Q1

a) True, Cause C->AB, so C->A and **C->B** //Decomposition Rule

b) False,

K L M N

R1 1 0 0 1

R2 1 1 1 1

c) TRUE

Cause A->C ,so AB->CB //augmentation

so AB->C //decomposition

cause C->D so AB->D //transitivity

so AB->CD //Union

Q2

a) Candidate Key: BD,

It is lossy, cause BC∩AD=null， which is not key (B or D)

b) Candidate Key: AB or BC .

it is lossless , cause ACD∩BC=C, and cause C->A ,C->D , C can be the key of ACD ,

It is not dependency preserving, cause ACD and BC doesn't cover AB->C

c) Candidate Key: A or C, cause C->A and A->C

it is lossless, cause ABC∩AD=A, and cause A->BC, , so A can be the key of ABC

it is not dependency preserving, cause ABC and AD doesn't cover C->AD

d) Candidate Key: A

It is lossless, cause AB∩ACD= A, and cause A->B, so A can be the key of AB

it is not dependency preserving ,cause AB and ACD doesn't cover B->C

e) Candidate Key: a

It is lossy, cause AD∩CD=D , and D cannot be key of AD or CD

3.

a)

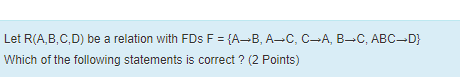
A,R,N,S, never appear at the rhs, so they must be part of key

A->B, so we don't need B

R->C so we don’t need C

NS->BT so we don't need BT

so ARNS is the only candidate key



A->D

b)

Step 1: Put FD’s into the simple form

F1 = {AB -> T, A -> B, R -> C, NS -> B, NS->T}

No Trivial, Done

Step 2: Minimize the lhs of each FD

AB->T,

try remove B, A+ = AB, therefore, we can remove B

NS->BT

try remove N, S+ = S, therefore, we cannot remove N

try remove S, N+ = N, therefore, we cannot remove S

F2 = {A ->T, A ->B, R -> C, NS -> B, NS -> T}

Step 3: Remove redundant FDS

Try remove A->T

A+ = AB, T doesn't belong to AB, not redundant

Try remove A->B

A+ = AT, B doesn't belong to AT, not redundant

Try remove R->C

R+ = R, C doesn't belong to R, not redundant

Try remove NS->B

NS+ = NST, B doesn't belong to NST not redundant

Try remove NS->T

NS+ = NSB, T doesn't belong to NSB not redundant

F3 = {A 🡪 T, A 🡪 B, R 🡪 C, NS 🡪 B, NS🡪T}

c) No, it can have multiple unique canonical covers, It just depends on the order of FDs that we consider to reduce