Data Cleaning

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

- Read the datasets

In [ ]:
```
data = pd.read_csv("Financials.csv")
```

- Copy the original datasets

In [ ]:
```
df = data.copy()
```

- Read the Datasets

In [ ]:
```
df.head()
```

Out[ ]:

	Segment	Country	Product	Discount Band	Units Sold	Manufacturing Price	Sale Price	Gr Sa
0	Government	Canada	Carretera	None	$1,618.50	$3.00	$20.00	$32,370
1	Government	Germany	Carretera	None	$1,321.00	$3.00	$20.00	$26,420
2	Midmarket	France	Carretera	None	$2,178.00	$3.00	$15.00	$32,670
3	Midmarket	Germany	Carretera	None	$888.00	$3.00	$15.00	$13,320
4	Midmarket	Mexico	Carretera	None	$2,470.00	$3.00	$15.00	$37,050

The dataset contains various financial and sales-related information, including:

- Segment: the segment to which the sale belongs (e.g., Government, Midmarket)
- Country: the country where the sale occurred
- Product: the name of the product sold
- Discount Band: the discount level applied to the sale
- Units Sold: the number of units sold
- Manufacturing Price: the price at which the product was manufactured
- Sale Price: the price at which the product was sold
- Gross Sales: the total sales before discounts
- Discounts: the amount discounted from the gross sales
- Sales: the total sales after discounts
- COGS (Cost of Goods Sold): the cost to produce the goods sold
- Profit: the profit from the sale (Sales - COGS)

- Date: the date of the sale
- Month Number: the month of the sale (numerical)
- Month Name: the month of the sale (name)
- Year: the year of the sale

Data Processing

- Handling teh columns

In [ ]:
```
# Remove leading and trailing spaces from column names
df.columns = df.columns.str.strip()

# Display the updated column names
df.columns
```

Out[ ]:
```
Index(['Segment', 'Country', 'Product', 'Discount Band', 'Units Sold',
       'Manufacturing Price', 'Sale Price', 'Gross Sales', 'Discounts',
       'Sales', 'COGS', 'Profit', 'Date', 'Month Number', 'Month Name',
       'Year'],
      dtype='object')
```

In [ ]:
```
df.shape
```

Out[ ]:
```
(700, 16)
```

- Checking the null values

In [ ]:
```
df.isnull().sum()
```

Out[ ]:
```
Segment                0
Country                0
Product                0
Discount Band          0
Units Sold             0
Manufacturing Price    0
Sale Price             0
Gross Sales            0
Discounts              0
Sales                  0
COGS                   0
Profit                 0
Date                   0
Month Number           0
Month Name             0
Year                   0
dtype: int64
```

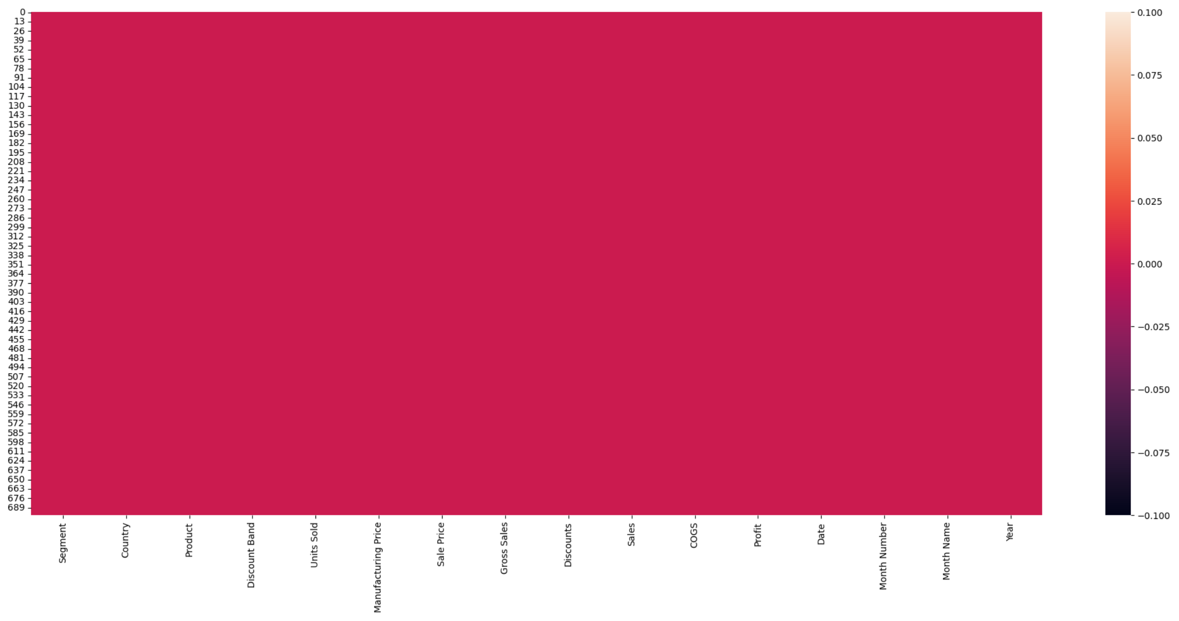
Import the visualisation libraries

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

In [ ]:
```python
plt.figure(figsize=(25,10))

sns.heatmap(df.isnull())
```

Out[ ]:
```
<Axes: >
```



- There are no missing values in the dataset, which is a good sign. The descriptive statistics provide some interesting insights:

- Checking the Data types for columns

In [ ]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 700 entries, 0 to 699
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Segment              700 non-null    object
 1   Country              700 non-null    object
 2   Product              700 non-null    object
 3   Discount Band        700 non-null    object
 4   Units Sold           700 non-null    object
 5   Manufacturing Price  700 non-null    object
 6   Sale Price           700 non-null    object
 7   Gross Sales          700 non-null    object
 8   Discounts            700 non-null    object
 9   Sales                700 non-null    object
 10  COGS                 700 non-null    object
 11  Profit               700 non-null    object
 12  Date                 700 non-null    object
 13  Month Number         700 non-null    int64 
 14  Month Name           700 non-null    object
 15  Year                 700 non-null    int64 
dtypes: int64(2), object(14)
memory usage: 87.6+ KB
```

Cleaning the unwated symbols from columns and replace that

In [ ]:
```
df['Units Sold'] = df['Units Sold'].str.replace('$','')
df['Units Sold'] = df['Units Sold'].str.replace('.','')
df['Units Sold'] = df['Units Sold'].str.replace(',','')
```

In [ ]:
```
df['Manufacturing Price'] = df['Manufacturing Price'].str.replace('$','')
df['Manufacturing Price'] = df['Manufacturing Price'].str.replace('.','')
```

In [ ]:
```
df['Sale Price'] = df['Sale Price'].str.replace('$','')
df['Sale Price'] = df['Sale Price'].str.replace('.','')
df['Sale Price'] = df['Sale Price'].str.replace(',','')
```

In [ ]:
```
df['Gross Sales'] = df['Gross Sales'].str.replace('$','')
df['Gross Sales'] = df['Gross Sales'].str.replace('.','')
df['Gross Sales'] = df['Gross Sales'].str.replace(',','')
```

In [ ]:
```
df['Sales'] = df['Sales'].str.replace('$','')
df['Sales'] = df['Sales'].str.replace('.','')
df['Sales'] = df['Sales'].str.replace(',','')
```

In [ ]:
```
df['COGS'] = df['COGS'].str.replace('$','')
df['COGS'] = df['COGS'].str.replace('.','')
df['COGS'] = df['COGS'].str.replace(',','')
```

Changing the Datatypes

In [ ]:
```
df['Units Sold'] = df['Units Sold'].astype(np.int64)
df['Manufacturing Price'] = df['Manufacturing Price'].astype(np.int64)
df['Sale Price'] = df['Sale Price'].astype(np.int64)
df['Gross Sales'] = df['Gross Sales'].astype(np.int64)
df['Sales'] = df['Sales'].astype(np.int64)
df['COGS'] = df['COGS'].astype(np.int64)
```

Update the Profit Columns

In [ ]:
```
df['Profit'] = df['Sales'] - df['COGS']
```

- Handling the "Discounts" Columns

In [ ]:
```
df['Discounts'] = df['Discounts'].str.replace('$','')
df['Discounts'] = df['Discounts'].str.replace('.','')
df['Discounts'] = df['Discounts'].str.replace(',','')
df['Discounts'] = df['Discounts'].str.strip()
df['Discounts'] = df['Discounts'].str.replace('-','0')
```

- Changing the Datatype for columns "Discounts"

```
In [ ]: df['Discounts'] = df['Discounts'].astype('float')
```

```
In [ ]: df['Date'] = pd.to_datetime(df['Date'])
```

```
In [ ]: df.info()
```

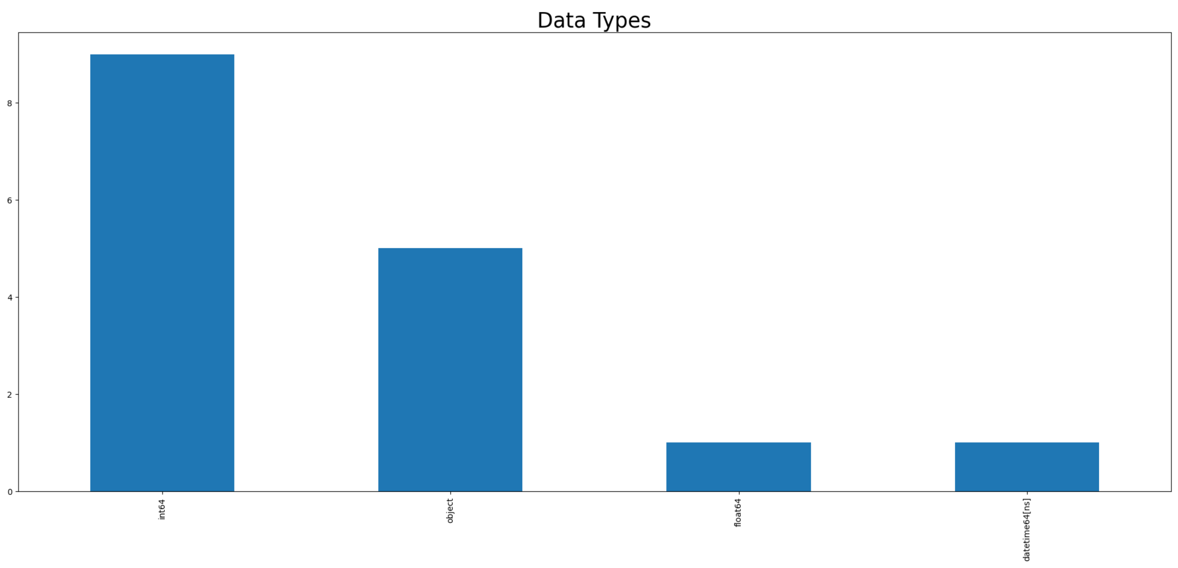
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 700 entries, 0 to 699
Data columns (total 16 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Segment              700 non-null    object
 1   Country              700 non-null    object
 2   Product              700 non-null    object
 3   Discount Band        700 non-null    object
 4   Units Sold           700 non-null    int64
 5   Manufacturing Price  700 non-null    int64
 6   Sale Price           700 non-null    int64
 7   Gross Sales          700 non-null    int64
 8   Discounts            700 non-null    float64
 9   Sales                700 non-null    int64
 10  COGS                 700 non-null    int64
 11  Profit               700 non-null    int64
 12  Date                 700 non-null    datetime64[ns]
 13  Month Number         700 non-null    int64
 14  Month Name           700 non-null    object
 15  Year                 700 non-null    int64
dtypes: datetime64[ns](1), float64(1), int64(9), object(5)
memory usage: 87.6+ KB
```

```
In [ ]: plt.figure(figsize=(25,10))

        plt.title('Data Types', fontsize=25)

        df.dtypes.value_counts().plot(kind="bar")
```

```
Out[ ]: <Axes: title={'center': 'Data Types'}>
```



```
In [ ]: df.to_csv('Financials_Final_Data.csv', index=False)
```

```
In [ ]: df
```

Out[ ]:

	Segment	Country	Product	Discount Band	Units Sold	Manufacturing Price	Sale Price	Gro Sal
0	Government	Canada	Carretera	None	161850	300	2000	32370
1	Government	Germany	Carretera	None	132100	300	2000	26420
2	Midmarket	France	Carretera	None	217800	300	1500	32670
3	Midmarket	Germany	Carretera	None	88800	300	1500	13320
4	Midmarket	Mexico	Carretera	None	247000	300	1500	37050
...	...	...	...	...	...	...	...	
695	Small Business	France	Amarilla	High	247500	26000	30000	742500
696	Small Business	Mexico	Amarilla	High	54600	26000	30000	163800
697	Government	Mexico	Montana	High	136800	500	700	9576
698	Government	Canada	Paseo	High	72300	1000	700	5061
699	Channel Partners	United States of America	VTT	High	180600	25000	1200	21672

700 rows × 16 columns