CPSC 304 – Administrative notes October 4 & October 8, 2024

- New "in class" exercise started today
- Project:
  - October 15: Milestone 2 due
  - Project repositories are ready (see piazza)
  - Oracle accounts are ready (see Piazza)
  - Milestone 1 will be returned on October 8
    - There will be a comment from me/your project TA. Any comment from a TA is from your project TA
  - Tutorial materials have been released to help you
    - Oracle/SQL Plus
    - PHP
    - JavaScript
- October 22: Midterm @ 6PM
  - Survey on exam conflicts has closed. We hope to have information to those who registered conflicts in the next 24 hours.

Now where were we…

- We were discussing Functional Dependencies & Normal Forms
- BCNF: for all non-trivial functional dependencies in a relation R of the form X→b, it must be the case that X is a superkey of R
- 3NF retains all functional dependencies
- 3NF Rule: for all non-trivial functional dependencies in a relation R of the form X→b, it must be the case that X is a superkey of R or b is part of a key

Question: BCNF, 3NF, both, or neither?

- Remember, 3NF definition:

  If X → b is a non-trivial dependency in R,
  then X is a superkey for R
  or b is part of a (minimal) key. ← **3NF only!**

- Is Relation R(A, B, C, D, E) with FDs
  A→B
  BC→E
  ED→A
  In BCNF, 3NF, both, or neither

A. BCNF only
B. 3NF only
C. Both
D. Neither

A+=AB
BC+=BCE
ED+=EDAB
Keys = ACD, BCD, CDE
A→B and others violate BCNF but not 3NF since RHS of all are part of a key.

So how do we decompose into 3NF?

- To do that, we need to have a way to make the set of functional dependencies as small as possible
- Otherwise, we may accidentally duplicate information (which defeats the whole point of this exercise)

To decompose to 3NF we rely on the Minimal Cover for a Set of FDs

Goal: Transform FDs to be as small as possible

- *Minimal cover* G for a set of FDs F:
  - Closure of F = closure of G (i.e., imply the same FDs)
  - Right hand side of each FD in G is a single attribute
  - If we delete an FD in G or delete attributes from an FD in G, the closure changes
- Intuitively, every FD in G is needed, and is "*as small as possible*" in order to get the same closure as F
- e.g., $A \rightarrow B$, $ABCD \rightarrow E$, $EF \rightarrow GH$, $ACDF \rightarrow EG$ has the following minimal cover:
  - $A \rightarrow B$, $ACD \rightarrow E$, $EF \rightarrow G$ and $EF \rightarrow H$

Finding minimal covers of FDs

1. Put FDs in standard form (have only one attribute on RHS)
2. Minimize LHS of each FD
3. Delete Redundant FDs

Example:

A→B, ABCD→E, EF→G, EF→H, ACDF → EG

- Replace last rule with
  - ACDF → E
  - ACDF → G

Finding minimal covers of FDs

1. Put FDs in standard form (have only one attribute on RHS)
2. Minimize LHS of each FD
3. Delete Redundant FDs

Example:

$A \rightarrow B$, $ABCD \rightarrow E$, $EF \rightarrow G$, $EF \rightarrow H$, $ACDF \rightarrow E$, $ACDF \rightarrow G$

- Can we take anything away from the LHS?
  - $ACD^+ = ABCDE$, (crucially includes B) so remove B from the FD

Finding minimal covers of FDs

1. Put FDs in standard form (have only one attribute on RHS)
2. Minimize LHS of each FD
3. Delete Redundant FDs

Example:

A→B, ACD→E, EF→G, EF→H, ACDF → E, ACDF → G

- Let's find $ACDF^+$ **without** considering the highlighted FDs
  - $ACDF^+$= ACDFEBGH, so I can remove the highlighted rules
- Final answer: A→B, ACD→E, EF→G, EF→H

Minimal cover example. Your turn (with some guidance)

- Consider the relation R(CSJDPQV) with FDs C→SJDPQV, JP→C, SD→P, J→S

Find a minimal cover. What are the steps?

1. Put FDs in standard form (have only one attribute on RHS)
2. Minimize LHS of each FD
3. Delete Redundant FDs

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs
  $C \rightarrow SJDPQV$, $JP \rightarrow C$, $SD \rightarrow P$, $J \rightarrow S$

Find a minimal cover

Step 1: RHS has only one attribute

$C \rightarrow S$, $C \rightarrow J$, $C \rightarrow D$, $C \rightarrow P$, $C \rightarrow Q$, $C \rightarrow V$, $JP \rightarrow C$, $SD \rightarrow P$, $J \rightarrow S$

Step 2: Can we minimize LHS of any FDs?

Can we minimize LHS of $JP \rightarrow C$

A. Yes

B. No

Goal: look at LHS of FDs. See if can generate any subset of it

$J^+ = JS$

$P^+ = P$

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs
  C→SJDPQV, JP→C, SD→P, J→S

Find a minimal cover

Step 1: RHS has only one attribute

C→S, C→J, C→D, C→P, C→Q, C→V, JP→C, SD→P, J→S

Step 2: Can we minimize LHS of any FDs?

Can we minimize LHS of SD→P?

A. Yes
B. No

$S^+ = S$ and $D^+ = D$

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs
  C→SJDPQV, JP→C, SD→P, J→S

Find a minimal cover

Step 1: RHS has only one attribute

C→S, C→J, C→D, C→P, C→Q, C→V, JP→C, SD→P, J→S

Step 2: Can we minimize LHS of any FDs?

There are no other LHSs with more than 1 attribute. So, no.

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs
  C→SJDPQV, JP→C, SD→P, J→S

Step 1: RHS has only one attribute

C→S, C→J, C→D, C→P, C→Q, C→V, JP→C, SD→P, J→S

Step 2: Can we minimize LHS of any FDs? No

Step 3: Can we remove any FDs?

- Can we remove C→S?

A. Yes

B. No

Find C^+ without considering this rule (C → S)

C^+ = CSJDPQV, so we can delete this FD
(i.e., C^+ does NOT change)

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs
  C→SJDPQV, JP→C, SD→P, J→S

Step 1: RHS has only one attribute

C→S, C→J, C→D, C→P, C→Q, C→V, JP→C, SD→P, J→S

Step 2: Can we minimize LHS of any FDs? No

Step 3: Can we remove any FDs?

- Can we remove C→J?

A. Yes

B. No

Find C^+ without considering this rule (C → J)

C^+ = CDPQV, which does not include J so we **CANNOT** delete this FD (i.e., C^+ changes)

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs C→SJDPQV, JP→C, SD→P, J→S

Step 1: Find a minimal cover

C→S, C→J, C→D, C→P, C→Q, C→V, JP→C, SD→P, J→S

Step 2: Can we minimize LHS of any FDs? No

Step 3: Can we remove any FDs?

- Can we remove C→P?

A. Yes

B. No

Find C^+ without considering this rule

C^+ = CSJDQVP, so we can delete this FD

Another minimal cover example

- Consider the relation R(CSJDPQV) with FDs C→SJDPQV, JP→C, SD→P, J→S

Step 1: RHS has only one attribute

C→S, C→J, C→D, C→P, C→Q, C→V, JP→C, SD→P, J→S

Step 2: Can we minimize LHS of any FDs? No

Step 3: Can we remove any FDs?

Are there others we can remove other than C→S and C→P?

A. Yes

B. No

No: Final answer: C→J, C→D, C→Q, C→V, JP→C, SD→P, J→S

Make sure to minimize LHS before deleting redundant FDs

- If step 3 is done prior to step 2, the final set of FDs could still contain redundant FDs. The wrong way:
  - Consider ABCD→E, E→D, A→B, AC→D
  - Already has one attribute on RHS
  - Let's delete redundant FDs.
    - None of the FDs are redundant.
- The right way:
  - ABCD→E, E→D, A→B, AC→D
  - Now let's shorten FDs
    - ABCD→E can be replaced by AC→E
  - However the current set of FDs are not minimal
    - AC→E, E→D, A→B, AC→D
    - The highlighted FD can be deleted

1. One attribute on RHS
2. Delete Redundant FDs
3. Minimize LHS of each FD

Does not work

Now we're ready to decompose into 3NF

- We'll cover two methods
- Both methods
  - Result in relations that do not violate 3NF
  - Are lossless (you don't get any additional tuples)
  - Preserve all functional dependencies
- The first one starts by ensuring that the decomposition is lossless and then preserves all functional dependencies
- The second one starts by preserving all functional dependencies and then ensures that the decomposition is lossless

There are other methods that do not ensure both of these properties. Do not use them!

Decomposition into 3NF method #1: lossless join method

Decomposition into 3NF using the lossless join method:

- Given the FDs F, compute F': the *minimal cover for F*
- Decompose using F' *if violating 3NF* similar to how it was done for BCNF – decompose *all the way down to BCNF*
- After each decomposition identify the set of dependencies N in F' that are not preserved by the decomposition.
- At the end, for each $X \rightarrow b$ in N create a relation $R_n(X \cup b)$ and add it to the decomposition

Intuition: first remove redundancy using lossless joins to ensure all results are valid. Then ensure that we maintain all functional dependencies.

3NF example (already a minimal cover)

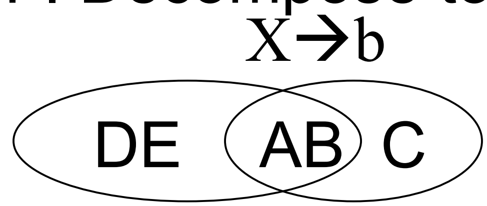
- Example: R(ABCDE) FD: AB→C, C→D

$AB \rightarrow A$
$AB \rightarrow B$
$AB \rightarrow C$
$AB \rightarrow D$

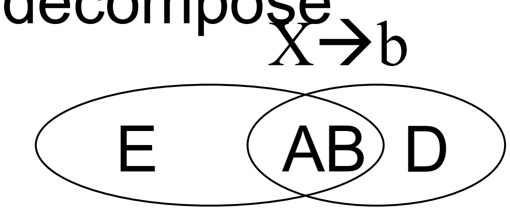
$ABE^+ = ABCDE$ only key

$AB^+ = ABCD$
$C^+ = CD$

- AB→C violates 3NF, so not in 3NF. Decompose to BCNF
  - R_1(ABC), R_2(ABDE)



- AB→D violates BCNF in R_2, so decompose
  - R_3(ABD), R_4(ABE)



- R_1(ABC), R3(ABD), R_4(ABE) are now in BCNF
- Are all FDs preserved? We lost C→D so add R_5(CD)

Another 3NF example: your turn

- Let R(CSJDPQV) be a relation with the following FDs
  - SD→P
  - JP→C
  - J→S
- Is this in 3NF? If not, decompose R into 3NF

$$SD^{+} = \{SDP\}$$

$$JP^{+} = \{JPCS\}$$

$$J^{+} = \{JS\}$$

Another 3NF example: your turn

- Let R(CSJDPQV) be a relation with the following FDs
  - SD→P
  - JP→C
  - J→S
- Is this in 3NF? If not, decompose R into 3NF
  - $JDQV^+$ = CSJDPQV Key

Left	Middle	Right
JDQV	PS	C

Another 3NF example: your turn

- Let R(CSJDPQV) be a relation with the following FDs
  - SD→P
  - JP→C
  - J→S
- Is this in 3NF? If not, decompose R into 3NF

1. One attribute on RHS
2. Minimize LHS of each FD
3. Delete Redundant FDs

Step 1: Done

Step 2

$S^+ = \{S\}$
$D^+ = \{D\}$

$J^+ = \{JS\}$
$P^+ = \{P\}$

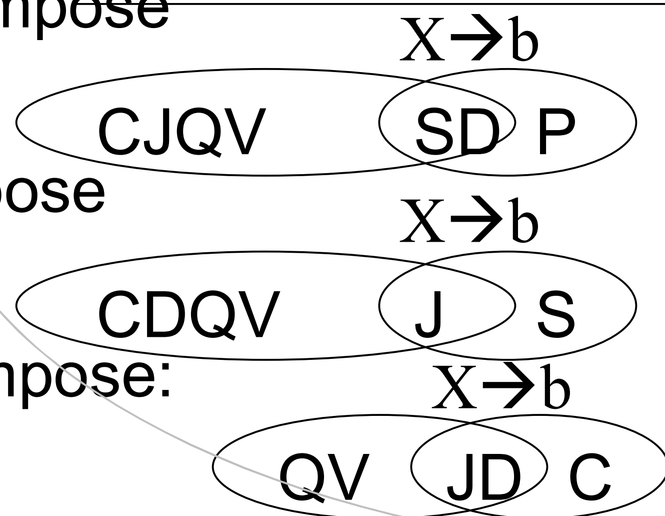
Step 3: No redundant FDs

Already in minimal cover

Another 3NF example

- Let R(CSJDPQV) be a relation with the following FDs
  - SD→P
  - JP→C
  - J→S
  - SD→P violates 3NF in R, so decompose
  - R_1(SDP) , R_2(CSJDQV)
- J→S violates BCNF in R_2, so decompose
  - R_3(JS), R_4(CJDQV)
- JD→C violates BCNF in R_4, so decompose:
  - R_5(JDC), R_6(JDQV)
- No more violations! Are all FDs preserved? No. Add: R_7(JPC)
- Final answer: R_1(SDP), R_3(JS), R_5(JDC), R_6(JDQV), R_7(JPC)

JP^+= JPSC
SD^+=SDP
J^+=JS
JD+=JDSPC* (next slide)
$JDQV^+$ = CSJDPQV Key



On the previous slide, we took the closure of JD

- The given FDs were:
  - SD→P
  - JP→C
  - J→S
- How did we know to do that?
- Most of it is practice
- Looking at the explanation of what has to be in a key can help:

Left	Middle	Right
JDQV	SP	C

- Q&V don't appear in any FDs, so no interesting closures to take there
- Only thing left that's interesting to try: JD

Decomposition into 3NF Using a Minimal Cover and 3NF Synthesis

- This is an alternative approach to finding a lossless-join, dependency-preserving decomposition into 3NF.
- Consider relation R and its FDs F. To use synthesis to decompose to 3NF:
  1. Find a minimal cover F'.
  2. For each FD X → b in F'
     Add relation Xb to the decomposition for R.
  3. If there are no relations in the decomposition that contain all of the attributes of a key, add in a relation that contains all the attributes of a key. This preserves lossless joins.

Intuition: first maintain all functional dependencies, then ensure that joins are lossless by adding a key.

Example: Decomposition into 3NF Using a Minimal Cover and 3NF Synthesis

- Suppose we have R(A,B,C) with FDs: A → B, C → B. This is not in 3NF. The only key is AC.
  1. Find a minimal cover F'. Already done.
  2. For each FD X→b, add relation Xb to the decomposition for R.

     Result: R1(A,B) and R2(B,C). Are we done? No.
  3. Does it contain a key? What are the keys of R? AC. Add R3(A,C).
- Let's see why we need step #3

Consider the following set of tuples with the previous relation and FDs

$A \rightarrow B$
$C \rightarrow B$

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

Decompose

Maintains $A \rightarrow B$

A	B
1	2
1	2
2	2

B	C
2	3
2	4
2	4

Maintains $C \rightarrow B$

Natural Join

Maintains $C \rightarrow B$ and $A \rightarrow B$

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

Not lossless!

Take two!

A→B
C→B

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

Decompose

Maintains A→B

A	B
1	2
1	2
2	2

Maintains C→B

B	C
2	3
2	4
2	4

Key!

A	C
1	3
1	4
2	4

Natural Join

Maintains C→B and A→B

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

Lossless!
Can't get (2, 2, 3) because there is no (2, 3) tuple in R3(A,C)

Exercise:
Step 0: find the closures and keys

Consider relation R(A,B,C,D,E,F,G) and FDs

- AB→C   AB+=ABCD
- C→D   C+=CD   Keys are ABEG, ABFG
- EG→F   EG+=EGF
- F→E   F+=EF

Left	Middle	Right
ABG	CEF	D

Is this in 3NF? If not, decompose using synthesis

Is this in 3NF?   A. Yes   B. No

No! AB→C violates 3NF To decompose using synthesis, find the minimal cover

Step 1: Find a minimal cover

Consider relation R(A,B,C,D,E,F,G) and FDs

- AB→C	AB+=ABCD
- C→D	C+=CD	Keys are ABEG, ABFG
- EG→F	EG+=EGF
- F→E	F+=EF

Is this set of functional dependencies a minimal cover?

A. Yes	B. No

Yes! There's nothing we can remove

Step 2: For each FD X→b, add relation Xb to the decomposition for R

Consider relation R(A,B,C,D,E,F,G) and FDs

- AB→C AB+=ABCD
- C→D C+=CD Keys are ABEG, ABFG
- EG→F EG+=EGF
- F→E F+=EF

This set of FDs is a minimal cover

Initial decomposition: R1(ABC), R2(CD), R3(EFG), R4(EF)

Step 3: Preserve lossless joins – make sure you have a key

Consider relation R(A,B,C,D,E,F,G) and FDs

- $AB \rightarrow C$ $AB^+ = ABCD$
- $C \rightarrow D$ $C^+ = CD$ Keys are ABEG, ABFG
- $EG \rightarrow F$ $EG^+ = EGF$
- $F \rightarrow E$ $F^+ = EF$

This set of FDs is a minimal cover

Initial decomposition: R1(ABC), R2(CD), R3(EFG), R4(EF)

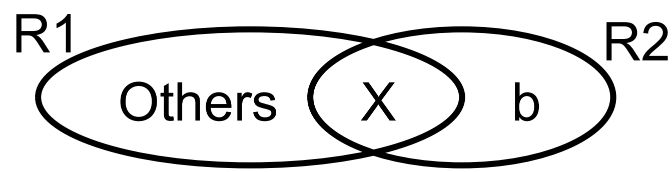
Initial decomposition does not include a key of R, add one in to give us: R1(ABC), R2(CD), R3(EFG), R4(EF), R5(ABEG)

Step 3: Preserve lossless joins – make sure you have a key

Consider relation R(A,B,C,D,E,F,G) and FDs

- $AB \rightarrow C$ $\quad AB^+=ABCD$
- $C \rightarrow D$ $\quad C^+=CD$
- $EG \rightarrow F$ $\quad EG^+=EGF$
- $F \rightarrow E$ $\quad F^+=EF$

Keys are ABEG, ABFG



This set of FDs is a minimal cover

Initial decomposition: R1(ABC), R2(CD), R3(EFG), R4(EF)

Initial decomposition does not include a key of R, add one in to give us: R1(ABC), R2(CD), **R3(EFG)**, **R4(EF)**, R5(ABEG)

Wait. Hang on a minute. That's a little weird.

In decompositions, you can often make some adjustments to make a “better” decomposition

- In this case, R1(ABC), R2(CD), **R3(EFG)**, **R4(EF)**, R5(ABEG) have redundant relations – you don’t need R4 because all information is contained in R3
- You can make the same optimizations in decompositions for BCNF
- Other optimizations exist
- For the purposes of this course, we ask you to remove redundancy when decomposing to either BCNF or 3NF
- We define redundancy as: If all of the attributes of a relation R are a subset of the attributes of relation S, then R is redundant.
    - E.g., in this example EF is a subset of EFG, so R4 is redundant and should be removed.

Clicker Question: BCNF and 3NF

- Consider the following relation and functional dependencies:
  R(ABCD) FD's: ACD → B ; AC → D ; D → C ; AC → B
  Which of the following is true:

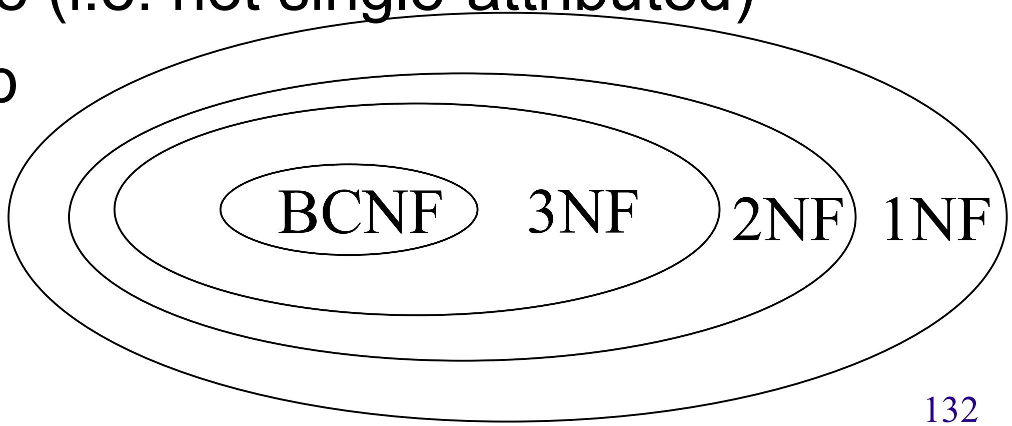
A. R is in neither BCNF nor 3NF
B. R is in BCNF but not 3NF
C. R is in 3NF but not in BCNF
D. R is in both BCNF and 3NF

ACD^+=ABCD
AC^+=ACDB
D^+=DC
AD^+=ADCB
Keys: AC, AD

D→C (D not superkey so NOT BCNF)
C is part of a minimal key
So D→ C does not violate 3NF
The rest violate neither BCNF nor 3NF

Comparing BCNF & 3NF

- BCNF guarantees removal of all anomalies
- 3NF has some anomalies, but preserves all dependencies
- If a relation R is in BCNF it is in 3NF.
- A 3NF relation R may not be in BCNF if all 3 of the following conditions are true:
  a. R has multiple keys
  b. Keys are composite (i.e. not single-attributed)
  c. These keys overlap



Normalization and Design

- Most organizations go to 3NF or better
- If a relation has only 2 attributes, it is automatically in 3NF and BCNF
- Our goal is to use lossless-join for all decompositions and preserve dependencies
- BCNF decomposition is always lossless, but may not preserve dependencies
- Good heuristic:
  - Try to ensure that all relations are in at least 3NF
  - Check for dependency preservation

On the other hand… Denormalization

- Process of intentionally violating a normal form to gain performance improvements
- Performance improvements:
  - Fewer joins
  - Reduces number of foreign keys
    - Since FD’s are often indexed, the number of indexes many be reduced
- Useful if certain queries often require (joined) results, and the queries are frequent enough

Learning Goals Revisited

- Debate the pros and cons of redundancy in a database.
- Provide examples of update, insertion, and deletion anomalies.
- Given a set of tables and a set of functional dependencies over them, determine all the keys for the tables.
- Show that a table is/isn't in 3NF or BCNF.
- Prove/disprove that a given table decomposition is a lossless join decomposition. Justify why lossless join decompositions are preferred decompositions.
- Decompose a table into a set of tables that are in 3NF, or BCNF.

In class exercise

- Normalization #4