Sprint -1 IPL Analysis

Batch - V7 MS AZURE 6th Sep PT Batch

Mentor -Anindita Basak

Sahib Singh - 46264890

Charan Reddy Thathykalva - 46265397

Vivek kumar - 46263355

konduru Daiva Preethi - 46263570

Vatsal Raj - 46263571

Project Documentation

Oct 19, 2022

Capgemini

ABSTRACT

The project's main focus is on analyzing the 2 ipl data sets matches and deliveries from year 2008-2019 and answer the following questions.

Basic Aggregations:

1. Top 5 scorers of all time.
2. Top 5 expensive bowlers of all time.

Ratio:

3. Share of various dismissals in a particular season.
4. Share of tie, no result matches in a particular season.
5. Share of sixes, fours, threes, twos, singles, extra run in total runs scored during a particular season.

Trends:

6. Stats of particular player season wise.
7. Performance of teams in a particular season.

XY 2D-Matrix:

8. Bowlers who bowled super overs ,their teams and result of match.
9. which team has given how much extra runs in a particular season.

Contents

List of Figures

IPL Analysis

1 Introduction

The Indian Premier League (IPL) is a 20-20 cricket league tournament held in India contested during April and May of every year where top players from all over the world take part. The IPL is the most-attended cricket league in the world and ranks sixth among all sports leagues.

The idea is to analyse the IPL data hosted by Kaggle to get insights from the data of IPL matches and deliveries from IPL season 2008-2019.

Dataset: matches and deliveries data set

Tools used: SQL server management studio (SSMS), Power BI

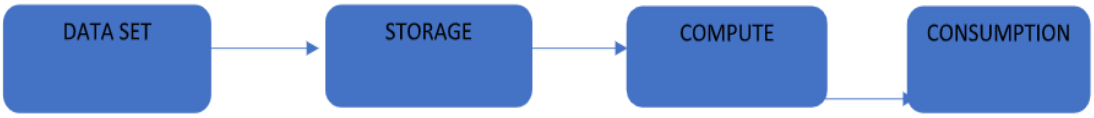


Figure 1: high level design

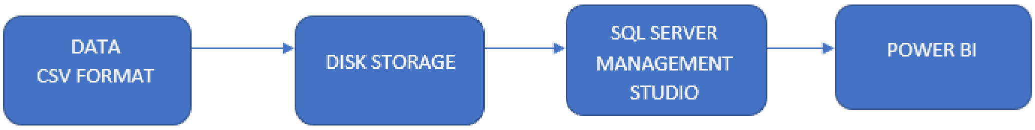


Figure 2: low level design

1.1 Implementation steps

1.1.1 Importing Data

Data is downloaded from Kaggle on to the device and then imported into SQL server management studio using 'import flat file 'option available. After giving the location of file to be imported, data can be previewed, data types can be checked and modified, we can allow null values to the columns if needed and can set primary key.

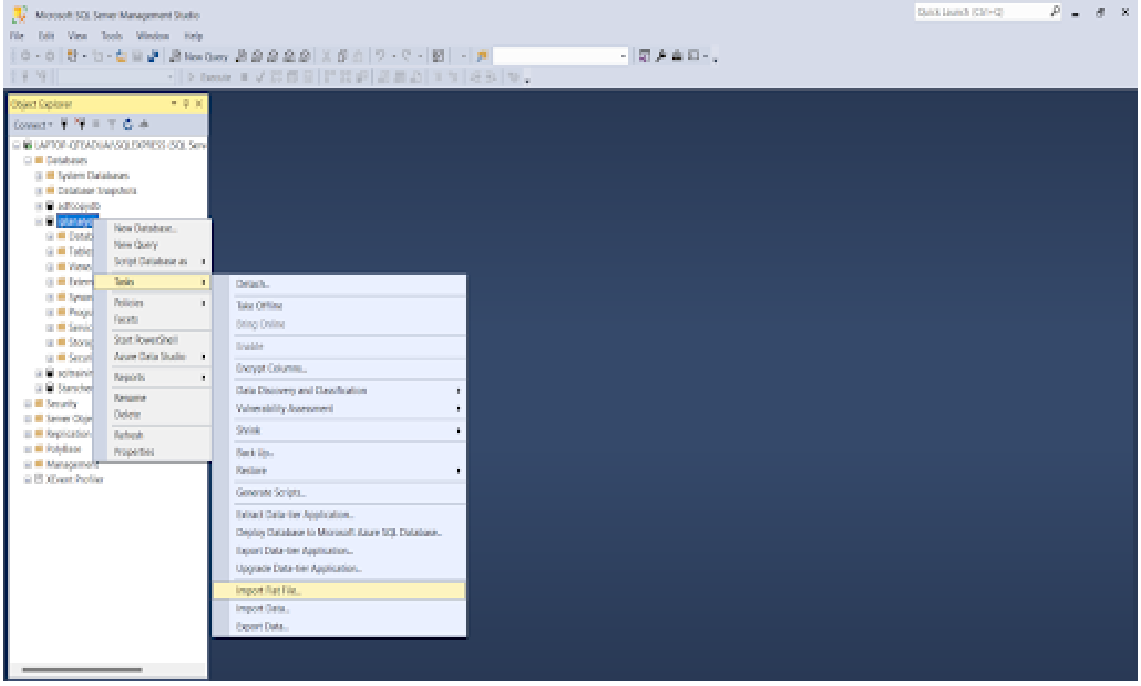


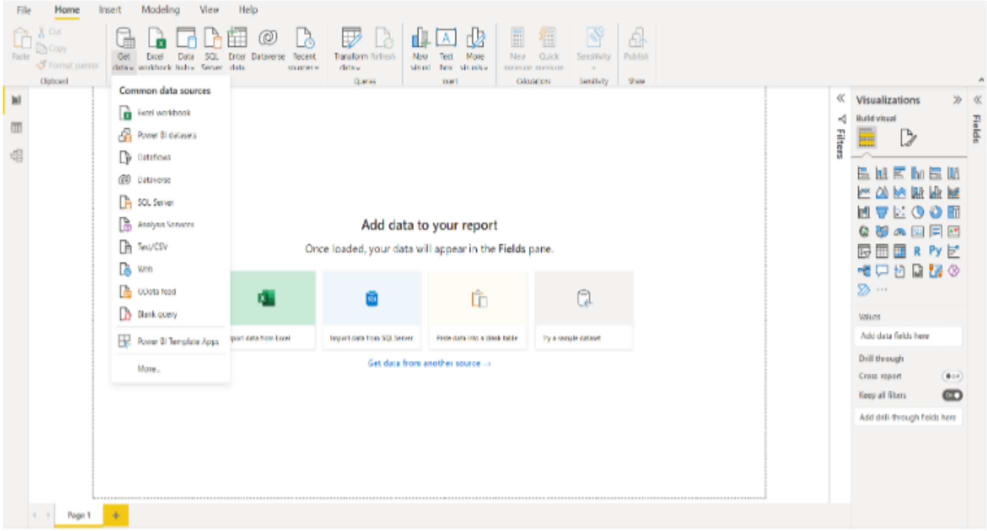
Figure 3: Importing Data to ssms

1.1.2 Analysis

Data is analysed and all necessary questions are answered by writing queries in SQL.

1.1.3 Visualization

For better understanding of results the output is visualized in power BI through some bar graphs and pie charts. For this we connected SSMS to power BI. For that we need to go to 'Get data' drop down and select 'SQL Server' as shown below.

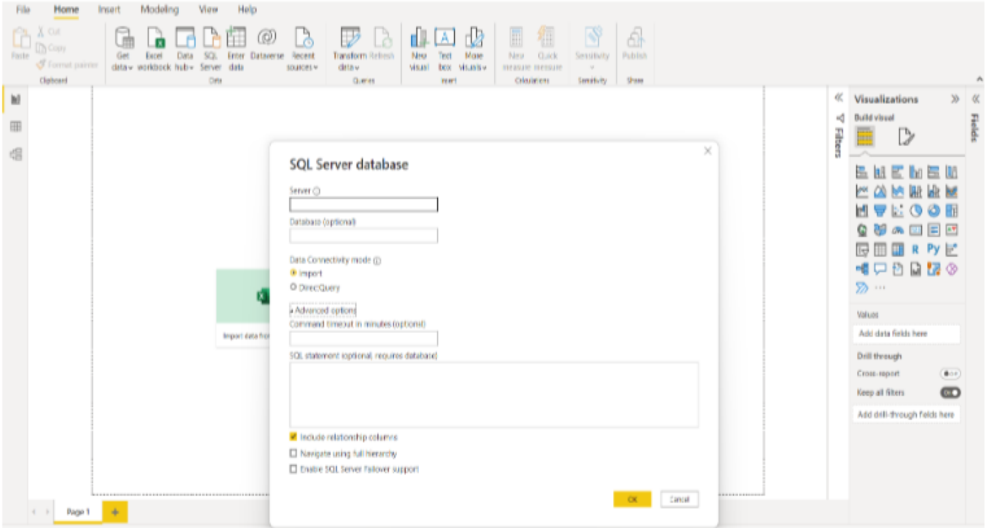


Add data to your report

Once loaded, your data will appear in the **Fields** pane.

Figure 4: Importing Data to power bi

After that we need to give details of server , data base name, and for specific query output table there is an option called‘ SQL statement’ where direct SQL query can be written.



SQL Server database

Server

Database (optional)

Data Connectivity mode
● Import
○ DirectQuery

Advanced options

Command timeout in minutes (optional)

SQL statement (optional, requires database)

☑ Include relationship columns
□ Navigate using full hierarchy
□ Enable SQL Server Failover support

Figure 5: Connecting db to power bi

1.1.4 Charts

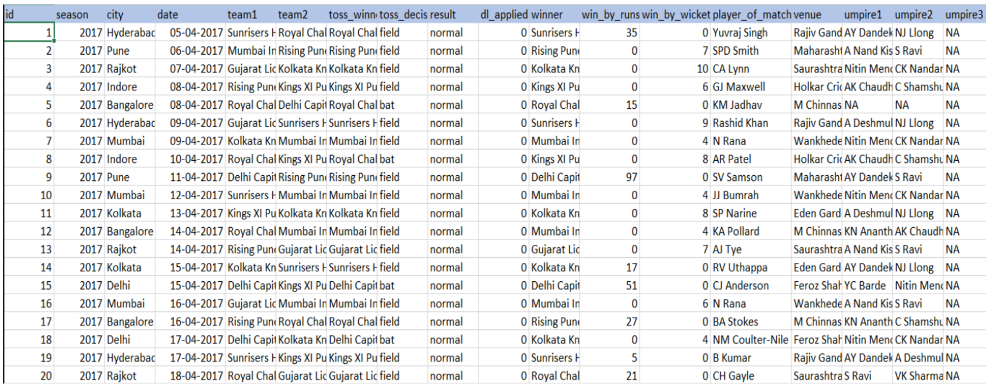
After importing necessary data into power BI, charts are build according to the requirement.

1.2 Understanding Data

1.2.1 Matches Data

The matches data set contains information of all the match ids, season, city, date of match, details of team1, team 2, the toss winner and the toss decision, result, winner of the match and by how many runs and by how many wickets the winner won the match, player of the match and details of umpires (umpire 1, umpire 2 , umpire3). All the data from matches table can be drawn using simple SQL query.

SELECT * FROM matches



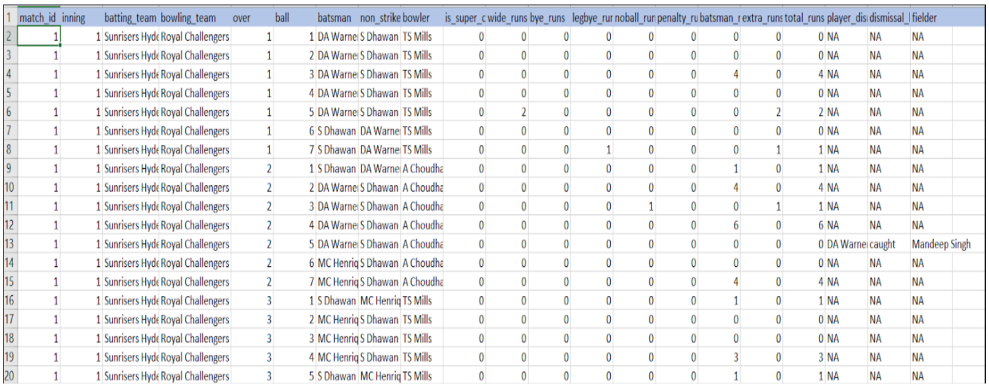
id	season	city	date	team1	team2	toss_winn	toss_decis	result	dl_applied	winner	win_by_runs	win_by_wicket	player_of_match	venue	umpire1	umpire2	umpire3
1	2017	Hyderabac	05-04-2017	Sunrisers H	Royal Chal	Royal Chal	field	normal	0	Sunrisers H	35	0	Yuvraj Singh	Rajiv Gand	AY Dandek	NJ Llong	NA
2	2017	Pune	06-04-2017	Mumbai In	Rising Pun	Rising Pun	field	normal	0	Rising Pun	0	7	SPD Smith	Maharasht	A Nand Kis	S Ravi	NA
3	2017	Rajkot	07-04-2017	Gujarat Lic	Kolkata Kn	Kolkata Kn	field	normal	0	Kolkata Kn	0	10	CA Lynn	Saurashtra	Nitin Menc	CK Nandar	NA
4	2017	Indore	08-04-2017	Rising Pun	Kings XI Pu	Kings XI Pu	field	normal	0	Kings XI Pu	0	6	GJ Maxwell	Holkar Cric	AK Chaudh	C Shamshu	NA
5	2017	Bangalore	08-04-2017	Royal Chal	Delhi Capit	Royal Chal	bat	normal	0	Royal Chal	15	0	KM Jadhav	M Chinnas	NA	NA	NA
6	2017	Hyderabac	09-04-2017	Gujarat Lic	Sunrisers H	Sunrisers H	field	normal	0	Sunrisers H	0	9	Rashid Khan	Rajiv Gand	A Deshmul	NJ Llong	NA
7	2017	Mumbai	09-04-2017	Kolkata Kn	Mumbai In	Mumbai In	field	normal	0	Mumbai In	0	4	N Rana	Wankhede	Nitin Menc	CK Nandar	NA
8	2017	Indore	10-04-2017	Royal Chal	Kings XI Pu	Royal Chal	bat	normal	0	Kings XI Pu	0	8	AR Patel	Holkar Cric	AK Chaudh	C Shamshu	NA
9	2017	Pune	11-04-2017	Delhi Capit	Rising Pun	Rising Pun	field	normal	0	Delhi Capit	97	0	SV Samson	Maharasht	AY Dandek	S Ravi	NA
10	2017	Mumbai	12-04-2017	Sunrisers H	Mumbai In	Mumbai In	field	normal	0	Mumbai In	0	4	JJ Bumrah	Wankhede	Nitin Menc	CK Nandar	NA
11	2017	Kolkata	13-04-2017	Kings XI Pu	Kolkata Kn	Kolkata Kn	field	normal	0	Kolkata Kn	0	8	SP Narine	Eden Gard	A Deshmul	NJ Llong	NA
12	2017	Bangalore	14-04-2017	Royal Chal	Mumbai In	Mumbai In	field	normal	0	Mumbai In	0	4	KA Pollard	M Chinnas	KN Ananth	AK Chaudh	NA
13	2017	Rajkot	14-04-2017	Rising Pun	Gujarat Lic	Gujarat Lic	field	normal	0	Gujarat Lic	0	7	AJ Tye	Saurashtra	A Nand Kis	S Ravi	NA
14	2017	Kolkata	15-04-2017	Kolkata Kn	Sunrisers H	Sunrisers H	field	normal	0	Kolkata Kn	17	0	RV Uthappa	Eden Gard	AY Dandek	NJ Llong	NA
15	2017	Delhi	15-04-2017	Delhi Capit	Kings XI Pu	Delhi Capit	bat	normal	0	Delhi Capit	51	0	CJ Anderson	Feroz Shah	YC Barde	Nitin Menc	NA
16	2017	Mumbai	16-04-2017	Gujarat Lic	Mumbai In	Mumbai In	field	normal	0	Mumbai In	0	6	N Rana	Wankhede	A Nand Kis	S Ravi	NA
17	2017	Bangalore	16-04-2017	Rising Pun	Royal Chal	Royal Chal	field	normal	0	Rising Pun	27	0	BA Stokes	M Chinnas	KN Ananth	C Shamshu	NA
18	2017	Delhi	17-04-2017	Delhi Capit	Kolkata Kn	Delhi Capit	bat	normal	0	Kolkata Kn	0	4	NM Coulter-Nile	Feroz Shah	Nitin Menc	CK Nandar	NA
19	2017	Hyderabac	17-04-2017	Sunrisers H	Kings XI Pu	Kings XI Pu	field	normal	0	Sunrisers H	5	0	B Kumar	Rajiv Gand	AY Dandek	A Deshmul	NA
20	2017	Rajkot	18-04-2017	Royal Chal	Gujarat Lic	Gujarat Lic	field	normal	0	Royal Chal	21	0	CH Gayle	Saurashtra	S Ravi	VK Sharma	NA

Figure 6: matches Data

1.2.2 Deliveries Data

The deliveries data set contains information of all the ball to ball data of all the matches in all the seasons from 2008-2019.. All the data from matches table can be drawn using simple SQL query.

SELECT * FROM deliveries



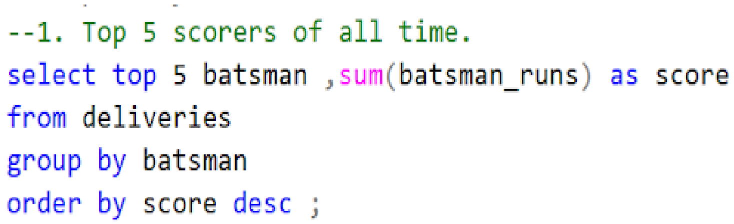
1	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_strike	bowler	is_super_c	wide_runs	bye_runs	legbye_rur	noball_rur	penalty_ru	batsman_r	extra_runs	total_runs	player_dis	dismissal_l	fielder
2	1	1	Sunrisers Hyde	Royal Challengers	1	1	DA Warne	S Dhawan	TS Mills	0	0	0	0	0	0	0	0	0	NA	NA	NA
3	1	1	Sunrisers Hyde	Royal Challengers	1	2	DA Warne	S Dhawan	TS Mills	0	0	0	0	0	0	0	0	0	NA	NA	NA
4	1	1	Sunrisers Hyde	Royal Challengers	1	3	DA Warne	S Dhawan	TS Mills	0	0	0	0	0	0	4	0	4	NA	NA	NA
5	1	1	Sunrisers Hyde	Royal Challengers	1	4	DA Warne	S Dhawan	TS Mills	0	0	0	0	0	0	0	0	0	NA	NA	NA
6	1	1	Sunrisers Hyde	Royal Challengers	1	5	DA Warne	S Dhawan	TS Mills	0	2	0	0	0	0	0	2	2	NA	NA	NA
7	1	1	Sunrisers Hyde	Royal Challengers	1	6	S Dhawan	DA Warne	TS Mills	0	0	0	0	0	0	0	0	0	NA	NA	NA
8	1	1	Sunrisers Hyde	Royal Challengers	1	7	S Dhawan	DA Warne	TS Mills	0	0	0	1	0	0	0	1	1	NA	NA	NA
9	1	1	Sunrisers Hyde	Royal Challengers	2	1	S Dhawan	DA Warne	A Choudha	0	0	0	0	0	0	1	0	1	NA	NA	NA
10	1	1	Sunrisers Hyde	Royal Challengers	2	2	DA Warne	S Dhawan	A Choudha	0	0	0	0	0	0	4	0	4	NA	NA	NA
11	1	1	Sunrisers Hyde	Royal Challengers	2	3	DA Warne	S Dhawan	A Choudha	0	0	0	0	1	0	0	1	1	NA	NA	NA
12	1	1	Sunrisers Hyde	Royal Challengers	2	4	DA Warne	S Dhawan	A Choudha	0	0	0	0	0	0	6	0	6	NA	NA	NA
13	1	1	Sunrisers Hyde	Royal Challengers	2	5	DA Warne	S Dhawan	A Choudha	0	0	0	0	0	0	0	0	0	DA Warne	caught	Mandeep Singh
14	1	1	Sunrisers Hyde	Royal Challengers	2	6	MC Henriq	S Dhawan	A Choudha	0	0	0	0	0	0	0	0	0	NA	NA	NA
15	1	1	Sunrisers Hyde	Royal Challengers	2	7	MC Henriq	S Dhawan	A Choudha	0	0	0	0	0	0	4	0	4	NA	NA	NA
16	1	1	Sunrisers Hyde	Royal Challengers	3	1	S Dhawan	MC Henriq	TS Mills	0	0	0	0	0	0	1	0	1	NA	NA	NA
17	1	1	Sunrisers Hyde	Royal Challengers	3	2	MC Henriq	S Dhawan	TS Mills	0	0	0	0	0	0	0	0	0	NA	NA	NA
18	1	1	Sunrisers Hyde	Royal Challengers	3	3	MC Henriq	S Dhawan	TS Mills	0	0	0	0	0	0	0	0	0	NA	NA	NA
19	1	1	Sunrisers Hyde	Royal Challengers	3	4	MC Henriq	S Dhawan	TS Mills	0	0	0	0	0	0	3	0	3	NA	NA	NA
20	1	1	Sunrisers Hyde	Royal Challengers	3	5	S Dhawan	MC Henriq	TS Mills	0	0	0	0	0	0	1	0	1	NA	NA	NA

Figure 7: deliveries Output

2 Basic Aggregations

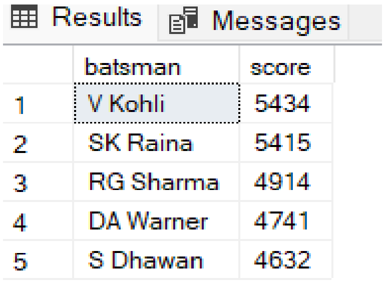
2.1 Top 5 scorers of all time

Total sum of batsman runs is used as the criteria to find out the top scorers. sum function is used to find the total batsman runs and top keyword is used to filter the first 5 results of the query.



```sql
--1. Top 5 scorers of all time.
select top 5 batsman ,sum(batsman_runs) as score
from deliveries
group by batsman
order by score desc ;
```

Figure 8: Q1 Query

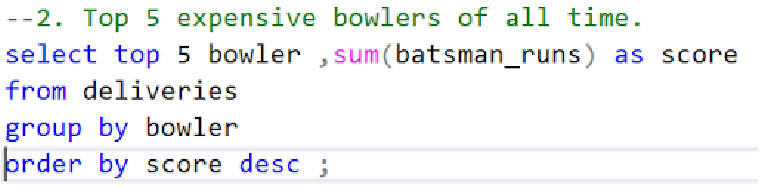


	batsman	score
1	V Kohli	5434
2	SK Raina	5415
3	RG Sharma	4914
4	DA Warner	4741
5	S Dhawan	4632

Figure 9: Q1 Output

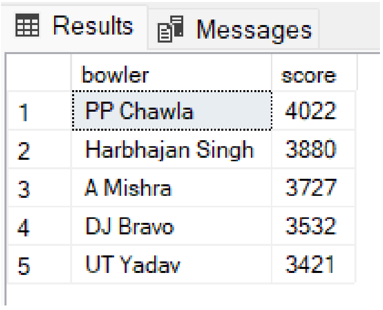
2.2 Top 5 expensive bowlers of all time.

Bowlers and their respective total batsman run is drawn out by filtering the first 5 rows to find the top 5 expensive bowlers.



```sql
--2. Top 5 expensive bowlers of all time.
select top 5 bowler ,sum(batsman_runs) as score
from deliveries
group by bowler
order by score desc ;
```

Figure 10: Q2 Query



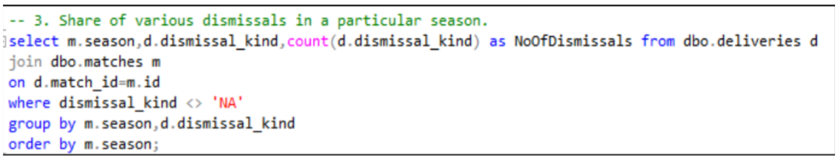
	bowler	score
1	PP Chawla	4022
2	Harbhajan Singh	3880
3	A Mishra	3727
4	DJ Bravo	3532
5	UT Yadav	3421

Figure 11: Q2 Output

3 Ratio

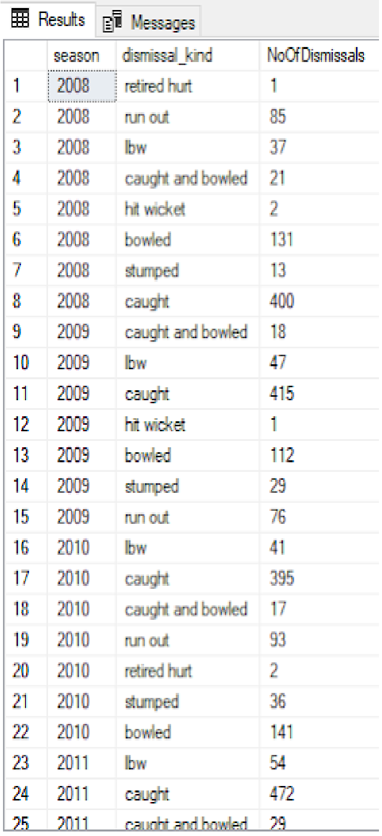
3.1 Share of various dismissals in a particular season

We performed inner join between the deliveries and matches table over match ids column and counted all the dismissal kinds and ordered them according to seasons of IPL.



```
-- 3. Share of various dismissals in a particular season.
select m.season,d.dismissal_kind,count(d.dismissal_kind) as NoOfDismissals from dbo.deliveries d
join dbo.matches m
on d.match_id=m.id
where dismissal_kind <> 'NA'
group by m.season,d.dismissal_kind
order by m.season;
```

Figure 12: Q3 Query



	season	dismissal_kind	NoOfDismissals
1	2008	retired hurt	1
2	2008	run out	85
3	2008	lbw	37
4	2008	caught and bowled	21
5	2008	hit wicket	2
6	2008	bowled	131
7	2008	stumped	13
8	2008	caught	400
9	2009	caught and bowled	18
10	2009	lbw	47
11	2009	caught	415
12	2009	hit wicket	1
13	2009	bowled	112
14	2009	stumped	29
15	2009	run out	76
16	2010	lbw	41
17	2010	caught	395
18	2010	caught and bowled	17
19	2010	run out	93
20	2010	retired hurt	2
21	2010	stumped	36
22	2010	bowled	141
23	2011	lbw	54
24	2011	caught	472
25	2011	caught and bowled	29

Figure 13: Q3 Output

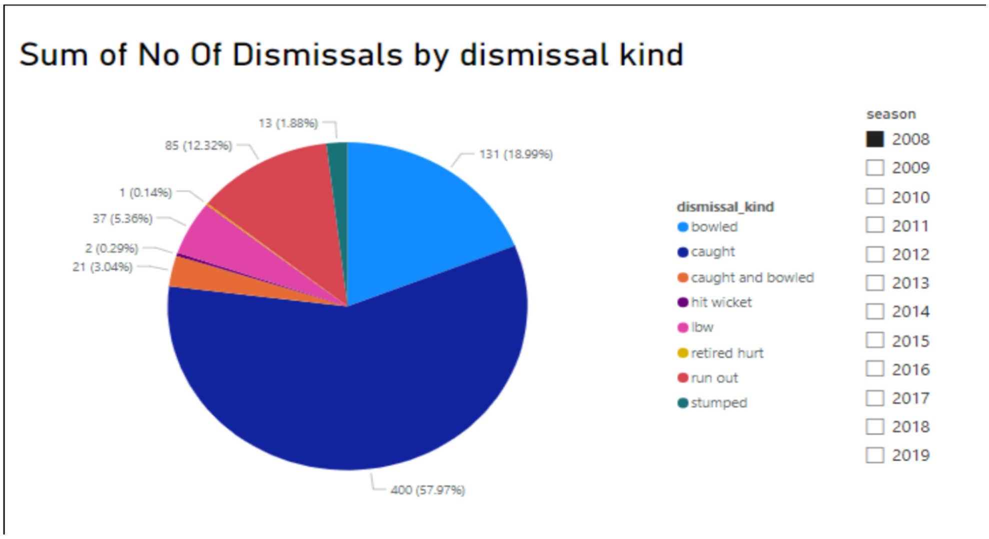


Figure 14: share of various kinds of dismissals in 2008

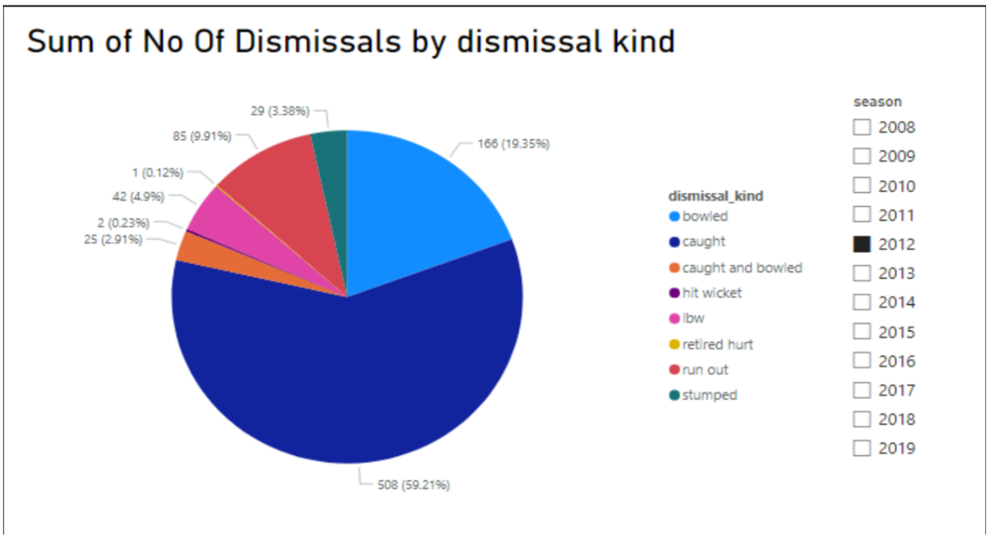
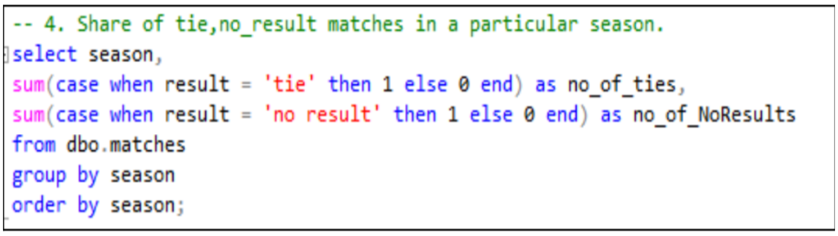


Figure 15: share of various kinds of dismissals in 2012

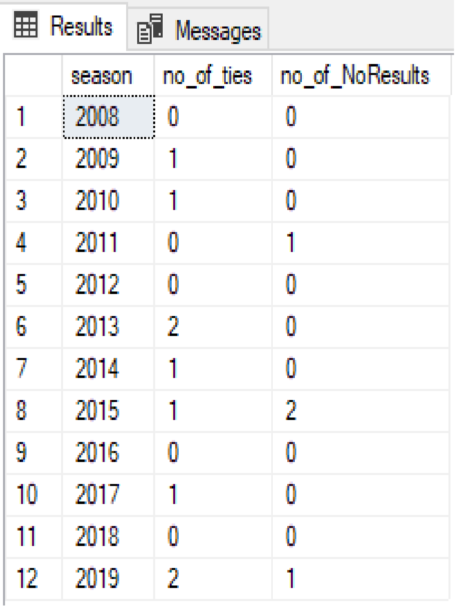
3.2 Share of tie, no result matches in a particular season

We counted the cells where result was tie and no result as two separate columns from the matches table using case functions and applying appropriate conditions, and ordered them as per different seasons of IPL.



```
-- 4. Share of tie,no_result matches in a particular season.
select season,
sum(case when result = 'tie' then 1 else 0 end) as no_of_ties,
sum(case when result = 'no result' then 1 else 0 end) as no_of_NoResults
from dbo.matches
group by season
order by season;
```

Figure 16: Q4 Query



	season	no_of_ties	no_of_NoResults
1	2008	0	0
2	2009	1	0
3	2010	1	0
4	2011	0	1
5	2012	0	0
6	2013	2	0
7	2014	1	0
8	2015	1	2
9	2016	0	0
10	2017	1	0
11	2018	0	0
12	2019	2	1

Figure 17: Q4 Output

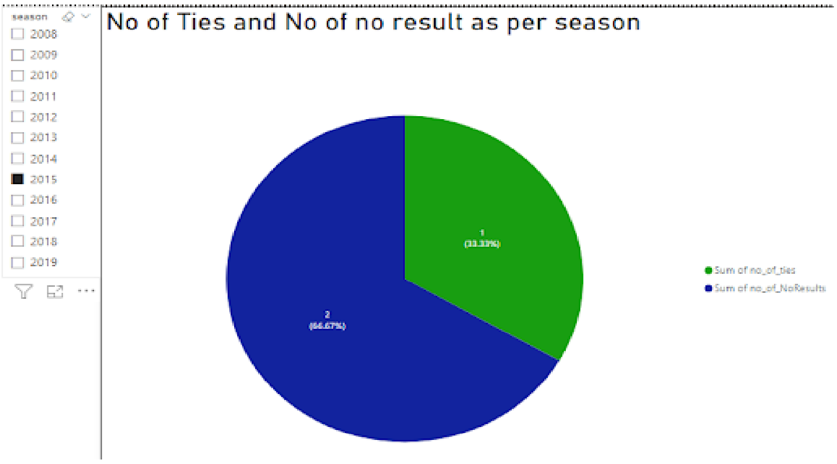


Figure 18: share of tie and no result matches in 2015

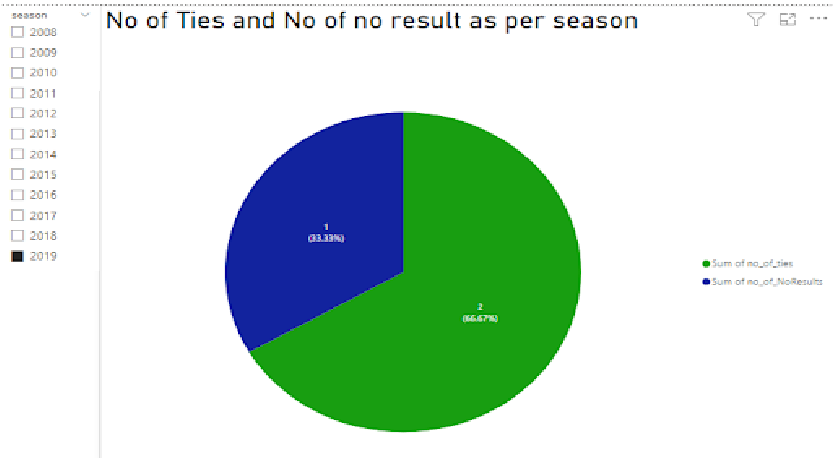
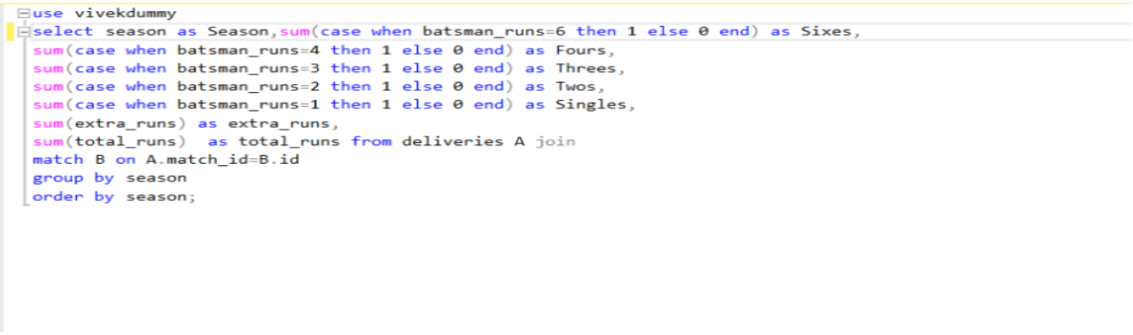


Figure 19: share of tie and no result matches in 2019

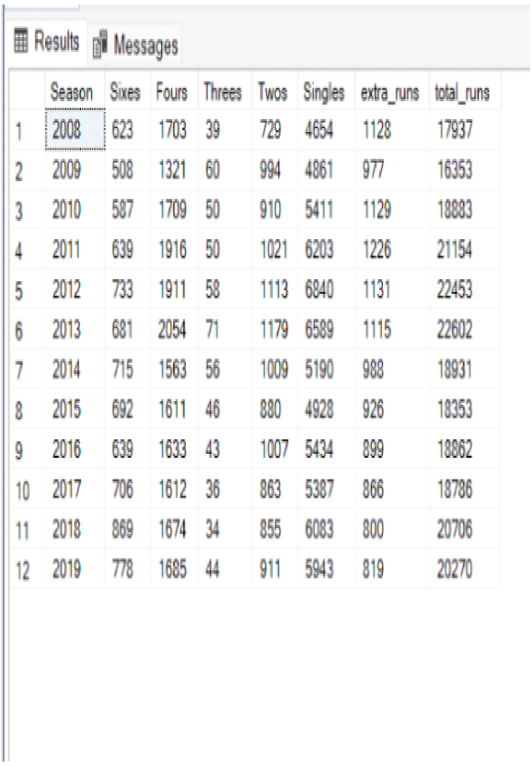
3.3 Share of Sixes,Fours,Threes,Twos,Singles,Extra Runs in total Runs during a particular season

We Performed Inner Join Operation between the deliveries table and Matches Table over the matching column of match Ids using Case functions and Applying some conditions. In this Process , We have counted the Share of Sixes, Fours,Threes ,Twos, Singles and Extra Runs in Total Runs seasonwise.



```
use vivekdummy
select season as Season,sum(case when batsman_runs=6 then 1 else 0 end) as Sixes,
sum(case when batsman_runs=4 then 1 else 0 end) as Fours,
sum(case when batsman_runs=3 then 1 else 0 end) as Threes,
sum(case when batsman_runs=2 then 1 else 0 end) as Twos,
sum(case when batsman_runs=1 then 1 else 0 end) as Singles,
sum(extra_runs) as extra_runs,
sum(total_runs)  as total_runs from deliveries A join
match B on A.match_id=B.id
group by season
order by season;
```

Figure 20: Q5 Query



	Season	Sixes	Fours	Threes	Twos	Singles	extra_runs	total_runs
1	2008	623	1703	39	729	4654	1128	17937
2	2009	508	1321	60	994	4861	977	16353
3	2010	587	1709	50	910	5411	1129	18883
4	2011	639	1916	50	1021	6203	1226	21154
5	2012	733	1911	58	1113	6840	1131	22453
6	2013	681	2054	71	1179	6589	1115	22602
7	2014	715	1563	56	1009	5190	988	18931
8	2015	692	1611	46	880	4928	926	18353
9	2016	639	1633	43	1007	5434	899	18862
10	2017	706	1612	36	863	5387	866	18786
11	2018	869	1674	34	855	6083	800	20706
12	2019	778	1685	44	911	5943	819	20270

Figure 21: Q5 Output

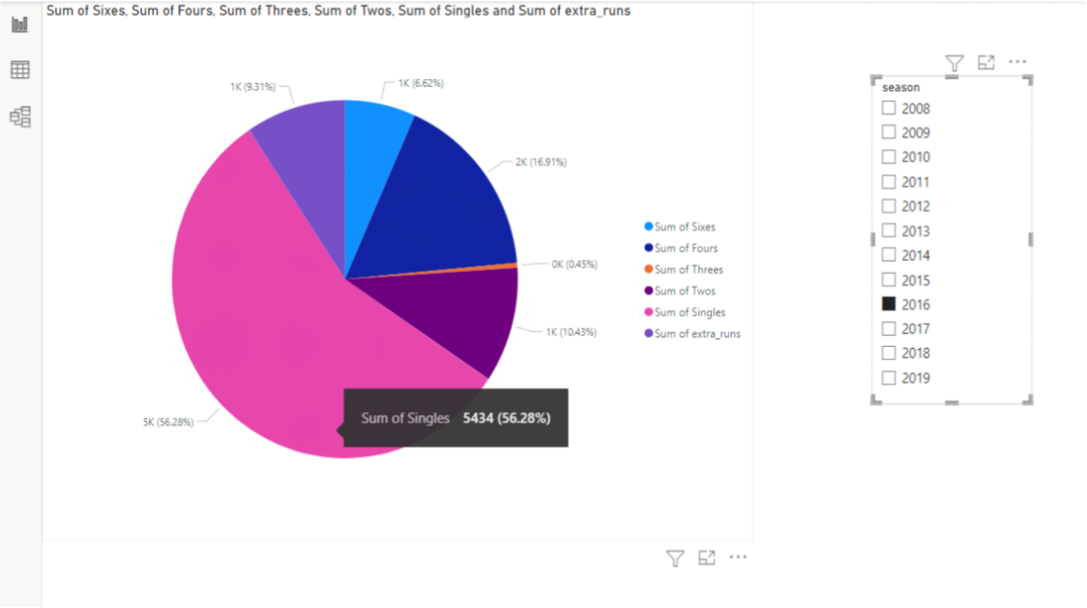


Figure 22: Share of Sixes,Fours,Threes,Twos,Singles,Extra Runs in 2016

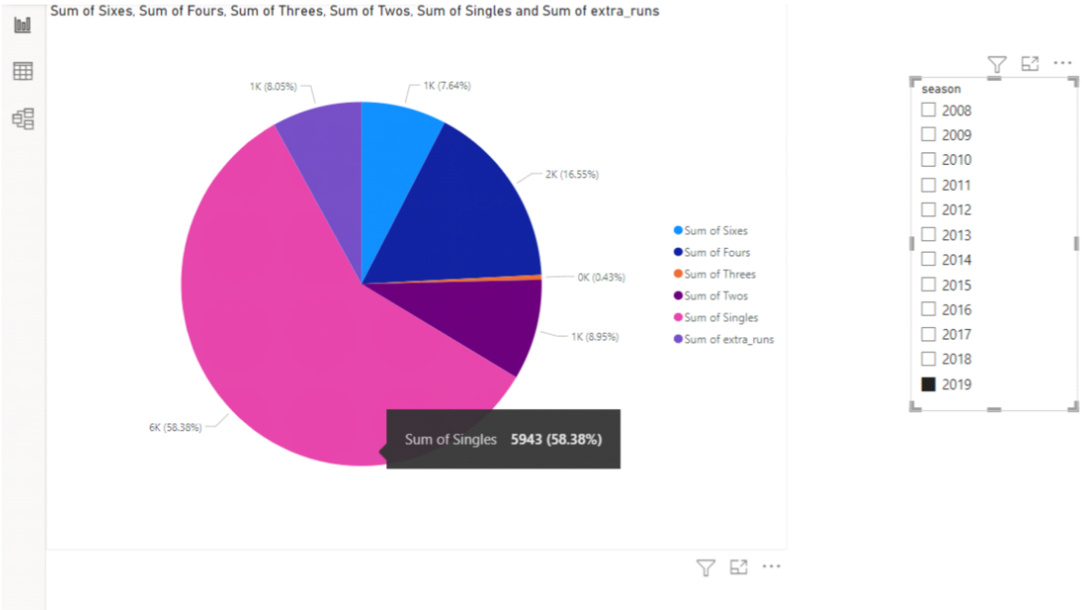
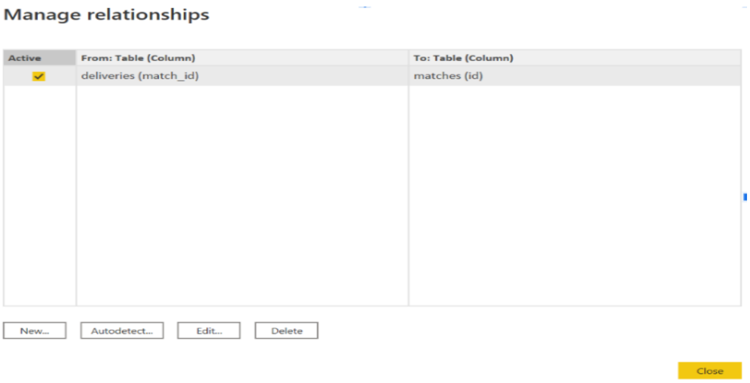


Figure 23: Share of Sixes,Fours,Threes,Twos,Singles,Extra Runs in 2019

4 Trends

4.1 Stats of particular player season wise.

Managing relations- in order to link the match ids from Deliveries and Matches data, we have used "manage relationship" functionality of power bi as shown below.

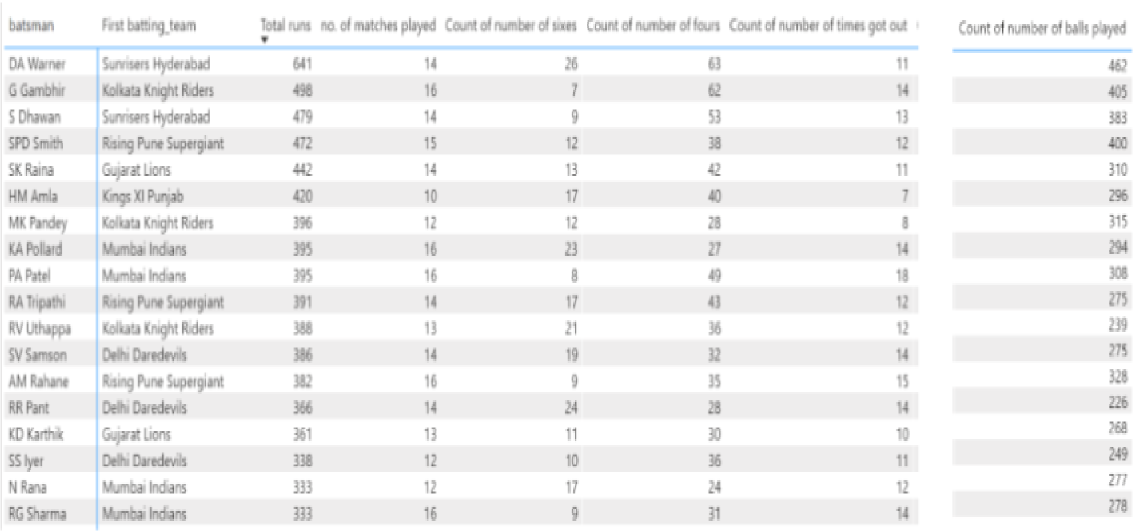


Manage relationships

Active	From: Table (Column)	To: Table (Column)
☑	deliveries (match_id)	matches (id)

Figure 24: managing relations

Once a relationship has been established, we begin to drag and drop the data as required by the question statement.

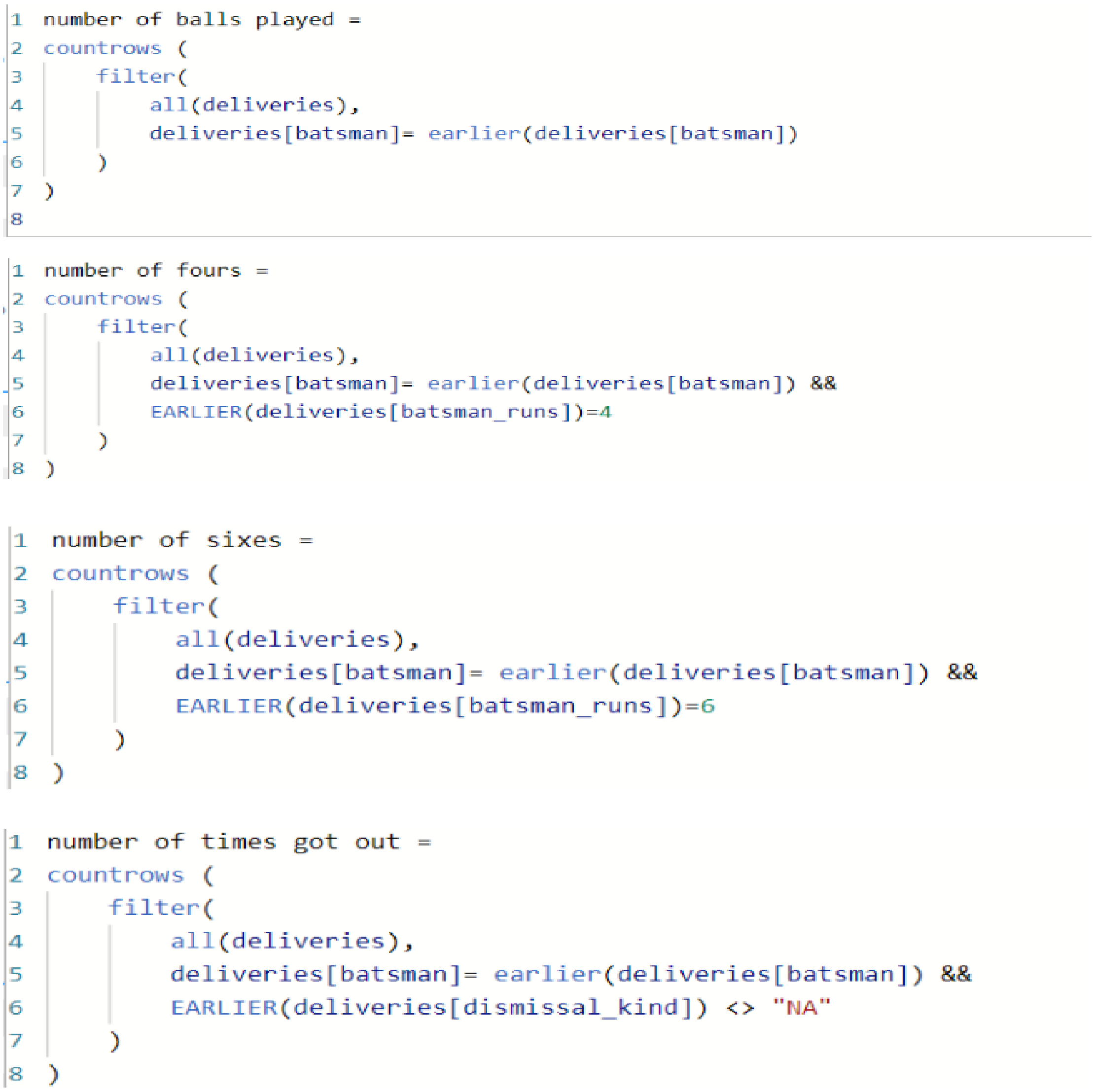


batsman	First batting_team	Total runs	no. of matches played	Count of number of sixes	Count of number of fours	Count of number of times got out	Count of number of balls played
DA Warner	Sunrisers Hyderabad	641	14	26	63	11	462
G Gambhir	Kolkata Knight Riders	498	16	7	62	14	405
S Dhawan	Sunrisers Hyderabad	479	14	9	53	13	383
SPD Smith	Rising Pune Supergiant	472	15	12	38	12	400
SK Raina	Gujarat Lions	442	14	13	42	11	310
HM Amla	Kings XI Punjab	420	10	17	40	7	296
MK Pandey	Kolkata Knight Riders	396	12	12	28	8	315
KA Pollard	Mumbai Indians	395	16	23	27	14	294
PA Patel	Mumbai Indians	395	16	8	49	18	308
RA Tripathi	Rising Pune Supergiant	391	14	17	43	12	275
RV Uthappa	Kolkata Knight Riders	388	13	21	36	12	239
SV Samson	Delhi Daredevils	386	14	19	32	14	275
AM Rahane	Rising Pune Supergiant	382	16	9	35	15	328
RR Pant	Delhi Daredevils	366	14	24	28	14	226
KD Karthik	Gujarat Lions	361	13	11	30	10	268
SS Iyer	Delhi Daredevils	338	12	10	36	11	249
N Rana	Mumbai Indians	333	12	17	24	12	277
RG Sharma	Mumbai Indians	333	16	9	31	14	278

Figure 25: stats for players in season 2017

4.1.1 DAX queries for creating different columns:

In order to obtain additional information, we wrote DAX queries to create new columns as per our requirements. The queries for number of balls played, number of sixes, number of fours, number of times a player got out has been shown below.



```
number of balls played =
countrows (
    filter(
        all(deliveries),
        deliveries[batsman]= earlier(deliveries[batsman])
    )
)

```

```
number of fours =
countrows (
    filter(
        all(deliveries),
        deliveries[batsman]= earlier(deliveries[batsman]) &&
        EARLIER(deliveries[batsman_runs])=4
    )
)
```

```
number of sixes =
countrows (
    filter(
        all(deliveries),
        deliveries[batsman]= earlier(deliveries[batsman]) &&
        EARLIER(deliveries[batsman_runs])=6
    )
)
```

```
number of times got out =
countrows (
    filter(
        all(deliveries),
        deliveries[batsman]= earlier(deliveries[batsman]) &&
        EARLIER(deliveries[dismissal_kind]) <> "NA"
    )
)
```

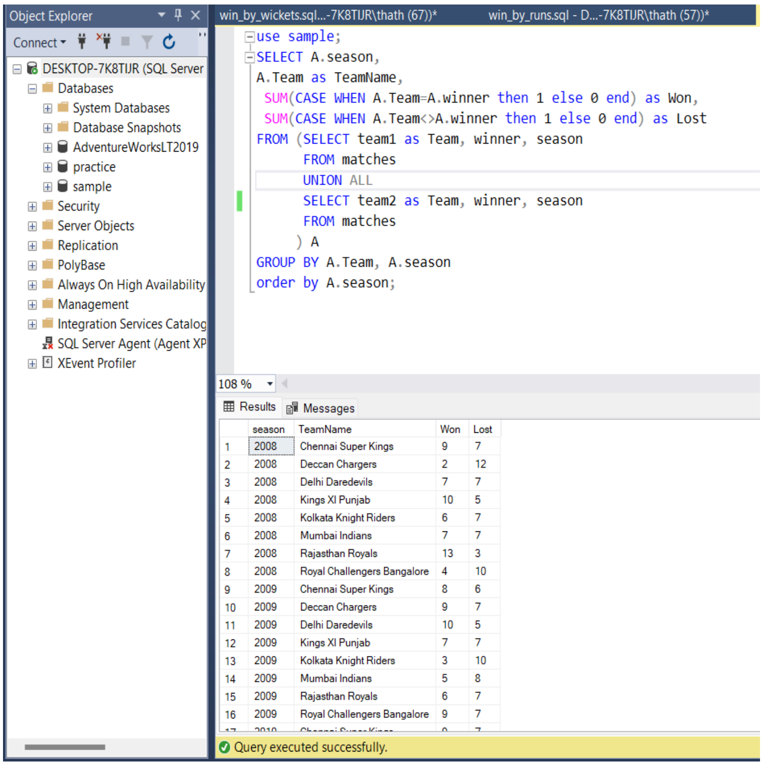
Figure 26: Dax query

4.2 Performance of teams in a particular season.

In this question we will analyze the performance of teams based on number of wins, Win by wickets , Win by runs By using visualizations we can get a better idea about each teams performance year wise

4.2.1 matches won and lost:

In this query we will try to find no.of matches won and lost. We will select 2 teams of match separately and make a larger set using union all operator and find try to find if team won or lost the match by matching team name and winner name.


Object Explorer
Connect
DESKTOP-7K8TIJR (SQL Server
Databases
System Databases
Database Snapshots
AdventureWorksLT2019
practice
sample
Security
Server Objects
Replication
PolyBase
Always On High Availability
Management
Integration Services Catalog
SQL Server Agent (Agent XP
XEvent Profiler
win_by_wickets.sql...-7K8TIJR\thath (67))*
win_by_runs.sql - D...-7K8TIJR\thath (57))*
108 %
Results
Messages

```sql
use sample;
SELECT A.season,
A.Team as TeamName,
 SUM(CASE WHEN A.Team=A.winner then 1 else 0 end) as Won,
 SUM(CASE WHEN A.Team<>A.winner then 1 else 0 end) as Lost
FROM (SELECT team1 as Team, winner, season
      FROM matches
      UNION ALL
      SELECT team2 as Team, winner, season
      FROM matches
     ) A
GROUP BY A.Team, A.season
order by A.season;
```

	season	TeamName	Won	Lost
1	2008	Chennai Super Kings	9	7
2	2008	Deccan Chargers	2	12
3	2008	Delhi Daredevils	7	7
4	2008	Kings XI Punjab	10	5
5	2008	Kolkata Knight Riders	6	7
6	2008	Mumbai Indians	7	7
7	2008	Rajasthan Royals	13	3
8	2008	Royal Challengers Bangalore	4	10
9	2009	Chennai Super Kings	8	6
10	2009	Deccan Chargers	9	7
11	2009	Delhi Daredevils	10	5
12	2009	Kings XI Punjab	7	7
13	2009	Kolkata Knight Riders	3	10
14	2009	Mumbai Indians	5	8
15	2009	Rajasthan Royals	6	7
16	2009	Royal Challengers Bangalore	9	7
17	2010	Chennai Super Kings	9	7

✓ Query executed successfully.

Figure 27: query and result

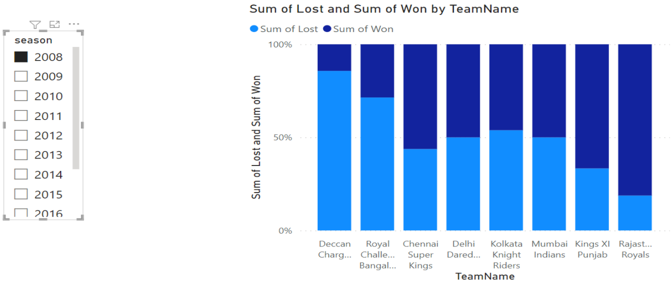


Figure 28: matches won and lost in 2008

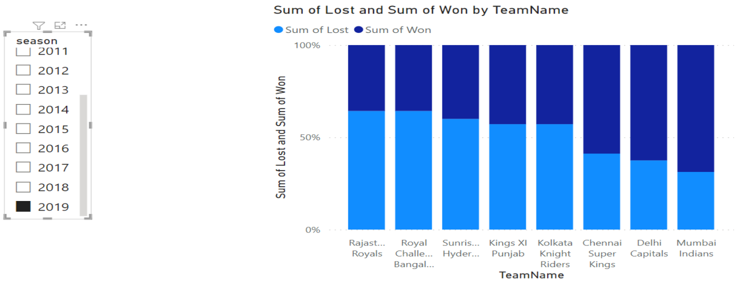
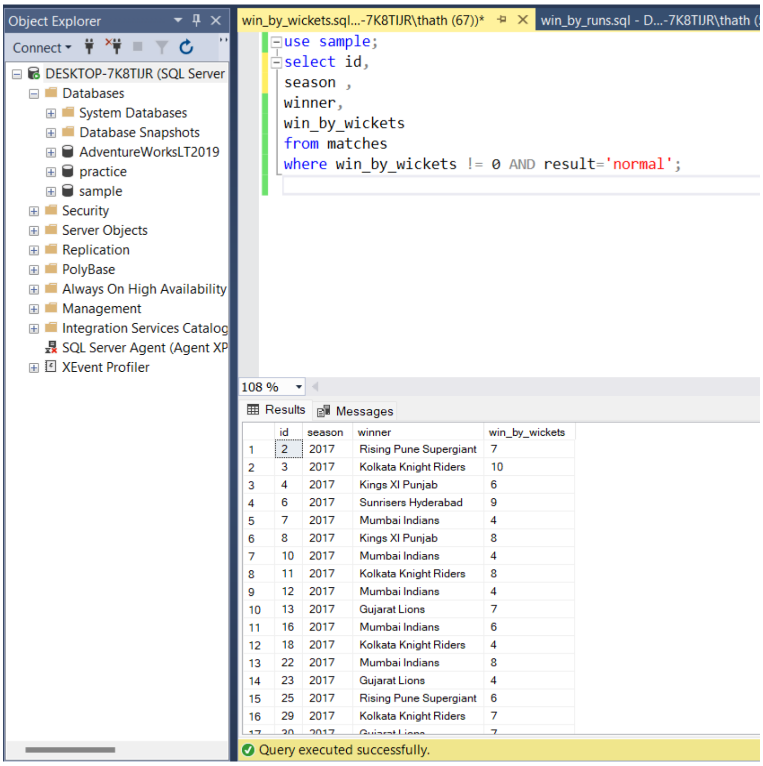


Figure 29: matches won and lost in 2019

4.2.2 win by wickets:

We will use win by wickets if a team is chasing the score set by the first playing team and won the match. To find the matches in which teams chasing the score won and the no of wickets by which they won we will only select the tuples in the data set where win by wickets is not equal to zero and the result is normal.

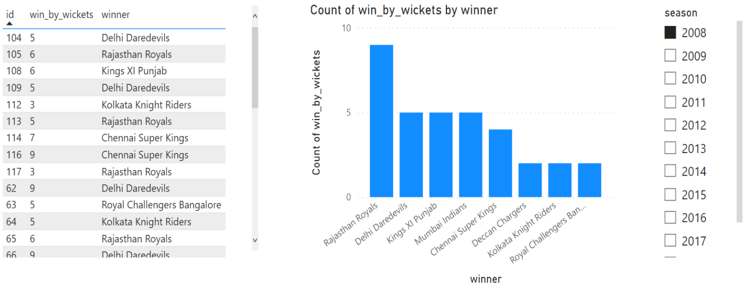

Object Explorer
Connect
DESKTOP-7K8TIJR (SQL Server
Databases
System Databases
Database Snapshots
AdventureWorksLT2019
practice
sample
Security
Server Objects
Replication
PolyBase
Always On High Availability
Management
Integration Services Catalog
SQL Server Agent (Agent XP
XEvent Profiler
win_by_wickets.sql...-7K8TIJR\thath (67))*
win_by_runs.sql - D...-7K8TIJR\thath (
108 %
Results
Messages

```
use sample;
select id,
season ,
winner,
win_by_wickets
from matches
where win_by_wickets != 0 AND result='normal';
```

	id	season	winner	win_by_wickets
1	2	2017	Rising Pune Supergiant	7
2	3	2017	Kolkata Knight Riders	10
3	4	2017	Kings XI Punjab	6
4	6	2017	Sunrisers Hyderabad	9
5	7	2017	Mumbai Indians	4
6	8	2017	Kings XI Punjab	8
7	10	2017	Mumbai Indians	4
8	11	2017	Kolkata Knight Riders	8
9	12	2017	Mumbai Indians	4
10	13	2017	Gujarat Lions	7
11	16	2017	Mumbai Indians	6
12	18	2017	Kolkata Knight Riders	4
13	22	2017	Mumbai Indians	8
14	23	2017	Gujarat Lions	4
15	25	2017	Rising Pune Supergiant	6
16	29	2017	Kolkata Knight Riders	7
17	30	2017	Gujarat Lions	7

✓ Query executed successfully.

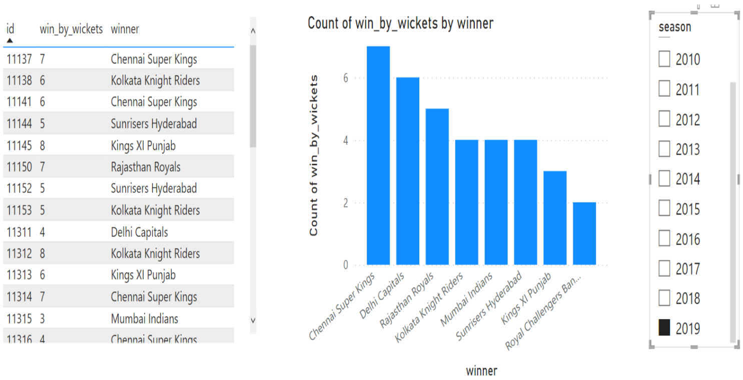
Figure 30: query and result



id	win_by_wickets	winner
104	5	Delhi Daredevils
105	6	Rajasthan Royals
108	6	Kings XI Punjab
109	5	Delhi Daredevils
112	3	Kolkata Knight Riders
113	5	Rajasthan Royals
114	7	Chennai Super Kings
116	9	Chennai Super Kings
117	3	Rajasthan Royals
62	9	Delhi Daredevils
63	5	Royal Challengers Bangalore
64	5	Kolkata Knight Riders
65	6	Rajasthan Royals
66	9	Delhi Daredevils

season
■ 2008
☐ 2009
☐ 2010
☐ 2011
☐ 2012
☐ 2013
☐ 2014
☐ 2015
☐ 2016
☐ 2017

Figure 31: Win by wickets in 2008



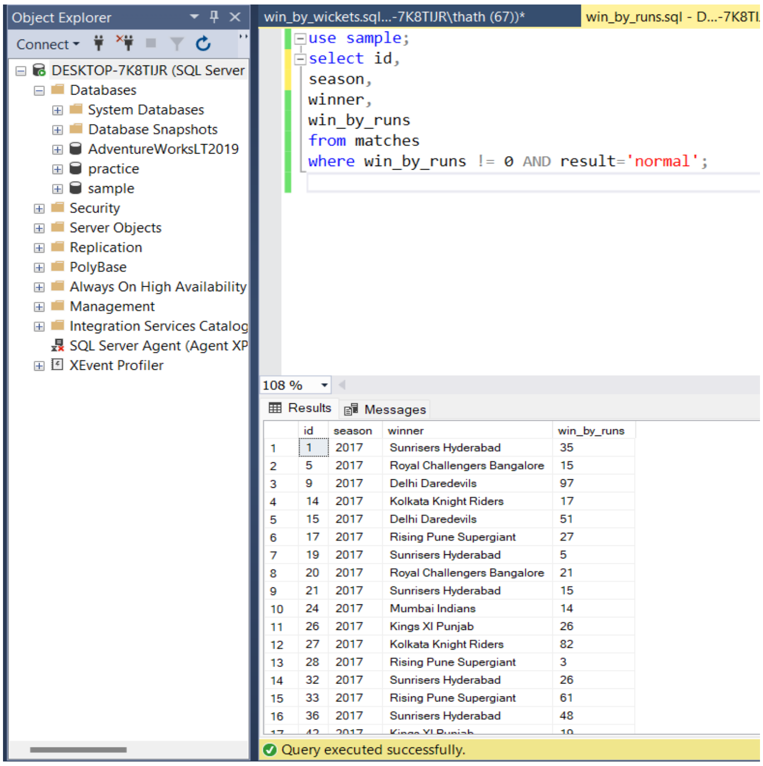
id	win_by_wickets	winner
11137	7	Chennai Super Kings
11138	6	Kolkata Knight Riders
11141	6	Chennai Super Kings
11144	5	Sunrisers Hyderabad
11145	8	Kings XI Punjab
11150	7	Rajasthan Royals
11152	5	Sunrisers Hyderabad
11153	5	Kolkata Knight Riders
11311	4	Delhi Capitals
11312	8	Kolkata Knight Riders
11313	6	Kings XI Punjab
11314	7	Chennai Super Kings
11315	3	Mumbai Indians
11316	4	Chennai Super Kings

season
☐ 2010
☐ 2011
☐ 2012
☐ 2013
☐ 2014
☐ 2015
☐ 2016
☐ 2017
☐ 2018
■ 2019

Figure 32: Win by wickets in 2019

4.2.3 win by runs:

We will use win by runs if a team is batting first and won the match by defending the score. To find the matches in which teams batted first won and the no of runs by which they won we will only select the tuples in the data set where win by runs is not equal to zero and the result is normal.

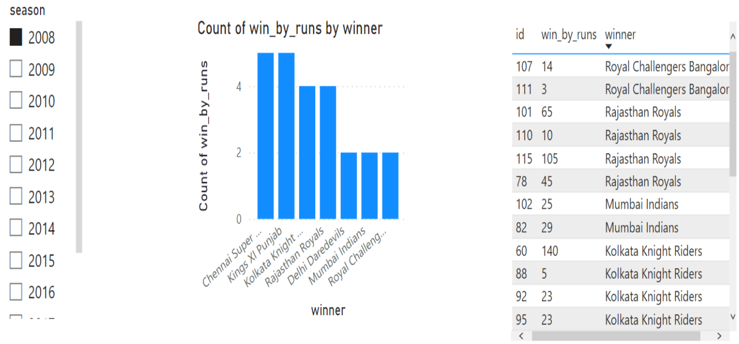


```sql
use sample;
select id,
season,
winner,
win_by_runs
from matches
where win_by_runs != 0 AND result='normal';
```

	id	season	winner	win_by_runs
1	1	2017	Sunrisers Hyderabad	35
2	5	2017	Royal Challengers Bangalore	15
3	9	2017	Delhi Daredevils	97
4	14	2017	Kolkata Knight Riders	17
5	15	2017	Delhi Daredevils	51
6	17	2017	Rising Pune Supergiant	27
7	19	2017	Sunrisers Hyderabad	5
8	20	2017	Royal Challengers Bangalore	21
9	21	2017	Sunrisers Hyderabad	15
10	24	2017	Mumbai Indians	14
11	26	2017	Kings XI Punjab	26
12	27	2017	Kolkata Knight Riders	82
13	28	2017	Rising Pune Supergiant	3
14	32	2017	Sunrisers Hyderabad	26
15	33	2017	Rising Pune Supergiant	61
16	36	2017	Sunrisers Hyderabad	48
17	42	2017	Kings XI Punjab	19

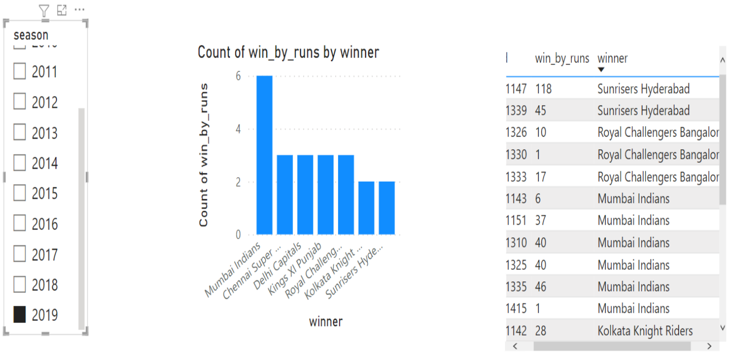
Query executed successfully.

Figure 33: query and result



id	win_by_runs	winner
107	14	Royal Challengers Bangalor
111	3	Royal Challengers Bangalor
101	65	Rajasthan Royals
110	10	Rajasthan Royals
115	105	Rajasthan Royals
78	45	Rajasthan Royals
102	25	Mumbai Indians
82	29	Mumbai Indians
60	140	Kolkata Knight Riders
88	5	Kolkata Knight Riders
92	23	Kolkata Knight Riders
95	23	Kolkata Knight Riders

Figure 34: Win by runs in 2008



season

□ 2011
□ 2012
□ 2013
□ 2014
□ 2015
□ 2016
□ 2017
□ 2018
■ 2019

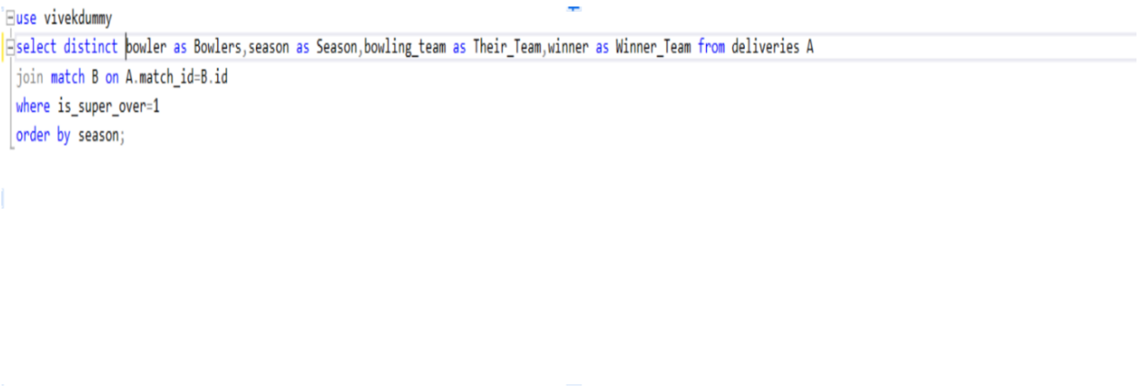
I	win_by_runs	winner
1147	118	Sunrisers Hyderabad
1339	45	Sunrisers Hyderabad
1326	10	Royal Challengers Bangalor
1330	1	Royal Challengers Bangalor
1333	17	Royal Challengers Bangalor
1143	6	Mumbai Indians
1151	37	Mumbai Indians
1310	40	Mumbai Indians
1325	40	Mumbai Indians
1335	46	Mumbai Indians
1415	1	Mumbai Indians
1142	28	Kolkata Knight Riders

Figure 35: Win by runs in 2019

5 XY 2D-Matrix:

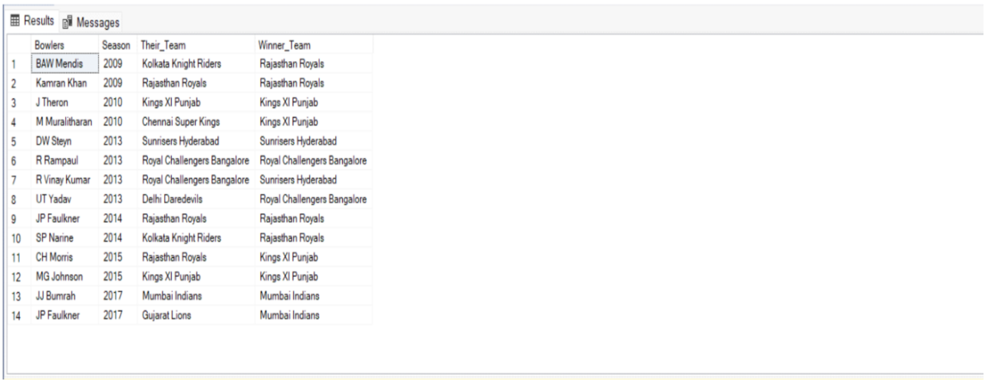
5.1 Bowlers Who Bowled Super Over ,Their Team and Result of the match

We Counted those Cells in which Super-Over was Played as Bowler ,Season in which they bowled,Their Team and the Winner Team by performing Inner Join Operation between deliveries and Matches Table and Ordered them as per different Season of IPL.



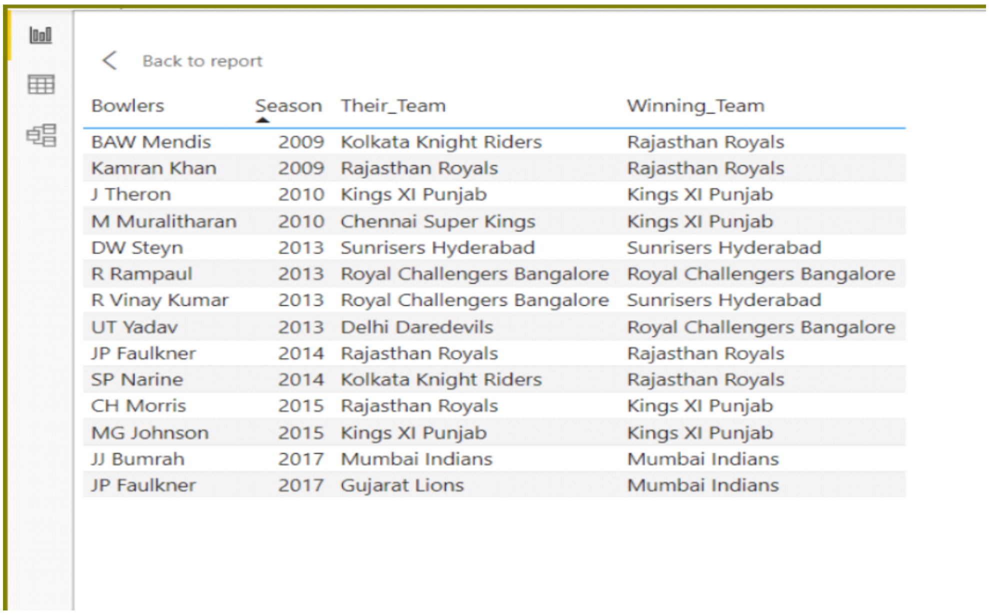
```
use vivekdummy
select distinct bowler as Bowlers,season as Season,bowling_team as Their_Team,winner as Winner_Team from deliveries A
join match B on A.match_id=B.id
where is_super_over=1
order by season;
```

Figure 36: Q8 Query



	Bowlers	Season	Their_Team	Winner_Team
1	BAW Mendis	2009	Kolkata Knight Riders	Rajasthan Royals
2	Kamran Khan	2009	Rajasthan Royals	Rajasthan Royals
3	J Theron	2010	Kings XI Punjab	Kings XI Punjab
4	M Muralitharan	2010	Chennai Super Kings	Kings XI Punjab
5	DW Steyn	2013	Sunrisers Hyderabad	Sunrisers Hyderabad
6	R Rampaul	2013	Royal Challengers Bangalore	Royal Challengers Bangalore
7	R Vinay Kumar	2013	Royal Challengers Bangalore	Sunrisers Hyderabad
8	UT Yadav	2013	Delhi Daredevils	Royal Challengers Bangalore
9	JP Faulkner	2014	Rajasthan Royals	Rajasthan Royals
10	SP Narine	2014	Kolkata Knight Riders	Rajasthan Royals
11	CH Morris	2015	Rajasthan Royals	Kings XI Punjab
12	MG Johnson	2015	Kings XI Punjab	Kings XI Punjab
13	JJ Bumrah	2017	Mumbai Indians	Mumbai Indians
14	JP Faulkner	2017	Gujarat Lions	Mumbai Indians

Figure 37: Q8 Output

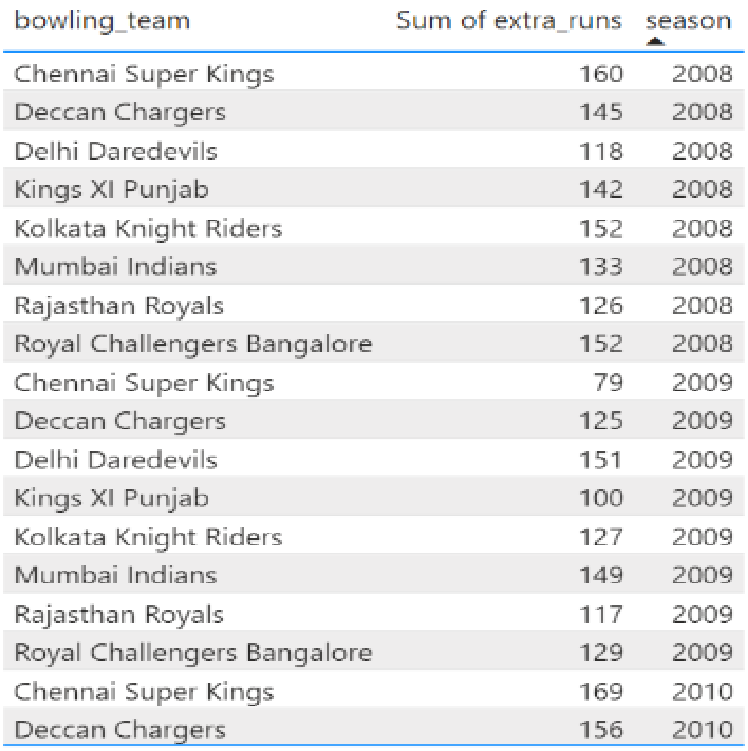


Bowlers	Season	Their_Team	Winning_Team
BAW Mendis	2009	Kolkata Knight Riders	Rajasthan Royals
Kamran Khan	2009	Rajasthan Royals	Rajasthan Royals
J Theron	2010	Kings XI Punjab	Kings XI Punjab
M Muralitharan	2010	Chennai Super Kings	Kings XI Punjab
DW Steyn	2013	Sunrisers Hyderabad	Sunrisers Hyderabad
R Rampaul	2013	Royal Challengers Bangalore	Royal Challengers Bangalore
R Vinay Kumar	2013	Royal Challengers Bangalore	Sunrisers Hyderabad
UT Yadav	2013	Delhi Daredevils	Royal Challengers Bangalore
JP Faulkner	2014	Rajasthan Royals	Rajasthan Royals
SP Narine	2014	Kolkata Knight Riders	Rajasthan Royals
CH Morris	2015	Rajasthan Royals	Kings XI Punjab
MG Johnson	2015	Kings XI Punjab	Kings XI Punjab
JJ Bumrah	2017	Mumbai Indians	Mumbai Indians
JP Faulkner	2017	Gujarat Lions	Mumbai Indians

Figure 38: power bi super over output

5.2 Which team has given how much extra runs in a particular season?

In order to obtain the required data, we simply drag and drop the required fields from the field menu of power Bi from Matches and Deliveries tables.



bowling_team	Sum of extra_runs	season
Chennai Super Kings	160	2008
Deccan Chargers	145	2008
Delhi Daredevils	118	2008
Kings XI Punjab	142	2008
Kolkata Knight Riders	152	2008
Mumbai Indians	133	2008
Rajasthan Royals	126	2008
Royal Challengers Bangalore	152	2008
Chennai Super Kings	79	2009
Deccan Chargers	125	2009
Delhi Daredevils	151	2009
Kings XI Punjab	100	2009
Kolkata Knight Riders	127	2009
Mumbai Indians	149	2009
Rajasthan Royals	117	2009
Royal Challengers Bangalore	129	2009
Chennai Super Kings	169	2010
Deccan Chargers	156	2010

Figure 39: season wise analysis for sum of extra runs for different teams

6 Conclusion

We have successfully completed the project and answered all the goal statements. We have also shown appropriate visualizations using Power BI in the necessary questions.