import modules

```
conda install psycopg2
```

```
2 channel Terms of Service accepted
Channels:
 - defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
    current version: 25.5.1
    latest version: 25.11.0

Please update conda by running

    $ conda update -n base -c defaults conda



# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
```

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import pickle
import os

# Set up plotting style
sns.set_style("whitegrid")
plt.rcParams['figure.figsize'] = (12, 6)

# Load pickled data files
pickled_path = "./resources/pickled_tables/"

# Load engine and engine_manufacturer data
with open(os.path.join(pickled_path, "engine.plk"), "rb") as f:
    engine_df = pickle.load(f)

with open(os.path.join(pickled_path, "engine_manufacturer.plk"), "rb")
as f:
    engine_manufacturer_df = pickle.load(f)

# Load related tables for comprehensive analysis
```

```
with open(os.path.join(pickled_path,
"season_engine_manufacturer.plk"), "rb") as f:
    season_engine_manufacturer_df = pickle.load(f)

with open(os.path.join(pickled_path, "season_entrant_engine.plk"),
"rb") as f:
    season_entrant_engine_df = pickle.load(f)

with open(os.path.join(pickled_path, "race.plk"), "rb") as f:
    race_df = pickle.load(f)

with open(os.path.join(pickled_path, "season.plk"), "rb") as f:
    season_df = pickle.load(f)

print(f"Engine Data Shape: {engine_df.shape}")
print(f"Engine Manufacturer Data Shape:
{engine_manufacturer_df.shape}")
print(f"Season Engine Manufacturer Data Shape:
{season_engine_manufacturer_df.shape}")
print(f"Season Entrant Engine Data Shape:
{season_entrant_engine_df.shape}")
```

```
Engine Data Shape: (419, 7)
Engine Manufacturer Data Shape: (76, 17)
Season Engine Manufacturer Data Shape: (555, 15)
Season Entrant Engine Data Shape: (2016, 5)
```

Display engine manufacturer data

```
print("Engine Manufacturer DataFrame:")
print(engine_manufacturer_df.head(10))
print("\nEngine Manufacturer Columns:",
engine_manufacturer_df.columns.tolist())
print("\nData Info:")
print(engine_manufacturer_df.info())
```

```
Engine Manufacturer DataFrame:
             id          name      country_id
best_championship_position  \
0          acer          Acer          taiwan
9.0
1    alfa-romeo    Alfa Romeo           italy
3.0
2          alta          Alta  united-kingdom
NaN
3        arrows        Arrows  united-kingdom
7.0
4      asiatech      Asiatech          france
```

```
9.0
5  aston-martin  Aston Martin  united-kingdom
NaN
6           ats           ATS           italy
NaN
7           bmw           BMW         germany
2.0
8      borgward      Borgward         germany
NaN
9           bpm           BPM           italy
NaN

   best_starting_grid_position  best_race_result
total_championship_wins  \
0                          4.0               5.0
0
1                          1.0               1.0
0
2                          6.0               3.0
0
3                          6.0               4.0
0
4                         13.0               5.0
0
5                          2.0               6.0
0
6                         13.0              11.0
0
7                          1.0               1.0
0
8                         16.0              10.0
0
9                          NaN               NaN
0

   total_race_entries  total_race_starts  total_race_wins
total_race_laps  \
0                  17                 17                0
1707
1                 225                215               12
17979
2                  29                 26                0
2610
3                  32                 32                0
2254
4                  34                 33                0
2826
5                   6                  5                0
518
```

```
6                     7                   7                 0
303
7                   273                 270                20
31993
8                     2                   1                 0
139
9                     1                   0                 0
0

   total_podiums  total_podium_races  total_points
total_championship_points  \
0              0                   0           4.0
4.0
1             40                  30         166.0
148.0
2              1                   1           0.0
0.0
3              0                   0           7.0
7.0
4              0                   0           3.0
3.0
5              0                   0           0.0
0.0
6              0                   0           0.0
0.0
7             86                  76        1021.0
1021.0
8              0                   0           0.0
0.0
9              0                   0           0.0
0.0

   total_pole_positions  total_fastest_laps
0                     0                   0
1                    15                  20
2                     0                   0
3                     0                   0
4                     0                   0
5                     0                   0
6                     0                   0
7                    33                  33
8                     0                   0
9                     0                   0

Engine Manufacturer Columns: ['id', 'name', 'country_id',
'best_championship_position', 'best_starting_grid_position',
'best_race_result', 'total_championship_wins', 'total_race_entries',
'total_race_starts', 'total_race_wins', 'total_race_laps',
'total_podiums', 'total_podium_races', 'total_points',
'total_championship_points', 'total_pole_positions',
```

```
'total_fastest_laps']

Data Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76 entries, 0 to 75
Data columns (total 17 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   id                            76 non-null     object 
 1   name                          76 non-null     object 
 2   country_id                    76 non-null     object 
 3   best_championship_position    45 non-null     float64
 4   best_starting_grid_position   72 non-null     float64
 5   best_race_result              65 non-null     float64
 6   total_championship_wins       76 non-null     int64  
 7   total_race_entries            76 non-null     int64  
 8   total_race_starts             76 non-null     int64  
 9   total_race_wins               76 non-null     int64  
 10  total_race_laps               76 non-null     int64  
 11  total_podiums                 76 non-null     int64  
 12  total_podium_races            76 non-null     int64  
 13  total_points                  76 non-null     float64
 14  total_championship_points     76 non-null     float64
 15  total_pole_positions          76 non-null     int64  
 16  total_fastest_laps            76 non-null     int64  
dtypes: float64(5), int64(9), object(3)
memory usage: 10.2+ KB
None
```

Sort by wins and display top performers

```
# Sort by wins and display top performers
top_engines = engine_manufacturer_df.nlargest(15, 'total_race_wins')[
    ['id', 'name', 'country_id', 'total_race_entries',
'total_race_wins',
     'total_podiums', 'total_pole_positions', 'total_fastest_laps']
].reset_index(drop=True)

print("Top 15 Engine Manufacturers by Wins:")
print(top_engines)
```

```
Top 15 Engine Manufacturers by Wins:
            id         name                country_id
total_race_entries  \
0       ferrari      Ferrari                     italy
1120
1      mercedes     Mercedes                   germany
607
```

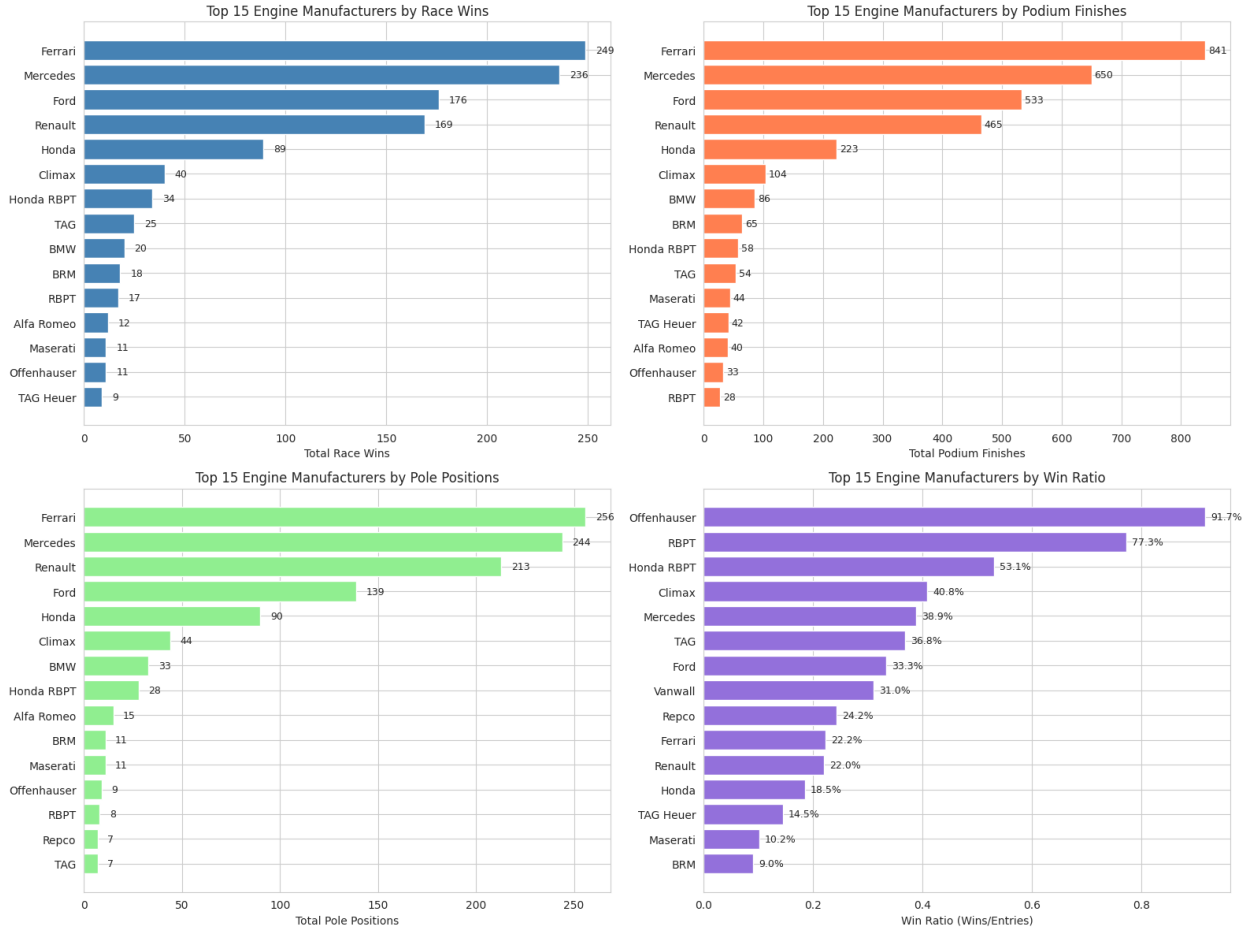
```
2          ford          Ford  united-states-of-america
528
3       renault       Renault                    france
768
4         honda         Honda                     japan
482
5        climax        Climax             united-kingdom
98
6    honda-rbpt   Honda RBPT                     japan
64
7           tag           TAG                luxembourg
68
8           bmw           BMW                   germany
273
9           brm           BRM             united-kingdom
200
10         rbpt          RBPT                     japan
22
11   alfa-romeo   Alfa Romeo                     italy
225
12     maserati     Maserati                     italy
108
13  offenhauser  Offenhauser  united-states-of-america
12
14    tag-heuer    TAG Heuer               switzerland
62

    total_race_wins  total_podiums  total_pole_positions
total_fastest_laps
0               249            841                   256
275
1               236            650                   244
236
2               176            533                   139
162
3               169            465                   213
177
4                89            223                    90
76
5                40            104                    44
45
6                34             58                    28
19
7                25             54                     7
18
8                20             86                    33
33
9                18             65                    11
14
```

```
10                17                28                 8
8
11                12                40                15
20
12                11                44                11
19
13                11                33                 9
10
14                 9                42                 3
13
```

Create subplot visualizations

```
fig, axes = plt.subplots(2, 2, figsize=(16, 12))

top_15_wins = engine_manufacturer_df.nlargest(15, 'total_race_wins')
axes[0, 0].barh(top_15_wins['name'], top_15_wins['total_race_wins'],
color='steelblue')
axes[0, 0].set_xlabel('Total Race Wins')
axes[0, 0].set_title('Top 15 Engine Manufacturers by Race Wins')
axes[0, 0].invert_yaxis()
for i, wins in enumerate(top_15_wins['total_race_wins']):
    axes[0, 0].text(wins + 5, i, f'{int(wins):,}', va='center',
fontsize=9)

top_15_podiums = engine_manufacturer_df.nlargest(15, 'total_podiums')
axes[0, 1].barh(top_15_podiums['name'],
top_15_podiums['total_podiums'], color='coral')
axes[0, 1].set_xlabel('Total Podium Finishes')
axes[0, 1].set_title('Top 15 Engine Manufacturers by Podium Finishes')
axes[0, 1].invert_yaxis()
for i, podiums in enumerate(top_15_podiums['total_podiums']):
    axes[0, 1].text(podiums + 5, i, f'{int(podiums):,}', va='center',
fontsize=9)

top_15_poles = engine_manufacturer_df.nlargest(15,
'total_pole_positions')
axes[1, 0].barh(top_15_poles['name'],
top_15_poles['total_pole_positions'], color='lightgreen')
axes[1, 0].set_xlabel('Total Pole Positions')
axes[1, 0].set_title('Top 15 Engine Manufacturers by Pole Positions')
axes[1, 0].invert_yaxis()
for i, poles in enumerate(top_15_poles['total_pole_positions']):
    axes[1, 0].text(poles + 5, i, f'{int(poles):,}', va='center',
fontsize=9)

engine_manufacturer_df['win_ratio'] =
(engine_manufacturer_df['total_race_wins'] /
```

```
engine_manufacturer_df['total_race_entries']).fillna(0)
top_15_ratio = engine_manufacturer_df.nlargest(15, 'win_ratio')
axes[1, 1].barh(top_15_ratio['name'], top_15_ratio['win_ratio'],
color='mediumpurple')
axes[1, 1].set_xlabel('Win Ratio (Wins/Entries)')
axes[1, 1].set_title('Top 15 Engine Manufacturers by Win Ratio')
axes[1, 1].invert_yaxis()
for i, ratio in enumerate(top_15_ratio['win_ratio']):
    axes[1, 1].text(ratio + 0.01, i, f'{ratio:.1%}', va='center',
fontsize=9)

plt.tight_layout()
plt.savefig('engine_manufacturers_summary.png', dpi=300,
bbox_inches='tight')
plt.show()
print("Engine manufacturer summary visualization saved!")
```




```
Engine manufacturer summary visualization saved!
```

Calculate efficiency metrics

```
# Calculate efficiency metrics
engine_stats = engine_manufacturer_df[
    ['name', 'total_race_entries', 'total_race_wins', 'total_podiums',

     'total_pole_positions', 'total_fastest_laps']
].copy()

# Calculate ratios
engine_stats['podium_ratio'] =
(engine_manufacturer_df['total_podiums'] /

engine_manufacturer_df['total_race_entries']).round(3)
engine_stats['win_ratio'] = (engine_manufacturer_df['total_race_wins']
/

engine_manufacturer_df['total_race_entries']).round(3)
engine_stats['pole_ratio'] =
(engine_manufacturer_df['total_pole_positions'] /

engine_manufacturer_df['total_race_entries']).round(3)
engine_stats['fastest_lap_ratio'] =
(engine_manufacturer_df['total_fastest_laps'] /

engine_manufacturer_df['total_race_entries']).round(3)

# Sort by podium ratio and display top performers
top_performers = engine_stats.sort_values('podium_ratio',
ascending=False).head(15)
print("\nTop 15 Engine Manufacturers by Efficiency (Podium Ratio):")
print(top_performers.to_string())


Top 15 Engine Manufacturers by Efficiency (Podium Ratio):
           name  total_race_entries  total_race_wins  total_podiums
total_pole_positions  total_fastest_laps  podium_ratio  win_ratio
pole_ratio  fastest_lap_ratio
48  Offenhauser                  12               11             33
9                   10         2.750      0.917       0.750
0.833
57         RBPT                  22               17             28
8                    8         1.273      0.773       0.364
0.364
43     Mercedes                 607              236            650
244                  236         1.071      0.389       0.402
0.389
16       Climax                  98               40            104
44                   45         1.061      0.408       0.449
0.459
```

```
25          Ford                  528               176             533
139                 162          1.009      0.333       0.263
0.307
28   Honda RBPT                    64                34              58
28                   19          0.906      0.531       0.438
0.297
66           TAG                   68                25              54
7                   18          0.794      0.368       0.103
0.265
59         Repco                   33                 8              25
7                    4          0.758      0.242       0.212
0.121
23       Ferrari                 1120               249             841
256                 275          0.751      0.222       0.229
0.246
65    TAG Heuer                    62                 9              42
3                   13          0.677      0.145       0.048
0.210
58       Renault                  768               169             465
213                 177          0.605      0.220       0.277
0.230
29         Honda                  482                89             223
90                   76          0.463      0.185       0.187
0.158
71       Vanwall                   29                 9              13
7                    6          0.448      0.310       0.241
0.207
39     Maserati                   108                11              44
11                   19          0.407      0.102       0.102
0.176
11           BRM                   200                18              65
11                   14          0.325      0.090       0.055
0.070
```

```
print(season_df.columns.tolist())
print(season_df.head())
```

```
['year']
   year
0  1950
1  1951
2  1952
3  1953
4  1954
```

Generate comprehensive summary

```
# Generate comprehensive summary
summary_stats = engine_manufacturer_df[
    ['total_race_entries', 'total_race_wins', 'total_podiums',
     'total_pole_positions', 'total_fastest_laps']
].describe()

print("\nSummary Statistics for Engine Manufacturers:")
print(summary_stats.round(2))

# Key insights
print("\n=== KEY INSIGHTS ===")
print(f"Total Manufacturers: {len(engine_manufacturer_df)}")
print(f"Total Race Entries:
{engine_manufacturer_df['total_race_entries'].sum()}")
print(f"Total Races Won:
{engine_manufacturer_df['total_race_wins'].sum()}")
print(f"Total Podiums:
{engine_manufacturer_df['total_podiums'].sum()}")

# Top performers
print("\nMost Successful Engine Manufacturer:")
top =
engine_manufacturer_df.loc[engine_manufacturer_df['total_race_wins'].i
dxmax()]
print(f"  {top['name']}: {int(top['total_race_wins'])} wins")

print("\nHighest Win Ratio:")
engine_manufacturer_df['win_ratio'] =
(engine_manufacturer_df['total_race_wins'] /

engine_manufacturer_df['total_race_entries'])
top_ratio =
engine_manufacturer_df.loc[engine_manufacturer_df['win_ratio'].idxmax(
)]
print(f"  {top_ratio['name']}: {top_ratio['win_ratio']:.3f} wins per
entry")
```

```
Summary Statistics for Engine Manufacturers:
       total_race_entries  total_race_wins  total_podiums  \
count               76.00            76.00          76.00
mean                89.43            15.04          45.11
std                183.62            48.03         144.38
min                  1.00             0.00           0.00
25%                  6.00             0.00           0.00
50%                 21.00             0.00           1.00
75%                 77.00             1.50          14.50
max               1120.00           249.00         841.00
```

```
       total_pole_positions  total_fastest_laps
count                 76.00               76.00
mean                  15.04               15.25
std                   49.70               49.29
min                    0.00                0.00
25%                    0.00                0.00
50%                    0.00                0.00
75%                    2.25                3.25
max                  256.00              275.00

=== KEY INSIGHTS ===
Total Manufacturers: 76
Total Race Entries: 6797
Total Races Won: 1143
Total Podiums: 3428

Most Successful Engine Manufacturer:
  Ferrari: 249 wins

Highest Win Ratio:
  Offenhauser: 0.917 wins per entry
```