# 硬件故障

硬件故障是指在某些特殊情况下（比如电源不稳定造成硬件损坏或者硬件本身老化等原因），导致硬件设备无法正常使用的故障。

出现硬件故障，大部分情况会导致存储服务器容量减少，或某些节点不能正常运行服务，极端情况下当有多个节点同时出现问题的时候，还会影响到业务使用、数据安全等。因此，在发现硬件问题的时候要及时处理，避免引发更多的问题。

系统出现硬件故障的情况下，首要的处理原则就是恢复业务，必要时启动应急流程；在业务恢复后，在进行故障的排查和处理。

# 计算节点

## Proxy启动失败

### 故障描述

Proxy状态异常。

### 故障分析

查看进程是否存在，结果不存在。

### 解决方法

修复步骤：

1. 登录proxy用户，检查告警日志（~/log/amrminfo），查看是否存在原因码为20001的“链路异常”的告警，若存在执行步骤3，否则执行步骤3
2. 登录proxy用户，查看proxy所在服务器与管理节点之间的链路是否正常；查看proxy配置文件中的各个端口和IP（域名）是否正确；若配置了proxymanager的IP为域名，请进一步确认该域名解析是否正确

操作命令：

ping 管理节点服务器IP

cat ~.etc/os.ini

cat ~/etc/proxy.ini

nslockup 域名

一切正常请执行dbmoni -start重启进程；若存在问题请修复问题后再重启proxy进程（域名若解析失败请查看DNS服务是否正常）

1. 查看告警日志中是否有原因码为2050303的告警

若存在，说明proxy启动时启用集群下所有连接实例失败，请执行步骤4，否则执行步骤10

1. 查看dbproxy.log的错误日志，查找管家你“can’t connect to MySQL server”，若存在该关键字，请根据日志中提示的IP信息，结合OMM界面拓扑图找到DB对应的服务器

若不存在跳转至步骤6，否则继续执行步骤5

1. 登录DB用户，检查DB进程，若DB进程不存在请检查DB配置文件是否正确，将不正确的地方修改后重启DB进程

继续执行步骤10

1. 查看dbproxy.log中错误日志，查找关键字“Access denied for user”，若存在该关键字，请在~/data/conninstance.xml中找到所启动失败的连接实例的密码信息，复制后在proxy用户下解码查看密码明文信息是否正确

操作命令：

dbtool -daes 连接实例密码

dbtool -dutil 连接实例用户名 连接实例密码

若decode\_type=2，请使用dbtool -dsec xxx解码，若decode\_type=3，请使用dbtool -dutil xxx xx解码；解码后的铭文信息不正确，请执行步骤7，否则执行步骤9

1. 登录OMM界面，选中proxy原来绑定的连接实例的用户，修改密码
2. 重启proxy进程，并且在OMM中重新绑定连接实例
3. 根据dbproxy.log中“Access denied for user”关键字日志描述的DB信息，结果OMM拓扑结构，找到对应DB服务器，登录DB所在用户，使用root用户进行mysql连接并查看连接实例用户在DB的权限，根据实际情况需要对用户赋权
4. 重启proxy进程

恢复方案：重新安装该proxy，并重新绑定连接实例

1. 登录管理节点所在服务器和异常proxy所在服务器，确认机器运转正常，网络正常
2. 登录异常proxy用户，停掉proxy，清空用户
3. 在OMM界面自动安装proxy
4. 登录proxy用户，检查配置文件以及DNS
5. 在OMM删除proxy
6. OMM绑定连接实例，重新设置加密方式等信息
7. 登录proxy用户，启动进程
8. 检查链路
9. OMM将该链路纳入管理
10. OMM启用连接实例

注：

1. 上述重新安装的步骤会清空proxy所在用户，如果有重要数据需要备份
2. 若集群中只有一个proxy，上述操作可能会造成GTM处存在残留活跃GTID，可在上述操作完成后重启proxy发起自启动回滚

## 客户端向proxy建链失败

### 故障描述

数据库客户端连不上启用的proxy连接实例

### 故障分析

1. 登录proxy所在用户，执行pgrep -u $USER | grep dbproxy查看proxy继承是否存在，若能查看到proxy进程，执行步骤2，否则判断故障场景为场景一：proxy进程退出；
2. 执行dbtool -p -m -i命令查看链接实例监听端口是否正常，若dbtool结果打印出监听端口号，执行步骤3,；若显示端口号为none或dbtool命令返回错误，额判断故障场景为场景二：proxy监听端口未正常监听
3. 在alrminfo.log中查看是否存在20510告警，若存在，则判断故障场景为场景三：客户端最大连接数不足

### 解决方法

修复步骤：

场景一：proxy进程退出

1. 登录proxy用户，根据之前proxy进程是否存在执行后续操作，若proxy进程不曾启动，执行步骤2，若正常运行，跳转到步骤3
2. 直接执行dbmoni -start启动proxy进程，确认该proxy是否已经加入集群并启用，若未加入集群，在OMM上添加到集群、绑定并启用连接实例
3. 查看alarminfo.log的告警和dbproxy.log日志分析proxy进程退出原因
4. 重新启动proxy，恢复连接实例监听端口

场景二：proxy监听端口未正常监听

1. 登录proxy用户，执行dbtool -p -m -i命令查看proxy连接实例信息

若步骤1中返回错误信息“proxy is not inited, request is permitted”执行步骤2；若能正常显示但端口显示为none，跳转至步骤4,；若结果中正常显示端口号，显示mysql连接该端口，连接成功则恢复正常，否则请根据场景一恢复步骤操作

1. 执行tail -f dbproxy.log查看是否持续发音“get total metadata failed,keep trying”

等待一段时间，若不再持续不断打印，重复执行步骤1；若长时间持续打印，执行步骤3

1. 查看dbproxy.log是否有error级别日志显示获取全量元数据失败，根据实际日志内容（MDS响应错误、管理节点链路不通等）解决环境问题后重新启动proxy后重复步骤1观察
2. 登录OMM，查看连接实例是否已经绑定并启用连接实例
3. 在步骤4基础上继续操作，若连接实例未绑定，需要绑定后启用；若已经绑定，直接启用
4. 在OMM，登录主GTM用户查看GTM状态是否正常，proxy与GTM之间链路是否正常

主GTM状态不正常则恢复主GTM，正常，则执行步骤7

1. 查看dbproxy.log，搜索error级别日志，查看连接实例启动失败的原因（DB异常或连接实例的用户密码有误），根据报错原因修复环境后启动proxy，重复执行步骤1

场景三：客户端最大连接数不足

1. 登录OMM，选中proxy查看最大连接数
2. 禁用该连接实例，解绑后修改最大连接数，然后绑定重启

恢复方案：强行杀掉proxy进程后，从OMM界面删除proxy再重新添加proxy，重新绑定并启用连接实例。

1. 登录OMM，查看整体链路
2. 登录proxy用户，查看进程是否存在，如果存在停掉proxy，清空data，检查proxy服务器是否正常
3. 等待月45s后，在OMM界面删除proxy
4. 登录proxy用户检查配置文件，然后启动proxy
5. 在proxy用户检查链路
6. 在OMM将proxy重新纳入管理
7. 绑定并启用连接实例

注：由于将proxy进程强制杀死并删除proxy，GTM处可能存在未释放的活跃GTID，若存在处于启用状态的对等proxy，活跃的gtid将被对等proxy回滚掉；若不存在处于启用状态的对等proxy，需要在执行完上述恢复步骤后，重启proxy进程发起自启动回滚。

## 通过proxy删除db上存在库表失败

### 故障描述

Proxy上执行drop database|table命令失败后，库表的元数据在GoldenDB中已经不存在，直接drop database | table报错：

ERROR 1008(HY000):Can’t drop database ‘test’;database doesn’t exists;part of DDL may succes,please manual recovery!

### 故障分析

proxy上执行drop database|table命令失败后，库表的元数据在GoldenDB中已经不存在，直接drop database | table报错，此时是因为proxy和db的元数据不一致导致的（即proxy的元数据认为表已经不存在了，但是db元数据中还保存着）。

### 解决方法

在drop语句中加上“if exists”:

drop databse if exists test;

## 通过proxy创建db上存在的表

### 故障描述

Proxy创建db上已经存在的库表报错：

ERROR 1008(HY000):Can’t create database ‘test’;database doesn’t exists;part of DDL may succes,please manual recovery!

### 故障分析

Proxy和db的元数据不一致。

### 解决方法

在create语句中加上“if not exists”。

## Proxy无法配置proxy最大连接数

### 故障描述

通过OMM界面配置的proxy连接实例最大连接数为5000，但是实际业务跟proxy之间创建的连接数达不到那么多（即客户端无法使用5000个连接），并且proxy会出现无法进一步创建客户端连接的情况，使用dbtool -p -c -i命令报错：link failure。

### 故障分析

实际proxy能够创建的最大客户端连接数在内部受到os控制，OS，ini中的配置项max\_link\_sessions决定了本proxy所有客户端连接数（包含dbtool命令创建的连接）。

### 解决方法

修改proxy下os.ini中max\_link\_session参数，使得值大于proxy绑定的所有连接实例中“客户端最大连接数”的总和加9（proxy内部耗费9个连接）。

## Proxy由于服务器配置低导致OMM

### 故障描述

GoldenDB的组件DB与proxy合设在一台2核7G内存的X86机器上，组件正常启动后，开启业务的情况下proxy会经常发生链接中断，连接不上。发现proxy进程会小时，PM日志会报错：proxy heart lose time[3]!Begin exception deal!

重启proxy后过段时间进程还是会消失，进一步了解后，DB进程也有一定概率被强制终止。

### 故障分析

通过分析系统的dmesg日志与mysqld日志，发现系统存在资源不足的情况。Linux系统在系统资源不足的情况下，会激活OMM机制，将占用资源的进程强制终止释放资源。结合机器的配置，2核PENTIUM的CPU、7G内存，并且是与DB合设的情况下，存在抢占资源的情况。

### 解决方法

将proxy迁移至其他机器，降低该机器DB的配置，例如：

max\_connection、open\_files\_limit、table\_open\_cache配置调低后，问题解决。

## 业务运行中有一组事务发生失败

### 故障描述

业务代码报错：unable to clear batch for prepare statement:Communication link failure。经过分析业务在执行跑批时执行clear batch，链路中断了，该中断是由服务端引起的。该时间段的proxy日志已经被清理。

### 故障分析

经过分析，发现在业务失败的时刻，产生了dbproxy\_dump.log，说明当时出现了proxy的正常宕机，同时在dbstop.log和history中均发现有人执行了dbstop的证据。因此，该问题是由于运维人员不通知业务的情况下重启proxy引起的。

### 解决方法

排查问题前需要先查看各个日志生成的时间，确认出现问题时的人工操作有哪些，尽可能避免人为问题导致的业务故障。

## Table lock/unlock proxy timeout

### 故障描述

用户执行DDL语句，例如ALTER、DROP操作时，报错：[table lock/unlock] the end of notification event(2467),fail num is 1,sucess num is 1,errinfo is [proxy 2 timeout]。

### 故障分析

通过对磁盘的监控，发现proxy网元所在磁盘的I/O存在很高的avgqu-sz，该指标表示等待处理的I/O请求，当请求超过磁盘处理能力，则该值将增加。假设该值等于2，表示持续有两倍读写能力的I/O请求在等待。由于过高的磁盘IO导致proxy修改本地元数据超时，导致报错。

### 解决方法

将proxy部署到单独的磁盘上，或者避免proxy所在磁盘有大量的IO。

## 业务遇到空指针问题

### 故障描述

业务返回空指针。

### 故障分析

根据业务代码分析，proxy返回的字段导致业务产生了空指针，proxy在oracle模式下返回的字段为纯大写形式，而业务比对的时候使用字符串比对的，导致业务结果集比对不出来，产生了空指针。

### 解决方法

Proxy的模式修改为mysql模式。

# 全局一致性控制节点

## GTM启动失败

### 故障描述

GTM进程启动后立即退出。

### 故障分析

登录GTM服务器，执行dbstate命令查看GTM进程状态，结果显示：[gtm] The GTM process is not running。

### 解决方法

1. 登录GTM服务器
2. 检查当前GTM用户下配置文件$HOME/etc/os.ini以及$HOME/etc/gtm.ini是否存在

如果不存在，则使用OMM界面自动安装功能重新安装GTM组件

1. 检查配置文件os.ini中的general段下ip地址配置项是否正确配置

GTM IP地址不支持域名设置，需要配置成ipv4格式地址

1. 检查配置文件os.ini中GTM段表下listen\_port配置的监听端口是否被其他进程占用：

操作命令：root用户下执行lsof -n IP:port

若端口被机器上其他进程占用，则在配置范围内重新配置新的listen\_port

1. 检查配置文件gtm.ini中general段下datadir数据目录配置项是否配置正确（数据目录datadir使用默认配置~/data即可）
2. 重启GTM进程：dbmoni -start

## GTM出现无主状态

### 故障描述

MDS告警：GTM切换异常。

### 故障分析

登录所有GTM服务器，执行dbtool -gtm -state命令查看GTM状态，发现没有GTM节点状态为MODE\_ACTIVE。

### 解决方法

**修复步骤：**

1. 修复主机服务器，重启主GTM：dbsatrt
2. 登录主GTM服务器执行dbtool命令查看主机状态是否为MODE\_ACTIVE：dbtool -gtm -state
3. 登录主GTM服务器执行dbtool命令查看主机与备机链路是否为LINKON：dbtool -gtm -ls

恢复方案：OMM页面上删除原主GTM后重新指定新主

## GTM出现双主状态

### 故障描述

GTM集群出现两个状态为MODE\_ACTIVE的GTM。

### 故障分析

登录所有GTM服务器，执行dbtool -gtm -state命令查看GTM状态，发现有两个GTM节点状态为MODE\_ACTIVE。

### 解决方法

1. 停止所有proxy进程：dbmoni -stop

防止proxy继续访问假主GTM造成获取到的活跃GTID数据不一致。

1. GTM出现双主状态是由于本地和同城机房间网络出现异常，需要对本地同城机房间网络进行恢复

场景：主GTM位于本地机房，管理节点位于同城机房，本地同城机房网络异常时，MDS将主GTM切换到同城机房，导致出现双主GTM状态。

1. 修复本地同城机房间网络故障后，对本地机房原主GTM状态进行检查，确认本地机房原主GTM已经降级为MODE\_INACTIVE状态

网络修复后，本地机房原主GTM上报给MDS的心跳消息中携带主机信息，MDS发现上报的主机与元数据库中不一致时通知该GTM降级。

1. 检查本地机房原主GTM与MDS以及新主GTM链路是否为LINKON状态
2. 重启所有proxy进程

## Proxy CR查询报错data is active

### 故障描述

通过mysql客户端连接proxy执行select...from db.tb...CR语句报错：“ERR:The data is active,please try again later!”

### 故障分析

登录主GTM服务器，执行dbtool -gtm -sc命令查看相应集群的MaxGTID值；通过mysql客户端连接相应集群下的一个主DB，执行select max(gtid) from db\_name.tb\_name查看B上的MaxGTID值，结果发现DB上的MaxGTID值比GTM上的MaxGTID大。

### 解决方法

1. 登录所有GTM服务器，按照先备后主的顺序依次停止所有GTM进程
2. 登录主GTM服务器，进入相应集群数据目录：cd $HOME/data/cluster1\_data
3. 删除old\_1.txt文件
4. 打开new\_1.txt文件，将new\_1.txt文件第一行的第一个“#”前的数字（MaxGTID）修改成比DB上查出来的MaxGTID大的数值，保存并退出
5. 按照先主后备的顺序依次重启所有GTM进程

## DB表入库GTID比GTM查找的GTID申请最大值大

### 故障描述

DB表数据入库的GTID的值比GTM上查找的GTID申请的最大值大。

### 故障分析

两个GoldenDB系统合并，其中迁移过来的数据的最大GTID大于现有系统的GTID。

### 解决方法

停止GTM进程，修改GTM的最大GTID。

## GTM单主异常状态INACTIVE后无法自动恢复

### 故障描述

在该版本，大部分集群会只有一个GTM的情况，如果这个GTM出现了异常，比如磁盘空间满内存不足状态被INACTIVE后，当资源被恢复后，状态无法自动恢复为ACTIVE。

### 故障分析

GTM目前的逻辑是故障后需要手动恢复，手动进行状态设置。但是针对单GTM的情况，由于没有修改状态的入口，OMM界面无法设置，dbtool也没有设置的命令，所以需要重启GTM进程。

### 解决方法

需要对GTM进程重启，重启后状态被MDS修改为ACTIVE。

## 磁盘I/O持续过高导致GTM Abnormal

### 故障描述

业务或者proxy客户端报错：ERROR(HY000):ERR:GTM Abnormal!

### 故障分析

现在通过对磁盘的监控，发现GTM网元所在的磁盘I/O存在很高的avgqu-sz，该指标表示等待处理的I/O请求，当请求超过磁盘处理能力，则该值将增加。假设该值等于2，表示持续有两倍读写能力的I/O请求在等待。

通过监控的数据可以发现，avgqu-sz的值达到了8395，从而导致GTM网元持久化增量数据时，等待写入的时间过长，从而在日志上反映出：GTM process run too too long，一直到最后的GTM Abnormal！

### 解决方法

由于GTM网元需要持续写增量数据到磁盘，请不要将GTM网元与其他网元共用磁盘。

# 元数据节点

## MDS操作数据库异常

### 故障描述

MDS的数据写入数据库很慢。

### 故障分析

检查告警文件有20004告警码。

### 解决方法

1. 在MDS上连接元数据库，如果不能正常连接，检查MDS配置文件中的数据库配置（IP、Port）是否正常
2. 执行select语句时间过长，如果能够访问数据库，但是DB的状态异常，则需要DBA介入修复DB

## MDS启动失败

### 故障描述

1. 元数据库未启动
2. DNS服务器未启动

### 故障分析

1. 元数据库未启动，查看MDS日志，有如下信息：

Can’t connect to MySQL server on “xxx.xxx.xxx.xxx”(Connection refused)

1. DNS服务器未启动，管理节点无法启动

### 解决方法

场景一：元数据未启动

1. 登录的GoldenDB用户，查看数据库状态：mysql.server status
2. 启动元数据库：mysql.server start

场景二：DNS服务器未启动

1. 启动DNS服务器：service named start
2. 检测DNS是否启动成功：nslookup 域名

# 集群管理节点

## 主DB异常无法切换

### 故障描述

CM检测到DBGroup切换失败。

### 故障分析

检查告警文件有无20205告警码。

### 解决方法

修复步骤：

1. 检测DBGroup内是否有可用备机

命令：ps -ef | grep mysqld

如果没有可用备机，则先修复该备机；修复完后，执行恢复方案；有可用备机，执行步骤3

1. 查看能够连上备机

命令：mysql -u -p -h -P

如果没有可用备机，则先修复该备机；修复完后，执行恢复方案；有可用备机，执行步骤3

1. 查看备机DBAgent进程是否正确

命令：dbstate

没有运行，执行步骤4，运行了执行步骤5

1. 启动DBAgent：dbstart
2. 在CM机器上查看备机的DBAGent和CM通信是否正常

命令：dbtool -cm -ls

查看备机与CM链路是否正常，链路正常，执行步骤6，链路不正常，排查配置及网络

1. 备机禁止切换标识是否被置位

命令：dbtool -cm -clear-disableswitch-falg -$ip -$port

可用性优先的场景下，备机可以切换为新主，执行恢复方案；一致性优先，禁切也没设置，还没切成功，进入步骤7

1. 如果是一致性优先的切换策略，查看是不是没有备机追上主机

命令：grep -sin “MDS Respond To CM That Consistent Copy Unexists” clustermanager.log

如果日志出现，说明该group的备机没有追上主机，需要修复备机，追上主机；当备机满足跟主机一致性的条件下，可以通过强制指定主，来把该备机切为新主，如果成功，执行修复方案

1. 查看CM日志

命令：grep -sin “cm\_switch\_fault” ~/log/clustermanager.log

如果前面的步骤依然不能解决问题，可以查看无法切换的具体步骤，找到具体原因。

恢复方案：强制指定主，此时备机没有追上主机，可能导致数据存在不一致的问题

1. 登录备机机器，连接mysql
2. 查看Gtid\_Set，从可用备机中找到Gtid\_Set最大备机

命令：show slave status

查看Executed\_Gtid\_Set，找到Gtid\_Set最大备机DB，执行步骤3

1. 在OMM界面上，执行强制指定主操作，等待切主成功

## CM启动失败

### 故障描述

CM启动不起来。

### 故障分析

登录CM服务器，执行dbstate命令查看CM进程结果不存在。

### 解决方法

1. 查看CM日志最后几行：tail -n 40 clustermanager.log
2. 若CM日志中出现“parameter not found or value is incorrect,CM exists”字样，则需要检查当前CM用户下配置文件$HOME/etc/clustermanager.ini相关配置是否正确，修改配置项
3. 重启CM进程

# 存储节点

## 备份binlog失败

### 故障描述

DBAGent在做binlog备份时，如果备份失败，就会产生备份失败告警。

### 故障分析

1. 备份所需要磁盘空间不足：df -h
2. DB上binlog日志未打开:my.cnf找那个log-bin段
3. 外部执行过reset master操作

如果外部对DB手动执行过“reset master;”操作，手动删除$HOME/etc/dbagent\_info/binlogbackup.info文件，重新进行备份

### 解决方法

针对不同的情况，修复的方案不同，按照上述原因，分别提供方案：

1. 手动清理备份目录所在的磁盘空间
2. 修改DB的配置文件my.cnf中的bin-log字段，打开binlog
3. 清理以前的binlog备份记录文件

## 备机不可用

### 故障描述

1. 备机短时间宕机，修复后主备差异数据量较少
2. 主备长时间宕机或长时间无法建立主备复制关系

### 故障分析

故障判断：

基于目前数据节点部署架构（本地两备，同城三备），备机宕机分为影响同城灾备高水位状态和不影响同城灾备高水位状态两种。

1. 影响同城灾备高水位状态，需立即应急处理，根据备机方剂时间长短选择不同处理方式，涉及一下场景：
2. 本地备机宕机
3. 同城备机全部宕机
4. 本地备机宕机，同城一台备机宕机
5. 本地和同城备机全部宕机
6. 不影响同城灾备高水位状态，可视当前故障缓急处理，根据备机宕机时间长短选择不同处理方式，涉及一下场景：

同城其中一台备机宕机

### 解决方法

1. 备机短时间宕机，修复后主备差异数据量较少
2. 若备机短时间宕机，修复后登陆主机，查看主备关系，若Slave\_IO\_Runing和Slave\_SQL\_Runing均为Yes，则主备复制关系正常，查看Seconds\_Behind\_Master进一步确认备机落后主机时间
3. 若备机启动后，主备复制关系异常，查看alarminfo错误日志查看，确认后续处理方式
4. 备机长时间宕机或长时间无法建立主备复制关系
5. 若备机长时间宕机，则选择最新的备机文件进行备份恢复
6. 检查Slave\_IO\_Runing和Slave\_SQL\_Runing均为Yes，则主备复制关系正常，查看Seconds\_Behind\_Master备机落后主机时间，直到显示为0，说明备机追平主机数据

## DBAgent日志报“gtid:xxx can’t find it”

### 故障描述

DBAgent在处理Proxy下发的提交事务回滚请求时，DBAGent返回回滚失败，并且DBAgent日志中报“gtid:xxx can’t find it!”

### 故障分析

事务回滚失败。

### 解决方法

修复步骤：binlog存在的情况下，从binlog文件中找到当前需要回滚的gtid，在DBAgent用户下手动执行dbtool命令，回滚掉当前gtid。

1. 根据DBAGent的日志提示，找到需要回滚的GTID和回滚发生的时间

命令：cat dbagent.log | grep “can’t find it”

如果有输出，执行步骤2

1. 根据日志中执行回滚的时间，找到binlog目录下，找到回滚时间附近的binlog文件，从这些binlog文件中找到需要回滚的gtid

命令：mysqlbinlog -vv mysql-bin.xxxxxx | grep gtid

如果找到gtid所在的binlog位置，执行步骤3，可能该事务所在的binlog文件被删除，则执行最终终极方案

1. dbtool手动执行事务的回滚

命令：dbtool -dbagent -roll-back gtid -start-time=2020-01-01 12:20:00

Start-time参数为回滚起始时间，要遭遇gtid所在的binlog文件的时间。查看dbtool返回结果，若失败执行最后终极恢复方案。

在DBAgent手动执行事务的回滚，但在GTM记录中DBAgent上回滚掉的事务的gtid的状态依然是活跃的。

**补救措施：**

1. 主GTM上执行dbtool，从记录中删除当前活跃的gtid：

dbtool -gtm -dg -cid=xxx gtid

1. 重启proxy

**终极修复方案：**

如果从当前binlog目录中没有找到需要回滚的gtid，需要从备份的binlog目录下找，找到后手动生成反向回滚SQL语句，DB上执行SQL语句。

## DBAGent启动失败

### 故障描述

mysqld、dbagent或loadserver启动不起来。

### 故障分析

登录DB用户执行ps -fu $USER，查看mysqld/dbagent/loadserver进程是否存在。

### 解决方法

1. 查看启动不起来的进程（mysql、dbagent、loadserver）日志
2. 启动失败主要是配置项配置错误导致，根据日志检查当前用户下配置文件的相关配置是否正确，修改不正确的配置项
3. 启动

## 集群缩容失败

### 故障描述

OMM界面发起删除dbgroup操作，失败日志如下：

MetaDataServer table [test.t1] ddl include groupid[1]

### 故障分析

待删除的dbgroup上仍有ddl和数据分片到其上。

### 解决方法

1. 登录到DB主上检查是否存在数据和表结构在该节点上；
2. 可能是重分布（缩容）后，原表rename之后的备份表（与原分布方式相同）仍然在该节点上

## 运行过程DB无法生成错误日志

### 故障描述

巡检环境发现，DB在运行过程中一直不输出日志到mysqld.log文件，并且mysqld.log文件也不存在。

### 故障分析

经过分析，维护人员的清理环境的时候，直接使用rm -rf命令删除所有日志，而运行态的DB不会再打开新的文件句柄输出日志文件。

### 解决方法

重启DB，启动DB时会自动生成日志文件。C语言的日志文件在运行态都不能直接删除。如果需要清理，可以使用echo > xx.log进行清空。

## 拷贝DATA目录的DB无法启动或建立主备关系

### 故障描述

备机的data目录是从主机全量拷贝的，拷贝完成后新的DB无法正常启动或者建立复制关系。

### 故障分析

Data目录中auto.cnf保存有原设备的uuid，master.info中保存有原设备（如果是备）的复制主关系，my.cnf中保存有原设备的serverid、ip等信息，均可能导致以上问题。

### 解决方法

把上述字段修改为原设备配置不同的即可（对于auto.cnf文件可以删除，重启DB的时候会自动生成的）。

## Slave sql回放线程很慢

### 故障描述

Slave回放线程很慢，导致备机严重落后于主机。

### 故障分析

1. 大事务
2. 回放的表没有索引

### 解决方法

1. 大事务需要慎用，可能会出现备机追不上主机的情况。如果多个大事务同时操作，可能会导致主备复制关系异常，甚至DBAgent无法与DB进行通信而导致集群异常
2. 添加索引或过滤该表不进行复制

## 备机IO线程出现1236报错

### 故障描述

备机复制中断，show slave status出现Last\_IO\_Error：Got fatal error 1236 from master binary log: ’The slave is connecting using CHANGE MASTER TO MASTER\_AUTO\_POSITION=1, but the master has purged binary logs containing GTIDs that the slave requires.’

### 故障分析

在主库上手动执行清除二进制日志文件，包括手动删除binlog以及reset master等操作

### 解决方法

1. 在主库上执行以下命令，查询gtid\_purged，记录下该值

show global variables like ‘%gtid%’\G

1. 在从库上执行以下命令，查询已经执行过的gtid即gtid\_executed，记录下主库的值，本机的不需要

show global variables like ‘%gtid%’\G

1. 在从库上执行以下命令停止同步线程及重置同步相关信息

stop slave;

reset slave;

reset master;

1. 在从库上设置gtid\_purged

该值有两个来源，一个在主库上查询gtid\_purged，二是在从库上查询的已经执行过的gtid\_executed值（本机的就不需要主库上GTID）

注意：一定记得加上从库上已经执行过的GTID，若设置了主库上的gtid\_purged，此时从库会重新拉取主库上所有的二进制日志文件，同步过程会出现其他错误，导致同步无法进行

set @@global.gtid\_purged=’\*\*\*\*:\*\*-\*\*,\*\*\*\*:\*\*-\*\*’;

注意：设置gtid\_purged值时，gtid\_executed值必须为空否则报错，该值清空的方法就是reset master命令。

执行完再次查看相关性信息

1. 重新开启同步

change master to master\_host=’’,master\_port=3306,

master\_user=’repl’,master\_password=’’,master\_auto\_position=1;

start slave;

当从库追赶上主库，此时测试主从数据是否一致，测试结果一切正常

## Join执行走全表扫描，未走主键匹配

### 故障描述

Join字段为主键字段进行匹配时，未走主键匹配而是全表扫描。

### 故障分析

mysql会对语句进行改写，而改写后的语句则可以在explain的show warnings中展示。

通过执行show warnings获取arcard\_auth取改写后的语句发现，在全表扫描的语句执行主键关联时多了一个“convert (‘1111’=`starcard\_autu`.`accl`.`ami\_account\_manage\_id` using utf8mb4)”，原来是做了主键的格式转换。

做全表扫描的关联表，其中一个表主键是utf8格式，另一个表的主键格式为utf8mb4，导致主键在做=关联的时候做了格式转换。

### 解决方法

将表重建，统一主键字段为UTF8或者utf8mb4

## MySQL网络超时错误

### 故障描述

mysql server网络超时。

### 故障分析

mysqld server和网络超时相关的参数有：

interactive\_timeout、wait\_timeout、net\_read\_timeout、net\_write\_timeout、connect\_timeout

wait\_timeout是给读请求用的，这个时候，连接是空闲的，等待用户的请求。等读完用户的请求包后，连接就变成active的，在调用dispatch\_command执行SQL前，把超时设置回net\_read\_timeout，之后在执行SQL请求过程中，server和client基本不会有网络交互，所以这个超时基本用不上。

一个特殊的情况是LOAD DATA LOCAL FILE命令，server在执行过程中，需要和client再做网络交互。

interactive\_timeout是给交互模式的客户端使用的，比如我们常用的mysql client工具，这个是在认证过程中设置的，如果设置的能力位上设置了CLIENT\_INTERACTIVE，会用interactive\_timeout的值覆盖wait\_timeout的值。而一般情况下，我们应用在建立连接时，是不会设置这个能力位的。

net\_write\_timeout对应写超时，在连接认证完成后，server和client交互过程中写超时一直是不变的。

connect\_timeout是给连接认证过程用的，读和写都用这个值，认证完成后，读和写分别设置为net\_read\_timeout和net\_write\_timeout。

### 解决方法

如果是认证过程中超时，不管是读还是写，都是connect\_timeout。

对于读网络超时，一般是wait\_timeout/interactive\_timeout，基本不会是net\_read\_timeout（特例是业务用LOAD DATA LOCAL FILE）；对于写网络超时，都是net\_write\_timeout。

在遇到超时情况时，可以根据这些原则判断对哪个参数做调整：

比如下面这种情况：

[Warning] Abort connection 6 to db:’unconnected’ user:’root’ host:’localhost’ (Got timeout reading communication packets)

很可能需要调整的wait\_timeout/interactive\_timeout。

[Warning] Aborted connection 12 to db:’test’ user:’root’ host”’localhost’(Got timeout writing communication packets)

需要调整net\_write\_timeout。

需要注意的是，MySQL的关于网络的错误，除了超时意外都认为是error，没有做进一步的细分，比如可能会看到下面这种日志，有可能是客户端异常退出了，也有可能是网络链路异常。

[Warning] Aborted connection 8 to db:’unconnected’ user:’root’ host:’localhost’ (Got an error reading communication packets)

[Warning] Aborted connection 13 to db:’test’ user:’root’ host:’localhost’(Got an error writing communication packets)

## packet for query is too large

### 故障描述

客户端返回错误日志packet for query is too large(19063787>1677216)

### 故障分析

MySQL根据配置文件会限制Server接受的数据包大小。有时候，插入、更新或者查询时数据包的大小，会受max\_allowed\_packet参数限制，导致操作失败。默认设置为16M。

### 解决方法

1. 在mysql控制台输入以下命令，设置max\_allowed\_packet为32M

set global max\_allowed\_packet = 33554432；

该方法仅对设置后的新链接生效（当前连接不生效）。

1. 修改$HOME/etc/my.cnf，设置max\_allowed\_packet=32M，然后重启生效

## show tables可以查看表名，但查询该表发现表结构不存在

### 故障描述

Show tables可以查看到表名，但是查询该表发现表结构不存在。

### 故障分析

在大小写敏感的时候，创建了大写的表，然后修改成大小写不敏感后，mysql对所有的表都修改成小写，对于大写表名就不识别了。

### 解决方法

1. 修改成大小写不敏感模式，然后重启mysql
2. 将所有大写表都rename成小写表，然后设置lower\_case\_table\_names=1，重启mysql

## waiting for table metadata lock

### 故障描述

Show processlist查看发现很多waiting for table metadata lock。

### 故障分析

一般是由于事务未提交导致元数据锁不释放引起的。

### 解决方法

**场景一：长事务运行，阻塞DDL，继而阻塞所有同表的后续操作**

通过show processlist可以看到tableA上有正在进行的操作（包括读），此时alter table语句无法获取到metadata独占锁，会进行等待。

这是最基本的一种场景，这个和mysql5.6中的online DDL并不冲突。一般alter table的操作过程中，在after create步骤会获取metadata独占锁，当进行到altering table的过程时（通常是最花时间的步骤），对该表的读写都可以正常运行，这就是online DDL的表现，并不会像之前在整个alter table过程中阻塞写入。（当然，也并不是所有类型的alter操作都能online的，具体参考官方手册说明）

处理方法：kill掉DDL所在的session。

**场景二：未提交事务，阻塞DDL，继而阻塞所有同表的后续操作**

通过show processlist看不到tableA上有任何操作，但是实际上存在有未提交的事务，可以在information\_schema.innodb\_trx中查看到。在事务没有完成之前，tableA的锁不会释放，alter table同样获取不到metadata的独占锁。

处理方法：通过select \* from information\_schema.innodb\_trx\G，找到未提交事务的sid，然后kill掉，让其回滚。

**场景三：执行失败操作未释放锁**

通过show processlist看不到tableA上有任何操作，在information\_schema.innodb\_trx中也没有任何进行中的事务，这很可能是因为在一个显式的事务中，对于tableA进行了一个失败的操作（比如查询了一个不存在的字段），这时候事务还没有开始，但是失败语句获取到的锁依然有效，没有释放。从performance\_schema.events\_statements\_current表中可以查看到失败的语句。

根据官方文档，除了语法错误，其他错误语句获取到的锁在这个事务提交或回滚之前，仍然不会释放掉。Because the failed statement is written to the binary log and the locks protect log consistency，但是解释这一行为的原因很难理解，因为错误的语句根本不会记录到二进制日志中。

处理方法：通过performance\_schema.events\_statement\_current找到其sid，kill掉该session，也可以kill掉DDL所在的sessioon。

# 数据迁移

## 大数据导入系统繁忙出现卡顿

### 故障描述

大数据导入导出时，系统出现卡顿现象，同时I/O操作不能正常完成。

### 故障分析

LDS导入导出过程中，文件下载（导入过程）、文件上传（导出过程）依赖于系统提供FTP功能，如果数据量过大，FTP上传下载过程中，占用大量的I/O，造成系统资源耗尽，无法正常处理其他I/O操作。

故障判断步骤：

1. 检查大数据导入导出时，系统有无出现卡顿
2. iostat -dx 1查看%util参数是否很满

### 解决方法

1. FTP server端/etc/vsftpd/vsftpd.conf流量控制配置项中annon\_max\_rate（数值单位为bytes/秒），可以通过此配置项控制导入导出过程中FTP对系统I/O占用

操作：适当减小该配置值，重启FTP服务：service vfstpd restart，重复此操作，直到系统卡顿现象减轻

1. 各个数据节点DBAgent配置项：max\_thread\_per\_optype=30可通过减少该配置项，控制FTP上传下载数目

操作：适当减小该配置值，重启服务DBAgent，pkill dbagent;dbstart;重复此操作，直到系统卡顿现象减轻

## 导入导出过程出现文件下载/上传失败

### 故障描述

导入导出时，文件下载/上传阶段，LDS报FTP失败。

### 故障分析

LDS上传/下载依赖于FTP服务，若LDS服务器未正常运行、服务器防火墙开启、服务器SELinux服务器开启以及FTP配置不正确，会出现上述错误。

故障判断步骤：

1. 文件导入导出时查看有无报错，报FTP失败
2. 检查LDS服务器FTP服务是否正常运行
3. 检查服务器防火墙有无开启
4. 检查SELinux服务是否运行
5. 检查FTP配置

### 解决方法

1. LDS服务器检查FTP服务是否正常

命令：service vsftpd status显示：vsftpd(pid 14412) is running...表示运行正常，如果不是，则需要启动FTP：service vsftpd start

1. LDS服务器检查SELinux是否关闭
2. 临时生效命令：getenforce显示：disabled表示关闭，若否，可以如下关闭：setenforce 0（临时）
3. 永久生效修改配置文件需要重启机器：修改/etc/selinux/config文件将SELINUX=enforcing改为SELINUX=disabled重启机器即可
4. 关闭防火墙

命令：systemctl stop firewalld

## LDS启动失败

### 故障描述

LDS启动不起来。

### 故障分析

登录异常的LDS所在用户，执行检查进程“loadserver”是否存在。

### 解决方法

1. 查看LDS日志最后几行
2. 若日志出现“... fail exit”字样，则需要检查当前LDS用户下配置文件$HOME/etc/loadserver.ini的相关配置是否正确，并修改不正确的项
3. 重启LDS进程

## 重分布前后数据不一致

### 故障描述

数据重分布，原表结构含有自增主键，重分布失败，报错全量数据导入后，数据总量检验不一致。

### 故障分析

Proxy建表主键是自增主键，当重分布由hash/range/list变更为duplicate时，由于主键冲突，导致重分布新表的数据量减少。

### 解决方法

1. 集群各个Group节点DB配置自增主键配置项后，重启DB

示例：集群2个group，主DB配置如下，主备配置一致：

g1：

auto\_increment\_increment=[group数目]

auto\_increment\_offset=1

g2：

auto\_increment\_increment=[group数目]

auto\_increment\_offset=2

1. 业务使用sequence代替自增列

## 导入oracle数据，无法识别

### 故障描述

将oracle的导出的数据文件导入GoldenDB的时候发现，文件导入失败，报错信息为字符无法识别。

### 故障分析

Oracle生成的文件编码格式为GB18030，而在导入的时候，GoldenDB设置的编码格式为GBK，GBK无法兼容所有的GB18030的字符。

### 解决方法

在导入文件之前对文件进行格式转换，将格式转换为UTF-8，并通过默认的UTF-8格式进行导入，在Linux执行命令“iconv -f GB18030 -t UTF-8 data.txt -o data\_utf8.txt”进行转码。

## LDS导入无法识别时间0001-01-01

### 故障描述

LDS导入数据报错：ERROR 1292：Incorrect datetime value:’0001-01-01’ for column ‘tb\_up\_time’ at row 1。

### 故障分析

tb\_up\_time字段类型为timestamp，而timestamp默认的时间是“0000-00-00”，而“0001-01-01”对mysql来说是非法的时间格式。

timestamp的时间范围UTC时间：1970-01-01 00:00:01.000000 to 2038-01-19 03:14:07.999999

### 解决方法

修改业务，将传递给LDS的默认时间修改为“0000-00-00”。

# 备份恢复

## 备份失败，DBAgent日志存在“Gtid is not contained”

### 故障描述

备机DBAgent收到ClusterManager的备份请求，执行备份命令，备份命令的日志无异常，但是DBAgent日志中有Gtid is not contained。

### 故障分析

备份完成后，DBAgent要比对备机位置是否追上主机，如果备机的位置并没有追上备份时刻的主机位置，那么这次备份是无效的，因此回复备份失败。

### 解决方法

修复步骤：检查主备关系是否正常，找到备机追不上主机的原因，并修复使得备机能够及时同步主机数据。

## 备份超时，DBAgent日志存在“Backup Timeout”

### 故障描述

DBAgent收到CM的备份请求，执行备份命令，备份命令的日志显示备份流程正常结束，DBAgent给CM上报的响应为超时。

### 故障分析

DBAgent的配置项backup\_timeout\_count配置的过小，innobackup执行备份时需要拷贝系统表文件，时间过小会超时。

### 解决方法

增大backup\_timeout\_count配置项

# HA组件

## 机房断电

### 故障描述

机房断电。

### 故障分析

机房断电。

### 解决方法

1. 机房内机器恢复运行后，首先分别登陆GTM和DB所在服务器启动GTM和DB
2. 启动数据库、OMM、管理节点

su - goldendb（omm用户名）

mysql.server start

jtool -mstart

若机房为双机HA环境，pcs cluster start（管理节点HA主备机均执行）

pcs resource enable rdbsh

pcs resource enable mdsssh

pcs resource enable ommsh

pcs resource enable pmsh

pcs resource enable cmsh

3、启动所有proxy进程

## HA2.0方案不支持资源不回切特性

### 故障描述

某些场景下，当集群切换到备节点后，主节点起来后，集群会自动切换到原主节点上去。

### 故障分析

在主节点未起来时，在备节点上执行pcs status，查看HA资源是在备节点上。主节点起来后，在备节点上执行pcs status，可看到HA资源启动在主节点上，自动完成了从备节点切换到主节点的过程。

### 解决方法

1. 该故障产生的原因是在搭建HA集群时，创建资源规则关系时设置的location规则的粘度不同，即假设设置如下：

pcs constraint location manager prefers dbmgr01=200

pcs constraint location manager prefers dbmgr02=100

可知dbmgr01的粘度大于dbmgr02，则一旦dbmgr01上线时，资源将会自动切换到dbmgr01上。

1. 若要不产生切换，则在设置location规则时设置两主机的粘度一致即可，操作命令：

在主节点上执行如下命令：

1. pcs constraint location manager prefers dbmgr01=100
2. pcs manager prefers dbmgr02=100

# 统一运维平台

## OMM安装并启动后界面无法访问

### 故障描述

OMM在安装启动后无法通过浏览器打开界面，出现以下几种情况错误：

1、404

2、空白页面

3、页面错乱报js错误

### 故障分析

1、404错误

业务在omm的机器上安装了其他应用，并配置了安装变量参数，导致omm读取该变量参数，导致界面打开失败。

2、空白页面

1）尝试在omm所在服务器通过wget命令访问omm

2）检查java进程是否存在

3）检查omm所连接的mysql/rddb是否启动，是否可访问，同时检查“was/tomcat/webapps/uniportal/WEB-INF/classes/com/zte/config/xml”中mysql的数据源、账号、密码配置是否正确

3、页面错乱

浏览器兼容问题

### 解决方法

1、404错误

将/etc/profile 中配置的spring的全局变量注释掉

2、空白页面

1）关闭防火墙

2）检查进程是否存在

3）检查访问数据库的配置是否正确

3、界面错乱

使用推荐的浏览器版本

## OMM安装后显示为中兴通讯界面

### 故障描述

OMM安装后，显示界面为中性通讯界面，行方生产环境不允许出现厂商信息。

### 故障分析

OMM默认安装使用的是中兴通讯表示，需要替换为行方的标识。

### 解决方法

1. 连接OMM底层数据库，执行语句

insert into zxinsys.portal\_sysparam(param\_name,param\_value,paramgroup) values(‘LONGLONG’,’XX银行’,’平台系统配置’)

upudate zxinsys.portal\_sysparam set param\_value=’XX银行’ where param\_name=’COMPANY\_NAME’;

1. 替换文件

/home/omm/was/tomcate/webapps/ROOT/icon.ico

/home/omm/was/tomcate/webapps/uniportal/\*

# 业务使用

## 业务侧连接数据库报错:Communications link failure

### 故障描述

业务侧连接数据库报错：Communications link failure，少量连接数可以连接，连接数一增大就报错。

### 故障分析

可能原因如下：

1. proxy连接实例中的连接池大小太小
2. DB的最大连接数过小
3. BD或proxy的句柄数过小

### 解决方法

1. OMM界面proxy连接实例中将最大连接数改大，proxy下os.ini的max\_link\_sessions改大
2. Mysql客户端连上mysql，show variables like ‘%max\_connections%’;确认是否是10000。如过小，则修改my.cnf中mysqld段，增加配置max\_connecions=10000，同时在客户端执行set global max\_connection=10000，不重启DB
3. DB和proxy用户下检查系统句柄数

## Communication link failure during commit() .Transaction resolution unknow

### 故障描述

业务抛出异常，日志如下：

Com.mysql.jdbc.exceptions.jdbc4.MySQLNonTransientConnectionExcep:Communications link failure during commit(),Transaction resolution unknow.

### 故障分析

此类异常抛出均在写语句的提交阶段。本模型中1insert+1update均会出现抛出异常日志。如下两种场景会导致该错误：

1. 原因是因为jdbc的socket超时时间小于dbgroup主备之间同步的超时时间（10s），当dbgroup的最后一个活跃备机异常（同步超时），或在业务socket超时时间内未完成提交则抛出此类错误
2. 单语句执行时间超时，超过jdbcsocket超时时间

### 解决方法

1. 请检查主备状态是否正常，复制关系是否正常
2. 请检查jdbccurl和业务代码中是否有socketTieout相关的配置

## 通过JDBC插入数据内容为空

### 故障描述

业务通过setString(‘2018-01-01’)的方式给date类型复制的时候发现赋值失败，导致数据未能入库，查询出来数据内容为空。

### 故障分析

原因是生成的连接类型是MariadbConnection导致消息不兼容。具体原因是业务在引用jar包的时候使用了mariadb的connect/j。

### 解决方法

修改引用jar包，删除mariadb的驱动包，改为mysql驱动包。