MariaDB基础培训

- 历史背景
- 基本操作
  - 安装操作 yum install xxxx
  - 初始化DB mysql_install_db
  - 启动数据库 systemctl start mariadb
  - 链接数据库 mysql -uxxx -pxxx -hxxx
  - 显示数据库列表 show databases;
  - 显示所有表 show tables;
  - 切换特定数据库实例 use xxx;
  - 查询特定库的特定表结构 show create table xxx; desc xxx;
- Galera相关
  - Galera介绍
  - 技术细节
  - Galera状态机
  - Primary Component
  - GTID
  - SST
  - IST
  - Gcache
  - Split-brain
  - grastate.dat
  - gvwstate.dat
- 常见问题
  - 第一时间判断是否是DB出了问题?
  - 如何在openstack的服务不能返回结果时，判断是否是数据库挂了?
  - 如何查看数据库当前的活动sql执行线程?
  - 如何收集Innodb引擎的日志用来分析系统异常?
  - 单节点故障之后如何恢复?
  - 事务不一致的情况?
  - 关于grastate.dat的seqno?
  - 初始化错误相关的?
  - 遇到初始化都搞不定的?
  - 遇到集群不能加入的情况?
  - 三节点都下电的情况，一般该怎么搞?
  - 集群检测以及恢复的脚本工具?
  - 关于备份的说明?
  - 仅在主节点发生异常时，才会出现三节点集群服务不可用的问题?
  - 脑裂一定发生?
  - 如何测试两节点集群的脑裂?
  - 上述造成集群服务最终不可用的改进措施？业界常用的方案有哪些?
  - 如何做流备复制?
  - 如何做自动分区表?
  - 官方错误集合

MariaDB基础培训

历史背景

- MariaDB 分支始于2009年的MySQL被Oracle收购。
- MariaDB 基于MySQL并遵循GPL v2授权使用。
- MariaDB 基金会的运作方式，确保开发过程公平、公开、公正。它将保证 MariaDB 始终开源。
- MySQL 是免费并且开源的，但它的开发者Oracle公司对其源代码拥有版权。该公司提供双重许可方式：一种是下普通版本遵循 GPL 的免费使用；一种是高级版本收费的商业许可证。
- MariaDB 提供更多的存储引擎，致力于更快的新特性开发、更稳定而快速的用户体验，完全开源。
- 兼容性良好，支持的SQL标准及大部分数据类型。

基本操作

安装操作 yum install xxxx

```
yum install mariadb-server.x86_64
```

初始化DB mysql_install_db

mysql_install_db

启动数据库 systemctl start mariadb

```
[root@test-controller1 ~]# systemctl start mariadb
```

```
[root@test-controller1 ~]# systemctl status mariadb
● mariadb.service - MariaDB 10.1 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor
preset: disabled)
   Active: active (running) since Wed 2017-08-30 09:52:56 CST; 5 days ago
 Main PID: 27716 (mysqld)
   Status: "Taking your SQL requests now..."
   CGroup: /system.slice/mariadb.service
           └─27716 /usr/libexec/mysqld --basedir=/usr

Warning: Journal has been rotated since unit was started. Log output is incomplete
or unavailable.
```

链接数据库 mysql -uxxx -pxxx -hxxx

```
[root@test-controller1 ~]# mysql -uroot -pAbc12345
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4706236
Server version: 10.1.12-MariaDB MariaDB Server
Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

显示数据库列表 show databases;

```
[root@test-controller1 ~]# mysql -uroot -pAbc12345
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4769832
Server version: 10.1.12-MariaDB MariaDB Server
Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+----------------------+
| Database             |
+----------------------+
| aodh                 |
| cascade_dashboard    |
| ceilometer           |
| cinder               |
| glance               |
| gnocchi              |
| heat                 |
| information_schema   |
| ironic               |
| keystone             |
| mysql                |
| neutron              |
| nova                 |
| nova_api             |
| octavia              |
| performance_schema   |
| portal               |
| test                 |
| zabbix               |
+----------------------+
19 rows in set (0.00 sec)
```

显示所有表 show tables;

切换特定数据库实例 use xxx;

```
[root@test-controller1 ~]# mysql -uroot -pAbc12345  zabbix
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4773457
Server version: 10.1.12-MariaDB MariaDB Server

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [zabbix]> use nova
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [nova]>
```

查询特定库的特定表结构 show create table xxx; desc xxx;

```
[root@test-controller1 ~]# mysql -uroot -pAbc12345  zabbix
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4771282
Server version: 10.1.12-MariaDB MariaDB Server

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [zabbix]> show create table history\G;
*************************** 1. row ***************************
       Table: history
Create Table: CREATE TABLE `history` (
  `itemid` bigint(20) unsigned NOT NULL,
  `clock` int(11) NOT NULL DEFAULT '0',
  `value` double(16,4) NOT NULL DEFAULT '0.0000',
  `ns` int(11) NOT NULL DEFAULT '0',
  KEY `history_1` (`itemid`,`clock`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
/*!50100 PARTITION BY RANGE ( clock)
(PARTITION p2017_08_16 VALUES LESS THAN (1502899200) ENGINE = InnoDB,
```

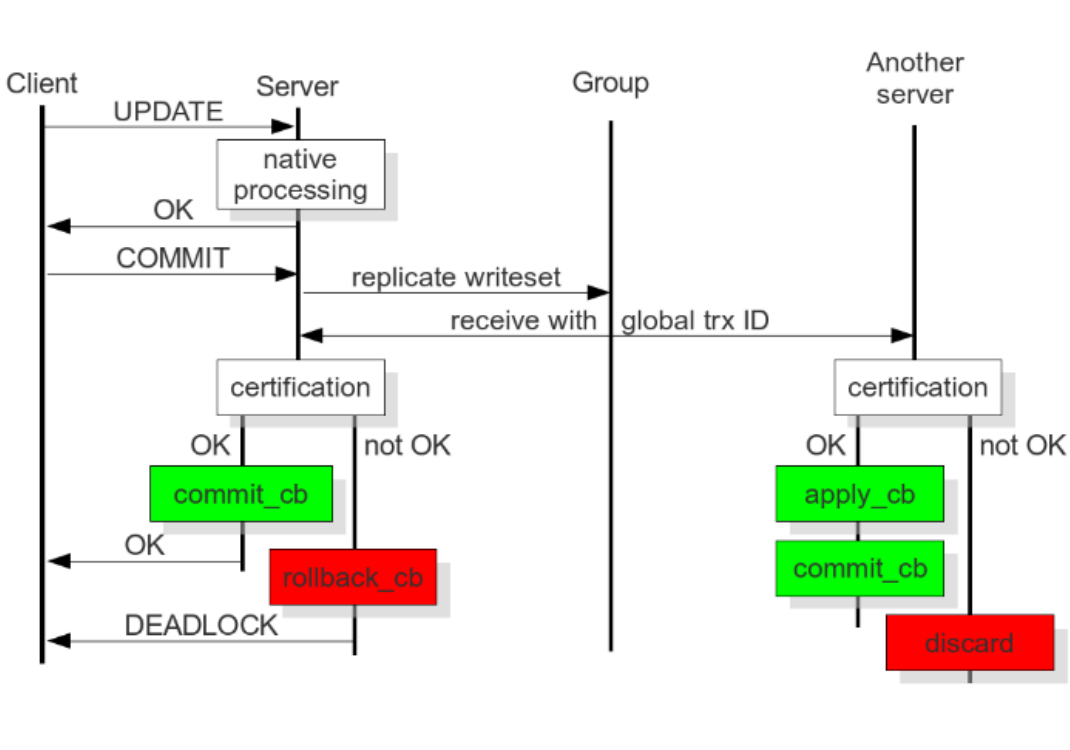
```
 PARTITION p2017_08_17 VALUES LESS THAN (1502985600) ENGINE = InnoDB,
 PARTITION p2017_08_18 VALUES LESS THAN (1503072000) ENGINE = InnoDB,
 PARTITION p2017_08_21 VALUES LESS THAN (1503331200) ENGINE = InnoDB,
 PARTITION p2017_08_22 VALUES LESS THAN (1503417600) ENGINE = InnoDB,
 PARTITION p2017_08_23 VALUES LESS THAN (1503504000) ENGINE = InnoDB,
 PARTITION p2017_08_24 VALUES LESS THAN (1503590400) ENGINE = InnoDB,
 PARTITION p2017_08_25 VALUES LESS THAN (1503676800) ENGINE = InnoDB,
 PARTITION p2017_08_26 VALUES LESS THAN (1503763200) ENGINE = InnoDB,
 PARTITION p2017_08_27 VALUES LESS THAN (1503849600) ENGINE = InnoDB,
 PARTITION p2017_08_28 VALUES LESS THAN (1503936000) ENGINE = InnoDB,
 PARTITION p2017_08_29 VALUES LESS THAN (1504022400) ENGINE = InnoDB,
 PARTITION p2017_08_30 VALUES LESS THAN (1504108800) ENGINE = InnoDB,
 PARTITION p2017_08_31 VALUES LESS THAN (1504195200) ENGINE = InnoDB) */
1 row in set (0.00 sec)
```

Galera相关

Galera介绍

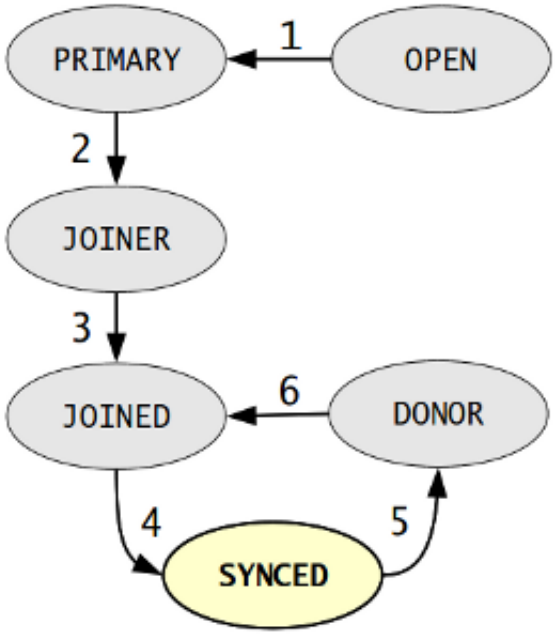
Galera是一个MySQL(也支持MariaDB，Percona)的同步多主集群软件，Galera集群的复制功能基于Galeralibrary实现，为了让MySQL与Galera library通讯，特别针对MySQL开发了wsrep API。 主要优点体现在高可用强，任意节点宕机不影响整体可用性，不丢数据，但要分区使用，不然容易出现大量的数据冲突。 主要缺点体现在性能差，节点越多性能越差。

技术细节



Galera Cluster的同步时事务级别的，其实现了基于认证的复制，使用集群间通信和事务排序技术来实现同步复制。自身是支持multi-master的，并且官方宣称写操作是同步的。大致过程如下： 创建事务时生成一个全局的事务ID，commit时严格按照ID顺序进行提交； 当客户端提交commit时，事务涉及所有修改的主键合并到一个write-set，这个write-set会送到集群中所有节点；节点收到write-set之后，会对其进行认证，检测是否会和本地的事务冲突，以此决定认证的成败； 当主节点收到其他节点认证成功的返回后，本地commit；如果失败，会对事务进行回滚，客户会收到dead lock。

Galera状态机



- **Open**：节点机器node启动成功，尝试与集群Primary节点建立连接，这个时候如果失败则根据配置退出或创建新的集群
- **Primary**：节点已经处于集群中，在新节点加入时会选取Donor进行数据同步
- **Joiner**：节点机器node发送state transfer状态成功，然后开始缓存write-set开始同步状态
- **Joined**：节点机器node收到state snapshot transfer快照，随即开始apply刚刚缓存的wirte-set，尝试保持和集群进度一致的过程状态
- **Synced**：节点机器node追平了集群的write-set，完成了同步并和集群进度保持一致。这个时候它的wsrep_ready状态变成1，开始处理事务操作
- **Donor**: 节点机器node收到一个state transfer请求，此时节点处于为新节点提供全量数据数据同步时的状态。此时该节点对客户端不提供服务

Primary Component

为避免单点故障，在网络故障发生时，集群可能分成几个部分。在此情形下，只有其中一个数据部分可以继续修改数据库状态用来避免历史数据的不一致，集群的这部分称为Primary Component。

GTID

全局事务ID，即Global Transaction ID，由UUID和sequence number偏移量组成。wsrep api中定义的集群内部全局事务id，一个顺序ID，用于记录集群中发生状态改变的唯一标识以及队列中的偏移量。

SST

状态快照迁移，即State Snapshot Transfer，集群中数据共享节点通过从一个节点到另外一个节点迁移完整的数据拷贝（全量拷贝）。当一个新的节点加入到集群中，新的节点从集群中已有节点同步数据，开始进行state transfer状态快照迁移（逻辑数据转移mysqldump或者物理数据迁移rsync|xtrabackup）。

IST

增量状态迁移，即Incremental State Transfer，集群一个节点通过识别新加入节点缺失的事务操作，将该操作发送，而并不像SST那样的全量数据拷贝。该方法只在特定条件下可用：新加入节点的状态UUID与集群组中节点一致；新加入节点所缺失的写数据集write-sets可以在Donor的写数据集write-sets存在。

Gcache

Galera集群将保存写数据集的一个特殊缓存称为Write-set Cacheh或Gcache，Gcache缓存为Gcache的内存分配程序。其目的主要是减少写数据集在内存上的占用空间。Galera集群对以上的改进是通过将内存中的写数据集存储卸载到硬盘上。

Split-brain

脑裂是数据库集群出现分区后每一部分都独立运行的一种情形。这种情形发生后，quorum算法难以发现Primary Component，然后导致无法执行任何查询操作。这是情形应尽力避免。

grastate.dat

识别最新节点ID是通过对比不同节点上GTID的值来进行，grastate.dat文件中保留了这些信息：在此文件中可以发现当前节点最新状态ID的值，当节点**正常退出**Galera集群时，会将GTID的值更新到该文件中。

gvwstate.dat

当集群形成或改变Primary Component时，节点会创建或更新gvwstate.dat文件，确保节点保留最新Primary Component的状态，如果节点无法连接，可以利用该文件进行参照，**如果节点正常关闭，该文件会被删除**。

常见问题

第一时间判断是否是DB出了问题?

判断DB是否OK，curl -i http://controller:9200 其中controller表示DB所在的vip。 如果这里返回503，则表示有问题。如果正常，一般会返回200。

```
[root@lzdc1ctl1 ~]# curl -i http://localhost:9200
HTTP/1.1 200 OK
Content-Type: text/plain
Connection: close
Content-Length: 40

MariaDB Galera Cluster Node is synced.
```

如何在openstack的服务不能返回结果时，判断是否是数据库挂了?

查日志 /var/log/mariadb/mariadb.log 查进程 ps aux|grep mysql 查参数

```
[root@test-controller1 ~]# mysql -uroot -pAbc12345 -e "SELECT * FROM
INFORMATION_SCHEMA.GLOBAL_STATUS where VARIABLE_NAME
in('WSREP_CLUSTER_SIZE','WSREP_READY','WSREP_LOCAL_STATE_COMMENT','WSREP_CLUSTER_S
TATE_UUID','WSREP_LOCAL_STATE_UUID');"
+---------------------------+--------------------------------------+
| VARIABLE_NAME             | VARIABLE_VALUE                       |
+---------------------------+--------------------------------------+
| WSREP_CLUSTER_SIZE        | 3                                    |
| WSREP_CLUSTER_STATE_UUID  | ea4e40f9-5f2c-11e7-9859-c2743c3de2fa |
| WSREP_LOCAL_STATE_COMMENT | Synced                               |
| WSREP_LOCAL_STATE_UUID    | ea4e40f9-5f2c-11e7-9859-c2743c3de2fa |
| WSREP_READY               | ON                                   |
+---------------------------+--------------------------------------+
```

如何查看数据库当前的活动sql执行线程?

```
show processlist;
```

如何收集Innodb引擎的日志用来分析系统异常?

```
show full processlist;
show open tables;
show status like '%lock%';
show engine innodb status\G;
```

单节点故障之后如何恢复?

- **单节点断电**引起的退出集群。当断电的节点重新上电开机并启动mysqld进程以后，会自动加入集群并完成同步。在我们的系统里，如果galera_daemon的进程已经存在，该进程会自动拉起这个节点。
- **单节点的mysqld进程被强制删除进程**引起的退出集群。当该节点的mysqld进程重新启动后，会

自动加入集群并完成同步。同上。

- **单节点的mysqld进程正常关闭**引起的退出集群。当该节点的mysqld进程重新启动后，会自动加入集群并完成同步。同上。
- **单节点的网络故障**引起的退出集群。当该节点的网络故障恢复后，会自动重新加入集群状态并完成同步。同上。

事务不一致的情况?

遇到如下报错，可以尝试commit。未遂的话就干脆rollback吧。/usr/libexec/mysqld --tc-heuristic-recover=ROLLBACK

```
[ERROR] Found 1 prepared transactions! It means that mysqld was not shut down
properly last time and critical recovery information (last binlog or tc.log file)
was manually deleted after a crash. You have to start mysqld with --tc-heuristic-
recover switch to commit or rollback pending transactions.
```

关于grastate.dat的seqno?

前面已经讲过，grastate.dat文件中保留了识别最新节点ID状态的信息。当节点**正常退出**Galera集群时，会将GTID的值更新到该文件中。 依次停止Galera不同节点上数据服务，会发现后来服务停止服务节点上**最后提交事务序号seqno越大**，这对于查找数据最新的节点非常有帮助。 如果该节点正在运行服务，grastate.dat文件中最后事务提交序号seqno为-1，文件如下：

```
[root@localhost mysql]# cat grastate.dat
# GALERA saved state
version: 2.1
uuid:    2c87abc6-96ea-11e8-a6ff-6f23f0fda450
seqno:   -1
cert_index:
```

最后提交事务ID不停再变化，可以通过以下命令查询：

```
show status like 'wsrep_last_committed';
```

初始化错误相关的?

遇到这样的，就删除数据目录data，重新初始化mysql_install_db，然后再启动。可能有隐藏 **.sst** 目录，记得一并删除干净。

```
Make sure the /var/lib/mysql is empty before running mysql-prepare-db-dir.
```

遇到初始化都搞不定的?

是不是配置似乎都正确，但是一启动该节点就作为孤立的集群提供服务，而且继上面的初始化之后，全是连接失败的报错，用户名密码全都有问题。 老老实实关掉服务好了。看看下面这段日志，是不是一模一样。那只能说明一个问题，出在了初始化集群的问题上。

```
2017-11-29 15:49:13 139928183229184 [Note] WSREP: Stop replication
2017-11-29 15:49:13 139928183229184 [Note] WSREP: Closing send monitor...
2017-11-29 15:49:13 139928183229184 [Note] WSREP: Closed send monitor.
2017-11-29 15:49:13 139928183229184 [Note] WSREP: gcomm: terminating thread
2017-11-29 15:49:13 139928183229184 [Note] WSREP: gcomm: joining thread
2017-11-29 15:49:13 139928183229184 [Note] WSREP: gcomm: closing backend
2017-11-29 15:49:13 139928183229184 [Note] WSREP: view((empty))
2017-11-29 15:49:13 139928183229184 [Note] WSREP: gcomm: closed
2017-11-29 15:49:13 139929480840960 [Note] WSREP: Received self-leave message.
2017-11-29 15:49:13 139929480840960 [Note] WSREP: Flow-control interval: [0, 0]
2017-11-29 15:49:13 139929480840960 [Note] WSREP: Received SELF-LEAVE. Closing
connection.
2017-11-29 15:49:13 139929480840960 [Note] WSREP: Shifting SYNCED -> CLOSED (TO:
0)
2017-11-29 15:49:13 139929480840960 [Note] WSREP: RECV thread exiting 0: Success
2017-11-29 15:49:13 139928183229184 [Note] WSREP: recv_thread() joined.
2017-11-29 15:49:13 139928183229184 [Note] WSREP: Closing replication queue.
2017-11-29 15:49:13 139928183229184 [Note] WSREP: Closing slave action queue.
2017-11-29 15:49:13 139929949494016 [Note] WSREP: New cluster view: global state:
afc39938-d4d9-11e7-bd12-0f8fae3ca3ab:0, view# -1: non-Primary, number of nodes: 0,
my index: -1, protocol version 3
```

拼人品的时候到了，应该是触发bug了。清掉环境变量然后重启即可。参考链接MDEV-7752

```
[root@host-1 mysql]# man systemctl
[root@host-1 mysql]# systemctl show-environment
LANG=en_US.UTF-8
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
_WSREP_NEW_CLUSTER=--wsrep-new-cluster
[root@host-1 mysql]# systemctl unset-environment "_WSREP_NEW_CLUSTER"
[root@host-1 mysql]# systemctl show-environment
LANG=en_US.UTF-8
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
[root@host-1 mysql]# systemctl start mariadb
```

遇到集群不能加入的情况?

认真**查防火墙**是否屏蔽了**4567**端口。可以用**nc**等查看

三节点都下电的情况，一般该怎么搞?

因为数据不值钱，在我们的系统里真丢一半个事务也是没所谓的。 在保证三个数据库节点都关闭的情况下，那就直接选任一数据库节点，操作该节点的**/etc/my.cnf.d/galera.cnf**文件，且看下面操作：

```
# Group communication system handle
wsrep_cluster_address="gcomm://"
```

一定要取消**gcomm**后面的三个ip地址，改成上面一行。然后启动mariadb的服务。等到该节点完全启动以后，连接另外两个数据库节点分别启动。不出意外的话，过两分钟整个数据库系统就恢复正常了。可以用curl -i http://controller:9200 去查看。再有问题，就召唤好了。

```
systemctl status mariadb.service
```

遇到数据文件比较大的情况，比如超过100G，可以将**/etc/my.cnf.d/galera.cnf**文件里的sst方式改为rsync的方式，会节省恢复的时间。

```
wsrep_sst_method=rsync
#wsrep_sst_method=xtrabackup-v2
```

集群检测以及恢复的脚本工具?

Linux环境下以命令行形式运行，直接添加集群中三个节点的名字即可。

```
cat galera.sh
#!/bin/sh
#set -x
password_galera_root=Abc12345
STOP_NODES=();
UUID=$(uuidgen)
rm -rf /tmp/GTID_*

findBootstrapNode(){
  for host in $(cat /tmp/GTID_${UUID}|grep "\-1"|awk '{print $2}')
  do
    VIEW_ID=$(ssh ${host} cat /var/lib/mysql/gvwstate.dat|grep view_id|awk '{print
$3}')
    MY_UUID=$(ssh ${host} cat /var/lib/mysql/gvwstate.dat|grep my_uuid|awk '{print
$2}')
    if [ $VIEW_ID = $MY_UUID  ];then
       echo $host
       break
    fi
  done
}

### 1. Check mariadb service in every nodes
for i in $@;
do
  FLAG=$(ssh $i systemctl status mariadb |grep Active:|grep running|wc -l)
  if [ "${FLAG}" = "0" ];then
```

```
    echo "[INFO] $i is down!"
    let INDEX=${#STOP_NODES[@]}+1
    STOP_NODES[INDEX]=$i
    seqno=$(ssh $i cat /var/lib/mysql/grastate.dat|grep seqno:|awk '{print $2}')
    echo $seqno" "$i >> /tmp/GTID_$UUID
  elif [ "$FLAG" = "1" ];then
    echo "[INFO] $i is up!"
  else
    echo "[ERROR] Get the status of $i mariadb is error!"
    exit 127
  fi
done
### 2. Recover Galera Cluster
let CLUSTER_SIZE=3-${#STOP_NODES[@]}
if [ "${CLUSTER_SIZE}" = "3" ]; then
  echo "[INFO] Galera is OK!"
elif [ "$CLUSTER_SIZE" = "2" -o "$CLUSTER_SIZE" = "1" ];then
  echo "[INFO] One or Two MariaDB nodes is down!"
  ## 2.1 Only start the mariadb service in stopped nodes
  for node in ${STOP_NODES[@]};
  do
    ssh ${node} systemctl start mariadb
  done
elif [ "${CLUSTER_SIZE}" = "0" ]; then
  echo "[INFO] All MariaDB nodes is down!"
  ABNORMAL_SIZE=$(cat /tmp/GTID_$UUID |grep "\-1"|wc -l)
  ## 2.2 Find the latest state node to bootstrap and start others nodes
  ## 2.2.1 All three nodes are gracefully stopped
  if [ "$ABNORMAL_SIZE" = "0" ];then
    BOOTSTARP_NODE=$(cat /tmp/GTID_$UUID|sort -n -r|head -n 1|awk '{print $2}')
    echo "[INFO] All three nodes are gracefully stopped!"
  ## 2.2.2 All nodes went down without proper shutdown procedure
  elif [ "$ABNORMAL_SIZE" = "1" ];then
    BOOTSTARP_NODE=$(cat /tmp/GTID_$UUID|grep "\-1"|awk '{print $2}')
    echo "[INFO] One node disappear in Galera Cluster! Two nodes are gracefully
stopped!"
  elif [ "$ABNORMAL_SIZE" = "2" ];then
    echo "[INFO] Two nodes disappear in Galera Cluster! One node is gracefully
stopped!"
    BOOTSTARP_NODE=$(findBootstrapNode)
  elif [ "$ABNORMAL_SIZE" = "3" ];then
    echo "[INFO] All nodes went down without proper shutdown procedure!"
    BOOTSTARP_NODE=$(findBootstrapNode)
  else
   echo "[ERROR] No grastate.dat or gvwstate.dat file!"
   exit 127
  fi
  ### Recover Galera
  echo "[INFO] The bootstarp node is:"$BOOTSTARP_NODE
```

```
  MYSQL_PID=$(ssh $BOOTSTARP_NODE netstat -ntlp|grep 4567|awk '{print $7}'|awk -F
"/" '{print $1}')
  ssh $BOOTSTARP_NODE /bin/bash << EOF
    kill -9 $MYSQL_PID
    mv /var/lib/mysql/gvwstate.dat /var/lib/mysql/gvwstate.dat.bak
    galera_new_cluster
EOF
  for i in $@;
  do
    if [ "$i" = $BOOTSTARP_NODE ];then
      echo "[INFO] $i's mariadb service status:"$(ssh $i systemctl status mariadb
|grep Active:)
    else
      echo "[INFO] $i start service:"
      ssh "$i" systemctl start mariadb
    fi
  done
else
  echo "[ERROR] Recover Galera Cluster is error!"
  exit 127
fi
### 3. Check Galera Status
sleep 5
WSREP_CLUSTER_SIZE=$(mysql -uroot -p$password_galera_root -e "SHOW STATUS LIKE
'wsrep_cluster_size';"|grep wsrep_cluster_size|awk '{print $2}')
echo "[INFO] Galera cluster CLUSTER_SIZE:"$WSREP_CLUSTER_SIZE
if [ "${WSREP_CLUSTER_SIZE}" = "3" ]; then
  echo "[INFO] Galera Cluster is OK!"
  exit 0
elif [ "$WSREP_CLUSTER_SIZE" = "2"  ];then
  echo "[INFO] One MariaDB nodes is down!"
  exit 2
elif [ "$WSREP_CLUSTER_SIZE" = "1"  ];then
  echo "[INFO] Two MariaDB nodes is down!"
  exit 1
else
  echo "[INFO] All MariaDB nodes is down!"
  exit 3
fi
```

关于备份的说明?

先解释**逻辑备份**和**物理备份**的区别：

- 逻辑备份，就是把数据库的结构定义语句以及数据内容的插入语句，全部存储下来。反映是数据库里DML操作层面的变化，一般都是以文本的形式。
- 物理备份，就是把存储好的所有文件保存下来，反映的是磁盘上的物理块的变化，一般都是以二进制的形式。

在我们的系统里，数据备份的方式采用的是物理备份，工具采用了innobackupex。会在每天凌晨01:15开始做一次全量备份，之后在每个小时的15分钟开始基于上一次的增量备份做新的增量备份，因此增量备份的压缩包反映的是每一次修改过的数据物理块的变化。

仅在主节点发生异常时，才会出现三节点集群服务不可用的问题?

不光主节点发生异常才会影响该问题（Galera 默认三节点都是主节点，我们使用场景下的区分只是通过Haproxy来区分主备节点的，从应用层打过来的链接而已，跟后端的Galera集群本质上没有任何关系）。Galera是一个对网络环境要求很高的类MySQL集群，不同程度的网络抖动会导致脑裂（也叫网络分区）将最终导致整个集群发生异常。

脑裂一定发生?

Weighted Quorum算法去保证当集群出现脑裂问题时，投票决定分裂以后的哪个Group将占有控制权。

而在三节点的场景中，当任一节点挂掉那么这个算法就失效了（只剩两节点存在），这个时候只能看pc.weight的权重了。而这个值的默认值是1，同样不能分出哪个节点的优先级搞，于是系统陷入两难。两节点集群的情况下，你们可以模拟网络抖动多操作几次，脑裂发生是迟早的问题，必然发生。

所以，正确的操作是，避免偶数个节点的集群运行。

如何测试两节点集群的脑裂?

- 断开两个节点的网络连接，Quorum丢失，所有节点不再接受请求
- 将网络连接恢复，Quorum仍然丢失，所有节点仍旧不接受请求。
- 登录其中一个节点，Reset Quorum:

```
SET GLOBAL wsrep_provider_options='pc.bootstrap=1';
```

- Quorum被重置，集群恢复

上述造成集群服务最终不可用的改进措施？业界常用的方案有哪些?

改成单节点状态，就是已知挂掉一个节点以后，还剩两个节点的情况下手动停止其中一个节点，只留一个节点来提供DB服务。业界的做法，一般会通过双网卡加双交换的方式（增加成本，架构变得复杂），做汇聚口来提供服务。另外，有参数配置可让pc.ignore_sb=true来取消集群判断脑裂（非多主场景下可以使用）。

互联网界很少有使用Galera大规模上生产环境的，首先因为它对网络环境要求很高，其次它的性能很弱，对并发环境不够友好（测试数据表明它在压测下大概是MySQL5.6原生主从环境下性能的60%不到）。Galera的网络既是约束也是硬伤，哪怕它很大程度上保证了数据的完整性。

上云之前，像京东这种大规模的生产环境，使用MHA来作为大集群的环境；腾讯的微信支付数据跑在基于PGXZ的集群系统上；阿里有自己的OceanBase；其它家如新浪用了MySQL主从；宣称使用Galera作为生产环境的Qunar也只是存了部分的非核心业务数据（核心的财务数据一部分部分跑在了PostgreSQL上）。

如何做流备复制?

如何做自动分区表?

官方错误集合

https://mariadb.com/kb/en/library/what-to-do-if-mariadb-doesnt-start/