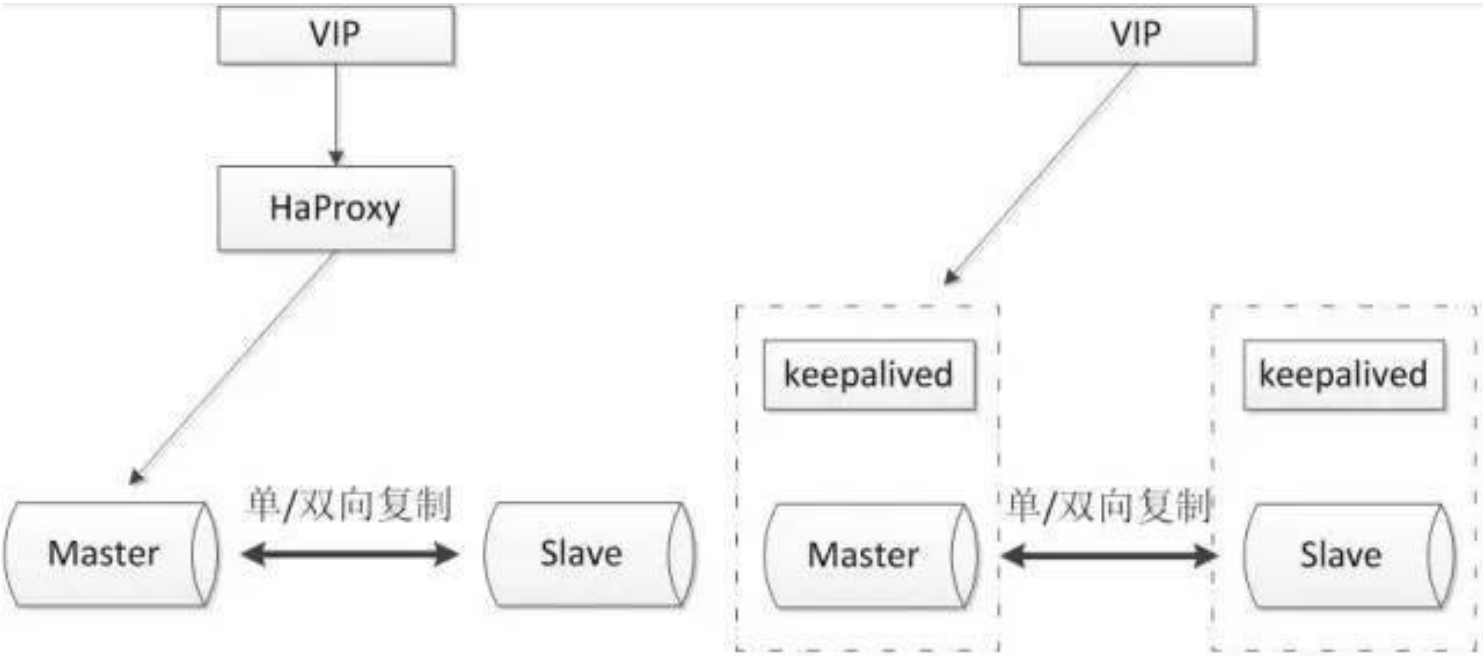


五大常见的MySQL高可用方案

UCloud...
已认证的官方帐号

1. 概述

我们在考虑MySQL数据库的高可用的架构时，主要要考虑如下几方面：

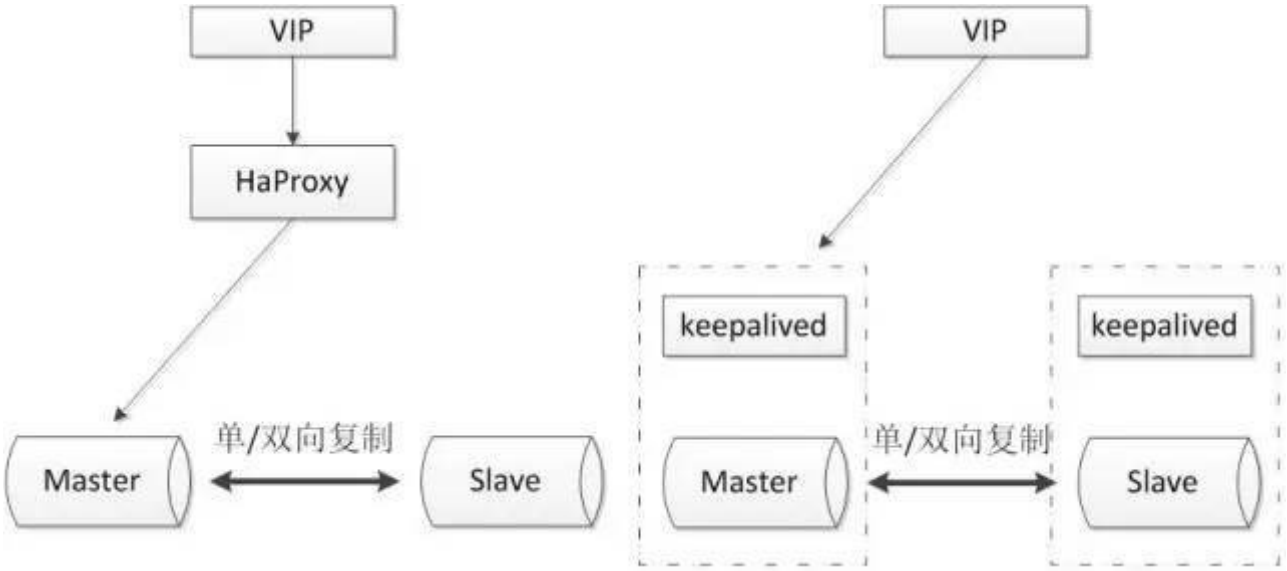
- 如果数据库发生了宕机或者意外中断等故障，能尽快恢复数据库的可用性，尽可能的减少停机时间，保证业务不会因为数据库的故障而中断。
- 用作备份、只读副本等功能的非主节点的数据应该和主节点的数据实时或者最终保持一致。
- 当业务发生数据库切换时，切换前后的数据库内容应当一致，不会因为数据缺失或者数据不一致而影响业务。

关于对高可用的分级在这里我们不做详细的讨论，这里只讨论常用高可用方案的优缺点以及高可用方案的选型。

2. 高可用方案

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lossless replication、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。

常见架构如下：



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。

优点：

1. 架构比较简单，使用原生半同步复制作为数据同步的依据；
2. 双节点，没有主机宕机后的选主问题，直接切换即可；
3. 双节点，需求资源少，部署简单；

缺点：

1. 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；
2. 需要额外考虑haproxy、keepalived的高可用机制。

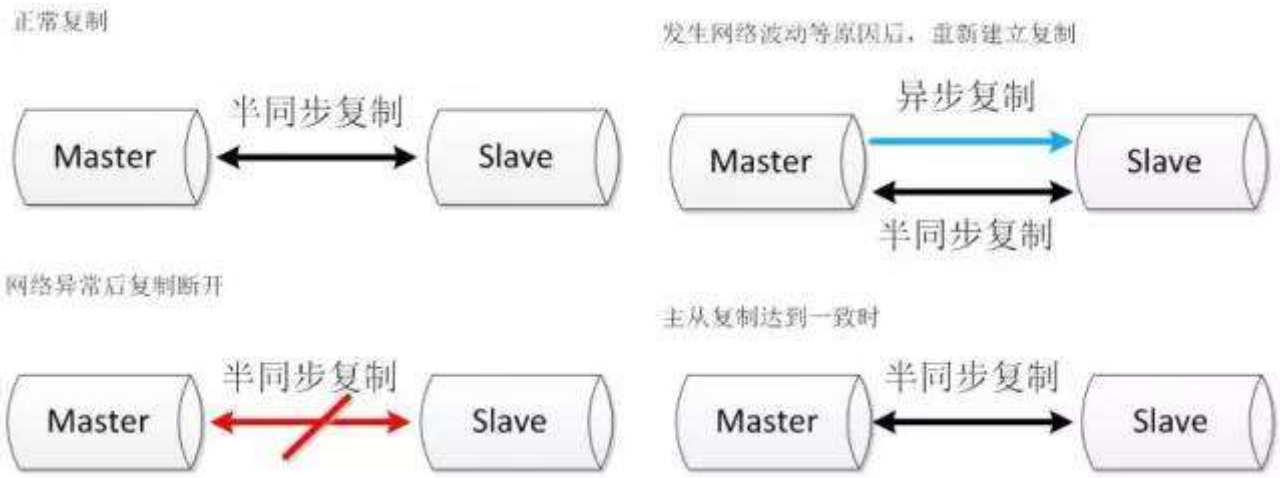
2.2. 半同步复制优化

据的一致性。所以尽可能的保证半同步复制，便可提高数据的一致性。

该方案同样使用双节点架构，但是在原有半同复制的基础上做了功能上的优化，使半同步复制的机制变得更加可靠。

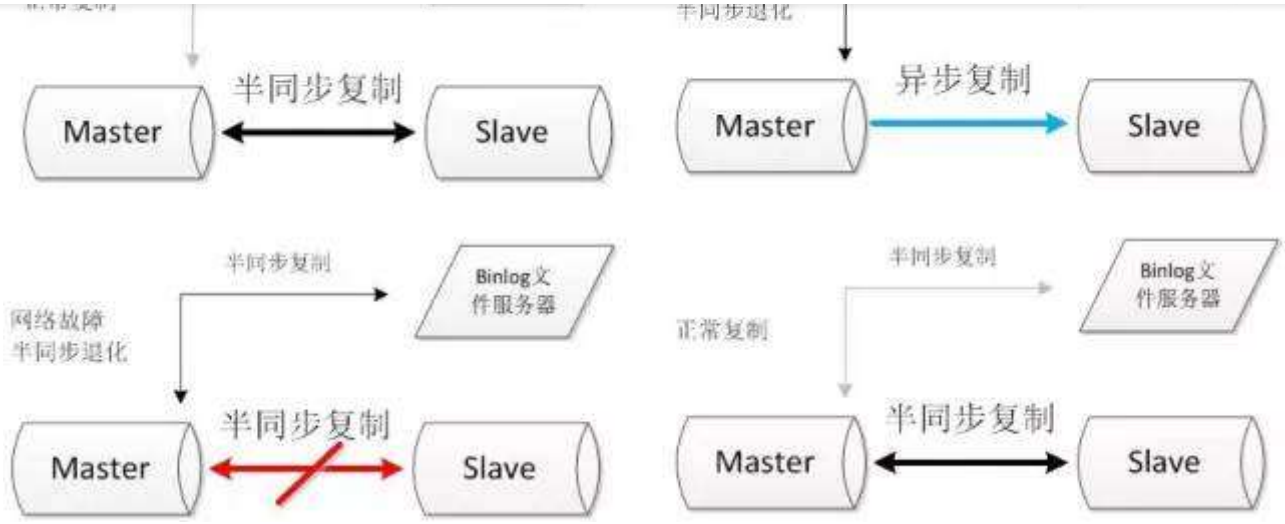
可参考的优化方案如下：

2.2.1. 双通道复制



半同步复制由于发生超时后，复制断开，当再次建立起复制时，同时建立两条通道，其中一条半同步复制通道从当前位置开始复制，保证从机知道当前主机执行的进度。另外一条异步复制通道开始追补从机落后的数据。当异步复制通道追赶到半同步复制的起始位置时，恢复半同步复制。

2.2.2. binlog文件服务器



搭建两条半同步复制通道，其中连接文件服务器的半同步通道正常情况下不启用，当主从的半同步复制发生网络问题退化后，启动与文件服务器的半同步复制通道。当主从半同步复制恢复后，关闭与文件服务器的半同步复制通道。

优点：

1. 双节点，需求资源少，部署简单；
2. 架构简单，没有选主的问题，直接切换即可；
3. 相比于原生复制，优化后的半同步复制更能保证数据的一致性。

缺点：

1. 需要修改内核源码或者使用mysql通信协议。需要对源码有一定的了解，并能做一定程度的二次开发。
2. 依旧依赖于半同步复制，没有从根本上解决数据一致性问题。

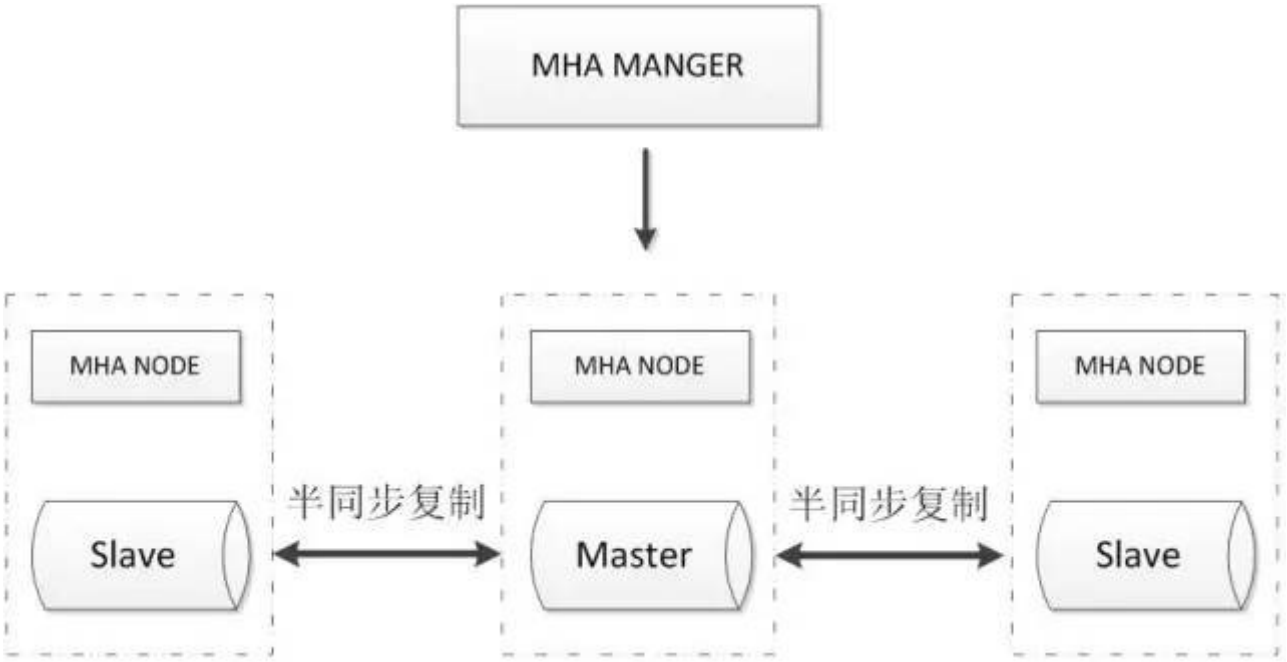
2.3. 高可用架构优化

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。

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：

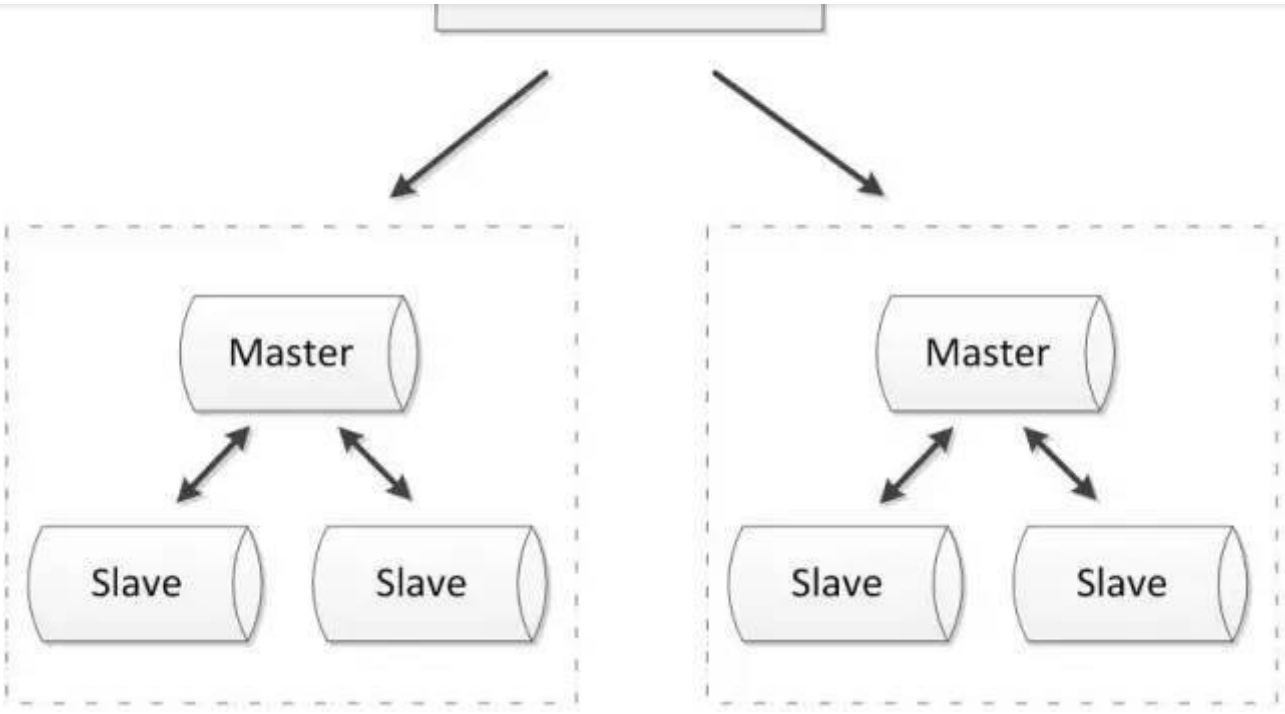
2.3.1. MHA+多节点集群



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：

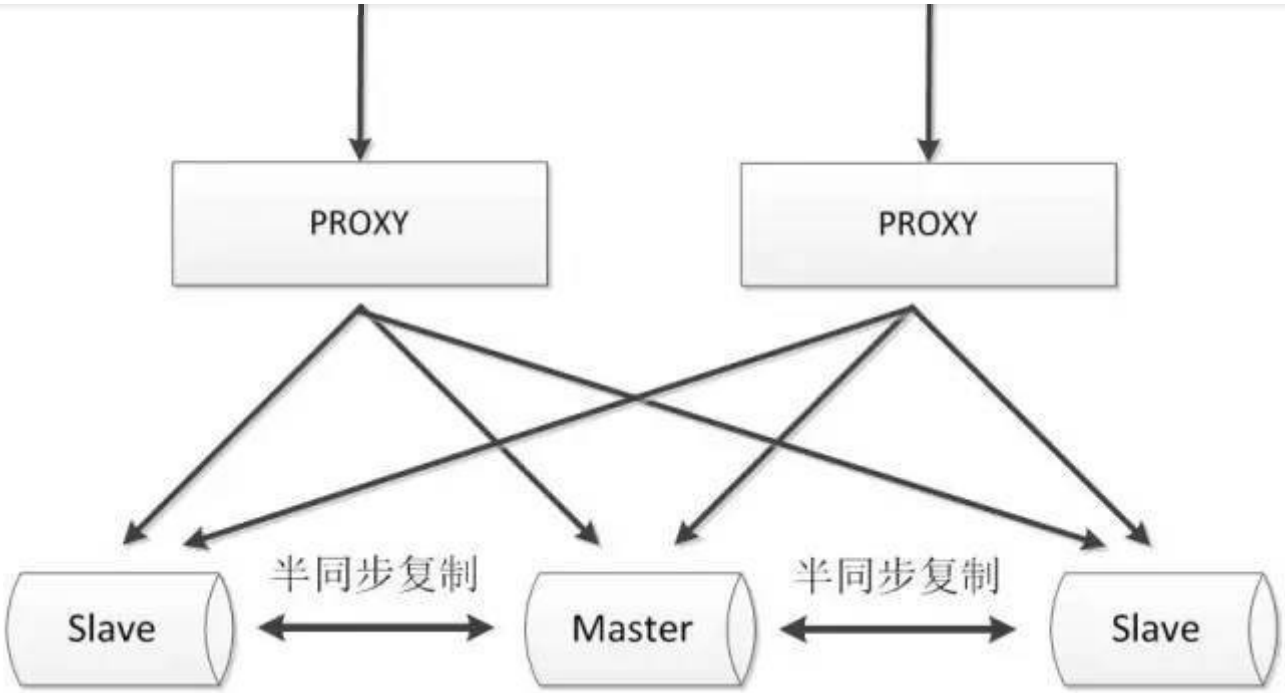
1. 可以进行故障的自动检测和转移;
2. 可扩展性较好，可以根据需要扩展MySQL的节点数量和结构;
3. 相比于双节点的MySQL复制，三节点/多节点的MySQL发生不可用的概率更低

缺点：

1. 至少需要三节点，相对于双节点需要更多的资源;
2. 逻辑较为复杂，发生故障后排查问题，定位问题更加困难;
3. 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;
4. 可能因为网络分区发生脑裂现象;

2.3.2. 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：

1. 较好的保证了整个系统的高可用性，包括proxy、MySQL;
2. 扩展性较好，可以扩展为大规模集群;

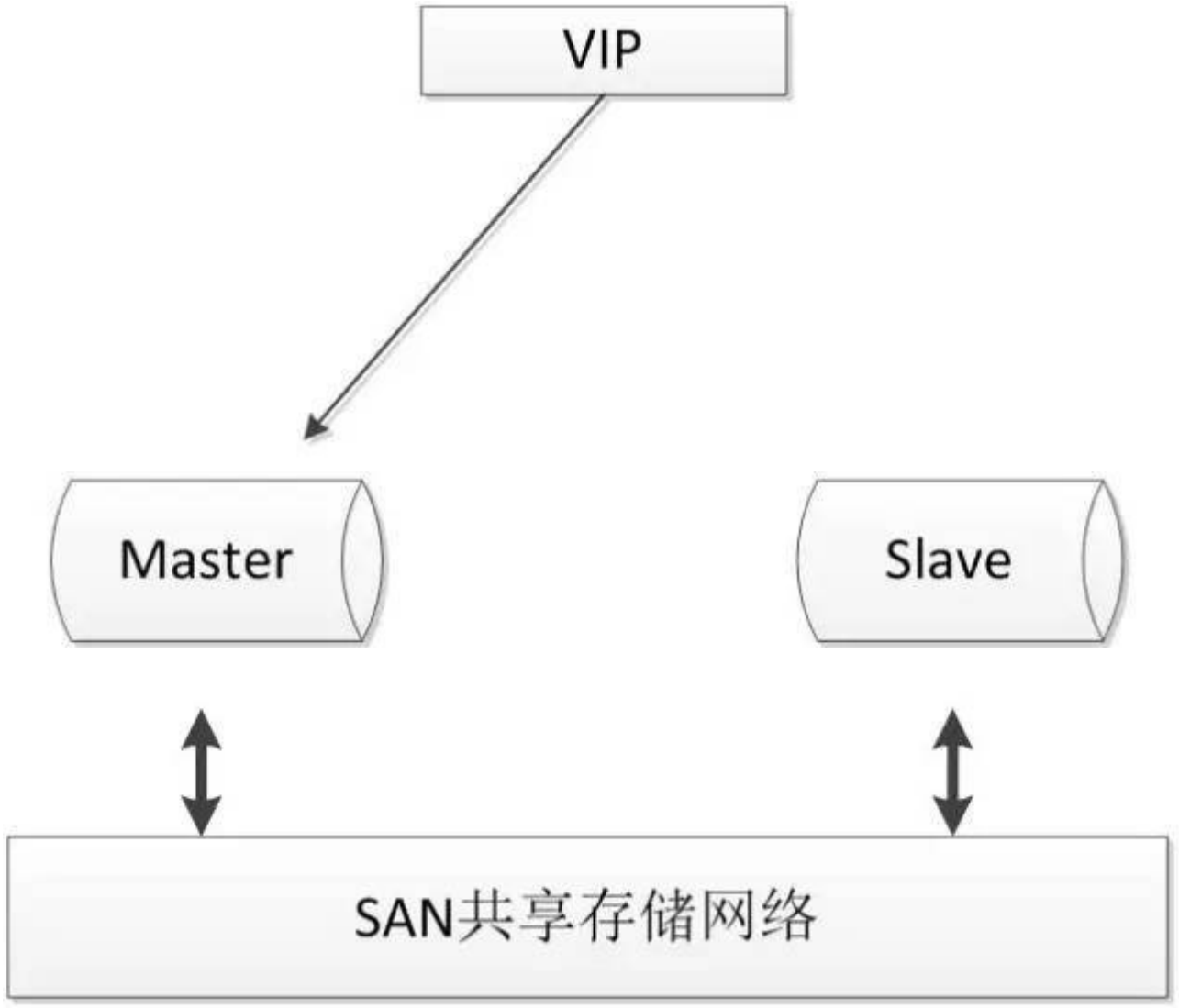
缺点：

1. 数据一致性仍然依赖于原生的mysql半同步复制;
2. 引入zk，整个系统的逻辑变得更加复杂;

2.4. 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。

2.4.1. SAN共享储存



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。

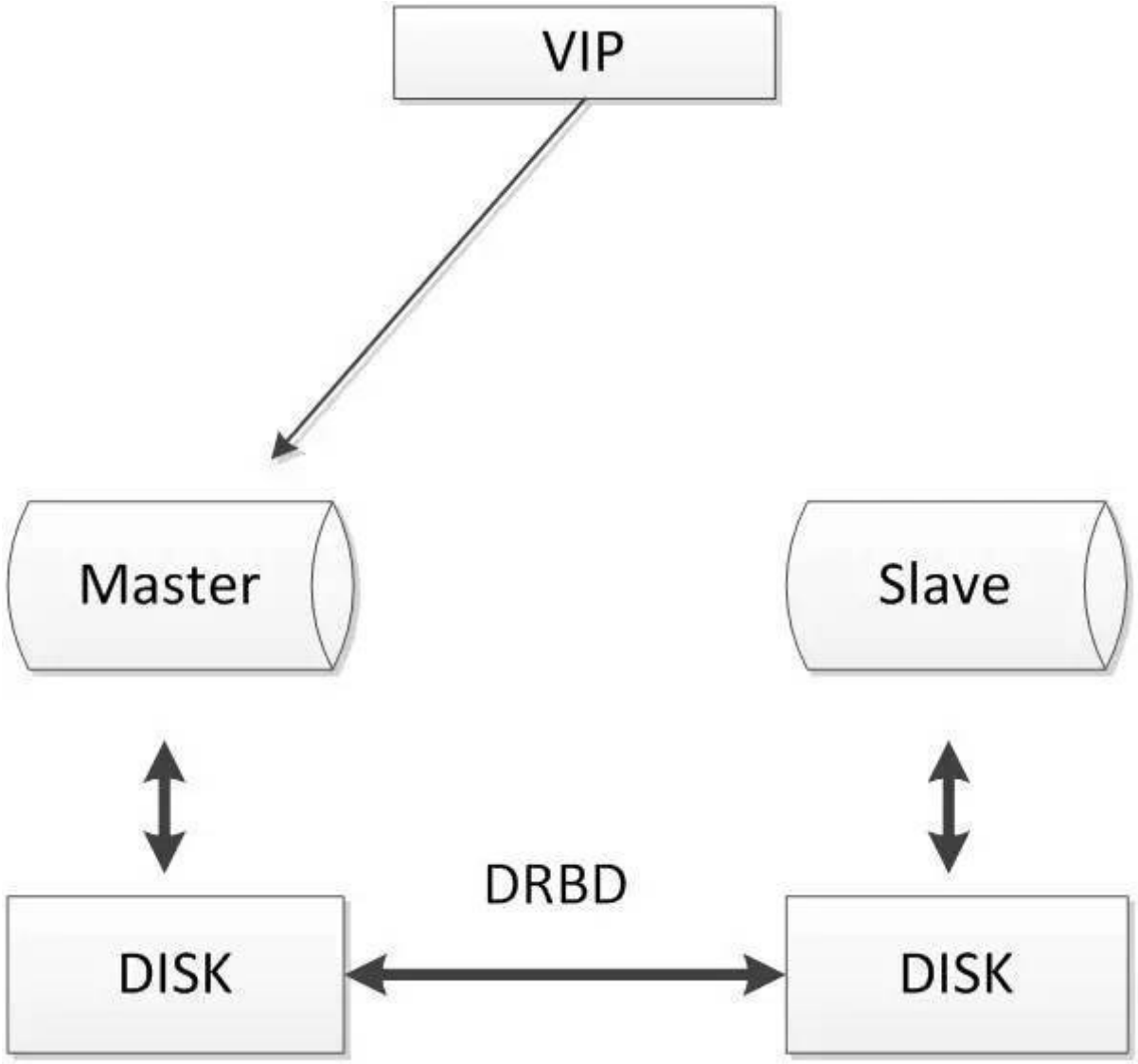
优点：

1. 两节点即可，部署简单，切换逻辑简单；
2. 很好的保证数据的强一致性；
3. 不会因为MySQL的逻辑错误发生数据不一致的情况；

1. 需要考虑共享存储的高可用；
2. 价格昂贵；

2.4.2. 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

优点：

1. 两节点即可，部署简单，切换逻辑简单；
2. 相比于SAN储存网络，价格低廉；
3. 保证数据的强一致性；

缺点：

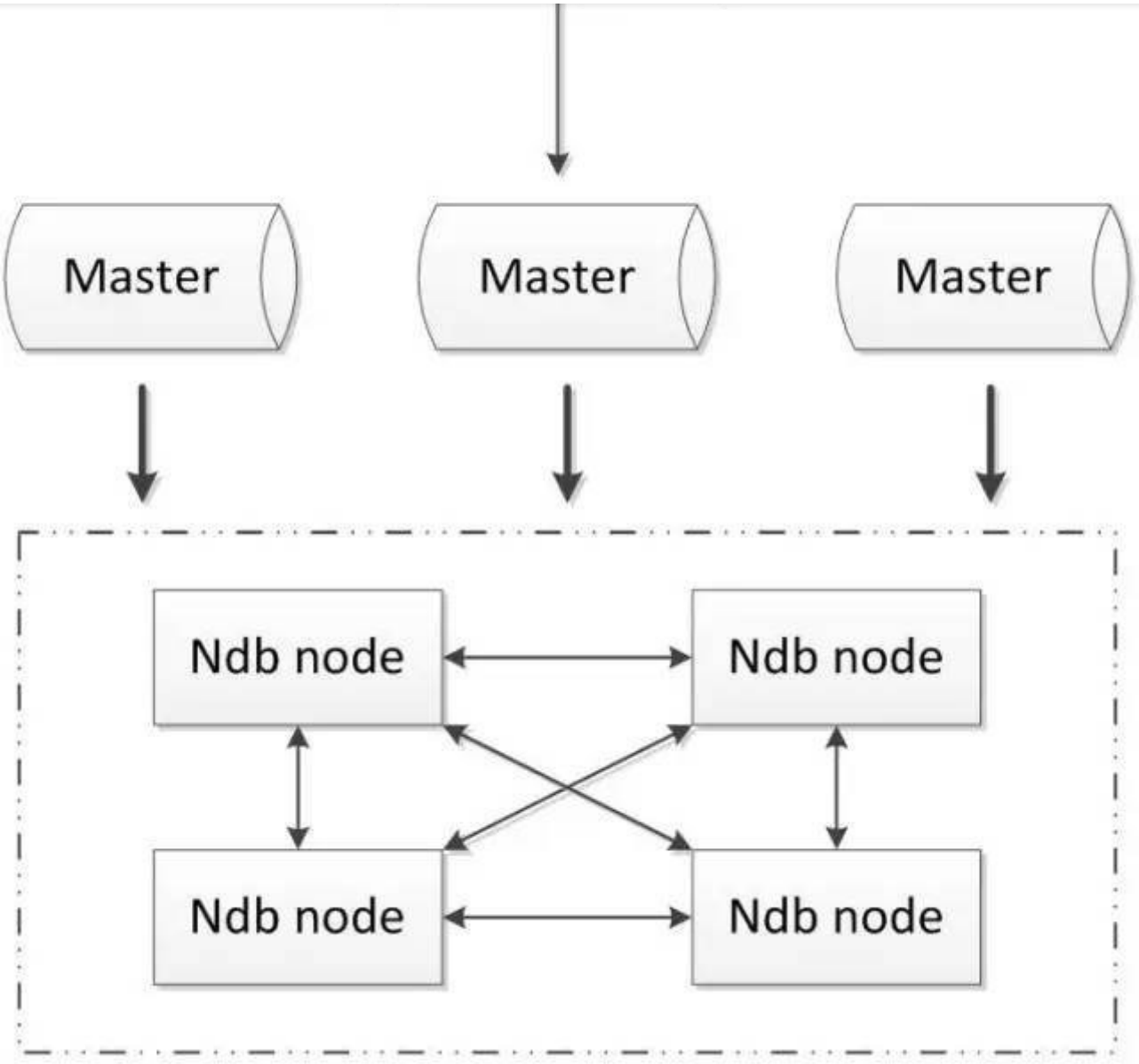
1. 对io性能影响较大；
2. 从库不提供读操作；

2.5. 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：

2.5.1. 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：

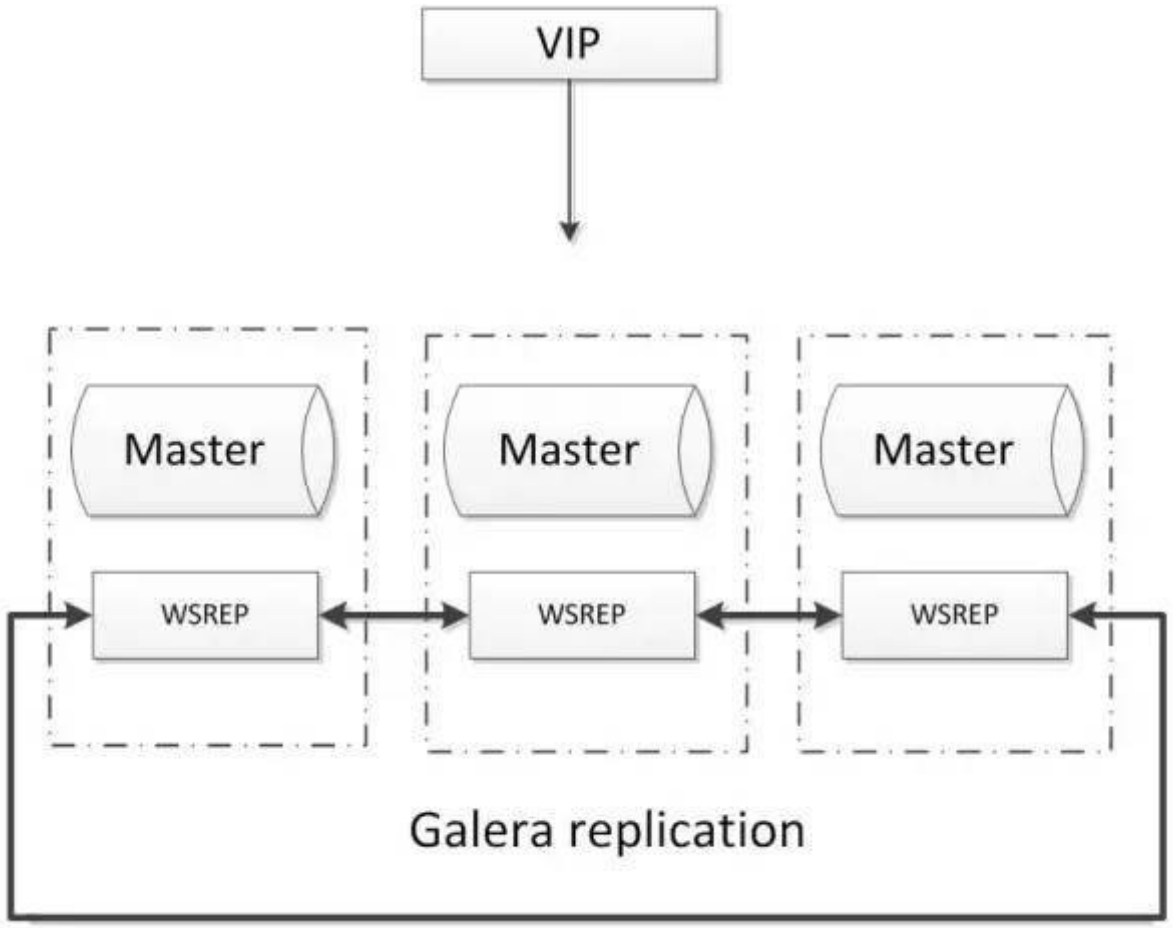
1. 全部使用官方组件，不依赖于第三方软件；
2. 可以实现数据的强一致性；

缺点：

1. 国内使用的较少；
2. 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；
3. 至少三节点；

2.5.2. 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：

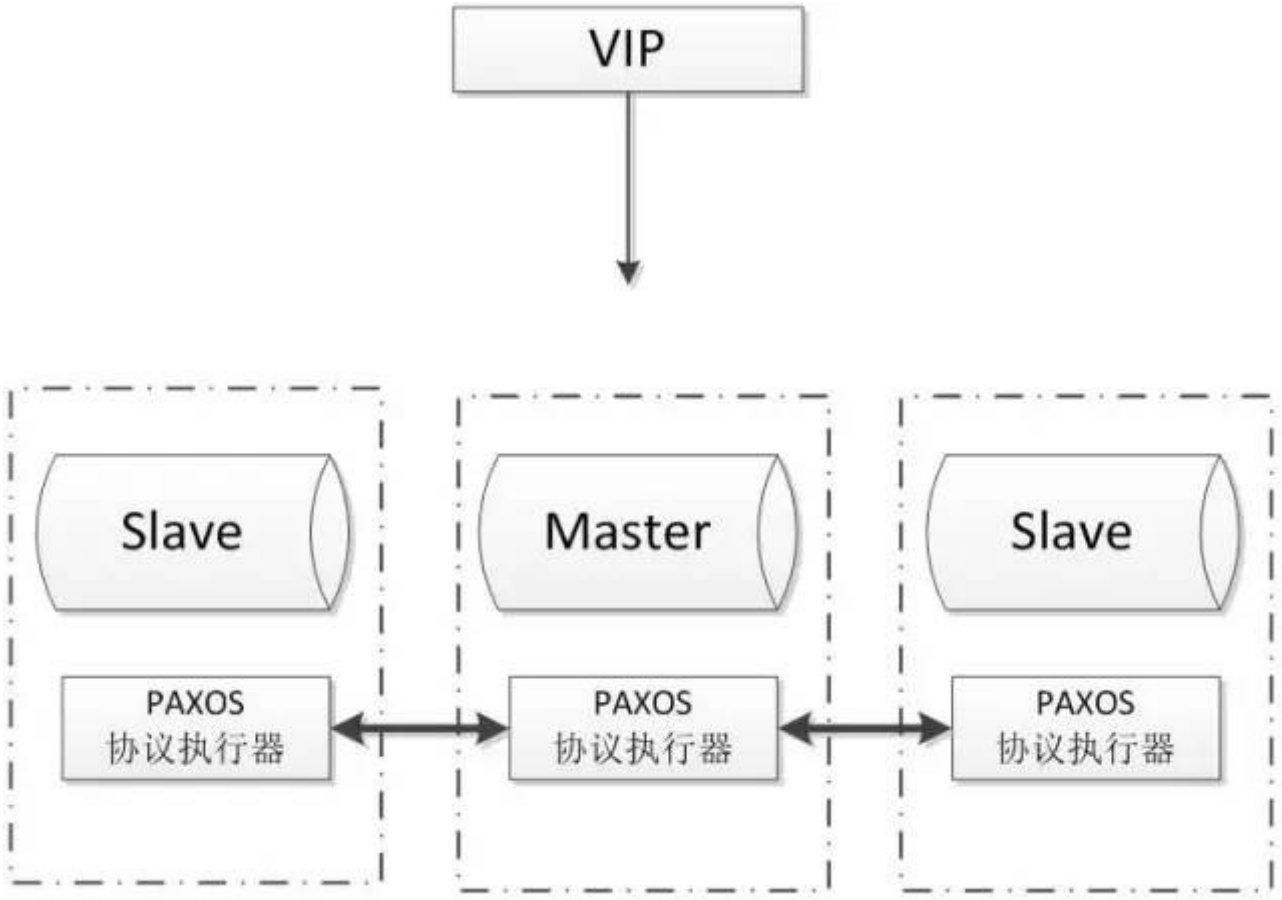
1. 多主写入，无延迟复制，能保证数据强一致性；
2. 有成熟的社区，有互联网公司在大规模的使用；
3. 自动故障转移，自动添加、剔除节点；

缺点：

3. 至少三节点；

2.5.3. 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：

1. 多主写入，无延迟复制，能保证数据强一致性；
2. 有成熟理论基础；
3. 自动故障转移，自动添加、剔除节点；

1. 只支持innodb储存引擎
2. 至少三节点；

3. 总结

随着人们对数据一致性的要求不断的提高，越来越多的方法被尝试用来解决分布式数据一致性的问题，如MySQL自身的优化、MySQL集群架构的优化、Paxos、Raft、2PC算法的引入等等。

而使用分布式算法用来解决MySQL数据库数据一致性的问题的方法，也越来越被人们所接受，一系列成熟的产品如PhxSQL、MariaDB Galera Cluster、Percona XtraDB Cluster等越来越多的被大规模使用。

随着官方MySQL Group Replication的GA，使用分布式协议来解决数据一致性问题已经成为了主流的方向。期望越来越多优秀的解决方案被提出，MySQL高可用问题可以被更好的解决。

参考文献

[2015 OTN]彭立勋-DoubleBinlog方案.pdf