```
In [1]: !pip install pandas numpy matplotlib seaborn scikit-learn
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packag
es (2.2.3)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-package
s (2.1.3)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-pa
ckages (3.10.0)
Requirement already satisfied: seaborn in c:\programdata\anaconda3\lib\site-packa
ges (0.13.2)
Requirement already satisfied: scikit-learn in c:\programdata\anaconda3\lib\site-
packages (1.6.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3
\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-
packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\sit
e-packages (from pandas) (2025.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\programdata\anaconda3\lib\s
ite-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-
packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anaconda3\lib
\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\programdata\anaconda3\lib
\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\si
te-packages (from matplotlib) (24.2)
Requirement already satisfied: pillow>=8 in c:\programdata\anaconda3\lib\site-pac
kages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\programdata\anaconda3\lib\s
ite-packages (from matplotlib) (3.2.0)
Requirement already satisfied: scipy>=1.6.0 in c:\programdata\anaconda3\lib\site-
packages (from scikit-learn) (1.15.3)
Requirement already satisfied: joblib>=1.2.0 in c:\programdata\anaconda3\lib\site
-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\programdata\anaconda3\l
ib\site-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-pack
ages (from python-dateutil>=2.8.2->pandas) (1.17.0)
```

```
In [2]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
```

```
In [3]: df = pd.read_excel("online_retail_II.xlsx")
```

```
In [4]: print("First 5 rows of the dataset:")
        df.head()
```

```
First 5 rows of the dataset:
```

Out[4]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	489434	85048	15CM CHRISTMAS GLASS BALL 20 LIGHTS	12	2009-12-01 07:45:00	6.95	13085.0	United Kingdom
1	489434	79323P	PINK CHERRY LIGHTS	12	2009-12-01 07:45:00	6.75	13085.0	United Kingdom
2	489434	79323W	WHITE CHERRY LIGHTS	12	2009-12-01 07:45:00	6.75	13085.0	United Kingdom
3	489434	22041	RECORD FRAME 7" SINGLE SIZE	48	2009-12-01 07:45:00	2.10	13085.0	United Kingdom
4	489434	21232	STRAWBERRY CERAMIC TRINKET BOX	24	2009-12-01 07:45:00	1.25	13085.0	United Kingdom

In [5]:
```python
print("Missing values per column:")
print(df.isnull().sum())
```

```
Missing values per column:
Invoice             0
StockCode           0
Description      2928
Quantity            0
InvoiceDate         0
Price               0
Customer ID    107927
Country             0
dtype: int64
```

In [6]:
```python
df = df[df['CustomerID'].notnull()]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805,
in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

KeyError: 'CustomerID'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[6], line 1
----> 1 df = df[df['CustomerID'].notnull()]

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in Dat
aFrame.__getitem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812,
in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'CustomerID'
```

```
In [7]: print(df.columns)
```

```
Index(['Invoice', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'Price', 'Customer ID', 'Country'],
      dtype='object')
```

```
In [8]: df = df[df['Customer ID'].notnull()]
```

```
In [10]: df = df[(df['Quantity'] > 0) & (df['Price'] > 0)]
```

In [11]:
```python
df = df[(df['Quantity'] > 0) & (df['Price'] > 0)]
```

In [12]:
```python
df = df.drop_duplicates()
```

In [13]:
```python
print("\nShape of cleaned data:", df.shape)
df.head()
```

```
Shape of cleaned data: (400916, 8)
```

Out[13]:

	Invoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
0	489434	85048	15CM CHRISTMAS GLASS BALL 20 LIGHTS	12	2009-12-01 07:45:00	6.95	13085.0	United Kingdom
1	489434	79323P	PINK CHERRY LIGHTS	12	2009-12-01 07:45:00	6.75	13085.0	United Kingdom
2	489434	79323W	WHITE CHERRY LIGHTS	12	2009-12-01 07:45:00	6.75	13085.0	United Kingdom
3	489434	22041	RECORD FRAME 7" SINGLE SIZE	48	2009-12-01 07:45:00	2.10	13085.0	United Kingdom
4	489434	21232	STRAWBERRY CERAMIC TRINKET BOX	24	2009-12-01 07:45:00	1.25	13085.0	United Kingdom

In [14]:
```python
df.columns = df.columns.str.strip()
```

In [16]:
```python
df.rename(columns={'Customer ID': 'CustomerID', 'Price': 'UnitPrice'}, inplace=T
```

In [17]:
```python
print(df.columns)
```

```
Index(['Invoice', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

In [18]:
```python
import datetime
```

In [19]:
```python
latest_date = df['InvoiceDate'].max() + pd.Timedelta(days=1)
```

In [20]:
```python
rfm = df.groupby('CustomerID').agg({
    'InvoiceDate': lambda x: (latest_date - x.max()).days,  # Recency
    'Invoice': 'nunique',                                     # Frequency
    'UnitPrice': lambda x: round((x * df.loc[x.index, 'Quantity']).sum(), 2)  #
})

rfm.columns = ['Recency', 'Frequency', 'Monetary']
rfm.reset_index(inplace=True)
```

In [22]:
```
print("RFM table preview:")
rfm.head()
```

```
RFM table preview:
```

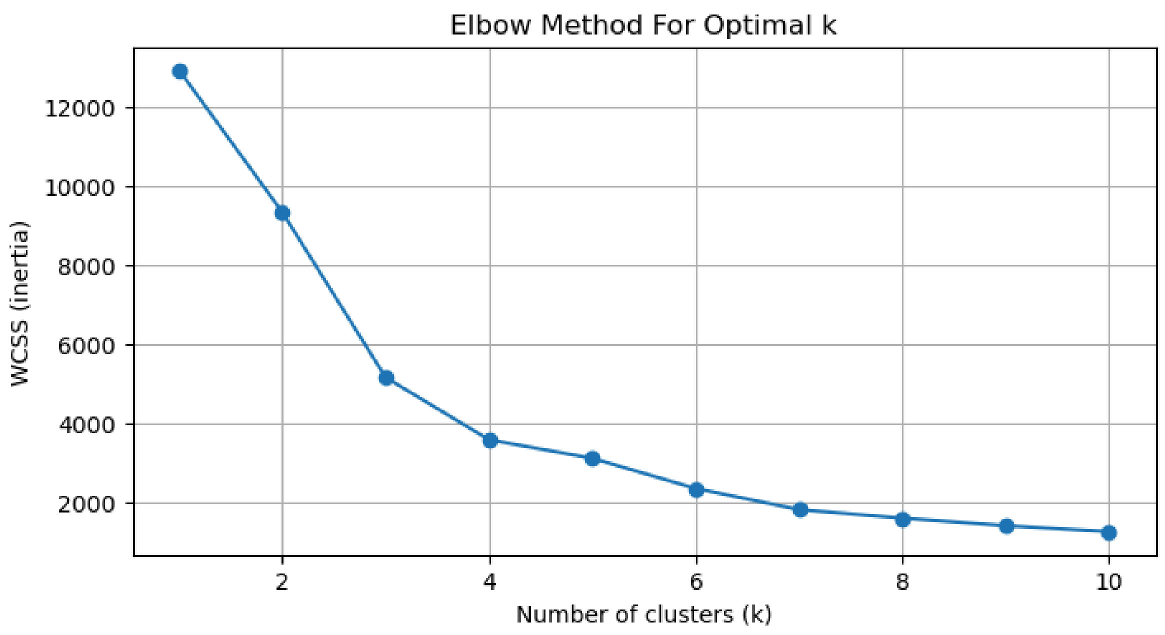
Out[22]:

	CustomerID	Recency	Frequency	Monetary
0	12346.0	165	11	372.86
1	12347.0	3	2	1323.32
2	12348.0	74	1	222.16
3	12349.0	43	3	2671.14
4	12351.0	11	1	300.93

In [23]:
```
from sklearn.preprocessing import StandardScaler

rfm_data = rfm[['Recency', 'Frequency', 'Monetary']]

scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm_data)

print("First 5 scaled values:")
print(rfm_scaled[:5])
```

```
First 5 scaled values:
[[ 0.76229851  0.80108727 -0.18713934]
 [-0.91040156 -0.3006029  -0.08047459]
 [-0.17730462 -0.42301292 -0.20405155]
 [-0.4973892  -0.17819288  0.07078363]
 [-0.82779909 -0.42301292 -0.19521163]]
```

In [24]:
```
from sklearn.cluster import KMeans
wcss = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, init='k-means++', random_state=42)
    kmeans.fit(rfm_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(8, 4))
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method For Optimal k')
plt.xlabel('Number of clusters (k)')
plt.ylabel('WCSS (inertia)')
plt.grid(True)
plt.show()
```

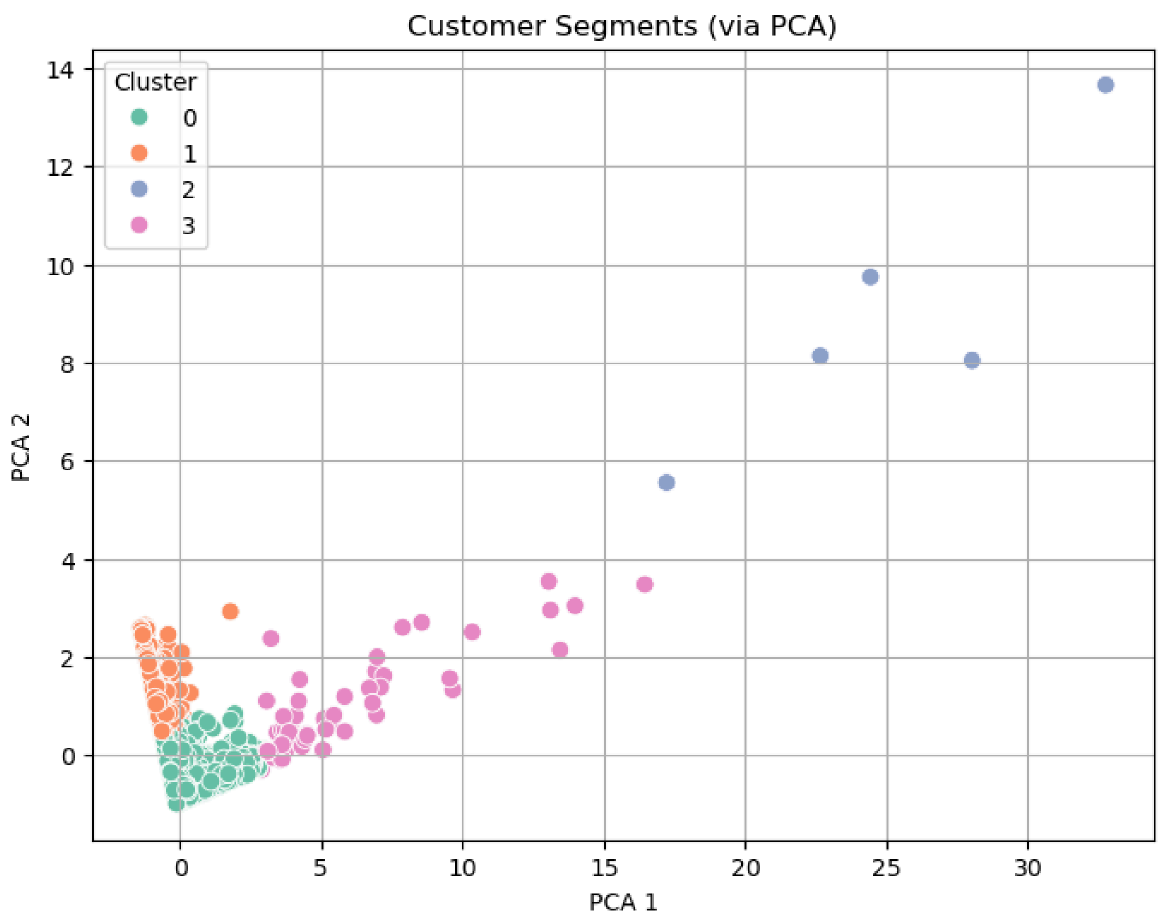


```
In [25]: kmeans = KMeans(n_clusters=4, init='k-means++', random_state=42)
         kmeans.fit(rfm_scaled)

         rfm['Cluster'] = kmeans.labels_

         rfm.head()
```

Out[25]:

	CustomerID	Recency	Frequency	Monetary	Cluster
0	12346.0	165	11	372.86	1
1	12347.0	3	2	1323.32	0
2	12348.0	74	1	222.16	0
3	12349.0	43	3	2671.14	0
4	12351.0	11	1	300.93	0

```
In [26]: from sklearn.decomposition import PCA

         pca = PCA(n_components=2)
         rfm_pca = pca.fit_transform(rfm_scaled)


         rfm['PCA1'] = rfm_pca[:, 0]
         rfm['PCA2'] = rfm_pca[:, 1]


         plt.figure(figsize=(8,6))
         sns.scatterplot(data=rfm, x='PCA1', y='PCA2', hue='Cluster', palette='Set2', s=6
         plt.title('Customer Segments (via PCA)')
         plt.xlabel('PCA 1')
         plt.ylabel('PCA 2')
         plt.grid(True)
         plt.show()
```



In [27]:
```
cluster_summary = rfm.groupby('Cluster').agg({
    'Recency': 'mean',
    'Frequency': 'mean',
    'Monetary': 'mean',
    'CustomerID': 'count'
}).rename(columns={'CustomerID': 'Num_Customers'})

cluster_summary = cluster_summary.round(2)
cluster_summary
```

Out[27]:

Cluster	Recency	Frequency	Monetary	Num_Customers
0	43.03	4.46	1710.65	3204
1	242.98	1.66	593.54	1047
2	5.60	113.60	215535.00	5
3	14.91	47.02	28896.42	56

In [28]:
```
cluster_summary = rfm.groupby('Cluster').agg({
    'Recency': 'mean',
    'Frequency': 'mean',
    'Monetary': 'mean',
    'CustomerID': 'count'
}).rename(columns={'CustomerID': 'Num_Customers'})

cluster_summary = cluster_summary.round(2)
cluster_summary
```

Out[28]:

	Recency	Frequency	Monetary	Num_Customers
Cluster				
0	43.03	4.46	1710.65	3204
1	242.98	1.66	593.54	1047
2	5.60	113.60	215535.00	5
3	14.91	47.02	28896.42	56

In [ ]: