**# Procedure for the manual function of creating cluster(code expalnation)**

##############################################################################

**#!/usr/bin/python**

**print "Content-Type: text/html"**

**print**

**import cgi**

**import datanodeman**

**import namenodeman**

**data= cgi.FormContent()**

**namenodeman.nn((data['nn'][0]))**

**datanodeman.dn((data['jt'][0]))**

**datanodeman.dn((data['dn'][0]))**

**datanodeman.dn((data['dn'][1]))**

**print "cluster created"**

**##########################################################**

*1.* "#!/usr/bin/python" shebang is added in order to make the file executable. /usr/bin/python is the path.

**2.**Modules are imported . 'namenodeman' and 'datanodeman' are the manually created modules and cgi is inbuilt module provided by python which enables user to use website related features via python.

**3.**the data sent by the user via the webpage displaying the ip's and their RAM is collected in the variable 'data' using 'FormContent' function of 'cgi' .

**4.** Now the corresponding data sent and collected in the variables 'nn' and 'dn ' ie ip's are now sent to the functions created in the manually created modules namenly ' namenodeman.nn()' and 'datanodeman.dn()'.

**5**.the remaining procedure ie replacing the core-site and hdfs-site files are being taken care by the function itself.Also the function starts the respective services.

**Code for datanodeman module ;-**

###################################################################

**#!/usr/bin/python**

**import os**

**import commands**

**def dn(s):**

**cmd=" hostnamectl set-hostname data"**

**commands.getstatusoutput("sshpass -p redhat ssh -l root {} {}".format(s,cmd))**

**commands.getstatusoutput("sshpass -p redhat scp /root/Desktop/final\_work/datanode/core-site.xml {}:/etc/hadoop/core-site.xml".format(s))**

**commands.getstatusoutput("sshpass -p redhat scp /root/Desktop/final\_work/datanode/hdfs-site.xml {}:/etc/hadoop/hdfs-site.xml".format(s))**

**service=" hadoop-daemon.sh start datanode"**

**commands.getstatusoutput("sshpass -p redhat ssh -l root {} {}".format(s,service))**

###################################################################

**i)** In the manually created modules function is created for both operations to be performed at both data-node and name-node seprately.

-- both the functions accept ip of the node as a string

**ii)** operations include

--'format' feature of python is used to place the strings collected in some variables to a desired location which is specified by "{}". this function supports more than one variables at a time to be placed.

--setting hostname using command 'hostname set-hostname namenode' or 'hostname set-hostname datanode' . this is a linux command . and then by using ssh coomand this command is made to run on the diiferent nodes.

-- 'sshpass -p redhat' this enables us to do the ssh command without manually entering the password .in this case the password for the nodes are 'redhat'.

--'ssh -l root ' this enables us to access the nodes with the permission of root.

-- the whole ssh command is run in python using the 'os' module of python and moreover using the system function, which runs command's of linux via python.

**iii)** replacing the core-site.xml file and the hdfs-site.xml file and starting their respective services using ssh.

--the file to be replaced are created on the main machine and placed on the Desktop (final path == /root/Desktop/final\_work/namemode/core-site.xml and /root/Desktop/final\_work/namemode/hdfs-site.xml) for both namenode and datanode respectively.

-- the procedure is completed by using 'os.system' module of python and 'ssh ' commands as used in the previous cases.

**iv)** the respective daemons ie services are started using the command 'hadoop-daemon.sh start namenode' with the help of 'os ' module and 'ssh ' commands as done in the previous cases.