# 七月在线Hadoop+HBase+Spark+Hive集群搭建教程

## 1、准备安装包

链接：系统镜像和各种大数据软件

密码：n2cn

此教程所需的安装包链接如上，安装包的软件不是最新的。
如果需要最新的软件，请访问以下地址下载：
apache所有软件清单 http://archive.apache.org/dist/

但不保证新安装包能不能适用此教程。

## 2、准备Ubuntu系统

Hadoop等大数据开源框架是不支持Windows系统的，所以需要先安装一个Linux双系统。当然，如果你有一台单独的电脑用来安装Ubuntu系统，就不需要安装双系统了。

本教程使用的是阿里云的服务器，下面是本教程三台服务器的信息。

spark001：**作为master主节点**

- 地址
  - ssh root@39.106.49.230
- 内网IP
  - 192.168.57.180

spark002： **作为slave1从节点**

- 地址
  - ssh root@39.106.47.28
- 内网IP
  - 192.168.57.181

spark003： **作为slave2从节点**

- 地址
  - ssh root@39.106.67.35
- 内网IP
  - 192.168.57.182

**以下教程是在Mac本上进行的操作。**

# 3、更新源

先新建三个终端，登陆三台服务器。把其中一台作为主服务器（master主节点节点）。

更新源操作在只要主服务器上更新就可以了，这里选择spark001作为master主节点。

1. 更新源

```
#root@spark001:~# 是其中一台服务器的名字
root@spark001:~# apt-get update
```

2. 安装vim编译器

```
root@spark001:~# apt-get install vim
```

3. 备份原始的官方源

```
root@spark001:~# cp /etc/apt/sources.list /etc/apt/sources.list.bak
```

4. 删除原始的官方源

```
root@spark001:~# rm /etc/apt/sources.list
```

5. 运行如下shell命令，重新创建sources.list文件

```
root@spark001:~# vim /etc/apt/sources.list
```

6. 按 i 进入vim的编辑模式，复制下面的阿里源到sources.list文件中，然后按 esc 退出编辑模式，最后输入:wq ，按回车保存，后面关于保存退出，不再说明。

```
## Note, this file is written by cloud-init on first boot of an instance
## modifications made here will not survive a re-bundle.
## if you wish to make changes you can:
## a.) add 'apt_preserve_sources_list: true' to /etc/cloud/cloud.cfg
##     or do the same in user-data
## b.) add sources in /etc/apt/sources.list.d
## c.) make changes to template file /etc/cloud/templates/sources.list.tmpl

# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.
deb http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial main
deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial main

## Major bug fix updates produced after the final release of the
## distribution.
deb http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial-updates main
deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial-updates main

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team. Also, please note that software in universe WILL NOT receive any
## review or updates from the Ubuntu security team.
deb http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial universe
deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial universe
deb http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial-updates universe
deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ xenial-updates universe

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
```

7. 运行如下shell命令，完成源的更新

```
root@spark001:~# apt-get update
```

# 4、安装JAVA环境(每台服务器都要配置)

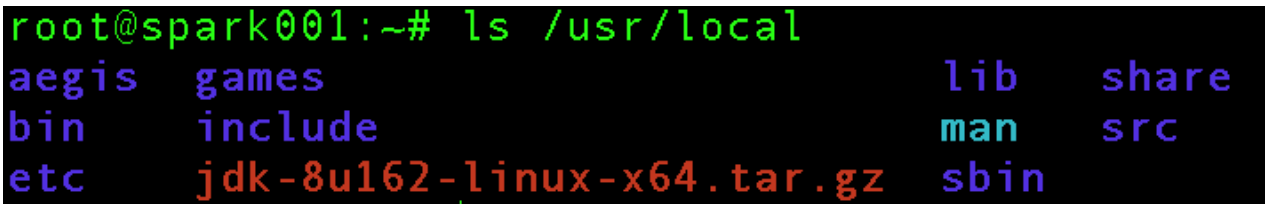
1. Java环境推荐使用 Oracle 的 JDK，首先，准备好文件 jdk-8u162-linux-x64.tar.gz(这里说下，以下所有的安装文件都在百度网盘内)，然后将文件移到/usr/local目录下：

**这里需要再另外新建个终端**，用如下命令传输JDK文件到一台服务器上。windows系统可以用其他的方式进行传输。这里注意需要传输到/usr/local文件目录下。

```
#这里本机文件的地址和服务器的地址需要改下自己的
#yygdeMac-mini:~ yyg$ 是我电脑的名字
yygdeMac-mini:~ yyg$ scp /Users/yyg/Downloads/大数据之路-安装包/软件安装/jdk-8u162-linux-x64.tar.gz root@39.106.49.230:/usr/local
```

回到服务器命令下，查看下安装包是否传输正确。

```
root@spark001:~# ls /usr/local
```




2. 解压安装包

```
root@spark001:~# cd /usr/local
root@spark001:/usr/local# tar -zxvf jdk-8u162-linux-x64.tar.gz
```

3. 重命名文件夹为java

```
root@spark001:/usr/local# mv jdk1.8.0_162 java
```

4. 用vim打开/etc/profile文件（Linux下配置系统环境变量的文件）

```
root@spark001:/usr/local# vim /etc/profile
```

5. 按i进入编辑模式，在文件末尾添加如下JAVA环境变量。

```
export JAVA_HOME=/usr/local/java
export JRE_HOME=/usr/local/java/jre
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JRE_HOME/lib
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
```

6. 添加环境变量后，结果如下图所示，按 esc 退出编辑模式，然后输入:wq ，按回车保存。

```
if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
  unset i
fi

export JAVA_HOME=/usr/local/java
export JRE_HOME=/usr/local/java/jre
export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JRE_HOME/lib
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
~
```

7. 最后，需要让该环境变量生效

```
root@spark001:/usr/local# source /etc/profile
```

8. 检验JAVA是否安装成功

```
#如果设置正确的话，java -version 会输出 java 的版本信息，java 和 javac 会输出命令的使用指导。
root@spark001:/usr/local# echo $JAVA_HOME     # 检验变量值
root@spark001:/usr/local# java -version
root@spark001:/usr/local# java
root@spark001:/usr/local# javac
```

如果设置正确的话，java -version 会输出 java 的版本信息，java 和 javac 会输出命令的使用指导。

```
root@spark001:/usr/local# java -version
java version "1.8.0_162"
Java(TM) SE Runtime Environment (build 1.8.0_162-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.162-b12, mixed mode)
root@spark001:/usr/local#
```

9. **按照如上方法，配置另外两台服务器的JAVA环境，操作步骤一模一样。**

# 5、安装Hadoop

**以下配置方式同JDK的步骤。**

这里到本机终端下传输文件，同JDk传输文件方式一样，**这里要注意的是只在master主节点（spark001服务器作为主节点）上配置就可以了。**

1. 传输文件

```
#这里本机文件的地址和服务器的地址需要改下自己的
#yygdeMac-mini:~ yyg$ 是我电脑的名字
yygdeMac-mini:~ yyg$ scp /Users/yyg/Downloads/大数据之路-安装包/软件安装/hadoop-2.7.6.tar.gz root@39.106.49.230:/usr/local
```

2. 解压、重命名、配置环境变量、查看版本

```
#解压
root@spark001:/usr/local# tar -zxvf hadoop-2.7.6.tar.gz

#文件夹重命名为hadoop
root@spark001:/usr/local# mv hadoop-2.7.6 hadoop

#配置环境变量，打开文件/etc/profile，文件末尾添加如下两行Hadoop环境变量
root@spark001:/usr/local# vim /etc/profile
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:/usr/local/hadoop/bin:/usr/local/hadoop/sbin

#同样，需要让该环境变量生效，执行如下代码：
root@spark001:/usr/local# source /etc/profile

#输入如下命令来检查 Hadoop 是否可用，成功则会显示 Hadoop 版本信息：
root@spark001:/usr/local# hadoop version
```

# 6、Hadoop集群配置

## 6.1、配置hosts文件(每台服务器都要配置)

**6.1节三台服务器都要配置。**

1. 修改主机名

```
#主节点spark001修改为master，从节点spark002、spark003修改为slave1、slave2等等。
root@spark001:/usr/local# vim /etc/hostname

root@spark002:/usr/local# vim /etc/hostname

root@spark003:/usr/local# vim /etc/hostname
```

2. 再次修改主机名

   上面文件里修改不够（不知道原因），需要命令行再次修改，运行下面命令。

```
#这里用hostname指令修改主机名字，主节点spark001修改为master，从节点spark002、spark003修改为slave1、slave2等等。
root@spark001:/usr/local# hostname master
root@spark001:/usr/local# hostname

root@spark002:/usr/local# hostname slave1
root@spark002:/usr/local# hostname

root@spark003:/usr/local# hostname slave2
root@spark003:/usr/local# hostname
```

2. 退出并重启服务器查看主机名是否修改成功

   **其他两台服务器同样操作， 查看主机名是否修改成功。**

```
root@master:~#
```

3. 编辑文件/etc/hosts

   **三台服务器都要复制**

```
root@master:~# vim /etc/hosts
#将以下数据修改并复制，三台服务器都要复制，192.168.57.XXX是三台服务器的私有内网ip地址。
#这里提醒下，登陆服务器用的是公网ip地址，这里用的是内网ip地址。
#下面是我的内网ip地址，每个人的服务器不一样，需要自己修改。
192.168.57.180  master  master
192.168.57.181  slave1  slave1
192.168.57.182  slave2  slave2
```

```
127.0.0.1       localhost


# The following lines are desirable for IPv6 capable hosts
::1     localhost       ip6-localhost   ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
127.0.1.1       iZuf6h1kfgutxc3el68z2lZ iZuf6h1kfgutxc3el68z2lZ

192.168.57.180  master  master
192.168.57.181  slave1  slave1
192.168.57.182  slave2  slave2
```

4. 使用以下指令在master主机中进行测试，可使用类似指令在slave1上测试：

下面输出说明连接成功，会一直不停的输出，可以用命令ctrl+c打断，如果ping能连通，说明网络连接正常，否则请检查网络连接或者IP信息是否正确。

```
root@master:~# ping slave1
PING slave1 (192.168.57.181) 56(84) bytes of data.
64 bytes from slave1 (192.168.57.181): icmp_seq=1 ttl=64 time=0.180 ms
64 bytes from slave1 (192.168.57.181): icmp_seq=2 ttl=64 time=0.157 ms
64 bytes from slave1 (192.168.57.181): icmp_seq=3 ttl=64 time=0.172 ms
64 bytes from slave1 (192.168.57.181): icmp_seq=4 ttl=64 time=0.190 ms
64 bytes from slave1 (192.168.57.181): icmp_seq=5 ttl=64 time=0.179 ms
64 bytes from slave1 (192.168.57.181): icmp_seq=6 ttl=64 time=0.168 ms
64 bytes from slave1 (192.168.57.181): icmp_seq=7 ttl=64 time=0.168 ms
```

## 6.2、SSH无密码登陆节点（master上配置）

**配置ssh服务器**

1. Ubuntu 默认已安装了 SSH client，此外还需要安装 SSH server：

```
#也有的已经装好了，运行也没关系，如果没权限，前面需要加 sudo
root@master:~# apt-get install openssh-server
```

2. 安装后，修改sshd_config配置

```
root@master:~# vim /etc/ssh/sshd_config
#在文件中设置如下属性：（按 / 可以进入搜索模式，按esc退出搜索模式）
#这里是在原有属性上修改，不是添加
PubkeyAuthentication yes
PermitRootLogin yes
```

阿里云服务器默认是yes，可以不用修改。

3. 重启ssh服务

```
root@master:~# /etc/init.d/ssh restart
```

**下面操作是要让 master 节点可以无密码 SSH 登陆到各个 slave 节点上。**

1. 首先生成 master 节点的公匙，在 master节点的终端中执行：

```
# 进入.ssh目录
root@master:~# cd ~/.ssh

# 如果是空的服务器，这步不需要，如果以前生成过，需要删除
root@master:~/.ssh#  rm ./id_rsa*

# 一直按回车就可以，输入密码也回车
root@master:~/.ssh#  ssh-keygen -t rsa
```

2. 让 master 节点需能无密码 SSH 本机，在 master 节点上执行：

```
root@master:~/.ssh# cat ./id_rsa.pub >> ./authorized_keys
```

3. 完成后可执行 ssh master 验证一下（可能需要输入 yes，成功后执行 exit 返回原来的终端，一定要exit下）。
4. 接着在 master 节点将上公匙传输到 slave1和slave2节点，需要输入slave1和slave2服务器的密码：

```
root@master:~# cd .ssh
root@master:~/.ssh# scp id_rsa.pub root@slave1:/root/
root@master:~/.ssh# scp id_rsa.pub root@slave2:/root/
```

5. 接着切换到 slave1 节点服务器上，将 ssh master的公匙加入授权

```
# 如果不存在该文件夹需先创建，若已存在则忽略，一般是存在的
mkdir /root/.ssh

#这步作用是把master的公钥给slave1
cat /root/id_rsa.pub >> /root/.ssh/authorized_keys

# 用完就可以删掉了
rm /root/id_rsa.pub
```

6. 接着切换到 slave2 节点服务器上，以上同样操作将 ssh master的公匙加入授权
7. 这样，在 master 节点上就可以无密码 SSH 到各个 slave 节点了，可在 master 节点上执行如下命令进行检验，然后按exit回到master节点。

```
root@master:~# ssh slave1
```

```
root@master:~/.ssh# ssh slave1
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
New release '18.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Welcome to Alibaba Cloud Elastic Compute Service !

Last login: Thu May  9 18:37:19 2019 from 192.168.57.180
root@slave1:~#
```

## 6.3、修改Hadoop配置文件（master上配置）

1. 修改配置文件 core-site.xml

```
#进入配置文件目录
root@master:~# cd /usr/local/hadoop/etc/hadoop
root@master:/usr/local/hadoop/etc/hadoop# vim core-site.xml
```

将当中的

```
<configuration>
</configuration>
```

修改为下面配置：

```
<configuration>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>file:/usr/local/hadoop/tmp</value>
        <description>Abase for other temporary directories.</description>
    </property>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://master:9000</value>
    </property>
</configuration>
```

2.修改配置文件 hdfs-site.xml

```
root@master:/usr/local/hadoop/etc/hadoop# vim hdfs-site.xml
```

```
<configuration>
        <property>
                <name>dfs.namenode.secondary.http-address</name>
                <value>master:50090</value>
        </property>
        <property>
                <name>dfs.replication</name>
                <value>2</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>file:/usr/local/hadoop/tmp/dfs/name</value>
        </property>
        <property>
                <name>dfs.datanode.data.dir</name>
                <value>file:/usr/local/hadoop/tmp/dfs/data</value>
        </property>
</configuration>
```

3. 修改配置文件 mapred-site.xml

```
# 默认文件名为 mapred-site.xml.template，重命名为mapred-site.xml
root@master:/usr/local/hadoop/etc/hadoop# mv mapred-site.xml.template mapred-site.xml
root@master:/usr/local/hadoop/etc/hadoop# vim mapred-site.xml
```

```
<configuration>
        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>
        <property>
                <name>mapreduce.jobhistory.address</name>
                <value>master:10020</value>
        </property>
        <property>
                <name>mapreduce.jobhistory.webapp.address</name>
                <value>master:19888</value>
        </property>
</configuration>
```

4.修改配置文件 yarn-site.xml

```
root@master:/usr/local/hadoop/etc/hadoop# vim yarn-site.xml
```

```
<configuration>
<property>
     <name>yarn.resourcemanager.hostname</name>
     <value>master</value>
</property>
<property>
     <name>yarn.nodemanager.resource.memory-mb</name>
     <value>10240</value>
</property>
<property>
     <name>yarn.nodemanager.aux-services</name>
     <value>mapreduce_shuffle</value>
</property>
</configuration>
```

5. 修改文件 hadoop-env.sh，在文件开始处添加Hadoop和Java环境变量。

```
root@master:/usr/local/hadoop/etc/hadoop# vim hadoop-env.sh
```

```
export JAVA_HOME=/usr/local/java
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:/usr/local/hadoop/bin
```

6. 配置slaves，删除默认的localhost，增加从节点:

```
root@master:/usr/local/hadoop/etc/hadoop# vim slaves
```

```
slave1
slave2
```

7. 配置好后，将 master 上的 /usr/local/hadoop 文件夹复制到各个节点上。

   这步就省下了从节点上还要再安装和配置hadoop的步骤。

```
# 删除 Hadoop 临时文件
root@master:/usr/local/hadoop/etc/hadoop# cd
root@master:~# rm -rf /usr/local/hadoop/tmp

# 删除日志文件
root@master:~# rm -rf /usr/local/hadoop/logs

#把hadoop的所有东西都传输给两个从节点
root@master:~# scp -r /usr/local/hadoop slave1:/usr/local
root@master:~# scp -r /usr/local/hadoop slave2:/usr/local
```

8. 在master节点上启动hadoop

```
#先格式化节点信息
root@master:~# /usr/local/hadoop/bin/hdfs namenode -format

#启动hadoop所有环境
root@master:~# /usr/local/hadoop/sbin/start-all.sh
```

9. 成功启动后，运行jps命令

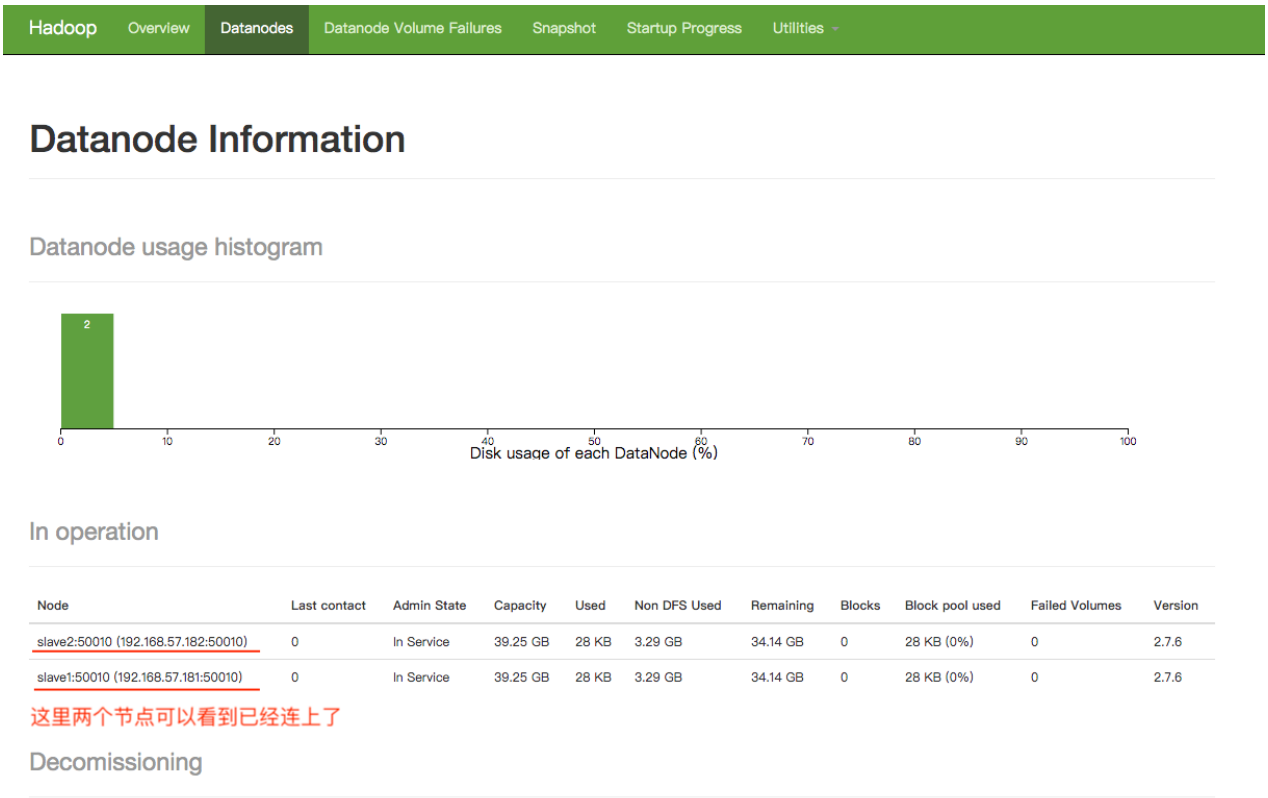
```
#让配置的环境生效
root@master:~# source /etc/profile

#查看节点信息
root@master:~# jps
```

```
root@master:~# jps
2309 Jps
1685 ResourceManager
1334 NameNode
1534 SecondaryNameNode
root@master:~#
```

10. 成功启动后，可以访问 Web 界面 http://39.106.49.230:50070查看 NameNode 和 Datanode 信息，39.106.49.230是服务器的ip地址，还可以在线查看 HDFS 中的文件。这里要注意你要看那个

端口的信息，你服务器上的对应的端口需要开放。




# 7. HBase集群配置

HBase是一个分布式的、面向列的开源数据库,源于Google的一篇论文《BigTable：一个结构化数据的分布式存储系统》。HBase以表的形式存储数据，表有行和列组成，列划分为若干个列族/列簇(column family)。欲了解HBase的官方资讯，请访问HBase官方网站。HBase的运行有三种模式：单机模式、伪分布式模式、分布式模式。

## 7.1、HBase安装

这里到本机终端下传输文件，同JDk和Hadoop传输文件方式一样，**这里要注意的是只在master主节点上配置就可以了。**

1. 传输文件

```
yygdeMac-mini:~ yyg$ scp /Users/yyg/Downloads/大数据之路-安装包/软件安装/hbase-2.0.0-bin.tar.gz root@39.106.49.230:/usr/local
```

2. 解压、重命名、配置环境变量、查看版本

```
#解压
root@master:~# cd /usr/local
root@master:/usr/local# tar -zxvf hbase-2.0.0-bin.tar.gz

#文件夹重命名为hbase
root@master:/usr/local# mv hbase-2.0.0 hbase
```

```
#配置环境变量，打开文件/etc/profile，文件末尾添加如下两行Hadoop环境变量
root@master:/usr/local# vim /etc/profile
export HBASE_HOME=/usr/local/hbase
export PATH=$HBASE_HOME/bin:$PATH
export HBASE_MANAGES_ZK=true

#同样，需要让该环境变量生效，执行如下代码：
root@master:/usr/local# source /etc/profile

#输入如下命令来检查 Hbase 是否可用，成功则会显示 Hbase 版本信息， 输出zhongtu：
root@master:/usr/local# hbase version
```

## 7.2、HBase集群模式配置

1. 修改配置文件 hbase-site.xml

```
#进入配置文件目录
root@master:~# cd /usr/local/hbase/conf
root@master:/usr/local/hbase/conf#  vim hbase-site.xml
```

将当中的

```
<configuration>
</configuration>
```

修改为下面配置：

这里hbase.zookeeper.quorum所在行添加了3个节点信息，如果有更多节点，可以加上。

```
<configuration>
        <property>
                <name>hbase.rootdir</name>
                <value>hdfs://master:9000/hbase</value>
        </property>
        <property>
                <name>hbase.cluster.distributed</name>
                <value>true</value>
        </property>
        <property>
                <name>hbase.zookeeper.quorum</name>
                <value>master,slave1,slave2</value>
        </property>
        <property>
                <name>hbase.temp.dir</name>
                <value>/usr/local/hbase/tmp</value>
        </property>
        <property>
```

```
                <name>hbase.zookeeper.property.dataDir</name>
                <value>/usr/local/hbase/tmp/zookeeper</value>
        </property>
        <property>
                <name>hbase.master.info.port</name>
                <value>16010</value>
        </property>
</configuration>
```

2. 修改文件 hbase-env.sh，在文件开始处添加添加hbase和java环境变量。

```
root@master:/usr/local/hbase/conf# vim hbase-env.sh
```

```
export JAVA_HOME=/usr/local/java
export HBASE_HOME=/usr/local/hbase
export PATH=$PATH/usr/local/hbase/bin
```

3. 配置slaves，删除默认的localhost，增加从节点:

```
root@master:/usr/local/hbase/conf# vim regionservers
```

```
master
slave1
slave2
```

3. 传送Hbase至其它slave节点，即将配置好的hbase文件夹传送到各个节点对应位置上。

```
root@master:/usr/local/hbase/conf# scp -r /usr/local/hbase
root@slave1:/usr/local/

root@master:/usr/local/hbase/conf# scp -r /usr/local/hbase
root@slave2:/usr/local/
```

4. 测试运行

```
#再启动HBase。命令如下
root@master:/usr/local/hbase/conf# cd
root@master:~# /usr/local/hadoop/sbin/start-all.sh  #启动hadoop，如果已启动，则不用
执行该命令

#启动hbase
root@master:~# /usr/local/hbase/bin/start-hbase.sh   #启动hbase

#进入hbase shell，如果可以进入hbase交互式命令行，说明HBase安装成功了，按exit可退出。
root@master:~# hbase shell
```

5. 如果hbase启动成功，则使用jps命令会出现如下进程

```
root@master:~# jps
```

```
root@master:~# jps
3170 HMaster
3107 HQuorumPeer
1685 ResourceManager
1334 NameNode
3321 HRegionServer
3723 Jps
1534 SecondaryNameNode
root@master:~#
```

```
root@slave1:/usr/local# jps
1729 HQuorumPeer
2003 Jps
1241 DataNode
1819 HRegionServer
1355 NodeManager
root@slave1:/usr/local#
```

# 8. Spark集群配置

Apache Spark 是一个新兴的大数据处理通用引擎，提供了分布式的内存抽象。Spark 最大的特点就是快，可比 Hadoop MapReduce 的处理速度快 100 倍。Spark基于Hadoop环境，Hadoop YARN为Spark提供资源调度框架，Hadoop HDFS为Spark提供底层的分布式文件存储。

## 8.1、Spark安装

这里到本机终端下传输文件，同JDk和Hadoop传输文件方式一样，**这里要注意的是只在master主节点上配置就可以了。**在已安装好 Hadoop 的前提下，经过简单配置即可使用。

1. 传输文件

```
yygdeMac-mini:~ yyg$ scp /Users/yyg/Downloads/大数据之路-安装包/软件安装/spark-2.3.0-bin-hadoop2.7.tgz root@39.106.49.230:/usr/local
```

2. 解压、重命名、配置环境变量、查看版本

```
#解压
root@master:~# cd /usr/local
root@master:/usr/local# tar -zxvf spark-2.3.0-bin-hadoop2.7.tgz

#文件夹重命名为spark
root@master:/usr/local# mv spark-2.3.0-bin-hadoop2.7 spark
```

```
#配置环境变量，打开文件/etc/profile，文件末尾添加如下两行Spark环境变量
root@master:/usr/local# vim /etc/profile
export SPARK_HOME=/usr/local/spark
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin

#同样，需要让该环境变量生效，执行如下代码：
root@master:/usr/local# source /etc/profile
```

## 8.2、Spark集群配置

1. 复制并重命名配置文件 spark-env.sh

```
#进入配置文件目录，复制并重命名spark-env.sh文件
root@master:~# cd /usr/local/spark/conf
root@master:/usr/local/spark/conf#  cp spark-env.sh.template spark-env.sh
```

2. 编辑spark-env.sh文件，在第一行添加以下配置信息:

```
root@master:/usr/local/spark/conf# vim spark-env.sh
```

```
export JAVA_HOME=/usr/local/java
export SCALA_HOME=/usr/local/scala
export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
export HADOOP_HDFS_HOME=/usr/local/hadoop
export SPARK_HOME=/usr/local/spark
export SPARK_MASTER_IP=master
export SPARK_MASTER_PORT=7077
export SPARK_MASTER_HOST=master
export SPARK_WORKER_CORES=2
export SPARK_WORKER_PORT=8901
export SPARK_WORKER_INSTANCES=1
export SPARK_WORKER_MEMORY=2g
export SPARK_DIST_CLASSPATH=$(/usr/local/hadoop/bin/hadoop classpath)
export SPARK_MASTER_WEBUI_PORT=8079
```

2. 保存并生效配置

```
root@master:/usr/local/spark/conf# source spark-env.sh
```

3. 配置slaves

```
#先复制slaves.template，并重命名slaves节点文件
root@master:/usr/local/spark/conf# cp slaves.template slaves
root@master:/usr/local/spark/conf# vim slaves
```

```
#删除默认的localhost，增加主从节点
master
slave1
slave2
```

3. 传送Spark至其它slave节点，即将配置好的spark文件夹传送到各个节点对应位置上。

```
root@master:/usr/local/spark/conf# scp -r /usr/local/spark
root@slave1:/usr/local
root@master:/usr/local/spark/conf# scp -r /usr/local/spark
root@slave2:/usr/local
```

4. 测试运行

```
root@master:/usr/local/hbase/conf# cd
#启动spark
root@master:~# /usr/local/spark/sbin/start-all.sh

#运行spark自带的例子求Pi的值
root@master:~# /usr/local/spark/bin/run-example SparkPi 2>&1 | grep "Pi is"
```

运行结果如下图所示，可以得到π 的 14位小数近似值：

```
root@master:~# /usr/local/spark/bin/run-example SparkPi 2>&1 |
 grep "Pi is"
Pi is roughly 3.1404757023785117
```

成功启动后，可以访问 Web 界面 http://39.106.49.230:8079就可以看到有三个节点在spark集群上。。

# 9. Hive配置

Hive是一个架构在Hadoop之上的数据仓库基础工具，用来处理结构化数据，为大数据查询和分析提供方便。最初，Hive是由Facebook开发，后来由Apache软件基金会开发，并作为进一步将它作为名义下Apache Hive为一个开源项目。Hive 不是一个关系数据库，也不是一个设计用于联机事务处（OLTP）实时查询和行级更新的语言。简单的说，Hive就是在Hadoop上架了一层SQL接口，可以将SQL翻译成MapReduce去Hadoop上执行，这样就使得数据开发和分析人员很方便的使用SQL来完成海量数据的统计和分析，而不必使用编程语言开发MapReduce那么麻烦。

## 9.1、 Hive安装

这里到本机终端下传输文件，同JDk和Hadoop传输文件方式一样，**这里要注意的是只在master主节点上配置就可以了。**

1. 传输文件

```
yygdeMac-mini:~ yyg$ scp /Users/yyg/Downloads/大数据之路-安装包/软件安装/apache-hive-1.2.2-bin.tar.gz root@39.106.49.230:/usr/local
```

2. 解压、重命名、配置环境变量、查看版本

```
#解压
root@master:~# cd /usr/local
root@master:/usr/local# tar -zxvf apache-hive-1.2.2-bin.tar.gz

#文件夹重命名为hive
root@master:/usr/local# mv apache-hive-1.2.2-bin hive

#配置环境变量，打开文件/etc/profile，文件末尾添加如下两行hive环境变量
root@master:/usr/local# vim /etc/profile
export HIVE_HOME=/usr/local/hive
export PATH=$PATH:$HIVE_HOME/bin

#同样，需要让该环境变量生效，执行如下代码：
root@master:/usr/local# source /etc/profile
```

## 9.2、 安装并配置MySQL

我们采用MySQL数据库保存Hive的元数据，而不是采用Hive自带的derby来存储元数据。ubuntu下Mysql的安装比较简单，直接运行如下命令。在安装过程中，会要求配置用户名和密码，这个一定要记住。

1. 安装mysql-server

```
root@master:/usr/local# cd
root@master:~# apt-get update

#安装mysql-server
#这步会跳到一个界面让你设置密码，要记住用户名和密码。用户名一般是root，密码自己设置，后面用的到。
root@master:~# apt-get install mysql-server
```

2. 启动并登陆mysql shell

```
root@master:~# service mysql start
root@master:~# mysql -u root -p
```

3. 配置hive数据库

```
#这个hive数据库与hive-site.xml中localhost:3306/hive的hive对应，用来保存hive元数据
mysql> create database hive;

#将hive数据库的字符编码设置为latin1（重要）
mysql> alter database hive character set latin1;
```

完成后，按exit退出。

## 9.3、 Hive配置

1. 将hive-default.xml.template重命名为hive-default.xml

```
root@master:~# cd /usr/local/hive/conf
root@master:/usr/local/hive/conf# mv hive-default.xml.template hive-default.xml
```

2. 使用vim编辑器新建一个配置文件hive-site.xml，命令如下：

```
root@master:/usr/local/hive/conf# vim hive-site.xml
```

在hive-site.xml中添加如下配置信息，其中：**USERNAME和PASSWORD是上面MySQL的用户名和密码。**这步自己填写更换。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
  <property>
    <name>javax.jdo.option.ConnectionURL</name>
    <value>jdbc:mysql://localhost:3306/hive?
createDatabaseIfNotExist=true</value>
    <description>JDBC connect string for a JDBC metastore</description>
  </property>
  <property>
    <name>javax.jdo.option.ConnectionDriverName</name>
    <value>com.mysql.jdbc.Driver</value>
    <description>Driver class name for a JDBC metastore</description>
  </property>
  <property>
    <name>javax.jdo.option.ConnectionUserName</name>
    <value>USERNAME</value>
    <description>username to use against metastore database</description>
  </property>
  <property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value>PASSWORD</value>
    <description>password to use against metastore database</description>
  </property>
</configuration>
```

3. 由于Hive在连接MySQL时需要JDBC驱动 ，所以首先需要下载对应版本的驱动，然后将驱动移动到/usr/local/hive/lib中。 这个驱动自己下载到本地，然后上传。

   **本机上传输驱动到/usr/local/hive/lib目录下。**

```
yygdeMac-mini:~ yyg$ scp /Users/yyg/Downloads/大数据之路-安装包/软件安装/mysql-
connector-java-5.1.47.tar.gz root@39.106.49.230:/usr/local/hive/lib
```

4. 解压和移动驱动

```
#解压到/usr/local/hive/lib目录下
root@master:/usr/local/hive/conf# cd /usr/local/hive/lib
root@master:/usr/local/hive/lib#  tar -zxvf mysql-connector-java-5.1.47.tar.gz

#把驱动里面的jar包复制到/usr/local/hive/lib目录下
root@master:/usr/local/hive/lib# cp mysql-connector-java-5.1.47/mysql-
connector-java-5.1.47-bin.jar /usr/local/hive/lib
```

5.启动hive（启动hive之前，请先启动hadoop集群）。

```
#启动hadoop，如果已经启动，则不用执行该命令
root@master:/usr/local/hive/lib# /usr/local/hadoop/sbin/start-all.sh

#启动hive，这个启动比较慢，要等一会，首行会出现个错误，下节解决。
root@master:/usr/local/hive/lib# hive
```

6. 如果启动hive出现如下错误：

   ls: cannot access '/usr/local/spark/lib/spark-assembly-*.jar': No such file or directory

   原因是这个jar包在新版本的spark中的位置已经改变！我们要做的只是将hive中的启动文件中的sparkAssemblyPath这一行更改为你安装的spark的jar包路径即可。具体如下：

   到Hive的bin目录下编辑hive。

```
root@master:~# cd /usr/local/hive/bin
#查看并编辑hive文件
root@master:/usr/local/hive/bin# ls
beeline  hive            hiveserver2  schematool
ext      hive-config.sh  metatool

root@master:/usr/local/hive/bin# vim hive
```

找到下图标记的这一行；

```
# add Spark assembly jar to the classpath
if [[ -n "$SPARK_HOME" ]]
then
  sparkAssemblyPath=`ls ${SPARK_HOME}/lib/spark-assembly-*.jar`
  CLASSPATH="${CLASSPATH}:${sparkAssemblyPath}"
fi
```



将上图红框内的内容更改为下图所示内容。

```
# add Spark assembly jar to the classpath
if [[ -n "$SPARK_HOME" ]]
then
  sparkAssemblyPath=`ls ${SPARK_HOME}/jars/*.jar`
  CLASSPATH="${CLASSPATH}:${sparkAssemblyPath}"
fi
```



再次启动hive后就没有报错了，问题解决！！

**到此，大数据集群环境的搭建暂时结束。**

# 10. 结语

1. 本教程介绍了大数据环境的搭建过程，大数据还有好些库不在本教程内，但原理都差不多，大家可以去网上找相应的教程、库的安装包并配置一下即可。
2. 如果用最新版本安装包配置的同学，可能会出现软件不兼容的现象，这点需要注意。
3. 理解配置的过程需要哪些东西非常重要，不然如果出现少操作，多操作，手误情况会很难定位到问题所在。
4. 本教程主要参考一篇博客，在此表示感谢：技术颜良