**Hadoop集群搭建手册**

**（hive，impala，kudu）**

目录

[1 环境信息 3](#_Toc47360564)

[1.1 版本信息 3](#_Toc47360565)

[1.2 服务器信息 3](#_Toc47360566)

[2 安装部署 3](#_Toc47360567)

[2.1 服务器配置 3](#_Toc47360568)

[2.1.1 关闭selinux 3](#_Toc47360569)

[2.1.2 关闭防火墙 4](#_Toc47360570)

[2.1.3 修改主机名 4](#_Toc47360571)

[2.1.4 配置host 4](#_Toc47360572)

[2.1.5 修改限制 4](#_Toc47360573)

[2.1.6 配置免密登陆 4](#_Toc47360574)

[2.2 Hadoop组件安装 5](#_Toc47360575)

[2.2.1 安装ntp 5](#_Toc47360576)

[2.2.2 安装hadoop 5](#_Toc47360577)

[2.2.3 安装mysql（单机版hadoop01） 12](#_Toc47360578)

[2.2.4 安装hive（hadoop01） 14](#_Toc47360579)

[2.2.5 安装kudu 16](#_Toc47360580)

[2.2.6 安装impala(hadoop01安装) 19](#_Toc47360581)

[2.2.7 安装hue 22](#_Toc47360582)

# 环境信息

## 版本信息

操作系统：oracle linux7.6

Java：jdk1.8

Mysql：5.7.28

cdh6.3.2： hadoop3.0.0

hbase2.1.0

hive2.1.1

impala3.2.0

kudu1.10.0

## 服务器信息

|  |  |  |  |
| --- | --- | --- | --- |
| ip | 主机名 | cpu | 内存 |
| 198.162.1.10 | hadoop01 | 24 | 96g |
| 198.162.1.11 | hadoop02 | 24 | 96g |
| 198.162.1.12 | hadoop03 | 24 | 96g |

# 安装部署

本例安装cdh6.3.2版本，下载地址：

https://archive.cloudera.com/cdh6/6.3.2/redhat7/yum/RPMS/

## 服务器配置

### 关闭selinux

#将 SELINUX 的值设为 disabled

vi /etc/selinux/config，

SELINUX=disabled

#关闭当前已开启的 SELinux 使用如下命令：

setenforce 0

### 关闭防火墙

systemctl stop firewalld.service

systemctl disable firewalld.service

### 修改主机名

#三台主机分别执行

hostnamectl set-hostname hadoop01

hostnamectl set-hostname hadoop02

hostnamectl set-hostname hadoop03

### 配置host

vi /etc/hosts

198.162.1.10 hadoop01

198.162.1.11 hadoop02

198.162.1.12 hadoop03

### 修改限制

vi /etc/security/limits.conf

# End of file

\* soft nofile 1000000

\* hard nofile 1000000

### 配置免密登陆

#三个节点都执行

ssh-keygen -t rsa

ssh-copy-id hadoop01

ssh-copy-id hadoop02

ssh-copy-id hadoop03

## Hadoop组件安装

### 安装jdk(三个节点执行)

将jdk1.8.0\_241.zip上传到/usr目录

#卸载openjdk

rpm -qa | grep jdk

rpm -e--nodeps上面查询出的openjdk

#安装上传的jdk

cd /usr

unzip jdk1.8.0\_241.zip

vi /etc/profile

export JAVA\_HOME=/usr/jdk1.8.0\_241

export PATH=${JAVA\_HOME}/jre/bin:${JAVA\_HOME}/bin:$PATH

#使环境变量生效

source/etc/profile

### 安装ntp

##安装ntp包，三个节点执行

yum install ntp ntpdate -y

##hadoop01节点ntp配置

vi /etc/ntp.conf

server 127.127.1.0

fudge 127.127.1.0 stratum 8

##hadoop02和hadoop03节点配置

#server 0.rhel.pool.ntp.org iburst

#server 1.rhel.pool.ntp.org iburst

#server 2.rhel.pool.ntp.org iburst

#server 3.rhel.pool.ntp.org iburst

server 147.12.213.132 iburst

##启动ntp服务

systemctl start ntpd

##设置开机启动

systemctl enable ntpd

### 安装hadoop

上传cdh包到三个服务器的/hadoop目录

#### 安装依赖包（三个节点执行）

yum -y install lsb nc portmap

#### 安装hadoop包（三个节点执行）

cd /hadoop/cdh6.3.2/RPMS/noarch

rpm –ivh bigtop\*

rpm -ivh avro\*

rpm -ivh parquet\* --nodeps

cd /hadoop/cdh6.3.2/RPMS/x86\_64

rpm -ivh bigtop\*

rpm -ivh zookeeper-3.4.5+cdh6.3.2-1605554.el7.x86\_64.rpm

cd /hadoop/cdh6.3.2/RPMS/x86\_64

rpm -ivh hadoop-3.0.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh hadoop-hdfs-3.0.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh hadoop-yarn-3.0.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh hadoop-mapreduce-3.0.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh hadoop-client-3.0.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh hadoop-libhdfs-3.0.0+cdh6.3.2-1605554.el7.x86\_64.rpm

#### 修改配置文件（hadoop01节点修改）

1. 修改hadoop-env.sh

vi /etc/hadoop/conf/hadoop-env.sh

export JAVA\_HOME=/usr/jdk1.8.0\_241

export HDFS\_DATANODE\_USER=root

export HDFS\_NAMENODE\_USER=root

export HDFS\_SECONDARYNAMENODE\_USER=root

export YARN\_RESOURCEMANAGER\_USER=root

export YARN\_NODEMANAGER\_USER=root

export HDFS\_DATANODE\_SECURE\_USER=root

export HADOOP\_CLIENT\_OPTS="-Xmx20g -Xms10g $HADOOP\_CLIENT\_OPTS"

1. 修改core-site.xml

vi /etc/hadoop/conf/core-site.xml

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://hadoop01:9000</value>

</property>

<property>

<name>io.file.buffer.size</name>

<value>131072</value>

</property>

<property>

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

<value>file:/hadoop/temp</value>

</property>

<property>

<name>hadoop.proxyuser.root.hosts</name>

<value>\*</value>

</property>

<property>

<name>hadoop.proxyuser.root.groups</name>

<value>\*</value>

</property>

<property>

<name>hadoop.proxyuser.sqoop2.hosts</name>

<value>\*</value>

</property>

<property>

<name>hadoop.proxyuser.sqoop2.groups</name>

<value>\*</value>

</property>

</configuration>

1. 修改hdfs-site.xml文件

vi /etc/hadoop/conf/hdfs-site.xml

<configuration>

<property>

<name>dfs.namenode.secondary.http-address</name>

<value>hadoop01:9001</value>

</property>

<property>

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

<value>file:/hadoop/hdfs/name</value>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>file:/hadoop/hdfs/data</value>

</property>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.webhdfs.enabled</name>

<value>true</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

<property>

<name>dfs.web.ugi</name>

<value>supergroup</value>

</property>

</configuration>

1. 修改mapred-site.xml文件

vi /etc/hadoop/conf/mapred-site.xml

<configuration>

<property>

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

<value>yarn</value>

</property>

<property>

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

<value>hadoop01:10020</value>

</property>

<property>

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

<value>hadoop01:19888</value>

</property>

<property>

<name>mapreduce.map.memory.mb</name>

<value>1024</value>

</property>

<property>

<name>mapreduce.reduce.memory.mb</name>

<value>4096</value>

</property>

<property>

<name>yarn.app.mapreduce.am.env</name>

<value>HADOOP\_MAPRED\_HOME=/usr/lib/hadoop</value>

</property>

<property>

<name>mapreduce.map.env</name>

<value>HADOOP\_MAPRED\_HOME=/usr/lib/hadoop</value>

</property>

<property>

<name>mapreduce.reduce.env</name>

<value>HADOOP\_MAPRED\_HOME=/usr/lib/hadoop</value>

</property>

<property>

<name>mapreduce.jobhistory.done-dir</name>

<value>${yarn.app.mapreduce.am.staging-dir}/history/done</value>

</property>

<property>

<name>mapreduce.jobhistory.intermediate-done-dir</name>

<value>${yarn.app.mapreduce.am.staging-dir}/history/done\_intermediate</value>

</property>

<property>

<name>yarn.app.mapreduce.am.staging-dir</name>

<value>/tmp/hadoop-yarn/staging</value>

</property>

</configuration>

1. 修改yarn-site.xml文件

vi /etc/hadoop/conf/yarn-site.xml

<configuration>

<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>

<property>

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

<value>hadoop01:8032</value>

</property>

<property>

<name>yarn.resourcemanager.scheduler.address</name>

<value>hadoop01:8030</value>

</property>

<property>

<name>yarn.resourcemanager.resource-tracker.address</name>

<value>hadoop01:8031</value>

</property>

<property>

<name>yarn.resourcemanager.admin.address</name>

<value>hadoop01:8033</value>

</property>

<property>

<name>yarn.resourcemanager.webapp.address</name>

<value>hadoop01:8088</value>

</property>

<property>

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

<value>true</value>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-mb</name>

<value>4096</value>

</property>

<property>

<name>yarn.scheduler.minimum-allocation-mb</name>

<value>4096</value>

</property>

<property>

<name>yarn.nodemanager.resource.memory-mb</name>

<value>20480</value>

</property>

<property>

<name>yarn.nodemanager.resource.cpu-vcores</name>

<value>20</value>

</property>

<property>

<name>mapred.child.java.opts</name>

<value>-Xmx1024m</value>

</property>

<property>

<description>Classpath for typical applications.</description>

<name>yarn.application.classpath</name>

<value>

/etc/hadoop/conf:/usr/lib/hadoop/lib/\*:/usr/lib/hadoop/.//\*:/usr/lib/hadoop-hdfs/./:/usr/lib/hadoop-hdfs/lib/\*:/usr/lib/hadoop-hdfs/.//\*:/usr/lib/hadoop-yarn/lib/\*:/usr/lib/hadoop-yarn/.//\*:/usr/lib/hadoop-mapreduce/lib/\*:/usr/lib/hadoop-mapreduce/.//\*

</value>

</property>

</configuration>

1. 修改workers文件

vi /etc/hadoop/conf/workers

hadoop01

hadoop02

hadoop03

1. 修改hdfs文件(三个节点执行)

vi/usr/lib/hadoop-hdfs/bin/hdfs

HADOOP\_SHELL\_EXECNAME="root"

1. 配置环境变量

vi /etc/profile

export JAVA\_LIBRARY\_PATH=/usr/lib/hadoop/lib/native

#### 拷贝配置文件

#将1节点的配置文件拷贝到其他节点

cd /etc/hadoop/conf

scp \* hadoop02: /etc/hadoop/conf

scp \* hadoop03: /etc/hadoop/conf

#### Hdfs格式化

#hadoop01执行

hadoop namenode -format

#提示：successfully formatted表示格式化成功

#### 启动hdfs

由于hadoop的rpm安装版没有启动命令，需要自行上传启动脚本



1. 启动hdfs

#将start-dfs.sh和stop-dfs.sh上传到/usr/lib/hadoop-hdfs/sbin目录下

cd /usr/lib/hadoop-hdfs/sbin

chmod 777 \*dfs.sh

cp -r /usr/lib/hadoop/libexec /usr/lib/hadoop-hdfs (三个节点执行)

#启动hdfs

./start-dfs.sh

#启动成功后hadoop01节点执行jps命令会有如下进程

NameNode

SecondaryNameNode

ResourceManager

DataNode

#启动成功后hadoop01节点执行jps命令会有如下进程

DataNode

#### 启动yarn

由于hadoop的rpm安装版没有启动命令，需要自行上传启动脚本



1. 启动yarn

#将start-yarn.sh和stop-yarn.sh上传到/usr/lib/hadoop-yarn/sbin目录下

cd /usr/lib/hadoop-yarn/sbin

chmod 777 \*yarn.sh

cp -r /usr/lib/hadoop/libexec /usr/lib/hadoop-yarn (三个节点执行)

#启动yarn（hadoop01节点执行）

./start-yarn.sh

#启动成功后所有节点执行jps命令会有如下进程

NodeManager

#### 启动jobhistory

cd /usr/lib/hadoop-mapreduce/sbin

mr-jobhistory-daemon.sh start historyserver

#访问地址

http://hadoop01:19888

#### 访问地址

http://100.12.72.87:50070/

3.0以后http://100.12.72.87:9870/

### 安装mysql（单机版hadoop01）

#### 卸载mariadb数据库

#linux7.6自带mariadb数据库，需要先卸载

rpm -qa | grep mariadb

rpm -e grep -nodeps–上一条命令查询的结果

#### 安装rpm包

将mysql-5.7.28-1.el7.x86\_64.rpm-bundle.tar.gz上传到hadoop01的/hadoop/mysql目录

cd /hadoop/mysql

tar –zxvf mysql-5.7.28-1.el7.x86\_64.rpm-bundle.tar.gz

yum-yinstallperl\*

rpm -Uvhmysql\*

#### 修改配置

vi /etc/my.cnf

datadir=/hadoop/mysql

socket=/hadoop/mysql/mysql.sock

log-error=/var/log/mysqld.log

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

max\_connections=16384

character\_set\_server = utf8

vi /usr/lib/systemd/system/mysqld.service

LimitNOFILE = 65535

#重新加载mysqld.service

systemctl daemon-reload

#### 启动mysql

#启动命令

systemctlstartmysqld

#由于修改了socket路径，启动会报错，需要将mysql目录下的文件做软链接到/var/lib/mysql目录下然后重启mysql

ln -s /hadoop/mysql/mysql.sock /var/lib/mysql/mysql.sock

systemctlrestartmysqld

#设置开机自启

systemctlenablemysqld

#### 修改mysql访问配置

#查看mysql root初始化密码

grep 'temporary password' /var/log/mysqld.log

#输出mysql用户的初始root密码

# 2020-07-20T07:53:58.409072Z 1 [Note] A temporary password is generated for root@localhost: VS?ebT%st1xd

#登陆mysql

mysql -uroot -p –输入上一步查询得到的密码

#修改root用户密码（登陆mysql后执行）

set global validate\_password\_policy=0;

set global validate\_password\_length=1;

SET PASSWORD=PASSWORD('123456');

#允许mysql远程访问

use mysql;

update user set host='%' where user='root' and host='localhost';

#使以上配置立即生效

flush privileges;

#创建hive数据库

create database hive;

alter database hive character set utf8;

### 安装hive（hadoop01）

#### 安装依赖

cd /hadoop/cdh6.3.2/RPMS/noarch

rpm -ivh sentry-2.1.0+cdh6.3.2-1605554.el7.noarch.rpm --nodeps

rpm -ivh solr-7.4.0+cdh6.3.2-1605554.el7.noarch.rpm --nodeps

#### 安装hive包

cd /hadoop/cdh6.3.2/RPMS/noarch

rpm -ivh hive-server-2.1.1+cdh6.3.2-1605554.el7.noarch.rpm

rpm -ivh hive-jdbc-2.1.1+cdh6.3.2-1605554.el7.noarch.rpm

rpm -ivh hive-metastore-2.1.1+cdh6.3.2-1605554.el7.noarch.rpm

rpm -ivh hive-server2-2.1.1+cdh6.3.2-1605554.el7.noarch.rpm

rpm -ivh hive-2.1.1+cdh6.3.2-1605554.el7.noarch.rpm

rpm -ivh hive-hcatalog-2.1.1+cdh6.3.2-1605554.el7.noarch.rpm

#### Hive配置

1. 配置hive-env.sh

vi /usr/lib/hive/conf/hive-env.sh

export JAVA\_HOME=/usr/jdk1.8.0\_241

export HADOOP\_HOME=/usr/lib/hadoop

export HIVE\_HOME=/usr/lib/hive

export HIVE\_CONF\_DIR=$HIVE\_HOME/conf

export CLASSPATH=$CLASSPATH:$JAVA\_HOME/lib:$HADOOP\_HOME/lib:$HIVE\_HOME/lib

1. 配置hive-site.xml

vi /usr/lib/hive/conf/hive-site.xml –复制<configuration>标签里面内容

<configuration>

<property>

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

<value>jdbc:mysql://147.12.213.132:3306/hive?createDatabaseIfNotExist=true&amp;useSSL=false</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>root</value>

<description>username to use against metastore database</description>

</property>

<property>

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

<value>123456</value>

<description>password to use against metastore database</description>

</property>

<property>

<name>hive.fetch.task.conversion</name>

<value>more</value>

</property>

<property>

<name>hive.server2.thrift.client.user</name>

<value>root</value>

</property>

<property>

<name>hive.server2.thrift.client.password</name>

<value>root1234</value>

</property>

</configuration>

#### 添加jar包

#将mysql的jar包放入hive目录

cp mysql-connector-java-5.1.27.jar /usr/lib/hive/lib/

#将java的tools.jar放入hive目录

cp $JAVA\_HOME/lib/tools.jar ${HIVE\_HOME}/lib

#### 初始化hive元数据库

cd /usr/lib/hive/bin

./schematool -dbType mysql -initSchema

#输出schemaTool completed结果即为成功

#解决hive库comment乱码问题，登陆mysql的hive库执行以下语句

alter table COLUMNS\_V2 modify column COMMENT varchar(256) character set utf8;

alter table TABLE\_PARAMS modify column PARAM\_VALUE varchar(4000) character set utf8;

alter table PARTITION\_PARAMS modify column PARAM\_VALUE varchar(4000) character set utf8 ;

**altertable** PARTITION\_KEYS **modifycolumn** PKEY\_COMMENT **varchar**(4000) **characterset** utf8;

#### 启动hive服务

cd /usr/lib/hive/bin

nohup ./hive --service metastore &

nohup ./hive --service hiveserver2&

### 安装kudu

#### 安装依赖

yum -y install cyrus-sasl-plain lsb

#### 安装kudu包

#三个节点服务器执行

cd /hadoop/cdh6.3.2/RPMS/x86\_64

rpm -ivh kudu-1.10.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh kudu-master-1.10.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh kudu-tserver-1.10.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh kudu-client0-1.10.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh kudu-client-devel-1.10.0+cdh6.3.2-1605554.el7.x86\_64.rpm

#### 配置kudu（配置单节点master）

1. 修改kudu-master

#修改/etc/default/kudu-master（三个节点分别修改各自主机名）

vi /etc/default/kudu-master

export FLAGS\_rpc\_bind\_addresses=hadoop01:7051

1. 修改kudu-tserver

#修改/etc/default/kudu-tserver（三个节点分别修改各自主机名）

vi /etc/default/kudu-tserver

export FLAGS\_rpc\_bind\_addresses=hadoop01:7050

1. 修改master.gflagfile

#修改/etc/kudu/conf/master.gflagfile（hadoop01节点操作）

vi /etc/kudu/conf/master.gflagfile –修改以下项目

--fs\_wal\_dir=/hadoop/kudu/master

--fs\_data\_dirs=/hadoop/kudu/master

--master\_addresses=hadoop01:7051

--default\_num\_replicas=1

--max\_clock\_sync\_error\_usec=20000000

1. 修改tserver.gflagfile

#修改/etc/kudu/conf/tserver.gflagfile（hadoop01节点操作）

vi /etc/kudu/conf/tserver.gflagfile –修改以下项目

--fs\_wal\_dir=/hadoop/kudu/tserver

--fs\_data\_dirs=/hadoop/kudu/tserver

--maintenance\_manager\_num\_threads=4

--max\_clock\_sync\_error\_usec=20000000

--block\_cache\_capacity\_mb=2048

--memory\_limit\_hard\_bytes=16106127360

--tserver\_master\_addrs=hadoop01:7051

1. 拷贝配置文件

#将第一个节点/etc/kudu/conf下的两个配置文件拷贝到其他两台机器

scp /etc/kudu/conf/\*hadoop02: /etc/kudu/conf

scp /etc/kudu/conf/\*hadoop02: /etc/kudu/conf

1. 设置权限

#创建kudu的master和tserver目录（三个节点执行）

mkdir /hadoop/kudu && chown kudu:kudu /hadoop/kudu

1. 启动kudu

#在第一个节点执行

service kudu-master start

#在三个节点执行

service kudu-tserver start

#### 配置kudu（增加master节点）

1. 停止kudu-master和kudu-tserver

#在1节点执行

service kudu-master stop

#在三个节点执行

service kudu-tserver stop

1. 删除master配置

#在三个节点执行

vi /etc/kudu/conf/master.gflagfile

删除--master\_addresses配置

vi /etc/kudu/conf/tserver.gflagfile

删除--tserver\_master\_addrs配置

1. 格式化数据目录

#在hadoop02和hadoop03节点执行

sudo -u kudu kudu fs format --fs\_wal\_dir=/hadoop/kudu/master --fs\_data\_dirs=/hadoop/kudu/master

1. 找出master的uuid

#在三个节点执行，并记录id号

sudo -u kudu kudu fs dump uuid --fs\_wal\_dir=/hadoop/kudu/master --fs\_data\_dirs=/hadoop/kudu/master 2>/dev/null

#三台机器执行结果如下

184594d866ee4fd283bb81a7eb4ff4e2 hadoop01

daaf0ad5250d4ef2a36a8aab3090b874 hadoop02

dd32c9df81e347c0a7fe3d37a6ebca6d hadoop03

1. 在现有master重写master的Raft配置

#以下为一条命令，在1节点执行

sudo -u kudu kudu local\_replica cmeta rewrite\_raft\_config --fs\_wal\_dir=/hadoop/kudu/master --fs\_data\_dirs=/hadoop/kudu/master 00000000000000000000000000000000 184594d866ee4fd283bb81a7eb4ff4e2:hadoop01:7051 daaf0ad5250d4ef2a36a8aab3090b874:hadoop02:7051 dd32c9df81e347c0a7fe3d37a6ebca6d:hadoop03:7051 --master\_addresses=hadoop01:7051,hadoop02:7051,hadoop03:7051

1. 修改所有master的master\_addresses属性

#所有节点修改

vi /etc/kudu/conf/master.gflagfile --增加下面行

--master\_addresses=hadoop01:7051,hadoop02:7051,hadoop03:7051

1. 启动1节点master

#在1节点执行

service kudu-master start

1. 复制1节点master

#在hadoop02和hadoop03节点执行

sudo -u kudu kudu local\_replica copy\_from\_remote --fs\_wal\_dir=/hadoop/kudu/master --fs\_data\_dirs=/hadoop/kudu/master 00000000000000000000000000000000 hadoop01:7051

1. 启动2、3节点master

#在hadoop02和hadoop03节点执行

service kudu-master start

1. 修改所有节点tserver-master\_addrs

#在三个节点执行

vi /etc/kudu/conf/tserver.gflagfile--增加以下参数

--tserver\_master\_addrs=hadoop01:7051,hadoop02:7051,hadoop03:7051

1. 启动tserver

#在三个节点执行

service kudu-tserver start

1. 配置cluster启动脚本

#配置cluster启动脚本，配置在一节点

cd /usr/lib/kudu/sbin

vi kudu-cluster.sh --粘入以下内容

#!/bin/bash

hosts="hadoop01 hadoop02 hadoop03"

for h in $hosts ;

do

tput setaf 2

echo ======== $h ========

tput setaf 7

ssh $h "source /etc/profile;service kudu-master $1" ;

ssh $h "source /etc/profile;service kudu-tserver $1" ;

done

#修改启动脚本权限

chmod 777 kudu-cluster.sh

#启动和停止kudu集群（仅在1节点执行即可）

./kudu-cluster.sh start

./kudu-cluster.sh stop

1. Kudu访问地址

http://hadoop01:8051

1. 检查kudu集群

kudu cluster ksck hadoop01,hadoop02,hadoop03

### 安装impala(hadoop01安装)

#### 安装依赖包

cd /hadoop/cdh6.3.2/RPMS/noarch

rpm -ivh bigtop-utils-0.7.0+cdh6.3.2-1605554.el7.noarch.rpm

rpm -ivh sentry-2.1.0+cdh6.3.2-1605554.el7.noarch.rpm--nodeps

cd /hadoop/cdh6.3.2/RPMS/x86\_64

rpm -ivh hbase-2.1.0+cdh6.3.2-1605554.el7.x86\_64.rpm

#### 安装impala包

cd /hadoop/cdh6.3.2/RPMS/x86\_64

rpm -ivh impala-3.2.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh impala-catalog-3.2.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh impala-state-store-3.2.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh impala-udf-devel-3.2.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh impala-server-3.2.0+cdh6.3.2-1605554.el7.x86\_64.rpm

rpm -ivh impala-shell-3.2.0+cdh6.3.2-1605554.el7.x86\_64.rpm

#### 检查安装

进入impala的lib下软连接的jar包是否都有效，确保没有闪动的软连接

#### 修改配置

1. 修改bigtop-utils和impala文件

##修改/etc/default/bigtop-utils文件，增加java\_home配置

vi /etc/default/bigtop-utils

export JAVA\_HOME=/usr/jdk1.8.0\_241

##修改/etc/default/impala文件，以下标红的为修改或新增内容

vi /etc/default/impala

IMPALA\_CATALOG\_SERVICE\_HOST=147.12.213.132

IMPALA\_STATE\_STORE\_HOST=147.12.213.132

IMPALA\_STATE\_STORE\_PORT=24000

IMPALA\_BACKEND\_PORT=22000

IMPALA\_LOG\_DIR=/var/log/impala

IMPALA\_CATALOG\_ARGS=" -log\_dir=${IMPALA\_LOG\_DIR} "

IMPALA\_STATE\_STORE\_ARGS=" -log\_dir=${IMPALA\_LOG\_DIR} -state\_store\_port=${IMPALA\_STATE\_STORE\_PORT}"

IMPALA\_SERVER\_ARGS=" \

-log\_dir=${IMPALA\_LOG\_DIR} \

-catalog\_service\_host=${IMPALA\_CATALOG\_SERVICE\_HOST} \

-state\_store\_port=${IMPALA\_STATE\_STORE\_PORT} \

-use\_statestore \

-state\_store\_host=${IMPALA\_STATE\_STORE\_HOST} \

-be\_port=${IMPALA\_BACKEND\_PORT} -mem\_limit=80%\

-kudu\_master\_hosts=hadoop01:7051,hadoop02:7051,hadoop03:7051"

ENABLE\_CORE\_DUMPS=true

# LIBHDFS\_OPTS=-Djava.library.path=/usr/lib/impala/lib

# MYSQL\_CONNECTOR\_JAR=/usr/share/java/mysql-connector-java.jar

# IMPALA\_BIN=/usr/lib/impala/sbin

# IMPALA\_HOME=/usr/lib/impala

# HIVE\_HOME=/usr/lib/hive

# HBASE\_HOME=/usr/lib/hbase

# IMPALA\_CONF\_DIR=/etc/impala/conf

# HADOOP\_CONF\_DIR=/etc/impala/conf

# HIVE\_CONF\_DIR=/etc/impala/conf

# HBASE\_CONF\_DIR=/etc/impala/conf

1. 修改core-site.xml

#将hadoop的core-site.xml移动到impala

cp /etc/hadoop/conf/core-site.xml /etc/impala/conf

#将以下内容增加到core-site.xml里面

vi /etc/impala/conf/core-site.xml

<property>

<name>dfs.client.read.shortcircuit</name>

<value>true</value>

</property>

<property>

<name>dfs.client.read.shortcircuit.skip.checksum</name>

<value>false</value>

</property>

<property>

<name>dfs.datanode.hdfs-blocks-metadata.enabled</name>

<value>true</value>

</property>

<property>

<name>fs.AbstractFileSystem.adl.impl</name>

<value>org.apache.hadoop.fs.adl.Adl</value>

</property>

<property>

<name>fs.adl.impl</name>

<value>org.apache.hadoop.fs.adl.AdlFileSystem</value>

</property>

1. 修改hdfs-site.xml

#将hadoop的hdfs-site.xml移动到impala

cp /etc/hadoop/conf/hdfs-site.xml /etc/impala/conf

#将以下内容增加到hdfs-site.xml里面

vi /etc/impala/conf/hdfs-site.xml

<property>

<name>dfs.datanode.hdfs-blocks-metadata.enabled</name>

<value>true</value>

</property>

<property>

<name>dfs.block.local-path-access.user</name>

<value>hadoop</value>

</property>

<property>

<name>dfs.client.file-block-storage-locations.timeout.millis</name>

<value>60000</value>

</property>

1. 修改hive-site.xml

#将hadoop的hive-site.xml移动到impala

cp /etc/hadoop/conf/hive-site.xml /etc/impala/conf

#### 启动服务

#启动服务

service impala-state-store start

service impala-catalog start

service impala-server start

#停止服务

service impala-server stop

service impala-catalog stop

service impala-state-store stop

### 安装hue

#### 安装依赖

yum install -y ant asciidoc cyrus-sasl-devel cyrus-sasl-gssapi gcc gcc-c++ krb5-devel libtidy libxm2-devel make mvn openldap-devel python-devel sqlite-devel openssl-devel gmp-devel libxml2-devel libxslt-devel

#### 安装nodejs

下载地址：<https://nodejs.ora/dist>

#安装nodejs

cd /Hadoop

tar –zxvf node-v13.9.0-linux-x64.tar.gz

mv node-v13.9.0-linux-x64 node

#配置环境便令

vi/etc/profile

export NODE\_HOME=/hadoop/node

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

#使环境变量生效

source /etc/profile

#查看版本

node –version

#### 下载

由于cdh6.3.2自带hue编译需要外网，故使用5.16.2的hue进行安装

下载地址<https://archive.cloudera.com/cdh5/cdh/5/hue-3.9.0-cdh5.16.2.tar.gz>

#### 编译安装

#编译安装

cd /hadoop/hue-3.9.0-cdh5.16.2

make apps

#### 修改配置文件

#修改hue.ini

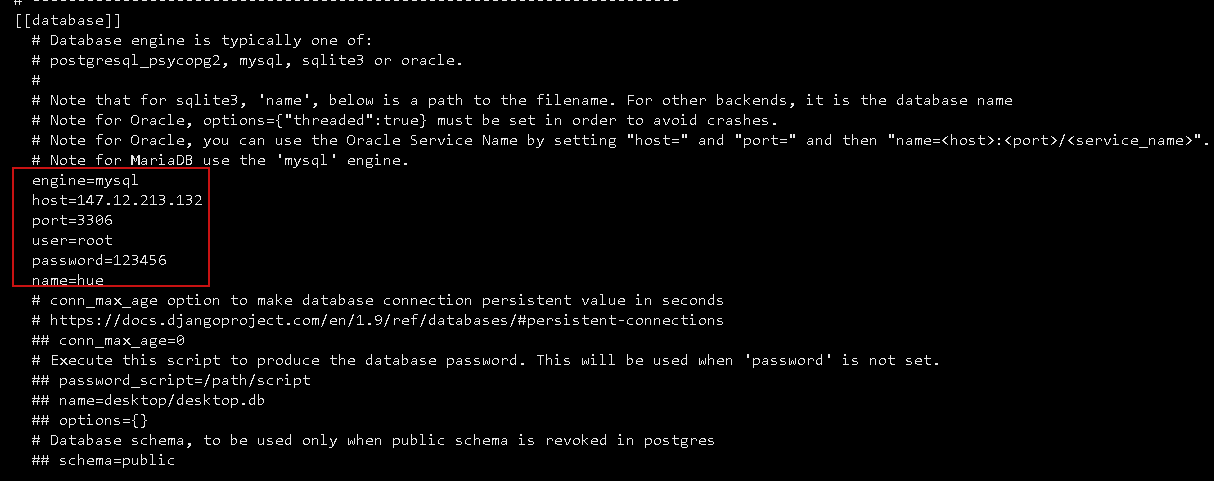
vi /hadoop/hue-3.9.0-cdh5.16.2/desktop/conf/hue.ini

#修改hue的ip和端口

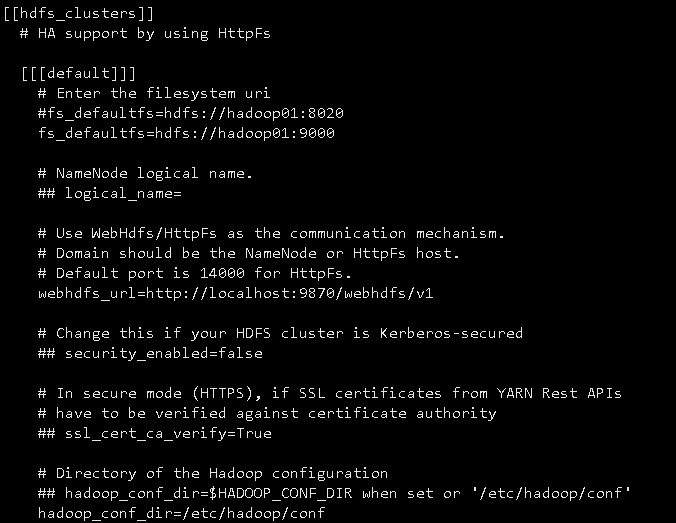
http\_host=147.12.213.132

http\_port=8888

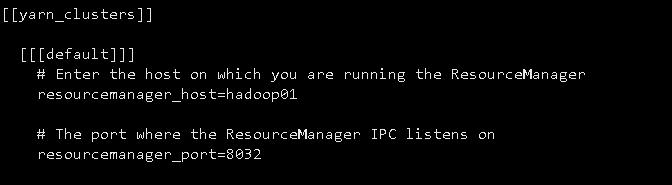
#修改hue的元数据库为mysql库



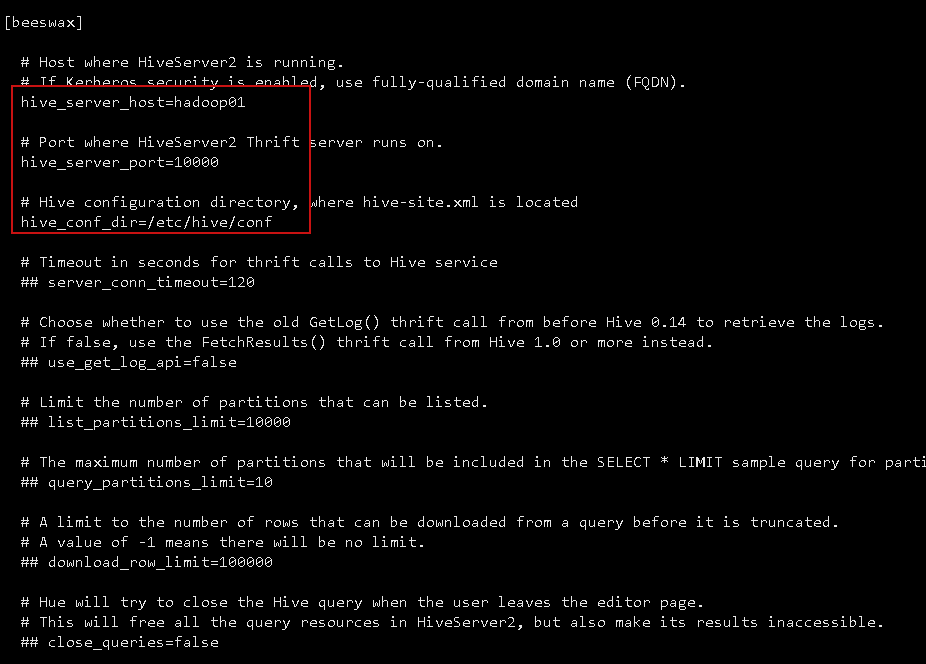
#修改hue连接hdfs



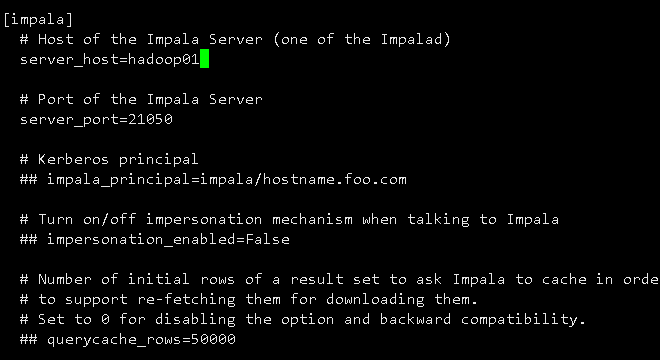
#修改hue连接yarn



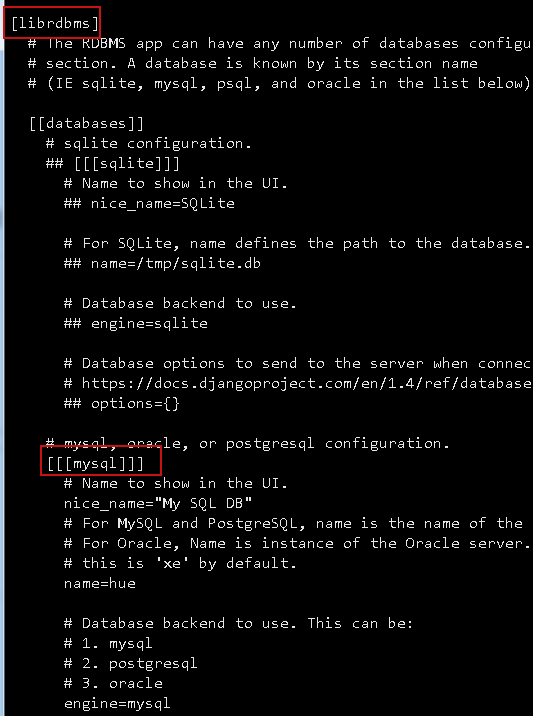
#修改hue连接hive



#修改hue连接impala



#修改hue连接mysql



#### 初始化数据库

#在mysql建立hue数据库

mysql –uroot –p123456

create database hue;

#初始化数据库

cd /hadoop/hue-3.9.0-cdh5.16.2/build/env/bin

./hue syncdb

#数据库生成相应的表

cd /hadoop/hue-3.9.0-cdh5.16.2/build/env/bin

./hue migrate

#### 创建用户

useraddhue

passwdhue

cd /hadoop

chown -R hue:hue hue-3.9.0-cdh5.16.2

#### 启动服务

#启动服务

cd /hadoop/hue-3.9.0-cdh5.16.2/build/env/bin

nohup ./supervisor &

#访问页面，用户名密码为mysql root用户的用户密码

http://hadoop01:8888