前言

大数据综合项目实训

团队: hello world

组长: 袁帅

成员: 袁帅 艾丽思 柏彬彬 施俊杰 欧阳瑞頔 彭佳欢

本文编撰: 彭佳欢

1. 初始化环境搭建

1.1 大数据集群所需的组件说明

1. 安装包上传的路径

   master节点的/opt/software目录下

2. 组件安装路径

   /usr/local/src

3. 组件版本说明:

组件	版本	下载地址
centos	7.9	https://www.centos.org/download/
jdk	1.8	https://www.oracle.com/java/technologies/javase/javase8u211-later-archive-downloads.html
mysql	8	https://dev.mysql.com/downloads/mysql/
hadoop	3.3.2	http://archive.apache.org/dist/hadoop/core/
zookeeper	3.7.1	https://archive.apache.org/dist/zookeeper/
hive	3.1.3	https://archive.apache.org/dist/hive/
mysql-connector-java	8.0.27	https://repo1.maven.org/maven2/mysql/mysql-connector-java/8.0.27/mysql-connector-java-8.0.27.jar
spark	3.3.0	https://archive.apache.org/dist/spark/
sqoop	1.4.7	https://archive.apache.org/dist/sqoop/

1.2 快速安装3台虚拟机

1. 基础环境:

   window10, VMware16, centos7.9

2. 具体安装步骤:

   可先创建一台模板虚拟机，然后再在此基础上克隆出三台虚拟机，

   - VMware16安装centos7.9，略.....
   - 修改主机名

     ```
     hostnamectl set-hostname hadoop100
     ```

   - 设置静态ip

     ```
     # 1. 编辑文件
         vi /etc/sysconfig/network-scripts/ifcfg-ens33
     # 2. 修改内容
         BOOTPROTO=static
         ONBOOT=yes

         IPADDR=192.168.137.10  # ip地址
         GATEWAY=192.168.137.2  # 网关
         NETMASK=255.255.255.0 # 子网掩码
         DNS1=8.8.8.8 #dns
     ```

- 重启虚拟机

```
reboot
```

- 使用xshell工具连接该节点
- 关闭防火墙

```
systemctl status firewalld # 查看防火墙状态
systemctl disable firewalld # 启动防火墙
systemctl stop firewalld # 关闭防火墙
systemctl disable firewalld # 禁止开机自启
systemctl enable firewalld # 开机自启
```

- 配置阿里yum源

```
# 1. 执行命令
    curl -o /etc/yum.repos.d/CentOS-Base.repo
http://mirrors.aliyun.com/repo/Centos-7.repo
该命表示把Centos-7.repo下载到/etc/yum.repos.d/目录下，如果该目录下有CentOS-
Base.repo，则会自动覆盖。
# 2. 缓存清理
    yum clean cache
# 3. 生成缓存
    yum makecache
# 4. 测试
    [root@hadoop100 bin]# yum install wget
    已加载插件：fastestmirror
    Loading mirror speeds from cached hostfile
     * base: `mirrors.aliyun.com`
     * extras: `mirrors.aliyun.com`
     * updates: mirrors.aliyun.com
```

- 安装JDK

`hadoop等组件依赖于jdk，所以在此处直接将jdk安装好`

```
# 解压安装
tar -zxvf /opt/software/jdk-8u301-linux-x64.tar.gz -C /usr/local/src/

cd /usr/local/src/

mv jdk1.8.0_301/ java8

# 配置环境变量
vim /etc/profile

export JAVA_HOME=/usr/local/src/java8
export PATH=$PATH:$JAVA_HOME/bin

# 刷新配置文件使环境变量生效
source /etc/profile
# 查看环境变量是否生效
[root@hadoop100 bin]# java -version
java version "1.8.0_301"
Java(TM) SE Runtime Environment (build 1.8.0_301-b09)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 25.301-b09, mixed mode)
```

- **克隆3台节点**

1.3 修改主机名以及静态ip

1. 节点规划

> 在上面配置的模板虚拟机的基础上克隆出三台节点

ip	hostname	username	password
192.168.137.11	master	root	123456
192.168.137.12	slave1	root	123456
192.168.137.13	slave2	root	123456

2. 分别修改3台节点的主机名以及设置静态ip

   注意要与节点规划的主机名/ip保持一致

```
==================`master`====================
1. hostnamectl set-hostname master  # 修改主机名

2. vi /etc/sysconfig/network-scripts/ifcfg-ens33
    IPADDR=192.168.137.11  # 设置静态ip
```

```
==================`slave1`====================
1. hostnamectl set-hostname slave1  # 修改主机名

2. vi /etc/sysconfig/network-scripts/ifcfg-ens33
    IPADDR=192.168.137.12  # 设置静态ip
```

```
==================`slave2`====================
1. hostnamectl set-hostname slave2  # 修改主机名

2. vi /etc/sysconfig/network-scripts/ifcfg-ens33
    IPADDR=192.168.137.13  # 设置静态ip
```

3. 重启三台节点, 查看配置是否生效, 并是用xshell工具连接

1.4 配置host映射以及免密登陆

1. 配置host映射

   分别在**3个节点**的/etc/hosts文件中添加如下内容

```
vim /etc/hosts

192.168.137.11 master
192.168.137.12 slave1
192.168.137.13 slave2
```

2. 免密登陆

- **每台主机都执行以下指令**, 生成密钥

  ```
  ssh-keygen -t rsa # 输入之后，敲三次回车即可
  ```

- 分发公钥

  每台主机都执行以下指令(在分布式系统中，每台主机与自身也需要配置免密登录)

  ```
  ssh-copy-id master
  ssh-copy-id slave1
  ssh-copy-id slave2
  ```

- ssh免密登陆配置成功后,会生成**/root/.ssh**目录

  ```
  [root@master ~]# cd /root/.ssh
  [root@master .ssh]# ll
  总用量 16
  -rw-------. 1 root root 1179 9月  20 06:38 authorized_keys # 记录允许哪个主机免密访问本机
  -rw-------. 1 root root 1675 9月  20 06:34 id_rsa  # 私钥
  -rw-r--r--. 1 root root  393 9月  20 06:34 id_rsa.pub  # 公钥
  -rw-r--r--. 1 root root  549 9月  20 06:38 known_hosts
  ```

- 测试免密是否配置成功`ssh slave1`

  ```
  [root@master ~]# ssh slave1
  Last failed login: Tue Sep 20 11:27:33 CST 2022 on tty1
  There was 1 failed login attempt since the last successful login.
  Last login: Tue Sep 20 06:40:06 2022 from slave2
  [root@slave1 ~]#
  ```

1.5 编写文件分发shell脚本

> 目的: 为了方便个各节点相互之间的文件分发
>
> 前提条件: 配置好免密登陆, **并且3台节点均下载yum install rsync -y**

1. 在/root/目录下创建子目录

   mkdir /root/bin

2. 编写shell脚本xsync

   vim /root/bin/xsync

   文件具体内容见: 附件_配置文件及shell脚本.md

3. 赋予可执行权限

   chmod +x /root/bin/xsync

4. 测试文件是否能够分发到其他节点

   命令: xsync 文件名

2. Hadoop HA 搭建

> **前提条件: 安装配置好zookeeper**, 安装配置见本文下一节
>
> 彭某友情提示: 在开始搭建集群之前, 请做好**快照**, 谨防意外发生

==========节点规划==========

master	**slave1**	**slave2**
NameNode	**NameNode**	**NameNode**
DataNode	**DataNode**	**DataNode**
ResourceManager	**ResourceManager**	**ResourceManager**
NodeManager	**NodeManager**	**NodeManager**
JobHistoryServer		
QuorumPeerMain	QuorumPeerMain	QuorumPeerMain
JournalNode	JournalNode	JournalNode
DFSZKFailoverController	DFSZKFailoverController	DFSZKFailoverController

2.1 解压文件&&配置环境变量

> 以下在master节点操作

1. 解压文件

```
tar -zxvf hadoop-3.3.2.tar.gz -C /usr/local/src/

cd /usr/local/src/

mv hadoop-3.3.2/ hadoop
```

2. 配置环境变量

```
vim /etc/profile.d/bigdata_env.sh

# 追加
export HADOOP_HOME=/usr/local/src/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin

# 是环境变量生效
source /etc/profile
```

2.2 修改配置文件

> hadoop配置文件位于：/usr/local/src/hadoop/etc/hadoop

1. workers

```
master
slave1
slave2
```

2. hadoop-env.sh

```
# 修改
export JAVA_HOME=/usr/local/src/java8
export HADOOP_HOME=/usr/local/src/hadoop

# 以下配置是由于在下面格式化集群时报错而额外添加的配置
export HDFS_NAMENODE_USER=root
export HDFS_DATANODE_USER=root
export HDFS_SECONDARYNAMENODE_USER=root
export YARN_RESOURCEMANAGER_USER=root
export YARN_NODEMANAGER_USER=root
export HDFS_JOURNALNODE_USER=root
export HDFS_ZKFC_USER=root
```

3. core-site.xml

   文件具体内容见: 附件_配置文件及shell脚本.md

4. hdfs-site.xml

   文件具体内容见: 附件_配置文件及shell脚本.md

5. mapred-site.xml

   文件具体内容见: 附件_配置文件及shell脚本.md

6. yarn-site.xml

   文件具体内容见: 附件_配置文件及shell脚本.md

2.3 创建shell脚本, 用于格式化集群

> **以下文件全部放置到/root/bin目录下**

1. **xcall**: 远程执行bash指令
2. **jpsall**: 查看所有节点服务器所有正在运行的Java进程
3. **xsync**: 集群同步文件
4. **zk**: 开关ZooKeeper集群
   - zk start 启动zookeeper集群
   - zk stop 停止zookeeper集群
   - zk status 查看zookeeper集群状态
5. **format-ha**: 初始化HA集群
6. **hadoop-ha**: 开关HA集群
   - hadoop-ha start 一键启动
   - hadoop-ha stop 一键停止

以上文件具体内容见: 附件_配置文件及shell脚本.md

编写完成后,**需要赋予可执行权限[命令: chmod +x 文件名],** 可以选择是否分发到其他两个节点

2.4 初始化hadoop HA集群

1. 分发hadoop到其他两个节点

```
xsync /usr/local/src/hadoop/
```

2. 配置环境变量

```
# 分发环境变量到其他节点上
xsync /etc/profile.d/bigdata_env.sh

# 分别在"两节点"中刷新环境变量
source /etc/profile
```

3. 执行脚本格式化集群

```
# 直接执行shell脚本, 一键初始化hadoop ha集群
format-ha
```

2.5 启动hadoop ha集群

1. 启动

```
hadoop-ha start
```

2. 查看个节点java进程

```
[root@master hadoop]# jpsall
=========== master ==========
7200 JournalNode
8384 Jps
8148 NodeManager
7416 NameNode
7992 DFSZKFailoverController
8297 JobHistoryServer
7099 QuorumPeerMain
7547 DataNode
8075 ResourceManager
=========== slave1 ==========
3921 DataNode
3762 JournalNode
3668 QuorumPeerMain
3844 NameNode
4533 ResourceManager
4662 NodeManager
4840 Jps
4123 DFSZKFailoverController
=========== slave2 ==========
3584 DataNode
3425 JournalNode
4241 Jps
3507 NameNode
3862 ResourceManager
3338 QuorumPeerMain
3786 DFSZKFailoverController
3934 NodeManager
[root@master hadoop]
```

3. 查看hadoop 集群报告, 是否有错误

```
[root@master hadoop]# hdfs dfsadmin -report
Configured Capacity: 54716792832 (50.96 GB)
```

```
Present Capacity: 42979905536 (40.03 GB)
DFS Remaining: 42979893248 (40.03 GB)
DFS Used: 12288 (12 KB)
DFS Used%: 0.00%
Replicated Blocks:
    Under replicated blocks: 0
    Blocks with corrupt replicas: 0
    Missing blocks: 0
    Missing blocks (with replication factor 1): 0
    Low redundancy blocks with highest priority to recover: 0
    Pending deletion blocks: 0
Erasure Coded Block Groups:
    Low redundancy block groups: 0
    Block groups with corrupt internal blocks: 0
    Missing block groups: 0
    Low redundancy blocks with highest priority to recover: 0
    Pending deletion blocks: 0

-------------------------------------------------
Live datanodes (3):

Name: 192.168.137.11:9866 (master)
Hostname: master
Decommission Status : Normal
Configured Capacity: 18238930944 (16.99 GB)
DFS Used: 4096 (4 KB)
Non DFS Used: 4346261504 (4.05 GB)
DFS Remaining: 13892665344 (12.94 GB)
DFS Used%: 0.00%
DFS Remaining%: 76.17%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 0
Last contact: Tue Sep 20 14:16:31 CST 2022
Last Block Report: Tue Sep 20 14:13:40 CST 2022
Num of Blocks: 0


Name: 192.168.137.12:9866 (slave1)
Hostname: slave1
Decommission Status : Normal
Configured Capacity: 18238930944 (16.99 GB)
DFS Used: 4096 (4 KB)
Non DFS Used: 3695837184 (3.44 GB)
DFS Remaining: 14543089664 (13.54 GB)
DFS Used%: 0.00%
DFS Remaining%: 79.74%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 0
Last contact: Tue Sep 20 14:16:32 CST 2022
Last Block Report: Tue Sep 20 14:13:38 CST 2022
```

```
Num of Blocks: 0


Name: 192.168.137.13:9866 (slave2)
Hostname: slave2
Decommission Status : Normal
Configured Capacity: 18238930944 (16.99 GB)
DFS Used: 4096 (4 KB)
Non DFS Used: 3694788608 (3.44 GB)
DFS Remaining: 14544138240 (13.55 GB)
DFS Used%: 0.00%
DFS Remaining%: 79.74%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 0
Last contact: Tue Sep 20 14:16:31 CST 2022
Last Block Report: Tue Sep 20 14:13:37 CST 2022
Num of Blocks: 0
```

2.6 访问web端口是否正常

建议在window C:\Windows\System32\drivers\etc\hosts中配置主机地址映射, 则可以使用**主机名:ip**访问

- hdfs: ip:9870　(hdfs处于active状态节点的ip)
- yarn: ip:8088　(yarn处于active状态节点的ip)
- JobHistory(历史服务器): http://192.168.137.11:19888/

3. zookeeper的安装配置

3.1 解压文件&&配置环境变量

1. 解压安装

```
# 解压
tar -zxvf /opt/software/apache-zookeeper-3.7.1-bin.tar.gz -C /usr/local/src/
# 重命名
cd /usr/local/src/
mv apache-zookeeper-3.7.1-bin/ zookeeper
```

2. 配置环境变量

```
# 创建一个单独的环境变量配置文件，将以后的组件环境变量全部追加到这个文件中
vim /etc/profile.d/bigdata_env.sh

export ZOOKEEPER_HOME=/usr/local/src/zookeeper
export PATH=$PATH:$ZOOKEEPER_HOME/bin
```

3.2 修改配置文件

1. 复制zookeeper/conf 这个目录下的 zoo_sample.cfg 为 zoo.cfg

```
mv zoo_sample.cfg zoo.cfg
```

2. 在zookeeper目录下创建zkData目录

```
mkdir /usr/local/src/zookeeper/zkData
```

3. 在刚创建的zkData目录下新建myid文件

```
vim myid

# 并追加内容
2
```

4. 修改zoo.cfg文件

```
vim zoo.cfg

# 修改
# dataDir：保存Zookeeper中的数据，注意：默认的tmp目录，容易被Linux系统定期删除，所以一般不用默认的tmp目录。
dataDir=/usr/local/src/zookeeper/zkData
# 追加
#######################cluster##########################
# 其中2,3,4是和myid文件对应的
server.2=master:2888:3888
server.3=slave1:2888:3888
server.4=slave2:2888:3888
```

5. 分发zookeeper到slave1, slave2

```
xsync /usr/local/src/zookeeper/
```

6. 在其他节点修改zkData/myid文件,

```
=======slave1======
vim myid
# 修改2
3

=======slave2======
vim myid
# 修改2
4
```

3.3 配置zookeeper群起脚本

1. 创建文件: `/root/bin/zk`
2. 编辑 `/root/bin/zk` 文件内容

   文件具体内容见: 附件_配置文件及shell脚本.md
3. 授予可执行权限

chmod +x /root/bin/zk

4. 命令:
    - zk start 启动zookeeper集群
    - zk stop 停止zookeeper集群
    - zk status 查看zookeeper集群状态
5. 可选择性将该`zk`文件分发到其他节点

4. hive 远程模式

4.1 安装mysql8数据库(master节点)

```
# 1 解压mysql包
    tar -xvf /opt/software/mysql-8.0.15-1.el7.x86_64.rpm-bundle.tar -C ./

# 2 安装前注意:
    # -- 卸载mariadb,否则mysql会出现冲突
    # -- 执行命令rpm -qa |grep mariadb
    # rpm -e mariadb-libs-5.5.68-1.el7.x86_64 --nodeps

# 3 使用rpm依次安装如下仅4个rpm包即可:
    rpm -ivh mysql-community-common-8.0.15-1.el7.x86_64.rpm
    rpm -ivh mysql-community-libs-8.0.15-1.el7.x86_64.rpm
    rpm -ivh mysql-community-client-8.0.15-1.el7.x86_64.rpm
    rpm -ivh mysql-community-server-8.0.15-1.el7.x86_64.rpm

# 4 启动mysql服务:
    systemctl start mysqld

# 5 查看mysql服务是否启动:
    systemctl status mysqld

# 6 查看mysql初始密码:
    grep 'temporary password' /var/log/mysqld.log

# 7 登陆mysql
    mysql -uroot -p

# 8 修改msyql密码安全策略:
    set global validate_password.policy=0;  # mysql8版本
    set global validate_password_policy=0;  # mysql5版本

# 9 修改mysql密码长度:
    set global validate_password.length=1;  # mysql8版本
    set global validate_password_length=1;  # mysql5版本

# 10 修改mysql密码:
    ALTER USER 'root'@'localhost' IDENTIFIED BY '你的新密码';

# 11 允许远程访问:
    use mysql;
    update user set host = '%' where user ='root';

# 12 刷新权限:
    flush privileges;
```

```
# 13 退出重启mysql服务且关闭防火墙
    exit;
# 14 安装完成后，可以删除software目录下的mysql安装包
```

4.2 安装hive&&配置环境变量

1. 安装hive (master节点)

```
tar -zxvf /opt/software/apache-hive-3.1.3-bin.tar.gz -C /usr/local/src/

mv /usr/local/src/apache-hive-3.1.3-bin/ /usr/local/src/hive
```

2. 配置环境变量

```
vim /etc/profile.d/bigdata_env.sh

export HIVE_HOME=/usr/local/src/hive
export PATH=$PATH:$HIVE_HOME/bin

source /etc/profile
```

4.3 修改hive的配置文件

1. 在hive/conf文件下

   `cp hive-env.sh.template hive-env.sh` 复制出一份hive-env.sh

2. 编辑**hive-env.sh**

```
export HADOOP_HOME=/usr/local/src/hadoop
export HIVE_CONF_DIR=/usr/local/src/hive/conf
export HIVE_AUX_JARS_PATH=/usr/local/src/hive/lib
```

3. 将mysql驱动放到**hive/lib**目录下

```
cp /opt/software/mysql-connector-java-8.0.27.jar /usr/local/src/hive/lib/
```

4. 分发hive到其他节点

```
xsync /etc/profile.d/bigdata_env.sh  # 分发环境变量，记得source一下

xsync /usr/local/src/hive/   # 分发hive
```

5. 这里我们将**slave1作为服务端**, **在slave1节点**添加`/usr/local/src/hive/conf/hive-site.xml`文件

   文件具体内容见: 附件_配置文件及shell脚本.md

6. 在slave1初始化元数据库(**记得提前启动hdfs**)

```
/usr/local/src/hive/bin/schematool -dbType mysql -initSchema
```

7. **master, slave2作为客户端,** 在这两台节点编辑**/usr/local/src/hive/conf/hive-site.xml**文件

   文件具体内容见: 附件_配置文件及shell脚本.md

8. 启动hive

```
# slave1启动metastore服务
nohup hive --service metastore >> /root/myhive.log  2>&1 &  # 后台

# 任意客户端执行hive命令, 连接metastore
hive

==============================================
# slave1启动hiveserver2
nohup hive --service hiveserver2 >> /root/myhive.log  2>&1 & # 后台运行
# 连接
beeline
beeline> !connect jdbc:hive2://slave1:10000
Connecting to jdbc:hive2://slave1:10000
Enter username for jdbc:hive2://slave1:10000: root
Enter password for jdbc:hive2://slave1:10000: ******
```

```
-- 建表测试
create table `emp`(
`dept_no` int,
`addr` string,
`tel` string)
partitioned by (sex string)
row format delimited fields terminated by '\t'
```

5. spark HA on yarn模式

5.1 安装&&配置环境变量

1. 解压安装

```
tar -zxvf /opt/software/spark-3.3.0-bin-hadoop3.tgz -C /usr/local/src/

mv /usr/local/src/spark-3.3.0-bin-hadoop3/ /usr/local/src/spark
```

2. 配置环境变量

```
vim /etc/profile.d/bigdata_env.sh

export SPARK_HOME=/usr/local/src/spark
export PATH=$PATH:$SPARK_HOME/bin/:$SPARK_HOME/sbin

source /etc/profile
```

5.2 修改配置文件

> 配置文件在spark/conf目录下

1. 复制一份`spark-env.sh.template`

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

2. 编辑`spark-env.sh`

```
export JAVA_HOME=/usr/local/src/java8
export YARN_CONF_DIR=/usr/local/src/hadoop/etc/hadoop

#Master 监控页面默认访问端口为 8080，但是可能会和 Zookeeper 冲突，所以改成 8989，也可以自定义，访问 UI 监控页面时请注意
SPARK_MASTER_WEBUI_PORT=8989
export SPARK_DAEMON_JAVA_OPTS="
-Dspark.deploy.recoveryMode=ZOOKEEPER
-Dspark.deploy.zookeeper.url=master,slave1,slave2
-Dspark.deploy.zookeeper.dir=/spark"

# 历史服务
export SPARK_HISTORY_OPTS="
-Dspark.history.ui.port=18080
-Dspark.history.fs.logDirectory=hdfs://mycluster:8020/directory
-Dspark.history.retainedApplications=30"
```

3. 复制一份cp workers.template

```
cp workers.template workers

# 修改内容
master
slave1
slave2
```

4. 配置spark-defaults.conf文件

```
mv spark-defaults.conf.template spark-defaults.conf

vim spark-defaults.conf

# 修改spark-defaults.conf文件内容
spark.eventLog.enabled true
# mycluster表hadoop ha的集群名 需要启动 hadoop 集群，HDFS 上的目录需要提前存在。
spark.eventLog.dir hdfs://mycluster:8020/directory
```

```
# sbin/start-history-server.sh  # 历史服务器启动命令
```

5. 修改 hadoop 配置文件hadoop/etc/hadoop/yarn-site.xml, **并分发到其他节点**

```
<!--追加内容-->
<!--是否启动一个线程检查每个任务正使用的物理内存量，如果任务超出分配值，则直接将其杀掉，默
认
是 true -->
<property>
    <name>yarn.nodemanager.pmem-check-enabled</name>
    <value>false</value>
</property>
<!--是否启动一个线程检查每个任务正使用的虚拟内存量，如果任务超出分配值，则直接将其杀掉，默
认
是 true -->
<property>
    <name>yarn.nodemanager.vmem-check-enabled</name>
    <value>false</value>
</property>
```

分发

```
xsync /usr/local/src/hadoop/etc/hadoop/yarn-site.xml
```

6. 分发spark到其他节点

```
# 分发环境变量,
xsync /etc/profile.d/bigdata_env.sh
# 分发spark
xsync /usr/local/src/spark
#并需要到其他节点source一下环境变量
```

5.3 启动spark ha on yarn集群

1. 启动hadoop集群

```
hadoop-ha

# 由于配置了历史服务器, 所以hdfs事先要创建好directory目录
hadoop fs -mkdir /directory  # `仅第一次启动时执行`
```

2. 启动spark集群

   master

```
/usr/local/src/spark/sbin/start-all.sh  # 启动spark
/usr/local/src/spark/sbin/start-history-server.sh # 启动历史服务器
```

   slave1手动启动master

```
/usr/local/src/spark/sbin/start-master.sh
```

3. 访问web端口查看状态

   spark:

   - master:8989
   - slave1:8989

history:

- master:18080

4. 停止集群

   彭某懒得写了, 自己在sbin目录下翻命令, 此处**略**

5.4 一键启动spark脚本(spark.sh)

文件具体内容见: 附件_配置文件及shell脚本.md

启动: spark.sh start

停止: spark.sh stop

6. 安装sqoop

> 仅在master安装即可

6.1 安装&&配置环境变量

1. 安装

```
tar -zxvf /opt/software/sqoop-1.4.7.bin__hadoop-2.6.0.tar.gz -C
/usr/local/src/

mv /usr/local/src/sqoop-1.4.7.bin__hadoop-2.6.0/ /usr/local/src/sqoop
```

2. 配置环境变量

```
vim /etc/profile.d/bigdata_env.sh

#追加内容
export SQOOP_HOME=/usr/local/src/sqoop
export PATH=$PATH:$SQOOP_HOME/bin

source /etc/profile
```

6.2 修改配置文件

1. 备份出一份sqoop-env.sh

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

2. 修改sqoop-env.sh内容\

```
mv  sqoop-env.sh


export HADOOP_COMMON_HOME=/usr/local/src/hadoop
export HADOOP_MAPRED_HOME=/usr/local/src/hadoop
export HIVE_HOME=/usr/local/src/hive
export ZOOCFGDIR=/usr/local/src/zookeeper
```

3. 在sqoop/lib目录下添加mysql连接驱动

```
cp /opt/software/mysql-connector-java-8.0.27.jar /usr/local/src/sqoop/lib/
```

6.3 测试是否安装成功

1. sqoop-version
2. 测试连接mysql

```
sqoop list-databases --connect jdbc:mysql://master:3306/ --username root --
password 123456

# 报如下错误
Exception in thread "main" java.lang.NoClassDefFoundError:
org/apache/commons/lang/StringUtils

# 可能commons-lang包版本不支持
[root@master lib]# ll commons-
commons-codec-1.4.jar       commons-io-1.4.jar          commons-lang3-
3.4.jar
commons-compress-1.8.1.jar  commons-jexl-2.1.1.jar      commons-logging-
1.1.1.jar

#解决方法：根据如下链接，下载commons-lang2的版本，并将commons-lang-2.6.jar上传到lib
文件夹下
https://dlcdn.apache.org//commons/lang/binaries/commons-lang-2.6-bin.zip

#重新执行 ok
root@master lib]# sqoop list-databases --connect jdbc:mysql://master:3306/ -
-username root --password 123456
Warning: /usr/local/src/sqoop/../hbase does not exist! HBase imports will
fail.
Please set $HBASE_HOME to the root of your HBase installation.
Warning: /usr/local/src/sqoop/../hcatalog does not exist! HCatalog jobs will
fail.
Please set $HCAT_HOME to the root of your HCatalog installation.
Warning: /usr/local/src/sqoop/../accumulo does not exist! Accumulo imports
will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
2022-09-21 00:13:54,565 INFO sqoop.Sqoop: Running Sqoop version: 1.4.7
2022-09-21 00:13:54,639 WARN tool.BaseSqoopTool: Setting your password on
the command-line is insecure. Consider using -P instead.
2022-09-21 00:13:54,777 INFO manager.MySQLManager: Preparing to use a MySQL
streaming resultset.
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver
class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered
via the SPI and manual loading of the driver class is generally unnecessary.
mysql
information_schema
performance_schema
sys
hive
```