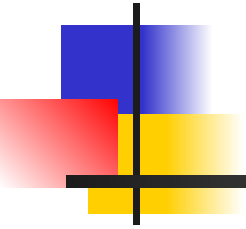


# Database Engineering



## Lecture #1

# Introduction to Database Systems

*Presented by:*

Dr. Suvasini Panigrahi

Associate Professor

Dept. of CSE, VSSUT, Burla



# Outline

---

- Traditional File Systems
- Drawbacks of Traditional File Systems
- Database Introduction
- Database Applications
- An Example
- Characteristics of the Database
- Database Users
- Advantages of using the DBMS approach
- Limitations of the DBMS approach



# Traditional File Systems

---

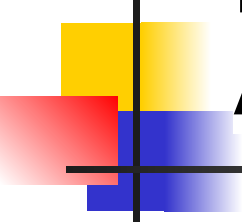
- Before the development of database management systems, organization stored information in various files
- The traditionally used “File Systems” were nothing but a manual way of storing data as “Files”. Each file is independent of each other and is called a flat file
- A flat file is a collection of data stored in a two-dimensional database in which information are stored as records in a table.
- Each user defines and maintains separate files needed for a specific application



# Traditional File Systems

---

- Various **application programs** are used with each file to extract records from, add records to and perform other manipulations on the files
- File systems have **no centralized control** of the data descriptions
- Table and field names may be used in different files to mean different things

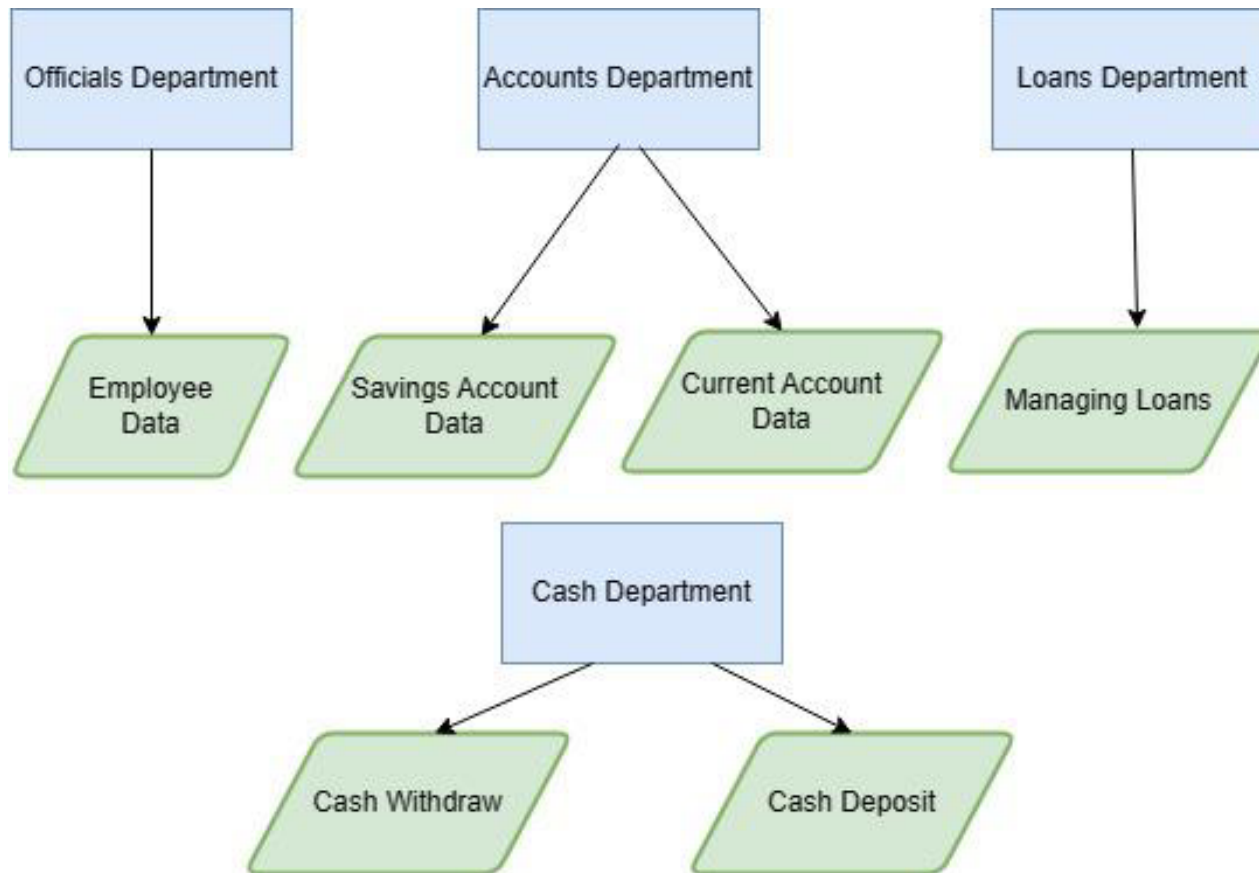


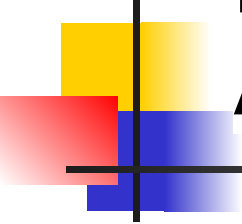
# Example of a Traditional File System: Banking system that uses File Based Approach

---

- Considering a scenario of a bank before the introduction of DBMS. For example, if someone went to the bank to deposit a certain amount in their account.
- As the DBMS is not available, so the bank employee has to manually register their account number, name, and amount in either a written manner or type and store them locally in the computer as a file.
- The problem which might arise is that while writing if the employee mistakenly writes any digit of the customer's account number or amount wrong then there would be a major issue and as there is no database, so, it would be really difficult to know what was the last state of that person's account.

# Example of a Traditional File System: Banking system that uses File Based Approach





# Example of a Traditional File System: Banking system that uses File Based Approach

---

- In the diagram we can see how the details used to be stored by bank employees before the introduction of DBMS
- Each department would handle some specific tasks and store the data locally in their computers or registers without knowing what is happening in the other department.



# Drawbacks of Traditional File Systems

---

- Data redundancy and inconsistency
  - Same information may be duplicated in different files. This redundancy leads to high storage and access cost.
  - Various copies of same data may disagree leading to **data inconsistency**.
- Difficulty in accessing data
  - Need to write a new program to carry out each new user request for retrieving the required information from the files.
- Data isolation
  - As data are scattered in various files and files may be in different format, writing new application programs to retrieve the appropriate data is difficult.





# Drawbacks of Traditional File Systems

---

- Integrity problems
  - Data stored in the files must satisfy certain integrity constraints. For eg: balance of certain types of accounts should never fall below a certain amount
  - Developers enforce these constraints by adding appropriate code in the various application programs
  - Difficult to change the programs each time new constraints are added or existing ones are changed



# Drawbacks of Traditional File Systems

---

- Atomicity of updates

- Failures in a computer system may leave database in an **inconsistent state** with partial updates carried out
- It is essential for maintaining the database consistency that either a transaction completes or not happen at all.
- For eg: In transfer of funds from one account to another, either both credit and debit occur or neither occur
- Difficult to ensure atomicity in a conventional file-processing system



# Drawbacks of Traditional File Systems

---

- **Concurrent Access Anomalies**
  - To improve the overall performance of a system and faster response, many systems allow multiple users to access and update the data simultaneously.
  - In such an environment, data inconsistency may occur if the concurrent updates are not carried out properly.
- **Security problems**
  - Not every user of a system should be able to access all the data. Hard to provide user access to some, but not all data.
- Database systems offer solutions to all the above problems of traditional file systems



# Introduction: Basic Definitions

---

- **Data:** Known facts that can be recorded and have an implicit meaning.
- **Database (DB):** A collection of related data which is designed, built and populated with data for a specific purpose.
- **Miniworld/Universe of Discourse (UoD):** Some part of the real world about which data is stored in a database.
- **Database Management System (DBMS):** A *general-purpose software* which facilitates the process of *defining, constructing* and *manipulating* databases for various applications.



# Introduction: Basic Definitions

---

- **Defining a database** involves specifying the data types, structures and constraints for the data to be stored in the database
- **Constructing a database** is the process of entering data in the database tables for storing them on some storage medium that is controlled by the DBMS
- **Manipulating a database** includes functions such as:
  - **Retrieval:** Querying the DB for retrieving certain data and generating reports
  - **Modification:** Insertions or deletions or updates to reflect changes in the miniworld



# Database Applications

---

- Databases are widely used in almost all aspects of our lives.
- Some of the database applications in real life are:
  - **Banking:** customer information, loans and banking transactions
  - **Credit Card/Debit Card Transactions:** purchases on credit/debit cards and generation of monthly statements, etc.
  - **Telecommunications:** Records of calls made, generating monthly bills, balances on prepaid calling cards, etc.
  - **Railways/Airlines:** reservations, schedule information, etc.
  - **Universities:** student information, course registration, grades, etc.
  - **Sales:** information relating to customers, products and purchases
  - **Manufacturing:** production, inventory, orders, supply chain, etc.



# Database Applications

---

- Many other applications ...
  - Social Media Platforms
  - eCommerce Websites
  - Email-Services
  - Airline Reservation Systems
  - Hotel Booking Systems
  - Healthcare Information Systems
  - Online Learning Platforms
  - Human Resources Management Systems (HRMS)



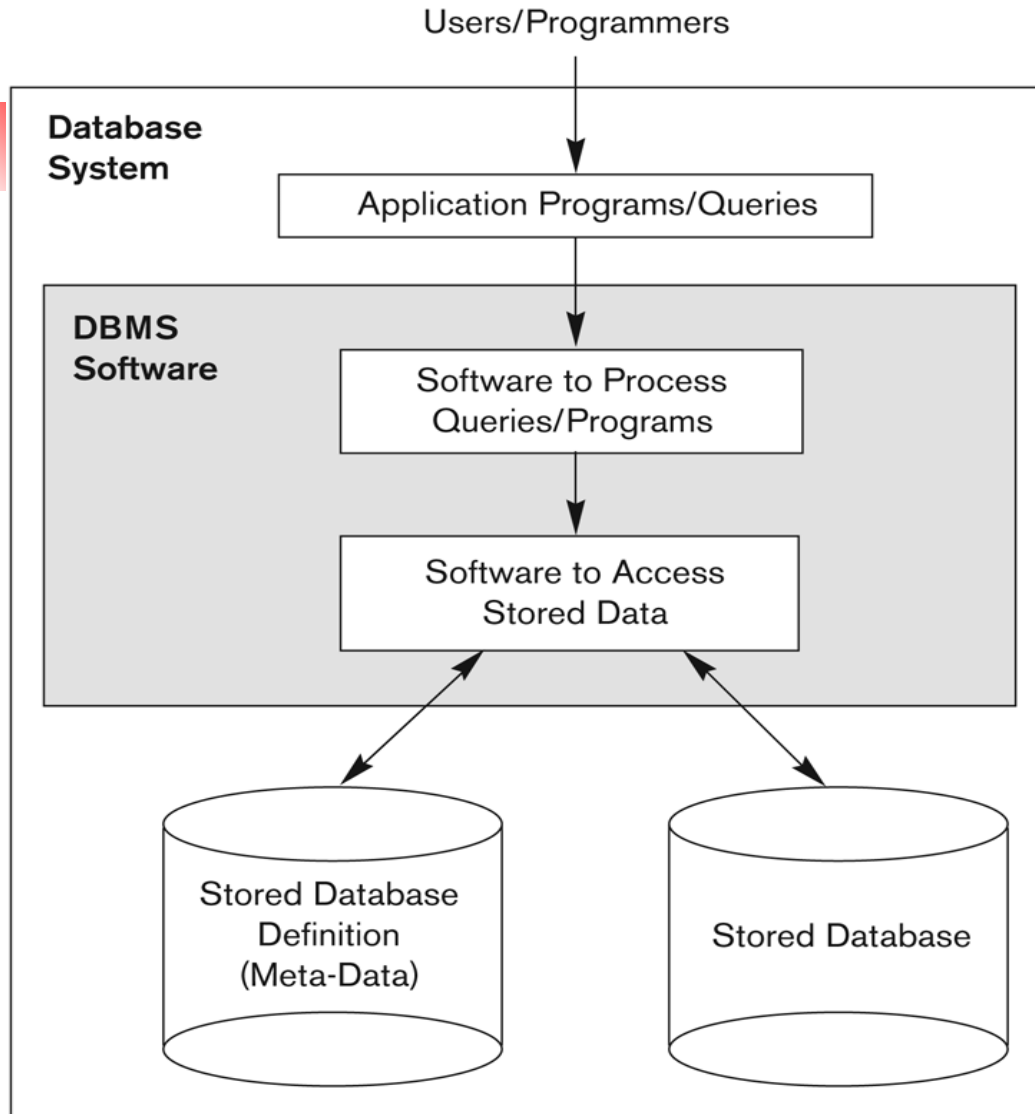
# Database System

---

- The **DBMS software** is a collection of programs that enables users to create and maintain a database
- **System Catalog/Data dictionary** is a collection of data about the database which are used by the DBMS for describing the **structure of a database** like *name of the tables, name of attributes of each table, data types and lengths of attributes*, etc.
- All this information called **metadata** (*data about data*) forms a miniature database
- The *database* and the *DBMS software* are together called a **Database System (DBS)**



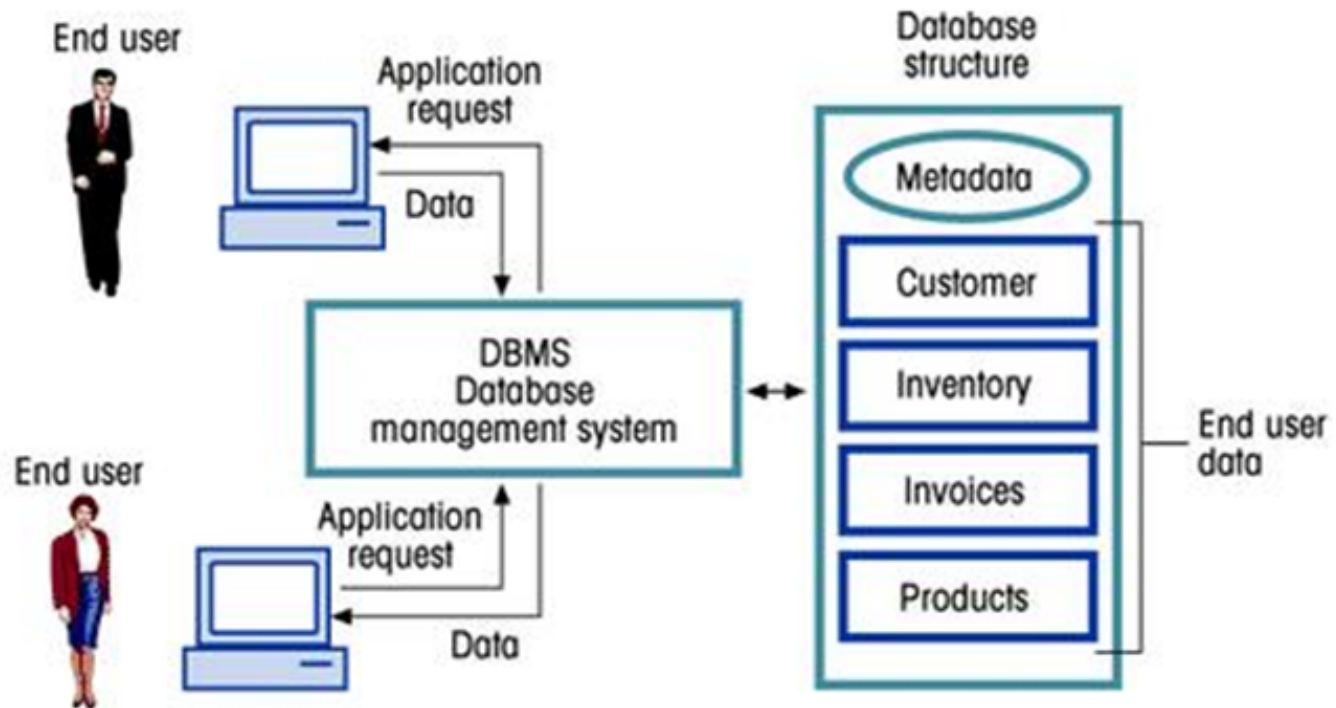
# Simplified Database System Environment



In the database approach, a **single repository of data is maintained** which is defined once and then accessed by various users

**Figure 1.1**  
A simplified database system environment.

# Role of Database Management System



**DBMS Manages the Interaction Between the End User and the Database**

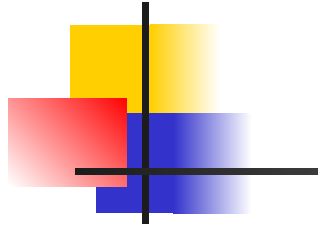


# Example of a Database System: An UNIVERSITY Example

---

- A UNIVERSITY database for maintaining information concerning students, courses and grades in a university environment
- Suppose, we have the following tables:
  - **STUDENT** file stores data on each student
  - **COURSE** file stores data on each course
  - **SECTION** file stores data on each section of each course
  - **GRADE\_REPORT** file stores the grades that students receive
  - **PREREQUISITE** file stores the prerequisites
- To construct the UNIVERSITY database, data is stored to represent each student, course, section, grade report and prerequisites as records in the appropriate table

# Example of a University Database



## COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

## SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

## STUDENT

Name	Roll No	Class	Major
Smith	17	1	CSE
Brown	8	2	CSE

## GRADE\_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

## PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

**Figure 1.2**

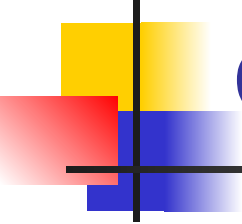
A database that stores student and course information.

# Characteristic of the Database Approach



---

- The main characteristic of the database approach versus the file processing approach are the following:
  - Self-describing nature of a DB
  - Insulation between programs and data and data abstraction
  - Support of multiple views of the data
  - Sharing of data and multiuser transaction processing



# Self-describing nature of a database system

---

- Database system contains not only the database itself but also a complete definition of the database structure and constraints
- The information stored in the **system catalog** is called **Meta-data** (data about data), and it describes the structure of the database

# Example of a simplified Meta-data

## RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

**Figure 1.3**

An example of a database catalog for the database in Figure 1.2.

## COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
....	....	....
....	....	....
....	....	....
Prerequisite_number	XXXXNNNN	PREREQUISITE

Note: Major\_type is defined as an enumerated type with all known majors. XXXXNNNN is used to define a type with four alpha characters followed by four digits

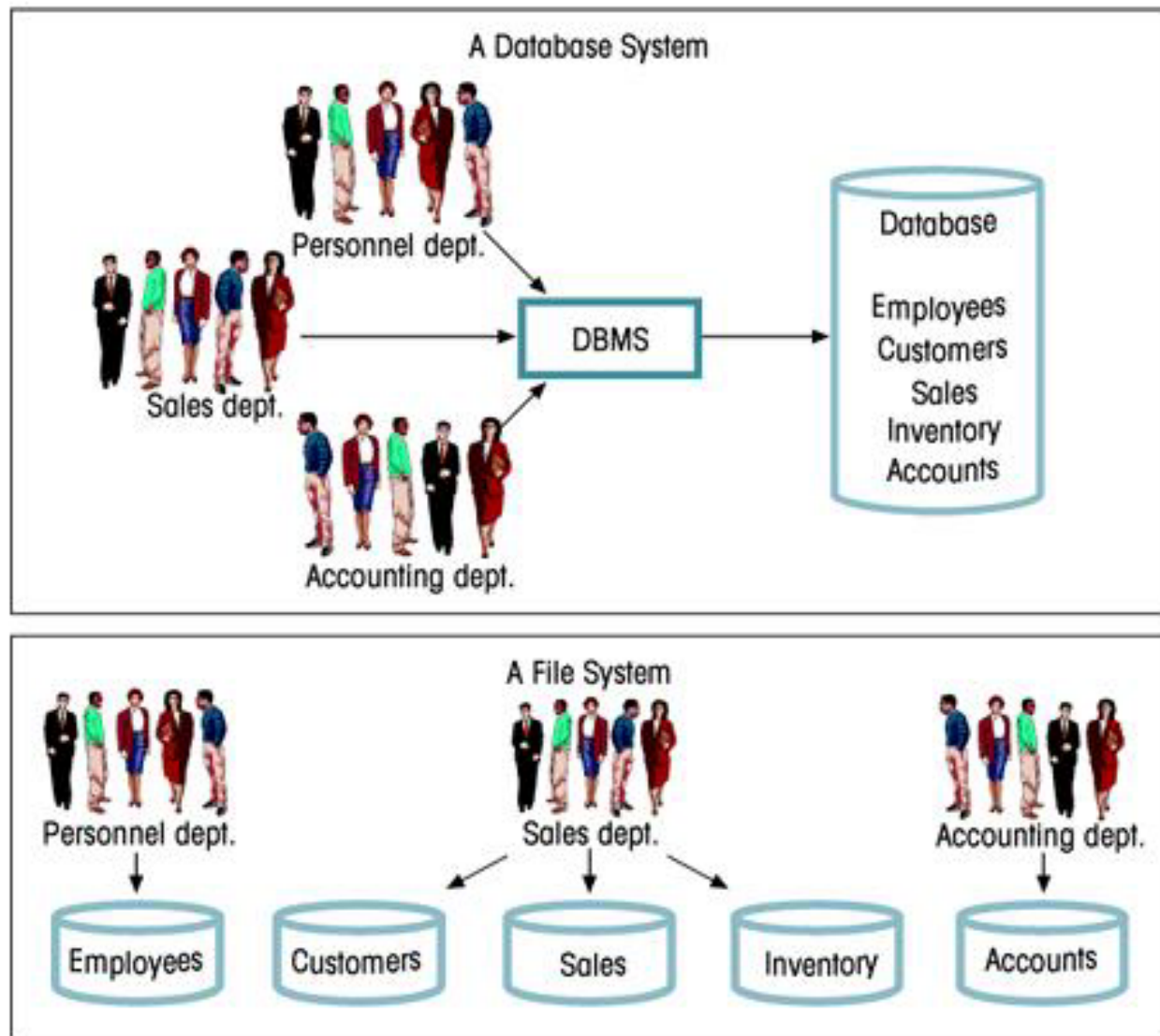


FIGURE 1.6  CONTRASTING DATABASE AND FILE SYSTEMS





# Insulation between programs and data and data abstraction

---

- In file processing, if any changes to the structure of a file may require changing all programs that access the file
- In database system, the structure of data files is stored in the DBMS catalog separately from the access programs
- DBS allows changing data storage structures and operations without having to change the DBMS access programs
- This is called **program-data independence**



# Insulation between programs and data and data abstraction

---

- Data Abstraction: DBMS provides a *conceptual view* of the database to the users that does not include the storage details of data and how the operations are implemented in the database
- A **data model** is a type of data abstraction that is used to provide this conceptual representation of the database



# Support of multiple views of the data

---

- A **view** is a virtual table based on the result-set of an SQL statement.
- A view contains rows and columns, just like a real table that acts as a proxy table created from the original table.
- Each user may see a different view of the database, which describes only the data of interest to that particular user
- Thus, a view may also contain some **virtual data** that is derived from the database tables but it is not explicitly stored in the database



# Sharing of data and multi-user transaction processing

---

- Allowing a set of **concurrent users** to retrieve from and to update the database.
- Concurrency control within the DBMS guarantees that each transaction is correctly executed or aborted
  - For example, when several reservation clerks try to assign a seat on an airplane flight
  - These types of applications are generally called **Online Transaction Processing (OLTP)**



# Database Users

---

- Many persons are involved in the design, use and maintenance of a large database with few hundred users
- Database users may be divided into the following groups:
  - Persons whose jobs involve the day-to-day use of a large database are called “**Actors on the Scene**” (**control the content of the database**)
  - Users who are associated with the development of the database and the design and implementation of the DBMS software are called “**Workers Behind the Scene**”



# Database Users: Actors on the Scene

---

- Database Administrators (DBAs):
  - In any organization when many persons use the same resource, there is a need for a chief administrator to manage these resources
  - In a database environment, the primary resource is the database itself and the secondary resource is the DBMS software.
  - Administering these resources is the responsibility of the DBA.
  - DBA is responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.



# Database Users: Actors on the Scene

---

- Database Designers:

- Responsible for identifying the data to be stored in the DB and for choosing the appropriate structure to represent and store the data.
- They must communicate with different database users, understand their requirements and come up with the design that meets the requirements of all users.
- These tasks are mostly undertaken before the database is actually implemented and populated with data.

- End Users:

- These are the people whose jobs require access to the database for querying, updating and generating reports
- Database primarily exists for these users



# Database Users: Actors on the Scene

---

## Categories of End-users

- **Casual:** Access database occasionally but may need different information each time. E.g. manager
- **Naive/Parametric:** These users make up a large section of the end-user population and are constantly involved in querying and updating the database. E.g. Reservation clerks
- **Sophisticated:** These include business analysts, scientists, engineers and others who are thoroughly familiar with the facilities of the DBMS so as to implement their applications to meet their requirements
- **Stand-alone:** Mostly maintain personal databases using ready-made packages that provide graphical interfaces.  
Eg. User of a tax package that stores a variety of personal financial data for tax calculation purpose.





# Database Users: Actors on the Scene

---

- System Analysts and Application Programmers (Software Engineers)
  - *System analysts* determine the requirements of end-users (naive) and develop specifications for transactions that meet these requirements
  - *Application programmers* implement these specifications as programs, they then test, debug, document and maintain these transactions



# Other DBS Users: **Workers** **Behind the Scene**

---

- **DBMS System Designers and Implementers**
  - Persons who design and implement the DBMS modules and interfaces as a s/w package
- **Tool Developers**
  - Persons who design and implement tools (optional packages for performance monitoring, simulation, test data generation, etc.)
- **Operators and maintenance personnel**
  - Responsible for actual running and maintenance of the h/w and s/w environment for the DBS



# Controlling Redundancy and Maintaining Data Consistency

---

- **Redundancy** in storing the same data multiple times leads to several problems:
  1. We need to update data several times (duplication of effort)
  2. Storage space is wasted
  3. The same data may become *inconsistent* as an update may be applied to some files and not to others



# Controlling Redundancy and Maintaining Data Consistency

---

- In DBMS, the data redundancy can be controlled or reduced but is not removed completely
- Sometimes, it is necessary to create duplicate copies of the same data items in order to relate tables with each other (*controlled redundancy*)
- By controlling the data redundancy, we can save storage space and avoid data inconsistency



# Advantages of using the DBMS approach: Restricting Unauthorized Access (Data Security)

---

- **Data security** is the protection of the database from unauthorized users
- Only the **authorized users** are allowed to access the database
- The database access is controlled by the **DBA**. Mostly, the DBA can access all the data in the database. Some users may be permitted only to retrieve data, whereas others are allowed to retrieve as well as to update data.
- Most of the DBMSs provide the **security and authorization subsystem**, which the DBA uses to create accounts of users protected by passwords and to specify account restrictions



# Advantages of using the DBMS approach: Enforcing Integrity Constraints

---

- ***Integrity constraints*** or *consistency rules* are applied to a database so that the correct data is entered into the database
- **Examples of integrity constraints are:**
  - Maximum obtained marks in a subject cannot exceed 100
  - 'Issue Date' in a library system cannot be later than the corresponding 'Return Date' of a book
  - There are also some standard constraints that are intrinsic in most of the DBMSs. Eg.: NOT NULL, PRIMARY KEY, etc.
- DBA identifies the integrity constraints during database design
- The integrity constraints are automatically checked at the time of data entry or when a record is updated.



# Other Advantages of using the DBMS approach:

---

- **Data Sharing:**

- DBMS allows the sharing of data under its control by any number of application programs or users
- Many users can be authorized by the DBA to access the same piece of information simultaneously
- **Concurrency control schemes** are used in a database to coordinate simultaneous transactions while preserving data integrity.
- It addresses conflicts with the simultaneous accessing or altering of data that can occur in a multi-user system



# Other Advantages of using the DBMS approach

---

- **Integration of Data and Representing complex relationships among data**
  - In database approach, a single repository of data is maintained which is defined once and then accessed by various users
  - A database contains data stored in multiple tables and a DBMS has the capability to represent a variety of complex relationships among the data which makes the access and updating of data easier

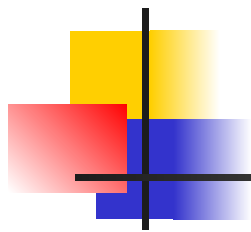




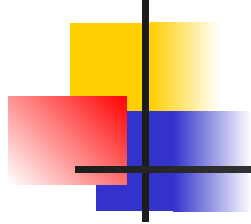
# Other Advantages of using the DBMS approach:

---

- Providing Backup and Recovery (Data Atomicity)
  - A DBMS provides facilities for recovering from hardware or software failures
  - The **backup and recovery subsystem** of the DBMS is responsible for recovery
  - For eg. if the computer system fails in the middle of a complex update program, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the program started executing



**ANY  
DOUBTS?**



---

**THANK  
YOU!**