

MySQL Backup and Recovery

— MySQL Backup

1.功能

mysqldump全量和增量备份，通过最近一次备份刷新产生binlog来定位执行增量。

- 脚本下载地址
 - [github](#)
- 场景一：
每天执行一次全量备份。
- 场景二：
每周日执行一次全量备份，然后每天3点执行增量备份。
- 应用场景：
 - 增量备份在周一到周六凌晨3点，会使用mysqlbinlog 导出sql并使用gzip压缩到指定目录
 - `mysqlbinlog -vv binlog.000044 binlog.000045 binlog.000046 > |gzip > $INCR_BACKUP_DIR/incr.sql.gz`
 - 全量备份则使用mysqldump将所有的数据库导出，每周日凌晨3点执行，并会删除N天之前的目录和文件。参数如下：
 - `MYSQLDUMP_OPTION='--single-transaction --master-data=2 -E -R --flush-logs --databases'`
 - 删除命令
 - `(find $BASE_DIR -mtime + $DELETE_DAYS -type d -name "full*" -exec rm -rf {} \;)`

2.使用方法

- 脚本需修改参数：

```
MY_USER "gcdb"           --备份帐号
MY_PASSWORD="iforgot"     --备份密码
MY_IP="192.168.49.247"    --本机ip，例如从库ip
MY_MASTER_USER="gcdb"     --master帐号
MY_MASTER_PASSWORD="iforgot" --master密码
MY_MASTER_IP="192.168.49.246" --指主库ip
BINLOG_FILE=/r2/mysqldata --binlog文件所在的目录，增量时需要用到
BASE_DIR=/mybak           --备份基础目录
DELETE_DAYS=15            --备份保存天数，即删除N天之前的备份，例如一周一个全备、每一天一个增量，该值必须大于配置为7，
FILTER="information_schema|test|mysql|sys|performance_schema" --过滤指定数据库，也就是不备份的数据库（注意mysql库）
```

- 备份基础目录以/mybak为例，目录的树形结构如下：

```
[root@node02 scripts]# tree /mybak/
/mybak/
├── full
│   ├── full_20180419
│   │   ├── backup.log
│   │   ├── dbname
│   │   ├── fullbak.sql.gz
│   │   ├── grants.sql
│   │   ├── master_grants.sql
│   │   ├── master_users.sql
│   │   ├── position
│   │   └── users.sql
│   └── full_20180420
│       ├── backup_full.log
│       ├── dbname
│       ├── fullbak.sql.gz
│       ├── grants.sql
│       ├── master_grants.sql
│       ├── master_users.sql
│       ├── position
│       └── users.sql
├── incr
│   ├── incr_20180420130001
│   │   ├── backup_incr.log
│   │   ├── incr.sql.gz
│   │   ├── new_binlogs_list
│   │   ├── old_binlogs_list
│   │   └── tmp_binlog_name
│   └── incr_20180420140001
│       ├── backup_incr.log
│       ├── incr.sql.gz
│       ├── new_binlogs_list
│       ├── old_binlogs_list
│       └── tmp_binlog_name
├── public_backup.log    --记录备份是否成功
└── public_position     --保存最新binlog文件名
```

2.1 全备

- 备份命令
 - `./bak_mysql.sh full`
- 计划任务
 - `crontab -e`
 - 每天做一次全备，凌晨3点进行全量备份，备份频率可根据项目情况自行调整。
 - `0 3 * * * /bin/sh /scripts/bak_mysql.sh full >/dev/null 2>&1`

2.2 增量

- 备份命令
 - `./bak_mysql.sh incr`

- 计划任务
 - crontab -e
 - 每个小时(除3点外)进行binglog增量备份,备份频率可根据项目情况自行调整。
 - 0 0-2,4-23 * * * /bin/sh /scripts/bak_mysql.sh incr >/dev/null 2>&1
- 使用参考如下：

```
+-----+
|Usage : ./bak_mysql.sh  (full|incr|oemu)                                |
+-----+
|全备          : ./bak_mysql.sh full                                  |
|增量          : ./bak_mysql.sh incr                                  |
|只导出master权限 : ./bak_mysql.sh oemu                              |
+-----+
计划任务参考
+-----+
|全备   : 30 0 * * * /bin/sh /scripts/bak_mysql.sh full >/dev/null 2>&1      |
|增量   : 30 2-23/2 * * * /bin/sh /scripts/bak_mysql.sh incr >/dev/null 2>&1 |
+-----+
[root@node02 scripts]#
```

3.执行备份

3.1 全备执行过程

```
[root@node01 scripts]# sh bak_mysql.sh full 2>/dev/null
```

```
+-----+
| Backup_Host |
+-----+
| node01.mysql.com |
+-----+
Backup_Host 连接正常
+-----+
| MY_Host |
+-----+
| node01.mysql.com |
+-----+
192.168.49.245开始导出帐号和权限信息
192.168.49.245成功导出 10 个用户权限
192.168.49.245成功导出 10 个用户帐号
1、20180425 16:51:25 开始备份.....
2、备份以下数据库：
mysql percona
3、20180425 16:51:26 备份成功.....
4、备份用时：1 秒
5、备份数据量大小：6.9M
6、记录最新的binlog文件名！
+-----+
| Master_Host |
+-----+
```

```
| slave7 |
+-----+
master 192.168.101.137开始导出帐号和权限信息
master 192.168.101.137成功导出 9 个用户权限
master 192.168.101.137成功导出 9 个用户帐号
全备成功
[root@node01 scripts]#
```

3.2 全备执行结果

```
[root@node01 scripts]# cat /mybak/public_position
binlog.000023

[root@node01 scripts]# cat /mybak/public_backup.log
Backup_Host 连接正常
全备成功
删除 /mybak/full 目录下 7 天之前的备份!
full_backup_ok

[root@node01 scripts]# tree /mybak/full/full_20180425/
/mybak/full/full_20180425/
├─ backup_full.log
├─ dbname          --备份的库名
├─ fullbak.sql.gz   --备份文件
├─ grants.sql       --本机授权文件(mysql5.7之后权限和帐号分开)
├─ master_grants.sql --master授权文件
├─ master_users.sql --master帐号文件
├─ position         --GTID和binlog文件名信息
└─ users.sql        --本机授权文件(mysql5.7之后权限和帐号分开)

[root@node02 scripts]# cat /mybak/full/full_20180420/position
-- GTID state at the beginning of the backup
SET @@GLOBAL.GTID_PURGED='7debec7f-4797-11e8-9274-0050569d16ce:1-3,
-- CHANGE MASTER TO MASTER_LOG_FILE='binlog.000023', MASTER_LOG_POS=234;

[root@node01 scripts]# cat /mybak/full/full_20180425/position
-- GTID state at the beginning of the backup
SET @@GLOBAL.GTID_PURGED='7debec7f-4797-11e8-9274-0050569d16ce:1-3,
-- CHANGE MASTER TO MASTER_LOG_FILE='binlog.000023', MASTER_LOG_POS=234;

[root@node01 scripts]# cat /mybak/full/full_20180425/backup_full.log
1、20180425 16:51:25 开始备份.....
2、备份以下数据库：
mysql percona
3、20180425 16:51:26 备份成功.....
4、备份用时：1 秒
5、备份数据量大小：6.9M
6、记录最新的binlog文件名！
binlog.000023
master 192.168.101.137开始导出帐号和权限信息
master 192.168.101.137成功导出 9 个用户权限
```

```
master 192.168.101.137成功导出 9 个用户帐号
[root@node01 scripts]#
```

3.3 增备执行结果

- 执行增量备份之前进行如下操作：

```
[2018-04-20 15:32:17.838][192.168.49.247-node02][000220][MYSQL]
UPDATE `ttt`.`t1` SET `name` = 'rrrrrssss' WHERE `id` = 3
Time: 0.001s

[2018-04-20 17:10:02.925][192.168.49.246-mycat][016413][MYSQL]
UPDATE `ttt`.`t1` SET `name` = 'xiaowen' WHERE `id` = 3
Time: 0.002s

[2018-04-20 17:11:42.657][192.168.49.246-mycat][016413][MYSQL]
insert into t1 values(8,'xiaomi')
Time: 0.001s
```

执行中

```
[root@node02 scripts]# ./bak_mysql.sh incr 2>/dev/null
+-----+
| Backup_Host      |
+-----+
| node02.mysql.com |
+-----+
mysql连接正常
创建INCR_BACKUP_DIR目录
/mybak/incr/incr_20180420171334
创建/mybak/incr/incr_20180420171334/backup_incr.log
000051 : PUBLIC_POSITION 有获取到数值
循环写入binlog名执行成功
mysqlbinlog 执行成功.....
写入最新的binlog名到公共文件中
增量备份成功

删除 /mybak/incr 目录下 7 天之前的备份！
```

3.4 增备执行结果

```
[root@node02 scripts]# ll /mybak/incr/incr_20180420171334
total 20
-rw-r--r-- 1 root root 2470 Apr 20 17:13 backup_incr.log
-rw-r--r-- 1 root root 1488 Apr 20 17:13 incr.sql.gz
-rw-r--r-- 1 root root 728 Apr 20 17:13 new_binlogs_list
-rw-r--r-- 1 root root 714 Apr 20 17:13 old_binlogs_list
-rw-r--r-- 1 root root 14 Apr 20 17:13 tmp_binlog_name

[root@node02 scripts]# cat /mybak/incr/incr_20180420171334/backup_incr.log
```

```

不需要备份, 后缀为 000001 binlog文件
不需要备份, 后缀为 000002 binlog文件
-----省略-----
不需要备份, 后缀为 000049 binlog文件
不需要备份, 后缀为 000050 binlog文件  --全量备份到binlog.000050, flush logs生成了binlog.000051
需备份后缀为 000051 binlog文件
mysqlbinlog 执行成功.....

[root@node02 scripts]# cat /mybak/public_backup.log
Backup_Host 连接正常
创建 /mybak/incr/incr_20180420171334/backup_incr.log
incr_backup_ok
增量备份成功
删除 /mybak/incr 目录下 7 天之前的备份!

[root@node02 scripts]# cat /mybak/incr/incr_20180420153720/tmp_binlog_name
binlog.000051

[root@node02 scripts]# gunzip < /mybak/incr/incr_20180420171334/incr.sql.gz |more
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!50003 SET @OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#180420 17:02:35 server id 49247 end_log_pos 123 CRC32 0xdf164269 Start: binlog v 4, server v
5.7.18-log created 180420
17:02:35
BINLOG '
q6zZWg9fwAAAdwAAAHsAAAAAAQANS43LjE4LWxvZwAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAEzgNAAGAEgAEBAQEgAAXwAEGggAAAAICAgCAAAACgoKKioAEjQA
AWlCFt8=
'/*!*/;
# at 123
#180420 17:02:35 server id 49247 end_log_pos 234 CRC32 0x3cf6d5c4 Previous-GTIDs
# 8a5dd931-42cc-11e8-aa39-0050569dc4ab:1-4,
# fda7506d-33ea-11e8-b187-000c298b03f2:28759-28763
# at 234
#180420 17:09:48 server id 49246 end_log_pos 299 CRC32 0xe827d4c9 GTID last_committed=0
sequence_number=1
SET @@SESSION.GTID_NEXT= 'fda7506d-33ea-11e8-b187-000c298b03f2:28764'/*!*/;
# at 299
#180420 17:09:48 server id 49246 end_log_pos 362 CRC32 0x52db9f29 Query thread_id=16413
exec_time=4294967295 error_
code=0
SET TIMESTAMP=1524215388/*!*/;
SET @@session.pseudo_thread_id=16413/*!*/;
SET @@session.foreign_key_checks=1, @@session.sql_auto_is_null=0, @@session.unique_checks=1,
@@session.autocommit=1/*!*/;
SET @@session.sql_mode=524288/*!*/;
SET @@session.auto_increment_increment=1, @@session.auto_increment_offset=1/*!*/;
/*!\C utf8mb4 *//*!*/;

SET

```

```

@@session.character_set_client=45,@@session.collation_connection=45,@@session.collation_server=1
92/*!*/;
SET @@session.lc_time_names=0/*!*/;
SET @@session.collation_database=DEFAULT/*!*/;
BEGIN
/*!*/;
# at 362
#180420 17:09:48 server id 49246 end_log_pos 409 CRC32 0x330344f1 Table_map: `ttt`.`t1` mapped
to number 100172
# at 409
#180420 17:09:48 server id 49246 end_log_pos 473 CRC32 0x7a37b361 Update_rows: table id 100172
flags: STMT_END_F

BINLOG '
XK7ZWWhNewAAALwAAAjKBAAAAAEyHAQAAAAEAA3R0dAACdDEAAgMPAjjwAAvFEAzM=
XK7ZWWh9ewAAAQAAAAANKBAAAAAEyHAQAAAAEAAgAC//8AwAAAlycnJycnNzc3P8AwAAAAAd4aWFv
d2VuYbM3eg==
'/*!*/;
### UPDATE `ttt`.`t1`
### WHERE
###   @1=3 /* INT meta=0 nullable=0 is_null=0 */
###   @2='rrrrrrssss' /* VARSTRING(60) meta=60 nullable=1 is_null=0 */
### SET
###   @1=3 /* INT meta=0 nullable=0 is_null=0 */
###   @2='xiaowen' /* VARSTRING(60) meta=60 nullable=1 is_null=0 */
# at 473
#180420 17:09:48 server id 49246 end_log_pos 504 CRC32 0xb3b8f41b5 Xid = 397443
COMMIT/*!*/;
# at 504
#180420 17:09:48 server id 49246 end_log_pos 569 CRC32 0xc05fd7e0 GTID last_committed=1
sequence_number=2
SET @@SESSION.GTID_NEXT= 'fda7506d-33ea-11e8-b187-000c298b03f2:28765'/*!*/;
# at 569
#180420 17:09:48 server id 49246 end_log_pos 632 CRC32 0x03de76ae Query thread_id=16413
exec_time=4294967295 error_
code=0
SET TIMESTAMP=1524215388/*!*/;
BEGIN
/*!*/;
# at 632
#180420 17:09:48 server id 49246 end_log_pos 679 CRC32 0xb60b3faa Table_map: `ttt`.`t1` mapped
to number 100172
# at 679
#180420 17:09:48 server id 49246 end_log_pos 726 CRC32 0xa2092f94 Write_rows: table id 100172
flags: STMT_END_F

BINLOG '
XK7ZWWhNewAAALwAAAKcCAAAAAEyHAQAAAAEAA3R0dAACdDEAAgMPAjjwAAqo/C7Y=
XK7ZWWh5ewAAALwAAANYCAAAAAEyHAQAAAAEAAgAC//wFAAABNhpyW9taZQvCaI=
'/*!*/;
### INSERT INTO `ttt`.`t1`
### SET
###   @1=5 /* INT meta=0 nullable=0 is_null=0 */

```

```

### @2='xiaomi' /* VARSTRING(60) meta=60 nullable=1 is_null=0 */
# at 726
#180420 17:09:48 server id 49246 end_log_pos 757 CRC32 0xf7d054a7 Xid = 397445
COMMIT/*!*/;
# at 757
#180420 17:11:28 server id 49246 end_log_pos 822 CRC32 0xb24fe15b GTID last_committed=2
sequence_number=3
SET @@SESSION.GTID_NEXT= 'fda7506d-33ea-11e8-b187-000c298b03f2:28766'/*!*/;
# at 822
#180420 17:11:28 server id 49246 end_log_pos 885 CRC32 0xdb150b55 Query thread_id=16413
exec_time=4294967295 error_
code=0
SET TIMESTAMP=1524215488/*!*/;
BEGIN
/*!*/;
# at 885
#180420 17:11:28 server id 49246 end_log_pos 932 CRC32 0xf91a1b61 Table_map: `ttt`.`t1` mapped
to number 100172
# at 932
#180420 17:11:28 server id 49246 end_log_pos 979 CRC32 0xafce0c68 Write_rows: table id 100172
flags: STMT_END_F

BINLOG '
wK7ZWWhNewAAALwAAAKQDAAAAAEyHAQAAAAEAA3R0dAACdDEAAgMPAjjwAAmEbGvk=
wK7ZWWh5ewAAALwAAANMDAAAAAEyHAQAAAAEAAgAC//wIAAAABnhpYW9taWgMzq8=
'/*!*/;
### INSERT INTO `ttt`.`t1`
### SET
### @1=8 /* INT meta=0 nullable=0 is_null=0 */
### @2='xiaomi' /* VARSTRING(60) meta=60 nullable=1 is_null=0 */
# at 979
#180420 17:11:28 server id 49246 end_log_pos 1010 CRC32 0x6e2f666e Xid = 397447
COMMIT/*!*/;
# at 1010
#180420 17:13:34 server id 49247 end_log_pos 1054 CRC32 0x02255fe1 Rotate to binlog.000052
pos: 4
SET @@SESSION.GTID_NEXT= 'AUTOMATIC' /* added by mysqlbinlog */ /*!*/;
DELIMITER ;
# End of log file
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
[root@node02 scripts]#

```

3.5 public_position文件是空时，执行增备

- 如果public_position文件是空的，就会从新执行全备

```

[root@node02 scripts]# > /mybak/public_position --清空文件

[root@node02 scripts]# ./bak_mysql.sh incr 2>/dev/null
+-----+
| Backup_Host |

```



```

+-----+
| node02.mysql.com |
+-----+

mysql连接正常
创建INCR_BACKUP_DIR目录
/mybak/incr/incr_20180420172634
/mybak/incr/incr_20180420172634/backup_incr.log 不存在，重新创建。
OLD_NUM : PUBLIC_POSITION 没有获取到数值,执行全备                                --这里执行全备
0、(1)成功导出 7 个用户权限
0、(2)成功导出 7 个用户帐号
1、20180420 17:26:35 开始备份.....
2、备份以下数据库：
   cmd   ttt
3、20180420 17:28:05 备份成功.....
4、备份用时：90 秒
5、备份数据量大小：256M
6、记录最新的binlog文件名！
开始导出master帐号和权限信息

+-----+
| Master_Host      |
+-----+

| mycat01.mysql.com |
+-----+

0、(1)master成功导出 8 个用户权限
0、(2)master成功导出 8 个用户帐号
全备执行成功
增量备份失败
增量失败,删除备份目录  --注意：删除刚才创建的增量

[root@node02 scripts]# ll /mybak/incr/incr_20180420172634  --增量目录，显示已被删除
ls: cannot access /mybak/incr/incr_20180420172634: No such file or directory

```

二 MySQL Recovery

恢复使用全备进行恢复

1.全备目录

```

[root@node02 scripts]# ls -l /mybak/full/full_20180424/
total 258336
-rw-r--r-- 1 root root      526 Apr 24 12:01 backup_full.log
-rw-r--r-- 1 root root      176 Apr 24 12:00 dbname
-rw-r--r-- 1 root root 264506232 Apr 24 12:01 fullbak.sql.gz
-rw-r--r-- 1 root root     1134 Apr 24 12:00 grants.sql
-rw-r--r-- 1 root root     1200 Apr 24 12:01 master_grants.sql
-rw-r--r-- 1 root root     1630 Apr 24 12:01 master_users.sql
-rw-r--r-- 1 root root      187 Apr 24 12:01 position
-rw-r--r-- 1 root root     1438 Apr 24 12:00 users.sql
[root@node02 scripts]#

```

2.GTID 模式下恢复

- 恢复从库再重做主从

GTID模式在原机上重做从库，需要reset master，清空master，再导入

- 步骤1

```
(root@localhost) 16:21:27 [(none)]> stop slave;
Query OK, 0 rows affected, 1 warning (0.00 sec)

(root@localhost) 16:21:58 [(none)]> reset master;
Query OK, 0 rows affected (0.16 sec)
```

- 步骤2

```
[root@node02 full_20180424]# cat position
-- GTID state at the beginning of the backup
SET @@GLOBAL.GTID_PURGED='84865d81-b573-11e7-9668-b8ca3a65693c:1-57436835';
-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin.000079', MASTER_LOG_POS=194;

[root@node02 full_20180424]# gunzip <fullbak.sql.gz |mysql -uroot -pxxxxxxx
mysql: [Warning] Using a password on the command line interface can be insecure.
```

- 步骤3 CHANGE MASTER

- GTID模式执行

- CHANGE MASTER TO

MASTER_HOST='192.168.xxx.xxx',MASTER_USER='repl',MASTER_PASSWORD='XXXXX',MASTER_AUTO_POSITION=1;

- 非GTID模式执行

- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin.000079',

MASTER_LOG_POS=194,MASTER_HOST='192.168.101.137',MASTER_PORT=3306,MASTER_USER='repl',MASTER_PASSWORD='XXXXX';

```
(root@localhost) 16:53:09 [(none)]> CHANGE MASTER TO
MASTER_HOST='192.168.101.137',MASTER_USER='repl',MASTER_PASSWORD='XXXXX',MASTER_AUTO_POSITION=1;
Query OK, 0 rows affected, 2 warnings (0.31 sec)

(root@localhost) 16:53:27 [(none)]> start slave ;
Query OK, 0 rows affected (0.00 sec)

(root@localhost) 16:53:34 [(none)]> show slave status \G;
***** 1. row *****
Slave_IO_State: Queueing master event to the relay log
Master_Host: 192.168.101.137
Master_User: repl
Master_Port: 3306
Connect_Retry: 60
Master_Log_File: mysql-bin.000081
```

```
Read_Master_Log_Pos: 348638
Relay_Log_File: node01-relay-bin.000002
Relay_Log_Pos: 869569
Relay_Master_Log_File: mysql-bin.000079
Slave_IO_Running: Yes
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: 869396
Relay_Log_Space: 29714750
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: 49801
Master_SSL_Verify_Server_Cert: No
Last_IO_Errno: 0
Last_IO_Error:
Last_SQL_Errno: 0
Last_SQL_Error:
Replicate_Ignore_Server_Ids:
Master_Server_Id: 1
Master_UUID: 84865d81-b573-11e7-9668-b8ca3a65693c
Master_Info_File: /r2/mysqldata/master.info
SQL_Delay: 0
SQL_Remaining_Delay: NULL
Slave_SQL_Running_State: update
Master_Retry_Count: 86400
Master_Bind:
Last_IO_Error_Timestamp:
Last_SQL_Error_Timestamp:
Master_SSL_Crl:
Master_SSL_Crlpath:
Retrieved_Gtid_Set: 84865d81-b573-11e7-9668-b8ca3a65693c:57436836-57469553
Executed_Gtid_Set: 84865d81-b573-11e7-9668-b8ca3a65693c:1-57437364
Auto_Position: 1
Replicate_Rewrite_DB:
Channel_Name:
Master_TLS_Version:
1 row in set (0.26 sec)
```

```
ERROR:
No query specified
```

3.只导出master用户帐号和权限

```
[root@node02 scripts]# sh bak_mysql.sh oemu 2>/dev/null
+-----+
| Backup_Host      |
+-----+
| node02.mysql.com |
+-----+
mysql连接正常
master 192.168.49.246开始导出帐号和权限信息
+-----+
| Master_Host      |
+-----+
| mycat01.mysql.com |
+-----+
master 192.168.49.246成功导出 8 个用户权限
master 192.168.49.246成功导出 8 个用户帐号
master 192.168.49.246导出用户帐号和权限成功
[root@node02 scripts]# ll /mybak/
total 20
drwxr-xr-x  4 root root   48 Apr 24 13:46 full
drwxr-xr-x 27 root root 4096 Apr 24 13:46 incr
-rw-r--r--  1 root root 1200 Apr 25 08:40 master_grants.sql  --导出权限
-rw-r--r--  1 root root 1630 Apr 25 08:40 master_users.sql  --导出帐号
-rw-r--r--  1 root root  232 Apr 25 08:40 public_backup.log
-rw-r--r--  1 root root   17 Apr 24 13:46 public_position
[root@node01 scripts]#
```

4.恢复主库

从库做全备在主库上恢复(步骤同上1-2), 创建帐号和权限 (master_users.sql , master_grants.sql) ,刷新权限 (flush privileges;),然后做全备,再做从库