Course on Database Design

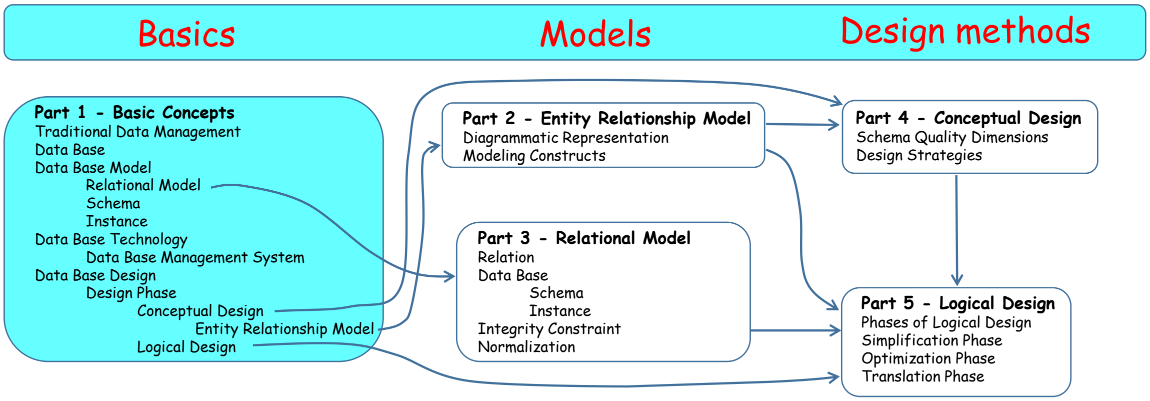
Carlo Batini

University of Milano Bicocca, Italy

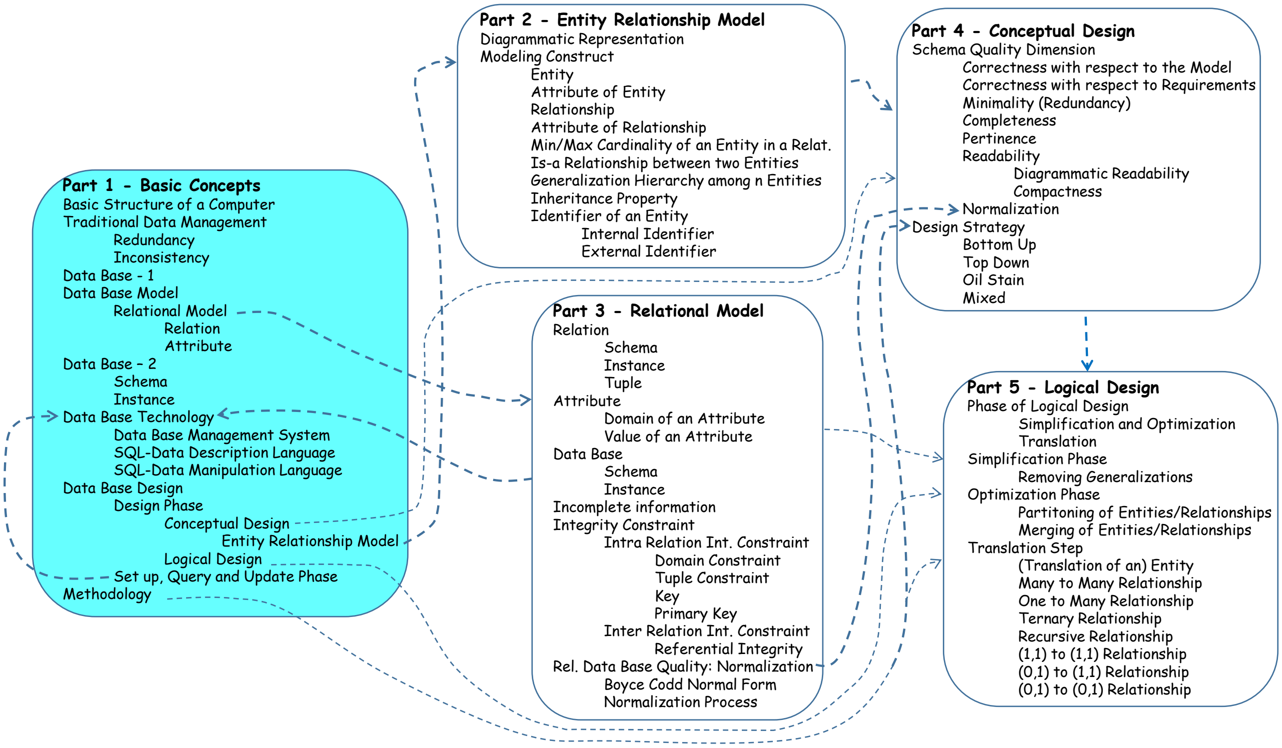
Part 1 – Introduction to Databases

Part 1 - Lesson 1 - Introduction to Data, Information, Organizational system, Information System and Computer Structure

This is the first lesson of the course on Database Modeling and Design, and the first lesson of the introductory part. You see in these figures at two different levels of detail where we are now in the course, and what follows.



High level conceptual map



Low-level conceptual map

After this introductory part, we will investigate models and design methods.

We focus now on basic concepts we will use in the other parts. And we start discussing five concepts that help us to understand which is the usage of databases in organizations. They are:

- Data
- Information
- Organizational system
- Information System
- Computer Architecture

<u>Data and Information</u>

Data is a discrimination between physical states of things (e.g. the color of a t-shirt is black, white etc.) that may convey or not convey information to an agent. Whether it does or not depends on the agent's prior stock of knowledge. So, data is a property of things (the world around us), while knowledge is a property of agents (human beings, software applications, computers).

Information establishes a relationship between things and agents.

Example - The fact that our body temperature, measured using a thermometer, is equal to 38.5 Celsius degrees is a *datum*. It becomes an *information* if we know:

1. What is a *body temperature* (38.5).
2. How it is measured (in this case, Celsius degrees).

A thermometer

Furthermore, if we know also which is the body temperature when we are healthy, namely about 36.5 Celsius degrees, we can deduce that --> when the Celsius temperature is 38.5, we have fever!

Organizational system and Information system

An *organizational system* (see next figure) makes reference both to companies and to public administrations. Typically, an organization receives in input goods or requests of services. It receives goods if its goal is to transform them in other goods, such as for instance a manufacturing company that produces and sells cars. It receives requests of services if it is a service provider, e.g. a company that sells flight tickets and allows you to a. make a reservation for a certain route in a given day and b. buy the ticket.

Organizational system and Information System

To produce goods or deliver services, organizations execute *processes*, namely activities. Such activities to be executed need for *resources*, and such resources can be of various types, such as money, human resources, or another resource in which we are specifically interested, *information*.

To execute a process or part of it, organizations make use of several types of technologies, among them we are interested to Information and Communication Technologies (ICT). ICT technologies can be very roughly divided into software, hardware, network and a fourth resource that is used by all of them, *data*. As we will see soon, data are managed by means of a technology widely used in the world, namely *databases*.

In the above figure, the components corresponding to information, ICT technologies and data are called as a whole with the term *Information System*, while the whole set of components is the *Organizational System*. Notice that the above model we have defined for organizations is also valid for physical systems, individuals and communities of individuals.

Computer structure

A final concept we have to discuss is the *computer structure*, see figure below.

Computer structure

Computers have significantly evolved in the last sixty years, but their basic structure has been always the same. The first block we consider is the secondary memory, and, more specifically, disks, on whose sectors data in databases are permanently stored.

The second block is the Central Processing Unit (CPU), where programs are executed, and, related to it, the main memory, where programs find data processed by programs. Finally we have Input Output devices, that allow the computer to interact with the external world, and channels, that enable the transfer of data from secondary memory to main memory, where data are processed by programs.

One of the most critical problems in this architecture (see again the above figure) concerns the dramatic gap between the time needed to access data in secondary memory and the time needed to access data in main memory, at least seven orders of magnitude. Data are permanently stored in secondary memory, while to be processed, they have to be transferred to main memory. In order to execute efficiently software applications (queries and transactions) that work on databases, we have to be very efficient in such transfers.

Part 1 - Lesson 2 - Traditional data management and modern data management with data bases

Traditional data management

We start with a motivating example, as we will do often in the following. Assume that we are looking at the information system of a Chinese university, which, among others, manages two files, represented as tables in the following figure.

Table 1 - Courses and Professors

Course #	Prof Id	Prof Surname	Level	Yearly Salary (in $)
37	1	Xu	4	35.000
29	1	Xu	4	35.000
16	2	Smith	3	30.000

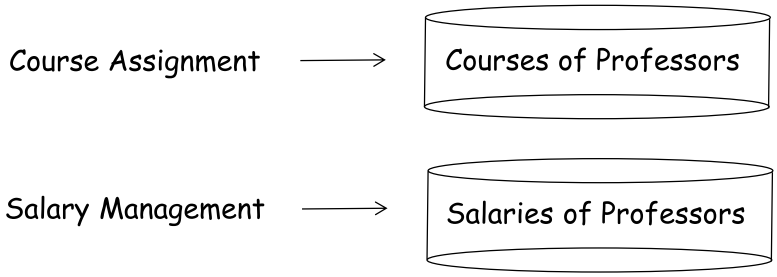
Table 2 - Salaries of Professors

Level	Yearly Salary
1	20.000
2	25.000
3	30.000
4	35.000

Two tables

The first table represents professors and courses they give. We represent in the columns the Course number, the Professor Id, Professor Surname, Level (of professors) and the Yearly salary. The second table represents Levels of professors and the Yearly salary.

We can assume that the first table is used by a software application "Course assignment" and the second table is used by another software application named "Salary management".



Two software applications using data

You should not be surprised of the presence of the same data managed by different software applications; (unfortunately) this was typical in traditional data management. Each application managed its own data, and so it was very usual the situation in the figure.

The above organization presents (at least) two problems.

1. The first problem is the *presence of redundancy of data*; that is, some data are represented more than once. If you look carefully at the two tables we have two types of redundancy:
    a. If one professor, say Xu, teaches more than one course, say three courses, the surname, the level and the yearly salary of the professor are represented three times in the data.
    b. Level and yearly salary are represented in two tables.

2. The second problem is the *risk of inconsistent representation* of the same data in different tables. E.g. when salaries are updated in the second table, we have to remember to update them also in the first table.

Question 1.1 - How can we solve the two problems above by modifying the tables? Try to provide an answer to this question, stop the video (or else stop the reading…) and when you have a solution, start again the video (or else turn the page…) and listen to the discussion (or else read the solution).

Remark - Here and in all the following parts of the book, after each question or exercise we insert a *page break* to encourage you to provide a solution, and compare subsequently such solution with the discussion you find in the following page.

Discussion on Question 1.1

First problem – Risk of inconsistent representation of the same data

We can delete from Table 1 the column Yearly Salary, resulting in the new table Courses and Professors.

Courses and Professors (Course #, Prof Id, Prof. Surname, Level)

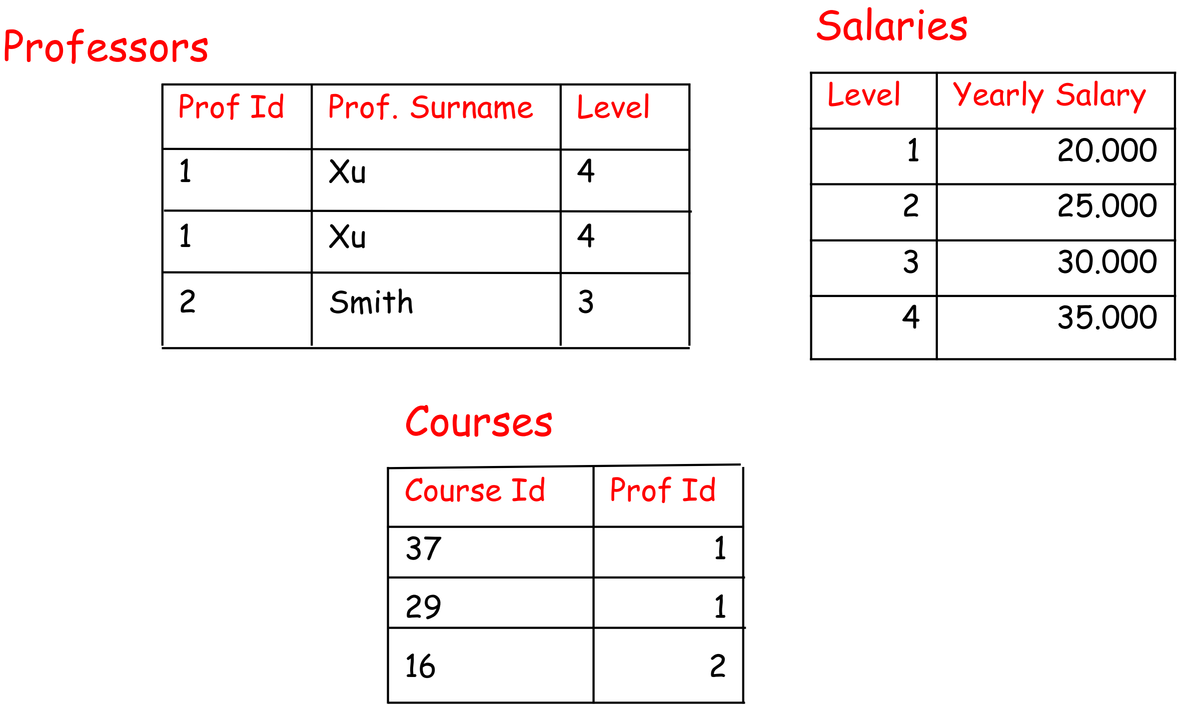
Notice that we cannot delete the column Level from the table, since we need to link its values with values in table Salaries of Professors.

Second problem – Presence of redundancy

We can split table Courses and Professors (Course #, Professor Id, Prof Surname, Level) into two tables

Courses (Course #, Professor Id) and
Professors (Professor Id, Prof Surname, Level)

resulting in the new database reproduced in the following figure.



Professors

Prof Id	Prof. Surname	Level
1	Xu	4
1	Xu	4
2	Smith	3

Salaries

Level	Yearly Salary
1	20.000
2	25.000
3	30.000
4	35.000

Courses

Course Id	Prof Id
37	1
29	1
16	2

The new set of tables

Notice that in the second step we have to keep in the table Courses the column Professor Id, in such a way that so we can link table Courses to Professors.

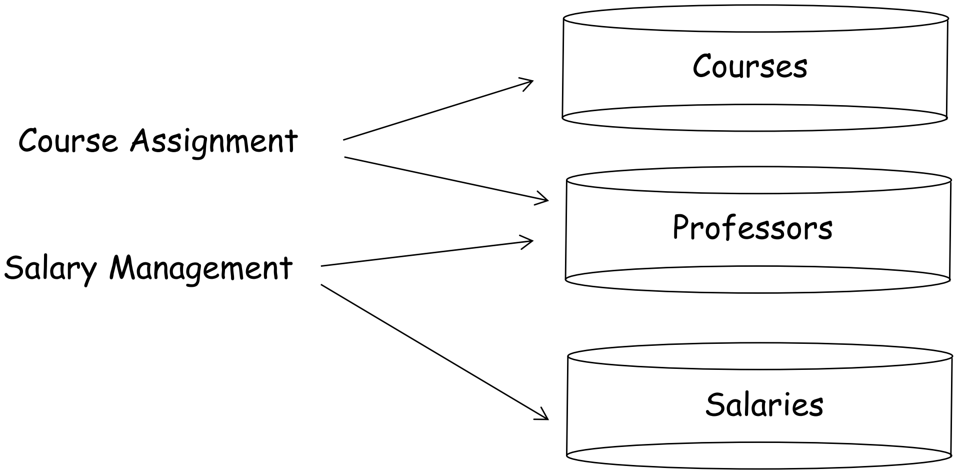
Database

In the 70ties of the last century, several researchers introduced the concept of *Database*. What is a database?

Definition - A *database* is a collection of data:

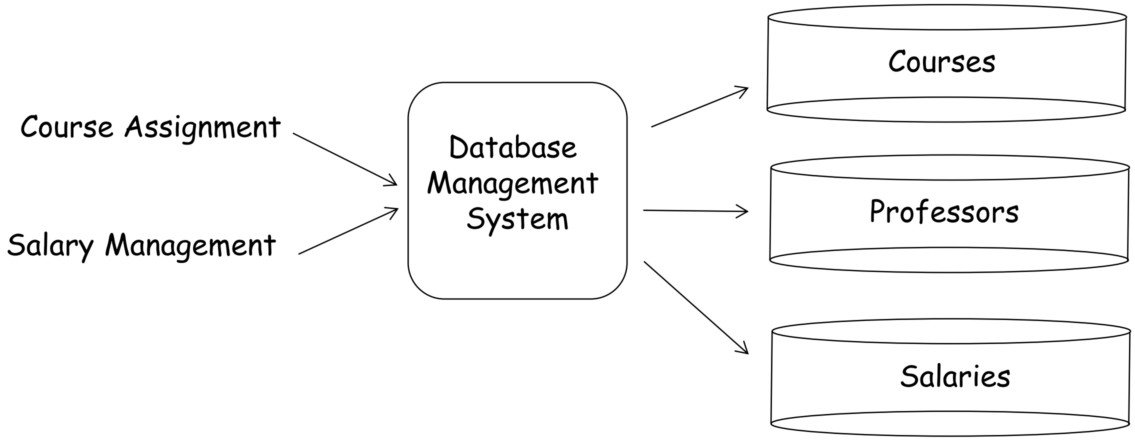
- used to represent information of interest to different sectors of an organization,
- shared between different software applications and different users of the organization,
- where each information of interest is represented only once in the collection of data.

Therefore, with the introduction of a database we have that the two previous applications, Course assignment and Salary management, instead of using separate files, share the same set of tables (see the following figure).



The data base approach

Now, in order to allow different users and applications to share the same data we need to build a new software application that manages such sharing, disciplining while making efficient the concurrent access of software applications and users to data sets in the database, see the following figure. We have encountered the concept of *Database Management System*.



Database Management Systems allow concurrent and efficient access to the data base

Data Base Management System

A *Data Base Management System* (DBMS) (see the following figure) is a software system able to manage requests of users and software applications in terms of queries and transactions, to access databases, namely collections of data that are:

- Large (bigger, often much bigger than the main memory available).
- Shared (used by various software applications and users).
- Persistent (with a lifespan that is not limited to single executions of the applications that use them).

Remember: *large, shared, and persistent.*

Database Management System and its relationship with a computer

Requests of users and software applications are of two types:

1. *Queries*, that retrieve from the database data that match certain selection criteria expressed in the query. As such, queries *do not change* the state of the database.

2. *Transactions*, that insert, delete, update data in the database. As such, transactions *change* the state of the database.

Which are the most relevant characteristics of a DBMS? Remember that a DBMS should guarantee the shared access of different applications to a set of data in the database, and that each application should access only some of such tables. Therefore, main characteristics of a Data Base Management System are:

- *Reliability*, preserving the database in case of hardware or software failure.
- *Privacy*, controlling accesses and authorizations.
- *Efficiency*, using the appropriate resources, such as *time* and *space*.

Part 1 - Lesson 3 - Database models

Database models and relational model

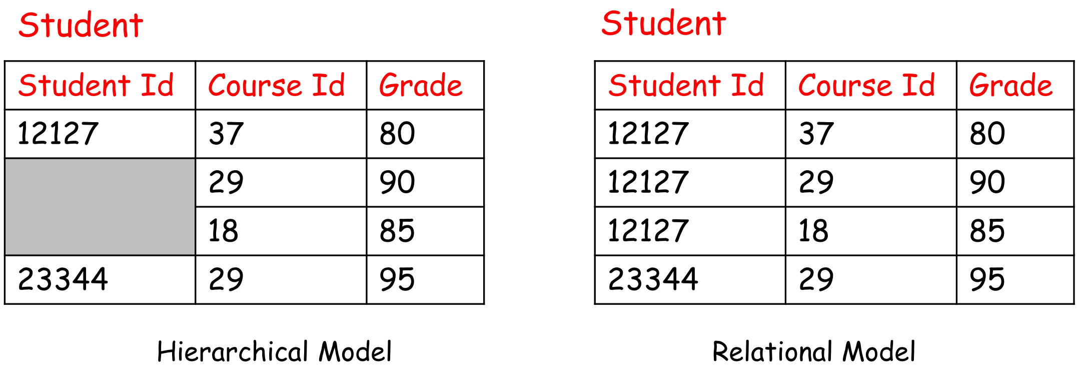
We start as always with an example.

Example - Assume we are in China (where grades are among 0 and 100). We want to represent:

a. three exams passed by a student with Student Id = 12127 referring to:
   1. Course with Course Id = 37 and exam passed with grade 80.
   2. Course with Course Id = 29 and exam passed with grade 90.
   3. Course with Course Id = 18 and exam passed with grade 85.

b. one exam passed by a student with Student Id = 23344 referring to:
   1. Course with Course Id = 29 and Exam passed with grade 95.

We can represent the above information at least in two different ways, see next figure *(notice that for tables we have adopted singular names, this convention will be adopted from now on).*



Student

Student Id	Course Id	Grade
12127	37	80
	29	90
	18	85
23344	29	95

Hierarchical Model

Student

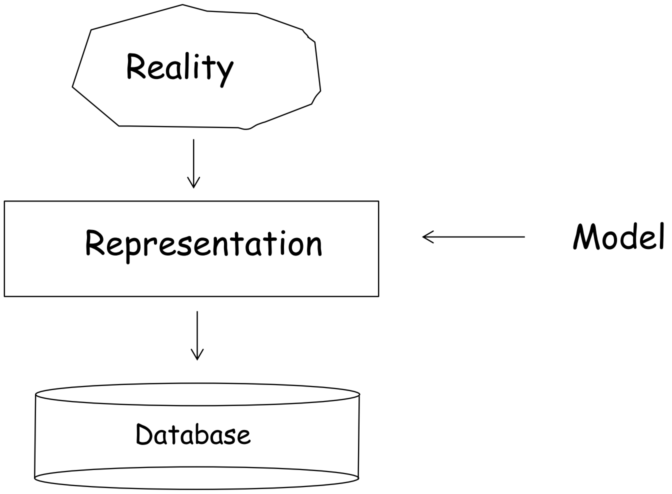
Student Id	Course Id	Grade
12127	37	80
12127	29	90
12127	18	85
23344	29	95

Relational Model

Hierarchical and relational models of data

We see in the previous example that a database (in our case a set of data on students, courses and exams) is a representation of a piece of reality (see next figure), using a given data organization that we call a *data (or database) model*.

Definition - A *database model* is a set of constructs capable of describing the data requirements of an application.



Databases as representation of a piece of reality using a data model

Modern database management systems adopt the relational model. The relational model adopts a unique representation structure, the *relation* or *table*, defined in terms of a set of columns also called *attributes*.

Student

Student Id	Surname	Given Name
37	Smith	John
29	Wang	Fuhang
18	Xu	Xiaofei
43	Batini	Carlo
……	……	………

A relation (or table)

Look at the above table; we may identify several concepts that are relevant in the relational mode.

Relation schema – The relational schema is made of the relation name (Student) and the column (or attribute) names. The schema represents concepts (e.g. Surname) of the real world and is rather stable over time.

Notice that we will also write the relation schema as in the following

```
Student (Student Id, Surname, Given Name)
```

Database schema – When the database is made of several tables, the database schema correspond to the set of their relation schemas.

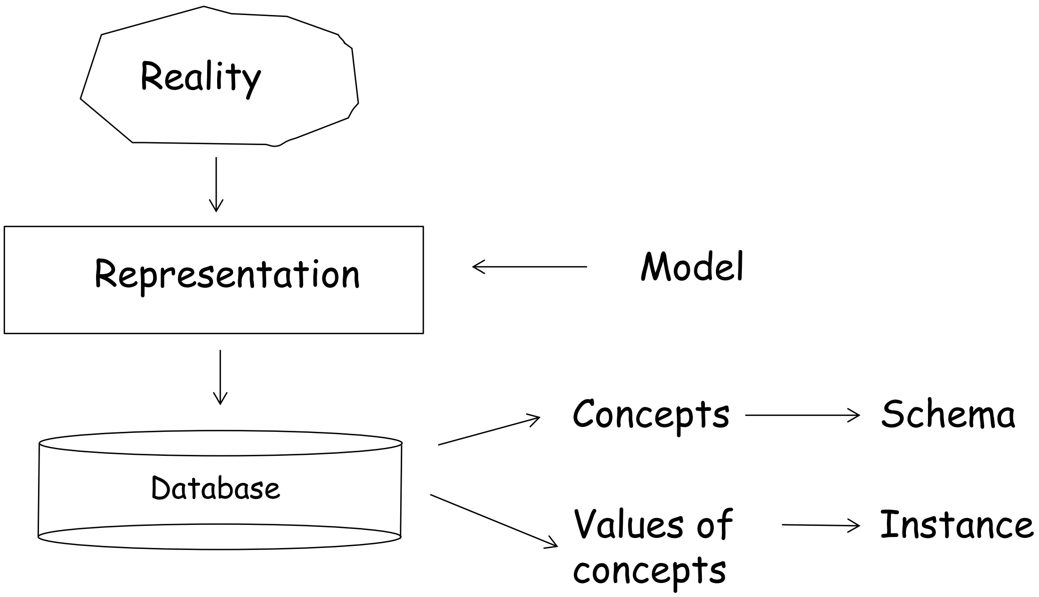
Relation instance – The relation instance represents the values of concepts in the relation (e.g., the Given Name John, the Surname Xu); values can change in time (e.g. a new student could be added).

Database instance – Is a set of relation instances all related to the same database.

A relation schema + the corresponding relation instance is also called *relation* (or table).

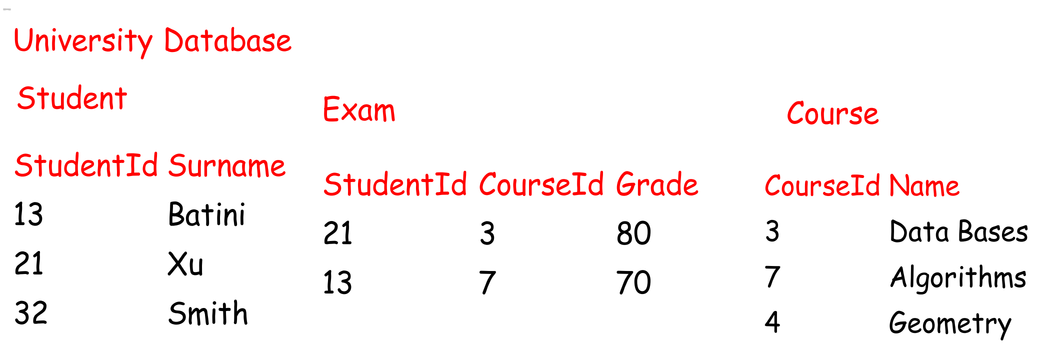
Tuple – Is a specific set of values in the relation instance corresponding to a single row in the table.

We can recap the above discussion on models by means of the following figure.



Reality, database, schema and instance

Exercise 1.1 - Given the following database, reply to the following questions.



University Database

Student

StudentId	Surname
13	Batini
21	Xu
32	Smith

Exam

StudentId	CourseId	Grade
21	3	80
13	7	70

Course

CourseId	Name
3	Data Bases
7	Algorithms
4	Geometry

a. How many relations are there in the database?
b. How many databases?
c. How many attributes?
d. How many tuples?
e. How many relation instances?
f. How many database instances?

Discussion on Exercise 1.1

a. How many relations in the database? 3
b. How many databases? 1
c. How many attributes? 7
d. How many tuples? 8
e. How many relation instances? 3
f. How many database instances? 1

Remark – Here and in the following, if you did not reply as in the discussion, look again at the definitions, examples and explicative text and try to understand the reasons of discrepancy.

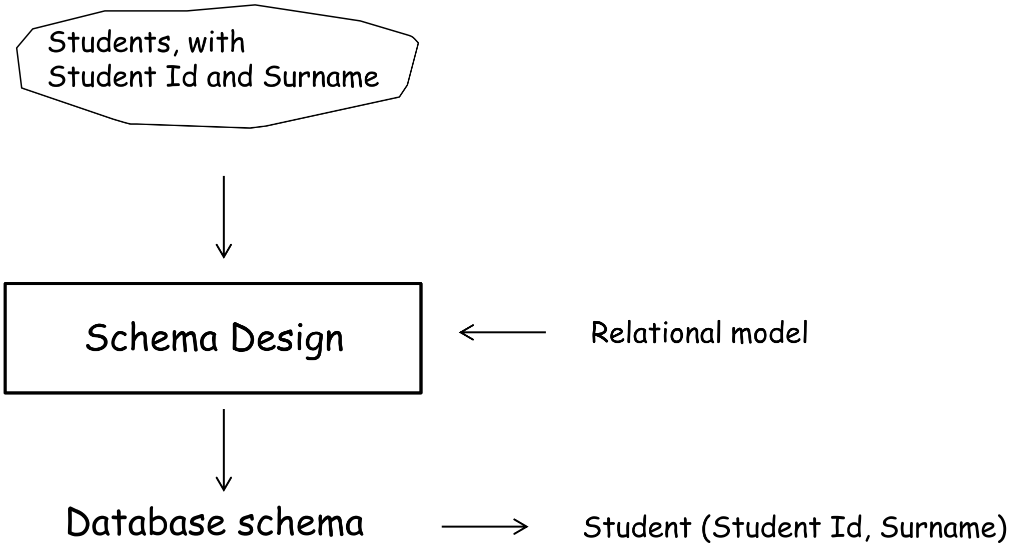
Part 1 – Lesson 4 – Course Organization and parts

We now reply to the fundamental question of the course:

What shall we do to design and use a database?

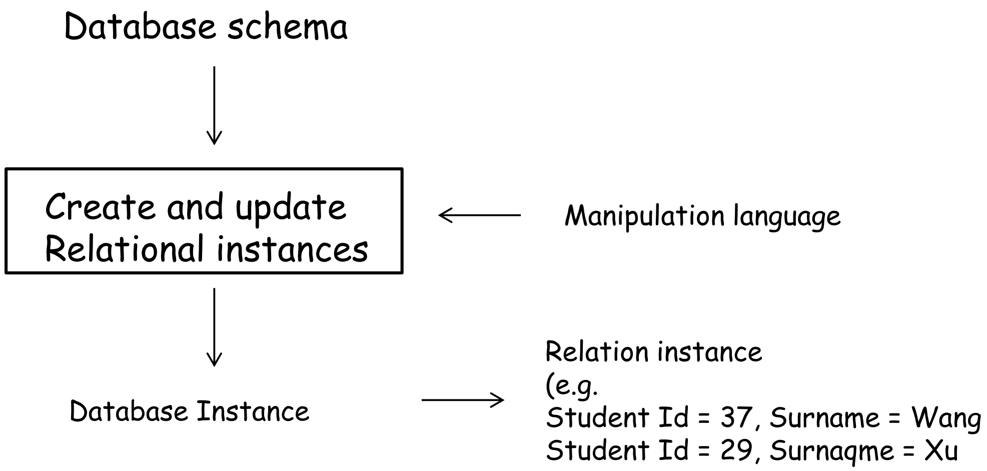
Replying to such question, we will discover the phases of the so-called Database Life Cycle.

1. First we have to design the database schema → *Design phase* (see the next figure an example of input and output to the phase. The input to the design phase is called with the term *Requirements*). The database schema is described according to a model, in this course the relational model. At the end of the Schema Design phase we can specify to the DBMS the relational schema structure using the SQL Data Description Lanugage.



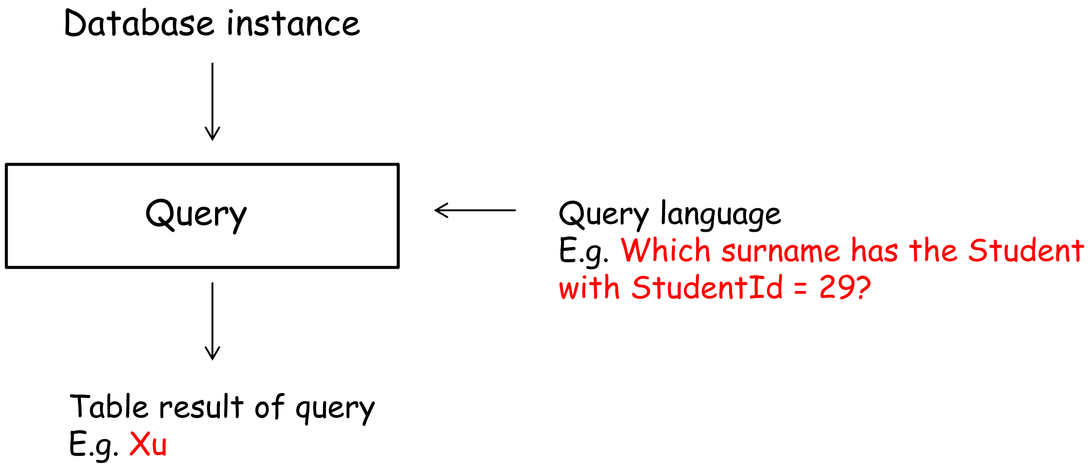
Design phase of the Database life cycle

2. Then we have to create and update over time the relation instances → Instance update phase.



Instance update phase

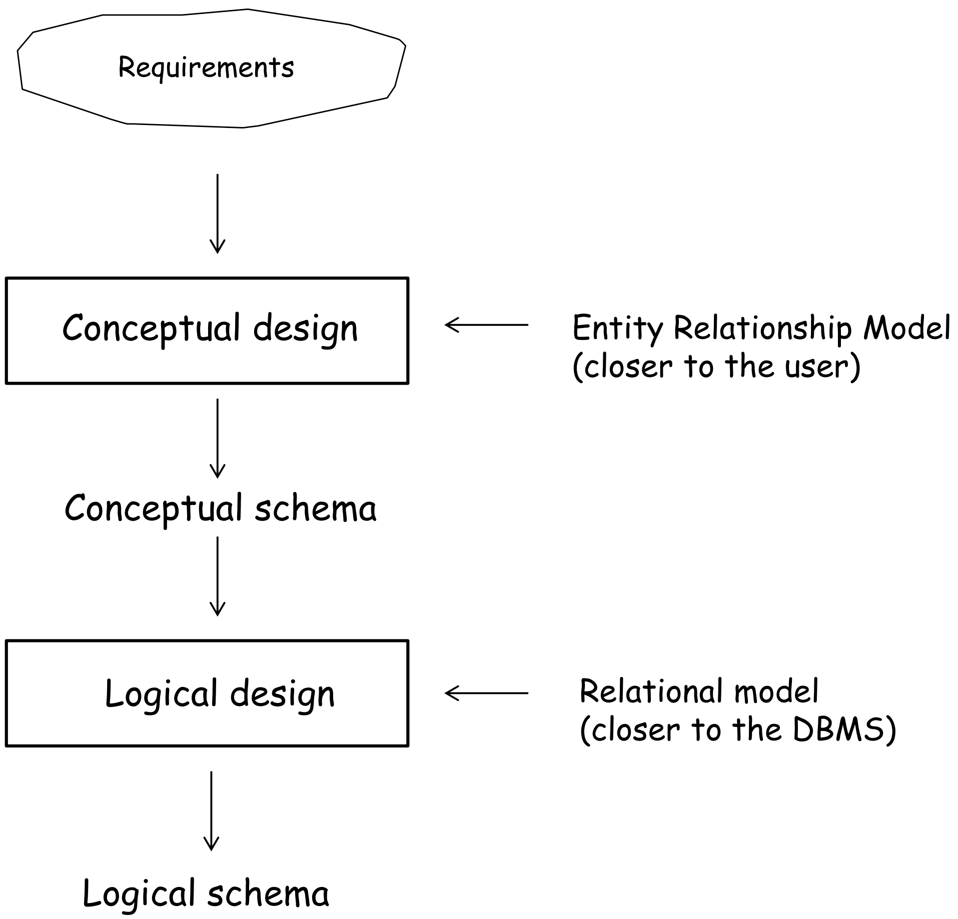
3. We can now access and query the database instance to find values we are interested in.
→ *Query phase* E.g. Which surname has the Student with StudentId = 29?



Query phase

Notice that both updates and queries can be specified in order to be processed by the DBMS using the SQL Data Manipulation Language.

Let us come back to the design phase. For reasons we will experience soon it is convenient to split the design phase in two sub phases.



Conceptual design and Logical design

To summarize, we can say that this course on Database Design provides answers to the following questions:

1. How can we represent concepts and values of a piece of reality in a data base? → Issue of MODELS.
2. How can we design a database? → Issue of DESIGN METHODS.

While another course on Database practice (also called in some universities "Database programming" or "Database query languages") provides answers to other two questions:

1. How can we create and update a database instance? Issue of MANIPULATION LANGUAGES
2. How can we query a database? → Issue of QUERY LANGUAGES.

Concepts introduced in Part 1

Part 1 - Basic Concepts

- Basic Structure of a Computer
- Traditional Data Management
  - Redundancy
  - Inconsistency
- Data Base - 1
- Data Base Model
  - Relational Model
    - Relation
    - Attribute
- Data Base – 2
  - Schema
  - Instance
- Data Base Technology
  - Data Base Management System
  - SQL -Data Description Language
  - SQL -Data Manipulation Language
- Data Base Design
  - Design Phase
    - Conceptual Design
      - Entity Relationship Model
    - Logical Design
  - Set up, Query and Update Phase
- Methodology