

Normalization

3NF and BCNF

CS 4750

Database Systems

[A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts, Ch.7]

[Ricardo and Urban, Database Illuminated, Ch.6]

[<https://www.w3schools.in/dbms/database-normalization/>]

Recap: Normalization

Goals:

- Reduce redundancy
- Improve data integrity
- Queries are logical and efficient

Decomposition: Three properties that must satisfied



3NF and Decomposition

- **Lossless-join**
- Always **dependency preserving**
- Possible to have extra data (there may be redundancy)

Questions:
Is the relation in 3NF?
Is any refinement needed?

To calculate 3NF

- Take **Canonical Cover (Fc)**
- Turn **(minimal set of) FDs into tables**

Canonical Cover (F_c)

- A **minimal set** of functional dependencies that has the same closure as the original set F
- Extraneous attributes = attribute of FDs that we can removed without changing the closure of FDs
- F logically implies all dependencies in F_c
- F_c logically implies all dependencies in F
- No FD in F_c contains an extraneous attribute

F and F_c are logically equivalent

Minimal basis for a set of FDs: For any set of FDs, there is at least one minimal basis, which is a set of FDs **equivalent to the original** (each set implies the other set), with singleton right sides, **no FD that can be eliminated** while preserving equivalence, and **no attribute in a left side that can be eliminated** while preserving equivalence

Canonical Cover (F_c)

Compute the canonical cover of a set of functional dependencies F

Always start with F and use rules to minimize

$F_c = F$

repeat

 apply union rule to replace any dependencies $f: X_1 \rightarrow Y_1$
 and $f: X_1 \rightarrow Y_2$ with $f: X_1 \rightarrow Y_1Y_2$

for each functional dependency f_i

if f_i contains an extraneous attribute either in X or in Y

then remove an extraneous attribute

until F_c does not change any further

Example 1: 3NF and Fc

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

Let's do this together

FDs = $\{ A \rightarrow B, AB \rightarrow D, B \rightarrow BDE, C \rightarrow D, D \rightarrow D \}$

Compute Fc and convert the relation into 3NF

Compute Fc without using reflexivity

(1) write all LHS

A \rightarrow
B \rightarrow
AB \rightarrow
C \rightarrow
D \rightarrow

(2) copy FDs as is

A \rightarrow B
B \rightarrow B DE
AB \rightarrow D
C \rightarrow D
D \rightarrow D

(3) remove reflexivity

A \rightarrow B
B \rightarrow ~~B~~ DE
AB \rightarrow D
C \rightarrow D
~~D \rightarrow D~~

(4) remove extraneous attr

A \rightarrow B
B \rightarrow DE
~~AB \rightarrow D~~
C \rightarrow D

A \rightarrow B and B \rightarrow D.
Thus, remove AB \rightarrow D

Example 1: 3NF and Fc

Let's do this together

(from previous page)

(4) Remove
extraneous attr

A	→	B
B	→	DE
C	→	D

Super key

Turn Fc into tables

AB
BDE
CD

A relation $R(A, B, C, D, E)$ is converted into 3NF by putting LHS and RHS of each FD in Fc together in **one** relation.

Dependency preserving

$R(A, B, C, D, E)$ becomes $R_1(A, B)$ and $R_2(B, D, E)$ and $R_3(C, D)$

Or write it in another format: AB // BDE // CD

Example 2: 3NF and Fc

Given $R(A,B,C)$

FDs = $\{ A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C \}$

Compute Fc and convert the relation into 3NF

Let's do this together

(1) write all LHS

$A \rightarrow$
 $B \rightarrow$
 $AB \rightarrow$

(2) copy FDs as is

$A \rightarrow BC$
 $B \rightarrow C$
 $AB \rightarrow C$

Combine the given FDs $A \rightarrow BC$ and $A \rightarrow B$

No reflexivity to remove

(3) remove extraneous attr

$A \rightarrow BC$
 $B \rightarrow C$
 ~~$AB \rightarrow C$~~

Consider $AB \rightarrow C$ and $B \rightarrow C$,
 A is an extraneous attr, remove A from $AB \rightarrow C$ (resulting in $B \rightarrow C$)

(4) remove extraneous attr

$A \rightarrow \overline{BC}$
 $B \rightarrow C$

Apply decomposition to $A \rightarrow BC$, thus, $A \rightarrow B$ and $A \rightarrow C$.

$A \rightarrow C$ is logically equivalent to $A \rightarrow B$ and $B \rightarrow C$ (transitivity).

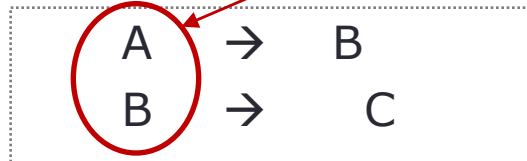
Thus, C is an extraneous attr, remove C from $A \rightarrow BC$

Example 2: 3NF and Fc

Let's do this together

(from previous page)

(4) Remove
extraneous attr



Super key

Turn Fc into tables

AB
BC

$R(A, B, C)$ becomes $R_1(A, B)$ and $R_2(B, C)$

Or write it in another format: AB // BC

BCNF and Decomposition

- **Lossless-join**
- Guarantee **redundancy free**
- May involve dependency across relations

Given a relation R ,

for every **nontrivial** FD $X \rightarrow Y$ in R , X is a **super key**

- For all FDs, “key \rightarrow everything”

Questions:

Is the relation in BCNF?

Is any refinement needed?

BCNF and Decomposition

To calculate BCNF

Compute F^+

repeat given a relation R (or a decomposed R) and FDs F

for each functional dependency f_i in a relation R

if f_i violates $X \rightarrow Y$

then decompose R into two relations:

one with $X \cup Y$ as its attributes (i.e., everything f)

one with $X \cup (\text{attrs}(R) - X - Y)$ as its attributes

until no violation

Example: BCNF and F+

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

Let's do this together

FDs = $\{ A \rightarrow B, AB \rightarrow D, B \rightarrow BDE, C \rightarrow D, D \rightarrow D \}$

Compute F+ and convert the relation into BCNF

Compute F+

(1) write all LHS
& remaining

A \rightarrow
B \rightarrow
AB \rightarrow
C \rightarrow
D \rightarrow
E \rightarrow

(2) copy FDs as is

A \rightarrow B
B \rightarrow B DE
AB \rightarrow D
C \rightarrow D
D \rightarrow D
E \rightarrow

(3) apply reflexivity

A \rightarrow AB
B \rightarrow B DE
AB \rightarrow AB D
C \rightarrow CD
D \rightarrow D
E \rightarrow E

(4) apply transitivity

A \rightarrow AB DE
B \rightarrow B DE
AB \rightarrow AB DE
C \rightarrow CD
D \rightarrow D
E \rightarrow E

Example: BCNF and F+

Let's do this together

(from previous page)

(4) apply transitivity

F+

A	→	AB DE
B	→	B DE
AB	→	AB DE
C	→	CD
D	→	D
E	→	E

Not trivial FDs. A, B, C are not super key

Trivial FDs (let's abbreviate to TR)

Based on F+, let's rewrite using the following format to help us calculate

R (A B C D E)
! ! ! TR TR

TR – trivial
SK – super key
X – neither trivial nor super key
! – (possibly) need to work on

Example: BCNF and F+

R (A B C ~~D~~ ~~E~~)
! ! ! TR TR

Let's do this together

F+	A	→	AB	DE
	B	→	B	DE
	AB	→	AB	DE
	C	→		CD
	D	→		D
	E	→		E

To choose which FD to work on, two ways:

- Choose the first FD, or
- Choose the longest FD (yield better solution)

Let's consider A:

A is not a super key, not trivial, thus $A \rightarrow ABDE$ violates BCNF, break a relation on A

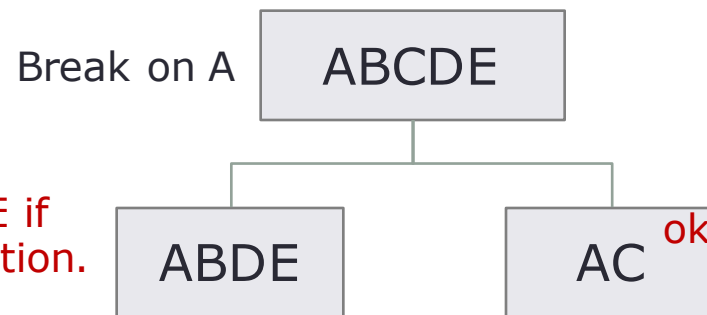
Example: BCNF and F+

R (A B C D E)
 ! ! ! TR TR

A → ABDE

Take *RHS*, make a table: ABDE

Take *LHS*, make a table where A is a key
 A plus (original – (*RHS*))



Verify table ABDE if there is any violation.

A is a super key.
 Still need to work on B.

Let's do this together

F+	A	→	AB	DE
	B	→	B	DE
	AB	→	AB	DE
	C	→	CD	
	D	→	D	
	E	→	E	

--- thus, AC

There are only 2 attrs,
 this relation is ok

Restriction: **Cannot** break on A 2 times in a row

Next: consider B. B is neither trivial nor super key, break on B

Example: Calculate BCNF

R (A B D E)
 X ! TR TR

Let's do this together

B → BDE

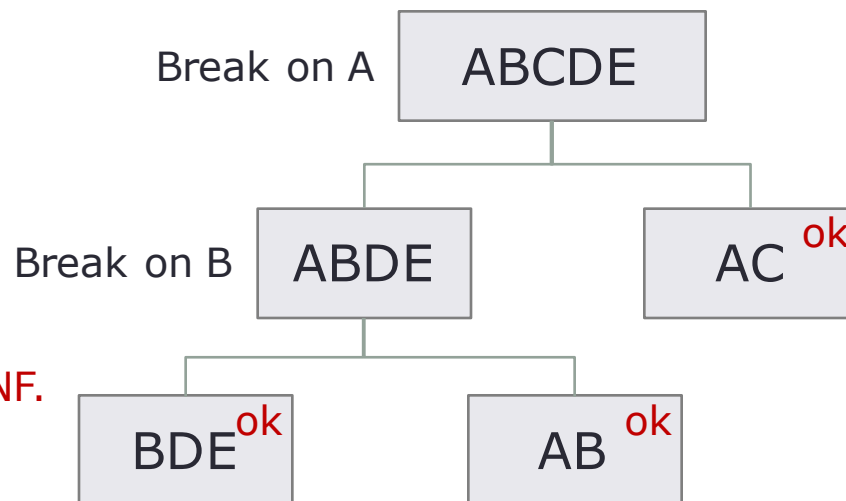
F+			
A	→	AB	DE
B	→	B	DE
AB	→	AB	DE
C	→		CD
D	→		D
E	→		E

Take *RHS*, make a table: BDE

Take *LHS*, make a table where B is a key
 B plus (original - (*RHS*))

--- thus, AB

There are only 2 attrs,
 this relation is ok



Verify that B → BDE
 does not violate BCNF.

B is super key.

D and E are trivial.

R(ABCDE) becomes
 AC // AB // BDE

(notice: results are different 3NF)

Wrap-Up

- Compute F^+ and F_c
- 3NF and decomposition
- BCNF and decomposition

What's next?

- SQL

Additional practice

Practice 1: Decomposition

Given $R(A, B, C)$

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

Supposed R is decomposed in two different ways :

1. $R_1(A, B), R_2(B, C)$

- Does this satisfy **lossless-join decomposition**?

- Does this satisfy **dependency preserving**?

2. $R_1(A, B), R_2(A, C)$

- Does this satisfy **lossless-join decomposition**?

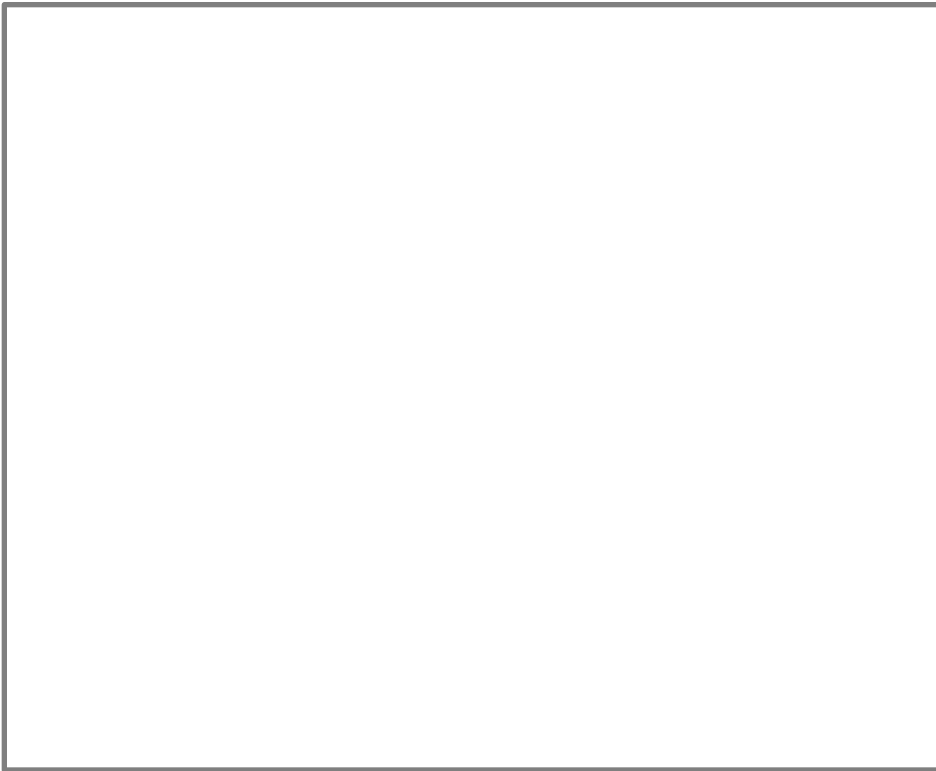
- Does this satisfy **dependency preserving**?

Practice 2: 3NF and BCNF

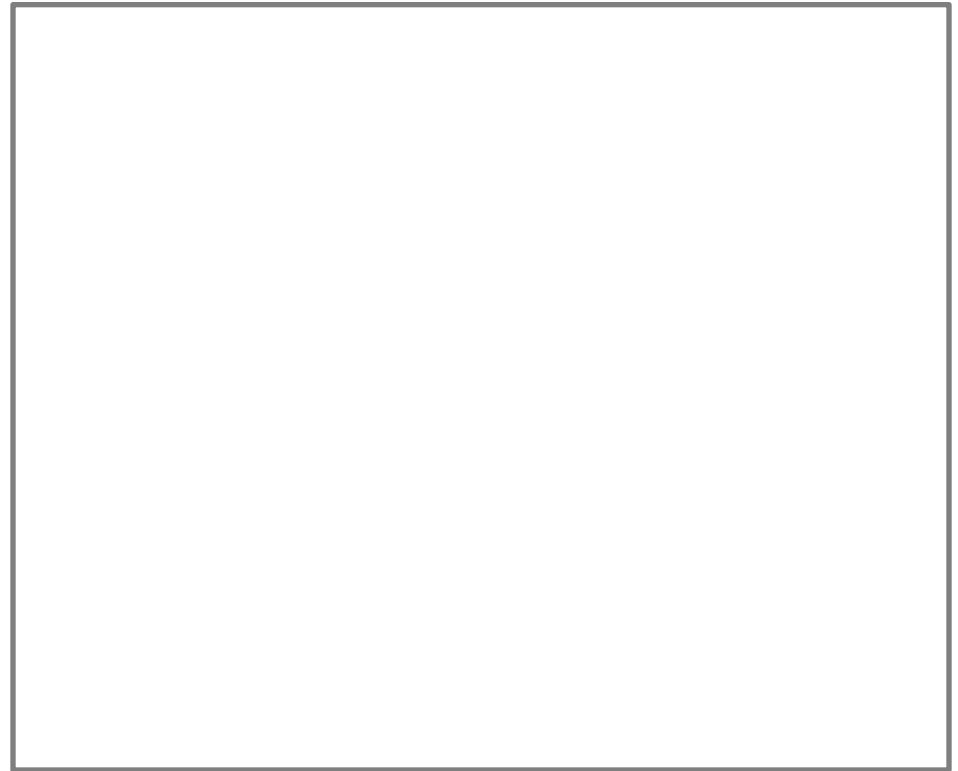
Given $R(A, B, C, D, E)$

FDs = $\{ A \rightarrow C, C \rightarrow DE, D \rightarrow B, A \rightarrow D \}$

Decompose table R using 3NF

A large empty rectangular box with a thin black border, intended for the student to write the decomposition of table R into 3NF.

Decompose table R using BCNF

A large empty rectangular box with a thin black border, intended for the student to write the decomposition of table R into BCNF.

Practice 3: 3NF

Does the Customer_order table satisfy 3NF requirements?
If not, convert the table into 3NF

Customer_order

OrderId	CustomerID	Date	Store	Address
1	2	10/1/2019	South	11 Sorth Str
2	1	9/25/2019	West	22 West Str
3	3	8/12/2019	East	33 East Str
4	4	10/23/2019	West	22 West Str
5	8	5/11/2019	North	44 North Str
6	6	5/11/2019	South	11 Sorth Str
7	5	7/31/2019	East	33 East Str
8	7	10/17/2019	West	22 West Str
9	6	9/19/2019	North	44 North Str
10	4	10/23/2019	North	44 North Str
...

Practice 3: 3NF - solution

Practice 4: BCNF

Does the Student_Major_Advisor table satisfy BCNF requirements?
If not, convert the table into BCNF

Student_Major_Advisor

ComputingID	Major	Advisor
ht1y	Computer Science	someone1
dt2y	Physics	someone2
dt2y	Engineering	somoone3
md3y	Computer Science	someone4
mn4e	Math	someone5
md3y	Computer Science	someone1

computingID, Major \rightarrow Advisor
Advisor \rightarrow Major

Assume:
(semantic/business rules)

- Each Student may major in several subjects.
- For each Major, a given Student has only one Advisor.
- Each Major has several Advisors.
- Each Advisor advises only one Major.
- Each Advisor advises several Students in one Major.

Practice 4: BCNF - solution
