```
# importing libraries
try:
    import os,sys
    import numpy as np
    import pandas as pd
except Exception as e:
    print("Error due to",e)
file_path = os.getcwd()


#Checking current working directory
print(os.getcwd())
```

```
C:\Users\murth\Desktop\DkIT\Sem2\Data_Visualisation_Insight\Assignments\Assignment1\M
```

```
# reading input dataset for orders
orders_df = pd.read_excel(file_path+"//Superstore_Sales.xlsx", sheet_name=0)
orders_df.head()
```

| | Order ID | Order Date | Ship Date | Ship Mode | Customer Name | Segment | Location | Market | Region | F |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | CA-2012-124891 | 2012-07-31 | 2012-07-31 | Same Day | Rick Hansen | Consumer | New York, United States | US | East | 10 |
| 1 | IN-2013-77878 | 2013-02-05 | 2013-02-07 | Second Class | Justin Ritter | Corporate | New South Wales, Australia | APAC | Oceania | 10 |
| 2 | IN-2013-71249 | 2013-10-17 | 2013-10-18 | First Class | Craig Reiter | Consumer | Queensland, Australia | APAC | Oceania | 10 |
| 3 | ES-2013-1579342 | 2013-01-28 | 2013-01-30 | First Class | Katherine Murray | Home Office | Berlin, Germany | EU | Central | 10 |
| 4 | SG-2013-4320 | 2013-11-05 | 2013-11-06 | Same Day | Rick Hansen | Consumer | Dakar, Senegal | Africa | Africa | TI 10 |

```
orders_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
```

```
Data columns (total 19 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Order ID        51290 non-null  object
 1   Order Date      51290 non-null  datetime64[ns]
 2   Ship Date       51290 non-null  datetime64[ns]
 3   Ship Mode       51290 non-null  object
 4   Customer Name   51290 non-null  object
 5   Segment         51290 non-null  object
 6   Location        51290 non-null  object
 7   Market          51219 non-null  object
 8   Region          51226 non-null  object
 9   Product ID      51290 non-null  object
 10  Category        51236 non-null  object
 11  Sub-Category    51290 non-null  object
 12  Product Name    51290 non-null  object
 13  Sales           51290 non-null  float64
 14  Quantity        51290 non-null  int64
 15  Discount        51290 non-null  float64
 16  Profit          51290 non-null  float64
 17  Shipping Cost   51290 non-null  float64
 18  Order Priority  51290 non-null  object
dtypes: datetime64[ns](2), float64(4), int64(1), object(12)
memory usage: 7.4+ MB
```

```
# reading input dataset for order returns
returns_df = pd.read_excel(file_path+"//Superstore_Sales.xlsx", sheet_name=1)
returns_df.head()
```

```
returns_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1173 entries, 0 to 1172
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Returned  1173 non-null   object
 1   Order ID  1173 non-null   object
 2   Market    1173 non-null   object
dtypes: object(3)
memory usage: 27.6+ KB
```

```
# reading input dataset for head of regions
heads_df = pd.read_excel(file_path+"//Superstore_Sales.xlsx", sheet_name=2)
heads_df.head()
```

```
heads_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13 entries, 0 to 12
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Person  13 non-null     object
 1   Region  13 non-null     object
dtypes: object(2)
memory usage: 336.0+ bytes
```

```
# changing column names to avoid errors in future operations and processing

# orders df
orders_df.rename(columns={'Order ID':'order_id', 'Order Date':'order_date', 'Ship Date':'ship_date', 'Shi
                          'Customer Name':'customer_name', 'Segment':'segment', 'Location':'location', 'M
                          'Region':'region', 'Product ID':'product_id', 'Category':'category', 'Sub-Categ
                          'Product Name':'product_name', 'Sales':'sales', 'Quantity':'quantity', 'Discoun
                          'Profit':'profit', 'Shipping Cost':'shipping_cost', 'Order Priority':'order_pri

# returns df
returns_df.rename(columns={'Returned':'returned', 'Order ID':'order_id', 'Market':'market'}, inplace=True

# heads df
heads_df.rename(columns={'Person':'person', 'Region':'region'},inplace=True)
```

```
orders_df.head()
```

```
returns_df.head()
```

```
heads_df.head()
```

```
# checking null values in data set
print(orders_df.isnull().sum())
print(100*"*")
print(returns_df.isnull().sum())
print(100*"*")
print(heads_df.isnull().sum())
```

```
order_id           0
order_date         0
ship_date          0
ship_mode          0
customer_name      0
segment            0
location           0
market            71
region            64
product_id         0
category          54
sub_category       0
```

```
product_name       0
sales              0
quantity           0
discount           0
profit             0
shipping_cost      0
order_priority     0
dtype: int64
**********************************************************************************
returned    0
order_id    0
market      0
dtype: int64
**********************************************************************************
person    0
region    0
dtype: int64
```

```
# checking Null values for market and region
orders_df[orders_df[['location','market','region']].isna().any(axis=1)]
```

```
# checking Null values for category
orders_df[orders_df[['category','sub_category']].isna().any(axis=1)]
```

```
# checking unique values for mapping
print(orders_df['market'].unique())
print(100*"*")
print(orders_df['region'].unique())
print(100*"*")
print(orders_df['category'].unique())
print(100*"*")
```

```
['US' 'APAC' 'EU' 'Africa' 'EMEA' 'LATAM' 'Canada' nan]
****************************************************************************************************
['East' 'Oceania' 'Central' 'Africa' 'West' 'South' 'Central Asia' 'EMEA'
 'North Asia' 'North' 'Caribbean' 'Southeast Asia' 'Canada' nan]
****************************************************************************************************
['Technology' 'Furniture' 'Office Supplies' nan]
****************************************************************************************************
```

```
# splitting city and country for location region
orders_df[['city', 'country']] = orders_df['location'].str.split(',', expand=True)
orders_df['country'] = orders_df['country'].str.strip()
orders_df.drop('location',axis=1,inplace=True)
orders_df.head()
```

```
orders_df.head()
```

```
# checking Null values for market and region
orders_df[orders_df[['city','country','market','region']].isna().any(axis=1)].head()
```

```
# checking for Newyork, United states
orders_df.loc[orders_df['country'] == 'United States']
```

```
print("empty values count for markets before", orders_df['market'].isnull().sum())
print("empty values count for region before", orders_df['region'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.market)) & (orders_df['country'] == 'United States'), 'market'] = 'US'
```

```
orders_df.loc[(pd.isnull(orders_df.region)) & (orders_df['country'] == 'United States') & (orders_df['cit
print("empty values count for markets after", orders_df['market'].isnull().sum())
print("empty values count for region after", orders_df['region'].isnull().sum())
```

```
empty values count for markets before 71
empty values count for region before 64
empty values count for markets after 44
empty values count for region after 37
```

```
orders_df[orders_df[['city','country','market']].isna().any(axis=1)].head()
```

```
# checking for Ar Riyad, Saudi Arabia
orders_df.loc[orders_df['country'] == 'Saudi Arabia']
```

```
print("empty values count for markets before", orders_df['market'].isnull().sum())
print("empty values count for regions before", orders_df['region'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.market)) & (orders_df['country'] == 'Saudi Arabia'), 'market'] = 'EMEA
orders_df.loc[(pd.isnull(orders_df.region)) & (orders_df['country'] == 'Saudi Arabia'), 'region'] = 'EMEA
print("empty values count for markets after", orders_df['market'].isnull().sum())
print("empty values count for regions after", orders_df['region'].isnull().sum())
```

```
empty values count for markets before 44
empty values count for regions before 37
empty values count for markets after 25
empty values count for regions after 25
```

```
orders_df[orders_df[['city','country','market']].isna().any(axis=1)].head()
```

```
# checking for Western Australia
orders_df.loc[orders_df['country'] == 'Australia']
```

```
print("empty values count for markets before", orders_df['market'].isnull().sum())
print("empty values count for regions before", orders_df['region'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.market)) & (orders_df['country'] == 'Australia'), 'market'] = 'APAC'
orders_df.loc[(pd.isnull(orders_df.region)) & (orders_df['country'] == 'Australia'), 'region'] = 'Oceania
print("empty values count for markets after", orders_df['market'].isnull().sum())
print("empty values count for regions after", orders_df['region'].isnull().sum())
```

```
empty values count for markets before 25
empty values count for regions before 25
empty values count for markets after 0
empty values count for regions after 0
```

```
orders_df[orders_df[['category','sub_category']].isna().any(axis=1)].head()
```

```
# checking for sub_category chairs for category assignment
orders_df.loc[orders_df['sub_category'] == 'Chairs']
```

```
orders_df.loc[orders_df['sub_category'] == 'Chairs']['category'].unique()
```

```
array(['Furniture', nan], dtype=object)
```

```
print("empty values count for Categories before", orders_df['category'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.category)) & (orders_df['sub_category'] == 'Chairs'), 'category'] = 'F
print("empty values count for Categories before", orders_df['category'].isnull().sum())
```

```
empty values count for Categories before 54
empty values count for Categories before 44
```

```
orders_df[orders_df[['category','sub_category']].isna().any(axis=1)].head()
```

```
# checking for sub_category book cases for category assignment
orders_df.loc[orders_df['sub_category'] == 'Bookcases']
```

```
orders_df.loc[orders_df['sub_category'] == 'Bookcases']['category'].unique()
```

```
array([nan, 'Furniture'], dtype=object)
```

```
print("empty values count for Categories before", orders_df['category'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.category)) & (orders_df['sub_category'] == 'Bookcases'), 'category'] =
print("empty values count for Categories before", orders_df['category'].isnull().sum())
```

```
empty values count for Categories before 44
empty values count for Categories before 28
```

```
# checking for sub_category Accessories for category assignment
orders_df.loc[orders_df['sub_category'] == 'Accessories']
```

```
print("empty values count for Categories before", orders_df['category'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.category)) & (orders_df['sub_category'] == 'Accessories'), 'category']
print("empty values count for Categories before", orders_df['category'].isnull().sum())
```

```
empty values count for Categories before 28
empty values count for Categories before 13
```

```
orders_df[orders_df[['category','sub_category']].isna().any(axis=1)].head()
```

```
# checking for sub_category Accessories for category assignment
orders_df.loc[orders_df['sub_category'] == 'Fasteners']
```

```
orders_df.loc[orders_df['sub_category'] == 'Fasteners']['category'].unique()
```

```
array(['Office Supplies', nan], dtype=object)
```

```
print("empty values count for Categories before", orders_df['category'].isnull().sum())
orders_df.loc[(pd.isnull(orders_df.category)) & (orders_df['sub_category'] == 'Fasteners'), 'category'] =
print("empty values count for Categories before", orders_df['category'].isnull().sum())
```

```
empty values count for Categories before 13
empty values count for Categories before 0
```

```
# checking Null values
orders_df.isnull().sum()
```

```
order_id          0
order_date        0
ship_date         0
ship_mode         0
customer_name     0
segment           0
market            0
region            0
product_id        0
category          0
sub_category      0
product_name      0
sales             0
quantity          0
discount          0
profit            0
shipping_cost     0
order_priority    0
city              0
country           0
dtype: int64
```

```
# merging tables for return and heads of regions
orders_returns = orders_df.merge(returns_df,how='left',left_on=['order_id','market'],right_on=['order_id'
sales_final_df = orders_returns.merge(heads_df,how='left',left_on=['region'],right_on=['region'])
```

```
print(orders_df.columns)
```

```
print(returns_df.columns)
print(heads_df.columns)
```

```
Index(['order_id', 'order_date', 'ship_date', 'ship_mode', 'customer_name',
       'segment', 'market', 'region', 'product_id', 'category', 'sub_category',
       'product_name', 'sales', 'quantity', 'discount', 'profit',
       'shipping_cost', 'order_priority', 'city', 'country'],
      dtype='object')
Index(['returned', 'order_id', 'market'], dtype='object')
Index(['person', 'region'], dtype='object')
```

```
sales_final_df
```

```
# checking Null Values
sales_final_df.isnull().sum()
```

```
order_id              0
order_date            0
ship_date             0
ship_mode             0
customer_name         0
segment               0
market                0
```