EECS 484 F13 Project 1
Database Design for Social Network Data

Overview

In Project 1, you will design a relational database for storing information about your Fakebook social network. You will begin with a detailed description of the content. Then, you will need to systematically go through the conceptual and logical database design process you learned about in class.

Part 1: ER Design

As a starting point, we have done the initial "requirements analysis" for you. The following is a brief description of the data that you will store in your database. (In real life, you would probably begin with much fuzzier information.)

User Information

There can be an unlimited number of users. Each user has the following information:

1.1 Profile information

This includes the following attributes: user ID, first name, last name, year of birth, month of birth, date of birth, gender.

1.2 Hometown Location

A user's hometown includes the following attributes: city, state, country. Each user can have at most one hometown.

1.3 Current Location

Exactly the same attributes as hometown location. Users could have primary current locations, secondary current locations, tertiary current locations and so on so forth (Most of the users would only have one current locations, but there are few users have multiple current locations).

1.4 Education History

Education history is for college programs and above. A user could have participated in multiple educational programs, and each one will have the following attributes: name of the institution (e.g., University of Michigan), year of graduation, concentration (e.g., CS, EE, etc.), degree (e.g., BS, MS, PhD, etc.).

1.5 Friendship information

Each user can have any number of friends. Each friend must also be a Fakebook user. Each user must have at least one friend, and he/she can not be friend with himself/herself.

Photos
"Photos" is an important Fakebook application. Each photo has the following associated information:

2.1 Album information

Each photo MUST belong to exactly one album. An album has the following attributes:
album_ID, owner_ID (this refers to the owner's Fakebook ID and albums can be shared between users, so an album could have more than one owner_ID), album_name, cover_photo_ID (this refers to a photo ID and no two albums can have the same cover), album_created_time, album_modified_time , album_link and album_visibility.

2.2 Other information

Each photo has the following attributes: photo_ID, photo_caption, photo_created_time, photo_modified_time, and photo_ link.

Photo Tags
A photo tag identifies a Fakebook user in a photo. It has the following associated attributes:

3.1 Tag subject

tag_subject_id (this refers to a Fakebook user ID)

3.2 Tag coordinates

tag_x_coordinate and tag_y_coordinate

3.3 Time created

tag_created_time

Note that there can be multiple tags at exactly the same (x, y) location. However, there can be only ONE tag for each subject in the photo; Fakebook doesn't allow multiple tags for the same subject in a single photo. For example, you cannot tag Lady Gaga twice in a photo, even if she appears at two different locations (whatever that means).

Events
"Events" is another useful Fakebook feature.

4.1 Basic event information

event_ID, event_creator_id (Fakebook user who created the event), event_name, event_tagline, event_description, event_host (this is a string, not a fakebook user), event_type, event_subtype, event_location, event_city, event_state, event_country, event_start_time, and event_end_time

4.2 Event participants

Participants in an event must be Fakebook users. Each participant must have a confirmation status value (attending, declined, unsure, or not-replied). The sample data does not have information on Event Participants at present. You can leave the information on Participants empty.

Task for Part 1

Your task in Part 1 is to perform "Conceptual Database Design" using ER Diagrams. There are many ER variants, but for this project, we expect you to use the conventions from the textbook and lecture.

Hints for Part 1

You need to identify the entity sets and relationship sets in a reasonable way. We expect there to be multiple correct solutions (remember that ER design is subjective!). Your goal should be to capture the given information using ER constructs that you have learned about in class (participation constraints, key constraints, weak entities, ISA hierarchies and aggregation) as necessary.

For the entity set, relationship set and attribute names, you can use the ones we have provided here, or you may also choose your own names, as long as they are intuitive and unambiguous.

Before you get started, you should also read Part 3 to understand the specifics of the data.

Part 2: Logical Database Design

For the second part of the project, your task is to convert your ER diagrams into relational tables. You are required to write SQL DDL statements for this part. You should turn in two files:

1. createTables.sql
2. dropTables.sql

Hints for Part 2

You should capture as many constraints from your ER diagrams as possible in your createTables.sql file. In your dropTables.sql, you should write the DROP TABLE statements necessary to destroy the tables you have created.

Using Oracle SQL*Plus, you can run your .sql files with the following commands:

```
sqlplus <accountName>/<password> @ dropTables.sql
```

```
sqlplus <accountName>/<password> @ createTables.sql
```

Part 3: Getting Fakebook Data

The next part of the project is actually populating your database. For this part, we will provide you with some fake data for Fakebook, our social network website. The following describes how to log into your Oracle account and how to access the fake data.

Logging in to your Oracle Account

First, connect to login.engin.umich.edu using SSH with your UMich account (uniqname and password).

Then execute:

```
module load eecs484/f13
sqlplus
```

And enter the user name and password for your Oracle account to login.

After that, you can type SQL commands to interact with the database system.

To disconnect from Oracle you can execute:

```
EXIT
```

Try this early! If you have trouble accessing your Oracle account, please speak to the GSI.

General Hints on Using SQLPlus Effectively

We have posted some hints on SQLPlus for this Project effectively at:

https://docs.google.com/document/d/1NI1bylNwu1Hh5vkYlUb80i-cEWF_MzG5u1IPyhs5aA4/edit?usp=sharing

The above document is not a tutorial on sqlplus and may not make sense right away. Refer to it as you run into problems during the project. It may have the answers.

Fake social network data

Everyone will have access to a fake data set, which is designed to emulate a social network dataset. The fake data includes the following five tables:

```
PUBLIC_USER_INFORMATION
PUBLIC_ARE_FRIENDS
PUBLIC_PHOTO_INFORMATION
PUBLIC_TAG_INFORMATION
PUBLIC_EVENT_INFORMATION
```

These tables are stored in the GSI's account (unnamed). You can access the public tables for the fake data using GSI's account name (unnamed). For example, to access the PUBLIC_USER_INFORMATION table, you need to refer to the table name as UNNAMED.PUBLIC_USER_INFORMATION. You can copy the data into your own account with the following command:

```
CREATE TABLE NEW_TABLE_NAME AS (SELECT * FROM UNNAMED.TABLE_NAME);
```

The data will then be stored into your personal Oracle space. You can login to SQLPlus to browse the data.

Here are some basic commands to browse your data.

- View all the existing tables:

  ```
  SELECT TABLE_NAME FROM USER_TABLES;
  ```

- View the schema of a table:

  ```
  DESC TABLE_NAME;
  ```

- Browse all the data in a table:

  ```
  SELECT * FROM TABLE_NAME;
  ```

- Browse the first n rows in a table:

  ```
  SELECT * FROM TABLE_NAME WHERE ROWNUM<N;
  ```

To change the output format of table columns you can use the “COLUMN” command with the “FORMAT” option. For example, the following two commands can be used to display the first 20 characters of USER1_ID and USER2_ID.

```
COLUMN USER1_ID FORMAT A20;

COLUMN USER2_ID FORMAT A20;
```

Then, the output of the following “SELECT” statement will be displayed as a table in user-friendlier format.

```
SELECT * FROM ARE_FRIENDS WHERE ROWNUM < 3;
```

Fake data raw schema

The fake data tables we provide actually give you some hints on the previous parts of the assignment. However, these tables are highly “denormalized” (poorly designed), and without any table constraints.

As mentioned earlier, the table names are:

```
PUBLIC_USER_INFORMATION
PUBLIC_ARE_FRIENDS
```

PUBLIC_PHOTO_INFORMATION
PUBLIC_TAG_INFORMATION
PUBLIC_EVENT_INFORMATION

The fields of those tables are as follows:

PUBLIC_USER_INFORMATION table:

1. USER_ID
   This is the Fakebook unique ID for users
2. FIRST_NAME
   Every user MUST have a first name on file
3. LAST_NAME
   Every user MUST have a last name on file
4. YEAR_OF_BIRTH
   Some user may not provide this information
5. MONTH_OF_BIRTH
   Some user may not provide this information
6. DAY_OF_BIRTH
   Some user may not provide this information
7. GENDER
   Some user may not provide this information
8. HOMETOWN_CITY
   Some user may not provide this information
9. HOMETOWN_STATE
   Some user may not provide this information
10. HOMETOWN_COUNTRY
    Some user may not provide this information
11. CURRENT_CITY
    Some user may not provide this information
12. CURRENT_STATE
    Some user may not provide this information
13. CURRENT_COUNTRY
    Some user may not provide this information
14. INSTITUTION_NAME
    Some user may not provide this information.
    A single person may have studied in multiple institutions (college and above).
15. PROGRAM_YEAR
    Some user may not provide this information.
    A single person may have enrolled in multiple programs.

16. PROGRAM_CONCENTRATION
    Some user may not provide this information.
    This is like a short description of the program.
17. PROGRAM_DEGREE
    Some user may not provide this information.

PUBLIC_ARE_FRIENDS table

1. USER1_ID
2. USER2_ID

Both USER1_ID and USER2_ID refer to the values in the USER_ID field of the USER_INFORMATION table. If two users appear on the same row, it means they are friends; otherwise they are not friends.

PUBLIC_PHOTO_INFORMATION table

1. ALBUM_ID
   ALBUM_ID is the Fakebook unique ID for albums.
2. OWNER_ID
   User ID of the album owner.
3. COVER_PHOTO_ID
   Each album MUST have a cover photo (you can assume there is only one). The values are the Fakebook unique IDs for photos.
4. ALBUM_NAME
5. ALBUM_CREATED_TIME
6. ALBUM_MODIFIED_TIME
7. ALBUM_LINK
   The URL directly to the album
8. ALBUM_VISIBILITY
   It is one of the following values: EVERYONE, FRIENDS_OF_FRIENDS, FRIENDS, MYSELF, CUSTOM
9. PHOTO_ID
   This is the Fakebook unique ID for photos.
10. PHOTO_CAPTION
    An arbitrary string describing the photo.
11. PHOTO_CREATED_TIME
12. PHOTO_MODIFIED_TIME
13. PHOTO_LINK
    The URL directly to the photo

PUBLIC_TAG_INFORMATION table

1. PHOTO_ID
   Unique Id of the corresponding photo

2. TAG_SUBJECT_ID
   Unique Id of the corresponding user
3. TAG_CREATED_TIME
4. TAG_X_COORDINATE
5. TAG_Y_COORDINATE

PUBLIC_EVENT_INFORMATION table

1. EVENT_ID
   This is the Fakebook unique ID for events.
2. EVENT_CREATOR_ID
   Unique Id of the user who created this event
3. EVENT_NAME
4. EVENT_TAGLINE
5. EVENT_DESCRIPTION
6. EVENT_HOST
7. EVENT_TYPE
   Fakebook has a fixed set of event types to choose from a drop-down menu.
8. EVENT_SUBTYPE
   Fakebook has a fixed set of event subtypes to choose from a drop-down menu.
9. EVENT_LOCATION
   User entered arbitrary string. For example, “my backyard”.
10. EVENT_CITY
11. EVENT_STATE
12. EVENT_COUNTRY
13. EVENT_START_TIME
14. EVENT_END_TIME

Part 4: Populate Your Database

For the final part of the project, you will populate your database with Fakebook data we just described. You should turn in the set of SQL statements (DML) to load data from the public tables (**PUBLIC_USER_INFORMATION**, etc.) into your tables. You should put all the statements into a file called “loadData.sql”.

Hints for Part 4

There will be some variations depending on the schema that you choose. In most cases, however, you can load data into your tables using very simple SQL commands.

For example, suppose (whether or not it is a good design) that you created a table LOCATION, which contains the attributes LOC_ID, CITY, STATE, and COUNTRY. Suppose that you want this table to contain a listing of all the different locations, without duplicates. You might load data into the table using the following command (UNION eliminates duplicates):

```
INSERT INTO LOCATION (CITY, STATE, COUNTRY)
SELECT DISTINCT HOMETOWN_CITY, HOMETOWN_STATE, HOMETOWN_COUNTRY FROM
PUBLIC_USER_INFORMATION
UNION
SELECT DISTINCT CURRENT_CITY, CURRENT_STATE, CURRENT_COUNTRY FROM
PUBLIC_USER_INFORMATION
UNION
SELECT DISTINCT EVENT_CITY, EVENT_STATE, EVENT_COUNTRY FROM
PUBLIC_EVENT_INFORMATION;
```

You may also find yourself in a situation where it would be useful to construct an internal key (i.e., a key whose value is meaningless outside the database), such as the LOC_ID mentioned above. You can do this in Oracle by declaring a sequence variable and a trigger. For example:

```
CREATE SEQUENCE loc_sequence
START WITH 1
INCREMENT BY 1;

CREATE TRIGGER loc_trigger
BEFORE INSERT ON LOCATION
FOR EACH ROW
BEGIN
SELECT loc_sequence.nextval into :new.LOC_ID from dual;
END;
.
RUN;
```

Whenever you insert a row into LOCATION, the above will automatically set the value of LOC_ID to the next integer in the sequence.

As a useful additional reference, you may also want to look at Toby Teorey's SQL user guide: http://www.eecs.umich.edu/~klefevre/eecs484/SQL-miniUG.pdf or a more extensive guide maintained by Jeff Ullman at Stanford: http://infolab.stanford.edu/~ullman/fcdb/oracle.html

There may be some pieces of data that we have asked you to represent in your database schema (ER diagram and relational tables), but for which we have given you no data. Please do represent these items in your schemas (ER diagrams and CREATE TABLE DDL). However, when you load the data from the provided schema, don't worry about populating these fields. That is, you should have either empty tables, or null values, depending on how you have designed the schema.

Submission Checklist

Please put all your files in a single tar file called project1_uniquename1_uniquename2.tgz and submit a single file via CTools Assignments. If you are doing the project alone, then name the tar file as project1_uniquename.tgz. The tar file should contain the following files:

1. A Word (doc or docx) or PDF document that contains your ER Diagram from Part 1. You may also draw the ER diagram by hand, and submit an electronic version by scanning the drawing. Name the file ER_Diagram.(pdf|doc|docx).

2. A short summary report in a file mytables.pdf that contains the following in a tabular form for all your tables: table name, table's schema, and the count of number of rows in that relation. You can retrieve the count for each table as follows (replace *table_name* by the table name):

   SELECT COUNT (*) FROM *table_name*;

3. 3 SQL files
   a. createTables.sql (Part 2)
   b. dropTables.sql (Part 2)
   c. loadData.sql (Part 4)

Only **single** submission is required per team (from either one of the member).

How to create a tar file?

Log into a Linux machine. Put all your submission files into one folder `project1_uniquename1_uniquename2`. Then from the directory above `project1_uniquename1_uniquename2`, type the command:

```
% tar czvf project1_uniquename1_uniquename2.tgz project1_uniquename1_uniquename2/
```