Normalization

CIS 331: Introduction to Database Systems

Normalization: Reminder

- Why do we need to normalize?
  - To avoid redundancy (less storage space needed, and data is consistent)
  - To avoid update/delete anomalies

Normalization: Reminder

Ssn	c-id	Grade	Name	Address
123	cs331	A	smith	Main
123	cs351	B	smith	Main
123	cs211	A	smith	Main

- **Clearly, Name and Address are redundant (larger relation + you have to update 3 rows to update the Address)**

Normalization: Reminder

Ssn	c-id	Grade	Name	Address
123	cs331	A	smith	Main
…	…	…	…	…
234	null	null	jones	Forbes

- **Insertion anomaly**:

  Cannot make a record Jones' address because he is not taking any classes

Normalization: Reminder

Ssn	c-id	Grade	Name	Address
123	cs331	A	jones	123 Main
124	cs351	B	smith	124 Broad
125	cs211	A	shmoo	42 Penn

- **Delete anomaly**:

  Cannot delete Shmoo’s enrolment without loosing his address as well

Normal Forms

- First Normal Form – 1NF
- Second Normal Form – 2NF
- Third Normal Form – 3NF
- Fourth Normal Form – 4NF
- Fifth Normal Form – 5NF

  (so far conveniently named)
- Boyce-Codd Normal Form – BCNF

First Normal Form (1NF)

- **1NF: all attributes are atomic ("no repeating groups")**

Last Name	First Name
Smith	Peter Mary John
Rumpelstiltskin	Anne Michael

Not in 1NF

First Normal Form (1NF)

Last Name	First Name
Smith	Peter
Smith	Mary
Smith	John
Rumpelstiltskin	Anne
Rumpelstiltskin	Michael

Normalized to 1NF

First Normal Form (1NF)

Name	Weight
Michael	187 lb
Raphael	192 lb
Gabriel	201 lb
Uriel	165 lb
Metatron	195 kg

Not in 1NF

First Normal Form (1NF)

Name	Weight	Unit
Michael	187	lb
Raphael	192	lb
Gabriel	201	lb
Uriel	165	lb
Metatron	195	kg

Normalized to 1NF

First Normal Form (1NF)

Supplier	Part
S1	P1, P3, P4
S2	P3
S3	P2, P3

Not in 1NF

First Normal Form (1NF)

Is this relation in 1NF?

Supplier	Part1	Part2	Part3
S1	P1	P3	P4
S2	P3	null	null
S3	P2	P3	null

Formally yes, but in essence, NO!

First Normal Form (1NF)

Supplier	Part
S1	P1
S1	P3
S1	P4
S2	P3
S3	P2
S3	P3

Normalized to 1NF

Second Normal Form (2NF)

- **2NF:**
  - **1NF and**
  - **all non-key attributes are fully dependent on the PK ("no partial dependencies")**

Student	Course_ID	Grade	Address
Erik	CIS331	A	80 Ericsson Av.
Sven	CIS331	B	12 Olafson St.
Inge	CIS331	C	192 Odin Blvd.
Hildur	CIS362	A	212 Reykjavik St.

Not in 2NF

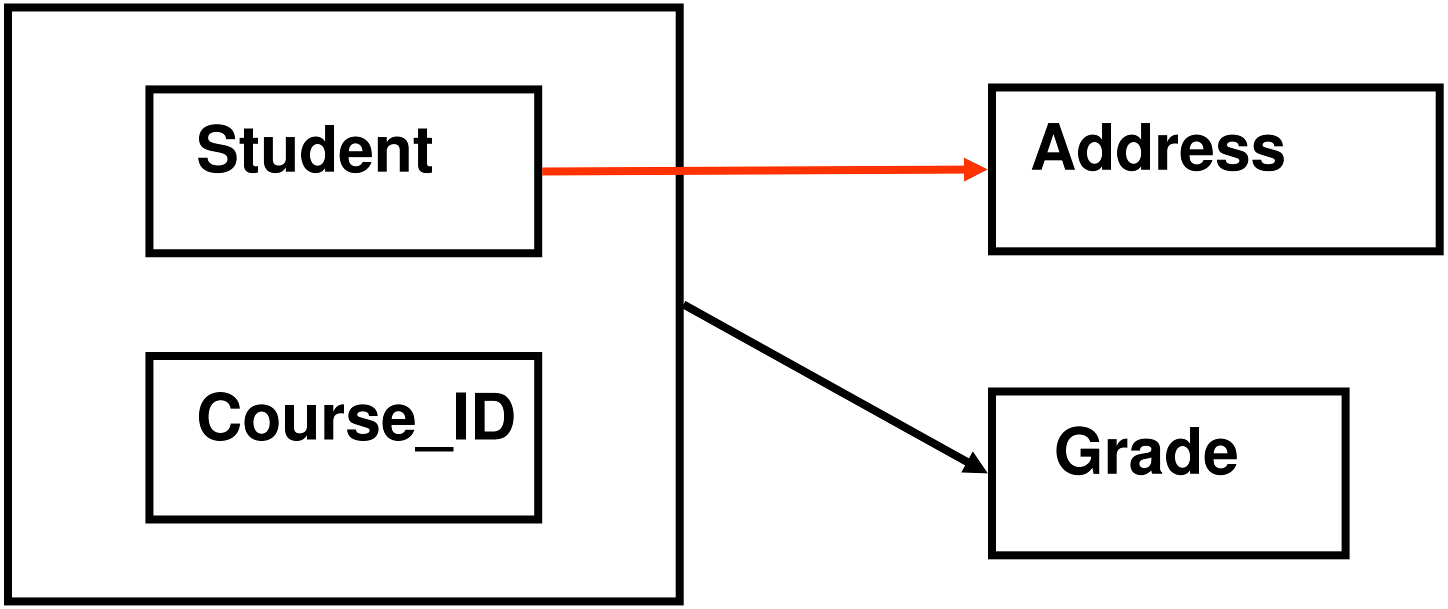
Second Normal Form (2NF)

Student	Address
Erik	80 Ericsson Av.
Sven	12 Olafson St.
Inge	192 Freya Blvd.
Hildur	212 Reykjavik St.

Student	Course_ID	Grade
Erik	CIS331	A
Sven	CIS331	B
Inge	CIS331	C
Hildur	CIS362	A

Normalized to 2NF

Second Normal Form (2NF)



Third Normal Form (3NF)

3NF:

- **2NF and**
- **no transitive dependencies**

Student	**Course_ID**	**Grade**	**Grade_value**
Erik	CIS331	A	4.00
Sven	CIS331	B	3.00
Inge	CIS331	C	2.00
Hildur	CIS362	A	4.00

Not in 3NF

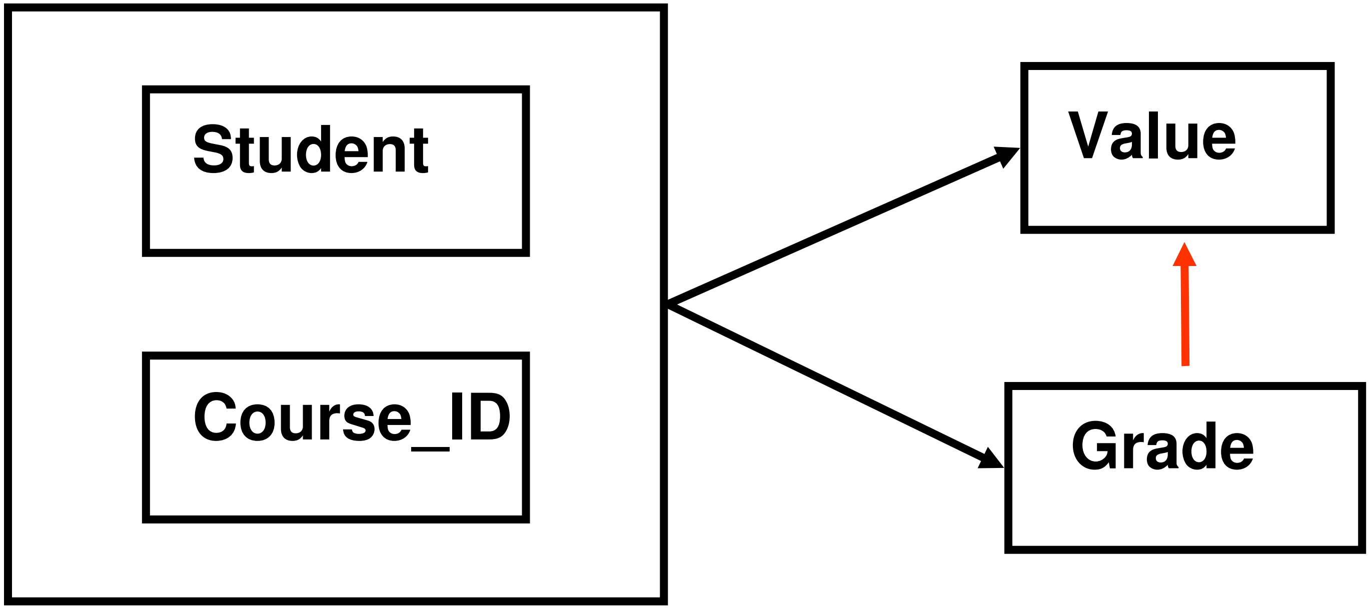
Third Normal Form (3NF)

Student	Course_ID	Grade
Erik	CIS331	A
Sven	CIS331	B
Inge	CIS331	C
Hildur	CIS362	A

Grade	Grade_value
A	4.00
B	3.00
C	2.00

Normalized to 3NF

Third Normal Form (3NF)



BCNF: Reminder

- Informally:

  Everything depends on the full key, and nothing but the key

- Formally:

  For every FD *a* → *b* in F+

  - ***a* → *b* is trivial (*a* is a superset of *b*) or**
  - ***a* is a superkey**
  - **(or both)**

Normal forms: BCNF vs. 3NF

- If a relation is in BCNF, it is always in 3NF (but not the converse)
- In practice, aim for
  - BCNF

  If that’s impossible, we accept
  - 3NF; but we insist on lossless join and dependency preservation

Normal forms: BCNF vs. 3NF

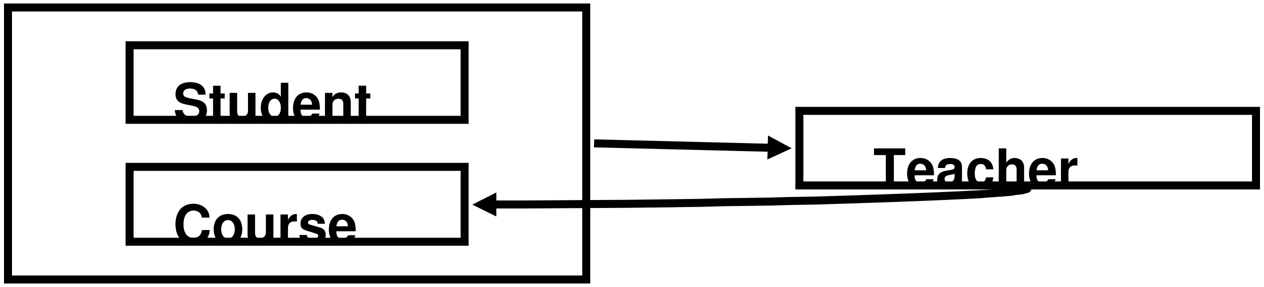
- **Let’s consider the ‘classic’ example:**

  STC (Student, Teacher, Course)

  Teacher → Course

  Student, Course → Teacher

  Is it in BCNF?



Normal forms: BCNF vs. 3NF

STC (Student, Teacher, Course)

Teacher → Course   Student, Course → Teacher

1) (Teacher, Course)   (Student, Course)

(BCNF? Y+Y - Lossless? No - Dep. pres.? No)

2) (Teacher, Course)   (Student, Teacher)

(BCNF? Y+Y - Lossless? Yes - Dep. pres.? No)

Normal forms: BCNF vs. 3NF

STC (Student, Teacher, Course)

Teacher → Course   Student, Course → Teacher

in this case it is impossible to have both

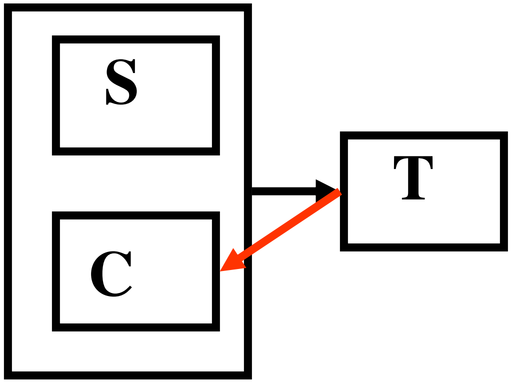
- BCNF and
- dependency preservation

So we have to use the (weaker) 3NF.

Normal forms: BCNF vs. 3NF

STC (Student, Teacher, Course)

Teacher → Course   Student, Course → Teacher



Informally, 3NF 'forgives' the red arrow

Normalization: Examples

Supplier	Name	Status_City	City	Part	Qty
S1	Jones	10	Paris	P3	257
S1	Jones	10	Paris	P1	500
S1	Jones	10	Paris	P4	125
S2	Spiritoso	12	London	P3	(null)
S7	Kohl	10	Paris	P4	342

- **Start by determining functional dependencies!**

Normalization: Example

FDs

- Supplier → Name
- Supplier → City → Status_City
- Supplier → Status_City
- (Supplier, Part) → Qty

Partial Dependencies

- (Supplier, Part) → Name
- (Supplier, Part) → City
- (Supplier, Part) → Status_City

Normalization: Example

Supplier	Name	Status_City	City
S1	Jones	10	Paris
S2	Spiritoso	12	London
S7	Kohl	10	Paris

Supplier	Part	Qty
S1	P1	500
S1	P3	257
S1	P4	125
S2	P3	(null)
S7	P4	342

Normalization: Example

- **We took care of the partial dependencies, but what about transitive dependencies?**

Supplier	**Name**	**Status_City**	**City**
S1	Jones	10	Paris
S2	Spiritoso	12	London
S7	Kohl	10	Paris

Normalization: Example

Supplier	Part	Qty
S1	P1	500
S1	P3	257
S1	P4	125
S2	P3	(null)
S7	P4	342

Supplier	Name	City
S1	Jones	Paris
S2	Spirit	London
S7	Kohl	Paris

City	Status_City
Paris	10
London	12

Normalization: Examples

- Silberschatz et al.:
  - 7.2 (lossless-join decomposition)

    hint: use the theorem!

    "A decomposition is a lossless-join decomposition if the joining attribute is a superkey in at least one of the new relations"
  - 7.16 (lossless-join decomposition)
  - 7.18 (dependency-preserving decomposition)

Normalization: Final Thoughts

- There are higher normal forms (4NF, 5NF), but we will not talk about them
- In practice, “normalized” means in BCNF or 3NF
- Luckily, in practice, ER diagrams lead to normalized tables (but do not rely on luck)

Normalization: Overview

- Why do we normalize?
  - To avoid redundancy (less storage space needed, and data is consistent)
  - To avoid update/delete anomalies

- A good decomposition should:
  - be a lossless join decomposition
    (you can recover original tables with a join)
  - preserve dependencies
    (FD's should not span two tables)

Normalization: Overview

- **Boyce-Codd Normal Form (BCNF):**
  - **“Everything should depend on the key, the whole key, and nothing but the key”** (so help me Codd – joke attributed to C.J. Date)

- **1NF (all attributes are atomic)**
- **2NF (no partial dependencies)**
- **3NF (no transitive dependencies)**