

SQL Final Project

IPL AUCTION OF 14TH SEASON 2021



Name:-Vikas Kumar

Gmail:-vikasdbg102@gmail.com

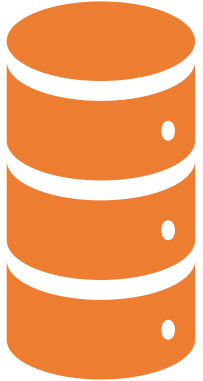
Internshala training-Data Science course

CONTENTS



- ❖ Create database, tables and import IPL-Ball & IPL-matches data's
- ❖ Bidding on batters
 - Aggressive batsman
 - Anchor batsman
 - Hard hitters
- ❖ Bidding on bowlers
 - Economical bowlers
 - Wicket taking bowlers
- ❖ Bidding on All rounders
- ❖ Bidding for wicketkeeper
- ❖ Additional questions for assessment
- ❖ Explanations
- ❖ Conclusion

Create database, tables and import IPL-Ball & IPL-matches data's



Create table Deliveries and import data from IPL-Ball

- 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);
- copy Deliveries from 'C:\Program Files\PostgreSQL\16\data\data_copy\IPL_Ball.csv' delimiter ',' csv header;

Create table Matches and import data from IPL-matches

- create table Matches(id int, city varchar, date varchar, 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);
- copy Matches from 'C:\Program Files\PostgreSQL\16\data\data_copy\IPL_matches.csv' delimiter ',' csv header;

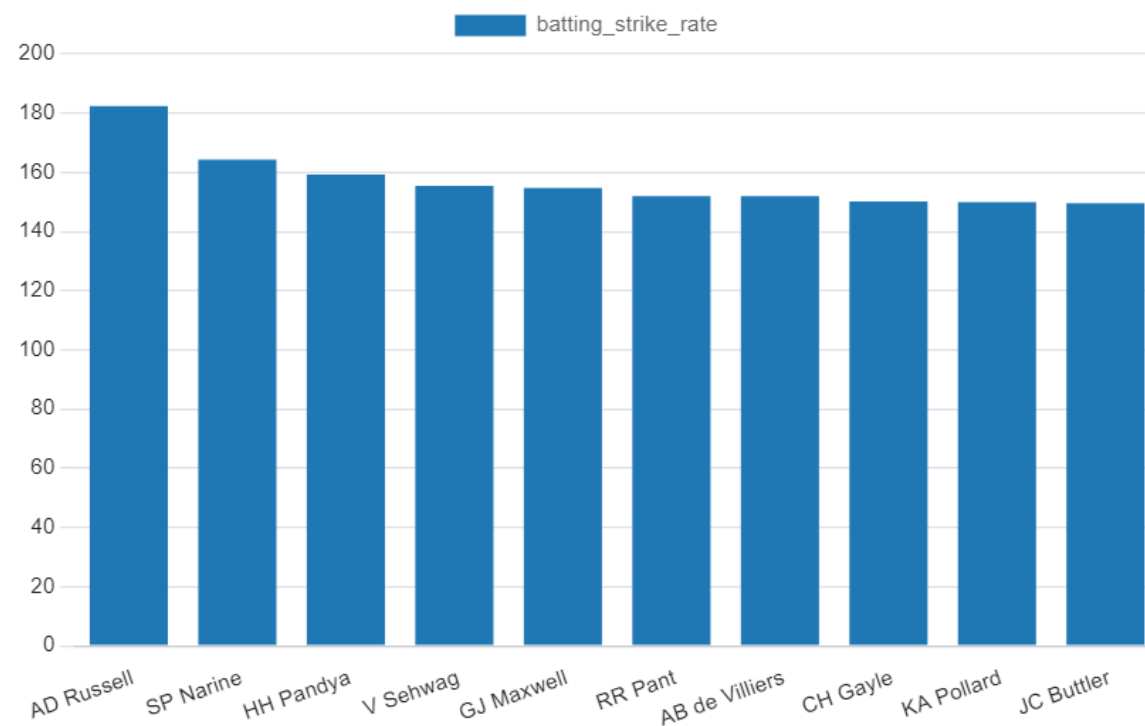


Bidding on Aggressive batsman

Bidding on batters

select batsman, batsman_total_score, total_ball_faced, round(cast(batsman_total_score as decimal)/cast(total_ball_faced as decimal)*100, 4) AS batting_strike_rate from (select batsman, sum(batsman_runs) as batsman_total_score, count(ball) as total_ball_faced from Deliveries where extras_type not in('wides') group by batsman) as subquery where total_ball_faced>=500 order by batting_strike_rate desc limit 10;

	batsman character varying	batsman_total_score bigint	total_ball_faced bigint	batting_strike_rate numeric
1	AD Russell	1517	832	182.3317
2	SP Narine	892	543	164.2726
3	HH Pandya	1349	847	159.2680
4	V Sehwag	2728	1755	155.4416
5	GJ Maxwell	1505	973	154.6763
6	RR Pant	2079	1368	151.9737
7	AB de Villiers	4849	3192	151.9110
8	CH Gayle	4772	3179	150.1101
9	KA Pollard	3023	2017	149.8761
10	JC Buttler	1714	1146	149.5637

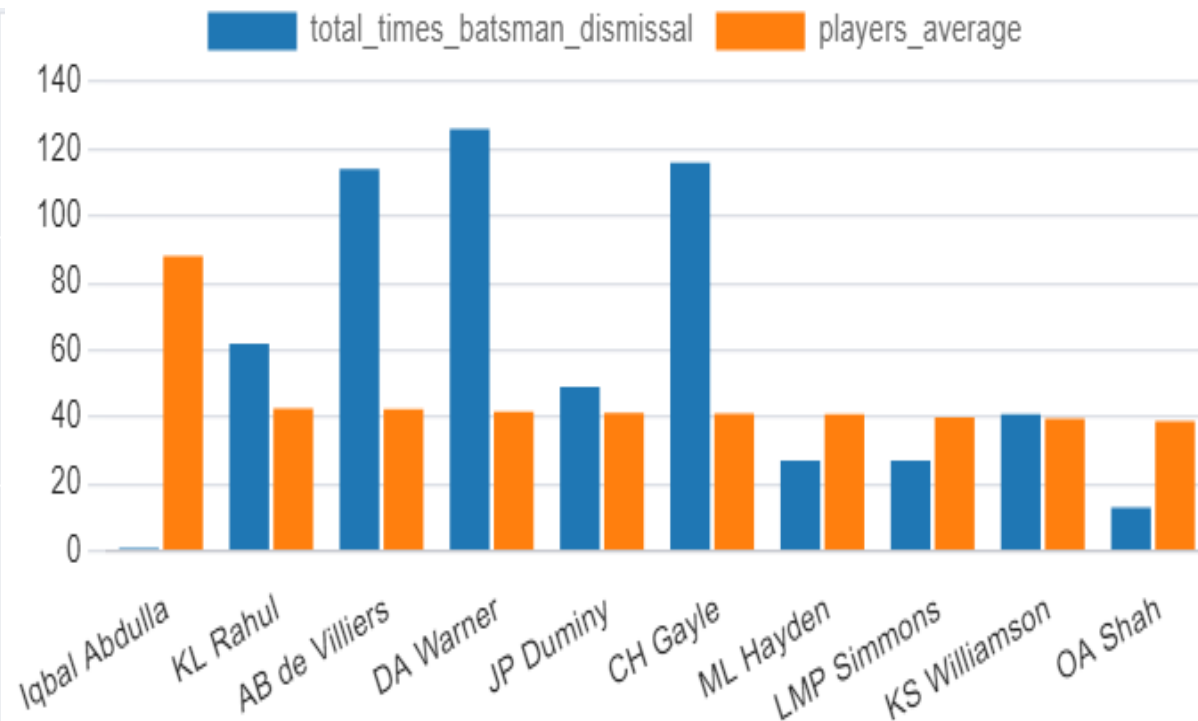




Bidding on anchor batsman

select batsman, played_IPL_season, batsman_total_score, total_times_dismissal, round(cast(batsman_total_score as decimal)/cast(total_times_dismissal as decimal),4) AS players_average from (select batsman, count(distinct substring(date from 7 for 4)) as played_IPL_season, sum(case when dismissal_kind in ('NA') then batsman_runs else 0 end) as batsman_total_score, count(case when dismissal_kind not in ('NA') then is_wicket end) as total_times_dismissal from (select a.*, b.date from Deliveries as a left join Matches as b on a.id=b.id) group by batsman) as subquery where total_times_dismissal>=1 and played_IPL_season>2 order by players_average desc limit 10;

	batsman character varying	played_ipl_season bigint	batsman_total_score bigint	total_times_dismissal bigint	players_average numeric
1	Iqbal Abdulla	8	88	1	88.0000
2	KL Rahul	7	2647	62	42.6935
3	AB de Villiers	13	4846	114	42.5088
4	DA Warner	11	5250	126	41.6667
5	JP Duminy	8	2026	49	41.3469
6	CH Gayle	12	4770	116	41.1207
7	ML Hayden	3	1107	27	41.0000
8	LMP Simmons	4	1079	27	39.9630
9	KS Williamson	6	1619	41	39.4878
10	OA Shah	4	504	13	38.7692









Bidding on batters continues....



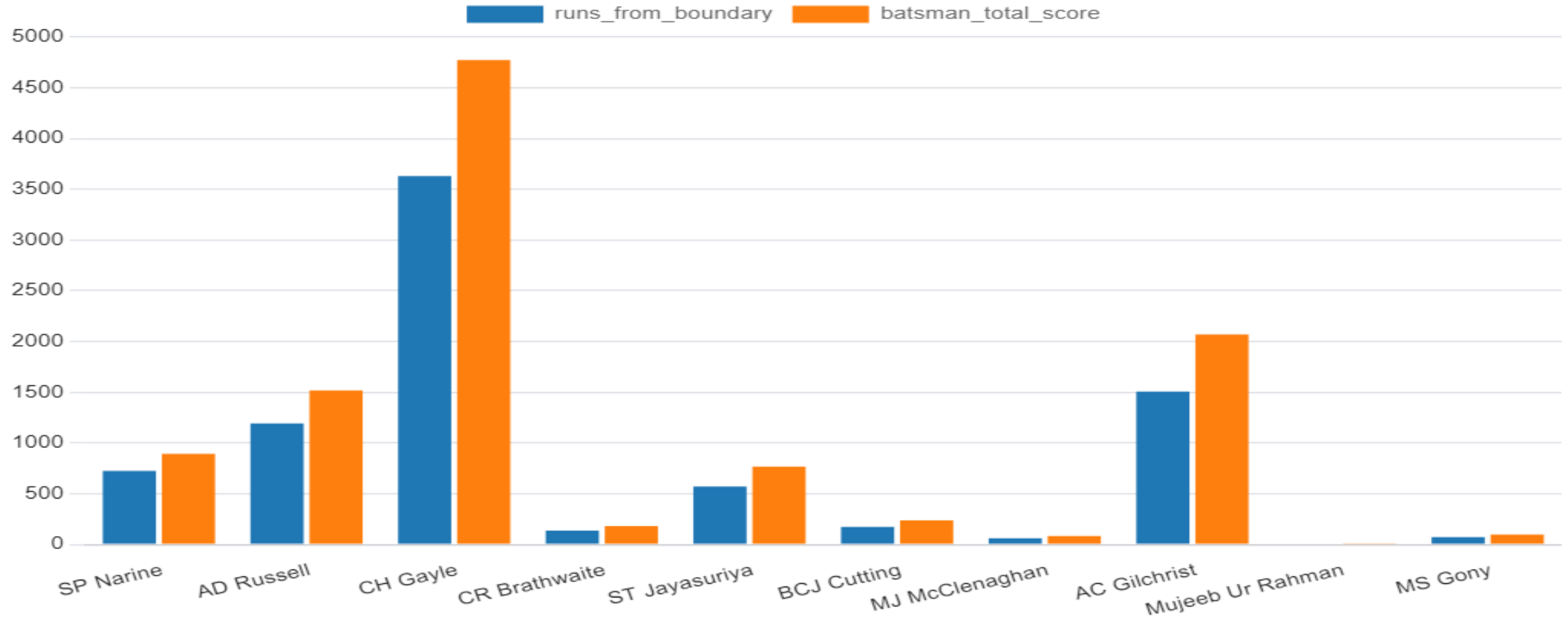
Bidding on hard hitters

```
select batsman, played_IPL_season, batsman_total_score, boundary_hits, runs_from_boundary,
round(cast(runs_from_boundary as decimal)/cast(batsman_total_score as decimal)*100, 4) AS
boundary_percentage from (select batsman, count(distinct substring(date from 7 for 4)) as played_IPL_season,
sum(batsman_runs) as batsman_total_score, count(case when batsman_runs in (4,6) then batsman_runs end) as
boundary_hits, sum(case when batsman_runs in (4,6) then batsman_runs else 0 end) as runs_from_boundary from
(select a.*, b.date from Deliveries as a left join Matches as b on a.id=b.id) group by batsman) as subquery where
played_IPL_season>2 order by boundary_precentage desc limit 10;
```

	batsman character varying 	played_ipl_season bigint 	batsman_total_score bigint 	boundary_hits bigint 	runs_from_boundary bigint 	boundary_precentage numeric 
1	SP Narine	9	892	155	724	81.1659
2	AD Russell	8	1517	234	1194	78.7080
3	CH Gayle	12	4772	733	3630	76.0687
4	CR Brathwaite	4	181	26	136	75.1381
5	ST Jayasuriya	3	768	123	570	74.2188
6	BCJ Cutting	5	238	34	174	73.1092
7	MJ McClenaghan	5	85	12	62	72.9412
8	AC Gilchrist	6	2069	331	1508	72.8855
9	Mujeeb Ur Rahm...	3	11	2	8	72.7273
10	MS Gony	6	99	14	72	72.7273



Bidding on hard hitters continues....

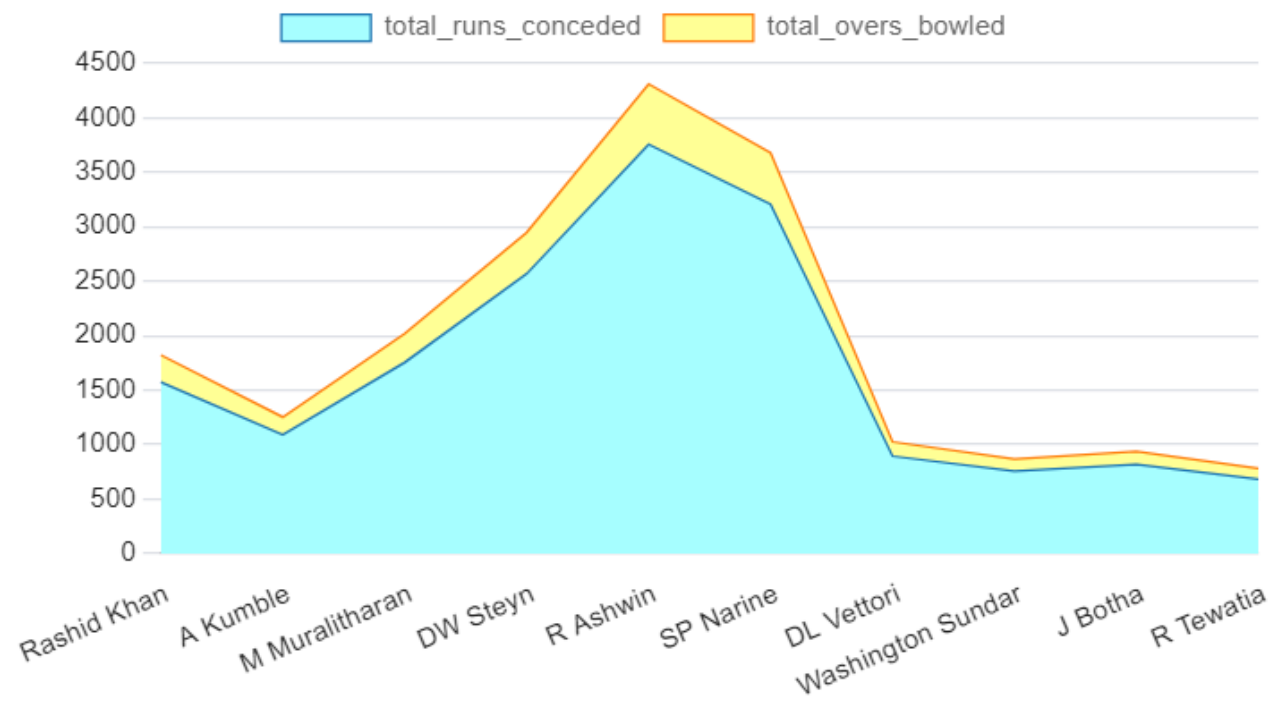


Bidding on bowlers

Bidding on Economical bowlers

select bowler, total_runs_conceded, total_balled, total_overs_bowled, round(cast(total_runs_conceded as decimal)/cast(total_overs_bowled as decimal), 4) AS bowling_economy from (select bowler, sum(total_runs) as total_runs_conceded, count(ball) as total_balled, round(cast(count(over) as decimal)/6, 4) as total_overs_bowled from Deliveries group by bowler) as subquery where total_balled >= 500 order by bowling_economy asc limit 10;

	bowler character varying	total_runs_conceded bigint	total_balled bigint	total_overs_bowled numeric	bowling_economy numeric
1	Rashid Khan	1573	1490	248.3333	6.3342
2	A Kumble	1089	983	163.8333	6.6470
3	M Muralitharan	1755	1577	262.8333	6.6772
4	DW Steyn	2568	2276	379.3333	6.7698
5	R Ashwin	3756	3327	554.5000	6.7737
6	SP Narine	3208	2824	470.6667	6.8159
7	DL Vettori	894	785	130.8333	6.8331
8	Washington Sundar	758	660	110.0000	6.8909
9	J Botha	818	709	118.1667	6.9224
10	R Tewatia	684	587	97.8333	6.9915



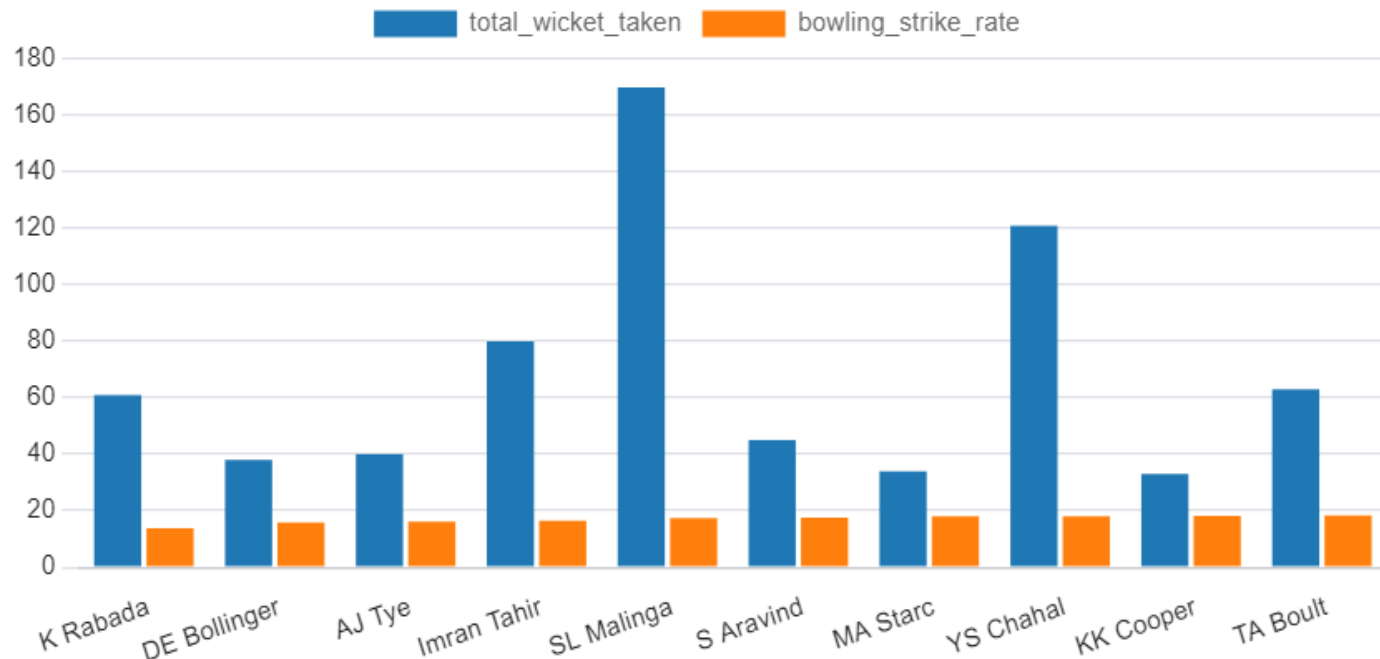


Bidding on Economical bowlers

Bidding on bowlers continues....









select bowler, total_balled, total_wicket_taken, round(cast(total_balled as decimal)/cast(total_wicket_taken as decimal), 4) AS bowling_strike_rate from (select bowler, count(ball) as total_balled, count(case when dismissal_kind not in ('NA', 'run out') then dismissal_kind end) as total_wicket_taken from Deliveries group by bowler) as subquery where total_balled>=500 order by bowling_strike_rate asc limit 10;

	bowler character varying	total_balled bigint	total_wicket_taken bigint	bowling_strike_rate numeric
1	K Rabada	840	61	13.7705
2	DE Bollinger	600	38	15.7895
3	AJ Tye	645	40	16.1250
4	Imran Tahir	1314	80	16.4250
5	SL Malinga	2974	170	17.4941
6	S Aravind	788	45	17.5111
7	MA Starc	612	34	18.0000
8	YS Chahal	2188	121	18.0826
9	KK Cooper	600	33	18.1818
10	TA Boult	1152	63	18.2857





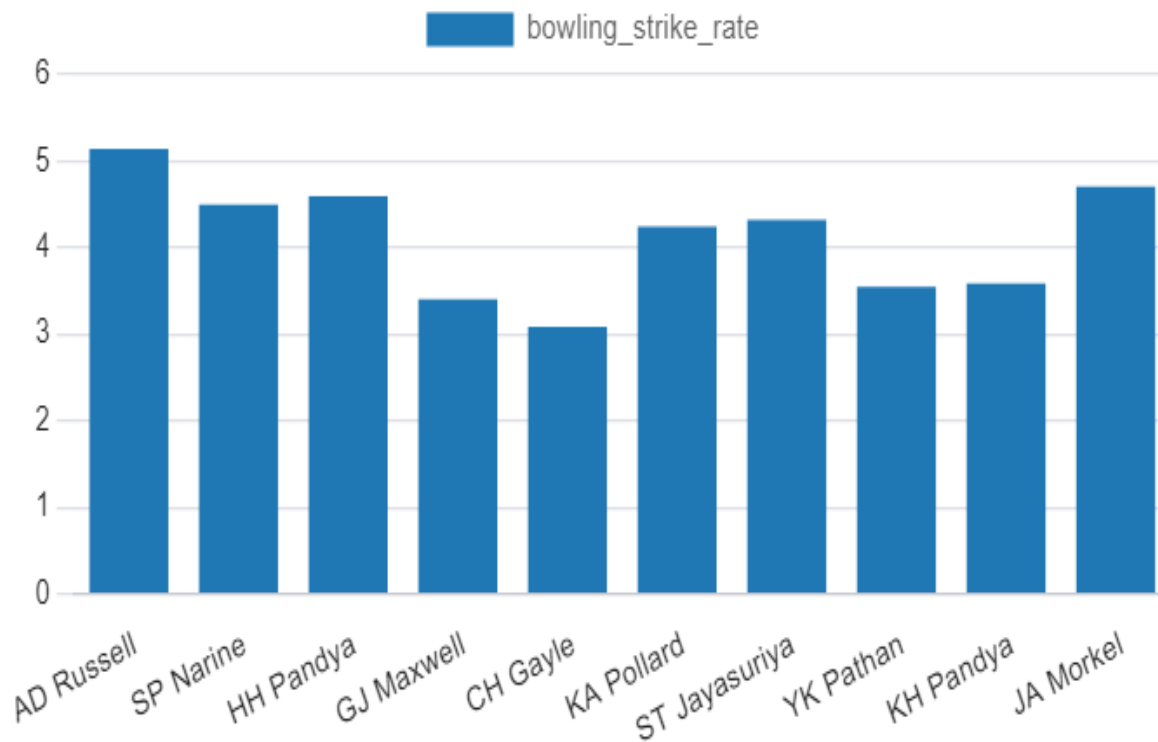
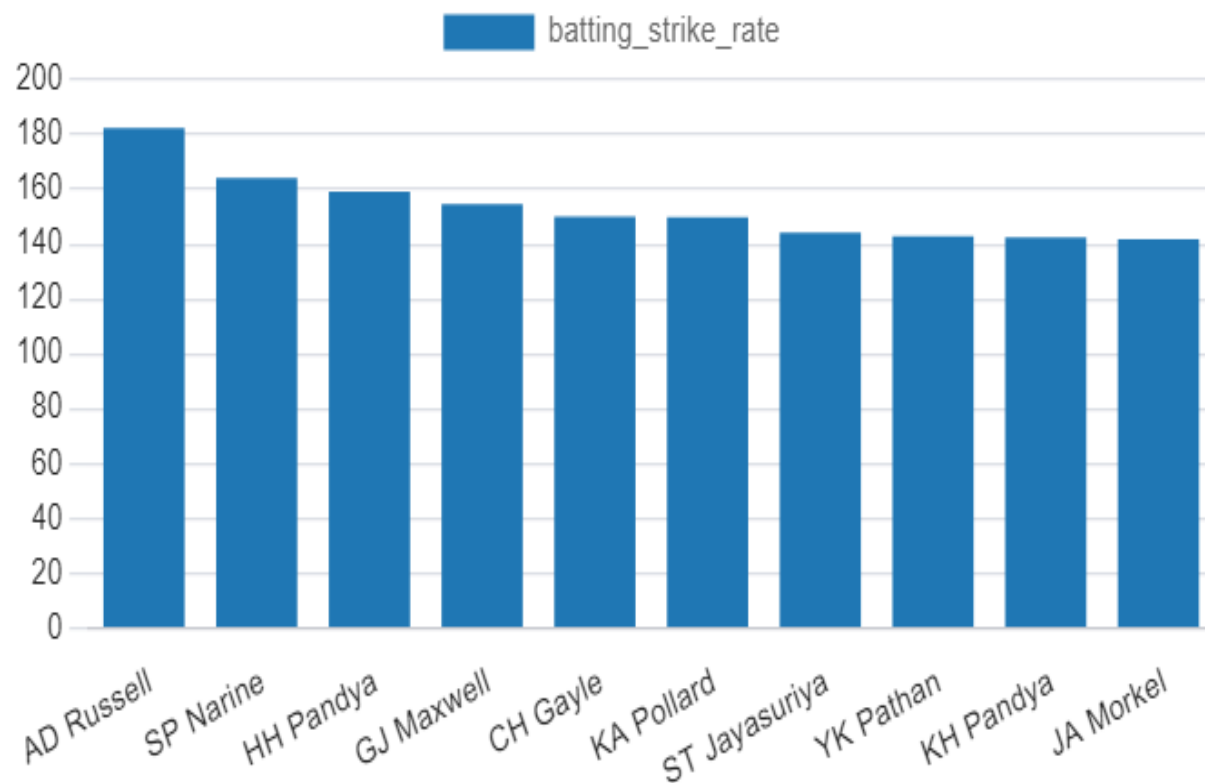
with batting_stats as (select batsman, batsman_total_score, total_ball_faced from (select batsman, sum(batsman_runs) as batsman_total_score, count(ball) as total_ball_faced from Deliveries where extras_type not in('wides') group by batsman) as subquery where total_ball_faced >= 500), bowling_stats as (select bowler, total_balled, total_wicket_taken from (select bowler, count(ball) as total_balled, count(case when dismissal_kind not in ('NA','run out') then dismissal_kind end) as total_wicket_taken from Deliveries group by bowler) as subquery where total_balled >= 300) select a.batsman as all_rounder, a.batsman_total_score, a.total_ball_faced, b.bowler as bowler, b.total_wicket_taken, b.total_balled, round(cast(a.batsman_total_score as decimal)/cast(a.total_ball_faced as decimal)*100, 4) as batting_strike_rate, round(cast(b.total_wicket_taken as decimal)/cast(b.total_balled as decimal)*100, 4) as bowling_strike_rate from batting_stats a inner join bowling_stats b on a.batsman = b.bowler order by batting_strike_rate desc, bowling_strike_rate asc limit 10;

	all_rounder character varying 	batsman_total_score bigint 	total_ball_faced bigint 	bowler character varying 	total_wicket_taken bigint 	total_balled bigint 	batting_strike_rate numeric 	bowling_strike_rate numeric 
1	AD Russell	1517	832	AD Russell	61	1186	182.3317	5.1433
2	SP Narine	892	543	SP Narine	127	2824	164.2726	4.4972
3	HH Pandya	1349	847	HH Pandya	42	914	159.2680	4.5952
4	GJ Maxwell	1505	973	GJ Maxwell	19	558	154.6763	3.4050
5	CH Gayle	4772	3179	CH Gayle	18	584	150.1101	3.0822
6	KA Pollard	3023	2017	KA Pollard	60	1414	149.8761	4.2433
7	ST Jayasuriya	768	532	ST Jayasuriya	13	301	144.3609	4.3189
8	YK Pathan	3204	2241	YK Pathan	42	1184	142.9719	3.5473
9	KH Pandya	1000	702	KH Pandya	46	1283	142.4501	3.5853
10	JA Morkel	974	686	JA Morkel	85	1807	141.9825	4.7039

Bidding on All-rounders continues...



- For batting strike rate and bowling strike rate, I **multiply by 100** to get the actual form of strike rate figures.
- Because of **huge the difference in numeric values**, I can't plot them on the same graph. So, I plotted separately for all-rounders with bowling and batting strike rates



- ❖ I will ensure that in the **list of wicketkeepers, bowlers' names are not listed** because it can cause difficulties in wicketkeeping when they themselves are bowling.
- ❖ A **wicketkeeper with a good batting strike rate** will be a valuable addition to the team.
- ❖ I will extract the **fielder names who fall under categories of dismissal-kind equal to 'catch' and 'run out'**, as it shows good fielder criteria.
- ❖ If we have **historical data** of **fall of wicket because of wicketkeepers with any dismissal-kind**, then we will count the total falls of wicket for the same and target the highest wicketkeeper.



Sample code based on above criteria

```
select w.player_name as wicketkeeper_name,case when w.batting_strike_rate>(select
avg(batting_strike_rate) from wicketkeepers) then 'Good Batting Strike Rate' else 'Regular
Batting Strike Rate' end as batting_performance, count(f.fall_of_wicket) as
total_falls_of_wicket from wicketkeepers w left join bowlers b on
w.player_name=b.player_name join historical_data h on w.player_name =
h.wicketkeeper_name join falls_of_wicket f on h.match_id = f.match_id where b.player_name
is NULL and f.dismissal_kind in ('catch', 'run out') group by w.player_name order by
total_falls_of_wicket desc limit 10;
```

Additional questions for assessment

1. Count of cities hosted an IPL match

- select count(distinct city) as cities_hosted_IPL_match from Matches;

	cities_hosted_ipL_match bigint
1	33

2. Create table deliveries_v02

- 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);



3. Total number of boundaries and dot balls

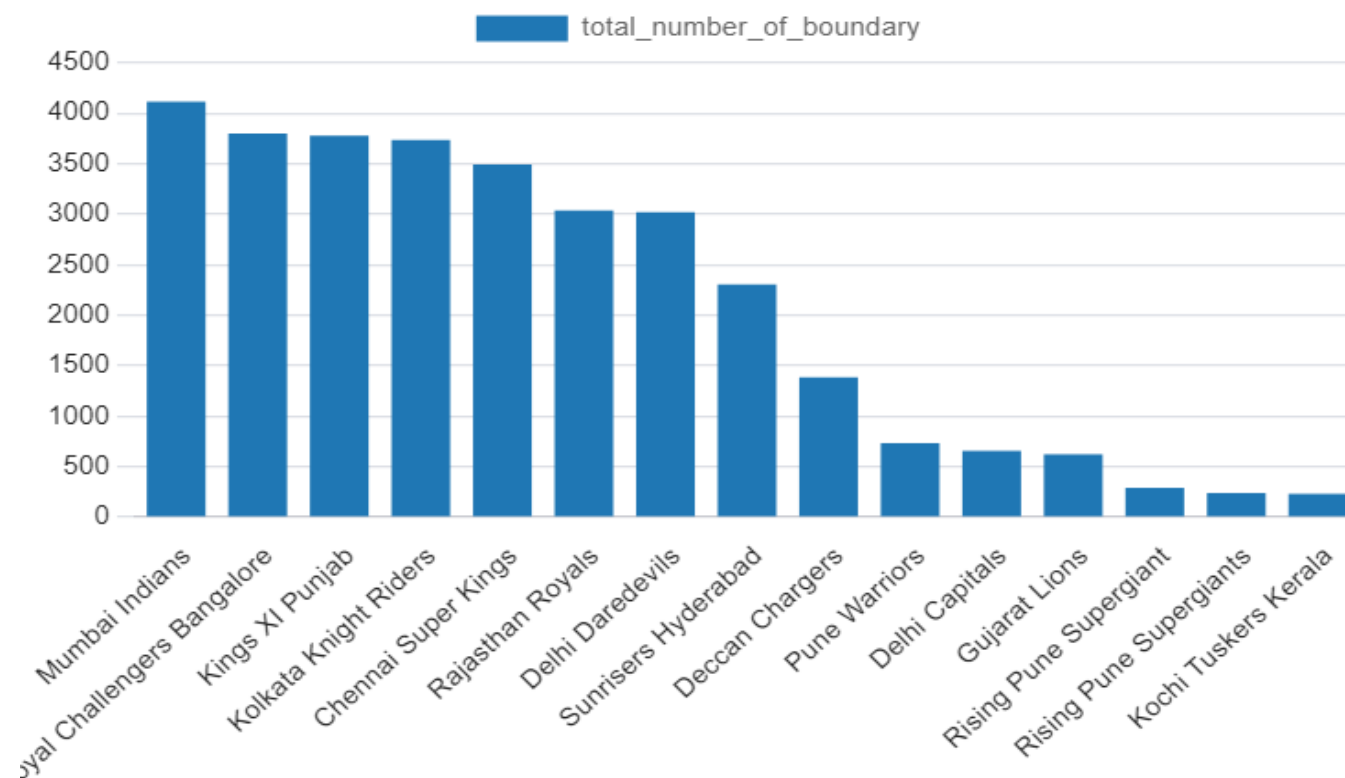
- create table deliveries_v02 as (select *, case when total_runs>=4 then 'boundary' when total_runs=0 then 'dot' else 'other' end as ball_result from Deliveries);

	total_number_of_boundary bigint	total_number_of_dot bigint
1	31468	67841

4. Total number of boundaries scored by each team

- select batting_team, count(case when ball_result in ('boundary') then ball_result end) as total_number_of_boundary from deliveries_v02 group by batting_team order by total_number_of_boundary desc;

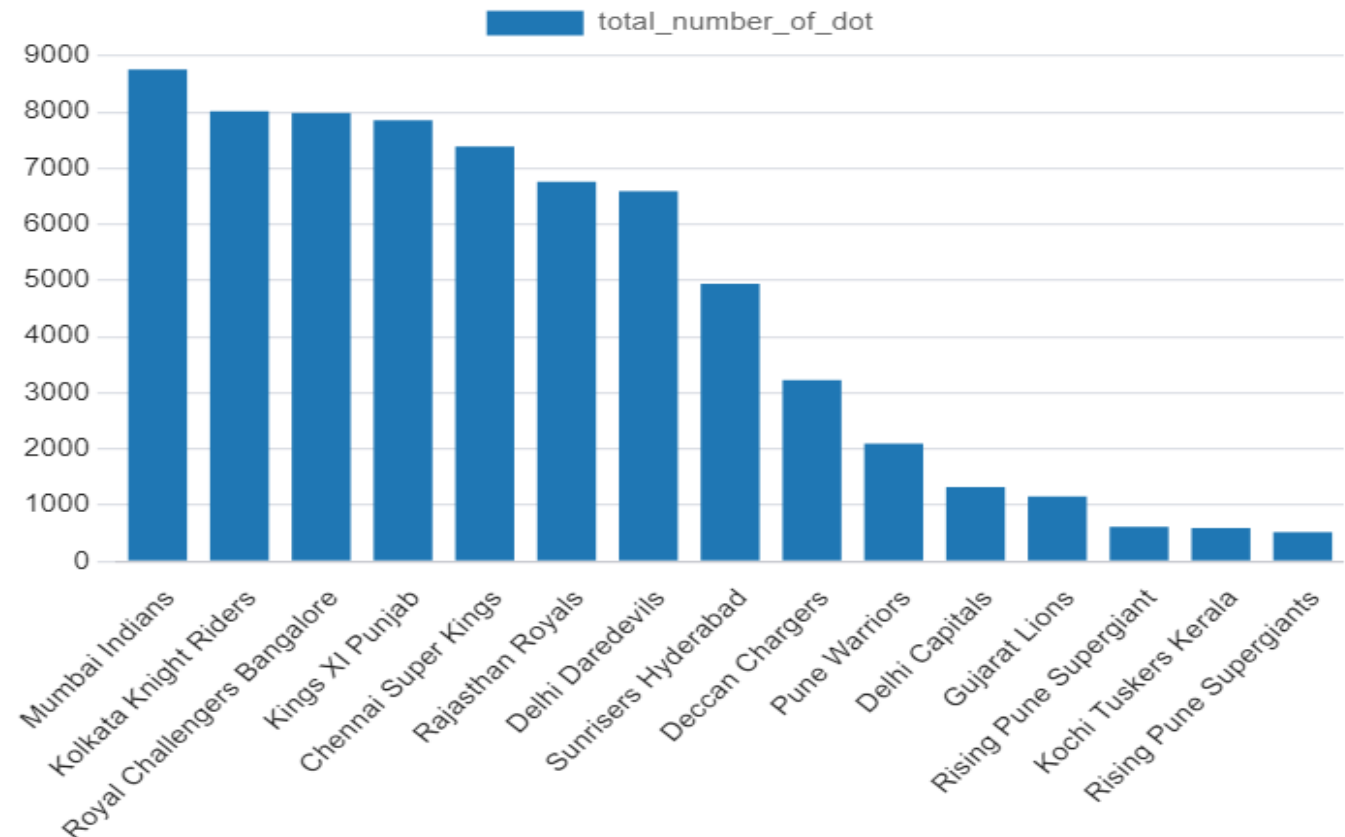
	batting_team character varying	total_number_of_boundary bigint
1	Mumbai Indians	4118
2	Royal Challengers Bangalore	3800
3	Kings XI Punjab	3780
4	Kolkata Knight Riders	3739
5	Chennai Super Kings	3496
6	Rajasthan Royals	3041
7	Delhi Daredevils	3022
8	Sunrisers Hyderabad	2306
9	Deccan Chargers	1387
10	Pune 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



5. Total number of dot balled by each team

➤ select batting_team, count(case when ball_result in ('dot') then ball_result end) as total_number_of_dot from deliveries_v02 group by batting_team order by total_number_of_dot desc;

	batting_team character varying	total_number_of_dot bigint
1	Mumbai Indians	8756
2	Kolkata Knight Riders	8017
3	Royal Challengers Bangalore	7988
4	Kings XI Punjab	7858
5	Chennai Super Kings	7389
6	Rajasthan Royals	6762
7	Delhi Daredevils	6592
8	Sunrisers Hyderabad	4944
9	Deccan Chargers	3227
10	Pune Warriors	2099
11	Delhi Capitals	1324
12	Gujarat Lions	1153
13	Rising Pune Supergiant	616
14	Kochi Tuskers Kerala	595
15	Rising Pune Supergiants	521

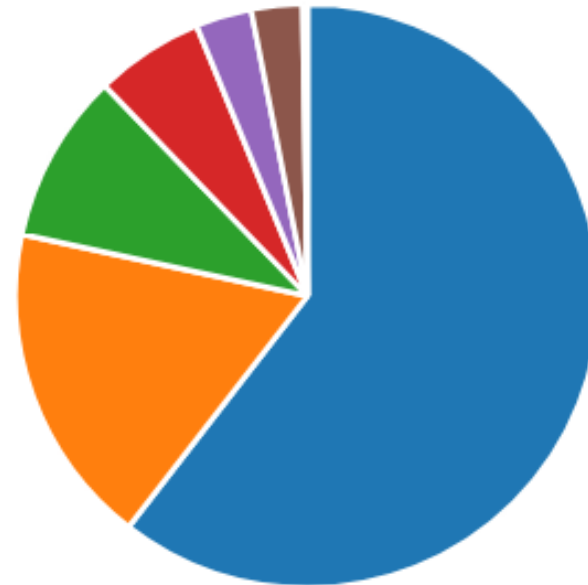


6. Total number of dismissals for dismissal kind is not equal to “NA”

- select dismissal_kind, count(*) as total_dismissals from deliveries_v03 where dismissal_kind not in('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

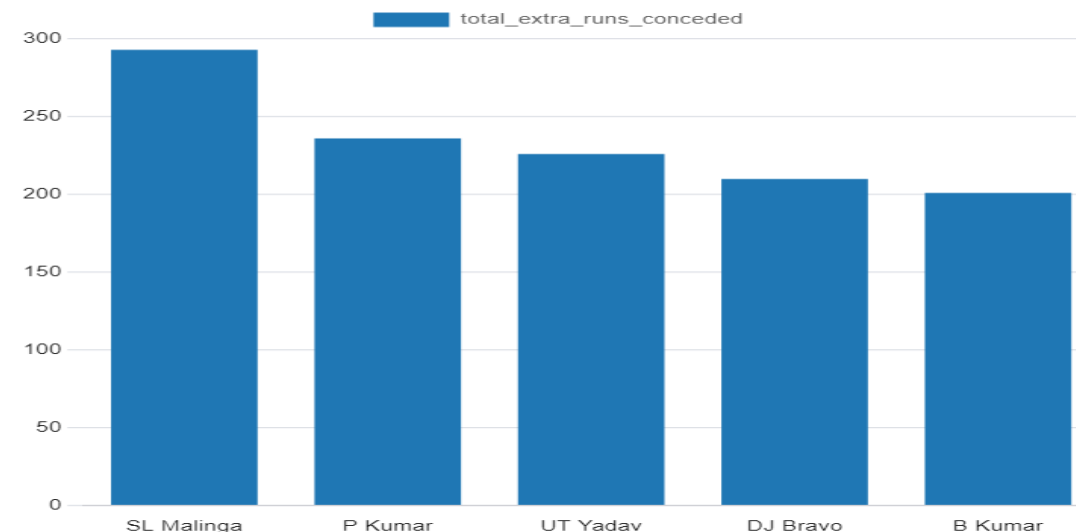
■ caught
 ■ bowled
 ■ run out
 ■ lbw
 ■ stumped
 ■ caught and bowled
 ■ hit wicket
 ■ retired hurt
 ■ obstructing the field



7. Top 5 bowlers who conceded maximum extra runs

- select bowler, sum(extra_runs) as total_extra_runs_conceded from Deliveries group by bowler order by total_extra_runs_conceded desc limit 5;

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



8. Create table deliveries_v03

- create table deliveries_v03 as (select a.*, b.venue as venue, b.date as match_date from deliveries_v02 as a inner join Matches as b on a.id=b.id);



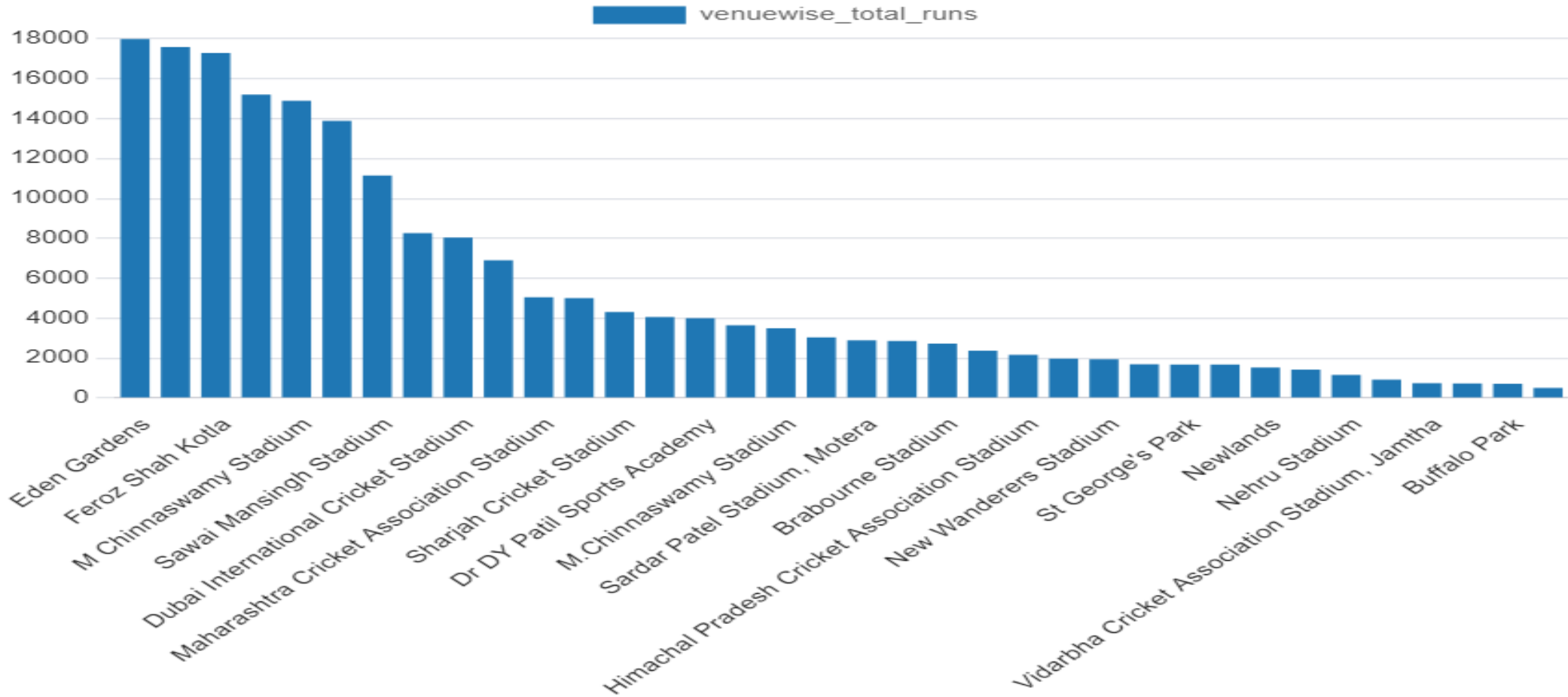
9. Total runs scored for each venue

➤ select venue, count(total_runs) as venuewise_total_runs from deliveries_v03 group by venue order by venuewise_total_runs desc;

Sl no.	venue	venuewise_total_runs
1	Eden Gardens	17988
2	Wankhede Stadium	17584
3	Feroz Shah Kotla	17294
4	Rajiv Gandhi International Stadium, Uppal	15200
5	M Chinnaswamy Stadium	14895
6	MA Chidambaram Stadium, Chepauk	13881
7	Sawai Mansingh Stadium	11150
8	Punjab Cricket Association Stadium, Mohali	8266
9	Dubai International Cricket Stadium	8038
10	Sheikh Zayed Stadium	6906
11	Maharashtra Cricket Association Stadium	5055
12	Punjab Cricket Association IS Bindra Stadium, Mohali	5003
13	Sharjah Cricket Stadium	4317
14	Subrata Roy Sahara Stadium	4064
15	Dr DY Patil Sports Academy	3993
16	Kingsmead	3643
17	M.Chinnaswamy Stadium	3494
18	Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium	3037

Sl no.	venue	venuewise_total_runs
19	Sardar Patel Stadium, Motera	2882
20	SuperSport Park	2866
21	Brabourne Stadium	2719
22	Saurashtra Cricket Association Stadium	2368
23	Himachal Pradesh Cricket Association Stadium	2159
24	Holkar Cricket Stadium	1965
25	New Wanderers Stadium	1940
26	Barabati Stadium	1695
27	St George's Park	1677
28	JSCA International Stadium Complex	1671
29	Newlands	1528
30	Shaheed Veer Narayan Singh International Stadium	1431
31	Nehru Stadium	1155
32	Green Park	921
33	Vidarbha Cricket Association Stadium, Jamtha	742
34	De Beers Diamond Oval	726
35	Buffalo Park	715
36	OUTsurance Oval	500

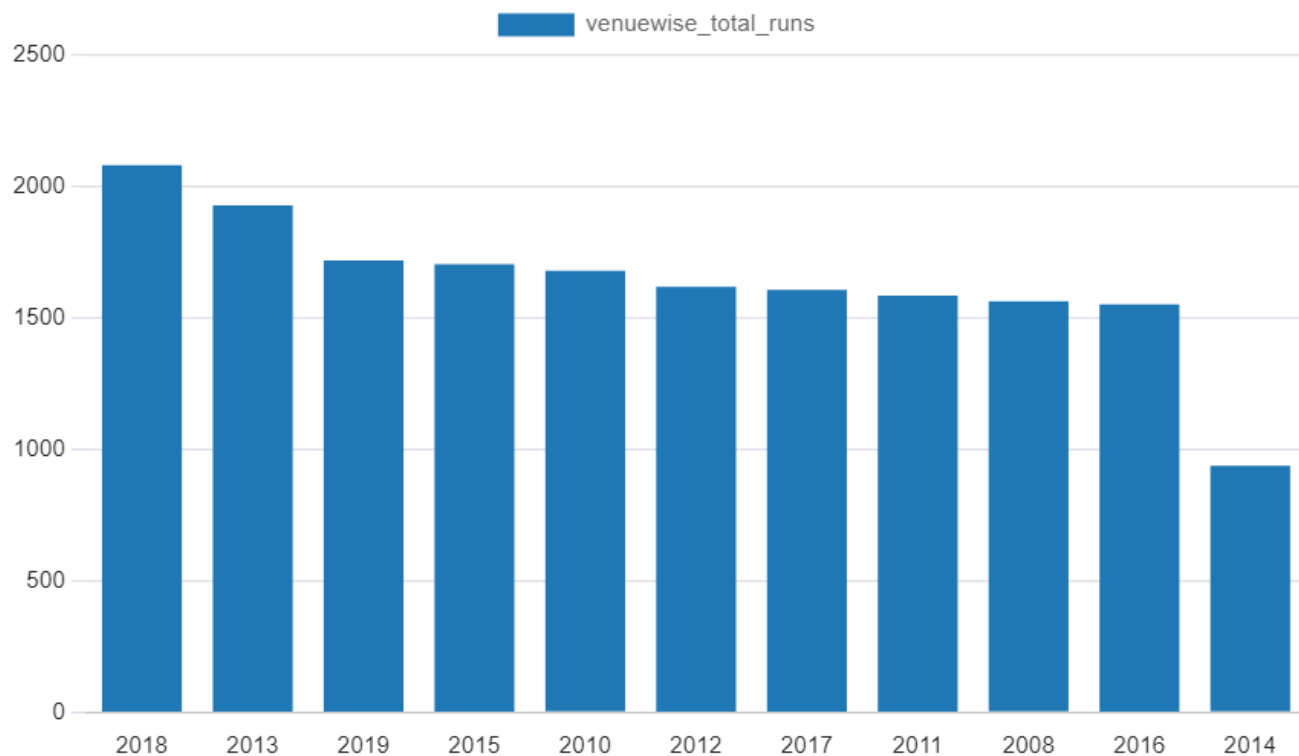
9. Total runs scored for each venue continues....



10. Total runs scored for each venue continues....

➤ select distinct substring(match_date from 7 for 4) as year, venue, count(total_runs) as venuewise_total_runs from deliveries_v03 where venue='Eden Gardens' group by year, venue order by venuewise_total_runs desc;

	year text	venue character varying	venuewise_total_runs bigint
1	2018	Eden Gardens	2082
2	2013	Eden Gardens	1929
3	2019	Eden Gardens	1720
4	2015	Eden Gardens	1706
5	2010	Eden Gardens	1681
6	2012	Eden Gardens	1620
7	2017	Eden Gardens	1608
8	2011	Eden Gardens	1586
9	2008	Eden Gardens	1565
10	2016	Eden Gardens	1553
11	2014	Eden Gardens	938





- Instead of multiplying by 1.0 to get values in decimal, I used **casting to round** values up to 4 decimals.
- In **question-2**, it's mentioned as `wicket_ball (0,1)`. I understood that it's about selecting whether a wicket is taken or not. So, if I'm not mistaken, based on that column (`is_wicket`), I wrote a query to obtain the desired outputs.
- For bidding on batters, bowlers, all-rounders, and for batting strike rate, bowling strike rate, and boundary percentage, I **multiply by 100 to get the actual form of strike rate figures**.
- **For the visualization, I used only the SQL platform because nowhere is it mentioned to use another platform. So, based on the limited options with visualization tools, I tried to extract meaningful charts/plots.**
- For bidding on `wicketkeeper`, I explained it on slide number 12.
- For **visualization**, I attempted to extract meaningful charts for all questions and kept all the details, along with codes and output, in the same slide

*Thank
you*



INTERNSHALA TRAININGS
Career Ki Guarantee