2.2备份的策略：

完全备份+增量备份+二进制日志

演示备份的过程;

2.3 先给数据库做完全备份：

代码如下:

[root@www ~]# mysqldump -uroot --single-transaction --master-data=2 --databases hellodb > /backup/hellodb\_`date +%F`.sql

--single-transaction: 基于此选项能实现热备InnoDB表；因此，不需要同时使用--lock-all-tables；   
--master-data=2  记录备份那一时刻的二进制日志的位置，并且注释掉，1是不注释的   
--databases hellodb 指定备份的数据库

然后回到mysql服务器端，

2.4回到mysql服务器端更新数据

代码如下:

mysql> create table tb1(id int); 创建表   
mysql> insert into tb1 values (1),(2),(3);  插入数据，这里只做演示，随便插入了几个数据

2.5先查看完全备份文件里边记录的位置：

代码如下:

[root@www backup]# cat hellodb\_2013-09-08.sql | less   
-- CHANGE MASTER TO MASTER\_LOG\_FILE='mysql-bin.000013', MASTER\_LOG\_POS=15684; 记录了二进制日志的位置

2.6 在回到服务器端：

代码如下:

mysql> show master status; 显示此时的二进制日志的位置   
 从备份文件里边记录的位置到我们此时的位置，即为增量的部分   
+------------------+----------+--------------+------------------+   
| File             | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |   
+------------------+----------+--------------+------------------+   
| mysql-bin.000004 |      15982 |              |                  |   
+------------------+----------+--------------+------------------+

2.7做增量备份

代码如下:

[root@www backup]# mysqlbinlog --start-position=15694 --stop-position=15982  
/mydata/data/mysql-bin.000013 > /backup/hellodb\_`date +$F\_%H`.sql

2.8再回到服务器

代码如下:

mysql> insert into tb1 values (4),(5); 在插入一些数值   
mysql> drop database hellodb;   删除hellodb库

2.9导出这次得二进制日志：

代码如下:

[root@www backup]# mysqlbinlog --start-position=15982 /mydata/data/mysql-bin.000013 查看删除操作时二进制日志的位置   
[root@www backup]# mysqlbinlog --start-position=15982 --stop-position=16176 /mydata/data/mysql-bin.000013 > /tmp/hellodb.sql  //导出二进制日志

2.10先让mysql离线

回到服务器端：

代码如下:

mysql> set sql\_log\_bin=0;  关闭二进制日志   
mysql> flush logs; 滚动下日志

2.11模拟数据库损坏

代码如下:

mysql> drop database hellodb;

2.12开始恢复数据：

代码如下:

[root@www ]# mysql < /backup/hellodb\_2013-09-08.sql  //导入完全备份文件   
[root@www ]# mysql < /backup/hellodb\_2013-09-08\_05.sql //导入增量备份文件   
[root@www ]# mysql< hellodb.sql //导入二进制文件

验证完成，显示结果为我们预想的那样

注：

1、真正在生产环境中，我们应该导出的是整个mysql服务器中的数据，而不是单个库，因此应该使用--all-databases  
2、在导出二进制日志的时候，可以直接复制文件即可，但是要注意的是，备份之前滚动下日志。  
3、利用lvm快照实现几乎热备的数据备份与恢复

3.1策略：

完全备份+二进制日志；

3.2准备：

注：事务日志必须跟数据文件在同一个LV上；

3.3创建lvm Lvm的创建这里就不多说了，想了解话点击[http://www.jb51.net/LINUXjishu/105937.html](http://www.jb51.net/LINUXjishu/105937.html" \t "_blank)

3.4 修改mysql主配置文件存放目录内的文件的权限与属主属组，并初始化mysql

代码如下:

[root@www ~]# mkdir /mydata/data             //创建数据目录   
[root@www ~]# chown mysql:mysql /mydata/data  //改属组属主   
[root@www ~]#   
[root@www ~]# cd /usr/local/mysql/    //必须站在此目录下         
[root@www mysql]# scripts/mysql\_install\_db --user=mysql --datadir=/mydata/data  //初始化mysql

3.5修改配置文件：

代码如下:

vim /etc/my.cof   
datadir=/mydata/data   添加数据目录   
sync\_binlog = 1  开启此功能

3.6 启动服务

代码如下:

[root@www mysql]# service mysqld start   
mysql> set session sql\_log\_bin=0;  关闭二进制日志   
mysql> source /backup/all\_db\_2013-09-08.sql   读取备份文件

3.7回到mysql服务器：

代码如下:

mysql> FLUSH TABLES WITH READ LOCK; 请求读锁   
注：不要退出，另起一个终端：   
mysql> SHOW MASTER STATUS;          查看二进制文件的位置   
+------------------+----------+--------------+------------------+   
| File             | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |   
+------------------+----------+--------------+------------------+   
| mysql-bin.000004 |      107 |              |                  |   
+------------------+----------+--------------+------------------+   
1 row in set (0.00 sec)   
mysql> FLUSH LOGS;  建议滚动下日志。这样备份日志的时候就会很方便了

3.8导出二进制文件，创建个目录单独存放

代码如下:

[root@www ~]# mkdir /backup/limian   
[root@www ~]# mysql -e 'show master status;' > /backup/limian/binlog.txt   
[root@www ~]#

3.9为数据所在的卷创建快照：

代码如下:

[root@www ~]# lvcreate -L 100M -s -p r -n mysql\_snap /dev/myvg/mydata

回到服务器端，释放读锁

代码如下:

mysql> UNLOCK TABLES;   
[root@www ~]# mount /dev/myvg/mysql\_snap /mnt/data   
[root@www data]# cp \* /backup/limian/   
[root@www data]#lvremove /dev/myvg/mylv\_snap

3.10更新数据库的数据，并删除数据目录先的数据文件，模拟数据库损坏

代码如下:

mysql>  create table limiantb (id int,name CHAR(10));   
mysql> insert into limiantb values (1,'tom');   
[root@www data]# mysqlbinlog --start-position=187 mysql-bin.000003 > /backup/limian/binlog.sql   
[root@www backup]# cd /mydata/data/   
[root@www data]#  rm -rf \*   
[root@www ~]# cp -a /backup/limian/\* /mydata/data/   
[root@www data]# chown mysql:mysql \*

3.11测试

启动服务

代码如下:

[root@www data]# service mysqld start   
[root@www data]# mysql 登陆测试   
mysql> SHOW DATABASES;   
mysql> SET sql\_log\_bin=0  
mysql> source/backup/limian/binlog.sql; #二进制恢复   
mysql> SHOW TABLES;         #查看恢复结果   
mysql> SET sql\_log\_bin=1;   #开启二进制日志

注：此方式实现了接近于热备的方式备份数据文件，而且数据文件放在lvm中可以根据数据的大小灵活改变lvm的大小，备份的方式也很简单。

## MYSQL增量备份方案

１.在 /etc/my.cnf 下面设置开启bin-log

         编辑

         vim /etc/my.cnf

         [mysqld]

         binlog\_format       = MIXED                 //binlog日志格式

         log\_bin             =目录/mysql-bin.log    //binlog日志名

         expire\_logs\_days    = 7                //binlog过期清理时间

         #max\_binlog\_size    100m                    //binlog每个日志文件大小 最大值和默认是１个Ｇ

         binlog-do-db=game     #需要备份的数据库名，如果备份多个数据库，重复设置这个选项即可

         binlog-do-db=platform #

         #binlog-ignore-db=不需要备份的数据库，如果备份多个数据库，重复设置这个选项即可



２.创建远程登陆数据库帐号

         grant all privileges on game.\* to 'jack'@'192.168.10.56' identified by 'admin'

         grant all privileges on platform.\* to 'jack'@'192.168.10.56' identified by 'admin'

3.将服务器上的数据，导入一份进远程服务器中

         mysqldump -ujack -padmin -h192.168.10.38 > 备份数据库地址(/home/mysql/db/mysql.sql);

4.写Ｌinux 计划任务:

         vim /etc/crontab

在底部加入

         0 3 \* \* \* root sh /home/wwwroot/sh/mysqlbackup.sh #每天3点以root帐号的权限执行目录下的sh

重启服务

         /etc/init.d/crond restart

---------------------------------------------------

如果遇到了问题，请查看一下

select super\_priv,user from user where host='192.168.10.39';

update user set super\_priv='Y' where user='game1';

#! /bin/Bash

##################

# 2013-9-24 #

# author jackluo #

##################

db\_user= #远程mysql用户名

db\_pwd= #远程mysql密码

db\_host= #远程ip

mysql\_bin=/usr/local/mysql/bin/ #mysql 执行【招待】目录

mysqlbindir=/usr/local/mysql/var/ #binlog日志存放目录

dailyBackupDir=/root/mysql/ #存放，导入成功的日志

binlogname="mysql-bin" #binlog日志前缀

cd $mysqlbindir

fileList=`cat mysql-bin.index`

iCounter=0

for file in $fileList

do

iCounter=`expr $iCounter + 1 `

echo ${file}

done

nextNum=0

for file in $fileList

do

binLogName=`basename $file `

nextNum=`expr $nextNum + 1 `

if [[ $nextNum == $iCounter ]]; then

echo "Skip lastest!" > /dev/null

else

dest=$dailyBackupDir/$binLogName

if [[ -e $dest ]]; then #检查是否已导入过了

echo "Skip exist $binLogName!" > /dev/null

else

${mysql\_bin}mysqlbinlog $binLogName|mysql -h${db\_host} -u${db\_user} -p${db\_pwd} #写入远程数据库

if [[ ! -d ${dailyBackupDir} ]];then

mkdir -p ${dailyBackupDir}

fi

touch ${dest}　#创建日志文件

find . -type f -ctime +6 -exec rm {} \; #删除６天后的日志文件

fi

fi

done