Name :Pranav Shinde

PRN No.202401050029

EDS ASSIGNMENT NO.01

CRICKET WORLD CUP DATASET

1. Problem Statement: Find the top 5 run scorers of the tournament.

```
top_scorers = df.groupby('Player')
['Runs'].sum().sort_values(ascending=Fal
se).head(5)
```

2. Problem Statement: Identify the player with the best batting strike rate (minimum 100 runs scored).

```
best_striker = df[df['Runs'] >=
100].sort_values('Strike_Rate',
ascending=False).head(1)
```

3. Problem Statement: Calculate the total number of sixes hit in the tournament.

```
total_sixes = df['Sixes'].sum()
```

4. Problem Statement: Find the bowler with the most wickets.

```
top_bowler = df.groupby('Player')
['Wickets'].sum().sort_values(ascending=
False).head(1)
```

5. Problem Statement: Calculate the average runs scored per match for each team.

```
team_avg_runs = df.groupby('Team')
['Runs'].mean()
```

6. Problem Statement: Determine the player with the most catches.

```
most_catches = df.groupby('Player')
['Catches'].sum().sort_values(ascending=
False).head(1)
```

7. Problem Statement: Find the total number of wickets taken by each team.

```
team_wickets = df.groupby('Team')
['Wickets'].sum()
```

8. Problem Statement: Identify players who scored centuries (100+ runs in a match).

```
century_players = df[df['Runs'] >= 100]
['Player'].unique()
```

9. Problem Statement: List matches where the economy rate was less than 4.0.

```
economical_bowling =
df[df['Economy_Rate'] < 4.0]
```

10. Problem Statement: Find the top 5 players with the highest number of boundaries (fours + sixes).

```
df['Boundaries'] = df['Fours'] +
df['Sixes']
top_boundaries = df.groupby('Player')
['Boundaries'].sum().sort_values(ascendi
ng=False).head(5)
```

11. Problem Statement: Find the number of maiden overs bowled by each bowler.

```
maidens_by_bowler = df.groupby('Player')
['Maidens'].sum()
```

12. Problem Statement: Identify which venue hosted the most matches.

```
venue_counts =
df['Venue'].value_counts().head(1)
```

13. Problem Statement: Find out the match with the highest team score.

```
match_scores = df.groupby(['Team',
'Date'])
['Runs'].sum().sort_values(ascending=Fal
se).head(1)
```

14. Problem Statement: List players who have bowled more than 50 overs in the tournament.

```
total_overs_bowled =
df.groupby('Player')
['Overs_Bowled'].sum()
players_50_overs =
total_overs_bowled[total_overs_bowled >
50]
```

15. Problem Statement: Find the average number of runs conceded per wicket.

```
df['Runs_per_Wicket'] =
df['Runs_Conceded'] /
df['Wickets'].replace(0, np.nan)
avg_runs_per_wicket =
df['Runs_per_Wicket'].mean()
```

16. Problem Statement: Which batsman has faced the most number of balls?

```
balls_faced = df.groupby('Player')
['Balls_Faced'].sum().sort_values(ascend
ing=False).head(1)
```

17. Problem Statement: Calculate team-wise strike rates (total runs / total balls faced).

```
team_strike_rate = (df.groupby('Team')
['Runs'].sum() / df.groupby('Team')
['Balls_Faced'].sum()) * 100
```

18. Problem Statement: Identify the top 5 all-rounders (at least 200 runs + 10 wickets).

```
all_rounders =
df.groupby('Player').agg({'Runs':'sum',
'Wickets':'sum'})
top_all_rounders =
all_rounders[(all_rounders['Runs'] >=
200) & (all_rounders['Wickets'] >=
10)].sort_values(['Runs', 'Wickets'],
ascending=False).head(5)
```

19. Problem Statement: Find out how many players participated in the tournament.

```
total_players = df['Player'].nunique()
```

20. Problem Statement: Find the player who contributed highest % of team runs in a single match.

```
df['Match_Team_Runs'] =
df.groupby(['Team', 'Date'])
['Runs'].transform('sum')
df['Run_Contribution'] = (df['Runs'] /
df['Match_Team_Runs']) * 100
top_contributor =
df.sort_values('Run_Contribution',
ascending=False).head(1)
```