Jaypee Institute of Information Technology

Database Systems & Web (15B11CI312)

Tutorial 10 (Normalization)

Q1. 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. $B \rightarrow C$, $D \rightarrow A$; decompose into BC and AD.
2. $AB \rightarrow C$, $C \rightarrow A$, $C \rightarrow D$; decompose into ACD and BC.
3. $A \rightarrow BC$, $C \rightarrow AD$; decompose into ABC and AD.
4. $A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$; decompose into AB and ACD.
5. $A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$; decompose into AB, AD and CD.

Solution:

1. Candidate key(s): BD. The decomposition into BC and AD is unsatisfactory because it is lossy (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 $ACD \cap BC$ (which is C) $\rightarrow ACD$. The projection of the FD's on ACD include C $\rightarrow$ D, C $\rightarrow$ 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 dependency preserving since the dependency AB $\rightarrow$ C is not preserved. So to enforce preservation of this dependency (if we do not want to use a join) we need to add ABC which introduces redundancy. So implicitly there is some redundancy across relations (although none inside ACD and BC).
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. Further more, the decompose is not dependency-preserving since $C \rightarrow AD$ can no longer be enforced.
4. Candidate key(s): A. The projection of the dependencies on AB are: $A \rightarrow B$ and those on ACD are: $A \rightarrow C$ and $C \rightarrow D$ (rest follow from these). The scheme 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 decomposition (since A is a key), but not dependency preserving, since $B \rightarrow C$ is not preserved.
5. Candidate key(s): A (just as before) This is a lossless BCNF decomposition (easy to check!) This is, however, not dependency preserving (B *consider* $\rightarrow C$). So it is not free of (implied) redundancy. This is not the best decomposition (the decomposition AB, BC, CD is better.)

Q2. Consider the following collection of relations and dependencies. Assume that each relation is obtained through decomposition from a relation with attributes ABCDEFGHI and that all the known dependencies over relation ABCDEFGHI are listed for each question. (The questions are independent of each other, obviously, since the given dependencies over ABCDEFGHI are different.) For each (sub)relation: (a) State the strongest normal form that the relation is in. (b) If it is not in BCNF, decompose it into a collection of BCNF relations.

1. $R1(A,C,B,D,E)$, $A \rightarrow B$, $C \rightarrow D$
2. $R2(A,B,F)$, $AC \rightarrow E$, $B \rightarrow F$
3. $R3(A,D)$, $D \rightarrow G$, $G \rightarrow H$
4. $R4(D,C,H,G)$, $A \rightarrow I$, $I \rightarrow A$
5. $R5(A,I,C,E)$

Solution:

1. 1NF. BCNF decomposition: AB, CD, ACE.
2. 1NF. BCNF decomposition: AB, BF
3. BCNF.
4. BCNF.
5. BCNF.

Q3. 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). (c) If R is not in BCNF, decompose it into a set of BCNF relations that preserve the dependencies.

1. $C \rightarrow D$, $C \rightarrow A$, $B \rightarrow C$
2. $B \rightarrow C$, $D \rightarrow A$
3. $ABC \rightarrow D$, $D \rightarrow A$
4. $A \rightarrow B$, $BC \rightarrow D$, $A \rightarrow C$
5. $AB \rightarrow C$, $AB \rightarrow D$, $C \rightarrow A$, $D \rightarrow B$

Solution:

1. (a) Candidate keys: B
   (b) R is in 2NF but not 3NF.
   (c) $C \rightarrow D$ and $C \rightarrow A$ both cause violations of BCNF. One way to obtain a (lossless) join preserving decomposition is to decompose R into AC, BC, and CD.

2. (a) Candidate keys: BD
   (b) R is in 1NF but not 2NF.
   (c) Both $B \rightarrow C$ and $D \rightarrow A$ cause BCNF violations. The decomposition: AD, BC, BD (obtained by first decomposing to AD, BCD) is BCNF and lossless and join-preserving.

3. (a) Candidate keys: ABC, BCD
   (b) R is in 3NF but not BCNF.
   (c) $ABCD$ is not in BCNF since $D \rightarrow A$ and D is not a key. However if we split up R as AD, BCD we cannot preserve the dependency $ABC \rightarrow D$. So there is no BCNF decomposition.

4. (a) Candidate keys: A
   (b) R is in 2NF but not 3NF (because of the FD: $BC \rightarrow D$).
   (c) $BC \rightarrow D$ violates BCNF since BC does not contain a key. So we split up R as in: BCD, ABC.

5. (a) Candidate keys: AB, BC, CD, AD
   (b) R is in 3NF but not BCNF (because of the FD: C $\rightarrow$ A).
   (c) $C \rightarrow A$ and $D \rightarrow B$ both cause violations. So decompose into: AC, BCD but this does not preserve $AB \rightarrow C$ and $AB \rightarrow D$, and BCD is still not BCNF because $D \rightarrow B$. So we need to decompose further into: AC, BD, CD. However, when we attempt to revive the lost functioanl dependencies by adding ABC and ABD, we that these relations are not in BCNF form. Therefore, there is no BCNF decomposition.

Q4. Consider the following two sets of functional dependencies: F = {A -> C, AC ->D, E -> AD, E -> H} and G = {A -> CD, E -> AH}. Check whether they are equivalent.