2024.04.01

CLI - Command Line Interface

- Port 22

- telnet

-> ssh

GUI - Graphic User Interface

# 디렉토리 변경

# cd

cd ~

# pwd

# 현재 폴더 위치

# 폴더 생성

mkdir workspace

sudo apt update

sudo apt install python3-pip

sudo apt install python3-pip -y

# jupyter 설치하기

pip install jupyter

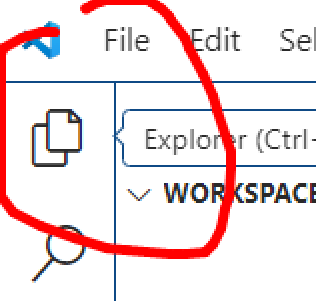
# vs code 설치하기

https://code.visualstudio.com/docs/?dv=win64user

# vs code 실행

extensions -> wsl 설치

f1 -> connect to wsl



text3 = "오늘도 아침엔 입에 빵을 물고"

text4 = """오늘도 아침엔 입에 빵을 물고

똑같이 하루를 시작하고

온종일 한 손엔 아이스 아메리카노

피곤해 죽겠네

지하철 속 이 장면 어제 꿈에서 봤나

아참 매일이지 지나치고

바쁜 이 삶에 그냥 흔한 날에

그 애를 보고 말야

평온했던 하늘이 무너지고

어둡던 눈앞이 붉어지며

뭔가 잊고 온 게 있는 것 같아

괜히 이상하게 막 울 것만 같고

그냥 지나치는 게 나을 것 같아

나는 생각은 딱 질색이니까

카페인으로 잡은 정신은 빠졌고

하루 종일 신경 쓰여 토할 것 같아

저녁이 돼도 배고픔까지 까먹고

그치 이상하지 근데 말야 있잖아

처음 본 순간 뭐라 할까 그립달까

나도 웃긴데 말야

평온했던 하늘이 무너지고

어둡던 눈앞이 붉어지며

뭔가 잊고 온 게 있는 것 같아

괜히 이상하게 막 울 것만 같고

그냥 지나치는 게 나을 것 같아

나는 생각은 딱 질색이니까

오랫동안 나를 아는

슬픈 표정을 하고 Oh

흔적 없는 기억 밖

혹 과거에 미래에 딴 차원에 세계에

1 2 3 4 5 6 7 8

평온했던 하늘이 무너지고

어둡던 눈앞이 붉어져도

다시 놓쳐버리는 것만 같아

괜히 이상하게 막 울 것만 같고

그냥 지나치는 게 나을 것 같아

나는 생각은 딱 질색이니까

아냐 지나치는 게 나을 것 같아

나는 아픈 건 딱 질색이니까"""

loan = "김미영팀장입니다. {}님께서는 최저이율로 최고 3000만원가지 입금 가능합니다."

name = ["권시은","기석광","김가현","김기호","김대건","김동현","김승주","김예송","김하영","김현지","노석현","박성우","변수현","신소영","원정인","유선우","유정연","이서연","이재연","이충원","이하은","이희재","정다인","조명아","조태식","조혜민","차민혁","최성현","최태성","한다솜",]

import requests

url = "https://www.starbucks.co.kr/store/getStore.do?r=EXRPGE2OU7"

payload = {"in\_biz\_cds" : "0","in\_scodes" : "0","ins\_lat" : "37.4947","ins\_lng" : "127.0493","search\_text" : "","p\_sido\_cd" : "01","p\_gugun\_cd" : "","isError" : "true","in\_distance" : "0","in\_biz\_cd" : "","iend" : "1000","searchType" : "C","set\_date" : "","rndCod" : "Q878EXUG04","all\_store" : "0","T03" : "0","T01" : "0","T27" : "0","T12" : "0","T09" : "0","T30" : "0","T05" : "0","T22" : "0","T21" : "0","T10" : "0","T36" : "0","T43" : "0","T48" : "0","Z9999" : "0","P02" : "0","P10" : "0","P50" : "0","P20" : "0","P60" : "0","P30" : "0","P70" : "0","P40" : "0","P80" : "0","whcroad\_yn" : "0","P90" : "0","P01" : "0","new\_bool" : "0",}

r = requests.post(url, data=payload)

data = r.json()

#int, #float, #list, #for #if~elif~else

### 내일 학습 :

# list comprehension

# dict

# 사용자 함수

# lambda

# break

# continue

# 데이터 수집

# 워드 카운트

2024.04.02

DNS

Domain Name Server

TCP/ IP

Protocol -> 약속

IP ->

jupyter notebook --generate-config

cd ~

pwd

# 목록 보기

ls

# 목록 보기 - 상세

ls -al

drwxr-xr-x 1 gen2 gen2 4096 Apr 2 09:48 .jupyter

drwxr-xr-x

- d : directory , - : file

rwx r-x r-x

내꺼 그룹 제3자

r: read, w: write, x: execute

echo "hi" > a.txt

chmod 646 a.txt

other

chmod o-w a.txt

chmod g+w a.txt

chmod u-w a.txt

cd .jupyter

vim

비주얼 모드

i 버튼 -> 쓰기 모드

: -> 명령어 모드

강제로 탈출 -> q!

쓰기 w

쓰고 나가기 wq

ipython

from jupyter\_server.auth import passwd

passwd()

vim jupyter\_notebook\_config.py

# password 키워드 찾기

:/password

n : 다음 찾기

N : 이전 찾기

:/notebook\_dir

c.ServerApp.notebook\_dir = ""

c.ServerApp.password = ""

# 쓰고 나가기

:/wq

empty\_list =[]

for x in range(0,10):

if x % 2 == 0:

empty\_list.append(x)

# list comprehension(리스트 축약형)

empty\_list2 = [x for x in range(0, 10) if x % 2 ==0]

from datetime import date, datetime, timedelta

for t in range(0, 1001, 100):

print(date.today() + timedelta(days=t))

master = dict(zip(range(0,7), "월화수목금토일"))

from datetime import date, datetime, timedelta

for t in range(0, 1001, 100):

# f-string 3.6이상

print(f"{t}일째 - {date.today() + timedelta(days=t)} - {master[(date.today() + timedelta(days=t)).weekday()]}")

# 요일 키 추가

for star in data['list']:

star['요일'] = master[datetime.strptime(star['open\_dt'], "%Y%m%d").weekday()]

#cnt\_dict = {}

cnt\_dict = {}

for x in data['list']:

# print(x['요일'])

if x['요일'] in cnt\_dict:

#cnt\_dict[x['요일']] = cnt\_dict[x['요일']] + 1

cnt\_dict[x['요일']] += 1

else:

cnt\_dict[x['요일']] = 1

[x['s\_name'] for x in data['list'] if x['s\_name'][-2:] == "DT"]

[x['addr'].split()[1] for x in data['list']]

len(list(set(([x['addr'].split()[1] for x in data['list']]))))

list\_data = ['1', '2', '3','4', '5', '6', '7', '8']

for x in list\_data:

print(int(x) \*\* 2)

def tmp(x):

return int(x) \*\* 2

list(map(tmp, list\_data))

url = "https://www.genie.co.kr/chart/top200"

head = {'User-Agent':

'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36 Edg/123.0.0.0'}

r = requests.get(url, headers=head)

r.text

from bs4 import BeautifulSoup

bs = BeautifulSoup(r.text)

for idx, x in enumerate(bs.findAll("td",class\_='info')):

title = x.find("a", class\_="title ellipsis").text.strip()

artist = x.find("a", class\_="artist ellipsis").text

print(f"{idx+1}위 -> {title} - {artist}")

lyric = "https://www.genie.co.kr/detail/songInfo?xgnm={}"

for idx, x in enumerate(bs.findAll("td",class\_='info')):

title = x.find("a", class\_="title ellipsis").text.strip()

artist = x.find("a", class\_="artist ellipsis").text

id\_ = x.find("a", class\_="title ellipsis")['onclick'].split("'")[1]

print(f"{idx+1}위 -> {title} - {artist} ")

print(lyric.format(id\_))

r2 = requests.get("https://www.genie.co.kr/detail/songInfo?xgnm=105544013", headers=head)

bs2 = BeautifulSoup(r2.text)

# 상대경로 -> ./ -> 현재 폴더

# ../ -> 현재 경로에서 상위 폴더

# 절대경로

f = open('./a.txt', "w", encoding='utf-8')

f.write(bs2.find("pre", id="pLyrics").text.strip())

f.close()

target = "./lyric"

if not os.path.isdir(target):

os.mkdir(target)

1

# 50위까지 노래 가사 저장하기

저장 위치 -> ./lyric

# 파일 형식 -> 가수\_제목.txt

2024.04.03

# 프로세스 확인

ps

## 모든 프로세스 확인

ps -ef

## |

## grep 패턴

ps -ef | grep python

# 프로세스 kill

kill -9 [pid]

cd ~/workspace

# shell

-> bash

ls -al ~/

# ssh 설치

sudo apt update

sudo apt install openssh-server

sudo service ssh start

sudo service ssh status

# 가상환경 만들기

pip install virtualenv

python3 -m virtualenv venv

ls -al

python3 -m virtualenv venv

# pip 설치 방법

pip install pandas

pip install pandas==1.2.0

pip uninstall pandas

# 패키지 버전 확인

pip list

pip freeze

# requirements.txt

pip install -r requirements.txt

# nohup -> session 유지

# & -> background

nohup jupyter-notebook &

import os

from tqdm import tqdm

lyrics\_url = "https://www.genie.co.kr/detail/songInfo?xgnm={}"

if not os.path.isdir("./lyrics"):

os.mkdir("./lyrics")

for x in tqdm(bs.find("div", {'class': \

"music-list-wrap"}).findAll("td", {"class" : "info"})):

title = x.find("a", class\_="title ellipsis").text.strip()

artist = x.find("a", class\_="artist ellipsis").text

f = open(f"./lyrics/{artist}\_{title}.txt", "w", encoding='utf-8')

#print(f"{artist}-{title}.txt")

r3 = requests.get(lyrics\_url.format(p.findall(str(x))[0]),headers=head)

f.write(BeautifulSoup(r3.text).find("pre", id="pLyrics").text.strip())

f.close()

# linux

# 이동 및 이름 바꾸기

-> mv [원본] [타켓]

# 삭제

rm

## 폴더 삭제

rm -rf [폴더]

# 복사

cp

# 하위폴더까지 복사

cp -R [대상]

https://finance.naver.com/

url = "https://m.stock.naver.com/front-api/v1/external/chart/domestic/info?symbol={}&requestType=1&startTime={}&endTime={}&timeframe=day"

r = requests.get(url.format('005930', '20230102', '20240402'))

import csv

f = open("./삼성전자.csv", 'w', newline='')

write\_stock = csv.writer(f)

write\_stock.writerows(data)

f.close()

# 아래 주소에서 api를 찾아서 데이터를 가져오고

# kospi 만

# 네이버 주식 api를 이용해서

# stock폴더를 만들고 그 안에 종목이름.csv

# 네이버에서 데이터 가져와서 저장하는 기능을 사용자 함수로 만들것

def get\_naver\_stock(code, start\_date, end\_date):

"""

code -> 종목 코드 넣어

start\_date -> 시작 날짜

end\_date -> 끝나는 날짜

"""

pass

http://data.krx.co.kr/contents/MDC/MDI/mdiLoader/index.cmd?menuId=MDC0201020201

import pandas as pd

2024.04.04

# 인터넷에 있는 파일 다운로드 받을때 사용 wget

wget https://dl.google.com/linux/direct/google-chrome-stable\_current\_amd64.deb

public ip

private ip

sudo apt install ./google-chrome-stable\_current\_amd64.deb -y

# 일반계정

nohup jupyter-notebook --ip=0.0.0.0 &

# root

nohup jupyter-notebook --allow-root --ip=0.0.0.0 &

# 네트워크 프로그램 설치

sudo apt install net-tools

# port

sudo netstat -ntlp | grep [포트]

# 실행되고 있는 실행파일의 위치

which

which google-chrome

cd /opt/google/

pip install webdriver\_manager

from selenium import webdriver

from webdriver\_manager.chrome import ChromeDriverManager

from selenium.webdriver.chrome.service import Service

driver = webdriver.Chrome(service=Service(ChromeDriverManager(driver\_version="123.0.6312.105").install()))

driver.get("https://www.koreabaseball.com/")

from selenium.webdriver.common.by import By

driver.find\_element(By.CSS\_SELECTOR, "#lnb > li:nth-child(3) > a").click()

driver.page\_source

import time

import re

pattern = re.compile("playerId=([0-9]+)")

select\_page = "#cphContents\_cphContents\_cphContents\_ucPager\_btnNo{}"

select\_team = "#cphContents\_cphContents\_cphContents\_ddlTeam > option:nth-child({})"

playid = []

for x in range(2,12):

for\_1 = select\_team.format(x)

driver.find\_element(By.CSS\_SELECTOR, for\_1).click()

time.sleep(2)

#playid.extend(pattern.findall(driver.page\_source))

for y in range(1,6):

f2 = select\_page.format(y)

try:

driver.find\_element(By.CSS\_SELECTOR, f2).click()

time.sleep(1)

playid.extend(pattern.findall(driver.page\_source))

except Exception as e:

print ("page 없음 ")

time.sleep(2)

import pickle

# binary save , load

with open("./kbo.pkl", "wb") as f:

pickle.dump(playid, f)

import pickle

with open("./kbo.pkl", "rb") as f:

abc = pickle.load(f)

#lnb > li:nth-child(3) > a

https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86\_64.exe

# conda 가상환경

conda create --name [이름] python=[버전]

conda install jupyter

pip install selenium webdriver\_manager

play\_url = "https://www.koreabaseball.com/Record/Player/PitcherDetail/Basic.aspx?playerId={}"

from bs4 import BeautifulSoup as BS

import requests

def get\_kbo(id\_):

kbo\_r = requests.get(play\_url.format(id\_))

bs = BS(kbo\_r.text)

data = {x.text.split(":")[0] : x.text.split(":")[1] for x in bs.find("div", class\_= "player\_basic").findAll("li")}

data['team'] = bs.find("h4", id="h4Team").text

return data

[' 강건', ' 강민성', ' 강백호', ' 강현우', ' 고영표', ' 김건웅', ' 김건형', ' 김규대',

' 김민', ' 김민석', ' 김민성', ' 김민수', ' 김민혁', ' 김병준', ' 김상수', ' 김성균',

' 김영현', ' 김정운', ' 김준태', ' 김지민', ' 김철호', ' 김태오', ' 로하스', ' 류현인',

' 문상철', ' 문용익', ' 박경수', ' 박민석', ' 박병호', ' 박세진', ' 박시영', ' 박시윤',

' 박영현', ' 박정현', ' 박태완', ' 배정대', ' 벤자민', ' 성재헌', ' 소형준', ' 손동현',

' 이종혁', ' 이준명', ' 이준희', ' 이채호', ' 이태규', ' 이호연', ' 장성우', ' 장준원',

' 전용주', ' 정준영', ' 정진호', ' 조대현', ' 조용근', ' 조용호', ' 조이현', ' 주권',

' 천성호', ' 최성민', ' 최윤서', ' 최정태', ' 쿠에바스', ' 하준호', ' 한민우', ' 한차현',

' 홍현빈', ' 황의준', ' 황재균']

2024.04.05

import requests

payload = {"ajax": "true",

"curCd": "",

"tmpInqStrDt": "2024-04-03",

"pbldDvCd": "0",

"pbldSqn": "",

"hid\_key\_data": "",

"inqStrDt": "20240403",

"inqKindCd": "1",

"hid\_enc\_data": "",

"requestTarget": "searchContentDiv",}

url = "https://www.kebhana.com/cms/rate/wpfxd651\_01i\_01.do"

r = requests.post(url, data=payload)

hana = pd.read\_html(r.text)[0]

def get\_exchange(code\_, date\_=None):

return hana[hana['통화\_통화'].str.find(f"{code\_}") > -1]

def get\_exchange(code\_="USD", date\_=None):

"""

code\_ = 통화코드

예) USD

date\_ = 예) '2024-01-02'

"""

payload = {"ajax": "true",

"curCd": "",

"pbldDvCd": "0",

"pbldSqn": "",

"hid\_key\_data": "",

"inqKindCd": "1",

"hid\_enc\_data": "",

"requestTarget": "searchContentDiv",}

payload['tmpInqStrDt'] = date\_

payload['inqStrDt'] = date\_.replace("-", "")

url = "https://www.kebhana.com/cms/rate/wpfxd651\_01i\_01.do"

r = requests.post(url, data=payload)

exchange = pd.read\_html(r.text)[0]

exchange.columns = ["\_".join(x[:2]) for x in exchange.columns]

return exchange.loc[exchange['통화\_통화'].str.find(f"{code\_.upper()}") > -1, '현찰\_파실 때'].iloc[0,0]

----

from datetime import date, datetime

import requests

import pandas as pd

def get\_exchange(code\_="USD", date\_=None):

"""

code\_ = 통화코드

예) USD

date\_ = 예) '2024-01-02'

"""

payload = {"ajax": "true",

"curCd": "",

"pbldDvCd": "0",

"pbldSqn": "",

"hid\_key\_data": "",

"inqKindCd": "1",

"hid\_enc\_data": "",

"requestTarget": "searchContentDiv",}

# 날짜 형식 검사

try:

datetime.strptime(date\_, "%Y-%m-%d")

except:

print ("값 확인해!!")

return -1

payload['tmpInqStrDt'] = date\_

payload['inqStrDt'] = date\_.replace("-", "")

url = "https://www.kebhana.com/cms/rate/wpfxd651\_01i\_01.do"

r = requests.post(url, data=payload)

exchange = pd.read\_html(r.text)[0]

exchange.columns = ["\_".join(x[:2]) for x in exchange.columns]

return exchange.loc[exchange['통화\_통화'].str.find(f"{code\_.upper()}") > -1, '현찰\_파실 때'].iloc[0,0]

f = open("./file.txt", "w", encoding="utf-8")

for roots, dirs, files in os.walk("/home/gen2"):

for file in files:

f.write(f"{roots}/{file}\n")

f.close()

# dict comprehension

code\_master = {x1.split()[0] : x1.split()[1] for x1 in hana['통화\_통화']}

kbo2 = kbo\_df[kbo\_df.연봉.str.find("달러") > -1].copy()

# warning 숨기기

import warnings

warnings.filterwarnings(action='ignore')

exchange = get\_exchange('usd', '2024-01-02')

kbo2['연봉'] = kbo2.연봉.apply(lambda x : int(x[:-2]) \* exchange) / 10000

kbo3 = kbo\_df[~kbo\_df.연봉.str.find("달러") > -1].copy()

# and -> %

# or -> |

# not -> ~

kbo4 = kbo3[~kbo3.연봉.apply(lambda x : len(x) < 3)].copy()

kbo4.연봉 = kbo4.연봉.apply(lambda x : int(x[:-2]))

final = pd.concat([kbo2, kbo4])

final.groupby(['team'])[['연봉']].mean().sort\_values(by=['연봉'], ascending=False)

final.groupby(['team'])[['연봉']].agg(['mean', 'median', 'var'])

공개키 (public key)

개인키(private key)

sudo apt install unzip

unzip ./이름\_생년\_성별\_10000.zip -d ./data

파일 분리하여 저장

남성 -> 남성 폴더에 저장

여성 -> 여성 폴더에 저장

'1', '3'

'2', '4'

남성, 여성 컬럼으로 DataFrame으로

.to\_csv("./report.csv", index=False, encoding='utf-8-sig')

# 파일 복사

import shutil

shutil.move("./result.csv", "/home/gen2")

2024.04.08

kbo['초등'] = kbo['경력'].apply(lambda x : x.split("-")[0])

kbo['생년월일'].apply(lambda x : datetime.strptime(x.strip(), "%Y년 %m월 %d일"))

kbo['생년월일'] = kbo['생년월일'].apply(lambda x : datetime.strptime(x.strip(), "%Y년 %m월 %d일"))

kbo['생존일'] = kbo['생년월일'].apply(lambda x : (datetime.now() - x).days)

kbo.sort\_values(by=['생존일', '선수명'], ascending=[False, True])

kbo['나이'] = kbo['생년월일'].apply(lambda x : datetime.now().year - x.year)

kbo.loc[kbo.team == "고양 히어로즈", 'team'] = '키움 히어로즈'

[매장/서비스 > 매장검색 | 대한민국 대표편의점 GS25 (gsretail.com)](http://gs25.gsretail.com/gscvs/ko/store-services/locations)

import requests

from bs4 import BeautifulSoup as BS

r= requests.get("http://gs25.gsretail.com/gscvs/ko/store-services/locations")

bs = BS(r.text)

#<form id="CSRFForm" action="/gscvs/ko/store-services/locations" method="post"><input type="hidden" name="CSRFToken" value="406690e8-0340-4931-b372-4b5519577c3c" />

bs.find("input", {'name': "CSRFToken"})

with requests.Session() as s:

r = s.get("http://gs25.gsretail.com/gscvs/ko/store-services/locations")

bs = BS(r.text)

token = bs.find("input", {'name': "CSRFToken"})

print(bs.find("input", {'name': "CSRFToken"}))

target = f"http://gs25.gsretail.com/gscvs/ko/store-services/locationList?CSRFToken={token}"

payload = {"pageNum": "1",

"pageSize": "5",

"searchShopName": "",

"searchSido": "26",

"searchGugun": "2671",

"searchDong": "26710310",

"searchType": "",

"searchTypeService": "0",

"searchTypeToto": "0",

"searchTypeCafe25": "0",

"searchTypeInstant": "0",

"searchTypeDrug": "0",

"searchTypeSelf25": "0",

"searchTypePost": "0",

"searchTypeATM": "0",

"searchTypeWithdrawal": "0",

"searchTypeTaxrefund": "0",

"searchTypeSmartAtm": "0",

"searchTypeSelfCookingUtensils": "0",

"searchTypeDeliveryService": "0",

"searchTypeParcelService": "0",

"searchTypePotatoes": "0",

"searchTypeCardiacDefi": "0",

"searchTypeFishShapedBun": "0",}

with requests.Session() as s:

r = s.get("http://gs25.gsretail.com/gscvs/ko/store-services/locations")

bs = BS(r.text)

token = bs.find("input", {'name': "CSRFToken"})['value']

#print(bs.find("input", {'name': "CSRFToken"}))

target = f"http://gs25.gsretail.com/gscvs/ko/store-services/locationList?CSRFToken={token}"

r2 = s.post(target, data=payload)

pd.DataFrame(eval(r2.json())['results'])

payload = {"pageNum": "1",

"pageSize": "5",

"searchShopName": "",

"searchSido": "",

"searchGugun": "",

"searchDong": "",

"searchType": "",

"searchTypeService": "0",

"searchTypeToto": "0",

"searchTypeCafe25": "0",

"searchTypeInstant": "0",

"searchTypeDrug": "0",

"searchTypeSelf25": "0",

"searchTypePost": "0",

"searchTypeATM": "0",

"searchTypeWithdrawal": "0",

"searchTypeTaxrefund": "0",

"searchTypeSmartAtm": "0",

"searchTypeSelfCookingUtensils": "0",

"searchTypeDeliveryService": "0",

"searchTypeParcelService": "0",

"searchTypePotatoes": "0",

"searchTypeCardiacDefi": "0",

"searchTypeFishShapedBun": "0",}

total = []

with requests.Session() as s:

r = s.get("http://gs25.gsretail.com/gscvs/ko/store-services/locations")

bs = BS(r.text)

token = bs.find("input", {'name': "CSRFToken"})['value']

#print(bs.find("input", {'name': "CSRFToken"}))

target = f"http://gs25.gsretail.com/gscvs/ko/store-services/locationList?CSRFToken={token}"

for x in range(1,3):

if x % 10 == 0: print(x)

payload['pageNum'] = x

r2 = s.post(target, data=payload)

total.append( pd.DataFrame(eval(r2.json())['results']))

gs25.reset\_index(drop=True, inplace=True)

2024.04.09

DataBase(RDBMS)

Mysql

MariaDB -

- Line

Postgresql

-> Graph DB

Oracle -> 비쌈

Google, SAP

DB2

SQL(Structured Query Language)

시퀄

ANSI SQL 2005

Hadoop

sudo apt install mysql-server

sudo service mysql status

sudo service mysql start[stop/status/restart]

wsl -> 윈도우

-> 리얼 ->

sudo systemctl enable mysql

ERP -> 전사적자원관리

1. SAP

- Samsung, LG, SK, ...

- Abap(월 2,3천)

- MM, SD, PP, HR, ....

2. HANA

- In-memory

sudo mysql -uroot -p

select user, host from mysql.user;

create user gen@'%' identified by 'encore';

# 모든 데이터베이스의 테이블 권한 주기

grant all privileges on \*.\* to gen@'%';

exit;

sudo mysql -ugen -p

# mysql 환경 파일

cd /etc/mysql/mysql.conf.d

sudo vim mysqld.cnf

bind-address = 0.0.0.0

# mysql 재가동

import sqlalchemy

from urllib import parse

user = '뷁뷁뷁뷁'

password = '뷁뷁뷁뷁뷁뷁뷁'

host='뷁뷁뷁뷁뷁'

port = 3306

database = 'encore'

password = parse.quote\_plus(password)

engine = sqlalchemy.create\_engine(f"mysql://{user}:{password}@{host}:{port}/{database}")

unique\_cu.to\_sql('cu', index=False, if\_exists="append", con=engine)

2024.04.11

import pickle

import pandas as pd

with open("./seven\_stores.pkl", "rb") as f:

seven\_df = pickle.load(f)

tmp = set()

seven\_df.able\_services.apply(lambda x : [tmp.add(y) for y in x])

for service in tmp:

seven\_df[service] = seven\_df.able\_services.apply(lambda x : 1 if service in x else 0)

seven\_df.drop(['able\_services'], axis=1, inplace=True)

seven\_df['sido'] = seven\_df['address'].apply(lambda x : x.split()[0])

pd.set\_option('display.max\_rows', None)

import requests

r = requests.post("https://www.7-eleven.co.kr/util/storeLayerPop.asp")

from bs4 import BeautifulSoup

sido\_name = [x.text for x in BeautifulSoup(r.text).find("select", id="storeLaySido").findAll('option')[1:]]

seven\_df.loc[seven\_df.sido == '주소', 'address'] = "서울시 서대문구 충정로7 구세군빌딩1층"

seven\_df.loc[seven\_df.sido == '주소', 'sido'] = "서울특별시"

seven\_df.loc[seven\_df.sido == '전라북도전주시', 'sido'] = "전라북도"

seven\_df.loc[seven\_df.sido == '경상북도영주시광복로', 'sido'] = "경상북도"

seven\_df['sido'].value\_counts()

seven\_df = seven\_df[['company', 'name', 'sido', 'address', '24시간', '무인택배접수', '카페', '베이커리', '시디', 'ATM', '무인락커',

'치킨', '의약품', '소프트아이스크림', '토토', '고구마·붕어빵', '스마트픽', '예약주문', '페덱스서비스',

]].copy()

seven\_melt = pd.melt(seven\_df, id\_vars=['company', 'name', 'sido', 'address'])

con =pymysql.connect(host='뷁뷁뷁', user='뷁뷁뷁', password='뷁뷁뷁뷁', db='뷁뷁뷁', charset='utf8')

cur = con.cursor()

CREATE Table store

(

`brand` VARCHAR(10) NOT NULL,

`store\_name` VARCHAR(30) NOT NULL,

`sido` CHAR(7) NOT NULL,

`store\_tel` CHAR(14) Default NULL,

`store\_code` CHAR(5) Default NULL,

`lat` FLOAT Default null,

`longs` FLOAT Default null,

`open\_time` CHAR(20) Default Null,

`store\_address` VARCHAR(100) NOT NULL,

`service` VARCHAR(20) NOT NULL,

`available` INT NOT NULL,

PRIMARY KEY(`store\_name`, `store\_address`)

);

INSERT INTO store VALUES ('엔코아', '서초동이야', '서울', '02-777-7777',

NULL, NULL, NULL, '08:00~21:30', '효령로 ....', '공부', 1);

SELECT \* FROM store;

python-->

sql = "INSERT INTO store VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"

# "" -> None 교체

df.store\_tel = df.store\_tel.replace("", None)

# NaN -> None 으로 replace

df2 = df.where(pd.notnull(df), None)

df2.available = df2.available.astype(int)

cur.execute(sql, df2.iloc[0].values.tolist())

con.commit()

drop table store;

CREATE Table store

(

`brand` VARCHAR(10) NOT NULL,

`store\_name` VARCHAR(30) NOT NULL,

`sido` CHAR(7) NOT NULL,

`store\_tel` CHAR(14) Default NULL,

`store\_code` CHAR(5) Default NULL,

`lat` FLOAT Default null,

`longs` FLOAT Default null,

`open\_time` CHAR(20) Default Null,

`store\_address` VARCHAR(100) NOT NULL,

`service` VARCHAR(20) NOT NULL,

`available` INT NOT NULL,

PRIMARY KEY(`store\_name`, `store\_address`, `service`)

);

import pymysql

import pandas as pd

con =pymysql.connect(host='172.23.78.97', user='gen', password='encore', db='encore', charset='utf8')

cur = con.cursor()

store\_df = pd.read\_sql\_query("select \* from store", con=con)

https://apis.data.go.kr/B552584/EvCharger/getChargerInfo?serviceKey=34lIyWAciAsJlGtIZ4ltpLy2sLZDR%2BBRWvAv8RgoADNEd%2BKCgHe84XiSRUwL8JMMIubzsFW3ddjcNlhZHhvJIQ%3D%3D&pageNo=1&numOfRows=10&zcode=11

import requests

r = requests.get("http://apis.data.go.kr/B552584/EvCharger/getChargerInfo?serviceKey=34lIyWAciAsJlGtIZ4ltpLy2sLZDR%2BBRWvAv8RgoADNEd%2BKCgHe84XiSRUwL8JMMIubzsFW3ddjcNlhZHhvJIQ%3D%3D&pageNo=1&numOfRows=10&zcode=11")

import xml.etree.ElementTree as ET

root = ET.fromstring(r.text)

items = root.iter('item')

df1 = pd.DataFrame([{y.tag : y.text for y in element} for element in items])

items = root.iter('item')

total = []

for element in items:

tmp = {}

for y in element:

tmp[y.tag] = y.text

total.append(tmp)

df2 = pd.DataFrame(total)

2024.04.12

import requests

import pandas as pd

url = "https://www.starbucks.co.kr/store/getStore.do?r=U1GAWL"

payload = {"in\_biz\_cds": "0","in\_scodes": "0","ins\_lat": "37.4909","ins\_lng": "127.0231","search\_text": "","p\_sido\_cd": "01","p\_gugun\_cd": "","in\_distance": "0","isError": "true","in\_biz\_cd": "","iend": "1000","searchType": "C","set\_date": "","rndCod": "1EMUI1PR4R","all\_store": "0","T03": "0","T01": "0","T27": "0","T12": "0","T09": "0","T30": "0","T05": "0","T22": "0","T21": "0","T10": "0","T36": "0","T43": "0","T48": "0","Z9999": "0","P02": "0","P10": "0","P50": "0","P20": "0","P60": "0","P30": "0","P70": "0","P40": "0","P80": "0","whcroad\_yn": "0","P90": "0","P01": "0","new\_bool": "0",}

df = pd.DataFrame(requests.post(url, data=payload).json()['list'])

pd.set\_option('display.max\_columns', None)

star\_df = df[['addr', 's\_name', 'tel', 'fax', 'gugun\_name', 'defaultimage', 'lat', 'lot']].copy()

pip install folium

import folum

map = folium.Map(location=[37.5502, 126.982], zoom\_start=11, tiles="cartodb positron")

import json

geo\_str = requests.get("https://t1.daumcdn.net/cfile/tistory/272C224C58B4BD540B").json()

df = pd.DataFrame(data)

df.reset\_index(inplace=True)

df.rename(columns={'count' : 'value'}, inplace=True)

folium.Choropleth(geo\_str,

data = df,

columns=['gugun\_name', 'value'],

fill\_color= "PuRd",

key\_on = 'feature.id').add\_to(map)

for idx, row in star\_df[['s\_name', 'lat', 'lot']].iterrows():

folium.Marker([row.lat, row.lot], popup=row.s\_name).add\_to(map)

map.save("./starbuck\_loaction.html")

실습

------

하나의 DataFrame 만들기

-> DB Insert

-> 1년 동안 승,하차 집계 -> 서울시 위에 히트맵

-> 강남 승,하차 시각화

2024.04.15

def fibo(n):

if n < 2:

return n

a, b = 0, 1

for i in range(n):

a, b = b, a + b

return a

# 재귀

def fibo\_recu(n):

if n < 2 : return n

return fibo\_recu(n-1) + fibo\_recu(n-2)

# 제너레이터(generator)

def fib\_generator():

a, b = 0, 1

while True:

yield b

a, b = b, a + b

fg = fib\_generator()

for \_ in range(10):

print(next(fg))

# notebook에서 sql 사용하기

pip install ipython-sql

import pandas as pd

pd.read\_sql\_query("select \* from customers", con=engine)

# mysqlclient 설치 에러나는 분들

sudo apt-get install python3-dev default-libmysqlclient-dev build-essential pkg-config

----

df1 = pd.read\_sql\_query("select \* from orders", con=con)

df2 = pd.read\_sql\_query("select \* from orderdetails", con=con)

pd.merge(df1, df2, left\_on=['orderNumber'], right\_on=['orderNumber'], how='inner')

SELECT \*

FROM orders A

LEFT JOIN

orderdetails B

ON A.orderNumber = B.orderNumber;

pd\_rt1 = pd.merge(df1, df2, left\_on=['orderNumber'], right\_on=['orderNumber'], how='left')

sql = """SELECT \*

FROM orders A

LEFT JOIN

orderdetails B

ON A.orderNumber = B.orderNumber;"""

sql\_rt1 = pd.read\_sql\_query(sql, con=con)

[Merge, join, concatenate and compare — pandas 2.2.2 documentation (pydata.org)](https://pandas.pydata.org/docs/user_guide/merging.html)

left = pd.DataFrame(

{

"key1": ["K0", "K0", "K1", "K2"],

"key2": ["K0", "K1", "K0", "K1"],

"A": ["A0", "A1", "A2", "A3"],

"B": ["B0", "B1", "B2", "B3"],

}

)

right = pd.DataFrame(

{

"key1": ["K0", "K1", "K1", "K2"],

"key2": ["K0", "K0", "K0", "K0"],

"C": ["C0", "C1", "C2", "C3"],

"D": ["D0", "D1", "D2", "D3"],

}

)

result = pd.merge(left, right, how="left", on=["key1", "key2"])

https://www.notion.so/SQL-323e197320d642f58a2fa106e6269ef8?pvs=4

CREATE VIEW tmp\_orders AS

SELECT \* FROM orders LIMIT 10;

select COUNT(\*)

FROM (SELECT DISTINCT(orderNumber) FROM orders) A

df2[df2.orderNumber.isin(orders\_df1.orderNumber.tolist())].shape

https://seasoned-tree-375.notion.site/SQL-323e197320d642f58a2fa106e6269ef8?pvs=4

import glob

subway\_total = [pd.read\_csv(file\_locate, encoding='cp949') \

for file\_locate in glob.glob("./data/\*.csv")]

import os

target = sorted(glob.glob("./data/\*.csv"), key=os.path.getctime)

subway\_total = [pd.read\_csv(file\_locate, encoding='cp949') \

for file\_locate in target ]

# 참고

# 변수명 데이터프레임 만들기

for file in target:

locals()[f'{file.split(".")[1].split("/")[-1]}'] = \

pd.read\_csv(file, encoding='cp949')

for x in subway\_total:

display(x.head(1))

print("-" \* 100)

subway\_df1= pd.concat(subway\_total[:5])

sub\_df1\_1 = subway\_df1.drop('할인', axis=1).groupby(['날짜', '호선', '역명', '구 분'],

as\_index=False).sum()

sub\_df1\_1.columns = sub\_df1\_1.columns.str.replace(" ", "").tolist()

import re

p = re.compile("(\([0-9]+\))")

sub\_df1\_1['역명'] = sub\_df1\_1.역명.apply(lambda x : re.sub(p, "", x))

----

subway\_df1= pd.concat(subway\_total[:5])

sub\_df1\_1 = subway\_df1.drop(['할인','호선'], axis=1).groupby(['날짜', '역명', '구 분'],

as\_index=False).sum()

sub\_df1\_1.columns = sub\_df1\_1.columns.str.replace(" ", "").tolist()

import re

p = re.compile("(\([0-9]+\))")

sub\_df1\_1['역명'] = sub\_df1\_1.역명.apply(lambda x : re.sub(p, "", x))

sub\_df1\_2 = sub\_df1\_1.groupby(['날짜', '역명', '구분'], as\_index=False).sum()

-----------

subway\_df2 = pd.concat(subway\_total[5:-1])

subway\_df2.drop("01~02", axis=1, inplace=True)

subway\_df2['역명'] = subway\_df2.역명.apply(lambda x : re.sub(p, "", x))

subway\_df2\_1 = subway\_df2.groupby(['역명', '날짜', '구분'],as\_index=False).sum()

sub\_01\_11 = pd.concat([sub\_df1\_2, subway\_df2\_1])

-------

sub\_01\_11.query(" 역명 == '사당' and 날짜 == '2016-05-01'")

sub\_01\_11[(sub\_01\_11.역명 == "사당") & (sub\_01\_11.날짜 == '2016-05-01')]

---

subway\_df3 = subway\_total[-1]

subway\_df3.columns = subway\_df3.columns.str.replace("시", "")

subway\_df3.drop('01~02', axis=1, inplace=True)

subway\_df3.역명 = subway\_df3.역명.apply(lambda x : re.sub(p, "", x))

subway\_df3\_1 = subway\_df3.groupby(['역명', '날짜', '구분'], as\_index=False).sum()

subway = pd.concat([sub\_01\_11, subway\_df3\_1])

sql = """CREATE TABLE subway(

date\_ DATE NOT NULL,

station VARCHAR(3) NOT NULL,

gubun CHAR(2) NOT NULL,

timeline CHAR(7) NOT NULL,

value INT(5) NOT NULL)"""

cur.execute(sql)

subway\_melt = pd.melt(subway, id\_vars=['날짜','역명', '구분'])

sql = "INSERT INTO subway VALUES (%s, %s, %s, %s, %s)"

cur.execute(sql, subway\_melt.iloc[0].tolist())