Өгөгдлийн сангийн үндэс (CSII202 - 3 кр)
Database Systems

Lecture 8: Normalisation to BCNF

МУИС, ХШУИС, МКУТ-ийн багш

Маг. Довдонгийн Энхзол

Хүн мэргэжилдээ үнэнч байвал, мэргэжил нь хүнийг тэжээдэг юм.

Хүний их эмч Доржпалам

In This Lecture

- More normalisation
  - Brief review of relational algebra
  - Lossless decomposition
  - Boyce-Codd normal form (BCNF)
  - Higher normal forms
  - Denormalisation
- For more information
  - Connolly and Begg chapter 14
  - Ullman and Widom chapter 3.6

Normalisation so Far

- First normal form
  - All data values are atomic
- Second normal form
  - In 1NF plus no non-key attribute is partially dependent on a candidate key
- Third normal form
  - In 2NF plus no non-key attribute depends transitively on a candidate key

Lossless decomposition

- To normalise a relation, we used projections
- If R(A,B,C) satisfies $A \rightarrow B$ then we can project it on A,B and A,C without losing information
- Lossless decomposition:

$R = \pi_{AB}(R) \bowtie \pi_{AC}(R)$

where $\pi_{AB}(R)$ is projection of R on AB and $\bowtie$ is natural join.

- Reminder of projection:

R

A	B	C

$\pi_{AB}(R)$

A	B

Relational algebra reminder: selection

R

A	B	C	D
1	x	c	c
2	y	d	e
3	z	a	a
4	u	b	c
5	w	c	d

$\sigma_{C=D}(R)$

A	B	C	D
1	x	c	c
3	z	a	a

Relational algebra reminder: product

R1

A	B
1	x
2	y

R2

A	C
1	w
2	v
3	u

R1×R2

A	B	A	C
1	x	1	w
1	x	2	v
1	x	3	u
2	y	1	w
2	y	2	v
2	y	3	u

While I am on the subject…

SELECT A,B

FROM R1, R2, R3

WHERE (some property α holds)

translates into relational algebra

$\pi_{A,B}\ \sigma_{\alpha}\,(R1 \times R2 \times R3)$

Relational algebra: natural join

$$R1 \bowtie R2 = \pi_{R1.A,B,C}\ \sigma_{R1.A = R2.A}\ (R1 \times R2)$$

R1

A	B
1	x
2	y

R2

A	C
1	w
2	v
3	u

$R1 \bowtie R2$

A	B	C
1	x	w
2	y	v

When is decomposition lossless: Module → Lecturer

R

Module	Lecturer	Text
DBS	nza	CB
DBS	nza	UW
RDB	nza	UW
APS	rcb	B

$\pi_{\text{Module,Lecturer}} R$

Module	Lecturer
DBS	nza
RDB	nza
APS	rcb

$\pi_{\text{Module,Text}} R$

Module	Text
DBS	CB
DBS	UW
RDB	UW
APS	B

When is decomposition is not lossless: no fd

S

First	Last	Age
John	Smith	20
John	Brown	30
Mary	Smith	20
Tom	Brown	10

$\pi_{\text{First,Last}} S$

First	Last
John	Smith
John	Brown
Mary	Smith
Tom	Brown

$\pi_{\text{First,Age}} S$

First	Age
John	20
John	30
Mary	20
Tom	10

When is decomposition is not lossless: no fd

$\pi_{First,Last} S \bowtie \pi_{First,Last} S$

First	Last	Age
John	Smith	20
John	*Smith*	*30*
John	*Brown*	*20*
John	Brown	30
Mary	Smith	20
Tom	Brown	10

$\pi_{First,Last} S$

First	Last
John	Smith
John	Brown
Mary	Smith
Tom	Brown

$\pi_{First,Age} S$

First	Age
John	20
John	30
Mary	20
Tom	10

Normalisation Example

- We have a table representing orders in an online store
- Each entry in the table represents an item on a particular order

- Columns
  - Order
  - Product
  - Customer
  - Address
  - Quantity
  - UnitPrice
- Primary key is {Order, Product}

Functional Dependencies

• Each order is for a single customer	$\{\text{Order}\} \rightarrow \{\text{Customer}\}$
• Each customer has a single address	$\{\text{Customer}\} \rightarrow \{\text{Address}\}$
• Each product has a single price	$\{\text{Product}\} \rightarrow \{\text{UnitPrice}\}$
• From FDs 1 and 2 and transitivity	$\{\text{Order}\} \rightarrow \{\text{Address}\}$

Normalisation to 2NF

- Second normal form means no partial dependencies on candidate keys
  - {Order} → {Customer, Address}
  - {Product} → {UnitPrice}

- To remove the first FD we project over
  {Order, Customer, Address} (R1)

and

{Order, Product, Quantity, UnitPrice} (R2)

Normalisation to 2NF

- R1 is now in 2NF, but there is still a partial FD in R2

  {Product} $\rightarrow$ {UnitPrice}

- To remove this we project over

{Product, UnitPrice} (R3)

and

{Order, Product, Quantity} (R4)

Normalisation to 3NF

- R has now been split into 3 relations - R1, R3, and R4
  - R3 and R4 are in 3NF
  - R1 has a transitive FD on its key
- To remove

  $\{\text{Order}\} \rightarrow \{\text{Customer}\} \rightarrow \{\text{Address}\}$
- we project R1 over
  - {Order, Customer}
  - {Customer, Address}

Normalisation

- 1NF:
  - {Order, Product, Customer, Address, Quantity, UnitPrice}
- 2NF:
  - {Order, Customer, Address}, {Product, UnitPrice}, and {Order, Product, Quantity}
- 3NF:
  - {Product, UnitPrice}, {Order, Product, Quantity}, {Order, Customer}, and {Customer, Address}

The Stream Relation

- Consider a relation, Stream, which stores information about times for various streams of courses
- For example: labs for first years
- Each course has several streams
- Only one stream (of any course at all) takes place at any given time
- Each student taking a course is assigned to a single stream for it

The Stream Relation

Student	Course	Time
John	Databases	12:00
Mary	Databases	12:00
Richard	Databases	15:00
Richard	Programming	10:00
Mary	Programming	10:00
Rebecca	Programming	13:00

Candidate keys: {Student, Course} and {Student, Time}

FDs in the Stream Relation

- Stream has the following non-trivial FDs
- {Student, Course} $\rightarrow$ {Time}
- {Time} $\rightarrow$ {Course}
- Since all attributes are key attributes, Stream is in 3NF

Anomalies in Stream

- INSERT anomalies
  - You can’t add an empty stream
- UPDATE anomalies
  - Moving the 12:00 class to 9:00 means changing two rows
- DELETE anomalies
  - Deleting Rebecca removes a stream

Student	Course	Time
John	Databases	12:00
Mary	Databases	12:00
Richard	Databases	15:00
Richard	Programming	10:00
Mary	Programming	10:00
Rebecca	Programming	13:00

Boyce-Codd Normal Form

- A relation is in Boyce-Codd normal form (BCNF) if for every FD $A \rightarrow B$ either
  - B is contained in A (the FD is trivial), or
  - A contains a candidate key of the relation,
- In other words: every determinant in a non-trivial dependency is a (super) key.
- The same as 3NF except in 3NF we only worry about non-key Bs
- If there is only one candidate key then 3NF and BCNF are the same

Stream and BCNF

- Stream is not in BCNF as the FD $\{\text{Time}\} \rightarrow \{\text{Course}\}$ is non-trivial and $\{\text{Time}\}$ does not contain a candidate key

Student	Course	Time
John	Databases	12:00
Mary	Databases	12:00
Richard	Databases	15:00
Richard	Programming	10:00
Mary	Programming	10:00
Rebecca	Programming	13:00

Conversion to BCNF

Student Course Time

Student	Course

Course	Time

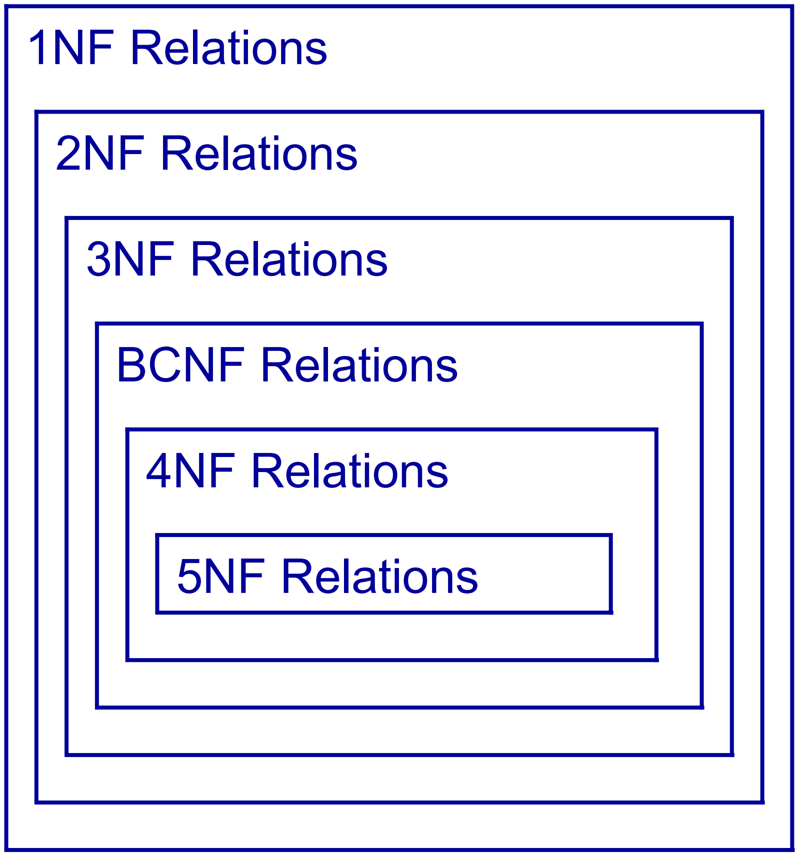
Stream has been put into BCNF but we have lost the FD
{Student, Course} → {Time}

Decomposition Properties

- Lossless: Data should not be lost or created when splitting relations up
- Dependency preservation: It is desirable that FDs are preserved when splitting relations up
- Normalisation to 3NF is always lossless and dependency preserving
- Normalisation to BCNF is lossless, but may not preserve all dependencies

Higher Normal Forms

- BCNF is as far as we can go with FDs
  - Higher normal forms are based on other sorts of dependency
  - Fourth normal form removes multi-valued dependencies
  - Fifth normal form removes join dependencies



Denormalisation

- Normalisation
  - Removes data redundancy
  - Solves INSERT, UPDATE, and DELETE anomalies
  - This makes it easier to maintain the information in the database in a consistent state
- However
  - It leads to more tables in the database
  - Often these need to be joined back together, which is expensive to do
  - So sometimes (not often) it is worth 'denormalising'

Denormalisation

- You *might* want to denormalise if
  - Database speeds are unacceptable (not just a bit slow)
  - There are going to be very few INSERTs, UPDATEs, or DELETEs
  - There are going to be lots of SELECTs that involve the joining of tables

Address

Number	Street	City	Postcode

Not normalised since

$\{\text{Postcode}\} \rightarrow \{\text{City}\}$

Address1

Number	Street	Postcode

Address2

Postcode	City