# MySQL集群需求收集与方案规划(初)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本** | **内容** | **备注** | **作者** | **时间** |
| V1.0 | 收集MySQL集群需求 |  | 张华,宋海鹏 | 2018.1.26 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

[MySQL集群需求收集与方案规划(初) 1](#_Toc21144)

[1. 目标 2](#_Toc13295)

[2. 读者 2](#_Toc31172)

[3. MySQL 集群需求 3](#_Toc11815)

[3.1.应用情况 3](#_Toc29961)

[3.2.集群需求 3](#_Toc21299)

[4.业界MySQL集群技术方案汇集 3](#_Toc17270)

[4.1.高可用方案 3](#_Toc10453)

[4.2.高可用架构优化 4](#_Toc4849)

[4.2.1.MHA+MySQL 5](#_Toc4065)

[4.2.2.zookeeper+proxy 6](#_Toc2424)

[4.3.共享存储 7](#_Toc9453)

[4.3.1.SAN共享存储 7](#_Toc28049)

[4.3.2.DRBD磁盘复制 8](#_Toc26505)

[4.4.分布式协议 9](#_Toc20266)

[4.4.1.MySQL Cluster 9](#_Toc25629)

[4.4.2.Galera 10](#_Toc7279)

[4.4.3.POAXS 10](#_Toc15847)

[4.4.4.Mycat 11](#_Toc10854)

[4.4.5.MySQL group replication 12](#_Toc8617)

[5.MySQL集群规划 15](#_Toc14156)

[5.1.集群规划目标 15](#_Toc12041)

[5.2.集群规划方向 15](#_Toc23824)

[5.2.1.中小项目(千万级别数据量) 15](#_Toc5555)

[5.2.2.大型项目(亿级别数据量) 16](#_Toc23639)

[5.2.3.超大型项目(10亿+级别数据量) 16](#_Toc19636)

[6.实施计划与资源需求 16](#_Toc11502)

[7.潜在风险与问题 16](#_Toc15801)

[8.参考资料 16](#_Toc26272)

# 目标

1).收集公司过去,当前项目中使用MySQL集群的情况,遇到的主要问题,诉求等.

2).规划MySQL集群实施的整体方案,降低未来项目成本,加速推动项目落地.

# 读者

架构师,项目经理,工程师

# MySQL 集群需求

通过了解公司一些项目情况,有使用或近似使用MySQL集群技术的项目如下:

1).网络配置项目(孙博)

2).贵州医疗(少杰)

3).国家邮政实名制系统(杨旻)

4).移动四川咪咕音乐

## 3.1.应用情况

1).网络配置项目

主要技术:VIP+Keepalived+HA+Mycat+MySQL

说明:

VIP+keepalived提供接入冗灾

HA提供负载均衡

Mycat提供MySQL集群及分库分表(经了解,目前基于业务需求,暂未分库分表,Mycat主要提供数据路由)

MySQL承担底层存储及数据分析/检索

2).贵州医疗

3).国家邮政实名制系统

主要技术:Mycat+MySQL

Mycat提供MySQL集群分库分表支持,主要应对高并发环境下分库分表数据存储.

MySQL承担底层存储及数据分析

4).移动四川咪咕

主要技术:MySQL cluster+MySQL

MySQL Cluster提供MySQL集群解决方案

MySQL承担底层存储

## 3.2.集群需求

经对各项目了解,梳理,分析,对MySQL集群的需求,主要集中在以下几个方面:

1).高可用:如意外宕机立即自动恢复.

2).数据热备与一致性.

3).读写分离,降低查询对写入的影响.

4).分库分表,水平扩容.

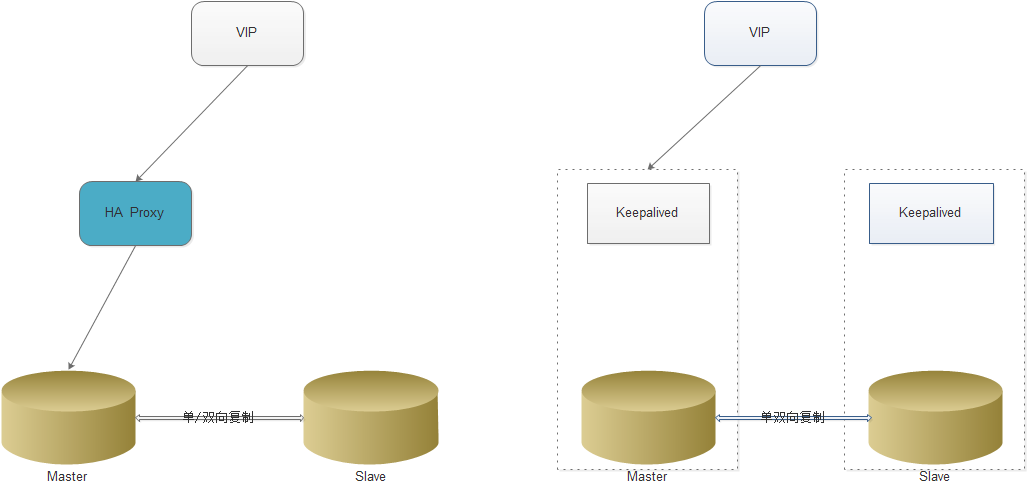
# 4.业界MySQL集群技术方案汇集

## 4.1.高可用方案

**主从或主主半同步复制**

使用双节点数据库，搭建单向或者双向的半同步复制。在5.7以后的版本中，由于losslessreplication、logical多线程复制等一些列新特性的引入，使得MySQL原生半同步复制更加可靠。

通常会和proxy、keepalived等第三方软件同时使用，即可以用来监控数据库的健康，又可以执行一系列管理命令。如果主库发生故障，切换到备库后仍然可以继续使用数据库。



注:以上为两种常见的应用方式.部分高可用场景,会结合使用HA和Keepalived,提供更高层面的高可用.

优点：

架构比较简单，使用原生半同步复制作为数据同步的依据；

双节点，没有主机宕机后的选主问题，直接切换即可；

双节点，需求资源少，部署简单；

缺点：

完全依赖于半同步复制，如果半同步复制退化为异步复制，数据一致性无法得到保证；

需要额外考虑haproxy、keepalived的高可用机制。

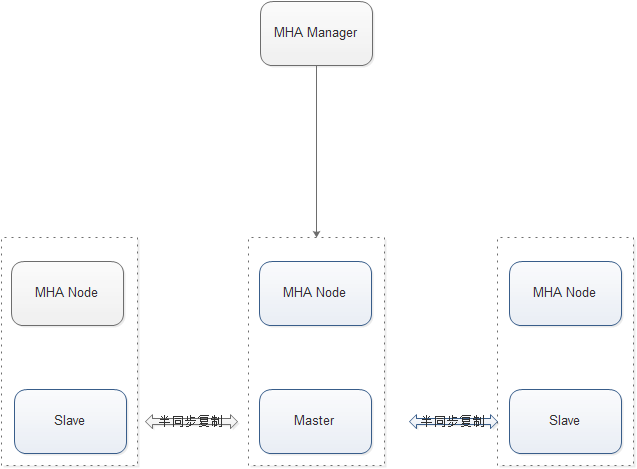
## 4.2.高可用架构优化

将双节点数据库扩展到多节点数据库，或者多节点数据库集群。可以根据自己的需要选择一主两从、一主多从或者多主多从的集群。

由于半同步复制，存在接收到一个从机的成功应答即认为半同步复制成功的特性，所以多从半同步复制的可靠性要优于单从半同步复制的可靠性。并且多节点同时宕机的几率也要小于单节点宕机的几率，所以多节点架构在一定程度上可以认为高可用性是好于双节点架构。

但是由于数据库数量较多，所以需要数据库管理软件来保证数据库的可维护性。可以选择MMM、MHA或者各个版本的proxy等等。常见方案如下：

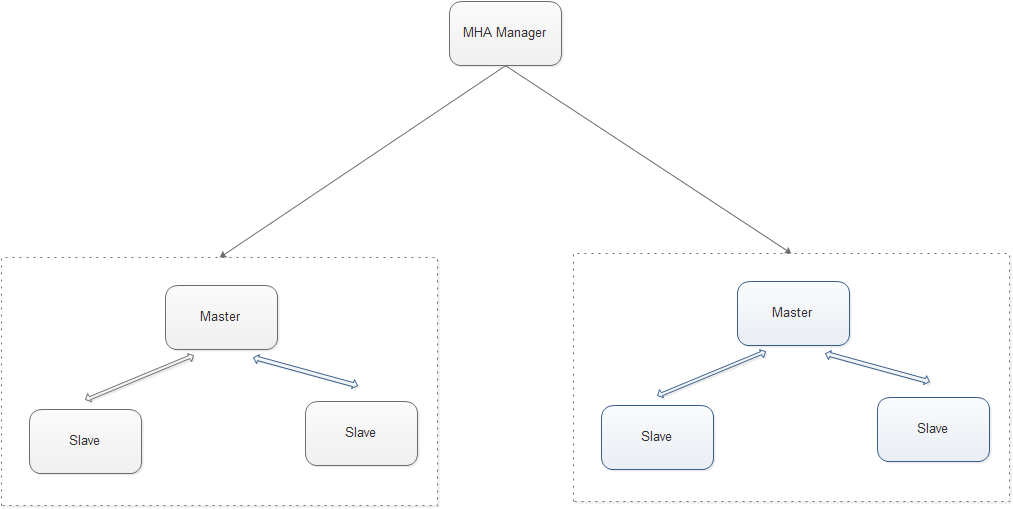
### 4.2.1.MHA+MySQL



MHA Manager会定时探测集群中的master节点，当master出现故障时，它可以自动将最新数据的slave提升为新的master，然后将所有其他的slave重新指向新的master，整个故障转移过程对应用程序完全透明。

MHA Node运行在每台MySQL服务器上，主要作用是切换时处理二进制日志，确保切换尽量少丢数据。

MHA也可以扩展到如下的多节点集群：



优点：

可以进行故障的自动检测和转移;

可扩展性较好，可以根据需要扩展MySQL的节点数量和结构;

相比于双节点的MySQL复制，三节点/多节点的MySQL发生不可用的概率更低

缺点：

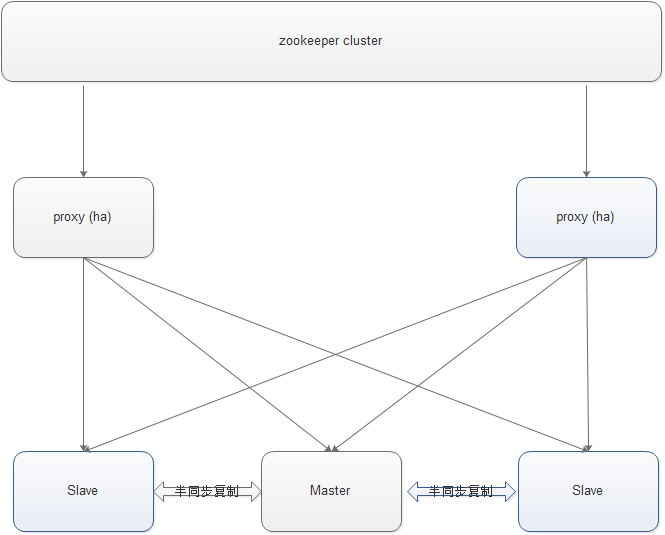
至少需要三节点，相对于双节点需要更多的资源;

逻辑较为复杂，发生故障后排查问题，定位问题更加困难;

数据一致性仍然靠原生半同步复制保证，仍然存在数据不一致的风险;

可能因为网络分区发生脑裂现象;

### 4.2.2.zookeeper+proxy

Zookeeper使用分布式算法保证集群数据的一致性，使用zookeeper可以有效的保证proxy的高可用性，可以较好的避免网络分区现象的产生。  


优点：

较好的保证了整个系统的高可用性，包括proxy、MySQL;

扩展性较好，可以扩展为大规模集群;

缺点：

数据一致性仍然依赖于原生的mysql半同步复制;

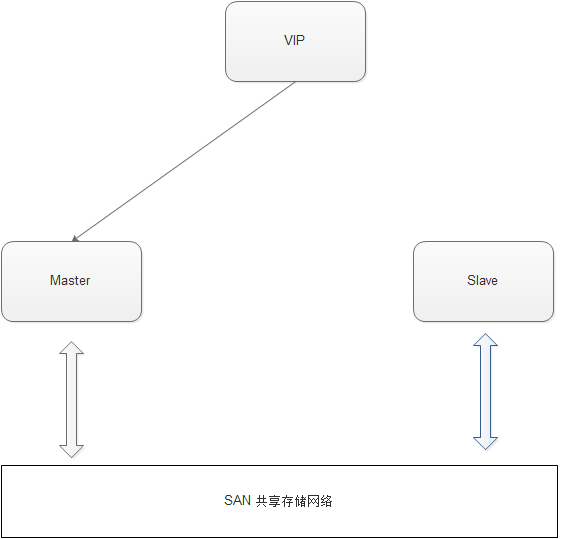
引入zk，整个系统的逻辑变得更加复杂;

## 4.3.共享存储

共享存储实现了数据库服务器和存储设备的解耦，不同数据库之间的数据同步不再依赖于MySQL的原生复制功能，而是通过磁盘数据同步的手段，来保证数据的一致性。

### 4.3.1.SAN共享存储

SAN的概念是允许存储设备和处理器（服务器）之间建立直接的高速网络（与LAN相比）连接，通过这种连接实现数据的集中式存储。常用架构如下：



使用共享存储时，MySQL服务器能够正常挂载文件系统并操作，如果主库发生宕机，备库可以挂载相同的文件系统，保证主库和备库使用相同的数据。

优点：

两节点即可，部署简单，切换逻辑简单；

很好的保证数据的强一致性；

不会因为MySQL的逻辑错误发生数据不一致的情况；

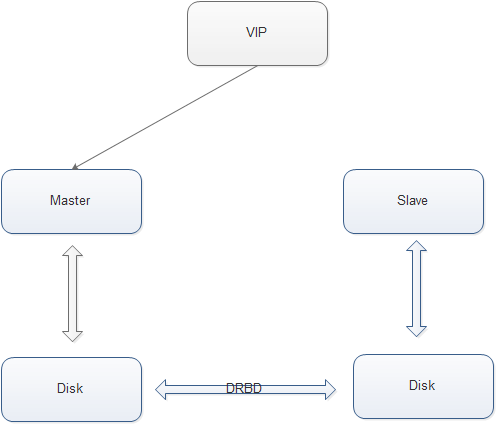
缺点：

需要考虑共享存储的高可用；

价格昂贵；

### 4.3.2.DRBD磁盘复制

DRBD是一种基于软件、基于网络的块复制存储解决方案，主要用于对服务器之间的磁盘、分区、逻辑卷等进行数据镜像，当用户将数据写入本地磁盘时，还会将数据发送到网络中另一台主机的磁盘上，这样的本地主机(主节点)与远程主机(备节点)的数据就可以保证实时同步。常用架构如下：



当本地主机出现问题，远程主机上还保留着一份相同的数据，可以继续使用，保证了数据的安全。

DRBD是linux内核模块实现的快级别的同步复制技术，可以与SAN达到相同的共享存储效果。

优点：

两节点即可，部署简单，切换逻辑简单；

相比于SAN储存网络，价格低廉；

保证数据的强一致性；

缺点：

对io性能影响较大；

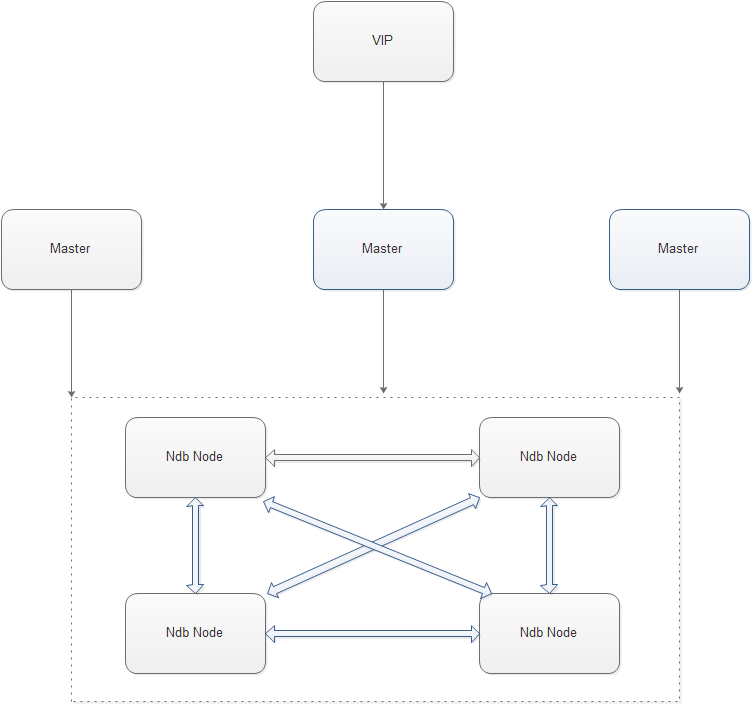
从库不提供读操作；

## 4.4.分布式协议

分布式协议可以很好解决数据一致性问题。比较常见的方案如下：

### 4.4.1.MySQL Cluster

MySQL cluster是官方集群的部署方案，通过使用NDB存储引擎实时备份冗余数据，实现数据库的高可用性和数据一致性。



优点：

全部使用官方组件，不依赖于第三方软件；

可以实现数据的强一致性；

缺点：

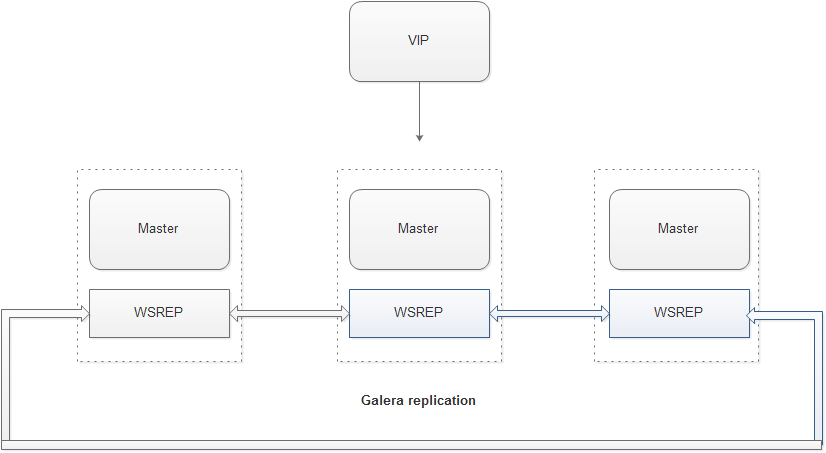
国内使用的较少；

配置较复杂，需要使用NDB储存引擎，与MySQL常规引擎存在一定差异；

至少三节点；

### 4.4.2.Galera

基于Galera的MySQL高可用集群， 是多主数据同步的MySQL集群解决方案，使用简单，没有单点故障，可用性高。常见架构如下：



优点：

多主写入，无延迟复制，能保证数据强一致性；

有成熟的社区，有互联网公司在大规模的使用；

自动故障转移，自动添加、剔除节点；

缺点：

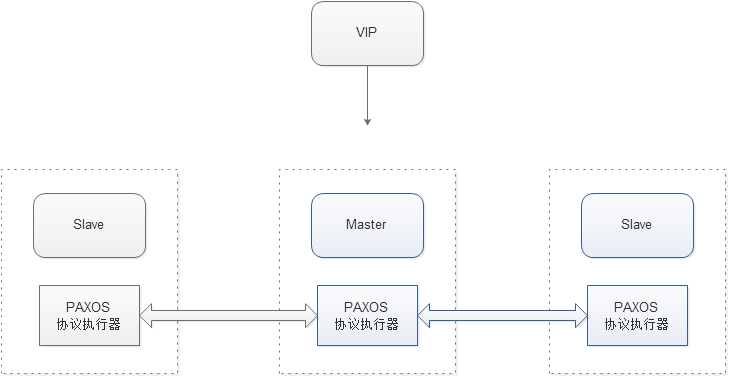
需要为原生MySQL节点打wsrep补丁

只支持innodb储存引擎

至少三节点；

### 4.4.3.POAXS

Paxos 算法解决的问题是一个分布式系统如何就某个值（决议）达成一致。这个算法被认为是同类算法中最有效的。Paxos与MySQL相结合可以实现在分布式的MySQL数据的强一致性。常见架构如下：



优点：

多主写入，无延迟复制，能保证数据强一致性；

有成熟理论基础；

自动故障转移，自动添加、剔除节点；

缺点：

只支持innodb储存引擎

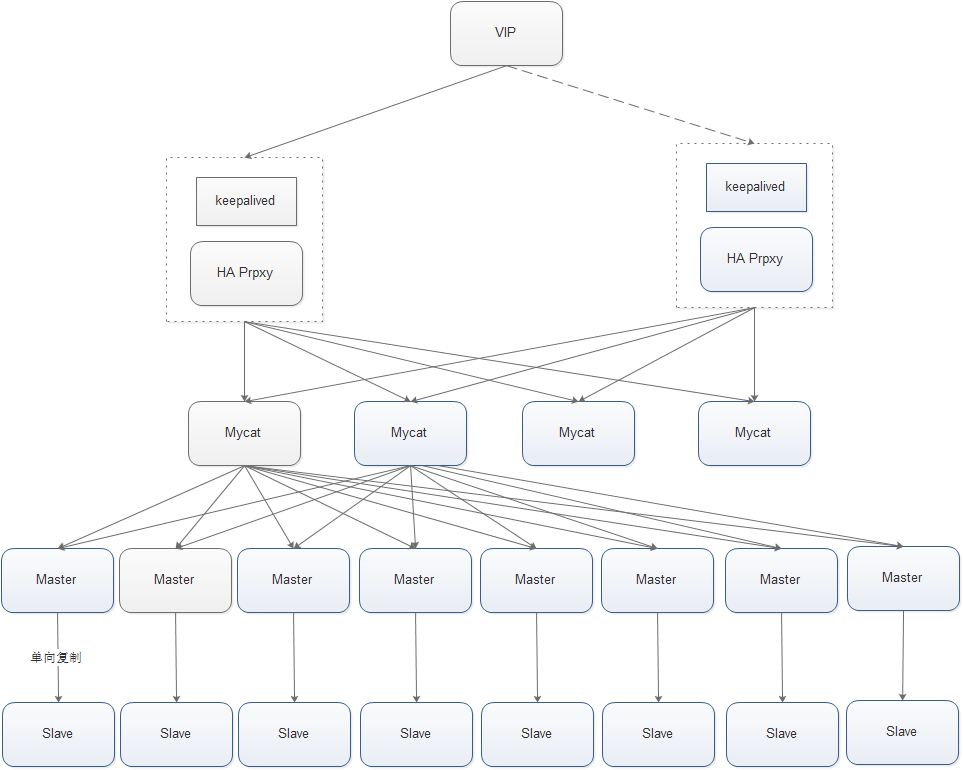
至少三节点；

### 4.4.4.Mycat

Mycat基于阿里开源的Cobar产品而研发，Cobar的稳定性、可靠性、优秀的架构和性能以及众多成熟的使用案例使得MYCAT一开始就拥有一个很好的起点，站在巨人的肩膀上，我们能看到更远。业界优秀的开源项目和创新思路被广泛融入到MYCAT的基因中，使得MYCAT在很多方面都领先于目前其他一些同类的开源项目，甚至超越某些商业产品。

MYCAT背后有一支强大的技术团队，其参与者都是5年以上资深软件工程师、架构师、DBA等，优秀的技术团队保证了MYCAT的产品质量。

常用高可用Mycat架构如下:



优点:

支持分库分表

支持水平扩容

支持全局ID

支持Galera for MySQL集群，Percona Cluster或者MariaDB cluster

基于Nio实现，有效管理线程，解决高并发问题。

缺点:

跨库数据库事务支持有限,在安全性要求较高的场景需慎重使用

跨库join使用有环境有限制,需结合手册使用

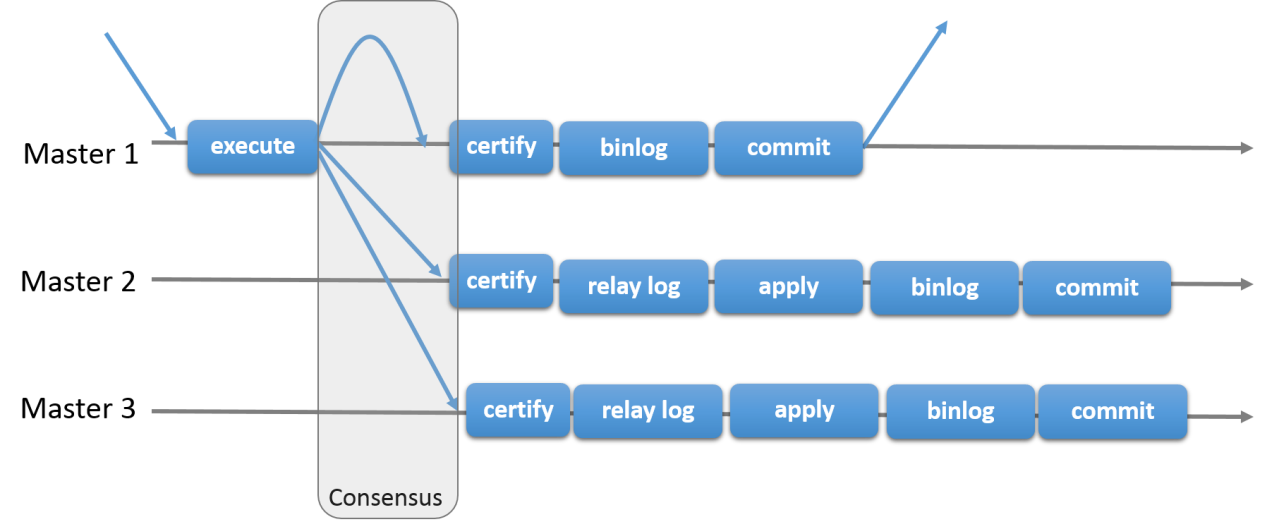
### 4.4.5.MySQL group replication

基于传统异步复制和半同步复制的缺陷——数据的一致性问题无法保证，MySQL官方在5.7.17版本正式推出组复制（MySQL Group Replication，简称MGR）。

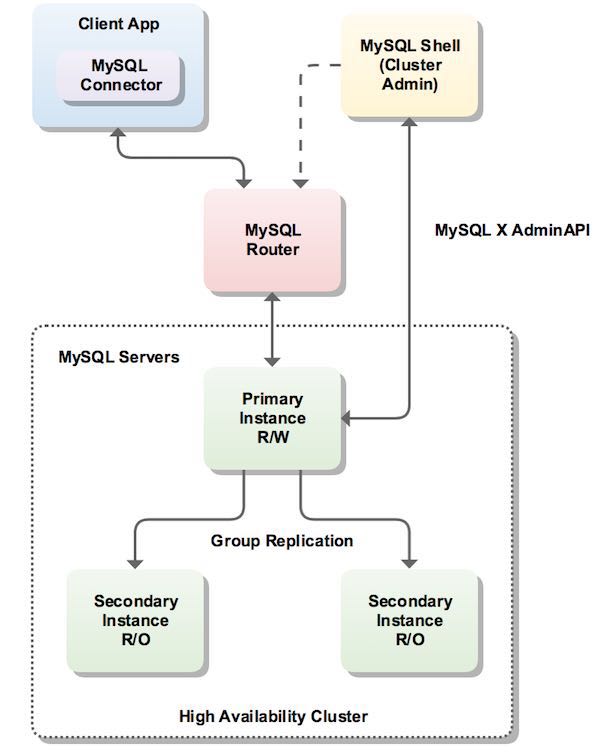
由若干个节点共同组成一个复制组，一个事务的提交，必须经过组内大多数节点（N / 2 + 1）决议并通过，才能得以提交。如上图所示，由3个节点组成一个复制组，Consensus层为一致性协议层，在事务提交过程中，发生组间通讯，由2个节点决议(certify)通过这个事务，事务才能够最终得以提交并响应。

引入组复制，主要是为了解决传统异步复制和半同步复制可能产生数据不一致的问题。组复制依靠分布式一致性协议(Paxos协议的变体)，实现了分布式下数据的最终一致性，提供了真正的数据高可用方案(是否真正高可用还有待商榷)。其提供的多写方案，给我们实现多活方案带来了希望。

一个复制组由若干个节点(数据库实例)组成，组内各个节点维护各自的数据副本(Share Nothing)，通过一致性协议实现原子消息和全局有序消息，来实现组内实例数据的一致。



架构图:



注:图片来自网络

优点:

数据一致性保证

事务并发冲突处理

节点故障自动检测

组成员自动管理

容错能力

GR基于分布式一致性算法实现，一个组允许部分节点挂掉，只要保证大多数节点仍然存活并且之间的通讯是没有问题的，那么这个组对外仍然能够提供服务。

缺点:

存储引擎必须为innodb

每个表必须提供主键

只支持ipv4，网络带宽要好

一个group最多只能有9个节点

不支持Replication event checksums，需要在my.cnf里面配置，在上节已经提及

不支持Savepoints

multi-primary mode部署方式不支持SERIALIZABLE事务隔离级别

multi-primary mode部署方式不能完全支持级联外键约束

multi-primary mode部署方式不支持在不同节点上对同一个数据库对象并发执行DDL(在不同节点上对同一行并发进行RW事务，后发起的事务会失败)

# 5.MySQL集群规划

## 5.1.集群规划目标

1. .快速搭建MySQL集群,满足业务需求,降低MySQL集群维护成本
2. .为不同规模项目,快速进行集群选型

## 5.2.集群规划方向

基于以上目标,对集群建设进行简单预设.

### 5.2.1.中小项目(千万级别数据量)

主要技术:Docker,MySQL(Master/slave)

### 5.2.2.大型项目(亿级别数据量)

选型1.Docker,MySQL Cluster

选型2.Docker,MySQL group replication

### 5.2.3.超大型项目(10亿+级别数据量)

主要技术:Docker,Mycat,MySQL5.7

# 6.实施计划与资源需求

# 7.潜在风险与问题

# 8.参考资料