CMPT 321
FALL 2017

Normalization

Summary

By Marina Barsky

Normal forms –testing for normalization

- The database is normalized when all its tables are normalized
- There are rules to test each relation – normal forms:
  - 1NF
  - 2NF
  - 3NF
  - BCNF
  - 4NF
  - 5NF
- In most cases, the relation is normalized if it is in **3NF**

Students: in 1NF!

Students (<u>ID</u>, Name, <u>Course</u>, Grade)

Students			
ID	**Course**	**Name**	**Grade**
1	**Databases**	Bob	In pr
2	**HCI**	Maria	A
3	**Python**	John	B
4	**HCI**	Tom	A
2	**Algorithms**	Maria	A
1	**HCI**	Bob	B
2	**Python**	Maria	A

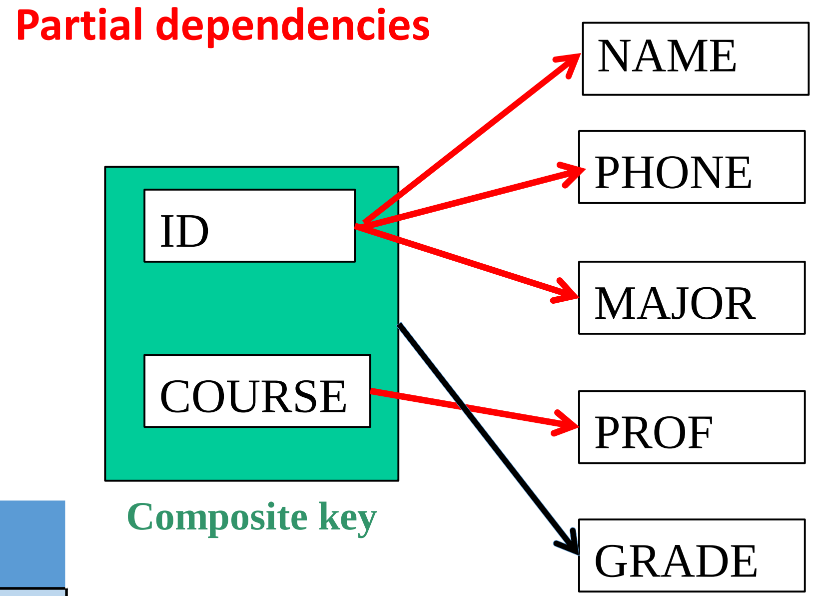
Students extended: problems

Students (<u>ID</u>, <u>Course</u>, Name, Phone, Major, Professor, Grade)

Students						
ID	**Course**	**Name**	**Phone**	**Major**	**Prof**	**Grade**
1	**Databases**	Bob	211-2112	CSCI	Dr. Monk	In pr
2	**HCI**	Maria	344-3344	BIOL	Dr. Pooh	A
3	**Python**	John	500-5005	MATH	Dr. Patel	B
4	**HCI**	Tom	601-6778	PHYS	Dr. Pooh	A
2	**Algorithms**	Maria	344-3344	BIOL	Dr. Monk	A
1	**HCI**	Bob	211-2112	CSCI	Dr. Pooh	B
2	**Python**	Maria	344-3344	BIOL	Dr. Patel	A

- Redundancy
- Insertion anomaly
- Deletion anomaly
- Update anomaly

Students in 2NF



Grades

ID	Course	Grade
1	Databases	In pr
2	HCI	A
3	Python	B
4	HCI	A
2	Algorithms	A
1	HCI	B
2	Python	A

Students

ID	Name	Phone	Major
1	Bob	211-2112	CSCI
2	Maria	344-3344	BIOL
3	John	500-5005	MATH
4	Tom	601-6778	PHYS

Courses

Course	Prof
Databases	Dr. Monk
HCI	Dr. Pooh
Python	Dr. Patel
Algorithms	Dr. Monk

Students (ID, Name, Phone, Major)

Courses (Course, Prof)

Grades (ID, Course, Grade)

Students relation: new information

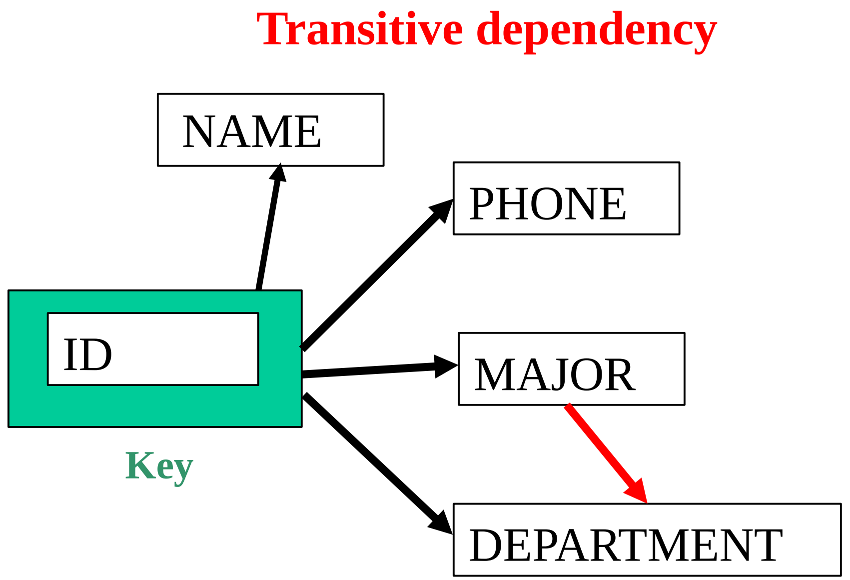
Students (<u>ID</u>, Name, Phone, Major, Department)

Students				
ID	**Name**	**Phone**	**Major**	**Department**
1	Bob	211-2112	CSCI	Computer Science
2	Maria	344-3344	BIOL	Life Sciences
3	John	500-5005	MATH	Mathematics and Statistics
4	Tom	601-6778	PHYS	Physics
5	Andrew	222-2341	CSCI	Computer Science
6	Ann	544-6778	STAT	Mathematics and Statistics

- Redundancy
- Update anomalies

Major → Department

Students in 3NF



Students			
ID	**Name**	**Phone**	**Major**
1	Bob	211-2112	CSCI
2	Maria	344-3344	BIOL
3	John	500-5005	MATH
4	Tom	601-6778	PHYS
5	Andrew	222-2341	CSCI
6	Ann	544-6778	STAT

MajorsOffered	
Major	Department
CSCI	Computer Science
BIOL	Life Sciences
PHYS	Physics
MATH	Mathematics and Statistics
STAT	Mathematics and Statistics

Students (<u>ID</u>, Name, Phone, Major)

MajorsOffered (<u>Major</u>, Department)

Boyce-Codd normal form - BCNF

- Relation is in 3NF
- All attributes **depend on the key, full key and nothing but the key**

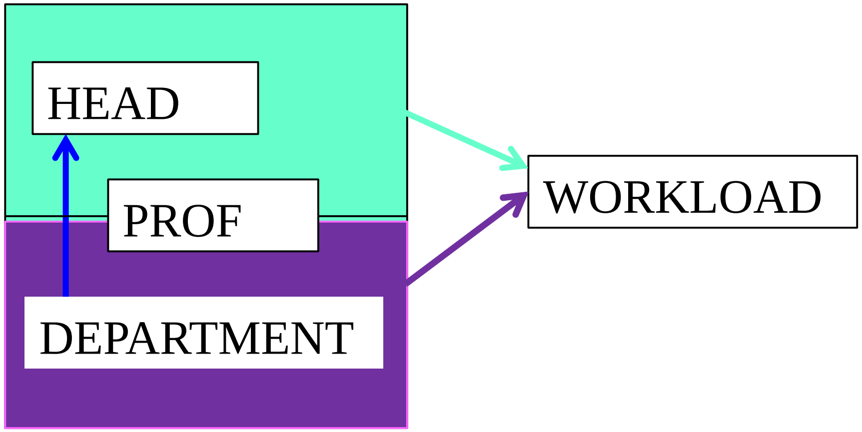
Professor workload: in BCNF?

	Professors		
Prof	**Department**	**Head**	**WorkLoad**
Dr. Monk	CSCI	Prof. Ming	30%
Dr. Pooh	MATH	Prof. Doe	70%
Dr. Patel	PHYS	Prof. Bond	100%
Dr. Pooh	CSCI	Prof. Ming	30%
Dr. Monk	BIOL	Prof. Bond	30%
Dr. Monk	MATH	Prof. Doe	40%

Department → Head

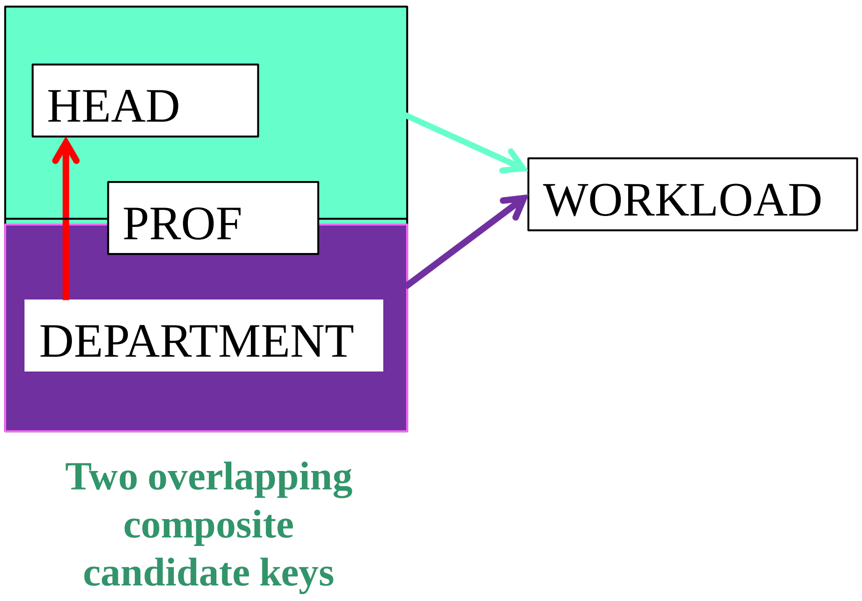
Prof, Department → Workload

Functional dependency diagram



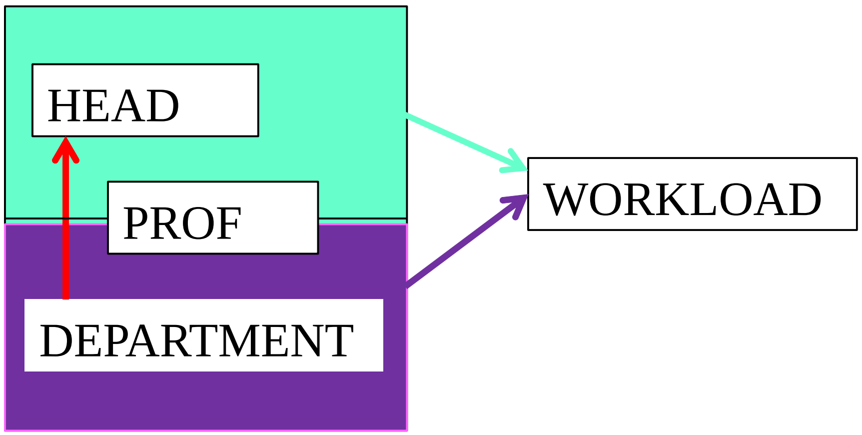
Two overlapping composite candidate keys

Functional dependency diagram



- BCNF violation: part of two candidate keys depends on another part

Professors in BCNF



Two overlapping composite candidate keys

BCNF violation: part of two candidate keys depends on another part

Professors		
Prof	**Department**	**WorkLoad**
Dr. Monk	**CSCI**	30%
Dr. Pooh	**MATH**	70%
Dr. Patel	**PHYS**	100%
Dr. Pooh	**CSCI**	30%
Dr. Monk	**BIOL**	30%
Dr. Monk	**MATH**	40%

Department	
Department	**Head**
CSCI	Prof. Ming
MATH	Prof. Doe
PHYS	Prof. Bond
BIOL	Prof. Bond

Professors (<u>Prof</u>, <u>Department</u>, Workload)

Department (<u>Department</u>, Head)

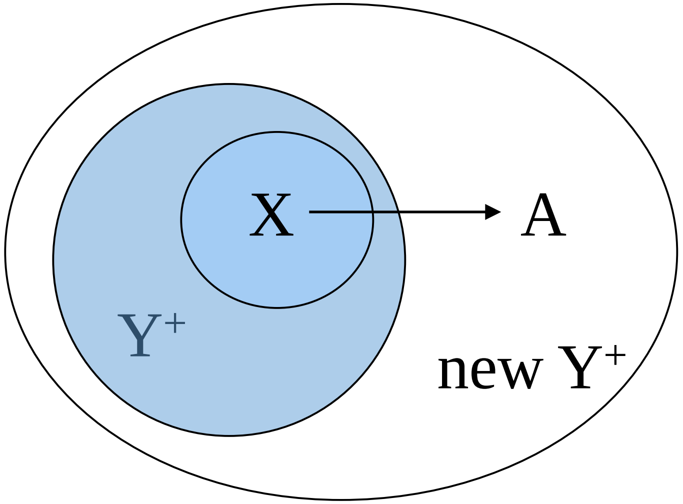
For full description of normal forms

- Read this article

Kent, W. (1983) A Simple Guide to Five Normal Forms in Relational Database Theory

BCNF decomposition: Step 1: for each FD compute closure

- Convert all FDs to LHS-singleton FD's using splitting rule
- Basis: $Y^+ = Y$.
- Induction: Look for an FD's left side X that is a subset of the current Y^+. If the FD is $X \rightarrow A$, add A to Y^+.



Example: computing closure: 1/4

- Given:

R(A,B,C,D) with FD’s AB → C, B → D, CD → A, AD → B.

- Computing closure for **AB**:

$\{AB\}^+ = \{ABC\}$ (from AB → C)

$\{ABC\}^+ = \{ABCD\}$ (from B → D)

- Answer:

$\{AB\}^+ = \{ABCD\}$

Example: computing closure: 2/4

- Given:

R(A,B,C,D) with FD's AB → C, B → D, CD → A, AD → B.

- Computing closure for **B**:

$\{B\}^+ = \{BD\}$ (from B → D)

- Answer:

$\mathbf{\{B\}^+ = \{BD\}}$

Example: computing closure: 3/4

- Given:

R(A,B,C,D) with FD's AB → C, B → D, CD → A, AD → B.

- Computing closure for CD:

$\{CD\}^+ = \{CDA\}$ (from CD → A)

{CDA}+={CDAB} (from AD → B)

- Answer:

$\{CD\}^+ = \{ABCD\}$

Example: computing closure: 4/4

- Given:

R(A,B,C,D) with FD's AB → C, B → D, CD → A, AD → B.

- Computing closure for **AD**:

$\{AD\}^+ = \{ADB\}$ (from AD → B)

{ADB}+={ADBC} (from AB → C)

- Answer:

$\{AD\}^+ = \{ABCD\}$

BCNF decomposition: step 2 – identify violations

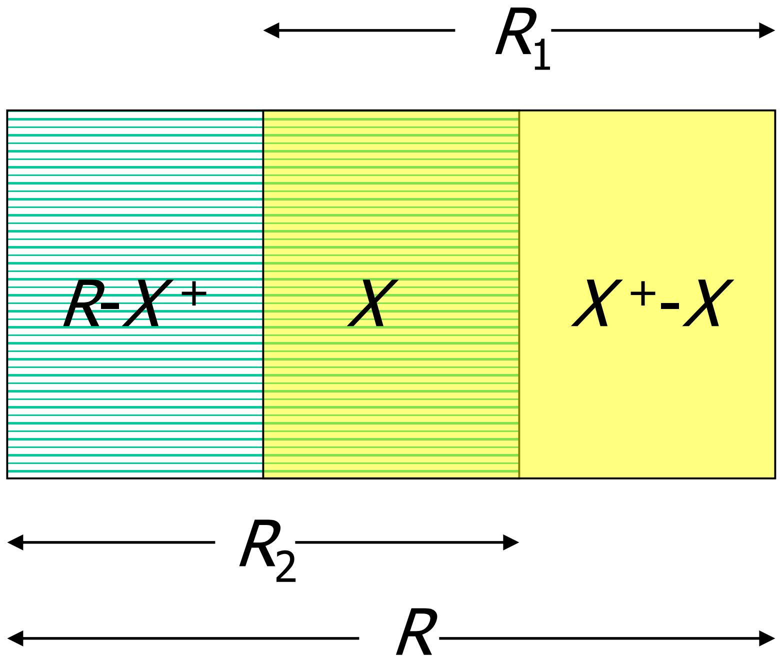
- Given:

R(A,B,C,D) with FD's AB → C, B → D, CD → A, AD → B.

- **{AB}$^+$ = {ABCD}**
- **{B}$^+$ = {BD}** ◄ B → D BCNF violation! B is not a key
- **{CD}$^+$ = {ABCD}**
- **{AD}$^+$ = {ABCD}**

BCNF decomposition: step 3 - decompose

- Replace R by relations with schemas:
  1. $R_1 = X^+$
  2. $R_2 = R - (X^+ - X)$



BCNF decomposition: step 3 – decompose

- Given:

R(A,B,C,D) with FD's AB → C, B → D, CD → A, AD → B.

- **$\{AB\}^+ = \{ABCD\}$**
- **$\{B\}^+ = \{BD\}$**
- **$\{CD\}^+ = \{ABCD\}$**
- **$\{AD\}^+ = \{ABCD\}$**

R(A,B,C,D) → **R1(B,D)**
R2(A,B,C)

Desired properties of normalization: after decomposition

- No redundancies and anomalies: guaranteed
- Recoverability of information: if decompose according to functional dependencies – this is guaranteed (Chase test)
- Preservation of original FD’s in decomposed relations

BCNF decomposition which does not preserve FD's

- There is one structure of FD's that causes trouble when we decompose.

AB → C and ***C → B***

- There are two keys, {*A*,*B*} and {*A*,*C*}
- ***C → B*** is a BCNF violation, so we must decompose into *AC*, *BC*

- The difference here that a violating FD ***C → B*** has B in RHS, and **B is a part of a primary key**
- An attribute that is a part of some key is called a ***prime***

Example: BCNF gone wrong

- Given **R (client, bank, banker)** with FD's:

{client, bank} → banker - **{client, bank} is the key**

banker → bank – violation

- We decompose into

R1 (banker, bank)

R2 (client, banker)

- However the original FD {client, bank} → banker is lost in this decomposition!

Example continued: at the moment of decomposition

- R (client, bank, banker)
- FD’s:

{client, bank} → banker

banker → bank

- Decomposition:

R1 (banker, bank)

R2 (client, banker)

{client, bank} → banker
banker → bank

R		
client	bank	banker
A	1	X
A	2	Y
B	1	X

banker → bank

R1	
banker	bank
X	1
Y	2

No FD’s

R1	
client	banker
A	X
A	Y
B	X

Example continued: lossless decomposition

banker → bank

R1	
banker	bank
X	1
Y	2

⋈

No FD's

R2	
client	banker
A	X
A	Y
B	X

{client, bank} → banker
banker → bank

R		
client	bank	banker
A	1	X
A	2	Y
B	1	X

The decomposition is lossless – requirement 2 is satisfied

Example continued: no original constraint {client, bank} → banker

banker → bank

R1	
banker	bank
X	1
Y	1

No FD's

R2	
client	banker
A	X
A	Y
B	X

The only requirement is that banker uniquely identifies bank

Now we can insert into R1 and R2 without the original constraints, and that will allow to insert invalid values

Example continued: no original constraint {client, bank} → banker

banker → bank

R1	
banker	bank
X	1
Y	**1**

⋈

No FD's

R2	
client	banker
A	X
A	Y
B	X

{client, bank} → banker
banker → bank

R		
client	bank	banker
A	1	X
A	1	Y
B	1	X

Invalid join! Tuple (A, 1, Y) should have been prevented by the original FD {client, bank} → banker

Relaxing normalization requirements: 3NF

- 3rd Normal Form (3NF) modifies the BCNF condition so we do not have to decompose in this problematic situation

- An attribute is ***prime*** if it is a member of any key.
- *X* → *A* violates 3NF if and only if *X* is not a superkey, and also *A* is not prime

Example: 3NF

- ***AB → C*** and ***C → B***
- In our situation with FD's *AB → C* and *C → B*, we have key *AB*
- Thus *A* and *B* are each prime.
- Although *C → B* violates BCNF, it **does not violate 3NF**
- So no decomposition is performed, and all the original FD's are preserved

Desired properties of normalization: after decomposition: BCNF

- No redundancies and anomalies ✓
- Recoverability of information ✓
- Preservation of original FD’s ✓ ✗

Desired properties of normalization: after decomposition: 3NF

- No redundancies and anomalies ✓ ✗
- Recoverability of information ✓
- Preservation of original FD’s ✓

Relationship between normal forms