```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import zipfile
with zipfile.ZipFile('/content/Online Retail.xlsx.zip', "r") as z:
    z.extractall("data")
df = pd.read_excel("data/Online Retail.xlsx")
print("Shape:", df.shape)
df.head()
```

```
Shape: (541909, 8)
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom

```
df = df.dropna(subset=["CustomerID"])
df = df[df["Quantity"] > 0]
df["TotalAmount"] = df["Quantity"] * df["UnitPrice"]
df["InvoiceDate"] = pd.to_datetime(df["InvoiceDate"])
print("Cleaned Data Shape:", df.shape)
df.head()
```

```
Cleaned Data Shape: (397924, 9)
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	TotalAmount
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	15.30
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	22.00
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	20.34

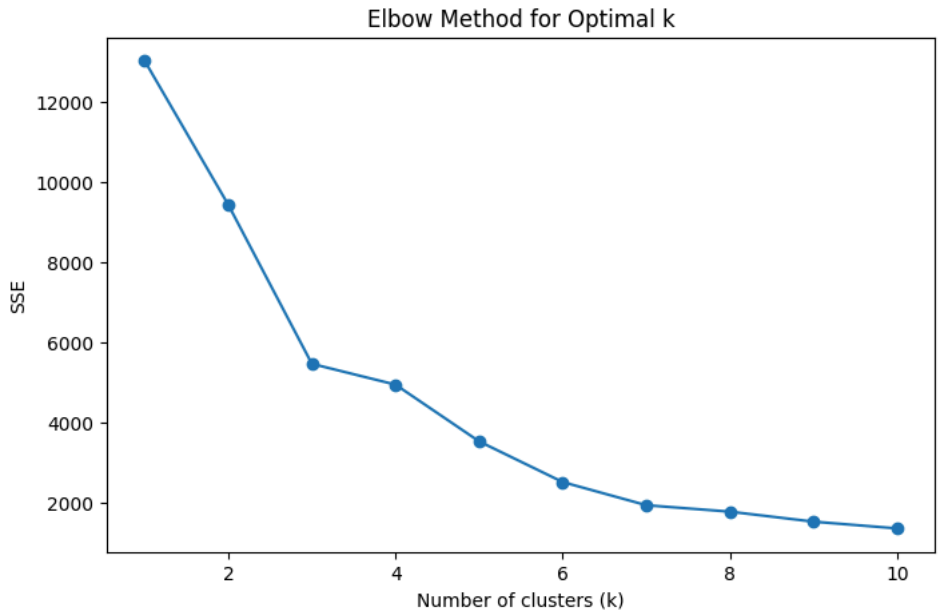
```
import datetime as dt
ref_date = df["InvoiceDate"].max() + dt.timedelta(days=1)
rfm = df.groupby("CustomerID").agg({
    "InvoiceDate": lambda x: (ref_date - x.max()).days,
    "InvoiceNo": "count",
    "TotalAmount": "sum"}).reset_index()
rfm.rename(columns={"InvoiceDate": "Recency",
                    "InvoiceNo": "Frequency",
                    "TotalAmount": "Monetary"}, inplace=True)
rfm.head()
```

	CustomerID	Recency	Frequency	Monetary
0	12346.0	326	1	77183.60
1	12347.0	2	182	4310.00
2	12348.0	75	31	1797.24
3	12349.0	19	73	1757.55
4	12350.0	310	17	334.40

Next steps: Generate code with rfm | View recommended plots | New interactive sheet

```
scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm[["Recency", "Frequency", "Monetary"]])
sse = {}
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(rfm_scaled)
```

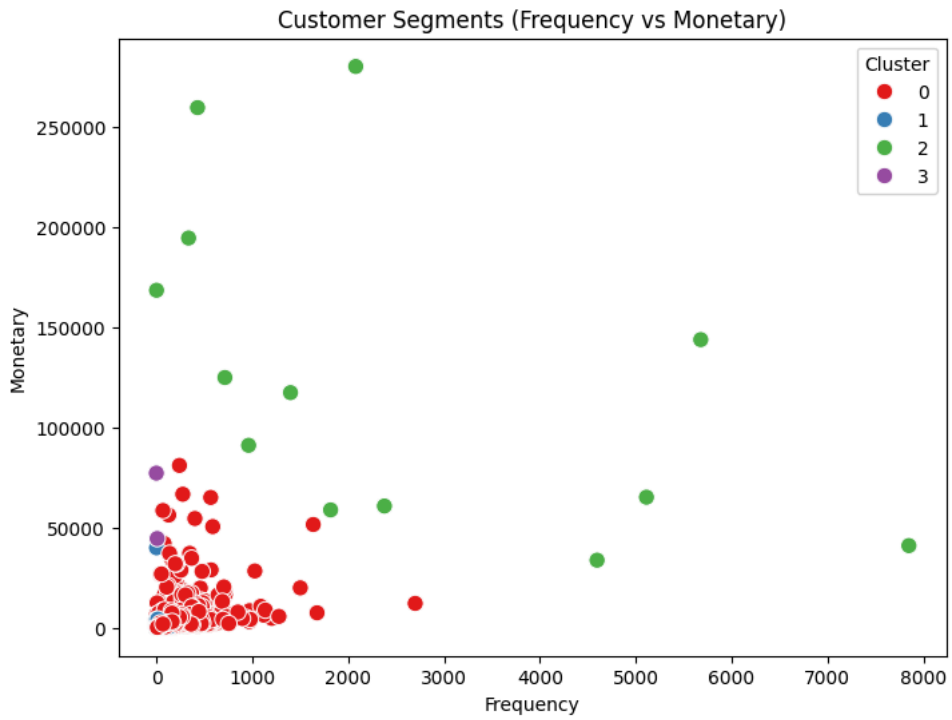
```
    sse[k] = kmeans.inertia_
plt.figure(figsize=(8,5))
plt.plot(list(sse.keys()), list(sse.values()), marker="o")
plt.xlabel("Number of clusters (k)")
plt.ylabel("SSE")
plt.title("Elbow Method for Optimal k")
plt.show()
```




```
kmeans = KMeans(n_clusters=4, random_state=42)
rfm["Cluster"] = kmeans.fit_predict(rfm_scaled)
plt.figure(figsize=(8,6))
sns.scatterplot(data=rfm, x="Recency", y="Monetary", hue="Cluster", palette="Set2", s=80)
plt.title("Customer Segments (Recency vs Monetary)")
plt.show()
```


```
plt.figure(figsize=(8,6))
sns.scatterplot(data=rfm, x="Frequency", y="Monetary", hue="Cluster", palette="Set1", s=80)
plt.title("Customer Segments (Frequency vs Monetary)")
plt.show()
cluster_summary = rfm.groupby("Cluster").agg({
    "Recency": "mean",
    "Frequency": "mean",
    "Monetary": "mean",
    "CustomerID": "count"
```

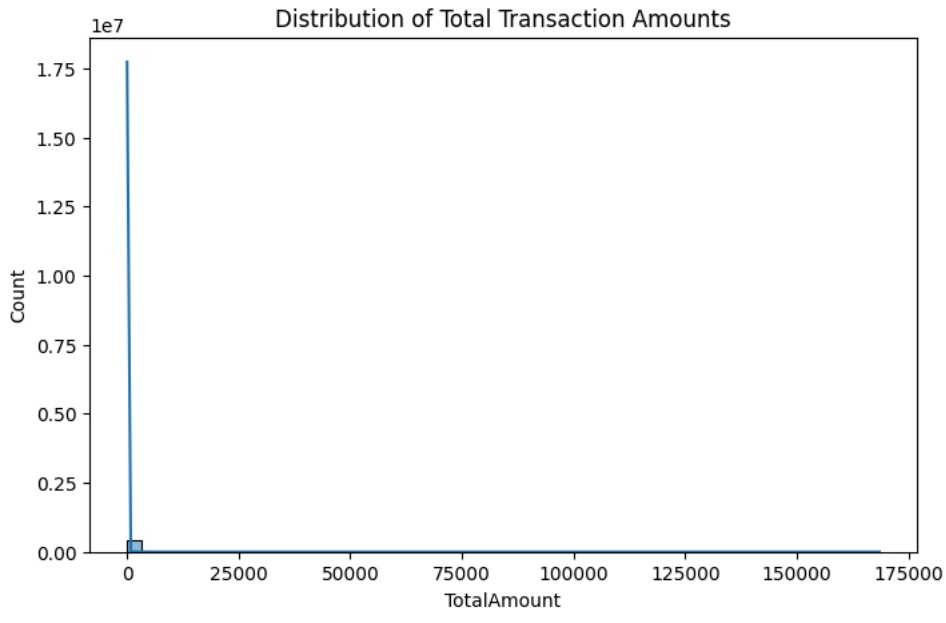
```
}).rename(columns={"CustomerID": "NumCustomers"}).reset_index()
cluster_summary
```



	Cluster	Recency	Frequency	Monetary	NumCustomers
0	0	20.886374	135.476212	2648.875746	2165
1	1	97.846732	37.891811	777.692744	1331
2	2	4.692308	2566.000000	126118.310000	13
3	3	272.196386	25.091566	604.752820	830

Next steps: Generate code with `cluster_summary` View recommended plots New interactive sheet

```
df["Country"].value_counts().head(10)
plt.figure(figsize=(8,5))
sns.histplot(df["TotalAmount"], bins=50, kde=True)
plt.title("Distribution of Total Transaction Amounts")
plt.show()
```




```
plt.figure(figsize=(6,4))
corr = rfm[["Recency", "Frequency", "Monetary"]].corr()
sns.heatmap(corr,
            annot=True,
            cmap="Blues",
```

```
            linewidths=0.5,
            fmt=".2f")
plt.title("Correlation Heatmap of RFM Features")
plt.show()
```