|  |  |  |  |
| --- | --- | --- | --- |
|  | **文 档 编 号** | **产品版本** | **密级** |
|  |  |  |
|  | | **共 页** |

数据库备份恢复文档

**北京瑞金麟网络技术服务有限公司**

**Rkylin Network Technology & Service Co. Ltd.**

**二○一五年十月**

改 版 履 历 表

|  |  |  |  |
| --- | --- | --- | --- |
| **版数** | **发行年月** | **作成者** | **变 更 内 容 概 要** |
| 1.0 | 2015.10.26 | 胡成龙 | 初版 |
|  |  |  |  |
|  |  |  |  |

目录

[1 引言 4](#_Toc433375276)

[1.1 编写目的 4](#_Toc433375277)

[1.2 项目背景 4](#_Toc433375278)

[1.3 目标读者 4](#_Toc433375279)

[1.4 参考文档 4](#_Toc433375280)

[1.5 定义 4](#_Toc433375281)

[2 备份方案 5](#_Toc433375282)

[2.1 备份方案 5](#_Toc433375283)

[2.2 备份脚本 5](#_Toc433375284)

[2.3 备份有效性检查 5](#_Toc433375285)

[3 恢复方案 6](#_Toc433375286)

[3.1 恢复方案 6](#_Toc433375287)

[3.2 恢复脚本 6](#_Toc433375288)

[3.3 恢复定期计划 6](#_Toc433375289)

# 引言

## 编写目的

本文档描述对于MySQL数据库的备份恢复策略。

## 项目背景

瑞金麟网络技术服务有限公司目前是中国领先的电子商务一体化服务公司，致力于为品牌商和零售商提供完善的电子商务和互动营销服务。2015年瑞金麟网络技术服务有限公司开始大力发展瑞金麟金融产品中心，力求在电商领域从信息流与资金流形成闭环；在此需求下，需要建立起以虚拟账户系统和资金清结算系统为基础、交易风控系统与运营平台为之保驾护航、代收付系统与智能多渠道路由为能力，最终提供托管记账、网关支付、代收付、BI决策等功能的现代化支付体系。

## 目标读者

需求编写团队、系统设计团队、开发团队、测试团队、运维团队，及所有参与的相关人员。

## 参考文档

Percono XtraBackup 官网：http://www.percona.com/docs/wiki/percona-xtrabackup:start

## 定义

|  |  |  |
| --- | --- | --- |
| 序号 | 定义 | 解释说明 |
| 1 | Mysqldump | MySQL自带的在线逻辑备份工具。 |
| 2 | Mysqlbinlog | MySQL自带的查看二进制日志工具。 |
| 3 | Percono XtraBackup | Percona公司开发的对MySQL在线物理备份工具。 |
| 4 | mysqlbackup | 企业版备份工具，热备，物理备份，备份速度是mysqldump的3.5倍，恢复速度是mysqldump的16倍。 |

# 备份方案

## 备份方案

1. 数据量在几个GB以内，使用mysqldump每天22:30对数据库(主备)做全备，每天12:00对单独的account库做备份。
2. 数据量超过10GB，使用Percono XtraBackup或mysqlbackup工具对数据库(主备)做全备，每周日22:30执行。

## 备份脚本

1. mysql\_backup.sh

#!/bin/bash

#### Make DIR####

mkdir -p /mydata/mysql/backup/log

mkdir -p /mydata/mysql/backup/mysqldump/{all-database,databases,tables}

LogFile=/mydata/mysql/backup/log/fullbackup.log

BakDir=/mydata/mysql/backup/mysqldump

####DECLARE variable####

Date=`date +%Y%m%d`

DumpFile=all-database\_$Date

DumpRoutineEvent=routines\_events\_$Date

DumpTable=tables\_$Date

DumpDatabase=database\_$Date

###mysqldump databases&&tables####

mysqldump='/mydata/mysql/install/bin/mysqldump'

#connect\_info='-u rkylinadmin -p'Rkylin\_web!1' -P3316'

connect\_info='-h127.0.0.1 -P3316'

#########all-databases backup#####

Begin=`date +"%Y-%m-%d %H:%M:%S"`

cd $BakDir/all-database

$mysqldump $connect\_info --quick --all-databases --flush-logs --single-transaction --master-data=2 > $DumpFile.sql

if [ $? -eq 0 ]; then

Last=`date +"%Y-%m-%d %H:%M:%S"`

echo "all\_database backup start:$Begin finish:$Last $DumpFile.sql succ" >> $LogFile

/bin/tar czvf $DumpFile.tar.gz $DumpFile.sql

/bin/rm -rf $DumpFile.sql

else

echo "all\_database backup start:$Begin failed!!!" >> $LogFile

exit 1

fi

$mysqldump $connect\_info --all-databases --no-data --no-create-db --no-create-info --routines --events > $DumpRoutineEvent.sql

if [ $? -eq 0 ]; then

echo "routines\_events backup succ" >> $LogFile

/bin/tar czvf $DumpRoutineEvent.tar.gz $DumpRoutineEvent.sql

/bin/rm -rf $DumpRoutineEvent.sql

else

echo "routines\_events backup failed!!!" >> $LogFile

exit 1

fi

#########database&&tables backup#####

Begin\_database=`date +"%Y-%m-%d %H:%M:%S"`

mkdir -p $BakDir/databases/$DumpDatabase

for DATABASE in `/mydata/mysql/install/bin/mysql $connect\_info -e "show databases"|egrep -v "Database|information\_schema|mysql|test|performance\_schema|sbtest"`

do

$mysqldump $connect\_info $DATABASE>$BakDir/databases/$DumpDatabase/$DATABASE.sql

if [ $? -eq 0 ]; then

Last\_database=`date +"%Y-%m-%d %H:%M:%S"`

echo "$DATABASE backup start:$Begin\_database finish:$Last\_database $DATABASE.sql succ" >> $LogFile

else

echo "$DATABASE backup start:$Begin\_database failed!!!" >> $LogFile

exit 1

fi

Begin\_table=`date +"%Y-%m-%d %H:%M:%S"`

mkdir -p $BakDir/tables/$DumpTable/$DATABASE

for TABLE in `/mydata/mysql/install/bin/mysql $connect\_info -e "USE $DATABASE;SHOW TABLES"|grep -v Tables\_in\_$DATABASE`

do

$mysqldump $connect\_info $DATABASE $TABLE >$BakDir/tables/$DumpTable/$DATABASE/$TABLE.sql

done

cd $BakDir/tables/$DumpTable

/bin/tar czvf $DATABASE.tar.gz $DATABASE

/bin/rm -rf $DATABASE

if [ $? -eq 0 ]; then

Last\_table=`date +"%Y-%m-%d %H:%M:%S"`

echo "$DATABASE".all\_tables" backup start:$Begin\_table finish:$Last\_table $GZDumpTable succ" >> $LogFile

else

echo "$DATABASE".all\_tables" backup start:$Begin\_table failed!!!" >> $LogFile

exit 1

fi

done

cd $BakDir/tables

/bin/tar czvf $DumpTable.tar.gz $DumpTable

/bin/rm -rf $DumpTable

cd $BakDir/databases

/bin/tar czvf $DumpDatabase.tar.gz $DumpDatabase

/bin/rm -rf $DumpDatabase

添加cron

# crontab –e

##每天晚22:30执行完全备份脚本

30 22 \* \* \* /mydata/scripts/mysqlfullbackup.sh >/dev/null 2>/mydata/mysql/backup/log/fullbackup\_err.log

1. account\_backup\_12.sh

#!/bin/bash

BakDir=/mydata/mysql/backup/mysqldump

LogFile=/mydata/mysql/backup/log/fullbackup.log

####DECLARE variable####

Date=`date +%Y-%m-%d\_%T`

###mysqldump databases&&tables####

mysqldump='/mydata/mysql/install/bin/mysqldump'

#connect\_info='-u rkylinadmin -p'Rkylin\_web!1' -P3316'

connect\_info='-h127.0.0.1 -P3316'

begin\_account=`date +"%Y-%m-%d %H:%M:%S"`

$mysqldump $connect\_info account>$BakDir/databases/account"\_"$Date.sql

if [ $? -eq 0 ]; then

Last\_account=`date +"%Y-%m-%d %H:%M:%S"`

echo "account backup start:$Begin\_account finish:$Last\_account account"\_"$Date.sql succ" >> $LogFile

else

echo "account backup start:$Begin\_account failed!!!" >> $LogFile

exit 1

fi

start\_time=`date -d "$begin\_account" +%s`

finish\_time=`date -d "$Last\_account" +%s`

interval=`expr $finish\_time - $start\_time`

echo $interval

if [ $interval -gt 60 ]; then

sendEmail -f rkylin2015@yeah.net -s smtp.yeah.net -u "account backup time longer than 1 minute;" -xu rkylin2015@yeah.net -xp 'othnzrexmapxtbdl' -t huchenglong@rongcapital.cn luzhidong@rongcapital.cn -m "account backup time longer than 1 minute;please put off the account backup of corntab!"

else

exit 1;

fi

# crontab –e

##每天12:00执行

0 12 \* \* \* /mydata/scripts/account\_backup\_12.sh >/dev/null 2>/mydata/mysql/backup/log/account\_backup\_12\_err.log

1. 脚本xtrabackup.sh

#!/bin/bash

mkdir -p /mydata/mysql/backup/log

mkdir -p /mydata/mysql/backup/xtrabackup

LogFile=/mydata/mysql/backup/log/xtrabackup.log

BakDir=/mydata/mysql/backup/xtrabackup

DumpFile=$BakDir/`date +%F\_%H-%M-%S`

GZDumpFile=$DumpFile.tgz

Begin=`date +"%Y年%m月%d日 %H:%M:%S"`

/usr/bin/innobackupex --defaults-file=/etc/my.cnf --user=rkylinadmin --password='Rkylin\_web!1' $BakDir

if [ $? -eq 0 ]; then

Last=`date +"%Y年%m月%d日 %H:%M:%S"`

echo "开始:$Begin 结束:$Last $GZDumpFile succ" >> $LogFile

/bin/tar czvf $GZDumpFile $DumpFile

/bin/rm -rf $DumpFile

else

echo "开始:$Begin failed!!!" >> $LogFile

exit 1

fi

添加cron

# crontab –e

##每周日22:30执行完全备份脚本

30 22 \* \* 0 /mydata/scripts/xtrabackup.sh >/dev/null 2>/mydata/mysql/backup/log/xtrabackup\_err.log

1. mysqlbackup

# crontab –e

30 22 \* \* 0 /path-to-mysqlbackup/mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/mydata/backup/meb --with-timestamp backup &>/dev/null

## 备份有效性检查

每周在测试机上对数据库进行恢复，然后检查表中数据的完整性。

# mysql < 20150830.sql

mysql> select count(\*) from information\_schema.tables;.

mysql> select count(\*) from information\_schema.columns;

mysql> select count(\*) from information\_schema.routines;

mysql> select count(\*) from information\_schema.events;

mysql> select count(\*) from account.ACCOUNT\_INFO;

mysql> select count(\*) from order.ORDER\_ACCOUNT\_INFO;

mysql> select count(\*) from user.ACCOUNT\_INFO;

# 恢复方案

## 恢复方案

1. 首先我们需要利用前一天的备份文件对数据库进行完全恢复，然后利用二进制文件进行增量恢复。

假设我们要使用的备份文件为all-database\_20151104.sql，二进制如下：

[root@iZ235f6xergZ binlog]# ll

total 33808

-rw-rw---- 1 mysql mysql 5150809 Oct 19 22:00 binlog.000177

-rw-rw---- 1 mysql mysql 4989099 Oct 20 22:00 binlog.000178

-rw-rw---- 1 mysql mysql 4666387 Oct 21 22:00 binlog.000179

-rw-rw---- 1 mysql mysql 4751267 Oct 22 22:00 binlog.000180

-rw-rw---- 1 mysql mysql 4535556 Oct 23 22:00 binlog.000181

-rw-rw---- 1 mysql mysql 2668181 Oct 24 22:00 binlog.000182

-rw-rw---- 1 mysql mysql 4137373 Oct 25 22:30 binlog.000183

备份时，--flush-logs选项会产生新的二进制日志binlog.000183，即备份以后的数据改变都会记录在binlog.000183中，--master-data=2选项会在备份文件中添加如下的注释：

-- Position to start replication or point-in-time recovery from

-- CHANGE MASTER TO MASTER\_LOG\_FILE='binlog.000183',MASTER\_LOG\_POS=4;

使用mysqlbinlog查看二进制文件的内容，找到需要恢复的时间点或位置。

1. xtrabackup的恢复见4.4中1.全备的恢复。

## 恢复脚本

**全库的恢复：**

# mysql< all-database\_20151104.sql

**binlog的恢复：**

# mysqlbinlog --stop-datetime="2015-10-26 11:25:56" binlog.000183 | mysql

* --start-datetime=datetime 从二进制日志中第1个日期时间等于或晚于datetime参量的事件开始读取。
* --stop-datetime=datetime 从二进制日志中第1个日期时间等于或晚于datetime参量的事件起停止读。
* --start-position=N 从二进制日志中第1个位置等于N参量时的事件开始读。
* --stop-position=N 从二进制日志中第1个位置等于和大于N参量时的事件起停止读。

**单库的恢复：**

# mysql account < account\_2015-11-05\_17:52:26.sql

**单表的恢复：**

# mysql account < account\_info.sql

account为指定的库名

## 加快恢复方法

**log-bin=OFF**

恢复时开启二进制日志显然是无意义的，增加了不必要的IO。因此关闭该选项

**sync\_binlog=0**

如果关闭了二进制日志，则这个选项不调整。

但如果因为特殊原因不能关闭二进制日志时，可以考虑减少binlog的fsync来减少磁盘IO压力

**Innodb\_flush\_log\_at\_trx\_commit=0**

同上，由于不存在宕机风险，无需那么卖力的刷写redo log。

**Innodb\_buffer\_pool\_size 尽可能大**

尽可能大的配置Innodb\_buffer\_pool\_size 来保证更多的脏页能够存在于BP中，增大潜在的写入合并的可能性，从而减少了磁盘的IO。

**Innodb\_logfile\_size=1G 或更大**

增大redolog的体积可以推迟blocking checkpoint发生的时间，也一定程度缓解adaptive flush的刷写频率。

调整这个参数对于恢复表体积远大于Innodb Buffer Size时非常有用。

**Innodb\_doublewrite=OFF**

由于不存在宕机风险（即使宕机，也就是重新再恢复一次），所以doublewrite也可以不需要了

**Innodb\_flush\_neighbors=1**

由于逻辑导入更多的是顺序写入，打开flush neighbor以后不单能把IO pattern更贴近顺序。同时，innodb的内部逻辑也会把多个page合并成一次IO进行提交，刷写性能更高。

## 恢复场景模拟

### 全库恢复

1. 模拟场景

假设整个数据库宕机，我们需要使用上次的全库备份加上备份之后产生的binlog对数据库进行全库恢复。为模拟真实场景，使用sysbench对数据进行压测，压测期间对数据库进行全备。备份结束后，在test库中创建表t，并插入数据，用来检查恢复有效性。之后将数据库进程杀掉，模拟宕机。再在另一台机器上进行全库恢复，与第一台进行比对，检查数据是否一致。

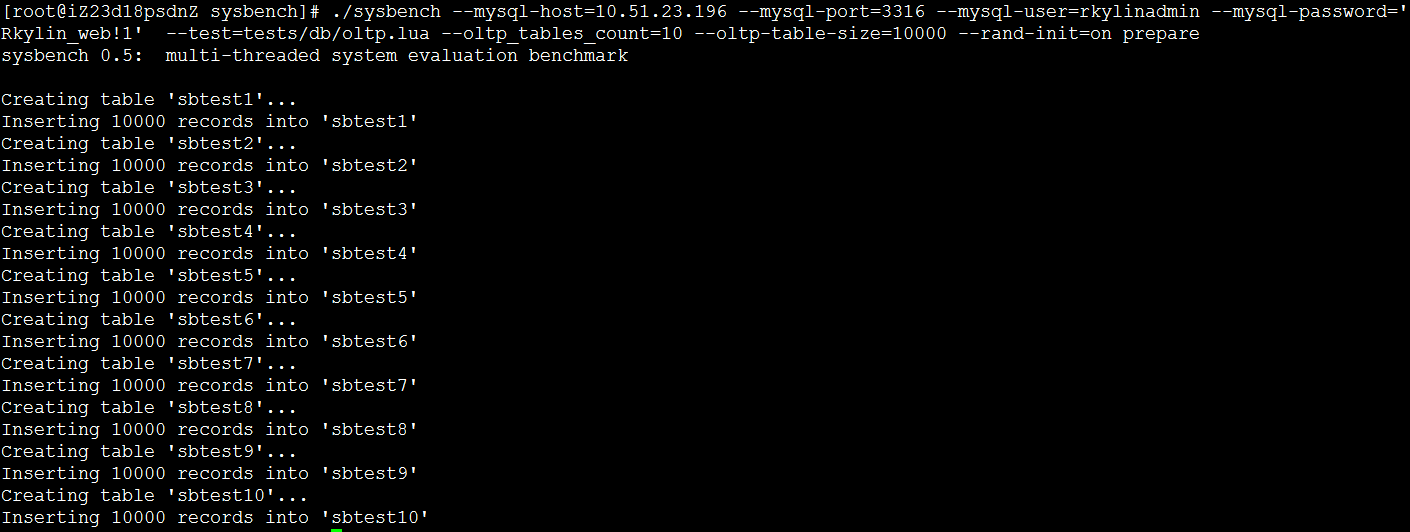
1. 操作步骤

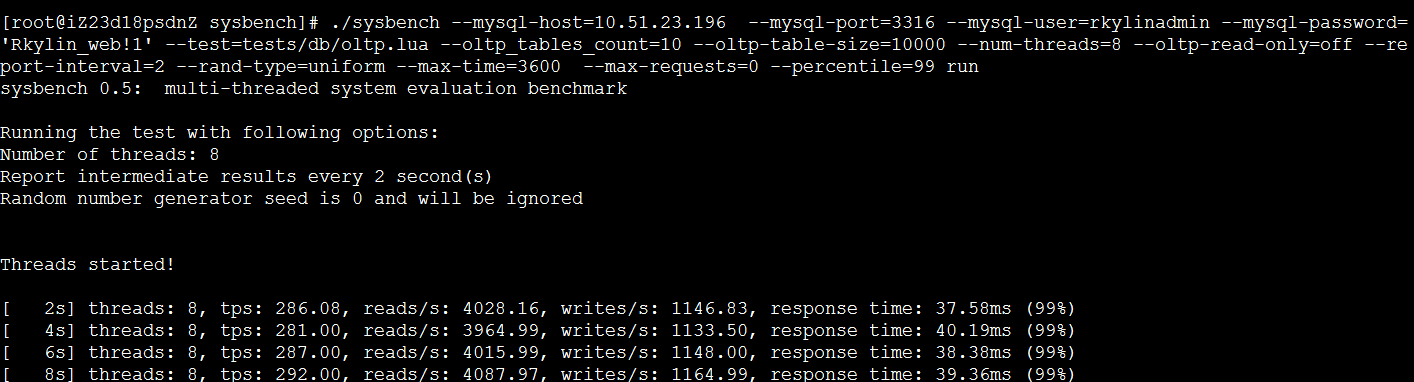
10.51.23.196为主机，用虚拟机来恢复。

创建压测数据库

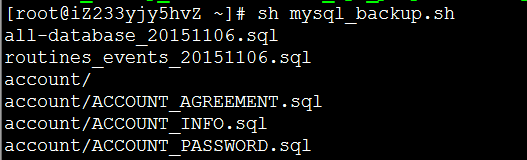
# mysqladmin create sbtest

进行压测





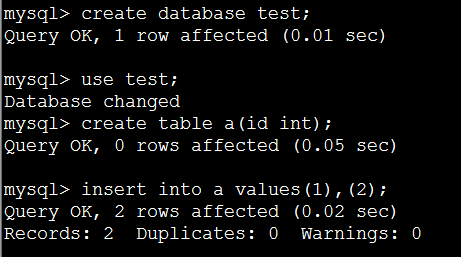
对数据库进行全备份



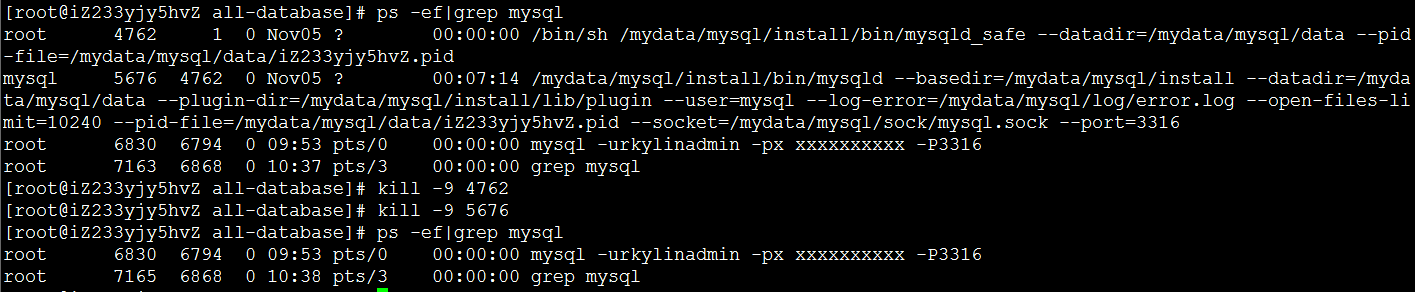
备份文件



在数据库中创建测试库并插入数据

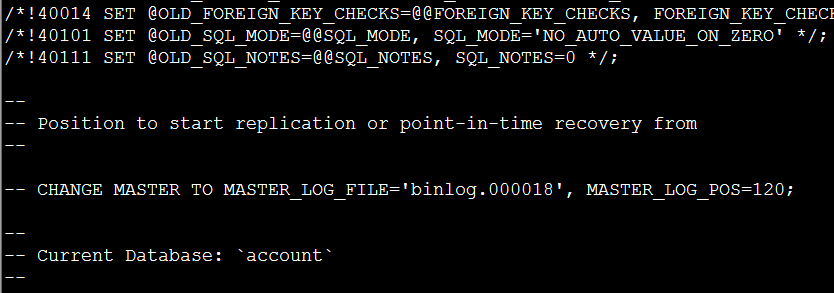


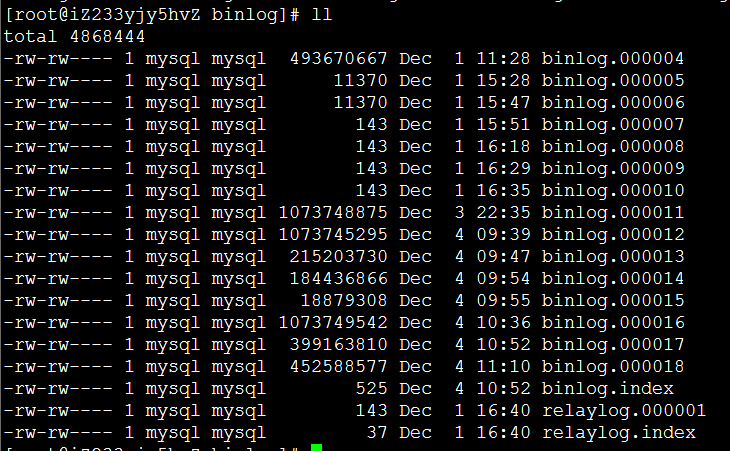
将数据库宕掉



查看binlog位置

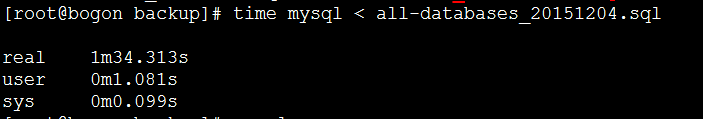
# more all-database\_20151106.sql

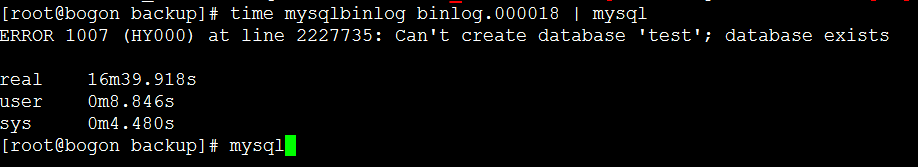




可知，备份后的操作都在binlog.000018中，该日志有400多MB。

在虚拟机上恢复备份

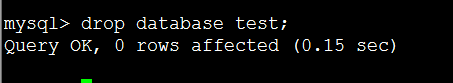


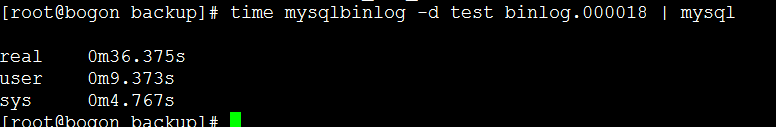


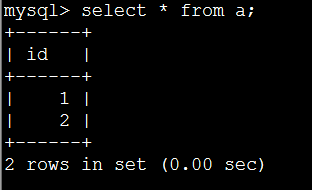
看到恢复二进制日志时报错，应为虚拟机数据库中存在test库，所以执行创建语句失败，不过这也证明了此种恢复方法的有效性。

### 单库恢复

根据上面的案例，并没有对test库恢复成功，因为虚拟机中存在test库，所以先将test库删除，再单独恢复test库







看到test库中的数据和主库一致。

# Percono XtraBackup

## 简介

Xtrabackup为所有版本的Percona Server, MySQL和MariaDB做MySQL热备份，它是一个对InnoDB、XtrDB、HailDB存储引擎做数据备份的工具，此外，它还可以支持以下存储引擎的备份通过短暂的停顿在备份结尾添加：MyISAM、Merge和Archive，包括分区表，触发器和数据库选项。是商业备份工具InnoDB Hotbackup的一个很好的替代品。

优点：

* xtrabackup只能备份InnoDB和XtraDB两种数据表，而不能备份MyISAM数据表
* 备份快速、可靠
* 备份过程中不间断事务的处理
* 节省磁盘空间和网络带宽
* 提高正常运行时间由于更快的恢复时间

特点：

* 创建innodb数据库的热备，不停止数据库
* 增量备份MySQL
* 流压缩MySQL备份到另一台服务器
* 在线在MySQL服务器之间移动表
* 轻松创建MySQL复制从服务器
* 在不增加服务器负载的情况下备份MySQL

Xtrabackup包含四个工具：innobackupex、xtrabackup、xbcrypt、xbstream。

* innobackupex：一个封装的脚本，提供备份所有MySQL数据库实例的MyISAM、InnoDB、XtraDB表的功能。
* xtrabackup：一个编译的C语言二进制文件，只能复制InnoDB和XtraDB数据。
* xbcrypt：加密解密备份文件的工具。
* xbstream：允许将文件转变为xbstream格式或将xbstream格式文件提取出来的工具。
* 仅使用xtrabackup二进制文件是可以的，但是建议的方式是使用innobackupex封装脚本，并让它去为你执行xtrabackup。

## 下载安装

下载地址：http://www.percona.com/downloads/XtraBackup/

rpm文件：percona-xtrabackup-2.2.9-5067.el6.x86\_64.rpm

安装：shell>yum install percona-xtrabackup-2.2.9-5067.el6.x86\_64.rpm

## 备份

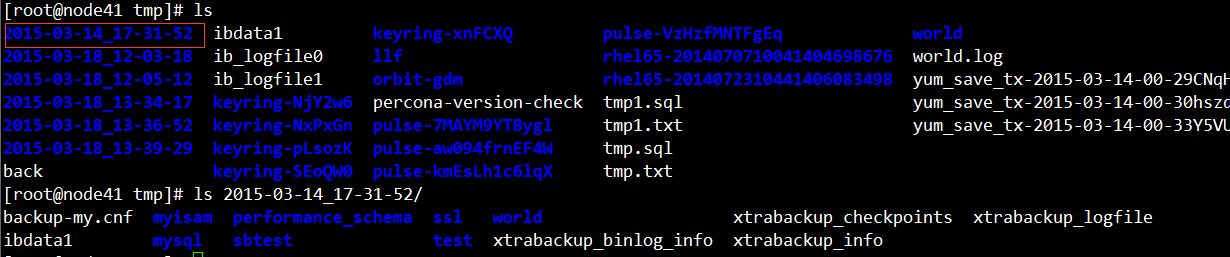
1. 全备份：

shell> innobackupex --user=root --password=oracle /tmp 2>/tmp/world.log

--defaults-file可以指定配置文件位置，但是该选项必须为第一个选项

2>/tmp/world.log将标准输出写入world.log文件

默认备份文件会自动生成对应的时间目录，如下：



如果不想生成日期目录，可以添加--no-timestamp选项，使备份数据生成在指定的目录下。

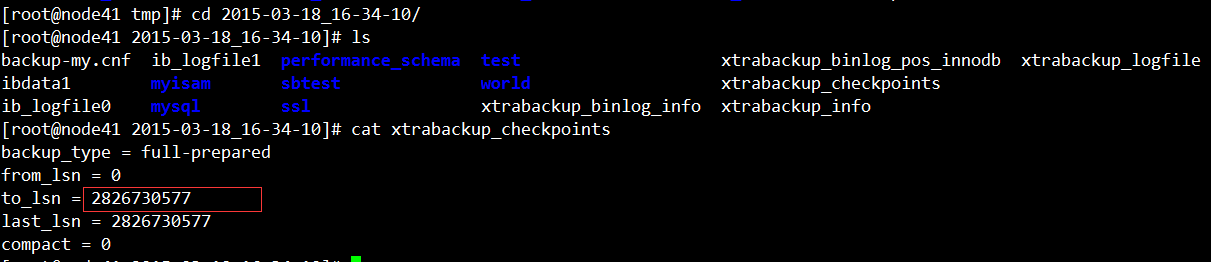
1. 增量备份：

首先我们需要有一份全备

shell> innobackupex --user=root --password=oracle /tmp

假设备份目录为/tmp/2015-03-14\_17-31-52

可以在/tmp/2015-03-14\_17-31-52/xtrabackup\_checkpoints里查看到lsn号，如图



创建增量备份

shell> innobackupex --user=root --password=oracle --incremental --incremental-basedir=/tmp/2015-03-14\_17-31-52 /tmp

其中--incremental-basedir=/tmp/2015-03-14\_17-31-52也可以用--incremental-lsn=2826730577替换

这样就在/tmp下创建了增量备份，新的目录为/tmp/2015-03-14\_17-35-22

继续创建新的增量备份时，--incremental-basedir将指定为/tmp/2015-03-14\_17-35-22，以此类推，下一个增量备份的--incremental-basedir设定为上次备份的目录。

1. 局部备份：

局部备份需要添加--databases选项，格式为--databases="databasename1[.tablename] databasename2[.tablename]"，例如：

shell> innobackupex --user=root --password=oracle --databases=“test world.City world.Country sbtest” /tmp

则会在日期目录文件中生成test数据库、world数据库中的City表和Country表、sbtest数据库的备份。

也可以使用选项--tables-file=/tmp/tables.txt，首先在tables.txt文件中写入要备份的表，格式必须为databasename.tablename，一行对应一张表。然后命令如下：

shell> innobackupex --user=root --password=oracle --tables-file=/tmp/talbes.txt /tmp

就会在日期目录下生成各个表的备份。

1. 压缩备份：

* 将备份文件压缩成xbstream格式

shell> innobackupex --user=root --password=oracle --stream=xbstream /tmp > back.xbstream

这会在/tmp目录下产生压缩后的备份文件back.xbstream，解压方式如下

shell> xbstream -x < back.xbstream

* 将备份文件进行compact压缩

shell> innobackupex --user=root --password=oracle --compact /tmp

* 压缩为.tar.gz

shell> innobackupex --defaults-file=/etc/my.cnf --user=root --password=oracle --stream=tar /home/tank/backup | gzip > /home/tank/backup/`date +%F\_%H-%M-%S`.tar.gz

## 恢复

1. 全备的恢复：

在转储数据之前需要先执行准备阶段，生成对应的日志文件，从而保证数据的一致性

shell> innobackupex --apply-log /tmp/2015-03-14\_17-31-52

添加--use-memory选项可以设定使用的内存，默认为100MB，例如：

shell> innobackupex --apply-log --use-memory=1G /tmp/2015-03-18\_13-39-29

接下来就是转储阶段，必须先关闭数据库服务

shell> service mysql stop

然后转储上步以准备好的数据

shell> innobackupex --copy-back /tmp/2015-03-14\_17-31-52

注意：转储位置为配置文件my.cnf中指定的datadir，在转储前必须保证该目录为空，或者添加--force-non-empty-directories选项。

转储完成后需要修改文件的所属用户

shell> chown mysql:mysql /var/lib/mysql

最后启动数据库

shell> service mysql start

1. 增量备份的恢复：

首先也要对备份进行准备阶段

假设全备份目录为/tmp/2015-03-14\_17-31-52，有以下依次的增量备份/tmp/2015-03-15\_17-35-22，/tmp/2015-03-16\_17-35-22，/tmp/2015-03-17\_17-35-22。

先准备全备份，但要添加--redo-only选项

shell> innobackupex --apply-log --redo-only /tmp/2015-03-14\_17-31-52

这里--redo-only作用为只重做日志中已提交的事务，不回滚日志中未提交的事务，会看到如下提示信息

120103 22:00:12 InnoDB: Shutdown completed;

log sequence number 1291135 120103 22:00:12 innobackupex: completed OK!

然后将第一份增量备份添加到全备份中

shell> innobackupex --apply-log --redo-only /tmp/2015-03-14\_17-31-52 --incremental-dir=/tmp/2015-03-15\_17-35-22

会看到如下提示信息，但是lsn号已经相应的改变了

120103 22:08:43 InnoDB: Shutdown completed;

log sequence number 1358967 120103 22:08:43 innobackupex: completed OK!

接下来添加第二份增量备份

shell> innobackupex --apply-log --redo-only /tmp/2015-03-14\_17-31-52 --incremental-dir=/tmp/2015-03-16\_17-35-22

最后添加第三份增量备份

shell> innobackupex --apply-log /tmp/2015-03-14\_17-31-52 --incremental-dir=/tmp/2015-03-17\_17-35-22

注意：这里因为第三份为最后一份增量备份，所以不需要添加--redo-only选项，此时已经可以回滚日志中未提交的事务了。

再完成了所有增量备份的合并之后，就可以回滚主全备份的未提交事务了

shell> innobackupex --apply-log /tmp/2015-03-14\_17-31-52

接下来就是转储备份，和转储全备份是一样的

shell> innobackupex --copy-back /tmp/2015-03-14\_17-31-52

1. 局部备份的恢复：

注意： 局部备份的恢复不可以使用--copy-back方式，否则会导致恢复的数据不一致，而是应该使用导入表的方式恢复数据。

准备阶段

shell> innobackupex --apply-log --export /tmp/2015-03-19\_10-19-31

完成后会在目录下生成.cfg文件和.exp文件，.cfg文件用于InnoDB，.exp文件用于XtraDB。



恢复阶段

首先要将数据库中原先的表空间文件扔掉

mysql> alter table test.tmp1 discard tablespace ;

然后将备份目录中的tmp1.cfg、tmp1.ibd复制到数据目录/var/lib/mysql/test下，接着导入表空间

mysql> alter table test.tmp1 import tablespace ;

1. compact备份的恢复：

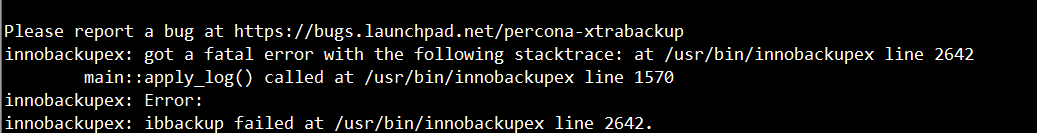
准备阶段需要重建二级索引

shell> innobackupex --apply-log --rebuild-indexes /tmp/2015-03-14\_17-31-52

转储备份

shell> innobackupex --apply-log /tmp/2015-03-14\_17-31-52

注意：测试时报错



## innobackup 常用参数说明

--defaults-file

同xtrabackup的--defaults-file参数

--apply-log

对xtrabackup的--prepare参数的封装

--copy-back

做数据恢复时将备份数据文件拷贝到MySQL服务器的datadir ；

--remote-host=HOSTNAME

通过ssh将备份数据存储到进程服务器上；

--stream=[tar]

备 份文件输出格式, tar时使用tar4ibd , 该文件可在XtarBackup binary文件中获得.如果备份时有指定--stream=tar, 则tar4ibd文件所处目录一定要在$PATH中(因为使用的是tar4ibd去压缩, 在XtraBackup的binary包中可获得该文件)。

在 使用参数stream=tar备份的时候，你的xtrabackup\_logfile可能会临时放在/tmp目录下，如果你备份的时候并发写入较大的话 xtrabackup\_logfile可能会很大(5G+)，很可能会撑满你的/tmp目录，可以通过参数--tmpdir指定目录来解决这个问题。

--tmpdir=DIRECTORY

当有指定--remote-host or --stream时, 事务日志临时存储的目录, 默认采用MySQL配置文件中所指定的临时目录tmpdir

--redo-only --apply-log组,

强制备份日志时只redo ,跳过rollback。这在做增量备份时非常必要。

--use-memory=#

该参数在prepare的时候使用，控制prepare时innodb实例使用的内存量

--throttle=IOS

同xtrabackup的--throttle参数

--sleep=是给ibbackup使用的，指定每备份1M数据，过程停止拷贝多少毫秒，也是为了在备份时尽量减小对正常业务的影响，具体可以查看ibbackup的手册 ；

--compress[=LEVEL]

对备份数据迚行压缩，仅支持ibbackup，xtrabackup还没有实现；

--include=REGEXP

对 xtrabackup参数--tables的封装，也支持ibbackup。备份包含的库表，例如：--include="test.\*"，意思是要备份 test库中所有的表。如果需要全备份，则省略这个参数；如果需要备份test库下的2个表：test1和test2,则写 成：--include="test.test1|test.test2"。也可以使用通配符，如：--include="test.test\*"。

--databases=LIST

列出需要备份的databases，如果没有指定该参数，所有包含MyISAM和InnoDB表的database都会被备份；

--uncompress

解压备份的数据文件，支持ibbackup，xtrabackup还没有实现该功能；

--slave-info,

备 份从库, 加上--slave-info备份目录下会多生成一个xtrabackup\_slave\_info 文件, 这里会保存主日志文件以及偏移, 文件内容类似于:CHANGE MASTER TO MASTER\_LOG\_FILE='', MASTER\_LOG\_POS=0

--socket=SOCKET

指定mysql.sock所在位置，以便备份进程登录mysql.

# mysqlbackup

## 下载安装

从mysql官网下载的文件名为V76894-01.zip

解压即可

unzip V76894-01.zip

tar zxvf meb-3.12.1-linux-glibc2.5-x86-64bit.tar.gz

将打包文件里的mysqlbackup拷贝到$PATH即可

cp mysqlbackup /usr/bin

chmod +x /usr/bin/mysqlbackup

## 使用前准备

在配置文件中添加：

[mysqlbackup]

backup-dir=/backups/meb

user=mysqlbackup

socket=/mydata/mysql/sock/mysql.sock

password=oracle

创建备份用户，当然也可以直接使用root用户，以下是备份用户需要的权限：

GRANT RELOAD ON \*.\* TO 'mysqlbackup'@'localhost' identified by ‘oracle’;

GRANT CREATE, INSERT, DROP, UPDATE ON mysql.backup\_progress TO 'mysqlbackup'@'localhost';

GRANT CREATE, INSERT, SELECT, DROP, UPDATE ON mysql.backup\_history TO 'mysqlbackup'@'localhost';

GRANT REPLICATION CLIENT ON \*.\* TO 'mysqlbackup'@'localhost';

GRANT SUPER ON \*.\* TO 'mysqlbackup'@'localhost';

GRANT LOCK TABLES, SELECT, CREATE, ALTER ON \*.\* TO 'mysqlbackup'@'localhost';

## 备份

1. 全备份

**备份到单个文件：**

mysqlbackup --defaults-file=/etc/my.cnf --backup-image=/backups/my.mbi --backup-dir=/backups/backup-tmp backup-to-image

--backup-image指定备份文件名

--backup-dir指定备份中产生的其他文件目录

backup-to-image表明备份到单独的image文件

**备份到目录：**

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb --with-timestamp backup

--with-timestamp将在备份目录下生成时间目录

1. 增量备份或差异备份

差异备份指每次做增量备份的时候都以全备份为基准，备份全备份到现在的改变；增量备份指每次做增量备份的时候以前一次增量备份为基准，备份上次到现在的改变。

mysqlbackup --defaults-file=/etc/my.cnf --incremental --incremental-base=dir:/backups/meb/2015-11-03\_10-34-02 --incremental-backup-dir=/backups/incre-backup --with-timestamp backup

--incremental-base指定上次的全备或增量备份目录

--incremental-backup-dir指定新增量备份的目录

1. 创建压缩的备份

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb --with-timestamp --compress --compress-level=5 backup

## 准备

在恢复之前需要对raw backup（原始备份）进行apply-log，使raw backup变为prepared backup，如下：

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb/2015-11-03\_10-34-02 apply-log

对于简单的备份（非压缩备份和非增量备份），可以将初始备份和应用日志结合，使用backup-and-apply-log命令，如下：

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb backup-and-apply-log

也可以使用copy-back-and-apply-log命令将apply-log和copy-back结合。

**对压缩后的备份应用日志：**

mysqlbackup --backup-dir=/export/backups/compressed --uncompress apply-log

**应用增量备份到全备上：**

mysqlbackup --backup-dir=/backups/meb/2015-11-03\_10-34-02 apply-log

mysqlbackup --backup-dir=/backups/meb/2015-11-03\_10-34-02 --incremental-backup-dir=/backups/incre-backup/2015-11-03\_16-20-31 apply-incremental-backup

## 恢复

**关闭并转储数据库(prepared backup)：**

mysqladmin --user=root --password shutdown

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb/2015-11-03\_10-34-02 copy-back

**使用copy-back-and-apply-log转储备份目录(raw backup)：**

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb/2015-11-03\_10-34-02 copy-back-and-apply-log

**转储单独文件备份：**

mysqlbackup --defaults-file=/etc/my.cnf --datadir=/mydata/mysql/data --backup-image=/backups/my.mbi --backup-dir=/backups/backup-tmp copy-back-and-apply-log

--defaults-file指定备份配置文件

--datadir指定要恢复到的数据目录(若在my.cnf已指定，这里可不添加该选项)

--backup-image指定备份文件

--backup-dir指定其他文件位置

**转储增量备份：**

分为两种方式，1）先恢复全备份，在分别恢复每次增量备份；2）先将增量备份应用到全备份，使全备份变为最新的，再恢复全备份。

1）

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb/2015-11-03\_10-34-02 copy-back-and-apply-log

mysqlbackup --defaults-file=/etc/my.cnf --incremental --incremental-backup-dir=/backups/incre-backup/2015-11-03\_16-20-31 copy-back-and-apply-log

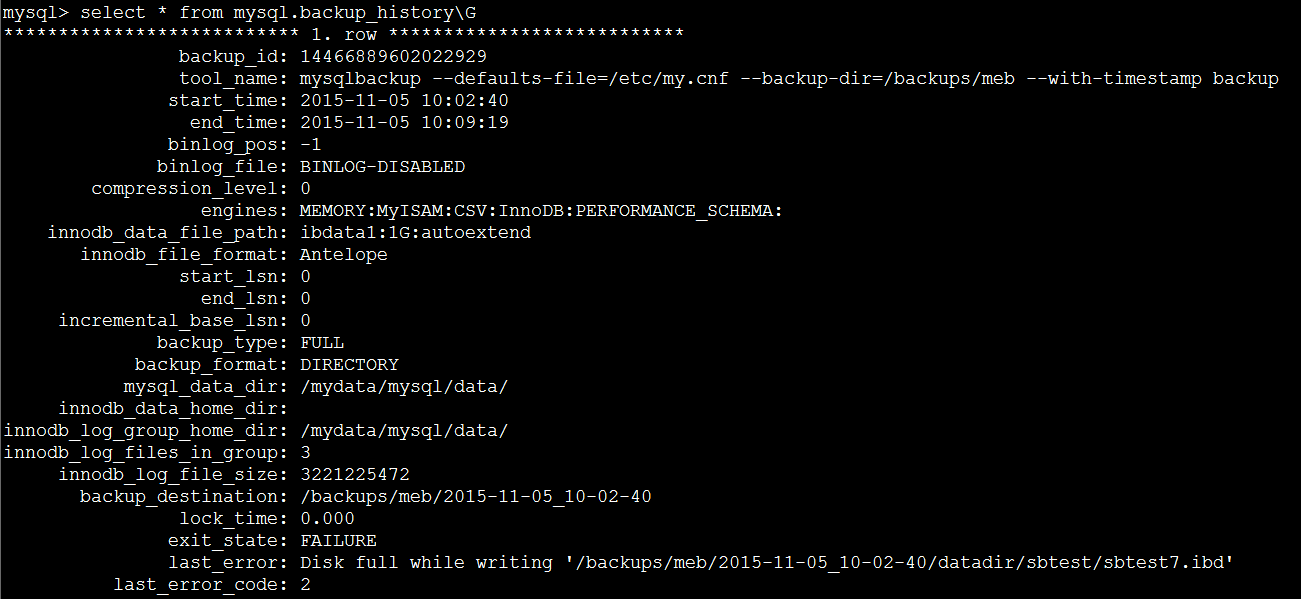
2)

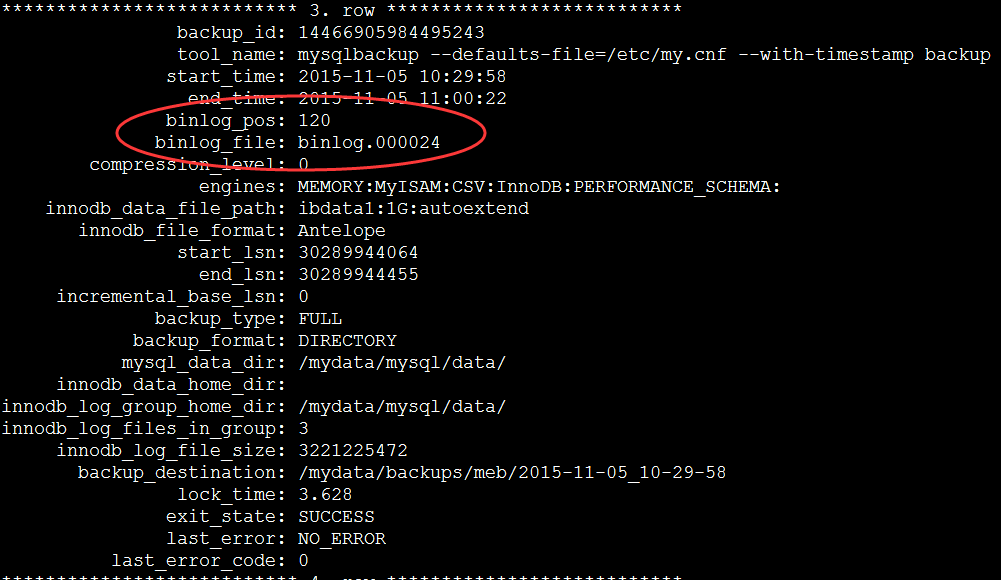
mysqlbackup --backup-dir=/backups/meb/2015-11-03\_10-34-02 apply-log

mysqlbackup --backup-dir=/backups/meb/2015-11-03\_10-34-02 --incremental-backup-dir=/backups/incre-backup/2015-11-03\_16-20-31 apply-incremental-backup

mysqlbackup --defaults-file=/etc/my.cnf --backup-dir=/backups/meb/2015-11-03\_10-34-02 copy-back

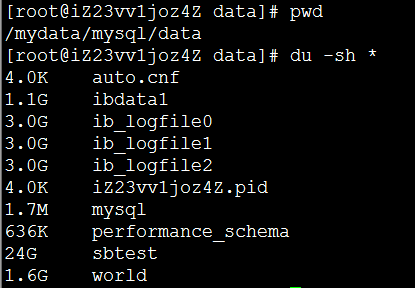
## 查看备份后的binlog位置





# mysqldump与mysqlbackup对比

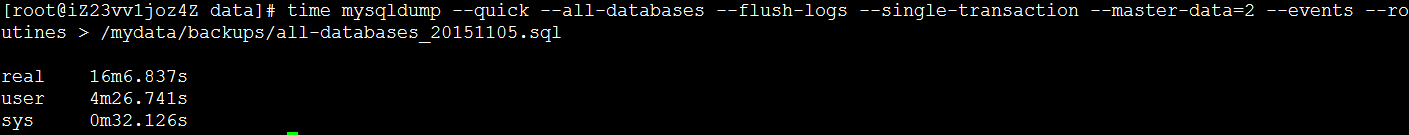
数据库数据如下：



数据目录总大小为35.6G

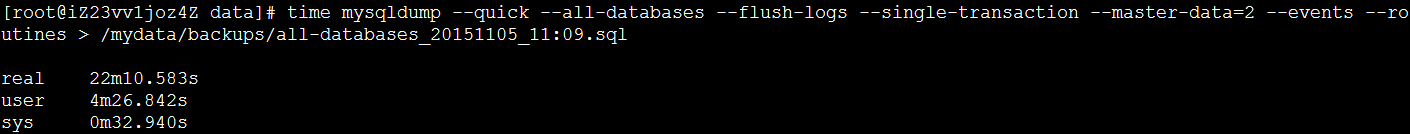
## 使用mysqldump执行全备份：

**第一次**



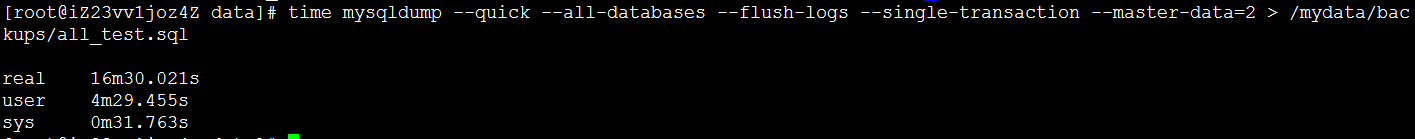
用时16分6.837秒

**第二次**



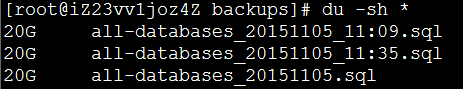
用时22分10秒，分析是从库在做复制任务，延长了时间。

**第三次**



用时16分30秒

**备份文件**



## 使用mysqlbackup执行全备份：

先在配置文件my.cnf中添加

[mysqlbackup]

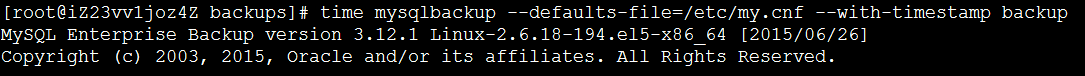
backup-dir=/mydata/backups/meb

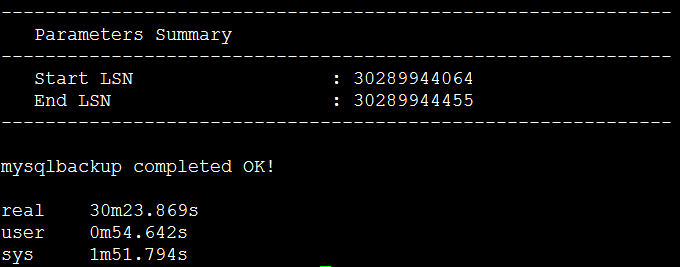
user=rkylinadmin

socket=/mydata/mysql/sock/mysql.sock

password=Rkylin\_web!1

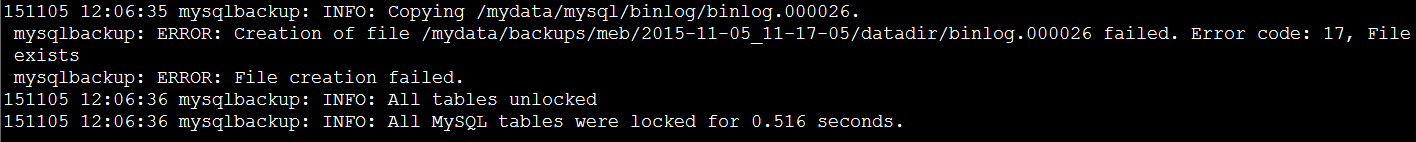
**第一次备份**





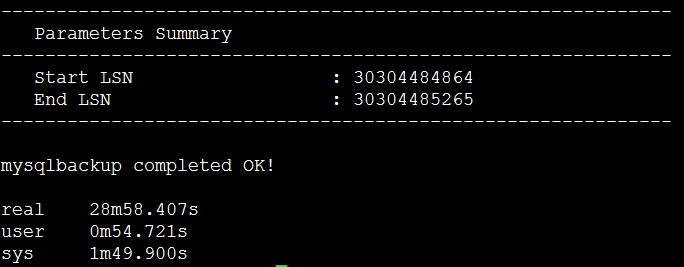
备份时间为30分23.869秒

**第二次备份**



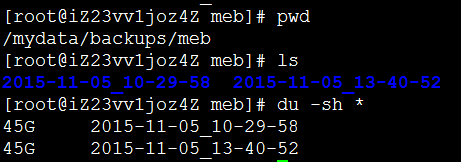
备份失败了…，分析原因可能是此时从库在做复制，导致失败了

**第三次备份**

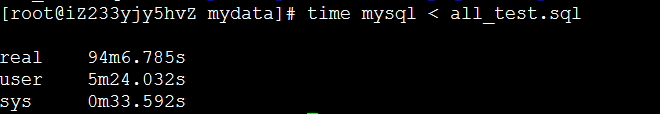


28分58.4秒

**备份文件为**

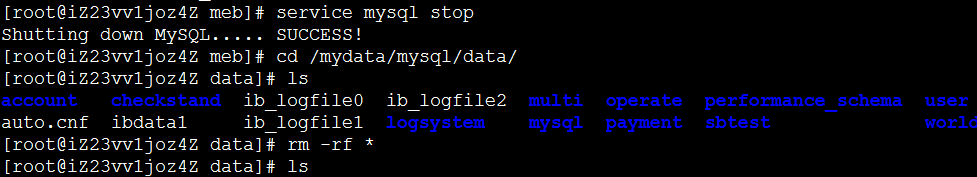


## 使用mysqldump恢复

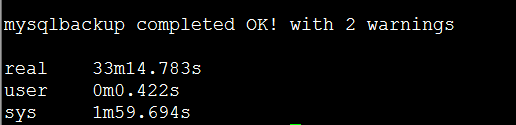


用时94分6秒

## 使用mysqlbackup恢复

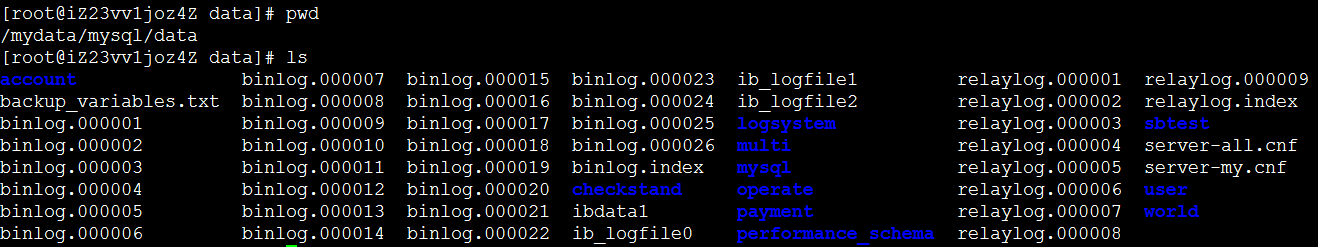


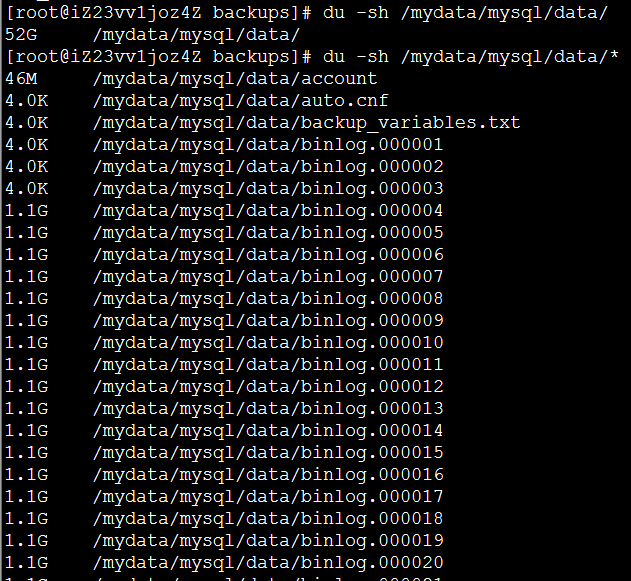




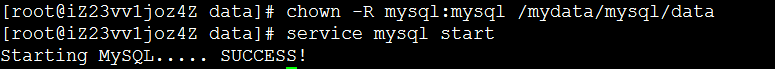
用时33分14秒

备份文件全部拷贝到数据目录，但是所属变为了root，需要改为mysql，binlog也备份了





chown -R mysql:mysql /mydata/mysql/data



## 总结

对于25.6G的数据，35.6G的数据目录，其中ibdata1文件1G，ib\_logfile\*文件 9G

|  |  |  |
| --- | --- | --- |
| 工具 | 备份时间 | 恢复时间 |
| Mysqldump | 16分20秒 | 94分6秒 |
| Mysqlbackup | 29分30秒 | 33分14秒 |

可见，mysqldump备份时间优于mysqlbackup，但恢复时间远大于mysqlbackup，如果数据量更大的话，那么mysqldump的恢复时间是难以忍受的。mysqlbackup的备份时间和恢复时间差不多，因为是物理方式，使用mysqlbackup备份期间出了一次错误，有必要对备份定期检查。