启动instance

心跳

心跳

心跳

NM

NM

NM

CanalCenter-SLAVE

CanalCenter-Master

……

NodeN

Node2

Node1

运行instance的Node的集群

ZK集群

1：CanalCenter由多台机器组成,通过分布式锁控制 只有一台作为Master,其他的SLAVE作为”冷备份”,当Master挂掉以后,SLAVE抢到分布式锁以后,执行Master职责,CanalCenter启动成功以后需要向ZK注册,这样Node就可以知道Server的机器

2:每个Node上面有一个NodeManager进程,该进程主要有两个职责

2.1:向CanalCenter发送心跳,表名自己是一个活跃的结点

2.2:负责处理CanalCenter给他发送的Command(一般为启动instance,类似于MR中的Task)

HA设计

1：当instance挂掉以后,基于zookeeper的listener机制,可以监控到instance下线,CanalCenter会在其他的node上面启动instance继续处理

2:当Node挂掉以后,同样 基于zookeeper的listener机制,将这个Node上面的所有instance(Zookeeper上面获取)在其他机器启动

ZK的树形结构

橙色表示”瞬时”结点,生命周期和创建该结点的进程不同,进程挂了,该结点就消失了

绿色结点为”持久”结点,除非手动删除,否则会一直都在

/datacanal/lock/ 该目录是分布式锁目录

/datacanal/lock/serverlock/ 这是canal server启动用到的分布式锁的

/datacanal/canal\_server/{server addr} 这边是存储服务端的IP和端口

/datacanal/node/{noden} 这边是存储所有的”运行结点的信息”

/datacanal/task 这边是存储所有的抽取任务

/datacanal/task/logictable 任务的逻辑表

/datacanal/task/logictable/physicstable 逻辑表的分片信息(物理表)

/datacanal/task/logictable/physicstable/{instance} 处理该分片的instance的信息

/datacanal/task/logictable/physicstable/{ status } canal server控制instance的运行