Databases
Further normalization

Hans Philippi

February 22, 2024

Normalization

Requirement 1:
Your decomposition should be *lossless*

Requirement 2:
Your decomposition should avoid *redundancy*

Normalization: refreshing your memory

r		
A	**B**	**C**
a	10	100
b	20	200
a	10	300
b	20	400
c	30	500
b	20	600

We notice an FD: $A \rightarrow B$

Normalization: prevent redundancy caused by an FD

Requirement 2
Prevent redundancy caused by $A \rightarrow B$;
split r into r_1 and r_2

r		
A	**B**	**C**
a	10	100
b	20	200
a	10	300
b	20	400
c	30	500
b	20	600

r_1	
A	**B**
a	10
b	20
c	30

r_2	
A	**C**
a	100
b	200
a	300
b	400
c	500
b	600

Normalization: guarantee losslessness

Requirement 1

Guarantee losslessness by splitting over $A \rightarrow B$:

$$r_1 \bowtie r_2 = r$$

r_1	
A	**B**
a	10
b	20
c	30

r_2	
A	**C**
a	100
b	200
a	300
b	400
c	500
b	600

r		
A	**B**	**C**
a	10	100
b	20	200
a	10	300
b	20	400
c	30	500
b	20	600

Normalization: the Decomposition Theorem

If we have a scheme $R(XYZ)$ and an FD $X \to Y$,

then the decomposition on $R_1(XY)$ and $R_2(XZ)$ is lossless

Normalization: Boyce-Codd Normal Form

Avoiding redundancy

Suppose we have a scheme R and a set FDs F

R is in BCNF (w.r.t. F) iff each left side of a non-trivial FD is a superkey

Normalization

Algorithm:

```
INPUT: a universe $R$, a set FDs $F$
OUTPUT:
    a lossless BCNF-decomposition van $R$
METHOD:
    while there is a scheme $S$ not in BCNF {
        suppose the villain has left side $X$;
        let $Y = X^+$ without $X$;
        let $Z$ be the remaining attributes;
        split $S(XYZ)$ into $S_1(XY), S_2(XZ)$;
    }
```

Normalization

properties of BCNF

- BCNF represents the 'ultimate' level of redundancy prevention caused by FDs
- implementation: in a BCNF scheme, the left sides of FDs are keys, so you are able to enforce FDs by defining indices

What is an index?

```
SELECT * FROM Patient
   WHERE patientId = 3141592653
```

- An index on patientId supports direct access to the tuple satisfying the selection predicate
- Primary keys are automatically supported by an index
- Attributes often used in selections (e.g. birth_date) might have an additional index
- Plural: indexes or indices

Observations

- the BCNF decomposition algorithm is not deterministic
- the number of possible BCNF decompositions may be very very large
- some BCNF decompositions turn out to be preferable to other BCNF decompositions

Back to the decomposition quality problem

Reminder: we have a relation scheme $R(ABCDE)$
and a set of fd's
$F = \{A \rightarrow BC, C \rightarrow D, D \rightarrow E\}$

Give at least two BCNF decompositions

Do you have a preference for one of the decompositions?

- (CD), (CE), (ABC)
- (DE), (CD), (ABC)

Let us try to to make the choice explicit

More observations

To support our search for *better* BCNF decompositions, we will need the notion of a

minimal representation of an FD set

FD sets may contain redundancy:

- $\{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$
- $\{A \rightarrow B, AB \rightarrow C\}$
- $\{A \rightarrow BC, B \rightarrow C\}$

Types of redundancy in FD sets

- redundant FD:
  $\{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$

  $A \rightarrow C$ is redundant
- reducable left side:
  $\{A \rightarrow B, AB \rightarrow C\}$

  B is superfluous in $AB \rightarrow C$
- reducable right side:
  $\{A \rightarrow BC, B \rightarrow C\}$

  C is superfluous in $A \rightarrow BC$

Maximal representation of FD set: closure

Definition:

F^+ is the set of all FD's derivable from F

We call F^+ the <u>closure</u> of F

don't try to calculate F^+ at home

Definition:

Two FD-sets F, G are <u>equivalent</u> if $F^+ = G^+$

INTERMEZZO

Suppose $F = \{A_1 \to A_2, A_2 \to A_3, \ldots, A_{n-1} \to A_n, A_n \to A_1\}$

Give an estimation (or a lower limit) for $|F^+|$ (the size of F^+).

What about $n = 100$?

Concise representation of FD set

Definition:

G is a minimal cover of F
if F and G are equivalent and

1. G does not contain redundant FDs
2. all right sides in G are minimal
3. all left sides in G are minimal

Terminology: minimal basis = minimal cover

Minimal cover calculation

Algorithm

INPUT: a set FDs F
OUTPUT: a minimal cover for F
METHOD:

1. split each FD into single right side FDs
2. reduce left sides
3. eliminate redundant FDs
4. (not mandatory) combine FDs with identical left sides

Minimal cover calculation

2. *reduce left sides:*

```
for each FD $U \to V$ in $F$ {
    for each attribute $A$ in $U$ {
        let $U' = U - \{A\}$;
        if $U' \to V$ can be derived from $F$
        then replace $U \to V$ with $U' \to V$;
    }
}
```

Minimal cover calculation

3. *eliminate redundant FDs:*

```
for each FD $U \to V$ in $F$ {
    let $F' = F - \{U \to V\}$;
    if $U \to V$ can be derived from $F'$
    then delete $U \to V$ from $F$
};
```

INTERMEZZO

We have a relation scheme R(ABCDEFGHKL)
and a set FD's
$\mathbf{F} = \{A \rightarrow D, B \rightarrow HA, GH \rightarrow AKL, CH \rightarrow BK, G \rightarrow BC\}$

Calculate a minimal cover for **F**

Step 1: split into single right side FD's

$A \rightarrow D$

$B \rightarrow H$

$B \rightarrow A$

$GH \rightarrow A$

$GH \rightarrow K$

$GH \rightarrow L$

$CH \rightarrow B$

$CH \rightarrow K$

$G \rightarrow B$

$G \rightarrow C$

Step 2: reduce left sides

Can $GH \rightarrow A$ be reduced to $G \rightarrow A$?

$A \rightarrow D$

$B \rightarrow H$

$B \rightarrow A$

$GH \rightarrow A$

$GH \rightarrow K$

$GH \rightarrow L$

$CH \rightarrow B$

$CH \rightarrow K$

$G \rightarrow B$

$G \rightarrow C$

Step 2: reduce left sides

Can $GH \rightarrow A$ be reduced to $G \rightarrow A$?
$G^{+} = GBCHAKLD$, so $G \rightarrow A$ holds!

$A \rightarrow D$
$B \rightarrow H$
$B \rightarrow A$
$G \rightarrow A$
$GH \rightarrow K$
$GH \rightarrow L$
$CH \rightarrow B$
$CH \rightarrow K$
$G \rightarrow B$
$G \rightarrow C$

Step 2: reduce left sides

$G^+ = GBCHAKLD$, so $G \rightarrow KL$ also holds

$A \rightarrow D$

$B \rightarrow H$

$B \rightarrow A$

$G \rightarrow A$

$G \rightarrow K$

$G \rightarrow L$

$CH \rightarrow B$

$CH \rightarrow K$

$G \rightarrow B$

$G \rightarrow C$

Step 2: reduce left sides

Can $CH \rightarrow B$ be reduced to $C \rightarrow B$?
$C^+ = C$, so $C \rightarrow B$ does not hold!
Neither do $H \rightarrow B, C \rightarrow K, H \rightarrow K$

$A \rightarrow D$
$B \rightarrow H$
$B \rightarrow A$
$G \rightarrow A$
$G \rightarrow K$
$G \rightarrow L$
$CH \rightarrow B$
$CH \rightarrow K$
$G \rightarrow B$
$G \rightarrow C$

Step 3: eliminate redundant FD's

Can $A \rightarrow D$ be eliminated?

$A \rightarrow D$

$B \rightarrow H$

$B \rightarrow A$

$G \rightarrow A$

$G \rightarrow K$

$G \rightarrow L$

$CH \rightarrow B$

$CH \rightarrow K$

$G \rightarrow B$

$G \rightarrow C$

Step 3: eliminate redundant FD's

Can $A \rightarrow D$ be eliminated?
Try to derive $A \rightarrow D$ from the remaining FD set $\mathbf{F}' = \mathbf{F} - \{A \rightarrow D\}$.
Within $\mathbf{F}'$, $A^+ = A$, so $A \rightarrow D$ cannot be derived.

$A \rightarrow D$

$B \rightarrow H$
$B \rightarrow A$
$G \rightarrow A$
$G \rightarrow K$
$G \rightarrow L$
$CH \rightarrow B$
$CH \rightarrow K$
$G \rightarrow B$
$G \rightarrow C$

Step 3: eliminate redundant FD's

Can $B \to H$ be eliminated?

$A \to D$

$B \to H$

$B \to A$

$G \to A$

$G \to K$

$G \to L$

$CH \to B$

$CH \to K$

$G \to B$

$G \to C$

Step 3: eliminate redundant FD's

Can $B \rightarrow H$ be eliminated?
Try to derive $B \rightarrow H$ from $\mathbf{F}' = \mathbf{F} - \{B \rightarrow H\}$.
Within $\mathbf{F}'$, $B^+ = BAD$, so $B \rightarrow H$ cannot be derived.

$A \rightarrow D$

$B \rightarrow H$

$B \rightarrow A$

$G \rightarrow A$

$G \rightarrow K$

$G \rightarrow L$

$CH \rightarrow B$

$CH \rightarrow K$

$G \rightarrow B$

$G \rightarrow C$

Step 3: eliminate redundant FD's

Can $G \to A$ be eliminated?

$A \to D$

$B \to H$

$B \to A$

$G \to A$

$G \to K$

$G \to L$

$CH \to B$

$CH \to K$

$G \to B$

$G \to C$

Step 3: eliminate redundant FD's

Can $G \to A$ be eliminated?
Try to derive $G \to A$ from $\mathbf{F}' = \mathbf{F} - \{G \to A\}$.
Within $\mathbf{F}'$, $G^+ = GKLBCDHA$, so $G \to A$ can be derived.

$A \to D$
$B \to H$
$B \to A$

$G \to A$

$G \to K$
$G \to L$
$CH \to B$
$CH \to K$
$G \to B$
$G \to C$

Step 3: eliminate redundant FD's

Final result, minimal cover

$A \rightarrow D$
$B \rightarrow H$
$B \rightarrow A$
$G \rightarrow L$
$CH \rightarrow B$
$CH \rightarrow K$
$G \rightarrow B$
$G \rightarrow C$

or, equivalently: $\{A \rightarrow D, B \rightarrow HA, G \rightarrow LBC, CH \rightarrow BK\}$

Back to the decomposition quality problem

Reminder: we have a relation scheme $R(ABCDE)$
and a set of fd's
$F = \{A \rightarrow BC, C \rightarrow D, D \rightarrow E\}$

Give at least two BCNF decompositions

Do you have a preference for one of the decompositions?

- (CD), (CE), (ABC)
- (DE), (CD), (ABC)

The second solution is *dependency preserving*

Projection of FD's

Definition:

The projection of FD $U \rightarrow V$ on scheme R is:

1. $U \rightarrow V$, if $UV \subseteq attr(R)$
2. void, if one of the attributes (left or right) is not in $attr(R)$

Dependency preserving decompositions

Definition:
Suppose we have a scheme R and a set FDs F.
A decomposition of R into R_1, R_2 is called
dependency preserving (DP) if:

$$(F_1 \cup F_2)^+ = F^+$$

where F_i is the projection of F^+ on R_i

Dependency preserving decomposition: examples

Suppose we have a scheme $R(ABCDE)$, with
$F = \{A \rightarrow BE, C \rightarrow DE\}$

Decomposition step 1:
$R_1(ABC), R_3(CDE)$

Decomposition step 2:
$R_1(AB), R_2(AC), R_3(CDE)$

$F_1 = \{A \rightarrow B, ...\}, F_2 = \{...\}, F_3 = \{C \rightarrow D, C \rightarrow E, ...\}$

Where is $A \rightarrow E$?

Can it be derived?

This decomposition is not DP!

Dependency preserving decomposition: examples

Suppose we have a scheme $R(ABCDE)$, with
$F = \{A \rightarrow BCE, C \rightarrow DE\}$

Decomposition step :
$R_1(ABC), R_2(CDE)$

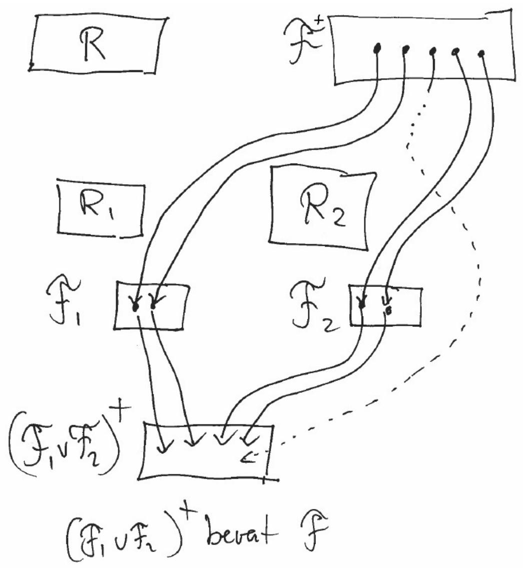
$F_1 = \{A \rightarrow B, A \rightarrow C, ...\}, F_2 = \{C \rightarrow D, C \rightarrow E, ...\}$

Where is $A \rightarrow E$?

It can be found in $(F_1 \cup F_2)^+$!

This decomposition is DP!

DPdecomposition: artist's impression


R
$\mathcal{F}^+$
R_1
R_2
$\mathcal{F}_1$
$\mathcal{F}_2$
$(\mathcal{F}_1 \cup \mathcal{F}_2)^+$
$(\mathcal{F}_1 \cup \mathcal{F}_2)^+$ bevat $\mathcal{F}$

INTERMEZZO

$R = (ABCDE)$
$F = \{A \to BC, D \to E\}$
$R_1 = (ABC), R_2 = (ADE)$
Is this decomposition lossless, BCNF, DP?

$R = (ABCDE)$
$F = \{A \to BC, D \to E\}$
$R_1 = (AB), R_2 = (AC), R_3 = (ADE)$
Is this decomposition lossless, BCNF, DP?

$R = (ABCDE)$
$F = \{A \to BCDE, C \to A, D \to E\}$
$R_1 = (ABC), R_2 = (CD), R_3 = (DE)$
Is this decomposition lossless, BCNF, DP?

Observations

- a DP decomposition is preferable to a non DP decomposition, because it enables efficient FD checking
- a DP/BCNF decomposition does not always exist
- the BCNF decomposition algorithm does not produce always DP decompositions (even if they do exist)

Normalization

Making a choice

- Minimize the level of redundancy: BCNF
- Accept that the DP property may be lost

or:

- Allow a bit more redundancy (3NF in stead of BCNF)
- Enforce the DP property

3NF (just out of the blue)

Definition:
An attribute is *prime* if it is contained in a candidate key.

Definition:
A relation scheme is in 3NF if for each non trivial FD $X \rightarrow A$ the following condition holds:

X is a superkey *or* A is prime

3NF: another view

A relation scheme is in 3NF if

- it is in BCNF, or
- if all FDs that violate the BCNF property have this shape: $X \to A$ where X is not a key, but A is part of a key

Note that the 3NF requirement is a bit less demanding than the BCNF requirement

BCNF versus 3NF

Address table for a specific town:

ADDRESS		
Street	**Number**	**Zipcode**
Ooievaarspad	1	3403 AM
Ooievaarspad	2	3403 AM
Meerkoetweide	2	3403 AK
Meerkoetweide	4	3403 AK
Meerkoetweide	6	3403 AL
Meerkoetweide	8	3403 AL

BCNF versus 3NF

ADDRESS		
Street	**Number**	**Zipcode**
Ooievaarspad	1	3403 AM
Ooievaarspad	2	3403 AM
Meerkoetweide	2	3403 AK
Meerkoetweide	4	3403 AK
Meerkoetweide	6	3403 AL
Meerkoetweide	8	3403 AL

Street, Number → Zipcode
Zipcode → Street
ADDRESS is in 3NF
ADDRESS is not in BCNF (redundancy)

BCNF versus 3NF

StrZip	
Street	**Zipcode**
O'pad	3403 AM
M'weide	3403 AK
M'weide	3403 AL

NoZip	
Number	**Zipcode**
1	3403 AM
2	3403 AM
2	3403 AK
4	3403 AK
6	3403 AL
8	3403 AL

StrZip and NoZip are in BCNF

This decomposition is not DP

An operation `insert(6,'3403 AK')` on NoZip is incorrect!

Detecting this violation requires a join with StrZip

3NF-Algorithm

```
INPUT: a scheme $R$, a set FDs $F$
OUTPUT: a lossless DP 3NF-decomposition of $R$
METHOD:
    create a minimal cover $G$ from $F$;
    generate for each FD $X \rightarrow A_1, ..., A_n$ a scheme $(XA_1A_2..A_n)$;
        // this scheme has local key $X$
    if there is a scheme containing a global key $K$
    then you are finished;
    else add a global key as an extra relation scheme;
        // a global key is a key for $R$
```

INTERMEZZO

- We have $R(ABCDEFGHKL)$ with
  $\mathbf{F} = \{A \rightarrow D, B \rightarrow HA, GH \rightarrow AKL, CH \rightarrow BK, G \rightarrow BC\}$
- Minimal cover
  $\mathbf{G} = \{A \rightarrow D, B \rightarrow HA, G \rightarrow LBC, CH \rightarrow BK\}$
- 3NF scheme?

INTERMEZZO

- We have $R(ABCDEFGHKL)$ with
  $\mathbf{F} = \{A \rightarrow D, B \rightarrow HA, GH \rightarrow AKL, CH \rightarrow BK, G \rightarrow BC\}$
- Minimal cover
  $\mathbf{G} = \{A \rightarrow D, B \rightarrow HA, G \rightarrow LBC, CH \rightarrow BK\}$
- 3NF scheme:
  $(AD), (BHA), (GLBC), (CHBK)$

INTERMEZZO

- We have $R(ABCDEFGHKL)$ with $\mathbf{F} = \{A \rightarrow D, B \rightarrow HA, GH \rightarrow AKL, CH \rightarrow BK, G \rightarrow BC\}$
- Minimal cover $\mathbf{G} = \{A \rightarrow D, B \rightarrow HA, G \rightarrow BCL, CH \rightarrow BK\}$
- 3NF scheme:

  $(AD), (BHA), (GBCL), (CHBK)$
- Lossless 3NF scheme:

  $(AD), (BHA), (GBCL), (CHBK), (GEF)$

Overview normal forms

normal form	*feasible*	*complexity*
3NF + DP	always	polynomial
BCNF	always	polynomial
BCNF + DP	not always	NP-hard

Requirement 2:
you should strive for a high normal form (BCNF or 3NF)
Requirement 3:
you should strive for a DP decomposition

Beyond BCNF: 4NF

Entity (*skater*) with two unrelated multivalued attributes (*distance*, *coach*)

Speed_skating_long_track		
name	**distance**	**coach**
Ireen Wüst	1000	Gerard Kemkers
Ireen Wüst	1500	Gerard Kemkers
Ireen Wüst	3000	Gerard Kemkers
Ireen Wüst	5000	Gerard Kemkers
Michel Mulder	500	Gerard van Velde
Michel Mulder	500	Jurre Trouw
Michel Mulder	1000	Gerard van Velde
Michel Mulder	1000	Jurre Trouw
Sven Kramer	5000	Gerard Kemkers
...	...	...

(Inspiration: Olympic winter games 2014)

4NF decomposition

Intuitive decomposition: lossless!

name	distance
Ireen Wüst	1000
Ireen Wüst	1500
Ireen Wüst	3000
Ireen Wüst	5000
Michel Mulder	500
Michel Mulder	1000
...	...

name	coach
Ireen Wüst	Gerard Kemkers
Michel Mulder	Gerard van Velde
Michel Mulder	Jurre Trouw
Sven Kramer	Gerard Kemkers
...	...

4NF

Compare: *skater* won a *medal* on a *distance*

Speed_skating_long_track_medal		
name	**distance**	**medal**
Ireen Wüst	1000	Silver
Ireen Wüst	1500	Silver
Ireen Wüst	3000	Gold
Ireen Wüst	5000	Silver
Michel Mulder	500	Gold
Michel Mulder	1000	Bronze
...	...	...

Decomposition on (sname, distance) and (sname, medal) would *not* be lossless!

4NF: intuition

A relation r with scheme $R(XYZ)$ obeys the *multivalued dependency (MVD)* $X \twoheadrightarrow Y$

if it is the case that Y and Z each have some connection with X, but do not have anything to do with each other.

4NF: definition

A relation r with scheme $R(XYZ)$ obeys the *multivalued dependency (MVD)* $X \twoheadrightarrow Y$ if the presence of t_1 and t_2 guarantees the presence of t_3

	MVD		
	X	**Y**	**Z**
	…	…	…
t_1	x	y_1	z_1
	…	…	…
t_2	x	y_2	z_2
	…	…	…
t_3	x	y_1	z_2

4NF: definition

A relation r with scheme $R(XYZ)$ obeys the *multivalued dependency (MVD)* $X \twoheadrightarrow Y$ if the presence of t_1 and t_2 guarantees the presence of t_3

	MVD		
	X	**Y**	**Z**
	…	…	…
t_1	x	y_1	z_1
	…	…	…
t_2	x	y_2	z_2
	…	…	…
t_3	x	y_1	z_2

Note that de definition of the MVD is symmetric with respect to Y and Z. MVD's always show up in pairs $X \twoheadrightarrow Y, X \twoheadrightarrow Z$ within a scheme XYZ.

4NF: terminology in retrospect

About the concept of *multivalued dependency (MVD)*:
. . . the term *multivalued* **in***dependency (MVD)* would have been a better choice

. . . but such is life

4NF

Theorem:
Suppose we have a scheme $R(XYZ)$.

$X \twoheadrightarrow Y$
$\Leftrightarrow$
the decomposition $R_1(XY), R_2(XZ)$ is lossless

4NF

Observation:
Suppose we have a scheme $R(XYZ)$.

$X \rightarrow Y \Rightarrow$
the decomposition $R_1(XY), R_2(XZ)$ is lossless $\Rightarrow$
$X \twoheadrightarrow Y$

4NF

Observation:
Suppose we have a scheme $R(XYZ)$.

$$X \rightarrow Y$$

$$\Rightarrow$$

$$X \twoheadrightarrow Y$$

This seems counterintuitive, but note that an FD is a very special case of an MVD.
The MVD states that for a specific value x of X, every combination of values y for Y and z for Z occurring together with this x is a valid triple. Under $X \rightarrow Y$, this is true in a trivial way, because there is only one value y occuring together with this x.

4NF

Definition:
Suppose we have a scheme $R(XYZ)$.
We call an MVD $X \twoheadrightarrow Y$ trivial
if $Y \subseteq X$ or if $Z = \emptyset$

Definition:
Suppose we have a relation r over a scheme $R(XYZ)$;
r is in 4NF if each left side of a non trivial MVD in R is a superkey

Consequence:
A scheme $R(XYZ)$ with $X \twoheadrightarrow Y$ should be decomposed into
$R_1(XY)$ and $R_2(XZ)$

INTERMEZZO 4NF

Een E-commerce-bedrijf. Van klanten worden de volgende gegevens bijgehouden:

- clientid, naam, postcode, huisnr, gebdatum : spreekt voor zich
- in welke producttypes hebben klanten interesse getoond (media, sport, persoonlijke verzorging, ...)
- welke betalingsmethoden heeft de klant gebruikt (Ideal, creditcard, PayPal, factuur achteraf, ...)
- heeft de klant nog kortingscodes tegoed
- kortingscodes hebben een verloopdatum (expdatum)

Client (clientid, naam, postcode, huisnr, gebdatum, producttype, betalingsmethode, kortingscode, expdatum)

- identificeer de FDs
- identificeer de MVDs
- geef een 4NF decompositie

INTERMEZZO 4NF

Client (clientid, naam, postcode, huisnr, gebdatum, producttype, betalingsmethode, kortingscode, expdatum)

- clientid → naam, postcode, huisnr, gebdatum
- clientid ↠ producttype
- clientid ↠ betalingsmethode
- clientid ↠ kortingscode, expdatum

Decompositie: 4NF

- Client (clientid, naam, postcode, huisnr, gebdatum)
- ClientProductType (clientid, producttype)
- ClientBetaling (clientid, betalingsmethode)
- ClientKorting (clientid, kortingscode, expdatum)

Design: the big picture

- Apply ERD diagrams for domain modeling
- Generate a rough DB scheme from the ER model
- Apply normalization theory to refine your design
- Aim at 3NF/DP in the first step, and check for BCNF/4NF afterwards