Task

Analyze the dataset provided in the `/content/ODI_Match_Data.csv.zip` file using sorting and filtering.

Extract data

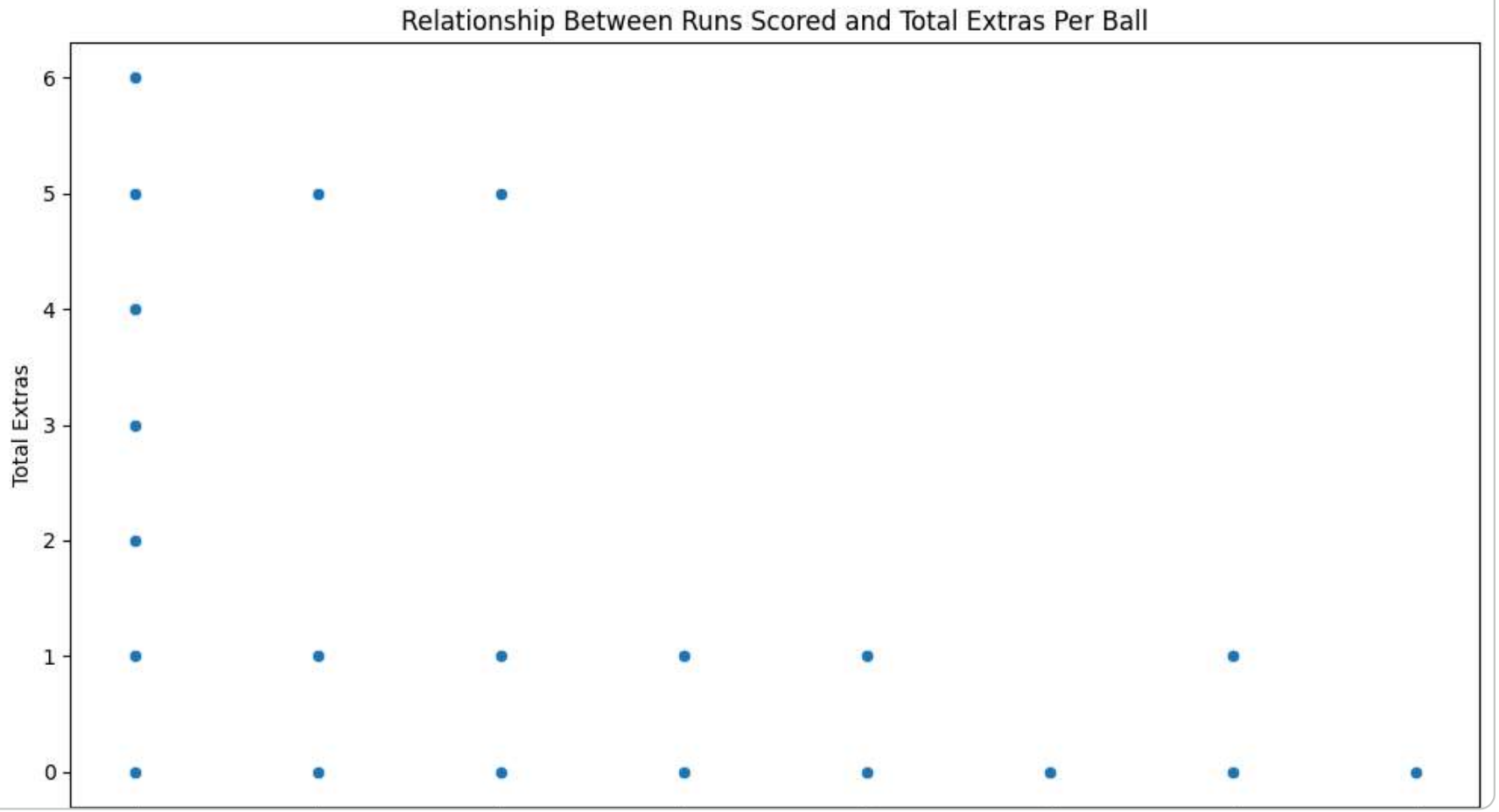
Subtask:

Unzip the `/content/ODI_Match_Data.csv.zip` file to access the CSV data.

Reasoning: Unzip the provided zip file to access the data for analysis.

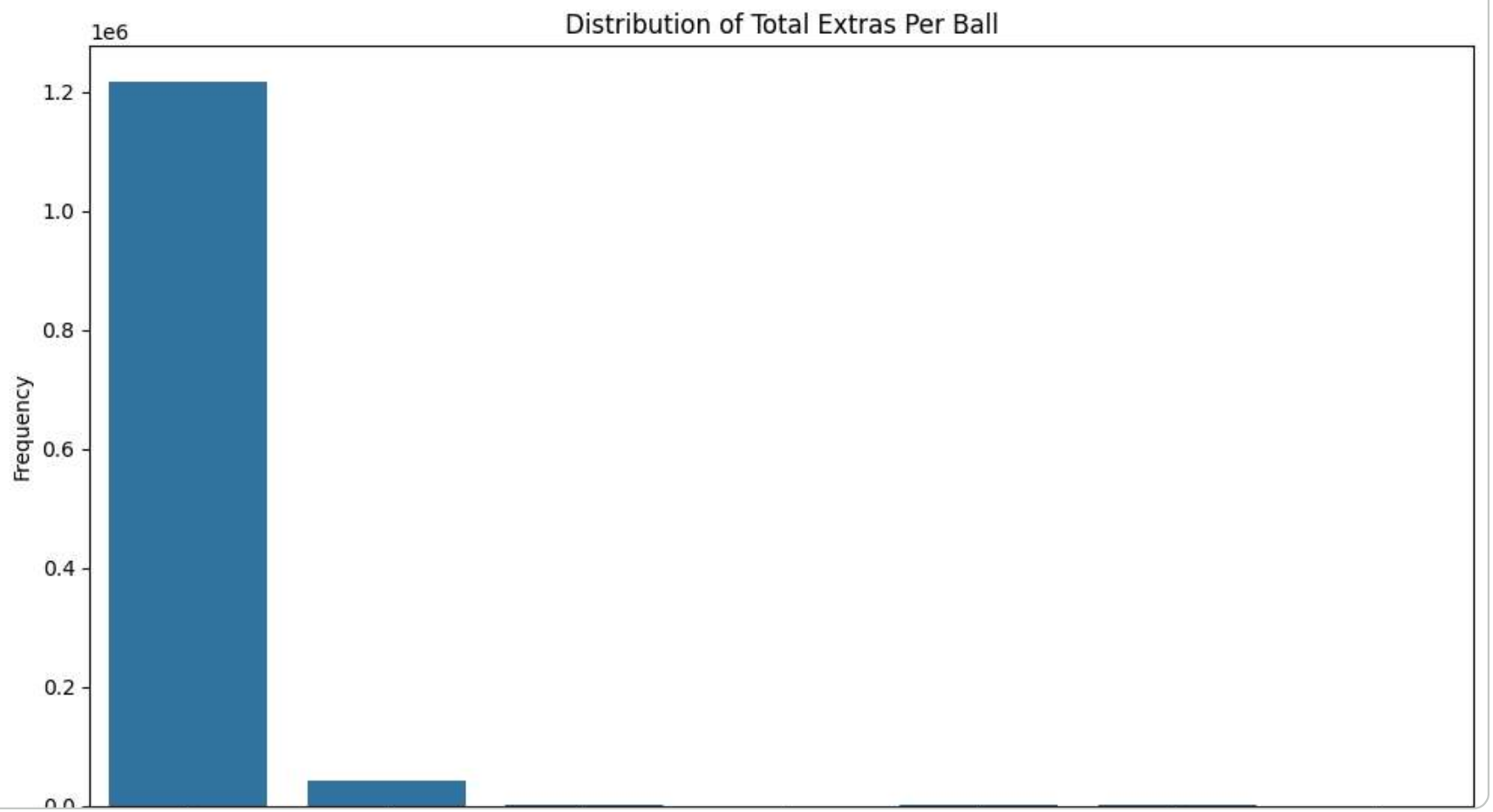
```
import zipfile

zipfile= "/content/ODI_Match_Data.csv.zip"
with zipfile.ZipFile(zip_file_path, 'r') as zip_ref:
    zip_ref.extractall("/content/")
```

```
# Visualize the relationship between 'runs_off_bat' and 'total_extras' using Seaborn scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='runs_off_bat', y='total_extras')
plt.title('Relationship Between Runs Scored and Total Extras Per Ball')
plt.xlabel('Runs Scored Off Bat')
plt.ylabel('Total Extras')
plt.tight_layout()
plt.show()
```




```
# Select the extra columns and fill NaN with 0
extra_cols = ['wides', 'noballs', 'byes', 'legbyes', 'penalty']
df[extra_cols] = df[extra_cols].fillna(0)

# Convert extra columns to numeric, coercing errors
for col in extra_cols:
    df[col] = pd.to_numeric(df[col], errors='coerce')

# Calculate total extras per ball
df['total_extras'] = df[extra_cols].sum(axis=1)

# Drop rows where total_extras is NaN after coercion
df_cleaned_extras = df.dropna(subset=['total_extras'])
```

```
# Visualize the distribution of total extras per ball using Seaborn
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned_extras, x='total_extras')
plt.title('Distribution of Total Extras Per Ball')
plt.xlabel('Total Extras')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

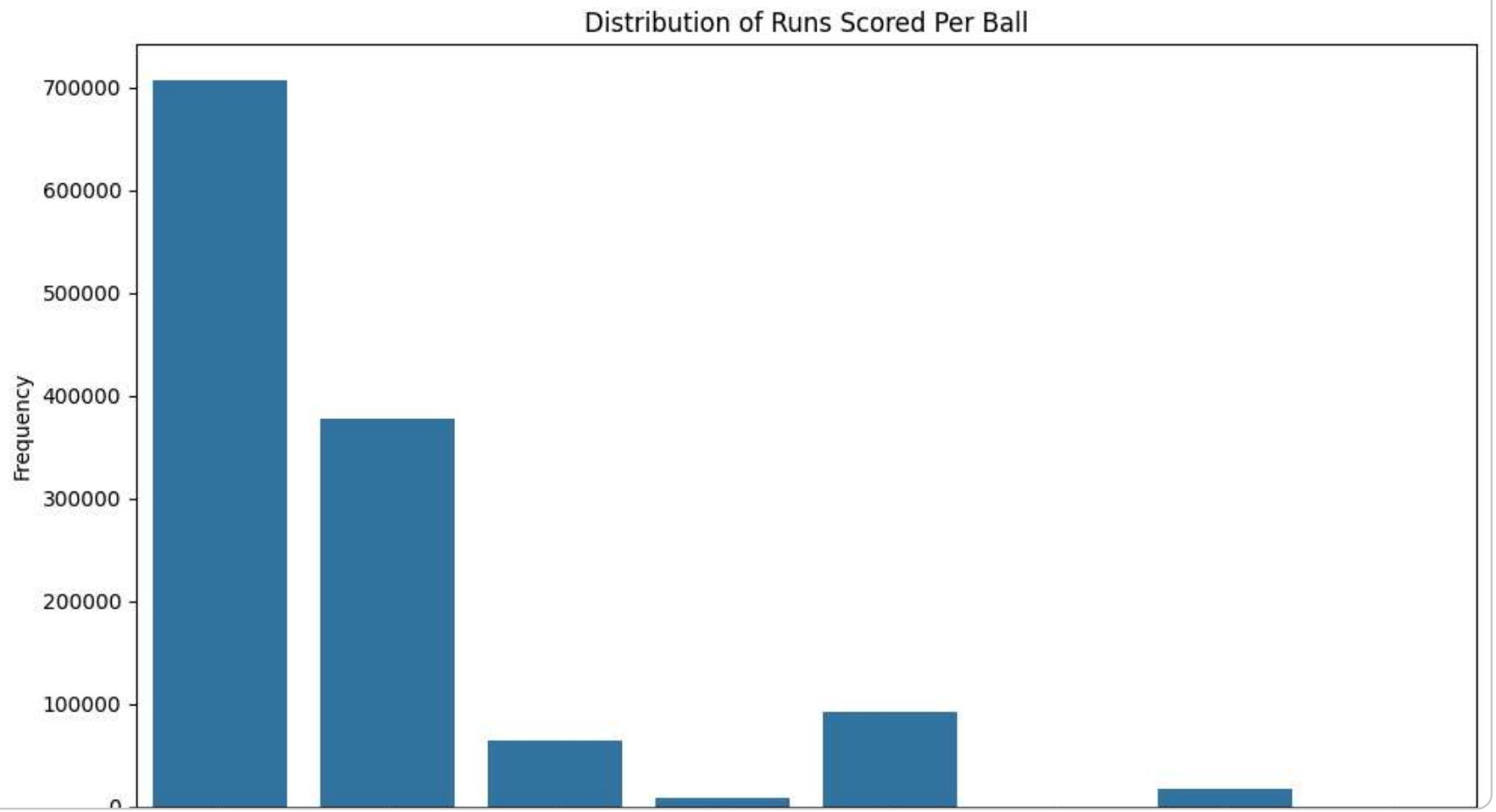



```
print(df.columns)
```

```
Index(['match_id', 'season', 'start_date', 'venue', 'innings', 'ball',
       'batting_team', 'bowling_team', 'striker', 'non_striker', 'bowler',
       'runs_off_bat', 'extras', 'wides', 'noballs', 'byes', 'legbyes',
       'penalty', 'wicket_type', 'player_dismissed', 'other_wicket_type',
       'other_player_dismissed', 'cricsheet_id'],
      dtype='object')
```

```
# Check data types of the 'runs_off_bat' column
print("Data type of 'runs_off_bat' column:", df['runs_off_bat'].dtype)

# Check for missing values in the 'runs_off_bat' column
print("Missing values in 'runs_off_bat' column:", df['runs_off_bat'].isnull().sum())

# If 'runs_off_bat' is not numeric, try to convert it. Coerce errors will turn non-numeric values into NaN.
df['runs_off_bat'] = pd.to_numeric(df['runs_off_bat'], errors='coerce')

# Drop rows where 'runs_off_bat' is NaN after coercion, as they can't be visualized in this way
df_cleaned_runs = df.dropna(subset=['runs_off_bat'])

# Visualize the distribution of runs scored per ball using Seaborn
plt.figure(figsize=(10, 6))
sns.countplot(data=df_cleaned_runs, x='runs_off_bat')
plt.title('Distribution of Runs Scored Per Ball')
plt.xlabel('Runs Scored')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

```
Data type of 'runs_off_bat' column: int64
Missing values in 'runs_off_bat' column: 0
```




```
df_india_venue = df[df['venue'].str.contains('India', na=False)]
print("\nMatches played in India:")
display(df_india_venue.head())
```

```
Matches played in India:
```

	match_id	season	start_date	venue	innings	ball	batting_team	bowling_team	striker	non_st
1052537	267709	2006/07	2007-01-31	Indian Petrochemicals Corporation Limited Spor...	1	0.1	India	West Indies	RV Uthappa	SC G
1052538	267709	2006/07	2007-01-31	Indian Petrochemicals Corporation Limited Spor...	1	0.2	India	West Indies	RV Uthappa	SC G
1052539	267709	2006/07	2007-01-31	Indian Petrochemicals Corporation Limited Spor...	1	0.3	India	West Indies	RV Uthappa	SC G
1052540	267709	2006/07	2007-01-31	Indian Petrochemicals Corporation Limited Spor...	1	0.4	India	West Indies	RV Uthappa	SC G
1052541	267709	2006/07	2007-01-31	Indian Petrochemicals Corporation Limited Spor...	1	0.5	India	West Indies	SC Ganguly	RV Ut

5 rows × 23 columns

Load data

Subtask:

Load the extracted CSV data into a pandas DataFrame.

Reasoning: The subtask is to load the extracted CSV data into a pandas DataFrame and display the first 5 rows. The previous execution failed due to mixed data types, so I will specify low_memory=False when reading the CSV.

```
df = pd.read_csv("/content/ODI_Match_Data.csv", low_memory=False)
print(df.head(5))
```

```
   match_id   season  start_date                          venue  innings  \
0   1389389  2023/24  2023-09-24  Holkar Cricket Stadium, Indore        1
1   1389389  2023/24  2023-09-24  Holkar Cricket Stadium, Indore        1
2   1389389  2023/24  2023-09-24  Holkar Cricket Stadium, Indore        1
3   1389389  2023/24  2023-09-24  Holkar Cricket Stadium, Indore        1
4   1389389  2023/24  2023-09-24  Holkar Cricket Stadium, Indore        1

   ball batting_team bowling_team     striker   non_striker  ... wides  \
0   0.1        India    Australia  RD Gaikwad  Shubman Gill  ...   NaN
1   0.2        India    Australia  RD Gaikwad  Shubman Gill  ...   NaN
2   0.3        India    Australia  RD Gaikwad  Shubman Gill  ...   NaN
3   0.4        India    Australia  RD Gaikwad  Shubman Gill  ...   NaN
4   0.5        India    Australia  RD Gaikwad  Shubman Gill  ...   NaN

   noballs  byes  legbyes  penalty  wicket_type  player_dismissed  \
0      NaN   NaN      NaN      NaN          NaN               NaN
1      NaN   NaN      NaN      NaN          NaN               NaN
2      NaN   NaN      NaN      NaN          NaN               NaN
3      NaN   NaN      NaN      NaN          NaN               NaN
4      NaN   NaN      NaN      NaN          NaN               NaN

   other_wicket_type other_player_dismissed cricsheet_id
0                NaN                    NaN      1389389
1                NaN                    NaN      1389389
2                NaN                    NaN      1389389
3                NaN                    NaN      1389389
4                NaN                    NaN      1389389

[5 rows x 23 columns]
```

Analyze data

Subtask:

Perform sorting and filtering operations on the DataFrame as requested.

Reasoning: Sort and filter the DataFrame as requested in the instructions.

```
df_sorted_date = df.sort_values(by='start_date', ascending=True)
df_innings_1 = df[df['innings'] == 1]
df_india_vs_aus = df[(df['batting_team'] == 'India') & (df['bowling_team'] == 'Australia')]
```

Reasoning: Display the heads of the created dataframes to verify the sorting and filtering operations.

```
print("df_sorted_date head:")
display(df_sorted_date.head())
print("\ndf_innings_1 head:")
display(df_innings_1.head())
print("\ndf_india_vs_aus head:")
display(df_india_vs_aus.head())
```

df_sorted_date head:

	match_id	season	start_date	venue	innings	ball	batting_team	bowling_team	striker	non_striker
1265102	64814	2002/03	2002-12-29	McLean Park, Napier	2	43.4	India	New Zealand	Z Khan	A Nehra
1264717	64814	2002/03	2002-12-29	McLean Park, Napier	1	31.4	New Zealand	India	NJ Astle	MS Sinclair
1264716	64814	2002/03	2002-12-29	McLean Park,	1	31.3	New Zealand	India	NJ Astle	MS Sinclair