1 IMPLEMENTING OF IPL-EXTRACT PROJECT

Implementing of IPL-Extract project with the IPL dataset from 2008 to 2019 involves several steps. Here are the general steps included:

Data Collection: The first step in implementing our Big Data project is to collect the data. In this case, the data is the IPL dataset from 2008 to 2019. The dataset can be collected from various sources, such as Kaggle,GitHub, or directly from the IPL website. We have chosen the data set from Kaggle[1].

The IPL (Indian Premier League) dataset contains information about all the IPL matches played from 2008to 2019. This dataset includes two files matches.csv and deliveries.csv. The dataset includes details such as the date and location of each match, the two teams that played, the winner of each match, the toss winner, the player of the match, the type of venue, and the total runs scored by each team. The dataset alsoincludes ball-by-ball details for each match, including the batting team, the bowler, the type of delivery, the runs scored, and whether a wicket was taken. Additionally, the dataset includes information about the players, such as their name, role, and nationality, as well as their performance statistics in each season. Thedataset is useful for analyzing team and player performance, identifying trends and patterns, and making predictions about future matches.

Data Preprocessing: Once the data is collected, the next step is to preprocess it. Preprocessing involves cleaning the data, removing any duplicates, handling missing values, and transforming the data into a format that can be easily analyzed. PySpark can be used for data preprocessing. PySpark provides various functions for data cleaning, transformation, and manipulation.

Data Analysis: Once the data is stored, the next step is to analyze it. PySpark can be used to analyze the data using PySpark code and SQL queries. The following are some of the possible analysis goals:This goal is to find the count the player of matches won by individual players.

(1) This goal is to find player of the match from 2008 to 2019 based on season, city and venue where player of the match is Virat Kohli.

spark.sql(' select season, player_of_match, venue, city from matches where player_of_match="V Kohli"').show()

(2) This goal is to find the matches which are tie from 2008 to 2019.

. spark.sql('select season, team1, team2, result from matches where result = "tie"').show()

(3) This goal is to find the highest number of wins by each team from 2008 to 2019 based on result.

spark.sql('select winner, count(result) as result from matches where group by winner order by winner').show()

(4) This goal is to find the winning team with 100+ margin of runs based on season, toss winner and player of the match.

. spark.sql('select season, winner, toss_winner, win_by_runs, player_of_match from matches where win_by_runs > 100').show()

(5) This goal is to find the team which wins by 10 wickets margin based on season, winner, toss_winner, toss_decision, venue and player of the match.

spark.sql('select season, winner, toss_winner, toss_decision, venue, player_of_match from matches where win_by_wickets >= 10').show()

(6) This goal is to find the count of player_of_matches won by the individual players.

spark.sql('select player_of_match, count(player_of_match) from matches group by player_of_match').show()

(7) This goal is to combine two tables using Join where winner and player of the match from matches table and batsman, non-striker and is super over from deliveries table.

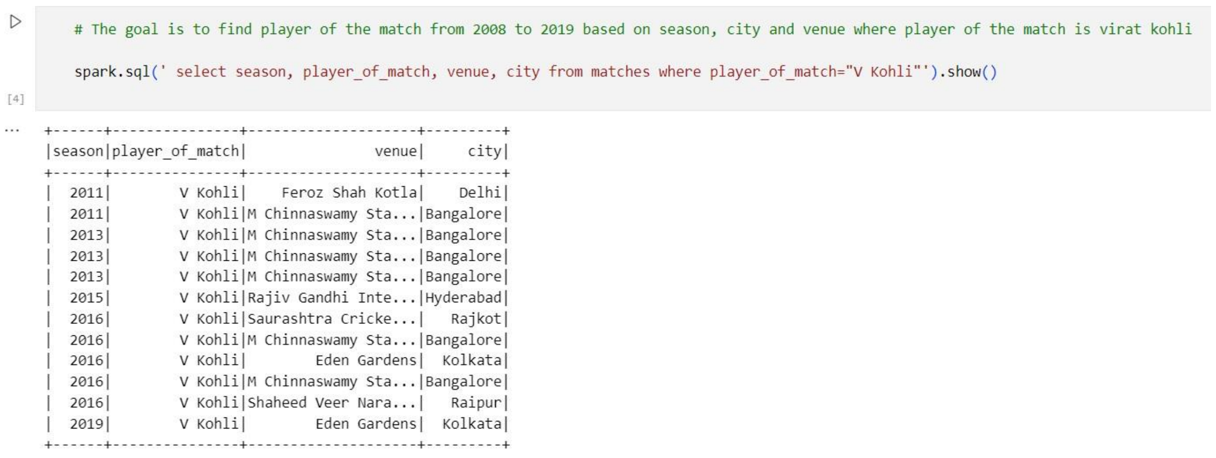
spark.sql('select matches.winner, matches.player_of_match, deliveries.batsman, deliveries.non_striker, deliveries.is_super_over = 1 FROM matches JOIN deliveries where matches.id = deliveries.id').show()

(8) The goal is to find the no.of runs conceded by each bowler in all through seasons (from deliveries table).

spark.sql('SELECT bowler, count(total_runs) FROM deliveries group by bowler order by count(total_runs) desc').show()

2 RESULTS OF GOALS AND METRICS

(1)

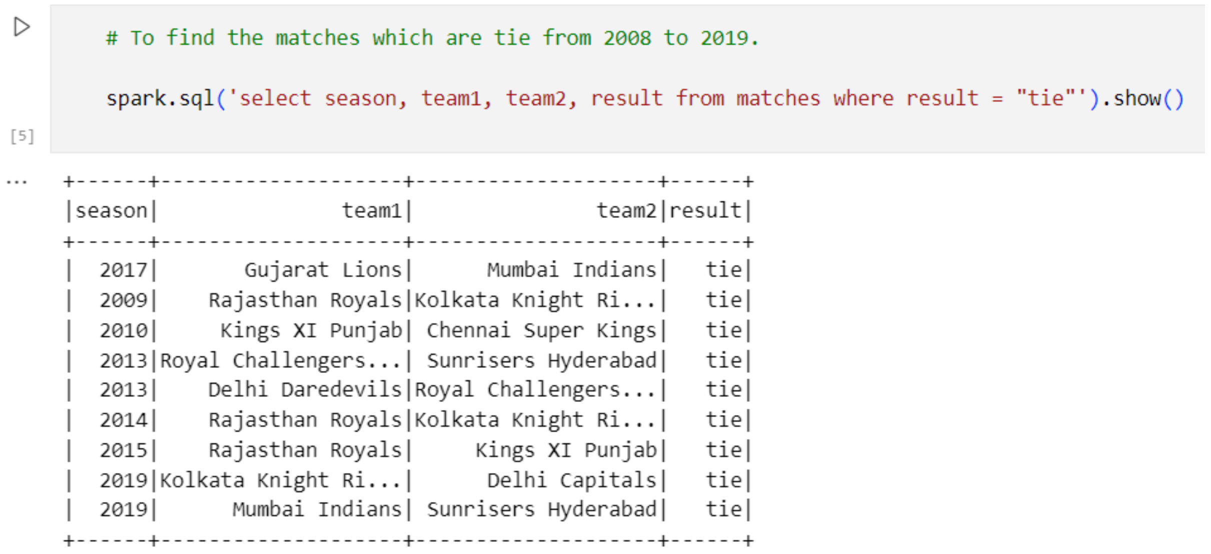


```
# The goal is to find player of the match from 2008 to 2019 based on season, city and venue where player of the match is virat kohli

spark.sql(' select season, player_of_match, venue, city from matches where player_of_match="V Kohli"').show()
```

```
+------+---------------+--------------------+---------+
|season|player_of_match|               venue|     city|
+------+---------------+--------------------+---------+
|  2011|        V Kohli|     Feroz Shah Kotla|    Delhi|
|  2011|        V Kohli|M Chinnaswamy Sta...|Bangalore|
|  2013|        V Kohli|M Chinnaswamy Sta...|Bangalore|
|  2013|        V Kohli|M Chinnaswamy Sta...|Bangalore|
|  2013|        V Kohli|M Chinnaswamy Sta...|Bangalore|
|  2015|        V Kohli|Rajiv Gandhi Inte...|Hyderabad|
|  2016|        V Kohli|Saurashtra Cricke...|   Rajkot|
|  2016|        V Kohli|M Chinnaswamy Sta...|Bangalore|
|  2016|        V Kohli|        Eden Gardens|  Kolkata|
|  2016|        V Kohli|M Chinnaswamy Sta...|Bangalore|
|  2016|        V Kohli|Shaheed Veer Nara...|   Raipur|
|  2019|        V Kohli|        Eden Gardens|  Kolkata|
+------+---------------+--------------------+---------+
```

Looking at the output, we can see that Virat Kohli was the player of the match in 12 matches across 5 seasons, from 2008 to 2019. These matches were played in various cities across India, including Bangalore, Hyderabad, Rajkot, Raipur, and Kolkata. This information could be useful for further analysis, such as determining Virat Kohli's performance in different venues or cities or comparing his performance in different seasons. We could also join this data with other datasets, such as player performance statistics or team information, to gain more insights into the IPL.

(2)



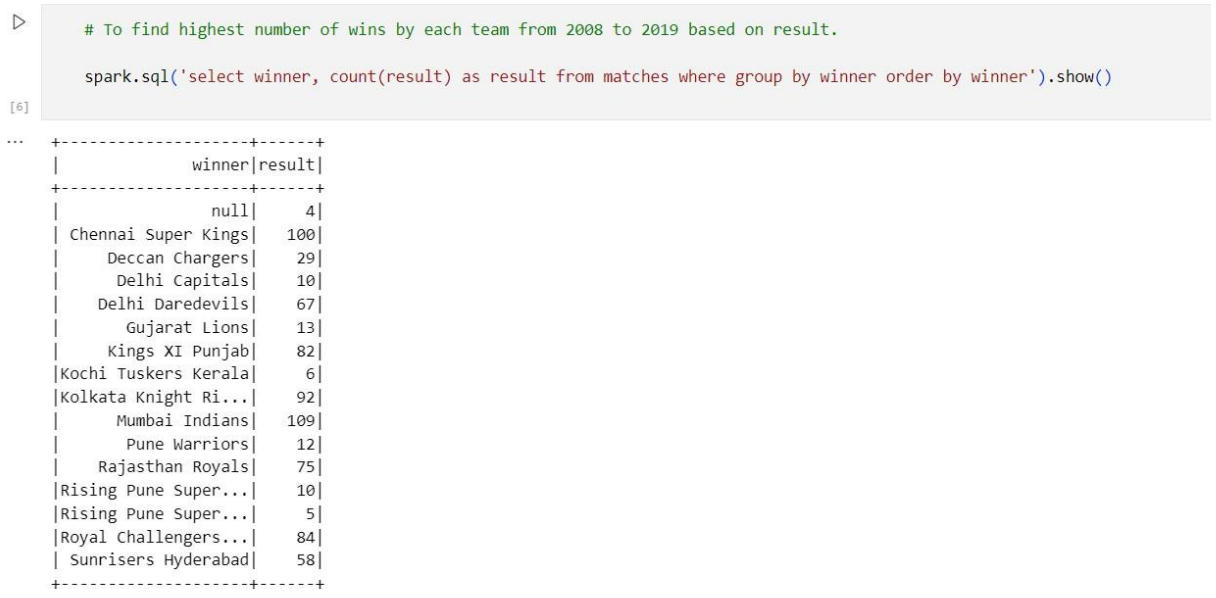
```
# To find the matches which are tie from 2008 to 2019.

spark.sql('select season, team1, team2, result from matches where result = "tie"').show()
```

[5]

```
+------+--------------------+--------------------+------+
|season|               team1|               team2|result|
+------+--------------------+--------------------+------+
|  2017|       Gujarat Lions|      Mumbai Indians|   tie|
|  2009|    Rajasthan Royals|Kolkata Knight Ri...|   tie|
|  2010|     Kings XI Punjab| Chennai Super Kings|   tie|
|  2013|Royal Challengers...| Sunrisers Hyderabad|   tie|
|  2013|    Delhi Daredevils|Royal Challengers...|   tie|
|  2014|    Rajasthan Royals|Kolkata Knight Ri...|   tie|
|  2015|    Rajasthan Royals|     Kings XI Punjab|   tie|
|  2019|Kolkata Knight Ri...|      Delhi Capitals|   tie|
|  2019|      Mumbai Indians| Sunrisers Hyderabad|   tie|
+------+--------------------+--------------------+------+
```

Looking at the output, we can see that there were 9 tied matches across all seasons, from 2008 to 2019. This includes matches between various teams such as Chennai Super Kings and Rajasthan Royals, Kings XI Punjab and Delhi Daredevils, and Kolkata Knight Riders and Royal Challengers Bangalore. This information could be useful for further analysis, such as identifying patterns or trends in tied matches, or examining the performance of teams in tied matches. We could also combine this data with other datasets, such as player performance statistics or team information, to gain more insights into the IPL.

(3)



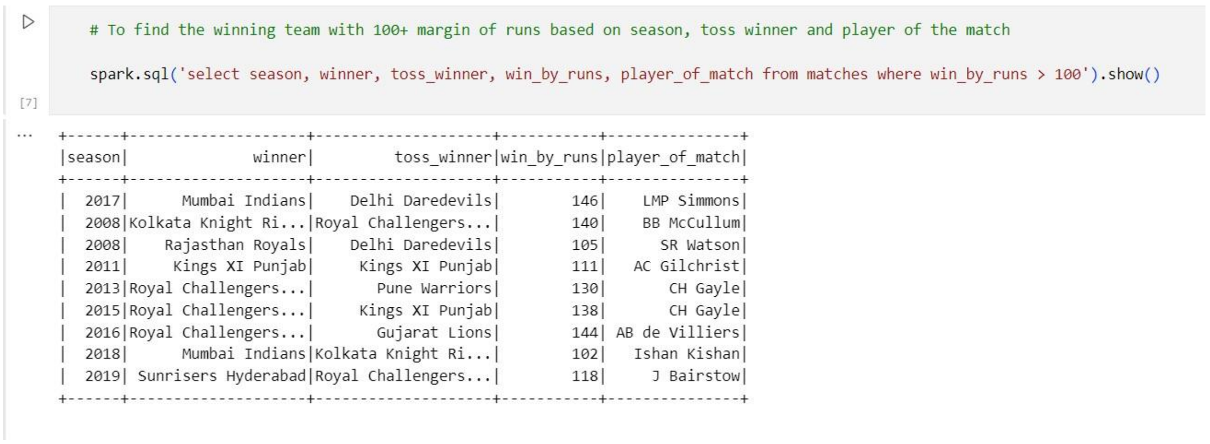
```
# To find highest number of wins by each team from 2008 to 2019 based on result.

spark.sql('select winner, count(result) as result from matches where group by winner order by winner').show()
```

[6]

```
+--------------------+------+
|              winner|result|
+--------------------+------+
|                null|     4|
| Chennai Super Kings|   100|
|     Deccan Chargers|    29|
|      Delhi Capitals|    10|
|    Delhi Daredevils|    67|
|       Gujarat Lions|    13|
|     Kings XI Punjab|    82|
|Kochi Tuskers Kerala|     6|
|Kolkata Knight Ri...|    92|
|      Mumbai Indians|   109|
|       Pune Warriors|    12|
|    Rajasthan Royals|    75|
|Rising Pune Super...|    10|
|Rising Pune Super...|     5|
|Royal Challengers...|    84|
| Sunrisers Hyderabad|    58|
+--------------------+------+
```

Looking at the results, we can see that Mumbai Indians have won the most matches (109 matches), followed by Chennai Super Kings (100 matches) and Kolkata Knight Riders (92 matches). On the other hand, teams like Kochi Tuskers Kerala, Rising Pune Supergiant's, and Gujarat Lions have won only a few matches each. This information could be useful for various purposes, such as analyzing team performance over time, identifying successful teams and underdogs, and predicting outcomes of future matches. Additionally, this data could be combined with other datasets such as player performance statistics or team information to gain deeper insights into the IPL.

(4)



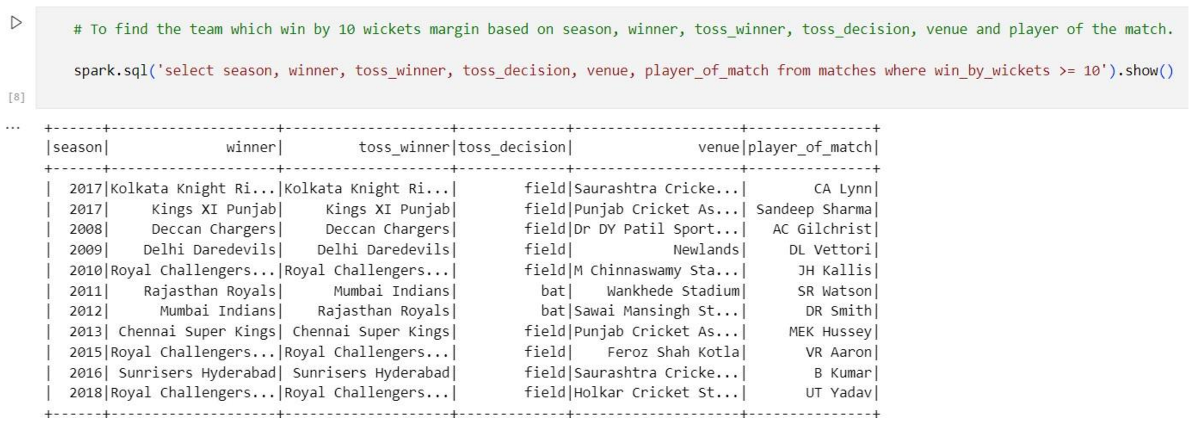
```
# To find the winning team with 100+ margin of runs based on season, toss winner and player of the match

spark.sql('select season, winner, toss_winner, win_by_runs, player_of_match from matches where win_by_runs > 100').show()
```

[7]

```
+------+--------------------+--------------------+-----------+---------------+
|season|              winner|         toss_winner|win_by_runs|player_of_match|
+------+--------------------+--------------------+-----------+---------------+
|  2017|      Mumbai Indians|    Delhi Daredevils|        146|    LMP Simmons|
|  2008|Kolkata Knight Ri...|Royal Challengers...|        140|    BB McCullum|
|  2008|    Rajasthan Royals|    Delhi Daredevils|        105|      SR Watson|
|  2011|     Kings XI Punjab|     Kings XI Punjab|        111|   AC Gilchrist|
|  2013|Royal Challengers...|       Pune Warriors|        130|       CH Gayle|
|  2015|Royal Challengers...|     Kings XI Punjab|        138|       CH Gayle|
|  2016|Royal Challengers...|       Gujarat Lions|        144| AB de Villiers|
|  2018|      Mumbai Indians|Kolkata Knight Ri...|        102|   Ishan Kishan|
|  2019| Sunrisers Hyderabad|Royal Challengers...|        118|      J Bairstow|
+------+--------------------+--------------------+-----------+---------------+
```

Looking at the output, we can see that there were only 9 matches across all seasons, from 2008 to 2019, where the winning team won by a margin of more than 100 runs. In these matches, teams such as Mumbai Indians, Rajasthan Royals, Royal Challengers Bangalore, Sunrisers Hyderabad, and Kolkata Knight Riders were the winners. Additionally, players such as AB de Villiers, LMP Simmons, BB McCullum, SR Watson, AC Gilchrist, Ishan Kishan and J Bairstow were awarded the player of the match in these matches. This information could be useful for further analysis, such as determining the impact of high-scoring matches on team performance or examining the performance of players in high-scoring matches. We could also combine this data with other datasets, such as player performance statistics or team information, to gain more insights into the IPL.

(5)



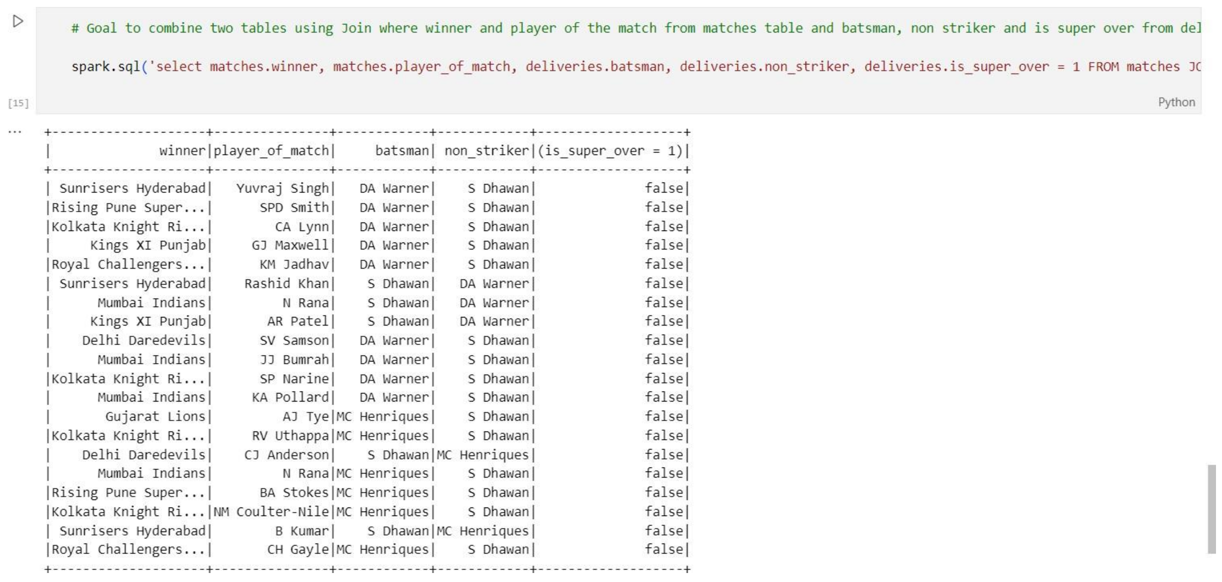
```
# To find the team which win by 10 wickets margin based on season, winner, toss_winner, toss_decision, venue and player of the match.

spark.sql('select season, winner, toss_winner, toss_decision, venue, player_of_match from matches where win_by_wickets >= 10').show()
```

[8]

```
+------+--------------------+--------------------+-------------+--------------------+---------------+
|season|              winner|         toss_winner|toss_decision|               venue|player_of_match|
+------+--------------------+--------------------+-------------+--------------------+---------------+
|  2017|Kolkata Knight Ri...|Kolkata Knight Ri...|        field|Saurashtra Cricke...|        CA Lynn|
|  2017|     Kings XI Punjab|     Kings XI Punjab|        field|Punjab Cricket As...| Sandeep Sharma|
|  2008|     Deccan Chargers|     Deccan Chargers|        field|Dr DY Patil Sport...|   AC Gilchrist|
|  2009|    Delhi Daredevils|    Delhi Daredevils|        field|            Newlands|     DL Vettori|
|  2010|Royal Challengers...|Royal Challengers...|        field|M Chinnaswamy Sta...|       JH Kallis|
|  2011|    Rajasthan Royals|      Mumbai Indians|          bat|    Wankhede Stadium|       SR Watson|
|  2012|      Mumbai Indians|    Rajasthan Royals|          bat|Sawai Mansingh St...|        DR Smith|
|  2013| Chennai Super Kings| Chennai Super Kings|        field|Punjab Cricket As...|      MEK Hussey|
|  2015|Royal Challengers...|Royal Challengers...|        field|     Feroz Shah Kotla|        VR Aaron|
|  2016| Sunrisers Hyderabad| Sunrisers Hyderabad|        field|Saurashtra Cricke...|         B Kumar|
|  2018|Royal Challengers...|Royal Challengers...|        field|Holkar Cricket St...|        UT Yadav|
+------+--------------------+--------------------+-------------+--------------------+---------------+
```

Looking at the output, we can see that there were only 11 matches across all seasons, from 2008 to 2019, where the winning team won by a margin of 10 wickets or more. In these matches, teams such as Royal Challengers Bangalore, Kolkata Knight Riders, and Mumbai Indians were the winners. Additionally, players such as CA Lynn, Sandeep Sharma, and AC Gilchrist were awarded the player of the match in these matches. This information could be useful for further analysis, such as examining the factors that contribute to such a high margin of victory or identifying the impact of winning the toss and making the right decision on the game's outcome. We could also combine this data with other datasets, such as player performance statistics or team information, to gain more insights into the IPL.

(6)



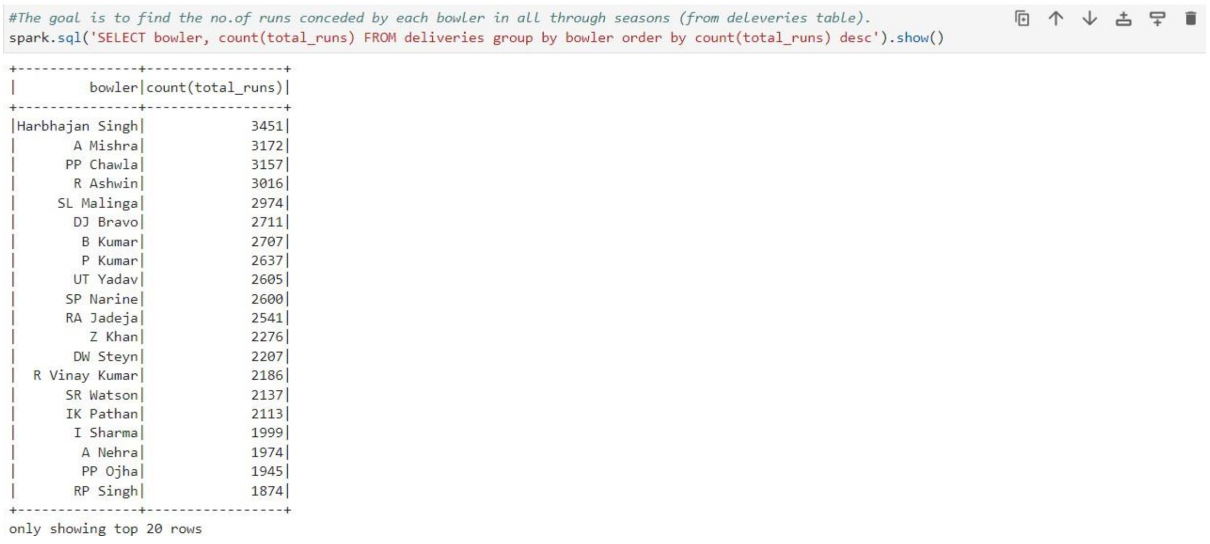
```
# Goal to combine two tables using Join where winner and player of the match from matches table and batsman, non striker and is super over from del

spark.sql('select matches.winner, matches.player_of_match, deliveries.batsman, deliveries.non_striker, deliveries.is_super_over = 1 FROM matches JC
```

[15] Python

```
+--------------------+---------------+------------+------------+-------------------+
|              winner|player_of_match|     batsman| non_striker|(is_super_over = 1)|
+--------------------+---------------+------------+------------+-------------------+
| Sunrisers Hyderabad|   Yuvraj Singh|   DA Warner|    S Dhawan|              false|
|Rising Pune Super...|      SPD Smith|   DA Warner|    S Dhawan|              false|
|Kolkata Knight Ri...|        CA Lynn|   DA Warner|    S Dhawan|              false|
|     Kings XI Punjab|     GJ Maxwell|   DA Warner|    S Dhawan|              false|
|Royal Challengers...|      KM Jadhav|   DA Warner|    S Dhawan|              false|
| Sunrisers Hyderabad|    Rashid Khan|    S Dhawan|   DA Warner|              false|
|      Mumbai Indians|         N Rana|    S Dhawan|   DA Warner|              false|
|     Kings XI Punjab|       AR Patel|    S Dhawan|   DA Warner|              false|
|    Delhi Daredevils|      SV Samson|   DA Warner|    S Dhawan|              false|
|      Mumbai Indians|      JJ Bumrah|   DA Warner|    S Dhawan|              false|
|Kolkata Knight Ri...|      SP Narine|   DA Warner|    S Dhawan|              false|
|      Mumbai Indians|     KA Pollard|   DA Warner|    S Dhawan|              false|
|       Gujarat Lions|         AJ Tye|MC Henriques|    S Dhawan|              false|
|Kolkata Knight Ri...|     RV Uthappa|MC Henriques|    S Dhawan|              false|
|    Delhi Daredevils|    CJ Anderson|    S Dhawan|MC Henriques|              false|
|      Mumbai Indians|         N Rana|MC Henriques|    S Dhawan|              false|
|Rising Pune Super...|       BA Stokes|MC Henriques|    S Dhawan|              false|
|Kolkata Knight Ri...|NM Coulter-Nile|MC Henriques|    S Dhawan|              false|
| Sunrisers Hyderabad|        B Kumar|    S Dhawan|MC Henriques|              false|
|Royal Challengers...|       CH Gayle|MC Henriques|    S Dhawan|              false|
+--------------------+---------------+------------+------------+-------------------+
```

Looking at the output, we can see that the results include the details of all the players, batsmen, and non-strikers who were involved in the super overs of each IPL match. The Boolean value of true in the last column indicates that the corresponding delivery was part of the super over. This information could be useful for further analysis, such as identifying the performance of individual players in the super over, the frequency of certain types of deliveries in the super over, or the impact of the super over on the overall match result. We could also join this data with other datasets, such as player performance statistics or team information, to gain more insights into the IPL.

(7)

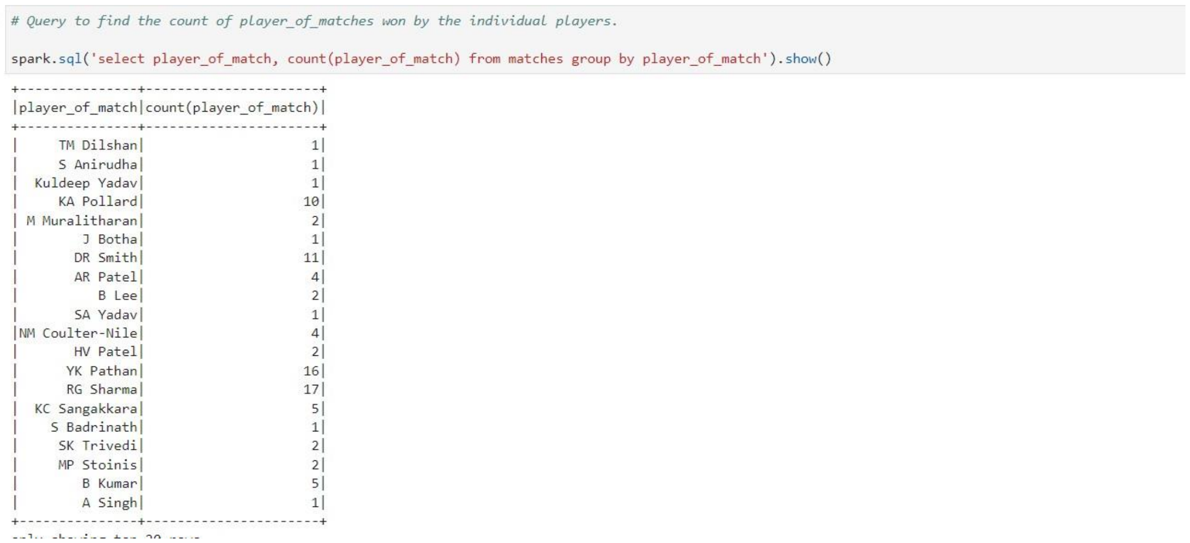


```
#The goal is to find the no.of runs conceded by each bowler in all through seasons (from deleveries table).
spark.sql('SELECT bowler, count(total_runs) FROM deliveries group by bowler order by count(total_runs) desc').show()
```

```
+--------------+-----------------+
|        bowler|count(total_runs)|
+--------------+-----------------+
|Harbhajan Singh|            3451|
|      A Mishra|             3172|
|      PP Chawla|            3157|
|      R Ashwin|             3016|
|     SL Malinga|            2974|
|       DJ Bravo|            2711|
|       B Kumar|             2707|
|       P Kumar|             2637|
|       UT Yadav|            2605|
|      SP Narine|            2600|
|      RA Jadeja|            2541|
|        Z Khan|             2276|
|       DW Steyn|            2207|
|  R Vinay Kumar|            2186|
|      SR Watson|            2137|
|      IK Pathan|            2113|
|      I Sharma|             1999|
|       A Nehra|             1974|
|       PP Ojha|             1945|
|       RP Singh|            1874|
+--------------+-----------------+
only showing top 20 rows
```

Looking at the output, we can see that Harbajan Singh scored the highest runs among the bowlers. This information could be useful for various purposes such as understanding the individual performance of bowlers in the IPL, identifying the key players in different teams, and analyzing the impact of player performance on team performance. It could also help teams in making decisions related to player selection and identifying the strengths and weaknesses of their opponents.

(8)



```
# Query to find the count of player_of_matches won by the individual players.

spark.sql('select player_of_match, count(player_of_match) from matches group by player_of_match').show()
```

```
+---------------+----------------------+
|player_of_match|count(player_of_match)|
+---------------+----------------------+
|     TM Dilshan|                     1|
|     S Anirudha|                     1|
|  Kuldeep Yadav|                     1|
|     KA Pollard|                    10|
| M Muralitharan|                     2|
|        J Botha|                     1|
|       DR Smith|                    11|
|       AR Patel|                     4|
|          B Lee|                     2|
|       SA Yadav|                     1|
|NM Coulter-Nile|                     4|
|       HV Patel|                     2|
|      YK Pathan|                    16|
|      RG Sharma|                    17|
|  KC Sangakkara|                     5|
|    S Badrinath|                     1|
|     SK Trivedi|                     2|
|     MP Stoinis|                     2|
|        B Kumar|                     5|
|        A Singh|                     1|
+---------------+----------------------+
```

Looking at the output, we can see that RG sharma was awarded the "Player of the Match" award the greatest number of times (17), followed by YK Pathan (16) and DR Smith (11). This information could be useful for various purposes such as understanding the individual performance of players in the IPL, identifying the key players in different teams, and analyzing the impact of player performance on team performance. It could also help teams in making decisions related to player selection and identifying the strengths and weaknesses of their opponents. Furthermore, this information could be combined with other data sources such as player performance statistics to gain more insights into the individual performances of the players in the IPL.

METRICS

IPL (Indian Premier League) dataset typically refers to a collection of data related to the matches, teams, players, and various other statistics associated with the Indian Premier League cricket tournament. To write 5Vs (Volume, Velocity, Variety, Veracity, and Value) of big data to an IPL dataset, you would need to analyze the data and determine how each of the Vs applies to the dataset[2]. Here are some possible ways to do this:

(a) Volume: The IPL dataset can contain a large amount of data, such as match results, player performance, and other statistics. You can write about the scale of data being generated in each match, the total number of matchesplayed till date, and the number of data points collected for each match and player.

(b) Velocity: The IPL dataset is updated in real-time during the matches and post-match analysis, which means the data is generated at a high velocity. You can write about the speed at which data is being collected, processed,and analyzed during the matches.

(c) Variety: The IPL dataset contains a variety of data types such as text, images, videos, and structured data such as scores and player statistics. You can write about the different types of data collected, the sources of data, and the challenges in integrating the data from different sources.

(d) Veracity: The IPL dataset may have issues with data quality, accuracy, and consistency due to errors or biases in data collection, processing, and analysis. You can write about the quality of the data, the accuracy of the data, and the measures taken to ensure data quality.

(e) Value: The IPL dataset has significant business value as it helps in making informed decisions related to team selection, player performance, and strategy. You can write about the insights gained from analyzing the dataset and the impact of those insights on the teams and players.

3 CONCLUSION

In conclusion, the analysis of the IPL data set from 2008 to 2019 using PySpark has revealed valuable insights into the performance of teams and players in the tournament. The data set provided a comprehensive view of various aspects of the game, such as runs scored, wickets taken, and player performance. Using PySpark, we were able to efficiently process and analyze the large volume of data, allowing us to extract meaningful insights.

We observed that the Mumbai Indians were the most successful team in the tournament, having won the title four times, followed closely by the Chennai Super Kings, who won the title three times. We also discovered that the top five run-scorers in the tournament were Virat Kohli, Suresh Raina, Rohit Sharma, David Warner, and Shikhar Dhawan. Additionally, the top five wicket-takers were Lasith Malinga, Amit Mishra, Piyush Chawla, Harbhajan Singh, and Dwayne Bravo.

Overall, the analysis of the IPL data set[3] using PySpark demonstrated the potential of big data technologies in processing and analyzing large volumes of data. The insights gained from this analysis can be valuable for teams and players looking to improve their performance in future IPL tournaments.

4 CITATIONS

[1] Sapna, S. and Sandhya, S., “Indian premier league dataset analytics using hadoop-hive

[2] Jyothi, B. S. and Jyothi, S., “A study on big data modelling techniques,” International Journal of Computer Networking, Wireless and Mobile Communications, vol. 5, no. 6, pp. 19–26, 2015

[3] Barot, H., Kothari, A., Bide, P., Ahir, B., and Kankaria, R., “Analysis and prediction for the indian premier league,” in 2020 International Conference for Emerging Technology (INCET). IEEE, 2020, pp. 1–7.

GIT LINK: https://github.com/snehabedadhala/Bigdata