Introduction

1. Purpose of Database Systems
2. View of Data
3. Data Models
4. Data Definition Language
5. Data Manipulation Language
6. Transaction Management
7. Storage Management
8. Database Administrator
9. Database Users
10. Overall System Structure

Database Management System (DBMS)

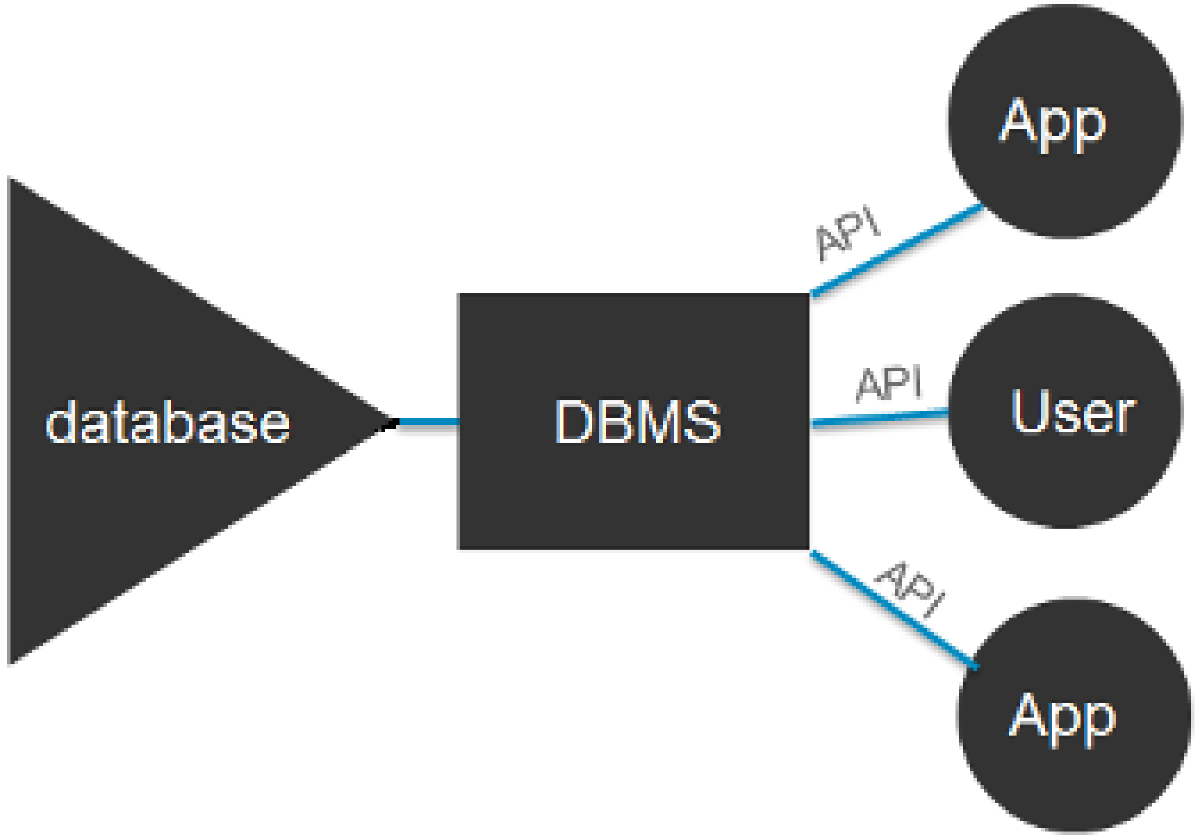
- Collection of interrelated data
- Set of programs to access the data
- DBMS contains information about a particular enterprise
- DBMS provides an environment that is both *convenient* and *efficient* to use.
- Database Applications:
  - Banking: all transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions
- Databases touch all aspects of our lives

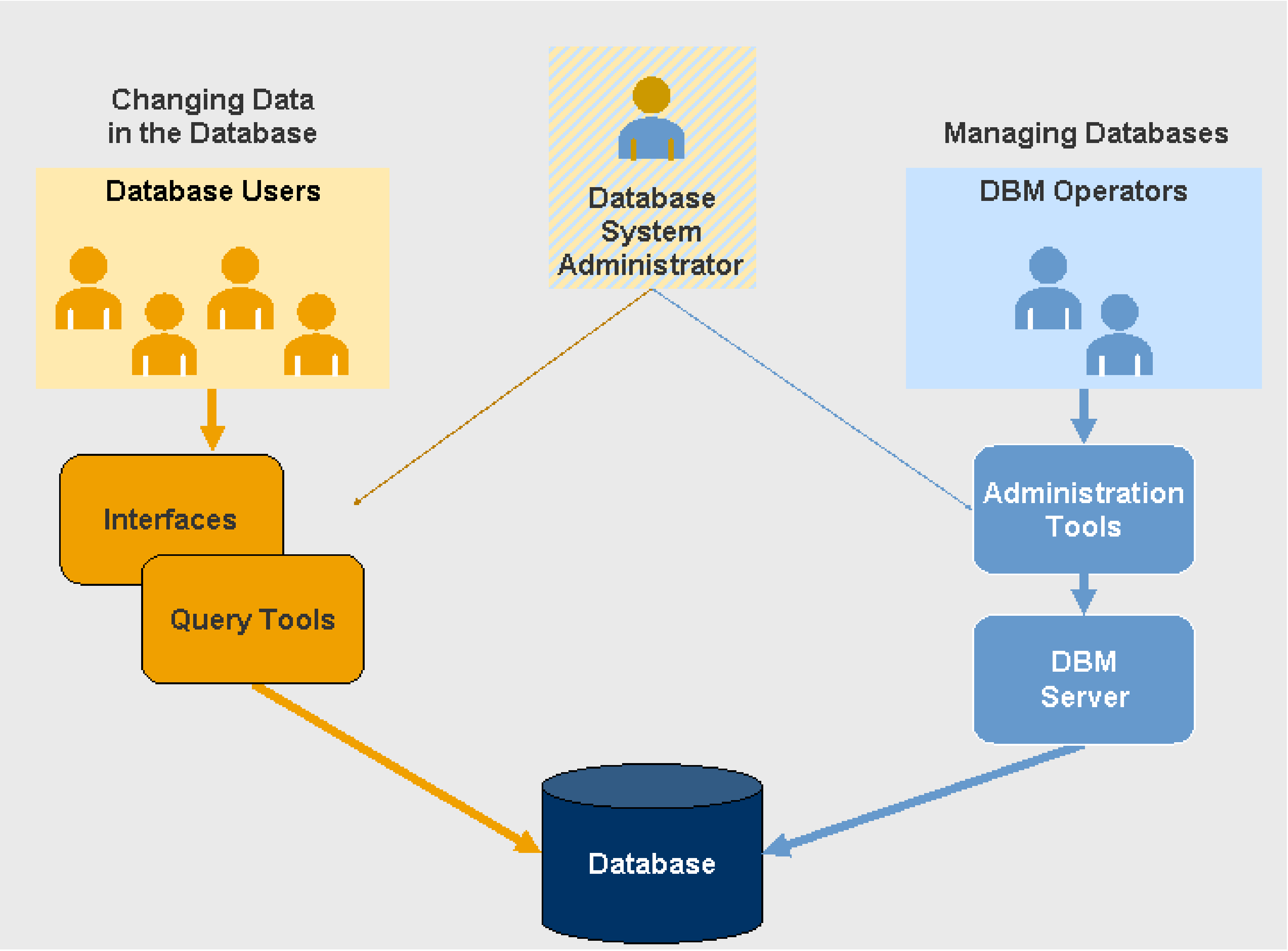
Cutting Edge Database Applications:

1. Cognitive Intelligence

2. Information Retrieval

3. NLP


database
DBMS
API
API
API
App
User
App


Changing Data
in the Database
Database Users
Database
System
Administrator
Managing Databases
DBM Operators
Interfaces
Query Tools
Administration
Tools
DBM
Server
Database

Data and Information:

DATA:

1. Data is the term, that may be new to beginners, but it is very interesting and simple to understand.

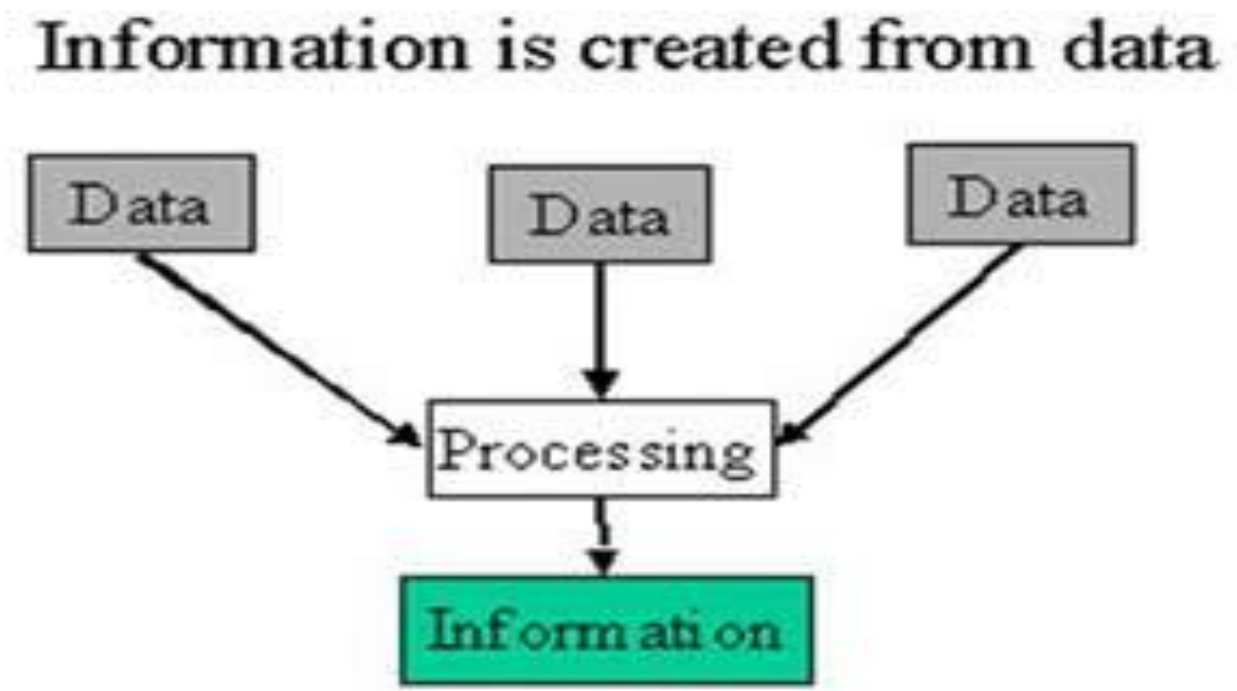
2. It can be anything like name of a person or a place or a number etc.

3. Data is the name given to basic facts and entities such as names and numbers.

4. The main examples of data are weights, prices, costs, numbers of items sold, employee names, product names, addresses, tax codes, registration marks etc.

Information

Information is that which informs. In other words, it is the answer to a question of some kind. It is thus related to data and knowledge, as data represents values attributed to parameters, and knowledge signifies understanding of real things or abstract concepts.



Data	Information
Data is raw fact and figures.	Information is a processes form of data.
For example: 23 is data.	For example: When 23 is stored in row column form as shown below in become information: | Age | 23 |
Data is not significant to a business and of itself.	Information is significant to a business and of itself; for example 23 is insignificant for business but age 23 is significant for a business like music.
Data are atomic level pieces of information.	Information is a collection of data, for example age and 23 collected together to form information.
For example in the healthcare industry, much activity surrounds data collection. Nurses collect data every day and sometimes hourly. Examples of data include vital signs, weight, and relevant assessment parameters.	Information, however, provides answers to questions that guide clinicians to change their practices. For example, the trending of vital signs over time provides a pattern that may lead to certain clinical decisions.
Data does not help in decision making.	As explained above information helps in decision-making.

Purpose of Database System

- In the early days, database applications were built on top of file systems
- Drawbacks of using file systems to store data:
  - Data redundancy and inconsistency
    - Multiple file formats, duplication of information in different files
  - Difficulty in accessing data
    - Need to write a new program to carry out each new task
  - Data isolation — multiple files and formats
  - Integrity problems
    - Integrity constraints (e.g. account balance > 0) become part of program code
    - Hard to add new constraints or change existing ones

Purpose of Database Systems (Cont.)

- Drawbacks of using file systems (cont.)
  - Atomicity of updates
    - Failures may leave 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
    - Concurrent accessed needed for performance
    - Uncontrolled concurrent accesses can lead to inconsistencies
      - E.g. two people reading a balance and updating it at the same time
  - Security problems
- Database systems offer solutions to all the above problems

	File System		DBMS
1	Less no. of files used	1	Large no. of files used
2	Doesn't provide Security	2	Better Security, username and passward
3	Data redundancy is problem Loss of integrity-(accurate and consistent data)	3	Redundancy problem is solved Data is independenat
4	Data is isolated	4	Data is integrated
5	Data access takes lot of time	5	Less time
6	Concurrency control is not possible Single user system	6	Multiple user access data at same time
7	Little preliminary design	7	Vast permanent design
8	Transaction concept not used Eg:	8	Important in DBMS Eg:
9		9	

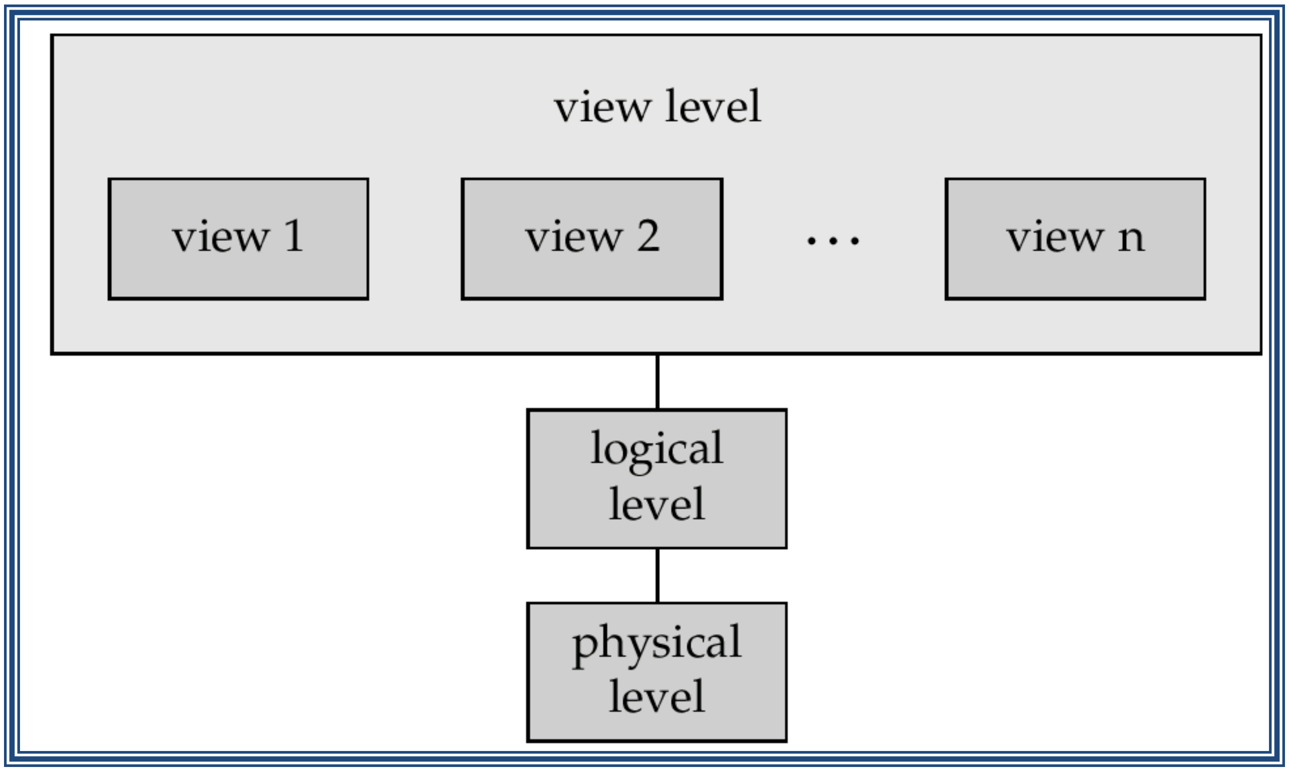
Levels of Abstraction

- Physical level describes how a record (e.g., customer) is stored.
- Logical level: describes data stored in database, and the relationships among the data.

type customer = **record**
name : string;
street : string;
city : integer;
end;

- View level: application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes.

View of Data



An architecture for a database system

Instances and Schemas

- Similar to types and variables in programming languages
- **Schema** – the logical structure of the database
  - e.g., the database consists of information about a set of customers and accounts and the relationship between them)
  - Analogous to type information of a variable in a program
  - **Physical schema**: database design at the physical level
  - **Logical schema**: database design at the logical level
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable

Data Independence-Achievement of Layered Architecture of DBMS

There are two kinds of data independence:
Logical data independence
Physical data independence

Logical data independence
Logical data independence indicates that the conceptual schema can be changed without affecting the existing external schemas. The change would be absorbed by the mapping between the external and conceptual levels.

Physical Data Independence – the ability to modify the physical schema without changing the logical schema

Applications depend on the logical schema

In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Data Models

- A collection of tools for describing
  - data
  - data relationships
  - data semantics
  - data constraints
- Entity-Relationship model
- Relational model
- Other models:
  - object-oriented model
  - Structured or the RDBMS
  - semi-structured data models
  - Unstructured data models
  - Older models: network model and hierarchical model

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram.

- **Relational model**
- **Hierarchical model**
- **Network model**
- **Object-oriented database model**
- **Object-relational model**
- **Entity-relationship model**
- **Other database models**
- **NoSQL database models**
- **Databases on the Web**

Relational model

The most common model, the relational model sorts data into tables, also known as relations, each of which consists of columns and rows. Each column lists an attribute of the entity in question, such as price, zip code, or birth date. Together, the attributes in a relation are called a domain. A particular attribute or combination of attributes is chosen as a primary key that can be referred to in other tables, when it’s called a foreign key.

Each row, also called a tuple, includes data about a specific instance of the entity in question, such as a particular employee.

Hierarchical model

The hierarchical model organizes data into a tree-like structure, where each record has a single parent or root. Sibling records are sorted in a particular order. That order is used as the physical order for storing the database. This model is good for describing many real-world relationships.

Network model

The network model builds on the hierarchical model by allowing many-to-many relationships between linked records, implying multiple parent records. Based on mathematical set theory, the model is constructed with sets of related records. Each set consists of one owner or parent record and one or more member or child records. A record can be a member or child in multiple sets, allowing this model to convey complex relationships.

It was most popular in the 70s after it was formally defined by the Conference on Data Systems Languages (CODASYL).

Object-oriented database model

This model defines a database as a collection of objects, or reusable software elements, with associated features and methods. There are several kinds of object-oriented databases:

A **multimedia database** incorporates media, such as images, that could not be stored in a relational database.

A **hypertext database** allows any object to link to any other object. It’s useful for organizing lots of disparate data, but it’s not ideal for numerical analysis.

The object-oriented database model is the best known post-relational database model, since it incorporates tables, but isn’t limited to tables. Such models are also known as hybrid database models.

Object-relational model

This hybrid database model combines the simplicity of the relational model with some of the advanced functionality of the object-oriented database model. In essence, it allows designers to incorporate objects into the familiar table structure.

Languages and call interfaces include SQL3, vendor languages, ODBC, JDBC, and proprietary call interfaces that are extensions of the languages and interfaces used by the relational model.