University of California, Los Angeles
Department of Computer Science

Computer Science 143 **Prof. Ryan Rosario**

Homework 3

Due Tuesday, May 22, 2018 11:59pm ***via CCLE***

Please remember the following:

1. Homework is mostly graded on completion. We may grade a few parts, but it will never be the majority of the grade on the assignment. So try your best, and focus on solving the problems. Consider homework (and studying the solutions) as practice for the final exam.
2. Homework must be submitted digitally, on CCLE. We will not do any paper grading. You can use a text file, but if you use Word, a PDF is preferred rather than a DOC file.
3. If there are any exercises that are difficult to do digitally (such as diagrams or math), consider scanning your drawing or math, or using a graphics program (even a readable MS Paint is fine) or Equation Editor.
4. **For the sanity of the grader** we will ask you to run the queries and submit the result. You may lose points if you only provide a query.
5. Solutions will be posted.

Part 1: Go Long or Go Wide?

The website `keyvalues.com` allows startups and some large companies to discuss their company culture, something that some of you will find to be much more important than a huge paycheck. The purpose of this site is to provide candidates and other interested parties an in depth look at the company culture on a variety of dimensions. The content tends to discuss cultural values that are often overlooked in interviews. The table below shows some of them:

Team Values	Personal Health	Daily Routines	Engineering	Career Growth
Engages with Community	Work/Life Balance	Eats Lunch Together	Open Source Contributor	Promotes from Within
Team is Diverse	Ideal for Parents	Light Meetings	Start-to-Finish Ownership	Good for Junior Devs
Risk Taking > Stability	Fosters Psychological Safety	Thoughtful Office Layout	Fast-Paced	High Retention

Strategy	Company Properties
Engineering-Driven	Remote Work OK
Data-Driven	
Rapidly Growing Team	

Exercise

Help your professor! (Though I've already done this) Take a look at the data. Write a query that creates a 0/1 matrix from this data. The rows should correspond to companies, the columns should correspond to cultural values. The value of each cell should be a 0 or 1 – a 1 if the value is associated with the company, 0 otherwise.

The query may be tedious, but in the "lots of copy/paste" way. Some of you may find a much better way to do this that does not require that. I do not know if it is possible, so if you are adventurous, you may want to research it.

To grade this problem, please submit your query, the number of rows in the output, and the number of columns in the output. The output itself you can submit as a text file if you wish, but it is probably too tedious to read.

With this 0/1 matrix, we can then create visualizations of which companies are similar to others, and which values are similar to others using singular value decomposition, principal component analysis, or multiple correspondence analysis. We won't do that for this class, but if you are interested, have some fun with the data.

Part 2: `EXPLAIN` Yourself

In the solutions for Homework 2, I gave several ways of writing the same query. For part 1B:

1. No Subquery with `DISTINCT`
2. `SELECT` within a `SELECT`
3. `JOIN` as a Filter
4. Using an `IN` Subquery as a Filter
5. Formal Left Semijoin

Run a SQL `EXPLAIN` on each of these queries. Just copy and paste the query from the solutions into a text editor and make any corrections so that it will run. The `EXPLAIN` output varies by implementation, but for MySQL, you can read more about it here: `https://dev.mysql.com/doc/refman/8.0/en/explain-output.html`.

Provide the output of *each* SQL `EXPLAIN`. Then write a paragraph (approximately) detailing what you notice overall (do not write a paragraph for each one). Which query appears to be the most efficient (intentionally vague)? What is your definition of efficient? Pay attention to the number of rows, the `filtered` column, the extra column, and any information provided about joins.

Part 3: Join and Optimization Algorithms

Chapter 12 is rather tedious, so I have asked everyone to read this chapter. It is so tedious that it would take multiple lectures for me to address the class and also answer questions. While we have not focused on nitty-gritty math involved in disk block reads (aside from on the exam), it is important to be aware of it as you may be asked in which situation to use which join algorithm, and you will probably want to use this math to justify your response. Remember, data wins arguments.

Exercises

1. Suppose we have two relations L and R. The nested-loop join algorithm is presented below:

```
for each t_l ∈ L:
  for each t_r ∈ R:
    if t_l.K = t_r.K then output tuple (t_l, t_r)
```

   If L has 100,000 tuples, how many times is relation R scanned?

2. The indexed nested-join loop is similar, but instead of doing a linear scan over R, we build an index on it.

```
for each t_l in L:
  X = index-lookup(t_l.K)
  for each r ∈ X:
    output tuple (t_l, r)
```

   (a) How many times is each tuple in both L and R scanned?

   (b) In the worst case, what is the number of block transfers required for this join if we assume R is indexed with a B+-tree?

3. In terms of disk I/O, which of the three join algorithms is the most efficient: naive nested-join loop, block nested-join loop, indexed nested-join loop and why? The answer probably depends on several factors. Describe these factors and come up with a heuristic describing when you may want to use each.

4. Prove that the following equivalencies hold. Explain how you can apply them to improve the efficiency of certain queries. Feel free to define relations if you would like:

$$E_1 \bowtie_\theta (E_2 - E_3) = (E_1 \bowtie_\theta E_2 - E_1 \bowtie_\theta E_3)$$

5. For the pair of expressions, give instances of relations that show the expressions are not equivalent.

$$\Pi_A (R - S) \text{ and } \Pi_A(R) - \Pi_A(S)$$