**멀티캠퍼스 [K-Digital] 프로젝트형 클라우드(MSA) 서비스 개발 과정**

**AWS 이용 매뉴얼**

문서 기록

|  |  |  |
| --- | --- | --- |
| 작성자 | 날짜 | 설명 |
| 구자윤 | 2022-02-07 | Ver 1.0 : 처음 생성 |
| 구자윤 | 2022-02-10 | Ver 20 : RDS 추가 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

목차

[1. VPC 생성 4](#_Toc95392157)

[VPC 생성하기 (VPC CIDR 생성) 4](#_Toc95392158)

[서브넷 생성하기 6](#_Toc95392159)

[2. 네트워크와 보안 생성 및 설정 11](#_Toc95392160)

[인터넷 게이트웨이 생성 후 VPC 연결하기 11](#_Toc95392161)

[라우팅 테이블 추가 13](#_Toc95392162)

[네트워크 ACL(NACL) 생성 16](#_Toc95392163)

[NACL 인바운드 아웃바운드 규칙 설정 20](#_Toc95392164)

[보안 그룹 생성 23](#_Toc95392165)

# 1. VPC 생성

## VPC 생성하기 (VPC CIDR 생성)

- 내용

VPC 생성 > CIDR 계산기 사이트에서 IP배분하고 CIDR 넘버 확인하기

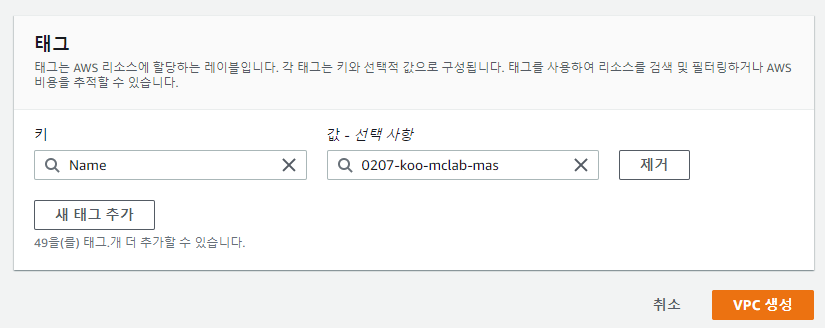
예시)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 실제 사용수 | 예상 사용수 | 범위 |  |
| CIDR | 2000 | 4000 | 10.0.0.0 - 10.0.15.255 | 10.0.0.0/20 |
| 웹 서버 | 200 | 1000 | 10.0.0.0 - 10.0.3.255 | 10.0.0.0/21 |
| API(WAS) 서버 | 100 | 500 | 10.0.8.0 - 10.0.9.255 | 10.0.8.0/22 |
| DB 서버1 | 5 | 20 | 10.0.13.0 - 10.0.13.255 | 10.0.13.0/24 |
| DB 서버2 | 5 | 20 | 10.0.14.0 - 10.0.14.255 | 10.0.14.0/24 |



CIDR 계산기에서 계산한 값 입력

이름 입력



## 서브넷 생성하기

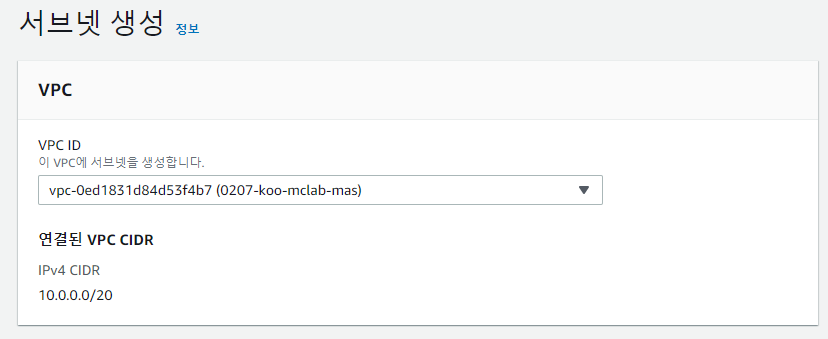
- 내용

위에 CIDR로 IP배분한걸 기억해서 입력

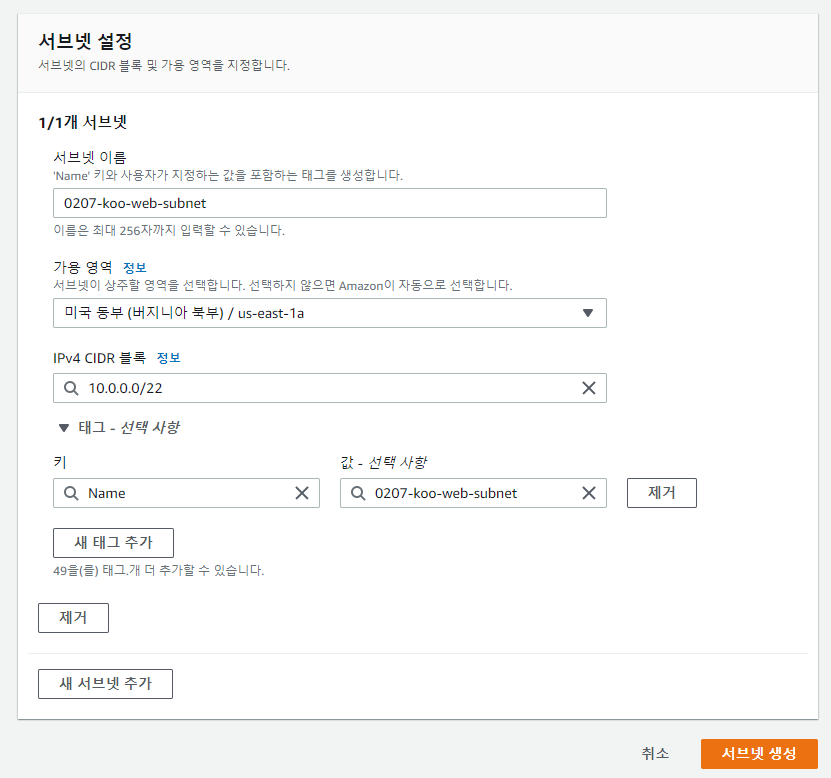
각 알맞은 서브넷의 이름과 CIDR를 올바르게 입력한다!

- 여기까지는 인터넷 연결이 안되어있으므로 프라이빗 서브넷이다

>> 추가내용 여기서 RDS용 서브넷을 만들 때 같은 가용 영역으로 하지 말 것!



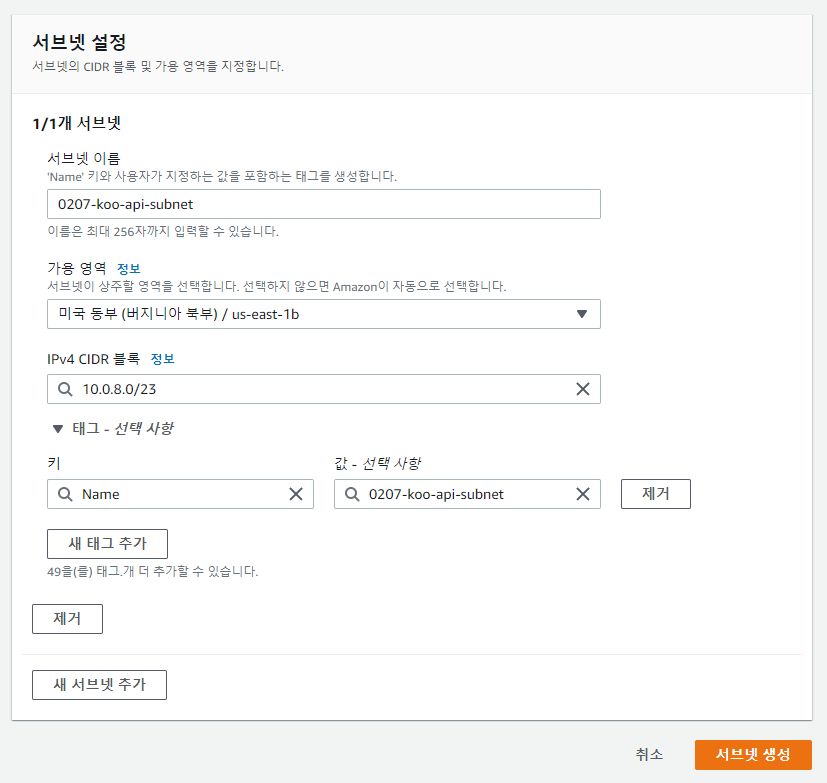
생성한 VPC 선택

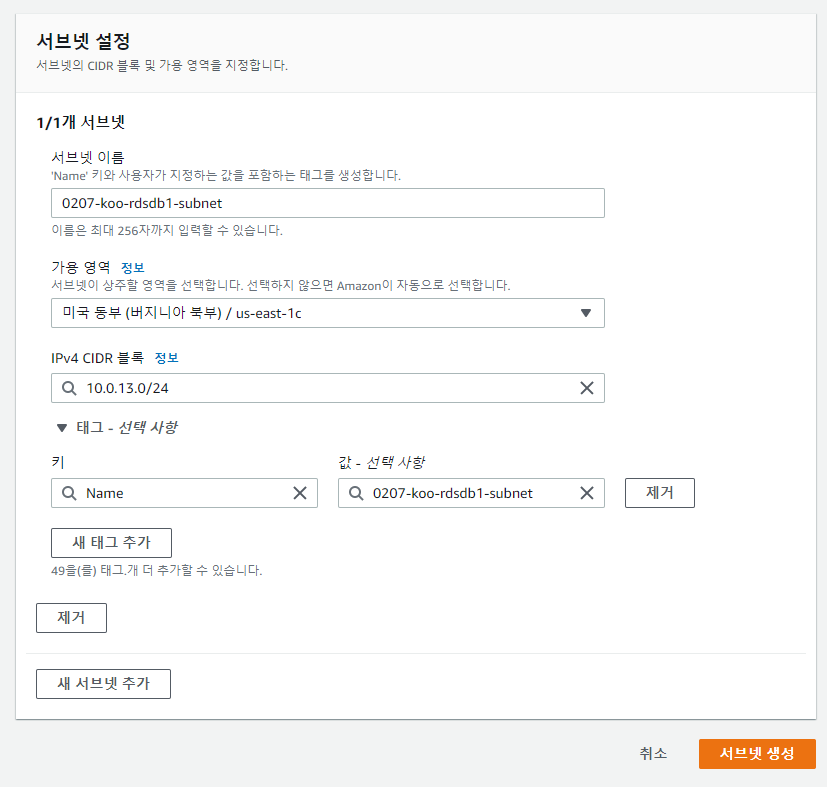
**※ 여기서 각 서브넷을 생성할 때 가용영역을 다르게 설정하는 것이 좋다. 현재 DB서브넷을 두개를 만들어서 다른 가용영역에 저장하게 할 것이기 때문에 DB서브넷은 꼭 다른 가용영역으로 선택할 것**

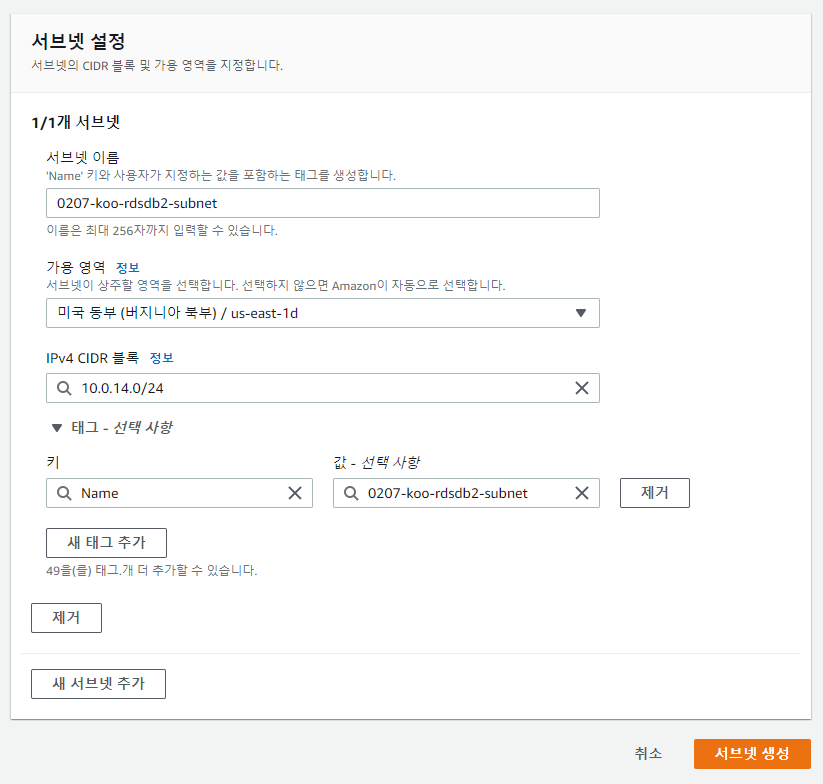
가용 영역 겹치지않게 설정

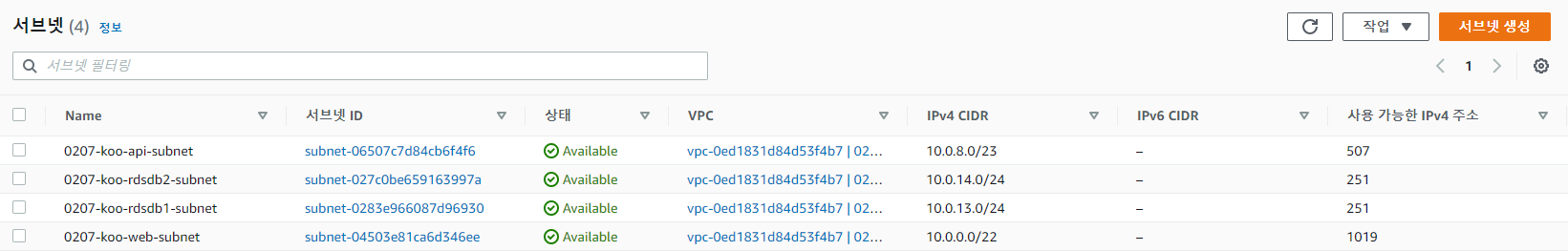
웹 서브넷CIDR 입력

웹 서브넷 이름 입력









서브넷에서 필터를 풀고 생성한 네가지 서브넷이 제대로 생성되어있는 지 확인

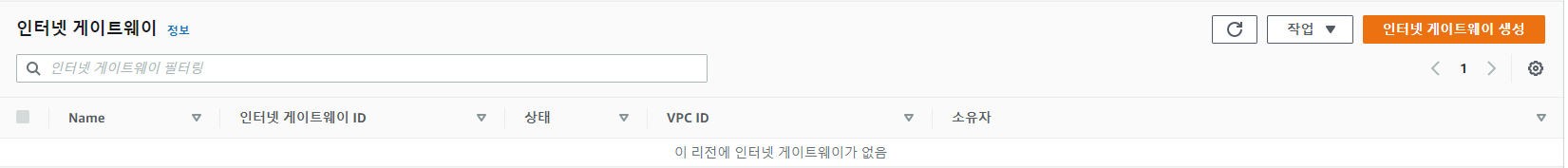
# 2. 네트워크와 보안 생성 및 설정

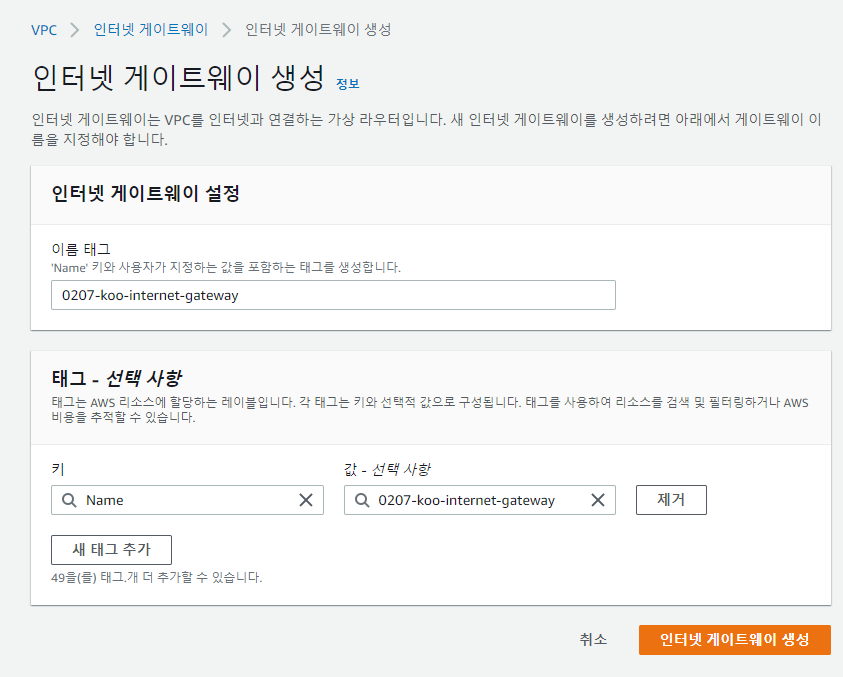
**※프라이빗에서 퍼블릭 서브넷으로 변경하기**

## 인터넷 게이트웨이 생성 후 VPC 연결하기

-내용

인터넷 게이트웨이 생성 > (Detached 상태) > 작업 - VPC 연결 (VPC 아이디를 확인 할 것!) > (Atached 상태로 변경 확인!)





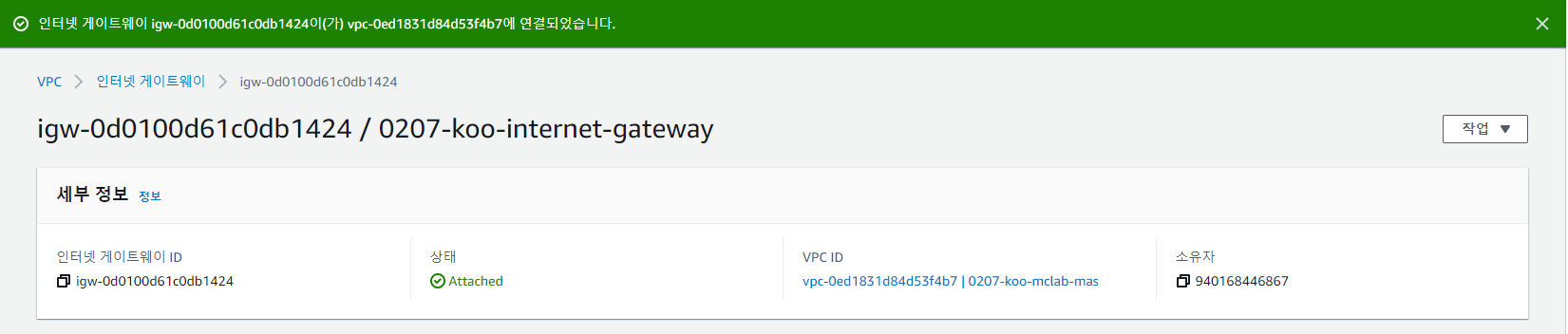
이름 태그에 이름 입력 > 인터넷 게이트웨이 생성 클릭



현재 Detached 상태 확인 > 상단 VPC에 연결 클릭



사용 가능한 VPC를 선택 후 인터넷 게이트웨이 연결 클릭

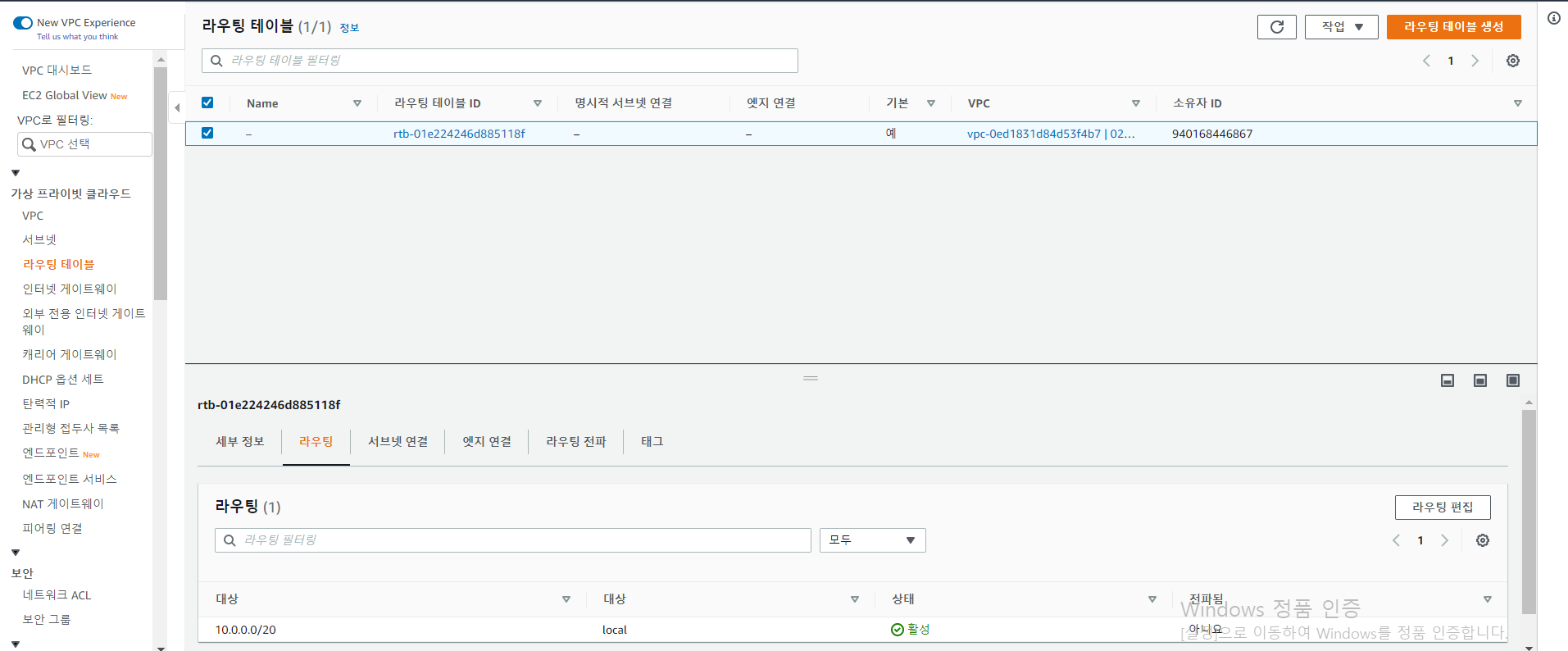


Attached 상태로 변경 확인

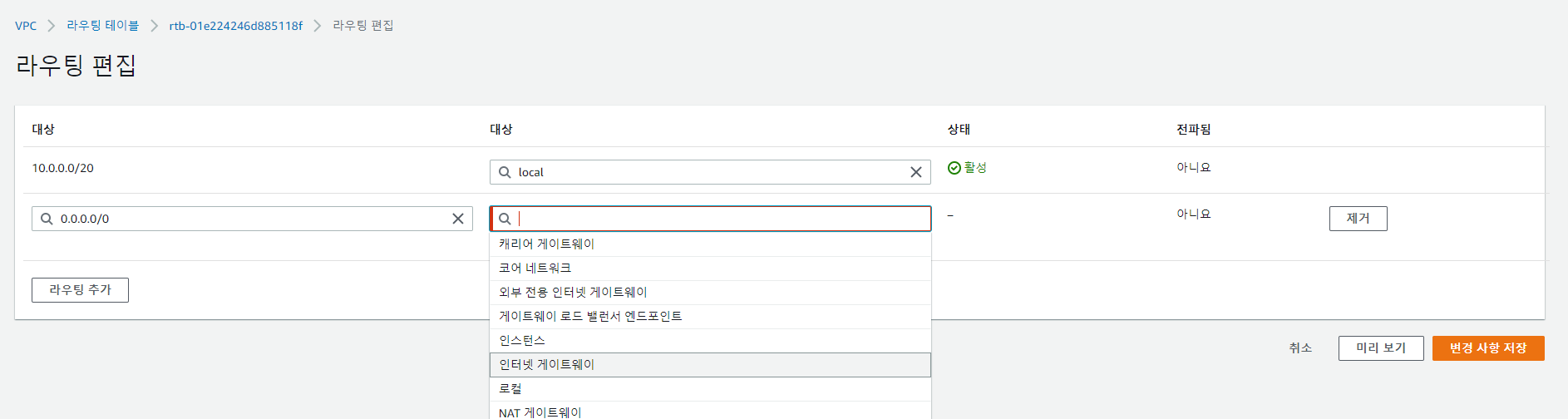
## 라우팅 테이블 추가

> 라우팅 - 편집 > 라우팅 추가 > [0.0.0.0/0] 입력(0.0.0.0/0 인터넷 연결) > 오른쪽 텍스트 테이블 > 인터넷 게이트웨이 클릭 > 생성한 인터넷 게이트웨이 클릭 > 변경 사항 저장

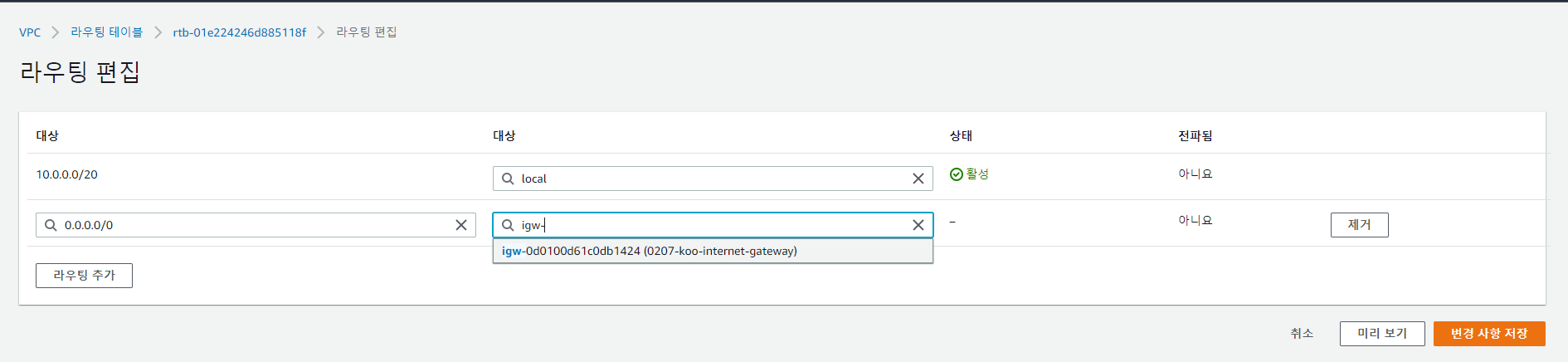
> 서브넷 연결 > 서브넷 연결 편집 > 인터넷 연결을 원하는 서브넷 선택 후 연결 저장



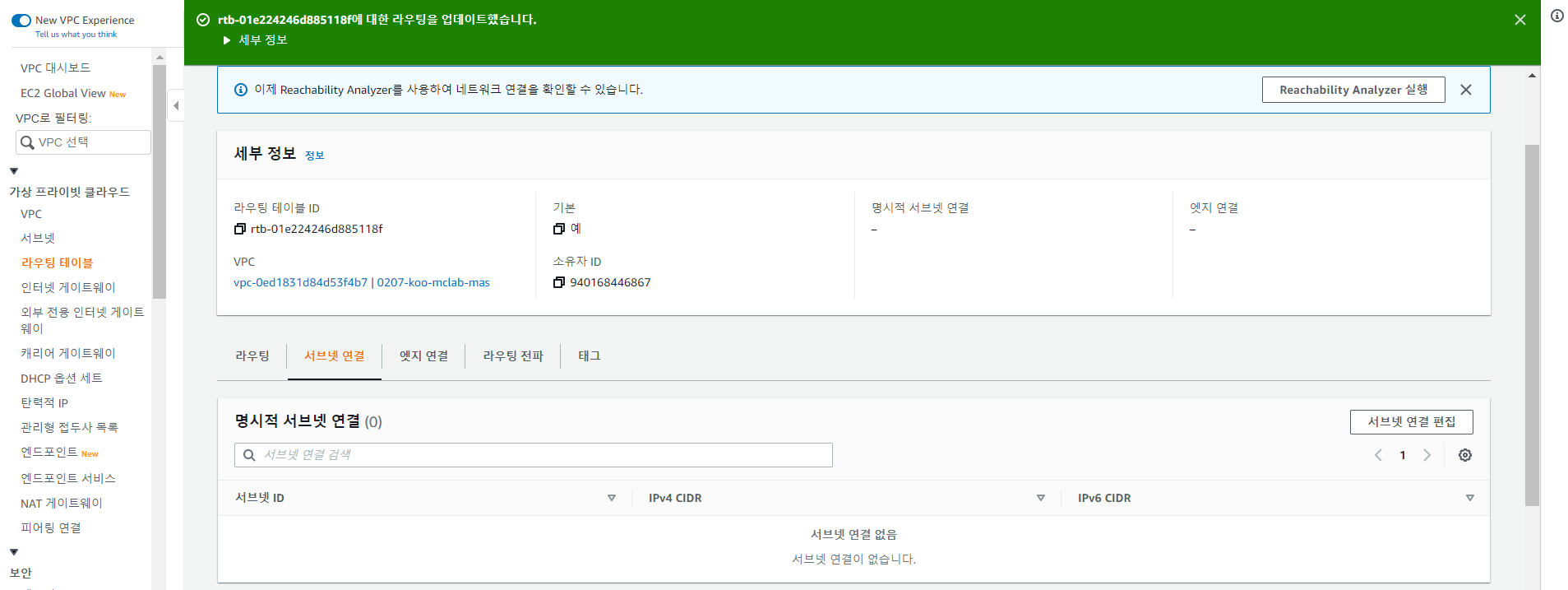
왼쪽 목차에 라우팅 테이블 클릭 후 하단 라우팅 > 라우팅 편집 클릭



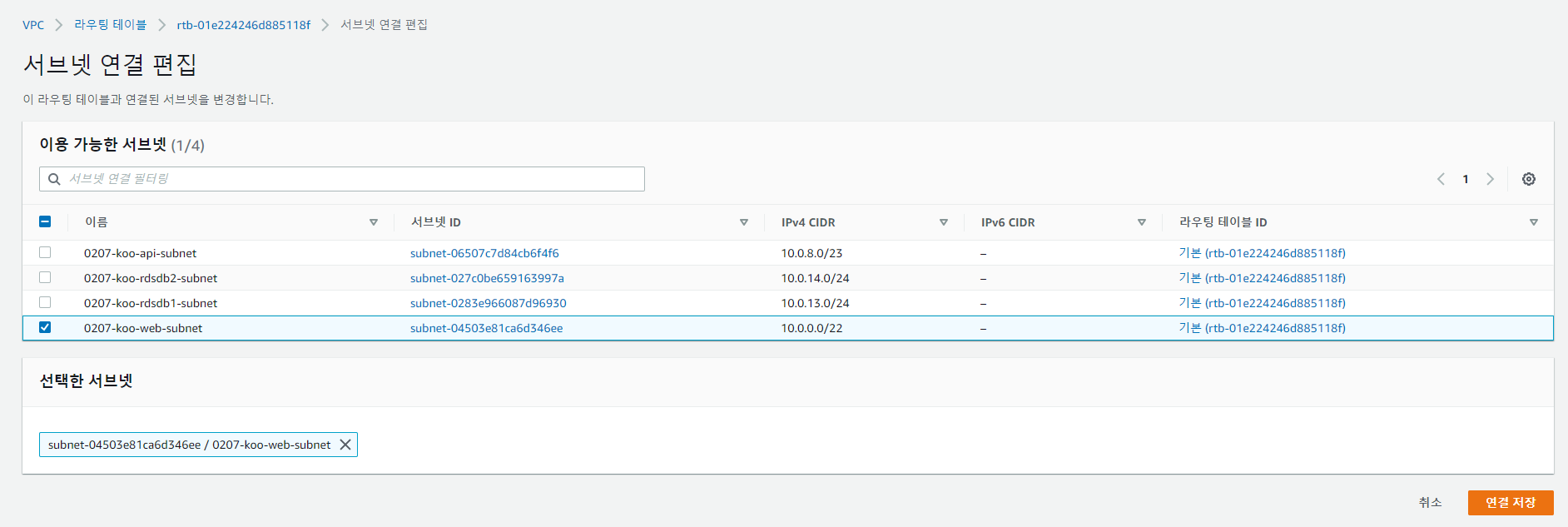
라우팅 추가 > [0.0.0.0/0] 입력(0.0.0.0/0 인터넷 연결) > 오른쪽 텍스트 테이블 > 인터넷 게이트웨이 클릭



생성한 인터넷 게이트웨이 클릭 > 변경 사항 저장



서브넷 연결 > 서브넷 연결 편집



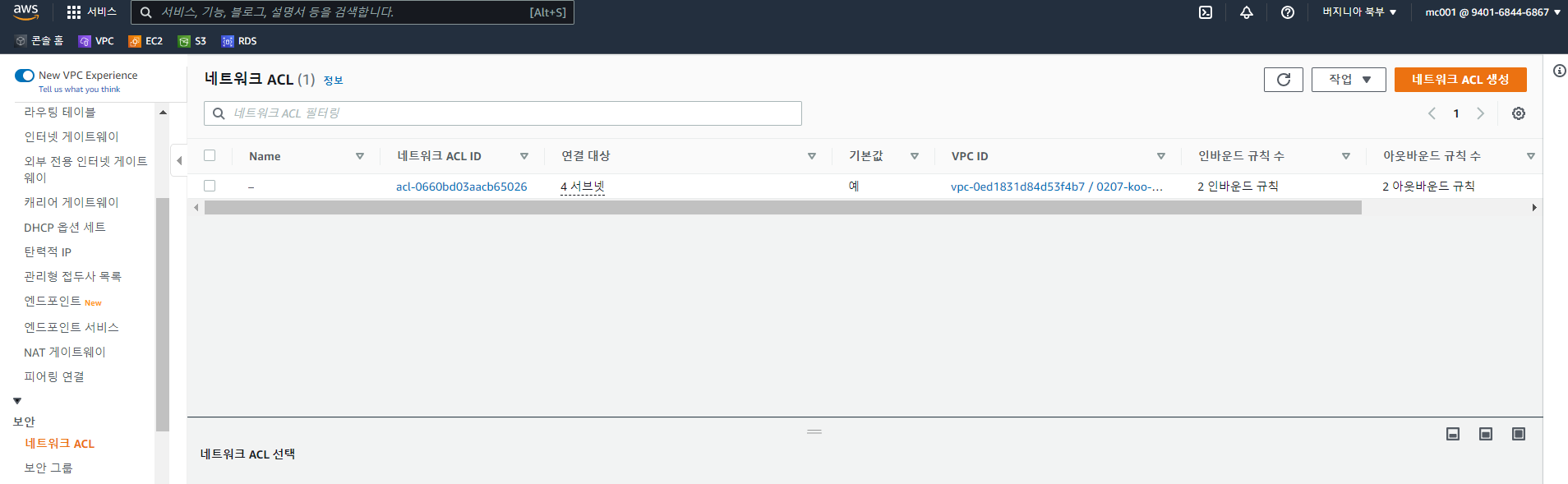
인터넷 연결을 원하는 서브넷 선택 후 연결 저장

## 네트워크 ACL(NACL) 생성

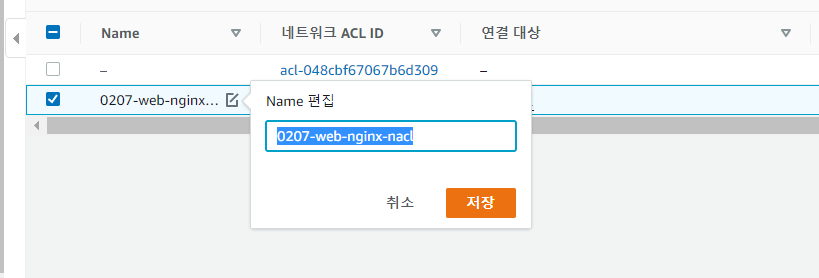
-내용

> 각 서브넷에 맞는 NACL 생성 (기본 생성된 NACL를 웹 서버로 활용하고 나머지 두 개를 만든다) > 생성 후 각 NACL에 서브넷 연결

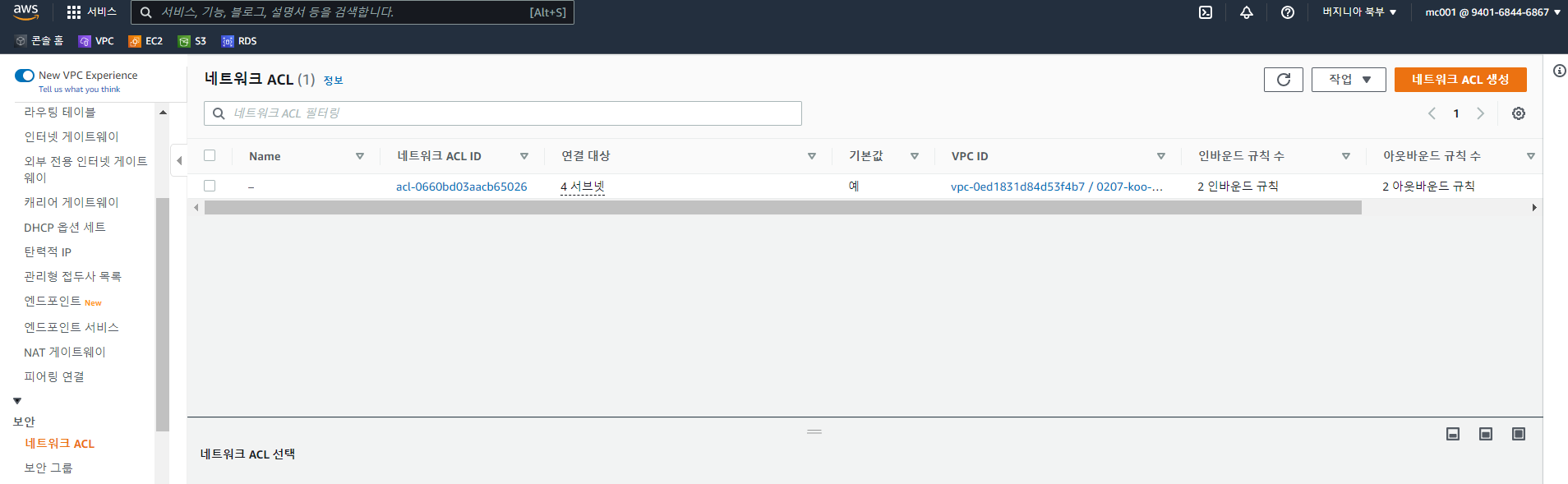
~추가내용 두가지 RDS는 같은 NACL로 묶기



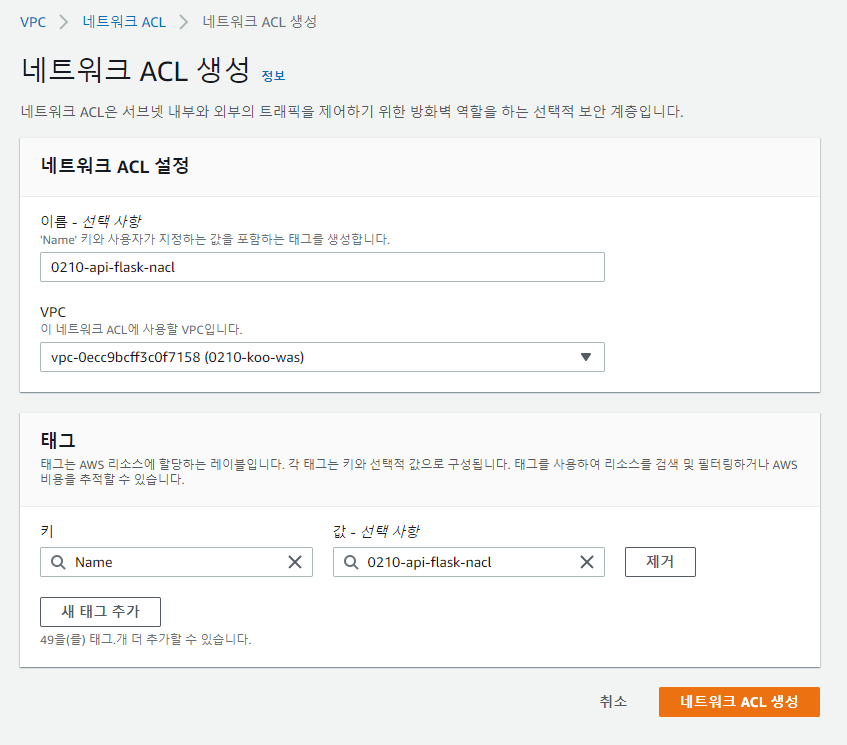
목차에서 보안 – 네트워크 ACL 클릭



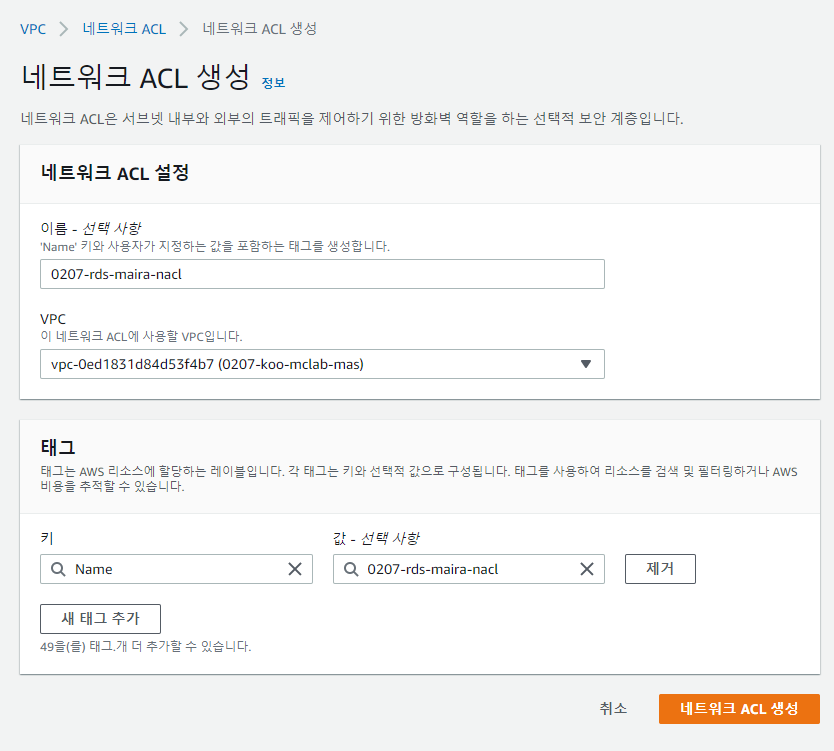
기본 생성된 NACL을 Name 편집으로 입력하고 웹 서버로 활용



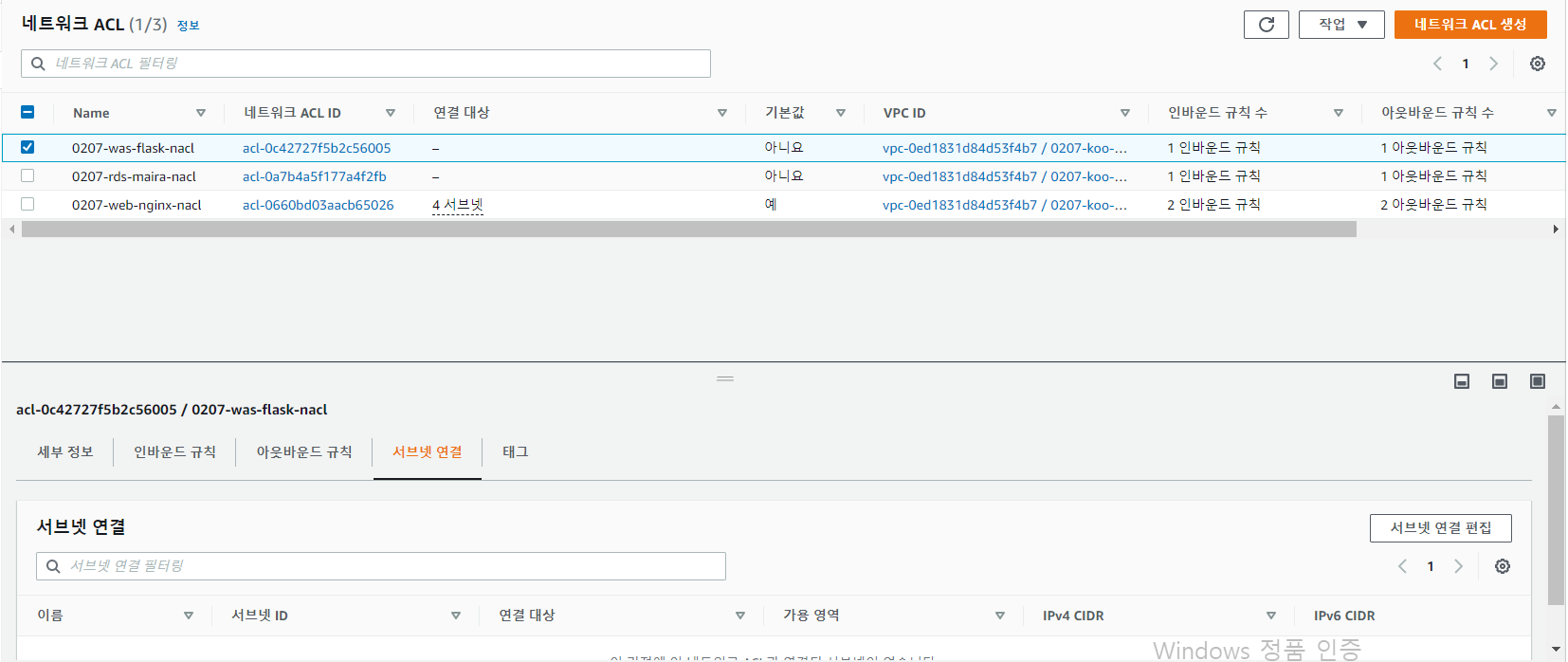
우측 상단에 네트워크 ACL 생성 클릭으로 NACL 생성



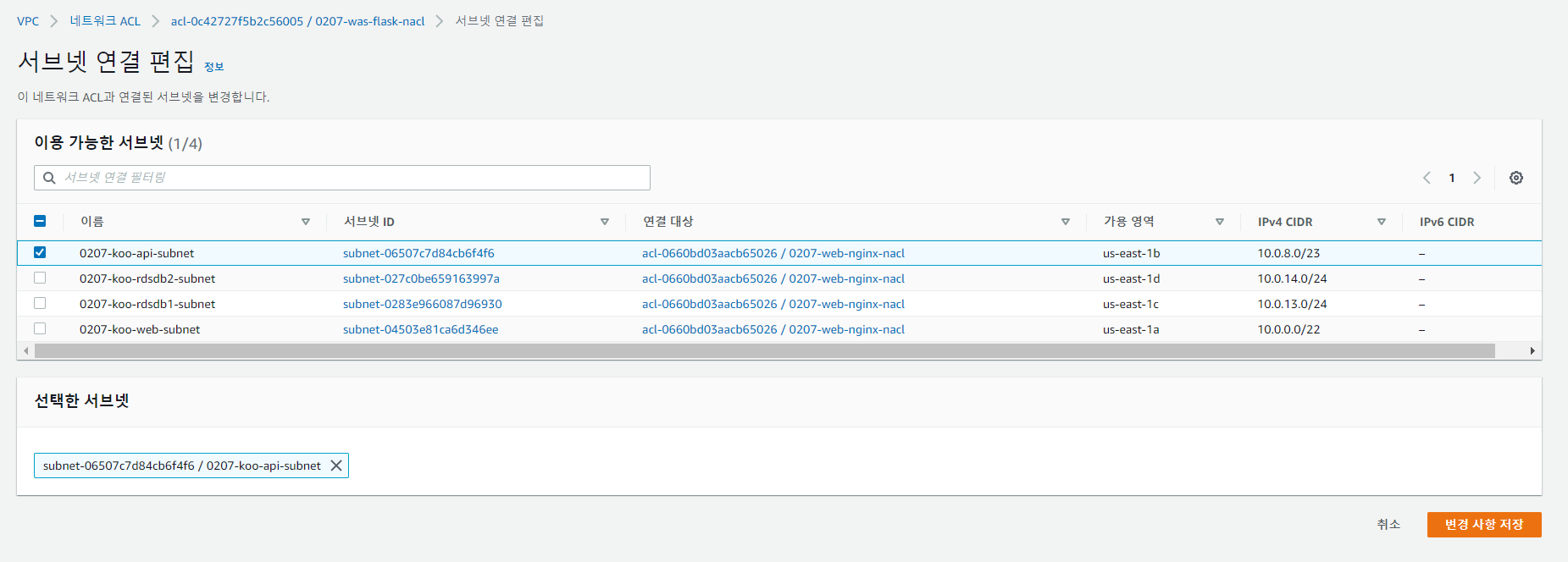
API NACL 생성 – 이름 입력 > VPC 선택 > 네크워크 ACL 생성 클릭



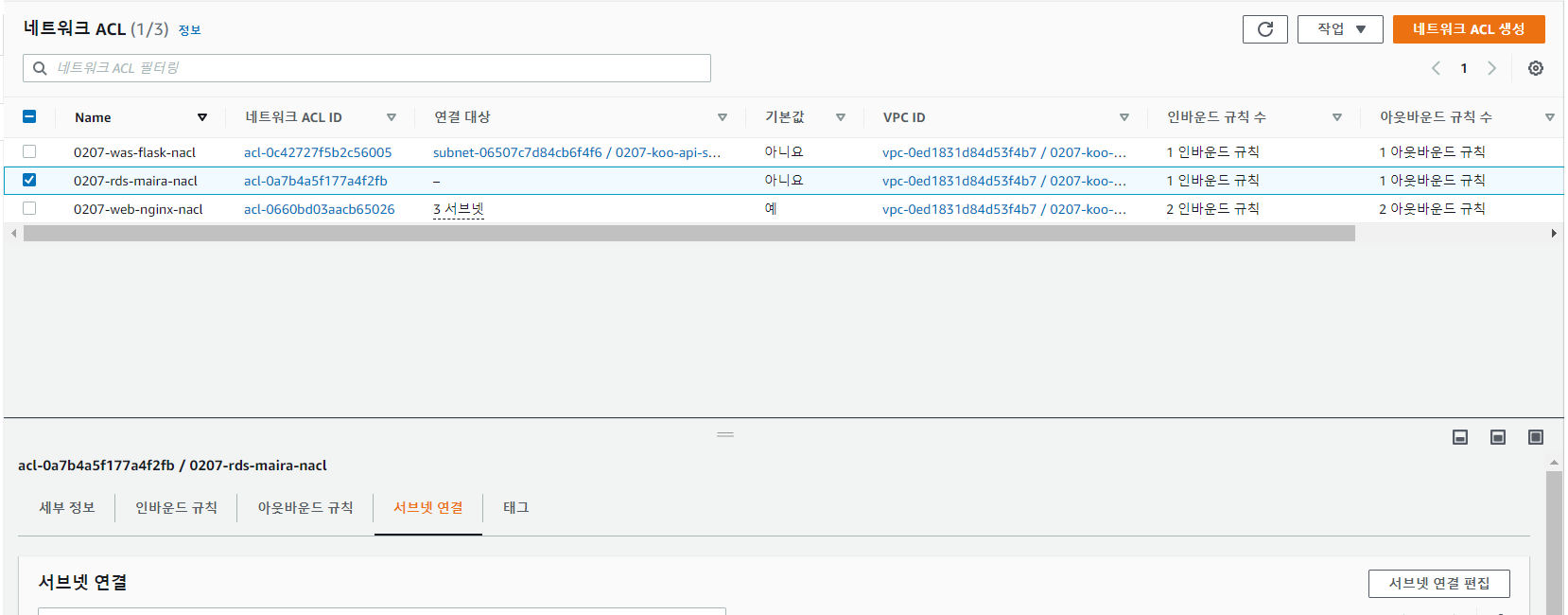
DB NACL 생성



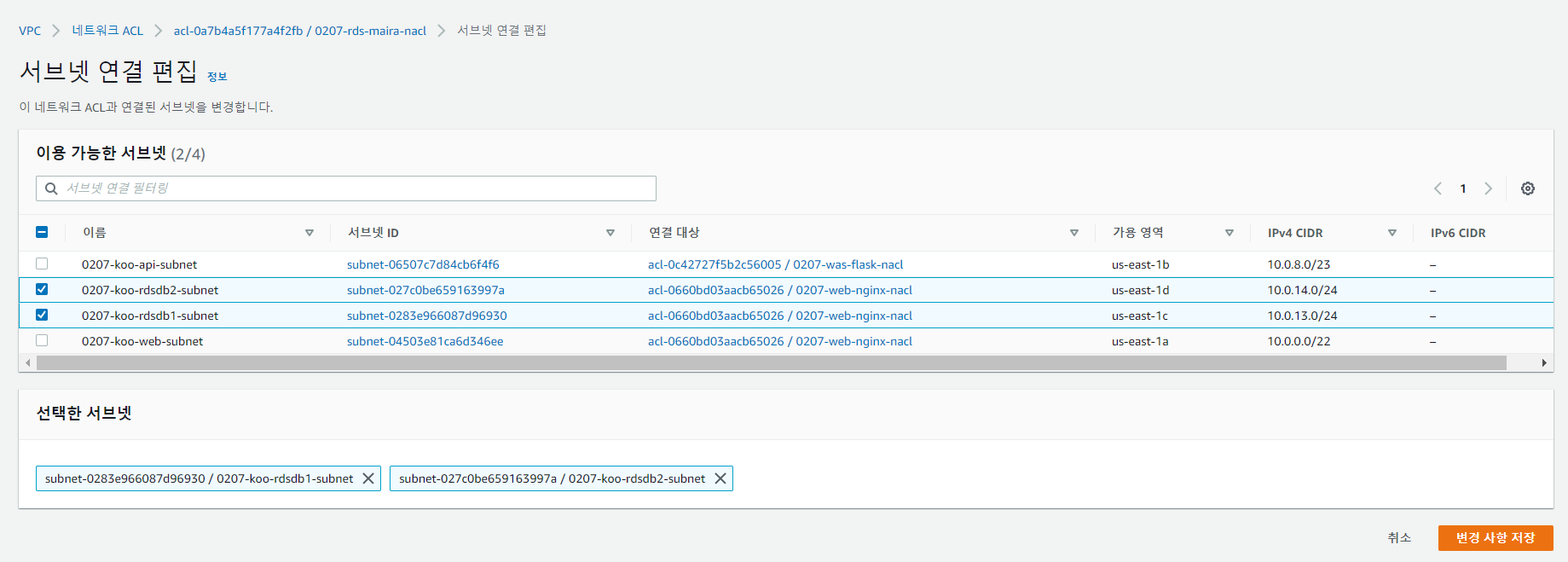
API NACL 선택 > 서브넷 연결 편집 클릭



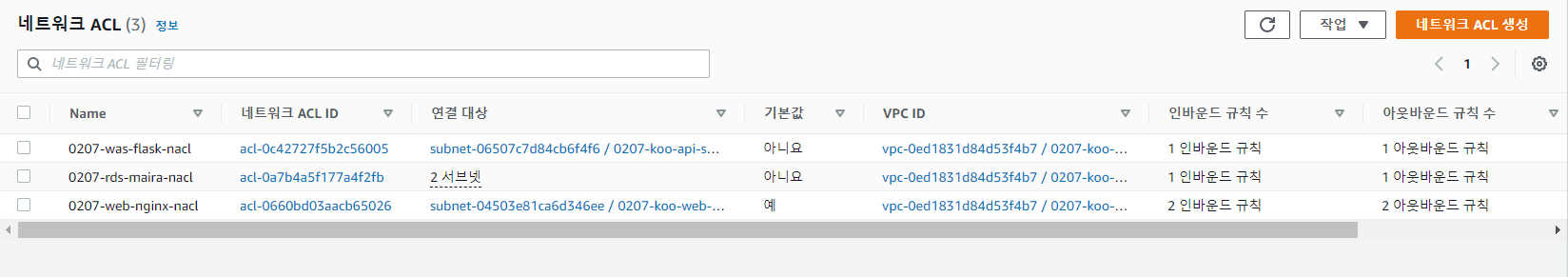
API 서브넷 > NACL에 연결



DB NACL 체크 > 하단에 서브넷 연결 편집



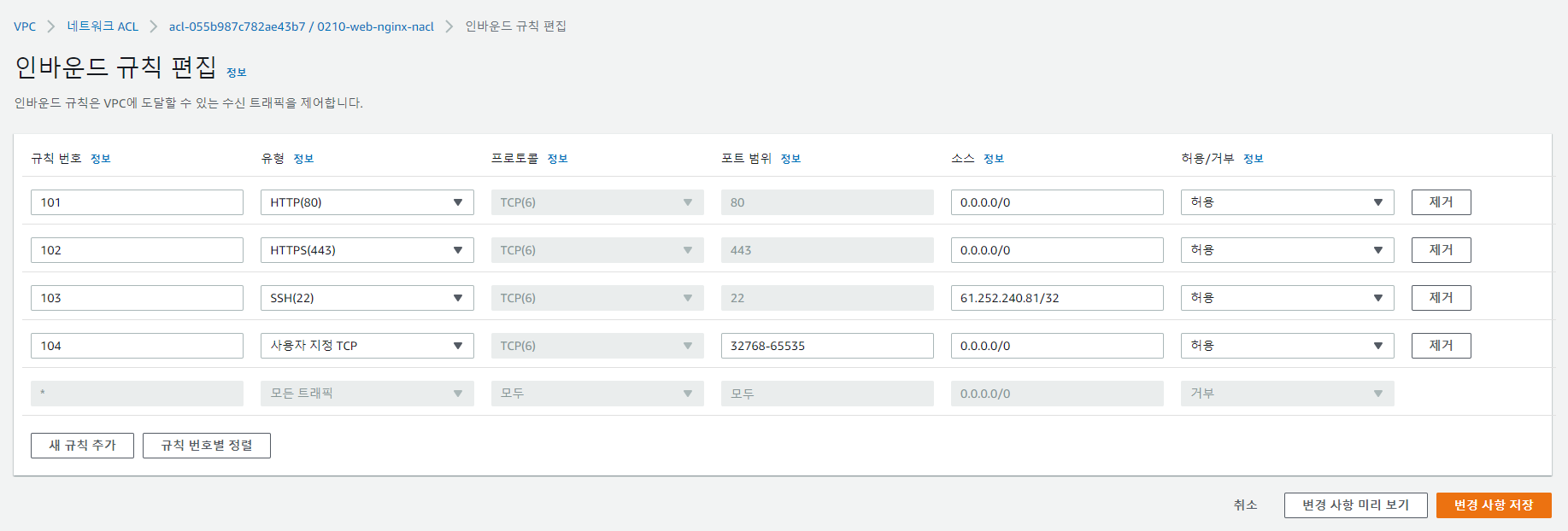
DB 서브넷 > NACL에 연결

NACL 생성 확인 및 서브넷 연결 확인

## NACL 인바운드 아웃바운드 규칙 설정

-내용

구성도에 맞게 각 NACL의 인-아웃바운드 규칙은 변경한다



먼저 웹 NACL 인바운드 규칙 편집

- 기존 규칙은 제거하고 4가지의 규칙을 추가하고 변경 사항 저장을 누른다.

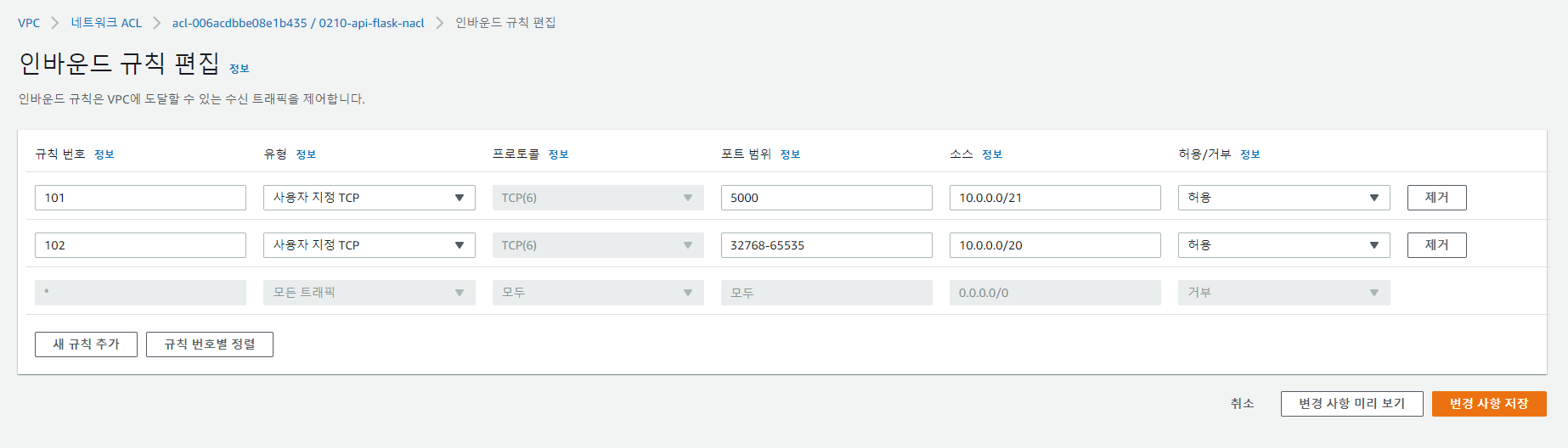
- HTTP(80), 0.0.0.0/0

- HTTPS(443), 0.0.0.0/0

- SSH(22), [자신의 IP]/32

- 동적 포트(32768-65535), 0.0.0.0/0

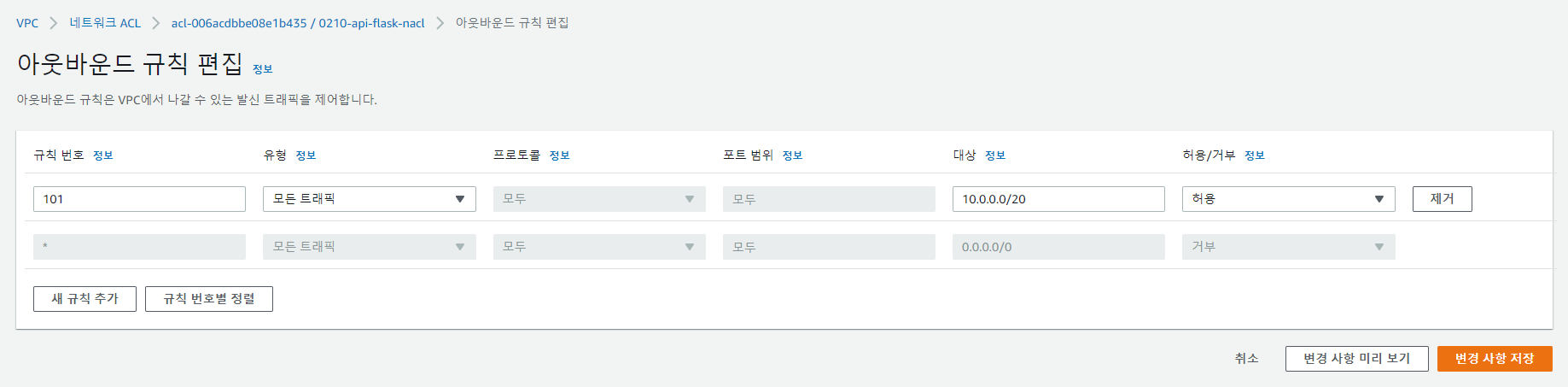
현재 웹 NACL 아웃바운드는 편집할 필요 없다 > 모든 인터넷과 연결되기 때문



다음 API NACL 인바운드 규칙 편집

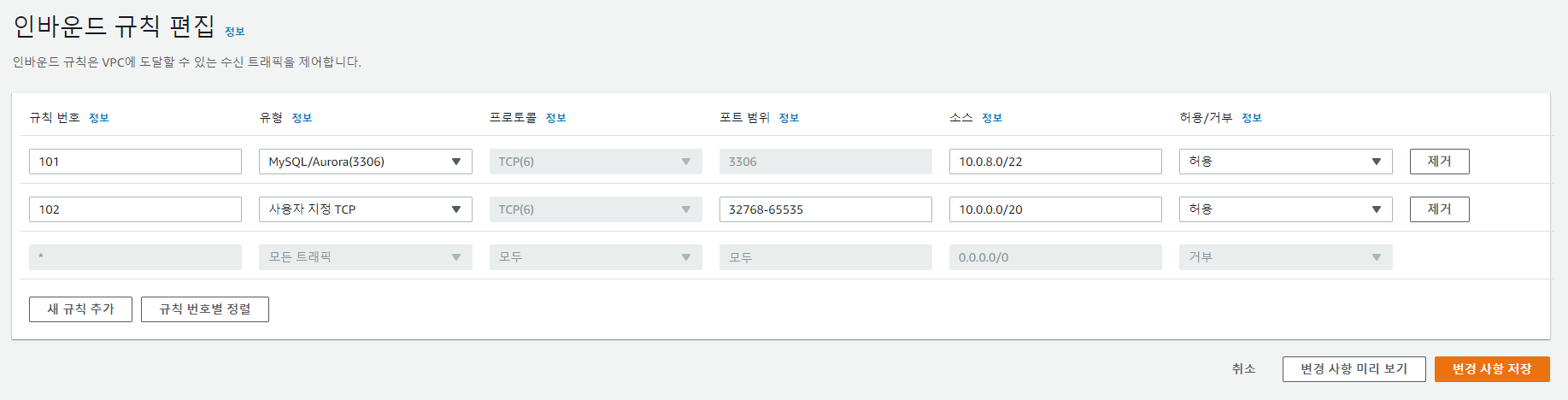
- 사용자 지정 TCP(5000) > Flask, 10.0.0.0/21 > 웹 네트워크 통신만 연결(웹 라우팅 주소값)

- 동적 포트(32768-65535), 10.0.0.0/20 > 저장한 모든 네트워트 통신 연결(모든 라우팅 주소값)

API NACL 아웃바운드 규칙 편집

1가지의 규칙을 추가하고 변경 사항 저장을 누른다.

- 모든 트래픽, 10.0.0.0/20 > 저장한 모든 네트워트 통신 연결(모든 라우팅 주소값)

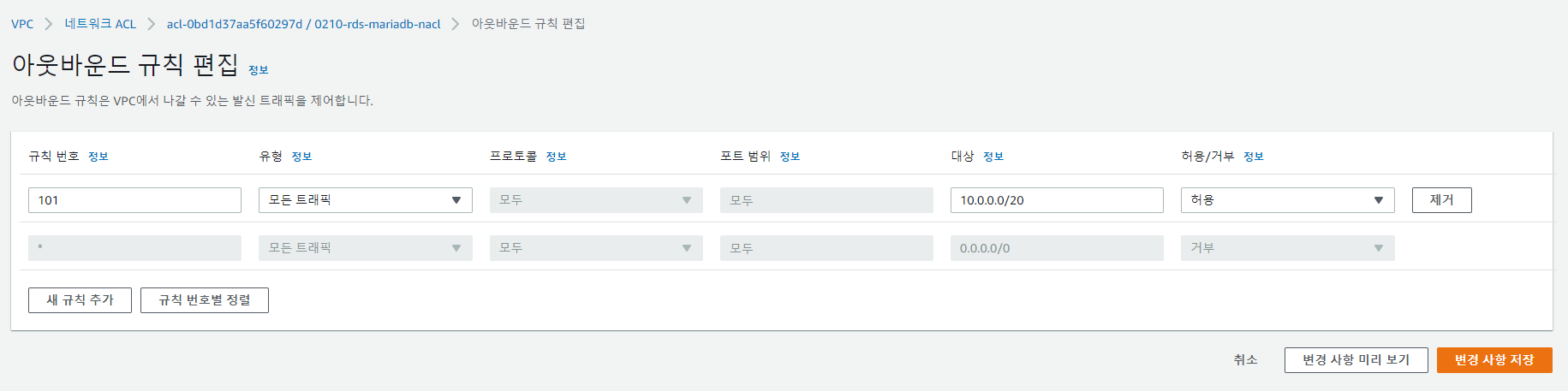


다음 RDS DB NACL 인바운드 규칙 편집

2가지의 규칙을 추가하고 변경 사항 저장을 누른다.

- MySQL/Aurora(5000) > MariaDB와 같다, 10.0.8.0/22> 웹 네트워크 통신만 연결(웹 라우팅주소값)

- 동적 포트(32768-65535), 10.0.0.0/20 > 저장한 모든 네트워트 통신 연결(모든 라우팅 주소값)



RDS DB NACL 아웃바운드 규칙 편집

1가지의 규칙을 추가하고 변경 사항 저장을 누른다.

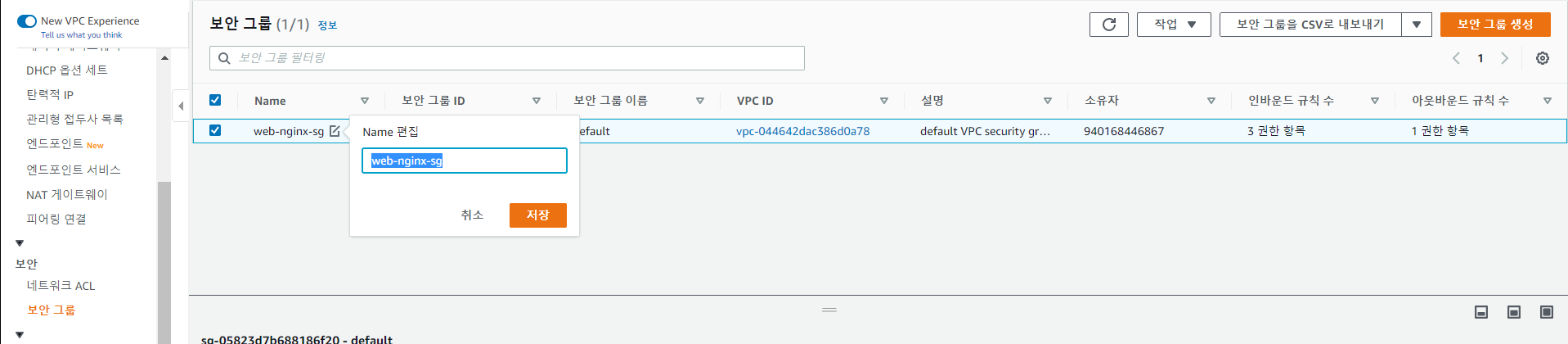
- 모든 트래픽, 10.0.0.0/20 20 > 저장한 모든 네트워트 통신 연결(모든 라우팅 주소값)

## 보안 그룹 생성

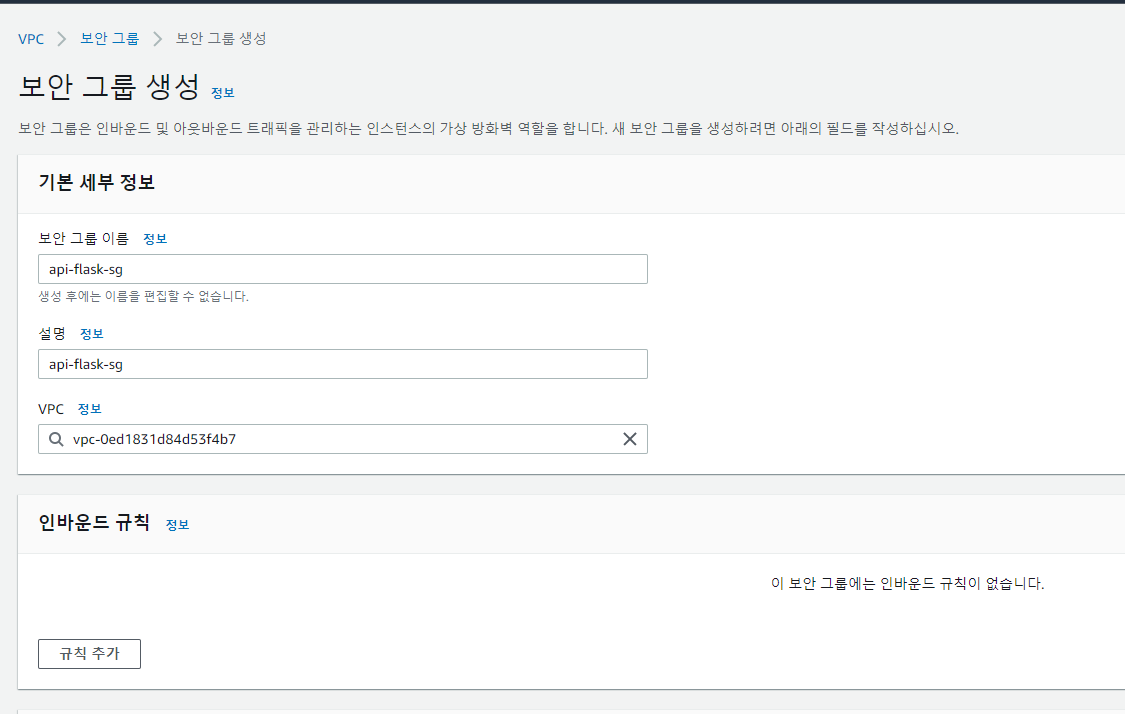
-내용

> 각 서브넷에 맞는 보안 그룹 생성 (기본 default 생성된 SG를 웹 서버로 활용해도 되고 안해도 되고 활용하면 나머지 두 개를 만든다)

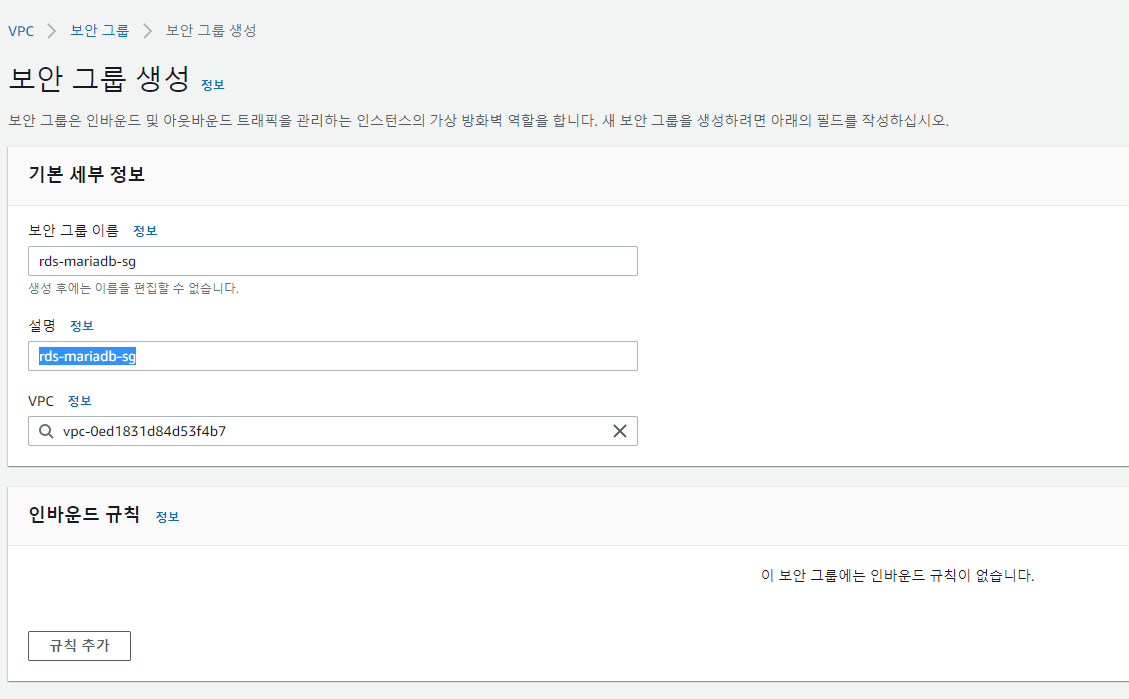
>> 보안 그룹에 NACL연결은 태그에 첫칸에 Name 입력하고 다음칸에 각 NACL의 이름을 찾아 입력



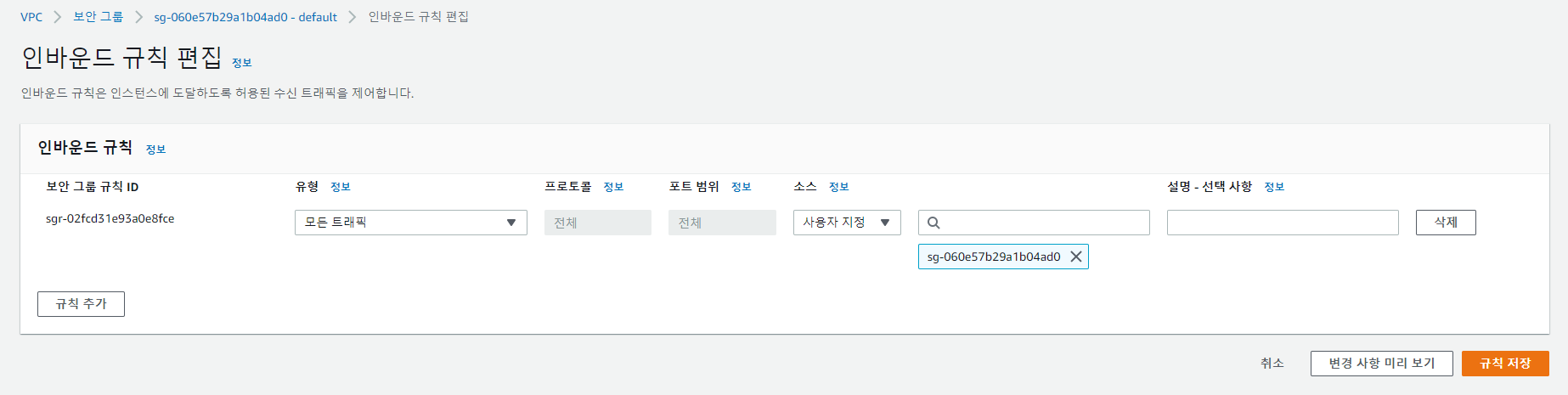
default SG의 Name을 편집하여 web-nginx-sg 로 변경



api-flask-sg 생성 >> 인바운드 아웃바운드 규칙은 나중에 설정

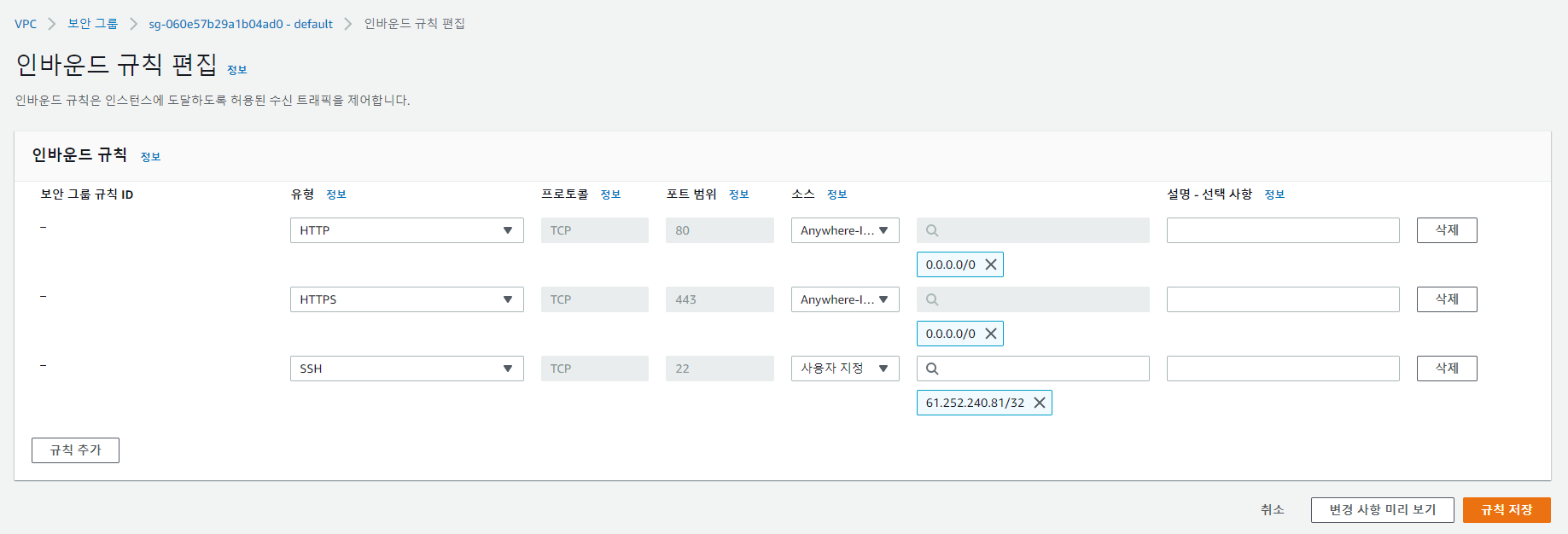


rds-mariadb-sg 생성 >> 인바운드 아웃바운드 규칙은 나중에 설정



web-nginx-sg(default) 인바운드 규칙 편집

-기존 default 규칙 삭제

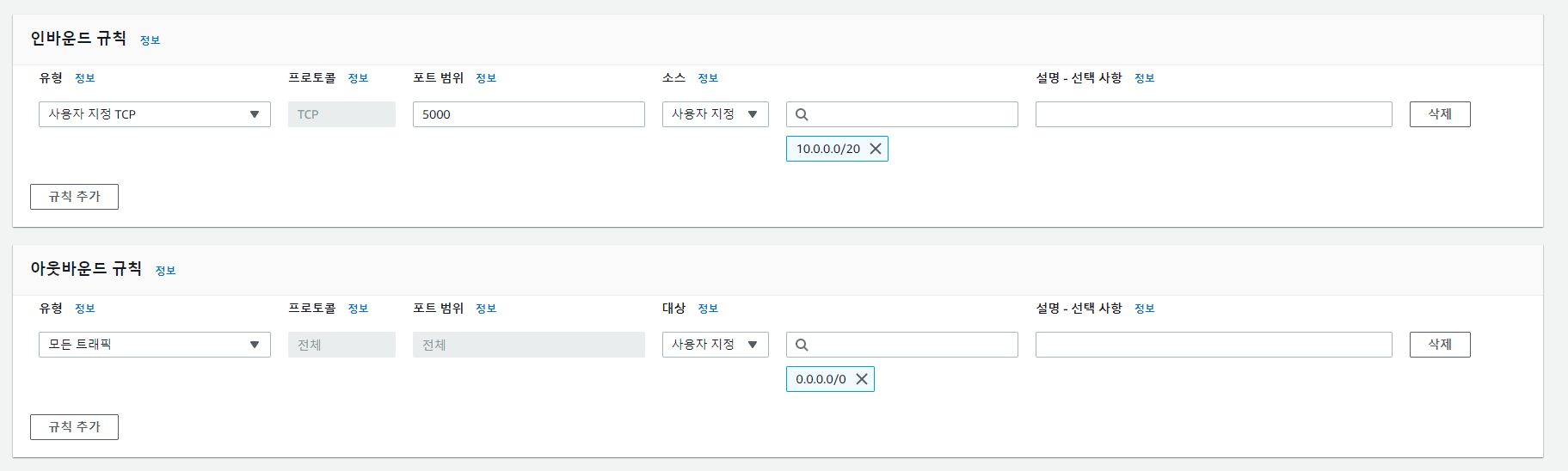


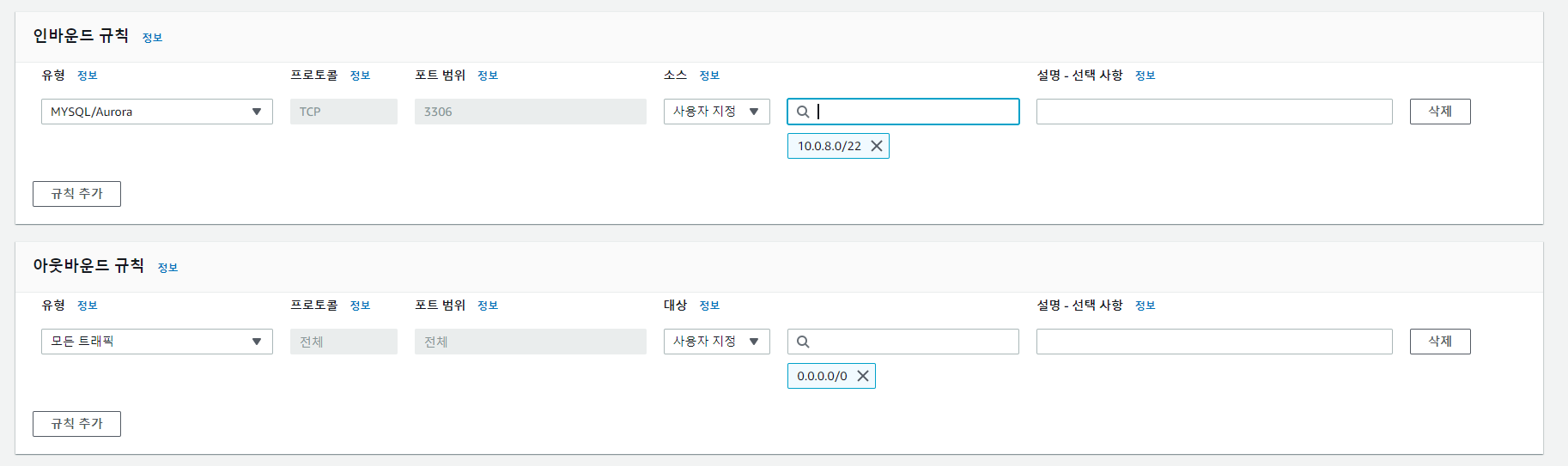
3가지 규칙 추가

- HTTP(80), 0.0.0.0/0

- HTTPS(443), 0.0.0.0/0

- SSH(22), [자신의 IP]/32

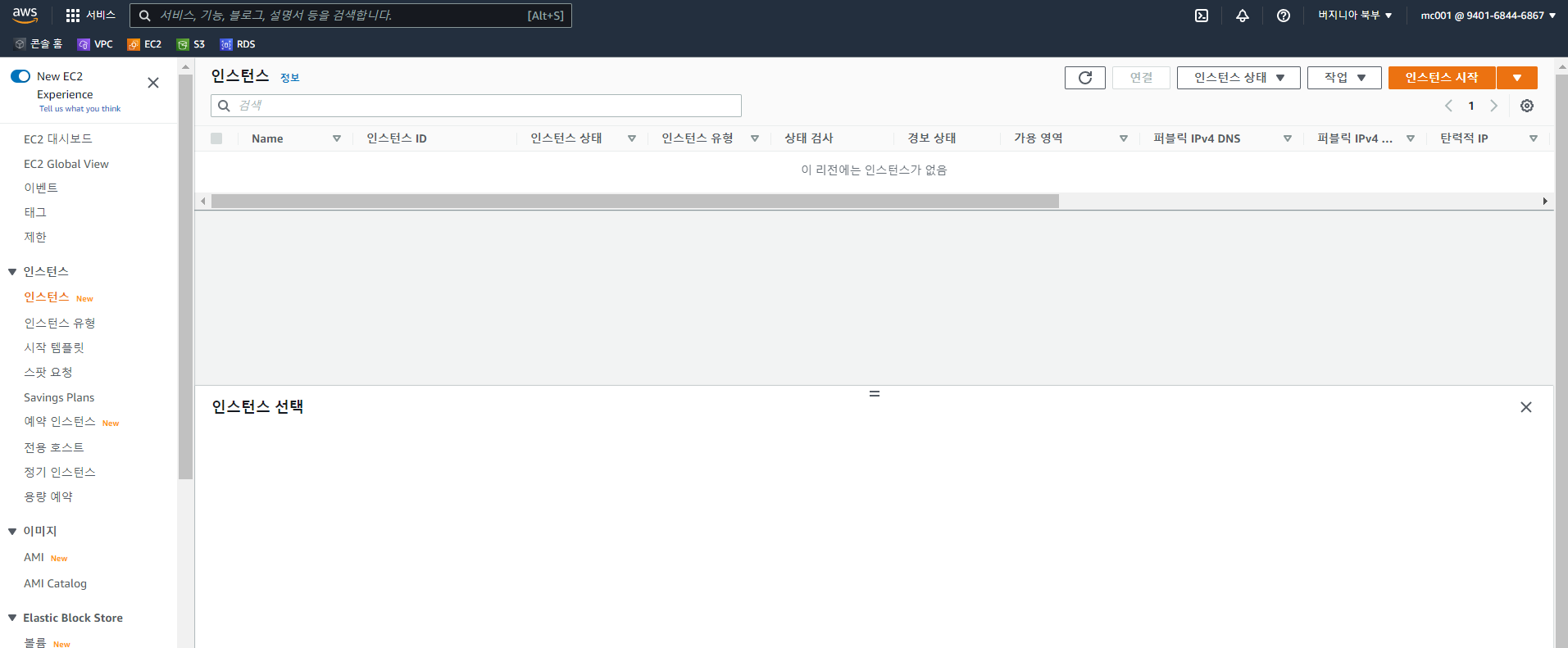




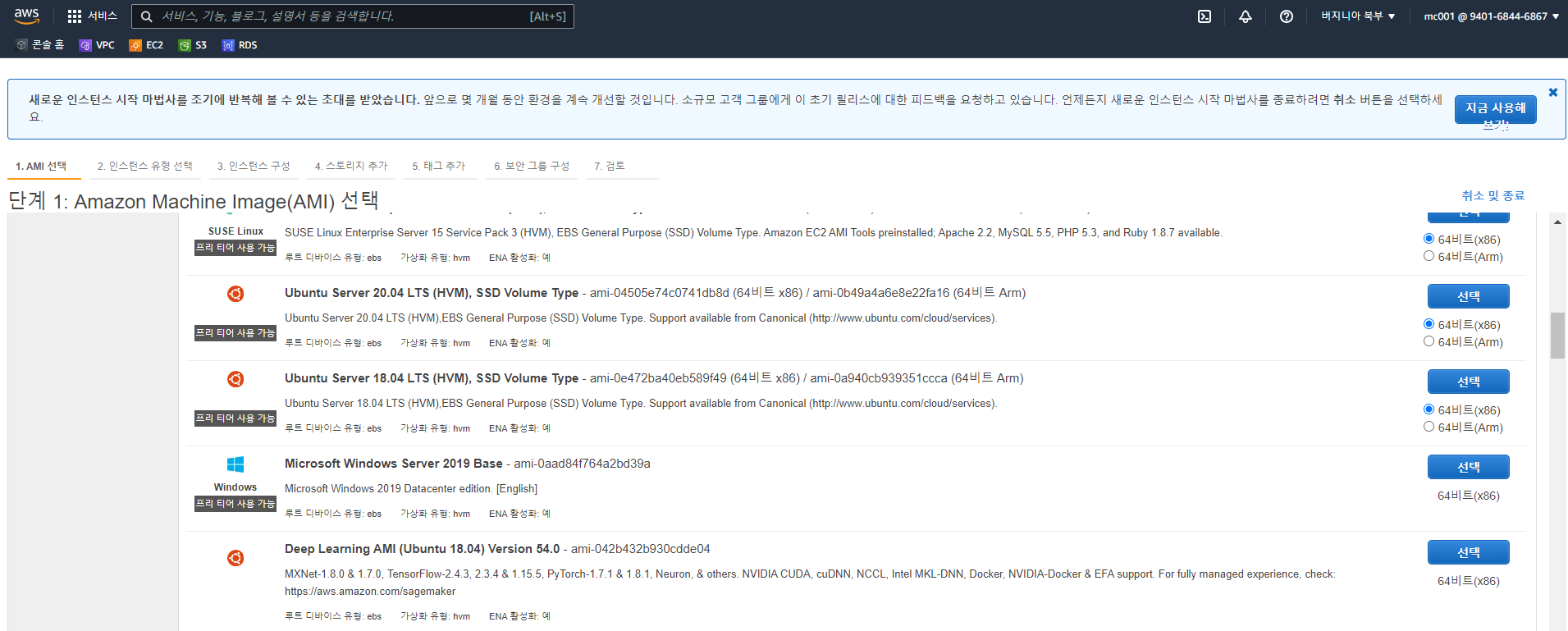
# 3. EC2 생성

## 인스턴스 생성하기

목차 인스턴스 클릭 후 우측 상단 인스턴스 시작 클릭



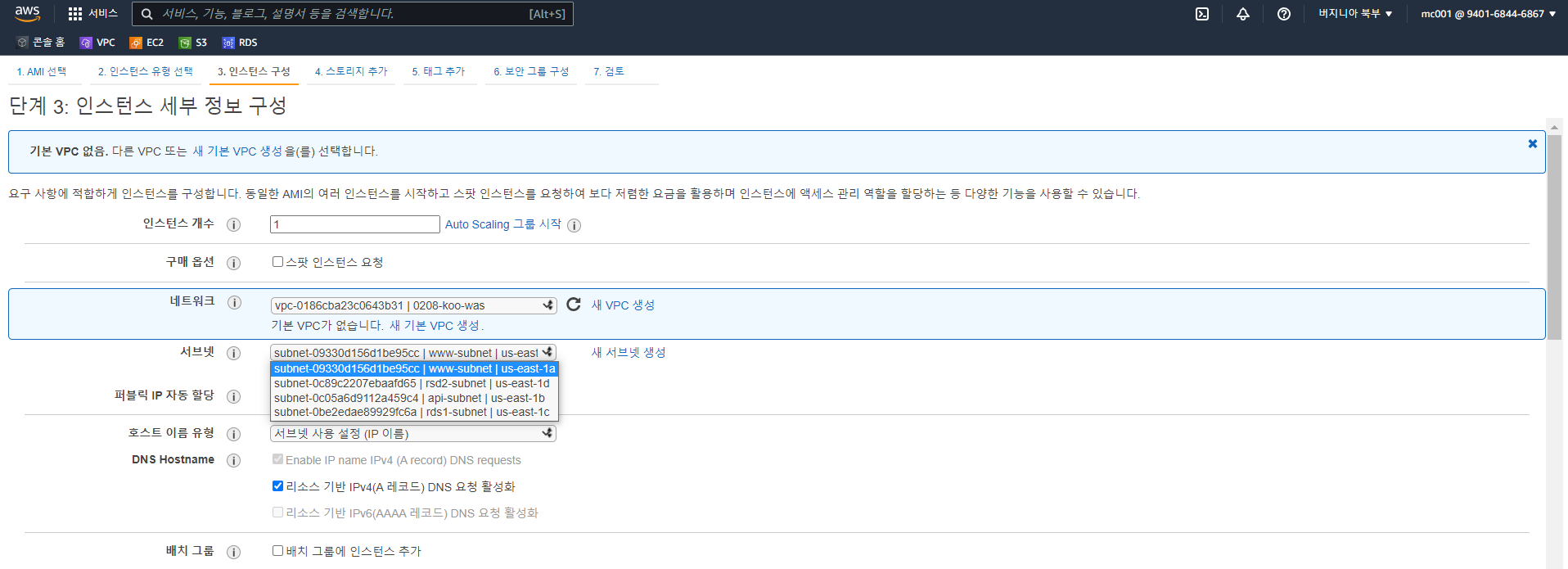
EC2 기본 화면 > 인스턴스 시작

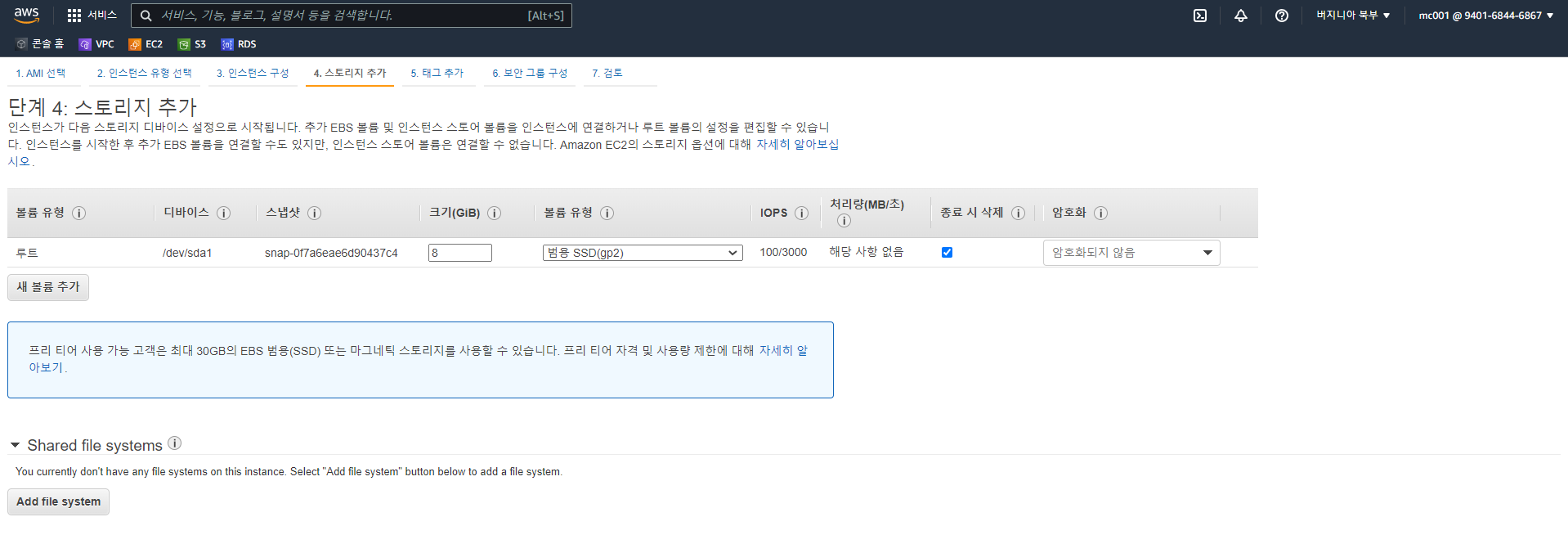


Ubuntu Server 20.04 LTS (HVM), SSD Volume Type > 64비트(x86) 선택

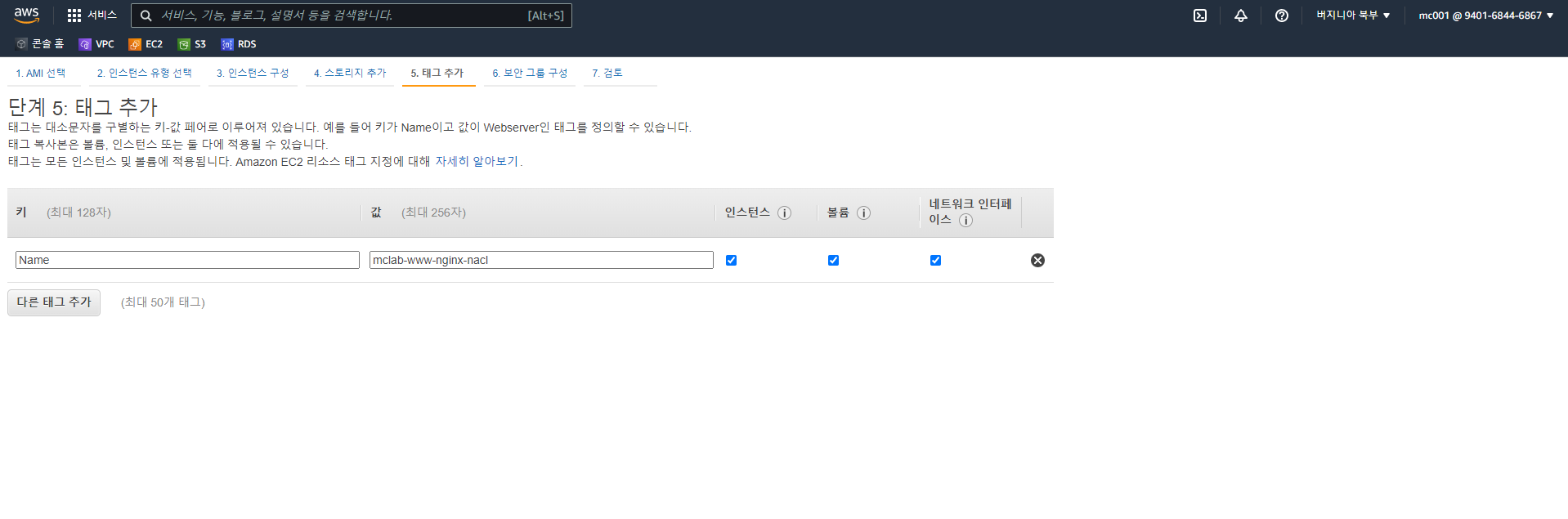


t2.micro 선택

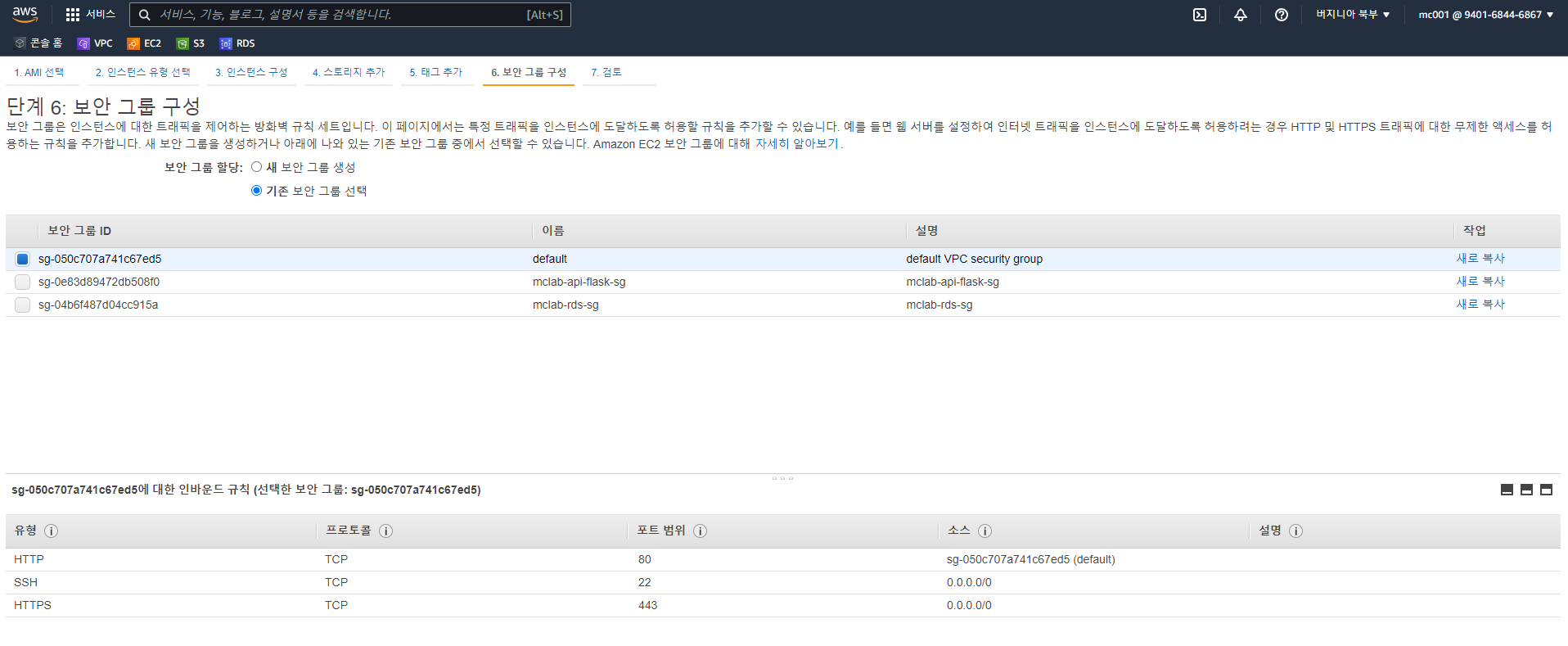


서브넷에서 웹 서브넷(예 www 서브넷)으로 선택

기본 8GiB

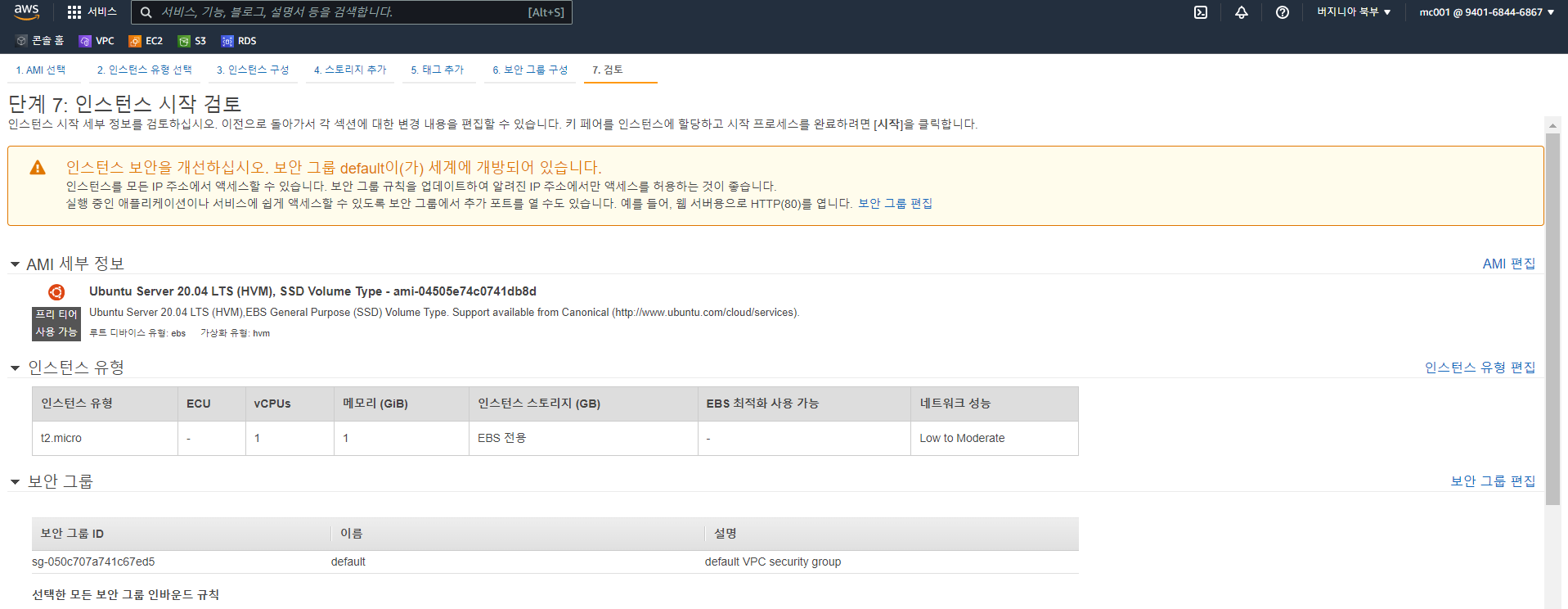


태그 추가 > 키 [Name] > 값 [mclab-www-nginx]

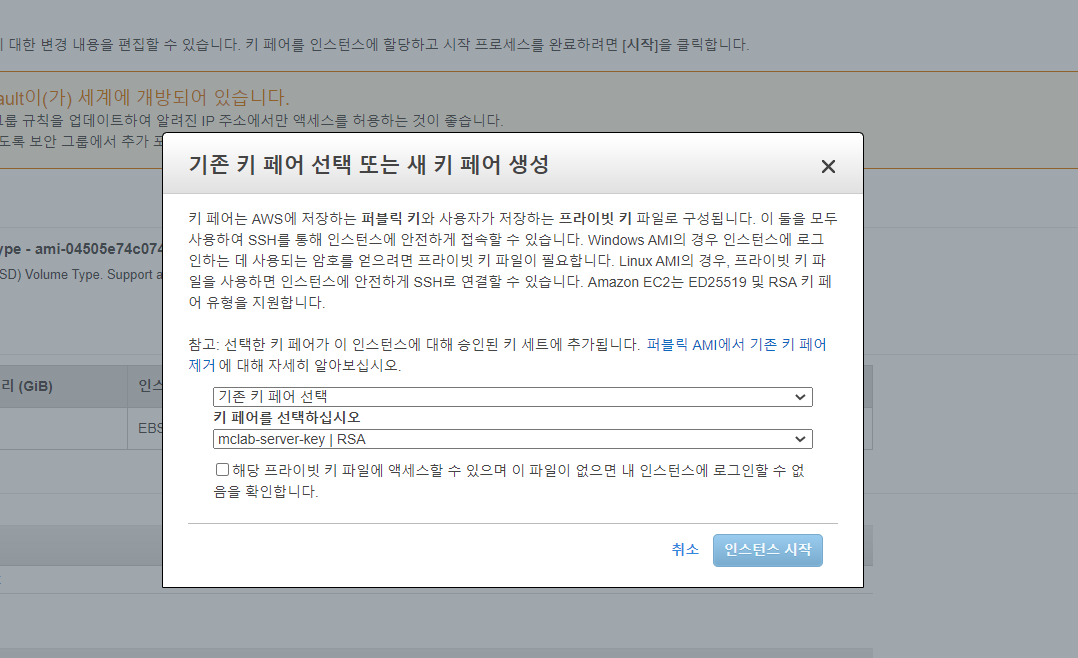


단계 6 > 새 보안 그룹 생성에서 기존 보안그룹 선택 체크

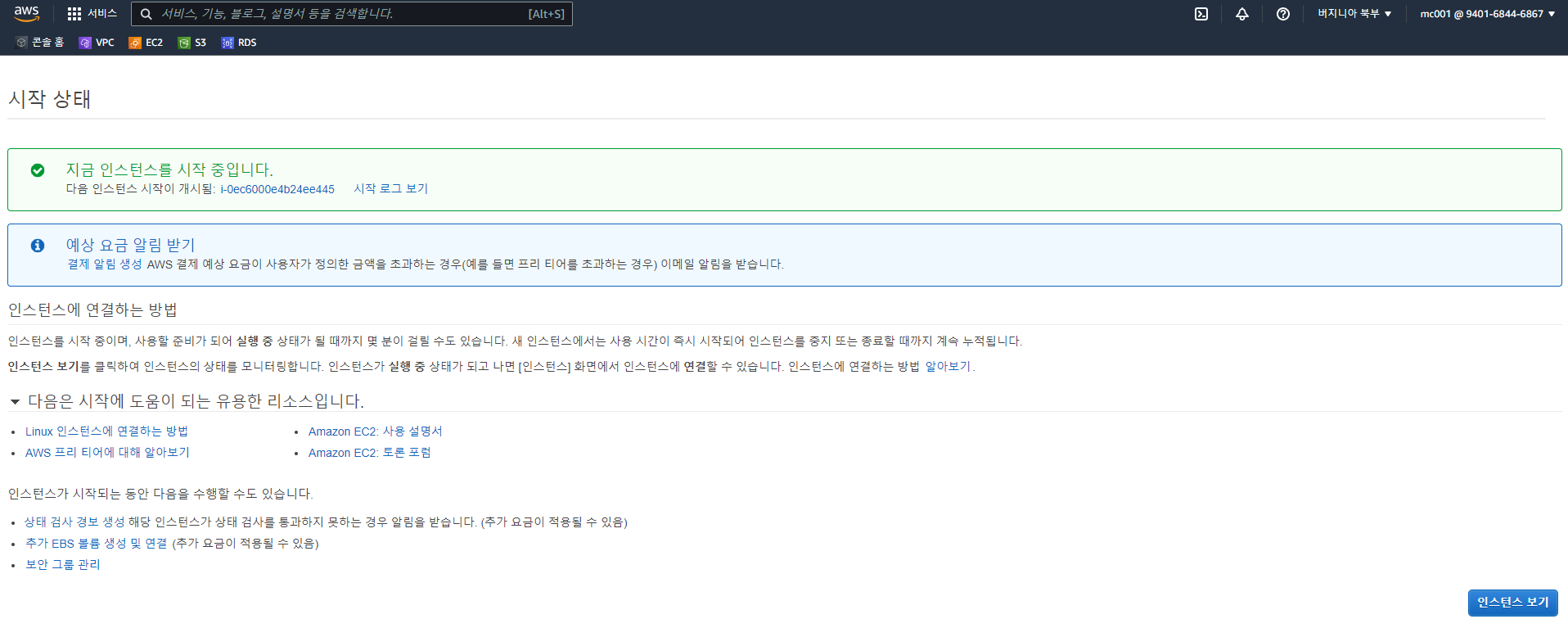
그 중 웹 보안그룹(defalt) 선택



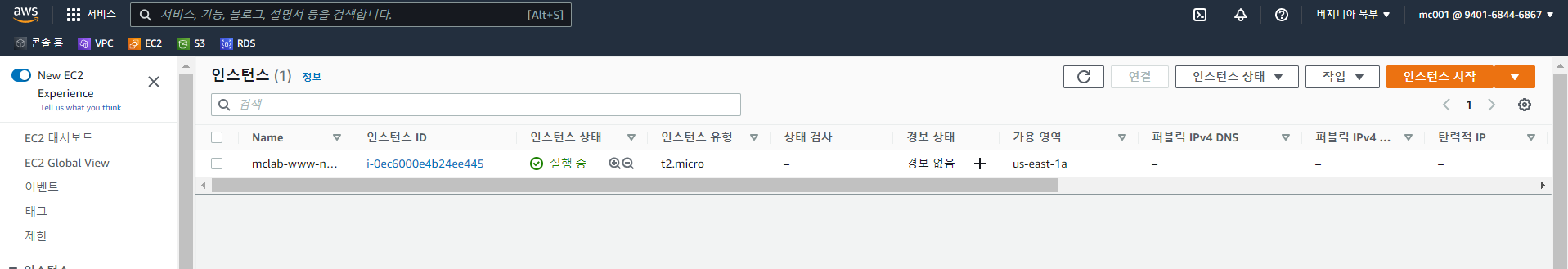
단계 7 > 설정한 값이 맞는지 확인 > 하단 시작하기 클릭



기존 키 페어 선택 또는 새 키 페어 생성을 선택하며 마지막 인스턴스 시작

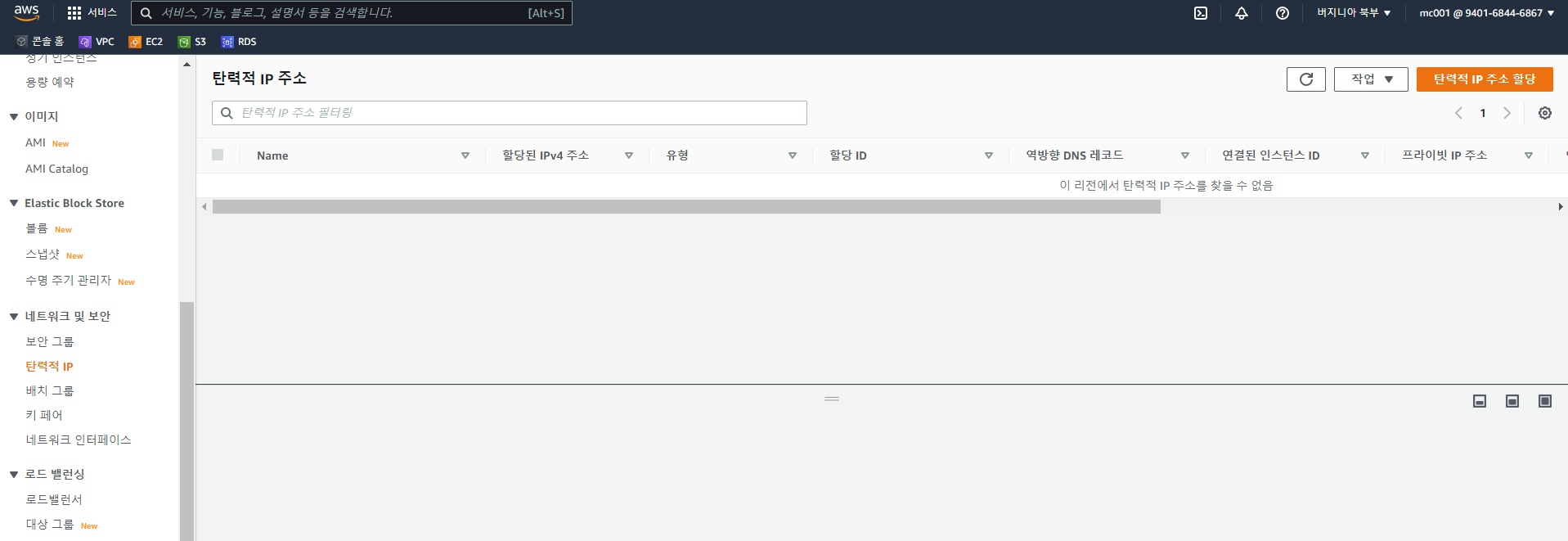


인스턴스 시작 중 확인 > 하단 인스턴스 보기 선택

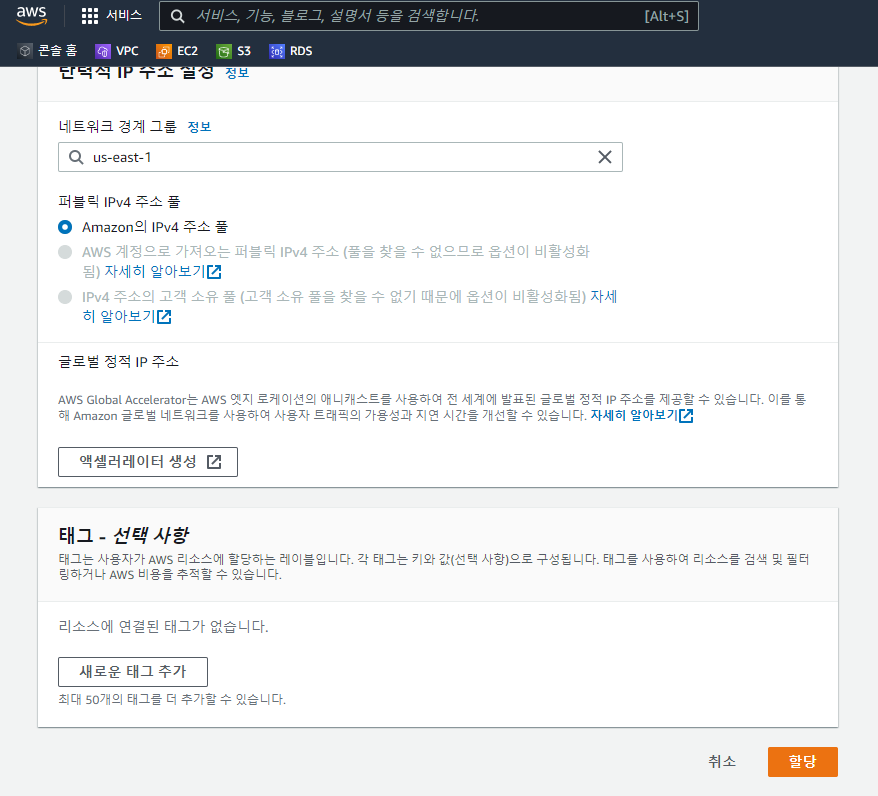


인스턴스 상태가 대기 중이 아닌 실행 중 상태 확인하면 EC2 인스턴스 생성 완료

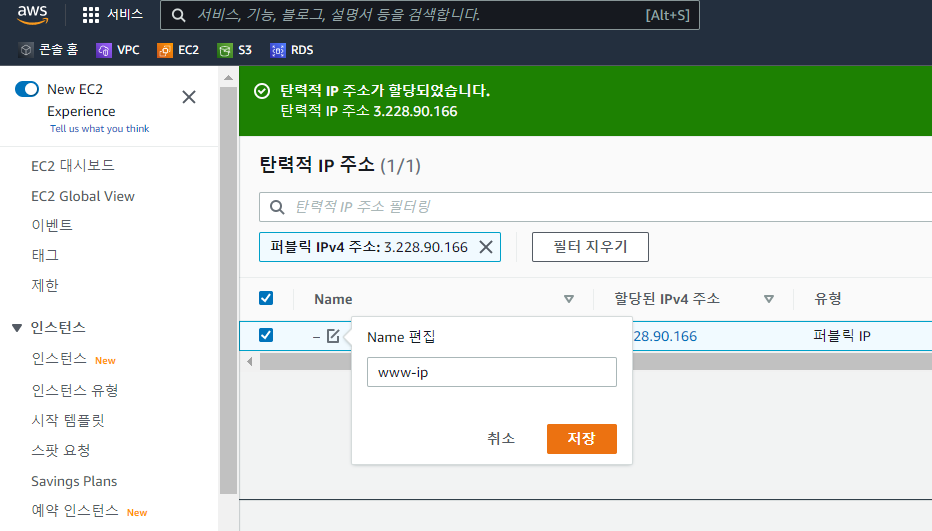
## 탄력적 IP 주소 생성하기



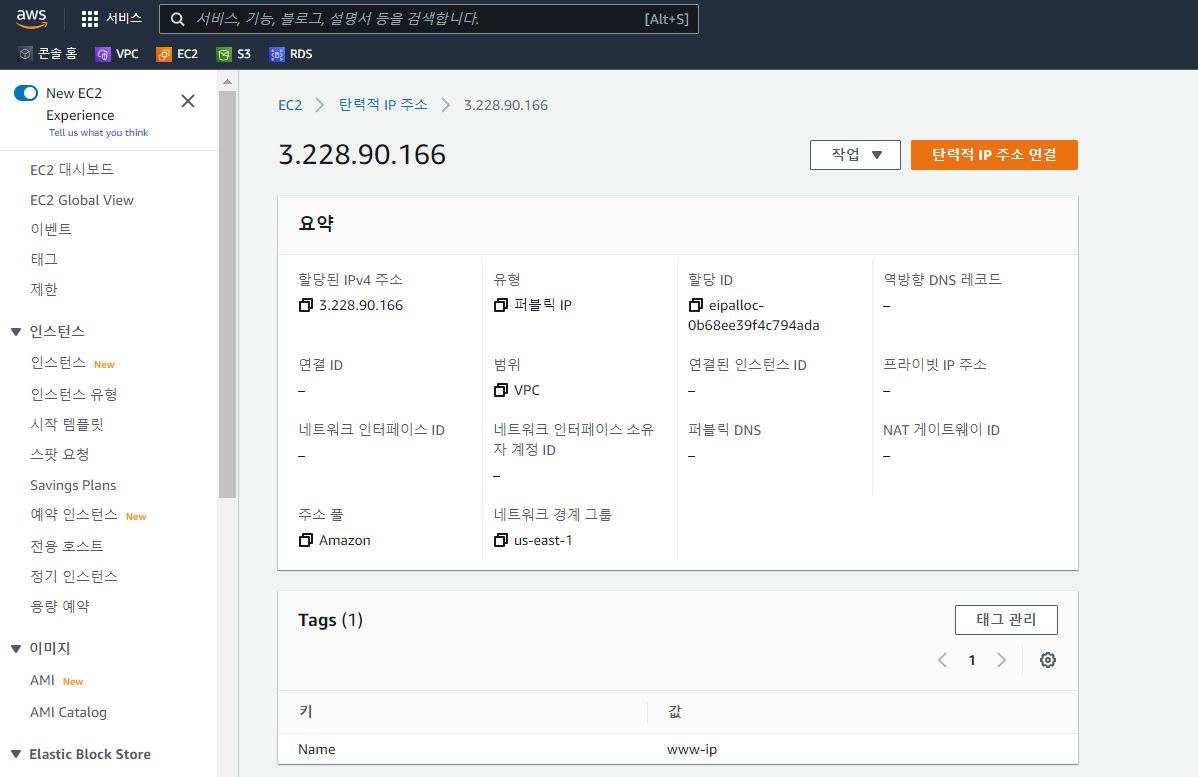
다음 탄력적 IP 주소 생성하기 > 우측 상단 탄력적IP 주소 할당 클릭



할당 클릭



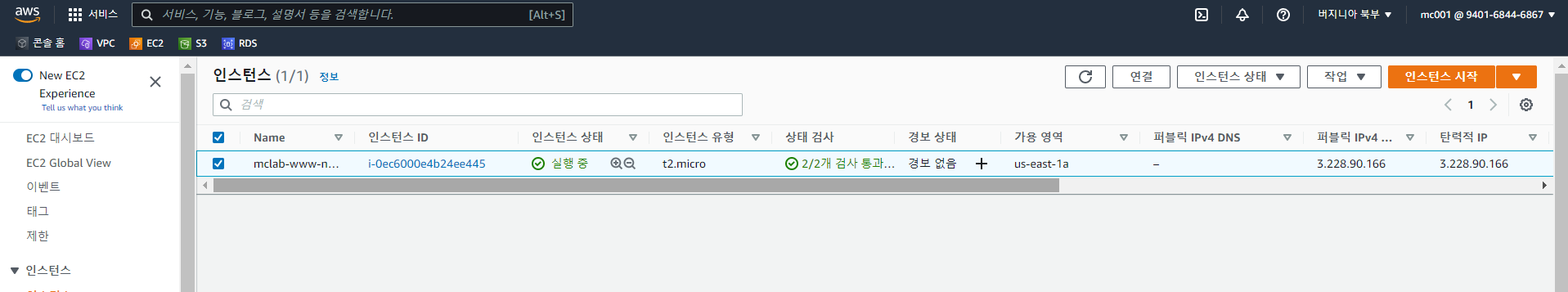
할당 확인 및 Name 편집 [www-ip]



할당 IP 클릭 후 탄력적 IP 주소 연결 클릭



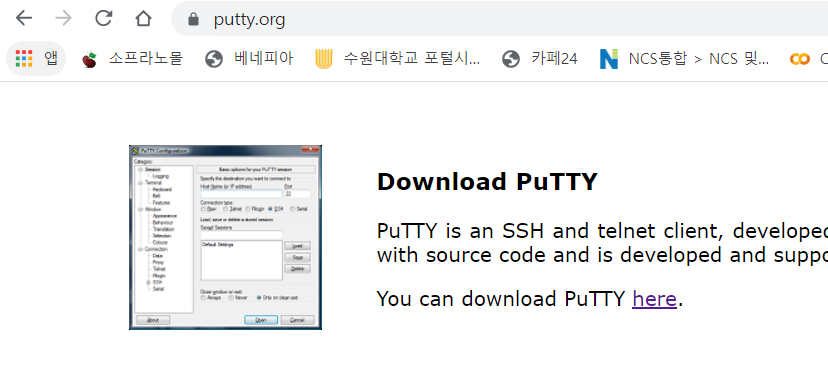
생성한 인스턴스 선택 > 생성한 프라이빗 IP 주소 선택 > 하단 연결 클릭



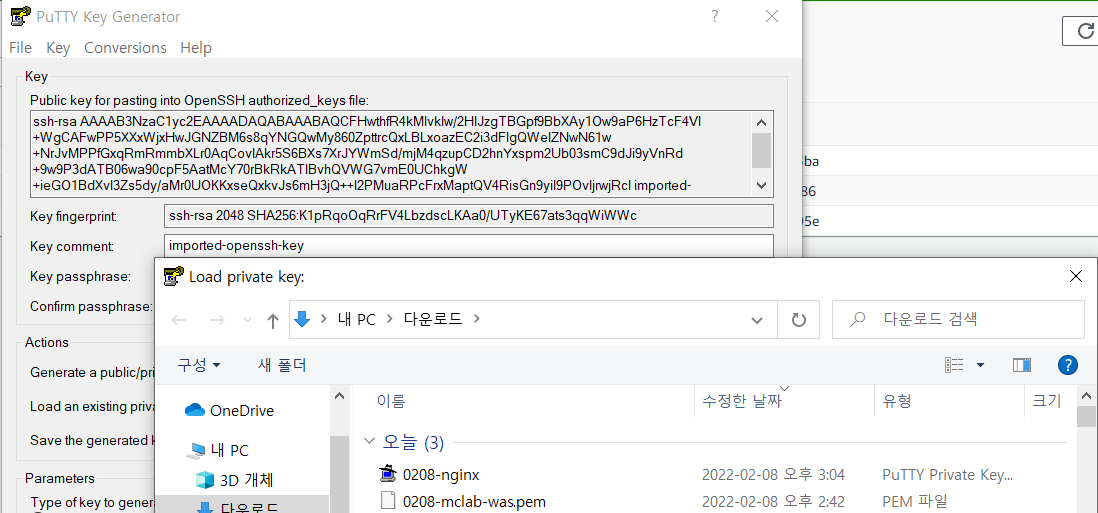
목차 인스턴스 클릭 > 생성한 인스턴스 체크 > 연결 상태 확인

# 4. PuTTy 설치 하기

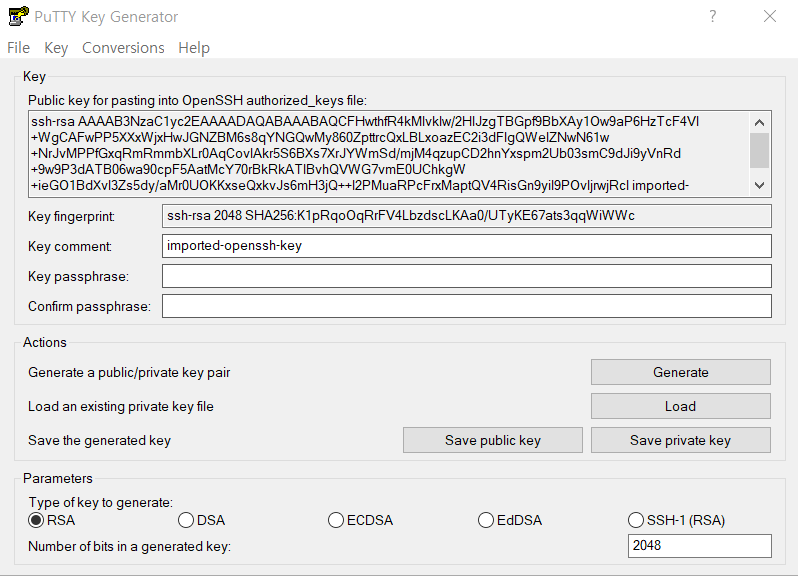
PuTTy 설치 방법

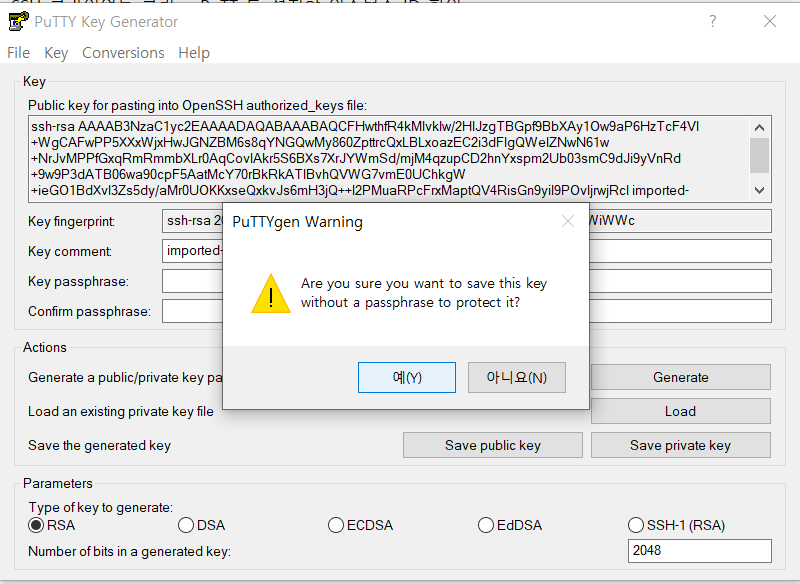


먼저 <https://putty.org/> 사이트로 가서 다운로드 받고 설치한다

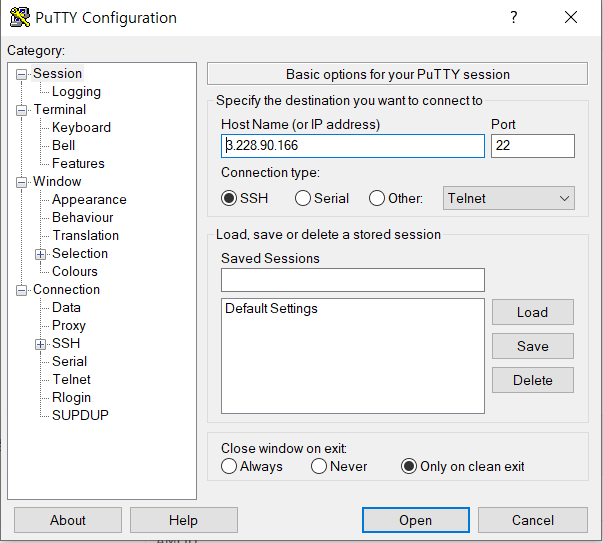


PuTTy에 연결 전 EC2에서 생성한 키를 ppk로 변형한다 > PuTTygen 앱 실행 > 상단 File 클릭 > Load private key 클릭 > EC2에서 생성한 페어키 선택

Save private key 선택



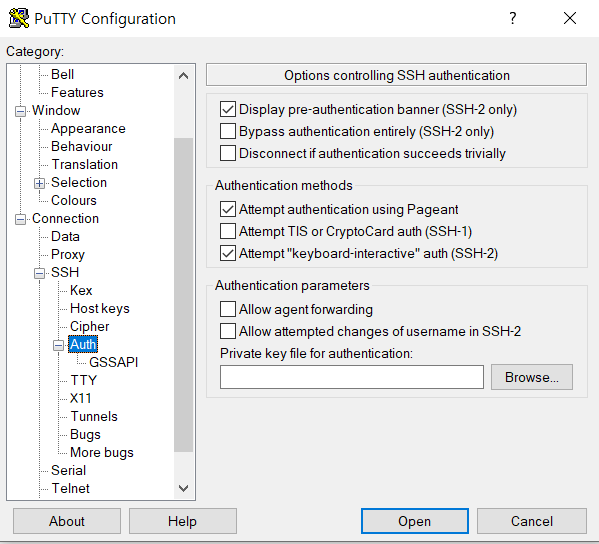
예 클릭 후 이름 설정 하여 저장



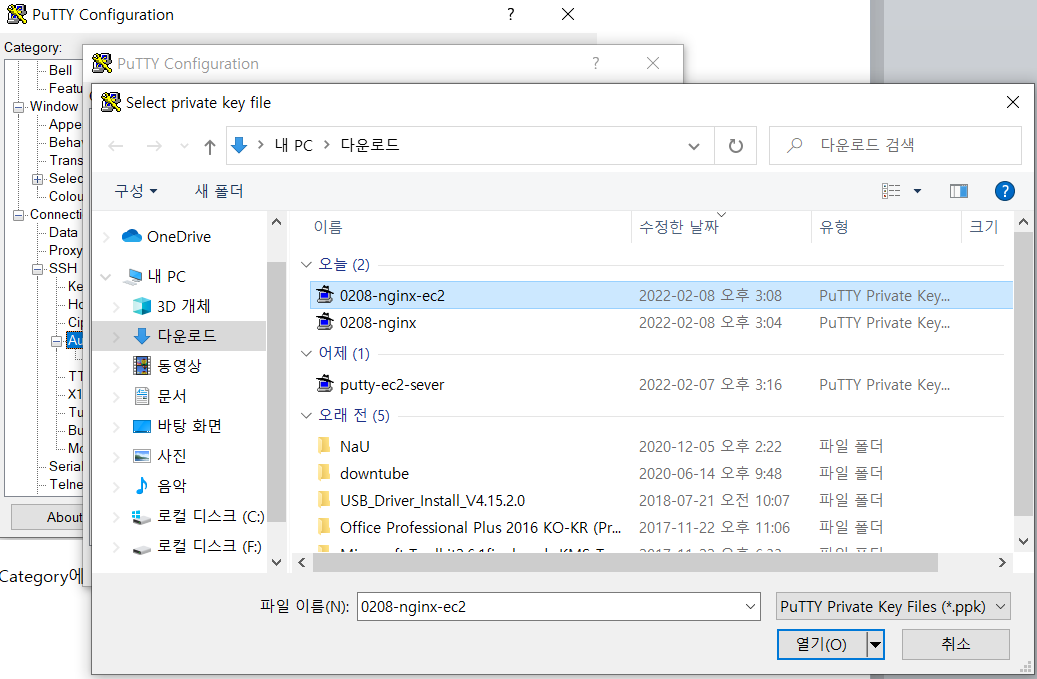
PuTTy 앱 열기 > Host Name 에 공인 IP 넣기



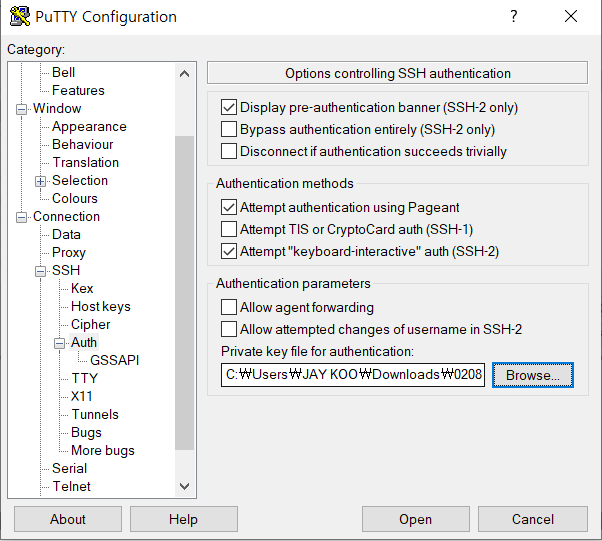
인스턴스에서 퍼블릭 IP가 공인 IP



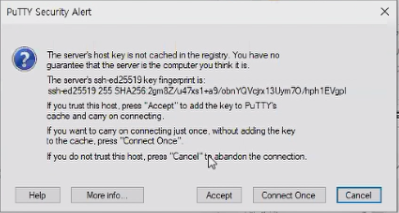
Category에서 SSH > Auth 클릭 후 왼쪽 Browse.. 클릭



변형한 ppk를 연다



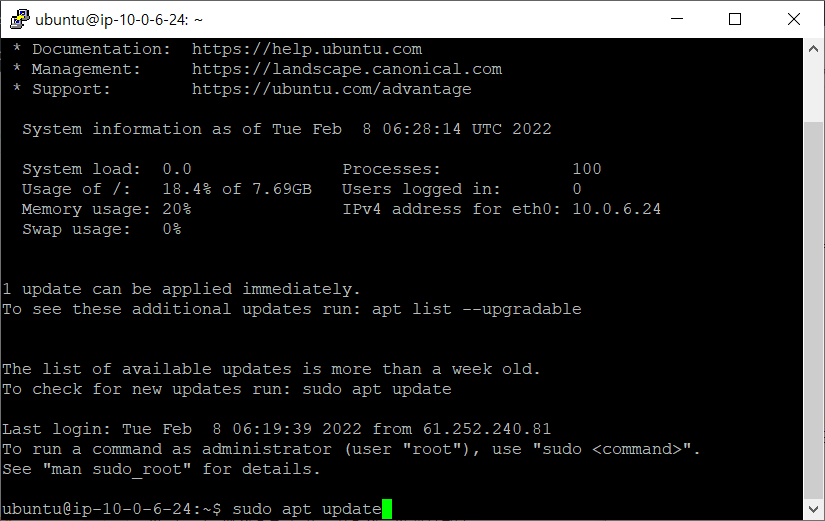
하단 Open 클릭



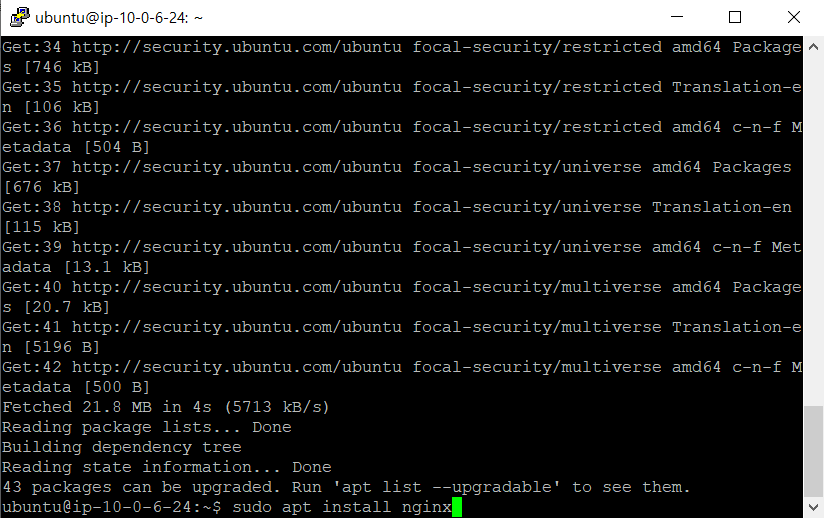
맞으면 Accept 클릭



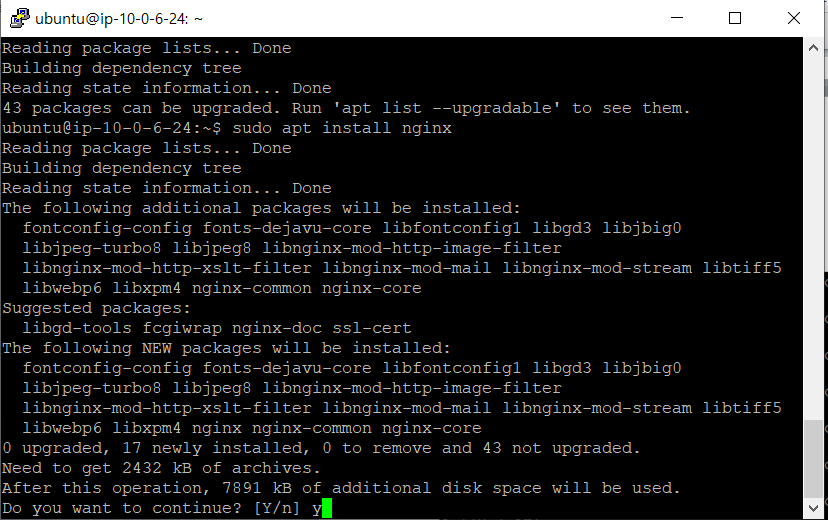
login as 에 ubuntu 입력 후 엔터



sudo apt update 입력 후 엔터



sudo apt install nginx 입력 후 엔터

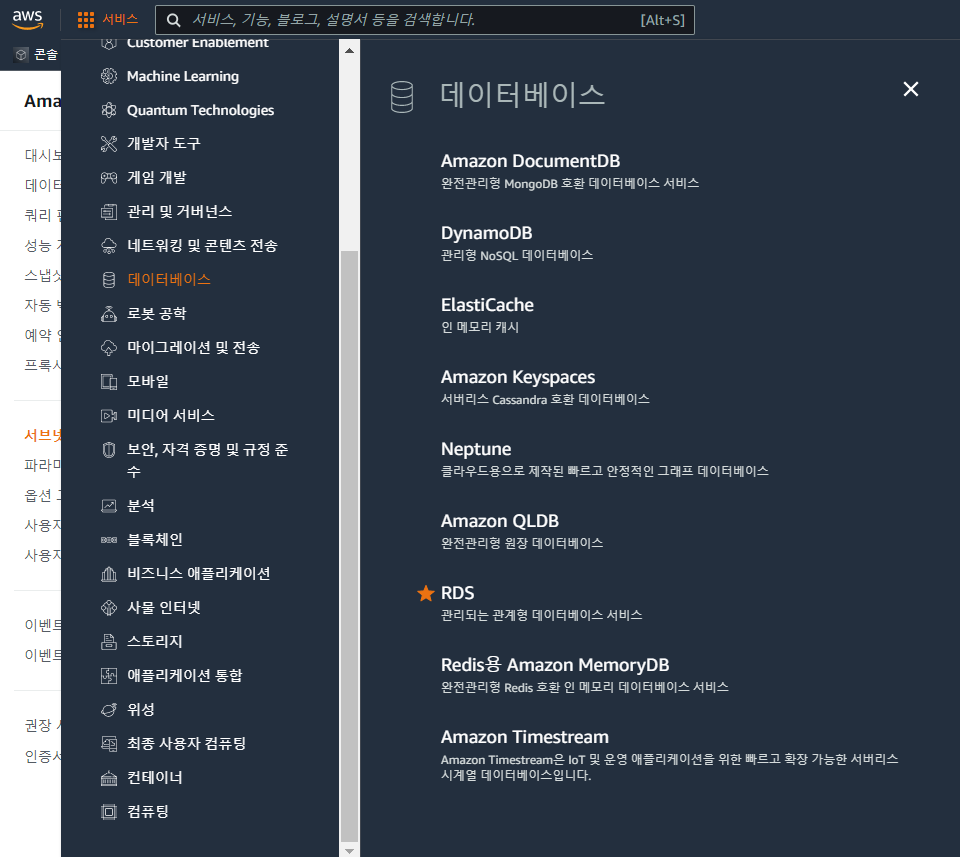


continue? > y 입력 후 엔터

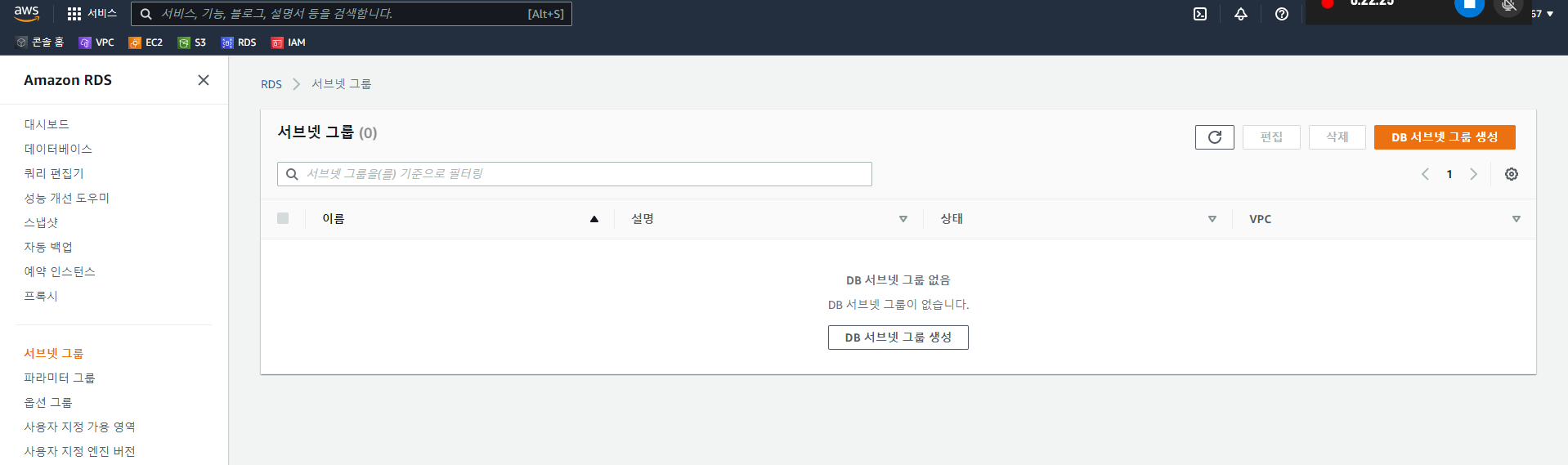
# 5. RDS 생성하기

## DB 서브넷 그룹 생성

해당 DB 서브넷을 그룹으로 묶는다



서비스 > 데이터베이스 > RDS 클릭



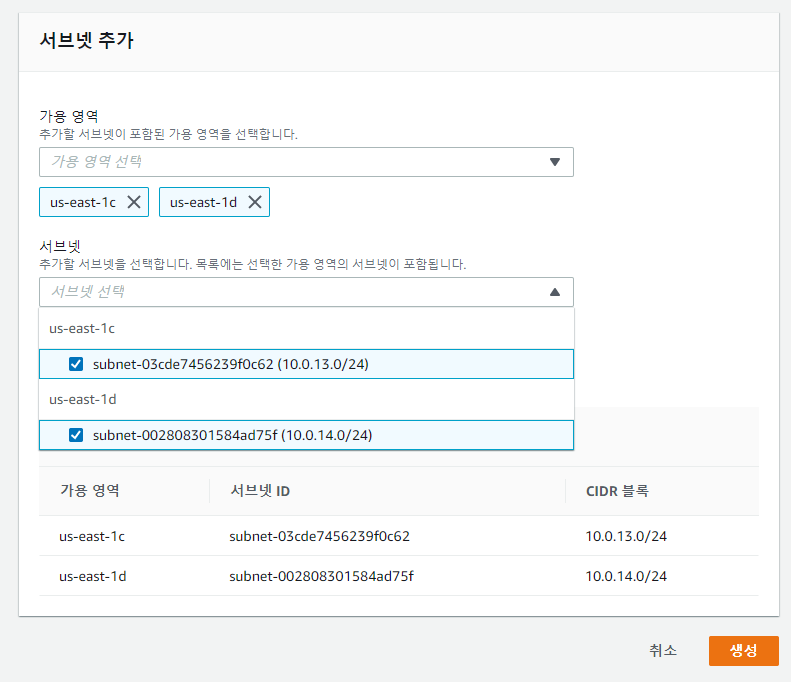
목차 서브넷 그룹 클릭 > 우측 상단 또는 가운데 DB 서브넷 그룹 생성 클릭



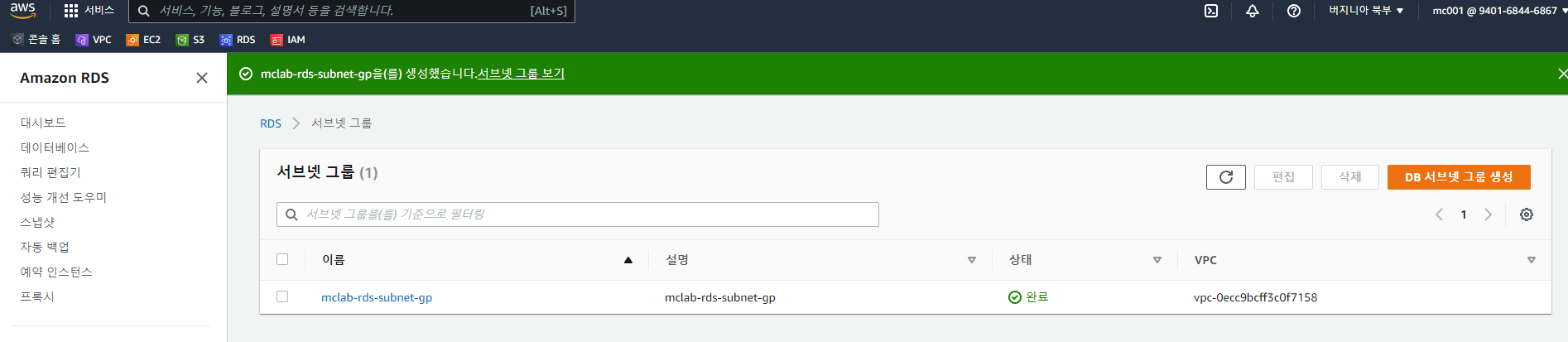
이름 설정 > 설명 이름 복붙 > VPC 선택



스크롤내려서 서브넷 추가 > 가용 영역 선택 – DB 서브넷으로 선택한 가용 영역 체크



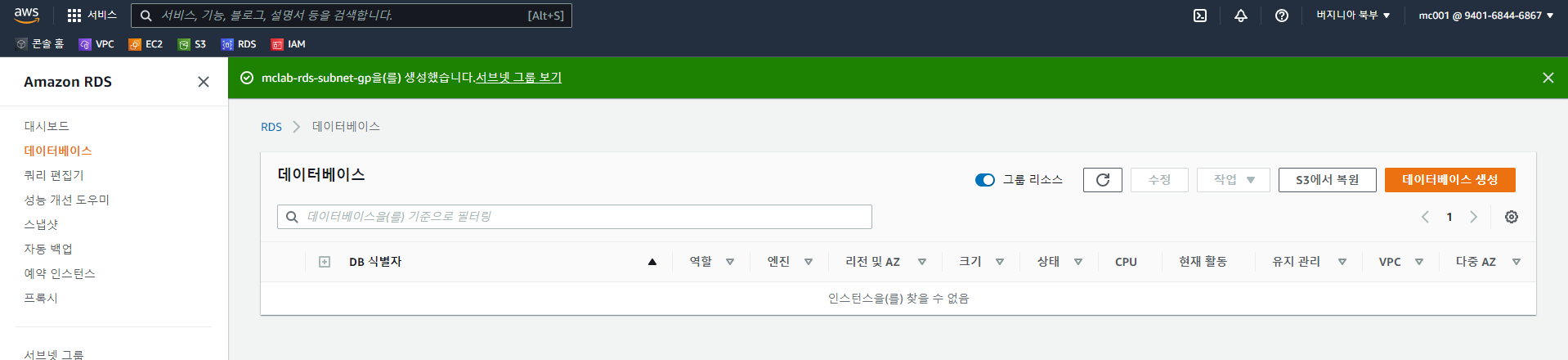
서브넷 선택 – 가용 영역을 제대로 선택하면 생성한 DB 서브넷이 나옴 > 마지막 생성 클릭



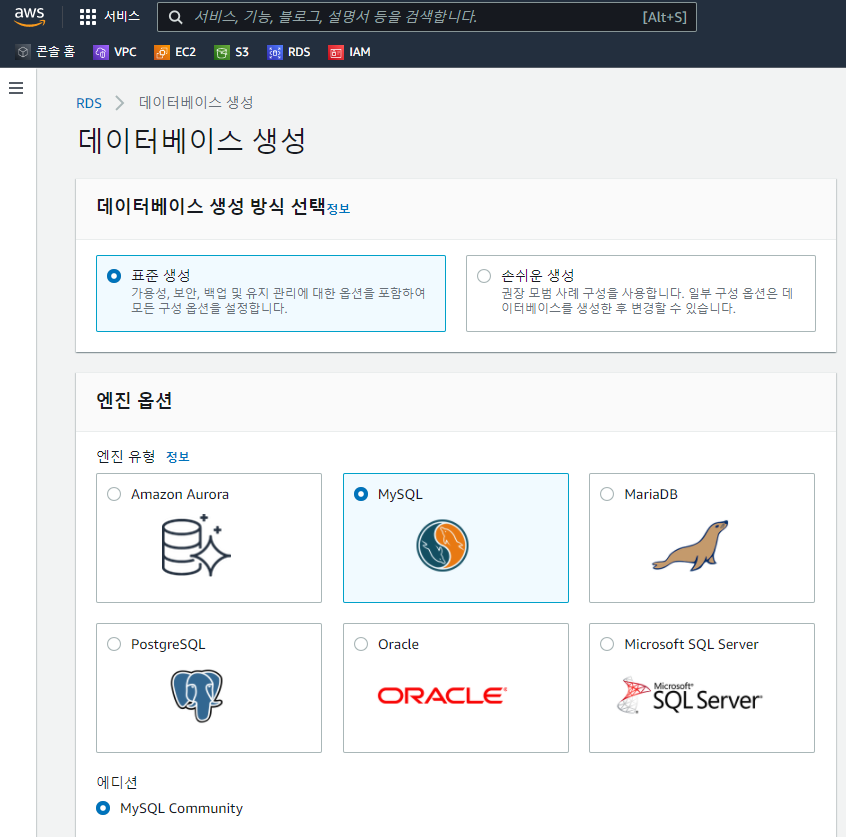
DB 서브넷 그룹 생성 완료

## 데이터베이스 생성

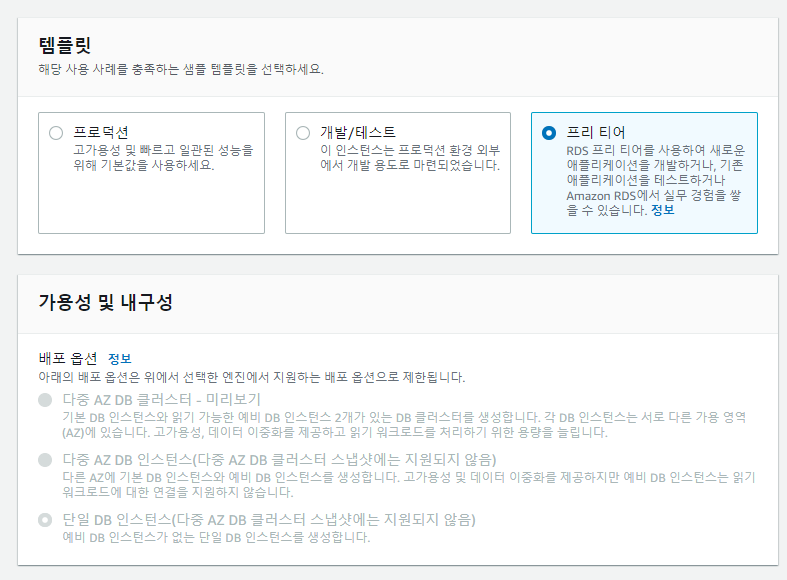
MaridDB 또는 MySQL등 원하는 데이터베이스를 선택해서 생성한다



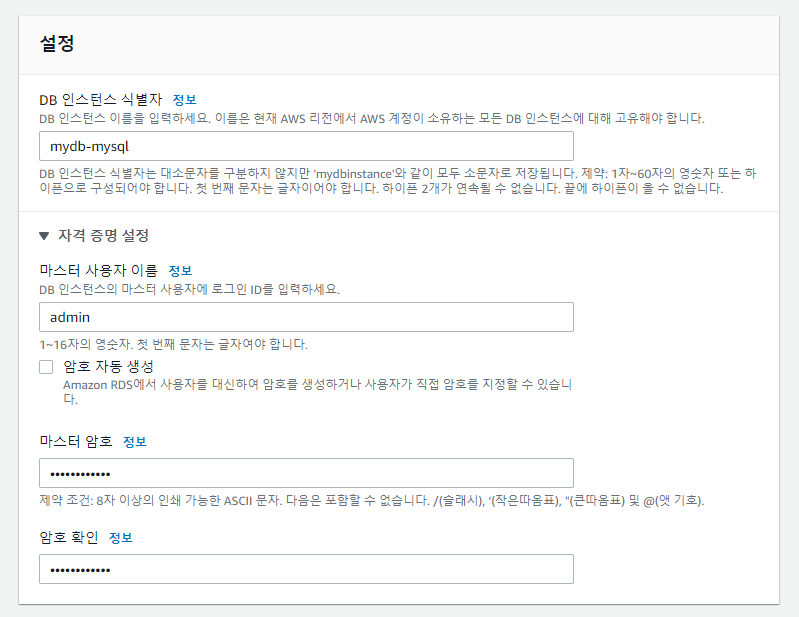
목차 데이터베이스 클릭 > 우측 상단 데이터베이스 생성 클릭



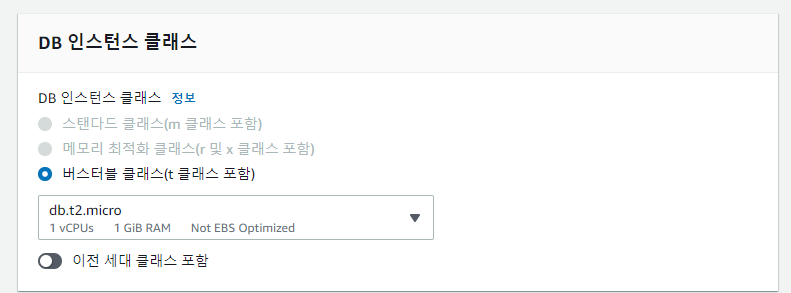
표준 생성 선택 > 엔진 옵션 - MySQL 선택



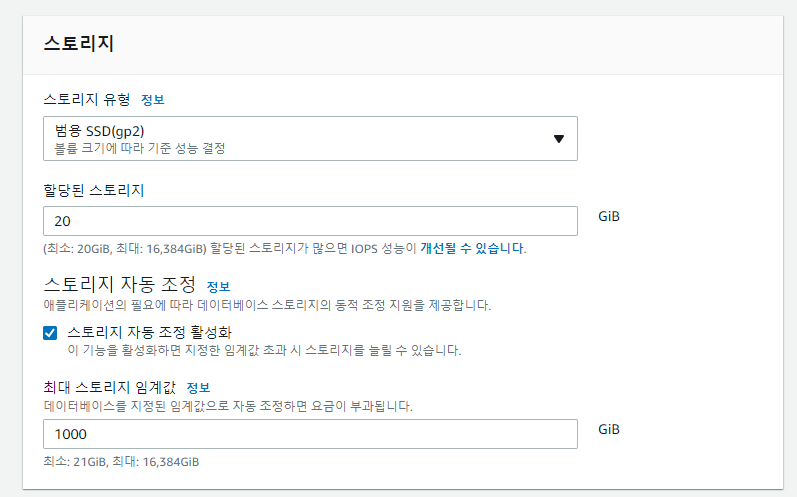
템플릿 - 프리 티어로 선택



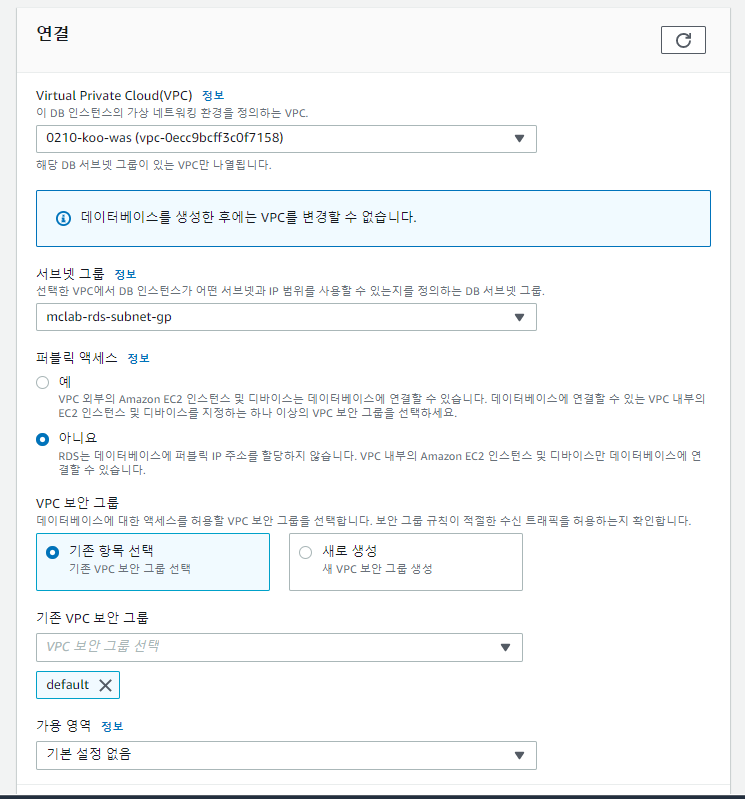
설정 – 이름 입력 > 마스터 암호 입력



DB 인스턴스 클래스 - 프리티어로 생성했으니 micro t2인지 확인



스토리지 – 기본 설정

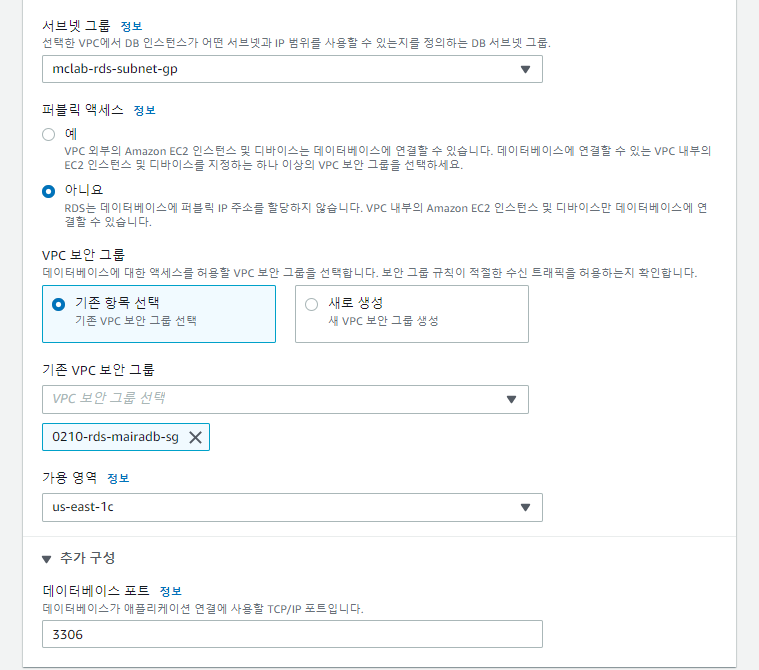


연결 – 퍼블릭 엑세스 : 아니요 선택

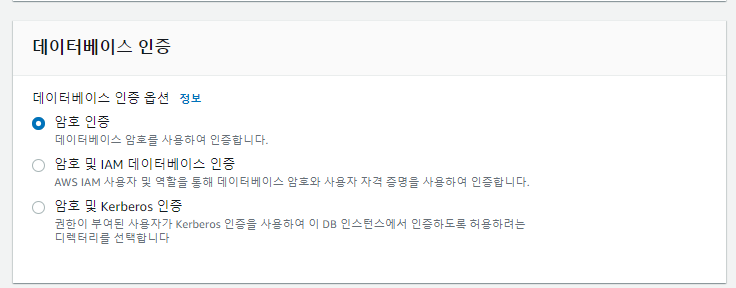
**※ 현재 내부 접속만 가능해서 액세스를 예로 선택해서 생성되지 않는다(이유는 퍼블릭이 아닌 프라이빗 서브넷이기 때문)**

**이후 웹 서버로 외부 접속도 가능하게 만들어보자**

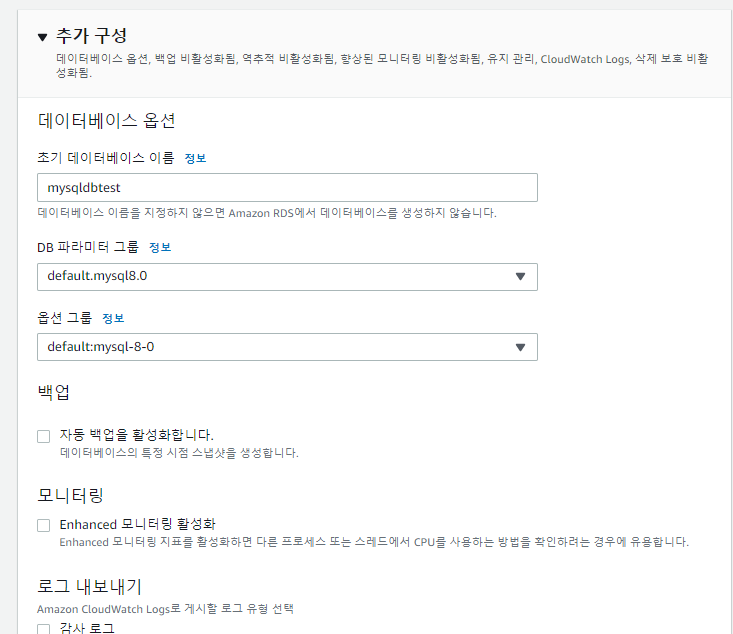
> 기본 VPC보안 그룹 해지(default 설정 해지)



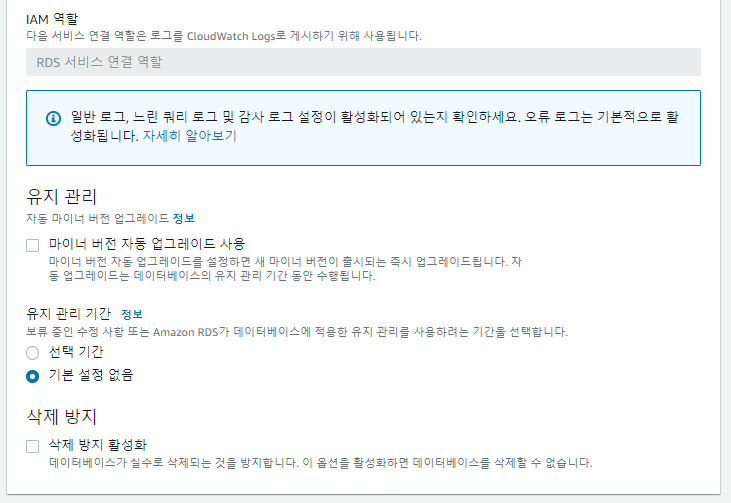
기존 VPC보안 그룹 : RDS SG(보안그룹)으로 선택 > 가용 영역 : 1c 선택(API 서브넷과 먼저 연결될 DB서브넷의 가용 영역으로 선택 할 것) > 추가 구성을 클릭해서 포트도 확인 할 것



데이터베이스 인증 – 기본 설정(암호 인증 선택)

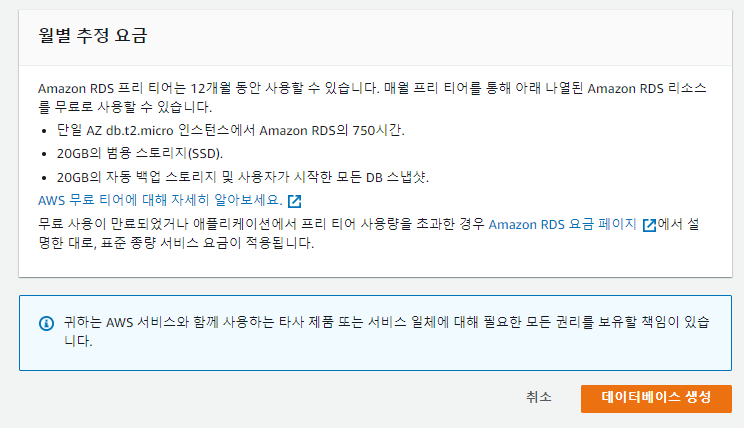


추가 구성 – 초기 데이터베이스 이름 입력 > 백업 선택 해지!!

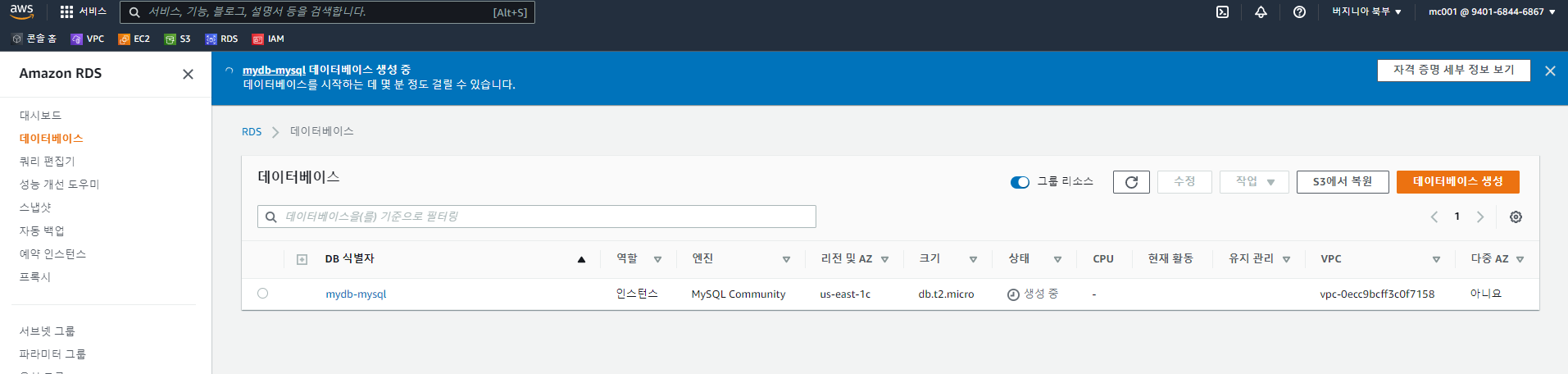


유지 관리 자동 업그레이드 사용 체크 해지!!

~모든 백업과 유지 관리 체크 해지



데이터베이스 생성 선택



생성 중 > 생성 완료 확인