```
print(sales_analysis.columns)



import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# Load the data
data = pd.read_csv(r"C:\Users\aan50\OneDrive\Desktop\
AusApparalSales4thQrt2020.csv")


# Check for missing Values
missing_data = data.isna().sum()
print("Missing Data:\n", missing_data)

# Treat missing data
data.fillna(method='ffill', inplace=True)
# Normalization
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
numeric_data = data.select_dtypes(include=[np.number])
normalized_data = pd.DataFrame(scaler.fit_transform(numeric_data),
columns=numeric_data.columns)


# GroupBy for analysis
non_numeric_data = data.select_dtypes(exclude=[np.number])
normalized_data = pd.DataFrame(scaler.fit_transform(numeric_data),
columns=numeric_data.columns)
final_data = pd.concat([normalized_data,
non_numeric_data.reset_index(drop=True)], axis=1)

# Data Analysis
# Descriptive statistics
sales_stats = data[['Sales', 'Unit']].describe()
print("Descriptive Statistics:\n", sales_stats)

# Highest and lowest sales
highest_sales = data.loc[data['Sales'].idxmax()]
lowest_sales = data.loc[data['Sales'].idxmin()]
print("Highest Sales:\n", highest_sales)
print("Lowest Sales:\n", lowest_sales)


data['Date'] = pd.to_datetime(data['Date'])
```

```
# Set 'Date' as the index
data.set_index('Date', inplace=True)

# Resample to get weekly sums
weekly_report = data.resample('W').sum()
monthly_report = data.resample('M',).sum()
quarterly_report = data.resample('Q',).sum()

# Data Visualization
plt.figure(figsize=(12, 6))
sns.boxplot(x='Group', y='Sales', data=data)
plt.title('Sales Distribution by Demographic')
plt.show()

# State-wise sales analysis
plt.figure(figsize=(12, 6))
sns.barplot(x='State', y='Sales', data=data)
plt.title('State-wise Sales Analysis')
plt.xticks(rotation=45)
plt.show()

# Group-wise sales Analysis
group_sales = data.groupby('Group')['Sales'].sum().reset_index()
plt.figure(figsize=(12, 6))
sns.barplot(x='Group', y='Sales', data=group_sales)
plt.title('Group-wise Sales Analysis')
plt.show()
```

```
print(data.head())

import pandas as pd
import matplotlib.pyplot as plt

# Load your data
data = pd.read_csv(r"C:\Users\aan50\OneDrive\Desktop\
AusApparalSales4thQrt2020.csv")

# Print the first few rOws to ensure data is loaded
print(data.head())

# Map time categories to numerical values
time_mapping = {
    'Morning': 1,
    'Afternoon': 2,
```

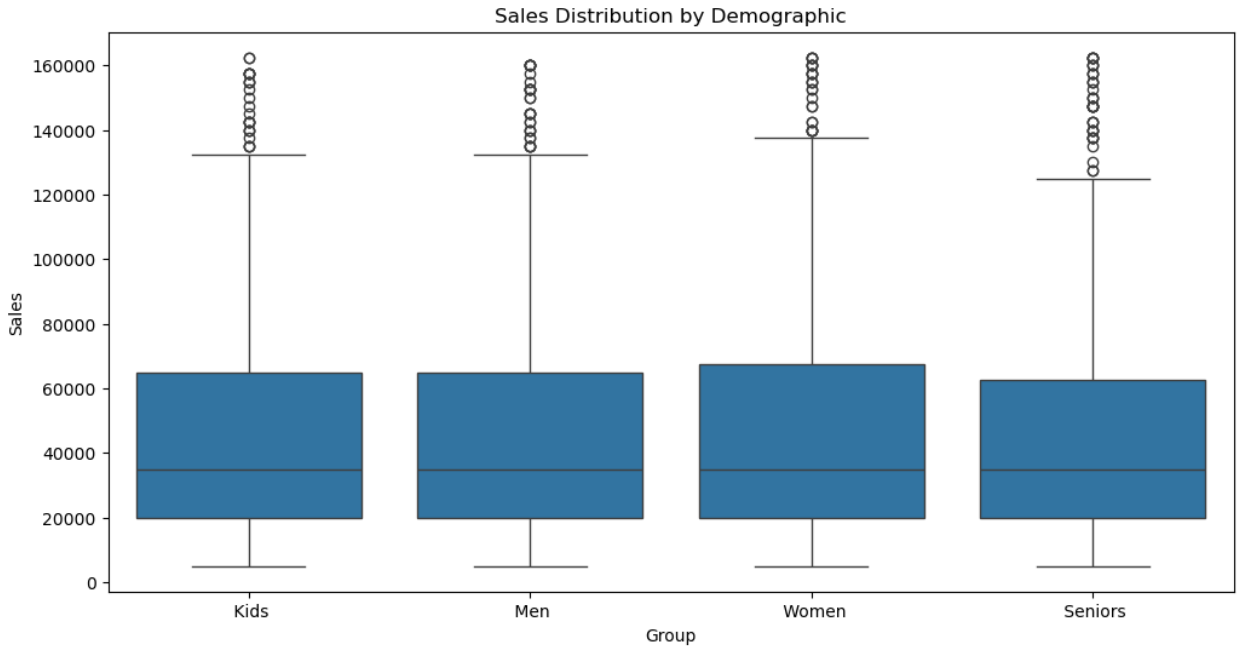
```
    'Evening': 3
}

# Ensure 'TimeOfDay' column exists and map
if 'Time' in data.columns:
    data['TimePeriod'] = data['Time'].map(time_mapping)

    # Check for NaN values
    print("Number of NaN values in TimePeriod:",
data['TimePeriod'].isna().sum())

    # Group by TimePeriod and sum Sales
    sales_analysis = data.groupby('TimePeriod')
['Sales'].sum().reset_index()
    sales_analysis.columns = ['TimePeriod', 'Total Sales']

    # Print sales_analysis to check for data
    print(sales_analysis)

    # Check if sales_analysis is empty before finding peak and off-
peak periods
    if not sales_analysis.empty:
        peak_period = sales_analysis.loc[sales_analysis['Total
Sales'].idxmax()]
        off_peak_period = sales_analysis.loc[sales_analysis['Total
Sales'].idxmin()]

        print("Peak Sales Period:")
        print(peak_period)

        print("\nOff-Peak Sales Period:")
        print(off_peak_period)

        # Plotting
        plt.figure(figsize=(10, 6))
        plt.bar(sales_analysis['TimePeriod'], sales_analysis['Total
Sales'], color='skyblue')
        plt.xlabel('Time ')
        plt.ylabel('Total Sales')
        plt.title('Total Sales by Time of Day')
        plt.xticks(ticks=sales_analysis['TimePeriod'],
labels=['Morning', 'Afternoon', 'Evening'])
        plt.grid(axis='y')
        plt.show()
    else:
        print("No sales data available in sales_analysis.")
else:
    print("'TimeOfDay' column not found in data.")
```

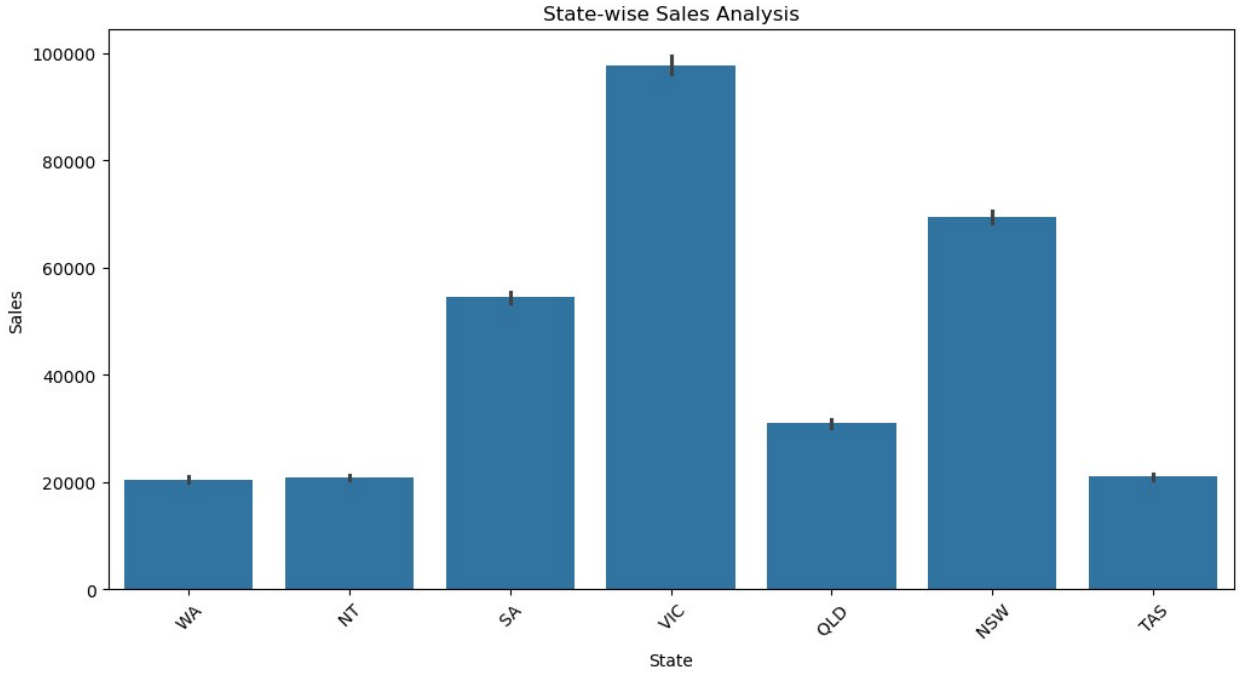
```
# Save the Reports to CSV
weekly_report.to_csv('weekly_report.csv')
monthly_report.to_csv('monthly_report.csv')
quarterly_report.to_csv('quarterly_report.csv')
```

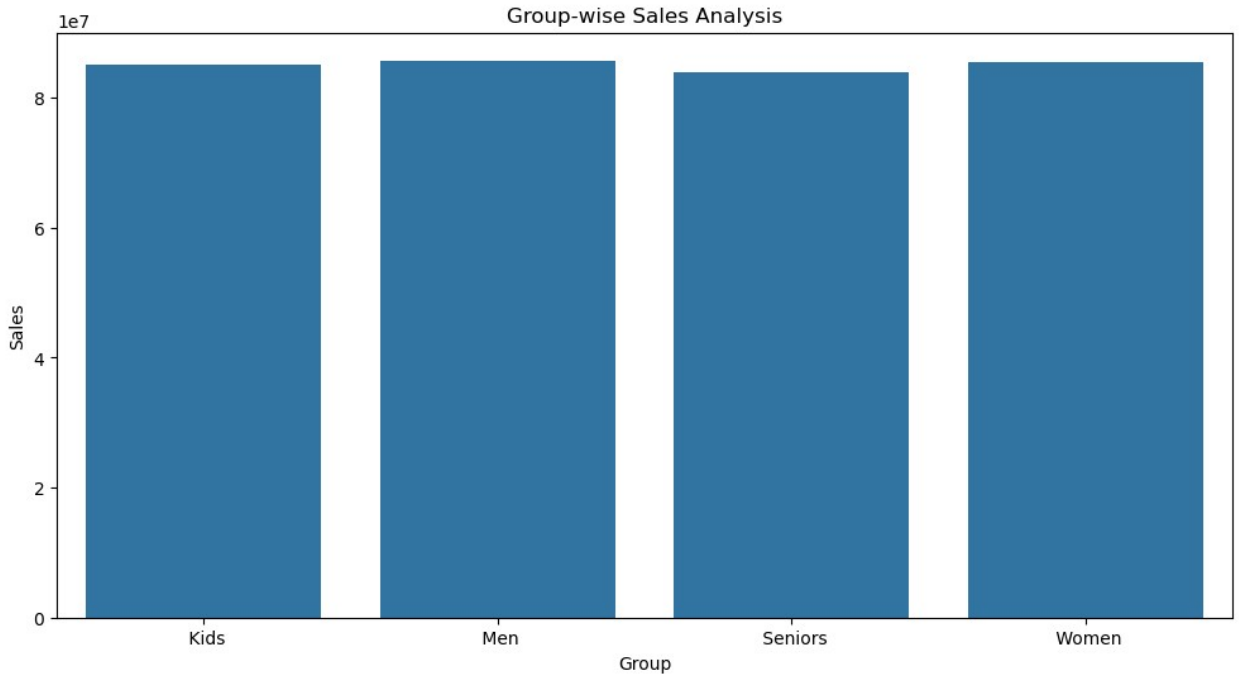
```
Missing Data:
 Date     0
Time      0
State     0
Group     0
Unit      0
Sales     0
dtype: int64
Descriptive Statistics:
                Sales         Unit
count    7560.000000  7560.000000
mean    45013.558201    18.005423
std     32253.506944    12.901403
min      5000.000000     2.000000
25%     20000.000000     8.000000
50%     35000.000000    14.000000
75%     65000.000000    26.000000
max    162500.000000    65.000000
Highest Sales:
 Date     5-Dec-2020
Time        Evening
State           VIC
Group       Seniors
Unit             65
Sales        162500
Name: 5423, dtype: object
Lowest Sales:
 Date     1-Nov-2020
Time        Morning
State            NT
Group           Men
Unit              2
Sales          5000
Name: 2533, dtype: object
```

```
C:\Users\aan50\AppData\Local\Temp\ipykernel_16112\1914659867.py:15:
FutureWarning: DataFrame.fillna with 'method' is deprecated and will
raise in a future version. Use obj.ffill() or obj.bfill() instead.
  data.fillna(method='ffill', inplace=True)
C:\Users\aan50\AppData\Local\Temp\ipykernel_16112\1914659867.py:49:
FutureWarning: 'M' is deprecated and will be removed in a future
version, please use 'ME' instead.
  monthly_report = data.resample('M',).sum()
C:\Users\aan50\AppData\Local\Temp\ipykernel_16112\1914659867.py:50:
FutureWarning: 'Q' is deprecated and will be removed in a future
```

```
version, please use 'QE' instead.
  quarterly_report = data.resample('Q',).sum()
```








```
                  Time State     Group  Unit  Sales
Date
2020-10-01     Morning    WA      Kids     8  20000
2020-10-01     Morning    WA       Men     8  20000
2020-10-01     Morning    WA     Women     4  10000
2020-10-01     Morning    WA   Seniors    15  37500
2020-10-01   Afternoon    WA      Kids     3   7500
         Date        Time State     Group  Unit  Sales
0  1-Oct-2020     Morning    WA      Kids     8  20000
1  1-Oct-2020     Morning    WA       Men     8  20000
2  1-Oct-2020     Morning    WA     Women     4  10000
3  1-Oct-2020     Morning    WA   Seniors    15  37500
4  1-Oct-2020   Afternoon    WA      Kids     3   7500
Number of NaN values in TimePeriod: 7560
Empty DataFrame
Columns: [TimePeriod, Total Sales]
Index: []
No sales data available in sales_analysis.
```

```
print(sales_analysis.columns)
```

```
Index(['TimePeriod', 'Total Sales'], dtype='object')
```

```
print(sales_analysis.columns)
```

```
Index(['TimePeriod', 'Total Sales'], dtype='object')
```