**Quiz 2 [LIVINGSTON]**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ RUID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Show one key for R(ABCDEFG) and F = {A -> C, B-> D, A->F, B->G, E->F, F->G}

**Answer:**

A, B, E must be in each key, since none of them is on right hand side of any functional dependencies in F.

We start from checking if ABE -> R here: {ABE}+ = ABCDEFG = R, so we can find that ABE -> R. Therefore, ABE is one key for R.

1. Show all nontrivial and minimal fds (minimal base) which are satisfied by the following instance

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 0 | 0 | 1 |

**Answer:**

B -> C, for tuple 1&3, they agree on B and they also agree on C;

C -> B, for tuple 1&3, they agree on C and they also agree on B;

AB -> C, for tuple 1&2&3, they all have different AB values;

AC -> B, for tuple 1&2&3, they all have different AC values.

Therefore, all nontrivial FDs are F = {B -> C, C -> B, AB -> C, AC -> B}.

We start to compute the minimal basis of F:

Since all the RHS of FDs are singleton, we don’t need to split them;

Then try to remove B -> C from F, we compute {B}+ from F - {B -> C}: {B}+ = B, we can’t remove it from F;

Try to remove C -> B from F, we compute {C}+ from F - {C -> B}: {C}+ = C, we can’t remove it from F;

Try to remove AB -> C from F, we compute {AB}+ from F - {AB -> C}: {AB}+ = ABC, we can remove it from F;

Try to remove AC -> B from F, we compute {AC}+ from F - {AC -> B}: {AC}+ = ABC, we can remove it from F;

Since all the LHS of remaining FDs are singleton, we finish computing. And the minimal basis of F is {B -> C, C -> B}

1. Consider the attribute set R = ABCDEG, and the FD set F = {AB→C, AC→B, AD→E, B→D, BC→A, E→G}. Given a decomposition R1(ABC), R2(ACDE), R3(ADG):
   1. Is this decomposition dependency preserving?
   2. Is this decomposition lossless-join?

**Answer:**

a. We start from computing the projection of FDs onto R1(ABC), R2(ACDE), R3(ADG)

For R1(ABC):

{A}+ = A, {B}+ = B~~D~~, {C}+ = C;

{AB}+ = ABC~~DEG~~, we get AB -> C;

{AC}+ = ABC~~DEG~~, we get AC -> B;

{BC}+ = ABC~~DEG~~, we get BC -> A;

We can get the projection of FDs on R1 is F1 = {AB -> C, AC -> B, BC -> A}

For R2(ACDE):

{A}+ = A, {C}+ = C, {D}+ = D, {E}+ = E~~G~~;

{AC}+ = A~~B~~CDE~~G~~, we get AC -> DE and we don’t need to compute supersets of AC;

{AD}+ = ADE~~G~~, we get AD -> E;

{AE}+ = AE~~G~~, {CD}+ = CD, {CE}+ = CE~~G~~, {DE}+ = DE~~G~~, {CDE}+ = CDE~~G~~;

We can get the projection of FDs on R2 is F2 = {AC -> DE, AD -> E}

For R3(ADG):

{A}+ = A, {D}+ = D, {G}+ = G;

{AD}+ = AD~~E~~G, we get AD -> G;

{AG}+ = AG, {DG}+ = DG;

We can get the projection of FDs on R3 is F3 = {AD -> G}

Therefore, all the FDs for R1, R2, R3 are F0 = {AB -> C, AC -> B, BC -> A, AC -> DE, AD -> E, AD -> G}. Now we will find if the original FDs {AB -> C, AC -> B, AD -> E, B -> D, BC -> A, E -> G} are preserved. Since AB -> C, AC -> B, BC -> A, AD->E is in F0, we don’t need to compute them. We start from checking if B -> D is preserved. {B}+ = B, we can find that B -> D is lost. (Also, we can check if E->G is preserved. {E}+=E, so E->G is also lost. Therefore, this decomposition is not dependency preserving.

b. Chase test:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b2 | c | d | e | g2 |
| R3 | a | b3 | c3 | d | e3 | g |

AB -> C, nothing changes;

AC -> B, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | ~~b2~~b | c | d | e | g2 |
| R3 | a | b3 | c3 | d | e3 | g |

AD -> E, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b | c | d | e | g2 |
| R3 | a | b3 | c3 | d | ~~e3~~e | g |

B -> D, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | ~~d1~~d | ~~e1~~e | g1 |
| R2 | a | b | c | d | e | g2 |
| R3 | a | b3 | c3 | d | e | g |

BC -> A, nothing changes;

E -> G, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d | e | ~~g1~~g |
| R2 | a | b | c | d | e | ~~g2~~g |
| R3 | a | b3 | c3 | d | e | g |

The test result is:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d | e | g |
| R2 | a | b | c | d | e | g |
| R3 | a | b3 | c3 | d | e | g |

Therefore, this decomposition is lossless-join.

1. Let R(ABCDEG) be a relation and F = {A→B, B→C, AD→G, D→E}. Decompose R into BCNF.

**Answer:**

Step 1: A -> B causes a violation in R because (A)+ = ABC, so A is not a superkey. Break in R1(AB) and R2(ACDEG) with projected FDs be:

F1 = {A -> B}

F2 = {A -> C, D -> E, AD -> CEG, AE -> C, AG -> C, CD -> E, DG -> E, AEG -> C, CDG -> E}

Step 2: A -> C causes a violation in R2 since (A)+ = AC. Break R2 in R21(AC) and R22(ADEG) with

F21 = {A -> C}

F22 = {D -> E, AD -> EG, DG -> E}

Step 3: D -> E causes a violation in R22 since (D)+ = DE. Break R22 in R221(DE) and R222(ADG) with

F221 = {D -> E}

F222 = {AD -> G}

Final decomposition: R1(AB), R21(AC), R221(DE) and R222(ADG) with the FDs listed above {A -> B, A -> C, D -> E, AD -> G}

**Quiz 2 [BUSCH]**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ RUID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Show one key for R(ABCDEFG) and F = {A -> C, A-> D, A->E, B->G, E->F, F->G}

**Answer:**

A, B must be in each key, since none of them is on right hand side of any functional dependencies in F.

We start from checking if AB -> R here: {AB}+ = ABCDEFG = R, so we can find that AB -> R. Therefore, AB is one key for R.

1. Show all nontrivial and minimal fds (minimal base) which are satisfied by the following instance

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 1 | 0 | 1 |
| 1 | 1 | 1 |
| 0 | 1 | 1 |

**Answer:**

A -> C, for tuple 1&2, they agree on A and they also agree on C;

B -> C, for tuple 2&3, they agree on B and they also agree on C;

AB -> C, for tuple 1&2&3, they all have different AB values.

Therefore, all nontrivial FDs are F = {A -> C, B -> C, AB -> C}.

We start to compute the minimal basis of F:

Since all the RHS of FDs are singleton, we don’t need to split them;

Then try to remove A -> C from F, we compute {A}+ from F - {A -> C}: {A}+ = A, we can’t remove it from F;

Try to remove B -> C from F, we compute {B}+ from F - {B -> C}: {B}+ = B, we can’t remove it from F;

Try to remove AB -> C from F, we compute {AB}+ from F - {AB -> C}: {AB}+ = ABC, we can remove it from F;

Since all the LHS of remaining FDs are singleton, we finish computing. And the minimal basis of F is {A -> C, B -> C}

1. Consider the attribute set R = ABCDEG, and the FD set F = {AB→C, AC→B, ABD→E, D→B, BC→A, E→G}. Given a decomposition R1(ABC), R2(ACDE), R3(ADG}:
   1. Is this decomposition dependency preserving?
   2. Is this decomposition lossless-join?

**Answer:**

a. We start from computing the projection of FDs onto R1(ABC), R2(ACDE), R3(ADG)

For R1(ABC):

{A}+ = A, {B}+ = B, {C}+ = C;

{AB}+ = ABC, we get AB -> C;

{AC}+ = ABC, we get AC -> B;

{BC}+ = ABC, we get BC -> A;

We can get the projection of FDs on R1 is F1 = {AB -> C, AC -> B, BC -> A}

For R2(ACDE):

{A}+ = A, {C}+ = C, {D}+ = D~~B~~, {E}+ = E~~G~~;

{AC}+ = A~~B~~C;

{AD}+ = A~~B~~CDE~~G~~, we get AD -> CE and we don’t need to compute supersets of AD;

{AE}+ = AE~~G~~;

{CD}+ = A~~B~~CDE~~G~~, we get CD -> AE and we don’t need to compute supersets of CD;

{CE}+ = CE~~G~~, {DE}+ = ~~B~~DE~~G~~;

{ACE}+ = A~~B~~CE~~G~~;

We can get the projection of FDs on R2 is F2 = {AD -> CE, CD -> AE}

For R3(ADG):

{A}+ = A, {D}+ = D~~B~~, {G}+ = G;

{AD}+ = A~~BC~~D~~E~~G, we get AD -> G;

{AG}+ = AG, {DG}+ = ~~B~~DG;

We can get the projection of FDs on R3 is F3 = {AD -> G}

Therefore, all the FDs for R1, R2, R3 are F0 = {AB -> C, AC -> B, BC -> A, AD -> CE, CD -> AE, AD -> G}. Now we will find if the original FDs {AB -> C, AC -> B, ABD -> E, D -> B, BC -> A, E -> G} are preserved. Since AB -> C, AC -> B, BC -> A is in F0, we don’t need to compute them:

{ABD}+ = ABCDEG, so ABD -> E is preserved;

{D}+ = D, we can find that D -> B is lost;

Also {E}+ = E. So E -> G is lost as well.

Therefore, this decomposition is not dependency preserving.

b. Chase test:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b2 | c | d | e | g2 |
| R3 | a | b3 | c3 | d | e3 | g |

AC -> B, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | ~~b2~~b | c | d | e | g2 |
| R3 | a | b3 | c3 | d | e3 | g |

D -> B, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b | c | d | e | g2 |
| R3 | a | ~~b3~~b | c3 | d | e3 | g |

AB -> C, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b | c | d | e | g2 |
| R3 | a | b | ~~c3~~c | d | e3 | g |

ABD -> E, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b | c | d | e | g2 |
| R3 | a | b | c | d | ~~e3~~e | g |

BC -> A, nothing changes;

E -> G, we get:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b | c | d | e | ~~g2~~g |
| R3 | a | b | c | d | e | g |

The test result is:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1 | a | b | c | d1 | e1 | g1 |
| R2 | a | b | c | d | e | g |
| R3 | a | b | c | d | e | g |

Therefore, this decomposition is lossless-join.

1. Let R = {C, T, H, R, S, G} be a relation and F = {CS→G, C→T, TH→R, HR→C, HS→R}. Decompose R into BCNF.

**Answer:**

Step 1: CS -> G causes a violation in R because (CS)+ = CSGT, so CS is not a superkey. Break in R1(CSGT) and R2(CSHR) with projected FDs be:

F1 = {C -> T, CS -> G, CG -> T, CSG -> T, CST -> G}

F2 = {CH -> R, HS -> R, HR -> C, CSH -> R, SHR -> C}

Step 2: C -> T causes a violation in R1 since (C)+ = CT. Break R1 in R11(CT) and R12(CSG) with

F11 = {C -> T}

F12 = {CS -> G}

Step 3: CH -> R causes a violation in R2 since (CH)+ = CHR. Break R2 in R21(SCH) and R22(CHR) with

F21 = {HS -> C}

F22 = {CH -> R, HR -> C}

Final decomposition: R11(CT), R12(CSG), R21(SCH) and R22(CHR) with the FDs listed above {CS -> G, C -> T, HS -> C, CH -> R, HR -> C}

**Other Ways:**

Step 1: CS -> G causes a violation in R because (CS)+ = CSGT, so CS is not a superkey. Break in R1(CSG) and R2(CTHRS) with projected FDs be:

F1 = {CS -> G}

F2 = {C -> T, CH -> TR, CR -> T, CS -> T, TH -> RC, HR -> CT, HS -> RCT, CTH -> R, CHR -> T, CRS -> T, THR -> C}

Step 2: C -> T causes a violation in R2 since (C)+ = CT. Break R2 in R21(CT) and R22(CHRS) with

F21 = {C -> T}

F22 = {CH -> R, HR -> C, HS -> RC}

Step 3: CH -> R causes a violation in R22 since (CH)+ = CHR. Break R22 in R221(SCH) and R222(CHR) with

F221 = {HS -> C}

F222 = {CH -> R, HR -> C}

Final decomposition: R1(CSG), R21(CT), R221(SCH) and R222(CHR) with the FDs listed above {CS -> G, C -> T, HS -> C, CH -> R, HR -> C}