

```

# Install required packages
!pip install -q pandas numpy matplotlib seaborn scikit-learn openpyxl

# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from google.colab import files

# Upload Excel file
uploaded = files.upload()
filename = next(iter(uploaded))

# Load data
df = pd.read_excel(filename)

# Clean column names
df.columns = df.columns.str.strip().str.replace(' ', '').str.lower()
print("Columns:", df.columns.tolist())

# Check necessary columns
required = ['customerid', 'invoice', 'invoicedate', 'quantity', 'price']
for col in required:
    if col not in df.columns:
        raise ValueError(f"Missing column: {col}")

# Remove invalid rows
df.dropna(inplace=True)
df = df[(df['quantity'] > 0) & (df['price'] > 0)]

# Add totalprice
df['totalprice'] = df['quantity'] * df['price']

# Convert InvoiceDate to datetime
df['invoicedate'] = pd.to_datetime(df['invoicedate'])

# Reference date = max date + 1
ref_date = df['invoicedate'].max() + pd.Timedelta(days=1)

# RFM table
rfm = df.groupby('customerid').agg({
    'invoicedate': lambda x: (ref_date - x.max()).days,
    'invoiceno': 'nunique',
    'totalprice': 'sum'
}).reset_index()

# Rename columns
rfm.columns = ['CustomerID', 'Recency', 'Frequency', 'Monetary']

# Standardize features

```

```

X = rfm[['Recency', 'Frequency', 'Monetary']]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Elbow method
wcss = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

# Plot elbow
plt.figure(figsize=(8,5))
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method for Optimal Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.grid(True)
plt.show()

# Final clustering
kmeans = KMeans(n_clusters=4, random_state=42)
rfm['Cluster'] = kmeans.fit_predict(X_scaled)

# 2D Plot
plt.figure(figsize=(8,6))
sns.scatterplot(data=rfm, x='Recency', y='Monetary', hue='Cluster', palette='Set2',
s=100)
plt.title('2D RFM Segmentation')
plt.xlabel('Recency (days)')
plt.ylabel('Monetary (Total Spent)')
plt.grid(True)
plt.show()

# 3D Plot
fig = plt.figure(figsize=(10,7))
ax = fig.add_subplot(111, projection='3d')
scatter = ax.scatter(
    rfm['Recency'], rfm['Frequency'], rfm['Monetary'],
    c=rfm['Cluster'], cmap='Set2', s=60
)
ax.set_title('3D RFM Clustering')
ax.set_xlabel('Recency')
ax.set_ylabel('Frequency')
ax.set_zlabel('Monetary')
plt.colorbar(scatter, ax=ax, label='Cluster')
plt.show()

# Output sample
print("\nSample of RFM Clusters:")
print(rfm.head())

```