NORMALIZATION
Tutorial

SWEN304/SWEN439

Lecturer: Dr Hui Ma

Engineering and Computer Science

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Outline

- Normal forms
- 3NF normalization: Synthesis algorithm
- BCNF normalization: Decomposition algorithm

Understanding Normal Forms

- The fact that each relation schema key functionally determines each relation schema attribute is crucial for understanding 2NF, 3NF, and BCNF
- e.g.

  $R = \{A, B, C, D\}$, $F = \{AB \rightarrow C, B \rightarrow D\}$, $K = AB$

  $AB \rightarrow A$, $AB \rightarrow B$ *trivial FDs*

  $AB \rightarrow C$ *in F*

  $B \rightarrow D$ *with D partially functional depends on B*

  Scheme key functionally determines every attribute in the schema

First Normal Form and Second Normal Form

- A relation schema is in **first normal form (1NF)** if the domain of its each attribute has only atomic values
  - No relation schema attribute is allowed to be composite or multi-valued

- A relation schema *R* is in **second normal form (2NF)** if no non-prime attribute in *R* is **partially** functionally dependent on any relation schema *R* key

First Normal Form Example

- ***Grades*** $(\{StudId, StName, NoOfPts, CourId, Grd\},$
  $\{StudId \rightarrow StName + NoOfPts, StudId + CourId \rightarrow Grd\})$
- $K(Grades) = StudId + CourId$
- in 1NF but not in 2NF

StudId	*CourId*	*StName*	*NoOfPts*	*Grd*
007	M114	James	80	A+
131	C102	Susan	18	B-
007	C102	James	80	A
555	M114	Susan	18	B+
007	C103	James	80	A+
131	M214	Susan	18	ω

A new student can not be inserted until she/he enrolls
If a student passes a new exam, all the tuples have to be examined…

Second Normal Form

$$\textbf{Lecturer}(\{LecId, LeName, CourId, CoName\}, \{LecId \to LeName, LecId \to CourId, LecId \to CoName, CourId \to CoName\})$$

$$K(Lecturer) = LecId$$

CourId	*CoName*	*LecId*	*LeName*
M114	Math	*777*	Mark
C102	Java	*101*	Ewan
M114	Math	*999*	Vladimir
C103	Algorith	*99*	Peter
M214	Math	*333*	Peter
C201	C++	*222*	Robert
C101	Inet	*820*	Ray

New Course data can not be inserted without knowing who is going to lecture it
If a lecturer resigns, Course data will be lost

Third Normal Form and BCNF

- A relation schema $N(R, F)$ with a set of keys $K(N)$ is in **third normal form** (3NF) if for each nontrivial functional dependency $X \rightarrow A$ holds in F, **either** X is a superkey of N, **or** A is a prime attribute of N

  - A relation schema is in **third normal form (3NF)** if it is in 2NF, *and* no non-prime attribute is **transitively** functionally dependent on any relation schema key

- The relation schema (R, F) is in the **Boyce-Codd Normal Form (BCNF)**, if the left-hand side of each nontrivial functional dependency in F contains a relation schema key

Third Normal Form

- ***Employee*** $(\{EmpId, EmpName, SSN\},$
  $\{EmpId \to SSN, SSN \to EmpId,$
  $EmpId \to EmpName, SSN \to EmpName\}),$
  $K(Employee) = \{EmpId, SSN\}$

- is in 3NF (even in BCNF)?

- LHS of each nontrivial FD in F is a superkey

Third Normal Form

- $\textbf{\textit{Lecturer}}(\{LecId,\ LeName,\ CourId\}, \{LecId \rightarrow LeName,\ LecId \rightarrow CourId\},$
  $K(Lecturer) = LecId,\ Null(Lecturer,\ CourId) = Yes$
- is in 3NF (and even in BCNF)

LecId	*LeName*	*CourId*
777	Mark	M114
101	Ewan	C102
999	Vladimir	M114
99	Peter	C103
333	Peter	M214
222	Robert	C201
444	Ian	ω

These relations are free of update anomalies:
- Ian is not teaching any course
- C101 does not have a teacher

CourId	*CoName*
M114	Math
C102	Java
C103	Algorithm
M214	Math
C201	C++
C101	Inet

3NF but not BCNF

3NF, but not BCNF

LecId	StudId	CourId	Grade
777	*007*	M114	A+
101	*131*	C102	B+
101	*007*	C102	B
999	*555*	M114	C
99	*007*	C103	A
333	*131*	M214	ω
222	*555*	C201	A
222	*007*	C201	A+

LecId	LeName	CourId
777	Mark	M114
101	Ewan	C102
999	Vladimir	M114
99	Peter	C103
333	Peter	M214
222	Robert	C201
444	Ian	ω

StudId	StName
007	James
131	Susan
555	Susan
909	Paul

CourId	CoName
M114	Math
C102	Java
C103	Algorit
M214	Math
C201	C++
C101	Inet

Given ***Stud_Cour_Lec*** ({*StudId*, *CourId*, *LecId*, *Grade* },
{***LecId*** $\rightarrow$ ***CourId***, *StudId+CourId* $\rightarrow$*LecId*, *StudId+CourId* $\rightarrow$*Grade* })

Problem:

- Information about the relationship between lecturers and courses is stored twice
- Update of CourId for any LecId need to check ***LecId*** $\rightarrow$ ***CourId***
- Delete a lecture will delete relationship of student and course.

Lossless 3NF Decomposition

Synthesis Algorithm (simplified)

Input: (U, F)

Output: $S = \{(R_i, K_i) | i = 1, \ldots, n\}$ (*K_i is the relation schema key*)

1. Find a minimal cover G of F
2. Group FDs from G according to the same left-hand side. For each group of FDs

$$(X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_k),$$

   make one relation schema in S

$$(\{X, A_1, A_2, \ldots, A_k\}, X)$$

3. If none of relation schemes in S contain a key of (U, F), create a new relation scheme in S that will contain only a key of (U, F)

Example 1: 3NF Decomposition (1)

- ***Faculty*** = (*U*, *F*)

  $U = \{StudId,\ StName,\ NoPts,\ CourId,\ CoName,\ LecId,\ LeName,\ Grade\}$

  $F = \{StudId \to StName,\ StudId \to NoPts,\ CourId \to CoName,\ LecId \to LeName,\ LecId \to CourId,\ StudId + CourId \to Grade,\ StudId + CourId \to LecId\}$

- Step 1. Minimal cover is *F*

  Step 2. Groups:

  $(StudId \to StName,\ StudId \to NoPts)$
  $(CourId \to CoName)$
  $(LecId \to LeName,\ LecId \to CourId)$
  $(StudId + CourId \to Grade,\ StudId + CourId \to LecId)$

Example 1: 3NF Decomposition (2)

- Step 2 Relation schemas:

  S = {(***Student*** (*<u>StudId</u>*, *StName*, *NoOfPts*),
  Course (*<u>CourId</u>*, *CoName*),
  Lecturer *(<u>LecId</u>*, *LeName*, *CourId*),
  St_Le_Pa (*<u>StudId, CourId</u>*, *LecId*, *Grade*)}

- Step 3 Universal relation key is in ***St_Le_Pa***

  (*<u>StudId + CourId</u>*)$^+$ ={*StudId*, *StName*, *NoPts*, *CourId*, *CoName*, *LecId*,*LeName*, *Grade* }

- So, the decomposition is lossless and dependency preserving

Example 1: 3NF Decomposition (3)

Faculty database

Student

StudId	*StName*	*NoPts*
007	James	80
131	Susan	18
555	Susan	18
010	John	0

Course

CourId	*CoName*
C102	Java
M114	Math
C103	Algorith
M214	Math
C201	C++

Example 1: 3NF Decomposition (4)

Faculty database

Lecturer

CourId	LecId	LeName
C102	101	Ewan
M114	999	Vladimir
C103	99	Peter
M214	333	Peter
C201	222	Robert
C101	820	Ray

An update anomaly would arise if a lecturer decides to resign or change the course

Cour_Stud_Lec

CourId	StudId	LecId	Grd
M214	007	333	A+
C102	131	101	B-
C102	007	101	A
M114	555	999	B+
C103	007	99	A+
M214	131	333	ω
C201	555	222	ω
C201	007	222	A+
C101	010	820	ω

Example 2: 3NF Decomposition (exercise)

- $U = \{EmpId, LicenceNo, IRNo, EmpName\}$
  $F = \{EmpId \rightarrow LicenceNo, LicenceNo \rightarrow EmpId, EmpId \rightarrow IRNo, IRNo \rightarrow EmpId, EmpId \rightarrow EmpName\}$
- Is *U* in 3NF? If not decompose it in 3NF

Example 3: 3NF Decomposition (exercise)

- $U = \{A, B, C, D\}$, $F = \{A \rightarrow B, B \rightarrow C\}$
- Is U in 3NF? If not decompose it into 3NF

BCNF Decomposition

Decomposition algorithm:

Input: (U, F)

Output: $S = \{(R_i, F_i) | i = 1,\ldots, n\}$

1. Set $S := \{(U, F)\}$
2. While there is a relation schema (R, G) in S that is not in BCNF do

   2.1 Choose a functional dependency $X \rightarrow Y$ in G that violates BCNF,

   2.2 Replace (R, G) with $(R - Y, G|_{R-Y})$ and $(XY, G|_{XY})$

- The final result will be a lossless BCNF-decomposition

Projection of a Set of FDs

Examples:

- $F_1 = \{A \to B,\ B \to C,\ C \to D\},\ W = \{A,\ D\}$

  $F_1|_W = \{A \to D\}$

- $F_2 = \{A \to B,\ B \to C,\ C \to A\},\ W = \{A,\ B\}$

  $F_2|_W = \{A \to B,\ B \to A\}$

Exercise: $F_3 = \{AB \to C,\ C \to D,\ D \to B\},\ W = \{A,\ C,\ D\},$

$F_3|_W = ?$

Example 4: BCNF Decomposition (1)

- Take the ***Faculty*** = (*U*, *F*) as in Example 1

$U = \{StudId, StName, NoPts, CourId, CoName, LecId, LeName, Grade\}$

$F = \{StudId \rightarrow StName + NoPts, CourId \rightarrow CoName, LecId \rightarrow LeName + CourId, StudId + CourId \rightarrow Grade + LecId\}$

$K = \{StudId + CourId, StudId + LecId\}$

- Step 2

Faculty is not BCNF due to, say, $StudId \rightarrow StName + NoPts$, so

$S_1 = \{$***Student*** $(\{StudId, StName, NoOfPts\}, \{StudId \rightarrow StName + NoPts\})$

Faculty1 $(\{StudId, CourId, CoName, LecId, LeName, Grade\}, \{CourId \rightarrow CoName, LecId \rightarrow LeName + CourId, StudId + CourId \rightarrow Grade + LecId\})\}$

Example 4: BCNF Decomposition (2)

- *Student* is BCNF, but *Faculty1* is not due to

  CourId → *CoName*

- So decompose alone *CourId* → *CoName*

S_2 = {

Student ({*StId, StName, NoPts* }, {*StId* →*StName* + *NoPts* }),

Course ({*CourId, CoName* }, {*CourId* → *CoName* }),

Faculty2 ({*StId, CourId, LecId, LeName, Grade* }, {*LecId* →*LeName* + *CourId, StId* + *CourId* →*Grade + LecId* })

}

- Now, *Course is BCNF, but Faculty2 is not due to*

  LecId →*LeName* + *CourId*

Example 4: BCNF Decomposition (3)

$S_3 = \{$***Student***$(\{StId, StName, NoPts\}, \{StId \to StName + NoPts\})$,

Course$(\{CourId, CoName\}, \{CourId \to CoName\})$,

Lecturer$(\{LecId, CourId, LeName\}, \{LecId \to LeName + CourId\})$,

Stud_Lect$(\{StId, LecId, Grade\}, \{StId + LecId \to Grade\})\}$

- S_3 is BCNF
- $StudId + CourId \to Grade$ is in $(\bigcup_{i=1}^{n} F_i)^+$
- But FD $StudId + CourId \to LecId$ is lost

Example 5: FDs cannot be preserved (3)

LecId	*StudId*
777	*007*
101	*131*
101	*007*
999	*555*
99	*007*
333	*131*
222	*555*
222	*007*

StudId	*CourId*	*Grade*
007	M114	A+
131	C102	B+
007	C102	B
555	M114	C
007	C103	A
131	M214	ω
555	C201	A
007	C201	A+

LecId	*CourId*
777	M114
101	C102
999	M114
99	C103
333	M214
222	C201
444	ω

1. If a lecturer resigns or starts teaching another course, students' grades are not lost
2. Information about the relationship between lecturers and courses is stored only once
3. **If a lecturer resigns, we loose only information regarding his/her relationship with students**