



TABA_DB/SQL실습2

전통적 데이터 처리 시스템

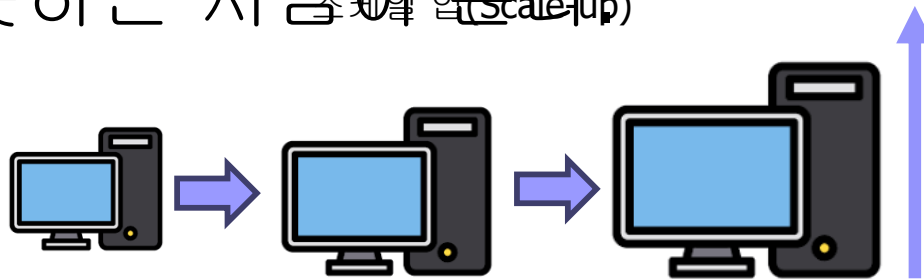
■ 전통적인 데이터 처리 시스템

- 과거에는 데이터 처리를 위한 시스템 확장이 어려웠음
- 전통적인 방식에서는 한 대의 컴퓨터의 처리 능력에 제한

전통적 데이터 처리 시스템

■ 스케일 업

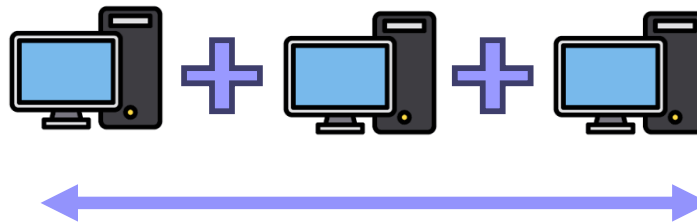
- 시스템을 확장할 때 구조를 바꿀 필요가 없다.
- 소프트웨어를 고사양 서버로 단순히 이관하는 방식으로 확장성을 높임 (생각보다 어려움)
- 언젠가는 스케일 업 방식으로 더 이상 확장하지 못하는 시점이 온다



전통적 데이터 처리 시스템

■ 스케일 아웃

- 여러 대의 장비에 처리를 분산 시키는 방식
- 스케일 업 방식에 비해 비용이 저렴
- 서버들에 데이터 처리 방법을 개발해야 했음.
- 스케줄링, 장애처리 고려 등
스케일 아웃(Scale-out)



전통적 데이터 처리 시스템

■ 한계점

- 큰 기업, 정부 기관, 학계를 제외한 곳에서 전통적인 스케일 업 및 스케일 아웃 방식을 그다지 사용하지 않았음.
- 스케일 업 : 비용이 비쌌음.
- 스케일 아웃 : 시스템 개발 및 관리가 어려움
- 여러 대의 호스트나 여러 개의 CPU 성능을 효과적으로 활용하기 어려움.
- 원하는 만큼 작업량을 효율적 처리하기 위한 전략은 엄청난 노력을 요함.

전통적 데이터 처리 시스템

■ 스케일 업의 한계

- 데이터양을 늘어나지만 하드웨어는 한계가 있음
- 고사양 서버를 한 대가 아닌 두 대, 세 대 ?
- 하이브리드 아키텍처는 하드웨어 구입 비용 및 클러스터 관리를 위한 로직 개발 두 가지 모두 필요.
- 빅데이터 처리 업계에서는 스케일 아웃 방식이 사실상 표준

전통적 데이터 처리 시스템

■ 스케일 업의 한계

- 많은 작업을 병렬 처리하는 하드웨어를 유연하게 사용하기 위해서는 소프트웨어를 똑똑하고, 하드웨어는 단순하게
- 하드웨어는 리소스 셋으로만 이용
- 소프트웨어가 처리 작업들에 하드웨어를 할당하는 역할

솔루션

■ 장애 예측

- 서버가 늘어나도 각 서버의 장애나 문제점은 영향을 주지 않게 해야함.
- 각각의 장비는 주기적으로 장애가 발생할 수 있다는 점을 고려

하둡(Hadoop)

■ 하둡

- 2003년, 2004년 구글 내부의 기술을 설명하는
- 구글 파일시스템(GFS)과 맵리듀스(MapReduce)
- 더그 커팅은 구글의 GFS와 맵리듀스 논문에서 영감을 받아 시스템을 구현
- 하둡은 아파치 오픈소스 재단의 최상위 레벨 프로젝트



출처 : <https://post.naver.com/viewer/postView.nhn?volumeNo=28925185&memberNo=50533718>

하둡(Hadoop)

■ 하둡 구성요소

□ 하둡 분산 파일 시스템(HDFS, Hadoop Distributed File System)

- 클러스터에 스케일 아웃 방식으로 대용량 데이터 셋을 저장하는 파일시스템
- 지연율보다 처리율에 최적화 / 이중화가 아닌 데이터 복제를 통해 고가용성을 얻음

□ 맵리듀스(MapReduce)

- 대용량 데이터를 병렬 처리 하기 위해 개발된 프로그래밍 모델
- 입력 데이터를 분산 처리하는 맵(Map) 함수 단계와 다시 하나의 결과물로 합치는 리듀스(Reduce) 함수 단계로 나눈다.

하둡(Hadoop)

■ 공통 구성 요소

□ HDFS와 맵리듀스는 아래와 같은 원칙을 지킨다.

- 저가 서버들로 구성된 클러스터에서 구동하도록 설계
- 서버를 추가함으로써 용량 및 성능을 확장하는 방식
- 장애 탐지 및 대응 메커니즘
- 사용자는 해결할 문제 자체만 집중하도록 시스템의 많은 부분을 드러내지 않는다.
- 물리적 시스템을 제어하는 소프트웨어 클러스터를 구성하는 아키텍처

하둡(Hadoop)

- 하둡의 세 가지 구동방식

- 로컬 자립형 방식(Standalone Mode)

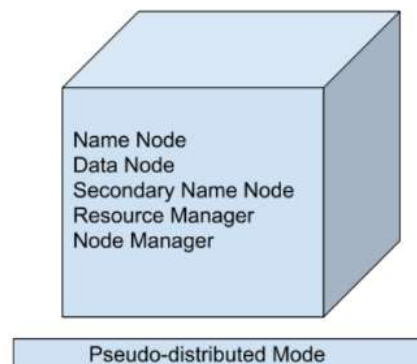
- Hadoop이 실행되는 기본 모드
 - HDFS를 사용하지 않음
 - mapred-site.xml, core-site.xml, hdfs-site.xml 등 구성 파일을 수정할 필요 없음
 - 디버깅 목적으로 사용

하둡(Hadoop)

■ 하둡의 세 가지 구동방식

□ 가분산 방식(Pseudo-distributed Mode)

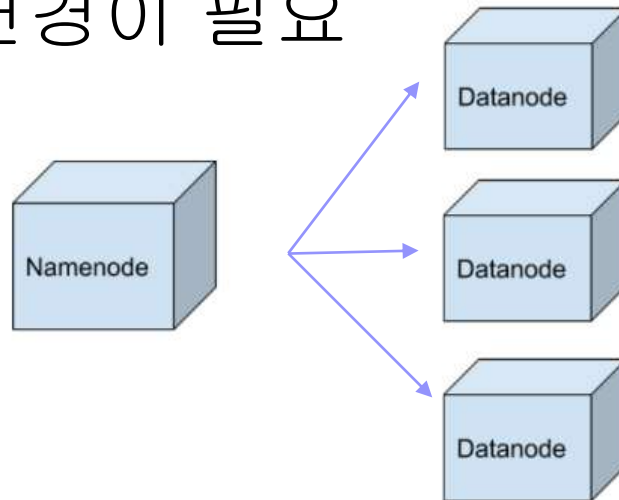
- Namenode와 Datanode가 모두 동일한 시스템에 있음
- 최적화된 미니 클러스터를 효과적으로 생성
- 구성 파일 변경이 필요



하둡(Hadoop)

■ 하둡의 세 가지 구동방식

- masternode와 datanode가 별도로 되어 있음
- 데이터가 여러 노드에 걸쳐 사용되고 분산
- 구성 파일 변경이 필요



VirtualBox설치

<https://www.virtualbox.org/>

VirtualBox

Home Download Documentation Comr

Powerful open source virtualization

For personal and enterprise use

VirtualBox is a general-purpose full virtualization software for x86_64 hardware (with version 7.1 additionally for macOS/Arm), targeted at laptop, desktop, server and embedded use.


Get Started

Download

Download VirtualBox binaries and platform packages

VirtualBox설치

본인 환경에 맞는 VirtualBox 설치



The screenshot shows the 'Download VirtualBox' page. At the top, it says 'Download VirtualBox' and provides a disclaimer about the PUEL license. Below this, there's a section titled 'VirtualBox Platform Packages'. A red rectangle highlights the 'VirtualBox 7.1.4 platform packages' list, which includes: Windows hosts, macOS / Intel hosts, macOS / Apple Silicon hosts, Linux distributions, Solaris hosts, and Solaris 11 IPS hosts. To the right, there's a section for the 'VirtualBox Extension Pack (PUEL)' with a 'PUEL License FAQ' button.

Download VirtualBox

The VirtualBox Extension Pack is available for personal and educational use on this page under the PUEL license. The PUEL is not available under commercial or enterprise terms. By downloading, you agree to the terms and conditions of the PUEL.

VirtualBox Platform Packages

VirtualBox 7.1.4 platform packages

- Windows hosts
- macOS / Intel hosts
- macOS / Apple Silicon hosts
- Linux distributions
- Solaris hosts
- Solaris 11 IPS hosts

Platform packages are released under the terms of the [GPL version 3](#).

This VirtualBox Extension Pack governs your access. The PUEL does not apply to the VirtualBox software, which is licensed under version 2 of the GPL.

See our [FAQ](#) for answers to common questions.

VirtualBox Extension Pack (PUEL)

[PUEL License FAQ](#)

CentOS 설치

iso 파일 다운(약 11GB)

https://mirror.stream.centos.org/9-stream/BaseOS/x86_64/iso/

CentOS Stream Mirror

This directory tree contains new CentOS Stream releases, starting from release "9-stream".
For previous CentOS Linux releases, see [CentOS mirror](#).

Name	Last modified	Size	Description
Parent Directory			
CentOS-Stream-9-20250910.x86_64-boot.iso	2025-01-01 03:58	11G	
CentOS-Stream-9-20250910.x86_64-boot.iso.MD5SUM	2025-01-01 04:28	147	
CentOS-Stream-9-20250910.x86_64-boot.iso.BaseOS.iso	2025-01-01 04:28	156	
CentOS-Stream-9-20250910.x86_64-boot.iso.SHA256SUM	2025-01-01 04:28	182	
CentOS-Stream-9-20250910.x86_64-boot.iso.manifest	2025-01-01 04:06	569	
CentOS-Stream-9-20250910.x86_64-dvd.iso	2025-01-01 04:13	0G	
CentOS-Stream-9-20250910.x86_64-dvd.iso.MD5SUM	2025-01-01 04:28	148	
CentOS-Stream-9-20250910.x86_64-dvd.iso.HISUM	2025-01-01 04:28	157	
CentOS-Stream-9-20250910.x86_64-dvd.iso.HASH256SUM	2025-01-01 04:28	183	
CentOS-Stream-9-20250910.x86_64-dvd.iso.manifest	2025-01-01 04:13	405K	
CentOS-Stream-9-latest.x86_64-boot.iso	2025-01-01 03:58	11G	
CentOS-Stream-9-latest.x86_64-boot.iso.MD5SUM	2025-01-02 10:58	139	
CentOS-Stream-9-latest.x86_64-boot.iso.HISUM	2025-01-02 10:58	148	
CentOS-Stream-9-latest.x86_64-boot.iso.SHA256SUM	2025-01-02 10:58	175	
CentOS-Stream-9-latest.x86_64-dvd.iso	2025-01-01 04:13	0G	
CentOS-Stream-9-latest.x86_64-dvd.iso.MD5SUM	2025-01-02 10:58	140	
CentOS-Stream-9-latest.x86_64-dvd.iso.HISUM	2025-01-02 10:58	149	
CentOS-Stream-9-latest.x86_64-dvd.iso.SHA256SUM	2025-01-02 10:58	175	
MD5SUM	2025-01-01 04:28	295	
SHA1SUM	2025-01-01 04:28	313	
SHA256SUM	2025-01-01 04:28	365	

CentOS 설치

Oracle VM VirtualBox 관리자

파일(F) | 편집(E) | 도구(T)

도구

ubuntu_vm
컨트롤 패널

ubuntu24_04 (실행)
컨트롤 패널

환경 설정(P)

가져오기

내보내기

새로 만들기(N)

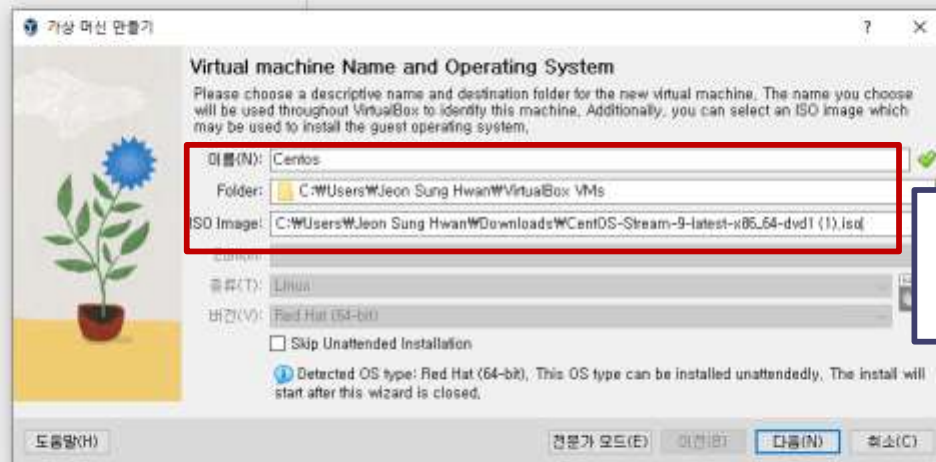
추가(A)

VirtualBox에 오신 것을 환영합니다.

이 프로그램의 왼쪽 부분에는 VirtualBox에 있는 모든 가상 머신과 가상 대선 그룹 목록을 표시합니다. 도구 모음의 단추를 사용하여 새로운 가상 머신을 만들거나, 추가하거나, 가져올 수 있습니다. 현재 선택한 구성 요소에 사용할 수 있는 도구 모음 단추를 눌러 해당하는 도구 모음을 호출할 수 있습니다.

F1 키를 누르면 상위에 있는 도움말을 볼 수 있으며, 최근 정보와 뉴스를 보려면 www.virtualbox.org를 방문하십시오.

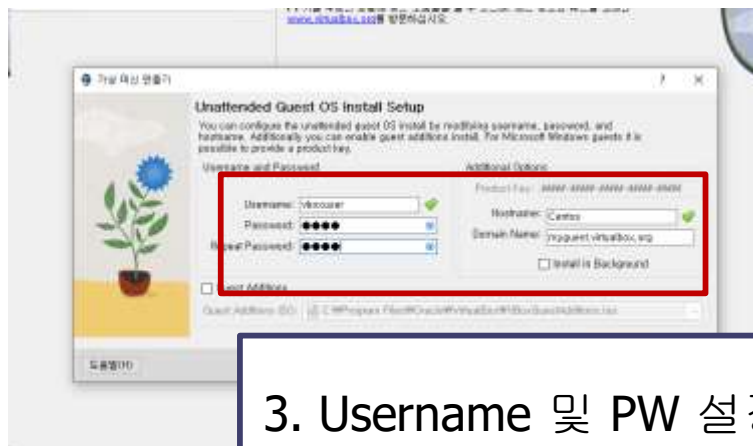
1. 새로만들기



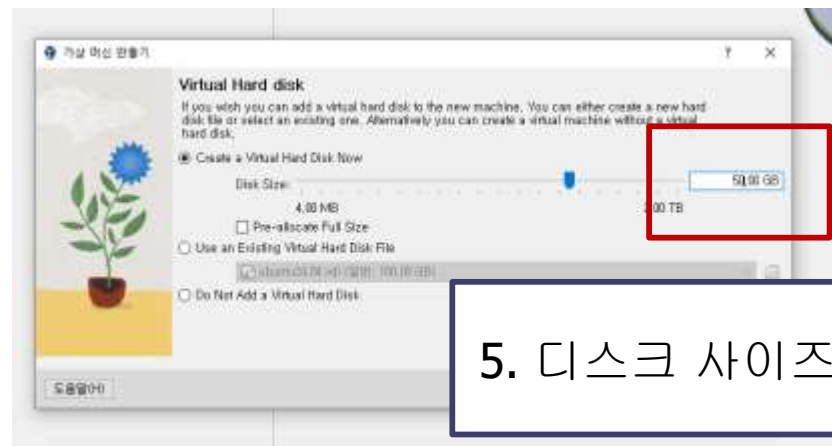
2. 이름, 폴더, ISO이미지 선택

CentOS 설치

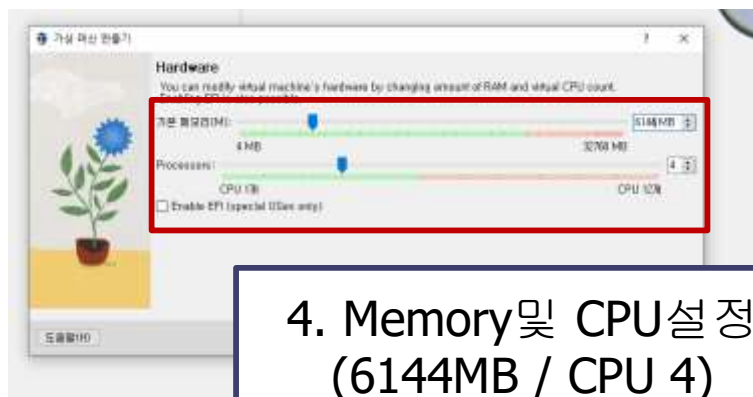
실습 환경은 본인 노트북 6144MB, CPU 4,
50GB를 권장하지만, 본인 스펙에 맞게 설정



3. Username 및 PW 설정



5. 디스크 사이즈 (50GB)

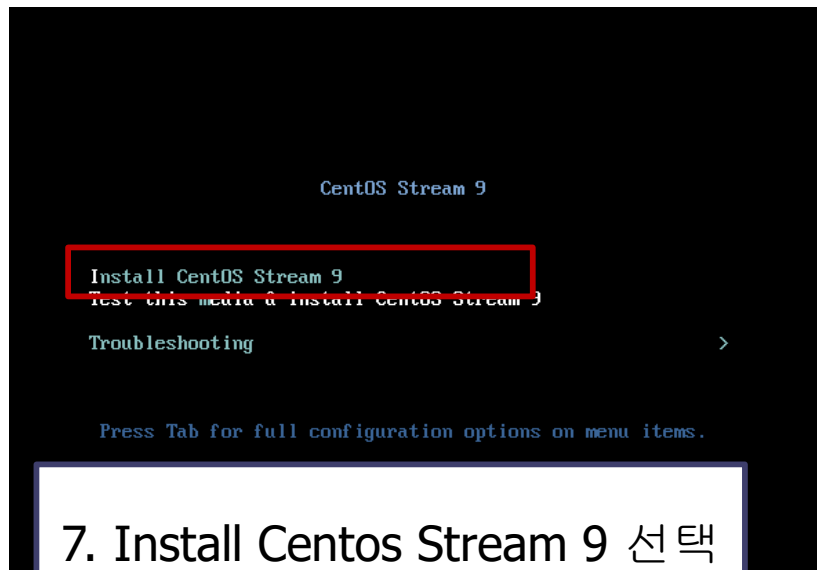


4. Memory 및 CPU 설정
(6144MB / CPU 4)

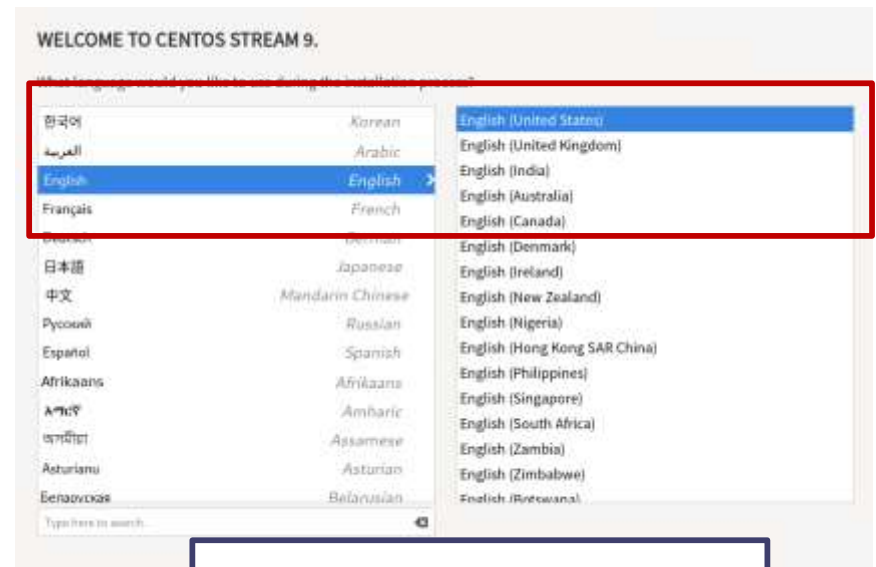


6. 요약 확인 후 Finish

CentOS 설치

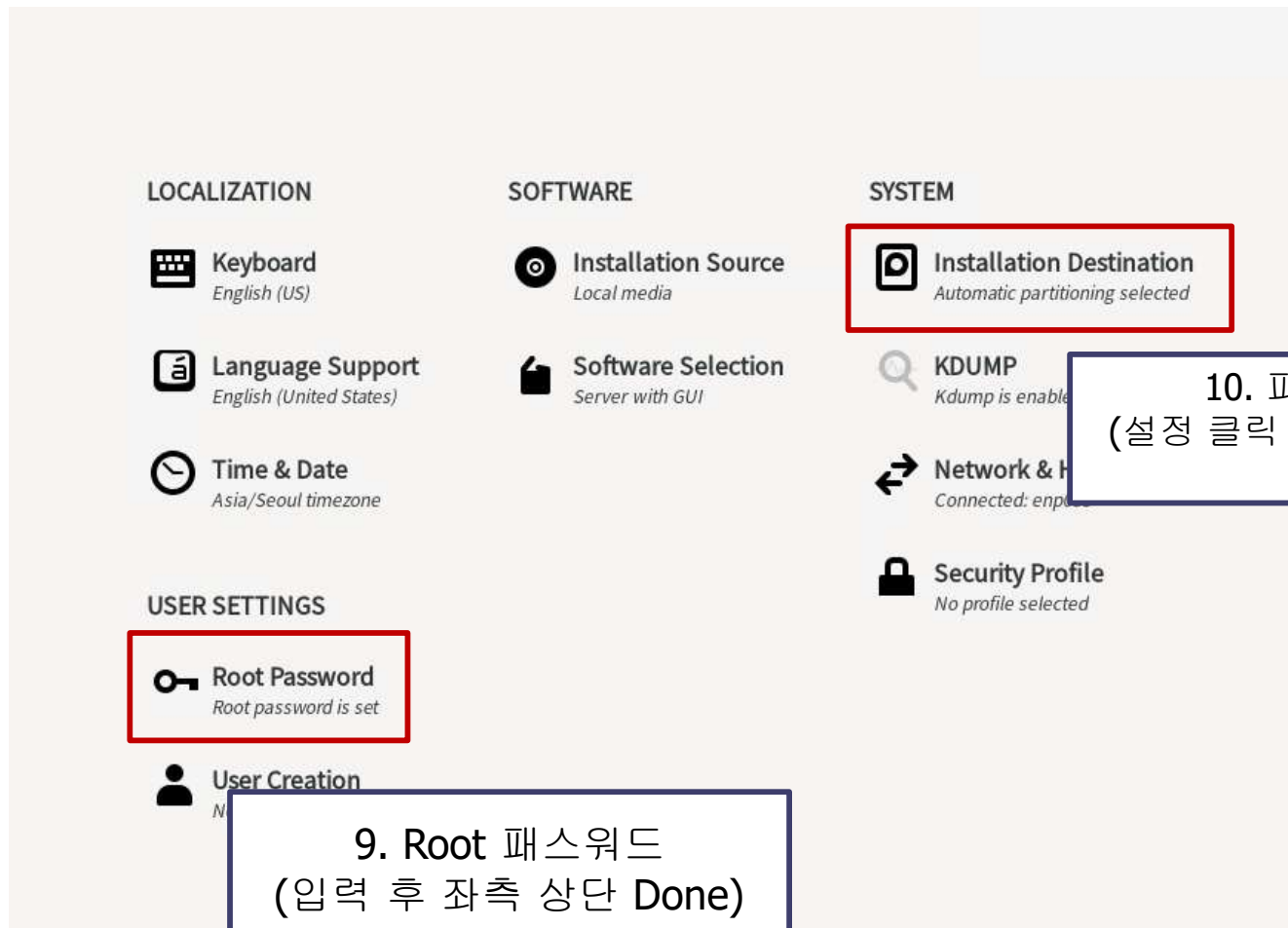


7. Install Centos Stream 9 선택



8. 언어 선택

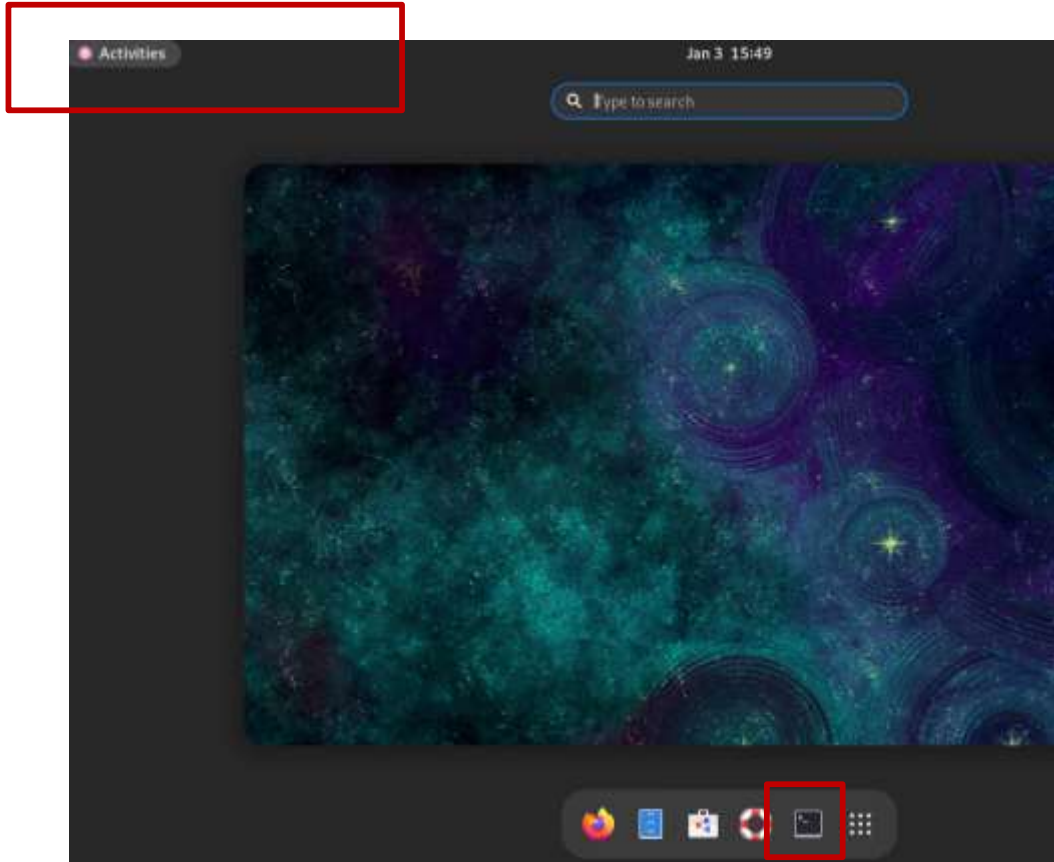
CentOS 설치



10. 파티셔닝 설정
(설정 클릭 후 좌측 상단 Done
클릭)

9. Root 패스워드
(입력 후 좌측 상단 Done)

CentOS 설치



11. 좌측 상단 Activities -
Terminal 클릭

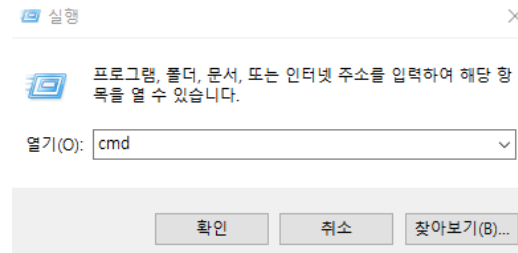
CentOS 설치

SSH VIM 등 필요한 프로그램 설치

```
sudo yum update && sudo yum install -y vim net-tools openssh-server
```

CentOS 설치

Windows cmd 실행(windows + R)
ipconfig라고 입력 후 ip확인



```
cmd 명령 프롬프트
Microsoft Windows [Version 10.0.19045.5247]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Jeon Sung Hwan>ipconfig

Windows IP 구성

이더넷 어댑터 이더넷 2:

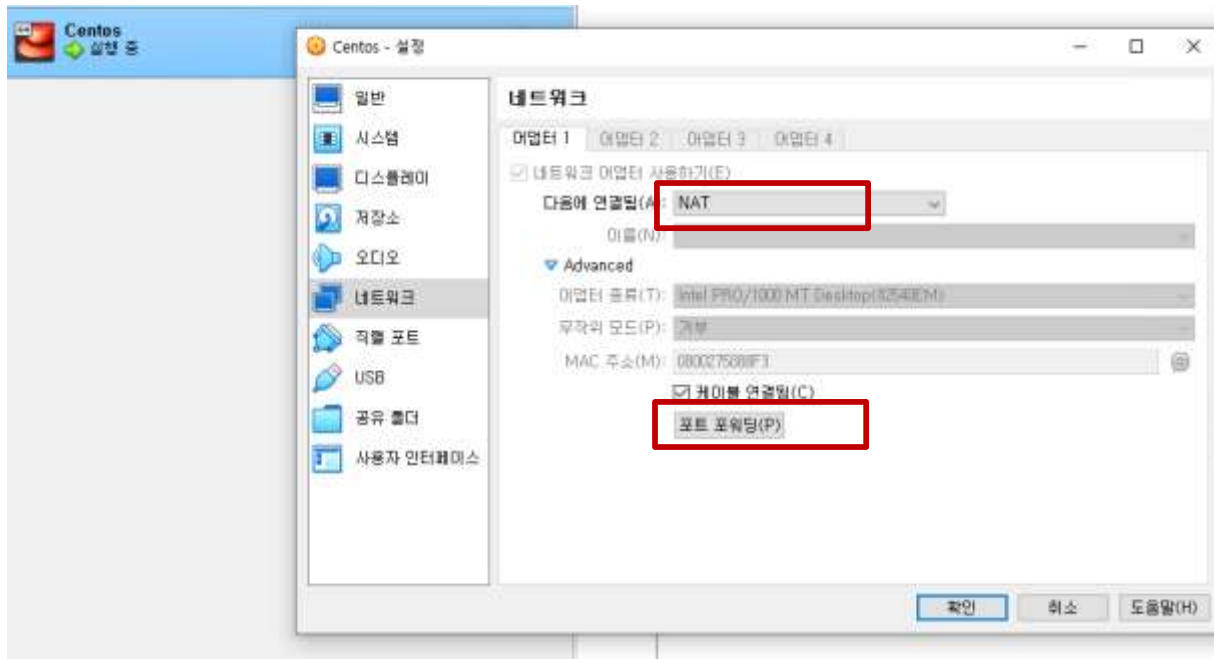
    연결별 DNS 접미사. . . . . : 
    링크-로컬 IPv6 주소. . . . . : fe80::2731:c802:6a91:c869%12
    자동 구성 IPv4 주소. . . . . : 169.254.216.241
    서브넷 마스크. . . . . : 255.255.0.0
    기본 게이트웨이. . . . . : 

이더넷 어댑터 이더넷 3:

    연결별 DNS 접미사. . . . . : 
    링크-로컬 IPv6 주소. . . . . : fe80::c5e9:276a:fa01:735d%40
    IPv4 주소. . . . . : 192.168.107.1
    서브넷 마스크. . . . . : 255.255.255.0
    기본 게이트웨이. . . . . :
```

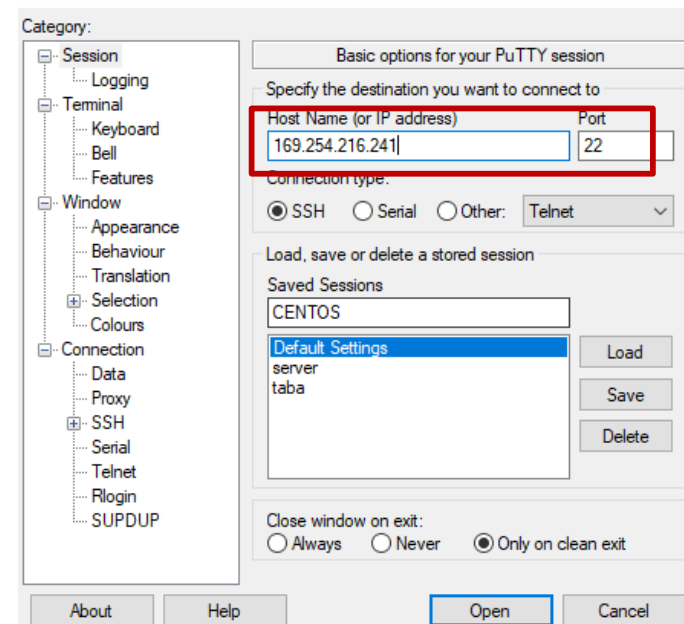
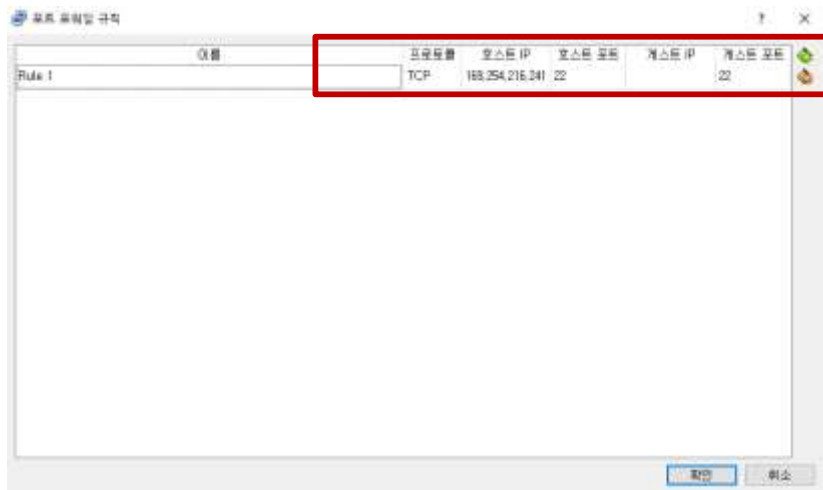

CentOS 설치

VirtualBox - 설정 - 네트워크



CentOS 설치

포워딩 규칙 추가 후 호스트 IP에 CMD에서 확인한 IP 입력 및 포트에 22 입력
Putty에서 IP입력 후 접속 가능



하둡 설치

- 실습환경 VirtualBox CentOS
- (CPU 4, Memory 6144(6G), 50GB이상) [권장사항임](#)
- AWS에서 실습 X

- **JAVA 설치 (1.8.0버전)**
 - `sudo yum install -y java-1.8.0-openjdk-devel`

하둡(Hadoop)

- 하둡 다운로드

`wget https://downloads.apache.org/hadoop/common/hadoop-3.2.4/hadoop-3.2.4.tar.gz --no-check-certificate`

- 하둡 압축 풀기

`tar -xzvf hadoop-3.2.4.tar.gz`

하둡(Hadoop) cont.

- ##JAVA_HOME 위치 찾는 법
- readlink -f /usr/bin/java
- 환경변수 설정(~/.bashrc 파일 설정)

vi ~/.bashrc

하둡(Hadoop) cont.

자바의 위치는 실습자료와 다를 수 있으니 꼭 확인

```
##JAVA_HOME##
```

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.362.b09-4.el9.x86_64/
```

```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# User specific environment
if ! [[ "$PATH" =~ "$HOME/.local/bin:$HOME/bin:" ]]
then
    PATH="$HOME/.local/bin:$HOME/bin:$PATH"
fi
export PATH

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions
if [ -d ~/.bashrc.d ]; then
    for rc in ~/.bashrc.d/*; do
        if [ -f "$rc" ]; then
            . "$rc"
        fi
    done
fi

unset rc

##JAVA_HOME##
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.362.b09-4.el9.x86_64/
```

하둡(Hadoop)

```
## HADOOP_HOME ##  
export HADOOP_HOME=/home/taeba/hadoop-3.2.4  
export HADOOP_INSTALL=$HADOOP_HOME  
export HADOOP_MAPRED_HOME=$HADOOP_HOME  
export HADOOP_COMMON_HOME=$HADOOP_HOME  
export HADOOP_HDFS_HOME=$HADOOP_HOME  
export YARN_HOME=$HADOOP_HOME  
export HADOOP_YARN_HOME=$HADOOP_HOME  
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native  
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin:$JAVA_HOME/bin  
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
```

:wq로 저장 후

source ~/.bashrc #환경변수 적용

echo \$HADOOP_HOME #적용 확인

```
## JAVA_HOME ##  
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.362.b09-4.el9.x86_64  
  
## HADOOP_HOME ##  
export HADOOP_HOME=/home/taeba/hadoop-3.2.4  
export HADOOP_INSTALL=$HADOOP_HOME  
export HADOOP_MAPRED_HOME=$HADOOP_HOME  
export HADOOP_COMMON_HOME=$HADOOP_HOME  
export HADOOP_HDFS_HOME=$HADOOP_HOME  
export YARN_HOME=$HADOOP_HOME  
export HADOOP_YARN_HOME=$HADOOP_HOME  
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native  
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin:$JAVA_HOME/bin  
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
```

하둡(Hadoop)

■ hadoop-env.sh 파일 설정

- vi \$HADOOP_HOME/etc/hadoop/hadoop-env.sh
- #export JAVA_HOME 주석 제거 후 JAVA_HOME 입력

```
###
# Generic settings for HADOOP
###

# Technically, the only required environment variable is JAVA_HOME.
# All others are optional. However, the defaults are probably not
# preferred. Many sites configure these options outside of Hadoop,
# such as in /etc/profile.d

# The java implementation to use. By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.362.b09-4.el9.x86_64

# Location of Hadoop. By default, Hadoop will attempt to determine
```


하둡(Hadoop)

경로 확인!
본인 계정에 맞는 경로로
설정

■ core-site.xml 설정

- HDFS와 하둡 핵심 property 정의
- vi \$HADOOP_HOME/etc/hadoop/core-site.xml

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/home/taeba/tmpdata</value>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/home/hadoop/tmpdata</value>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://127.0.0.1:9000</value>
  </property>
</configuration>
```

- mkdir /home/taeba/tmpdata #디렉토리 생성

하둡(Hadoop)

경로 확인!
본인 계정에 맞는 경로로
설정

■ hdfs-site.xml 설정

- vi \$HADOOP_HOME/etc/hadoop/hdfs-site.xml
- namenode와 datanode 저장소 디렉토리 설정

```
<configuration>
  <property>
    <name>dfs.data.dir</name>
    <value>/home/taba/dfsdata/namenode</value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>/home/taba/dfsdata/datanode</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

```
<configuration>
  <property>
    <name>dfs.data.dir</name>
    <value>/home/hadoop/dfsdata/namenode</value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>/home/hadoop/dfsdata/datanode</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

하둡(Hadoop)

■ mapred-site.xml 설정

- vi \$HADOOP_HOME/etc/hadoop/mapred-site.xml
- 맵리듀스 파일 값을 정의

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

하둡(Hadoop)

■ yarn-site.xml 설정

- vi \$HADOOP_HOME/etc/hadoop/yarn-site.xml
- YARN에 관련된 세팅을 정의하는 파일

```
<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.hostname</name>
  <value>0.0.0.0</value>
</property>
<property>
  <name>yarn.resourcemanager.address</name>
  <value>0.0.0.0:8032</value>
</property>
```

하둡(Hadoop)

■ yarn-site.xml 설정

□ YARN에 관련된 세팅을 정의하는 파일

```
<property>
  <name>yarn.web-proxy.address</name>
  <value>0.0.0.0:8089</value>
</property>
<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>
  <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREPEND_DISTCACHE,
  HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
</property>
```

하둡(Hadoop)

■ yarn-site.xml 설정

```
<configuration>
  <!-- Site specific Hadoop configuration properties -->
  <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.hostname</name>
    <value>0.0.0.0</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>0.0.0.0:8032</value>
  </property>
  <property>
    <name>yarn.web-proxy.address</name>
    <value>0.0.0.0:8080</value>
  </property>
  <property>
    <name>yarn.acl.enable</name>
    <value>0</value>
  </property>
  <property>
    <name>yarn.nodemanager.env-whitelist</name>
    <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
  </property>
</configuration>
```

하둡(Hadoop)

- SSH-KEYGEN

- 하둡은 각 노드들이 통신을 해야하기 때문에 패스워드 없이 접속하도록 ssh-keygen 설정

```
ssh-keygen -t rsa -P " " -f ~/.ssh/id_rsa  
cat ~/.ssh/id_rsa.pub >>  
~/.ssh/authorized_keys  
chmod 0600 ~/.ssh/authorized_keys
```

하둡(Hadoop)

■ 네임노드 포맷

- **hadoop**을 가동하기 전 **HDFS** 파일시스템을 포맷해야함

hdfs namenode -format

```
hadoop@ip-172-31-2-77:~/hadoop-3.2.4$ hdfs namenode -format
WARNING: /home/hadoop/hadoop-3.2.4/logs does not exist. Creating.
2024-03-19 02:58:58,941 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = ip-172-31-2-77/172.31.2.77
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.2.4
STARTUP_MSG:   classpath = /home/hadoop/hadoop-3.2.4/etc/hadoop:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/commons-collections-3.2.2.jar:/home/
/hadoop/hadoop-3.2.4/share/hadoop/common/lib/kerb-admin-1.0.1.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jsr311-api-1.1.1.jar:/home/hadoop/h
hadoop/hadoop-3.2.4/share/hadoop/common/lib/commons-math3-3.1.1.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/paranamer-2.3.jar:/home/hadoop/h
hadoop-3.2.4/share/hadoop/common/lib/gson-2.9.0.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jsr305-3.0.2.jar:/home/hadoop/hadoop-3.2.4/share
2.4/share/hadoop/common/lib/kerb-common-1.0.1.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jetty-server-9.4.43.v20210629.jar:/home/hadoop/had
/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jetty-io-9.4.43.v20210629.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/httpcore-4.4.13.jar:/home
ome/hadoop/hadoop-3.2.4/share/hadoop/common/lib/animal-sniffer-annotations-1.17.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jersey-server-1
.5.2.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/kerb-core-1.0.1.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jackson-databind-2.10
a-2.5.0.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jaxb-api-2.2.11.jar:/home/hadoop/hadoop-3.2.4/share/hadoop/common/lib/jaxb-impl-2.2.3-1
```


하둡(Hadoop)

■ 하둡 실행하기

- start-dfs.sh
- start-yarn.sh
- jps

```
hadoop@ip-172-31-2-77:~/hadoop-3.2.4$ jps
5968 ResourceManager
5763 SecondaryNameNode
5459 NameNode
6262 WebAppProxyServer
6088 NodeManager
5581 DataNode
6509 Jps
```

하둡 예제 (Wordcount)

■ 하둡 명령어

hdfs dfs [GENERIC_OPTIONS]
[COMMAND_OPTIONS]

- cat -파일 내용 출력

hdfs dfs -cat

- cp -hdfs 내부에서 파일을 복사

hdfs dfs -cp

- mkdir -특정 path에 폴더 생성

hdfs dfs -mkdir

- mv -hdfs 내부에서 파일 옮기기

hdfs dfs -mv

- put -local에서 파일을 hdfs에 저장

hdfs dfs -input

- copyToLocal -hdfs에 있는 파일을 local에 다운

hdfs dfs -copyToLocal

- du -hdfs 내부 특정 file이나 디렉토리의 사이즈를 보여줌

hdfs dfs -du

- ls -특정 디렉토리의 파일 혹은 디렉토리 출력

hdfs dfs -ls

- rm - hdfs에서 폴더 혹은 파일 삭제

hdfs dfs -rm

하둡 예제 (Wordcount)

- Wordcount에 사용될 예제 파일 다운

- wget <http://www.gutenberg.org/cache/epub/1661/pg1661.txt>

- hdfs에 폴더 생성하기

- hdfs dfs -mkdir -p /sample/input

- hdfs에 예제파일 이동하기

- hdfs dfs -put pg1661.txt /sample/input

하둡 예제 (Wordcount)

■ 워드카운트 실행하기

```
hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.4.jar wordcount /sample/input /sample/output
```

■ 결과물 확인

```
hdfs dfs -cat /sample/output/part*
```

```
sundial 1  
sundial, 1  
sundial,' 1  
sundial.''' 1  
sundial?' 1  
sundials 1  
sunk 8  
sunk, 1  
sunlight; 1  
sunset, 1  
sunshine. 1  
superb 1  
superior 1  
superior, 1  
superscribed 1  
superscription 1  
supper 4  
supper, 2  
supper." 1  
supplementing 1
```

하둡 예제 (Wordcount)

■ Teragen / terasort

- 하둡 성능을 측정하기 위해 사용
- 일정 데이터를 생성하고 데이터를 정렬할 때 속도를 측정

```
hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.4.jar  
teragen \  
-Ddfs.replication=1 \  
-Dmapred.map.tasks=36 \  
-Dmapred.reduce.tasks=16 \  
5000 /teragen/teragen.data
```

복사본 1개 맵 테스트 36 / 리듀스 테스트 16 / 생성 용량 5,000/ 생성된 데이터 저장할 곳

하둡 예제 (Wordcount)

■ Teragen

■ 생성된 파일 확인

```
oem@ubuntu-server:~$ hdfs dfs -ls /teragen
Found 2 items
-rw-r--r-- 1 oem supergroup 0 2024-03-24 18:07 /teragen/_SUCCESS
-rw-r--r-- 1 oem supergroup 10737418200 2024-03-24 18:07 /teragen/part-m-00000
```

```
oem@ubuntu-server:~$ hdfs dfs -ls /teragen
Found 2 items
-rw-r--r-- 1 oem supergroup 0 2024-03-24 18:07 /teragen/_SUCCESS
-rw-r--r-- 1 oem supergroup 10737418200 2024-03-24 18:07 /teragen/part-m-00000
```

하둡 예제 (Wordcount)

■ terasort

```
hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.4.jar  
terasort /teragen/teragen.data /teragen/terasort.data
```

하둡 예제 (Wordcount)

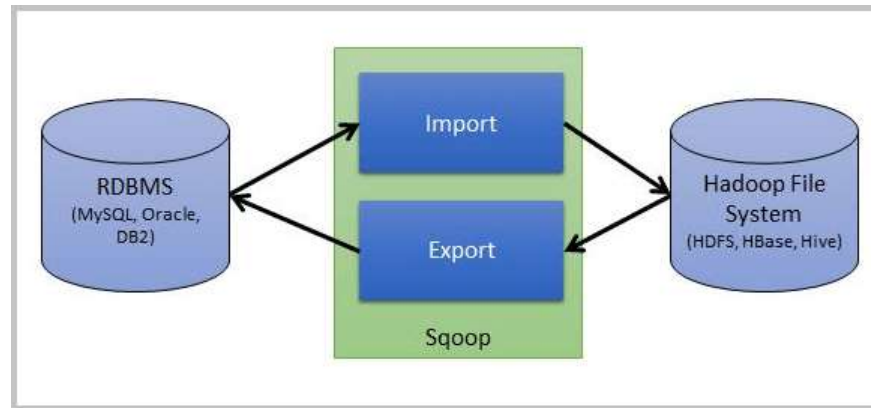
```
2024-03-24 18:26:34,219 INFO mapreduce.Job: Counters: 36
  File System Counters
    FILE: Number of bytes read=485795790745
    FILE: Number of bytes written=938992151454
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=446412866780
    HDFS: Number of bytes written=10737418200
    HDFS: Number of read operations=8570
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=164
    HDFS: Number of bytes read erasure-coded=0
  Map-Reduce Framework
    Map input records=107374182
    Map output records=107374182
    Map output bytes=11852168564
    Map output materialized bytes=11166915408
    Input split bytes=8680
    Combine input records=0
    Combine output records=0
    Reduce input groups=107374182
    Reduce shuffle bytes=11166915408
    Reduce input records=107374182
    Reduce output records=107374182
    Spilled Records=322122546
    Shuffled Maps=80
    Failed Shuffles=0
    Merged Map outputs=0
    GC time elapsed (ms)=2992
    Total committed heap usage (bytes)=147702939640
  Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
  File Input Format Counters
    Bytes Read=10737418200
  File Output Format Counters
    Bytes Written=10737418200
2024-03-24 18:26:34,219 INFO terasort.TeraSort: done

real    5m30.607s
user    5m15.890s
sys     8m44.482s
```


Apache Sqoop

■ Apache Sqoop(SQL to Hadoop)

- Hadoop과 관계형 데이터베이스 간 데이터를 전송할 수 있도록 설계된 오픈소스 소프트웨어
- Oracle, MySQL 등 RDBMS 특정 테이블, 데이터를 HDFS로 옮길 수 있음
- 반대로 HDFS에 저장된 데이터를 RDBMS로 옮길 수 있음
- Sqoop은 2009년 처음 버전이 나왔고, 현재 프로젝트는 종료됨



Apache Sqoop

■ 스쿱 설치

cd /home/taeba

wget https://archive.apache.org/dist/sqoop/1.4.7/sqoop-1.4.7.bin__hadoop-2.6.0.tar.gz

tar -xzf sqoop-1.4.7.bin__hadoop-2.6.0.tar.gz #압축해제

mv sqoop-1.4.7.bin__hadoop-2.6.0 sqoop #폴더명 변경

rm sqoop-1.4.7.bin__hadoop-2.6.0.tar.gz #tar파일 삭제

vi ~/.bashrc #환경변수 추가

export SQOOP_HOME=/home/taeba/sqoop

```
export HADOOP_HOME=/home/hadoop/hadoop-3.2.4
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
export SQOOP_HOME=/home/hadoop/sqoop
```

Apache Sqoop

- sqoop 환경설정

PATH에 Sqoop도 추가

```
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin:$SQOOP_HOME/bin  
source ~/.bashrc
```

```
# SQOOP ##  
export SQOOP_HOME=/home/hadoop/sqoop  
  
## HADOOP ##  
export HADOOP_HOME=/home/hadoop/hadoop-3.2.4  
export HADOOP_INSTALL=$HADOOP_HOME  
export HADOOP_MAPRED_HOME=$HADOOP_HOME  
export HADOOP_COMMON_HOME=$HADOOP_HOME  
export HADOOP_HDFS_HOME=$HADOOP_HOME  
export YARN_HOME=$HADOOP_HOME  
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native  
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"  
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin:$SQOOP_HOME/bin
```

Apache Sqoop

■ sqoop 환경 설정

cp \$SQOOP_HOME/conf/sqoop-env-template.sh sqoop/conf/sqoop-env.sh

vi \$SQOOP_HOME/conf/sqoop-env.sh

export HADOOP_COMMON_HOME=\$HADOOP_HOME

export HADOOP_MAPRED_HOME=\$HADOOP_HOME

```
#Set path to where bin/hadoop is available
export HADOOP_COMMON_HOME=$HADOOP_HOME

#Set path to where hadoop-*.core.jar is available
export HADOOP_MAPRED_HOME=$HADOOP_HOME

#Set the path to where bin/hbase is available
#export HBASE_HOME=

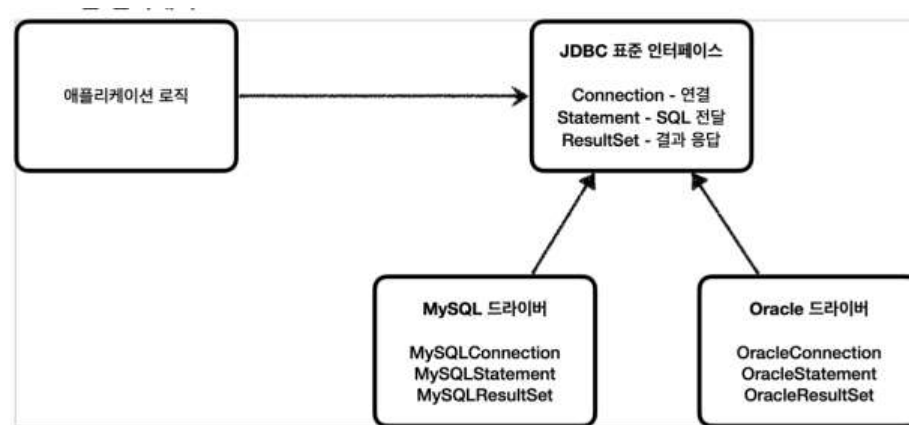
#Set the path to where bin/hive is available
#export HIVE_HOME=

#Set the path for where zookeeper config dir is
#export ZOOKEEPER=

#
```

Sqoop 예제

- MySQL JDBC 다운
- JDBC(Java Database Connectivity) 는 Java기반 애플리케이션의 애플리케이션의 데이터를 데이터베이스에서 저장 및 업데이트하거나, 데이터베이스에 저장된 데이터를 Java에서 사용할 수 있도록 하는 자바 API



Sqoop 예제

- MySQL JDBC 다운 및 Sqoop lib 디렉토리에 옮기기

wget <https://downloads.mysql.com/archives/get/p/3/file/mysql-connector-j-8.0.33.tar.gz>

tar -xzf mysql-connector-j-8-0.33.tar.gz

cp mysql-connector-j-8.0.33/mysql-connector-j-8.0.33.jar \$SQOOP_HOME/lib/

- StringUtils 관련 오류 예방으로 commons 라이브러리 설치

wget <https://dlcdn.apache.org/commons/lang/binaries/commons-lang-2.6-bin.tar.gz>

tar -zxvf commons-lang-2.6-bin.tar.gz

cp commons-lang-2.6/commons-lang-2.6.jar \$SQOOP_HOME/lib

Sqoop 예제

■ MySQL 설치

`sudo yum install -y mysql-server`

`sudo systemctl enable mysqld && sudo systemctl start mysqld && sudo systemctl status mysqld`

```
[taba@localhost ~]$ sudo systemctl enable mysqld && sudo systemctl start mysqld && sudo systemctl status mysqld
Created symlink /etc/systemd/system/multi-user.target.wants/mysqld.service → /usr/lib/systemd/system/mysqld.service.
● mysqld.service - MySQL 8.0 database server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; preset: disabled)
   Active: active (running) since Fri 2025-01-03 18:35:08 KST; 35ms ago
     Process: 43465 ExecStartPre=/usr/libexec/mysql-check-socket (code=exited, status=0/SUCCESS)
     Process: 43487 ExecStartPre=/usr/libexec/mysql-prepare-db-dir mysqld.service (code=exited, status=0/SUCCESS)
    Main PID: 43563 (mysqld)
      Status: "Server is operational"
        Tasks: 38 (limit: 35863)
       Memory: 450.8M
          CPU: 1.405s
      CGroup: /system.slice/mysqld.service
              └─43563 /usr/libexec/mysqld --basedir=/usr
```

Sqoop 예제

- MySQL 패스워드 설정
- 초기 mysql 접속 시 패스워드 없이 접속 가능
- 혹은 임시 패스워드 확인 가능 `sudo grep 'temporary password' /var/log/mysqld.log`

`mysql -u root -p`

패스워드 설정 (대소문자, 숫자, 특수기호, 8자리 이상 들어가야함.)

`ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'Tibero1!';`

```
[taba@localhost ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.36 Source distribution

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'Tibero';
Query OK, 0 rows affected (0.05 sec)
```


Sqoop 예제

- MySQL Test 데이터베이스 다운 및 압축풀기

wget https://github.com/datacharmer/test_db/releases/download/v1.0.7/test_db-1.0.7.tar.gz

tar -xvzf test_db-1.0.7.tar.gz

- MySQL 에 데이터 로드

cd test_db/

mysql -u root -p < employees.sql

```
hadoop@ubuntu-VirtualBox:~/test_db$ mysql -u root -p < employees.sql
Enter password:
INFO
CREATING DATABASE STRUCTURE
INFO
storage engine: InnoDB
INFO
LOADING departments
INFO
LOADING employees
INFO
LOADING dept_emp
INFO
LOADING dept_manager
INFO
LOADING titles
INFO
LOADING salaries
data_load_time_diff
00:01:18
hadoop@ubuntu-VirtualBox:~/test_db$ mysql -u root -p
```

Sqoop 예제

- 테스트 데이터베이스 조회하기

```
mysql -u root -p  
show databases;  
use employees;  
show tables;
```

```
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| employees |  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
+-----+  
5 rows in set (0.05 sec)
```

```
mysql> show tables;  
+-----+  
| Tables_in_employees |  
+-----+  
| current_dept_emp |  
| departments |  
| dept_emp |  
| dept_emp_latest_date |  
| dept_manager |  
| employees |  
| salaries |  
| titles |  
+-----+  
8 rows in set (0.01 sec)
```

Sqoop 예제

- sqoop으로 MySQL 특정 데이터베이스의 테이블 조회하기
sqoop list-tables --connect jdbc:mysql://localhost/employees \
--username root \
--password Tiberio1!
- sqoop으로 MySQL -> HDFS 데이터 보내기
sqoop import --connect jdbc:mysql://localhost/employees \
--username root --password Tiberio1! --table employees -m 1

Sqoop 예제

```
2024-03-26 03:41:51,677 INFO mapreduce.Job: Job job_1711361092141_0001 running in uber mode : false
2024-03-26 03:41:51,689 INFO mapreduce.Job: map 8% reduce 0%
2024-03-26 03:42:12,855 INFO mapreduce.Job: map 100% reduce 0%
2024-03-26 03:42:17,590 INFO mapreduce.Job: Job job_1711361092141_0001 completed successfully
2024-03-26 03:42:18,984 INFO mapreduce.Job: Counters: 33
  File System Counters
    FILE: Number of bytes read=0
    FILE: Number of bytes written=146178
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=87
    HDFS: Number of bytes written=13821993
    HDFS: Number of read operations=6
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
    HDFS: Number of bytes read erasure-coded=0
  Job Counters
    Launched map tasks=1
    Other local map tasks=1
    Total time spent by all maps in occupied slots (ms)=18891
    Total time spent by all reducers in occupied slots (ms)=0
    Total time spent by all map tasks (ms)=18891
    Total vcore-millisecnds taken by all map tasks=18801
    Total megabyte-millisecnds taken by all map tasks=19344184
  Map-Reduce Framework
    Map input records=300024
    Map output records=300024
    Input split bytes=87
    Spilled Records=0
    Failed Shuffles=0
    Merged Map outputs=0
    GC time elapsed (ms)=242
    CPU time spent (ms)=5229
    Physical memory (bytes) snapshot=139124736
    Virtual memory (bytes) snapshot=2494951424
    Total committed heap usage (bytes)=32571392
    Peak Map Physical memory (bytes)=139124736
    Peak Map Virtual memory (bytes)=2494951424
  File Input Format Counters
    Bytes Read=0
  File Output Format Counters
    Bytes Written=13821993
2024-03-26 03:42:19,959 INFO mapreduce.ImportJobBase: Transferred 13.1817 MB in 155.7499 seconds (86.6048 KB/sec)
2024-03-26 03:42:19,972 INFO mapreduce.ImportJobBase: Retrieved 300024 records.
```

Sqoop 예제

- 명령어로 하둡 데이터 조회하기
- `hdfs dfs -ls`

```
hdoop@ubuntu-VirtualBox:~$ hdfs dfs -ls
Found 1 items
drwxr-xr-x  - hdoop supergroup          0 2024-03-26 03:42 employees
```