University of California, Los Angeles
Department of Computer Science

Computer Science 143 **Prof. Ryan Rosario**

Homework 4

Due Friday, May 24, 2019 at 11:59pm on 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. Solutions will be posted.

Part 1: `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 PostgreSQL, you can read more about it here: `https://www.postgresql.org/docs/11/using-explain.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 what is displayed in the output. PostgreSQL 11 is available on your Project 2 VM, but if you have issues with it, the Project 1 VM will still work.

Part 2: Here we Go *Loopty*-Loo!

Chapter 15/16 are 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 $t_l \in L$:
  for each $t_r \in R$:
    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 $r \in X$:
    output tuple $(t_l, r)$
```

   (a) How many times is each tuple in both L and R scanned, assuming the index has not already been built?

   (b) Suppose the cardinality of R and L is $|R|$ and $|L|$ respectively, and the number of blocks they occupied is b_R and b_L. The memory has M blocks. The index occupies N blocks. On average, there are J matching L tuples per an R tuple. 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? Perform this computation both with and without the index *precomputed* on the join key.

Part 3: There's Nothing Wrong with Being Abnormal Unless you are a Relation

1. Suppose that we decompose the schema $R(A, B, C, D, E, F)$ into $R_1 = (A, B, C, F)$ and $R_2 = (A, D, E)$. Given the following functional dependencies hold, is the decomposition lossless? Explain your answer.

$$F = \{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$$

2. List non-trivial functional dependencies satisfied by the following relation. You do not need to find *all* dependencies. In other words, please F, but there is no need to find F^+. It is enough to identify a set F of functional dependencies that imply all functional dependencies satisfied by this relation.

A	B	C
a_1	b_1	c_2
a_1	b_1	c_2
a_2	b_1	c_1
a_2	b_1	c_3

3. Assume the following set of functional dependencies hold for the relation $R(A, B, C, D, E, F) : F = \{A \to BC, C \to E, B \to D\}$

   Is R in Boyce-Codd Normal Form (BCNF)? Explain your answer. If it is not, normalize it into a set of relations in BCNF such that the decomposition is lossless.

4. Suppose we have a relation $R(A, B, C, D)$ with a multivalued dependency (MVD) $A \twoheadrightarrow BC$. If we know that the tuples $(a, b_1, c_1, d_1), (a, b_2, c_2, d_2), (a, b_3, c_3, d_3)$ are in the current instance of R, what other tuples do we know must also be in R?

5. For relation $R(A, B, C, D, E, F)$, suppose a functional dependency $AB \to E$ and two multivalued dependencies $AB \twoheadrightarrow C$ and $A \twoheadrightarrow B$ hold. Is R in 4NF? Explain your answer. If not, normalize it into 4NF.

6. Consider the following relational schema describing musical events in Los Angeles in some prior decade. Assume each event has at least one band. Bands stick to one genre of music and they do not visit the same venue twice in the same year.

Venue	Year	Band	Genre
X	1999	Mighty Mighty Bosstones	ska
Z	1999	Mighty Mighty Bosstones	ska
Y	2001	Mighty Mighty Bosstones	ska
Y	2001	Porcupine Tree	rock

   Is this relation in 2NF? 3NF? Determine the functional dependencies and keys and justify your answer.

7. We now modify the schema to include the number of attendees:

Venue	Year	Band	Attendees
X	1999	Mighty Mighty Bosstones	10000
Z	1999	Mighty Mighty Bosstones	8000
Y	2001	Mighty Mighty Bosstones	90000
Y	2001	Porcupine Tree	10000

   Is the new schema in 2NF? 3NF?