第1章 错误日志

1.作用

```
1  记录数据库启动以来，状态、警告、报错。诊断数据库报错问题。
```

2.配置

```
1  默认:  开启状态。存放在数据目录下(/data/3306/data),名字: 主机名.err
```

3.查看

```
1  mysql> select @@log_error;
2  +-------------+
3  | @@log_error |
4  +-------------+
5  | ./db-51.err |
6  +-------------+
7  1 row in set (0.00 sec)
8
9  mysql> select @@datadir;
10 +-------------------+
11 | @@datadir         |
12 +-------------------+
13 | /data/mysql_3306/ |
14 +-------------------+
15 1 row in set (0.00 sec)
```

4.自定义配置

修改配置:

```
[root@db-51 ~]# vim /etc/my.cnf
[mysqld]
#新增加参数
log_error=/data/mysql_3306/logs/mysql.err
```

创建日志目录并更改授权:

```
[root@db-51 ~]# mkdir /data/mysql_3306/logs/ -p
[root@db-51 ~]# chown -R mysql:mysql /data/mysql_3306/logs/
```

重启mysql:

```
systemctl status mysqld.service
```

重启后报错:

```
9月 13 17:18:57 db-51 mysqld[1439]: Starting MySQL.2020-09-
13T09:18:57.124858Z mysqld_safe error: log-error set to
'/data/mysql_3306/logs/mysql.err', however file don't exists. Create
writable for user 'mysq
```

解决方法:

```
mkdir /data/mysql_3306/logs/ -p
touch /data/mysql_3306/logs/mysql.err
chown -R mysql:mysql /data/mysql_3306/logs/
systemctl restart mysqld.service
```

第2章 慢日志

1.作用

```
1.记录MySQL工作过程中较慢的语句
2.默认没有开启,按需求打开。
```

2.配置

在线配置:

```
mysql> select @@slow_query_log;       # 开关
mysql> set global slow_query_log=1;  # 在线改
mysql> select @@slow_query_log_file; # 文件位置。离线改。
mysql> select @@long_query_time;      # 慢查询时间设定。
mysql> set global long_query_time=0.1; # 在线设置，最低微秒级别。
mysql> select @@log_queries_not_using_indexes #如果没走索引会被记录
mysql> set global log_queries_not_using_indexes=1; #在线设置
```

永久生效:

```
vim /etc/my.cnf
slow_query_log=1     #是否启用慢查询日志，1为启用，0为禁用
slow_query_log_file=/data/mysql_3306/logs/slow.log  #慢日志路径
long_query_time=0.1  #SQL语句运行时间阈值，执行时间大于参数值的语句才会被记录下来
log_queries_not_using_indexes=1  #将没有使用索引的语句记录到慢查询日志
```

3.模拟慢语句

```
select * from t100w as a join t100w as b limit N;
select k1,count(*) from t100w where id<N group by k1 having count(*)>N;
select k1,count(*) from t100w where num<N group by k1,k2;
select * from t100w where id<N order by num  desc;
select k1,count(*) from t100w where id<N group by k1,k2;
```

4.慢日志分析

```
[root@db01 logs]# mysqldumpslow -s c -t 5 slow.log
Reading mysql slow query log from slow.log
Count: 7  Time=0.01s (0s)  Lock=0.00s (0s)  Rows=177.1 (1240), root[root]@localhost
  select * from t100w as a join t100w as b limit N
Count: 6  Time=0.57s (3s)  Lock=0.00s (0s)  Rows=33.7 (202), root[root]@localhost
  select k1,count(*) from t100w where id<N group by k1 having count(*)>N
Count: 5  Time=0.59s (2s)  Lock=0.00s (0s)  Rows=893.0 (4465), root[root]@localhost
  select k1,count(*) from t100w where num<N group by k1,k2
Count: 5  Time=0.61s (3s)  Lock=0.00s (0s)  Rows=37.0 (185), root[root]@localhost
  select * from t100w where id<N order by num  desc
Count: 4  Time=0.61s (2s)  Lock=0.00s (0s)  Rows=496.0 (1984), root[root]@localhost
  select k1,count(*) from t100w where id<N group by k1,k2
```

5.拓展

```
工具: pt-query-digest
```

第3章 二进制日志binlog

1.作用

```
1  1.数据恢复
2  2.主从复制
```

2.记录的内容

```
1  记录修改类操作(逻辑日志, 类似于SQL记录)
2  DML: insert update delete
3  DDL: create drop alter trucate
4  DCL: grant revoke
```

3.配置方法

3.1 基础参数查看

```
1  mysql> select @@log_bin;
2  mysql> select @@log_bin_basename;
3  mysql> select @@server_id;
```

3.2 设置基础参数

```
1  vim /etc/my.cnf
2  [mysqld]
3  #新增加参数
4  server_id=51                                  #主机ID,在主从复制会使用
5  log_bin=/data/mysql_3306/logs/mysql-bin   #开关+文件路径+文件名前缀,最终格式:
   mysql-bin.000001
```

3.3 重启并查看

```
[root@db-51 ~]# systemctl restart mysqld
[root@db-51 ~]# ll /data/mysql_3306/logs/
总用量 20
-rw-r----- 1 mysql mysql   154 9月  13 17:29 mysql-bin.000001
-rw-r----- 1 mysql mysql    39 9月  13 17:29 mysql-bin.index
-rw-r--r-- 1 mysql mysql 11256 9月  13 17:29 mysql.err
```

4.binlog内容的记录格式

4.1 事件(event)的记录方式

每个事件:

```
1.事件描述: 时间戳、server_id、加密方式、开始的位置（start_pos）、结束位置点（end_pos）
2.事件内容: 修改类的操作: SQL 语句 或者 数据行变化。
```

重点关注:

```
开始的位置（start_pos）
结束位置点（end_pos）
事件内容
```

4.2 二进制日志事件内容格式

查看日志格式:

```
mysql> select @@binlog_format;
+-----------------+
| @@binlog_format |
+-----------------+
| ROW             |
+-----------------+
```

作用:

```
对于DDL、DCL语句，直接将SQL本身记录到binlog中
对于DML ： insert、update、delete 受到binlog_format参数控制。
SBR ： Statement ： 语句模式。之前版本，默认模式
RBR ： ROW       ： 行记录模式。5.7以后，默认模式
MBR ： mixed     ： 混合模式。
```

区别:

```
Statement、ROW区别:
update t1 set name='zhangsan' where id<100;

Statement: 记录SQL本身 。
ROW: 100个数据行的变化。

Statement日志量少
ROW日志量大

Statement记录不够准确
ROW记录够准确。

例如函数操作:
now()
rand()
```

5.binlong查询

5.1 日志文件情况查询

查看所有的日志文件信息

```
mysql> show binary logs;
+------------------+-----------+
| Log_name         | File_size |
+------------------+-----------+
| mysql-bin.000001 |       154 |
+------------------+-----------+
```

刷新新日志

```
mysql> flush logs;
mysql> flush logs;
mysql> flush logs;
mysql> flush logs;
mysql> show binary logs;
+------------------+-----------+
| Log_name         | File_size |
+------------------+-----------+
| mysql-bin.000001 |   2074265 |
| mysql-bin.000002 |      1181 |
| mysql-bin.000003 |       201 |
| mysql-bin.000004 |       201 |
| mysql-bin.000005 |       154 |
+------------------+-----------+
```

当前数据库使用的二进制日志

```
mysql> show master status;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000005 |      154 |              |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
```

5.2 内容查询

数据模拟

```
mysql> create database ku charset utf8mb4;
mysql> use ku
mysql> create table biao (id int);
mysql> insert into biao values(1);
mysql> commit;
```

查看日志事件

```
mysql> show binlog events in 'mysql-bin.000005';
+------------------+-----+----------------+-----------+-------------+-------------------------------------+
| Log_name         | Pos | Event_type     | Server_id | End_log_pos | Info                                |
+------------------+-----+----------------+-----------+-------------+-------------------------------------+
| mysql-bin.000005 |   4 | Format_desc    |         6 |         123 | Server ver: 5.7.28-log, Binlog ver: 4 |
```

```
| mysql-bin.000005 | 123 | Previous_gtids |         6 |         154 |
                                    |
| mysql-bin.000005 | 154 | Anonymous_Gtid |         6 |         219 |
SET @@SESSION.GTID_NEXT= 'ANONYMOUS'  |
| mysql-bin.000005 | 219 | Query          |         6 |         323 |
create database ku charset utf8mb4    |
| mysql-bin.000005 | 323 | Anonymous_Gtid |         6 |         388 |
SET @@SESSION.GTID_NEXT= 'ANONYMOUS'  |
| mysql-bin.000005 | 388 | Query          |         6 |         484 |
use `ku`; create table biao (id int)  |
| mysql-bin.000005 | 484 | Anonymous_Gtid |         6 |         549 |
SET @@SESSION.GTID_NEXT= 'ANONYMOUS'  |
| mysql-bin.000005 | 549 | Query          |         6 |         619 |
BEGIN                                 |
| mysql-bin.000005 | 619 | Table_map      |         6 |         664 |
table_id: 111 (ku.biao)               |
| mysql-bin.000005 | 664 | Write_rows     |         6 |         704 |
table_id: 111 flags: STMT_END_F       |
| mysql-bin.000005 | 704 | Xid            |         6 |         735 |
COMMIT /* xid=88 */                   |
+------------------+-----+----------------+-----------+-------------+--
-----------------------------------+
```

5.3 查看日志内容

查看日志内容

```
[root@db01 logs]# mysqlbinlog /data/mysql_3306/logs/mysql-bin.000005
```

Create database日志内容

```
# at 219
...略
create database ku charset utf8mb4
```

create table 日志内容

```
# at 388
...略
create table biao (id int)
```

insert 操作的日志内容

```
# at 664
#200914  8:15:15 server id 6  end_log_pos 704 CRC32 0x0a91b6f8
Write_rows: table id 111 flags: STMT_END_F

BINLOG '
E7ZeXxMGAAAALQAAAJgCAAAAAG8AAAAAAAEAAmt1AARiaWFvAAEDAAGlD2wp
E7ZeXx4GAAAAKAAAAMACAAAAAG8AAAAAAAEAAgAB//4BAAAA+LaRCg==
'/*!*/;
# at 704
#200914  8:15:15 server id 6  end_log_pos 735 CRC32 0x1e620e90  Xid =
88
COMMIT/*!*/;
```

查看解密后的insert

```
[root@db-51 ~]# mysqlbinlog --base64-output=decode-rows -vv
/data/mysql_3306/logs/mysql-bin.000005
.......略
# at 664
#200914  8:15:15 server id 6  end_log_pos 704 CRC32 0x0a91b6f8
Write_rows: table id 111 flags: STMT_END_F
### INSERT INTO `ku`.`biao`
### SET
###   @1=1 /* INT meta=0 nullable=1 is_null=0 */
```

6.binlog日志截取及恢复演练

6.1 前提说明

```
创建或导入数据库之前就配置并开启了binlog
```

6.2 故障说明

```
模拟误删库，要求恢复到删库之前
```

6.3 模拟故障

```
1.创建库
create database linux5;

2.创建表
use linux5;
CREATE TABLE `user` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT 'id',
  `name` varchar(10) NOT NULL COMMENT 'name',
  `age` tinyint(4) NOT NULL COMMENT 'age',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

3.写入数据
insert into user(name,age)
values
('z3',22),
('l4',22),
('w5',22);

4.查看数据
mysql> select * from user;
+----+------+-----+
| id | name | age |
```

```
+----+------+-----+
|  1 | z3   |  18 |
|  2 | l4   |  20 |
|  3 | w5   |  21 |
+----+------+-----+

5.模拟删除
drop database linux5;
```

6.4 恢复思路

第一步：截取从建库以来到删库之前的所有binlog

查看当前处于什么哪个binlog：

```
mysql> show master status;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000002 |     1134 |              |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
```

第二步: 找到起点,建库的位置点（position）

```
mysql> show binlog events in 'mysql-bin.000002';
....略
| mysql-bin.000002 |  219 | Query          |         6 |         319 | create database linux5
```

第三步: 找到终点

```
.....略
| mysql-bin.000002 | 1036 | Query          |         6 |        1134 |
drop database linux5
```

导出数据:

```
mysqlbinlog --start-position=219 --stop-position=1036
/data/mysql_3306/logs/mysql-bin.000002 >/tmp/bin.sql
```

将截取的日志进行回放

```
mysql> set sql_log_bin=0;
mysql> source /tmp/bin.sql;
mysql> set sql_log_bin=1;
```

7.生产中日志在多个文件中,如何截取?

7.1 场景模拟

```
flush logs;

#mysql-bin.000005
show master status ;
create database tongdian charset=utf8mb4;
use tongdian
create table t1 (id int);
flush logs;

#mysql-bin.000006
show master status ;
insert into t1 values(1),(2),(3);
commit;
flush logs;

```

```
#mysql-bin.000007
show master status ;
create table t2(id int);
insert into t2 values(1),(2),(3);
commit;
flush logs;

#mysql-bin.000008
show master status ;
insert into t2 values(11),(22),(33);
commit;
drop database tongdian;
```

7.2 恢复方法

方法1:分段截取

```
--start-position    --stop-position
```

方法2:时间戳截取

a.找起点: 建库的时间戳

起点posting号

```
show binlog events in 'mysql-bin.000005';
```

通过position过滤时间戳

```
mysqlbinlog --start-position=951  --stop-position=1073 mysql-bin.000005
|grep -A 1 '^\#\ at\ 951'
```

b.找终点

```
mysql -e "show binlog events in  'mysql-bin.000008'"
```

c.截取日志

```
mysqlbinlog  --start-datetime="2020-05-09 17:11:23"  --stop-datetime="2020-05-09 17:14:01"   mysql-bin.000005 mysql-bin.000006 mysql-bin.000007 mysql-bin.000008 >/tmp/data.sql
```

7.3 binlog其他注意

binlog属于全局日志，日志中有其他库的操作，怎么排除掉?

```
mysqlbinlog -d oldboy mysql-bin.000008 > /tmp/bin.sql
```

binlog中100w个事件，怎么快速找到drop database的位置点?

```
[root@db01 ~]# mysql -e "show binlog events in 'mysql-bin.000014'" |less
[root@db01 ~]# mysql -e "show binlog events in 'mysql-bin.000014'" |grep
```

比如删除的库，建库是在2年前操作的。这种情况怎么办?

```
每天全备，binlog完好的。
可以使用 全备+binlog方式实现恢复数据故障之前。
```

8.基于GTID的binlog应用

8.1 GTID全局事务ID

```
对每个事务，进行单独编号。连续不断进行增长。
```

8.2 表示方式

```
server_uuid:N
```

8.3 GTID配置

查看参数:

```
mysql> show variables like '%GTID%';
+---------------------------------+-----------+
| Variable_name                   | Value     |
+---------------------------------+-----------+
| binlog_gtid_simple_recovery     | ON        |
| enforce_gtid_consistency        | OFF       |
| gtid_executed_compression_period | 1000      |
| gtid_mode                       | OFF       |
| gtid_next                       | AUTOMATIC |
| gtid_owned                      |           |
| gtid_purged                     |           |
| session_track_gtids             | OFF       |
+---------------------------------+-----------+
```

设置参数:

```
vim /etc/my.cnf
[mysqld]
gtid_mode=ON                    #开关
enforce_gtid_consistency=ON     #强制GTID一致性
log_slave_updates=ON            #强制从库更新binlog
```

重启服务:

```
systemctl restart mysqld
```

建议:

```
5.7版本以后，都开启GTID。最好是搭建环境就开启。
```

8.4 GTID应用

模拟环境:

a.创建库并查看gtid

```
mysql> create database gtdb charset utf8mb4;
mysql> show master status ;
+------------------+----------+--------------+------------------+-------
---------------------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
Executed_Gtid_Set                |
+------------------+----------+--------------+------------------+-------
---------------------------------+
| mysql-bin.000006 |      329 |              |                  |
9b52b744-eb82-11ea-986c-000c294983f8:1 |
+------------------+----------+--------------+------------------+-------
---------------------------------+
```

b.创建表并查看gtid

```
mysql> use gtdb;
mysql> create table t1(id int);
mysql> show master status ;
+------------------+----------+--------------+------------------+-------
-----------------------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
Executed_Gtid_Set                        |
+------------------+----------+--------------+------------------+-------
-----------------------------------+
| mysql-bin.000006 |      491 |              |                  |
9b52b744-eb82-11ea-986c-000c294983f8:1-2 |
+------------------+----------+--------------+------------------+-------
-----------------------------------+
```

c.插入数据并查看

```
begin;
insert into t1 values(1);
insert into t1 values(2);
insert into t1 values(3);
commit;
show master status ;
+------------------+----------+--------------+------------------+------
-----------------------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
Executed_Gtid_Set                        |
+------------------+----------+--------------+------------------+------
-----------------------------------+
| mysql-bin.000006 |      914 |              |                  |
9b52b744-eb82-11ea-986c-000c294983f8:1-3 |
+------------------+----------+--------------+------------------+------
-----------------------------------+
```

d.查看事件

```
mysql> show binlog events in 'mysql-bin.000006';
+------------------+-----+---------------+-----------+-------------+-----------------------------------------------------------------------+
| Log_name         | Pos | Event_type    | Server_id | End_log_pos | Info                                                                  |
+------------------+-----+---------------+-----------+-------------+-----------------------------------------------------------------------+
| mysql-bin.000006 |   4 | Format_desc   |         6 |         123 | Server ver: 5.7.28-log, Binlog ver: 4                                 |
| mysql-bin.000006 | 123 | Previous_gtids |         6 |         154 |                                                                       |
| mysql-bin.000006 | 154 | Gtid          |         6 |         219 | SET @@SESSION.GTID_NEXT= '9b52b744-eb82-11ea-986c-000c294983f8:1'     |
| mysql-bin.000006 | 219 | Query         |         6 |         329 | create database gtdb charset utf8mb4                                  |
| mysql-bin.000006 | 329 | Gtid          |         6 |         394 | SET @@SESSION.GTID_NEXT= '9b52b744-eb82-11ea-986c-000c294983f8:2'     |
| mysql-bin.000006 | 394 | Query         |         6 |         491 | use `gtdb`; create table t1(id int)                                   |
| mysql-bin.000006 | 491 | Gtid          |         6 |         556 | SET @@SESSION.GTID_NEXT= '9b52b744-eb82-11ea-986c-000c294983f8:3'     |
| mysql-bin.000006 | 556 | Query         |         6 |         628 | BEGIN                                                                 |
| mysql-bin.000006 | 628 | Table_map     |         6 |         673 | table_id: 108 (gtdb.t1)                                               |
| mysql-bin.000006 | 673 | Write_rows    |         6 |         713 | table_id: 108 flags: STMT_END_F                                       |
| mysql-bin.000006 | 713 | Table_map     |         6 |         758 | table_id: 108 (gtdb.t1)                                               |
| mysql-bin.000006 | 758 | Write_rows    |         6 |         798 | table_id: 108 flags: STMT_END_F                                       |
| mysql-bin.000006 | 798 | Table_map     |         6 |         843 | table_id: 108 (gtdb.t1)                                               |
| mysql-bin.000006 | 843 | Write_rows    |         6 |         883 | table_id: 108 flags: STMT_END_F                                       |
```

```
19  | mysql-bin.000006 | 883 | Xid            |         6 |         914 |
    COMMIT /* xid=12 */                                                  |
20  +------------------+-----+----------------+-----------+-------------+--
    ------------------------------------------------------------------+
```

8.5 通过GTID方式截取日志

错误的截取命令:

```
1  mysqlbinlog --include-gtids='9b52b744-eb82-11ea-986c-000c294983f8:1-3'
   /data/mysql_3306/logs/mysql-bin.000006 >/tmp/gt.sql
```

为什么恢复报错?

```
1  gtid有“幂等性”检查。GTID的生成，通过Set gtid_next命令实现的。
2  例如:
3  SET @@SESSION.GTID_NEXT= '202628e9-9265-11ea-b4a0-000c29248f69:1'
4  执行Set命令时，自动检查当前系统是否包含这个GTID信息，如果有就跳过。
```

正确的方式:

```
1  mysqlbinlog --skip-gtids --include-gtids='2ddd7a11-4747-11eb-b274-
   000c29116b18:1-3' /data/mysql_3306/binlog/mysql-bin.000009 >/tmp/gt.sql
```

8.6 恢复操作

```
1  set sql_log_bin=0;
2  source /tmp/gt.sql;
3  set sql_log_bin=1;
```

9.日志滚动

命令触发：

```
mysql> flush logs;
shell# mysqladmin flush-logs
shell# mysql -e "flush logs"
shell# mysqldump -F
```

自动触发：

```
mysql> select @@max_binlog_size;
+-------------------+
| @@max_binlog_size |
+-------------------+
|        1073741824 |
+-------------------+

重启数据库，会触发刷新
```

10.日志删除

10.1 默认方式

```
不自动清理。直到空间写满。
```

10.2 配置自动清理

```
mysql> select @@expire_logs_days;
最少设置多少天合适?
参考全备时间周期。
例如:  全备周期是7天。可以保留8天。一般生产中保留两轮备份周期的日志，15天。

设置命令:
set GLOBAL expire_logs_days=7;
```

10.3 手工清理

```
Examples:
PURGE BINARY LOGS TO 'mysql-bin.010';
PURGE BINARY LOGS BEFORE '2019-04-02 22:46:26';
```