 5.0.45-log Source distribution
7. Type 'help;' or '\h' **for** help. Type '\c' **to** clear the buffer.
9. mysql> slave start;
10. Query OK, 0 **rows** affected, 1 warning (0.00 sec)
11. ###查看同步情况
12. mysql > show slave status;  或 show master status;

2.3 结果测试

在 Master 端进行数据库 abc 的一些操作，如下所示：

1. jian.ma@camlit ~: mysql  -u root -p
2. Enter **password**:
3. Welcome **to** the MySQL monitor.  Commands **end** **with** ; or \g.
4. Your MySQL **connection** id **is** 3
5. Server version: 5.0.77-log Source distribution
7. Type 'help;' or '\h' **for** help. Type '\c' **to** clear the buffer.
9. mysql> use abc;
10. **Database** changed
11. mysql> **create** **table** test1 (IP **VARCHAR**(20),USER **VARCHAR**(100), MAIL
12. **VARCHAR**(100));
13. Query OK, 0 **rows** affected (1.20 sec)
14. mysql> **insert** **into** test1(IP,USER,MAIL) **values**('192.168.7.66', 'test', 'test@test.com.cn');
15. Query OK, 1 row affected (0.06 sec)

在 Slave 端查看是否能够更新：

1. root@test2 ~: mysql -u root -p
2. Enter **password**:
3. Welcome **to** the MySQL monitor.  Commands **end** **with** ; or \g.
4. Your MySQL **connection** id **is** 6
5. Server version: 5.0.45-log Source distribution
7. Type 'help;' or '\h' **for** help. Type '\c' **to** clear the buffer.
9. mysql> show databases;
10. +--------------------+
11. | **Database**   |
12. +--------------------+
13. | information\_schema |
14. | foo|
15. | mysql  |
16. | test   |
17. |abc  |
18. +--------------------+
19. 5rows in **set** (0.00 sec)
20. mysql> use abc;
21. Reading **table** information **for** completion **of** **table** and **column** names
22. You can turn **off** this feature **to** get a quicker startup **with** -A
23. **Database** changed
24. mysql> show tables;
25. +---------------+
26. | Tables\_in\_abc |
27. +---------------+
28. | test1 |
29. +---------------+
30. 1 row in **set** (0.03 sec)
31. mysql> **select** \* **from** test1;
32. +--------------+------+------------------+
33. | IP   | USER | MAIL |
34. +--------------+------+------------------+
35. | 192.168.7.66 | test | test@test.com.cn |
36. +--------------+------+------------------+
37. 1 row in **set** (0.00 sec)

从上面的结果可以看到 Master 端的数据可以同步到 Slave 端里面。说明此时主从数据库的同步问题已经成功解决。

**三.补充资料**

关于如何连接到远程 MySQL 问题，可以采取下面的步骤：

首先先登录到远程机器：

1. jian.ma@camlit ~: ssh root@192.168.56.103
2. password: xxx
3. root@test2 ~:

编辑配置文件：

1. root@test2 ~: vi /etc/my.cnf

增加下面一行内容：

1. [mysqld]
2. datadir=/var/lib/mysql
3. socket=/var/lib/mysql/mysql.sock
4. user=mysql
5. old\_passwords=1
6. bind-address=192.168.56.103###此 IP 地址为 MySQL 本机的 IP 地址
7. [mysqld\_safe]
8. log-error=/var/log/mysqld.log
9. pid-file=/var/run/mysqld/mysqld.pi

重启服务：

1. root@test2 ~: /etc/init.d/mysqld restart

创建测试数据库：

1. root@test2 ~: mysql -u root -p
2. Enter password:
3. Welcome to the MySQL monitor.  Commands end with ; or \g.
4. Your MySQL connection id is 2
5. Server version: 5.0.45 Source distribution
7. Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
9. mysql> create database foo ;
10. Query OK, 1 row affected (0.00 sec)
11. ###增加用户 test123 从任何主机登录到 MySQL
12. mysql> grant all privileges on \*.\* to 'test123'@'%' identified by 'test123' with grant  option;
13. Query OK, 0 rows affected (0.00 sec)
14. ###增加用户 test1 从 192.168.7.67 主机登录到 MySQL
15. mysql> grant all privileges on foo.\* to 'test1'@'192.168.7.67' identified by 'test1' with grant   option;
16. Query OK, 0 rows affected (0.00 sec)

如果有防火墙的设置的话，可以如下设置：

1. root@test2 ~: iptables -A INPUT -i eth0 -s 192.168.7.67 -p tcp --dport 3306 -j ACCEPT
2. root@test2~: /etc/init.d/iptales save

最后在客户端就可以输入下面命令来远程进入 MySQL 数据库：

1. jian.ma@camlit ~: mysql -u test1 -h 192.168.56.103 -p
2. Enter password:
3. Welcome to the MySQL monitor.  Commands end with ; or \g.
4. Your MySQL connection id is 13
5. Server version: 5.0.45 Source distribution
6. Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysqld.pl内容如下：

1. #!/usr/bin/perl
3. #This script is used to check if the mysql replication is ok
5. use strict;
6. use DBI;
7. use POSIX "strftime";
9. my $host           = "192.168.56.103";
10. my $user           = "test1";
11. my $passwd         = "test1";
12. my $port           = "3306";
13. my $max\_behind     = "120";
14. my $check\_log      = "./mysql\_check.log";

17. #Open the log file
18. open (FH, ">> $check\_log") or die $!;
20. #Connect the mysql server
21. my $dbh = &MysqlConnect ($host, $port, $user, $passwd);

24. #Get slave sql status
25. my $slave\_status = &MysqlQuery( $dbh, 'show slave status');
26. print FH "Error: SQL Query Error:" . $dbh->errstr;

29. my $slave\_IO              = $slave\_status->{Slave\_IO\_Running};
30. my $slave\_SQL             = $slave\_status->{Slave\_SQL\_Running};
31. my $seconds\_behind\_master = $slave\_status->{Seconds\_Behind\_Master};
32. my $now\_time              = POSIX::strftime ("[%Y-%m-%d %H:%M:%S]", localtime);

35. print "Check the Slave MySQL stauts....\n";
36. print "\_" x 50, "\n";
37. print "Time:\t\t\t$now\_time\n";
38. print "Slave IO Running:\t\t$slave\_IO\n";
39. print "Slave SQL Running::\t\t$slave\_SQL\n";
40. print "Behind Master Seconds:\t\t$seconds\_behind\_master\n";

43. if ($seconds\_behind\_master > $max\_behind){
44. print "Slave SQL Server is far behind master ";
45. }

48. #---Functions----#
49. sub MysqlConnect  {
51. my ($host, $port, $user, $passwd) = @\_;
52. my $dsn  = "DBI:mysql:host=$host;port=$port";
53. return DBI->connect($dsn, $user, $passwd, {RaiseError => 1});
54. }

57. sub MysqlQuery {
59. my ($dbh , $query) = @\_;
60. my $sth = $dbh->prepare($query);
61. my $res = $sth->execute;
62. return undef unless $res;
63. my $row = $sth->fetchrow\_hashref;
64. $sth->finish;
65. return $row;
66. }