# mysql数据库误删除后的数据恢复操作说明

在日常运维工作中，对于mysql数据库的备份是至关重要的！数据库对于网站的重要性使得我们对mysql数据的管理不容有失！
然后，是人总难免会犯错误，说不定哪天大脑短路了来个误操作把数据库给删除了，怎么办？？？
下面，就mysql数据库误删除后的恢复方案进行说明。

## 一、工作场景

（1）MySQL数据库每晚12:00自动完全备份。
（2）某天早上上班，9点的时候，一同事犯晕drop了一个数据库！
（3）需要紧急恢复！可利用备份的数据文件以及增量的binlog文件进行数据恢复。

## 二、数据恢复思路

（1）利用全备的sql文件中记录的CHANGE MASTER语句，binlog文件及其位置点信息，找出binlog文件中增量的那部分。
（2）用mysqlbinlog命令将上述的binlog文件导出为sql文件，并剔除其中的drop语句。
（3）通过全备文件和增量binlog文件的导出sql文件，就可以恢复到完整的数据。

## 三、实例说明

---

首先，要确保mysql开启了binlog日志功能
在/etc/my.cnf文件里的[mysqld]区块添加：
log-bin=mysql-bin
然后重启mysql服务

---

（1）在ops库下创建一张表customers

```
mysql> use ops;
mysql> create table customers(
-> id int not null auto_increment,
-> name char(20) not null,
-> age int not null,
-> primary key(id)
-> )engine=InnoDB;
Query OK, 0 rows affected (0.09 sec)
mysql> show tables;
+---------------+
| Tables_in_ops |
+---------------+
| customers |
+---------------+
1 row in set (0.00 sec)
mysql> desc customers;
+-------+----------+------+-----+---------+----------------+
| Field | Type | Null | Key | Default | Extra |
+-------+----------+------+-----+---------+----------------+
| id | int(11) | NO | PRI | NULL | auto_increment |
| name | char(20) | NO | | NULL | |
| age | int(11) | NO | | NULL | |
+-------+----------+------+-----+---------+----------------+
3 rows in set (0.02 sec)
mysql> insert into customers values(1,"wangbo","24");
Query OK, 1 row affected (0.06 sec)
mysql> insert into customers values(2,"guohui","22");
Query OK, 1 row affected (0.06 sec)
mysql> insert into customers values(3,"zhangheng","27");
Query OK, 1 row affected (0.09 sec)
mysql> select * from customers;
+----+-----------+-----+
| id | name | age |
+----+-----------+-----+
| 1 | wangbo | 24 |
| 2 | guohui | 22 |
```

| 3 | zhangheng | 27 |

+----+-----------+-----+

3 rows in set (0.00 sec)

（2）现在进行全备份

[root@vm-002 ~]# mysqldump -uroot -p -B -F -R -x --master-data=2 ops|gzip >/opt/backup/ops_$(date +%F).sql.gz

Enter password:

[root@vm-002 ~]# ls /opt/backup/

ops_2016-09-25.sql.gz

---

参数说明：

-B：指定数据库

-F：刷新日志

-R：备份存储过程等

-x：锁表

--master-data：在备份语句里添加CHANGE MASTER语句以及binlog文件及位置点信息

---

（3）再次插入数据

mysql> insert into customers values(4,"liupeng","21");

Query OK, 1 row affected (0.06 sec)

mysql> insert into customers values(5,"xiaoda","31");

Query OK, 1 row affected (0.07 sec)

mysql> insert into customers values(6,"fuaiai","26");

Query OK, 1 row affected (0.06 sec)

mysql> select * from customers;

+----+-----------+-----+

| id | name | age |

+----+-----------+-----+

| 1 | wangbo | 24 |

| 2 | guohui | 22 |

| 3 | zhangheng | 27 |

| 4 | liupeng | 21 |

| 5 | xiaoda | 31 |

| 6 | fuaiai | 26 |

+----+-----------+-----+

6 rows in set (0.00 sec)

（4）此时误操作，删除了test数据库

mysql> drop database ops;

Query OK, 1 row affected (0.04 sec)

此时，全备之后到误操作时刻之间，用户写入的数据在binlog中，需要恢复出来!

(5)查看全备之后新增的binlog文件

[root@vm-002 ~]# cd /opt/backup/

[root@vm-002 backup]# ls

ops_2016-09-25.sql.gz

[root@vm-002 backup]# gzip -d ops_2016-09-25.sql.gz

[root@vm-002 backup]# ls

ops_2016-09-25.sql

[root@vm-002 backup]# grep CHANGE ops_2016-09-25.sql

-- CHANGE MASTER TO MASTER_LOG_FILE='mysql-bin.000002', MASTER_LOG_POS=106;

这是全备时刻的binlog文件位置

即mysql-bin.000002的106行，因此在该文件之前的binlog文件中的数据都已经包含在这个全备的sql文件中了

（6）移动binlog文件，并导出为sql文件，剔除其中的drop语句

查看mysql的数据存放目录，有下面可知是在/var/lib/mysql下

[root@vm-002 backup]# ps -ef|grep mysql

root 9272 1 0 01:43 pts/1 00:00:00 /bin/sh /usr/bin/mysqld_safe --datadir=/var/lib/mysql --socket=/var/lib/mysql/mysql.sock --pid-file=/var/run/mysqld/mysqld.pid --basedir=/usr --user=mysql

mysql 9377 9272 0 01:43 pts/1 00:00:00 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql --user=mysql --log-error=/var/log/mysqld.log --pid-file=/var/run/mysqld/mysqld.pid --socket=/var/lib/mysql/mysql.sock

[root@vm-002 backup]# cd /var/lib/mysql/

```
[root@vm-002 mysql]# ls
ibdata1 ib_logfile0 ib_logfile1 mysql mysql-bin.000001 mysql-bin.000002 mysql-bin.index mysql.sock test
[root@vm-002 mysql]# cp mysql-bin.000002 /opt/backup/
```

将binlog文件导出sql文件，并vim编辑它删除其中的drop语句

```
[root@vm-002 backup]# mysqlbinlog -d ops mysql-bin.000002 >002bin.sql
[root@vm-002 backup]# ls
002bin.sql mysql-bin.000002 ops_2016-09-25.sql
[root@vm-002 backup]# vim 002bin.sql #删除里面的drop语句
```

注意：

在恢复全备数据之前必须将该binlog文件移出，否则恢复过程中，会继续写入语句到binlog，最终导致增量恢复数据部分变得比较混乱

（7）恢复数据

```
[root@vm-002 backup]# mysql -uroot -p < ops_2016-09-25.sql
Enter password:
[root@vm-002 backup]#
```

查看数据库，看看ops库在不在

```
mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
| ops |
| test |
+--------------------+
4 rows in set (0.00 sec)
mysql> use ops;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
mysql> select * from customers;
+----+-----------+-----+
| id | name | age |
+----+-----------+-----+
| 1 | wangbo | 0 |
| 2 | guohui | 0 |
| 3 | zhangheng | 0 |
+----+-----------+-----+
3 rows in set (0.00 sec)
```

此时恢复了全备时刻的数据

接着，使用002bin.sql文件恢复全备时刻到删除数据库之间，新增的数据

```
[root@vm-002 backup]# mysql -uroot -p ops <002bin.sql
Enter password:
[root@vm-002 backup]#
```

再次查看数据库，发现全备份到删除数据库之间的那部分数据也恢复了！！

```
mysql> select * from customers;
+----+-----------+-----+
| id | name | age |
+----+-----------+-----+
| 1 | wangbo | 24 |
| 2 | guohui | 22 |
| 3 | zhangheng | 27 |
| 4 | liupeng | 21 |
| 5 | xiaoda | 31 |
| 6 | fuaiai | 26 |
+----+-----------+-----+
6 rows in set (0.00 sec)
```

以上就是mysql数据库增量数据恢复的实例过程!

**********

最后，总结几点：

1）本案例适用于人为SQL语句造成的误操作或者没有主从复制等的热备情况宕机时的修复

2）恢复条件为mysql要开启binlog日志功能，并且要全备和增量的所有数据

3）恢复时建议对外停止更新，即禁止更新数据库

4）先恢复全量，然后把全备时刻点以后的增量日志，按顺序恢复成SQL文件，然后把文件中有问题的SQL语句删除（也可通过时间和位置点），再恢复到数据库。

**********当你发现自己的才华撑不起野心时，就请安静下来学习吧**********

参考 ： www.cnblogs.com/kevingrace/p/5904800.html