Sales Analysis Project

In [2]:
```
# importing libraries
import pandas as pd
import glob
import os

# merging the files
joined_files = os.path.join("Sales*.csv")

# A list of all joined files is returned
joined_list = glob.glob(joined_files)

# Finally, the files are joined
all_data = pd.concat(map(pd.read_csv, joined_list), ignore_index=True)
all_data.head(10)
```

Out[2]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
1	NaN	NaN	NaN	NaN	NaN	NaN
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001
6	176562	USB-C Charging Cable	1	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016
7	176563	Bose SoundSport Headphones	1	99.99	04/02/19 07:46	668 Center St, Seattle, WA 98101
8	176564	USB-C Charging Cable	1	11.95	04/12/19 10:58	790 Ridge St, Atlanta, GA 30301
9	176565	Macbook Pro Laptop	1	1700	04/24/19 10:38	915 Willow St, San Francisco, CA 94016

Step 1: Data Cleaning

In [3]:
```
# To check the number of null values in the data frame
all_data.isna().sum()
```

Out[3]:
```
Order ID            545
Product             545
Quantity Ordered    545
Price Each          545
Order Date          545
Purchase Address    545
dtype: int64
```

In [4]:
```
# To drop the rows with null values
all_data.dropna(how='all',inplace= True)
```

In [5]:
```
# To drop the rows with order date written inccurately, instead of date 'or' was writt
all_data= all_data[all_data['Order Date'].str[0:2]!= 'Or']
```

In [6]:
```
#This is clean data
all_data.head()
```

Out[6]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001

Data transformation

In [7]:
```
all_data.dtypes
```

Out[7]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
dtype: object
```

In [8]:
```
# Convert quntity to int
all_data['Quantity Ordered']= all_data.loc[:,('Quantity Ordered')].astype('int32')
# Convert price to float
all_data['Price Each']= all_data.loc[:,('Price Each')].astype('float64')
# Convert order date to datetime
all_data['Order Date']= pd.to_datetime(all_data['Order Date'])
```

In [9]:
```
# Check if the conversion is correct
all_data.dtypes
```

Out[9]:
```
Order ID                    object
Product                     object
Quantity Ordered             int32
Price Each                 float64
Order Date          datetime64[ns]
Purchase Address            object
dtype: object
```

Data Visualization

Q1. What is the best Month for sales? How much was earned in that month?

In [10]:
```
# First, a new column was created by extracting month from order date
all_data['Month']= all_data['Order Date'].dt.month
all_data.head()
```

Out[10]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4

In [11]:
```
# Sales column was created by multiplying unit price with quantity
all_data['Sales']= all_data.loc[:,('Quantity Ordered')]*all_data.loc[:,('Price Each')]
all_data.head()
```

Out[11]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99

In [12]:
```
# Created a new dataframe containing Month and sales columns only according to the req
df1 = all_data[['Month','Sales']]
df1.head()
```

Out[12]:

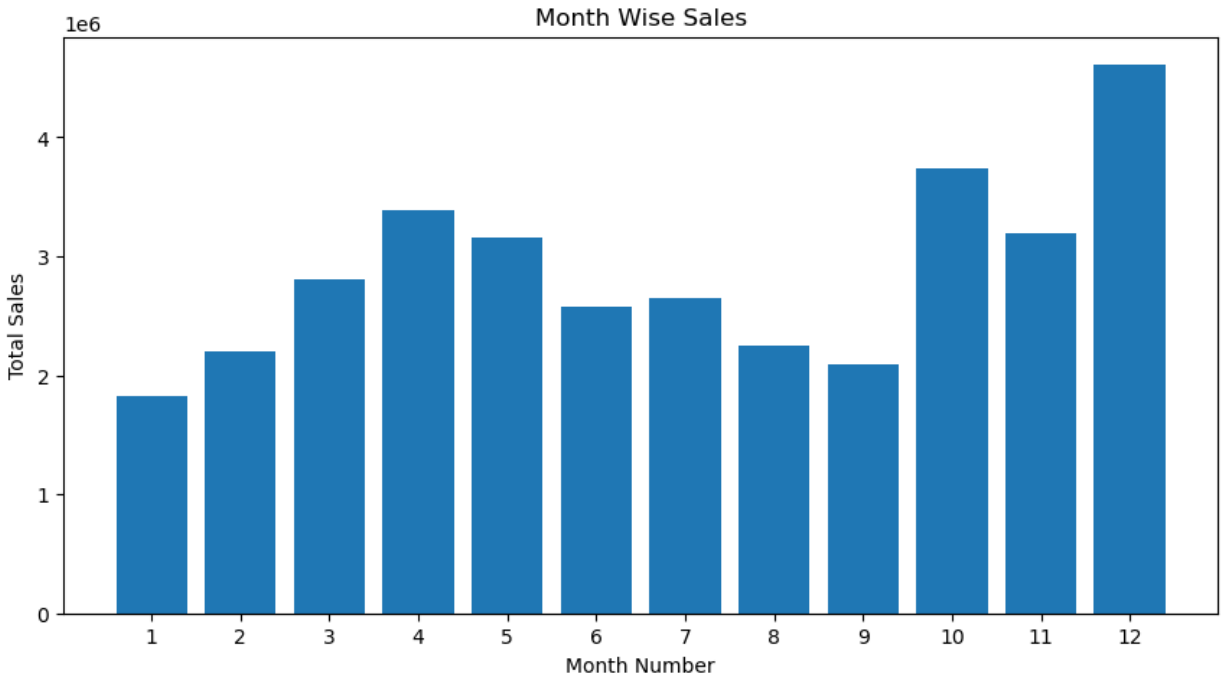
	Month	Sales
0	4	23.90
2	4	99.99
3	4	600.00
4	4	11.99
5	4	11.99

In [13]:
```
# Applied the aggregation function to make a summerized table showing month wise total
df1= df1.groupby(by='Month').sum('Sales').reset_index()
df1
```

Out[13]:

	Month	Sales
0	1	1822256.73
1	2	2202022.42
2	3	2807100.38
3	4	3390670.24
4	5	3152606.75
5	6	2577802.26
6	7	2647775.76
7	8	2244467.88
8	9	2097560.13
9	10	3736726.88
10	11	3199603.20
11	12	4613443.34

In [14]:
```
# Bar graph is plotted to compare the sales values over 12 months
import matplotlib.pyplot as plt
fig = plt.figure(figsize=(8,4))
ax= fig.add_axes(rect=[0,0,1,1])
ax.set_xlabel('Month Number')
ax.set_ylabel('Total Sales')
ax.set_title('Month Wise Sales')
plt.xticks([1,2,3,4,5,6,7,8,9,10,11,12],[1,2,3,4,5,6,7,8,9,10,11,12])
plt.bar(x= df1['Month'],height=df1['Sales'])
plt.show()
```



This graph shows that maximum sales were recorded in the month of December. So December is the best month of Sales and total sales amount is nearly 4.6 million USD

Q2: Which city has highest number of sales?

In [15]:
```
all_data.head()
```

Out[15]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99

In [16]:
```
# Add a city column
# create a function to grab city from address
def get_city(address):
    return address.split(',')[1]
# create a new column named city by applying the function on address column
all_data['City']= all_data.loc[:,('Purchase Address')].apply(lambda x: get_city(x))
# change the data type of city to string
all_data['City']= all_data['City'].astype('str')
all_data.head()
```

Out[16]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90	Dallas
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99	Boston
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles

In [17]:
```
# Make a new dataframe to extract the City and Sales Columns
df2= all_data[['City','Sales']]
```

In [18]:
```
df2
```

Out[18]:

	City	Sales
0	Dallas	23.90
2	Boston	99.99
3	Los Angeles	600.00
4	Los Angeles	11.99
5	Los Angeles	11.99
...	...	...
186845	Los Angeles	8.97
186846	San Francisco	700.00
186847	San Francisco	700.00
186848	San Francisco	379.99
186849	San Francisco	11.95

185950 rows × 2 columns

In [19]:
```
# Aggregated the city column by sum of sales by using groupby
df2= df2.groupby(by='City',as_index=False).sum('Sales')
```

In [20]:
```
#Sorted the data frame to get a sorted graph
df2.sort_values('Sales', inplace= True)
```

In [21]:
```
df2
```

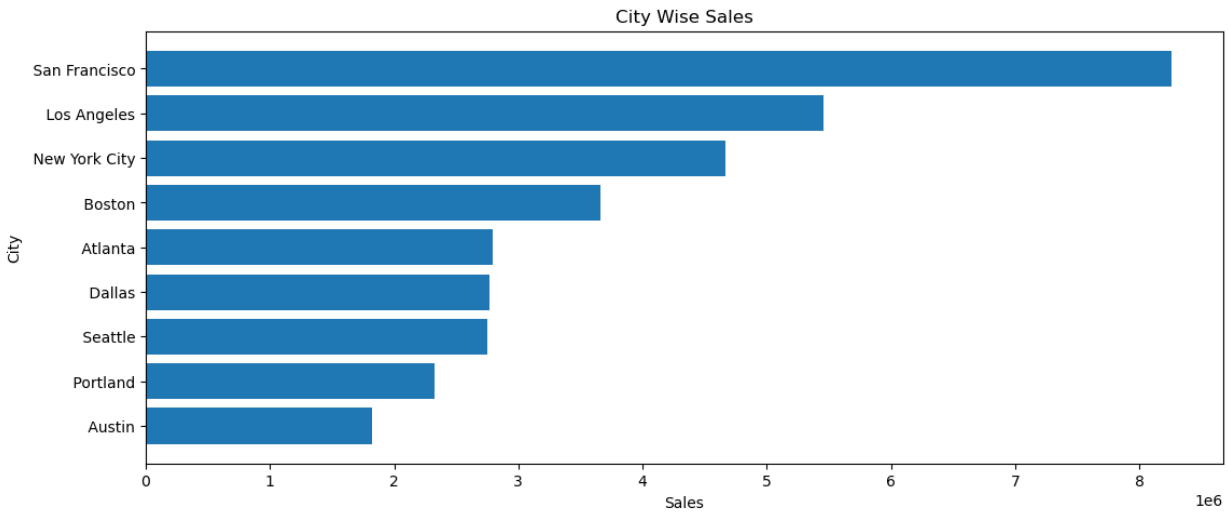
Out[21]:

	City	Sales
1	Austin	1819581.75
6	Portland	2320490.61
8	Seattle	2747755.48
3	Dallas	2767975.40
0	Atlanta	2795498.58
2	Boston	3661642.01
5	New York City	4664317.43
4	Los Angeles	5452570.80
7	San Francisco	8262203.91

In [22]:
```
# Plotted a horizonal bar graph to show the vaiation of sales by city
fig = plt.figure(figsize=(10,4))
```

```
ax= fig.add_axes(rect=[0,0,1,1])
ax.set_xlabel('Sales')
ax.set_ylabel('City')
ax.set_title('City Wise Sales')
plt.barh(df2['City'],df2['Sales'])
```

Out[22]:
```
<BarContainer object of 9 artists>
```



Heighest Sales are recorded in San Francisco

Q3: What should be the best time to display advertisememts to increase the sales?

In [23]:
```
all_data.head()
```

Out[23]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90	Dallas
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99	Boston
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles

In [24]:
```
# We will insert a new column for time
all_data['Hour']=all_data['Order Date'].dt.hour
```

In [25]:
```
df3= all_data[['Hour','Sales']]
```

In [26]:
```
df3
```

Out[26]:

	Hour	Sales
0	8	23.90
2	22	99.99
3	14	600.00
4	14	11.99
5	9	11.99
...	...	...
186845	20	8.97
186846	16	700.00
186847	7	700.00
186848	17	379.99
186849	0	11.95

185950 rows × 2 columns

In [27]:
```
# Used group by clause to aggregate hour with sum sales
df3= df3.groupby(by='Hour',as_index = False).sum('Sales')
```

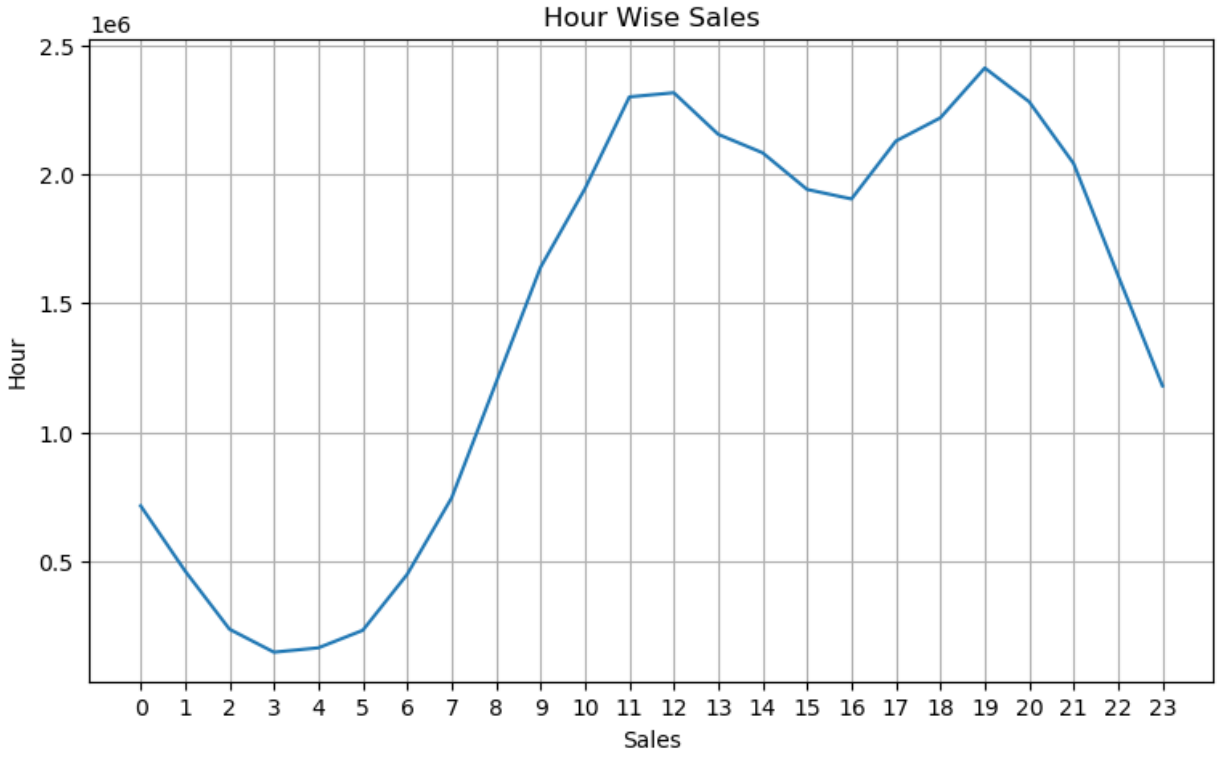
In [28]:
```
df3
```

Out[28]:

	Hour	Sales
0	0	713721.27
1	1	460866.88
2	2	234851.44
3	3	145757.89
4	4	162661.01
5	5	230679.82
6	6	448113.00
7	7	744854.12
8	8	1192348.97
9	9	1639030.58
10	10	1944286.77
11	11	2300610.24
12	12	2316821.34
13	13	2155389.80
14	14	2083672.73
15	15	1941549.60
16	16	1904601.31
17	17	2129361.61
18	18	2219348.30
19	19	2412938.54
20	20	2281716.24
21	21	2042000.86
22	22	1607549.21
23	23	1179304.44

In [29]:
```
# Plotted a line chart to show the hourly Sales
fig = plt.figure(figsize=(7,4))
ax= fig.add_axes(rect=[0,0,1,1])
ax.set_xlabel('Sales')
ax.set_ylabel('Hour')
ax.set_title('Hour Wise Sales')
plt.xticks(df3['Hour'])
plt.grid()
plt.plot(df3['Hour'],df3['Sales'])
```

Out[29]:
```
[<matplotlib.lines.Line2D at 0x15424fd9b70>]
```



Maximum orders were placed at 11 am and 7 pm . So the adds should be displayed during this time

Q4: Which product is sold the most?

In [30]:
```
all_data.head()
```

Out[30]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90	Dallas	8
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99	Boston	22
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles	14
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles	14
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles	9

In [31]:
```
# Created a data frame for required columns
df5= all_data[['Product','Quantity Ordered']]
```

In [32]:
```
# Aggregated the dataframe by using group by function to group sales with products
df5=df5.groupby(by='Product',as_index = False).sum()
```

In [33]:
```
# Sorted the columns to get a good visual
df5= df5.sort_values('Quantity Ordered')
```

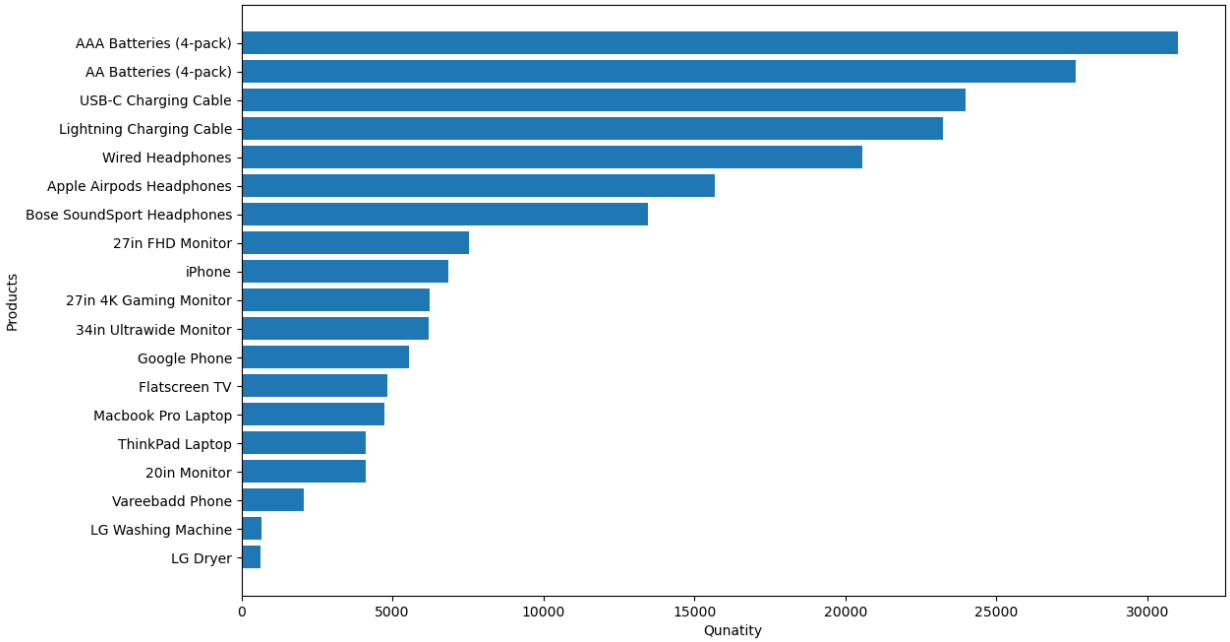
In [34]:
```
df5
```

Out[34]:

	Product	Quantity Ordered
10	LG Dryer	646
11	LG Washing Machine	666
16	Vareebadd Phone	2068
0	20in Monitor	4129
14	ThinkPad Laptop	4130
13	Macbook Pro Laptop	4728
8	Flatscreen TV	4819
9	Google Phone	5532
3	34in Ultrawide Monitor	6199
1	27in 4K Gaming Monitor	6244
18	iPhone	6849
2	27in FHD Monitor	7550
7	Bose SoundSport Headphones	13457
6	Apple Airpods Headphones	15661
17	Wired Headphones	20557
12	Lightning Charging Cable	23217
15	USB-C Charging Cable	23975
4	AA Batteries (4-pack)	27635
5	AAA Batteries (4-pack)	31017

In [35]:
```
# Plotted a horizontal bar graph to compare the sales of individual products
fig = plt.figure(figsize=(10,6))
ax= fig.add_axes(rect=[0,0,1,1])
ax.set_xlabel('Qunatity')
ax.set_ylabel('Products')
plt.barh(df5['Product'],df5['Quantity Ordered'])
```

Out[35]:
```
<BarContainer object of 19 artists>
```



Bar graph shows that the mostly sold products include Batteries, Charging Cables and headphones. Least sold products are Washing machines, Laptop and TV. The reason behind is due to price difference and age of the product