Database Concepts

Chapter 1

Content

- Types of Databases and Database Applications
- Basic Definitions
- Typical DBMS Functionality
- Example of a Database
- Main Characteristics of the Database Approach
- Levels of Abstraction
- View of Data
- Instances and Schemas
- Database Users
- Categories of End-users
- Advantages of Using the Database Approach
- Historical Development of Database Technology
- Extending Database Capabilities
- Overall System Structure
- Application Architecture

Problem of Using file-based applications

- DATA REDUNDANCY
- DATA ACCESS DIFFICULTY
- DATA SHARING/ CONCURRENCY
- CONTROL PROBLEMS
- INCOHERENCE RISK
- SECURITY AND DATA
- CONFIDENTIALITY PROBLEMS
- NON PORTABILITY OF THE
- APPLICATIONS
- MAINTENANCE PROBLEMS
- COST MULTIPLICITY

Solution

- Unique and centralized administration of data in a given company

  ➔ DATABASE

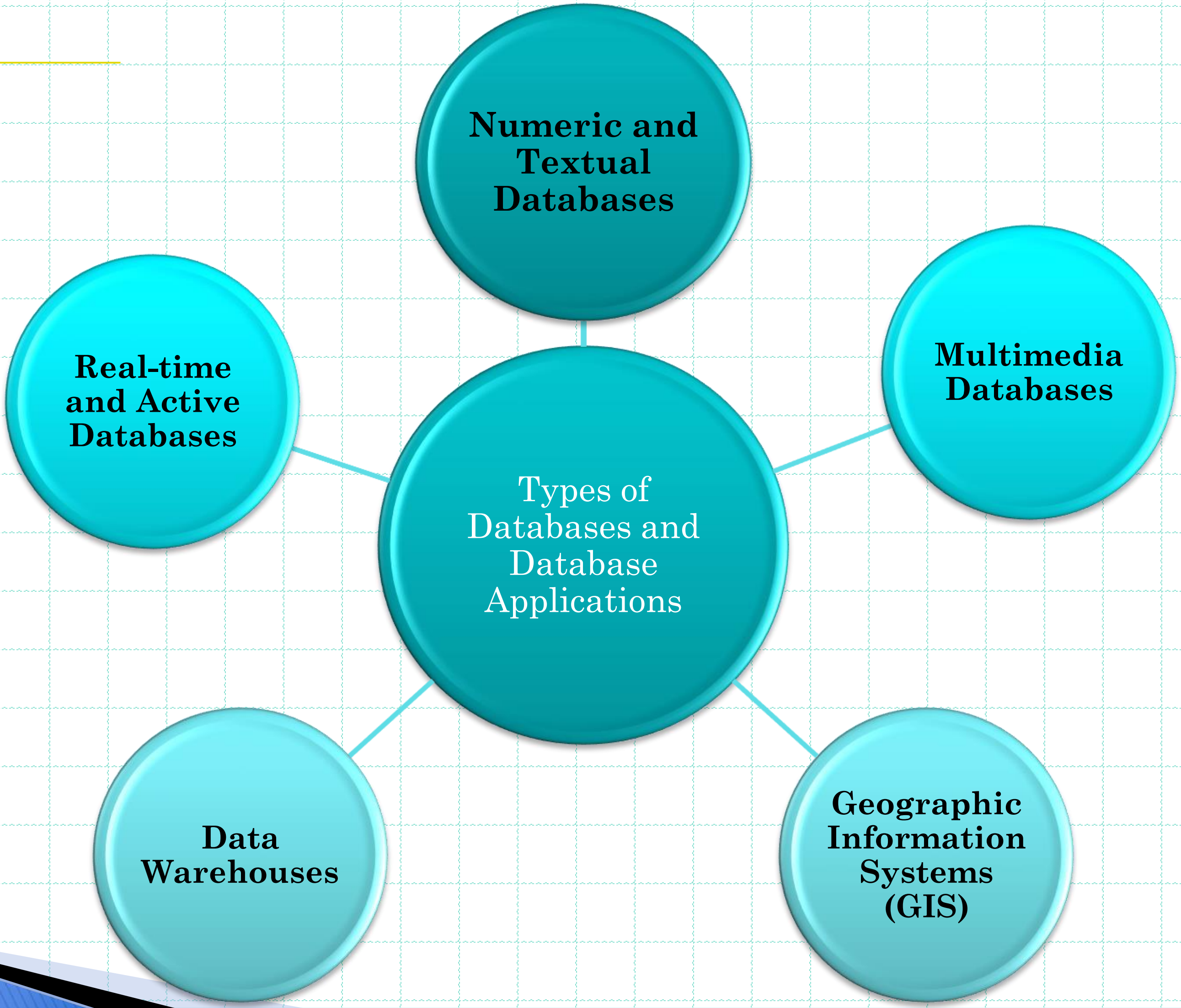
- To do so, it is necessary to look for a canonical data structure

  ➔ DATA MODEL

- Data definition / Manipulation tools for:
  - 1. Administrators (DBA)
  - 2. Designers
  - 3. Application Developers
  - 4. Non Specialists / End users

➔ Database Management System (DBMS)

Types of Databases and Database Applications


Numeric and Textual Databases
Real-time and Active Databases
Types of Databases and Database Applications
Multimedia Databases
Data Warehouses
Geographic Information Systems (GIS)

Basic Definitions

- **Database**: A collection of related data.
- **Data**: Known facts that can be recorded and have an implicit meaning.
- **Database Management System (DBMS)**: A software package/ system to facilitate the creation and maintenance of a computerized database.
- **Database System**: The DBMS software together with the data itself. Sometimes, the applications are also included.
- **Database schema**: A description of a database, specified during the database design

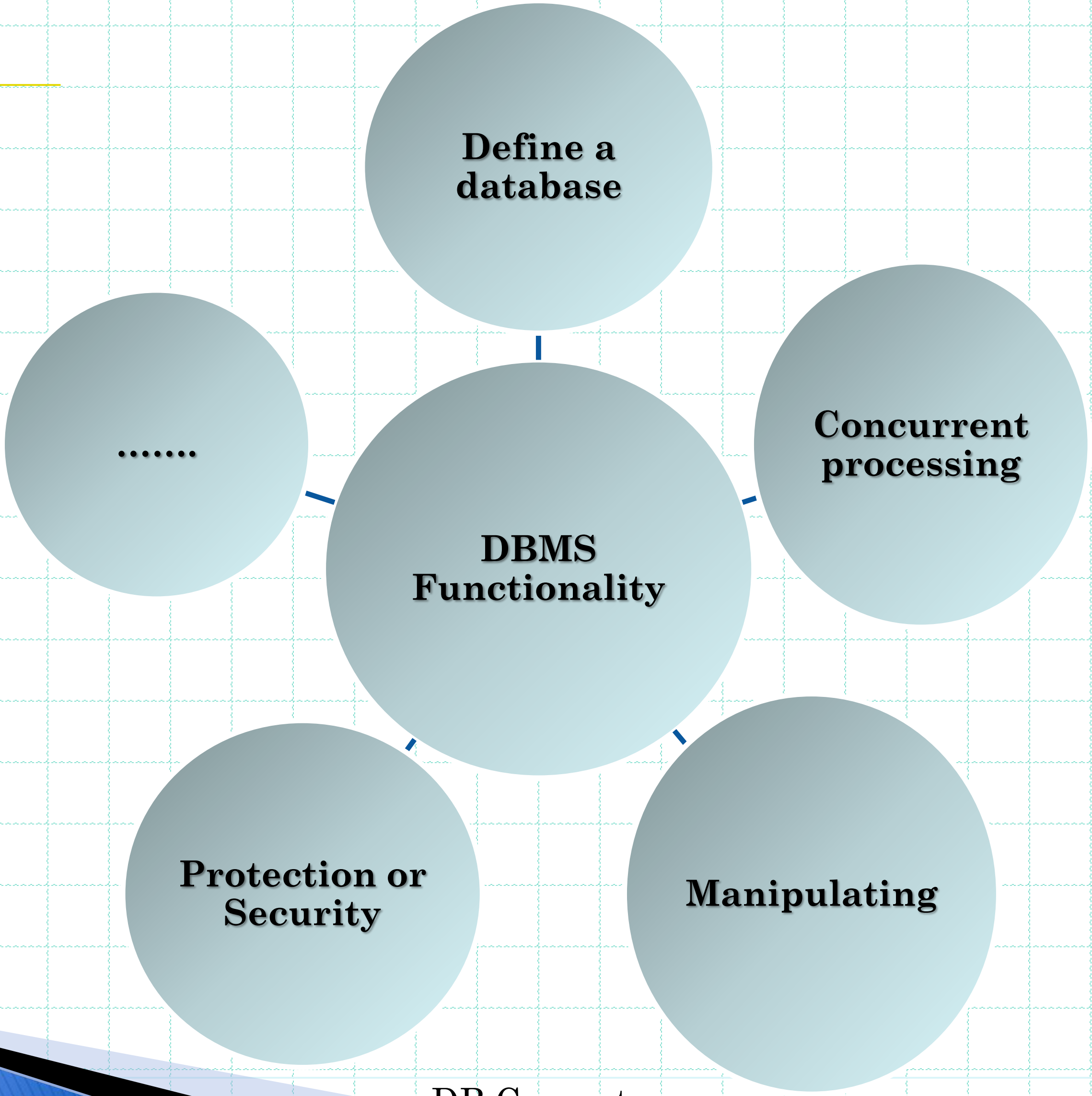
Database Management System (DBMS)

- 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.

DBMS (Cont.)

- 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

Functionalities



Typical DBMS Functionality

- **Define a database** : in terms of data types, structures and constraints
- Construct or Load the Database on a secondary storage medium
- **Manipulating the database** : querying, generating reports, insertions, deletions and modifications to its content
- **Concurrent Processing** and Sharing by a set of users and programs – yet, keeping all data valid and consistent

Typical DBMS Functionality

- **Other features:**
  - Protection or Security measures to prevent unauthorized access
  - “Active” processing to take internal actions on data
  - Presentation and Visualization of data

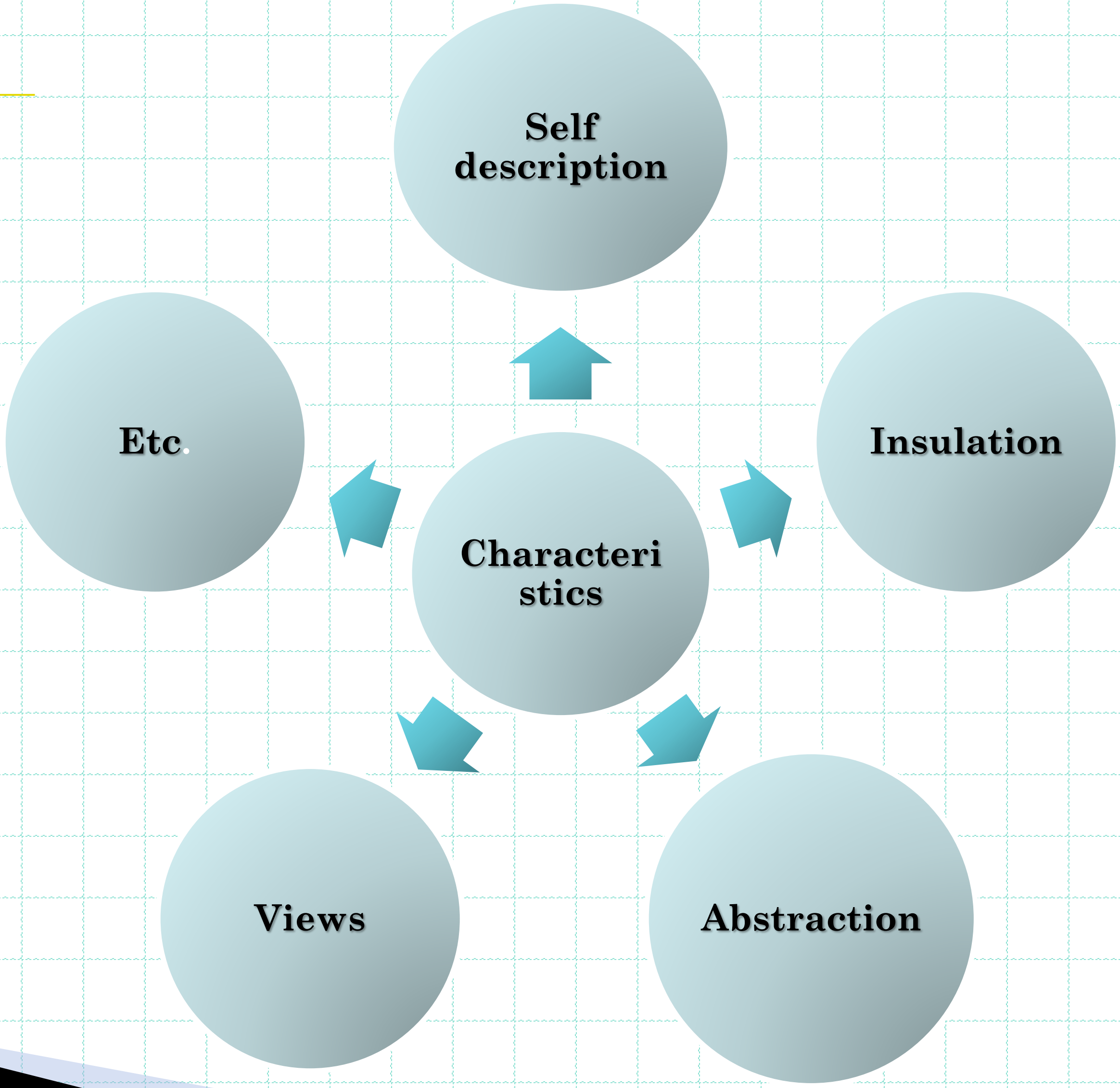
Example of a Database (with a Conceptual Data Model)

- **example**: Part of a UNIVERSITY environment.
- **Some *entities***:
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - (academic) DEPARTMENTs
  - INSTRUCTORs
- *Note*: The above could be expressed in the ENTITY-RELATIONSHIP data model.

Example of a Database

- Some ***relationships***:
  - SECTIONs *are of* specific COURSEs
  - STUDENTs *take* SECTIONs
  - COURSEs *have* prerequisite COURSEs
  - INSTRUCTORs *teach* SECTIONs
  - COURSEs *are offered by* DEPARTMENTs
  - STUDENTs *major in* DEPARTMENTs

Characteristics



Main Characteristics of the Database Approach

- **Self-describing** nature of a database system: A DBMS catalog stores the description of the database. The description is called meta-data). This allows the **DBMS** software to work with different databases.
- **Insulation** between programs and data: Called program-data independence. Allows changing data storage structures and operations without having to change the DBMS access programs.

Main Characteristics of the Database Approach

- Data **Abstraction**: A data model is used to hide storage details and present the users with a conceptual view of the database.
- Support of multiple **views** of the data: Each user may see a different view of the database, which describes only the data of interest to that user.

Levels of Abstraction

- **Physical level** describes how a record is stored. It is the physical storage of a database (Data storage and access paths).
- **Logical level**: describes data stored in database, and the relationships among the data. It hides the details of the physical storage
- **View level**: application programs hide details of data types. Views can also hide information (e.g., salary) for security purposes. Each external schema describes the part of the database that a particular group of users is interest in.

Levels of Abstraction

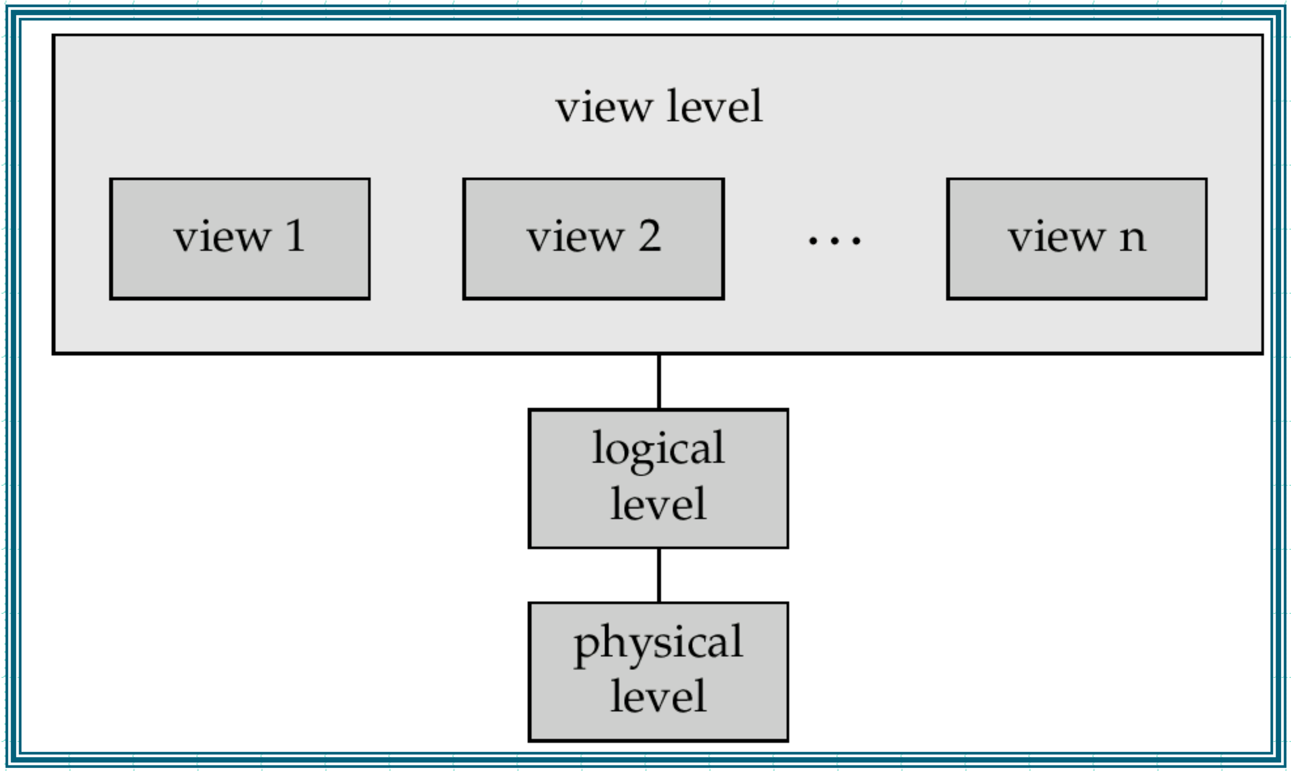
1. INTERNAL /PHYSICAL: Computer reality
   - Files (name, size, organization, ...)
   - Record (name, length, . . .)
   - Field (name, format, . ..)
   - Access path (key, association, . . .)

2. CONCEPTUAL: Reality as it is perceived
   - Entity (name, attributes, ...)
   - Association (name1, name2, . . .)

3. EXTERNAL: Reality as a user group shows it

View of Data

- An architecture for a database system



Instances and Schemas

- **Schema** – the logical structure of the database
  - **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
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema

DATA MODEL

- The independence of data with respect to the processing involves the description of data itself.

- This description could be performed from an abstract view of the reality

- The Need of a Tool to Perform this Description

→DATA MODEL

DATA MODEL

- Data model should allow the representation and the description :
  - Entities and elementary data that constitute / organize it
  - Links (Associations, Relationships, Correspondences, ...)
  - Certain assertion that data should
- It verify: Integrity Constraint

➔ We talk about a Conceptual Data Model

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
  - semi-structured data models
  - Older models: network model and hierarchical model

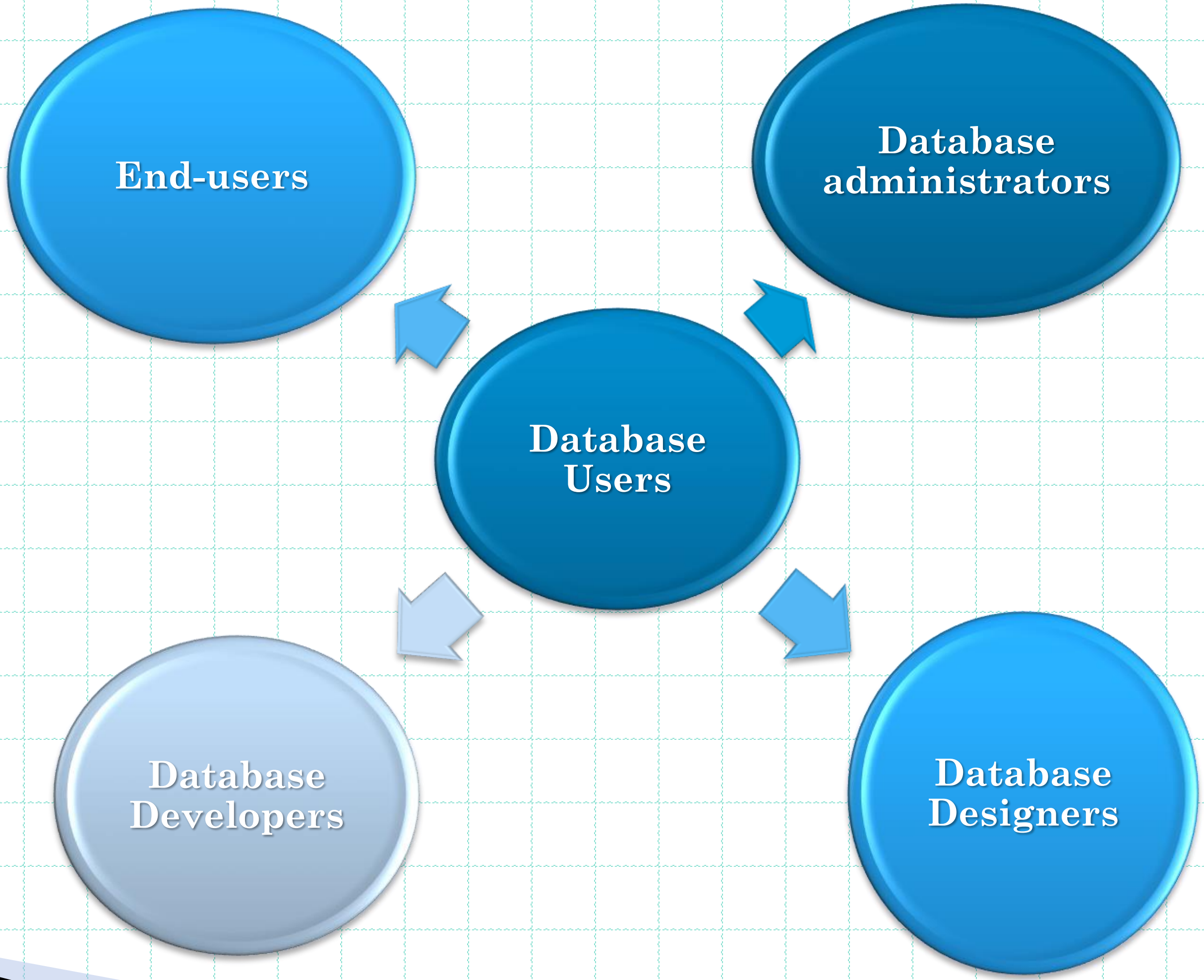
Main Characteristics of the Database Approach

- Sharing of data and multiuser transaction processing : allowing a set of concurrent users to retrieve and to update the database.
- Concurrency control within the DBMS guarantees that each transaction is correctly executed or completely aborted.

Database Users

- Users may be divided into those who actually use and control the content (called “Actors on the Scene”) and those who enable the database to be developed and the DBMS software to be designed and implemented (called “Workers Behind the Scene”)

Database Users



Database Users

- **Database administrators**: responsible for authorizing access to the database, for co-ordinating and monitoring its use, acquiring software, and hardware resources, controlling its use and monitoring efficiency of operations.

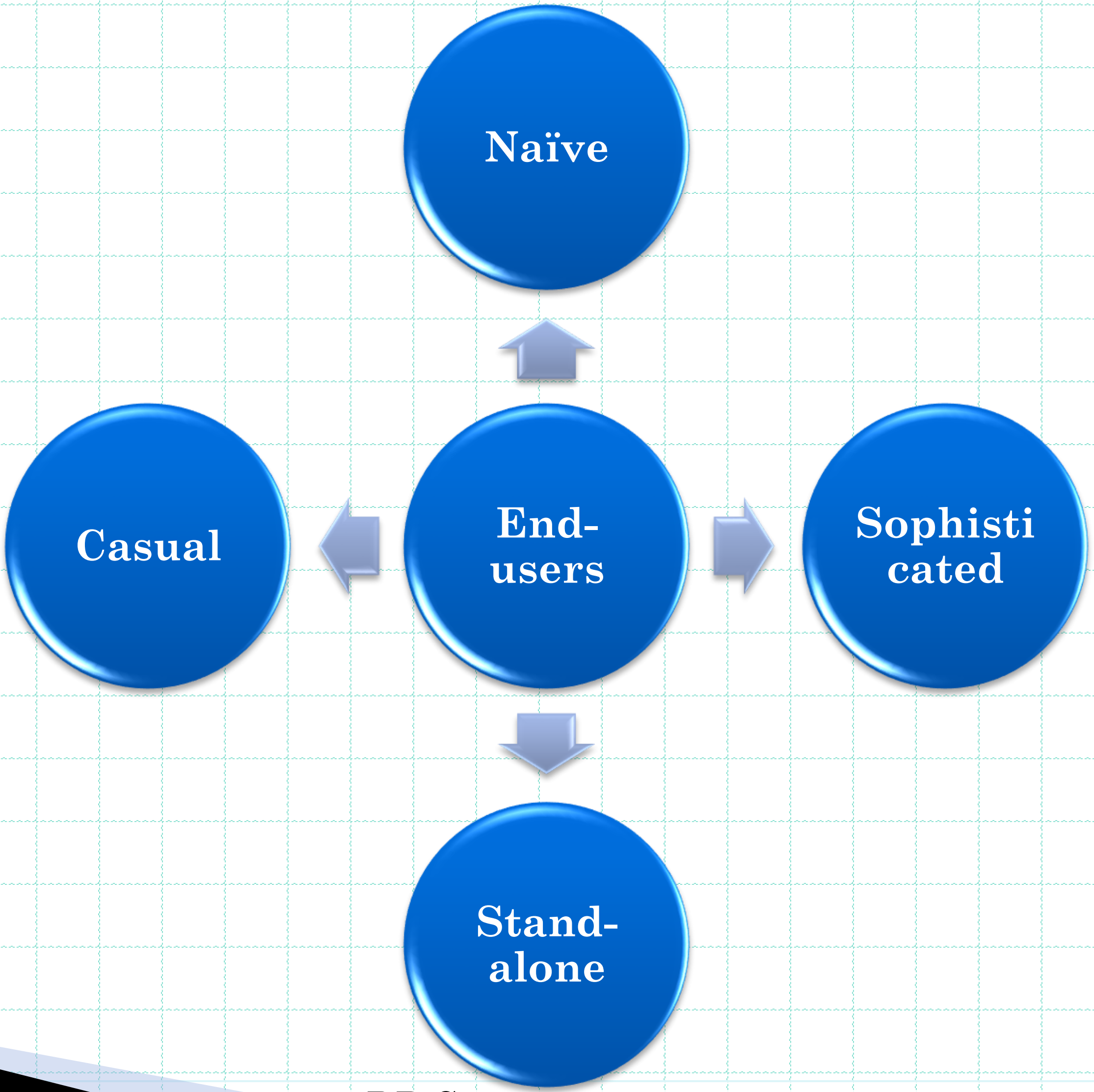
- **Database Designers**: responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

Database Users

- **Database Developers/ Database Programmers**: interact with system through DML calls

- **End-users**: they use the data for queries, reports and some of them actually update the database content.

Categories of End-users



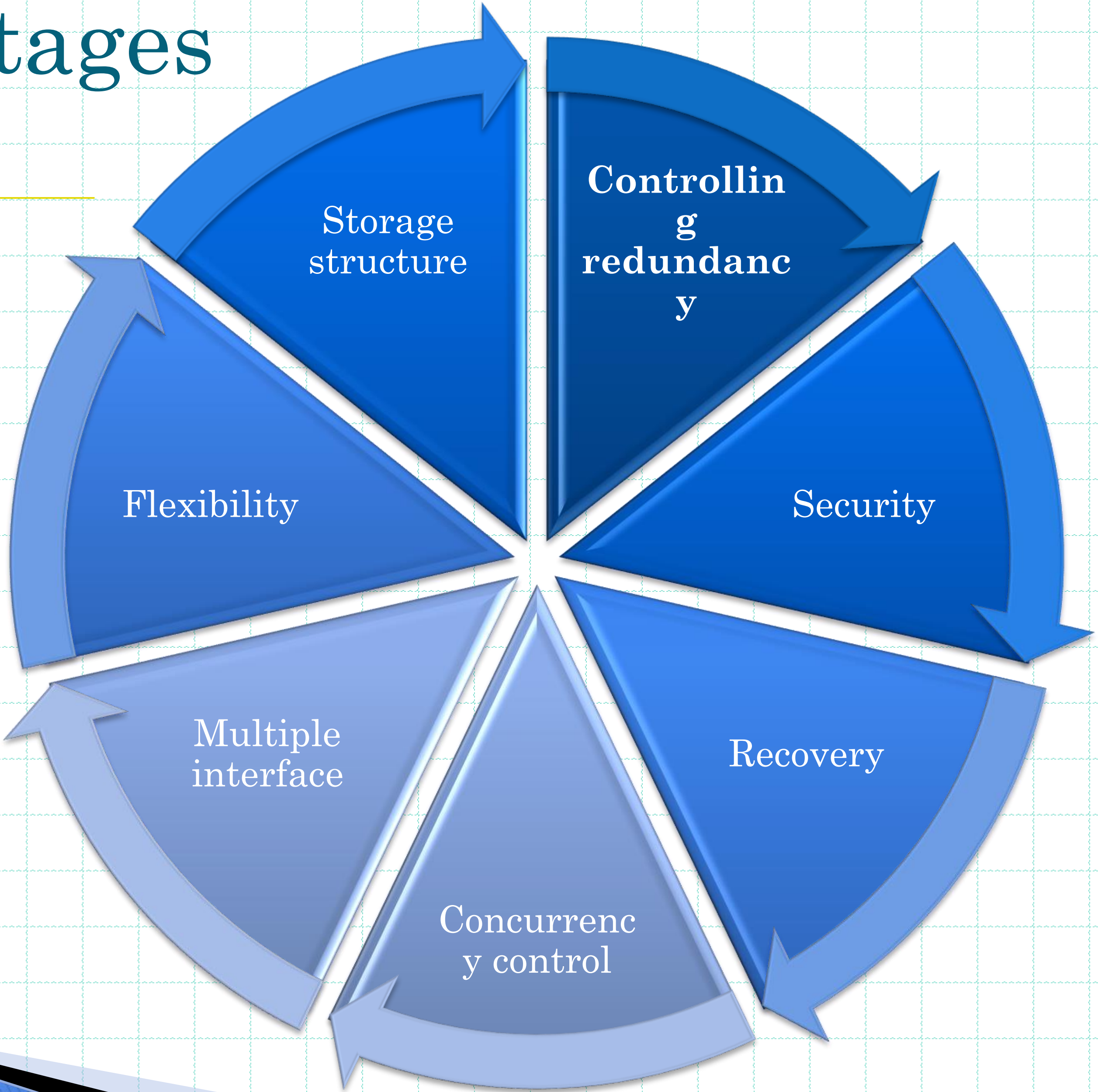
Categories of End-users

- **Casual** : access database occasionally when needed
- **Naïve or Parametric** : they make up a large section of the end-user population. They use previously well-defined functions in the form of "canned transactions" against the database. Examples are bank-tellers or reservation clerks who do this activity for an entire shift of operations.

Categories of End-users

- **Sophisticated** : these include business analysts, scientists, engineers, others thoroughly familiar with the system capabilities. Many use tools in the form of software packages that work closely with the stored database.
- **Stand-alone** : mostly maintain personal databases using ready-to-use packaged applications. An example is a tax program user that creates his or her own internal database.

Advantages



Controlling redundancy

Security

Recovery

Concurrency control

Multiple interface

Flexibility

Storage structure

Advantages of Using the Database Approach

- **Controlling redundancy** in data storage and in development and maintenance efforts.
- **Sharing of data** among multiple users.
- **Restricting unauthorized** access to data.
- Providing **Storage Structures** for efficient Query Processing

Advantages of Using the Database Approach

- Providing **backup** and **recovery** services.
- Providing **multiple interfaces** to different classes of users.
- Representing **complex relationships** among data.
- Enforcing **integrity constraints** on the database.
- Drawing **Inferences** and Actions using rules

Additional Implications of Using the Database Approach

- **Flexibility** to change data structures: database structure may evolve as new requirements are defined.
- **Availability** of up-to-date information – very important for on-line transaction systems such as airline, hotel, car reservations.
- **Economies of scale**: by consolidating data and applications across departments wasteful overlap of resources and personnel can be avoided.

Historical Development of Database Technology

- **Early Database Applications**: The Hierarchical and Network Models were introduced in mid 1960’s and dominated during the seventies. A bulk of the worldwide database processing still occurs using these models.
- **Relational Model based Systems**: The model that was originally introduced in 1970 was heavily researched and experimented with in IBM and the universities. Relational DBMS Products emerged in the 1980’s.

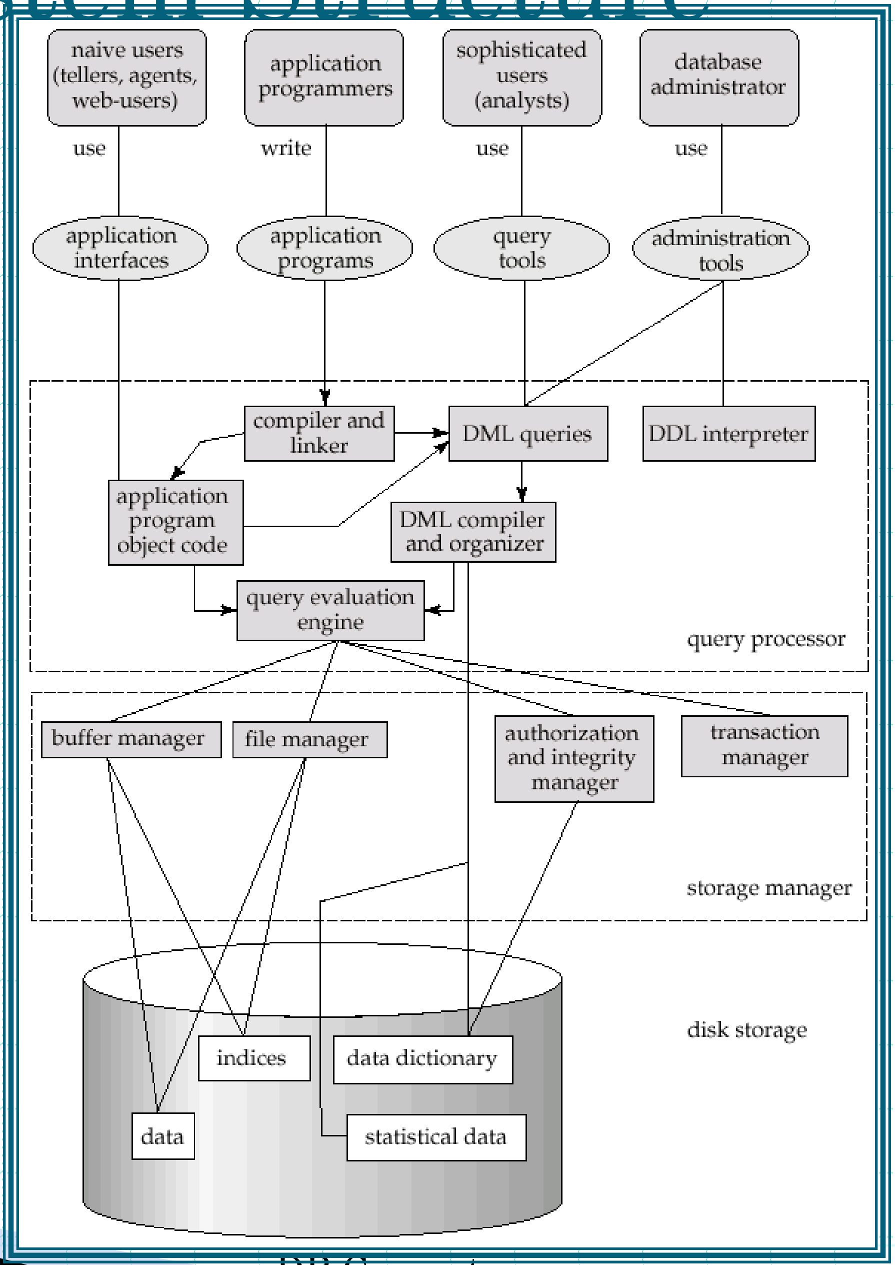
Historical Development of Database Technology

- **Object-oriented** applications: OODBMSs were introduced in late 1980's and early 1990's to cater to the need of complex data processing in CAD and other applications. Their use has not taken off much.
- Data on the **Web** and E-commerce Applications: Web contains data in HTML (Hypertext markup language) with links among pages. This has given rise to a new set of applications and E-commerce is using new standards like XML (eXtended Markup Language).

Extending Database Capabilities

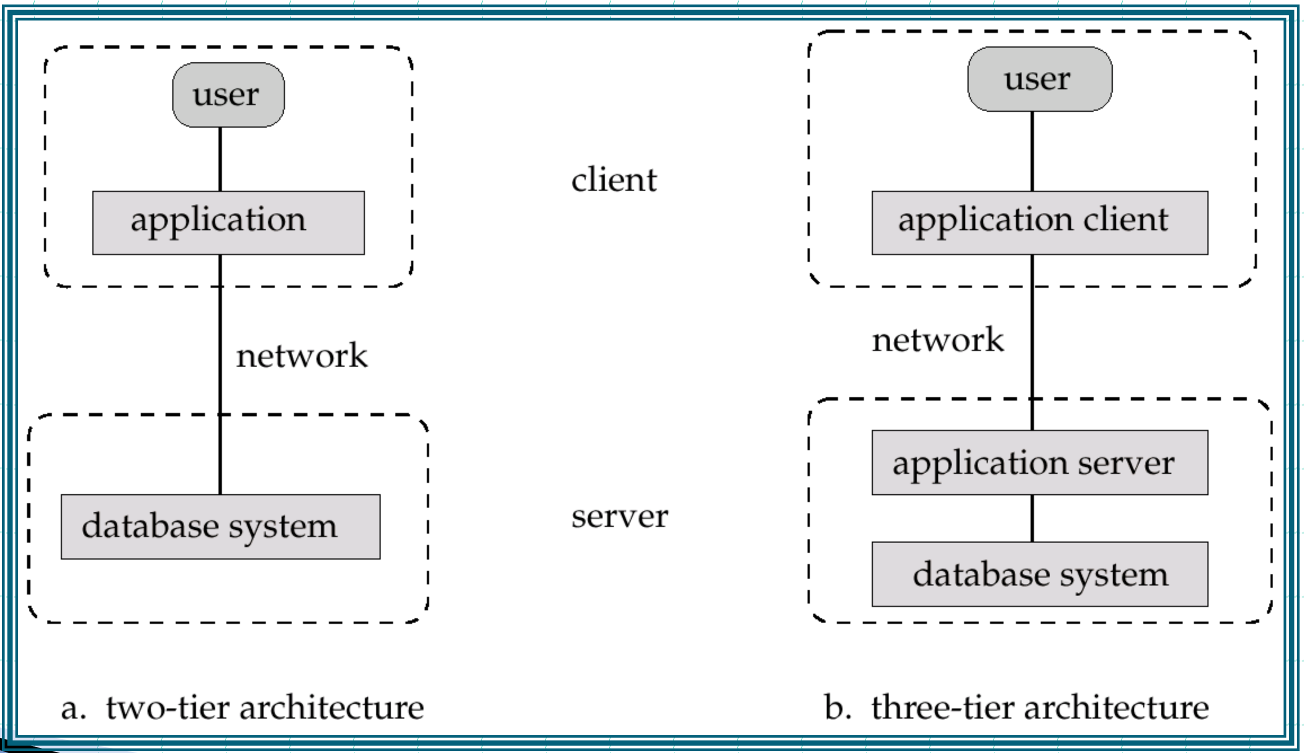
- New functionality is being added to DBMSs in the following areas:
  - Scientific Applications
  - Image Storage and Management
  - Audio and Video data management
  - Data Mining
  - Spatial data management
  - Time Series and Historical Data Management

Overall System Structure



Application Architecture

- **Two-tier architecture**: E.g. client programs using ODBC/JDBC to communicate with a database
- **Three-tier architecture**: E.g. web-based applications, and applications built using "middleware"



a. two-tier architecture

b. three-tier architecture