```python
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving Stores.csv to Stores (1).csv
```

```python
import pandas as pd
```

```python
df = pd.read_csv("Stores.csv")
```

```python
print(df.head())
print(df.shape)
print(df.columns)
```

```
   Store ID   Store_Area  Items_Available  Daily_Customer_Count
Store_Sales
0          1        1659             1961                   530
66490
1          2        1461             1752                   210
39820
2          3        1340             1609                   720
54010
3          4        1451             1748                   620
53730
4          5        1770             2111                   450
46620
(896, 5)
Index(['Store ID ', 'Store_Area', 'Items_Available',
'Daily_Customer_Count',
       'Store_Sales'],
      dtype='object')
```

```python
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 896 entries, 0 to 895
Data columns (total 5 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Store ID              896 non-null    int64
 1   Store_Area            896 non-null    int64
 2   Items_Available       896 non-null    int64
 3   Daily_Customer_Count  896 non-null    int64
 4   Store_Sales           896 non-null    int64
dtypes: int64(5)
memory usage: 35.1 KB
None
```

```python
print(df.describe(include="all").T)
```

```
                      count          mean           std      min
25%  \
Store ID              896.0    448.500000    258.797218      1.0
224.75
Store_Area            896.0   1485.409598    250.237011    775.0
1316.75
Items_Available       896.0   1782.035714    299.872053    932.0
1575.50
Daily_Customer_Count  896.0    786.350446    265.389281     10.0
600.00
Store_Sales           896.0  59351.305804  17190.741895  14920.0
46530.00

                          50%       75%       max
Store ID                448.5    672.25     896.0
Store_Area             1477.0   1653.50    2229.0
Items_Available        1773.5   1982.75    2667.0
Daily_Customer_Count    780.0    970.00    1560.0
Store_Sales           58605.0  71872.50  116320.0
```

```python
print(df.isnull().sum())
```

```
Store ID                0
Store_Area              0
Items_Available         0
Daily_Customer_Count    0
Store_Sales             0
dtype: int64
```

```python
for col in df.columns:
    print(f"{col} -> {df[col].nunique()} unique values")
```

```
Store ID  -> 896 unique values
Store_Area -> 583 unique values
Items_Available -> 616 unique values
Daily_Customer_Count -> 130 unique values
Store_Sales -> 816 unique values
```

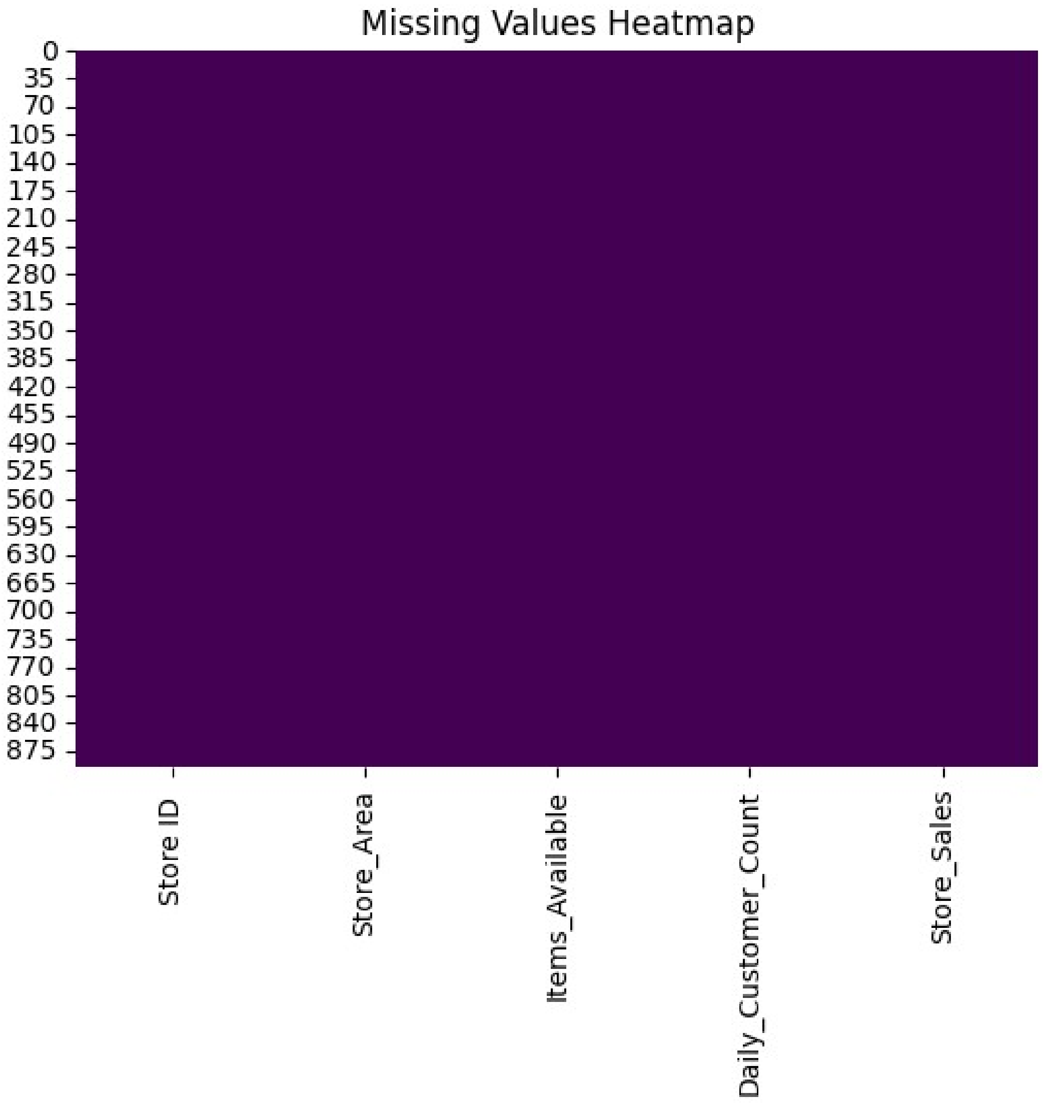
```python
print(df.skew())
```

```
Store ID                0.000000
Store_Area              0.030367
Items_Available         0.034439
Daily_Customer_Count    0.074633
Store_Sales             0.148794
dtype: float64
```

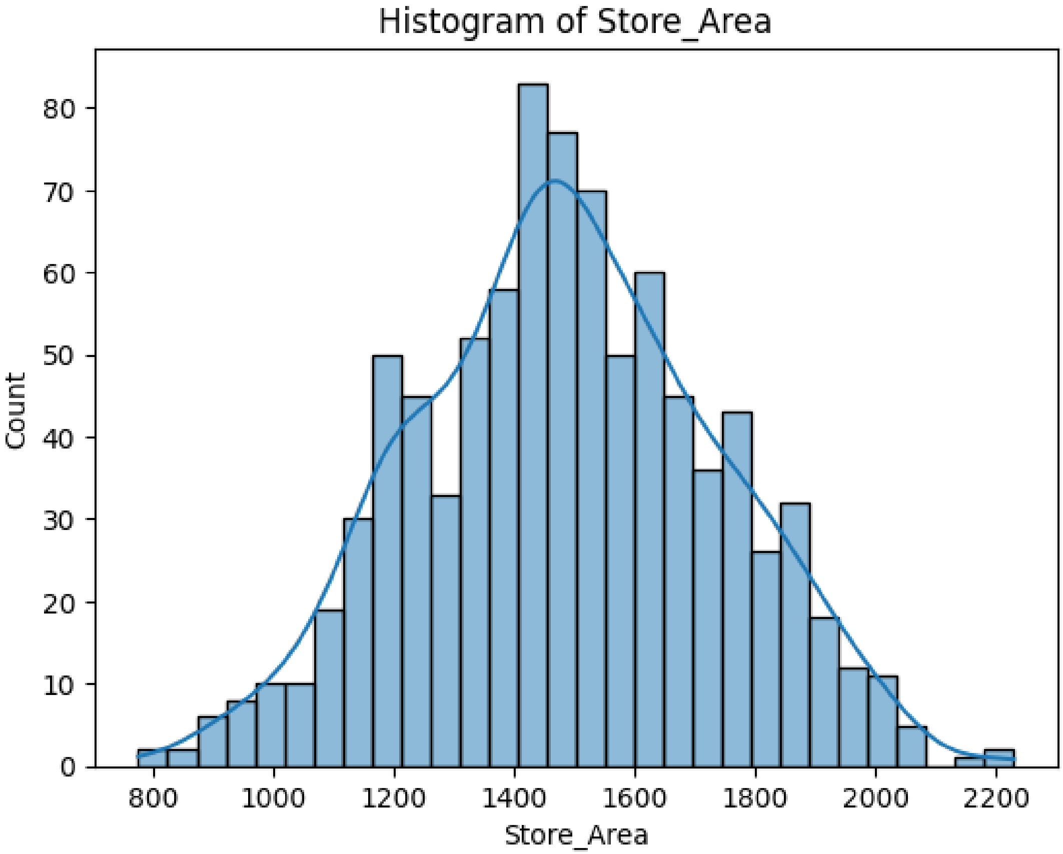
```python
for col in df.select_dtypes(include=['int64','float64']).columns:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5*IQR
    upper = Q3 + 1.5*IQR
    print(f"{col}: {((df[col] < lower) | (df[col] > upper)).sum()} outliers")
```

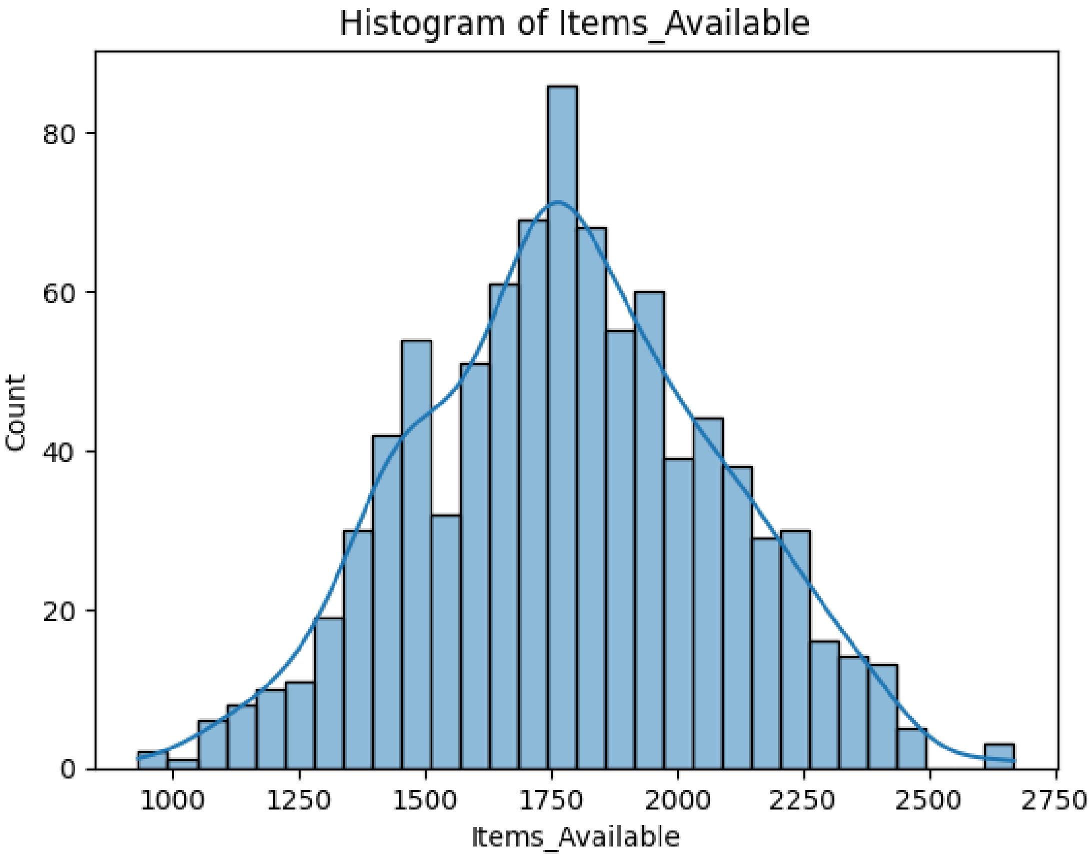
```
Store ID : 0 outliers
Store_Area: 5 outliers
Items_Available: 5 outliers
Daily_Customer_Count: 3 outliers
Store_Sales: 1 outliers
```

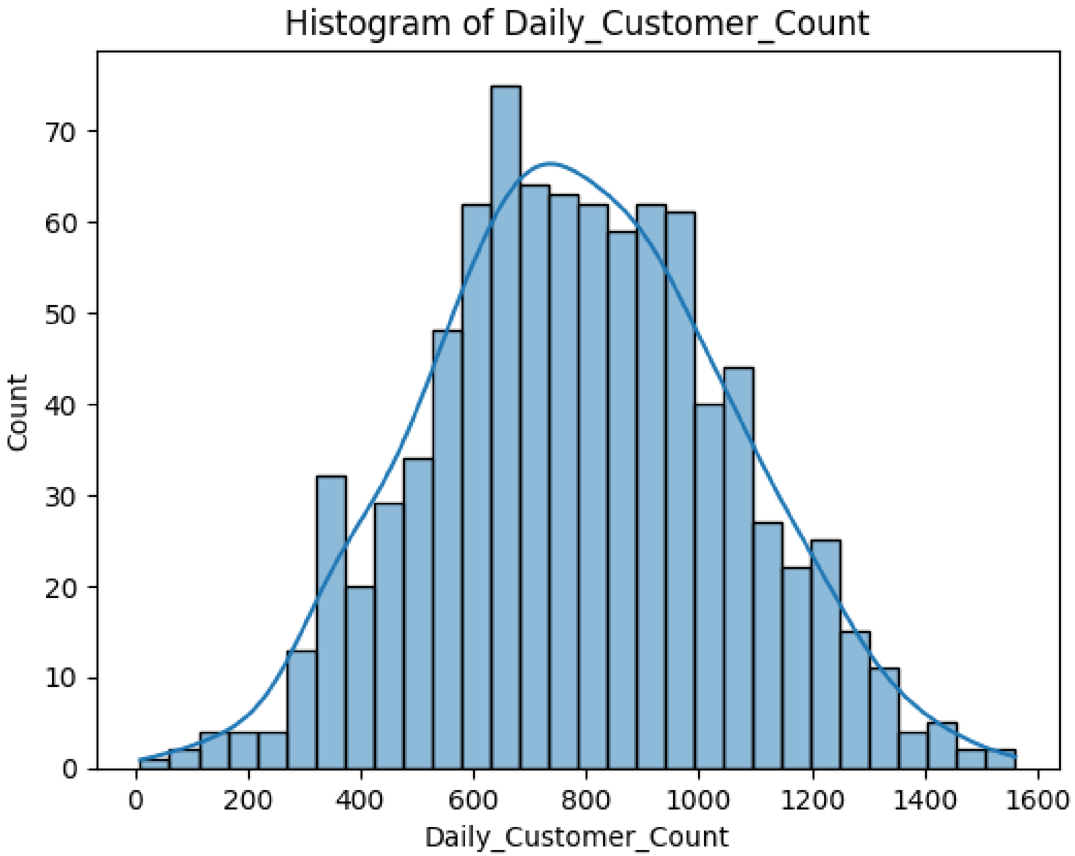
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("Stores.csv")

sns.heatmap(df.isnull(), cbar=False, cmap="viridis")
plt.title("Missing Values Heatmap")
plt.show()
```

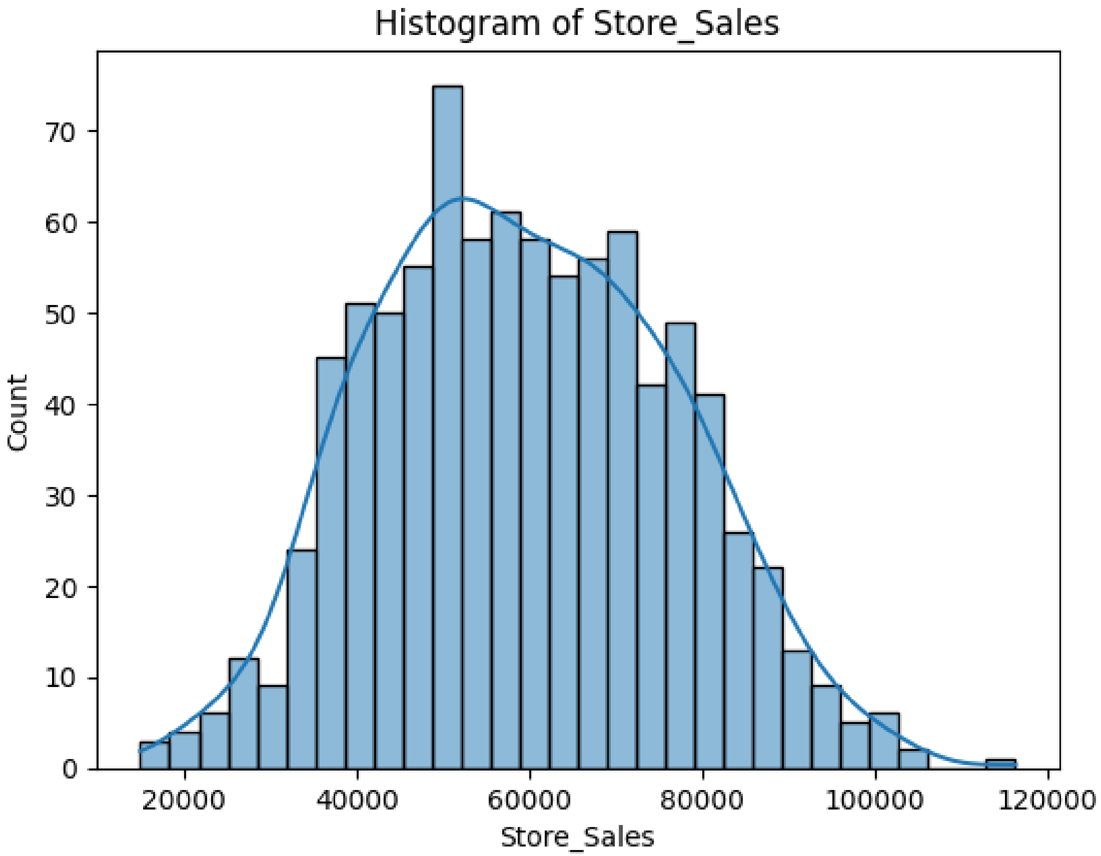


```
for col in df.columns[1:]:
    sns.histplot(df[col], kde=True, bins=30)
    plt.title(f"Histogram of {col}")
    plt.show()
```

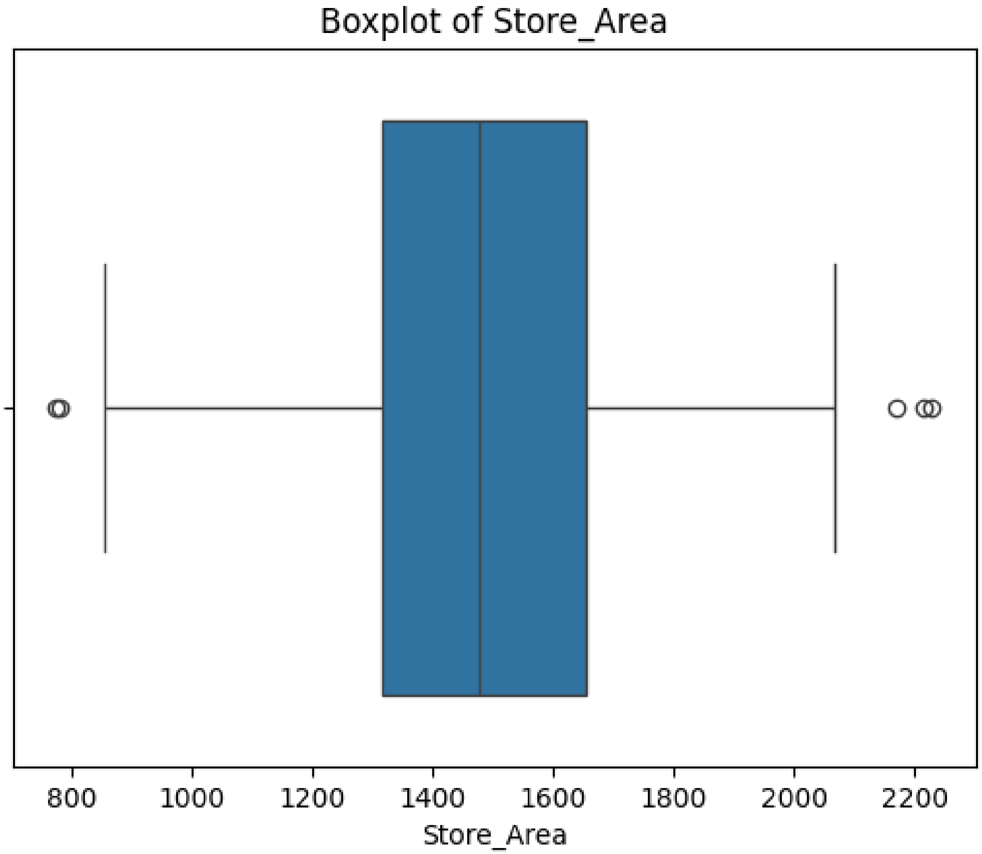

Histogram of Store_Area
Count
0
10
20
30
40
50
60
70
80
800
1000
1200
1400
1600
1800
2000
2200
Store_Area

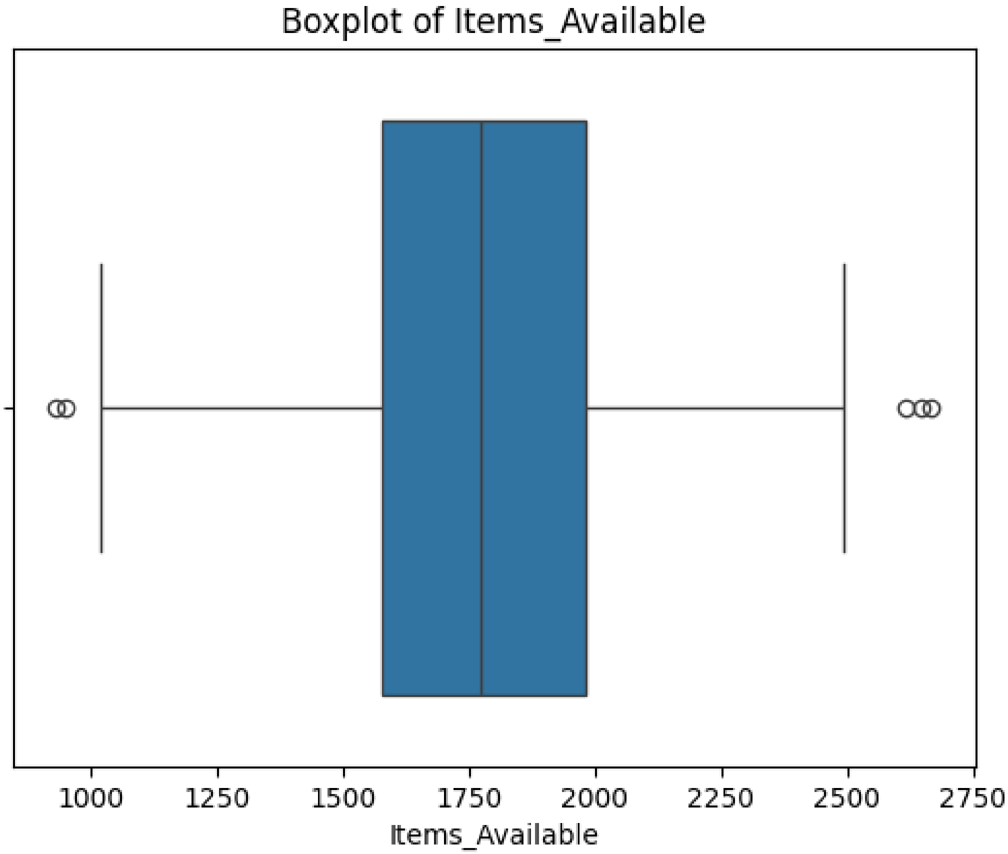

Histogram of Items_Available
Count
0
20
40
60
80
1000
1250
1500
1750
2000
2250
2500
2750
Items_Available

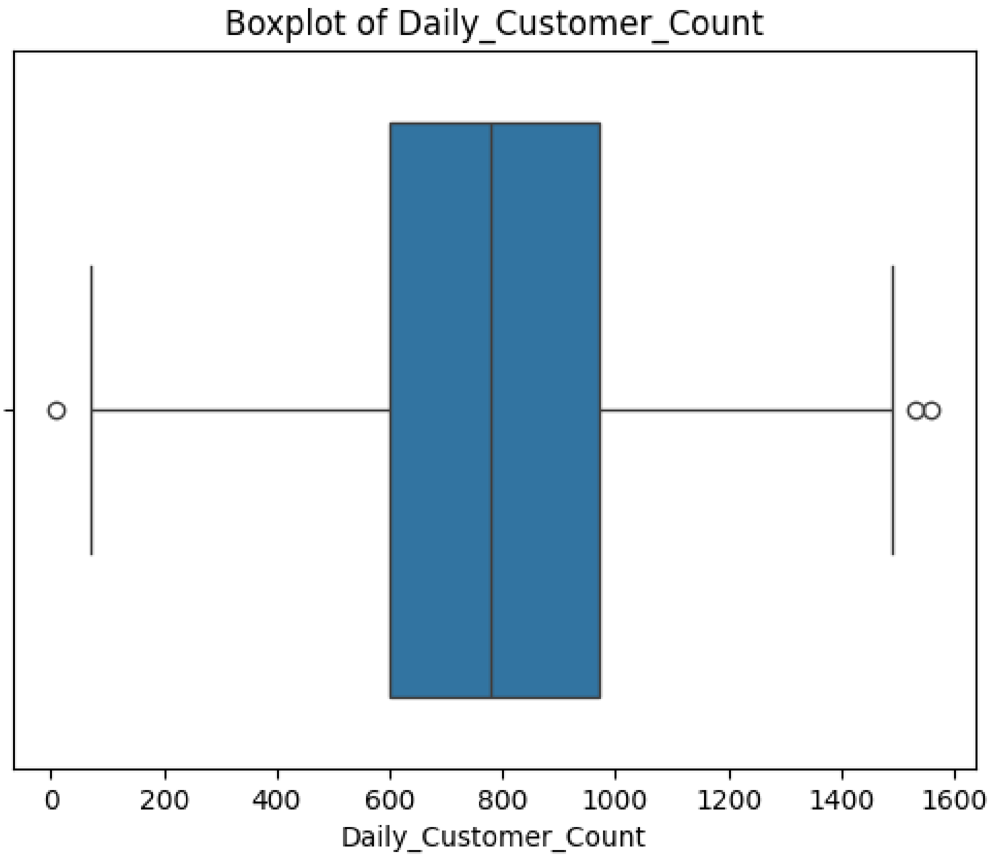

Histogram of Daily_Customer_Count
Count
0
10
20
30
40
50
60
70
0
200
400
600
800
1000
1200
1400
1600
Daily_Customer_Count

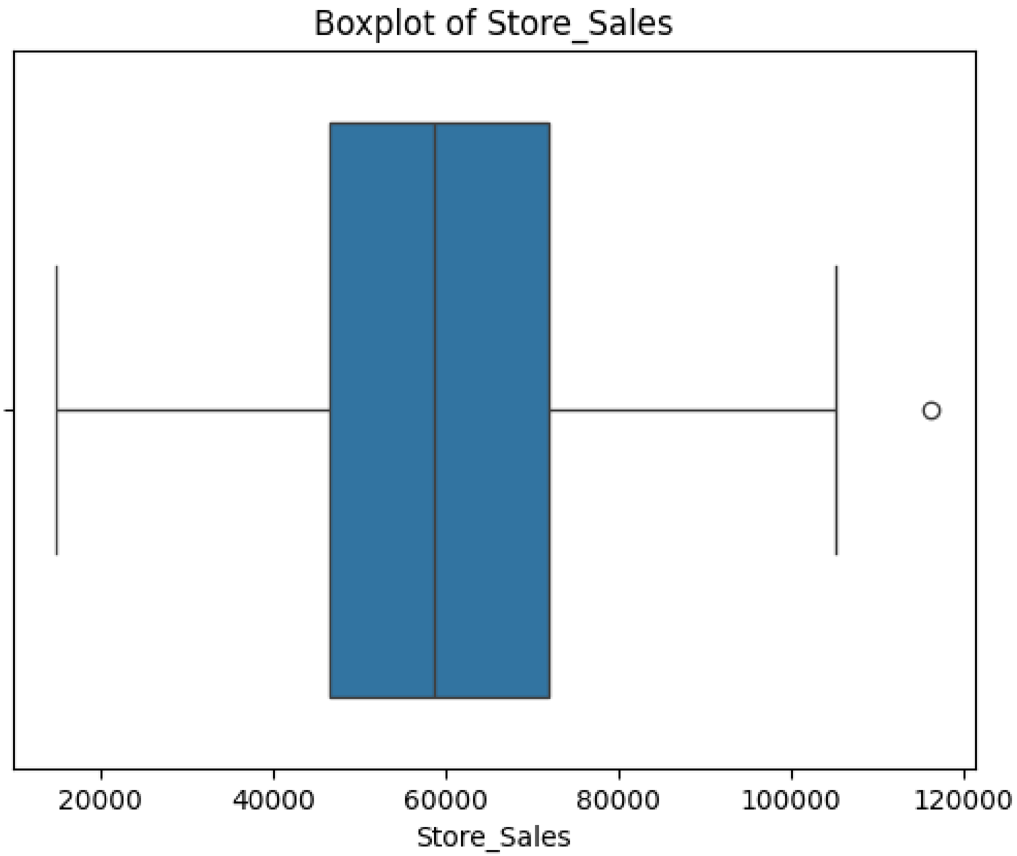


```
for col in df.columns[1:]:
    sns.boxplot(x=df[col])
    plt.title(f"Boxplot of {col}")
    plt.show()
```


Boxplot of Store_Area
800
1000
1200
1400
1600
1800
2000
2200
Store_Area

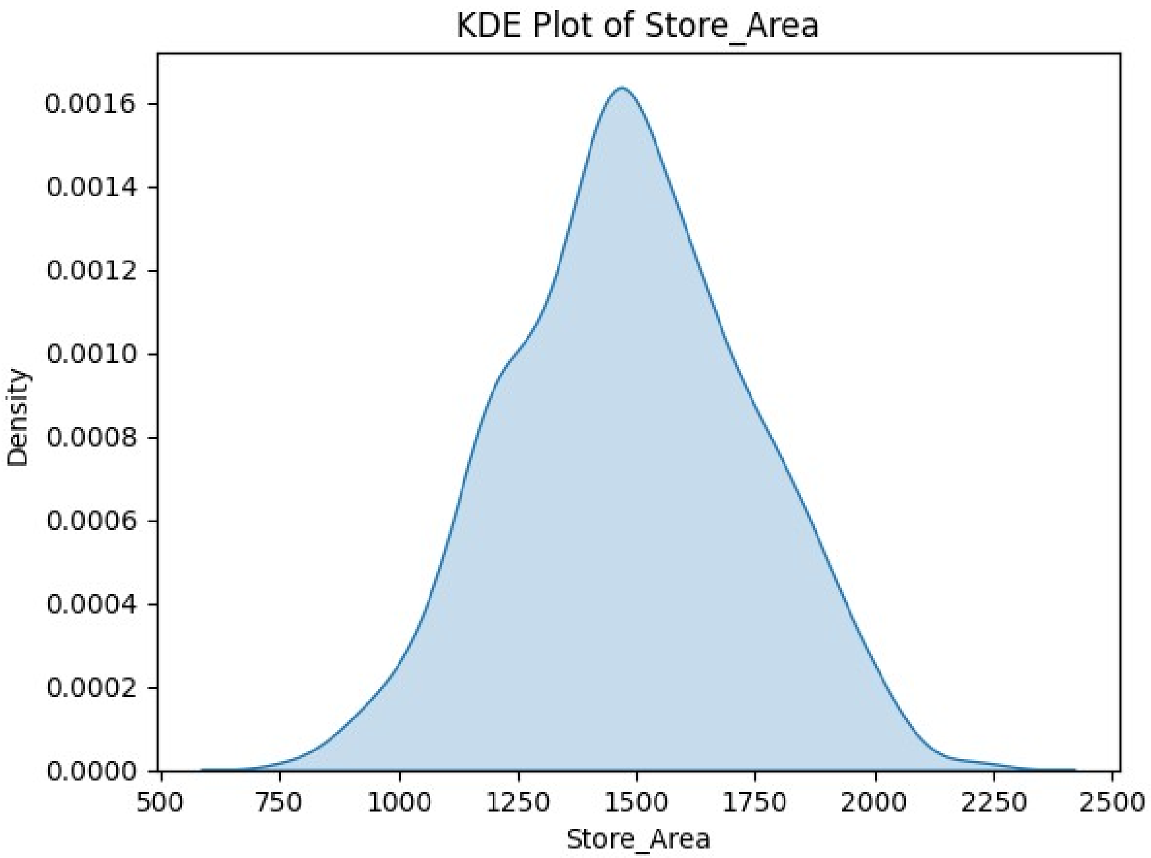

Boxplot of Items_Available
1000
1250
1500
1750
2000
2250
2500
2750
Items_Available


Boxplot of Daily_Customer_Count
0
200
400
600
800
1000
1200
1400
1600
Daily_Customer_Count

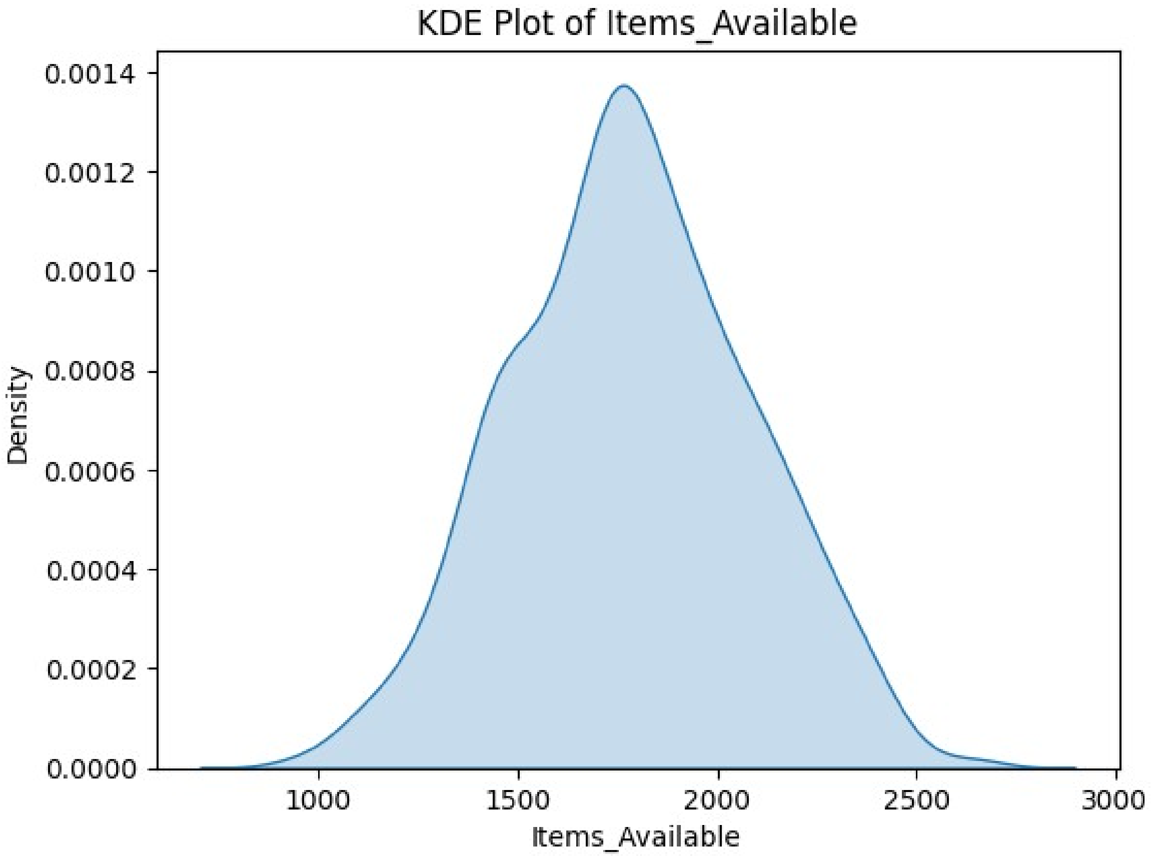


```
for col in df.columns[1:]:
    sns.kdeplot(df[col], shade=True)
    plt.title(f"KDE Plot of {col}")
    plt.show()
```

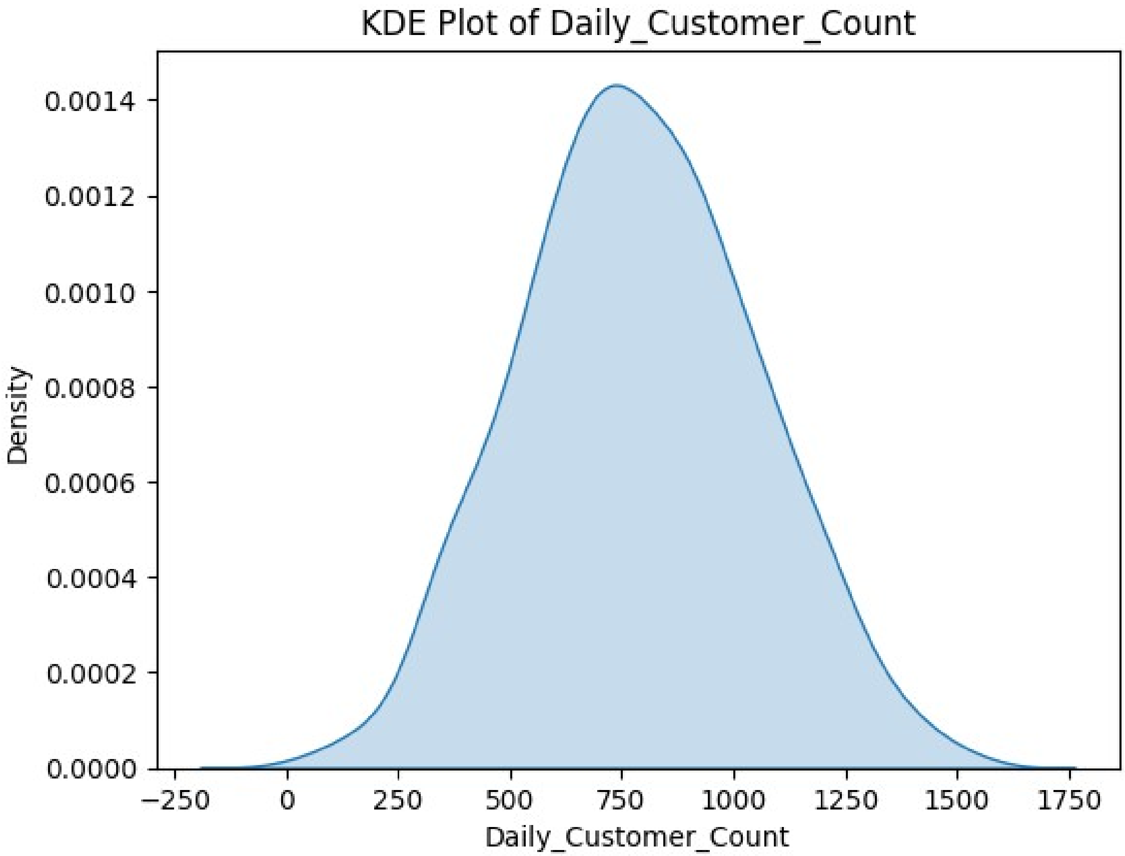
```
/tmp/ipython-input-1999266753.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(df[col], shade=True)
```



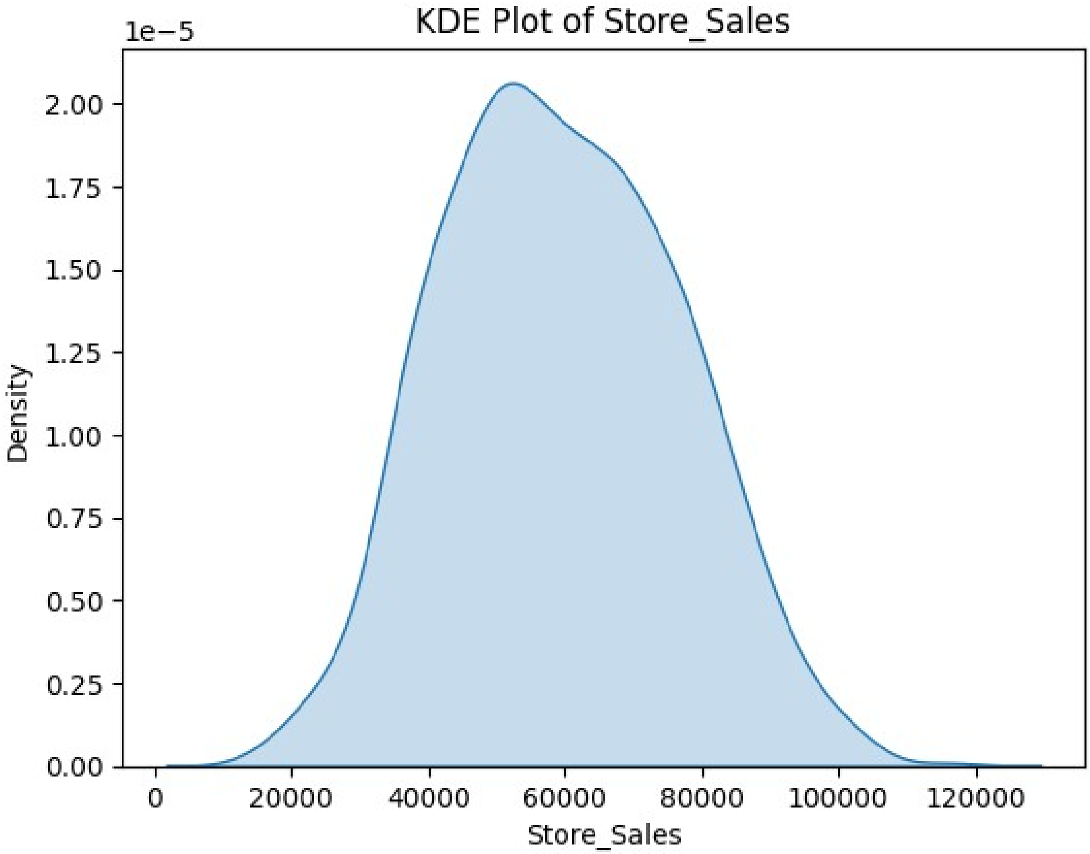
```
/tmp/ipython-input-1999266753.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(df[col], shade=True)
```



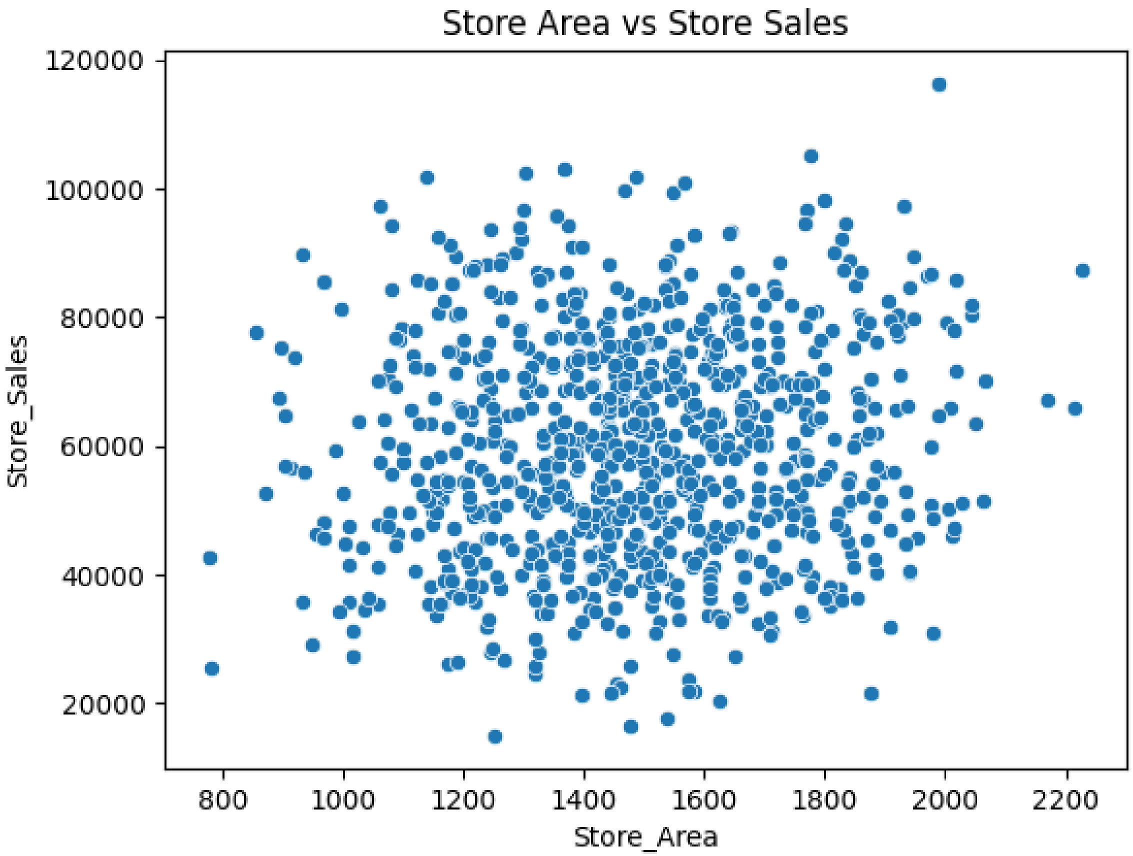
```
/tmp/ipython-input-1999266753.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(df[col], shade=True)
```



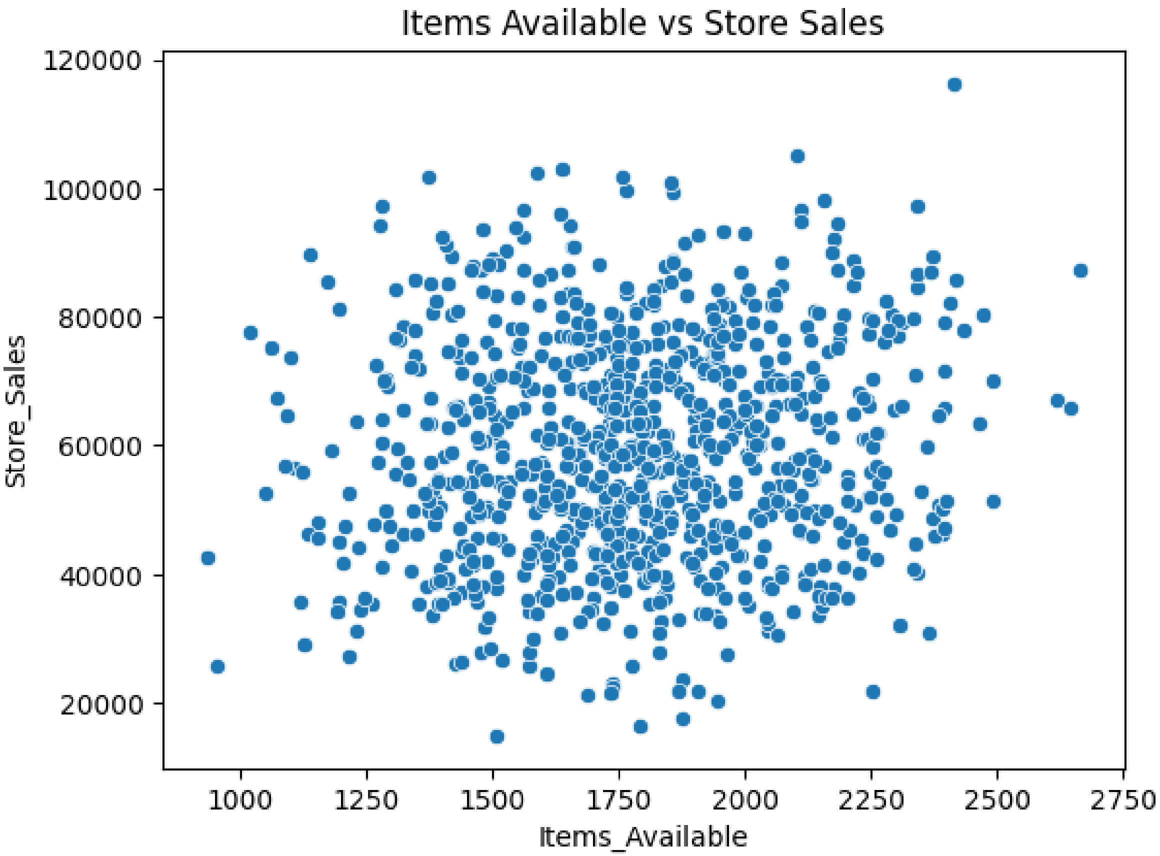
```
/tmp/ipython-input-1999266753.py:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(df[col], shade=True)
```



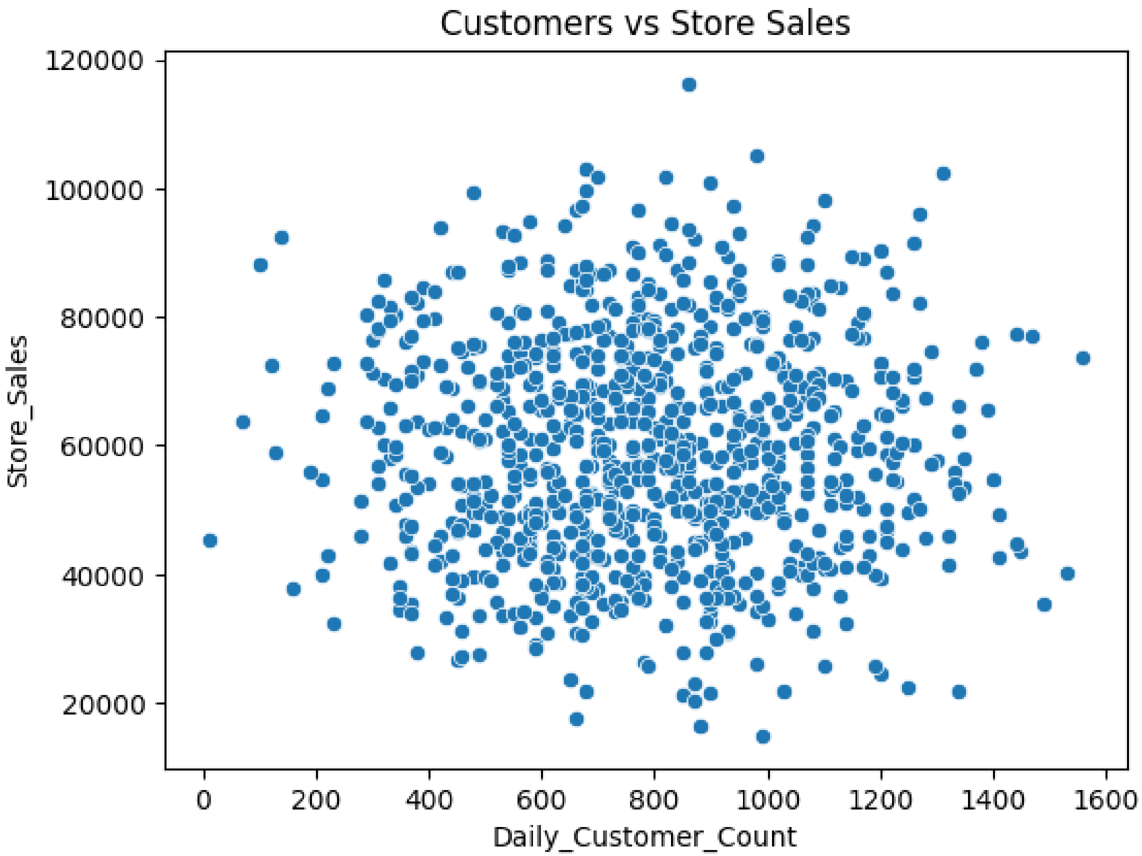
```
sns.scatterplot(x="Store_Area", y="Store_Sales", data=df)
plt.title("Store Area vs Store Sales")
plt.show()
```



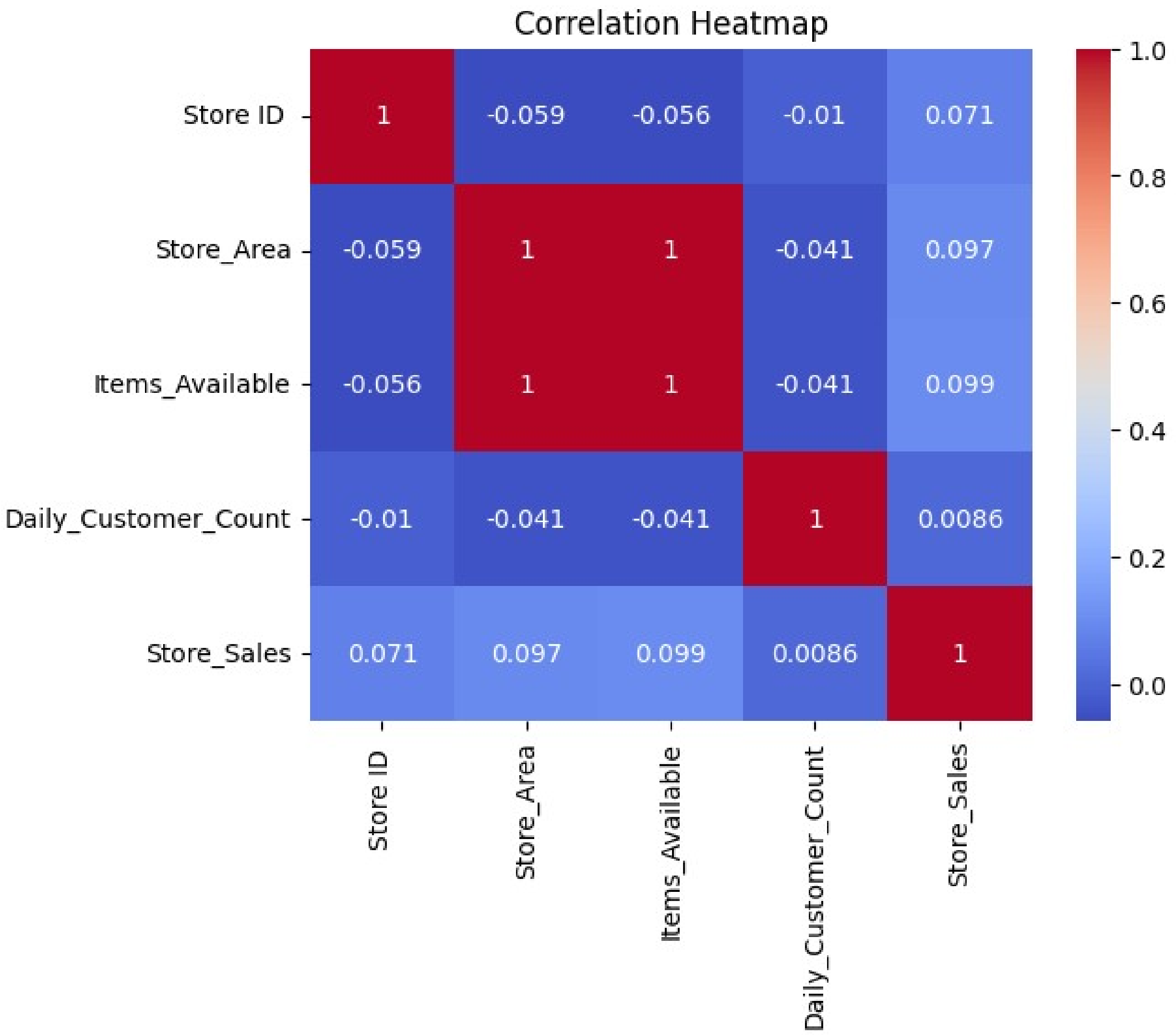
```
sns.scatterplot(x="Items_Available", y="Store_Sales", data=df)
plt.title("Items Available vs Store Sales")
plt.show()
```



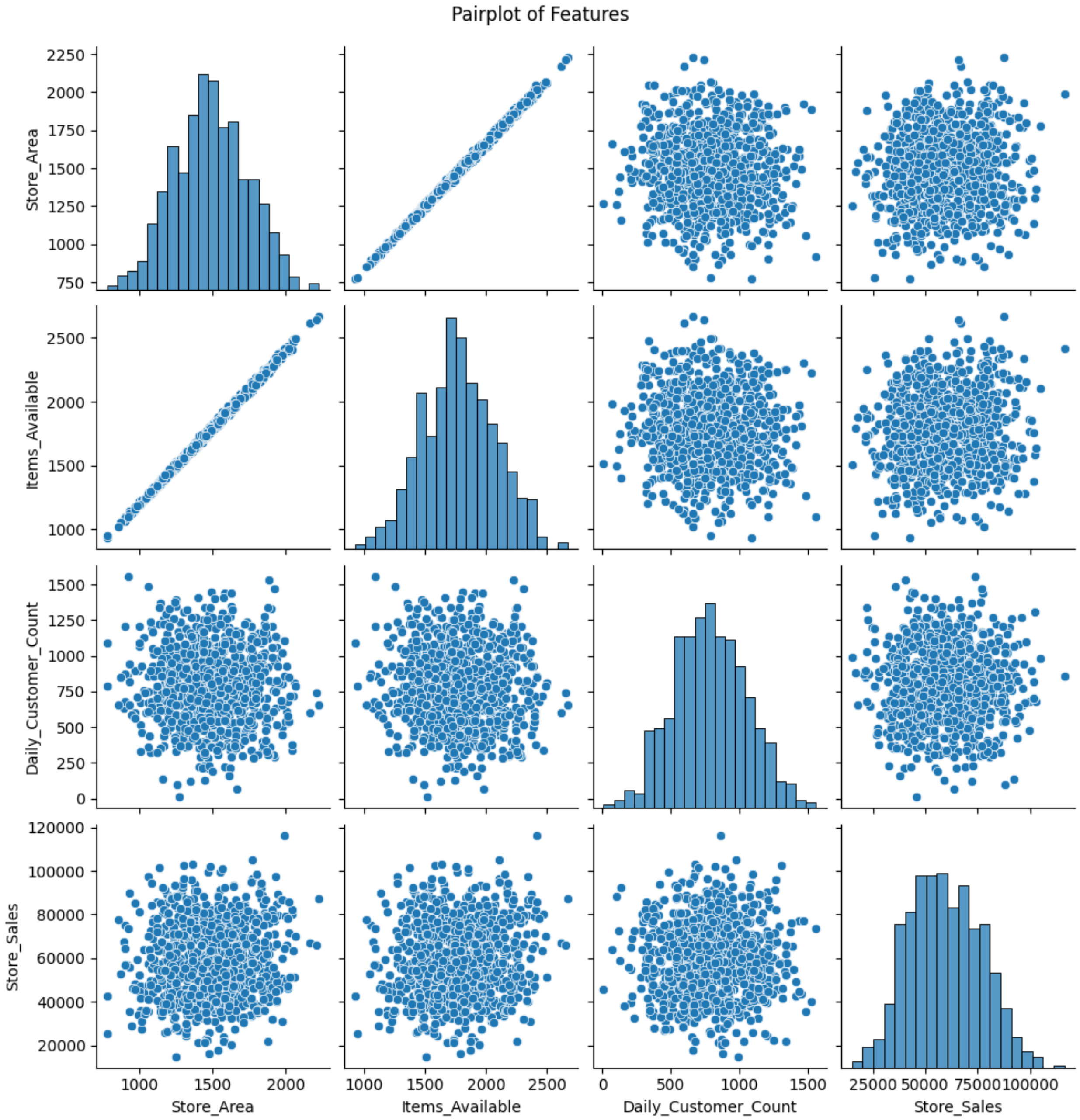
```
sns.scatterplot(x="Daily_Customer_Count", y="Store_Sales", data=df)
plt.title("Customers vs Store Sales")
plt.show()
```



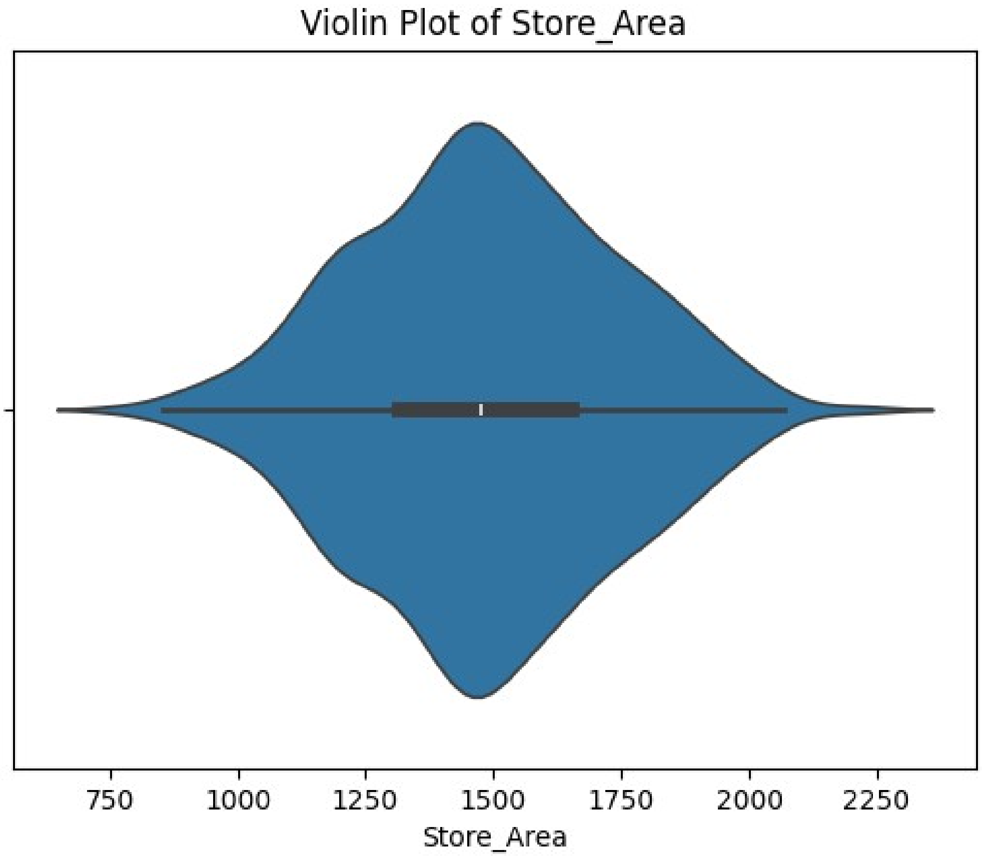
```
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

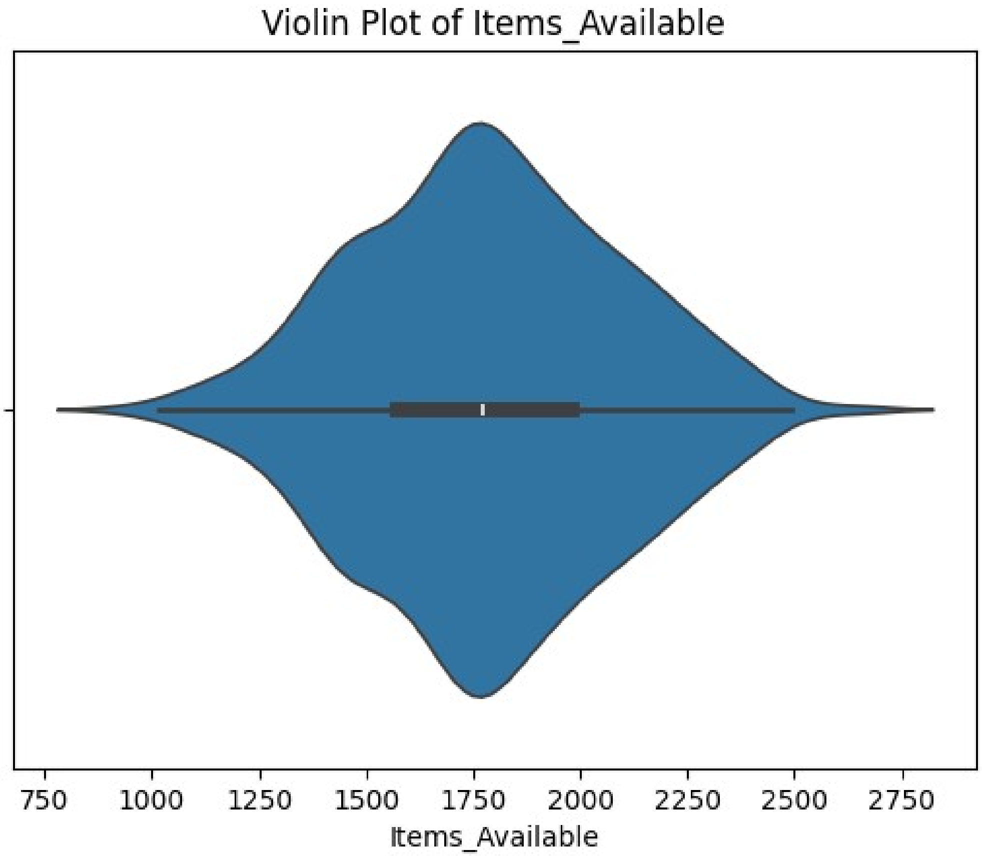


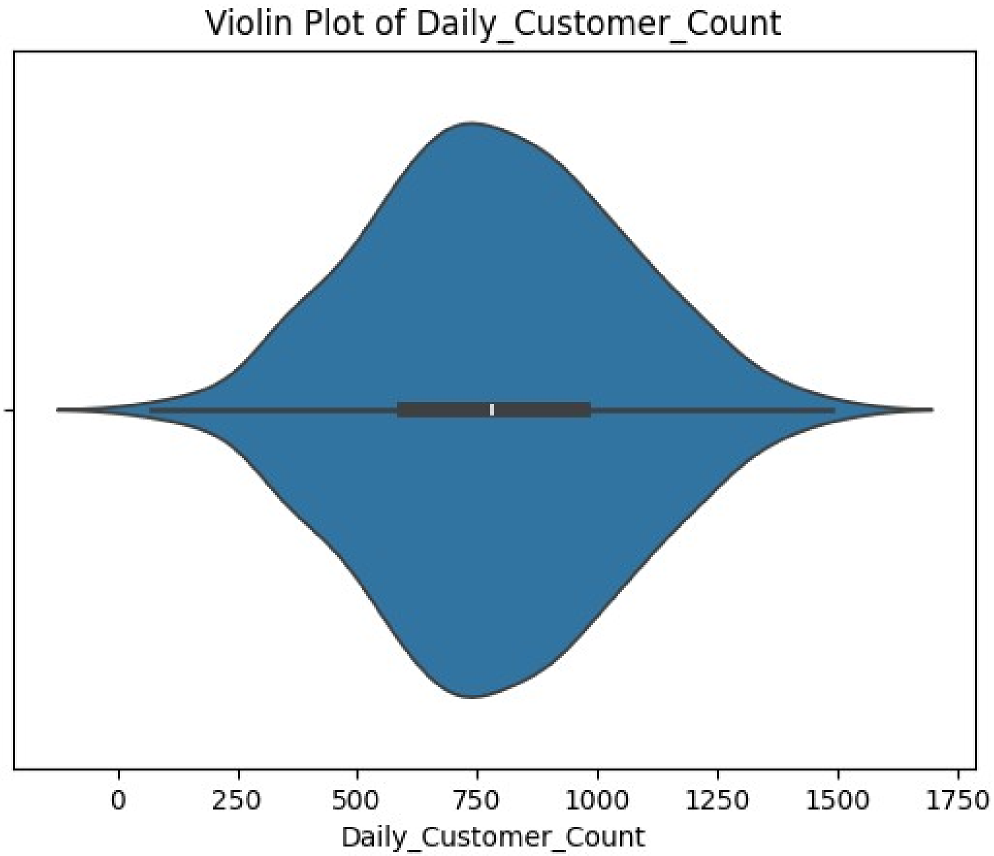
```
sns.pairplot(df.drop("Store ID ", axis=1))
plt.suptitle("Pairplot of Features", y=1.02)
plt.show()
```

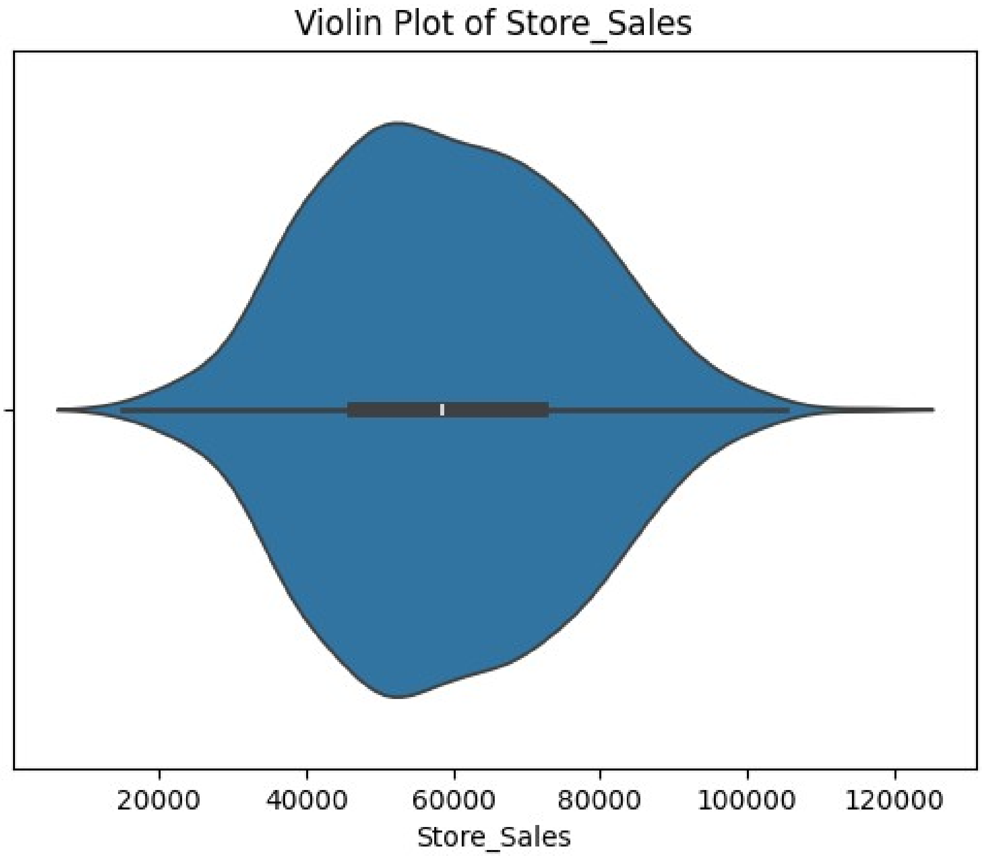


```python
for col in ["Store_Area", "Items_Available", "Daily_Customer_Count",
"Store_Sales"]:
    sns.violinplot(x=df[col])
    plt.title(f"Violin Plot of {col}")
    plt.show()
```

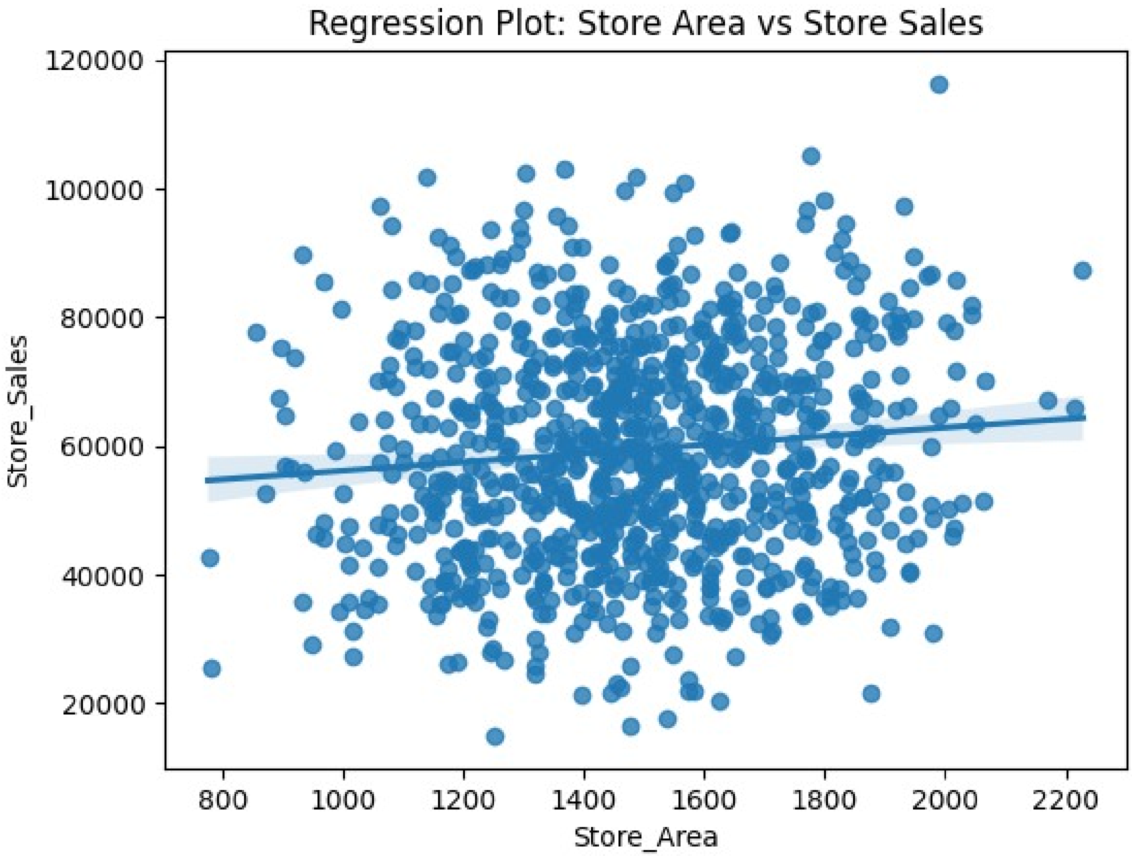

Violin Plot of Store_Area
750
1000
1250
1500
1750
2000
2250
Store_Area


Violin Plot of Items_Available
750
1000
1250
1500
1750
2000
2250
2500
2750
Items_Available

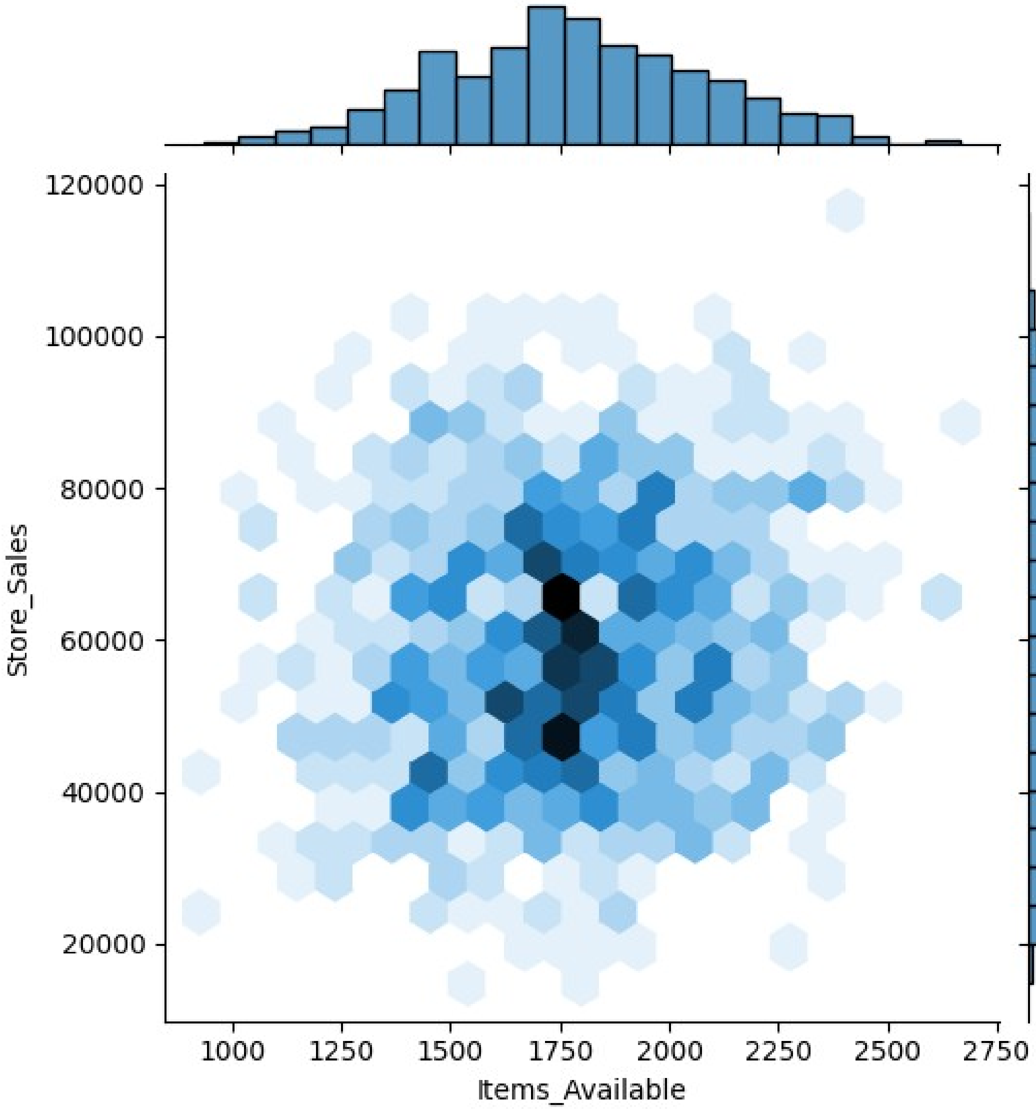

Violin Plot of Daily_Customer_Count
0
250
500
750
1000
1250
1500
1750
Daily_Customer_Count



```
sns.regplot(x="Store_Area", y="Store_Sales", data=df)
plt.title("Regression Plot: Store Area vs Store Sales")
plt.show()
```



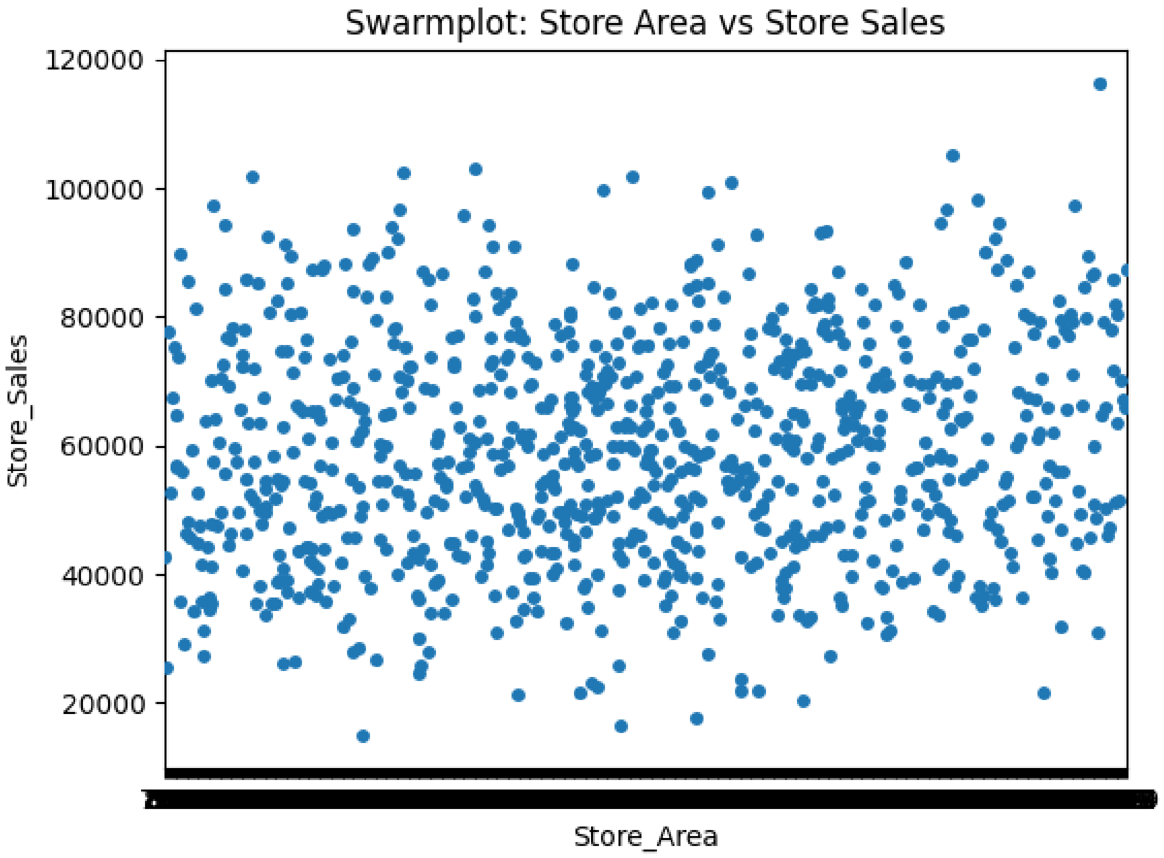
```
sns.jointplot(x="Items_Available", y="Store_Sales", data=df,
kind="hex")
plt.show()
```



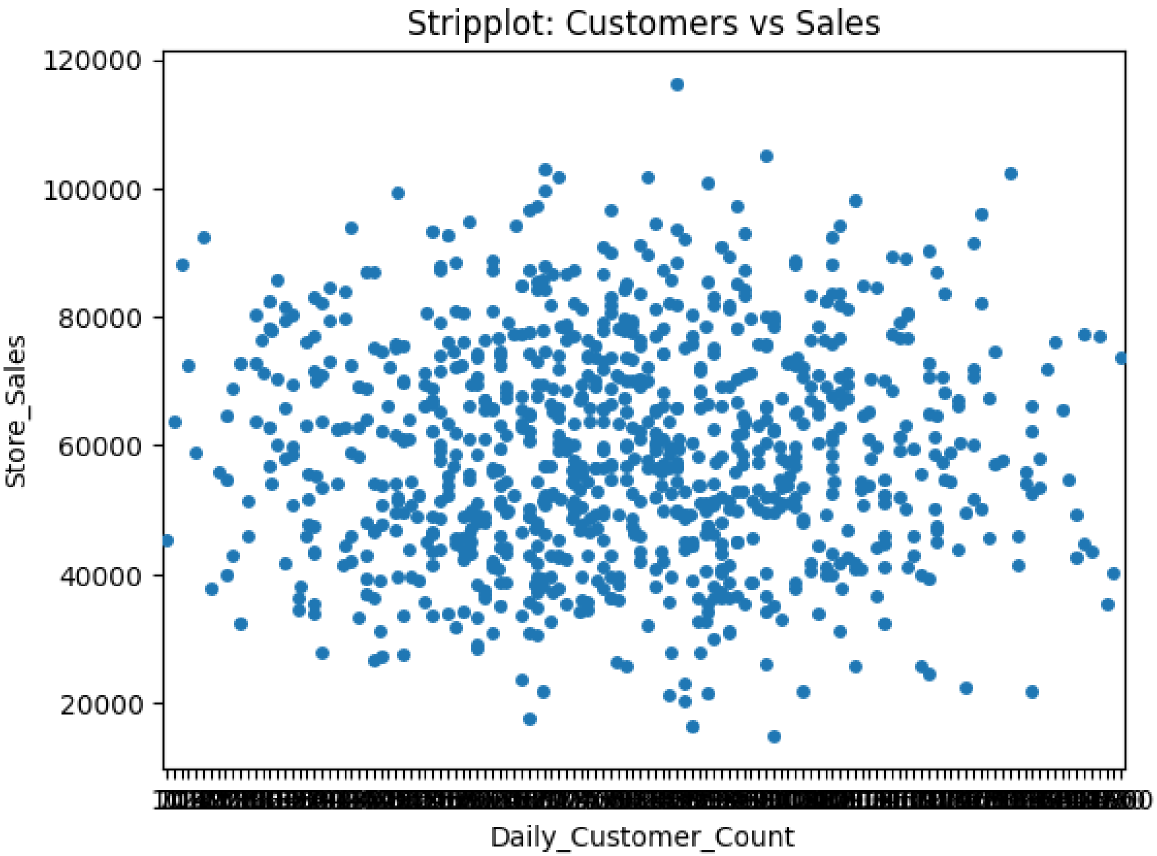
```
sns.swarmplot(x=df["Store_Area"], y=df["Store_Sales"])
plt.title("Swarmplot: Store Area vs Store Sales")
plt.show()
```

```
/usr/local/lib/python3.12/dist-packages/seaborn/categorical.py:3399:
UserWarning: 50.0% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.12/dist-packages/seaborn/categorical.py:3399:
UserWarning: 33.3% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.12/dist-packages/seaborn/categorical.py:3399:
UserWarning: 20.0% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
```

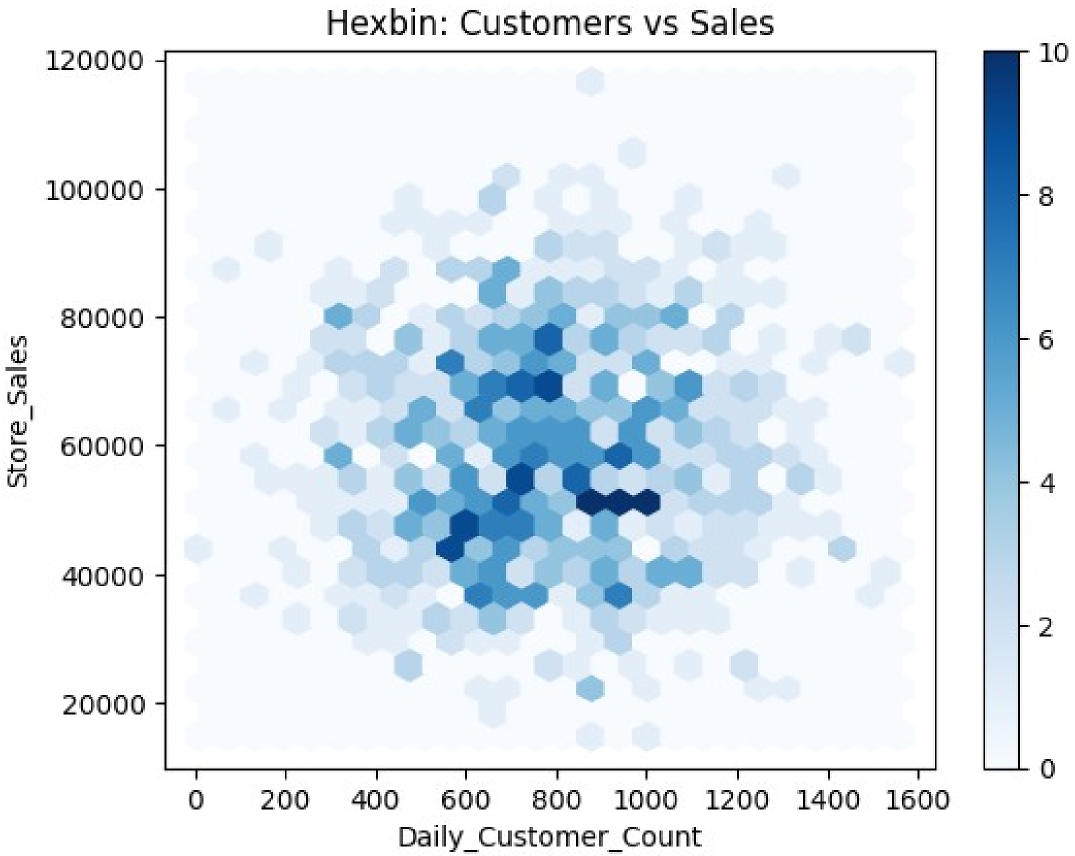
```
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.12/dist-packages/seaborn/categorical.py:3399:
UserWarning: 40.0% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.12/dist-packages/seaborn/categorical.py:3399:
UserWarning: 25.0% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```



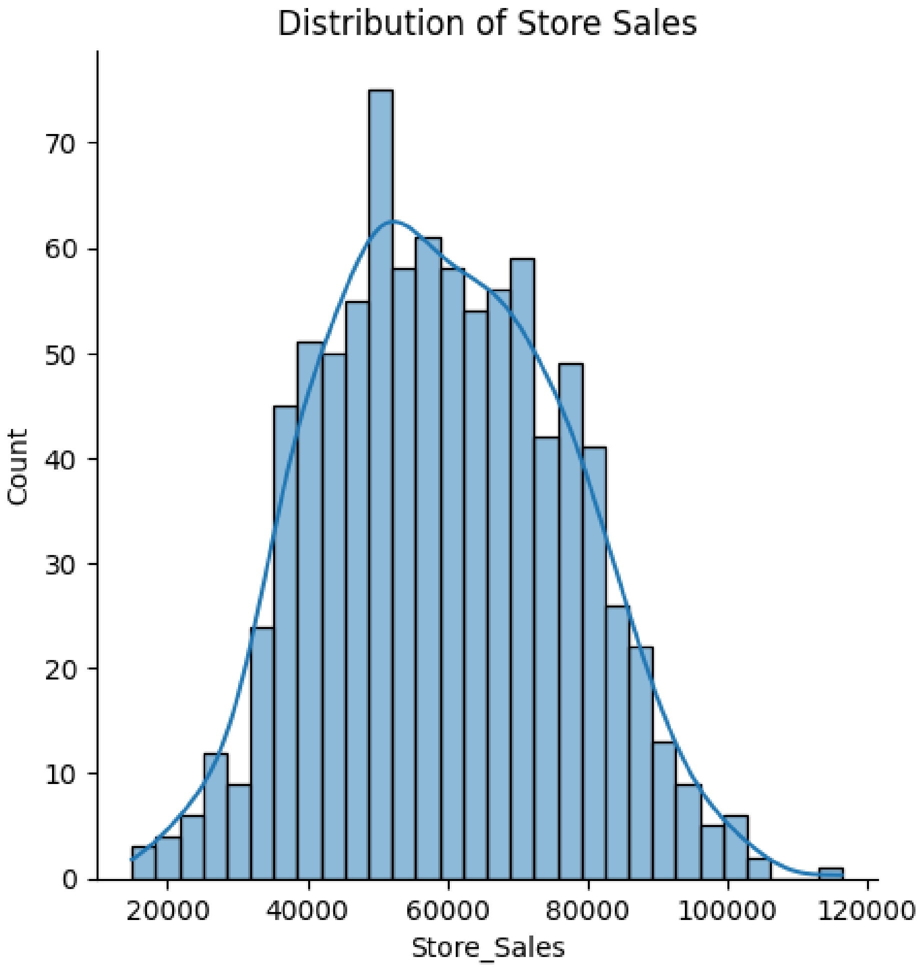
```
sns.stripplot(x=df["Daily_Customer_Count"], y=df["Store_Sales"])
plt.title("Stripplot: Customers vs Sales")
plt.show()
```



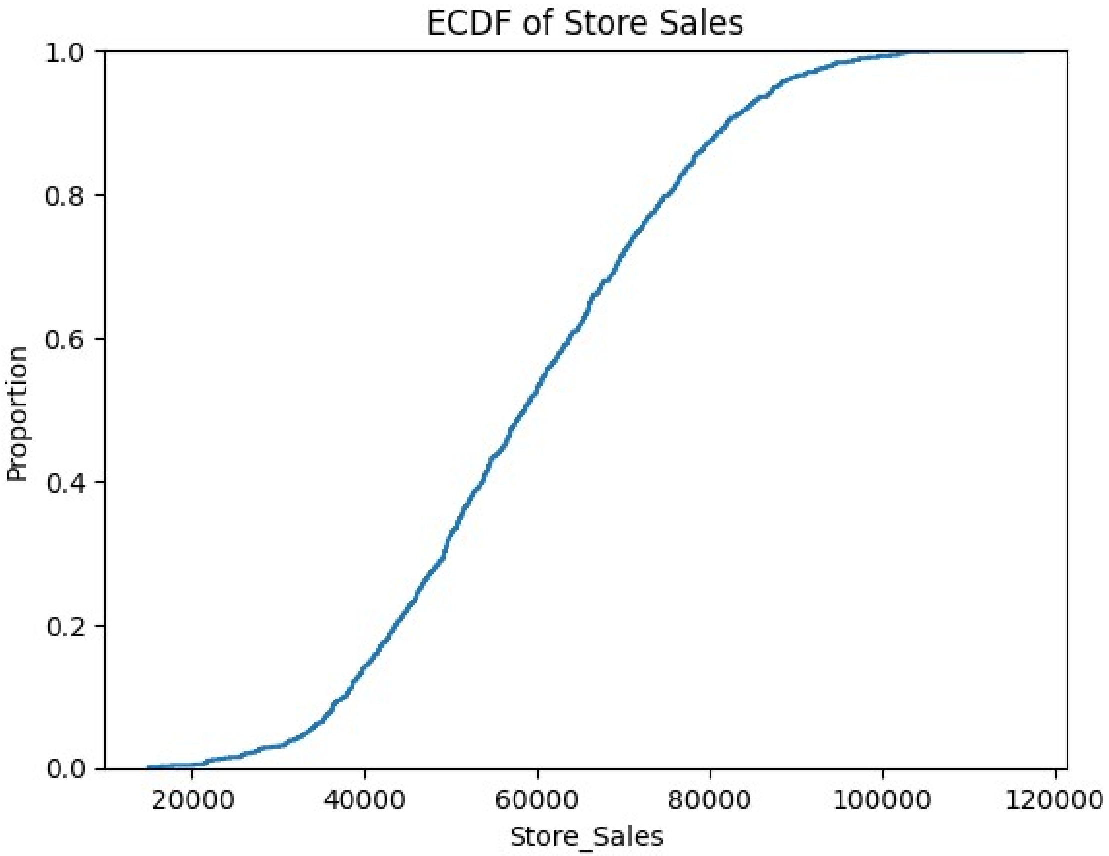
```
df.plot.hexbin(x="Daily_Customer_Count", y="Store_Sales", gridsize=25,
cmap="Blues")
plt.title("Hexbin: Customers vs Sales")
plt.show()
```



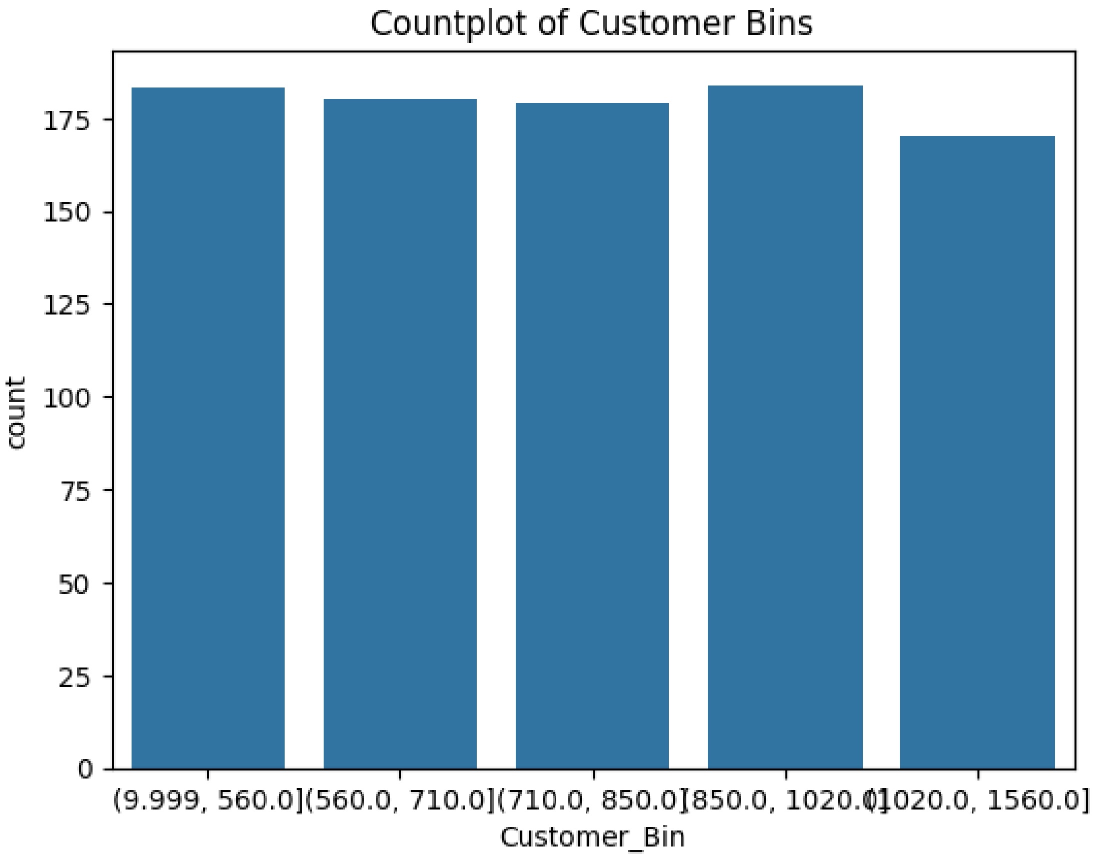
```
sns.displot(df["Store_Sales"], kde=True, bins=30)
plt.title("Distribution of Store Sales")
plt.show()
```



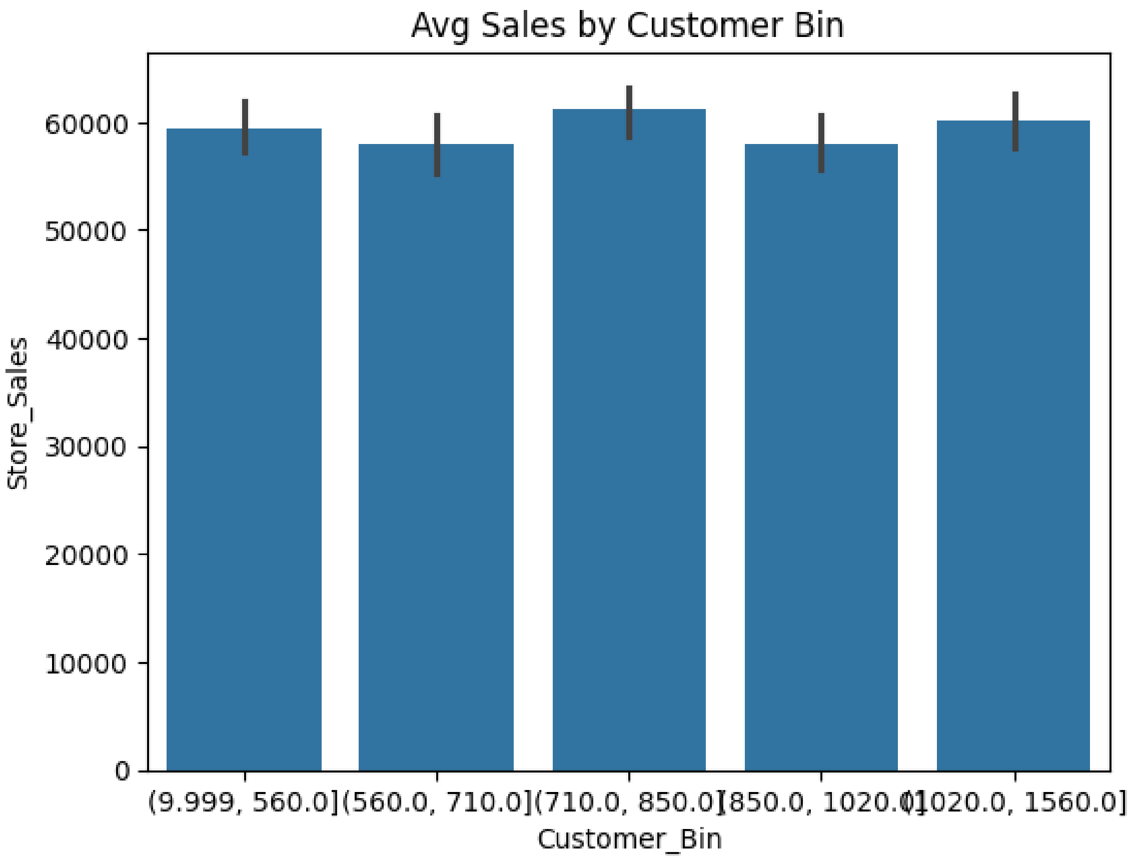
```
sns.ecdfplot(df["Store_Sales"])
plt.title("ECDF of Store Sales")
plt.show()
```



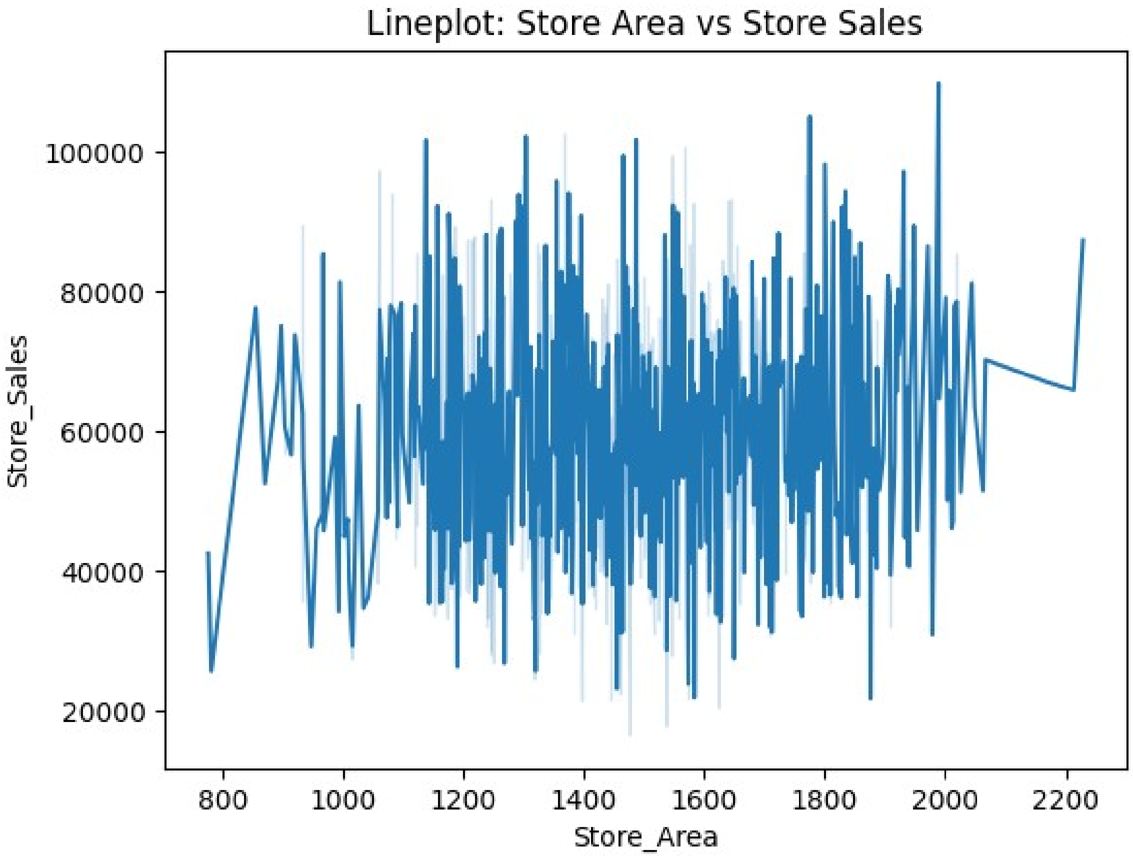
```python
df["Customer_Bin"] = pd.qcut(df["Daily_Customer_Count"], q=5)
sns.countplot(x="Customer_Bin", data=df)
plt.title("Countplot of Customer Bins")
plt.show()
```



```python
import numpy as np

sns.barplot(x="Customer_Bin", y="Store_Sales", data=df,
estimator=np.mean)
plt.title("Avg Sales by Customer Bin")
plt.show()
```

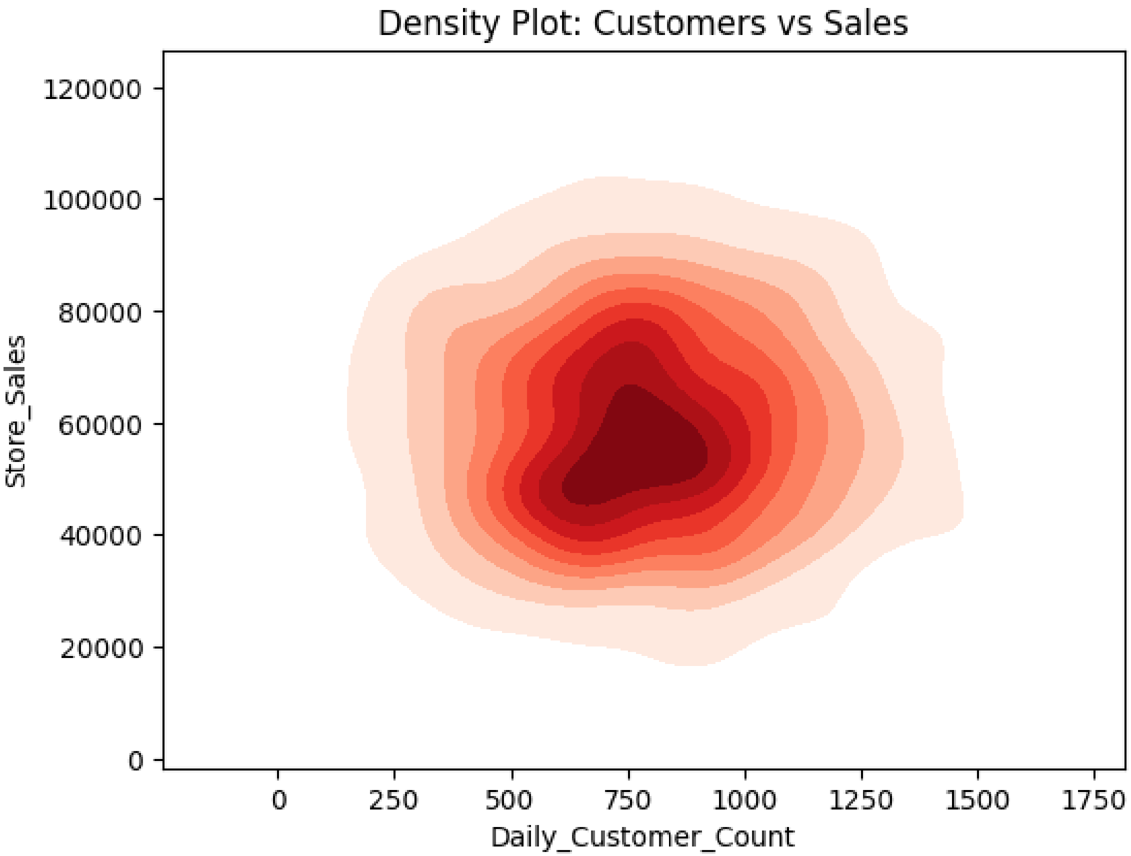


```
sns.lineplot(x="Store_Area", y="Store_Sales",
data=df.sort_values("Store_Area"))
plt.title("Lineplot: Store Area vs Store Sales")
plt.show()
```



```python
sns.kdeplot(x=df["Daily_Customer_Count"], y=df["Store_Sales"], 
cmap="Reds", shade=True)
plt.title("Density Plot: Customers vs Sales")
plt.show()
```

```
/tmp/ipython-input-1113677589.py:1: FutureWarning: 

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(x=df["Daily_Customer_Count"], y=df["Store_Sales"], 
cmap="Reds", shade=True)
```

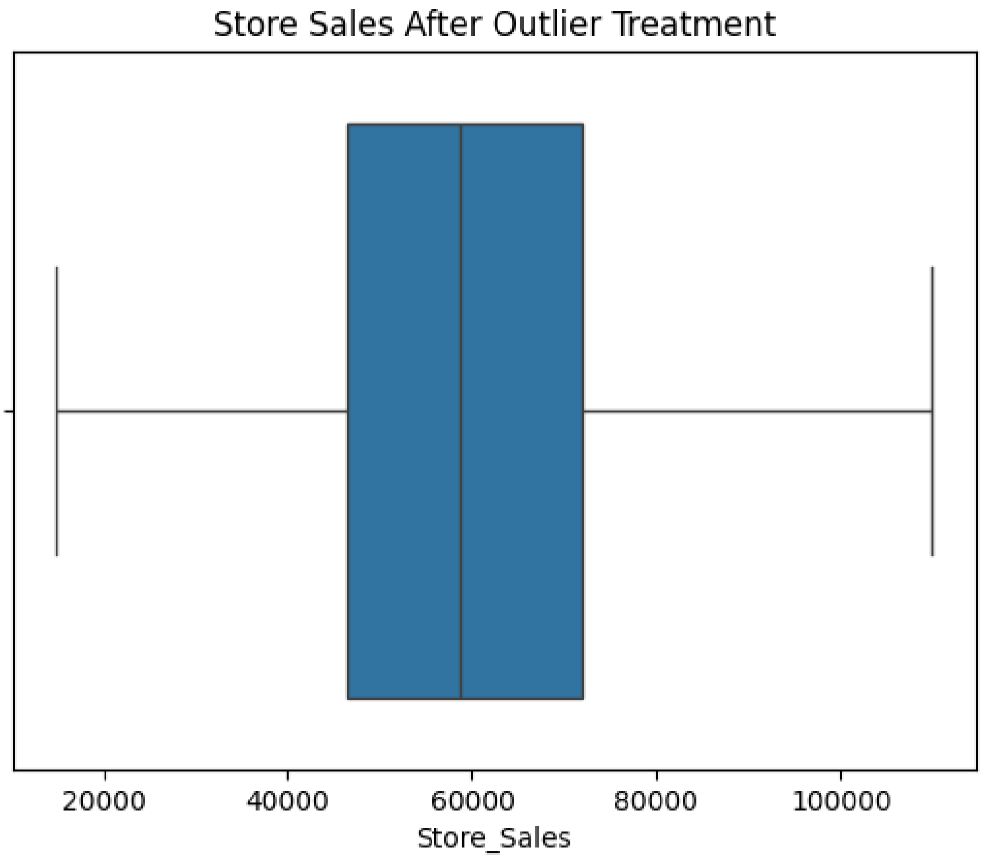


```
from scipy.stats import f_oneway, chi2_contingency


f_stat, p_val = f_oneway(df['Daily_Customer_Count'],
df['Store_Sales'])
print("ANOVA p-value:", p_val)
```

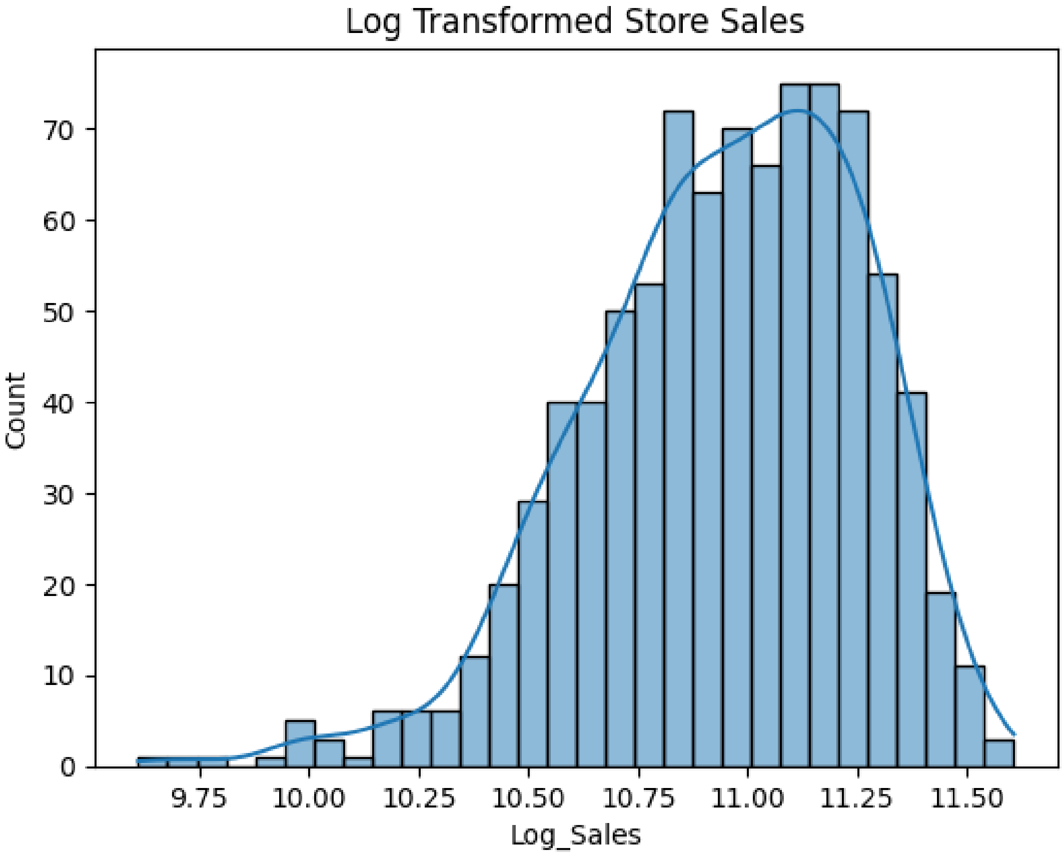
```
ANOVA p-value: 0.0
```

```
df['Sales_per_Customer'] = df['Store_Sales'] /
df['Daily_Customer_Count']
df['Items_Density'] = df['Items_Available'] / df['Store_Area']

Q1 = df['Store_Sales'].quantile(0.25)
Q3 = df['Store_Sales'].quantile(0.75)
IQR = Q3 - Q1
lower = Q1 - 1.5*IQR
upper = Q3 + 1.5*IQR
df['Store_Sales'] = np.where(df['Store_Sales'] > upper, upper,
df['Store_Sales'])

df['Log_Sales'] = np.log1p(df['Store_Sales'])
```

```
print(df.select_dtypes(include=['int64','float64']).corr()
['Store_Sales'].sort_values(ascending=False))

Store_Sales            1.000000
Log_Sales              0.979127
Sales_per_Customer     0.139473
Items_Available        0.098097
Store_Area             0.096759
Store ID               0.071254
Items_Density          0.022556
Daily_Customer_Count   0.008524
Name: Store_Sales, dtype: float64

sns.boxplot(x=df['Store_Sales'])
plt.title("Store Sales After Outlier Treatment")
plt.show()
```



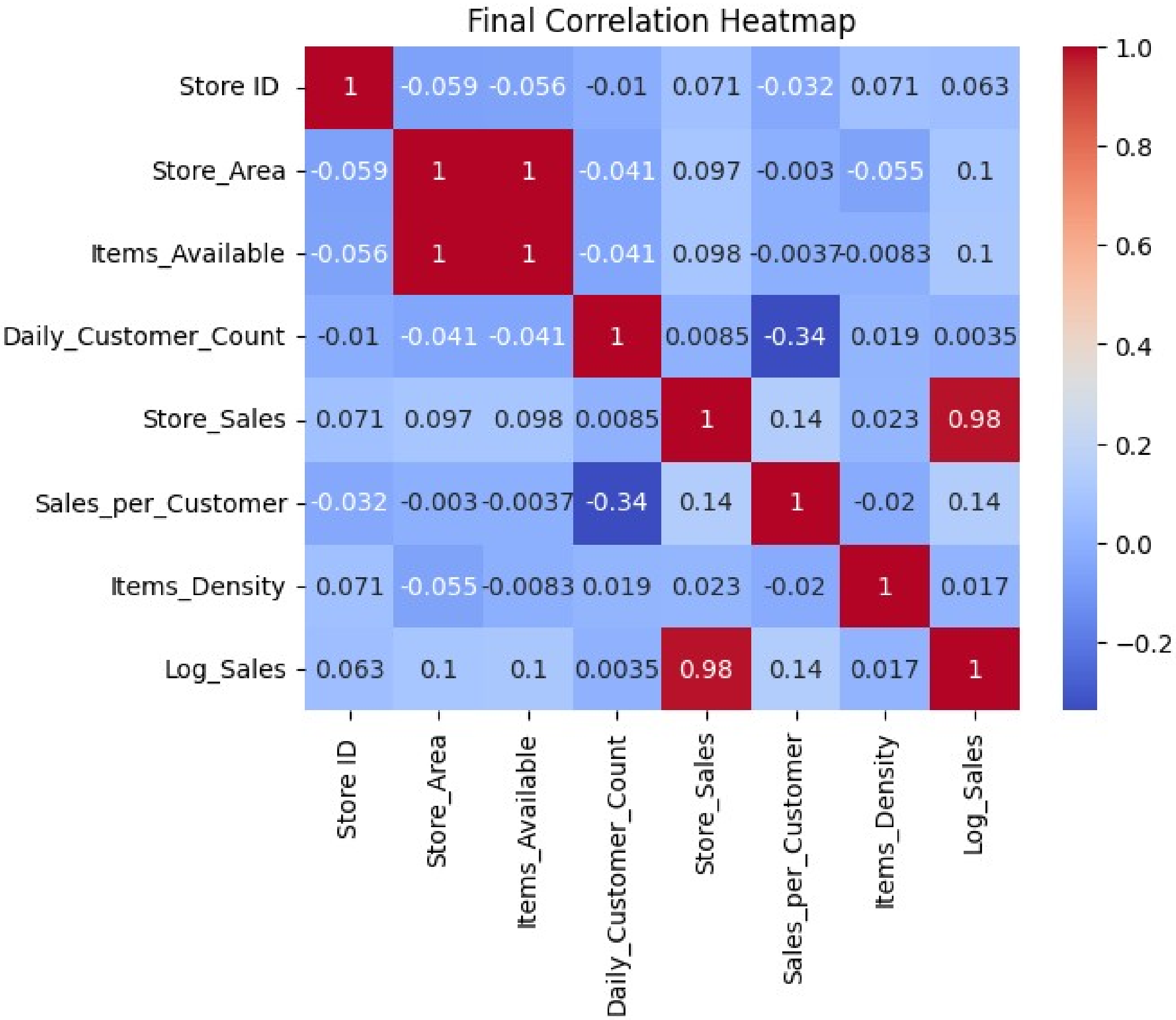
```
sns.histplot(df['Log_Sales'], kde=True, bins=30)
plt.title("Log Transformed Store Sales")
plt.show()
```



```
print("Final Shape:", df.shape)
print("Final Columns:", df.columns)

Final Shape: (896, 9)
Final Columns: Index(['Store ID ', 'Store_Area', 'Items_Available',
'Daily_Customer_Count',
       'Store_Sales', 'Customer_Bin', 'Sales_per_Customer',
'Items_Density',
       'Log_Sales'],
      dtype='object')

print(df.describe(include="all").T)

                      count unique              top freq          mean
\
Store ID              896.0    NaN              NaN  NaN         448.5

Store_Area            896.0    NaN              NaN  NaN   1485.409598

Items_Available       896.0    NaN              NaN  NaN   1782.035714

Daily_Customer_Count  896.0    NaN              NaN  NaN    786.350446

Store_Sales           896.0    NaN              NaN  NaN  59344.125279
```

```
Customer_Bin            896      5  (850.0, 1020.0]  184           NaN

Sales_per_Customer    896.0    NaN              NaN  NaN     94.034825

Items_Density         896.0    NaN              NaN  NaN       1.19978

Log_Sales             896.0    NaN              NaN  NaN     10.945311


                               std        min        25%        50%  \
Store ID                258.797218        1.0     224.75      448.5
Store_Area              250.237011      775.0    1316.75     1477.0
Items_Available         299.872053      932.0     1575.5     1773.5
Daily_Customer_Count    265.389281       10.0      600.0      780.0
Store_Sales           17168.248608    14920.0    46530.0    58605.0
Customer_Bin                   NaN        NaN        NaN        NaN
Sales_per_Customer      162.817175  15.070707  55.875573  75.409239
Items_Density             0.009409    1.17359    1.19323   1.199695
Log_Sales                 0.313244   9.610525  10.747872  10.978592

                             75%        max
Store ID                  672.25      896.0
Store_Area                1653.5     2229.0
Items_Available          1982.75     2667.0
Daily_Customer_Count       970.0     1560.0
Store_Sales              71872.5  109886.25
Customer_Bin                 NaN        NaN
Sales_per_Customer    102.853319     4548.0
Items_Density           1.206217    1.22833
Log_Sales              11.182662   11.60721
```

```
sns.heatmap(df.select_dtypes(include=['int64','float64']).corr(),
annot=True, cmap="coolwarm")
plt.title("Final Correlation Heatmap")
plt.show()
```



```
df.to_csv("Stores_Cleaned.csv", index=False)

from google.colab import files
files.download("Stores_Cleaned.csv")
```