4 – DATABASE DESIGN THEORY AND NORMALIZATION

DATABASES

BASES DE DADOS

Licenciatura em Engenharia Informática (L-EI)

2019-2020

Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Jacinto Estima
jacinto.estima@universidadeeuropeia.pt

OUTLINE

3 - DATABASE DESIGN THEORY AND NORMALIZATION

- Informal Design Guidelines for Relation Schemas
- Functional Dependencies
- Normal Forms
- Multivalued Dependency and Fourth Normal Form
- Join Dependencies and Fifth Normal Form

INFORMAL DESIGN GUIDELINES FOR RELATION SCHEMAS

- The implicit goals of the design activity are ***information preservation*** and ***minimum redundancy***
- Four informal guidelines that may be used as measures to determine the *quality of relation schema design*:
  - Making sure that the semantics of the attributes is clear in the schema
  - Reducing the redundant information in tuples
  - Reducing the NULL values in tuples
  - Disallowing the possibility of generating spurious tuples

FUNCTIONAL DEPENDENCIES

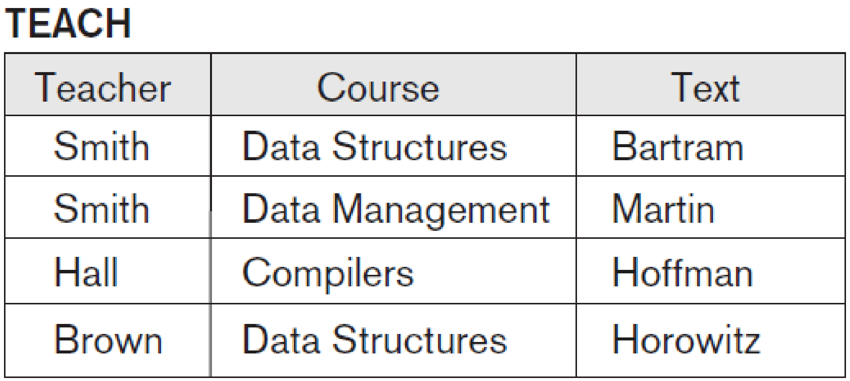
- A **Functional Dependency (FD)** is a constraint between *two sets of attributes* from the database
- Consider a *hypothetical scenario* where we have an entire database as being described by a single universal relation schema *R = {A1, A2, … , An}*
- Definition

A **functional dependency**, denoted by *X* → *Y*, between two sets of attributes *X* and *Y* that are subsets of *R* specifies a *constraint* on the possible tuples that can form a relation state *r* of *R*. The constraint is that, for any two tuples *t1* and *t2* in *r* that have *t1[X] = t2[X]*, they must also have *t1[Y] = t2[Y]*

- *The values of the Y depend on, or are determined by, the values of the X*
- *the values of the X uniquely (or functionally) determine the values of the Y*
- *We also say that there is a functional dependency from X to Y, or that Y is* ***functionally dependent*** *on X*
- *The set of attributes X is called the* ***left-hand side*** *of the FD, and Y is called the* ***right-hand side***
- *If X → Y in R, this does not say whether or not Y → X in R*

FUNCTIONAL DEPENDENCIES

- An *FD* cannot be inferred automatically from a given relation extension *r* but must be defined explicitly by someone who knows the semantics of the attributes of *R*
- Figure below shows a particular state of the TEACH relation schema. Although at first glance we may think that *Text* → *Course*, we cannot confirm this unless we know that it is true for all possible legal states of TEACH. It is, however, ***sufficient to demonstrate a single counterexample to disprove an FD***. For example, because ‘Smith’ teaches both ‘Data Structures’ and ‘Database Systems,’ we can conclude that Teacher does not functionally determine Course.



TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

NORMAL FORMS

- The normalization process, as first proposed by Codd (1972a), takes a relation schema through a series of tests to *certify* whether it satisfies a certain **normal form**
- **Normalization of data** can be considered a process of analyzing a given relation schema based on their FDs and PKs to achieve the desirable properties of:
  - Minimizing redundancy
  - Minimizing the insertion, deletion, and update anomalies
- Definition

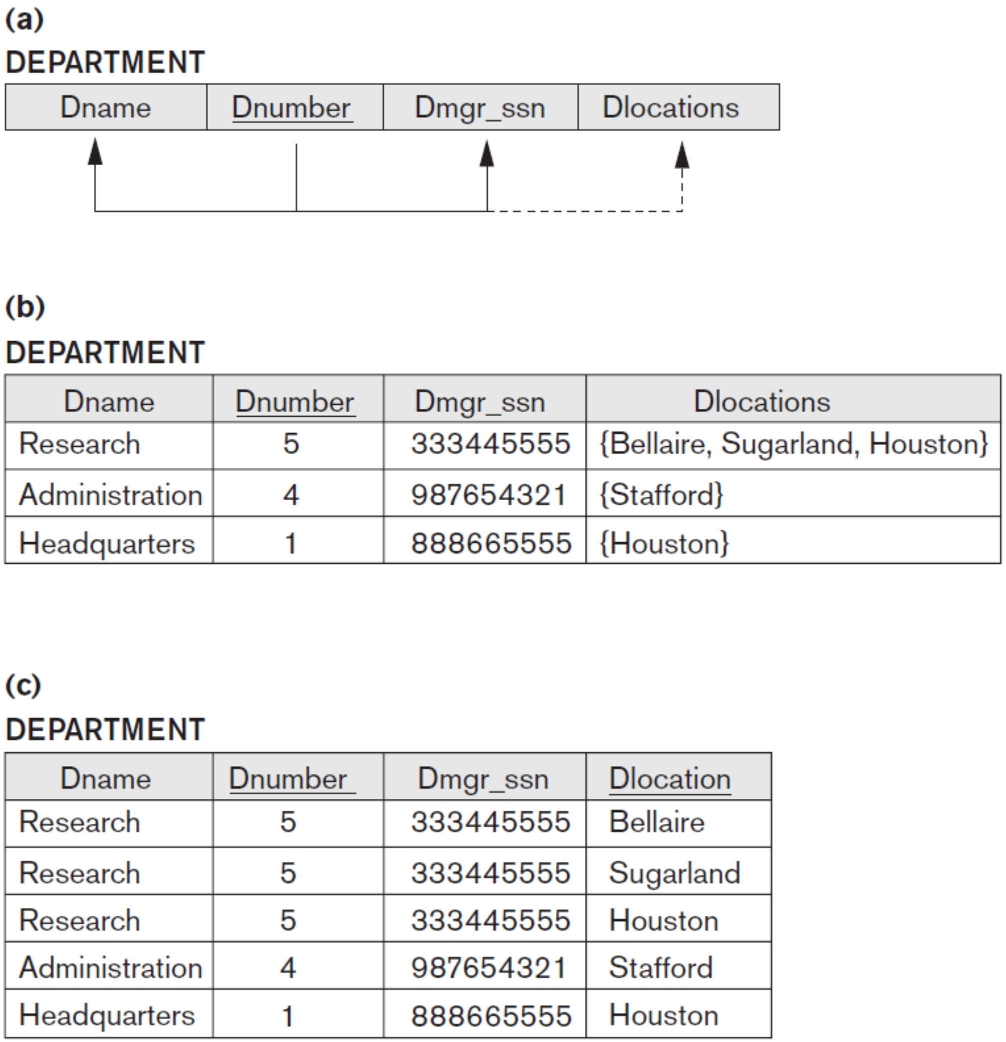
The **normal form** of a relation refers to the highest normal form condition that it meets, and hence indicates the degree to which it has been normalized

NORMAL FORMS - FIRST NORMAL FORM

- **First normal form (1NF)** is now considered to be part of the formal definition of a relation in the basic (flat) relational model; historically, it was defined to disallow *multivalued attributes*, composite attributes, and their combinations
- It states that the domain of an attribute must include only *atomic* (simple, indivisible) values and that the value of any attribute in a tuple must be a *single value* from the domain of that attribute
- The only attribute values permitted by 1NF are single **atomic** (or **indivisible**) values

NORMAL FORMS - FIRST NORMAL FORM

Normalization into 1NF. (a) A relation schema that is not in 1NF. (b) Sample state of relation DEPARTMENT. (c) 1NF version of the same relation with redundancy.



(a)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations

(b)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

NORMAL FORMS - FIRST NORMAL FORM

3 potential solutions:

1. Remove the attribute Dlocations that violates 1NF and place it in a separate relation DEPT_LOCATIONS along with the primary key Dnumber of DEPARTMENT. The primary key of this newly formed relation is the combination {Dnumber, Dlocation}. This decomposes the non-1NF relation into two 1NF relations
2. Expand the key so that there will be a separate tuple in the original DEPARTMENT relation for each location of a DEPARTMENT. This solution introduces redundancy in the relation and hence is **rarely adopted**
3. If a maximum number of values is known for the attribute, replace the Dlocations attribute by three atomic attributes: Dlocation1, Dlocation2, and Dlocation3. This solution introduces NULL values if most departments have fewer than three locations; querying on this attribute becomes more difficult. It is **best to avoid** this alternative

NORMAL FORMS - SECOND NORMAL FORM

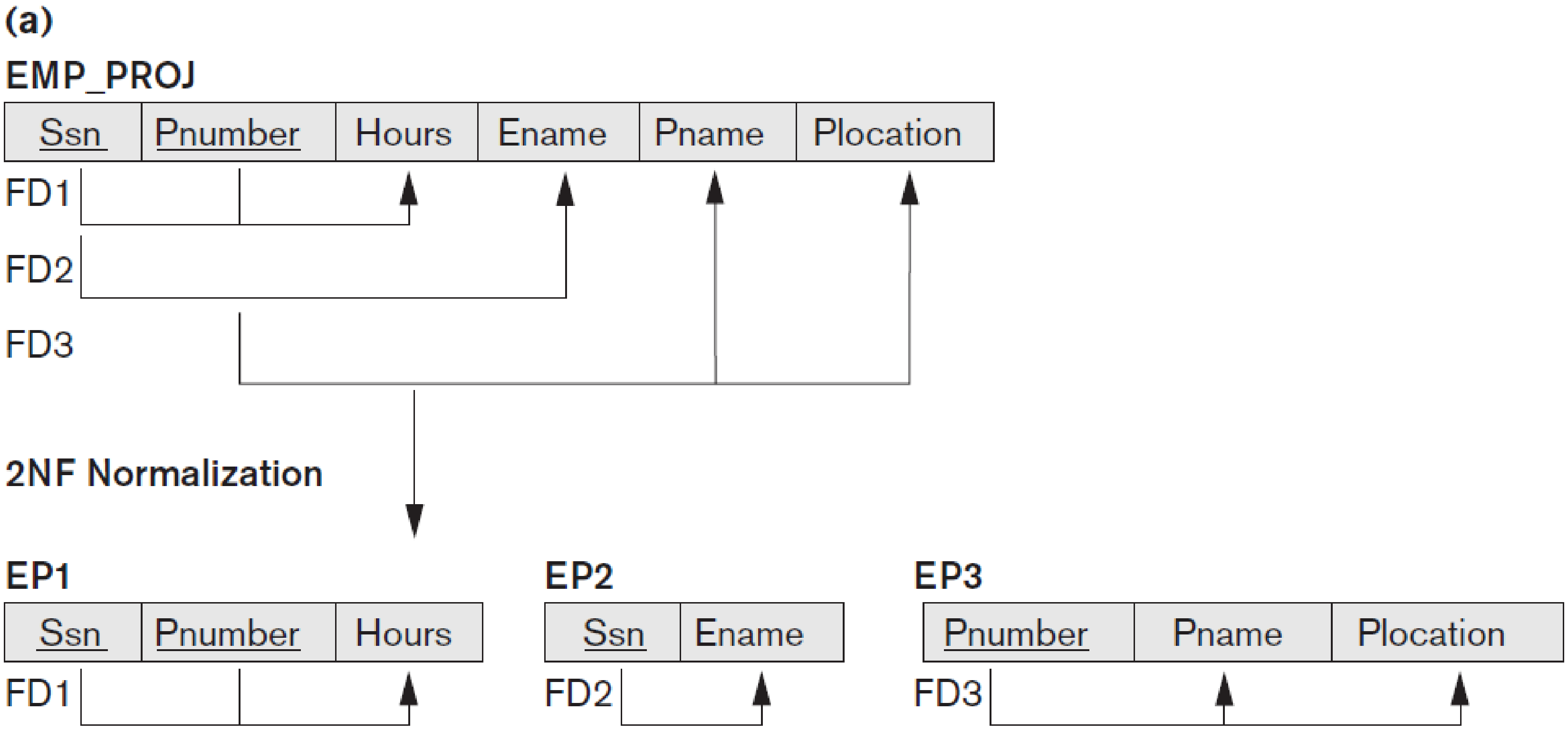
- **Second normal form (2NF)** is based on the concept of ***full functional dependency***
- Definition

A relation schema *R* is in **2NF** if every nonprime attribute A in R is *fully functionally dependent* on the primary key of *R*

- A functional dependency X → Y is a ***partial dependency*** if some attribute A ε X can be removed from X and the dependency still holds

NORMAL FORMS - SECOND NORMAL FORM

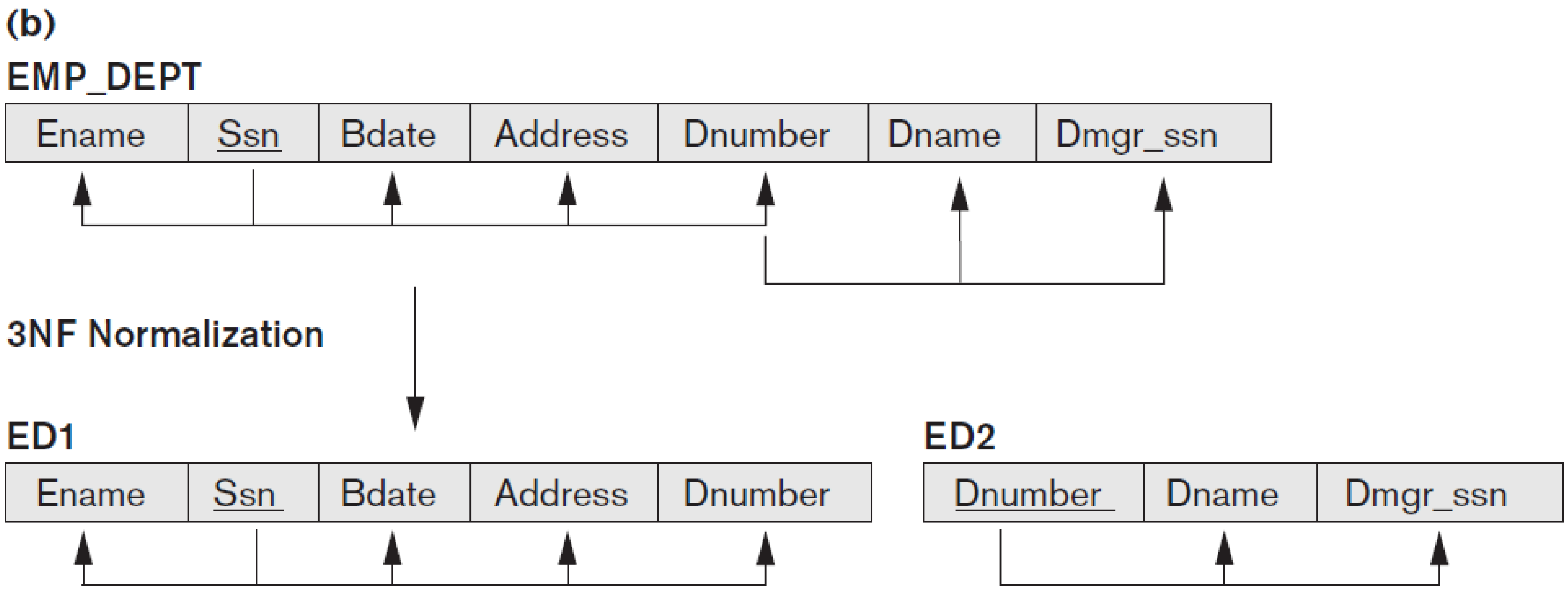
- Normalizing EMP_PROJ into 2NF relations



NORMAL FORMS - THIRD NORMAL FORM

- **Third normal form (3NF)** is based on the concept of ***transitive dependency***
- <u>Definition</u>

According to Codd's original definition, a relation schema *R* is in **3NF** if it satisfies 2NF *and* no nonprime attribute of *R* is *transitively dependent* on the primary key



(b)

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn

3NF Normalization

ED1

Ename	Ssn	Bdate	Address	Dnumber

ED2

Dnumber	Dname	Dmgr_ssn

NORMAL FORMS

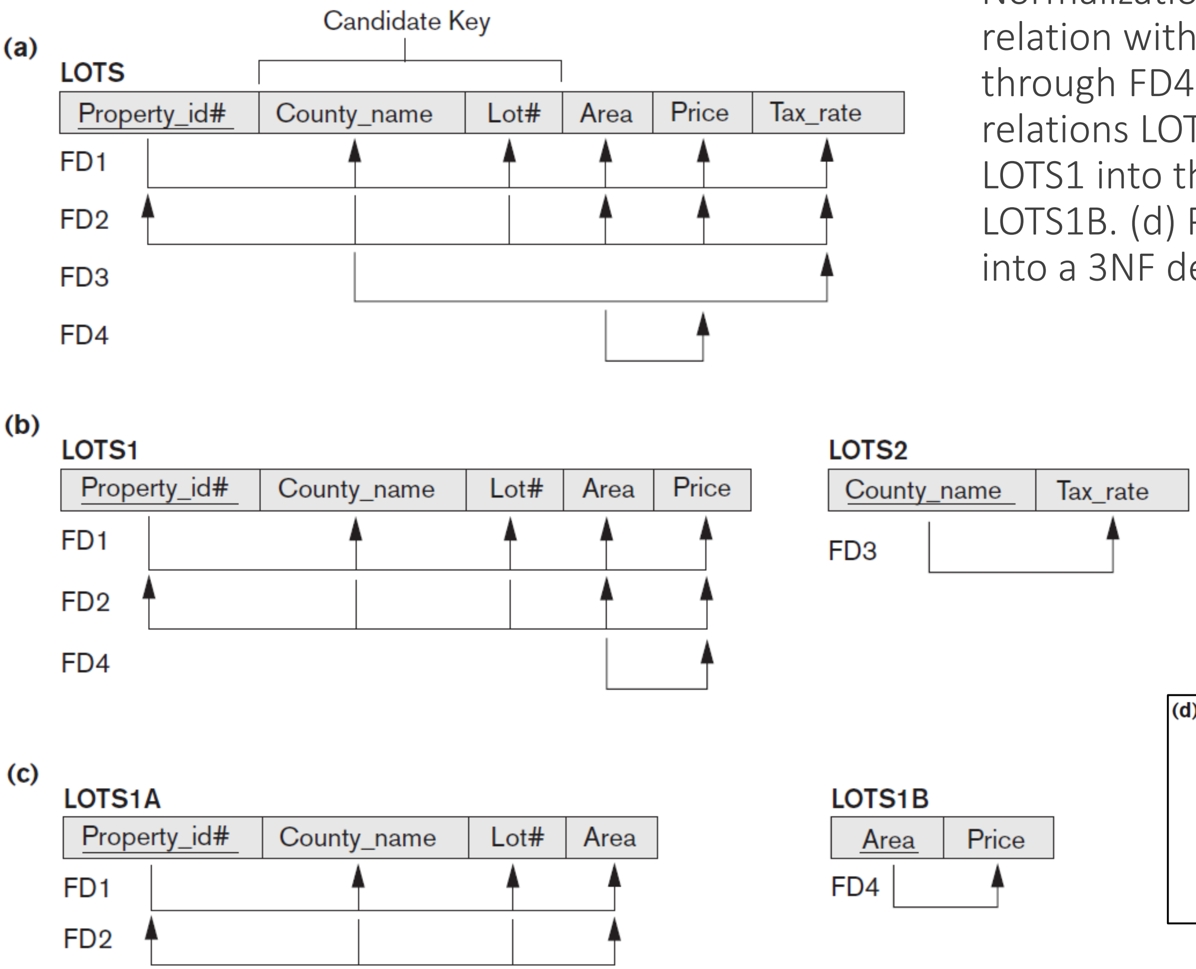
Normal Form	Test	Remedy (Normalization)
First (1NF)	Relation should have no multivalued attributes or nested relations	Form new relations for each multivalued attribute or nested relation
Second (2NF)	For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key	Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it
Third (3NF)	Third (3NF) Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s)

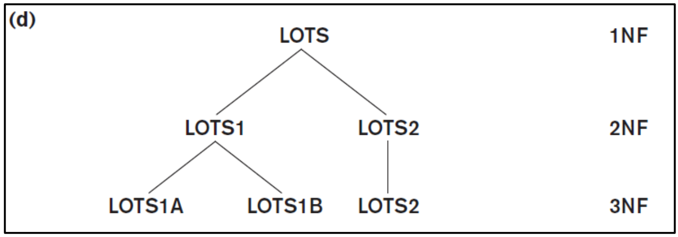
NORMAL FORMS – BOYCE-CODD NORMAL FORM

- **Boyce-Codd normal form (BCNF)** was proposed as a simpler form of 3NF, but it was found to be stricter than 3NF

- In the next 2 slides we show an example of relations in the 3FN but not in the BCNF because of an exception:
  - Suppose also that lot sizes in DeKalb County are only 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0 acres, whereas lot sizes in Fulton County are restricted to 1.1, 1.2, … , 1.9, and 2.0 acres
  - In such a situation we would have the additional functional dependency FD5: *Area* → *County_name*

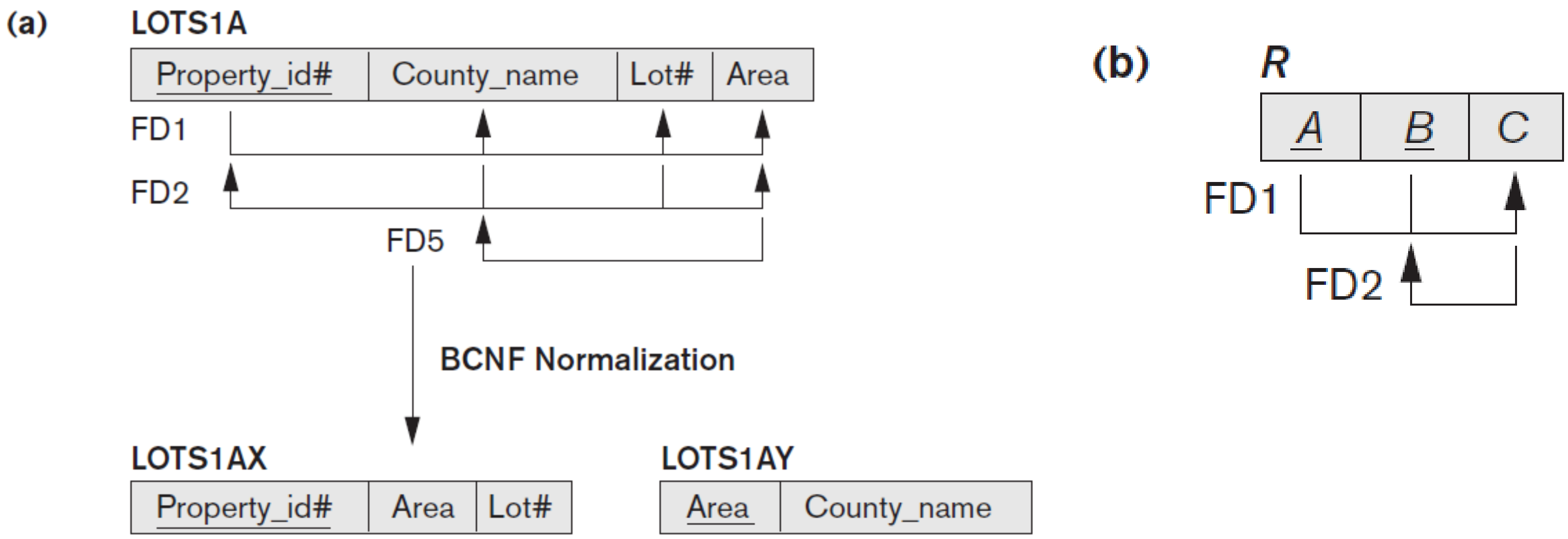
NORMAL FORMS – BOYCE-CODD NORMAL FORM





Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Progressive normalization of LOTS into a 3NF design.

NORMAL FORMS – BOYCE-CODD NORMAL FORM



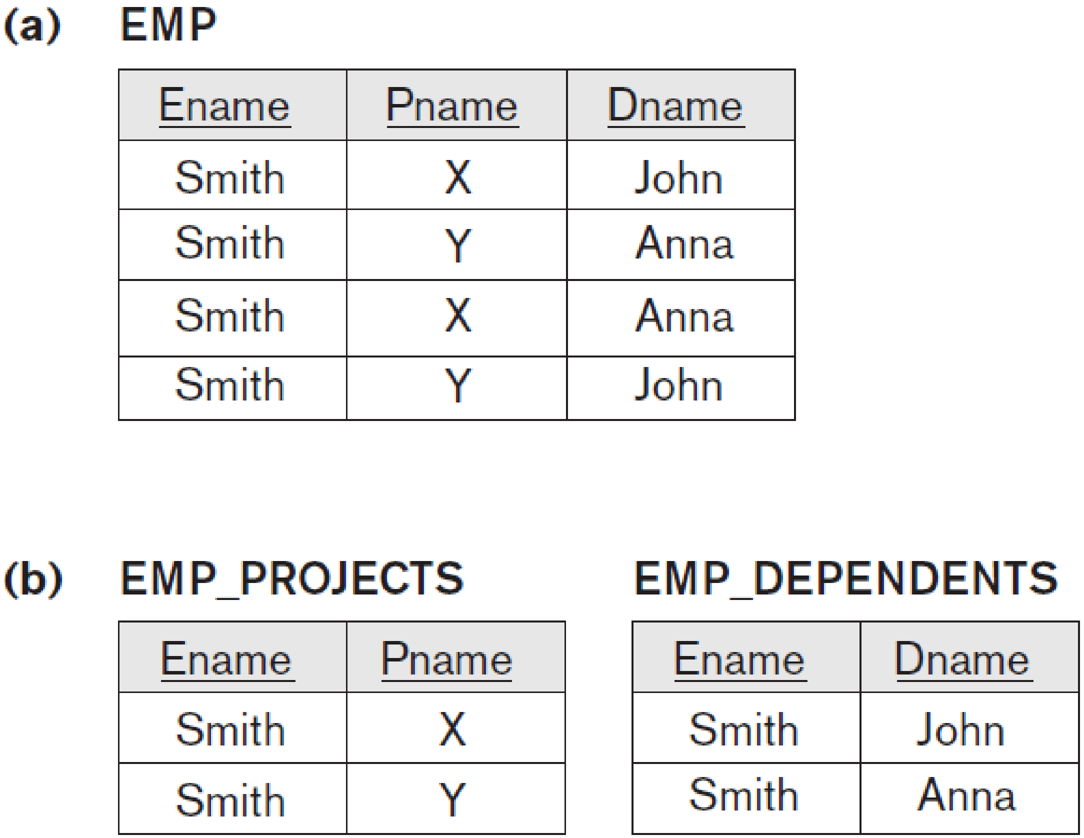
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. C → B.

MULTIVALUED DEPENDENCY AND FOURTH NORMAL FORM

- (a) The EMP relation with two MVDs:

Ename→→ Pname and
Ename →→ Dname

- (b) Decomposing the EMP relation into two 4NF relations EMP_PROJECTS and EMP_DEPENDENTS



(a) **EMP**

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

(b) **EMP_PROJECTS**

Ename	Pname
Smith	X
Smith	Y

EMP_DEPENDENTS

Ename	Dname
Smith	John
Smith	Anna

MULTIVALUED DEPENDENCY AND FOURTH NORMAL FORM

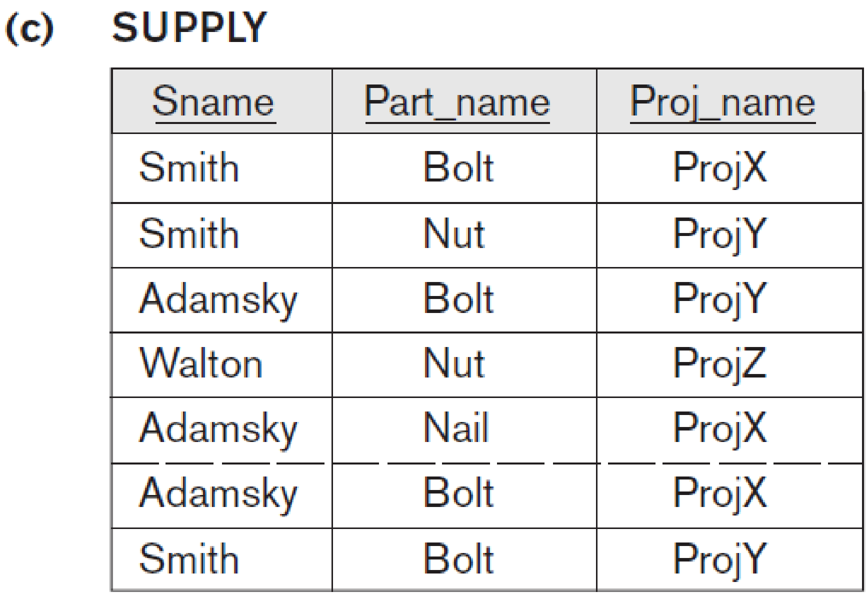
- Definition

A relation schema *R* is in **4NF** with respect to a set of dependencies *F* (that includes functional dependencies and multivalued dependencies) if, for every nontrivial multivalued dependency *X* →→ *Y* in F+, X is a superkey for *R*

- We can state the following points:
  - An all-key relation is always in BCNF since it has no FDs.
  - An all-key relation such as the EMP relation in Figure 14.15(a), which has no FDs but has the MVD *Ename* →→ *Pname* | *Dname*, is not in 4NF.
  - A relation that is not in 4NF due to a nontrivial MVD must be decomposed to convert it into a set of relations in 4NF.
  - The decomposition removes the redundancy caused by the MVD

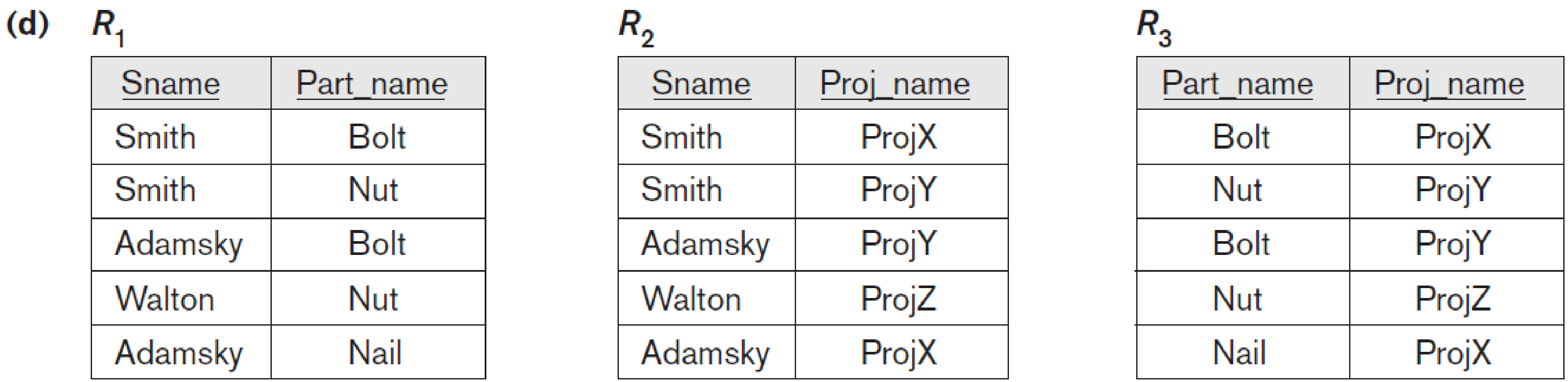
JOIN DEPENDENCIES AND FIFTH NORMAL FORM

- (c) The relation SUPPLY with no MVDs is in 4NF but not in 5NF if it has the JD(R1, R2, R3)
- (d) Decomposing the relation SUPPLY into the 5NF relations R1, R2, R3



(c) **SUPPLY**

Sname	Part_name	Proj_name
Smith	Bolt	ProjX
Smith	Nut	ProjY
Adamsky	Bolt	ProjY
Walton	Nut	ProjZ
Adamsky	Nail	ProjX
Adamsky	Bolt	ProjX
Smith	Bolt	ProjY



(d) R_1

Sname	Part_name
Smith	Bolt
Smith	Nut
Adamsky	Bolt
Walton	Nut
Adamsky	Nail

R_2

Sname	Proj_name
Smith	ProjX
Smith	ProjY
Adamsky	ProjY
Walton	ProjZ
Adamsky	ProjX

R_3

Part_name	Proj_name
Bolt	ProjX
Nut	ProjY
Bolt	ProjY
Nut	ProjZ
Nail	ProjX

Thank you!

Jacinto Estima

Jacinto.Estima@universidadeeuropeia.pt