Problem Set #1

Note: In this homework, you do not need to create tables and execute queries using an actual DBMS. A written solution is sufficient.

1. Suppose you have a database modeling an online auction site somewhat similar to eBay, given by the following schema:

USER (uid, uname, city, state)
ITEMTYPE (iid, itemname, description)
AUCTION (seller, iid, starttime, endtime, minbid, condition)
Note: seller references uid in USER, and iid references iid in ITEMTYPE
BID (bidder, seller, iid, starttime, bidtime, bidprice)
Note: bidder references uid in USER, and (seller, iid, starttime) references AUCTION

In this schema, users can sell items at auction, and can also place bids on auctions by other users. An auction has a start time, when bidding begins, and an end time (both of type datetime). The auction is won by the highest bid made before the end time, provided the bid is higher than the minimum bid. The description of the item that is being sold in the auction is kept in a separate table ITEMTYPE, because several items of the same type (say, a certain book or model of sneakers) may be sold in different auctions by the same or different users, and this way we avoid storing the same information repeatedly. (Note that the condition of the item, such as ``new'', ``like new'', ``slightly damaged'' etc., is kept in the AUCTION table since the same type of item may come in different conditions.) Finally, a bid by a user is for a particular auction, and contains the time of the bid and the bid price (which should be higher than the minimum bid and all previous bids for this item, to have a chance to be successful, but this is not enforced by the schema).

(a) In the AUCTION table, (seller, iid, starttime) is the primary key. Could (seller, iid) be a primary key? Why or why not?

(b) For table BID, does bidtime have to be part of the key? What happens if we remove bidtime from the key?

(c) Write statements in SQL for the following queries.

I. For each user, output how many items they put up for auction in 2018.
II. For each auction that was successfully sold, output the uids of the buyer and seller, the iid, and the price at which it was old.
III. Output the highest bid that was made for an item with itemname “IPhone 12”.
IV. Output the uid of any user who has placed more than ten bids but who did not win even a single auction.
V. For each itemname and condition, list the average winning bid for auctions involving such an item during 2018. (So, what is the average price for which such an item sold in 2018?)
VI. Output the uid of any user who placed a bid in his/her own auction.
VII. List any pair of users that placed bids in the same auction in more than ten auctions in 2018. (Meaning, users that have often competed for the same item.)

(d) Write expressions in Relational Algebra for queries I to VI.

(e) Write either DRC or TRC queries for queries I to V. Or explain the reason if you think a particular query cannot be done in DRC or TRC.

2. In this problem, you have to design a relational schema for a web-based service, similar to Pinterest, that allows people to maintain online "pinboards" with pictures that they want to share with others. Users can sign up for the service, and can then create one or more pinboards. Later, users can upload pictures and "pin" them to one or more of their pinboards. Users can also "re-pin" pictures that they find on other user's pinboards, which adds them to their own boards. Users can also follow other people's pinboards, and can like and comment on pictures they find on other boards.

As an example, consider two users, Erica and Timmy. Erica likes to travel, and also loves antique furniture. She signs up and creates two pinboards, "Furniture" and "Dream Vacations". Whenever she sees a picture that she likes and wants to show to her friends, say a picture of a nice sofa on a website, or a picture of a beautiful beach that she took herself, she uploads it and pins it to one of her boards. Erica also has friends who often look at her images and sometimes like the pictures or leave comments such as "Cute" or "love it". Timmy is six years old, likes dinosaurs and monsters, and when he grows up he wants to become a pirate. He creates boards named "Super Dinosaurs" and "Pirates" and whenever he sees a picture of dinosaurs or pirates (or even better, dinosaurs *and* pirates) he pins it to his boards. He also follows several pinboards by others, and sometimes re-pins some of these pictures so they appear on his own boards.

For this homework, you only need to design a suitable relational schema for this application, and thus you do not have to worry about the looks and functionality of the service. You may store pictures either as type clob, or just assume that each picture is identified by a picture id that can be used by the web frontend to retrieve the actual picture from some external storage system (say, a file system or a tuple store). Thus, you need to store information about users (e.g., their name, hometown, email), about boards (e.g., the name of the board, when it was created, who created the board, and who follows the board), and about pictures (the name or description of the picture, when it was uploaded, who pinned and re-pinned the picture, and who liked it and commented on it).

(a) Design a relational database schema for this application that supports the above functionality. Specify all primary and foreign key constraints, and state any assumptions you are making. You can decide which exact attributes make sense for this schema.

(b) Write SQL statements for the following queries. If your schema does not support these, you need to modify it appropriately. (For this first homework, you may use informal expressions such as year(ts) = ``2018'', where ts is a timestamp, to check if the year is 2018.)

I. We say that two users are friends if each has liked at least three pictures from the other user's boards. Output all pairs of friends.

II. For each picture, output the number of times it has been re-pinned.

III. Output the names of all users who follow a board containing a picture with the description "cactus at sunset".

IV. Output the names of all users who posted more than ten comments in the last 24 hours. (You may use "currenttime()" to refer to the current time in your query.)