**Mapred-env.sh，hadoop-env.sh，yarn-env.sh:**

配置JAVA\_HOME

**Slave:**

Ha1

Ha2

Ha3

**Core-site.xml:**

<!-- 指定hdfs的nameservice服务名为dj（自己在hdfs-site.xml定义） -->

<property>

<name>fs.defaultFS</name>

<value>hdfs://dj</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/opt/module/hadoop/data</value>

</property>

<!-- hdfs指定连接zookeeper集群的地址和端口 -->

<property>

<name>ha.zookeeper.quorum</name>

<value>ha1:2181,ha2:2181,ha3:2181</value>

</property>

**Hdfs-site.xml:**

<!-- 定义hdfs集群的名称为dj -->

<property>

<name>dfs.nameservices</name>

<value>dj</value>

</property>

<!-- 定义hdfs集群dj中的namenode节点nn1，nn2 -->

<property>

<name>dfs.ha.namenodes.dj</name>

<value>nn1,nn2</value>

</property>

<!-- 定义namenode的主机名和rcp通信地址的端口 -->

<property>

<name>dfs.namenode.rpc-address.dj.nn1</name>

<value>ha1:9000</value>

</property>

<property>

<name>dfs.namenode.rpc-address.dj.nn2</name>

<value>ha2:9000</value>

</property>

<!-- 指定NameNode的元数据在JournalNode上的存放位置  -->

<property>

<name>dfs.namenode.shared.edits.dir</name>

<value>qjournal://ha1:8485;ha2:8485;ha3:8485/dj</value>

</property>

<!-- 指定JournalNode在本地磁盘存放数据的位置 -->

<property>

<name>dfs.journalnode.edits.dir</name>

<value>/opt/module/hadoop/jounaldata</value>

</property>

<!-- hdfs集群中两个namenode切换状态时的隔离方法为ssh -->

<property>

<name>dfs.ha.fencing.methods</name>

<value>

sshfence

shell(/bin/true) (必要，否则两个namenode不能切换状态)

</value>

</property>

<!-- 两个namenode切换状态时的隔离方法的秘钥文件路径 -->

<property>

<name>dfs.ha.fencing.ssh.private-key-files</name>

<value>/home/pt/.ssh/id\_rsa</value>

</property>

<!-- ha的hdfs集群自动切换namenode的开关，开启自动故障转移 -->

<property>

<name>dfs.ha.automatic-failover.enabled</name>

<value>true</value>

</property>

<!--是否开启hdfs的权限---必要-->

<property>

<name>dfs.permissions.enabled</name>

<value>true</value>

</property>

<property>

       <name>dfs.client.failover.proxy.provider.dj</name>

     <value>org.apache.hadoop.hdfs.server.namenode.ha.

ConfiguredFailoverProxyProvider</value>

</property>

**mapred-site.xml:**

<!-- 指定mr框架为yarn方式 -->

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

**Yarn-site.xml:**

<!-- 开启RM高可用 -->  
<property>  
   <name>yarn.resourcemanager.ha.enabled</name>  
   <value>true</value>  
</property>

<!-- 给yarn集群起名字 -->

<property>

<name>yarn.resourcemanager.cluster-id</name>

<value>cluster1</value>

</property>

<!-- 定义yarn集群中rm的节点 -->

<property>

<name>yarn.resourcemanager.ha.rm-ids</name>

<value>rm1,rm2</value>

</property>

<!-- rm1节点的主机名 -->

<property>

<name>yarn.resourcemanager.hostname.rm1</name>

<value>ha1</value>

</property>

<!-- rm2节点的主机名 -->

<property>

<name>yarn.resourcemanager.hostname.rm2</name>

<value>ha2</value>

</property>

<!-- 指定zk集群地址 -->

<property>

<name>yarn.resourcemanager.zk-address</name>

<value>ha1:2181,ha2:2181,ha3:2181</value>

</property>

<!-- nodemanager配置 -->

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

</property>

先启动zookeeper，和journalnode

然后hdfs namenode -format

第一个节点启动 hadoop-daemon.sh start namenode

第二个节点hdfs namenode -bootstrapStandby同步

第二个节点开启 hadoop-daemon.sh start namenode

在其中一个namenode上初始化hdfs zkfc -formatZK

停止dfs服务

然后启动所有服务

测试两个namenode是否会自动切换

**Hadoop集群节点的动态增删：**

**Hdfs-site.xml中添加：**

<property>

<name>dfs.hosts</name> <value>/home/hadoop/hadoop2.7/conf/datanode-allow.list</value>

<description>允许加入集群的节点列表</description>

</property>

<property>

<name>dfs.hosts.exclude</name>

<value>/home/hadoop/hadoop2.7/conf/datanode-deny.list</value>

<description>拒绝加入集群的节点列表</description>

</property>

**1.动态添加节点：**

Hdfs集群运行时，添加一台机器，在所有节点的hosts文件，添加该节点的ip和主机名，配置该节点与namenode节点的ssh，传递hadoop和jdk，需要删除DataNode不需要的文件，如：journaldata，data目录的数据

配置namenode的slaves文件，添加该节点的主机名，传递slaves文件到各个节点

配置允许连接的节点列表datanode-allow.list //非必需（默认所有节点都可以连接，如果进行配置的话，只有此文件中的节点能够连接。）

Namennode节点刷新节点：hdfs dfsadmin -refreshNodes

去增加的节点开启DataNode，用hadoop-deamon.sh start datanode开启

Namenode节点查看：hdfs dfsadmin -report 看是否出现新增节点

平衡各个节点中数据块的大小：/start-balancer.sh -threshold 5（**hadoop-daemon.sh start tasktracker代替这行命令**）

启动nodeManager，yarn-daemon.sh start nodemanager,通过yarn node -list 查看所有node节点

成功！

**动态删除节点：**

在主节点中配置拒绝连接的节点列表datanode-deny-list，将删除的节点名添加在列表中。

vi /home/hadoop/hadoop2.7/con/datanode-denylist

刷新节点：hdfs dfsadmin -refreshNodes

查看状态：hdfs dfsadmin -report

该节点的datanode状态是否转为Decommissioned，正常是（Narmal）

手动停止已经下线的datanode节点 hadoop-deamon.sh stop datanode，

删除下线节点的hosts

删除slaves中下线节点的hosts

刷新：hdfs dfsadmin -refreshNodes

**4. 上线节点**

上线节点基本与下线相同

1) slaves文件里每行添加一个上线服务器名，同时保证dfs.exclude文件为空。

2) ./hdfs dfsadmin refreshNodes

3) 在要上线的节点上启动datanode：

./hadoop-daemon.sh start datanode

4) 如需要启动nodemanager,则执行：

./yarn-daemon.sh start nodemanager

1. 修改slaves，添加上线节点的hosts

Yarn增删节点：

resourcemanager节点上yarn.exclude文件里添加下线节点主机名

添加（或修改）如下配置项到mapred-site.xml

增加节点：

<property>

<name>mapred.hosts</name>

<value>yarn.host路径</value>

</property>

删除节点：

<property>

<name>mapred.hosts.exclude</name>

<value>yarn.exclude路径</value>

</property>

resourcemanager节点上刷新：yarn rmadmin -refreshNodes

yarn node -list #查看

成功！