import numpy as np  
import pandas as pd  
from pandas import Series, DataFrame  
from pandas.io.parsers import TextFileReader  
  
#Read CSV's into Project  
CD = pd.read\_csv('transformed\_data.csv')  
RD = pd.read\_csv('raw\_data.csv')  
EU = pd.read\_csv('states.csv')  
Countries = pd.read\_csv('countries\_of\_the\_world.csv')  
# Create Key to merge data  
CD.sort\_values('DATE')  
CD['KEYCODE&DATE'] = CD['CODE'] + CD['DATE']  
RD.sort\_values('date')  
RD['KEYCODE&DATE'] = RD['iso\_code'] + RD['date']  
print(RD['KEYCODE&DATE'])  
print(CD['KEYCODE&DATE'])  
#print(RD.columns)  
#print(RD.head)  
RD['DATE'] = pd.to\_datetime(RD['date'], format='%Y/%m/%d')  
CD['DATE'] = pd.to\_datetime(CD['DATE'], format='%Y/%m/%d')  
  
print(CD.columns)  
print(RD.columns)  
print(EU.columns)  
print(Countries)  
print(Countries.columns)  
  
print(EU.index)  
# for val in EU :  
# print(val)  
#EU2=(pd.concat([RD, Countries, sort=True))  
#print(EU2)  
# print(EU.head(5))  
# print(EU.columns)  
  
# Merge EU data into CD Data  
CD = (CD.merge(EU, left\_on='COUNTRY', right\_on='Country', how='inner'))  
print(CD.head)  
print(CD.columns)  
print(CD.isna().sum())  
CD.to\_csv('CD.csv')  
  
  
# import matplotlib.pyplot as plt  
# COVID\_EU2.plot(x='STI',y='Date', kind='line')  
# plt.show()  
  
  
RDCD = (RD.merge(CD, on='KEYCODE&DATE', how='outer', suffixes=('\_RD', '\_CD')))  
RDCD['DATE'] = pd.to\_datetime(RDCD['date'], format='%Y/%m/%d')  
  
# Create Key Fields  
RDCD['Week\_Num'] = RDCD['DATE'].dt.strftime('%U')  
RDCD['day'] = RDCD['DATE'].dt.strftime('%A')  
RDCD['dayz'] = RDCD['DATE'].dt.strftime('%j')  
RDCD['Dates'] = RDCD['DATE'].dt.strftime('%Y%m%d')  
RDCD['Wednesdays'] = RDCD['day'] == 'Wednesday'  
# Calculate Weekly Change in Cases  
RDCD['Weekly\_Cases'] = RDCD.total\_cases.diff()  
RDCD['Weekly\_Deaths'] = RDCD.total\_deaths.diff()  
# add columns with comparable measures  
RDCD['CasesxArea'] = RDCD['total\_cases'] / RDCD['Area (km²)']  
RDCD['CasesxPop'] = RDCD['total\_cases'] / RDCD['population']  
RDCD['DeathsxCases'] = RDCD['total\_deaths'] / RDCD['total\_cases']  
  
  
# park/remove unneeded rows  
del RDCD['Unnamed: 9']  
del RDCD['Unnamed: 10']  
del RDCD['Unnamed: 11']  
del RDCD['Unnamed: 12']  
del RDCD['Unnamed: 13']  
del RDCD['GDP per capita ($, millions)']  
del RDCD['GDP ($, millions)']  
del RDCD['GDP (€, millions)']  
del RDCD['Population Density']  
del RDCD['Population']  
del RDCD['Language']  
del RDCD['Currency Code']  
del RDCD['Currency']  
del RDCD['European Monetary Union']  
del RDCD['European Single Market']  
del RDCD['European Free Trade Agreement']  
del RDCD['European Parliament Seats']  
del RDCD['Council Votes']  
del RDCD['Accession Year']  
del RDCD['GDPCAP']  
del RDCD['POP']  
del RDCD['STI']  
del RDCD['TD']  
del RDCD['TC']  
del RDCD['HDI']  
del RDCD['DATE\_CD']  
del RDCD['COUNTRY']  
del RDCD['CODE']  
del RDCD['KEYCODE&DATE']  
del RDCD['DATE\_RD']  
  
# Create groups of Economic Blocks using Dict method  
econ\_blocks = {  
 'Country': ['Austria', 'Belgium', 'Bulgaria', 'Croatia', 'Cyprus', 'Czech Republic', 'Denmark', 'Estonia',  
 'Finland', 'France', 'Germany', 'Greece', 'Hungary', 'Ireland', 'Italy', 'Latvia', 'Lithuania',  
 'Luxembourg', 'Malta', 'Netherlands', 'Poland', 'Portugal', 'Romania', 'Slovakia', 'Slovenia', 'Spain',  
 'Sweden', 'United Kingdom', 'Brazil', 'Russia', 'India', 'China', 'South Africa', 'United States',  
 'Japan'],  
 'Econ\_Block': ['EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU',  
 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'EU', 'BRICS', 'BRICS', 'BRICS', 'BRICS',  
 'BRICS', 'US', 'Japan']}  
Econ\_BLocks = pd.DataFrame(econ\_blocks)  
RDCD = (RDCD.merge(Econ\_BLocks, left\_on='location', right\_on='Country', how='outer'))  
# check for missing data  
print(RDCD.isna().sum())  
RDCD.fillna(0)  
  
#Replace Missing Values by filling 'na's with 'No\_Affiliation' identifier  
RDCD['Econ\_Block'] = RDCD['Econ\_Block'].fillna('No\_Affiliation')  
print(RDCD.columns)  
print(RDCD.isna().any())  
print(RDCD.isna().sum())  
# Use Wednesday data as weekly reporting day  
RDCD1=RDCD.query('day == "Wednesday"')  
  
# define 2020 period of analysis  
print(RDCD1.loc[:'Week\_Num'])  
#RDCD2=RDCD1.query('Week\_Num <= "39"')  
RDCD2=RDCD1.query('date <= "2020-09-30"')  
  
del RDCD['Country\_y']  
  
  
#forward fill for missing entries  
  
#RDCD2EB = RDCD2.groupby('Econ\_Block')['total\_cases', 'total\_deaths', 'Deaths/Cases'].sum()  
#print(RDCD2EB)  
RDCD3=(RDCD2.pivot\_table(values='CasesxPop', index='Week\_Num', columns='Econ\_Block', fill\_value=0, margins=False))  
  
RDCD.to\_csv('RDCD.csv')  
RDCD2.to\_csv('RDCD2.csv')  
RDCD3.to\_csv('RDCD3.csv')  
  
print(RDCD3)  
# Import the matplotlib.pyplot submodule and name it plt  
import matplotlib.pyplot as plt  
  
# Create a Figure and an Axes with plt.subplots  
#fig, ax = plt.subplots()  
  
# Plot MLY-PRCP-NORMAL from seattle\_weather against MONTH  
#RDCD2=RDCD2.sort\_values(['iso\_code', 'Econ\_Block', 'Week\_Num'])  
#ax.plot(RDCD2["date"], RDCD2["total\_cases"])  
#plt.show()  
  
#print(type('Week\_Num'))  
# RDCD.groupby('Econ\_Block')['Cases/Pop'].mean())  
# print(RDCD)  
  
  
# print(EUCD2.head  
# print(EUCD2.columns)  
# print(EUCD2.isna().sum())  
import numpy as np  
  
#Filter df to Wks1-25 2020  
  
#filter\_criteria=((RDCD['Week\_Num'] <='25') & (RDCD['Econ\_Block'] !='No\_Affiliation') & (RDCD['day'] =='Wednesday') )  
#print(RDCD.loc[filter\_criteria, ['Cases/Pop', 'population']].mean())  
  
#print(Econ)  
  
import matplotlib.pyplot as plt  
#Econ.plot(x='Week\_Num', y='Econ\_Block', kind='line', rot=45)  
#plt.show()  
  
# Drop duplicate store/department combinations  
#CODE\_Dupes = df.drop\_duplicates(subset=["CODE", "COUNTRY"])  
#print(CODE\_Dupes)  
  
# Subset the rows where Wednesday is True and drop duplicate dates  
#Wednesday\_only\_Data = df[df["Wednesdays"]].drop\_duplicates(subset="Wkdayz2")  
#print(Wednesday\_only\_Data)  
  
# Print date col of Wednesday\_only\_Data  
#print(Wednesday\_only\_Data["Wkdayz2"])  
  
# still need to work out how to exclude the range outside of H1  
#RDCD['Active Range'] = RDCD['Week\_Num']<=25)  
  
#still need to work out how you are going to report the econ blocks mean or pivot  
  
  
#OLD CODE  
# EU['EU\_Membership']=(EU['European Union']=='Member')  
#BRICS = ['Brazil', 'Russia', 'India', 'China', 'South Africa']  
#condition = CD["COUNTRY"].isin(BRICS)  
#print(CD[condition])  
  
  
#print(EU[EU['European Union'] == 'EU\_Membership'])  
# COVID\_EU2=(EU2.merge(COVID\_DATA, left\_on='Code', right\_on='CODE', how='outer'))  
# COVID\_EU2['Cases per Head of Pop']=COVID\_EU2['TC']/COVID\_EU2['Population']  
  
# print(COVID\_EU2.head(10))  
# print(COVID\_EU2.columns)  
# print(COVID\_EU2.shape)  
# print(COVID\_EU2.isna().sum())