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

In [2]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\16_Sleep_health_and_lifestyle_dataset.csv")
df.fillna(0,inplace=True)
df
```

Out[2]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	Heart Rate
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/83	77
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/80	75
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/80	75
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	85
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	85
...	...	...	...	...	...	...	...	...	...	...	...
369	370	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68
370	371	Female	59	Nurse	8.0	9	75	3	Overweight	140/95	68
371	372	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68
372	373	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68
373	374	Female	59	Nurse	8.1	9	75	3	Overweight	140/95	68

374 rows × 13 columns

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

Out[3]:

	Person ID	Gender	Age	Occupation	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	BMI Category	Blood Pressure	Heart Rate	D St
0	1	Male	27	Software Engineer	6.1	6	42	6	Overweight	126/83	77	4
1	2	Male	28	Doctor	6.2	6	60	8	Normal	125/80	75	10
2	3	Male	28	Doctor	6.2	6	60	8	Normal	125/80	75	10
3	4	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	85	3
4	5	Male	28	Sales Representative	5.9	4	30	8	Obese	140/90	85	3

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 374 entries, 0 to 373
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Person ID                374 non-null    int64  
 1   Gender                   374 non-null    object 
 2   Age                      374 non-null    int64  
 3   Occupation               374 non-null    object 
 4   Sleep Duration           374 non-null    float64
 5   Quality of Sleep         374 non-null    int64  
 6   Physical Activity Level  374 non-null    int64  
 7   Stress Level             374 non-null    int64  
 8   BMI Category             374 non-null    object 
 9   Blood Pressure           374 non-null    object 
 10  Heart Rate               374 non-null    int64  
 11  Daily Steps              374 non-null    int64  
 12  Sleep Disorder           374 non-null    object 
dtypes: float64(1), int64(7), object(5)
memory usage: 38.1+ KB
```

In [5]:
```
import seaborn as sns
```

In [6]:
```
df.describe()
```

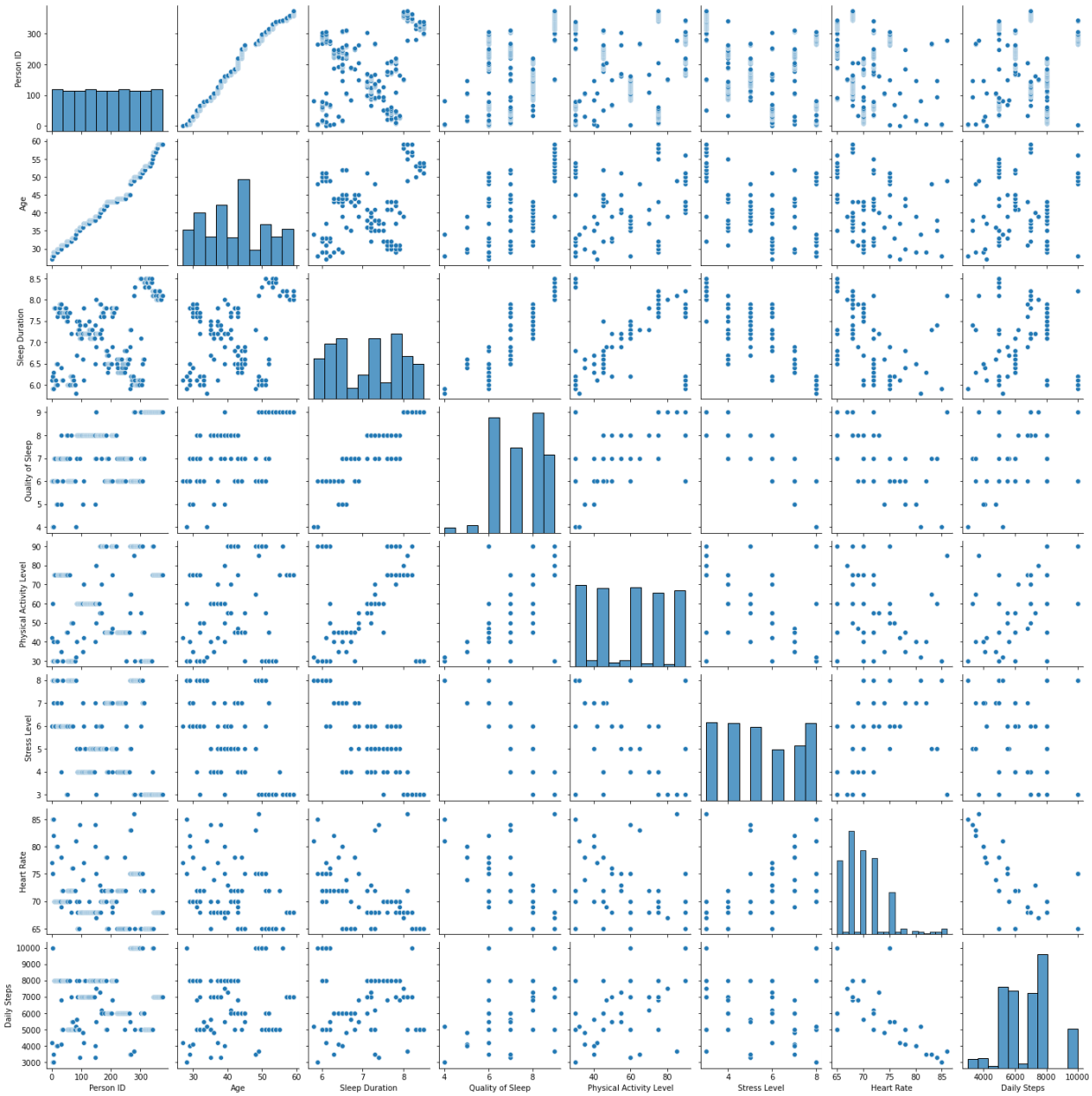
Out[6]:

	Person ID	Age	Sleep Duration	Quality of Sleep	Physical Activity Level	Stress Level	Heart Rate	Daily Steps
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000	374.000000
mean	187.500000	42.184492	7.132086	7.312834	59.171123	5.385027	70.165775	6816.844920
std	108.108742	8.673133	0.795657	1.196956	20.830804	1.774526	4.135676	1617.915679
min	1.000000	27.000000	5.800000	4.000000	30.000000	3.000000	65.000000	3000.000000
25%	94.250000	35.250000	6.400000	6.000000	45.000000	4.000000	68.000000	5600.000000
50%	187.500000	43.000000	7.200000	7.000000	60.000000	5.000000	70.000000	7000.000000
75%	280.750000	50.000000	7.800000	8.000000	75.000000	7.000000	72.000000	8000.000000
max	374.000000	59.000000	8.500000	9.000000	90.000000	8.000000	86.000000	10000.000000

Type *Markdown* and LaTeX: α^2

In [7]:
```
sns.pairplot(df)
```

Out[7]: <seaborn.axisgrid.PairGrid at 0x1df58870610>

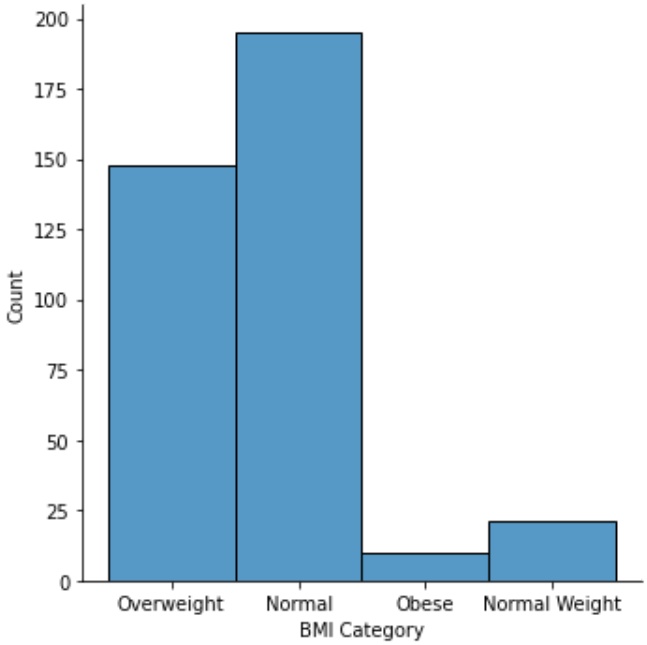


In [8]:
```
df1=df.drop(['Stress Level'],axis=1)
df1
df1=df1.drop(df1.index[1537:])
df1.isna().sum()
```

Out[8]:
```
Person ID                  0
Gender                     0
Age                        0
Occupation                 0
Sleep Duration             0
Quality of Sleep           0
Physical Activity Level    0
BMI Category               0
Blood Pressure             0
Heart Rate                 0
Daily Steps                0
Sleep Disorder             0
dtype: int64
```

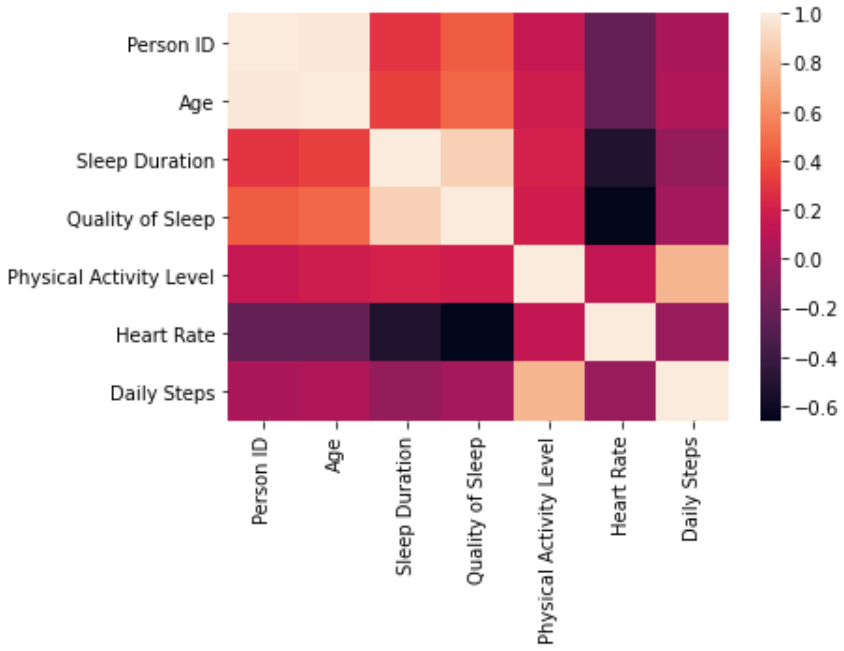
In [9]:
```
sns.displot(df['BMI Category'])
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x1df5b455a00>
```



In [10]:
```
sns.heatmap(df1.corr())
```

Out[10]: <AxesSubplot:>



In [11]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [12]:
```
df1.isna().sum()
```

Out[12]:
```
Person ID                  0
Gender                     0
Age                        0
Occupation                 0
Sleep Duration             0
Quality of Sleep           0
Physical Activity Level    0
BMI Category               0
Blood Pressure             0
Heart Rate                 0
Daily Steps                0
Sleep Disorder             0
dtype: int64
```

In [13]:
```
y=df1['Age']
x=df1.drop(['Gender','BMI Category','Sleep Disorder','Occupation','Blood Pressure'],axi
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
print(x_train)
```

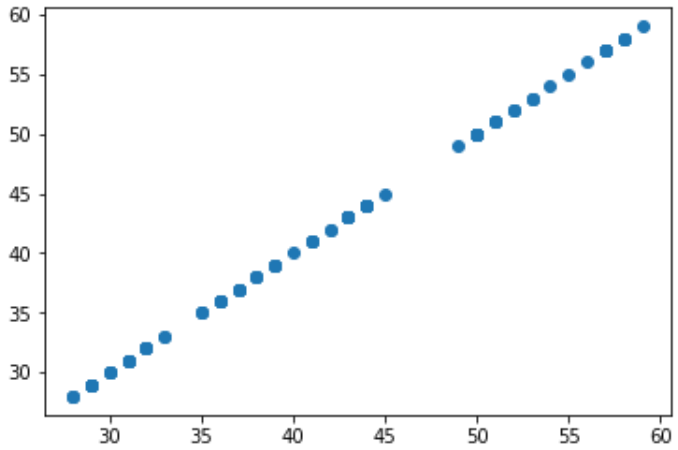
```
     Person ID  Age  Sleep Duration  Quality of Sleep  \
71          72   33             6.1                 6
19          20   30             7.6                 7
175        176   41             7.6                 8
60          61   32             6.0                 6
237        238   44             6.5                 7
..         ...  ...             ...               ...
82          83   35             6.7                 7
260        261   45             6.6                 7
141        142   38             7.1                 8
52          53   32             6.0                 6
63          64   32             6.2                 6

     Physical Activity Level  Heart Rate  Daily Steps
71                        30          72         5000
19                        75          70         8000
175                       90          70         8000
60                        30          72         5000
237                       45          65         6000
..                       ...         ...          ...
82                        40          70         5600
260                       45          65         6000
141                       60          68         8000
52                        30          72         5000
63                        30          72         5000

[261 rows x 7 columns]
```

In [14]:
```
model=LinearRegression()
model.fit(x_train,y_train)
model.intercept_
```

Out[14]:
```
-2.4940050025179517e-12
```

In [15]:
```
prediction=model.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]:
```
<matplotlib.collections.PathCollection at 0x1df5e6d5e20>
```



In [16]:
```
model.score(x_test,y_test)
```

Out[16]:
```
1.0
```

In [17]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [18]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[18]:
```
Ridge(alpha=10)
```

In [19]:
```
rr.score(x_test,y_test)
```

Out[19]:
```
0.9999871713014661
```

In [20]:
```
la =Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[20]:
```
Lasso(alpha=10)
```

In [21]:
```
la.score(x_test,y_test)
```

Out[21]:
```
0.9830345298585271
```

In [22]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[22]:
```
ElasticNet()
```

In [23]:
```
print(en.coef_)
```

```
[ 4.04552180e-02  4.88119337e-01  0.00000000e+00  0.00000000e+00
  6.35579605e-03 -0.00000000e+00 -1.88739823e-05]
```

In [24]:
```
print(en.intercept_)
```

```
13.754683960016738
```

In [25]:
```
print(en.predict(x_test))
```

```
[27.69554377 35.58127657 43.16705927 37.99502911 30.35816479 50.88163953
 43.70089624 56.00046888 53.07153475 41.18735929 52.95016909 50.59958761
 56.81222996 31.73629893 39.29225281 43.66044103 30.91119541 45.2353197
 42.12048602 37.08077635 27.75835342 36.55485852 43.49862015 50.03321456
 37.12123157 53.03107953 44.3453049  51.8956767  54.77879164 34.6858978
 41.95866515 45.07349883 27.73599899 32.12750867 28.38150417 43.81673035
 44.38576012 44.91167796 44.66894665 31.57273783 28.77661935 33.03352262
 53.80238539 37.79275302 45.51850623 57.42171495 29.65442693 35.82400787
 44.02453799 55.87910322 32.01948545 28.67802068 29.85670302 50.39731152
 46.33026731 51.72940835 34.84771867 34.76680823 51.1430261  49.95230412
 56.6908643  50.3568563  37.14281281 43.38547677 31.03256107 37.91411867
 28.7213002  31.61319305 50.11412499 41.2278145  40.08544669 30.95165063
 38.95744629 50.15458021 51.33043825 33.15488828 30.56974723 43.78180668
 55.96001366 35.45991091 56.12183453 35.54082135 39.53498411 44.99258839
 41.10644885 29.89715824 31.07301628 56.04092409 56.85268518 56.89314039
 28.59711025 55.30833322 51.81031879 31.85766458 55.91955844 29.61397171
 39.41361846 35.74309744 42.52216337 44.70940187 45.15440927 49.99275934
 49.58441583 45.27577492 28.85752979 48.97872217 39.4945289  40.85148522
 51.41134869 50.07366977 37.85208222 44.50712578 36.75713461]
```

In [27]:
```
print(en.score(x_test,y_test))
```

```
0.9958694565228404
```

In [28]:
```
from sklearn import metrics
```

In [29]:
```
print("Mean Absolute Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolute Error 1.783902425974694e-13
```

In [30]:
```
print("Mean Squared Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Squared Error 5.294376437654206e-26
```

In [32]:
```
print("Root Mean Squared Error:",np.sqrt(metrics.mean_squared_error(y_test,prediction))
```

```
Root Mean Squared Error: 2.3009512027972707e-13
```