# 데이터과학
# Hadoop
# 8주차

201202156
오희빈

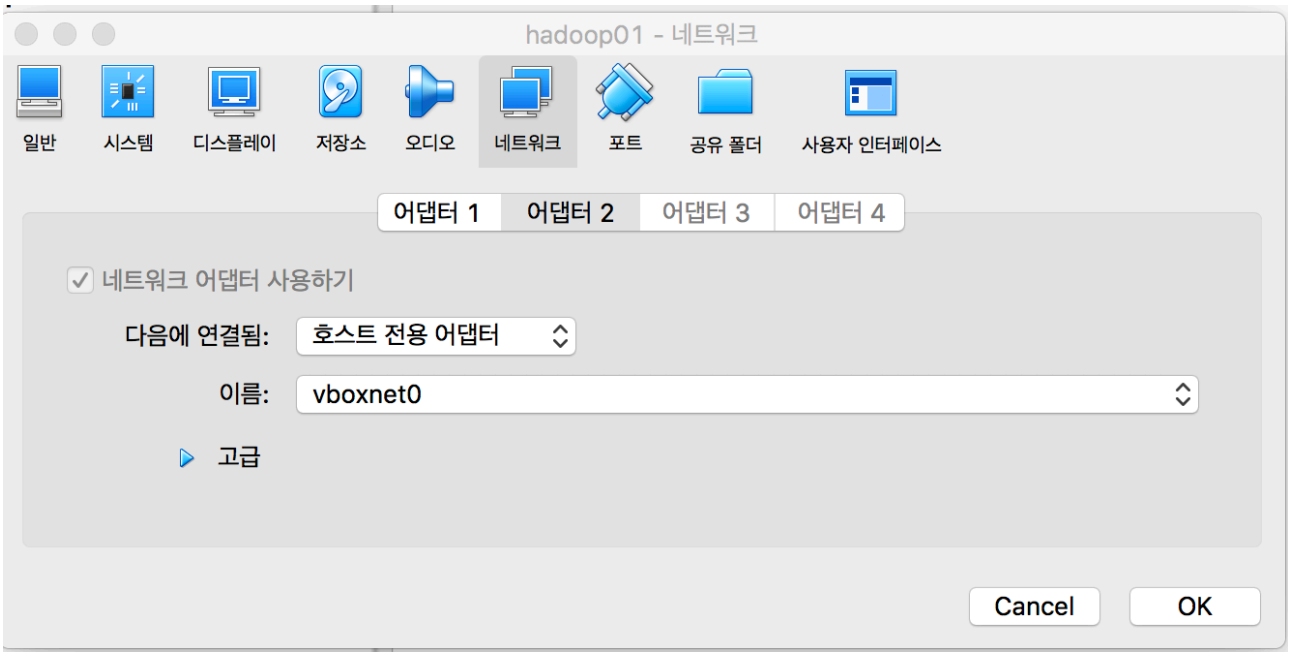
1. Hadoop fully Distriduted 설치 및 setup

```
hadoop01@hadoop01-VirtualBox: ~/hadoop-2.7.3
hadoop01@hadoop01-VirtualBox:~$ cd hadoop-2.7.3/
hadoop01@hadoop01-VirtualBox:~/hadoop-2.7.3$ bin/hadoop
Usage: hadoop [--config confdir] [COMMAND | CLASSNAME]
  CLASSNAME            run the class named CLASSNAME
 or
  where COMMAND is one of:
  fs                   run a generic filesystem user client
  version              print the version
  jar <jar>            run a jar file
                       note: please use "yarn jar" to launch
                             YARN applications, not this command.
  checknative [-a|-h]  check native hadoop and compression libraries availabilit
y
  distcp <srcurl> <desturl> copy file or directories recursively
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archi
ve
  classpath            prints the class path needed to get the
  credential           interact with credential providers
                       Hadoop jar and the required libraries
  daemonlog            get/set the log level for each daemon
  trace                view and modify Hadoop tracing settings

Most commands print help when invoked w/o parameters.
hadoop01@hadoop01-VirtualBox:~/hadoop-2.7.3$
```

=> hadoop-2.7.3 설치 후 실행시 아무 이상 없이 실행이 되는 것을 확인할 수 있다.

```
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME.  All others are
# optional.  When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=/usr/lib/jvm/java-8-oracle

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol.  Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}
                                                          33,1          13%
```

=> hadoop_env.sh 의 JAVA_HOME을 설치한 java8의 위치에 맞게 변경해준다.




=> 복제하기 전 각각 머신들이 비밀번호 확인 없이 로그인이 가능하게 해주기 위해서 host_only로 네트워크를 하나 더 추가 시킨 후에 복제를 해준다.


=> 처음에 hadoop과 java를 설치해주고 네트워크를 host_only로 추가해준 머신 1개를 3개 복제시켜 준다.
namenode(secondarynode) => hadoop01

resourcemanager => hadoop03

datanode, nodemanager => hadoop01, hadoop02, hadoop03, hadoop04

```
127.0.0.1       localhost
127.0.1.1       hadoop01-VirtualBox
192.168.56.101  hadoop00
192.168.56.104  hadoop02
192.168.56.102  hadoop03
192.168.56.103  hadoop04
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
~
~
~
~
~
~
~
~
~
~
~
"hosts" [readonly] 12L, 329C                                  1,1           All
```

=> 복제 시켰던 머신들의 호스트 전용 어뎁터의 네트워크 주소를 확인 시켜준후 각각의 머신들의 etc/hosts에 머신들의 네트워크를 저장시켜준다. 그리고 각 머신들의 hostname을 hadoop00, hadoop02, hadoop03, hadoop04로 바꾼후 저장시킨다.

```
hadoop01@hadoop00:~/hadoop-2.7.3/dfs$ ls
data  name
```

=> datanode와 namenode의 데이터를 저장하기 위해서 hadoop 디렉토리에 dfs 디렉토리를 만들고 난후에 data와 name 디렉토리를 만들어 준다.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
        <property>
                <name>fs.defaultFS</name>
                <value>hdfs://hadoop00</value>
        </property>
</configuration>
~
~
~
"core-site.xml" 24L, 860C                                          1,1           All
```

=> core-site.xml 에 fs.defaultFS를 namenode로 설정한 hadoop00을 hdfs://hadoop00으로 설정해줌으로써 namenode를 지정해준다.

```xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>
</configuration>
~
~
~
"mapred-site.xml" 24L, 844C                                        1,1           All
```

=> mapred-site.xml에 mapreduce.framework.name을 hadoop2에서는 yarn을 사용하기에 value를 yarn으로 지정해준다.

```
  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
        <property>
                <name>dfs.replication</name>
                <value>4</value>
        </property>
        <property>
                <name>dfs.datanode.data.dir</name>
                <value>/home/hadoop01/hadoop-2.7.3/dfs/data</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>/home/hadoop01/hadoop-2.7.3/dfs/name</value>
        </property>
        <property>
                <name>dfs.hosts</name>
                <value>/home/hadoop01/hadoop-2.7.3/include</value>
        </property>
</configuration>
                                                                36,1          Bot
```

=> hdfs-site.xml 안에 dfs.replication은 datanode의 갯수이기에 4를 값으로 준다.
그리고 dfs.datanode.data.dir datanode의 data를 저장해주기위해 전에 만든 data디렉토리의 위치를 값으로 주어주고 dfs.namenode.name.dir은 namenode의 data를 저장해주는 것이기에 전에 만든 name 디렉토리의 위치를 값으로 준다.
dfs.hosts는 hadoop 디렉토리에 존재하는 include 디렉토리의 위치를 값으로 지정해준다.
hdfs-site.xml을 저장시켜준다.

```
hadoop01@hadoop03: ~/hadoop-2.7.3/etc/hadoop
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# User for YARN daemons
export HADOOP_YARN_USER=${HADOOP_YARN_USER:-yarn}

# resolve links - $0 may be a softlink
export YARN_CONF_DIR="${YARN_CONF_DIR:-$HADOOP_YARN_HOME/conf}"

# some Java parameters
export JAVA_HOME=/usr/lib/jvm/java-8-oracle
if [ "$JAVA_HOME" != "" ]; then
  #echo "run java in $JAVA_HOME"
  JAVA_HOME=/usr/lib/jvm/java-8-oracle
fi

if [ "$JAVA_HOME" = "" ]; then
  echo "Error: JAVA_HOME is not set."
                                                                23,43         7%
                                                                Left ⌘
```

=> yarn-env.sh에 존재하는 JAVA_HOME의 환경변수를 java8의 위치에 맞게 저장시켜준다.

```
<?xml version="1.0"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->
<configuration>

<!-- Site specific YARN configuration properties -->
        <property>
                <name>yarn.nodemanager.aux-services</name>
                <value>mapreduce_shuffle</value>
        </property>
        <property>
                <name>yarn.resourcemanager.hostname</name>
                <value>hadoop03</value>
        </property>
</configuration>
~
"yarn-site.xml" 26L, 890C                                          1,1           All
```

=> yarn-site.xml에 yarn.nodemanager.aux-services를 mapreduce_shuffle로 설정해준다.
그리고 yarn.resourcemanager.hostname을 resourcemanager를 관리해줄 hadoop03으로 값을 주고 yarn-site.xml을 저장시킨다.

```
hadoop01@hadoop00: ~/hadoop-2.7.3/etc/hadoop
hadoop00
hadoop02
hadoop03
hadoop04
~
~
~
~
~
```

=> etc/hadoop에 존재하는 slaves 파일에 hadoop00, hadoop02, hadoop03, hadoop04를 모두 입력해준다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ rsync -avz /home/hadoop01/hadoop-2.7.3 hadoop0
1@hadoop02:/home/hadoop01
hadoop01@hadoop00:~/hadoop-2.7.3$ rsync -avz /home/hadoop01/hadoop-2.7.3 hadoop0
1@hadoop03:/home/hadoop01
hadoop01@hadoop00:~/hadoop-2.7.3$ rsync -avz /home/hadoop01/hadoop-2.7.3 hadoop0
1@hadoop04:/home/hadoop01
```

=> hadoop에 대한것을 hadoop00에서 설정을 해준 다음 hadoop 디렉토리를 hadoop02, hadoop03, hadoop04에 rsync를 이용해서 변경된 부분을 전부 copy 시켜준다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs namenode -format
17/05/13 20:02:36 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = hadoop00/192.168.56.101
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.7.3
STARTUP_MSG:   classpath = /home/hadoop01/hadoop-2.7.3/etc/hadoop:/home/hadoop01
/hadoop-2.7.3/share/hadoop/common/lib/zookeeper-3.4.6.jar:/home/hadoop01/hadoop-
2.7.3/share/hadoop/common/lib/jettison-1.1.jar:/home/hadoop01/hadoop-2.7.3/share
/hadoop/common/lib/httpclient-4.2.5.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop
/common/lib/jaxb-api-2.2.2.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/l
ib/java-xmlbuilder-0.4.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/lib/j
sr305-3.0.0.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/lib/junit-4.11.j
ar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/lib/snappy-java-1.0.4.1.jar:/
home/hadoop01/hadoop-2.7.3/share/hadoop/common/lib/xmlenc-0.52.jar:/home/hadoop0
1/hadoop-2.7.3/share/hadoop/common/lib/jets3t-0.9.0.jar:/home/hadoop01/hadoop-2.
7.3/share/hadoop/common/lib/gson-2.2.4.jar:/home/hadoop01/hadoop-2.7.3/share/had
oop/common/lib/protobuf-java-2.5.0.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/
common/lib/commons-cli-1.2.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/l
ib/hamcrest-core-1.3.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/lib/cur
ator-client-2.7.1.jar:/home/hadoop01/hadoop-2.7.3/share/hadoop/common/lib/common
```

=> hadoop을 실행시키기 전에 bin/hdfs namenode -format 을 실행 시켜준다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ sbin/start-dfs.sh
Starting namenodes on [hadoop00]
hadoop00: starting namenode, logging to /home/hadoop01/hadoop-2.7.3/logs/hadoop-
hadoop01-namenode-hadoop03.out
hadoop04: starting datanode, logging to /home/hadoop01/hadoop-2.7.3/logs/hadoop-
hadoop01-datanode-hadoop00.out
hadoop00: starting datanode, logging to /home/hadoop01/hadoop-2.7.3/logs/hadoop-
hadoop01-datanode-hadoop03.out
hadoop03: starting datanode, logging to /home/hadoop01/hadoop-2.7.3/logs/hadoop-
hadoop01-datanode-hadoop04.out
hadoop02: datanode running as process 7857. Stop it first.
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/hadoop01/hadoop-2.7.3/logs
/hadoop-hadoop01-secondarynamenode-hadoop00.out
```

=> sbin/start-dfs.sh 를 실행 시켜서 namenode와 secondary namenode 그리고 datanode를 실행 시켜준다.

```
hadoop01@hadoop03:~/hadoop-2.7.3$ sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/hadoop01/hadoop-2.7.3/logs/yarn-hadoo
p01-resourcemanager-hadoop03.out
hadoop04: nodemanager running as process 4732. Stop it first.
hadoop00: starting nodemanager, logging to /home/hadoop01/hadoop-2.7.3/logs/yarn
-hadoop01-nodemanager-hadoop00.out
hadoop02: nodemanager running as process 6767. Stop it first.
hadoop03: nodemanager running as process 5988. Stop it first.
```

=> sbin/start-yarn.sh를 실행 시켜 resourcemanager와 nodemanager를 실행시켜준다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ jps
4130 NodeManager
4308 Jps
3750 DataNode
3944 SecondaryNameNode
3582 NameNode
```

=> hadoop00의 jps

```
hadoop01@hadoop02:~$ jps
7508 Jps
7318 DataNode
6767 NodeManager
```

=> hadoop02의 jps

```
hadoop01@hadoop03:~/hadoop-2.7.3$ jps
7378 Jps
5988 NodeManager
6935 DataNode
7065 ResourceManager
```

=> hadoop03의 jps

```
hadoop01@hadoop04:~$ jps
5289 DataNode
4732 NodeManager
5485 Jps
```

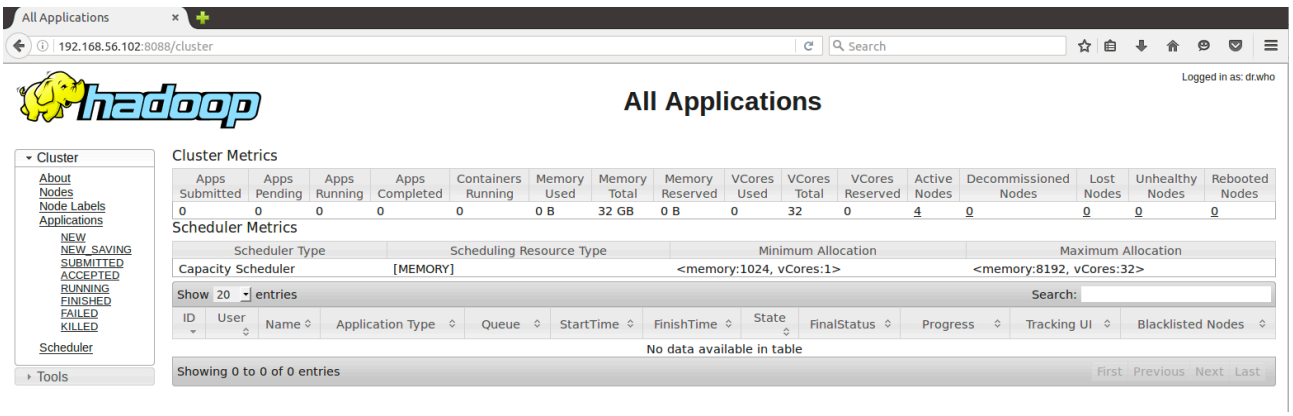
=> hadoop04의 jps

```
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfsadmin -report
Configured Capacity: 81369726976 (75.78 GB)
Present Capacity: 55781777408 (51.95 GB)
DFS Remaining: 55781679104 (51.95 GB)
DFS Used: 98304 (96 KB)
DFS Used%: 0.00%
Under replicated blocks: 0
Blocks with corrupt replicas: 0
Missing blocks: 0
Missing blocks (with replication factor 1): 0

-------------------------------------------------
Live datanodes (4):

Name: 192.168.56.101:50010 (hadoop00)
Hostname: hadoop00
Decommission Status : Normal
Configured Capacity: 20342431744 (18.95 GB)
DFS Used: 24576 (24 KB)
Non DFS Used: 6353633280 (5.92 GB)
DFS Remaining: 13988773888 (13.03 GB)
DFS Used%: 0.00%
DFS Remaining%: 68.77%
Configured Cache Capacity: 0 (0 B)
```

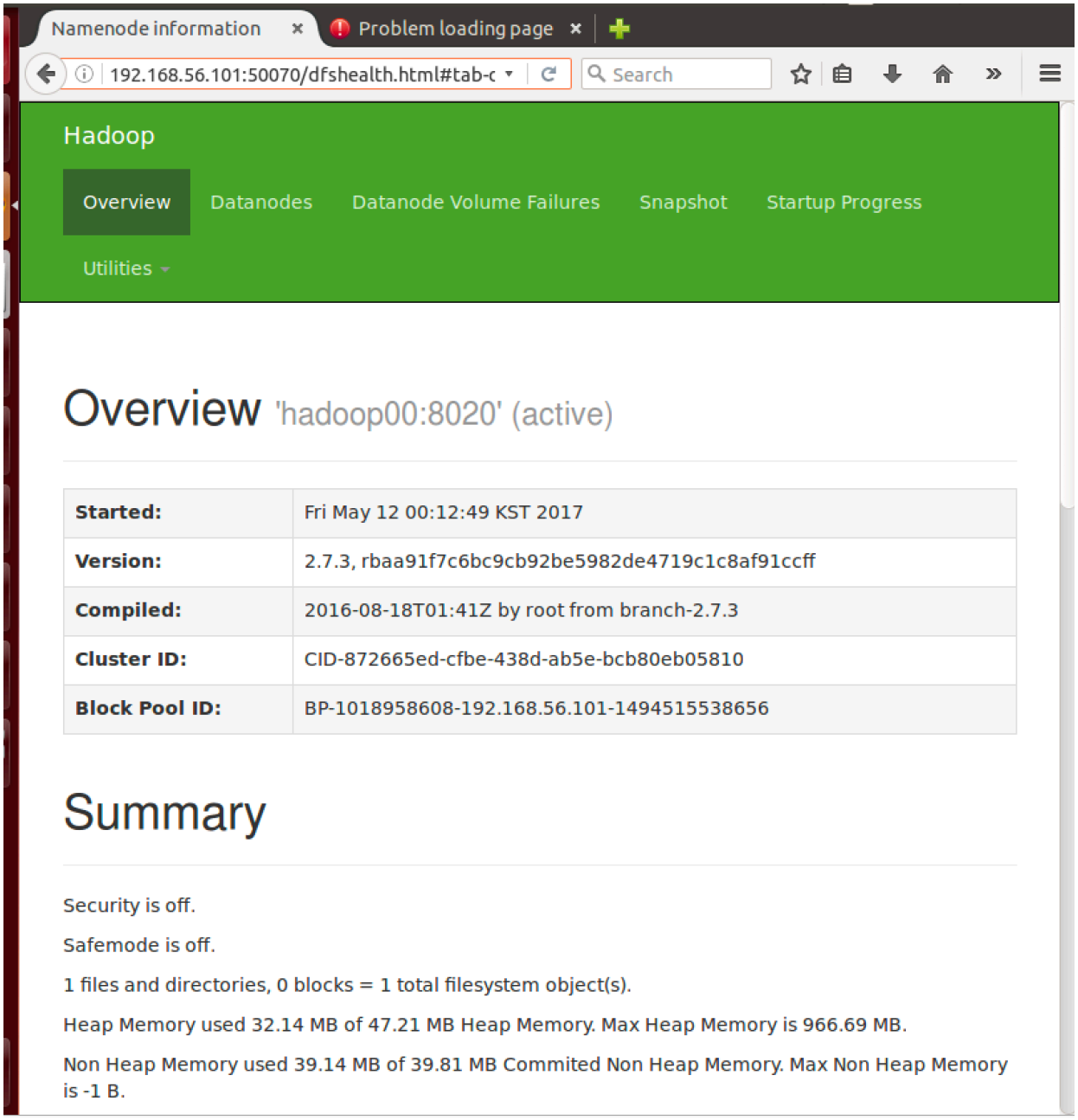
=> bin/hdfs dfsadmin -report 실행 화면

```
hadoop01@hadoop03:~/hadoop-2.7.3$ bin/yarn node -list
17/05/12 01:32:11 INFO client.RMProxy: Connecting to ResourceManager at hadoop03
/192.168.56.102:8032
Total Nodes:4
         Node-Id             Node-State Node-Http-Address       Number-of-Runnin
g-Containers
  hadoop03:38901                RUNNING     hadoop03:8042
           0
  hadoop02:44777                RUNNING     hadoop02:8042
           0
  hadoop04:42355                RUNNING     hadoop04:8042
           0
  hadoop00:36102                RUNNING     hadoop00:8042
           0
```

=> bin/yarn node -list 실행화면




=> resourcemanager인 192.168.56.102(hadoop03):8088 실행 화면




=> namenode인 192.168.56.101(hadoop00):50070 실행화면

2. wordcount 예제 (Fully-Distribution 상태에서 진행)

```
hadoop01@hadoop00:~/hadoop-2.7.3$ echo "Hello world in HDFS" > /home/hadoop01/testfile1
hadoop01@hadoop00:~/hadoop-2.7.3$ echo "Hadoop word count example in HDFS" > /home/hadoop
le2
```

=> wordCount를 하기위해서 file에 Hello world in HDFS와 Hadoop word count example in HDFS를 써서 testfile1, testfile2에 저장시킨다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -mkdir /user
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -mkdir /user/hadoop01
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -mkdir /user/hadoop01/input
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -put /home/hadoop01/testfile1 /user/hadoop
01/input
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -put /home/hadoop01/testfile2 /user/hadoop
01/input
```

=> bin/hdfs dfs 를 이용해서 /user/hadoop01/input 디렉토리를 만든 후 그안에 testfile1과 testfile2를 put해준다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-
examples-2.7.3.jar wordcount /user/hadoop01/input /user/hadoop01/output
17/05/12 02:21:11 INFO client.RMProxy: Connecting to ResourceManager at hadoop03/192.168.
56.102:8032
17/05/12 02:21:13 INFO input.FileInputFormat: Total input paths to process : 2
17/05/12 02:21:13 INFO mapreduce.JobSubmitter: number of splits:2
17/05/12 02:21:13 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_14945200673
26_0008
17/05/12 02:21:14 INFO impl.YarnClientImpl: Submitted application application_14945200673
26_0008
17/05/12 02:21:14 INFO mapreduce.Job: The url to track the job: http://hadoop03:8088/prox
y/application_1494520067326_0008/
17/05/12 02:21:14 INFO mapreduce.Job: Running job: job_1494520067326_0008
17/05/12 02:21:27 INFO mapreduce.Job: Job job_1494520067326_0008 running in uber mode : f
alse
17/05/12 02:21:27 INFO mapreduce.Job:  map 0% reduce 0%
17/05/12 02:21:41 INFO mapreduce.Job:  map 100% reduce 0%
17/05/12 02:21:51 INFO mapreduce.Job:  map 100% reduce 100%
17/05/12 02:21:52 INFO mapreduce.Job: Job job_1494520067326_0008 completed successfully
17/05/12 02:21:52 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=120
                FILE: Number of bytes written=356654
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=274
                HDFS: Number of bytes written=62
```

=> share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar 파일을 이용해서
bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount /user/hadoop01/input /user/hadoop01/output
위에 저장한 testfile에 대해서 wordcount를 실시해준후 output 디렉토리에 그 결과를 저장시켜준다.

```
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -ls /user/hadoop01/output
Found 2 items
-rw-r--r--   4 hadoop01 supergroup          0 2017-05-12 02:21 /user/hadoop01/output/_SUC
CESS
-rw-r--r--   4 hadoop01 supergroup         62 2017-05-12 02:21 /user/hadoop01/output/part
-r-00000
hadoop01@hadoop00:~/hadoop-2.7.3$ bin/hdfs dfs -cat /user/hadoop01/output/part-r-00000
HDFS    2
Hadoop  1
Hello   1
count   1
example 1
in      2
word    1
world   1
hadoop01@hadoop00:~/hadoop-2.7.3$
```

=> 저장시켜준 output파일에 존재하는 part-r-00000 파일을 확인해 보면 위에서 만들어준 2개의 testfile을 wordCount 한 결과를 확인 할 수 있다.