```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
import pandas as pd

# Load CSV files - Deze hebben wij al manueel schoongemaakt aan de hand van onze controles,
match_results_df = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/csv/wedstrijden_onve
goal_events_df = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/csv/doelpunten_onverwe
standings_df = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/csv/klassement_onverwerk

#Filter voor valid goals
goal_events_df = goal_events_df[goal_events_df['valid_goal'] == True]

# Filter of vervang niet-numerieke waarden
match_results_df['doelpunten_thuisploeg'] = pd.to_numeric(match_results_df['doelpunten_thui
match_results_df['doelpunten_uitploeg'] = pd.to_numeric(match_results_df['doelpunten_uitplo

# Conversie naar int
match_results_df['doelpunten_thuisploeg'] = match_results_df['doelpunten_thuisploeg'].astyp
match_results_df['doelpunten_uitploeg'] = match_results_df['doelpunten_uitploeg'].astype(in


# Vind alle verplaatste matches, namelijk de match id zal dezelfde zijn als de vorige
match_results_df['same_as_previous_match'] = match_results_df['id_match'].eq(match_results_

# Selecteer alleen de rijen waar 'same_as_previous_match' True is
duplicate_matches = match_results_df[match_results_df['same_as_previous_match']]
duplicate_matches.head(30)
```

	seizoen	speeldag	datum	tijd	id_match	thuisploeg_stamnummer	thuisploeg	uitplo

```
# Bereken goals
goals_home_team = goal_events_df[goal_events_df['goal_team_stamnummer'] == goal_events_df[
goals_away_team = goal_events_df[goal_events_df['goal_team_stamnummer'] == goal_events_df[

# Omzetting
goals_home_team_df = goals_home_team.reset_index(name='calculated_doelpunten_thuisploeg')
goals_away_team_df = goals_away_team.reset_index(name='calculated_doelpunten_uitploeg')

# Vervang NaN values in goal kolom met 0
merged_df = pd.merge(match_results_df, goals_home_team_df, on='id_match', how='outer')
merged_df = pd.merge(merged_df, goals_away_team_df, on='id_match', how='outer')
merged_df['calculated_doelpunten_thuisploeg'] = merged_df['calculated_doelpunten_thuisploeg
merged_df['calculated_doelpunten_uitploeg'] = merged_df['calculated_doelpunten_uitploeg'].f
```

```
# Bereken verschil tussen berekening en resultaat
discrepancies_goals = merged_df[
    ((merged_df['calculated_doelpunten_thuisploeg'] != merged_df['doelpunten_thuisploeg'])
    (merged_df['calculated_doelpunten_uitploeg'] != merged_df['doelpunten_uitploeg'])) &
    ~((merged_df['doelpunten_thuisploeg'] == 5) & (merged_df['doelpunten_uitploeg'] == 0))
    ~((merged_df['doelpunten_thuisploeg'] == 0) & (merged_df['doelpunten_uitploeg'] == 5))
]

# Toon alle matches waarbij de berekende goals niet overeenkomen met echte - Lommel 2002, E
discrepancies_goals.head(1000)
```

	seizoen	speeldag	datum	tijd	id_match	thuisploeg_stamnummer	thuisp
13329	1977	14	1977/11/20	15:00:00	4012716	19	KV K
13330	1977	15	1977/11/26	19:30:00	4012718	22	R Cha
13335	1977	15	1977/11/27	15:00:00	4012722	2300	Bev
13337	1977	15	1977/11/27	15:00:00	4012724	30	Liers
13340	1977	16	1977/12/03	20:00:00	4012726	35	Ander
...	...	...	...	...	...	...	
15610	1969	15	1969/12/14	14:30:00	4010466	16	Star
15611	1969	15	1969/12/14	14:30:00	4010467	35	Ander
15612	1969	15	1969/12/14	14:30:00	4010468	373	Truid
15613	1969	15	1969/12/14	14:30:00	4010469	2300	Bev
15614	1969	15	1969/12/14	14:30:00	4010470	22	R Cha

1000 rows × 14 columns

```
#Controle voor (5:0 of 0:5's)
discrepancies_goals = merged_df[
    (merged_df['calculated_doelpunten_thuisploeg'] != merged_df['doelpunten_thuisploeg'])
    (merged_df['calculated_doelpunten_uitploeg'] != merged_df['doelpunten_uitploeg'])
]
#Wordt gebruikt om manueel alle 5:0 of 0:5's te controleren aangezien het technisch gezie
discrepancies_goals.head(1000)
```

```
display(standings_df.head(1000))
```

	seizoen	speeldag	datum	tijd	id_match	thuisploeg_stamnummer	thuisp
356	2022	14	2022/10/23	18:30:00	3851182	16	Star
380	2022	17	2022/11/12	20:45:00	3851244	22	R Cha
728	2021	22	2022/01/15	16:15:00	3598295	18	Hev Le
841	2021	34	2022/04/10	18:30:00	3598351	10	U S Gi
2427	2014	11	2014/10/19	14:30:00	2503727	16	Star
...	...	...	...	...	...	...	
15573	1969	10	1969/11/09	15:00:00	4010432	10	U S Gi
15574	1969	10	1969/11/09	15:00:00	4010433	53	Oost
15576	1969	10	1969/11/11	15:00:00	4010435	373	Truid
15577	1969	10	1970/03/22	16:00:00	4010565	30	Liers
15578	1969	11	1969/11/15	20:00:00	4010436	22	R Cha

1000 rows × 14 columns

```
# controle van aantal_wedstrijden
# Geen enkel record met meer wedstrijden dan speeldagen
number_of_games_check = standings_df[standings_df.speeldag < standings_df.aantal_wedstrij
number_of_games_check.head(30)
```

	seizoen	speeldag	stand	naam_ploeg	id_ploeg	aantal_wedstrijden	aantal_gewonnen

```
# controle van aantal_wedstrijden
# wel records met minder wedstrijden dan speeldagen, maar bij nazicht van een aantal reco
number_of_games_check = standings_df[standings_df.speeldag > standings_df.aantal_wedstrij
```

```
number_of_games_check.seizoen.value_counts()
```

```
1975    551
2002    442
2009    360
Name: seizoen, dtype: int64
```

```
# geen records met aantal wedstrijden != win + gelijk + verlies
number_of_games_check = standings_df[standings_df.aantal_wedstrijden != (standings_df.aan
number_of_games_check.count()
```

```
seizoen               0
speeldag              0
stand                 0
naam_ploeg            0
id_ploeg              0
aantal_wedstrijden    0
aantal_gewonnen       0
aantal_gelijk         0
aantal_verloren       0
doelpunten_voor       0
doelpunten_tegen      0
verschil              0
punten                0
dtype: int64
```

```
# geen records met doelpunten voor - doelpunten tegen != doelpunten verschil
standings_df['calculated_goal_verschil'] = standings_df['doelpunten_voor'] - standings_df
controle_aantal_wedstrijden = standings_df[standings_df['calculated_goal_verschil'] != st
controle_aantal_wedstrijden.count()
```

```
seizoen                    0
speeldag                   0
stand                      0
naam_ploeg                 0
id_ploeg                   0
aantal_wedstrijden         0
aantal_gewonnen            0
aantal_gelijk              0
aantal_verloren            0
doelpunten_voor            0
doelpunten_tegen           0
verschil                   0
punten                     0
calculated_goal_verschil   0
dtype: int64
```

```
# het totaal aantal gewonnen matchen per seizoen en per ploeg
points_df = standings_df.groupby(['seizoen','speeldag','id_ploeg', 'punten'])[['aantal_ge
points_df = points_df.reset_index()
points_df = points_df.sort_values(['seizoen','speeldag','aantal_gewonnen','aantal_gelijk'

#Controleer de punten, alle waarden voor 1960 herberekenen we later dus alles is goed
```

```
points_df['punten_check'] = points_df['aantal_gewonnen'] * 3 + points_df['aantal_gelijk']
discrepancies_df = points_df[points_df['punten'] != points_df['punten_check']]
discrepancies_df.head(100)
```

	seizoen	speeldag	id_ploeg	punten	aantal_gewonnen	aantal_gelijk	aantal_verl
1	1960	1	2	2:0	1	0	
3	1960	1	4	2:0	1	0	
5	1960	1	10	2:0	1	0	
6	1960	1	13	2:0	1	0	
7	1960	1	16	2:0	1	0	
...	...	...	...	...	...	...	
95	1960	6	3434	2:10	0	2	
103	1960	7	16	12:2	5	2	
97	1960	7	2	10:4	4	2	
99	1960	7	4	10:4	4	2	
107	1960	7	90	9:5	4	1	

100 rows × 8 columns

Next steps: View recommended plots

```
match_results_df['datum'] = pd.to_datetime(match_results_df['datum'])
goal_events_df['datum'] = pd.to_datetime(goal_events_df['datum'])

merged_df = pd.merge(goal_events_df, match_results_df, on='id_match', suffixes=('_goal',

# Check dat de goal datum de matchdatum match
date_discrepancies_df = merged_df[merged_df['datum_goal'] != merged_df['datum_match']]
date_discrepancies_df.head(30)
```

	seizoen_goal	dag	datum_goal	tijd_goal	id_match	thuisploeg_stamnummer_goal	thu

0 rows × 27 columns

```
# Groepeer de gegevens op 'Day' en controleer de datums
sorted_dates = match_results_df.sort_values(by=['speeldag', 'datum'])
date_discrepancies_df = sorted_dates.groupby('speeldag')['datum'].apply(lambda x: x.is_mo

date_discrepancies_df = date_discrepancies_df[date_discrepancies_df == False]
date_discrepancies_df.head(30)
```

```
Series([], Name: datum, dtype: bool)
```

```
# Sorting the data by season, day, and date
sorted_dates = match_results_df.sort_values(by=['seizoen', 'speeldag', 'datum'])

# Group by 'season' and 'day', and check if dates are in chronological order within each
grouped = sorted_dates.groupby(['seizoen', 'speeldag'])
date_order_check = grouped['datum'].apply(lambda x: x.is_monotonic_increasing)

# Alle seizoenen hebben de datum in chronologische orde
date_discrepancies = date_order_check[date_order_check == False]
date_discrepancies.head(30)
```

```
Series([], Name: datum, dtype: bool)
```

```
# Filter matches met 0-0
matches_0_0 = match_results_df[(match_results_df['doelpunten_thuisploeg'] == 0) & (match_

match_ids_0_0 = matches_0_0['id_match']

# Check dat deze matches oprecht geen goals hebben
goals_in_0_0_matches = goal_events_df[goal_events_df['id_match'].isin(match_ids_0_0)]
goals_in_0_0_matches.head(100)
```

	seizoen	dag	datum	tijd	id_match	thuisploeg_stamnummer	thuisploeg	uitploeg	u:

```
import pandas as pd

# Zet relevante kolommen om naar numeriek
standings_df['punten'] = pd.to_numeric(standings_df['punten'], errors='coerce')
standings_df['aantal_gewonnen'] = pd.to_numeric(standings_df['aantal_gewonnen'], errors='
standings_df['calculated_goal_verschil'] = pd.to_numeric(standings_df['calculated_goal_ve
standings_df['doelpunten_voor'] = pd.to_numeric(standings_df['doelpunten_voor'], errors='

#Bereken twee punten systeem
def calculate_two_point_system_left(row):
    return 2 * row['aantal_gewonnen'] + 1 * row['aantal_gelijk']

def calculate_two_point_system_right(row):
    total_games = row['aantal_gewonnen'] + row['aantal_gelijk'] + row['aantal_verloren']
    return 2 * total_games - calculate_two_point_system_left(row)

# Functie om punten in het driepuntensysteem te berekenen voor het gehele seizoen
def calculate_three_point_system(row):
    return 3 * row['aantal_gewonnen'] + 1 * row['aantal_gelijk']

# Voeg de berekende kolommen toe aan de DataFrame
standings_df['punten_2p_links'] = standings_df.apply(calculate_two_point_system_left, axi
```

```
standings_df['punten_2p_links'] = standings_df.apply(calculate_two_point_system_left, axi
standings_df['punten_2p_rechts'] = standings_df.apply(calculate_two_point_system_right, a
standings_df['punten_3p'] = standings_df.apply(calculate_three_point_system, axis=1)

# Bereid een DataFrame voor om de resultaten op te slaan
all_seasons_ranked = pd.DataFrame()

# Loop over elk seizoen
for seizoen in standings_df['seizoen'].unique():
    # Filter de DataFrame voor het huidige seizoen
    filtered_df = standings_df[standings_df['seizoen'] == seizoen]

    # Sorteer en bereken rangorde
    sorted_df = filtered_df.sort_values(by=['speeldag', 'punten_3p', 'aantal_gewonnen', '
    sorted_df['calculated_rank'] = sorted_df.groupby('speeldag')['stand'].rank(method='mi

    # Voeg de berekende rangorde toe aan de oorspronkelijke DataFrame
    filtered_df = filtered_df.merge(sorted_df[['seizoen', 'speeldag', 'id_ploeg', 'calcul

    # Voeg toe aan het totale resultaat
    all_seasons_ranked = pd.concat([all_seasons_ranked, filtered_df], ignore_index=True)

# Controleer op discrepanties voor alle seizoenen
discrepancies_standing = all_seasons_ranked[all_seasons_ranked['stand'] != all_seasons_ra

#De huidige errors hebben te maken met verwijderde data dat wij anders aanpakte dan de we
discrepancies_standing.head(1000)
```

	seizoen	speeldag	stand	naam_ploeg	id_ploeg	aantal_wedstrijden	aantal_gew
11246	2002	1	12	St-Truidense VV	373	1	
11247	2002	1	12	KRC Genk	322	1	
11248	2002	1	14	La Louviere	94	1	
11249	2002	1	15	R Charleroi SC	22	1	
11250	2002	1	16	KVC Westerlo	2024	1	
11251	2002	1	17	RAEC Mons	44	1	
11252	2002	1	17	Standard Luik	16	1	
11268	2002	2	17	St-Truidense VV	373	1	
11269	2002	2	18	KVC	2024	2	

				Westerlo		
11286	2002	3	18	KVC Westerlo	2024	3

Next steps: View recommended plots

```
# Toon de eerste 10 rijen om te controleren manueel
standings_df.head(30)
```

	seizoen	speeldag	stand	naam_ploeg	id_ploeg	aantal_wedstrijden	aantal_gewonn
0	2023	1	1	KRC Genk	322	1	
1	2023	1	2	Union SG	10	1	
2	2023	1	3	KAA Gent	7	1	
3	2023	1	3	St-Truidense VV	373	1	
4	2023	1	3	Antwerp FC	1	1	
5	2023	1	6	R Charleroi SC	22	1	
6	2023	1	6	KV Mechelen	25	1	
7	2023	1	6	KVC Westerlo	2024	1	
8	2023	1	6	KAS Eupen	4276	1	
9	2023	1	6	Club Brugge	3	1	
10	2023	1	6	OH Leuven	18	1	
11	2023	1	12	Cercle Brugge	12	1	
12	2023	1	12	KV Kortrijk	19	1	
13	2023	1	12	Standard Luik	16	1	
14	2023	1	15	RSC Anderlecht	35	1	
15	2023	1	16	RWDM	5479	1	
16	2023	2	1	Union SG	10	2	
17	2023	2	2	KAA Gent	7	2	

18	2023	2	2	St-Truidense VV	373	2
19	2023	2	4	KAS Eupen	4276	2
20	2023	2	4	Club Brugge	3	2
21	2023	2	6	KRC Genk	322	2
22	2023	2	7	Cercle Brugge	12	2
23	2023	2	8	Antwerp FC	1	2
24	2023	2	9	RSC Anderlecht	35	2
25	2023	2	10	RWDM	5479	2
26	2023	2	11	KV Mechelen	25	2
27	2023	2	11	KVC Westerlo	2024	2
28	2023	2	11	OH Leuven	18	2
29	2023	2	14	R Charleroi SC	22	2

Next steps: View recommended plots

```
standings_df = standings_df.drop(['punten', 'calculated_goal_verschil'], axis=1)
# Huidige volgorde van kolommen
huidige_volgorde = list(standings_df.columns)

# Index van de kolommen die moeten worden omgewisseld
index_kolom1 = huidige_volgorde.index('naam_ploeg')
index_kolom2 = huidige_volgorde.index('id_ploeg')

# Omwisselen van de volgorde
huidige_volgorde[index_kolom1], huidige_volgorde[index_kolom2] = huidige_volgorde[index_k

# Nieuwe DataFrame met gewijzigde kolomvolgorde
standings_df = standings_df[huidige_volgorde]
standings_df.to_csv('/content/drive/MyDrive/Colab_Notebooks/csv/klassement.csv', sep=';',
standings_df.tail(30)
```

	seizoen	speeldag	stand	id_ploeg	naam_ploeg	aantal_wedstrijden	aantal_gew

35898	1960	29	3	35	RSC Anderlecht	29
35899	1960	29	4	2	Daring Club	29
35900	1960	29	5	13	Beerschot AC	29
35901	1960	29	6	553	Waterschei THOR	29
35902	1960	29	7	373	St-Truidense VV	29
35903	1960	29	8	90	Eendracht Aalst	29
35904	1960	29	9	1	Antwerp FC	29
35905	1960	29	10	3	Club Brugge	29
35906	1960	29	11	7	ARA Gent	29
35907	1960	29	12	30	Lierse SK	29
35908	1960	29	13	246	Olymp. Charleroi	29
35909	1960	29	14	10	Union SG	29
35910	1960	29	15	3434	K Patro Eisden	29
35911	1960	29	16	33	Verviétois	29
35912	1960	30	1	16	Standard Luik	30
35913	1960	30	2	4	RFC Luik	30
35914	1960	30	3	35	RSC Anderlecht	30
35915	1960	30	4	13	Beerschot AC	30
35916	1960	30	5	2	Daring Club	30
35917	1960	30	6	553	Waterschei THOR	30
35918	1960	30	7	373	St-Truidense VV	30
35919	1960	30	8	1	Antwerp FC	30

35920	1960	30	9	3	Club Brugge	30
35921	1960	30	10	7	ARA Gent	30
35922	1960	30	11	30	Lierse SK	30
35923	1960	30	12	90	Eendracht Aalst	30
35924	1960	30	13	246	Olymp. Charleroi	30
35925	1960	30	14	10	Union SG	30
35926	1960	30	15	33	Verviétois	30
35927	1960	30	16	3434	K Patro Eisden	30

```
#Exporteer het doelpunten bestand in het gewenste formaat
goal_events_export_df = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/csv/doelpunte
goal_events_export_df = goal_events_export_df[goal_events_export_df['valid_goal'] == True
goal_events_export_df['stand_thuis'] = goal_events_export_df['stand_thuis'].astype(int)
goal_events_export_df = goal_events_export_df.drop(['valid_goal'], axis=1)
goal_events_export_df['real_time_goal'] = pd.to_datetime(goal_events_export_df['real_time
goal_events_export_df['real_time_goal'] = goal_events_export_df['real_time_goal'].dt.strf
goal_events_export_df['tijd'] = pd.to_datetime(goal_events_export_df['tijd'], format='%H:
goal_events_export_df['tijd'] = goal_events_export_df['tijd'].dt.strftime('%H:%M')
goal_events_export_df['datum'] = pd.to_datetime(goal_events_export_df['datum'], format='%
goal_events_export_df['datum'] = goal_events_export_df['datum'].dt.strftime('%Y-%m-%d')
goal_events_export_df.to_csv('/content/drive/MyDrive/Colab_Notebooks/csv/doelpunten.csv',

#Exporteer het wedstrijden bestand in het gewenste formaat
match_results_export_df = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/csv/wedstrijd
match_results_export_df['tijd'] = pd.to_datetime(match_results_export_df['tijd'], format='%
match_results_export_df['tijd'] = match_results_export_df['tijd'].dt.strftime('%H:%M')
match_results_export_df['datum'] = pd.to_datetime(match_results_export_df['datum'], format=
match_results_export_df['datum'] = match_results_export_df['datum'].dt.strftime('%Y-%m-%d')
match_results_export_df.to_csv('/content/drive/MyDrive/Colab_Notebooks/csv/wedstrijden.csv
```