Alp Tuğrul Ağçalı

21801799

Section-2

Q1-) To determine whether the decomposition of R is lossless, we first need to find common attributes of R1 and R2. In our example common attribute is A.

a-) F = {A → C, A → D, E → D}

We see that A → C and A → D. So, we can say that A → ACD by Union Rule and ACD != R1 || R2. So, decomposition is not lossless.

b-) F = {A → B, A → D, D → E}.

We see that A → D and D → E. So, A → E by Transitivity Rule. A → ADE by Union Rule Rule. ADE = R2 so, this composition is lossless.

Q2-)

a-) By using Decomposition Rule if A → BC we can say that A → C. And we know that C → E. So, using Transitivity Rule A → E holds.

b-) We only have B → D. So, B → E does not hold.

c-) R is not BCNF and initially all of the FDs violate BCNF.

1. ABCDE (A → BC violates)
2. ABC (BCNF) & ADE (BCNF Because, in A → BC FD, A is super key. (A → B by Decomposition and B → D by transitivity A → D, A → C by Decomposition and C → E. by transitivity A → E))

Q3-)

a-) To find candidate key(s) we should try all functional dependencies. If we cannot find a candidate key, we should combine them to find minimal key.

Firstly, we should try the shortest possibilities.

(E)+ = EC (E → C) (-)

(G)+ = GEC (G → E), (E → C) (-)

(AD)+ = ADFBC (AD →F), (DF → BC) (-)

(AE)+ = AEGC (AE → G), (E → C) (-)

(DF)+ = DFBC (DF → BC) (-)

(ADE)+ = ABCDEFG (AD → F), (DF → BC), (AE → G) (+)

(ADG)+ = ABCDEFG (AD ⟶ F), (DF ⟶ BC), (G ⟶ E) (+)

(AEG)+ = AEGC (E → C) (-)

(DFE)+ = BCDFE (DF → BC) (-)

(DFG)+ = BCDFEG (DF → BC), (G → E) (-)

As you can see, only ADE and ADG have + near them, so they are candidate keys.

b-) We found that none of our FDs is a super key and none of them is trivial. So, R is not in BCNF.

C-) R is not BCNF, so we do not need to check the first two condition on 3NF. We only check that for a → b FD, each attribute x in b-a is part of candidate key.

AD → F, F is not part of candidate key, so this FD violates 3NF.

Q4-)

a-) Firstly, there are no combinations in FDs.

1. B is extraneous in AB → C because without it we can imply by A → C, C → DE, D → BE. Set is { A → C, A→ E, C→DE, D → BE}
2. If we remove E from C→DE we can imply E using A → E. Set is {A → C, A → E, C → D, D → BE}
3. We can remove A → E because it is not changing F+. Set is {A → C, C →D, D → BE }

b-) we know that A is the candidate key of R.

1. None of FDs are trivial.
2. Not all of FD's left side is super key.
3. C is not part of candidate key. (from FD A → C).

So, R is not 3NF.
Decomposing 3NF:

1. A ⟶ C: No schema contains A and C, R1 = AC.
2. A ⟶ E: No schema contains A and E, R2 = AE.
3. C ⟶ DE: No schema contains C and DE, R3 = CDE.
4. R1 and R2 contains the candidate key Aa.
5. No schema contained by another.
6. R1, R2, R3 are decompositions of R.

Q5-)

a-) Employee is not BCNF because, none of FDs are trivial. And except id, from, to other left sides are not super key.

Decomposing to BCNF:

1. id, name, from, to, section, email, address (id → name violates)
2. id, name (BCNF) id, from, to, section, email, address (BCNF because id, from, to is superkey and does not violate.)

b-) we know that id → name then id→→name so R1 is 4NF.

R2 is not 4NF because id →→ email and id →→ address violates

1. id, email (4NF) id, from, to, section, address (NOT 4NF)
2. id ,email (4NF), id, address (4NF) id, from, to, section (is 4NF because id, from, to →→section and id, from, to is super key.)