Raft算法在选择主节点的过程中，也是通过多个节点之间的投票竞争。

说到这里，不得不说一下Raft算法的状态机。Raft算法为节点定义了三种角色：

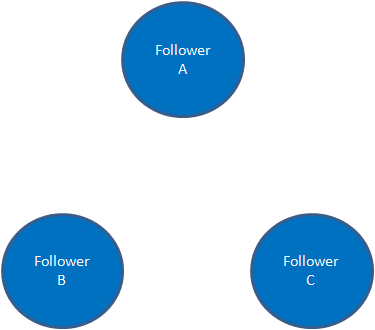
1.Leader（主节点）

2.Follower（从节点）

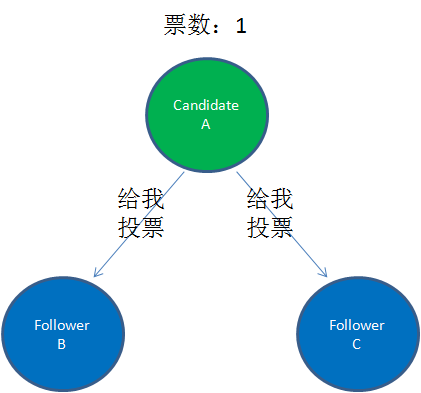
3.Candidate（参与投票竞争的节点）

让我们来看一看选主的具体流程：

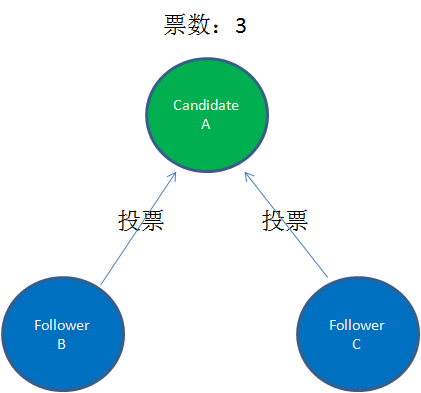
第一步，在最初，还没有一个主节点的时候，所有节点的身份都是Follower。每一个节点都有自己的计时器，当计时达到了超时时间（Election Timeout），该节点会转变为Candidate。



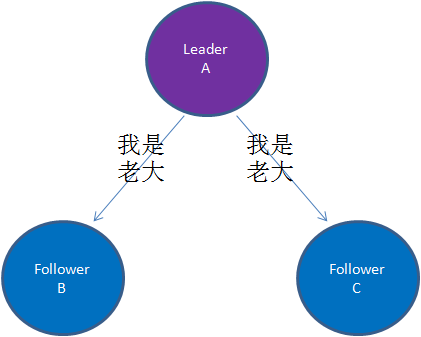
第二步，成为Candidate的节点，会首先给自己投票，然后向集群中其他所有的节点发起请求，要求大家都给自己投票。



第三步，其他收到投票请求且还未投票的Follower节点会向发起者投票，发起者收到反馈通知后，票数增加。



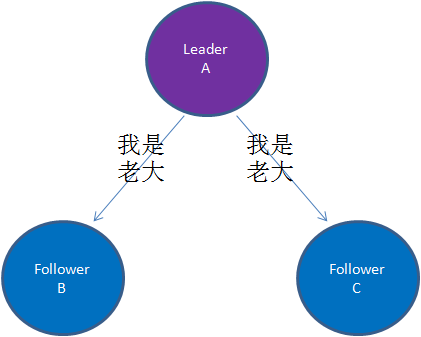
第四步，当得票数超过了集群节点数量的一半，该节点晋升为Leader节点。Leader节点会立刻向其他节点发出通知，告诉大家自己才是老大。收到通知的节点全部变为Follower，并且各自的计时器清零。



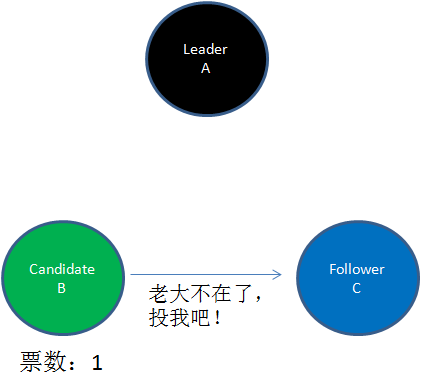
这里需要说明一点，每个节点的超时时间都是不一样的。比如A节点的超时时间是3秒，B节点的超时时间是5秒，C节点的超时时间是4秒。这样一来，A节点将会最先发起投票请求，而不是所有节点同时发起。

为什么这样设计呢？设想如果所有节点同时发起投票，必然会导致大家的票数差不多，形成僵局，谁也当不成老大。

那么，成为Leader的节点是否就坐稳了老大的位置呢？并不是。Leader节点需要每隔一段时间向集群其他节点发送心跳通知，表明你们的老大还活着。

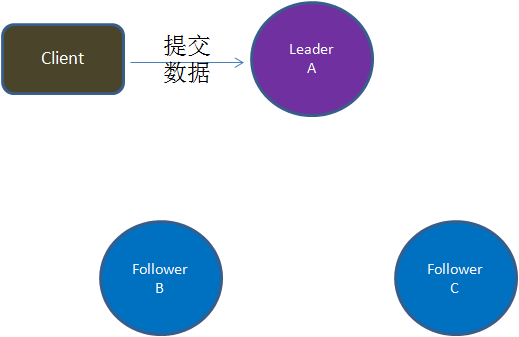


一旦Leader节点挂掉，发不出通知，那么计时达到了超时时间的Follower节点会转变为Candidate节点，发起选主投票，周而复始......

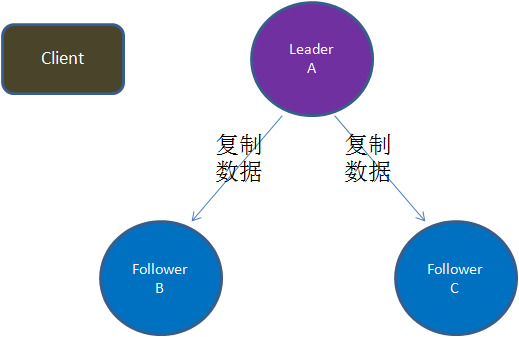


让我们来看一看数据同步的流程：

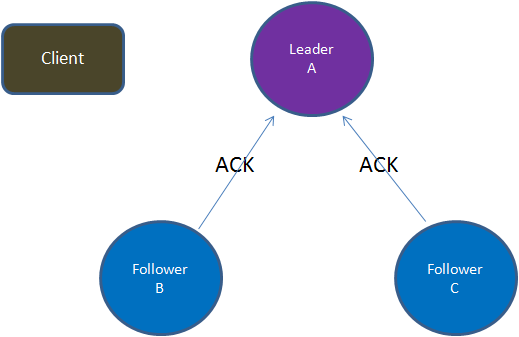
第一步，由客户端提交数据到Leader节点。



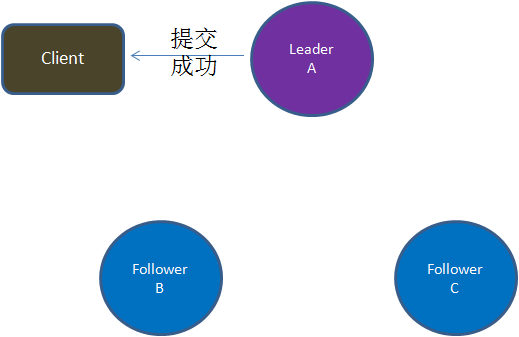
第二步，由Leader节点把数据复制到集群内所有的Follower节点。如果一次复制失败，会不断进行重试。



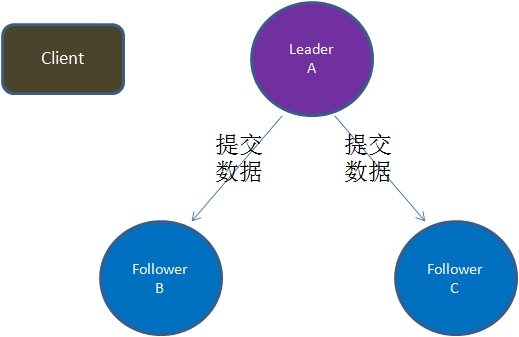
第三步，Follower节点们接收到复制的数据，会反馈给Leader节点。



第四步，如果Leader节点接收到超过半数的Follower反馈，表明复制成功。于是提交自己的数据，并通知客户端数据提交成功。



第五步，由Leader节点通知集群内所有的Follower节点提交数据，从而完成数据同步流程。



共识算法的应用场景？

Paxos 算法：

早期的共识算法，由拜占庭将军问题的提出者 Leslie Lamport 所发明。谷歌的分布式锁服务 Chubby 就是以 Paxos 算法为基础。

ZAB 算法：

Zookeeper 所使用的一致性算法，在流程上和 Raft 算法比较接近。

PBFT 算法：

区块链技术所使用的共识算法之一，适用于私有链的共识。

————————————————

版权声明：本文为CSDN博主「程序员小灰」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/bjweimengshu/article/details/80222416