Hadoop全平台搭建流程

目录

[虚拟机安装及网络配置 2](#_Toc10475)

[创建虚拟机 2](#_Toc7936)

[设置网络 3](#_Toc23099)

[Jdk安装配置 4](#_Toc25738)

[Hadoop安装配置 5](#_Toc5194)

[1.配置免密登录 5](#_Toc20721)

[2.搭建Hadoop集群 6](#_Toc20339)

[Zookeeper安装配置 10](#_Toc11768)

[Mysql安装配置 13](#_Toc24057)

[Hive安装配置（前置mysql） 16](#_Toc7736)

[Hbase安装配置( 前置：Hadoop,zookeeper,jdk) 20](#_Toc5097)

[Sqoop安装配置及使用 21](#_Toc23032)

[安装配置 21](#_Toc29466)

[使用方法 24](#_Toc950)

[Flume安装及使用 30](#_Toc9224)

[安装配置 30](#_Toc23061)

[使用 33](#_Toc2710)

[Spark安装配置 36](#_Toc31237)

所用软件版本：centos7; jdk8u171; haoop2.7.6; hive1.2.1; sqoop1.4.7; spark2.4.5

虚拟机安装及网络配置

创建虚拟机

1. 打开vmmare
2. 创建虚拟机
3. 稍后安装
4. 选择linux centos7
5. 虚拟机命名 设置保存路径 cpu核数 内存大小 网络类型NAT 创建新磁盘 设置大小 拆分成多个文件
6. 虚拟机设置 设置iso镜像为centos7
7. 启动虚拟机安装centos7
8. 最小化安装
9. 网络和主机名开启以太网连接
10. 设置root密码为123456
11. 等待安装完成
12. 重启虚拟机 使用root用户登录

设置网络

1.修改主机名

vi /etc/hostname

修改主机名为 要求的主机名 例如master2021

2.关闭防火墙

systemctl stop firewalld

查看防火墙状态

systemctl status firewalld

取消防火墙自启

systemctl disable firewalld

1. 静态IP配置

编辑网络配置文件

vim /etc/sysconfig/network-scripts/ifcfg-ens33

BOOTPROTO=static

ONBOOT=yes

#IP地址

IPADDR=192.168.XXX.120

NETMASK=255.255.255.0

#网关

GATEWAY=192.168.XXX.2

DNS1=114.114.114.114

关闭NetworkManager，并取消开机自启

systemctl stop NetworkManager

systemctl disable NetworkManager

重启网络服务

systemctl restart network

添加主机映射

vim /etc/hosts

添加三个节点名称

例如：192.168.2.100 master2021

192.168.2.101 node12021

192.168.2.102 node22021

重启虚拟机

Jdk安装配置

1.创建路径

cd /usr/local

mkdir soft

2.上传jdk

3.解压

tar -zxvf jdk-8u171-linux-x64.tar.gz

4. 修改环境变量

vim /etc/profile

export JAVA\_HOME=/usr/local/soft/jdk1.8.0\_171

export PATH=$JAVA\_HOME/bin:$PATH

#使环境变量生效

source /etc/profile

5.克隆虚拟机 创建完整克隆 建立node12021 node22021

6.设置node1的网络 和 node2 的网络

Hadoop安装配置

1.配置免密登录

注意：只需要在master2021上执行以下命令即可

# 生成密钥（按3次回车即可）

ssh-keygen -t rsa

# 配置免密登录（输入密码123456）

ssh-copy-id master2021

ssh-copy-id node12021

ssh-copy-id node22021

# 测试免密登录

ssh node12021

2.搭建Hadoop集群

上传安装包并解压

tar -zxvf hadoop-2.7.6.tar.gz

#配置环境变量

vim /etc/profile

export HADOOP\_HOME=/usr/local/soft/hadoop-2.7.6

export PATH=$HADOOP\_HOME/sbin:$HADOOP\_HOME/bin:$PATH

# 重新加载环境变量

source /etc/profile

#修改Hadoop配置文件

cd /usr/local/soft/hadoop-2.7.6/etc/hadoop

对core-site.xml进行修改

在<configuration>下添加以下内容

<property>

<name>fs.defaultFS</name>

<value>hdfs://***master2021:9000***</value>

</property>

<property>

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

<value>/usr/local/soft/hadoop-2.7.6/tmp</value>

</property>

<property>

<name>fs.trash.interval</name>

<value>1440</value>

</property>

对hadoop-env.sh进行修改

在export JAVA\_HOME=${JAVA\_HOME}内添加以下内容

export JAVA\_HOME=/usr/local/soft/jdk1.8.0\_171

对hdfs-site.xml进行修改

在<configuration>下添加以下内容

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

对mapred-site.xml.template复制后进行mapred-site.xml修改

cp mapred-site.xml.template mapred-site.xml

在<configuration>下添加以下内容

<property>

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

<value>yarn</value>

</property>

<property>

<name>mapreduce.jobhistory.address</name>

<value>***master2021:10020***</value>

</property>

<property>

<name>mapreduce.jobhistory.webapp.address</name>

<value>***master2021:19888***</value>

</property>

对slaves进行修改

vi slaves

在文件内直接添加：

***node12021***

***node22021***

对yarn-site.xml进行修改

在<configuration>下添加以下内容

<property>

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

<value>***master2021***</value>

</property>

<property>

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

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>yarn.log-aggregation-enable</name>

<value>true</value>

</property>

<property>

<name>yarn.log-aggregation.retain-seconds</name>

<value>604800</value>

</property>

分发Hadoop到node1、node2

cd /usr/local/soft/

注意` 是反单引号 键盘~的键 不是’ 分号

scp -r hadoop-2.7.6/ node12021:`pwd`

scp -r hadoop-2.7.6/ node22021:`pwd`

格式化namenode（第一次启动的时候需要执行）

hdfs namenode -format

启动Hadoop集群

start-all.sh

使用jps命令验证启动是否成功

Zookeeper安装配置

1.上传安装包 解压

tar -zxvf zookeeper-3.4.6.tar.gz

2.配置环境变量

vim /etc/profile

export ZOOKEEPER\_HOME=/usr/local/soft/zookeeper-3.4.6

export PATH=$PATH:$ZOOKEEPER\_HOME/bin

source /etc/profile

3、修改配置文件

cd conf

cp zoo\_sample.cfg zoo.cfg

修改

dataDir=/usr/local/soft/zookeeper-3.4.6/data

增加

server.0=***master2021***:2888:3888

server.1=***node12021***:2888:3888

server.2=***node22021***:2888:3888

4、同步到其它节点

scp -r zookeeper-3.4.6 node12021:`pwd`

scp -r zookeeper-3.4.6 node22021:`pwd`

配置node1和node2的环境变量

scp /etc/profile node12021:/etc/

scp /etc/profile node22021:/etc/

在所有节点执行

source /etc/profile

5、创建/usr/local/soft/zookeeper-3.4.6/data目录,所有节点都要创建

mkdir /usr/local/soft/zookeeper-3.4.6/data

在data目录下创建myid文件

vim myid

master,node1,node2分别加上0，1，2

6、启动zk

zkServer.sh start 三台都需要执行

zkServer.sh status 查看状态

当有一个leader的时候启动成功

连接zk

zkCli.sh

zk 是一个目录结构 ，每个节点可以存数据，同时可以有子节点

zk shell

创建目录

create /test test

create /test/a 1

获取数据

get /test

ls /test

delete 只能删除没有子节点的节点

rmr /test 删除节点

Mysql安装配置

1.上传解压

tar -zxvf mysql-5.7.34-linux-glibc2.12-x86\_64.tar.gz

改名

mv mysql-5.7.34-linux-glibc2.12-x86\_64 mysql

2.在 mysql 目录下生成 data 目录

mkdir /usr/local/soft/mysql/data

3.创建 my.cnf 文件(vi my.cnf),并添加如下内容

[mysqld]

basedir = /usr/local/soft/mysql

datadir = /usr/local/soft/mysql/data

socket = /tmp/mysql.sock

lower\_case\_table\_names=1

user=root

character\_set\_server = utf8

symbolic-links =0

[client]

default-character-set = utf8

[mysql-safe]

Log-error=/var/log/mysql.log

pid-file = /var/run/mysql/mysql.pid

.4.覆盖/etc 下原有的 my.cnf

cp my.cnf /etc/

删除my.cnf

rm -rf my.cnf

5.初始化数据库

/usr/local/soft/mysql/bin/mysqld --initialize-insecure

--basedir=/usr/local/soft/mysql --datadir=/usr/local/soft/mysql/data --user=root

6.将 mysql 加入服务

cp /usr/local/soft/mysql/support-files/mysql.server

/etc/init.d/mysql

7.设置 mysql 开机自启

chkconfig mysql on

8.启动 mysql 服务

service mysql start

9.配置 mysql 环境变量

vi /etc/profile

export MYSQL\_HOME=/usr/local/soft/mysql

export PATH=$PATH:$MYSQL\_HOME/bin

source /etc/profile

10.登录 mysql，密码为空直接登录

mysql -u root -p

11.在 mysql 登录进去后更改密码

set password=password('123456');

12.创建 hive 数据库

create database hive default charset utf8;

13.赋权限

grant all privileges on \*.\* to 'root'@'%' identified by '123456';

14.刷新权限

flush privileges ;

15.退出 mysql :

exit

Hive安装配置（前置mysql）

1.上传并解压hive

tar -zxvf apache-hive-1.2.1-bin.tar.gz

2.改名

mv apache-hive-1.2.1-bin hive-1.2.1

3.配置hive的环境变量

vi /etc/profile

export HIVE\_HOME=/usr/local/soft/hive-1.2.1

export PATH=$PATH:$HIVE\_HOME/bin

source /etc/profile

4.进入hive-1.2.1/conf目录，复制备份文件并重命名

cp hive-env.sh.template hive-env.sh

cp hive-default.xml.template hive-site.xml

5.修改hive-env.sh和hive-site.xml

#hive-env.sh 添加如下语句

HADOOP\_HOME=/usr/local/soft/hadoop-2.7.6

JAVA\_HOME=/usr/local/soft/jdk1.8.0\_171

HIVE\_HOME=/usr/local/soft/hive-1.2.1

（hive的配置文件比较大，在linux中查找某项配置比较难，可以先将hive-site.xml文件复制到windows用文本编辑打开，然后ctrl+f查关键字修改，修改之后再放回到hive 的conf目录）

#hive-site.xml 更改如下语句

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://***master2021***:3306/hive?useSSL=false</value>

</property>

<property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value>

</property>

<property>

<name>javax.jdo.option.ConnectionUserName</name>

<value>root</value>

</property>

<property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>123456</value>

</property>

<property>

<name>hive.querylog.location</name>

<value>/usr/local/soft/hive-1.2.1/tmp</value>

</property>

<property>

<name>hive.exec.local.scratchdir</name>

<value>/usr/local/soft/hive-1.2.1/tmp</value>

</property>

<property>

<name>hive.downloaded.resources.dir</name>

<value>/usr/local/soft/hive-1.2.1/tmp</value>

</property>

<property>

<name>hive.cli.print.header</name>

<value>true</value>

</property>

<property>

<name>hive.cli.print.current.db</name>

<value>true</value>

6.上传mysql驱动

cd /usr/local/soft/hive-1.2.1/lib/

使用xftp上传 mysql-connector-java-5.1.49-bin.jar

7.将hive的jline-2.12.jar拷贝到hadoop对应目录下

cp /usr/local/soft/hive-1.2.1/lib/jline-2.12.jar /usr/local/soft/hadoop-2.7.6/share/hadoop/yarn/lib/

8.启动hive的元数据（后台启动hive的元数据）

nohup hive --service metastore &

nohup hive --service hiveserver2 &

9.输入hive启动数据库

Hbase安装配置( 前置：Hadoop,zookeeper,jdk)

1.hadoop及zookeeper处于启动状态

使用zkServer.sh status验证zookeeper是否处于启动状态

2.上传解压

tar -zxvf hbase-1.4.6-bin.tar.gz

3.配置环境变量

export HBASE\_HOME=/usr/local/soft/hbase-1.7.1

export PATH=$PATH:$HBASE\_HOME/bin

source /etc/profile

4.修改hbase-env.sh文件（cd /usr/local/soft/hbase-1.7.1/conf）

直接在最下面添加

export JAVA\_HOME=/usr/local/soft/jdk1.8.0\_171

export HBASE\_MANAGES\_ZK=false

5.修改hbase-site.xml文件

<property>

<name>hbase.rootdir</name>

<value>hdfs://***master2021:9000***/hbase</value>

</property>

<property>

<name>hbase.cluster.distributed</name>

<value>true</value>

</property>

<property>

<name>hbase.zookeeper.quorum</name>

<value>***node12021,node22021,master2021***</value>

</property>

6.修改regionservers文件

添加

***node12021***

***node22021***

7.同步到所有节点

scp -r hbase-1.7.1 node12021:`pwd`

scp -r hbase-1.7.1 node22021:`pwd`

8.启动hbase集群 ， 在master上执行

start-hbase.sh

7.验证hbase

http://master2021:16010

Sqoop安装配置及使用

安装配置

1.上传并解压

tar -zxvf sqoop-1.4.7.bin\_\_hadoop-2.0.4-alpha.tar.gz

2.修改文件夹名字

mv sqoop-1.4.7.bin\_\_hadoop-2.0.4-alpha/ sqoop-1.4.7

3.修改配置文件

cd /usr/local/soft/sqoop-1.4.7/conf

# 复制配置文件并重命名

cp sqoop-env-template.sh sqoop-env.sh

#vim sqoop-env.sh 编辑配置文件，并加入以下内容

export HADOOP\_COMMON\_HOME=/usr/local/soft/hadoop-2.7.6

export HADOOP\_MAPRED\_HOME=/usr/local/soft/hadoop-2.7.6/share/hadoop/mapreduce

export HBASE\_HOME=/usr/local/soft/hbase-1.7.1

export HIVE\_HOME=/usr/local/soft/hive-1.2.1

export ZOOCFGDIR=/usr/local/soft/zookeeper-3.4.6/conf

export ZOOKEEPER\_HOME=/usr/local/soft/zookeeper-3.4.6

# 切换到bin目录

cd /usr/local/soft/sqoop-1.4.6/bin

# vim configure-sqoop

注释以下所有内容（前面加#）

## Moved to be a runtime check in sqoop.

if [ ! -d "${HCAT\_HOME}" ]; then

echo "Warning: $HCAT\_HOME does not exist! HCatalog jobs will fail."

echo 'Please set $HCAT\_HOME to the root of your HCatalog installation.'

fi

if [ ! -d "${ACCUMULO\_HOME}" ]; then

echo "Warning: $ACCUMULO\_HOME does not exist! Accumulo imports will fail."

echo 'Please set $ACCUMULO\_HOME to the root of your Accumulo installation.'

Fi

4.修改环境变量

export SQOOP\_HOME=/usr/local/soft/sqoop-1.4.7

export PATH=$PATH:$SQOOP\_HOME/bin

source /etc/profile

5.添加MySQL连接驱动

从HIVE中复制MySQL连接驱动到$SQOOP\_HOME/lib

cp /usr/local/soft/hive-1.2.1/lib/mysql-connector-java-5.1.49.jar /usr/local/soft/sqoop-1.4.7/lib/

6.测试

sqoop version

# 测试MySQL连通性

sqoop list-databases -connect jdbc:mysql://***master2021***:3306/ -username root -password 123456

使用方法：

补充：Mysql导入数据方法

例如：

Mysql中执行

source /usr/local/soft/bigdata17/scrips/student.sql;

导入student.sql数据库

MySQLToHDFS

编写脚本，保存为MySQLToHDFS.conf

sqoop执行脚本有两种方式：第一种方式：直接在命令行窗口中直接输入脚本；第二种方式是将命令封装成一个脚本文件，然后使用另一个命令执行

第一种方式：

sqoop import \

--append \

--connect jdbc:mysql://***master2021:3306***/student \

--username root \

--password 123456 \

--table student \

--m 4 \

--split-by age \

--target-dir /shujia/bigdata17/student1/ \

--fields-terminated-by '\t'

第二种方式：

import

--append

--connect

jdbc:mysql://***master2021***:3306/student （此处连接数据库）

--username

root

--password

123456

--table

student （此处设置要导出的表）

--m （设置mapreducetask个数 可以不设置）

4

--split-by （设置切分字段 可以不设置）

age

--target-dir

/shujia/bigdata17/student21/ （设置输出路径）

--fields-terminated-by

','

执行脚本

sqoop --options-file MySQLToHDFS.conf

Mysql to Hive

import

--connect

jdbc:mysql://***master2021:3306***/student

--username

root

--password

123456

--table

score

--m

3

--split-by

student\_id

--fields-terminated-by

'\t'

--hive-import

--hive-overwrite

--create-hive-table

--hive-database（设置hive的数据库）

sqooptest

--hive-table（设置导入的表名）

mysqltoscore

执行sqoop --options-file MySQLToHive.conf

查询的使用

import

--connect

jdbc:mysql://master:3306/student

--username

root

--password

123456

--m

2

--split-by

student\_id

--e（跟上查询语句，必须加$CONDITIONS"）

"select \* from score where student\_id=1500100001 and $CONDITIONS"

--target-dir

/testE

--fields-terminated-by

'\t'

--hive-import

--hive-overwrite

--create-hive-table

--hive-database

sqooptest

--hive-table

mysqltoscore3

--direct

MySQLToHBase

import

--connect

jdbc:mysql://master:3306/student

--username

root

--password

123456

--table

student

--hbase-table

studentsq

--column-family

cf1

--hbase-row-key

id

--m

1

在HBase中创建student表

create 'studentsq','cf1'

执行sqoop --options-file MySQLToHBase.conf

HDFSToMySQL

//需要事先在mysql中建好表

export

--connect

jdbc:mysql://master:3306/student?useUnicode=true&characterEncoding=UTF-8

--username

root

--password

123456

--table

student

--m

1

--columns

id,name,age,gender,clazz

--export-dir

/shujia/bigdata17/sqoopinput/

--fields-terminated-by

','

执行sqoop --options-file HDFSToMySQL.conf

Flume安装及使用

安装配置

1.上传至虚拟机，并解压

tar -zxvf apache-flume-1.6.0-bin.tar.gz -C /usr/local/soft/

2.重命名目录，并配置环境变量

mv apache-flume-1.6.0-bin/ flume-1.6.0

vim /etc/profile

export FLUME\_HOME=/usr/local/soft/flume-1.6.0

export PATH=$PATH:$FLUME\_HOME/bin

source /etc/profile

3.查看flume版本

flume-ng version

4.测试flume

创建一个文件

spoolingtest.conf

写入

# a表示给agent命名为a

# 给source组件命名为r1

a.sources = r1

# 给sink组件命名为k1

a.sinks = k1

# 给channel组件命名为c1

a.channels = c1

#指定spooldir的属性

a.sources.r1.type = spooldir

a.sources.r1.spoolDir = /root/data

a.sources.r1.fileHeader = true

a.sources.r1.interceptors = i1

a.sources.r1.interceptors.i1.type = timestamp

#指定sink的类型

a.sinks.k1.type = logger

#指定channel

a.channels.c1.type = memory

a.channels.c1.capacity = 1000

# 表示sink每次会从channel里取多少数据

a.channels.c1.transactionCapacity = 100

# 组装

a.sources.r1.channels = c1

a.sinks.k1.channel = c1spoolingToHDFS.conf

配置文件

a.channels.c1.type = memory

a.channels.c1.capacity = 1000

# 表示sink每次会从channel里取多少数据

a.channels.c1.transactionCapacity = 100

# 组装

a.sources.r1.channels = c1

a.sinks.k1.channel = c1

新建/root/data目录

mkdir /root/data

启动

flume-ng agent -n a -f ./spoolingtest.conf

-Dflume.root.logger=DEBUG,console

在/root/data/目录下新建文件，输入内容，观察flume进程打印的日志

# 随意在a.txt中加入一些内容

vim /root/data/a.txt

5.使用

spoolingToHDFS.conf

# a表示给agent命名为a

# 给source组件命名为r1

a.sources = r1

# 给sink组件命名为k1

a.sinks = k1

# 给channel组件命名为c1

a.channels = c1

#指定spooldir的属性

a.sources.r1.type = spooldir

a.sources.r1.spoolDir = /root/data

a.sources.r1.fileHeader = true

a.sources.r1.interceptors = i1

a.sources.r1.interceptors.i1.type = timestamp

#指定sink的类型

a.sinks.k1.type = hdfs

a.sinks.k1.hdfs.path = /flume/data/dir1

# 指定文件名前缀

a.sinks.k1.hdfs.filePrefix = student

# 指定达到多少数据量写一次文件 单位：bytes

a.sinks.k1.hdfs.rollSize = 102400

# 指定多少条写一次文件

a.sinks.k1.hdfs.rollCount = 1000

# 指定文件类型为 流 来什么输出什么

a.sinks.k1.hdfs.fileType = DataStream

# 指定文件输出格式 为text

a.sinks.k1.hdfs.writeFormat = text

# 指定文件名后缀

a.sinks.k1.hdfs.fileSuffix = .txt#指定channel

a.channels.c1.type = memory

a.channels.c1.capacity = 1000

# 表示sink每次会从channel里取多少数据

a.channels.c1.transactionCapacity = 100

# 组装

a.sources.r1.channels = c1

a.sinks.k1.channel = c1

运行测试同上

hbaseLogToHDFS

# a表示给agent命名为a

# 给source组件命名为r1

a.sources = r1

# 给sink组件命名为k1

a.sinks = k1

# 给channel组件命名为c1

a.channels = c1

#指定spooldir的属性

a.sources.r1.type = exec

a.sources.r1.command = tail -f /usr/local/soft/hbase-1.4.6/logs/hbaseroot-master-master.log

#指定sink的类型

a.sinks.k1.type = hdfs

a.sinks.k1.hdfs.path = /flume/data/dir2

# 指定文件名前缀

a.sinks.k1.hdfs.filePrefix = hbaselog

# 指定达到多少数据量写一次文件 单位：bytes

a.sinks.k1.hdfs.rollSize = 102400

# 指定多少条写一次文件

a.sinks.k1.hdfs.rollCount = 1000

# 指定文件类型为 流 来什么输出什么

a.sinks.k1.hdfs.fileType = DataStream

# 指定文件输出格式 为text

a.sinks.k1.hdfs.writeFormat = text

# 指定文件名后缀

a.sinks.k1.hdfs.fileSuffix = .txt

#指定channel

a.channels.c1.type = memory

a.channels.c1.capacity = 1000

# 表示sink每次会从channel里取多少数据

a.channels.c1.transactionCapacity = 100

# 组装

a.sources.r1.channels = c1

a.sinks.k1.channel = c1

运行测试同上

hbaselogToHBase

在hbase中创建log表

create 'log','cf1'

配置文件

# a表示给agent命名为a

# 给source组件命名为r1

a.sources = r1

# 给sink组件命名为k1

a.sinks = k1

# 给channel组件命名为c1

a.channels = c1

#指定spooldir的属性

a.sources.r1.type = exec

a.sources.r1.command = cat /usr/local/soft/hbase-1.4.6/logs/hbase-rootmaster-master.log

#指定sink的类型

a.sinks.k1.type = hbase

a.sinks.k1.table = log

a.sinks.k1.columnFamily = cf1

#指定channel

a.channels.c1.type = memory

a.channels.c1.capacity = 100000

# 表示sink每次会从channel里取多少数据

a.channels.c1.transactionCapacity = 100

# 组装

a.sources.r1.channels = c1

a.sinks.k1.channel = c1

Spark安装配置

1、上传解压，配置环境变量 配置bin目录

解压

tar -xvf spark-2.4.5-bin-hadoop2.6.tgz

重命名

mv spark-2.4.5-bin-hadoop2.6 spark-2.4.5

配置环境变量

vim /etc/profile

export SPARK\_HOME=/usr/local/soft/spark-2.4.5

export PATH=$PATH:$SPARK\_HOME/bin

source /etc/profile

2、修改配置文件 conf(cd /usr/local/soft/spark-2.4.5/conf)

cp spark-env.sh.template spark-env.sh

增加配置(vim spark-env.sh)

export SPARK\_MASTER\_IP=***master2021***

export SPARK\_MASTER\_PORT=7077

export SPARK\_WORKER\_CORES=2

export SPARK\_WORKER\_INSTANCES=1

export SPARK\_WORKER\_MEMORY=2g

export JAVA\_HOME=/usr/local/soft/jdk1.8.0\_171

master相当于RM worker相当于NM

增加从节点配置

cp slaves.template slaves

vim slaves

***node12021***

***node22021***

增加从节点

3、复制到其它节点

scp -r spark-2.4.5 node12021:`pwd`

scp -r spark-2.4.5 node22021:`pwd`

4、在主节点执行启动命令

启动集群，在master中执行

一定要进入到(cd /usr/local/soft/spark-2.4.5/sbin)

./start-all.sh(不在sbin目录下)

./sbin/start-all.sh（在sbin目录下）

http://master:8080/ 访问spark ui

1. standalone client模式 日志在本地输出，一般用于上线前测试(bin/下执行)

需要进入到spark-examples\_2.11-2.4.5.jar 包所在的目录下执行

cd /usr/local/soft/spark-2.4.5/examples/jars

提交spark任务

spark-submit --class org.apache.spark.examples.SparkPi --master spark://master:7077 --executor-memory 512m --total-executor-cores 1 spark-examples\_2.11-2.4.5.jar 100

2. standalone cluster模式 上线使用，不会再本地打印日志

spark-submit --class org.apache.spark.examples.SparkPi --master spark://master:7077 --executor-memory 512M --total-executor-cores 1 --deploy-mode cluster spark-examples\_2.11-2.4.5.jar 100

spark-shell spark 提供的一个交互式的命令行，可以直接写代码(使用:q退出spark-shell)

spark-shell master spark://master:7077