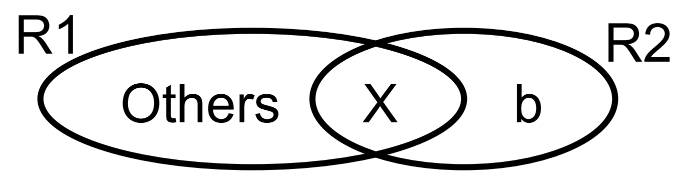
CPSC 304 – Administrative notes October 2 & October 3, 2024

- New “in class” exercise started today
- October 15: Milestone 2 due
- October 22: Midterm @ 6PM
  - Do you have a conflict? If so fill out the midterm conflict form on Piazza by the end of Monday
  - More information on Piazza ~1 week before the midterm.

Now where were we...

- We'd been discussing normalization
- We'd discussed BCNF (for all FDs of the form $X \rightarrow b$, X must be a superkey for the relation)
- We'd discussed how to decompose relations into BCNF. Repeatedly do:



Let's revisit a discussion that we had about which FDs hold after a decomposition…

Revisiting a previous clicker question:

- Take the closure of the attributes using *all* FDs
- For an FD X→b, if the decomposed relation S contains {X U b}, then the FD holds for S:
- For example. Consider relation R(A,B,C,D,E) with functional dependencies $AB \rightarrow C$, $BC \rightarrow D$, $CD \rightarrow E$, $DE \rightarrow A$, and $AE \rightarrow B$.

  Project these FD's onto the relation S(A,B,C,D).
- Which of the following hold in S?

A. A→B — A+=A

B. AB→E — AB+=ABCDE, but E is not in S

C. AE→B — AE+=ABCDE, but E is not in S

D. BCD→A — Yes. BCD+=ABCDE; all in S

E. None of the above

Why do we need to consider FDs that aren't present in decomposed relations? (1/3)

Consider a relation R(A, B, C, D) with FDs A→B, and B→C.

Consider the following set of tuples that may or may not adhere to the FDs above:

A	B	C	D
1	3	1	1
1	3	2	2
2	4	3	3
2	4	4	4

Which of the following FDs hold in the above tuples?

A. A→B only
B. B→C only
C. Both A→B and B→C
D. Neither A→B nor B→C

Does A→C hold? No!

Why do we need to consider FDs that aren't present in decomposed relations? (2/3)

Consider a relation R(A, B, C, D) with FDs A→B, and B→C.

Consider a new set of tuples where A→B and B→C:

A	B	C	D
1	3	5	1
1	3	5	2
2	4	6	3
2	4	6	4

Does A→C hold for this instance of R?

A. Yes

B. No

Is it possible, given that A→B and B→C, hold to create a set of tuples where A→C does *NOT* hold?

A. Yes

B. No

Why do we need to consider FDs that aren't present in decomposed relations? (3/3)

Consider a relation R(A, B, C, D) with FDs A→B, and B→C.
Consider the same set of tuples where the FDs hold, *including* A→C:

Let's assume that we decompose on A→B, resulting in two relations: R_1(A,B), R_2(A, C, D)

A	B	C	D
1	3	5	1
1	3	5	2
2	4	6	3
2	4	6	4

R_1

A	B
1	3
2	4

R_2

A	C	D
1	5	1
1	5	2
2	6	3
2	6	4

Does A→C hold for this instance of R_2?

A. Yes
B. No

Is it possible to create a set of tuples for R_2 where A→C does *NOT* hold?

A. Yes
B. No

Yet Another BCNF Example: Is it in BCNF? If so, why? If not, decompose.

R(A,B,C,D,E,F)

FD = A→B

DE→ F,

B→C

A+ = {A,B,C}

B+ = {B,C}

DE+ = {D,E,F}

ADE is the only key

Decompose on A→B

CDEF A B

R1(A, B), R2(A, C, D, E, F)

Decompose R2 on DE→F

AC DE F

R3(D,E, F), R4(A, C, D, E)

What is A+ on R4?

A. A
B. AC
C. ABC
D. Other

Decompose again:

DE A C

R5(A,C), R6(A, D, E)

Final answer: R1(A, B), R3(D, E, F), R5(A, C), R6(A, D, E)

Clicker exercise: More BCNF

- Let R(ABCD) be a relation with functional dependencies $A \rightarrow B$, $C \rightarrow D$, $AD \rightarrow C$, $BC \rightarrow A$
- Decompose into BCNF.

Clicker exercise: More BCNF

- Let R(ABCD) be a relation with functional dependencies $A \rightarrow B$, $C \rightarrow D$, $AD \rightarrow C$, $BC \rightarrow A$
- Decompose into BCNF. Which of the following is a lossless-join decomposition of R into BCNF?

A. {AB, AC, BD}
B. {AB, AC, CD}
C. {AB, AC, BCD}
D. All are
E. None are

Clicker exercise: More BCNF Explained

- Let R(ABCD) be a relation with functional dependencies $A \rightarrow B$, $C \rightarrow D$, $AD \rightarrow C$, $BC \rightarrow A$
- Decompose into BCNF. Which of the following is a lossless-join decomposition of R into BCNF?

Closures:

- A+ = {AB}
- C+ = {CD}
- AD+ = {ADBC} ← Key
- BC+ = {BCDA} ← Key

A→B violates BCNF in R: Decompose to R1(<u>A</u>, B), R2(<u>A</u>, <u>C</u>, D)
C→D violates BCNF in R2: Decompose to R3(<u>C</u>,D), R4(<u>C, A</u>)
All are two attribute relations, all are in BCNF. Final answer:
R1(<u>A</u>,B), R3(<u>C</u>,D), R4(<u>C, A</u>)

Clicker Exercise: Option 'A' exposed

- Let R(ABCD) be a relation with functional dependencies $A \rightarrow B$, $C \rightarrow D$, $AD \rightarrow C$, $BC \rightarrow A$

Is {AB, AC, BD} a lossless join?

Imagine tuples:

A	B	C	D
1	2	5	6
1	2	3	7
8	2	9	4

decompose

A	B
1	**2**
8	2

A	C
1	5
1	**3**
8	9

B	D
2	6
2	7
2	**4**

join

A	B	C	D
1	2	5	6
1	2	3	7
8	2	9	4
1	**2**	**3**	**4**

In class exercise

- Do Normalization #2 now

This BCNF stuff is great and easy!

- Guaranteed that there will be no redundancy of data
- Easy to understand (just look for superkeys)
- Easy to do.
- So what is the main problem with BCNF?
  - For one thing, BCNF may not preserve all dependencies

An illustrative BCNF example

Unit	Company	Product

$\boxed{\begin{array}{l}\text{Unit} \rightarrow \text{Company} \\ \text{Company, Product} \rightarrow \text{Unit}\end{array}}$

Is Unit a key?

Unit	Company

$\boxed{\text{Unit} \rightarrow \text{Company}}$

Unit	Product

We lose the FD: Company, Product → Unit !!

So What's the Problem (1/2)?

Assume that we start with a table with a single tuple:

Unit	Company	Product
SKYWill	UBC	Databases

We decompose on Unit→Company:

<u>Unit</u>	Company
SKYWill	UBC

Unit	Product
SKYWill	Databases

Now someone adds a new team, Team Meat to each table:

<u>Unit</u>	Company
SKYWill	UBC
Team Meat	UBC

Unit	Product
SKYWill	Databases
Team Meat	Databases

So What's the Problem (2/2)?

Unit	Company
SKYWill	UBC
Team Meat	UBC

Unit	Product
SKYWill	Databases
Team Meat	Databases

Unit $\rightarrow$ Company

No problem so far. All *local* FD's are satisfied.
Let's put all the data back into a single table again:

Unit	Company	Product
SKYWill	UBC	Databases
Team Meat	UBC	Databases

Violates the FD: Company, Product $\rightarrow$ Unit

3NF to the rescue!

A relation R is in 3NF if:

If X → b is a non-trivial dependency in R,
then X is a superkey for R

BCNF

or b is part of a (minimal) key.

(must be true for every such functional dependency)

Note: b must be part of a key *not* part of a superkey (if a key exists, *all* attributes are part of a superkey)

Example: R(Unit,Company, Product)

- Unit → Company
- Company, Product → Unit
  Keys: {Company, Product}, {Unit,Product}

BCNF or 3NF? The clicker question

- 3NF retains all functional dependencies
- Rule: for all non-trivial functional dependencies in a relation R of the form $X \rightarrow b$, it must be the case that X is a superkey of R or b is part of a key
  - Ex: is R(ABCD) in BCNF, 3NF, or both given the following FDs:
    - $AB \rightarrow C$
    - $CD \rightarrow A$

A. Both BCNF and 3NF
B. BCNF but not 3NF
C. 3NF but not BCNF
D. Neither BCNF nor 3NF