**CRICKET DATABASE**

**Abstract**

The purpose of the assignment is build a conceptual for cricket domain with entities falling under ‘Comoany’, ‘Producer’ and ‘Consumer’ and subsequently build a physical model using real world data. The data was obtained from three sources were cleaned, reformatted and then combined to fit the conceptual database schema. The data was obtained from three sources and was cleaned, reformatted and then combined to fit the conceptual database schema and physical model.

**Data Source:-**

Source 1: Data was extracted from Twitter using Twitter API

Source 2: Data was extracted from Reddit using PRAW (The Python Reddit API Wrapper)

Source 3: Dataset obtained from Kaggle

**Part 1: Conceptual Model:**

Cricket was chosen as the domain subject.

Conceptual model entities were created as a result of data extraction and merging. The conceptual model was constructed with real world data extracted from two social media sites namely Twitter and Reddit and one dataset from Kaggle, which was extracted from ESPN cricinfo statsguru.

The created conceptual entities fell under three basic categories namely Companies, Producers and Consumers

The classification of the conceptual model entities based on the category are as follows:

**Company:** Logically, ‘Team’ corresponds to Company category, which houses lot of players, who are being followed by several users on social media. ‘TeamBattingStatistics’ and ‘TeamBowlingStatistics’ are the supporting entities related to ‘Team’

**Producer:** ‘Player’ corresponds to the Producer category, who belongs to a team(company) and who are being followed by users(consumers). ‘PlayerBattingStatistics’, ‘PlayerBowlingStatistics’ and ‘PlayerFieldingStatistics’ are entities which support the ‘Player’ entity.

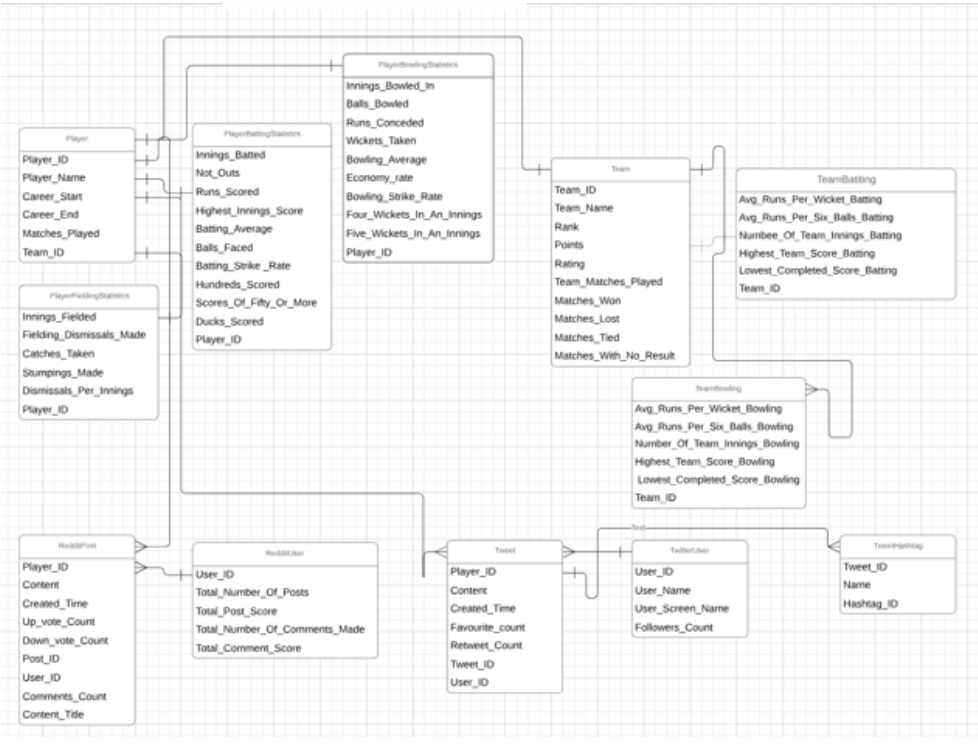
**Consumer:** ‘TwiiterUser’ and ‘RedditUser’ correspond to the consumer category, as they consume or follow the players, who belong to a team. ‘TwitterHashtag’, ‘Tweet’ are related entities to ‘TwiterUser and ‘RedditPost’ is related to ‘RedditUser’.

In the below table, the entities are cateogorized and the main entities are marked in bold, while the supporting entities are left in normal font.

|  |  |  |
| --- | --- | --- |
| **Company** | **Producer** | **Consumer** |
| **Team** | **Player** | **TwitterUser** |
| TeamBattingStatistics | PlayerBattingStatistics | TwitterHashtag |
| TeamBowlingStatistics | PlayerBowlingStatistics | Tweet |
|  | PlayerFieldingStatistics | **RedditUser** |
|  |  | RedditPost |
|  |  |  |

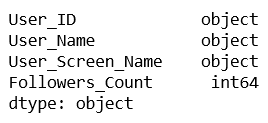
**The overall conceptual model :**

The model displays the conceptual entities and their relationships.

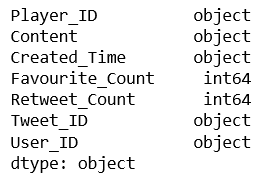


Datatype information for the entities are as follows:

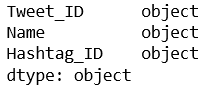
1. TwitterUser:



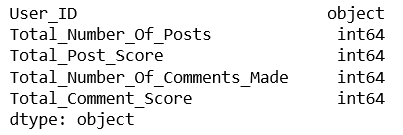
1. Tweet:



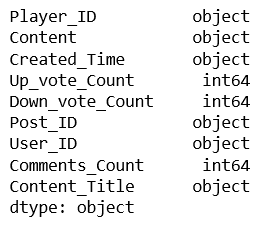
1. TwitterHashtag:



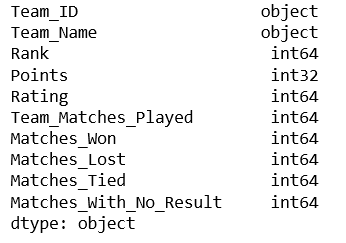
1. RedditUser:



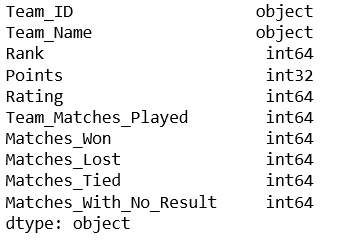
1. RedditPost:



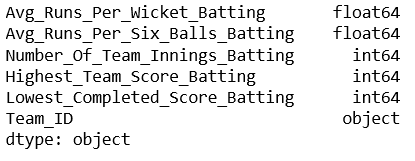
1. Team:



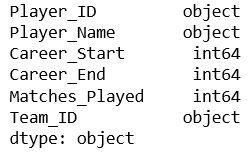
1. TeamBattingStatistics:



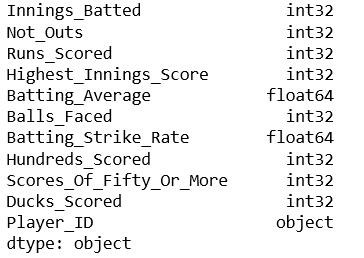
1. TeamBowlingStatistics:



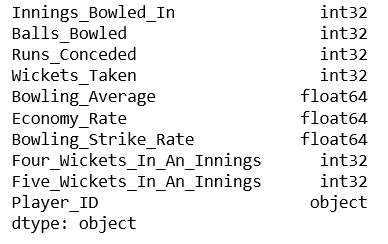
1. Player:



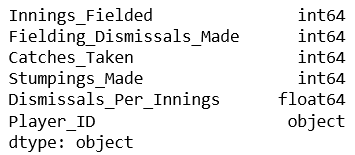
1. PlayerBattingStatistics:



1. PlayerBowlingStatistics:



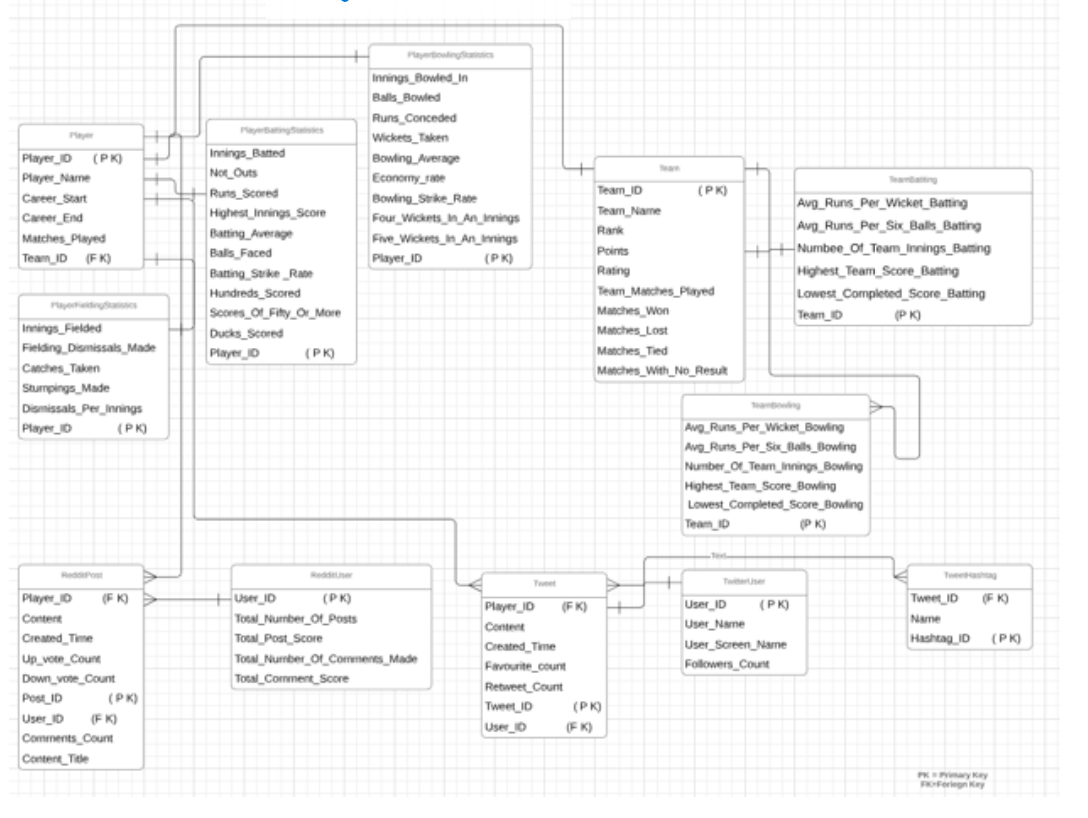
1. PlayerFieldingStatistics:



**Relationships between the entities:**

1. Player and PlayerBattingStatistics: **One to one relationship**, since one player can have only one batting statistical record corresponding to him/her in batting statistics table.
2. Player and PlayerBowlingStatistics: **One to one relationship**, since one player can have only one bowling statistical record corresponding to him/her in bowling statistics table.
3. Player and PlayerFieldingStatistics: **One to one relationship**, since one player can have only one fielding statistical record corresponding to him/her in fielding statistics table.
4. Player and Team: **Many to one relationship**, since many players can be part of the same team.
5. Team and TeamBattingStatistics: **One to one relationship**, since one team can have only one batting statistical record corresponding to it in batting statistics table.
6. Team and TeamBowlingStatisics: **One to one relationship**, since one team can have only one bowling statistical record corresponding to it in bowling statistics table.
7. Player and Tweet: One to many relationship, since there can be numerous tweets made about a single player.
8. Tweet and TwitterUser: **Many to one relationship**, since one user may post may tweets.
9. Tweet and TwitterHashtag: **One to many relationship**, since one tweet can contain many hashtags.
10. Player and RedditPost: **One to many relationship**, since there can be numerous posts made about a single player.
11. Tweet and TwitterUser: **Many to one relationship**, since one user may post may posts.

The ER diagram developed for the conceptual model is as follows:



The ER diagram gives a more clearer picture with relationships, as it highlights the primary key and foreign keys within the entities.

All the entities mentioned in the model qualified to be entities as all of them posed as a thing and their properties were modelled as their corresponding attributes.

For example, Player cannot be just made as an attribute within team, since it possess lot of information within it to qualify as an entity.

Similary, Player\_Name, Matches\_Played can’t be individua entities themselves, since at max they convey information about the player and qualify themselves to the attributes of Player.

There were no design tradeoffs, while classifying entities or attributes.

**Primary and Foreign key classification among entities:**

**Player:**

PK: Player\_ID, as it uniquely identifies every record in the Player table.

FK: Team\_ID, as every player will be part of team and there should be a key for handling the connection.

**PlayerBattingStaistics:**

PK: Player\_ID: Uniquely identifies each record in the table, as one player cannot have more than one batting statistical record.

**PlayerBowlingStaistics**:

PK: Player\_ID: Uniquely identifies each record in the table, as one player cannot have more than one bowling statistical record.

**PlayerFieldingStaistics:**

PK: Player\_ID: Uniquely identifies each record in the table, as one player cannot have more than one fielding statistical record.

**Team:**

PK: Team\_ID, as it uniquely identifies every record in the Team table.

**TeamBattingStaistics:**

PK: Team\_ID: Uniquely identifies each record in the table, as one team cannot have more than one batting statistical record.

**TeamBowlingStaistics:**

PK: Team\_ID: Uniquely identifies each record in the table, as one team cannot have more than one bowling statistical record.

**Tweet:**

PK: Tweet\_ID, , as it uniquely identifies every record in the Tweet table.

FK1: User\_ID: One user can post more than one tweet,so a key must be present to handle the connection.

FK2: Player\_ID: One player can more than one tweet, o a key must be present to handle the connection.

**TwitterUser:**

PK: User\_ID, as it uniquely identifies every record in the TwitterUser table.

**TwitterHashtag:**

PK: Hashtag\_D, as it uniquely identifies every record in the TwitterHashtag table.

FK1: Tweet\_ID, as one tweet can have more than one hashtags, so a connection must be made to handle the same.

**RedditPost:**

PK: Post\_ID, as it uniquely identifies every record in the RedditPost table.

FK1: User\_ID, as one user can post more than one post, so a key is required to hande the connection.

FK2: Player\_ID, as more than one post can be made about a playe. So, a key must be present to handle the connection.

**RedditUser:**

PK: User\_ID, as it uniquely identifies every record in the RedditUser table.

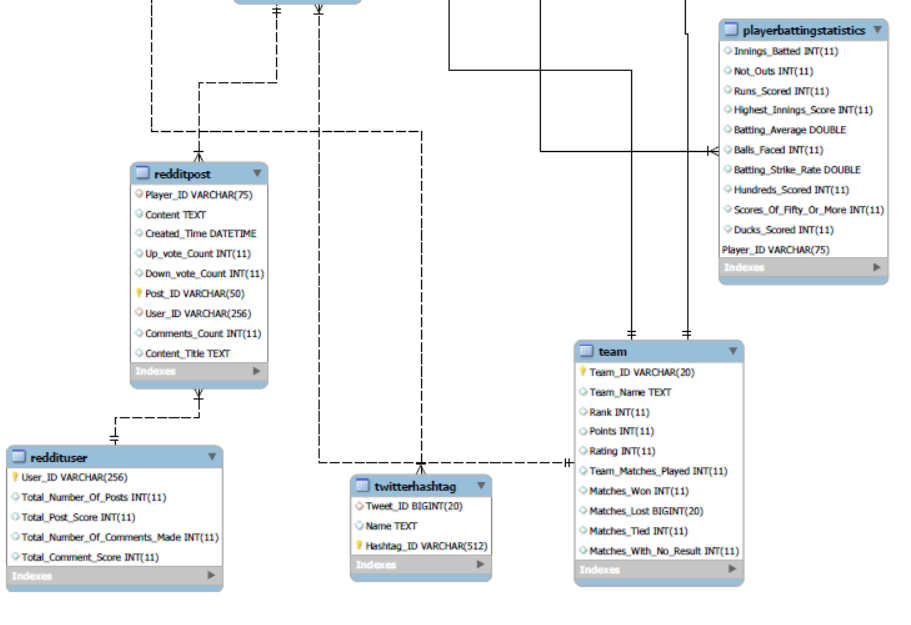
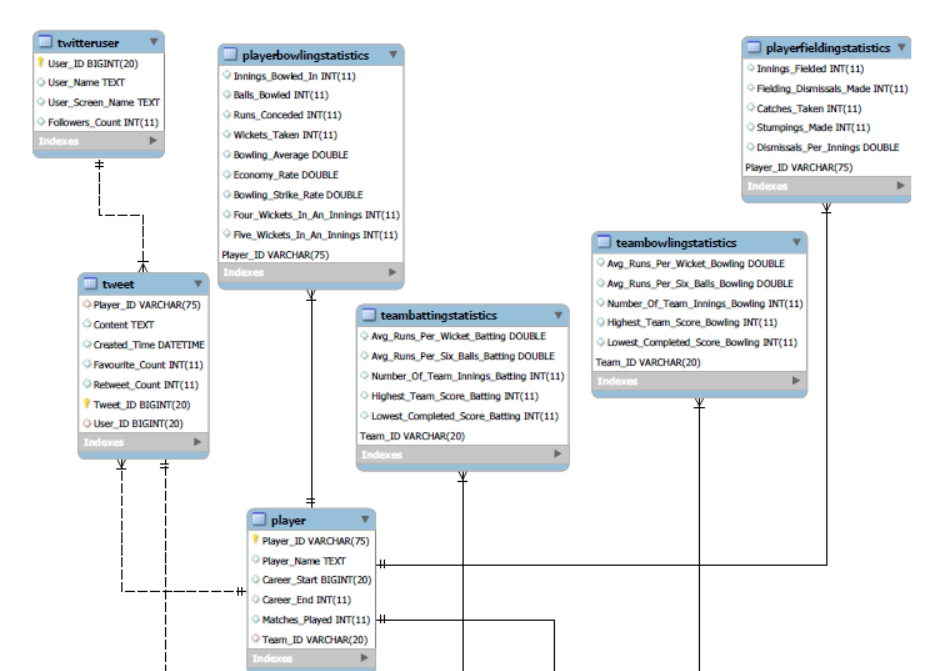
ISA design element was not used to model hierarchies, as it was not required.

**Part 2: Physical Model:**

The ER diagram, which was developed out of the conceptual model was not used for plotting the physical model, as no changes with regard to entities, attributes and keys were required.

In addition to the ER diagram, the physical model displays the data type of each and every entity along with relationships(cardinality), keys and attributes.

The physical model is as follows:



**Queries:**

The Queries, which were required to be answered from physical model(In SQL):

a. What user posted this (e.g. tweet, facebook post, IG post, etc.)?

Query:

*#Twitter:*

*#Displaying information about user, who had tweeted the particular tweet about cricketer Suresh Raina*

*select t.Tweet\_ID, t.Content as "Tweet\_Content", u.User\_ID, u.User\_Name, u.User\_Screen\_Name*

*from cricket\_socialmediaanalysis.tweet t, cricket\_socialmediaanalysis.twitteruser u*

*where t.User\_ID = u.User\_ID*

*and t.Content like '%@ImRaina But we r missing our hero%'*

*order by u.User\_Name;*

*#Reddit:*

*#Displaying information about user, who had posted the particular reddit post about cricketer Kane Williamson*

*select p.Post\_ID, p.Content\_Title as "Post\_Title", p.Content as "Post\_Content", u.User\_ID*

*from cricket\_socialmediaanalysis.redditpost p, cricket\_socialmediaanalysis.reddituser u*

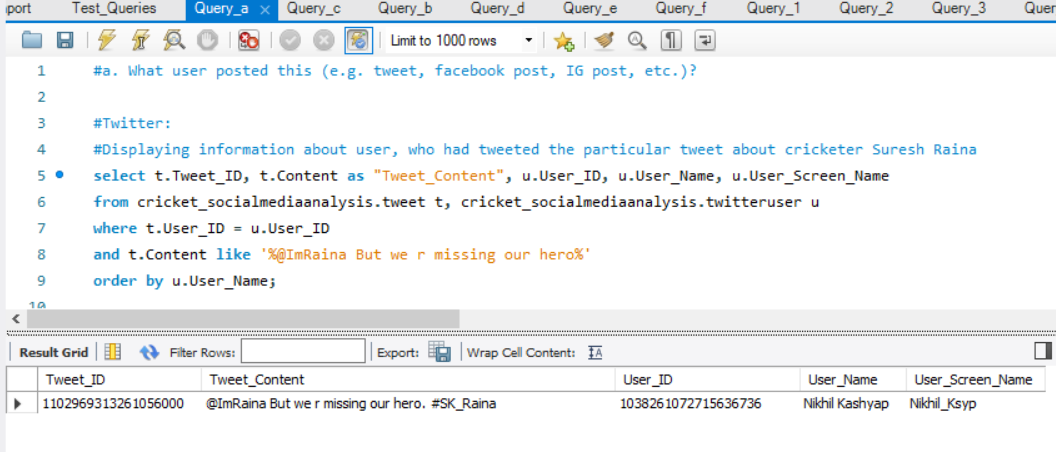
*where p.User\_ID = u.User\_ID*

*and p.Content\_Title like '%Kane Williamsons exceptional performance as test captain%'*

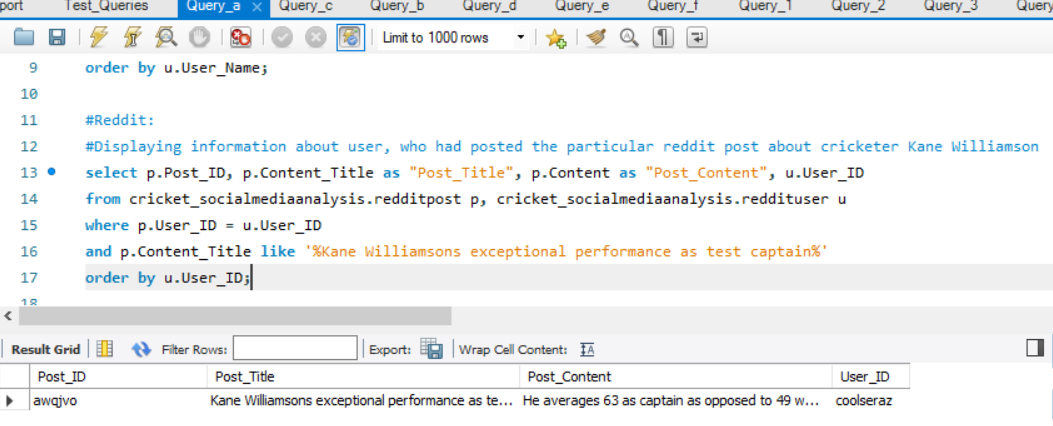
*order by u.User\_ID;*

Query result:

Twitter:



Reddit:



b. When did the user post this (e.g. tweet, facebook post, IG post, etc.)??

Query:

*#Twitter:*

*#Displaying information about when the user had tweeted the particular tweet about cricketer Suresh Raina*

*select t.Tweet\_ID, t.Content as "Tweet\_Content", u.User\_ID, u.User\_Name, u.User\_Screen\_Name, t.Created\_Time as "Tweet\_Datetime"*

*from cricket\_socialmediaanalysis.tweet t, cricket\_socialmediaanalysis.twitteruser u*

*where t.User\_ID = u.User\_ID*

*and t.Content like '%@ImRaina But we r missing our hero%'*

*order by u.User\_Name;*

*#Reddit:*

*#Displaying information about when the user had posted the particular reddit post about cricketer Kane Williamson*

*select p.Post\_ID, p.Content\_Title as "Post\_Title", p.Content as "Post\_Content", u.User\_ID, p.Created\_Time as "Post\_Datetime"*

*from cricket\_socialmediaanalysis.redditpost p, cricket\_socialmediaanalysis.reddituser u*

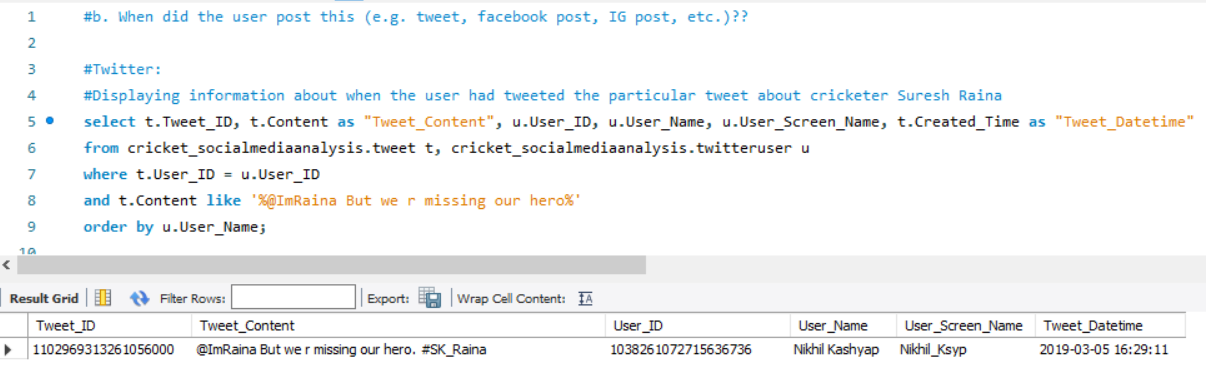
*where p.User\_ID = u.User\_ID*

*and p.Content\_Title like '%Kane Williamsons exceptional performance as test captain%'*

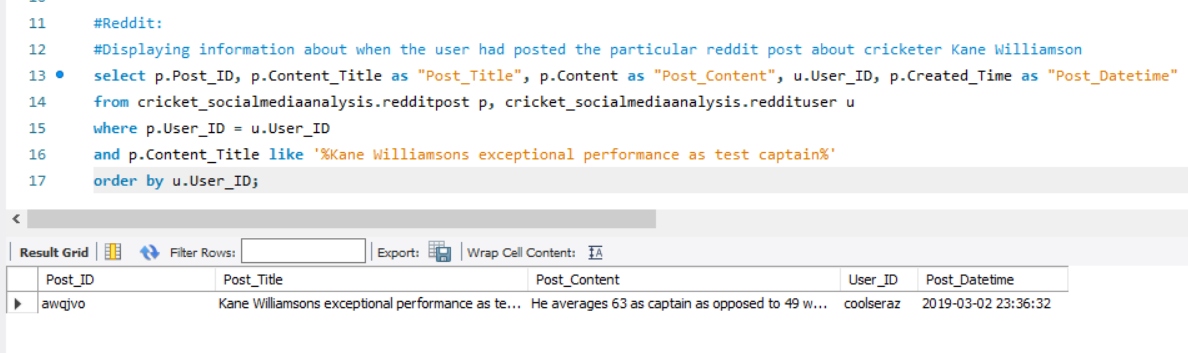
*order by u.User\_ID;*

Query Result:

Twitter:



Reddit:



c. What posts has this user posted in the past 24 hours?

Query:

*#Twiitter*

*#Displaying information about what user 'Pranaybolla' had tweeted in past 24 hours*

*select t.Tweet\_ID, t.Content as "Tweet\_Content", u.User\_ID, u.User\_Name, u.User\_Screen\_Name, t.Created\_Time as "Tweet\_Datetime"*

*from cricket\_socialmediaanalysis.tweet t, cricket\_socialmediaanalysis.twitteruser u*

*where t.User\_ID = u.User\_ID*

*and u.User\_Name = 'Pranaybolla'*

*and t.Created\_Time >= NOW() - INTERVAL 1 DAY*

*order by u.User\_Name;*

*#Reddit:*

*#Displaying information about what user 'rokkmrt3' had posted in reddit in past 24 hours*

*select p.Post\_ID, p.Content\_Title as "Post\_Title", p.Content as "Post\_Content", u.User\_ID, p.Created\_Time as "Post\_Datetime"*

*from cricket\_socialmediaanalysis.redditpost p, cricket\_socialmediaanalysis.reddituser u*

*where p.User\_ID = u.User\_ID*

*and u.User\_ID = 'rokkmrt3'*

*and p.Created\_Time >= NOW() - INTERVAL 1 DAY*

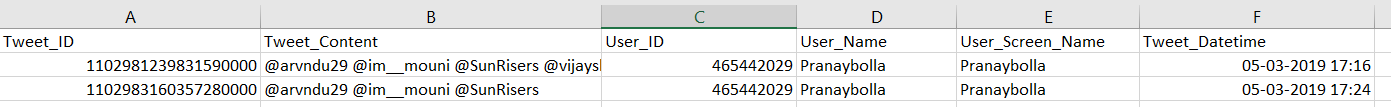
*order by u.User\_ID;*

Query result:

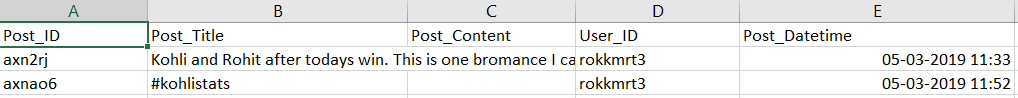
Since the data was loaded into database 24 hours before report preparation, the result is presently unavailable.

But have attached the exported and saved query results

Twitter:



Reddit:



d. How many post has this user posted in the past 24 hours?

Since the data was loaded into database 24 hours before report preparation, the result is presently unavailable.

But have attached the exported and saved query results

Query:

*#Twitter*

*#Displaying information about how many tweets has user 'Pranaybolla' tweeted in past 24 hours*

*select count(t.Tweet\_ID) as "Number\_Of\_Tweets", u.User\_Name*

*from cricket\_socialmediaanalysis.tweet t, cricket\_socialmediaanalysis.twitteruser u*

*where t.User\_ID = u.User\_ID*

*and u.User\_Name = 'Pranaybolla'*

*and t.Created\_Time >= NOW() - INTERVAL 1 DAY*

*group by u.User\_Name*

*order by u.User\_Name;*

*#Reddit:*

*#Displaying information about how many posts has user 'rokkmrt3' posted in reddit in past 24 hours*

*select count(p.Post\_ID) as "Number\_Of\_Reddit\_Posts", u.User\_ID*

*from cricket\_socialmediaanalysis.redditpost p, cricket\_socialmediaanalysis.reddituser u*

*where p.User\_ID = u.User\_ID*

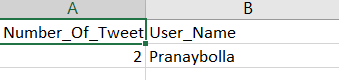
*and u.User\_ID = 'rokkmrt3'*

*and p.Created\_Time >= NOW() - INTERVAL 1 DAY*

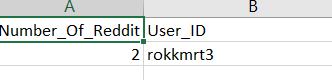
*order by u.User\_ID;*

Query\_Result:

Twitter:



Reddit:



e. What keywords/ hashtags are popular?

Query:

*#Twitter*

*#Displaying information about the twitter hashtag used for most number of times among the stored data*

*select count(ht.Hashtag\_ID) as "Usage\_Count", ht.Name*

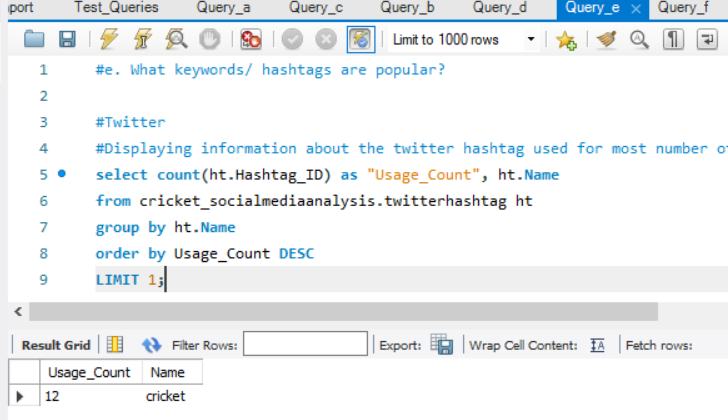
*from cricket\_socialmediaanalysis.twitterhashtag ht*

*group by ht.Name*

*order by Usage\_Count DESC*

*LIMIT 1;*

Query\_Result:



f. What posts are popular?

Query:

*#Twitter*

*#Displaying the top 50 popular tweet ids, calculated in terms of sum of retweet count and favourite count*

*select t.Tweet\_ID, (SUM(t.Retweet\_Count) + SUM(t.Favourite\_Count)) as "Total\_Popularity\_Count"*

*from cricket\_socialmediaanalysis.tweet t*

*group by t.Tweet\_ID*

*order by Total\_Popularity\_Count DESC*

*LIMIT 50;*

*#Reddit*

*#Displaying the popular post ids, calculated in terms of sum of Up\_vote\_Count and Comments\_Count*

*select p.Post\_ID, (SUM(p.Up\_vote\_Count) + SUM(p.Comments\_Count)) as "Total\_Popularity\_Count"*

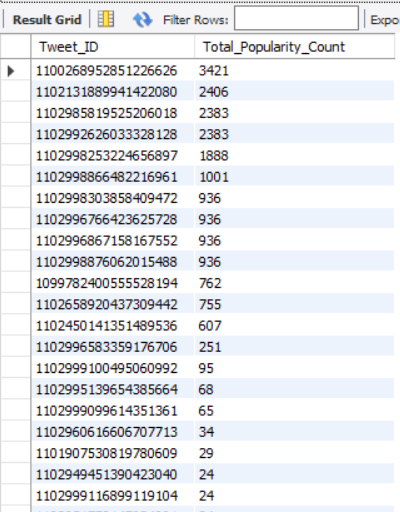
*from cricket\_socialmediaanalysis.redditpost p*

*group by p.Post\_ID*

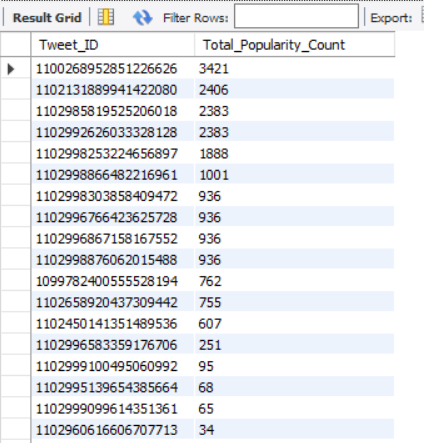
*order by Total\_Popularity\_Count DESC;*

Query Result:

Twitter:



Reddit:



**Other user cases:**

1.

#Use case (1)

#Description: Displaying players with most twitter popularity count(Retweet count plus Favourite count)

#Precondition: Only Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: Find the tweets related to players and then calculate the popularity count (Retweet count plus Favourite count) of each tweet, thereby summing up the total for every player.

#Response: Return the list of players and their popularity count.

#Displaying the top 50 popular tweet ids, calculated in terms of sum of retweet count and favourite count

*select p.Player\_Name, (SUM(t.Retweet\_Count) + SUM(t.Favourite\_Count)) as "Total\_Popularity\_Count"*

*from cricket\_socialmediaanalysis.player p*

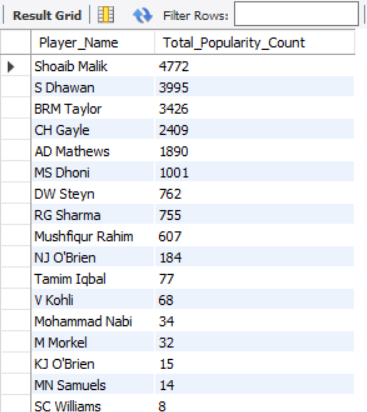
*INNER join cricket\_socialmediaanalysis.tweet t*

*on p.Player\_ID = t.Player\_ID*

*group by p.player\_Name*

*order by Total\_Popularity\_Count DESC;*

Result:



2.

Use case (2)

#Description: Displaying players with most reddit popularity count(Up vote count plus Comments count)

#Precondition: Only Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Also, posts under r/Cricket channel were alone considered for evaluation.

#Steps: Find the posts related to players and then calculate the popularity count (Up vote count plus Comment count) of each post, thereby summing up the total for every player.

#Response: Return the list of players and their popularity count.

*select p.Player\_Name, (SUM(post.Up\_vote\_Count) + SUM(post.Comments\_Count)) as "Total\_Popularity\_Count"*

*from cricket\_socialmediaanalysis.player p*

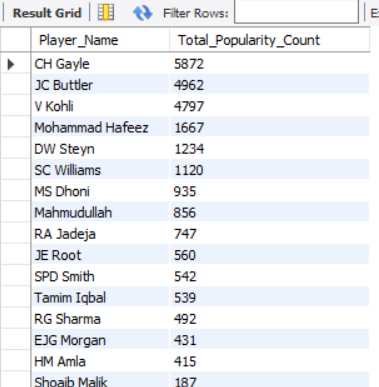
*inner join cricket\_socialmediaanalysis.redditpost post*

*on p.Player\_ID = post.Player\_ID*

*group by p.Player\_Name*

*order by Total\_Popularity\_Count DESC;*

Result:



3.

#Use case (3)

#Description: Displaying tweet count per team from the stored tweets information, thereby accessing team popularity in social media.

#Precondition: Only team with Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: Find the tweets related to players and then relate it to teams, thereby summing the tweets per team.

#Response: Return the list of teams with their related tweet count.

*select te.Team\_Name,count(Tweet\_ID) as "Tweet\_Count"*

*from cricket\_socialmediaanalysis.tweet t*

*inner join cricket\_socialmediaanalysis.player p*

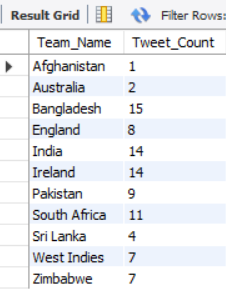
*on t.Player\_ID = p.Player\_ID*

*inner join cricket\_socialmediaanalysis.team te*

*on te.Team\_ID = p.Team\_ID*

*group by te.Team\_Name;*

Result:



4.

#Use case (4)

#Description: Displaying number of reddit posts per day for a particular cricket team given as input(India in this case). The resulting date can be correlated with important cricketing events related to the respective team on that day, leading to increased social media presence.

#Precondition: Only Team with Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Also, posts under r/Cricket channel were alone considered for evaluation.

#Steps: Find the posts related to players, who belong to the team given in input and sum up the reddit posts per day, displaying the per day breakdown for the team

#Input: Any cricket team available under ‘team’ table.

#Response: Return the per day breakdown of number of reddit posts for the input team.

*select DATE(post.Created\_Time) as "Post\_Date", count(post.Post\_ID) 'Number\_Of\_Posts'*

*from cricket\_socialmediaanalysis.redditpost post*

*inner join cricket\_socialmediaanalysis.player p*

*on post.Player\_ID = p.Player\_ID*

*inner join cricket\_socialmediaanalysis.team te*

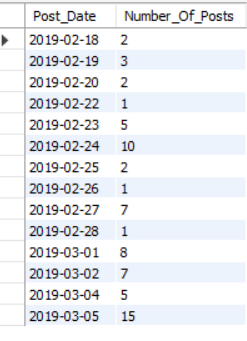
*on te.Team\_ID = p.Team\_ID*

*where te.Team\_Name = "India"*

*group by Post\_Date*

*order by Post\_Date;*

Result:



5.

#Use case (5)

#Description: Displaying number of hashtags used in tweets against each player, thereby assessing the social media popularity of the player from the stored data.

#Precondition: Only Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: Find the hashtags related to tweets , which are again related to players, thereby summing the number of hashtags per player.

#Response: Return the list of players with their related hashtag count from the stored data..

*select p.Player\_Name, count(ht.Name) as "Number\_Of\_Hashtags"*

*from cricket\_socialmediaanalysis.twitterhashtag ht*

*inner join cricket\_socialmediaanalysis.tweet t*

*on t.Tweet\_ID = ht.Tweet\_ID*

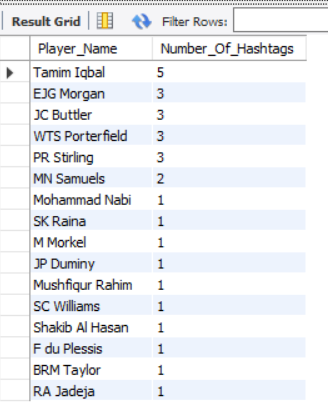
*inner join cricket\_socialmediaanalysis.player p*

*on p.Player\_ID = t.Player\_ID*

*group by p.Player\_Name*

*order by Number\_Of\_Hashtags DESC;*

Result:



6.

#Use case (6)

#Description: Displaying the list of players and their respective teams, who have a batting average greater than 30 and bowling economy rate less than 5, so that they can be classified for all round performance and put up in all round rankings.

#Precondition: Only Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: Find the players who have batting average greater than 30 in a subquery. Then use the list to find the players who have bowling economy rate less than 5.0 in another subquery. Then use this restricted list to find which team the player belongs to.

#Response: Return the list of all round players and their team name.

*select p.Player\_Name,t.Team\_Name*

*from cricket\_socialmediaanalysis.player p*

*inner join cricket\_socialmediaanalysis.team t*

*on p.Team\_ID = t.Team\_ID*

*where p.Player\_ID in*

*(select bowlst.Player\_ID*

*from cricket\_socialmediaanalysis.playerbowlingstatistics bowlst*

*where bowlst.Economy\_Rate <= 5.0 and*

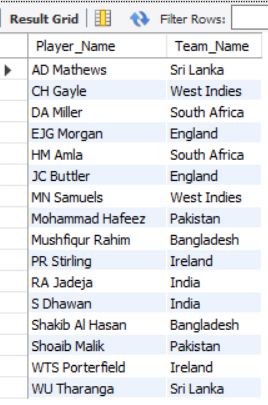
*bowlst.Player\_ID in*

*(select batst.Player\_ID*

*from cricket\_socialmediaanalysis.playerbattingstatistics batst*

*where batst.Batting\_Average > 30));*

Result:



7.

#Use case (7)

#Description: Displaying the list of players from all teams, who have scored more runs than all players of a particular team(Australia is used in this case).

#Precondition: Only Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: First total runs scored by Australian players is written with a subquery and is compared with total runs scored by all players of other teams using the ‘ALL’ clause.

#Response: Return the list of all players, who have score more runs than every other player in Australia.

*select p.Player\_Name,batst.Runs\_Scored, t.Team\_Name*

*from cricket\_socialmediaanalysis.playerbattingstatistics batst*

*inner join cricket\_socialmediaanalysis.player p*

*on p.Player\_ID = batst.Player\_ID*

*inner join cricket\_socialmediaanalysis. team t*

*on p.Team\_ID = t.Team\_ID*

*where batst.Runs\_Scored > all*

*(select batst.Runs\_Scored*

*from cricket\_socialmediaanalysis.playerbattingstatistics batst*

*inner join cricket\_socialmediaanalysis.player p*

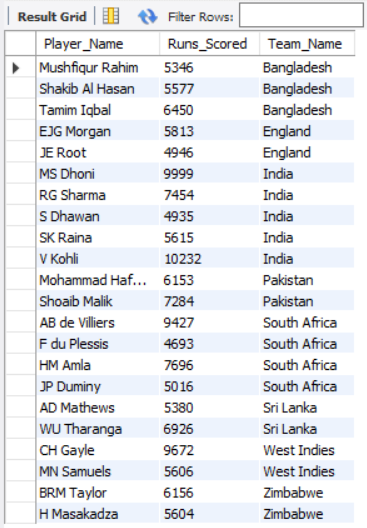
*on p.Player\_ID = batst.Player\_ID*

*inner join cricket\_socialmediaanalysis. team t*

*on p.Team\_ID = t.Team\_ID*

*and t.Team\_Name = "Australia");*

Result:



8.

#Use case (8)

#Description:

#Picking the teams with strong bowling figures by evaluating the team with good bowling average first and then considering the good bowlers within it with significant bowling average.

#Precondition: Only Players, who are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: First teams with good bowling average are listed though a subquery and is then used for comparison using ‘Exists’ to find the players within the team with good bowling average.

#Response: Return the list of all players, their respective bowling average and their team name.

*select Player\_Name, bowst.Bowling\_Average as "Player\_Bowling\_Average"*

*,t.Team\_Name*

*#count(p.Player\_ID) as "Number\_Of\_Players"*

*from cricket\_socialmediaanalysis.player p*

*inner join cricket\_socialmediaanalysis.team t*

*on p.Team\_ID = t.Team\_ID*

*inner join cricket\_socialmediaanalysis.playerbowlingstatistics bowst*

*on bowst.Player\_ID = p.Player\_ID*

*where bowst.Bowling\_Average <= 30.0 and*

*bowst.Bowling\_Average != 0 and*

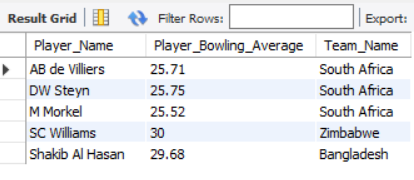
*EXISTS*

*(select \**

*from cricket\_socialmediaanalysis.teambowlingstatistics tbowst*

*where tbowst.Team\_ID = t.Team\_ID and tbowst.Avg\_Runs\_Per\_Wicket\_Bowling <= 40.0 );*

Result:



9.

#Use case (9)

#Description: Displaying the list of players and their respective teams, who had been in the team for more than 15 years for the teams to consider swapping chances for younger players.

#Precondition: Only Players, who have playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: Numerical operation(difference) is performed between ‘Career End’ and ‘Career Start; columns of Player table and is connected with team table to extract the team information

#Response: Return the list of all players, their tenure in the team and their team name.

*select p.Player\_Name, (p.Career\_End - p. Career\_Start ) as "Tenure", t.Team\_Name*

*from cricket\_socialmediaanalysis.player p*

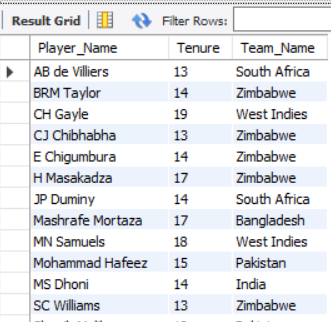
*inner join cricket\_socialmediaanalysis.team t*

*on p.Team\_ID = t.Team\_ID*

*where (p.Career\_End - p. Career\_Start ) > 12*

*order by p.Player\_Name;*

Result:



10.

#Use case (10)

#Description: Listing the best fielder from every team with their field dismissals figure.

#Precondition: Only Players, who have are playing in present time(Career End = 2018, as per database) and have played greater than or equal to 100 matches are considered.

#Steps: ‘Max()’ aggregate function is applied on field dismissals value of players and the best player is from every country is picked up group by function.

#Response: Return the list of all players, their field dismissals figure and their team name.

*select p.Player\_Name, max(pfs.Fielding\_Dismissals\_Made) as "Field\_Dismissals", t.Team\_Name*

*from cricket\_socialmediaanalysis.player p*

*inner join cricket\_socialmediaanalysis.playerfieldingstatistics pfs*

*on p.Player\_ID = pfs.Player\_ID*

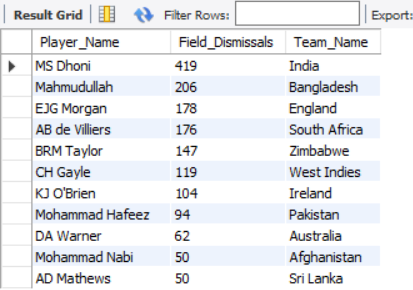
*inner join cricket\_socialmediaanalysis.team t*

*on p.Team\_ID = t.Team\_ID*

*group by t.Team\_Name*

*order by Field\_Dismissals DESC;*

Result:



**Audit Validity-**

The whole dataset had more information than required for the proposed conceptual database model. To fit in to the database schema, the dataset was reformatted. Some of the bowling figures were in ‘-‘, which were replaced by 0. A few of the columns have been renamed for the purpose of making it simple and understandable.

**Audit Completeness**

The dataset is up to date, needs no more cleaning and matches the quality of the real world data. The data was cross verified with ESPN criconfo.

**Audit Consistence/Uniformity:**

The possible range of the dataset is covered from the new resultant data set. The data does not have any null values, limitations, negative values.

Also the values appear to remain constant with what is represented in ESPN cricinfo.

**Conclusion:**

In the assignment, all of the CSV tables were populated are populated with real-word data collected from three sources: Twitter API, Reddit API and kaggle. A conceptual schema was designed for cricket domain involving companies, producers, consumers and subsequently a physical model was also established.

The collected data was reformatted, cleaned to fit the database schema.

Citations and References:

1. https://www.geeksforgeeks.org

2. https://github.com/nikbearbrown/INFO\_6210

3.google

4.kaggle