Hdoop command:

$ hdfs dfs –copyFromLocal input.txt /WordCountFiles

$ hadoop jar ./WordCount.jar WordCount /WordCountFiles/input.txt /WordCountFiles/output

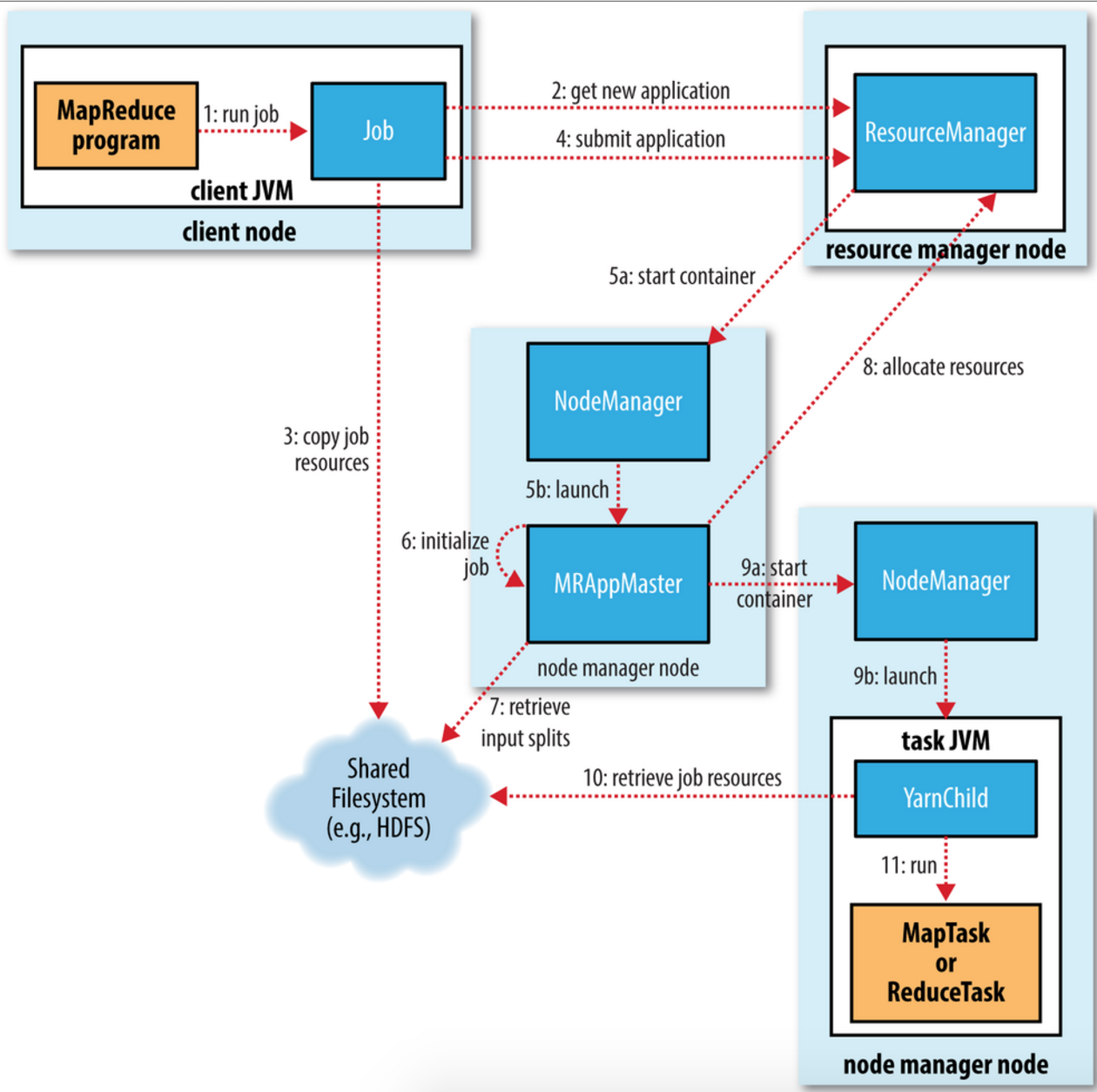
Application Scene:

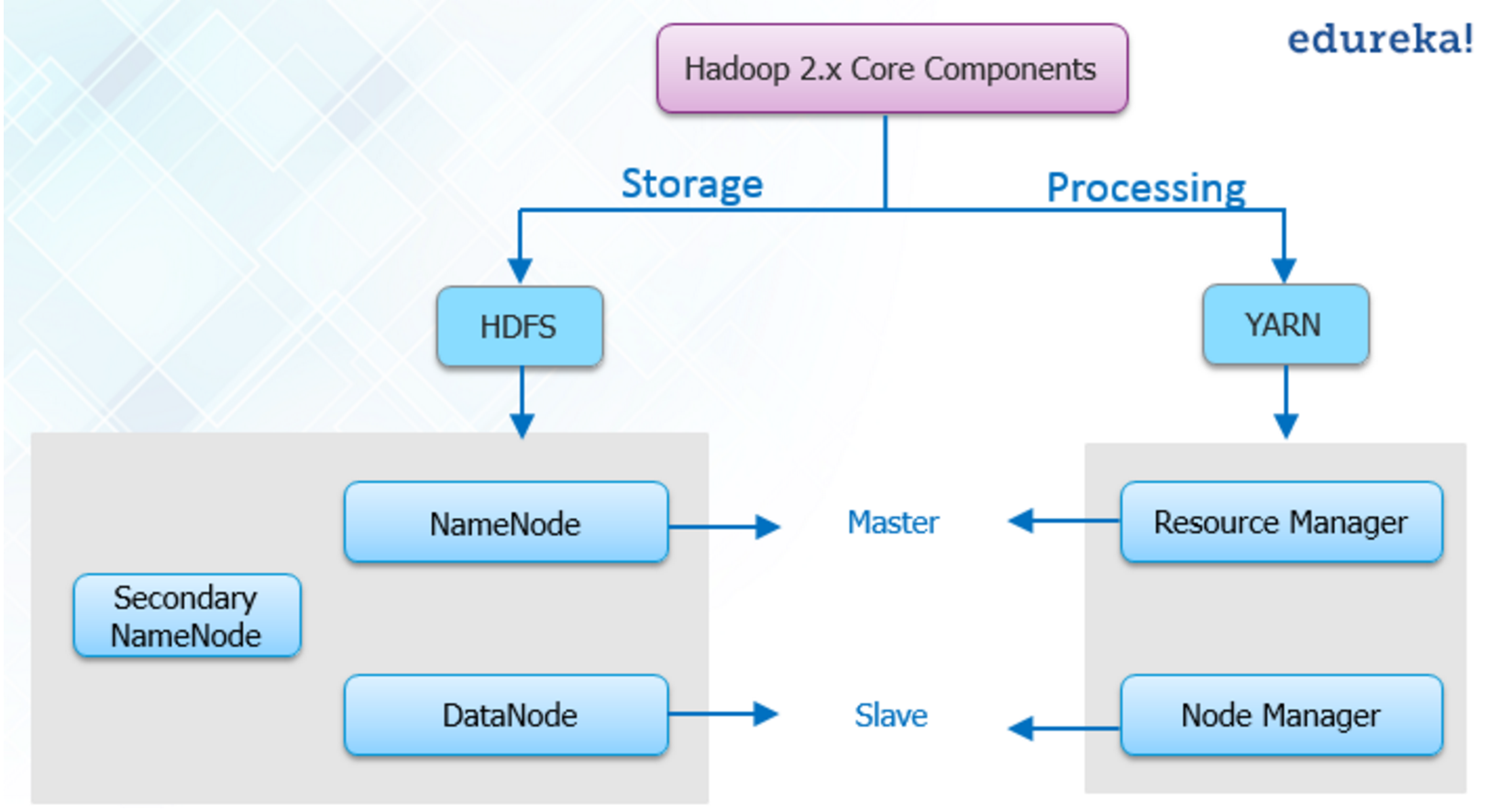
* OTA
* Mobile Data
* E-Commerce
* Energy Mining
* Save Energy
* Base Framework Regulation
* Image Processing
* Cheating inspection
* Medical

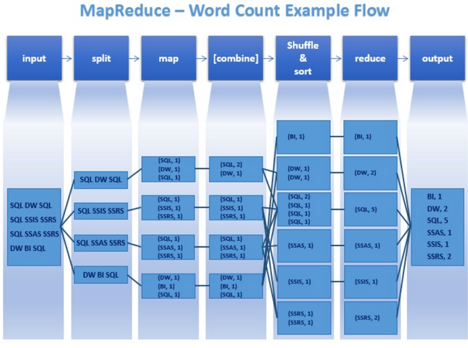
Why Hadoop

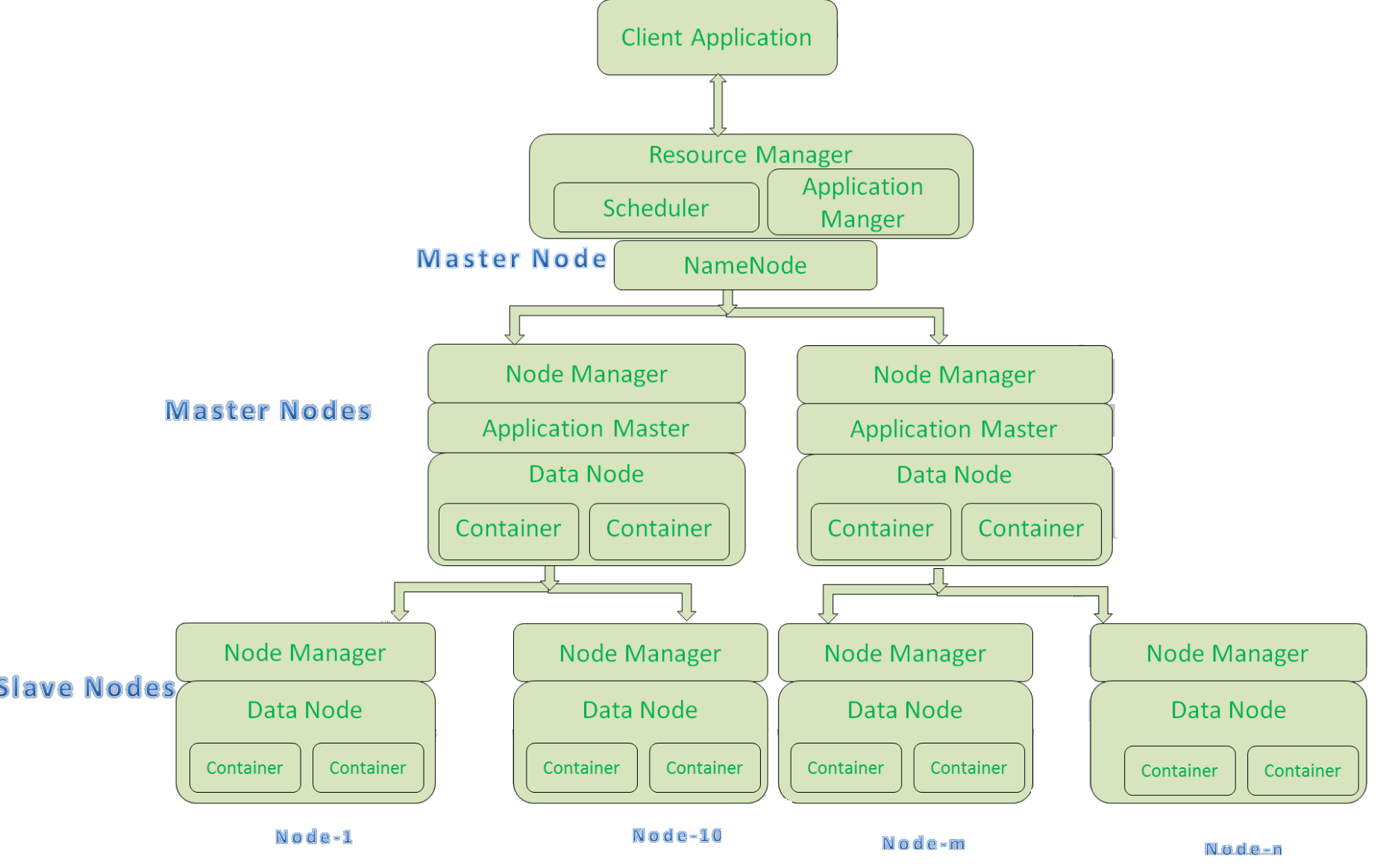
* Scalability
* Efficiency
* Flexibility

How Hadoop runs a MapReduce job









Problem Solving

* DataNode cannot run:

Plan 1:

1. Stop dfs
2. Open hdfs-site.xml
3. Remove the data.dir and name.dir properties from hdfs-site.xml and -format namenode again.
4. Then remove the hadoopdata directory and add the data.dir and name.dir in hdfs-site.xml and again format namenode.
5. Then start dfs again.

**Plan 2:**

* 1. Delete dfs/data
* DataNode cannot show in webpage:

1. Check localhost

SSH不通

Solve1：ssh-add

Hadoop Cluster Installation

Master: Ubuntu14.04, 4G, 40G

Slave01: Ubuntu14.04, 4G, 40G

Slave02: Ubuntu14.04, 4G, 40G

Slave03: Ubuntu14.04, 4G, 40G

1. without password login node

base on master node:

ssh-keygen –t rsa –p

base on slave node:

mkdir ~/.ssh

copy to all slaves:

cat ~/id\_rsa.pub ~/.ssh/authorized\_keys

scp /home/hadoop/.ssh/ authorized\_keys hadoop@slave01.node:~/

1. install java environment

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

download linux x64 jdk-8u91-linux-x64.tar.gz and export

modify environment variable

sudo mkdir /usr/java/jdk1.8

sudo mv jdk1.8.0\_91/ /usr/java/jdk1.8

sudo nano /etc/profile

export JAVA\_HOME=/usr/java/jdk1.8

export CLASSPATH=.:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib:$CLASSPATH

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

export HADOOP\_HOME=/usr/hadoop-2.6.4

source /etc/profile

java –version

1. install Hadoop

[**http://mirror.csclub.uwaterloo.ca/apache/hadoop/common/hadoop-2.6.4/hadoop-2.6.4.tar.gz**](http://mirror.csclub.uwaterloo.ca/apache/hadoop/common/hadoop-2.6.4/hadoop-2.6.4.tar.gz)

change dirctoory

sudo mv Hadoop-2.6.4 /usr

modify /etc/hostname file

sudo nano /etc/hostname

enter password, and eg.change username to slave02

sudo nano /etc/hosts

enter all members ip + name

modify 7 files

hadoop-env.sh

yarn-env.sh

slaves

core-site.xml

hdfs-site.xml

mapred-site.xml

yarn-site.xml

core-site.xml

   <property>  
        <name>fs.defaultFS</name>  
        <value>hdfs://master:9000</value>  
    </property>  
    <property>  
        <name>io.file.buffer.size</name>  
        <value>131072</value>  
    </property>  
    <property>  
        <name>hadoop.tmp.dir</name>  
        <value>file:/usr/ hadoop-2.6.4/tmp</value>  
        <description>Abase for other temporary directories.</description>  
    </property>  
    <property>  
        <name>hadoop.proxyuser.hduser.hosts</name>  
        <value>\*</value>  
    </property>  
    <property>  
        <name>hadoop.proxyuser.hduser.groups</name>  
        <value>\*</value>  
    </property>  
  
</configuration>  
  
 hdfs-site.xml

<configuration>  
    <property>  
        <name>dfs.namenode.secondary.http-address</name>  
        <value>master:9001</value>  
    </property>  
    <property>  
        <name>dfs.namenode.name.dir</name>  
        <value>file:/usr/hadoop-2.6.4/dfs/name</value>  
    </property>  
    <property>  
        <name>dfs.datanode.data.dir</name>  
        <value>file:/usr/ hadoop-2.6.4/dfs/data</value>  
    </property>  
    <property>  
        <name>dfs.replication</name>  
        <value>3</value>  
    </property>  
    <property>  
        <name>dfs.webhdfs.enabled</name>  
        <value>true</value>  
    </property>  
  
</configuration>  
  
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>  
  
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>master:8032</value>  
    </property>  
    <property>  
        <name>yarn.resourcemanager.scheduler.address</name>  
        <value>master:8030</value>  
    </property>  
    <property>  
        <name>yarn.resourcemanager.resource-tracker.address</name>  
        <value>master:8031</value>  
    </property>  
    <property>  
        <name>yarn.resourcemanager.admin.address</name>  
        <value>master:8033</value>  
    </property>  
    <property>  
        <name>yarn.resourcemanager.webapp.address</name>  
        <value>master:8088</value>  
    </property>  
  
</configuration>

1. start service

initial namenode(only need once):

hadoop namenode –format

Start Hadoop:

./start-all.sh

jps

stop-all.sh

IO

写入HDFS一个文件

package hadoop.panda;

import java.io.BufferedInputStream;

import java.io.FileInputStream;

import java.io.InputStream;

import java.io.OutputStream;

import java.net.URI;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IOUtils;

import org.apache.hadoop.util.Progressable;

/\*\*

\*

\* Description: 这个类展示如何将一个文件从本地文件系统复制到 HDFS

\*

\* @author charles.wang

\* @created May 26, 2012 1:00:39 PM

\*

\*/

public class Write {

/\*\*

\* @param args

\*/

public static void main(String[] args)throws Exception{

// TODO Auto-generated method stub

//两个参数分别是本地文件系统的的输入文件路径和HDFS中的输出文件位置

//如果这段代码最终运行在Hadoop所在的服务器上，那么本地文件系统是相对于那台服务器的本地文件系统

//如果这段代码运行在我们Windows PC上，那么本地文件系统是这台Window PC的文件系统

String localSrc = "/Users/panda/Desktop/1901";

String dst= "hdfs://137.207.163.146:9000/usr/input/1.txt";

//因为本地文件系统是基于java.io包的，所以我们创建一个本地文件输入流

InputStream in = new BufferedInputStream( new FileInputStream(localSrc));

//读取hadoop文件系统的配置

Configuration conf = new Configuration();

conf.set("hadoop.job.ugi", "hadoop,hadoop");

//仍然用FileSystem和HDFS打交道

//获得一个对应HDFS目标文件的文件系统

FileSystem fs = FileSystem.get(URI.create(dst), conf);

//创建一个指向HDFS目标文件的输出流

OutputStream out = fs.create(new Path(dst) );

//用IOUtils工具将文件从本地文件系统复制到HDFS目标文件中

IOUtils.copyBytes(in, out, 4096,true);

System.out.println("复制完成");

}

}

从HDFS读文件 在console显示

package hadoop.panda;

import java.net.URI;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FSDataInputStream;

import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IOUtils;

/\*\*

\*

\* Description: 查看Hadoop文件系统中的文件，利用hadoop FileSystem接口中的FSDataInputStream

\* FSDataInputStream还具有流定位的能力，可以从文件的任意位置开始读取

\*

\* @author charles.wang

\* @created May 26, 2012 12:28:49 PM

\*

\*/

public class Read {

public static void main(String[] args) throws Exception{

// TODO Auto-generated method stub

//第一个参数传递进来的是hadoop文件系统中的某个文件的URI,以hdfs://ip 的theme开头

String uri = "hdfs://137.207.163.146:9000/usr/input/1.txt";

//读取hadoop文件系统的配置

Configuration conf = new Configuration();

conf.set("hadoop.job.ugi", "hadoop,hadoop");

//FileSystem是用户操作HDFS的核心类，它获得URI对应的HDFS文件系统

FileSystem fs = FileSystem.get(URI.create(uri),conf);

FSDataInputStream in = null;

try{

//实验一：输出全部文件内容

System.out.println("实验一：输出全部文件内容");

//让FileSystem打开一个uri对应的FSDataInputStream文件输入流，读取这个文件

in = fs.open( new Path(uri) );

//用Hadoop的IOUtils工具方法来让这个文件的指定字节复制到标准输出流上

IOUtils.copyBytes(in, System.out,50,false);

System.out.println();

//实验二:展示FSDataInputStream文件输入流的流定位能力,用seek进行定位

System.out.println("实验二:展示FSDataInputStream文件输入流的流定位能力,用seek进行定位");

//假如我们要吧文件输出3次

//第一次输入全部内容，第二次输入从第20个字符开始的内容，第3次输出从第40个字符开始的内容

for (int i=1;i<=3;i++){

in.seek(0+20\*(i-1));

System.out.println("流定位第 "+i+" 次：" );

IOUtils.copyBytes(in, System.out,4096,false);

}

}finally{

IOUtils.closeStream(in);

}

}

}

从HDFS读文件 写到一个本地文件中

package hadoop.panda;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.FileSystem;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IOUtils;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.net.URI;

/\*\*

\* 3 2的反向操作，在HDFS中生成一个大约100字节的文本文件，写一段程序（可以利用Java API或C API），

\* 读入这个文件，并将其第101-120字节的内容写入本地文件系统成为一个新文件，提供代码和执行结果演示抓图

\*/

public class App {

public static void main(String[] args) throws Exception {

//第一个参数传递进来的是hadoop文件系统中的某个文件的URI,以hdfs://ip 的theme开头，你要读的文件路径

String uri = "hdfs://137.207.163.146:9000/usr/input/1.txt";

Configuration conf = new Configuration();

FileSystem fs = FileSystem.get(URI.create(uri), conf);

InputStream in = null;

FileOutputStream fos = new FileOutputStream(new File("/Users/panda/Desktop/poem\_download.txt"));

try {

in = fs.open(new Path(uri));

in.skip(100);

IOUtils.copyBytes(in, fos, 4096, false);

} finally {

IOUtils.closeStream(in);

IOUtils.closeStream(fos);

}

}

}

安装HIVE-2.0.0

<https://mongodblog.wordpress.com/2016/02/27/apache-hive-2-0-0-installation-on-ubuntu-linux-14-04-lts/>

Download

Copy file to usr/hive-2.0.0

mv apache-hive-0.14.0-bin /usr/hive-2.0.0

set environment variable for hive

sudo nano /etc/profile

export HIVE\_HOME=/usr/hive-2.0.0

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

source /etc/profile

configuring hive

$ cd $HIVE\_HOME/conf

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

Edit the **hive-env.sh** file by appending the following line:

export

HADOOP\_HOME=/usr/hadoop-2.6.4

**Downloading Apache Derby**

cd ~

$ wget http://archive.apache.org/dist/db/derby/db-derby-10.4.2.0/db-derby-10.4.2.0-bin.tar.gz

### Extracting and verifying Derby archive

$ tar zxvf

db-derby-10.4.2.0-bin.tar.gz

$ ls

On successful

download, you get to see the following response:

db-derby-10.4.2.0-bin

db-derby-10.4.2.0-bin.tar.gz

### Copying files to /usr/derby directory

# cd /usr/hadoop-2.6.4

# mv db-derby-10.4.2.0-bin /usr/derby

### Setting up environment for Derby

export

DERBY\_HOME=/usr/derby

export

PATH=$PATH:$DERBY\_HOME/bin

export

CLASSPATH=$CLASSPATH:$DERBY\_HOME/lib/derby.jar:$DERBY\_HOME/lib/derbytools.jar

### Create a directory to store Metastore

mkdir $DERBY\_HOME/data

## **Configuring Metastore of Hive**

cd $HIVE\_HOME/conf

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

Edit **hive-site.xml** and append the following lines between the <configuration> and </configuration> tags:

<property>

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

<value>jdbc:derby://localhost:1527/metastore\_db;create=true

</value>

<description>JDBC connect string for a

JDBC metastore </description>

</property>

Create a file named jpox.properties and add the following lines into it:

javax.jdo.PersistenceManagerFactoryClass =

org.jpox.PersistenceManagerFactoryImpl

org.jpox.autoCreateSchema = false

org.jpox.validateTables = false

org.jpox.validateColumns = false

org.jpox.validateConstraints = false

org.jpox.storeManagerType = rdbms

org.jpox.autoCreateSchema = true

org.jpox.autoStartMechanismMode = checked

org.jpox.transactionIsolation = read\_committed

javax.jdo.option.DetachAllOnCommit = true

javax.jdo.option.NontransactionalRead = true

javax.jdo.option.ConnectionDriverName =

org.apache.derby.jdbc.ClientDriver

javax.jdo.option.ConnectionURL =

jdbc:derby://hadoop1:1527/metastore\_db;create = true

javax.jdo.option.ConnectionUserName = APP

javax.jdo.option.ConnectionPassword = mine

## **Verifying Hive Installation**

Before running Hive, you need to create the **/tmp** folder and a separate Hive folder in HDFS. Here, we use the**/user/hive/warehouse** folder. You need to set write permission for these newly created folders as shown below:

chmod g+w

Now set them in HDFS before verifying Hive. Use the following commands:

$ $HADOOP\_HOME/bin/hadoop-2.4.6 fs -mkdir /tmp

$ $HADOOP\_HOME/bin/hadoop-2.4.6 fs -mkdir

/usr/hive/warehouse

$ $HADOOP\_HOME/bin/hadoop-2.4.6 fs -chmod g+w /tmp

$

$HADOOP\_HOME/bin/hadoop fs -chmod g+w /usr/hive/warehouse

$HADOOP\_HOME/bin/hadoop fs -chmod g+w /usr/hive/tmp

$HADOOP\_HOME/bin/hadoop fs -chmod g+w /usr/hive/log

The following commands are used to verify Hive installation:

$ cd $HIVE\_HOME

$ bin/hive

schematool -initSchema -dbType derby

hive>

安装HUE

<http://tutorialforlinux.com/2014/05/23/how-to-install-hue-hadoop-web-gui-on-ubuntu-14-04-trusty-lts-easy-guide/>

1. **Installing Pre-Requisite Stuff**

|  |
| --- |
| sudo su -c "apt-get install ant gcc g++ libkrb5-dev libmysqlclient-dev libssl-dev \  libsasl2-dev libsasl2-modules-gssapi-mit libsqlite3-dev libtidy-0.99-0 libxml2-dev \  libxslt-dev maven libldap2-dev python-dev python-setuptools git" |

1. **Installing Latest Cloudera Hue**

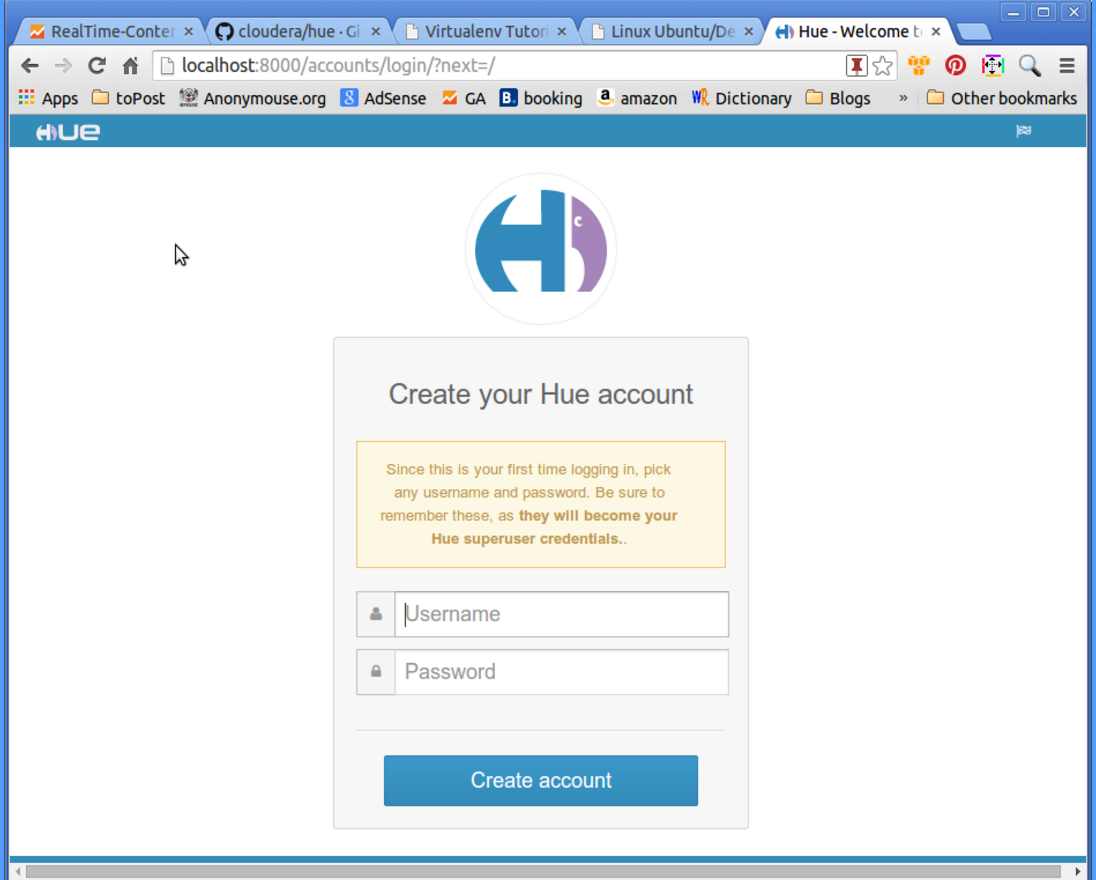
|  |
| --- |
| git clone http://github.com/cloudera/hue.git |
| cd hue |

|  |
| --- |
| make apps |

**3.Running Hue Dev Server**

|  |
| --- |
| build/env/bin/hue runserver |

4.After You can Login on: **http://localhost:8000**



安装hue 会使你的java vision变成1.7, 如果安装过程中出错， 下载一个javas1.7。