Introduction to Databases

Databases

2018-2019

Jesús Correas – jcorreas@ucm.es

Departamento de Sistemas Informáticos y Computación
Universidad Complutense de Madrid
(Based on slides from Mercedes García Merayo and Luis Garmendia)

Bibliography

- Basic Bibliography:
  - R. Elmasri, S.B. Navathe. **Fundamentals of Database Systems** (6th ed). Addison-Wesley, 2010. (in Spanish: **Fundamentos de Sistemas de Bases de Datos** (5a ed). Addison-Wesley, 2007).
    Chapters 1 and 2.
- Additional Bibliography:
  - A. Silberschatz , H. F. Korth, S. Sudarshan. **Database Systems Concepts** (5th ed) McGraw-Hill, 2006. (in Spanish: **Fundamentos de Bases de Datos** (5a ed) McGraw-Hill, 2006).
    Chapter 1.

Contents

- Introduction. Information storage and representation.
- Definition of Database. Example.
- Database Management System.
  - Characteristics.
  - Properties.
  - Operation languages.
- Actors in a DBMS.
- Architecture of a DBMS.
- Database Design.

Information storage and representation

- All computer systems need to **represent information,** and to **store and manipulate data.**
- We have used **files** in the subjects on programming:
  - Provide **persistency,**
  - Allow different modes to access the data: **sequential** (text files), **direct** (binary files).
- We have also seen several kinds of **data structures.**
- However, in all cases it is defined in terms of the **physical structure of the storage devices (disks, memory).**
- The structure of the information required by a sw application **can be extremely complex.**

We have to use techniques to **represent and store information** independently of the **device** and the **storage mode.**

An Example of Database

- University database: System for managing university information.
- **Elements** (*entities*): information regarding **students**, **subjects**, **departments**, **professors**, **grades**, **requirements**, ...
- **Information items** (*attributes*): properties that the elements of the system hold: **student age**, **# credits of a subject**, **subject classification (core, elective)**, **itinerary**, **classroom**, ...
- **Relations** between elements:
  - Students enrol in several subjects.
  - Each subject is proposed by a department.
  - A subject must be taught by at least one educator.
  - Each educator works in one department (and only one).
  - Subjects may have enrolment requirements.
  - Each subject (or group) has a maximum number of students that can enrol to it.
  - ...

An Example of Database

- **Operations** that must be available in the university database:
  - Query the list of students enrolled to a given subject.
  - Query the academic record of a student.
  - Compute the average grade of a subject in a given course.
  - Cancel the enrolment of a student to a subject.
  - Create a new subject in the database.
  - Query the students that have passed all compulsory subjects and have an average grade above 8 out of 10 (e.g, for request a grant).
  - Query the students that have been taught by the same educator in two or more subjects.
  - ...

Information Storage and Representation

The **database system** must provide a **solution to the following issues:**

1. Avoid inconsistencies produced by duplicate information: **redundancy.**
2. Allow the representation of **complex relations** among the data in the database.
3. Allow several ways of **efficient access** to data:
   - **Complex queries**, multiple *views* of the data.
4. **Homogeneous and uniform format** of the data.
5. Mechanisms for ensuring **data integrity.**
6. Allow complex operations **as if they were atomic.**
7. Centralized data storage: **allow concurrent access.**
8. **Security** in data accesses (prevent from unauthorized access).
9. **safety and recovery** from system failures.

Definition of Database

- Database systems provide an **integral solution** to all these issues.

Database (DB)

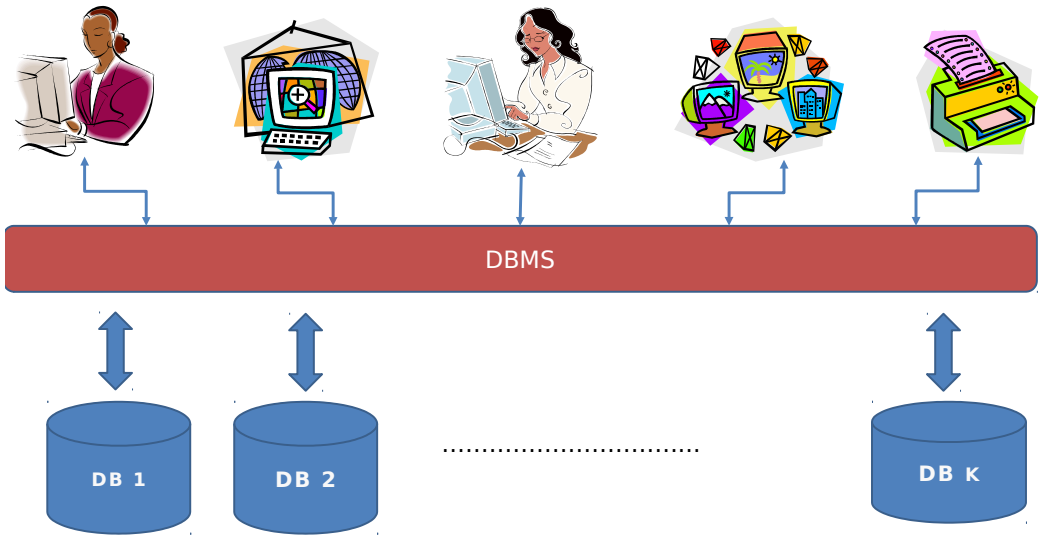
Is a **collection of related data.**

- A DB represents some specific aspect of the real world: the **universe of discourse** (or "miniworld").
- It is a logically coherent colection of data with some inherent meaning.
- It is built with a specific purpose, an intended group of users, and a set of applications for this data.
- We will focus on a specific type of DB: **relational databases.**

Database System

- A **Database Management System (DBMS)** is a **general-purpose set of programs** that allow us to build and maintain a database.
- **General purpose:** the DBMS is the same for any database.

Database System = DB + DBMS



- A Database System provides a series of **characteristics:**

Database System. **Characteristics**

1. A DB System is **self-descriptive:**
   - The definition of a DB is stored in a special DB named **catalog** that contains the structure of de DB.
   - This information is called **meta-data.**
2. **Insulation between programs and data:**
   - The structure of the DB is stored separately from the access programs: we can modify the DB structure without requiring the modification of (all) programs that access to its data.
3. A DB System provides an **abstract view of data:**
   - A **conceptual representation** of data.
   - Hides the storage and implementation details of operations on the data.

Database System. **Characteristics**

4. A DB System supports **multiple views of the data:**
   - A **subset** of the data (e.g., for a professor, show the data of just the students enrolled to his subject).
   - A **partial view** of the data (e.g., hiding confidential information).
   - *Virtual* data, derived from other data: consolidated information, summarized data, etc. that is not actually stored in the DB.
5. **Sharing of data among multiple concurrent users:**
   - The data is stored in a single storage that can be accessed by multiple users and applications.
   - Use of **transactions** to control **concurrent** accesses.
6. **Security** against **unauthorized acceses** and **safety** against system failures.
7. **Efficiency.**

Concurrency. **ACID** Properties

- Relational DBMS can group multiple operations that modify data in a single **transaction.**
  - **Example:** in the University Database, when a student enrols in a subject, the system must:
    - ★ Increment the number of students enroled in the group.
    - ★ Register the subject in the student's academic record.
    - ★ Update the accounting system to allow the payment.
- **A DBMS must provide the ACID properties on transactions:**
  - **Atomicity:** Inside a transaction, either all operations are performed completely, or none of them is performed.
  - **Consistency:** A query must be consistent with the state of the database **when the query starts** executing.
  - **Isolation:** A unfinished transaction is invisible to the rest of the world (other transactions).
  - **Durability:** When a transaction is finished with a commit instruction, it is impossible that the database discards it (unless it is undone by another transaction).

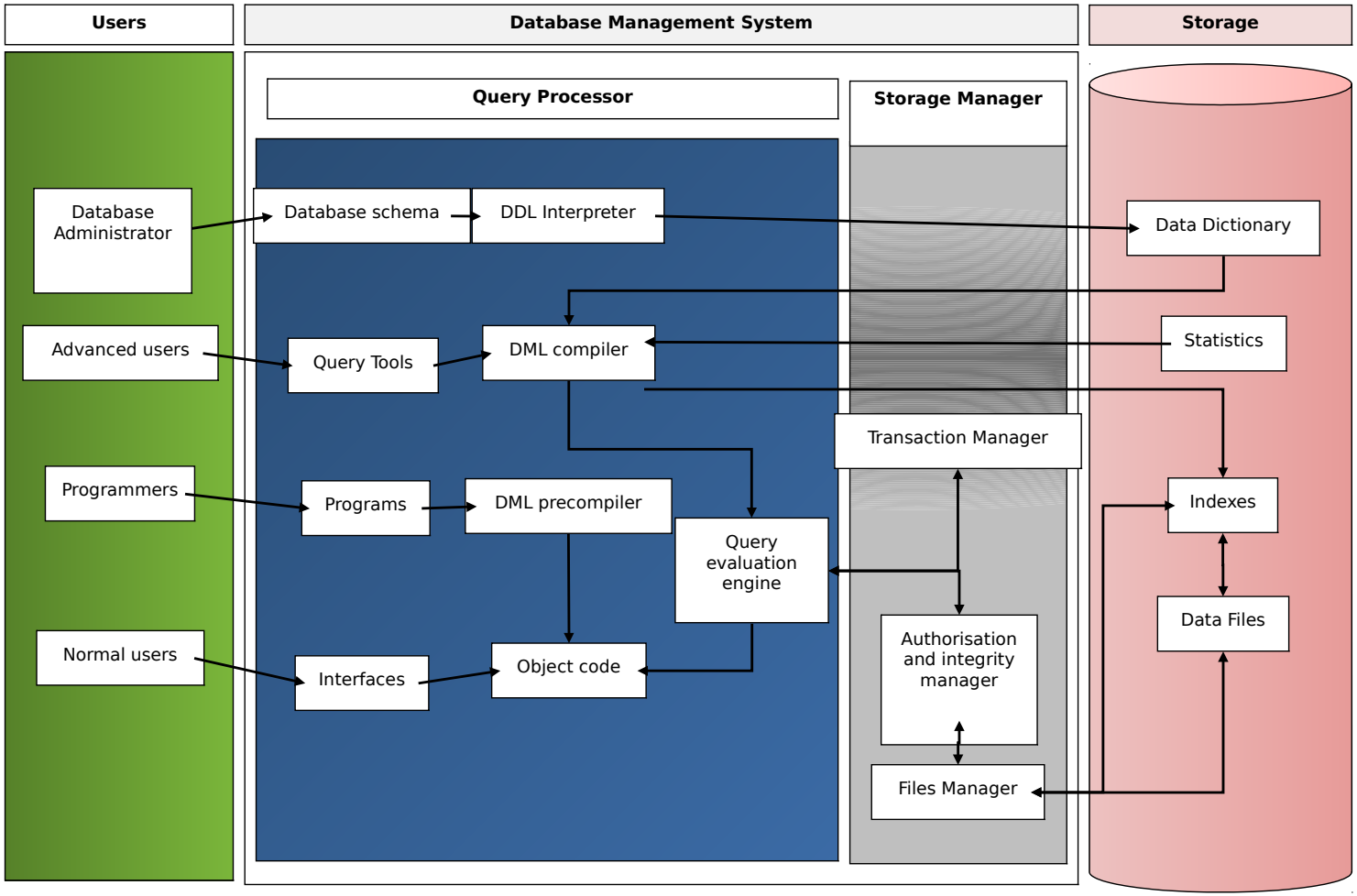
Languages operating on a DBMS

- Several **languages** are used for defining and operating a DB:
  - **DDL:** Data **Definition** Language.
    - ★ Used to define the structure (*schema*) of the DB.
    - ★ Constraints: domain, referential integrity, assertions.
    - ★ Logic and physical access mechanisms (storage).
    - ★ DB object creation and removal (tables, indices, etc.)
  - **DML:** Data **Manipulation** Language.
    - ★ Complex queries:

      ```
      SELECT dpt.name, subj.name FROM dpt, subj
      WHERE dpt.id = subj.idDpt AND subj.credits > 9
      ```

    - ★ Data modification: `INSERT`, `DELETE`, `UPDATE`.
  - **DCL:** Data **Control** Language.
    - ★ Access control to data in a DB.
    - ★ User definitions, roles, user groups.
- The standard language for **relational DB** is **SQL** (*Structured Query Language*), that contains the three languages.

DBMS Structure

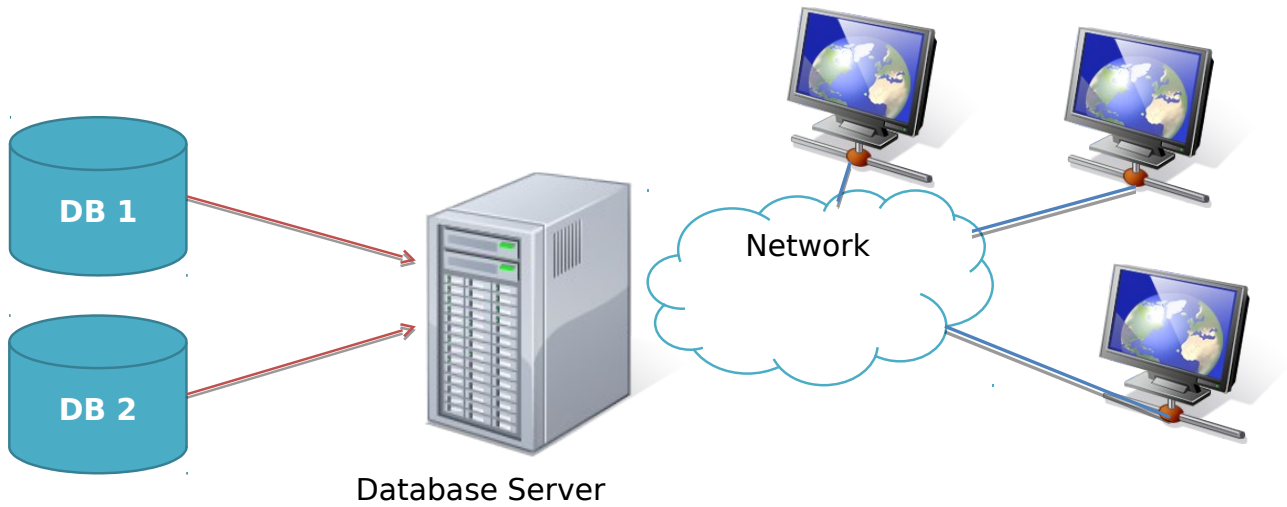


Actors participating in a DB system

- **End users** use the DB by means of applications with preconfigured DML instructions.
- **Advanced users** use a (more sophisticated) query language or SQL.
- **Application developers** create applications that use DML.
- **DB designers** build the conceptual and logic structure of the DB.
- **DB administrators:**
  - Define and modify the DB schema.
  - Define the storage, access and physical organization of the DB.
  - Create user roles and assign access permissions to data.
  - Maintenance: security, safety, *backup & recovery*, efficiency (indices, resource consumption, deadlocks, optimization, etc.).

DBMS Operating Structure

- DBMS use a **client-server architecture:**
  - The clients are application programs, web browsers (invoking server-side associated programs), etc. that make queries to the DB.
  - The server processes requests from clients.
  - DBMS administration tools are also clients that connect to the DBMS.



DBMS Operating Structure

- **Query processor.**
  - Translate SQL sentences into DB internal operations.
  - DML sentences can be **extremely complex:** execution plans must be generated.
  - The most efficient plan (w.r.t. DB statisics) is chosen.
- **Transaction manager and concurrency control.**
  - Supports multiple simultaneous accesses.
  - Guarantees DB consistency (**ACID** properties).
- **Storage manager.**
  - Provides the interface with the low-level DB structures: file system manager, intermediate memory (in Oracle, datafiles, System Global Area, *redo log*, etc.)

Database design

- A database is an essential component of the **information system** of an organization.
- DB (and I.S.) design is performed in several steps:

1. **Requirements Collection and Analysis.**
2. **Conceptual Design.**
   - A **conceptual data model** is built using the requirements.
   - High-level description: physical storage details are not specified.
   - We will use the **Entity-Relationship Model (ER).**
3. **Logical Design.**
   - Translates the conceptual schema into a **logical data model.**
   - **Normalization** techniques are used to detect and solve potential issues.
   - The result is a data model that can be implemented in a specific DBMS.
   - We will use the **Relational Model (MR).**
4. **Physical Design.**
   - Refinement for performance optimization.
   - File organization, indices, de-normalization, etc.

Outline of the rest of the course

In the next classes we will study different aspects of databases in the following order:

2. Conceptual Design: The Entity-Relationship Model.
3. Logical Design: The Relational Database Model. Relational Algebra.
4. SQL: Structured Query Language.
5. Introduction to PL/SQL and Triggers.
6. Introduction to Transactions and Concurrency Control.
7. Advanced Concepts.