你現在是位非常專業且資深的工程師，幫我看一段Python程式，首先我要讀取在data資料夾中'今日'交易的excel檔，'今日'格式為yyyymmdd，檔案名稱為'今日'\_20230406\_TraderPosition.xlsx，這裡的今日是指數字，是今天的日期，例如今天是2023年4月8號的話，'今日'就是20230408的意思，我的工作表名稱有Jesse Aaron Mark1 Gary Peggy PETE，請幫我寫一個df\_xxx (xxx為名稱，例如讀取Jesse的工作表時就是df\_Jesse)，這樣的程式，我只要讀取Jesse Aaron Mark1 Gary Peggy PETE這六個工作表，從以下程式開始

import datetime

import pandas as pd

# 取得今天日期，並轉換為字串 'yyyymmdd' 格式

today = datetime.date.today().strftime('%Y%m%d')

# 設定檔案路徑，檔名為今天日期\_TraderPosition.xlsx

file\_path = f'data/{today}\_TraderPosition.xlsx'

#有哪些人

names = ['Jesse', 'Aaron', 'Mark1', 'Gary', 'Peggy', 'PETE']

# 要讀取的工作表名稱

sheet\_names = names

# 讀取 Excel 檔案中指定的工作表, 不讀取前兩列 & 只讀取A到I欄

df\_dict = pd.read\_excel(file\_path, sheet\_name=sheet\_names, skiprows=2, usecols='A:I')

# 將每個工作表的 DataFrame 存入以工作表名稱為 key 的字典中

for sheet\_name, df in df\_dict.items():

var\_name = f'df\_{sheet\_name}'

globals()[var\_name] = df

print(df\_dict)

但是請幫我縮小擷取範圍，對於每張工作表，我只要擷取第一欄有包含3這數字、但不包含total的列數(例如RXM3裡有3且沒有total，就要擷取到這列，RXM3\_total就不要擷取到，TYM3這列也要擷取到，但TYM3\_total有total，不要擷取到：

traderposition報表...............................................................................................................

import datetime

import pandas as pd

# 取得今天日期，並轉換為字串 'yyyymmdd' 格式

today = datetime.date.today().strftime('%Y%m%d')

# 設定檔案路徑，檔名為今天日期\_TraderPosition.xlsx

file\_path = f'data/{today}\_TraderPosition.xlsx'

# 有哪些人

names = ['Jesse', 'Aaron', 'Mark1', 'Gary', 'Peggy', 'PETE']

# 要讀取的工作表名稱

sheet\_names = names

# 將每個工作表的 DataFrame 存入以工作表名稱為 key 的字典中, 不讀取前兩列 & 只讀取A到I欄, 只擷取第一欄中前面兩字包含"US"或"TY"或"FV"或"TU"的row、但不包含total的列數

df\_dict = {}

for sheet\_name in sheet\_names:

df = pd.read\_excel(file\_path, sheet\_name=sheet\_name, skiprows=2, usecols='A:I')

df = df[df.iloc[:, 0].astype(str).str.slice(0, 2).isin(['US', 'TY', 'FV', 'TU']) & ~df.iloc[:, 0].astype(str).str.contains('total', case=False)]

df\_dict[sheet\_name] = df

# 將 DataFrame 存入變數 df\_{sheet\_name} 中

var\_name = f'df\_{sheet\_name}'

globals()[var\_name] = df

#print(df\_dict)

海外債券一覽表...............................................................................................................

import pandas as pd

import datetime

# 取得今天日期，並轉換為字串 'yymmdd' 格式

today = datetime.date.today().strftime('%y%m%d')

# 設定檔案路徑，檔名為海外債券一覽表{today}.xlsm

file\_path = f'data/海外債券一覽表{today}.xlsm'

# 要讀取的工作表名稱

sheet\_names = ['Jesse\_UR', 'Jesse\_Real', 'Aaron\_UR', 'Aaron\_Real', 'Mark1\_UR', 'Mark1\_Real',

'Gary\_UR', 'Gary\_Real', 'Peggy\_UR', 'Peggy\_Real', 'PETE\_UR', 'PETE\_Real']

# 讀取Excel檔案中指定的工作表，並存入以工作表名稱為 key 的字典中

po\_dict = {}

for sheet\_name in sheet\_names:

po = pd.read\_excel(file\_path, sheet\_name=sheet\_name, usecols="A:I", header=0)

start\_row = po.loc[po.iloc[:, 0] == "USD\_Futures"].index[0] + 1

end\_row = po.loc[po.iloc[:, 0] == "USD\_total"].index[0]

po = po.iloc[start\_row:end\_row, :]

# 將 USD\_Futures 那一列的資料轉換成對應的 column name

column\_names = list(po.iloc[0])

po.columns = column\_names

po = po.iloc[1:, :]

po = po.reset\_index(drop=True) # 重新編號 row index

po\_dict[sheet\_name] = po

# 將 DataFrame 存入變數 po\_{sheet\_name} 中

var\_name = f'po\_{sheet\_name}'

globals()[var\_name] = po

po\_dict[sheet\_name].dropna(axis=0, how='all', inplace=True)

#print(po\_dict)

# 創建一個空的 dataframe，列名為姓名，列數為0

for name in names:

globals()[f'po\_{name}\_Real\_New'] = pd.DataFrame(columns=po\_dict[f'{name}\_Real'].columns)

叫ChatGPT做事：

你現在是位非常專業且資深的工程師，我現在有兩張報表，第一份是Traderposition表，第二份是海外債券一覽表。

我們的人名names list有這些人：Jesse,Aaron,Mark1,Gary,Peggy,PETE

這是第一份traderposition的code

import datetime

import pandas as pd

# 取得今天日期，並轉換為字串 'yyyymmdd' 格式

today = datetime.date.today().strftime('%Y%m%d')

# 設定檔案路徑，檔名為今天日期\_TraderPosition.xlsx

file\_path = f'data/{today}\_TraderPosition.xlsx'

#有哪些人

names = ['Jesse', 'Aaron', 'Mark1', 'Gary', 'Peggy', 'PETE']

# 要讀取的工作表名稱

sheet\_names = names

# 將每個工作表的 DataFrame 存入以工作表名稱為 key 的字典中, 不讀取前兩列 & 只讀取A到I欄, 只擷取第一欄有包含3這數字、但不包含total的列數

df\_dict = {}

for sheet\_name in sheet\_names:

df = pd.read\_excel(file\_path, sheet\_name=sheet\_name, skiprows=2, usecols='A:I')

df = df[df.iloc[:, 0].astype(str).str.contains('3') & ~df.iloc[:, 0].astype(str).str.contains('total', case=False)]

df\_dict[sheet\_name] = df

# 將 DataFrame 存入變數 df\_{sheet\_name} 中

var\_name = f'df\_{sheet\_name}'

globals()[var\_name] = df

這是第二份海外債券一覽表的code

import pandas as pd

import datetime

# 取得今天日期，並轉換為字串 'yymmdd' 格式

today = datetime.date.today().strftime('%y%m%d')

# 設定檔案路徑，檔名為海外債券一覽表{today}.xlsm

file\_path = f'data/海外債券一覽表{today}.xlsm'

# 要讀取的工作表名稱

sheet\_names = ['Jesse\_UR', 'Jesse\_Real', 'Aaron\_UR', 'Aaron\_Real', 'Mark1\_UR', 'Mark1\_Real',

'Gary\_UR', 'Gary\_Real', 'Peggy\_UR', 'Peggy\_Real', 'PETE\_UR', 'PETE\_Real']

# 讀取Excel檔案中指定的工作表，並存入以工作表名稱為 key 的字典中

po\_dict = {}

for sheet\_name in sheet\_names:

po = pd.read\_excel(file\_path, sheet\_name=sheet\_name, usecols="A:K", header=0)

start\_row = po.loc[po.iloc[:, 0] == "USD\_Futures"].index[0] + 1

end\_row = po.loc[po.iloc[:, 0] == "USD\_total"].index[0]

po = po.iloc[start\_row:end\_row, :]

# 將 USD\_Futures 那一列的資料轉換成對應的 column name

column\_names = list(po.iloc[0])

po.columns = column\_names

po = po.iloc[1:, :]

po = po.reset\_index(drop=True) # 重新編號 row index

po\_dict[sheet\_name] = po

# 將 DataFrame 存入變數 po\_{sheet\_name} 中

var\_name = f'po\_{sheet\_name}'

globals()[var\_name] = po

po\_dict[sheet\_name].dropna(axis=0, how='all', inplace=True)

#print(po\_dict)

# 創建一個空的 dataframe，列名為姓名，列數為0

for name in names:

globals()[f'po\_{name}\_Real\_New'] = pd.DataFrame(columns=po\_dict[f'{name}\_Real'].columns)

我這裡的人名names list是指'Jesse', 'Aaron', 'Mark1', 'Gary', 'Peggy', 'PETE'，而我現在是不是有了所有人名的df\_dict 以及 po\_UR跟po\_Real的dataframe了不是嗎?以Jesse為例，df dataframe為df\_Jesse，po\_人名\_UR dataframe則是指po\_Jesse\_UR的意思，還有po\_人名\_Real其實是指po\_Jesse\_Real此dataframe的意思，

然後開始幫我寫code我會有一個大前提跟四個情況要你注意，每個情況都要遵照大前提：

大前提：對於每個人的df\_人名 dataframe中，對於每個row中，首先看該row的ProductID column的值，是否在po\_人名\_UR的報表中的ProductID中有一樣的值，若有，則看po\_人名\_UR報表欄位Lots中是否有非0的數值，沒有的話就看下一個有著同樣ProductID的row的Lots是否有非0的數值，再來我也需要Broker column一樣，若在po\_人名\_UR發現ProductID一樣且Broker一樣且Lots有非0數字的話則做以下處理：

情況一：對這個row來說，若在df\_人名 dataframe中Lots的值>0而po\_人名\_UR發現ProductID一樣且Lots<0且絕對值是df\_人名 dataframe中Lots的值大於等於時，則將po\_人名\_UR的那條row新增到po\_人名\_Real\_New，然後將新增的那條row的MKT Price column欄位輸入跟df\_人名dataframe的該row的Cost欄位一樣的值，再來就將df\_人名該row的Lots column值則變成原本的Lots值減去po\_人名\_UR該row的Lots column的絕對值，而po\_人名\_UR該row的Lots column值改為0，都弄好後，再重複執行以上全部動作值到沒有符合條件為止

情況二：對這個row來說，若在df\_人名 dataframe中Lots的值>0而po\_人名\_UR發現Product ID一樣且Lots<0且絕對值是df\_人名 dataframe中Lots的值較小時，則將po\_人名\_UR的那條row新增到po\_人名\_Real\_New，然後將新增的那條row的Lots column值為跟df\_人名dataframe的該row的Lots欄位的值正負號相反、而新增的那條row的MKT Price column欄位輸入跟df\_人名dataframe的該row的Cost欄位一樣的值，再來就將po\_人名\_UR該row的Lots column值變成原本的Lots值加上df\_人名該row的Lots column值，df\_人名該row的Lots column值則變成0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止

情況三：對這個row來說，若在df\_人名 dataframe中Lots的值<0而po\_人名\_UR發現ProductID一樣且Lots>0且絕對值是df\_人名 dataframe中Lots的值大於等於時，則將po\_人名\_UR的那條row新增到po\_人名\_Real\_New，然後將新增的那條row的MKT Price column欄位輸入跟df\_人名dataframe的該row的Cost欄位一樣的值，再來就將df\_人名該row的Lots column值則變成原本的Lots值加上po\_人名\_UR該row的Lots column值，而po\_人名\_UR該row的Lots column值改為0，都弄好後，再重複執行以上全部動作值到沒有符合條件為止

情況四：對這個row來說，若在df\_人名 dataframe中Lots的值<0而po\_人名\_UR發現Product ID一樣且Lots>0且絕對值是df\_人名 dataframe中Lots的值較小時，則將po\_人名\_UR的那條row新增到po\_人名\_Real\_New，然後將新增的那條row的Lots column值為負的df\_人名dataframe的該row的Lots欄位的值、而新增的那條row的MKT Price column欄位輸入跟df\_人名dataframe的該row的Cost欄位一樣的值，再來就將po\_人名\_UR該row的Lots column值變成原本的Lots值加上df\_人名該row的Lots column值，df\_人名該row的Lots column值則變成0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止

情況五：執行完情況一到四後，若df\_人名dataframe中還有Lots值非為0的row，則對每個row進行以下處理：將該row的ProductID column到Cost column貼到po\_人名\_UR Lots欄位為0的row中，也就是只要貼第一到第九column的意思，貼完後df\_人名該row的Lots column值變成0，再重複執行以上

........................................................................................................................................

再來，我要繼續對po\_人名\_UR做以下動作，跟之前的很像，幫我寫一段Python code，我會有一個大前提跟四個情況要你注意，每個情況都要遵照大前提：

大前提：對於每個人的po\_人名\_UR dataframe中，對於每個row中，首先看該row的ProductID column的值，是否在同一個dataframe中的ProductID中有一樣的值，若有，則看po\_人名\_UR的Lots column中是否有非0的數值，沒有的話就看下一個有著同樣ProductID的row的Lots是否有非0的數值，再來我也需要Broker column一樣，若在po\_人名\_UR發現ProductID一樣且Broker一樣且Lots有非0數字的話則做以下處理：

情況一：對這個row (假設叫 row1)來說，若該row1 Lots的值>0而在po\_人名\_UR其他row(假設叫row2)發現ProductID一樣且Lots<0且row1 Lots值的絕對值大於等於row2的 Lots的值時，則將row2新增到po\_人名\_Real\_New (假設叫row3)，然後將row3的MKT Price column欄位輸入跟row1的Cost Column一樣的值，再來就將row1的Lots column值則變成原本的Lots值減去row2的Lots column的絕對值，而row2的Lots column值改為0，都弄好後，再重複執行以上全部動作值到沒有符合條件為止

情況二：對這個row (假設叫 row1)來說，該row1 Lots的值>0而在po\_人名\_UR其他row(假設叫row2)發現ProductID一樣且Lots<0且row1 Lots值的絕對小於row2的 Lots的值時，則將row2新增到po\_人名\_Real\_New (假設叫row3)，然後row3的Lots column值為跟row1的Lots Column值正負號相反的值、而row3的MKT Price column欄位輸入跟row1的Cost column一樣的值，再來就將row2的Lots column值變成原本row2的Lots值加上row1的Lots column值，row1的Lots column值則變成0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止

情況三：對這個row (假設叫 row1)來說，若該row1 Lots的值<0而在po\_人名\_UR其他row(假設叫row2)發現ProductID一樣且Lots>0且row1 Lots值的絕對值大於等於row2的 Lots的值時，則將row2新增到po\_人名\_Real\_New (假設叫row3)，然後將row3的MKT Price column欄位輸入跟row1的Cost Column一樣的值，再來就將row1的Lots column值則變成原本的Lots值加上row2的Lots column值，而row2的Lots column值改為0，都弄好後，再重複執行以上全部動作值到沒有符合條件為止

情況四：對這個row (假設叫row1)來說，該row1 Lots的值<0而在po\_人名\_UR其他row(假設叫row2)發現ProductID一樣且Lots>0且row1 Lots值的絕對小於row2的 Lots的值時，則將row2新增到po\_人名\_Real\_New (假設叫row3)，然後row3的Lots column值為跟row1的Lots Column值正負號相反的值、而row3的MKT Price column欄位輸入跟row1的Cost column一樣的值，再來就將row2的Lots column值變成原本row2的Lots值加上row1的Lots column值，row1的Lots column值則變成0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止

........................................................................................................................................

從頭到尾寫一次

import datetime

import pandas as pd

# 取得今天日期，並轉換為字串 'yyyymmdd' 格式

today = datetime.date.today().strftime('%Y%m%d')

# 設定檔案路徑，檔名為今天日期\_TraderPosition.xlsx

file\_path = f'data/{today}\_TraderPosition.xlsx'

# 有哪些人

names = ['Jesse', 'Aaron', 'Mark1', 'Gary', 'Peggy', 'PETE']

# 要讀取的工作表名稱

sheet\_names = names

# 將每個工作表的 DataFrame 存入以工作表名稱為 key 的字典中, 不讀取前兩列 & 只讀取A到I欄, 只擷取第一欄中前面兩字包含"US"或"TY"或"FV"或"TU"的row、但不包含total的列數

df\_dict = {}

for sheet\_name in sheet\_names:

df = pd.read\_excel(file\_path, sheet\_name=sheet\_name, skiprows=2, usecols='A:I')

df = df[df.iloc[:, 0].astype(str).str.slice(0, 2).isin(['US', 'TY', 'FV', 'TU']) & ~df.iloc[:, 0].astype(str).str.contains('total', case=False)]

df\_dict[sheet\_name] = df

# 將 DataFrame 存入變數 df\_{sheet\_name} 中

var\_name = f'df\_{sheet\_name}'

globals()[var\_name] = df

#print(df\_dict)

import pandas as pd

import datetime

# 取得今天日期，並轉換為字串 'yymmdd' 格式

today = datetime.date.today().strftime('%y%m%d')

# 設定檔案路徑，檔名為海外債券一覽表{today}.xlsm

file\_path = f'data/海外債券一覽表{today}.xlsm'

# 要讀取的工作表名稱

sheet\_names = ['Jesse\_UR', 'Jesse\_Real', 'Aaron\_UR', 'Aaron\_Real', 'Mark1\_UR', 'Mark1\_Real',

'Gary\_UR', 'Gary\_Real', 'Peggy\_UR', 'Peggy\_Real', 'PETE\_UR', 'PETE\_Real']

# 讀取Excel檔案中指定的工作表，並存入以工作表名稱為 key 的字典中

po\_dict = {}

for sheet\_name in sheet\_names:

po = pd.read\_excel(file\_path, sheet\_name=sheet\_name, usecols="A:I", header=0)

start\_row = po.loc[po.iloc[:, 0] == "USD\_Futures"].index[0] + 1

end\_row = po.loc[po.iloc[:, 0] == "USD\_total"].index[0]

po = po.iloc[start\_row:end\_row, :]

# 將 USD\_Futures 那一列的資料轉換成對應的 column name

column\_names = list(po.iloc[0])

po.columns = column\_names

po = po.iloc[1:, :]

po = po.reset\_index(drop=True) # 重新編號 row index

po\_dict[sheet\_name] = po

# 將 DataFrame 存入變數 po\_{sheet\_name} 中

var\_name = f'po\_{sheet\_name}'

globals()[var\_name] = po

po\_dict[sheet\_name].dropna(axis=0, how='all', inplace=True)

#print(po\_dict)

# 創建一個空的 dataframe，列名為姓名，列數為0

for name in names:

globals()[f'po\_{name}\_Real\_New'] = pd.DataFrame(columns=po\_dict[f'{name}\_Real'].columns)

import pandas as pd

def process\_conditions(df\_name, po\_name\_UR, po\_name\_Real\_New):

condition\_met = True

while condition\_met:

condition\_met = False

for index, row in df\_name.iterrows():

product\_id = row['ProductID']

broker = row['Broker']

matching\_rows = po\_name\_UR[(po\_name\_UR['ProductID'] == product\_id) & (po\_name\_UR['Broker'] == broker)]

for ur\_index, matching\_row in matching\_rows.iterrows():

if matching\_row['Lots'] != 0:

# 情況一

if row['Lots'] > 0 and matching\_row['Lots'] < 0 and abs(row['Lots']) >= abs(matching\_row['Lots']):

matching\_row['MKT Price'] = row['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, pd.DataFrame([matching\_row])], ignore\_index=True)

df\_name.at[index, 'Lots'] = row['Lots'] - abs(matching\_row['Lots']) # 更新 df\_人名 的 Lots 欄位

po\_name\_UR.at[ur\_index, 'Lots'] = 0 # 更新 po\_人名\_UR 的 Lots 欄位

condition\_met = True

break

# 情況二

elif row['Lots'] > 0 and matching\_row['Lots'] < 0 and abs(row['Lots']) < abs(matching\_row['Lots']):

new\_row = matching\_row.copy()

new\_row['Lots'] = -row['Lots'] # 更新 Lots 欄位

new\_row['MKT Price'] = row['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, pd.DataFrame([new\_row])], ignore\_index=True)

po\_name\_UR.at[ur\_index, 'Lots'] = matching\_row['Lots'] + row['Lots'] # 更新 po\_人名\_UR 的 Lots 欄位

df\_name.at[index, 'Lots'] = 0 # 更新 df\_人名 的 Lots 欄位

condition\_met = True

break

# 情況三

elif row['Lots'] < 0 and matching\_row['Lots'] > 0 and abs(row['Lots']) >= abs(matching\_row['Lots']):

matching\_row['MKT Price'] = row['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, pd.DataFrame([matching\_row])], ignore\_index=True)

df\_name.at[index, 'Lots'] = row['Lots'] + matching\_row['Lots'] # 更新 df\_人名 的 Lots 欄位

po\_name\_UR.at[ur\_index, 'Lots'] = 0 # 更新 po\_人名\_UR 的 Lots 欄位

condition\_met = True

break

# 情況四

elif row['Lots'] < 0 and matching\_row['Lots'] > 0 and abs(row['Lots']) < abs(matching\_row['Lots']):

new\_row = matching\_row.copy()

new\_row['Lots'] = row['Lots'] # 更新 Lots 欄位

new\_row['MKT Price'] = row['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, pd.DataFrame([new\_row])], ignore\_index=True)

po\_name\_UR.at[ur\_index, 'Lots'] = matching\_row['Lots'] + row['Lots'] # 更新 po\_人名\_UR 的 Lots 欄位

df\_name.at[index, 'Lots'] = 0 # 更新 df\_人名 的 Lots 欄位

condition\_met = True

break

# 情況五

if not condition\_met:

for index, row in df\_name.iterrows():

if row['Lots'] != 0:

matching\_rows = po\_name\_UR[(po\_name\_UR['Lots'] == 0)]

for ur\_index, matching\_row in matching\_rows.iterrows():

po\_name\_UR.loc[ur\_index] = row # 使用 .loc 來更新 po\_人名\_UR 的整行資料

df\_name.at[index, 'Lots'] = 0 # 更新 df\_人名 的 Lots 欄位

break # 當找到匹配的行並完成更新後，跳出迴圈

return df\_name, po\_name\_UR, po\_name\_Real\_New

for name in names:

df\_name = globals()[f'df\_{name}']

po\_name\_UR = globals()[f'po\_{name}\_UR']

po\_name\_Real\_New = globals()[f'po\_{name}\_Real\_New']

# 大前提和所有情況

df\_name, po\_name\_UR, po\_name\_Real\_New = process\_conditions(df\_name, po\_name\_UR, po\_name\_Real\_New)

globals()[f'po\_{name}\_Real\_New'] = po\_name\_Real\_New

#上面是對從df\_人名到po\_人名\_UR以及po\_人名\_Real\_New的code

#這個是對po\_人名\_UR自己的code

def process\_conditions\_v3(df\_name, po\_name\_UR, po\_name\_Real\_New):

condition\_met = True

while condition\_met:

condition\_met = False

for index, row1 in po\_name\_UR.iterrows():

if row1['Lots'] != 0:

product\_id = row1['ProductID']

broker = row1['Broker']

matching\_rows = po\_name\_UR[(po\_name\_UR['ProductID'] == product\_id) & (po\_name\_UR['Broker'] == broker)]

for ur\_index, row2 in matching\_rows.iterrows():

if row2['Lots'] != 0:

# 情況一

if row1['Lots'] > 0 and row2['Lots'] < 0 and abs(row1['Lots']) >= abs(row2['Lots']):

row3 = row2.copy()

row3['MKT Price'] = row1['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, row3.to\_frame().T], ignore\_index=True)

po\_name\_UR.at[index, 'Lots'] = row1['Lots'] - abs(row2['Lots']) # 更新 po\_人名\_UR 的 Lots 欄位

po\_name\_UR.at[ur\_index, 'Lots'] = 0 # 更新 row2 的 Lots 欄位

condition\_met = True

break

# 情況二

elif row1['Lots'] > 0 and row2['Lots'] < 0 and abs(row1['Lots']) < abs(row2['Lots']):

row3 = row2.copy()

row3['Lots'] = -row1['Lots'] # 更新 Lots 欄位

row3['MKT Price'] = row1['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, row3.to\_frame().T], ignore\_index=True)

po\_name\_UR.at[ur\_index, 'Lots'] = row2['Lots'] + row1['Lots'] # 更新 row2 的 Lots 欄位

po\_name\_UR.at[index, 'Lots'] = 0 # 更新 row1 的 Lots 欄位

condition\_met = True

break

# 情況三

elif row1['Lots'] < 0 and row2['Lots'] > 0 and abs(row1['Lots']) >= abs(row2['Lots']):

row3 = row2.copy()

row3['MKT Price'] = row1['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, row3.to\_frame().T], ignore\_index=True)

po\_name\_UR.at[index, 'Lots'] = row1['Lots'] + row2['Lots'] # 更新 row1 的 Lots 欄位

po\_name\_UR.at[ur\_index, 'Lots'] = 0 # 更新 row2 的 Lots 欄位

condition\_met = True

break

# 情況四

elif row1['Lots'] < 0 and row2['Lots'] > 0 and abs(row1['Lots']) < abs(row2['Lots']):

row3 = row2.copy()

row3['Lots'] = -row1['Lots'] # 更新 Lots 欄位

row3['MKT Price'] = row1['Cost'] # 更新 MKT Price 欄位

po\_name\_Real\_New = pd.concat([po\_name\_Real\_New, row3.to\_frame().T], ignore\_index=True)

po\_name\_UR.at[ur\_index, 'Lots'] = row2['Lots'] + row1['Lots'] # 更新 row2 的 Lots 欄位

po\_name\_UR.at[index, 'Lots'] = 0 # 更新 row1 的 Lots 欄位

condition\_met = True

break

return po\_name\_UR, po\_name\_Real\_New

for name in names:

po\_name\_UR = globals()[f'po\_{name}\_UR']

po\_name\_Real\_New = globals()[f'po\_{name}\_Real\_New']

# 執行新的大前提和情況

po\_name\_UR, po\_name\_Real\_New = process\_conditions\_v3(df\_name, po\_name\_UR, po\_name\_Real\_New)

globals()[f'po\_{name}\_Real\_New'] = po\_name\_Real\_New

#將po\_UR的DV01調整

def update\_dv01(row):

product\_id = row['ProductID']

lots = row['Lots']

if product\_id.startswith('TU'):

return -18.6 \* 2 \* lots

elif product\_id.startswith('FV'):

return -42.7 \* lots

elif product\_id.startswith('TY'):

return -77 \* lots

elif product\_id.startswith('US'):

return -198 \* lots

else:

return row['DV01']

for name in names:

po\_name\_UR = globals()[f'po\_{name}\_UR']

po\_name\_UR['DV01'] = po\_name\_UR.apply(update\_dv01, axis=1)

globals()[f'po\_{name}\_UR'] = po\_name\_UR

#將po\_Real\_New新增欄位MTM P&L，並用公式計算

def calculate\_mtm\_pnl(row):

if row['ProductID'][:2] == "TU":

return (row['MKT Price'] - row['Cost']) \* row['Lots'] \* 2000

else:

return (row['MKT Price'] - row['Cost']) \* row['Lots'] \* 1000

for name in names:

po\_name\_Real\_New = globals()[f'po\_{name}\_Real\_New']

po\_name\_Real\_New['MTM P&L'] = po\_name\_Real\_New.apply(calculate\_mtm\_pnl, axis=1)

globals()[f'po\_{name}\_Real\_New'] = po\_name\_Real\_New

# 將所有 DataFrame 寫入一個新的 Excel 檔案中，並將df\_人名工作表放在所有其他工作表之後。

output\_file\_path = f'data/{today}\_TraderPosition\_Output.xlsx'

with pd.ExcelWriter(output\_file\_path) as writer:

for name in names:

po\_name\_UR = globals()[f'po\_{name}\_UR']

po\_name\_Real\_New = globals()[f'po\_{name}\_Real\_New']

po\_name\_UR.to\_excel(writer, sheet\_name=f'po\_{name}\_UR', index=False)

po\_name\_Real\_New.to\_excel(writer, sheet\_name=f'po\_{name}\_Real\_New', index=False)

for name in names:

df\_name = globals()[f'df\_{name}']

df\_name.to\_excel(writer, sheet\_name=f'df\_{name}', index=False)

幫我寫code我會有一個大前提跟四個情況要你注意，每個情況都要遵照大前提：

大前提：對於每個人的df\_人名 dataframe中，對於每個row中，首先看該row的ProductID column的值，是否在po\_人名\_UR的報表中的ProductID中有一樣的值，若有，則看po\_人名\_UR報表欄位Lots中是否有非0的數值，沒有的話就看下一個有著同樣ProductID的row的Lots是否有非0的數值，再來我也需要Broker column一樣，若在po\_人名\_UR發現ProductID一樣且Broker一樣且Lots有非0數字的話則做以下處理：

情況一：對這個row來說(假設為row1)，若在row1中Lots的值>0而在po\_人名\_UR發現有條row(假設叫row2)，ProductID一樣且Lots<0且絕對值是row1的Lots的值大於等於row2的Lots值時，則將row2新增到po\_人名\_Real\_New(假設新增的那條為row3)，然後將row3的MKT Price column欄位輸入跟row1的Cost column一樣的值，再來就將row1的Lots column值則變成原本row1 Lots值減去row2的Lots column的絕對值，而row2的Lots column值改為0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止。

情況二：對這個row來說(假設為row1)，若在row1中Lots的值>0而在po\_人名\_UR發現有條row(假設叫row2)，ProductID一樣且Lots<0且絕對值是row1的Lots的值小於row2的Lots值時，則將row2新增到po\_人名\_Real\_New(假設為row3)，然後將row3的Lots column值為跟row1的Lots欄位的值正負號相反、而row3的MKT Price column欄位輸入跟row1的Cost欄位一樣的值，再來就將row2的Lots column值變成原本row2的Lots值加上row1的Lots column值，row1的Lots column值則變成0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止。

情況三：對這個row來說(假設為row1)，若在row1中Lots的值<0而在po\_人名\_UR發現有條row(假設叫row2)，ProductID一樣且Lots>0且絕對值是row1的Lots的值大於等於row2的Lots值時，則將row2新增到po\_人名\_Real\_New(假設新增的那條為row3)，然後將row3的MKT Price column欄位輸入跟row1的Cost column一樣的值，再來就將row1的Lots column值則變成原本row1 Lots值加上row2的Lots column的值，而row2的Lots column值改為0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止。

情況二：對這個row來說(假設為row1)，若在row1中Lots的值<0而在po\_人名\_UR發現有條row(假設叫row2)，ProductID一樣且Lots>0且絕對值是row1的Lots的值小於row2的Lots值時，則將row2新增到po\_人名\_Real\_New(假設為row3)，然後將row3的Lots column值為跟row1的Lots欄位的值正負號相反、而row3的MKT Price column欄位輸入跟row1的Cost欄位一樣的值，再來就將row2的Lots column值變成原本row2的Lots值加上row1的Lots column值，row1的Lots column值則變成0，都弄好後，再重複執行以上全部動作直到沒有符合條件為止。

情況五：執行完情況一到四後，若df\_人名dataframe中還有Lots值非為0的row，則對每個row進行以下處理：將該row的ProductID column到Cost column貼到po\_人名\_UR Lots欄位為0的row中，整條貼上去，貼完後df\_人名該row的Lots column值變成0，再重複執行以上