



豆包

抖音旗下AI工具

你的全能AI助手，助力每日工作学习

立即体验



ZhenXing_Yu

邮箱地址：zhenxing1004@hotmail.com | 做你害怕做的事。然后你会发现，不过如此

博客园

首页

联系

管理

MySQL 在线开启GTID的每个阶段是要做什么

MySQL 在线开启GTID的每个阶段是要做什么

目录

- MySQL 在线开启GTID的每个阶段是要做什么
 - 基本概述
 - 在线开启GTID
 - 1. 设置GTID校验ENFORCE GTID CONSISTENCY为WARN
 - 2. 设置GTID校验ENFORCE GTID CONSISTENCY为ON
 - 3. 设置GTID MODE为OFF PERMISSIVE
 - 4. 设置GTID MODE为ON PERMISSIVE
 - 5. (关键点)确保匿名事务回放完毕
 - 6. 触发一轮日志切换FLUSH LOGS
 - 7. 正式开启GTID MODE为ON
 - 8. 修改配置文件确保GTID参数持久化
 - 9. 修改复制模式为GTID方式
 - 在线关闭GTID
 - 1. 将复制改为基于POS点方式

- 2. 设置GTID MODE为ON PERMISSIVE
- 3. 设置GTID MODE为OFF PERMISSIVE
- 4. (关键点)确保GTID事务回放完毕
- 5. 触发FLUSH LOGS
- 6. 设置GTID MODE为OFF
- 7. 设置ENFORCE GTID CONSISTENCY为OFF
- 8. 修改my.cnf配置文件中GTID相关参数为OFF
- 命令简版
 - 1. 在线开启GTID
 - 2. 在线关闭GTID
- 技术总结
- 参考链接

基本概述

我们知道MySQL有2种方式指定复制同步的方式,分别为:

1. 基于binlog文件名及位点的指定方式
 - 匿名事务(Anonymous_gtid_log_event)
2. 基于GTID(全局事务ID)的指定方式
 - GTID事务(Gtid_log_event)

而基于GTID的方式在一主多从的架构下主从切换有着明显优势外,对于日常复制异常的故障诊断也更为方便,从个人角度而言,我也更倾向于大家做在线开启或关闭**GTID**的操作,一方面该操作能尽可能小的影响数据库停机时间,另一方面在开启或关闭的过程中也顺便可以验证该参数的调整是否会对应用造成影响,从MySQL 5.7.6之后便开始支持动态开启和关闭GTID模式,其参数GTID_MODE有以下取值

- OFF - 只允许匿名事务被复制同步
- OFF_PERMISSIVE - 新产生的事务都是匿名事务,但也允许有GTID事务被复制同步
- ON_PERMISSIVE - 新产生的都是GTID事务,但也允许有匿名事务被复制同步

- ON - 只允许GTID事务被复制同步

其实从该参数的几个取值我们就能看出,在线修改是循序渐进的将匿名事务转化为GTID事务过程(反之也一样),我们先看看在线开启GTID分别要做哪些事

在线开启GTID

1. 设置GTID校验ENFORCE_GTID_CONSISTENCY为WARN

该操作的目的是允许在主库执行的SQL语句违反GTID一致性校验,且只在主库的错误日志中输出warning级别日志以作提醒,我们知道GTID复制还是有一些限制条件的,其实这里就是为了考虑如果复制方式直接改为GTID模式后应用程序因为GTID的一些限制导致异常报错,这样做的好处是当我需要开启GTID前,我可以把ENFORCE_GTID_CONSISTENCY参数开成WARN观测一段时间,比如一天,如果观测周期内在错误日志中并没有发现相关的Warning信息,那我们再考虑正式开启GTID的操作

- 示例:使用 `CREATE TABLE AS SELECT` 语法在GTID模式下不支持(题外话:CTAS语法在8.0.21以后GTID模式下也是支持的,该语法变更为了一个特殊的原子性DDL操作),而ENFORCE_GTID_CONSISTENCY设置为WARN时,只会在错误日志提示,不会直接报错,

```
mysql> create table sbtest5 as select * from sbtest1;
Query OK, 100000 rows affected, 1 warning (1.28 sec)
Record: 100000 Duplicates: 0 Warnings: 1

mysql> show warnings;
+-----+-----+-----+
| Level | Code | Message                                     |
+-----+-----+-----+
| Warning | 1786 | Statement violates GTID consistency: CREATE TABLE ... SELECT. |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

```
10.186.63.65
[root@10-186-63-65 ~]# tail -f /data/zhenxing/3310/data/mysql-error.log
2021-11-25T22:14:14.760456+08:00 0 [Note] Server socket created on IP: '::'.
2021-11-25T22:14:14.853588+08:00 0 [Note] Event Scheduler: Loaded 0 events
2021-11-25T22:14:14.853874+08:00 0 [Note] /data/zhenxing/3310/base/bin/mysqld: ready for connections.
Version: '5.7.31-log' socket: '/data/zhenxing/3310/data/mysqld.sock' port: 3310 MySQL Community Server (GPL)
2021-11-25T22:14:14.863212+08:00 0 [Note] InnoDB: Buffer pool(s) load completed at 211125 22:14:14
2021-11-25T22:15:09.153529+08:00 2 [Warning] Timeout waiting for reply of binlog (file: mysql-bin.000003, pos: 426), semi-sync up to file , position 0.
2021-11-25T22:15:09.153645+08:00 2 [Note] Semi-sync replication switched OFF.
2021-11-25T22:16:03.041194+08:00 3 [Note] Start binlog_dump to master_thread_id(3) slave_server(63663310), pos(mysql-bin.000001, 4)
2021-11-25T22:16:03.041329+08:00 3 [Note] Start semi-sync binlog_dump to slave (server_id: 63663310), pos(mysql-bin.000001, 4)
2021-11-25T22:16:03.054279+08:00 0 [Note] Semi-sync replication switched ON at (mysql-bin.000003, 426)

2021-11-25T22:51:49.741877+08:00 2 [Note] Changed ENFORCE_GTID_CONSISTENCY from OFF to WARN.
^@2021-11-25T22:53:10.290969+08:00 2 [Warning] Statement violates GTID consistency: CREATE TABLE ... SELECT.
^@^@^@^@^@
```

该操作在主从库均执行

```
SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = WARN;
```

2. 设置GTID校验ENFORCE_GTID_CONSISTENCY为ON

确认上一个步骤未在错误日志中出现相关Warning信息后,正式开启GTID一致性校验,当设置为ON后,如果再执行CREATE TABLE AS SELECT语句则会直接报错

```
mysql> create table sbtest6 as select * from sbtest1;
ERROR 1786 (HY000): Statement violates GTID consistency: CREATE TABLE ... SELECT.
mysql> █
```

该操作在主从库均执行
SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = ON;

3. 设置GTID_MODE为OFF_PERMISSIVE

如前面的值描述,该操作表示新产生的事务依旧是匿名事务,但也允许有GTID事务被复制同步,对于在线开启GTID模式而言,该步骤就是一个单纯的过渡属性(注意是为在线关闭GTID准备的),执行完后可快速到下一个阶段

该操作在主从库均执行
SET @@GLOBAL.GTID_MODE = OFF_PERMISSIVE;

4. 设置GTID_MODE为ON_PERMISSIVE

该操作依旧是一个过渡属性,其表示的则是新产生的都是GTID事务,但也允许有匿名事务被复制,从这个阶段开始就已经是一个正式转化的过程,但依旧是对两种事务做兼容.

该操作在主从库均执行
SET @@GLOBAL.GTID_MODE = ON_PERMISSIVE;

5. (关键点)确保匿名事务回放完毕

该步骤的目的是确保在正式转换为完整的GTID模式前,老的匿名事务均以被回放完毕,确保GTID_MODE设置为ON时,不会因为残留的匿名事务导致复制同步报错,有以下2种方式进行校验

该操作仅在从库执行即可
方式1: 确保该状态值输出的匿名事务数显示为0 (注意: 只要出现过0即可表示已经转换完成, 即使后续该状态值从0变为1)
SHOW STATUS LIKE 'ONGOING_ANONYMOUS_TRANSACTION_COUNT';

在从库上多次执行该语句
方式2: 查询该视图中LAST_SEEN_TRANSACTION可以观测当前同步的事务是否还存在ANONYMOUS事务
select * from performance_schema.replication_applier_status_by_worker;

确保匿名事务数为0

```
mysql> SHOW STATUS LIKE 'ONGOING_ANONYMOUS_TRANSACTION_COUNT';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Ongoing_anonymous_transaction_count | 0 |
+-----+-----+
1 row in set (0.01 sec)
```

确保回放线程回放的事务都已是GTID事务

```
mysql> select * from performance_schema.replication_applier_status_by_worker;
+-----+-----+-----+-----+-----+-----+-----+-----+
| CHANNEL_NAME | WORKER_ID | THREAD_ID | SERVICE_STATE | LAST_SEEN_TRANSACTION | LAST_ERROR_NUMBER | LAST_ERROR_MESSAGE | LAST_ERROR_TIMESTAMP |
+-----+-----+-----+-----+-----+-----+-----+-----+
|              | 1 | 41 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565457 | 0 |  | 0000-00-00 00:00:00 |
|              | 2 | 42 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565458 | 0 |  | 0000-00-00 00:00:00 |
|              | 3 | 43 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565454 | 0 |  | 0000-00-00 00:00:00 |
|              | 4 | 44 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565444 | 0 |  | 0000-00-00 00:00:00 |
|              | 5 | 45 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565445 | 0 |  | 0000-00-00 00:00:00 |
|              | 6 | 46 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565446 | 0 |  | 0000-00-00 00:00:00 |
|              | 7 | 47 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565395 | 0 |  | 0000-00-00 00:00:00 |
|              | 8 | 48 | ON | 99a9940c-4df9-11ec-8171-02000aba3f41:565396 | 0 |  | 0000-00-00 00:00:00 |
+-----+-----+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

6. 触发一轮日志切换FLUSH LOGS

该操作的目的是为了在主库触发binlog的轮换,使新生成的binlog都是包含GTID的事务(防止一个binlog中包含2种类型的事务日志)

```
## 该操作仅在主库执行即可
FLUSH LOGS;
```

7. 正式开启GTID_MODE为ON

正式开启GTID

```
## 该操作在主从库均执行
SET @@GLOBAL.GTID_MODE = ON;
SELECT @@GTID_MODE,@@ENFORCE_GTID_CONSISTENCY;
```

8. 修改配置文件确保GTID参数持久化

在my.cnf配置文件中增加GTID参数,确保重启不会失效,该操作也可在第一步进行

```
## 该操作在主从库均执行
gtid-mode = ON
enforce-gtid-consistency = 1
```

9. 修改复制模式为GTID方式

在开启GTID模式后我们也要将复制模式从基于POS点改为基于GTID,操作比较简单

```
## 停止复制
STOP SLAVE;

## 修改为GTID模式
CHANGE MASTER TO MASTER_AUTO_POSITION = 1;

## 开启复制
START SLAVE;

## 观测复制同步状态
SHOW SLAVE STATUS\G
```

在线关闭GTID

在线关闭的方式基本就类似于在线开启GTID的逆向操作,以下只写出步骤和具体命令,不做详细解释

1. 先将GTID模式的复制改为基于POS点的复制
2. 设置GTID_MODE为ON_PERMISSIVE
3. 设置GTID_MODE为OFF_PERMISSIVE
4. 观测GTID_OWNED状态变量变为空值及replication_applier_status_by_worker表中事务均转为匿名事务
5. 触发FLUSH LOGS
6. 设置GTID_MODE为OFF
7. 设置ENFORCE_GTID_CONSISTENCY为OFF
8. 修改my.cnf配置文件中GTID相关参数为OFF

1. 将复制改为基于POS点方式

```
stop slave;
show slave status\G

## 取show slave status\G中的Relay_Master_Log_File和Exec_Master_Log_Pos填入
## 这里一定不要漏掉MASTER_AUTO_POSITION = 0这个配置
CHANGE MASTER TO
  MASTER_AUTO_POSITION = 0,
  MASTER_LOG_FILE='mysql-bin.000017',
  MASTER_LOG_POS=224126137;
start slave;
show slave status\G
```

2. 设置GTID_MODE为ON_PERMISSIVE

```
## 该操作在主从库均执行
SET @@GLOBAL.GTID_MODE = ON_PERMISSIVE;
```

3. 设置GTID_MODE为OFF_PERMISSIVE

```
## 该操作在主从库均执行
SET @@GLOBAL.GTID_MODE = OFF_PERMISSIVE;
```

4. (关键点)确保GTID事务回放完毕

观测GTID_OWNED状态变量变为空值及replication_applier_status_by_worker表中事务均转为匿名事务

```
## 该操作在从库执行即可
SELECT @@GLOBAL.GTID_OWNED;
select * from performance_schema.replication_applier_status_by_worker;
```

5. 触发FLUSH LOGS

```
## 该操作在主库执行即可
FLUSH LOGS;
```

6. 设置GTID_MODE为OFF

```
## 该操作在主从库均执行
SET @@GLOBAL.GTID_MODE = OFF;
```

7. 设置ENFORCE_GTID_CONSISTENCY为OFF

```
## 该操作在主从库均执行
SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = OFF;
```

8. 修改my.cnf配置文件中GTID相关参数为OFF

```
## 该操作在主从库均执行
gtid-mode = OFF
enforce-gtid-consistency = 1
```

命令简版

1. 在线开启GTID

自行判断命令在主库还是从库执行

```
SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = WARN;
SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = ON;
SET @@GLOBAL.GTID_MODE = OFF_PERMISSIVE;
SET @@GLOBAL.GTID_MODE = ON_PERMISSIVE;
SHOW STATUS LIKE 'ONGOING_ANONYMOUS_TRANSACTION_COUNT';
select * from performance_schema.replication_applier_status_by_worker;
FLUSH LOGS;
SET @@GLOBAL.GTID_MODE = ON;

## 配置文件修改
gtid-mode = ON
enforce-gtid-consistency = 1

## 将复制模式从基于POS点改为基于GTID
STOP SLAVE;
CHANGE MASTER TO MASTER_AUTO_POSITION = 1;
START SLAVE;
SHOW SLAVE STATUS\G
```

2. 在线关闭GTID

自行判断命令在主库还是从库执行

```
stop slave;
show slave status\G

## 取show slave status\G中的Master_Log_File和Exec_Master_Log_Pos填入
CHANGE MASTER TO
  MASTER_AUTO_POSITION = 0,
  MASTER_LOG_FILE='mysql-bin.000017',
  MASTER_LOG_POS=224126137;
start slave;
show slave status\G

SET @@GLOBAL.GTID_MODE = ON_PERMISSIVE;
SET @@GLOBAL.GTID_MODE = OFF_PERMISSIVE;
SELECT @@GLOBAL.GTID_OWNED;
select * from performance_schema.replication_applier_status_by_worker;
FLUSH LOGS;
SET @@GLOBAL.GTID_MODE = OFF;
SET @@GLOBAL.ENFORCE_GTID_CONSISTENCY = OFF;

## 修改my.cnf配置文件中GTID相关参数为OFF
gtid-mode = OFF
enforce-gtid-consistency = 1
```


技术总结

其实在线开启和在线关闭GTID看上去命令较多,但实际基本都是很快的速度就能完成且对业务几乎没有影响,其中更重要的反而是在正式开启之前的一个校验过程.

参考链接

- <https://dev.mysql.com/doc/refman/5.7/en/replication-mode-change-online-enable-gtids.html>
- <https://dev.mysql.com/doc/refman/5.7/en/replication-mode-change-online-disable-gtids.html>

转载请说明出处 |QQ:327488733@qq.com

分类: [MySQL Fundamental](#)

标签: [mysql](#), [GTID](#), [在线变更GTID](#)

好文要顶

关注我

收藏该文

微信分享



ZhenXing_Yu

粉丝 - 20 关注 - 5

+加关注

0

推荐

0


反对

升级成为会员

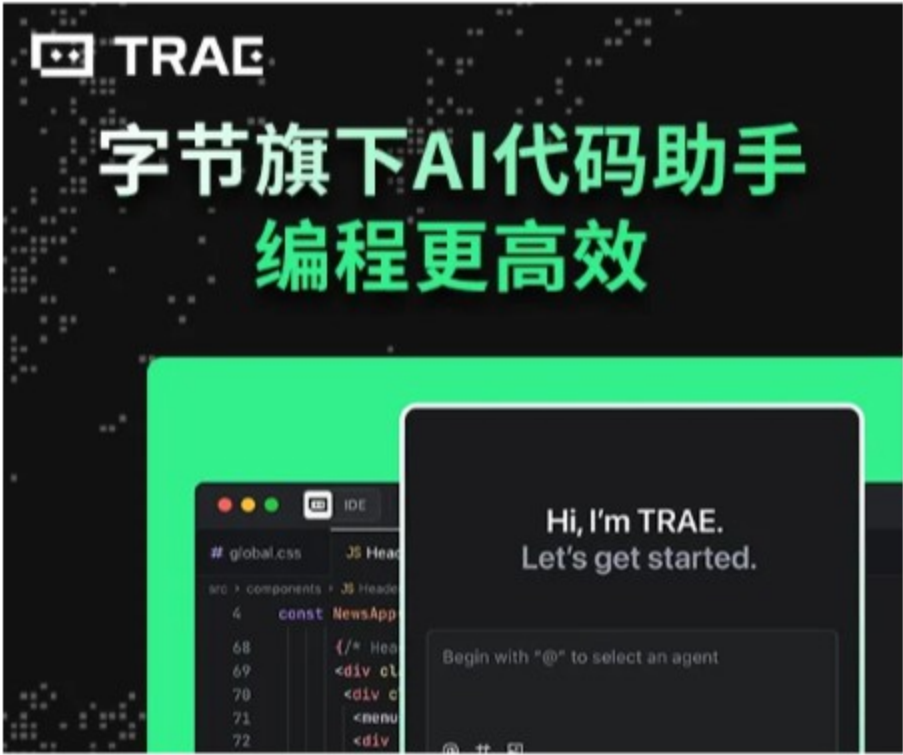
« 上一篇: [MySQL log rotate配置](#)
» 下一篇: [使用init_connect记录MySQL登录日志](#)

posted @ 2021-11-27 19:17 ZhenXing_Yu 阅读(1810) 评论(0) 收藏 举报

[刷新页面](#) [返回顶部](#)

 登录后才能查看或发表评论, 立即 [登录](#) 或者 [逛逛](#) [博客园首页](#)

- 【推荐】飞算科技, 让代码飞: 欢迎体验 JavaAI 开发助手, 参加炫技赛
- 【推荐】100%开源! 大型工业跨平台软件C++源码提供, 建模, 组态!
- 【推荐】AI 的力量, 开发者的翅膀: 欢迎使用 AI 原生开发工具 TRAE
- 【推荐】2025 HarmonyOS 鸿蒙创新赛正式启动, 百万大奖等你挑战



编辑推荐：

- 记一次 C# 平台调用中因非托管 union 类型导致的内存访问越界
- [EF Core] 聊聊“复合”属性
- 那些被推迟的 C# 14 特性及其背后的故事
- 我最喜欢的 C# 14 新特性
- 程序员究竟要不要写文章

阅读排行：

- 遭遇疯狂 cc 攻击的一个周末
- C#/.NET/.NET Core技术前沿周刊 | 第 49 期（2025年8.1-8.10）
- 美丽而脆弱的天体运动：当C#遇见宇宙混沌
- GPT-5 重磅发布
- 【EF Core】聊聊“复合”属性

公告

昵称： ZhenXing_Yu
园龄： 11年5个月
粉丝： 20
关注： 5
[+加关注](#)

2025年8月						
日	一	二	三	四	五	六
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

常用链接

- 我的随笔
- 我的评论
- 我的参与
- 最新评论
- 我的标签

最新随笔

- 1.a
- 2.OB学习笔记
- 3.MySQL 简单造数
- 4.一些MySQL有意思的知识点
- 5.Linux 下NC比较实用的用法示例
- 6.Linux下NFS配置
- 7.MySQL 8.0 Undo Tablespace管理
- 8.关于MySQL时间直接相减的问题
- 9.MongoDB对日志文件进行归档的方法
- 10.如何开启MySQL coredump

我的标签

- mysql(15)
- Percona Xtrabackup(4)
- Oracle(3)
- MongoDB(3)
- 备份恢复(3)
- nc(2)
- mysqlbinlog(2)
- MGR(2)
- logrotate(2)
- linux(2)
- 更多

随笔分类

- Linux(6)
- MongoDB(3)
- MySQL Fundamental(20)
- MySQL 安装部署(3)
- MySQL 备份恢复(5)
- MySQL 插件(2)
- MySQL 高可用(1)
- MySQL 故障诊断(2)
- MySQL 数据迁移(2)
- MySQL 体系结构(2)
- MySQL 性能优化(6)
- MySQL8.0(2)
- Oracle DataGuard(2)
- Oracle Fundamental(6)
- Oracle 实用语句(4)
- 更多

随笔档案

- 2025年4月(1)
- 2024年8月(1)
- 2022年11月(1)
- 2022年6月(5)
- 2022年2月(1)
- 2022年1月(2)
- 2021年12月(2)
- 2021年11月(1)
- 2021年8月(18)
- 2020年8月(2)
- 2020年5月(2)
- 2020年3月(3)
- 2019年6月(1)
- 2016年7月(1)

2016年5月(3)
更多

认证

Oracle OCP 11g
MySQL OCP 5.6
MySQL OCP 5.7
MongoDB 4.0 MCA
TiDB PCTA

阅读排行榜

- 1. Oracle 索引的失效和重建(16480)
- 2. MySQL 插件之 连接控制插件(Connection-Control)(13783)
- 3. NTP时间服务器配置与解析(12231)
- 4. MySQL FEDERATED 存储引擎的使用(11866)
- 5. MongoDB TTL索引的使用(11357)

评论排行榜

- 1. MySQL Shell import_table数据导入(2)
- 2. MongoDB TTL索引的使用(2)
- 3. MySQL 插件之 连接控制插件(Connection-Control)(2)
- 4. 使用Oracle 11g新特性 Active Database Duplication 搭建Dataguard环境(2)
- 5. MySQL LOAD DATA的常见用法(1)

推荐排行榜

- 1. 使用Docker-compose部署MySQL测试环境(4)
- 2. MySQL FEDERATED 存储引擎的使用(2)
- 3. MySQL 8.0 Undo Tablespace管理(1)
- 4. MySQL log rotate配置(1)
- 5. MySQL InnoDB Cluster(1)

最新评论

- 1. Re:MySQL 插件之 连接控制插件(Connection-Control)
@ZIKT 是的,感谢纠正,笔误了...

--ZhenXing_Yu

2. Re:MongoDB TTL索引的使用

又学到了一个知识点

--D调亿点

3. Re:MySQL LOAD DATA的常见用法

如果是定长文件(每一行字节数固定, 每列有字节范围, 比如1-4, 5-8), 里面含有中文. 第四种方式 substr就用不了了把, 因为定长文件是用字节来限定范围的.

--大乔上碗茶

4. Re:MySQL 插件之 连接控制插件(Connection-Control)

博主, "connection_control_min_connection_delay"这个参数的单位应该是毫秒吧?

--ZIKT

5. Re:MySQL Shell import_table数据导入

@愚夫c 嗯, secure_file_priv的修改需要重启,不过这个参数属于安装部署规范类的概念了,一般数据库创建时建议提前明确配置好,比如我的参数配置 cat /data/mysql/3306/...

--ZhenXing_Yu