Objective Questions

1. List the different types of columns in table “ball\_by\_ball” (using information schema)

**Ans:** There are 11 columns with the data type integer.

A screenshot of a computer

Description automatically generated

This SQL query retrieves the column names and their corresponding data types from the ball\_by\_ball table using the INFORMATION\_SCHEMA.COLUMNS view.

A white background with blue text

Description automatically generated

* **INFORMATION\_SCHEMA.COLUMNS:** A system catalog view that contains metadata about all columns in all tables within the database.
* **COLUMN\_NAME:** The name of each column in the specified table.
* **DATA\_TYPE:** The data type of each column (e.g., INT, VARCHAR, DATE).
* **TABLE\_NAME = 'ball\_by\_ball':** Filters the results to show only the columns for the ball\_by\_ball table.

1. What is the total number of runs scored in 1st season by RCB (bonus: also include the extra runs using the extra runs table)

**Ans:**

"There is no available data for the 1st season, so instead of the 1st season, we are assuming that the 6th season is considered the 1st season."

This SQL query calculates the **total runs scored by RCB (Royal Challengers Bangalore) in Season 6** of the IPL by summing up:

1. **Runs\_Scored**: Runs scored by batsmen, using the ball\_by\_ball table.
2. **Extra\_Runs**: Additional runs (e.g., wides, no-balls) from the extra\_runs table.

A screenshot of a computer code

Description automatically generated

**Key Points:**

1. **SUM(COALESCE(..., 0))**: Ensures that null values are treated as 0 during the summation.
2. **Joins:**
   * **matches and ball\_by\_ball**: Links match details with ball-by-ball data.
   * **extra\_runs**: Links scoring details using match, over, ball, and innings numbers.
   * **team**: Identifies the batting team by its ID.
3. **Filters:**
   * **t.Team\_Id = 2** : Restricts the data to Royal Challengers Bangalore (RCB).
   * **m.Season\_Id = 6**: Limits the analysis to Season 6.
4. The query sums up all the runs scored by Royal Challengers Bangalore (RCB) in the first season, including both runs scored by the batsmen and any extra runs (such as wides, no-balls, leg-byes, etc.).
5. The extra runs are included in the calculation via the extra\_runs table, which is joined to the ball\_by\_ball table. This accounts for runs not directly credited to the but added to the team's total, such as no-balls, wides, and byes.

**Result Output:**



1. How many players were more than the age of 25 during season 2014?

**ANS:**

**A screenshot of a computer code

Description automatically generated**

**Explanation:**

1. **COUNT(DISTINCT pm.Player\_Id)**: Counts the unique players from the player\_match table.
2. **JOINs:**

* **matches:** Links players to their respective matches.
* **player:** Retrieves player details, including their date of birth (DOB).

1. **Age Calculation:**

* **TIMESTAMPDIFF(YEAR, p.DOB, m.Match\_Date) > 25:** Calculates the player's age on the match date and filters out players older than 25.

1. **Filter:**

* **m.Match \_Date LIKE ‘2014%’ :** Limits the data to matches from 2014.

**Result Output:**

****

1. How many matches did RCB win in 2013?

**Ans:**

A screenshot of a computer screen

Description automatically generated

**Explanation:**

This SQL query counts the number of matches won by the Royal Challengers Bangalore (RCB) in 2013.

1. **Selecting Win Count:** The outer query selects a count of all rows that meet the specified criteria, labelling the result as **RCB\_Wins**.
2. **Joining Tables:**
   * matches (m): This table records the match details, including the winner of each match.
   * team (t): This table contains team information, which is joined to the matches table to identify the winning team.
3. **Filtering Conditions:**
   * The where clause specifies that the query should only consider matches in 2013 (**m. Match\_Date like '2013%'**).
   * It further filters the results to include only those matches where the winning team is the Royal Challengers Bangalore (**t.team\_name='Royal Challengers Bangalore'**).

The final output provides the total count of matches won by RCB in 2013, helping analyse their performance during that season.

**Result Output:**

****

1. List the top 10 players according to their strike rate in the last 4 seasons

**Ans:**

To list the top 10 players by strike rate in the last 4 seasons, we will need the following tables:

A screenshot of a computer screen

Description automatically generated

**Explanation:**

* **Tables Required:**
* **player**: Provides the player's name (Player\_Name) and Player\_Id to identify each player uniquely.
* **matches**: Used to filter the matches that occurred in the last 4 seasons.
* **season**: Helps identify the last 4 seasons using Season\_Id.
* **ball\_by\_ball**: Provides the details of each ball faced by the player, allowing us to calculate the total balls faced. And the runs scored by the player on each ball, allowing us to calculate the total runs.
* **Query Approach:**
* We find the total runs and balls faced by each player.
* Then, calculate the strike rate.
* Finally, we list the top 10 based on this rate.
* **Recommendations:**
* Teams should utilize these high-strike-rate players in critical overs, especially in the death overs of T20 matches, to maximize the team's total score.
* Strategic planning should focus on rotating these players in middle-to-lower batting orders to exploit their ability to score quickly in high-pressure situations.

**Result Output:**

A screenshot of a table

Description automatically generated

A graph of blue rectangular bars

Description automatically generated with medium confidence

1. What are the average runs scored by each batsman considering all the seasons?

**Ans:**

**A screenshot of a computer program

Description automatically generated**

**Explanation:**

1. **Main Tables Involved**:
   * ball\_by\_ball (bb): Tracks events for each ball, linking it to players (Striker, Match\_Id, Runs\_Scored, etc.).
   * player (p): Contains player details (Player\_Name, Player\_Id).
2. **Joins**:
   * ball\_by\_ball (bb) is joined with the player (p):
     + Fetches the name of the player (Player\_Name) associated with the Striker (batsman on strike).
3. **Columns Selected**:
   * p.Player\_Name: The name of the player.
   * SUM(bb.Runs\_Scored) AS Total\_Runs: Total runs scored by each player.
   * COUNT(DISTINCT CONCAT(bb.Match\_Id, bb.Innings\_No)) AS Innings\_Played:
     + Calculates the number of distinct innings played by combining Match\_Id and Innings\_No.
   * ROUND(SUM(bb.Runs\_Scored) / COUNT(DISTINCT CONCAT(bb.Match\_Id, bb.Innings\_No)), 2) AS Average\_Runs:
     + Batting average = Total runs / Number of innings played.
     + The result is rounded to 2 decimal places.
4. **Grouping**:
   * The data is grouped by p.Player\_Name, ensuring calculations are done for each player.
5. **Ordering**:
   * Results are sorted in descending order of Average\_Runs to highlight players with the best batting averages.

**Final Summary:**

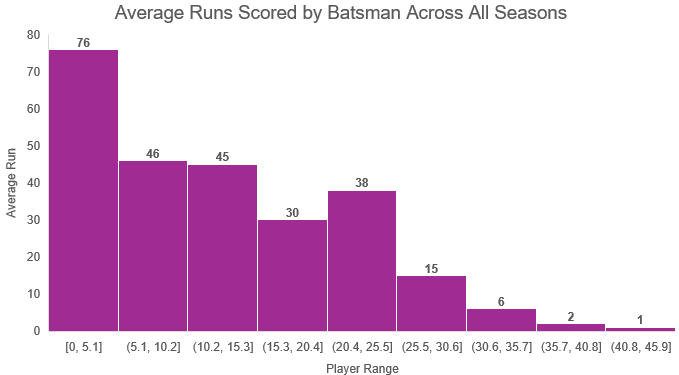
This query calculates and ranks players based on their batting averages across all matches. It provides the following details for each player:

* Player Name,
* Total Runs Scored,
* Innings Played,
* Batting Average (Average Runs per Innings).

**Result Output:**

A screenshot of a table

Description automatically generated



1. What are the average wickets taken by each bowler considering all the seasons?

**Ans:**

**A screen shot of a computer

Description automatically generated**

**Explanation:**

1. **Tables Involved**:
   * **ball\_by\_ball (bb)**: Contains ball-by-ball data, runs scored on each ball, including Bowler and Match\_Id.
   * **player (p)**: Stores player details like Player\_Name and Player\_Id.
2. **Joins**:
   * **bb joined with p**: Links the Bowler in bb to their name in the player table.
3. **Filters**: bb.Runs\_Scored = 0: Assumes that when Runs\_Scored is 0, it indicates the batsman was out, i.e., a wicket.
4. **Aggregations**:
   * **COUNT(bb.Bowler)**: Counts the total number of wickets taken by each bowler.
   * **COUNT(DISTINCT bb.Match\_Id)**: Counts the number of unique matches the bowler participated in.
   * **COUNT(bb.Bowler) / NULLIF(COUNT(DISTINCT bb.Match\_Id), 0)**:
     + Calculates the **average wickets per match**.
     + NULLIF prevents division by zero by returning NULL if the denominator is 0.
5. **Grouping**:
   * Results are grouped by p.Player\_Name to calculate stats for each bowler.
6. **Ordering**:
   * Bowlers are ordered by Average\_Wickets in descending order to highlight the most effective bowlers.

**Final Summary:**

This query retrieves bowlers’ performance metrics:

* **Player\_Name**: Name of the bowler.
* **Wickets**: Total wickets taken.
* **Average\_Wickets**: Average number of wickets per match.

It filters out instances where no wickets were taken (bb.Runs\_Scored = 0) and ranks bowlers by their **average wickets per match**, showing the most impactful bowlers at the top.

**Result Output:**

A screenshot of a data

Description automatically generated

A graph of a number of cricket players

Description automatically generated with medium confidence

1. List all the players who have average runs scored greater than the overall average and who have taken wickets greater than the overall average

**Ans:**

To list all the players who have average runs scored greater than the overall average and who have taken wickets greater than the overall average, we will need the following tables:

* **Tables Required:**
* **player:** Provides player names and IDs.
* **ball\_by\_ball:** Links each ball to the respective players (batsman and bowler) and also contains information about the runs scored by each batsman.
* **wicket\_taken:** Contains information about the wickets taken by each bowler.

**A screenshot of a computer program

Description automatically generated**

* **Query Approach:**
* **Calculate Overall Averages**: Determine the overall average runs scored and wickets taken across all players.
* **Compare Player Averages:** List players who exceed both the overall average runs and overall average wickets.
* **Insights:**
* Players like AB de Villiers, AD Russell, stand out as having a balanced impact both with the bat and ball, consistently exceeding overall averages in runs and wickets.
* All-rounders such as JP Faulkner, and GJ Maxwell are highly versatile, contributing significantly in both scoring runs and taking wickets.
* Bowlers who also contribute significantly with the bat, like YK Pathan, provide great balance and flexibility to their teams.
* **Recommendations:**
* Teams should prioritize players who consistently outperform the overall average in both batting and bowling, as they can offer strategic depth and versatility in varying match situations.
* Focus on nurturing and retaining all-rounders who can balance both batting and bowling to maximize team performance across different match conditions.

**Result Output:**

**A screenshot of a table

Description automatically generated**

**A graph of blue bars

Description automatically generated with medium confidence**

1. Create a table rcb\_record table that shows the wins and losses of RCB in an individual venue.

**Ans:**

**A screenshot of a computer code

Description automatically generated**

**Explanation:**

* **Create the Table**:
* A table named rcb\_record is created with three columns: venue\_name, wins, and losses.
* **Insert Data**:
* The table is filled with data showing how many matches the Royal Challengers Bangalore (RCB) has won and lost at each venue.
* It counts wins and losses by checking the match results where RCB is involved.

**Summary:**

The SQL script creates a table called rcb\_record to show the Royal Challengers Bangalore's (RCB) wins and losses at different venues. It counts how many matches RCB won and lost at each venue and stores this information in the table. Finally, it retrieves and displays all the records from the rcb\_record table.

**Result Output:**

A screenshot of a sports schedule

Description automatically generated

**A pie chart with numbers and text

Description automatically generated**

1. What is the impact of bowling style on wickets taken?

**Ans:**

To Analyze the impact of bowling style on wickets taken, the following tables are required:

* **player**: To get information about the players, including their bowling style.
* **bowling\_style**: To determine the type of bowling style associated with each player.
* **wicket\_taken**: To get the details about the wickets taken by each bowler.
* **ball\_by\_ball**: To link the wickets taken with the bowler in each ball.

A screenshot of a computer code

Description automatically generated

**Query Approach:**

* Identify each bowler's style.
* Count the total wickets taken by bowlers of each style.
* Compare the wicket counts to assess the impact of different bowling styles.

**Insights:**

1. Right-arm medium bowlers are the most successful, taking 1173 wickets, which indicates that this bowling style is more effective in multiple match conditions. RCB could leverage this by strengthening their right-arm medium pace options.
2. Right-arm fast-medium and off break bowlers are also highly effective, with 735 and 642 wickets respectively, suggesting that bowlers who combine pace with movement or spin create difficulties for batsmen. These styles might be valuable in critical overs.
3. Slow left-arm orthodox bowlers have taken 567 wickets, indicating that spin options continue to play a vital role, particularly on slower pitches. RCB can benefit from having versatile spin bowlers to tackle subcontinent conditions.
4. Leg break and googly bowlers have fewer wickets, but their value lies in specific match-ups where they can exploit batting weaknesses, particularly in the middle overs.

**Recommendations:**

1. RCB should consider a **balanced bowling attack** with emphasis on right-arm medium and fast-medium bowlers for versatility across different match conditions.
2. Developing and including **quality spin bowlers**, especially left-arm orthodox spinners, can help gain an edge in crucial overs and in home conditions where spin is often more effective.

**Result Output:**

A screenshot of a computer

Description automatically generated

A graph of purple bars

Description automatically generated

1. Write the SQL query to provide a status of whether the performance of the team is better than the previous year's performance on the basis of the number of runs scored by the team in the season and the number of wickets taken

**Ans:**

To provide a status of whether the performance of the team better than the previous year performance on the basis of number of runs scored by the team in season and number of wickets taken the following tables are required:

A screenshot of a computer program

Description automatically generated

* **Tables Required:**
* **team**: Contains information about teams, including Team\_Id and Team\_Name.
* **player\_match**: Links players to matches and teams, allowing us to associate players with specific teams in specific matches.
* **matches**: Holds details about the matches, including Match\_Id, Team\_1, Team\_2, and Season\_Id.
* **ball\_by\_ball**: Records the runs scored by batsmen in each match, identified by Match\_Id.
* **wicket\_taken**: Records the wickets taken in each match, identified by Match\_Id.
* **season**: Contains information about each season, including Season\_Id and Season\_Year.
* **Query Approach:**
* **Calculate Team Performance per Season:**
* The query first calculates the total runs scored (TotalRuns) and total wickets taken (TotalWickets) by each team in each season.
* This is done by joining the team, player\_match, and matches tables to get the team’s matches, then summing the runs from ball\_by\_ball and counting the wickets from wicket\_taken.
* **Compare Current Year to Previous Year:**
* The query then performs a self-join on the TeamPerformance CTE (Common Table Expression) to compare the performance of a team between two consecutive seasons (current year vs. previous year).
* The Performance\_Status is calculated based on the comparison of runs and wickets between the current and previous seasons:
* **Better:** If both runs scored and wickets taken in the current year are greater than the previous year.
* **Same:** If both runs scored and wickets taken in the current year are equal to the previous year.
* **Mixed:** If one of the metrics (runs or wickets) is better, but the other is the same as the previous year.
* Worse: If both runs scored and wickets taken in the current year are less than the previous year.
* **Final Output**:
* The query selects the team name, the previous season's year, the current season's year, and the corresponding runs and wickets. It also provides the performance status based on the conditions described above.

**Recommendations:**

* **Focus on Teams Showing Stagnation or Decline**: Teams like Delhi Daredevils and Rajasthan Royals, which show minimal improvement or even decline, should reassess their strategy, player development, or tactics to reverse these trends.
* **Leverage Growth for RCB and Other Improving Teams**: Teams showing growth in both metrics, like RCB, can build on this momentum to further strengthen their performance and possibly invest in areas like bowling or all-rounders to continue their upward trajectory.

**Result Output:**

**A table with numbers and lines

Description automatically generated**

1. Can you derive more KPIs for the team strategy?

**Ans:**

**Potential KPIs for Team Strategy:**

1. Top Order Contribution
2. Boundary Frequency
3. Powerplay Performance
4. Death Over Efficiency
5. Win/Loss Ratio by Venue

**Let’s Deep dive one-by-one to get more understanding:**

1. **Top Order Contribution**

**A screenshot of a computer program

Description automatically generated**

* **Objective:**
  + To get the average percentage of a team's total runs that were scored by their top three batsmen across all matches.
  + It gives an insight into how much the top three batsmen contribute to the team's overall performance.
* **Tables Required:**
  + **matches**: To identify each match and link it to the batting performance.
  + **ball\_by\_ball**: To capture the batting order of the players and runs scored.
  + **team**: To associate the performance with a specific team.
* **Approach:**
* **Identify Matches & Teams:** Use the matches and team tables to get match details and team information.
* **Filter Top-Order Batsmen:** Use ball\_by\_ball to focus on players in the top three batting positions.
* **Calculate Runs:** Sum up the runs scored by these top-order batsmen using the ball\_by\_ball table.
* **Aggregate Data:** Calculate the percentage of total team runs that came from the top three batsmen for each match.
* **Compute Averages:** Finally, average these percentages across all matches for each team.

**Result Output:**

A screenshot of a computer

Description automatically generated

A graph of a number of green bars

Description automatically generated with medium confidence

1. **Boundary Frequency**

A computer code with text

Description automatically generated with medium confidence

* **Objective:** Determine how often the team hits boundaries (fours and sixes) per 100 balls, helping to assess the team's attacking play style.
* **Tables Required:**
* **player**: Contains player information including Player\_Id and Player\_Name.
* **ball\_by\_ball**: Records the details of every ball bowled, including Striker (the player facing the ball) and Team\_Batting,runs scored on each ball, including whether the ball resulted in a boundary (4 or 6 runs).
* **Approach:**
* **Count Boundaries:** Calculate the total number of boundaries (fours and sixes) scored by each player.
* **Total Balls Faced:** Count the total number of balls each player faced.
* **Boundary Frequency Calculation:** Determine the boundary frequency by dividing the number of boundaries by the total balls faced, then multiply by 100.
* **Ranking:** Rank the players based on their boundary frequency and select the top 10.

**Result Output:**

**A screenshot of a computer

Description automatically generated A circular chart with different colored arrows

Description automatically generated**

1. **Powerplay Performance**

**A screen shot of a computer

Description automatically generated**

* **Objective:**
* Teams with a high Run-to-Wicket ratio are not only scoring well but also managing to preserve their wickets, which is a sign of a well-rounded batting strategy.
* Teams with a low Run-to-Wicket ratio might need to reconsider their approach in the Powerplay, either by shuffling the batting order or by adopting a more conservative strategy initially.
* **Tables Required:**

1. **matches**: Links the match details.
2. **ball\_by\_ball**: Retrieves balls bowled during the Powerplay overs (1-6).
3. **wicket\_taken**: Identifies wickets lost during the Powerplay.
4. **team**: Links to identify the teams by their IDs.

* **Approach:**
* **Calculate Powerplay Performance:**
  + First, calculate the total runs scored and wickets lost during the Powerplay (first 6 overs) for each match and team.
* **Aggregate Performance:**
  + Then, calculate the average Powerplay runs and wickets lost per team across all matches.
* **Compare Teams:**

Finally, compare the teams based on their average performance in the Powerplay, focusing on the balance between runs scored and wickets lost.

**Result Output:**

**A screenshot of a graph

Description automatically generated**

**A graph of purple bars

Description automatically generated with medium confidence**

1. **Death Over Efficiency**

**A screenshot of a computer program

Description automatically generated**

* **Objective:** Top 10 players based on the total runs scored and wickets taken in the last 4 overs of matches.
* **Derived Insights:**
  + **Key Finishers**: We can identifiy the top 10 players who excel at scoring runs during the last 4 overs, showcasing the team's most reliable finishers in pressure situations.
  + **Clutch Bowlers**: Also highlights players who consistently take wickets during the death overs, indicating who the team can rely on to turn the tide in critical match moments.
  + **Team Contribution**: By linking player performance to their teams, you can see which teams have the strongest death-over performers, providing insights into overall team strategies and strengths in the final overs.
* **Tables Required:**
* **matches**: Links to the match details.
* **ball\_by\_ball**: Used to identify balls bowled, runs scored in the death overs.
* **extra\_runs**: Includes extra runs given during the death overs.
* **wicket\_taken**: Identifies wickets taken in the death overs.
* **player**: Links player names to their performances.
* **team**: Identifies the teams to which the players belong.
* **Approach:**
  + **Aggregate Runs and Wickets**: Sum the runs scored and count the wickets taken by each player during the last 4 overs.

**Rank and Filter**: Rank players based on their death-over performance and filter to get the top

**Result Output:**

**A screenshot of a sports report

Description automatically generated**

**A graph with numbers and a number of people

Description automatically generated**

1. **Win/Loss Ratio by Venue**

**A computer screen shot of a computer code

Description automatically generated**

* **Derived Insights:**
  + Teams with the highest number of wins and the lowest number of losses at specific venues, showing the venues where teams perform exceptionally well.
  + Certain teams consistently perform better at specific venues, indicating a potential home-ground or venue-specific advantage that can be crucial for strategic planning.
* **Tables Required:**
* **matches:** To link the match details, including which teams played and who won.
* **team:** To retrieve team names and their corresponding IDs.
* **venue:** To get the names and details of the venues where matches were played.
* **Approach:**
* Calculate the number of wins and losses for each team at each venue.
* Determine the win/loss ratio based on these counts.
* Rank teams by their performance at different venues, prioritizing those with the most wins.

**Result Output:**

**A screenshot of a sports report

Description automatically generated**

**A graph of a match

Description automatically generated with medium confidence**

1. Using SQL, write a query to find out the average wickets taken by each bowler in each venue. Also, rank the gender according to the average value.

**Ans:**

**A screen shot of a computer

Description automatically generated**

**Explanation of the Query**

**1. Common Table Expression (CTE): each\_venue\_BowlerWickets**

* **Purpose**:
  + The CTE is used to gather statistics on each bowler's performance at different venues. It calculates the total wickets taken and the number of matches played by each bowler at each venue.
* **Tables Involved**:
  + **player (p)**: Contains player details (such as Player\_Name).
  + **player\_match (pm)**: Links players to the matches they have played.
  + **matches (m)**: Contains details about the matches, including the venue.
  + **ball\_by\_ball (bb)**: Contains the ball-by-ball data for each match.
  + **wicket\_taken (wt)**: Contains information on the wickets taken, linking it to the ball-by-ball data.
  + **venue (v)**: Contains details about the venues where matches were held.
* **Logic**:
  + **COUNT(DISTINCT bb.Match\_Id, bb.Over\_Id, bb.Ball\_Id, bb.Innings\_No)**: Counts the number of wickets taken (distinct combinations of match, over, ball, and innings).
  + **COUNT(DISTINCT bb.Match\_Id, bb.Innings\_No)**: Counts the number of matches played by the bowler at each venue.

The data is grouped by the bowler, player name, and venue.

**2. Main Query**

* **SELECT**:
  + **Bowler AS Player\_Id**: The bowler's ID (represented by the Bowler column in the ball-by-ball data).
  + **Player\_Name**: The name of the player.
  + **Venue\_Name**: The name of the venue where the bowler played.
  + **ROUND(Total\_Wickets / Matches\_Played, 1) AS Average\_Wickets**: The average number of wickets taken per match by the bowler at each venue, rounded to one decimal place.
  + **DENSE\_RANK() OVER (ORDER BY ROUND(Total\_Wickets / Matches\_Played, 1) DESC)**: This assigns a rank to each bowler based on their average wickets per match, ordered in descending order of their performance. DENSE\_RANK() ensures that no ranks are skipped (i.e., if two bowlers have the same average, they will have the same rank, and the next rank will be the next number).
* **ORDER BY**:
  + The results are sorted by Average\_Wickets in descending order, and in case of ties, the Rank column is used for secondary sorting in descending order.

**Result Output:**

A screenshot of a sports event

Description automatically generated

1. Which of the given players have consistently performed well in past seasons? (will you use any visualization to solve the problem)

**Ans:**

**A screenshot of a computer program

Description automatically generated**Players like **DA Warner, V. Kohli** and **G. Gambhir** consistently improved their performance over the past seasons. In contrast, players such as **RG Sharma**, **SR Watson** had fluctuating performances with periods of high runs and declines.

**Analysis of Each Player's Performance**

1. **V. Kohli**
   * The total runs show a generally increasing trend from 2013 to 2016, with significant peaks in 2013 and 2016.
   * Kohli's performance consistently improved, especially with a notable peak in 2016.
2. **G. Gambhir**
   * Gambhir’s performance shows fluctuations but mostly an upward trend from 2013 to 2016. He had a peak in 2014, followed by a slight dip and then an increase in 2016.
   * Overall, Gambhir had consistent performances, with some variations but a positive trend across seasons.
3. **SR Watson**
   * SR Watson had a strong peak in 2014, but performances were not consistently increasing. There were significant drops in 2015.
   * Not a consistent upward trend; his performance varied across seasons.
4. **RG Sharma**
   * Sharma performance fluctuated significantly, with a peak in 2015. However, there was a steep decline by 2016.
   * His performance was not consistently increasing; it had high peaks and notable declines.
5. **DA Warner**
   * Warner showed a consistent increase from 2013 to 2016, peaking in 2016. There was a slight drop in 2015.
   * His performance improved and significant growth from 2013.

**Visualization in Excel**

To better understand and present these trends, you can use the following chart types:

1. **Column Chart**:
   * We can use column charts to compare the total runs for each player by season.
   * This can highlight the peaks and drops for different players in the same visual.

These visualizations can effectively communicate which players have had consistent performance improvements and who had fluctuating results over the seasons.

**Result Output:**

**A screenshot of a table

Description automatically generated**

A blue rectangle with black text

Description automatically generated

A graph with blue rectangles and numbers

Description automatically generated

A graph with blue rectangles

Description automatically generated

A graph with numbers and a bar

Description automatically generated

A graph with numbers and a bar

Description automatically generated

1. Are there players whose performance is more suited to specific venues or conditions? (how would you present this using charts?)

**Ans:**

Based on the below 2 points analysed this question.

1. **The top 3 players in each venue scored the highest runs**

A screenshot of a computer code

Description automatically generated

**Explanation:**

* This SQL query identifies the top three players who scored the highest runs at each venue in India. Here's a step-by-step breakdown of the process:

1. **Calculate Total Runs by Player at Each Venue:**

The result CTE aggregates the total runs scored by each player at each venue. It joins multiple tables, including player, player\_match, ball\_by\_ball to link players with their performances in matches played at specific venues. The query filters for matches held in India and groups the results by player and venue.

1. **Rank Players by Runs Scored:**

The rank\_player CTE ranks the players within each venue based on their total runs using the DENSE\_RANK() function. This creates a ranking for the players at each venue, allowing for ties to be handled appropriately.

1. **Select Top 3 Players:**

The final query retrieves players who rank in the top three for runs scored at each venue by filtering the results where top\_rank is between 1 and 3.

1. **Order the Results:**

The output is ordered by venue name and the ranking of the players, ensuring a clear presentation of the top scorers at each venue.

**Result Output:**

**A screenshot of a sports stadium

Description automatically generated**

**A graph of a player

Description automatically generated with medium confidence**

1. **The top 3 players in each venue took the highest wicket**

**A screenshot of a computer

Description automatically generated**

**Explanation:**

This SQL query retrieves the top three wicket-takers at each cricket venue in India. Here’s a breakdown of how it works:

1. **Count Wickets Taken:**

* The result Common Table Expression (CTE) counts the distinct wickets taken by each player at various venues. It joins several tables, including player, player\_match, ball\_by\_ball, wicket\_taken, matches, venue, season, city, and country.
* The query filters for matches played in India and groups the results by player and venue, counting the distinct combinations of match ID, innings number, over ID, and ball ID to accurately reflect the number of wickets taken.

1. **Ranking Players:**

* The rank\_player CTE assigns a rank to each player based on their total wickets at each venue using the DENSE\_RANK() function. This function creates a ranking within each venue based on the total wickets, allowing for ties.

1. **Selecting Top 3 Wicket-Takers:**

* The final selection retrieves players ranked in the top three for each venue, ordering the results first by rank and then by total wickets taken in descending order

**Result Output:**

**A screenshot of a computer

Description automatically generated**

**A graph of a number of people

Description automatically generated**

Subjective Questions

1. How does the toss decision affect the result of the match? (which visualizations could be used to present your answer better) And is the impact limited to only specific venues?

**Ans:**

Key points

1. **Toss win match win and toss win match loss of all teams.**

A screenshot of a computer screen

Description automatically generated

A computer screen shot

Description automatically generated

The query is broken down into multiple steps using **Common Table Expressions (CTEs)**. Each CTE handles a different part of the process, making the query easier to read and understand.

**Counting Toss Winner Wins**:

* The first part of the query focuses on determining how many times the team that won the toss also went on to win the match.
* This is done separately for two teams (team 1 and team 2) for each match.
* The results are combined and summed up to get the total number of toss winners who also won the match.

**Counting Toss Winner Losses**:

* The second part calculates how many times the team that won the toss ended up losing the match.
* Similarly, this is done separately for both teams (team 1 and team 2) for each match.
* The results are combined and summed up to get the total number of toss winners who lost the match.

**Combining the Results**:

* After calculating the wins and losses, the query combines both results using a join.
* It links the counts of toss winners winning and losing using a common identifier (number), which is always set to 1.

**Result Output:**

The final output of the query will be a single row with two key pieces of information:

* The **total count of toss winners who won the match**.
* The **total count of toss winners who lost the match**.



A graph of a match

Description automatically generated

1. **Toss win and match win percentage of each team.**

A screenshot of a computer program

Description automatically generated

**Explanation:**

**match\_winner\_count CTE**:

* This Common Table Expression (CTE) calculates how many matches a team won when they also won the toss.
* It groups data by toss\_winner and counts matches where toss\_winner = match\_winner.
* The result includes toss\_winner and match\_win\_count.

**toss\_winner\_count CTE**:

* This CTE calculates the total number of tosses won by each team.
* It groups data by toss\_winner and counts all matches won during the toss.

**Main Query**:

Combines data from the two CTEs and the team table:

* match\_winner\_count (mwc) is joined with toss\_winner\_count (twc) on the toss\_winner column.
* This ensures each team’s toss win count and toss win + match win count are available for analysis.
* Another join is made with the team table to fetch the team name (t.team\_name).

**Columns Selected**:

* t.team\_name: Name of the team.
* mwc.match\_win\_count: Number of matches a team won after winning the toss.
* twc.total\_toss\_win\_count: Total tosses won by the team.
* toss\_win\_match\_win\_percentage: The percentage of matches won after winning the toss, calculated as: toss\_win\_match\_win\_percentage =(match\_win\_count/total\_toss\_win\_count)×100

**Order**:

* Results are ordered by toss\_win\_match\_win\_percentage in descending order to highlight teams with the highest success rate.

**Result Output:**

A screenshot of a game

Description automatically generated

A colorful pie chart with white text

Description automatically generated

1. **Match win percentage after won toss in each venue**

**A screenshot of a computer program

Description automatically generated**

**Explanation:**

**CTE: Toss\_Win\_Stats:**

* Calculates toss-related statistics for each venue and toss decision (batting or fielding first).
* Joins the matches, toss\_decision, and venue tables to gather relevant data.
* Columns Calculated:

1. Venue\_Name: Name of the venue from the venue table.
2. Toss\_Decision: Toss decision (e.g., bat or field) from the toss\_decision table.
3. Total\_Matches: Total matches played at the venue with a specific toss decision, counted using COUNT(\*).
4. Matches\_Won\_After\_Toss: Matches where the toss winner also won the match.
5. Adds 1 for every match where Match\_Winner = Toss\_Winner, else adds 0.
6. Win\_Percentage: The percentage of matches won after winning the toss, calculated as: Win\_Percentage=(Matches\_Won\_After\_TossTotal\_Matches)×100\text{Win\\_Percentage} = \left(\frac{\text{Matches\\_Won\\_After\\_Toss}}{\text{Total\\_Matches}}\right) \times 100Win\_Percentage=(Total\_MatchesMatches\_Won\_After\_Toss​)×100
7. Data is grouped by Venue\_Name and Toss\_Name using GROUP BY.

**Main Query:**

* Selects the columns from the Toss\_Win\_Stats CTE.
* Filters the results to include only venues with at least 10 matches (WHERE Total\_Matches >= 10), ensuring sufficient data for meaningful analysis.

**Final Output:**

1. Venue\_Name: Venue name.
2. Toss\_Decision: Toss decision (e.g., bat or field first).
3. Total\_Matches: Total matches played with the toss decision at the venue.
4. Matches\_Won\_After\_Toss: Matches won by the toss-winning team.
5. Win\_Percentage: Percentage of matches won after winning the toss, rounded to the nearest decimal place**.**

**Result Output:**

**A screenshot of a table

Description automatically generated**

1. Suggest some of the players who would be best fit for the team.

**Ans:**

Considering the below points suggest the best players for the team

1. Top 5 score rank in each season--players and number of times they are in the top 5 in the last 3 seasons.
2. Top 10 players with the highest strike rate and minimum 1500 runs scored.
3. Top 10 bowlers taken highest wicket.
4. Top 10 bowlers with best economy, minimum 100 overs bowled.
5. **Top 5 score rank in each season--players and number of times they are in the top 5 in the last 3 seasons.**

A screenshot of a computer program

Description automatically generated

**Explanation:**

**player\_match\_runs CTE:**

* + Calculates the total runs scored by each batsman in each match.
  + ball\_by\_ball tables to get runs\_scored data.
  + Groups by match\_id and striker (batsman) to calculate total runs per player per match.

**result CTE:**

* + Calculates total runs scored by each player in each IPL season.
  + Joins player\_match\_runs with the matches table to get season details.
  + Uses DENSE\_RANK to rank players in each season based on their total runs (season\_runs\_rank).
  + Groups by player\_id and season\_year for aggregation.

**top\_run\_rank\_player CTE:**

* + Filters players ranked in the top 5 run-scorers (season\_runs\_rank BETWEEN 1 AND 5) for seasons between 2014 and 2016.
  + Orders the result by season\_year and season\_runs\_rank.

**Main Query:**

* + Joins top\_run\_rank\_player with the player table to get player names.
  + Counts how many times each player appeared in the top 5 run-scorers during the specified seasons (no\_of\_times\_in\_top5).
  + Groups by player\_id and player\_name.
  + Orders the players by their appearances in descending order, returning the top 10 players (LIMIT 10).

**Result Output:**

A screenshot of a computer

Description automatically generated

1. **Top 10 players with the highest strike rate and minimum 1500 runs scored.**

A screenshot of a computer program

Description automatically generated

**Explanation:**

* This query identifies the top 10 players with the highest strike rate who have scored at least 1500 runs. Here’s a brief explanation of each step:

**runs\_scored CTE:**

* Calculates the total runs scored by each player.
* Only includes players with more than 1500 total runs (HAVING (SUM(bs.runs\_scored)) >= 1500).
* Orders players by total\_runs in descending order.

**balls\_faced CTE:**

* Counts the total balls faced by each player (each runs\_scored entry).
* Groups by player and orders by total\_ball\_faced in descending order.

**Final Query:**

* + - * Joins runs\_scored and balls\_faced on player ID to obtain both total runs and balls faced.
      * Calculates each player’s strike rate as (total\_runs / total\_ball\_faced) \* 100.
      * Orders players by strike rate in descending order and limits the result to the top 10.

This final list shows the top 10 players with the highest strike rates among those who have scored over 1500 runs.

**Result Output:**

A table of numbers and names

Description automatically generated

1. **Top 10 bowlers taken highest wicket.**

A screen shot of a computer code

Description automatically generated

**Explanation:**

* This query retrieves the top 10 bowlers who have taken the highest number of wickets. Here's a brief breakdown:
* **Join Tables**:
  + Joins the player, player\_match, ball\_by\_ball, and wicket\_taken tables to gather data on bowlers and their wickets in each match.
* **Calculate Wickets**:
  + Counts the occurrences of player\_out in the wicket\_taken table for each bowler, representing the number of wickets taken by that bowler.
* **Group and Order**:
  + Group results by bowler and player\_name to get the total wickets for each bowler.
  + Orders by number\_of\_wickets in descending order and limits the result to the top 10 bowlers.
* This final output shows the 10 bowlers with the most wickets, ranked by the highest wicket count

**Result Output:**

A screenshot of a table

Description automatically generated

1. **Top 10 bowlers with best economy, minimum 100 overs bowled.**

A screenshot of a computer code

Description automatically generated

**Explanation:**

This query identifies the top 10 bowlers with the best economy rates (lowest runs conceded per over) who have bowled at least 100 overs.

**Steps:**

* Calculate Runs Conceded: Sum up total runs conceded by each bowler.
* Calculate Overs Bowled: Count unique overs bowled by each bowler.
* Compute Economy Rate: Divide runs conceded by overs bowled for each bowler with more than 100 overs.
* Filter and Sort: Select the top 10 bowlers with the lowest economy rates.

This provides the 10 most economical bowlers with a minimum threshold of overs bowled.

**Result Output:**

A screenshot of a computer

Description automatically generated

1. What are some of the parameters that should be focused on while selecting the players?

**Ans:**

When selecting players for an IPL team, focusing on the right parameters ensures that the team is well-balanced and competitive. Here are key parameters to consider:

**1. Player Performance Metrics:**

* **Batting Average**: Indicates a batsman's consistency and reliability. A higher average suggests that a batsman score runs consistently.
* **Strike Rate**: Shows how quickly a batsman scores runs, which is essential for the T20 format where quick scoring is crucial.
* **Total Runs/Wickets**: Helps identify top performers with a proven track record in scoring runs or taking wickets.
* **Bowling Economy Rate**: Measures a bowler’s effectiveness in restricting runs, particularly important in T20 matches.
* **Wicket-taking Ability***:* Indicates a bowler’s impact in breaking partnerships and taking key wickets.

1. **Consistency:**

* **Recent Form**: Current season or recent performances should be evaluated to understand the player's recent form.
* **Consistency Across Matches**: Use averages and median scores/wickets to identify players who perform consistently over time.

1. **Player Role and Team Balance:**

* **Role Classification**: Ensure a balanced mix of batsmen, bowlers, all-rounders, and a wicketkeeper.
* **Specialists vs. All-rounders**: Decide the proportion of specialist batsmen and bowlers versus all-rounders based on the team’s strategy.
* **Finisher Capability**: Identify batsmen who can score quickly and effectively in the final overs.

1. **Adaptability to Conditions:**

* **Performance at Specific Venues**: Review a player's record at various venues to see if they adapt well to different pitches (e.g., spin-friendly or pace-friendly).
* **Performance Against Top Teams**: Analyze how players perform under pressure against strong opposition.

1. **Skill Specialization:**

* **Batting Style**: Choose players with varying styles (e.g., left-handers, aggressive hitters, technical players) to handle different bowling attacks.
* **Bowling Type**: Have a mix of seamers and spinners, and consider variations like wrist spin or left-arm pace for diversity in attack.
* **Powerplay and Death Over Specialists**: Pick bowlers who excel at specific phases of the game.

1. **Fielding Ability:**

* **Fielding Skills**: Good fielders can save runs and create match-winning moments with catches and run-outs.
* **Wicketkeeping Performance**: If selecting a wicketkeeper, evaluate their dismissals, stumping speed, and catching reliability.

1. **Injury and Fitness Levels:**

* **Recent Injuries**: Review any recent injuries that could affect the player's availability and performance.
* **General Fitness**: Ensure that players can endure the physical demands of the tournament.

1. **Experience and Leadership:**

* **Seasoned Players**: Include experienced players who bring calm and strategic thinking during crucial moments.
* **Young Talent**: Balance the squad with emerging players who bring energy and are eager to prove themselves.
* **Captaincy Potential**: Consider players with leadership skills to assist or take over as captain if needed.

1. **Player Adaptability:**

* **Versatility**: Players who can adapt to different batting positions or bowling situations provide more strategic options.
* **All-Weather Performance**: Players who perform well in both day and night matches or in different climate conditions.

1. **Team Chemistry and Work Ethic:**

* **Compatibility**: Assess how well a player integrates with the team’s culture and dynamics.
* **Work Ethic and Attitude**: Choose players known for their strong work ethic and positive impact on team morale.

Focusing on these parameters ensures that the team is skilled, adaptable, cohesive, and strategically equipped for the challenges of the tournament.

1. Which players offer versatility in their skills and can contribute effectively with both bat and ball? (can you visualize the data for the same)

**Ans:**

**A screenshot of a computer program

Description automatically generated**

**Explanation:**

**Step 1: Identify Top Bowlers (CTE top\_bowler)**

* **Goal:** Find players who have taken the most wickets.
* **Key Steps:**
  + Join the player, player\_match, ball\_by\_ball, and wicket\_taken tables to track bowlers and their associated wickets.
  + Use COUNT(wt.player\_out) to count the number of wickets taken by each bowler.
  + Group by bb. bowler and p.player\_name to aggregate data per player.
  + Sort the results by number\_of\_wickets in descending order.

**Step 2: Identify Top Batsmen (CTE top\_batsman)**

* **Goal:** Find players who have scored the most runs.
* **Key Steps:**
  + Use SUM(bb.runs\_scored) to sum the runs scored by each batsman.
  + Group by bb. striker and p.player\_name to aggregate data per player.
  + Sort the results by total\_runs in descending order.

**Step 3: Combine Bowlers and Batsmen (Main Query)**

* **Goal:** Find players who excel as both bowlers and batsmen.
* **Key Steps:**
  + Join top\_bowler and top\_batsman on the player's ID (tb.bowler = tbs.striker).
  + Filter the results to include only players with more than or equals to 12 wickets and more than 500 runs (WHERE tb.number\_of\_wickets >=12 AND tbs.total\_runs > 500).
  + Select relevant fields like player\_id, player\_name, number\_of\_wickets, and total\_runs.

A screenshot of a table

Description automatically generated**Result Output:**

A graph of different colored bars

Description automatically generated

1. Are there players whose presence positively influences the morale and performance of the team? (justify your answer using visualization)

**Ans:**

This analysis aims to identify players who positively influence team morale and performance by consistently earning 'Man of the Match' awards. By examining their impact percentage, we highlight players who contribute significantly to their team's success

A screenshot of a computer code

Description automatically generated

* **Tables Required:**
* **player**: To get the player's name and ID.
* **matches**: To count the number of "Man of the Match" awards.
* **player\_match**: To calculate total matches played by each player.
* **Approach:**
* **Step 1**: Join player with matches to identify players who won "Man of the Match" awards.
* **Step 2**: Join with player\_match to calculate the total matches each player has played.
* **Step 3**: Calculate the impact percentage by dividing the number of MoM awards by total matches.
* **Step 4**: Order the results by impact percentage to highlight players who have a positive influence.
* **Derived Insights:**
* Players like DR Smith and MG Johnson frequently win "Man of the Match" awards, indicating their strong influence on team performance.
* Players with a higher Impact Percentage suggest they significantly boost team morale and success when performing well.
* Limiting to players with more than 5 MoM awards ensures that only consistently impactful players are considered.
* **Recommendations:**
* Consider players with a high Impact Percentage as key to boosting team morale during critical matches.
* Use this data for selecting captains or pivotal players in high-stakes games for maximum influence.

A table with numbers and letters

Description automatically generated**Result Output:**

A graph of different colored and blue bars

Description automatically generated with medium confidence

**Win Rates with Player Participation**:

* Analyse the win rates of teams when certain players are part of the lineup compared to when they are not. Higher win rates with their participation might indicate a positive influence.

A computer screen shot of a program

Description automatically generated

**Explanation:**

This SQL query calculates the win rate of players based on their presence in matches. Here’s a short breakdown of its components and functionality:

**Explanation of the Query**

**FROM and JOIN Clauses:**

* + The query starts from the player\_match table (pm), which connects players to specific matches.
  + It joins the matches table (m) to access match-related information, such as the match winner.
  + It also joins the player table (p) to get the player's name.

**Selecting Fields:**

* + **pm.Player\_Id and p.player\_name**: These columns identify each player.
  + **COUNT(DISTINCT m.Match\_Id) AS Matches\_Played:** Counts the total number of unique matches each player has participated in.
  + **SUM(CASE WHEN m.Match\_Winner = pm.Team\_Id THEN 1 ELSE 0 END) AS Wins:** Sums up the number of matches won by the player's team. If the player's team won, it counts as a win; otherwise, it counts as zero.

**Calculating Win Rate:**

* + The win rate is calculated as the total wins divided by the total matches played, multiplied by 100 to convert it to a percentage.

**GROUP BY Clause:**

* + Groups the results by player ID and name, allowing aggregation functions (like COUNT and SUM) to operate within each group.

**HAVING Clause**:

* + Filters the results to only include players who have participated in more than 80 matches.

**ORDER BY Clause:**

* + Orders the final results by win rate in descending order, so the players with the highest win rates appear first.

**Result Output:**

A screenshot of a table

Description automatically generated

A graph of a number of players

Description automatically generated

1. What would you suggest to RCB before going to the mega auction?

Ans:

Before the mega auction, Ill suggest Royal Challengers Bangalore (RCB) to consider the following strategies and recommendations.

* Players with the highest runs.
* Players with the best average.
* Players with the highest strike rate.
* Players with the most wickets taken.
* Players with the best economy in bowling.
* Players with the most success rate in the team winning including the highest man of the match award. Etc

By considering the above factors the below players will suggest to RCB before going to the mega auction.

**` 1. Players with the best average.**

A screenshot of a computer program

Description automatically generated

**Result Output:**

A screenshot of a data

Description automatically generated

1. **Players with the highest strike rate.**

A screenshot of a computer program

Description automatically generated

**Result Output:**

A screenshot of a data

Description automatically generated

1. **Players with the best economy in bowling.**

A screenshot of a computer program

Description automatically generated

**Result Output:**

A screenshot of a data

Description automatically generated

1. **Players with the most wickets taken.**

A computer screen shot of text

Description automatically generated

**Result Output:**

A screenshot of a computer

Description automatically generated

1. **Players with the most success rate in the team winning including the highest man of the match award.etc**

A screenshot of a computer program

Description automatically generated

**Result Output:**

A screenshot of a computer

Description automatically generated

* **Strengthen the Middle Order**: Secure a reliable middle-order batsman to add depth and stability.
* **All-rounders**: Target versatile players who can contribute with both bat and ball for better team balance.
* **Death-over Specialists**: Prioritize bowlers skilled at bowling Yorkers and variations in the death overs.
* **Spin Reinforcement**: Acquire quality spinners to control the middle overs on spin-friendly pitches.
* **Invest in Emerging Talent**: Focus on young, uncapped players with potential to ensure long-term team stability.

1. What do you think could be the factors contributing to the high-scoring matches and the impact on viewership and team strategies

**Ans:**

High-scoring matches in the IPL and other T20 formats have several contributing factors. These factors not only affect the on-field play but also have a significant impact on viewership and team strategies.

**Factors Contributing to High-Scoring Matches:**

1. **Batting-Friendly Pitches**: Many T20 venues, especially in India, prepare flat and hard pitches that are conducive to batting. These pitches offer little assistance to bowlers, especially pacers, and allow batsmen to play through the line and score quickly. This results in frequent high-scoring matches.
2. **Shorter Boundaries:** The dimensions of some venues have relatively shorter boundaries, making it easier for batsmen to hit sixes and fours. With smaller playing areas, even mistimed shots often clear the ropes, leading to higher run totals.
3. **Powerplay Overs and Free Hit Rule:** The fielding restrictions during the first six overs allow batsmen to take aggressive shots with minimal risk. Additionally, the free hit rule for no-balls gives batsmen more opportunities to swing freely without fear of dismissal, adding to the run rate.
4. **Advancement in Batting Skills and Technology:** Modern-day cricketers are more specialized in power-hitting techniques, and many invest in developing 360-degree shots (like reverse sweeps and scoops) that challenge traditional bowling. Additionally, advancements in bat technology have led to lighter, more powerful bats, making it easier to score quickly.

**Impact on Viewership:**

1. **Increased Entertainment Value:** High-scoring matches tend to attract more viewership due to the fast-paced action and constant boundary-hitting. Fans love seeing big scores and dramatic chases, which keeps them glued to their screens until the very end. This surge in excitement translates into higher television ratings and more engagement on social media.
2. **Global Appeal:** T20 cricket, particularly in the IPL, has drawn an international audience. The thrill of watching big totals and exciting finishes appeals to a broader demographic, not just traditional cricket fans. High-scoring matches, therefore, help cricket reach new markets and fan bases globally.

**Impact on Team Strategies:**

1. **Aggressive Batting Approach:** Teams are increasingly adopting an aggressive approach to batting, aiming for higher scores rather than playing conservatively. This has shifted the focus towards power hitters who can accelerate the scoring rate, even under pressure.
2. **Specialization in Death-Bowling:** High-scoring matches have made it essential for teams to focus on death-over specialists—bowlers who can keep the run rate in check in the final overs. Variations like slower balls, Yorkers, and wide-angle deliveries are now key to surviving the power-hitting barrage.
3. **Comparison between Runs Scored during Power Play (1 to 6 overs) & Death Overs (17 to 20 overs) and Runs Scored during Middle Overs (7 to 16 Overs):**

**A screenshot of a computer code

Description automatically generated**

**Result Output:**

**A screenshot of a computer

Description automatically generated**

**A graph of numbers and a number of green bars

Description automatically generated with medium confidence**

1. **Analyzing High-Scoring IPL Matches: Venue Performance and Contributing Factors.**

A screenshot of a computer code

Description automatically generated

**Result Output:**

**A screenshot of a data

Description automatically generated**

**A graph of different colored bars

Description automatically generated**

**A graph with numbers and text

Description automatically generated**

1. Analyze the impact of home-ground advantage on team performance and identify strategies to maximize this advantage for RCB.

**Ans:**

**A screenshot of a computer program

Description automatically generated**

Key points on home ground,

The concept of home-ground advantage in sports has a significant impact on team performance. Analysing this effect involves understanding how familiar conditions, fan support, and the psychological comfort of playing at home contribute to a team’s success. Here’s a breakdown of the impact and key factors involved in maximizing this advantage:

* **Pitch and Weather Conditions**: Teams are familiar with their home ground’s pitch behaviour, such as whether it Favors batting, spin, or pace.
* **Crowd Influence**: Home crowd support boosts player morale and can pressure the visiting team.
* **Win Rate**: Teams often have higher win rates at home compared to away matches. By analysing historical win rates, teams can evaluate the extent of their advantage when playing on their home turf.
* **Batting and Bowling Averages**: Batsmen and bowlers often perform differently at home versus away due to familiarity with pitch conditions. Analysing these averages can reveal whether players consistently perform better at home.
* **Fielding Metrics:** Teams tend to make fewer fielding errors at home due to familiarity with ground dimensions, sightlines, and lighting, which can reduce the likelihood of dropped catches or mis fielding incidents.

**Result Output:**

**A screenshot of a sports report

Description automatically generated**

* Total played matches for every team are more in home ground compared to other venue.

It will increase the chance of winning the team.

* RCB does have not a great winning percentage in the home ground with 55.17%.
* **High-Scoring Games**: The stadium’s batting-friendly conditions lead to high scores, often requiring RCB to maintain a powerful batting lineup.

**Conclusion:**

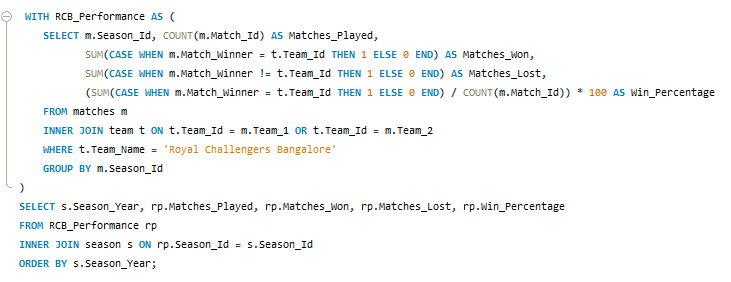
* The home-ground advantage in cricket has a clear impact on team performance, often providing a boost through familiarity, fan support, and reduced fatigue. By strategically selecting condition-specific players, enhancing fan engagement, and preparing through data-driven analysis, teams can maximize their home-ground advantage. However, they must manage expectations and avoid complacency to ensure consistent performance.

1. Come up with a visual and analytical analysis of the RCB's past season's performance and potential reasons for them not winning a trophy.

**Ans:**

**RCB Past Performance Analysis:**

The following analysis examines Royal Challengers Bangalore’s performance across IPL seasons, focusing on their match outcomes, win percentage, and key factors that may have impacted their ability to win a trophy. Below are the tables required for the analysis.

****

**Required Tables**

* **matches:** For match details like winner, season, and teams.
* **team:** To identify RCB in matches.
* **season:** For season-wise performance data (year).
* **Approach**
* **Filter RCB Matches:** Extract all matches involving RCB.
* **Calculate Wins and Losses:** For each season, count matches played, won, and lost.
* **Win Percentage Calculation:** Calculate win percentage by season.
* **Season Mapping:** Join the performance data with the season table to display season-wise results.
* **Derived Insights**
* **Inconsistent Performance:** RCB’s performance fluctuates, with strong seasons followed by weaker ones.
* **Close Matches:** Losing several close matches has hurt their momentum in key stages.
* **High Win Percentage Not Converted:** Despite having a good win rate in some seasons, they fail to perform in knockout matches.
* **Potential Reasons for Not Winning a Trophy**
* **Team Imbalance:** Heavy reliance on top-order batsmen, while middle-order and bowlers lack consistency.
* **Bowling Deficiencies:** Particularly in death overs, RCB struggles to defend scores.
* **Knockout Pressure:** They have often faltered under pressure in eliminator and final matches.
* **Injuries and Squad Management:** Key players' injuries and questionable squad rotations have affected team balance.
* **Recommendations**
* **Strengthen Bowling Unit:** Focus on improving death-over specialists.
* **Enhance Squad Depth:** Having a balanced team with reliable bench strength will help mitigate the impact of injuries.
* **Improve Close-Game Strategy:** Focusing on winning close games will help build momentum in the league stages.

**Result Output:**

**A screenshot of a game

Description automatically generated**

**A graph with numbers and a bar chart

Description automatically generated with medium confidence**

1. How would you approach this problem, if the objective and subjective questions weren't given?

**Ans:**

If the objective and subjective questions weren’t provided, I’d approach analysing RCB’s performance with a slightly different framework, focusing on broader performance evaluation and scenario-based analysis.

**Step 1: Understand the Context and Define Scope**

* Objective Setting: Without predefined goals, I’d focus on understanding RCB’s overall success and failure patterns in the IPL.
* Scope of Analysis: Emphasize team dynamics, and individual performances, and match scenarios to identify areas for improvement.

**Step 2: Perform Data Profiling and Exploration**

* Database Exploration: Review available tables, relationships, and data quality. Identify key tables like matches, ball\_by\_ball, player\_match, etc.
* Data Patterns: Understand recurring trends in team totals, player scores, and match outcomes.

**Step 3: Create Analytical Frameworks**

Instead of predefined KPIs, I’d derive new analytical frameworks:

1. **Game Situations Analysis**

* Performance Under Pressure: Examine batting collapses or clutch bowling in high-pressure situations (e.g., final overs in tight games).
* Toss and Conditions: Analyse how decisions (batting or fielding first) influenced match results based on toss outcomes and venues.

1. **Momentum Metrics**

* Impact of Powerplay: Measure runs scored or wickets taken during powerplay overs to assess early momentum.
* Death Overs Efficiency: Analyse batting strike rates and bowling economy in death overs (16–20).
* Middle Overs Consistency: Evaluate stability in overs 7–15, focusing on partnerships and dot ball percentages.

1. **Player Role Optimization**

* Role-Based Impact: Assess contributions of players in specific roles (e.g., finishers, strike bowlers, all-rounders).
* Match-Winning Performances: Identify games where individual performances were decisive.

1. **Opponent-Specific Insights**

* Examine RCB’s performance against specific teams to uncover potential rivalries or challenges.
* Study head-to-head data to identify favourable matchups or frequent pitfalls.

**Step 4: Statistical Modelling for Deeper Insights**

* Win Probability Models: Develop models to predict match outcomes based on live match situations like wickets in hand or required run rate.
* Clustering Analysis: Group players based on their performance metrics to identify consistent contributors and outliers.

**Step 5: Formulate Hypotheses and Test Them**

1. **Hypotheses:**

* RCB struggles with chasing high totals at specific venues.
* Over-reliance on top-order batting impacts middle-order contributions.
* Bowling economy in death overs correlates strongly with match outcomes.

1. **Test these using SQL queries and visual analytics.**

**Step 6: Recommendations Based on Data-Driven Insights**

1. **Game Strategy:**

* Prioritize partnerships and role clarity to avoid batting collapses.
* Optimize bowler selection for specific venues based on historical performance.

1. **Squad Adjustments:**

* Identify gaps in player roles and suggest acquisitions during auctions or team restructuring.

1. **Scenario-Specific Training:**

* Focus on improving death-over performances, both in batting and bowling.

**Step 7: Advanced Visualization Techniques**

* Use flow charts to illustrate match progression and turning points.
* Create radar charts to compare batting and bowling strengths across teams.
* Build interactive dashboards to allow detailed exploration of venue-specific and player-specific performance.

This approach would provide actionable insights even without predefined objectives or questions.

1. In the "Match" table, some entries in the "Opponent\_Team" column are incorrectly spelled as "Delhi\_Capitals" instead of "Delhi\_Daredevils". Write an SQL query to replace all occurrences of "Delhi\_Capitals" with "Delhi\_Daredevils".

**Ans:**

There is no Match table and no column with Opponent\_team in any of the tables.The available column with the team name is Team\_name in the table Team.And there is no such name available like "Delhi\_Capitals"

If the table and column exist the query will be like this.

A computer code with text

Description automatically generated with medium confidence**Explanation:**

* **UPDATE TEAMS**: Targets the Teams table for updates.
* **SET Team\_name = 'Delhi\_Daredevils'**: Replace the team name in the team table.
* **WHERE Team\_name= 'Delhi\_Capitals'**: Ensures that only rows with the value "Delhi\_Capitals".

After running this query, all instances of "Delhi\_Capitals" in Opponent\_Team will be corrected to "Delhi\_Daredevils.

**Result Output:**

**A screenshot of a sports team

Description automatically generated**