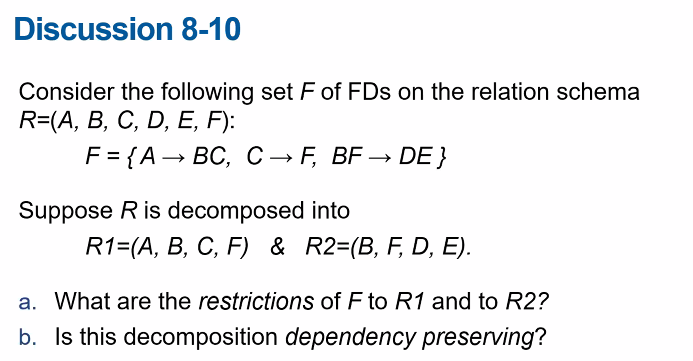
**Discussion 05/04**

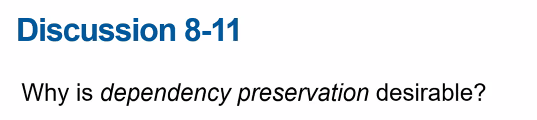


a. F1 = { A -> BC, C -> F, A -> F }

F2 = { BF -> DE }

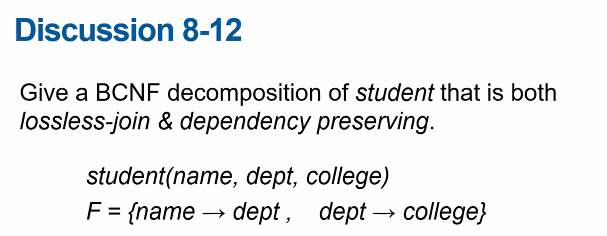
b. Yes.

=> 교수님 설명: F+ = (F1 U F2)+ 를 봐야함. 이번 경우는 F1 U F2 = F이기 때문에 Yes.



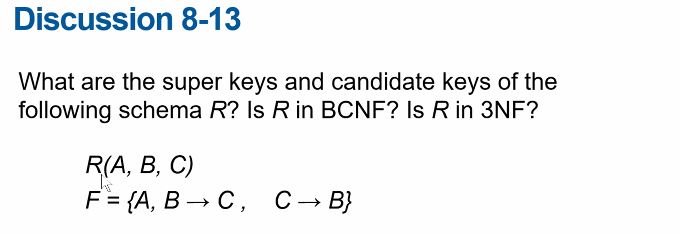
Dependency를 보존하면서 분해해야 나중에 functional dependency를 join 없이 확인할 수 있음. Join은 비싼 연산이기 때문에 안 할 수 있으면 안 하는 게 좋다.

=> Foreign key 외에는 테이블 사이에 constraints를 확인하는 것이 힘들어서 안 하는 것이 좋음.



R1 = (name, dept), R2 = (dept, college)

F1 = {name -> dept}, F2 = {dept -> college}

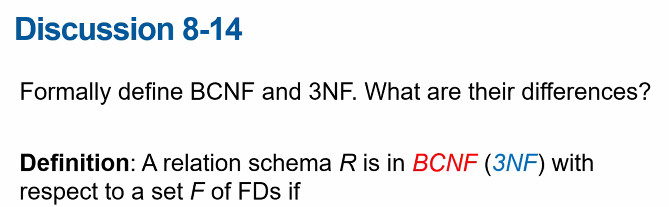


Super key = (A, B, C), (A, B), (A, C)

Candidate key = (A, B), (A, C)

BCNF는 X => C-> B에서 C가 Super key가 아니기 때문.

3NF도 X => C -> B에서 B가 candidate key에 포함되어 있기 때문.



F+의 각각의 a -> b에 대해서BCNF는 다음 중 하나를 만족해야 함.

1. a->b가 trivial.
2. a가 R의 super key.

3NF는 다음 중 하나를 만족해야 함.

1. BCNF임.
2. b – a의 각각의 attribute가 R의 candidate key에 포함되어야 함.

둘의 가장 큰 차이는 3NF는 R의 FD를 보존한다는 점. 그로 인해 redundancy가 조금 생길 수 있음.

=> 정답: 