In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
#The first file - Sales
path_file='C://Users//MAfzalinejad//OneDrive - Bixlers//Desktop//Mojgan//Sample Projec
Sales=pd.read_excel(path_file)
```

In [3]:
```
print("the first excel file records:"+str(len(Sales)))
```

```
the first excel file records:49153
```

In [4]:
```
#The Second file - Inventory
path_file='C://Users//MAfzalinejad//OneDrive - Bixlers//Desktop//Mojgan//Sample Projec
Inventory=pd.read_excel(path_file)
```

In [7]:
```
Sales.tail(2)
```

Out[7]:

	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Item Department	Unit Cost	Transaction Amount	Item No.	DocType	I
49151	2022-10-29	October	2022	25	90.0	0.0	30.0	MREP3008	LAF	
49152	2022-10-29	October	2022	25	90.0	0.0	25.0	MREP3008	SLC	

2 rows × 54 columns

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

Out[8]:

	SKU	Location	Total OH-Inventory
7403	SYPCT00002	25	3
7404	SYPCT00001	25	3

In [9]:
```
type('Location')
```

Out[9]:
```
str
```

In [10]:
```
Inventory['Location']= Inventory['Location'].astype(str)
Inventory['SKU']= Inventory['SKU'].str.lstrip()
Inventory ['Items-Stores']=Inventory['SKU']+'-'+Inventory['Location']
#Checking Duplication
Inventory[Inventory.duplicated(subset=['Items-Stores'], keep=False)]
```

Out[10]:

	SKU	Location	Total OH-Inventory	Items-Stores

In [11]:
```
# Data cleaning - Sales
```

In [12]:
```
Sales['Location Number'].unique()
```

Out[12]:
```
array([24, 'Burlington', 2, 25, 12, 99], dtype=object)
```

In [13]:
```
Sales['Location Number']=Sales['Location Number'].replace('Burlington',24)
```

In [14]:
```
Sales['Location Number'].unique()
```

Out[14]:
```
array([24,  2, 25, 12, 99], dtype=int64)
```

In [15]:
```
# Handling Missing Value and drop unused columns
```

In [16]:
```
Sales.isna().sum()
```

Out[16]:

OrderDate	0
RETAIL MONTH	0
RETAIL Year	0
Location Number	0
Item Department	29648
Unit Cost	29276
Transaction Amount	29245
Item No.	0
DocType	29648
ItemPieces	29648
QTY	0
TOTAL SALES	0
$ SALES	4015
QTY SALES	2337
$ RETURNS	46962
QTY RETURNS	46929
TOTAL COSTS	20677
Item Department Description	27873
ITEM CATEGORY	0
Group Code	27548
Group Code Description	27549
Vendor Number	29484
VENDOR CATEGORY	0
VENDOR GROUP	0
BANNER	0
VendorsItemNum	7458
Vendor Name	18387
Description	215
Week	0
ITEM GROUP	0
Type of Transaction	0
CALENDAR MONTH	0
CALENDAR YEAR	0
Quarter	0
UNIT PRICE	3
Net Sales for AUR	1824
Net QTY for AUR	2337
Retail Month #	0
Month + Year	0
Billboard	49144
Batch Number	29648
Ticket Number	0
Ring Type	40778
Serial Number	44937
Certificate	48702
Concatenation Sales	0
Stors Name.Location Name	0
LAB & Natural Stones	0
Metal Type	32505
kt	14036
Day of Year	0
Full Name - Customer	20302
Customer ID - LS	20236
Customer ID - Craftsman	49153

dtype: int64

In [17]:
```
Sales=Sales.drop(columns=['ItemPieces','Item Department','DocType','Item Department De
```

In [18]:
```
Sales.tail(2)
```

Out[18]:

	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Unit Cost	Transaction Amount	Item No.	QTY	TOTAL SALES	$ SALES	...
49151	2022-10-29	October	2022	25	0.0	30.0	MREP3008	1	30.0	30.0	...
49152	2022-10-29	October	2022	25	0.0	25.0	MREP3008	1	25.0	25.0	...

2 rows × 42 columns

In [19]:
```
# Trim & Standardize Text Fields
Sales['Location Number']= Sales['Location Number'].astype(str)
```

In [20]:
```
Sales['Item No.']= Sales['Item No.'].str.lstrip()
```

In [21]:
```
Sales ['Items-Stores']=Sales['Item No.']+'-'+Sales['Location Number']
```

In [22]:
```
Sales.tail(2)
```

Out[22]:

	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Unit Cost	Transaction Amount	Item No.	QTY	TOTAL SALES	$ SALES	...
49151	2022-10-29	October	2022	25	0.0	30.0	MREP3008	1	30.0	30.0	...
49152	2022-10-29	October	2022	25	0.0	25.0	MREP3008	1	25.0	25.0	...

2 rows × 43 columns

In [23]:
```
# Join- Adding Inventory data
Sales = pd.merge(Sales, Inventory[['Items-Stores']], on='Items-Stores', how='left')
```

In [24]:
```
Sales.tail(2)
```

Out[24]:

	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Unit Cost	Transaction Amount	Item No.	QTY	TOTAL SALES	$ SALES	...
49151	2022-10-29	October	2022	25	0.0	30.0	MREP3008	1	30.0	30.0	...
49152	2022-10-29	October	2022	25	0.0	25.0	MREP3008	1	25.0	25.0	...

2 rows × 43 columns

In [25]:
```
(Sales['$ SALES']>=0).all()
```

Out[25]:
```
False
```

In [26]:
```
# Validate data
```

In [27]:
```
Sales ['$ RETURNS']=Sales ['$ RETURNS'].fillna(0)+Sales['$ SALES'].where(Sales ['$ SAL
```

In [28]:
```
Sales['$ SALES']=Sales['$ SALES'].where(Sales['$ SALES']>0,0)
```

In [29]:
```
(Sales['$ SALES']>=0).all()
```

Out[29]:
```
True
```

In [30]:
```
type('OrderDate')
```

Out[30]:
```
str
```

In [31]:
```
#Outlier Detection
```

In [32]:
```
Q1 = Sales['TOTAL SALES'].quantile(.25)
Q3 = Sales['TOTAL SALES'].quantile(.75)
```

In [33]:
```
# Total Analysis
Total_Sales = Sales['TOTAL SALES'].sum()
Sales_2023 = Sales[Sales['RETAIL Year']==2023]
Total_Sales_2023= Sales_2023['TOTAL SALES'].sum()
```

In [34]:
```
Sales_2022 = Sales[Sales['RETAIL Year']==2022]
Total_Sales_2022= Sales_2022['TOTAL SALES'].sum()
```

In [35]:
```
Sales_2023.tail(2)
```

Out[35]:

	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Unit Cost	Transaction Amount	Item No.	QTY	TOTAL SALES	$ SALES
46051	2024-01-15	January	2023	24	NaN	NaN	WRXG0454	1	11450.0	11450.0
46052	2024-01-14	January	2023	25	NaN	NaN	DPQTF0104	1	825.0	825.0

2 rows × 43 columns

In [36]:
```
Sales_2022.tail(2)
```

Out[36]:

	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Unit Cost	Transaction Amount	Item No.	QTY	TOTAL SALES	$ SALES	...
49151	2022-10-29	October	2022	25	0.0	30.0	MREP3008	1	30.0	30.0	...
49152	2022-10-29	October	2022	25	0.0	25.0	MREP3008	1	25.0	25.0	...

2 rows × 43 columns

In [37]:
```
Total_Sales_2023
```

Out[37]:
```
19146997.83
```

In [38]:
```
Sales_2023_per_store = Sales_2023.groupby('Location Number')['TOTAL SALES'].sum().rese
print(Sales_2023_per_store)
```

```
  Location Number  TOTAL SALES
0              12    947210.79
1               2   1698721.26
2              24  12754571.88
3              25   3738967.80
4              99      7526.10
```

In [39]:
```
Sales_2022_per_store = Sales_2023.groupby('Location Number')['TOTAL SALES'].sum().rese
print(Sales_2023_per_store)
```

```
  Location Number  TOTAL SALES
0              12    947210.79
1               2   1698721.26
2              24  12754571.88
3              25   3738967.80
4              99      7526.10
```

In [40]:
```
type('OrderDate')
```

Out[40]:
```
str
```

In [41]:
```
Sales_2023.tail(2)
```

Out[41]:

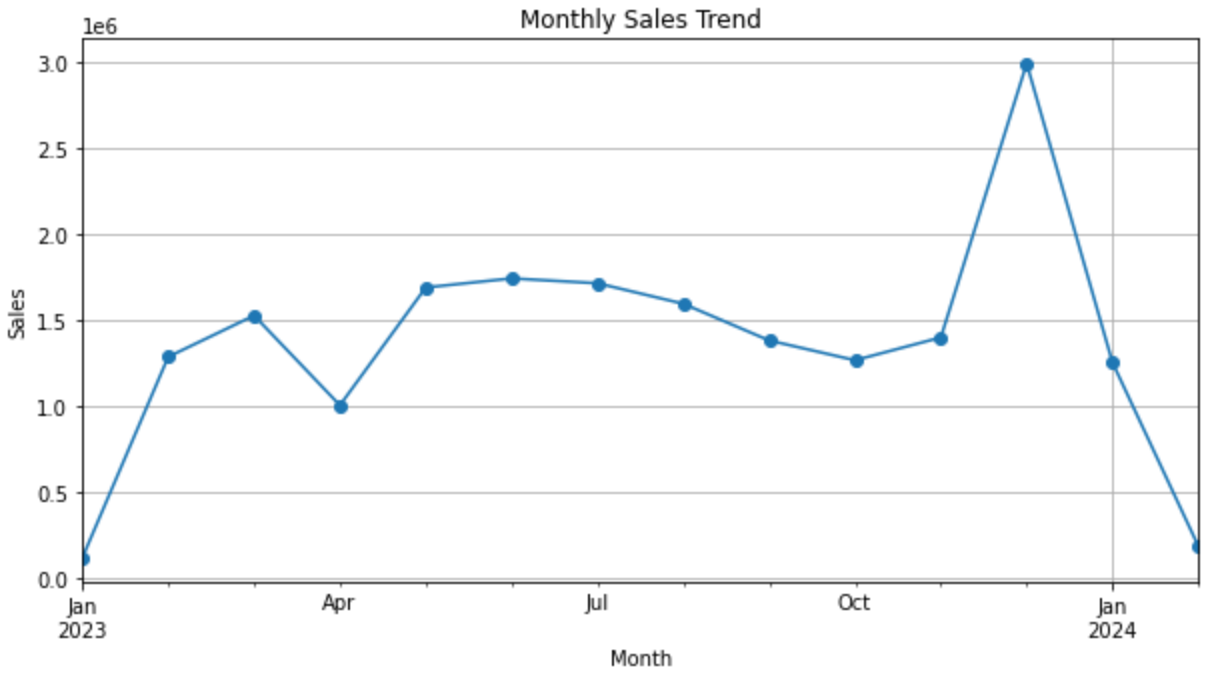
	OrderDate	RETAIL MONTH	RETAIL Year	Location Number	Unit Cost	Transaction Amount	Item No.	QTY	TOTAL SALES	$ SALES
46051	2024-01-15	January	2023	24	NaN	NaN	WRXG0454	1	11450.0	11450.0
46052	2024-01-14	January	2023	25	NaN	NaN	DPQTF0104	1	825.0	825.0

2 rows × 43 columns

In [42]:
```
import matplotlib.pyplot as plt
```

In [43]:
```
#Trend Analysis
monthly_sales_2023 = Sales_2023.groupby(Sales_2023['OrderDate'].dt.to_period('M'))['TO
```

In [44]:
```
monthly_sales_2023.plot(kind='line', marker='o', figsize=(10,5))
plt.title('Monthly Sales Trend')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.grid(True)
plt.show()
```

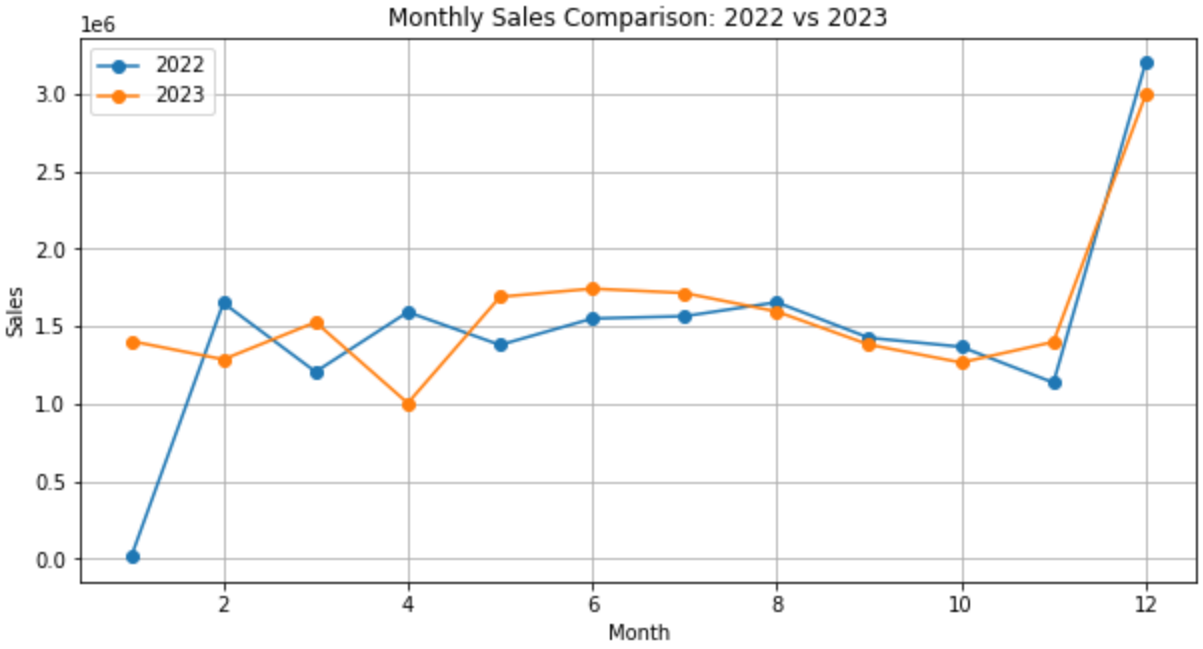


In [45]:
```
# Comparison Sales 2022 vs 2023
Sales_filtered = Sales[Sales['OrderDate'].dt.year.isin([2022, 2023])].copy()

Sales_filtered['Year']=Sales_filtered['OrderDate'].dt.year
Sales_filtered['Month']=Sales_filtered['OrderDate'].dt.month
Monthly_Sales= Sales_filtered.groupby(['Year', 'Month'])['TOTAL SALES'].sum().reset_ir
```

In [46]:
```
#Plot
plt.figure(figsize=(10,5))
for year in [2022, 2023]:
    data = Monthly_Sales[Monthly_Sales['Year'] == year]
    plt.plot(data['Month'], data['TOTAL SALES'], marker='o', label=str(year))

plt.title('Monthly Sales Comparison: 2022 vs 2023')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.legend()
plt.grid(True)
plt.show()
```



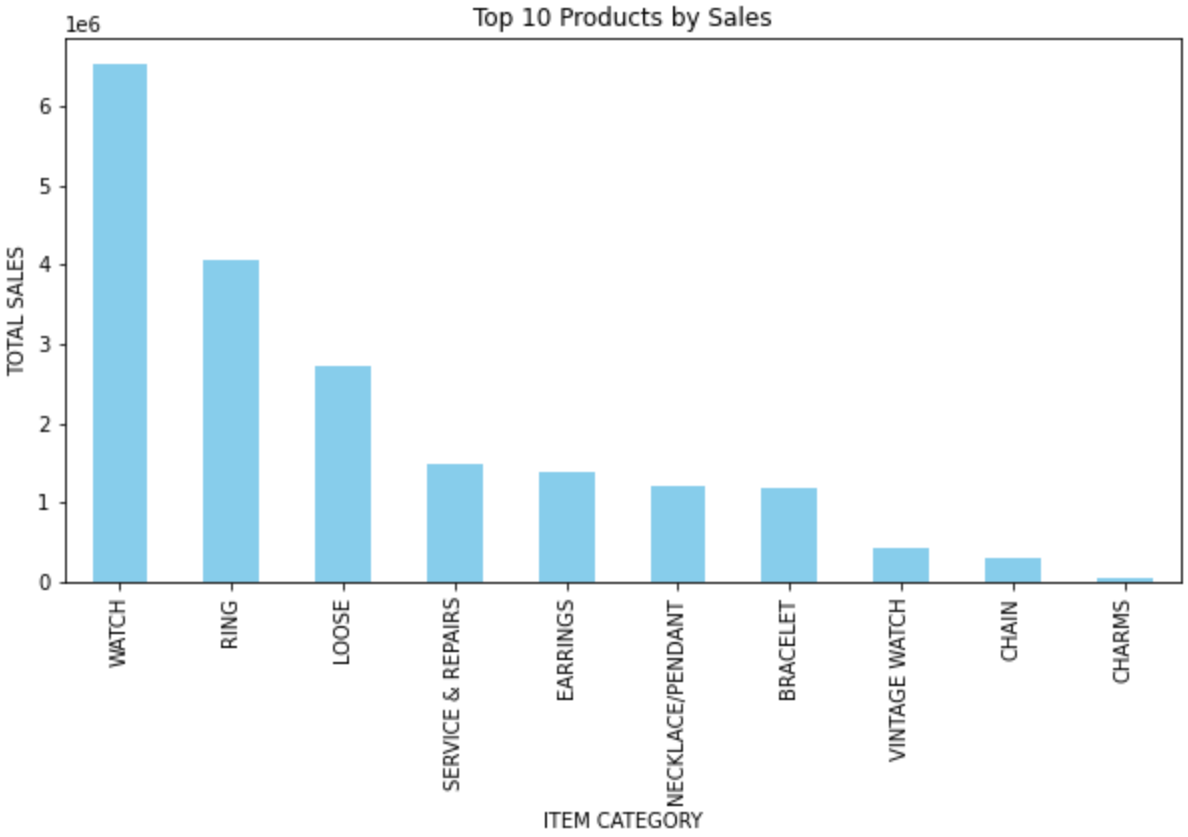
In [47]:
```
#Top Products - 2023

top_products = Sales_2023.groupby('ITEM CATEGORY')['TOTAL SALES'].sum().sort_values(as

print(top_products)
```

```
ITEM CATEGORY
WATCH                6523002.05
RING                 4045626.89
LOOSE                2708346.25
SERVICE & REPAIRS    1478759.38
EARRINGS             1376231.66
NECKLACE/PENDANT     1196093.52
BRACELET             1189104.04
VINTAGE WATCH         420871.00
CHAIN                 295422.06
CHARMS                 38522.36
Name: TOTAL SALES, dtype: float64
```

In [48]:
```
# Plot Top Products
top_products.plot(kind='bar', figsize=(10,5), color='skyblue')
plt.title('Top 10 Products by Sales')
plt.ylabel('TOTAL SALES')
plt.show()
```



In [66]:
```
# Growth Rate / Trends
Sales_2023['OrderDate'] = pd.to_datetime(Sales_2023['OrderDate'], errors='coerce')
monthly_sales_2023 = Sales_2023.groupby(Sales_2023['OrderDate'].dt.to_period('M'))['TO

monthly_sales_2023['TOTAL SALES'] = monthly_sales_2023['TOTAL SALES'].astype(float)
monthly_sales_2023['Growth_Rate'] = monthly_sales_2023['TOTAL SALES'].pct_change() * 1
monthly_sales_2023['Growth_Rate'] = monthly_sales_2023['Growth_Rate'].fillna(0)
monthly_sales_2023['Growth_Rate']= monthly_sales_2023['Growth_Rate'].apply(lambda x: f

monthly_sales_2023
```

```
C:\Users\MAfzalinejad\AppData\Local\Temp\ipykernel_5668\2944234870.py:2: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  Sales_2023['OrderDate'] = pd.to_datetime(Sales_2023['OrderDate'], errors='coerce')
```

Out[66]:

	OrderDate	TOTAL SALES	Growth_Rate
0	2023-01	117032.36	0.00%
1	2023-02	1285807.54	998.68%
2	2023-03	1526093.84	18.69%
3	2023-04	1003948.96	-34.21%
4	2023-05	1689284.59	68.26%
5	2023-06	1742203.86	3.13%
6	2023-07	1713943.73	-1.62%
7	2023-08	1594642.29	-6.96%
8	2023-09	1381006.81	-13.40%
9	2023-10	1265901.67	-8.33%
10	2023-11	1398643.64	10.49%
11	2023-12	2992586.90	113.96%
12	2024-01	1251914.36	-58.17%
13	2024-02	183987.28	-85.30%

In [61]:
```
#Pivot Table Style Insights
pivot_Sales_2023 = Sales_2023.pivot_table(values='TOTAL SALES', index='Location Number
pivot_Sales_2023
```

Out[61]:

ITEM CATEGORY	ACCESSORIES	BRACELET	CHAIN	CHARMS	CUFF LINK	CUSTOM	EARRINGS	EXCLUDE	FI
Location Number									
12	515	62337.99	24099.05	3929.15	44.5	0.00	117151.50	21.98	
2	1225	103391.10	45785.17	4720.37	119.0	0.00	210471.05	0.00	
24	600	816127.87	169680.12	21896.39	2100.0	4157.18	751304.59	0.00	
25	0	207247.08	55857.72	7976.45	660.0	0.00	296790.50	0.00	
99	0	0.00	0.00	0.00	0.0	0.00	514.02	0.00	

5 rows × 26 columns

In [ ]: