

mysql binlog 基本使用

作者: [moloee](#)

原文链接: <https://ld246.com/article/1510886875242>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

先查看下数据库binlog的状态

```
show variables like 'log_bin';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| log_bin      | OFF  |
+-----+-----+
```

如果关闭状态的话，需求配置下开启，修改[windows]my.ini/[linux]my.cnf文件，linux下/etc/my.cnf在[mysqld]下添加log-bin=[日志名]

```
log-bin=mysql-bin
```

重启mysql，在centos 下安装的是mariadb，所以重启:

```
systemctl mariadb.service restart
```

然后查看状态，就是 log_bin=on了。

一次执行一些数据库的操作测试下:

```
MariaDB [binlog]> create table test( id int auto_increment not null primary key, val int, data v
rchar(20));
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [binlog]> insert into test(val, data) values (10, 'wu');
Query OK, 1 row affected (0.01 sec)
```

```
MariaDB [binlog]> insert into test(val, data) values (20, 'yang');
Query OK, 1 row affected (0.00 sec)
```

```
MariaDB [binlog]> flush logs;
Query OK, 0 rows affected (0.20 sec)
```

```
MariaDB [binlog]> insert into test(val, data) values (20, 'ping');
Query OK, 1 row affected (0.01 sec)
```

```
MariaDB [binlog]> insert into test(val, data) values (20, 'pinlogng');
Query OK, 1 row affected (0.00 sec)
```

```
MariaDB [binlog]> delete from test where id between 4 and 5;
Query OK, 1 row affected (0.00 sec)
```

```
MariaDB [binlog]> flush logs;
Query OK, 0 rows affected (0.02 sec)
```

以下是第一次flush之前，和之后的变化

```
# flush 之前
-rw-rw---- 1 mysql mysql    958 11月 17 10:31 mysql-bin.000001
-rw-rw---- 1 mysql mysql    19 11月 17 10:25 mysql-bin.index
```

```
#flush 之后
```

```
-rw-rw---- 1 mysql mysql 1001 11月 17 10:32 mysql-bin.000001
-rw-rw---- 1 mysql mysql 245 11月 17 10:32 mysql-bin.000002
-rw-rw---- 1 mysql mysql 38 11月 17 10:32 mysql-bin.index
```

.index是索引文件，其他三个是binlog文件，我们可以用[mysqlbinlog] 工具来恢复数据。为了下面解的方便，我们先将binlog文件解析成txt文件，如下：

```
mysqlbinlog data\mysql-bin.000001 > ./1.txtmysqlbinlog data\mysql-bin.000002 > ./2.txtmysqlbinlog data\mysql-bin.000003 > ./3.txt
```

重复第一个log文件的操作

```
mysqlbinlog data\mysql-bin.000001 | mysql -uroot -p123456
```

在第二个binlog里面我们进行了delete操作，我们并不想将delete的操作恢复到数据库，这样我们可以通过读取2.txt文件：

```
...../*!*/;# at 653#140902 16:07:43 server id 1 end_log_pos 759 Query thread_id=
exec_time=0 error_code=0SET TIMESTAMP=1409645263/*!*/;delete from test where id betw
en 4 and 5/*!*/;# at 759#140902 16:07:43 server id 1 end_log_pos 786 Xid = 175COMMIT/*!*/;.
.....
```

这里介绍下

```
--start-position=N #设置实际的偏移量
--stop-position=N #设置终止的偏移量
```

在这个文件中，我们可以看到DELETE的操作的起始位置是653，终止位置是759.那么我们只要重做第一个日志文件的开头到653的操作，然后再从759到末尾的操作，我们就可以把数据给恢复回来，而不会DELETE数据。所以执行两个命令

```
mysqlbinlog data\mysql-bin.000002 --stop-pos=653 | mysql -uroot -p123456mysqlbinlog da
a\mysql-bin.000002 --start-pos=759 | mysql -uroot -p123456mysqlbinlog data\mysql-bin.00
003 --stop-pos=587 | mysql -uroot -p123456
```