1. Top 10 batsmen based on past 3 years total runs scored.

[41]:
```
df_summary = df_fact_batting_summary.groupby('batsmanName').agg({'runs': 'sum'})

top_10 = df_summary.nlargest(10, 'runs')
print(format_dataframes(top_10))
```

```
+------------------+--------+
| batsmanName      |   runs |
+==================+========+
| Shubman Gill     |   1851 |
+------------------+--------+
| Fafdu Plessis    |   1831 |
+------------------+--------+
| Ruturaj Gaikwad  |   1593 |
+------------------+--------+
| KLRahul          |   1516 |
+------------------+--------+
| Jos Buttler      |   1509 |
+------------------+--------+
| Shikhar Dhawan   |   1392 |
+------------------+--------+
| Virat Kohli      |   1385 |
+------------------+--------+
| Sanju Samson     |   1304 |
+------------------+--------+
| Suryakumar Yadav |   1225 |
+------------------+--------+
| Glenn Maxwell    |   1214 |
+------------------+--------+
```

2. Top 10 batsmen based on past 3 years batting average. Minimum 60 balls faced in each season.

- Total Runs = SUM(fact_batting_summary[runs])
- Total Innings Dismissed = SUM(fact_batting_summary[out])
- Batting Avg = Total Runs/Total Innings Dismissed

[42]:
```
df_batting_avg = all_seasons.copy()

df_batting_avg["batting_avg"] = (np.where(df_batting_avg['out']!=0,␣
 ↪df_batting_avg['runs']/df_batting_avg['out'], 0)).round(2)
top_10 = df_batting_avg.nlargest(10, 'batting_avg')[["batting_avg"]]
print(format_dataframes(top_10))
```

```
+------------------+---------------+
| batsmanName      |   batting_avg |
+==================+===============+
| KLRahul          |         50.53 |
+------------------+---------------+
```

```
| Fafdu Plessis    |         43.6  |
+------------------+---------------+
| David Miller     |         43.2  |
+------------------+---------------+
| Jos Buttler      |         41.92 |
+------------------+---------------+
| Shimron Hetmyer  |         40.67 |
+------------------+---------------+
| Shubman Gill     |         40.24 |
+------------------+---------------+
| Shikhar Dhawan   |         39.77 |
+------------------+---------------+
| Ruturaj Gaikwad  |         37.93 |
+------------------+---------------+
| David Warner     |         37.9  |
+------------------+---------------+
| Suryakumar Yadav |         35    |
+------------------+---------------+
```

3. Top 10 batsmen based on past 3 years strike rate. Minimum 60 balls faced in each season. Strike Rate = (Total Runs/Total Balls Faced)*100

[43]:
```
df_batsman_SR = all_seasons.copy()

df_batsman_SR["strike_rate"] = ((df_batsman_SR['runs']/
  ↪df_batsman_SR['balls'])*100).round(2)
top_10 = df_batsman_SR.nlargest(10, 'strike_rate')[["strike_rate"]]
print(format_dataframes(top_10))
```

```
+------------------+---------------+
| batsmanName      |   strike_rate |
+==================+===============+
| Glenn Maxwell    |        161.44 |
+------------------+---------------+
| Suryakumar Yadav |        160.55 |
+------------------+---------------+
| Andre Russell    |        159.19 |
+------------------+---------------+
| Shimron Hetmyer  |        157.27 |
+------------------+---------------+
| Nicholas Pooran  |        157.11 |
+------------------+---------------+
| Prithvi Shaw     |        153.2  |
+------------------+---------------+
| Dinesh Karthik   |        152.64 |
+------------------+---------------+
| Yashasvi Jaiswal |        152.15 |
+------------------+---------------+
```

```
| Jos Buttler      |        146.93 |
+------------------+---------------+
| Shivam Dube      |        145.95 |
+------------------+---------------+
```

4. Top 10 bowlers based on past 3 years total wickets taken. Total Wickets = SUM(fact_bowling_summary[wickets])

[44]:
```
df_wickets = df_fact_bowling_summary.groupby(['bowlerName']).agg({'wickets':␣
↪'sum'})
top_10 = df_wickets.nlargest(10, 'wickets')
print(format_dataframes(top_10))
```

```
+--------------------+-----------+
| bowlerName         |   wickets |
+====================+===========+
| Mohammed Shami     |        67 |
+--------------------+-----------+
| Yuzvendra Chahal   |        66 |
+--------------------+-----------+
| Harshal Patel      |        65 |
+--------------------+-----------+
| Rashid Khan        |        63 |
+--------------------+-----------+
| Avesh Khan         |        47 |
+--------------------+-----------+
| Arshdeep Singh     |        45 |
+--------------------+-----------+
| Kagiso Rabada      |        45 |
+--------------------+-----------+
| Varun Chakravarthy |        44 |
+--------------------+-----------+
| Shardul Thakur     |        43 |
+--------------------+-----------+
| Trent Boult        |        42 |
+--------------------+-----------+
```

5. Top 10 bowlers based on past 3 years bowling average. Minimum 60 balls bowled in each season.

- Runs Conceded = SUM(fact_bowling_summary[wickets])
- Bowling Average = Runs Conceded/Wickets

[45]:
```
df_bowling_average = bowler_all_seasons.copy()

df_bowling_average["bowling_avg"] = (np.where(df_bowling_average['wickets']!=0,␣
↪df_bowling_average['runs']/df_bowling_average['wickets'], 0)).round(2)
top_10 = df_bowling_average.nsmallest(10, 'bowling_avg')[["bowling_avg"]]
```

```
print(format_dataframes(top_10))
```

```
+------------------+---------------+
| bowlerName       |   bowling_avg |
+==================+===============+
| Andre Russell    |         18.23 |
+------------------+---------------+
| Yuzvendra Chahal |         20.2  |
+------------------+---------------+
| Harshal Patel    |         20.35 |
+------------------+---------------+
| Rashid Khan      |         20.9  |
+------------------+---------------+
| Mohammed Shami   |         20.97 |
+------------------+---------------+
| Avesh Khan       |         23.72 |
+------------------+---------------+
| Kagiso Rabada    |         23.76 |
+------------------+---------------+
| Moeen Ali        |         23.86 |
+------------------+---------------+
| Anrich Nortje    |         24.77 |
+------------------+---------------+
| Umran Malik      |         26.1  |
+------------------+---------------+
```

6. Top 10 bowlers based on past 3 years economy rate. Minimum 60 balls bowled in each season.

- Balls Bowled = SUM(fact_bowling_summary[balls])
- Runs Conceded = SUM(fact_bowling_summary[runs])
- Economy = Runs Conceded/(balls Bowled/6)

[46]:
```
df_economy = bowler_all_seasons.copy()

df_economy["economy"] = (df_economy['runs']/(df_economy['balls']/6)).round(2)
top_10 = df_economy.nsmallest(10, 'economy')[["economy"]]
print(format_dataframes(top_10))
```

```
+--------------------+-----------+
| bowlerName         |   economy |
+====================+===========+
| Sunil Narine       |      6.6  |
+--------------------+-----------+
| Moeen Ali          |      7.04 |
+--------------------+-----------+
| Axar Patel         |      7.11 |
+--------------------+-----------+
| Rashid Khan        |      7.2  |
```

```
+---------------------+-----------+
| Krunal Pandya       |      7.45 |
+---------------------+-----------+
| Ravindra Jadeja     |      7.46 |
+---------------------+-----------+
| Ravichandran Ashwin |      7.5  |
+---------------------+-----------+
| Varun Chakravarthy  |      7.57 |
+---------------------+-----------+
| Harpreet Brar       |      7.6  |
+---------------------+-----------+
| Rahul Chahar        |      7.63 |
+---------------------+-----------+
```

7. Top 5 batsmen based on past 3 years boundary percentage. Minimum 60 balls faced in each season. Boundary % = SUM(fact_batting_summary[Boundary runs])/SUM(fact_batting_summary[Total Runs])

[47]:
```
df_boundary_perc = all_seasons.copy()
df_boundary_perc["boundary%"] = ((df_boundary_perc["boundary_runs"]/
  ↪df_boundary_perc["runs"])*100).round(2)
top_5 = df_boundary_perc.nlargest(5, 'boundary%')[["boundary%"]]
print(format_dataframes(top_5))
```

```
+------------------+-------------+
| batsmanName      |   boundary% |
+==================+=============+
| Andre Russell    |       75.7  |
+------------------+-------------+
| Yashasvi Jaiswal |       74.56 |
+------------------+-------------+
| Prithvi Shaw     |       70.67 |
+------------------+-------------+
| Jos Buttler      |       68.92 |
+------------------+-------------+
| Glenn Maxwell    |       68.7  |
+------------------+-------------+
```

8. Top 5 bowlers based on past 3 years dot ball percentage. Minimum 60 balls bowled in each season. Dot ball % = SUM(fact_bowling_summary[zeros])/SUM(fact_bowling_summary[balls])

[48]:
```
df_dotball_perc = bowler_all_seasons.copy()
df_dotball_perc["dotball%"] = ((df_dotball_perc["0s"]/
  ↪df_dotball_perc["balls"])*100).round(2)
top_5 = df_dotball_perc.nlargest(5, 'dotball%')[["dotball%"]]
print(format_dataframes(top_5))
```

```
+----------------+------------+
| bowlerName     |   dotball% |
+================+============+
| Mohammed Siraj |      47.71 |
+----------------+------------+
| Mohammed Shami |      47.57 |
+----------------+------------+
| Trent Boult    |      46.37 |
+----------------+------------+
| Umran Malik    |      44.15 |
+----------------+------------+
| Khaleel Ahmed  |      43.61 |
+----------------+------------+
```

9. Top 4 teams based on past 3 years winning %.

- Total Winnings = Count match_id from 'df_fact_match_summary' if (teams == winners)
- Total Matches = Count match_id from 'df_fact_match_summary'
- Winning Percentage = (Total Winnings/Total Matches)*100

[49]:
```
# Total matches played by each team
df_total_matches = df_fact_match_summary.groupby('teams').agg({'match_id':
 ↪'count'})
df_total_matches = df_total_matches.rename(columns={'match_id':'total_matches'})

# Total winnings of each team
df_total_winnings =␣
 ↪df_fact_match_summary[df_fact_match_summary["teams"]==df_fact_match_summary["winner"]].
 ↪groupby('teams').agg({'match_id':'count'})
df_total_winnings = df_total_winnings.rename(columns={'match_id':
 ↪'total_winnings'})

# Merge df_total_matches and df_total_winnings on the column 'teams'. Find␣
 ↪Winning%
df_winning_perc = pd.merge(df_total_winnings, df_total_matches, on='teams',␣
 ↪how='inner')
df_winning_perc["winning%"] = ((df_winning_perc["total_winnings"]/
 ↪df_winning_perc["total_matches"])*100).round(2)
top_4 = df_winning_perc.nlargest(4, 'winning%')[["winning%"]]
print(format_dataframes(top_4))
```

```
+--------------+------------+
| teams        |   winning% |
+==============+============+
| Titans       |      69.7  |
+--------------+------------+
| Super Giants |      58.62 |
+--------------+------------+
```

```
| RCB           |      55.56 |
+---------------+------------+
| Super Kings   |      55.56 |
+---------------+------------+
```

10. Top 2 teams with the highest number of wins achieved by chasing targets over the past 3 years..

[50]:
```
df_wins_achieved =␣
 ↪df_fact_match_summary[(df_fact_match_summary["team2"]==df_fact_match_summary["winner"])␣
 ↪&
                  ␣
 ↪(df_fact_match_summary["teams"]==df_fact_match_summary["winner"])].
 ↪groupby('teams').agg({'match_id':'count'})
df_wins_achieved = df_wins_achieved.rename(columns={'match_id':
 ↪'wins_achieved_by_chasing_targets'})

top_2 = df_wins_achieved.nlargest(2,␣
 ↪'wins_achieved_by_chasing_targets')[["wins_achieved_by_chasing_targets"]]

print(format_dataframes(top_2))
```

```
+----------+-------------------------------------+
| teams    |   wins_achieved_by_chasing_targets  |
+==========+=====================================+
| Capitals |                                  14 |
+----------+-------------------------------------+
| KKR      |                                  14 |
+----------+-------------------------------------+
```

[ ]:
```

```