DMIF, University of Udine

Data Management for Big Data

Introduction to Databases

Andrea Brunello
andrea.brunello@uniud.it

April 2022

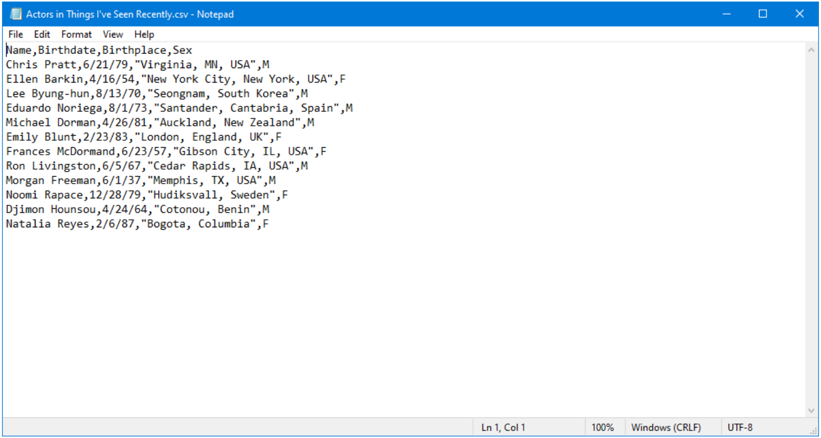
Database Management System (DBMS)

- A DBMS contains information on a particular domain
  - Collection of interrelated data (database)
  - Set of programs to manage the data:
    - Definition of structures to store data
    - Mechanisms to retrieve/modify data
    - Safety and concurrency aspects
- Database applications:
  - Banking: bank transfers, interactions with the ATMs
  - Airlines: reservations, schedules
  - Universities: student registrations, grade assignment
  - Sales: customers, inventory, products and sales
  - E-commerce: e.g, when you buy something from Amazon
- In general, databases can be very large
- They touch all aspects of our lives, although user interfaces hide access details

Example: University database

- A University database could support interactions such as:
  - Add new students, instructors, and courses
  - Register students for courses, and generate class rosters
  - Plan exams, assign grades to students, compute grade point averages
- In the early days, such applications were built directly on top of the file system
- Ad-hoc solutions: data stored in files (e.g., .csv) and applications coded in scripts
  - Over time, they both tend to grow in their number
- First relational database in early 80s, after 10 years of work
  - So, the need for a DBMS was seen from the beginning

CSV file example



Actors in Things I've Seen Recently.csv - Notepad
File Edit Format View Help
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

```
Name,Birthdate,Birthplace,Sex
Chris Pratt,6/21/79,"Virginia, MN, USA",M
Ellen Barkin,4/16/54,"New York City, New York, USA",F
Lee Byung-hun,8/13/70,"Seongnam, South Korea",M
Eduardo Noriega,8/1/73,"Santander, Cantabria, Spain",M
Michael Dorman,4/26/81,"Auckland, New Zealand",M
Emily Blunt,2/23/83,"London, England, UK",F
Frances McDormand,6/23/57,"Gibson City, IL, USA",F
Ron Livingston,6/5/67,"Cedar Rapids, IA, USA",M
Morgan Freeman,6/1/37,"Memphis, TX, USA",M
Noomi Rapace,12/28/79,"Hudiksvall, Sweden",F
Djimon Hounsou,4/24/64,"Cotonou, Benin",M
Natalia Reyes,2/6/87,"Bogota, Columbia",F
```

Drawbacks of using file system to store data

- Data redundancy
  - Multiple file formats and programming languages
  - Duplication of information in different files
  - Other than wasting memory, brings to inconsistencies
- Difficulty in accessing data
  - Need to write a new program to carry out each new task
    - E.g., a program to get info about all professors of Computer Science, another for those teaching Economy, . . .
    - Also, data access is not “interactive”
- Data isolation
  - Multiple files and formats are difficult to manage
- Constraints management
  - Integrity constraints (e.g., account balance > 0) become “buried” in program code rather being stated explicitly
  - Hard to add new constraints or change existing ones

Drawbacks of using file system to store data (Cont.)

- Atomicity of updates
  - Failures may leave the database in an inconsistent state with partial updates carried out
  - E.g., transfer of funds from one account to another should either complete or not happen at all
- Concurrent access by multiple users
  - What happens if two users try to access and modify the same data?
  - E.g., two users booking the last airline ticket
  - For efficiency reasons, we must support concurrency
- Security problems
  - Hard to provide a user access to some, but not all, data

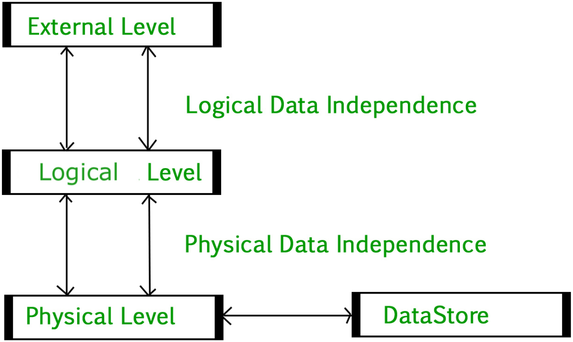
⇝ DBMSs provide solutions to all these problems

Abstractions

- To tackle the complexity inherent in their development, DBMSs make use of **abstractions**
- Abstractions are a fundamental concept in computer science, for instance, in the Operating Systems domain
- They keep complexity under control allowing the user to focus on what is important, leaving out irrelevant details that would generate confusion
- For instance, when driving a car, we do not (typically) care about how the engine is operating
- In practice, for what concerns the database realm, it involves stratifying the DBMS into several layers, each with a specific function

DBMS levels of abstraction

- From lowest to highest:
  - **Physical level:** describes *how* data is stored on the disk (e.g., possibly very complex data structures)
  - **Logical level:** describes *what* data is stored and represented in the database and their relationships
    - describes an entire database in terms of a small number of relatively simple structures (data model)
    - e.g., by means of tables in the relational model
    - they are *tables*, we do not care about how they are stored on the disk (the physical level deals with it)
  - **View/External level:** manages the presentation of information to users and programs. Views can also hide information to some kinds of users
- Each level encapsulates/hides the details it deals with, and can take advantage of the functionalities made available by lower levels, without caring about their implementation



- **Physical data independence:** ability to make changes to the physical level without having to modify the logical one
- **Logical data independence:** ability to make changes to the logical level without affecting external views or programs

Instances and schemas

- The levels of abstraction of a database are described by means of schemas:
  - **Physical schema:** it lays out how data are stored physically on a storage system in terms of files and data structures
  - **Logical schema:** it encodes the overall logical structure of the database
    - E.g., the database consists of information about a set of customers (each characterized by name, address, and phone number) and their accounts in a bank; the relationships among them; and, some constraints over them
    - It allows to describe at an abstract level the characteristics of the elements I am interested in dealing with
- While the schemas lay down the “structure of the database” at different levels, the **instance** of the database refers to its actual content at a particular moment
  - E.g., the specific, concrete customers and accounts

Data model

- A data model is collection of tools for describing:
  - Data (and its structure)
  - Data relationships
  - Data constraints
- For the logical schema, we can make use of two main kinds of data models:
  - **Conceptual data models:** they describe in an abstract way a domain without reference to a specific “database technology” (Entity-Relationship model, for DB design)
  - **Logical data models:** they are specific of a given database technology (although still independent from the physical details)
    - Relational model
    - Graph data model
    - Object-based data models
    - Semistructured data models (XML)
    - Older models: Network model, Hierarchical model

Database design

The process of designing a database is typically articulated into different phases:

1. Brainstorming meetings with IT personnel and all interested stakeholders
   - Collection of requirements
   - Design of a conceptual model (e.g., E-R model)
   - The E-R model has a specific notation, the *E-R diagram*, that helps all involved parts to discuss about the future database
2. Translation of the conceptual model into a logical model
   - Typically, a set of translation rules is followed
   - At this stage, a specific DBMS technology must be chosen (e.g., relational DB)
3. Addition to the logical model of details regarding the physical, low-level aspects (e.g., usage of indexes)
   - The physical schema is thus obtained

References

A. Silberschatz, H.F. Korth, S. Sudarshan *Database system concepts*, 7th Edition, 2020.