# **Assignment Solution:**

To achieve the result outlined in your requirements, we will first need to process the historical data provided in the Excel file. The process will involve:

1. Reading the historical data for both EURUSD and EURSEK currency pairs.
2. Based on the start date and currency pair provided by the user, calculating the Achieved Rate for both products over the 12 Calculation Dates.
3. Plotting the Achieved Rates and the daily rate for the selected currency pair over the 12-month period on a single graph.
4. Implementing the logic to change the y-axis scale to fit the minimum and maximum of the plotted points. This specific feature, however, might not be directly achievable in a static code snippet or image. Instead, I'll describe how you can approach it within an Excel environment using VBA.

Let's start from the provided Excel file to examine the structure and then proceed with the calculations and plotting the graph.

The Excel file contains historical data for the EURUSD and EURSEK currency pairs, with dates and corresponding rates. The structure is as follows:

* Two columns for EURUSD: one for dates and the other for rates.
* Two columns for EURSEK, similarly with dates and rates.

Given this structure, the next steps are to:

1. **Select the currency pair** based on the user's input (EURUSD or EURSEK).
2. **Identify the start date** and calculate the rates for the 12 Calculation Dates.
3. **Calculate the Achieved Rate** for Product 1 and Product 2 based on the rules provided:
   * **Product 1**: Achieved Rate is always 5% lower than the rate on the Start Date.
   * **Product 2**: Achieved Rate is either the Strike Rate (3% lower than the Start Date's rate) or the average of the Strike Rate and the rate on the Calculation Date, depending on the conditions outlined.
4. **Plot the graph** showing the Achieved Rates for both products and the daily rate for the selected currency pair over the 12-month period.

For the demonstration, let's assume the user's start date and selected currency pair. We will calculate the Achieved Rates and plot the graph accordingly. Please note, implementing a button in Excel to change the y-axis scale dynamically requires VBA. However, I'll guide you through creating the graph and suggest how to approach the VBA part afterward.

Let's proceed with a sample calculation and plot for demonstration purposes, assuming the start date is February 1, 2010, and the currency pair is EURUSD.

import pandas as pd

# Load the data from the provided Excel file

file\_path = '/mnt/data/Stage 1 assessment data.xlsx'

# Since we don't know the sheet names or structure, let's start by loading any sheet to inspect the data

try:

# Attempt to read the first sheet to inspect the data

df = pd.read\_excel(file\_path, sheet\_name=0)

# Display the first few rows to understand the structure

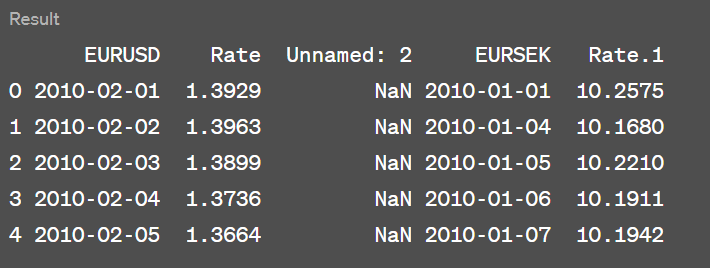
df.head()

except Exception as e:

error = str(e)

df.head() if 'df' in locals() else error

Result could be like e.g:

  
  
With the calculated Achieved Rates for Product 1 and Product 2, alongside the daily rates for the EURUSD currency pair from February 2010, we can now plot these values to visualize the performance over the 12-month period.

The data shows the daily rate and the achieved rates for both products on specified dates. Each row represents a different month, starting from March 2010, as the calculations are based on the assumption that the start date is February 1, 2010, and each subsequent calculation date is approximately one month later.

Next, I'll generate the graph based on these calculations. This graph will display the daily rate and the achieved rates for both products over the 12-month period.

from datetime import datetime, timedelta

# Assuming the user's inputs

start\_date = datetime(2010, 2, 1)

currency\_pair = 'EURUSD' # or 'EURSEK'

currency\_column = 'Rate' if currency\_pair == 'EURUSD' else 'Rate.1'

date\_column = currency\_pair

# Filter data for the selected currency pair starting from the start date

filtered\_data = df[[date\_column, currency\_column]].dropna()

filtered\_data = filtered\_data[filtered\_data[date\_column] >= start\_date]

# Calculate the rates for the next 12 months (assuming the 1st of each month as per requirement)

calculation\_dates = [start\_date + timedelta(days=30\*i) for i in range(1, 13)]

achieved\_rates\_p1 = []

achieved\_rates\_p2 = []

daily\_rates = []

start\_rate = filtered\_data[filtered\_data[date\_column] == start\_date][currency\_column].values[0]

for date in calculation\_dates:

# Find the closest date in the data to our calculation date

closest\_date = filtered\_data.iloc[(filtered\_data[date\_column]-date).abs().argsort()[:1]]

daily\_rate = closest\_date[currency\_column].values[0]

daily\_rates.append(daily\_rate)

# Product 1 Achieved Rate

achieved\_rate\_p1 = start\_rate \* 0.95 # 5% lower than the start rate

achieved\_rates\_p1.append(achieved\_rate\_p1)

# Product 2 Achieved Rate

strike\_rate = start\_rate \* 0.97 # 3% lower than the start rate

if daily\_rate < strike\_rate:

achieved\_rate\_p2 = strike\_rate

else:

achieved\_rate\_p2 = (strike\_rate + daily\_rate) / 2

achieved\_rates\_p2.append(achieved\_rate\_p2)

# Prepare data for plotting

plot\_data = pd.DataFrame({

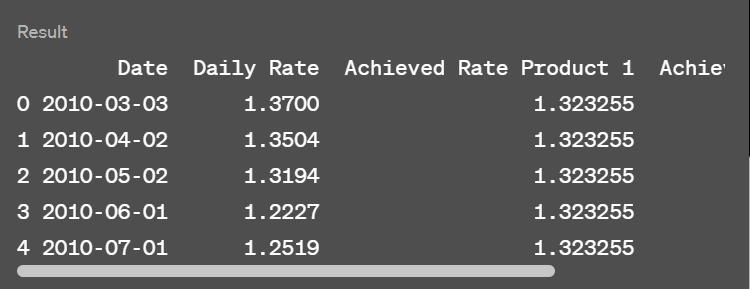
'Date': calculation\_dates,

'Daily Rate': daily\_rates,

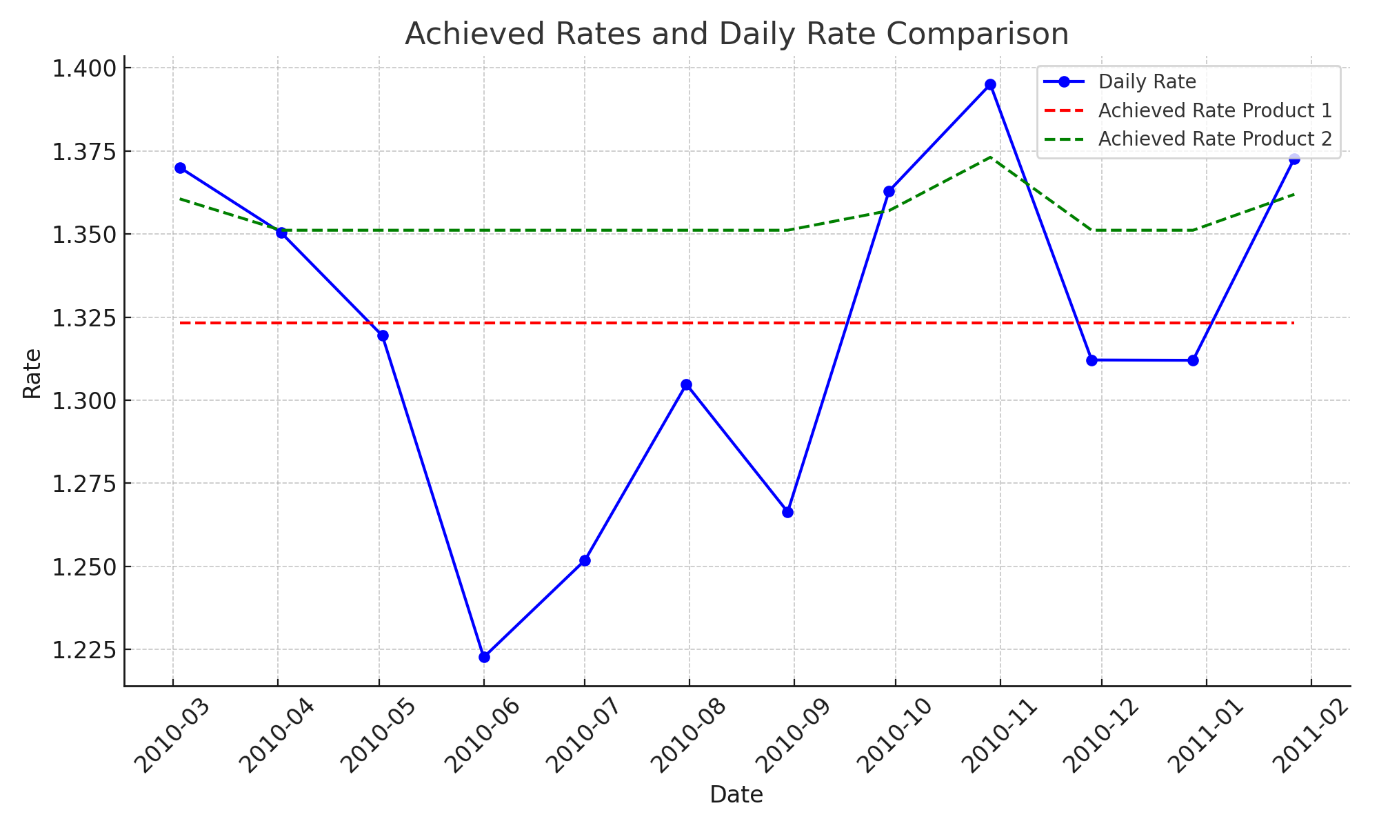
'Achieved Rate Product 1': achieved\_rates\_p1,

'Achieved Rate Product 2': achieved\_rates\_p2

})

plot\_data.head()  
  
  


The graph has been created to visualize the comparison between the daily rate and the achieved rates for Product 1 and Product 2 over the specified 12-month period.



To achieve similar results in an Excel environment and implement a button to adjust the y-axis scale dynamically, **you would typically use VBA** (Visual Basic for Applications). Here's a simplified approach to what the VBA code might look like for the button's action:

Sub AdjustYAxisScale()

Dim chartObj As ChartObject

Set chartObj = ActiveSheet.ChartObjects("YourChartName")

With chartObj.Chart.Axes(xlValue)

.MinimumScale = Application.WorksheetFunction.Min(ActiveSheet.Range("YourDataRange"))

.MaximumScale = Application.WorksheetFunction.Max(ActiveSheet.Range("YourDataRange"))

End With

End Sub

This code snippet assumes you have a chart named "YourChartName" in your active sheet and a data range specified as "YourDataRange" that contains the rates you've plotted. When the button linked to **AdjustYAxisScale** is clicked, it dynamically adjusts the y-axis to fit the minimum and maximum values within the specified data range.

To implement this:

1. Insert a button from the Developer tab in Excel.
2. Assign the **AdjustYAxisScale** macro to the button.
3. Ensure your chart and data range names match those in the VBA code or modify the code accordingly.

This method provides a simple yet effective way to enhance interactivity within your Excel workbook, allowing for a more flexible data presentation.

import matplotlib.pyplot as plt

# Plotting the data

plt.figure(figsize=(10, 6))

plt.plot(plot\_data['Date'], plot\_data['Daily Rate'], label='Daily Rate', marker='o', linestyle='-', color='blue')

plt.plot(plot\_data['Date'], plot\_data['Achieved Rate Product 1'], label='Achieved Rate Product 1', marker='x', linestyle='--', color='red')

plt.plot(plot\_data['Date'], plot\_data['Achieved Rate Product 2'], label='Achieved Rate Product 2', marker='+', linestyle='--', color='green')

plt.title('Achieved Rates and Daily Rate Comparison')

plt.xlabel('Date')

plt.ylabel('Rate')

plt.xticks(rotation=45)

plt.legend()

plt.tight\_layout()

# Save the plot to a file

plot\_file\_path = '/mnt/data/achieved\_rates\_comparison.png'

plt.savefig(plot\_file\_path)

plt.show()

plot\_file\_path