**Report on Project**

**Database Design issues from Giant Table:**

1. Database is not Normalized as there are Functional Dependencies which are not well formed. So, to convert Database into 3NF we must make separate tables. This will eliminate Data duplication.
2. The table is storing everything in excel in one format where the it is storing User and Tweet information in one row where values are repeated which is (duplication) in Data. Data will have insertion, deletion and update anomalies as doing query’s as it will create more redundancy which will cause inconsistency in data. Furthermore, deleting something may delete the Whole Tweet ID or User data which we Still need i.e. (necessary or important Data). Hence, we must divide the Data into separate tables.

E.g.: Deleting Tweet\_ID or User\_ID may delete the row itself.

1. In Giant Table the Tweet ID’s and User ID’s are repeating in different columns of excel. (i.e. Redundancy in data).
2. Status ID is never mentioned in any table and directly Taken when user Retweets with Tweet with Username. So, we can say that Data is dirty as status ID is available for only some users. So, we can eliminate the status ID or just keep as it is in another table with other attributes. Therefore, we can assume that data is Missed by human errors or no particular data is present for this Part.
3. Retweet Tweet ID table has Null values for those where no Tweet is Retweeted. So, this is causing Data Storage issues (Memory Wastage).

1. Hashtags are not stored in easy format and it contains null values where the maximum limit is 6. So, it stores null values when users use limited Hashtags. Again, this can cause memory Wastage. Furthermore, it is also hard for someone to understand the data in first go.
2. Null Values in User Description, Location so this must be stored in different table with user\_id.
3. Huge Flaw in this table is that it stores everything in one cell so, when we will run queries on data it can cause problems for insert, update and may give errors and not accurate query answers. It may also delete the whole cell for a query where we need to particularly

**Observations on Hashtags:**

Hashtags are stored very differently, when user enters only one hashtag it is stored in one column where for every user who enters only one Hashtag it is stored in that column.

So, like this it is stored for every user up till 6 Hashtags (where 6 is limit).

So, we need to remove null values from the table and put them in one table.

Functional Dependencies:

According to Database these assumptions can be made by looking at Data.

Created\_at, text, tweet\_source, tweet\_count is dependent on Tweet\_ID as Tweet\_ID can describe all these attributes and this table then will have no duplicate values by making Primary key.

*Tweet ID -> Created\_at, text, tweet\_source, tweet\_count, retweet\_of\_tweet\_id, reply\_to\_screen\_name, reply\_to\_status\_id, hashtags.*

For User we will make Same table where User can describe some attributes.

User\_ID can describe Username, user\_screen\_name, user\_followers\_count, user\_friends\_count, user\_lang, user\_status\_count, user\_created\_at as from User ID we can describe all these things. We will make User\_ID Primary key for this.

*User\_ID -> Username, user\_screen\_name, user\_followers\_count, user\_friends\_count, user\_lang, user\_status\_count, user\_created\_at, location, utc\_offset, timezone, description.*

Some of the attributes contain null values so we will take them in another table with Dependant value.

i.e. Tweet\_id will become foreign key for this new table and it will contain attribute which has null values in this Table.

**No Transitive Dependency:**

According to each table now each attribute is dependent on only 1 attribute which is Primary Key.

In Tweet Table now each attribute is dependent on Tweet ID. There were some attributes which are also dependent on other attributes including primary key so we made separate table for them to make them in 3NF.

**New Database Design:**

User ID and Tweet ID are only primary key in two different tables which are linked in one table. And all transitive dependency will be stored in different tables.

According to assumptions made we can make separate tables where there are transitive dependencies and null values are present.

e.g.

For now, every table we will have no transitive functional dependencies as each attribute will be dependent on primary key of table.

New table for Tweet, Retweet and Reply will be made as follows to eliminate null values and Transitive dependency.

*Tweet -> created\_at,text,tweet\_id,tweet\_source,retweet\_count*

*Table(Retweet) -> tweet\_id, retweed\_of\_tweet\_ID*

*Table(Reply) -> tweet\_id, in\_reply\_to\_screen\_name,in\_reply\_to\_status\_id,in\_reply\_to\_user\_id*

Same table will be made for users:

New table for location as user\_location is dependent on user ID and null values will be eliminated by taking it another table. So, it will create table for user\_id where user\_location is present.

Now each attribute is dependent on User\_id in each table

*Table[User] -> user\_id,user\_name,user\_screen\_name,user\_followers\_count,user\_friends\_count,user\_lang,user\_status\_count,user\_created\_at*

*Table[Location] ->user\_id, user\_location*

*Table(Time) -> user\_id, user\_utc\_offset, user\_time\_zone*

*Table(Descr) -> user\_id, user\_description*

And for Final we connect user and tweet with their key in another table as both acting as Foreign key will make this all in 3NF.

*Table[Tweet\_User] -> tweet\_id,user\_id*

FK: twee\_id and user\_id

**As Conclusion:**

Now table is in 3nf as each attribute in every table is dependent on only primary key and not on any other attribute. And table is easy to understand. So, table is now normalized and queries can run smoothly without deleting any important data.