University of Puerto Rico - Mayagüez
Electrical and Computer Engineering
Department

ICOM 5016/ CIIC 4060
Introduction to Database Systems

Term Project – Backend System for Twitter-Style Service
Phase II
Initial System Implementation

Andrea C. Mirada Acevedo
Emmanuel Nieves González
Jean C. Rivera Hernández
Carlos M. Santiago López
Monday April 12, 2021

Mapping ER to Tables:

Entities:

- **Users**
  Since Users is as entity a table Users is created that contains parameters: uid which is users unique id, first_name, last_name, email, username and password.
  -create table Users(uid serial primary key , first_name varchar(20) , last_name varchar(20),email char(54),username char(15),password char(15));
- **Messages**
  Since Messages is an entity, a table is created with parameters:
  Mid which is messages unique id, message which is the message that is written by the user.
  - create table Messages(mid serial primary key, message varchar(20),uid integer references Users(uid), replyingto integer references Messages(mid), sharing integer references Messages(mid), isshare boolean , isreply boolean);
  In this table messages it is specified if a message is a message or a reply or a share. Since Reply and share is treated as a new post, where post is a message. When a new message is posted the parameters mid, uid and message are posted, isshare and isreply are specified as false while sharing and replyingto are null. When a reply is posted the same parameters are written with a new mid, the uid of who is replying, the message of the reply, the mid of the message that is replied that is replyingto and isreply is true since it’s a reply. When a share is posted a new mid is given, uid for the user that is sharing and sharingto that references mid of the message that is shared and also the attribute is share is true, all the other parameters such as message, repliyingto are null and isreply is false.

Relationships:

<u>One to Many:</u>

- The relation Posts is a one to many relationship between Users(one side) and Messages(many side), where a user posts a message and even if the same user posts the same message their Id's are different. Since post/messages cannot exist if they are no messages that means that messages has total participation. In this case a table is not created, but the one side primary key uid from Users is passed to the Messages table as a foreign key. The table is shown in Messages**.**
- The relation Shares is one to many self-relationship of messages since share is like a new post from a message, where post is seen in this case as a message. Since is one to many there is a reference as foreign key from messages mid with name sharing to know with message is the one that is shared and a Boolean attribute called isshare to let know in the same table of messages if the message is a share or not.
- The relation Reply is one to many self-relationship of messages since reply is like a new message within a message. Since is one to many there is a reference as foreign key from messages mid with name replyingto to know with message is the one that is replied and a Boolean attribute called isreply to let know in the same table of messages if the message is a reply or not.

Many to Many:

- The relation Follows is a many to many self-relationship of Users since a user follows another user. Since is many to many a table is created with parameters fid it's unique id, followingid that refers to Users uid as the person who is following and the other parameter followerid which references to Users uid too, but this one to signify the one who is the follower.
  - create table Follows(fid serial primary key, followingid integer references Users(uid), followerid integer references Users(uid));
- The relation Blocks is a many to many self-relationship of Users similar to Users since a user blocks another user. A table is created with parameters: bid which is the unique id, blockingid which

references to Users ui in the meaning the one that will be blocked and uid which references to Users uid referencing the one that is blocking.
- create table Blocks(bid serial primary key, blockingid integer references Users(uid), uid integer references Users(uid));

- The relation Reacts is a many to many relation, therefore a table was created. The parameters were rid which is the unique key, uid that refers to Users uid to get track of who gave the reaction, the mid that refers to Messages mid to get track of the message and isLiked a boolean attribute that if it’s true is a like, if it’s false is a unlike.
  - create table Reacts(rid serial primary key, uid integer references Users(uid), mid integer references Messages(mid), isLiked boolean);

Changes to ER Diagrams:

- Some attributes like join_date of Users and mdate for Messages were removed, because there are not of a lot of use for this implementation.
- Reacts a relation between Users and Messages was changed from one to many to many to many, since a user can react to a message and a message can have different reactions from other users.
- Shares and reply were changed to one to many self-relationship since they are treated as post where in this case posts like tweets or retweets and replies are messages.
- The Explanation of the tables done were the ones used for this phase 2. Things that were expected for phase 3 were left out.