Numaira Zaib

Step # 1

Load Dataset

```
import pandas as pd

# Load the dataset from Excel
file_path = "global_sales.xlsx"
df = pd.read_excel(file_path)

# Display first few rows
df.head()
```

```
   Row ID                  Order ID Order Date  Ship Date      Ship
Mode  \
0   40098  CA-2014-AB10015140-41954 2014-11-11 2014-11-13    First
Class
1   26341    IN-2014-JR162107-41675 2014-02-05 2014-02-07   Second
Class
2   25330    IN-2014-CR127307-41929 2014-10-17 2014-10-18    First
Class
3   13524   ES-2014-KM1637548-41667 2014-01-28 2014-01-30    First
Class
4   47221   SG-2014-RH9495111-41948 2014-11-05 2014-11-06      Same
Day

    Customer ID     Customer Name      Segment  Postal Code
City  \
0  AB-100151402     Aaron Bergman     Consumer      73120.0  Oklahoma
City
1     JR-162107     Justin Ritter    Corporate          NaN
Wollongong
2     CR-127307      Craig Reiter     Consumer          NaN
Brisbane
3    KM-1637548  Katherine Murray  Home Office          NaN
Berlin
4    RH-9495111       Rick Hansen     Consumer          NaN
Dakar

   ...    Product ID    Category Sub-Category  \
```

```
0  ...  TEC-PH-5816  Technology      Phones
1  ...  FUR-CH-5379   Furniture      Chairs
2  ...  TEC-PH-5356  Technology      Phones
3  ...  TEC-PH-5267  Technology      Phones
4  ...  TEC-CO-6011  Technology     Copiers

                                Product Name     Sales Quantity
Discount  \
0                           Samsung Convoy 3   221.980        2
0.0
1  Novimex Executive Leather Armchair, Black  3709.395        9
0.1
2          Nokia Smart Phone, with Caller ID  5175.171        9
0.1
3             Motorola Smart Phone, Cordless  2892.510        5
0.1
4             Sharp Wireless Fax, High-Speed  2832.960        8
0.0

     Profit  Shipping Cost  Order Priority
0   62.1544          40.77            High
1 -288.7650         923.63        Critical
2  919.9710         915.49          Medium
3  -96.5400         910.16          Medium
4  311.5200         903.04        Critical

[5 rows x 24 columns]
```

Step 2: Identify Key Issues

```
#Filter "Tables" Sub-Category

# Filter data for "Tables"
tables_data = df[df["Sub-Category"] == "Tables"]

# Display summary statistics
tables_data.describe()

             Row ID                     Order Date  \
count    861.000000                            861
mean   26285.506388  2014-05-04 16:16:43.484320768
min       38.000000            2012-01-03 00:00:00
25%    13573.000000            2013-05-25 00:00:00
50%    29786.000000            2014-06-24 00:00:00
75%    37167.000000            2015-06-04 00:00:00
max    51156.000000            2015-12-31 00:00:00
std    13835.593663                            NaN

                           Ship Date   Postal Code        Sales
Quantity  \
```

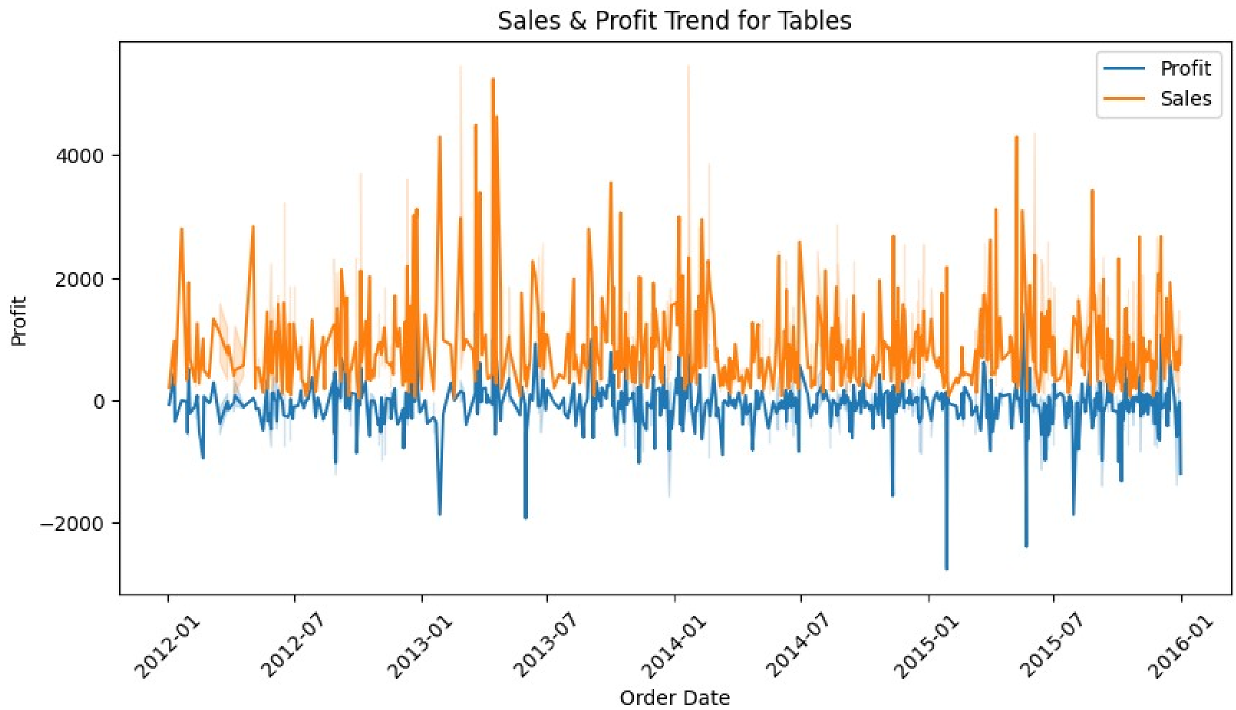
```
count                            861    319.000000   861.000000
861.000000
mean   2014-05-08 16:31:46.620208896  58331.749216   879.258913
3.580720
min              2012-01-07 00:00:00   1841.000000    24.368000
1.000000
25%              2013-05-30 00:00:00  27716.000000   330.588000
2.000000
50%              2014-06-27 00:00:00  61107.000000   629.064000
3.000000
75%              2015-06-07 00:00:00  90036.000000  1114.272000
5.000000
max              2016-01-04 00:00:00  99207.000000  5451.300000
14.000000
std                              NaN  32271.739155   796.402495
2.249972

         Discount       Profit  Shipping Cost
count  861.000000   861.000000     861.000000
mean     0.290732   -74.429023      92.756555
min      0.000000 -2750.280000       1.160000
25%      0.200000  -205.608000      28.240000
50%      0.300000   -34.647000      56.380000
75%      0.450000   103.040000     109.860000
max      0.850000  2071.440000     878.380000
std      0.220513   402.973963     113.654723
```

Visualize Profit Trends

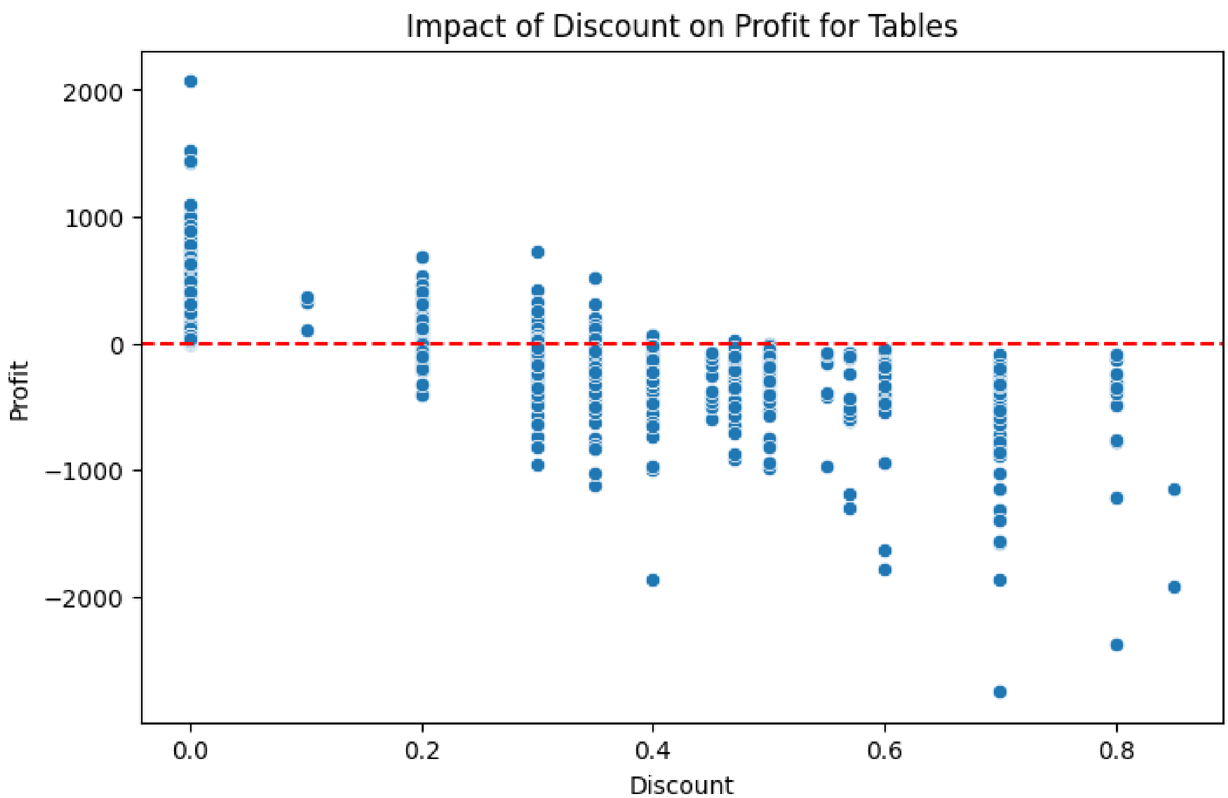
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Line chart: Sales vs. Profit for Tables
plt.figure(figsize=(10,5))
sns.lineplot(data=tables_data, x="Order Date", y="Profit",
label="Profit")
sns.lineplot(data=tables_data, x="Order Date", y="Sales",
label="Sales")
plt.xticks(rotation=45)
plt.title("Sales & Profit Trend for Tables")
plt.legend()
plt.show()
```



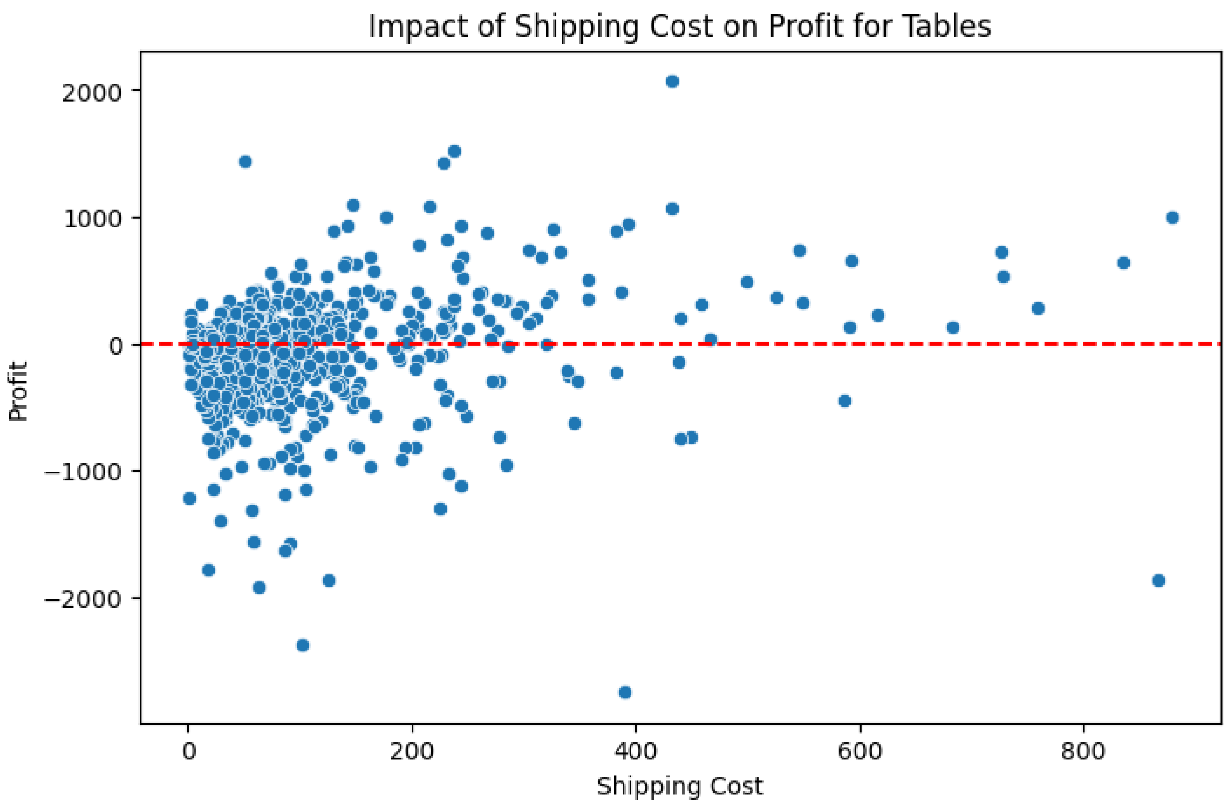
Analyze Discounts Given on Tables

```
# Scatter plot: Discount vs. Profit
plt.figure(figsize=(8,5))
sns.scatterplot(data=tables_data, x="Discount", y="Profit")
plt.axhline(0, color="red", linestyle="dashed")
plt.title("Impact of Discount on Profit for Tables")
plt.show()
```



Check Shipping Costs

```
# Scatter plot: Shipping Cost vs. Profit
plt.figure(figsize=(8,5))
sns.scatterplot(data=tables_data, x="Shipping Cost", y="Profit")
plt.axhline(0, color="red", linestyle="dashed")
plt.title("Impact of Shipping Cost on Profit for Tables")
plt.show()
```



Step 3: Hypothesis Testing

We now test if high discounts or shipping costs significantly impact profit.

Hypothesis for Discounts

Null Hypothesis (H_0): Discount has no impact on profit.

Alternative Hypothesis (H_1): High discounts lead to lower profit.

```
from scipy.stats import ttest_ind

# Split data into high and low discount groups
high_discount = tables_data[tables_data["Discount"] > 0.3]["Profit"]
low_discount = tables_data[tables_data["Discount"] <= 0.3]["Profit"]

# Perform t-test
t_stat, p_value = ttest_ind(high_discount, low_discount,
equal_var=False)

print("T-statistic:", t_stat)
print("P-value:", p_value)

# Interpretation
if p_value < 0.05:
```

```
    print("Reject H₀: Discounts significantly reduce profit.")
else:
    print("Fail to reject H₀: No significant effect of discounts.")
```

```
T-statistic: -18.536020501105984
P-value: 3.439121006897415e-61
Reject H₀: Discounts significantly reduce profit.
```

Hypothesis for Shipping Costs

Null Hypothesis (H_0): Shipping costs do not affect profit.

Alternative Hypothesis (H_1): High shipping costs reduce profit.

```
# Split data into high and low shipping cost groups
high_shipping = tables_data[tables_data["Shipping Cost"] >
tables_data["Shipping Cost"].median()]["Profit"]
low_shipping = tables_data[tables_data["Shipping Cost"] <=
tables_data["Shipping Cost"].median()]["Profit"]

# Perform t-test
t_stat, p_value = ttest_ind(high_shipping, low_shipping,
equal_var=False)

print("T-statistic:", t_stat)
print("P-value:", p_value)

# Interpretation
if p_value < 0.05:
    print("Reject H₀: High shipping costs significantly reduce
profit.")
else:
    print("Fail to reject H₀: No significant effect of shipping
costs.")
```

```
T-statistic: 2.6859757014004932
P-value: 0.007421585620149922
Reject H₀: High shipping costs significantly reduce profit.
```

Hypothesis for Relationship Between High-Value Orders and Shipping Costs

Null Hypothesis (H_0): There is no significant difference in shipping costs between high-value and low-value orders.

Alternative Hypothesis (H_1): High-value orders have significantly higher shipping costs.

```
import pandas as pd
import scipy.stats as stats

# Load data
df = pd.read_excel("global_sales.xlsx")

# Define high-value threshold (e.g., top 25% of orders)
high_value_threshold = df["Sales"].quantile(0.75)
df["High Value Order"] = df["Sales"] >= high_value_threshold

# Perform independent t-test
high_value_shipping = df[df["High Value Order"]]["Shipping Cost"]
low_value_shipping = df[~df["High Value Order"]]["Shipping Cost"]

t_stat, p_value = stats.ttest_ind(high_value_shipping,
low_value_shipping, equal_var=False)

# Interpretation
print(f"T-Statistic: {t_stat}")
print(f"P-value: {p_value}")

if p_value < 0.05:
    print("Reject H₀: High-value orders have significantly higher
shipping costs.")
else:
    print("Fail to reject H₀: No significant difference in shipping
costs between high and low-value orders.")

# Visualization
df.groupby("High Value Order")["Shipping
Cost"].mean().plot(kind="bar", title="Average Shipping Cost by Order
Value")
```

```
T-Statistic: 88.19517056691333
P-value: 0.0
Reject H₀: High-value orders have significantly higher shipping costs.

<Axes: title={'center': 'Average Shipping Cost by Order Value'},
xlabel='High Value Order'>
```


Average Shipping Cost by Order Value
80
70
60
50
40
30
20
10
0
False
True
High Value Order