ANALISIS POLA PEMBELIAN PADA DATA E-COMMERCE DENGAN ALGORITMA APRIORI

Kelompok 2

- Christopher Abie Diaz Doviano (00000067692)
- Jovanka Suryajaya (00000069834)
- Juanito Arvin William (00000069843)
- Julius Calvin Saputra (00000068626)

Import Library

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# associative learning libraries
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
# Membaca dataset 1
de=pd.read_csv('Order Details.csv')
de
```

Out[2]:

	Order ID	Amount	Profit	Quantity	Category	Sub-Category
0	B-25601	1275.0	-1148.0	7	Furniture	Bookcases
1	B-25601	66.0	-12.0	5	Clothing	Stole
2	B-25601	8.0	-2.0	3	Clothing	Hankerchief
3	B-25601	80.0	-56.0	4	Electronics	Electronic Games
4	B-25602	168.0	-111.0	2	Electronics	Phones
...	...	...	...	...	...	...
1495	B-26099	835.0	267.0	5	Electronics	Phones
1496	B-26099	2366.0	552.0	5	Clothing	Trousers
1497	B-26100	828.0	230.0	2	Furniture	Chairs
1498	B-26100	34.0	10.0	2	Clothing	T-shirt
1499	B-26100	72.0	16.0	2	Clothing	Shirt

1500 rows × 6 columns

In [3]:
```
# Membaca dataset 2
li=pd.read_csv('List of Orders.csv')
li
```

Out[3]:

	Order ID	Order Date	CustomerName	State	City
0	B-25601	01-04-2018	Bharat	Gujarat	Ahmedabad
1	B-25602	01-04-2018	Pearl	Maharashtra	Pune
2	B-25603	03-04-2018	Jahan	Madhya Pradesh	Bhopal
3	B-25604	03-04-2018	Divsha	Rajasthan	Jaipur
4	B-25605	05-04-2018	Kasheen	West Bengal	Kolkata
...	...	...	...	...	...
555	NaN	NaN	NaN	NaN	NaN
556	NaN	NaN	NaN	NaN	NaN
557	NaN	NaN	NaN	NaN	NaN
558	NaN	NaN	NaN	NaN	NaN
559	NaN	NaN	NaN	NaN	NaN

560 rows × 5 columns

In [4]:
```
# Melihat Tipe Data dataframe de
de.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Order ID      1500 non-null   object 
 1   Amount        1500 non-null   float64
 2   Profit        1500 non-null   float64
 3   Quantity      1500 non-null   int64  
 4   Category      1500 non-null   object 
 5   Sub-Category  1500 non-null   object 
dtypes: float64(2), int64(1), object(3)
memory usage: 70.4+ KB
```

In [5]:
```
# Melihat Tipe Data dataframe li
li.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 560 entries, 0 to 559
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Order ID      500 non-null    object
 1   Order Date    500 non-null    object
 2   CustomerName  500 non-null    object
 3   State         500 non-null    object
 4   City          500 non-null    object
dtypes: object(5)
memory usage: 22.0+ KB
```

Data Preparation

In [6]:
```
# Cek Missing Value de
de.isnull().sum()
```

Out[6]:
```
Order ID        0
Amount          0
Profit          0
Quantity        0
Category        0
Sub-Category    0
dtype: int64
```

In [7]:
```
# Cek Missing Value li
li.isnull().sum()
```

Out[7]:
```
Order ID        60
Order Date      60
CustomerName    60
State           60
City            60
dtype: int64
```

In [8]:
```
# Menghapus Missing Value dataframe de
de = de.dropna()
de
```

Out[8]:

	Order ID	Amount	Profit	Quantity	Category	Sub-Category
0	B-25601	1275.0	-1148.0	7	Furniture	Bookcases
1	B-25601	66.0	-12.0	5	Clothing	Stole
2	B-25601	8.0	-2.0	3	Clothing	Hankerchief
3	B-25601	80.0	-56.0	4	Electronics	Electronic Games
4	B-25602	168.0	-111.0	2	Electronics	Phones
...	...	...	...	...	...	...
1495	B-26099	835.0	267.0	5	Electronics	Phones
1496	B-26099	2366.0	552.0	5	Clothing	Trousers
1497	B-26100	828.0	230.0	2	Furniture	Chairs
1498	B-26100	34.0	10.0	2	Clothing	T-shirt
1499	B-26100	72.0	16.0	2	Clothing	Shirt

1500 rows × 6 columns

In [9]:

```
# Menghapus Missing Value dataframe li
li = li.dropna()
li
```

Out[9]:

	Order ID	Order Date	CustomerName	State	City
0	B-25601	01-04-2018	Bharat	Gujarat	Ahmedabad
1	B-25602	01-04-2018	Pearl	Maharashtra	Pune
2	B-25603	03-04-2018	Jahan	Madhya Pradesh	Bhopal
3	B-25604	03-04-2018	Divsha	Rajasthan	Jaipur
4	B-25605	05-04-2018	Kasheen	West Bengal	Kolkata
...	...	...	...	...	...
495	B-26096	28-03-2019	Atharv	West Bengal	Kolkata
496	B-26097	28-03-2019	Vini	Karnataka	Bangalore
497	B-26098	29-03-2019	Pinky	Jammu and Kashmir	Kashmir
498	B-26099	30-03-2019	Bhishm	Maharashtra	Mumbai
499	B-26100	31-03-2019	Hitika	Madhya Pradesh	Indore

500 rows × 5 columns

In [10]:

```
# Menggabungkan kedua dataframe
order = pd.merge(de, li, on='Order ID')
order
```

Out[10]:

	Order ID	Amount	Profit	Quantity	Category	Sub-Category	Order Date	CustomerNam
0	B-25601	1275.0	-1148.0	7	Furniture	Bookcases	01-04-2018	Bhara
1	B-25601	66.0	-12.0	5	Clothing	Stole	01-04-2018	Bhara
2	B-25601	8.0	-2.0	3	Clothing	Hankerchief	01-04-2018	Bhara
3	B-25601	80.0	-56.0	4	Electronics	Electronic Games	01-04-2018	Bhara
4	B-25602	168.0	-111.0	2	Electronics	Phones	01-04-2018	Pea
...	...	...	...	...	...	...	...	.
1495	B-26099	835.0	267.0	5	Electronics	Phones	30-03-2019	Bhishn
1496	B-26099	2366.0	552.0	5	Clothing	Trousers	30-03-2019	Bhishn
1497	B-26100	828.0	230.0	2	Furniture	Chairs	31-03-2019	Hitik
1498	B-26100	34.0	10.0	2	Clothing	T-shirt	31-03-2019	Hitik
1499	B-26100	72.0	16.0	2	Clothing	Shirt	31-03-2019	Hitik

1500 rows × 10 columns

In [11]:

```
# Drop kolom yang tidak digunakan
order = order.drop(['CustomerName', 'State', 'City'], axis=1)
order
```

Out[11]:

	Order ID	Amount	Profit	Quantity	Category	Sub-Category	Order Date
0	B-25601	1275.0	-1148.0	7	Furniture	Bookcases	01-04-2018
1	B-25601	66.0	-12.0	5	Clothing	Stole	01-04-2018
2	B-25601	8.0	-2.0	3	Clothing	Hankerchief	01-04-2018
3	B-25601	80.0	-56.0	4	Electronics	Electronic Games	01-04-2018
4	B-25602	168.0	-111.0	2	Electronics	Phones	01-04-2018
...	...	...	...	...	...	...	...
1495	B-26099	835.0	267.0	5	Electronics	Phones	30-03-2019
1496	B-26099	2366.0	552.0	5	Clothing	Trousers	30-03-2019
1497	B-26100	828.0	230.0	2	Furniture	Chairs	31-03-2019
1498	B-26100	34.0	10.0	2	Clothing	T-shirt	31-03-2019
1499	B-26100	72.0	16.0	2	Clothing	Shirt	31-03-2019

1500 rows × 7 columns

In [12]:

```
# Mengurutkan Kolom
order_date_index = order.columns.get_loc('Order Date')
order.insert(1, 'Order Date', order.pop('Order Date'))
order
```

Out[12]:

	Order ID	Order Date	Amount	Profit	Quantity	Category	Sub-Category
0	B-25601	01-04-2018	1275.0	-1148.0	7	Furniture	Bookcases
1	B-25601	01-04-2018	66.0	-12.0	5	Clothing	Stole
2	B-25601	01-04-2018	8.0	-2.0	3	Clothing	Hankerchief
3	B-25601	01-04-2018	80.0	-56.0	4	Electronics	Electronic Games
4	B-25602	01-04-2018	168.0	-111.0	2	Electronics	Phones
...	...	...	...	...	...	...	...
1495	B-26099	30-03-2019	835.0	267.0	5	Electronics	Phones
1496	B-26099	30-03-2019	2366.0	552.0	5	Clothing	Trousers
1497	B-26100	31-03-2019	828.0	230.0	2	Furniture	Chairs
1498	B-26100	31-03-2019	34.0	10.0	2	Clothing	T-shirt
1499	B-26100	31-03-2019	72.0	16.0	2	Clothing	Shirt

1500 rows × 7 columns

In [13]:

```
# Cek Tipe Data Dataframe Baru
order.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Order ID      1500 non-null   object 
 1   Order Date    1500 non-null   object 
 2   Amount        1500 non-null   float64
 3   Profit        1500 non-null   float64
 4   Quantity      1500 non-null   int64  
 5   Category      1500 non-null   object 
 6   Sub-Category  1500 non-null   object 
dtypes: float64(2), int64(1), object(4)
memory usage: 82.2+ KB
```

Karena Order Date masih dalam bentuk object, maka perlu dilakukan pengubahan tipe data menjadi datetime.

In [14]:

```
# Mengubah tipe dapa
order['Order Date'] = pd.to_datetime(order['Order Date'], format='%d-%m-%Y')
```

In [15]:
```
order.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Order ID      1500 non-null   object
 1   Order Date    1500 non-null   datetime64[ns]
 2   Amount        1500 non-null   float64
 3   Profit        1500 non-null   float64
 4   Quantity      1500 non-null   int64
 5   Category      1500 non-null   object
 6   Sub-Category  1500 non-null   object
dtypes: datetime64[ns](1), float64(2), int64(1), object(3)
memory usage: 82.2+ KB
```

In [16]:
```
# Manipulasi dan Ekstrak Data Hari dan Bulan

# Bulan
order['month'] = order['Order Date'].dt.month
order['month'] = order['month'].replace((1,2,3,4,5,6,7,8,9,10,11,12),
('January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
'September', 'October', 'November', 'December'))

# Hari
order['weekday'] = order['Order Date'].dt.weekday
order['weekday'] = order['weekday'].replace((0,1,2, 3,4, 5,6),
('Monday', 'Tuesday', 'Wednesday', 'Thursday',
'Friday', 'Saturday', 'Sunday'))

order
```

Out[16]:

	Order ID	Order Date	Amount	Profit	Quantity	Category	Sub-Category	month	weekc
0	B-25601	2018-04-01	1275.0	-1148.0	7	Furniture	Bookcases	April	Sunc
1	B-25601	2018-04-01	66.0	-12.0	5	Clothing	Stole	April	Sunc
2	B-25601	2018-04-01	8.0	-2.0	3	Clothing	Hankerchief	April	Sunc
3	B-25601	2018-04-01	80.0	-56.0	4	Electronics	Electronic Games	April	Sunc
4	B-25602	2018-04-01	168.0	-111.0	2	Electronics	Phones	April	Sunc
...	...	...	...	...	...	...	...	...	
1495	B-26099	2019-03-30	835.0	267.0	5	Electronics	Phones	March	Saturc
1496	B-26099	2019-03-30	2366.0	552.0	5	Clothing	Trousers	March	Saturc
1497	B-26100	2019-03-31	828.0	230.0	2	Furniture	Chairs	March	Sunc
1498	B-26100	2019-03-31	34.0	10.0	2	Clothing	T-shirt	March	Sunc
1499	B-26100	2019-03-31	72.0	16.0	2	Clothing	Shirt	March	Sunc

1500 rows × 9 columns

In [17]:

```
# Mengurutkan Kolom
order_date_index = order.columns.get_loc('Order Date')
order.insert(order_date_index + 1, 'month', order.pop('month'))
order.insert(order_date_index + 2, 'weekday', order.pop('weekday'))
order
```

Out[17]:

	Order ID	Order Date	month	weekday	Amount	Profit	Quantity	Category	Su Catego
0	B-25601	2018-04-01	April	Sunday	1275.0	-1148.0	7	Furniture	Bookca
1	B-25601	2018-04-01	April	Sunday	66.0	-12.0	5	Clothing	St
2	B-25601	2018-04-01	April	Sunday	8.0	-2.0	3	Clothing	Hankerch
3	B-25601	2018-04-01	April	Sunday	80.0	-56.0	4	Electronics	Electro Gan
4	B-25602	2018-04-01	April	Sunday	168.0	-111.0	2	Electronics	Phoi
...	...	...	...	...	...	...	...	...	
1495	B-26099	2019-03-30	March	Saturday	835.0	267.0	5	Electronics	Phoi
1496	B-26099	2019-03-30	March	Saturday	2366.0	552.0	5	Clothing	Trous
1497	B-26100	2019-03-31	March	Sunday	828.0	230.0	2	Furniture	Cha
1498	B-26100	2019-03-31	March	Sunday	34.0	10.0	2	Clothing	T-sl
1499	B-26100	2019-03-31	March	Sunday	72.0	16.0	2	Clothing	Sl

1500 rows × 9 columns

Exploratory Data Analysis

In [18]:

```
# Menampilkan Statistik Umum
order.describe()
```

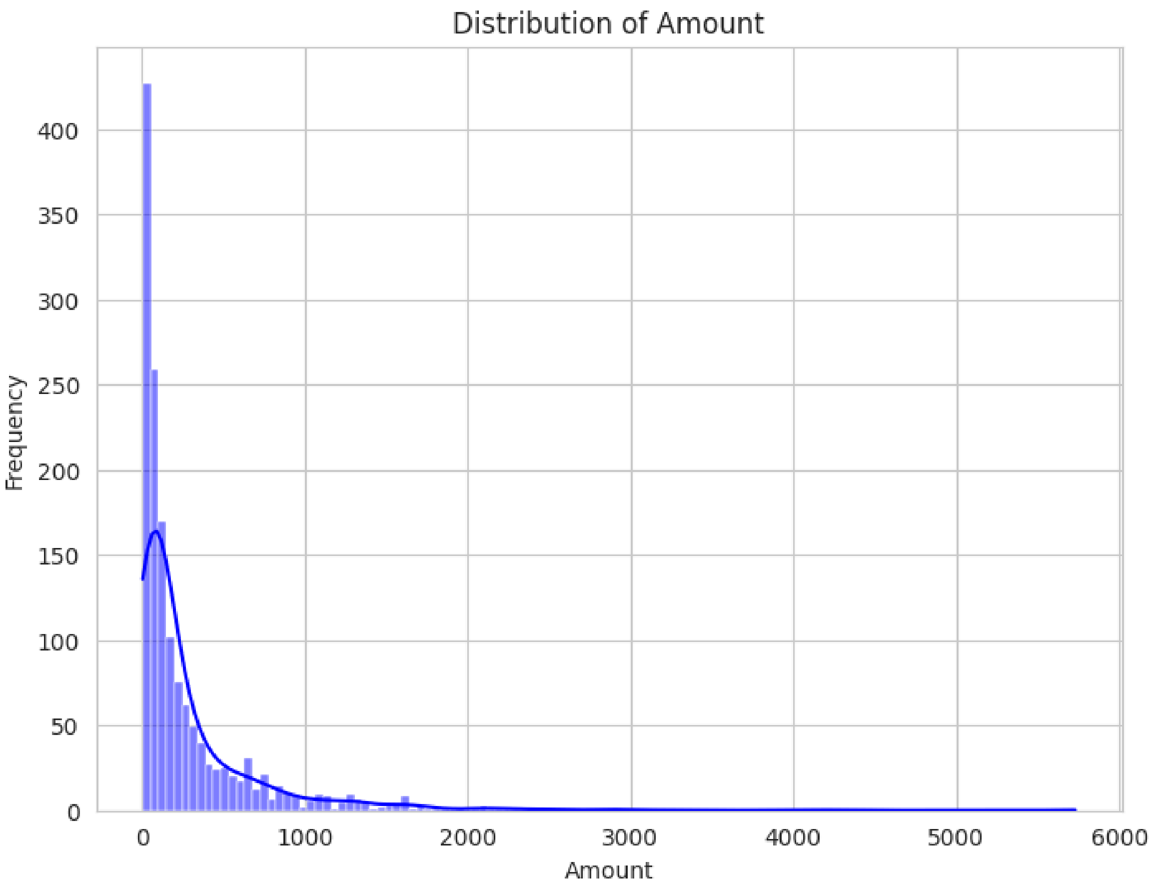
Out[18]:

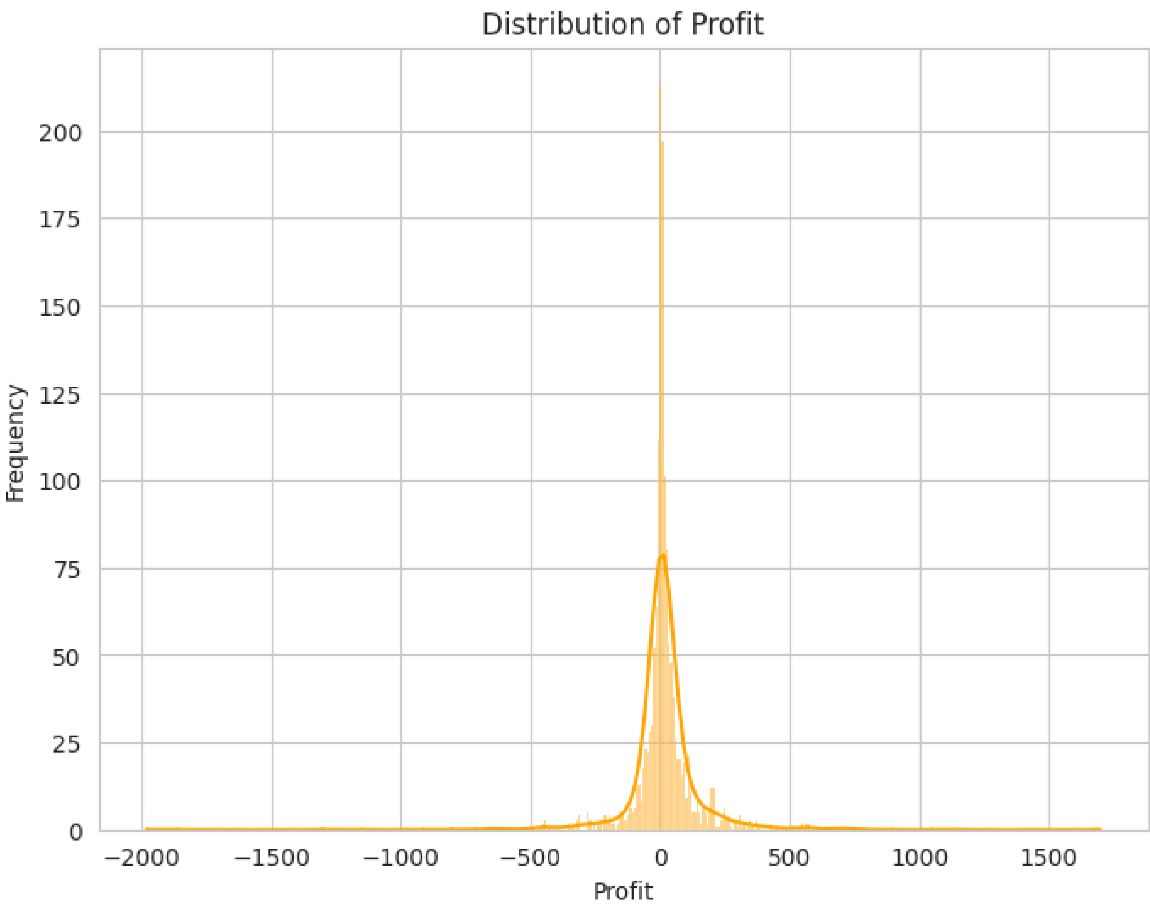
	Order Date	Amount	Profit	Quantity
count	1500	1500.000000	1500.000000	1500.000000
mean	2018-10-23 14:00:57.600000	287.668000	15.970000	3.743333
min	2018-04-01 00:00:00	4.000000	-1981.000000	1.000000
25%	2018-07-26 18:00:00	45.000000	-9.250000	2.000000
50%	2018-11-08 00:00:00	118.000000	9.000000	3.000000
75%	2019-01-27 00:00:00	322.000000	38.000000	5.000000
max	2019-03-31 00:00:00	5729.000000	1698.000000	14.000000
std	NaN	461.050488	169.140565	2.184942

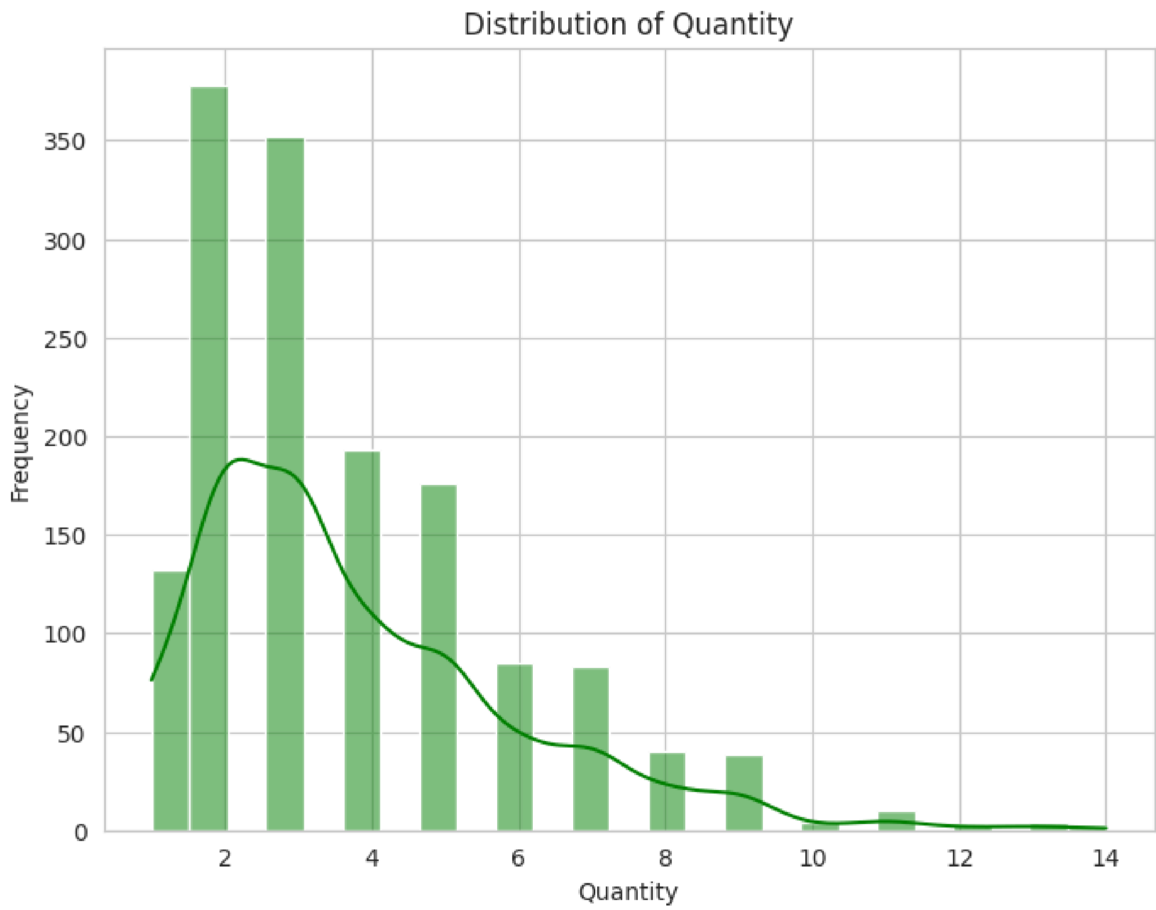
Visualisasi

In [19]:

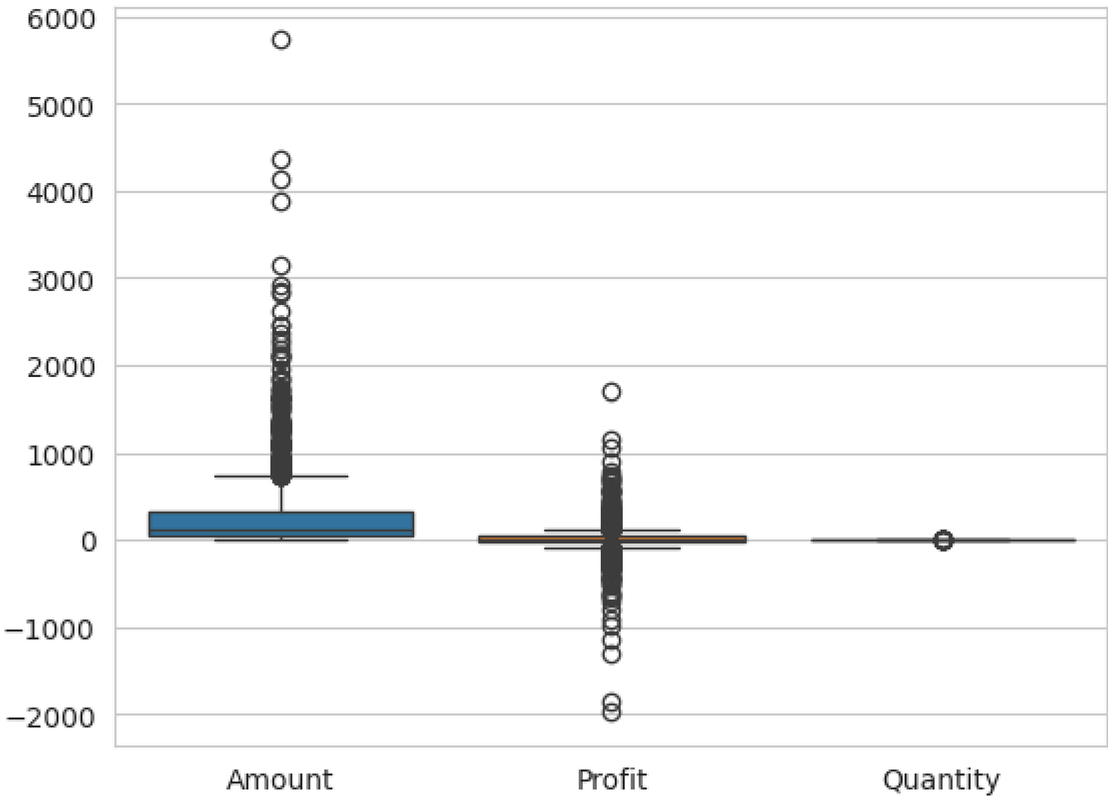
```
sns.set_style("whitegrid")

# Distribusi Variabel Amount
plt.figure(figsize=(8, 6))
sns.histplot(data=order, x='Amount', kde=True, color='blue')
plt.title('Distribution of Amount')
plt.xlabel('Amount')
plt.ylabel('Frequency')
plt.show()

# Distribusi Variabel Profit
plt.figure(figsize=(8, 6))
sns.histplot(data=order, x='Profit', kde=True, color='orange')
plt.title('Distribution of Profit')
plt.xlabel('Profit')
plt.ylabel('Frequency')
plt.show()

# Distribusi Variabel Quantity
plt.figure(figsize=(8, 6))
sns.histplot(data=order, x='Quantity', kde=True, color='green')
plt.title('Distribution of Quantity')
plt.xlabel('Quantity')
plt.ylabel('Frequency')
plt.show()
```


Distribution of Amount
Frequency
0
50
100
150
200
250
300
350
400
0
1000
2000
3000
4000
5000
6000
Amount


Distribution of Profit
Frequency
0
25
50
75
100
125
150
175
200
−2000
−1500
−1000
−500
0
500
1000
1500
Profit

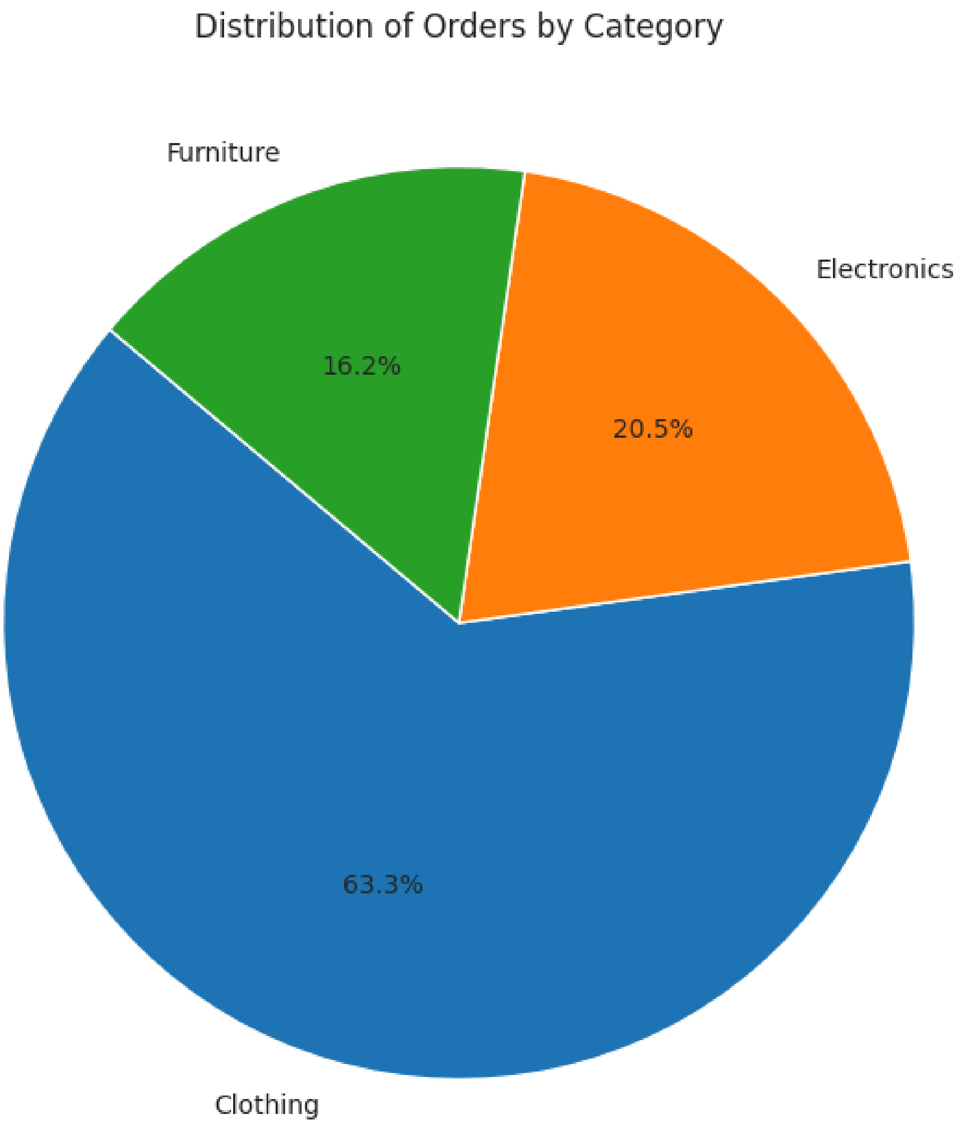


In [20]:
```
# Cek Outlier Variabel Numerik
sns.boxplot(data=order)
plt.show()
```

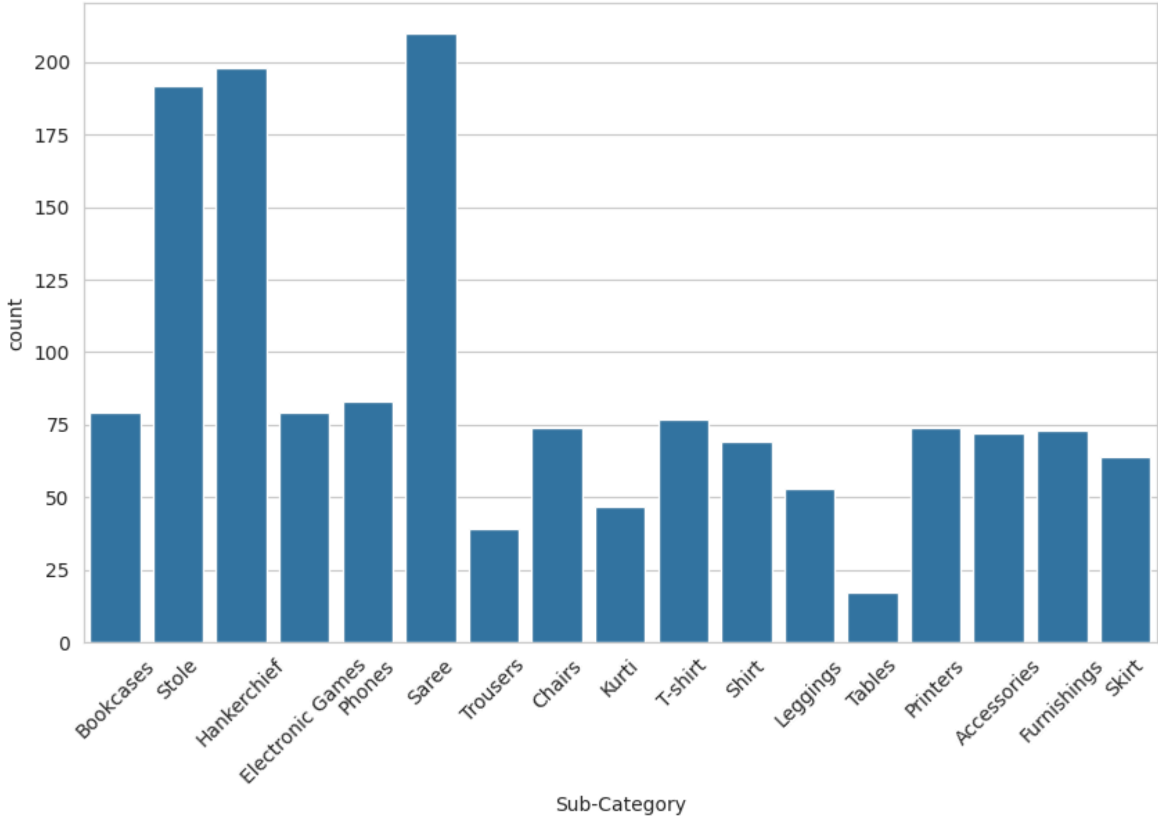


Dalam penelitian ini, tidak akan dilakukan penghapusan outlier pada data numerik karena data tersebut kemungkinan memiliki informasi mengenai pola atau asosiasi yang bermakna, sehingga dapat berpengaruh terhadap hasil analisis.

In [21]:
```
# Distribusi Order berdasarkan Kategori
category_counts = order['Category'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(category_counts, labels=category_counts.index, autopct='%1.1f%%', starta
plt.title('Distribution of Orders by Category')
plt.show()
```

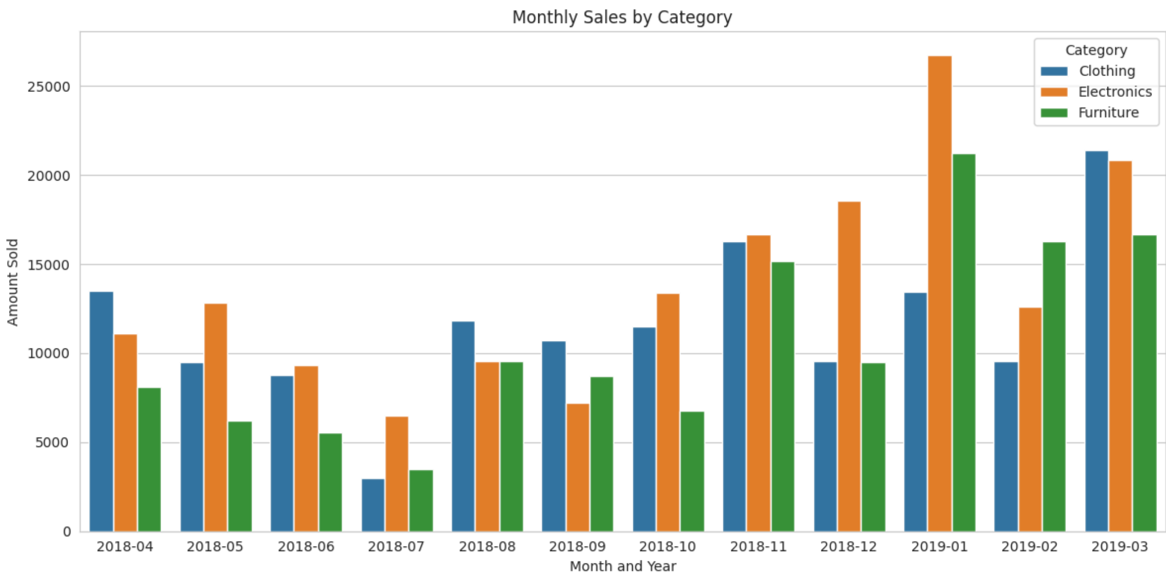


In [22]:
```
# Count plot untuk Variabel Kategorikal
plt.figure(figsize=(10, 6))
sns.countplot(data=order, x='Sub-Category')
plt.xticks(rotation=45)
plt.show()
```



In [23]:

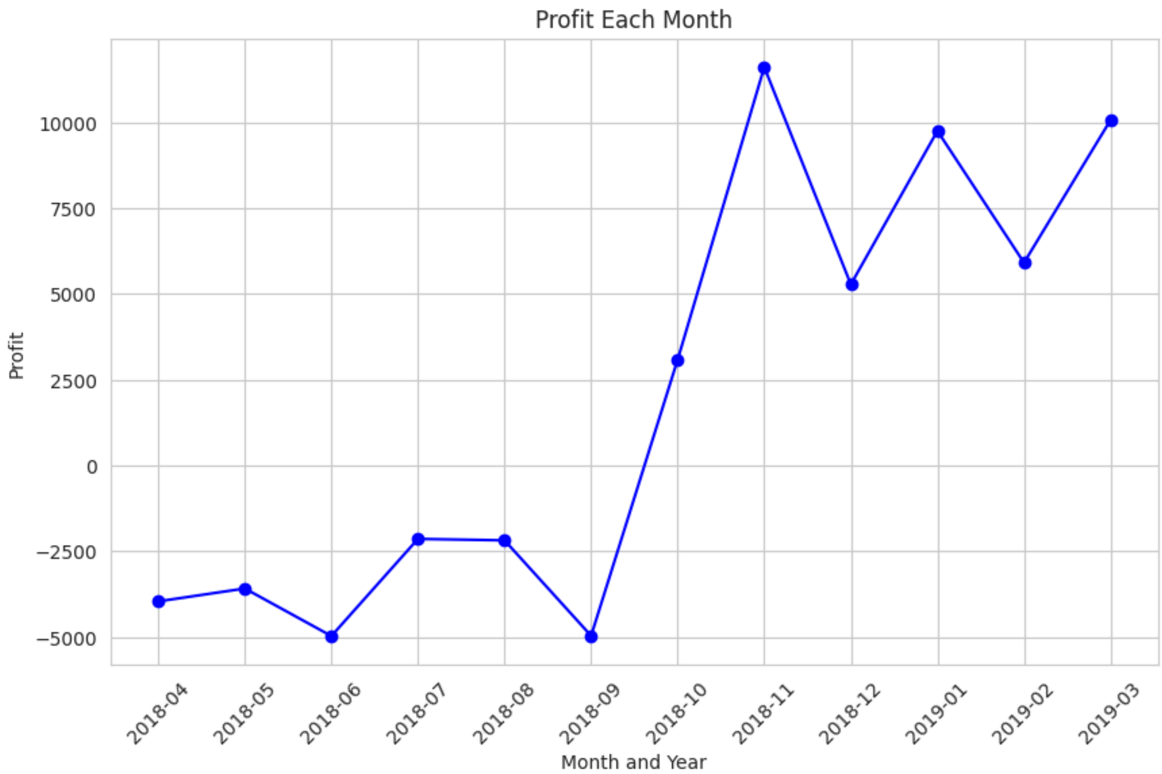
```
# Bar Plot Penjualan Bulanan berdasarkan Kategori
order['month_year'] = order['Order Date'].dt.strftime('%Y-%m')
monthly_sales = order.groupby(['month_year', 'Category'])['Amount'].sum().reset_
plt.figure(figsize=(12, 6))
sns.barplot(data=monthly_sales, x='month_year', y='Amount', hue='Category')
plt.title('Monthly Sales by Category')
plt.xlabel('Month and Year')
plt.ylabel('Amount Sold')
plt.legend(title='Category')
plt.tight_layout()
plt.show()
```



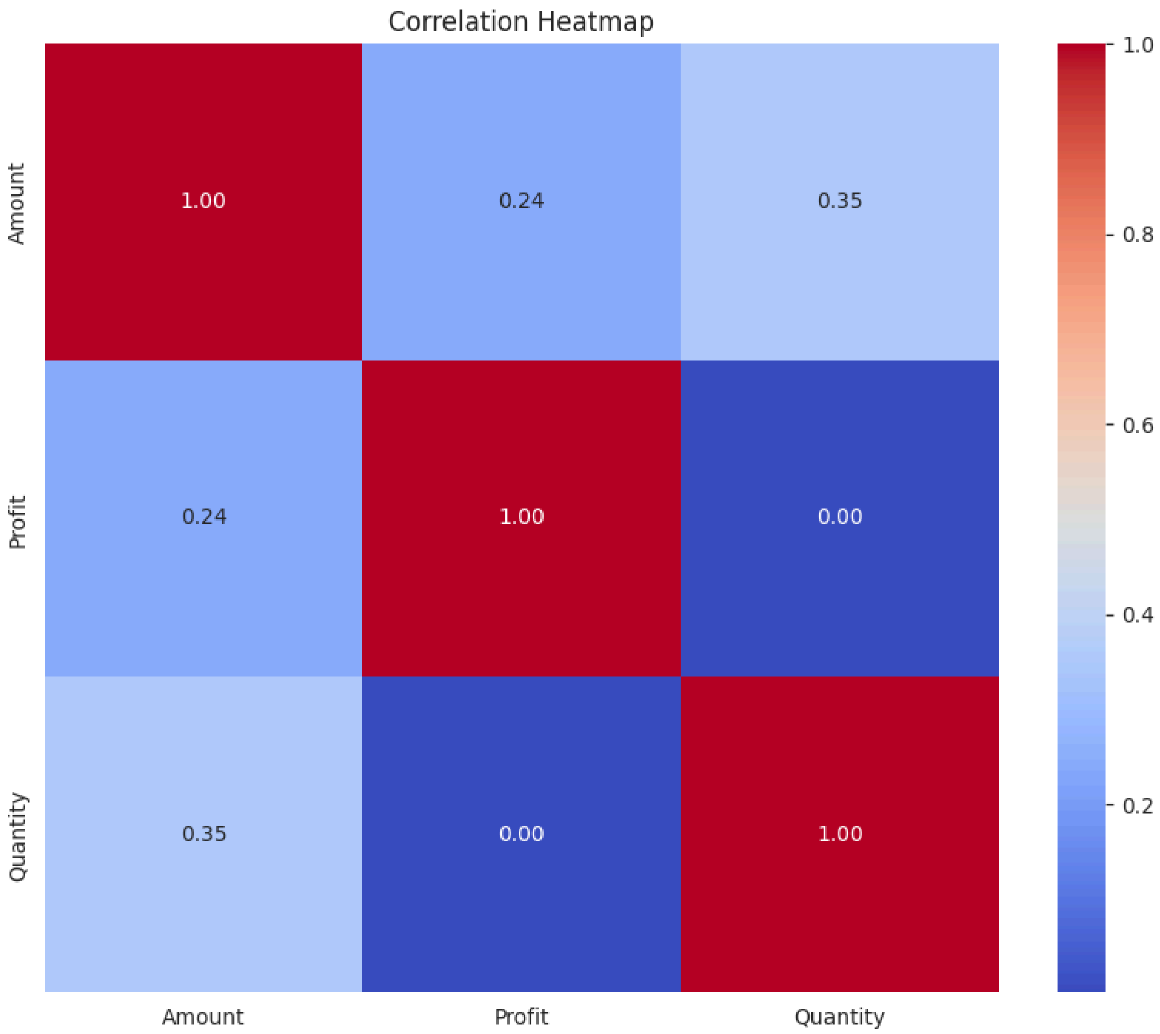
In [24]:

```
# Line Chart Profit setiap Bulan
monthly_profit = order.groupby('month_year')['Profit'].sum().reset_index()
plt.figure(figsize=(10, 6))
plt.plot(monthly_profit['month_year'], monthly_profit['Profit'], marker='o', col
```

```
plt.title('Profit Each Month')
plt.xlabel('Month and Year')
plt.ylabel('Profit')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```



In [25]:
```
# Heatmap Korelasi
numeric=order[['Amount','Profit','Quantity']]
plt.figure(figsize=(10, 8))
sns.heatmap(numeric.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```



- Kolom Quantity dengan Amount memiliki korelasi positif yang lemah. Hal ini dibuktikan dengan nilai korelasi sebesar 0.35, yang lebih mendekati 0 dibanding 1.
- Kolom Profit dengan Amount memiliki korelasi positif yang lemah karena memiliki nilai korelasi sebesar 0.24.

Rekayasa Fitur

In [26]:
```
# Transformasi data ke dalam format True/False
grouped = order.groupby('Order ID')['Sub-Category'].apply(list)
transactions = grouped.tolist()
unique_items = list(set(item for sublist in transactions for item in sublist))
df = pd.DataFrame(index=range(len(transactions)), columns=unique_items)
for i, transaction in enumerate(transactions):
    for item in transaction:
        df.at[i, item] = True
df = df.fillna(False)
df.insert(0, 'Transaction_ID', range(len(df)))
df.head()
```

Out[26]:

	Transaction_ID	Tables	Bookcases	Electronic Games	Trousers	Printers	Shirt	Furnishings
0	0	False	True	True	False	False	False	False
1	1	False	False	False	False	False	False	False
2	2	False	False	False	True	False	False	False
3	3	False	False	False	False	False	False	False
4	4	False	False	False	False	False	False	False

Pemodelan Data dan Evaluasi

Pembanding 1 (Min. Support 0.01)

In [27]:

```
# Pemodelan Apriori Pembanding 1
frequent_itemsets1 = apriori(df.drop(columns=['Transaction_ID']), min_support=0.
frequent_itemsets1['length'] = frequent_itemsets1['itemsets']. apply(lambda x: l
frequent_itemsets1
```

Out[27]:

	support	itemsets	length
0	0.032	(Tables)	1
1	0.144	(Bookcases)	1
2	0.146	(Electronic Games)	1
3	0.074	(Trousers)	1
4	0.134	(Printers)	1
...	...	...	...
329	0.010	(Saree, Stole, Accessories, Phones)	4
330	0.010	(T-shirt, Saree, Stole, Phones)	4
331	0.010	(Hankerchief, Saree, Stole, Phones)	4
332	0.010	(T-shirt, Hankerchief, Stole, Accessories)	4
333	0.012	(Hankerchief, Stole, Accessories, Phones)	4

334 rows × 3 columns

In [28]:

```
# Pembatasan Panjang Itemset
frequent_itemsets1 = frequent_itemsets1[(frequent_itemsets1['length'] <= 3)]
frequent_itemsets1
```

Out[28]:

	support	itemsets	length
0	0.032	(Tables)	1
1	0.144	(Bookcases)	1
2	0.146	(Electronic Games)	1
3	0.074	(Trousers)	1
4	0.134	(Printers)	1
...	...	...	...
288	0.018	(Hankerchief, Kurti, Stole)	3
289	0.012	(Hankerchief, Kurti, Phones)	3
290	0.014	(Hankerchief, Stole, Skirt)	3
291	0.016	(Hankerchief, Stole, Phones)	3
292	0.010	(Kurti, Stole, Phones)	3

293 rows × 3 columns

In [29]:

```
# Aturan Asosiasi Pembanding 1
rules1 = association_rules(frequent_itemsets1, metric='confidence', min_threshol
rules_sorted1 = rules1.sort_values(by='confidence', ascending=False)
rules_filtered1 = rules_sorted1.loc[:, 'antecedents':'lift']
rules_filtered1
```

Out[29]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
3	(Furnishings, Kurti)	(Hankerchief)	0.012	0.276	0.010	0.833333	3.019324
2	(T-shirt, Shirt)	(Stole)	0.022	0.314	0.018	0.818182	2.605675
1	(T-shirt, Electronic Games)	(Stole)	0.018	0.314	0.014	0.777778	2.476999
0	(Bookcases, Skirt)	(Saree)	0.016	0.312	0.012	0.750000	2.403846
4	(Accessories, Chairs)	(Saree)	0.016	0.312	0.012	0.750000	2.403846

Pembanding 2 (Min. Support 0.07)

In [30]:

```
# Pemodelan Data Pembanding 2
frequent_itemsets2 = apriori(df.drop(columns=['Transaction_ID']), min_support=0.
frequent_itemsets2['length'] = frequent_itemsets2['itemsets']. apply(lambda x: l
frequent_itemsets2
```

Out[30]:

	support	itemsets	length
0	0.144	(Bookcases)	1
1	0.146	(Electronic Games)	1
2	0.074	(Trousers)	1
3	0.134	(Printers)	1
4	0.132	(Shirt)	1
5	0.132	(Furnishings)	1
6	0.312	(Saree)	1
7	0.098	(Leggings)	1
8	0.128	(Chairs)	1
9	0.130	(Accessories)	1
10	0.140	(T-shirt)	1
11	0.276	(Hankerchief)	1
12	0.082	(Kurti)	1
13	0.112	(Skirt)	1
14	0.314	(Stole)	1
15	0.142	(Phones)	1
16	0.092	(Hankerchief, Saree)	2
17	0.116	(Saree, Stole)	2
18	0.118	(Hankerchief, Stole)	2

In [31]:

```
# Aturan Asosiasi Pembanding 2
rules2 = association_rules(frequent_itemsets2, metric='confidence', min_threshol
rules_sorted2 = rules2.sort_values(by='confidence', ascending=False)
rules_filtered2 = rules_sorted2.loc[:, 'antecedents':'lift']
rules_filtered2
```

Out[31]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
4	(Hankerchief)	(Stole)	0.276	0.314	0.118	0.427536	1.361580
5	(Stole)	(Hankerchief)	0.314	0.276	0.118	0.375796	1.361580
2	(Saree)	(Stole)	0.312	0.314	0.116	0.371795	1.184060
3	(Stole)	(Saree)	0.314	0.312	0.116	0.369427	1.184060
0	(Hankerchief)	(Saree)	0.276	0.312	0.092	0.333333	1.068376
1	(Saree)	(Hankerchief)	0.312	0.276	0.092	0.294872	1.068376

CONCLUSION

Berdasarkan kedua aturan asosiasi yang telah dilakukan, terlihat bahwa aturan asosiasi pertama (minimum support 0.01) memiliki nilai minimum confidence yang lebih tinggi, serta support yang masih cukup tinggi dengan minimum support 0.01. Aturan asosiasi ini menunjukkan hubungan yang kuat antara barang-barang yang dibeli bersama dalam transaksi, dengan confidence di atas 75% dan lift yang tinggi, menunjukkan relevansi yang signifikan. Sementara itu, aturan asosiasi kedua (minimum support 0.07) menggunakan nilai minimum confidence yang lebih rendah, dan meskipun support yang lebih tinggi dengan minimum support 0.05, confidence dan lift yang dihasilkan cenderung lebih rendah. Hal ini menunjukkan bahwa aturan asosiasi pembanding kedua kurang kuat dan memiliki dampak yang lebih rendah dalam memprediksi pola pembelian pelanggan. Pilihan minimum support 0.01 untuk aturan asosiasi pertama digunakan untuk menemukan hubungan yang kuat dan spesifik antara sub-kategori produk, sementara minimum support 0.07 digunakan pada aturan asosiasi kedua untuk melihat pola pembelian yang lebih umum dan luas.

In [31]:

```

```