CompSci 316 Fall 2016: Homework #1

100 points (8.75% of course grade) + 10 points extra credit
Assigned: Thursday, September 1
Due: Tuesday, September 20

This homework should be done in parts as soon as relevant topics are covered in lectures. If you wait until the last minute, you might be overwhelmed.

For Problems 1, 3, and 5, you will need to use Gradiance. Gradiance is an online testing system that provides immediate feedback to your answers, and allows you to retry a problem multiple times until you get it right. Please read the "Help → Getting Started with Gradiance" section of the course website for instructions on how to get started. There is no need to turn in anything else for these problems; your scores will be tracked automatically.

For other problems, you will need to turn in the required files electronically. Please read the "Help → Submitting Non-Gradiance Work" section of the course website for instructions. For Problems 4 and X1, you may prepare your answers electronically or on paper (handwritten). In the latter case, scan or photograph the pages and submit the resulting PDF (preferred) or JPG files. Please name your files informatively, e.g., as `n.pdf`, where `n` is the problem number.

Problem 2 should be completed on a virtual machine (VM). Please read the VM-related section under "Help" on the course website, and follow the instructions therein to get your VM running. (You will receive an email from the instructor by Monday, September 5 concerning Google Cloud credits, if you opt to run your VM on Google Cloud.)

Problem 1 (13 points)

Complete the Gradiance homework titled "Homework 1.1 (Relational Algebra Basics)." Note that some of the online exercises use English names of relational algebra operators instead of symbols.

Problem 2 (32 points)

Consider a database containing information about bars, beers, and bar-goers.

> *Drinker*(<u>*name*</u>, *address*)
> *Bar*(<u>*name*</u>, *address*)
> *Beer*(<u>*name*</u>, *brewer*)
> *Frequents*(<u>*drinker*</u>, <u>*bar*</u>, *times_a_week*)
> *Likes*(<u>*drinker*</u>, <u>*beer*</u>)
> *Serves*(<u>*bar*</u>, <u>*beer*</u>, *price*)

Write the following queries in relational algebra using RA, our homegrown relational algebra interpreter. To set up the sample database called `beers`, issue this command in your VM shell:

```
/opt/dbcourse/examples/db-beers/setup.sh
```

Then, type “`ra beers`” to run RA. See “Help → RA: A Relational Algebra Interpreter” on the course website for additional instructions on using RA, including syntax of relational operators.

When you run RA, as soon as you get a working solution, record the query in a plain-text file named `2-query.txt` (use Java/C++-style comments in the file to indicate which problem each query corresponds to). When you are done with all queries, run

```
ra beers 2-queries.txt > 2-answers.txt
```

to generate the final answers. Submit the files `2-queries.txt` and `2-answers.txt` through the submission website. If you cannot get a query to parse correctly or return the right answer, include your best attempt and explain it in comments, to earn possible partial credit.

(a) Find names of beers served at *James Joyce Pub*.
(b) Find names and addresses of bars that serve some beer for less than $2.25.
(c) Find names of bars serving some beer *Amy* likes for no more than $2.50.
(d) Find pairs of drinkers who like the same beer. (Just list the drinker names, not the beer. Don’t list (*drinkerA*, *drinkerA*). If you list (*drinkerA*, *drinkerB*) in the answer, don’t list (*drinkerB*, *drinkerA*) again.)
(e) Find names of all drinkers who like *Dixie* but frequent none of the bars serving it.
(f) For each drinker, find the bar that he or she frequents the most. (If multiple bars tie for the most frequented by the drinker, list all of them.)
(g) Find names of all drinkers who frequent ***only*** those bars that serve some beers they like.
(h) Find names of all drinkers who frequent ***every*** bar that serves some beers they like.

Problem 3 (15 points)

Complete the Gradiance homework titled “Homework 1.3 (E/R Design).”

Problem 4 (20 points)

Professor Oak’s server crashed and he lost his master Pokedex and Pokemon GO trainer database. He has hired you to redo his database.

- There are many species of Pokemon. Each species has a unique name as well as base attack, defense, and stamina ratings. For example, the species named Chamander has 128 base attack, 108 base defense, and 78 base stamina.
- Some species can evolve into other species (e.g., Chamanders evolve into Charmeleons, which further evolve into Charizards). Some species don’t evolve (e.g., Snorlaxes). Some species (e.g., Eevees) can evolve into multiple different species (e.g., Vaporeons, Jolteons, or Flareons). However, no two species evolve directly into the same species.
- An individual Pokemon belongs to a species, and has its own “IVs” (individual values) that stack on top of base attack, defense, and stamina ratings. It also has a unique id, a name (not necessarily unique), and a level. For example, trainer Bart’s level-1 starter Chamander is named “Fire6a11” and has perfect IVs of 15/15/15.
- Pokemon have “moves” that they use in combat. Every move has a unique name, deals a certain amount of base damage per hit, and has a certain cooldown time between hits. A move is either “quick” or “charged.” A quick move gains a certain amount of energy per hit, and each charge move

consumes a certain amount of energy per hit. For example, Water Gun, a quick move, deals 6 base damage and gains 7 energy per hit, with a cooldown of 0.5 second; Hydro Pump, a charged move, deals 90 base damage per hit and consumes 100 energy, with a cooldown of 3.8 seconds.

- Each individual Pokemon has exactly one quick move and one charged move. (In reality, one species can acquire only a specific subset of all possible moves. Luck for you, Professor Oak told you not to worry about that for this assignment.)
- Trainers have a unique id, a nick name (not necessarily unique), and a certain number of experience points. Each individual Pokemon may optionally have one trainer. A trainer may "favorite" or "un-favorite" each individual Pokemon of his or hers. A trainer can have multiple favorites.

If you think some aspects of the above description are unclear, feel free to make additional, reasonable assumptions, but state them clearly in your answer. Also, keep in mind that there is no single "correct" design; if you think you are making a non-obvious design decision, please explain it briefly.

(a) Design an E/R diagram for this database. Very briefly explain the intuitive meaning of any entity and relationship sets. Do not forget to indicate keys and multiplicity of relationships, as well as ISA relationships and weak entity sets (if any), using appropriate notation.

(b) Design a relational schema for this database. (You can start by translating the E/R design.) You may ignore attribute types, and you do not need to show any sample data. Indicate all keys and non-trivial functional dependencies in the schema. Check if the schema is in BCNF. If not, decompose the schema into BCNF.

Problem 5 (20 points)

Complete the Gradiance homework titled "Homework 1.5 (Relational Design Theory: FD)."

Extra Credit Problem X1 (5 points)

As discussed in class, the core operators in relational algebra are selection (σ_p), projection (π_L), cross product ($\times$), union ($\cup$), and difference ($-$). Prove that the selection operator is necessary; that is, some queries that use this operator cannot be expressed using any combination of the other operators.

Extra Credit Problem X2 (5 points)

Continuing with Problem 4, design five queries based on the Pokemon GO database, like those in Problem 2 for the beer drinker database. At two out of the five queries should be about the same level of difficulty as the last four queries of Problem 2. Submit the English descriptions and solutions (in relational algebra) for your queries. (You do not need to create a sample database or use RA for this problem.)