Tutorial 3

Functional Dependencies and Normalisation

CZ2007
Introduction to Databases

Tutorial 3

Functional Dependencies:

- In general, how do we know whether a combination of attributes is bad?
- We need to check the correlations among those attributes

Formal Definition of FD

- Attributes $A_1, A_2, \ldots, A_m$, $B_1, B_2, \ldots, B_n$
- $A_1A_2 \ldots A_m \rightarrow B_1B_2 \ldots B_n$
- Meaning: There do not exist two objects that
  - Have the same values on $A_1, A_2, \ldots, A_m$
  - but different values on $B_1, B_2, \ldots, B_n$

Where Do FDs Come From?

- From common sense, knowledge in real world
- From application's requirements

Reasoning with FDs

- Armstrong's Axioms
  - Three axioms for FD reasoning
  - Easy to understand, but not easy to apply
- Axiom of Reflexivity
  - A set of attributes → A subset of the attributes
- Axiom of Augmentation
  - Given A → B
  - We always have AC → BC, for any C
- Axiom of Transitivity
  - Given A → B and B → C
  - We always have A → C

Reasoning with FDs

- Given A→B, BC→D
- Can you prove that AC → D?
- Proof
  - Given A→B, we have AC → BC (Augmentation)
  - Given AC→BC and BC → D, we have AC → D (Transitivity)

Tutorial 3

Functional Dependencies:

- We need to check the correlations among those attributes

Closure

- Let S = $\{A_1, A_2, \ldots, A_n\}$ be a set S of n attributes
- Closure of S is the set of attributes that can be determined by $A_1, A_2, \ldots, A_n$
- Notation: $\{A_1, A_2, \ldots, A_n\}^+$
- Example
  - Given $A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$, $D \rightarrow E$
  - $\{A\}^+ = \{A, B, C, D, E\}$
  - $\{B\}^+ = \{B, C, D, E\}$
  - $\{D\}^+ = \{D, E\}$
  - $\{E\}^+ = \{E\}$

Closure & FD

- To prove that $X \rightarrow Y$ holds, we only need to show that $\{X\}^+$ contains Y

Keys of a Table

- Superkeys
  - A set of attributes that determines all other attributes in a table
- Keys
  - A minimal set of attributes that determines all other attributes in a table
- Example
  - R(A, B, C, D)
  - Given: $A \rightarrow BCD$, $BC \rightarrow A$
  - A is a key and a superkey
  - BC is also a key and a superkey
  - AB is not a key; it is only a superkey
- A table may have multiple keys
- In that case, each key is referred to as a candidate key

Finding the Keys: Algorithm

- Check all possible combinations of attributes in the table
  - Example: A, B, C, AB, BC, AC, ABC
- For each combination, compute its closure
  - Example: $\{A\}^+ = \ldots$, $\{B\}^+ = \ldots$, $\{C\}^+ = \ldots$, …
- If a closure contains ALL attributes, then the combination might be a key (or superkey)
  - Example: $\{A\}^+ = \{ABC\}$
- Make sure that you select only keys
  - Example: $\{A\}^+ = \{ABC\}$, $\{AB\}^+ = \{ABC\}$, don't select AB

Small Trick

- Always check small combinations first
- In general, if an attribute that does not appear in the right hand side of any FD, then it must be in every key

Tutorial 3

Boyce-Codd Normal Form (BCNF)

- A table R is in BCNF, if and only if
  - The left hand side of every non-trivial FD contains a key of R
- R(A, B)
- Given FDs: A→B
- Key: A
- All FDs on R: A→B, AB→A, AB→B, AB→AB
  - AB→A, AB→B, AB→AB: trivial
  - A→B: The left hand side contains a key
- Therefore, R is in BCNF

BCNF: Straightforward Checking

- Given: A table R, A set of FDs on R
- Step 1: Derive the keys of R
- Step 2: Derive all non-trivial FDs on R
  - This is too time-consuming
  - Trick: Only check the FDs given on R instead of all FDs
- Step 3: For each non-trivial FD, check if its left hand side contains a key
- Step 4: If all FDs pass the check, then R is in BCNF; otherwise, R is not in BCNF

BCNF: Intuition

- Basically, BCNF requires that there forbids any non-trivial X→Y where X does not contain a key
- Why does this make sense?
- Intuition: Such X→Y indicates that the table has some redundancies

BCNF Intuition: Example

Name	{ NRIC	Phone }	Address
Alice	1234	67899876	Jurong East
Alice	1234	83848384	Jurong East
Bob	5678	98765432	Pasir Ris

- NRIC → Name, Address
- Key: {NRIC, Phone} (single composite key)
- NRIC decides Name and Address
- Therefore, every time NRIC is repeated, Name and Address would also be repeated
- Since NRIC is not a key, the same NRIC can appear multiple times in the table
- This leads to redundancy
- BCNF prevents this

Tutorial 3

BCNF Decomposition Algorithm

- Input: A table R BCNF
- Step 1: Find a FD $X \rightarrow Y$ on R that violates BCNF
  - If cannot find, stop, R in BCNF
- Step 2: Compute the closure $\{X\}^+$
- Step 3: Break R into two tables R_1 and R_2
  - R_1 contains all attributes in $\{X\}^+$
  - R_2 contains all attributes NOT in $\{X\}^+$ plus X
- Repeat Steps 1-3 on R_1 and R_2

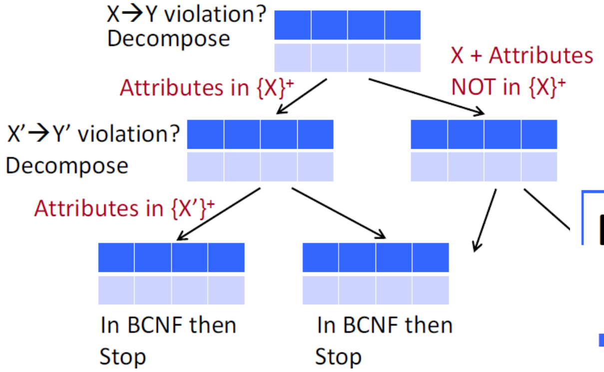
BCNF Decomposition: Example

- R(A, B, C, D)
- Given: $A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$
- Key of R: A
- Step 1: $B \rightarrow C$ is a violation
- Step 2: $\{B\}^+ = \{BCD\}$
- Step 3: Decompose R into two tables R_1 and R_2
  - R_1(B, C, D) contains all attributes in the closure
  - R_2(A, B) contains all attributes NOT in the closure plus B, LHS of violating FD
- Step 4: Check R_1 and R_2, decompose if necessary

BCNF Decomposition: Example

- R(A, B, C, D)
- Given: $A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$
- Key of R: A
- Previous results: R_1(B, C, D), R_2(A, B)
- Is R_2 in BCNF?
  - Yes. So R_2 is done
- Is R_1 in BCNF?
  - No. Key of R_1 is B and $C \rightarrow D$ is a violation.
- Decompose R_1 into R_3 and R_4
  - $\{C\}^+ = \{CD\}$
  - R_3(C, D) contains all attributes in $\{C\}^+$
  - R_4(B, C) contains C and all attribute NOT in $\{C\}^+$
- Are R_3 and R_4 in BCNF?
  - Yes. So we stop here.
- Final BCNF Decomposition
  - R_2(A, B)
  - R_3(C, D)
  - R_4(B, C)

Decompose, until all are in BCNF



Properties of BCNF Decomposition

- Good properties
  - No update or deletion anomalies
  - Very small redundancy
  - The original table can always be reconstructed from the decomposed tables if functional dependencies are preserved (this is called the lossless join property)
  - Reconstruction is at the schema level only if some FDs not preserved

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)
- Show, with suitable examples, the insertion anomalies that the schema is liable to encounter.

Example of insertion anomalies:

Data Anomalies

Name	NRIC	PhoneNumber	HomeAddress
Alice	1234	67899876	Jurong East
Alice	1234	83848384	Jurong East
Bob	5678	98765432	Pasir Ris

- Primary key of the table:
  (NRIC, PhoneNumber), one composite key
- Fourth, insertion anomalies:
  - Name = Cathy, NRIC = 9394, HomeAddress = YiShun
  - Can we insert this information into the table?
  - No. (Note: Primary key attributes cannot be NULL)

Question 1

APPOINTMENT (patient-id, patient-name, doctor-id, doctor-name, appointment-date, appointment-time, clinic-room-no)

- Clearly doctor-id → doctor-name is an FD and LHS is not superkey
- In order to insert a new patient, all the attributes of the doctor who is treating the patient must be also included.
- We must ensure that the doctor attributes are consistent with other patients under this doctor.
- If a new doctor joins and has not seen any patients, we cannot enter the doctor's details.
- We cannot fill patient information with nulls since this will violate the primary key condition (since the patient id is primary key).

Question 2

2. Consider the relation ADDRESS having attributes Street, City, State and Zip. Assume that for any given zipcode, there is just one city and state. Also, for any given street, city, and state, there is just one zipcode.

(a) Infer all possible functional dependencies (FDs) for this relation.
(b) Which are possible minimal keys?

Question 2

- Let us denote attributes STREET, CITY, STATE, ZIP as A, B, C and D respectively.
- Then, we have D->BC and ABC->D.
- Closure:

{A}+={A}; {B}+={B}; {C}+={C}; {D}+={DBC}

{AB}+={AB}; {AC}+={AC}; {AD}+={ABCD}; {BC}+={BC}; {BD}+={BDC}; {CD}+={CDB};

{ABC}+={ABCD}; {ABD}+={ABCD}; {ACD}+={ABCD}; {BCD}+={BCD}

- (a) FD's: D->BC; AD->BC; BD->C; CD->B; ABC->D; ADB->C; ACD->B;
- (b) Minimal keys: ABC, AD

Question 3

3. Prove the following properties using Armstrong's axioms or reject it by counterexample relations.

   (a) $A \rightarrow B \Rightarrow AC \rightarrow B$

   (b) $A \rightarrow C$ and $AB \rightarrow C \Rightarrow B \rightarrow C$

Question 3(a)

(a) $A \rightarrow B \Rightarrow AC \rightarrow B$

- $A \rightarrow B \Rightarrow AC \rightarrow BC$ (Augmentation Rule)
- $BC \rightarrow B$ (Reflexivity Rule)
- $AC \rightarrow BC$ and $BC \rightarrow B \Rightarrow AC \rightarrow B$ (Transitivity Rule)

Question 3(b)

(b) $A \rightarrow C$ and $AB \rightarrow C \Rightarrow B \rightarrow C$

- Consider the records: [by using a counterexample]

  (a1, b1, c1)

  (a2, b1, c2)

- Both $(A \rightarrow C)$ and $(AB \rightarrow C)$ are true but $(B \rightarrow C)$ does not hold

Question 4

4. Consider a relation R(A, B, C, D) with the following FDs: $B \rightarrow C$, $D \rightarrow B$

   (a) Find the key(s) of R.
   (b) Is this relation in BCNF? Why or why not? If it is not, decompose R

   into a collection of relations that are in BCNF.

Question 4

Consider a relation R(A, B, C, D) with the following FDs: $B \to C$, $D \to B$

(a) Find the key(s) of R.

- (a) Key: AD

  A must be in every key:

  {AB}+={ABC},
  {AC}+={AC},
  {AD}+={ABCD}

Question 4

Consider a relation R(A, B, C, D) with the following FDs: $B \rightarrow C$, $D \rightarrow B$

(b) Is this relation in BCNF? Why or why not? If it is not, decompose R into a collection of relations that are in BCNF.

- Not BCNF –

- A table R is in BCNF, if and only if
  - The left hand side of every non-trivial FD contains a key of R

- $B \rightarrow C$

- Step 1: Find a FD X→Y on R that violates BCNF

- {B}+={BC}

- Step 2: Compute the closure $\{X\}^+$

Question 4

- Step 3: Break R into two tables R_1 and R_2
  - R_1 contains all attributes in $\{X\}^+$
  - R_2 contains all attributes NOT in $\{X\}^+$ plus X

- By first decomposing on $B \rightarrow C$, we get $R1(B, C)$ and $R2(A, B, D)$.

  Is R1 in BCNF? Yes – key of R1 is B

  - If a table has only two attributes, then it must be in BCNF

  Is R2 in BCNF? No - Key of R2 is AD, therefore $D \rightarrow B$ is a violation

  - Repeat Steps 1-3 on R_1 and R_2

- $R2$ is not in BCNF due to violating FD: $D \rightarrow B$, so we decompose further: {D}+={BD}

- R3(A,D) and R4(B, D). [both are in BCNF]

Question 4

Note:

- The BCNF decomposition of a table may not be unique

<u>Another way of doing it:</u>

- By first decomposing on $D \rightarrow B$, we get $R1(B, C, D)$ and $R2(A, D)$.
- $R1$ is not in BCNF due to violating FD $B \rightarrow C$, so we must decompose further.
- In both cases, we end up with three relations, $Ra(A, D)$, $Rb(B, C)$ and $Rc(B, D)$.

Question 5

5. Prove that every two-attribute relationis in BCNF.

Question 5

- Case 1: $A \rightarrow B$ holds, but $B \rightarrow A$ does not

A is the key. The only nontrivial FD is $A \rightarrow B$; no BCNF violation.

- Case 2: $B \rightarrow A$ holds, but $A \rightarrow B$ does not

B is the key. The only nontrivial FD is $B \rightarrow A$; no BCNF violation.

- Case 3: Both $A \rightarrow B$ and $B \rightarrow A$ hold

A and B are both keys; no BCNF violation.

- Case 4: AB is key

So we have $AB \rightarrow AB$, which is trivial.