ANIMATION – ON CLICK

Note - In Slideshow mode, each graphic element with text will appear/animate on mouse click OR press space bar/enter key/arrow key (right ► or down ▼) to trigger the remaining animation.

- All The Criteria Used Queries Are Given In Query Slides, If You Want To Run The Query, Just Do Copy And Paste.

- All The Criteria Questions Also Written For Better Understanding.
- Players Criteria’s Are Different And Present In First Of Every Sections Slide.
- Individual Player Criteria’s OR Explanation Also Mention Before The Query Slide.


CRICKET

CRICKET GROUND

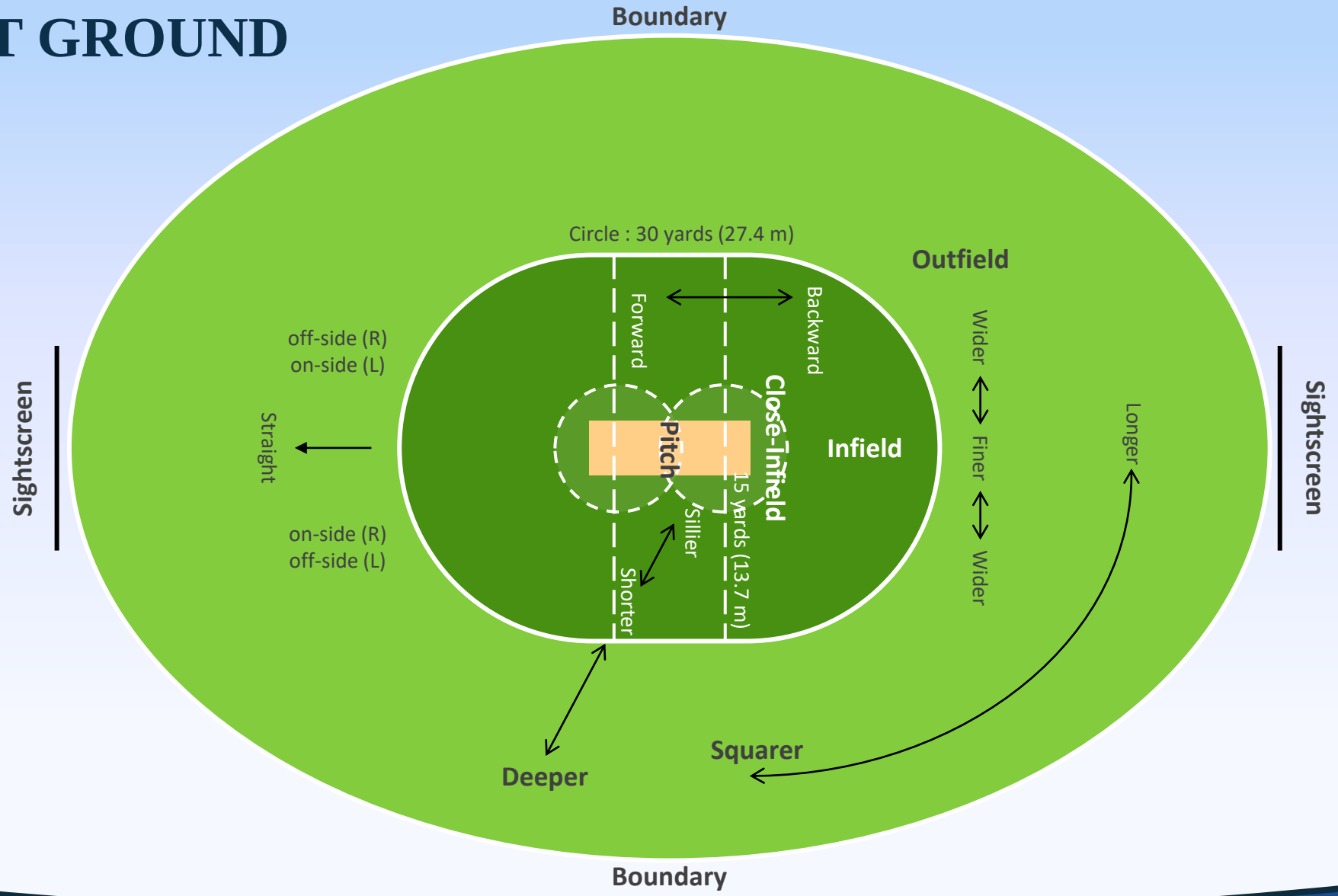


Table Creation :

IPL_BALL Table Creation Query.

```
CREATE TABLE IPL_BALL
(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);
```

Importing CSV File.

```
COPY IPL_BALL
FROM 'C:\Program Files\PostgreSQL\15\data\A - SQL - Data Sources\M1_T4_V1 Restore\IPL Dataset\IPL_Ball.csv' DELIMITER ',' CSV HEADER;
```

IPL_MATCH Table Creation Query.

```
CREATE TABLE IPL_MATCHES
(ID int, CITY varchar, DATE date, PLAYER_OF_MATCH varchar, VENUE varchar, NEUTRAL_VENUE int, 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);
```

Importing CSV File.

```
COPY IPL_MATCHES
FROM 'C:\Program Files\PostgreSQL\15\data\A - SQL - Data Sources\M1_T4_V1 Restore\IPL Dataset\IPL_matches.csv' DELIMITER ',' CSV HEADER;
```

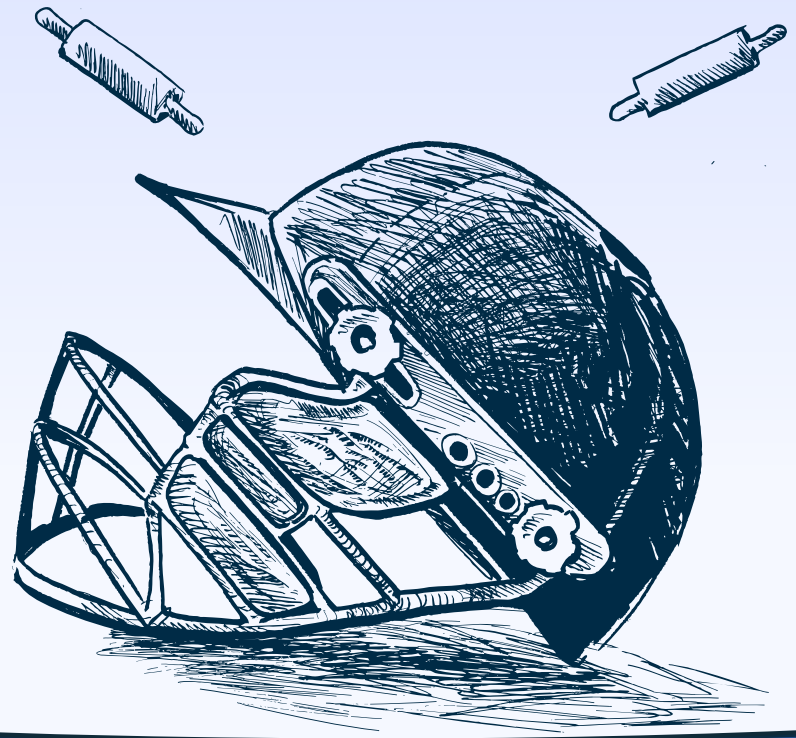
Batter’s Criteria :

Anchor Batter’s

The priority to get good Average players who have played more than 2 IPL seasons with the list of top 10 players.

Aggressive Batter’s

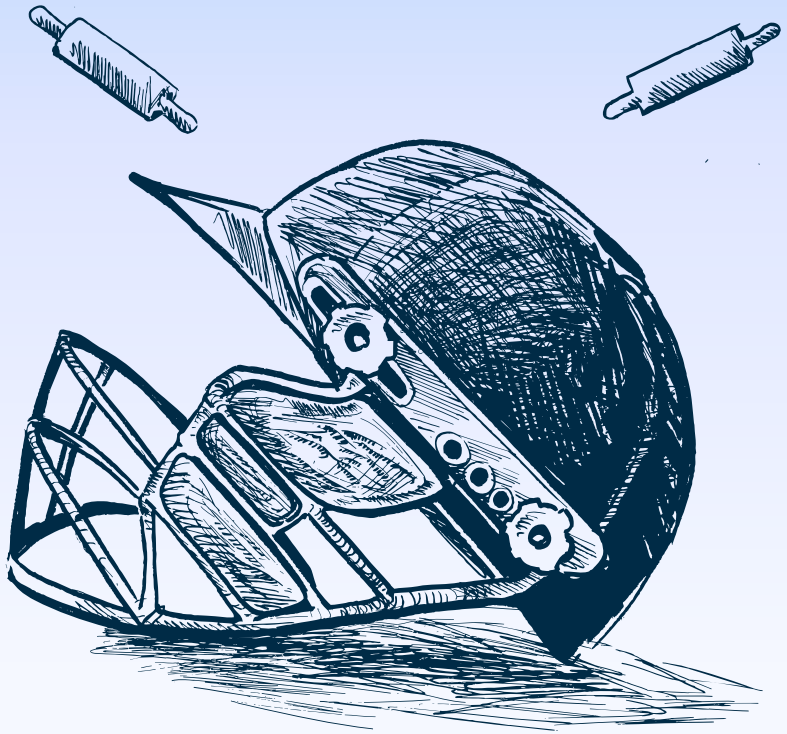
The priority is to get the players with high S.R who have faced at least 500 balls(excluding wides) with the top 10 players.



Hard Hit Batter’s

The priority to get the Hard-hitting players who scored most runs in boundaries and have played more the 2 IPL season with the list of top 10 players.

Aggressive Batter's Criteria (OR) Explanation :



To Find The Aggressive Batter's I Use The IPL Ball Table Only.

- *First I Find The Batsman Who Have Played More Than 500 Balls (Excluding Wides).*
- *Second I Find The Batsman Who Have The High Strike Rate(Excluding Wides).*
- *Third In The Last I Combine Both The Queries To Find The Players With High Strike Rate And Have Played More Than 500 Balls (Excluding Wides).*

Aggressive Batter's Query :

1. **Batsman Who Faced More Than 500 Balls Of Top 10 Players (Excluding Wides).**

```
SELECT batsman AS Player_Name, SUM(CASE WHEN extras_type != 'wides' THEN batsman_runs ELSE 0 END) AS RunsScored,
COUNT(ball) AS Balls_faced FROM ipl_ball WHERE extras_type != 'wides'
GROUP BY batsman HAVING COUNT(ball) >= 500 order by balls_faced DESC LIMIT 10
```

2. **Batsman Who Have High Strike Rate Of Top 10 Players (Excluding Wides).**

```
SELECT batsman AS PlayerName,
(SUM(CASE WHEN extras_type != 'wides' THEN batsman_runs ELSE 0 END) * 100.0 / NULLIF(COUNT(CASE WHEN extras_type != 'wides' THEN 1
                                                                                              ELSE 0 END), 0)) AS StrikeRate
FROM ipl_ball WHERE extras_type != 'wides' GROUP BY batsman ORDER BY StrikeRate DESC LIMIT 10;
```

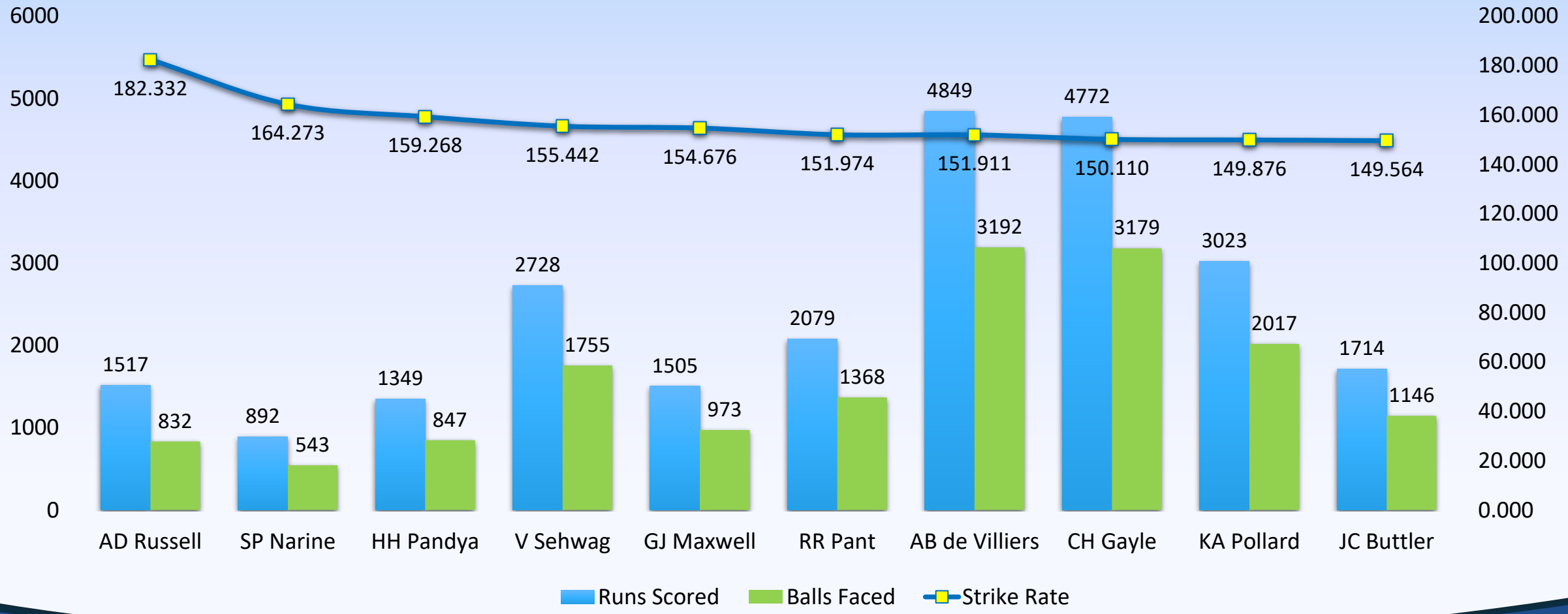
3. **Batsman Who Faced More Than 500 Balls And Have High Strike Rate Of Top 10 Players (Excluding Wides).**

```
SELECT batsman AS PlayerName, SUM(CASE WHEN extras_type != 'wides' THEN batsman_runs ELSE 0 END) AS RunsScored,
COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE 0 END) AS BallsFaced,
(SUM(CASE WHEN extras_type != 'wides' THEN batsman_runs ELSE 0 END) * 100.0 / NULLIF(COUNT(CASE WHEN extras_type != 'wides' THEN 1
                                                                                              ELSE 0 END), 0)) AS StrikeRate
FROM ipl_ball WHERE extras_type != 'wides'
GROUP BY batsman HAVING COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE 0 END) >= 500
ORDER BY StrikeRate DESC LIMIT 10;
```

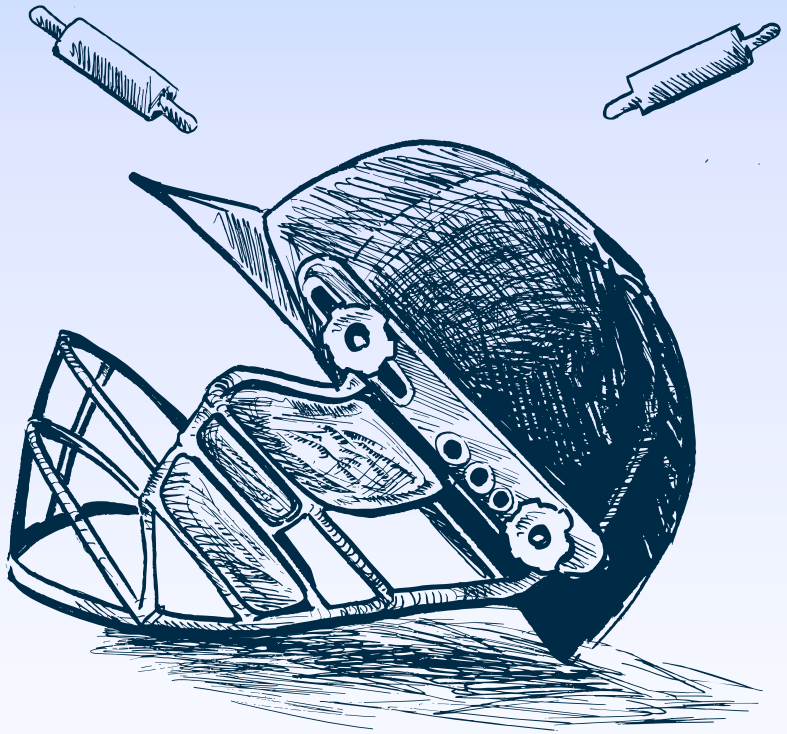
Aggressive Batter’s

Rank	Player Name	Runs Scored	Balls Faced	Strike Rate
1	AD Russell	1517	832	182.33
2	SP Narine	892	543	164.27
3	HH Pandya	1349	847	159.26
4	V Sehwag	2728	1755	155.44
5	GJ Maxwell	1505	973	154.67
6	RR Pant	2079	1368	151.97
7	AB de Villiers	4849	3192	151.91
8	CH Gayle	4772	3179	150.11
9	KA Pollard	3023	2017	149.87
10	JC Buttler	1714	1146	149.56

Aggressive Batter's



Anchor Batter’s Criteria (OR) Explanation :



To Find The Anchor Batter’s I Use The IPL Ball Table And IPL Match Table.

- *First I Find The Batsman Who Have Played More Than 2 IPL Seasons.*
- *Second The Find The Batsman Who Have Good Average With Dismissals.*
- *Third In The Last I Combine Both The Queries To Find The Players Who Have Good Average with Dismissals And Have Played More Than 2 IPL Seasons.*

Anchor Batter's Query :

1. **Batsman Who Have Played More Than 2 IPL Seasons.**

```
SELECT b.batsman AS player_name, COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) AS num_seasons_played
FROM ipl_ball AS b JOIN ipl_matches AS m ON b.id = m.id
GROUP BY b.batsman HAVING COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) > 2
ORDER BY num_seasons_played DESC LIMIT 10;
```

2. **Batsman Who Have Good Average With Dismissals.**

```
SELECT b.batsman AS player_name,
SUM(b.is_wicket) AS Dismissals, (SUM(b.batsman_runs) / NULLIF(SUM(b.is_wicket), 0)) AS Batting_Average
FROM IPL_BALL as b JOIN IPL_MATCHES as m ON b.id = m.id
GROUP BY b.batsman HAVING SUM(b.is_wicket) > 0
ORDER BY Batting_Average DESC LIMIT 10;
```

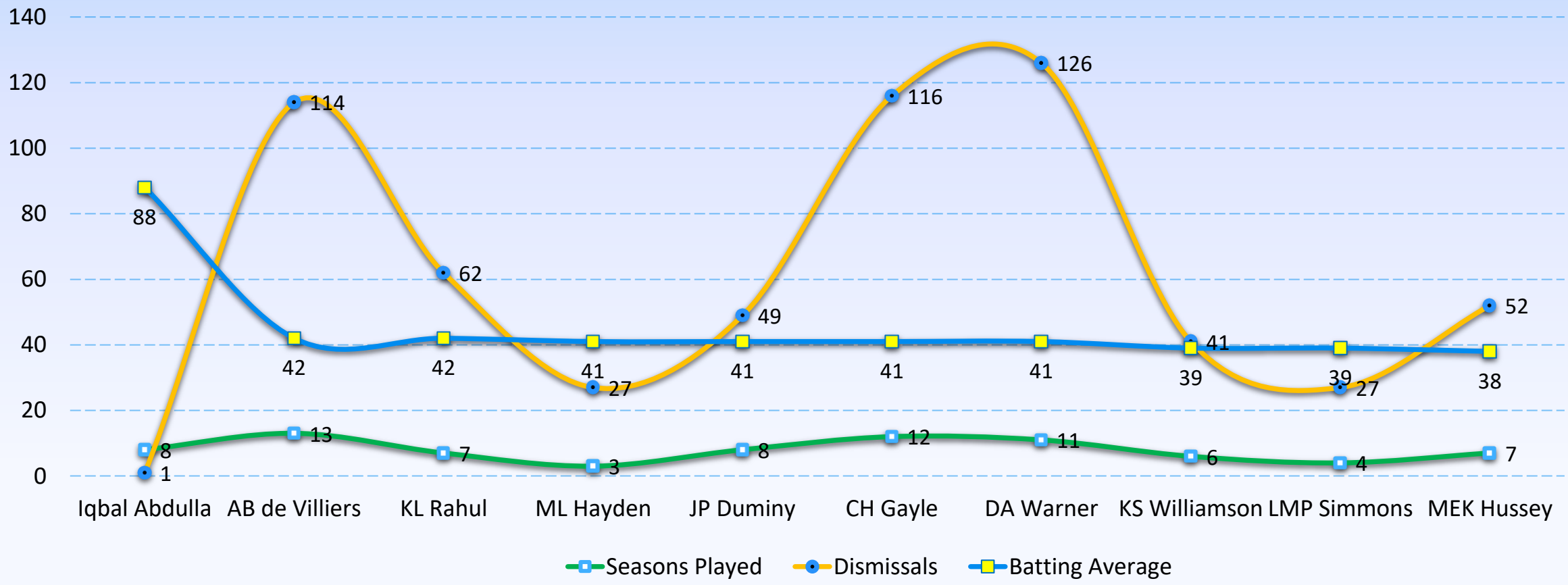
3. **Batsman Who Have Played More Than 2 IPL Seasons And Having Good Averages With Dismissals.**

```
SELECT b.batsman AS player_name, COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) AS seasons_played,
SUM(b.is_wicket) AS Dismissals, (SUM(b.batsman_runs) / NULLIF(SUM(b.is_wicket), 0)) AS batting_average
FROM ipl_ball as b JOIN ipl_matches as m ON b.id = m.id
GROUP BY b.batsman HAVING COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) > 2 AND SUM(b.is_wicket) > 0
ORDER BY batting_average DESC LIMIT 10;
```

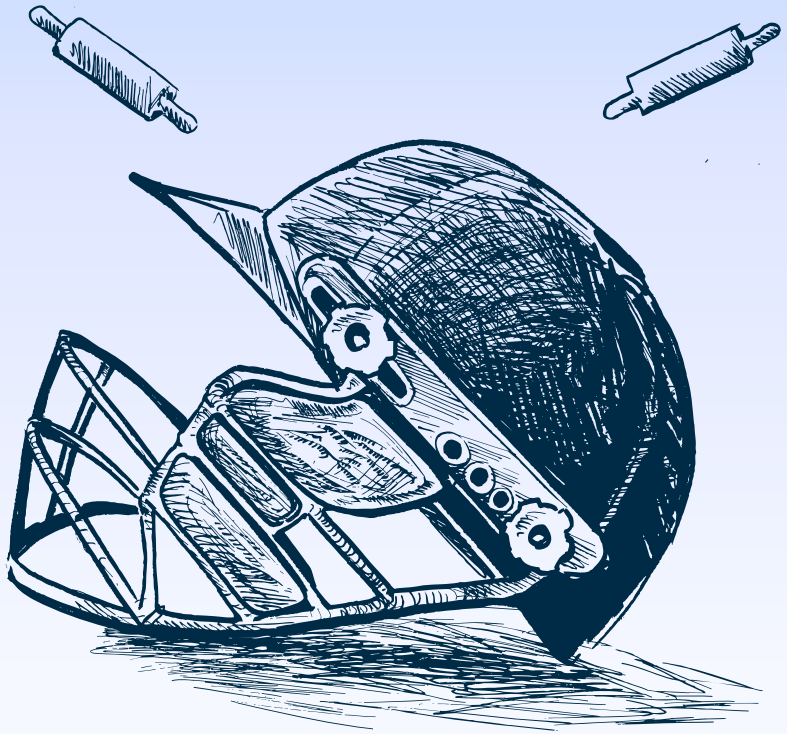
Anchor Batter’s

Rank	Player Name	Seasons Played	Dismissals	Batting Average
1	Iqbal Abdulla	8	1	88
2	AB de Villiers	13	114	42
3	KL Rahul	7	62	42
4	ML Hayden	3	27	41
5	JP Duminy	8	49	41
6	CH Gayle	12	116	41
7	DA Warner	11	126	41
8	KS Williamson	6	41	39
9	LMP Simmons	4	27	39
10	MEK Hussey	7	52	38

Anchor Batter's



Hard Hit Batter's Criteria (OR) Explanation :



To Find The Hard Hit Batter's I Use The IPL Ball Table And IPL Match Tables.

- *First I Find The Batsman Who Have Played More Than 2 IPL Seasons.*
- *Second The Find The Batsman Who Have Most Runs In Boundaries.*
- *Third In The Last I Combine Both The Queries To Find The Players Who Have Most Runs In Boundaries And Have Played More Than 2 IPL Seasons.*

Hard Hit Batter's Query :

1. Batsman Who Have Played More Than 2 IPL Seasons.

```
SELECT b.batsman AS player_name, COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) AS num_seasons_played
FROM ipl_ball AS b JOIN ipl_matches AS m ON b.id = m.id
GROUP BY b.batsman HAVING COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) > 2 ORDER BY num_seasons_played DESC LIMIT 10;
```

2. Batsman Who Have Most Runs In Boundaries In Fours And Sixes.

```
SELECT batsman AS player_name,
SUM(CASE WHEN batsman_runs = 4 THEN 1 ELSE 0 END) AS fours, SUM(CASE WHEN batsman_runs = 6 THEN 1 ELSE 0 END) AS sixes,
SUM(CASE WHEN batsman_runs IN (4, 6) THEN 1 ELSE 0 END) AS total_boundaries,
SUM(CASE WHEN batsman_runs IN (4, 6) THEN batsman_runs ELSE 0 END) AS boundary_runs,
ROUND(SUM(CASE WHEN batsman_runs IN (4, 6) THEN batsman_runs ELSE 0 END) * 100.0 / SUM(total_runs), 2) AS boundary_percentage
FROM ipl_ball WHERE batsman_runs IN (4, 6) GROUP BY batsman ORDER BY boundary_runs DESC LIMIT 10;
```

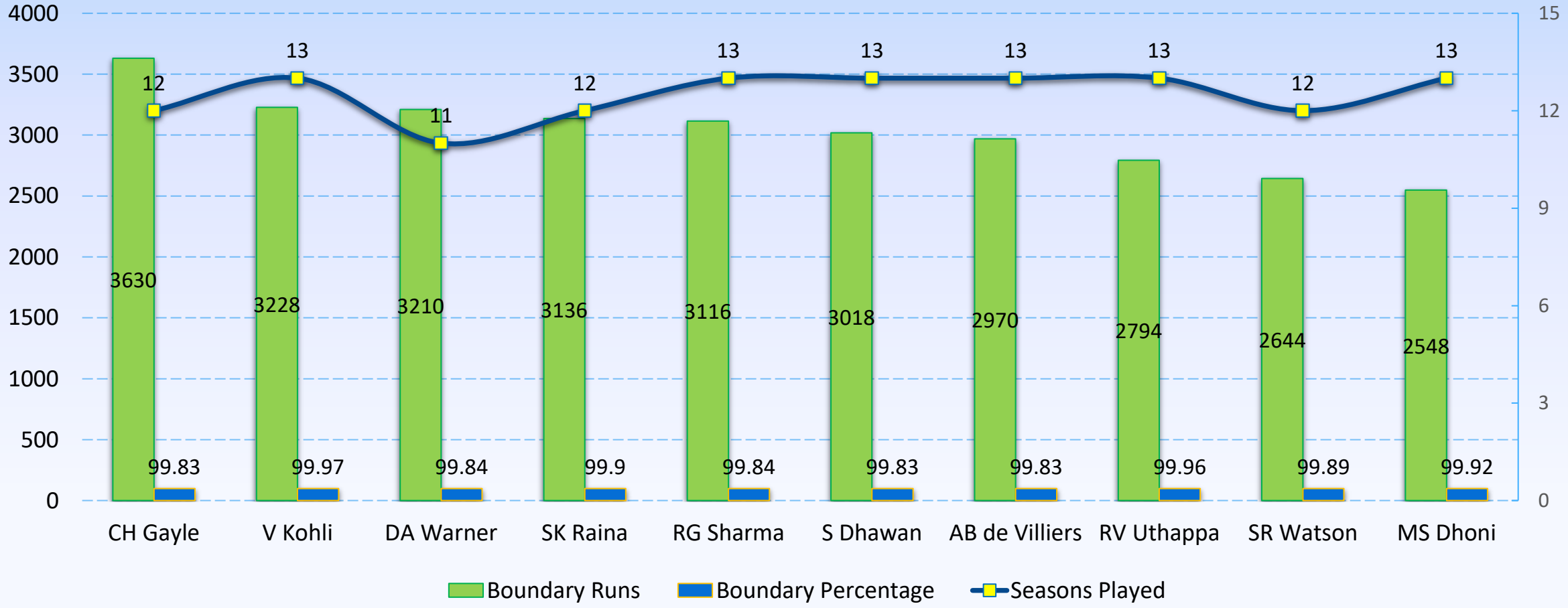
3. Batsman Who Have Played More Than 2 IPL Seasons And Most Runs In Boundaries In Fours And Sixes.

```
SELECT b.batsman AS player_name, SUM(CASE WHEN b.batsman_runs IN (4, 6) THEN b.batsman_runs ELSE 0 END) AS boundary_runs,
ROUND(SUM(CASE WHEN b.batsman_runs IN (4, 6) THEN b.batsman_runs ELSE 0 END) * 100.0 / SUM(b.total_runs), 2) AS boundary_percentage,
COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) AS num_seasons_played
FROM ipl_ball AS b JOIN ipl_matches AS m ON b.id = m.id WHERE b.batsman_runs IN (4, 6)
GROUP BY b.batsman HAVING COUNT(DISTINCT EXTRACT(YEAR FROM m.date)) > 2 ORDER BY boundary_runs DESC LIMIT 10;
```

Hard Hit Batter’s

Rank	Player Name	Boundary Runs	Percentage	Seasons Played
1	CH Gayle	3630	99.83	12
2	V Kohli	3228	99.97	13
3	DA Warner	3210	99.84	11
4	SK Raina	3136	99.90	12
5	RG Sharma	3116	99.84	13
6	S Dhawan	3018	99.83	13
7	AB de Villiers	2970	99.83	13
8	RV Uthappa	2794	99.96	13
9	SR Watson	2644	99.89	12
10	MS Dhoni	2548	99.92	13

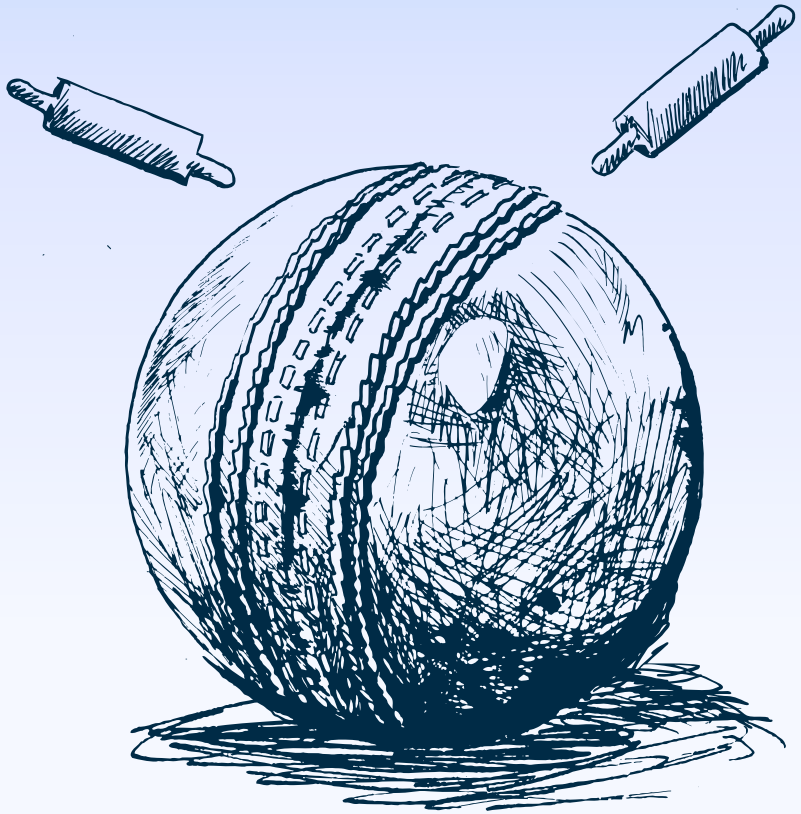
Hard Hit Batter's



Bowler’s Criteria :

Economic Bowler’s

The priority is to get the players with Good Economy and have bowled more than 500 balls.



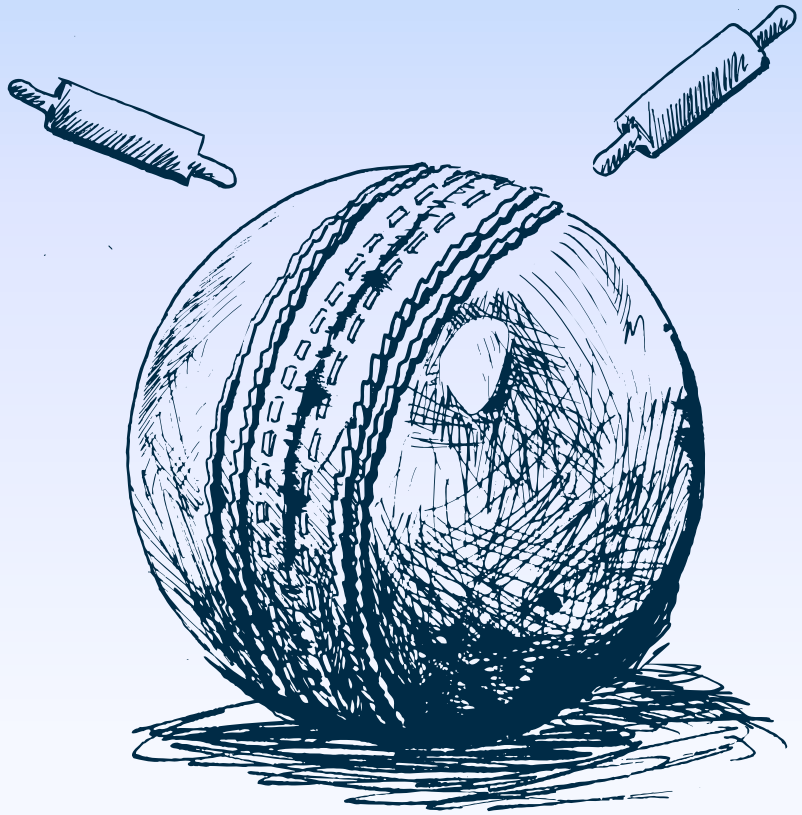
Wicket Taker Bowler’s

The priority is to get the players with High strike Rate and have bowled more than 500 balls.

Economy Bowler’s Criteria (OR) Explanation :

To Find The Economic Bowler’s I Use The IPL Ball Table Only.

- *First I Find The Bowler’s Who bowled More Than 500 balls.*
- *Second I Find The Bowler’s Who Have Good Economy.*
- *Third In The Last I Combine Both The Queries To Find The Players Who Have bowled More Than 500 balls And Have Good Economy.*



Economy Bowler’s Query :

1. **Batsman Who Bowled More Than 500 Balls.**

- *SELECT bowler AS player_name, COUNT(ball) AS balls_bowled FROM ipl_ball*
- *GROUP BY bowler HAVING COUNT(ball) >= 500 ORDER BY balls_bowled DESC LIMIT 10;*

2. **Batsman Who Have Good Economy.**

- *SELECT bowler, SUM(total_runs) AS total_runs_conceded,*
- *COUNT(DISTINCT CONCAT(over, '.', ball)) AS total_overs_bowled,*
- *ROUND(SUM(total_runs) / COUNT(DISTINCT CONCAT(over, '.', ball)), 2) AS economy_rate*
- *FROM ipl_ball GROUP BY bowler HAVING COUNT(DISTINCT CONCAT(over, '.', ball)) > 0*
- *ORDER BY economy_rate DESC LIMIT 10;*

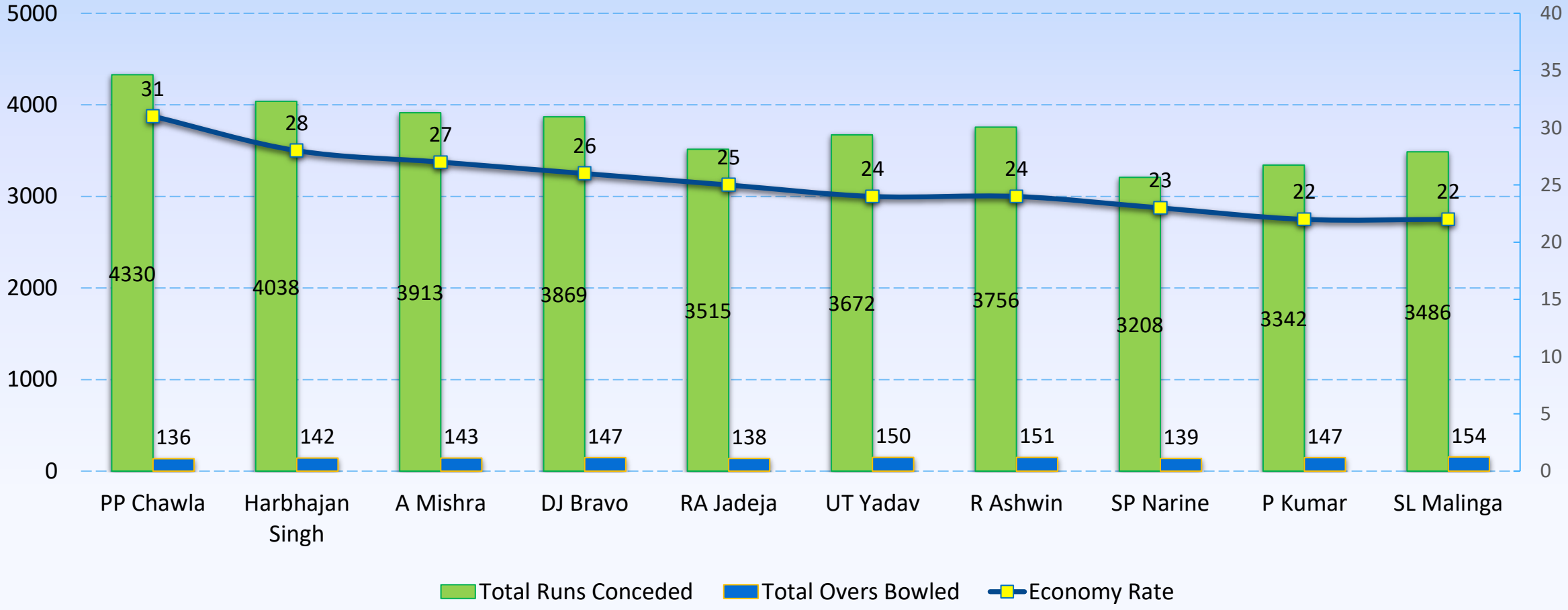
3. **Batsman Who Bowled More Than 500 Balls And Have Good Economy.**

- *SELECT bowler AS player_name, SUM(total_runs) AS total_runs_conceded,*
- *COUNT(DISTINCT CONCAT(over, '.', ball)) AS total_overs_bowled,*
- *ROUND(SUM(total_runs) / COUNT(DISTINCT CONCAT(over, '.', ball)), 2) AS economy_rate*
- *FROM ipl_ball GROUP BY bowler HAVING COUNT(ball) >= 500 AND COUNT(DISTINCT CONCAT(over, '.', ball)) > 0*
- *ORDER BY economy_rate DESC LIMIT 10;*

Economy Bowler’s

Rank	Player Name	Runs Conceded	Overs Bowled	Economy Rate
1	PP Chawla	4330	136	31
2	Harbhajan Singh	4038	142	28
3	A Mishra	3913	143	27
4	DJ Bravo	3869	147	26
5	RA Jadeja	3515	138	25
6	UT Yadav	3672	150	24
7	R Ashwin	3756	151	24
8	SP Narine	3208	139	23
9	P Kumar	3342	147	22
10	SL Malinga	3486	154	22

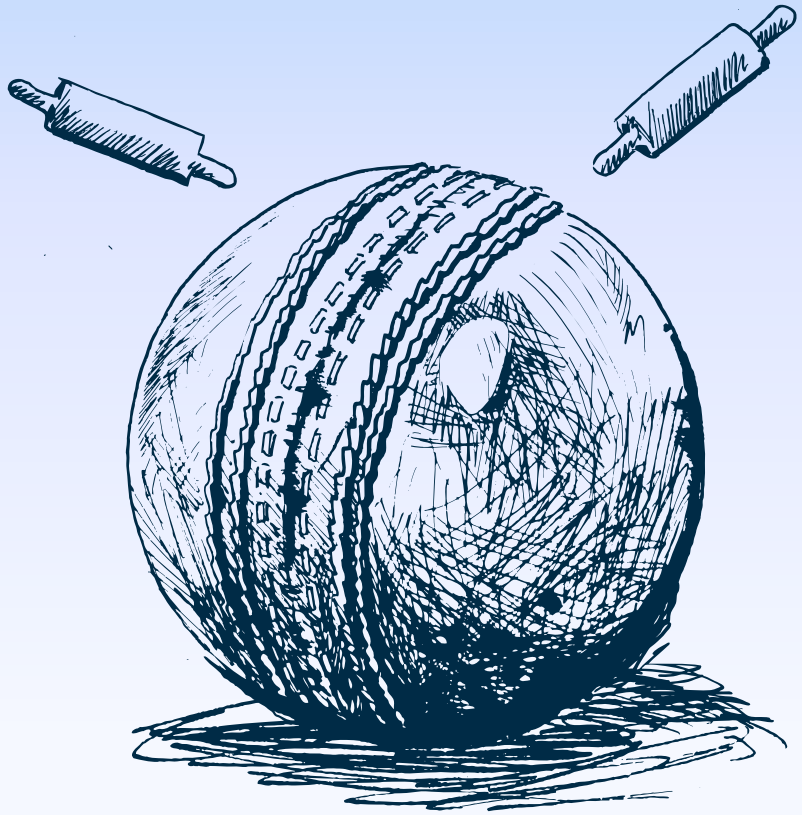
Economy Bowler's



Wicket Taker Bowler's Criteria (OR) Explanation :

To Find The Wicket Taker Bowler's I Use The IPL Ball Table Only.

- *First I Find The Bowler's Who bowled More Than 500 balls.*
- *Second I Find The Bowler's Who Have Best Strike Rate.*
- *Third In The Last I Combine Both The Queries To Find The Players Who Have bowled More Than 500 balls And Have Best Strike Rate.*



Wicket Taker Bowler's Query :

1. **Batsman Who Bowled More Than 500 Balls.**

- *SELECT bowler AS player_name, COUNT(ball) AS balls_bowled FROM ipl_ball*
- *GROUP BY player_name HAVING COUNT(*) > 0 AND COUNT(ball) >= 500 ORDER BY balls_bowled DESC LIMIT 10;*

2. **Batsman Who Have Best Strike Rate.**

- *SELECT bowler AS player_name, SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) AS total_wickets_taken*
- *FROM ipl_ball GROUP BY bowler*
- *HAVING COUNT(*) > 0 AND SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) > 0*
- *ORDER BY total_wickets_taken DESC LIMIT 10;*

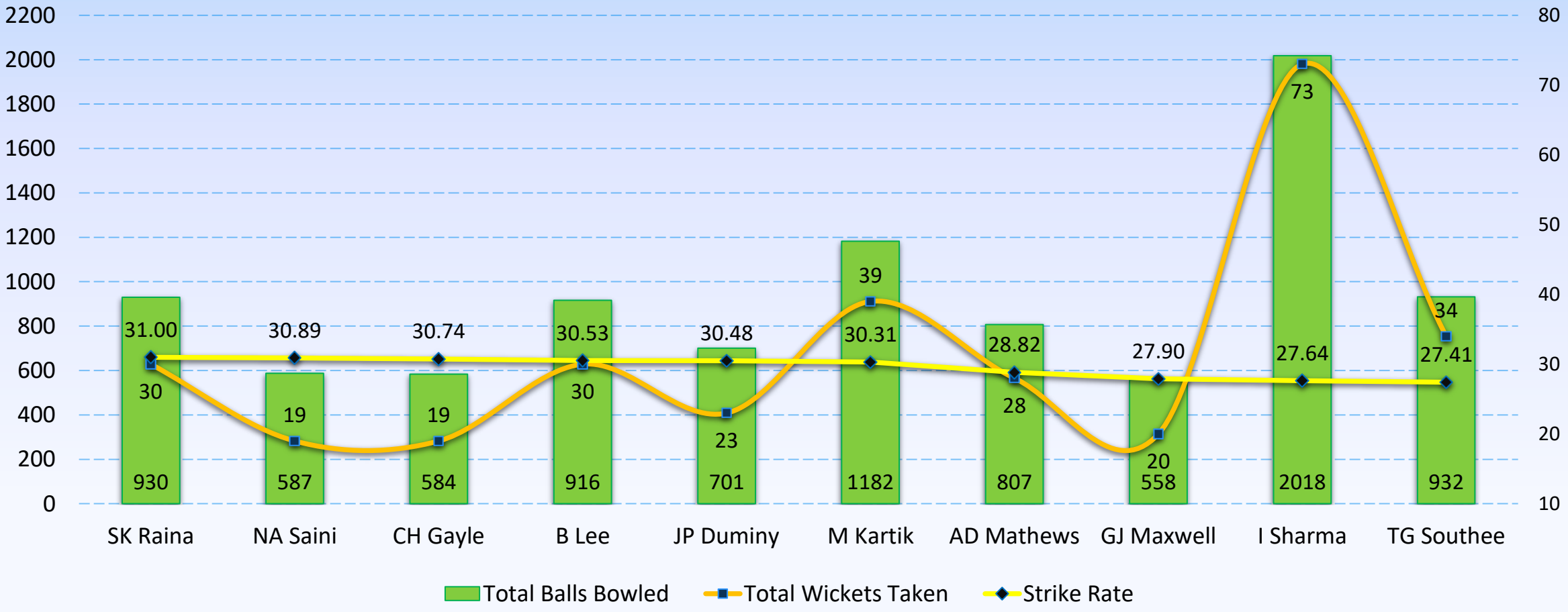
3. **Batsman Who Bowled More Than 500 Balls And Have Best Strike Rate.**

- *SELECT bowler AS player_name, COUNT(ball) AS TotalBallsBowled, SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) AS TotalWicketsTaken,*
- *(COUNT(ball) * 1.0 / NULLIF(SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END), 0)) AS StrikeRate*
- *FROM ipl_ball GROUP BY bowler*
- *HAVING COUNT(*) > 0 AND COUNT(ball) >= 500 AND SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) > 0*
- *ORDER BY StrikeRate DESC LIMIT 10;*

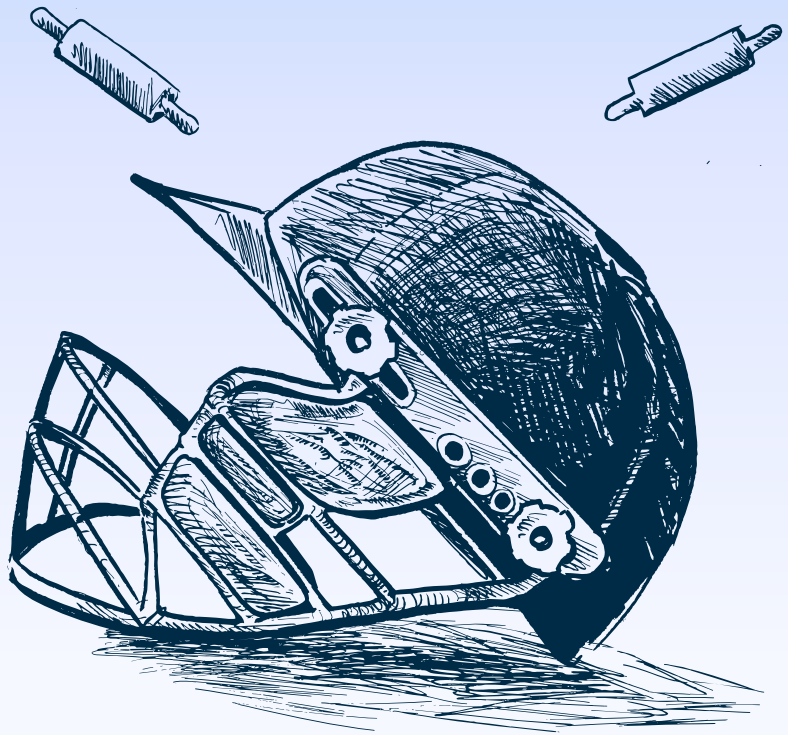
Wicket Taker Bowler’s

Rank	Player Name	Balls Bowled	Wickets Taken	Strike Rate
1	SK Raina	930	30	31.00
2	NA Saini	587	19	30.89
3	CH Gayle	584	19	30.74
4	B Lee	916	30	30.53
5	JP Duminy	701	23	30.48
6	M Kartik	1182	39	30.31
7	AD Mathews	807	28	28.82
8	GJ Maxwell	558	20	27.90
9	I Sharma	2018	73	27.64
10	TG Southee	932	34	27.41

Wicket Taker Bowler's

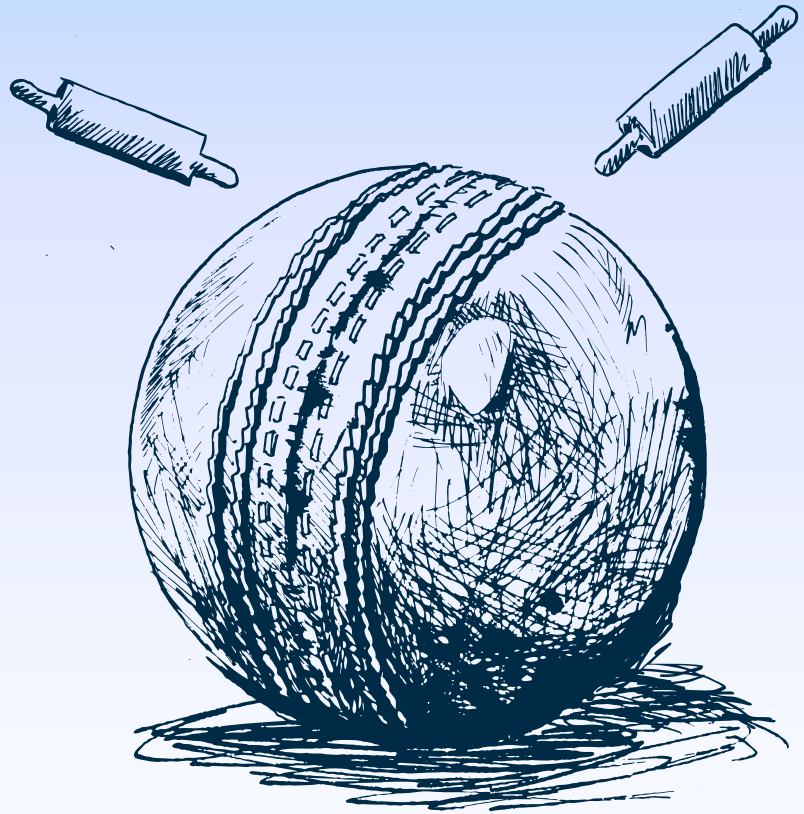


All-Rounder’s Criteria :

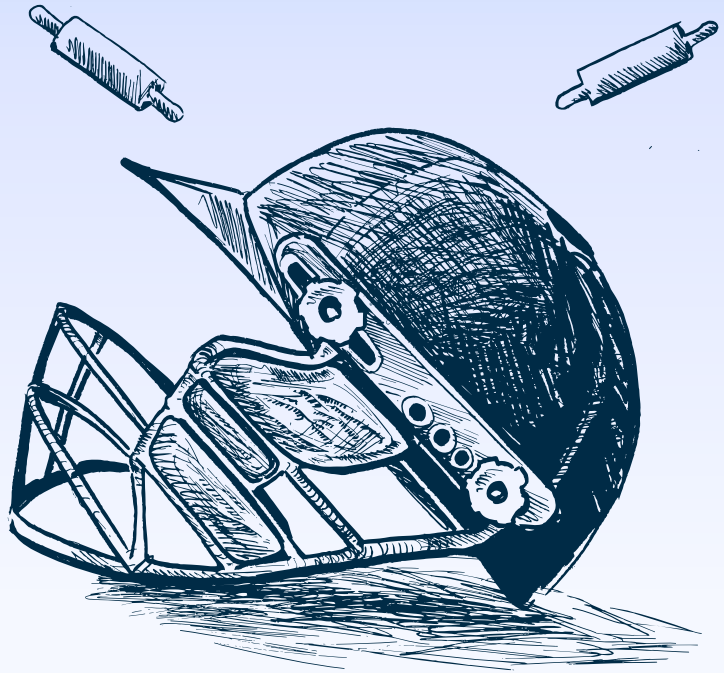


All-Rounder’s

The priority is to get the All-Rounder’s Strike Rate from Batter’s And Bowler’s SR’s.

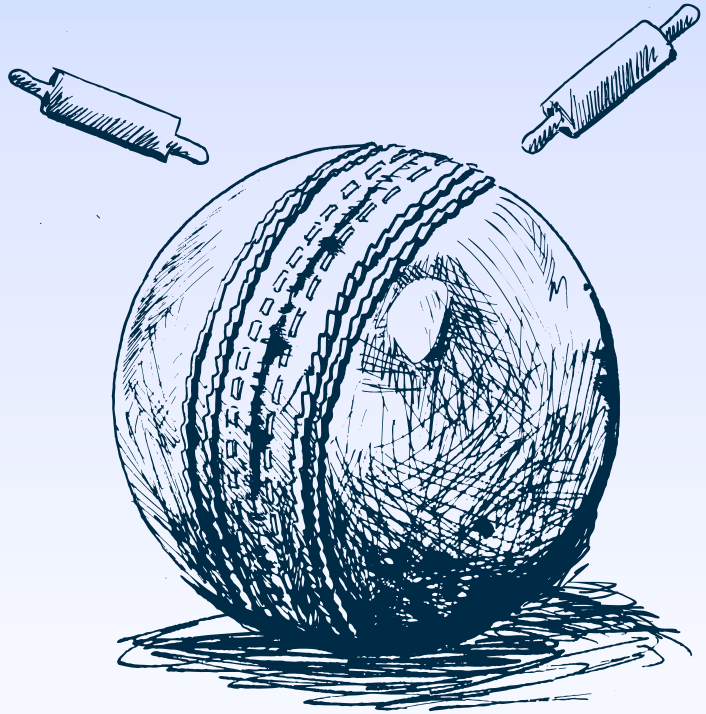


All-Rounder’s Criteria (OR) Explanation :



To Find The All-Rounder’s I Use The IPL Ball Table Only.

- *First I Find The Batter’s Strike Rate.*
- *Second I Find The Bowler’s Strike Rate.*
- *Third In The Last I Combine Both The Strike Rate’s To Find The All-Rounder’s Who Have Batter’s And Bowler’s Strike Rate.*



All-Rounder’s Query :

1. Batter’s Strike Rate.

- *SELECT batsman AS PlayerName, SUM(CASE WHEN extras_type != 'wides' THEN batsman_runs ELSE 0 END) AS RunsScored,*
- *COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE 0 END) AS BallsFaced,*
- *(SUM(CASE WHEN extras_type != 'wides' THEN batsman_runs ELSE 0 END) * 100.0 / NULLIF(COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE 0 END), 0)) AS StrikeRate*
- *FROM ipl_ball WHERE extras_type != 'wides' GROUP BY batsman HAVING COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE 0 END) >= 500*
- *ORDER BY StrikeRate DESC LIMIT 10;*

2. Batsman Who Have Best Strike Rate.

- *SELECT bowler AS PlayerName, COUNT(ball) AS TotalBallsBowled, SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) AS TotalWicketsTaken,*
- *(COUNT(ball) * 1.0 / NULLIF(SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END), 0)) AS StrikeRate*
- *FROM ipl_ball GROUP BY bowler HAVING COUNT(*) > 0 AND COUNT(ball) >= 300 AND SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) > 0*
- *ORDER BY StrikeRate DESC LIMIT 10;*

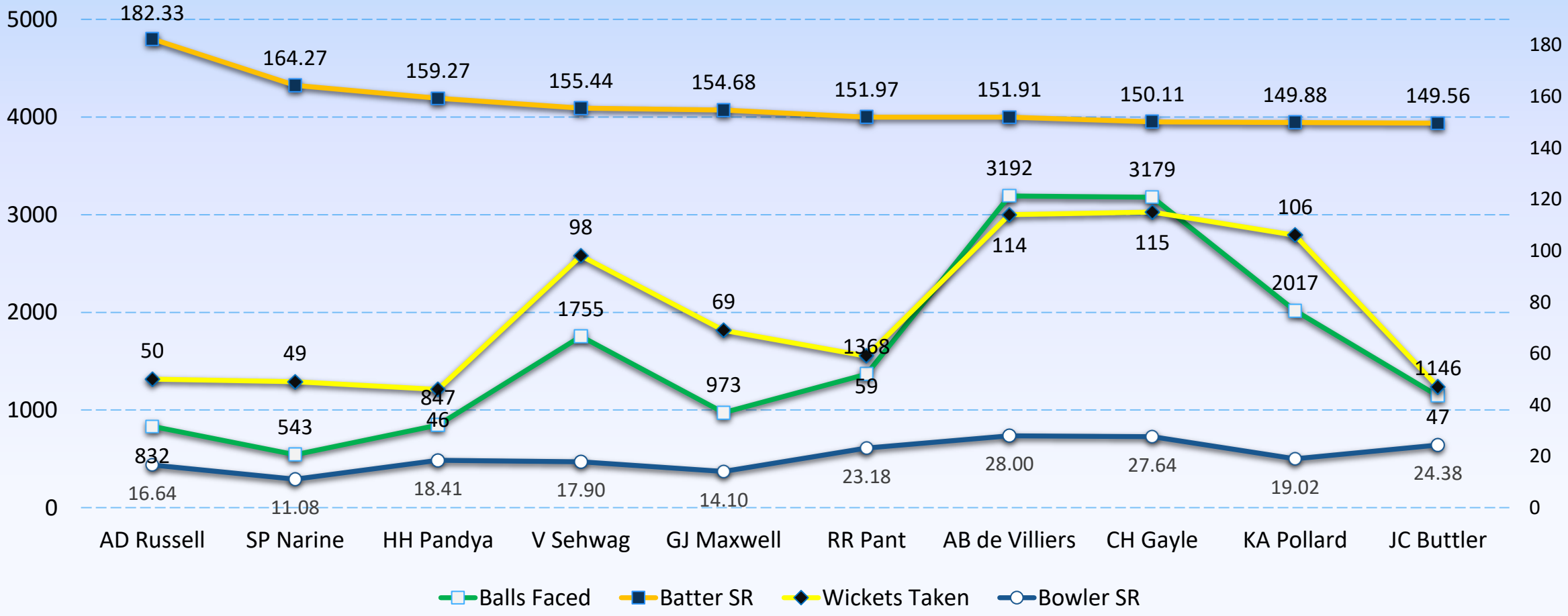
3. Batsman Who Bowled More Than 500 Balls And Have Best Strike Rate.

- *SELECT batsman AS PlayerName, COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE NULL END) AS BallsFaced,*
- *(SUM(batsman_runs) * 100.0 / NULLIF(COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE NULL END), 0)) AS Batter_StrikeRate,*
- *SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) AS total_wickets_taken, (COUNT(ball) * 1.0 / NULLIF(SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END), 0)) AS Bowler_StrikeRate*
- *FROM ipl_ball WHERE extras_type != 'wides' GROUP BY batsman HAVING COUNT(CASE WHEN extras_type != 'wides' THEN 1 ELSE NULL END) >= 500 AND SUM(CASE WHEN is_wicket = 1 THEN 1 ELSE 0 END) > 0 AND COUNT(ball) >= 300*
- *ORDER BY Batter_StrikeRate DESC, Bowler_StrikeRate DESC LIMIT 10;*

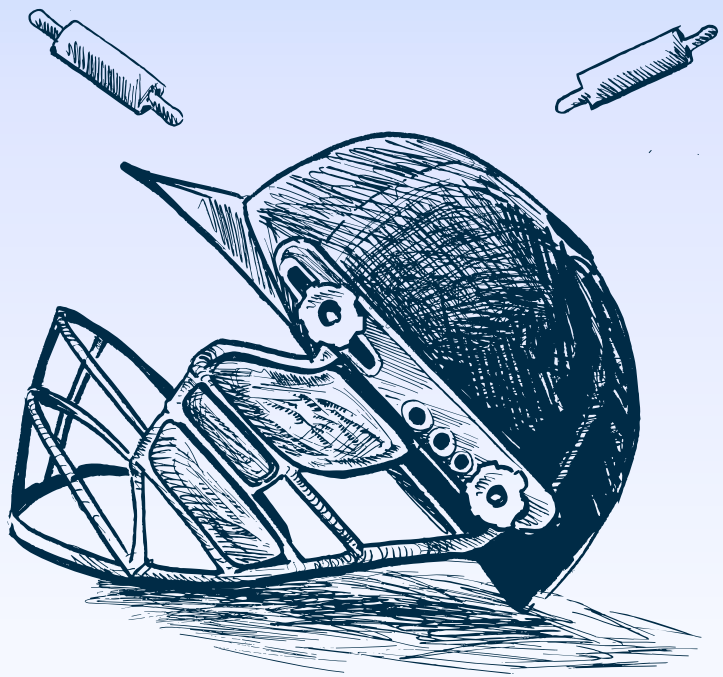
All-Rounder’s

Rank	Player Name	Balls Faced	Batter SR	Wickets Taken	Bowler SR
1	AD Russell	832	182.33	50	16.64
2	SP Narine	543	164.27	49	11.08
3	HH Pandya	847	159.27	46	18.41
4	V Sehwag	1755	155.44	98	17.90
5	GJ Maxwell	973	154.68	69	14.10
6	RR Pant	1368	151.97	59	23.18
7	AB de Villiers	3192	151.91	114	28.00
8	CH Gayle	3179	150.11	115	27.64
9	KA Pollard	2017	149.88	106	19.02
10	JC Buttler	1146	149.56	47	24.38

All-Rounder's

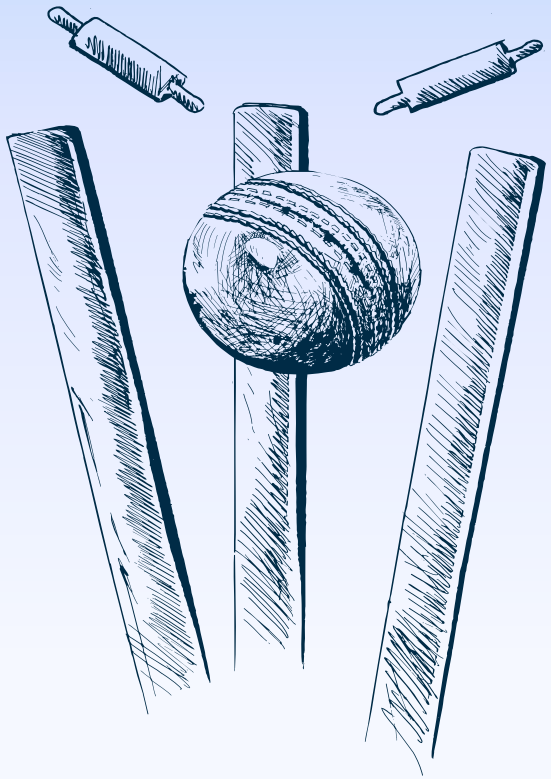


Wicket Keeper Criteria’s (OR) Explanation :



INTRODUCTION -

To find the top listed wicket keepers we have find the wicket keeper who’s skills are a good catcher, ability to read the ball off the pitch and effecting stumps as many as possible.



Wicket Keeper’s Criteria–

Find the top 10 wicket keepers who catched and bowled atleast 200 balls and stumped the wickets atleast 200 times.(use dismissals to find the best wicket keeper).

Additional Questions for Final Assessment

Additional Questions Table Creation :

Deliveries Table Creation Query.

```
CREATE TABLE DELIVERIES
(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);
```

Importing CSV File.

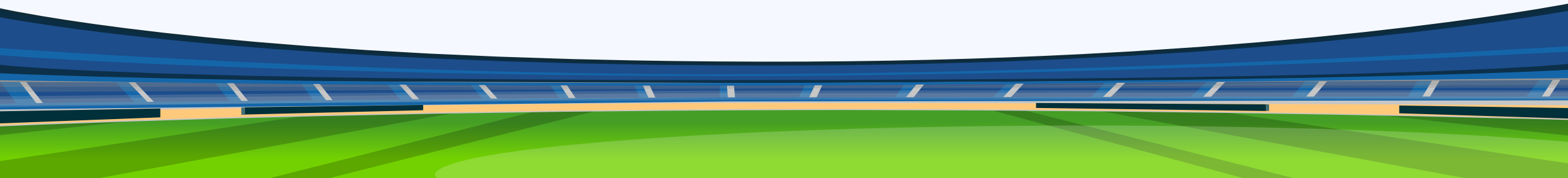
```
COPY IPL_BALL
FROM 'C:\Program Files\PostgreSQL\15\data\A - SQL - Data Sources\M1_T4_V1 Restore\IPL Dataset\IPL_Ball.csv' DELIMITER ',' CSV HEADER;
```

Matches Table Creation Query.

```
CREATE TABLE MATCHES
(ID int, CITY varchar, DATE date, PLAYER_OF_MATCH varchar, VENUE varchar, NEUTRAL_VENUE int, 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);
```

Importing CSV File.

```
COPY IPL_MATCHES
FROM 'C:\Program Files\PostgreSQL\15\data\A - SQL - Data Sources\M1_T4_V1 Restore\IPL Dataset\IPL_matches.csv' DELIMITER ',' CSV HEADER;
```



Question 1 - Get the count of cities that have hosted an IPL match?

Query :

```
SELECT COUNT(DISTINCT city) AS total_cities
FROM matches;
```

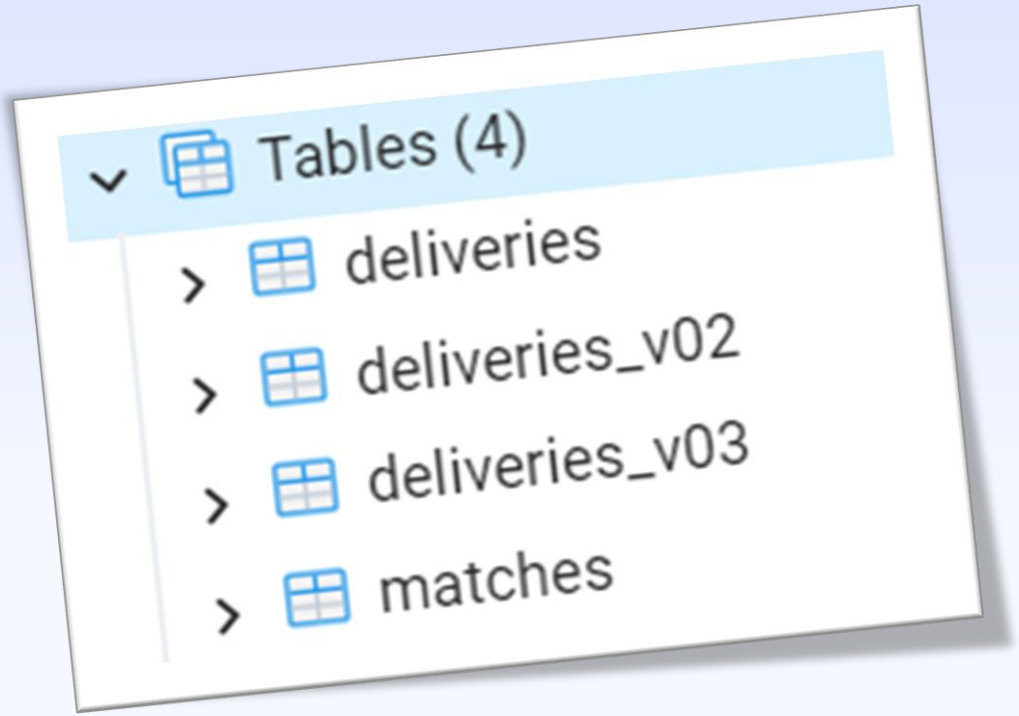
Output

Total_Cities
33

Question 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].?

Query :

```
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;
```



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

Query :

```
SELECT  ball_result,
        COUNT(*) AS count
FROM deliveries_v02
WHERE ball_result IN ('boundary', 'dot')
GROUP BY ball_result;
```

Output

Ball_Result	Count
Boundary	31468
Dot	67841

Question 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?

Query –

```
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	Total Boudaries
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

Question 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?

Query –

```
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	Total Dot Balls
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
NA	71

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

Query –

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

Dismissal Kind	Total Dismissals
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

Question 7 - Write a query to get the top 5 bowlers who conceded maximum extra runs from the deliveries table?

Query –

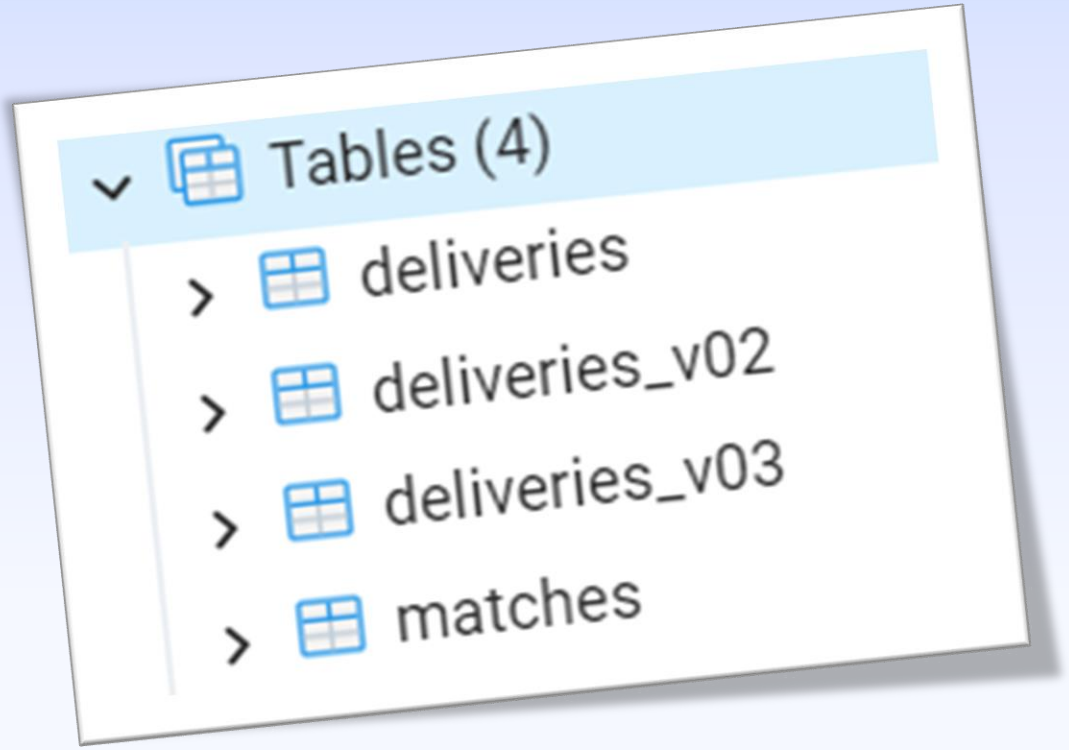
```
SELECT
    bowler,
    SUM(extra_runs) AS total_extra_runs
FROM deliveries_v02
GROUP BY bowler
ORDER BY total_extra_runs DESC
LIMIT 5;
```

Bowler	Total Extra Runs
SL Malinga	293
P Kumar	236
UT Yadav	226
DJ Bravo	210
B Kumar	201

Question 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

Query –

```
CREATE TABLE deliveries_v03 AS
        SELECT d.*, m.venue, m.date
        FROM deliveries_v02 AS d
        JOIN matches AS m ON d.id = m.id;
```



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

Query –

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

Venue	Total Runs Scored
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

Question 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?

Query –

```
SELECT
    EXTRACT(YEAR FROM date) AS year,
    SUM(total_runs) AS total_runs_scored
FROM deliveries_v03
WHERE venue = 'Eden Gardens'
GROUP BY EXTRACT(YEAR FROM date)
ORDER BY total_runs_scored DESC;
```

Years	Total Runs Scored
2018	2885
2019	2651
2015	2386
2013	2304
2017	2194
2010	2167
2016	2073
2012	2012
2011	1854
2008	1843
2014	1289

THANK YOU