NYU Tandon School of Engineering **October 3, 2017**
Computer Science and Engineering
CS6083, Fall 2017

Problem Set #2 (due October 16)

Problem 1:

In this problem, you have to write SQL and RA queries for a database modeling the short-term leasing of houses or apartments, in a system somewhat similar to Airbnb. Here are the tables:

Customer (cid, cname, cphone, ccity);
Landlord (lid, lname, lphone, lcity)
Residence (rid, rname, rstate, rcity, raddr, rtype, rarea, lid);
Leases (cid, rid, startdate, enddate, price);
Rating (cid, rid, rtime, score);

Customers are identified by a cid, and we also store their name, phone number, and the city they live in. Landlords are identified by an lid, and have a name, phone number, and city. One landlord could own multiple houses/apartments, but a house could only have one owner. Residences are identified by an rid, and have an rname, and rstate, rcity, and raddr to store the precise address of the house (e.g. rcity = 'Brooklyn', raddr = '3rd floor, 308 45st, 6 Avenue'), an rtype that stores the type of residence (e.g studio or 2BR-1BA), and harea indicating the square footage. Leases are identified by cid, rid, lid, and startdate, and we also store enddate and price. The startdate and enddate attributes store both time and date information. When the lease ends, customers could give a rating to this house, where a rating contains cid, hid, rtime, and a score. rtime is a timestamp which indicates the date and time when the customer made the rating; scores are ranging between 1 star (Terrible!) and 5 stars (Awesome!).

(a) Suppose the *Residence* table did not have the attribute hid,. Can you identify a new primary key? If you can, identify a new primary key; if not, write down the reasons.

(b) Identify suitable foreign-key relationships for the above tables.

(c) Draw an ER diagram that is consistent with the above relational schema. Show the cardinalities of the relationships, and identify any weak entities.

(d) Create the above schema in a database system, choose appropriate attributes types, and define primary keys, foreign keys, and other constraints. Data for this schema will be made available on NYU Classes; please use that data. Load the data into the database using either insert statements or the bulk-loading facilities. You may use any mainstream relational database system.

(e) Write the following SQL queries and execute them in your database. Show the queries and the results:
(i) List the cids and cnames of all customers who have rented residences in Chicago.
(ii) List the cids of any customers living in Chicago who had a lease with a landlord living in Seattle.
(iii) Output the cid and cname of any customers who have only rented residences in Chicago. (That is, they have rented at least one residence in Chicago, and have not rented anything anywhere else.)
(iv) Output the residence type that is owned by the largest number of distinct landlords.
(v) Output the cid and cname of the customer(s) who spent the most money (sum of prices) for rentals with starting dates in 2016.
(vi) For each lcity, output the lids of the landlords living in that city who made the most money overall.

(vii) For each month in 2016, output the rcity whose houses had the best average rating, based only on ratings that were given during that month.
(viii) Output the average duration of leases (enddate - startdate) for customers living in San Diego, for leases with start date in 2016. The unit of average time should be days.

(f) Write expressions in Relational Algebra for queries (iii) to (viii).

(g) Write SQL statements to perform the following updates to the database:
(i) For every house whose landlord is named "Bob Daniels", change it to "Amy Daniels".
(ii) Delete tuples of any customer who has not made a lease in the last 3 years from the *Customer* table.
(iii) For any house with average rating below 3 stars, delete all of its rating records with rating 1 star.

Problem 2:

In this problem, you have to create views and then write queries on the views, and create triggers, for the schema in Problem 1. Execute everything on your database system, and report the results.
(a) Define a view that contains for each residence the lid, lname, lcity, rid, rname, and rcity. Using this view, try to answer the following queries:
(i) For each lcity, output the lname that owns the most residences.
(ii) Change the name of any residence with name "Little Cabin" to "Big Cabin".

(b) Consider the last query in Problem 1(g). Can you write a trigger that automatically deletes ratings with rating 1 star whenever a residence's average rating drops below 3 stars?

(c) Write a trigger that prevents additional ratings for any customer who has already done five 1-star ratings in the past 2 weeks.

Problem 3:

In this problem, you are asked to design the relational backend for a company called HappyDelivery. The idea is that sometimes people want to buy products from a convenience store without leaving their home, but not every convenience store provides delivery service. HappyDelivery is making this possible, by maintaining product and price information for many participating chains of stores. Thus, a customer can visit their site, select some products for purchase, and pay the bill by using a credit card. After the order is successfully placed, the convenience store will prepare the order for pickup. When the order is ready, HappyDelivery will send a driver to pick up the products and deliver them to the customer. After the delivery is done, HappyDelivery credits the delivery fee to the driver's account and adds the purchase price to the convenience store's account. (HappyDelivery makes profits by getting referral fees from the stores, but you do not have to model this part).

HappyDelivery assigns a unique ID to each driver, and the database also stores their real name, address and phone number. Customers are identified by a unique customer ID, and they also need to provide their name, phone number, address, and credit card number. Each credit card consists of a card number and an expiration date. A customer may have more than one credit card. An order may involve several items, for example three cans of coke and one donut, but all items must be from the same store (otherwise you need to place two orders). For each order, we need to specify which products were bought, the customer, the store, the driver who delivered, the cost of the products, the delivery fee, and which credit card was used. Each store (i.e., convenience store chain) has a unique name. Products are identified by product IDs that are globally unique – that is, two items in different stores have the same product ID if and only if they are in fact the same product -- and have a short description. Of course, different stores may charge different prices for the same product. After the deliver is done, the customer can rate the service on the order from one to five stars.

(a) Design an ER diagram that can model the above scenario. Identify suitable keys and the cardinalities of the relationships. Also identify any weak entities. Discuss any assumptions that you are making in your design.

(b) Convert your ER diagram into a relational schema. Show primary keys and foreign key constraints.

(c) Write statements in SQL for the following queries. Note that if your schema does not allow you to answer a query, you may have to go back and change your design.

- (i) For each driver, output her ID, name, and the total amount of delivery fees she has earned from doing deliveries for a convenience store named "Tom's Store".
- (ii) Output the ID of any driver who has never received a rating of five stars.
- (iii) Output the name and ID of any customer who has ordered at least four times from the same store.
- (iv) Output the name and ID of the customer(s) who spent the most money during 2016. Include both the product prices and the delivery fees in your calculation.

(d) Create the tables in your database system, and insert some sample data (maybe 5-10 tuples per table). Choose an interesting and meaningful data set. Then execute the queries from part (c). Submit your sample data, and screenshots or logs of the queries and outputs.