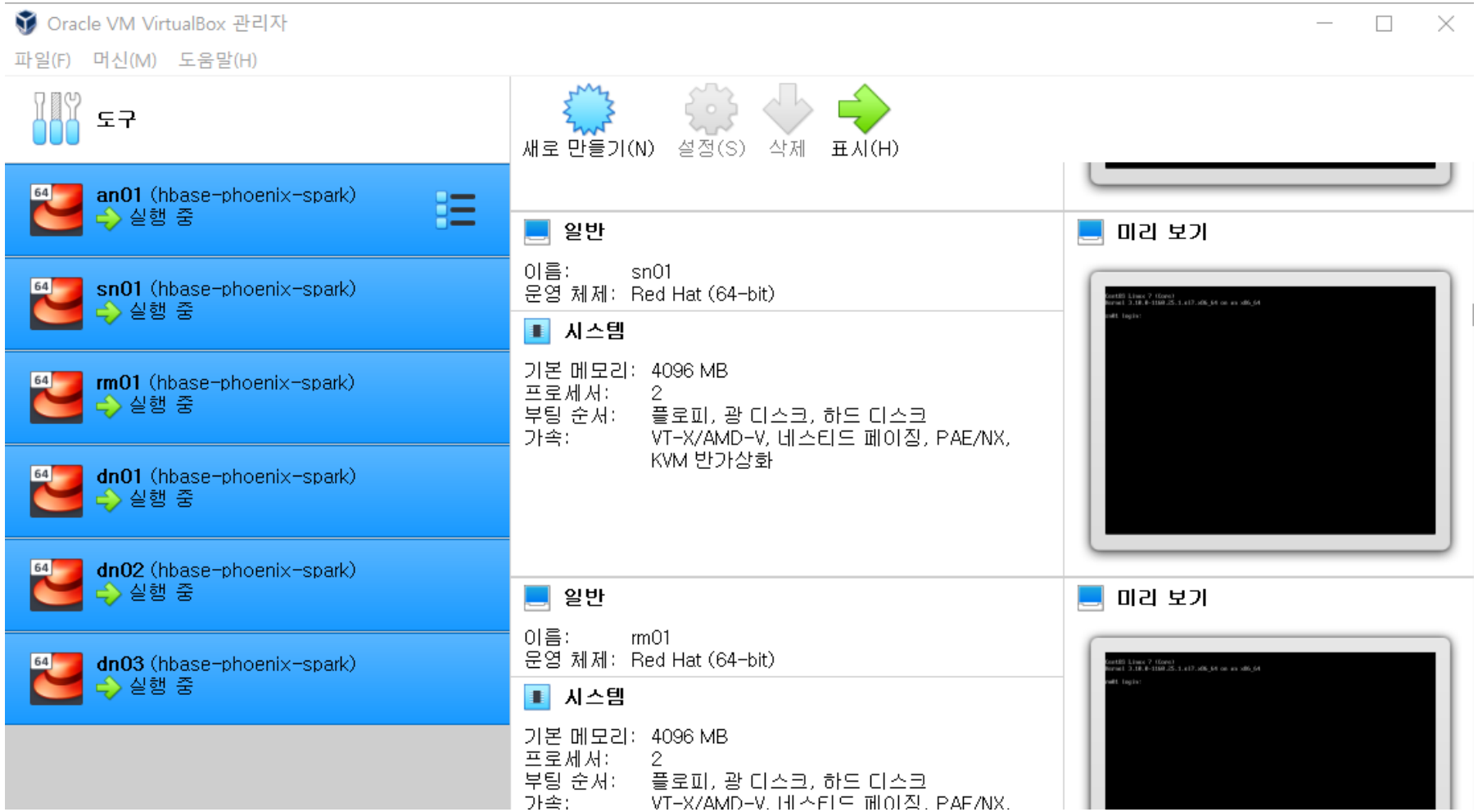
# Hadoop & HBase & Phoenix
# install and connect

- 서버 6대 가이드를 계속 진행한다는 가정
- Hadoop 3.1.2
- centos 7
- zookeeper 3.4.10
- HBase 2.2.5



# 1. HBase install

- **HBase 설치**
- **Hadoop 계정에서 운영**
- **an01 서버에서 설정 후 데이터 노드들에게 배포**
- **an01: master, dn01, dn02, dn03: region server**

action server: an01
pwd: /home/hadoop

wget http://archive.apache.org/dist/hbase/2.2.5/hbase-2.2.5-bin.tar.gz

mv hbase-2.2.5 hbase // 폴더명 변경

cd hbase/conf 경로 변경

# 1. HBase install

- **HBase 설정**

action server: an01
pwd: /home/hadoop/hbase/conf

vi hbase-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
    <property>
        <name>hbase.rootdir</name>
        <value>hdfs://NNHA/user/hbase</value>
    </property>

    <property>
        <name>hbase.master</name>
        <value>an01:6000</value>
    </property>

    <property>
        <name>hbase.cluster.distributed</name>
        <value>true</value>
    </property>
```

```
    <property>
        <name>hbase.zookeeper.quorum</name>
        <value>an01,sn01,rm01</value>
    </property>

    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>

    <property>
        <name>hbase.zookeeper.property.clientPort</name>
        <value>2181</value>
    </property>

    <property>
        <name>hbase.zookeeper.property.dataDir</name>
        <value>/home/hadoop/data/zookeeper</value>
    </property>

    <property>
        <name>dfs.datanode.max.xcievers</name>
        <value>4096</value>
    </property>
    <property>
        <name>hbase.unsafe.stream.capability.enforce</name>
        <value>false</value>
    </property>
</configuration>
```

# 1. HBase install

- **하둡 설정 파일 복사**

```
action server: an01
pwd: /home/hadoop/hbase/conf

cp /home/hadoop/hadoop-3.1.2/etc/hadoop/hdfs-site.xml /home/hadoop/hbase/conf/hdfs-site.xml
cp /home/hadoop/hadoop-3.1.2/etc/hadoop/core-site.xml /home/hadoop/hbase/conf/core-site.xml
```

- **region servers 설정**

```
action server: an01
pwd: /home/hadoop/hbase/conf

vi regionservers

dn01
dn02
dn03
```

# 1. HBase install

- **hbase 설정**

action server: an01
pwd: /home/hadoop/hbase/conf

vi hbase-env.sh

java 경로 설정
regionservers 파일 경로 설정
HBASE_MANAGES_ZK = false 설정 // 직접 구축한 주키퍼 환경에서 실행하겠다는 의미

```
# The java implementation to use.  Java 1.8+ required.
export JAVA_HOME=/opt/apps/jdk8
```

```
# File naming hosts on which HRegionServers will run.  $HBASE_HOME/conf/regionservers by default.
export HBASE_REGIONSERVERS=/home/hadoop/hbase/conf/regionservers
```

```
# Tell HBase whether it should manage it's own instance of ZooKeeper or not.
export HBASE_MANAGES_ZK=false
```

# 1. HBase install

- **하둡 jar 파일 HBase/lib 복사**

action server: an01
pwd: /home/hadoop/hbase/conf

vi hbase-env.sh

cp /home/hadoop/hadoop-3.1.2/share/hadoop/yarn/timelineservice/lib/htrace-core-3.1.0-incubating.jar
/home/hadoop/hbase/lib/htrace-core-3.1.0-incubating.jar

cd ~ // 경로 변경

- **재압축 후 배포**

action server: an01
pwd: /home/hadoop

tar xvfz hbase.tar.gz hbase-2.2.5

scp hbase.tar.gz hadoop@dn01:/home/hadoop
scp hbase.tar.gz hadoop@dn02:/home/hadoop
scp hbase.tar.gz hadoop@dn03:/home/hadoop

# 1. HBase install

- **압축 해제**

```
action server: dn01, dn02, dn03
pwd: /home/hadoop

tar xvfz hbase.tar.gz

exit // root 계정 전환
```

- **HBase 환경 변수 설정 및 폴더 생성**

```
action server: an01, dn01, dn02, dn03
pwd: /root

vi /etc/profile.d/hbase.sh

export HBASE_HOME=/home/hadoop/hbase
export PATH=$PATH:$HBASE_HOME/bin

source /etc/profile.d/hbase.sh

su - hadoop // 하둡 계정 재접속
mkdir data
cd data
mkdir zookeeper
cd /home/hadoop/hbase
```

# 1. HBase install

- **HBase 실행**
- **정상 실행 시 HMaster, HRegionServer가 실행됨**

action server: an01
pwd: /home/hadoop/hbase

./bin/start-hbase.sh
jps // 상태 확인

**an01**

```
[hadoop@an01 hbase]$ jps
1520 JournalNode
1682 DFSZKFailoverController
8807 HMaster
9048 Jps
1598 NameNode
[hadoop@an01 hbase]$
```

**dn01**

```
[hadoop@dn01 hbase]$ jps
1921 DataNode
3761 Jps
1429 NodeManager
3445 HRegionServer
[hadoop@dn01 hbase]$
```

**dn02**

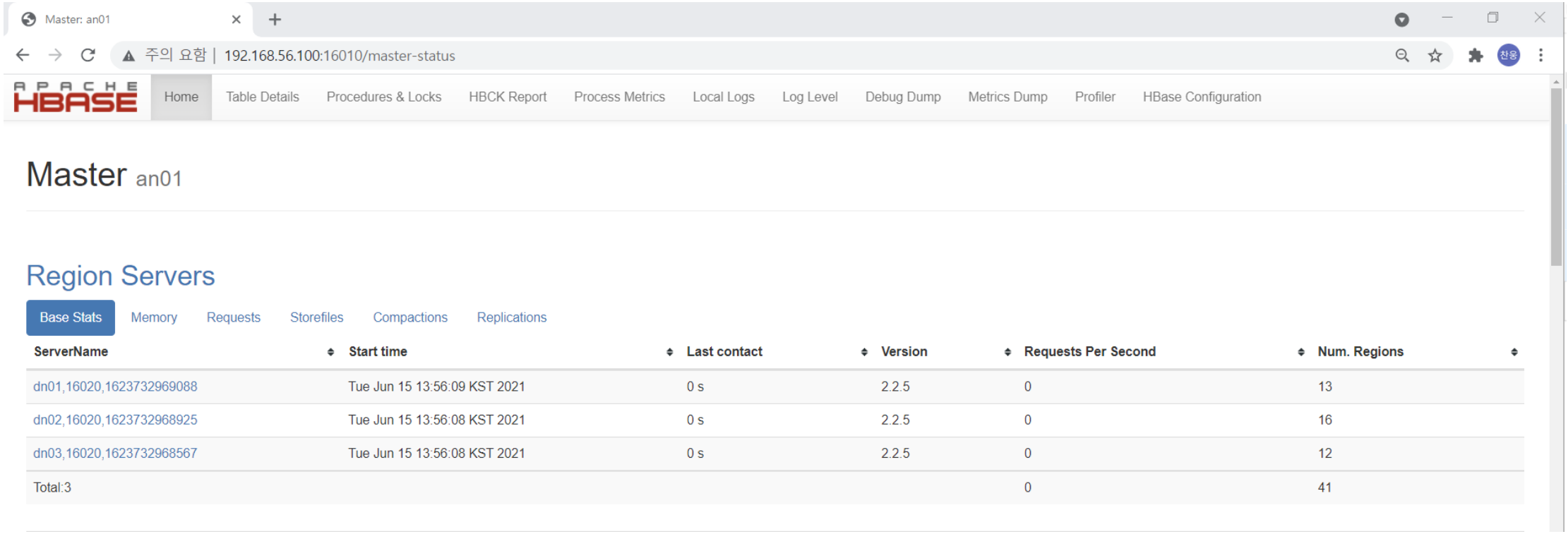
```
[hadoop@dn02 hbase]$ jps
1952 DataNode
1426 NodeManager
3893 HRegionServer
4219 Jps
[hadoop@dn02 hbase]$
```

**dn03**

```
[hadoop@dn03 hbase]$ jps
1425 NodeManager
3713 HRegionServer
4025 Jps
1917 DataNode
[hadoop@dn03 hbase]$
```

- **웹 확인**

http://192.168.56.100:16010



# 1. HBase install

- **csv 파일 insert 테스트**
- **지하철 예제 데이터가 an01:/home/hadoop/testdata 로컬에 저장되어있음을 가정**
- **csv 파일을 삽입할 빈 테이블 생성**

action server: an01
pwd: /home/hadoop/hbase

hdfs dfs -mkdir /user/batch_data // hdfs 폴더 생성
hdfs dfs -put /home/hadoop/testdata/subway2015.csv /user/batch_data // 로컬 csv -> hdfs

hbase shell // hbase 명령어 모드
create "SUBWAY2015", {NAME => "cf2015"} // 컬럼 패밀리가 cf2015인 테이블 생성
list // 테이블 확인
exit // 명령어 모드 종료

```
hbase(main):001:0> list
TABLE
SUBWAY2015
```

# 1. HBase install

- **csv 파일 insert**

action server: an01
pwd: /home/hadoop/hbase

./bin/hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator="," -Dimporttsv.columns="HBASE_ROW_KEY,cf2015:use_dt,cf2015:line_num,cf2015:sub_sta_nm,cf2015:ride_pasgr_num,cf2015:alight_pasgr_num,cf2015:work_dt" SUBWAY2015 /user/batch_data/subway2015.csv

1번째 줄: ImportTsv 임포트, 쉼표로 구분됨을 정의

2, 3번째 줄: 컬럼들을 정의, HBase 특성 상 고유한 row key가 있어야 함. 예제 csv파일에서는 이미 인덱스 열이 존재.
앞서 hbase shell을 통해 테이블 지정시 cf2015라는 컬럼 패밀리를 지정하였고, 컬럼 패밀리 내부의 컬럼들의 정의 후
테이블 명과 삽입할 hdfs 상의 csv파일의 경로를 정의

후 맵 리듀스 잡과 함께 데이터가 삽입

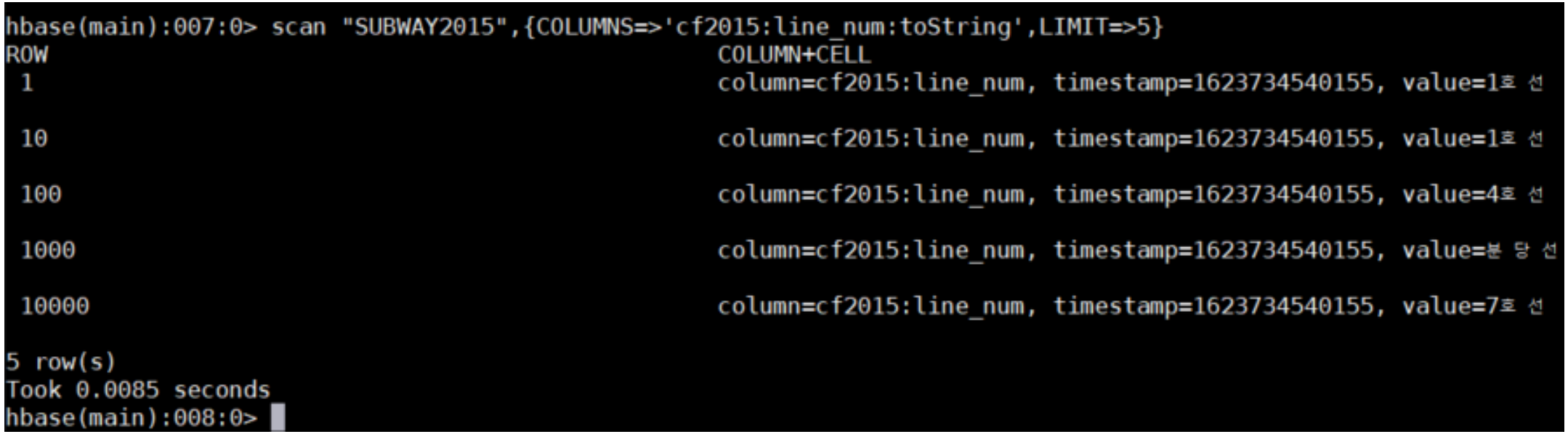
hbase shell // 명령어 모드 접속
scan 'SUBWAY2015', {LIMIT => 5} // SUBWAY2015 테이블의 5행만 출력, HBase의 특성 상 바이트 형식으로 출력된다.

```
hbase(main):002:0> scan 'SUBWAY2015', {LIMIT => 5}
ROW                                    COLUMN+CELL
 1                                     column=cf2015:alight_pasgr_num, timestamp=1623734540155, value=40197
 1                                     column=cf2015:line_num, timestamp=1623734540155, value=1\xED\x98\xB8\xEC\x84\xA0
 1                                     column=cf2015:ride_pasgr_num, timestamp=1623734540155, value=47071
 1                                     column=cf2015:sub_sta_nm, timestamp=1623734540155, value=\xEC\x84\x9C\xEC\x9A\xB8\xEC\x97\xAD
 1                                     column=cf2015:use_dt, timestamp=1623734540155, value=20150101
 1                                     column=cf2015:work_dt, timestamp=1623734540155, value=20151217
 10                                    column=cf2015:alight_pasgr_num, timestamp=1623734540155, value=5654
 10                                    column=cf2015:line_num, timestamp=1623734540155, value=1\xED\x98\xB8\xEC\x84\xA0
 10                                    column=cf2015:ride_pasgr_num, timestamp=1623734540155, value=5345
 10                                    column=cf2015:sub_sta_nm, timestamp=1623734540155, value=\xEB\x8F\x99\xEB\xAC\x98\xEC\x95\x9E
 10                                    column=cf2015:use_dt, timestamp=1623734540155, value=20150101
 10                                    column=cf2015:work_dt, timestamp=1623734540155, value=20151217
 100                                   column=cf2015:alight_pasgr_num, timestamp=1623734540155, value=7836
 100                                   column=cf2015:line_num, timestamp=1623734540155, value=4\xED\x98\xB8\xEC\x84\xA0
 100                                   column=cf2015:ride_pasgr_num, timestamp=1623734540155, value=8432
 100                                   column=cf2015:sub_sta_nm, timestamp=1623734540155, value=\xEB\xAF\xB8\xEC\x95\x84
 100                                   column=cf2015:use_dt, timestamp=1623734540155, value=20150101
 100                                   column=cf2015:work_dt, timestamp=1623734540155, value=20151217
 1000                                  column=cf2015:alight_pasgr_num, timestamp=1623734540155, value=7414
 1000                                  column=cf2015:line_num, timestamp=1623734540155, value=\xEB\xB6\x84\xEB\x8B\xB9\xEC\x84\xA0
 1000                                  column=cf2015:ride_pasgr_num, timestamp=1623734540155, value=6443
 1000                                  column=cf2015:sub_sta_nm, timestamp=1623734540155, value=\xEA\xB0\x80\xEC\xB2\x9C\xEB\x8C\x80
 1000                                  column=cf2015:use_dt, timestamp=1623734540155, value=20150102
 1000                                  column=cf2015:work_dt, timestamp=1623734540155, value=20151217
 10000                                 column=cf2015:alight_pasgr_num, timestamp=1623734540155, value=55118
 10000                                 column=cf2015:line_num, timestamp=1623734540155, value=7\xED\x98\xB8\xEC\x84\xA0
 10000                                 column=cf2015:ride_pasgr_num, timestamp=1623734540155, value=53647
 10000                                 column=cf2015:sub_sta_nm, timestamp=1623734540155, value=\xEA\xB0\x80\xEC\x82\xB0\xEB\x94\x94\xEC\xA7\x80\xED\x84\xB8\xEB\x8B\xA8\xEC\xA7\x80
 10000                                 column=cf2015:use_dt, timestamp=1623734540155, value=20150119
 10000                                 column=cf2015:work_dt, timestamp=1623734540155, value=20151217
5 row(s)
Took 0.0454 seconds
hbase(main):003:0>
```

# 1. HBase install

- **바이트 형식의 데이터 출력**

```
scan "SUBWAY2015",{COLUMNS=>'cf2015:line_num:toString',LIMIT=>5} // 저장된 데이터 형식에 맞게 튜닝
```



# 1. HBase install

- **테이블 삭제**
- **HBase의 테이블을 삭제하기 위해서는 먼저 테이블을 비활성화 해야함**
- **다음 Phoenix와의 연동 테스트를 위해 삭제 하지 않음.**

```
disable "SUBWAY2015" // 테이블 비활성화
drop "SUBWAY2015" // 테이블 삭제

list // 삭제 확인

exit // 명령어 모드 종료
```

- **HBase 종료**

```
action server: an01
pwd: /home/hadoop/hbase

./bin/stop-hbase.sh
```

# 2. Phoenix install

- **HBase를 SQL로 활용하기 위한 phoenix 연동**
- **Hbase 서버를 종료한 채 진행**
- **HMaster 서버에 설치**

action server: an01
pwd: /home/hadoop

wget https://mirror.navercorp.com/apache/phoenix/apache-phoenix-5.0.0-HBase-2.0/bin/apache-phoenix-5.0.0-HBase-2.0-bin.tar.gz

tar xvfz apache-phoenix-5.0.0-HBase-2.0-bin.tar.gz

mv /apache-phoenix-5.0.0-HBase-2.0-bin phoenix // 폴더명 변경

cd phoenix // 경로 변경

# 2. Phoenix install

- **phoenix-5.0.0-HBase-2.0-client.jar**
- **phoenix-5.0.0-HBase-2.0-server.jar**
- **phoenix-core-5.0.0-HBase-2.0.jar**
- **모든 HBase 클러스터의 lib 폴더에 복사**

```
action server: an01
pwd: /home/hadoop/phoenix

cp ./phoenix-5.0.0-HBase-2.0-client.jar /home/hadoop/hbase/lib/phoenix-5.0.0-HBase-2.0-client.jar
cp ./phoenix-5.0.0-HBase-2.0-server.jar /home/hadoop/hbase/lib/phoenix-5.0.0-HBase-2.0-server.jar
cp ./phoenix-core-5.0.0-HBase-2.0.jar /home/hadoop/hbase/lib/phoenix-core-5.0.0-HBase-2.0.jar

scp phoenix-5.0.0-HBase-2.0-server.jar hadoop@dn01:/home/hadoop/hbase/lib
scp phoenix-5.0.0-HBase-2.0-client.jar hadoop@dn01:/home/hadoop/hbase/lib
scp phoenix-core-5.0.0-HBase-2.0.jar hadoop@dn01:/home/hadoop/hbase/lib

scp phoenix-5.0.0-HBase-2.0-server.jar hadoop@dn02:/home/hadoop/hbase/lib
scp phoenix-5.0.0-HBase-2.0-client.jar hadoop@dn02:/home/hadoop/hbase/lib
scp phoenix-core-5.0.0-HBase-2.0.jar hadoop@dn03:/home/hadoop/hbase/lib

scp phoenix-5.0.0-HBase-2.0-server.jar hadoop@dn03:/home/hadoop/hbase/lib
scp phoenix-5.0.0-HBase-2.0-client.jar hadoop@dn03:/home/hadoop/hbase/lib
scp phoenix-core-5.0.0-HBase-2.0.jar hadoop@dn03:/home/hadoop/hbase/lib
```

# 2. Phoenix install

- **phoenix 환경 변수 설정**

```
action server: an01
pwd: /root

vi /etc/profile.d/phoenix.sh

export PHOENIX_HOME=/home/hadoop/phoenix
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin

source /etc/profile.d/phoenix.sh

su - hadoop
cd phoenix
```

# 2. Phoenix install

- **하둡, HBase 설정 파일을 phoenix/bin에 복사**

```
action server: an01
pwd: /home/hadoop/phoenix

cp /home/hadoop/hadoop-3.1.2/etc/hadoop/core-site.xml /home/hadoop/bin/phoenix/core-site.xml
cp /home/hadoop/hadoop-3.1.2/etc/hadoop/hdfs-site.xml /home/hadoop/bin/phoenix/hdfs-site.xml
cp /home/hadoop/hbase/conf/hbase-site.xml /home/hadoop/phoenix/bin/hbase-site.xml

cd .. // 경로 변경
```

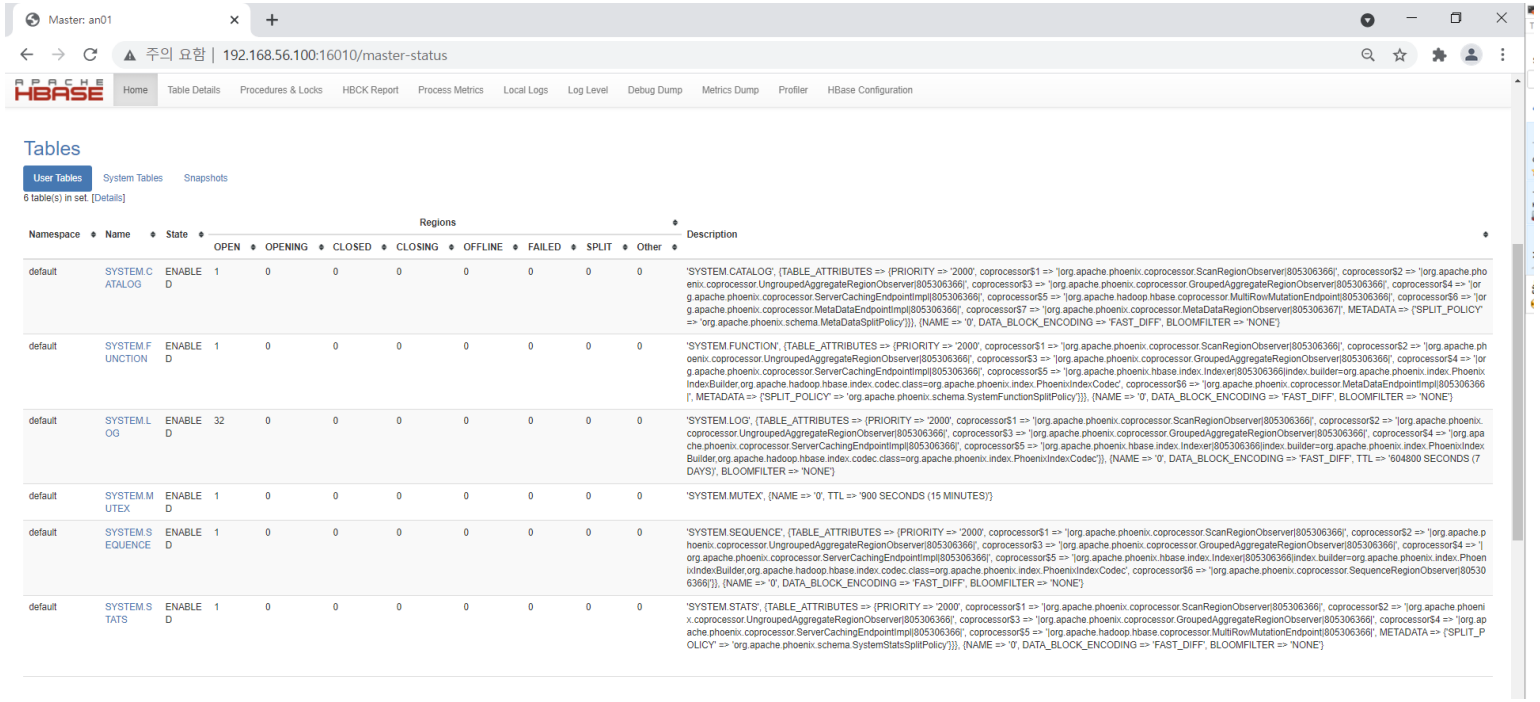
- **hbase, phoenix 실행**

```
action server: an01
pwd: /home/hadoop

cd hbase
./bin/start-hbase.sh
cd ~

cd phonix
./bin/sqlline.py // 명령어 모드 접속
• print error 시: python 버전 다운, phoenix는 python2와 호환됨

!tables // 테이블 확인
```

- **phoenix 실행 시 system과 관련된 테이블이 자동 생성됨**

```
0: jdbc:phoenix:> !tables
+-----------+-------------+-------------+---------------+---------+-----------+---------------------------+----------------+--------------+----------------+--------------+--------------+
| TABLE_CAT | TABLE_SCHEM | TABLE_NAME  |  TABLE_TYPE   | REMARKS | TYPE_NAME | SELF_REFERENCING_COL_NAME | REF_GENERATION | INDEX_STATE | IMMUTABLE_ROWS | SALT_BUCKETS | MULTI_TENANT |
+-----------+-------------+-------------+---------------+---------+-----------+---------------------------+----------------+--------------+----------------+--------------+--------------+
|           | SYSTEM      | CATALOG     | SYSTEM TABLE  |         |           |                           |                |              | false          | null         | false        |
|           | SYSTEM      | FUNCTION    | SYSTEM TABLE  |         |           |                           |                |              | false          | null         | false        |
|           | SYSTEM      | LOG         | SYSTEM TABLE  |         |           |                           |                |              | true           | 32           | false        |
|           | SYSTEM      | SEQUENCE    | SYSTEM TABLE  |         |           |                           |                |              | false          | null         | false        |
|           | SYSTEM      | STATS       | SYSTEM TABLE  |         |           |                           |                |              | false          | null         | false        |
+-----------+-------------+-------------+---------------+---------+-----------+---------------------------+----------------+--------------+----------------+--------------+--------------+
0: jdbc:phoenix:>
```

- **hbase web을 통해 자동 생성된 phoenix system 테이블이 hbase 테이블에도 생성되면 연동되었음을 확인(http://192.168.56.100:16010)**




| Namespace | Name | State | Regions | | | | | | | | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | OPEN | OPENING | CLOSED | CLOSING | OFFLINE | FAILED | SPLIT | Other | |
| default | SYSTEM.CATALOG | ENABLED | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 'SYSTEM.CATALOG', {TABLE_ATTRIBUTES => {PRIORITY => '2000', coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver\|805306366\|', coprocessor$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver\|805306366\|', coprocessor$3 => '|org.apache.phoenix.coprocessor.GroupedAggregateRegionObserver\|805306366\|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpointImpl\|805306366\|', coprocessor$5 => '|org.apache.hadoop.hbase.coprocessor.MultiRowMutationEndpoint\|805306366\|', coprocessor$6 => '|org.apache.phoenix.coprocessor.MetaDataEndpointImpl\|805306366\|', coprocessor$7 => '|org.apache.phoenix.coprocessor.MetaDataRegionObserver\|805306367\|', METADATA => {'SPLIT_POLICY' => 'org.apache.phoenix.schema.MetaDataSplitPolicy'}}}, {NAME => '0', DATA_BLOCK_ENCODING => 'FAST_DIFF', BLOOMFILTER => 'NONE'} |
| default | SYSTEM.FUNCTION | ENABLED | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 'SYSTEM.FUNCTION', {TABLE_ATTRIBUTES => {PRIORITY => '2000', coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver\|805306366\|', coprocessor$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver\|805306366\|', coprocessor$3 => '|org.apache.phoenix.coprocessor.GroupedAggregateRegionObserver\|805306366\|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpointImpl\|805306366\|', coprocessor$5 => '|org.apache.phoenix.hbase.index.Indexer\|805306366\|index.builder=org.apache.phoenix.index.PhoenixIndexBuilder,org.apache.hadoop.hbase.index.codec.class=org.apache.phoenix.index.PhoenixIndexCodec', coprocessor$6 => '|org.apache.phoenix.coprocessor.MetaDataEndpointImpl\|805306366\|', METADATA => {'SPLIT_POLICY' => 'org.apache.phoenix.schema.SystemFunctionSplitPolicy'}}}, {NAME => '0', DATA_BLOCK_ENCODING => 'FAST_DIFF', BLOOMFILTER => 'NONE'} |
| default | SYSTEM.LOG | ENABLED | 32 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 'SYSTEM.LOG', {TABLE_ATTRIBUTES => {PRIORITY => '2000', coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver\|805306366\|', coprocessor$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver\|805306366\|', coprocessor$3 => '|org.apache.phoenix.coprocessor.GroupedAggregateRegionObserver\|805306366\|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpointImpl\|805306366\|', coprocessor$5 => '|org.apache.phoenix.hbase.index.Indexer\|805306366\|index.builder=org.apache.phoenix.index.PhoenixIndexBuilder,org.apache.hadoop.hbase.index.codec.class=org.apache.phoenix.index.PhoenixIndexCodec'}, {NAME => '0', DATA_BLOCK_ENCODING => 'FAST_DIFF', TTL => '604800 SECONDS (7 DAYS)', BLOOMFILTER => 'NONE'} |
| default | SYSTEM.MUTEX | ENABLED | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 'SYSTEM.MUTEX', {NAME => '0', TTL => '900 SECONDS (15 MINUTES)'} |
| default | SYSTEM.SEQUENCE | ENABLED | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 'SYSTEM.SEQUENCE', {TABLE_ATTRIBUTES => {PRIORITY => '2000', coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver\|805306366\|', coprocessor$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver\|805306366\|', coprocessor$3 => '|org.apache.phoenix.coprocessor.GroupedAggregateRegionObserver\|805306366\|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpointImpl\|805306366\|', coprocessor$5 => '|org.apache.phoenix.hbase.index.Indexer\|805306366\|index.builder=org.apache.phoenix.index.PhoenixIndexBuilder,org.apache.hadoop.hbase.index.codec.class=org.apache.phoenix.index.PhoenixIndexCodec', coprocessor$6 => '|org.apache.phoenix.coprocessor.SequenceRegionObserver\|805306366\|'}}, {NAME => '0', DATA_BLOCK_ENCODING => 'FAST_DIFF', BLOOMFILTER => 'NONE'} |
| default | SYSTEM.STATS | ENABLED | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 'SYSTEM.STATS', {TABLE_ATTRIBUTES => {PRIORITY => '2000', coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver\|805306366\|', coprocessor$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver\|805306366\|', coprocessor$3 => '|org.apache.phoenix.coprocessor.GroupedAggregateRegionObserver\|805306366\|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpointImpl\|805306366\|', coprocessor$5 => '|org.apache.hadoop.hbase.coprocessor.MultiRowMutationEndpoint\|805306366\|', METADATA => {'SPLIT_POLICY' => 'org.apache.phoenix.schema.SystemStatsSplitPolicy'}}}, {NAME => '0', DATA_BLOCK_ENCODING => 'FAST_DIFF', BLOOMFILTER => 'NONE'} |

# 2. Phoenix install

- **hbase table을 view로 가져오기**

sqlline.py 모드에서

```
CREATE VIEW  " SUBWAY2015 "  (
ROWKEY VARCHAR PRIMARY KEY,
 " cf2015 " . " use_dt "  VARCHAR,
 " cf2015 " . " line_num "  VARCHAR,
 " cf2015 " . " sub_sta_nm "  VARCHAR,
 " cf2015 " . " ride_pasgr_num "  VARCHAR,
 " cf2015 " . " alight_pasgr_num "  VARCHAR,
 " cf2015 " . " work_dt "  VARCHAR);

select * from SUBWAY2015 limit 5;
select count(*) from SUBWAY2015;
```

- **HBase와 Phoenix의 연동으로 SQL을 사용할 수 있음**

```
0: jdbc:phoenix:> select * from SUBWAY2015 limit 5;
+---------+-----------+-----------+-------------+-----------------+-------------------+-----------+
| ROWKEY  |  use_dt   | line_num  | sub_sta_nm  | ride_pasgr_num  | alight_pasgr_num  |  work_dt  |
+---------+-----------+-----------+-------------+-----------------+-------------------+-----------+
| 1       | 20150101  | 1호 선     | 서 울 역      | 47071           | 40197             | 20151217  |
| 10      | 20150101  | 1호 선     | 동 묘 앞      | 5345            | 5654              | 20151217  |
| 100     | 20150101  | 4호 선     | 미 아         | 8432            | 7836              | 20151217  |
| 1000    | 20150102  | 분 당 선    | 가 천 대      | 6443            | 7414              | 20151217  |
| 10000   | 20150119  | 7호 선     | 가 산 디 지 털 단 지 | 53647           | 55118             | 20151217  |
+---------+-----------+-----------+-------------+-----------------+-------------------+-----------+
5 rows selected (0.2 seconds)
0: jdbc:phoenix:> select count(*) from SUBWAY2015;
+-----------+
| COUNT(1)  |
+-----------+
| 199380    |
+-----------+
1 row selected (1.49 seconds)
0: jdbc:phoenix:>
0: jdbc:phoenix:>
```