mysqlbinlog_flashback 数据回滚测试

> 2017-08-23 驻云科技 黄静雪

参考地址：https://github.com/58daojia-dba/mysqlbinlog_flashback

> 目前运行情况 现在已经在阿里的rds上，db为utf8字符集的生产环境下使用。其他环境没有在生产环境下使用，请小心。 #工具简介 ##概述 mysqlbinlog_back.py 是在线读取row格式的mysqld的binlog,然后生成反向的sql语句的工具。一般用于数据恢复的目的。 所谓反向的sql语句就是如果是insert，则反向的sql为delete。如果delete,反向的sql是insert,如果是update, 反向的sql还是update,但是update的值是原来的值。

一、基本信息

1.1 rds 基本信息

产品名称	实例配置	数量	付费方式	资费
服务商：阿里云计算有限公司				
1. 关系型数据库（RDS按量计费）	地域：华东 1 可用区：可用区D 数据库类型：MySQL 数据库版本号：5.6 存储空间：20G 网络类型：经典网络 规格：（1核1G）(连接数:300 IOPS:600) 系列：高可用版	1台	按量付费	￥0.324 / 时

实例ID	rm-bp1k1hlze8ix9rw2z
地域可用区	华东1可用区D
内网地址	rm-bp1k1hlze8ix9rw2z.mysql.rds.aliyuncs.com
内网端口	3306

用户名：hjx

密码：Zhuyun@123

1.2 ECS基本信息

二、测试过程

2.1 ECS安装mysql_flashback

```
[test@/var/log]#wget https://github.com/58daojia-dba/mysqlbinlog_flashback/archive/master.zip
[test@/root]#unzip master.zip
Archive:  master.zip
555fd1e8c0b49bdf3d7b0f4de94039b957ff3161
   creating: mysqlbinlog_flashback-master/
  inflating: mysqlbinlog_flashback-master/.gitignore
  inflating: mysqlbinlog_flashback-master/CHANGELOG.txt
  inflating: mysqlbinlog_flashback-master/LICENSE
  inflating: mysqlbinlog_flashback-master/README.md
  inflating: mysqlbinlog_flashback-master/binlogstream.py.diff
  inflating: mysqlbinlog_flashback-master/constant.py
  inflating: mysqlbinlog_flashback-master/flashback.py
  inflating: mysqlbinlog_flashback-master/func.py
   creating: mysqlbinlog_flashback-master/internal/
  inflating: mysqlbinlog_flashback-master/internal/flashback_internal.ppt
  inflating: mysqlbinlog_flashback-master/joint_sql.py
   creating: mysqlbinlog_flashback-master/log/
 extracting: mysqlbinlog_flashback-master/log/test.txt
  inflating: mysqlbinlog_flashback-master/mysql_table.py
  inflating: mysqlbinlog_flashback-master/mysqlbinlog_back.py
   creating: mysqlbinlog_flashback-master/pymysqlreplication/
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/__init__.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/binlogstream.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/bitmap.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/column.py
   creating: mysqlbinlog_flashback-master/pymysqlreplication/constants/
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/constants/BINLOG.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/constants/FIELD_TYPE.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/constants/__init__.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/event.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/gtid.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/packet.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/row_event.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/table.py
   creating: mysqlbinlog_flashback-master/pymysqlreplication/tests/
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/tests/__init__.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/tests/base.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/tests/benchmark.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/tests/test_basic.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/tests/test_data_objects.py
  inflating: mysqlbinlog_flashback-master/pymysqlreplication/tests/test_data_type.py
   creating: mysqlbinlog_flashback-master/test/
   creating: mysqlbinlog_flashback-master/test/log/
 extracting: mysqlbinlog_flashback-master/test/log/test.txt
  inflating: mysqlbinlog_flashback-master/test/test_generate_table_name.py
  inflating: mysqlbinlog_flashback-master/test/test_mysql_table.py
  inflating: mysqlbinlog_flashback-master/test/test_mysqlbinlog_back.py
  inflating: mysqlbinlog_flashback-master/test/test_parameter.py

  [test@/root]#cd mysqlbinlog_flashback-master/
[test@/root/mysqlbinlog_flashback-master]#ll
total 100
-rw-r--r-- 1 root root  4766 Dec 19  2016 binlogstream.py.diff
```

```
-rw-r--r-- 1 root root   524 Dec 19  2016 CHANGELOG.txt
-rw-r--r-- 1 root root  1365 Dec 19  2016 constant.py
-rw-r--r-- 1 root root 13730 Dec 19  2016 flashback.py
-rw-r--r-- 1 root root  1800 Dec 19  2016 func.py
drwxr-xr-x 2 root root  4096 Dec 19  2016 internal
-rw-r--r-- 1 root root 10015 Dec 19  2016 joint_sql.py
-rw-r--r-- 1 root root 11357 Dec 19  2016 LICENSE
drwxr-xr-x 2 root root  4096 Dec 19  2016 log
-rw-r--r-- 1 root root  9192 Dec 19  2016 mysqlbinlog_back.py
-rw-r--r-- 1 root root  2782 Dec 19  2016 mysql_table.py
drwxr-xr-x 4 root root  4096 Dec 19  2016 pymysqlreplication
-rw-r--r-- 1 root root  5033 Dec 19  2016 README.md
drwxr-xr-x 3 root root  4096 Dec 19  2016 test
```

2.2 mysqlbinlog_back.py帮助

```
# python mysqlbinlog_back.py帮助
[test@/root/mysqlbinlog_flashback-master]#python mysqlbinlog_back.py --help
===log will also  write to .//mysqlbinlog_flashback.log===
Usage: python mysqlbinlog_back.py [options]
sample1:python  mysqlbinlog_back.py --host="127.0.0.1" --username="root" --port=43306 --
password="" --schema=test --table="test5"
sample2:python mysqlbinlog_back.py --host="127.0.0.1" --username="root" --port=43306 --
password="" --schema=test --table="test5,test6" --binlog_end_time="2016-11-05 11:27:13" --
binlog_start_file_name="mysql-bin.000024"  --binlog_start_file_position=4 --
binlog_start_time="2016-11-04 11:27:13"  --skip_delete  --skip_insert --add_schema_name
sample3:python mysqlbinlog_back.py  --host="127.0.0.1" --username="root" --port=43306 --
password="" --schema=test --table="test5,test6" --binlog_start_file_name="mysql-bin.000022"

Options:
  -h, --help            show this help message and exit
  -H HOST, --host=HOST  mandatory,mysql hostname
  -P PORT, --port=PORT  mysql port,default 3306
  -u USERNAME, --username=USERNAME
                        mandatory,username
  -p PASSWORD, --password=PASSWORD
                        password
  -s SCHEMA, --schema=SCHEMA
                        mandatory,mysql schema
  -t TABLES, --tables=TABLES
                        mandatory,mysql tables,suport multiple tables,use
                        comma as separator
  -N BINLOG_END_TIME, --binlog_end_time=BINLOG_END_TIME
                        binlog end time,format yyyy-mm-dd hh24:mi:ss,default
                        is current time
  -S BINLOG_START_FILE_NAME, --binlog_start_file_name=BINLOG_START_FILE_NAME
                        binlog start file name,default is current logfile of
                        db
  -L BINLOG_START_FILE_POSITION, --binlog_start_file_position=BINLOG_START_FILE_POSITION
                        binlog start file name
  -E BINLOG_START_TIME, --binlog_start_time=BINLOG_START_TIME
                        binlog start time,format yyyy-mm-dd hh24:mi:ss
  -l OUTPUT_FILE_PATH, --output_file_path=OUTPUT_FILE_PATH
                        file path that sql generated,,default ./log
  -I, --skip_insert     skip insert(WriteRowsEvent) event
  -U, --skip_update     skip update(UpdateRowsEvent) event
  -D, --skip_delete     skip delete(DeleteRowsEvent) event
  -a, --add_schema_name
                        add schema name for flashback sql
  -v, --version         version info
```

2.3 创建测试环境

```
#创建表DDL
[test@/root]#mysql -uhjx -p -hrm-bp1k1hlze8ix9rw2z.mysql.rds.aliyuncs.com
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 805764173
Server version: 5.6.34 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use hjxdb;
Database changed
mysql> CREATE TABLE `user` (
    ->   `id` bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '用户唯一标志符 UID',
    ->   `username` varchar(64) DEFAULT NULL COMMENT '用户名，不区分大小写',
    ->   `email` varchar(128) DEFAULT NULL COMMENT '注册邮箱，不区分大小写',
    ->   `cell_phone` bigint(11) DEFAULT NULL COMMENT '手机号码',
    ->   `password` char(32) NOT NULL COMMENT '密码hash值',
    ->   `school_code` bigint(20) unsigned DEFAULT NULL COMMENT '学校代码',
    ->   `register_time` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00' COMMENT '注册时间',
    ->   `usertype` int(5) NOT NULL DEFAULT '1' COMMENT '1为微信关注用户，2为微信登录app的用户，3为
APP端微信登录的微信用户',
    ->   `state` tinyint(4) NOT NULL DEFAULT '1' COMMENT '1<0: UID是否有效 1<1: 是否设置用户名密码
1<2: 是否认证邮箱 1<3: 是否认证手机号码',
    ->   PRIMARY KEY (`id`)
    -> ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
Query OK, 0 rows affected (0.01 sec)

#插入数据
mysql> insert into user values
(1,'apple','apple@123.com','12233334444','hahaha',123,current_time(),1,2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into user values
(2,'pple','pple@123.com','12233334445','hahaha',124,current_time(),1,2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into user values
(3,'ple','ple@123.com','12233334446','hahaha',125,current_time(),1,2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into user values
(4,'le','le@123.com','12233334447','hahaha',126,current_time(),1,2);
Query OK, 1 row affected (0.00 sec)

#查询数据
mysql> select * from user;

+----+----------+--------------+-------------+----------+-------------+---------------------+---
```

```
-------+-------+
| id | username | email         | cell_phone  | password | school_code | register_time       |
usertype | state |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
|  1 | apple    | apple@123.com | 12233334444 | hahaha   |         123 | 2017-08-23 13:58:47 |
     1 |     2 |
|  2 | pple     | pple@123.com  | 12233334445 | hahaha   |         124 | 2017-08-23 13:59:15 |
     1 |     2 |
|  3 | ple      | ple@123.com   | 12233334446 | hahaha   |         125 | 2017-08-23 13:59:35 |
     1 |     2 |
|  4 | le       | le@123.com    | 12233334447 | hahaha   |         126 | 2017-08-23 13:59:52 |
     1 |     2 |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
4 rows in set (0.00 sec)
```

2.4 误修改数据

```
#误将所有记录cell_phone 值更改为15811111111
mysql> update user set cell_phone=15811111111;
Query OK, 4 rows affected (0.00 sec)
Rows matched: 4  Changed: 4  Warnings: 0

#查看数据
mysql> select * from user;
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
| id | username | email         | cell_phone  | password | school_code | register_time       |
usertype | state |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
|  1 | apple    | apple@123.com | 15811111111 | hahaha   |         123 | 2017-08-23 13:58:47 |
     1 |     2 |
|  2 | pple     | pple@123.com  | 15811111111 | hahaha   |         124 | 2017-08-23 13:59:15 |
     1 |     2 |
|  3 | ple      | ple@123.com   | 15811111111 | hahaha   |         125 | 2017-08-23 13:59:35 |
     1 |     2 |
|  4 | le       | le@123.com    | 15811111111 | hahaha   |         126 | 2017-08-23 13:59:52 |
     1 |     2 |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
4 rows in set (0.00 sec)
```

> 数据当前是误操作更改状态

2.5 闪回操作

```
[test@/root/mysqlbinlog_flashback-master]#python mysqlbinlog_back.py --host="rm-bp1k1hlze8ix9rw2z.mysql.rds.aliyuncs.com" --username="hjx" --port=3306 --password="Zhuyun@123" --schema=hjxdb --table="user"
===log will also  write to .//mysqlbinlog_flashback.log===
parameter={'start_binlog_file': u'mysql-bin.000004', 'stream': None, 'keep_data': True, 'file': {'data_create': None, 'flashback': None, 'data': None}, 'add_schema_name': False, 'start_time': None, 'keep_current_data': False, 'start_to_timestamp': None, 'mysql_setting': {'passwd': 'Zhuyun@123', 'host': 'rm-bp1k1hlze8ix9rw2z.mysql.rds.aliyuncs.com', 'charset': 'utf8', 'port': 3306, 'user': 'hjx'}, 'table_name': 'user', 'skip_delete': False, 'schema': 'hjxdb', 'stat': {'flash_sql': {}}, 'table_name_array': ['user'], 'one_binlog_file': False, 'output_file_path': './log', 'start_position': 4, 'skip_update': False, 'dump_event': False, 'end_to_timestamp': 1503468644.0, 'skip_insert': False, 'schema_array': ['hjxdb']}
scan 10000 events ....from binlogfile=mysql-bin.000004,timestamp=2017-08-23T13:24:44
scan 20000 events ....from binlogfile=mysql-bin.000004,timestamp=2017-08-23T13:24:45
scan 30000 events ....from binlogfile=mysql-bin.000004,timestamp=2017-08-23T13:24:46
===statistics===
scan 34856 events
{'flash_sql': {u'hjxdb': {u'user': {'insert': 0, 'update': 4, 'delete': 4}}}}

[test@/root/mysqlbinlog_flashback-master]#ls -l log/**
-rw-r--r-- 1 root root 1295 Aug 23 14:10 log/flashback_hjxdb_20170823_141044.sql
-rw-r--r-- 1 root root  565 Aug 23 14:10 log/save_data_create_table_hjxdb_20170823_141044.sql
-rw-r--r-- 1 root root 3567 Aug 23 14:10 log/save_data_dml_hjxdb_20170823_141044.sql
-rw-r--r-- 1 root root    0 Dec 19  2016 log/test.txt

[test@/root/mysqlbinlog_flashback-master/log]#cat flashback_hjxdb_20170823_141044.sql
#end_log_pos 4160028  2017-08-23T13:58:47 1503467927 mysql-bin.000004;
delete from `user` where `id`=1;
#end_log_pos 4161142  2017-08-23T13:59:15 1503467955 mysql-bin.000004;
delete from `user` where `id`=2;
#end_log_pos 4161989  2017-08-23T13:59:35 1503467975 mysql-bin.000004;
delete from `user` where `id`=3;
#end_log_pos 4162569  2017-08-23T13:59:52 1503467992 mysql-bin.000004;
delete from `user` where `id`=4;
#end_log_pos 4166746  2017-08-23T14:02:09 1503468129 mysql-bin.000004;
update `user` set`username`='apple',`cell_phone`=12233334444,`register_time`='2017-08-23 13:58:47',`school_code`=123,`email`='apple@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id`=1 where `id`=1;
update `user` set`username`='pple',`cell_phone`=12233334445,`register_time`='2017-08-23 13:59:15',`school_code`=124,`email`='pple@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id`=2 where `id`=2;
update `user` set`username`='ple',`cell_phone`=12233334446,`register_time`='2017-08-23 13:59:35',`school_code`=125,`email`='ple@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id`=3 where `id`=3;
update `user` set`username`='le',`cell_phone`=12233334447,`register_time`='2017-08-23 13:59:52',`school_code`=126,`email`='le@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id`=4 where `id`=4;

# 它会在线连接参数指定mysql,读取binlog,仅仅抽取对schema为test 表名test5的binlog，生成反向sq文件保存在log目录下,其中flash_开头的文件是反向的sql语句。



# 详细描述 mysqlbinlog_back.py在线连接参数指定mysql,读取binlog,如果缺省，它通过show binary logs命令找
```

到最近的binlog文件，从文件开头开始解析，一直解析到当前时间退出。

如果指定开始binary log文件名和位置（BINLOG_START_FILE_NAME，BINLOG_START_FILE_POSITION），会从指定binary log文件名和位置开始解析，一直BINLOG_END_TIME结束，中间会自动扫描跨多个binlog.

#生成文件目录可以通过OUTPUT_FILE_PATH来指定。目录下有2个类： 一类是反向解析的文件,格式为flashback_schema名_当前时间.sql . 另一类用于审查数据的sql,审查数据的sql用于记录操作类型，sql的老、新值。其中，save_data_create_table_开头的文件用于生成建表语句，save_data_dml用于插入到新的表中。

```
[test@/root/mysqlbinlog_flashback-master/log]#cat
save_data_create_table_hjxdb_20170823_141044.sql
CREATE TABLE `_user_keep_data_` (op varchar(64),op_datetime datetime,bfr_id bigint(20)
unsigned,bfr_username varchar(64),bfr_email varchar(128),bfr_cell_phone bigint(11),bfr_password
char(32),bfr_school_code bigint(20) unsigned,bfr_register_time timestamp,bfr_usertype
int(5),bfr_state tinyint(4),aft_id bigint(20) unsigned,aft_username varchar(64),aft_email
varchar(128),aft_cell_phone bigint(11),aft_password char(32),aft_school_code bigint(20)
unsigned,aft_register_time timestamp,aft_usertype int(5),aft_state tinyint(4)) ENGINE=InnoDB
DEFAULT CHARSET=utf8mb4;

[test@/root/mysqlbinlog_flashback-master/log]#cat save_data_dml_hjxdb_20170823_141044.sql
#end_log_pos 4160028  2017-08-23T13:58:47 1503467927 mysql-bin.000004;
insert into
`_user_keep_data_`(`aft_school_code`,`aft_id`,`op_datetime`,`aft_username`,`aft_register_time`,`a
ft_cell_phone`,`aft_usertype`,`aft_state`,`aft_password`,`aft_email`,`op`) values(123,1,'2017-08-
23 13:58:47','apple','2017-08-23 13:58:47',12233334444,1,2,'hahaha','apple@123.com','insert');
#end_log_pos 4161142  2017-08-23T13:59:15 1503467955 mysql-bin.000004;
insert into
`_user_keep_data_`(`aft_school_code`,`aft_id`,`op_datetime`,`aft_username`,`aft_register_time`,`a
ft_cell_phone`,`aft_usertype`,`aft_state`,`aft_password`,`aft_email`,`op`) values(124,2,'2017-08-
23 13:59:15','pple','2017-08-23 13:59:15',12233334445,1,2,'hahaha','pple@123.com','insert');
#end_log_pos 4161989  2017-08-23T13:59:35 1503467975 mysql-bin.000004;
insert into
`_user_keep_data_`(`aft_school_code`,`aft_id`,`op_datetime`,`aft_username`,`aft_register_time`,`a
ft_cell_phone`,`aft_usertype`,`aft_state`,`aft_password`,`aft_email`,`op`) values(125,3,'2017-08-
23 13:59:35','ple','2017-08-23 13:59:35',12233334446,1,2,'hahaha','ple@123.com','insert');
#end_log_pos 4162569  2017-08-23T13:59:52 1503467992 mysql-bin.000004;
insert into
`_user_keep_data_`(`aft_school_code`,`aft_id`,`op_datetime`,`aft_username`,`aft_register_time`,`a
ft_cell_phone`,`aft_usertype`,`aft_state`,`aft_password`,`aft_email`,`op`) values(126,4,'2017-08-
23 13:59:52','le','2017-08-23 13:59:52',12233334447,1,2,'hahaha','le@123.com','insert');
#end_log_pos 4166746  2017-08-23T14:02:09 1503468129 mysql-bin.000004;
insert into
`_user_keep_data_`(`aft_cell_phone`,`bfr_cell_phone`,`aft_id`,`op_datetime`,`aft_register_time`,`
aft_username`,`aft_school_code`,`aft_state`,`bfr_email`,`aft_usertype`,`bfr_usertype`,`bfr_passw
ord`,`bfr_register_time`,`aft_password`,`bfr_id`,`bfr_username`,`bfr_school_code`,`aft_email`,`bf
r_state`,`op`) values(15811111111,12233334444,1,'2017-08-23 14:02:09','2017-08-23
13:58:47','apple',123,2,'apple@123.com',1,1,'hahaha','2017-08-23
13:58:47','hahaha',1,'apple',123,'apple@123.com',2,'update');




insert into
```

```
`_user_keep_data_`(`aft_cell_phone`,`bfr_cell_phone`,`aft_id`,`op_datetime`,`aft_register_time`,`
aft_username`,`aft_school_code`,`aft_state`,`bfr_email`,`aft_usertype`,`bfr_usertype`,`bfr_passw
ord`,`bfr_register_time`,`aft_password`,`bfr_id`,`bfr_username`,`bfr_school_code`,`aft_email`,`bf
r_state`,`op`) values(15811111111,12233334445,2,'2017-08-23 14:02:09','2017-08-23
13:59:15','pple',124,2,'pple@123.com',1,1,'hahaha','2017-08-23
13:59:15','hahaha',2,'pple',124,'pple@123.com',2,'update');
insert into
`_user_keep_data_`(`aft_cell_phone`,`bfr_cell_phone`,`aft_id`,`op_datetime`,`aft_register_time`,`
aft_username`,`aft_school_code`,`aft_state`,`bfr_email`,`aft_usertype`,`bfr_usertype`,`bfr_passw
ord`,`bfr_register_time`,`aft_password`,`bfr_id`,`bfr_username`,`bfr_school_code`,`aft_email`,`bf
r_state`,`op`) values(15811111111,12233334446,3,'2017-08-23 14:02:09','2017-08-23
13:59:35','ple',125,2,'ple@123.com',1,1,'hahaha','2017-08-23
13:59:35','hahaha',3,'ple',125,'ple@123.com',2,'update');
insert into
`_user_keep_data_`(`aft_cell_phone`,`bfr_cell_phone`,`aft_id`,`op_datetime`,`aft_register_time`,`
aft_username`,`aft_school_code`,`aft_state`,`bfr_email`,`aft_usertype`,`bfr_usertype`,`bfr_passw
ord`,`bfr_register_time`,`aft_password`,`bfr_id`,`bfr_username`,`bfr_school_code`,`aft_email`,`bf
r_state`,`op`) values(15811111111,12233334447,4,'2017-08-23 14:02:09','2017-08-23
13:59:52','le',126,2,'le@123.com',1,1,'hahaha','2017-08-23
13:59:52','hahaha',4,'le',126,'le@123.com',2,'update');

[test@/root/mysqlbinlog_flashback-master/log]#cat test.txt
```

三、插入新数据

```
# 插入新数据
mysql> insert into user values (5,'e','e@123.com','12233334448','hahaha',127,current_time(),1,2);
Query OK, 1 row affected (0.00 sec)

# 查询新数据
mysql> select * from user;
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
| id | username | email         | cell_phone  | password | school_code | register_time       |
usertype | state |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
|  1 | apple    | apple@123.com | 15811111111 | hahaha   |         123 | 2017-08-23 13:58:47 |
     1 |     2 |
|  2 | pple     | pple@123.com  | 15811111111 | hahaha   |         124 | 2017-08-23 13:59:15 |
     1 |     2 |
|  3 | ple      | ple@123.com   | 15811111111 | hahaha   |         125 | 2017-08-23 13:59:35 |
     1 |     2 |
|  4 | le       | le@123.com    | 15811111111 | hahaha   |         126 | 2017-08-23 13:59:52 |
     1 |     2 |
|  5 | e        | e@123.com     | 12233334448 | hahaha   |         127 | 2017-08-23 14:22:39 |
     1 |     2 |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
5 rows in set (0.00 sec)
```

> 数据当前是误修改数据与插入新数据共存状态

3.1 闪回的数据做恢复

```
#用之前闪回的文件截取更改的部分做恢复（也可直接采用python mysqlbinlog_back.py --help sample2的方式）
[test@/root/mysqlbinlog_flashback-master]#cat u.sql
update `user` set`username`='apple',`cell_phone`=12233334444,`register_time`='2017-08-23
13:58:47',`school_code`=123,`email`='apple@123.com',`usertype`=1,`state`=2,`password`='hahaha',`i
d`=1 where `id`=1;
update `user` set`username`='pple',`cell_phone`=12233334445,`register_time`='2017-08-23
13:59:15',`school_code`=124,`email`='pple@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id
`=2 where `id`=2;
update `user` set`username`='ple',`cell_phone`=12233334446,`register_time`='2017-08-23
13:59:35',`school_code`=125,`email`='ple@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id`
=3 where `id`=3;
update `user` set`username`='le',`cell_phone`=12233334447,`register_time`='2017-08-23
13:59:52',`school_code`=126,`email`='le@123.com',`usertype`=1,`state`=2,`password`='hahaha',`id`=
4 where `id`=4;


[test@/root/mysqlbinlog_flashback-master]#mysql -uhjx -p -hrm-
bp1k1hlze8ix9rw2z.mysql.rds.aliyuncs.com --default-character-set=utf8 hjxdb < u.sql
Enter password:

mysql> select * from user;
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
| id | username | email         | cell_phone  | password | school_code | register_time       |
usertype | state |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
|  1 | apple    | apple@123.com | 12233334444 | hahaha   |         123 | 2017-08-23 13:58:47 |
     1 |     2 |
|  2 | pple     | pple@123.com  | 12233334445 | hahaha   |         124 | 2017-08-23 13:59:15 |
     1 |     2 |
|  3 | ple      | ple@123.com   | 12233334446 | hahaha   |         125 | 2017-08-23 13:59:35 |
     1 |     2 |
|  4 | le       | le@123.com    | 12233334447 | hahaha   |         126 | 2017-08-23 13:59:52 |
     1 |     2 |
|  5 | e        | e@123.com     | 12233334448 | hahaha   |         127 | 2017-08-23 14:22:39 |
     1 |     2 |
+----+----------+---------------+-------------+----------+-------------+---------------------+---
-------+-------+
5 rows in set (0.00 sec)
```

```
mysql> insert into user values (5,'e','e@123.com','12233334448','hahaha',127,current_time(),1,2);
Query OK, 1 row affected (0.00 sec)

mysql> select * from user;
+----+----------+--------------+-------------+----------+-------------+---------------------+----------+-------+
| id | username | email        | cell_phone  | password | school_code | register_time       | usertype | state |
+----+----------+--------------+-------------+----------+-------------+---------------------+----------+-------+
|  1 | apple    | apple@123.com | 15811111111 | hahaha   |         123 | 2017-08-23 13:58:47 |        1 |     2 |
|  2 | pple     | pple@123.com  | 15811111111 | hahaha   |         124 | 2017-08-23 13:59:15 |        1 |     2 |
|  3 | ple      | ple@123.com   | 15811111111 | hahaha   |         125 | 2017-08-23 13:59:35 |        1 |     2 |
|  4 | le       | le@123.com    | 15811111111 | hahaha   |         126 | 2017-08-23 13:59:52 |        1 |     2 |
|  5 | e        | e@123.com     | 12233334448 | hahaha   |         127 | 2017-08-23 14:22:39 |        1 |     2 |
+----+----------+--------------+-------------+----------+-------------+---------------------+----------+-------+
5 rows in set (0.00 sec)

mysql> select * from user;
+----+----------+--------------+-------------+----------+-------------+---------------------+----------+-------+
| id | username | email        | cell_phone  | password | school_code | register_time       | usertype | state |
+----+----------+--------------+-------------+----------+-------------+---------------------+----------+-------+
|  1 | apple    | apple@123.com | 12233334444 | hahaha   |         123 | 2017-08-23 13:58:47 |        1 |     2 |
|  2 | pple     | pple@123.com  | 12233334445 | hahaha   |         124 | 2017-08-23 13:59:15 |        1 |     2 |
|  3 | ple      | ple@123.com   | 12233334446 | hahaha   |         125 | 2017-08-23 13:59:35 |        1 |     2 |
|  4 | le       | le@123.com    | 12233334447 | hahaha   |         126 | 2017-08-23 13:59:52 |        1 |     2 |
|  5 | e        | e@123.com     | 12233334448 | hahaha   |         127 | 2017-08-23 14:22:39 |        1 |     2 |
+----+----------+--------------+-------------+----------+-------------+---------------------+----------+-------+
5 rows in set (0.00 sec)
```

> 至此，rds上有张表，有实时数据一直在写入，人为把这张表的某一列给全部更新掉（这个定为故障点），过一段时间后开始恢复数据，要保证更新的那一列恢复到故障点之前且后续写入的数据不丢失测试完成。