CS 353 HW 5

Section: 1
Utku Kurtulmuş - 21903025

Q.1 [12 pts, 6 pts each] Given relation R(A, B, C, D, E), determine whether the decomposition of R into ABC and ADE is lossless with the following set of functional dependencies:

(a) F = {A → C, A → D, E → D}.

(b) F = {A → B, A → D, D → E}.

a) R1 = ABC and R2 = ADE

TEST: if R1 ∩ R2 → R1, or R1 ∩ R2 → R2 the composition is lossless.

R1 ∩ R2 = A, A → ACD, since ACD != R1 or R2 the composition is not lossless.

b) R1 = ABC and R2 = ADE

TEST: if R1 ∩ R2 → R1, or R1 ∩ R2 → R2 the composition is lossless.

R1 ∩ R2 = A, A → ABDE since R1 ∩ R2 → R2 the composition is lossless.

Q.2 [20 pts] Given a relation R (A, B, C, D, E) with FDs A → BC, B → D, C → E.

(a) [5 pts] Determine if A → E holds on R.

(b) [5 pts] Determine if B → E holds on R.

(c) [10 pts] Determine if R in BCNF. If not, decompose it into BCNF relations using the BCNF decomposition algorithm discussed in the class. Indicate which FD violates BCNF in each step of decomposition.

a) A → A and A → BC ⇒ A → ABC ⇒ A → B and A → C,
A → B and B → D ⇒ A → D ⇒ A → ABCD,
A → C and C → E ⇒ A → E ⇒ A → ABCDE, A is a candidate key,
therefore part (a) is correct A → E holds on R.

b) For B we have: B → B and B → D ⇒ B → BD,
For D we have: D → D, therefore for B we have only B → BD,
B → E doesn't hold on R.

c) For the A → BC, A is a super key we proved it in part A, hence does not violate BCNF.
For the B → D, B is not a super key and it can only determine itself and D, and B → D is not a trivial FD, hence it violates BCNF.

For the C → E, this is not a trivial FD and C is not a super key, E can determine only itself hence we have C → CE. Therefore, C → E violates BCNF.

We will start the decomposition by using C → E first:

We can decompose R into, R1 = CE and R2 = ABCD, R1 is in BCNF C is a super key, for R1, but for the R2 we need to check B → D, B → D still violates R2, we will decompose R2 into R3 = BD and R4 = ABC.

In the final we have decomposed R into, R1 = CE, R3 = BD, and R4 = ABC

Q.3 [20 pts] Given a relation R(A, B, C, D, E, F, G) and its functional dependencies:
AD → F, AE → G, DF → BC, E → C, G → E
(a) [10 pts] Find the candidate key(s) of R. Show how you derived the key(s).
(b) [5 pts] Check if R is in BCNF. Why or why not?
(c) [5 pts] Check if R is in 3NF. Why or why not?

(a)

From the given FDs we can see that A is not determined by any attribute, D is not determined by any attribute. So A and D must a part of any candidate keys.

Then we will first check if AD is a primary key if so, it will be the candidate key since it is minimal, the candidate key has to include A and D and the minimal form is AD.

For AD → F , we have AD → ADF,
For AE → G, we have AE → AEG,
For DF → BC, we have DF → DFBC
For E → C, we have E → EC
For G → E, we have G → GE. (just added the trivial dependencies)

AD → ADF and DF → DFBC ⇒ AD → ADFBC or ABCDF, E and G is missing.
We can't use any of the left of the FDs. AE, E, and G is not exist in ADFBC.
Hence AD is not a super key.

We need E and G, but since G → GE, we can have:
ADG → ADFBCGE or ABCDEFG, hence ADG is minimal and primary key and one of the candidate keys. But we need to check ADE too, and decide if it is primary key.

With E added to the relation, we have ADE → ADFBCE, G looks missing from the relations related the E, E → EC is already in the relation. However, now we have AE → AEG, therefore ADE also determines G. Hence, ADE is also a candidate key.

Candidate Keys: ADG and ADE

(b) $AD \rightarrow F$ is not trivial and AD is not the candidate key therefore it violates BCNF, R is not in BCNF we don't need the check for the other FDs.

(c) $AD \rightarrow F$ is not trivial and AD is not the candidate key and F is not part of a candidate key. Therefore, it violates 3NF.

Q.4 [24 pts] Given the relation schema R(A, B, C, D) with the functional dependency set
$F = \{AB \rightarrow C, A \rightarrow E, C \rightarrow DE, D \rightarrow BE\}$
(a) [10 pts] Find a Canonical Cover Fc of F.
Show all your work.
(b) [14 pts] Check if R is in 3NF. If not, decompose it into 3NF relations using the lossless and dependency-preserving decomposition algorithm that makes use of the canonical cover you found in part (a).

(a)
START: $Fc = F = \{AB \rightarrow C, A \rightarrow E, C \rightarrow DE, D \rightarrow BE\}$
we can't use: $\alpha 1 \rightarrow \beta 1$ and $\alpha 1 \rightarrow \beta 2$ with $\alpha 1 \rightarrow \beta 1\ \beta 2$ relation.

Removing from the LHS:
For $AB \rightarrow C$, check if A is extraneous, delete A, is $B \rightarrow C$ implied by other FDs?
No, therefore A is not extraneous in $AB \rightarrow C$.
Check if B is extraneous, Delete B, is $A \rightarrow C$ implied by other FDs?
No, therefore A is not extraneous in $AB \rightarrow C$.

$A \rightarrow E$, no extraneous attribute.
$C \rightarrow DE$, no extraneous attribute.
$D \rightarrow BE$, no extraneous attribute.

Removing from the RHS:
For $AB \rightarrow C$ we can't remove attribute from right handside.
$A \rightarrow E$, no extraneous attribute.

$C \rightarrow DE$, Remove E from DE, we have $C \rightarrow D$,
we can still refer $C \rightarrow E$ since $C \rightarrow D$ and $D \rightarrow BE \Rightarrow C \rightarrow DBE$

$Fc = \{AB \rightarrow C, A \rightarrow E, C \rightarrow D, D \rightarrow BE\}$

$D \rightarrow BE$, Remove E from BE, we have $D \rightarrow B$,
can we still refer to $D \rightarrow E$? no. we can't remove E from BE.
Remove B from BE, we have $D \rightarrow E$,
can we still refer to $D \rightarrow B$? no we can't remove B from BE.

from the first round we have
Fc = Fc = {AB → C, A→ E, C→D, D → BE}
if we repeat this process Fc won't change hence Fc is in the canonical form.

(b) To check if the given relation is in 3NF we first need to find the candidate keys.,
R(A,B,C,D,E), I assume there was a typo since the FDs include E.
F = {AB → C, A→ E, C→DE, D → BE}

From the given FDs we can see that A is not determined by any of the FDs. Therefore it must be a part of the candidate key(s).

First Let's check if the A is a super key, therefore the only candidate key.

For AB → C ⇒ AB → ABC
For A→ E ⇒ A → AE
For C→DE ⇒ C → CDE
For D → BE ⇒ D → DBE (just added the trivial dependencies)

A is not a super key, A → AE, and E → E. B,C,D are missing.
Check for AB: AB → ABC, and C → CDE ⇒ AB → ABCDE, AB is a super key and a candidate key.

Check for AC: A → AE, and C → CDE ⇒ AC → ACDE,
AC → ACDE and D → DBE ⇒ AC → ABCDE, AC is a super key and a candidate key.

Check for AD: A → AE, and D → DBE ⇒ AD → ADBE,
AD → ABDE and AB → C ⇒ AD → ABCDE. AD is also a super key and a candidate key.

Candidate keys: AB, AC, AD

We will check for 3NF:
AB → C, AB is a candidate key, therefore does not violate 3NF.
A → E, A is not a candidate key and E is not part of a candidate key, therefore it violates 3NF.

Fc = {AB → C, A→ E, C→D, D → BE}

The for loop generates following 3NF schema:
R1 = ABC, R2 = AE, R3 = CD, R4 = DBE

We observe that at least one of the schemas contains a candidate key for the original relation. R1 contains AB.

At the end of the for loop delete schemas, which are subsets of other schemas:
There isn't a scheme that is a subset of one of the schemas.

Therefore the resulting schemas are:
(A,B,C), (A,E), (C,D), (D,B,E)

Q.5 [24 pts, 12 pts each] The following relation is used to maintain some information about the employees of a company. In addition to the years each employee worked in each particular section, various emails and addresses used by the employee are also stored in the relation.

Employee(id, name, from, to, section, email, address)

The following functional dependencies and multivalued dependencies are given for the Employee relation:

id → name
id, from, to → section
id→>> email
id→>> address

(a) Check if Employee is in BCNF. If not, decompose it into BCNF relations.
(b) Using the multivalued dependencies, check if each relation you found in part (a) is in 4NF. If not, decompose it into 4NF relations.

(a) id → name, is not trivial and id is not a super key. Therefore the relation is not in BCNF.
We will separate R into R1 = (id,name) and R2 = (id,from,to,section, email, address)

. We can check for the id, from, to → section for the R2 since R1 is in BCNF.
id,from,to is non-trivial and it is not the super-key due to multivalued dependencies.
Hence, we will separate R2 into R3 = (id,from,to,section)
and R4 = (id, from, to, email, address)

R3 is in BCNF. R4 is also in BCNF since the BCNF doesn't deal with multivalued dependencies.

(b) id $\rightarrow$ name $\Rightarrow$ id $\twoheadrightarrow$ name, id,from,to $\rightarrow$ section $\Rightarrow$ id,from,to $\twoheadrightarrow$ section
MVDs: { id $\twoheadrightarrow$ name, id $\twoheadrightarrow$ email, id $\twoheadrightarrow$ address , id,from,to $\twoheadrightarrow$ section}

Examine R1(id,name) :
R1 is in 4NF. Since id $\twoheadrightarrow$ name, and id+ = id,name. id is a super key for R1.

Examine R3(id,from,to,section):
R3 is in 4NF. Since id,from,to $\twoheadrightarrow$ section, and {id,from,to}+ = id,from,to,section {id,from,to} is a superkey for R2.

Examine R4(id,from,to,email,address):
R4 is not in 4NF. Since id $\twoheadrightarrow$ email, and id is not a superkey for R4. And id $\twoheadrightarrow$ email is not trivial.

We will separate R4 into two: R5 = (id,email) and R6 = (id,from,to,address)
R5 is in 4NF. But we need to check R6. R6 is not in 4NF. Since id $\twoheadrightarrow$ address, but id is not a super key, and id $\twoheadrightarrow$ address not trivial.

We will separate R6 into two: R7 = (id,address) and R8 (id,from,to). Both R7 and R8 is in 4NF now. We are done.

The decomposed relations, with re-ordering of numbers:
R1 = (id,name)
R2 = (id,from,to,section)
R3 = (id, email)
R4 = (id,address)
R5 = (id,from,to)