IPL DATA

Vijay Shankar

INDEX

- Slides 3 & 4 → CREATE TABLE QUERIES
- Slides 5 & 6 → BATSMAN STRIKE RATE
- Slides 7 to 9 → AVERAGE SCORE OF BATSMAN
- Slides 10 & 11 → BOUNDARY PERCENTAGE OF BATSMAN
- Slides 12 & 13 → BOWLER ECONOMY RATE
- Slides 14 to 16 → BOWLER STRIKE RATE
- Slides 17 & 18 → ALL ROUNDER
- Slides 19 & 20 → CRITERIA FOR WICKET KEEPER
- Slides 21 to 29 → FINAL ASSESSMENT QUESTIONS

CREATE TABLE QUERY FOR MATCHES TABLE

```
CREATE TABLE Matches
 (id INT NOT NULL PRIMARY KEY, city VARCHAR(50) NOT
NULL,
  date DATE NOT NULL, player_of_match VARCHAR(50),
  venue VARCHAR, neutral_venue BIT,
  team1 VARCHAR(50), team2 VARCHAR(50),
  toss_winner VARCHAR(50), toss_decision  VARCHAR(10),
  winner VARCHAR(50), result VARCHAR(15),
  result_margin INT, eliminator VARCHAR(10),
  method VARCHAR(10), umpire1 VARCHAR(50),
  umpire2 VARCHAR(50));

COPY Matches FROM 'C:\Program Files\PostgreSQL\IPL
Dataset\IPL Dataset\IPL_matches.csv'
DELIMITER ',' CSV HEADER;
```

CREATE TABLE QUERY FOR DELIVERIES TABLE

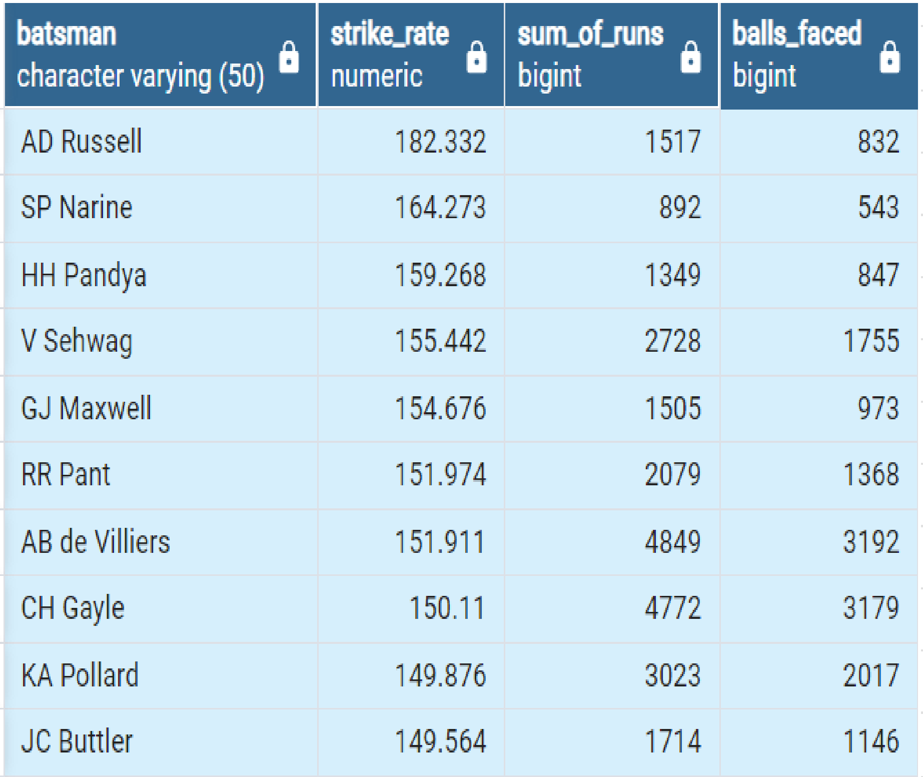
```
CREATE TABLE Deliveries
( id INT NOT NULL, CONSTRAINT FKEY_ID FOREIGN KEY (id)
REFERENCES Matches(id),
  inning INT, over INT,
  ball INT, batsman VARCHAR(50),
  non_striker VARCHAR(50), bowler VARCHAR(50),
  batsman_runs INT, extra_runs INT,
  total_runs INT, is_wicket INT,
  dismissal_kind VARCHAR(50), player_dismissed VARCHAR(50),
  fielder VARCHAR(50), extras_type VARCHAR(20),
  batting_team VARCHAR(100), bowling_team VARCHAR(50));

COPY Deliveries FROM 'C:\Program Files\PostgreSQL\IPL
Dataset\IPL Dataset\IPL_Ball.csv'
DELIMITER ',' CSV HEADER;
```

Q1 - Your first priority is to get 2-3 players with high S.R who have faced at least 500 balls. And to do that you have to make a list of 10 players you want to bid in the auction so that when you try to grab them in auction you should not pay the amount greater than you have in the purse for a particular player.

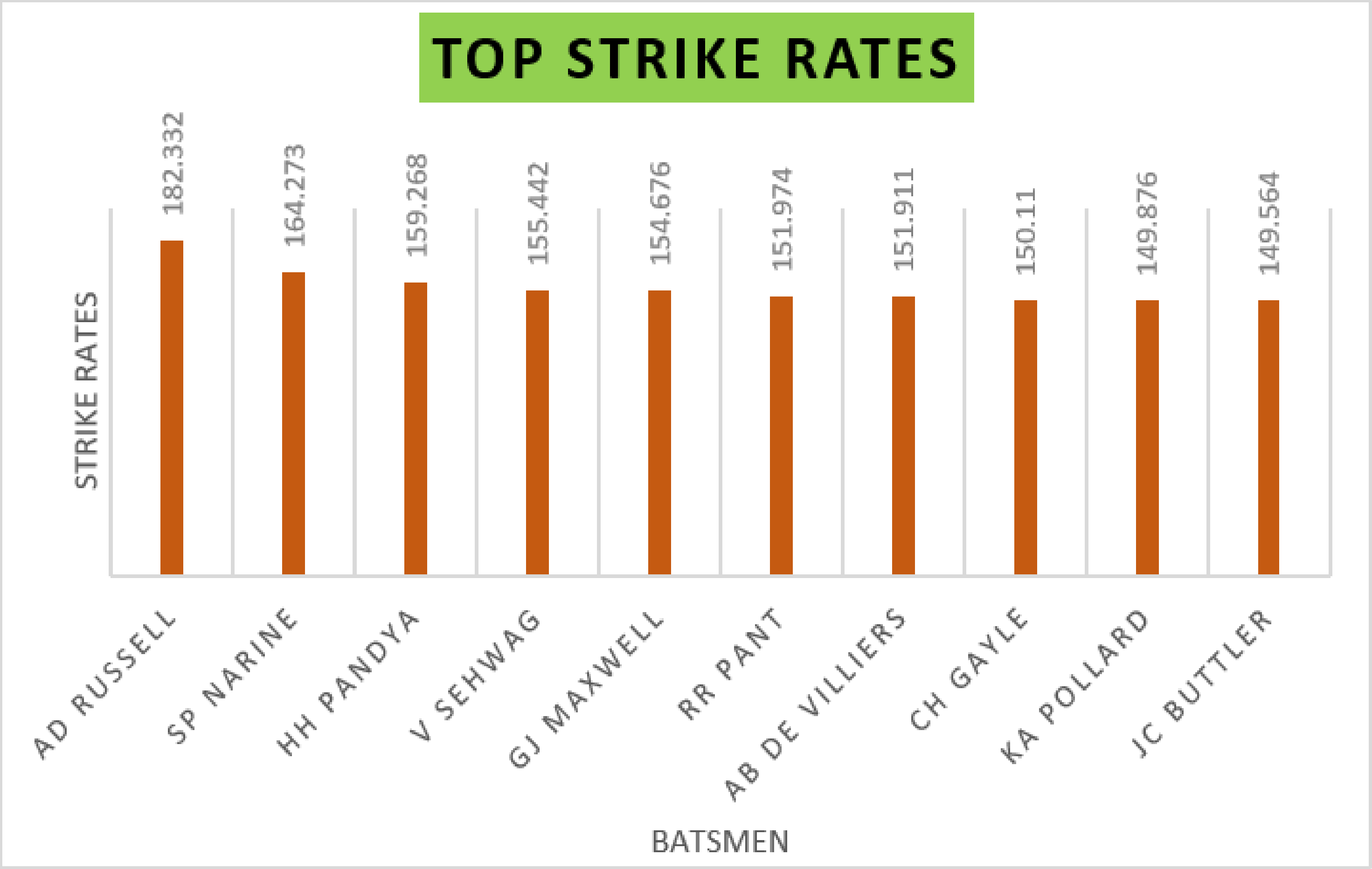
(strike rate is total runs scored by batsman divided by number of balls faced but remember when extras_type is 'wides' it is not counted as a ball faced neither counted as batsmen runs)

QUERY

```
SELECT batsman,
ROUND(SUM(batsman_runs)/CAST(COUNT(ball) as
DECIMAL)*100,3) AS Strike_Rate,
SUM(batsman_runs) AS sum_of_runs, COUNT(ball)
AS balls_faced
FROM deliveries
WHERE extras_type NOT IN ('wides')
GROUP BY batsman
HAVING COUNT(ball) >= 500
ORDER BY Strike_Rate DESC
LIMIT 10;
```



batsman character varying (50)	strike_rate numeric	sum_of_runs bigint	balls_faced bigint
AD Russell	182.332	1517	832
SP Narine	164.273	892	543
HH Pandya	159.268	1349	847
V Sehwag	155.442	2728	1755
GJ Maxwell	154.676	1505	973
RR Pant	151.974	2079	1368
AB de Villiers	151.911	4849	3192
CH Gayle	150.11	4772	3179
KA Pollard	149.876	3023	2017
JC Buttler	149.564	1714	1146

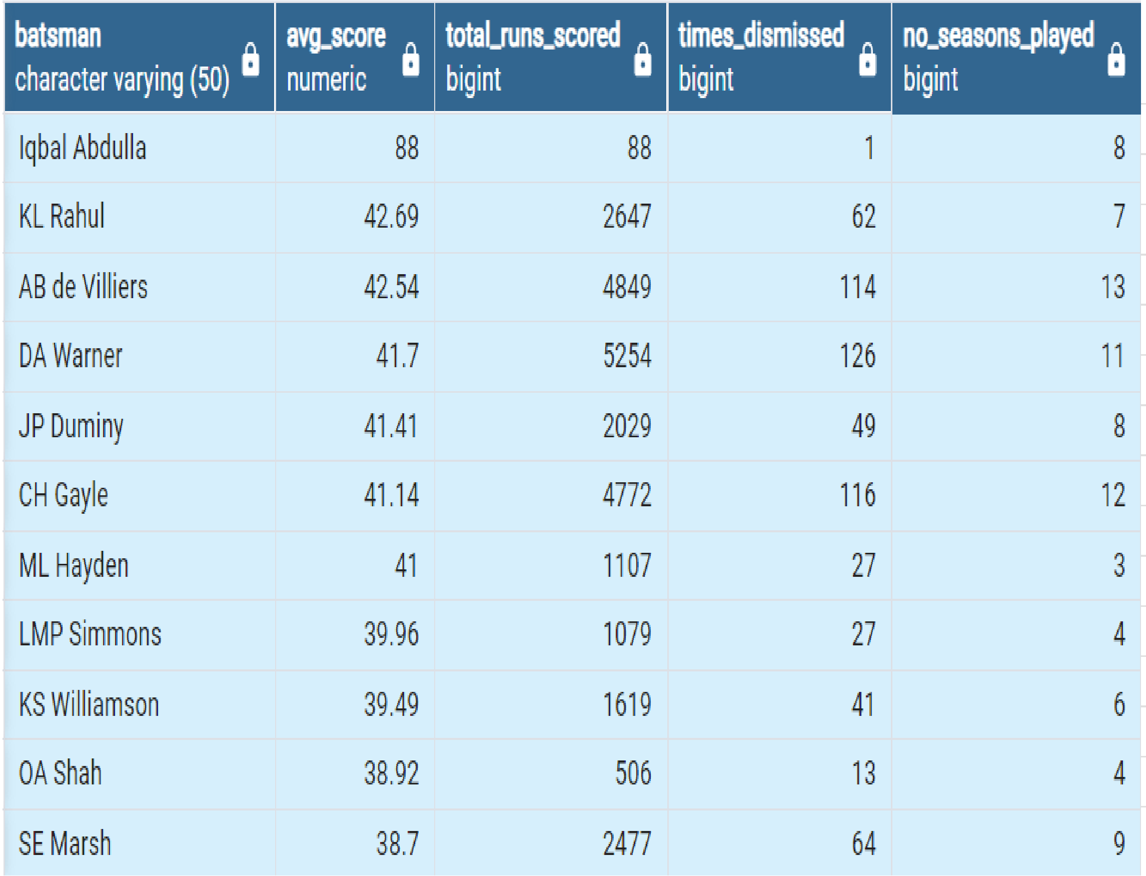


Q2 - Now you need to get 2-3 players with good Average who have played more than 2 ipl seasons. And to do that you have to make a list of 10 players you want to bid in the auction so that when you try to grab them in auction you should not pay the amount greater than you have in the purse for a particular player.

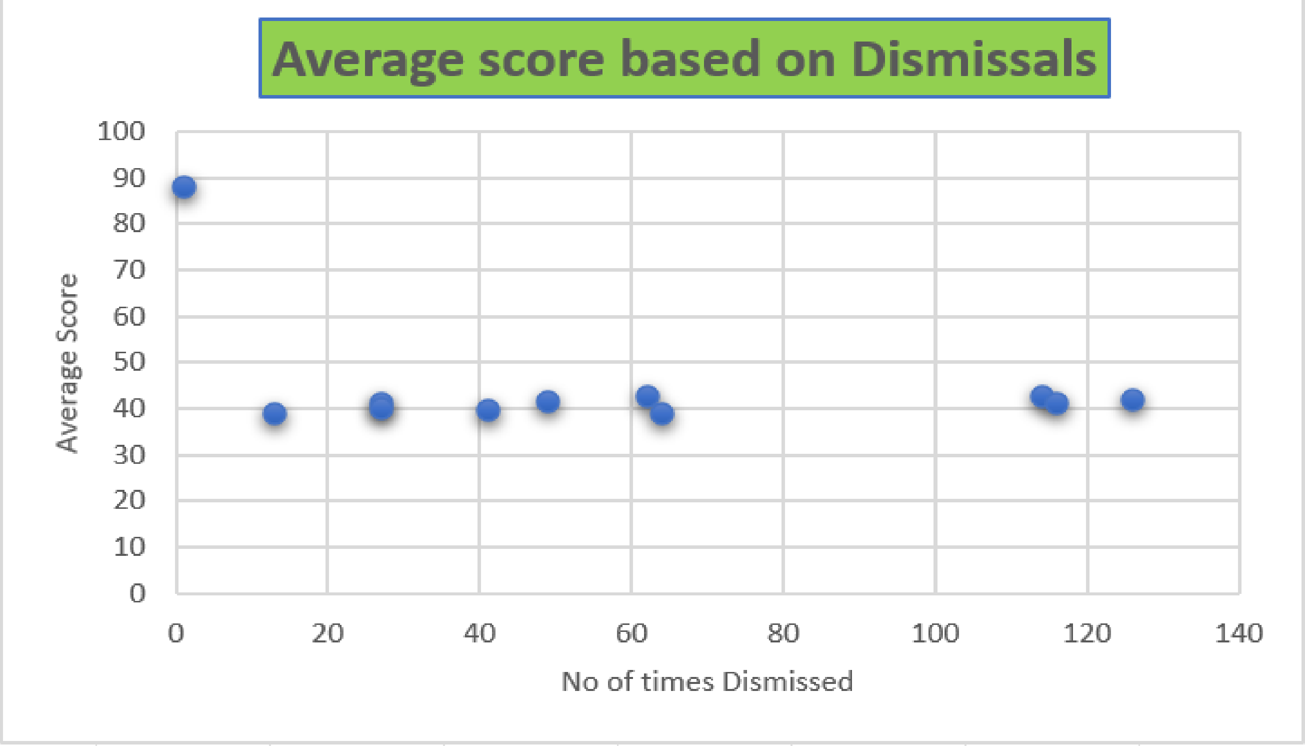
(Average is calculated as total runs scored divided by number of times batsman has been dismissed which can be calculated using wicket_ball field as 1 indicates out and 0 indicates not out, a batsman should've been dismissed at least once to calculate the sr i.e., you can exclude those players who have not been dismissed once)

QUERY

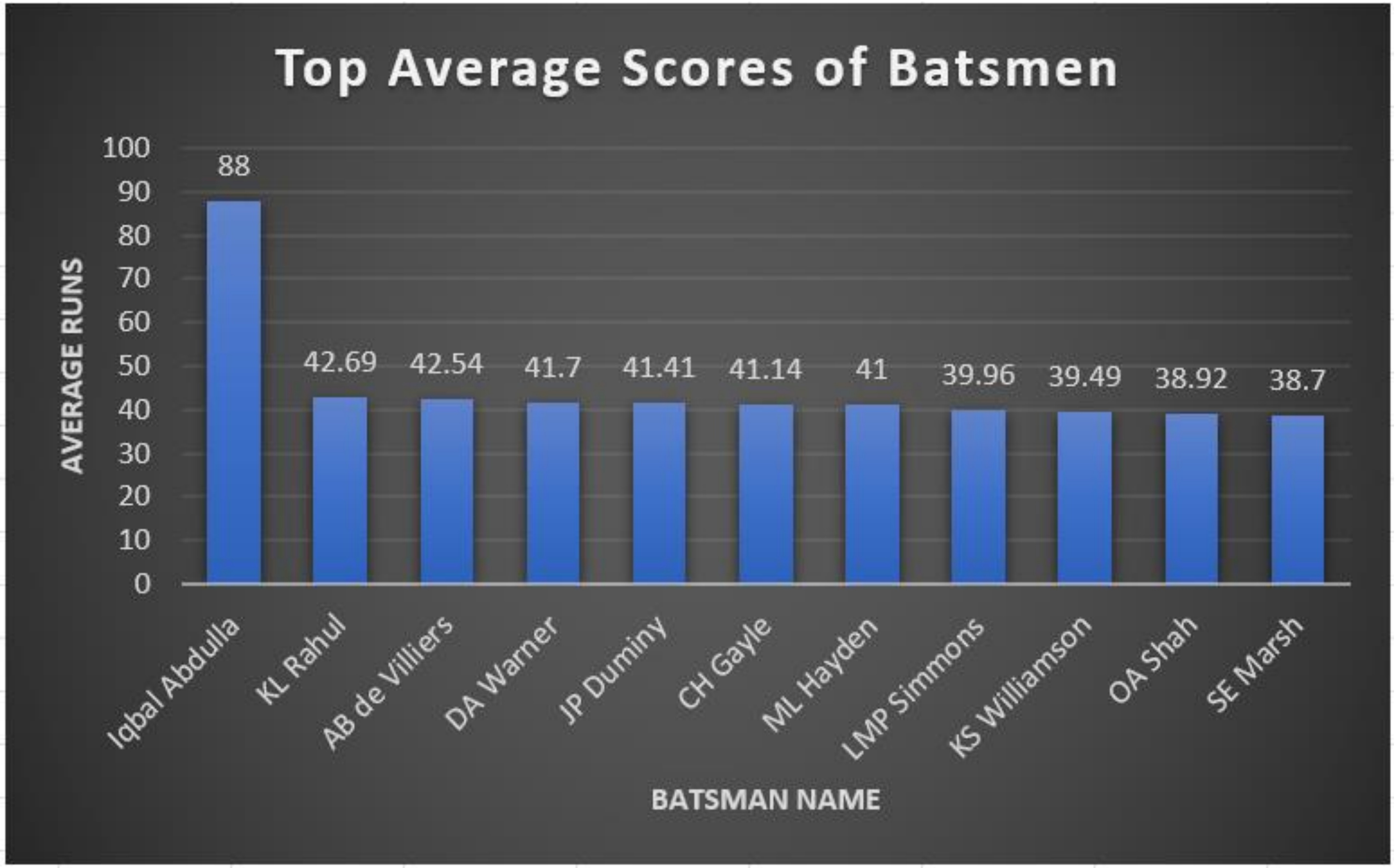
```
SELECT batsman,
ROUND(CAST(SUM(d.batsman_runs) AS
DECIMAL)/SUM(d.is_wicket),2) AS avg_score,
SUM(d.batsman_runs) AS total_runs_scored,
SUM(is_wicket) AS times_dismissed,
COUNT(DISTINCT(EXTRACT(YEAR FROM
m.date))) AS no_seasons_played
FROM deliveries AS d INNER JOIN matches AS m
ON d.id = m.id
GROUP BY d.batsman
HAVING SUM(is_wicket)>=1 AND
COUNT(DISTINCT(EXTRACT(YEAR FROM
m.date))) > 2
ORDER BY avg_score DESC
LIMIT 11;
```



batsman character varying (50)	avg_score numeric	total_runs_scored bigint	times_dismissed bigint	no_seasons_played bigint
Iqbal Abdulla	88	88	1	8
KL Rahul	42.69	2647	62	7
AB de Villiers	42.54	4849	114	13
DA Warner	41.7	5254	126	11
JP Duminy	41.41	2029	49	8
CH Gayle	41.14	4772	116	12
ML Hayden	41	1107	27	3
LMP Simmons	39.96	1079	27	4
KS Williamson	39.49	1619	41	6
OA Shah	38.92	506	13	4
SE Marsh	38.7	2477	64	9



Average score based on Dismissals

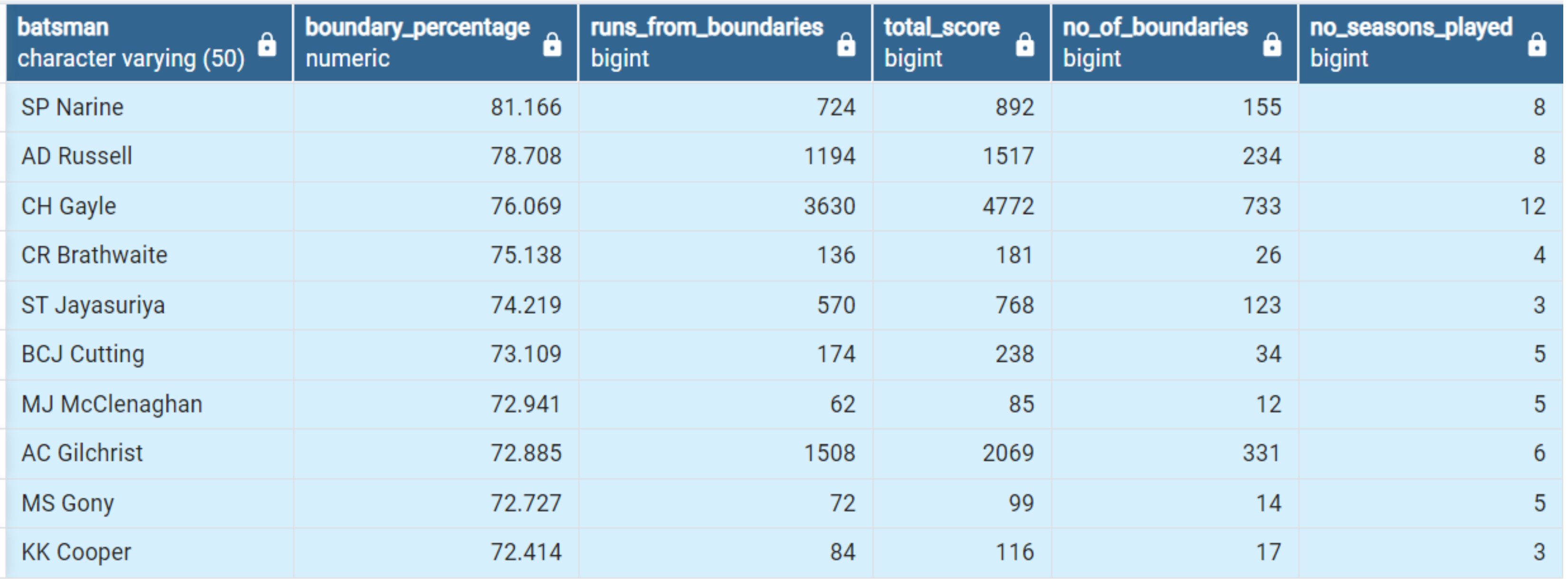


NOTE: In the query for Average score, 11 batsmen have been included because the batsman with the top average (88) has been dismissed only once and may be considered as exception.

Q3 - Now you need to get 2-3 Hard-hitting players who have scored most runs in boundaries and have played more the 2 ipl season. To do that you have to make a list of 10 players you want to bid in the auction so that when you try to grab them in auction you should not pay the amount greater than you have in the purse for a particular player. (only 4 and 6 will be counted as boundaries so calculate how many 4 and 6 has been hit by each batsman and also calculate total runs scored to get the output as boundary percentage which will be runs in boundary divided by total runs scored)

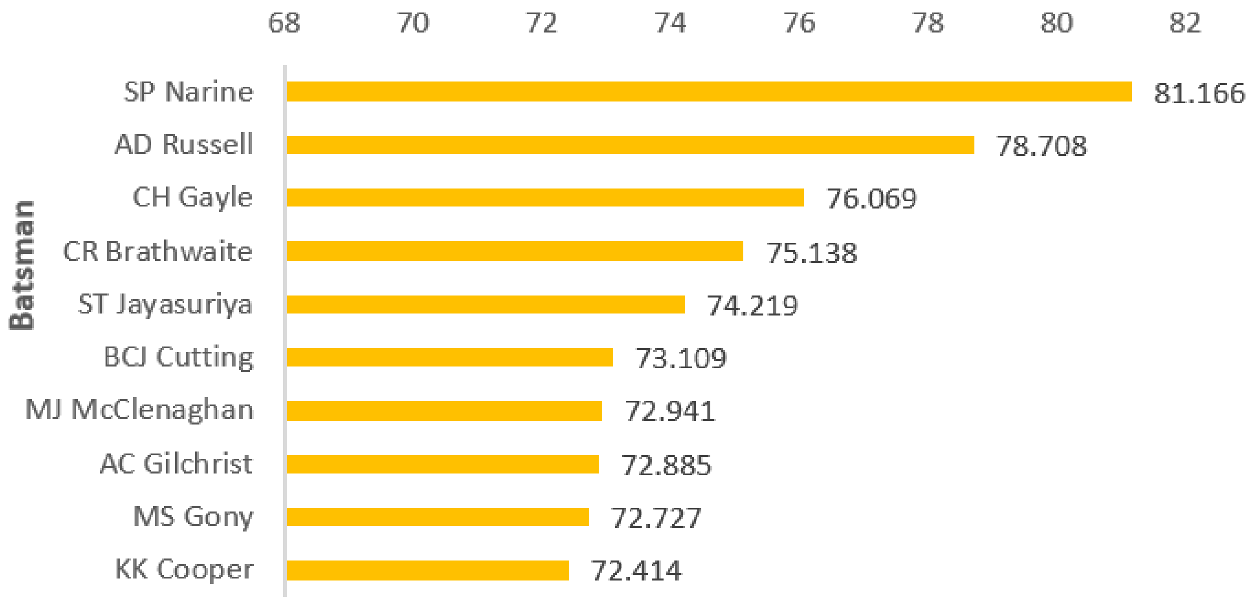
QUERY

```
SELECT d.batsman,
ROUND(SUM(CASE WHEN batsman_runs IN (4,6) THEN batsman_runs
ELSE 0 END)/CAST(SUM(batsman_runs) AS DECIMAL) * 100,3)
AS boundary_percentage,
SUM(CASE WHEN batsman_runs IN (4,6) THEN batsman_runs ELSE 0
END) AS runs_from_boundaries,
SUM(batsman_runs) AS total_score,
SUM(CASE WHEN batsman_runs IN (4,6) THEN 1 ELSE 0 END) AS
no_of_boundaries,
COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) AS
no_seasons_played
FROM Deliveries AS d INNER JOIN Matches AS m
ON d.id = m.id
WHERE batsman_runs > 0
GROUP BY d.batsman
HAVING COUNT(DISTINCT(EXTRACT(YEAR FROM m.date))) > 2
ORDER BY boundary_percentage DESC
LIMIT 10;
```



batsman character varying (50)	boundary_percentage numeric	runs_from_boundaries bigint	total_score bigint	no_of_boundaries bigint	no_seasons_played bigint
SP Narine	81.166	724	892	155	8
AD Russell	78.708	1194	1517	234	8
CH Gayle	76.069	3630	4772	733	12
CR Brathwaite	75.138	136	181	26	4
ST Jayasuriya	74.219	570	768	123	3
BCJ Cutting	73.109	174	238	34	5
MJ McClenaghan	72.941	62	85	12	5
AC Gilchrist	72.885	1508	2069	331	6
MS Gony	72.727	72	99	14	5
KK Cooper	72.414	84	116	17	3

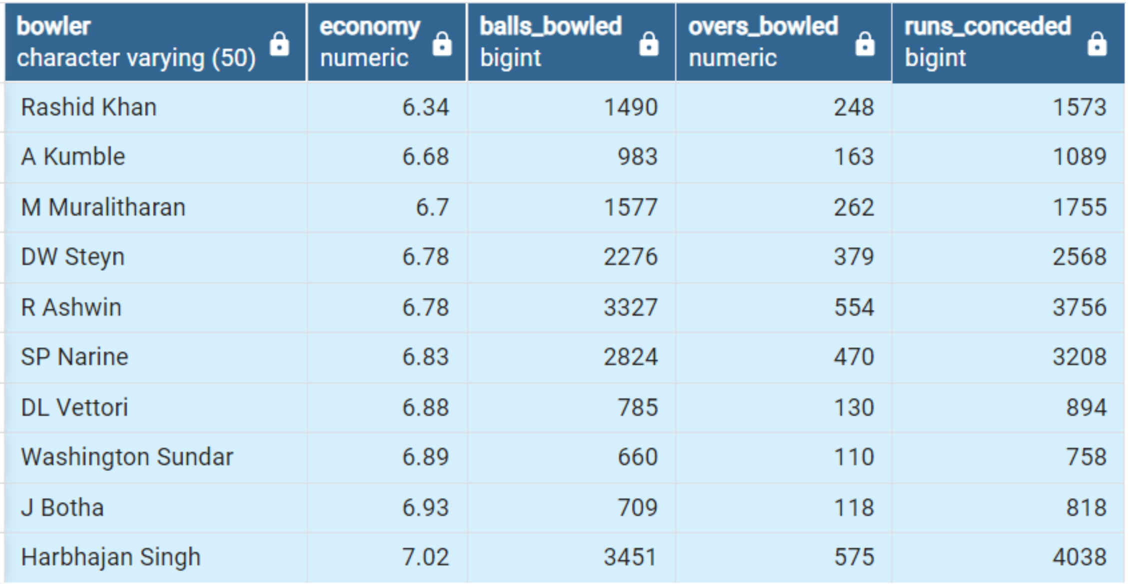
BOUNDARY PERCENTAGES



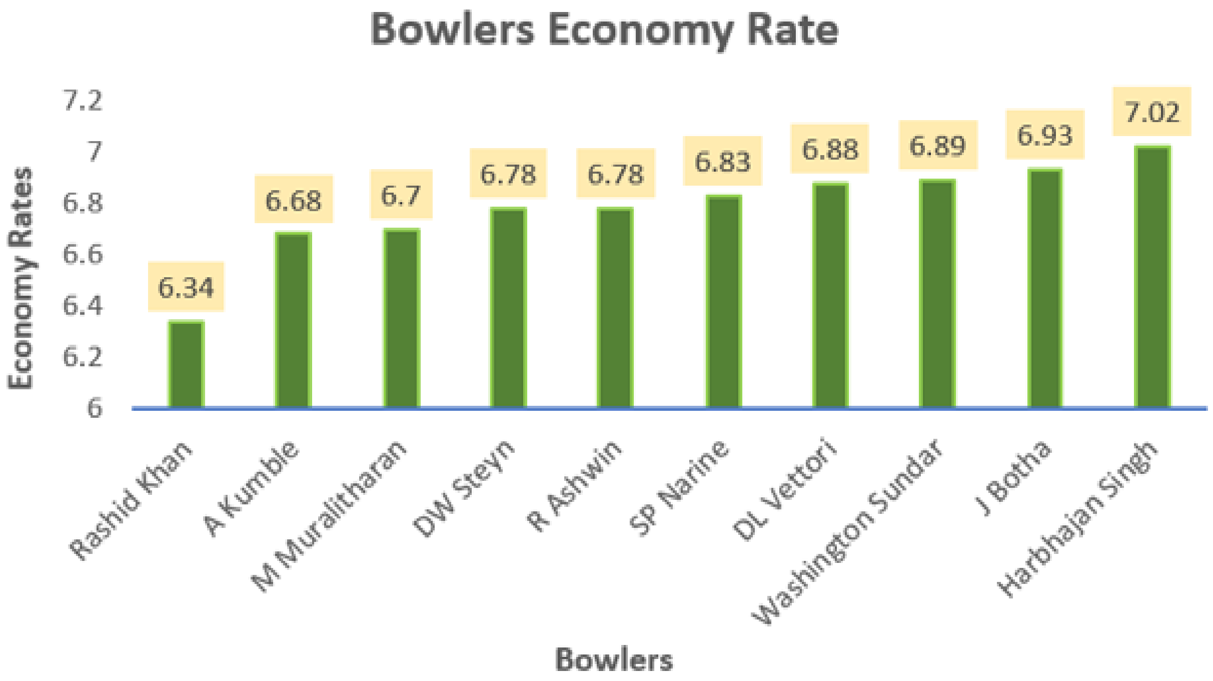
Q4 - Your first priority is to get 2-3 bowlers with good economy who have bowled at least 500 balls in IPL so far. To do that you have to make a list of 10 players you want to bid in the auction so that when you try to grab them in auction you should not pay the amount greater than you have in the purse for a particular player.(economy can be calculated by dividing total runs conceded with total overs bowled)

QUERY

```
SELECT bowler,
ROUND(CAST(SUM(total_runs) AS
decimal)/(COUNT(ball)/6),2) AS economy,
COUNT(ball) AS balls_bowled,
CAST(COUNT(ball)/6 AS DECIMAL) AS
overs_bowled,
SUM(total_runs) AS runs_conceded
FROM deliveries
GROUP BY bowler
HAVING COUNT(ball) >= 500
ORDER BY economy
LIMIT 10;
```



bowler character varying (50)	economy numeric	balls_bowled bigint	overs_bowled numeric	runs_conceded bigint
Rashid Khan	6.34	1490	248	1573
A Kumble	6.68	983	163	1089
M Muralitharan	6.7	1577	262	1755
DW Steyn	6.78	2276	379	2568
R Ashwin	6.78	3327	554	3756
SP Narine	6.83	2824	470	3208
DL Vettori	6.88	785	130	894
Washington Sundar	6.89	660	110	758
J Botha	6.93	709	118	818
Harbhajan Singh	7.02	3451	575	4038

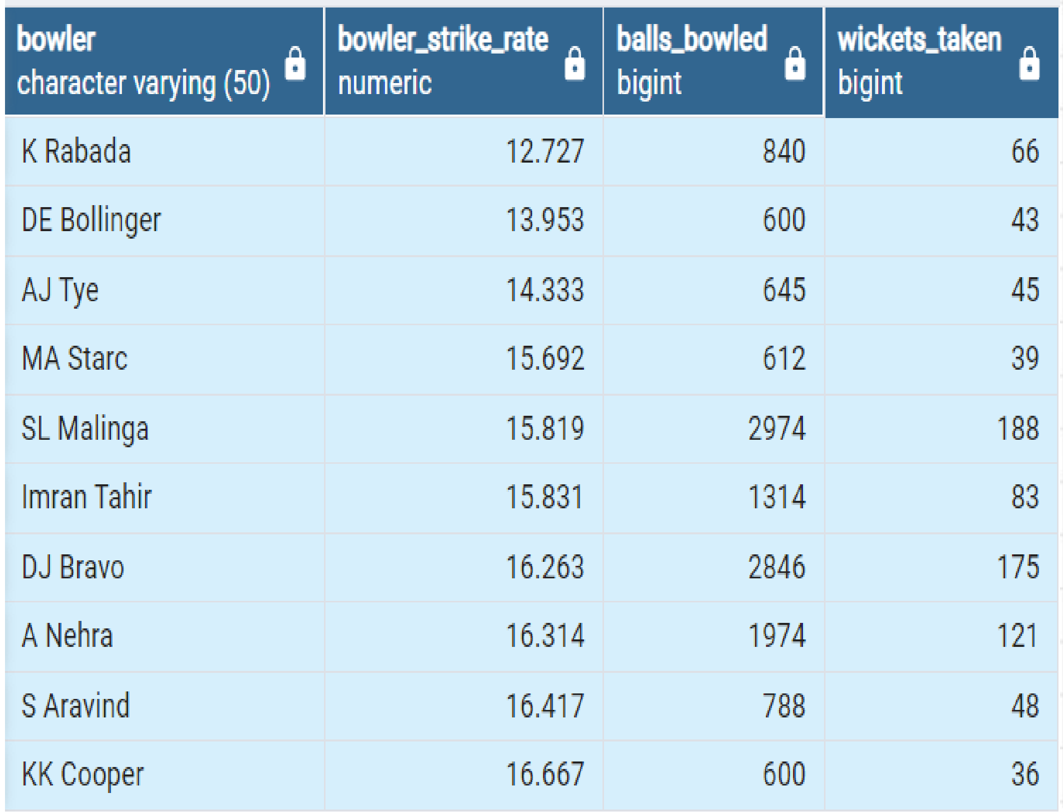


Q5- Now you need to get 2-3 bowlers with the best strike rate and who have bowled at least 500 balls in IPL so far. To do that you have to make a list of 10 players you want to bid in the auction so that when you try to grab them in auction you should not pay the amount greater than you have in the purse for a particular player.

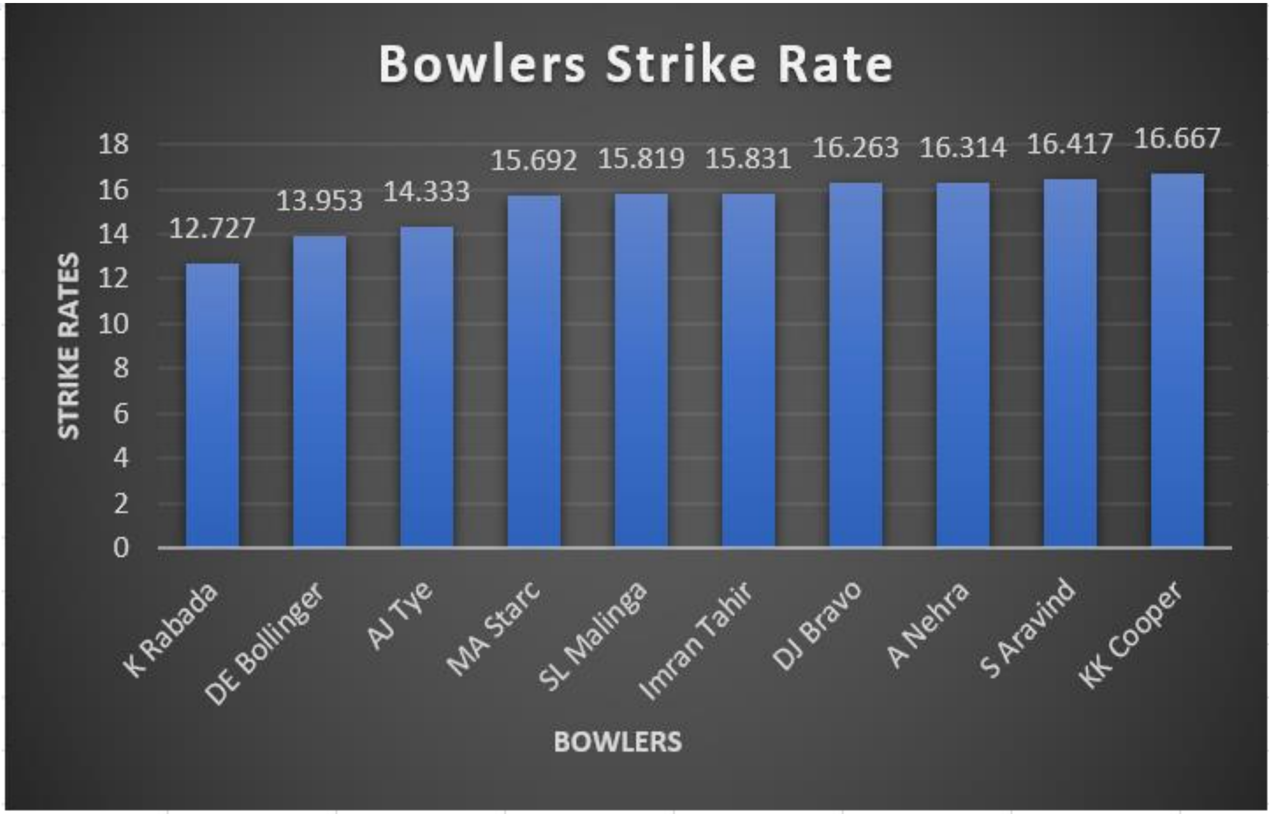
(strike rate of a bowler can be calculated by number of balls bowled divided by total wickets taken)

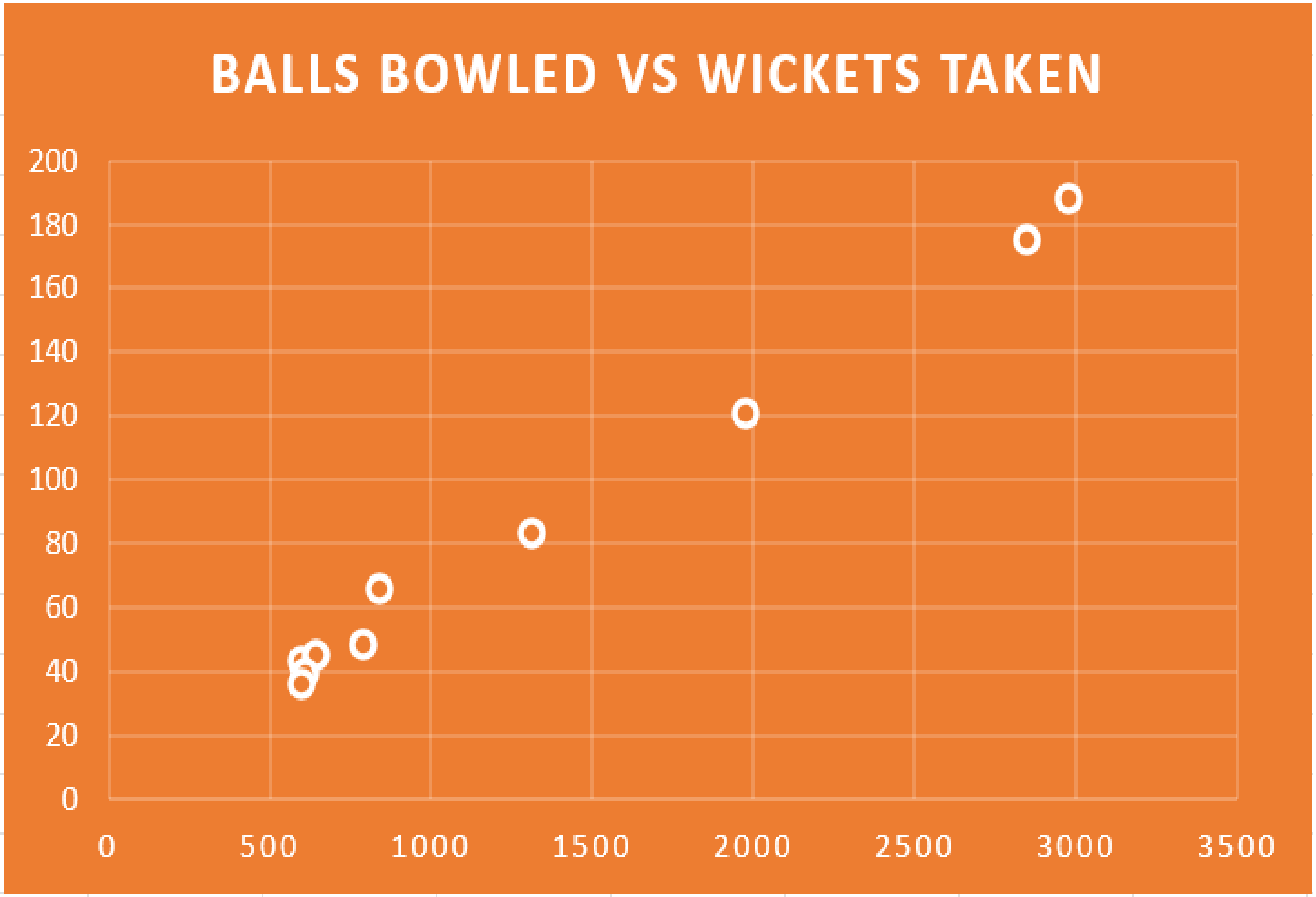
QUERY

```
SELECT bowler,
ROUND(CAST(COUNT(ball) AS
DECIMAL)/SUM(is_wicket),3) AS
bowler_strike_rate,
COUNT(ball) AS balls_bowled,
SUM(is_wicket) AS wickets_taken
FROM deliveries
GROUP BY bowler
HAVING COUNT(ball) >= 500 AND SUM(is_wicket)
!= 0
ORDER BY bowler_strike_rate
LIMIT 10;
```



bowler character varying (50)	bowler_strike_rate numeric	balls_bowled bigint	wickets_taken bigint
K Rabada	12.727	840	66
DE Bollinger	13.953	600	43
AJ Tye	14.333	645	45
MA Starc	15.692	612	39
SL Malinga	15.819	2974	188
Imran Tahir	15.831	1314	83
DJ Bravo	16.263	2846	175
A Nehra	16.314	1974	121
S Aravind	16.417	788	48
KK Cooper	16.667	600	36



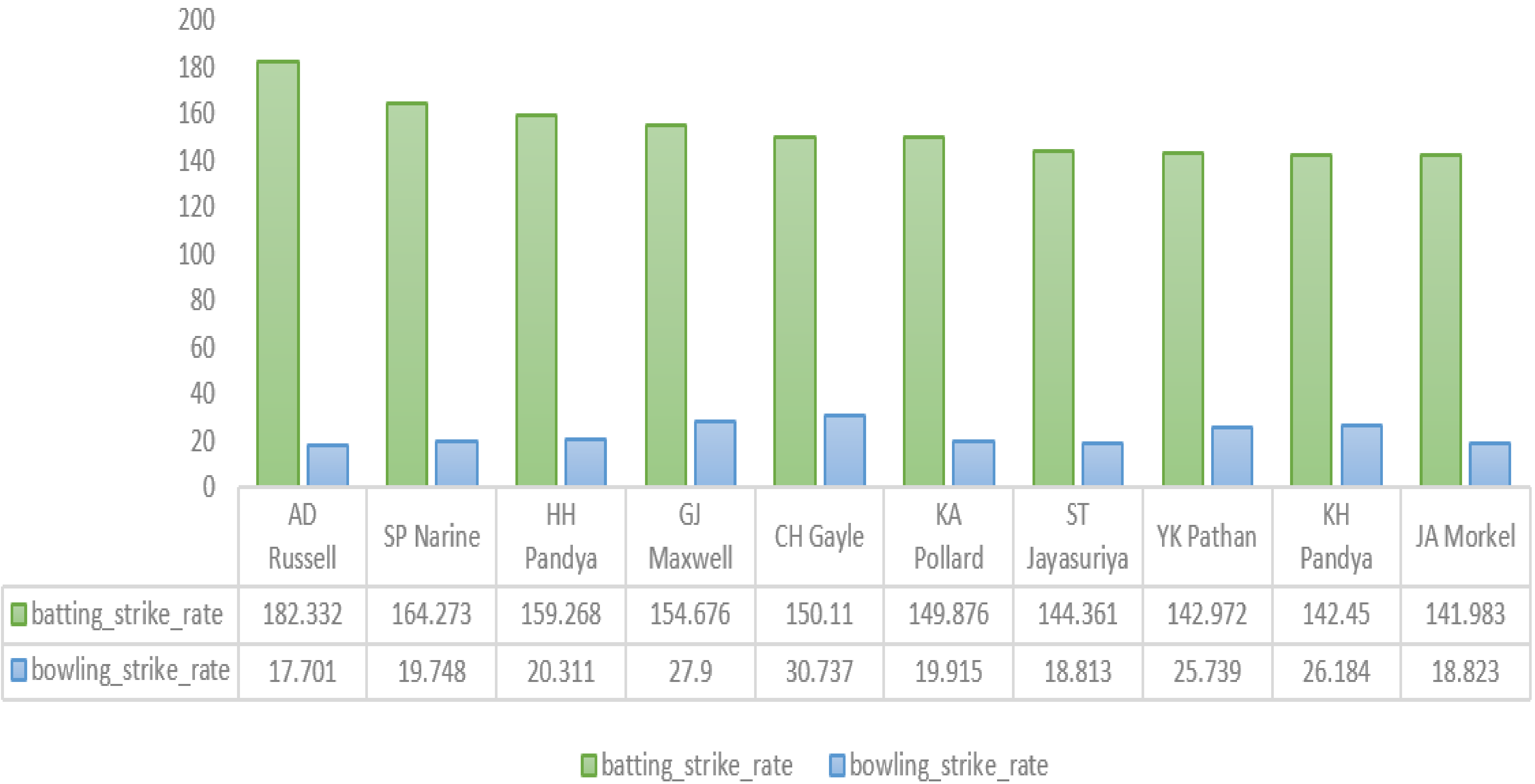

BALLS BOWLED VS WICKETS TAKEN
200
180
160
140
120
100
80
60
40
20
0
0
500
1000
1500
2000
2500
3000
3500

Q6 - Now you need to get 2-3 All_rounders with the best batting as well as bowling strike rate and who have faced at least 500 balls in IPL so far and have bowled minimum 300 balls.To do that you have to make a list of 10 players you want to bid in the auction so that when you try to grab them in auction you should not pay the amount greater than you have in the purse for a particular player.

(strike rate of an all rounder can be calculated using the same criteria of batsman similarly the bowling strike rate can be calculated using the criteria of a bowler)

QUERY

```
SELECT a.batsman AS all_rounder,
ROUND(SUM(a.batsman_runs)/CAST(COUNT(a.ball) AS DECIMAL)*100,3) AS batting_strike_Rate,
bowling_strike_rate
FROM deliveries AS a
INNER JOIN
(SELECT bowler, COUNT(ball) AS balls_bowled,
ROUND(CAST(COUNT(ball) AS DECIMAL)/SUM(is_wicket),3) AS bowling_strike_rate FROM
deliveries
GROUP BY bowler
HAVING COUNT(ball) >= 300 AND SUM(is_wicket) != 0
ORDER BY bowling_strike_rate) AS b
ON a.batsman = b.bowler
WHERE extras_type NOT IN ('wides')
GROUP BY batsman, bowling_strike_rate
HAVING COUNT(ball) >= 500
ORDER BY batting_strike_rate DESC
LIMIT 10;
```



	AD Russell	SP Narine	HH Pandya	GJ Maxwell	CH Gayle	KA Pollard	ST Jayasuriya	YK Pathan	KH Pandya	JA Morkel
🟩 batting_strike_rate	182.332	164.273	159.268	154.676	150.11	149.876	144.361	142.972	142.45	141.983
🟦 bowling_strike_rate	17.701	19.748	20.311	27.9	30.737	19.915	18.813	25.739	26.184	18.823

CRITERIA FOR WICKET KEEPER

- **A wicket keeper has to be agile and must have taken many wickets by stumping. In addition to this, he must be a hard hitter meaning that he should have hit lot of boundaries.**

- **Such a person could be a middle order batsman who can add runs to the score-card swiftly with boundaries.**

- **Thus to select wicket keepers, the number of stumpings done by player and his boundary percentage could be taken into account.**

- **The Wicket keeper can be identified from the given data by filtering the data with respect to the dismissal_kind as stumped.**

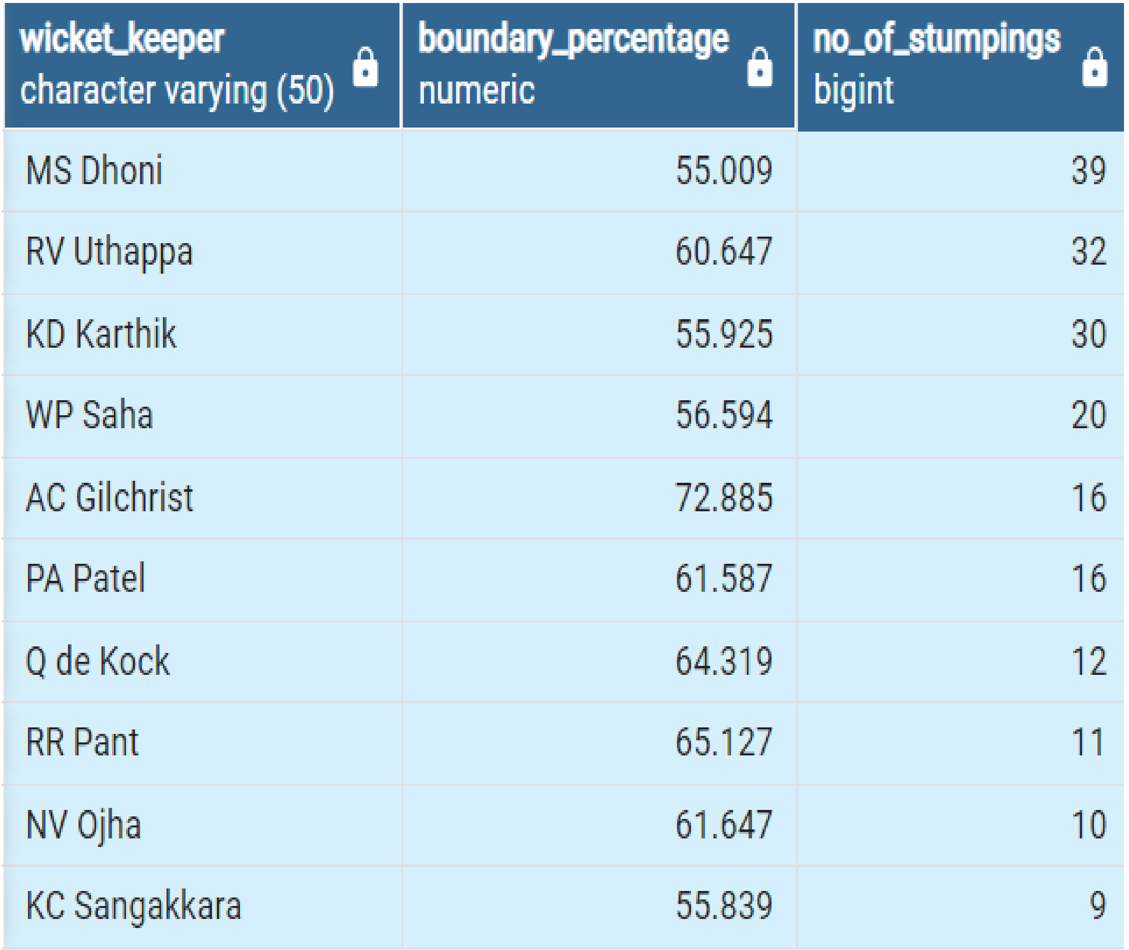
- **The stumping count could be grouped based on the wicket_keeper and sorted in descending order.**

- **In addition to this, the boundary percentage which is the proportion of runs which came from boundaries can be also included to get a better picture of their hitting abilities.**

SAMPLE QUERY

```
SELECT a.batsman AS wicket_keeper,
ROUND(SUM(CASE WHEN batsman_runs IN (4,6)
THEN batsman_runs ELSE 0
END)/CAST(SUM(batsman_runs) AS DECIMAL) *
100,3)
AS boundary_percentage, no_of_stumpings
FROM deliveries AS a
INNER JOIN
(SELECT fielder, COUNT(dismissal_kind) AS
no_of_stumpings FROM deliveries
WHERE dismissal_kind = 'stumped'
GROUP BY fielder
ORDER BY no_of_stumpings DESC) AS b
ON a.batsman = b.fielder
GROUP BY batsman, no_of_stumpings
HAVING SUM(batsman_runs) > 0
ORDER BY no_of_stumpings DESC;
```

SAMPLE OUTPUT TABLE

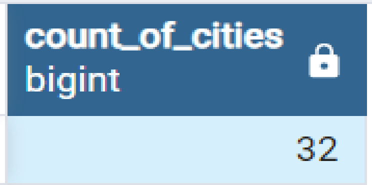


wicket_keeper character varying (50)	boundary_percentage numeric	no_of_stumpings bigint
MS Dhoni	55.009	39
RV Uthappa	60.647	32
KD Karthik	55.925	30
WP Saha	56.594	20
AC Gilchrist	72.885	16
PA Patel	61.587	16
Q de Kock	64.319	12
RR Pant	65.127	11
NV Ojha	61.647	10
KC Sangakkara	55.839	9

ADDITIONAL QUESTIONS FOR FINAL ASSESSMENT

1. Get the count of cities that have hosted an IPL match

```
SELECT COUNT(DISTINCT(city)) AS count_of_cities
FROM matches
WHERE city != 'NA';
```



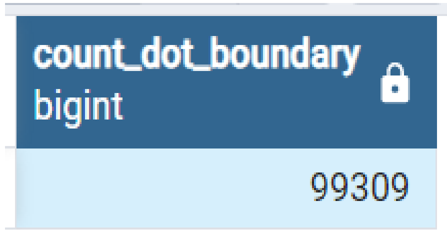
count_of_cities bigint
32

2. Create table *deliveries_v02* with all the columns of the table '*deliveries'* and an additional column *ball_result* containing values *boundary*, *dot* or *other* depending on the *total_run* (boundary for >= 4, dot for 0 and other for any other number)
(Hint 1 : CASE WHEN statement is used to get condition based results)
(Hint 2: To convert the output data of the select statement into a table, you can use a subquery. Create table *table_name* as *[entire select statement].*

```
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 deliveries;

SELECT * FROM deliveries_v02;
```

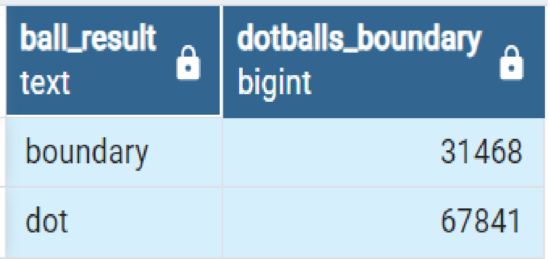
3. Write a query to fetch the total number of boundaries and dot balls from the *deliveries_v02* table.

```
SELECT COUNT(ball_result) AS
count_dot_boundary
FROM deliveries_v02
WHERE ball_result IN ('dot', 'boundary');
```



count_dot_boundary bigint
99309

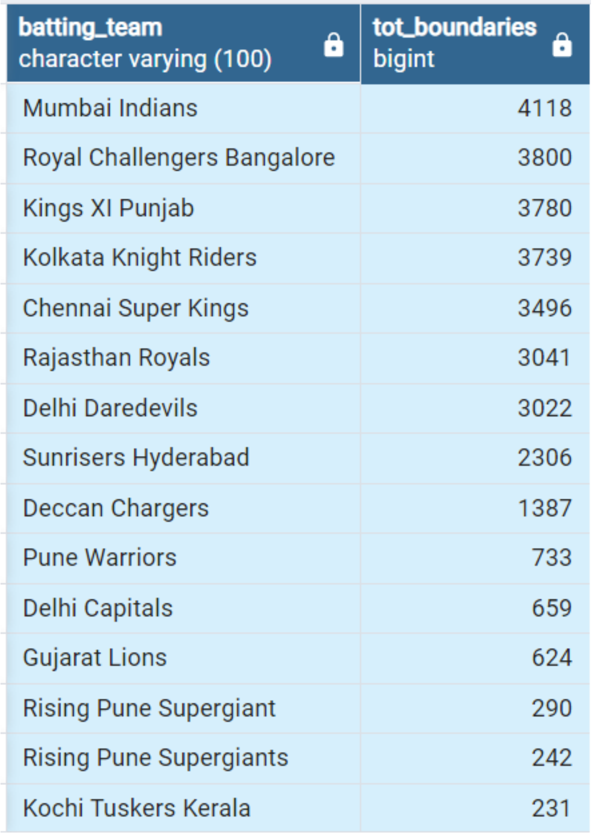
```
SELECT ball_result, COUNT(ball_result) as
dotballs_boundary
FROM deliveries_v02
WHERE ball_result IN ('boundary', 'dot')
GROUP BY ball_result;
```



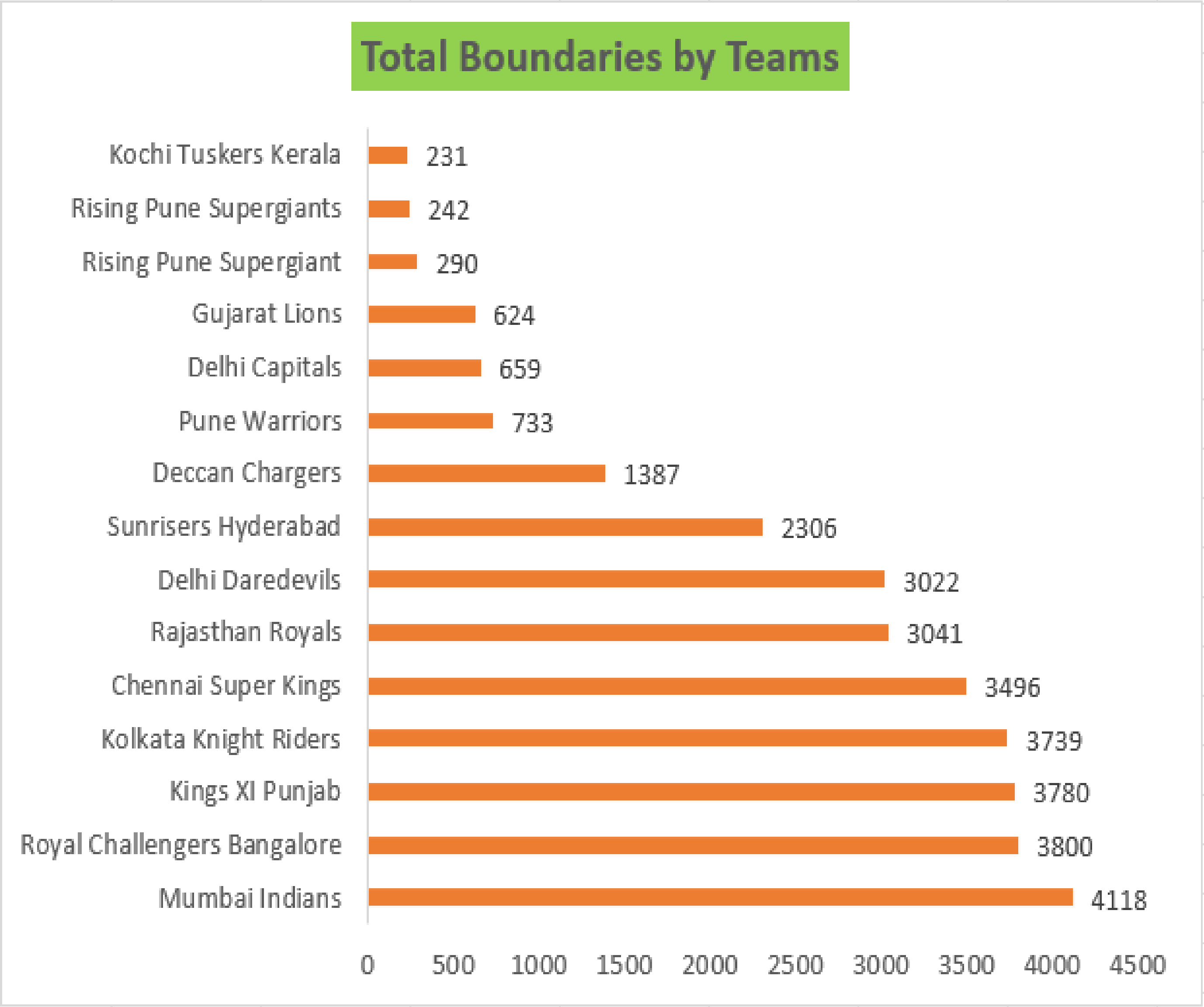
ball_result text	dotballs_boundary bigint
boundary	31468
dot	67841

4.Write a query to fetch the total number of boundaries scored by each team from the *deliveries_v02* table and order it in descending order of the number of boundaries scored.

```
SELECT batting_team, COUNT(ball_result) AS tot_boundaries
FROM deliveries_v02
WHERE ball_result = 'boundary'
GROUP BY batting_team
ORDER BY tot_boundaries DESC;
```

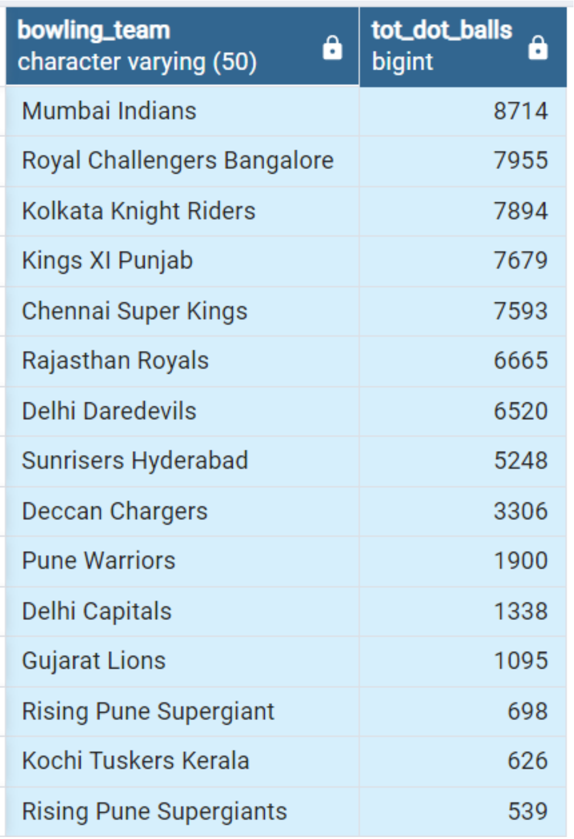


batting_team character varying (100)	tot_boundaries bigint
Mumbai Indians	4118
Royal Challengers Bangalore	3800
Kings XI Punjab	3780
Kolkata Knight Riders	3739
Chennai Super Kings	3496
Rajasthan Royals	3041
Delhi Daredevils	3022
Sunrisers Hyderabad	2306
Deccan Chargers	1387
Pune Warriors	733
Delhi Capitals	659
Gujarat Lions	624
Rising Pune Supergiant	290
Rising Pune Supergiants	242
Kochi Tuskers Kerala	231

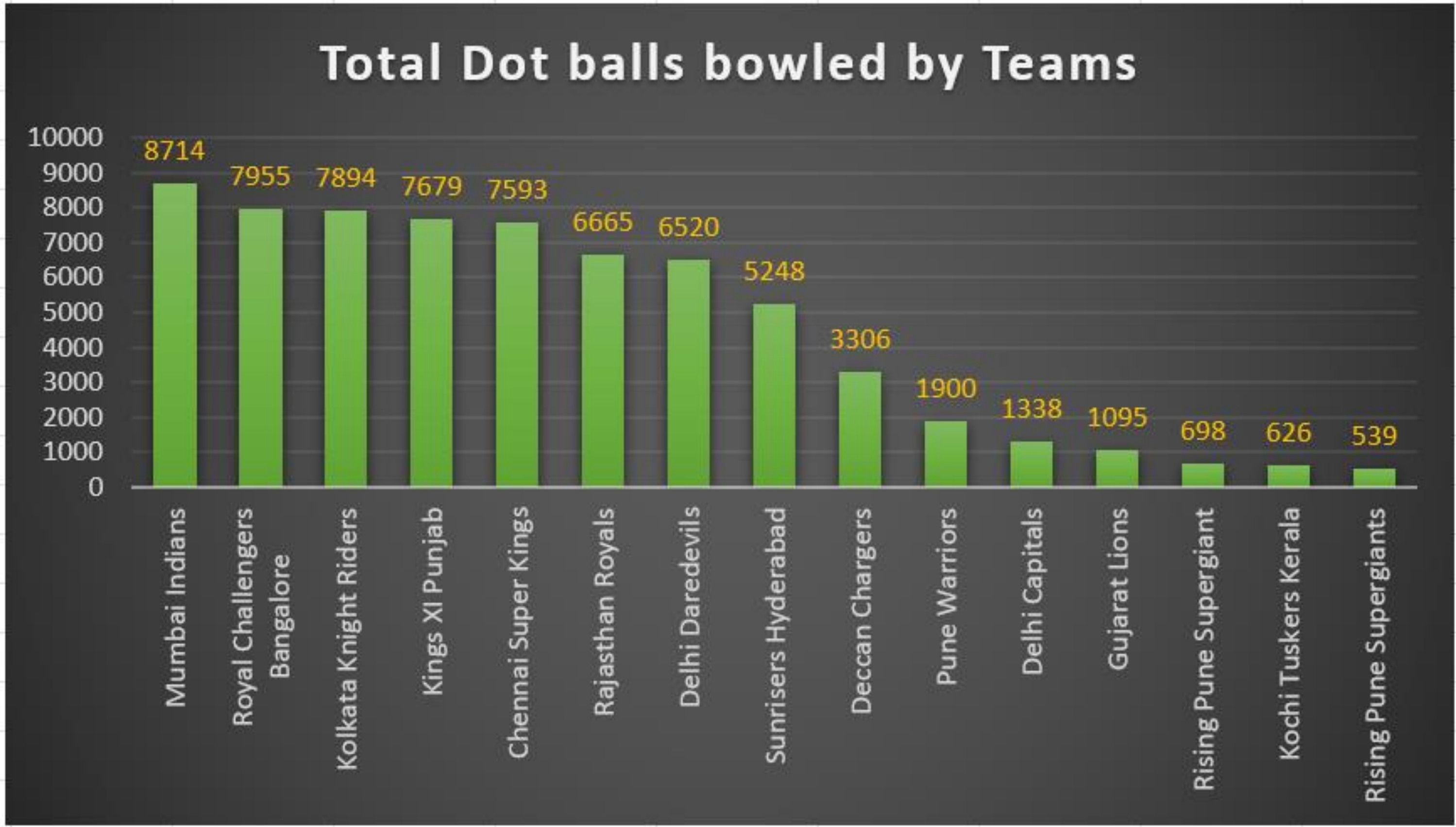


5. Write a query to fetch the total number of dot balls bowled by each team and order it in descending order of the total number of dot balls bowled.

```
SELECT bowling_team, COUNT(ball_result) AS tot_dot_balls
FROM deliveries_v02
WHERE ball_result = 'dot' AND bowling_team != 'NA'
GROUP BY bowling_team
ORDER BY tot_dot_balls DESC;
```

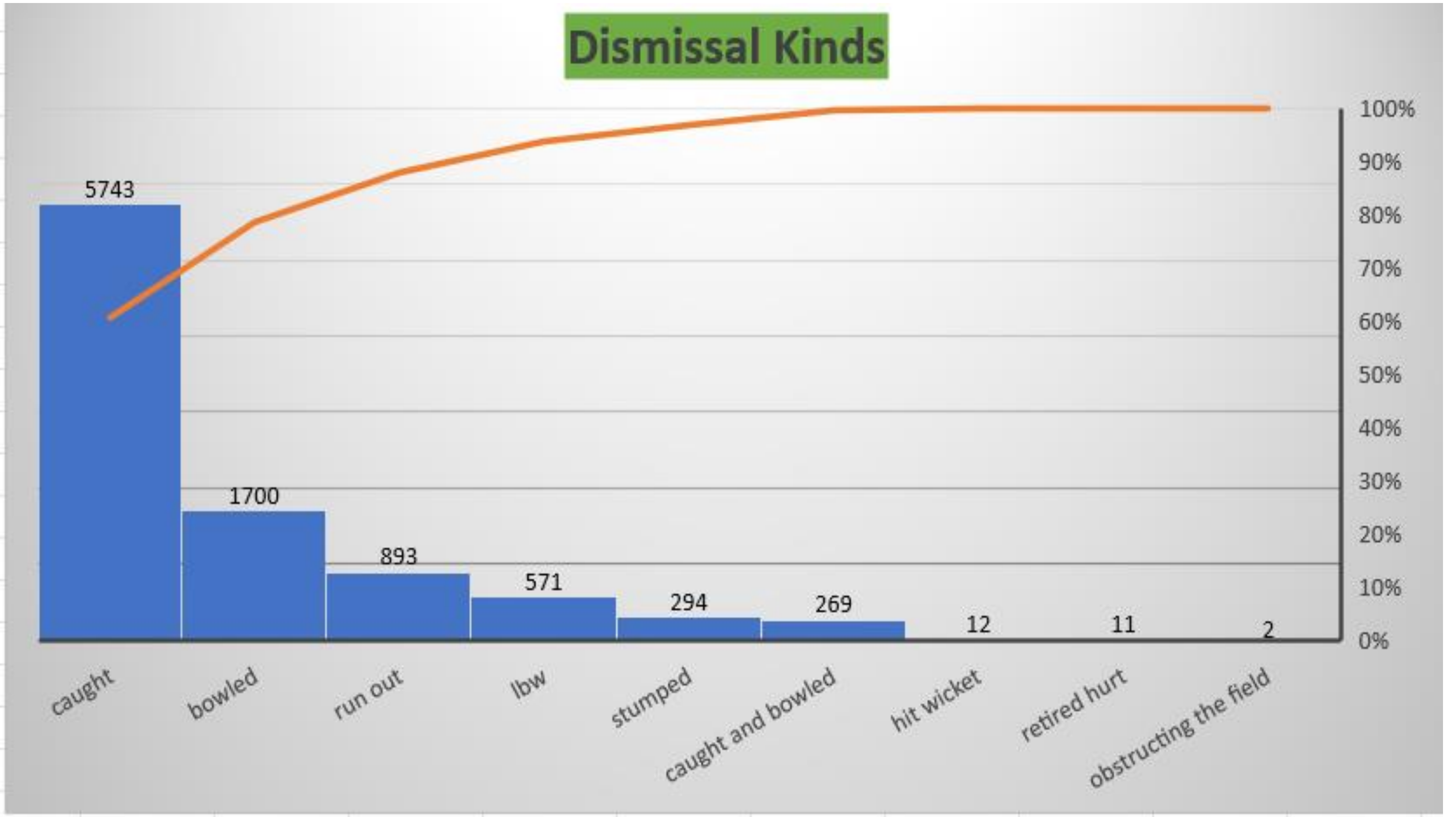


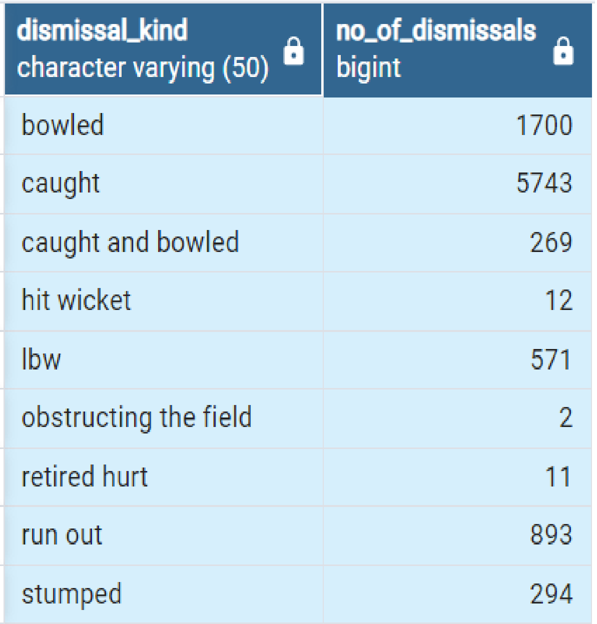
bowling_team character varying (50) 🔒	tot_dot_balls bigint 🔒
Mumbai Indians	8714
Royal Challengers Bangalore	7955
Kolkata Knight Riders	7894
Kings XI Punjab	7679
Chennai Super Kings	7593
Rajasthan Royals	6665
Delhi Daredevils	6520
Sunrisers Hyderabad	5248
Deccan Chargers	3306
Pune Warriors	1900
Delhi Capitals	1338
Gujarat Lions	1095
Rising Pune Supergiant	698
Kochi Tuskers Kerala	626
Rising Pune Supergiants	539



6. Write a query to fetch the total number of dismissals by dismissal kinds where dismissal kind is not NA.

```
SELECT dismissal_kind, COUNT(dismissal_kind) as
no_of_dismissals
FROM deliveries_v02
WHERE dismissal_kind != 'NA'
GROUP BY dismissal_kind;
```

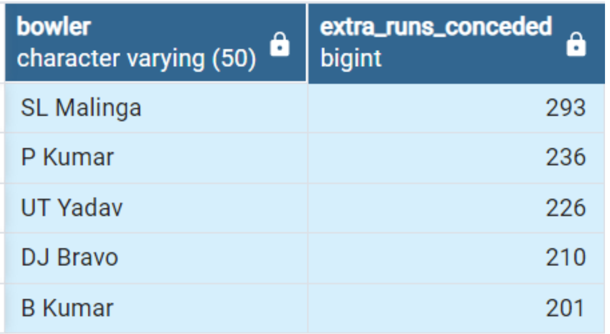




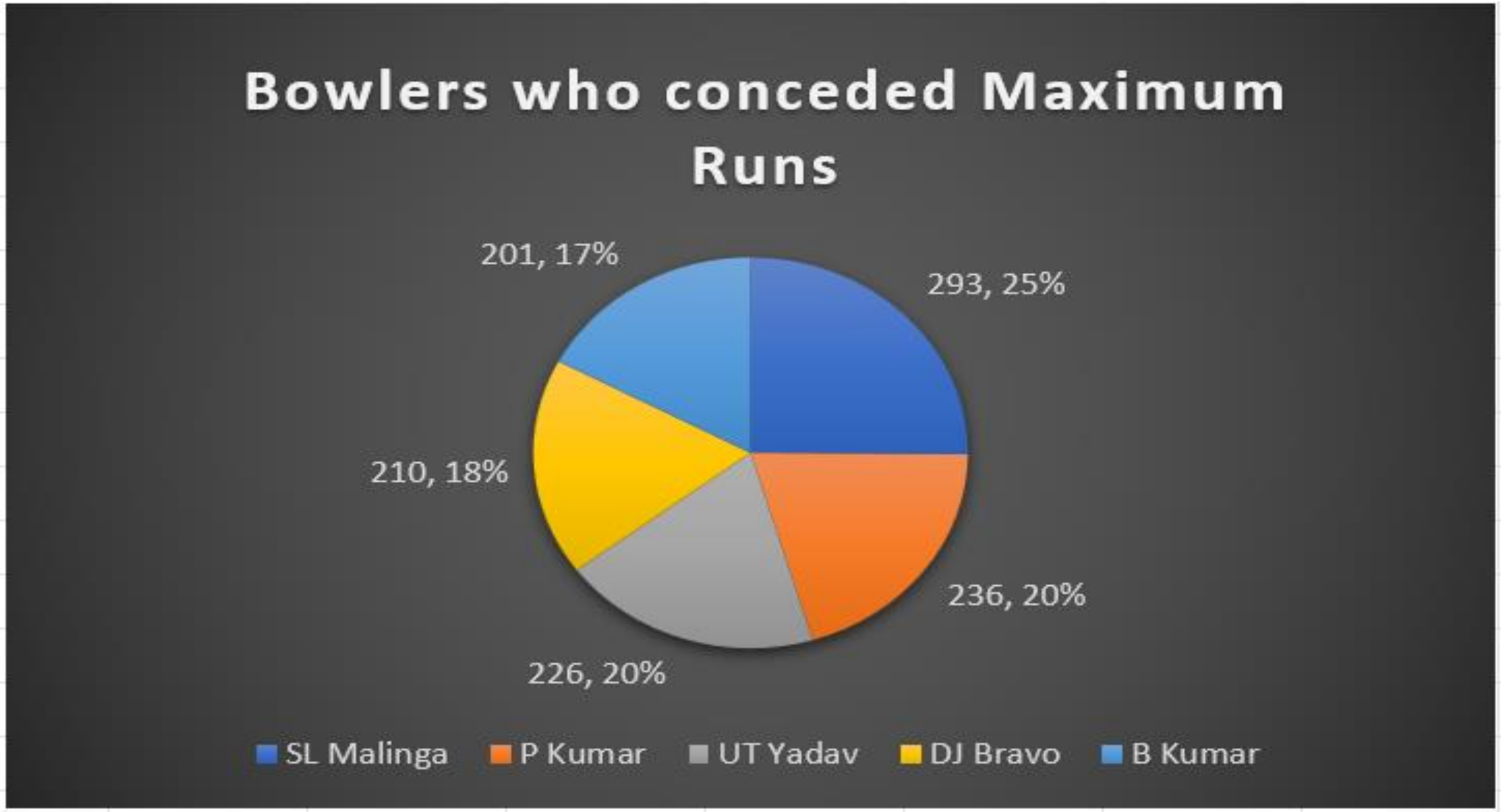
dismissal_kind character varying (50)	no_of_dismissals bigint
bowled	1700
caught	5743
caught and bowled	269
hit wicket	12
lbw	571
obstructing the field	2
retired hurt	11
run out	893
stumped	294

7. Write a query to get the top 5 bowlers who conceded maximum extra runs from the *deliveries* table

```
SELECT bowler, SUM(extra_runs) AS
extra_runs_conceded
FROM deliveries
GROUP BY bowler
ORDER BY extra_runs_conceded DESC
LIMIT 5;
```



bowler character varying (50)	extra_runs_conceded bigint
SL Malinga	293
P Kumar	236
UT Yadav	226
DJ Bravo	210
B Kumar	201

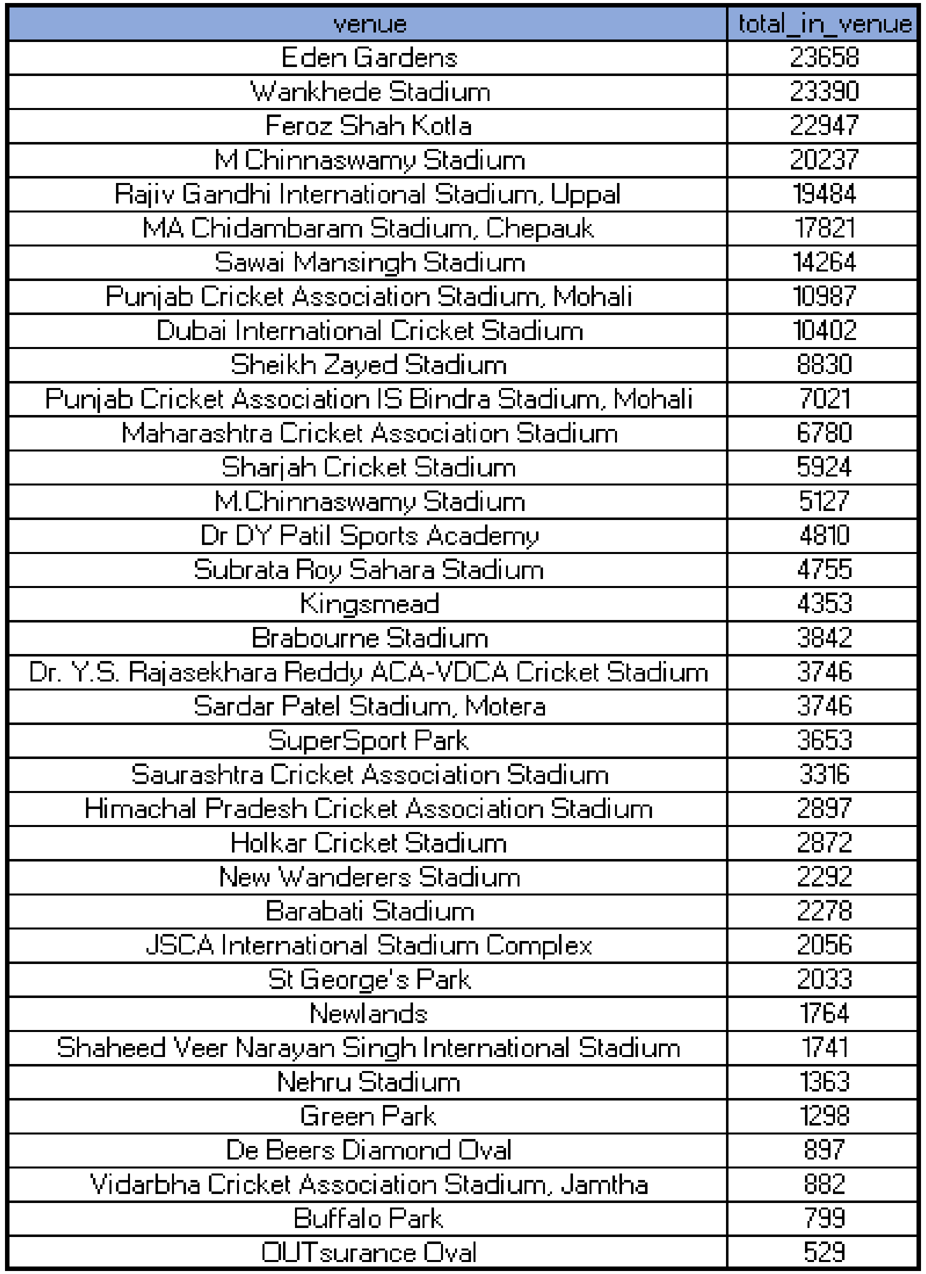


8. Write a query to create a table named *deliveries_v03* with all the columns of *deliveries_v02* table and two additional column (named *venue* and *match_date*) of *venue* and *date* from table *matches*

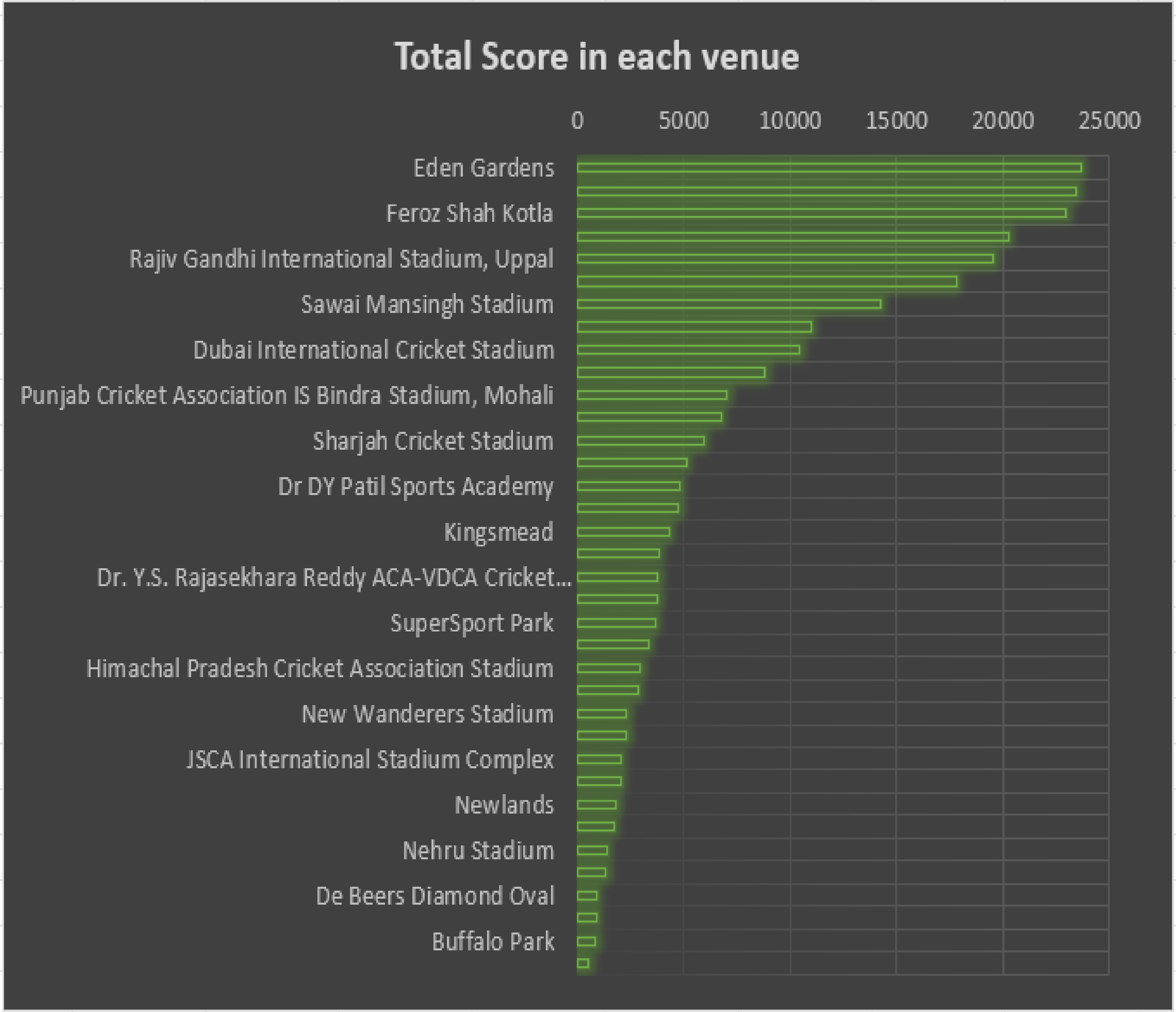
```
CREATE TABLE deliveries_v03 AS
SELECT a.*, b.venue, b.date AS
match_date FROM deliveries_v02 AS a
LEFT JOIN matches as b
ON a.id = b.id;

SELECT * FROM deliveries_v03;
```

9. Write a query to fetch the total runs scored for each venue and order it in the descending order of total runs scored.

```
SELECT venue, SUM(total_runs)
AS total_in_venue
FROM deliveries_v03
GROUP BY venue
ORDER BY total_in_venue DESC;
```

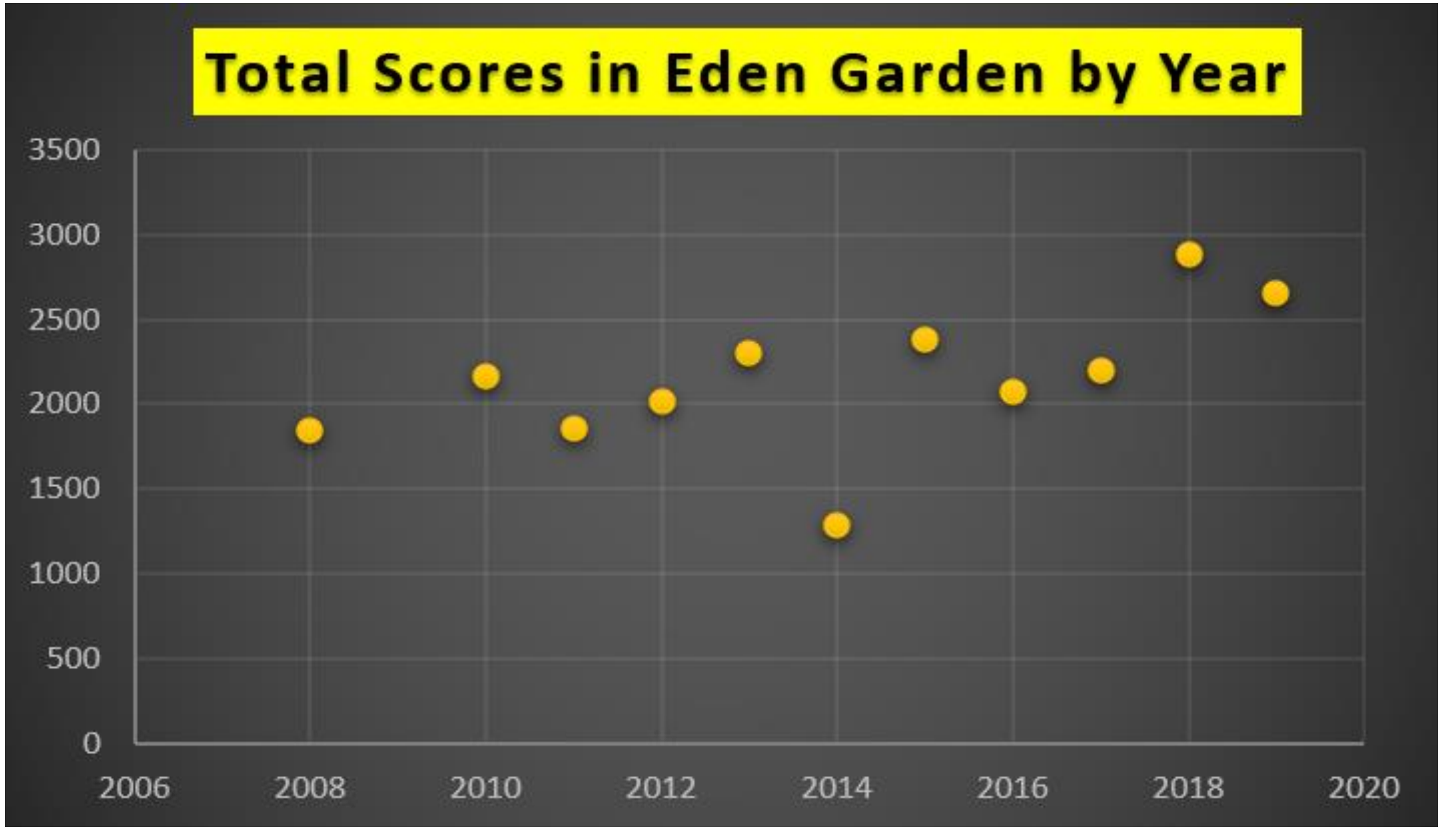


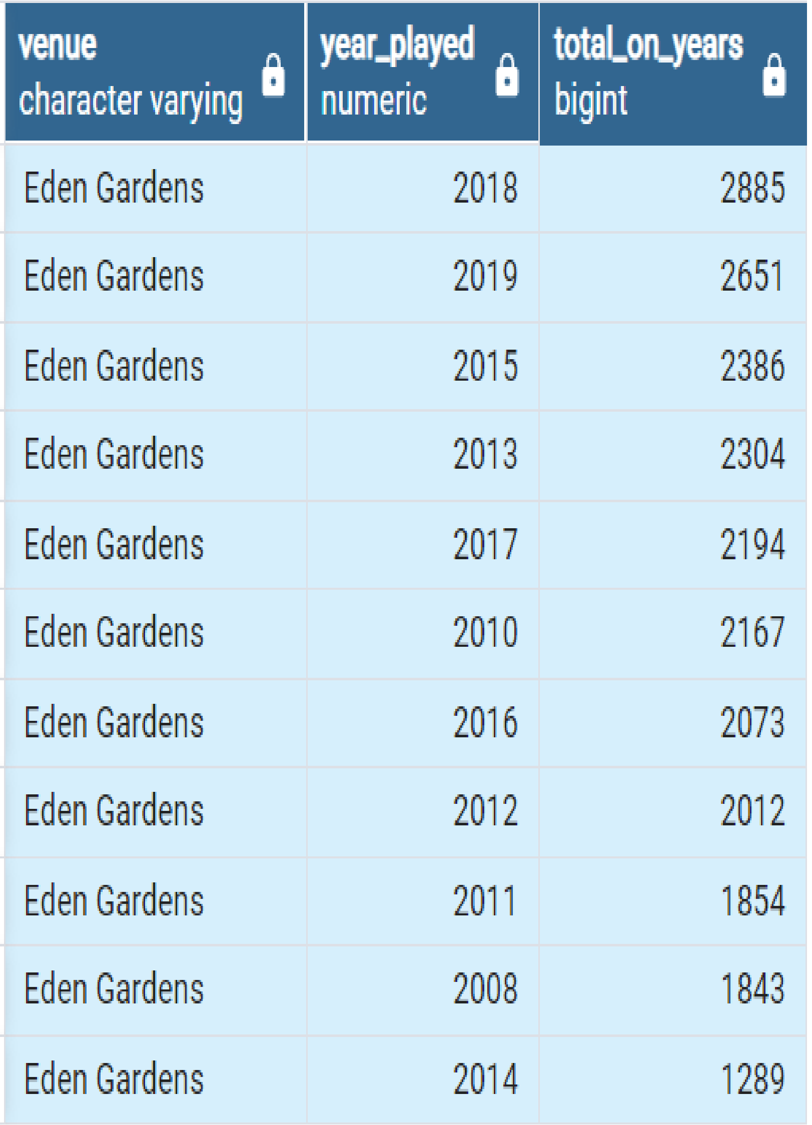
venue	total_in_venue
Eden Gardens	23658
Wankhede Stadium	23390
Feroz Shah Kotla	22947
M Chinnaswamy Stadium	20237
Rajiv Gandhi International Stadium, Uppal	19484
MA Chidambaram Stadium, Chepauk	17821
Sawai Mansingh Stadium	14264
Punjab Cricket Association Stadium, Mohali	10987
Dubai International Cricket 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



10. Write a query to fetch the year-wise total runs scored at *Eden Gardens* and order it in the descending order of total runs scored.

```
SELECT venue, EXTRACT(YEAR FROM
match_date) AS year_played, SUM(total_runs)
AS total_on_years
FROM deliveries_v03
WHERE venue = 'Eden Gardens'
GROUP BY venue, year_played
ORDER BY total_on_years DESC;
```





venue character varying	year_played numeric	total_on_years bigint
Eden Gardens	2018	2885
Eden Gardens	2019	2651
Eden Gardens	2015	2386
Eden Gardens	2013	2304
Eden Gardens	2017	2194
Eden Gardens	2010	2167
Eden Gardens	2016	2073
Eden Gardens	2012	2012
Eden Gardens	2011	1854
Eden Gardens	2008	1843
Eden Gardens	2014	1289