CS186 Discussion #8

(Functional Dependencies)

Functional Dependencies

- X → Y reads "X determines Y"
- Used to detect redundancies and refine schema

id	rating	wage
1	3	20
2	2	10
3	3	20
4	2	10

Functional Dependencies

- X → Y reads “X determines Y”
- Used to detect redundancies and refine schema

id	rating	wage
1	3	20
2	2	10
3	3	20
4	2	10

rating → wage

Functional Dependencies

- X → Y reads "X determines Y"
- Used to detect redundancies and refine schema

id	rating	wage
1	3	20
2	2	10
3	3	20
4	2	10

id → rating, wage

Functional Dependencies

- Key → All attributes of relation

id	rating	wage
1	3	20
2	2	10
3	3	20
4	2	10

id → rating, wage

Armstrong's Axioms

- **Reflexivity**: if $X \supseteq Y$, then $X \rightarrow Y$
  - **Examples**: $A \rightarrow A$, $AB \rightarrow A$
- **Augmentation**: if $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
- **Transitivity**: if $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- Useful rules derived from Armstrong's Axioms:
  - **Union**: if $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
  - **Decomposition**: if $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$

Armstrong’s Axioms

- If $XA \rightarrow YA$, can you infer $X \rightarrow Y$?

Armstrong's Axioms

- If XA → YA, can you infer X → Y?
  - Trivial example: A → YA

Worksheet: Page 2

Consider the **Works_In**(**S**sn, parking_**L**ot_num, **D**epartment_id, s**I**nce) relation. If **S** (ssn) is a key for this relationship, what is the functional dependency we can infer from that?

Consider the **Works_In**(**S**sn, parking_**L**ot_num, **D**epartment_id, s**I**nce) relation. If **S** (ssn) is a key for this relationship, what is the functional dependency we can infer from that?

- $S \rightarrow SLDI$

If employees in the same department are given the same parking lot number, what additional functional dependency can we infer?

If employees in the same department are given the same parking lot number, what additional functional dependency can we infer?

- $D \rightarrow L$

Find the set of functional dependencies:

Flights(<u>**F**light_no, **D**ate</u>, f**R**om, **T**o, **P**lane_id),
ForeignKey(**P**lane_id)

Planes(<u>**P**lane_id</u>, t**Y**pe)

Seat(<u>**S**eat_no, **P**lane_id</u>, **L**egroom), ForeignKey(**P**lane_id)

Find the set of functional dependencies:

Flights(**F**light_no, **D**ate, f**R**om, **T**o, **P**lane_id),
ForeignKey(**P**lane_id)

Planes(**P**lane_id, t**Y**pe)

Seat(**S**eat_no, **P**lane_id, **L**egroom), ForeignKey(**P**lane_id)

- FD → RTP
- P → Y
- SP → L

Closures

- Functional dependency closure: F+
  - Set of all FDs implied by F, including trivial dependencies
  - Example: F = {A → B, B → C}
    - F+ = {A → B, B → C, A → C, A → A, A → AB, …}

Closures

- Attribute closure: X+
  - Given just X, what can we determine?
  - Example: F = {A → B, B → C}
    - A+ = ABC

Attribute Closures

- A methodical algorithm, given a set of FDs F:
  - Initialize X+ := X
  - Repeat until no change:
    - If U → V is in F such that U is in X+, add V to X+

- R = ABCDE
- F = {B → CD, D → E, B → A, E → C, AD → B }
- B+ = ?

Attribute Closures

- A methodical algorithm, given a set of FDs F:
  - Initialize X+ := X
  - Repeat until no change:
    - If U → V is in F such that U is in X+, add V to X+

- R = ABCDE
- F = {B → CD, D → E, B → A, E → C, AD → B }
- B+ = B

Attribute Closures

- A methodical algorithm, given a set of FDs F:
  - Initialize X+ := X
  - Repeat until no change:
    - If U → V is in F such that U is in X+, add V to X+

- R = ABCDE
- F = {B → CD, D → E, B → A, E → C, AD → B }
- B+ = BCD

Attribute Closures

- A methodical algorithm, given a set of FDs F:
  - Initialize X+ := X
  - Repeat until no change:
    - If U → V is in F such that U is in X+, add V to X+

- R = ABCDE
- F = {B → CD, D → E, B → A, E → C, AD → B }
- B+ = BCDE

Attribute Closures

- A methodical algorithm, given a set of FDs F:
  - Initialize X+ := X
  - Repeat until no change:
    - If U → V is in F such that U is in X+, add V to X+

- R = ABCDE
- F = {B → CD, D → E, B → A, E → C, AD → B }
- B+ = ABCDE

Attribute Closures

- A methodical algorithm, given a set of FDs F:
  - Initialize X+ := X
  - Repeat until no change:
    - If U → V is in F such that U is in X+, add V to X+

- R = ABCDE
- F = {B → CD, D → E, B → A, E → C, AD → B }
- B+ = ABCDE B is a key of R!

Boyce-Codd Normal Form (BCNF)

- Motivation: Schema design is hard, want a way to ensure a reasonable design
- BCNF ≈ “reasonable schema”

Boyce-Codd Normal Form (BCNF)

- Definition: Relation R with FDs F is in BCNF if for all $X \rightarrow A$ in $F+$:
  - $X \rightarrow A$ is reflexive (a trivial FD) OR
  - X is a superkey for R
    - Superkey: Key that does not need to be minimal

BCNF Decomposition

- If $X \rightarrow A$ violates BCNF, decompose R into R - A and XA
  - Repeat as necessary

- $R = ABCEG$
- $F = \{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$

BCNF Decomposition

- If $X \rightarrow A$ violates BCNF, decompose R into R - A and XA
  - Repeat as necessary

- R = ABCEG
- $F = \{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$
- ABEG, ABC

BCNF Decomposition

- If X → A violates BCNF, decompose R into R - A and XA
  - Repeat as necessary

- R = ABCEG
- F = {AB → C, AC → B, BC → G, E → G}
- ABEG, ABC

BCNF Decomposition

- If $X \rightarrow A$ violates BCNF, decompose R into R - A and XA
  - Repeat as necessary

- R = ABCEG
- F = $\{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$
- ABEG, ABC

BCNF Decomposition

- If $X \rightarrow A$ violates BCNF, decompose R into R - A and XA
  - Repeat as necessary

- R = ABCEG
- F = $\{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$
- ABEG, ABC
- ABE, EG, ABC

BCNF Decomposition

- If $X \to A$ violates BCNF, decompose R into R - A and XA
  - Repeat as necessary

- R = ABCEG
- F = {$AB \to C$, $AC \to B$, $BC \to G$, $E \to G$}
- ABEG, ABC
- ABE, EG, ABC

Decomposition Properties

- Lossless join: Can we reconstruct R?
  - Decomposing R into X and Y is lossless iff:
    - $X \cap Y \rightarrow X$, or
    - $X \cap Y \rightarrow Y$

Decomposition Properties

- Lossless join: Can we reconstruct R?
  - Decomposing R into X and Y is lossless iff:
    - $X \cap Y \rightarrow X$, or
    - $X \cap Y \rightarrow Y$
- Example:
  - ABC decomposed to AB, BC
  - FDs: $A \rightarrow B$, $C \rightarrow B$
  - This is lossy! $AB \cap BC = B \rightarrow B$

Decomposition Properties

- Lossless join: Can we reconstruct R?
  - Decomposing R into X and Y is lossless iff:
    - $X \cap Y \rightarrow X$, or
    - $X \cap Y \rightarrow Y$
- Example:
  - ABC decomposed to AC, BC
  - FDs: $A \rightarrow B$, $C \rightarrow A$
  - This is lossless! $AC \cap BC = C \rightarrow AC$

Decomposition Properties

- Dependency-preserving: Can we verify all FDs without joins?

- Example: ABE, EG, ABC
  - $F = \{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$

Decomposition Properties

- Dependency-preserving: Can we verify all FDs without joins?

- Example: ABE, EG, ABC
  - $F = \{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$

Decomposition Properties

- Dependency-preserving: Can we verify all FDs without joins?
- Example: ABE, EG, ABC
  - F = {AB → C, AC → B, BC → G, E → G}

Decomposition Properties

- Dependency-preserving: Can we verify all FDs without joins?
- Example: ABE, EG, ABC
  - $F = \{AB \rightarrow C, AC \rightarrow B, BC \rightarrow G, E \rightarrow G\}$
  - This dependency was not preserved!
  - We can fix this by adding BCG, but this may break BCNF.

Minimal Cover

- G for a set of FDs F
  - Closure of G = closure of F
  - Right hand side of each FD is G is a single attribute
- Implies lossless join and dependency preserving decomposition

- Example: $A \rightarrow B$, $ABCD \rightarrow E$, $EF \rightarrow GH$, $ACDF \rightarrow EG$

Minimal Cover

- G for a set of FDs F
  - Closure of G = closure of F
  - Right hand side of each FD is G is a single attribute
- Implies lossless join and dependency preserving decomposition

- Example: $A \rightarrow B$, $ABCD \rightarrow E$, $EF \rightarrow GH$, $ACDF \rightarrow EG$
- $A \rightarrow B$

Minimal Cover

- G for a set of FDs F
  - Closure of G = closure of F
  - Right hand side of each FD is G is a single attribute
- Implies lossless join and dependency preserving decomposition

- Example: A → B, ABCD → E, EF → GH, ACDF → EG
- A → B, ACD → E

Minimal Cover

- G for a set of FDs F
  - Closure of G = closure of F
  - Right hand side of each FD is G is a single attribute
- Implies lossless join and dependency preserving decomposition

- Example: A → B, ABCD → E, EF → GH, ACDF → EG
- A → B, ACD → E, EF → G, EF → H

Minimal Cover

- G for a set of FDs F
  - Closure of G = closure of F
  - Right hand side of each FD is G is a single attribute
- Implies lossless join and dependency preserving decomposition

- Example: $A \rightarrow B$, $ABCD \rightarrow E$, $EF \rightarrow GH$, $ACDF \rightarrow EG$
- $A \rightarrow B$, $ACD \rightarrow E$, $EF \rightarrow G$, $EF \rightarrow H$

Worksheet: Page 3

Now consider the attribute set R = ABCDE and the FD set $F = \{AB \rightarrow C, A \rightarrow D, D \rightarrow E, AC \rightarrow B\}$. Compute the closure for the following attributes.

- A:
- AB:
- B:
- D:

Now consider the attribute set R = ABCDE and the FD set F = {AB → C, A → D, D → E, AC → B}. Compute the closure for the following attributes.

- A: ADE
- AB: ABCDE
- B: B
- D: DE

Consider the relation with attributes ABCDE and FD's: AB → ABCDE, D → A, E → B

Is this in BCNF?

Consider the relation with attributes ABCDE and FD's: AB → ABCDE, D → A, E → B Is this in BCNF?

- **This is not BCNF**
  - D → A and E → B are not trivial, and neither D nor E are keys.
  - This is 3NF because both A and B are parts of a key of the relation (see the definition of 3NF).

Decompose R = ABCDEFG into BCNF, given the FD set:
$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

Decompose R = ABCDEFG into BCNF, given the FD set:
$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- ABEFG, ABCD

Decompose R = ABCDEFG into BCNF, given the FD set:
$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- ABEFG, ABCD

Decompose R = ABCDEFG into BCNF, given the FD set:
F = {AB → CD, C → EF, G → A, G → F, CE → F}.

- ABEFG, ABCD
- BEFG, ABCD, AG

Decompose R = ABCDEFG into BCNF, given the FD set:
F = {AB → CD, C → EF, G → A, G → F, CE → F}.

- ABEFG, ABCD
- BEFG, ABCD, AG
- BEG, FG, ABCD, AG

Decompose R = ABCDEFG into BCNF, given the FD set:
F = {AB → CD, C → EF, G → A, G → F, CE → F}.

- ABEFG, ABCD
- BEFG, ABCD, AG
- BEG, FG, ABCD, AG

Does BEG, FG, ABCD, AG preserve dependencies?

$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

Does BEG, FG, ABCD, AG preserve dependencies?
$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- No, $C \rightarrow EF$ and $CE \rightarrow F$ are not preserved.

Give a minimal cover for:

$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

Give a minimal cover for:

$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- $AB \rightarrow C$
- $AB \rightarrow D$

Give a minimal cover for:

$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- $AB \rightarrow C$
- $AB \rightarrow D$
- $C \rightarrow F$
- $C \rightarrow E$

Give a minimal cover for:

F = {AB → CD, C → EF, G → A, G → F, CE → F}.

- AB → C
- AB → D
- C → F
- C → E
- G → A

Give a minimal cover for:

$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- $AB \rightarrow C$
- $AB \rightarrow D$
- $C \rightarrow F$
- $C \rightarrow E$
- $G \rightarrow A$
- $G \rightarrow F$

Give a minimal cover for:

$F = \{AB \rightarrow CD,\ C \rightarrow EF,\ G \rightarrow A,\ G \rightarrow F,\ CE \rightarrow F\}$.

- $AB \rightarrow C$
- $AB \rightarrow D$
- $C \rightarrow F$
- $C \rightarrow E$
- $G \rightarrow A$
- $G \rightarrow F$

Give a minimal cover for:
$F = \{AB \rightarrow CD, C \rightarrow EF, G \rightarrow A, G \rightarrow F, CE \rightarrow F\}$.

- $AB \rightarrow C$
- $AB \rightarrow D$
- $C \rightarrow F$
- $C \rightarrow E$
- $G \rightarrow A$
- $G \rightarrow F$

More lossless practice:

F = {AB -> CDE, BE -> X, A -> E}

Is ABC, BCDEX a lossless decomposition?

More lossless practice:
F = {AB -> CDE, BE -> X, A -> E}
Is ABC, BCDEX a lossless decomposition?

- No, it is lossy. ABC ∩ BCDEX = BC, which is not a superkey of ABC nor BCDEX.

R: ABCDE

Given FD $=\{AE \rightarrow BC, AC \rightarrow D, CD \rightarrow BE, D \rightarrow E\}$

Give three candidate keys.

R: ABCDE

Given FD ={AE → BC, AC → D, CD → BE, D → E}

Give three candidate keys.

- AE, AC and AD are candidate keys, as each of their attribute closures include all attributes and no subset of them is a super key by itself.

R: ABCDE

Given FD ={AE → BC, AC → D, CD → BE, D → E}

Is R already in BCNF?

R: ABCDE

Given FD ={AE → BC, AC → D, CD → BE, D → E}

Is R already in BCNF?

- No, because both CD → BE and D → E violate BCNF.

R: ABCD

Given FD ={$A \rightarrow B$, $B \rightarrow D$, $C \rightarrow D$}

Decomposed to AB, CD, AC. Is this lossless?

R: ABCD
Given FD ={A → B, B → D, C → D}
Decomposed to AB, CD, AC. Is this lossless?

- Yes, a lossless decomposition would be: ABC CD which is lossless because C is a key for CD and then a further decomposition of ABC into AB and AC which is lossless because A is a key for AB.