# You

Has power query any functionality like VBA User Defined Function (UDF) whereby it accepts input arguments from user and provides excel output.

See following extract from python script:

url = f'http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F{investingid}%3Fstart-date%3D{from\_date}%26end-date%3D{to\_date}%26time-frame%3DDaily%26add-missing-rows%3Dfalse'

In above variables within curly braces are user input arguments. Is such a thing possible in power query?

# Copilot

Yes, Power Query does have a functionality similar to VBA User Defined Function (UDF). It allows you to create **custom functions**[**1**](https://learn.microsoft.com/en-us/power-query/custom-function). A Power Query custom function is a mapping from a set of input values to a single output value, and is created from native M functions and operators[**1**](https://learn.microsoft.com/en-us/power-query/custom-function).

However, it's important to note that Power Query does not support VBA directly[**2**](https://stackoverflow.com/questions/76802785/how-to-reference-user-defined-vba-functions-in-power-query). If you have custom functions defined in VBA and want to use them in Power Query, you will need to rewrite them in M code, or in R or Python if you are using Power BI[**2**](https://stackoverflow.com/questions/76802785/how-to-reference-user-defined-vba-functions-in-power-query).

In your case, you can create a custom function in Power Query that accepts user input arguments and provides output. The variables within curly braces in your Python script can be replaced with parameters in your Power Query custom function.

Please note that the process of creating custom functions in Power Query is different from creating UDFs in VBA[**1**](https://learn.microsoft.com/en-us/power-query/custom-function)[**4**](https://www.sfmagazine.com/articles/2018/march/access-user-defined-functions/). You might need to familiarize yourself with the Power Query M language and its user interface to effectively create and use custom functions[**1**](https://learn.microsoft.com/en-us/power-query/custom-function).

Source(s)

1. [Using custom functions in Power Query - Power Query](https://learn.microsoft.com/en-us/power-query/custom-function)

2. [How to reference user-defined VBA functions in Power Query](https://stackoverflow.com/questions/76802785/how-to-reference-user-defined-vba-functions-in-power-query)

3. [How To Create User-Defined Functions In VBA (With Examples)](https://zerotomastery.io/blog/user-defined-functions-in-vba/)

4. [Access: User-defined Functions | IMA](https://www.sfmagazine.com/articles/2018/march/access-user-defined-functions/)

# You

I have following python code (UDF). I want that to be converted to power query UDF.

import pandas as pd import datetime import requests import xlwings as xw

@xw.func def get\_url\_data\_id\_new(investingid, from\_date, to\_date): # Convert date strings to the required format from\_date = datetime.datetime.strptime(from\_date, "%d/%m/%Y").strftime("%Y-%m-%d") to\_date = datetime.datetime.strptime(to\_date, "%d/%m/%Y").strftime("%Y-%m-%d")

# Construct the URL

url = f'http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F{investingid}%3Fstart-date%3D{from\_date}%26end-date%3D{to\_date}%26time-frame%3DDaily%26add-missing-rows%3Dfalse'

# Fetch data from the API

response = requests.get(url)

json\_output = response.json()

data\_list = json\_output['data']

# Convert data\_list into a pandas dataframe

df = pd.DataFrame(data\_list)

# Remove commas from integer and float columns

int\_float\_columns = ['last\_close','last\_open','last\_max','last\_min','change\_precent','last\_closeRaw', 'last\_openRaw', 'last\_maxRaw', 'last\_minRaw', 'change\_precentRaw']

for col in int\_float\_columns:

df[col] = df[col].astype(str).str.replace(',', '').astype('float')

# Format columns

df['direction\_color'] = df['direction\_color'].astype(str)

df['rowDate'] = pd.to\_datetime(df['rowDate'], format='%b %d, %Y').apply(lambda x: x.date())

df['rowDateTimestamp'] = pd.to\_datetime(df['rowDateTimestamp']).apply(lambda x: x.date())

# Keep only the columns you want

# columns\_to\_keep = ['direction\_color', 'rowDate', 'rowDateRaw', 'rowDateTimestamp', 'last\_close', 'last\_open', 'last\_max', 'last\_min', 'volume', 'volumeRaw', 'change\_precent']

# df = df[columns\_to\_keep]

# Create a matrix (list of lists) from the dataframe

matrix = df.values.tolist()

# Add headers at the beginning of the list

headers = df.columns.tolist()

matrix.insert(0, headers)

return matrix

As stated above, in url variable, items within curly braces are user inputs.

# Copilot

Here's how you can create a similar function in Power Query. Please note that Power Query doesn't support all the functionalities of Python, so the function might not be exactly the same, but it should give you a similar result.

let

get\_url\_data\_id\_new = (investingid as text, from\_date as date, to\_date as date) =>

let

// Convert dates to the required format

from\_date = Date.ToText(from\_date, "yyyy-MM-dd"),

to\_date = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Format columns

df = Table.TransformColumnTypes(df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df

in

get\_url\_data\_id\_new

This function get\_url\_data\_id\_new takes three parameters: investingid, from\_date, and to\_date. It constructs the URL, fetches data from the API, converts the data into a table, and formats the columns. The resulting table is returned by the function.

Please note that this is a simplified version of your Python function. Some features like removing commas from integer and float columns, and creating a matrix (list of lists) from the dataframe are not included because Power Query handles data differently. You might need to adjust the function according to your specific needs. Also, please replace the email in the URL with your actual email.

Remember to enable the **Fast Combine** option in Power Query to avoid privacy level settings issues when accessing web content. You can do this by going to **File > Options and settings > Query options > Privacy > Ignore the Privacy Levels and potentially improve performance**.

Please note that Power Query doesn't support all the functionalities of Python, so the function might not be exactly the same, but it should give you a similar result. If you need more advanced features, you might consider using Power BI, which supports Python scripts.

I hope this helps! Let me know if you have any other questions. 😊

# You

But whether it is interactive? How it will call inputs from user?

# Copilot

Yes, Power Query custom functions can be interactive and accept inputs from users.

When you define a custom function in Power Query, you specify parameters that the function will accept. These parameters act as placeholders for the values that users will input when they call the function.

Here's an example of how you might call the get\_url\_data\_id\_new function with user inputs:

let

result = get\_url\_data\_id\_new("your\_investingid", #date(2024, 1, 1), #date(2024, 12, 31))

in

result

In this example, "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) are the user inputs for investingid, from\_date, and to\_date respectively.

You can replace "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) with the actual values you want to use.

When you load the query, Power Query will execute the function with the provided inputs and return the result.

Please note that Power Query does not support real-time interaction in the same way that a VBA UserForm or an input box does. The inputs to the function need to be provided when the function is called in the Power Query editor. If you need to frequently change the inputs, you might consider creating parameters in Power Query that you can easily change.

I hope this helps! Let me know if you have any other questions. 😊

# You

There is a way to incorporate:

let result = get\_url\_data\_id\_new("your\_investingid", #date(2024, 1, 1), #date(2024, 12, 31)) in result

within udf by nesting another let statement. Can you structure the query so that aove is covered with UDF?

# Copilot

Yes, you can incorporate the function call within the function definition itself by using another let statement. Here's how you can structure the query:

let

get\_url\_data\_id\_new = (investingid as text, from\_date as date, to\_date as date) =>

let

// Convert dates to the required format

from\_date = Date.ToText(from\_date, "yyyy-MM-dd"),

to\_date = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Format columns

df = Table.TransformColumnTypes(df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}}),

// Call the function with user inputs

result = get\_url\_data\_id\_new("your\_investingid", #date(2024, 1, 1), #date(2024, 12, 31))

in

result

in

get\_url\_data\_id\_new

In this example, "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) are the user inputs for investingid, from\_date, and to\_date respectively. You can replace these with the actual values you want to use.

Please note that this will result in a recursive function call, which might not be what you want. If you want to use the function with different inputs, you should call it outside of its own definition.

I hope this helps! Let me know if you have any other questions. 😊

# You

Okay lets go back to previous option without recursion. When I called the udf by clicking on it in power query editor, it asked for arguments which I gave: 18343 for investingid, date from: 01/01/2018 and date to: 09/04/2024. But it resulted in error:

An error occurred in the ‘Web Scraping’ query. DataFormat.Error: We reached the end of the buffer.

Problem seems to be date input. Can you include command in udf to convert dd/mm/yyyy format ted date input into yyyy-mm-dd format both from and to dates?

# Copilot

Sure, I can help with that. In Power Query, you can use the Date.ToText function to convert a date to a text string in the "yyyy-MM-dd" format. However, if your input dates are in the "dd/MM/yyyy" format, you'll first need to convert them to the date type before using Date.ToText.

Here's how you can modify the get\_url\_data\_id\_new function to accept dates in the "dd/MM/yyyy" format:

let

get\_url\_data\_id\_new = (investingid as text, from\_date as text, to\_date as text) =>

let

// Convert date strings to the required format

from\_date = Date.ToText(DateTime.FromText(from\_date), "yyyy-MM-dd"),

to\_date = Date.ToText(DateTime.FromText(to\_date), "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Format columns

df = Table.TransformColumnTypes(df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df

in

get\_url\_data\_id\_new

In this version of the function, from\_date and to\_date are expected to be text strings in the "dd/MM/yyyy" format. The DateTime.FromText function is used to convert these strings to the datetime type, and then Date.ToText is used to convert them to the "yyyy-MM-dd" format.

Please replace the email in the URL with your actual email.

I hope this helps! Let me know if you have any other questions. 😊

# You

See the error in above code:

An error occurred in the ‘Web Scraping’ query. Expression.Error: We cannot convert the value #datetime(2018, 1, 1, 0, 0, 0) to type Date. Details: Value=01-01-2018 00:00:00 Type=[Type]

Why time has also been included? 00:00:00

# Copilot

I apologize for the confusion. The DateTime.FromText function in Power Query returns a datetime value, which includes both a date and a time. When you convert a date string to a datetime value, the time is set to 00:00:00 by default if no time is specified.

To get a date value without a time, you can use the Date.FromText function instead. Here's how you can modify the get\_url\_data\_id\_new function:

let

get\_url\_data\_id\_new = (investingid as text, from\_date as text, to\_date as text) =>

let

// Convert date strings to the required format

from\_date = Date.ToText(Date.FromText(from\_date), "yyyy-MM-dd"),

to\_date = Date.ToText(Date.FromText(to\_date), "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Format columns

df = Table.TransformColumnTypes(df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df

in

get\_url\_data\_id\_new

In this version of the function, from\_date and to\_date are expected to be text strings in the "dd/MM/yyyy" format. The Date.FromText function is used to convert these strings to the date type, and then Date.ToText is used to convert them to the "yyyy-MM-dd" format.

Please replace the email in the URL with your actual email.

I hope this helps! Let me know if you have any other questions. 😊

# You

Can you break UDF below in power query because I want to debug error and want step wise output for each step.

let get\_url\_data\_id\_new = (investingid as text, from\_date as date, to\_date as date) => let // Convert dates to the required format from\_date = Date.ToText(from\_date, "yyyy-MM-dd"), to\_date = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Format columns

df2 = Table.TransformColumnTypes(df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df2

in get\_url\_data\_id\_new

# Copilot

Sure, I can help you break down the function into smaller steps for debugging. Here's how you can structure the query:

let

// Define your parameters

investingid = "your\_investingid",

from\_date = #date(2024, 1, 1),

to\_date = #date(2024, 12, 31),

// Convert dates to the required format

from\_date\_text = Date.ToText(from\_date, "yyyy-MM-dd"),

to\_date\_text = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date\_text & "%26end-date%3D" & to\_date\_text & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Format columns

df2 = Table.TransformColumnTypes(df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df2

In this version of the query, each step is separated into its own line, and the results of each step are stored in a separate variable. This should make it easier to debug the query and see the output of each step.

Please replace "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) with the actual values you want to use. Also, replace the email in the URL with your actual email.

I hope this helps! Let me know if you have any other questions. 😊

# You

You missed the step of expanding list of records (df is one column with all rows showing Record)to columns and rows in above.

# Copilot

I apologize for the oversight. You're correct, the data returned from the API is a list of records, and we need to expand these records into separate columns. Here's how you can modify the query to include this step:

let

// Define your parameters

investingid = "your\_investingid",

from\_date = #date(2024, 1, 1),

to\_date = #date(2024, 12, 31),

// Convert dates to the required format

from\_date\_text = Date.ToText(from\_date, "yyyy-MM-dd"),

to\_date\_text = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date\_text & "%26end-date%3D" & to\_date\_text & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

// Format columns

df2 = Table.TransformColumnTypes(expanded\_df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df2

In this version of the query, after converting data\_list into a table (df), we use the Table.ExpandRecordColumn function to expand the records in the "Column1" column into separate columns. The second argument to Table.ExpandRecordColumn is the name of the column to expand, and the third argument is a list of the field names to include in the output table.

Please replace "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) with the actual values you want to use. Also, replace the email in the URL with your actual email.

I hope this helps! Let me know if you have any other questions. 😊

# You

One column: rowDateTimestamp shows error in all rows: Error is: DataFormat.Error: We couldn't parse the input provided as a Date value. Details: 2024-04-09T00:00:00Z

You have to convert text like above into date format for that column.

# Copilot

I apologize for the oversight. The rowDateTimestamp column contains date and time information in the ISO 8601 format, which includes a "Z" at the end to indicate that the time is in Coordinated Universal Time (UTC).

In Power Query, you can use the DateTimeZone.FromText function to convert this text into a datetimezone value, and then use the DateTimeZone.ToLocal function to convert it to your local time zone. If you only want the date part, you can use the DateTime.Date function to extract the date from the datetimezone value.

Here's how you can modify the query to include this step:

let

// Define your parameters

investingid = "your\_investingid",

from\_date = #date(2024, 1, 1),

to\_date = #date(2024, 12, 31),

// Convert dates to the required format

from\_date\_text = Date.ToText(from\_date, "yyyy-MM-dd"),

to\_date\_text = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date\_text & "%26end-date%3D" & to\_date\_text & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

// Convert rowDateTimestamp to date

converted\_df = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.From(DateTimeZone.ToLocal(DateTimeZone.FromText(\_))), type date}),

// Format columns

df2 = Table.TransformColumnTypes(converted\_df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df2

In this version of the query, after expanding the records into separate columns, we use the Table.TransformColumns function to convert the rowDateTimestamp column to a date type. The DateTimeZone.FromText, DateTimeZone.ToLocal, and Date.From functions are used to convert the text in the rowDateTimestamp column to a date value.

Please replace "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) with the actual values you want to use. Also, replace the email in the URL with your actual email.

I hope this helps! Let me know if you have any other questions. 😊

# You

Can you not integrate: // Convert rowDateTimestamp to date converted\_df = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.From(DateTimeZone.ToLocal(DateTimeZone.FromText(\_))), type date}),

Within the last step here:

{"rowDateTimestamp", type date}?

# Copilot

I understand your question. However, the {"rowDateTimestamp", type date} syntax in Power Query is used to change the type of an existing column to a date. It doesn't perform any transformations on the data in the column.

In your case, the rowDateTimestamp column contains date and time information in the ISO 8601 format, which includes a "Z" at the end to indicate that the time is in Coordinated Universal Time (UTC). This format is not directly convertible to a date using the type date syntax.

That's why we need to use the Table.TransformColumns function with DateTimeZone.FromText, DateTimeZone.ToLocal, and Date.From to convert the text in the rowDateTimestamp column to a date value.

So, unfortunately, we can't integrate the transformation of rowDateTimestamp into the Table.TransformColumnTypes step. They need to be two separate steps.

I hope this clarifies your question. Let me know if you have any other questions! 😊

# You

Let's resolve that issue another way. Just strip chracters: T00:00:00Z and then convert to date using a nested function within {"rowDateTimestamp", type date}. What I mean is you wrap "rowDateTimestamp" with some string function to strip additional charcters. Would that resolve the issue?

# Copilot

Yes, you can use the Text.Start function in Power Query to strip the "T00:00:00Z" part from the rowDateTimestamp column. This function returns the first n characters of a text value.

Here's how you can modify the query to include this step:

let

// Define your parameters

investingid = "your\_investingid",

from\_date = #date(2024, 1, 1),

to\_date = #date(2024, 12, 31),

// Convert dates to the required format

from\_date\_text = Date.ToText(from\_date, "yyyy-MM-dd"),

to\_date\_text = Date.ToText(to\_date, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date\_text & "%26end-date%3D" & to\_date\_text & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

// Strip "T00:00:00Z" from rowDateTimestamp and convert to date

stripped\_df = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.FromText(Text.Start(\_, 10)), type date}),

// Format columns

df2 = Table.TransformColumnTypes(stripped\_df,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}})

in

df2

In this version of the query, after expanding the records into separate columns, we use the Table.TransformColumns function with Text.Start and Date.FromText to strip the "T00:00:00Z" part from the rowDateTimestamp column and convert the remaining text to a date value.

Please replace "your\_investingid", #date(2024, 1, 1), and #date(2024, 12, 31) with the actual values you want to use. Also, replace the email in the URL with your actual email.

I hope this helps! Let me know if you have any other questions. 😊

# You

Why don't you use text.start within {"rowDateTimestamp", type date} to avoid one additional step?

# Copilot

I apologize for any confusion, but the Table.TransformColumnTypes function in Power Query is used to change the data type of existing columns. It doesn't support transformations on the data within the columns.

The syntax {"rowDateTimestamp", type date} is used to change the type of the rowDateTimestamp column to a date. It doesn't perform any transformations on the data in the column.

That's why we need to use the Table.TransformColumns function with Text.Start and Date.FromText to strip the "T00:00:00Z" part from the rowDateTimestamp column and convert the remaining text to a date value. This operation cannot be performed within the Table.TransformColumnTypes function.

I hope this clarifies your question. Let me know if you have any other questions! 😊

# You

Why doing like this: {Text.Start([rowDateTimestamp], 10), type date} results in error:

Expression.Error: There is an unknown identifier. Did you use the [field] shorthand for a \_[field] outside of an 'each' expression?

Somewhere I need to include each.

# Copilot

You're correct. In Power Query, when you're transforming a column with a function that operates on each individual value, you need to use the each keyword. The each keyword is used to define a function that transforms each value in the column.

Here's how you can modify the transformation to include the each keyword:

Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.FromText(Text.Start(\_, 10)), type date})

In this line of code, each defines a function that takes a single argument (\_), applies the Text.Start function to get the first 10 characters of the text, and then converts the result to a date with Date.FromText.

I hope this helps! Let me know if you have any other questions. 😊

# You

I want to convert text like "01/01/2018" to like this "2018-01-01". What is proper power query function to do that?

# Copilot

In Power Query, you can use the Date.ToText function to convert a date to a text string in a specific format. However, if your input is a text string in the "dd/MM/yyyy" format, you'll first need to convert it to a date type using the Date.FromText function.

Here's how you can convert a date string from the "dd/MM/yyyy" format to the "yyyy-MM-dd" format:

let

// Define your date string

date\_string = "01/01/2018",

// Convert the date string to a date

date = Date.FromText(date\_string),

// Convert the date to a text string in the "yyyy-MM-dd" format

formatted\_date\_string = Date.ToText(date, "yyyy-MM-dd")

in

formatted\_date\_string

In this example, date\_string is your input date string in the "dd/MM/yyyy" format. The Date.FromText function is used to convert this string to a date type, and then Date.ToText is used to convert it to a text string in the "yyyy-MM-dd" format.

The result, formatted\_date\_string, is the converted date string in the "yyyy-MM-dd" format.

I hope this helps! Let me know if you have any other questions. 😊

# You

Okay let's see another example where UDF is used within power query withour cyclical reference error:

let fn\_aging=(days as number)=>let input={ {days<=30,"Up To 30 Days"}, {days>30 and days<=60,"31-60 Days"}, {days>61 and days<=90,"61-90 Days"}, {days>90 and days<=180,"91-180 Days"}, {days>180 and days<=365,"6 Months To 1 Year"}, {days>365,"More Than 1 Year"}, {Logical.From(days)=true,"Error"} }, output=List.First(List.Select(input,each \_{0})){1} in output, Source = Excel.Workbook(File.Contents("D:\OneDrive - 0yt2k\Excel Examples Including add-in xlwings and connected python - vba - dlls, etc\aging buckets power query example.xlsm"), null, true), aging\_days\_range\_DefinedName = Source{[Item="aging\_days\_range",Kind="DefinedName"]}[Data], #"Promoted Headers" = Table.PromoteHeaders(aging\_days\_range\_DefinedName, [PromoteAllScalars=true]), #"Changed Type" = Table.TransformColumnTypes(#"Promoted Headers",{{"Days", Int64.Type}}), #"Added Custom" = Table.AddColumn(#"Changed Type", "Aging Buckets", each fn\_aging([Days])), #"Removed Columns" = Table.RemoveColumns(#"Added Custom",{"Days"}) in #"Removed Columns"

No cyclical reference error in above. Works perfactly. Why that error in code below:

let get\_url\_data\_id\_new\_2 = (investingid as text, from\_date as text, to\_date as text) => let // Convert dates to the required format date\_from\_text = Date.FromText(from\_date), date\_to\_text =Date.FromText(to\_date), from\_date = Date.ToText(date\_from\_text, "yyyy-MM-dd"), to\_date = Date.ToText(date\_to\_text, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

date\_transformation = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.FromText(Text.Start(\_, 10)), type date}),

// Format columns

df2 = Table.TransformColumnTypes(date\_transformation,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}}),

#"Reordered Columns" = Table.ReorderColumns(df2,{"direction\_color", "rowDate", "rowDateTimestamp", "last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw"})

in #"Reordered Columns", invoked\_function = get\_url\_data\_id\_new\_2("18343","01/01/2018","31/03/2024") in invoked\_function

How to make this code work as previous one. Can some placeholder or variable be useful so that I can call function in similar way as previous code?

# Copilot

The cyclical reference error occurs because the function get\_url\_data\_id\_new\_2 is being invoked within its own definition. This creates a loop where the function is continuously calling itself, hence the cyclical reference.

To avoid this, you should invoke the function outside of its own definition, similar to how the fn\_aging function is invoked in the first example you provided.

Here's how you can modify your code to avoid the cyclical reference error:

let

get\_url\_data\_id\_new\_2 = (investingid as text, from\_date as text, to\_date as text) =>

let

// Convert dates to the required format

date\_from\_text = Date.FromText(from\_date),

date\_to\_text = Date.FromText(to\_date),

from\_date = Date.ToText(date\_from\_text, "yyyy-MM-dd"),

to\_date = Date.ToText(date\_to\_text, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

// Transform rowDateTimestamp to date

date\_transformation = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.FromText(Text.Start(\_, 10)), type date}),

// Format columns

df2 = Table.TransformColumnTypes(date\_transformation,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}}),

#"Reordered Columns" = Table.ReorderColumns(df2,{"direction\_color", "rowDate", "rowDateTimestamp", "last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw"})

in

#"Reordered Columns",

// Invoke the function outside of its definition

invoked\_function = get\_url\_data\_id\_new\_2("18343","01/01/2018","31/03/2024")

in

invoked\_function

In this version of the code, the get\_url\_data\_id\_new\_2 function is invoked outside of its own definition, which avoids the cyclical reference error.

I hope this helps! Let me know if you have any other questions. 😊

# You

Are you sure about your above code. See my provided code. It is similar to yours. Again reproducing below:

let get\_url\_data\_id\_new\_2 = (investingid as text, from\_date as text, to\_date as text) => let // Convert dates to the required format date\_from\_text = Date.FromText(from\_date), date\_to\_text =Date.FromText(to\_date), from\_date = Date.ToText(date\_from\_text, "yyyy-MM-dd"), to\_date = Date.ToText(date\_to\_text, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

date\_transformation = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.FromText(Text.Start(\_, 10)), type date}),

// Format columns

df2 = Table.TransformColumnTypes(date\_transformation,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}}),

#"Reordered Columns" = Table.ReorderColumns(df2,{"direction\_color", "rowDate", "rowDateTimestamp", "last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw"})

in #"Reordered Columns", invoked\_function = get\_url\_data\_id\_new\_2("18343","01/01/2018","31/03/2024") in invoked\_function

# Copilot

I apologize for the confusion. The cyclical reference error occurs because the function get\_url\_data\_id\_new\_2 is being invoked within its own definition. This creates a loop where the function is continuously calling itself, hence the cyclical reference.

To avoid this, you should invoke the function outside of its own definition. Here's how you can modify your code to avoid the cyclical reference error:

let

get\_url\_data\_id\_new\_2 = (investingid as text, from\_date as text, to\_date as text) =>

let

// Convert dates to the required format

date\_from\_text = Date.FromText(from\_date),

date\_to\_text =Date.FromText(to\_date),

from\_date = Date.ToText(date\_from\_text, "yyyy-MM-dd"),

to\_date = Date.ToText(date\_to\_text, "yyyy-MM-dd"),

// Construct the URL

url = "http://api.scraperlink.com/investpy/?email=asharindani51@gmail.com&url=https%3A%2F%2Fapi.investing.com%2Fapi%2Ffinancialdata%2Fhistorical%2F" & investingid & "%3Fstart-date%3D" & from\_date & "%26end-date%3D" & to\_date & "%26time-frame%3DDaily%26add-missing-rows%3Dfalse",

// Fetch data from the API

source = Json.Document(Web.Contents(url)),

data\_list = source[data],

// Convert data\_list into a table

df = Table.FromList(data\_list, Splitter.SplitByNothing(), null, null, ExtraValues.Error),

// Expand the records into separate columns

expanded\_df = Table.ExpandRecordColumn(df, "Column1", {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}, {"last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw", "direction\_color", "rowDate", "rowDateTimestamp"}),

// Transform rowDateTimestamp to date

date\_transformation = Table.TransformColumns(expanded\_df, {"rowDateTimestamp", each Date.FromText(Text.Start(\_, 10)), type date}),

// Format columns

df2 = Table.TransformColumnTypes(date\_transformation,{{"last\_close", type number}, {"last\_open", type number}, {"last\_max", type number}, {"last\_min", type number}, {"change\_precent", type number}, {"last\_closeRaw", type number}, {"last\_openRaw", type number}, {"last\_maxRaw", type number}, {"last\_minRaw", type number}, {"change\_precentRaw", type number}, {"direction\_color", type text}, {"rowDate", type date}, {"rowDateTimestamp", type date}}),

#"Reordered Columns" = Table.ReorderColumns(df2,{"direction\_color", "rowDate", "rowDateTimestamp", "last\_close", "last\_open", "last\_max", "last\_min", "change\_precent", "last\_closeRaw", "last\_openRaw", "last\_maxRaw", "last\_minRaw", "change\_precentRaw"})

in

#"Reordered Columns",

// Invoke the function outside of its definition

invoked\_function = get\_url\_data\_id\_new\_2("18343","01/01/2018","31/03/2024")

in

invoked\_function

In this version of the code, the get\_url\_data\_id\_new\_2 function is invoked outside of its own definition, which avoids the cyclical reference error.

I hope this helps! Let me know if you have any other questions. 😊

# You

Is there a way to copy conditional formatting rule from one tab to another (I mean on worksheet in excel to another) like copy / paste?

# Copilot

Yes, you can copy conditional formatting rules from one worksheet to another in Excel. Here's how you can do it:

1. Select the cells with the conditional formatting rules you want to copy.
2. Press Ctrl+C to copy the cells.
3. Go to the other worksheet where you want to apply the conditional formatting.
4. Select the cell or range of cells where you want to paste the conditional formatting.
5. Click on the arrow below the Paste button on the Home tab, then select Paste Special.
6. In the Paste Special dialog box, select Formats, then click OK.

This will copy the conditional formatting rules from the cells in the first worksheet to the cells in the second worksheet.

Please note that this will also copy other formatting (like font color, cell color, etc.) from the source cells to the destination cells. If you only want to copy the conditional formatting, you'll need to clear the other formatting from the destination cells after pasting.

I hope this helps! Let me know if you have any other questions. 😊