Season List:

“MRData”

“SeasonTable”

“Seasons”

Year

Wikipedia URL

Race Schedule

MRData {

RaceTable {

Races [{

season

round

raceName

Circuit {

CircuitId

circuitName

Location {

lat

long

locality

country

date

time

Race Results

MRData {

RaceTable {

Races [{

season

round

race name

Results [{

position

points

Driver[‘driverId’]

Constructor[‘constructorId’]

gird

status

points

Time[‘millis’]

Time[‘time’]

AverageSpeed[‘unit’]

AverageSpeed[‘speed’]

Qualifying Results

MRData

RaceTable

Races [{

season

round

raceName

QualifyingResults {[

Driver[‘driverId’]

Constructor[‘constructorId’]

Q1

Q2

Q3

Link to the data

<https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020>

<https://www.kaggle.com/datasets/jtrotman/formula-1-race-data>

<https://www.kaggle.com/datasets/muhammadehsan02/formula-1-world-championship-history-1950-2024>

pitstop

<https://www.kaggle.com/code/sevrussnape/data-analytics-project#4.-Pit-Stop-Impact-Analysis>

classifier

<https://www.kaggle.com/code/anandaramg/f1-champ-eda-classification-100-accuracy>

pitstop

<https://www.kaggle.com/code/kevinkwan/formula-1-pit-stops-analysis#Total-Time-in-the-Pit-Lane>

lap time

<https://www.kaggle.com/code/kevinkwan/formula-1-lap-time-analysis>

pitstop

<https://www.kaggle.com/code/nandumenon/pitstop-analysis-2>

mechanical failures

<https://www.kaggle.com/code/samarth8/f1-mechanical-failures-and-unlucky-drivers>

1. **Driver Ranking vs Year (Line Chart)**:

**2. Points Accumulated vs Year (Line Chart or Bar Chart):**

**3. Wins by Driver (Bar Chart):**

**4. Year-by-Year Wins per Driver (Grouped Bar Chart):**

**5. Podium Finishes per Driver (Pie Chart or Bar Chart):**

**6. Driver Standings Over Time (Heatmap or Bubble Chart):**

**7. Average Points per Race per Driver (Bar Chart):**

Merging the data

When it comes to merging these datasets, the goal is to combine relevant data from different aspects of the race (e.g., drivers, constructors, circuits, results) into a unified structure that allows for a more comprehensive analysis. Here are some strategies for merging the dataframes based on common keys and how they relate to each other:

### \*\*1. Merging `df\_race\_results` with `df\_driver\_info` and `df\_constructor\_info`\*\*

- \*\*Purpose:\*\* Enrich race results with driver and constructor information.

- \*\*Common Keys:\*\*

- `driverId` from `df\_race\_results` and `df\_driver\_info`

- `constructorId` from `df\_race\_results` and `df\_constructor\_info`

\*\*Example Code:\*\*

```python

# Merging race results with driver information

df\_merged = pd.merge(df\_race\_results, df\_driver\_info, on='driverId', how='left')

# Merging with constructor information

df\_merged = pd.merge(df\_merged, df\_constructor\_info, on='constructorId', how='left')

# Saving the final merged dataframe

df\_merged.to\_csv('merged\_race\_results\_with\_info.csv', index=False)

```

### \*\*2. Merging `df\_quali\_results`, `df\_race\_results`, and `df\_sprint\_results`\*\*

- \*\*Purpose:\*\* Combine race, qualifying, and sprint results for comprehensive performance analysis.

- \*\*Common Keys:\*\*

- `season`, `round`, and `driverId` are common across these datasets.

\*\*Example Code:\*\*

```python

# Merge qualifying results with race results

df\_merged = pd.merge(df\_quali\_results, df\_race\_results, on=['season', 'round', 'driverId'], how='left')

# Merge sprint results with the combined qualifying and race results

df\_merged = pd.merge(df\_merged, df\_sprint\_results, on=['season', 'round', 'driverId'], how='left')

# Saving the final merged dataframe

df\_merged.to\_csv('merged\_race\_quali\_sprint\_results.csv', index=False)

```

### \*\*3. Merging `df\_race\_schedule` with `df\_circuit\_info`\*\*

- \*\*Purpose:\*\* Add circuit information to the race schedule to analyze race performance by circuit.

- \*\*Common Keys:\*\*

- `circuitId` is common between `df\_race\_schedule` and `df\_circuit\_info`.

\*\*Example Code:\*\*

```python

# Merge race schedule with circuit information

df\_merged = pd.merge(df\_race\_schedule, df\_circuit\_info, on='circuitId', how='left')

# Saving the final merged dataframe

df\_merged.to\_csv('merged\_race\_schedule\_with\_circuits.csv', index=False)

```

### \*\*4. Merging `df\_lap\_times` with `df\_race\_results`\*\*

- \*\*Purpose:\*\* Add lap times to race results for lap-by-lap analysis of drivers.

- \*\*Common Keys:\*\*

- `season`, `round`, and `driverId` are common across `df\_lap\_times` and `df\_race\_results`.

\*\*Example Code:\*\*

```python

# Merge lap times with race results

df\_merged = pd.merge(df\_lap\_times, df\_race\_results, on=['season', 'round', 'driverId'], how='left')

# Saving the final merged dataframe

df\_merged.to\_csv('merged\_lap\_times\_with\_race\_results.csv', index=False)

```

### \*\*5. Merging `df\_driver\_standings` with `df\_race\_results`\*\*

- \*\*Purpose:\*\* Analyze driver standings alongside race performance.

- \*\*Common Keys:\*\*

- `season` and `driverId` are common between these datasets.

\*\*Example Code:\*\*

```python

# Merge driver standings with race results

df\_merged = pd.merge(df\_driver\_standings, df\_race\_results, on=['season', 'driverId'], how='left')

# Saving the final merged dataframe

df\_merged.to\_csv('merged\_driver\_standings\_with\_race\_results.csv', index=False)

```

### \*\*6. Merging `df\_pit\_results` with `df\_race\_results`\*\*

- \*\*Purpose:\*\* Add pit stop information to race results for pit stop strategy analysis.

- \*\*Common Keys:\*\*

- `season`, `round`, and `driverId` are common across these datasets.

\*\*Example Code:\*\*

```python

# Merge pit stop results with race results

df\_merged = pd.merge(df\_pit\_results, df\_race\_results, on=['season', 'round', 'driverId'], how='left')

# Saving the final merged dataframe

df\_merged.to\_csv('merged\_pit\_stops\_with\_race\_results.csv', index=False)

```

### \*\*General Strategy:\*\*

- \*\*Left Joins (`how='left'`)\*\*: I recommend using `left` joins because it ensures that you keep all records from the main DataFrame (e.g., race results) even if there isn’t a matching record in the other DataFrame (e.g., driver info or lap times). This prevents loss of data when there are missing matches.

- \*\*Unique Identifiers\*\*: Always ensure that the columns you use to merge (like `driverId`, `season`, `round`, `circuitId`, etc.) are correct and unique to avoid issues with duplicate or missing data.

These merged datasets will provide a richer set of information for analysis and visualization. Let me know if you need help with specific merges or other processing tasks!