In [ ]:
```
# WHAT WAS THE COUNT OF MATCHES IN EACH SEASON

import pandas as pd
df=pd.read_csv("D:\major_pr2\IPL Matches 2008-2020.csv")
df['year']=pd.DatetimeIndex(df['date']).year
df.groupby("year").size()
```

Out[ ]:
```
year
2008    58
2009    57
2010    60
2011    73
2012    74
2013    76
2014    60
2015    59
2016    60
2017    59
2018    60
2019    60
2020    60
dtype: int64
```

In [ ]:
```
# who has umpired the most

# Assuming you have columns like 'umpire1' and 'umpire2' for the two on-field umpir

# Combine the two umpire columns into one Series
umpires = pd.concat([df['umpire1'], df['umpire2']])

# Count the occurrences of each umpire
umpire_counts = umpires.value_counts()

# Find the umpire who has umpired the most matches
most_umpired = umpire_counts.idxmax()
matches_umpired = umpire_counts.max()

print(f"The umpire who has umpired the most matches is {most_umpired} with {matches
```

```
The umpire who has umpired the most matches is S Ravi with 121 matches.
```

In [ ]:
```
# WHAT TEAM HAS WON THE MOST TOSSES)

print("TEAM THAT HAS WON MAXIMUM TOSSES")
df.groupby('toss_winner').size().sort_values(ascending=False).head(1)
```

```
TEAM THAT HAS WON MAXIMUM TOSSES
```

Out[ ]:
```
toss_winner
Mumbai Indians    106
dtype: int64
```

In [ ]:
```
# WHAT DOES THE TEAM DECIDE AFTER WINNING THE TOSS


# Calculate the most common decision made by the team that won the toss
```

```
most_common_decision = df[df['toss_winner'] == df['winner']]['toss_decision'].mode(

print(f"The team that wins the toss decides to {most_common_decision}.")
```

```
The team that wins the toss decides to field.
```

In [ ]:
```
# how does the toss decision vary across the season



# Group the data by season and toss decision
seasonal_toss_decisions = df.groupby(['year', 'toss_decision']).size().unstack().fi

# Print the count of toss decisions for each season
for season, decisions in seasonal_toss_decisions.iterrows():
    bat_count = decisions['bat']
    field_count = decisions['field']
    print(f"Season {season}:")
    print(f"  Bat: {bat_count} matches")
    print(f"  Field: {field_count} matches")
    print()
```

```
Season 2008:
  Bat: 26 matches
  Field: 32 matches

Season 2009:
  Bat: 35 matches
  Field: 22 matches

Season 2010:
  Bat: 39 matches
  Field: 21 matches

Season 2011:
  Bat: 25 matches
  Field: 48 matches

Season 2012:
  Bat: 37 matches
  Field: 37 matches

Season 2013:
  Bat: 45 matches
  Field: 31 matches

Season 2014:
  Bat: 19 matches
  Field: 41 matches

Season 2015:
  Bat: 25 matches
  Field: 34 matches

Season 2016:
  Bat: 11 matches
  Field: 49 matches

Season 2017:
  Bat: 11 matches
  Field: 48 matches

Season 2018:
  Bat: 10 matches
  Field: 50 matches

Season 2019:
  Bat: 10 matches
  Field: 50 matches

Season 2020:
  Bat: 27 matches
  Field: 33 matches
```

In [ ]:
```
#  DOES WINNING THE TOSS IMPLY WINNING THE GAME

# Calculate the number of matches where the toss winner is also the match winner
```

```
matches_won_by_toss_winner = df[df['toss_winner'] == df['winner']]

# Calculate the total number of matches
total_matches = len(df)

# Calculate the percentage of matches won by the toss winner
percentage_won_by_toss_winner = len(matches_won_by_toss_winner) / total_matches * 1

print(f"Percentage of matches won by the toss winner: {percentage_won_by_toss_winne
```

```
Percentage of matches won by the toss winner: 51.23%
```

In [ ]:
```
#  HOW MANY TIMES HAS THE CHASING TEAM WON THE GAME

# Count the number of matches where the chasing team won
chasing_team_wins = len(df[df['toss_decision'] == 'field'][df['toss_winner'] == df[

print(f"The chasing team has won {chasing_team_wins} matches.")
```

```
The chasing team has won 273 matches.
C:\Users\SHOBAN RAJ N\AppData\Local\Temp\ipykernel_20040\1715185744.py:4: UserWarnin
g: Boolean Series key will be reindexed to match DataFrame index.
  chasing_team_wins = len(df[df['toss_decision'] == 'field'][df['toss_winner'] == df
['winner']])
```

In [ ]:
```
#  WHICH ALL THE TEAM HAS WON THE TOURNAMNET

# Group the data by season and find the winner of each season
season_winners = df.groupby('year')['winner'].tail(1).unique()

print("Teams that have won the IPL tournament:")
for winner in season_winners:
    print(winner)
```

```
Teams that have won the IPL tournament:
Rajasthan Royals
Deccan Chargers
Chennai Super Kings
Kolkata Knight Riders
Mumbai Indians
Sunrisers Hyderabad
```

In [ ]:
```
#  WHICH TEAM HAS PLAYED MOST NUMBER OF MATCHES

# Combine 'team1' and 'team2' columns to count appearances
team_appearances = pd.concat([df['team1'], df['team2']])
# Count the number of appearances for each team
team_appearance_counts = team_appearances.value_counts()

# Get the team with the most appearances
team_with_most_appearances = team_appearance_counts.idxmax()
most_appearances = team_appearance_counts.max()

print(f"The team that has played the most matches is {team_with_most_appearances} w
```

```
The team that has played the most matches is Mumbai Indians with 203 matches.
```

In [ ]:
```
#  WHICH TEAM HAS WON THE MOST NUMBER OF MATCHES

# Count the number of matches won by each team
match_wins = df['winner'].value_counts()

# Get the team with the most match wins
team_with_most_wins = match_wins.idxmax()
most_wins = match_wins.max()

print(f"The team that has won the most matches is {team_with_most_wins} with {most_
```

```
The team that has won the most matches is Mumbai Indians with 120 wins.
```

In [ ]:
```
# IS THERE ANY LUCKY VENUE FOR A PARTICULAR TEAM

# Create a pivot table to calculate the win rate for each team at each venue
venue_team_win_rate = pd.crosstab(index=df['venue'], columns=df['winner'], normaliz

# Identify if any team has a significantly high win rate at a specific venue
threshold = 70  # You can adjust the threshold as needed

lucky_venues = {}
for team in venue_team_win_rate.columns:
    for venue in venue_team_win_rate.index:
        if venue_team_win_rate.loc[venue, team] >= threshold:
            if team not in lucky_venues:
                lucky_venues[team] = []
            lucky_venues[team].append(venue)

if not lucky_venues:
    print("There are no particularly 'lucky' venues for any team.")
else:
    print("Lucky venues for each team:")
    for team, venues in lucky_venues.items():
        print(f"{team}: {', '.join(venues)}")
```

```
Lucky venues for each team:
Chennai Super Kings: MA Chidambaram Stadium, Chepauk
```

In [ ]:
```
#  INNINGS WISE COMPARISON BETWEEN TEAMS

# Choose the two teams you want to compare
team1 = 'Team1'  # Replace 'Team1' with the first team's name
team2 = 'Team2'  # Replace 'Team2' with the second team's name

# Filter matches where the two chosen teams played
team1_matches = df[(df['team1'] == team1) | (df['team2'] == team1)]
team2_matches = df[(df['team1'] == team2) | (df['team2'] == team2)]

# Count the number of matches won by each team in the 1st innings
team1_wins_1st_innings = len(team1_matches[team1_matches['toss_decision'] == 'bat']
team2_wins_1st_innings = len(team2_matches[team2_matches['toss_decision'] == 'bat']

# Count the number of matches won by each team in the 2nd innings
team1_wins_2nd_innings = len(team1_matches[team1_matches['toss_decision'] == 'field
team2_wins_2nd_innings = len(team2_matches[team2_matches['toss_decision'] == 'field
```

```
print(f"{team1} wins in the 1st innings: {team1_wins_1st_innings} matches")
print(f"{team2} wins in the 1st innings: {team2_wins_1st_innings} matches")
print(f"{team1} wins in the 2nd innings: {team1_wins_2nd_innings} matches")
print(f"{team2} wins in the 2nd innings: {team2_wins_2nd_innings} matches")
```

```
Team1 wins in the 1st innings: 0 matches
Team2 wins in the 1st innings: 0 matches
Team1 wins in the 2nd innings: 0 matches
Team2 wins in the 2nd innings: 0 matches
```

In [ ]:
```
#   which is the biggest win in terms of run margin

# Find the biggest win by run margin
biggest_win = df[df['result_margin'] == df['result_margin'].max()]

# Extract the relevant information
winning_team = biggest_win['winner'].values[0]
run_margin = biggest_win['result_margin'].values[0]

print(f"The biggest win by run margin was {run_margin} runs, achieved by {winning_t
```

```
The biggest win by run margin was 146.0 runs, achieved by Mumbai Indians.
```

In [ ]:
```
#  which batsman have played most number of balls

import pandas as pd

# Load your CSV file
file_path = "D:\major_pr2\IPL Ball-by-Ball 2008-2020.csv"
data = pd.read_csv(file_path)

# Group the data by batsman and sum the number of balls faced
batsman_balls_faced = data.groupby('batsman')['ball'].count().reset_index()

# Find the batsman who has played the most number of balls
most_balls_faced = batsman_balls_faced[batsman_balls_faced['ball'] == batsman_balls

most_balls_batsman = most_balls_faced['batsman'].values[0]
balls_faced = most_balls_faced['ball'].values[0]

print(f"The batsman who has played the most number of balls is {most_balls_batsman}
```

```
The batsman who has played the most number of balls is V Kohli with 4609 balls face
d.
```

In [ ]:
```
# which batsman have played most number of balls

import pandas as pd

# Load your CSV file
file_path = "D:\major_pr2\IPL Ball-by-Ball 2008-2020.csv"
data = pd.read_csv(file_path)

# Group the data by batsman and sum the number of balls faced
batsman_balls_faced = data.groupby('batsman')['ball'].count().reset_index()

# Find the batsman who has played the most number of balls
```

```
most_balls_faced = batsman_balls_faced[batsman_balls_faced['ball'] == batsman_balls

most_balls_batsman = most_balls_faced['batsman'].values[0]
balls_faced = most_balls_faced['ball'].values[0]

print(f"The batsman who has played the most number of balls is {most_balls_batsman}
```

```
The batsman who has played the most number of balls is V Kohli with 4609 balls face
d.
```

In [ ]:
```
# who has hit the most number of 4's

import pandas as pd

# Load your CSV file
file_path = "D:\major_pr2\IPL Ball-by-Ball 2008-2020.csv"
data = pd.read_csv(file_path)

# Filter the data to only include fours (4 runs)
fours = data[data['batsman_runs'] == 4]

# Group the data by batsman and count the number of fours hit by each batsman
fours_count = fours['batsman'].value_counts()

# Find the batsman with the most fours
most_fours_batsman = fours_count.idxmax()
fours_hit = fours_count.max()

print(f"The batsman who has hit the most number of fours is {most_fours_batsman} wi
```

```
The batsman who has hit the most number of fours is S Dhawan with 591 fours.
```

In [ ]:
```
#  who has hit the most number of 6's

import pandas as pd

# Load your CSV file
file_path = "D:\major_pr2\IPL Ball-by-Ball 2008-2020.csv"
data = pd.read_csv(file_path)

# Filter the data to only include sixes (6 runs)
sixes = data[data['batsman_runs'] == 6]

# Group the data by batsman and count the number of sixes hit by each batsman
sixes_count = sixes['batsman'].value_counts()

# Find the batsman with the most sixes
most_sixes_batsman = sixes_count.idxmax()
sixes_hit = sixes_count.max()

print(f"The batsman who has hit the most number of sixes is {most_sixes_batsman} wi
```

```
The batsman who has hit the most number of sixes is CH Gayle with 349 sixes.
```

In [ ]:
```
#  who has the highest strike rate

import pandas as pd
```

```
# Load your CSV file
file_path = "D:\major_pr2\IPL Ball-by-Ball 2008-2020.csv"
data = pd.read_csv(file_path)

# Group the data by batsman and calculate their total runs and balls faced
batsman_stats = data.groupby('batsman').agg({'batsman_runs': 'sum', 'ball': 'count'

# Calculate the strike rate for each batsman
batsman_stats['strike_rate'] = (batsman_stats['batsman_runs'] / batsman_stats['ball

# Find the batsman with the highest strike rate
highest_strike_rate_batsman = batsman_stats[batsman_stats['strike_rate'] == batsman

best_batsman = highest_strike_rate_batsman['batsman'].values[0]
highest_strike_rate = highest_strike_rate_batsman['strike_rate'].values[0]

print(f"The batsman with the highest strike rate is {best_batsman} with a strike ra
```

```
The batsman with the highest strike rate is B Stanlake with a strike rate of 250.00.
```

In [ ]:

```
#  who is  the leading wicket-taker

import pandas as pd

# Load your CSV file
file_path = "D:\major_pr2\IPL Ball-by-Ball 2008-2020.csv"
data = pd.read_csv(file_path)

# Group the data by bowler and calculate the total number of wickets taken by each
bowler_wickets = data.groupby('bowler')['player_dismissed'].count().reset_index()

# Rename the columns for clarity
bowler_wickets.columns = ['Bowler', 'Wickets']

# Find the bowler with the most wickets
leading_wicket_taker = bowler_wickets[bowler_wickets['Wickets'] == bowler_wickets['

leading_bowler = leading_wicket_taker['Bowler'].values[0]
most_wickets = leading_wicket_taker['Wickets'].values[0]

print(f"The leading wicket-taker is {leading_bowler} with {most_wickets} wickets.")
```

```
The leading wicket-taker is SL Malinga with 188 wickets.
```

In [ ]:

```
 #  which stadium has hosted the most number of matches

import pandas as pd

# Load your CSV file
file_path = "D:/major_pr2/IPL Matches 2008-2020.csv"
data = pd.read_csv(file_path)

# Count the number of matches hosted at each stadium
stadium_counts = data['venue'].value_counts()

# Find the stadium that has hosted the most matches
```

```
most_hosted_stadium = stadium_counts.idxmax()
matches_hosted = stadium_counts.max()

print(f"The stadium that has hosted the most number of matches is {most_hosted_stad
```

```
The stadium that has hosted the most number of matches is Eden Gardens with 77 match
es.
```