IPL Auction

Name: Prerna Rai
Course: Data Science
Project: IPL Auction

In the same folder I attached
a text file where
all queries present , this pptx is
for presentations

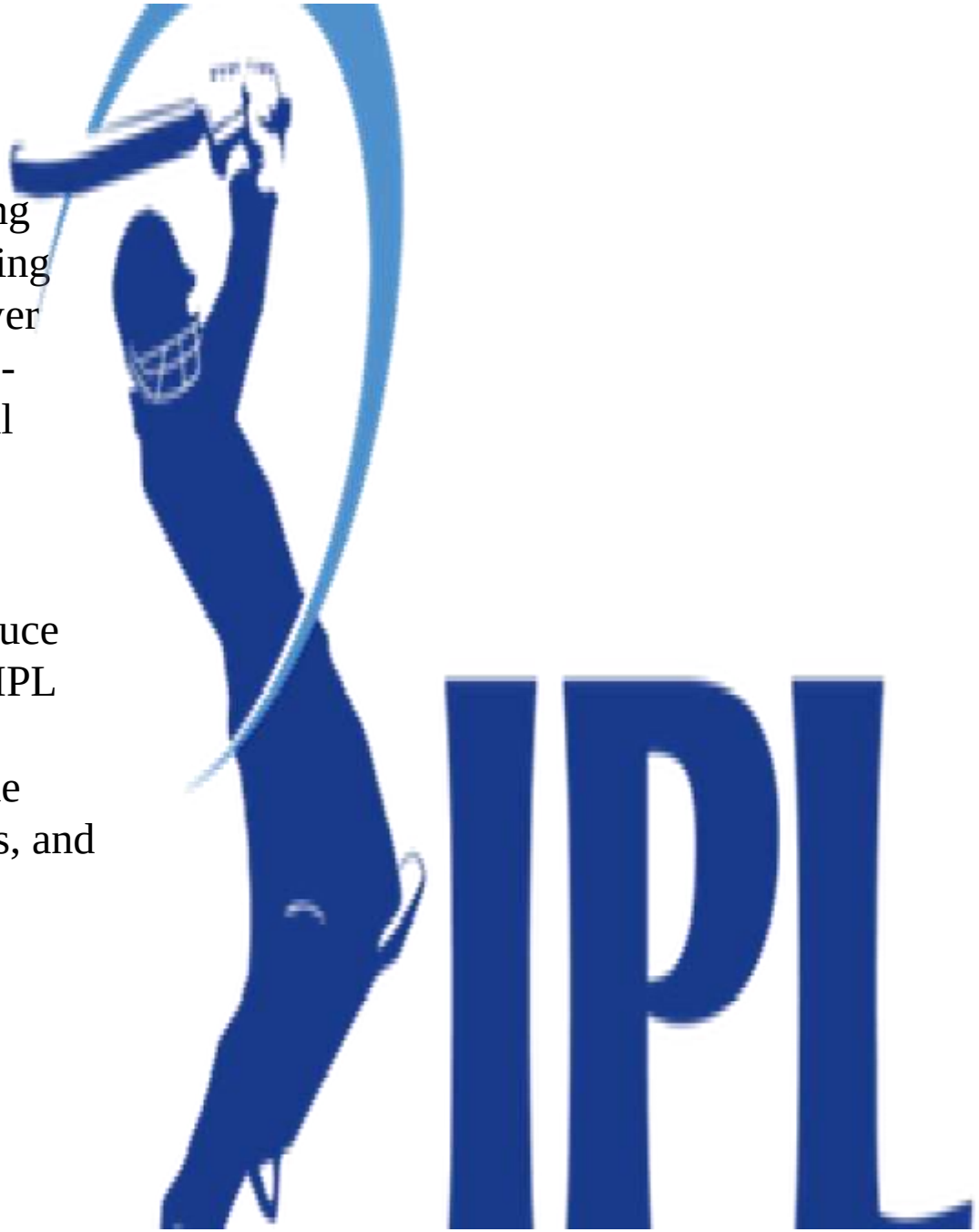
Agenda:

1. Brief overview of the project
2. Importance of the IPL auction strategy
3. Data Preparation
4. SQL queries
5. Additional Questions Queries
6. Wicketkeeper Criteria
7. List of all the players for auction

Brief overview of the project

It appears that you have a well-thought-out plan in place for getting ready for the IPL auction by studying historical data and formulating plans to assemble a potent and well-rounded team. The many player types that are required have been delineated by you, including big-hitters, finishers, aggressive batsmen, anchor batsmen, economical bowlers, wicket-taking bowlers, all-rounders, and wicketkeepers.

Furthermore, you have given the team management thorough instructions on how to choose players, evaluate the data, and produce visual aids to be presented before to the auction. A wide range of IPL data analysis-related subjects are covered in the supplementary questions for the final evaluation. Examples of these topics include classifying deliveries, counting the cities that have hosted matches, and retrieving certain information from the database.



Importance of the IPL auction strategy

Among the main justifications for the significance of the IPL auction strategy are:

Putting together a well-rounded team requires a fair balance between seasoned players and up-and-coming talent. Teams can find players who compliment one other's talents and fit certain roles by using a strong auction strategy.

Budget optimization: Every club has a certain amount of money to spend at the auction, and it's critical to manage this money wisely in order to sign the top players for the group. Teams may make well-informed selections about which players to pursue and at what price with the aid of a clever auction strategy.

Finding important players: Teams can find and sign important players who have the potential to have a big impact on their performance during the auction. Teams can target players who could be game-changers in the event with the aid of a solid auction plan.

Data Preparation

Tools and technologies used:

Relational databases are managed and altered using a programming language called SQL, or Structured Query Language. In addition to enabling users to add, update, delete, and retrieve data, it also lets them see and alter data that is stored in a database. Working with big and complicated datasets requires the usage of SQL, which is extensively utilized in the database administration industry.
Because SQL is a declarative language, users define the data they wish to access or alter rather than defining the method. This makes SQL, especially for those with no programming knowledge, quite simple to learn and use.

Data description language (DDL), data manipulation language (DML), and data control language (DCL) are the fundamental building blocks of SQL. DDL is used to specify the database's structure, including the creation of tables, constraints, and indexes. DML is used to add, update, and remove entries among other manipulations of data kept in databases. Permissions on tables and other objects can be granted or revoked using DCL, which is used to manage database access.

Sql queries:[table4 represent IPL_Ball Data]

Step-by-Step Approach:

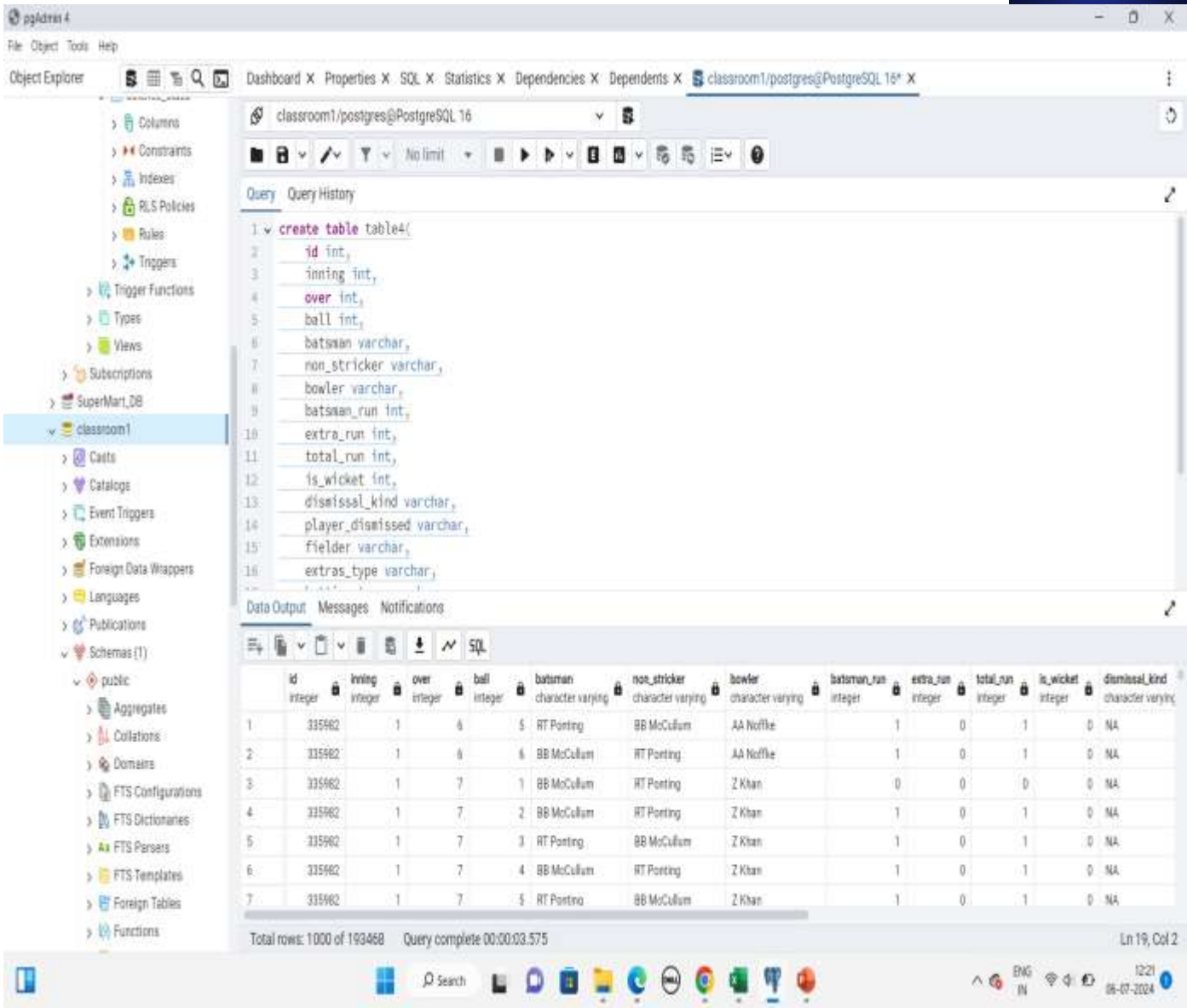
First, you need to set up your SQL Server and create the necessary tables to store IPL data. Based on the project requirements.

IPL_Ball.csv

```
create table table4(  id int, inning int, over int, ball int, batsman varchar, non_stricker varchar, bowler varchar,   batsman_run int, extra_run int, total_run int, is_wicket int, dismissal_kind varchar,player_dismissed varchar, fielder varchar, extras_type varchar, batting_team varchar, bowling_team varchar);

copy table4 from 'C:\Program Files\PostgreSQL\16\IPL_Ball.csv' delimiter ',' csv header;

select*from table4;
```



In the same folder I attached a text file where all queries present , this pptx is for presentations

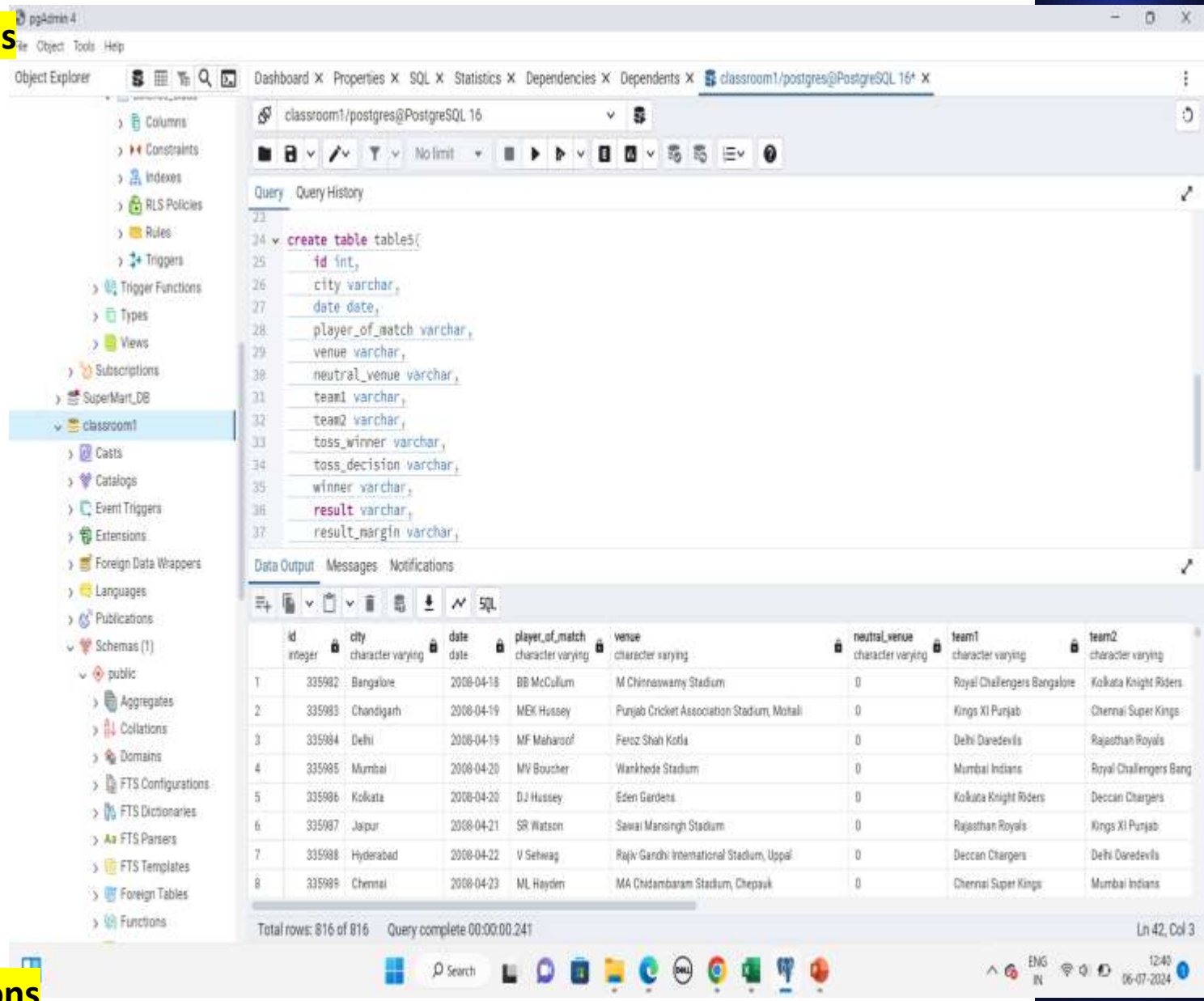
IPL_matches.csv [table5 represent ipl_matches data

```
create table table5( id int, city
varchar,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
varchar, eliminator varchar, method
varchar, umpire1 varchar, umpire2
varchar);

copy table5 from 'C:\Program
Files\PostgreSQL\16\IPL_matches.csv'
delimiter ',' csv header;

select*from table5;
```

In the same folder I attached a text file where all queries present , this pptx is for presentations

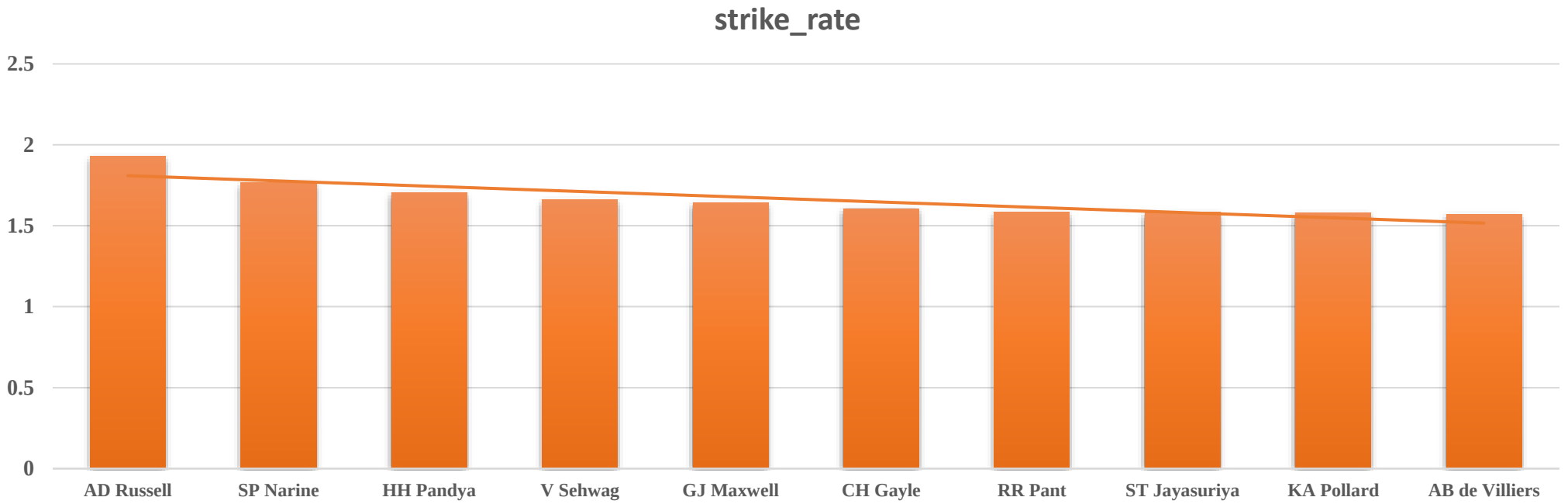


Bidding on Batsman: [table4 represent IPL_Ball Data]

Aggressive Batters (High Strike Rate):

```
SELECT batsman as player, (SUM(total_run) * 1.0 /
COALESCE(COUNT(CASE WHEN extras_type != 'wides'
THEN table4.ball ELSE NULL END), 1)) AS strike_rate
FROM table4 GROUP BY batsman HAVING
COUNT(CASE WHEN extras_type != 'wides' THEN
table4.ball ELSE NULL END) >= 500 ORDER BY
strike_rate DESC LIMIT 10;
```

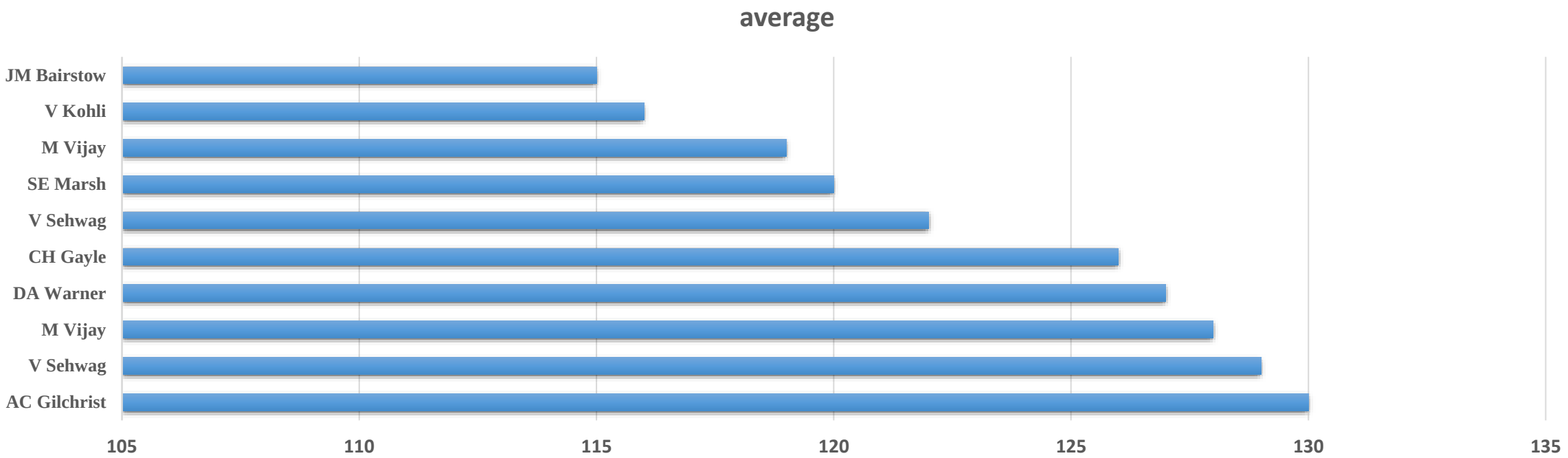
	player character varying	strike_rate numeric
1	AD Russell	1.9278846153846154
2	SP Narine	1.7661141804788214
3	HH Pandya	1.7048406139315230
4	V Sehwag	1.6609686609686610
5	GJ Maxwell	1.6402877697841727
6	CH Gayle	1.6052217678515256
7	RR Pant	1.5847953216374269
8	ST Jayasuriya	1.5845864661654135
9	KA Pollard	1.5805651958353991
10	AB de Villiers	1.5714285714285714



Anchor Batters (Good Average):

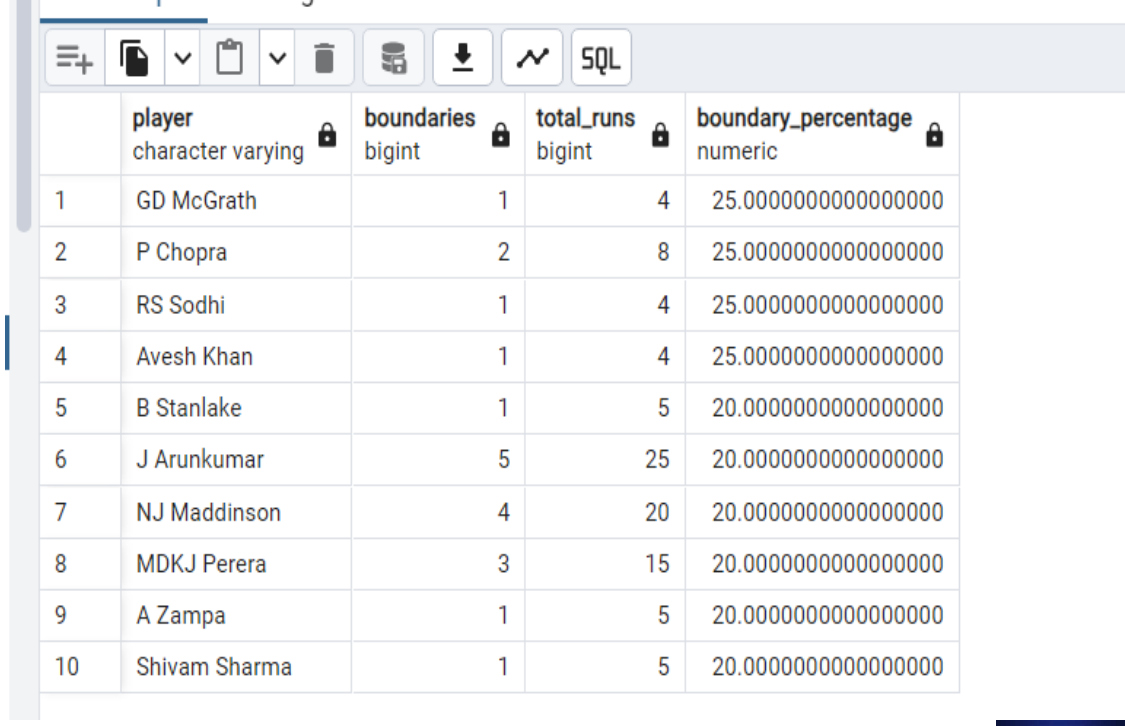
```
SELECT id, batsman, (SUM(total_run) * 1.0 /
COUNT(CASE WHEN is_wicket = 1 THEN 1 ELSE
NULL END)) AS average     FROM table4
 GROUP BY id, batsman
HAVING COUNT(CASE WHEN is_wicket= 1 THEN 1
ELSE NULL END) > 0     ORDER BY average DESC
limit 10;
```

	id integer	batsman character varying	average numeric
1	501260	AC Gilchrist	130.0000000000000000
2	734047	V Sehwag	129.0000000000000000
3	419137	M Vijay	128.0000000000000000
4	1082627	DA Warner	127.0000000000000000
5	829785	CH Gayle	126.0000000000000000
6	501243	V Sehwag	122.0000000000000000
7	336019	SE Marsh	120.0000000000000000
8	548380	M Vijay	119.0000000000000000
9	980999	V Kohli	116.0000000000000000
10	1175366	JM Bairstow	115.0000000000000000



Hard-hitting Batters (Boundary Percentage):

```
SELECT batsman AS player,SUM(CASE WHEN total_run = 4
THEN 1 ELSE 0 END) + SUM(CASE WHEN total_run = 6 THEN
1 ELSE 0 END) AS boundaries,SUM(total_run) AS
total_runs,CASE WHEN SUM(total_run) = 0 THEN 0 ELSE
(SUM(CASE WHEN total_run= 4 THEN 1 ELSE 0 END) +
SUM(CASE WHEN total_run= 6 THEN 1 ELSE 0 END)) * 100.0
/ SUM(total_run) END AS boundary_percentage FROM
table4 WHERE batsman IS NOT NULL GROUP BY batsman
ORDER BY boundary_percentage DESC
LIMIT 10;
```



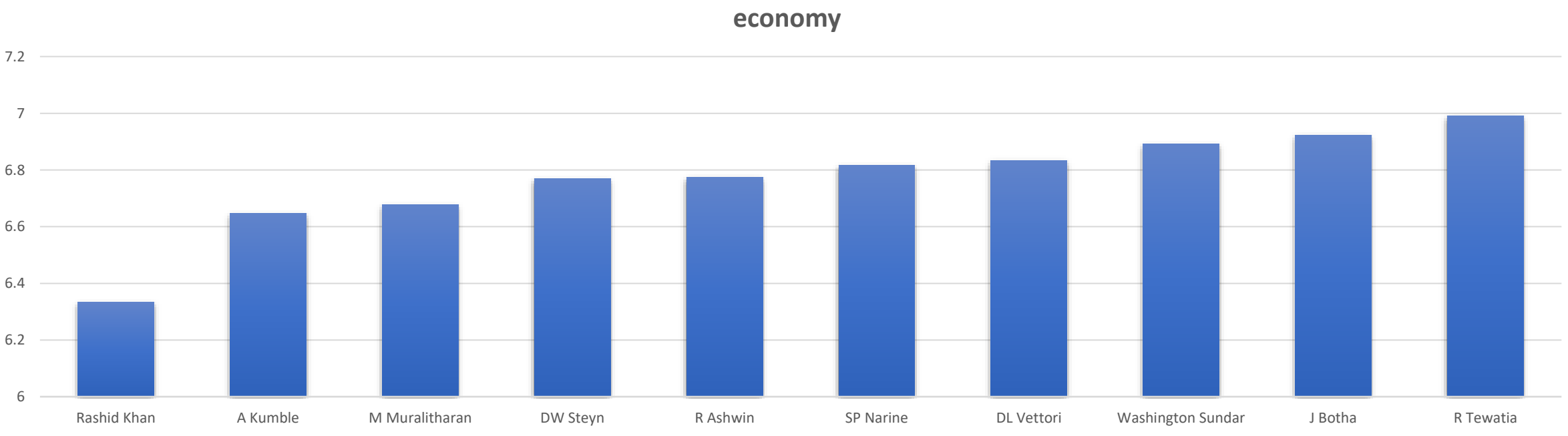
	player character varying	boundaries bigint	total_runs bigint	boundary_percentage numeric
1	GD McGrath	1	4	25.0000000000000000
2	P Chopra	2	8	25.0000000000000000
3	RS Sodhi	1	4	25.0000000000000000
4	Avesh Khan	1	4	25.0000000000000000
5	B Stanlake	1	5	20.0000000000000000
6	J Arunkumar	5	25	20.0000000000000000
7	NJ Maddinson	4	20	20.0000000000000000
8	MDKJ Perera	3	15	20.0000000000000000
9	A Zampa	1	5	20.0000000000000000
10	Shivam Sharma	1	5	20.0000000000000000

Bidding on Bowlers:

Economical Bowlers:

```
SELECT bowler,SUM(total_run) / (COUNT(ball)
/ 6.0) AS economy,COUNT(ball) AS total_balls
FROM table4 WHERE bowler IS NOT NULL
GROUP BY bowler HAVING COUNT(ball) >=
500 ORDER BY economy ASC
LIMIT 10;
```

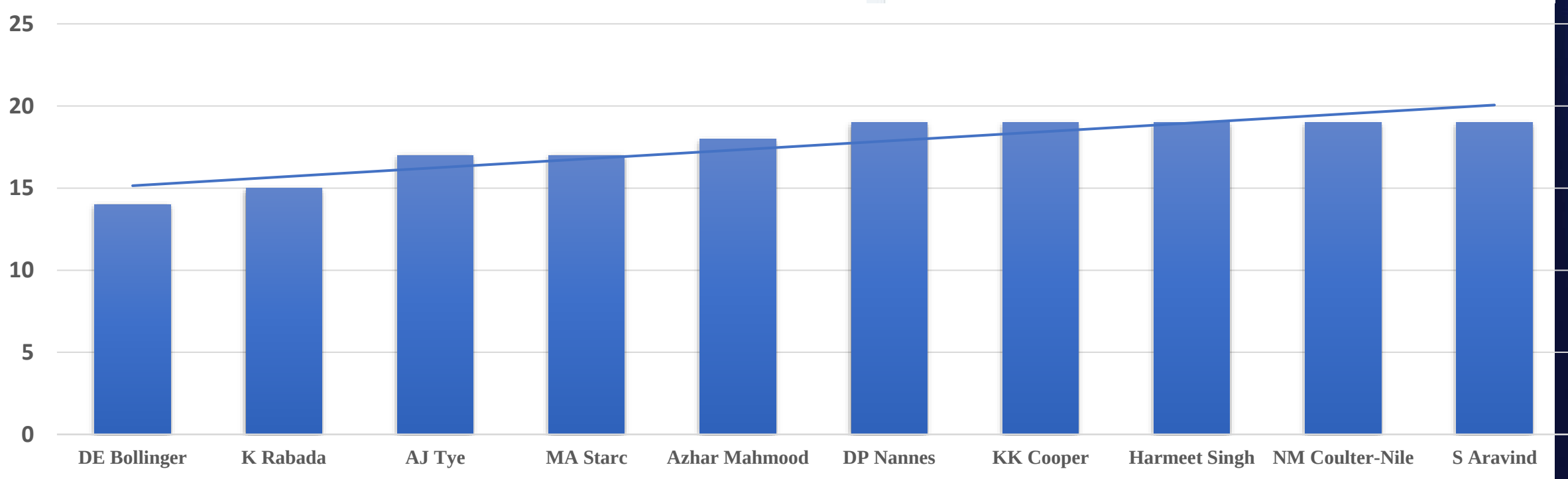
	bowler character varying	economy numeric	total_balls bigint
1	Rashid Khan	6.3342281879194631	1490
2	A Kumble	6.6469989827060020	983
3	M Muralitharan	6.6772352568167406	1577
4	DW Steyn	6.7697715289982425	2276
5	R Ashwin	6.7736699729486023	3327
6	SP Narine	6.8158640226628895	2824
7	DL Vettori	6.8331210191082803	785
8	Washington Sundar	6.8909090909090909	660
9	J Botha	6.9224259520451340	709
10	R Tewatia	6.9914821124361158	587



Wicket-taking Bowlers (Best Strike Rate):

```
SELECT bowler,COUNT(ball) / COUNT(DISTINCT
player_dismissed) AS strike_rate,COUNT(ball) AS
total_balls FROM table4 WHERE bowler IS NOT NULL
GROUP BY bowler HAVING COUNT(ball) >= 500 ORDER BY
strike_rate ASC LIMIT 10;
```

	bowler character varying	strike_rate bigint	total_balls bigint
1	DE Bollinger	14	600
2	K Rabada	15	840
3	AJ Tye	17	645
4	MA Starc	17	612
5	Azhar Mahmood	18	552
6	DP Nannes	19	689
7	KK Cooper	19	600
8	Harmeet Singh	19	549
9	NM Coulter-Nile	19	751
10	S Aravind	19	788



All-rounders

Best Batting and Bowling Strike Rate:

```
SELECT batsman, (SUM(total_run) * 1.0 /
COALESCE(COUNT(CASE WHEN extras_type != 'wides' THEN
table4.ball ELSE NULL END), 1)) AS batting_strike_rate,
(COUNT(table4.ball) / COUNT(DISTINCT player_dismissed)) AS
bowling_strike_rate FROM table4 WHERE batsman IS NOT NULL
GROUP BY batsman HAVING COUNT(CASE WHEN extras_type !=
'wides' THEN table4.ball ELSE NULL END) >= 500 AND
COUNT(table4.ball) >= 300 ORDER BY batting_strike_rate DESC,
bowling_strike_rate ASC LIMIT 10;
```

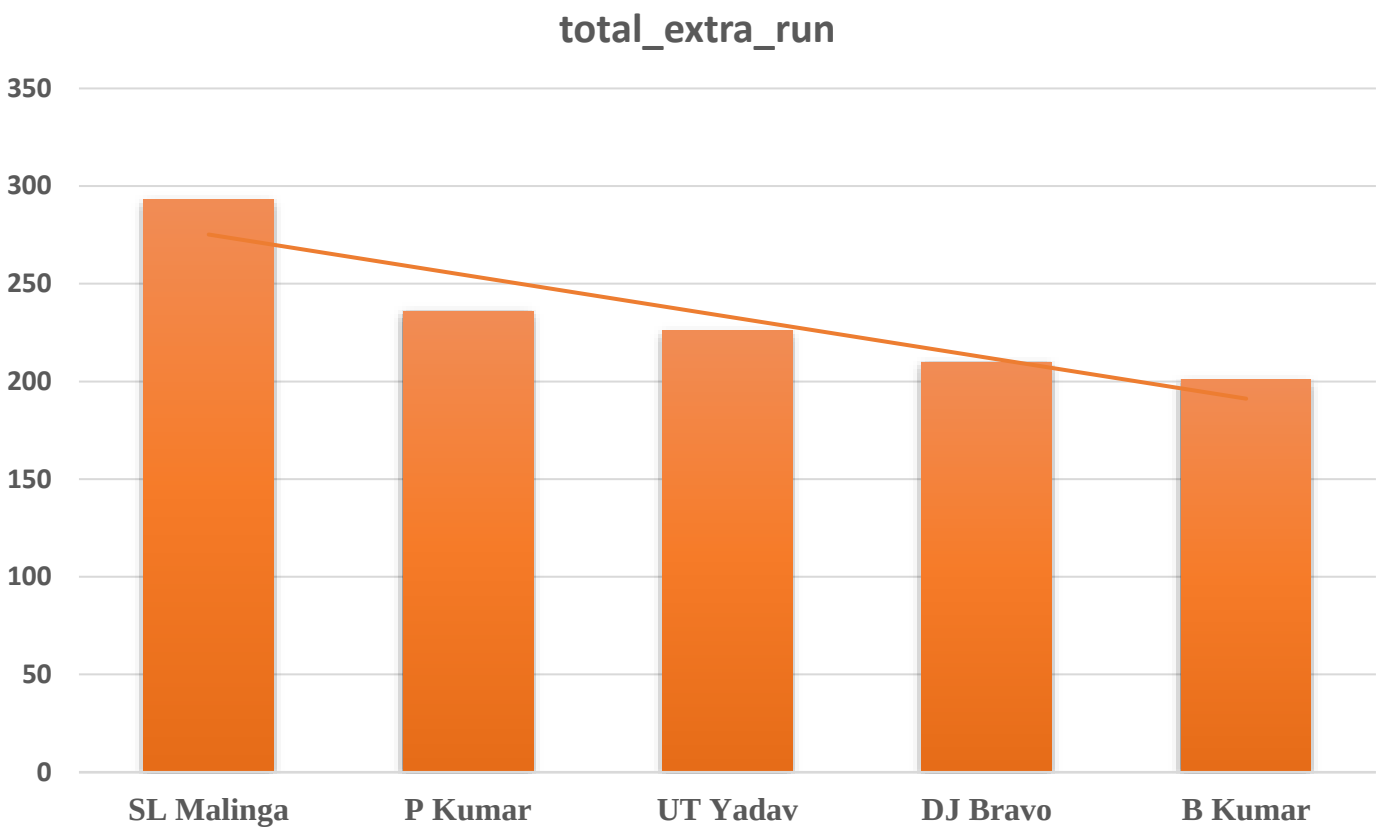
or

```
SELECT bowler, (SUM(total_run) * 1.0 / COALESCE(COUNT(CASE
WHEN extras_type != 'wides' THEN table4.ball ELSE NULL END),
1)) AS batting_strike_rate, (COUNT(table4.ball) /
COUNT(DISTINCT player_dismissed)) AS bowling_strike_rate
FROM table4 WHERE batsman IS NOT NULL GROUP BY bowler
HAVING COUNT(CASE WHEN extras_type != 'wides' THEN
table4.ball ELSE NULL END) >= 500 AND COUNT(table4.ball) >=
300 ORDER BY batting_strike_rate DESC, bowling_strike_rate ASC
LIMIT 10;
```

	batsman character varying	batting_strike_rate numeric	bowling_strike_rate bigint
1	AD Russell	1.9278846153846154	294
2	SP Narine	1.7661141804788214	286
3	HH Pandya	1.7048406139315230	224
4	V Sehwag	1.6609686609686610	366
5	GJ Maxwell	1.6402877697841727	202
6	CH Gayle	1.6052217678515256	1114
7	RR Pant	1.5847953216374269	354
8	ST Jayasuriya	1.5845864661654135	191
9	KA Pollard	1.5805651958353991	301
10	AB de Villiers	1.5714285714285714	816

	bowler character varying	batting_strike_rate numeric	bowling_strike_rate bigint
1	MP Stoinis	1.5944954128440367	23
2	AS Rajpoot	1.5658627087198516	24
3	M Prasidh Krishna	1.5547024952015355	25
4	HH Pandya	1.5366972477064220	24
5	Mohammed Siraj	1.5312934631432545	21
6	DR Smith	1.5193370165745856	20
7	VR Aaron	1.5164609053497942	25
8	Mohammed Shami	1.5132547864506627	24
9	AD Russell	1.5117085862966175	23
10	SN Thakur	1.5005537098560354	22

Top 5 bowlers who conceded maximum extra runs

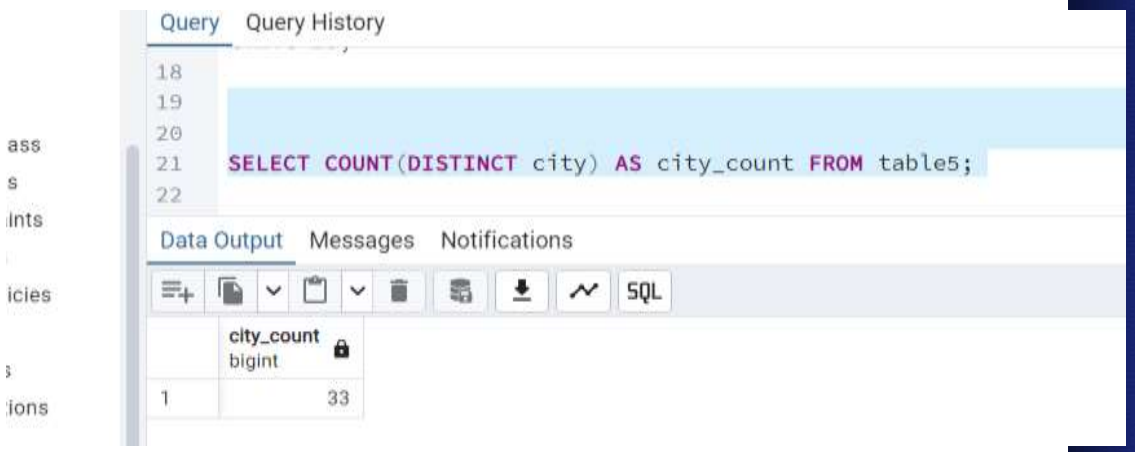


	bowler character varying	total_extra_run bigint
1	SL Malinga	293
2	P Kumar	236
3	UT Yadav	226
4	DJ Bravo	210
5	B Kumar	201

Additional Questions:

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

```
SELECT COUNT(DISTINCT city) AS city_count FROM table5;
```



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)

```
CREATE TABLE deliveries_v02 ASSELECT   *,  CASE
WHEN total_run >= 4 THEN 'boundary'    WHEN
total_run = 0 THEN 'dot'    ELSE 'other'  END AS
ball_resultFROM table4;
```

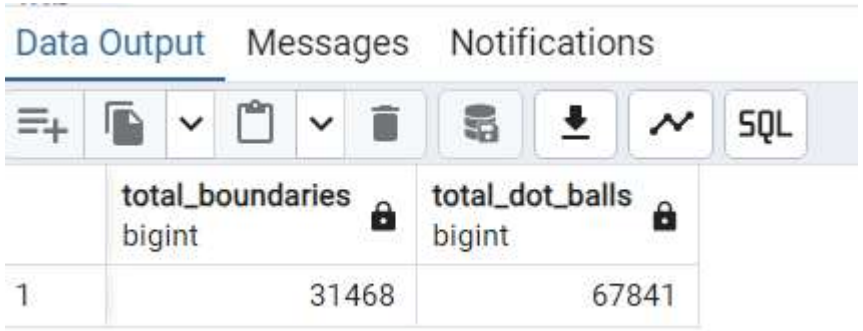
In the same folder I attached a text file where all queries present , this pptx is for presentations

	total_run integer	is_wicket integer	dismissal_kind character varying	player_dismissed character varying	fielder character varying	extras_type character varying	batting_team character varying	bowling_team character varying	ball_result text
25	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	dot
26	4	4	0	NA	NA	NA	byes	Kolkata Knight Riders	Royal Challengers Bangalore	boundary
27	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	dot
28	0	6	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	boundary
29	0	2	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	other
30	0	1	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	other
31	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	dot
32	0	1	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	other
33	0	0	1	caught	RT Ponting	P Kumar	NA	Kolkata Knight Riders	Royal Challengers Bangalore	dot
34	0	4	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	boundary
35	0	0	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	dot
36	0	2	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	other
37	1	1	0	NA	NA	NA	wides	Kolkata Knight Riders	Royal Challengers Bangalore	other
38	0	4	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	boundary
39	0	1	0	NA	NA	NA	NA	Kolkata Knight Riders	Royal Challengers Bangalore	other

[table4 represent IPL_Ball Data

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

```
SELECT   SUM(CASE WHEN ball_result =
'boundary' THEN 1 ELSE 0 END) AS
total_boundaries,  SUM(CASE WHEN ball_result
= 'dot' THEN 1 ELSE 0 END) AS total_dot_balls
FROM deliveries_v02;
```



	total_boundaries bigint	total_dot_balls bigint
1	31468	67841

[table4 represent IPL_Ball Data

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, SUM(CASE WHEN ball_result = 'boundary' THEN 1 ELSE 0 END) AS total_boundaries FROM deliveries_v02 GROUP BY batting_team ORDER BY total_boundaries DESC;
```

	batting_team character varying	total_boundaries bigint
1	Mumbai Indians	4118
2	Royal Challengers Bangalore	3800
3	Kings XI Punjab	3780
4	Kolkata Knight Riders	3739
5	Chennai Super Kings	3496
6	Rajasthan Royals	3041
7	Delhi Daredevils	3022
8	Sunrisers Hyderabad	2306
9	Deccan Chargers	1387
10	Pune Warriors	733
11	Delhi Capitals	659
12	Gujarat Lions	624
13	Rising Pune Supergiant	290
14	Rising Pune Supergiants	242
15	Kochi Tuskers Kerala	231

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.

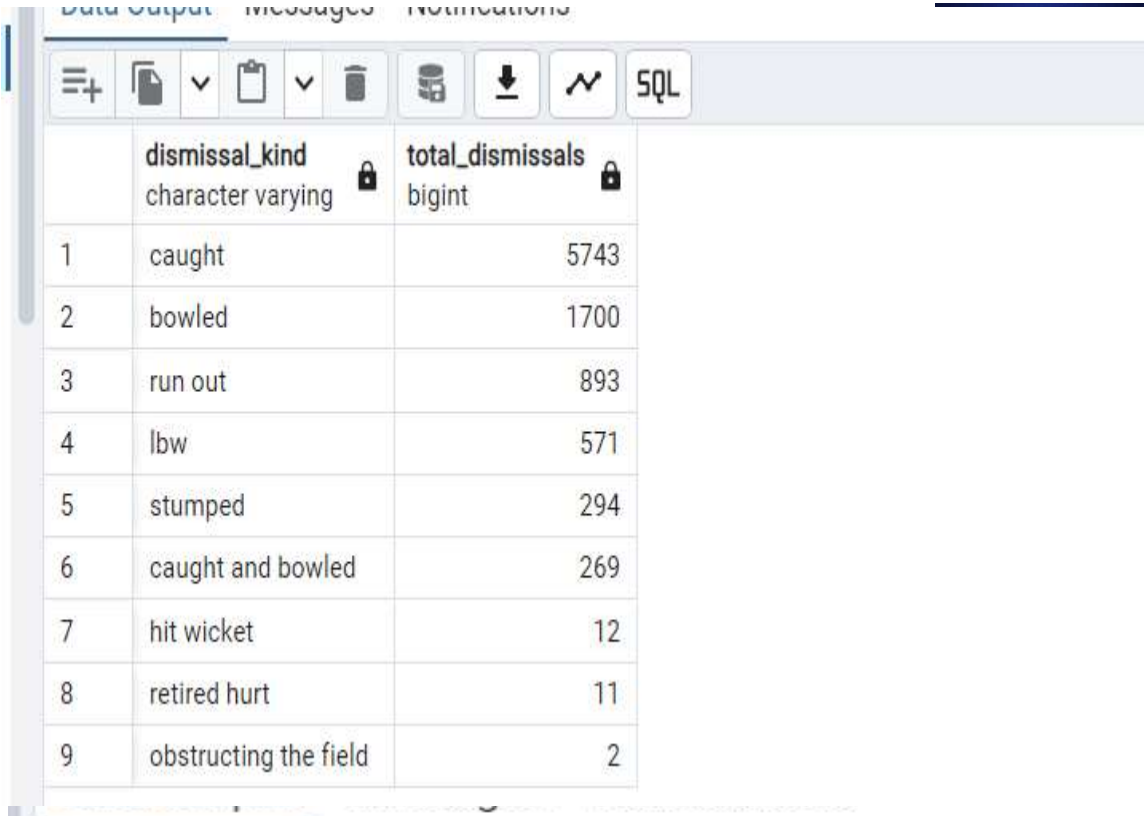
```
SELECT bowling_team, SUM(CASE WHEN ball_result = 'dot' THEN 1 ELSE 0 END) AS total_dot_balls FROM deliveries_v02 GROUP BY bowling_team ORDER BY total_dot_balls DESC;
```

	bowling_team character varying	total_dot_balls bigint
1	Mumbai Indians	8714
2	Royal Challengers Bangalore	7955
3	Kolkata Knight Riders	7894
4	Kings XI Punjab	7679
5	Chennai Super Kings	7593
6	Rajasthan Royals	6665
7	Delhi Daredevils	6520
8	Sunrisers Hyderabad	5248
9	Deccan Chargers	3306
10	Pune Warriors	1900
11	Delhi Capitals	1338
12	Gujarat Lions	1095
13	Rising Pune Supergiant	698
14	Kochi Tuskers Kerala	626
15	Rising Pune Supergiants	539

[table4 represent IPL_Ball Data

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

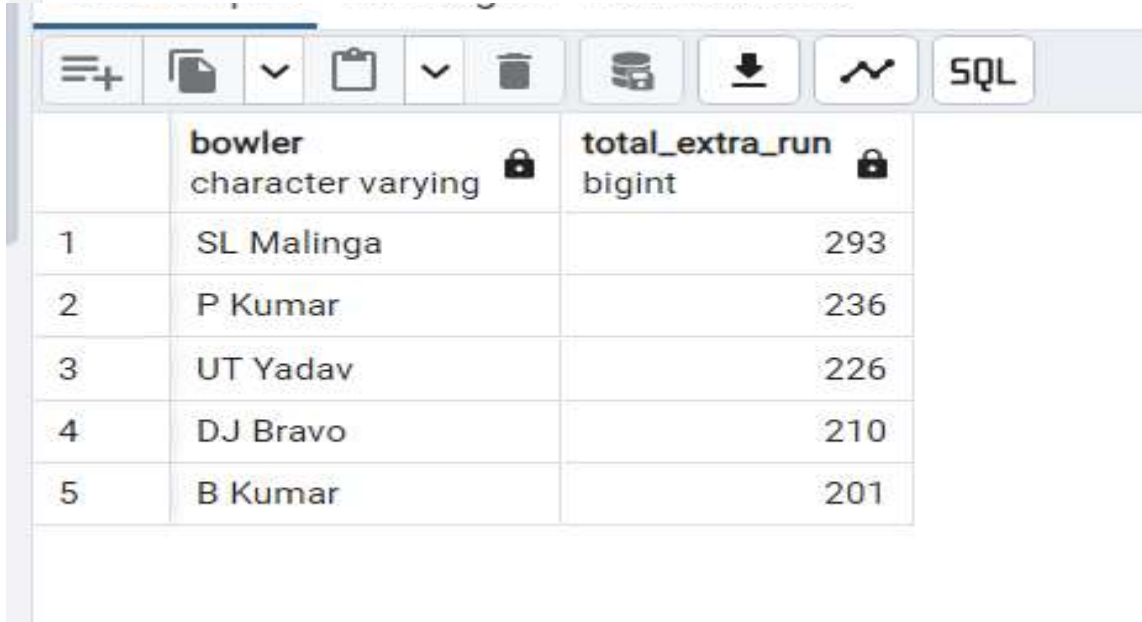
```
SELECT   dismissal_kind,   COUNT(*) AS total_dismissals
FROM deliveries_v02 WHERE dismissal_kind !=
'NA'GROUP BY dismissal_kind ORDER BY
total_dismissals DESC;
```



	dismissal_kind character varying	total_dismissals bigint
1	caught	5743
2	bowled	1700
3	run out	893
4	lbw	571
5	stumped	294
6	caught and bowled	269
7	hit wicket	12
8	retired hurt	11
9	obstructing the field	2

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

```
SELECT bowler, SUM(extra_run) AS total_extra_run
FROM table4 GROUP BY bowler ORDER BY
total_extra_run DESC
LIMIT 5;
```



	bowler character varying	total_extra_run bigint
1	SL Malinga	293
2	P Kumar	236
3	UT Yadav	226
4	DJ Bravo	210
5	B Kumar	201

[table4 represent IPL_Ball Data
[table5 represent ipl_matches data]

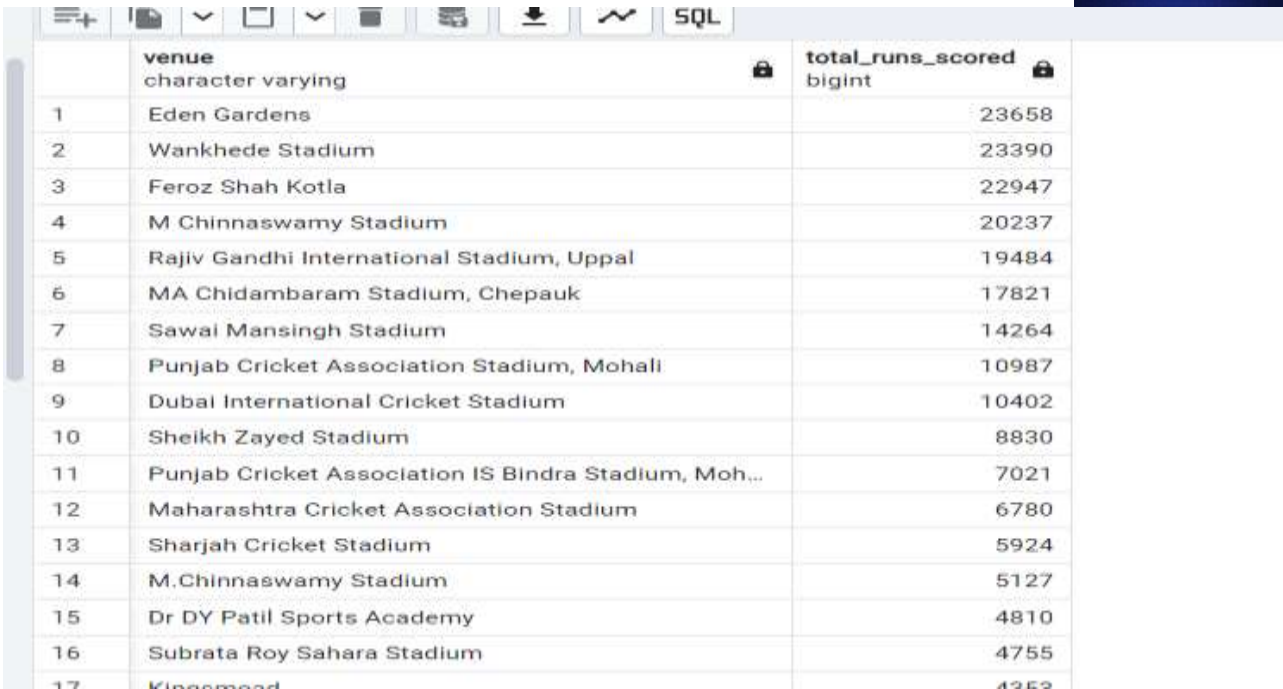
Write a query to create a table named deliveries_v03 with all the columns of deliveries_v02 table and two additional column (named venue and match_date) of venue and date from table matches

```
CREATE TABLE deliveries_v03 AS SELECT
 d.*, m.venue, m.date AS
match_date FROM table4 d
JOIN table5 m ON d.id = m.id;
```

	id integer	inning integer	over integer	ball integer	batsman character varying	non_stricker character varying	bowler character varying	batsman_run integer	extra_run integer	total_run integer	is_wicket integer	dismissal_kind character varying
1	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	1	0	NA
2	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	1	0	NA
3	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	0	0	NA
4	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	1	0	NA
5	335982	1	7	3	RT Ponting	BB McCullum	Z Khan	1	0	1	0	NA
6	335982	1	7	4	BB McCullum	RT Ponting	Z Khan	1	0	1	0	NA
7	335982	1	7	5	RT Ponting	BB McCullum	Z Khan	1	0	1	0	NA
8	335982	1	7	6	BB McCullum	RT Ponting	Z Khan	1	0	1	0	NA
9	335982	1	8	1	BB McCullum	RT Ponting	JH Kallis	0	0	0	0	NA
10	335982	1	8	2	BB McCullum	RT Ponting	JH Kallis	0	0	0	0	NA
11	335982	1	8	3	BB McCullum	RT Ponting	JH Kallis	0	0	0	0	NA
12	335982	1	8	4	BB McCullum	RT Ponting	JH Kallis	1	0	1	0	NA

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_run) AS total_runs_scored
FROM deliveries_v03 GROUP BY venue ORDER BY
total_runs_scored DESC;
```

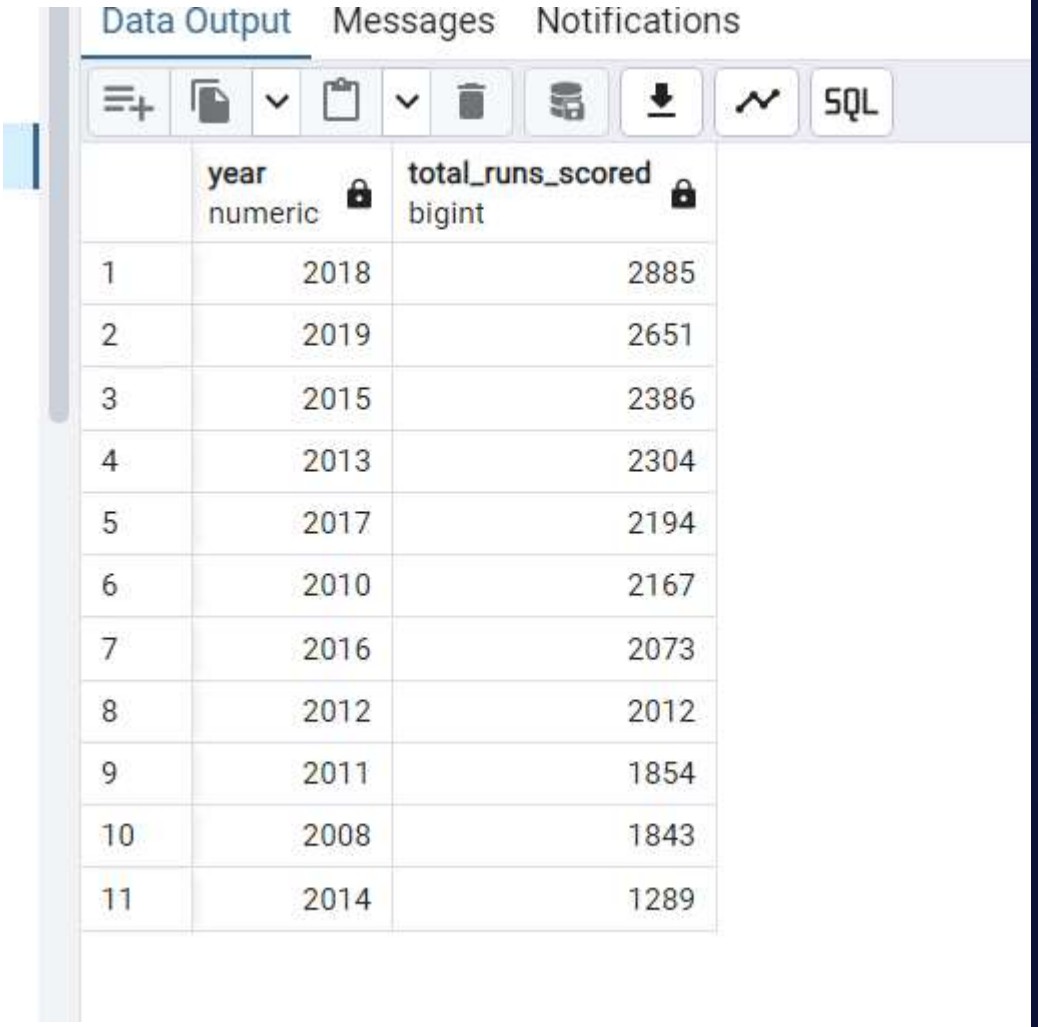


	venue character varying	total_runs_scored bigint
1	Eden Gardens	23658
2	Wankhede Stadium	23390
3	Feroz Shah Kotla	22947
4	M Chinnaswamy Stadium	20237
5	Rajiv Gandhi International Stadium, Uppal	19484
6	MA Chidambaram Stadium, Chepauk	17821
7	Sawai Mansingh Stadium	14264
8	Punjab Cricket Association Stadium, Mohali	10987
9	Dubai International Cricket Stadium	10402
10	Sheikh Zayed Stadium	8830
11	Punjab Cricket Association IS Bindra Stadium, Moh...	7021
12	Maharashtra Cricket Association Stadium	6780
13	Sharjah Cricket Stadium	5924
14	M.Chinnaswamy Stadium	5127
15	Dr DY Patil Sports Academy	4810
16	Subrata Roy Sahara Stadium	4755

[table4 represent IPL_Ball Data

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 EXTRACT(YEAR FROM match_date) AS
Year,SUM(total_run) AS total_runs_scored
FROM deliveries_v03 WHERE venue = 'Eden
Gardens'GROUP BY EXTRACT(YEAR FROM
match_date)
ORDER BY total_runs_scored DESC;
```



Data Output | Messages | Notifications

	year numeric	total_runs_scored bigint
1	2018	2885
2	2019	2651
3	2015	2386
4	2013	2304
5	2017	2194
6	2010	2167
7	2016	2073
8	2012	2012
9	2011	1854
10	2008	1843
11	2014	1289

Wicketkeeper Criteria

I would define the following criteria to determine which wicketkeeper is the best, taking into account the needs of a T20 team:

1. **Batting Strike Rate**: To demonstrate their ability to score runs fast and forcefully, the wicketkeeper should have a high batting strike rate (SR) of at least 120.

2. **Batting Average**: A strong batting average of at least 25 demonstrates a player's reliability in scoring runs and interpersonal skills.

3. **Dismissals**: A high rate of dismissals (catches + stumpings) each game, with at least 20 dismissals in each of the previous two IPL seasons.

4. **Catching Efficiency**: The ability to hold onto catches and influence dismissals is demonstrated by a catching efficiency of at least 90%.

5. **Running between wickets**: The capacity to move swiftly between wickets and score runs; in the previous two IPL seasons, this ability has resulted in at least 50 runs.

By taking these factors into account, we can find a wicketkeeper who can help the team win in all facets of the game and who is also a strong gloveman and hitter.

List of suggested player name for wicketkeeper:
1.AD Russell
2.AC Glichrist
3.DE Bolinger
Etc.