Report

오픈소스를 활용한 3-Tier 환경 구축 #4주차

● 3 Tier 구축 (2) -

작성자	코더 – 정지호, 최예진, 김재현
검수자	송인섭 이사, 윤상훈 수석
작성일	2022-10-10

목차

CLOIT

1. 개요 3
2. WEB/WAS/DB 설치하기 4
1) 3-TIER 아키텍처구성 4
2) NGIINX 설치 11
3) JBOSS 설치 14
4) MYSQL 설치 22
5) CDB FOR MYSQL 설치 30
3. WEB/WAS/DB 연동 및 테스트 34
1) NGIINX - JBOSS 34
2) JBOSS -MYSQL 41

개요

CLOIT

1. web-was-db 설치하기

 1) 3-tier 아키텍처 구성

 2) nginx설치

 3) JBoss설치

 4) MySQL 설치

 5) CDB for MySQL을 이용한 설치

2. web-was-db 연동 및 테스트

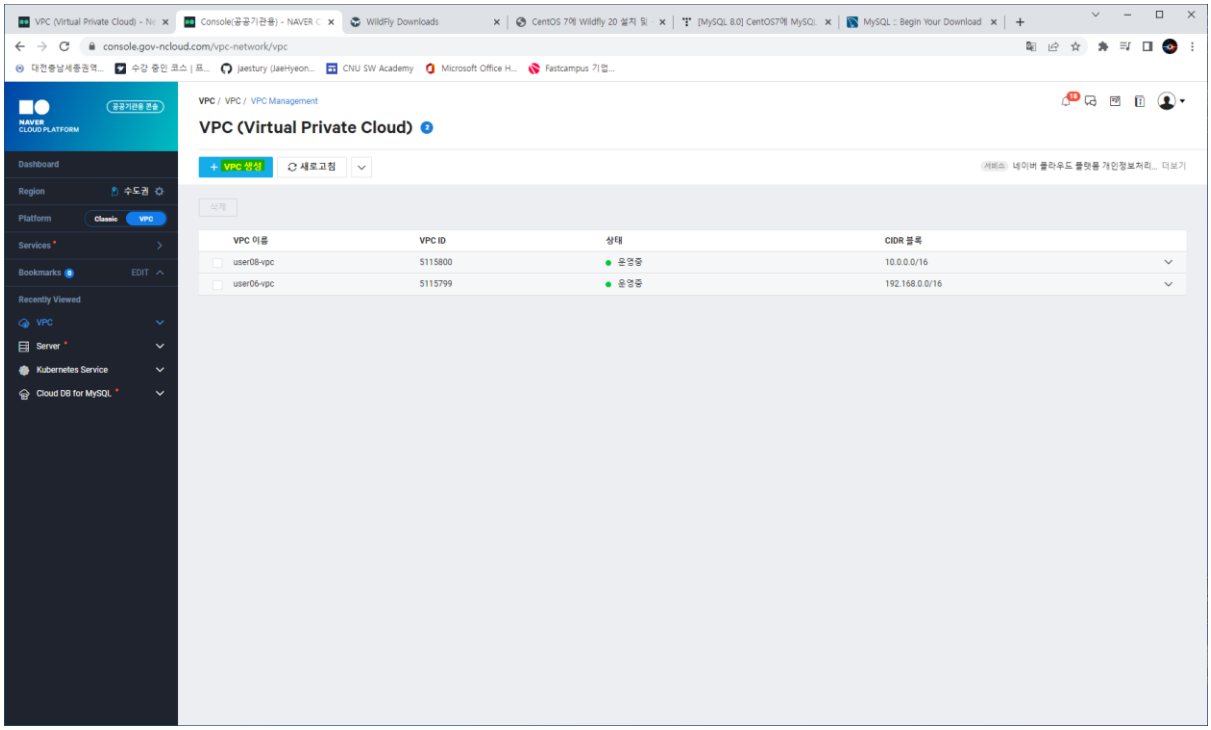
 1) nginx-JBoss

 2) JBoss- MySQL

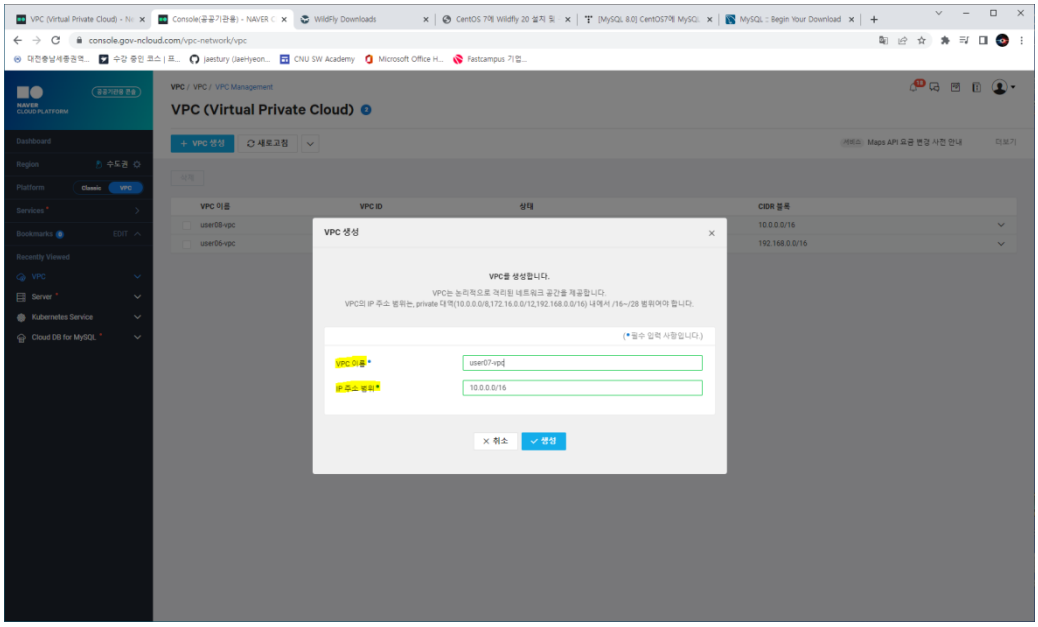
1.web-was-db설치

1) 3-tier 아키텍쳐 만들기

1. VPC 만들기

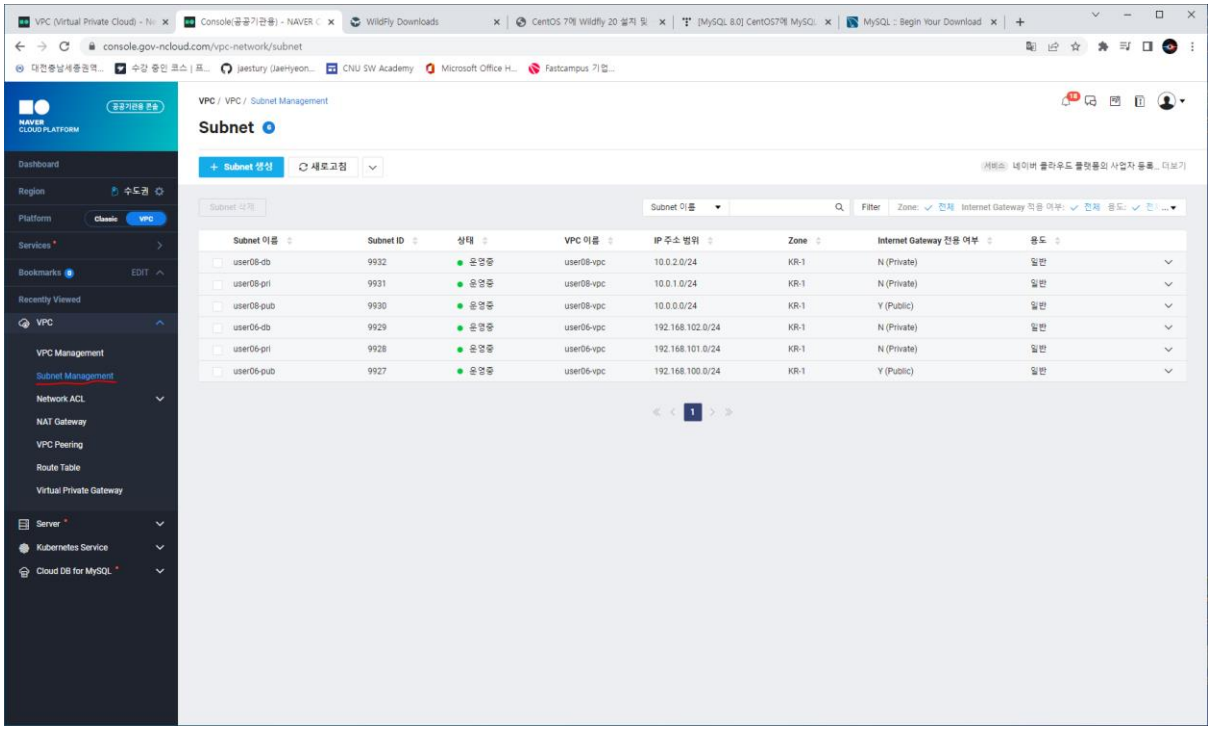


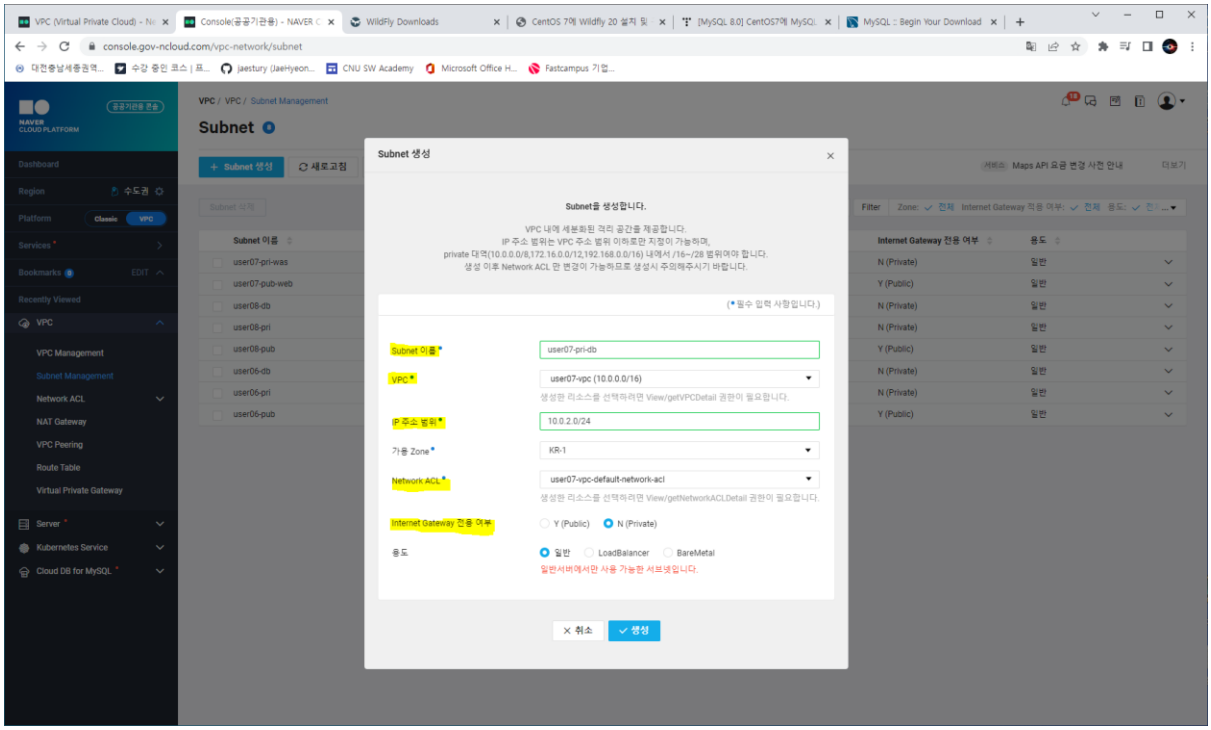
● VPC를 만들어보자. Service >Networking >VPC



● 이름과 IP 주소범위를 입력해주자. 범위는 10.0.0.0/16으로 해준다.

2. Subnet 쪼개기



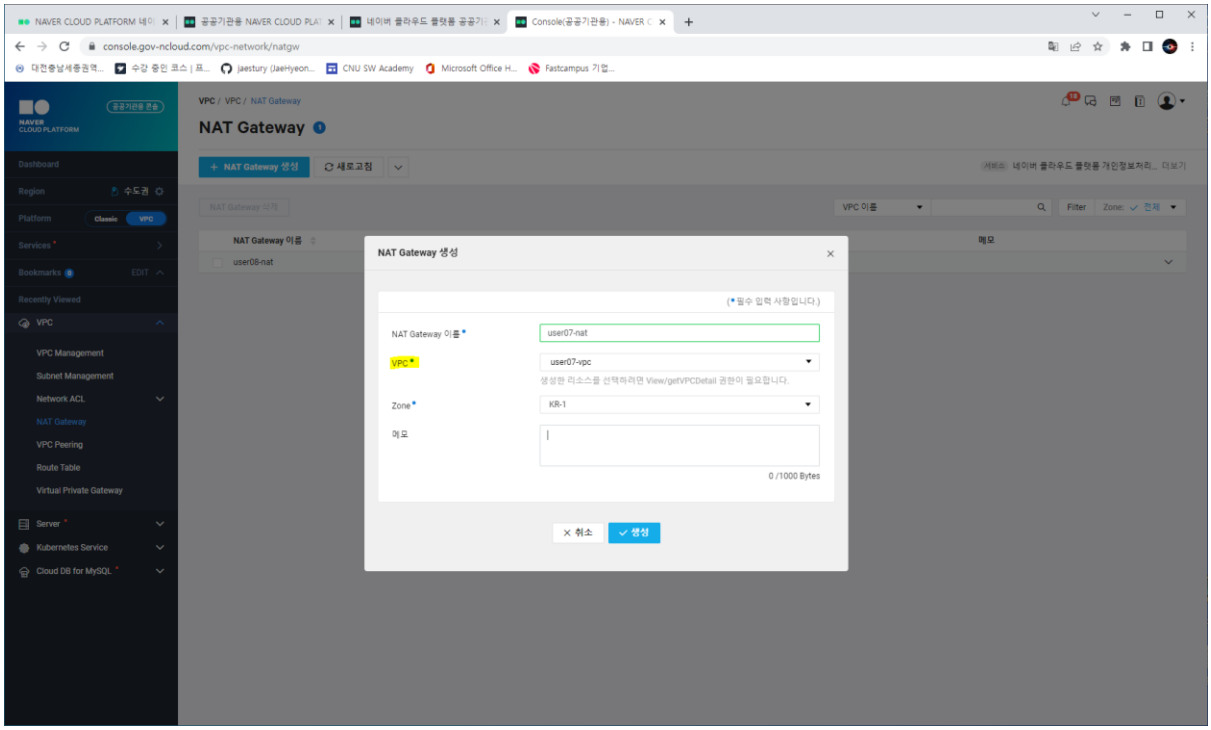


- Subnet 이름 : 잘 구분할 수 있게 만들어준다.
- VPC : 아까 만들어두었던 VPC 적용

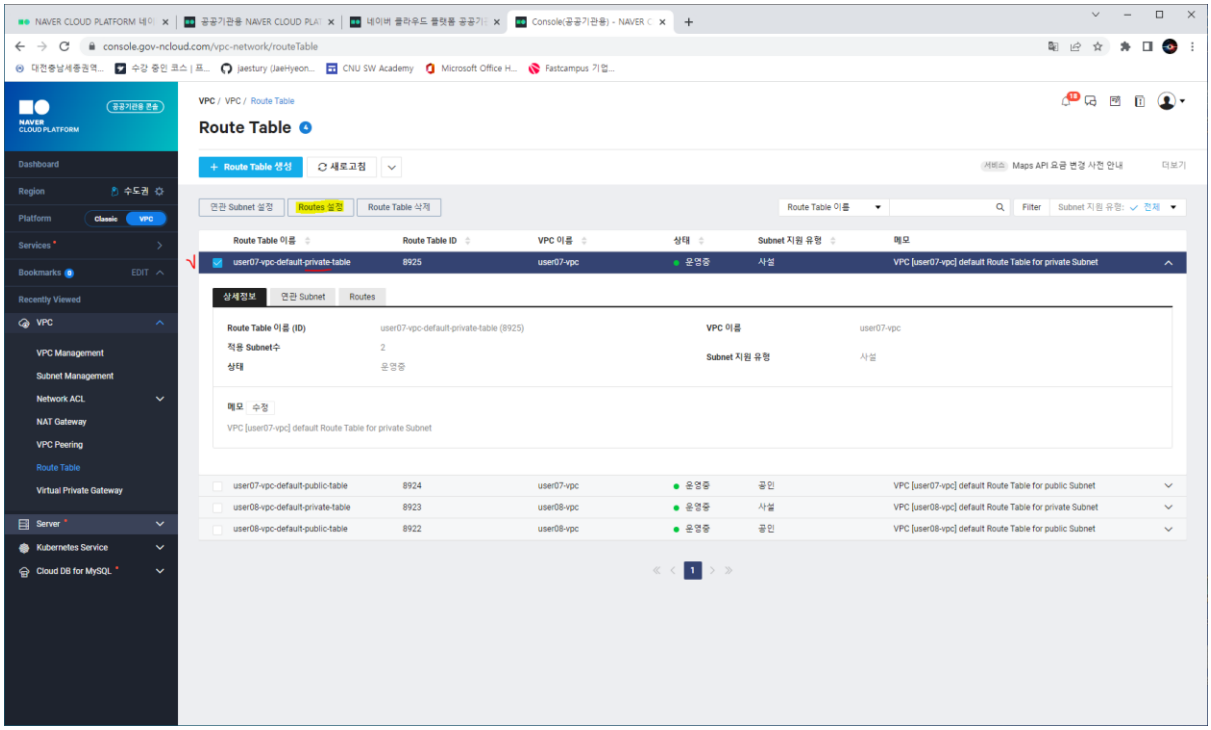
- IP 주소 범위 : 10.0.x.0/24로 설정해준다.public인 web은 10.0.0.0/24로 private인 was와 db는 10.0.1.0/24 | 10.0.2.0/24로 설정해주었다.
- Network ACL : 디폴트로 한다.
- IGW 전용여부 : public으로 사용할 web은 Y로, 나머지 둘은 N으로 설정해준다.

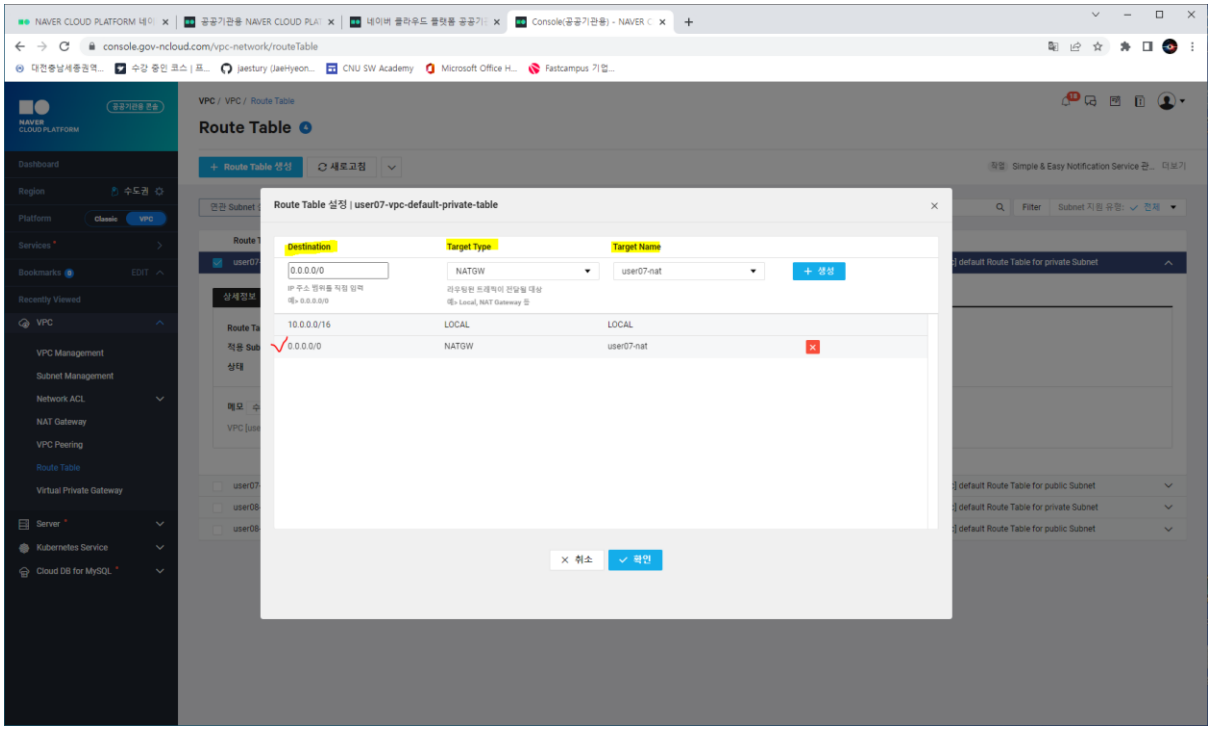
3. NAT Gateway와 Route Table 설정

- private subnet에서 이뤄지는 yum 프로그램 설치 작업을 위해 NAT Gateway 생성 및 Route Table 설정을 한다.



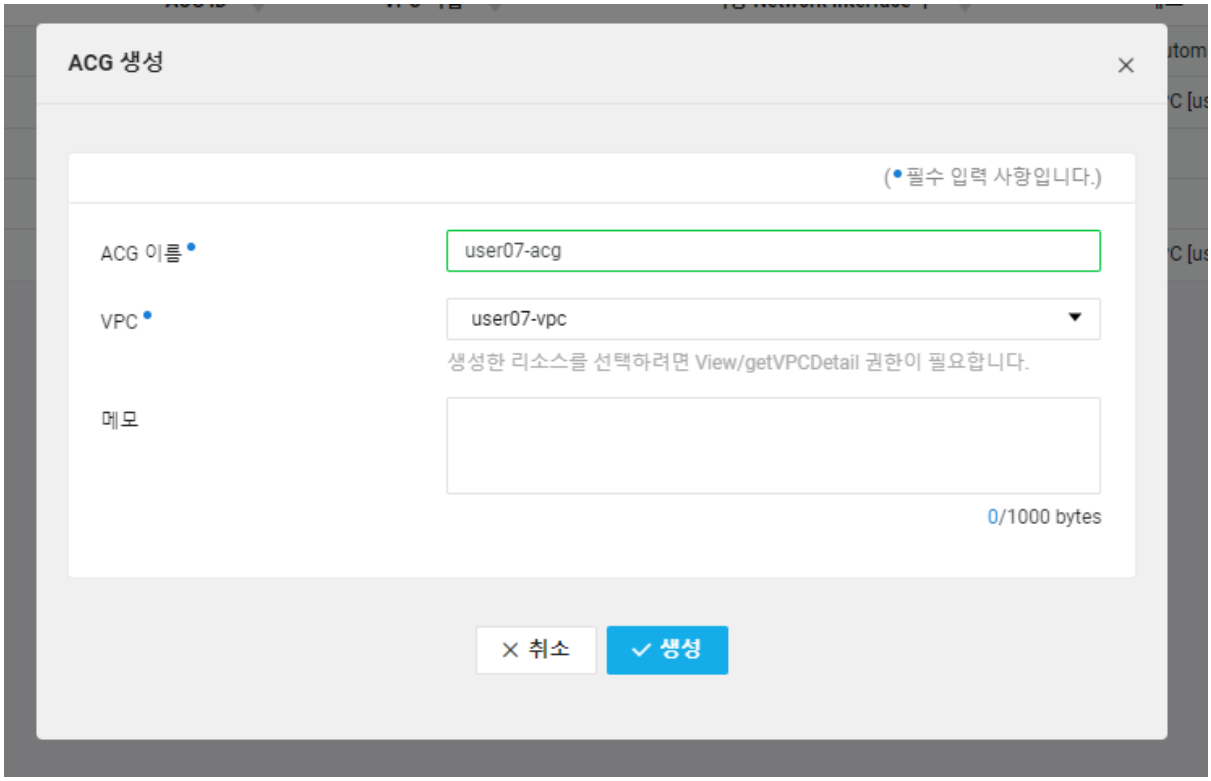
- VPC 설정만 주의해주자.
- Route Table 같은 경우, public subnet을 만져줄 필요가 없다.private subnet만 신경써주자.

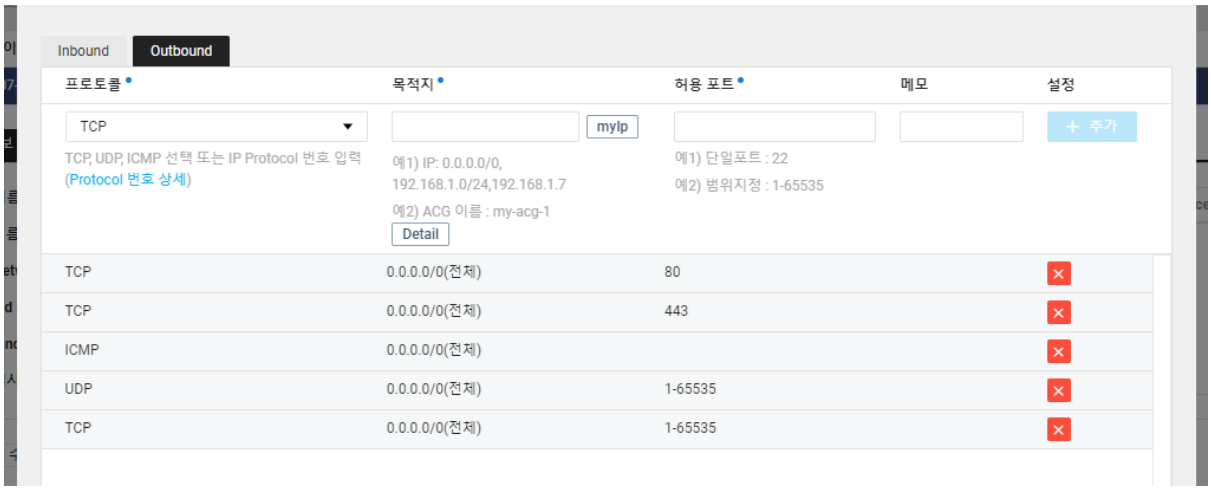




4. ACG 설정

- Server >ACG 로 이동해 ACG를 설정해주자.





● 실제 환경에서는 보안에 신경 써야하기 때문에 Inbound/Outbound를 신경 써서 설정해 주어야 하지만, 실습해보는 것이기 때문에 0.0.0.0/0으로 통일하도록 한다.

● Inbound의 경우 SSH 접속을 위한 22 포트와, Ping 테스트를 위한 ICMP를 설정한다.

● Outbound에는 HTTPS를 위한 443포트와 HTTP를 위한 80번 포트를 추가해준다. 둘다 TCP.

5. 서버 생성

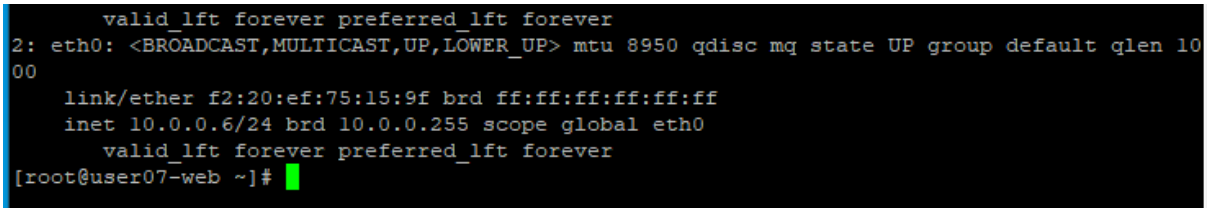
● 각 서브넷에 서버를 할당해준다. 각각 Web 서버, WAS, DB가 자리할 서버.

서버 이름	서버 이미지 이름	서버 구성	상태	비공인 IP	공인 IP	VPC	Subnet
user07-db	centos-7.8-64	[Standard] 2vCPU, 8GB Mem [g2]	생성중	10.0.2.6		user07-vpc	user07-db
user07-was	centos-7.8-64	[High-CPU] 2vCPU, 4GB Mem [g2]	생성중	10.0.1.6		user07-vpc	user07-was
user07-web	centos-7.8-64	[High-CPU] 2vCPU, 4GB Mem [g2]	부팅중	10.0.0.6		user07-vpc	user07-web

6. 접속 확인

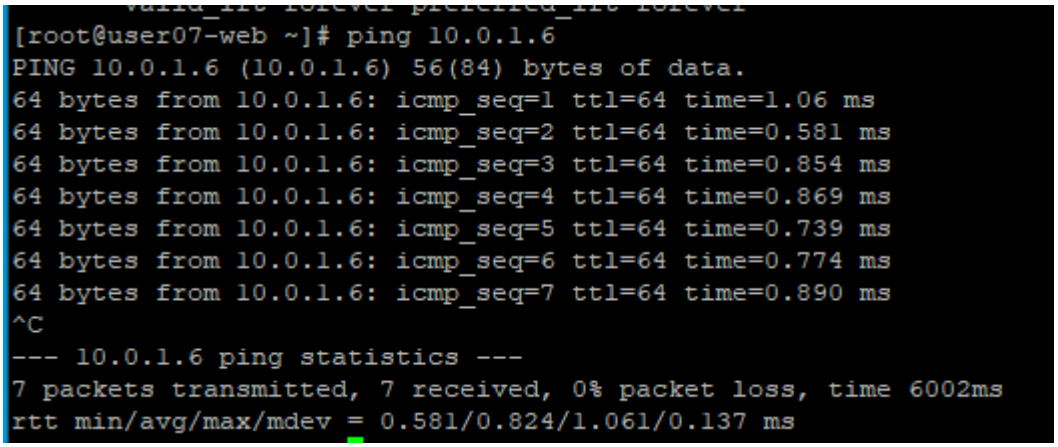
● PuTTY를 이용해서 웹 서버에 접속해준다.

● 다음 ip a를 입력하면 내 IP를 확인할 수 있다.



```
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 8950 qdisc mq state UP group default qlen 10
00
    link/ether f2:20:ef:75:15:9f brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.6/24 brd 10.0.0.255 scope global eth0
       valid_lft forever preferred_lft forever
[root@user07-web ~]#
```

● 서버들이 잘 연결되어 있는지 확인해보자.ping (비공인ip)를 통해 통신요청을 보낼 수 있다.



```
[root@user07-web ~]# ping 10.0.1.6
PING 10.0.1.6 (10.0.1.6) 56(84) bytes of data.
64 bytes from 10.0.1.6: icmp_seq=1 ttl=64 time=1.06 ms
64 bytes from 10.0.1.6: icmp_seq=2 ttl=64 time=0.581 ms
64 bytes from 10.0.1.6: icmp_seq=3 ttl=64 time=0.854 ms
64 bytes from 10.0.1.6: icmp_seq=4 ttl=64 time=0.869 ms
64 bytes from 10.0.1.6: icmp_seq=5 ttl=64 time=0.739 ms
64 bytes from 10.0.1.6: icmp_seq=6 ttl=64 time=0.774 ms
64 bytes from 10.0.1.6: icmp_seq=7 ttl=64 time=0.890 ms
^C
--- 10.0.1.6 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6002ms
rtt min/avg/max/mdev = 0.581/0.824/1.061/0.137 ms
```

```
[root@user07-web ~]# ping 10.0.2.6
PING 10.0.2.6 (10.0.2.6) 56(84) bytes of data.
64 bytes from 10.0.2.6: icmp_seq=1 ttl=64 time=1.39 ms
64 bytes from 10.0.2.6: icmp_seq=2 ttl=64 time=0.851 ms
64 bytes from 10.0.2.6: icmp_seq=3 ttl=64 time=0.757 ms
64 bytes from 10.0.2.6: icmp_seq=4 ttl=64 time=1.05 ms
64 bytes from 10.0.2.6: icmp_seq=5 ttl=64 time=1.12 ms
64 bytes from 10.0.2.6: icmp_seq=6 ttl=64 time=0.926 ms
^C
--- 10.0.2.6 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5005ms
rtt min/avg/max/mdev = 0.757/1.017/1.397/0.208 ms
[root@user07-web ~]#
```

● 두 서버, was용 서버와 db용 서버 모두 잘 연결되어 있는 것을 확인할 수 있다

.

● 다음은 private subnet을 사용하고 있는 각 서버에서 외부로 통신이 되고 있는지 확인해 보자.ssh (관리자 이름)@(할당 IP)를 이용해 다른 서버로 건너갈 수 있다.

```
[root@user07-web ~]# ssh ncloud@10.0.1.6
The authenticity of host '10.0.1.6 (10.0.1.6)' can't be established.
ECDSA key fingerprint is SHA256:qfg4xJX7ob/rPdoTpxcJ7xswTqfvzphNzcHalVFTqG8.
ECDSA key fingerprint is MD5:ca:ae:1e:f5:92:9f:c0:f1:67:28:12:32:9d:6d:8e:f8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.1.6' (ECDSA) to the list of known hosts.
ncloud@10.0.1.6's password:
Permission denied, please try again.
ncloud@10.0.1.6's password:
[ncloud@user07-was ~]$ sudo -i
[root@user07-was ~]#
```

● was 서버로 넘어왔다.

```
    link/ether f2:20:ef:75:99:48 brd ff:ff:ff:ff:ff:ff
    inet 10.0.1.6/24 brd 10.0.1.255 scope global eth0
       valid_lft forever preferred_lft forever
[root@user07-was ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=112 time=33.7 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=112 time=33.3 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=112 time=33.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=112 time=33.3 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=112 time=33.4 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 33.355/33.466/33.777/0.227 ms
[root@user07-was ~]#
```

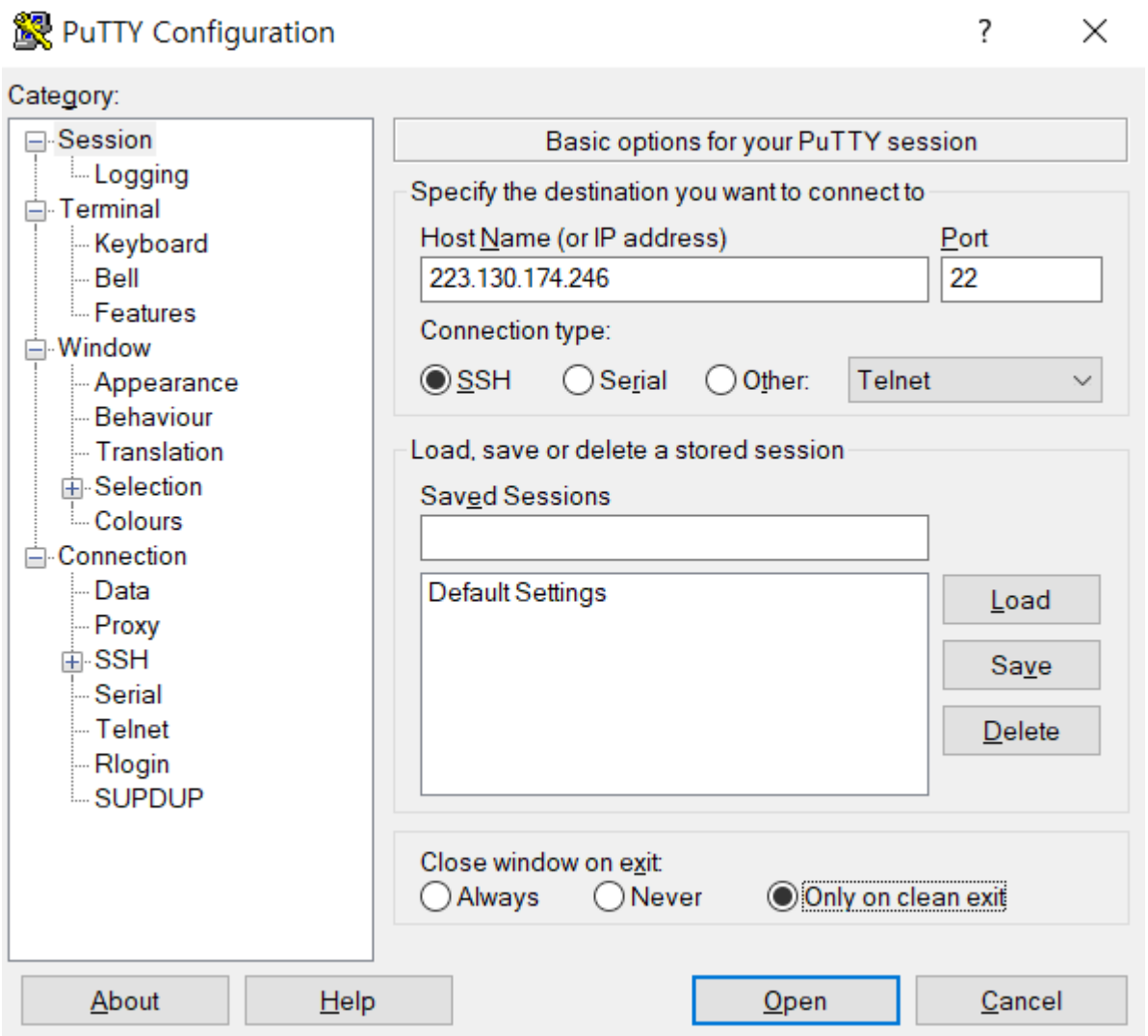
● ip a명령어를 사용해서 ip를 확인해보고, ping명령어를 사용해서 외부와 연결되고 있는지 확인해 보았다. NAT Gateway를 설정해 놓았기 때문에 잘 통신하는 모습이다.

● 이것으로 3-tier 구축은 완료되었다. 각 서버에 web-was-db를 다운로드해보자.

2) nginx 설치

1.Nginx 설치하기

- putty 접속



● Package 관리 도구인 yum에 Nginx 저장소를 추가

:#v /etc/yum.repos.d/nginx.repo

:[nginx]name=nginx repo

baseurl=http://nginx.org/packages/centos/7/$basearch/

gpgcheck=0

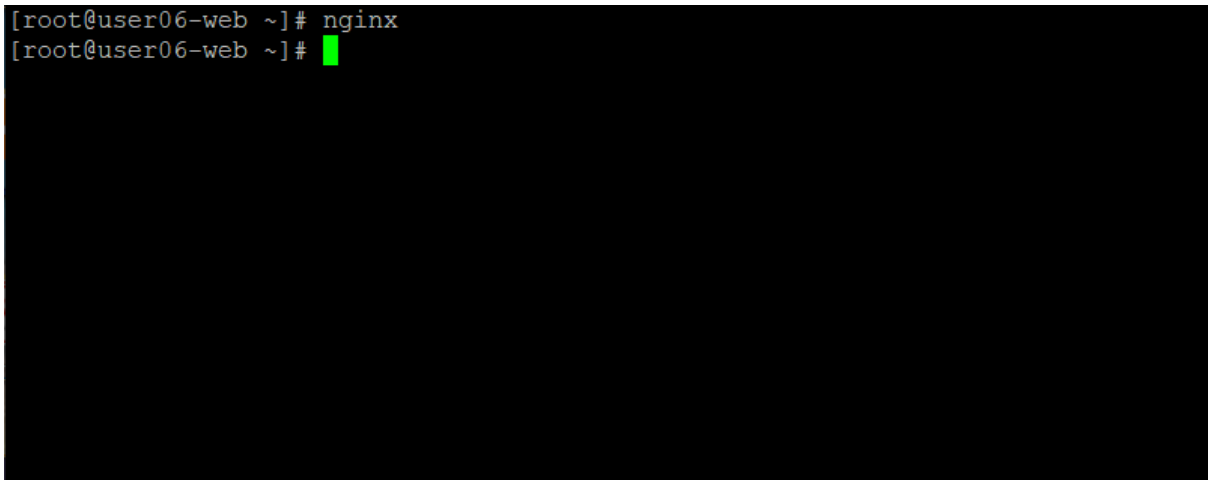
enabled=1

```
[root@user06-web ~]# vi /etc/yum.repos.d/nginx.repo
[root@user06-web ~]# cat /etc/yum.repos.d/nginx.repo
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/centos/7/$basearch/
gpgcheck=0
enabled=1


[root@user06-web ~]#
```

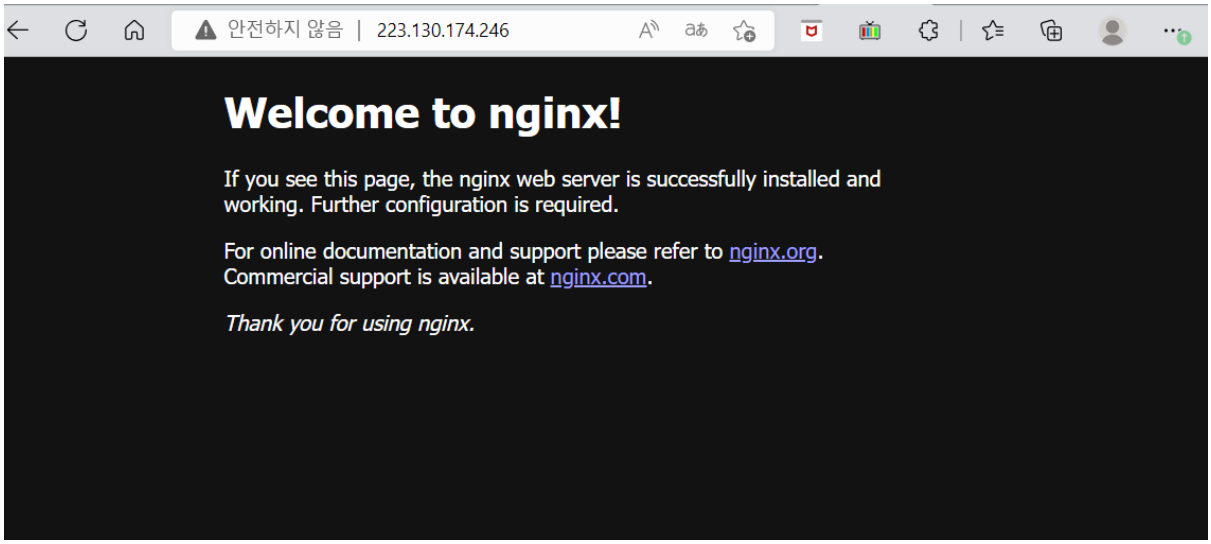
● Nginx를 설치: yum install nginx –y

```
[root@user06-web ~]# yum install nginx -y
Loaded plugins: fastestmirror, langpacks
Determining fastest mirrors
base                                                     | 3.6 kB     00:00
extras                                                   | 2.9 kB     00:00
nginx                                                    | 2.9 kB     00:00
update                                                   | 2.9 kB     00:00
(1/5): base/7/x86_64/group_gz                              | 153 kB   00:00
(2/5): extras/7/x86_64/primary_db                          | 243 kB   00:00
(3/5): base/7/x86_64/primary_db                            | 6.1 MB   00:00
(4/5): update/7/x86_64/primary_db                          |  12 MB   00:00
(5/5): nginx/x86_64/primary_db                             |  76 kB   00:01
Resolving Dependencies
--> Running transaction check
---> Package nginx.x86_64 1:1.22.0-1.el7.ngx will be installed
--> Processing Dependency: libpcre2-8.so.0()(64bit) for package: 1:nginx-1.22.0-
1.el7.ngx.x86_64
```

● Nginx 서비스를 시작: nginx



2. 정상 동작으로 확인



Cloud Search 서버에 할당한 공인 IP를 웹 브라우저에 입력한 후, 실행했을 때 테스트 페이지가 조회되면 정상 동작으로 확인

3)JBoss 설치

1. was 서버로 이동한다. ('ssh 사용자계정@원격지 ip' 명령어 사용!)

```
[root@user07-web yum.repos.d]# ssh ncloud@10.0.1.6
ncloud@10.0.1.6's password:
[ncloud@user07-was ~]$
```

(여기서 사용자 계정은 ncloud / 원격지 ip는 was의 ip 10.0.1.6이다)

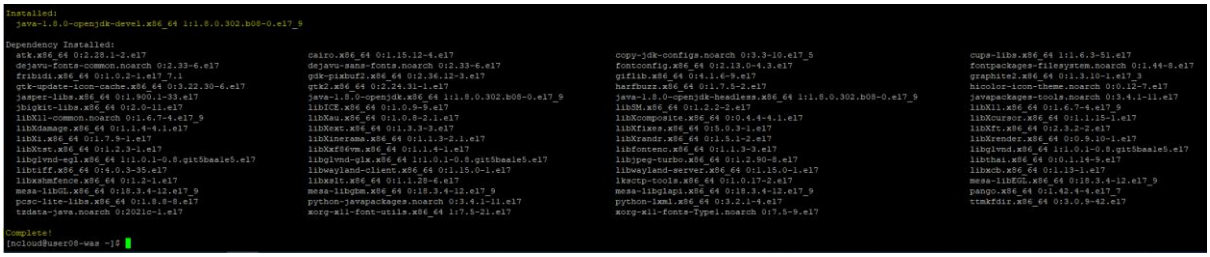
```
ncloud@10.0.1.6's password:
[ncloud@user08-was ~]$
```

(비밀번호를 입력해 로그인해준다.)

2. Java Open JDK를 설치해준다.

- Java 플랫폼의 오픈소스 구현인 OpenJDK를 설치해준다.

```
[ncloud@user08-was ~]$ sudo yum install java-1.8.0-openjdk-devel
Loaded plugins: fastestmirror, langpacks
Determining fastest mirrors
base                                                     | 3.6 kB     00:00
extras                                                   | 2.9 kB     00:00
update                                                   | 2.9 kB     00:00
(1/4): base/7/x86_64/group_gz                              | 153 kB   00:00
(2/4): extras/7/x86_64/primary_db                          | 243 kB   00:00
(3/4): base/7/x86_64/primary_db                            | 6.1 MB   00:00
(4/4): update/7/x86_64/primary_db                          |  12 MB   00:00
Resolving Dependencies
--> Running transaction check
---> Package java-1.8.0-openjdk-devel.x86_64 1:1.8.0.302.b08-0.el7_9 will be ins
talled
--> Processing Dependency: java-1.8.0-openjdk(x86-64) = 1:1.8.0.302.b08-0.el7_9
for package: 1:java-1.8.0-openjdk-devel-1.8.0.302.b08-0.el7_9.x86_64
--> Processing Dependency: libjvm.so()(64bit) for package: 1:java-1.8.0-openjdk-
devel-1.8.0.302.b08-0.el7_9.x86_64
```



(installed, complete 를 확인할 수 있다.)

3. 사용자 생성해주기

- WildFly를 루트 사용자로 실행하는 것은 보안상 위험하다. 그러므로 wildfly라는 새 시스템 사용자 및 그룹을 만들어주자.

```
[root@user07-was ~]#
[root@user07-was ~]# groupadd -r wildfly
[root@user07-was ~]# useradd -r -g wildfly -d /opt/wildfly -s /sbin/nologin wildfly
[root@user07-was ~]#
[root@user07-was ~]# grep wildfly /etc/passwd
wildfly:x:996:994::/opt/wildfly:/sbin/nologin
```

groupadd : 리눅스 시스템 내에 새 그룹을 추가하기 위한 명령어. 여러 사용자들을 하나의 그룹으로 구분하기 위해 사용한다.

groupadd -r (~~)는 시스템용 그룹(GID 499 이하)을 생성하는 명령어이다. 0~499 까지 할당되어 있지 않은 GID 중 가장 높은 번호를 할당해준다.

useradd : 신규 사용자를 추가하는 명령어.
-g 는 그룹을 지정할 때 사용. -d 는 홈 디렉터리 지정. -s 는 사용자 생성 시 사용자가 사용할 셸을 지정한다.

4. 와일드 플라이 설치

```
[root@user07-was ~]# wget https://download.jboss.org/wildfly/24.0.1.Final/wildfly-24.0.1.Fi
nal.tar.gz
--2022-10-10 16:07:40--  https://download.jboss.org/wildfly/24.0.1.Final/wildfly-24.0.1.Fin
al.tar.gz
Resolving download.jboss.org (download.jboss.org)... 23.200.75.74, 23.200.75.93, 2600:1417:
e::48f6:675b, ...
Connecting to download.jboss.org (download.jboss.org)|23.200.75.74|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 210680621 (201M) [application/x-gzip]
Saving to: 'wildfly-24.0.1.Final.tar.gz'

100%[=============================================>] 210,680,621 1.67MB/s   in 1m 43s

2022-10-10 16:09:24 (1.95 MB/s) - 'wildfly-24.0.1.Final.tar.gz' saved [210680621/210680621]
```

- 설치파일을 다운받고 나서 압축을 풀어주자. 그리고 /opt 디렉토리로 이동해준다.

```
[ncloud@user08-was ~]$ sudo tar xf /tmp/wildfly-$WILDFLY_VERSION.tar.gz -C /opt/
[ncloud@user08-was ~]$
```

- 그 다음 링크파일을 생성해준다. 링크는 wildfly 설치 디렉토리를 가리킨다.

```
[ncloud@user08-was ~]$ sudo ln -s /opt/wildfly-$WILDFLY_VERSION /opt/wildfly
[ncloud@user08-was ~]$
```

5. 디렉토리 소유권을 와일드 플라이로 변경

```
[ncloud@user08-was ~]$ sudo tar xf /tmp/wildfly-$WILDFLY_VERSION.tar.gz -C /opt/
[ncloud@user08-was ~]$ sudo ln -s /opt/wildfly-$WILDFLY_VERSION /opt/wildfly
[ncloud@user08-was ~]$ sudo chown -RH wildfly: /opt/wildfly
[ncloud@user08-was ~]$
```

(wildfly는 wildfly 사용자 아래에서 실행된다.)

(chown 명령어는 change와 owner를 조합한 명령어로, 파일 소유권과 그룹을 변경한다.
`chown (OPTION) (OWNER) (:GROUP) FILE` 순으로 사용.)

6. 와일드 플라이 구성 파일을 저장할 디렉토리를 생성한다. 그리고 구성 파일을 생성한 디렉토리에 복사한다.

```
[ncloud@user08-was ~]$ sudo mkdir -p /etc/wildfly
[ncloud@user08-was ~]$
```

```
[ncloud@user08-was ~]$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /etc/wildfly/
[ncloud@user08-was ~]$
```

-wildfly.conf 파일을 사용하면 wildfly 모드와 바인드 주소를 지정할 수 있다. 이는 필요에 따라 편집이 가능하다.

`vi /etc/wildfly/wildfly.conf`

```
# The configuration you want to run
WILDFLY_CONFIG=standalone.xml

# The mode you want to run
WILDFLY_MODE=standalone

# The address to bind to
WILDFLY_BIND=0.0.0.0
~
```

7. WildFly launch.sh 스크립트를 /opt/wildfly/bin/ 디렉토리에 복사한다.

```
[root@user07-was wildfly]#
[root@user07-was wildfly]# cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/wild
fly/bin/
[root@user07-was wildfly]# cd /opt/wildfly
```

```
-rwxr-xr-x 1 wildfly wildfly    2172 Jul 27  2021 jdr.sh
-rw-r--r-- 1 wildfly wildfly   50086 Jul 27  2021 launcher.jar
-rwxr-xr-x 1 root    root        217 Oct 10 16:28 launch.sh
-rw-r--r-- 1 wildfly wildfly      49 Jul 27  2021 product.conf
-rw-r--r-- 1 wildfly wildfly   11028 Jul 27  2021 standalone.bat
-rw-r--r-- 1 wildfly wildfly    3330 Jul 27  2021 standalone.conf
-rw-r--r-- 1 wildfly wildfly    3847 Jul 27  2021 standalone.conf.bat
-rw-r--r-- 1 wildfly wildfly    3640 Jul 27  2021 standalone.conf.ps1
-rw-r--r-- 1 wildfly wildfly    1951 Jul 27  2021 standalone.ps1
-rwxr-xr-x 1 wildfly wildfly   13003 Jul 27  2021 standalone.sh
```

8. bin 디렉토리의 *.sh 실행권한을 추가한다. (sh -c 옵션은 문자열에 대한 명령을 읽는다.)

```
[root@user08-was ~]# sudo sh -c 'chmod +x /opt/wildfly/bin/*.sh'
[root@user08-was ~]#
```

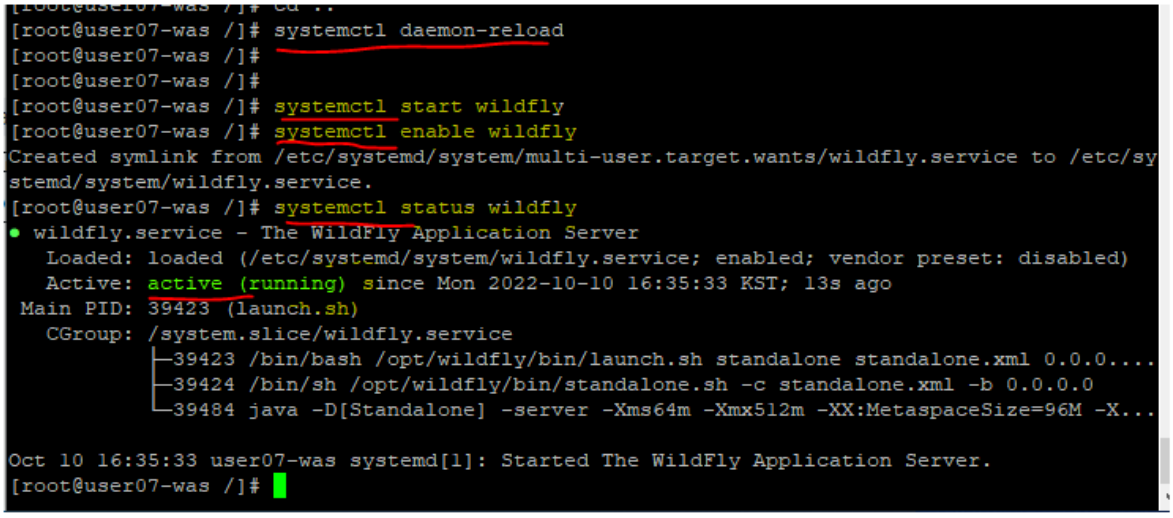
9. 이름이 지정된 system 유닛파일을 /etc/system/ 디렉토리에 복사한다.

```
[root@user07-was bin]# cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service /etc/sy
stemd/system/
[root@user07-was bin]# cd /etc/systemd/system
[root@user07-was system]# ls -l
total 20
drwxr-xr-x. 2 root root   57 Jun 13 18:15 basic.target.wants
lrwxrwxrwx. 1 root root   37 Jun 13 18:19 default.target -> /lib/systemd/system/multi-user.
target
drwxr-xr-x. 2 root root   87 Jun 13 18:15 default.target.wants
drwxr-xr-x. 2 root root   32 Jun 13 18:15 getty.target.wants
drwxr-xr-x. 2 root root   35 Jun 13 18:15 local-fs.target.wants
drwxr-xr-x. 2 root root 4096 Jun 13 18:37 multi-user.target.wants
-rw-rw-r--  1 root root  198 Jun 13 18:37 nsemodule.service
-rw-r--r--. 1 root root  236 Jun 13 18:30 nsight-agent.service
drwxr-xr-x. 2 root root   51 Jun 13 18:31 sockets.target.wants
drwxr-xr-x. 2 root root 4096 Jun 13 18:31 sysinit.target.wants
drwxr-xr-x. 2 root root   44 Jun 13 18:15 system-update.target.wants
-rw-r--r--  1 root root  409 Oct 10 16:32 wildfly.service
[root@user07-was system]#
```

10. 시스템에 새 장치 파일을 생성했음을 알린다.

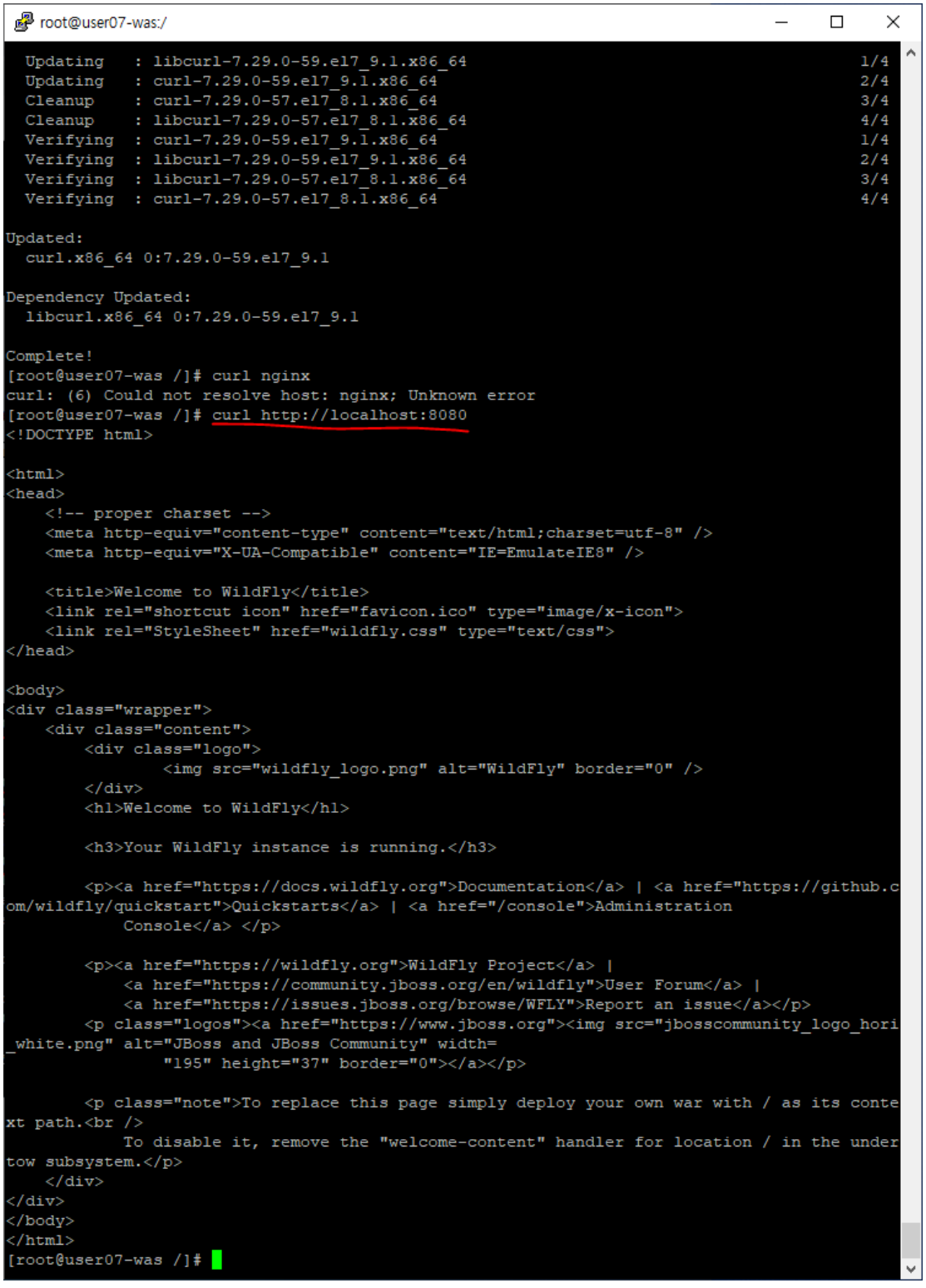
```
[root@user08-was ~]# sudo systemctl daemon-reload
[root@user08-was ~]#
```

11. wildfly 서비스를 시작하고 다음 부팅 시에도 자동으로 실행되게 설정한다. 서비스 실행 중인지도 확인해보자



12. 잘 설치되어 있는지 확인해보자.

- curl http://local host:8080 명령어를 입력해본다.



(잘 설치되어 있다는 것을 확인할 수 있다.)

(공인 IP를 URL로 입력하여 nginx가 잘 설치되었다는 것을 확인 할 수 있었지만, WAS, 그러니까 wildfly는 private subnet에 위치하고 있으므로 nginx처럼 확인이 불가능하다. 만약 nginx처럼 확인이 된다면 이것은 보안상의 취약점으로 작용할 것이다.

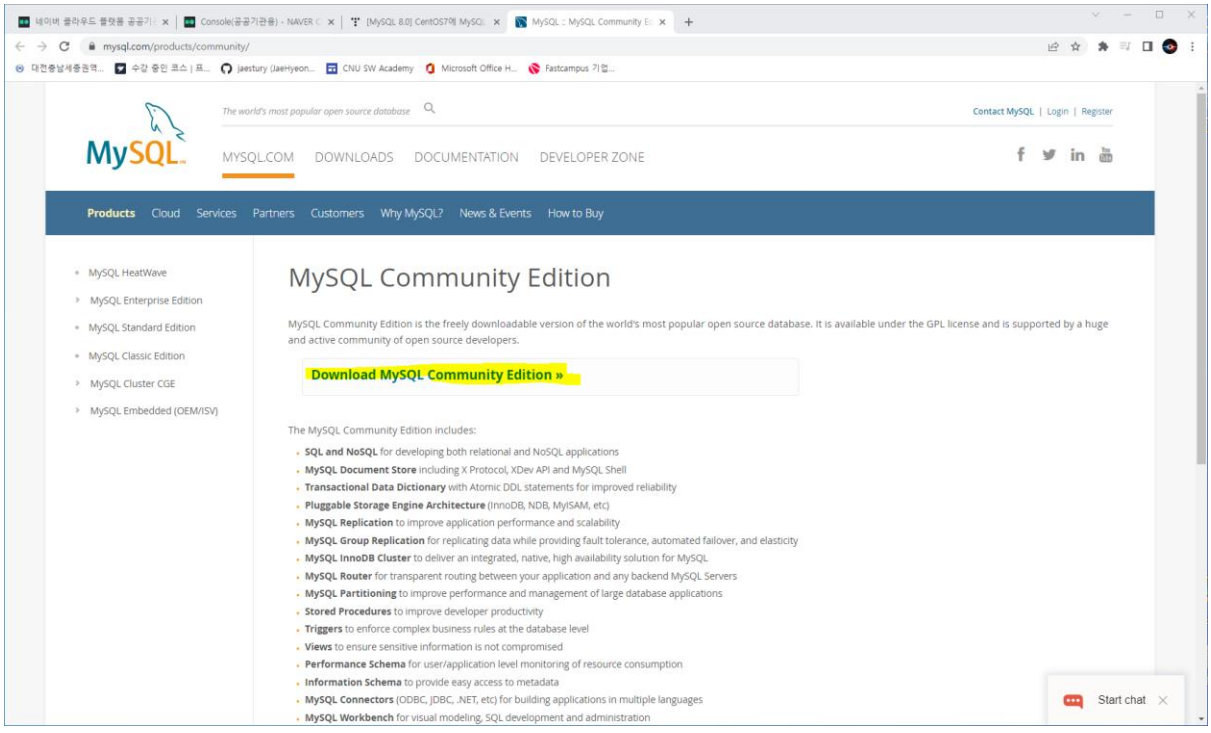
4) MySQL 설치

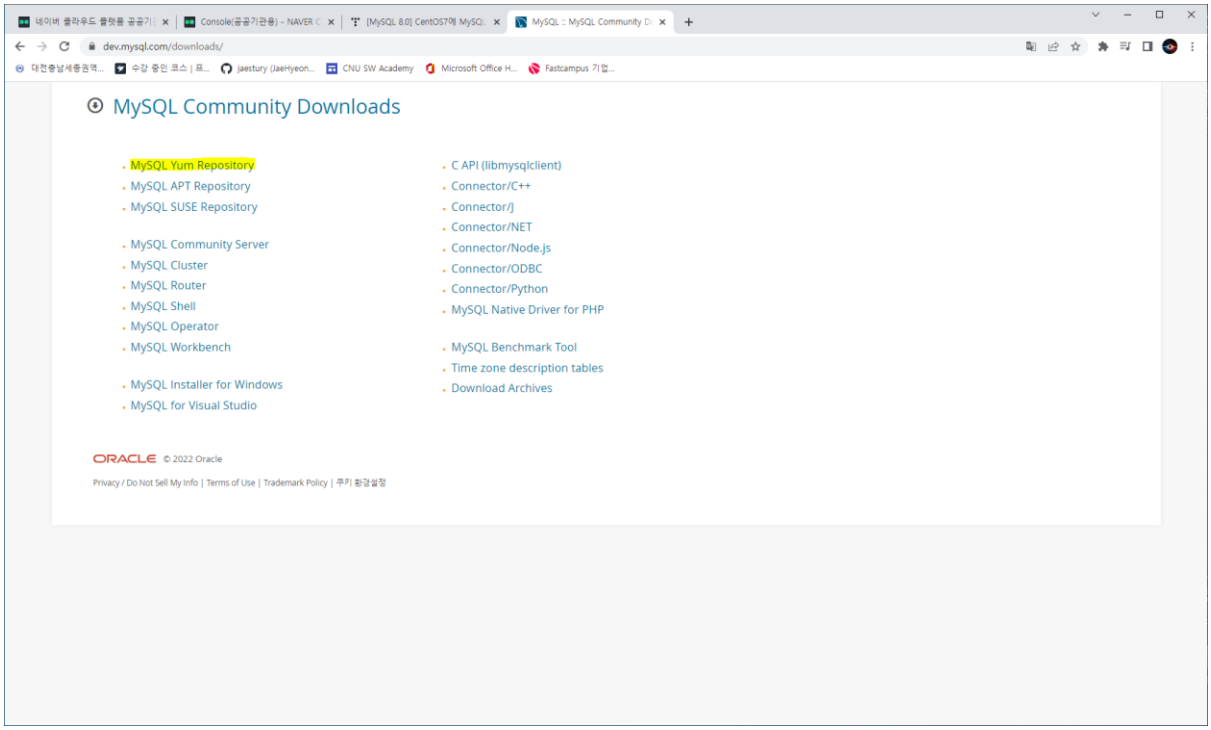
● DB 서버에 접속해주자. MySQL 8.0 버전을 다운로드 받는 것이 목표이다.

```
</html>
[root@user07-was /]# ssh ncloud@10.0.2.6
The authenticity of host '10.0.2.6 (10.0.2.6)' can't be established.
ECDSA key fingerprint is SHA256:qfg4xJX7ob/rPdoTpxcJ7xswTqfvzphNzcHalVFTqG8.
ECDSA key fingerprint is MD5:ca:ae:1e:f5:92:9f:c0:f1:67:28:12:32:9d:6d:8e:f8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.2.6' (ECDSA) to the list of known hosts.
ncloud@10.0.2.6's password:
[ncloud@user07-db ~]$ sudo -i
[root@user07-db ~]#
[root@user07-db ~]#
```

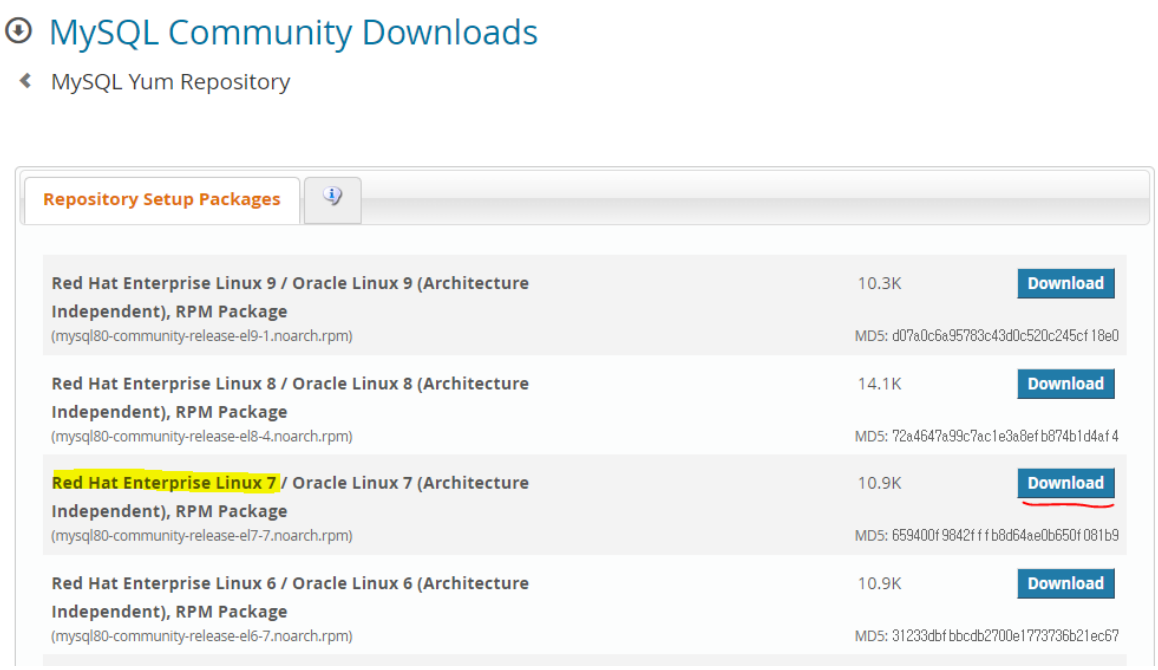
1. MySQL 설치다운로드 링크 확인

● MySQL Community 버전으로 다운받기 위해서는 링크 확인이 필요하다.





- CentOS에서는 Yum을 사용 가능하니, Yum을 사용하자.



- CentOS7니까 7버전 선택.

- No thanks, just start my download를 오른쪽 클릭해서 링크 주소를 복사해주자.

2. MySQL Repository 설치

- PuTTY에서 다음 명령어를 실행해준다

: yum install -y (아까 복사해둔 링크)

yum install -y https://dev.mysql.com/get/mysql80-community-release-el7-7.noarch.rpm

```
[root@user07-db ~]# yum install -y https://dev.mysql.com/get/mysql80-community-release-el7-
7.noarch.rpm
Loaded plugins: fastestmirror, langpacks
mysql80-community-release-el7-7.noarch.rpm                          |  11 kB  00:00:00
Examining /var/tmp/yum-root-jUSvDX/mysql80-community-release-el7-7.noarch.rpm: mysql80-comm
unity-release-el7-7.noarch
Marking /var/tmp/yum-root-jUSvDX/mysql80-community-release-el7-7.noarch.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package mysql80-community-release.noarch 0:el7-7 will be installed
--> Finished Dependency Resolution
base/7/x86_64                                                       | 3.6 kB  00:00:00
base/7/x86_64/group_gz                                              | 153 kB  00:00:00
base/7/x86_64/primary_db                                            | 6.1 MB  00:00:00
extras/7/x86_64                                                     | 2.9 kB  00:00:00
extras/7/x86_64/primary_db                                          | 243 kB  00:00:00
update/7/x86_64                                                     | 2.9 kB  00:00:00
update/7/x86_64/primary_db                                          |  12 MB  00:00:00

Dependencies Resolved

================================================================================
 Package                   Arch    Version Repository                        Size
================================================================================
Installing:
 mysql80-community-release  noarch  el7-7   /mysql80-community-release-el7-7.noarch   10 k

Transaction Summary
================================================================================
Install  1 Package

Total size: 10 k
Installed size: 10 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : mysql80-community-release-el7-7.noarch                       1/1
  Verifying  : mysql80-community-release-el7-7.noarch                       1/1

Installed:
  mysql80-community-release.noarch 0:el7-7

Complete!
```

● MySQL을 설치할 수 있는 레포지토리가 설치되었다. yum repolist 명령어로 방금 설치된 mysql 레포지토리 목록을 확인할 수 있다.

```
[root@user07-db ~]# yum repolist enabled | grep "mysql.*"
mysql-connectors-community/x86_64       MySQL Connectors Community           199
mysql-tools-community/x86_64            MySQL Tools Community                 92
mysql80-community/x86_64                MySQL 8.0 Community Server           346
[root@user07-db ~]#
```

● yum search mysql을 사용해서 설치 가능한 mysql 패키지 목록도 확인할 수 있다.

```
[root@user07-db ~]#
[root@user07-db ~]# yum search mysql
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
================================== N/S matched: mysql ==================================
MySQL-python.x86_64 : An interface to MySQL
akonadi-mysql.x86_64 : Akonadi MySQL backend support
apr-util-mysql.x86_64 : APR utility library MySQL DBD driver
dovecot-mysql.x86_64 : MySQL back end for dovecot
freeradius-mysql.x86_64 : MySQL support for freeradius
libdbi-dbd-mysql.x86_64 : MySQL plugin for libdbi
mysql-community-client.i686 : MySQL database client applications and tools
mysql-community-client.x86_64 : MySQL database client applications and tools
mysql-community-client-plugins.i686 : Shared plugins for MySQL client applications
mysql-community-client-plugins.x86_64 : Shared plugins for MySQL client applications
mysql-community-common.i686 : MySQL database common files for server and client libs
mysql-community-common.x86_64 : MySQL database common files for server and client libs
mysql-community-devel.i686 : Development header files and libraries for MySQL database
                           : client applications
mysql-community-devel.x86_64 : Development header files and libraries for MySQL database
                             : client applications
mysql-community-embedded-compat.i686 : MySQL embedded compat library
mysql-community-embedded-compat.x86_64 : MySQL embedded compat library
mysql-community-icu-data-files.i686 : MySQL packaging of ICU data files
mysql-community-icu-data-files.x86_64 : MySQL packaging of ICU data files
mysql-community-libs.i686 : Shared libraries for MySQL database client applications
```

3. 진짜로 설치

- yum install -y mysql-server

```
[root@user07-db ~]#
[root@user07-db ~]# yum install -y mysql-server
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package mysql-community-server.x86_64 0:8.0.30-1.el7 will be installed
--> Processing Dependency: mysql-community-common(x86-64) = 8.0.30-1.el7 for package: mysql
```

```
Dependencies Resolved

================================================================================
 Package                    Arch      Version            Repository          Size
================================================================================
Installing:
 mysql-community-libs       x86_64    8.0.30-1.el7       mysql80-community   1.5 M
     replacing  mariadb-libs.x86_64 1:5.5.65-1.el7
 mysql-community-libs-compat x86_64   8.0.30-1.el7       mysql80-community   670 k
     replacing  mariadb-libs.x86_64 1:5.5.65-1.el7
 mysql-community-server     x86_64    8.0.30-1.el7       mysql80-community    54 M
Installing for dependencies:
 mysql-community-client     x86_64    8.0.30-1.el7       mysql80-community    14 M
 mysql-community-client-plugins x86_64 8.0.30-1.el7       mysql80-community   2.5 M
 mysql-community-common     x86_64    8.0.30-1.el7       mysql80-community   645 k
 mysql-community-icu-data-files x86_64 8.0.30-1.el7       mysql80-community   2.1 M

Transaction Summary
================================================================================
Install  3 Packages (+4 Dependent packages)

Total download size: 75 M
Downloading packages:
warning: /var/cache/yum/x86_64/7/mysql80-community/packages/mysql-community-client-plugins-
8.0.30-1.el7.x86_64.rpm: Header V4 RSA/SHA256 Signature, key ID 3a79bd29: NOKEY
Public key for mysql-community-client-plugins-8.0.30-1.el7.x86_64.rpm is not installed
(1/7): mysql-community-client-plugins-8.0.30-1.el7.x86_64.rpm       | 2.5 MB  00:00:00
(2/7): mysql-community-common-8.0.30-1.el7.x86_64.rpm               | 645 kB  00:00:00
(3/7): mysql-community-icu-data-files-8.0.30-1.el7.x86_64.rpm       | 2.1 MB  00:00:00
(4/7): mysql-community-client-8.0.30-1.el7.x86_64.rpm               |  14 MB  00:00:00
(5/7): mysql-community-libs-8.0.30-1.el7.x86_64.rpm                 | 1.5 MB  00:00:00
(6/7): mysql-community-libs-compat-8.0.30-1.el7.x86_64.rpm          | 670 kB  00:00:00
(7/7): mysql-community-server-8.0.30-1.el7.x86_64.rpm               |  54 MB  00:00:00
--------------------------------------------------------------------------------
Total                                              58 MB/s |  75 MB  00:00:01
```

● mysqld -V / mysql -version으로 잘 설치되었는지 확인하자.

```
[root@user07-db ~]# mysql --version
mysql  Ver 8.0.30 for Linux on x86_64 (MySQL Community Server - GPL)
[root@user07-db ~]#
[root@user07-db ~]#
[root@user07-db ~]# mysqld -V
/usr/sbin/mysqld  Ver 8.0.30 for Linux on x86_64 (MySQL Community Server - GPL)
[root@user07-db ~]#
```

4. MySQL 서버 시작 및 접속

● 서버를 시작해보자.

: systemctl start mysqld

: systemctl enable mysqld

: systemctl status mysqld

```
[root@user07-db ~]# systemctl start mysqld
[root@user07-db ~]# systemctl enable mysqld
[root@user07-db ~]# systemctl status mysqld
● mysqld.service - MySQL Server
   Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled
)
   Active: active (running) since Mon 2022-10-10 18:05:00 KST; 18s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
 Main PID: 41965 (mysqld)
   Status: "Server is operational"
   CGroup: /system.slice/mysqld.service
           └─41965 /usr/sbin/mysqld

Oct 10 18:04:55 user07-db systemd[1]: Starting MySQL Server...
Oct 10 18:05:00 user07-db systemd[1]: Started MySQL Server.
[root@user07-db ~]#
```

● 서버 설치과정에서 임시 비밀번호가 생성되며, 이 비밀번호로 접속이 가능하다.grep 'temporary password' /var/log/mysqld.log로 확인해보자.

```
[root@user07-db ~]# grep 'temporary password' /var/log/mysqld.log
2022-10-10T09:04:57.333541Z 6 [Note] [MY-010454] [Server] A temporary password is generated
 for root@localhost: CBHfngZq9s+y
[root@user07-db ~]#
```

● 접속을 위해선 저 비밀번호가 필요하다. 이제 접속해보자

: .mysql -u root -p

```
[root@user07-db ~]# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.30

Copyright (c) 2000, 2022, 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>
```

5. Root 계정 비밀번호 변경

```
mysql> show databases;
ERROR 1820 (HY000): You must reset your password using ALTER USER statement before executin
g this statement.
mysql>
```

● database 조회를 좀 하려니까 오류가 생겨버린다.

"ERROR 1820 (HY000): You must reset your password using ALTER USER statement before executing this statement."

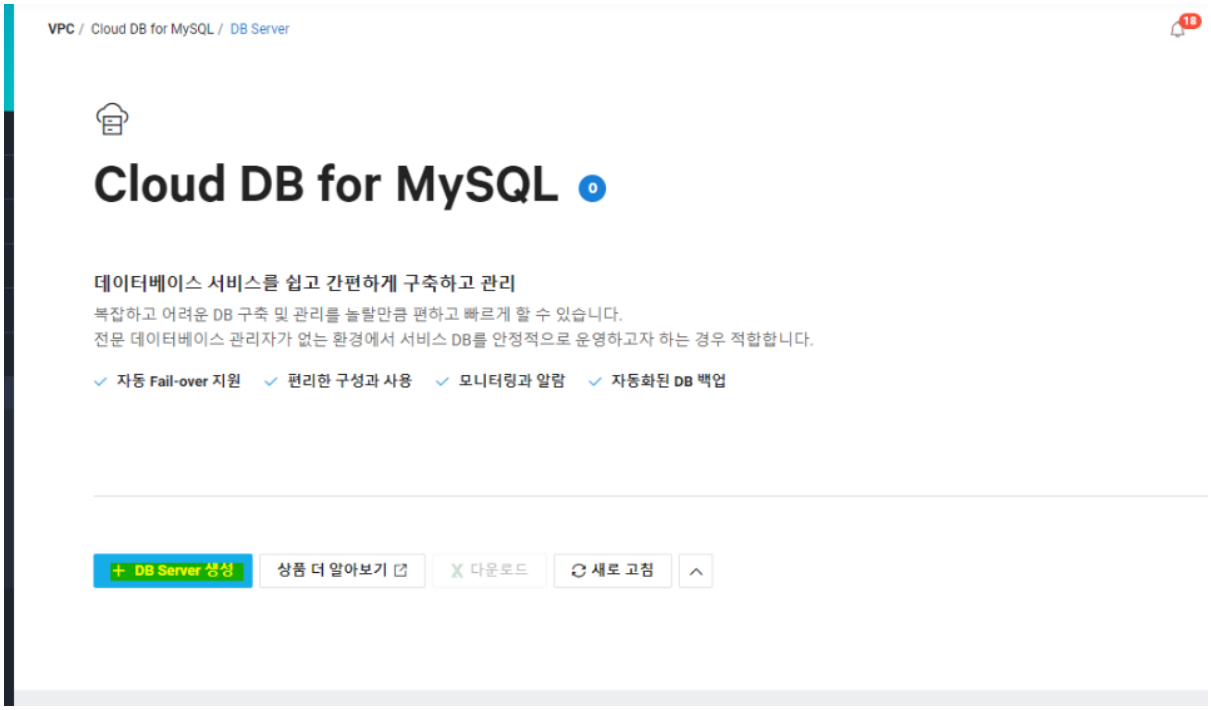
● 비밀번호 바꾸라는 말이다. root 유저의 비밀번호를 변경해주자.

```
mysql>
mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'Wogusgus123123#@!';
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

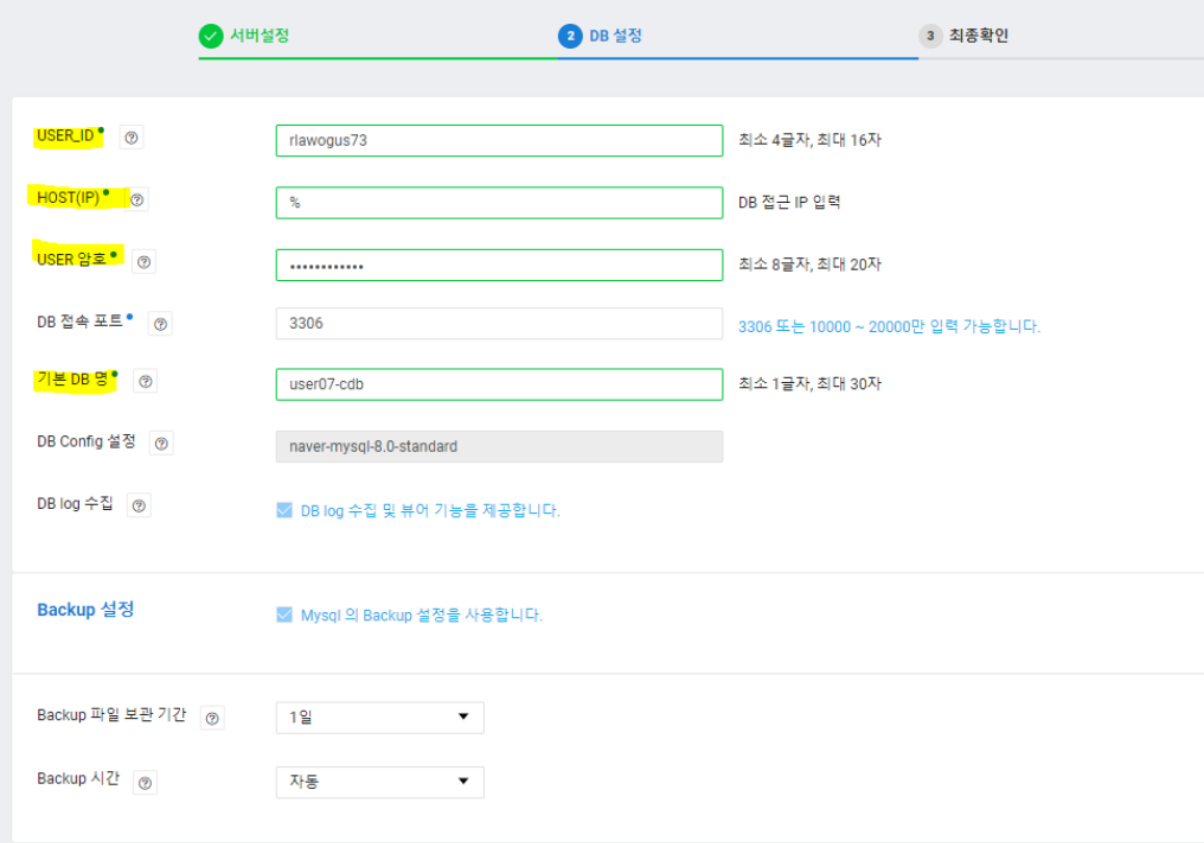
● 비밀번호를 바꾸면 잘 된다. 굿굿! root 비밀번호를 잘 저장해 두도록 하자.

5) CDB for MySQL을 이용한 설치

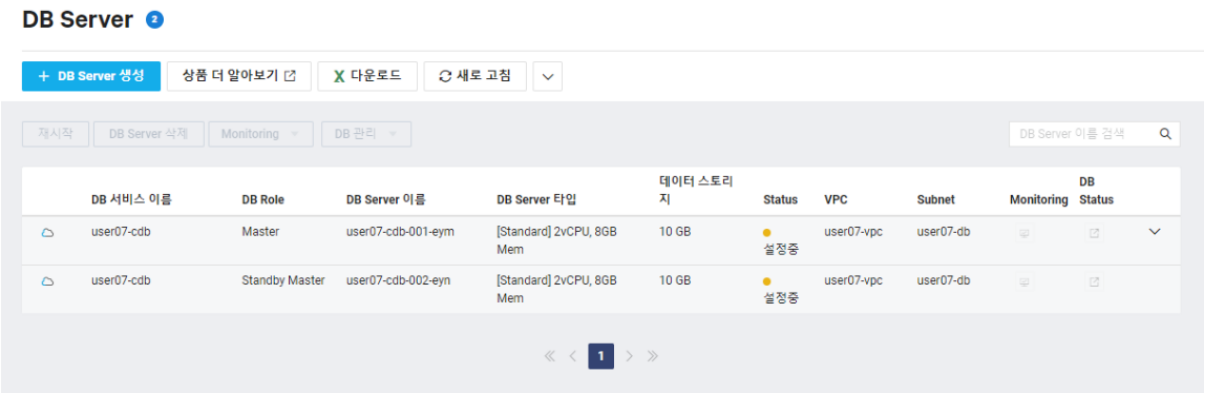


- DB 서버를 생성한다.

- VPC는 앞에 만들어 놓았던 user07로, Subnet도 user07의 DB용 서버에 연결해준다.
- 서버 이름과 서비스 이름은 적당히 설정해준다.



- 실제 DB 접속 정보와 DB를 구성하는 정보를 입력하는 페이지로,
- User_ID는 DB에 접속할 때 사용하는 ID
- Host(IP)는 DB에 접속 허용할 IP를 정하는 옵션이다. 여기서는 모든 IP를 허용하기 위해 '%'를 입력했다.
- 실제에서는 지양해야 하는 방법이다. 보안을 위해서 WAS IP 대역을 주는 것을 권장하지만, 실습 환경이니 이렇게 사용한다.
- USER 암호는 접속 암호다. 반드시 기억하자.
- 기본 DB명은 최초 생성할 DB이름이다.



- 기본적으로 서버가 두 개 생성된 것을 볼 수 있다.
- Master와 Standby Master로 구성되는데, 마스터는 기본 DB, Standby Master는 보조 DB로 장애 발생 시 자동으로 Standby Master로 교체된다고 한다.
- 서비스 자체에서 이중화와 fail over 기능을 제공하는 셈.

2. Web-Was-Db 연동 및 테스트

CLOIT

1) Web(Nginx)-Was(JBoss)

```
[root@user07-web ~]# curl http://10.0.1.6:8080
<!DOCTYPE html>

<html>
<head>
    <!-- proper charset -->
    <meta http-equiv="content-type" content="text/html;charset=utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=EmulateIE8" />

    <title>Welcome to WildFly</title>
    <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
    <link rel="StyleSheet" href="wildfly.css" type="text/css">
</head>

<body>
<div class="wrapper">
    <div class="content">
        <div class="logo">
                <img src="wildfly_logo.png" alt="WildFly" border="0" />
        </div>
        <h1>Welcome to WildFly</h1>

        <h3>Your WildFly instance is running.</h3>

        <p><a href="https://docs.wildfly.org">Documentation</a> | <a href="https://github.c
om/wildfly/quickstart">Quickstarts</a> | <a href="/console">Administration
            Console</a> </p>

        <p><a href="https://wildfly.org">WildFly Project</a> |
            <a href="https://community.jboss.org/en/wildfly">User Forum</a> |
            <a href="https://issues.jboss.org/browse/WFLY">Report an issue</a></p>
        <p class="logos"><a href="https://www.jboss.org"><img src="jbosscommunity_logo_hori
_white.png" alt="JBoss and JBoss Community" width=
                "195" height="37" border="0"></a></p>

        <p class="note">To replace this page simply deploy your own war with / as its conte
xt path.<br />
            To disable it, remove the "welcome-content" handler for location / in the under
tow subsystem.</p>
    </div>
</div>
</body>
</html>
```

- Curl http://10.0.1.6:8080 명령어를 사용해서 nginx 서버에서 JBoss에게 요청을 주고, HTML 파일을 받을 수 있다.
- 잘 연동된 것을 확인할 수 있다.

2. 테스트

- Web 서버에서 웹 띄워 보기

– 준비 : git, node, npm 다운받기.

1) Node.js 설치

- epel 저장소를 확인하고, 없을 경우 설치 받는다.

```
[root@s183c48ca3be /]# yum install epel-release
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
---> Package epel-release.noarch 0:7-11 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                Arch                Version              Repository          Size
================================================================================
Installing:
 epel-release           noarch              7-11                 extras              15 k

Transaction Summary
================================================================================
Install  1 Package
```

```
[root@s183c48ca3be /]# yum repolist
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
epel/x86_64/metalink                                            | 7.5 kB  00:00:00
 * epel: mirror.01link.hk
https://mirror.01link.hk/epel/7/x86_64/repodata/repomd.xml: [Errno 14] curl#60 - "Peer's Certificate
has expired."
Trying other mirror.
It was impossible to connect to the CentOS servers.
This could mean a connectivity issue in your environment, such as the requirement to configure a prox
y,
or a transparent proxy that tampers with TLS security, or an incorrect system clock.
You can try to solve this issue by using the instructions on https://wiki.centos.org/yum-errors
If above article doesn't help to resolve this issue please use https://bugs.centos.org/.

epel                                                            | 4.7 kB  00:00:00
(1/3): epel/x86_64/group_gz                                     |  97 kB  00:00:00
(2/3): epel/x86_64/updateinfo                                   | 1.0 MB  00:00:00
(3/3): epel/x86_64/primary_db                                   | 7.0 MB  00:00:02
repo id                      repo name                                      status
base/7/x86_64                CentOS-7 - Base                                10,072
epel/x86_64                  Extra Packages for Enterprise Linux 7 - x86_64 13,733+1
extras/7/x86_64              CentOS-7 - Extras                                 500
nginx/x86_64                 nginx repo                                        282
update/7/x86_64              CentOS-7 - Updates                              2,902
repolist: 27,489
[root@s183c48ca3be /]#
```

- node.js 설치

```
[root@s183c48ca3be /]# yum -y install nodejs
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * epel: ftp.riken.jp
Resolving Dependencies
--> Running transaction check
---> Package nodejs.x86_64 1:16.17.0-1.el7 will be installed
--> Processing Dependency: nodejs-libs(x86-64) = 1:16.17.0-1.el7 for package: 1:nodejs-16.17.0-1.el7.
x86_64
```

2) npm 설치

```
[root@s183c48ca3be /]# yum install -y npm
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * epel: ftp.iij.ad.jp
Resolving Dependencies
--> Running transaction check
---> Package npm.x86_64 1:8.15.0-1.16.17.0.1.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package          Arch              Version                        Repository          Size
================================================================================
Installing:
 npm              x86_64            1:8.15.0-1.16.17.0.1.el7       epel               2.1 M

Transaction Summary
================================================================================
Install  1 Package
```

- npm –v 로 설치확인 할 수 있다.

3) git에서 이전에 작성해 놓은 프로그램을 다운받는다.

- git clone (git URL)

```
[root@s183c48ca3be /]# git clone https://github.com/jaestury/react-springboot-rest-api
Cloning into 'react-springboot-rest-api'...
remote: Enumerating objects: 322, done.
remote: Counting objects: 100% (133/133), done.
remote: Compressing objects: 100% (81/81), done.
remote: Total 322 (delta 51), reused 76 (delta 42), pack-reused 189
Receiving objects: 100% (322/322), 401.03 KiB | 0 bytes/s, done.
Resolving deltas: 100% (77/77), done.
```

- git에서 내려 받은 폴더들

```
drwxr-xr-x 5 root root 131 Oct 10 19:01 1
drwxr-xr-x 5 root root  97 Oct 10 18:57 react-springboot-rest-api
```

```
drwxr-xr-x 4 root root   90 Oct 10 18:57 gc-coffee
drwxr-xr-x 6 root root  144 Oct 10 18:58 kdt-react-order-ui
-rw-r--r-- 1 root root  713 Oct 10 18:57 README.md
-rw-r--r-- 1 root root 4800 Oct 10 18:57 sample.html
```

- gc-coffee는 스프링, kdt-react-order-ui가 리액트다.

```
drwxr-xr-x 802 root root   24576 Oct 10 18:58 node_modules
-rw-r--r--   1 root root     845 Oct 10 18:57 package.json
-rw-r--r--   1 root root 1186358 Oct 10 18:58 package-lock.json
drwxr-xr-x   2 root root     120 Oct 10 19:24 public
-rw-r--r--   1 root root    3359 Oct 10 18:57 README.md
drwxr-xr-x   3 root root     167 Oct 10 19:26 src
[root@user07-web kdt-react-order-ui]#
```

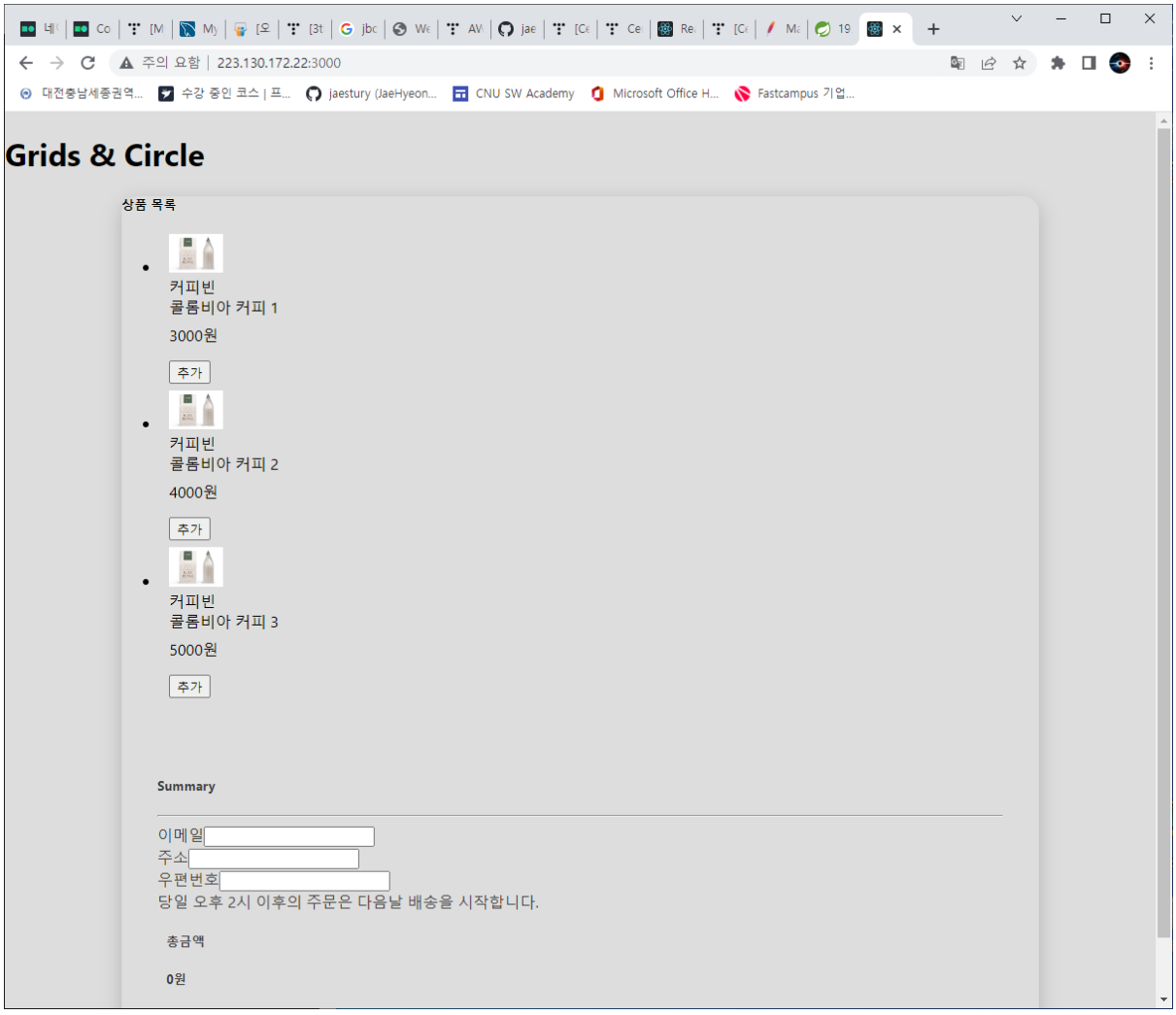
- npm start로 리액트를 시작해준다.

```
Failed to compile.

Module not found: Error: Can't resolve 'bootstrap/dist/css/bootstrap.css' in '/react-springboot-rest-
api/kdt-react-order-ui/src'
ERROR in ./src/App.js 7:0-42
Module not found: Error: Can't resolve 'bootstrap/dist/css/bootstrap.css' in '/react-springboot-rest-
api/kdt-react-order-ui/src'

webpack compiled with 1 error
```

- 컴파일 오류가 발생했다.
- App.js 라는 파일에 문제가 발생해 오류가 생긴 것 같으니 수정하자.
-

```
Compiled successfully!

You can now view kdt-react-order-ui in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://10.0.0.6:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

- 리액트 컴파일 성공!
- 공인IP:3000 으로 접속해보려고 했지만, 안된다. 당연한 일이다. 3000포트를 뚫어 놓지 않았기 때문.
- ACG 설정으로 가서 포트 설정을 해준다.



- 공인IP:3000로 nginx 서버에서 출력된 모습
- 원래 Bootstrap을 사용해서 조금 더 깔끔한 모습으로 출력되는데, bootstrap이 끼면 컴파일 오류가 나기 때문에 빼줬다.
- 하지만 지금은 nginx에서 리액트를 직접 받아서 출력한 것이므로 nginx를 proxy로 사용해서 리액트를 출력해보고자 한다.

- vi /etc/nginx/conf.d/default.conf 의 설정을 조금 바꿔준다.

```
server {
    listen       80;
    server_name  localhost;

    #access_log  /var/log/nginx/host.access.log  main;

    location / {
        proxy_pass http://localhost:3000;
    }

    #error_page  404              /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root   /usr/share/nginx/html;
    }

    # proxy the PHP scripts to Apache listening on 127.0.0.1:80
    #
    #location ~ \.php$ {
    #    proxy_pass   http://127.0.0.1;
    #}

    # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
    #
    #location ~ \.php$ {
    #    root           html;
    #    fastcgi_pass   127.0.0.1:9000;
    #    fastcgi_index  index.php;
    #    fastcgi_param  SCRIPT_FILENAME  /scripts$fastcgi_script_name;
    #    include        fastcgi_params;
    #}

    # deny access to .htaccess files, if Apache's document root
    # concurs with nginx's one
    #
    #location ~ /\.ht {
    #    deny  all;
    #}
}

~
```

- location을 바꾸자.
- proxy_pass를 localhost:3000으로 바꿔서 nginx에 리액트 프로그램을 얹어 실행할 수 있다.
- 공인IP:80으로 리액트를 띄울 수 있다

- 3000 포트를 삭제해도 공인IP:80에서 리액트 창이 출력된다.

2) Was(JBoss)-DB(MySQL)

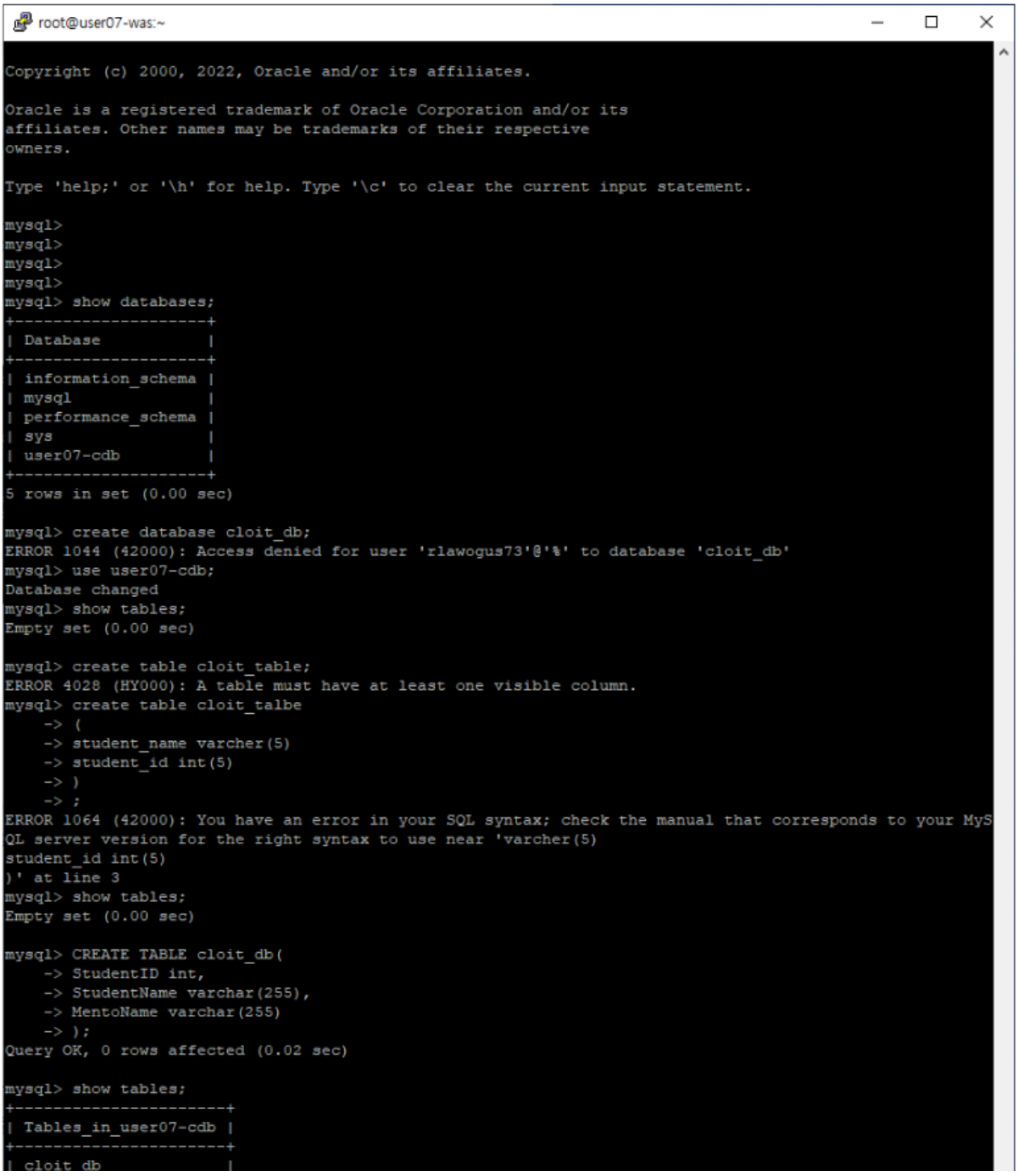
1. 연동하기

- WAS 서버에서 접속해서 MySQL을 설치해준다.

```
[root@user07-was ~]# yum install -y https://dev.mysql.com/get/mysql80-community-release-el7-3.
noarch.rpm
Loaded plugins: fastestmirror, langpacks
mysql80-community-release-el7-3.noarch.rpm                              |  25 kB  00:00:00
Examining /var/tmp/yum-root-LSKXBx/mysql80-community-release-el7-3.noarch.rpm: mysql80-communi
ty-release-el7-3.noarch
Marking /var/tmp/yum-root-LSKXBx/mysql80-community-release-el7-3.noarch.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package mysql80-community-release.noarch 0:el7-3 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                    Arch     Version Repository                      Size
================================================================================
Installing:
 mysql80-community-release  noarch   el7-3   /mysql80-community-release-el7-3.noarch   31 k

Transaction Summary
================================================================================
Install  1 Package

Total size: 31 k
Installed size: 31 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : mysql80-community-release-el7-3.noarch                       1/1
  Verifying  : mysql80-community-release-el7-3.noarch                       1/1

Installed:
  mysql80-community-release.noarch 0:el7-3

Complete!
```

- 명령어 예시를 입력하여 DB 서버에 접속할 수 있다.
- `# mysql -h db-16aac-kr1.vpc-cdb.gov-ntruss.com -u rlawogus73 -p --port 3306`

```
[root@user07-was ~]# mysql -h db-16aac-kr1.vpc-cdb.gov-ntruss.com -u rlawogus73 -p --port 3306
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1808
Server version: 8.0.25 MySQL Community Server - GPL

Copyright (c) 2000, 2022, 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>
```



- WAS 서버에서 DB 서버에 접속한 후 테이블을 생성할 수 있다.

2. 테스트

- WAS에서 서비스 띄워보기

- 준비 : Maven, spring boot 다운받기.

1) Maven 설치

- Yum install maven

```
</html>
[root@user07-was ~]# yum install maven
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
Resolving Dependencies
--> Running transaction check
```

- mvn -version으로 설치 확인을 해준다.

```
[root@user07-was ~]# mvn -version
Apache Maven 3.0.5 (Red Hat 3.0.5-17)
Maven home: /usr/share/maven
Java version: 1.8.0_302, vendor: Red Hat, Inc.
Java home: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.302.b08-0.el7_9.x86_64/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-1127.el7.x86_64", arch: "amd64", family: "unix"
[root@user07-was ~]#
```

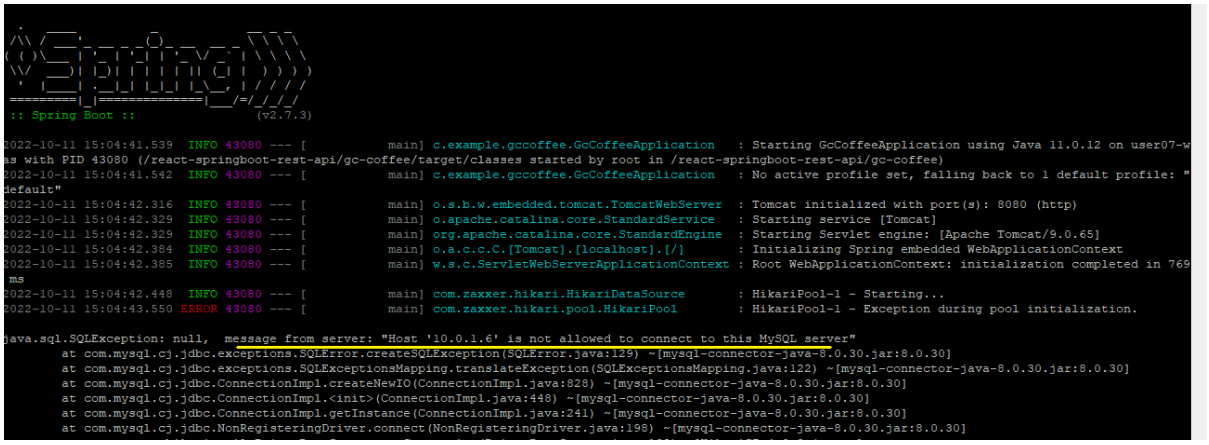
- git clone으로 WAS 서버에도 폴더를 내려 받는다.

```
dr-xr-xr-x  120 root root    0 Oct 11 12:35 proc
drwxr-xr-x    5 root root   97 Oct 11 12:58 react-springboot-rest-api
dr-xr-x---.   7 root root  240 Oct 11 15:19 root
```

- 여기서는 pom.xml 파일에서 Java 버전을 맞춰줘야 한다.
- mvn package로 target 폴더를 생성한다.

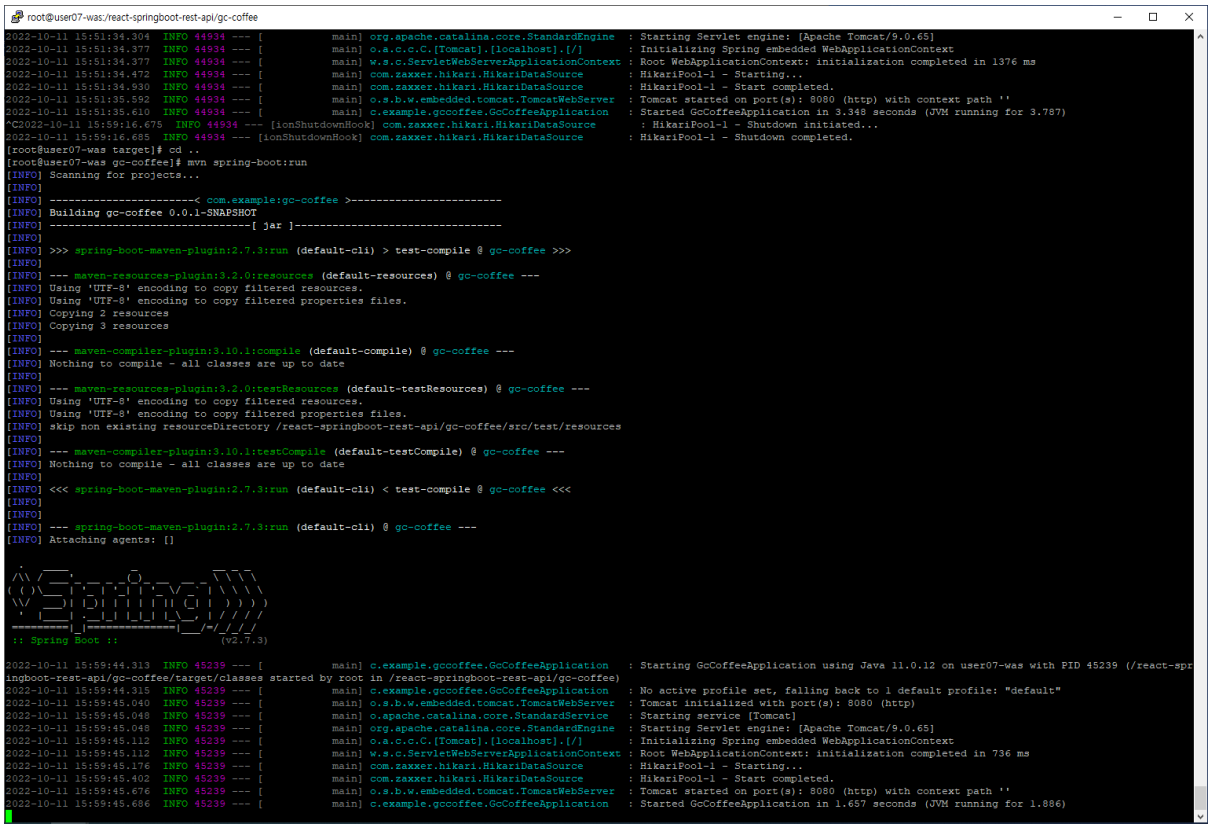
```
total 24592
drwxr-xr-x 4 root root      106 Oct 11 12:58 classes
-rw-r--r-- 1 root root 25146178 Oct 11 14:17 gc-coffee-0.0.1-SNAPSHOT.jar
-rw-r--r-- 1 root root    28045 Oct 11 14:17 gc-coffee-0.0.1-SNAPSHOT.jar.original
drwxr-xr-x 3 root root       25 Oct 11 12:58 generated-sources
drwxr-xr-x 3 root root       30 Oct 11 12:58 generated-test-sources
drwxr-xr-x 2 root root       28 Oct 11 13:55 maven-archiver
drwxr-xr-x 3 root root       35 Oct 11 12:58 maven-status
drwxr-xr-x 2 root root     4096 Oct 11 13:02 surefire-reports
drwxr-xr-x 3 root root       17 Oct 11 12:58 test-classes
```

- 빌드가 성공하면 gc-coffee-0.0.1-SNAPSHOT.jar 가 생성된다.
- java -jar gc-coffee-0.0.1-SNAPSHOT.jar 명령어를 통해 실행해준다.



- MySQL 서버에 접속할 수 있는 권한이 없다는 메시지가 나온다.
- MySQL 사용자를 추가해서 해결할 수 있다.

- Root 사용자를 추가하였고, 외부에서도 MySQL 접속이 가능하게 되었다.



- 빌드가 성공한 모습

능력부족으로 실패했습니다

WAS와 Web서버를 연결해서 웹에서 보내는 데이터를 WAS가 DB에게 전달해 저장하는 모습을 보고 싶었는데, 거기까지는 하지 못했습니다.