Problem Set #2: Relational Design Theory

Due: October 13, 2014, 10:59am

1 Comments about Grading

These problem sets will be graded roughly.

- If your answer is well written and contains a complete, correct answer for a question, you will get bonus points. These bonus points are rare.
- If you get basically everything right, but we find some mistake or gap in your answer, we will give you full credit. We hope to give you essentially full credit for each question.
- Partial credit will be given aggressively–so turn something in!

There will be provided solutions. As a result, problem set material is fair game for tests and quizzes.

2 Design Problems

Problem 1

Your task is to write SQL queries to check satisfaction of dependencies on table instances. Suppose you have a table $T(A, B, C, D)$. You may assume there are no NULL values in T, but otherwise make no assumptions about the data. For each of the following four subquestions, write a query that returns an empty result if and only if the current contents of table T satisfy the dependency. That is, if the dependency is satisfied then the result of the query is empty, while if the dependency is not satisfied then the query result is nonempty (the specific contents of the result don't matter).

(a) Functional dependency A $\rightarrow$ D

(b) A pair of Functional dependencies A $\rightarrow$ B and $B \rightarrow C$.

(c) Multivalued dependency AB $\twoheadrightarrow$ C

(d) Suppose there is a table $U(D, E)$. Check what is called an *inclusion dependency* that is $U[D] \subseteq T[D]$, i.e., every value in $U[D]$ is present in $T[D]$.

Remark to Problem 1. *We're looking for solutions that use the standard declarative constructs of SQL, not esoteric constructs supported by specific systems. A full credit answer will include a transcript of running your SQL queries over sample tables on SQLite, MySQL, or some other SQL DBMS. Your sample tables should demonstrate a variety of instances of table T, including some instances satisfying the dependencies and some not. Please include your data as well as your query results.*

Problem 2

Consider a relation $R(A_1, A_2, ..., A_n)$ with $n > 0$ attributes. Your goal is to construct an instance (set of tuples) for this relation that satisfies the functional dependencies $A_iA_{i+1} \rightarrow A_{i+2}$ for all $i = 1, \ldots, n-2$. That is, in your instance:

$$
\begin{gathered}
A_1, A_2 \rightarrow A_3 \\
A_2, A_3 \rightarrow A_4 \\
A_3, A_4 \rightarrow A_5 \\
\cdots \\
A_{n-2}A_{n-1} \rightarrow A_n
\end{gathered}
$$

Furthermore, the only functional dependencies your relation instance should satisfy are the ones above, and those that follow from them. Aside from dependencies that follow from the ones already mentioned, all other functional dependencies (e.g., $A_2 \rightarrow A_1$, $A_1 \rightarrow A_2$, or $A_4, A_3 \rightarrow A_2$, etc.) should be violated by the set of tuples in your relation instance.

Construct the set of tuples that satisfy the above requirements for $n = 2$, $n = 3$, and $n = 4$. Try to find an instance that has as few tuples as possible. Provide a brief explanation of how you arrived at your solutions, and give a very general description of how you could con struct such instances for larger n.

Problem 3

Consider a relation $R(A, B, C)$. In this question, you wil give a pair of instances I_1 and I_2 as part of your answer. We will only consider pairs (I_1, I_2) such that I_1 and I_2 differ by a single tuple.

- $A \rightarrow B$ holds in I_1 but not in I_2.
  Can I_2 be larger, smaller, or both larger and smaller than I_1?

- $A \twoheadrightarrow B$ holds in I_1 but not in I_2.
  Can I_2 be larger, smaller, or both larger and smaller than I_1?

If you state both, then give examples to support your position. Otherwise, give your reasons why you believe I_2 cannot be larger or smaller.

Problem 4

Prove the augmentation for multivalued dependencies from the definition. Specifically, consider a relation R, and let A, B, C and D be four sets of attributes in R. Prove if $A \twoheadrightarrow B$ and $C \subseteq D$ hold for R, then $AD \twoheadrightarrow BC$ also holds.

Remark to Problem 4. *Your proof should be based on the formal definition of MVDs, not on other rules. In other words, you should argue that if two distinct tuples t_1 and t_2 that agree on AD (i.e. $t_1[AD] = t_2[AD]$) then there is some tuple t_3 such that $t_3[AD] = t_1[AD] = t_2[AD]$ and $t_3[BC] = t_1[BC]$ and $t_3[R \backslash BC] = t_2[R \backslash BC]$. Here, "$R \backslash BC$" means the set of all attributes in R minus those that are in B or C.*

3 Submission Instructions

To submit Problem Set 2, save your answers as a PDF (preferably typewritten using LaTeX or Microsoft Word, but a handwritten & scanned PDF is fine as long as it is legible).

Solutions to the SQL problems should be stored as problem_#.sql. Include comments and a transcript of running the queries in your PDF file.

When you are ready to submit, gather all of the following files in a single submission directory:

- Problem_Set_2.pdf
- problem_1.sql

Once your submission directory is properly assembled, **with no extraneous files**, execute the following script from your submission directory:

```
/usr/class/cs145/bin/submit-pset
```

Be sure to select "Pset2" when the script prompts you for which assignment you're submitting!

You may resubmit as many times as you like; however, only the latest submission and timestamp will be saved, and we will use your latest submission for grading your work and determining any late penalties that may apply. Submissions via email will not be accepted!