Step 1 - Importing of Libraries

In [4]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Step 2 - Reading of data in Data Frame

In [7]:
```
df=pd.read_excel("F:/DATA SC/Python notebooks/old notebooks/data/storedata.xl
df.head(2)
```

Out[7]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...
0	1	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...
1	2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...

2 rows × 21 columns

In [8]:
```
df.tail(2)
```

Out[8]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	C
9992	9993	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Me
9993	9994	CA-2017-119914	2017-05-04	2017-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westmins

2 rows × 21 columns

Step 3 - Finding missing Values

In [9]:
```
df.isnull().sum()

# Country - 1 missing value
# City - 1 missing value
#Posta Code - 2 missing Value
```

Out[9]:
```
Row ID           0
Order ID         0
Order Date       0
Ship Date        0
Ship Mode        0
Customer ID      0
Customer Name    0
Segment          0
Country          1
City             1
State            0
Postal Code      2
Region           0
Product ID       0
Category         0
Sub-Category     0
Product Name     0
Sales            0
Quantity         0
Discount         0
Profit           0
dtype: int64
```

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   datetime64[ns]
 3   Ship Date      9994 non-null   datetime64[ns]
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9993 non-null   object
 9   City           9993 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9992 non-null   float64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: datetime64[ns](2), float64(4), int64(2), object(13)
memory usage: 1.6+ MB
```

Interpretation of the below output
Rangeindex - total number of records present in database file - 9994

```
#index - 0 to 9993
#County column is havinf 9993 so there is a one missing value
#City column is havinf 9993 so there is a one missing value
#Postal code is havinf 9992 so 2 missing value
```

Number of Columns - 21
memory usage: 1.6+ MB
dtypes: datetime64[ns](2), float64(4), int64(2), object(13)

Step 4 - Data Cleaning - Preprocessing

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Row ID         9994 non-null   int64
 1   Order ID       9994 non-null   object
 2   Order Date     9994 non-null   datetime64[ns]
 3   Ship Date      9994 non-null   datetime64[ns]
 4   Ship Mode      9994 non-null   object
 5   Customer ID    9994 non-null   object
 6   Customer Name  9994 non-null   object
 7   Segment        9994 non-null   object
 8   Country        9993 non-null   object
 9   City           9993 non-null   object
 10  State          9994 non-null   object
 11  Postal Code    9992 non-null   float64
 12  Region         9994 non-null   object
 13  Product ID     9994 non-null   object
 14  Category       9994 non-null   object
 15  Sub-Category   9994 non-null   object
 16  Product Name   9994 non-null   object
 17  Sales          9994 non-null   float64
 18  Quantity       9994 non-null   int64
 19  Discount       9994 non-null   float64
 20  Profit         9994 non-null   float64
dtypes: datetime64[ns](2), float64(4), int64(2), object(13)
memory usage: 1.6+ MB
```

Step 4.1 - missing value treatment

In [12]:
```
# Displays all rows having missing value in any column
rows_with_missing = df[df.isna().any(axis=1)]
rows_with_missing
```

Out[12]:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	..
9	10	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	..
10	11	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	NaN	Los Angeles	..
14	15	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office	United States	NaN	..
16	17	CA-2014-105893	2014-11-11	2014-11-18	Standard Class	PK-19075	Pete Kriz	Consumer	United States	Madison	..

4 rows × 21 columns

In [14]:
```
# Country - "united state"
# city = " Fort Worth"
# Postal code - United States   Los Angeles--> California -- 90032
# Postal code - United States   Madison---> Wisconsin -- 53711

df['City'].fillna('Fort Worth', inplace=True)
df['Country'].fillna('United States', inplace=True)
df.isnull().sum()
```

Out[14]:
```
Row ID           0
Order ID         0
Order Date       0
Ship Date        0
Ship Mode        0
Customer ID      0
Customer Name    0
Segment          0
Country          0
City             0
State            0
Postal Code      2
Region           0
Product ID       0
Category         0
Sub-Category     0
Product Name     0
Sales            0
Quantity         0
Discount         0
Profit           0
dtype: int64
```

In [24]:
```
rows_with_missing = df[df.isna().any(axis=1)]
rows_with_missing
```

Out[24]:

Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	R

0 rows × 21 columns

In [ ]:
```
df.loc[9, :]=df.loc[9, :].fillna('90032')
```

In [23]:
```
df.loc[16, :]=df.loc[16, :].fillna('53711')
```

Exploratory Data Analysis - EDA provides insightful information that helps with hypothesis creation and decision-making by improving knowledge of data distribution, variable correlations, and anomalies. When all is said and done, the efficacy of data-driven projects is enhanced by EDA's capacity to identify trends and anomalies.

Univariate Analysis

Column-wise analysis - Region Column

In [25]:
```
df
```

Out[25]:

rder ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Post Co
CA-016-2156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420
CA-016-2156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420
CA-016-8688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	90036
US-015-8966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311
US-015-8966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...	33311
...	...	...	...	...	...	...	...	...	...	
CA-014-0422	2014-01-21	2014-01-23	Second Class	TB-21400	Tom Boeckenhauer	Consumer	United States	Miami	...	33180
CA-017-1258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627
CA-017-1258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627
CA-017-1258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...	92627
CA-017-9914	2017-05-04	2017-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westminster	...	92683

columns

In [29]:
```
# in How many region of unites state out superstore deals
df['Region'].unique()
```

Out[29]:
```
array(['South', 'West', 'Central', 'East'], dtype=object)
```

In [30]:
```
# In which region of unites states our superstore deliver
df['Region'].value_counts()
```

Out[30]:
```
West       3203
East       2848
Central    2323
South      1620
Name: Region, dtype: int64
```

In [28]:
```
df['Region']=df['Region'].replace('Central ','Central')
```

Column-wise analysis - Category Column

In [31]:
```
df['Category'].value_counts()
```

Out[31]:
```
Office Supplies    6026
Furniture          2121
Technology         1847
Name: Category, dtype: int64
```

In [32]:
```
df.head(2)
```

Out[32]:

Ship Mode	Customer ID	Customer Name	Segment	Country	City	...	Postal Code	Region	Product ID	Ca
Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420.0	South	FUR-BO-10001798	Fu
Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...	42420.0	South	FUR-CH-10000454	Fu

In [33]:
```
df['Category'].unique()
```

Out[33]:
```
array(['Furniture', 'Office Supplies', 'Technology'], dtype=object)
```

In [34]:
```
df['Sub-Category'].unique()
```

Out[34]:
```
array(['Bookcases', 'Chairs', 'Labels', 'Tables', 'Storage',
       'Furnishings', 'Art', 'Phones', 'Binders', 'Appliances', 'Paper',
       'Accessories', 'Envelopes', 'Fasteners', 'Supplies', 'Machines',
       'Copiers'], dtype=object)
```

In [35]:
```
df['Ship Mode'].unique()
```

Out[35]:
```
array(['Second Class', 'Standard Class', 'First Class', 'Same Day'],
      dtype=object)
```

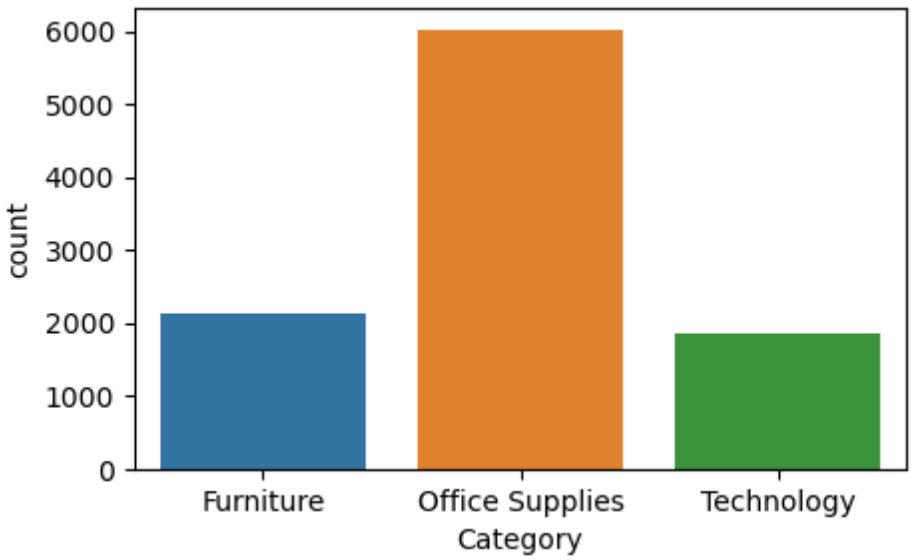
In [36]:
```
df['Ship Mode'].value_counts()
```

Out[36]:
```
Standard Class    5968
Second Class      1945
First Class       1538
Same Day           543
Name: Ship Mode, dtype: int64
```

In [37]:
```
df['Category'].value_counts()
```

Out[37]:
```
Office Supplies    6026
Furniture          2121
Technology         1847
Name: Category, dtype: int64
```

In [39]:
```
plt.figure(figsize=(5,3))
sns.countplot(x="Category", data=df)
plt.title("Count of Category")
plt.show()
```



ship mode unvariate analysis

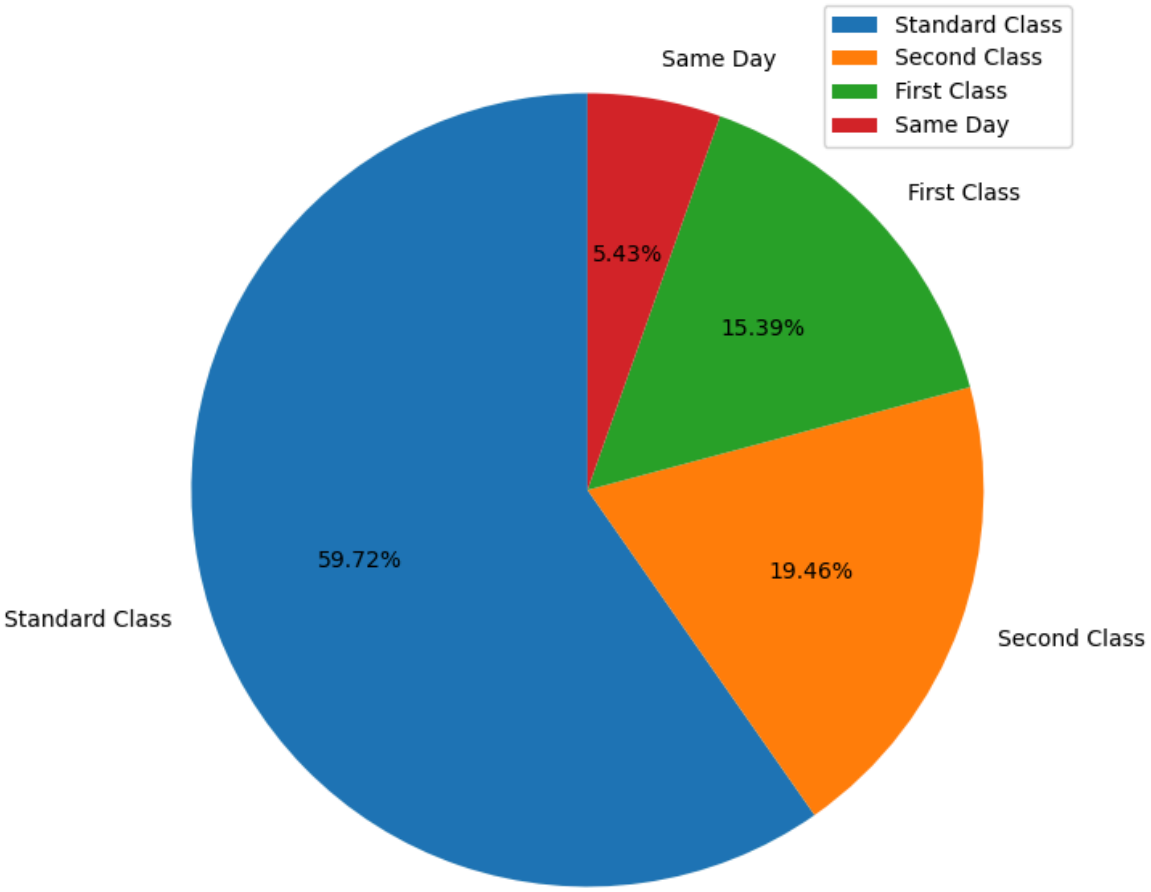
In [40]:
```
df['Ship Mode'].value_counts()
```

Out[40]:
```
Standard Class    5968
Second Class      1945
First Class       1538
Same Day           543
Name: Ship Mode, dtype: int64
```

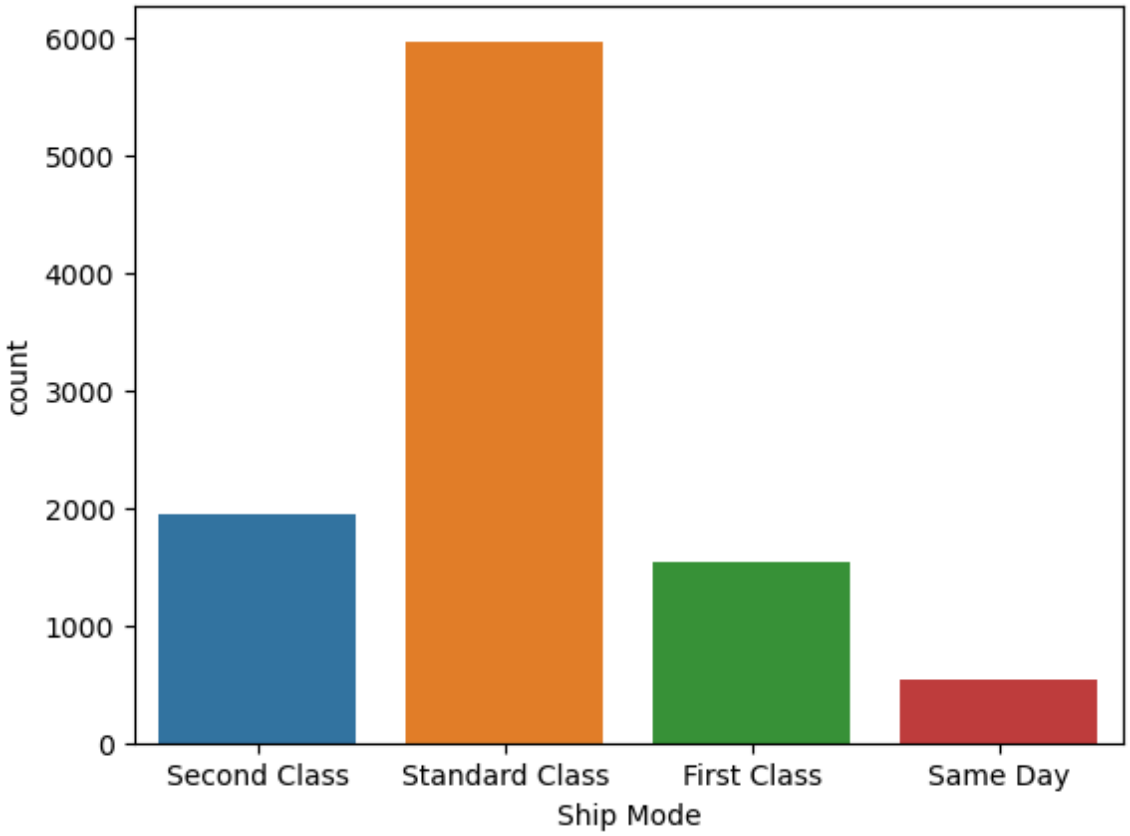
In [42]:
```
x=df['Ship Mode'].value_counts().index
y=df['Ship Mode'].value_counts().values
print("indexes=",x)
print("Values=",y)
```

```
indexes= Index(['Standard Class', 'Second Class', 'First Class', 'Same Da
y'], dtype='object')
Values= [5968 1945 1538  543]
```

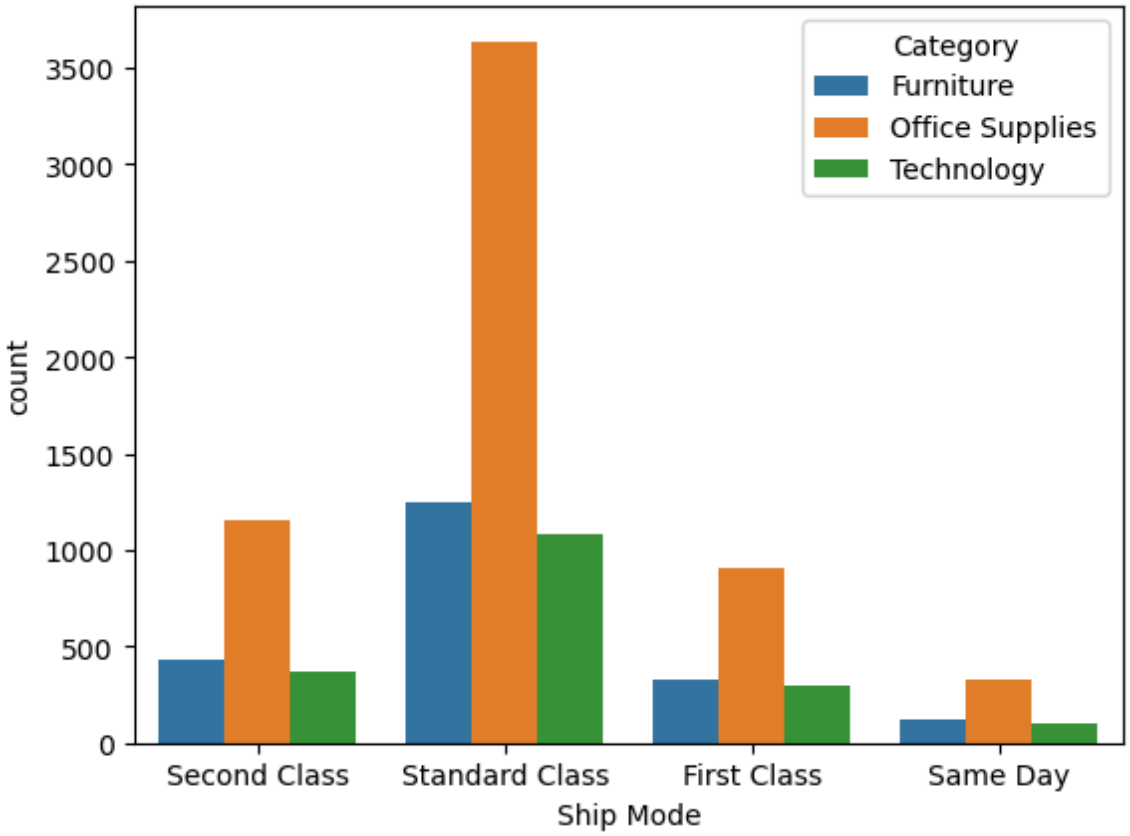
In [57]:
```
plt.figure(figsize=(9,8))
plt.pie(y, labels=x , startangle=90 , autopct="%0.2f%%")
plt.legend()
plt.show()
```



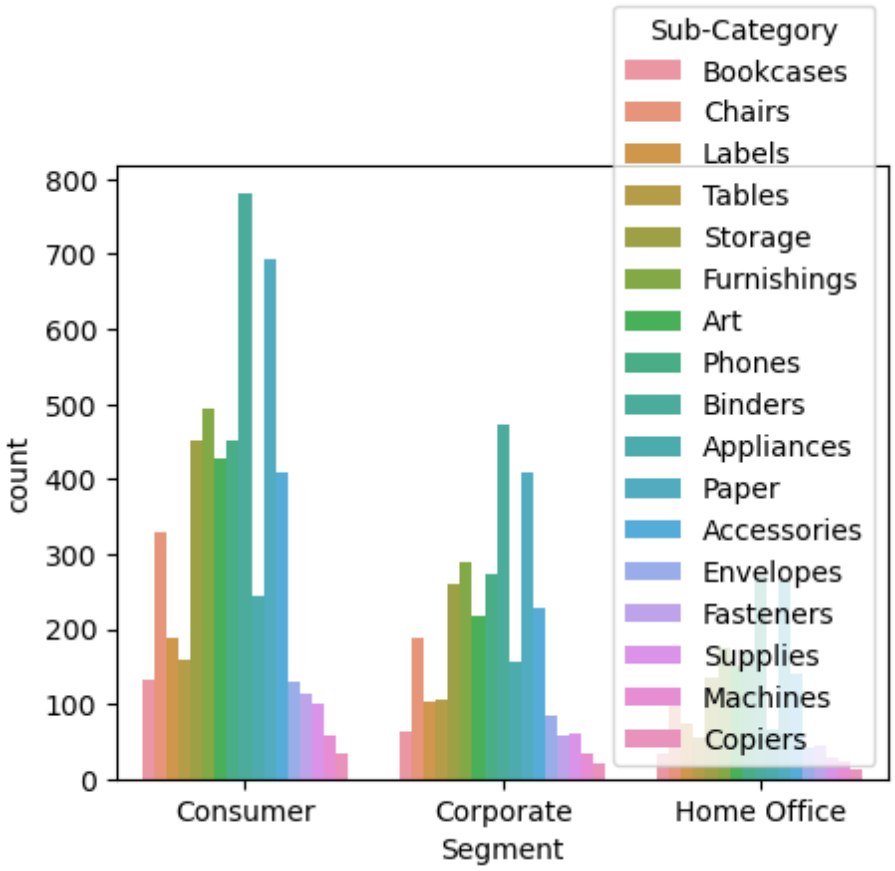
In [58]:
```
sns.countplot(x="Ship Mode", data=df)
plt.show()
```



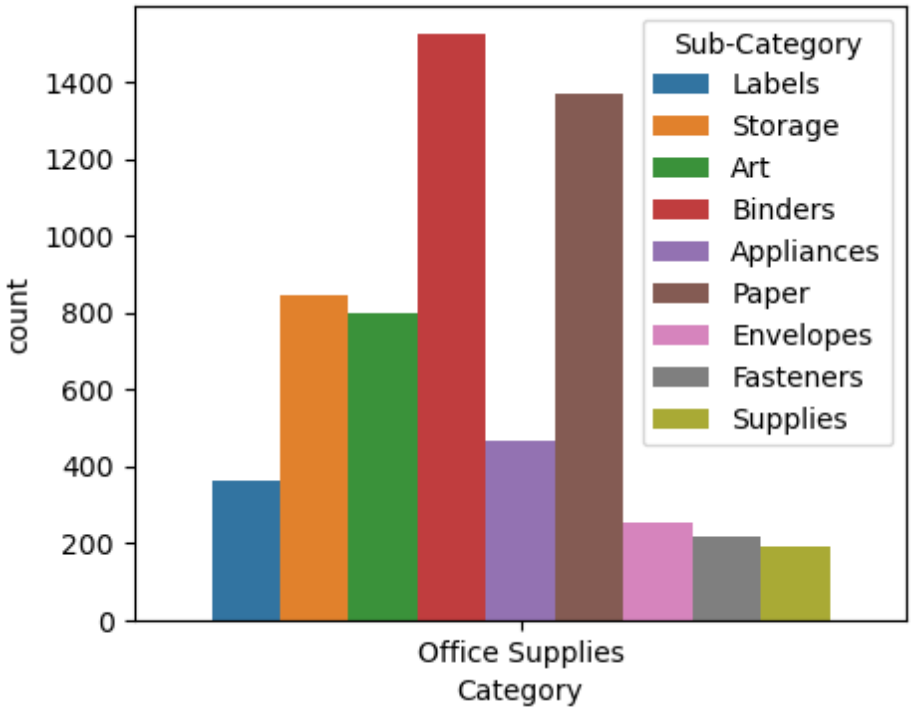
In [59]:
```
sns.countplot(x="Ship Mode", data=df, hue="Category")
plt.show()
```



In [62]:
```
plt.figure(figsize=(5,4))
sns.countplot(x="Segment", data=df , hue="Sub-Category")
plt.show()
```



In [63]:
```
plt.figure(figsize=(5,4))
sns.countplot(x="Category", data=df[df["Category"]=="Office Supplies"], hue="
plt.show()
```



In [ ]:
```

```