Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable.

Name - Suraj Sawant Roll No - TEB38

```
import pandas as pd
```

```
df=pd.DataFrame()
```

```
df[ 'Age Group']=['Young Adult', 'Young Adult', 'Young Adult', 'Young Adult', 'Mid Age Adult', 'Mid Age Adult',
'Senior Adult', 'Senior Adult', 'Senior Adult', 'Senior Adult']
```

```
df['Income']=[30000,35000,32000,33000, 50000,55000,75000,80000,78000,82000]
```

```
df
```

	Age Group	Income
0	Young Adult	30000
1	Young Adult	35000
2	Young Adult	32000
3	Young Adult	33000
4	Mid Age Adult	50000
5	Mid Age Adult	55000
6	Senior Adult	75000
7	Senior Adult	80000
8	Senior Adult	78000
9	Senior Adult	82000

```
df.info()
```

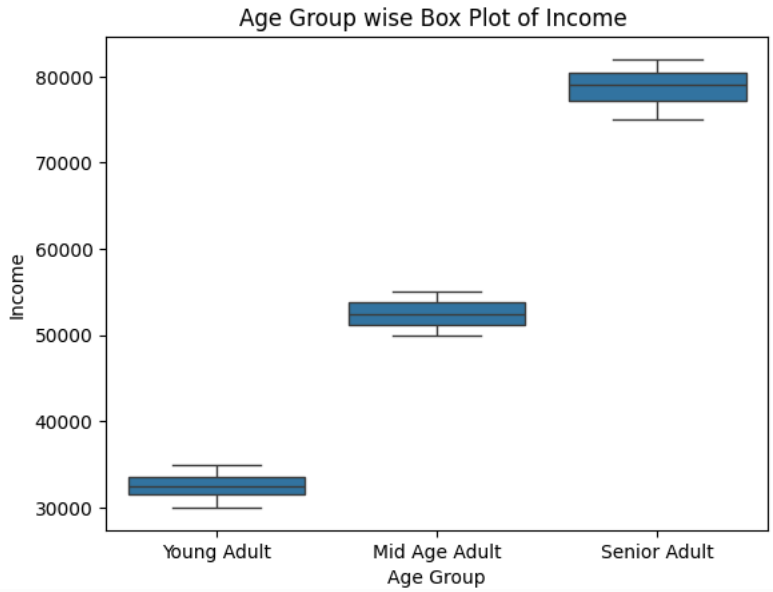
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Age Group  10 non-null     object
 1   Income     10 non-null     int64
dtypes: int64(1), object(1)
memory usage: 292.0+ bytes
```

```
df.describe()
```

	Income
count	10.000000
mean	55000.000000
std	21974.732965
min	30000.000000
25%	33500.000000
50%	52500.000000
75%	77250.000000
max	82000.000000

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(x='Age Group', y='Income', data=df)
plt.xlabel('Age Group')
plt.ylabel('Income')
plt.title('Age Group wise Box Plot of Income ')
plt.title('Age Group wise Box Plot of Income ')
```

```
Text(0.5, 1.0, 'Age Group wise Box Plot of Income ')
```




```
print(df.groupby('Age Group')['Income'].mean())
```

```
Age Group
Mid Age Adult    52500.0
Senior Adult     78750.0
Young Adult      32500.0
Name: Income, dtype: float64
```

```
print(df.groupby('Age Group')['Income'].median())
```

```
Age Group
Mid Age Adult    52500.0
Senior Adult     79000.0
Young Adult      32500.0
Name: Income, dtype: float64
```

```
df['Income'].mode()
```

	Income
0	30000
1	32000
2	33000
3	35000
4	50000
5	55000
6	75000
7	78000
8	80000
9	82000

dtype: int64

```
df['Age Group'].mode()
```

	Age Group
0	Senior Adult
1	Young Adult

dtype: object

```
print(df.groupby('Age Group')['Income'].min())
```

```
Age Group
Mid Age Adult    50000
Senior Adult     75000
```

```
Young Adult      30000
Name: Income, dtype: int64
```

```
print(df.groupby('Age Group')['Income'].max())
```

```
Age Group
Mid Age Adult    55000
Senior Adult     82000
Young Adult      35000
Name: Income, dtype: int64
```

```
print(df.groupby('Age Group')['Income'].max())
```

```
Age Group
Mid Age Adult    55000
Senior Adult     82000
Young Adult      35000
Name: Income, dtype: int64
```

```
print(df.groupby('Age Group')['Income'].std())
```

```
Age Group
Mid Age Adult    3535.533906
Senior Adult     2986.078811
Young Adult      2081.665999
Name: Income, dtype: float64
```

```
print(df.groupby('Age Group')['Income'].describe())
```

```
                count     mean          std      min      25%      50%  \
Age Group
Mid Age Adult    2.0  52500.0  3535.533906  50000.0  51250.0  52500.0
Senior Adult     4.0  78750.0  2986.078811  75000.0  77250.0  79000.0
Young Adult      4.0  32500.0  2081.665999  30000.0  31500.0  32500.0

                   75%      max
Age Group
Mid Age Adult  53750.0  55000.0
Senior Adult   80500.0  82000.0
Young Adult    33500.0  35000.0
```

Start coding or generate with AI.