CS 122A: Introduction to Data Management – Spring 2016

Homework 6: Relational Design Theory (75 points)

Due Date: **Thursday, May 26, 2016 11:45 PM, on EEE**

Relational Design Theory [75 pts]

1. [20 pts] Suppose you are given a relation R with four attributes ABCD. For each of the following sets of FDs, assuming those are the only dependencies that hold for R, do the following: (a)Identify the candidate key(s) for R. (b)Identify the best normal form that R satisfies (1NF, 2NF, 3NF, or BCNF), and point out the dependency that violates the normal form.

1. C->D,C->A,B->A
2. A->B,B->C,D->A
3. AB->D,D->B
4. A->B,A->C, C->D
5. BC->A, BC->D, A->C, D->B

1. BC, not 2NF, C->D
2. D, not 3NF, A->B, A->C
3. ACD, ABC, not BCNF, AB->D
4. A, not 3NF, A->C, C->D
5. AB,AD,BC,CD, not BCNF, A->C, D->B

2. [20pts] Answer following questions:

(1) Give a set of FDs for the relation schema R(A,B,C,D) with primary key AB under which R is in 1NF but not in 2NF.

Consider the set of FD: AB→CD and B→C.

AB is obviously a key for this relation since AB → CD implies AB → ABCD. It is a primary key since there are no smaller subsets of keys that hold over R(A,B,C,D). The FD: B → C violates 2NF since:

C ∈ B is false; that is, it is not a trivial FD

B is not a superkey

C is not part of some key for R

B is a proper subset of the key AB (transitive dependency)

(2) Give a set of FDs for the relation schema R(A,B,C,D) with primary key AB under which R is in 2NF but not in 3NF.

Consider the set of FD: AB→CD and C→D.

AB is obviously a key for this relation since AB → CD implies AB → ABCD. It is a primary key since there are no smaller subsets of keys that hold over R(A,B,C,D). The FD: C → D violates 3NF but not 2NF since:

D ∈ C is false; that is, it is not a trivial FD C is not a superkey

D is not part of some key for R

3. [10 pts] Suppose we have a relation R with 6 attributes ABCDEF, part of its instances are listed below:

A	B	C	D	E	F
1	2	4	6	3	6
2	1	5	7	9	7
3	2	4	6	3	8
4	1	4	0	9	9

~~**Try to infer all the function dependencies in relations R, and listed as follow.**~~

Write down a ***minimal*** set of functional dependencies with at most two attributes on the left-hand side that are not violated by this instance. By "minimal" we mean no FDs in the set can be derived by the other FDs in the set. For instance, the set {X → Y, Y → Z, X → Z} is not minimal since the last FD can be derived by the first two. The set {X->Y, Y-> Z} is minimal.

(A → BCDEF and F → A, or , F → ABCDE and A → F), B → E, E → B, D → BCE, BC → D

To solve this question, you first need to know how to check X → Y. If X → Y, this means X = x1, the value of Y in the same record will always be Y = y1, and we will not have x1 → y1 and x1 → y2 mapping in one relation. However, it's possible that x1 → y1 and x2 → y1.

With this condition, we can easily know A → BCDEF, and F → ABCDE, since no duplicate values in A and F. Then we start from 1 attribute on left, and other attributes on right. Check condition for B → C, B → D, B → E, C → B ... We don't need to include A and F on the right side, because the values in these two column are distinct and the condition will always be true. After first round, we move on to 2 attributes on left, BC → D, BC → E, BD → C, ... You may have BD → CE and CD → BE evaluated as true with the condition, but they are redundant with D → BCE. So they will be removed from the result set.

4. [25pts] Consider the attribute set R = ABCDEGH and the FD set F = {AB -> CD, AC -> B, AD -> E, B -> C, BD -> A, E -> G}.

For each of the following attribute sets, do the following: (i) Compute the set of dependencies that hold over the set (ii) Name the strongest normal form that is not violated by the relation containing these attributes.

(a) ABC,

AC → B, B → C

3NF , AB, AC

(b) ABCD,

1NF

AC → B, B → C, AB → CD, BD → A

(c) ABCEG,

1NF

AC → B, B → C, E → G

(d) DCEGH,

1NF

E → G

(e) ACEH

BCNF None FDs OR 1NF, AC → E

*As we didn't specify that you cannot use FDs which include other attributes to make dependency reasoning, some of you use AC → B, AB → CD, AD→E to get AC→E. In this case, both answers are correct. Same for the extra credit part.

(b) Which of the following decompositions of R = ABCDEGH, with the same set of dependencies F, is dependency-preserving? Explain why.

(a) {AB, BC, ABDE, EG }

NO. AC → B is not preserved.

(b) {ABC, ACDE, ADG }

NO. E → G is not preserved.

Extra Credits

[10pts] Decompose each of the attribute sets in Question 4(a) into a collection of BCNF relations if it is not in BCNF.

(1) No dependency preserving decomposition/AB,BC (Not dependency preserving)
(2) No dependency preserving decomposition/ABD,BC (Not dependency preserving)
(3) No dependency preserving decomposition/ABE,BC,EG (Not dependency preserving)
(4) DCEH, EG
(5) ACEH is already BCNF/ACE ACH