Jasmine Omeke

CSC 453 Database Technologies 701/710 Assignment 4 (9/28)

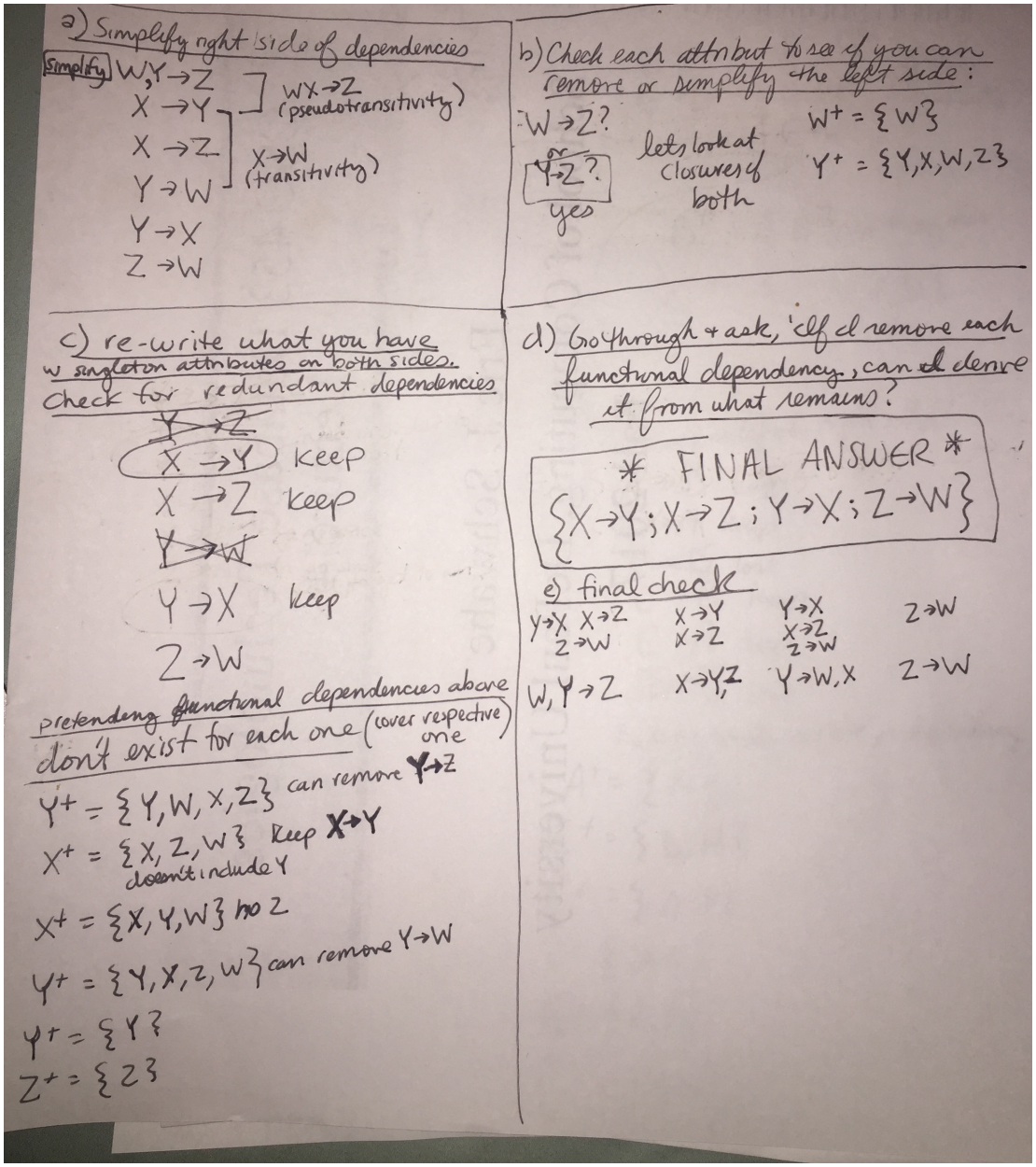
Due 11:59:00pm, Monday 10/9.

1. Find a minimal cover for the following set *F* of functional dependencies.

WY→Z X→YZ Y→WX Z→W

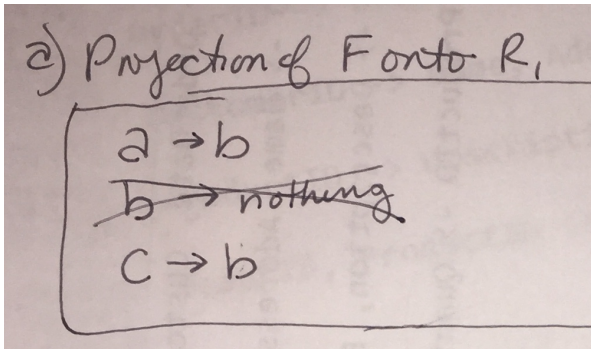
(Note that when attribute names are single letters, I will use shorthand of the form WX→YZ for the functional dependency W,X→Y,Z .)

Final Answer: { X → Y; X → Z; Y → X ; Z → W }



2. For the universal relation R(a,b,c,d), consider the decomposition D consisting of R_1(a,b,c) and R_2(b,d), and the set F of functional dependencies { a→b,d ; b→d ; c→d,b }.

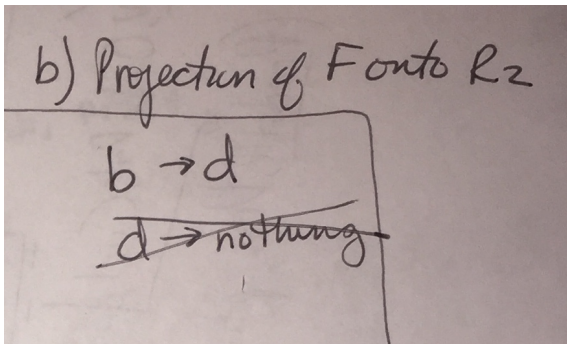
a. Compute the projection of F on R_1.



a→b

c→b

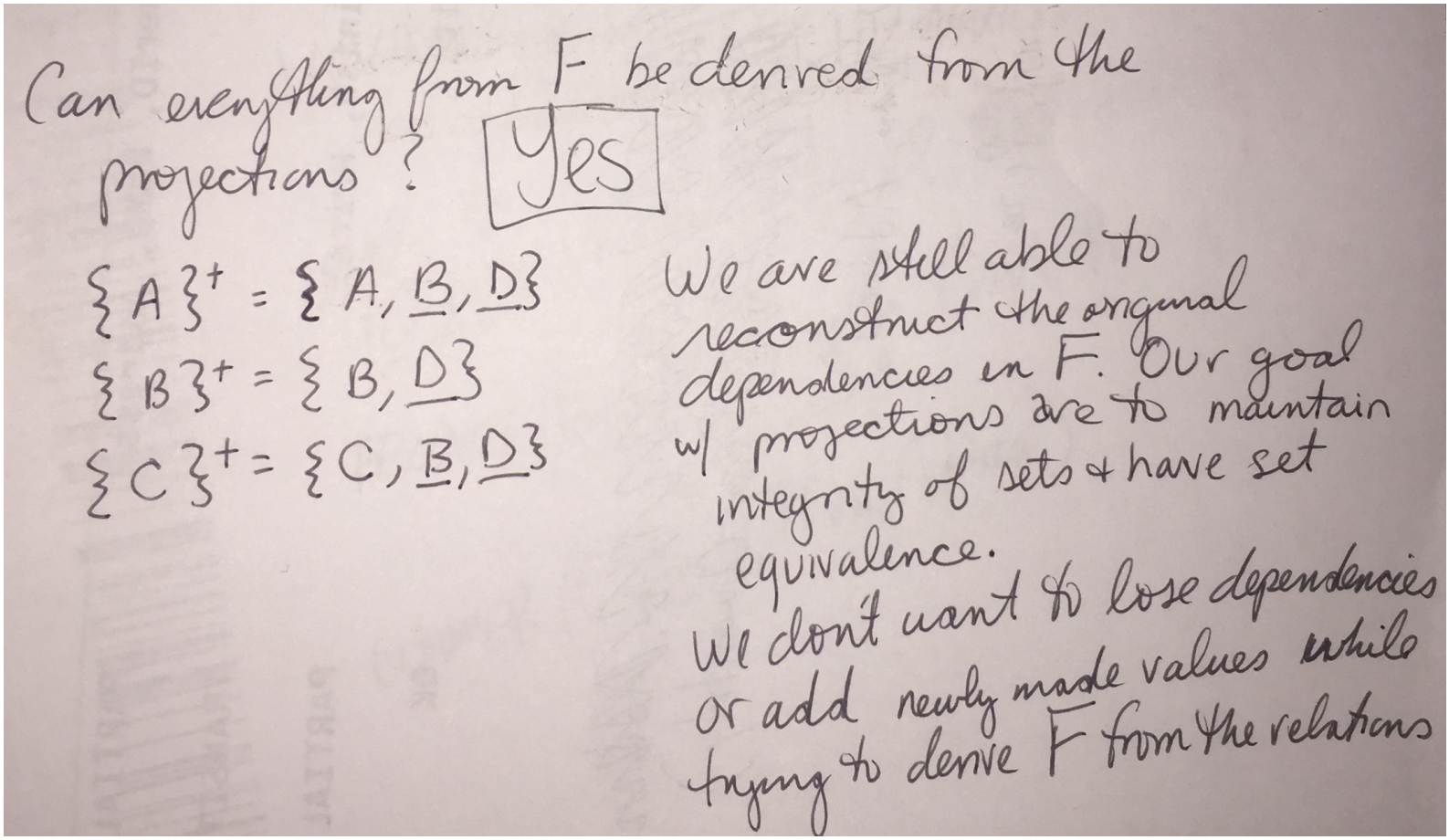
b. Compute the projection of F on R_2.



b→d

c. Does the decomposition D preserve the set of dependencies F? Give a detailed explanation why or why not. (That is, don't just repeat back the

definition of the dependency preservation property, but rather show why the decomposition *D* either has or does not have this property.)



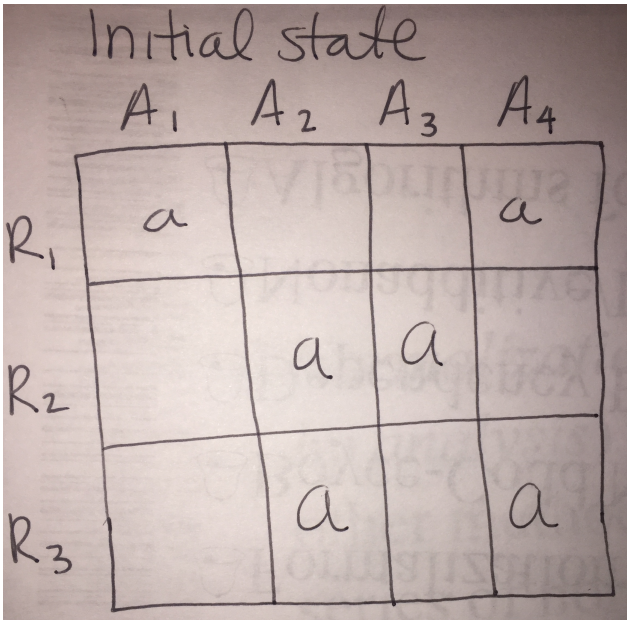
The dependencies in F are maintained because everything from one set can be derived from the other. For instance, one can get {A}+ being {A, B, D} because a→b in R^1 and b→d in R^2. Since everything in R^1 and R^2 was derive from F, our main aim is to see that F can be derived from both relations without losing functional dependencies or gaining any new ones that were not in the original set.

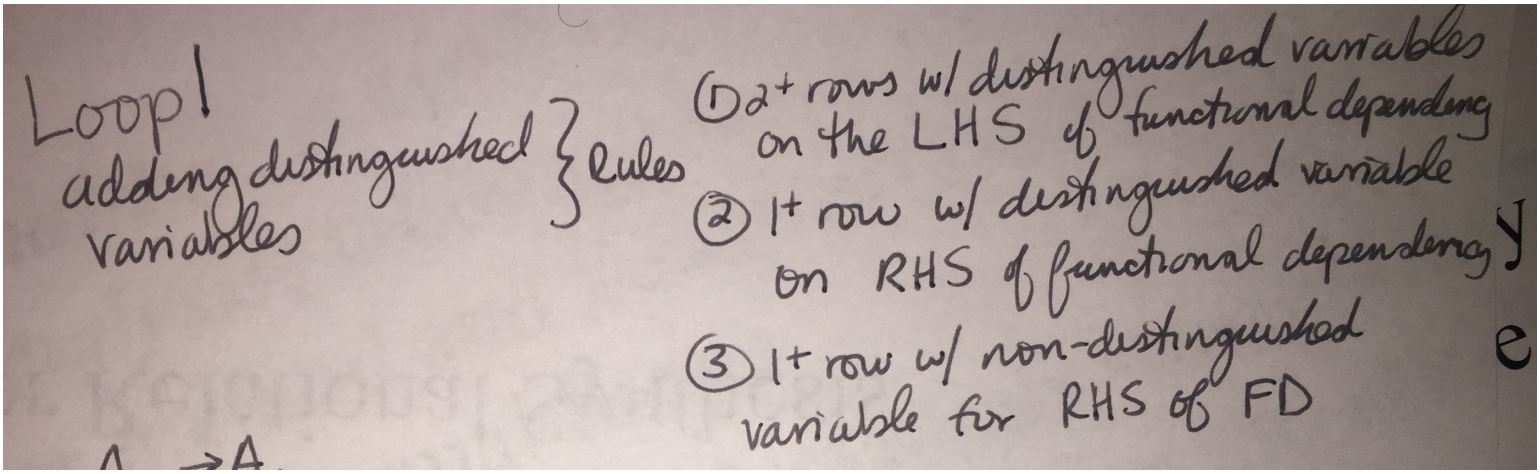
3. Perform the test for the nonadditive join property for the universal relation $R(A_1, A_2, A_3, A_4)$, and the decomposition *D* and set of functional dependencies *F* given below:

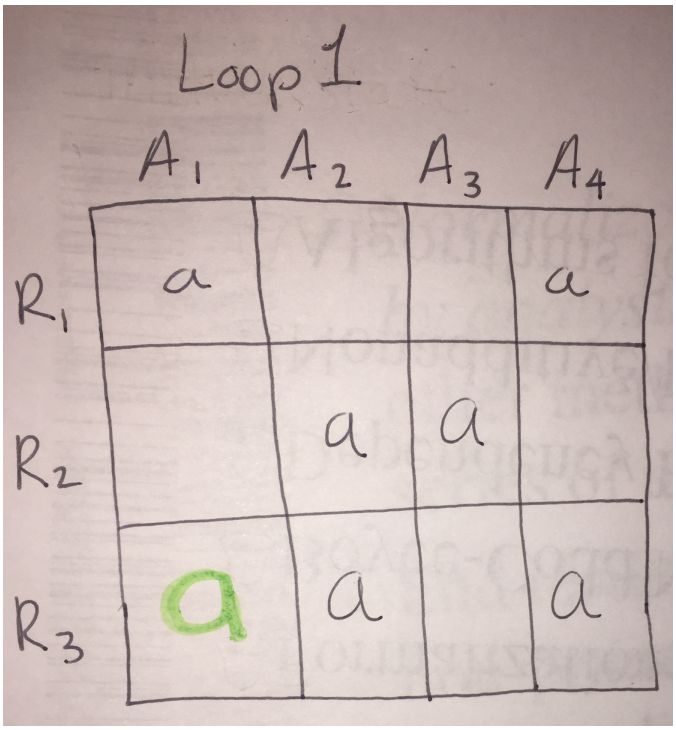
$D = \{ R_1(A_1, A_4) , R_2(A_2, A_3) , R_3(A_2, A_4) \}$

$F=\{ A_4 \rightarrow A_1 , A_2 \rightarrow A_4 , A_1 \rightarrow A_2 \}$

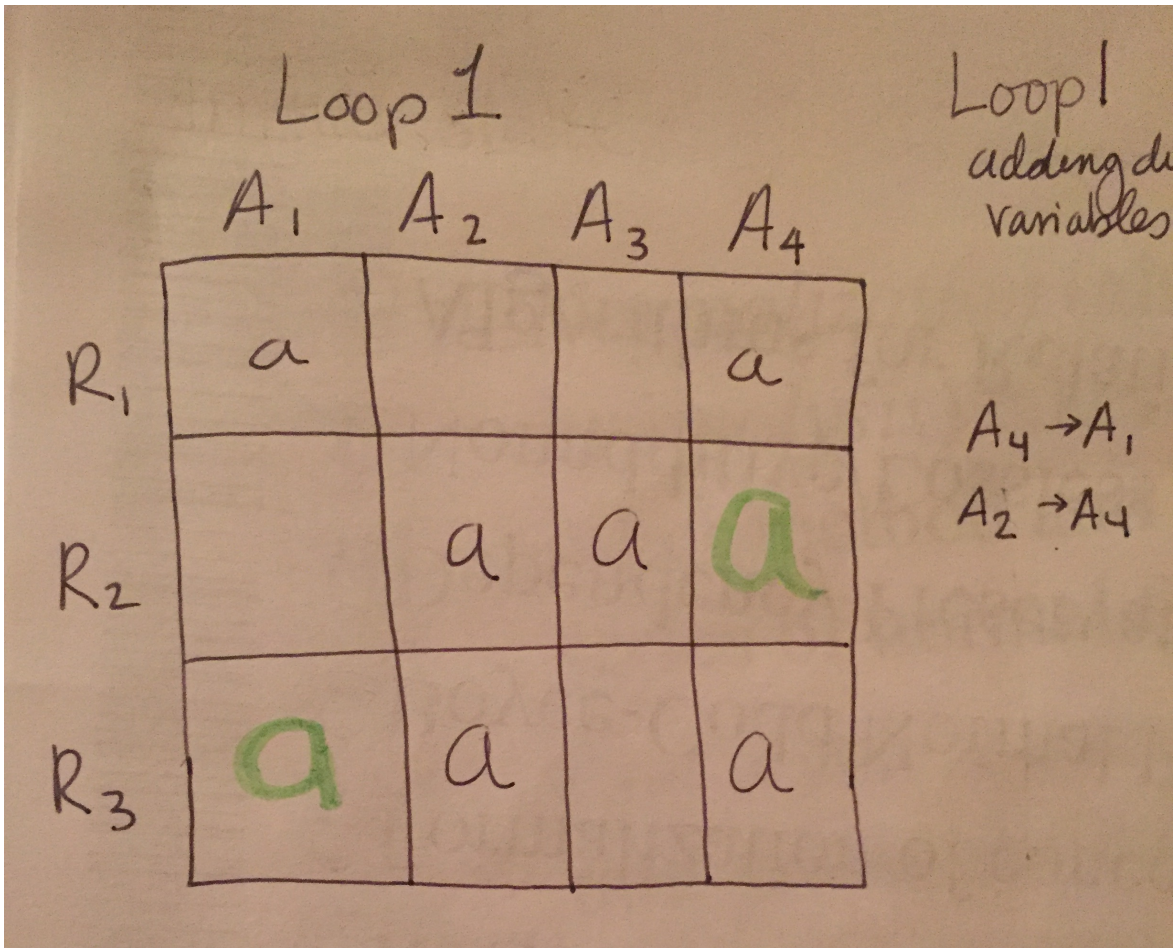
a. Show the initial state of the matrix *S*, and also its state at the end of each iteration of the loop (i.e., after each pass through the set of functional dependencies).







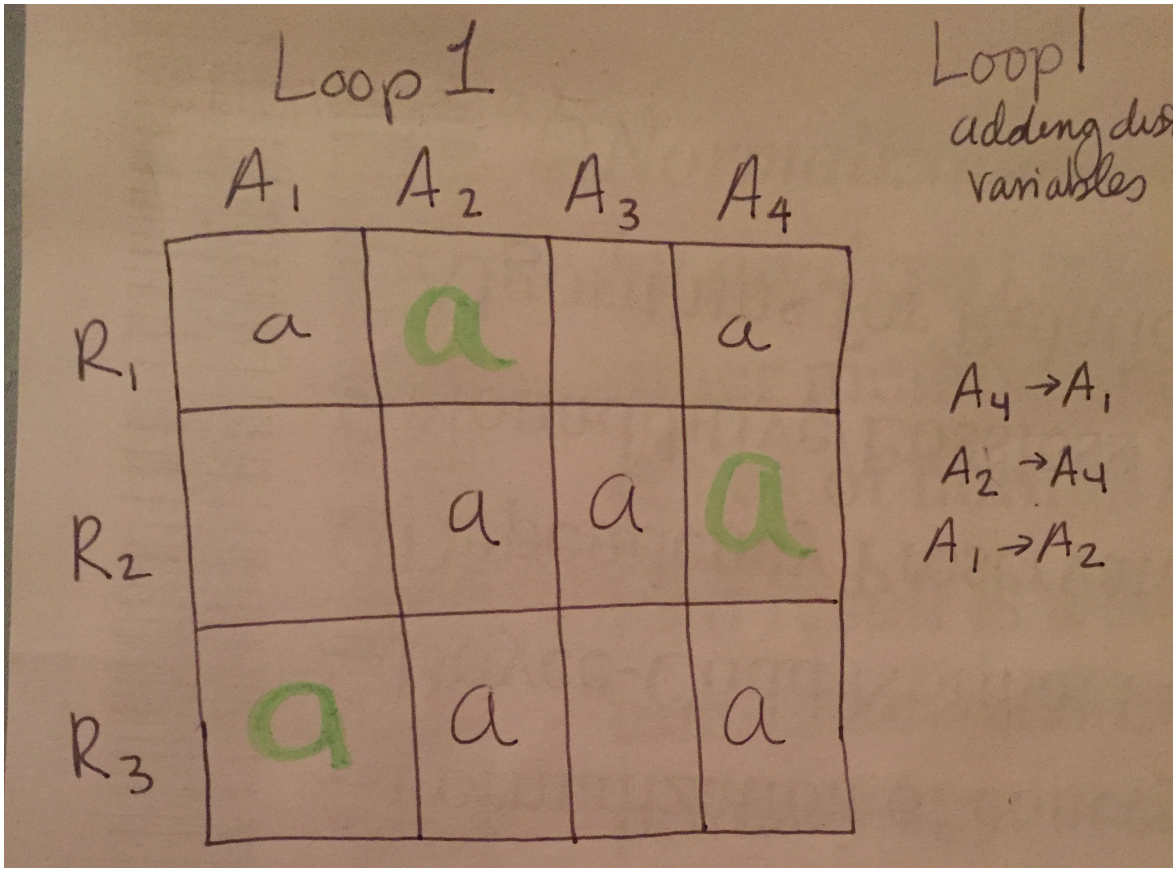
Loop 1



Loop 1
adding du
variables

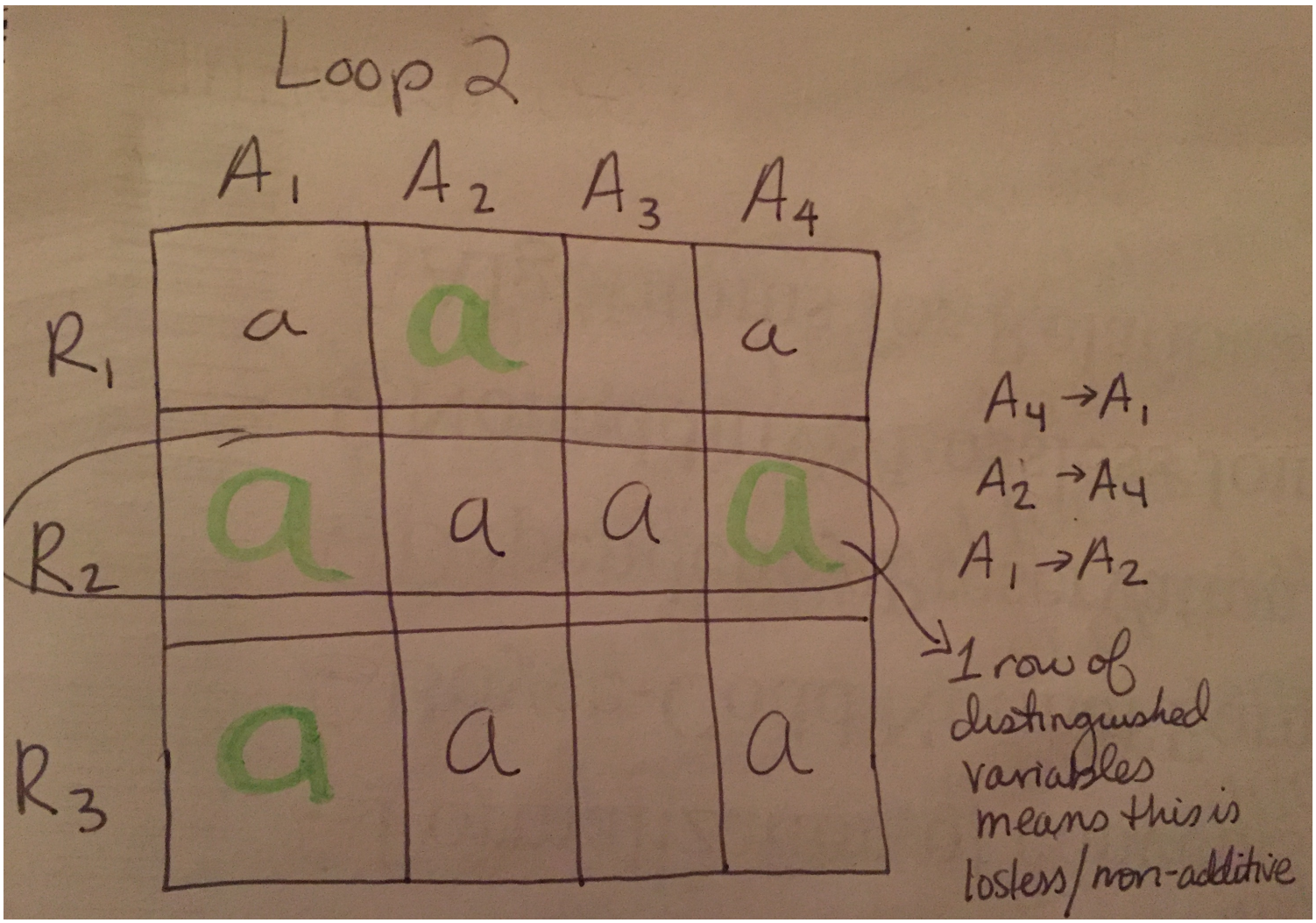
$A_4 \rightarrow A_1$
$A_2 \rightarrow A_4$

Loop 1



Loop 1
adding dus
variables

$A_4 \rightarrow A_1$
$A_2 \rightarrow A_4$
$A_1 \rightarrow A_2$



b. Does the decomposition *D* have the nonadditive join property? Explain why or why not in terms of the final state of the matrix *S*.

It does. after going through the matrix according to the rules for adding distinguished variables to this matrix (written above), I was able to continually adding distinguished variables until I had an entire row with all distinguish variables. If I had kept going through the loop without getting this result, then the decomposition wouldn't have had the nonadditive property.

4. Consider the universal relation
EMPLOYEE(ID, First, Last, Team, Dept, Salary) with the following set F of functional dependencies:

ID → First
ID → Last
First, Last→ID

Last→Team

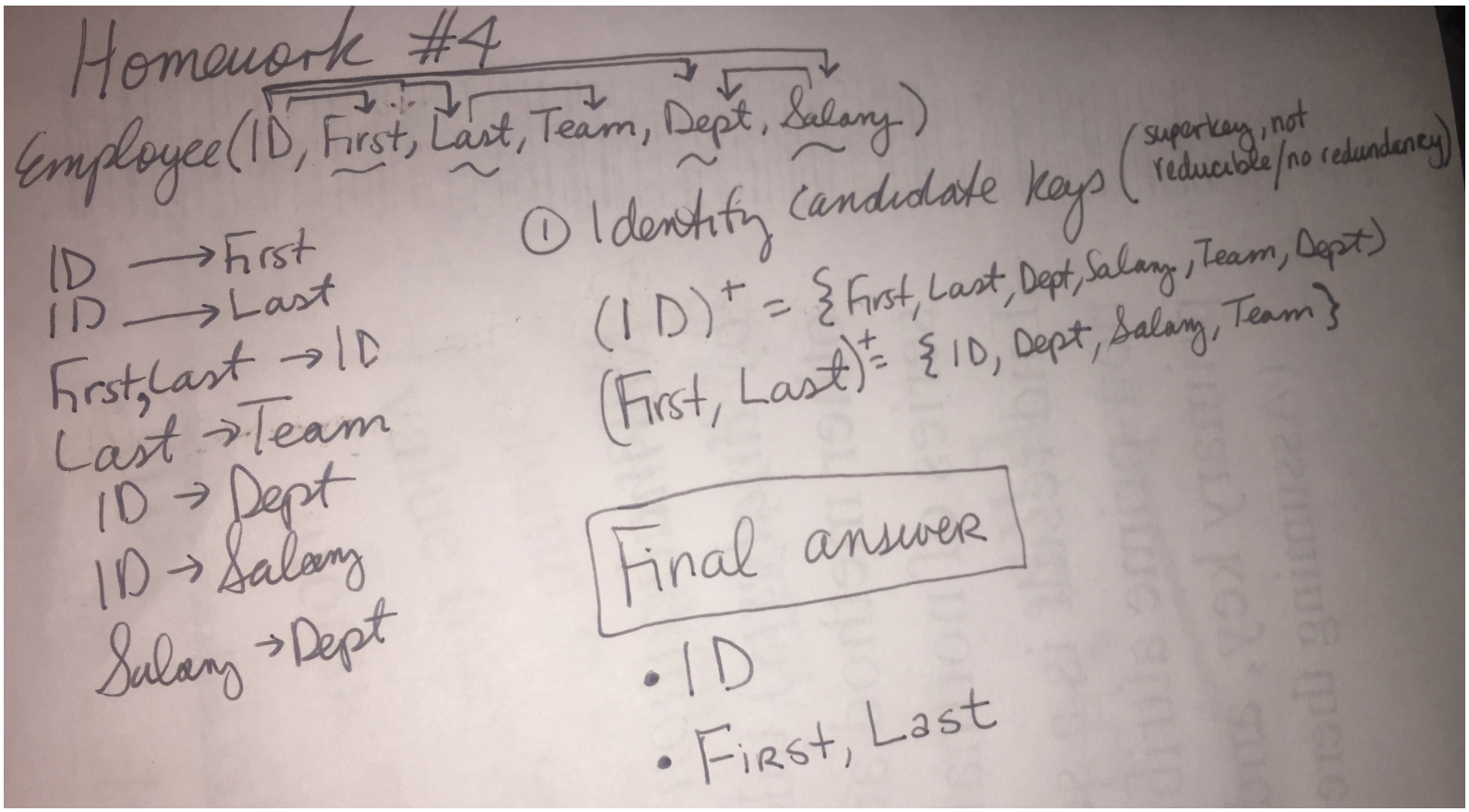
ID → Dept
ID → Salary

Salary → Dept

a. Identify all candidate keys of EMPLOYEE.

ID

First, Last



b. Construct a decomposition of EMPLOYEE into relations in 3NF that preserves dependencies. Be sure that you show that dependencies are preserved by either 1) showing it directly or 2) using an algorithm that is known to guarantee the dependency preservation property. (Hint: If you remove ID→Dept from F, the resulting set is a minimal cover of F.)

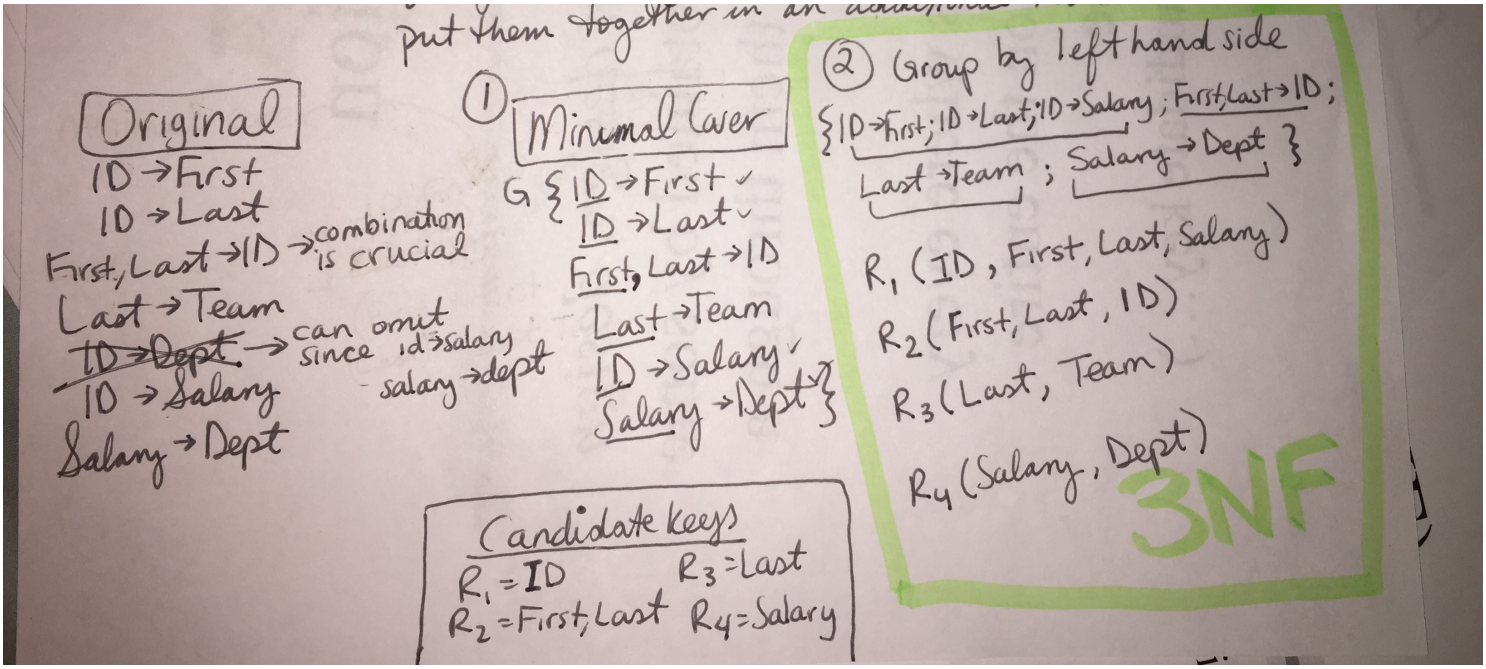
3NF =

R1 (ID, First, Last, Salary)

R2 (First, Last, ID)

R3 (Last, Team)

R4 (Salary, Dept)



The dependencies are preserved when looking at the projection of each relation (below)

projection onto R_1

ID → First
ID → Last
ID → Salary
First, Last → ID

projection onto R_2

First, Last → ID
ID → First
ID → Last

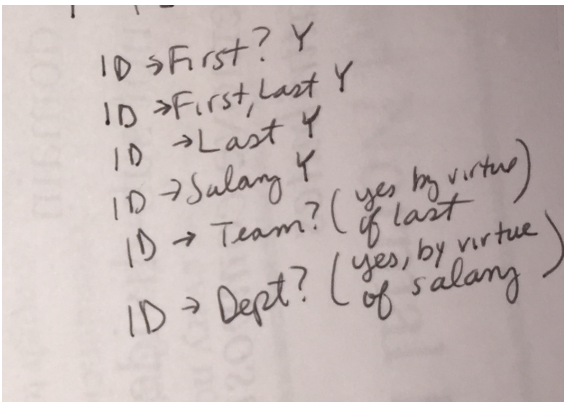
projection onto R_3

Last → Team

projection onto R_4

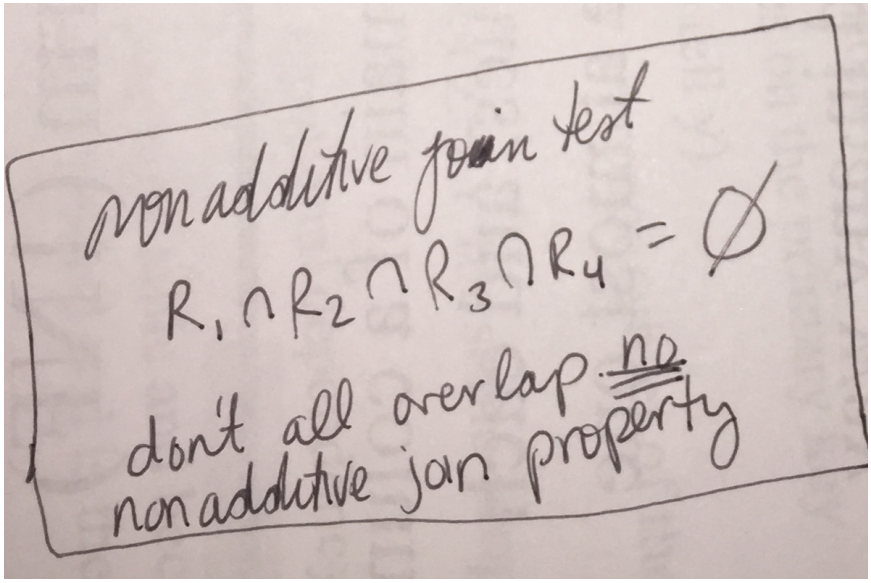
Salary → Dept

Going back through relations and checking dependencies



Testing for nonadditive/lossless property:

The union of all relations brings back an empty set. The nonadditive/lossless property is not upheld.



c. Are all of the relations in your decomposition in BCNF? Either explain why they are, or identify one that is not and explain why it is not. (Note that for a relation to be in BCNF, the determinants of all functional dependencies in the relation must be superkeys *of that relation* – not superkeys of the original universal relation.)

YES. Each determinant of the functional dependencies is a super key of a relation (not the universal relation).