Quiz: Is EmToPrHoSkLoRo in BCNF?

- (Em → ToPr), (To → Pr), (EmTo → Ho),
  (SkLo → Ro), (Ro → Lo)

- Keys are \{Em, Sk, Lo\} and \{Em, Sk, Ro\}

- Do any of the rules violate the BCNF irredundancy condition?
Decomposition into BCNF

• Suppose $X \rightarrow Y$ violates BCNF, and $X \cap Y = \emptyset$

• Let $Z$ be the other attributes

• Then decompose $XYZ$ into $XY$ and $XZ$

• This gives a valid decomposition:
  $\pi_{XY}(R) \bowtie \pi_{XZ}(R) = R$

• If $XY$ or $YZ$ are not in BCNF, then recurse
BCNF Decomposition of PZC

- (Person → Zip), (Zip → City)
- Key = \{Person\}, so (Zip → City) violates BCNF
- The algorithm gives \{Zip, City\} and \{Zip, Person\}
- Zip is a key of \{Zip, City\}, and so (Zip → City) no longer violates BCNF
- We’re done, since each part is in BCNF
A Flaw with the BCNF Method

- \((\text{Addr, City} \rightarrow \text{Zip}), (\text{Zip} \rightarrow \text{City})\)
- Keys = \{\text{Addr, City}\} and \{\text{Addr, Zip}\}
- \((\text{Zip} \rightarrow \text{City})\) violates BCNF
- The method gives \{\text{Zip, City}\} and \{\text{Zip, Addr}\}
- But this fails to preserve \((\text{Addr, City} \rightarrow \text{Zip})\)

\[
\begin{array}{cccccccc}
\hline
10000 & NY & 10000 & 27 & Mott & 27 & Mott & NY & 10000 \\
20000 & NY & 20000 & 27 & Mott & 27 & Mott & NY & 20000 \\
\end{array}
\]
3NF: A Modification of BCNF

• $(Addr, City \rightarrow Zip), (Zip \rightarrow City),$

• Keys = \{Addr, City\} and \{Addr, Zip\}

• $(Zip \rightarrow City)$ violates BCNF:
  – But decomposing via $(Zip \rightarrow City)$ went too far, and broke a dependency

• Solution: Define a weaker condition, 3NF, for which $(Zip \rightarrow City)$ is not a violation

• Definition of 3NF:
  – For all nontrivial $(X \rightarrow Y)$, $X$ is a superkey or $Y$ is a subkey
Which Schemas are in 3NF?

- City, Addr, Zip
  - (City, Addr → Zip), (Zip → City)

- Person, Zip, City
  - (Person → Zip), (Zip → City)

- Person, Age, Weight
  - (Person → Age), (Person → Weight)

- Em, To, Pr, Ho, Sk, Lo, Ro
  - (Em → ToPr), (To → Pr), (EmTo → Ho), (SkLo → Ro), (Ro → Lo)
  - Keys = \{Em, Sk, Lo\} and \{Em, Sk, Ro\}
Redundant Functional Dependencies

- Consider the dependencies:
  1. (P → C)
  2. (P → Z)
  3. (Z → C)
  4. (P → ZC)

- This set is not minimal, because:
  - Removing (1) and (4) doesn’t change the overall constraint

- What algorithm can find and delete the redundant dependencies?
Removing Redundant Dependencies

• Let \( D \) be a set of dependencies

• Repeat until none can be removed:
  1. Choose a rule \( (X \rightarrow Y) \) in \( D \)
  2. Let \( D' \) be the other rules in \( D \)
  3. Let \( X^+ \) be the closure of \( X \) under \( D' \)
  4. If \( Y \subseteq X^+ \), then \( (X \rightarrow Y) \) is redundant, so remove it, by assigning \( D = D' \)

• The final value of \( D \) has no redundancies
**3NF Decomposition Algorithm**

1. Find keys, and check if the relation is already in 3NF
2. Decompose the right hand sides of the dependencies
3. Remove redundant attributes on the left hand sides
4. Remove redundant functional dependencies
5. Combine dependencies with same left hand sides
6. Create a relation for each functional dependency
7. Remove relations contained in other relations
8. If no relation contains a key of the original relation, add a relation whose attributes form such a key
Finding the Keys

- \((\text{Em} \rightarrow \text{ToPr}), (\text{To} \rightarrow \text{Pr}), (\text{EmTo} \rightarrow \text{Ho}), (\text{SkLo} \rightarrow \text{Ro}), (\text{Ro} \rightarrow \text{Lo})\)

- \(L = \text{attributes appearing only on left sides} = \text{EmSk}\)

- \(R = \text{attributes appearing only on right sides} = \text{PrHo}\)

- \(N = \text{attributes not appearing at all} = \{\}\)

- Every key must include \(L\) and \(N\)

- Every key must be disjoint from \(R\)

- Hence, \(\{\text{Em,Sk}\} \subseteq \text{key} \subseteq \{\text{Em,Sk,To,Lo,Ro}\}\)
Finding the Keys (cont’d)

- \(\{\text{Em,Sk}\} \subseteq \text{key} \subseteq \{\text{Em,Sk,To,Lo,Ro}\}\)
- \((\text{Em} \rightarrow \text{ToPr}), (\text{To} \rightarrow \text{Pr}), (\text{EmTo} \rightarrow \text{Ho}), (\text{SkLo} \rightarrow \text{Ro}), (\text{Ro} \rightarrow \text{Lo})\)
- \(\{\text{Em,Sk}\}^+ = \text{Em,Sk,To,Pr,Ho}\)
- \(\{\text{Em,Sk,Lo}\}^+ = \text{Em,Sk,To,Pr,Ho,Lo,Ro}\)
- \(\{\text{Em,Sk,Ro}\}^+ = \text{Em,Sk,To,Pr,Ho,Ro,Lo}\)
- Hence, keys are \(\{\text{Em,Sk,Lo}\}\) and \(\{\text{Em,Sk,Ro}\}\)
Decomposing right hand sides

Decompose the right hand sides of the functional dependencies, e.g., $X \rightarrow YZ$ becomes $X \rightarrow Y$ and $X \rightarrow Z$.

This gives us:

- $Em \rightarrow To$
- $Em \rightarrow Pr$
- $To \rightarrow Pr$
- $EmTo \rightarrow Ho$
- $SkLo \rightarrow Ro$
- $Ro \rightarrow Lo$
**Remove redundant attributes**

- Remove redundant attributes from the left sides
- First check \((\text{EmTo} \rightarrow \text{Ho})\) for redundancy on left:
  - Since \(\text{To}^+ = \text{Pr}\), \(\text{Em}\) is needed to obtain \(\text{Ho}\)
  - Since \(\text{Em}^+ = \text{EmToPrHo}\), \(\text{Em}\) suffices for \(\text{Ho}\)
  - Hence, we can replace with \((\text{Em} \rightarrow \text{Ho})\)
- Then check \((\text{SkLo} \rightarrow \text{Ho})\) for redundancy on left:
  - Both \(\text{Sk}\) and \(\text{Lo}\) are needed to get \(\text{Ho}\), so the rule cannot be replaced
The Simplified Rules

1. Em $\rightarrow$ To
2. Em $\rightarrow$ Pr
3. To $\rightarrow$ Pr
4. Em $\rightarrow$ Ho
5. SkLo $\rightarrow$ Ro
6. Ro $\rightarrow$ Lo
**Remove the Redundant Rules**

- \((\text{Em} \rightarrow \text{To}), (\text{Em} \rightarrow \text{Pr}), (\text{To} \rightarrow \text{Pr}), (\text{Em} \rightarrow \text{Ho}), (\text{SkLo} \rightarrow \text{Ro}), (\text{Ro} \rightarrow \text{Lo})\)

- Is \((\text{Em} \rightarrow \text{To})\) redundant? Compute \(\text{Em}^+\) using rules 2, 3, 4, 5, 6:
  - \(\text{Em}^+ = \text{EmPrHo}\)
  - Since \(\text{To}\) not in \(\text{EmPrHo}\), this rule is not redundant

- \((\text{Em} \rightarrow \text{Pr})\). Compute \(\text{Em}^+\) using 1, 3, 4, 5, 6:
  - \(\text{Em}^+ = \text{EmToPrHo}\)
  - Redundant, since \(\text{Pr}\) is in \(\text{Em}^+\)
– Hence, remove rule 2

● \((\text{To} \rightarrow \text{Pr})\). Compute \(\text{To}^+\) using 1, 4, 5, 6 (but not 2!):
  – \(\text{To}^+ = \text{To}\)
  – Hence, not redundant

● Similarly, rules 4, 5 and 6 are not redundant

● Resulting rules, after renumbering:
  1. \(\text{Em} \rightarrow \text{To}\)
  2. \(\text{To} \rightarrow \text{Pr}\)
  3. \(\text{Em} \rightarrow \text{Ho}\)
  4. \(\text{SkLo} \rightarrow \text{Ro}\)
  5. \(\text{Ro} \rightarrow \text{Lo}\)
Combining Functional Dependencies

• Combine all dependencies with the same left side

• This gives:
  1. Em → ToHo
  2. To → Pr
  3. SkLo → Ro
  4. Ro → Lo
Creating Relations

• Create a relation for each dependency

• This gives:
  1. EmToHo
  2. ToPr
  3. SkLoRo
  4. RoLo
Remove Redundant Relations

- Remove relations contained in other relations
- This gives:
  1. EmToHo
  2. ToPr
  3. SkLoRo
Assuring Storage of a Global Key

- If no relation contains a key of the original relation, then add a relation whose attributes form such a key.
- None of the relations contains EmSkLo or EmSkRo.
- Hence, one of these has to be added.
- Two choices for the final decomposition into 3NF:
  - EmToHo, ToPr, SkLoRo, EmSkLo
  - EmToHo, ToPr, SkLoRo, EmSkRo
Why was the Global Key Needed?

- Consider $R(A,B,X,Y)$, with $(A \rightarrow X)$ and $(B \rightarrow Y)$
- The global key is $AB$
- Hence, the decomposition $AX, BY$ is not in 3NF
- But $AX, BY, AB$ is in 3NF
Topics Covered in this (and previous) Unit

• Functional Dependencies
  – Closures, Superkeys, Keys, Implied Dependencies

• Desirable Properties of a Decomposition:
  – Non-redundancy, Lossless-Join, Preservation of Dependencies

• Normal Forms: BCNF and 3NF

• BCNF Decomposition Algorithm

• 3NF Decomposition Algorithm