Third Normal Form 2

SAPIENZA
UNIVERSITÀ DI ROMA

The content of the slides is authored by Prof. Maria de Marsico, Sapienza University of Rome
Part of the material is taken from the book: Atzeni, Ceri, Paraboschi, Torlone Basi di dati - Modelli e linguaggi di interrogazione , McGraw-Hill, 2002

Partial dependencies

- Curriculum (**Matr**, TaxC, SurN, Name, DateB, Town, Prov, **C#**, Tit, Doc, DateP, Vote)

any matriculation number identifies a particular surname:

- Matr → SurN

so, any pair consisting of a matriculation number and a course code identifies a surname: Matr C# → SurN

- the depenency Matr C# → SurN is a consequence of the dependency Matr → SurN
- Matr → SurN is called **partial dependency**

Transitive dependencies

- Student (**Matr**, TaxC, SurN, Name, DateB, Town, Province)

- a matriculation number identifies only one town of birth: Matr→Town
- a town is located in only one province: Town→Prov

conclusion:

- a matriculation number corresponds to only one province: Matr→Prov

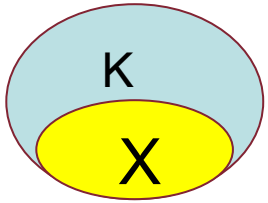
- the functional dependency Matr→Prov is a consequence of the two functional dependencies Matr→Town and Town→Prov
- Town → Prov is called **transitive dependency**

Formal definitions

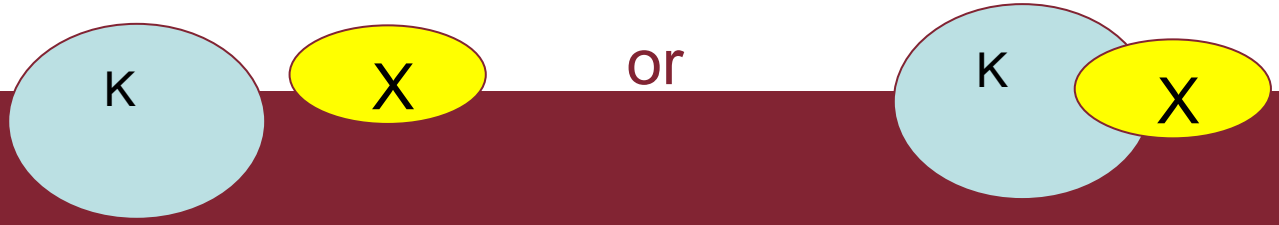
Definitions

- let R be a relation schema and F a set of functional dependencies on R

- A **partially depends on** a key K if $\exists X \subset R$ such that $X \rightarrow A \in F^+$ with $A \notin X$ and such that $X \subset K$ and A is not part of a key



- A **transitively depends on** a key K if $\exists X \subset R$ such that $K \rightarrow X \in F^+$ with $A \notin X$ and $X \rightarrow A \in F^+$ and X is not a key and A is not part of a key



Alternative definition

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- given a schema R and a set of functional dependencies F, R is in 3NF if and only if there are no attributes that partially or transitively depend on a key

The two definitions are equivalent

Theorem.
Let *R* be a relation schema and *F* a set of functional dependencies on *R*. A schema *R* is in 3NF **if and only if** neither partial dependencies nor transitive dependencies exist in *R*

What we want to achieve

- we have seen that a 3NF schema has **good properties** that make it preferable to one that is not in 3NF

- a goal to keep in mind when designing a database is to produce a schema in which **every** relation is in 3NF
- normally, in the **conceptual design** phase, the Entity-Relationship model is used
- in the conceptual design phase the **concepts** to be represented in the database are identified

What we want to achieve

- if the identification work is done properly, the relational schema that can be derived automatically with **appropriate rules**, is in 3NF

- if, however, after this process we obtain a schemas that are **not in 3NF,** we can **decompose them, so they are in 3NF**

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- a schema that is not in 3NF can be **decomposed** in **multiple ways** into a set of schemas in 3NF
- for example, the schema $R = ABC$ with the set of functional dependencies $F\{A \rightarrow B, B \rightarrow C\}$ is not in 3NF, due to the presence in F^+ of the transitive dependency $B \rightarrow C$ (the key is A)
- R can be decomposed into:
  - $R1=AB$ with $A \rightarrow B$ and
  - $R2=BC$ with $B \rightarrow C$
  - **or**
  - $R1=AB$ with $A \rightarrow B$ and
  - $R2=AC$ with $A \rightarrow C$
- both schemas **are** in 3NF, however the second solution **is not satisfactory**

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- if we consider two **legal** instances of the obtained schemas:

R1

A	B
a1	b1
a2	b1

R2

A	C
a1	c1
a2	c2

- the instance of the original *R* schema that I can reconstruct from this (the only way is to reconstruct it by doing a natural join!) is:

R

A	B	C
a1	b1	c1
a2	b1	c2

- **BUT** it is not a legal instance of R, since it **does not satisfy** the functional dependence $B \rightarrow C$

we want to preserve ALL dependencies in F^+

Example

let us consider the schema *R = {Matriculation, Town, Province}* with the set of functional dependencies:

F={Matriculation→Town, Town→Province}

the schema is not in 3NF due to the presence in F^+ *of* the **transitive dependence** Town→*Province (*the key is *Matriculation and Province* transitively depends on *Matriculation)*

R can be decomposed into:

R1(Matriculation, Town) with Matriculation → *Town*

R2(Town, Province) with Town → *Province*

or

R1(Matriculation, Town) with Matriculation → *Town*

R2(Matriculation, Province) with Matriculation → *Province*

both schemas **are** in 3NF, but the second solution **is not satisfactory**

What we want to achieve - 3NF is not enough

SAPIENZA UNIVERSITÀ DI ROMA

- consider the **legal** instances of the obtained schemas:

R1

Matriculation	Town
501	Tivoli
502	Tivoli

R2

Matriculation	Province
501	Rome
502	Rieti

the instance of the original *R* schema that we can reconstruct from this with a natural join is:

R

Matriculation	Town	Province
501	Tivoli	Rome
502	Tivoli	Rieti

- **but** it is not a legal instance of R, since it **does not satisfy** the functional dependence Town→*Province*
- **clearly there was an error in the data, but we could not detect it**

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- we now consider the schema $R=ABC$ with the set of functional dependencies $F=\{A \rightarrow B, C \rightarrow B\}$ (the schema is not in 3NF due to the presence in F^+ of the partial dependencies $A \rightarrow B$ and $C \rightarrow B$, since the key is AC)
- this schema can be decomposed into:
  - $R1=AB$ with $A \rightarrow B$
  - $R2=BC$ with $C \rightarrow B$
- the resulting schemas, even if they **preserve all dependencies in F^+**, is still not satisfactory.

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- consider the **legal** instance of *R:*

R

A	B	C
a1	b1	c1
a2	b1	c2

the two facts *(a1,b1,c1)* and *(a2,b1,c2)* are true and not others

- we decompose it by obtaining:

R1

A	B
a1	b1
a2	b1

R2

B	C
b1	c1
b1	c2

What we want to achieve - 3NF is not enough

- if we now compute the natural join we obtain:

R

A	B	C
a1	b1	c1
a2	b1	c2
a1	b1	c2
a2	b1	c1

these tuples did not exist in the original instance!

- **we must ensure that the decomposition and the following natural join does not result in any loss of information**

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- consider the schema:
- *R={EmployeeID, ProjectID, Manager}* with the set of functional dependencies :
- *F={EmployeeID→ProjectID, Manager→ProjectID}*
- a project can have multiple managers but each manager has only one project, and an employee on a project reports to only one manager
- the schema is not in 3NF due to the presence in F^+ of the partial dependencies *EmployeeID→ProjectID* and *ProjectID→Manager*, since the key is (*EmployeeID*, *Manager*)
- the schema can be decomposed into:
  - *R1 = {Matriculation, Project}* with *EmployeeID→ProjectID* and
  - *R2 = {Project, Boss}* with *Manager→ProjectID*
- such a schema, **while preserving all dependencies in F^+**, is not satisfactory

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- consider the **legal instance** of *R:*

R

EmployeeID	ProjectID	Manager
501	30	E1
502	30	E2

only the two facts *(501,30,E1)* and *(501,30,E2)* are true!

- based on the given decomposition, this instance decomposes into:

R1

EmployeeID	ProjectID
501	30
502	30

R2

ProjectID	Manager
30	E1
30	E2

What we want to achieve - 3NF is not enough

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- ...and instead if you join the two legal instances resulting from the decomposition you get

R

EmployeeID	ProjectID	Manager
501	30	E1
502	30	E2
501	30	E2
502	30	E1

tuples unrelated to the reality of interest

What we want to achieve - 3NF is not enough

- in conclusion, two other requirements of the decomposed schema must be kept in mind when decomposing a schema to obtain one in 3NF:
  - we must **preserve the functional dependencies** that apply to each legal instance of the **original schema**
  - we must allow to **reconstruct by natural join** every **legal instance of the original schema** (without adding any extra information)

Boyce-Codd Normal Form

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

the 3NF is not the most restrictive normal form that can be achieved
there exist others, including the Boyce-Codd Normal Form

Definition: a relation is in Boyce-Codd Normal Form (BCNF) when **every determinant** in it **is a superkey (recall that a key is also a superkey)**

a relation that respects Boyce-Codd Normal Form is **also** in 3rd Normal Form, but the opposite is not true

Example

- consider a schema describing the allocation of operating rooms in a hospital
- the operating rooms are booked, day by day, at scheduled times, to perform operations on patients by the hospital's surgeons
- during a day, an operating room is **always** occupied by the same surgeon who performs several operations at different times.
- knowing the Patient and Date, we also know: time of surgery, surgeon, and operating room
- **schema**: Interventions = (Patient, Date, Time, Surgeon, Room)
- based on the above description, functional dependencies apply in the Interventions schema:
  - Patient, Date → Time, Surgeon, Room
  - Surgeon, Date, Time → Patient, Room
  - Room, Date, Time → Patient, Surgeon
  - Surgeon, Date → Room
- there are three keys: {Patient, Date}, {Surgeon, Date, Time}, {Room, Date, Time}

Example (continued)

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- Patient, Date → Time, Surgeon, Room
- Surgeon, Date, Time → Patient, Room
- Room, Date, Time → Patient, Surgeon
- Surgeon, Date → Room

- K1 = Patient, Date
- K2 = Surgeon, Date, Time
- K3 = Room, Date, Time

- whichever primary key we choose, e.g. {Patient, Date}, the determinants in the first 3 functional dependencies are sets of attributes that can perform the key function and thus BCNF is definitely not violated in these cases
- in contrast, BCNF **is not** satisfied by the fourth functional dependency that has a set of **non-key** attributes as determinant.It follows that Interventions is not in BCNF
- **but**
- Interventions **is** in 3FN, as the fourth functional dependency does not violate the definition, as the attribute Room is an attribute that is part of the key {Room, Date, Time} and therefore is **prime**

Example (continued)

- consequence: Interventions, although in 3NF, has a certain redundancy in the data that can create problems in the updating phase
- if, for any reason, we have to change the operating room used by a surgeon on a certain date, we would have to update several rows: for example, to move Romano from Room2 to Room3, we have to modify two rows of the ttable:

Interventions

Patient	Date	Time	Surgeon	Room
Bianchi	25/10/2005	8.00	De Bakey	Sala1
Rossi	25/10/2005	8.00	Romano	Sala2
Negri	26/10/2005	9.30	Veronesi	Sala1
Viola	25/10/2005	10.30	De Bakey	Sala1
Verdi	25/10/2005	11.30	Romano	Sala2

Example (continued)

the Interventions schema can be normalized, resulting in the two schemas:
Occupation (Surgeon, Date, Room)
Interventions (Patient, Date, Time, Surgeon)
the attribute Room is removed from Interventions and appears in a new table whose key is the determinant of the functional dependency that did not comply with BCNF

Interventions

Patient	Date	Time	Surgeon
Bianchi	25/10/2005	8.00	De Bakey
Rossi	25/10/2005	8.00	Romano
Negri	26/10/2005	9.30	Veronesi
Viola	25/10/2005	10.30	De Bakey
Verdi	25/10/2005	11.30	Romano

Occupation

Surgeon	Date	Room
De Bakey	25/10/2005	Sala1
Romano	25/10/2005	Sala2
Veronesi	26/10/2005	Sala1

Problem

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

suppose you want to keep track of patients who need to undergo multiple surgeries, in different departments, for the treatment of more complicated pathologies
a relation representing this requirement is shown in the table below:

MultipleSurgery

Patient	Department	Surgeon
Rossi	Cardiac Surgery	De Bakey
Rossi	General Surgery	Romano
Bianchi	General Surgery	Romano
Bianchi	Oncological Surg.	/eronesi
Verdi	General Surgery	ızetta

each tuple in MultipleSurgey associates a patient with the surgeon who operated them, and the department in which the surgery took place
the following functional dependencies apply:
Surgeon → Department
Patient, Department → Surgeon
Patient, Department is the key and the first dependency violates BCNF

Problem (continued)

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

let's try to proceed as before:

Surgeons

Surgeon	Department
De Bakey	Cardiac Surgery
Romano	General Surgery
Veronesi	Oncological Surg.
Lanzetta	General Surgery

Patients

Patient	Surgeon
Rossi	De Bakey
Rossi	Romano
Bianchi	Romano
Bianchi	Veronesi
Verdi	Lanzetta

The decomposition does not preserve the second of the two functional dependencies
if, for example, one wanted to record the (incorrect) fact that patient Bianchi was operated by Lanzetta in the Cardiac Surgery department, adding the tuple ("Bianchi", "Lanzetta") would be allowed.
Only when trying to reconstruct the data from the MultipleSurgery relation with a natural join between Patients and Surgeons we would obtain the n-uple:
("Bianchi", "Lanzetta", "Chir. Generale") highlighting the error in the data, as ("Bianchi", "general Surgery") should be associated with ("Romano")

Conclusion

it may not be possible to decompose a schema into BCNF sub-schemas while preserving all dependencies

but

it is **always** possible to do that with the 3NF

so

in the following, we will consider only the 3NF