Import Necessary libraries

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

Read the Excel file

In [2]:
```
df= pd.read_excel('sales.xlsx')
df.head()
```

Out[2]:

	Date	Time	State	Group	Unit	Sales
0	2020-10-01	Morning	WA	Kids	8	20000
1	2020-10-01	Morning	WA	Men	8	20000
2	2020-10-01	Morning	WA	Women	4	10000
3	2020-10-01	Morning	WA	Seniors	15	37500
4	2020-10-01	Afternoon	WA	Kids	3	7500

Data Wrangling

In [3]:
```
df.describe()
```

Out[3]:

	Unit	Sales
count	7560.000000	7560.000000
mean	18.005423	45013.558201
std	12.901403	32253.506944
min	2.000000	5000.000000
25%	8.000000	20000.000000
50%	14.000000	35000.000000
75%	26.000000	65000.000000
max	65.000000	162500.000000

In [4]:
```
#now we found out the shape of the dataset
df.shape
```

Out[4]:
```
(7560, 6)
```

In [5]:
```
#with this we will find out that is there any null value in the dataset
df.isnull().sum()
```

Out[5]:
```
Date      0
Time      0
State     0
Group     0
Unit      0
Sales     0
dtype: int64
```

In [6]:
```
#now we find the incorrect data
df.isna()
```

Out[6]:

	Date	Time	State	Group	Unit	Sales
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
7555	False	False	False	False	False	False
7556	False	False	False	False	False	False
7557	False	False	False	False	False	False
7558	False	False	False	False	False	False
7559	False	False	False	False	False	False

7560 rows × 6 columns

In [7]:
```
df.tail()
```

Out[7]:

	Date	Time	State	Group	Unit	Sales
7555	2020-12-30	Afternoon	TAS	Seniors	14	35000
7556	2020-12-30	Evening	TAS	Kids	15	37500
7557	2020-12-30	Evening	TAS	Men	15	37500
7558	2020-12-30	Evening	TAS	Women	11	27500
7559	2020-12-30	Evening	TAS	Seniors	13	32500

Data Analysis

In [8]:
```
#Determine which group is generating the highest sales, and which group is generating the lowest sales.
gb_group= df.groupby('Group')
gb_group.first()
```

Out[8]:

	Date	Time	State	Unit	Sales
Group					
Kids	2020-10-01	Morning	WA	8	20000
Men	2020-10-01	Morning	WA	8	20000
Seniors	2020-10-01	Morning	WA	15	37500
Women	2020-10-01	Morning	WA	4	10000

In [9]:
```
sorted_gb_group = gb_group['Sales'].sum().sort_values(ascending=False)
sorted_gb_group
```

Out[9]:
```
Group
Men        85750000
Women      85442500
Kids       85072500
Seniors    84037500
Name: Sales, dtype: int64
```

In [10]:
```
#Determine which state is generating the highest sales, and which group is generating the lowest sales.
gb_state=df.groupby('State')
gb_state.first()
```

Out[10]:

	Date	Time	Group	Unit	Sales
State					
NSW	2020-10-01	Morning	Kids	39	97500
NT	2020-10-01	Morning	Kids	13	32500
QLD	2020-10-01	Morning	Kids	20	50000
SA	2020-10-01	Morning	Kids	12	30000
TAS	2020-10-01	Morning	Kids	13	32500
VIC	2020-10-01	Morning	Kids	49	122500
WA	2020-10-01	Morning	Kids	8	20000

In [11]:
```
sorted_gb_state = gb_state['Sales'].sum().sort_values(ascending=False)
sorted_gb_state
```

Out[11]:
```
State
 VIC    105565000
 NSW     74970000
 SA      58857500
 QLD     33417500
 TAS     22760000
 NT      22580000
 WA      22152500
Name: Sales, dtype: int64
```

weekly, monthly and quarterly reports

In [12]:
```
#Generate weekly, monthly and quarterly reports for the analysis made.
df['Date'] = pd.to_datetime(df['Date'])
```

In [13]:
```
df.set_index('Date', inplace=True)
```

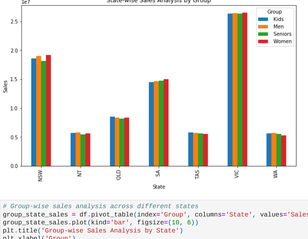
In [14]:
```
weekly_report = df.resample('W').sum()
print(weekly_report)
```
```
            Unit     Sales
Date
2020-10-04   6018  15045000
2020-10-11  10801  27002500
2020-10-18  16056  26640000
2020-10-25  10726  26815000
2020-11-01   8723  21807500
2020-11-08   8346  20865000
2020-11-15   8469  21172500
2020-11-22   8445  21112500
2020-11-29   8591  21477500
2020-12-06  11849  29622500
2020-12-13  12610  31525000
2020-12-20  12662  31655000
2020-12-27  12708  31770000
2021-01-03   5517  13792500
```

In [15]:
```
monthly_report = df.resample('M').sum()
print(monthly_report)
```
```
            Unit     Sales
Date
2020-10-31  45716  114290000
2020-11-30  36273   90682500
2020-12-31  54132  135330000
```

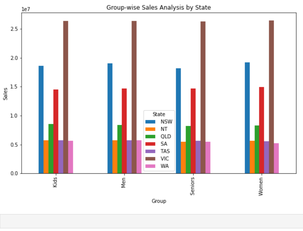
In [16]:
```
quarterly_report = df.resample('Q').sum()
print(quarterly_report)
```
```
            Unit     Sales
Date
2020-12-31  136121  340302500
```

In [17]:
```
monthly_report.plot(kind='bar', figsize=(12, 6))
plt.title('Monthly Sales Report')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.show()
```

In [18]:
```
# State-wise sales analysis for different groups
state_group_sales = df.pivot_table(index='State', columns='Group', values='Sales', aggfunc='sum')
state_group_sales.plot(kind='bar', figsize=(10, 6))
plt.title('State-wise Sales Analysis by Group')
plt.xlabel('State')
plt.ylabel('Sales')
plt.show()
```



In [19]:
```
# Group-wise sales analysis across different states
group_state_sales = df.pivot_table(index='Group', columns='State', values='Sales', aggfunc='sum')
group_state_sales.plot(kind='bar', figsize=(10, 6))
plt.title('Group-wise Sales Analysis by State')
plt.xlabel('Group')
plt.ylabel('Sales')
plt.show()
```



In [ ]:
```
```