Exploratory Data Analysis – Student Performance

1. Introduction

The dataset used in this analysis is the Student Performance dataset.
It contains information about students' scores in math, reading, and writing, along with demographic features like gender, parental level of education, lunch type, race/ethnicity, and test preparation course status.

The purpose of this EDA is to explore patterns and relationships between these features and academic performance.

---

2. Analysis Techniques

The following EDA techniques were applied:

- Displaying top records and data types
- Checking for null and duplicate values
- Summary statistics using describe()
- Countplots for categorical features
- Histograms and boxplots for numeric scores
- Correlation heatmap and pairplots
- Grouped analysis by gender and race/ethnicity

---

3. Key Findings

- Female students perform better in reading and writing
- Male students slightly outperform in math
- Reading and writing scores show strong positive correlation
- Test preparation course improves overall scores
- Group E students have the highest average scores
- Students with standard lunch generally score higher

---

4. Visualizations Used

- Countplots: Gender, parental education, lunch, test prep course
- Histograms and Boxplots: Math, Reading, Writing Scores
- Correlation Heatmap: Relationships between scores
- Pairplot: Multivariate relationships

- Bar Charts: Grouped averages by gender and race/ethnicity

---

5. Conclusion

The Student Performance dataset reveals trends and dependencies between demographic and academic performance factors.
Insights from this EDA can help educators identify areas of support for different student groups.
It demonstrates how visualizations and statistics together provide meaningful understanding of student outcomes.

```
[5] import pandas as pd

    # Load CSV into a DataFrame
    df = pd.read_csv('StudentsPerformance.csv')

    # Show top 5 rows
    df.head()
```

	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

```
[7] # Dataset Shape
    print("Dataset Shape:", df.shape)

    # Dataset Info
    print("\nDataset Info:")
    df.info()

    # Summary Statistics
    print("\nSummary Statistics:")
    display(df.describe())

    # Check for missing values
    print("\nMissing Values:")
    print(df.isnull().sum())
```

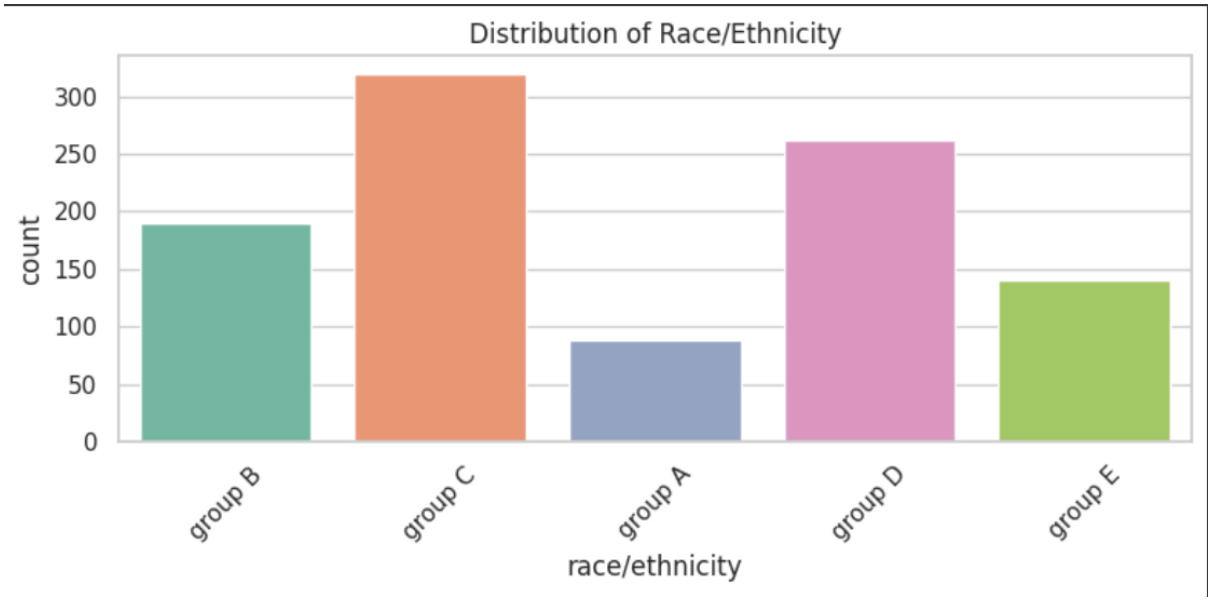
```
Dataset Shape: (1000, 8)

Dataset 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
```

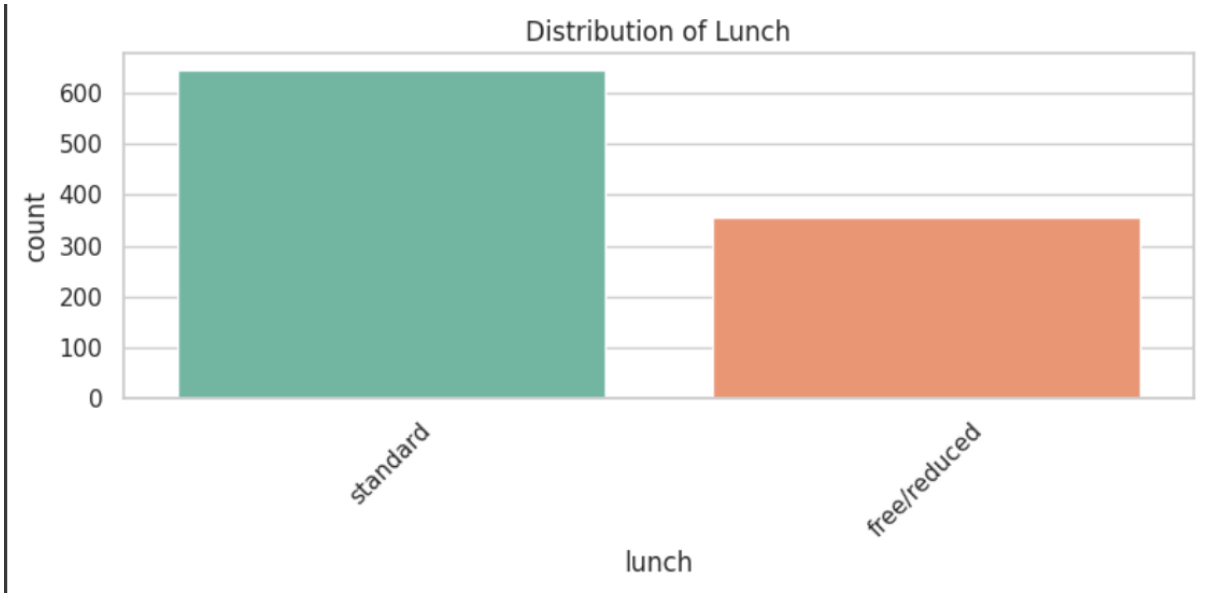
Summary Statistics:

	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

Missing Values:

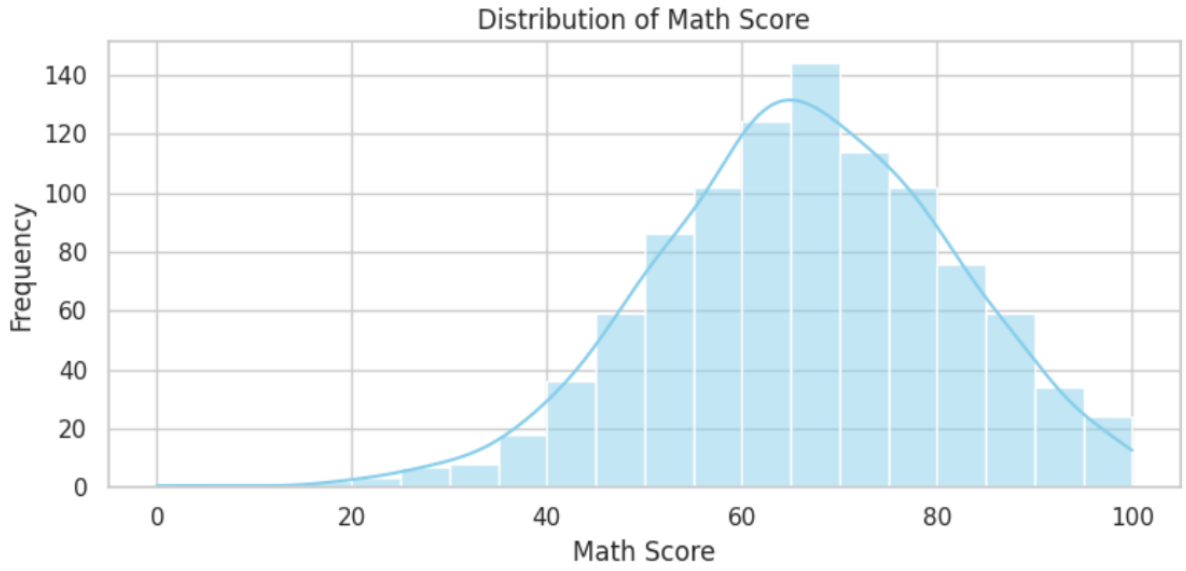
```
gender                       0
race/ethnicity               0
parental_level_of_education  0
lunch                        0
test_preparation_course      0
math_score                   0
reading_score                0
writing_score                0
dtype: int64
```


Distribution of Race/Ethnicity
count
300
250
200
150
100
50
0
group B
group C
group A
group D
group E
race/ethnicity

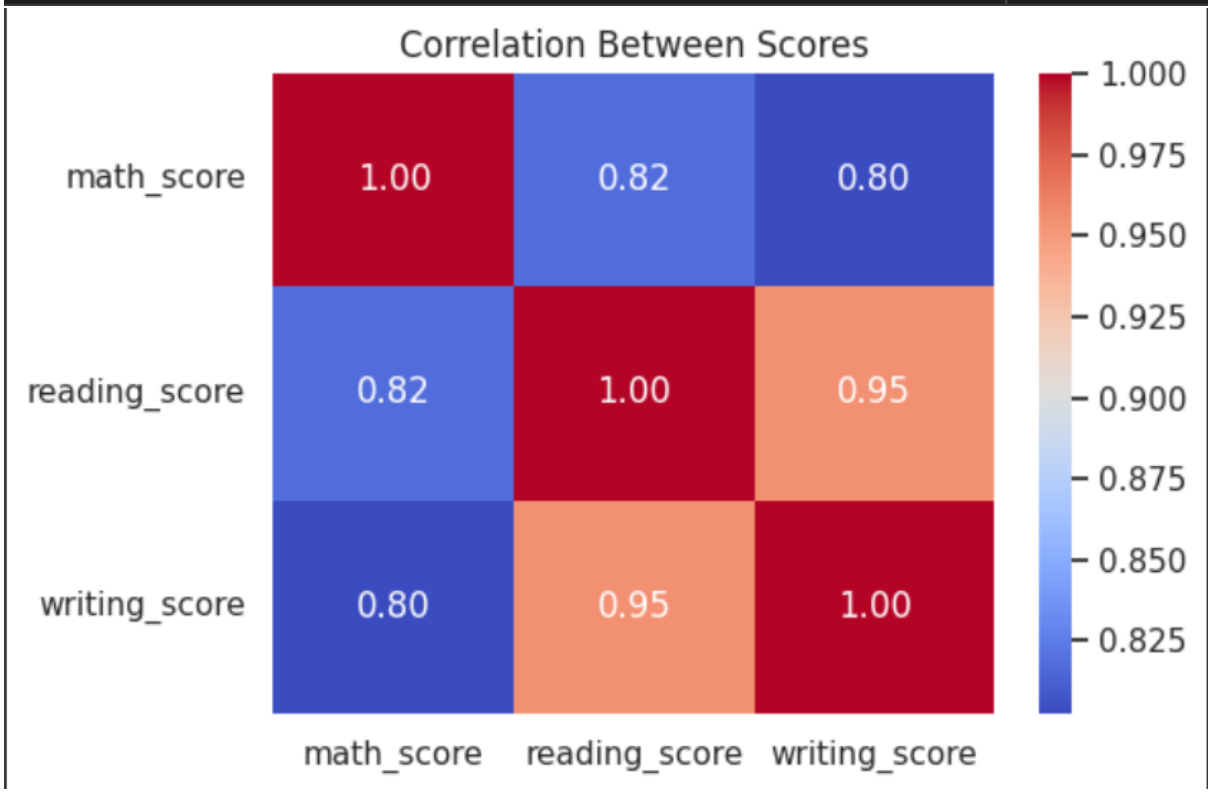

Distribution of Lunch
count
600
500
400
300
200
100
0
standard
free/reduced
lunch

```
# Display basic statistics for numerical columns
df.describe()[['math_score', 'reading_score', 'writing_score']]
```

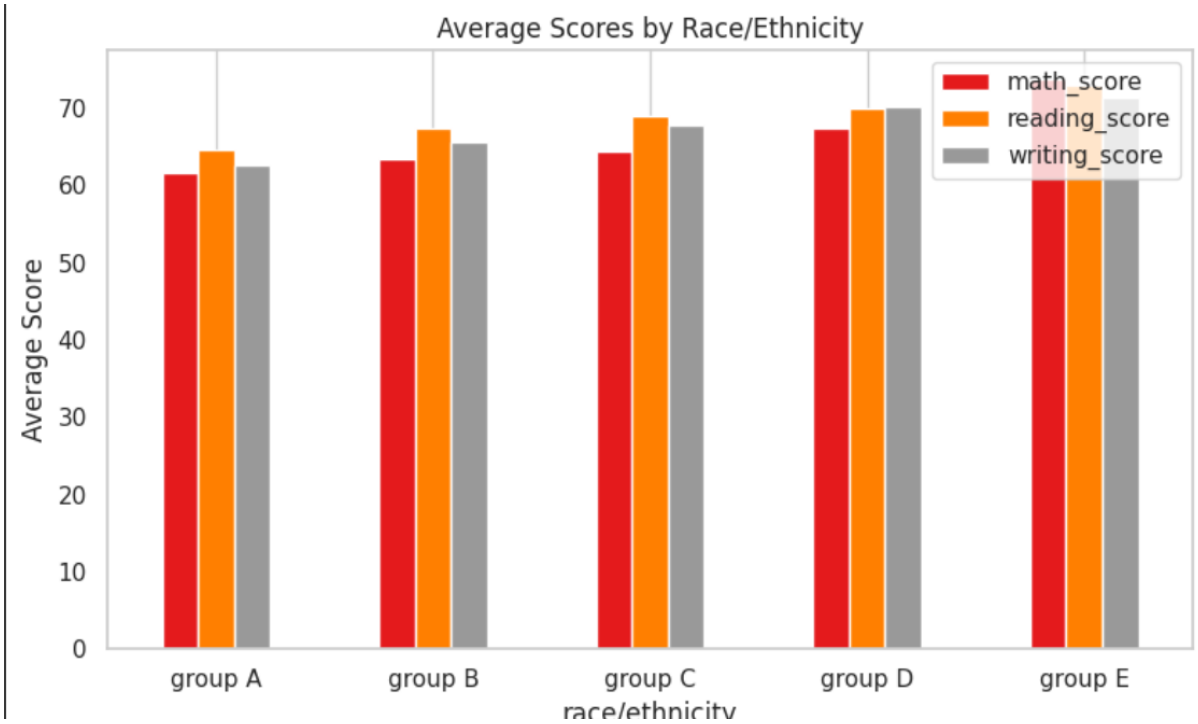
	math_score	reading_score	writing_score
count	1000.00000	1000.000000	1000.000000
mean	66.08900	69.169000	68.054000
std	15.16308	14.600192	15.195657
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	57.750000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000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



```
# Correlation matrix for numerical columns
correlation_matrix = df[['math_score', 'reading_score', 'writing_score']].corr()

# Plot heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Between Scores")
plt.tight_layout()
plt.show()
```




```
# Group by race/ethnicity and calculate mean scores
race_group = df.groupby('race/ethnicity')[['math_score', 'reading_score', 'writing_score']].mean().reset_index()

# Plot
race_group.plot(x='race/ethnicity', kind='bar', figsize=(8,5), colormap='Set1')
plt.title("Average Scores by Race/Ethnicity")
plt.ylabel("Average Score")
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.tight_layout()
plt.show()
```


Average Scores by Race/Ethnicity
math_score
reading_score
writing_score
Average Score
0
10
20
30
40
50
60
70
group A
group B
group C
group D
group E
race/ethnicity