Virginia Tech. **CS 4604 – Introduction to DBMS**
Computer Science **Fall 2018, Prakash**

Project Assignment 2

(due Nov 1, 2018, 9:30am, in class—hard-copy please)

Reminders:

a. Out of 100 points. Contains 6 pages.
b. Rough time-estimates: 7-9 hours. Has to be done in groups.
c. Please type your answers. Illegible handwriting may get no points, at the discretion of the grader. Only drawings may be hand-drawn, as long as they are neat and legible.
d. There could be more than one correct answer. We shall accept them all.
e. Whenever you are making an assumption, please state it clearly.
f. Unless otherwise mentioned, you may use any SQL operator seen in class. Feel free to create intermediate views for SQL.
g. Important:
    a. For E/R diagrams, use only the style and notation given in the lecture slides.
    b. A useful tool for creating E/R diagrams: http://logicnet.dk/DiagramDesigner/. You may have to manually draw-in some things though (like adding proper constraints etc.). There are other such programs too.

Q1. Designing the schema [40 points]

We now want to design an appropriate schema for our TweetVT system. We will work with user information, tweets they are created, hashtags, urls, user-mentions in their tweets and the location.

- The system can have many users. Every user has an "id" that should be unique. Other important information about users that we want to have: name, age, screen_name, followers_count, friends_count, statuses_count and language.
- Users can have profile image; so a profile_image_url will be kept.
- We also want to track the date a user was created.
- Just like Twitter, a user can follow other users; therefore, every user can have multiple followers and follow multiple users.
- The date when a user starts following some other user should also be recorded.
- Users can create tweets (again like Twitter). When a user tweets something, some information about the tweet should be recorded. Tweets should have a unique "id", "text" (which is the content of tweet), the date it was created and the language it was written in.
- A tweet when posted, can be favorited by other users. So, we would like to know if a tweet is favorited or not (favorite value could be 1 or 0), and the number of times it is favorited (the favorite_count).

- Users can retweet tweets created by other users. So, we should record if the tweet was retweeted or not (retweeted value is 1 if it is retweeted by others and 0 otherwise). And also retweet count will be recorded (retweet_count).
- If a tweet is a retweet of another tweet, we are interested in knowing the parent tweet. (for example, if t2 is a retweet of t1; t1 will be the parent of t2. We should record parent tweets if it is a retweet.)
- Tweets can contain hashtags; hashtags are identified by text. Each hashtag can be in several tweets. And tweets can have multiple hashtags.
- Users can also be mentioned in the tweets. We want to track which user(s) were mentioned in which tweet. In addition, the word position of a user-mention in the text should be recorded.
- Tweets could have urls as well. We want to keep both the shortened form and long form of the urls.
- We want to track location information as well. Each location has an id (unique), its latitude and longitude, and the corresponding city, country.
- Every user should define his/her location.
- Tweets are created at a location. So, we are interested to know the location of each tweet.

Q1.1 (15 points) Draw an ER diagram for this database. Make sure to indicate primary keys, cardinality constraints, weak entities (if any), and participation constraints. There might be extra constraints which cannot be captured by the E/R diagram, make sure you mention them below the diagram. List any assumptions you make in the process.
Hint: The E/R diagram should contain at least ~4 (may be more) entities; otherwise it is not of sufficient complexity for a CS 4604 project.

Q1.2 (10 points) For each entity set and relationship, write a short description in plain English of what it represents or models. One or two sentences per entity set and relationship is enough. These descriptions are primarily to help us understand that you are modeling the TweetVT database correctly.

Q1.3 (15 points) Translate the ER diagram in Q2.1 into relational database tables (i.e. give the SQL DDL statements). Make sure that the translation captures key constraints (primary keys and foreign keys if applicable) and participation constraints in the ER diagram. Identify constraints, if any, that you are not able to capture.

Common Mistakes to avoid in design:

1. Modeling a database administrator explicitly in your E/R diagram. The DBMS usually has its own internal representation for administrators.

2. Missing arrows or rounded arrows in a many-one and/or a one-one relationship.
3. Missing arrows from a weak set to the set(s) that provide its key attribute(s).
4. Using inheritance when there is no "ISA" connection between two sets.
5. Forgetting that when entity set *B* inherits from entity set *A*, *B* inherits set *A*, *B* inherits **everything** that *A* has. In addition, *B* can define its own attributes of its own. Therefore, there is no need to repeat all the attributes/relationships that *A* has again for *B*.
6. "Cooking up" multi-way relationships, weak entity sets, or inheritance when they are not needed.
7. Forgetting to underline key attributes in the E/R model.
8. Repeating (reusing) names for different entity sets or for different relationships within the same entity set, i.e., using the same name to denote two different things.
9. When converting a multiway relationship to many two-way relationships using a connecting entity set, forgetting to introduce many-one relationships!

Q2. Refining the schema [30 points]

In this question we want to refine relational schemas using functional dependencies and normalization. You should also be able see how our original 'recipe' for converting ER-diagrams to relational schemas does in fact do a good job of getting good schemas (assuming your ER diagram is correct of course!).

Let's call the schema you get in Q1.3 as `Schema 1`.

Q2.1 (10 points) Using the description given in Q1, for *each* of the relations in `Schema 1`, list all completely non-trivial Functional Dependencies (FDs) that apply to that relation. Only list FDs that have one attribute on the right hand side. It is enough to list only the FDs in a minimal basis.
Hint: You essentially need to check each line in the description, if it results in any FD(s) for any relation.

Q2.2 (10 points) Using the FDs you got in Q2.1, for each relation in `Schema 1`, write down if it is in BCNF. Explain each answer. If any of relations are not in BCNF, decompose and normalize them to BCNF.

Q2.3 (5 points) Are the resulting relations from Q2.2 in 3NF?

Q2.4 (5 points) List (any) differences of the schema you get in Q2.2 above from `Schema 1`.

Q3. Creating the database [30 points]

We now want to create the database for using it in the final assignment.

Let's call the schema you get in Q2.2 as `Schema 2`.

We have stored all the data you will use in your project in the following **two (2)** tables (`Schema 3,` with primary key underlined) on our PostgreSQL server. Note that `Schema 3` obviously seems pretty bad.

`Schema 3` **Tweeter** (u_id, u_name, u_screen_name,u_location_id, u_location_longitude, u_location_latitude, u_location_city, u_location_country, u_followers_count, u_friends_count, u_created_at, u_statuses_count, u_lang, u_profile_image_url, u_age, t_id, t_created_at, t_text, t_location_id, t_location_longitude, t_location_latitude, t_location_city, t_location_country, t_retweet_count, t_favorite_count, t_lang, t_retweeted, t_favorited, t_parent_id, h_text, url_short, expanded_url, um_user_id, um_position)

Follower_Followee (Follower_id, Followee_id, follower_age, followee_age, date)

The description of fields:
u_id: user id,
u_name: name of user,
u_screen_name: screen name or user name of user,
u_location_id: id of location that user set in his/her profile,
u_location_longitude: longitude of location that user set in his/her profile,
u_location_latitude: latitude of location that user set in his/her profile,
u_location_city: city of location that user set in his/her profile,
u_location_country: longitude of location that user set in his/her profile,
u_followers_count: number of followers of user,
u_friends_count: number of friends (followees) of user,
u_created_at: the time a user create profile in twitter,
u_statuses_count: the number of statuses user posts,
u_lang: a language a user defined in his/her profile,
u_profile_image_url: the url address of profile picture of user,
u_age: age of user,
t_id: the tweet id,
t_created_at: the time a tweet is posted (created) by user,
t_text: the tweet text,
t_location_id: id of location where a tweet posted in,
t_location_longitude: longitude of location where a tweet posted in,
t_location_latitude: latitude of location where a tweet posted in,

t_location_city: city of location where a tweet posted in,
t_location_country: country of location where a tweet posted in,
t_retweet_count: the number of time this tweet is retweeted,
t_favorite_count: the number of times this tweet is favorited,
t_lang: language of tweet,
t_retweeted: if this tweet is retweeted or not,
t_favorited: if this tweet is favorite or not,
t_parent_id: the id of parent tweet (if the it is not a retweet of other tweets, the value will be null),
h_text: text of hashtag in tweet,
url_short: short url in tweet,
expanded_url: expanded url in tweet,
um_user_id: id of a user which is mentioned in tweet,
um_position: the position where the user-mentioned happens in the tweet.

We have stored the project dataset in Schema 3 form in the 'cs4604f18_project' database on the server. Similar to HW4, this time you can download all of these tables to your *group database* by using the following at the *command-prompt (NOT at the psql prompt)* of cs4604.cs.vt.edu (i.e. run this on the first prompt you get after ssh-ing to the server):

"pg_dump -U YOUR-PID cs4604f18_project | psql -d your-group_database"

Note: your-group_database name is the name you defined for your group with 'db' at the end, lowercase and without any space. For example, If your group name is: "The Twitter Guys", the name of your group database will be: "thetwitterguysdb". If you are still in doubt please contact the TAs.

Q3.1 (5 points) Formally show why Schema 3 is bad i.e. for each of the two relations write whether it is in BCNF or 3NF (use the set of FDs you wrote down in Q2.1).
Hint: You do not need to go through all the FDs to arrive at the conclusion.

Q3.2 (25 points) First create the TweetVT refined schema (`Schema 2`) in the PGSQL server. Then we have to insert the right tuples into them: but we want you to insert the data *from* the 2 tables in `Schema 3`.

For each table in `Schema 2`, first show your DDL statements (CREATE TABLE and INSERT INTO), and then output only *the total number* of tuples for each table.

Example: Suppose your Schema 2 has a table FF which stores all the follower-followee pairs. Clearly it should get data only from the Follower_Followee table in Schema 3. So show the following in your answers:

1. Create the table FF:

   CREATE TABLE FF (
   date character varying(100),
   Follower_id character varying(100),
   Followee_id character varying(100),
   primary key (Follower_id, Followee_id);

   (note that this create table is incomplete and just an example, as we haven't shown foreign keys)

2. Insert proper data; (this part is easy in this example)

   INSERT INTO FF(date,Follower_id, Followee_id)
   (SELECT date , follower_id, followee_id FROM Follower_Followee);

3. Count the number of tuples;

   SELECT COUNT(*) FROM FF;
   Output: 15766

Note 1: In order to fill out the location table, you should use UNION of location information for both users (u_location_id, u_location_longitude, u_location_latitude,…) and tweets (t_location_id, t_location_longitude, t_location_latitude,…) in the **Tweeter** table.

Note 2: please note that some fields of the two tables may be NULLs for some rows; for example, for tweets, t_Parent_id would be NULL if it is not a retweet of other tweets. Similarly not all tweets may not have hashtag, user_mention or url. So consider the NULL values for these fields while creating the smaller tables.

Important: After creating your tables, please delete *your copy* of the `Schema 3` tables in your group database to save space. Also, please do not use your personal database for the project data (we won't have enough space if everyone copies data to their personal databases).