**<Datebase 201433707 이형욱>**

**8.1 Suppose that we decompose the schema r(A, B, C, D, E) into**

**r1(A, B, C)**

**r2(A, D, E)**

**Show that this decomposition is a lossless decomposition if the following set F of functional dependencies holds:**

**A → BC**

**CD → E**

**B → D**

**E → A**

Answer: A decomposition {R1, R2} is a lossless-join decomposition if R1 ∩ R2 → R1 or R1 ∩ R2 → R2. Let R1 = (A, B, C), R2 = (A, D, E), and R1 ∩ R2 = A. Since A is a candidate key (see Practice Exercise 8.6), Therefore R1 ∩ R2 → R1.

**8.4 Use Armstrong’s axioms to prove the soundness of the union rule. (Hint: Use the augmentation rule to show that, if α→ β, then α→αβ . Apply the augmentation rule again, using α→ γ, and then apply the transitivity rule.)**

Answer: To prove that:

if α → β and α → γ then α → β γ

Following the hint, we derive:

α → β given

α α → α β augmentation rule

α → α β union of identical sets

α → γ given

α β → γ β augmentation rule

α → β γ transitivity rule and set union commutativity

**8.5 Use Armstrong’s axioms to prove the soundness of the pseudotransitivity rule.**

Answer: Proof using Armstrong’s axioms of the Pseudotransitivity Rule:

if α → β and γ β → δ, then αγ → δ.

α → β given

α γ → γβ augmentation rule and set union commutativity

γ β → δ given

α γ → δ transitivity rule

**8.6 Compute the closure of the following set F of functional dependencies for relation schema r (A, B, C, D, E).**

**A → BC**

**CD → E**

**B → D**

**E → A**

**List the candidate keys for R.**

Answer: Note: It is not reasonable to expect students to enumerate all of F +. Some shorthand representation of the result should be acceptable as long as the nontrivial members of F + are found.

Starting with A → BC, we can conclude: A → B and A → C.

Since A → B and B → D, A → D (decomposition, transitive)

Since A → C D and C D → E, A → E (union, decomposition, transitive) Since A → A, we have (reflexive) A → ABC DE from the above steps (union)

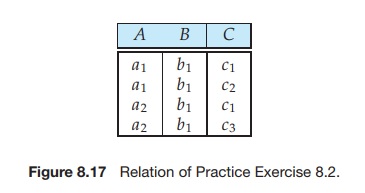
Since E → A, E → ABC DE (transitive)

Since C D → E, C D → ABC DE (transitive)

Since B → D and BC → C D, BC → ABC DE (augmentative, transitive)

Also, C → C, D → D, B D → D, etc. Therefore, any functional dependency with A, E, BC, or C D on the left hand side of the arrow is in F +, no matter which other attributes appear in the FD. Allow \* to represent any set of attributes in R, then F + is B D → B, B D → D, C → C, D → D, B D → B D, B → D, B → B, B → B D, and all FDs of the form A ∗ → a, BC ∗ → a, C D ∗ → a, E ∗ → a where a is any subset of {A, B, C, D, E}. The candidate keys are A, BC, C D, and E.

**8.7 Using the functional dependencies of Practice Exercise 8.6, compute the canonical cover Fc .**

****

Answer: The given set of FDs F is:-

A → BC

CD → E

B → D

E → A

The left side of each FD in F is unique. Also none of the attributes in the left side or right side of any of the FDs is extraneous. Therefore the canonical cover Fc is equal to F.

**8.19 Give a lossless-join decomposition into BCNF of schema R of Practice Exercise 8.1**.

Answer: First Exercise 8.6 we know that B ->D is nontrivial and the left hand side is not a superkey. By the algorithm of Figure 8.11 we derive the relations {(A,B,C,E),(B,D)} . This is in BCNF

**8.20 Give a lossless-join, dependency-preserving decomposition into 3NF of schema R of Practice Exercise 8.**1.

Answer: First we note that the dependencies given in Practice Exercise8.1 from a canonical cover, Generating the schema from the algorithm of Figure 8.12 we get

R’={(A,B,C),(C,D,E),(B,D),(E,A)}

Schema (A,B,C) contains a candidate key. Therefore R’ is a third normal form dependency-preserving lossless-join decomposition.

Note that the original schema R =(A,B,C,D,E) is already in 3NF .Thus, it was not necessary to apply the algorithm, as we have done above. The single original schema is trivially a lossless join dependency preserving decomposition.

**8.25 Consider the following proposed rule for functional dependencies: If α→β and γ→β , then α→γ . Prove that this rule is not sound by showing a relation r that satisfies α→β and γ→β , but does not satisfy α→γ** .

Consider the following rule : if A ->B and C->B, then A ->C . That is, α = A , β = B , γ = C. The following relation r is a counterexample to the rule.

r:

|  |  |  |
| --- | --- | --- |
| A | B | C |
| a1 | b1 | c1 |
| a2 | b2 | c2 |

Note: A -> B and C-> B, (since no 2 tuples have the same C value, C->B is true trivially) .However, it is not the case that A-> C since the same A value is in two tuples, but the C value is in two tuples, but the C value in those tuples disagree.

**8.26 Use Armstrong’s axioms to prove the soundness of the decomposition rule.**

If α →βγ , then α→β and α →γ

α →βγ given

βγ→β reflexivity rule

α →β transitivity rule

βγ→γ reflexive rule

α →γ transitive rule