NYU School of Engineering
Computer Science and Engineering
CS 6083, Fall 2015

October 27, 2015

CS 6083 Section A – Midterm Exam

Instructions. This is a 150-minute exam, with a total of 100 points available.

- Do not open this test booklet until you are told to do so.
- Write your name and student ID *neatly* on this cover sheet.
- The exam is open book, open nodes. No electronic devices of any kind may be used, including calculators. Make sure to switch off your phones.
- Please write your answers *neatly* in the space provided. Use the back of the sheets if you need additional space.
- Good luck!

Name: _____

Student ID: _____

Problem 1. (52 Points) Suppose we have a database consisting of the following six relations:

Bar(<u>barID</u>, barname);
Customer(<u>custID</u>, custname);
Beer(<u>beerID</u>, beername, beercompany);
Frequents(<u>custID, barID</u>);
Serves(<u>barID, beerID</u>);
Likes(<u>custID, beerID</u>);

The first three relations describe bars, customers, and beers. The fourth relation indicates the bars that each customer visits regularly; the fifth relation models what beers each bar serves; and finally the last relation indicates which beers each customer likes.

(a) (8 points) Draw an ER diagram for the above relational scheme. Identify keys and any weak entities.

(b) (16 points) Write SQL statements for the following queries:

- Find any customer that likes a beer made by "Brooklyn Brewery".
- Find any customer who frequents a bar where she does not like a single beer.
- Find any bar that serves every beer made by "Brooklyn Brewery".
- For each beer, list the name of the beer and the number of bars serving it.

(c) (16 points) Write relational algebra queries for the above four queries.

(d) (12 points) Write relational calculus queries for the first three queries only.

Problem 2. (48 Points) In this problem, you are asked to design a relational database schema for a startup called `Unter`^{TM} that allows people to share rides on private airplanes. The idea is that many private planes, including small airplanes owned by individuals and corporate jets owned by companies, often have empty seats on their trips, and that it would be better for everyone if other people could use those seats. So this service is a little similar to Uber or airbnb, but for airplanes.

There are two types of people in this system, flyers (customers who want to fly) and pilots (pilots of planes who have seats available). Your system needs to store some amount of information about these two groups, such as usernames, email, hometown, etc. (Choose a suitable set of attributes.)

When a pilot plans to take a trip, say next week, they post this information on the web. In particular, they post the origin and destination airport of the trip, a date, and a time window when the trip will leave and arrive (e.g., departure between 10am and 2pm, and arrival between 4pm and 10pm). They also post information about the airplane, including the name and model of the airplane, the year it was built, and a description that might include special features (e.g., showers on board, or an espresso machine). Note that many of these airplanes have customized interiors, so the model of the airplane does not tell you all its features. A pilot might fly several airplanes, and an airplane could be flown by several pilots.

For each planned flight, the pilot also has to provide information about each seat that is available on the airplane for that particular flight, with a seat number, a price, and maybe a short description (e.g., leather seat with extra leg room). Different seats on the same flight might have different prices as they offer different amenities. For simplicity, all flights are one-way and there are no connections.

Flyers who want to do a trip may also post information about their plans; e.g, a customer might be interested in a flight from Nashville to Madison that departs in the morning of October 20 and arrives by 6pm, and that costs at most $2000. This might potentially motivate a pilot to fly this route if enough people are looking for a seat.

Flyers and pilots can find and contact each other (the mechanisms for doing this do not have to be modeled by you). Once they agree, you have to store a reservation that includes information such as the flyer, the pilot, the flight, the seat number, and the agreed-upon price (which might be different from the advertised one). If the flyer cancels the reservation, they lose their payment. If the pilot cancels the reservation, they need to return the payment plus 50% penalty to the flyer. If the plane departs or arrives more than 4 hours late, they need to return half of the price to the flyer. Of course, if the pilot cancels a planned flight before any reservations were made, no penalty needs to be paid. Thus, you need to store if a flight actually took place, and also when it actually departed and arrived.

(a) (20 points) Design an ER schema that can model the above scenario. Identify any weak entities, suitable keys, and the cardinalities of the relationships. Discuss any assumptions that you are making in your design. Explain the schema in a few sentences!

(b) (12 points) Convert your ER diagram into a relational schema. Show primary keys and foreign key constraints.

(c) (16 points) Write statements in SQL for the following queries. Note that if your schema does not allow you to answer a query at all, you should go back and change your design. (But do not try to fine-tune your schema to these particular questions; keep it simple and general.)

(i) Check if any seat is available for a flight from JFK to LAX on November 10, 2015 that arrives before 6pm and that costs at most $1000.

(ii) Imagine you are a pilot trying to decide when to make a flight from JFK to LAX. Write a query that lists, for each day in November 2015, the number of flyers offering more than $800 for a seat on that route for that day.

(iii) Output the names of any pilots that lost money during October 2014, because the total penalties they paid for cancellations and lateness were more than what they got paid.

(iv) Output the name of any pilot who cancelled more than 3 flights in 2014 after accepting at least 5 reservations for each flight.