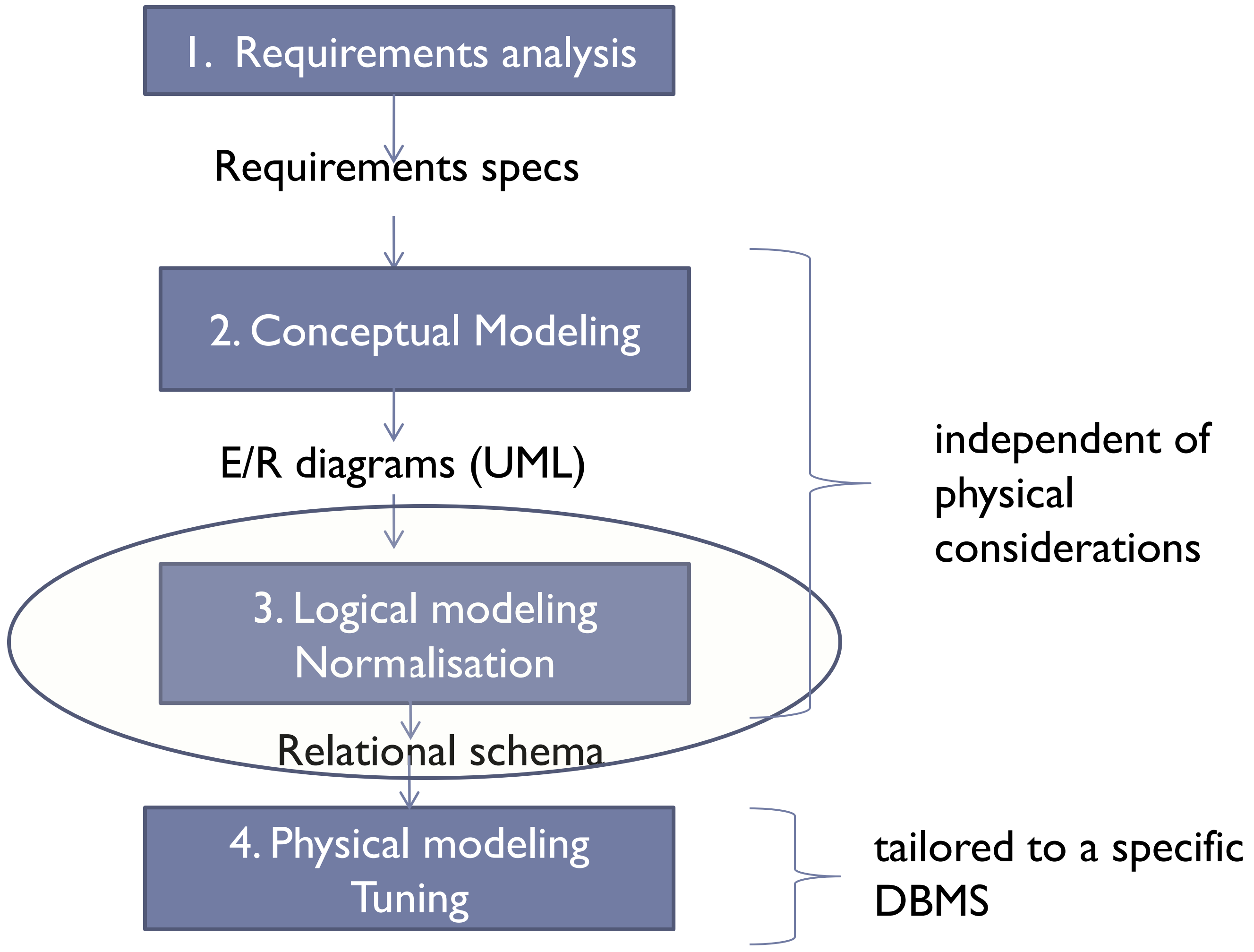
SELECT *lecture*
FROM **Databases**

WHERE INITCAP(chapter) = 'Relational Database Design'
AND TRIM(topic) = 'Database Normalization'

Mihaela Elena Breabăn

Relational Database Design Methodology



Outline

- Schema anomalies
- Functional dependences (revisited)
- Normal forms:
  - The 1st Normal Form (1NF)
  - The 2nd Normal Form (2NF)
  - The 3rd Normal Form (3NF)
  - Boyce-Codd Normal Form (BCNF)
- Multivalued dependencies (revisited)
- The 4th Normal Form (4NF)
- Denormalization

Normalization

- A theory for relational database design proposed by Codd between ’70-’74 based on relational algebra concepts, aiming at:
  - Eliminating possible anomalies when updating stored data
  - Minimizing schema redesign when new extensions are necessary
  - Avoiding unwanted query performance biases

Schema Anomalies
An example

- Designing a centralized data store about the national university admission process.
  - Data:
    - Universities – suppose they are uniquely identified by name
    - High schools – there may exist the same high school name in several cities
    - Candidates – may be uniquely identified by personal numerical codes (CNPs), have names, graduated high schools
    - Candidates apply to several universities
    - The candidates may have obtained awards at national or international contests at one or several study disciplines (subjects), bringing them advantages in the selection step for specific faculties

Possible schema:

```
Candidates(CNP, studFName, studLName, univName, hsName, hsCity,
   awdSubject)
```

Possible application:

Ioana having CNP **2810605222111** studied at **Negruzzi** highschool in **Iaşi**, applies for **Cuza**, **Asachi** and **Babes-Bolyai universities**, received medals/awards at two national contests at physics and maths.

How may tuples are necessary to be inserted in the Candidates table to store all the information about Ioana ?

Design anomalies

- Redundancy
  - Repeated information: how many times (**‘2810605222111’, ‘Ioana’**) or (**2810605222111, ‘physiscs’)**, **or** (**2810605222111, ‘maths’), or** (**2810605222111, ‘Cuza’)** in the discussed example?
- Anomalies at updates
  - One of the universities observes that Ioana’s name was misspelled, or one of the awarded subjects was misspelled. How ensure correct and complete updates?
- Anomalies at deletes
  - Some subjects are no more relevant for the admission process. How eliminate them?

Anomaly-free schema
An example

- Designing a centralized data store about the national university admission process.
  - Data:
    - Universities – suppose they are uniquely identified by name
    - High schools – there may exist the same high school name in several cities
    - Candidates – may be uniquely identified by personal numerical codes (CNPs), have names, graduated high schools
    - Candidates apply to several universities
    - The candidates may have obtained awards at national or international contests at one or several study disciplines (subjects), bringing them advantages in the selection step for specific faculties

```
Candidates(CNP, studFName, studLName)
Highschools(hsCode, hsName, hsCity)
Graduates(CNP, hsCode)
Applicants(CNP, univName)
AwardedSubjects(CNP, awdSubject)
```

Quiz

- Store information about the courses graduated by each student
  - Students are identified by IDs assigned by the faculty and have (not unique) names
  - Courses are uniquely identified by an ID and have (not unique) names
  - Students enroll for courses in a specific year and receive a grade

- Which is the recommended schema?
  - Study(sID, sName, cID, cName, year, grade)
  - Courses(cID, cName, year), Study(sID,cID, grade)
  - Students(sID, sName), Courses(cID, cName), Grades(sID, cID, year, grade)
  - Students(sID, sName), Courses(cID, cName), Grades(sName, cName, year, grade)

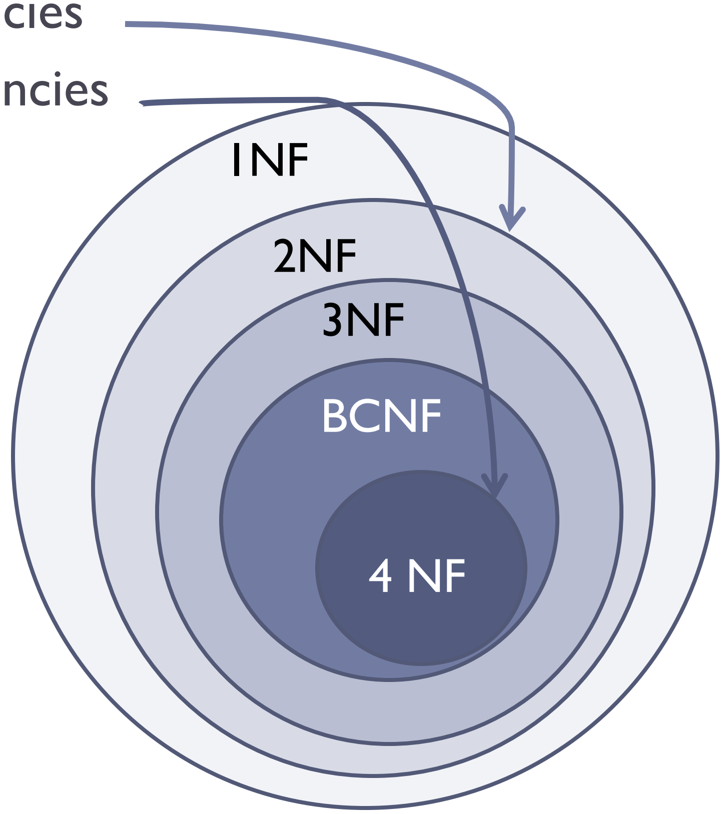
Design by decomposition

- Start with "mega-relations" containing all the information
- Decompose in smaller relations which can be used to reconstruct the original information

- Can be automated:
  - Input: Mega-relations + *properties/constraints of the data*
  - Decomposition is performed based on the *properties/constraints*
  - Output: a set of relations satisfying some *normal forms* which
    - ***Have no anomalies***
    - ***Do not loose information***

Properties/constraints and Normal forms

- Properties/constraints:
  - Functional dependencies
  - Multivalued dependencies
- Normal forms:



Functional dependencies (revisited)

- Rules that restrict/reduce the set of possible tuples in a relation

$$\boxed{\begin{array}{l} X \rightarrow Y \text{ iff} \\ \forall t_1, t_2 \in r, t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y] \end{array}}$$

r – relation over a set of attributes U
X,Y – subsets of U

Why *functional?*

Functional dependencies - Example

- `Relational schema:`

```
HSGraduates(CNP, name, address, hsCode, hsName, hsCity,
  gradScore)

Candidates(CNP, univName, univCity, date, faculty)
```

What functional dependences would you formulate for HSGraduates?

What functional dependences would you formulate for Candidates?

Explain what constraint is imposed by {CNP, univName $\rightarrow$date}

Quiz

- R(A,B,C,D,E,F)
- $ABC \rightarrow D$
- $DE \rightarrow F$
- Every attribute in {A,B,C,E} has at most 3 distinct values.

- What is the **maximum** number of distinct values for attribute F? (3,9,27,81?)

Functional dependencies - inference Armstrong's axioms (revisited)

- Reflexivity (FD1,A1)
  - Trivial dependencies
- Decomposition (FD6, A2.1)
- Union (FD5,A2.2)
- Transitivity (FD3,A3)

- Completeness and soundness:

  Let F be a set of functional dependencies. A functional dependency $X \to Y$ is a consequence of F if and only if

  $$F \mid\!-_A X \to Y$$

Functional dependencies vs. superkeys

- Functional dependencies – in practice!
  - Given an instantiation of the subset of attributes X of r, there is only one possible instantiation of the subset of attributes Y in r.
    - Are formulated based on knowing the logic of data;
    - *We specify only a minimal set of functional dependencies that can serve to obtain all the dependencies (as consequences) satisfied by r.*

- Superkeys
  - Given an instantiation of the subset of attributes X of r, there does not exist two distinct tuples equal on X => there is only one possible instantiation of attributes U-X in r.

Superkeys are a special case of functional dependencies

X is a superkey iff $X \rightarrow U$

Closures

- *The closure of a set of functional dependencies (f.ds.)* Σ, denoted Σ^+:
  - Contains Σ and all the functional dependencies which are consequences of Σ
- *The closure of a set of attributes* X under the set of *f.ds.* Σ, denoted X_Σ^+
  - Contains all attributes B for which $X \to \mathrm{B} \in \Sigma^+$

<u>How compute the closure of X under set F of f.ds.?</u>

Closures and keys

- Given a relational schema R(U) and a set Σ of f.ds. satisfied by R,

set X of attributes is a key for R iff $X_{\Sigma}^{+} = U$ and

$$\forall X' \subset X, X'^{+}_{\Sigma} \neq U$$

- Example

Relational schema:

`HSGraduates(CNP, name, address, hsCode, hsName, hsCity,gradScore)`

Functional dependencies:

```
CNP → name, adress
CNP -> gradScore
hsCode → hsName, hsCity
```

The Key: **`{CNP, hsCode}`**

<u>Given a relational schema and a set of f.ds. ,</u>
<u>how identify all the keys?</u>

Quiz

- Relational schema:

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

- Functional dependencies:

$AB \to C$

$AE \to D$

$D \to B$

- Compute all the keys for R

Non-prime attributes

- Prime attribute
  - There must exist at least one key that contains it
- Non-prime attribute
  - There is no key to contain it

- Example
  - *R1(A, B, C, D)*
  - $F = \{AB \to C, B \to D, BC \to A\}$.
  - *AB and BC are the keys for F, therefore A, B, C are prime attributes*
  - *D is a non-prime attribute.*

Full functional dependencies

- Let R be a relational schema over attributes U, and F be a set of f.ds.

A functional dependency $X \rightarrow B \in F^+$ ($X \subset U$, $B \in U$, $B \notin X$) is a full dependency of R (or B is fully dependent of X under F), if there does not exist a subset $X' \subset X$, such that $X' \rightarrow B \in F^+$.

- Example
  - *R(A, B, C, D)*
  - $F = \{AB \rightarrow C, B \rightarrow D, BC \rightarrow A\}$.
  - *All functional dependencies of F are full dependences.*
  - $AB \rightarrow D \in F^+$ *is not a full dependency*

Transitively dependent attributes

- Let R be a relational schema over attributes U, and F a set of functional dependencies. An attribute B in U is *transitively dependent of* X ($X \subset U$, $B \notin X$), if there exists $Y \subset U$ such that:
  - $B \in U-Y$,
  - $X \rightarrow Y \in F^+$ and $Y \rightarrow B \in F^+$ - non-trivial dependencies
  - but $Y \rightarrow X \notin F^+$.

1NF

A relational schema

is in the 1st NF

if the domain of each attribute has elementary/atomic values and for any instantiation of the schema, the each attribute contains only a single value from that domain.

Generally, an atomic value is a value for which we do not require, at any time, parts of it, but we always use it as a whole.

2NF

A relational schema R in 1NF, which satisfies a set F of f.ds.
is in the 2nd NF
if every non-prime attribute of R is fully dependent of every key of R.

- Obs:
  A relation that does not have multivalued keys is in 2NF.

2NF Example

- *R(A, B, C, D)*
- *$F = \{AB \rightarrow C, B \rightarrow D, BC \rightarrow A\}$.*

Keys: {AB, BC}

D is a non-prime attribute and $B \rightarrow D \in F^+$

$\Rightarrow$ *D is not fully dependent of AB, nor BC*

$\Rightarrow$ *R (under F) does not satisfy 2NF.*

Quiz

- Relational schema:

HSGraduates(CNP, name, awdSubject)

- Rules

A highschool graduate, uniquely identified by CNP, has assigned only one name but received awards at several subjects

Prove that HSGraduates is not in 2NF.

What anomalies may appear because not satisfying 2NF?

Quiz

- Relational schema:

Olympics(contest, year, CNP, name)

- Rules

In a given year there is only one winner at a given contest. The winner is identified by CNP and has assigned a single name.

Is Olympics in 2NF?

3NF

A relational schema R in 1NF, which satisfies a set F of f.ds.

is in the 3rd NF

if it satisfies 2NF and

every non-prime attribute of R is NOT transitively dependent of any key of R.

3NF Example

- Relational schema

$R(O, S, C)$

- Functional dependencies

$F = \{OS \rightarrow C, C \rightarrow O\}$

Keys=$\{OS, SC\}$

All attributes are prime

$\Rightarrow$ *R is in 2NF and 3NF.*

Quiz

- Relational schema:

Olympics(contest, year, CNP, name)

- Rules

In a given year there is only one winner at a given contest. The winner is identified by CNP and has assigned a single name.

Show that Olympics is not in 3NF.

What data inconsistencies may appear?

Quiz

- Relational schema:

Candidates(CNP, univName, date, faculty)

- Rules

A candidate identified by CNP may apply at any university (identified by univName) only once (at only one date) and at only one of its faculties

Universities have application dates which do not overlap

- Rules formalization:

  CNP, univName $\rightarrow$ date, faculty

  date $\rightarrow$ univName

Is Candidates in 3NF?

Boyce Codd NF

A relational schema R in 1NF, which satisfies a set F of f.ds.

is in BCNF

if for any non-trivial f.d. $X \rightarrow A \in F^{+}, X$ *is a (super)key of R.*

Every relational schema that satisfies BCNF, also satisfies 3NF.

PROVE IT!

Quiz

- Relational schema:

Candidates(CNP, univName, date, faculty)

- Rules

A candidate identified by CNP may apply at any university (identified by univName) only once (at only one date) and at only one of its faculties

Universities have application dates that do not overlap

- Rules formalization:

  CNP, univName $\rightarrow$ date, faculty

  date $\rightarrow$ univName

Candidates is not in BCNF.

Inconsistencies?

Designing DBs that satisfy BCNF

- A database schema is in BCNF if each of its relational schemas satisfies BCNF.

- Design by decomposition:
  - Input: a mega-relational schema containing all attributes and a set of functional dependencies
  - Output: a set of relational schemas, each satisfying BCNF

- By applying natural join operators between resultant tables/relations, the original mega-relation will be retrieved

Schema decomposition

- Let $R[A_1, A_2, \ldots, A_n]$ be a relational schema.
  $\rho = \{R_1, \ldots, R_k\}$, where $R_i[A_{i1}, \ldots, A_{ih_i}]$ is a *decomposition* of R if

$$\bigcup_{i=1}^{k} \bigcup_{j=1}^{h_i} A_{ij} = \{A_1, \ldots, A_n\}$$

- ρ is said to be a *lossless-join decomposition* of R if for any relation r over R,
  $r = r[R_1] \bowtie \ldots \bowtie r[R_k]$ – meaning that r can be obtained by applying the natural join over ρ.

Schema decomposition Example

- Relational schema:

HSGraduates(CNP, name, address, hsCode, hsName, hsCity, gradScore)

- Two schema decompositions:

ρ_1={S_1(CNP, name, address, $\underline{\text{hsCode}}$, gradScore),

S_2($\underline{\text{hsCode}}$, hsName, hsCity)}

ρ_2={S_1(CNP, $\underline{\text{name}}$, adress, hsCode, $\underline{\text{hsName}}$, hsCity),

S_2($\underline{\text{name, hsName}}$, gradScore)}

- ρ1 – is a lossless-join decomposition
- ρ2 – is NOT a lossless-join decomposition

Lossless-join decompositions

- Theoreme
  - Let ρ = (R_1, R_2) be a decomposition of R and F a set of f.ds.,;
    ρ is a lossless-join decomposition of R over F iff
    $R_1 \cap R_2 \rightarrow R_1 - R_2 \in F^+$ or
    $R_1 \cap R_2 \rightarrow R_2 - R_1 \in F^+$.

  (By schema intersection $R_1 \cap R_2$ *we denote attribute sets intersection)*

- Example
  - Relational schema: R (A,B,C)
  - Functional dependencies: F = {A → B}.

  - Decomposition: ρ_1 = (R_1(A,B), R_2(A,C))

  AB ∩ AC = A, AB-AC = B, $A \rightarrow B \in F^+$

  ⇒ ρ_1 is a lossless-join decomposition of R over F.

  - Decomposition: ρ_2 = (R_1(A,B), R_2(B,C)).

  AB ∩ BC = B, AB-BC = A, $B \rightarrow A \notin F^+$,

  AB ∩ BC = B, BC-AB = C, $B \rightarrow C \notin F^+$,

  ⇒ ρ_2 is NOT a lossless-join decomposition of R over F.

Lossless-join decomposition in BCNF

- Input:
  - Relational schema R and set of f.ds. F.
- Output:
  - Lossless-join decomposition $\rho = (R_1,\dots,R_k)$ of R over F, where (R_i, F_i) is in BCNF

$\forall$ i = 1, k.

- Initialization
  - $\rho = R = R_1$
  - Compute the keys for R
- Repeat
  - Let R_i be a relational schema in ρ and F_i its set of functional dependencies (over attributes in R_i) such that (R_i, F_i) is NOT in BCNF.
  - Identify $X \rightarrow A \in F_i^+, A \notin X$ and X NOT a superkey.
  - Build $S_1 = X \cup \{A\}$, $S_2 = Ri - A$
  - Replace R_i in ρ with S_1, S_2.
  - Compute F_{S1}^+ F_{S2}^+ and keys for S_1, S_2
- Until all (R_i,F_i) in R are in BCNF.

Lossless-join decomposition in BCNF Example

- Relational schema:

```
HSGraduates(CNP, name, address, hsCode, hsName,
 hsCity, gradScore)
```

- Functional dependencies:

```
CNP → name, address, gradScore
hsCode → hsName, hsCity
```

- Lossless-join decomposition in BCNF:

```
{R1(hsCode,hsName,hsCity),
 R2(CNP,name,address, gradScore),
 R3(CNP,hsCode)}
```

For a given relation R is it possible to exist several lossless-join decompositions in BCNF?

Does the decomposition alg. in BCNF guarantee a good schema?

- Can we reconstruct the original relation? YES
- Does it ALWAYS eliminate redundancy? NO

  `Candidates(CNP, univName, awdSubject)`

  - Any f.ds.? NO
  - Keys? The whole set of attributes
  - In BCNF? YES
  - A good schema? …

Multivalued dependencies

- Rules that generate tuples

$X \rightarrow\rightarrow Y$ if

$\forall t_1, t_2 \in r, t_1[X] = t_2[X]$, there exists $t_3, t_4 \in r$ such that

(i) $t_3[X] = t_1[X], t_3[Y] = t_1[Y]$ and $t_3[Z] = t_2[Z]$

(ii) $t_4[X] = t_2[X], t_4[Y] = t_2[Y]$ and $t_4[Z] = t_1[Z]$

r – relation over attributes U
X,Y – subsets of U
Z=U-XY

- If r satisfies $X \rightarrow Y$, it also satisfies $X \rightarrow\rightarrow Y$

Multivalued dependencies
Example

- Relational schema:

  `Candidates(CNP, univName, awdSubject)`

- Rule:

  `The awards obtained by the candidates must  be visible for all universities where the candidate applies`

- Rule formalization:

  $CNP \twoheadrightarrow univName$

Multivalued dependencies
Extended example

- Relational schema:

`Candidates(CNP, univName, date, faculty, awdSubject)`

- Rules:

  The awards are selectively introduced per university

  A candidate may apply at several faculties within a university but must do it in the same day

  All awards declared at a given university must be visible for all the faculties where the candidate applies within the university

- Rules formalization:

  CNP,univName→date

  CNP,univName,date→→faculty

Quiz

- Let R(A,B,C) be a relational schema and $A \rightarrow\rightarrow B$ a multivalued dependency satisfied by R
- A takes at least 3 distinct values and every value of A is associated with at least 4 distinct values of B and at least 5 distinct values of C.
- What is the **minimum** number of tuples in any relation r - instance of R?

Multivalued dependencies – inference Some rules

- Trivial dependencies:
  - if $Y \subseteq X$ then $X \rightarrow\rightarrow Y$
  - if $X \cup Y = U$ then $X \rightarrow\rightarrow Y$

- Complementation
  - if $\mathrm{X} \rightarrow\rightarrow \mathrm{Y}$, $XYZ = U$ and $\mathrm{Y} \cap \mathrm{Z} \subseteq \mathrm{X}$ then $X \rightarrow\rightarrow Z$

- Transitivity (!=d.fs.)
  - if $\mathrm{X} \rightarrow\rightarrow \mathrm{Y}, Y \rightarrow\rightarrow \mathrm{Z}$ then $X \rightarrow\rightarrow Z - Y$

- Decomposition

if $\mathrm{X} \rightarrow\rightarrow \mathrm{Y}$and $X \rightarrow\rightarrow \mathrm{Z}$ then $X \rightarrow\rightarrow Z \cap Y, X \rightarrow\rightarrow Z - Y, X \rightarrow\rightarrow Y - Z$

4NF

A relational schema R in 1NF, which satisfies a set F of fds. and a set MV of mv.ds.

is in the 4th NF

if for any non-trivial mv.d. $X \rightarrow\rightarrow A \in MV^{+}, X$ *is a (super)key of R.*

Any relational schema in 4NF is in BCNF.

Prove it!

Lossless-join decomposition in 4NF

- Input:
  - Relational schema R, set of f.ds. F. , set of mv.ds. MV
- Output:
  - Lossless-join decomposition $\rho = (R_1,\ldots,R_k)$ of R over F and MV, where (R_i, F_i, MV_i) is in 4NF $\forall$ i = 1, k.

- Initialization
  - $\rho = R = R_1$
  - Compute (F^+, MV^+) over R and the keys for R
- Repeat
  - Let R_i be a relational schema in ρ and (F_i, MV_i) its set of f.ds. and mv.ds. (over attributes in R_i) such that (R_i, F_i, MV_i) is NOT in 4NF.
  - Identify $X \rightarrow\rightarrow A \in MV_i^+$,nontrivial, X NOT a superkey.
  - Build $S_1 = X \cup \{A\}$, $S_2 = Ri - A$
  - Replace R_i in ρ with S_1, S_2.
  - Compute F_{S1}^+ , F_{S2}^+ , MV_{S1}^+ , MV_{S2}^+ and keys for S_1, S_2
- Until all (R_i, F_i, MV_i) in R are in 4NF.

Lossless-join decomposition in 4NF Example

- Relational schema:

  `Candidates(CNP, univName, awdSubject)`

- Dependencies:

  `CNP→→univName`

- Decomposition:
  - $\rho = \{A_1(\texttt{CNP,univName}), A_2(\texttt{CNP,awdSubject})\}$

  Number of tuples: `u*h => u+h`

Lossless-join decomposition in 4NF Extended example

- Relational schema:

  `Candidates(CNP,univName,date,faculty,awdSubject)`
- Dependencies:

  `CNP,univName→date`

  `CNP,univName,date→→faculty`

- Decomposition:

  ρ ={A_1`(CNP,univName,date),`

  A_2`(CNP,univName,faculty),`

  A_3`(CNP,univName,awdSubject)`}

Normalization - shortcomings Example

- Relational schema:

  `Candidates(CNP,univName,date,faculty)`
- Dependencies:

  CNP, univName→date, faculty

  date→univName

IS {A1(date,univName),A2(CNP,date,faculty)} in 4NF a better schema?

Normalization – shortcomings

- Over-decomposition
- Overloaded queries

- Possible solution: ***denormalization***

Bibliography

- Hector Garcia-Molina, Jeff Ullman, **Jennifer Widom**: *Database Systems: The Complete Book,* Prentice Hall; 2nd edition (June 15, 2008)

- In Romanian: Victor Felea: *Baze de date relationale. Dependente*. Editura Universitatii "Al.I.Cuza" Iasi, 1996

- *Some of the examples are **adapted** from the Databases course at Stanford (author Jennifer Widom)*