MySQL基于binlog的主从复制原理及搭建

从库只读，用来分担主库的读压力。

利用binlog，主库将自己的binlog复制到从库，从库应用主库传来的binlog，写入数据，实现主从数据同步。因而在从库可以查询到主库的数据，主库则只负责增删改等DML, DDL操作。

操作步骤 ：

环境 ：

主 ：192.168.40.122

从 ：192.168.40.121

1、在主从服务器上都装上MySQL数据库

2. 配置主库

1） 在Master MySQL上创建一个用户‘slave’，并允许其他Slave服务器可以通过远程访问Master，通过该用户读取二进制日志，实现数据同步。

```
grant replication slave on *.* to 'slave'@'%' identified by 'mysteel';
```

2） 修改主库配置文件，启动二进制日志，在[mysqld]下增添以下 ：

```
      server-id=122   //给数据库服务的唯一标识，一般设置为服务器Ip的末尾号
  log-bin=master-bin
  log-bin-index=master-bin.index
      binlog-ignore-db=mysql
       ## 复制过滤：也就是指定哪个数据库不用同步（mysql库一般不同步）
```

3）查看binlog状态，重启MySQL服务

```
show master status;
/etc/rc.d/mysqld restart
```

3. 配置从库

1） 修改从库配置文件，在[mysqld]下增添以下 ：

```
     [mysqld]
     basedir=/usr/local/mysql
     datadir=/usr/local/mysql/data
     port=3306
 server-id=121  #服务器唯一ID，默认是1，一般取IP最后一段
 relay-log-index=slave-relay-bin.index
 relay-log=slave-relay-bin
     log-bin=slave-bin
     ## 开启二进制日志功能，以备Slave作为其它Slave的Master时使用
```

2）重启MySQL服务

```
/etc/rc.d/mysqld restart
```

如果主库里面已经有数据了，从库里面还没有，则需要将主库全备考到备库，并在备库应用这些备份，使得主库和备库的所有数据一致，才能复制成功（感觉这和Oracle 10g 的物理DG RMAN duplicate 很相似）

```
/usr/local/mysql/bin/mysqldump --single-transaction -u root -pmysteel --all-databases  > backup.sql
```

3）连接主库 ：

```
change master to master_host='192.168.40.122',   //Master服务器Ip
master_port=3306,
master_user='slave',
master_password='mysteel',
master_log_file='mysql-bin.000002', //Master服务器产生的日志，指定Slave从哪个日志文件开始读复制数据
master_log_pos=32475;   //Master服务器产生的pos点，指定Slave从哪个POSITION号开始读
masterconnectretry=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。

00:16:00 (none)> change master to master_host='192.168.40.122', master_port=3306, master_user='slave', master_password='mysteel',
master_log_file='mysql-bin.000002', master_log_pos=32475;
Query OK, 0 rows affected, 2 warnings (0.08 sec)

00:16:27 (none)>
```

4）启动从库，检查主从同步

```
start slave;
00:16:27 (none)> start slave;
Query OK, 0 rows affected (0.03 sec)
00:16:39 (none)>
show slave status\G;
00:16:54 (none)> show slave status \G;
*************************** 1. row ***************************
               Slave_IO_State: Connecting to master
                  Master_Host: 192.168.40.122
                  Master_User: slave
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000002
          Read_Master_Log_Pos: 32475
               Relay_Log_File: mysql-relay.000001
                Relay_Log_Pos: 4
        Relay_Master_Log_File: mysql-bin.000002
             Slave_IO_Running: Connecting
            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: 32475
              Relay_Log_Space: 154
              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: NULL
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 1045
                Last_IO_Error: error connecting to master 'slave@192.168.40.122:3306' - retry-time: 60  retries: 1
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 0
                  Master_UUID:
             Master_Info_File: /mysqldata/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: 170719 00:16:39
     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)

ERROR:
No query specified

00:16:59 (none)>
```

注：Slave_IO及Slave_SQL进程必须正常运行，即YES状态，否则都是错误的状态(如：其中一个NO均属错误)。

over

这时可以在主库中进行测试，在主库创建个库，表，在从库看能否查询到。

编写一shell脚本，用nagios监控slave的两个yes（Slave_IO及Slave_SQL进程），如发现只有一个或零个yes，就表明主从有问题了，发短信警报吧。

参考 ： http://www.cnblogs.com/alvin_xp/p/4162249.html

编写一shell脚本，用nagios监控slave的两个yes（Slave_IO及Slave_SQL进程），如发现只有一个或零个yes，就表明主从有问题了，发短信警报吧。

Slave_IO_Running: Connecting

原因 ：

1. 密码错误
2. 网络不通
3. pos不对

reset slave; ：重置从库

参考 ：http://blog.csdn.net/i_bruce/article/details/17055135