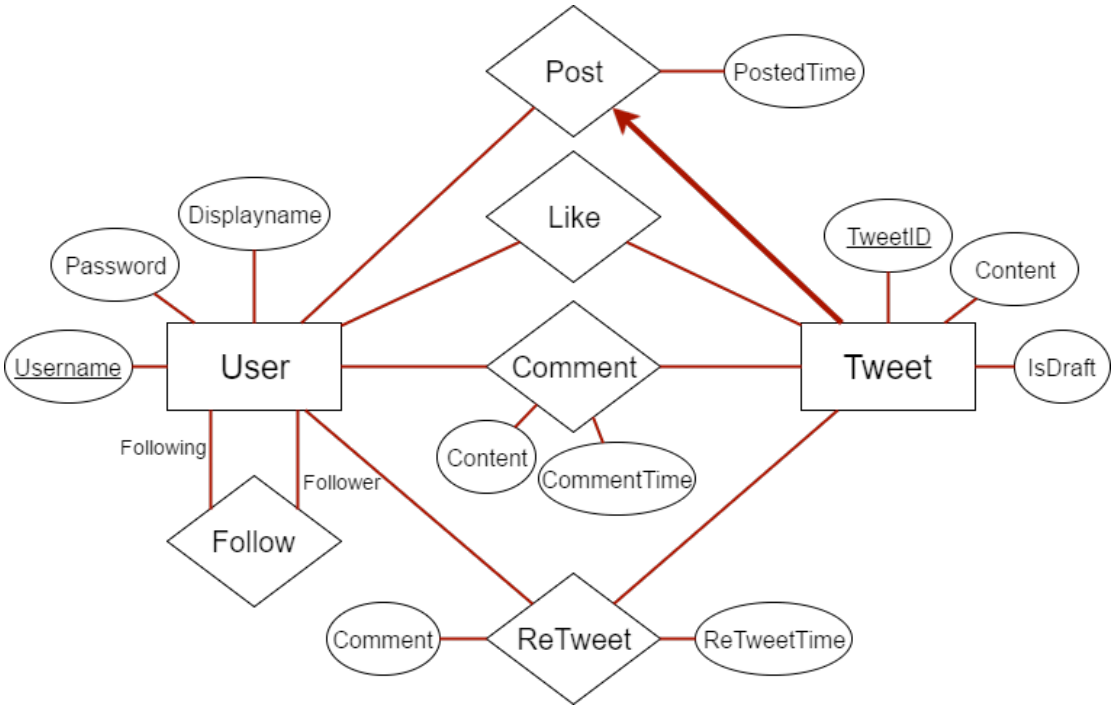
ER model

There are 2 entities and 5 relations in the model, the rectangle is the entity set, the oval is the attributes which includes the key, the diamond is the relationship.

Only User-Post-Tweet is “One to Many” relationship, it is also the participation relationship, so it is the bold line with an arrow. The arrow initiated from the side denoted as the “Many” and point to the One side.

It is participation relationship since every Tweet must be belonged to one User, also, One user can have more than one Tweet, but one Tweet can only have one user to post, so it is many(Twitter) to the one(User) relationship, the others are all “many to many relationship”.



Relational database schema

Map

So, we get the below relations:

The underlined attributes are the candidate key.

The Primary key is used for indexing, it must be not null.

User (Username, Displayname, Password);

Tweet (TweetID, Username, PostedTime, Content, IsDraft);

Follow (Username, Follower);

Like (Username, TweetID);
Comment (CommentID, Username, TweetID, CommentTime, Content);
ReTweet (ReTweetID, Username, TweetID, Comment, ReTweetTime);

Satisfying the requirements:

(1) All mentioned information in 3.1 are stored in the entity set of “User”. The username is associated with an unique email address of the user and is set to be primary key because this is not null and easy to index, The display name is set to be unique as well, each username corresponds to only one display name.

(2) All mentioned information in 3.2 are stored in relation “Tweet”. We use a serial number type to identify each unique tweet. We use “IsDraft” as te Boolean to distinguish post or un-posted tweet.

(3) As for the Follow relationship, each

user can follow many users. So it is the many to many relationship, so We use a query joining “Follow” and “Tweet” to get all posted tweets from user’s following users in the order from the newest one to the oldest one as the news feed.

We use relation as the “Like”, “Comment” and “ReTweet” to represent User-Tweet relationships. All of these 3 relationships are “Many to Many”. For “Comment” and “ReTweet”, we add “ID” attribute because one user may have many comments or re-tweets of one tweet.

As for the “Unfollow”, I didn’t define it since to reduce the redundancy. As long as it is not in the table then that is the unfollow relationship.

Further discussion

Advantages

(1) All necessary information included.

(2) Reduced the data redundancy, quite simple and clean. Easy to query.
(3) Clear data structure.

Disadvantages

(1) Data, such as number of “likes” of a tweet, are not stored, so we still need query them each time, this will increase the time complexity.
(2) The follow and unfollow histories are not stored in the database, we can only see the current following relationship.