Tutorial 4

Normalisation

CZ2007

Introduction to Databases

Tutorial 4

- BCNF: In all dependencies (FDs), LHS must contain key (cannot depend on partial key)

Third Normal Form (3NF)

- Definition: A table satisfies 3NF, if and only if for every non-trivial X→Y
  - Either X contains a key
  - Or each attribute in Y is contained in a key
- Example:
  - Given FDs: C→B, AB→C, BC→C
  - Keys: {AB}, {AC}
  - AB→C is OK, since AB is a key of R
  - C→B is OK, since B is in a key of R
  - BC→C is OK, since C is in AC and in BC (it is also trivial)
  - So R is in 3NF

R

A	B	C

Minimal Basis

- Given a set S of FDs, the minimal basis of S is a simplified version of S
- Previous example:
  - S = {A→BD, AB→C, C→D, BC→D}
  - A minimal basis: {A→B, A→C, C→D}
- Three conditions.
- Condition 1: For any FD in the minimal basis, its right hand side has only one attribute.
- Condition 2: No FD in the minimal basis is redundant.
- Condition 3: For each FD in the minimal basis, none of the attributes on the left hand side is redundant

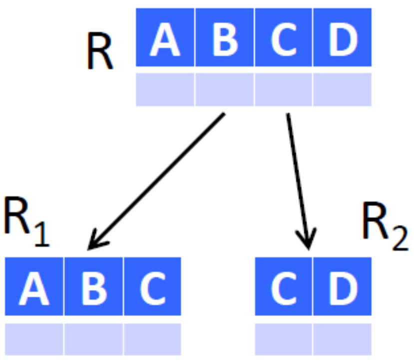
Algorithm for Minimal Basis

- Step 1: Transform the FDs, so that each right hand side contains only one attribute
- Step 2: Remove redundant FDs
- Step 3: Remove redundant attributes on the left hand side (lhs) of each FD

Tutorial 4

3NF Decomposition Algorithm

- Given: A table R, and a set S of FDs
  - e.g., R(A, B, C, D)
    S = {A→BD, AB→C, C→D, BC→D}
- Step 1: Derive a minimal basis of S
  - e.g., a minimal basis of S is
    {A→B, A→C, C→D}
- Step 2: In the minimal basis, combine the FDs whose left hand sides are the same
  - e.g., after combining A→B and A→C, we have {A→BC, C→D}
- Step 3: Create a table for each FD remained
  - R_1(A, B, C), R_2(C, D)
- Step 4: If none of the tables contain a key of the original table R, create a table that contains a key of R
- Step 5: Remove redundant tables (schema is a subset of another)



BCNF vs. 3NF

- BCNF Decomposition:
  - Avoids insertion, deletion, and update anomalies
  - Eliminates most redundancies
  - But does not always preserve all FDs
- 3NF Decomposition:
  - Avoids insertion, deletion, and update anomalies
  - May lead to a bit more redundancy than BCNF
  - Always preserve all FDs

Question 1

- A medical clinic database schema contains the following:
  - APPOINTMENT (patient-id, patient-name, doctor-id, doctor-name, appointment-date, appointment-time, clinic-room-no)
- Identify the functional dependencies in the schema, stating any assumptions made.
- Using these functional dependencies, normalise the schema to Third Normal Form.

Question 1

- APPOINTMENT(<u>patient-id</u>, patient-name, doctor-id, doctor-name, appointment-date, appointment-time, clinic-room-no)
- Identified FDs are:
  - patient-id → patient-name
  - doctor-id → doctor-name
  - appointment-date, appointment-time, clinic-room-no → doctor-id, patient-id
- Is APPOINTMENT in BCNF, 3NF?
  - Key: patient-id
  - Not in BCNF - doctor-id → doctor-name: doctor-id is not the key
  - Not in 3NF - doctor-id → doctor-name: doctor-name is not in a key of Appointment
- Do the FDs form a minimal basis? → NO

Question 1

- APPOINTMENT(<u>A</u>, B, C, D, E, F, G)
- Identified FDs are:
  - S = {A → B, C → D, EFG → AC}
- A minimal basis
  - Step 1: {A → B, C → D, EFG → A, EFG → C}
  - Step 2: A → B, C → D not redundant,

    EFG → A not redundant, {EFG}+ = {EFGC} does not contain A

    EFG → C not redundant, {EFG}+ = {EFGAB} does not contain C
  - Step 3: EFG → A has more than one attribute on the lhs,

    remove E, FG → A; given S, {FG}+={FG}, does not contain A, attribute E not redundant

    remove F, EG → A; given S, {EG}+={EG}, does not contain A, attribute F not redundant

    remove G, EF → A; given S, {EF}+={EF}, does not contain A, attribute G not redundant

    EFG → C, attributes on lhs are not redundant, similarly
  - Minimal basis: {A → B, C → D, EFG → A, EFG → C}

- Step 1: Transform the FDs, so that each right hand side contains only one attribute
- Step 2: Remove redundant FDs
- Step 3: Remove redundant attributes on the left hand side (lhs) of each FD

Question 1

- Step 2: In the minimal basis, combine the FDs whose left hand sides are the same
  - e.g., after combining A→B and A→C, we have {A→BC, C→D}
- Step 3: Create a table for each FD remained
  - R_1(A, B, C), R_2(C, D)
- Step 4: If none of the tables contain a key of the original table R, create a table that contains a key of R

- Minimal basis: {A → B, C → D, EFG → A, EFG → C}
- 3NF decomposition – S = {A → B, C → D, EFG → AC}
- Using the above identified FDs, we decompose the schema into 3NF:
  - PATIENT(<u>patient-id</u>, patient-name)
  - DOCTOR(<u>doctor-id</u>, doctor-name)
  - APPT(doctor-id, patient-id, <u>appointment-date, appointment-time, clinic-room-no</u>)

Question 2

2. Consider the relation `Courses(C, T, H, R, S, G)` whose attributes may be thought informally as course, teacher, hour, room, student, and grade. Let the set of FD's of Courses be:

   $C \rightarrow T$, $HR \rightarrow C$, $HT \rightarrow R$, $HS \rightarrow R$, and $CS \rightarrow G$.

   (a) What are all the keys for `Courses`?
   (b) Verify that the given FDs are their own minimal basis.
   (c) Use the 3NF decomposition algorithm to find a lossless-join, dependency-preserving decomposition.

Question 2(a)

(a) What are all the keys for Courses?

$C \rightarrow T$, $HR \rightarrow C$, $HT \rightarrow R$, $HS \rightarrow R$, and $CS \rightarrow G$.

- The usual procedure to find the keys is to take the closure of all 63 nonempty subsets.
- However, if we notice that none of the right sides of the FDs contains attributes H and S.
- Thus, we know that attributes H and S must be part of any key.
- We eventually will find out that **HS** is the only key for the Courses relation.

  i.e. $\{HS\}^+=\{HSRCTG\}$

Question 2(b)

(b) Verify that the given FDs are their own minimal basis.

$C \rightarrow T$, $HR \rightarrow C$, $HT \rightarrow R$, $HS \rightarrow R$, and $CS \rightarrow G$.

- Check if any of the FDs can be removed.
  - If we remove any one of the five FDs, the remaining four FDs do not imply the removed FD.
  - For example: Try C->T, if removed, then HR->C, and etc. {C}+={C}, does not contain T, so cannot be removed. Similarly for other FDs.

- Check if any of the LHS attribute of an FD can be removed without losing the dependencies.
  - This is not the case for the four FDs that contain two attributes on the left side. Thus, the given set of FDs is a minimal basis.
  - For example, HR->C, (i) remove H, we have R->C, given the set of FDs, {R}+={R}, does not contain C, so cannot be removed. (ii) remove R, we have H->C, given the set of FDs, {H}+={H}, does not contain C, so cannot be removed. Similarly for other FDs with more than one attributes on lhs.

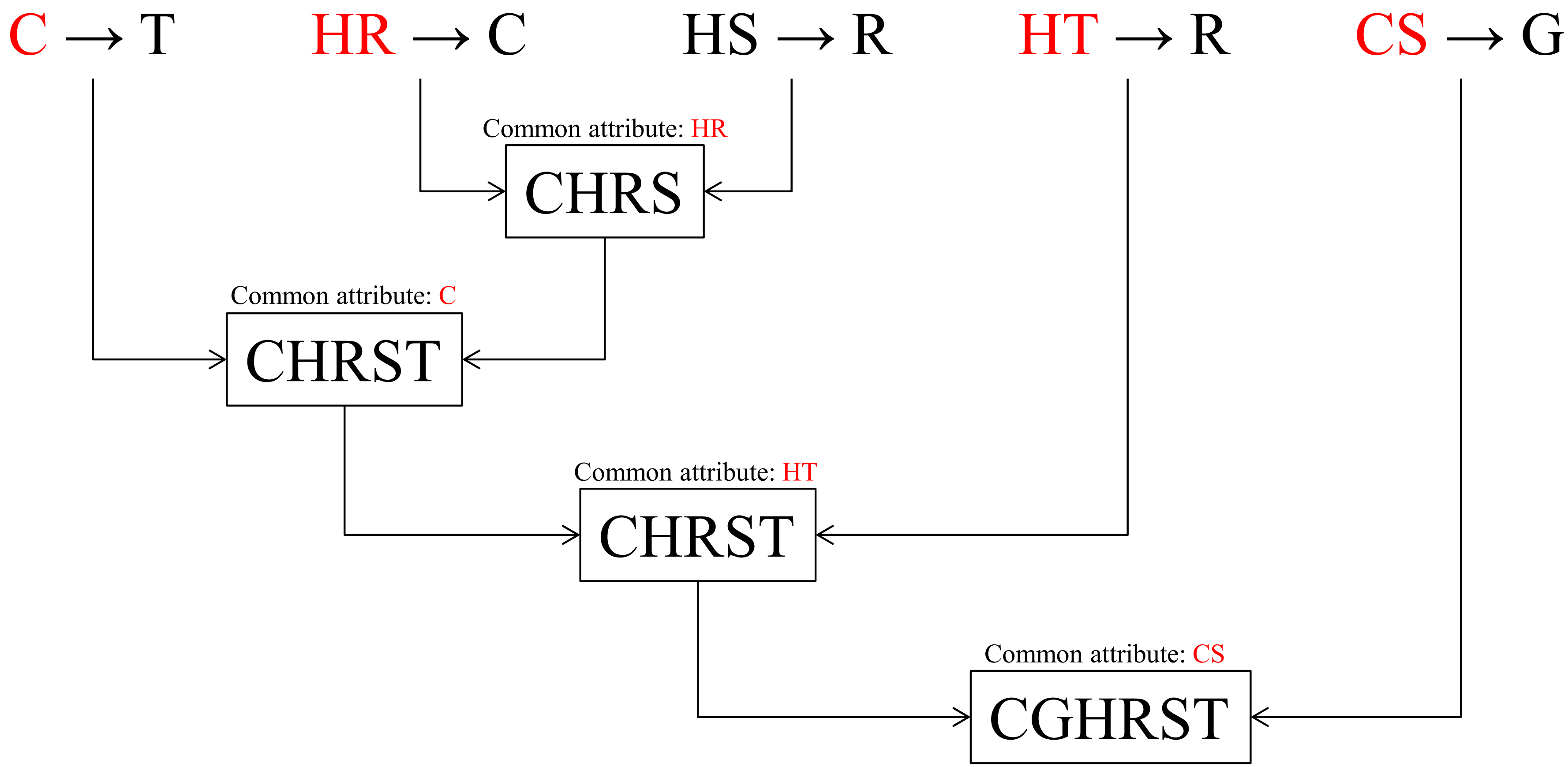
Question 2(c)

(c) Use the 3NF decomposition algorithm to find a lossless-join, dependency-preserving decomposition.

- Since the only key is HS, the given set of FDs has some dependencies that violate 3NF.
- We also know that the given set of FDs is a minimal basis.
- Thus, the decomposed relations are CT, HRC, HTR, HSR and CSG.
- Since the relation HSR contains a key, we do not need to add an additional relation.
- The final set of decomposed relations is CT, HRC, HTR, HSR and CSG.
- Since each decomposed relation came from a FD, the decomposition is FD preserving.

Question 2(c)

- The following sequence of joins shows that the decomposition is loss-less:



Question 3

3. Consider a relation $R(W, X, Y, Z)$ which satisfies the following set of FDs $G = \{Z \to W, Y \to X, Y \to Z, XW \to Y\}$, where G is a minimal basis.

(i) Decompose R into a set of relations in 3NF.
(ii) Is the decomposition also in BCNF? Explain your answer.

Question 3

(i) Decompose R into a set of relations in 3NF. R(W, X, Y, Z)

$G = \{Z \rightarrow W, Y \rightarrow X, Y \rightarrow Z, XW \rightarrow Y\}$

- G ={Z→W, Y→X, Y→Z, XW→Y}, where G is a minimal basis
- Use closure method to find keys
- Possible keys: Y, WX, XZ, i.e. {Y}+={WXYZ}, {WX}+={WXYZ}, {XZ}+={WXYZ}
- Minimal basis:
  - 1. M = {Z->W, Y->X, Y->Z, XW->Y}
  - 2. M = {Z->W, Y->X, Y->Z, XW->Y} – no redundant FD
  - 3. M = {Z->W, Y->X, Y->Z, XW->Y} – no redundant attribute lhs

- Step 1: Transform the FDs, so that each right hand side contains only one attribute
- Step 2: Remove redundant FDs
- Step 3: Remove redundant attributes on the left hand side (lhs) of each FD

- Combine FDs: {Z->W, Y->XZ, XW->Y}
- Decomposed relations: R1(Z, W), R2(X, Y, Z), R3(X, Y, W)

Question 3

(ii) Is the decomposition also in BCNF? Explain your answer.

- FD: {Z->W, Y->XZ, XW->Y}

Then all LHS of FDs are superkeys; therefore relations are in BCNF

Question 4

4. Consider a relation R(A,B,C,D,E) and FD's $A \rightarrow BC$, $CD \rightarrow E$, $E \rightarrow A$, and $B \rightarrow D$.

   (a) Is the decomposition R1(A,B,C) and R2(A,D,E) of R lossless or lossy? Justify your answer. Is this decomposition dependency preserving? If your answer is NO, then what is not preserved?

   (b) Is the decomposition R3(A,B,C,D) and R4(C,D,E) of R lossless or lossy? Justify your answer. Is this decomposition dependency preserving? If your answer is NO, then what is not preserved?

Question 4(a)

Consider a relation R(A,B,C,D,E) and FD's $A \rightarrow BC$, $CD \rightarrow E$, $E \rightarrow A$, and $B \rightarrow D$.

(a) Is the decomposition R1(A,B,C) and R2(A,D,E) of R lossless or lossy? Justify your answer. Is this decomposition dependency preserving? If your answer is NO, then what is not preserved?

- FDs: $A \rightarrow BC$, $E \rightarrow A$, $CD \rightarrow E$, $B \rightarrow D$
- Keys? A? E? CD? BC?
- (a) Decomposition: R1(A,B,C) and R2(A,D,E)
- (b) Decomposition: R3(A,B,C,D) and R4(C,D,E)
- Is decomposition lossless (can be joined back)?
- Is decomposition dependency preserving (no FDs lost)?

Question 4(a)

- Decomposition is lossless since
  - The two tables have a common attribute A, and
  - A is a superkey for R1(A,B,C)
- FDs that hold on R1(A,B,C)
  - $A \rightarrow BC$ since R1 contains A, B, and C
- FDs that hold on R2(A,D,E)
  - $E \rightarrow A$, since R2 contains A and E
- Two other FDs need to be checked: $CD \rightarrow E$, $B \rightarrow D$
  - Given $A \rightarrow BC$ and $E \rightarrow A$, we have:
  - $\{B\}^+ = \{B\}$, so $B \rightarrow D$ is not preserved
  - $\{CD\}^+ = \{CD\}$, so $CD \rightarrow E$ is not preserved
- Decomposition is NOT dependency-preserving

$\boxed{A \rightarrow BC,\ E \rightarrow A,\ CD \rightarrow E,\ B \rightarrow D}$

Question 4(b)

(b) Is the decomposition `R3(A,B,C,D)` and `R4(C,D,E)` of `R` lossless or lossy? Justify your answer. Is this decomposition dependency preserving? If your answer is NO, then what is not preserved?

$A \rightarrow BC$, $E \rightarrow A$, $CD \rightarrow E$, $B \rightarrow D$

- Decomposition is lossless since
  - The two tables have common attributes CD, and
  - CD is a superkey for R4(C,D,E)
- FDs that hold on R3(A,B,C,D)
  - $A \rightarrow BC$, $CD \rightarrow A$, and $B \rightarrow D$ since R3 contains A, B, C, and D
- FDs that hold on R4(C,D,E)
  - $CD \rightarrow E$, $E \rightarrow CD$
- One other FDs needs to be checked: $E \rightarrow A$
  - Given $A \rightarrow BC$, $B \rightarrow D$, $CD \rightarrow E$, $E \rightarrow A$ and, we have:
  - $\{E\}^+ = \{A, B, C, D, E\}$, so $E \rightarrow CD$ in R4, $CD \rightarrow A$ in R3, $E \rightarrow A$ is preserved
- Decomposition is dependency-preserving