Efe Beydoğan

21901548

CS353 Section 2

HW5

Q1)

a) A -> B is violated because in the 1st row there is (a1, b1) but in the 2nd row there is (a1, b2).

b) B -> C is not violated because for every same B value the C values are the same.

c) C -> A is violated because in the 1st row there is (c1, b1) but in the 3rd row there is (c1, b3).

d) AB -> C is not violated as there are no two rows with the same A and B values.

e) AC -> B is violated because in the 3rd row there is (a2, c1, b3) but in the 5th row there is (a2, c1, b1).

f) BC -> A is violated because in the 1st row there is (b1, c1, a1) but in the 5th row there is (b1, c1, a2).

Q2)

a) R1 ∩ R2 = A and A -> R1 = AB holds since A -> B, hence the decomposition is lossless.

b)

R:

A	B	C
a1	b1	c1
a1	b1	c2
a2	b1	c3

R1:

A	B
a1	b1
a2	b1

R2:

B	C
b1	c1
b1	c2
b1	c3

R1 ⋈ R2:

A	B	C
a1	b1	c1
a1	b1	c2
a1	b1	c3
a2	b1	c1
a2	b1	c2
a2	b1	c3

R is a subset of R1 ⋈ R2, hence the decomposition is not lossless.

Q3)

a) C is not implied by other attributes, so C is a part of any candidate key.

C^+ -> using the algorithm for computing the closure of an attribute

result = c -> terminates after 1st turn of while loop because there is no change in result, **C is not a candidate key.**

$(AC)^+$:

result = AC

1st while:

result = result ∪ D (ACD)

result = result ∪ AB (ABCD)

2nd while:

result = result ∪ D (ABCD)

result = result ∪ E (ABCDE)

result = result ∪ AB (ABCDE)

No change in 3rd while, so terminate.

$(AC)^+$ = ABCDE => AC is a superkey. It is also minimal, so it is a candidate key.

$(BC)^+$:

result = BC

1st while:

result = result ∪ E (BCE)

2nd while:

result = result ∪ E (BCE)

No change in 2nd while, so terminate.

$(BC)^+$ = BCE => BC is not a candidate key.

$(DC)^+$:

result = DC

1st while:

result = result ∪ AB (ABCD)

2nd while:

result = result ∪ D (ABCD)

result = result ∪ E (ABCDE)

result = result ∪ AB (ABCDE)

No change in 3rd while, so terminate.

$(DC)^+$ = ABCDE => DC is a superkey. It is also minimal, so it is a candidate key.

$(EC)^+$:

result = EC

No change in 1st while, so terminate. EC is not a candidate key.

AC and DC are candidate keys.

b) R is not in BCNF, as in the nontrivial functional dependency A -> D, A is not a superkey.

c) R is not in 3NF, as in the nontrivial functional dependency BC -> E, BC is not a superkey and E – BC = E is not part of a candidate key either.

Q4)

a)

- Using the decomposition rule, we can decompose A -> BC in G as A -> B and A -> C, so A -> B in F can be inferred from G.
- Using the augmentation rule, we can augment A -> BC in G as AB -> BC, then with the decomposition rule we can decompose it into AB -> B an AB -> C, hence AB -> C in F can be inferred from G.
- Using the decomposition rule, we can decompose D -> AE in G as D -> A an D -> E. We can also decompose A -> BC as A -> B and A -> C. As D -> A and A ->C, we can use transitivity to show D -> C. Since we now have both D -> A and D -> C, we can say D -> AC, so D -> AC in F can be inferred from G.
- Using the decomposition rule, we can decompose D -> AE in G as D -> A an D -> E. Hence, D -> E in F can be inferred from G.

As a result, all of the functional dependencies in F can be inferred from G and G covers F.

b) F doesn't cover G, as the functional dependency E -> B in G cannot be obtained by the dependencies in F, E doesn't appear on the left hand side in any of the dependencies in F.

c) As F doesn't cover G, F and G are not equivalent.

Q5)

a) F = {A → BD, CD → B, C→D, B → D}

There are no functional dependencies with the same left hand side, so no replacement is done.

Eliminating extraneous attributes on the lefthand side:

for CD → B:

for C:

D^+ under F:

result = D

no changes to result in the first turn of the while loop, so $D^+ = D$

$D^+ = D$ does not contain B, so C is not extraneous in CD → B.

for D:

C^+ under F:

result = C

1st while loop:

result = result ∪ D (CD)

2nd while loop:

result = result ∪ B (BCD)

no changes to result in the 3rd while loop, so $C^+ = BCD$

$C^+ = BCD$ contains B, so D is extraneous in CD → B.

After eliminating extraneous attribute on LHS, F' = { A → BD, C → B, C→ D, B → D } and if we unite C → B and C→D, F' = { A → BD, C → BD, B → D }

Eliminating extraneous attributes on the righthand side:

for A → BD:

for B:

Compute A^+ under { A → D, C → BD, B → D }

result = A

1st while loop:

result = result ∪ D (AD)

no changes to result in the 2nd while loop, so $A^+ = AD$

$A^+ = AD$ does not contain B, so B is not extraneous in A → BD

for D:

Compute A^+ under { A → B, C → BD, B → D }

result = A

1st while loop:

result = result ∪ B (AB)

result = result ∪ D (ABD)

no changes to result in the 2nd while loop, so A^+ = ABD

A^+ = ABD contains D, so D is extraneous in A → BD.

F'' = { A → B, C → BD, B → D }

for C → BD:

for B:

Compute C^+ under { A → B, C → D, B → D }

result = C

1st while loop:

result = result ∪ D (CD)

no changes to result in the 2nd while loop, so C^+ = CD

C^+ = CD does not contain B, so B is not extraneous in C → BD.

for D:

Compute C^+ under { A → B, C → B, B → D }

result = C

1st while loop:

result = result ∪ B (BC)

result = result ∪ D (BCD)

no changes to result in the 2nd while loop, so C^+ = BCD

C^+ = BCD contains D, so D is extraneous in C → BD.

F''' = { A → B, C → B, B → D } = F_c (no other eliminations)

b) A and C don't appear on the right side of any functional dependency, so they must be part of any candidate key.

$(AC)^+$:

result = AC

1st while:

result = result ∪ BD (ABCD)

There is no change to result in the second while, so algorithm terminates.

$(AC)^+$ = ABCD, so AC is a superkey. It is also minimal, so it is a candidate key.

R is not in 3NF, as in the nontrivial functional dependency A → BD, A is not a superkey and the attributes in BD - A = BD are not part of any candidate key.

Lossless and dependency preserving decomposition:

F_c = { A → B, C → B, B → D }

Set of decomposed relation: AB, CB, BD

None of the relations contains a candidate key, so add one more relation which contains the attributes of a candidate key: AB, CB, BD, AC

Decomposed relations: AB, BC, BD, AC