Seaborn Library Implementation by Python

Import Libraries

In [74]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os
print(os.listdir("../input"))
import warnings
warnings.filterwarnings("ignore")
```

```
['StudentsPerformance.csv']
```

EDA

In the data discovery analysis, we will firstly recognize and analyze our data using a wide variety of functions in the pandas library.

In [75]:
```
data=pd.read_csv('../input/StudentsPerformance.csv')
```

In [76]:
```
data.head()
```

Out[76]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [77]:
```
data.tail()
```

Out[77]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
995	female	group E	master's degree	standard	completed	88	99	95
996	male	group C	high school	free/reduced	none	62	55	55
997	female	group C	high school	free/reduced	completed	59	71	65
998	female	group D	some college	standard	completed	68	78	77
999	female	group D	some college	free/reduced	none	77	86	86

In [78]:
```
data.sample(5)
```

Out[78]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
229	female	group C	some college	standard	completed	88	95	94
397	female	group C	associate's degree	standard	none	85	89	95
997	female	group C	high school	free/reduced	completed	59	71	65
3	male	group A	associate's degree	free/reduced	none	47	57	44
482	male	group C	some college	free/reduced	none	68	68	61

In [79]:
```
data.sample(frac=0.1)
```

Out[79]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
50	male	group E	some college	standard	none	53	55	48
308	female	group B	associate's degree	free/reduced	none	53	71	67
406	male	group B	associate's degree	standard	completed	65	65	63
143	male	group A	high school	standard	none	57	43	47
677	female	group C	some high school	standard	completed	70	82	76
446	male	group D	some college	free/reduced	none	69	66	60
332	male	group E	associate's degree	standard	completed	62	56	53
290	male	group C	associate's degree	standard	none	76	70	68
810	male	group A	some high school	standard	none	51	31	36
242	female	group D	high school	standard	none	56	52	55
724	male	group B	some college	standard	none	47	43	41
469	male	group C	some college	standard	none	91	74	76
876	male	group D	some college	standard	none	81	82	84
314	female	group C	bachelor's degree	standard	completed	59	64	75
872	male	group B	associate's degree	standard	completed	82	84	78
107	male	group E	associate's degree	standard	completed	66	63	64
474	female	group B	associate's degree	standard	completed	90	90	91
102	female	group D	associate's degree	standard	none	85	91	89
510	male	group D	some college	standard	none	76	71	73
990	male	group E	high school	free/reduced	completed	86	81	75
309	female	group D	high school	free/reduced	none	49	57	52
722	female	group B	some high school	free/reduced	completed	74	90	88
493	female	group C	bachelor's degree	standard	none	81	88	90
733	male	group D	some high school	standard	none	55	47	44
848	female	group C	high school	standard	none	59	72	68
123	male	group D	high school	free/reduced	none	63	57	56
289	male	group E	some high school	standard	completed	77	76	77
824	female	group C	some high school	free/reduced	none	48	58	52
987	male	group E	some high school	standard	completed	81	75	76
866	male	group C	high school	free/reduced	none	59	53	52
...	...	...	...	...	...	...	...	...
941	female	group D	master's degree	standard	none	78	91	96
971	male	group C	some high school	standard	completed	78	72	69
775	female	group B	some high school	free/reduced	none	49	58	55
858	male	group B	high school	standard	completed	52	49	46
216	female	group E	associate's degree	free/reduced	completed	83	86	88
801	male	group C	some high school	standard	completed	76	80	73
755	female	group E	associate's degree	standard	none	84	95	92
816	female	group A	bachelor's degree	standard	none	45	59	64
542	female	group C	associate's degree	standard	none	81	77	79
901	female	group C	master's degree	standard	none	73	78	74
351	male	group E	some college	standard	none	66	57	52
419	male	group E	high school	free/reduced	completed	57	56	54
692	female	group C	bachelor's degree	free/reduced	completed	66	74	81
606	female	group C	associate's degree	standard	none	85	84	82
161	female	group E	some college	free/reduced	completed	75	88	85
359	female	group D	some college	standard	none	80	90	89
388	female	group D	high school	standard	none	62	64	64
807	female	group E	high school	free/reduced	none	41	45	40
287	female	group B	some high school	standard	none	67	89	82
938	male	group D	some college	standard	completed	85	81	85
464	male	group A	bachelor's degree	standard	completed	75	58	62
432	male	group C	high school	standard	none	61	56	55
922	male	group D	high school	standard	none	72	66	66

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
106	female	group D	master's degree	standard	none	87	100	100
95	male	group C	associate's degree	free/reduced	completed	78	81	82
924	male	group D	high school	free/reduced	none	74	70	69
907	female	group D	some college	standard	completed	79	84	91
39	male	group B	associate's degree	free/reduced	none	57	56	57
637	female	group D	some high school	standard	completed	80	92	88
796	male	group D	high school	standard	none	70	70	70

100 rows × 8 columns

In [80]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
gender                         1000 non-null object
race/ethnicity                 1000 non-null object
parental level of education    1000 non-null object
lunch                          1000 non-null object
test preparation course        1000 non-null object
math score                     1000 non-null int64
reading score                  1000 non-null int64
writing score                  1000 non-null int64
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

In [81]:
```
data.iloc[:,0:3].dtypes
```

Out[81]:
```
gender                         object
race/ethnicity                 object
parental level of education    object
dtype: object
```

**

- Count : Shows the total number.
- Mean : Shows the average.
- Std : Standard deviation value
- Min : Minimum value
- %25 : First Quantile
- %50 : Median or Second Quantile
- %75 : Third Quantile
- Max : Maximum value

What is quantile?

- 1,4,5,6,7,11,12,13,14,15,16,17
- The median is the number that is in middle of the sequence. In this case It would be 11
- The lower quartile is the median in between the smallest number and the median etc in between 1 and 11, which is 6
- The upper quartile you find the median between the median and the largest number etc. betweeb 11 and 17,which will be 14 according to the question above.

In [82]:
```
#It is a function that shows the analysis of numerical values.
data.describe()
```

Out[82]:

	math score	reading score	writing score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000

In [83]:
```
#It shows the data types in the data set.
data.dtypes
```

Out[83]:
```
gender                         object
race/ethnicity                 object
parental level of education    object
lunch                          object
test preparation course        object
math score                      int64
reading score                   int64
writing score                   int64
dtype: object
```

In [84]:
```
#It is a function that shows the analysis of proximity values between data.
data.corr()
```

Out[84]:

	math score	reading score	writing score
math score	1.000000	0.817580	0.802642
reading score	0.817580	1.000000	0.954598
writing score	0.802642	0.954598	1.000000

In [85]:
```
data.iloc[:,1:].corr()
```

Out[85]:

	math score	reading score	writing score
math score	1.000000	0.817580	0.802642
reading score	0.817580	1.000000	0.954598
writing score	0.802642	0.954598	1.000000

In [86]:
```
#control data
data.isnull().values.any()
```

Out[86]:
```
False
```

In [87]:
```
#all data control for null values
data.isnull().sum()
```

Out[87]:
```
gender                         0
race/ethnicity                 0
parental level of education    0
lunch                          0
test preparation course        0
math score                     0
reading score                  0
writing score                  0
dtype: int64
```

In [88]:
```
#show columns
for i,col in enumerate(data.columns):
    print(i+1,". column is ",col)
```

```
1 . column is  gender
2 . column is  race/ethnicity
3 . column is  parental level of education
4 . column is  lunch
5 . column is  test preparation course
6 . column is  math score
7 . column is  reading score
8 . column is  writing score
```

In [89]:
```
#rename columns
data.rename(columns=({'gender':'Gender','race/ethnicity':'Race/Ethnicity'
                      ,'parental level of education':'Parental_Level_of_Education'
                      ,'lunch':'Lunch','test preparation course':'Test_Preparation_Course'
                       ,'math score':'Math_Score','reading score':'Reading_Score'
                      ,'writing score':'Writing_Score'}),inplace=True)
```

In [90]:
```
#show columns
for i,col in enumerate(data.columns):
    print(i+1,". column is ",col)
```

```
1 . column is  Gender
2 . column is  Race/Ethnicity
3 . column is  Parental_Level_of_Education
4 . column is  Lunch
5 . column is  Test_Preparation_Course
6 . column is  Math_Score
7 . column is  Reading_Score
8 . column is  Writing_Score
```

In [91]:
```
#show count Gender
data['Gender'].value_counts()
```

Out[91]:
```
female    518
male      482
Name: Gender, dtype: int64
```

In [92]:
```
#show Gender's unique
data['Gender'].unique()
```

Out[92]:
```
array(['female', 'male'], dtype=object)
```

Seaborn

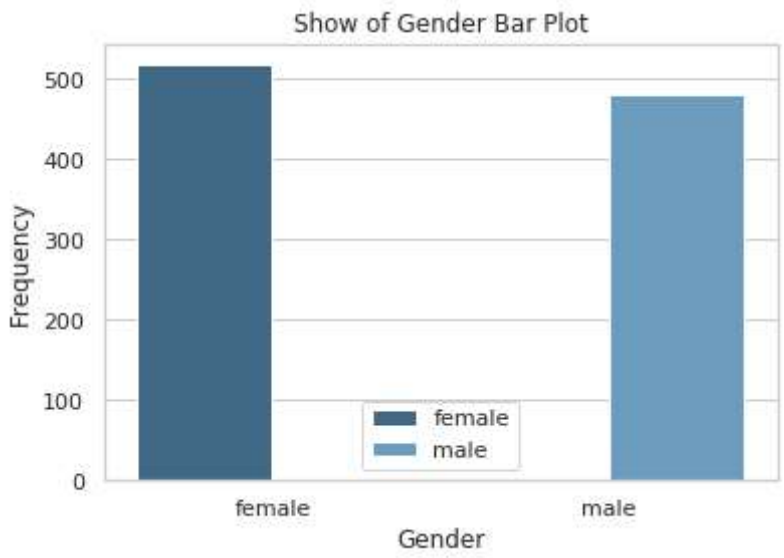
Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. For a brief introduction to the ideas behind the library, you can read the introductory notes. Visit the installation page to see how you can download the package. You can browse the example gallery to see what you can do with seaborn, and then check out the tutorial and API reference to find out how. To see the code or report a bug, please visit the github repository. General support issues are most at home on stackoverflow, where there is a seaborn tag.

Bar Plot

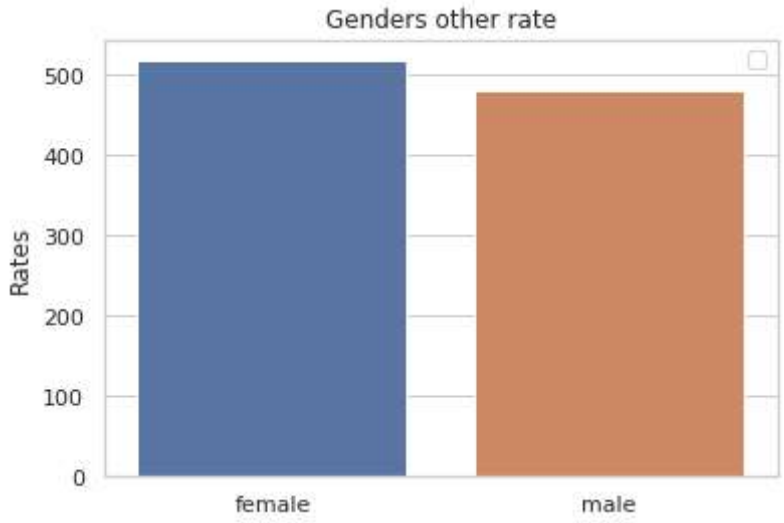
seaborn.barplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, estimator=, ci=95, n_boot=1000, units=None, orient=None, color=None, palette=None, saturation=0.75, errcolor='.26', errwidth=None, capsize=None, dodge=True, ax=None, **kwargs)

- x,y,hue : names of variable in data or vector data
- data : DataFrame,array or list of array,optional
- color :matplotlib color,optional
- palette : palette name,list, or dict,optional
- ax : matplotlib Axes,optional

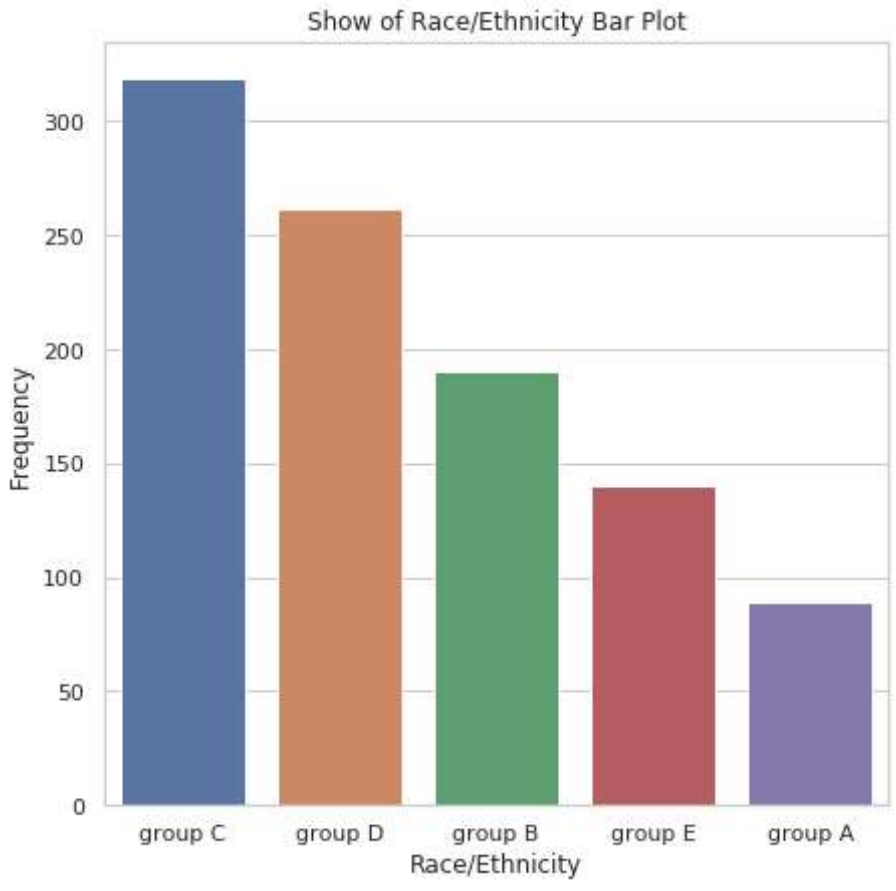
In [93]:
```
#Gender show bar plot
sns.set(style='whitegrid')
ax=sns.barplot(x=data['Gender'].value_counts().index,y=data['Gender'].value_counts().values,palette="Blues_d",hue=['fem
plt.legend(loc=8)
plt.xlabel('Gender')
plt.ylabel('Frequency')
plt.title('Show of Gender Bar Plot')
plt.show()
```



In [94]:
```
sns.barplot(x=data['Gender'].value_counts().index,y=data['Gender'].value_counts().values)
plt.title('Genders other rate')
plt.ylabel('Rates')
plt.legend(loc=0)
plt.show()
```



In [95]:
```
plt.figure(figsize=(7,7))
sns.barplot(x=data['Race/Ethnicity'].value_counts().index,
            y=data['Race/Ethnicity'].value_counts().values)
plt.xlabel('Race/Ethnicity')
plt.ylabel('Frequency')
plt.title('Show of Race/Ethnicity Bar Plot')
plt.show()
```

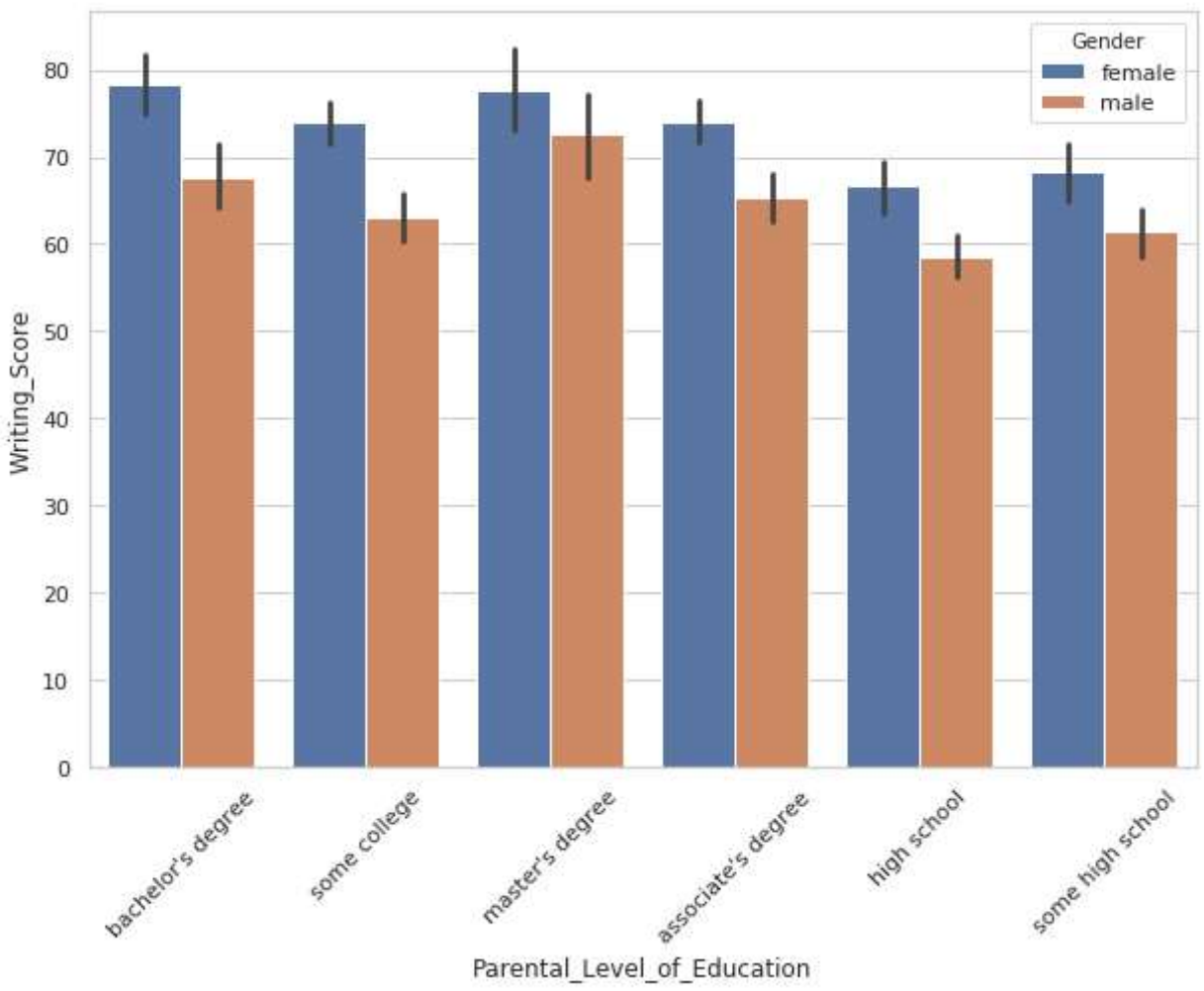


In [96]:
```
data.head()
```

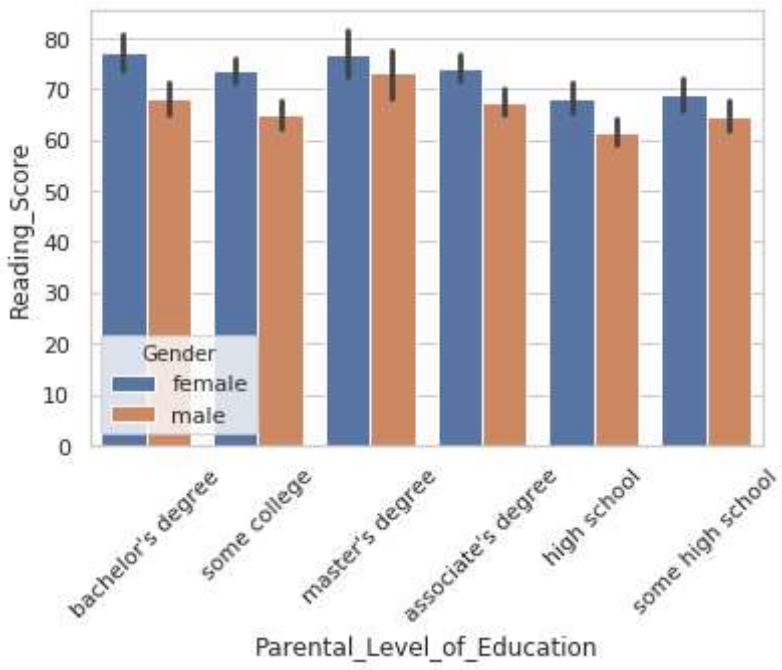
Out[96]:

	Gender	Race/Ethnicity	Parental_Level_of_Education	Lunch	Test_Preparation_Course	Math_Score	Reading_Score	Writing_Score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

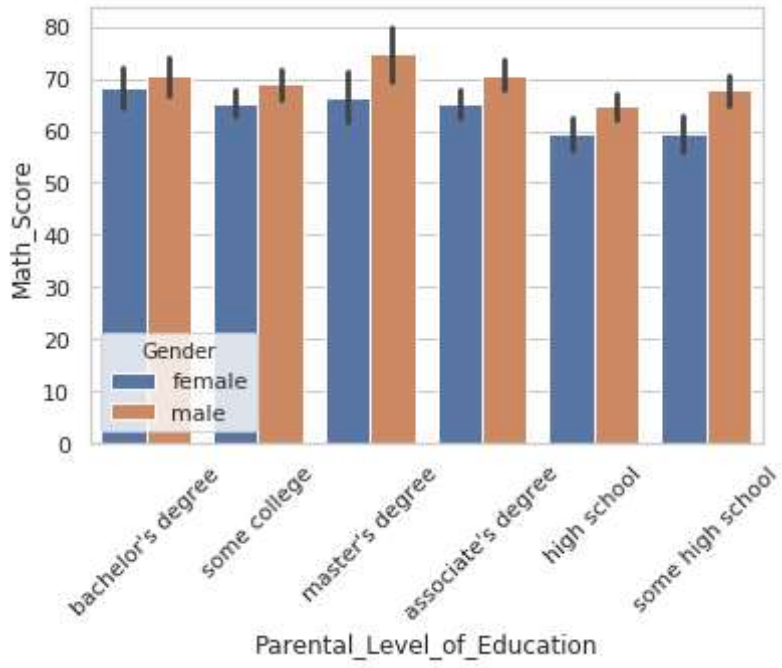
In [97]:
```
plt.figure(figsize=(10,7))
sns.barplot(x = "Parental_Level_of_Education", y = "Writing_Score", hue = "Gender", data = data)
plt.xticks(rotation=45)
plt.show()
```



In [98]:
```
sns.barplot(x = "Parental_Level_of_Education", y = "Reading_Score", hue = "Gender", data = data)
plt.xticks(rotation=45)
plt.show()
```

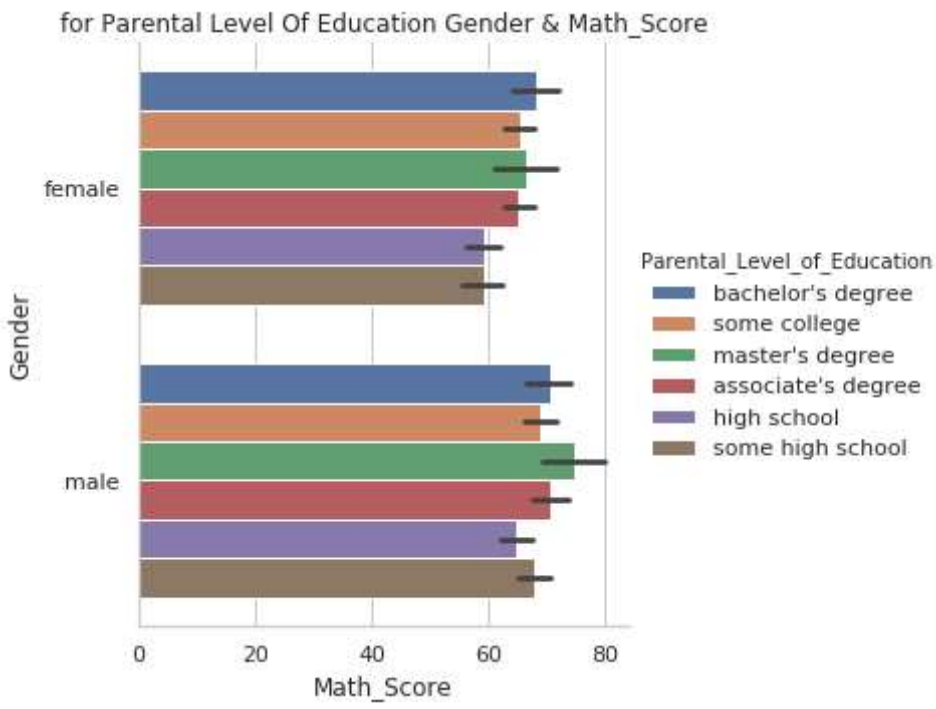


In [99]:
```
sns.barplot(x = "Parental_Level_of_Education", y = "Math_Score", hue = "Gender", data = data)
plt.xticks(rotation=45)
plt.show()
```



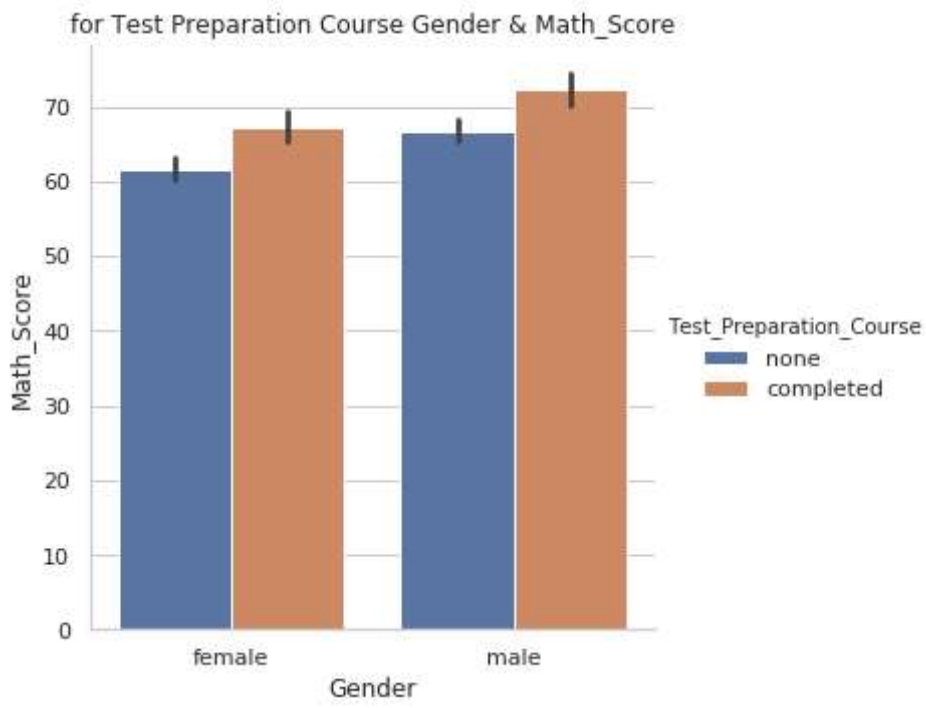
In [100…
```
plt.figure(figsize=(12,7))
sns.catplot(y="Gender", x="Math_Score",
                 hue="Parental_Level_of_Education",
                 data=data, kind="bar")
plt.title('for Parental Level Of Education Gender & Math_Score')
plt.show()
```

```
<Figure size 864x504 with 0 Axes>
```

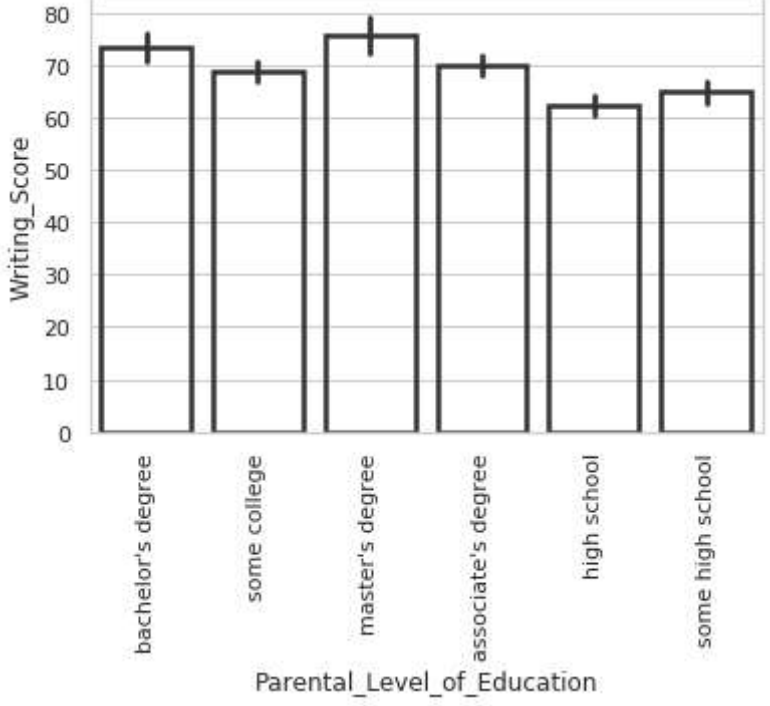


In [101…
```
plt.figure(figsize=(10,10))
sns.catplot(x="Gender", y="Math_Score",
                 hue="Test_Preparation_Course",
                 data=data, kind="bar")
plt.title('for Test Preparation Course Gender & Math_Score')
plt.show()
```

```
<Figure size 720x720 with 0 Axes>
```



In [102…
```
ax = sns.barplot("Parental_Level_of_Education", "Writing_Score", data=data,
                 linewidth=2.5, facecolor=(1, 1, 1, 0),
                 errcolor=".2", edgecolor=".2")
plt.xticks(rotation=90)
plt.show()
```



In [103…
```
# Draw a nested barplot to show survival for class and sex
data.head()
```

Out[103]:

	Gender	Race/Ethnicity	Parental_Level_of_Education	Lunch	Test_Preparation_Course	Math_Score	Reading_Score	Writing_Score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [104…
```
data['Test_Preparation_Course'].unique()
```

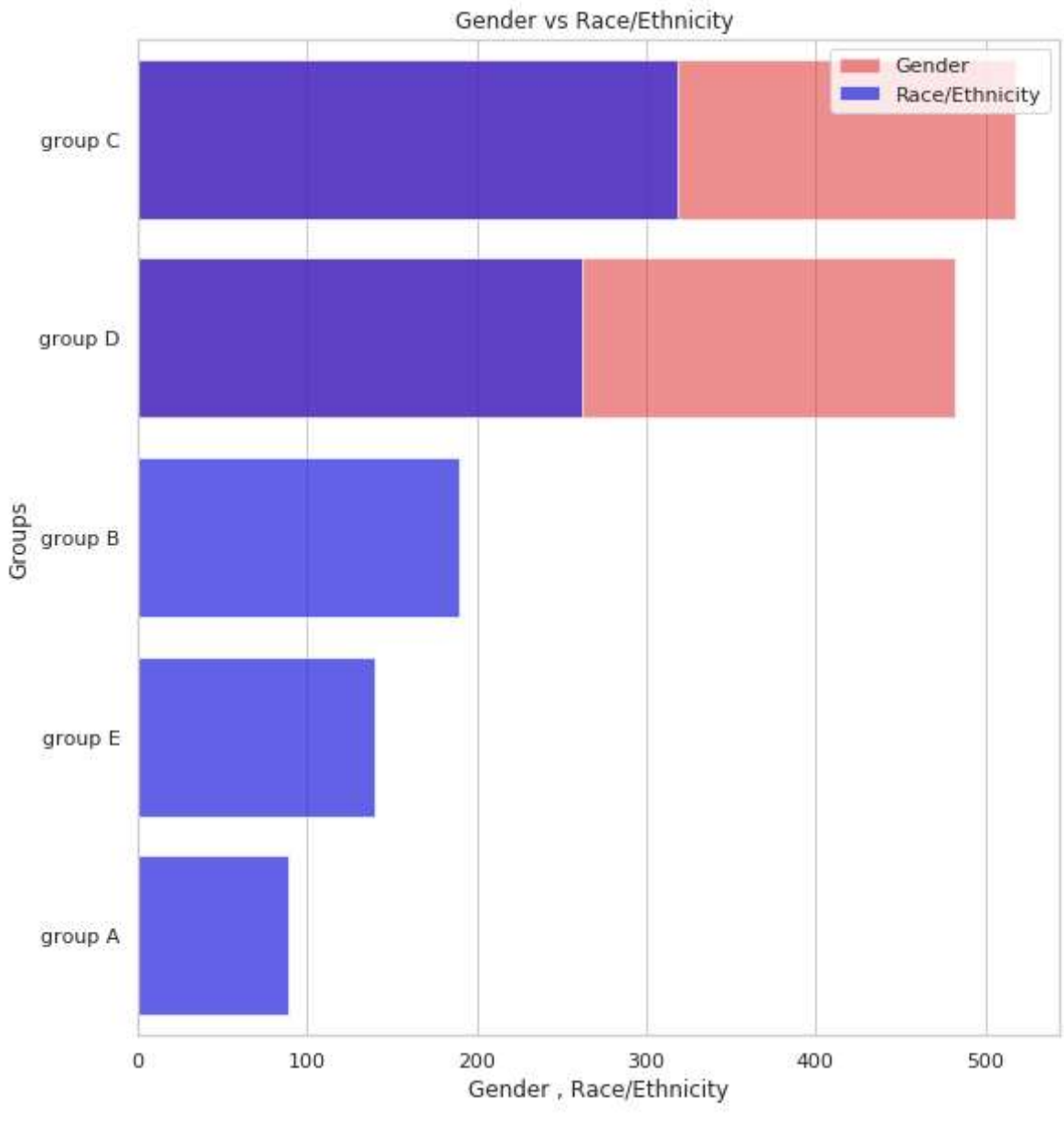
Out[104]:
```
array(['none', 'completed'], dtype=object)
```

In [105…
```
data_lunch_score=data[data['Test_Preparation_Course']=="completed"].groupby(data['Lunch']).Writing_Score.sum()
```

In [106…
```
plt.title("Lunch - Free/reduced & standard")
sns.barplot(x=data_lunch_score.index,y=data_lunch_score.values)
plt.show()
```

In [107…

```
f,ax=plt.subplots(figsize=(9,10))
sns.barplot(x=data['Gender'].value_counts().values,y=data['Gender'].value_counts().index,alpha=0.5,color='red',label='G
sns.barplot(x=data['Race/Ethnicity'].value_counts().values,y=data['Race/Ethnicity'].value_counts().index,color='blue',a
ax.legend(loc='upper right',frameon=True)
ax.set(xlabel='Gender , Race/Ethnicity',ylabel='Groups',title="Gender vs Race/Ethnicity ")
plt.show()
```



Point Plot

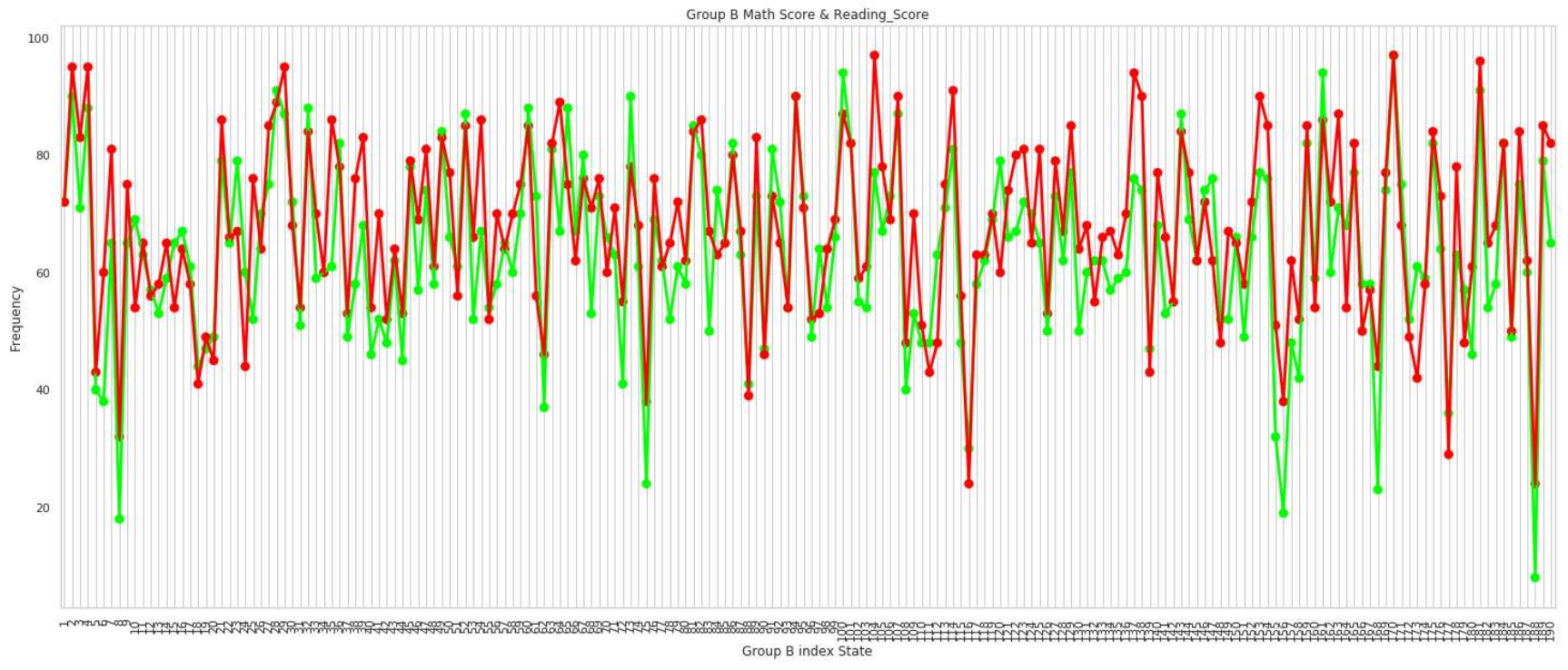
seaborn.pointplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, estimator=, ci=95, n_boot=1000, units=None, markers='o', linestyles='-', dodge=False, join=True, scale=1, orient=None, color=None, palette=None, errwidth=None, capsize=None, ax=None, **kwargs)

- x, y, hue : names of variables in data or vector data, optional
- data : DataFrame, array, or list of arrays, optional
- order, hue_order : lists of strings, optional
- markers : string or list of strings, optional
- linestyles : string or list of strings, optional
- color : matplotlib color, optional
- palette : palette name, list, or dict, optional
- ax : matplotlib Axes, optional

In [108…

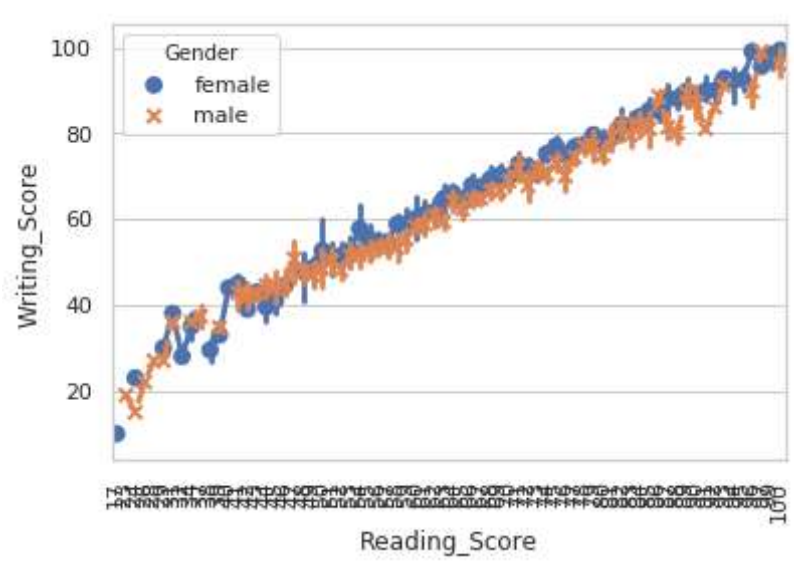
```
#Gender show point plot
data['Race/Ethnicity'].unique()
len(data[(data['Race/Ethnicity']=='group B')].Math_Score)
f,ax1=plt.subplots(figsize=(25,10))
sns.pointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Math_Score,color='lime',alpha=0.8)
sns.pointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Reading_Score,color='red',alpha=0.5)
#sns.pointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Math_Score,color='lime',alpha=0.8)
plt.xlabel('Group B index State')
plt.ylabel('Frequency')
plt.title('Group B Math Score & Reading_Score')
```

```
plt.xticks(rotation=90)
plt.grid()
plt.show()
```



In [109…

```
ax = sns.pointplot(x="Reading_Score", y="Writing_Score", hue="Gender",data=data,markers=["o", "x"],linestyles=["-", "--
plt.xticks(rotation=90)
plt.show()
```



Joint Plot

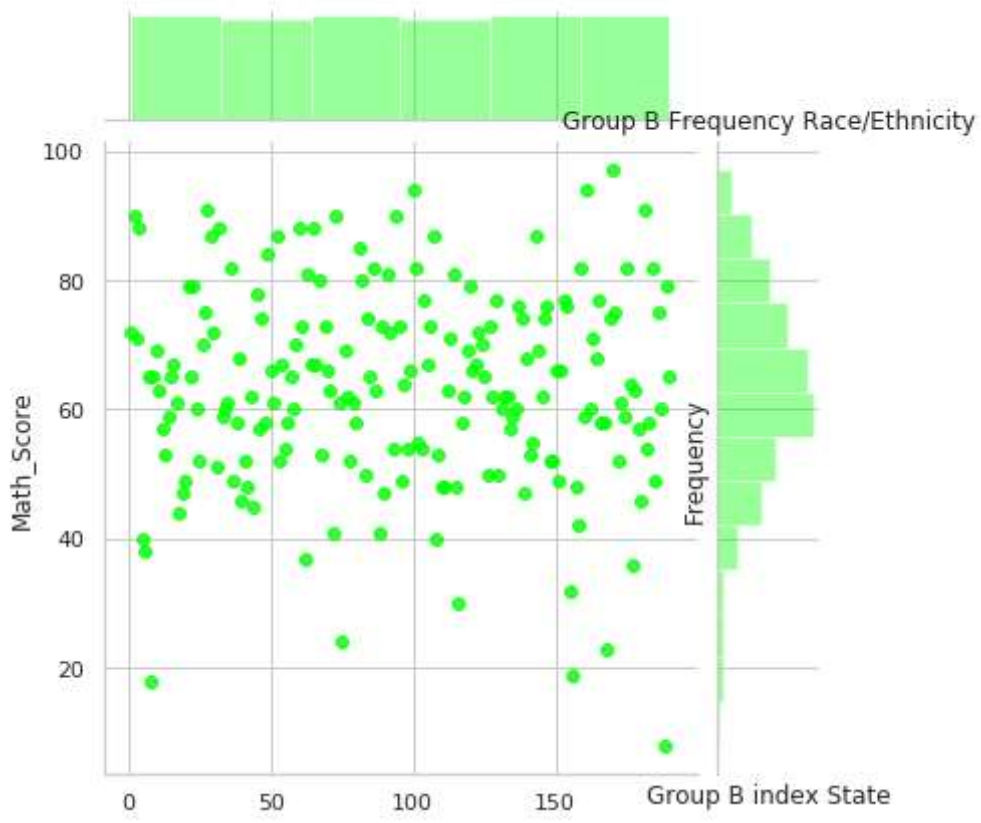
seaborn.jointplot(x, y, data=None, kind='scatter', stat_func=None, color=None, height=6, ratio=5, space=0.2, dropna=True, xlim=None, ylim=None, joint_kws=None, marginal_kws=None, annot_kws=None, **kwargs)

- x, y : strings or vectors
- data : DataFrame, optional
- kind : { “scatter” | “reg” | “resid” | “kde” | “hex” }, optional
- color : matplotlib color, optional
- dropna : bool, optional

In [110…

```
plt.figure(figsize=(10,10))
sns.jointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Math_Score,color='lime',alpha=0.8)
plt.xlabel('Group B index State')
plt.ylabel('Frequency')
plt.title('Group B Frequency Race/Ethnicity')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

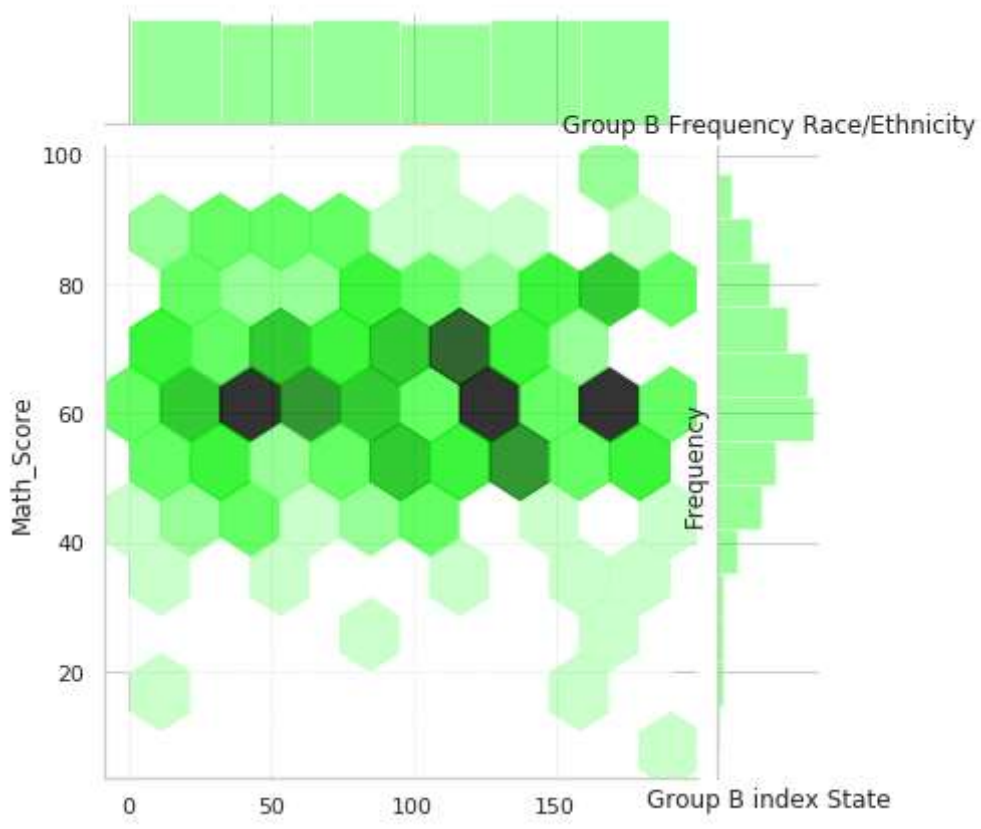
```
<Figure size 720x720 with 0 Axes>
```



In [111…

```
plt.figure(figsize=(10,10))
sns.jointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Math_Score,color='lime',kind='hex',alpha=0
plt.xlabel('Group B index State')
plt.ylabel('Frequency')
plt.title('Group B Frequency Race/Ethnicity')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

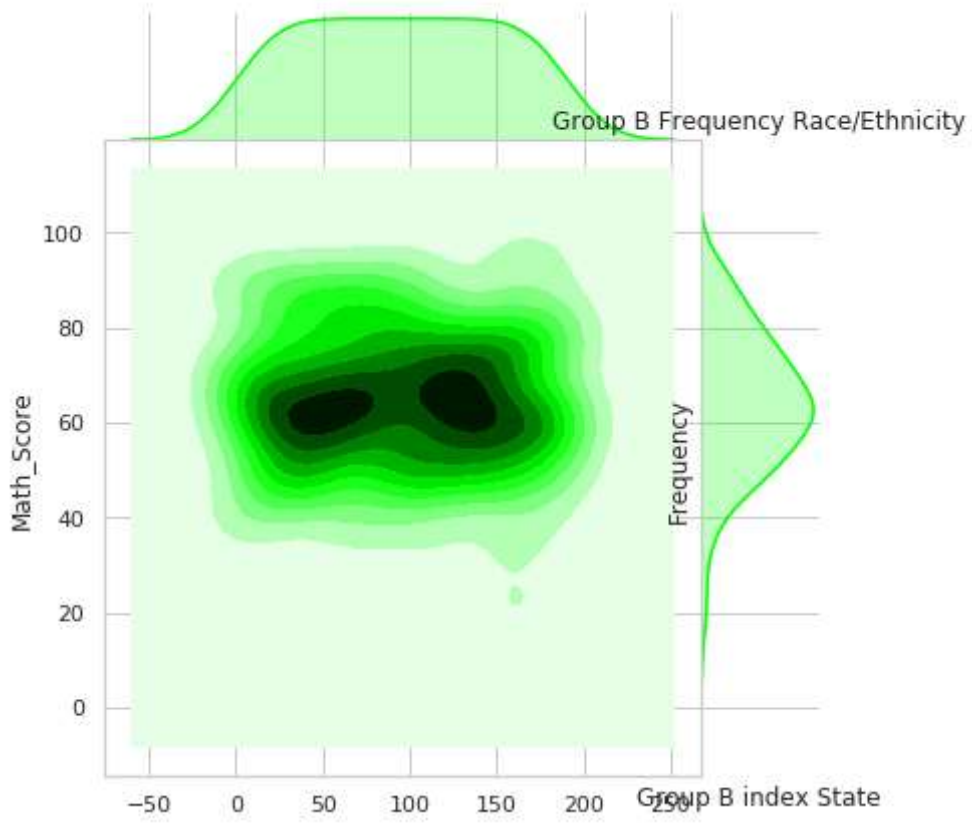
```
<Figure size 720x720 with 0 Axes>
```



In [112…

```
plt.figure(figsize=(10,10))
sns.jointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Math_Score,color='lime',space=0,kind='kde'
plt.xlabel('Group B index State')
plt.ylabel('Frequency')
plt.title('Group B Frequency Race/Ethnicity')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

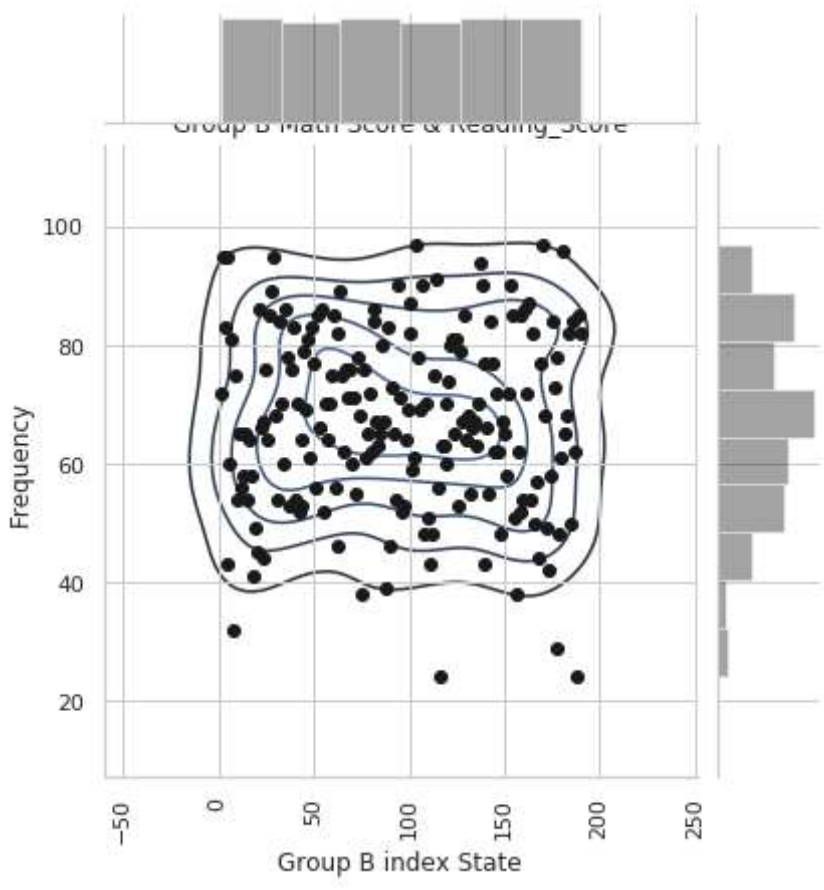
```
<Figure size 720x720 with 0 Axes>
```



In [113…

```
#Gender show point plot
data['Race/Ethnicity'].unique()
len(data[(data['Race/Ethnicity']=='group B')].Math_Score)
plt.figure(figsize=(10,10))
sns.jointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Reading_Score,color='k').plot_joint(sns.kd
plt.xlabel('Group B index State')
plt.ylabel('Frequency')
plt.title('Group B Math Score & Reading_Score')
plt.xticks(rotation=90)
plt.show()
```

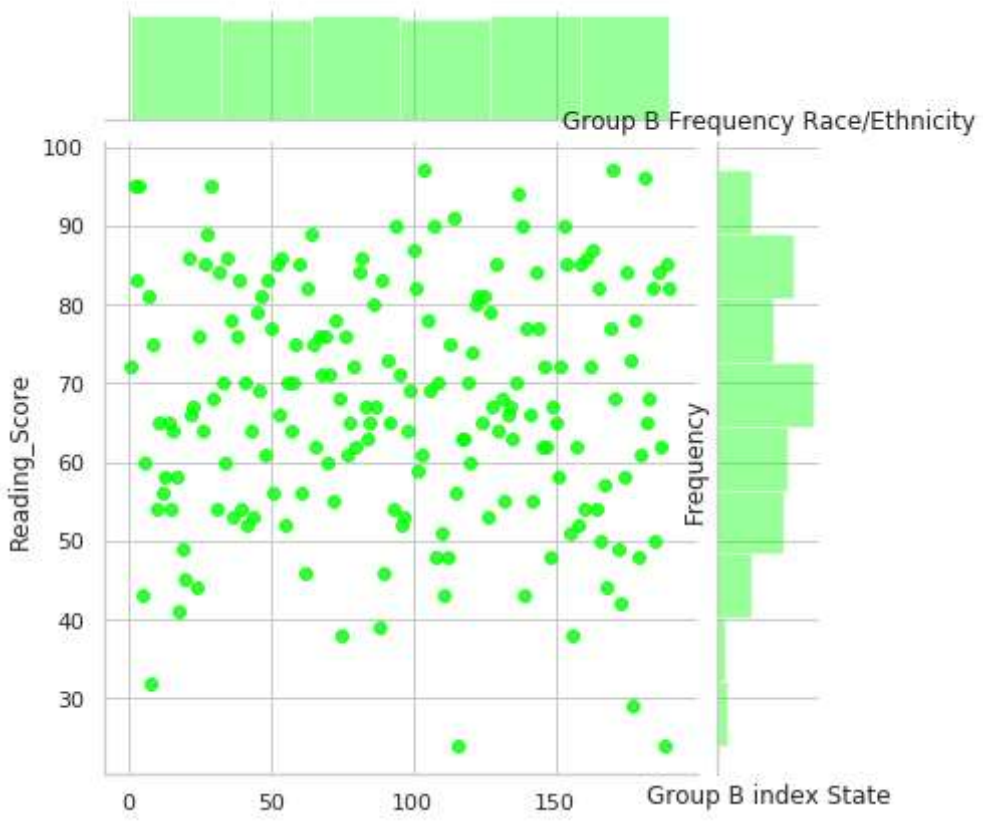
```
<Figure size 720x720 with 0 Axes>
```



In [114…

```
plt.figure(figsize=(10,10))
sns.jointplot(x=np.arange(1,191),y=data[(data['Race/Ethnicity']=='group B')].Reading_Score,color='lime',alpha=0.8)
plt.xlabel('Group B index State')
plt.ylabel('Frequency')
plt.title('Group B Frequency Race/Ethnicity')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

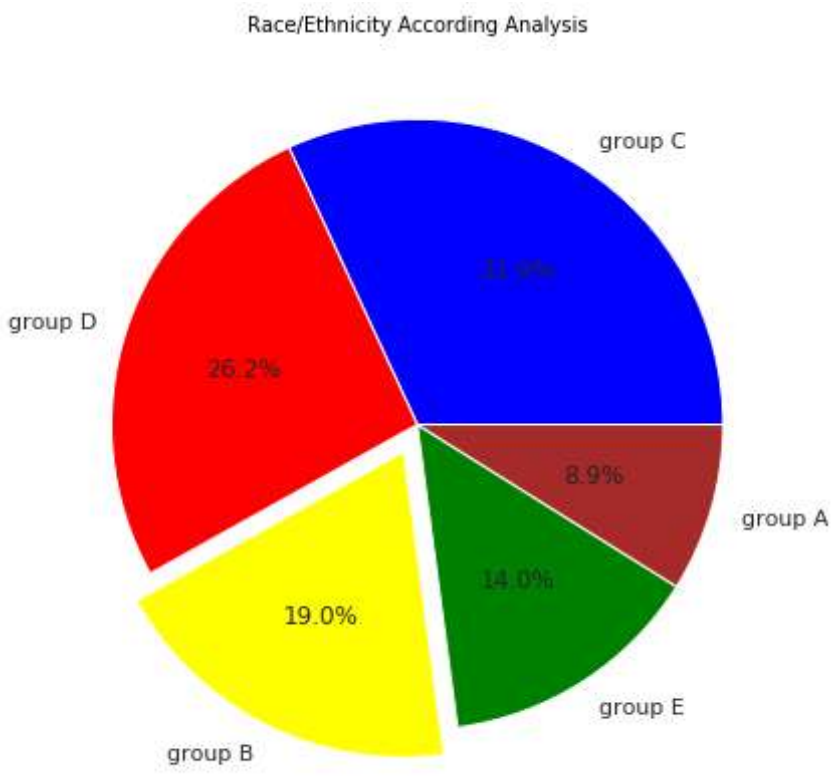
```
<Figure size 720x720 with 0 Axes>
```



Pie Chart

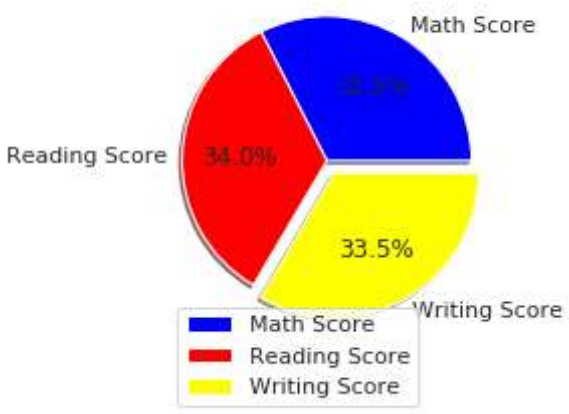
In [115…

```
labels=data['Race/Ethnicity'].value_counts().index
colors=['blue','red','yellow','green','brown']
explode=[0,0,0.1,0,0]
values=data['Race/Ethnicity'].value_counts().values

#visualization
plt.figure(figsize=(7,7))
plt.pie(values,explode=explode,labels=labels,colors=colors,autopct='%1.1f%%')
plt.title('Race/Ethnicity According Analysis',color='black',fontsize=10)
plt.show()
```



In [116…

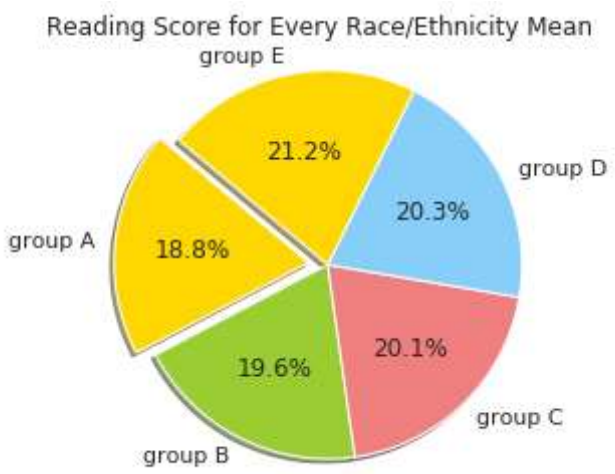
```
plt.figure(figsize=(4,4))
labels=['Math Score', 'Reading Score', 'Writing Score']
colors=['blue','red','yellow']
explode=[0,0,0.1]
values=[data.Math_Score.mean(),data.Reading_Score.mean(),data.Writing_Score.mean()]

plt.pie(values,labels=labels,colors=colors,explode=explode,autopct='%1.1f%%',shadow=True)
plt.legend(['Math Score', 'Reading Score', 'Writing Score'] , loc=3)
plt.axis('equal')
plt.tight_layout()
plt.show()
```



In [117…
```
data.groupby('Race/Ethnicity')['Reading_Score'].mean()
```

Out[117]:
```
Race/Ethnicity
group A    64.674157
group B    67.352632
group C    69.103448
group D    70.030534
group E    73.028571
Name: Reading_Score, dtype: float64
```

In [118…
```
# Data to plot
labels = 'group A', 'group B', 'group C', 'group D','group E'
sizes = data.groupby('Race/Ethnicity')['Reading_Score'].mean().values
colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue']
explode = (0.1, 0, 0, 0,0)  # explode 1st slice

# Plot
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=140)
plt.title('Reading Score for Every Race/Ethnicity Mean')
plt.axis('equal')
plt.show()
```

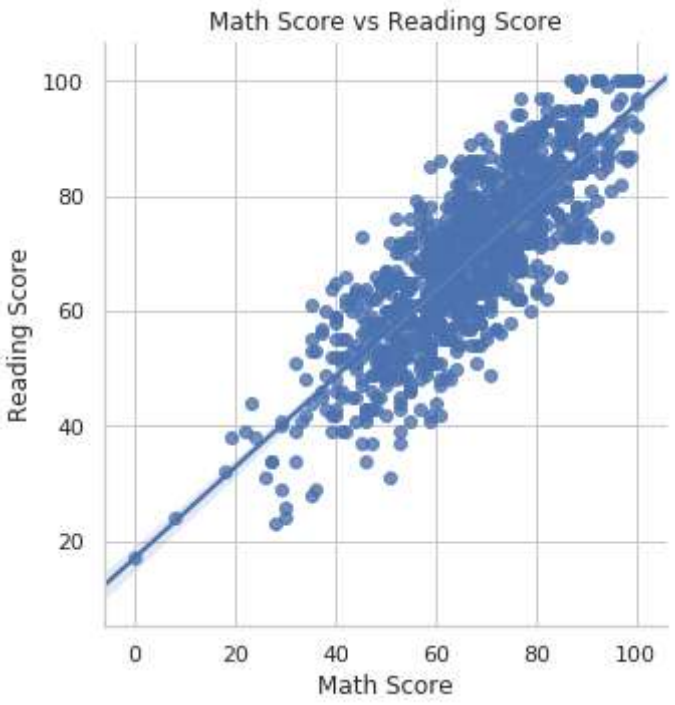


Lm Plot

seaborn.lmplot(x, y, data, hue=None, col=None, row=None, palette=None, col_wrap=None, height=5, aspect=1, markers='o', sharex=True, sharey=True, hue_order=None, col_order=None, row_order=None, legend=True, legend_out=True, x_estimator=None, x_bins=None, x_ci='ci', scatter=True, fit_reg=True, ci=95, n_boot=1000, units=None, order=1, logistic=False, lowess=False, robust=False, logx=False, x_partial=None, y_partial=None, truncate=False, x_jitter=None, y_jitter=None, scatter_kws=None, line_kws=None, size=None)

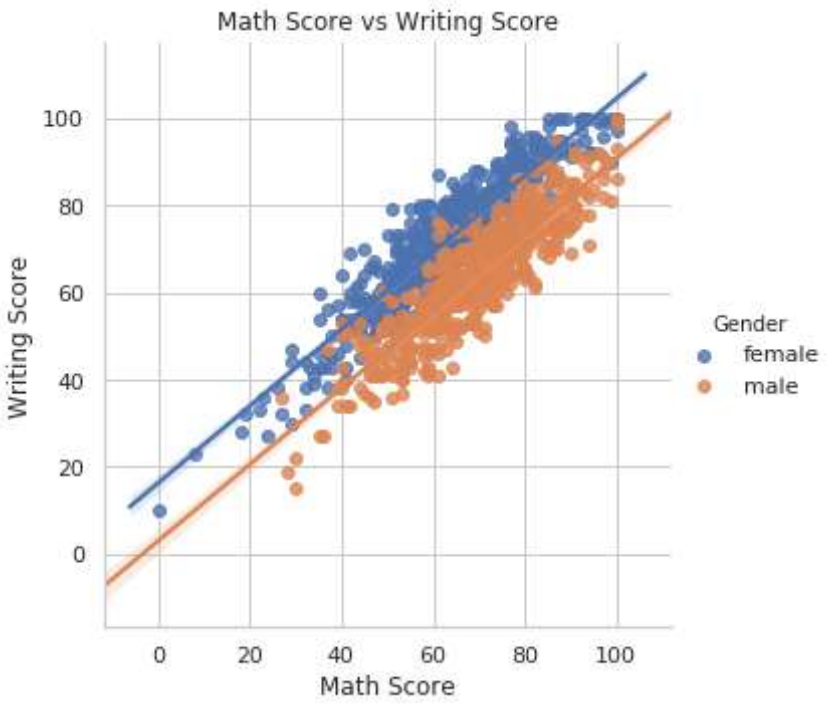
- x, y : strings, optional
- data : DataFrame
- hue, col, row : strings
- palette : palette name, list, or dict, optional
- markers : matplotlib marker code or list of marker codes, optional
- legend : bool, optional
- scatter : bool, optional

In [119…
```
sns.lmplot(x='Math_Score',y='Reading_Score',data=data)
plt.xlabel('Math Score')
plt.ylabel('Reading Score')
plt.title('Math Score vs Reading Score')
plt.show()
```



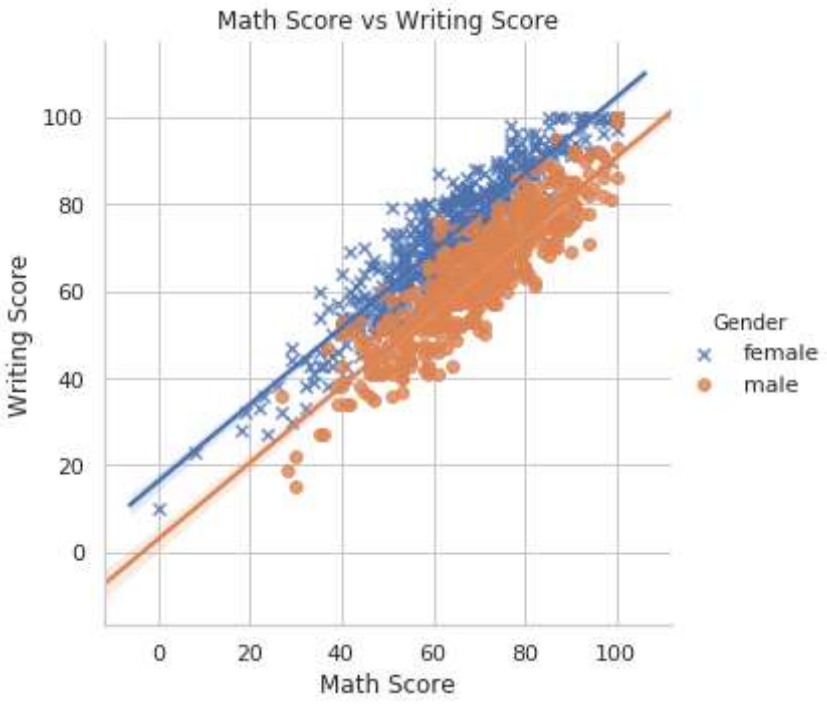
In [120…

```
sns.lmplot(x='Math_Score',y='Writing_Score',hue='Gender',data=data)
plt.xlabel('Math Score')
plt.ylabel('Writing Score')
plt.title('Math Score vs Writing Score')
plt.show()
```



In [121…

```
sns.lmplot(x='Math_Score',y='Writing_Score',hue='Gender',data=data,markers=['x','o'])
plt.xlabel('Math Score')
plt.ylabel('Writing Score')
plt.title('Math Score vs Writing Score')
plt.show()
```



KDE Plot

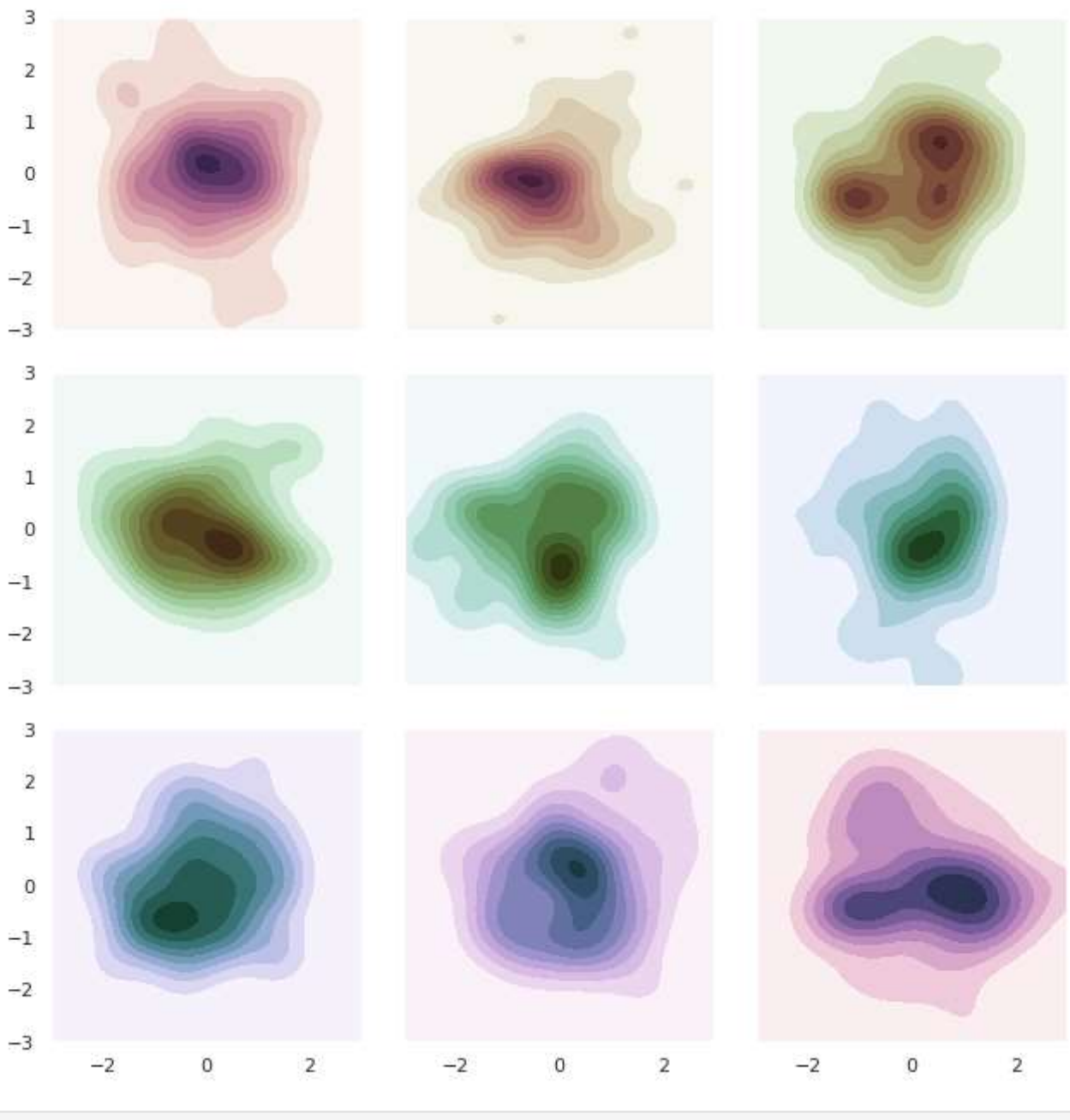
seaborn.kdeplot(data, data2=None, shade=False, vertical=False, kernel='gau', bw='scott', gridsize=100, cut=3, clip=None, legend=True, cumulative=False, shade_lowest=True, cbar=False, cbar_ax=None, cbar_kws=None, ax=None, **kwargs)

- data : 1d array-like
- data2: 1d array-like, optional
- shade : bool, optional
- vertical : bool, optional
- kernel : {‘gau’ | ‘cos’ | ‘biw’ | ‘epa’ | ‘tri’ | ‘triw’ }, optional

- cut : scalar, optional
- legend : bool, optional
- ax : matplotlib axes, optional

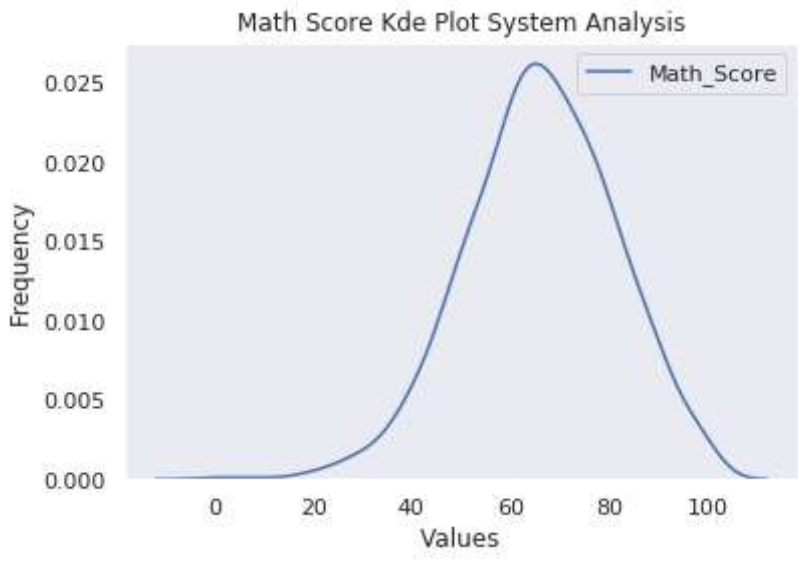
In [122…

```
sns.set(style="dark")
rs = np.random.RandomState(50)

# Set up the matplotlib figure
f, axes = plt.subplots(3, 3, figsize=(9, 9), sharex=True, sharey=True)

# Rotate the starting point around the cubehelix hue circle
for ax, s in zip(axes.flat, np.linspace(0, 3, 10)):

    # Create a cubehelix colormap to use with kdeplot
    cmap = sns.cubehelix_palette(start=s, light=1, as_cmap=True)

    # Generate and plot a random bivariate dataset
    x, y = rs.randn(2, 50)
    sns.kdeplot(x, y, cmap=cmap, shade=True, cut=5, ax=ax)
    ax.set(xlim=(-3, 3), ylim=(-3, 3))

f.tight_layout()
plt.show()
```



In [123…

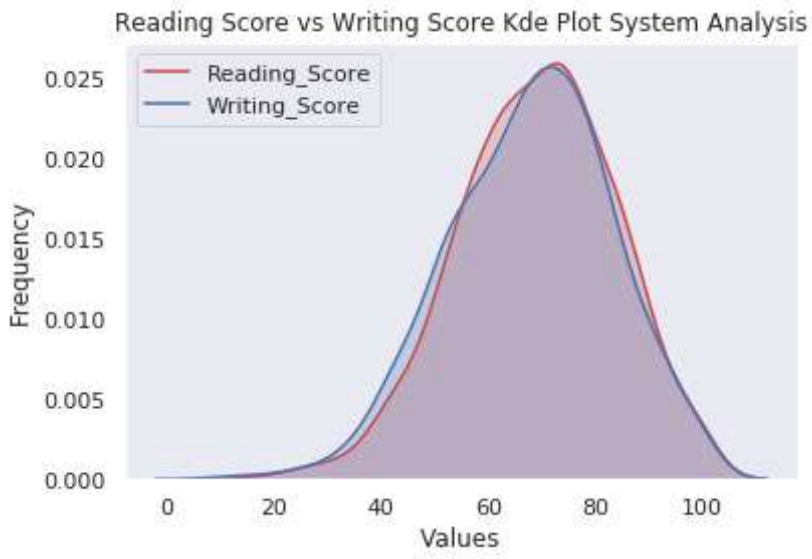
```
sns.kdeplot(data['Math_Score'])
plt.xlabel('Values')
plt.ylabel('Frequency')
plt.title('Math Score Kde Plot System Analysis')
plt.show()
```



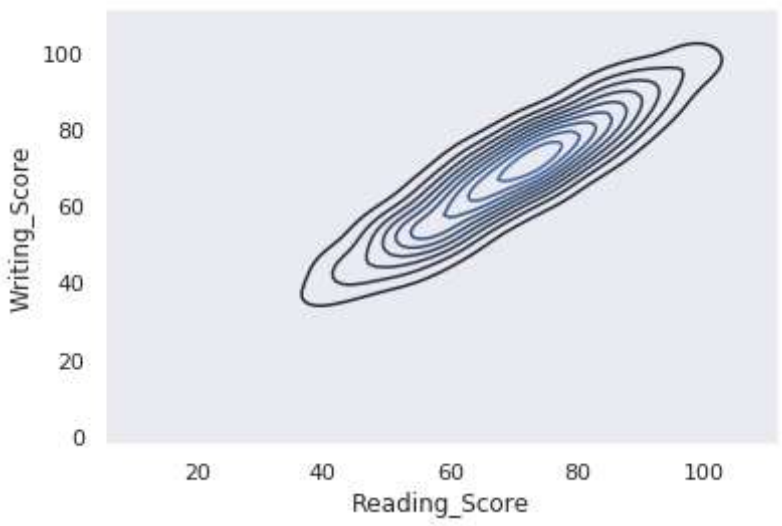
In [124…

```
sns.kdeplot(data['Reading_Score'],shade=True,color='r')
sns.kdeplot(data['Writing_Score'],shade=True,color='b')
plt.xlabel('Values')
plt.ylabel('Frequency')
```

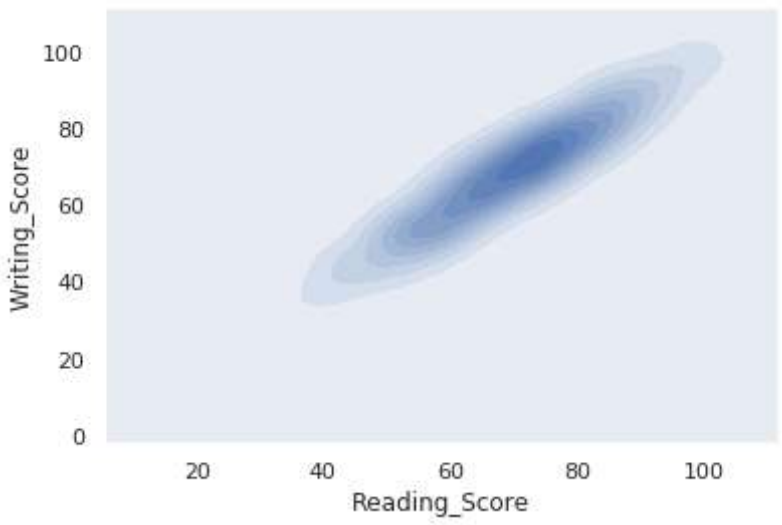
```
plt.title('Reading Score vs Writing Score Kde Plot System Analysis')
plt.show()
```



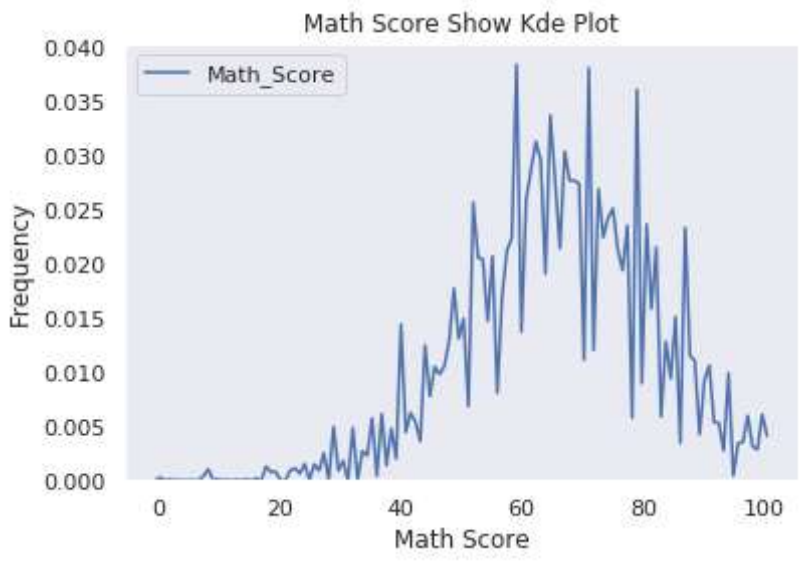
In [125…
```
sns.kdeplot(data['Reading_Score'],data['Writing_Score'])
plt.show()
```



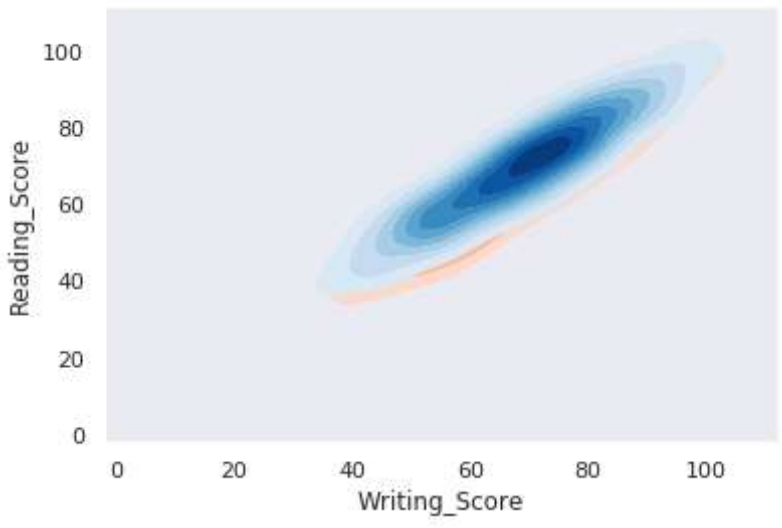
In [126…
```
sns.kdeplot(data['Reading_Score'],data['Writing_Score'],shade=True)
plt.show()
```



In [127…
```
sns.kdeplot(data['Math_Score'],bw=.15)
plt.xlabel('Math Score')
plt.ylabel('Frequency')
plt.title('Math Score Show Kde Plot')
plt.show()
```



In [128…
```
sns.kdeplot(data['Reading_Score'],data['Writing_Score'],cmap='Reds',shade=True,shade_lowest=False)
sns.kdeplot(data['Writing_Score'],data['Reading_Score'],cmap='Blues',shade=True,shade_lowest=False)
plt.show()
```



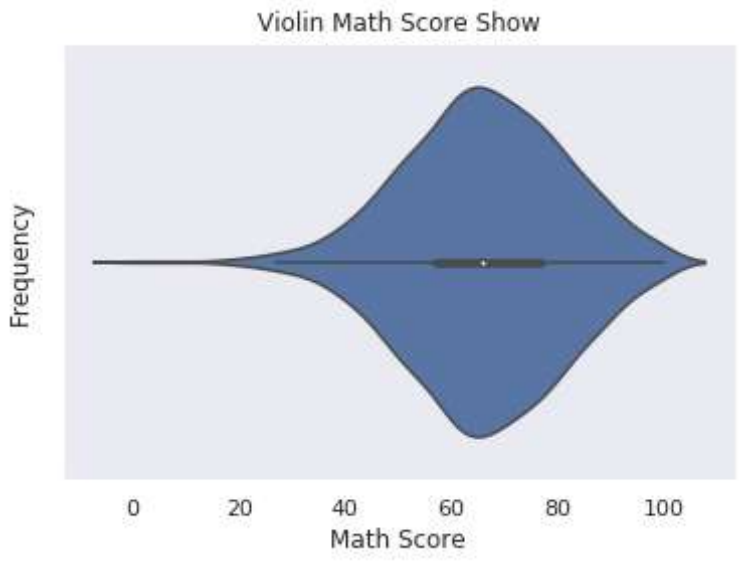
Violin Plot

seaborn.violinplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, bw='scott', cut=2, scale='area', scale_hue=True, gridsize=100, width=0.8, inner='box', split=False, dodge=True, orient=None, linewidth=None, color=None, palette=None, saturation=0.75, ax=None, **kwargs)

- x, y, hue : names of variables in data or vector data, optional
- data : DataFrame, array, or list of arrays, optional
- scale : {“area”, “count”, “width”}, optional
- linewidth : float, optional
- color : matplotlib color, optional
- palette : palette name, list, or dict, optional
- ax : matplotlib Axes, optional
- saturation : float, optional

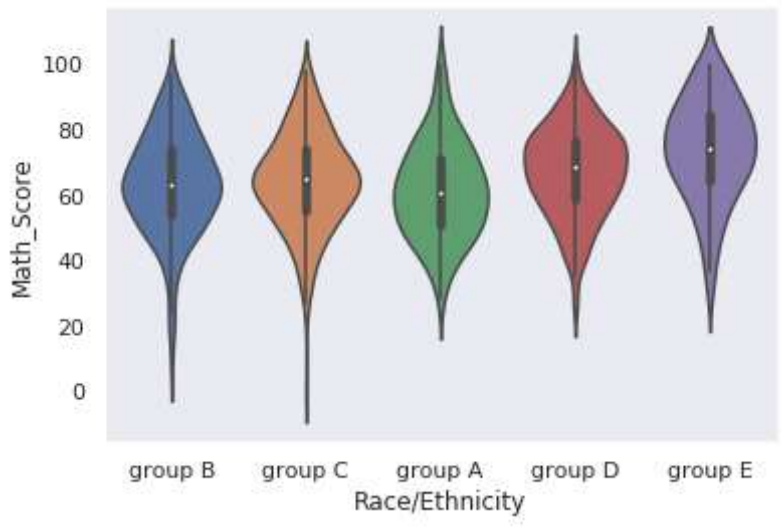
In [129…

```
sns.violinplot(data['Math_Score'])
plt.xlabel('Math Score')
plt.ylabel('Frequency')
plt.title('Violin Math Score Show')
plt.show()
```



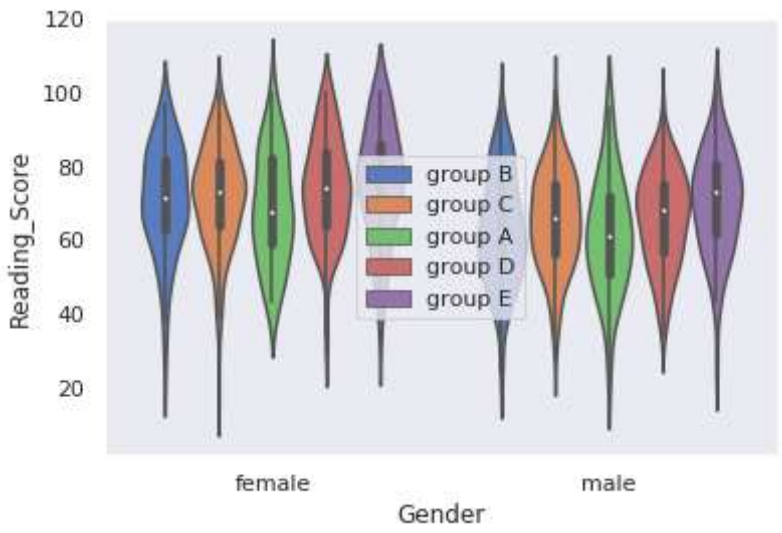
In [130…

```
sns.violinplot(x=data['Race/Ethnicity'],y=data['Math_Score'])
plt.show()
```

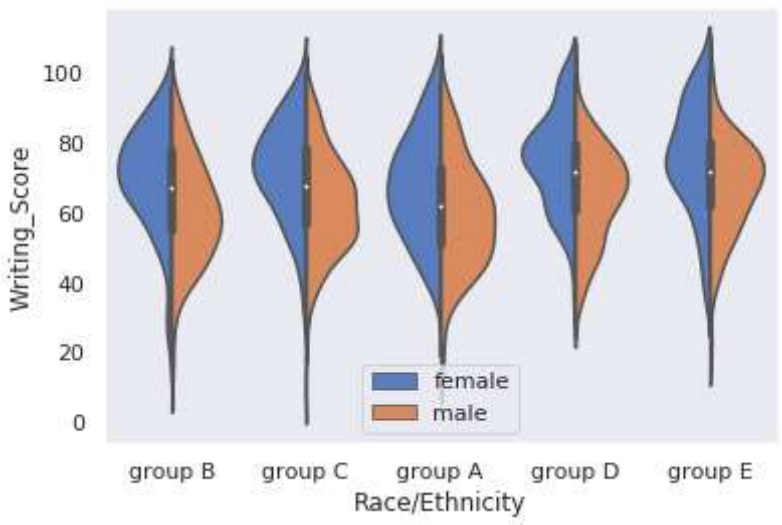


In [131…

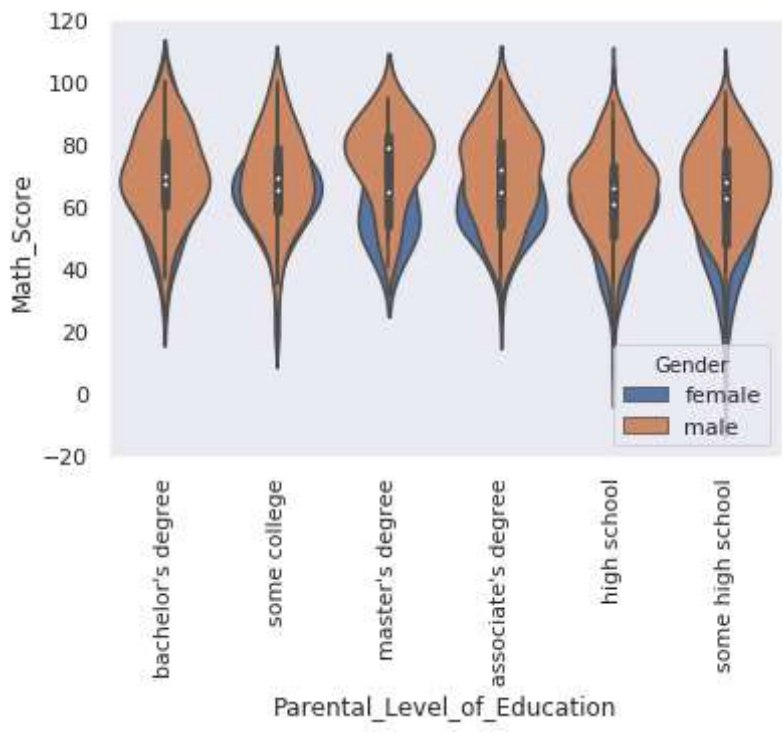
```
sns.violinplot(data['Gender'],y=data['Reading_Score'],hue=data['Race/Ethnicity'],palette='muted')
plt.legend(loc=10)
plt.show()
```



In [132…
```
sns.violinplot(data['Race/Ethnicity'],data['Writing_Score'],
               hue=data['Gender'],palette='muted',split=True)
plt.legend(loc=8)
plt.show()
```



In [133…
```
sns.violinplot(data['Parental_Level_of_Education'],data['Math_Score'],hue=data['Gender'],dodge=False)
plt.xticks(rotation=90)
plt.show()
```

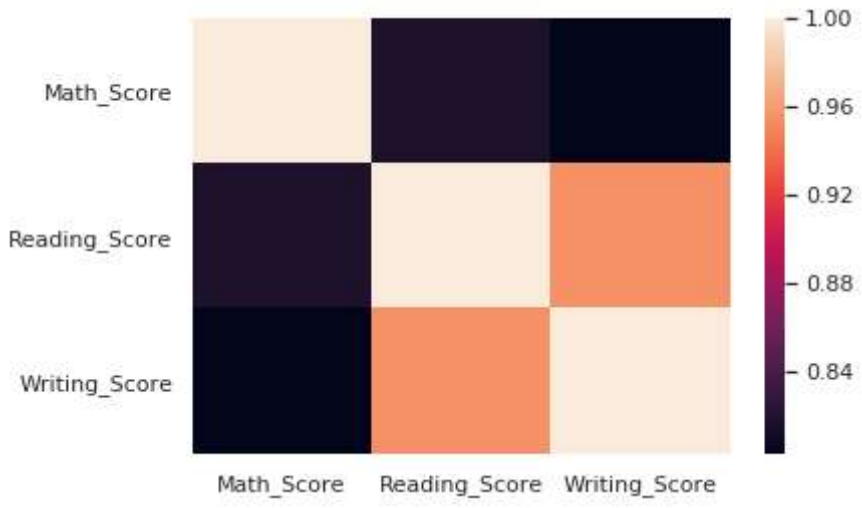


Heatmap Plot

seaborn.heatmap(data, vmin=None, vmax=None, cmap=None, center=None, robust=False, annot=None, fmt='.2g', annot_kws=None, linewidths=0, linecolor='white', cbar=True, cbar_kws=None, cbar_ax=None, square=False, xticklabels='auto', yticklabels='auto', mask=None, ax=None, **kwargs)

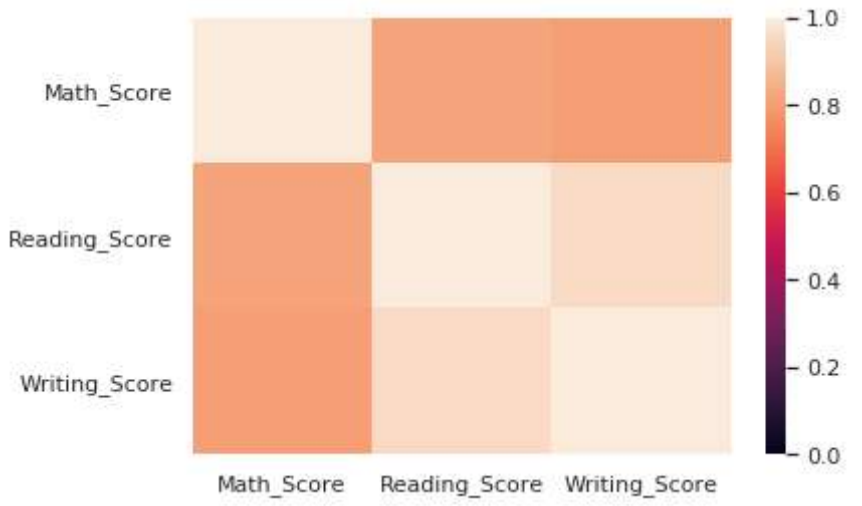
- data : rectangular dataset
- vmin, vmax : floats, optional
- cmap : matplotlib colormap name or object, or list of colors, optional
- annot : bool or rectangular dataset, optional
- fmt : string, optional
- linewidths : float, optional
- ax : matplotlib Axes, optional

In [134…
```
sns.heatmap(data.corr())
plt.show()
```



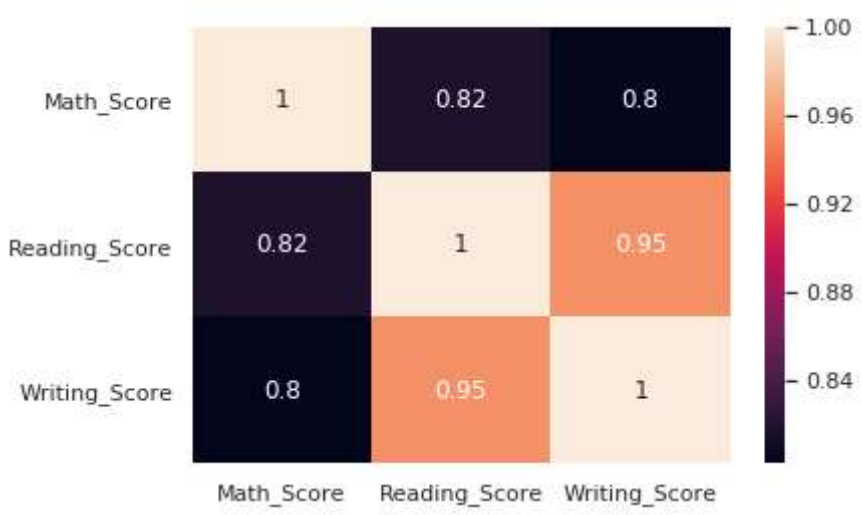
In [135…

```
sns.heatmap(data.corr(),vmin=0,vmax=1)
plt.show()
```



In [136…

```
sns.heatmap(data.corr(),annot=True)
plt.show()
```



In [137…

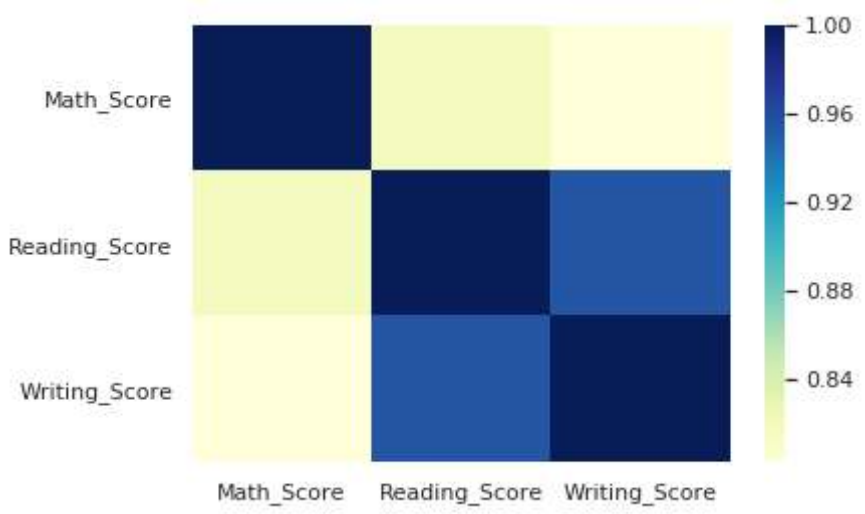
```
# Compute the correlation matrix
corr = data.corr()

# Generate a mask for the upper triangle
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
```

Out[137]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f84d9c67898>
```

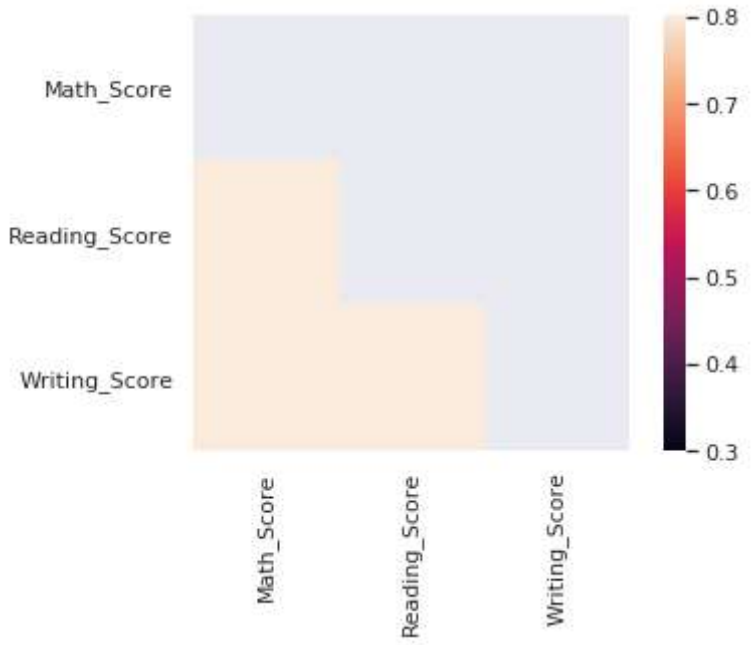
In [138…

```
sns.heatmap(data.corr(),cmap='YlGnBu')
plt.show()
```



In [139…

```
sns.axes_style("white")
mask = np.zeros_like(data.corr())
mask[np.triu_indices_from(mask)] = True
sns.heatmap(data.corr(),vmax=.3,mask=mask,square=True)
plt.show()
```



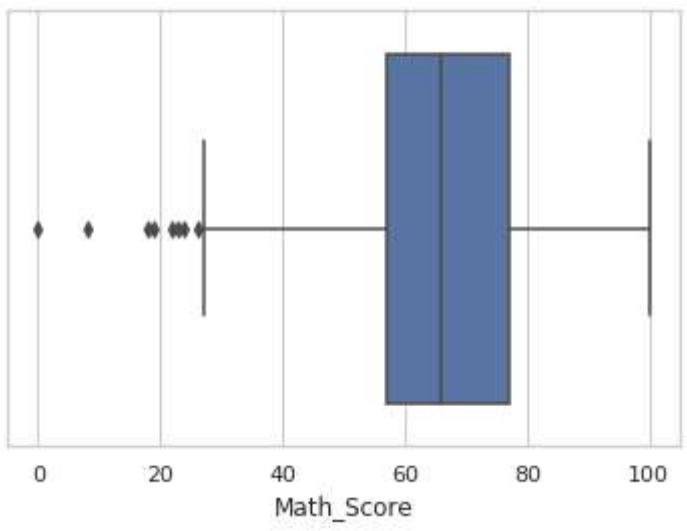
Box Plot

seaborn.boxplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, orient=None, color=None, palette=None, saturation=0.75, width=0.8, dodge=True, fliersize=5, linewidth=None, whis=1.5, notch=False, ax=None, **kwargs)

- x, y, hue : names of variables in data or vector data, optional
- data : DataFrame, array, or list of arrays, optional
- color : matplotlib color, optional
- dodge : bool, optional
- linewidth : float, optional

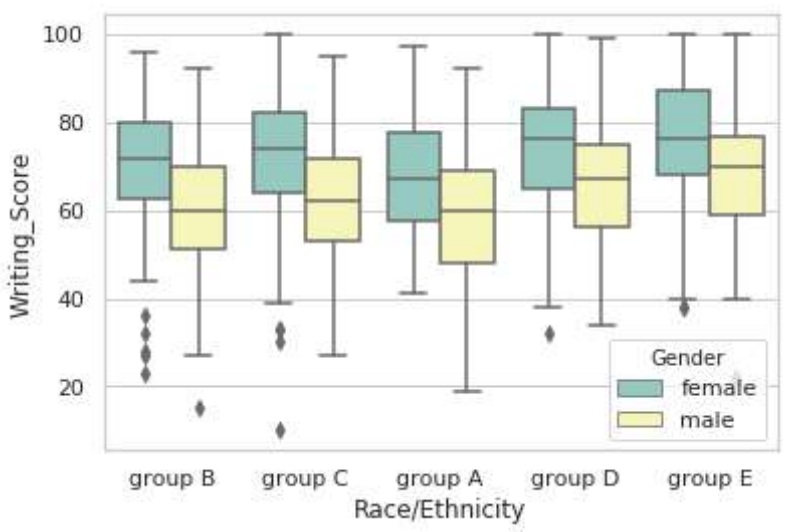
- ax : matplotlib Axes, optional

In [140…
```
sns.set(style='whitegrid')
sns.boxplot(data['Math_Score'])
plt.show()
```

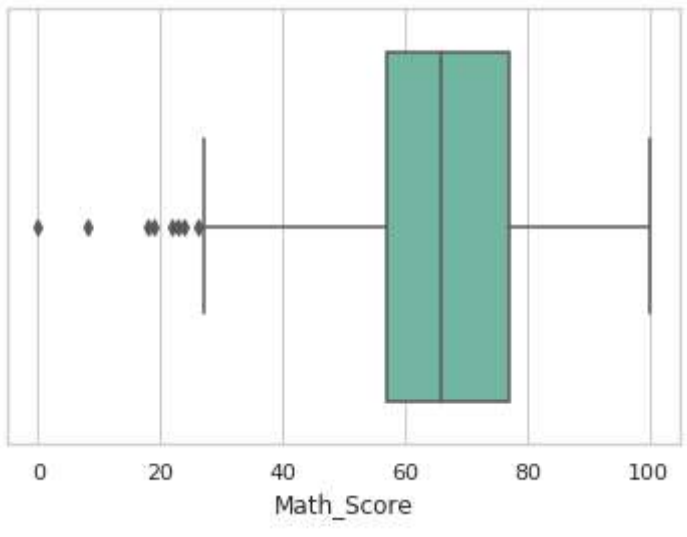


In [141…
```
sns.boxplot(x=data['Gender'],y=data['Math_Score'])
plt.show()
```

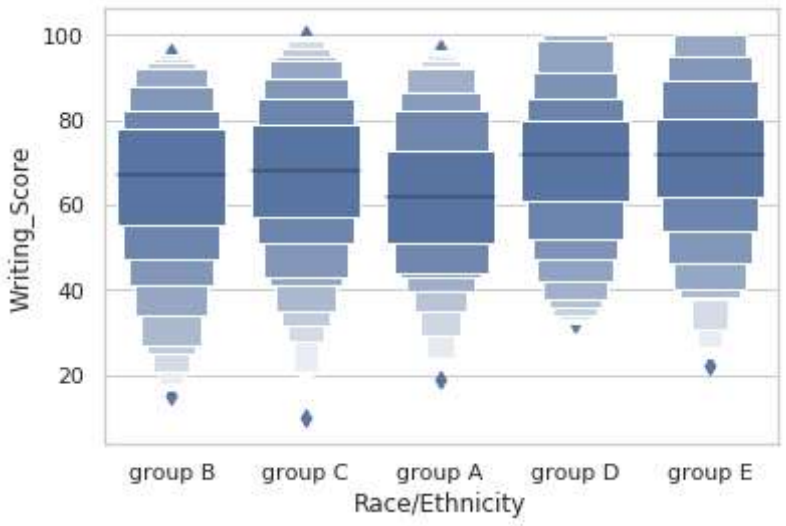

In [142…
```
sns.boxplot(x=data['Race/Ethnicity'],y=data['Writing_Score'],hue=data['Gender'],palette="Set3")
plt.show()
```



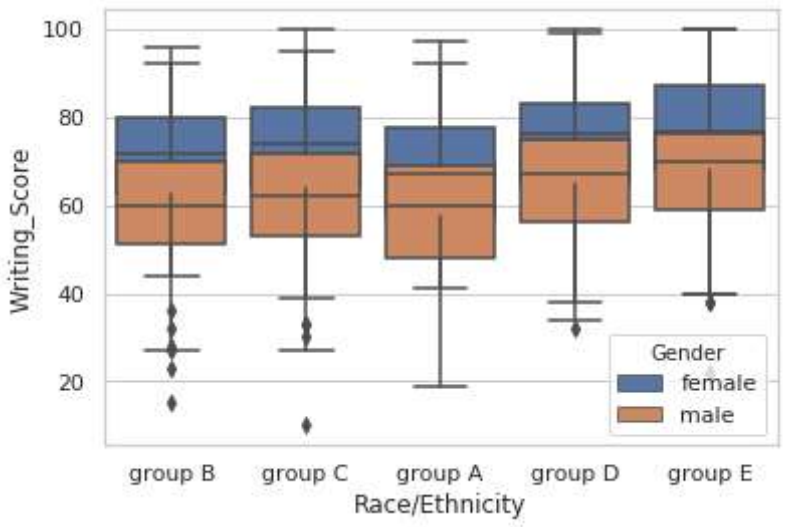
In [143…
```
sns.boxplot(data['Math_Score'],orient='h',palette='Set2')
plt.show()
```



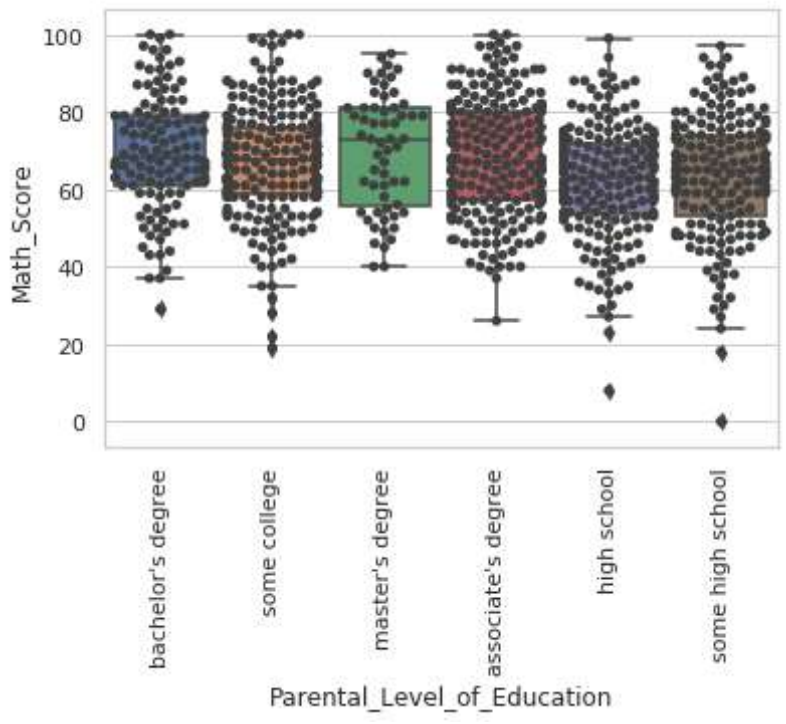
In [144…
```
sns.boxenplot(x="Race/Ethnicity", y="Writing_Score",
              color="b",
              scale="linear", data=data)
plt.show()
```



In [145…
```
sns.boxplot(x=data['Race/Ethnicity'],y=data['Writing_Score'],hue=data['Gender'],dodge=False)
plt.show()
```



In [146…
```
sns.boxplot(x=data['Parental_Level_of_Education'],y=data['Math_Score'])
plt.xticks(rotation=90)
sns.swarmplot(x=data['Parental_Level_of_Education'],y=data['Math_Score'],color=".25")
plt.xticks(rotation=90)
plt.show()
```

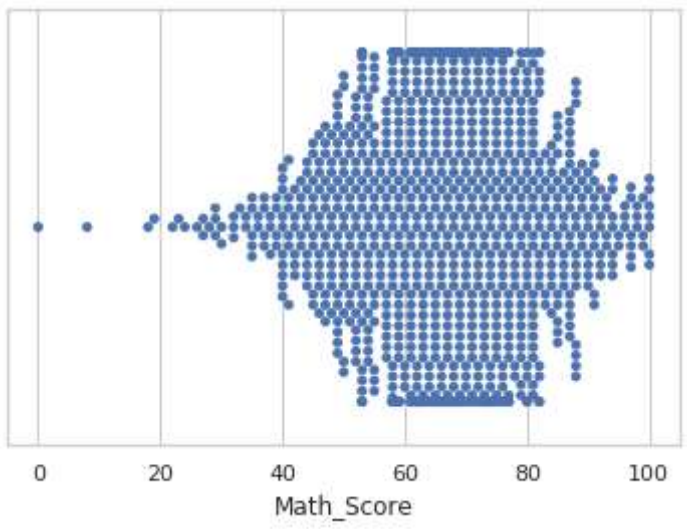


Swarm Plot

seaborn.swarmplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, dodge=False, orient=None, color=None, palette=None, size=5, edgecolor='gray', linewidth=0, ax=None, **kwargs)

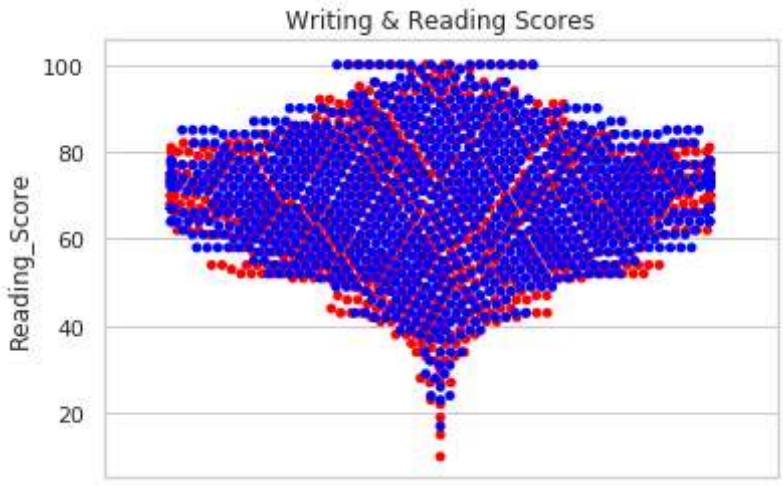
- x, y, hue : names of variables in data or vector data, optional
- data : DataFrame, array, or list of arrays, optional
- dodge : bool, optional
- palette : palette name, list, or dict, optional
- size : float, optional
- ax : matplotlib Axes, optional
- linewidth : float, optional
- edgecolor : matplotlib color, “gray” is special-cased, optional

In [147…
```
sns.set(style='whitegrid')
sns.swarmplot(x=data['Math_Score'])
plt.show()
```



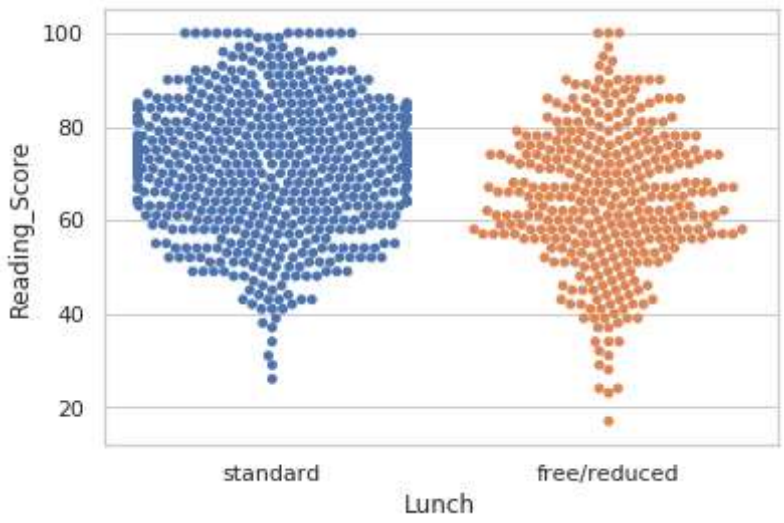
In [148…

```
sns.set(style="whitegrid")

sns.swarmplot(y=data["Writing_Score"],color='red')
sns.swarmplot(y=data["Reading_Score"],color='blue')
plt.title('Writing & Reading Scores')
plt.show()
```



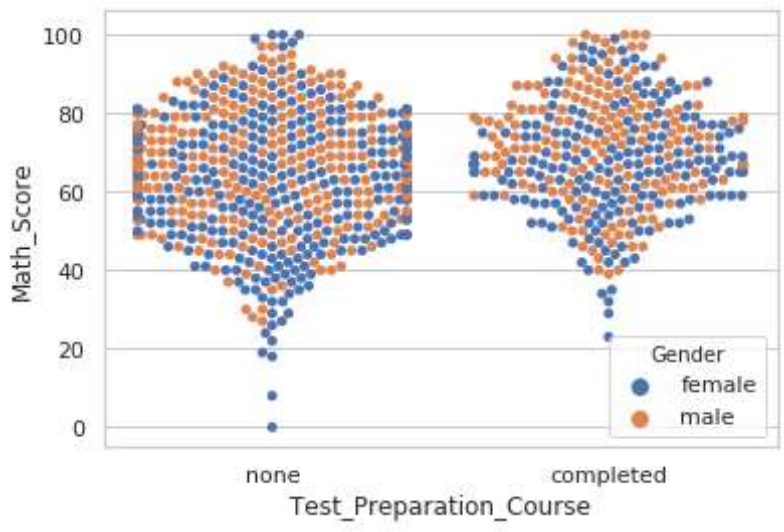
In [149…

```
sns.swarmplot(x=data['Lunch'],y=data['Reading_Score'])
plt.show()
```



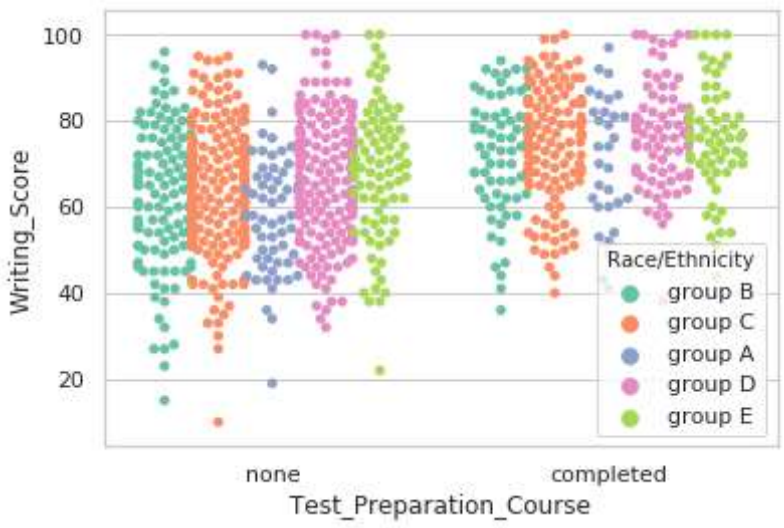
In [150…

```
sns.swarmplot(x=data['Test_Preparation_Course'],y=data['Math_Score'],hue=data['Gender'])
plt.show()
```

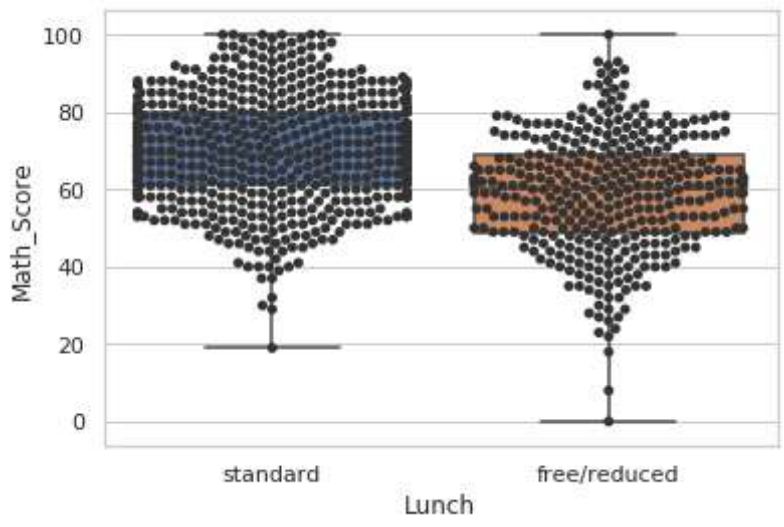


In [151…

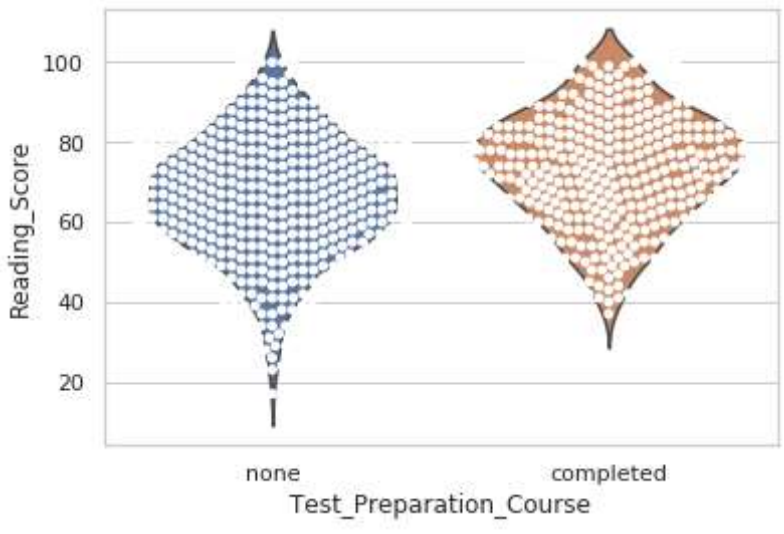
```
sns.swarmplot(x=data['Test_Preparation_Course'],y=data['Writing_Score'],hue=data['Race/Ethnicity'],palette='Set2',dodge
plt.show()
```



In [152…
```
sns.boxplot(x=data['Lunch'],y=data['Math_Score'],whis=np.inf)
sns.swarmplot(x=data['Lunch'],y=data['Math_Score'],color='.2')
plt.show()
```



In [153…
```
sns.violinplot(x=data['Test_Preparation_Course'],y=data['Reading_Score'],inner=None)
sns.swarmplot(x=data['Test_Preparation_Course'],y=data['Reading_Score'],color='white',edgecolor='gray')
plt.show()
```

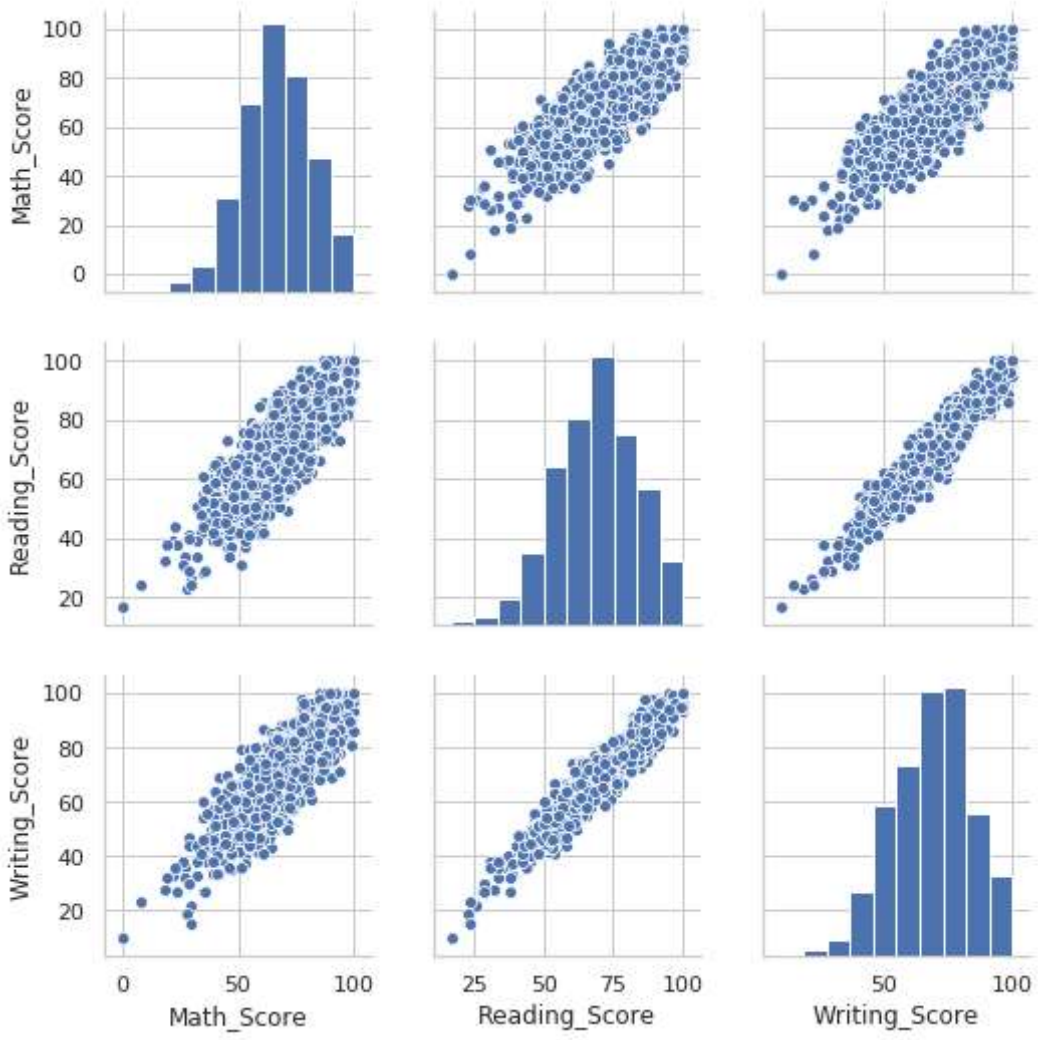


Pair Plot

seaborn.pairplot(data, hue=None, hue_order=None, palette=None, vars=None, x_vars=None, y_vars=None, kind='scatter', diag_kind='auto', markers=None, height=2.5, aspect=1, dropna=True, plot_kws=None, diag_kws=None, grid_kws=None, size=None)

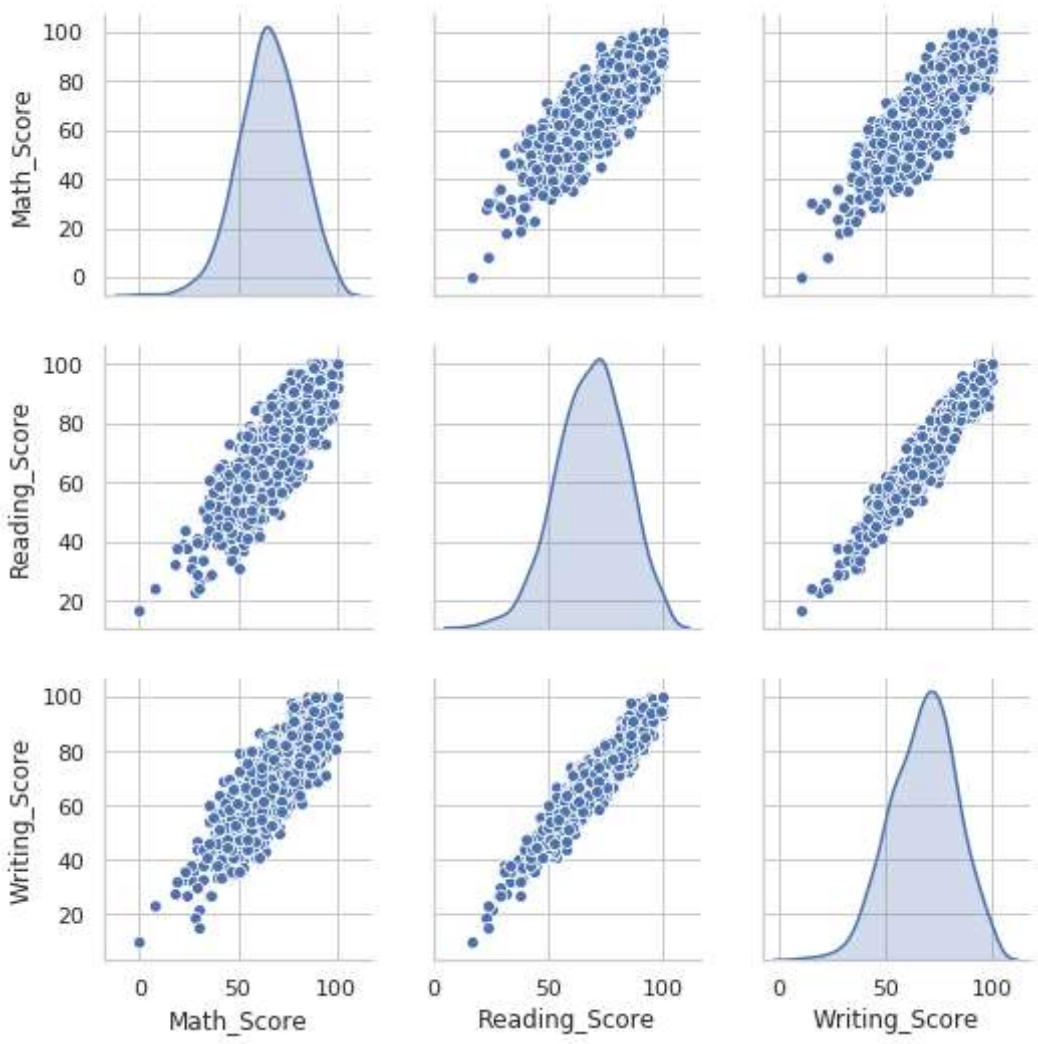
- data : DataFrame
- hue : string (variable name), optional
- hue_order : list of strings
- palette : dict or seaborn color palette
- markers : single matplotlib marker code or list, optional
- dropna : boolean, optional
- height : scalar, optional

In [154…
```
sns.pairplot(data)
plt.show()
```



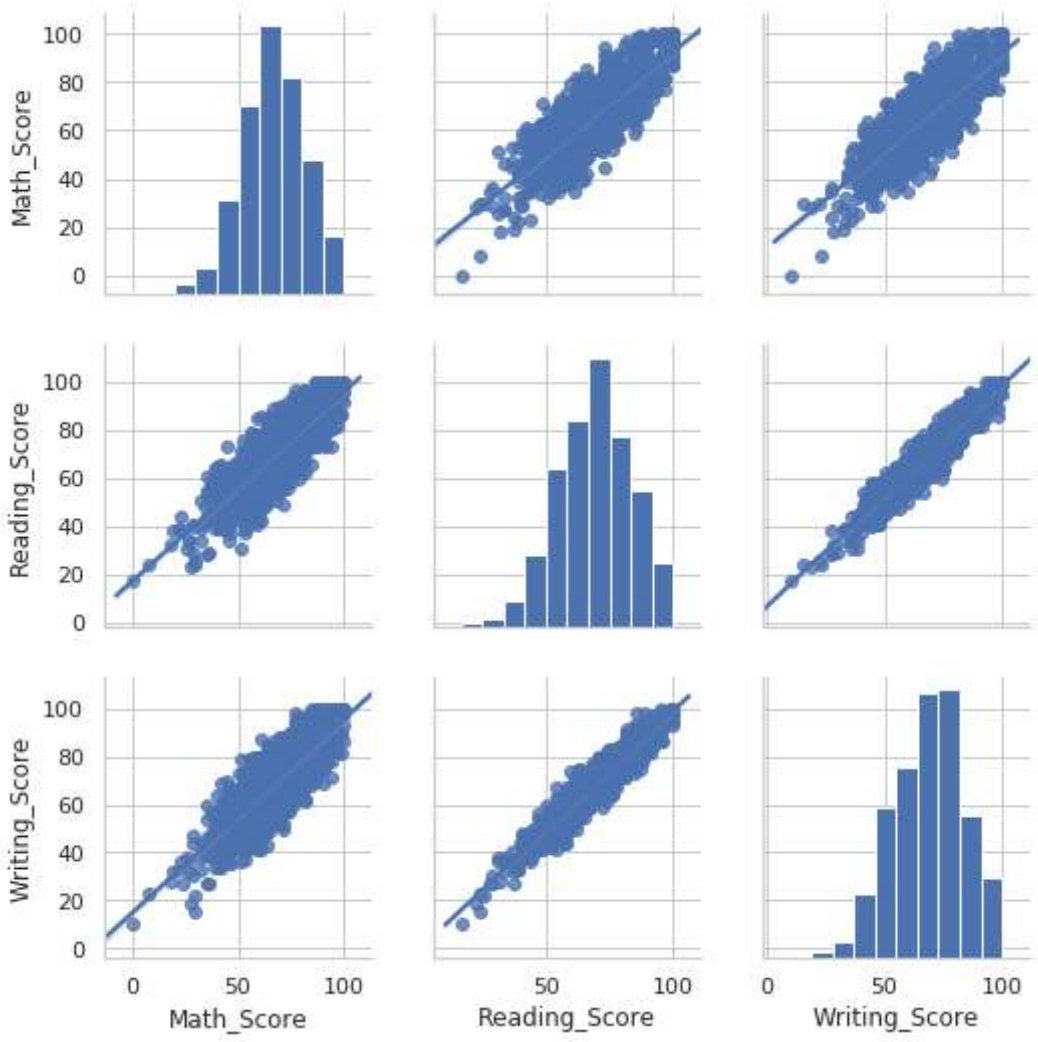
In [155…

```
sns.pairplot(data,diag_kind='kde')
plt.show()
```



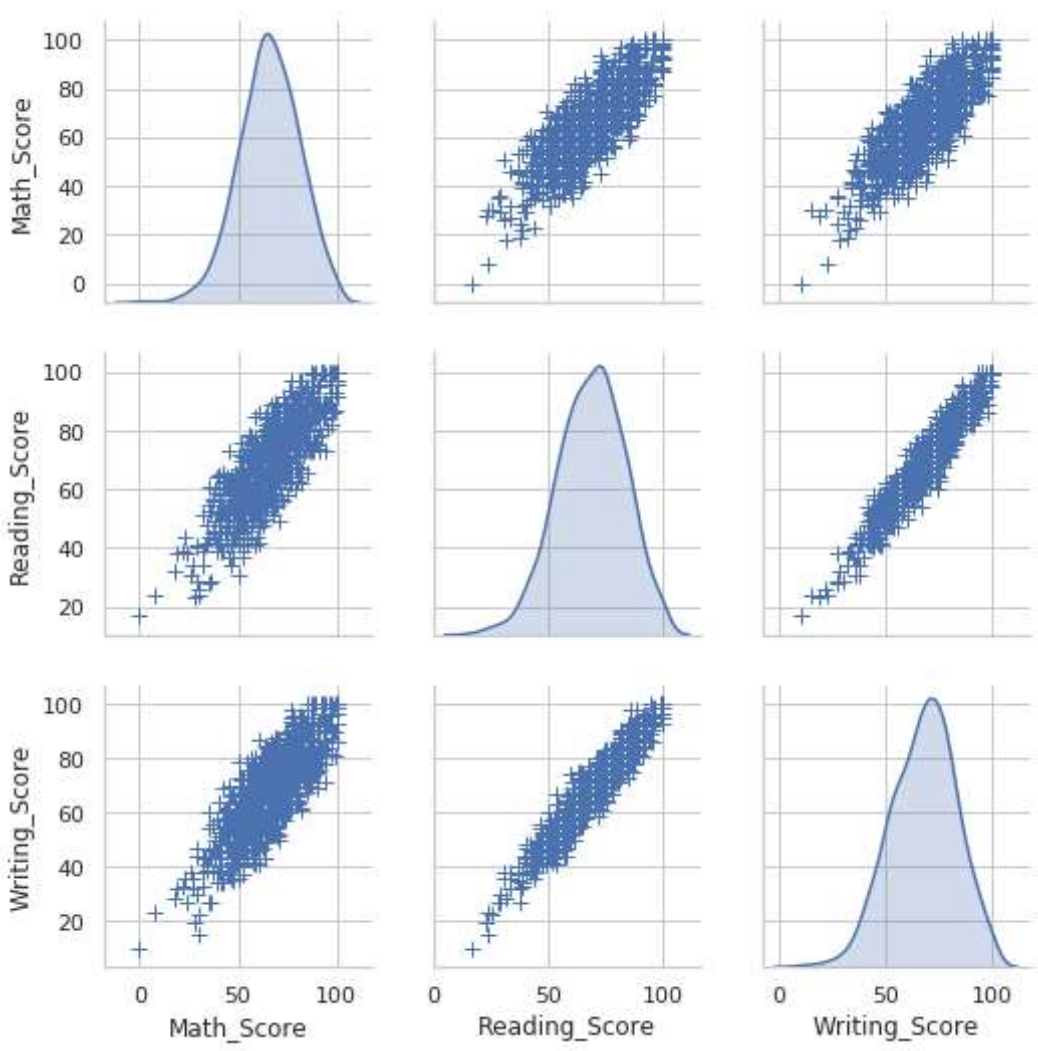
In [156…

```
sns.pairplot(data,kind='reg')
plt.show()
```



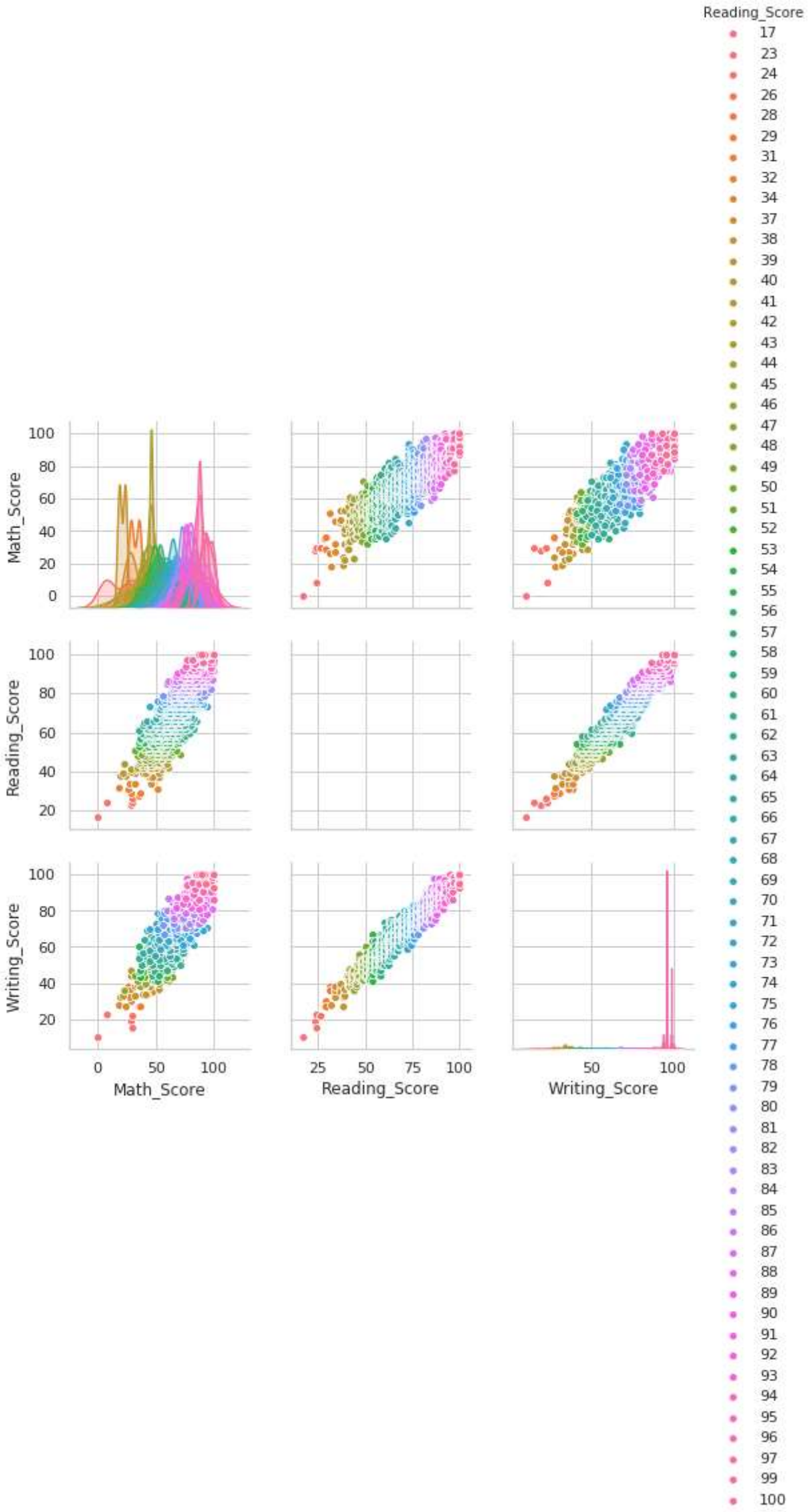
In [157…

```
sns.pairplot(data, diag_kind="kde", markers="+",
                  plot_kws=dict(s=50, edgecolor="b", linewidth=1),
                  diag_kws=dict(shade=True))
plt.show()
```



In [158…

```
sns.pairplot(data, hue="Reading_Score")
plt.show()
```



Count Plot

seaborn.countplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, orient=None, color=None, palette=None, saturation=0.75, dodge=True, ax=None, **kwargs)

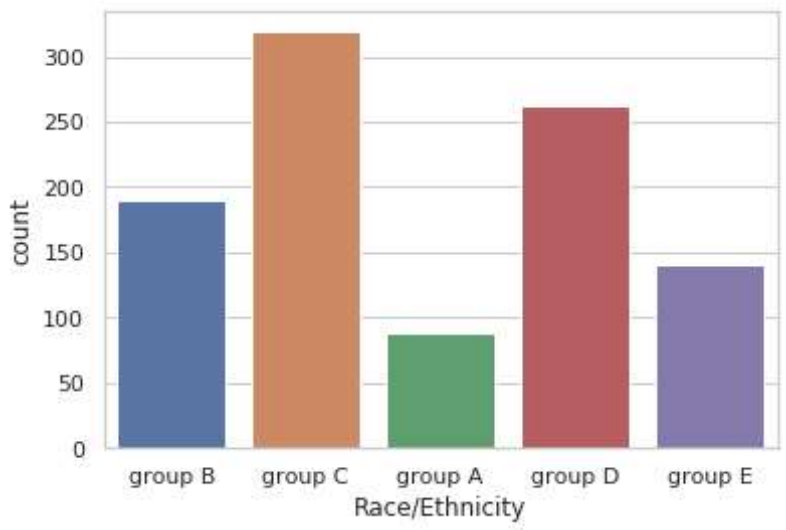
- x, y, hue : names of variables in data or vector data, optional
- data : DataFrame, array, or list of arrays, optional
- color : matplotlib color, optional
- palette : palette name, list, or dict, optional
- dodge : bool, optional
- ax : matplotlib Axes, optional
- ax : matplotlib Axes, optional

In [159…

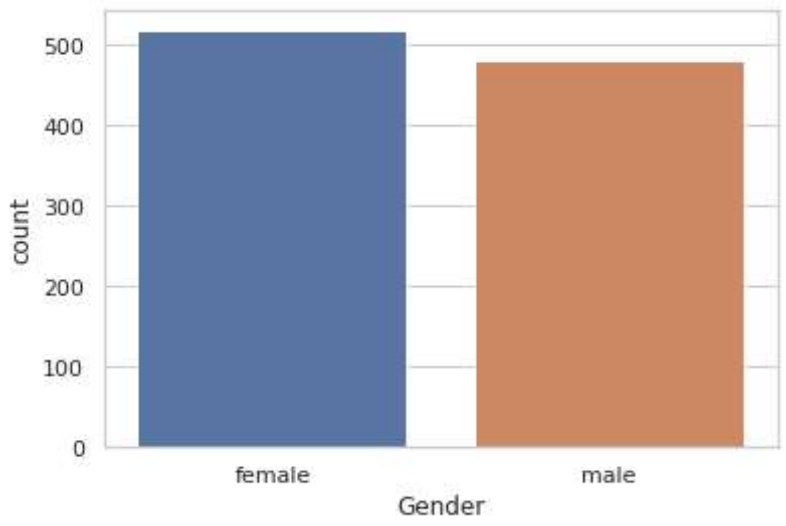
```
data.columns
```

Out[159]:
```
Index(['Gender', 'Race/Ethnicity', 'Parental_Level_of_Education', 'Lunch',
       'Test_Preparation_Course', 'Math_Score', 'Reading_Score',
       'Writing_Score'],
      dtype='object')
```

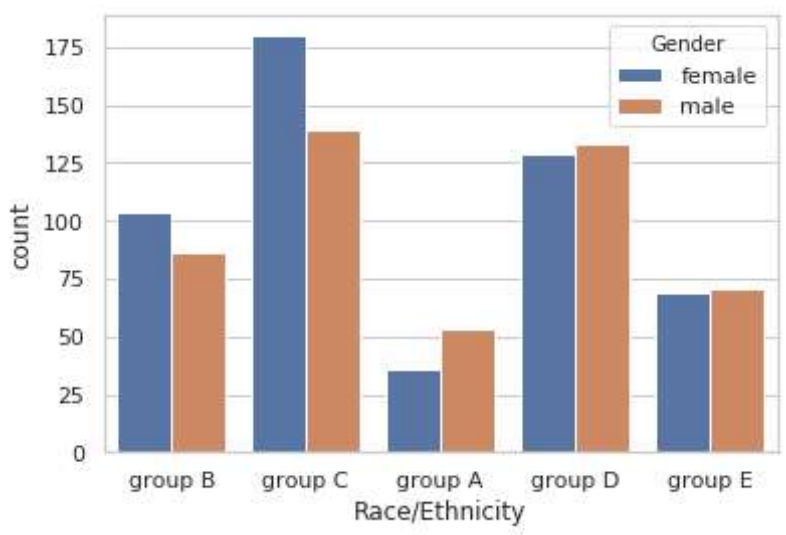
In [160…
```
sns.countplot(data['Race/Ethnicity'])
plt.show()
```



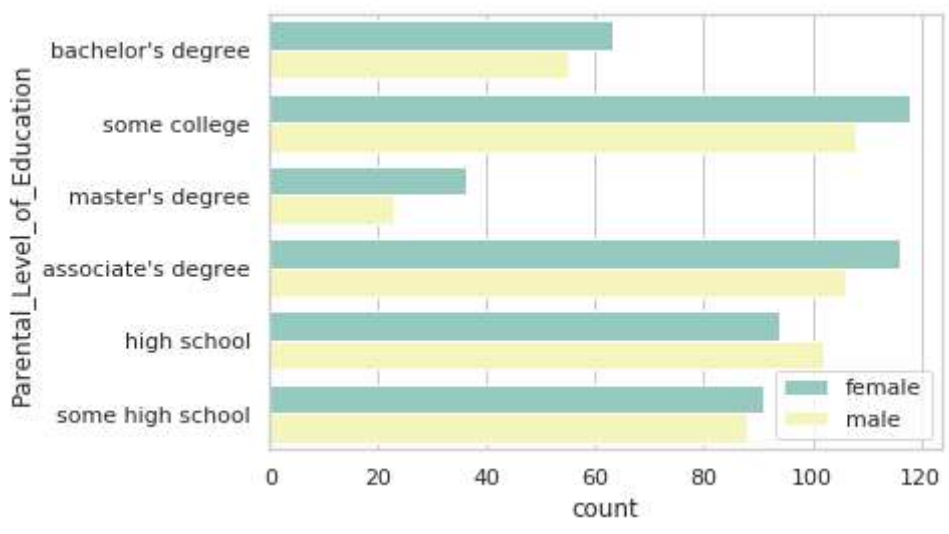
In [161…
```
sns.countplot(data['Gender'])
plt.show()
```



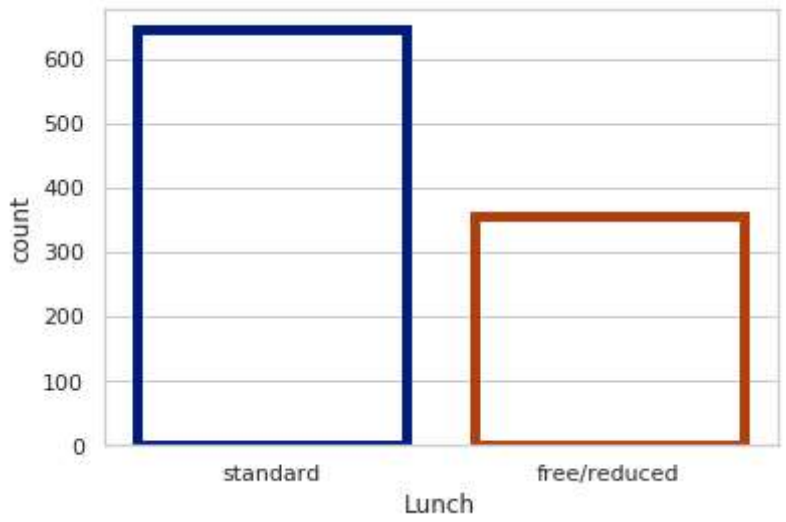
In [162…
```
sns.countplot(data['Race/Ethnicity'],hue=data['Gender'])
plt.show()
```



In [163…
```
sns.countplot(y=data['Parental_Level_of_Education'],palette="Set3",hue=data['Gender'])
plt.legend(loc=4)
plt.show()
```

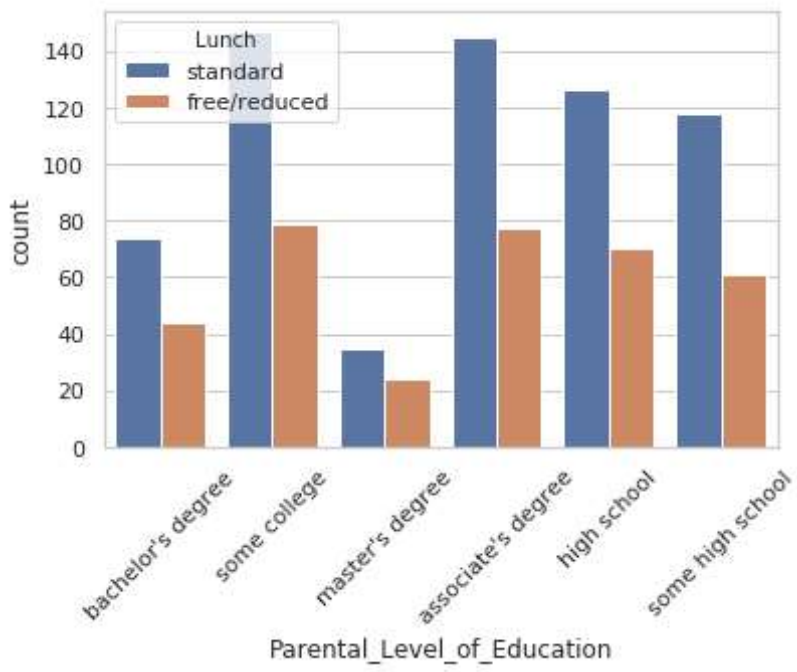


In [164…
```
sns.countplot(x=data['Lunch'],facecolor=(0,0,0,0),linewidth=5,edgecolor=sns.color_palette('dark',3))
plt.show()
```



In [165…

```
sns.countplot(x="Parental_Level_of_Education", hue="Lunch",
                    data=data)
plt.xticks(rotation=45)
plt.show()
```



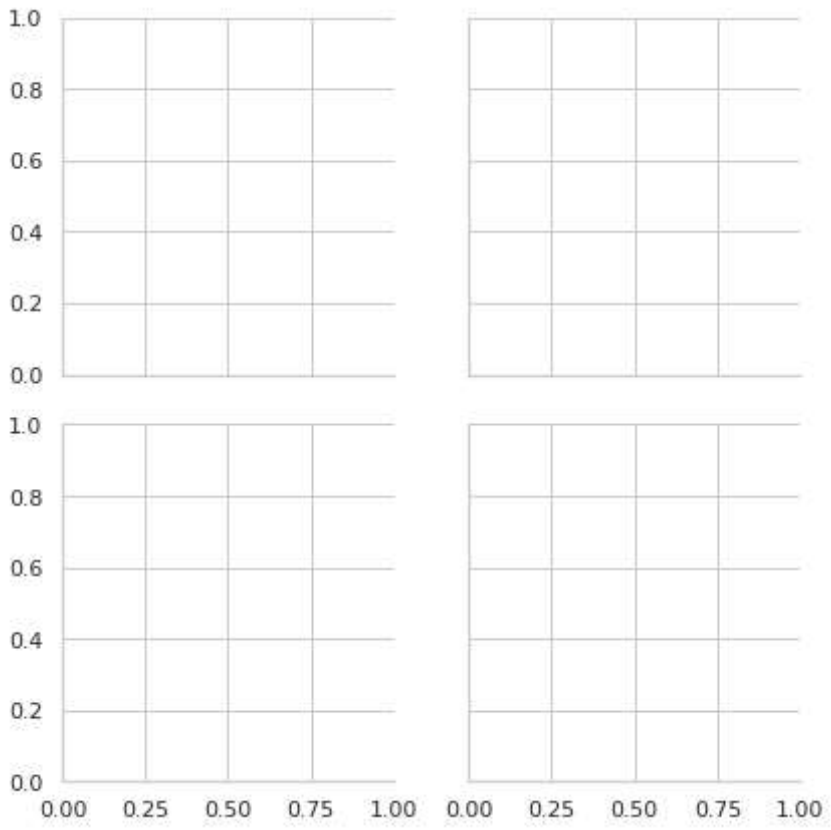
FacetGrid

class seaborn.FacetGrid(data, row=None, col=None, hue=None, col_wrap=None, sharex=True, sharey=True, height=3, aspect=1, palette=None, row_order=None, col_order=None, hue_order=None, hue_kws=None, dropna=True, legend_out=True, despine=True, margin_titles=False, xlim=None, ylim=None, subplot_kws=None, gridspec_kws=None, size=None)

- data : DataFrame
- row, col, hue : strings
- col_wrap : int, optional
- share{x,y} : bool, ‘col’, or ‘row’ optional
- palette : palette name, list, or dict, optional
- legend_out : bool, optional
- despine : boolean, optional
- subplot_kws : dict, optional

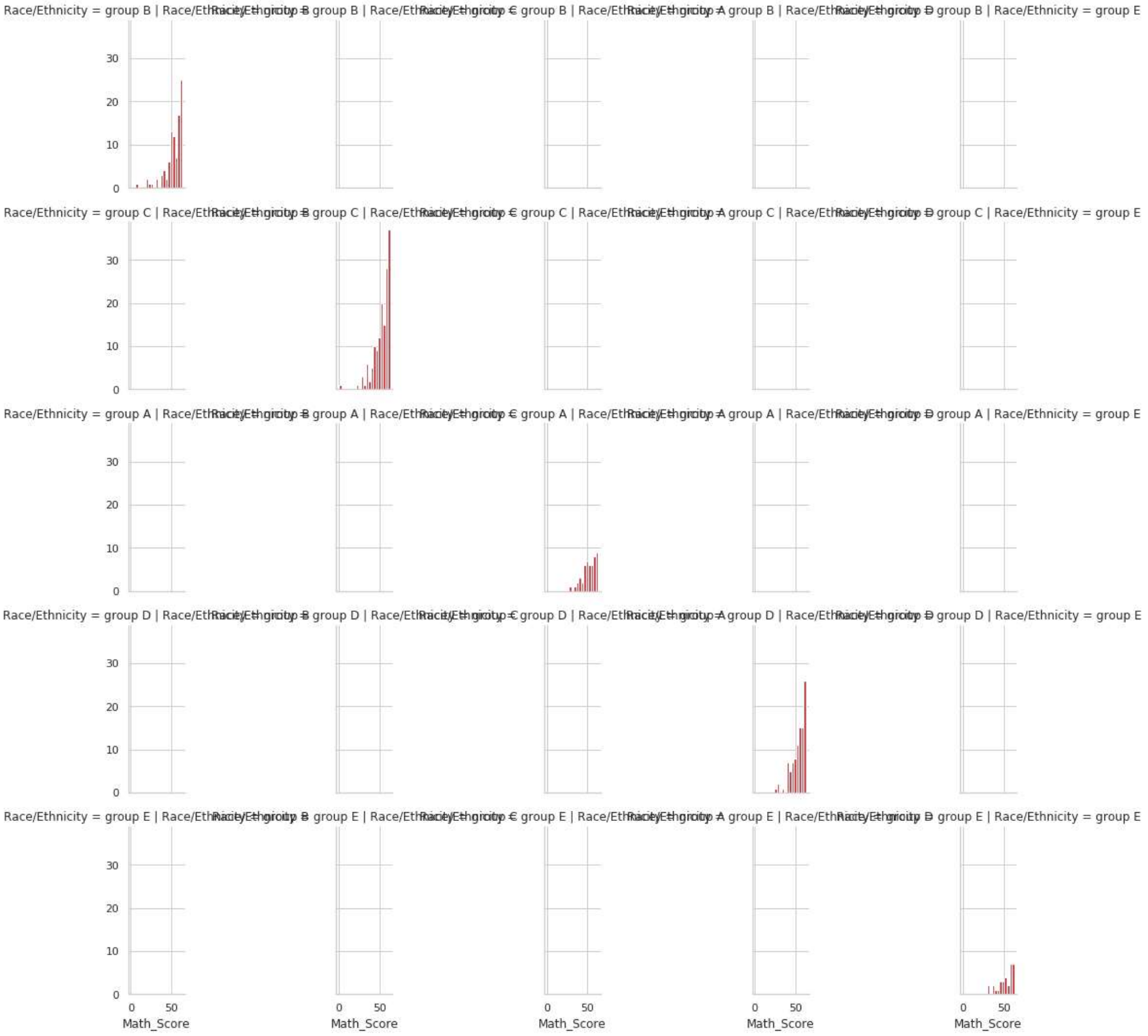
In [166…

```
sns.FacetGrid(data,col='Gender',row='Gender')
plt.tight_layout()
plt.show()
```



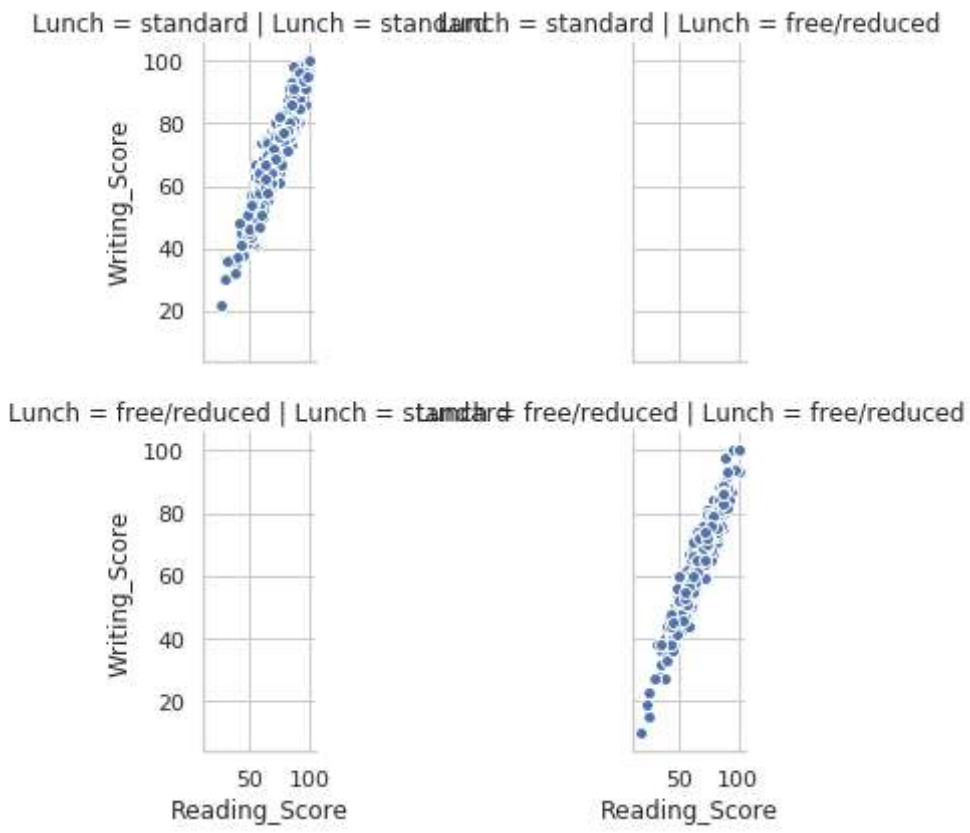
In [167…

```
g=sns.FacetGrid(data,col='Race/Ethnicity',row='Race/Ethnicity')
g=g.map(plt.hist,"Math_Score",bins=np.arange(0,65,3),color='r')
plt.show()
```



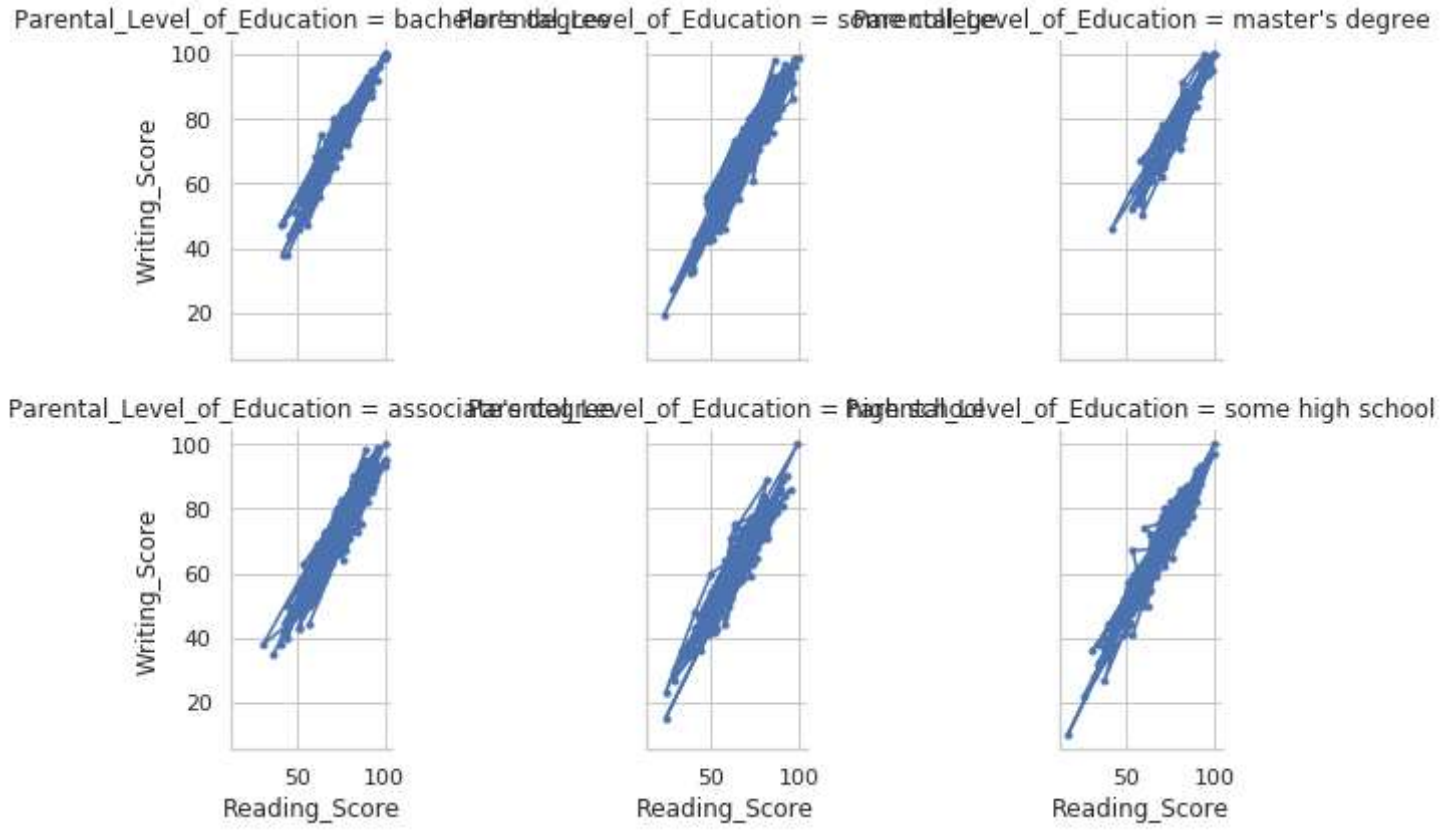
In [168…

```
g=sns.FacetGrid(data,col='Lunch',row='Lunch')
g=(g.map(plt.scatter,"Reading_Score",'Writing_Score',edgecolor='w').add_legend())
plt.tight_layout()
plt.show()
```



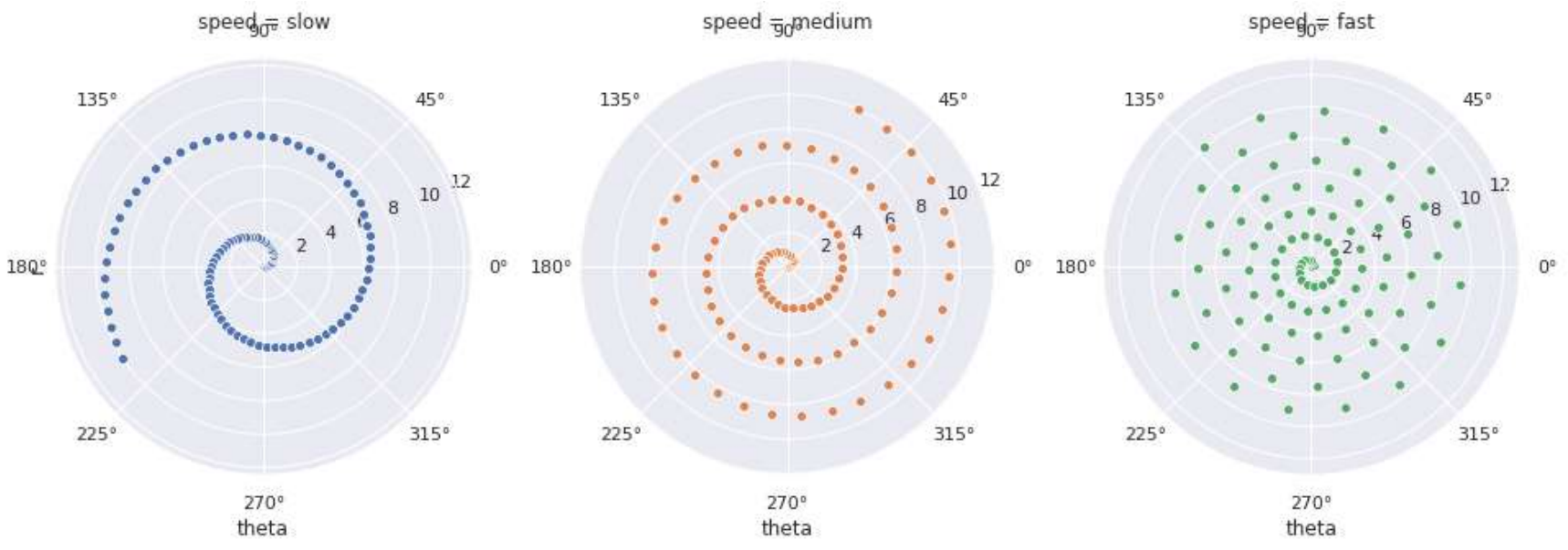
In [169…

```
g = sns.FacetGrid(data, col="Parental_Level_of_Education", col_wrap=3)
g = g.map(plt.plot, "Reading_Score", "Writing_Score", marker=".")
plt.show()
```



In [170…

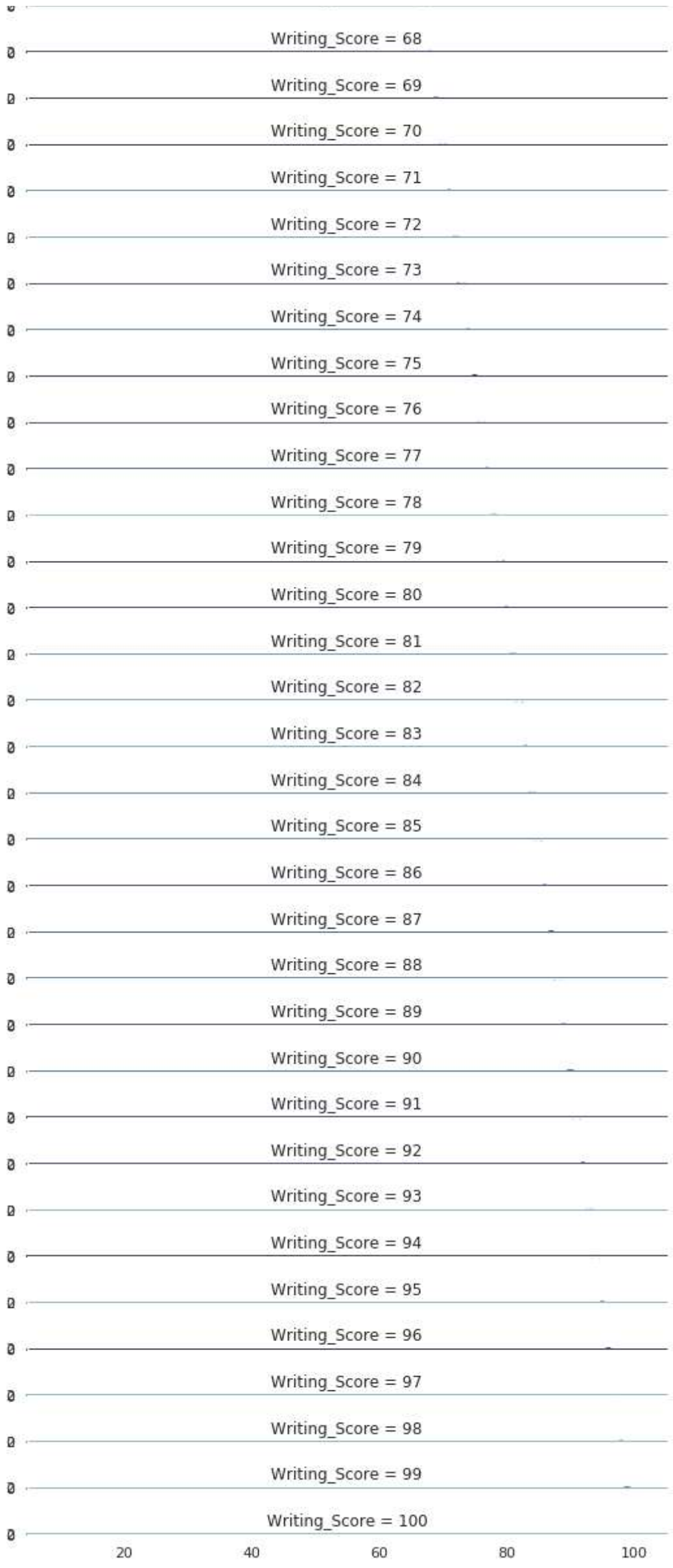
```
sns.set()

# Generate an example radial datast
r = np.linspace(0, 10, num=100)
df = pd.DataFrame({'r': r, 'slow': r, 'medium': 2 * r, 'fast': 4 * r})

# Convert the dataframe to long-form or "tidy" format
df = pd.melt(df, id_vars=['r'], var_name='speed', value_name='theta')

# Set up a grid of axes with a polar projection
g = sns.FacetGrid(df, col="speed", hue="speed",
                  subplot_kws=dict(projection='polar'), height=4.5,
                  sharex=False, sharey=False, despine=False)

# Draw a scatterplot onto each axes in the grid
g.map(sns.scatterplot, "theta", "r")
plt.show()
```



In [171…

```
pal = sns.cubehelix_palette(10, rot=-.25, light=.7)
g = sns.FacetGrid(data, row="Writing_Score", hue="Reading_Score", aspect=15, height=.5, palette=pal)

# Draw the densities in a few steps
g.map(sns.kdeplot, "Writing_Score", clip_on=False, shade=True, alpha=1, lw=1.5, bw=.2)
g.map(sns.kdeplot, "Reading_Score", clip_on=False, color="w", lw=2, bw=.2)
g.map(plt.axhline, y=0, lw=2, clip_on=False)
plt.show()
```

Writing_Score = 10
0
Writing_Score = 15
0
Writing_Score = 19
0
Writing_Score = 22
0
Writing_Score = 23
0
Writing_Score = 27
0
Writing_Score = 28
0
Writing_Score = 30
0
Writing_Score = 32
0
Writing_Score = 33
0
Writing_Score = 34
0
Writing_Score = 35
0
Writing_Score = 36
0
Writing_Score = 37
0
Writing_Score = 38
0
Writing_Score = 39
0
Writing_Score = 40
0
Writing_Score = 41
0
Writing_Score = 42
0
Writing_Score = 43
0
Writing_Score = 44
0
Writing_Score = 45
0
Writing_Score = 46
0
Writing_Score = 47
0
Writing_Score = 48
0
Writing_Score = 49
0
Writing_Score = 50
0
Writing_Score = 51
0
Writing_Score = 52
0
Writing_Score = 53
0
Writing_Score = 54
0
Writing_Score = 55
0
Writing_Score = 56
0
Writing_Score = 57
0
Writing_Score = 58
0
Writing_Score = 59
0
Writing_Score = 60
0
Writing_Score = 61
0
Writing_Score = 62
0
Writing_Score = 63
0
Writing_Score = 64
0
Writing_Score = 65
0
Writing_Score = 66
0
Writing_Score = 67
0



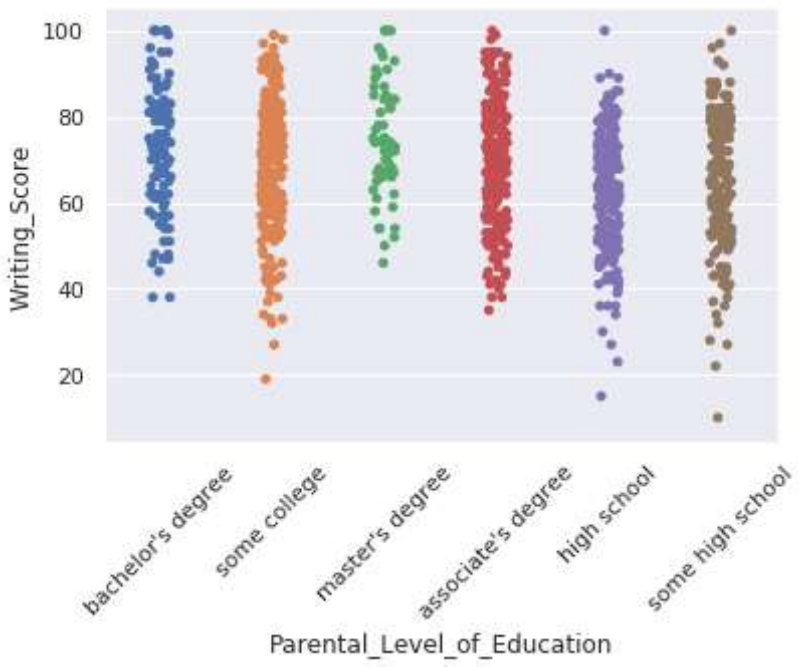
Strip Plot

seaborn.stripplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None, jitter=True, dodge=False, orient=None, color=None, palette=None, size=5, edgecolor='gray', linewidth=0, ax=None, **kwargs)

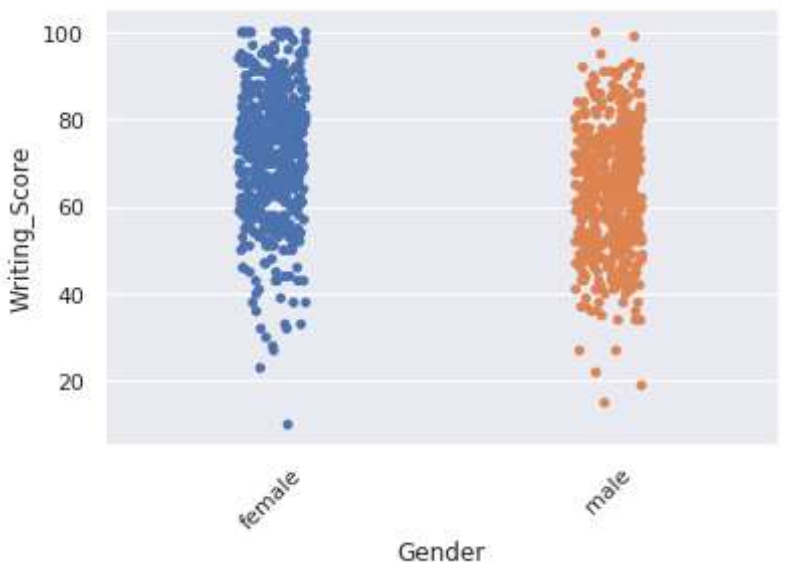
- x, y, hue : names of variables in data or vector data, optional
- data : DataFrame, array, or list of arrays, optional
- order, hue_order : lists of strings, optional
- jitter : float, True/1 is special-cased, optional
- dodge : bool, optional
- color : matplotlib color, optional
- edgecolor : matplotlib color, “gray” is special-cased, optional
- linewidth : float, optional

In [172…
```
sns.stripplot(x=data['Reading_Score'])
plt.show()
```

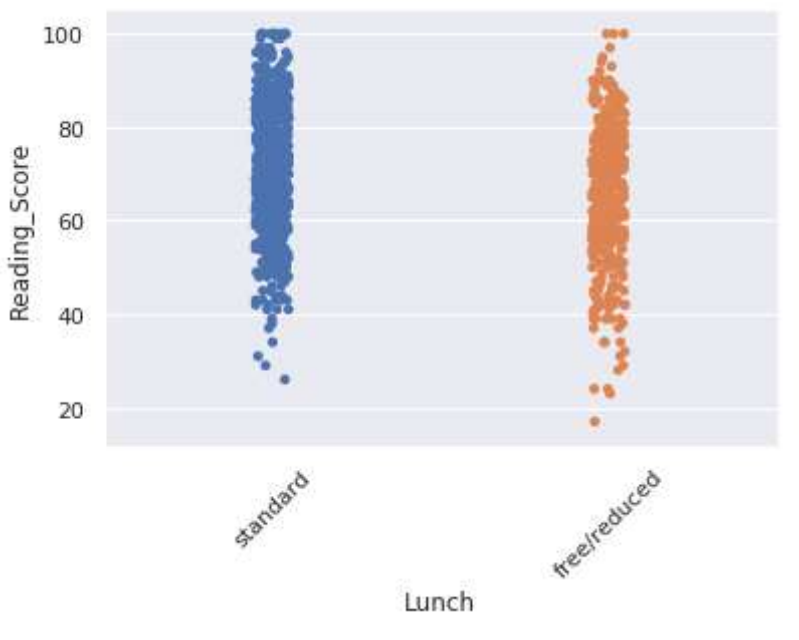
In [173…
```
sns.stripplot(x="Parental_Level_of_Education",y='Writing_Score',data=data)
plt.xticks(rotation=45)
plt.show()
```



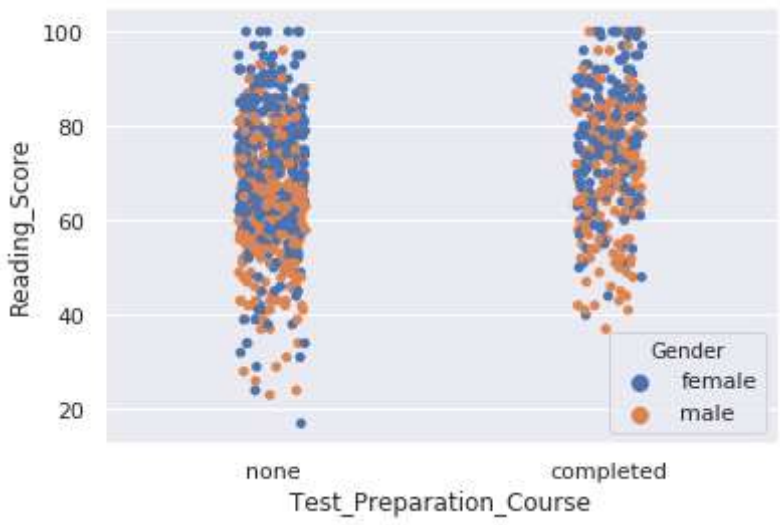
In [174…
```
sns.stripplot(x="Gender",y='Writing_Score',jitter=True,data=data)
plt.xticks(rotation=45)
plt.show()
```



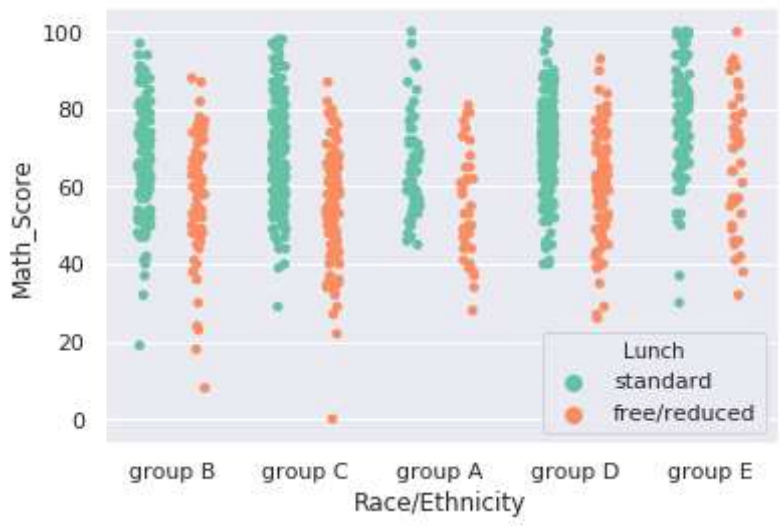
In [175…
```
sns.stripplot(x="Lunch",y='Reading_Score',jitter=0.05,data=data)
plt.xticks(rotation=45)
plt.show()
```



In [176…
```
sns.stripplot(x='Test_Preparation_Course',y='Reading_Score',hue='Gender',jitter=True,data=data)
plt.show()
```



In [177…
```
sns.stripplot(x='Race/Ethnicity',y='Math_Score',hue='Lunch',jitter=True,dodge=True,palette="Set2",data=data)
plt.show()
```



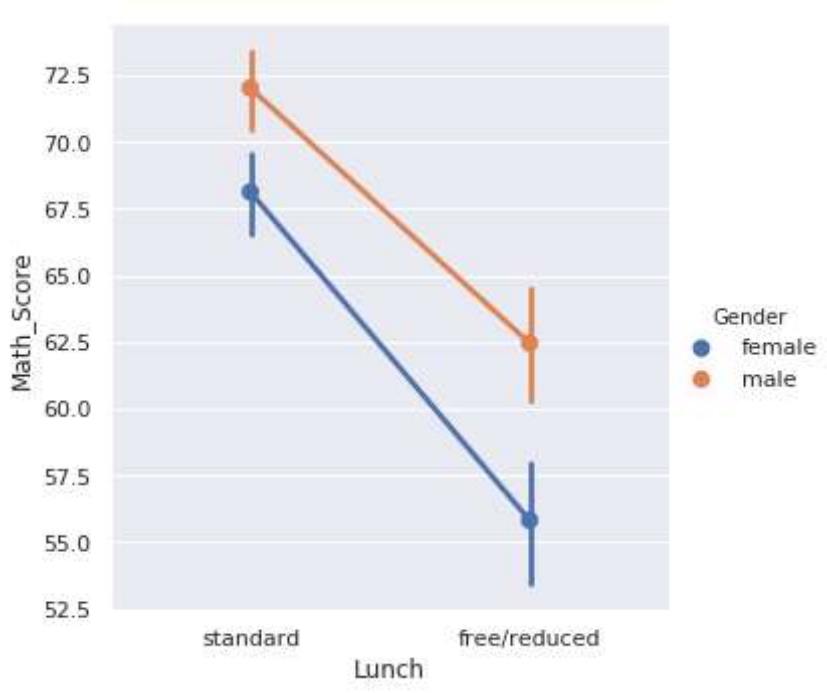
In [178…
```
sns.stripplot(x='Lunch',y='Math_Score',hue='Lunch',jitter=True,dodge=True,size=20,marker='D',edgecolor='gray',alpha=.25
plt.legend(loc=10)
plt.show()
```

Factor Plot

seaborn.catplot(x=None, y=None, hue=None, data=None, row=None, col=None, col_wrap=None, estimator=, ci=95, n_boot=1000, units=None, order=None, hue_order=None, row_order=None, col_order=None, kind='strip', height=5, aspect=1, orient=None, color=None, palette=None, legend=True, legend_out=True, sharex=True, sharey=True, margin_titles=False, facet_kws=None, **kwargs)

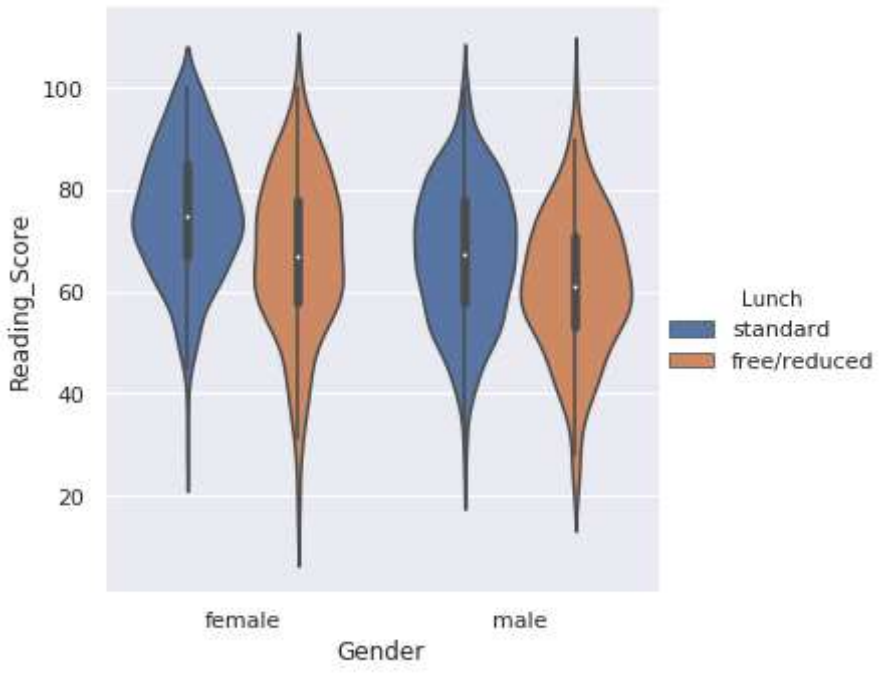
- x, y, hue : names of variables in data
- data : DataFrame
- row, col : names of variables in data, optional
- col_wrap : int, optional
- kind : string, optional
- orient : “v” | “h”, optional
- height : scalar, optional
- color : matplotlib color, optional
- palette : palette name, list, or dict, optional
- legend : bool, optional

In [179…
```
sns.factorplot(x="Lunch", y="Math_Score", hue="Gender", data=data)
plt.show()
```



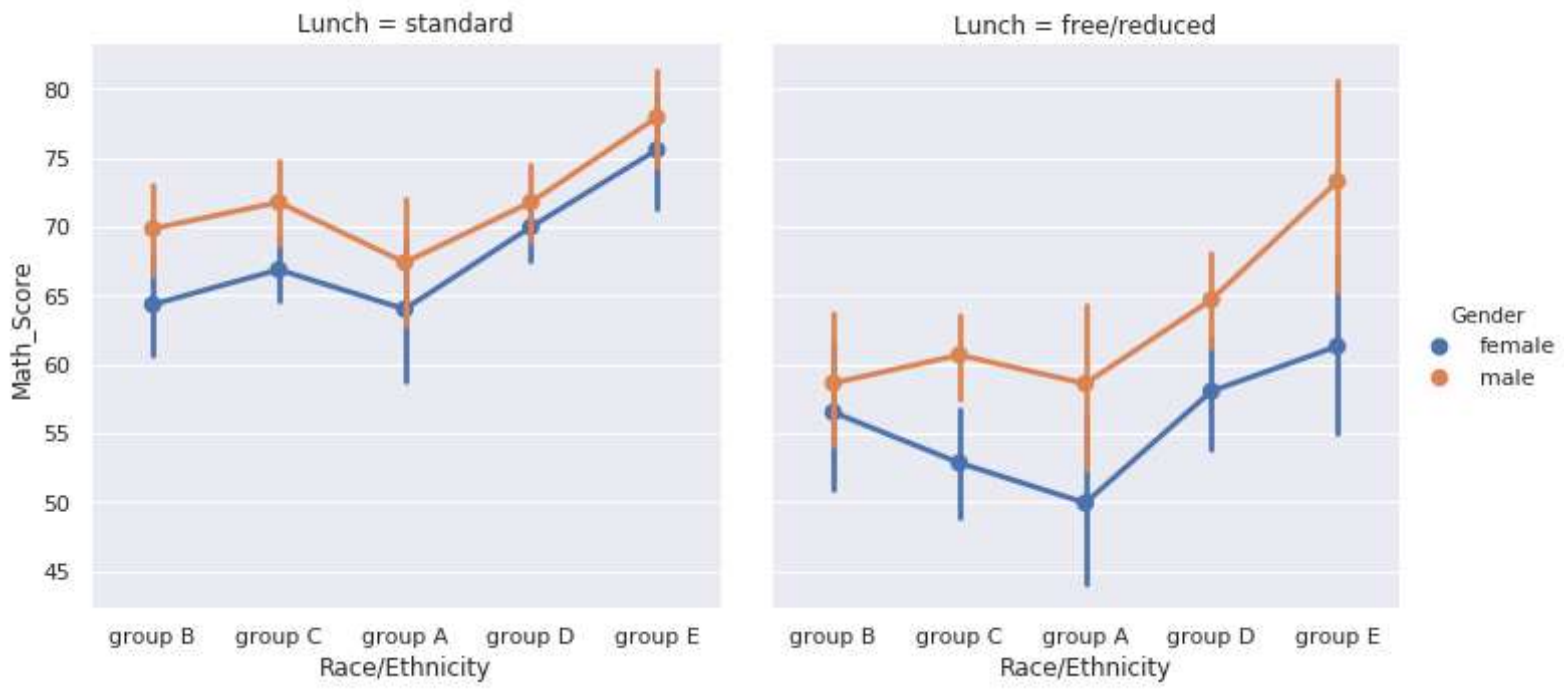
In [180…

```
sns.factorplot(x="Gender", y="Reading_Score", hue="Lunch", kind='violin',data=data)
plt.show()
```



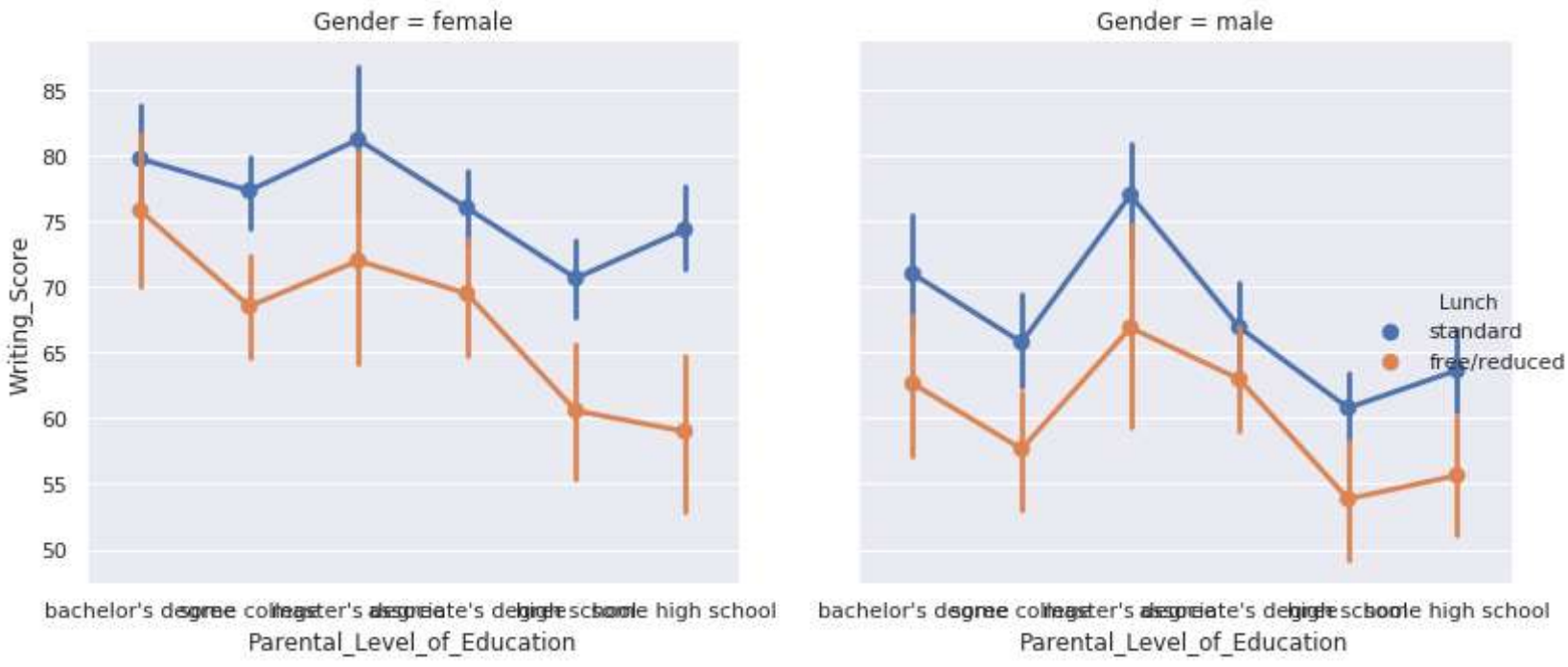
In [181…

```
sns.factorplot(x="Race/Ethnicity", y="Math_Score", hue="Gender",col='Lunch',data=data)
plt.show()
```



In [182…

```
g=sns.factorplot(x="Parental_Level_of_Education", y="Writing_Score", hue="Lunch",
                 col="Gender", data=data)
plt.tight_layout()
plt.show()
```

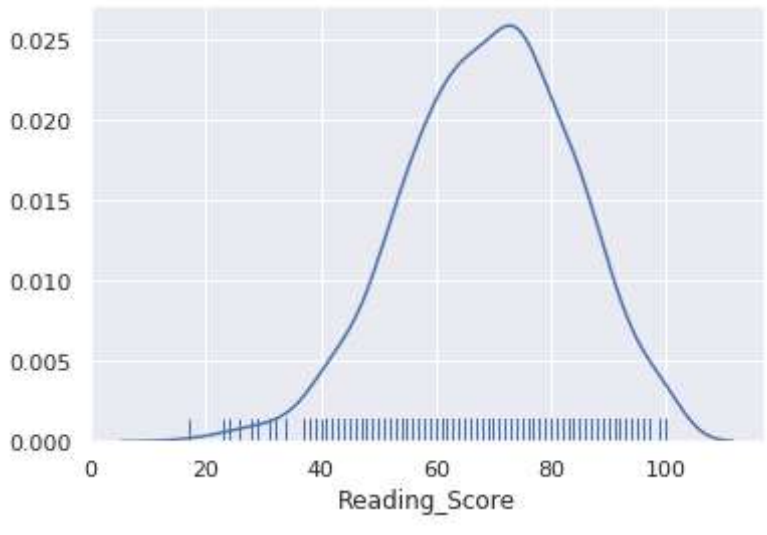


DisPlot

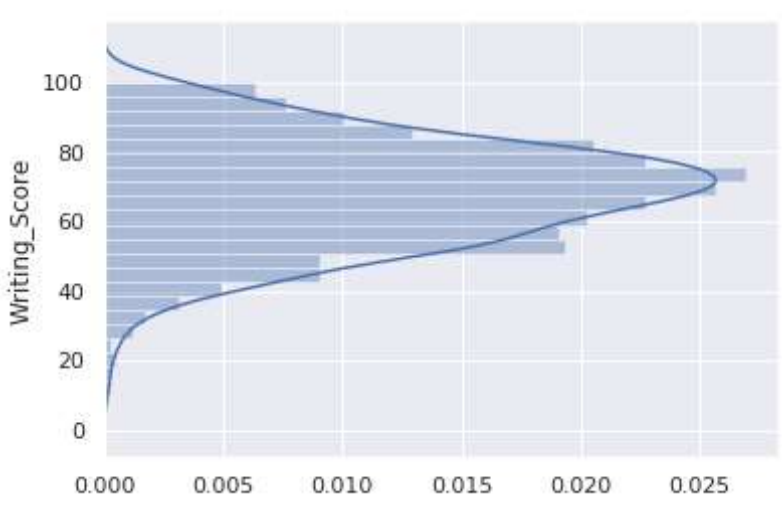
seaborn.distplot(a, bins=None, hist=True, kde=True, rug=False, fit=None, hist_kws=None, kde_kws=None, rug_kws=None, fit_kws=None, color=None, vertical=False, norm_hist=False, axlabel=None, label=None, ax=None)

- a : Series, 1d-array, or list
- bins : argument for matplotlib hist(), or None, optional
- hist : bool, optional
- kde : bool, optional
- color : matplotlib color, optional
- label : string, optional

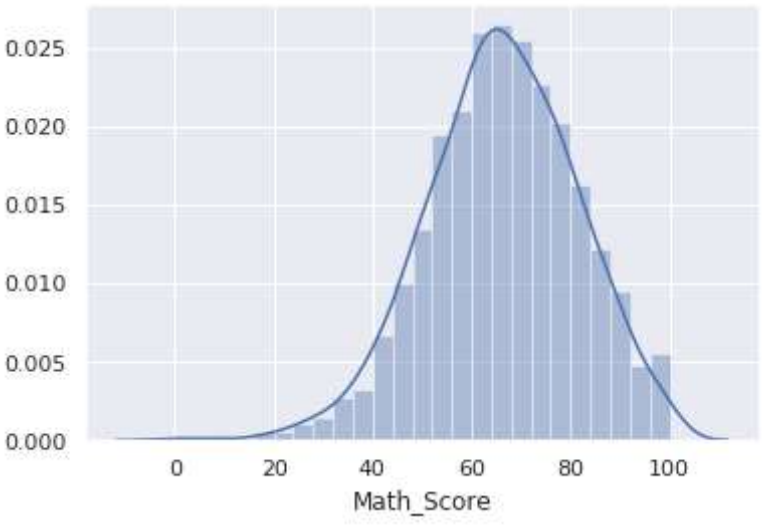
In [183…
```
ax = sns.distplot(data['Reading_Score'], rug=True, hist=False)
plt.show()
```



In [184…
```
ax = sns.distplot(data['Writing_Score'], vertical=True)
plt.show()
```

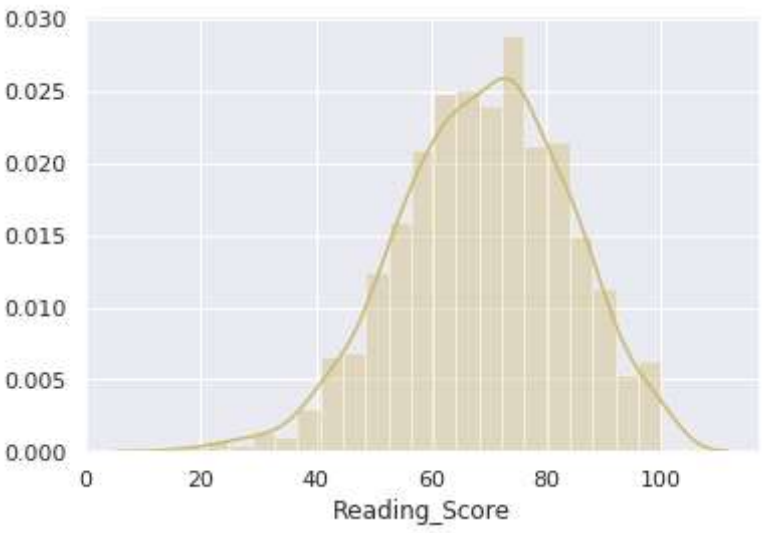


In [185…
```
ax = sns.distplot(data['Math_Score'])
plt.show()
```



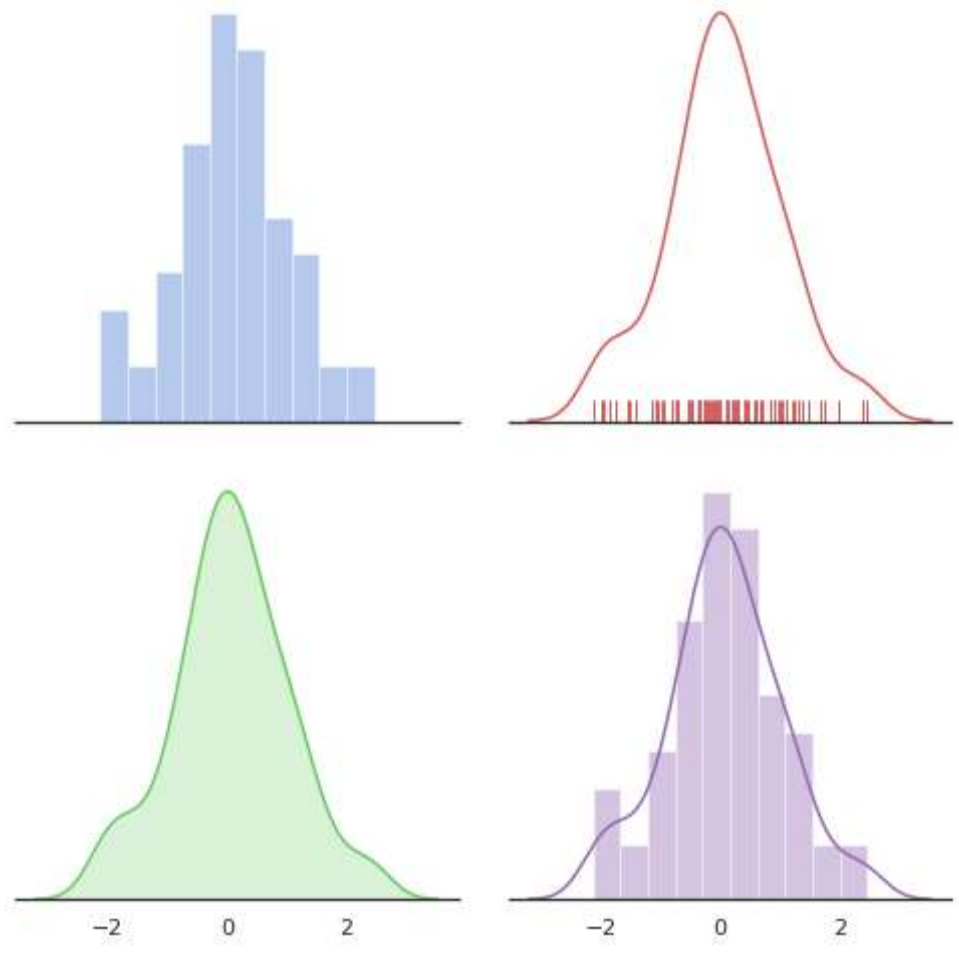
In [186…

```
ax = sns.distplot(data['Reading_Score'], color="y")
plt.show()
```



In [187…

```
sns.set(style="white", palette="muted", color_codes=True)
rs = np.random.RandomState(10)

f, axes = plt.subplots(2, 2, figsize=(7, 7), sharex=True)
sns.despine(left=True)

# Generate a random univariate dataset
d = rs.normal(size=100)

# Plot a simple histogram with binsize determined automatically
sns.distplot(d, kde=False, color="b", ax=axes[0, 0])

# Plot a kernel density estimate and rug plot
sns.distplot(d, hist=False, rug=True, color="r", ax=axes[0, 1])

# Plot a filled kernel density estimate
sns.distplot(d, hist=False, color="g", kde_kws={"shade": True}, ax=axes[1, 0])

# Plot a historgram and kernel density estimate
sns.distplot(d, color="m", ax=axes[1, 1])

plt.setp(axes, yticks=[])
plt.tight_layout()
plt.show()
```



Line Plot

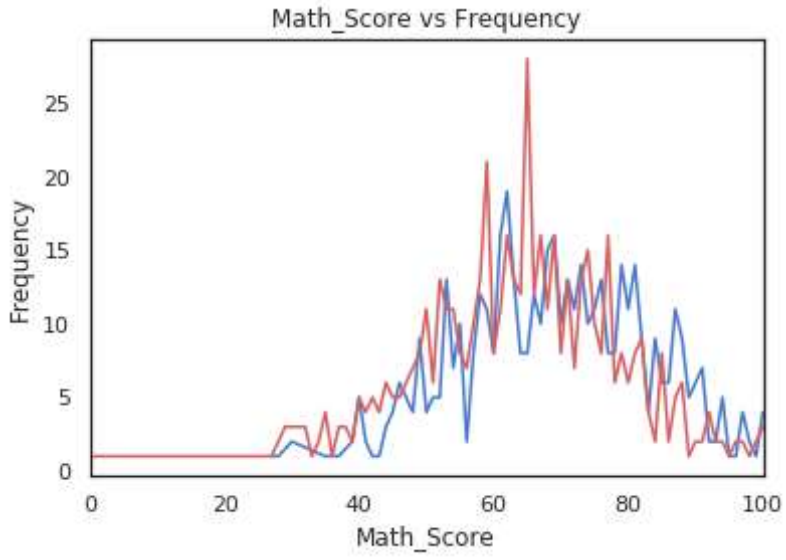
seaborn.lineplot(x=None, y=None, hue=None, size=None, style=None, data=None, palette=None, hue_order=None, hue_norm=None, sizes=None, size_order=None, size_norm=None, dashes=True, markers=None, style_order=None, units=None, estimator='mean', ci=95, n_boot=1000, sort=True, err_style='band', err_kws=None, legend='brief', ax=None, **kwargs)

- x, y : names of variables in data or vector data, optional
- hue : name of variables in data or vector data, optional
- size : name of variables in data or vector data, optional
- style : name of variables in data or vector data, optional
- data : DataFrame
- palette : palette name, list, or dict, optional

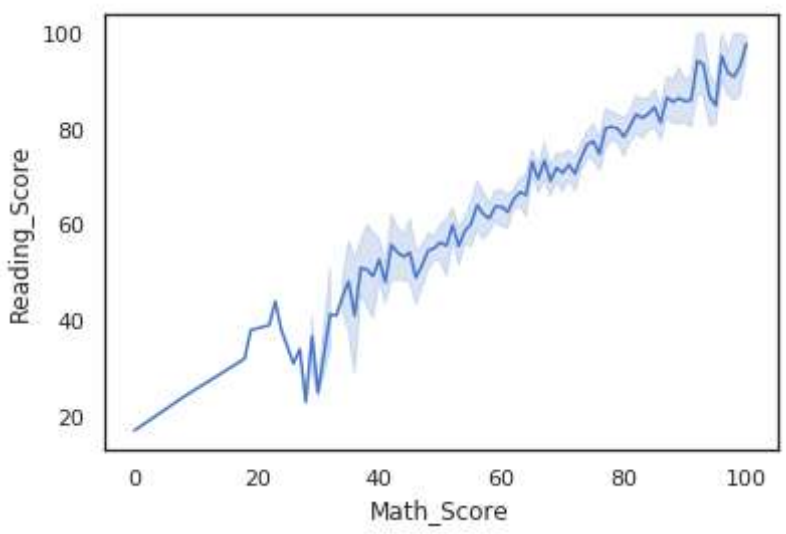
In [188…
```
data.columns
```

Out[188]:
```
Index(['Gender', 'Race/Ethnicity', 'Parental_Level_of_Education', 'Lunch',
       'Test_Preparation_Course', 'Math_Score', 'Reading_Score',
       'Writing_Score'],
      dtype='object')
```

In [189…
```
data[data['Gender']=='male']['Math_Score'].value_counts().sort_index().plot.line(color='b')
data[data['Gender']=='female']['Math_Score'].value_counts().sort_index().plot.line(color='r')
plt.xlabel('Math_Score')
plt.ylabel('Frequency')
plt.title('Math_Score vs Frequency')
plt.show()
```

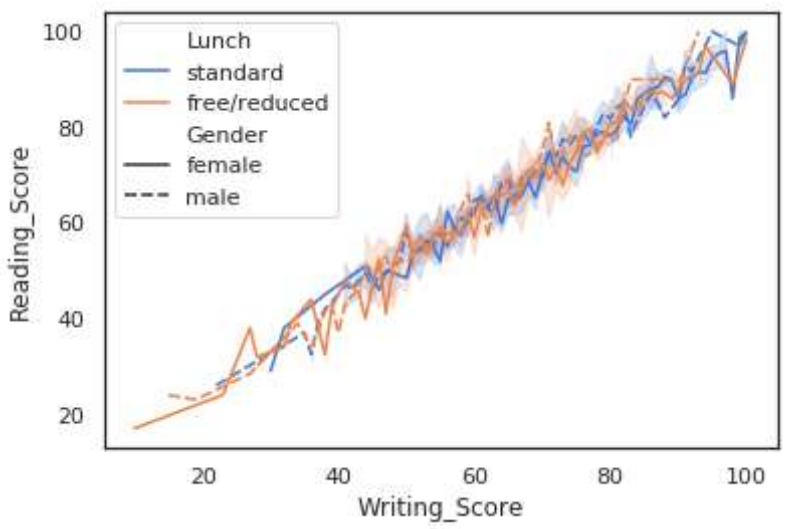


In [190…
```
sns.lineplot(x='Math_Score',y='Reading_Score',data=data)
plt.show()
```

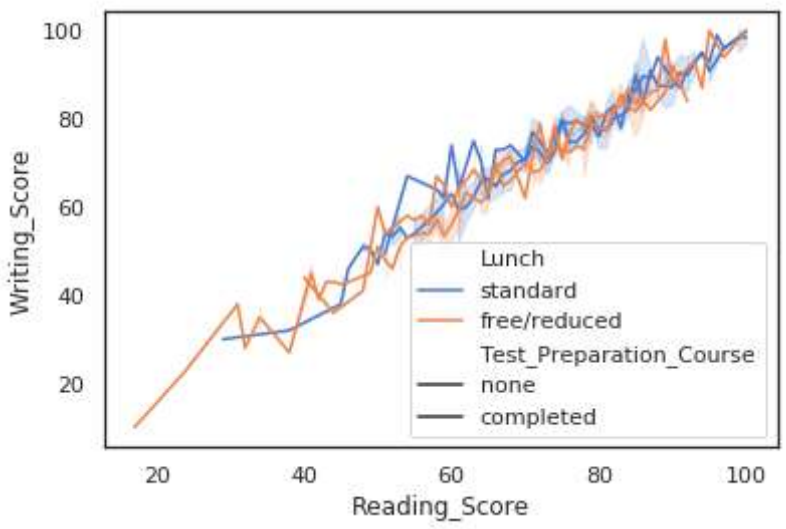


In [191…
```
sns.lineplot(x='Reading_Score',y='Writing_Score',hue='Lunch',data=data)
plt.show()
```

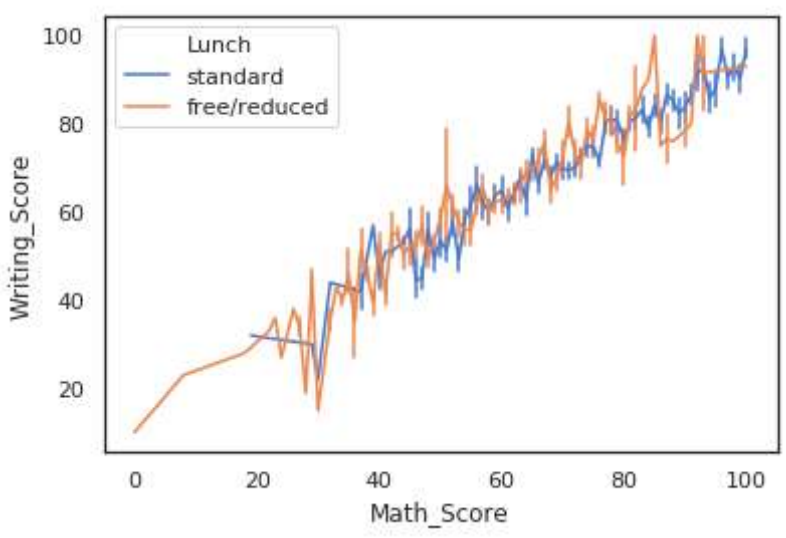
In [192…
```
sns.lineplot(x='Writing_Score',y='Reading_Score',data=data,hue='Lunch',
             style='Gender')
plt.show()
```



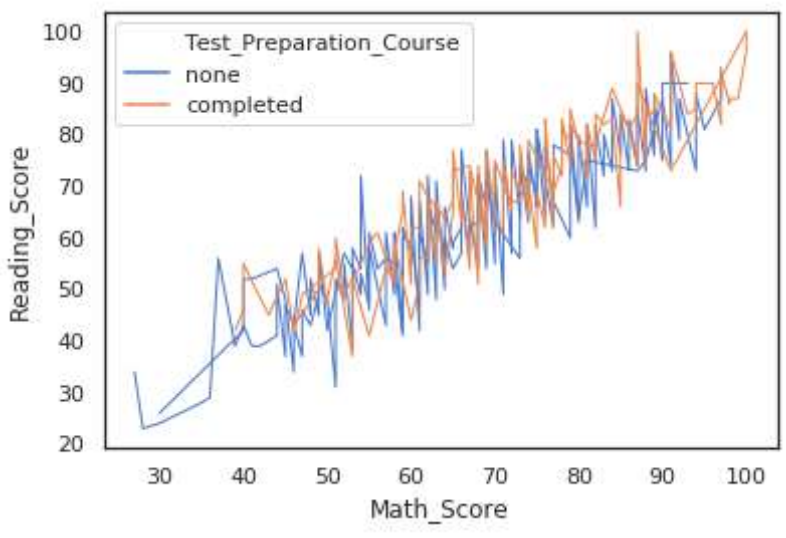
In [193…
```
female_filter=data[data['Gender']=='female']
sns.lineplot(x='Reading_Score',y='Writing_Score',data=female_filter,
             hue='Lunch',style='Test_Preparation_Course',dashes=False)
plt.show()
```



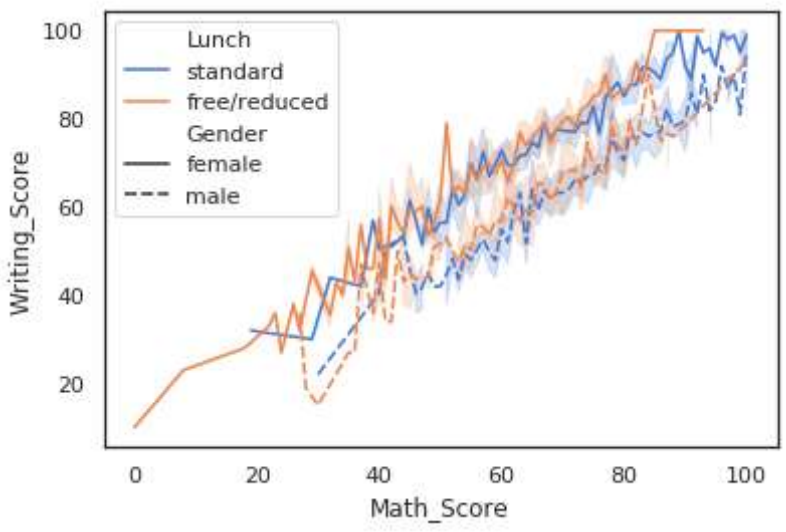
In [194…
```
sns.lineplot(x="Math_Score", y="Writing_Score", hue="Lunch",err_style="bars", ci=68, data=data)
plt.show()
```



In [195…
```
ax = sns.lineplot(x="Math_Score", y="Reading_Score", hue="Test_Preparation_Course",
                  units="Lunch", estimator=None, lw=1,
                  data=data.query("Gender == 'male'"))
```



In [196…
```
ax = sns.lineplot(x="Math_Score", y="Writing_Score",
                  hue="Lunch", style="Gender",
                  data=data)
plt.show()
```



In [197…
```
data.groupby('Gender')[['Writing_Score','Reading_Score']].mean()
```

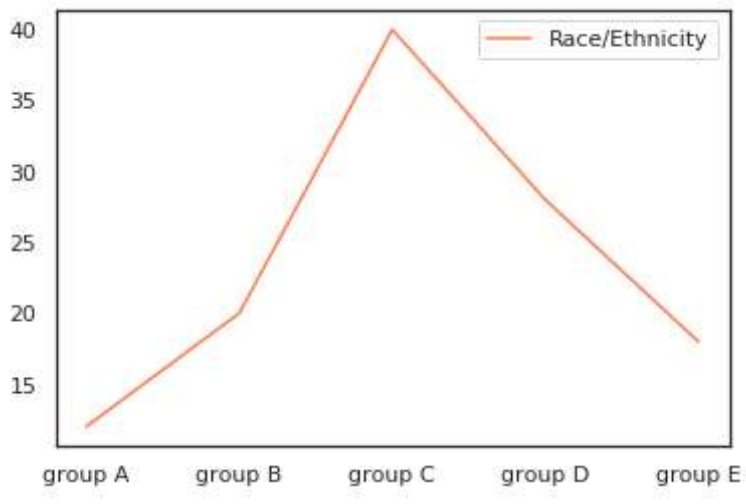
Out[197]:

Gender	Writing_Score	Reading_Score
female	72.467181	72.608108
male	63.311203	65.473029

In [198…
```
x=data[data.Parental_Level_of_Education=='bachelor\'s degree'].groupby('Race/Ethnicity')['Math_Score'].count()
x
```

Out[198]:
```
Race/Ethnicity
group A    12
group B    20
group C    40
group D    28
group E    18
Name: Math_Score, dtype: int64
```

In [199…
```
sns.lineplot(data=x,color='coral',label='Race/Ethnicity')
plt.show()
```

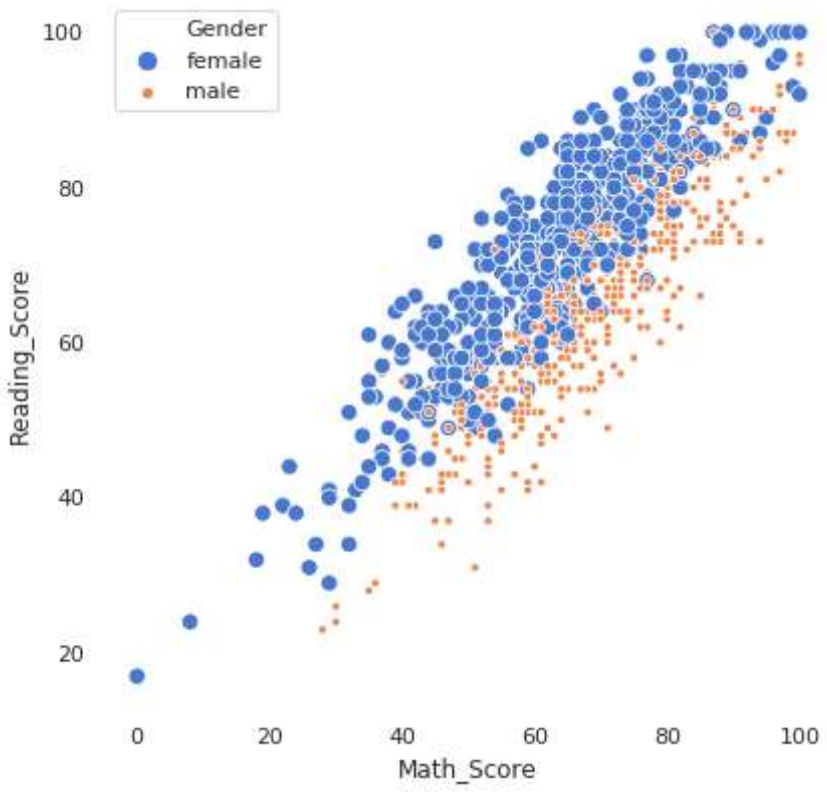


Despine

seaborn.despine(fig=None, ax=None, top=True, right=True, left=False, bottom=False, offset=None, trim=False)

- fig : matplotlib figure, optional
- ax : matplotlib axes, optional
- top, right, left, bottom : boolean, optional
- offset : int or dict, optional
- trim : bool, optional

In [200…
```
f, ax = plt.subplots(figsize=(6.5, 6.5))
sns.despine(f, left=True, bottom=True)
sns.scatterplot(x="Math_Score", y="Reading_Score",
                hue="Gender", size="Gender",data=data)
plt.show()
```



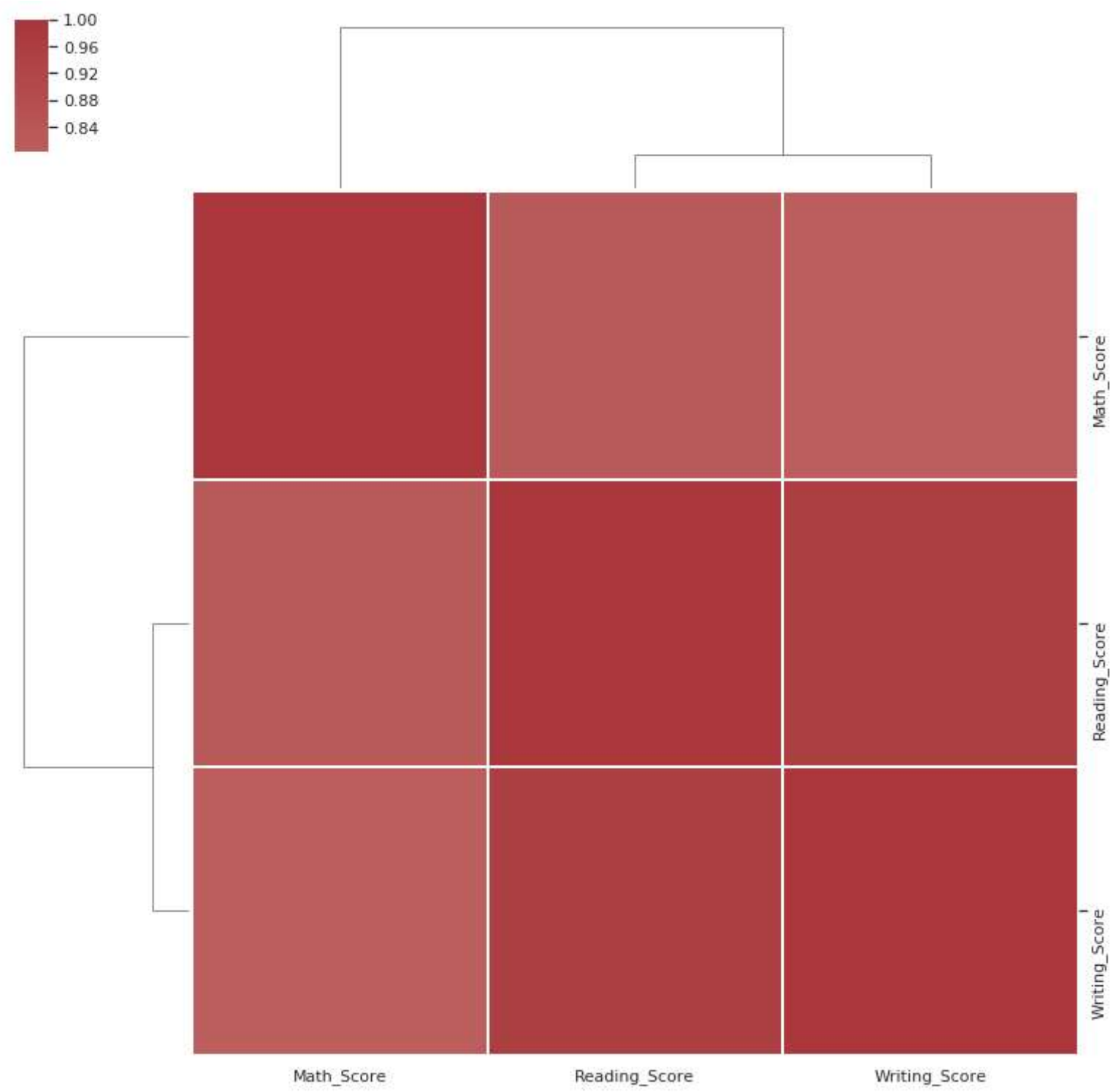
In [201…
```
data.head()
```

Out[201]:

	Gender	Race/Ethnicity	Parental_Level_of_Education	Lunch	Test_Preparation_Course	Math_Score	Reading_Score	Writing_Score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

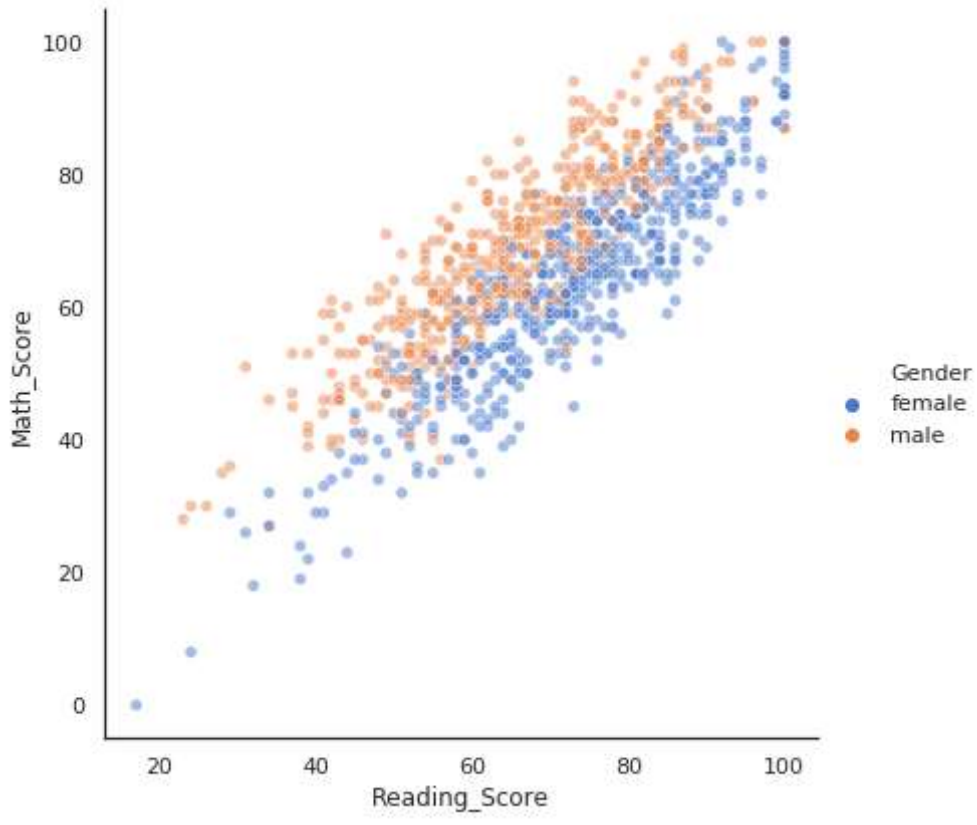
In [202…
```
# Draw the full plot
sns.clustermap(data.corr(), center=0, cmap="vlag",
               linewidths=.75, figsize=(13, 13))
```

Out[202]:
```
<seaborn.matrix.ClusterGrid at 0x7f84c2d0a128>
```



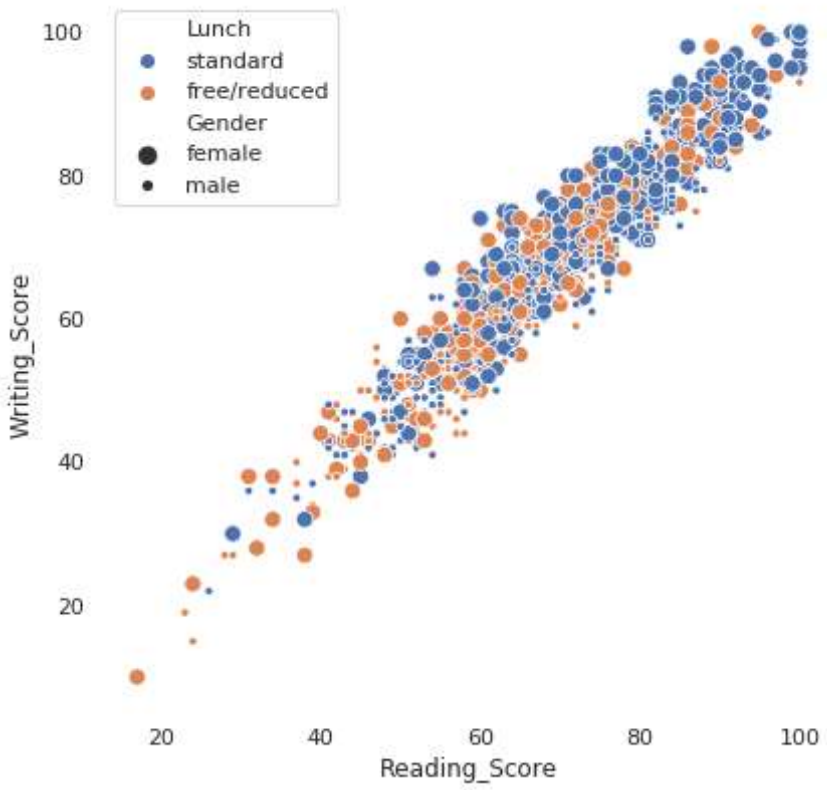
In [203…
```
sns.set(style="white")
# Plot miles per gallon against horsepower with other semantics
```

```
sns.relplot(x="Reading_Score",y="Math_Score",hue="Gender",
            sizes=(40, 400), alpha=.5, palette="muted",
            height=6, data=data)
plt.show()
```



In [204…

```
f, ax = plt.subplots(figsize=(6.5, 6.5))
sns.despine(f, left=True, bottom=True)
sns.scatterplot(x="Reading_Score", y="Writing_Score",
                hue="Lunch", size="Gender",data=data)
plt.show()
```



In [205…

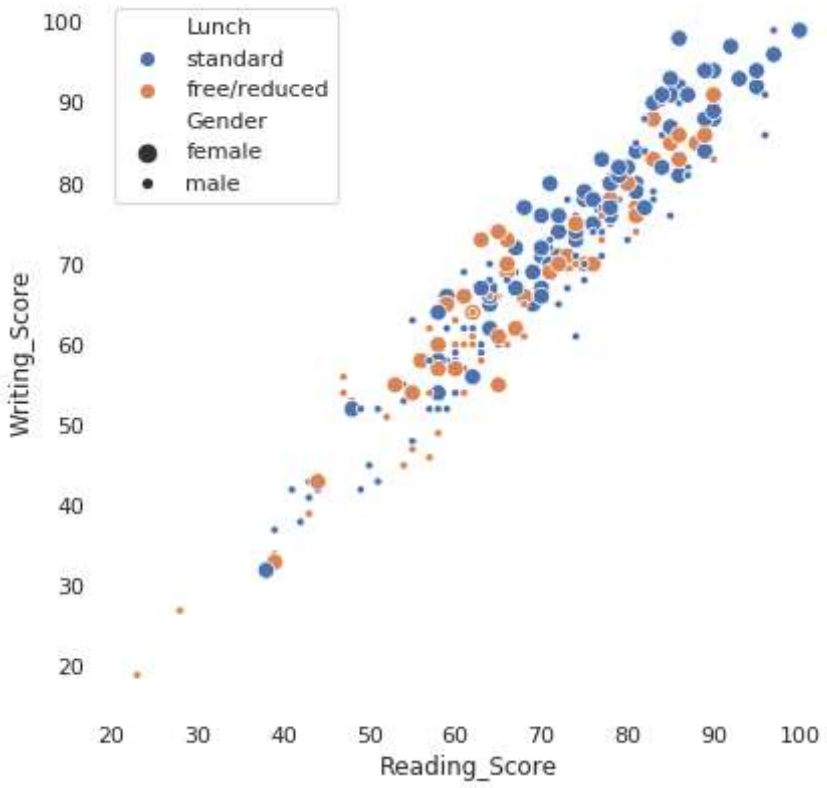
```
data.head()
```

Out[205]:

	Gender	Race/Ethnicity	Parental_Level_of_Education	Lunch	Test_Preparation_Course	Math_Score	Reading_Score	Writing_Score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

In [206…

```
f, ax = plt.subplots(figsize=(6.5, 6.5))
sns.despine(f, left=True, bottom=True)
sns.scatterplot(x="Reading_Score", y="Writing_Score",
                hue="Lunch", size="Gender",data=data[data['Parental_Level_of_Education']=="some college"])
plt.show()
```



In [207…
```
data[np.logical_and(data['Race/Ethnicity']=='group A',data['Parental_Level_of_Education']=='some college')].head()
```

Out[207]:

	Gender	Race/Ethnicity	Parental_Level_of_Education	Lunch	Test_Preparation_Course	Math_Score	Reading_Score	Writing_Score
13	male	group A	some college	standard	completed	78	72	70
82	male	group A	some college	free/reduced	completed	50	47	54
88	female	group A	some college	standard	none	58	70	67
300	male	group A	some college	free/reduced	completed	81	78	81
305	male	group A	some college	standard	none	69	67	69

In [208…
```
f, ax = plt.subplots(figsize=(6.5, 6.5))
sns.despine(f, left=True, bottom=True)
sns.scatterplot(x="Reading_Score", y="Writing_Score",
                hue="Lunch", size="Gender",data=data[np.logical_and(data['Race/Ethnicity']=='group C',data['Parental_Le
plt.show()
```