NAME- Amruta Kanase.

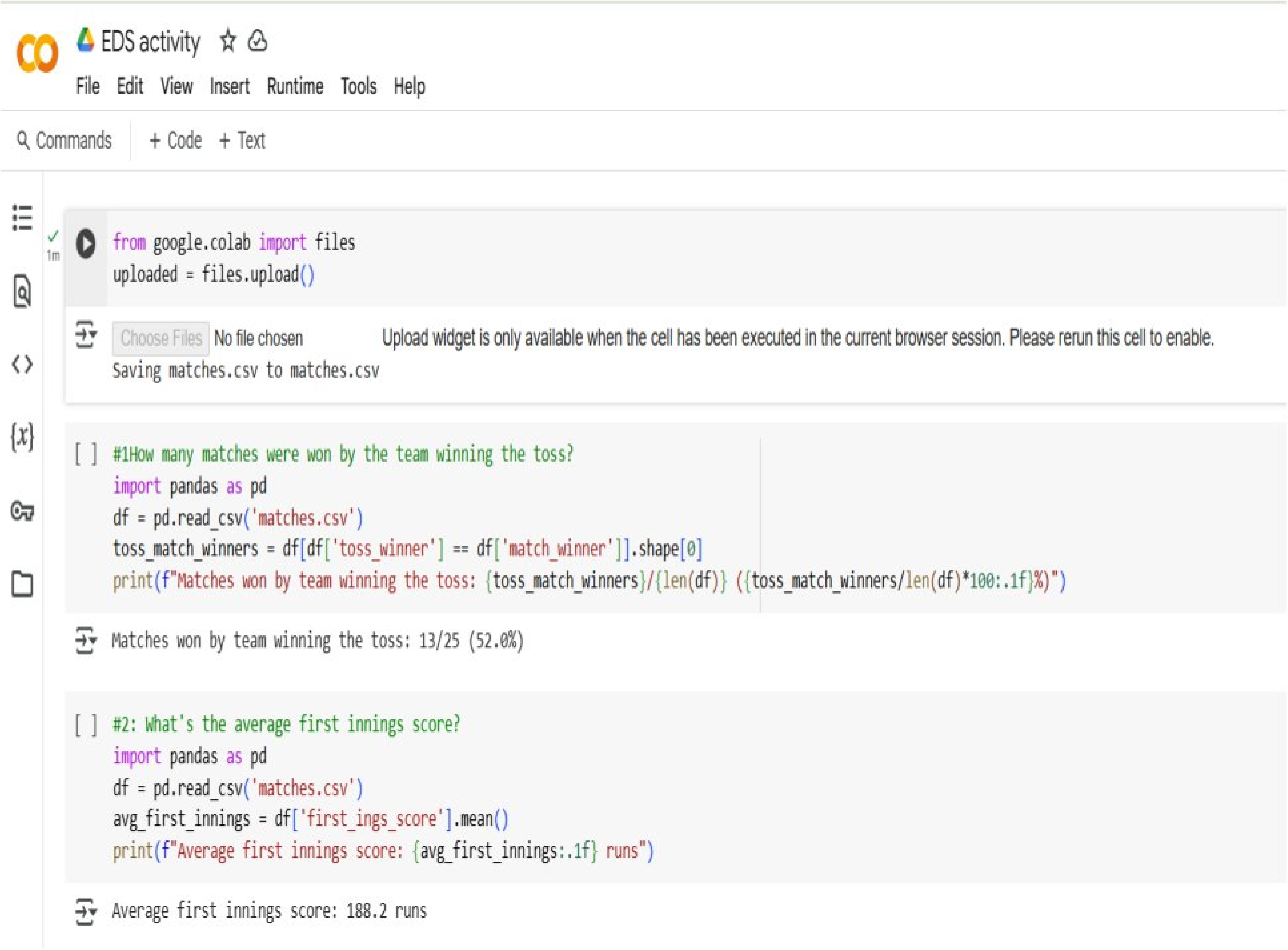
Roll no-CS7-52

PRN-202401110042

Div – CS7

DATASET - IPL

Google colab link -
https://colab.research.google.com/drive/1rdI6fCO3k2DtTRbgI3Mj5QQKxIBLcPm3?usp=sharing



```
from google.colab import files
uploaded = files.upload()
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

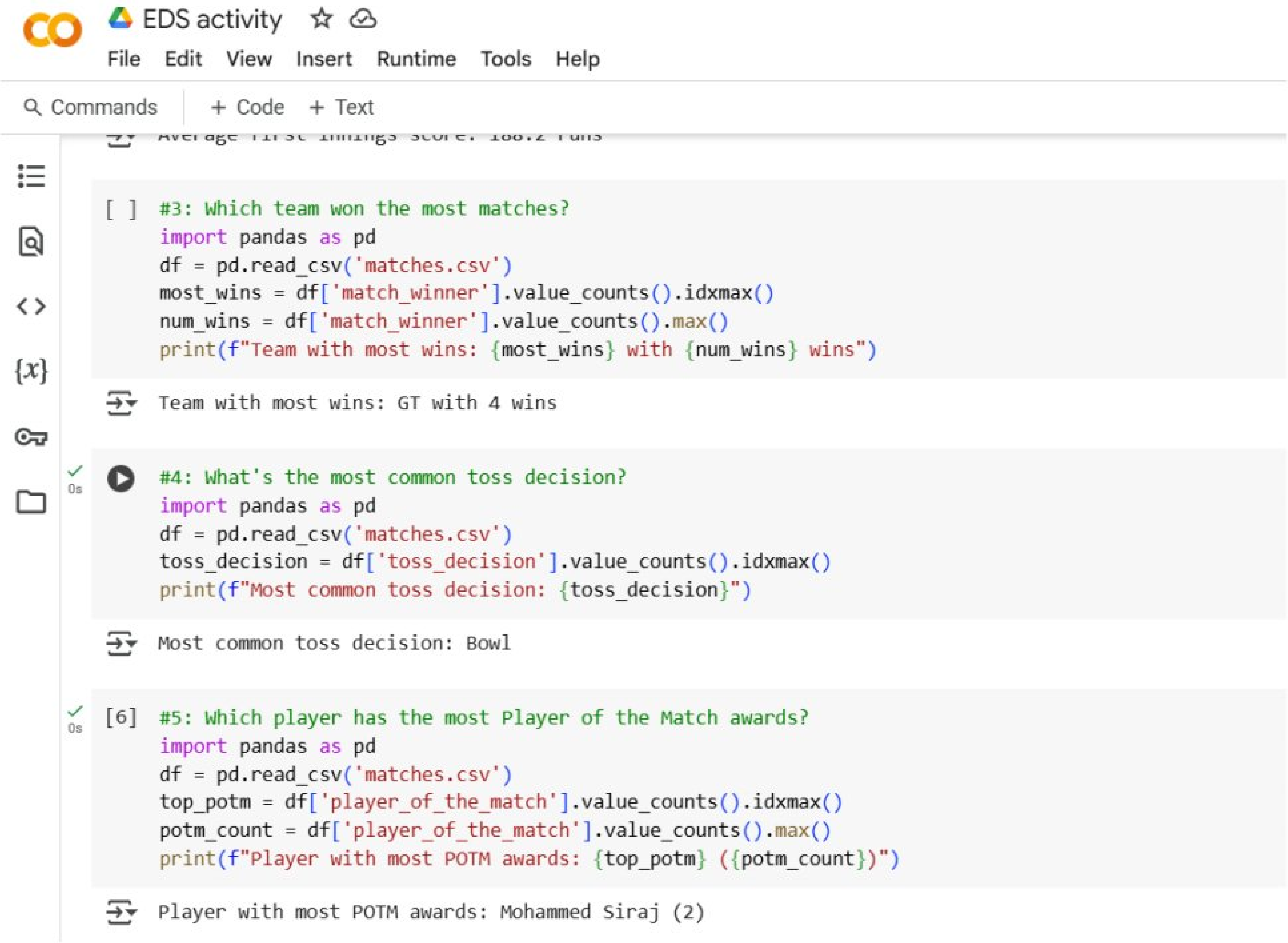
```
Saving matches.csv to matches.csv
```

```
#1How many matches were won by the team winning the toss?
import pandas as pd
df = pd.read_csv('matches.csv')
toss_match_winners = df[df['toss_winner'] == df['match_winner']].shape[0]
print(f"Matches won by team winning the toss: {toss_match_winners}/{len(df)} ({toss_match_winners/len(df)*100:.1f}%)")
```

```
Matches won by team winning the toss: 13/25 (52.0%)
```

```
#2: What's the average first innings score?
import pandas as pd
df = pd.read_csv('matches.csv')
avg_first_innings = df['first_ings_score'].mean()
print(f"Average first innings score: {avg_first_innings:.1f} runs")
```

```
Average first innings score: 188.2 runs
```



```
Average first innings score: 188.2 runs
```

```
[ ] #3: Which team won the most matches?
    import pandas as pd
    df = pd.read_csv('matches.csv')
    most_wins = df['match_winner'].value_counts().idxmax()
    num_wins = df['match_winner'].value_counts().max()
    print(f"Team with most wins: {most_wins} with {num_wins} wins")
```

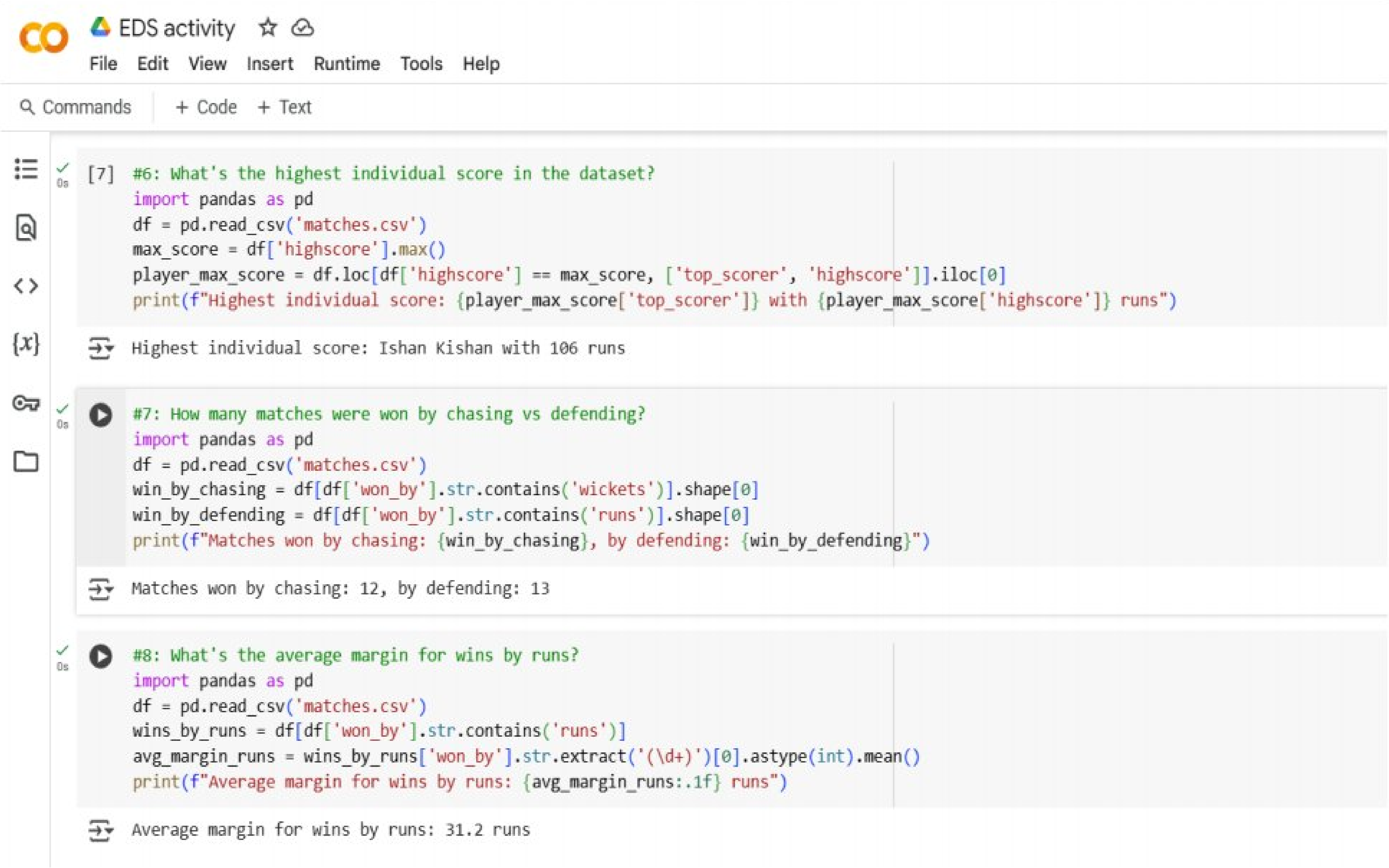
```
Team with most wins: GT with 4 wins
```

```
#4: What's the most common toss decision?
import pandas as pd
df = pd.read_csv('matches.csv')
toss_decision = df['toss_decision'].value_counts().idxmax()
print(f"Most common toss decision: {toss_decision}")
```

```
Most common toss decision: Bowl
```

```
[6] #5: Which player has the most Player of the Match awards?
    import pandas as pd
    df = pd.read_csv('matches.csv')
    top_potm = df['player_of_the_match'].value_counts().idxmax()
    potm_count = df['player_of_the_match'].value_counts().max()
    print(f"Player with most POTM awards: {top_potm} ({potm_count})")
```

```
Player with most POTM awards: Mohammed Siraj (2)
```



```
[7] #6: What's the highest individual score in the dataset?
    import pandas as pd
    df = pd.read_csv('matches.csv')
    max_score = df['highscore'].max()
    player_max_score = df.loc[df['highscore'] == max_score, ['top_scorer', 'highscore']].iloc[0]
    print(f"Highest individual score: {player_max_score['top_scorer']} with {player_max_score['highscore']} runs")
```

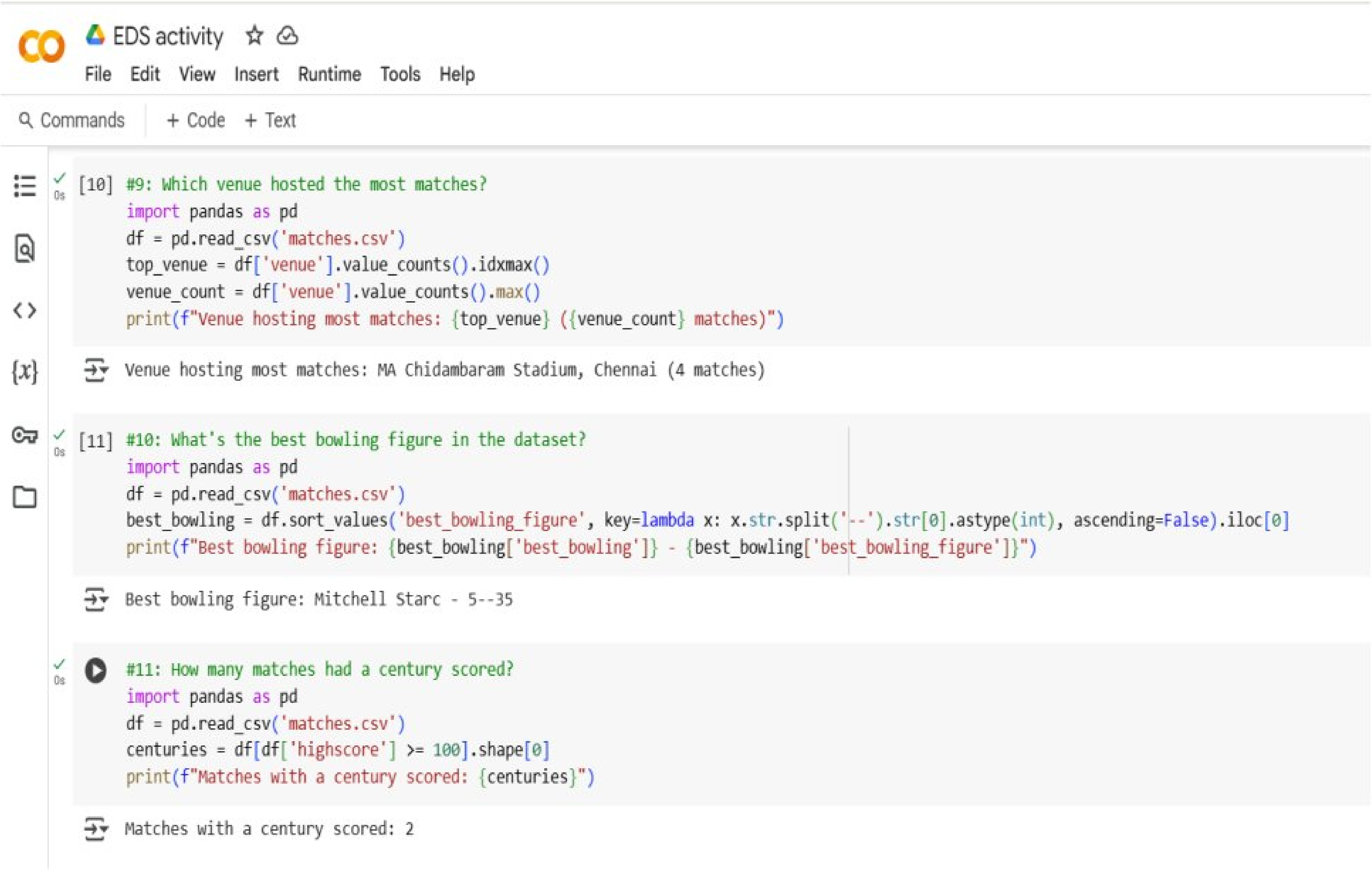
```
Highest individual score: Ishan Kishan with 106 runs
```

```
#7: How many matches were won by chasing vs defending?
import pandas as pd
df = pd.read_csv('matches.csv')
win_by_chasing = df[df['won_by'].str.contains('wickets')].shape[0]
win_by_defending = df[df['won_by'].str.contains('runs')].shape[0]
print(f"Matches won by chasing: {win_by_chasing}, by defending: {win_by_defending}")
```

```
Matches won by chasing: 12, by defending: 13
```

```
#8: What's the average margin for wins by runs?
import pandas as pd
df = pd.read_csv('matches.csv')
wins_by_runs = df[df['won_by'].str.contains('runs')]
avg_margin_runs = wins_by_runs['won_by'].str.extract('(\d+)')[0].astype(int).mean()
print(f"Average margin for wins by runs: {avg_margin_runs:.1f} runs")
```

```
Average margin for wins by runs: 31.2 runs
```



```
[10] #9: Which venue hosted the most matches?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     top_venue = df['venue'].value_counts().idxmax()
     venue_count = df['venue'].value_counts().max()
     print(f"Venue hosting most matches: {top_venue} ({venue_count} matches)")
```

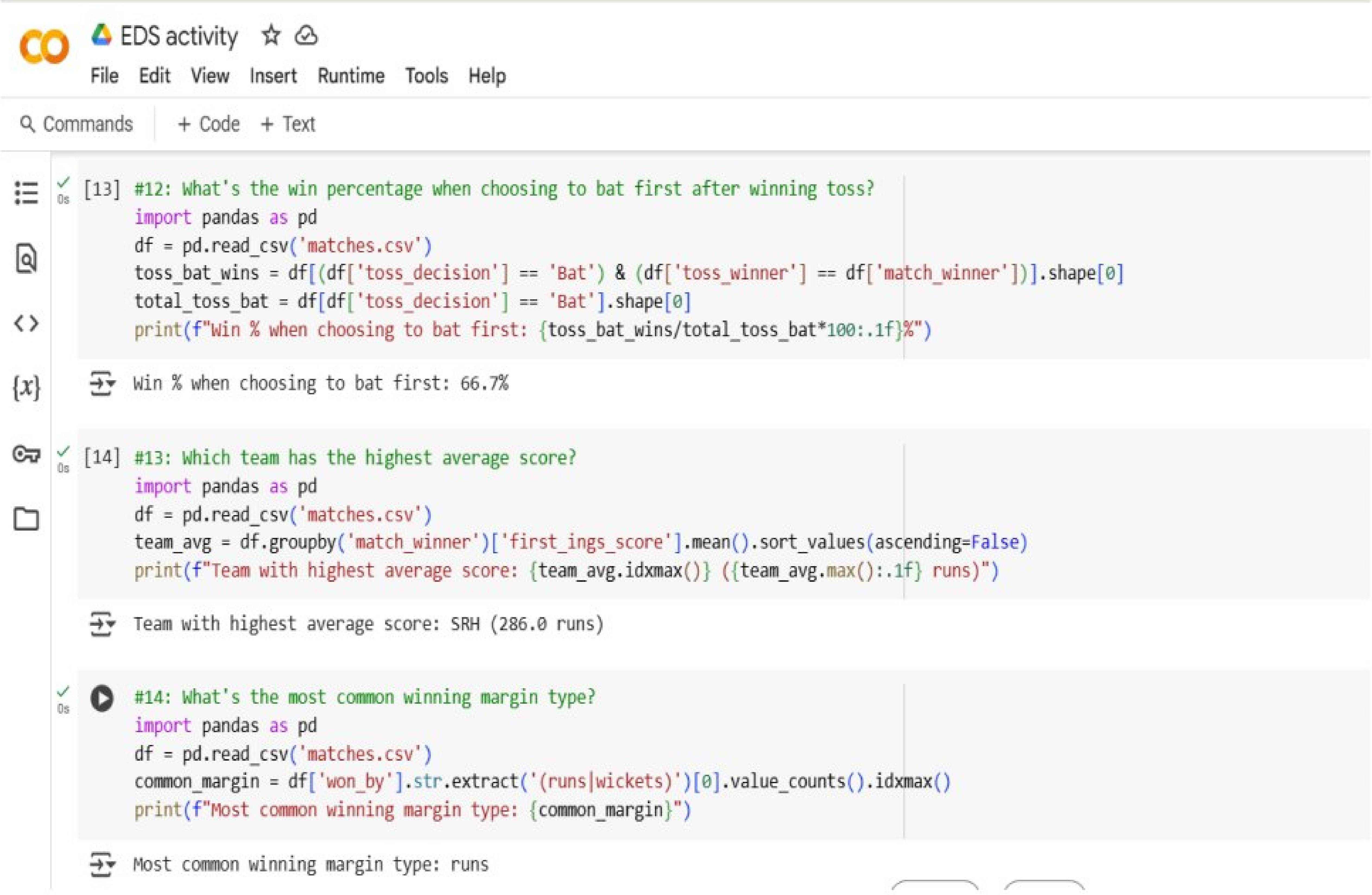
```
Venue hosting most matches: MA Chidambaram Stadium, Chennai (4 matches)
```

```
[11] #10: What's the best bowling figure in the dataset?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     best_bowling = df.sort_values('best_bowling_figure', key=lambda x: x.str.split('--').str[0].astype(int), ascending=False).iloc[0]
     print(f"Best bowling figure: {best_bowling['best_bowling']} - {best_bowling['best_bowling_figure']}")
```

```
Best bowling figure: Mitchell Starc - 5--35
```

```
#11: How many matches had a century scored?
import pandas as pd
df = pd.read_csv('matches.csv')
centuries = df[df['highscore'] >= 100].shape[0]
print(f"Matches with a century scored: {centuries}")
```

```
Matches with a century scored: 2
```



```
[13] #12: What's the win percentage when choosing to bat first after winning toss?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     toss_bat_wins = df[(df['toss_decision'] == 'Bat') & (df['toss_winner'] == df['match_winner'])].shape[0]
     total_toss_bat = df[df['toss_decision'] == 'Bat'].shape[0]
     print(f"Win % when choosing to bat first: {toss_bat_wins/total_toss_bat*100:.1f}%")
```

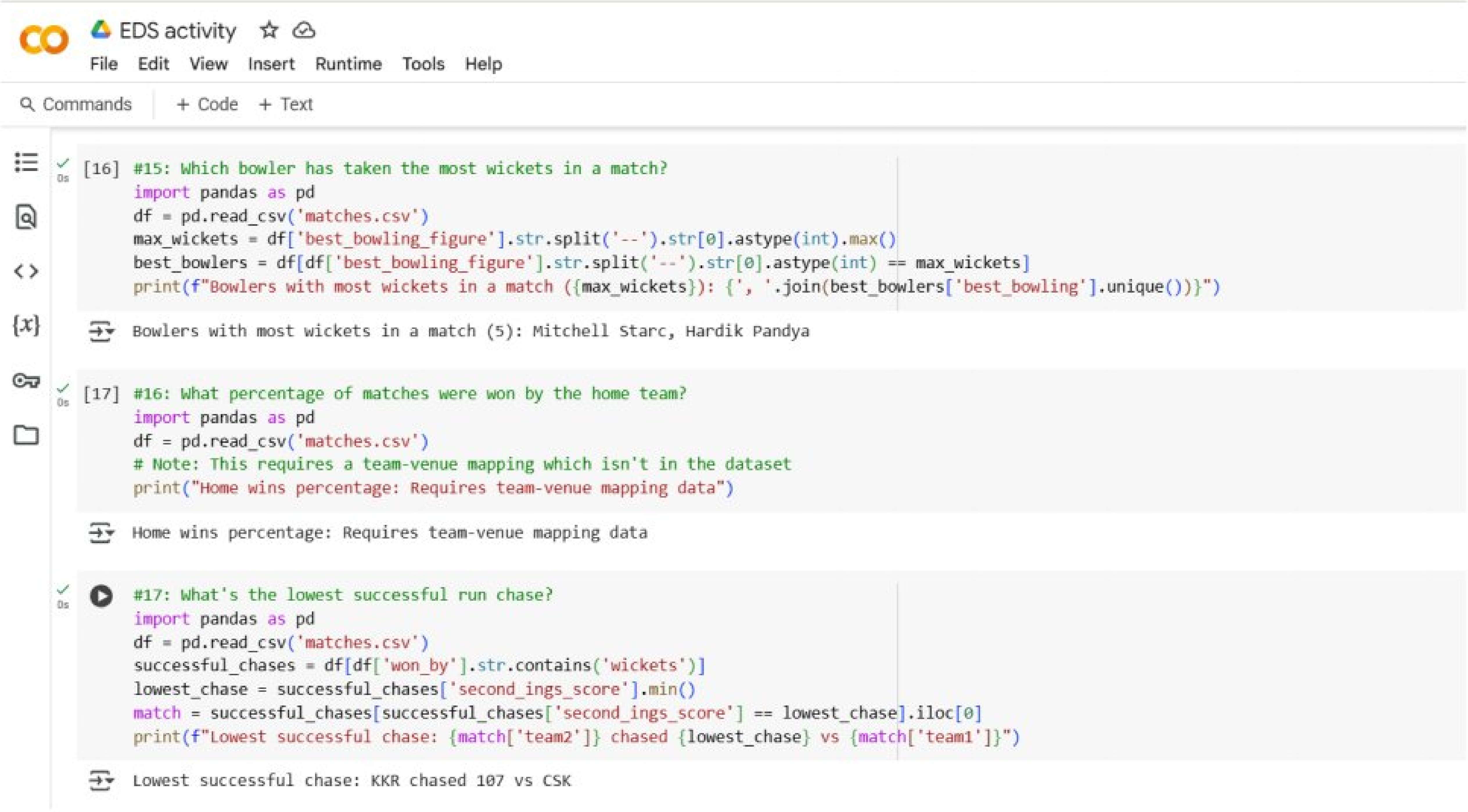
```
Win % when choosing to bat first: 66.7%
```

```
[14] #13: Which team has the highest average score?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     team_avg = df.groupby('match_winner')['first_ings_score'].mean().sort_values(ascending=False)
     print(f"Team with highest average score: {team_avg.idxmax()} ({team_avg.max():.1f} runs)")
```

```
Team with highest average score: SRH (286.0 runs)
```

```
#14: What's the most common winning margin type?
import pandas as pd
df = pd.read_csv('matches.csv')
common_margin = df['won_by'].str.extract('(runs|wickets)')[0].value_counts().idxmax()
print(f"Most common winning margin type: {common_margin}")
```

```
Most common winning margin type: runs
```



```
[16] #15: Which bowler has taken the most wickets in a match?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     max_wickets = df['best_bowling_figure'].str.split('--').str[0].astype(int).max()
     best_bowlers = df[df['best_bowling_figure'].str.split('--').str[0].astype(int) == max_wickets]
     print(f"Bowlers with most wickets in a match ({max_wickets}): {', '.join(best_bowlers['best_bowling'].unique())}")
```

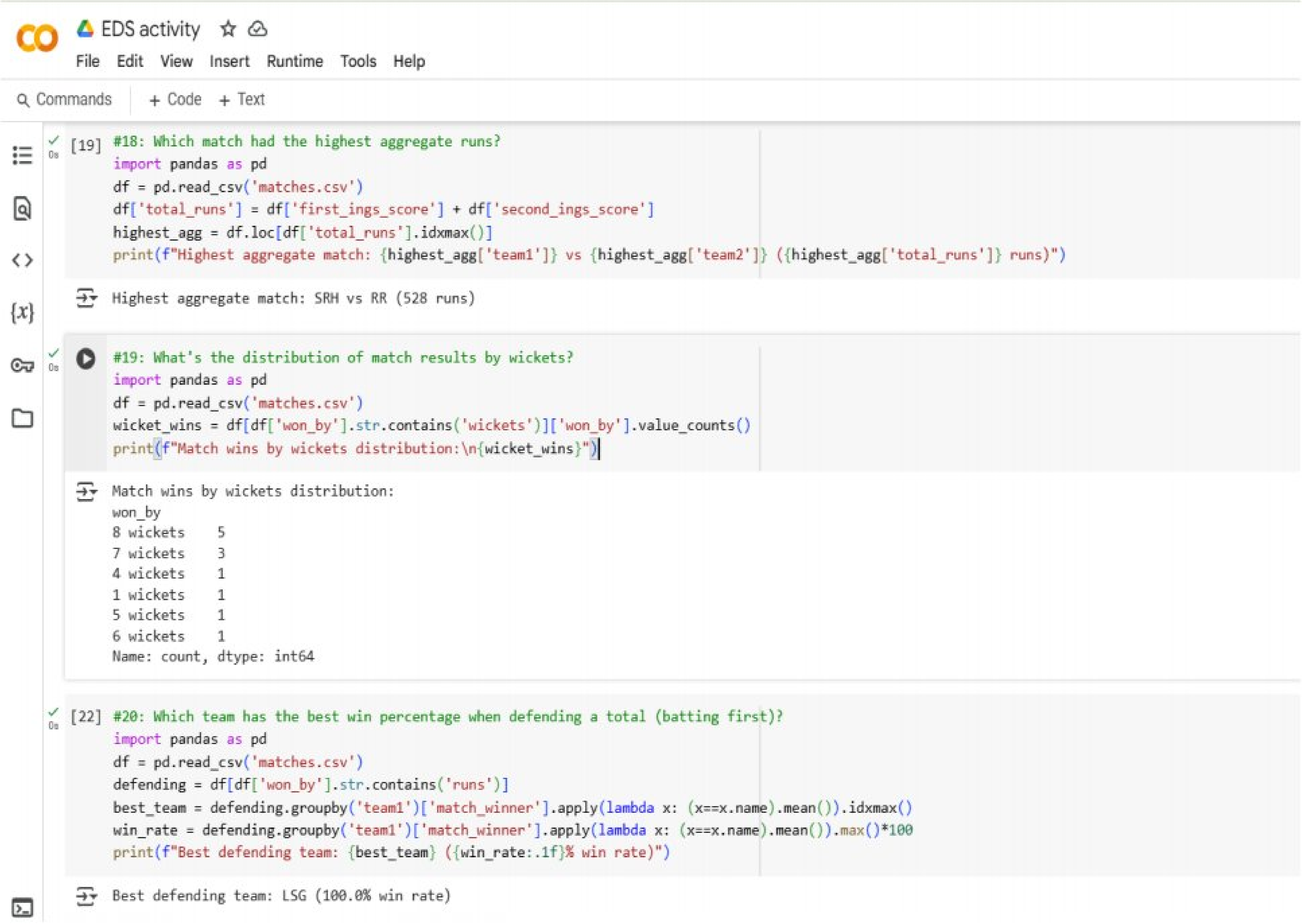
```
Bowlers with most wickets in a match (5): Mitchell Starc, Hardik Pandya
```

```
[17] #16: What percentage of matches were won by the home team?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     # Note: This requires a team-venue mapping which isn't in the dataset
     print("Home wins percentage: Requires team-venue mapping data")
```

```
Home wins percentage: Requires team-venue mapping data
```

```
#17: What's the lowest successful run chase?
import pandas as pd
df = pd.read_csv('matches.csv')
successful_chases = df[df['won_by'].str.contains('wickets')]
lowest_chase = successful_chases['second_ings_score'].min()
match = successful_chases[successful_chases['second_ings_score'] == lowest_chase].iloc[0]
print(f"Lowest successful chase: {match['team2']} chased {lowest_chase} vs {match['team1']}")
```

```
Lowest successful chase: KKR chased 107 vs CSK
```



```
[19] #18: Which match had the highest aggregate runs?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     df['total_runs'] = df['first_ings_score'] + df['second_ings_score']
     highest_agg = df.loc[df['total_runs'].idxmax()]
     print(f"Highest aggregate match: {highest_agg['team1']} vs {highest_agg['team2']} ({highest_agg['total_runs']} runs)")
```

```
Highest aggregate match: SRH vs RR (528 runs)
```

```
#19: What's the distribution of match results by wickets?
import pandas as pd
df = pd.read_csv('matches.csv')
wicket_wins = df[df['won_by'].str.contains('wickets')]['won_by'].value_counts()
print(f"Match wins by wickets distribution:\n{wicket_wins}")
```

```
Match wins by wickets distribution:
won_by
8 wickets    5
7 wickets    3
4 wickets    1
1 wickets    1
5 wickets    1
6 wickets    1
Name: count, dtype: int64
```

```
[22] #20: Which team has the best win percentage when defending a total (batting first)?
     import pandas as pd
     df = pd.read_csv('matches.csv')
     defending = df[df['won_by'].str.contains('runs')]
     best_team = defending.groupby('team1')['match_winner'].apply(lambda x: (x==x.name).mean()).idxmax()
     win_rate = defending.groupby('team1')['match_winner'].apply(lambda x: (x==x.name).mean()).max()*100
     print(f"Best defending team: {best_team} ({win_rate:.1f}% win rate)")
```

```
Best defending team: LSG (100.0% win rate)
```