Boyce-Codd Normal Form

Jianjun Chen

Previous Lecture

- In the previous lecture we demonstrated how 2NF and 3NF disallow partial and transitive dependencies on the primary key of a relation, respectively.

- Relations that have these types of dependencies may suffer from the update anomalies

In this Lecture

- More on functional dependencies
  - Lossless decomposition
- BCNF
  - Boyce-Codd normal form (BCNF)
  - Higher normal forms
  - Denormalisation
- For more information
  - Connolly and Begg Chapter 15

More on FDs

- The set of all functional dependencies that are implied by a given set of functional dependencies X is called the **closure** of X, written as X^+.
  - “implied” -> We need inference rules.
- *Armstrong’s axioms* specifies how new FDs can be inferred from given ones.
  - Reflexivity: If B is a subset of A, then A -> B
  - Augmentation: If A -> B, then A, C -> B, C
  - Transitivity: If A -> B and B -> C, then A -> C

Rules Derived

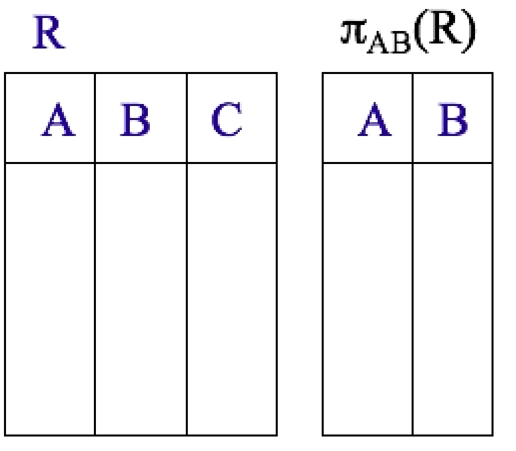
- The following rules can be derived from the three rules given previously:
  - Self-determination A -> A
  - Decomposition If A -> B,C, Then A -> B and A -> C
  - Union If A -> B and A -> C, then A -> B,C
  - Composition If A -> B and C -> D, then A,C -> B,D

Lossless Decomposition

- To normalise a relation, we used projections
- If R(A,B,C) satisfies A→B,C
  - 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.



Example of Lossless Decomposition

Assume that a module is taught by one lecturer

R

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

$\pi_{\text{Module,Lecturer}}(\mathbf{R})$

Module	Lecturer
DBS	rzb
RDB	nza
APS	rcb

$\pi_{\text{Module,Text}}(\mathbf{R})$

Module	Text
DBS	CB
DBS	UW
RDB	UW
APS	B

When is Decomposition not Lossless: No FD

S

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

$\pi_{First,Age} S$

First	Age
John	20
John	30
Mary	20
Tom	10

$\pi_{First,Last} S$

First	Last
John	Smith
John	Brown
Mary	Smith
Tom	Brown

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

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

Question:

- Is the normalisation process from 1NF to 3NF lossless or lossy?

- If we break a FD $A \rightarrow B$ by splitting them into two separate tables {A} and {B}, will it be lossy or lossless?

Boyce-Codd Normal Form

The Stream Relation

- Consider a relation, Stream, which stores information about times for various streams of courses (Online teaching)
- Assume:
  - Each course has several streams
  - Only one stream (of any course at all) takes place at any given time
  - Each (student, course) pair is assigned to a single stream for it. That is, (student, course) is the primary 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

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}

What are the functional dependencies?

FDs in the Stream Relation

- Stream has the following non-trivial FDs

{Student, Course} → {Time}

{Student, Time} → {Course}

{Time} → {Course}

- Trivial FD: (A, B) → B
  - B is a subset of (A, B).
- non-trivial FD: A → B
  - B is not a subset of A
- Is Stream table in 3NF?

Anomalies in Stream

- INSERT anomalies:
  - You can't add an empty stream (a stream with no students)
- Modification 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 **BCNF**, if and only if, every determinant is a candidate key."

- The difference between 3NF and BCNF is that for a functional dependency $A \rightarrow B$, The process of normalising to 3NF does not care whether the determinant A is a candidate key or not.
- whereas BCNF insists that for this dependency to remain in a relation, A must be a candidate key.

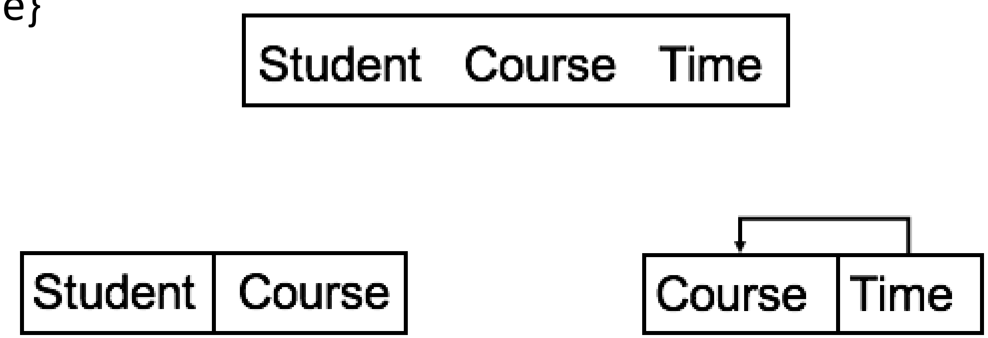
Stream and BCNF

- Stream is not in BCNF as the FD {Time} → {Course} is non-trivial and {Time} is not 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, Time} → {Course}
{Time} → {Course}



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

BCNF & Decomposition Properties

- In relation {A, B, C} with primary key (A, B), we have functional dependency (A, B) → C.

- If we also have FD C → B, <u>and C cannot be considered as a candidate key in the relation</u>, then this relation is in 3NF but not in BCNF. Stream table shows that such a relation has update anomalies.
  - We have FD (A, B) → C → B, which looks like a transitive dependency, but (A, B) → B is a trivial FD.
  - That’s the reason why this table is in 3NF.

BCNF & Decomposition Properties

- To convert {A, B, C} into BCNF, split it into {A, B} and {C, B}
  - But we lose FD (A, B) → C. We can no longer reproduce such relationship.
- As a result, BCNF in this case is lossy.

Decomposition Properties

1. Lossless: Data should not be lost or created when splitting relations up
2. Dependency preservation: It is desirable that FDs are preserved when splitting relations up

- Normalisation to BCNF may not preserve all dependencies.
  - As a result, BCNF is NOT always necessary.
  - Depends on whether anomalies are acceptable or not.

Another Example

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

- What if (student, time) is the primary key?

FD1: {Student, Course} → {Time}

FD2: {Student, Time} → {Course}

FD3: {Time} → {Course}

- In FD2, Course is partially dependent on the primary key. The table is not in 2NF.
- Splitting the table to {Time, Course} and {Student, Time} results in a lossy decomposition.
  - FD {Student, Course} → {Time} is broken
  - You can check the result by yourself.
- Normalisation to 3NF is NOT always lossless and dependency preserving

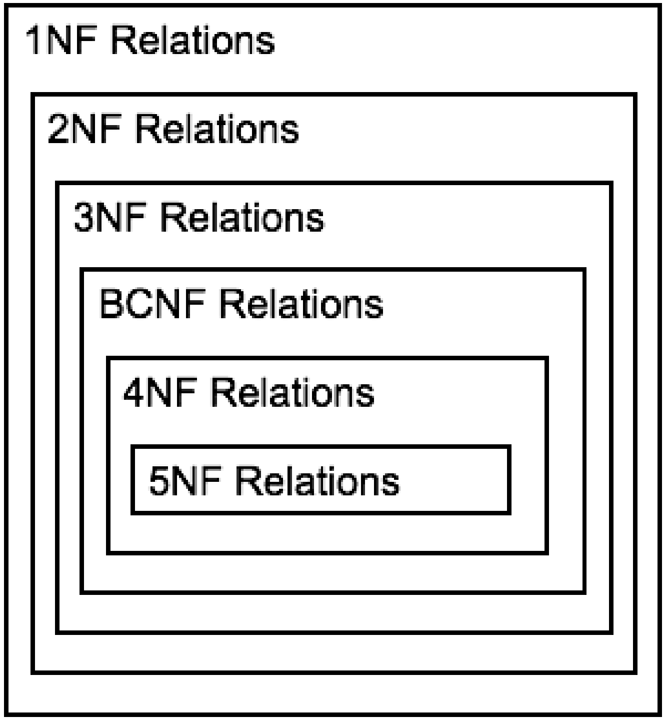
Higher Normal Forms

- Violation of BCNF is quite rare.

- The potential to violate BCNF may occur in a relation that:
  - contains two (or more) composite candidate keys;
  - the candidate keys overlap, that is have at least one attribute in common.

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

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