Mysql gtid复制

# 一、GTID的概述：

1、全局事物标识：global transaction identifieds。

2、GTID事物是全局唯一性的，且一个事务对应一个GTID。

3、一个GTID在一个服务器上只执行一次，避免重复执行导致数据混乱或者主从不一致。

4、GTID用来代替classic的复制方法，不在使用binlog+pos开启复制。而是使用master\_auto\_postion=1的方式自动匹配GTID断点进行复制。

5、[**MySQL**](http://lib.csdn.net/base/mysql)-5.6.5开始支持的，MySQL-5.6.10后开始完善。

6、在传统的slave端，binlog是不用开启的，但是在GTID中，slave端的binlog是必须开启的，目的是记录执行过的GTID（强制）。

# 二、GTID的组成部分：

前面是server\_uuid：后面是一个序列号

例如：server\_uuid：sequence number

7800a22c-95ae-11e4-983d-080027de205a:10

UUID：每个mysql实例的唯一ID，由于会传递到slave，所以也可以理解为源ID。

Sequence number：在每台MySQL服务器上都是从1开始自增长的序列，一个数值对应一个事务。

# 三、GTID比传统复制的优势：

1、更简单的实现failover，不用以前那样在需要找log\_file和log\_Pos。

2、更简单的搭建主从复制。

3、比传统复制更加安全。

4、GTID是连续没有空洞的，因此主从库出现数据冲突时，可以用添加空事物的方式进行跳过。

# 四、GTID的工作原理：

1、master更新数据时，会在事务前产生GTID，一同记录到binlog日志中。

2、slave端的i/o 线程将变更的binlog，写入到本地的relay log中。

3、sql线程从relay log中获取GTID，然后对比slave端的binlog是否有记录。

4、如果有记录，说明该GTID的事务已经执行，slave会忽略。

5、如果没有记录，slave就会从relay log中执行该GTID的事务，并记录到binlog。

6、在解析过程中会判断是否有主键，如果没有就用二级索引，如果没有就用全部扫描。

要点：

1、slave在接受master的binlog时，会校验master的GTID是否已经执行过（一个服务器只能执行一次）。

2、为了保证主从数据的一致性，多线程只能同时执行一个GTID。

# 五、使用GTID搭建mysql的主从复制的主要参数：

[mysqld]

#GTID:

gtid\_mode=on

enforce\_gtid\_consistency=on

server\_id=2003306    #每天实例的server\_id都要不一样

#binlog

log-bin=mysqlbin

log-slave-updates=1   #允许下端接入slave

binlog\_format=row      #强烈建议，其他格式可能造成数据不一致

#relay log

skip\_slave\_start=1

注意：建议使用mysql-5.6.5以上的最新版本。

（二）、启动GTID的两种方法：

方法一、

1、如果是在已经跑的服务器，你需要重启一下mysql server。

2、启动之前，一定要先关闭master的写入，保证所有slave端都已经和master端数据保持同步。

3、所有slave需要加上skip\_slave\_start=1的配置参数，避免启动后还是使用老的复制协议。

方法二、

1、如果是新搭建的服务器，直接启动就行了。

# 六、master-slave搭建的注意事项：

（一）、使用GTID的方式，把salve端挂载master端：

1、启动以后最好不要立即执行事务，而是先change master上。

2、然后在执行事务，当然知不是必须的。

3、使用下面的sql切换slave到新的master。

stop slave;

change master to

master\_host = 192.168.100.200,

master\_port = 3306,

master\_user = abobo,

master\_password=123,

master\_auto\_position = 1;

(二)、如果给已经运行的GTID的master端添加一个新的slave

 有两种方法：

方法一、适用于master也是新建不久的情况。

1、如果你的master所有的binlog还在。可以选择类似于上面的方法，安装slave，直接change master to到master端。

2、原理是直接获取master所有的GTID并执行。

3、优点：简单方便。

4、缺点：如果binlog太多，数据完全同步需要时间较长，并且master一开始就启用了GTUD。

方法二、适用于拥有较[**大数据**](http://lib.csdn.net/base/spark)的情况。（推荐）

1、通过master或者其他slave的备份搭建新的slave。（看第三部分）

2、原理：获取master的数据和这些数据对应的GTID范围，然后通过slave设置@@global.gtid\_purged跳过备份包含的gtid。

3、优点：是可以避免第一种方法的不足。

4、缺点：相对来说有点复杂。

（三）、通过备份搭建新的slave：（方法二的扩展）

两种方法：

方法一、mysqldump的方式：

1、在备份的时候指定--master-data=2（来保存binlog的文件号和位置的命令）。

2、使用mysqldump的命令在dump文件里可以看到下面两个信息：

SET @@SESSION.SQL\_LOG\_BIN=0;

SET @@GLOBAL.GTID\_PURGED='7800a22c-95ae-11e4-983d-080027de205a:1-8';

3、将备份还原到slave后，使用change master to命令挂载master端。

注意：在mysql5.6.9以后的命令才支持这个功能。

方法二、percona Xtrabackup

1、Xtrabackup\_binlog\_info文件中，包含global.gtid\_purged='XXXXXX:XXXX'的信息。

2、然后到slave去手工的 SET GLOBAL.GTID\_PURGED='XXXXXX:XXXX'。

3、恢复备份，开启change master to 命令。

注意：如果系统运行了很久，无法找到GTID的变好了，可以通过上面的方式进行查找。

# 七、GTID如何跳过事务冲突：

1、这个功能主要跳过事务，代替原来的set global sql\_slave\_skip\_counter = 1。

2、由于在这个GTID必须是连续的，正常情况同一个服务器产生的GTID是不会存在空缺的。所以不能简单的skip掉一个事务，只能通过注入空事物的方法替换掉一个实际操作事务。

3、注入空事物的方法：

stop slave;

set gtid\_next='xxxxxxx:N';

begin;commit;

set gtid\_next='AUTOMAIC';

start slave;

4、这里的xxxxx:N 也就是你的slave sql thread报错的GTID，或者说是你想要跳过的GTID。

# 八、GTID的参数注释：

[master]>show global variables like '%gtid%';

1、enforce\_gtid\_consistency：开启gtid的一些安全限制（介意开启）。

2、gtid\_executed：全局和seeeion级别都可以用。用来保存已经执行过的GTIDs。

贴士：show  master status\G;输出结果中的Executed\_Gtid\_Set和gitd\_executed一致。reset master时，此值会被清空。

3、gtid\_owned：全局和session级别都可用，全局表示所有服务器拥有GTIDs，session级别表示当前client拥有所有GTIDs。（此功能用的少）

4、gtid\_mode：是否开启GTID功能。

5、gtid\_purged：全局参数，设置在binlog中，已经purged的GTIDs，并且purged掉的GTIDs会包含到gtid\_executed中。

贴士：从而导致slave不会再去master请求这些GTIDs，并且Executed\_Gtid\_Set为空时，才可以设置此值。

6、gtid\_next：这个时session级别的参数：

[master]>show session variables like '%gtid\_next%';

# 九、关于GTID的一些功能限制：

（一）、更新非事务引擎：

1、Case重现：

master：对一个innodb表做一个多sql更新的事物，效果是产生一个GTID。

slave：对应的表是MYISAM引擎，执行这个GTID的第一个语句后就会报错，因为非事务引擎一个sql就是一个事务。

2、错误编号：

last\_Errno:1756

3、异常恢复方案：

（1）、简单的stop slave; start slave;就能够忽略错误。但是这个时候主从的一致性已经出现问题。需要手工的把slave差的数据补上。

（2）、首先将引擎调整为一样的，slave也改为事务引擎。

（二）、create table ....select statements

1、case重现：

 master：直接执行一个create table select \* from table;的sql

2、报错：

error 1786

3、原理：

由于create table ...select语句会生成两个sql，一个是DDL创建表SQL，一个是insert into 插入数据的sql。由于DDL会导致自动提交，所以这个sql至少需要两个GTID，但是GTID模式下，只能给这个sql生成一个GTID，如果强制执行会导致和上面更新非事务引擎一样的结果。

（三）、一个sql同事操作innodb引擎和myisam引擎：

case重现：t1表是innodb，t2表是myisam

1、update t1,t2 set t1.id=1000,t2.id=1000 where t1.id=t2.id;

2、报错：1785

3、原理和第二个相同。

（四）、在一个replication grouop 中，所有的mysql必须要统一开启或者关闭GTID功能。

1、case重现：

将一个未开启gtid的slave通过原始的binlog和pos方式连接到开启GTID的master。

2、报错：

The slave IO thread stops because the master has @@GLOBAL.GTID\_MODE ON and this server has @@GLOBAL.GTID\_MODE OFF。

（五）、在一个replication group中，如果开启GTID以后，就不再允许使用classic的复制方式：

1、case重现：

将一个开启gtid的slave通过原始的binlog和pos方式连接到开启GTID的master。

2、报错：

ERROR 1776(HY000)：Parameters MASTER\_LOG\_FILE，MASTER\_LOG\_POS，RELAY\_LOG\_FILE and RELAY\_LOG\_POS cannot be set when MASTER\_AUTO\_POSITION is active。

（六）、GTID\_MODE是not online的：

需要重启才能生效，官方暂时不支持平滑的从classic replication切换到GTID replication。

贴士：

由于GTID开启需要重启系统，一个复制组中所有的实例必须统一开启或者关闭GTID,开启GTID以后不能在使用classic复制。

问题：

也就是说在线业务必须统一关闭，然后再启动，会导致服务中断。

解决方案：

1、针对这种情况，社区有两种对应的平滑升级的方案：

一种是booking.com出品，这两个差别在淘宝9月份数据库月报里有说明，加了一个桥接的服务器，既可以运行GTID模式下，也可以运行classic模式下。

另外一种是facebook.com出品。所有的slave可以在开启GTID模式的情况下，可以连接到没有开启GTID模式的master。

2、可以关闭一个部分，停止写操作，但是读不用，将另一部分改成GTID模式。

（七）、Temporary tables。

1、create temporary table和drop temporary table语句一样在GTID环境下不支持。

如果--enforce\_gtid\_consistency参数开启，并且autocommit=1，那么可以使用。

 （八）、关于Errant transaction

1、Errant transaction：所谓的errant transaction也就是没有规范的从master执行，而是直接从slave执行的事务。

2、由于GTID协议的原因，最开始已经提过（参见GTID architecture）。

3、如果slave有errant transaction产生，由于GTID协议中的规则，很容易导致failover失败。主要有两种情况：

a、在slave上做了无用的或者临时的errant transaction操作，如果该slave升级成为master的话，连接到它的所有数据库都会获取到这个事务。如果一样就会产生冲突。

b、由于做了这个errant transaction这个事务以后，其他的slave还没有获取这个errant transaction的GTID，需要从master上发同步给其他的slave，但是主的binlog又被删掉了，这时将会报错。

4、总之：尽量避免产生errant transaction。可以通过：set sql\_log\_bin=off的方式在slave执行sql，但是也要考虑到数据一致性。