Lesson 1 - Introduction to databases and database management systems

Maurizio Mancini
m.mancini@di.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

The content of the slides is authored by Prof. Maria de Marsico and by Prof. Marina Moscarini, Sapienza University of Rome

Module topics

we will cover the following main topics:

- relational algebra
- database design: third normal form (3NF)
- physical organization
- concurrency

Structured information

information in electronic format can be recorded as:

- **structured data**: objects are represented by short strings of symbols and numbers
- **unstructured data**: texts written in a natural language

Information System

- component of an organization that is used to manage (acquire, process, store, communicate) information
- normally, the Information System operates in support of the other components of the organization
- the notion of Information System is independent of its computerization
- examples of Information System ...

What happened earlier?

- each application had its own private file

- **file**: sequential organization
- **application**: written in a file management oriented language (Cobol, PL/1)
- **data management**: file system

What happened earlier?

disadvantages:

- **redundancy**: if two applications used the same data, it was replicated
- **inconsistency**: the update of a data item could relate to a single copy of the data
- **data dependency**: each application organized its data according to its intended use

DBMS

- a **Database** (DB) is a set of mutually linked files
- data is organized in different data structures that facilitate their creation, access and updating and optimize the management of physical resources
- the **Database Management System** (DBMS) is a software tool for the management of large amounts (structured, processable, shared) of data

Components of an information system

- Database (DB)
- Database Management System (DBMS)
- Application software
- Computer hardware (e.g., storage devices)
- Personnel developing, managing or using the system

Storage

database systems use files to store data

but

they provide users with an abstract view of the data, so that storage and manipulation details are transparent to them

Structured information

- the structure of the information depends on its use and can be changed over time
- example: to store data about a person over time:
  - first and last name (until a few centuries ago, this was not obvious either)
  - name, surname, date of birth and place of birth
  - tax number
  - ...

Processing information

- **goal**: to facilitate the processing of data based on its properties

- **structured data**
  - individual access to the elements of the structure is possible through queries (interrogations), to retrieve information or perform calculations
  - relationships between individual data items are represented in the record structure

Shared information

- in an organization, each component is interested in a portion of the Information System
- these portions may overlap
- a database is an integrated resource shared by several components
- integration and sharing allow to reduce redundancies (partially or totally replicated data) and consequent inconsistencies

Shared information

- database sharing is never complete:
  - control privacy and access regulation

- database sharing implies the need to manage simultaneous access to the same data: control of concurrency

Information System

- an **Information System** is a set of data physically organized in secondary memory and managed in such a way as to allow its creation, updating and interrogation

Information System

- data is conceptually organized in aggregates of homogeneous information that constitute the components of the information system, and each update operation is targeted to a single aggregate, while a query may involve one or more than one aggregate
- in databases:
  - aggregates of homogeneous information: files
  - indexes: files that allow you to quickly retrieve information from the "main files"

Data vs information

- in computer systems, information is represented in the form of data:
  - raw facts that need to be interpreted and correlated to provide information

- Example:
  - "Maurizio Mancini" and 0649255161 are a string and a number, i.e., two pieces of data
  - if they are returned in response to the question "who is the course instructor and what is their telephone number" then they constitute information

Data Model

- structures to be used to organize the data of interest and their relationships

- essential component: type constructors
    - example: the relational model provides the relation builder: it organizes data as a set of homogeneous records (types)

Two main types of models

- **logical models**: independent of physical structures but available in DBMSs: e.g. network, hierarchical, **relational**, object-oriented

- **conceptual models**: independent from the modalities of realization, they have the scope to represent the entities of the real world and their relations in the first phases of the planning: e.g., **Entity-Relationship**

Historical notes

mid-60s: first systems

hierarchical

Generalized Update Access Method
(IBM, Project Apollo, 1964)

⇩

DL/1 (Data Language 1)
(IBM, on the market in 1966)

⇩

IMS (Information Management System)

netted

I-D-S (Integrated Data Store)
(General Electric)

⇩

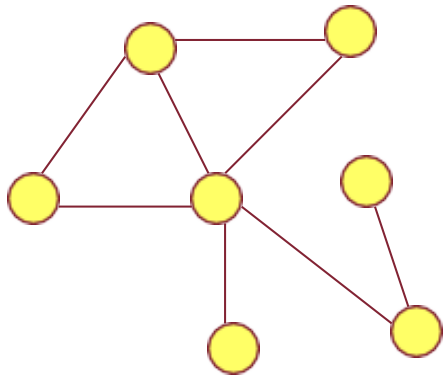
CODASYL / DBTG / network systems

Mesh model

- the data is represented as a collection of *records* of homogeneous type
- binary relationships are represented as *links* (implemented as pointers = dependence on the physical structure of the database)

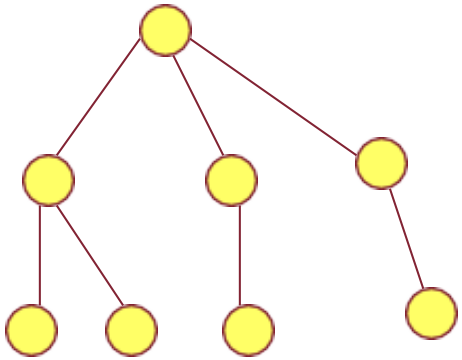
- the model is represented as a graph structure where :
  - nodes = records
  - arches = link

- the most popular mesh model: CODASYL



Hierarchical model

- restricted type of mesh model:
  - hierarchy = mesh composed of a collection of trees (forest)
  - each node has only one parent



Historical notes

- 1970: E.F. Codd (IBM) introduces the relational model
- 70s: relational systems prototypes (System R, IBM)
- 80's: commercial relational systems (Ingress, Oracle, ...)

Relational model

- data and relationships are represented as **values**
- there are no explicit references, i.e. pointers as in the mesh and hierarchical models

- => higher level representation

Relational model

- object = record
- fields/attributes = information of interest

- object = "staff member"
- information of interest = code, surname, first name, role, hiring year

CODE	SURNAME	NAME	ROLE	HIRING
COD1	Rossi	Mario	Analyst	1995

Relational model

- table = set of records of homogeneous type

- staff table = set of records of type "staff member"

CODE	SURNAME	NAME	ROLE	HIRING
COD1	Rossi	Mario	Analyst	1995
COD2	Bianchi	Peter	Analyst	1990
COD3	Neri	Paolo	Admin Hours	1985

Example of relational DB

STUDENTS

Matric	Last name	Name	Birthday
276545	Smith	Mary	25/11/1980
485745	Black	Anna	23/04/1981
200768	Greens	Paolo	12/02/1981
587614	Smith	Lucy	10/10/1980
937653	Brown	Mavis	01/12/1980

COURSES

Code	Title	Tutor
01	Physics	Grant
03	Chemistry	Beale
04	Chemistry	Clark

EXAMS

Stud	Vote	Course
276545	C	01
276545	B	04
937653	B	01

Example of relational DB

STUDENTS

Matric	Last name	Name	Birthday
276545	Smith	Mary	25/11/1980
485745	Black	Anna	23/04/1981
200768	Greens	Paolo	12/02/1981
587614	Smith	Lucy	10/10/1980
937653	Brown	Mavis	01/12/1980

COURSES

Code	Title	Tutor
01	Physics	Grant
03	Chemistry	Beale
04	Chemistry	Clark

EXAMS

Stud	Vote	Course
276545	C	01
276545	B	04
937653	B	01

STUDENTS

Matric	Last name	Name	Birthday
276545	Smith	Mary	25/11/1980
485745	Black	Anna	23/04/1981
200768	Greens	Paolo	12/02/1981
587614	Smith	Lucy	10/10/1980
937653	Brown	Mavis	01/12/1980

EXAMS

Stud	Vote	Course
	C	
	B	
	B	
	B	

COURSES

Code	Title	Tutor
01	Physics	Grant
03	Chemistry	Beale
04	Chemistry	Clark

Example of mesh DB

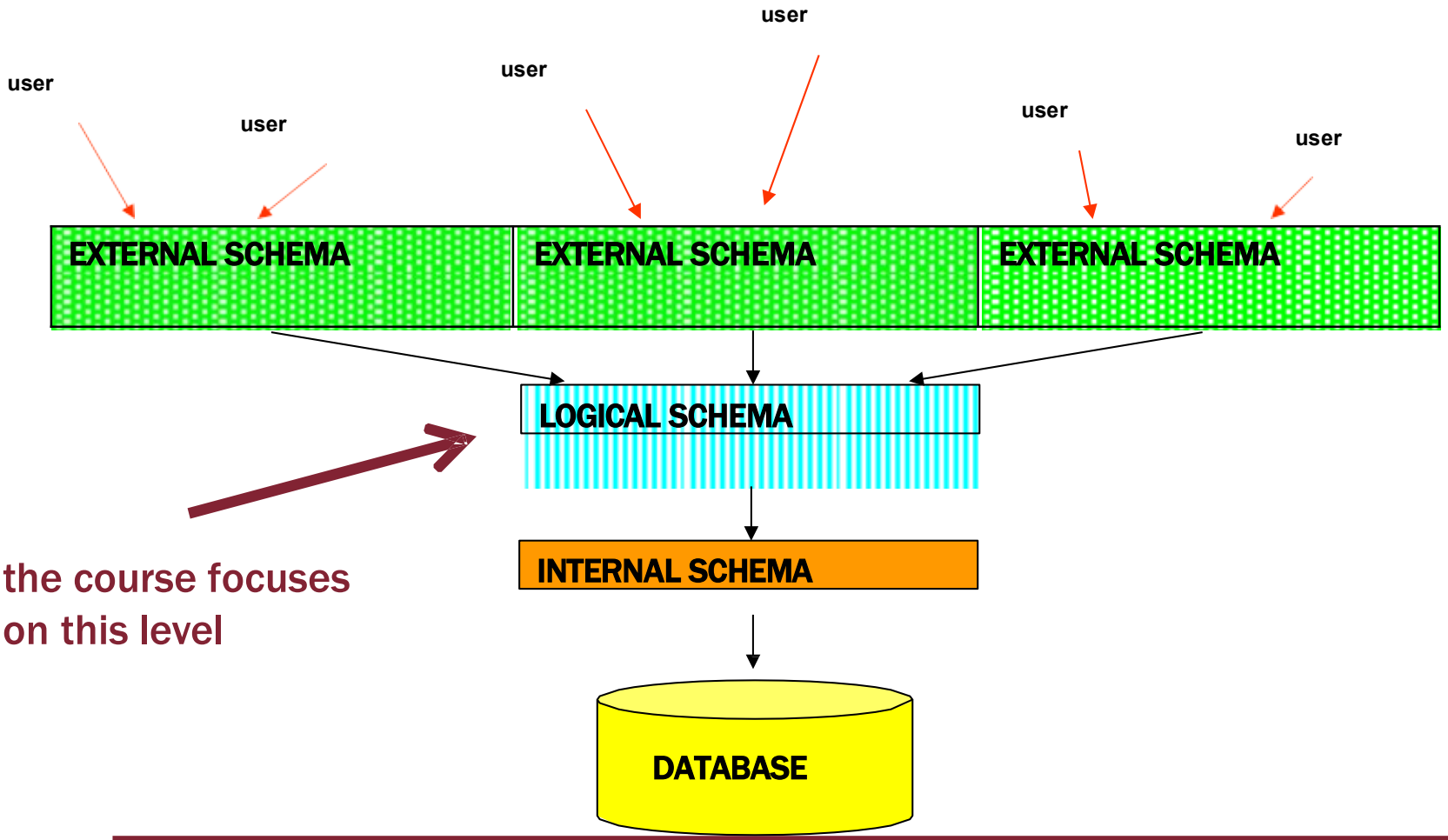
Historical notes

- mid-80s: first object-oriented systems
  (O2, initially INRIA and later O2 Technology)
- starting from '93: definition of a standard
  (Object Data Management Group)

Object model

- model based on objects, classes, etc.
- attributes: describe the state of an object
- methods (actions) describe the behavior of an object
- objects encapsulate both states and behaviors
- there is no universally recognised model yet

The three abstraction levels of a DB



The three abstraction levels of a DB

- External schema: description of a portion of the database in a logical model through partial, or derived, "views" that may provide different data organizations than those used in the logical schema, and that reflect the needs and access privileges of particular types of users; more than one external schema may be associated with a logical schema
- Logical schema: description of the entire database in the "main" logical model of the DBMS, e.g., table structure
- Physical schema: representation of the logical schema by means of physical storage structures, i.e., files

A view (external schema)

LOGICAL SCHEMA

Courses

Course	Lecturer	Room
DBs	Mancini	DS1
Systems	Rossi	N3
Networks	Bianchi	N3
Controls	Bruni	G

Classrooms

Name	Building	Floor
DS1	IMO	G
N3	IMO	G
G	Math	1

EXTERNAL SCHEMA

Courses Locations

Course	Room	Building	Plan
Systems	N3	IMO	G
Network	N3	IMO	
Controls	G	Math	1

VIEW

Data access

database access occur only through the external schema, which may coincide completely with the logical schema

Data independence

- **physical independence**

  the logical and external levels are independent from the physical one

  - a relation is used in the same way whatever its physical realization (organization of files and their physical allocation)
  - the physical implementation can change without having to change the programmes

Data independence

- **logical independence**

  the external level is independent of the logical level

  - additions or changes to views do not require changes to the logical layer
  - changes to the logic schema that leave the external schema unchanged are transparent

Schemes and instances

- in every database there exist:
  - the schema, substantially invariant in time, that describes its structure (intentional aspect): in the relational model, table headers = list of attribute names and their types
  - the instance, the current values, which can change also very quickly (extensional aspect): in the relational model, the "body" of each table

Schemes and instances

STAFF

NAME	SURNAME	BIRTH	TOWN
Piero	Naples	22-10-63	Bari
Marco	Bianchi	01-05-54	Rome
Maria	Rossi	09-02-68	Milan
Maria	Bianchi	07-12-70	Bari
Paolo	Sossi	15-03-75	Palermo

Schemes and instances

STAFF

NAME	SURNAME	BIRTH	TOWN
Piero	Naples	22-10-63	Bari
Marco	Bianchi	01-05-54	Rome
Maria	Rossi	09-02-68	Milan
Maria	Bianchi	07-12-70	Bari
Paolo	Sossi	15-03-75	Palermo

SCHEME

Schemes and instances

NAME	SURNAME	BIRTH	TOWN
Piero	Naples	22-10-63	Bari
Marco	Bianchi	01-05-54	Rome
Maria	Rossi	09-02-68	Milan
Maria	Bianchi	07-12-70	Bari
Paolo	Sossi	15-03-75	Palermo

INSTANCE

Database languages

- **data definition language (DDL)**
  - for the definition of **schemes** (logical, external, physical) and other general operations
- **data manipulation language (DML)**
  - for querying and updating **(instances** of) databases

- **SQL** (Structured Query Language) is a standardized language for databases based on the relational model (RDBMS)
- in SQL the two types of functionality are integrated into a single command language

Databases: summarizing

- features
  - multipurpose
  - integration
  - independence of data
  - centralised control (DBA: database administrator)
- advantages
  - minimum redundancy
  - independence of data
  - integrity
  - security

Integrity

- The data must satisfy “constraints” that exist in the context of interest

- a student resides in only one city (functional dependencies)
- the matriculation number uniquely identifies a student (key constraints)
- a grade is a positive integer between 18 and 30 (domain constraints)
- the overtime of an employee is given by the product of the number of hours and the hourly wage
- the salary of an employee cannot decrease (dynamic constraints)

Security

- data must be protected from unauthorized access
- the DBA must consider:
  - current value of the information for the organization
  - who can access what data and in what way
- and then decide:
  - access regulation
  - effects of a violation

Security

- data must be protected from hardware and software malfunctions and from concurrent access to the database

DB Transactions

- **Transaction**: sequence of operations constituting a single logical transaction

"Transfer €1000 from account c1 to account c2"

- search c1
- change balance to balance-1000
- search c2
- change balance to balance+1000

- A transaction must be fully executed (**committed**) or not executed at all (**rolled back**)

Restore

- To restore a correct database value:

- **transaction log** (contains transaction details: values before and after the change)
- **dump** (periodic copy of the database)

Competition

- **Transaction 1: ″Credit euro 1000 to c/c c1″**
- **Transaction 2: ″Credit euro 500 to c/c c1″**

NOTE!!! Once a value has been read, each transaction modifies it in its own memory space.

Transaction 1	Weather	Transaction 2
search c1	t1	
	t2	search c1
change *balance* to *balance+1000*	t3	
	t4	change *balance* to *balance+500*

initial value *balance*: 2000

final value *balance*: 2500

DBA Tasks

- design definition and description of:
  - logic diagram
  - physical scheme
  - sub-schemas and/or views (Data Definition Language)

- maintenance:
  - changes for new requirements or efficiency reasons
  - (routines: load, copy and restore,
  - reorganization, statistics, analysis)

Course content

To recap

course will cover the following main topics:

- relational Algebra: Procedural Query Language
- database design: how to guarantee/verify the Third Normal Form (3NF), how to decompose a schema preserving dependencies and information
- physical organization of data
- concurrency control