**Solution for Homework 6**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Maximum Points:** 100 points

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Problem 1 [50 pts]

Consider the attribute set *R* = *ABCDEGH* and the FD set *F* = *{AB →* C, *AC →* *B, AD →E,*

*B →D, BC→A,E→G}.*

1. For each of the following five attribute sets, do the following: (i) compute the set of dependencies that hold over the set and write down a minimal cover, if the current cover is not minimal. (ii) Name the strongest normal form that is not violated by the relation containing these attributes. (iii) Decompose it into a collection of BCNF relations if it is not in BCNF.
   1. *ABC,* (b) *ABCD,* (c) *ABCEG,* (d) *DCEGH,* (e) *ACEH*
2. Which of the following decompositions of *R* = *ABCDEC,* with the same set of dependencies

*F,* is dependency-preserving? Lossless-join?

* 1. *{AB, BC, ABDE, EC}*
  2. *{ABC, ACDE, ADC}*

**Problem 1 Answer**

1. (a) i. *R1* = *ABC:* The FDs are *AB→*C, *AC→ B, BC→ A*
   1. This is already a minimal cover.
   2. This is in BCNF since *AB, AC* and *BC* are candidate keys for *R1.*

In fact, these are all the candidate keys for *R1.*

* 1. Given it is already in BCNF, no decomposition is needed.

1. i. *R2= ABCD.* The FDs are *AB→*C, *AC→B, B→ D, BC→A.*
2. This is a minimal cover already.
3. The keys are: *AB, AC, BC. R2* is not in BCNF. R2 also is not in 3NF, because Dis not part of any key. It is not even in 2NF because of the FD: *B→ D,* because B is a proper subset of a key. However, it is in lNF.
4. Decompose into *R1=ABC* and *R2=BD using B→ D.* AB is candidate key for the

first relation, and B is candidate key for the second relation. Both are in BCNF.

* 1. *R3* = *ABCEG:* The FDs are *AB→*C, *AC→B, BC→A, E→G, AB→E.*
  2. This is in minimal cover already.
  3. The keys are: *AB, AC, BC.* It is not in BCNF, because E is not a key. It is not in 3NF, because for example G is not part of any key.
  4. Decompose as follows. First use *E→ G* to decompose into ABCE and EG. EG is in BCNF with E its key. ABCE has as its candidate keys AB, AC, and BC; and thus it is in BCNF.

1. i. *R4* = *DCEGH;* The FD is *E→ G.*
2. This is in minimal cover already.
3. The key is *DCEH;* It is not in BCNF since in the FD E→G, E isnot a key. It is also not in 3NF, because G is not part of a key. Since E is a subset of the key, it is not even in 2NF either. It is, however in 1NF.
4. Decompose into *DCEH, EG* by using *E→G.*
   1. *RS* = *ACEH;* FDs projected from the F for R to RS include : AC*→* E
   2. This is a minimal cover.
   3. Key is *ACE*.
   4. It is not in BCNF form, because AC is a partial key and not a full key of RS.

**Problem 1 Answer Continued.**

**Which of the following decompositions of R = ABCDEG, with the same set of dependencies**

FDsetF= *{AB→*C, *AC→B,AD→E,B→D,BC→A,E→G}* **is**

**(a) dependency-preserving and (b) a lossless-join?**

1. **{AB, BC, ABDE, EG** }
2. **{ABC, ACDE, ADG}**

**Answer:**

* 1. The decomposition. *{* AB, BC, ABDE, EG *}* is *not* lossless. To prove this, consider the following instances of R = **ABCDEG:** *{* (a1, *b,* c1, d1, e1, g1), (a2, *b,* c2, *c2, e2, g2,) }*

Because of the functional dependencies *BC→A* and *AB→*C and *AC→B, we notice that*

a1!=a2if and only if c1!*=* c2.

Thus the projection of the above relation will lead to

AB= *{* (a1, *b), (a2, b) } and BC* = *{ (b,* c1), *(b, c2) } etc*

It is easy to see that the join AB *JOIN* BC contains 4 tuples:

*{(a1, b,* c1),

(a1, *b,* c2),

*(a2, b,* c1),

(a2, *b, c2)}*

So the join of *AB, BC, ABDE* and *EC* will also contain at least 4 tuples, actually it contains 8 tuples. Thus we have a lossy decomposition. That is, we would produce many more than the original 2 tuples that were modeled by the initial base relation R before it got decomposed.

The given decomposition does not preserve the FD : *AB→* C (nor *AC→B);* thus it is not dependency-preserving.

* 1. The decomposition *{ABC,* ACDE, ADG *}* is lossless.

For ABC, we have *{AB→*C, *AC→B, BC→A}. Thus AB, AC and BC are candidate keys.*

For ACDE, we have *AC→B, B→D,* thus *AC→D. Also, we have AD→E.*

*AC is key of ACDE.*

For ADG, we have *B→ D. We have AD→ E, E→G then AD→G.* Thus AD is candidate key.

Intuitively, the join of *ABC, ACDE* and *ADG* can be constructed in two steps.

First we construct the join of ABC and ACDE: this is lossless because their (attribute) intersection is AC which is a key for *ABCDE* (in fact *ABCDEG)* so this is lossless.

Next, we join this intermediate join result with *ADG.* This is also lossless because the attribute intersection is *AD,* and *AD→G,* that is, AD is key in the 3rd decomposed relation. So by the test mentioned above, this step is also a lossless decomposition.

The projection of the FDs of *R* onto ABC gives us: *AB→*C, *AC→B* and *BC→A.* The projection of the FDs of *R* onto *ACDE* gives us: *AD→E* and the projection of the FDs of *R* onto *ADG* gives us: *AD→G* (by transitivity). The closure of this set of dependencies does not contain *E→G* nor does it contain *B→ D.* So this decomposition is not dependency preserving.

**Problem 2 [50 pts]**

Suppose you are given a relation *R (A,B,C,D).* For each of the following sets of FDs, assuming they are the only dependencies that hold for *R,* do the following: (a) Identify the candidate key(s) for *R.* (b) State whether or not the proposed decomposition of *R* into smaller relations is a good decomposition and briefly explain why or why not.

1. *B →*C, *D →A;* decompose into *BC* and *AD.*
2. *AB→*C, C*→A,* C*→D,* decompose into *ACD* and *BC.*
3. *A →BC,* C*→AD,* decompose into *ABC* and *AD.*
4. *A →B, B →*C, C*→D,* decompose into *AB* and *ACD.*

**Answer**

1. Candidate key(s): *BD.* The decomposition into *BC* and *AD* is unsatisfactory because it is lossy, as the join of *BC* and *AD* is the Cartesian product which could be much bigger than *ABCD.*
2. Candidate key(s): AB, BC. The decomposition into ACD and BC is lossless since ACD∩ BC (which is C) →ACD. The projection of the FDs on ACD includes C→D, C→A (so C is a key for ACD) and the projection of FD on BC produces no nontrivial dependencies. In particular this is a BCNF decomposition (check that R is not!). However, it is not a dependency-preserving since the dependency AB →C is not preserved.
3. Candidate key(s): *A,* C. Since *A* and C are both candidate keys for *R,* it is already in BCNF. So from a normalization standpoint it makes no sense to decompose *R.*
4. Candidate keys: A. The projection of the dependencies on AB are: A→B (with A key) and those on ACD are A*→*C and C→D(with A again key). The schema ACD is not even in 3NF, since C is not a superkey, and D is not part of a key. This is a lossless-join dcomposition (since A is a key), but not dependency preserving, since B→C is not preserved.