

19.02.21 AWS 사용 / 리눅스 DB 설정 / oauth 관리

노트북: [TIL-MY]

만든 날짜: 2019-03-22 오후 4:56

수정한 날짜: 2019-03-27 오전 8:55

작성자: 황인범

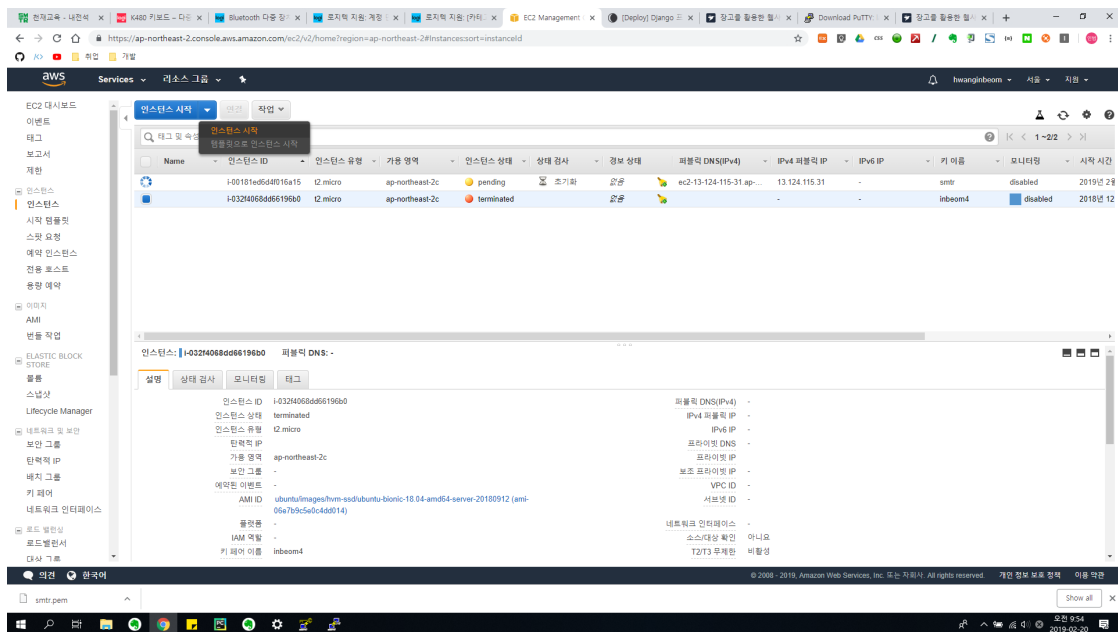
URL: <http://blog.freezner.com/archives/925>

1.gunicorn ☐

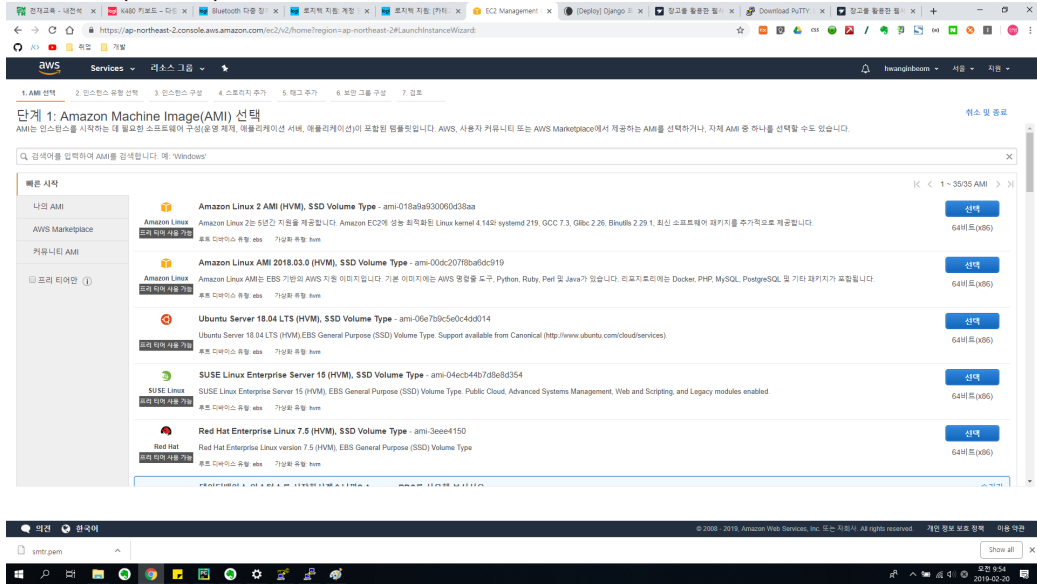
2.nginex ☐

3.AWS에 올리기 ☐

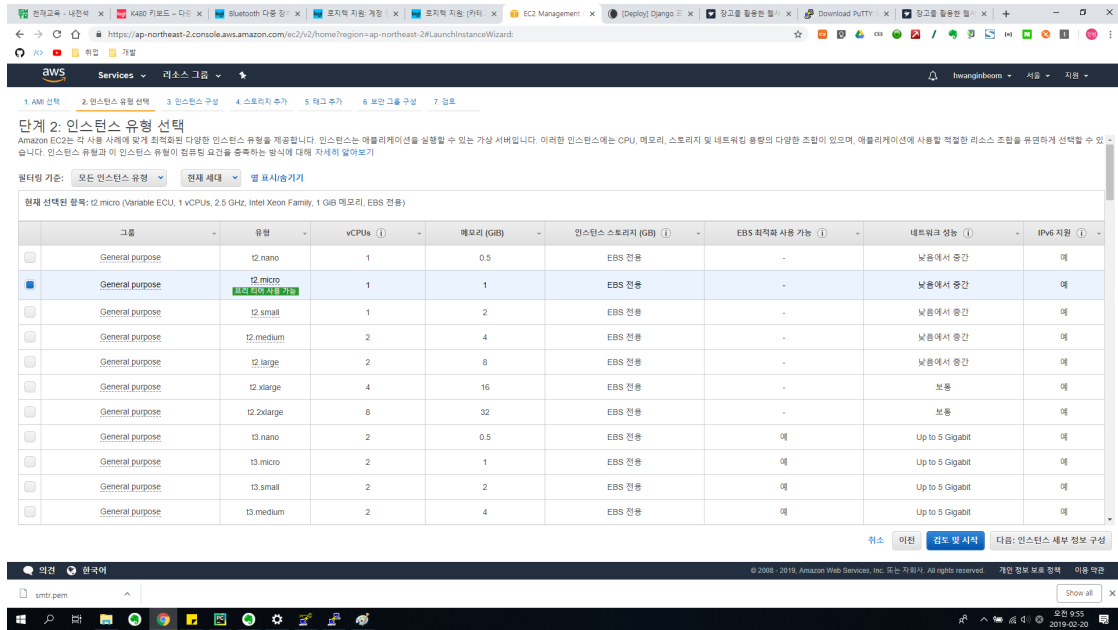
1. AWS 가입 후 인스턴스 시작



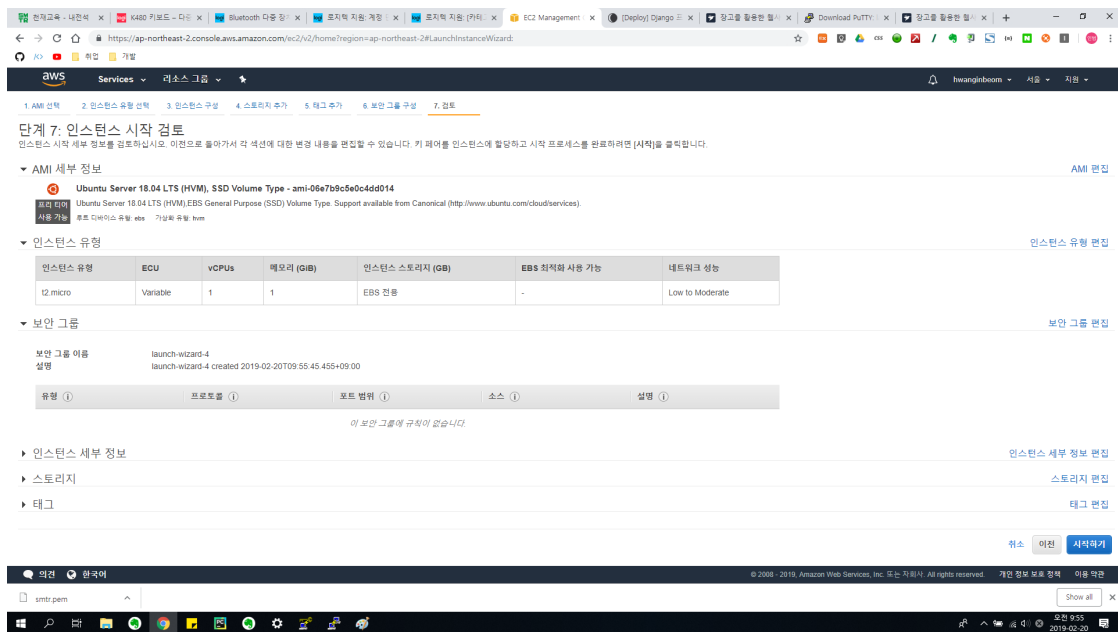
2. AMI 선택 (Ubuntu)



3. 기본 프리티어 사용가능으로 설정

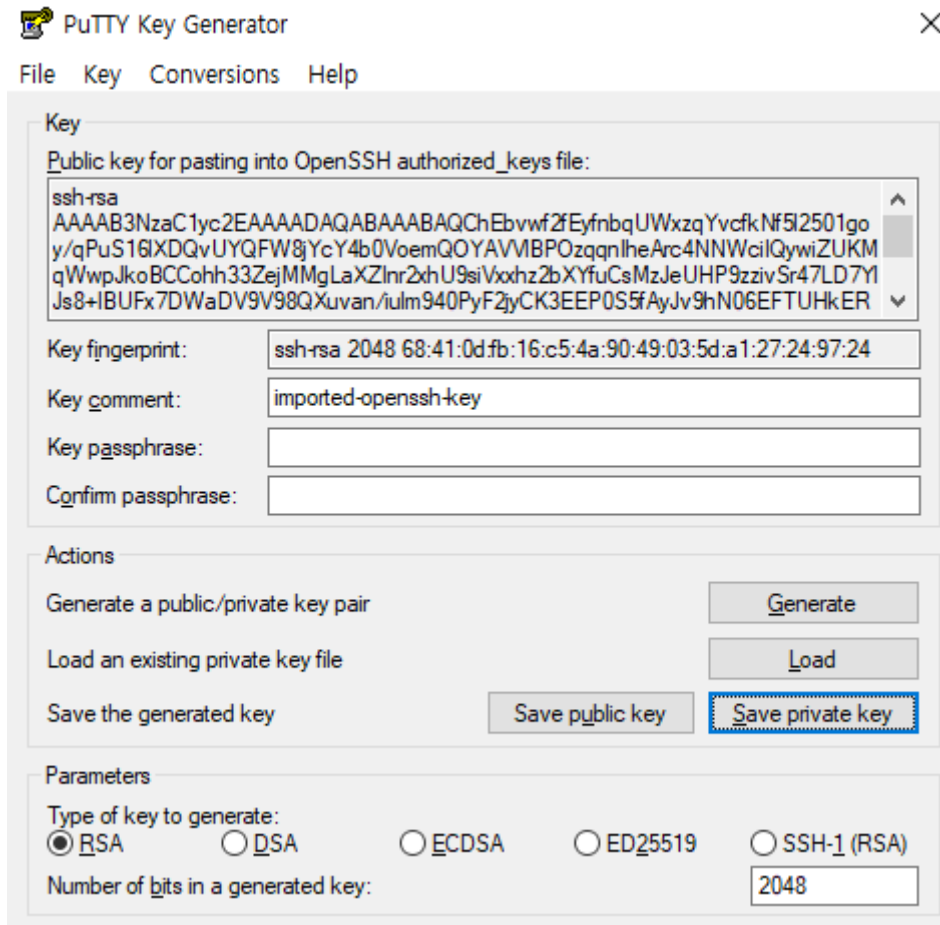


4. 시작하고 키페어 다운로드 받기 (이걸 통해 접속을 할 수 있다)



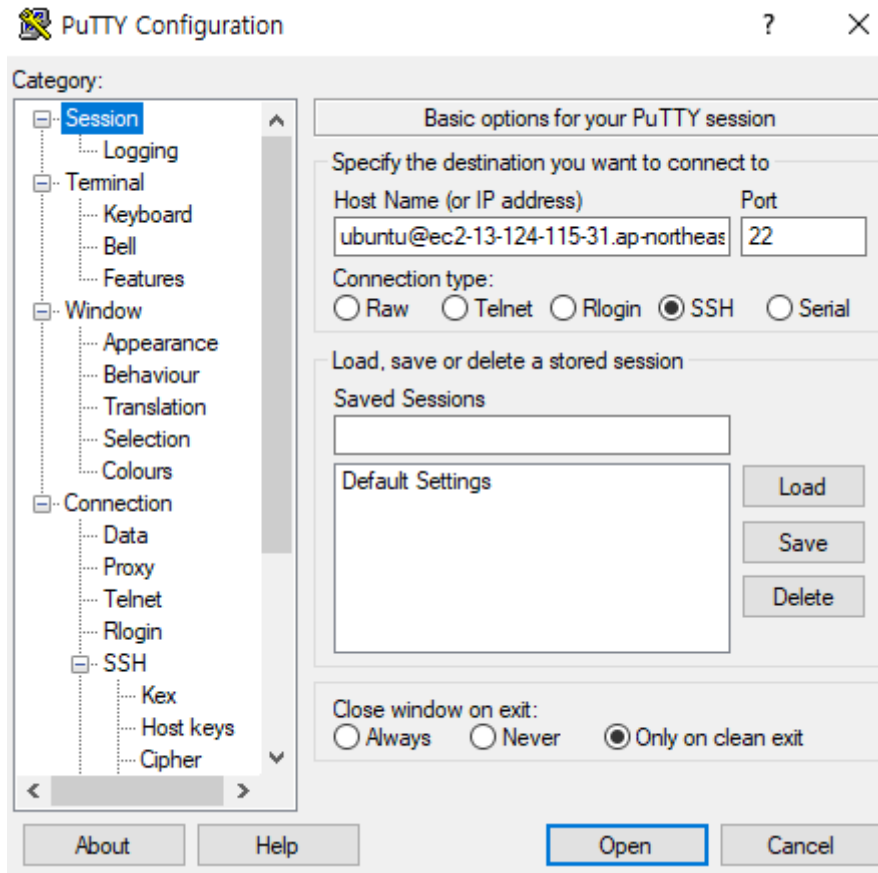
5. putty / puttygen 설치

6. putty gen 을 사용하여 load 하고 해당 pem 파일을 가져온다. 그 후 save 를 하는데
 ???pem.ppk 파일로 만들어 확장자를 바꿔준다 ,

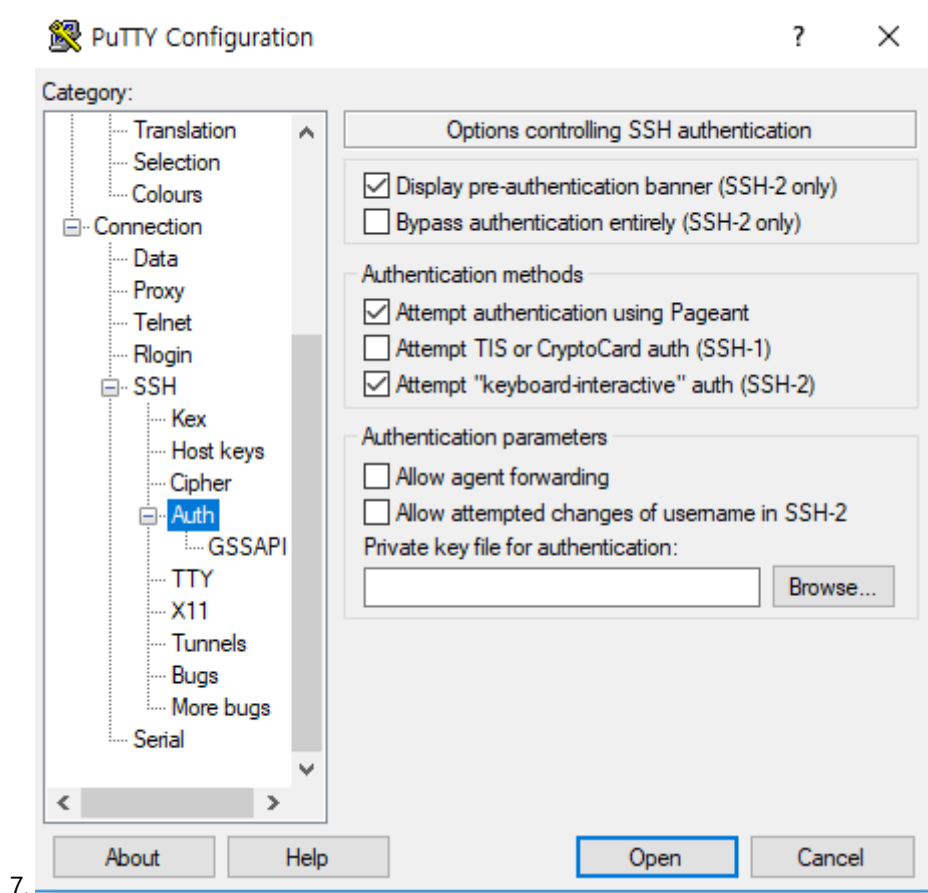


7. putty 설정을 해준다. 앞에는 기본적으로 받는 아이디 (ubunut 일경우에는 ubuntu로 받게
 된다.)를 적고 ubuntu@ ec2-13-124-115-31.ap-northeast-

2.compute.amazonaws.com (AWS에서 주는 퍼블리 DNS 주소를 넣어준다.)



8. 그 후 이부분에서 putty gen을 통해 만들었던 ppk 파일을 로드한다.



7.

9. 비트버킷에 있는 소스를 clone한다.

10. 가상환경에 들어가 설치를 해준다. `source myvenv/bin/activate`

1. 통신까지의 설정

```
0.sudo apt-get update
$ python3 -m venv smtr_venv
```

1. `$ source smtr_venv/bin/activate`

2. `(myvenv) ~$ python3 -m pip install --upgrade pip`

3. `(myvenv) ~$ pip install django`

4. `(myvenv) ~/djangogirls$ django-admin startproject smtrsite .`

5. `(myvenv) pip install django`

6. (myvenv) ~/djangogirls\$ python manage.py migrate
7. ps -ef | grep python (어떤게 켜져있는지 확인)
8. netstat -anp | grep 8000 (8000포트에 모가있는지 확인)
9. setting에 들어가 allow host를 [*]로 바꿔준다. - 장고(django)에서 자체적으로 내릴 경우에도 이 경우다
10. window에서 tcping ec2-13-125-151-40.ap-northeast-2.compute.amazonaws.com 8000 / 퍼블릭 DNS와 포트를 통해 통신이 되는지 확인한다.
11. 이게 된다면 url에 ec2-13-125-151-40.ap-northeast-2.compute.amazonaws.com:8000을 입력한다. - 이상 통신끝

11. 이 다음에 소스를 옮겨주는 작업을 한다 !

12. mysql 설치

```
~$ sudo apt-get install mysql-server
```

MySQL 접속

- u(username): root
- p(password): 설치할 때 입력한 비밀번호

```
~$ mysql -u root -p
Enter password:
```

- 안되면 그냥 sudo mysql 로 접속한다.

create database smtr_dev

```
sudo mysql -u root
DROP USER 'root'@'localhost';
CREATE USER 'root'@'%' IDENTIFIED BY '';
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%';
FLUSH PRIVILEGES;

3번 python manage.py makemigrations smtrapp

4번 python manage.py migrate

5번 데이터 넣기

view 에서 api로 탄다음에 미들웨어로 탈 일이 없게끔 만든다.
```

이것도 안되면

root 비밀번호 변경

mysql 접속

```
$ mysql -u root -p
Enter password:
```

mysql database 선택

```
mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

비밀번호 변경

검색을 해보면 `update user set password=password('1234') where user='root';` 명령을 이용하면 된다고 나와있다.

그대로 복사해서 입력해보면

```
mysql> update user set password=password('1111') where user='root';
ERROR 1054 (42S22): Unknown column 'password' in 'field list'
```

`password` field 를 찾을 수 없다는 에러 메세지가 출력된다.

확인해보기 위해 user table의 field 들을 출력해봤더니

```
mysql> describe user;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Host  | char(60) | NO | PRI | | |
| User  | char(32) | NO | PRI | | |
| ...  |         |    |    |         |       |
| authentication_string | text | YES | | NULL | |
| password_expired | enum('N','Y') | NO | | N | |
| password_last_changed | timestamp | YES | | NULL | |
| password_lifetime | smallint(5) unsigned | YES | | NULL | |
| account_locked | enum('N','Y') | NO | | N | |
+-----+-----+-----+-----+-----+-----+
45 rows in set (0.00 sec)
```

`password` field를 찾을 수 없었다. 대신 `authentication_string` 이라는 field가 존재한다.

위의 명령에서 `password` 대신 `authentication_string`을 업데이트하는 것으로 수정하여 실행하면 제대로 동작한다.

```
mysql> update user set authentication_string=password('1234') where user='root';
```

```
Query OK, 1 row affected, 1 warning (0.00 sec)
Rows matched: 1 Changed: 1 Warnings: 1
```

변경사항을 적용하기 위해 아래 명령을 실행한 후에 종료하면 비밀번호 변경이 완료된다.

```
mysql> flush privileges;
mysql> quit
Bye
```

irt_zip 압축 풀기위해

`apt-get install unzip`

예로 `lib_irt.zip` 이라는 파일이 있다고 가정하면 아래와 같은 명령어로 압축을 풀 수 있습니다.

명령어 : `unzip lib_irt.zip`

하고 `python lib_irt.py`

`python manage.py makemigrations`

`python manage.py migrate`

`mysql> changed smtr_dev` 확인하기 위해

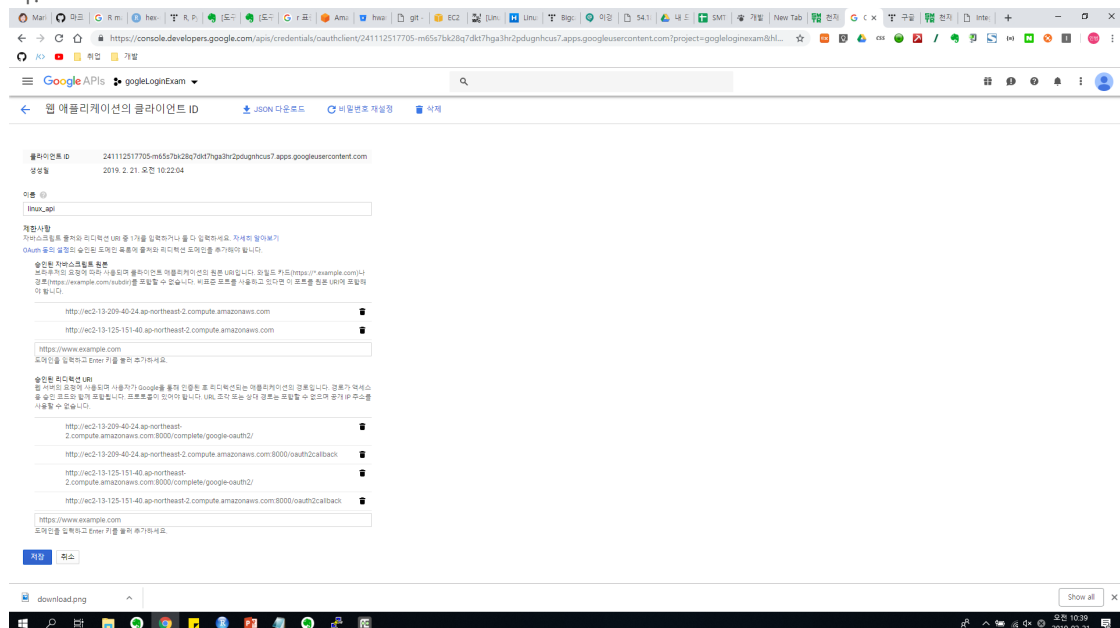
`mysql> show databases` 제대로 됐는지 확인

```
# cd /sql (sql 파일이 있는 디렉토리로 이동)
# mysql -u 접속계정 -p (mysql 접속)
데이터베이스 변경해주고
use smtr_dev
```

```
mysql> source ./data.sql
source ./smtrapp_icc.sql
source ./smtrapp_profile.sql
source ./smtrapp_cdf.sql
source ./smtrapp_iif.sql
source ./smtrapp_question.sql
source ./smtrapp_examscope.sql
source ./smtrapp_univlist.sql
source ./smtrapp_maxiif.sql
```



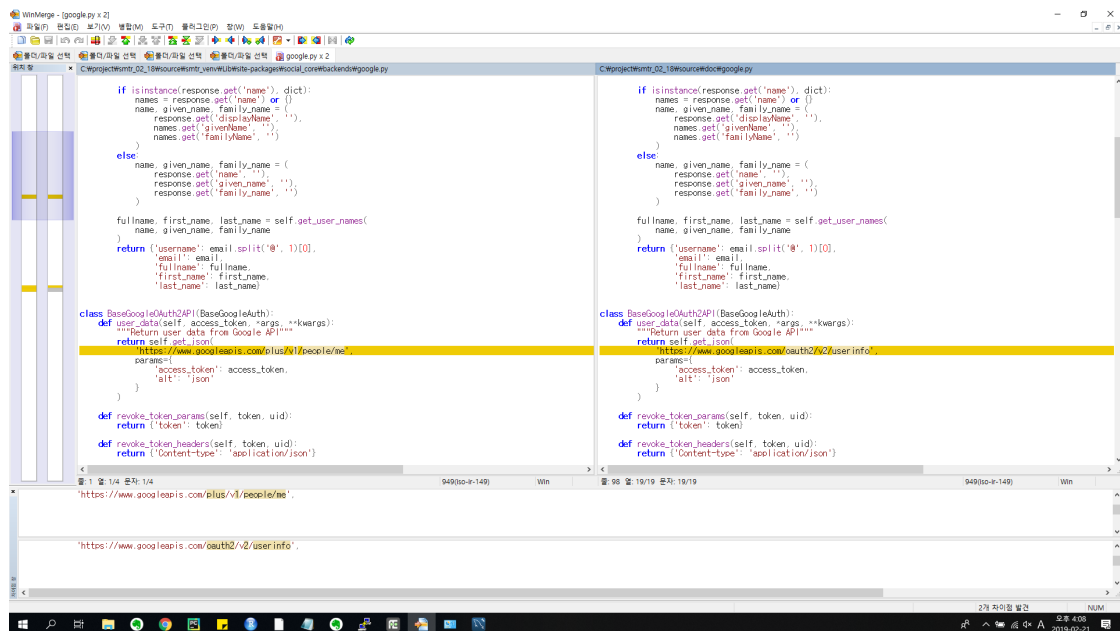
```
mysql> ALTER TABLE 테이블이름 convert to charset utf8
```



이것을 하고

doc 에 있는 google.py를

venv-lib-site packages - social_core - backend - google.py 에 있는 소스와 같게 만든다.



The screenshot shows a WinMerge window comparing two versions of a file named 'google.py'. The left pane shows the file from 'C:\project\src\social_core\backend\google.py' and the right pane shows the file from 'C:\project\src\social_core\backend\google.py'. The code is written in Python and defines a class 'BaseGoogleAuthAPI' that inherits from 'BaseGoogleAuth'. The code includes methods for getting user data, revoking tokens, and headers. The left pane has a yellow highlight on the line 'return self.get_json()' in the 'get_user_data' method. The right pane has a yellow highlight on the line 'return self.get_json()' in the 'get_user_data' method. The status bar at the bottom shows '2개 차이점 발견' (2 differences found).

```
if isinstance(response.get('name'), dict):
    names = response.get('name') or {}
    name, given_name, family_name = (
        response.get('displayName', ''),
        names.get('givenName', ''),
        names.get('familyName', '')
    )
else:
    name, given_name, family_name = (
        response.get('name', ''),
        response.get('givenName', ''),
        response.get('familyName', '')
    )

fullname, first_name, last_name = self.get_user_names(
    name, given_name, family_name
)
return {'username': email.split('@')[0],
        'email': email,
        'fullname': fullname,
        'first_name': first_name,
        'last_name': last_name}

class BaseGoogleAuthAPI(BaseGoogleAuth):
    def user_data(self, access_token, *args, **kwargs):
        """Return user data from Google API"""
        return self.get_json(
            'https://www.googleapis.com/plus/v1/people/me',
            params={
                'access_token': access_token,
            },
        )

    def revoke_token_params(self, token, uid):
        return {'token': token}

    def revoke_token_headers(self, token, uid):
        return {'Content-type': 'application/json'}
```

이걸 이용해 찾는다.

api에서 방화벽이 풀려있는지 확인하고 풀려있으면 이제 된다.