Database



Dr. Abdelhafid Dakhia

University of Mohamed Khider, Biskra

Faculty of Exact Sciences

Department of Computer science

Email: abdelhafid.dakhia@univ-biskra.dz

2.0

27 April 2025

Table of contents

I - Chapter I: General information about Databases 3

1. Specific Objectives for Chapter I 3
2. Problems posed by Classic Organizations: SGF 4

2.1. Limitations of Traditional File-Based Systems 4
2.2. Information Redundancy 4
2.3. Dependency Between Data and Programs 4

3. Limits of the File System Approach 6

3.1. Problems of Classical Data Management 6

4. What Is a Database? 7

4.1. Definition of a Database 7
4.2. Database: Applications 7

5. Database Management System 8

5.1. Generalities 8
5.2. DBMS: Objectives 8
5.3. Different Types of DBMS Users 9
5.4. DBMS Data Description Levels 9
5.5. Relational DBMS 11
5.6. Relational DBMS Architecture 12
5.7. Main DBMS Functions 12
5.8. Functional Architecture of a Relational DBMS 13

Glossary 14

References 15

Bibliography 16

Web bibliography 17

I Chapter I: General information about Databases

1. Specific Objectives for Chapter I

Fundamental

- Identify the definition of databases and their fundamental role in information systems.
- Explain different types of database models and their distinguishing features.
- Use knowledge of database components to describe how data is structured and accessed.
- Analyse the organization and operation of database management systems in various contexts.
- Evaluate the effectiveness of database management in improving data handling, consistency, and accessibility.

2. Problems posed by Classic Organizations: SGF

2.1. Limitations of Traditional File-Based Systems

In traditional organizations, implementing a computer application usually involved writing a set of programs designed* to work with a collection of files, which also had to be created.

The main problems with this approach were the redundancy of information and the dependency between the data and the programs that handled them.

2.2. Information Redundancy

Redundancy, i.e. the same information can be found in several separate files.

Example

An employee's identity (surname, first name, address, etc.) can appear in an F1 file specific to a P1 payroll application, and in an F2 file specific to another P2 application for managing medical expense reimbursements.

Une telle situation complique les opérations de mise à jour car il est nécessaire pour une même information mise à jour dans un fichier de la mettre à jour dans tous les fichiers où elle est supposée exister.

Example

If we change an employee's address in the F1 file, we must also make this change in all files containing this information.

2.3. Dependency Between Data and Programs

Generally speaking, creating an application that handles files involves a number of operations*, such as :

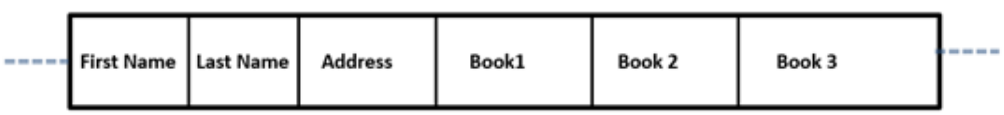
- ***Record structuring:*** defining fields, their types and order
- ***Choice of organization:*** sequential, indexed sequential, direct, etc.
- ***Choice of media:*** hard disk, removable disk, etc.

Note

This means that the data contained in the files is directly associated with the programs that use them, by means of a description contained in the programs themselves.

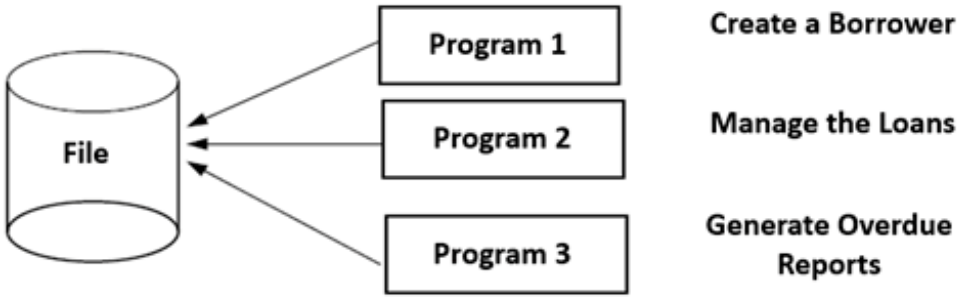
Example

If you have to set up a loan management application in which it is assumed that a borrower can borrow up to three (3) books, you need to create a file with approximately the structure shown in the figure below.



Programs that use such a file define its structure within their own code.

- If this file is used by programs P1, P2, and P3, and a modification is needed to meet the requirements of P2 (for example, to include the borrower's age), then P1 and P3 will also need to be updated. This is because all the programs access the file in the same way, even if they don't require the same information.
- In the same way, a change in the organization of the file on disk for the needs of a Pi program (moving, for example, from a sequential organization to a direct organization in order to optimize Pi's access times) will force the other programs to take this change into account and, in a way, to follow Pi, even if they don't really need this type of organization.
- In the same way, a reorganization of the record's fields (changing the order of the fields within the record) will lead to a modification in the programs and a reconstruction of the file on the medium.



3. Limits of the File System Approach

3.1. Problems of Classical Data Management

Redundant Information

Example: In a financial institution, an employee's address and phone number might appear in both the payroll system file and the personnel management file.

Cumbersome Updating

Example: When an employee's address changes, the update must be made in all relevant files, increasing the risk of errors.

Difficulty Accessing Data

A specific program is required for each new request or report. This makes the system inflexible and slow to adapt.

Dependency Between Data and Programs

Changes to data structures require modifying all associated programs.

Lack of Security

There is no built-in protection from unauthorized access.

Example: Staff might access payroll data when they shouldn't.

Data Integrity Issues

It's hard to enforce rules or constraints.

Example: The account balance should never drop below zero, but there may be no automatic check for that.

4. What Is a Database?

4.1. Definition of a Database

Definition

A database is a collection of data that is structured, organized and stored on a storage medium.*

A database is a structured set of data (1) stored on computer-accessible media (2) to simultaneously satisfy several users (3) selectively (4) at a convenient time (5).

A database fulfills the following functions:

1. Data organization and description.
2. Disk-based storage.
3. Data sharing among users.
4. Confidentiality and security.
5. Performance and accessibility.

Fundamental

The main feature of a database is the sharing of a single description of the data, which can be used by all programs. This is provided by a software program called DBMS.



The purpose of organizing data in a database is to ensure fast and accurate retrieval whenever needed.

A database is often considered the heart of an information system.

4.2. Database: Applications

Today, databases are an integral part of every day life:

management, finance, industry, engineering, law, medicine, library science..., etc.

Examples of applications in the humanities and social sciences :

- Library catalogs.
- Historical databases (historical demography, genealogy, archives).
- Morphological dictionaries for machine translation.
- Terminology databases.

5. Database Management System

5.1. Generalities

Definition

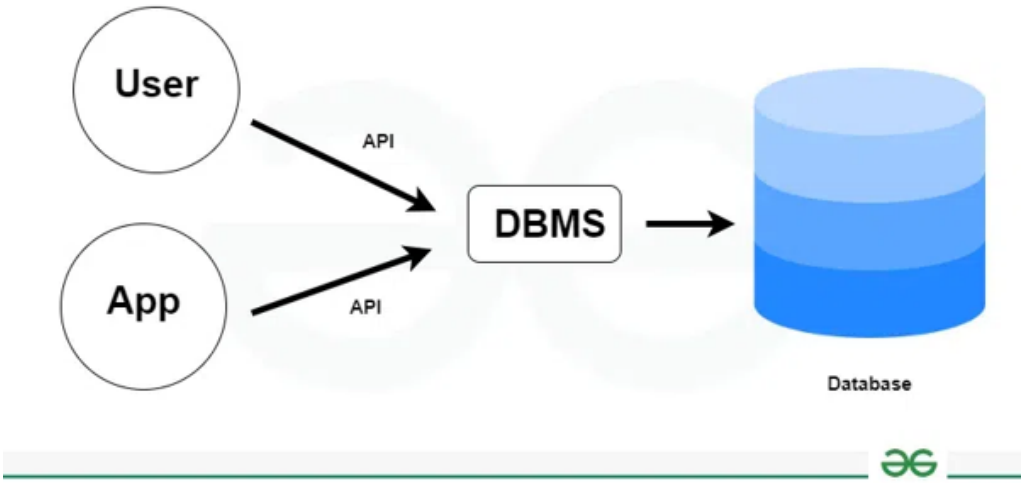
DATABASE MANAGEMENT SYSTEM (DBMS) " Système de Gestion de Base de Données (SGBD).

Operating a database requires the implementation of a computer system commonly referred to as a Database Management System (DBMS).*

- A DBMS is a software program for creating, manipulating and storing databases.
- A DBMS is a coordinated set of software programs for describing, storing, manipulating, processing and querying the data sets that make up the database.

Example

PostgreSQL, MySQL, Oracle, Microsoft SQL Server, Sybase, Informix



See "Getting Started with SQL: Full Training Video"

5.2. DBMS: Objectives

1. ***Data independence***
   Physical independence:
   - Independence between data structures and storage structures.
   - A DBMS makes it possible to describe data in a logical way, without worrying about how this is going to be done physically in the files. (1)*

   Logical independence:
   - The same data set canbe viewed differently by different users.
   - All these personal visions (views) of the data must be integrated into a global vision.
2. ***Manipulation by non-computer specialists:*** You need to be able to access data without knowing how to program, which means using query languages*.

3. *Efficient data access:*
    - These languages must be able to provide answers to queries in a "reasonable" time.
    - They must therefore beoptimized and minimize disk accesses.
4. *Data consistency:*
    - Data is subject to a certain number of constraints that define a coherent state of the database.
    - It must be possible to express these constraints simply, and to check them automatically each time data is inserted, modified or deleted: A person's age greater than zero, salary >0, ...As soon as you try to enter a value that does not respect this constraint, the DBMS rejects the value.
5. *Centralized data administration:*
   Differing views of data (among other things) are more easily resolved if data is administered centrally.
6. *Data sharing:*
    - Enabling multiple users to access the same data at the same time. While this problem is easy to solve when only queries are involved, and in a single-user context, it is no longer the case when modifications are required in a multi-user context. In this case, you need to be able to: Allow two (or more) users to modify the same data "at the same time ensure consistent query results for a user consulting a table while another is modifying it.
7. *Non-redundancy of data:* To avoid problems during updates, each piece of data must only be present once in the database.
8. *Data security:*
   Data must be protected against unauthorized access.
9. *Resistance to failure:*
   A DBMS must integrate fail over mechanisms (e.g. power failure).

5.3. Different Types of DBMS Users

This section looks at the different roles that an individual or group of individuals must play to design, create, implement and operate a database.

The database administrator: is responsible for describing the entities in the database and indicating the links between these entities, using the LDD offered by the DBMS.

The application programmer: is responsible for creating programs to operate the database, based on the description provided by the application administrator. The application programmer uses the LMD offered by the DBMS and other sub-programs generally stored in a library (i.e. sub-program library).

The user: This is the person who simply uses the database, commonly referred to as the End User.

Example

In an air ticket booking agency, the person who types a few commands on his terminal to make a reservation is a user in the same way as a company director, who from time to time requests a certain amount of information from his company's database, reflecting the state of his business (unsold products, pending orders, etc.).

5.4. DBMS Data Description Levels

Let's take an ABC company with four departments:

1. Purchasing
2. Sales
3. Finance
4. HR

All these employees, in their daily tasks, use and manipulate data stored in a database on the company's servers, using a DBMS.

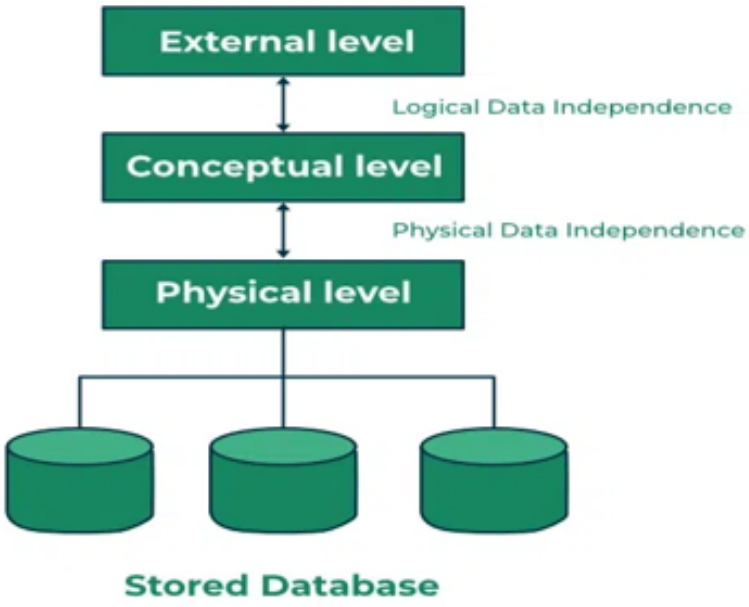
This database contains data from all 4 departments.

An important point is that each employee should only have access to the data he or she is concerned with. For example, an ***HR*** department manager should only have access to HR data, and should not, for example, have access to ***Finance department data***. The same applies to a person who should only have access to finance data, and should not have access to HR data.

This means that each user should only have access to part of the data, and should only view the tables relevant to his or her function. This corresponds to the external data level.

There are commonly three (3) levels of DB description in a DBMS:

1. ***Conceptual level***
2. ***External level***
3. ***Internal level***



Data Independence in DBMS

External diagram

Describes how a user or program perceives the data to which it has access.

→***View: subset of a DB***

View1: HR data

View2: financial data

Conceptual diagram

Describe, in a formal and easily understandable way, the company's data that will be used by its IS.

There are four main classes of data model, distinguished by the nature of the associations they model. These are :

- *Hierarchical models*
- *Network models*
- *Relational models*
- *Object models*

→The conceptual diagram describes :

- Real-world objects.
- Properties of real-world objects
- Links between objects

Internal diagram

The internal schema defines the physical storage structure of the database. The internal schema is a very low-level representation of the entire database. It contains several occurrences of various internal record types.

Describe the data as it is stored in the machine.

- The storage space reserved for each item of information
- Files containing data and their location
- Representation of data in bits.

5.5. Relational DBMS

Fundamental

A database is made up of a set of tables. A table is structured in rows and columns: rows are records and columns are fields or attributes.

Definition

A relational DBMS (RDBMS) is a DBMS whose database is organized in the form of relational tables following the relational model.

Example

SYSTEM-R: from IBM: 1st RDBMS (1975) proving the feasibility of the relational model.

ORACLE→Oracle: (since 1979). Oracle is constantly evolving to include new technologies: object-oriented, internet, big data, cloud...etc.

SQL SERVER: Microsoft SQL SERVER

MYSQL→ Oracle 1st version since 1995.

5.6. Relational DBMS Architecture

Unlike hierarchical, networked DBMSs, the relational approach unifies database ***definition, manipulation and control functions in a single language***.

5.7. Main DBMS Functions

A DBMS is a set of procedures for :

- Description of data and data relations.
- Querying the database.
- Updating.
- Data sharing.
- database data protection.

Today, most RDBMSs use ***SQL*** (Structured Query Language) to perform the main functions of a DBMS.

Data description

The ***DBMS*** provides a data description language ***(LDD)*** for describing data (Type, Length, Nature, etc.), relationships between data, management rules, value domains, etc...

This description is based on a data model.

A data model is a set of concepts used to describe the structure of a database. Most data models include operations for updating and querying the database. The most widely used data model is the relational model.

Data handling

Data Manipulation Language: the DBMS must be equipped with a Data Manipulation Language ***(DML)***, enabling data to be added, deleted or modified. It can also be used to query data.

Control (sharing, integrity, confidentiality, security)

Data sharing: access to the same information by several users at the same time. The DBMS includes a concurrency control mechanism based on data locking techniques (to prevent anyone from reading information that is currently being updated, for example).

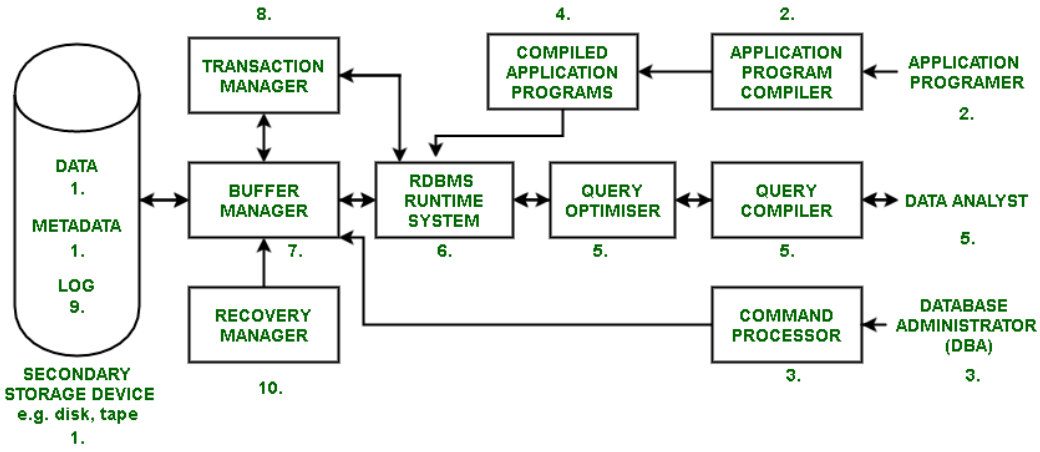
Data integrity through the definition of data constraints. The DBMS ensures that all constraints are checked each time data is inserted, deleted or modified.

Confidentiality: several users can use a database at the same time, raising the issue of data confidentiality. Data rights need to be managed: read, update, create rights, etc., which can be fine-tuned.

(manage data access).

Security: a database is often vital to the functioning of an organization, and it's unacceptable for a breakdown to jeopardize its long-term operation. DBMSs provide mechanisms to ensure this security.

5.8. Functional Architecture of a Relational DBMS



Users interact with the ***DB*** using ***SQL*** commands, either by writing queries directly into an ***SQL*** interface or by embedding queries in an application.

Commands are evaluated in a query evaluation engine. First they are syntactically analyzed, then an algebraic tree is generated.

The optimizer can be used to optimize the execution path of ***a query's*** operations (since the same query can have several execution paths, the optimizer can be used to choose the best one).

Generation of a query execution plan.

The concurrent access manager module ensures that multiple users have concurrent access to the database and generates access conflicts.

The failure manager maintains a log of the various instructions running on the DB to ensure the recovery of the DB in the event of a failure.

The physical space manager generates memory and provides data storage and access.

Glossary

Query Language

A language used to make queries in a database, like SQL (Structured Query Language).

References

(1)
Ramakrishnan, Raghu, Johannes Gehrke, and Johannes Gehrke. Database management systems. Vol. 3. New York: McGraw-Hill, 2003.

Bibliography

Silberschatz, Abraham, Henry F. Korth, and Shashank Sudarshan. "Database system concepts." (2011).

Powell, Gavin. Beginning database design. John Wiley & Sons, 2006.

Date, Chris J. SQL and relational theory: how to write accurate SQL code. " O'Reilly Media, Inc.", 2011.

Web bibliography

https://www.lucidchart.com/pages/tutorial/database-models