NORMALIZATION TO BCNF

Introduction to Database Systems

Mohammad Tanhaei
Ilam University

IN THIS LECTURE

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

NORMALIZATION 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 normalize a relation, we used projections
- If R(A,B,C) satisfies A→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:

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

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

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

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

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

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

NORMALIZATION 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

  {Order} → {Customer}

- Each customer has a single address

  {Customer} → {Address}

- Each product has a single price

  {Product} → {UnitPrice}

- From FDs 1 and 2 and transitivity

  {Order} → {Address}

R1(<u>Order</u>, <u>Product</u>, Customer, Address, Quantity, UnitPrice)

NORMALIZATION 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

{<u>Order</u>, Customer, Address} (R1)

and

{<u>Order</u>, <u>Product</u>, Quantity, UnitPrice} (R2)

NORMALIZATION TO 2NF

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

  {Product} → {UnitPrice}

- To remove this we project over

  {Product, UnitPrice} (R3)

  and

  {Order, Product, Quantity} (R4)

NORMALIZATION 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

  {Order} → {Customer} → {Address}
- We project R1 over
  - {Order, Customer}
  - {Customer, Address}

NORMALIZATION

- 1NF:
  - {<u>Order, Product</u>, Customer, Address, Quantity, UnitPrice}
- 2NF:
  - {<u>Order</u>, Customer, Address},
  - {<u>Product</u>, UnitPrice},
  - {<u>Order, Product</u>, Quantity}
- 3NF:
  - {<u>Product</u>, UnitPrice},
  - {<u>Order, Product</u>, Quantity},
  - {<u>Order</u>, Customer},
  - {<u>Customer</u>, Address}

کوییز

➤ اگر در رابطه‌ی R1 و R2 و R3 مجموعه FD به صورت زیر باشد، آنگاه رابطه در چه سطحی قرار دارد؟ آیا می‌توانید رابطه را به سطوح بالاتر نرمال ببرید؟

$FD=\{X \rightarrow W,\ X \rightarrow Z,\ W \rightarrow Y\}$

$R1(\underline{X}, \underline{Y}, Z, \underline{W})$

$R2(\underline{X}, Y, Z, \underline{W})$

$R3(\underline{X}, Y, Z, W)$

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} → {Time}

{Time} → {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

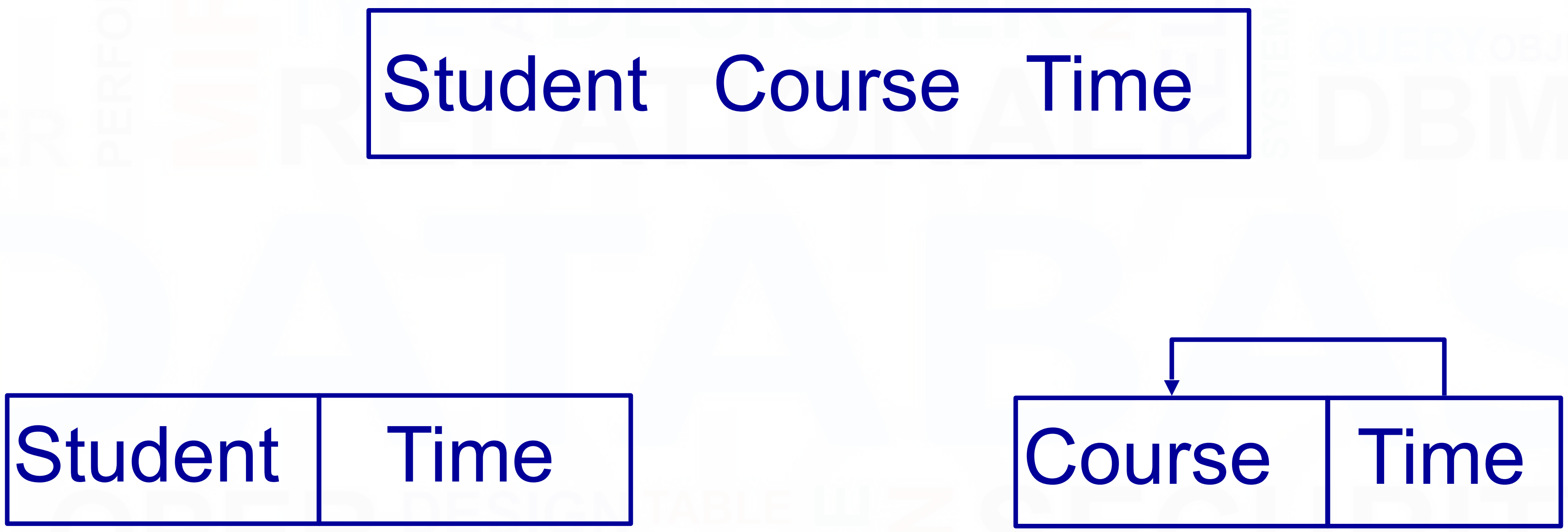
- A relation is in **B**oyce-**C**odd **N**ormal **F**orm (BCNF) if for every FD A → B either
  - B is contained in A (the FD is trivial), or
  - A contains a **candidate key** of the relation,
- In other words: every determinant in a non-trivial dependency is a (super) key.
- The same as 3NF except 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 {Time} → {Course} is non-trivial and {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



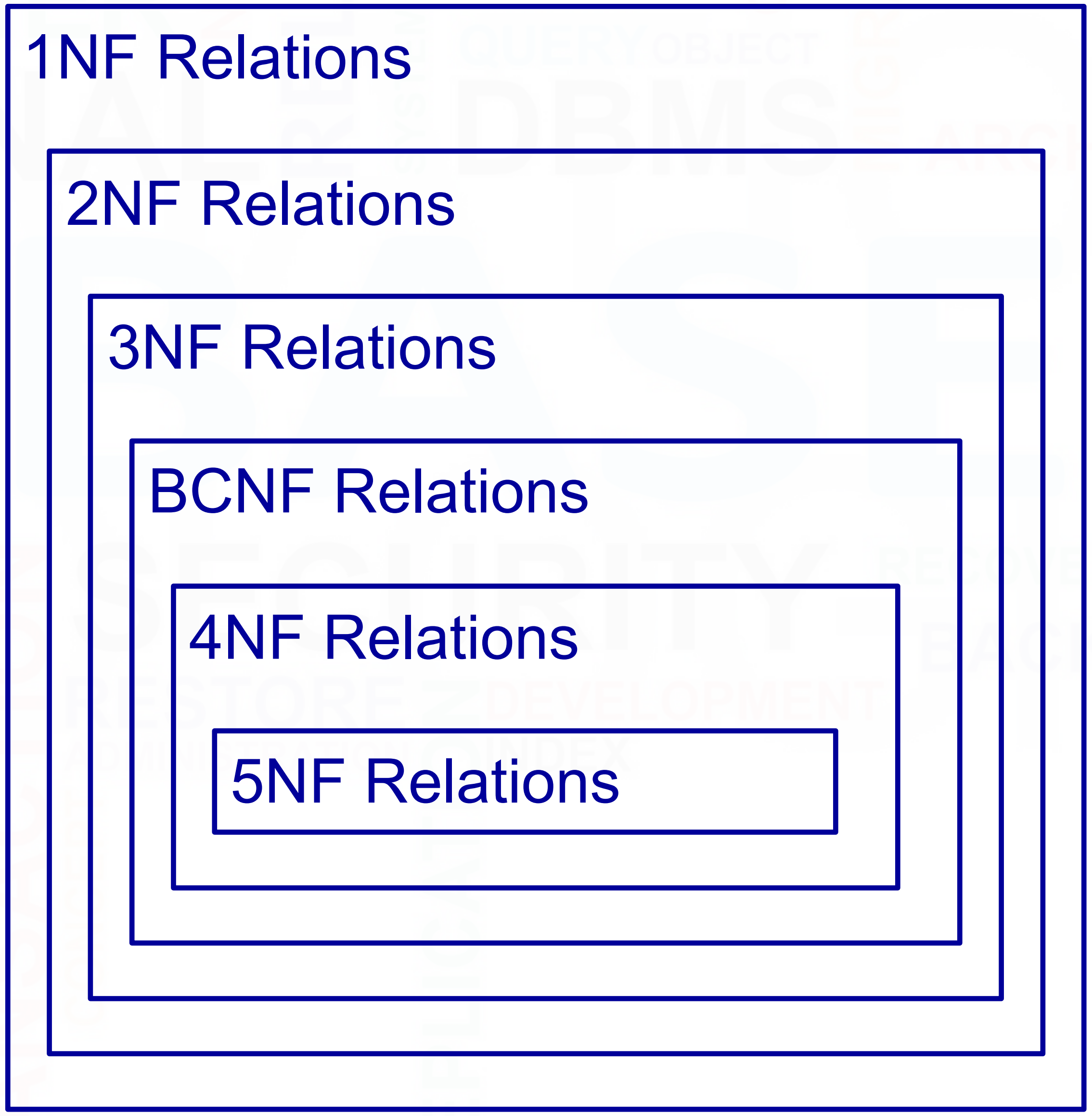
Stream has been put into BCNF but we have lost the FD
{Student, Course} → {Time}

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
- Normalization to 3NF is always lossless and dependency preserving
- Normalization 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



DENORMALIZATION

- A Normalization
  - 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 'denormalizing'

DENORMALIZATION

- You *might* want to denormalize 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 normalized since
{Postcode} → {City}

Address1

Number	Street	Postcode

Address2

Postcode	City

NORMALIZATION IN EXAMS

- Given a relation with scheme {ID, Name, Address, Postcode, CardType, CardNumber}, the candidate key {ID}, and the following functional dependencies:

{ID} → {Name, Address, Postcode, CardType, CardNumber}

{Address} → {Postcode}

{CardNumber} → {CardType}

- (i) Explain why this relation is in second normal form, but not in third normal form.

(3 marks)

NORMALIZATION IN EXAMS

- (ii) Show how this relation can be converted to third normal form. You should show what functional dependencies are being removed, explain why they need to be removed, and give the relation(s) that result.

(4 marks)

- (iii) Give an example of a relation that is in third normal form, but that is not in Boyce-Codd normal form, and explain why it is in third, but not Boyce-Codd, normal form.

(4 marks)

END

But take a look at next slide!

NEXT LECTURE

- Physical DB Issues
  - RAID arrays for recovery and speed
  - Indexes and query efficiency
- Query optimisation
  - Query trees
- For more information
  - Connolly and Begg chapter 21 and appendix C.5, Ullman and Widom 5.2.8