UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

CS411 - Normal Forms

illinois.edu

Announcements

- MP1 Due Feb 15
- Project Stage 1 due Feb 17
- Midterm 1 March 1st

Review

- What is a functional dependency?
- What are some examples?
- What is a key?
- What is a superkey?
- What rules did we learn about manipulating FDs?

Big picture

- Today
  - Finish FD discussion
  - Learn how FDs help us improve relations by decomposing them
  - Iteratively improve our database schema
- Next week
  - We'll learn how to design database schemas

Functional Dependency

- **IF** a tuple agrees on the *values* of $A_1, A_2, \ldots, A_n$
- **THEN** they must agree on the *values* of $B_1, B_2, \ldots B_m$

$$A_1, A_2, \ldots, A_n \rightarrow B_1, B_2, \ldots, B_m$$

Keys

- A key is a set of attributes $\{A_1,A_2,\ldots,A_n\}$ for a relation R such that:
    1. $\{A_1,A_2,\ldots,A_n\}$ functionally determine all other attributes of R
    2. No subset of $\{A_1,A_2,\ldots,A_n\}$ functionally determines all other attributes of R

Superkeys

- A set of attributes that contains a key is called a *superkey*
  - "superset of a key"

illinois.edu

All the rules

1. Reflexivity
2. Augmentation
3. Transitivity
4. Union
5. Decomposition
6. Pseudotransitivity

illinois.edu

Attribute Closure

- Given a set of attributes $\{A_1,A_2,\ldots,A_n\}$ and S is a set of FDs
- Find all attributes B such that $\{A_1,A_2,\ldots,A_n\} \rightarrow B$
- We call the set of all such B's the *closure* of $\{A_1,A_2,\ldots,A_n\}$
- We use the notation $\{A_1,A_2,\ldots,A_n\}^+$

Attribute Closure Algorithm

- INPUT:
  - a set of attributes $\{A_1,A_2,\ldots,A_n\}$
  - a set of FDs S
- OUTPUT:
  - The attribute closure $\{A_1,A_2,\ldots,A_n\}^+$

Attribute Closure Algorithm

1. Use decomposition (Rule 5) to split all FDs in S so the right hand side has one attribute
2. Initialize $X=\{A_1,A_2,\ldots,A_n\}$
3. Loop
    1. Find C such that C is not in X, $B_1,B_2,\ldots,B_m$ are in X, and $B_1,B_2,\ldots,B_m \rightarrow C$
    2. If C exists, add it to X. If not, break.
4. Return X

Example

R(A,B,C,D,E,F)

$F=\{AB \rightarrow C, BC \rightarrow AD, D \rightarrow E, CF \rightarrow B\}$

Find $\{A,B\}^+$

Example

R(A,B,C,D,E,F)

$F=\{AB\rightarrow C, BC\rightarrow AD, D\rightarrow E, CF\rightarrow B\}$

Find $\{A,B\}^+$

1. Split up F using decomposition:
   $F=\{AB\rightarrow C, BC\rightarrow A, BC\rightarrow D, D\rightarrow E, CF\rightarrow B\}$

Example

R(A,B,C,D,E,F)

$F=\{AB\rightarrow C, BC\rightarrow AD, D\rightarrow E, CF\rightarrow B\}$

Find $\{A,B\}^+$

1. Split up F using decomposition:
   $F=\{AB\rightarrow C, BC\rightarrow A, BC\rightarrow D, D\rightarrow E, CF\rightarrow B\}$
2. Initialize $X=\{AB\}$

Example

$R(A,B,C,D,E,F)$

$F=\{AB \rightarrow C, BC \rightarrow A, BC \rightarrow D, D \rightarrow E, CF \rightarrow B\}$

Find $\{A,B\}^+$

3. $X=\{A,B\}$

Find new attributes to add to X

Example

R(A,B,C,D,E,F)

F={AB→C,BC→A, BC→D,D→E,CF→B}

Find $\{A,B\}^+$

3. X={A,B,C}

Find new attribute to add to X

Example

R(A,B,C,D,E,F)

$F=\{AB\rightarrow C, BC\rightarrow A, BC\rightarrow D, D\rightarrow E, CF\rightarrow B\}$

Find $\{A,B\}^+$

3. $X=\{A,B,C,D\}$

Find new attribute to add to X

Example

R(A,B,C,D,E,F)

F={AB→C,BC→A, BC→D,D→E,CF→B}

Find $\{A,B\}^+$

3. X={A,B,C,D,E}
Find new attribute to add to X

Example

R(A,B,C,D,E,F)

$F=\{AB\rightarrow C, BC\rightarrow A, BC\rightarrow D, D\rightarrow E, CF\rightarrow B\}$

Find $\{A,B\}^+$

3. $X=\{A,B,C,D,E\}$

Nothing new to add, so

4. Return $X=\{A,B,C,D,E\}$

Uses of attribute closure

- Text if X is a superkey
  - Check if all attributes are in X^+
- Check if some FD (e.g. $A \rightarrow B$) holds
  - Check if Y is in X^+
  - Easier than applying all the rules!

Example

R(A,B,C,D,E,F,G,H,I,J)

F={AB → E, AG → J, BE → I, E → G, GI → H}

Does AB → GH?

$\{A,B\}^+$={A,B,E,I,G,H,J}

FD Closure

- Discovering ALL of the FDs that follow from a given set of FDs
- We annotate it S^+

FD Closure Algorithm

- INPUT:
  - A set of FDs S
- Output:
  - The FD closure S^+

FD Closure Algorithm

1. Initialize a set X=S
2. For each FD f in S, apply reflexivity and augmentation
3. Add new FDs to X
4. For each pair of FDs in S, apply transitivity
5. Add new FDs to X
6. If X hasn't been updated, return X
7. Else GOTO 2

Example

R=(A,B,C)

F={AB→ C, C→ B}

F^+={A→ A, AB→ A, AC→ A, AB→ B, BC→ B, ABC→ B, C→ C, AC→ C, BC→ C, ABC→ C, AB→ AB, ...AB→ ABC, C→ B, C→ BC, AC→ B, AC→ AB}

Closures

- What's the difference between ***attribute*** closure and ***functional dependency*** closure?

Reminder

- Two sets of FDs S and T are ***equivalent*** if the set of relations satisfying S is the same as the set of relations satisfying T
- Another way of saying this is that S follows from T and T follows from S
  - Example: S^+ follows from S

More vocabulary

- If we are given S, all equivalent sets of FDs T is called a *basis* for S
- A *minimal basis* satisfies
    1. All the FDs have singleton right sides
    2. If any FD is removed, it is no longer a basis
    3. If we remove one or more attributes from the left side of a FD, it is no longer a basis

Example

$R(A,B,C)$

$F=\{A \rightarrow B, A \rightarrow C, B \rightarrow A, B \rightarrow C, C \rightarrow A, C \rightarrow B\}$

$B_1=\{A \rightarrow B,\ B \rightarrow A,\ B \rightarrow C,\ C \rightarrow B\}$

$B_2=\{A \rightarrow B,\ B \rightarrow C,\ C \rightarrow A\}$

Projecting FDs

- Say we take the projection of a relation R with FDs F
- $S=\pi_L(R)$
- What FDs hold for S?
  - For each subset A_i of L, compute A_i^+
  - Throw out attributes not in S
  - Union them all together

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

Find G (FDs for S)

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

$\{A\}^{+}=\{A,B,C,D\}$

$G=\{A\rightarrow C, A\rightarrow D\}$

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

$\{C\}^{+}=\{C,D\}$

$G=\{A \rightarrow C,\ A \rightarrow D,\ C \rightarrow D\}$

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

$\{D\}^{+}=\{D\}$

$G=\{A\rightarrow C,\ A\rightarrow D,\ C\rightarrow D\}$

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

$\{A,C\}^{+}=\{A,B,C,D\}$

$G=\{A \rightarrow C,\ A \rightarrow D,\ C \rightarrow D\}$

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

Skipping some uninteresting steps…

Example

$R(A,B,C,D)$

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

$S=\pi_{A,C,D}(R)$

$\{C,D\}^{+}=\{D\}$

$G=\{A\rightarrow C,\ A\rightarrow D,\ C\rightarrow D\}$

illinois.edu

Moving on...

- We have a formal model of interdependence of data in a relation
- We want to iteratively improve our relations
  - decompose relations
  - eliminate anomalies

Anomalies

1. Redundancy - repeated values
2. Update anomalies - changes to related values aren't propagated
3. Deletion anomalies - deletion of some values causes a loss of unrelated data

Example - Redundancy

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone with the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Example - Update Anomaly

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	125	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
Gone with the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne’s World	1992	95	comedy	Paramount	Dana Carvey
Wayne’s World	1992	95	comedy	Paramount	Mike Meyers

Example - Deletion Anomaly

title	year	length	genre	studioName	starName
Star Wars	1977	124	SciFi	Fox	Carrie Fisher
Star Wars	1977	124	SciFi	Fox	Mark Hamill
Star Wars	1977	124	SciFi	Fox	Harrison Ford
~~Gone with the Wind~~	~~1939~~	~~231~~	~~drama~~	~~MGM~~	~~Vivien Leigh~~
Wayne's World	1992	95	comedy	Paramount	Dana Carvey
Wayne's World	1992	95	comedy	Paramount	Mike Meyers

Normal Forms

- A formal theory for evaluating a relation’s schema
- The “higher” the form, the more theoretically robust the relation

Normal Forms

- First Normal Form (1NF)
- Second Normal Form (2NF)
- Third Normal Form (3NF)
- Boyce Codd Normal Form (BCNF)
- Fourth Normal Form (4NF)

Normal Forms

- ~~First Normal Form (1NF)~~ trivial
- ~~Second Normal Form (2NF)~~ obsolete
- Third Normal Form (3NF)
- Boyce Codd Normal Form (BCNF)
- Fourth Normal Form (4NF)

BCNF (formally)

- A relation R is in BCNF if:
  - For any nontrivial FD $\{A_1, A_2, \ldots, A_n\} \rightarrow B$, $\{A_1, A_2, \ldots, A_n\}$ is a superkey

BCNF (informally)

- If a set of attributes determines ***some*** of the other attributes, then it determines ***all*** of the attributes

Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade,TA)

$$\underline{section} \rightarrow TA$$

$$student, courseNumber \rightarrow grade, section$$

$$\underline{courseNumber} \rightarrow roomNumber, instructorName$$

illinois.edu

mber,roomNumber,

uctor lentName,section,

le,TA)

$$secti$$

$$nt, courseNumber \quad s$$

$$cou \rightarrow roomNumber \quad ame$$

Example

Person(ssn,name,birthdate,address)

$$ssn \rightarrow name, birthdate, address$$

ame,birthdate,add

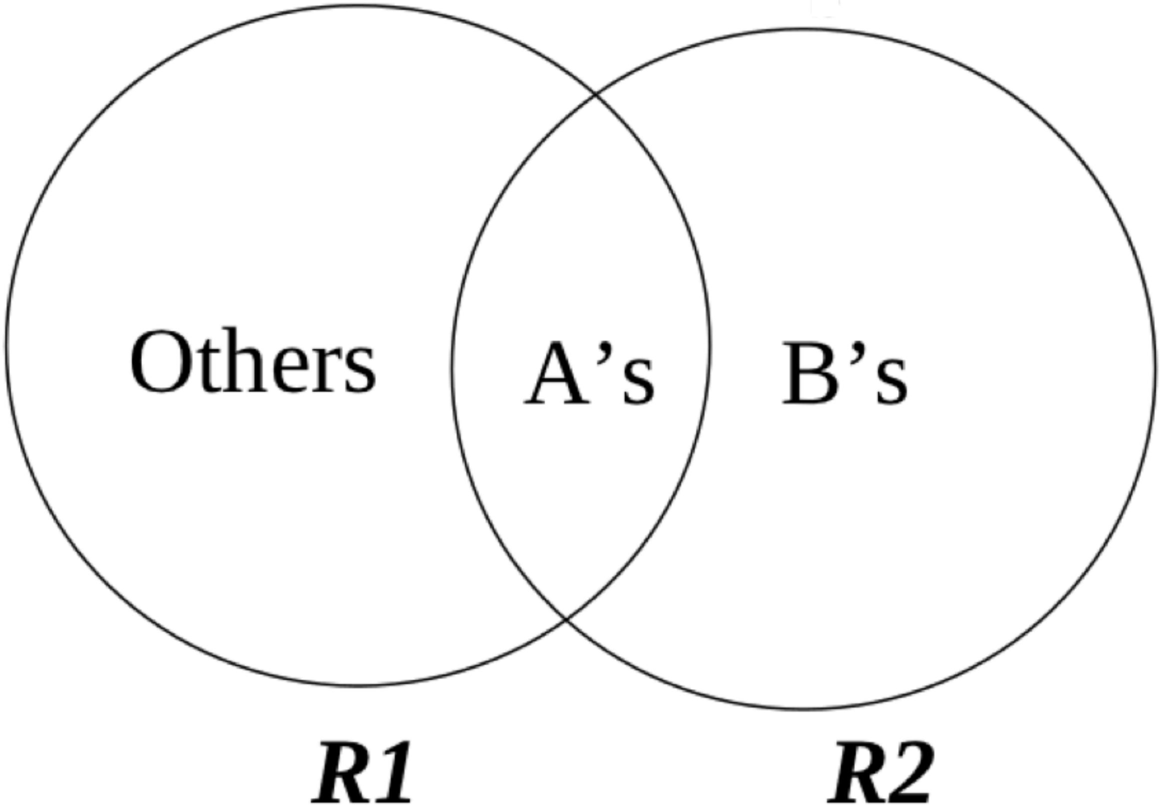
$$n \rightarrow name, birthdate, addre$$

BCNF Decomposition

- Find a dependency that violates BCNF:
  - $\{A_1,A_2,\ldots,A_n\} \rightarrow \{B_1,B_2,\ldots,B_m\}$ is a FD, but $\{A_1,A_2,\ldots,A_n\}$ is not a superkey
  - Ideally, find the biggest $\{B_1,B_2,\ldots,B_m\}$ we can
  - Decompose the relation into two new relations
  - Repeat until none of the resulting relations violate BCNF

BCNF Decomposition

Decompose:



Example

Class(courseNumber,roomNumber, instructorName,studentName,section, grade,TA)

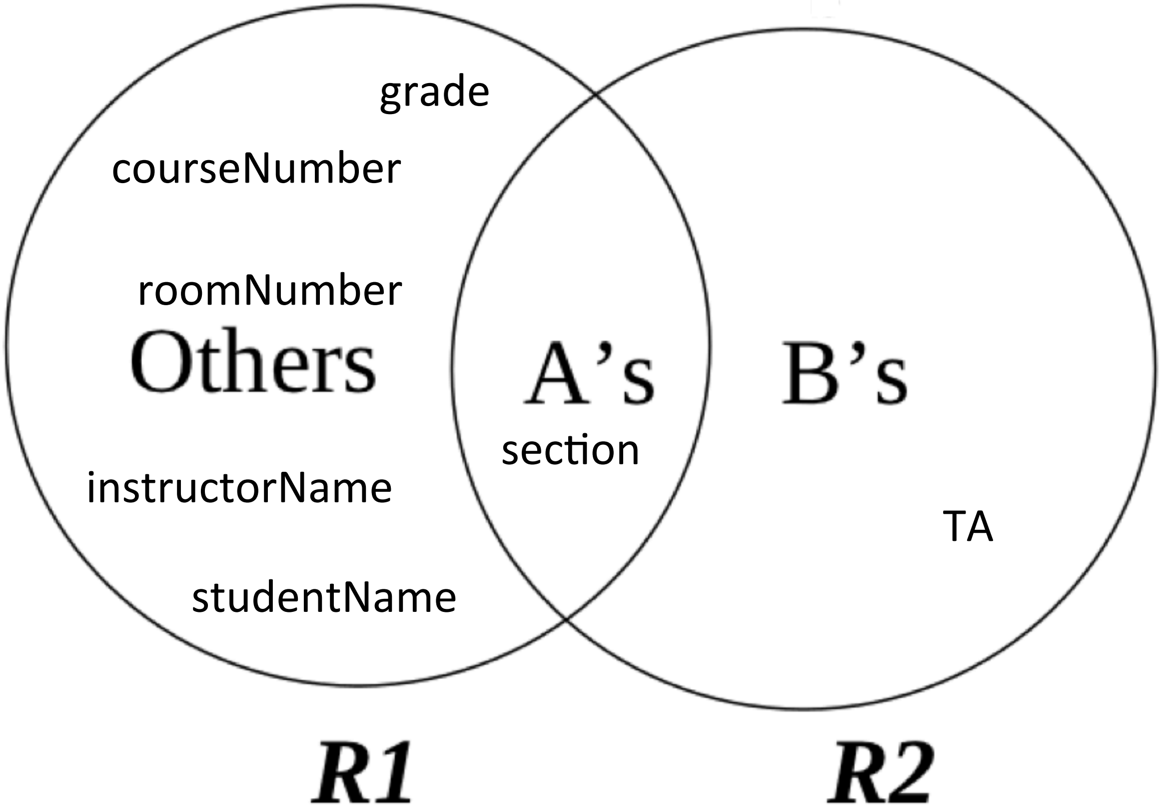
$$\underline{section \rightarrow TA}$$

$$student, courseNumber \rightarrow grade, section$$

$$courseNumber \rightarrow roomNumber, instructorName$$

Example

Decompose:



illinois.edu

Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade)

Section(section,TA)

BCNF Decomposition Algorithm

- INPUT:
  - an input relation R with attributes α
  - a set of FDs S
- OUTPUT:
  - a set of relations in BCNF

BCNF Decomposition Algorithm

1. Compute S^+
2. Identify all keys for R
3. Check if R is in BCNF
   a) identify a violating FD $\{A_1,A_2,\ldots,A_n\} \rightarrow \{B_1,B_2,\ldots,B_m\}$
   b) Compute $\{A_1,A_2,\ldots,A_n\}^+$
   c) Create $R_1=\{A_1,A_2,\ldots,A_n\}^+$ and $R_2=\{A_1,A_2,\ldots,A_n\} \cup (\alpha-\{A_1,A_2,\ldots,A_n\}^+)$
   d) Recurse on R_1 and R_2

Example

Class(courseNumber,roomNumber, instructorName,studentName,section, grade,TA)

$$section \rightarrow TA$$

$$student, courseNumber \rightarrow grade, section$$

$$courseNumber \rightarrow roomNumber, instructorName$$

Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade,TA)

Keys: student,courseNumber

Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade,TA)

Keys: student,courseNumber
Violating FD: section→TA

Example

Class(courseNumber,roomNumber, instructorName,studentName,section, grade,TA)

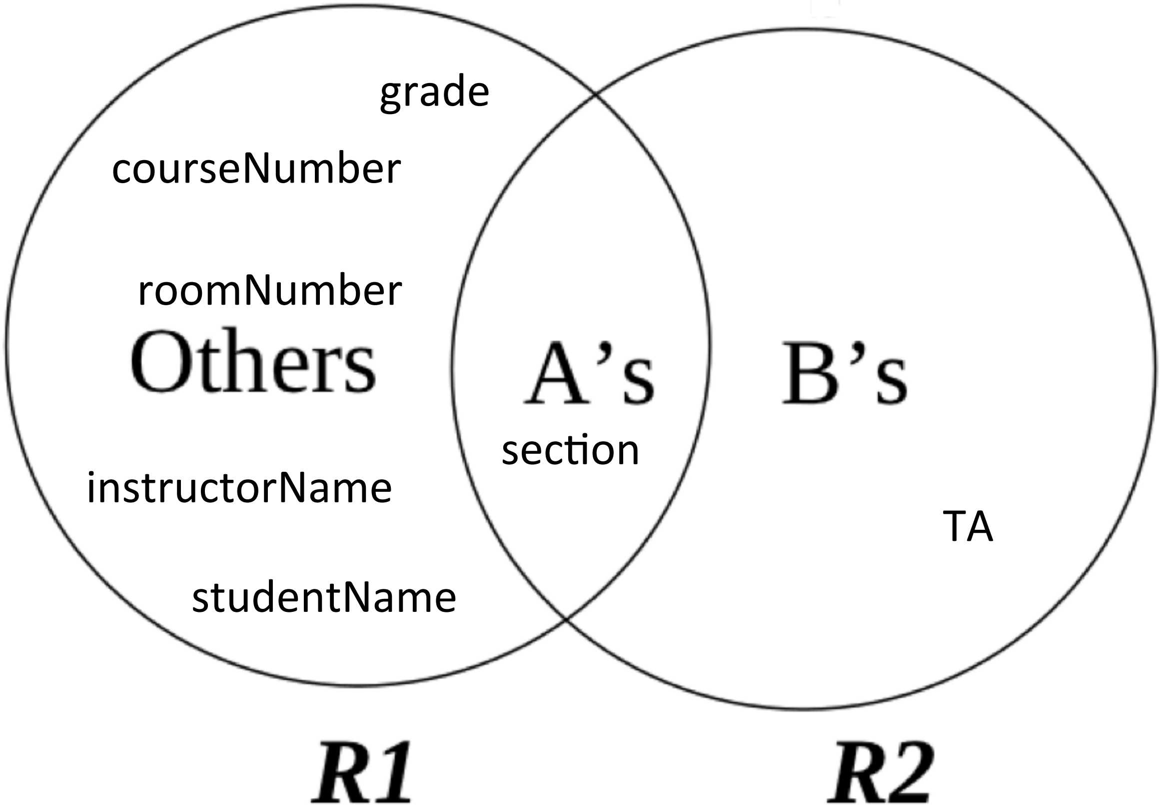
Keys: student,courseNumber

Violating FD: section→TA

$\{section\}^+=\{section,TA\}$

Example

Decompose:



Example

Class(courseNumber,roomNumber, instructorName,studentName,section, grade)

Section(section,TA)

Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade) ← let's work on this relation

Section(section,TA) ← already in BCNF

Example

Class(courseNumber,roomNumber, instructorName,studentName,section, grade)

$$student, courseNumber \rightarrow grade, section$$

$$courseNumber \rightarrow roomNumber, instructorName$$

illinois.edu

Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade)

Keys: student,courseNumber

Example

Class(courseNumber,roomNumber, instructorName,studentName,section, grade)

Violating FD:

courseNumber→room,instructorName

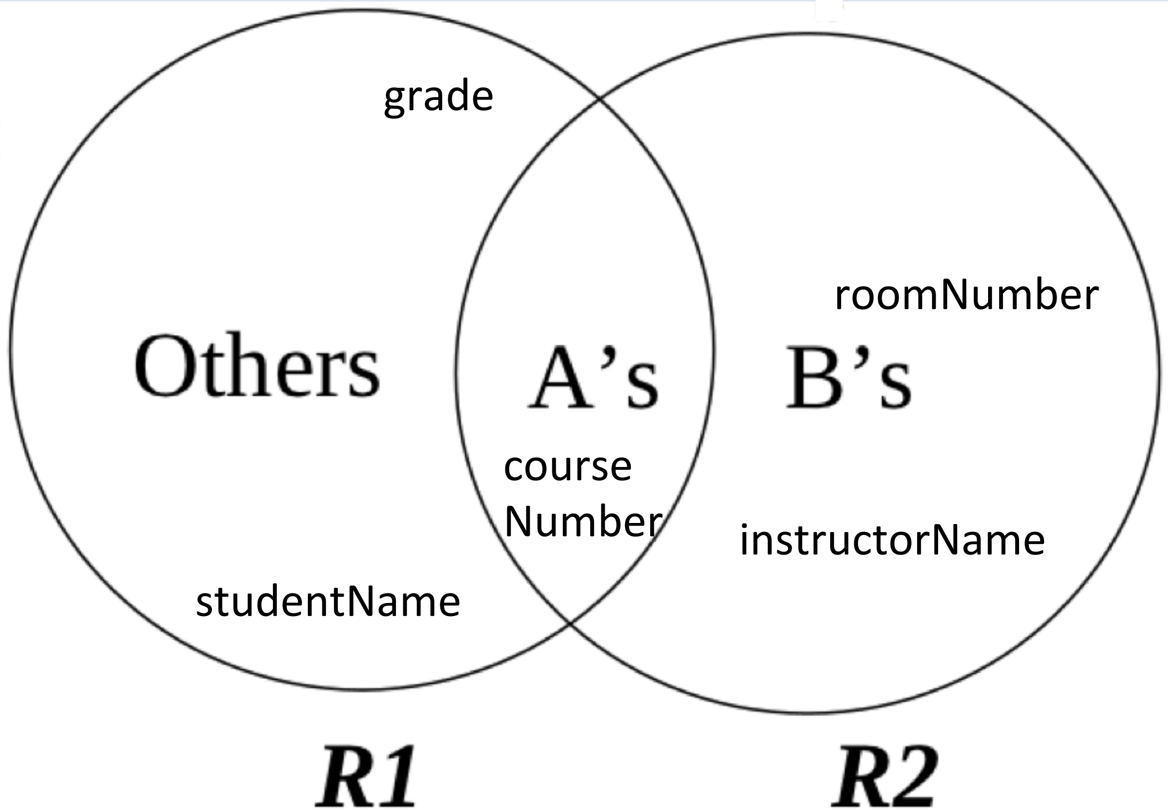
Example

Class(courseNumber,roomNumber,
instructorName,studentName,section,
grade)

$\{courseNumber\}^+$
$=\{courseNumber,room,instructorName\}$

Example

Decompose:



Example

Course(courseNumber,roomNumber,
instructorName)
Record(studentName,section,grade)
Section(section,TA)

BCNF Properties

- Eliminates anomalies ☺
- Not unique 😐
- Always exists ☺
- Information is recoverable ☺
- Does *not* preserve dependencies ☹

Recovering Information

- After decomposition, we can always recover the original relation
- How?

Recovering Information

- After decomposition, we can always recover the original relation
- How?

Example

Course(courseNumber,roomNumber, instructorName)

Record(studentName,section,grade)

Section(section,TA)

Course $\bowtie$ Record $\bowtie$ Section = Class

Dependency Preservation

- Are FDs preserved by decomposition?
  - NOT in BCNF

Example

Concert(band,venue,city)

venue→city

band,city→venue

Example

Concert(band,venue,city)

venue→city

band,city→venue

Keys: {band,city},{venue,band}

Example

Concert(band,venue,city)

venue→city ← VIOLATION OF BCNF

band,city→venue

Keys: {band,city},{venue,band}

Example

Using BCNF Decomposition:

Concert1(venue,city)

Concert2(venue,band)

Example

Using BCNF Decomposition:

Concert1(venue,city)

Concert2(venue,band)

band,city→venue ← Uh oh...

illinois.edu

Example

venue	city
Assembly Hall	Champaign
High Dive	Champaign

venue	band
High Dive	Death Cab
Assembly Hall	Death Cab

venue	city	band
Assembly Hall	Champaign	Death Cab
High Dive	Champaign	Death Cab

band,city→venue

3NF

- Third normal form relaxes BCNF to preserve FDs

3NF (formally)

- A relation R is in BCNF if:
  - For any nontrivial FD $\{A_1,A_2,\ldots,A_n\} \rightarrow B$, $\{A_1,A_2,\ldots,A_n\}$ is a superkey **or B is part of a key**

3NF (informally)

- Everything must depend on the key ***or be part of the key***

3NF Properties

- Doesn't eliminate all anomalies ☹
- Not unique 😐
- Always exists ☺
- Information is recoverable ☺
- Preserve dependencies ☺

Design principle

- Aim for BCNF, settle for 3NF