In [38]:

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

In [39]:

```
url = 'https://raw.githubusercontent.com/Mauliklm10/Cartwheel.csv/master/datasetN
df = pd.read_csv(url)
```

In [40]:

```
df.head()
```

Out[40]:

	SEQN	ALQ101	ALQ110	ALQ130	SMQ020	RIAGENDR	RIDAGEYR	RIDRETH1	DMDCITZN	DI
0	83732	1.0	NaN	1.0	1	1	62	3	1.0	
1	83733	1.0	NaN	6.0	1	1	53	3	2.0	
2	83734	1.0	NaN	NaN	1	1	78	3	1.0	
3	83735	2.0	1.0	1.0	2	2	56	3	1.0	
4	83736	2.0	1.0	1.0	2	2	42	4	1.0	

5 rows × 28 columns

In [41]:

```
df.DMDEDUC2.value_counts()
```

Out[41]:

```
4.0    1621
5.0    1366
3.0    1186
1.0     655
2.0     643
9.0       3
Name: DMDEDUC2, dtype: int64
```

In [42]:

```
df.shape # to get the rows and columns
```

Out[42]:

```
(5735, 28)
```

In [43]:

```
df.DMDEDUC2.value_counts().sum()
# to get the rows in DMDEDUC2
```

Out[43]:

```
5474
```

In [44]:

```
# b[0]-a or
pd.isnull(df.DMDEDUC2).sum() # to find the null/mssing values
```

Out[44]:

```
261
```

In [45]:
```
# we created a new variable and gave the numbers above a better meaning
df['DMDEDUC2x'] = df.DMDEDUC2.replace({1: "<9", 2: "9-11",
                                      3: "HS grad", 4: "Undergrad",
                                      5: "Graduate", 7: "Refused",
                                      9: "Don't know"})
df.DMDEDUC2x.value_counts()
```

Out[45]:
```
Undergrad     1621
Graduate      1366
HS grad       1186
<9             655
9-11           643
Don't know       3
Name: DMDEDUC2x, dtype: int64
```

In [46]:
```
# we will create a new variable for gender
df['RIAGENDRx'] = df.RIAGENDR.replace({1: 'Male', 2: "Female"})
```

In [47]:
```
x = df.DMDEDUC2x.value_counts()
x/x.sum()
```

Out[47]:
```
Undergrad     0.296127
Graduate      0.249543
HS grad       0.216661
<9            0.119657
9-11          0.117464
Don't know    0.000548
Name: DMDEDUC2x, dtype: float64
```

In [48]:
```
df['DMDEDUC2x'] = df.DMDEDUC2x.fillna('Missing')
x = df.DMDEDUC2x.value_counts()
x / x.sum()
```

Out[48]:
```
Undergrad     0.282650
Graduate      0.238187
HS grad       0.206800
<9            0.114211
9-11          0.112119
Missing       0.045510
Don't know    0.000523
Name: DMDEDUC2x, dtype: float64
```

In [49]:

```
df.BMXWT.dropna().describe()
```

Out[49]:

```
count    5666.000000
mean       81.342676
std        21.764409
min        32.400000
25%        65.900000
50%        78.200000
75%        92.700000
max       198.900000
Name: BMXWT, dtype: float64
```

In [50]:

```
# x = df.BMXWT.dropna()
# using pandas
print(x.mean())
print(x.median())
print(x.quantile(0.75))

# using numpy
print(np.mean(x))
print(np.percentile(x, 50))
print(np.percentile(x, 75))
```

```
819.2857142857143
655.0
1276.0
819.2857142857143
655.0
1276.0
```

In [51]:

```
# proportion of subjects who are pre hypertension based
# on their systolic bp
np.mean((df.BPXSY1 >= 120) & (df.BPXSY2 <= 139))
```

Out[51]:

```
0.3741935483870968
```

In [52]:

```
# proportion of subjects who are pre hypertension based
# on their diastolic bp
np.mean((df.BPXDI1 >= 80) & (df.BPXDI2 <= 89))
```

Out[52]:

```
0.14803836094158676
```

In [53]:

```
# to check how many people are pre hypertension by using EITHER OR systolic or di
a = (df.BPXSY1 >= 120) & (df.BPXSY2 <= 139)
b = (df.BPXDI1 >= 80) & (df.BPXDI2 <= 89)
print(np.mean(a | b))
```

```
0.43975588491717527
```

In [54]:

```
# to check how many people are pre hypertension by using AND systolic and diastol
a = (df.BPXSY1 >= 120) & (df.BPXSY2 <= 139)
b = (df.BPXDI1 >= 80) & (df.BPXDI2 <= 89)
print(np.mean(a & b))
```

```
0.08247602441150828
```

In [55]:

```
# People are measured 3 times to take into account for people who may have higher
# be taken care of by taking BP multiple times.
print(np.mean(df.BPXSY1 - df.BPXSY2))
print(np.mean(df.BPXDI1 - df.BPXDI2))
```

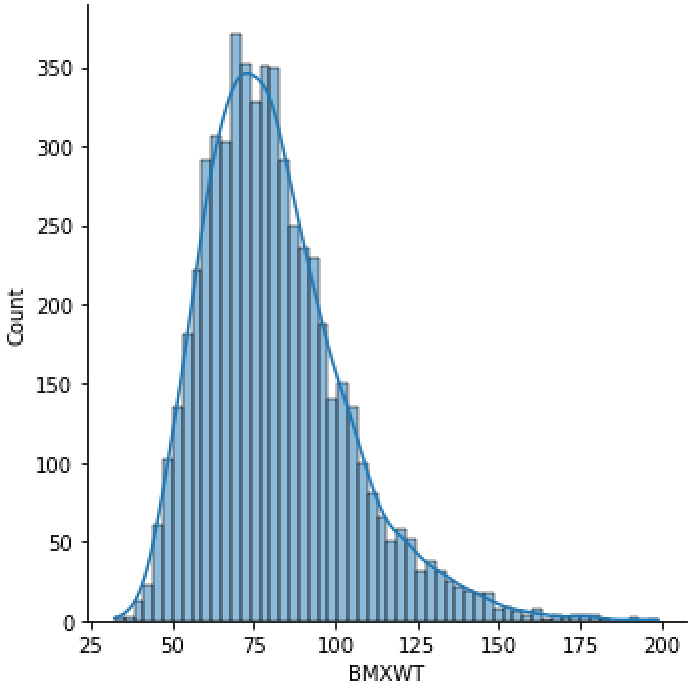
```
0.6749860309182343
0.3490407897187558
```

In [56]:

```
sns.displot(df['BMXWT'].dropna(), kde = True)
```

Out[56]:

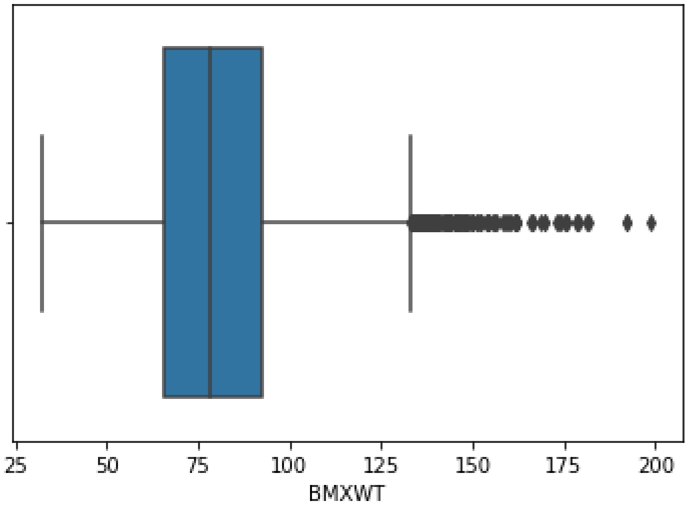
```
<seaborn.axisgrid.FacetGrid at 0x1a13d7f5370>
```



In [57]:
```
sns.boxplot(df['BMXWT'].dropna())
```

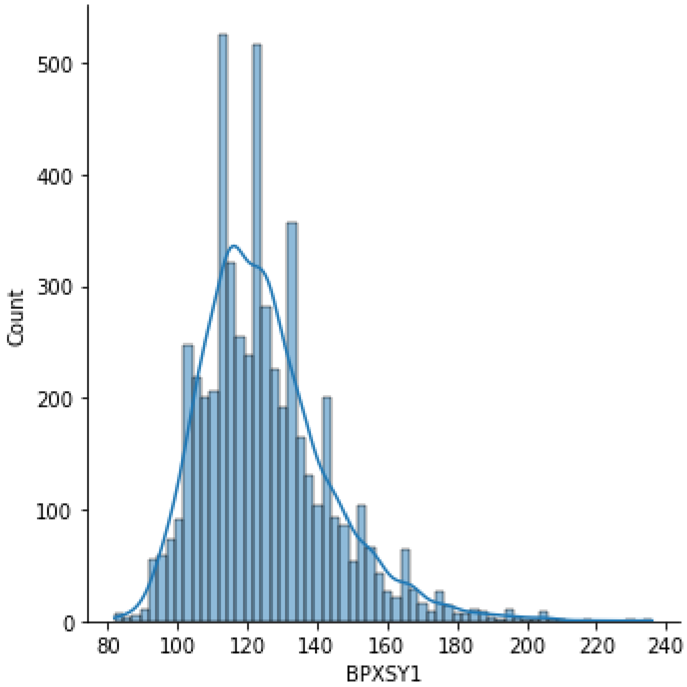
```
C:\Users\askma\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the o
nly valid positional argument will be `data`, and passing other arguments witho
ut an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[57]:
```
<AxesSubplot:xlabel='BMXWT'>
```

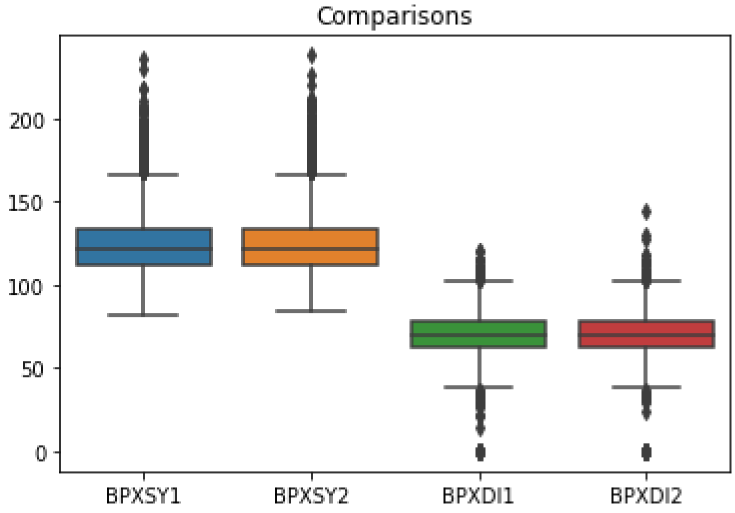


In [58]:
```
sns.displot(df['BPXSY1'].dropna(), kde = True)
```

Out[58]:
```
<seaborn.axisgrid.FacetGrid at 0x1a13d7ef5b0>
```

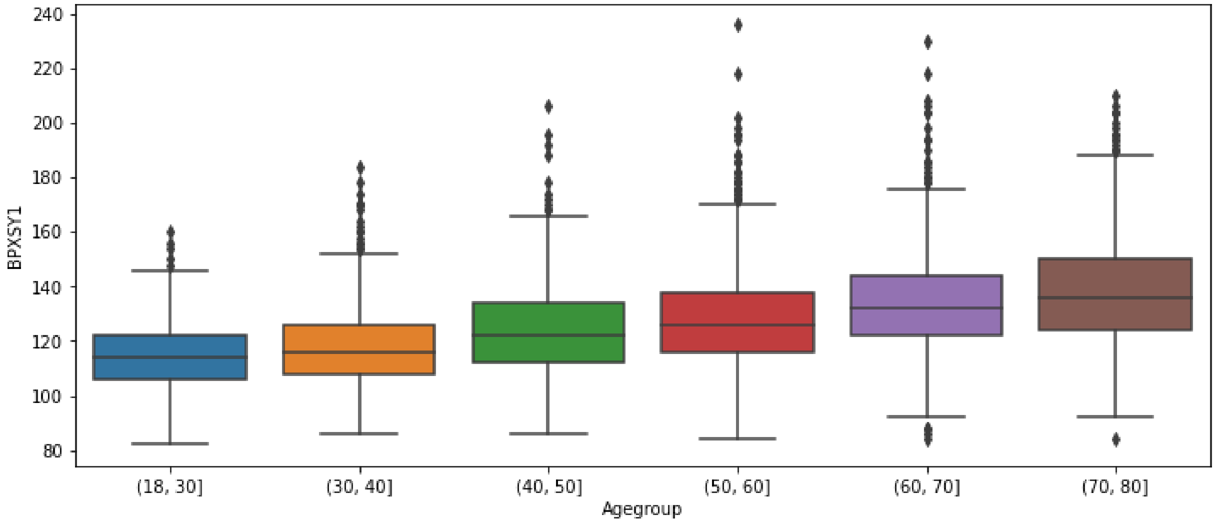


In [59]:
```
sns.boxplot(data=df.loc[:,['BPXSY1','BPXSY2','BPXDI1','BPXDI2']])
plt.title('Comparisons')
plt.show()
```



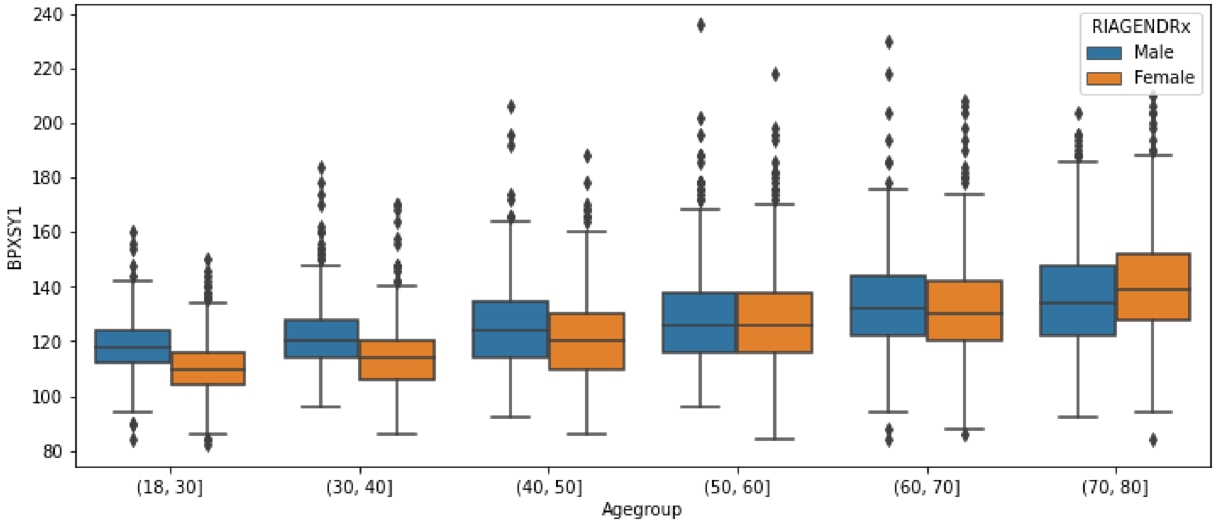
In [60]:
```
# Create age strata based on these cut points
df["Agegroup"] = pd.cut(df.RIDAGEYR,[18, 30, 40, 50, 60, 70, 80])
# Make the figure wider than default (12cm wide by 5cm tall)
plt.figure(figsize = (12, 5))
# Make boxplot of BPXSY1 stratified by age group
sns.boxplot(x = df['Agegroup'], y = df['BPXSY1'])
```

Out[60]:
```
<AxesSubplot:xlabel='Agegroup', ylabel='BPXSY1'>
```



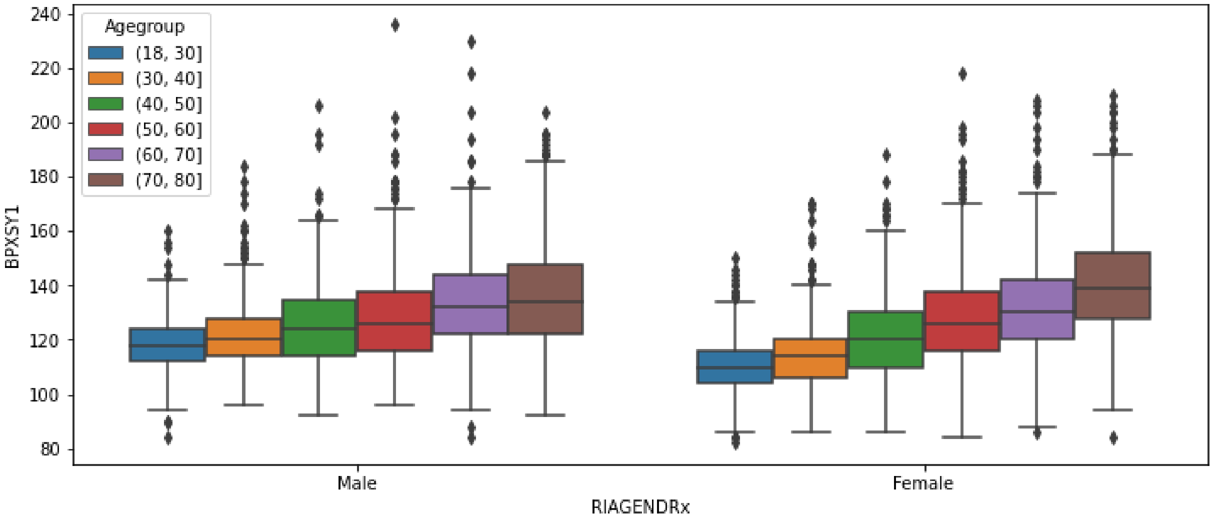
In [61]:
```
# Create age strata based on these cut points
df["Agegroup"] = pd.cut(df.RIDAGEYR,[18, 30, 40, 50, 60, 70, 80])
# Make the figure wider than default (12cm wide by 5cm tall)
plt.figure(figsize = (12, 5))
# Make boxplot of BPXSY1 stratified by age group but now with
# GENDER separartion as well
sns.boxplot(x = df['Agegroup'], y = df['BPXSY1'], hue = "RIAGENDRx", data=df)
```

Out[61]:
```
<AxesSubplot:xlabel='Agegroup', ylabel='BPXSY1'>
```



In [62]:
```
df["Agegroup"] = pd.cut(df.RIDAGEYR,[18, 30, 40, 50, 60, 70, 80])
# Make the figure wider than default (12cm wide by 5cm tall)
plt.figure(figsize = (12, 5))
# Make boxplot of BPXSY1 stratified by age group but now with
# GENDER separartion as well but now we stratify first based
# on geder then on age
sns.boxplot(x = df['RIAGENDRx'], y = df['BPXSY1'], hue = "Agegroup", data=df)
```

Out[62]:
```
<AxesSubplot:xlabel='RIAGENDRx', ylabel='BPXSY1'>
```



```
In [63]: df.groupby("Agegroup")["DMDEDUC2x"].value_counts()
```

```
Out[63]: Agegroup  DMDEDUC2x
         (18, 30]  Undergrad     364
                   Graduate      278
                   HS grad       237
                   Missing       128
                   9-11           99
                   <9             47
         (30, 40]  Undergrad     282
                   Graduate      264
                   HS grad       182
                   9-11          111
                   <9             93
         (40, 50]  Undergrad     262
                   Graduate      260
                   HS grad       171
                   9-11          112
                   <9             98
         (50, 60]  Undergrad     258
                   Graduate      220
                   HS grad       220
                   9-11          122
                   <9            104
         (60, 70]  Undergrad     238
                   HS grad       192
                   Graduate      188
                   <9            149
                   9-11          111
         (70, 80]  Undergrad     217
                   HS grad       184
                   <9            164
                   Graduate      156
                   9-11           88
                   Don't know      3
         Name: DMDEDUC2x, dtype: int64
```

In [67]:

```
dx = df.loc[~df.DMDEDUC2x.isin(["Don't know","Missing"]), :] # eliminate missing
dx = df.groupby(['Agegroup','RIAGENDRx'])["DMDEDUC2x"].value_counts() # this give
# out of DMDEDUC2x
# dx = dx.value_counts()
dx = dx.unstack()
dx = dx.apply(lambda x: x/x.sum(), axis=1) # to get the proportions within each s
print(dx.to_string(float_format="%.3f")) # simply limiting the proportion to 3 de
```

```
DMDEDUC2x           9-11    <9  Don't know  Graduate  HS grad  Missing  Undergr
ad
Agegroup RIAGENDRx
(18, 30] Female    0.072 0.044         NaN     0.256    0.195    0.092      0.3
40
         Male      0.101 0.037         NaN     0.224    0.217    0.132      0.2
89
(30, 40] Female    0.089 0.097         NaN     0.314    0.165      NaN      0.3
35
         Male      0.151 0.103         NaN     0.251    0.227      NaN      0.2
69
(40, 50] Female    0.110 0.106         NaN     0.299    0.173      NaN      0.3
13
         Male      0.142 0.112         NaN     0.274    0.209      NaN      0.2
62
(50, 60] Female    0.117 0.102         NaN     0.245    0.234      NaN      0.3
02
         Male      0.148 0.123         NaN     0.231    0.242      NaN      0.2
56
(60, 70] Female    0.118 0.188         NaN     0.195    0.206      NaN      0.2
93
         Male      0.135 0.151         NaN     0.233    0.231      NaN      0.2
49
(70, 80] Female    0.105 0.224       0.002     0.149    0.239      NaN      0.2
80
         Male      0.112 0.179       0.005     0.236    0.214      NaN      0.2
54
```

In [ ]:

In [ ]:

In [ ]: