PROJECT

IE 6700 Data Management for Analytics

TITLE

Cricket Auction Player Performance Database Management System

GROUP NUMBER - 6

Yash Bhadreshwara
Sparsh Marwah

PROBLEM STATEMENT

In the world of professional cricket, team owners attend auctions to pick skilled players who can help their teams win. Choosing the right players during these auctions is super important because it directly affects how well the team does in games and competitions. To make smart choices, team owners and managers need to look at how well a player has done in the past, what role they play, and how they can help the team win. The main issue we are tackling is creating a strong system that uses data to help team owners pick the best players during these cricket auctions. We want this system to help team owners and managers make smart decisions that will make their cricket teams better and more competitive.

ENTITIES & THEIR FUNCTIONALITIES

1. **Player** - It stores information about players such as their ID, Name, Stance, Age, Batting Order, Bowler Type, Wicketkeeper and their roles as a subclass i.e Batsmen, Bowler, All-Rounder.

2. **Team** - It gives us information about the team which is Name, Owner, Home City & Home Ground.

3. **Match** - It tells us about the match details such as Venue, Match No, Date, Result, Winning and Losing Team.

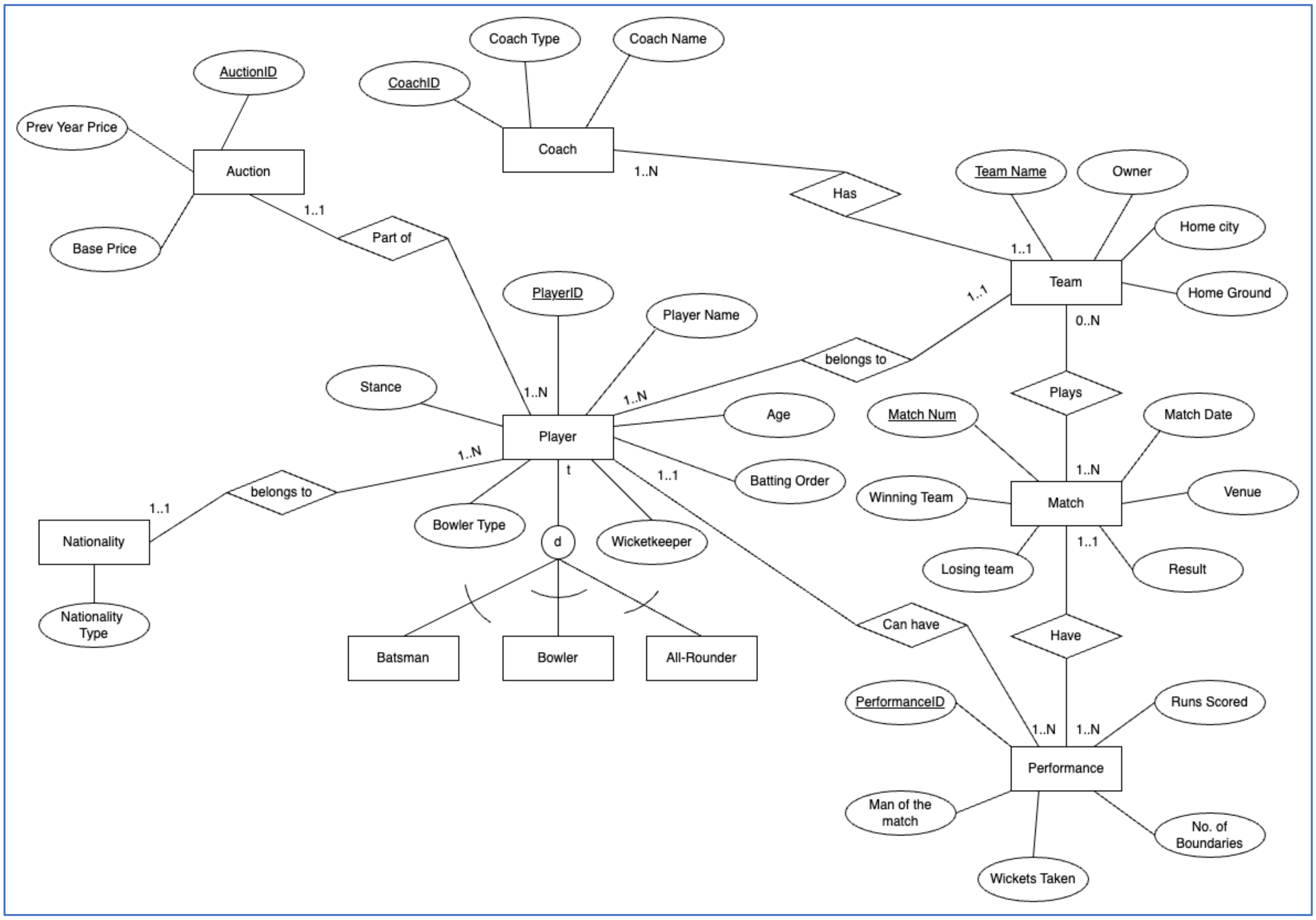
4. **Performance** - It shows us the performance details about the player such as Runs Scored, Wickets Taken, No of boundaries and Man of the Match.

5. **Nationality** - It tells us about the nationality type of the player (National or Overseas).

6. **Auction** - It stores information about the auction details of the player which are Auction ID, Previous Year Price and Base Price.

7. **Coach** - It gives us information about the type of coach, Coach Name and name of the team whom they are coaching to.

EER DIAGRAM



REFERENCE DATA

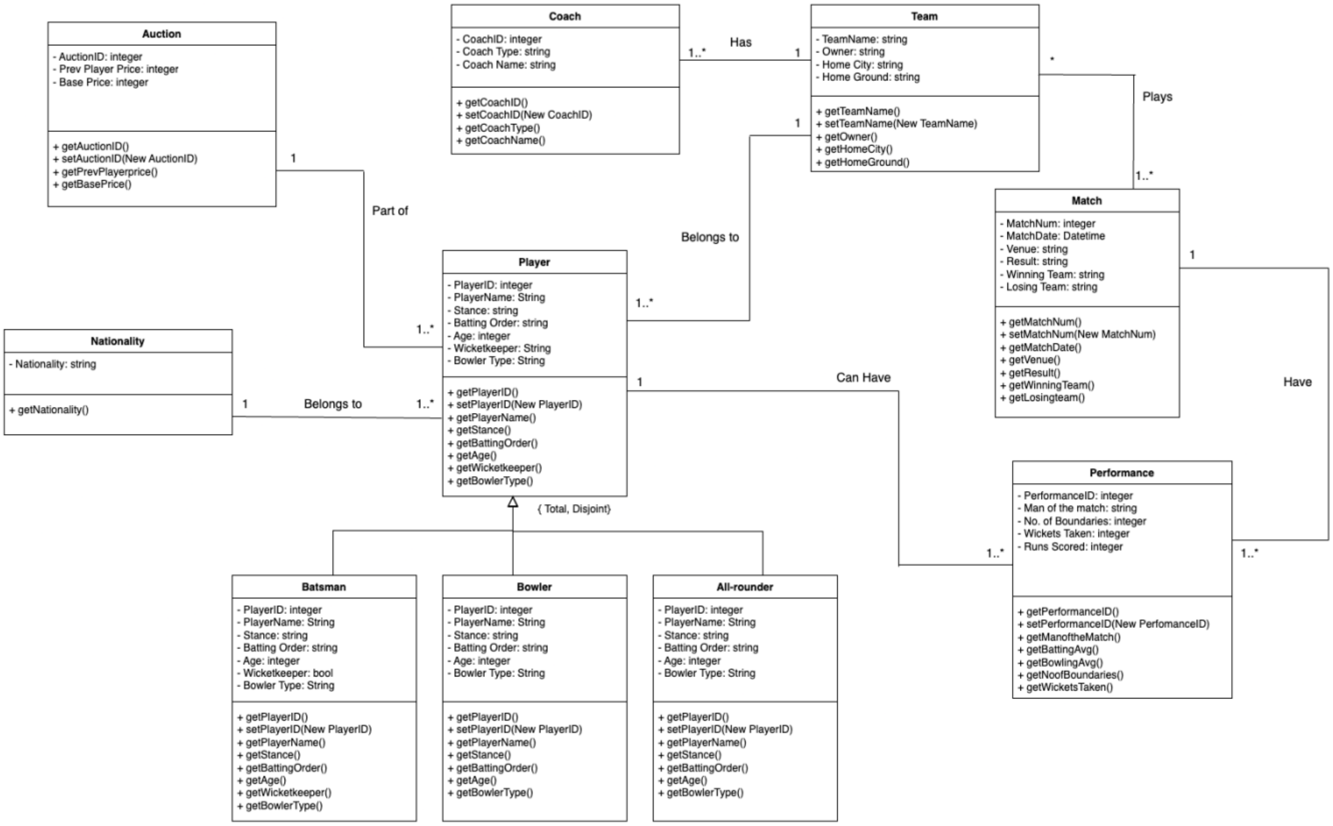
Primary Keys: Player ID, Team Name, Match ID, Performance ID, Auction ID, Coach ID.

Foreign Keys: Player ID, Winning Team, Losing Team, Match ID.

TRANSACTIONAL DATA

Previous Year Price, Base Price

UML


Auction
- AuctionID: integer
- Prev Player Price: integer
- Base Price: integer
+ getAuctionID()
+ setAuctionID(New AuctionID)
+ getPrevPlayerprice()
+ getBasePrice()
Coach
- CoachID: integer
- Coach Type: string
- Coach Name: string
+ getCoachID()
+ setCoachID(New CoachID)
+ getCoachType()
+ getCoachName()
1..*
Has
1
Team
- TeamName: string
- Owner: string
- Home City: string
- Home Ground: string
+ getTeamName()
+ setTeamName(New TeamName)
+ getOwner()
+ getHomeCity()
+ getHomeGround()
*
Plays
1..*
1
1
Part of
Belongs to
Match
- MatchNum: integer
- MatchDate: Datetime
- Venue: string
- Result: string
- Winning Team: string
- Losing Team: string
+ getMatchNum()
+ setMatchNum(New MatchNum)
+ getMatchDate()
+ getVenue()
+ getResult()
+ getWinningTeam()
+ getLosingteam()
1
Player
- PlayerID: integer
- PlayerName: String
- Stance: string
- Batting Order: string
- Age: integer
- Wicketkeeper: String
- Bowler Type: String
+ getPlayerID()
+ setPlayerID(New PlayerID)
+ getPlayerName()
+ getStance()
+ getBattingOrder()
+ getAge()
+ getWicketkeeper()
+ getBowlerType()
1..*
1..*
Nationality
- Nationality: string
+ getNationality()
1
Belongs to
1..*
1
Can Have
Have
{ Total, Disjoint}
Performance
- PerformanceID: integer
- Man of the match: string
- No. of Boundaries: integer
- Wickets Taken: integer
- Runs Scored: integer
+ getPerformanceID()
+ setPerformanceID(New PerfomanceID)
+ getManoftheMatch()
+ getBattingAvg()
+ getBowlingAvg()
+ getNoofBoundaries()
+ getWicketsTaken()
1..*
1..*
Batsman
- PlayerID: integer
- PlayerName: String
- Stance: string
- Batting Order: string
- Age: integer
- Wicketkeeper: bool
- Bowler Type: String
+ getPlayerID()
+ setPlayerID(New PlayerID)
+ getPlayerName()
+ getStance()
+ getBattingOrder()
+ getAge()
+ getWicketkeeper()
+ getBowlerType()
Bowler
- PlayerID: integer
- PlayerName: String
- Stance: string
- Batting Order: string
- Age: integer
- Bowler Type: String
+ getPlayerID()
+ setPlayerID(New PlayerID)
+ getPlayerName()
+ getStance()
+ getBattingOrder()
+ getAge()
+ getBowlerType()
All-rounder
- PlayerID: integer
- PlayerName: String
- Stance: string
- Batting Order: string
- Age: integer
- Bowler Type: String
+ getPlayerID()
+ setPlayerID(New PlayerID)
+ getPlayerName()
+ getStance()
+ getBattingOrder()
+ getAge()
+ getBowlerType()

RELATIONAL MODEL

Player (Player ID, Player Name, Age, Stance, Batting Order, Wicketkeeper, Bowler Type*, Auction ID, Team Name*)
Primary Key – Player ID
Foreign Key – Auction ID refers to primary key of Auction and is NOT NULL
Team Name refers to primary key of team and is NOT NULL

Batsmen (Player ID, Player Name, Age, Stance, Batting Order, Wicketkeeper, Bowler Type)
Primary Key – Player ID

Bowler (Player ID, Player Name, Age, Stance, Batting Order, Bowler Type)
Primary Key – Player ID

All Rounder (Player ID, Player Name, Age, Stance, Batting Order, Bowler Type)
Primary Key – Player ID

Nationality (Player ID, Nationality Type)

Auction (Auction ID, Prev Year Price, Base Price, *Player ID*)
Primary Key – Auction ID
Foreign Key – Player ID refers to primary key of Player and is NOT NULL

Coach (Coach ID, Coach Name, Coach Type, *Team Name*)
Primary Key – Coach ID
Foreign Key – Team Name refers to primary key of Team

Team (Team Name, Owner, Home City, Home Ground)
Primary Key – Team Name

Match (Match Num, Match Date, Winning Team, Losing Team, Venue, Result)
Primary Key – Match Num

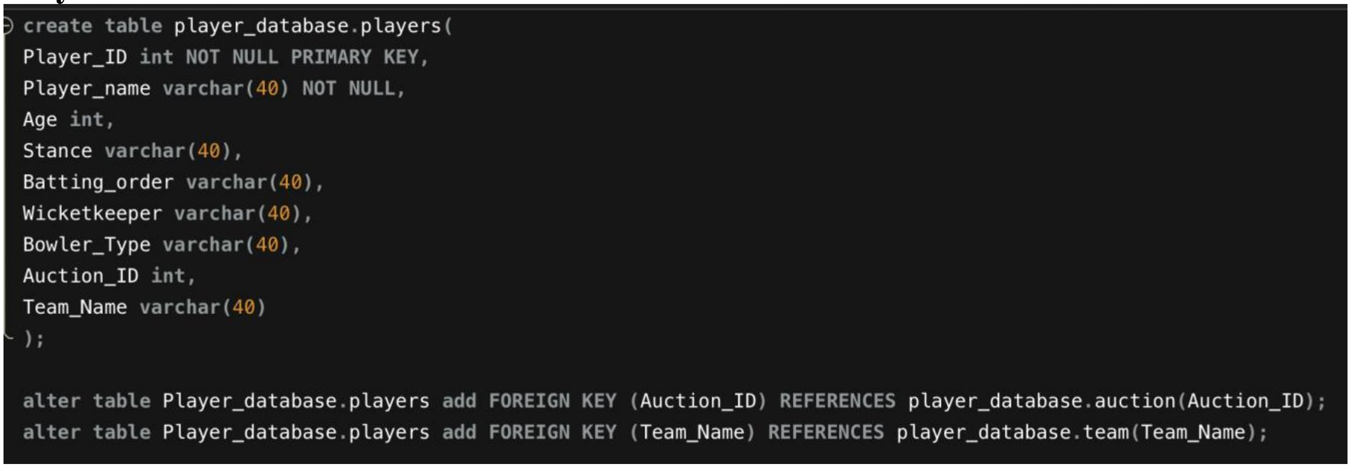
Plays (*Team Name, Match Num*)
Primary Key – Team Name and Match Num together make the primary key for plays
Foreign Key – Team Name refers to primary key of Team and NULL is allowed
Match Num refers to primary key of Match and is NOT NULL

Performance (Performance ID, No. of Boundaries, Runs Scored, Man of the match, Wickets Taken, *Player ID*, *Match Num*)
Primary Key – Performance ID
Foreign Key – Player ID refers to primary key of Player and is NOT NULL
Match Num refers to primary key of Match and is NOT NULL

SQL QUERIES

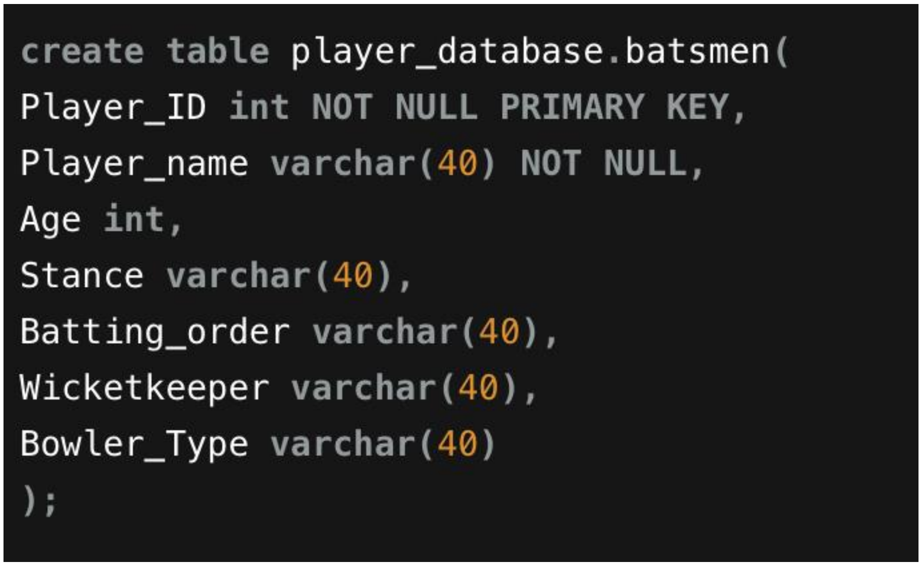
CREATE STATEMENTS

Players Table:



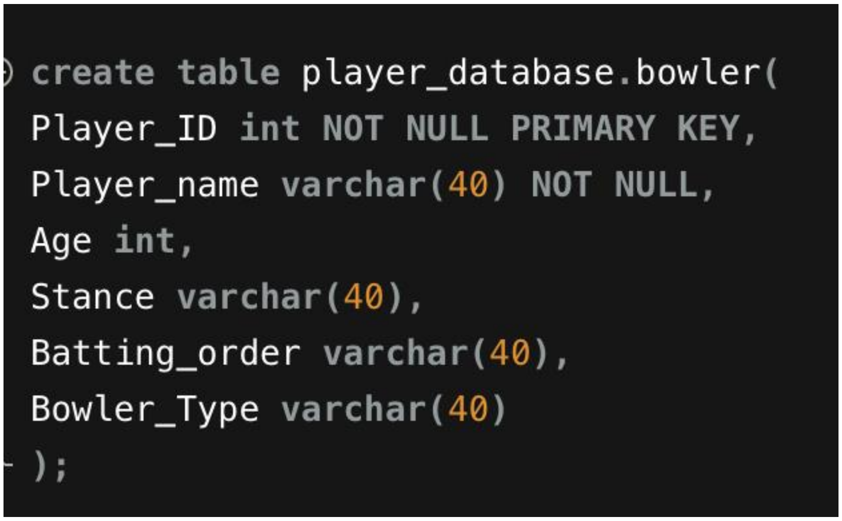
```sql
create table player_database.players(
Player_ID int NOT NULL PRIMARY KEY,
Player_name varchar(40) NOT NULL,
Age int,
Stance varchar(40),
Batting_order varchar(40),
Wicketkeeper varchar(40),
Bowler_Type varchar(40),
Auction_ID int,
Team_Name varchar(40)
);

alter table Player_database.players add FOREIGN KEY (Auction_ID) REFERENCES player_database.auction(Auction_ID);
alter table Player_database.players add FOREIGN KEY (Team_Name) REFERENCES player_database.team(Team_Name);
```

Batsmen Table:



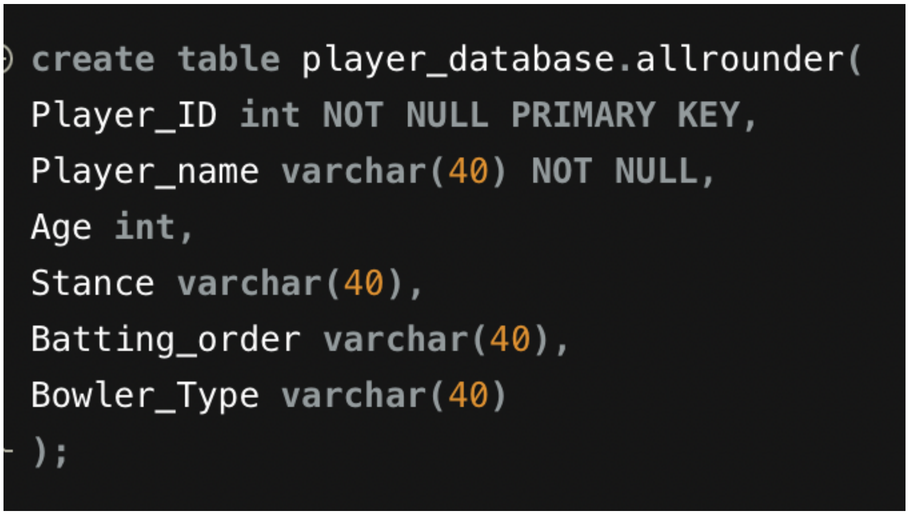
```sql
create table player_database.batsmen(
Player_ID int NOT NULL PRIMARY KEY,
Player_name varchar(40) NOT NULL,
Age int,
Stance varchar(40),
Batting_order varchar(40),
Wicketkeeper varchar(40),
Bowler_Type varchar(40)
);
```

Bowler Table:



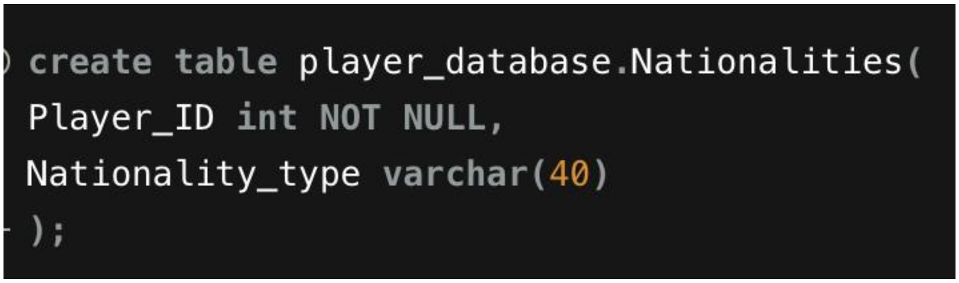
```
create table player_database.bowler(
Player_ID int NOT NULL PRIMARY KEY,
Player_name varchar(40) NOT NULL,
Age int,
Stance varchar(40),
Batting_order varchar(40),
Bowler_Type varchar(40)
);
```

All-rounder Table:



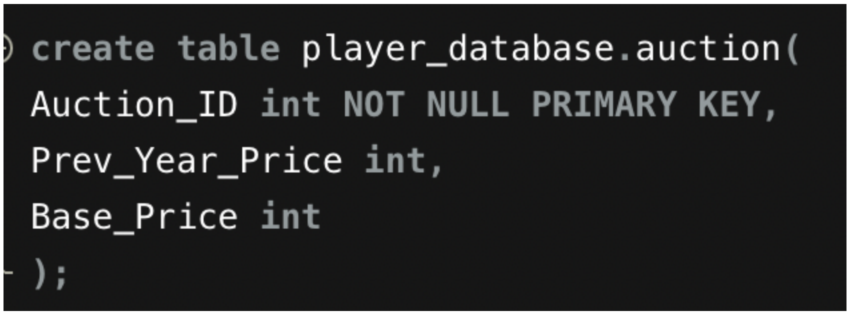
```
create table player_database.allrounder(
Player_ID int NOT NULL PRIMARY KEY,
Player_name varchar(40) NOT NULL,
Age int,
Stance varchar(40),
Batting_order varchar(40),
Bowler_Type varchar(40)
);
```

Nationalities Table:



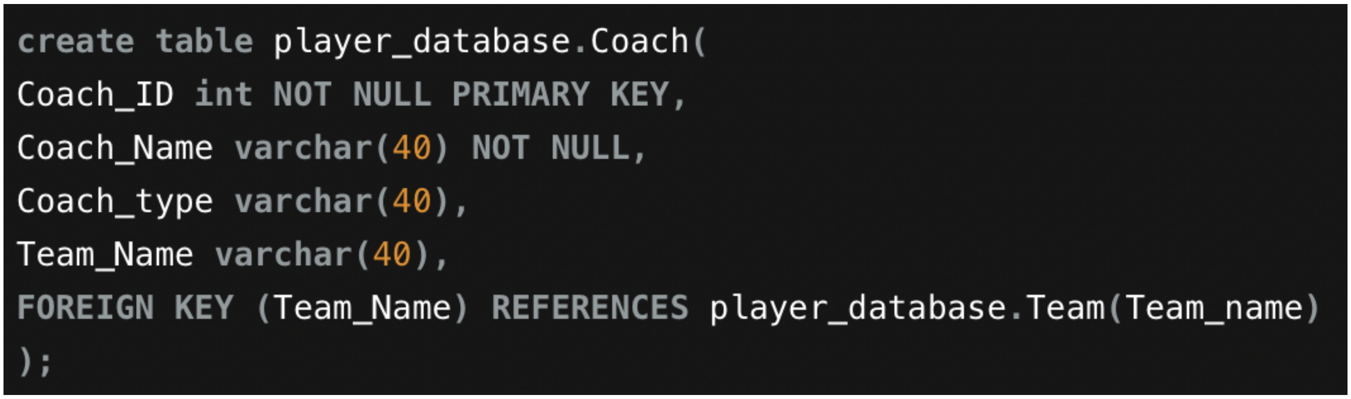
```
create table player_database.Nationalities(
Player_ID int NOT NULL,
Nationality_type varchar(40)
);
```

Auction Table:



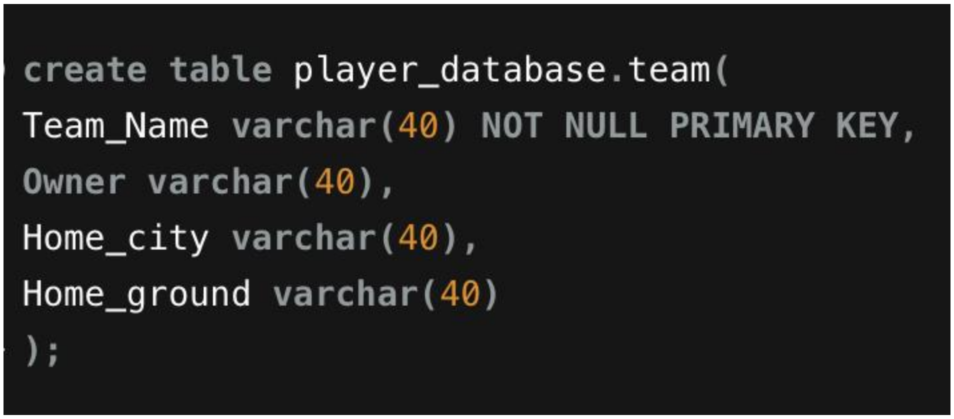
```sql
create table player_database.auction(
Auction_ID int NOT NULL PRIMARY KEY,
Prev_Year_Price int,
Base_Price int
);
```

Coach Table:



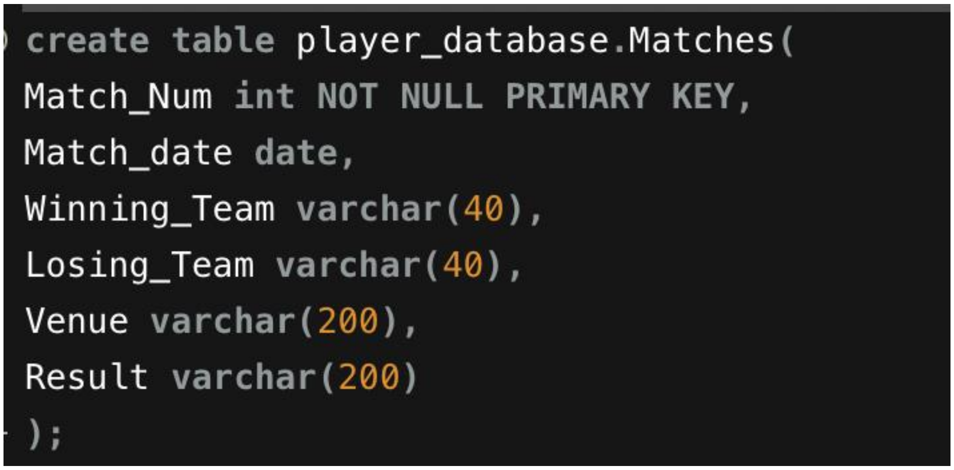
```sql
create table player_database.Coach(
Coach_ID int NOT NULL PRIMARY KEY,
Coach_Name varchar(40) NOT NULL,
Coach_type varchar(40),
Team_Name varchar(40),
FOREIGN KEY (Team_Name) REFERENCES player_database.Team(Team_name)
);
```

Team Table:



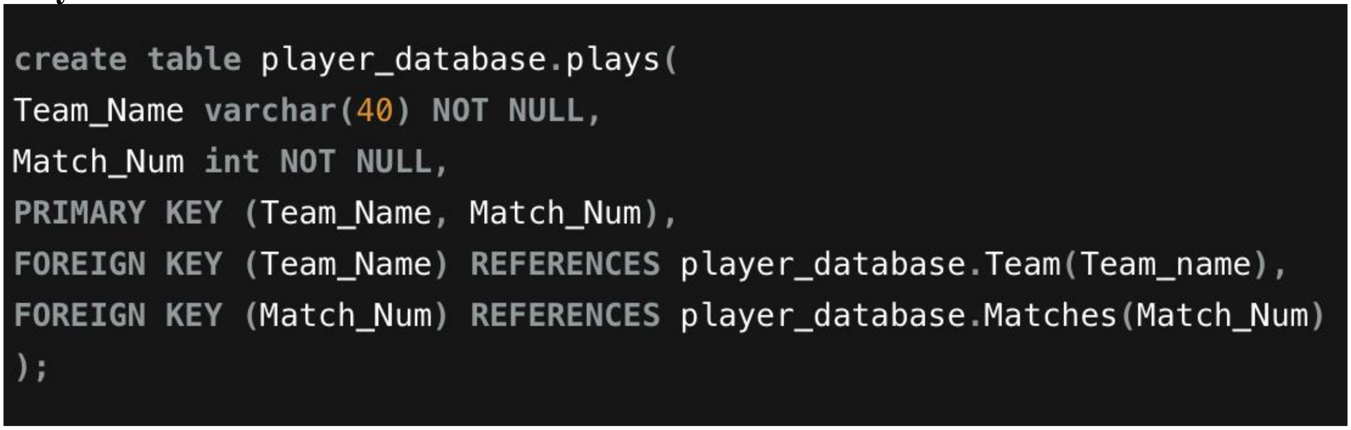
```sql
create table player_database.team(
Team_Name varchar(40) NOT NULL PRIMARY KEY,
Owner varchar(40),
Home_city varchar(40),
Home_ground varchar(40)
);
```

Matches Table:



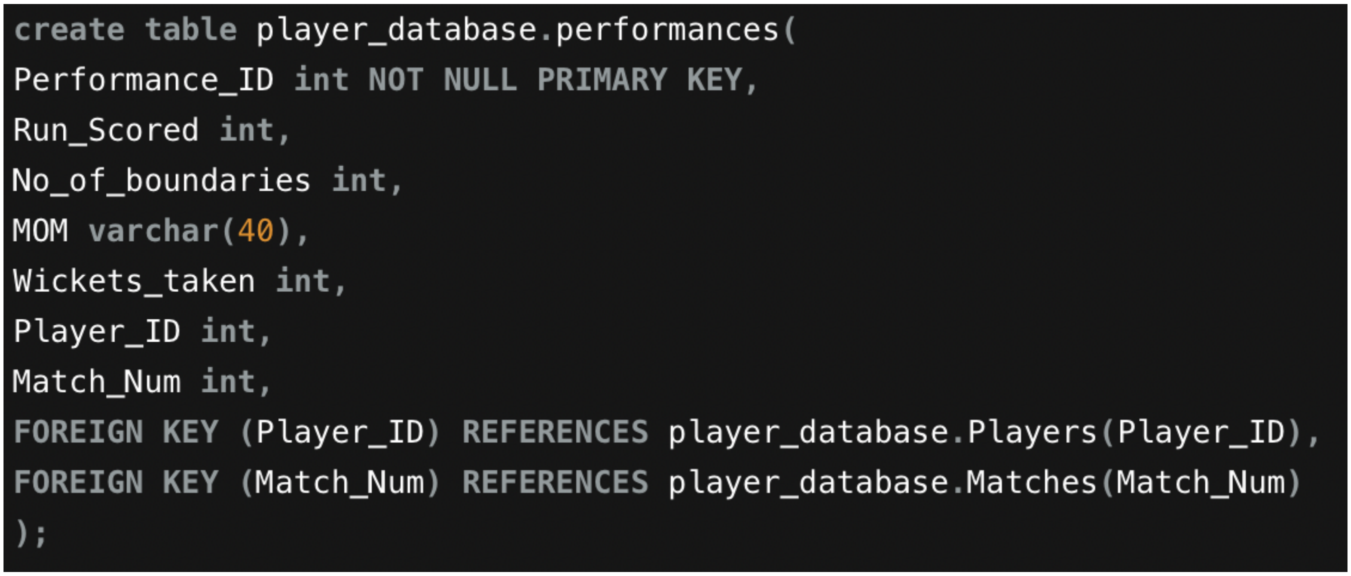
```
create table player_database.Matches(
Match_Num int NOT NULL PRIMARY KEY,
Match_date date,
Winning_Team varchar(40),
Losing_Team varchar(40),
Venue varchar(200),
Result varchar(200)
);
```

Plays Table:



```
create table player_database.plays(
Team_Name varchar(40) NOT NULL,
Match_Num int NOT NULL,
PRIMARY KEY (Team_Name, Match_Num),
FOREIGN KEY (Team_Name) REFERENCES player_database.Team(Team_name),
FOREIGN KEY (Match_Num) REFERENCES player_database.Matches(Match_Num)
);
```

Performances Table:



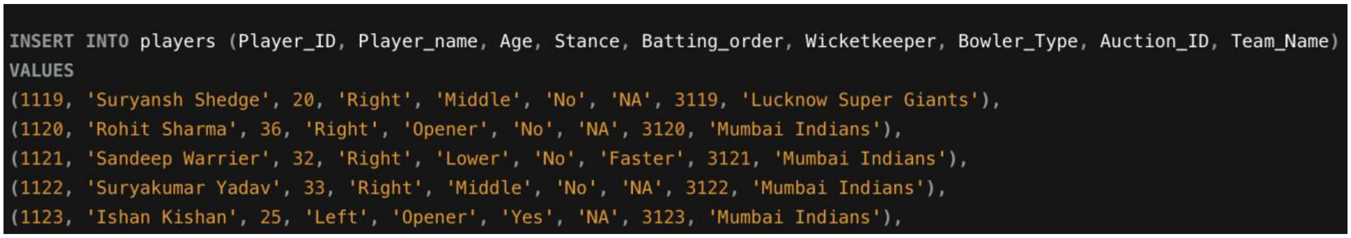
```
create table player_database.performances(
Performance_ID int NOT NULL PRIMARY KEY,
Run_Scored int,
No_of_boundaries int,
MOM varchar(40),
Wickets_taken int,
Player_ID int,
Match_Num int,
FOREIGN KEY (Player_ID) REFERENCES player_database.Players(Player_ID),
FOREIGN KEY (Match_Num) REFERENCES player_database.Matches(Match_Num)
);
```

INSERT STATEMENTS

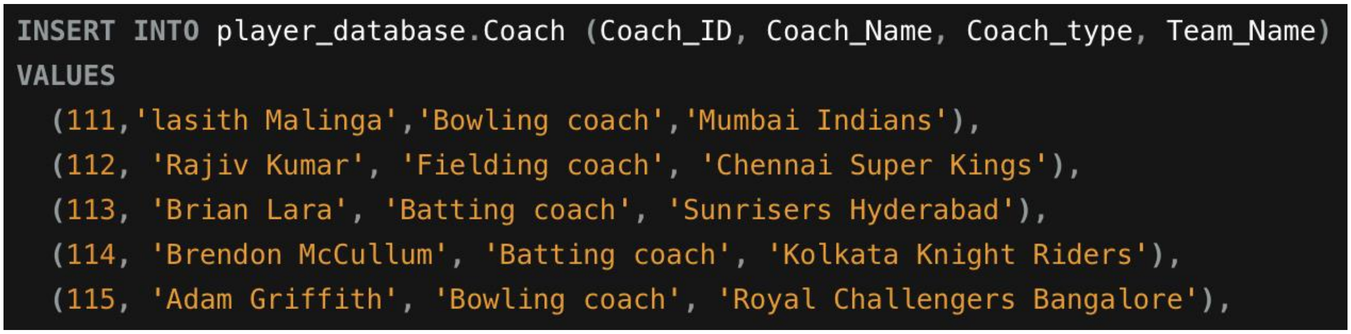
The Data was insert into the tables in two ways:

1. CSV file
2. Insert statements.

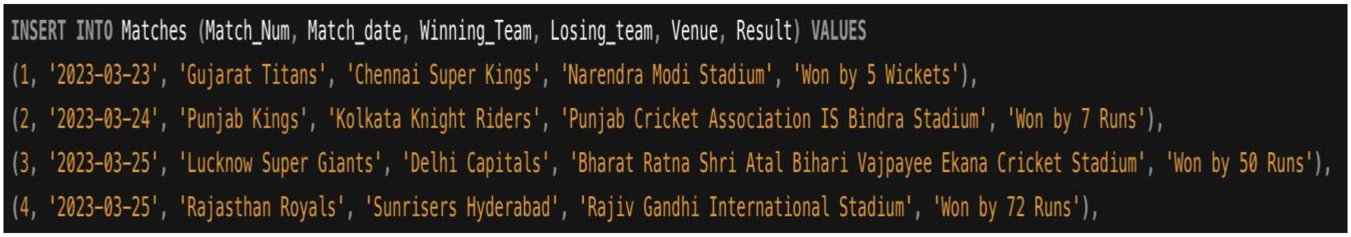
Depending on the requirements the data was inserted into the tables using either of the ways. Some of the sample insert statements are as follows:



```
INSERT INTO players (Player_ID, Player_name, Age, Stance, Batting_order, Wicketkeeper, Bowler_Type, Auction_ID, Team_Name)
VALUES
(1119, 'Suryansh Shedge', 20, 'Right', 'Middle', 'No', 'NA', 3119, 'Lucknow Super Giants'),
(1120, 'Rohit Sharma', 36, 'Right', 'Opener', 'No', 'NA', 3120, 'Mumbai Indians'),
(1121, 'Sandeep Warrier', 32, 'Right', 'Lower', 'No', 'Faster', 3121, 'Mumbai Indians'),
(1122, 'Suryakumar Yadav', 33, 'Right', 'Middle', 'No', 'NA', 3122, 'Mumbai Indians'),
(1123, 'Ishan Kishan', 25, 'Left', 'Opener', 'Yes', 'NA', 3123, 'Mumbai Indians'),
```



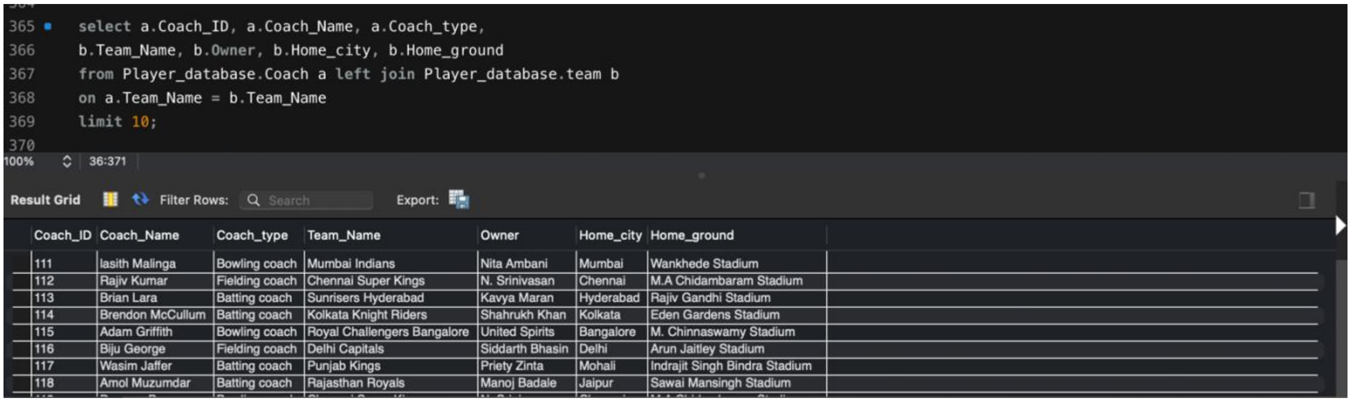
```
INSERT INTO player_database.Coach (Coach_ID, Coach_Name, Coach_type, Team_Name)
VALUES
  (111,'lasith Malinga','Bowling coach','Mumbai Indians'),
  (112, 'Rajiv Kumar', 'Fielding coach', 'Chennai Super Kings'),
  (113, 'Brian Lara', 'Batting coach', 'Sunrisers Hyderabad'),
  (114, 'Brendon McCullum', 'Batting coach', 'Kolkata Knight Riders'),
  (115, 'Adam Griffith', 'Bowling coach', 'Royal Challengers Bangalore'),
```



```
INSERT INTO Matches (Match_Num, Match_date, Winning_Team, Losing_team, Venue, Result) VALUES
(1, '2023-03-23', 'Gujarat Titans', 'Chennai Super Kings', 'Narendra Modi Stadium', 'Won by 5 Wickets'),
(2, '2023-03-24', 'Punjab Kings', 'Kolkata Knight Riders', 'Punjab Cricket Association IS Bindra Stadium', 'Won by 7 Runs'),
(3, '2023-03-25', 'Lucknow Super Giants', 'Delhi Capitals', 'Bharat Ratna Shri Atal Bihari Vajpayee Ekana Cricket Stadium', 'Won by 50 Runs'),
(4, '2023-03-25', 'Rajasthan Royals', 'Sunrisers Hyderabad', 'Rajiv Gandhi International Stadium', 'Won by 72 Runs'),
```

ANALYTICAL QUERIES

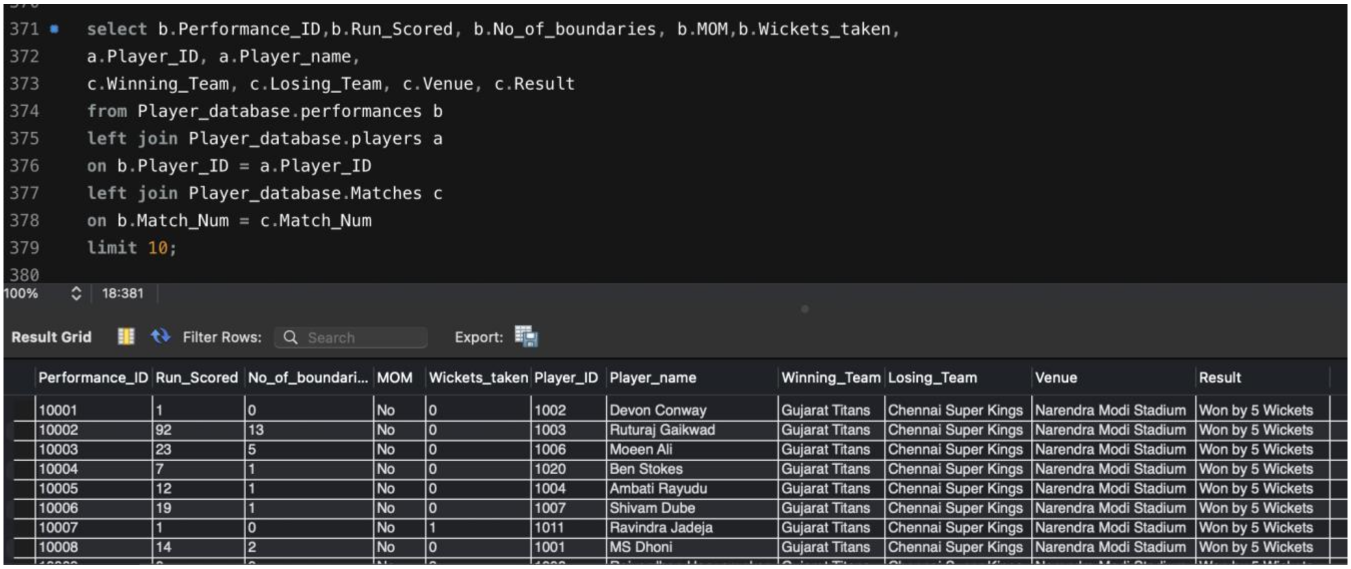
These queries will be used in the analysis of the data. These are:



```
select a.Coach_ID, a.Coach_Name, a.Coach_type,
b.Team_Name, b.Owner, b.Home_city, b.Home_ground
from Player_database.Coach a left join Player_database.team b
on a.Team_Name = b.Team_Name
limit 10;
```

Coach_ID	Coach_Name	Coach_type	Team_Name	Owner	Home_city	Home_ground
111	lasith Malinga	Bowling coach	Mumbai Indians	Nita Ambani	Mumbai	Wankhede Stadium
112	Rajiv Kumar	Fielding coach	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
113	Brian Lara	Batting coach	Sunrisers Hyderabad	Kavya Maran	Hyderabad	Rajiv Gandhi Stadium
114	Brendon McCullum	Batting coach	Kolkata Knight Riders	Shahrukh Khan	Kolkata	Eden Gardens Stadium
115	Adam Griffith	Bowling coach	Royal Challengers Bangalore	United Spirits	Bangalore	M. Chinnaswamy Stadium
116	Biju George	Fielding coach	Delhi Capitals	Siddarth Bhasin	Delhi	Arun Jaitley Stadium
117	Wasim Jaffer	Batting coach	Punjab Kings	Priety Zinta	Mohali	Indrajit Singh Bindra Stadium
118	Amol Muzumdar	Batting coach	Rajasthan Royals	Manoj Badale	Jaipur	Sawai Mansingh Stadium

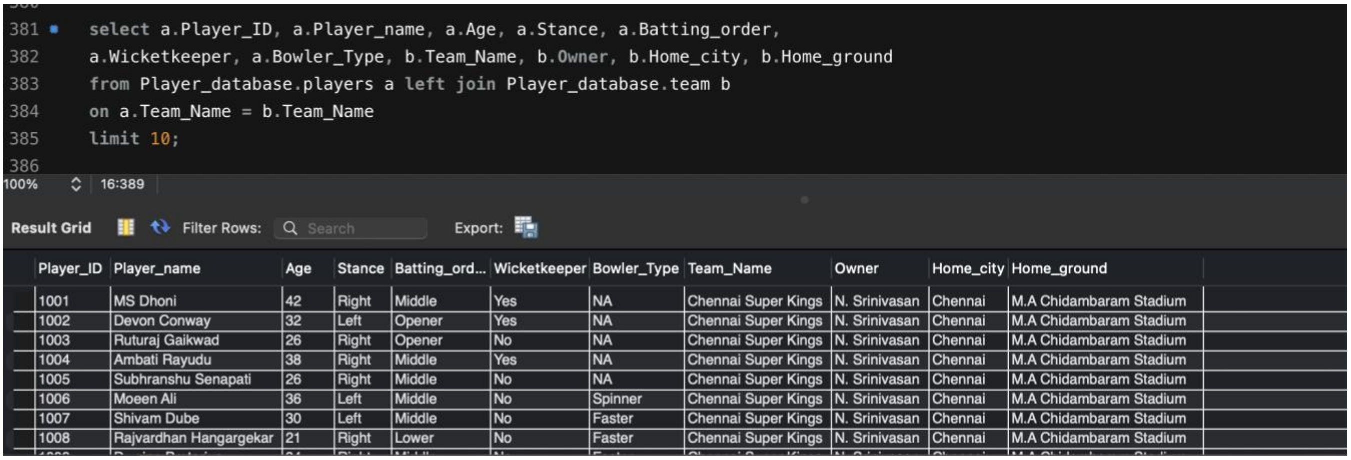
This query tells us about the coaches of the respective teams. e.g Lasith Malinga – Mumbai Indians



```
select b.Performance_ID,b.Run_Scored, b.No_of_boundaries, b.MOM,b.Wickets_taken,
a.Player_ID, a.Player_name,
c.Winning_Team, c.Losing_Team, c.Venue, c.Result
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
left join Player_database.Matches c
on b.Match_Num = c.Match_Num
limit 10;
```

Performance_ID	Run_Scored	No_of_boundari...	MOM	Wickets_taken	Player_ID	Player_name	Winning_Team	Losing_Team	Venue	Result
10001	1	0	No	0	1002	Devon Conway	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10002	92	13	No	0	1003	Ruturaj Gaikwad	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10003	23	5	No	0	1006	Moeen Ali	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10004	7	1	No	0	1020	Ben Stokes	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10005	12	1	No	0	1004	Ambati Rayudu	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10006	19	1	No	0	1007	Shivam Dube	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10007	1	0	No	1	1011	Ravindra Jadeja	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets
10008	14	2	No	0	1001	MS Dhoni	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets

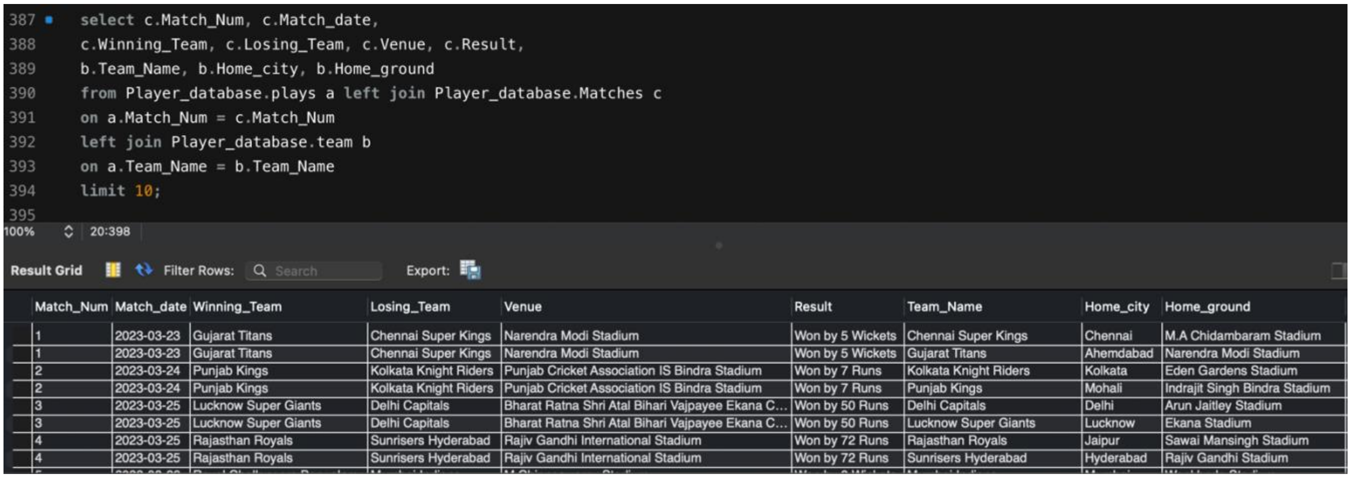
This query tells us about every player performance in all the matches the particular player has played.



```sql
select a.Player_ID, a.Player_name, a.Age, a.Stance, a.Batting_order,
a.Wicketkeeper, a.Bowler_Type, b.Team_Name, b.Owner, b.Home_city, b.Home_ground
from Player_database.players a left join Player_database.team b
on a.Team_Name = b.Team_Name
limit 10;
```

Player_ID	Player_name	Age	Stance	Batting_ord...	Wicketkeeper	Bowler_Type	Team_Name	Owner	Home_city	Home_ground
1001	MS Dhoni	42	Right	Middle	Yes	NA	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1002	Devon Conway	32	Left	Opener	Yes	NA	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1003	Ruturaj Gaikwad	26	Right	Opener	No	NA	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1004	Ambati Rayudu	38	Right	Middle	Yes	NA	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1005	Subhranshu Senapati	26	Right	Middle	No	NA	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1006	Moeen Ali	36	Left	Middle	No	Spinner	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1007	Shivam Dube	30	Left	Middle	No	Faster	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium
1008	Rajvardhan Hangargekar	21	Right	Lower	No	Faster	Chennai Super Kings	N. Srinivasan	Chennai	M.A Chidambaram Stadium

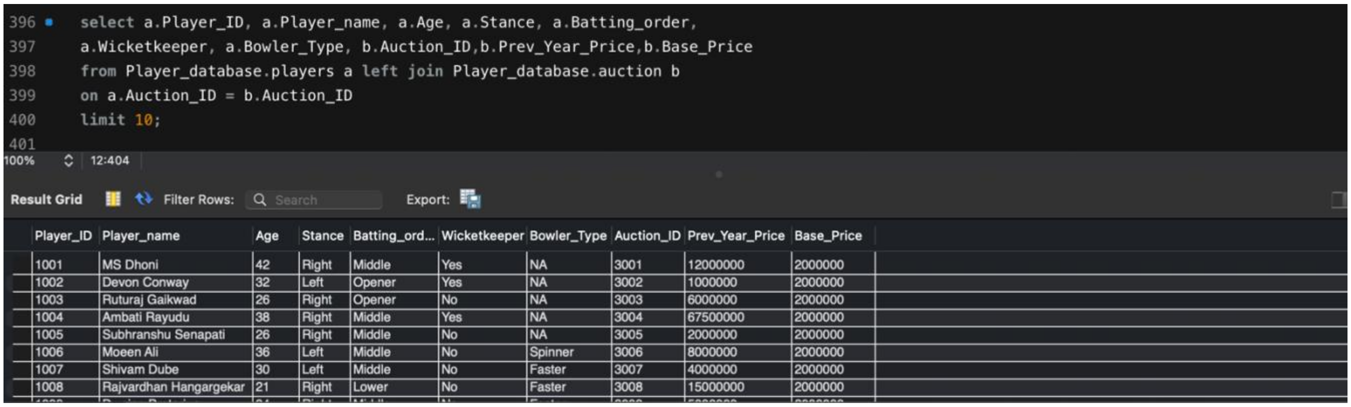
This query tells about the details of the all the players who were in the teams in the season of 2023.



```sql
select c.Match_Num, c.Match_date,
c.Winning_Team, c.Losing_Team, c.Venue, c.Result,
b.Team_Name, b.Home_city, b.Home_ground
from Player_database.plays a left join Player_database.Matches c
on a.Match_Num = c.Match_Num
left join Player_database.team b
on a.Team_Name = b.Team_Name
limit 10;
```

Match_Num	Match_date	Winning_Team	Losing_Team	Venue	Result	Team_Name	Home_city	Home_ground
1	2023-03-23	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets	Chennai Super Kings	Chennai	M.A Chidambaram Stadium
1	2023-03-23	Gujarat Titans	Chennai Super Kings	Narendra Modi Stadium	Won by 5 Wickets	Gujarat Titans	Ahemdabad	Narendra Modi Stadium
2	2023-03-24	Punjab Kings	Kolkata Knight Riders	Punjab Cricket Association IS Bindra Stadium	Won by 7 Runs	Kolkata Knight Riders	Kolkata	Eden Gardens Stadium
2	2023-03-24	Punjab Kings	Kolkata Knight Riders	Punjab Cricket Association IS Bindra Stadium	Won by 7 Runs	Punjab Kings	Mohali	Indrajit Singh Bindra Stadium
3	2023-03-25	Lucknow Super Giants	Delhi Capitals	Bharat Ratna Shri Atal Bihari Vajpayee Ekana C...	Won by 50 Runs	Delhi Capitals	Delhi	Arun Jaitley Stadium
3	2023-03-25	Lucknow Super Giants	Delhi Capitals	Bharat Ratna Shri Atal Bihari Vajpayee Ekana C...	Won by 50 Runs	Lucknow Super Giants	Lucknow	Ekana Stadium
4	2023-03-25	Rajasthan Royals	Sunrisers Hyderabad	Rajiv Gandhi International Stadium	Won by 72 Runs	Rajasthan Royals	Jaipur	Sawai Mansingh Stadium
4	2023-03-25	Rajasthan Royals	Sunrisers Hyderabad	Rajiv Gandhi International Stadium	Won by 72 Runs	Sunrisers Hyderabad	Hyderabad	Rajiv Gandhi Stadium

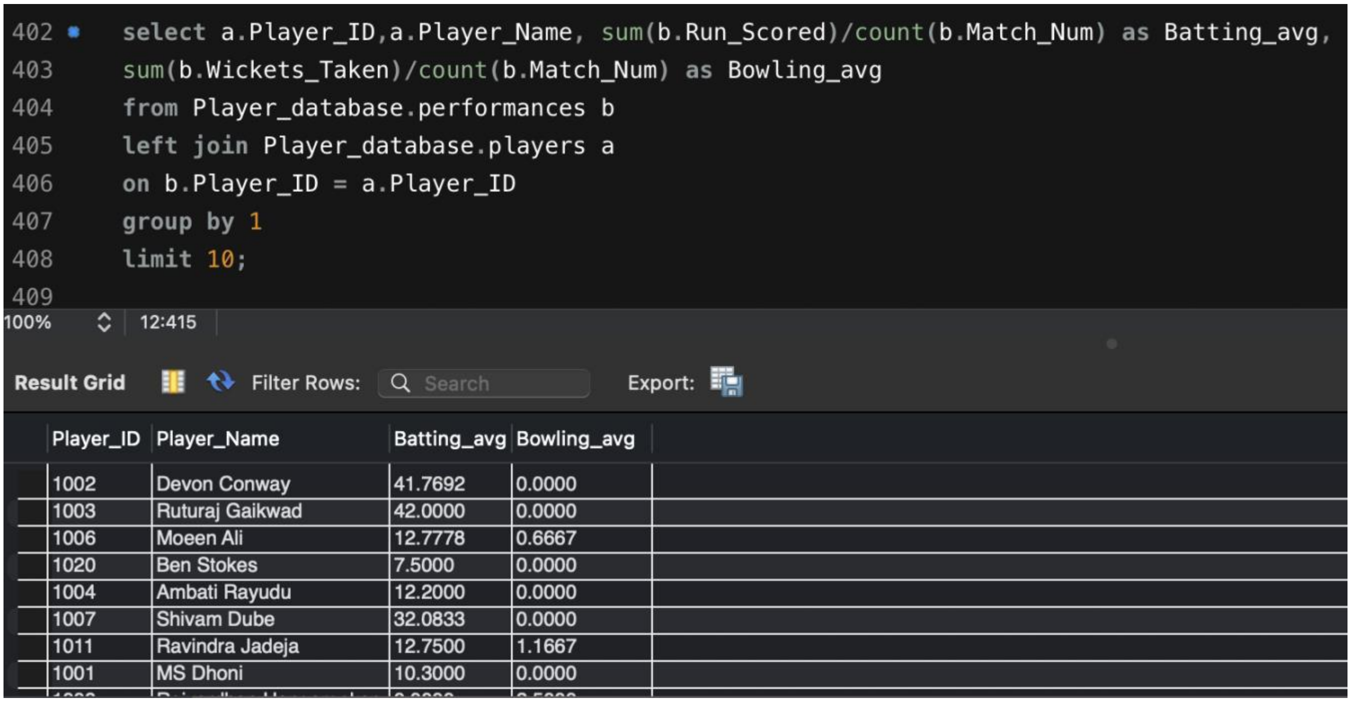
This query tells about every match details in the season of IPL 2023.



```
select a.Player_ID, a.Player_name, a.Age, a.Stance, a.Batting_order,
a.Wicketkeeper, a.Bowler_Type, b.Auction_ID,b.Prev_Year_Price,b.Base_Price
from Player_database.players a left join Player_database.auction b
on a.Auction_ID = b.Auction_ID
limit 10;
```

Player_ID	Player_name	Age	Stance	Batting_ord...	Wicketkeeper	Bowler_Type	Auction_ID	Prev_Year_Price	Base_Price
1001	MS Dhoni	42	Right	Middle	Yes	NA	3001	12000000	2000000
1002	Devon Conway	32	Left	Opener	Yes	NA	3002	1000000	2000000
1003	Ruturaj Gaikwad	26	Right	Opener	No	NA	3003	6000000	2000000
1004	Ambati Rayudu	38	Right	Middle	Yes	NA	3004	67500000	2000000
1005	Subhranshu Senapati	26	Right	Middle	No	NA	3005	2000000	2000000
1006	Moeen Ali	36	Left	Middle	No	Spinner	3006	8000000	2000000
1007	Shivam Dube	30	Left	Middle	No	Faster	3007	4000000	2000000
1008	Rajvardhan Hangargekar	21	Right	Lower	No	Faster	3008	15000000	2000000

This query tells about the player details and the auction details.

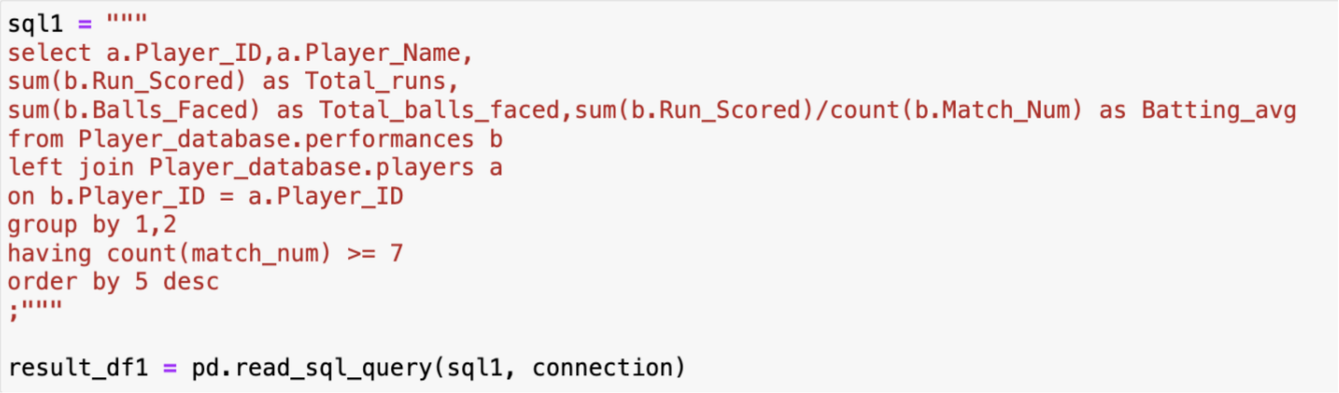


```
select a.Player_ID,a.Player_Name, sum(b.Run_Scored)/count(b.Match_Num) as Batting_avg,
sum(b.Wickets_Taken)/count(b.Match_Num) as Bowling_avg
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
group by 1
limit 10;
```

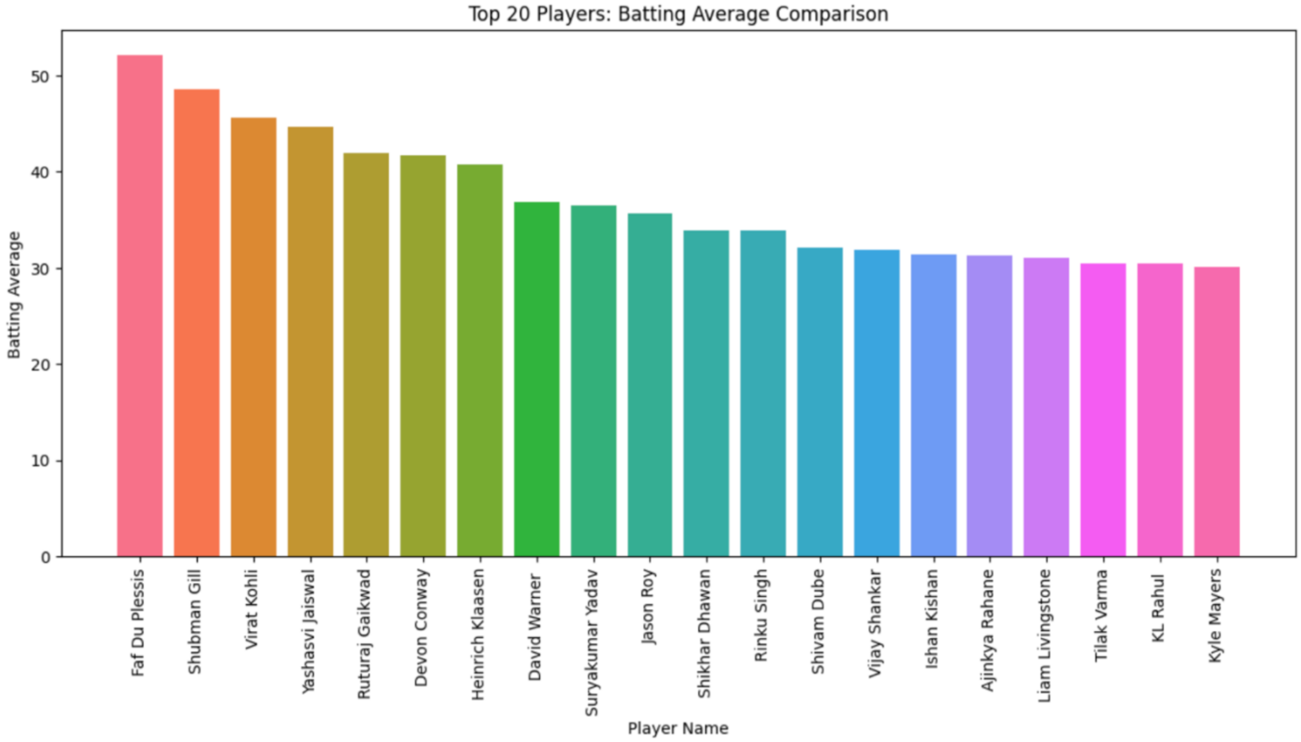
Player_ID	Player_Name	Batting_avg	Bowling_avg
1002	Devon Conway	41.7692	0.0000
1003	Ruturaj Gaikwad	42.0000	0.0000
1006	Moeen Ali	12.7778	0.6667
1020	Ben Stokes	7.5000	0.0000
1004	Ambati Rayudu	12.2000	0.0000
1007	Shivam Dube	32.0833	0.0000
1011	Ravindra Jadeja	12.7500	1.1667
1001	MS Dhoni	10.3000	0.0000

This query tells about the batting average and bowling of each player in the database.

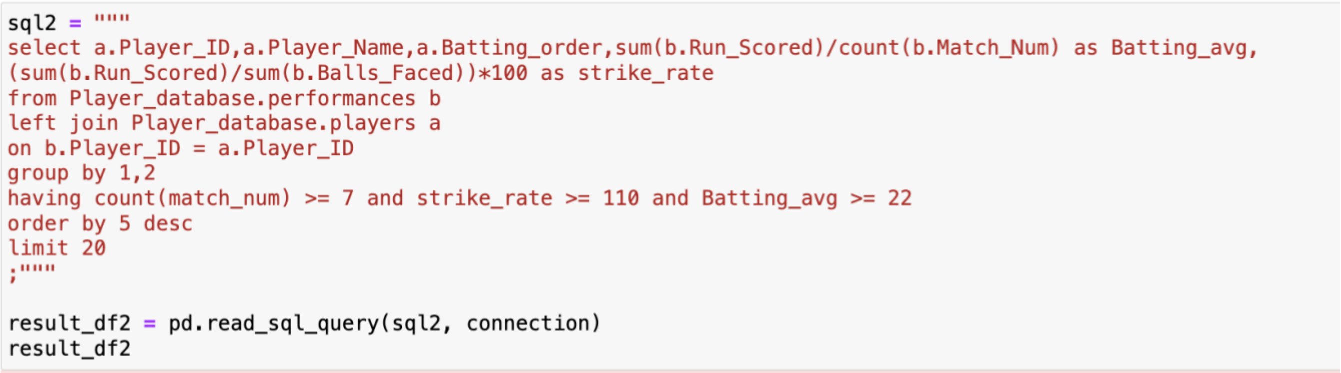
Other Analytical Queries with Visualization (Using Python)



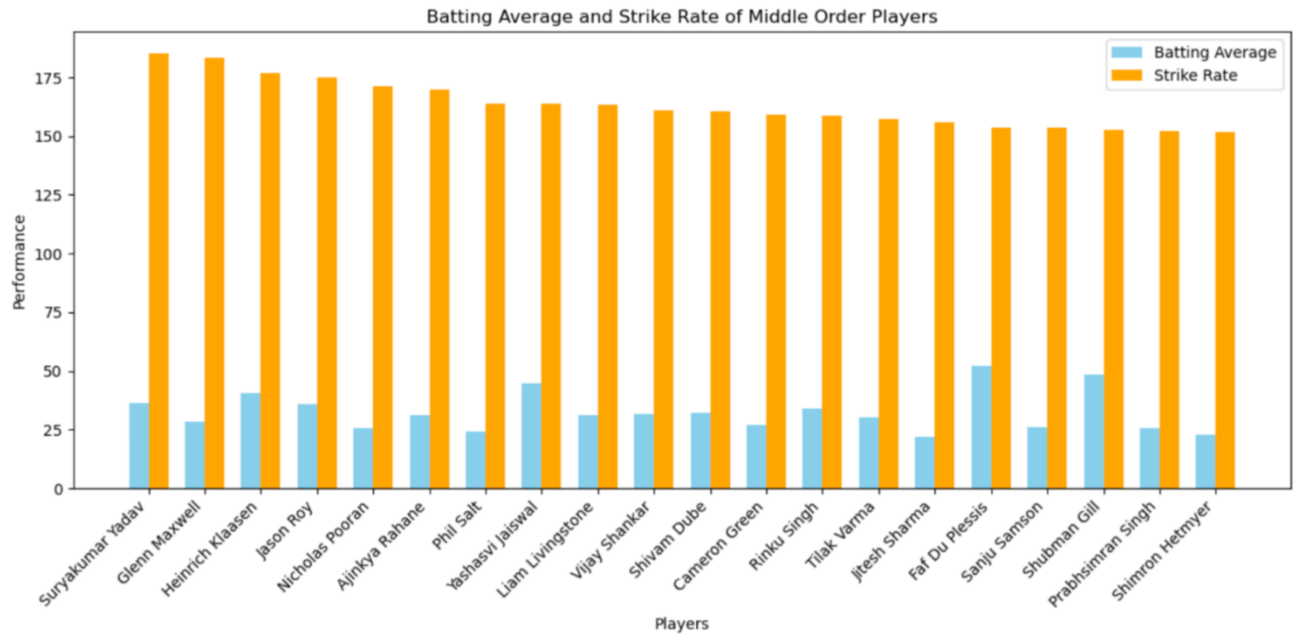
```
sql1 = """
select a.Player_ID,a.Player_Name,
sum(b.Run_Scored) as Total_runs,
sum(b.Balls_Faced) as Total_balls_faced,sum(b.Run_Scored)/count(b.Match_Num) as Batting_avg
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
group by 1,2
having count(match_num) >= 7
order by 5 desc
;"""

result_df1 = pd.read_sql_query(sql1, connection)
```



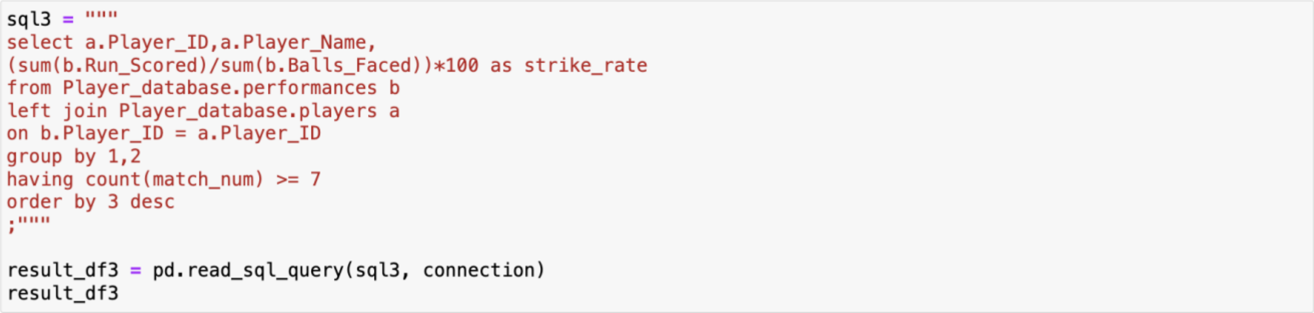
This query gives the top 20 players with the best batting average.



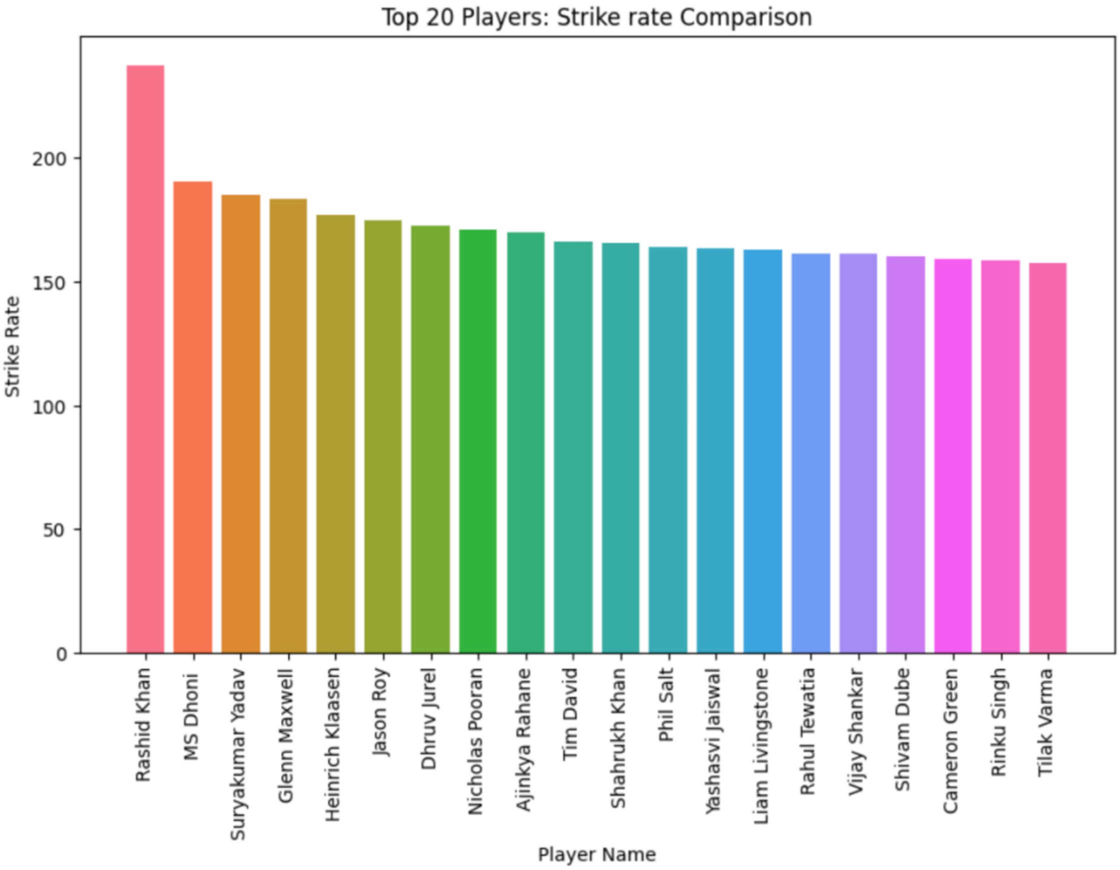
```
sql2 = """
select a.Player_ID,a.Player_Name,a.Batting_order,sum(b.Run_Scored)/count(b.Match_Num) as Batting_avg,
(sum(b.Run_Scored)/sum(b.Balls_Faced))*100 as strike_rate
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
group by 1,2
having count(match_num) >= 7 and strike_rate >= 110 and Batting_avg >= 22
order by 5 desc
limit 20
;"""

result_df2 = pd.read_sql_query(sql2, connection)
result_df2
```



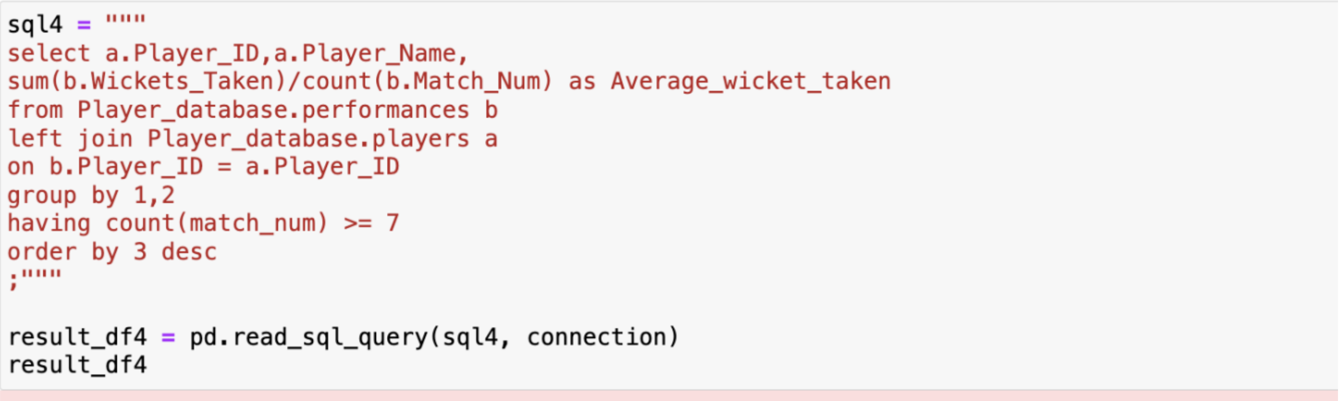
This query shows that Surya Kumar Yadav is the best player in the middle order among all the players driven from the query.



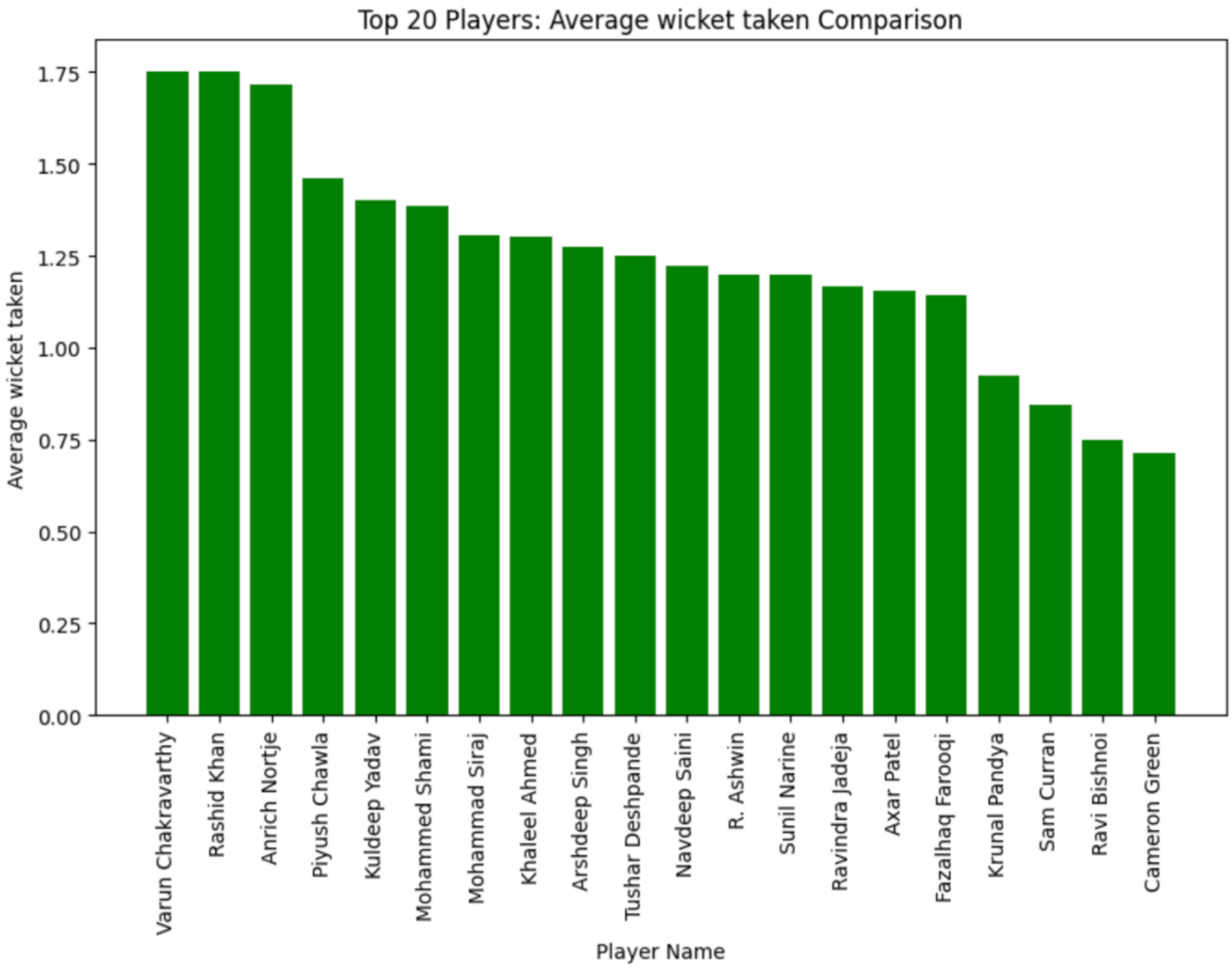
```
sql3 = """
select a.Player_ID,a.Player_Name,
(sum(b.Run_Scored)/sum(b.Balls_Faced))*100 as strike_rate
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
group by 1,2
having count(match_num) >= 7
order by 3 desc
;"""

result_df3 = pd.read_sql_query(sql3, connection)
result_df3
```



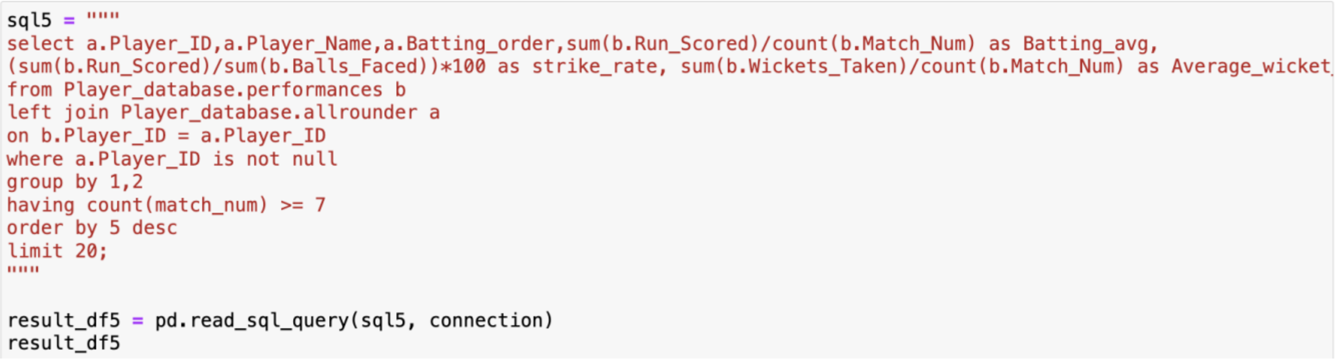
This query tells the comparison of the players with the best strike rate in the season of 2023 IPL.



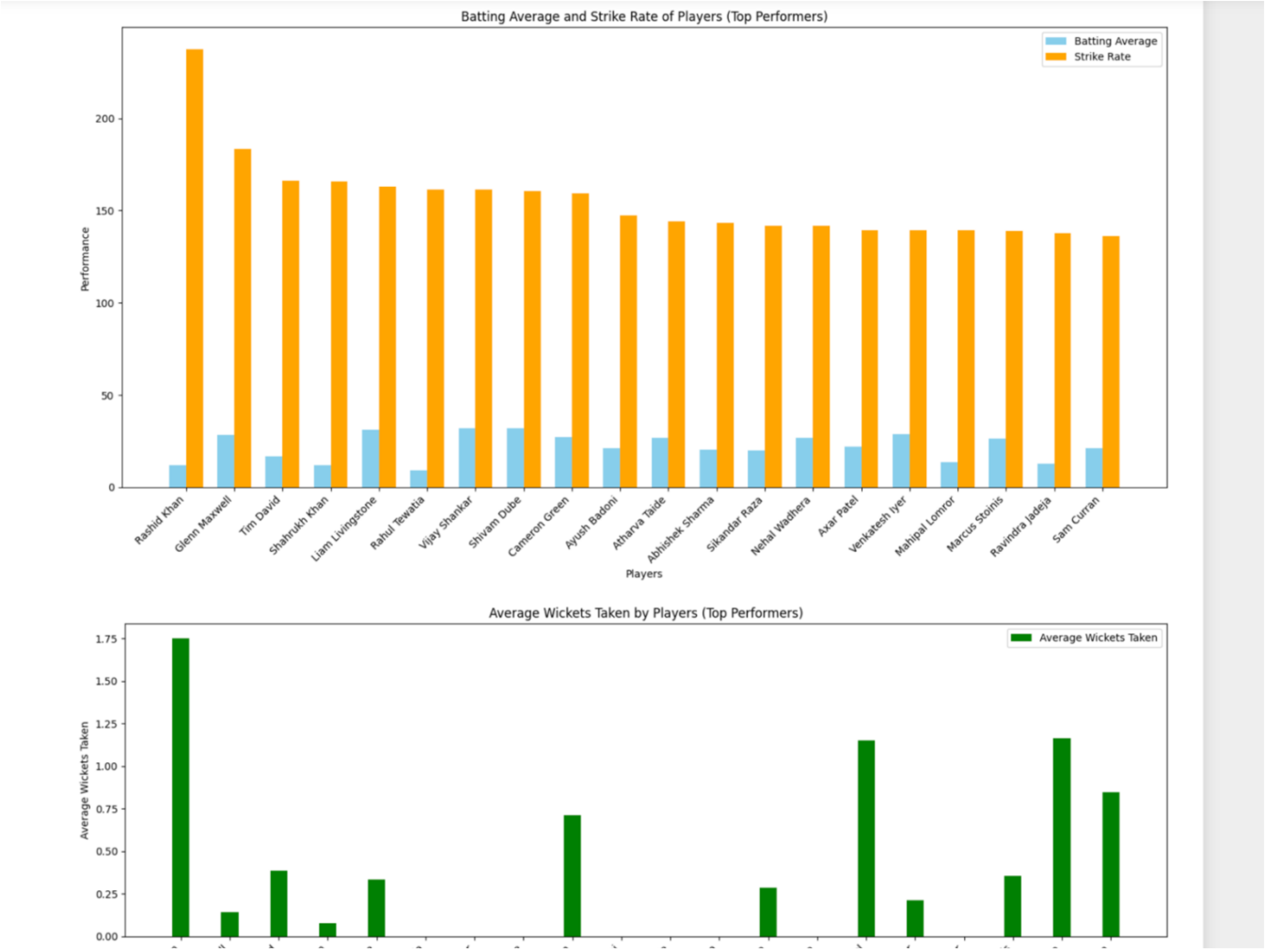
```
sql4 = """
select a.Player_ID,a.Player_Name,
sum(b.Wickets_Taken)/count(b.Match_Num) as Average_wicket_taken
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
group by 1,2
having count(match_num) >= 7
order by 3 desc
;"""

result_df4 = pd.read_sql_query(sql4, connection)
result_df4
```



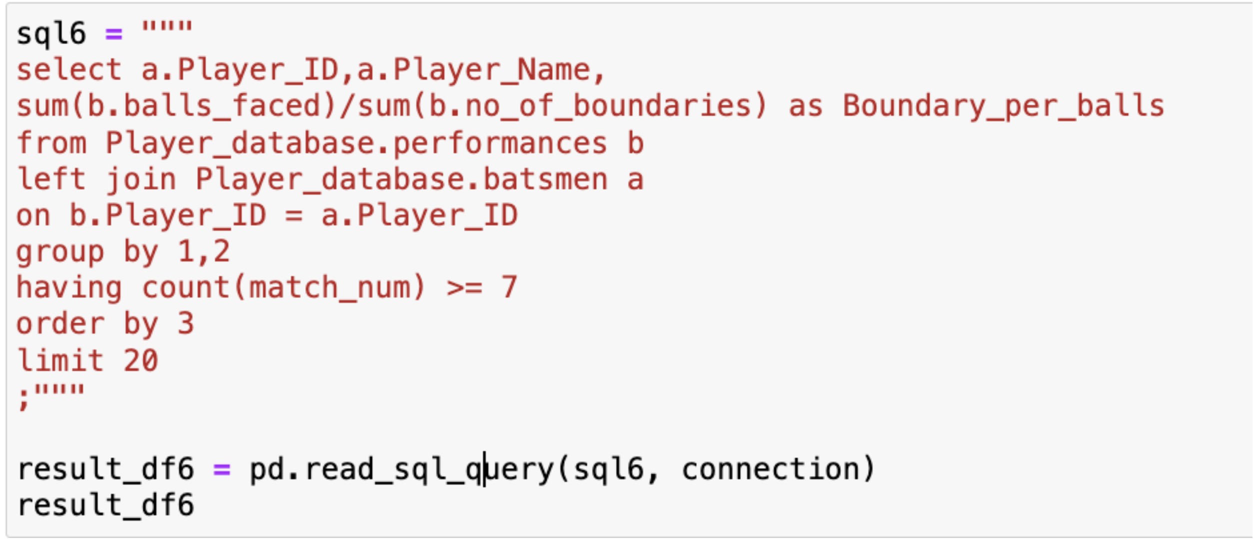
This query tells about the top 20 bowlers with the best average for taking wickets more frequently in their respective spells in a match.



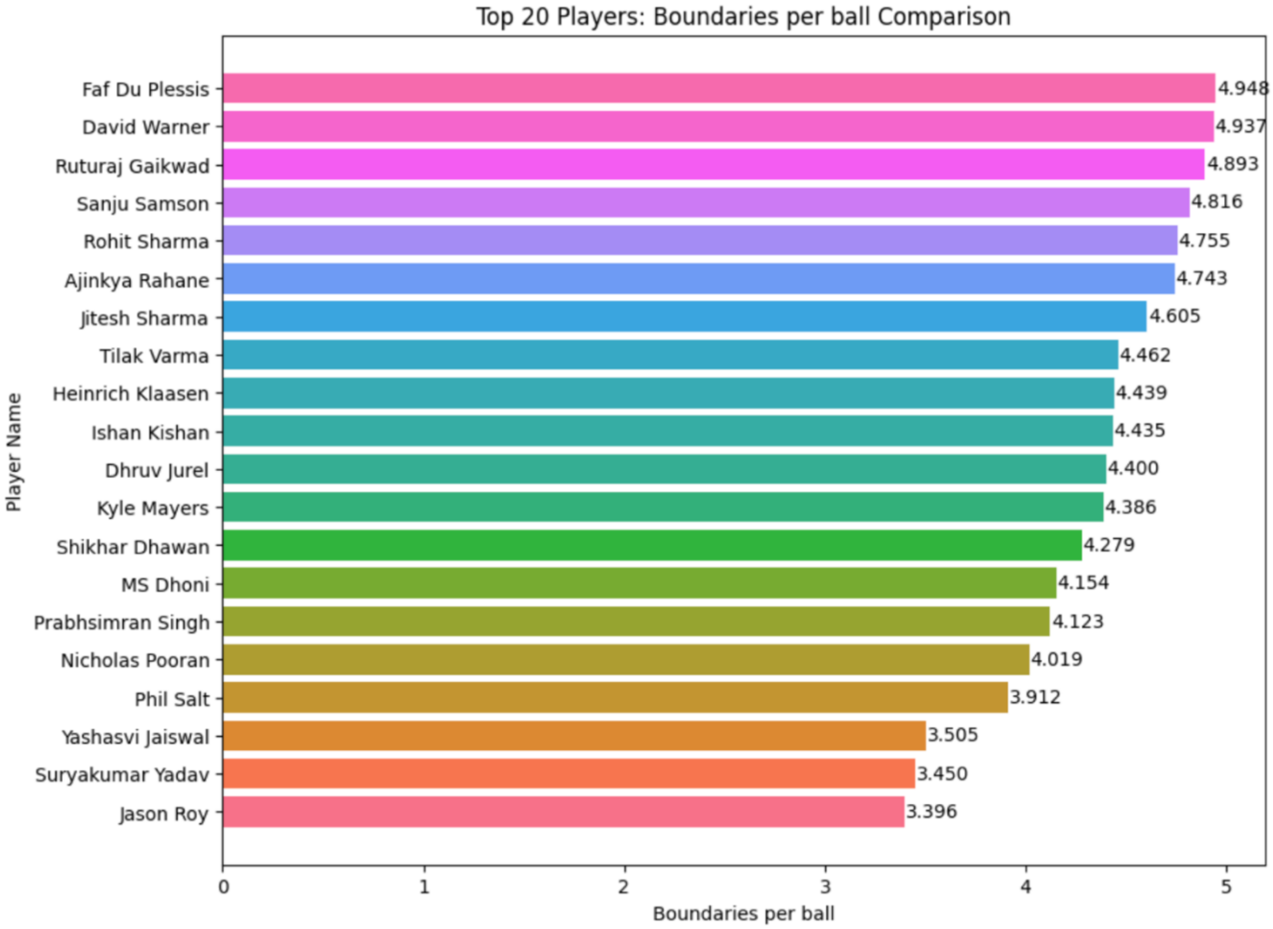
```
sql5 = """
select a.Player_ID,a.Player_Name,a.Batting_order,sum(b.Run_Scored)/count(b.Match_Num) as Batting_avg,
(sum(b.Run_Scored)/sum(b.Balls_Faced))*100 as strike_rate, sum(b.Wickets_Taken)/count(b.Match_Num) as Average_wicket
from Player_database.performances b
left join Player_database.allrounder a
on b.Player_ID = a.Player_ID
where a.Player_ID is not null
group by 1,2
having count(match_num) >= 7
order by 5 desc
limit 20;
"""

result_df5 = pd.read_sql_query(sql5, connection)
result_df5
```



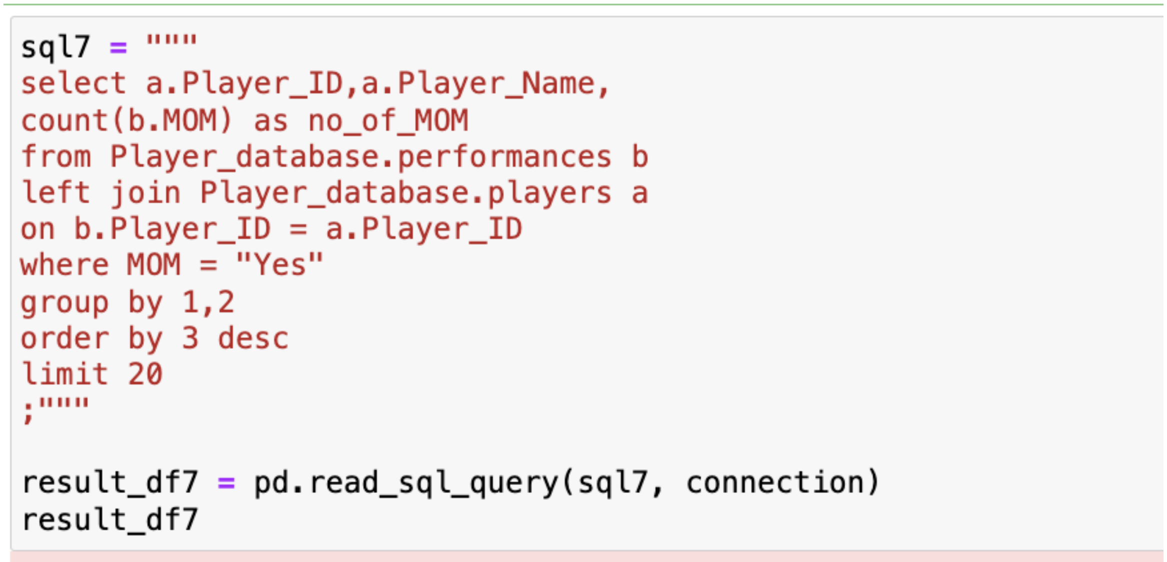
This query tells about the top all-rounders in the season of 2023 IPL.



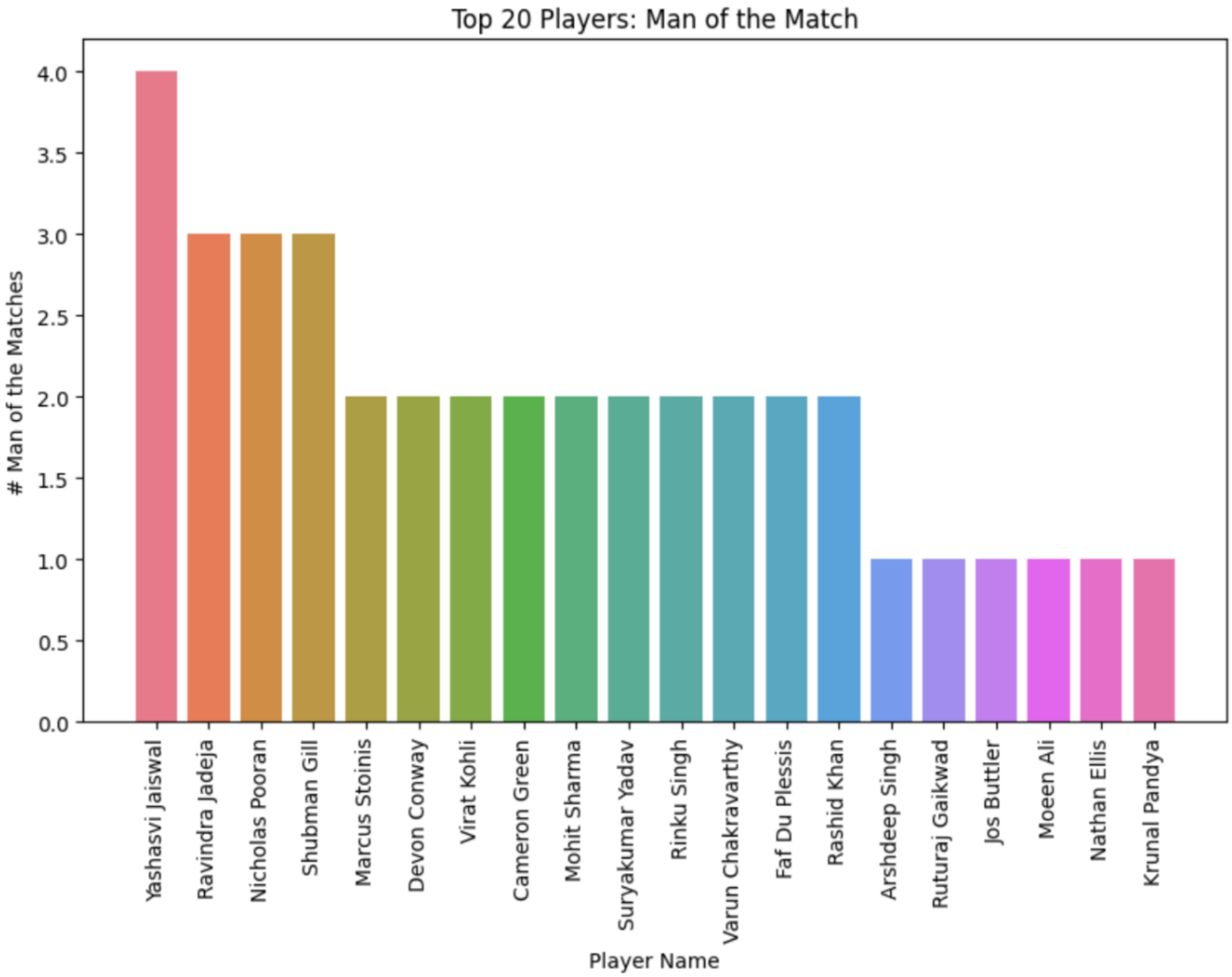
```
sql6 = """
select a.Player_ID,a.Player_Name,
sum(b.balls_faced)/sum(b.no_of_boundaries) as Boundary_per_balls
from Player_database.performances b
left join Player_database.batsmen a
on b.Player_ID = a.Player_ID
group by 1,2
having count(match_num) >= 7
order by 3
limit 20
;"""

result_df6 = pd.read_sql_query(sql6, connection)
result_df6
```



This query tells about the players taking minimum balls to hit boundaries where Jason Roy and Surya Kumar Yadav taking only 3-4 balls to hit their first and the next boundary.

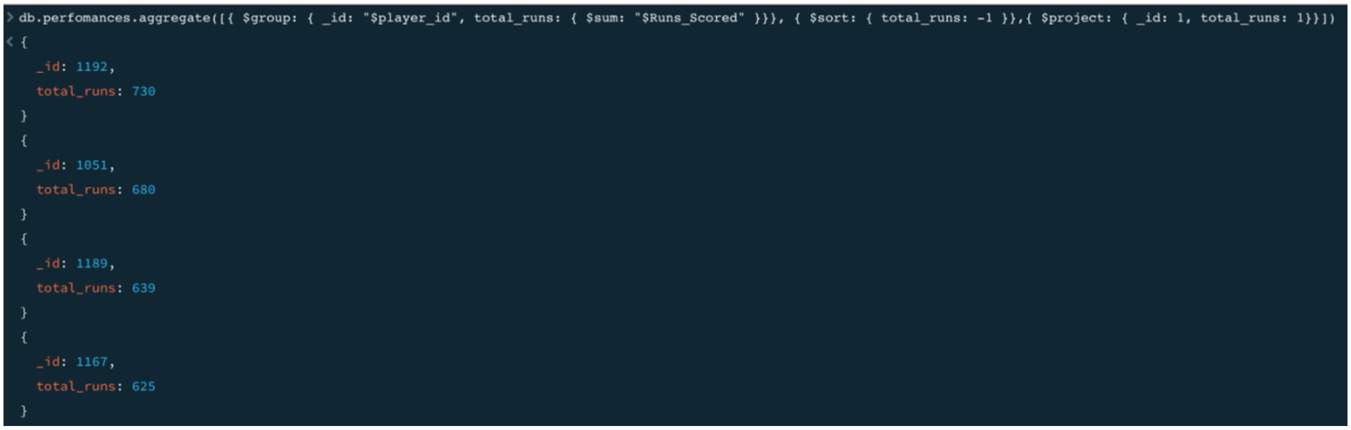


```
sql7 = """
select a.Player_ID,a.Player_Name,
count(b.MOM) as no_of_MOM
from Player_database.performances b
left join Player_database.players a
on b.Player_ID = a.Player_ID
where MOM = "Yes"
group by 1,2
order by 3 desc
limit 20
;"""

result_df7 = pd.read_sql_query(sql7, connection)
result_df7
```



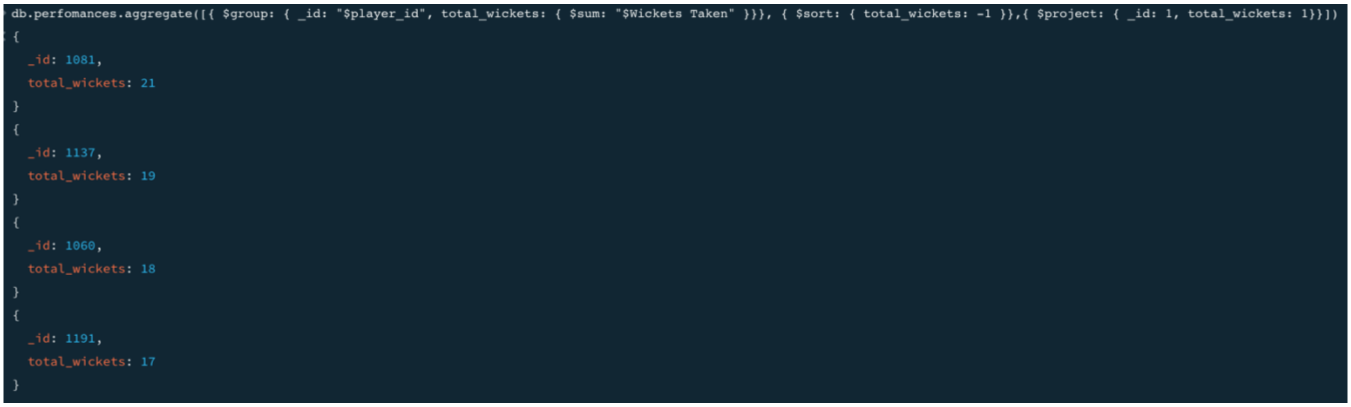
This query tells about the players with the most man of the matches in the season of IPL 2023.

NOSQL Implementation



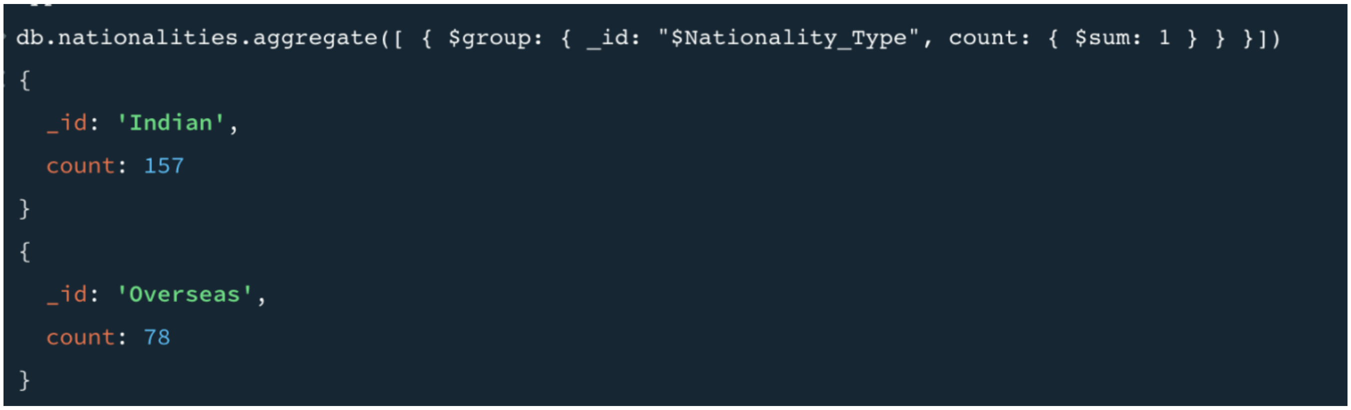
```
> db.perfomances.aggregate([{ $group: { _id: "$player_id", total_runs: { $sum: "$Runs_Scored" }}}, { $sort: { total_runs: -1 }},{ $project: { _id: 1, total_runs: 1}}])
< {
    _id: 1192,
    total_runs: 730
  }
  {
    _id: 1051,
    total_runs: 680
  }
  {
    _id: 1189,
    total_runs: 639
  }
  {
    _id: 1167,
    total_runs: 625
  }
```

This query gives the number of runs scored by players in the season of IPL 2023.



```
db.perfomances.aggregate([{ $group: { _id: "$player_id", total_wickets: { $sum: "$Wickets Taken" }}}, { $sort: { total_wickets: -1 }},{ $project: { _id: 1, total_wickets: 1}}])
{
  _id: 1081,
  total_wickets: 21
}
{
  _id: 1137,
  total_wickets: 19
}
{
  _id: 1060,
  total_wickets: 18
}
{
  _id: 1191,
  total_wickets: 17
}
```

This query gives the number of wickets taken by players in the season of IPL 2023.



```
db.nationalities.aggregate([ { $group: { _id: "$Nationality_Type", count: { $sum: 1 } } }])
{
  _id: 'Indian',
  count: 157
}
{
  _id: 'Overseas',
  count: 78
}
```

This query gives the distribution of the players according to their nationality in the database.

Conclusion

Our Database system will provide teams the solution to get the best player out of the bunch who can fit in all different positions and different situations in the match. Our database gives the comprehensive view of every player's performance which will be beneficial for the teams to purchase appropriate players for the next season.