**Python Code für Filialanalyse in Excel**

import pandas as pd

from openpyxl import load\_workbook

from openpyxl.utils.dataframe import dataframe\_to\_rows

from openpyxl import Workbook

# === 1. Pfad und Dateinamen ===

pfad = r'C:\Hier Ihren Pfad eingeben'

ist\_datei = f'{pfad}\\Ist Filialen.xlsx'

plan\_datei = f'{pfad}\\Plan Filialen.xlsx'

output\_datei = f'{pfad}\\Auswertung Filialen.xlsx'

# === 2. Excel-Dateien einlesen ===

ist\_df = pd.read\_excel(ist\_datei, sheet\_name='Tabelle1', skiprows=1, names=['Filiale', 'Ist'])

plan\_df = pd.read\_excel(plan\_datei, sheet\_name='Tabelle1', skiprows=1, names=['Filiale', 'Plan'])

# === 3. Daten zusammenführen und durch 1000 teilen ===

df = pd.merge(ist\_df, plan\_df, on='Filiale')

df[['Ist', 'Plan']] = df[['Ist', 'Plan']] / 1000

# === 4. Abweichung berechnen ===

df['Abw €'] = df['Ist'] - df['Plan']

df['Abw %'] = (df['Abw €'] / df['Plan']).replace([float('inf'), -float('inf')], 0)

# === 5. Gesamtsummen berechnen ===

gesamt\_ist = df['Ist'].sum()

gesamt\_plan = df['Plan'].sum()

gesamt\_abw\_eur = gesamt\_ist - gesamt\_plan

gesamt\_abw\_pct = gesamt\_abw\_eur / gesamt\_plan if gesamt\_plan != 0 else 0

# === 6. Top 5 Filialen mit negativster Abweichung (in €) ===

top5\_negativ = df.nsmallest(5, 'Abw €')

# === 7. Neue Excel-Datei vorbereiten ===

wb = Workbook()

wb.remove(wb.active)

# Sheet 1: Gesamtsummen

sheet\_sum = wb.create\_sheet("Gesamtsummen")

sheet\_sum.append(['Kennzahl', 'Wert'])

sheet\_sum.append(['Gesamt IST', gesamt\_ist])

sheet\_sum.append(['Gesamt PLAN', gesamt\_plan])

sheet\_sum.append(['Abweichung €', gesamt\_abw\_eur])

sheet\_sum.append(['Abweichung %', gesamt\_abw\_pct])

# Sheet 2: Top 5 negativ

sheet\_top5 = wb.create\_sheet("Top 5 negativ")

sheet\_top5.append(df.columns.tolist())

for row in dataframe\_to\_rows(top5\_negativ, index=False, header=False):

 sheet\_top5.append(row)

# Sheet 3: Alle Filialen

sheet\_all = wb.create\_sheet("Alle Filialen")

sheet\_all.append(df.columns.tolist())

for row in dataframe\_to\_rows(df, index=False, header=False):

 sheet\_all.append(row)

# === 8. Excel-Formatierung anwenden ===

wb.save(output\_datei) # Erst speichern, dann erneut laden für Formatierung

wb = load\_workbook(output\_datei)

def format\_sheet(sheet, euro\_rows=None, percent\_rows=None, col\_index=1):

 for row\_idx, row in enumerate(sheet.iter\_rows(min\_row=2), start=2):

 cell = row[col\_index]

 if euro\_rows and row\_idx in euro\_rows:

 cell.number\_format = '#,##0'

 if percent\_rows and row\_idx in percent\_rows:

 cell.number\_format = '0%'

# Formatierung für „Gesamtsummen“: Zeilen 2-4 als Euro, Zeile 5 als Prozent

sheet\_sum = wb['Gesamtsummen']

format\_sheet(sheet\_sum, euro\_rows=[2, 3, 4], percent\_rows=[5], col\_index=1)

# Formatierung für Top 5 negativ und Alle Filialen

for sheet\_name in ['Top 5 negativ', 'Alle Filialen']:

 sheet = wb[sheet\_name]

 for row in sheet.iter\_rows(min\_row=2):

 if len(row) >= 5:

 row[1].number\_format = '#,##0' # Ist

 row[2].number\_format = '#,##0' # Plan

 row[3].number\_format = '#,##0' # Abw €

 row[4].number\_format = '0%' # Abw %

# === 9. Excel-Datei speichern ===

wb.save(output\_datei)

print("✅ Auswertung mit korrekter Formatierung erstellt unter:\n", output\_datei)