In [ ]:
```
# MANISH KUMAR 7:30 PM
```

In [17]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import scipy
```

In [19]:
```
df=pd.read_csv("StudentPerformance.csv")
df
```

Out[19]:

	Hours_Studied	Attendance	Parental_Involvement	Access_to_Resources	Extracurricula
0	23	84	Low	High	
1	19	64	Low	Medium	
2	24	98	Medium	Medium	
3	29	89	Low	Medium	
4	19	92	Medium	Medium	
...	...	...	...	...	
6602	25	69	High	Medium	
6603	23	76	High	Medium	
6604	20	90	Medium	Low	
6605	10	86	High	High	
6606	15	67	Medium	Low	

6607 rows × 20 columns

steps1 Business problem undersranding

In [4]:
```
# To improve the Student performance in the examination
```

steps2 data understanding and exploration

In [21]:
```
df.columns.to_list()
```

Out[21]:
```
['Hours_Studied',
 'Attendance',
 'Parental_Involvement',
 'Access_to_Resources',
 'Extracurricular_Activities',
 'Sleep_Hours',
 'Previous_Scores',
 'Motivation_Level',
 'Internet_Access',
 'Tutoring_Sessions',
 'Family_Income',
 'Teacher_Quality',
 'School_Type',
 'Peer_Influence',
 'Physical_Activity',
 'Learning_Disabilities',
 'Parental_Education_Level',
 'Distance_from_Home',
 'Gender',
 'Exam_Score']
```

In [23]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6607 entries, 0 to 6606
Data columns (total 20 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Hours_Studied               6607 non-null   int64
 1   Attendance                  6607 non-null   int64
 2   Parental_Involvement        6607 non-null   object
 3   Access_to_Resources         6607 non-null   object
 4   Extracurricular_Activities  6607 non-null   object
 5   Sleep_Hours                 6607 non-null   int64
 6   Previous_Scores             6607 non-null   int64
 7   Motivation_Level            6607 non-null   object
 8   Internet_Access             6607 non-null   object
 9   Tutoring_Sessions           6607 non-null   int64
 10  Family_Income               6607 non-null   object
 11  Teacher_Quality             6529 non-null   object
 12  School_Type                 6607 non-null   object
 13  Peer_Influence              6607 non-null   object
 14  Physical_Activity           6607 non-null   int64
 15  Learning_Disabilities       6607 non-null   object
 16  Parental_Education_Level    6517 non-null   object
 17  Distance_from_Home          6540 non-null   object
 18  Gender                      6607 non-null   object
 19  Exam_Score                  6607 non-null   int64
dtypes: int64(7), object(13)
memory usage: 1.0+ MB
```

In [11]:
```
df.shape
```

Out[11]:
```
(6607, 20)
```

In [13]:
```
df.dtypes
```

Out[13]:
```
Hours_Studied                int64
Attendance                   int64
Parental_Involvement        object
Access_to_Resources         object
Extracurricular_Activities  object
Sleep_Hours                  int64
Previous_Scores              int64
Motivation_Level            object
Internet_Access             object
Tutoring_Sessions            int64
Family_Income               object
Teacher_Quality             object
School_Type                 object
Peer_Influence              object
Physical_Activity            int64
Learning_Disabilities       object
Parental_Education_Level    object
Distance_from_Home          object
Gender                      object
Exam_Score                   int64
dtype: object
```

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
Hours_Studied                0
Attendance                   0
Parental_Involvement         0
Access_to_Resources          0
Extracurricular_Activities   0
Sleep_Hours                  0
Previous_Scores              0
Motivation_Level             0
Internet_Access              0
Tutoring_Sessions            0
Family_Income                0
Teacher_Quality             78
School_Type                  0
Peer_Influence               0
Physical_Activity            0
Learning_Disabilities        0
Parental_Education_Level    90
Distance_from_Home          67
Gender                       0
Exam_Score                   0
dtype: int64
```

explore column wise

Hours_Studied

In [13]:
```
# this coulums is continous variable and correct data type & there is no null value
```

In [28]:
```
df.rename(columns={"Hours_Studied":"Studied_Hours"},inplace=True)
df.head(2)
```

Out[28]:

	Studied_Hours	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_A
0	23	84	Low	High	
1	19	64	Low	Medium	

In [7]:
```
df["Studied_Hours"].isnull().sum()
```

Out[7]:
```
0
```

In [30]:
```
df["Studied_Hours"].dtypes
```

Out[30]:
```
dtype('int64')
```

Attendence

In [ ]:
```
# this is continous variable& correct data type & there is no null value &
# columns name is correct.
```

In [9]:
```
df["Attendance"].dtypes
```

Out[9]:
```
dtype('int64')
```

In [11]:
```
df["Attendance"].isnull().sum()
```

Out[11]:
```
0
```

Parental_Involvement

In [ ]:
```
# this is discrete variable & correct data type & there is no null value & columns
# there is 3 unique value(low,medium, high)
# medium value more then this two value and low value is too less
```

In [13]:
```
df["Parental_Involvement"].dtypes
```

Out[13]:
```
dtype('O')
```

In [15]:
```
df["Parental_Involvement"].isnull().sum()
```

Out[15]:
```
0
```

In [27]:
```
df["Parental_Involvement"].unique()
```

Out[27]:
```
array(['Low', 'Medium', 'High'], dtype=object)
```

In [29]:
```
df["Parental_Involvement"].value_counts()
```

Out[29]:
```
Parental_Involvement
Medium    3362
High      1908
Low       1337
Name: count, dtype: int64
```

Access_to_Resources

In [ ]:
```
# this is a discrete  variable and correct data types and there is no null value
# there is 3 unique value (low,medium and high) meduem value is more & low value is
```

In [31]:
```
df["Access_to_Resources"].dtypes
```

Out[31]:
```
dtype('O')
```

In [33]:
```
df["Access_to_Resources"].isnull().sum()
```

Out[33]:
```
0
```

In [35]:
```
df["Access_to_Resources"].unique()
```

Out[35]:
```
array(['High', 'Medium', 'Low'], dtype=object)
```

In [37]:
```
df["Access_to_Resources"].value_counts()
```

Out[37]:
```
Access_to_Resources
Medium    3319
High      1975
Low       1313
Name: count, dtype: int64
```

Extracurricular_Activities

In [ ]:
```
# this is discerete variable and correct data types & there is no null value
# there is 2 unique value (yes and no) yes>no
```

In [113…
```
df["Extracurricular_Activities"].dtypes
```

Out[113…
```
dtype('O')
```

In [115…
```
df["Extracurricular_Activities"].isnull().sum()
```

Out[115…
```
0
```

In [117…
```
df["Extracurricular_Activities"].unique()
```

Out[117… array(['No', 'Yes'], dtype=object)

In [119…
```
df["Extracurricular_Activities"].value_counts()
```

Out[119…
```
Extracurricular_Activities
Yes    3938
No     2669
Name: count, dtype: int64
```

Sleep_Hours

In [ ]:
```
# this is continous variable and correct data types & there is no null value
```

In [121…
```
df["Sleep_Hours"].dtypes
```

Out[121… dtype('int64')

In [127…
```
df["Sleep_Hours"].isnull().sum()
```

Out[127… 0

Previous_Scores

In [ ]:
```
# this is continous variable and correct data types & there is no null value
```

In [129…
```
df["Previous_Scores"].dtypes
```

Out[129… dtype('int64')

In [32]:
```
df["Previous_Scores"].isnull().sum()
```

Out[32]: 0

Motivation_Level

In [ ]:
```
# this is continous variable and correct data types & there is no null value
# there is 3 unique value(medium>low>high)
```

In [34]:
```
df["Motivation_Level"].dtypes
```

Out[34]: dtype('O')

In [36]:
```
df["Motivation_Level"].unique()
```

Out[36]: array(['Low', 'Medium', 'High'], dtype=object)

In [38]:
```
df["Motivation_Level"].value_counts()
```

Out[38]:
```
Motivation_Level
Medium    3351
Low       1937
High      1319
Name: count, dtype: int64
```

In [40]:
```
df["Motivation_Level"].isnull().sum()
```

Out[40]:
```
0
```

Internet_Access

In [ ]:
```
# this is discrete variable and correct data types & there is no null value
# there is 2 unique value(yes and no) yes=6108 and no=499
```

In [42]:
```
df["Internet_Access"].dtypes
```

Out[42]:
```
dtype('O')
```

In [44]:
```
df["Internet_Access"].unique()
```

Out[44]:
```
array(['Yes', 'No'], dtype=object)
```

In [46]:
```
df["Internet_Access"].value_counts()
```

Out[46]:
```
Internet_Access
Yes    6108
No      499
Name: count, dtype: int64
```

Tutoring_Sessions

In [ ]:
```
# this is continous variable and correct data types & there is no null value
```

In [48]:
```
df["Tutoring_Sessions"].dtypes
```

Out[48]:
```
dtype('int64')
```

In [50]:
```
df["Tutoring_Sessions"].unique()
```

Out[50]:
```
array([0, 2, 1, 3, 4, 5, 6, 7, 8], dtype=int64)
```

In [52]:
```
df["Tutoring_Sessions"].value_counts()
```

Out[52]:
```
Tutoring_Sessions
1    2179
2    1649
0    1513
3     836
4     301
5     103
6      18
7       7
8       1
Name: count, dtype: int64
```

In [54]:
```
df["Tutoring_Sessions"].isnull().sum()
```

Out[54]:
```
0
```

Family_Income

In [ ]:
```
# this is discrite variable and correct data types & there is no null value
# there is  3 unique value(low,medium,high) low>medium>high
```

In [56]:
```
df["Family_Income"].dtypes
```

Out[56]:
```
dtype('O')
```

In [58]:
```
df["Family_Income"].unique()
```

Out[58]:
```
array(['Low', 'Medium', 'High'], dtype=object)
```

In [60]:
```
df["Family_Income"].value_counts()
```

Out[60]:
```
Family_Income
Low       2672
Medium    2666
High      1269
Name: count, dtype: int64
```

In [62]:
```
df["Family_Income"].isnull().sum()
```

Out[62]:
```
0
```

Teacher_Quality

In [ ]:
```
# this is count/discrete variable and correct data types & there is some missing va
# there is  3 unique value(low,medium,high)medium>high>low
```

In [64]:
```
df["Teacher_Quality"].dtypes
```

Out[64]:
```
dtype('O')
```

In [66]:
```
df["Teacher_Quality"].unique()
```

Out[66]:
```
array(['Medium', 'High', 'Low', nan], dtype=object)
```

In [68]:
```
df["Teacher_Quality"].value_counts()
```

Out[68]:
```
Teacher_Quality
Medium    3925
High      1947
Low        657
Name: count, dtype: int64
```

In [70]:
```
df["Teacher_Quality"].isnull().sum()
```

Out[70]:
```
78
```

School_Type

In [ ]:
```
# this is discrete variable and correct data types & there is no null value
# there is  2 unique value(publicand private) public>private
```

In [72]:
```
df["School_Type"].dtypes
```

Out[72]:
```
dtype('O')
```

In [74]:
```
df["School_Type"].unique()
```

Out[74]:
```
array(['Public', 'Private'], dtype=object)
```

In [76]:
```
df["School_Type"].value_counts()
```

Out[76]:
```
School_Type
Public     4598
Private    2009
Name: count, dtype: int64
```

In [78]:
```
df["School_Type"].isnull().sum()
```

Out[78]:
```
0
```

In [80]:
```
df.head(2)
```

Out[80]:

	Studied_Hours	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_A
0	23	84	Low	High	
1	19	64	Low	Medium	

Peer_Influence

In [ ]:
```
# this is discrete variable and correct data types & there is no missing value
# there is  3 unique value(+ve,-ve,neutral) +ve> neutral> -ve
```

In [82]:
```
df["Peer_Influence"].dtypes
```

Out[82]:
```
dtype('O')
```

In [84]:
```
df["Peer_Influence"].unique()
```

Out[84]:
```
array(['Positive', 'Negative', 'Neutral'], dtype=object)
```

In [86]:
```
df["Peer_Influence"].value_counts()
```

Out[86]:
```
Peer_Influence
Positive    2638
Neutral     2592
Negative    1377
Name: count, dtype: int64
```

In [88]:
```
df["Peer_Influence"].isnull().sum()
```

Out[88]:
```
0
```

Physical_Activity

In [ ]:
```
# this is continous variable and correct data types & there is some missing value
```

In [90]:
```
df["Physical_Activity"].dtypes
```

Out[90]:
```
dtype('int64')
```

In [92]:
```
df["Physical_Activity"].unique()
```

Out[92]:
```
array([3, 4, 2, 1, 5, 0, 6], dtype=int64)
```

In [94]:
```
df["Physical_Activity"].value_counts()
```

Out[94]:
```
Physical_Activity
3    2545
2    1627
4    1575
1     421
5     361
0      46
6      32
Name: count, dtype: int64
```

In [137…
```
df["Physical_Activity"].isnull().sum()
```

Out[137…
```
0
```

Learning_Disabilities

In [ ]:
```
# this is discrete variable and correct data types & there is no null value
# there is 2 unique value(yes and no) yes=6108 and no=499
```

In [139…
```
df["Learning_Disabilities"].dtypes
```

Out[139…
```
dtype('O')
```

In [141…
```
df["Learning_Disabilities"].unique()
```

Out[141…
```
array(['No', 'Yes'], dtype=object)
```

In [143…
```
df["Learning_Disabilities"].value_counts()
```

Out[143…
```
Learning_Disabilities
No     5912
Yes     695
Name: count, dtype: int64
```

In [145…
```
df["Learning_Disabilities"].isnull().sum()
```

Out[145…
```
0
```

Parental_Education_Level

In [ ]:
```
# this is discrete variable and correct data types & there is some missing value
# there is  3 unique value(college,highschool,postgraduate)
```

In [147…
```
df["Parental_Education_Level"].dtypes
```

Out[147…
```
dtype('O')
```

In [149…
```
df["Parental_Education_Level"].unique()
```

Out[149…
```
array(['High School', 'College', 'Postgraduate', nan], dtype=object)
```

In [151…
```
df["Parental_Education_Level"].value_counts()
```

Out[151…
```
Parental_Education_Level
High School     3223
College         1989
Postgraduate    1305
Name: count, dtype: int64
```

In [153…
```
df["Parental_Education_Level"].isnull().sum()
```

Out[153…
```
90
```

Distance_from_Home

In [ ]:
```
# this is discrete variable and correct data types & there is some  missing value
# there is  3 unique value(near,moderate,far)
```

In [155…
```
df["Distance_from_Home"].dtypes
```

Out[155…
```
dtype('O')
```

In [157…
```
df["Distance_from_Home"].unique()
```

Out[157…
```
array(['Near', 'Moderate', 'Far', nan], dtype=object)
```

In [159…
```
df["Distance_from_Home"].value_counts()
```

Out[159…
```
Distance_from_Home
Near        3884
Moderate    1998
Far          658
Name: count, dtype: int64
```

In [161…
```
df["Distance_from_Home"].isnull().sum()
```

Out[161…
```
67
```

In [163…
```
df.head(2)
```

Out[163…

	Studied_Hours	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_A
0	23	84	Low	High	
1	19	64	Low	Medium	

Gender

In [ ]:
```
# this is discrete variable and correct data types & there is no null value
# there is 2 unique value(male,female) male> female
```

In [165…
```
df["Gender"].dtypes
```

Out[165…
```
dtype('O')
```

In [167…
```
df["Gender"].unique()
```

Out[167…
```
array(['Male', 'Female'], dtype=object)
```

In [169…
```
df["Gender"].value_counts()
```

Out[169…
```
Gender
Male      3814
Female    2793
Name: count, dtype: int64
```

In [171…
```
df["Gender"].isnull().sum()
```

Out[171…
```
0
```

Exam_Score

In [ ]:
```
# this is continous variable and correct data types & there is no null value
```

In [173…
```
df["Exam_Score"].dtypes
```

Out[173…
```
dtype('int64')
```

In [177…
```
df["Exam_Score"].isnull().sum()
```

Out[177…
```
0
```

In [179…
```
df.columns.to_list()
```

Out[179…
```
['Studied_Hours',
 'Attendance',
 'Parental_Involvement',
 'Access_to_Resources',
 'Extracurricular_Activities',
 'Sleep_Hours',
 'Previous_Scores',
 'Motivation_Level',
 'Internet_Access',
 'Tutoring_Sessions',
 'Family_Income',
 'Teacher_Quality',
 'School_Type',
 'Peer_Influence',
 'Physical_Activity',
 'Learning_Disabilities',
 'Parental_Education_Level',
 'Distance_from_Home',
 'Gender',
 'Exam_Score']
```

In [96]:
```
discrete=['Studied_Hours','Attendance','Parental_Involvement','Access_to_Resources'
 'Extracurricular_Activities','Sleep_Hours','Previous_Scores','Motivation_Level',
 'Internet_Access','Tutoring_Sessions','Family_Income','Teacher_Quality','School_Ty
 'Peer_Influence','Physical_Activity','Learning_Disabilities','Parental_Education_L
 'Distance_from_Home','Gender','Exam_Score']
```

In [183…
```
discrete
```

Out[183…
```
['Studied_Hours',
 'Attendance',
 'Parental_Involvement',
 'Access_to_Resources',
 'Extracurricular_Activities',
 'Sleep_Hours',
 'Previous_Scores',
 'Motivation_Level',
 'Internet_Access',
 'Tutoring_Sessions',
 'Family_Income',
 'Teacher_Quality',
 'School_Type',
 'Peer_Influence',
 'Physical_Activity',
 'Learning_Disabilities',
 'Parental_Education_Level',
 'Distance_from_Home',
 'Gender',
 'Exam_Score']
```

step 3 Data Preprocessing

In [ ]:
```
# Data cleaning
# demension reduction
# Data transformation
```

In [185…
```
df.duplicated().sum()
```

Out[185…
```
0
```

In [187…
```
df.isnull().sum()
```

Out[187…
```
Studied_Hours                 0
Attendance                    0
Parental_Involvement          0
Access_to_Resources           0
Extracurricular_Activities    0
Sleep_Hours                   0
Previous_Scores               0
Motivation_Level              0
Internet_Access               0
Tutoring_Sessions             0
Family_Income                 0
Teacher_Quality              78
School_Type                   0
Peer_Influence                0
Physical_Activity             0
Learning_Disabilities         0
Parental_Education_Level     90
Distance_from_Home           67
Gender                        0
Exam_Score                    0
dtype: int64
```

In [189…
```
# in this columns some missing value
missing_value=["Teacher_Quality","Parental_Education_Level","Distance_from_Home"]
```

In [29]:
```
# fiil the missing value with mode
df["Teacher_Quality"]=df["Teacher_Quality"].fillna(df["Teacher_Quality"].mode()[0])
df["Parental_Education_Level"]=df["Parental_Education_Level"].fillna(df["Parental_E
df["Distance_from_Home"]=df["Distance_from_Home"].fillna(df["Distance_from_Home"].m
```

In [ ]:
```
```

In [31]:
```
df.isnull().sum()
```

Out[31]:
```
Studied_Hours                0
Attendance                   0
Parental_Involvement         0
Access_to_Resources          0
Extracurricular_Activities   0
Sleep_Hours                  0
Previous_Scores              0
Motivation_Level             0
Internet_Access              0
Tutoring_Sessions            0
Family_Income                0
Teacher_Quality              0
School_Type                  0
Peer_Influence               0
Physical_Activity            0
Learning_Disabilities        0
Parental_Education_Level     0
Distance_from_Home           0
Gender                       0
Exam_Score                   0
dtype: int64
```

step 4 & 5 Data analysis & visualization

In [33]:
```
df.head(2)
```

Out[33]:

	Studied_Hours	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_A
0	23	84	Low	High	
1	19	64	Low	Medium	

In [35]:
```
df.describe()
```

Out[35]:

	Studied_Hours	Attendance	Sleep_Hours	Previous_Scores	Tutoring_Sessions	Physi
count	6607.000000	6607.000000	6607.00000	6607.000000	6607.000000	6
mean	19.975329	79.977448	7.02906	75.070531	1.493719	
std	5.990594	11.547475	1.46812	14.399784	1.230570	
min	1.000000	60.000000	4.00000	50.000000	0.000000	
25%	16.000000	70.000000	6.00000	63.000000	1.000000	
50%	20.000000	80.000000	7.00000	75.000000	1.000000	
75%	24.000000	90.000000	8.00000	88.000000	2.000000	
max	44.000000	100.000000	10.00000	100.000000	8.000000	

In [39]:
```
df.describe(include="object").T
```

Out[39]:

	count	unique	top	freq
Parental_Involvement	6607	3	Medium	3362
Access_to_Resources	6607	3	Medium	3319
Extracurricular_Activities	6607	2	Yes	3938
Motivation_Level	6607	3	Medium	3351
Internet_Access	6607	2	Yes	6108
Family_Income	6607	3	Low	2672
Teacher_Quality	6607	3	Medium	4003
School_Type	6607	2	Public	4598
Peer_Influence	6607	3	Positive	2638
Learning_Disabilities	6607	2	No	5912
Parental_Education_Level	6607	3	High School	3313
Distance_from_Home	6607	3	Near	3951
Gender	6607	2	Male	3814

In [187…

```
score=df[["Studied_Hours","Exam_Score"]].corr()
score
```

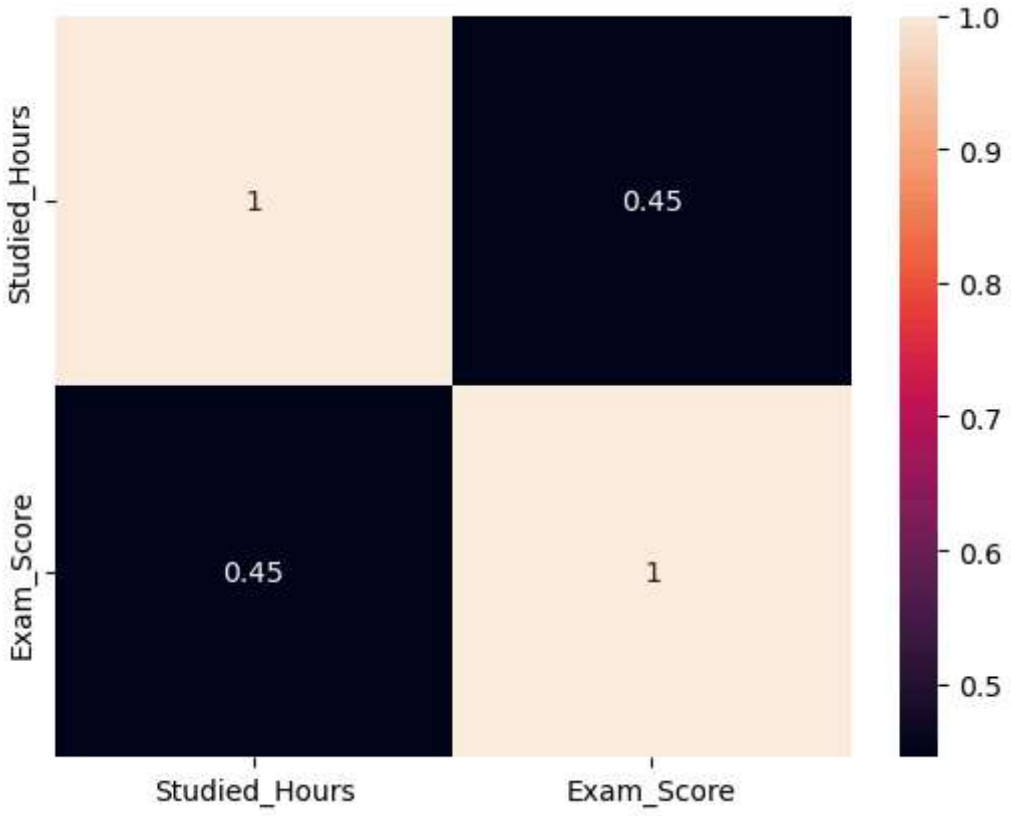
Out[187…

	Studied_Hours	Exam_Score
Studied_Hours	1.000000	0.445455
Exam_Score	0.445455	1.000000

In [189…

```
sns.heatmap(score ,annot=True)
```

Out[189… <Axes: >



In [ ]:
```
# the correlation b/w studied_ hours and Exam_score is weak correlation
# so that there is some effect on exam_score  due to the studied_hours.
```

In [ ]:
```
```

In [215…
```
df[["Attendance","Exam_Score"]].corr()
```

Out[215…

	Attendance	Exam_Score
Attendance	1.000000	0.581072
Exam_Score	0.581072	1.000000

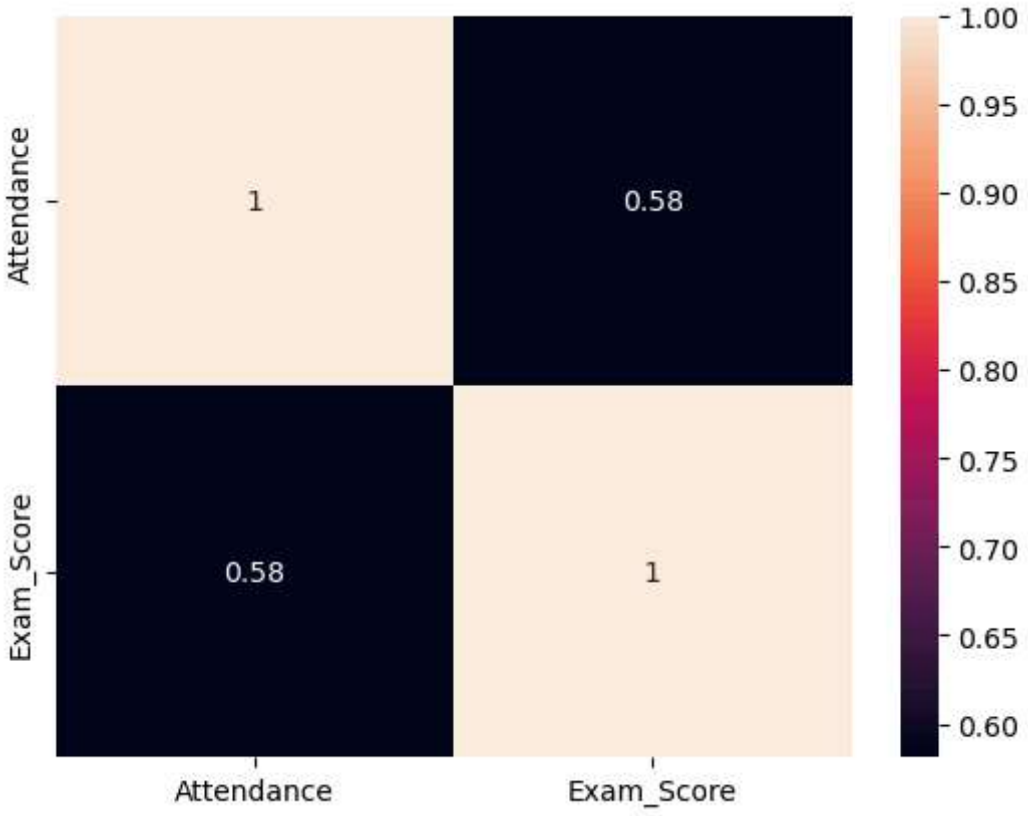
In [197…
```
score= df[["Attendance","Exam_Score"]].corr()
score
```

Out[197…

	Attendance	Exam_Score
Attendance	1.000000	0.581072
Exam_Score	0.581072	1.000000

In [199…
```
sns.heatmap(score ,annot=True)
```

Out[199… <Axes: >



In [ ]:
```
# the correlation b/w attendance and Exam_score is moderate correlation
# so that there is few effect on exam_score  due to the attendance.
```

In [ ]:
```
```

In [131…
```
df.groupby("Parental_Involvement")["Exam_Score"].min()
```

Out[131…
```
Parental_Involvement
High      57
Low       56
Medium    55
Name: Exam_Score, dtype: int64
```

In [133…
```
df.groupby("Parental_Involvement")["Exam_Score"].max()
```

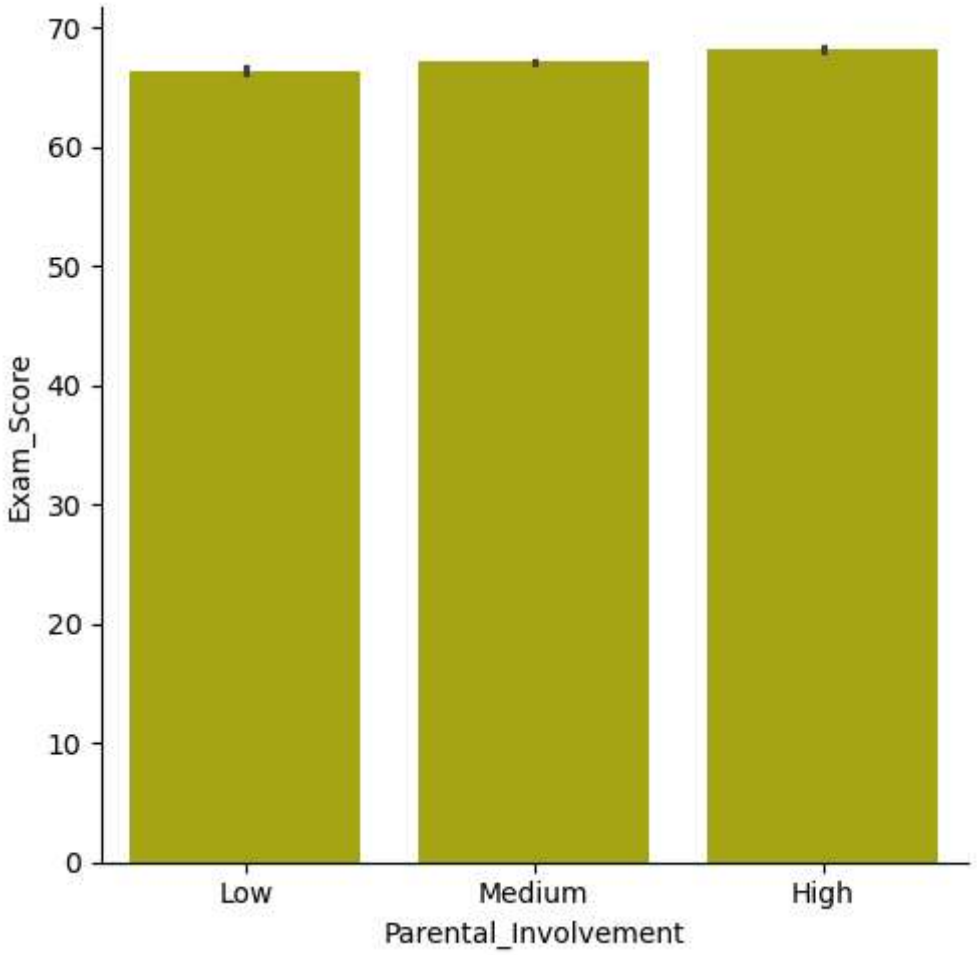
Out[133…
```
Parental_Involvement
High      100
Low       101
Medium     97
Name: Exam_Score, dtype: int64
```

In [135…
```
df.groupby("Parental_Involvement")["Exam_Score"].mean()
```

Out[135…
```
Parental_Involvement
High      68.092767
Low       66.358265
Medium    67.098156
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Parental_Involvement
# so that there is no effect on exam_score  due to the Parental_Involvement.
```

In [171…
```
sns.catplot(x="Parental_Involvement",y="Exam_Score",data=df,kind="bar",color="y")
plt.show()
```



In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Access_to_Resources
# so that there is no effect on exam_score  due to the Access_to_Resources.
```

In [137…
```
df.groupby("Access_to_Resources")["Exam_Score"].mean()
```

Out[137…
```
Access_to_Resources
High      68.092152
Low       66.203351
Medium    67.134378
Name: Exam_Score, dtype: float64
```

In [141…
```
df.groupby("Access_to_Resources")["Exam_Score"].min()
```

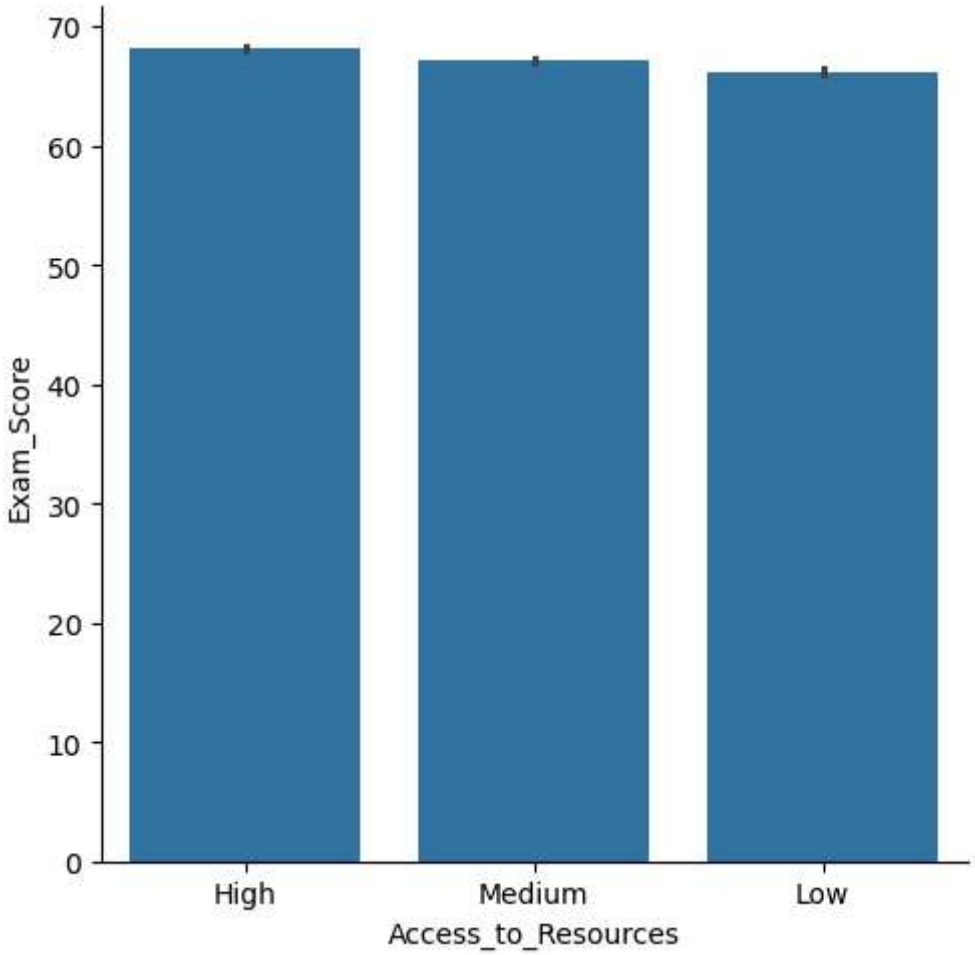
Out[141…
```
Access_to_Resources
High      56
Low       55
Medium    58
Name: Exam_Score, dtype: int64
```

In [139…
```
df.groupby("Access_to_Resources")["Exam_Score"].max()
```

Out[139…
```
Access_to_Resources
High       99
Low        98
Medium    101
Name: Exam_Score, dtype: int64
```

In [ ]:

In [145…
```
sns.catplot(x="Access_to_Resources",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:

In [173…
```
df.groupby("Extracurricular_Activities")["Exam_Score"].min()
```

Out[173…
```
Extracurricular_Activities
No     55
Yes    57
Name: Exam_Score, dtype: int64
```

In [177…
```
df.groupby("Extracurricular_Activities")["Exam_Score"].max()
```

Out[177…
```
Extracurricular_Activities
No      97
Yes    101
Name: Exam_Score, dtype: int64
```

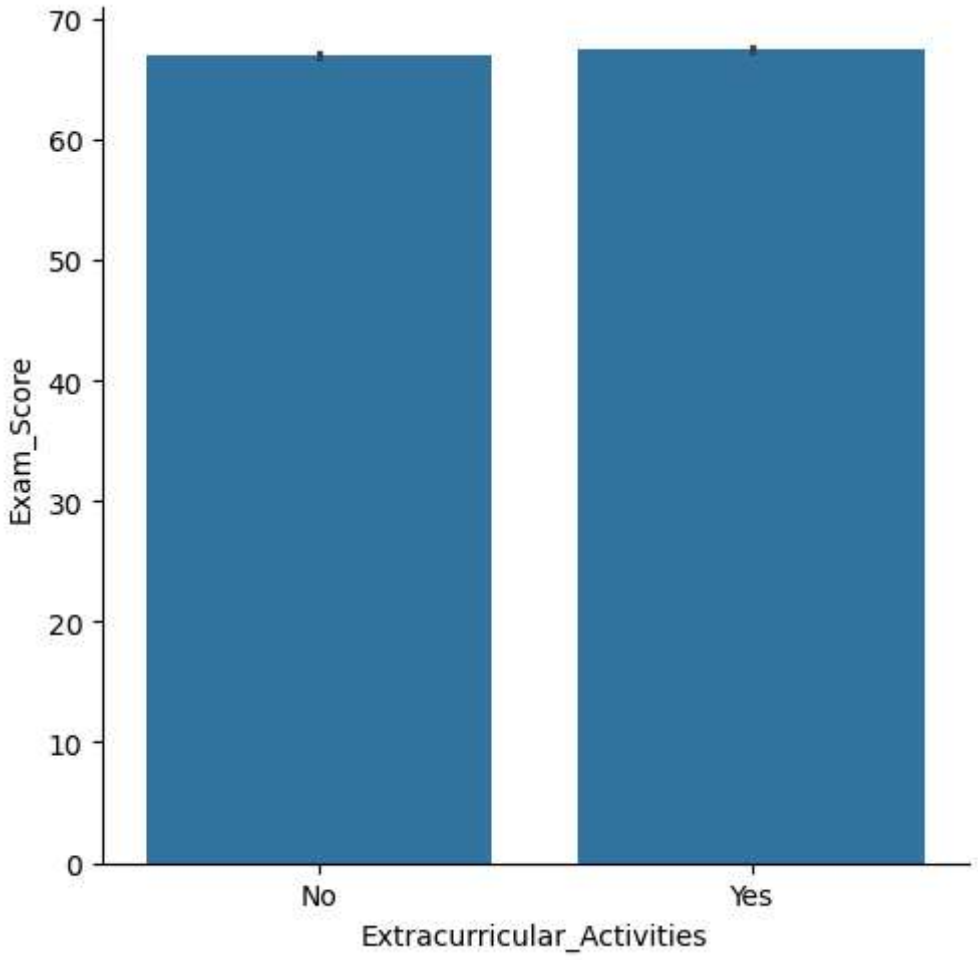
In [175…
```
df.groupby("Extracurricular_Activities")["Exam_Score"].mean()
```

Out[175…
```
Extracurricular_Activities
No     66.931435
Yes    67.441849
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Extracurricular_Activities
# so that there is no effect on exam_score  due to the Extracurricular_Activities.
```

In [183…
```
sns.catplot(x="Extracurricular_Activities",y="Exam_Score",data=df,kind="bar")
```

Out[183…
```
<seaborn.axisgrid.FacetGrid at 0x28ccaa54620>
```



In [ ]:
```

```

In [217…
```
corr=df[["Sleep_Hours","Exam_Score"]].corr()
corr
```

Out[217…

	Sleep_Hours	Exam_Score
Sleep_Hours	1.000000	-0.017022
Exam_Score	-0.017022	1.000000

In [209…
```
sns.heatmap(corr,annot=True)
```

Out[209… <Axes: >

In [ ]:
```
# the correlation b/w Sleep_Hours and Exam_score is (-ve) correlation
# so that there is few effect on exam_score  due to the sleep_hours.
# if student sleep_hours is more atumatically thier exam score is less  that s mean
```

In [ ]:
```

```

In [219…
```
corr=df[["Previous_Scores","Exam_Score"]].corr()
corr
```

Out[219…

	Previous_Scores	Exam_Score
Previous_Scores	1.000000	0.175079
Exam_Score	0.175079	1.000000

In [221…
```
sns.heatmap(corr,annot=True)
```

Out[221… <Axes: >

In [ ]:
```
# the correlation b/w previous_Scores and Exam_score is weak correlation
# so that there is few effect on exam_score  due to the previous_Scores.
```

In [ ]:
```
```

In [223…
```
df.groupby("Motivation_Level")["Exam_Score"].max()
```

Out[223…
```
Motivation_Level
High       98
Low       101
Medium    100
Name: Exam_Score, dtype: int64
```

In [225…
```
df.groupby("Motivation_Level")["Exam_Score"].min()
```

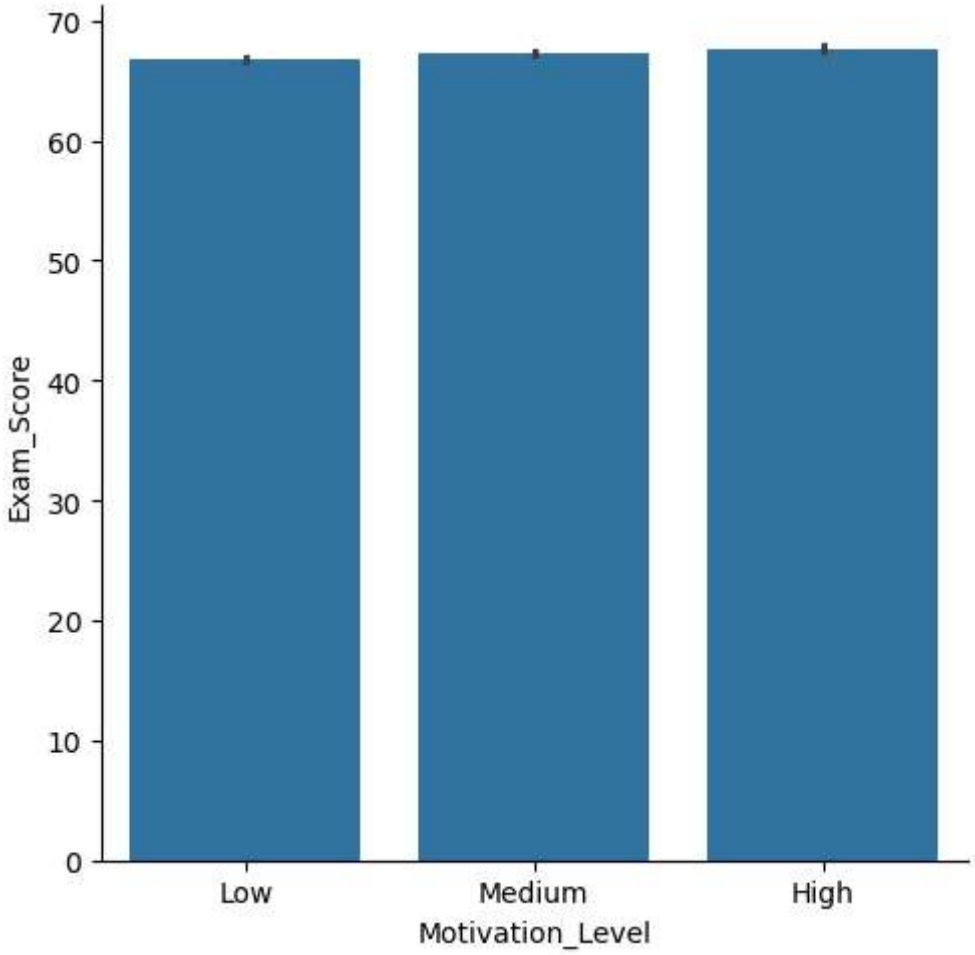
Out[225…
```
Motivation_Level
High      57
Low       57
Medium    55
Name: Exam_Score, dtype: int64
```

In [227…
```
df.groupby("Motivation_Level")["Exam_Score"].mean()
```

Out[227…
```
Motivation_Level
High      67.704321
Low       66.752194
Medium    67.330648
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Motivation_Level
# so that there is no effect on exam_score  due to the Motivation_Level.
```

In [231…
```
sns.catplot(x="Motivation_Level",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [233…
```
df.groupby("Internet_Access")["Exam_Score"].max()
```

Out[233…
```
Internet_Access
No     101
Yes    100
Name: Exam_Score, dtype: int64
```

In [235…
```
df.groupby("Internet_Access")["Exam_Score"].min()
```

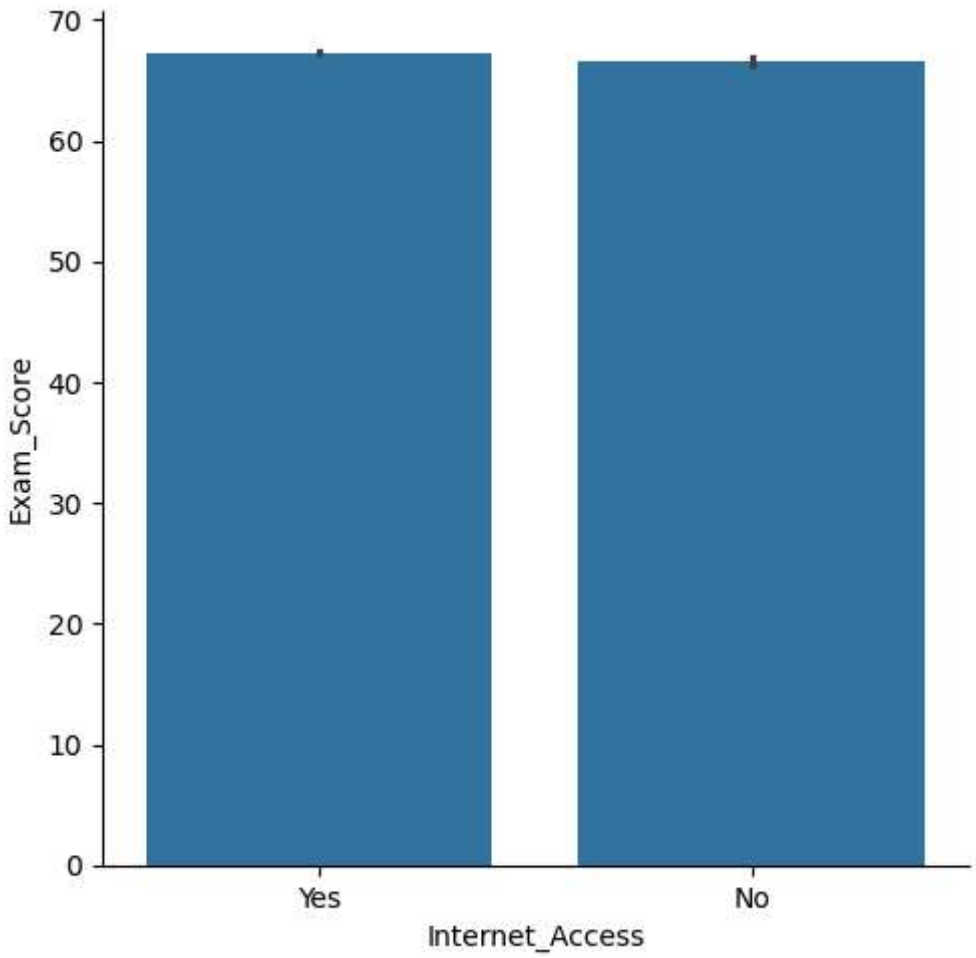
Out[235…
```
Internet_Access
No     58
Yes    55
Name: Exam_Score, dtype: int64
```

In [237…
```
df.groupby("Internet_Access")["Exam_Score"].mean()
```

Out[237…
```
Internet_Access
No     66.535070
Yes    67.292895
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Internet_Access
# so that there is no effect on exam_score  due to the Internet_Access.
```

In [239…
```
sns.catplot(x="Internet_Access",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [245…
```
corr=df[["Tutoring_Sessions","Exam_Score"]].corr()
corr
```

Out[245…

	Tutoring_Sessions	Exam_Score
Tutoring_Sessions	1.000000	0.156525
Exam_Score	0.156525	1.000000

In [249…
```
sns.heatmap(corr,annot=True)
```

Out[249… <Axes: >

In [ ]:
```
# the correlation b/w Tutoring_Sessions and Exam_score is weak correlation
# so that there is few effect on exam_score  due to the Tutoring_Sessions.
```

In [ ]:
```
```

In [251…
```
df.groupby("Family_Income")["Exam_Score"].max()
```

Out[251…
```
Family_Income
High      101
Low        97
Medium     99
Name: Exam_Score, dtype: int64
```

In [275…
```
df.groupby("Family_Income")["Exam_Score"].min()
```

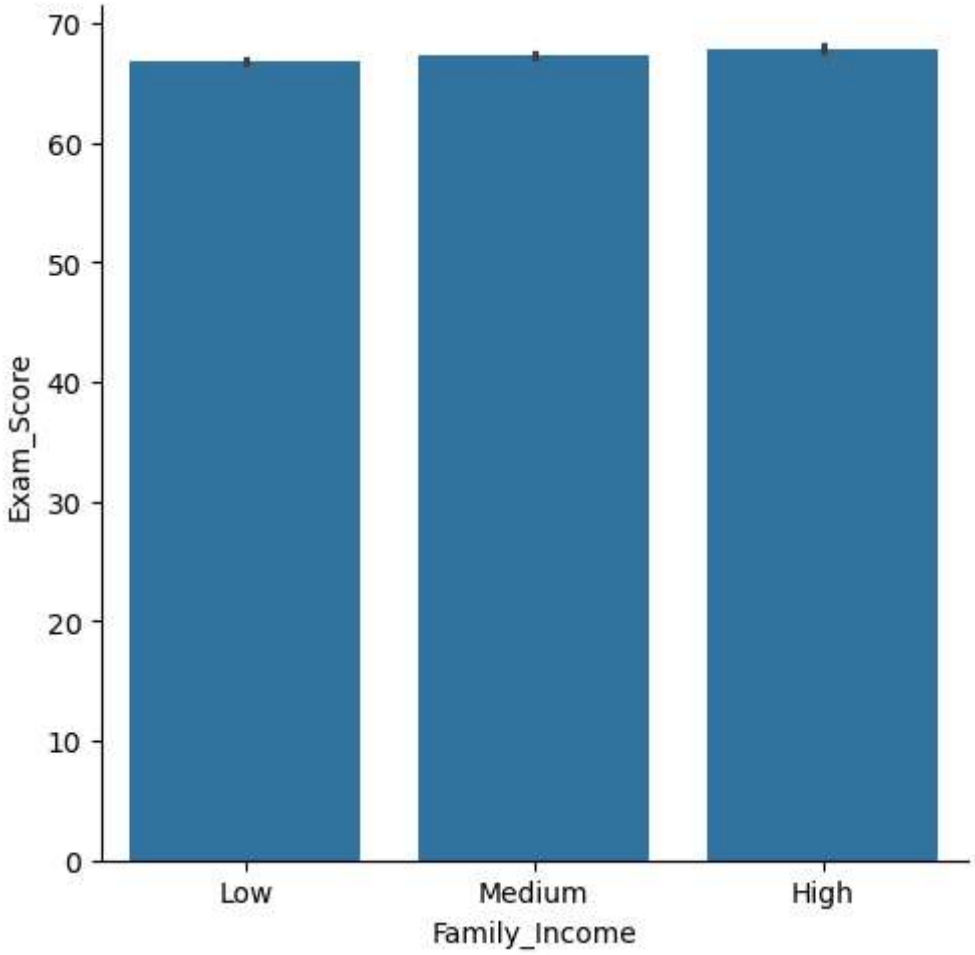
Out[275…
```
Family_Income
High      58
Low       55
Medium    57
Name: Exam_Score, dtype: int64
```

In [277…
```
df.groupby("Family_Income")["Exam_Score"].mean()
```

Out[277…
```
Family_Income
High      67.842396
Low       66.848428
Medium    67.334959
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Family_Income
# so that there is no effect on exam_score  due to the Family_Income.
```

In [279…
```
sns.catplot(x="Family_Income",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [259…
```
df.groupby("Teacher_Quality")["Exam_Score"].max()
```

Out[259…
```
Teacher_Quality
High      101
Low        94
Medium    100
Name: Exam_Score, dtype: int64
```

In [281…
```
df.groupby("Teacher_Quality")["Exam_Score"].min()
```

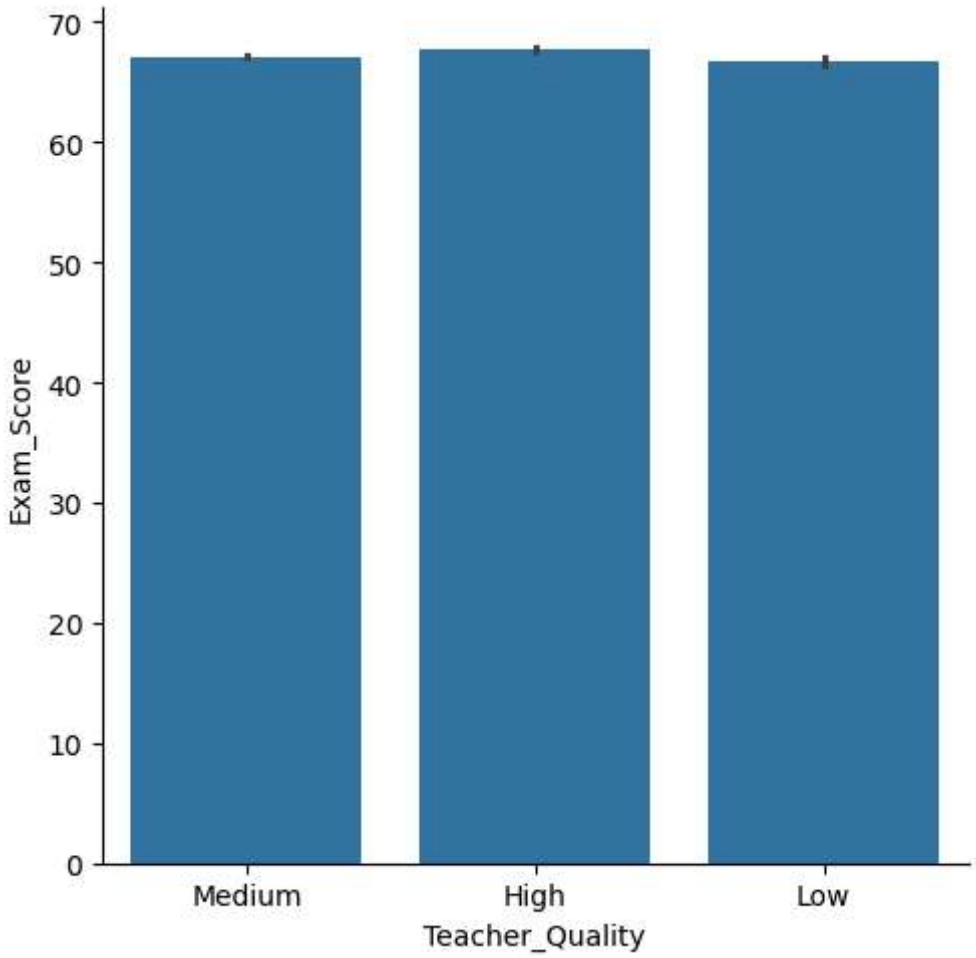
Out[281…
```
Teacher_Quality
High      58
Low       58
Medium    55
Name: Exam_Score, dtype: int64
```

In [285…
```
df.groupby("Teacher_Quality")["Exam_Score"].mean()
```

Out[285…
```
Teacher_Quality
High      67.676939
Low       66.753425
Medium    67.109299
Name: Exam_Score, dtype: float64
```

In [265…
```
# the mean,minmum,maximum exam_score is not effected by Teacher_Quality
# so that there is no effect on exam_score  due to the Teacher_Quality
```

In [287…
```
sns.catplot(x="Teacher_Quality",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [269…
```
df.groupby("School_Type")["Exam_Score"].max()
```

Out[269…
```
School_Type
Private    100
Public     101
Name: Exam_Score, dtype: int64
```

In [289…
```
df.groupby("School_Type")["Exam_Score"].min()
```

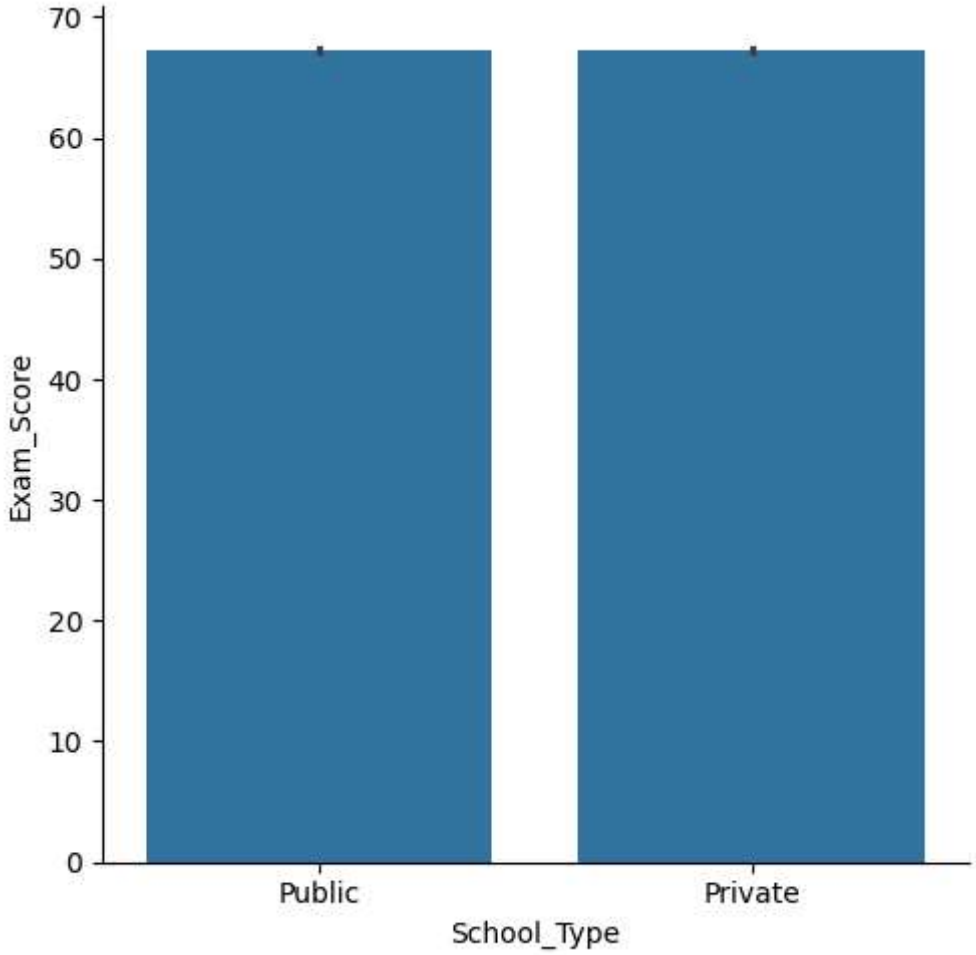
Out[289…
```
School_Type
Private    56
Public     55
Name: Exam_Score, dtype: int64
```

In [291…
```
df.groupby("School_Type")["Exam_Score"].mean()
```

Out[291…
```
School_Type
Private    67.287705
Public     67.212919
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by School_Type
# so that there is no effect on exam_score  due to the School_Type.
```

In [293…
```
sns.catplot(x="School_Type",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [295…
```
df.groupby("Peer_Influence")["Exam_Score"].max()
```

Out[295…
```
Peer_Influence
Negative     99
Neutral      97
Positive    101
Name: Exam_Score, dtype: int64
```

In [297…
```
df.groupby("Peer_Influence")["Exam_Score"].min()
```

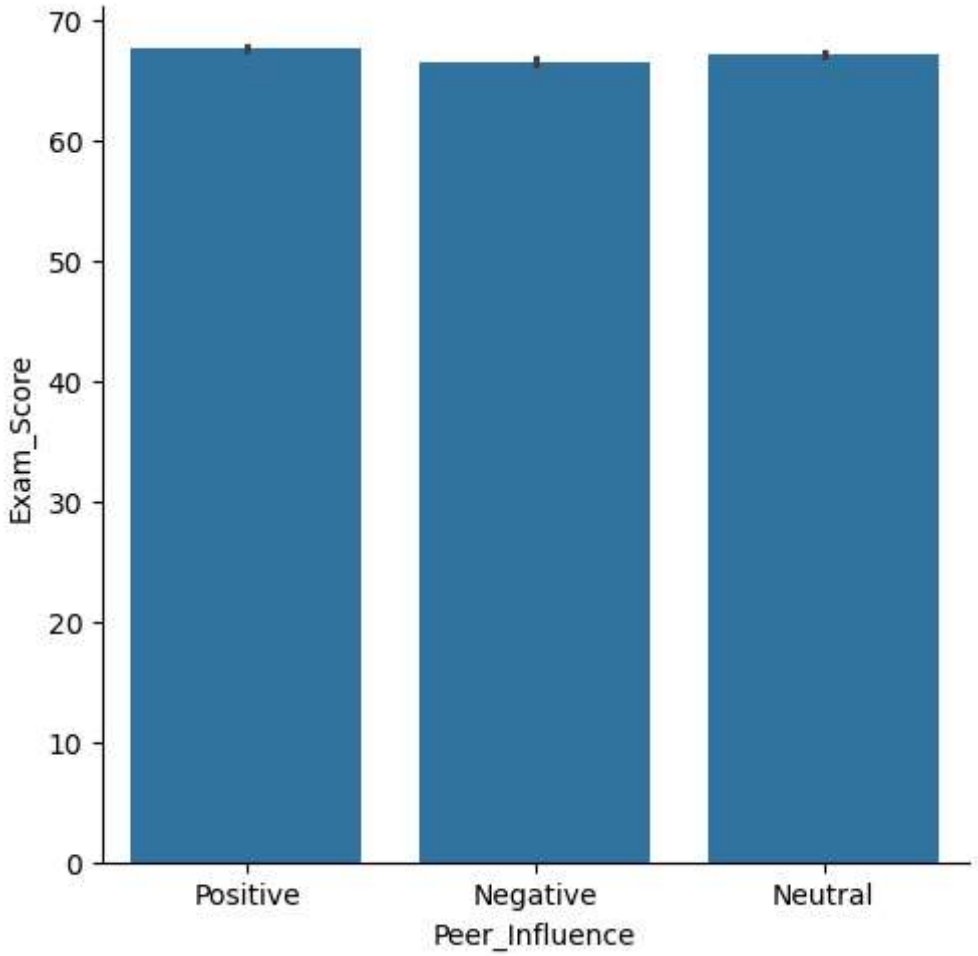
Out[297…
```
Peer_Influence
Negative    55
Neutral     57
Positive    57
Name: Exam_Score, dtype: int64
```

In [299…
```
df.groupby("Peer_Influence")["Exam_Score"].mean()
```

Out[299…
```
Peer_Influence
Negative    66.564270
Neutral     67.197917
Positive    67.623199
Name: Exam_Score, dtype: float64
```

In [301…
```
# the mean,minmum,maximum exam_score is not effected by Peer_Influence
# so that there is no effect on exam_score  due to the Peer_Influence
```

In [303…
```
sns.catplot(x="Peer_Influence",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:

In [305…
```
df.head(2)
```

Out[305…

	Studied_Hours	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_A
0	23	84	Low	High	
1	19	64	Low	Medium	

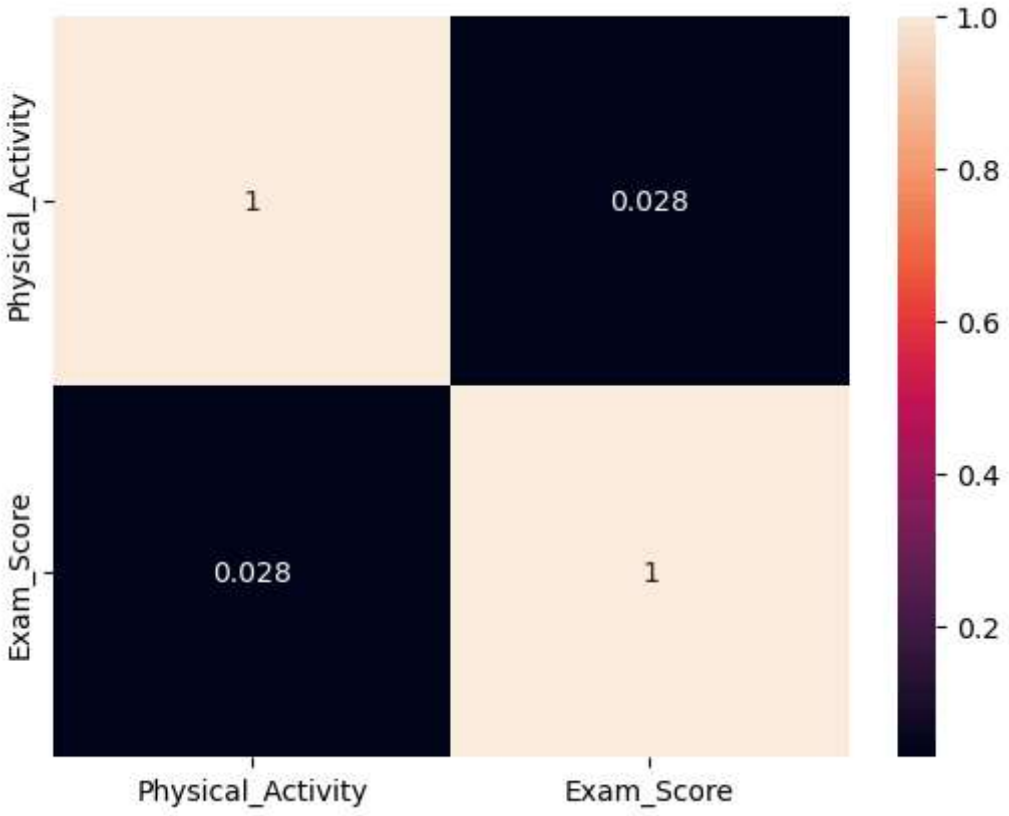
In [309…
```
corr=df[["Physical_Activity","Exam_Score"]].corr()
corr
```

Out[309…

	Physical_Activity	Exam_Score
Physical_Activity	1.000000	0.027824
Exam_Score	0.027824	1.000000

In [311…
```
sns.heatmap(corr,annot=True)
```

Out[311… <Axes: >



In [ ]:
```
# the correlation b/w Physical_Activity and Exam_score is weak correlation
# so that there is few effect on exam_score  due to the Physical_Activity.
```

In [ ]:
```
```

In [313…
```
df.groupby("Learning_Disabilities")["Exam_Score"].max()
```

Out[313…
```
Learning_Disabilities
No     101
Yes     89
Name: Exam_Score, dtype: int64
```

In [315…
```
df.groupby("Learning_Disabilities")["Exam_Score"].min()
```

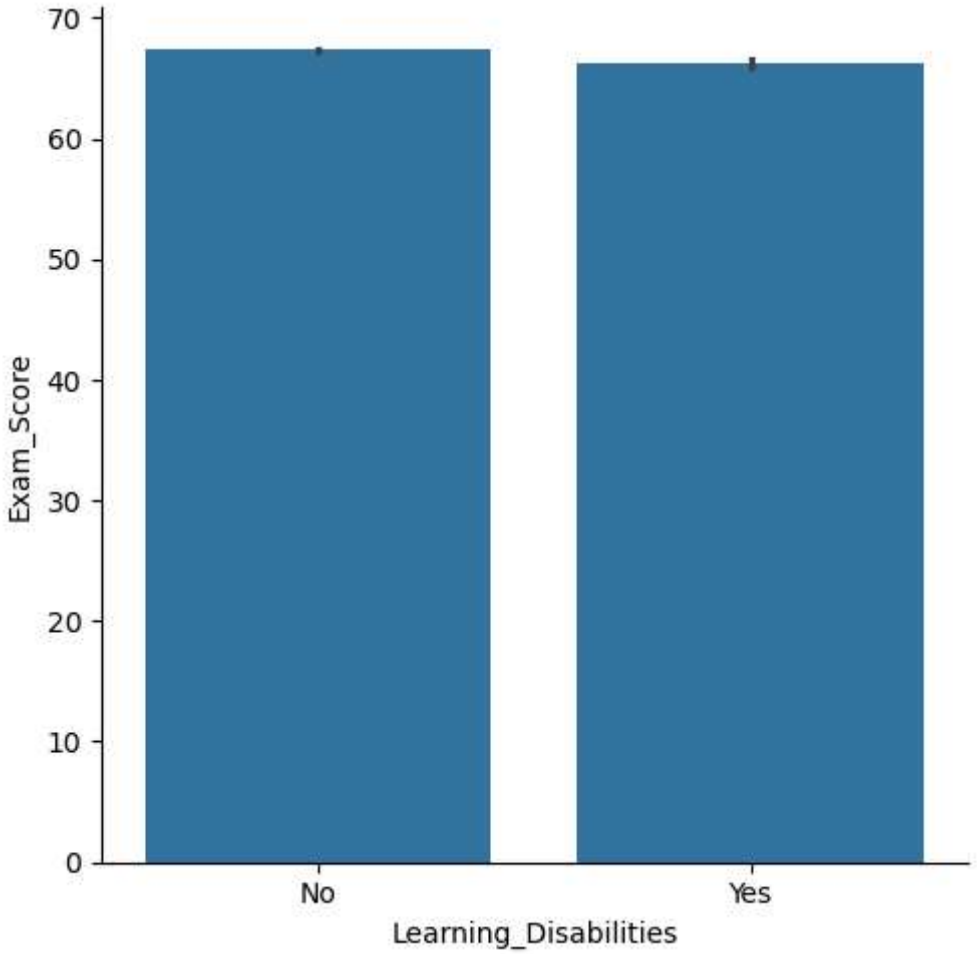
Out[315…
```
Learning_Disabilities
No     55
Yes    57
Name: Exam_Score, dtype: int64
```

In [317…
```
df.groupby("Learning_Disabilities")["Exam_Score"].mean()
```

Out[317…
```
Learning_Disabilities
No     67.349120
Yes    66.270504
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Learning_Disabilities
# so that there is no effect on exam_score  due to the Learning_Disabilities
```

In [319…
```
sns.catplot(x="Learning_Disabilities",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [321…
```
df.groupby("Parental_Education_Level")["Exam_Score"].max()
```

Out[321…
```
Parental_Education_Level
College          100
High School      101
Postgraduate      98
Name: Exam_Score, dtype: int64
```

In [323…
```
df.groupby("Parental_Education_Level")["Exam_Score"].min()
```

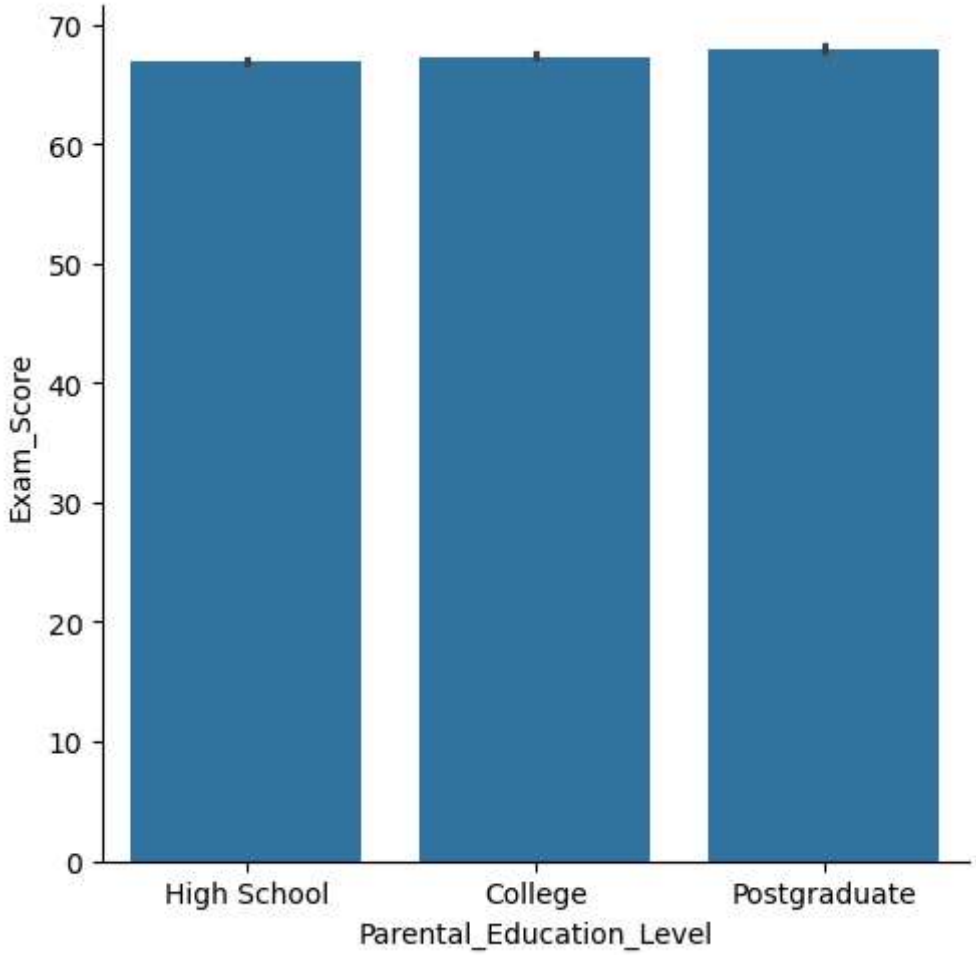
Out[323…
```
Parental_Education_Level
College          56
High School      55
Postgraduate     57
Name: Exam_Score, dtype: int64
```

In [325…
```
df.groupby("Parental_Education_Level")["Exam_Score"].mean()
```

Out[325…
```
Parental_Education_Level
College         67.315737
High School     66.893577
Postgraduate    67.970881
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Parental_Education_Level
# so that there is no effect on exam_score  due to the Parental_Education_Level
```

In [327…
```
sns.catplot(x="Parental_Education_Level",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [329…
```
df.groupby("Distance_from_Home")["Exam_Score"].min()
```

Out[329…
```
Distance_from_Home
Far         56
Moderate    57
Near        55
Name: Exam_Score, dtype: int64
```

In [331…
```
df.groupby("Distance_from_Home")["Exam_Score"].max()
```

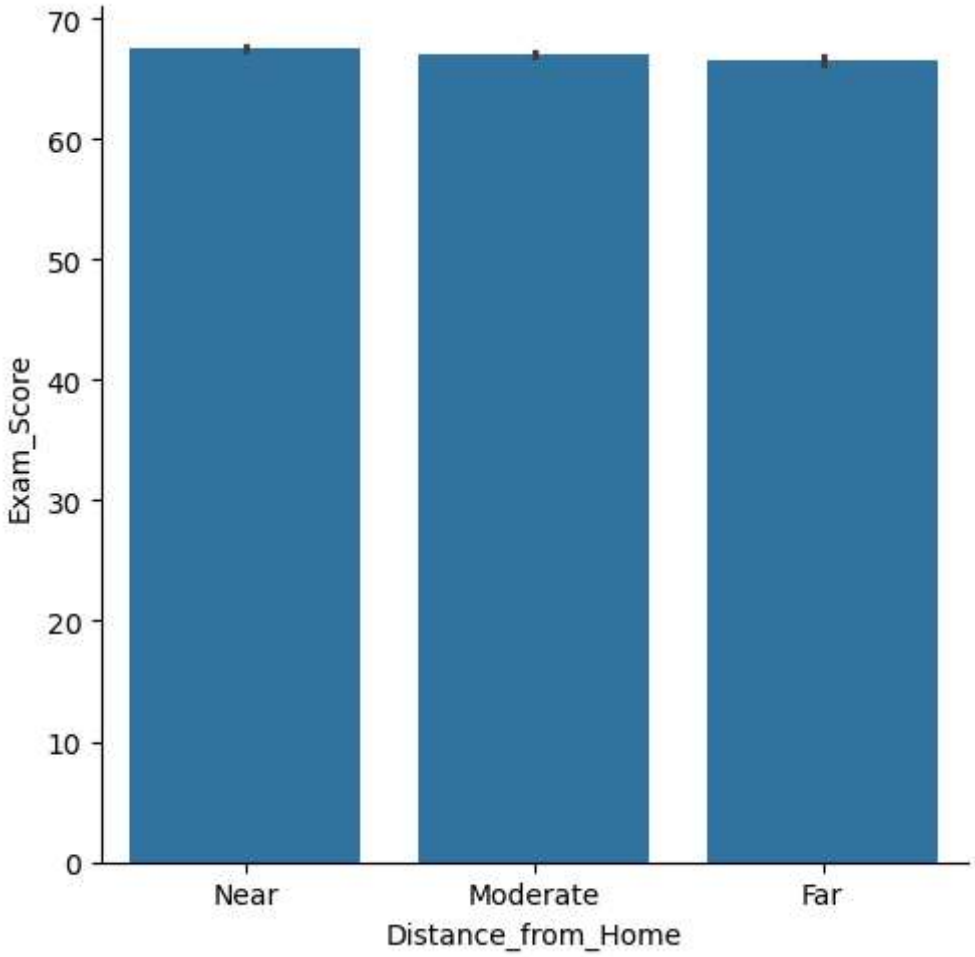
Out[331…
```
Distance_from_Home
Far          99
Moderate    101
Near        100
Name: Exam_Score, dtype: int64
```

In [333…
```
df.groupby("Distance_from_Home")["Exam_Score"].mean()
```

Out[333…
```
Distance_from_Home
Far         66.457447
Moderate    66.981481
Near        67.512101
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by Distance_from_Home
# so that there is no effect on exam_score  due to the Distance_from_Home
```

In [335…
```
sns.catplot(x="Distance_from_Home",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [ ]:
```
```

In [337…
```
df.groupby("Gender")["Exam_Score"].max()
```

Out[337…
```
Gender
Female    101
Male       99
Name: Exam_Score, dtype: int64
```

In [339…
```
df.groupby("Gender")["Exam_Score"].min()
```

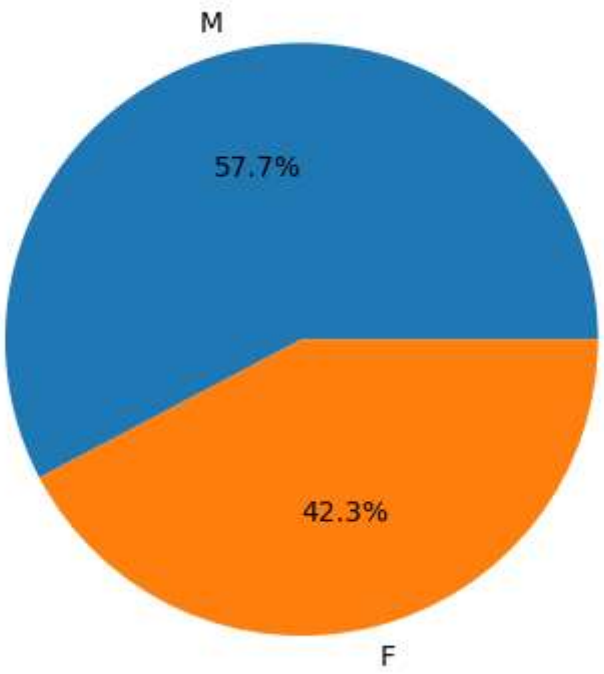
Out[339…
```
Gender
Female    57
Male      55
Name: Exam_Score, dtype: int64
```

In [341…
```
df.groupby("Gender")["Exam_Score"].mean()
```

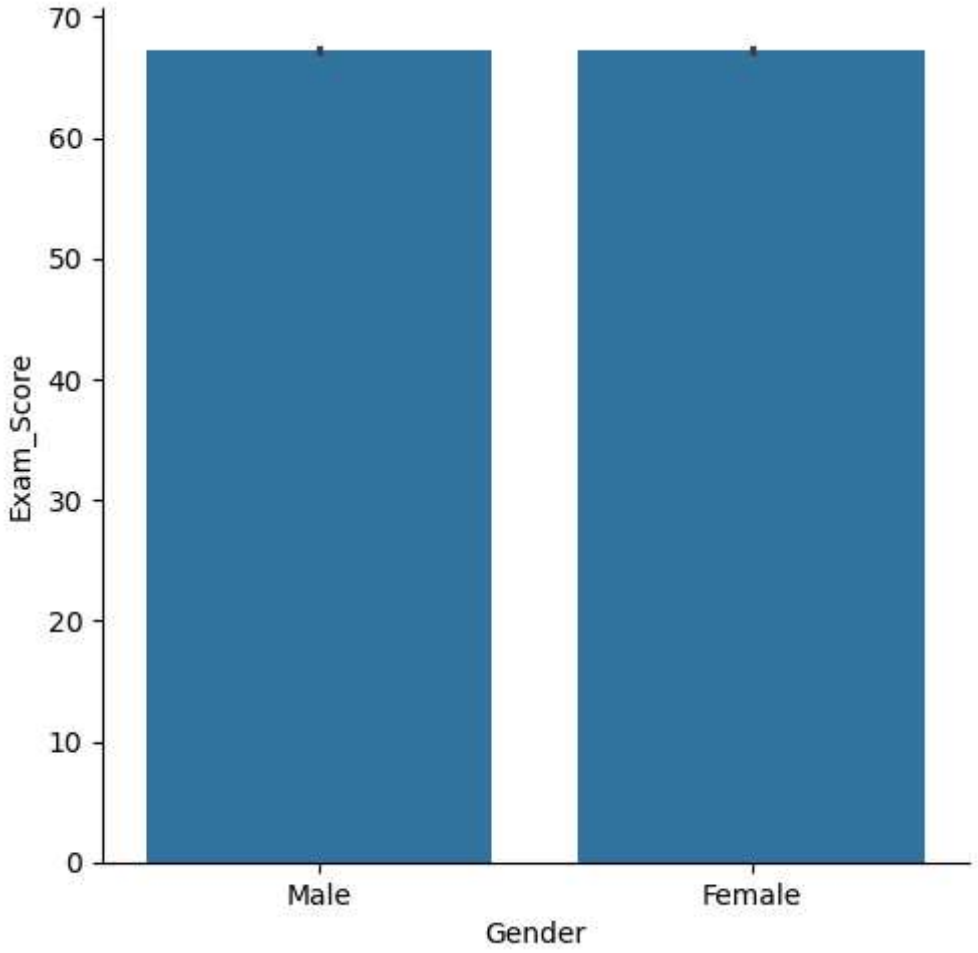
Out[341…
```
Gender
Female    67.244898
Male      67.228894
Name: Exam_Score, dtype: float64
```

In [ ]:
```
# the mean,minmum,maximum exam_score is not effected by gender
# so that there is no effect on exam_score  due to the gender
```

In [361…
```
plt.pie(df["Gender"].value_counts(),labels=["M","F"],autopct="%0.1f%%")
plt.show()
```



In [343…
```
sns.catplot(x="Gender",y="Exam_Score",data=df,kind="bar")
plt.show()
```



In [345…
```
df.head(1)
```

Out[345…

	Studied_Hours	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_A
0	23	84	Low	High	

In [ ]:
```
non_effected_columns=["Parental_Involvement","Access_to_Resources","Extracurricular
                      "Previous_Scores","Motivation_Level","Internet_Access","Family_
                       "Teacher_Quality","School_Type","Peer_Influence","Physical_Act
                       "Learning_Disabilities","Parental_Education_Level","Distance_f
```

In [357…
```
moderate_effected_columns=df[["Studied_Hours","Attendance","Sleep_Hours","Tutoring_
moderate_effected_columns
```

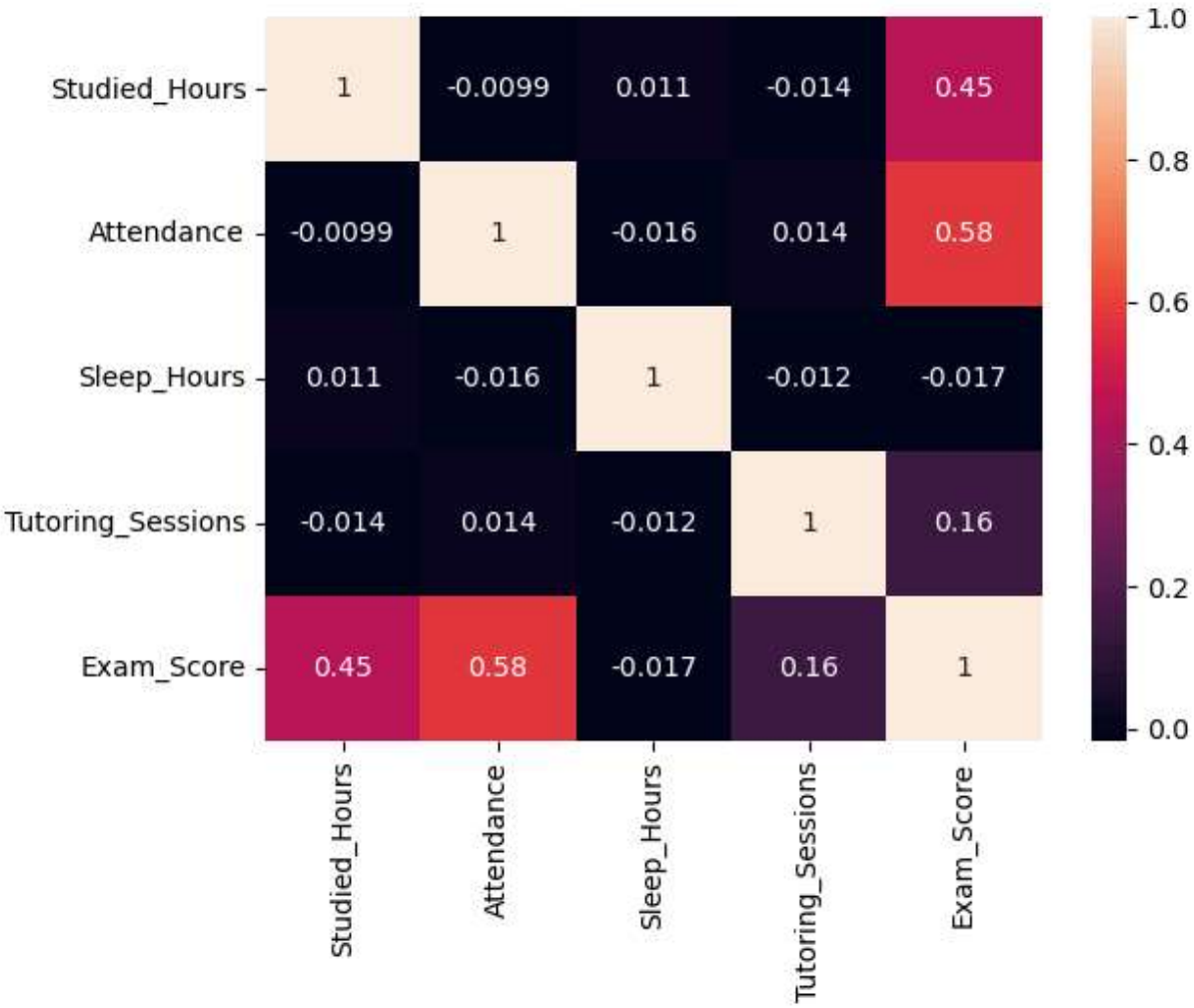
Out[357…

	Studied_Hours	Attendance	Sleep_Hours	Tutoring_Sessions	Exam_Score
Studied_Hours	1.000000	-0.009908	0.010977	-0.014282	0.445455
Attendance	-0.009908	1.000000	-0.015918	0.014324	0.581072
Sleep_Hours	0.010977	-0.015918	1.000000	-0.012216	-0.017022
Tutoring_Sessions	-0.014282	0.014324	-0.012216	1.000000	0.156525
Exam_Score	0.445455	0.581072	-0.017022	0.156525	1.000000

In [359…

```
sns.heatmap(moderate_effected_columns,annot=True)
```

Out[359… <Axes: >



In [ ]:

```
#    Moderately influential factors for better exam scores are:

#           Attendance (most impactful)
#           Studied_Hours

#    Less influential factors:

#            Tutoring_Sessions (minor effect)
```

```
#   No significant effect:

#            Sleep_Hours
```

In [ ]: