EDA Students Performance Indicator

1) Problem Statement

- This project understands how the student's performance (test scores) is affected by the other variables such as Gender, Ethinicity, Parental level of education, Lunch and Test prepartion course.

2) Data Collection

- Dataset source - https://www.kaggle.com/datasets/spscientist/students-performance-in-exams?datasetId=74977
- The data consists of 8 column and 1000 rows

3) Dataset Information

- Gender:sex of students->(male/female)
- race/ethinicity:ethinicity of students->(Group A,B,C,D,E)
- Parental level of education : parents' final eduction ->(Bachelor's degree, some college, master's degree, associate's degree, high school)
- lunch: havimg lunch before test(standard or free reduced)
- test preparation course: complete or not complete before test
- math score
- reading score
- writing score

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
## Read the dataset
df=pd.read_csv('stud.csv')
df.head()
```

Out[2]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	

In [3]:
```
df.shape
```

Out[3]:
```
(1000, 8)
```

3. Data Checks to perform

- Check Missing values
- Check duplicates
- Check data type
- Check the number of unique values of each column
- Check statistics of data set
- Check various categories present in the different categorical column

In [4]:
```
## Check missing values
df.isnull().sum()
```

Out[4]:
```
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
```

Insights or Observation

There are no missing values

In [5]:
```
##another way of executing the code to check the missing values

df.isna().sum()
```

Out[5]:
```
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 [6]:
```
## Check duplicates
df.duplicated().sum()
```

Out[6]:
```
0
```

In [8]:
```
df.duplicated()
```

Out[8]:
```
0      False
1      False
2      False
3      False
4      False
       ...
995    False
996    False
997    False
998    False
999    False
Length: 1000, dtype: bool
```

Insights or Observation

There are no duplicates values in the dataset

In [7]:
```
## Check datatypes
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   gender                       1000 non-null   object
 1   race_ethnicity               1000 non-null   object
 2   parental_level_of_education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test_preparation_course      1000 non-null   object
 5   math_score                   1000 non-null   int64 
 6   reading_score                1000 non-null   int64 
 7   writing_score                1000 non-null   int64 
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

In [9]:
```
## 3.1 Checking the number of unique values of each columns
df.nunique()
```

Out[9]:
```
gender                         2
race_ethnicity                 5
parental_level_of_education    6
lunch                          2
test_preparation_course        2
math_score                    81
reading_score                 72
writing_score                 77
dtype: int64
```

In [10]:
```
## Check the statistics of the dataset
df.describe()
```

Out[10]:

	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

Insights or Observation

- From the above description of numerical data, all means are very close to each other- between 66 and 69
- All the standard deviation are also close - between 14.6-15.9

- While there is a minimum of 0 for maths, others are having 17 and 10 value

In [11]:
```
## Explore more info about the data
df.head()
```

Out[11]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	

In [18]:
```
[feature for feature in df.columns]
```

Out[18]:
```
['gender',
 'race_ethnicity',
 'parental_level_of_education',
 'lunch',
 'test_preparation_course',
 'math_score',
 'reading_score',
 'writing_score']
```

In [19]:
```
[feature for feature in df.columns if df[feature].dtype!='O']
```

Out[19]:
```
['math_score', 'reading_score', 'writing_score']
```

In [20]:
```
## Segrregate numerical and ctegorical features

numerical_features=[feature for feature in df.columns if df[feature].dtype!='O']
categorical_features=[feature for feature in df.columns if df[feature].dtype=='O']
```

In [21]:
```
numerical_features
```

Out[21]:
```
['math_score', 'reading_score', 'writing_score']
```

In [22]:
```
categorical_features
```

Out[22]:
```
['gender',
 'race_ethnicity',
 'parental_level_of_education',
 'lunch',
 'test_preparation_course']
```

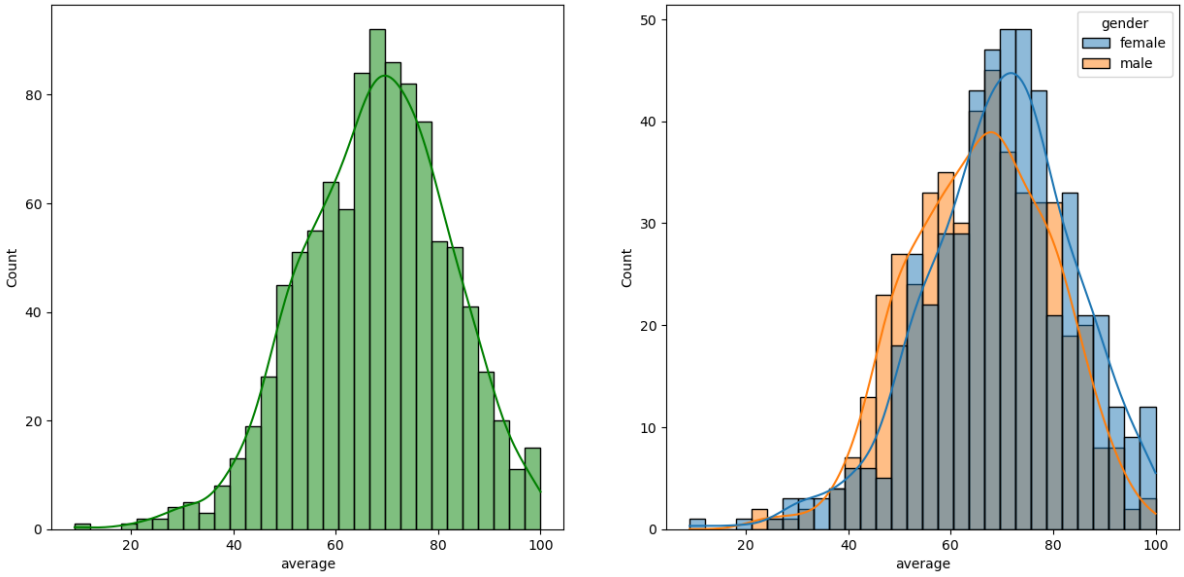
In [23]:
```
## Aggregate the total score with mean

df['total_score']=(df['math_score']+df['reading_score']+df['writing_score'])
df['average']=df['total_score']/3
df.head()
```

Out[23]:

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_
0	female	group B	bachelor's degree	standard	none	
1	female	group C	some college	standard	completed	
2	female	group B	master's degree	standard	none	
3	male	group A	associate's degree	free/reduced	none	
4	male	group C	some college	standard	none	

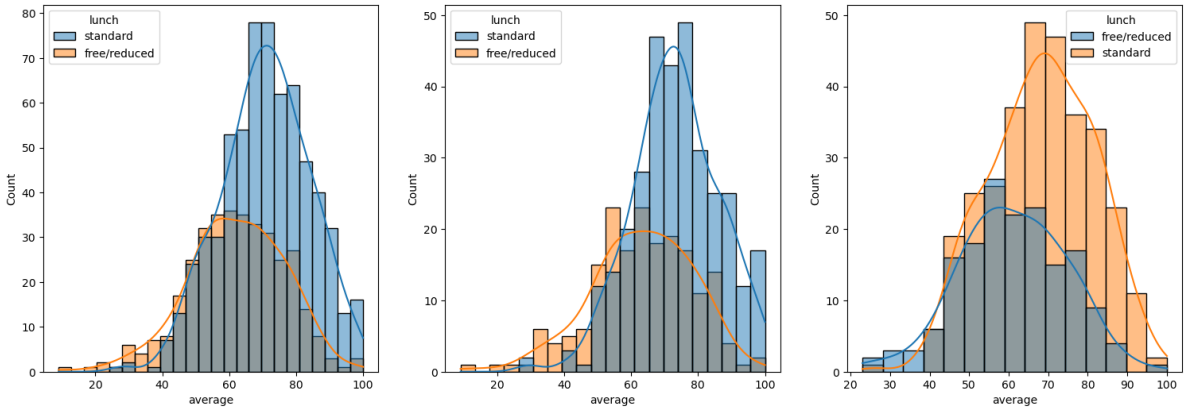
In [28]:
```
## Explore more visualization
fig,axis=plt.subplots(1,2,figsize=(15,7))
plt.subplot(121)
sns.histplot(data=df,x='average', bins=30,kde=True,color='g')
plt.subplot(122)
sns.histplot(data=df,x='average',bins=30,kde=True,hue='gender')
plt.show()
```



Insights

- Female students tend to perform well than male students

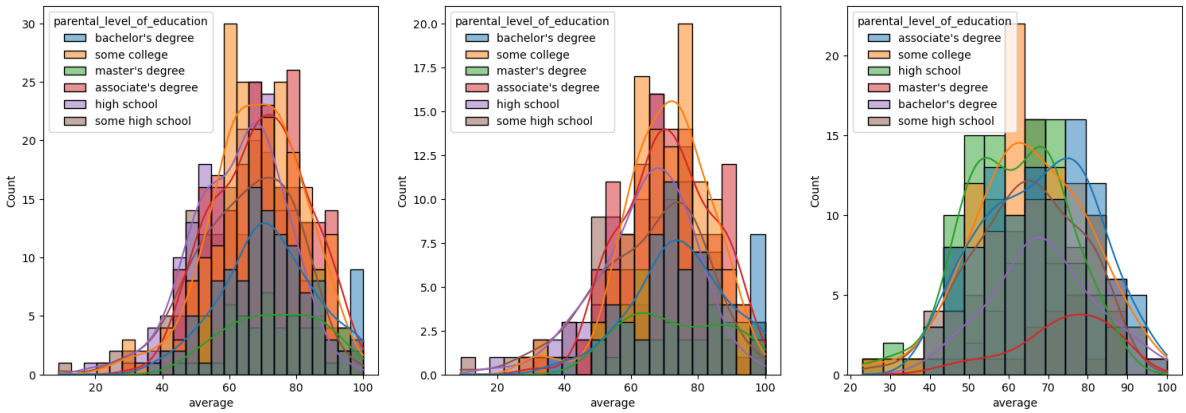
In [30]:
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
sns.histplot(data=df,x='average',kde=True,hue='lunch')
plt.subplot(142)
sns.histplot(data=df[df.gender=='female'],x='average',kde=True,hue='lunch')
plt.subplot(143)
sns.histplot(data=df[df.gender=='male'],x='average', kde=True,hue='lunch')
plt.show()
```



Insights

- Standard lunch helps students to perform well
- Standard lunch helps perform well im the exams be it a male or female

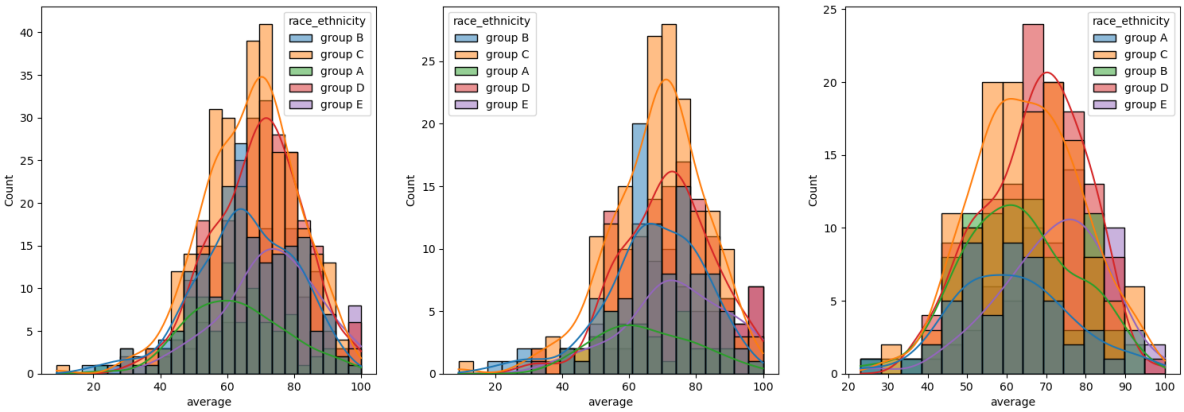
In [31]:
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
sns.histplot(data=df,x='average',kde=True,hue='parental_level_of_education')
plt.subplot(142)
sns.histplot(data=df[df.gender=='female'],x='average',kde=True,hue='parental_level
plt.subplot(143)
sns.histplot(data=df[df.gender=='male'],x='average', kde=True,hue='parental_level_
plt.show()
```



Insights

- In general parent's education doesn't help student perform well
- 2nd plot shows that parent's whosse education is of associate's degree or master's degree their male child tend to perform well in exam
- 3rd plot we can see there is no effect of parent's education on female students

In [32]:
```
plt.subplots(1,3,figsize=(25,6))
plt.subplot(141)
ax= sns.histplot(data=df,x='average',kde=True,hue='race_ethnicity')
plt.subplot(142)
ax=sns.histplot(data=df[df.gender=='female'],x='average',kde=True,hue='race_ethnic
plt.subplot(143)
sns.histplot(data=df[df.gender=='male'],x='average', kde=True,hue='race_ethnicity'
plt.show()
```

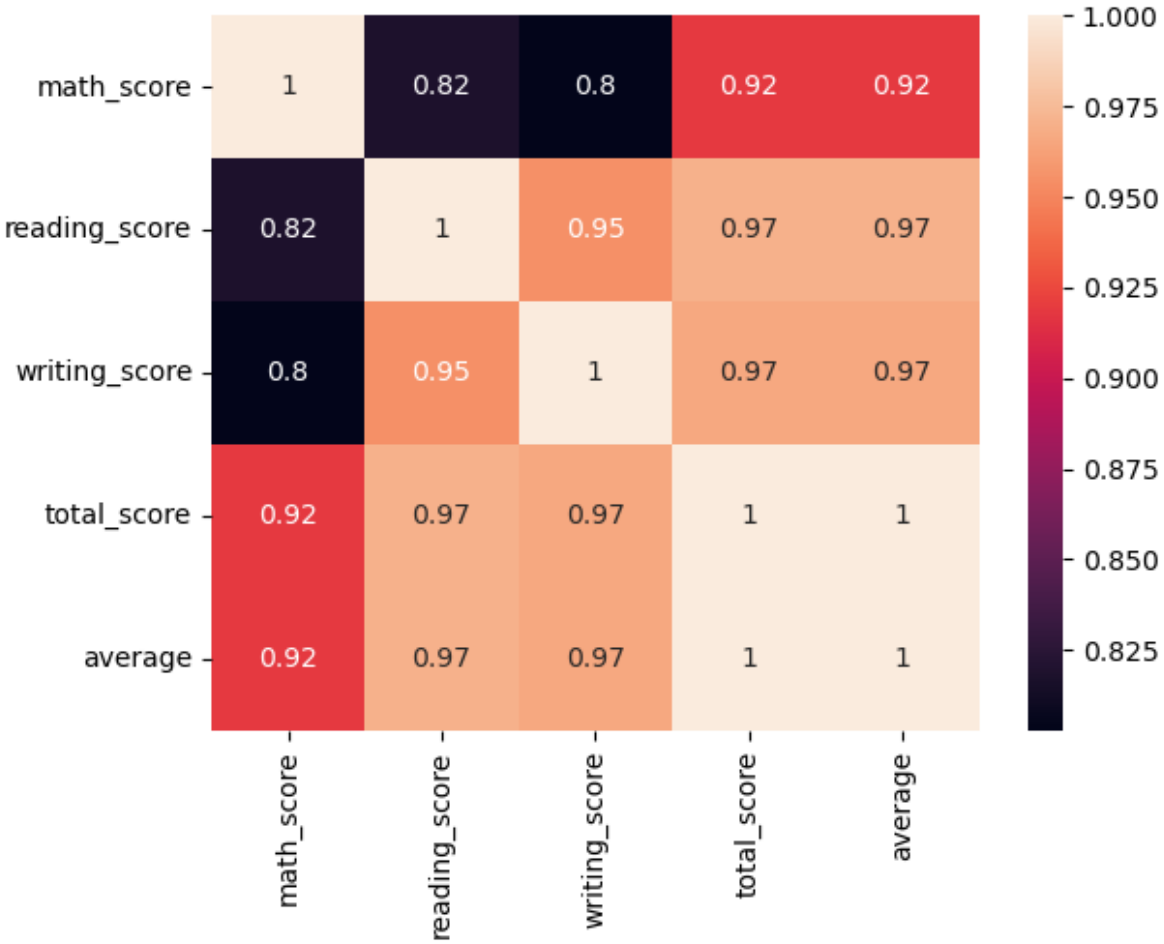


Insights

- Students of group A and group B tends to perform poorlyin exam
- Students of group A and group B tends to performpoorly in exam irrespective of whether they are male or female.

In [35]:
```
sns.heatmap(df.corr(),annot=True)
```

Out[35]:
```
<AxesSubplot: >
```



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