Unit 4:
Relational Databases

4.1. Database Design Fundamentals

4.2. Conceptual Design

4.3. Logical Design

Bases de Datos y Sistemas de información
Departamento de Sistemas Informáticos y Computación / Universidad Politécnica de Valencia

V. 16.1

Unit 4.1 DB Design Fundamentals

1. Introduction

2. Methodology

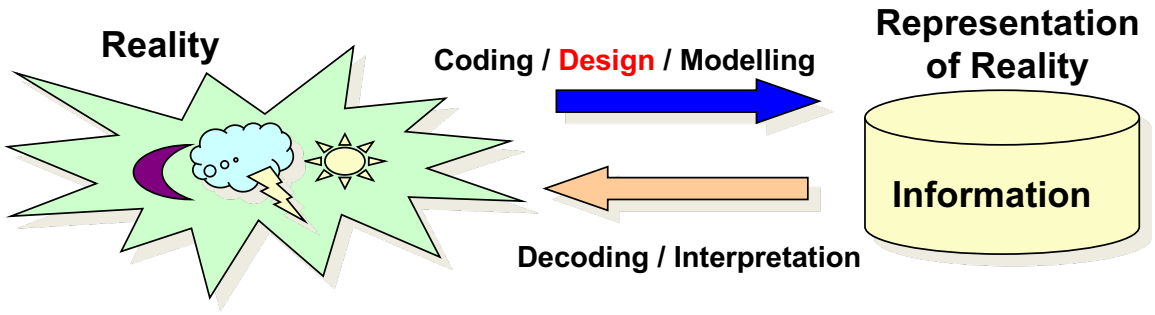
3. Data models

4. Database Design

5. Example

1. Introduction

In this unit we will present a methodology for the design of relational databases



We will focus on:

- Methodology issues: Strategies and recommendations to address the design problem.
- Modelling languages issues: Presentation of an appropriate (graphical) language to represent the sistem (data model).

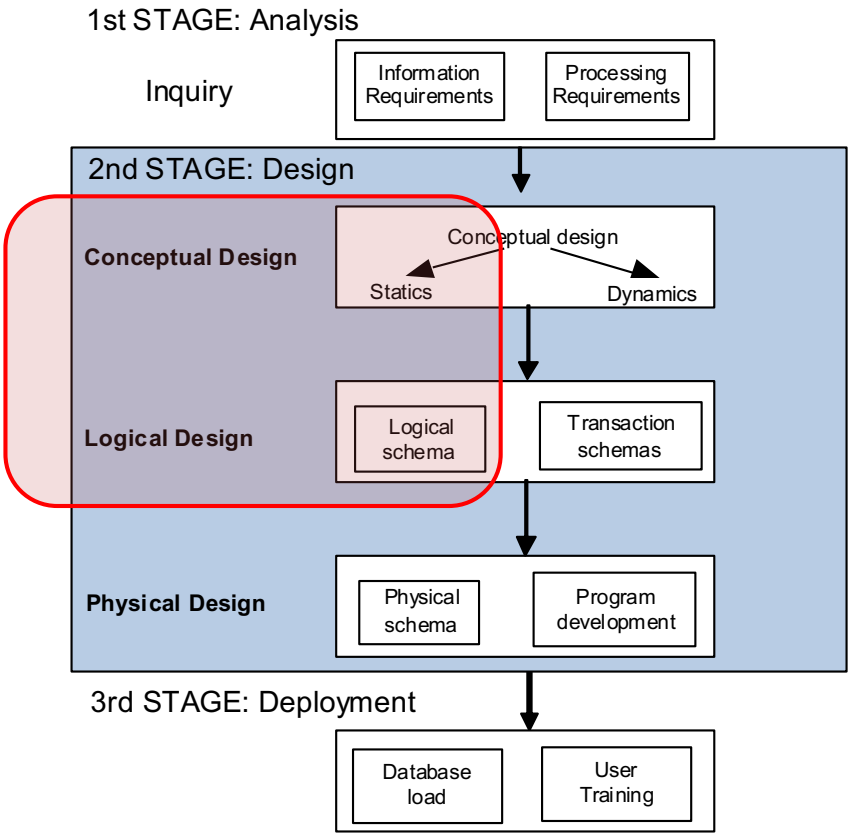
UD 4.1 DB Design Fundamentals

2. Methodology

A methodology is a set of standard procedures, techniques and documentation for the development of a product (a database in our case).

A methodology is supported by:

- Techniques: how to deal with each of the steps and activities in the methodology
- Models: Way to represent or think abstractly about the reality, the problem or the solution.
- CASE tools: (optionally) software tools to automate or assist on the development of techniques and models.

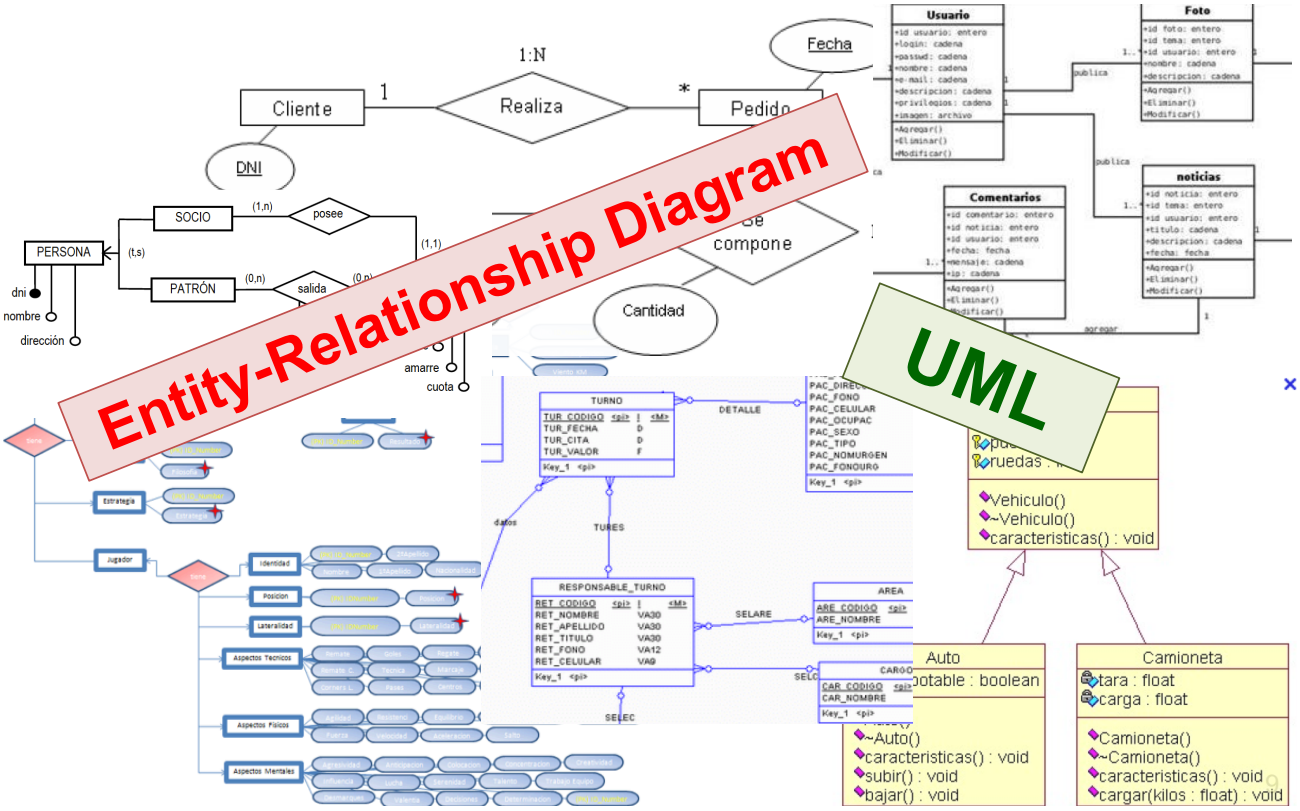


3. Data models

A **data model** is a way in which static and dynamic properties of reality (entities, relationships,…) are represented.

Example: The relational data model

There are many models, many notations,…



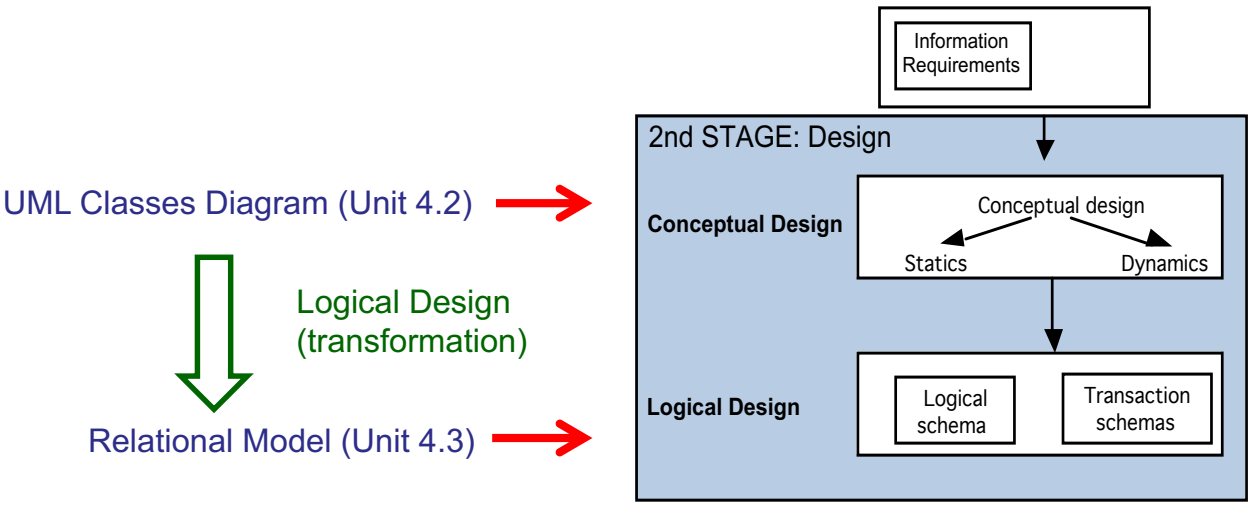
We are going to use a conceptual data model that:

- incorporates notions from the Entity-Relationship Model using UML
- is more abstract , expressive, and system-independent than the classical relational model
- is essentially graphical (based on UML)

UD 4.1 DB Design Fundamentals

1. Introduction
2. Methodology
3. Data models
4. **Database Design**
5. Example

4. Databases Design



1. Analysis stage: information requirements

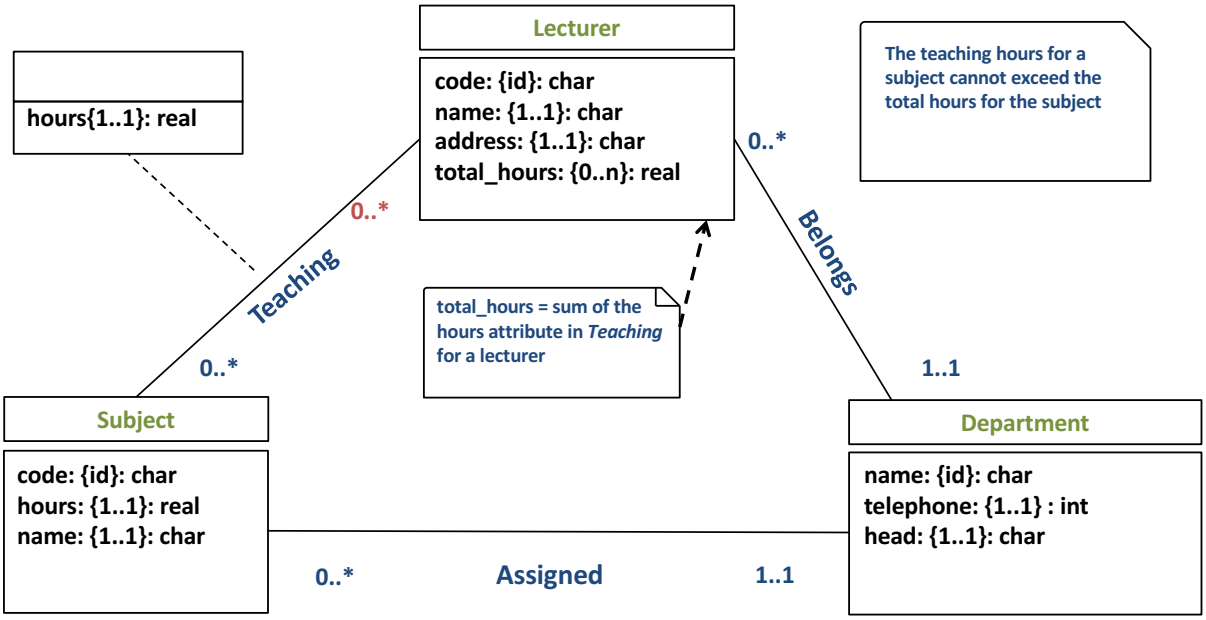
Lecturer:
- code, name and address
- department where the lecturer belongs to
- subjects s/he teaches, and how many hours each
- total number of teaching hours s/he is assigned

Department:
- name, head and telephone.

Subject:
- code and name of the subject,
- total number of hours in the degree syllabus
- department which is assigned to.

INTEGRITY CONSTRAINTS:
- A lecturer must belong to one and only one department.
- A subject must be assigned to one and only one department.
- There can't be two departments with the same name.
- There can't be two lecturers with the same code.
- There can't be two subjects with the same code.
- The number of taught hours for a subject cannot exceed the total number of hours for the subject in the degree syllabus.
- A lecturer cannot teach more than 12 hours in one subject.
- A subject cannot have more than 24 hours.

2a. Design stage: Conceptual Design (static)



UML Classes Diagram

2a. Design stage: Conceptual Design (dynamic)

Transaction *Inserta_lecturer*
- Insert into Lecturer
- Insert into Belongs

Transaction *Insert_subject*
- Insert into Subject
- Insert into Assigned

Transaction *Insertar_departament*
- Insert into Departamen

…

Transaction description

2b. Design stage: Logical Design (static)

Department (name: char(50), head: char(50), telephone: char(8))
PK: {name}

Lecturer (code: char(9), name: char(50), address: char(50), dname: char(50))
PK: {code}
FK: {dname} → Department
NNV: {name, address, dname}

Subject (code: char(5), name: char(50), hours: number, dname: char(50))
PK: {code}
FK: {dname} → Department
NNV: {name, hours, dname}

Teaching (lcode: char(9), scode: char(5), hours: number)
PK: {lcode, scode}
FK: {lcode} → Lecturer
FK: {scode} → Subject
NNV: {hours}

(*) The attribute total_hours is not included and will be calculated every time it is needed.

(**) The number of taught hours for a subject cannot exceed the total number of hours for the subject in the degree syllabus.

2b. Design stage: Logical Design (dynamic)

TRANSACTION Insert_lecturer (code: char(9), name: char(50), address: char(50), dname: char(50))
INSERT INTO Lecturer **VALUES** (code, name, address, dname)

TRANSACTION Insert_subject (code: char(5), name: char(50), hours: number, dname: char(50))
INSERT INTO Subject **VALUES** (code, name, hours, dname)

TRANSACTION Insert_departament (name: char(50), head: char(50), telephone: char(8))
INSERT INTO Department **VALUES** (name, head, telephone)

2c. Design stage: Physical design

Lecturer:

File indexed by *code;* index on *name*

Subject:

File indexed by *code;* index on *name*

Department:

Sequential file; index on *name*

Teaching:

File indexed by *scode;* index on *lcode*