

premier-league-table

March 19, 2025

1 Premier league table for season 2024/2025

This script processes match results from structured CSV datasets to generate a dynamic league standings table. It calculates team rankings based on points, tracks games played (GP), and removes unnecessary name extensions for a cleaner display. The script also identifies the latest match date to indicate when the results were last updated. Using Pandas for data processing and Tabulate for structured output, it provides a formatted, easy-to-read table that closely resembles official league standings. Ideal for football analysts, data enthusiasts, and developers, this script simplifies league table generation with just a few lines of code.

1.0.1 Libraries

```
[ ]: # install if needed

import pandas as pd
from tabulate import tabulate # Using tabulate for better formatting
import zipfile
```

1.0.2 Data import

Downloading data from Kaggle using their API.

```
[69]: # Download without unzipping
!kaggle datasets download -d davidcariboo/player-scores

# Unzip only the required files

with zipfile.ZipFile("player-scores.zip", 'r') as z:
    z.extract("clubs.csv")
    z.extract("games.csv")

print("Extracted clubs.csv and games.csv!")

# Load the data
games_df = pd.read_csv("games.csv", delimiter=",") # Adjust delimiter if needed
clubs_df = pd.read_csv("clubs.csv", delimiter=",") # Adjust delimiter if needed
print("\n First 5 rows of clubs.csv:")
print(clubs_df.head())
```

```
print("\n First 5 rows of games.csv:")
print(games_df.head())
```

Dataset URL: <https://www.kaggle.com/datasets/davidcariboo/player-scores>

License(s): CC0-1.0

player-scores.zip: Skipping, found more recently modified local copy (use --force to force download)

Extracted clubs.csv and games.csv!

First 5 rows of clubs.csv:

	club_id	club_code	name \
0	105	sv-darmstadt-98	SV Darmstadt 98
1	11127	ural-ekaterinburg	Ural Yekaterinburg
2	114	besiktas-istanbul	Beşiktaş Jimnastik Kulübü
3	12	as-rom	Associazione Sportiva Roma
4	148	tottenham-hotspur	Tottenham Hotspur Football Club

	domestic_competition_id	total_market_value	squad_size	average_age \
0	L1	NaN	27	25.6
1	RU1	NaN	30	26.5
2	TR1	NaN	30	26.5
3	IT1	NaN	26	26.3
4	GB1	NaN	30	25.5

	foreigners_number	foreigners_percentage	national_team_players \
0	13	48.1	1
1	11	36.7	3
2	15	50.0	8
3	18	69.2	17
4	21	70.0	19

	stadium_name	stadium_seats	net_transfer_record \
0	Merck-Stadion am Böllenfalltor	17810	+€3.05m
1	Yekaterinburg Arena	23000	+€880k
2	Beşiktaş Park	42445	€-25.26m
3	Olimpico di Roma	70634	€-76.90m
4	Tottenham Hotspur Stadium	62850	€-120.05m

	coach_name	last_season	filename \
0	NaN	2023	../data/raw/transfermarkt-scraper/2023/clubs.j...
1	NaN	2023	../data/raw/transfermarkt-scraper/2023/clubs.j...
2	NaN	2024	../data/raw/transfermarkt-scraper/2024/clubs.j...
3	NaN	2024	../data/raw/transfermarkt-scraper/2024/clubs.j...
4	NaN	2024	../data/raw/transfermarkt-scraper/2024/clubs.j...

url

```

0 https://www.transfermarkt.co.uk/sv-darmstadt-9...
1 https://www.transfermarkt.co.uk/ural-ekaterinb...
2 https://www.transfermarkt.co.uk/besiktas-istan...
3 https://www.transfermarkt.co.uk/as-rom/startse...
4 https://www.transfermarkt.co.uk/tottenham-hots...

```

First 5 rows of games.csv:

	game_id	competition_id	season	round	date	home_club_id	\
0	2321027	L1	2013	1. Matchday	2013-08-11	33.0	
1	2321033	L1	2013	1. Matchday	2013-08-10	23.0	
2	2321044	L1	2013	2. Matchday	2013-08-18	16.0	
3	2321060	L1	2013	3. Matchday	2013-08-25	23.0	
4	2321072	L1	2013	5. Matchday	2013-09-14	16.0	

	away_club_id	home_club_goals	away_club_goals	home_club_position	...	\
0	41.0	3.0	3.0	8.0	...	
1	86.0	0.0	1.0	13.0	...	
2	23.0	2.0	1.0	1.0	...	
3	24.0	0.0	2.0	18.0	...	
4	41.0	6.0	2.0	1.0	...	

	stadium	attendance	referee	\
0	Veltins-Arena	61973.0	Manuel Gräfe	
1	EINTRACHT-Stadion	23000.0	Deniz Aytekin	
2	SIGNAL IDUNA PARK	80200.0	Peter Sippel	
3	EINTRACHT-Stadion	23325.0	Wolfgang Stark	
4	SIGNAL IDUNA PARK	80645.0	Tobias Welz	

	url	home_club_formation	\
0	https://www.transfermarkt.co.uk/fc-schalke-04_...	4-2-3-1	
1	https://www.transfermarkt.co.uk/eintracht-brau...	4-3-2-1	
2	https://www.transfermarkt.co.uk/borussia-dortm...	4-2-3-1	
3	https://www.transfermarkt.co.uk/eintracht-brau...	4-3-2-1	
4	https://www.transfermarkt.co.uk/borussia-dortm...	4-2-3-1	

	away_club_formation	home_club_name	\
0	4-2-3-1	FC Schalke 04	
1	4-3-1-2	Eintracht Braunschweig	
2	4-3-2-1	Borussia Dortmund	
3	4-2-3-1	Eintracht Braunschweig	
4	3-5-2	Borussia Dortmund	

	away_club_name	aggregate	competition_type
0	Hamburger SV	3:3	domestic_league
1	Sportverein Werder Bremen von 1899	0:1	domestic_league
2	Eintracht Braunschweig	2:1	domestic_league
3	Eintracht Frankfurt Fußball AG	0:2	domestic_league
4	Hamburger SV	6:2	domestic_league

[5 rows x 23 columns]

1.0.3 Data manipulation

```
[70]: # Convert "date" column to datetime format
games_df["date"] = pd.to_datetime(games_df["date"], errors="coerce")

# Find the latest match date in the dataset for Premier League (GB1)
latest_game_date = games_df[games_df["competition_id"] == "GB1"]["date"].max()

# Filter for Premier League (GB1) and Season 2024
filtered_games = games_df[(games_df["season"] == 2024) &
    ↪(games_df["competition_id"] == "GB1")].copy()

# Merge to get club names instead of IDs
clubs_map = clubs_df.set_index("club_id")["name"].to_dict()
filtered_games["home_club_name"] = filtered_games["home_club_id"].map(clubs_map)
filtered_games["away_club_name"] = filtered_games["away_club_id"].map(clubs_map)

# Initialize points table and games played table
points_table = {}
games_played = {}

# Calculate points and games played
for _, row in filtered_games.iterrows():
    home_team = row["home_club_name"]
    away_team = row["away_club_name"]
    home_goals = row["home_club_goals"]
    away_goals = row["away_club_goals"]

    # Initialize team data if not already present
    points_table.setdefault(home_team, 0)
    points_table.setdefault(away_team, 0)
    games_played.setdefault(home_team, 0)
    games_played.setdefault(away_team, 0)

    # Count games played
    games_played[home_team] += 1
    games_played[away_team] += 1

    # Assign points based on match results
    if home_goals > away_goals:
        points_table[home_team] += 3 # Home team wins
    elif home_goals < away_goals:
        points_table[away_team] += 3 # Away team wins
    else:
```

```
points_table[home_team] += 1 # Draw
points_table[away_team] += 1 # Draw
```

1.0.4 Data formation for displaying results

```
[73]: # Function to clean team names
def clean_team_name(name):
    return (
        name.replace(" Football Club", "")
        .replace("Association Football Club ", "")
        .strip()
    )

# Convert dictionary to a DataFrame
points_df = pd.DataFrame({
    "Team": [clean_team_name(team) for team in points_table.keys()],
    "Points": points_table.values(),
    "GP": [games_played[team] for team in points_table.keys()]
})

# Sort by points in descending order and reset index to create "Position"
points_df = points_df.sort_values(by="Points", ascending=False).
    ↪reset_index(drop=True)
points_df.index += 1 # Start index from 1 instead of 0
points_df.index.name = "Position" # Rename index to "Position"
```

1.0.5 Displaying the results

```
[74]: # Display final table using tabulate
print("Premier League Table (season 2024/2025):")
print(tabulate(points_df, headers="keys", tablefmt="fancy_grid"))
print(f"\nResults updated: {latest_game_date.strftime('%Y-%m-%d')}")
```

Premier League Table (season 2024/2025):

Position	Team	Points	GP
1	Liverpool	70	29
2	Arsenal	58	29
3	Nottingham Forest	54	29
4	Chelsea	49	29
5	Manchester City	48	29

6	Newcastle United	47	28
7	Brighton and Hove Albion	47	29
8	Fulham	45	29
9	Aston Villa	45	29
10	Association Bournemouth	44	29
11	Brentford	41	29
12	Crystal Palace	39	28
13	Manchester United	37	29
14	Tottenham Hotspur	34	29
15	West Ham United	34	29
16	Everton	34	29
17	Wolverhampton Wanderers	26	29
18	Leicester City	17	29
19	Ipswich Town	17	29
20	Southampton	9	29

Results updated: 2025-03-16