SQL FINAL PROJECT

- by Shriyam Chatterjee

CREATING NEW TABLES AND IMPORTING DATA

1. create table ipl_player(id int , inning int , over int , ball int , batsman varchar , non_striker varchar ,
bowler varchar , batsman_runs int ,
extra_runs int , total_runs int ,is_wicket int ,dismissal_kind varchar , player_dismissed varchar , fielder varchar , extras_type varchar , batting_team varchar ,
bowling_team varchar);

2. create table ipl_match_data (id int, city varchar , match_date date ,player_of_match varchar , venue varchar , neutral_venue varchar ,team1 varchar ,
team2 varchar , toss_winner varchar , toss_decision varchar , winner varchar , result varchar , result_margin int ,
eliminator varchar , method varchar , umpire1 varchar , umpire2 varchar);

DATA IMPORT :

```
copy ipl_player from 'c:\program files\postgresql\16\data\data_copy\ipl
dataset\ipl_ball.csv'
delimiter ',' csv header ;
```

```
copy ipl_match_data from 'c:\program files\postgresql\16\data\data_copy\ipl
dataset\ipl_matches.csv'
delimiter ',' csv header ;
```

ANALYSIS 1 :

1. create table sr as (select batsman,
count (ball) as ball_faced,
sum(total_runs) as runs_scored,
sum(case when extras_type = 'wides' then 1 else 0 end) as wides_count,
sum(case when extras_type <> 'wides' then 1 else 0 end) as actual_ball_faced
from ipl_player
group by batsman
order by runs_scored desc);

2.select * from sr;

3. create table stat as (select * , ball_faced - wides_count as actual_ball_faced,
runs_scored - wides_count as actual_runs_scored from sr);

4. select * from stat;

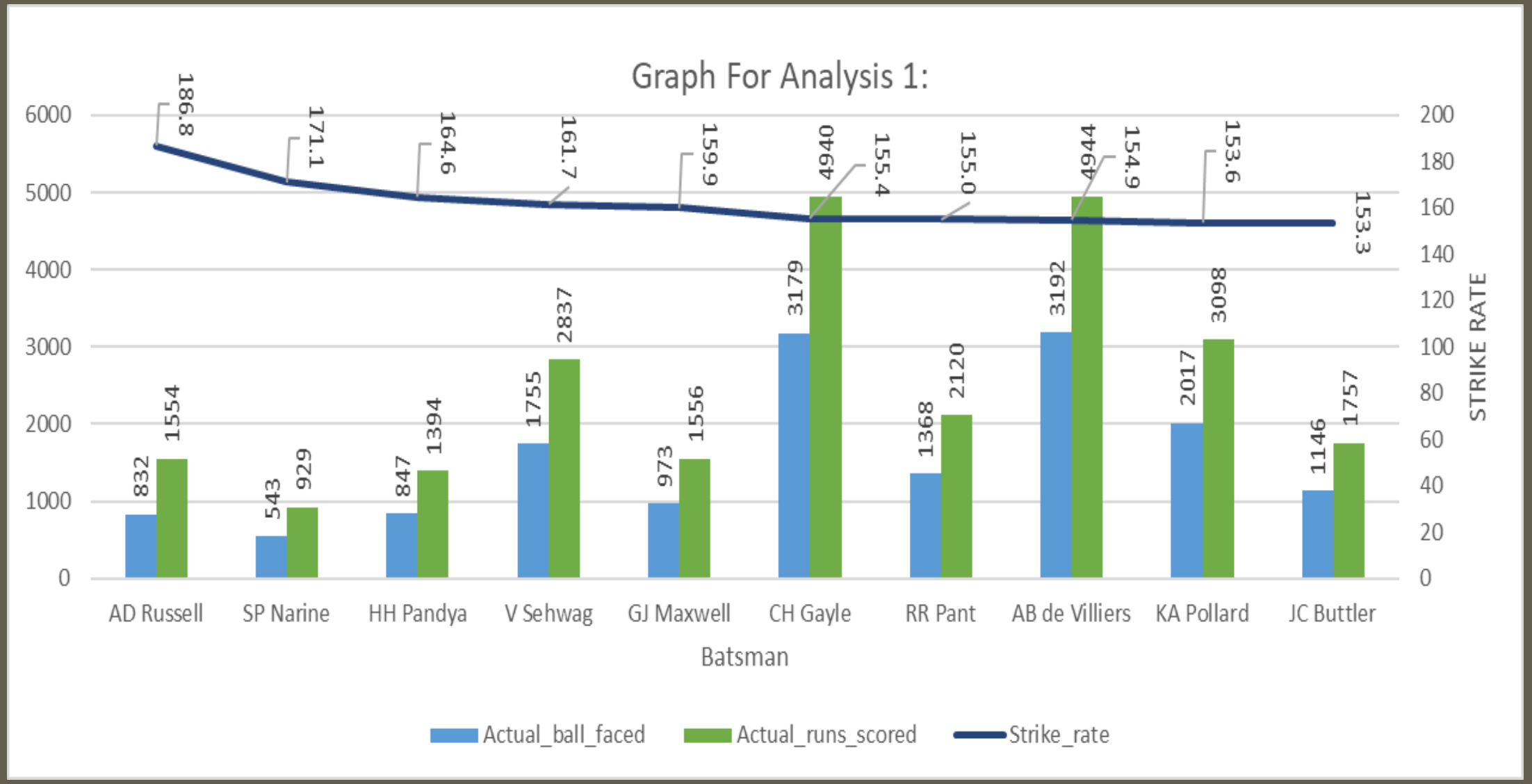
5. alter table stat
alter column actual_ball_faced type decimal ;

6.alter table stat
alter column actual_runs_scored type decimal ;

7. select * , (actual_runs_scored/actual_ball_faced)*100 as strike_rate
from stat where actual_ball_faced >= 500 order by strike_rate desc limit 10;

OUTPUT OF ANALYSIS 1:

Batsman	Ball_faced	Runs_score	Wides_count	Other_extras_count	Actual_ball_faced	Actual_runs_scored	Strike_rate
AD Russell	882	1604	50	832	832	1554	186.7788462
SP Narine	573	959	30	543	543	929	171.0865562
HH Pandya	897	1444	50	847	847	1394	164.5808737
V Sehwag	1833	2915	78	1755	1755	2837	161.6524217
GJ Maxwell	1013	1596	40	973	973	1556	159.9177801
CH Gayle	3342	5103	163	3179	3179	4940	155.3947782
RR Pant	1416	2168	48	1368	1368	2120	154.9707602
AB de Villiers	3264	5016	72	3192	3192	4944	154.887218
KA Pollard	2107	3188	90	2017	2017	3098	153.5944472
JC Buttler	1184	1795	38	1146	1146	1757	153.3158813


Graph For Analysis 1:
6000
5000
4000
3000
2000
1000
0
200
180
160
140
120
100
80
60
40
20
0
STRIKE RATE
188.8
171.1
164.6
161.7
159.9
155.4
155.0
154.9
153.6
153.3
832
1554
543
929
847
1394
1755
2837
973
1556
3179
4940
1368
2120
3192
4944
2017
3098
1146
1757
AD Russell
SP Narine
HH Pandya
V Sehwag
GJ Maxwell
CH Gayle
RR Pant
AB de Villiers
KA Pollard
JC Buttler
Batsman
Actual_ball_faced
Actual_runs_scored
Strike_rate

Analysis 2 :

```
1. select * from ipl_player;

2. select * from ipl_match_data;

3. create table player as ( select player_of_match,id,match_date from
   ipl_match_data group by player_of_match ,
      id, match_date order by player_of_match);

4. select * from player ;

5. create table player_year as(select * , extract(year from match_date) as year from
   player);

6. select * from player_year;

7. select player_of_match , max(year) as recent_season ,
      min(year) as oldest_season from player_year group by player_of_match order
   by player_of_match  ;

8. create table season as (select player_of_match , max(year) as recent_season ,
      min(year) as oldest_season from player_year group by player_of_match order
   by player_of_match ) ;
9. select * from season ;
10. create table sp as (select * , (recent_season - oldest_season)+1 as season_played
   from season);
11. create table dc as(select batsman ,sum(total_runs) as runs_scored,
   SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) AS dismissal_count,
    SUM(CASE WHEN extras_type = 'wides' THEN 1 ELSE 0 END) AS wides_count
   from ipl_player group by batsman order by runs_scored desc );
12. create table  dci as(select * , runs_scored-wides_count as actual_runs_scored
   from dc);
13. create table avg_players as (select *,actual_runs_scored/dismissal_count as
   average from dci
      where dismissal_count <> 0 order by average desc) ;
14. select g.batsman,g.actual_runs_scored,g.average,sp.season_played from
   avg_players
      as g inner join sp on g.batsman=sp.player_of_match where
   sp.season_played>2 order by g.average desc limit 10;
15. /*other types of join in analysis 2 */
select g.batsman,g.actual_runs_scored,g.average,sp.season_played from avg_players
      as g left join sp on g.batsman=sp.player_of_match where sp.season_played>2
   order by g.average desc limit 10;
```

Analysis 3:

```
1. create table bd as(select batsman , sum(total_runs) as
   total_runs_scored,
SUM(CASE WHEN batsman_runs = 4 THEN 1 ELSE 0 END) AS fours_count,
         SUM(CASE WHEN batsman_runs = 6 THEN 1 ELSE 0 END)
   AS six_count from ipl_player group by batsman order by
total_runs_scored desc);

2. select * from bd ;
3. ALTER TABLE bd
   ALTER COLUMN total_runs_scored  TYPE decimal;
4. ALTER TABLE bd
   ALTER COLUMN fours_count  TYPE decimal;

5. ALTER TABLE bd
   ALTER COLUMN six_count  TYPE decimal ;

6. create table boundary_percent as (select * , (fours_count*4 +
   six_count *6 ) as boundary_runs ,
   ((fours_count*4 + six_count *6 )/total_runs_scored)*100 as
boundary_percentage
   from bd  where total_runs_scored >0 )  ;

7. create table boundary_percent_sorted as ( select * from
   boundary_percent
   where total_runs_scored >1000 order by boundary_percentage desc);
8. select * from boundary_percent;
9. select * from boundary_percent_sorted ;

10. select bound.* , sp.season_played from boundary_percent_sorted as
   bound
         left join sp on bound.batsman = sp.player_of_match where
         sp.season_played is not null and  sp.season_played>2
         and bound.total_runs_scored >1000 order by
   bound.boundary_percentage desc limit 10 ;
```

OUTPUT OF ANALYSIS 2:

Batsman	Actual_runs_score	Average	Season_played
LH Ferguson	45	45	4
KL Rahul	2700	43	3
ML Hayden	1172	43	3
AB de Villiers	4944	43	12
CH Gayle	4940	42	12
DA Warner	5415	42	10
JP Duminy	2087	42	7
KS Williamson	1652	40	4
LMP Simmons	1101	40	4
MEK Hussey	2028	39	7

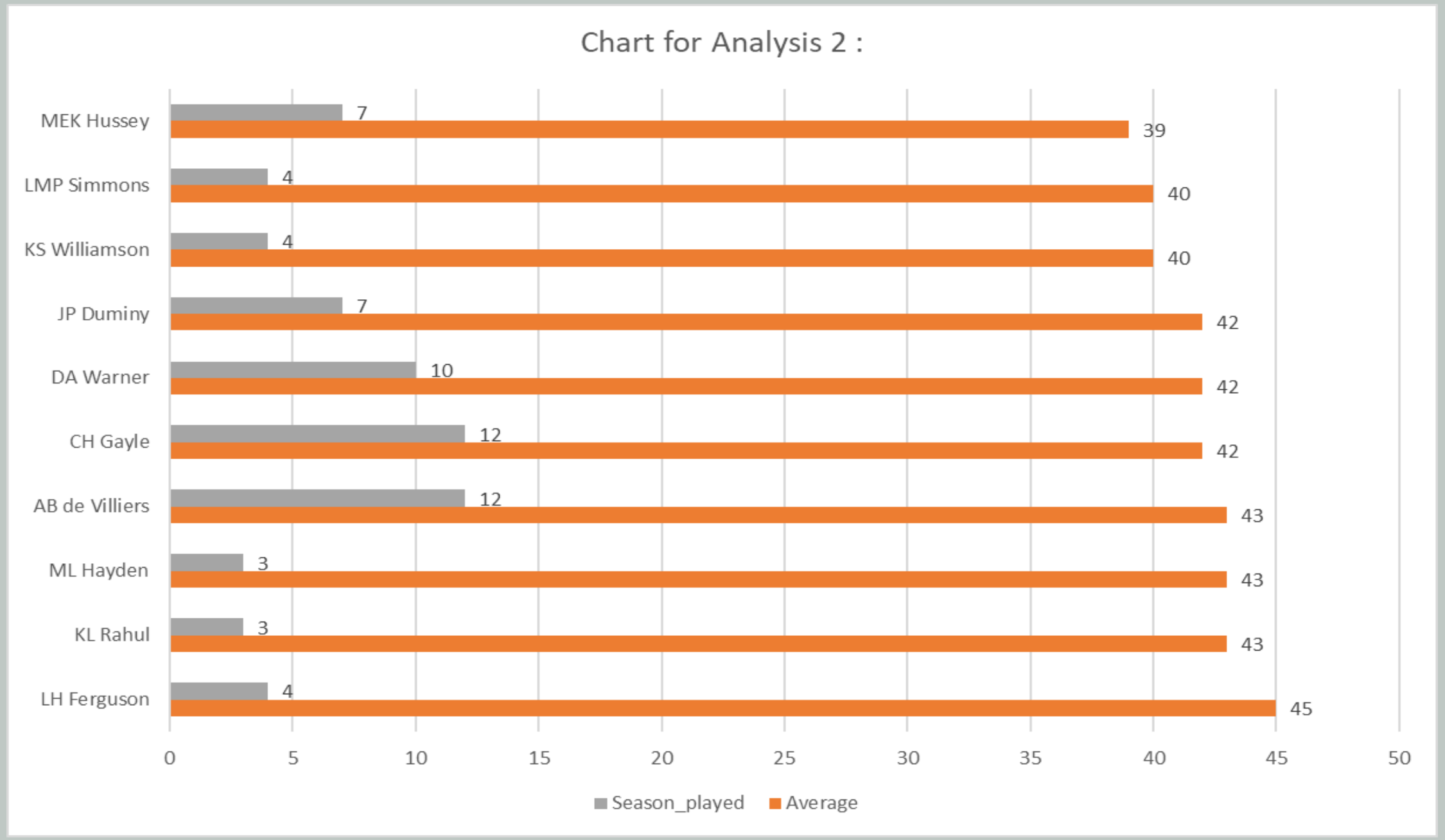
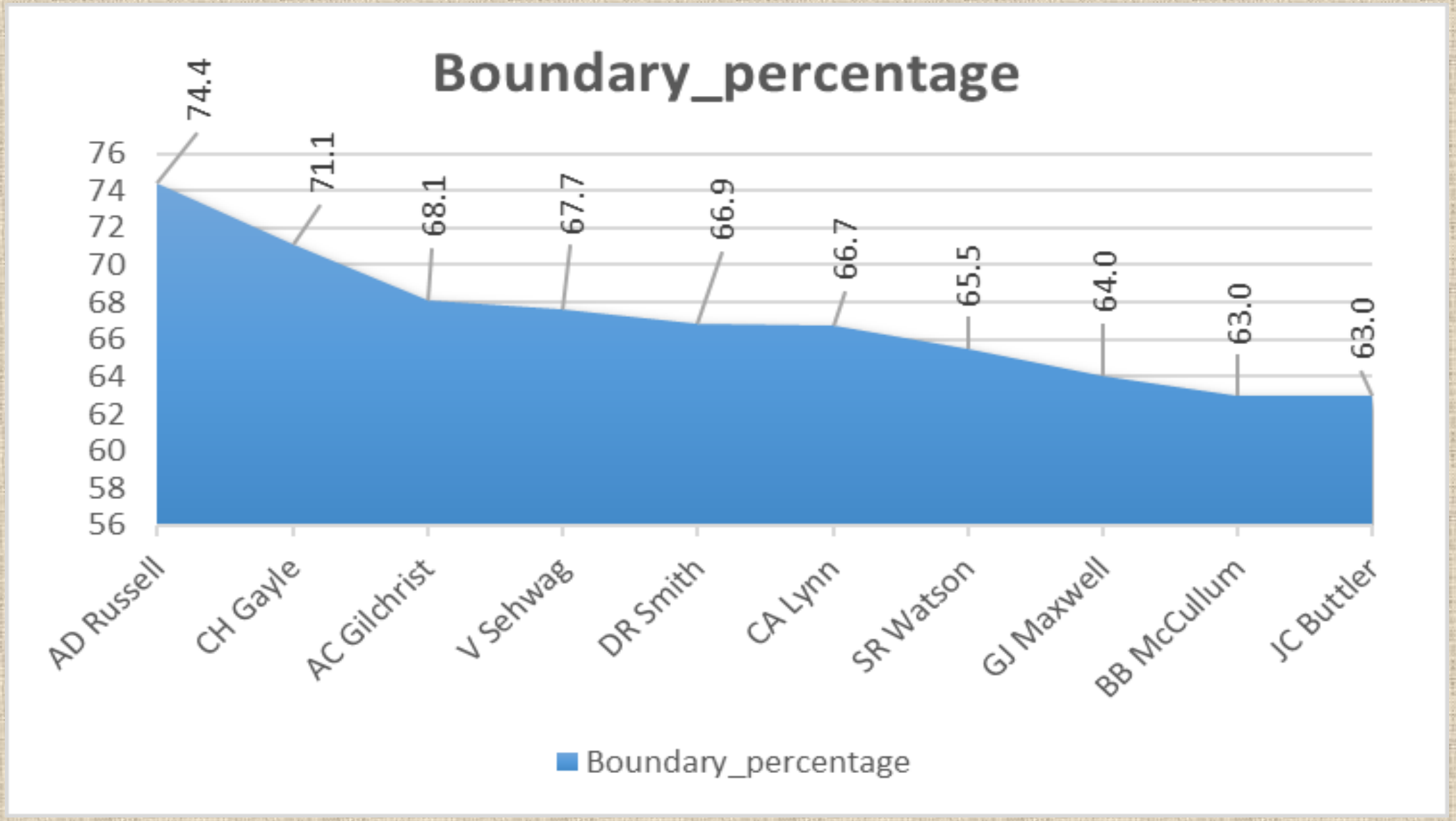


Chart for Analysis 2 :
MEK Hussey
LMP Simmons
KS Williamson
JP Duminy
DA Warner
CH Gayle
AB de Villiers
ML Hayden
KL Rahul
LH Ferguson
7
39
4
40
4
40
7
42
10
42
12
42
12
43
3
43
3
43
4
45
0
5
10
15
20
25
30
35
40
45
50
Season_played
Average

OUTPUT OF ANALYSIS 3:

Batsman	Total_runs_scored	Fours_count	Six_count	Boundary_runs	Boundary_percentage	Season_played
AD Russell	1604	105	129	1194	74.43890274	5
CH Gayle	5103	384	349	3630	71.13462669	12
AC Gilchrist	2215	239	92	1508	68.08126411	6
V Sehwag	2915	334	106	1972	67.65008576	7
DR Smith	2515	245	117	1682	66.87872763	9
CA Lynn	1334	128	63	890	66.71664168	5
SR Watson	4036	376	190	2644	65.51040634	13
GJ Maxwell	1596	119	91	1022	64.03508772	4
BB McCullum	3099	293	130	1952	62.98806066	8
JC Buttler	1795	167	77	1130	62.95264624	4


Boundary_percentage
74.4
71.1
68.1
67.7
66.9
66.7
65.5
64.0
63.0
63.0
76
74
72
70
68
66
64
62
60
58
56
AD Russell
CH Gayle
AC Gilchrist
V Sehwag
DR Smith
CA Lynn
SR Watson
GJ Maxwell
BB McCullum
JC Buttler
Boundary_percentage

Analysis 4:

1. select * from ipl_player;

2. create table bowlers as (select bowler , count(ball) as balls_bowled ,
   count(ball)/6 as overs_count, sum(total_runs)
   as runs_conceded from ipl_player group by bowler order by balls_bowled desc);
3. select * from bowlers;

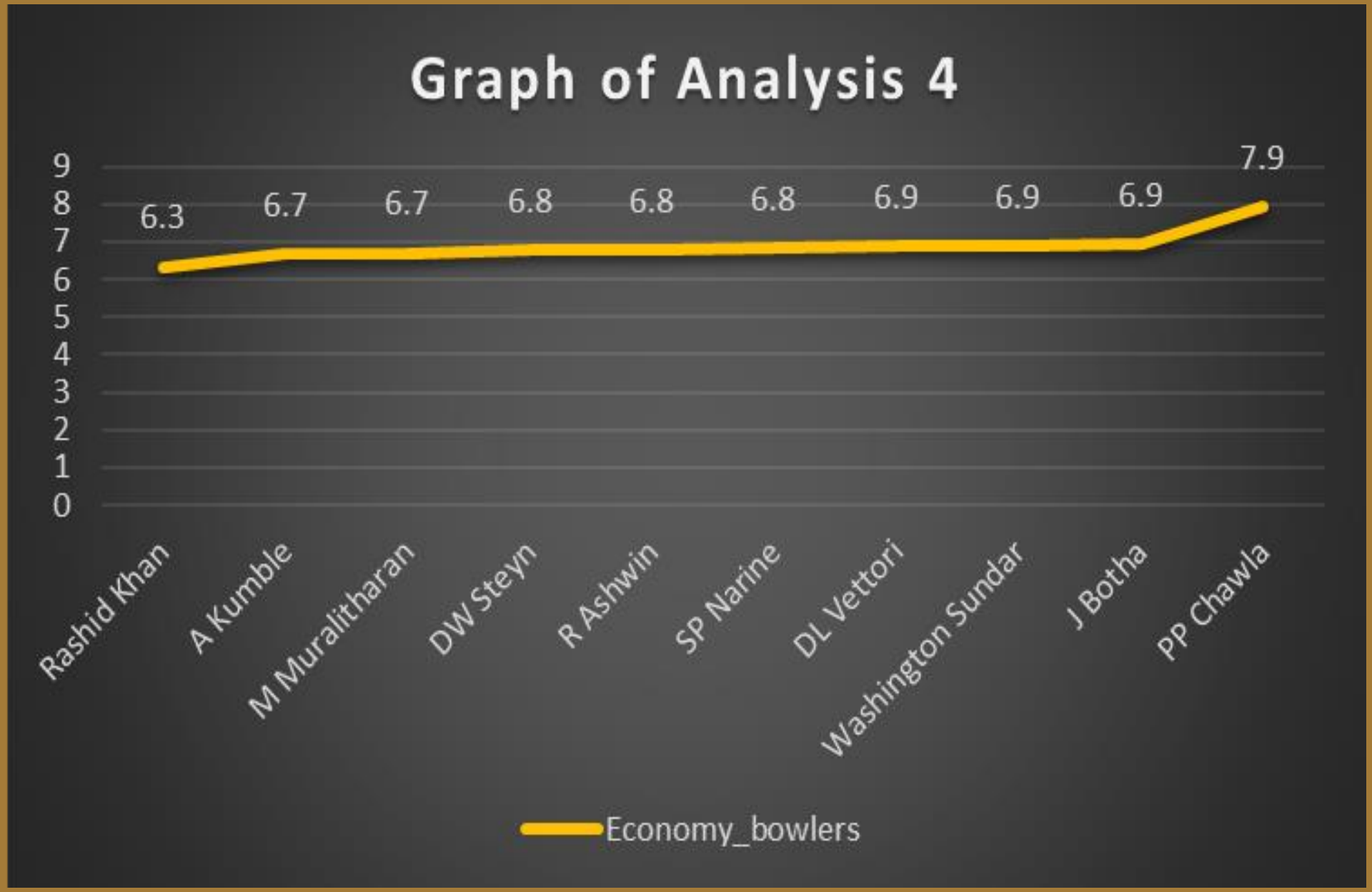
4. create table economy as (select *, (runs_conceded/overs_count) as economy
   from bowlers where overs_count>0);
5. select * from economy where balls_bowled >500 or balls_bowled = 500 order by economy limit 10;

ANALYSIS 4 (contd.) :

```
create table finaloutput4 as (select * from economy where
balls_bowled >500 or balls_bowled = 500 order by economy limit 10);

select * from finaloutput4 ;

alter table finaloutput4 drop economy ;

alter table finaloutput4 alter column runs_conceded type decimal;

alter table finaloutput4 alter column overs_count type decimal;


create table  output4 as ( select * , cast (runs_conceded/overs_count as decimal ) as economy_bowlers
from finaloutput4 order by economy_bowlers ) ;

SELECT *,
rank() OVER (
ORDER BY economy_bowlers ) as Rank_bowlers
FROM output4 ;
```

OUTPUT OF ANALYSIS 4:

Bowler	Balls_bowle	Overs_cour	Runs_concede	Economy_bowle	Rank_bowle
Rashid Khan	1490	248	1573	6.342741935	1
A Kumble	983	163	1089	6.680981595	2
M Muralitharan	1577	262	1755	6.698473282	3
DW Steyn	2276	379	2568	6.775725594	4
R Ashwin	3327	554	3756	6.779783394	5
SP Narine	2824	470	3208	6.825531915	6
DL Vettori	785	130	894	6.876923077	7
Washington Sundar	660	110	758	6.890909091	8
J Botha	709	118	818	6.93220339	9
PP Chawla	3285	547	4330	7.915904936	10


Graph of Analysis 4
9
8
7
6
5
4
3
2
1
0
6.3
6.7
6.7
6.8
6.8
6.8
6.9
6.9
6.9
7.9
Rashid Khan
A Kumble
M Muralitharan
DW Steyn
R Ashwin
SP Narine
DL Vettori
Washington Sundar
J Botha
PP Chawla
Economy_bowlers

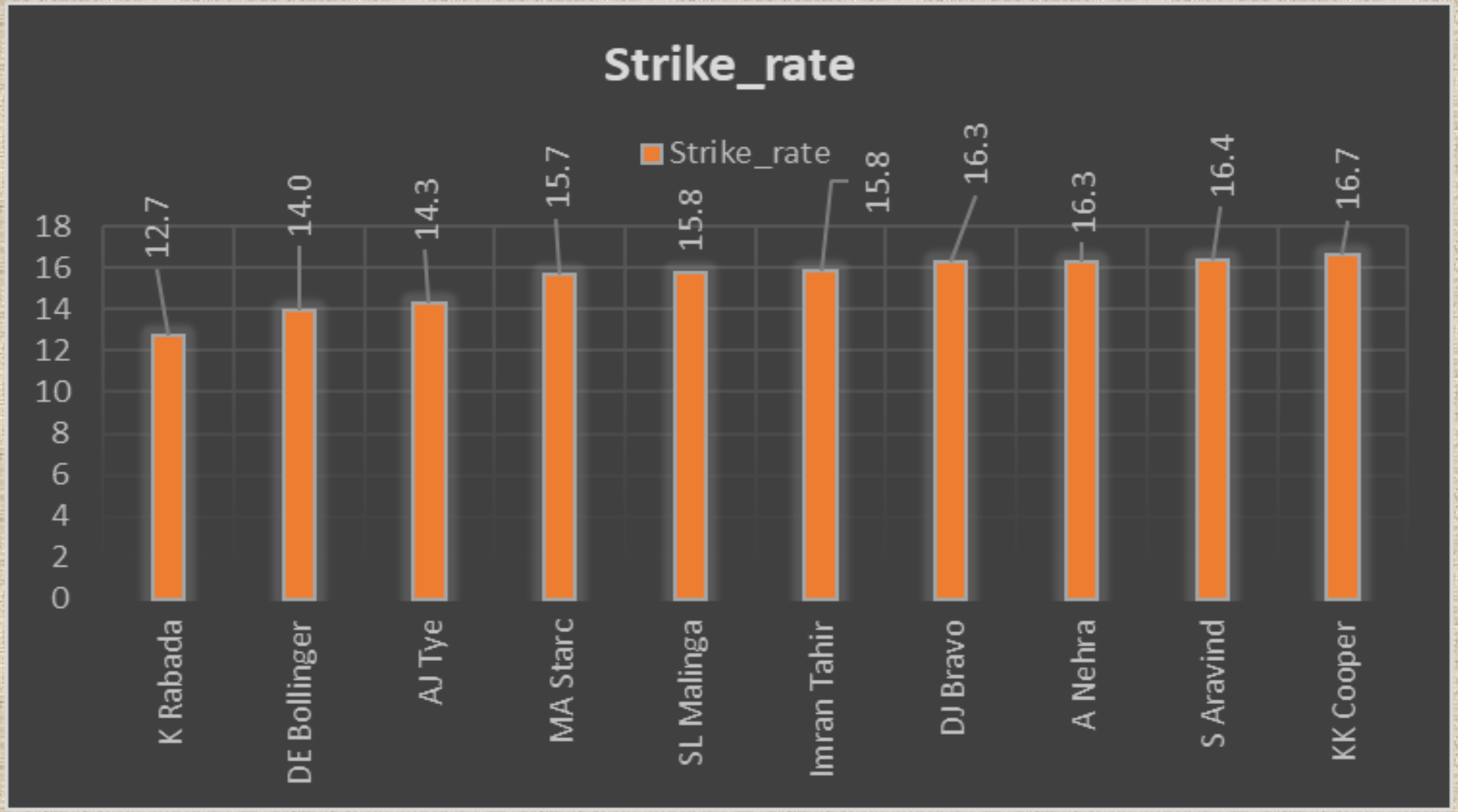
Analysis 5:

```
1. select * from ipl_player;

2. create table bsr as
    (select bowler ,count(ball) as balls_bowled ,
        sum (case when is_wicket = 1 then 1 else 0 end ) as
wickets_count
        from ipl_player group by bowler order by bowler)  ;
3. alter table bsr alter column balls_bowled type decimal ;
4. alter table bsr alter column wickets_count type  decimal;

5. select * from bsr ;

    /*select bowler ,count(ball) as balls_bowled ,
        sum (case when is_wicket = 1 then 1 else 0 end ) as
    wickets_count,
    ((count(ball))/(sum (case when is_wicket = 1 then 1 else 0 end )))
        as strike_rate from ipl_player
        where sum (case when is_wicket = 1 then 1 else 0 end ) > 0
    group by bowler;*/
6. select *, balls_bowled/wickets_count as strike_rate
        from bsr where wickets_count >0 and balls_bowled >= 500
    order by strike_rate LIMIT 10;
```

OUTPUT OF ANALYSIS 5

Bowler	Balls_bowled	Wickets_count	Strike_rate	Rank_sr
K Rabada	840	66	12.72727273	1
DE Bollinger	600	43	13.95348837	2
AJ Tye	645	45	14.33333333	3
MA Starc	612	39	15.69230769	4
SL Malinga	2974	188	15.81914894	5
Imran Tahir	1314	83	15.8313253	6
DJ Bravo	2846	175	16.26285714	7
A Nehra	1974	121	16.31404959	8
S Aravind	788	48	16.41666667	9
KK Cooper	600	36	16.66666667	10

GRAPH FOR ANALYSIS 5



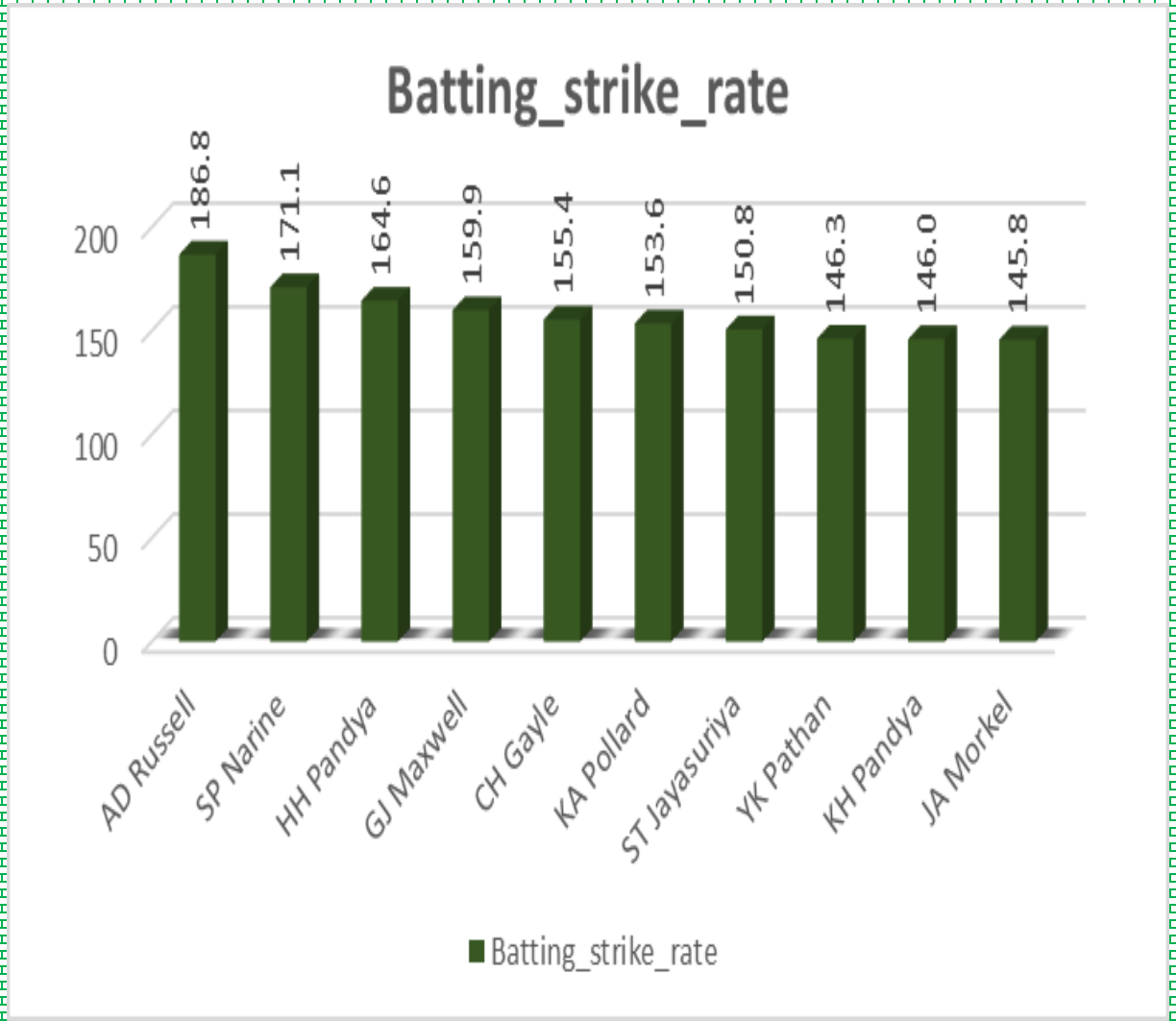
Analysis 6:

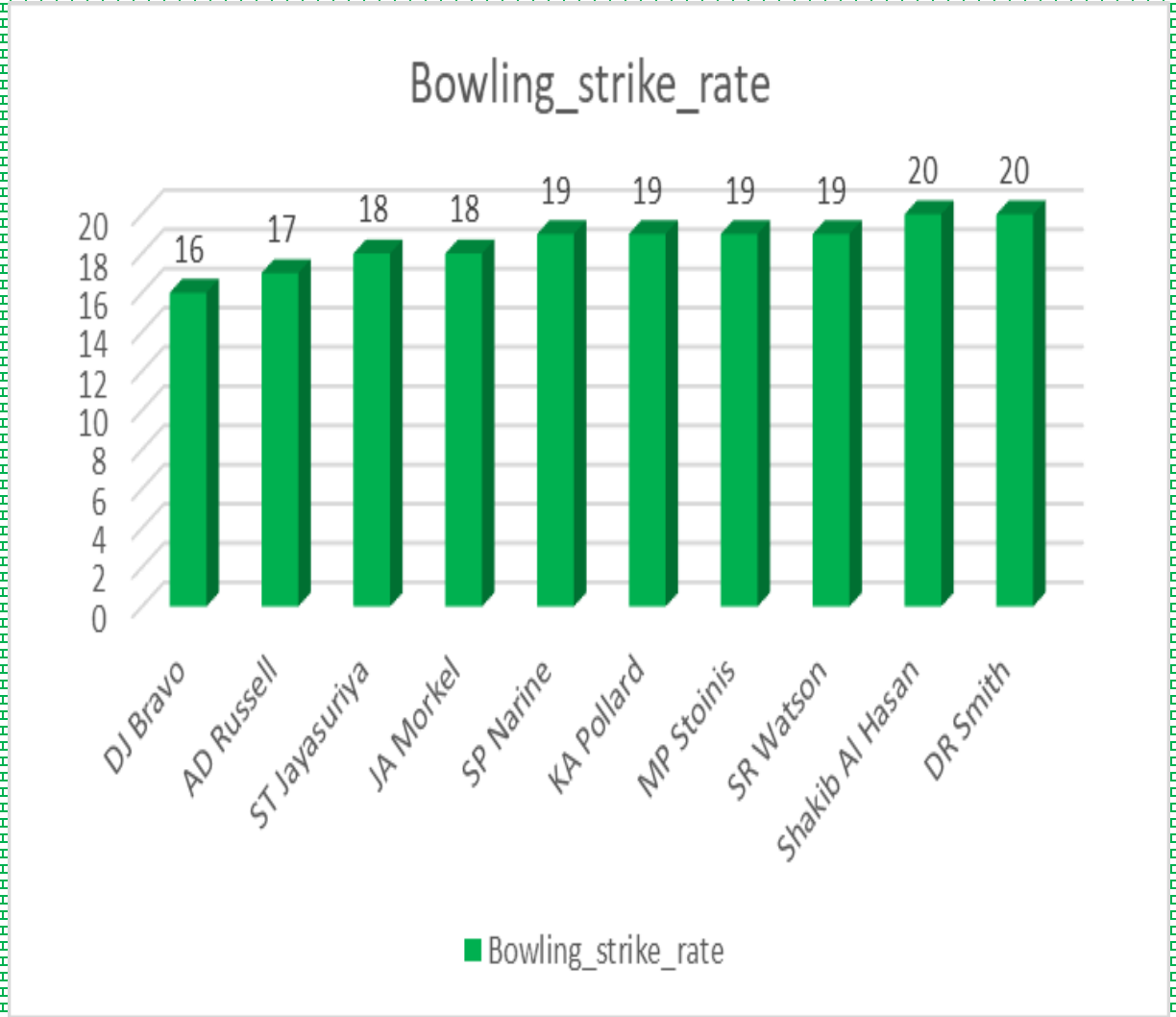
```
1. select * from ipl_player;
2. create table ba as (select batsman , sum(total_runs)
   as total_runs , count(ball) as balls_faced,
        sum(case when extras_type = 'wides' then 1 else
   0 end ) as wides_count ,
        (sum(total_runs) - sum(case when extras_type =
   'wides' then 1 else 0 end )) as actual_runs ,
   (count(ball)-sum(case when extras_type = 'wides'
   then 1 else 0 end ))
        as actual_balls_faced from ipl_player group by
   batsman  order by actual_runs desc);
3. select * from ba ;
4. alter table ba alter column actual_runs type decimal;
5. alter table ba alter column actual_balls_faced type
   decimal ;
6. create table abat as (select batsman , actual_runs,
   actual_balls_faced ,
        (actual_runs/actual_balls_faced)*100 as
   batting_strike_rate from ba);
7. select * from abat ;
8. create table boa as(select bowler , sum(total_runs) as
   runs_conceded, count(ball) as balls_bowled
        ,sum (case when is_wicket = 1 then 1 else 0 end )
   as wickets_count from ipl_player group by bowler) ;
9. select * from boa ;
10. create table aball as (select bowler ,
    runs_conceded,balls_bowled,wickets_count ,
        (balls_bowled/wickets_count) as
    bowling_strike_rate from boa where wickets_count >
    0 );
11. select * from aball ;
12. create table ars as (select abat.batsman as
    players_all_rounders, abat.actual_balls_faced ,
    abat.batting_strike_rate ,
    aball.balls_bowled,aball.bowling_strike_rate from
    abat full join aball on abat.batsman=aball.bowler
        where abat.batsman is not null and
    abat.actual_balls_faced  is not null and
        abat.batting_strike_rate  is not null and
    aball.balls_bowled is not null and
        aball.bowling_strike_rate is not null ) ;
13. select * from ars ;
14. create table allrounders_stat as (select * from ars
    where
        actual_balls_faced >= 500 and balls_bowled >=
    300) ;
15. select * from allrounders_stat ;
16. select
    players_all_rounders,actual_balls_faced,batting_strik
    e_rate,balls_bowled,bowling_strike_rate,
    rank () OVER (
    ORDER BY batting_strike_rate desc) as
    Rank_Batting_Strike_Rate
        FROM allrounders_stat limit 10 ;
17. select
    players_all_rounders,actual_balls_faced,batting_strik
    e_rate,balls_bowled,bowling_strike_rate,
    rank () OVER (
    ORDER BY bowling_strike_rate ) as
    Rank_Bowling_Strike_Rate
    from allrounders_stat limit 10;

18. select *  , rank () over (order by batting_strike_rate
    desc)  as Rank_Batting_Strike_Rate,
    dense_rank () OVER (
    ORDER BY bowling_strike_rate ) as
    Rank_Bowling_Strike_Rate
    from allrounders_stat limit 10;
```

OUTPUT OF ANALYSIS 6

Players_all_rounders	Actual_balls_faced	Batting_strike_rate	Balls_bowled	Bowling_strike_rate	Rank_batting_strike_rate
AD Russell	832	186.7788462	1186	17	1
SP Narine	543	171.0865562	2824	19	2
HH Pandya	847	164.5808737	914	20	3
GJ Maxwell	973	159.9177801	558	27	4
CH Gayle	3179	155.3947782	584	30	5
KA Pollard	2017	153.5944472	1414	19	6
ST Jayasuriya	532	150.7518797	301	18	7
YK Pathan	2241	146.2739848	1184	25	8
KH Pandya	702	146.011396	1283	26	9
JA Morkel	686	145.7725948	1807	18	10

Players_all_rounders	Actual_balls_faced	Batting_strike_rate	Balls_bowled	Bowling_strike_rate	Rank_bowling_strike_rate
DJ Bravo	1162	132.616179	2846	16	1
AD Russell	832	186.7788462	1186	17	2
ST Jayasuriya	532	150.7518797	301	18	3
JA Morkel	686	145.7725948	1807	18	3
SP Narine	543	171.0865562	2824	19	5
KA Pollard	2017	153.5944472	1414	19	5
MP Stoinis	601	140.4326123	562	19	5
SR Watson	2809	140.8686365	2137	19	5
Shakib Al Hasan	589	130.8998302	1358	20	9
DR Smith	1764	140.3628118	557	20	9


Batting_strike_rate
186.8
171.1
164.6
159.9
155.4
153.6
150.8
146.3
146.0
145.8
200
150
100
50
0
AD Russell
SP Narine
HH Pandya
GJ Maxwell
CH Gayle
KA Pollard
ST Jayasuriya
YK Pathan
KH Pandya
JA Morkel
Batting_strike_rate


Bowling_strike_rate
16
17
18
18
19
19
19
19
20
20
20
18
16
14
12
10
8
6
4
2
0
DJ Bravo
AD Russell
ST Jayasuriya
JA Morkel
SP Narine
KA Pollard
MP Stoinis
SR Watson
Shakib Al Hasan
DR Smith
Bowling_strike_rate

ANALYSIS 7 – CRITERIA FOR BUYING WICKET KEEPER IN THE AUCTION

- After buying the remaining players , the management will have a clear idea about the strength and weaknesses of the squad.
- I will suggest to select few batsman-wicketkeeper(options for injury) and also few bowler-wicket keeper(options for injury).
- Having these many options , will allow the captain and the coach to select a strong and balanced playing team because in the playing team there might be sufficient number of bowlers for a match (depending on the pitch) and then they might need good batting depth where batting wicket keeper might serve the purpose and vice versa.
- Criteria for selecting wicket keeper batsman : Good Average , Sufficiently Good Strike Rate , Medium Boundary Percentage , skilled with gloves i.e. good wicket keeping skills , have played sufficiently good number of matches to deal with extra ordinary situations in the field.
- Criteria for selecting wicket keeper bowler : economy < 10 , lesser bowling strike rate , skilled in wicket keeping

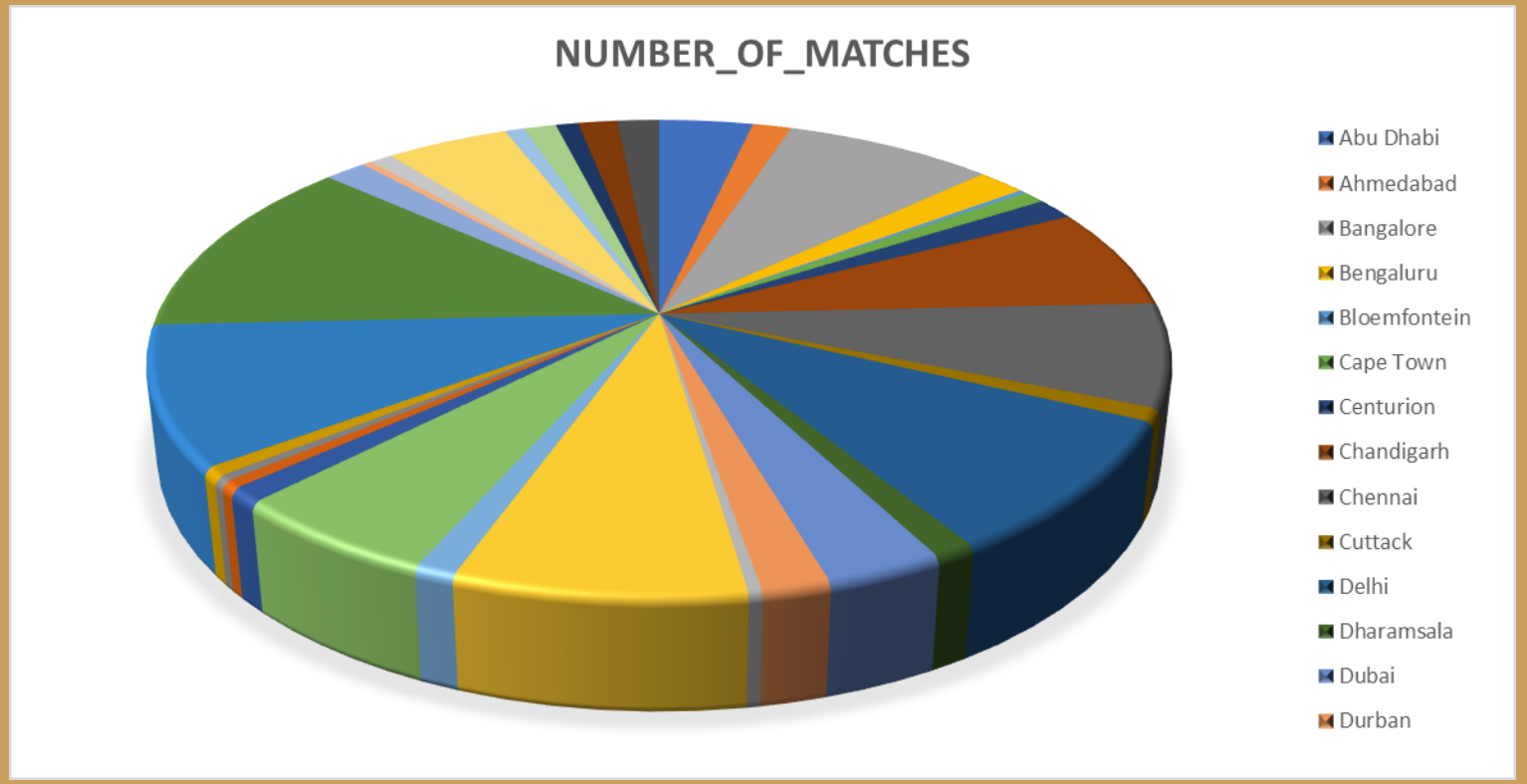
Additional Questions :

1. select city , count(match_date) as number_of_matches from ipl_match_data group by city order by city;
2. create table deliveries_v02 as (select * , case when total_runs >= 4 then 'Boundary'
   when total_runs = 0 then 'dot' else 'other'
   end as ball_result from ipl_player) ;
3. select sum(case when ball_result = 'Boundary' then 1 else 0 end) as boundary_count,
   sum(case when ball_result = 'dot' then 1 else 0 end) as dot_count
   from deliveries_v02 ;
4. select batting_team , sum(case when ball_result = 'Boundary' then 1 else 0 end)
   as boundary_num from deliveries_v02 group by batting_team order by boundary_num desc ;
5. select batting_team , sum(case when ball_result = 'dot' then 1 else 0 end)
   as dot_num from deliveries_v02 group by batting_team order by dot_num desc ;
6. select dismissal_kind , count(player_dismissed) as dismissal_count from
   deliveries_v02 where dismissal_kind <> 'NA' group by dismissal_kind order by dismissal_count desc ;
7. select bowler , sum(extra_runs) as extra_runs_conceded
   from deliveries_v02 group by bowler order by extra_runs_conceded desc limit 5;

8. create table deliveries_v03 as (select deliveries_v02.*, ipl_match_data.venue, ipl_match_data.match_date
   from deliveries_v02 full join ipl_match_data on
   deliveries_v02.id =ipl_match_data.id);
9. select venue , sum(total_runs) as total_runs from deliveries_v03 group by venue order by total_runs desc ;
10. select distinct extract (year from match_date) as yearwise_total_run,sum(total_runs) as total_runs from
    deliveries_v03 where venue ='Eden Gardens' group by yearwise_total_run;

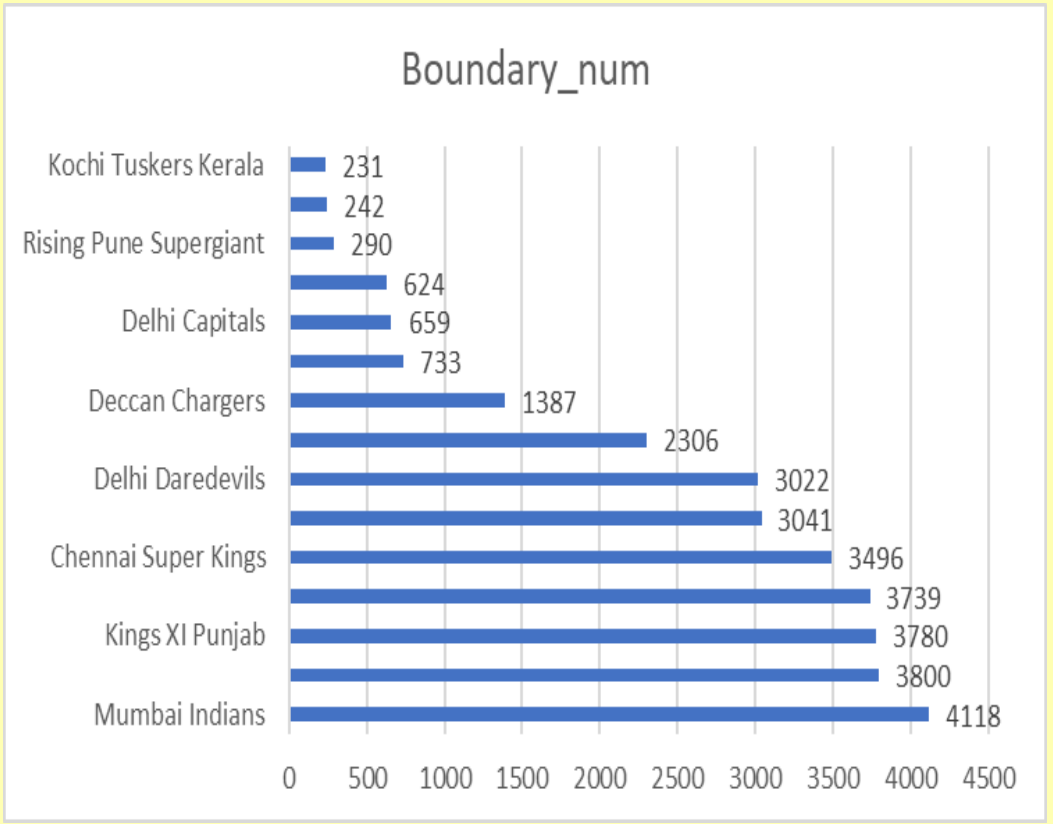
Additional Question 1

City	Number_of_matche
Abu Dhabi	29
Ahmedabad	12
Bangalore	65
Bengaluru	15
Bloemfontein	2
Cape Town	7
Centurion	12
Chandigarh	56
Chennai	57
Cuttack	7
Delhi	74
Dharamsala	9
Dubai	26
Durban	15
East London	3
Hyderabad	64
Indore	9
Jaipur	47
Johannesburg	8
Kanpur	4
Kimberley	3
Kochi	5
Kolkata	77
Mumbai	101
NA	13
Nagpur	3
Port Elizabeth	7
Pune	38
Raipur	6
Rajkot	10
Ranchi	7
Sharjah	12
Visakhapatnam	13



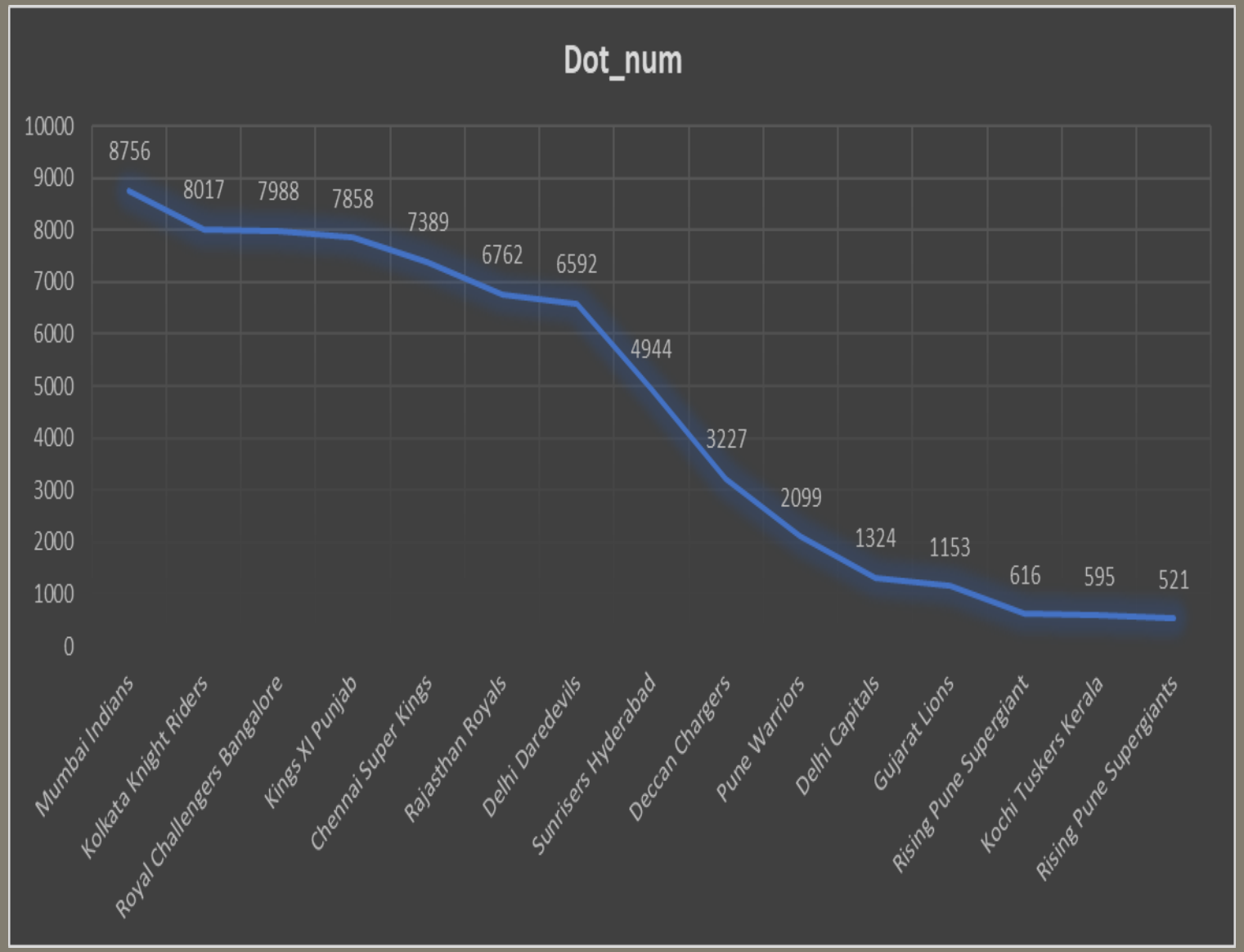
Additional Question 4

Batting_team	Boundary_num
Mumbai Indians	4118
Royal Challengers Bangalore	3800
Kings XI Punjab	3780
Kolkata Knight Riders	3739
Chennai Super Kings	3496
Rajasthan Royals	3041
Delhi Daredevils	3022
Sunrisers Hyderabad	2306
Deccan Chargers	1387
Pune Warriors	733
Delhi Capitals	659
Gujarat Lions	624
Rising Pune Supergiant	290
Rising Pune Supergiants	242
Kochi Tuskers Kerala	231



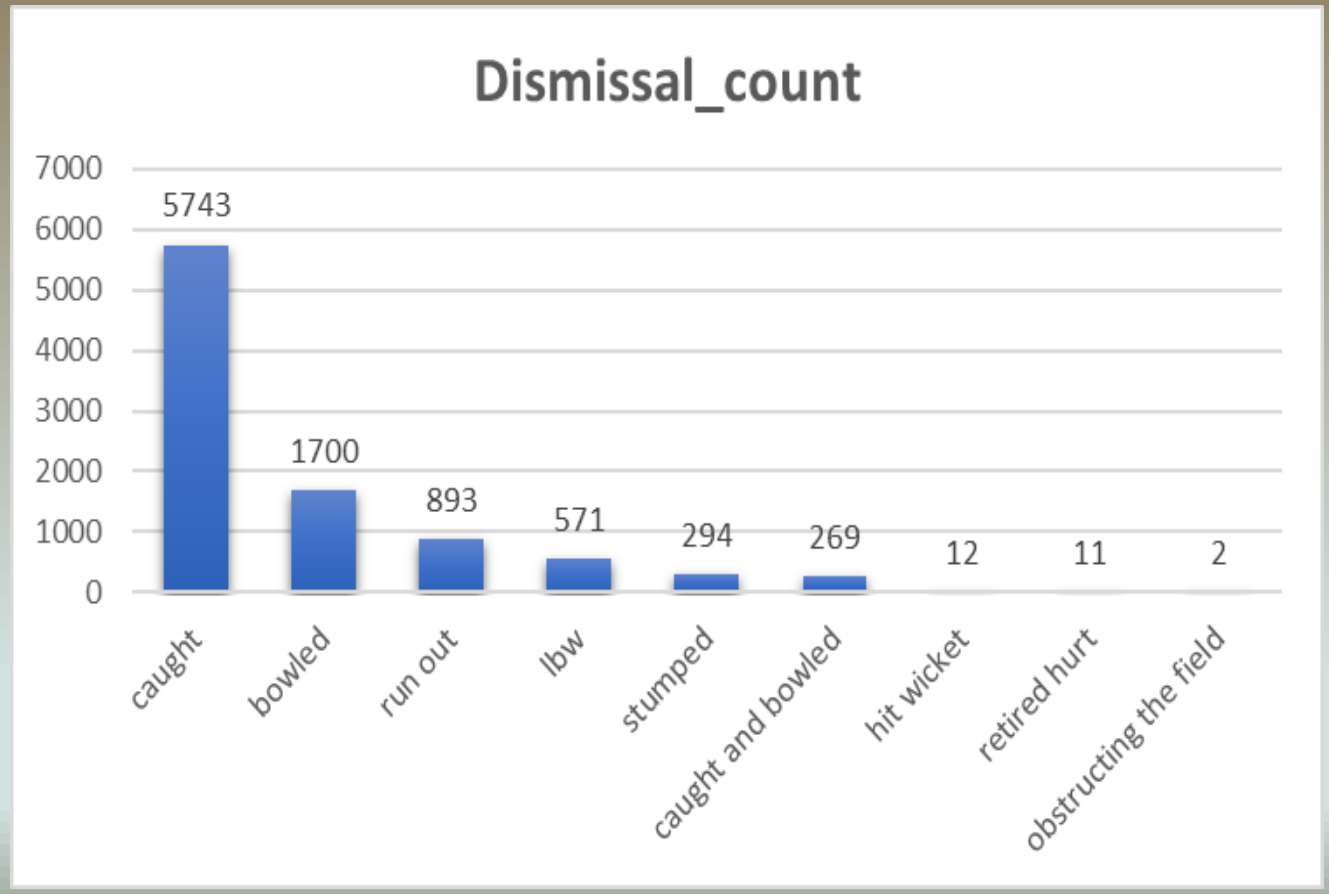
ADDITIONAL QUESTION 5

Batting_team	Dot_num
Mumbai Indians	8756
Kolkata Knight Riders	8017
Royal Challengers Bangalore	7988
Kings XI Punjab	7858
Chennai Super Kings	7389
Rajasthan Royals	6762
Delhi Daredevils	6592
Sunrisers Hyderabad	4944
Deccan Chargers	3227
Pune Warriors	2099
Delhi Capitals	1324
Gujarat Lions	1153
Rising Pune Supergiant	616
Kochi Tuskers Kerala	595
Rising Pune Supergiants	521



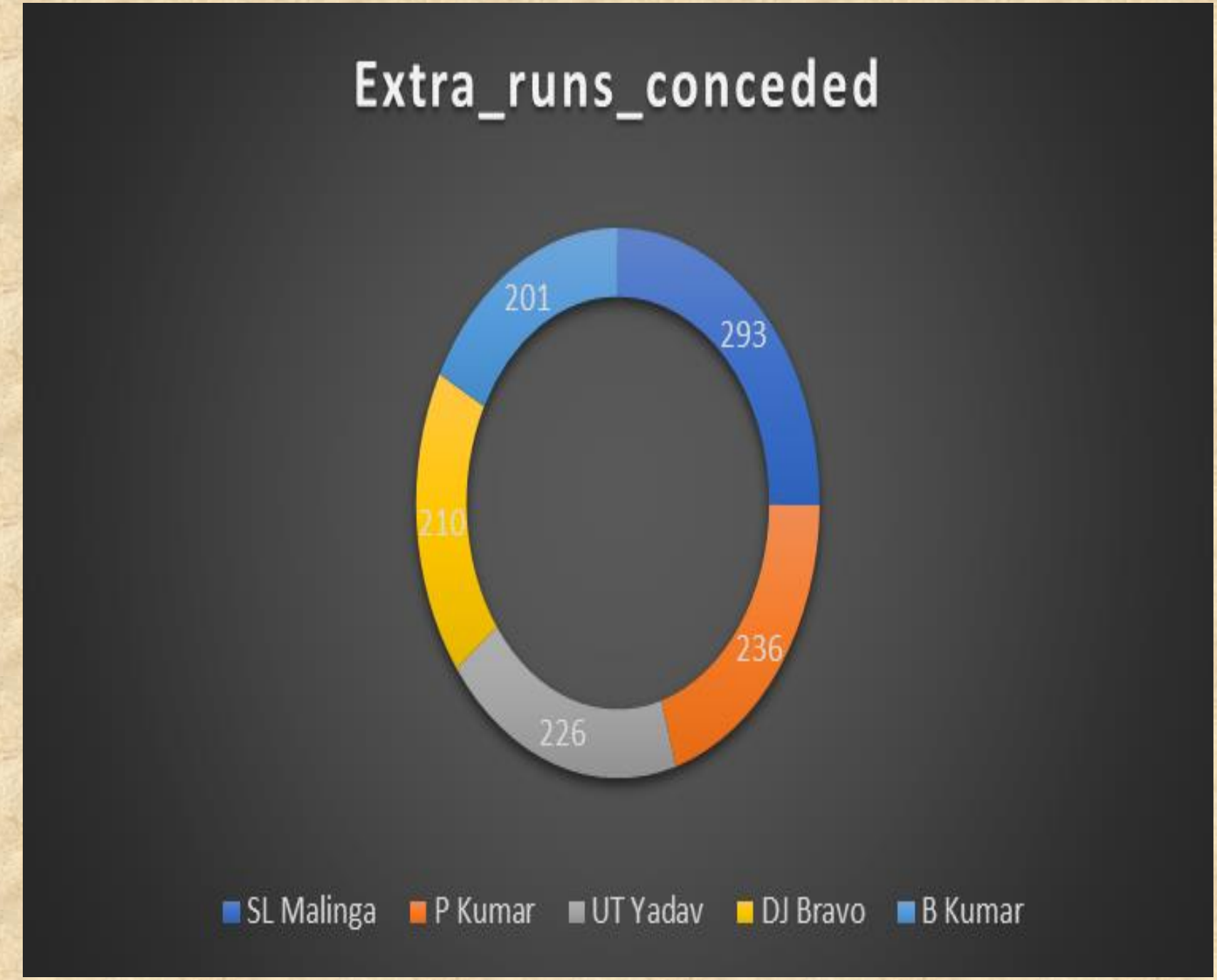
Additional Question 6

Dismissal_kind	Dismissal_count
caught	5743
bowled	1700
run out	893
lbw	571
stumped	294
caught and bowled	269
hit wicket	12
retired hurt	11
obstructing the field	2



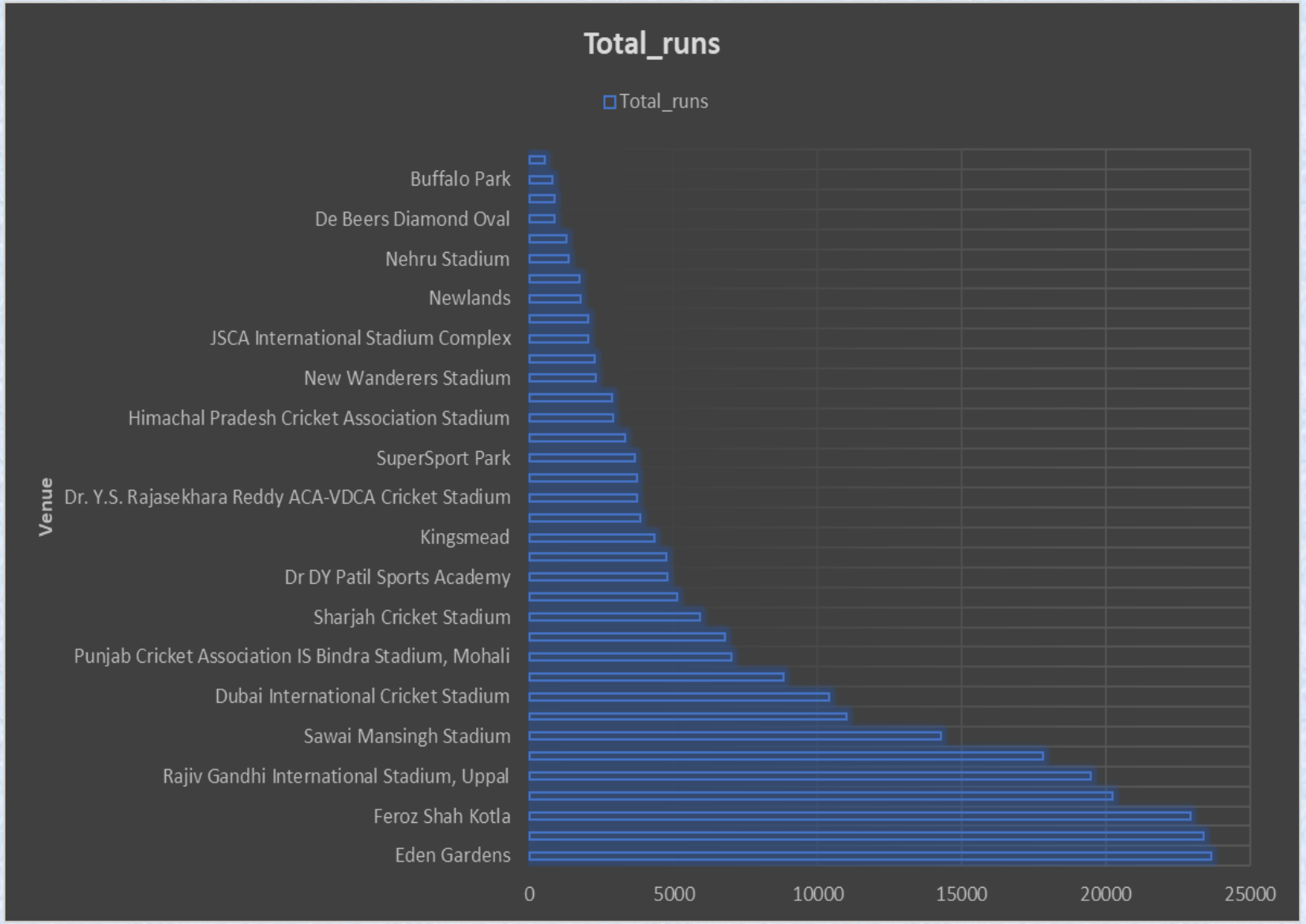
Additional Question 7

Bowler	Extra_runs_conceded
SL Malinga	293
P Kumar	236
UT Yadav	226
DJ Bravo	210
B Kumar	201



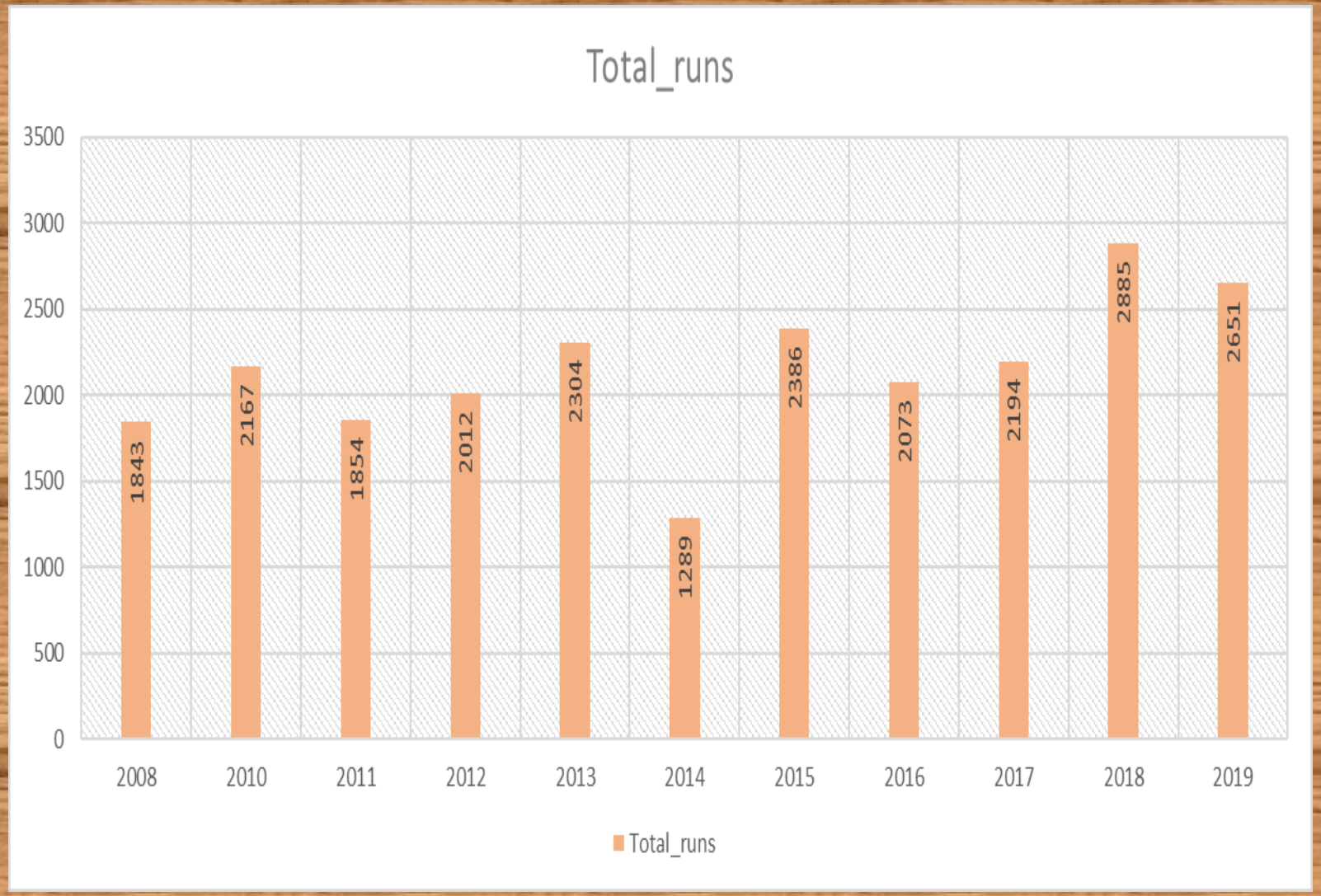
Additional Question 9

Venue	Total_run
Eden Gardens	23658
Wankhede Stadium	23390
Feroz Shah Kotla	22947
M Chinnaswamy Stadium	20237
Rajiv Gandhi International Stadium, Uppal	19484
MA Chidambaram Stadium, Chepauk	17821
Sawai Mansingh Stadium	14264
Punjab Cricket Association Stadium, Mohali	10987
Dubai International Cricket Stadium	10402
Sheikh Zayed Stadium	8830
Punjab Cricket Association IS Bindra Stadium, Mohali	7021
Maharashtra Cricket Association Stadium	6780
Sharjah Cricket Stadium	5924
M.Chinnaswamy Stadium	5127
Dr DY Patil Sports Academy	4810
Subrata Roy Sahara Stadium	4755
Kingsmead	4353
Brabourne Stadium	3842
Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium	3746
Sardar Patel Stadium, Motera	3746
SuperSport Park	3653
Saurashtra Cricket Association Stadium	3316
Himachal Pradesh Cricket Association Stadium	2897
Holkar Cricket Stadium	2872
New Wanderers Stadium	2292
Barabati Stadium	2278
JSCA International Stadium Complex	2056
St George's Park	2033
Newlands	1764
Shaheed Veer Narayan Singh International Stadium	1741
Nehru Stadium	1363
Green Park	1298
De Beers Diamond Oval	897
Vidarbha Cricket Association Stadium, Jamtha	882
Buffalo Park	799
OUTsurance Oval	529



Additional Question 10

Yearwise_total_runs Of Eden Gardens	Total_runs
2008	1843
2010	2167
2011	1854
2012	2012
2013	2304
2014	1289
2015	2386
2016	2073
2017	2194
2018	2885
2019	2651



THANK YOU