```
Start coding or generate with AI.
```

```
import pandas as pd

df = pd.read_csv('/content/Student_Performance.csv')
display(df.head())
```

	student_id	age	gender	school_type	parent_education	study_hours	attendance_percentage	internet_access	travel_time	extr
0	1	14	male	public	post graduate	3.1	84.3	yes	<15 min	
1	2	18	female	public	graduate	3.7	87.8	yes	>60 min	
2	3	17	female	private	post graduate	7.9	65.5	no	<15 min	
3	4	16	other	public	high school	1.1	58.1	no	15-30 min	
4	5	16	female	public	high school	1.3	61.0	yes	30-60 min	

1. Group Scores by Gender

```
gender_scores = df.groupby('gender')[['math_score', 'science_score', 'english_score']].mean()
print('Average scores by gender:')
display(gender_scores)
```

```
Average scores by gender:
```

gender	math_score	science_score	english_score
female	64.045428	63.965862	63.893317
male	63.872754	63.852104	63.661150
other	63.447170	63.425227	63.495167

2. Group Scores by Parental Education

```
parent_education_scores = df.groupby('parent_education')[['math_score', 'science_score', 'english_score']].mean()
print('Average scores by parental education:')
display(parent_education_scores)
```

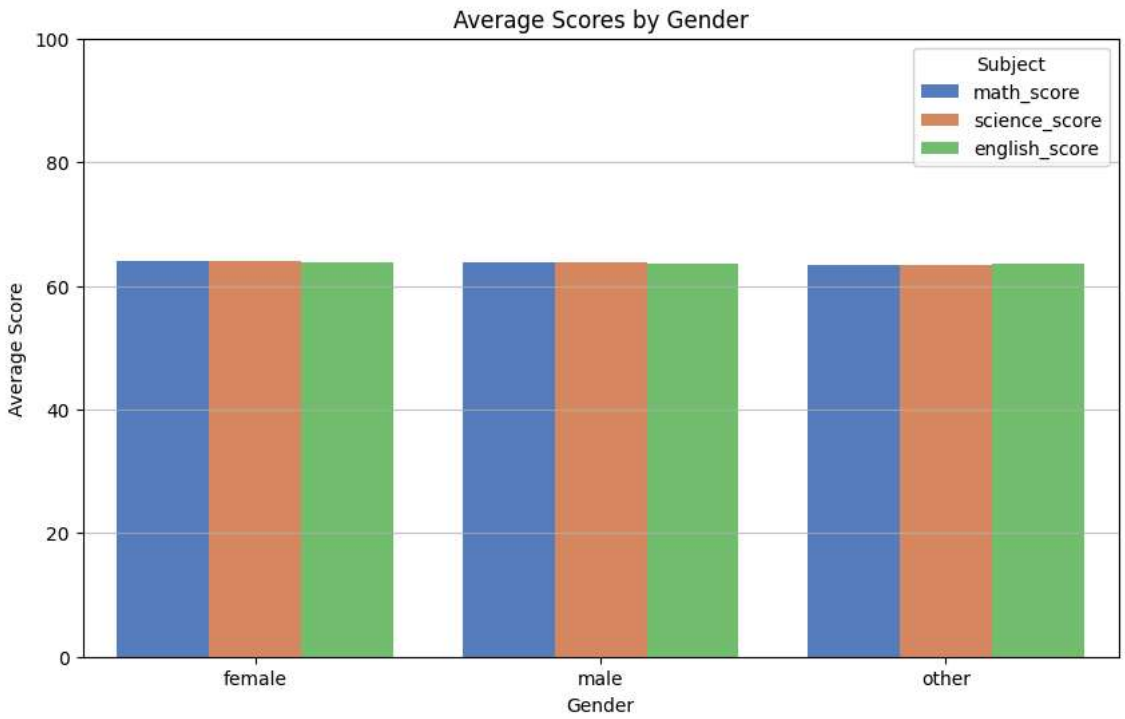
```
Average scores by parental education:
```

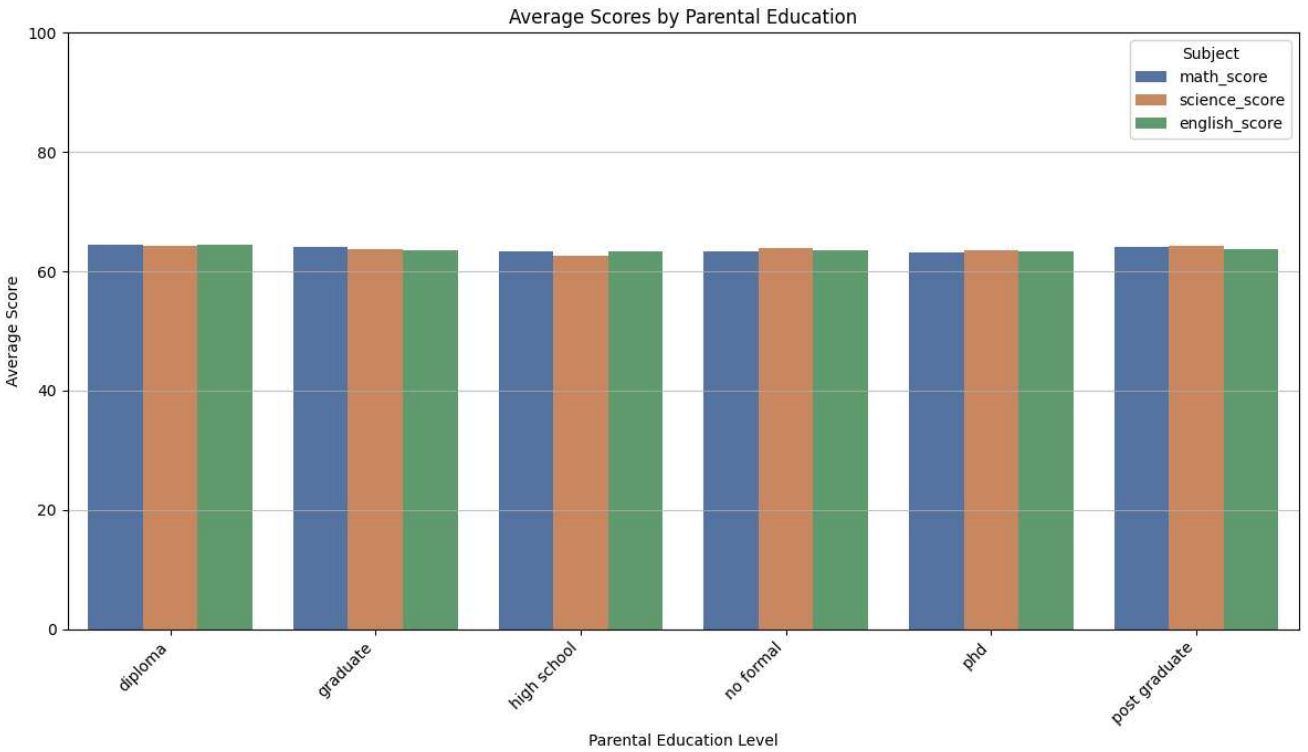
parent_education	math_score	science_score	english_score
diploma	64.546847	64.349374	64.539569
graduate	64.040974	63.794766	63.468839
high school	63.413127	62.622568	63.358502
no formal	63.384531	63.930326	63.594680
phd	63.158813	63.443859	63.304633
post graduate	64.126287	64.314013	63.785582

3. Visualize Group-wise Average Scores using Bar Charts

```
import matplotlib.pyplot as plt
import seaborn as sns

# Bar chart for Average Scores by Gender
gender_scores_melted = gender_scores.reset_index().melt('gender', var_name='Subject', value_name='Average Score')

plt.figure(figsize=(10, 6))
sns.barplot(x='gender', y='Average Score', hue='Subject', data=gender_scores_melted, palette='muted')
plt.title('Average Scores by Gender')
plt.xlabel('Gender')
```

```
plt.ylabel('Average Score')
plt.ylim(0, 100)
plt.grid(axis='y', alpha=0.75)
plt.show()

# Bar chart for Average Scores by Parental Education
parent_education_scores_melted = parent_education_scores.reset_index().melt('parent_education', var_name='Subject', value_name=

plt.figure(figsize=(12, 7))
sns.barplot(x='parent_education', y='Average Score', hue='Subject', data=parent_education_scores_melted, palette='deep')
plt.title('Average Scores by Parental Education')
plt.xlabel('Parental Education Level')
plt.ylabel('Average Score')
plt.ylim(0, 100)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', alpha=0.75)
plt.tight_layout()
plt.show()
```

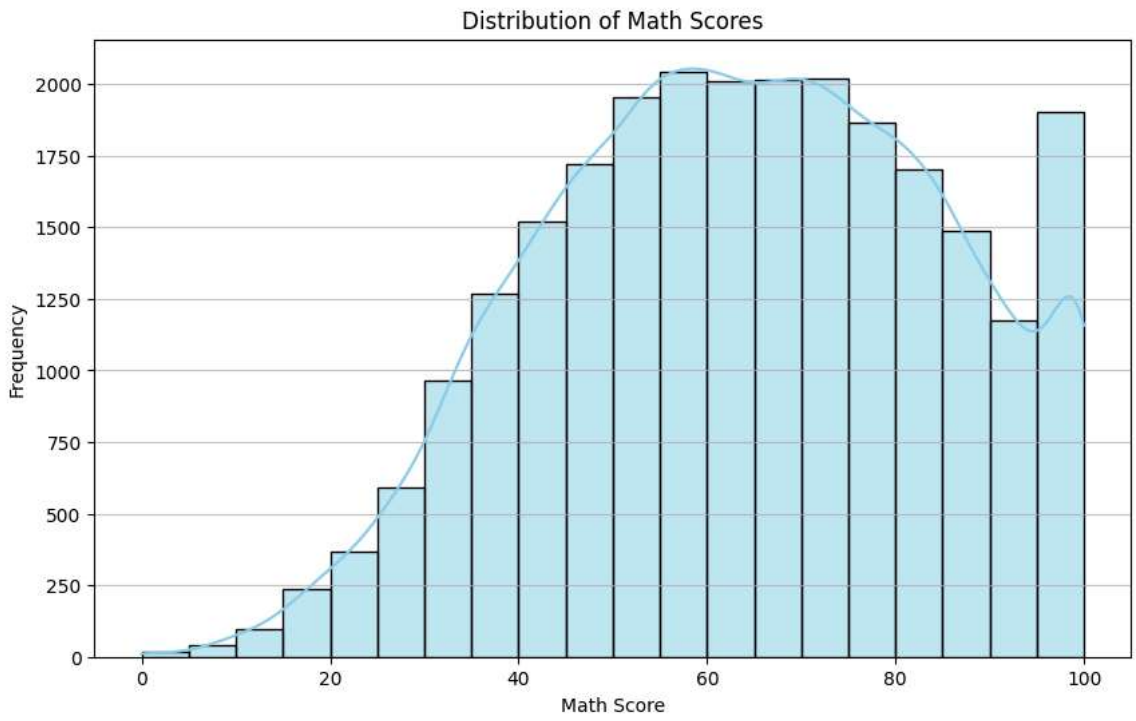




1. Plot Histogram of Math Scores

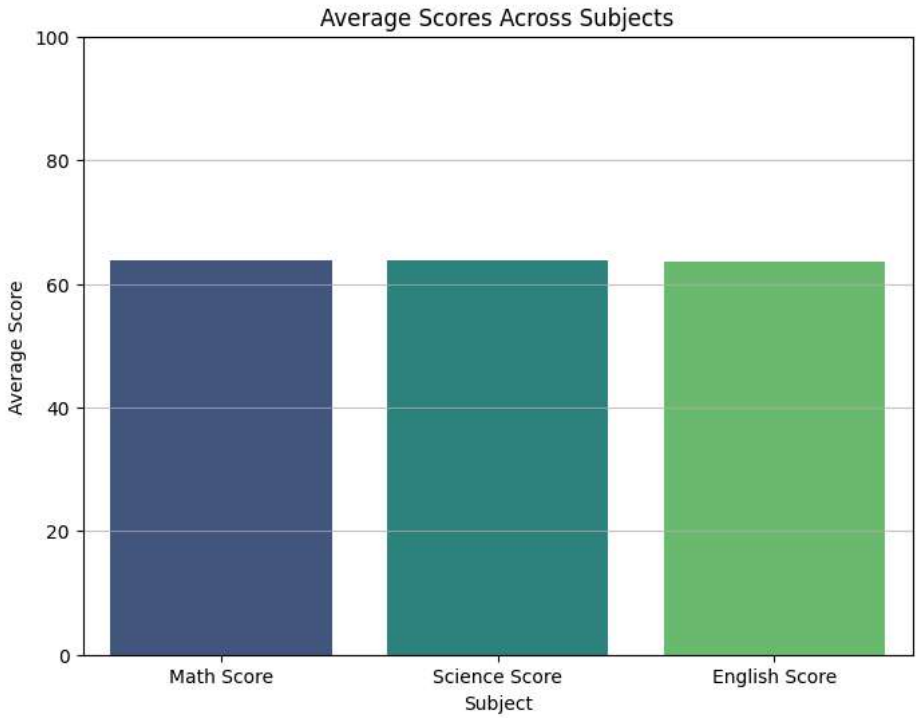
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 6))
sns.histplot(df['math_score'], bins=20, kde=True, color='skyblue')
plt.title('Distribution of Math Scores')
```

```
plt.xlabel('Math Score')
plt.ylabel('Frequency')
plt.grid(axis='y', alpha=0.75)
plt.show()
```



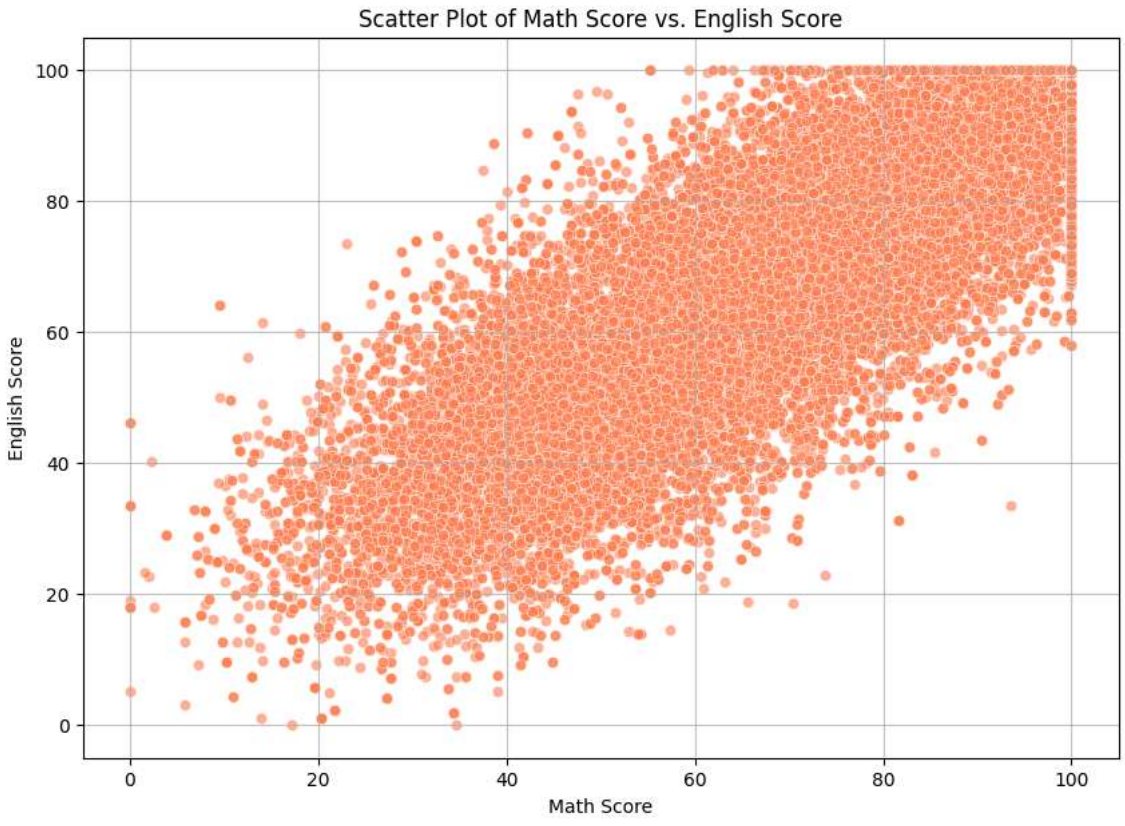
2. Plot Bar Chart of Average Scores

```
score_types = ['math_score', 'science_score', 'english_score']
average_scores = df[score_types].mean()

plt.figure(figsize=(8, 6))
sns.barplot(x=average_scores.index, y=average_scores.values, hue=average_scores.index, palette='viridis', legend=False)
plt.title('Average Scores Across Subjects')
plt.xlabel('Subject')
plt.ylabel('Average Score')
plt.xticks(ticks=range(len(score_types)), labels=[col.replace('_', ' ').title() for col in score_types])
plt.ylim(0, 100) # Scores are out of 100
plt.grid(axis='y', alpha=0.75)
plt.show()
```



3. Create Scatter Plot Between Math and English Scores

```
plt.figure(figsize=(10, 7))
sns.scatterplot(x=df['math_score'], y=df['english_score'], alpha=0.6, color='coral')
plt.title('Scatter Plot of Math Score vs. English Score')
plt.xlabel('Math Score')
plt.ylabel('English Score')
plt.grid(alpha=0.75)
plt.show()
```



1. Invert Score Columns into NumPy Arrays and Compute Statistics

```
import numpy as np

score_columns = ['math_score', 'science_score', 'english_score', 'overall_score']

print('Statistics for Score Columns (as NumPy arrays):\n')

for col in score_columns:
    # Convert column to NumPy array
    score_array = df[col].to_numpy()

    print(f'--- {col.replace("_", " ").title()} ---\n')
    print(f'  Mean: {np.mean(score_array):.2f}')
    print(f'  Median: {np.median(score_array):.2f}')
    print(f'  Standard Deviation: {np.std(score_array):.2f}')
    print(f'  Minimum: {np.min(score_array):.2f}')
    print(f'  Maximum: {np.max(score_array):.2f}')
    print('\n')
```

```
Statistics for Score Columns (as NumPy arrays):

--- Math Score ---

  Mean: 63.79
  Median: 64.10
  Standard Deviation: 20.87
  Minimum: 0.00
  Maximum: 100.00


--- Science Score ---

  Mean: 63.75
  Median: 64.10
  Standard Deviation: 20.97
  Minimum: 0.00
  Maximum: 100.00


--- English Score ---

  Mean: 63.68
  Median: 64.20
  Standard Deviation: 20.79
  Minimum: 0.00
  Maximum: 100.00


--- Overall Score ---

  Mean: 64.01
  Median: 64.20
  Standard Deviation: 18.93
  Minimum: 14.50
  Maximum: 100.00
```

1. Select Score-Related Columns

```
score_columns = ['math_score', 'science_score', 'english_score', 'overall_score', 'final_grade']
score_df = df[score_columns]
print('Score-related columns:')
display(score_df.head())
```

```
Score-related columns:
```

	math_score	science_score	english_score	overall_score	final_grade
0	42.7	55.4	57.0	53.1	e
1	57.6	68.8	64.8	61.3	d
2	84.8	95.0	79.2	89.6	b
3	44.4	27.5	54.7	41.6	e
4	8.9	32.7	30.0	25.4	f

2. Filter Students Scoring Above 70 in Math

```
high_math_scores_df = df[df['math_score'] > 70]
print('Students with math scores above 70:')
display(high_math_scores_df.head())
```

```
Students with math scores above 70:
```

	student_id	age	gender	school_type	parent_education	study_hours	attendance_percentage	internet_access	travel_time	ext
2	3	17	female	private	post graduate	7.9	65.5	no	<15 min	
9	10	14	female	public	diploma	6.8	62.4	yes	>60 min	
10	11	17	female	private	graduate	6.1	90.5	yes	15-30 min	
12	13	18	female	private	high school	6.8	58.2	yes	>60 min	
14	15	18	other	public	high school	4.9	85.3	yes	<15 min	

3. Filter Data Based on Gender (e.g., 'female')

```
female_students_df = df[df['gender'] == 'female']
print('Data for female students:')
display(female_students_df.head())
```

```
Data for female students:
```

	student_id	age	gender	school_type	parent_education	study_hours	attendance_percentage	internet_access	travel_time	extr
1	2	18	female	public	graduate	3.7	87.8	yes	>60 min	
2	3	17	female	private	post graduate	7.9	65.5	no	<15 min	
4	5	16	female	public	high school	1.3	61.0	yes	30-60 min	
6	7	14	female	private	post graduate	1.8	81.6	yes	30-60 min	
7	8	18	female	private	post graduate	5.6	59.4	yes	>60 min	

4. Count Number of Students in Each Category (e.g., gender, school_type, final_grade)

```
print('Count of students by gender:')
display(df['gender'].value_counts())

print('\nCount of students by school type:')
display(df['school_type'].value_counts())

print('\nCount of students by final grade:')
display(df['final_grade'].value_counts())
```

```
Count of students by gender:
```

gender	count
other	8463
female	8290
male	8247

dtype: int64

```
Count of students by school type:
```

school_type	count
private	12725
public	12275

Displaying First and Last Five Rows

```
print('First 5 rows:')
display(df.head())

print('\nLast 5 rows:')
display(df.tail())
```

```
First 5 rows:
```

	student_id	age	gender	school_type	parent_education	study_hours	attendance_percentage	internet_access	travel_time	extr
0	1	14	male	public	post graduate	3.1	84.3	yes	<15 min	
1	2	18	female	public	graduate	3.7	87.8	yes	>60 min	
2	3	17	female	private	post graduate	7.9	65.5	no	<15 min	
3	4	16	other	public	high school	1.1	58.1	no	15-30 min	
4	5	16	female	public	high school	1.3	61.0	yes	30-60 min	

dtype: int64

```
Last 5 rows:
```

	student_id	age	gender	school_type	parent_education	study_hours	attendance_percentage	internet_access	travel_time
24995	12047	17	female	public	phd	1.8	55.2	yes	15-30 min
24996	1102	16	female	private	diploma	2.7	97.1	yes	<15 min
24997	4422	19	other	private	post graduate	1.0	63.0	yes	<15 min
24998	7858	14	male	private	diploma	1.0	69.4	yes	15-30 min
24999	11621	18	other	public	no formal	0.7	60.3	yes	30-60 min

Finding Shape, Columns, and Data Types