**✅ Cell 1**

python

CopyEdit

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

* **Problem Step**: Import all required Python libraries.
* **Definition**:
  + pandas as pd: For data loading, cleaning, and manipulation.
  + seaborn as sns: For advanced statistical visualizations.
  + matplotlib.pyplot as plt: For creating basic plots and customizing visuals.
* **Why**: These libraries are essential for loading the dataset and visualizing it to explore trends and patterns.

**✅ Cell 2**

python

CopyEdit

df = pd.read\_csv("StudentsPerformance.csv")

df.head()

* **Problem Step**: Load dataset and preview the first few rows.
* **Definition**:
  + pd.read\_csv(): Reads the dataset from a CSV file.
  + df.head(): Displays the first five rows of the DataFrame.
* **Why**: Allows quick inspection of the dataset structure and initial data.

**✅ Cell 3**

python

CopyEdit

df.info()

* **Problem Step**: Get structural information about the dataset.
* **Definition**:
  + .info(): Displays column names, non-null values, and data types.
* **Why**: To check for missing values and ensure correct data types.

**✅ Cell 4**

python

CopyEdit

df.describe()

* **Problem Step**: Get statistical summary of numerical data.
* **Definition**:
  + .describe(): Provides count, mean, std deviation, min, max, and percentiles.
* **Why**: Offers a quick statistical overview of students’ scores.

**✅ Cell 5**

python

CopyEdit

df.isnull().sum()

* **Problem Step**: Check for missing values in each column.
* **Definition**:
  + .isnull(): Identifies missing values.
  + .sum(): Aggregates total missing entries per column.
* **Why**: Ensures data completeness before analysis.

**✅ Cell 6**

python

CopyEdit

df.duplicated().sum()

* **Problem Step**: Check for duplicate rows.
* **Definition**:
  + .duplicated(): Flags rows already present earlier in the dataset.
  + .sum(): Counts the number of such rows.
* **Why**: To maintain data quality by identifying and possibly removing duplicates.

**✅ Cell 7**

python

CopyEdit

df['total score'] = df['math score'] + df['reading score'] + df['writing score']

df['average score'] = df['total score']/3

df.head()

* **Problem Step**: Add new columns for total and average scores.
* **Definition**:
  + Adds up individual subject scores to get total, then divides by 3 for the average.
* **Why**: These aggregate metrics allow easier comparisons of student performance.

**✅ Cell 8**

python

CopyEdit

sns.histplot(df['average score'], kde=True, bins=20)

plt.title("Distribution of Average Scores")

plt.xlabel("Average Score")

plt.ylabel("Frequency")

plt.show()

* **Problem Step**: Visualize the distribution of average scores.
* **Definition**:
  + sns.histplot(): Creates a histogram with optional KDE (kernel density estimate).
  + kde=True: Adds a smooth line showing distribution curve.
  + bins=20: Splits the data into 20 intervals.
* **Why**: To understand how student performance is spread — whether it's normal, skewed, etc.

**✅ Cell 9**

python

CopyEdit

sns.boxplot(x='gender', y='average score', data=df)

plt.title("Average Score Distribution by Gender")

plt.show()

* **Problem Step**: Compare average scores across genders.
* **Definition**:
  + sns.boxplot(): Shows distribution, median, and outliers for each gender group.
* **Why**: Reveals any performance differences between male and female students.

**✅ Cell 10**

python

CopyEdit

sns.violinplot(x='race/ethnicity', y='average score', data=df)

plt.title("Average Score Distribution by Race/Ethnicity")

plt.xticks(rotation=45)

plt.show()

* **Problem Step**: Examine score distributions across ethnic groups.
* **Definition**:
  + sns.violinplot(): Combines boxplot with a KDE to show distribution shape and spread.
* **Why**: Highlights differences and variability in performance between racial/ethnic groups.

**✅ Cell 11**

python

CopyEdit

plt.figure(figsize=(10,6))

sns.barplot(x='parental level of education', y='average score', data=df)

plt.title("Average Score by Parental Level of Education")

plt.xticks(rotation=45)

plt.show()

* **Problem Step**: Analyze influence of parental education on student scores.
* **Definition**:
  + sns.barplot(): Calculates and displays average scores for each education level.
  + plt.figure(figsize=(10,6)): Resizes the figure for readability.
* **Why**: Educational background of parents may impact student outcomes.

**✅ Cell 12**

python

CopyEdit

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

plt.title("Correlation Matrix of Scores")

plt.show()

* **Problem Step**: Explore correlations between numerical variables.
* **Definition**:
  + df.corr(): Returns correlation coefficients between numeric columns.
  + sns.heatmap(): Visualizes the correlations as a color-coded grid.
  + annot=True: Adds numeric correlation values to the cells.
* **Why**: Identifies which scores are most related (e.g., reading vs. writing).