Introduction Database Concepts

Dr. Jyoti Wadmare

Contents

- Traditional database applications, in which most of the information that is stored and accessed is either textual or numeric
- bank to deposit or withdraw funds
- hotel or airline reservation,
- access a computerized library catalog to search for a bibliographic item
- purchase something online—such as a book, toy, or computer.
- Even purchasing items at a supermarket often automatically updates the database that holds the inventory of grocery items

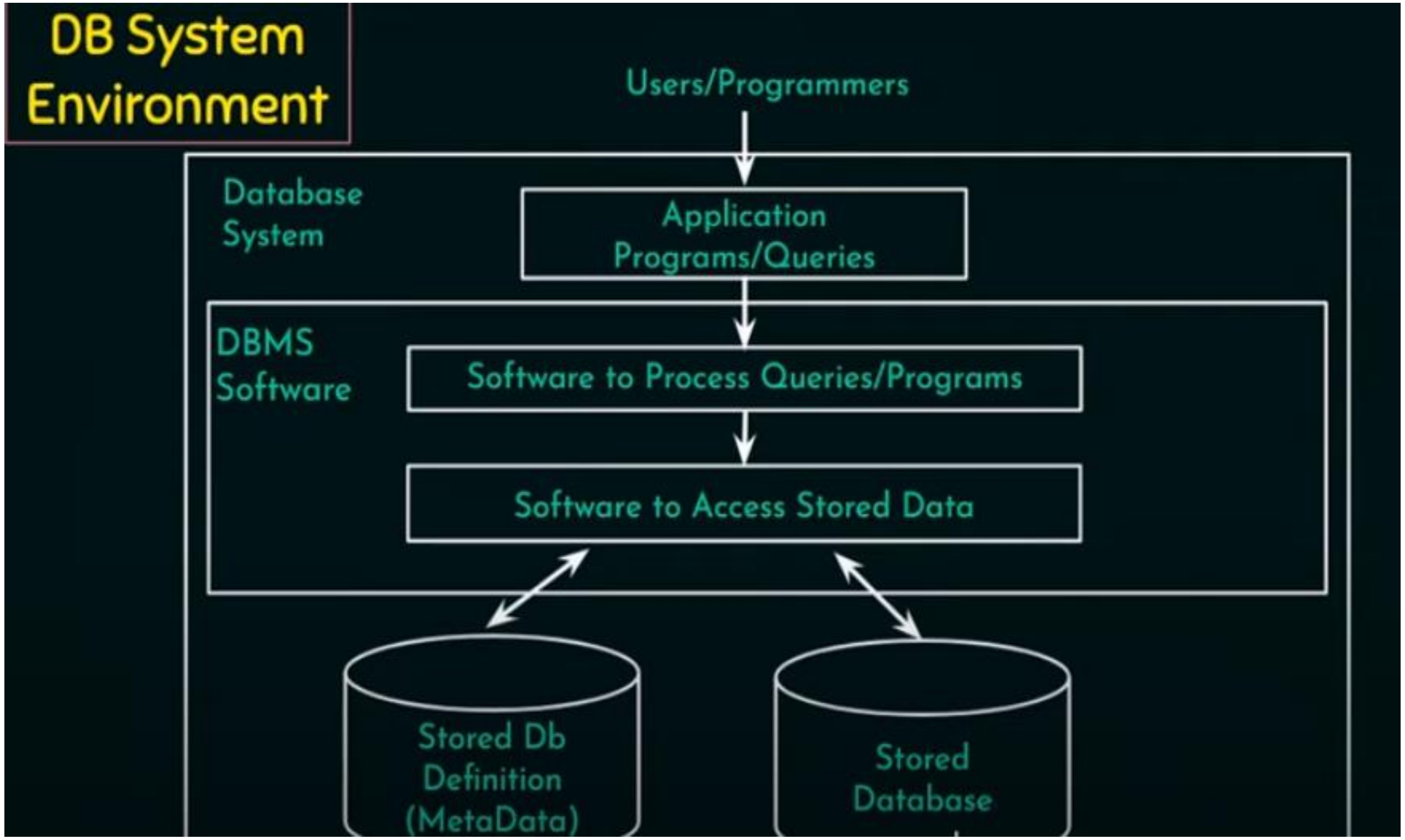
- Multimedia databases -store images, audio clips, and video streams digitally. These types of files are becoming an important component.

- Geographic information systems (GIS) can store and analyze maps, weather data, and satellite images.

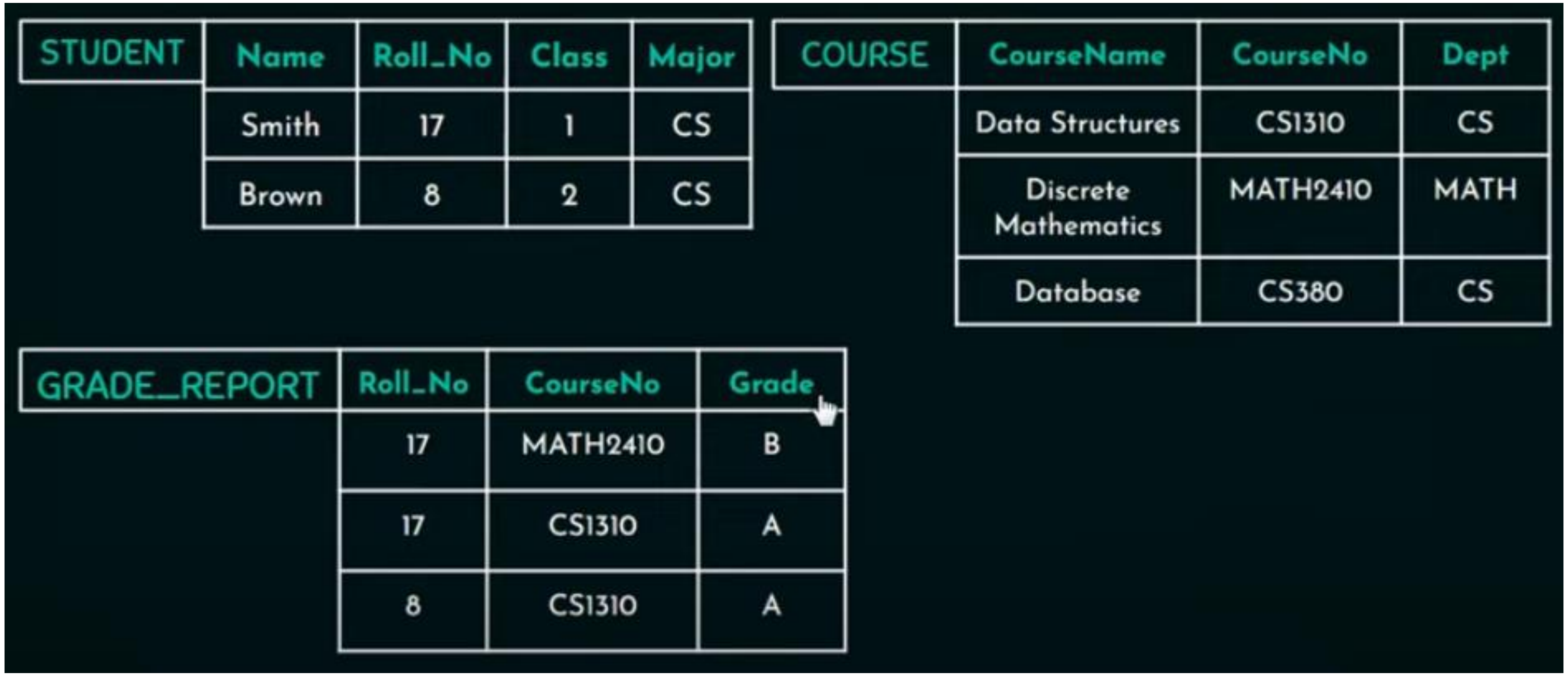
- Data warehouses and online analytical processing (OLAP) systems are used in many companies to extract and analyze useful business information from very large databases to support decision making.

DBMS

- Definition: It is collection of **interrelated data** and a **set of programs** to access those data.
- **The collection of data- database**, contains information relevant to the enterprise.
- The primary goal of the DBMS is to provide a way to store and retrieve database information that is **convenient** and **efficient**.
- Database system must ensure the **safety** of the information stored, despite system crashes or unauthorized access
- If data are to be shared among several users, the system must avoid possible anomalous results.


DB System Environment
Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Db Definition (MetaData)
Stored Database

- A database that stores Student and course information



STUDENT

Name	Roll_No	Class	Major
Smith	17	1	CS
Brown	8	2	CS

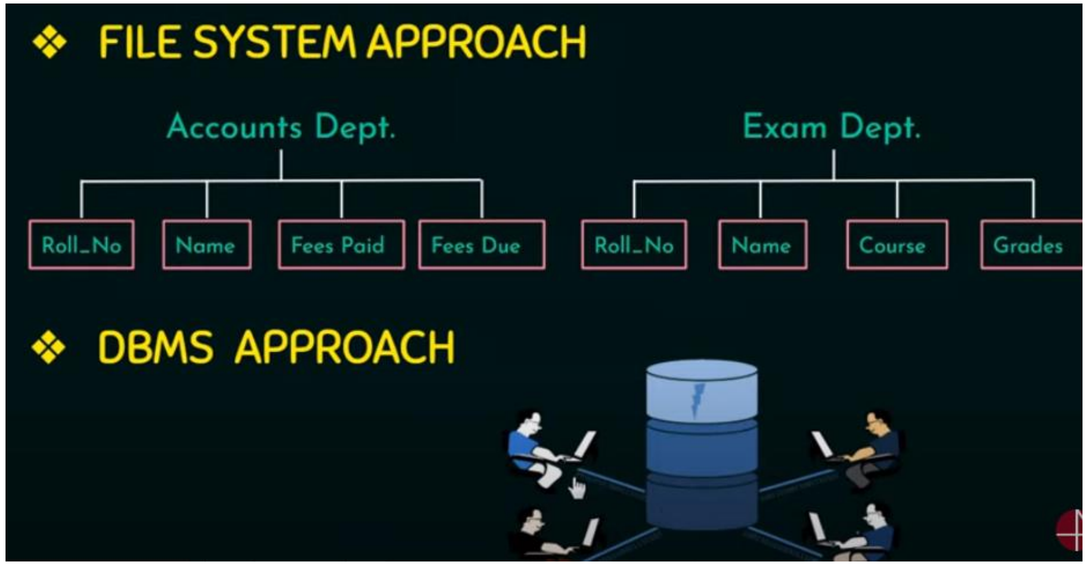
COURSE

CourseName	CourseNo	Dept
Data Structures	CS1310	CS
Discrete Mathematics	MATH2410	MATH
Database	CS380	CS

GRADE_REPORT

Roll_No	CourseNo	Grade
17	MATH2410	B
17	CS1310	A
8	CS1310	A

Characteristics of Database Approach



FILE SYSTEM APPROACH:

Eg. Exam Department, may keep files on students and their grades.

A second user, Account Department, may keep track of students' fees and their payments.

Although both users are interested in data about students, each user maintains separate files

each requires some data not available from the other user's files.

This redundancy in defining and storing data results in wasted storage space

Characteristics of Database Approach(cont…)

In the database approach, a single repository maintains data that is defined once and then accessed by various users.

In file systems, each application is free to name data elements independently.

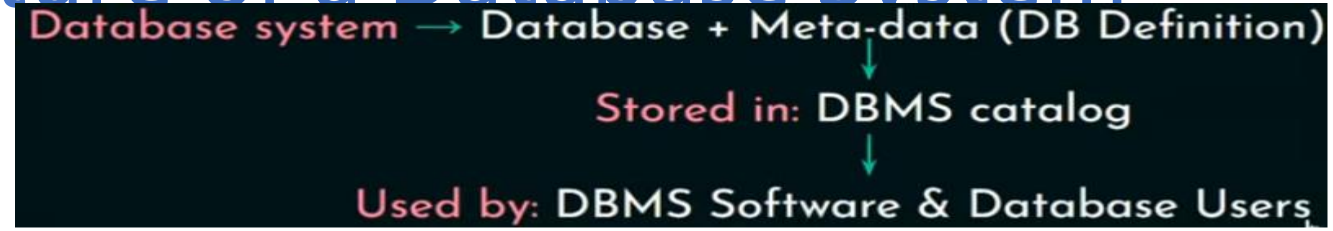
In contrast, in a database, the names or labels of data are defined once, and used repeatedly by queries, transactions, and applications.

Characteristics of Database Approach(cont…)

The main characteristics of the database approach versus the file-processing approach are the following:

- Self-describing nature of a database system
- Insulation between programs and data, and data abstraction
- Support of multiple views of the data
- Sharing of data and multi user transaction processing

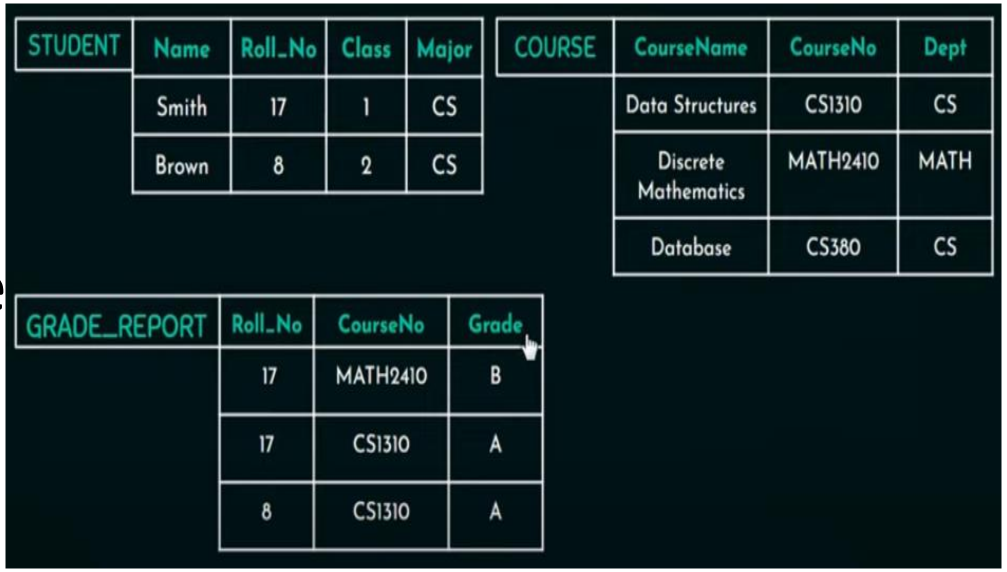
1.Self-Describing Nature of a Database System


Database system → Database + Meta-data (DB Definition)
↓
Stored in: DBMS catalog
↓
Used by: DBMS Software & Database Users

A DBMS catalog stores the description of a particular database (e.g. data structures, types, and constraints)

- The description is called meta-data.
- This allows the DBMS software to work with different database applications.

Figure 1.1, the DBMS catalog will store the definitions of all the files shown.


STUDENT

Name	Roll_No	Class	Major
Smith	17	1	CS
Brown	8	2	CS

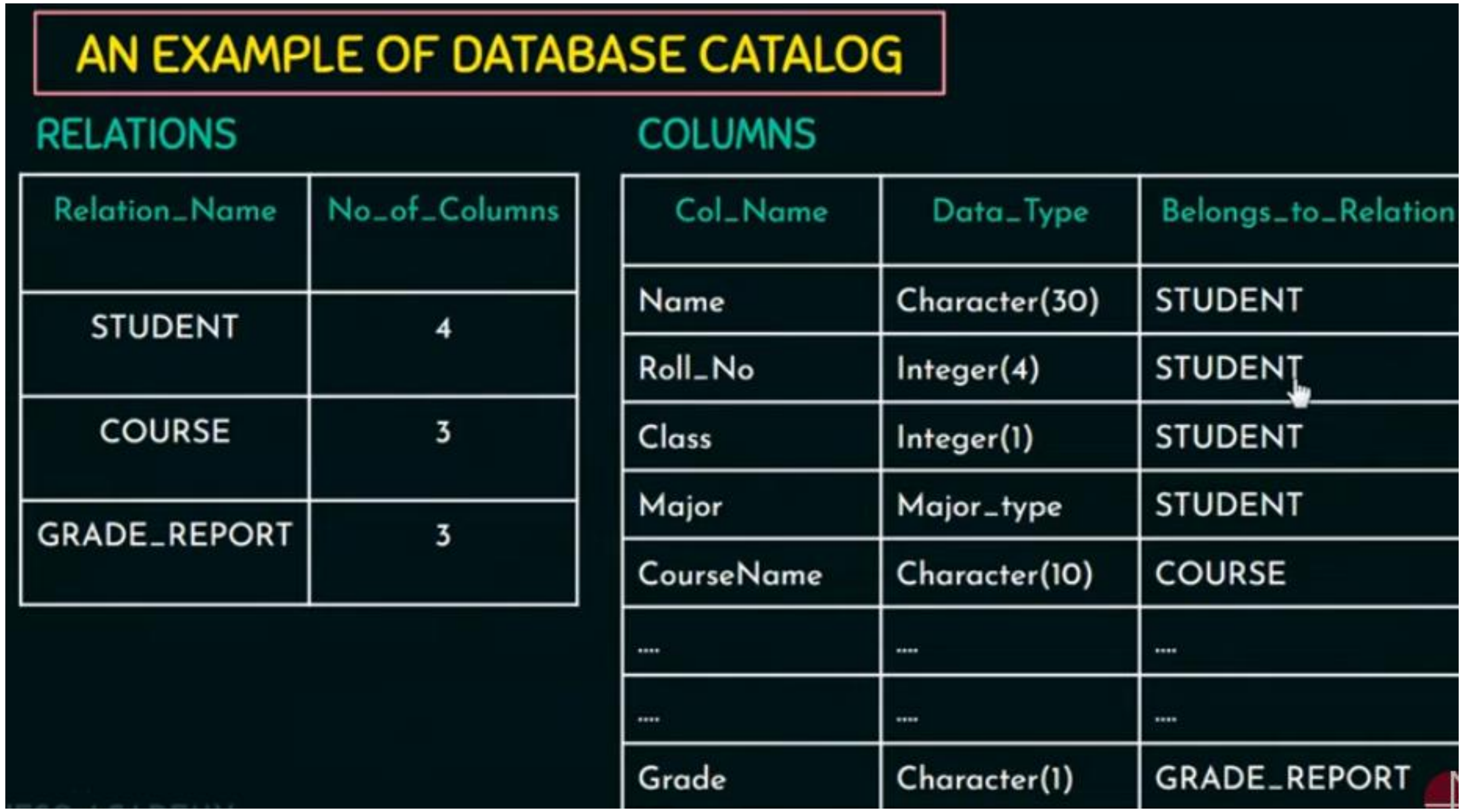
COURSE

CourseName	CourseNo	Dept
Data Structures	CS1310	CS
Discrete Mathematics	MATH2410	MATH
Database	CS380	CS

GRADE_REPORT

Roll_No	CourseNo	Grade
17	MATH2410	B
17	CS1310	A
8	CS1310	A

Figure 1.2 shows some sample entries in a database catalog



AN EXAMPLE OF DATABASE CATALOG

RELATIONS

Relation_Name	No_of_Columns
STUDENT	4
COURSE	3
GRADE_REPORT	3

COLUMNS

Col_Name	Data_Type	Belongs_to_Relation
Name	Character(30)	STUDENT
Roll_No	Integer(4)	STUDENT
Class	Integer(1)	STUDENT
Major	Major_type	STUDENT
CourseName	Character(10)	COURSE
....	....	....
....	....	....
Grade	Character(1)	GRADE_REPORT

These definitions are specified by the database designer prior to creating the actual database and are stored in the catalog

In traditional file processing, data definition is typically part of the application programs themselves. Hence, these programs are constrained to work with only one specific database

2.Insulation between Programs and Data, and Data Abstraction

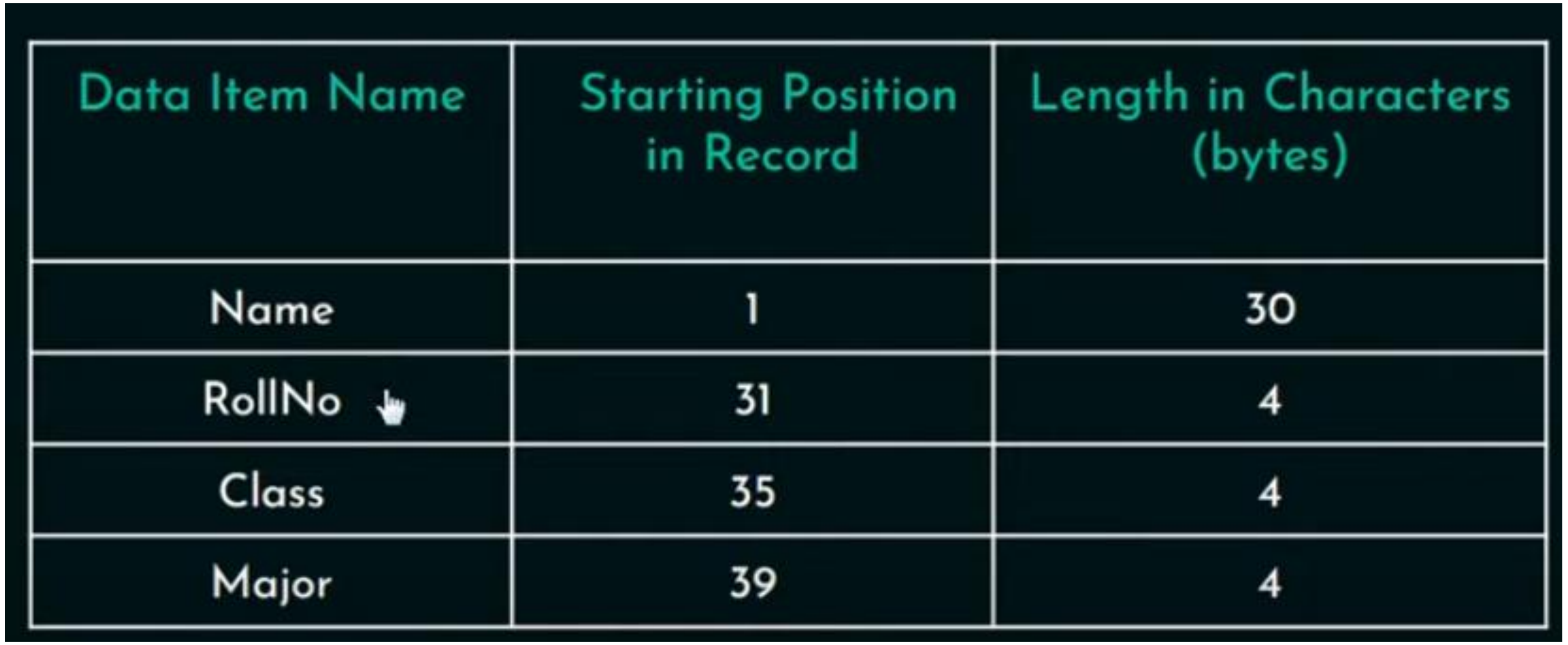
1)In traditional file processing, the structure of data files is embedded in the application programs,

so any changes to the structure of a file may require changing all programs that access that file.

The structure of data files is stored in the DBMS catalog separately from the access programs. We call this property program-data independence

The characteristic that allows program-data independence and program-operation independence is called data abstraction.

Internal storage format for a STUDENT record



Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
RollNo	31	4
Class	35	4
Major	39	4

In DBMS: Here we can add new data item as Date of birth without affecting the application program

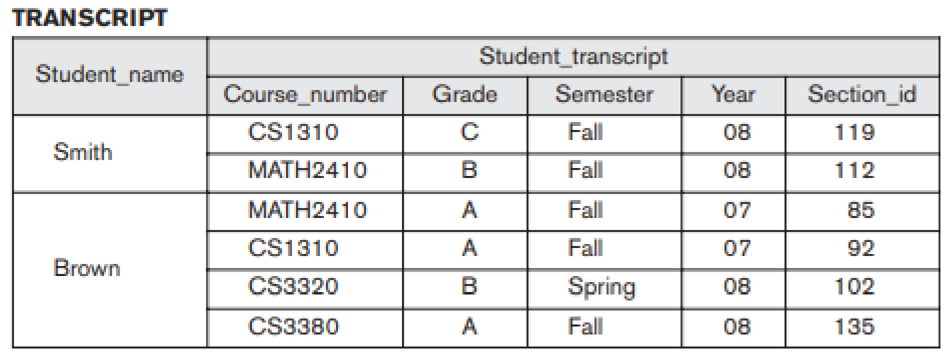
3.Support of Multiple Views of the Data

Each user may see a different view of the database, which

describes only the data of interest to that user.

Figure 1.2 may be interested only in accessing and printing the transcript of each student; the view for this user is shown in Transcript.

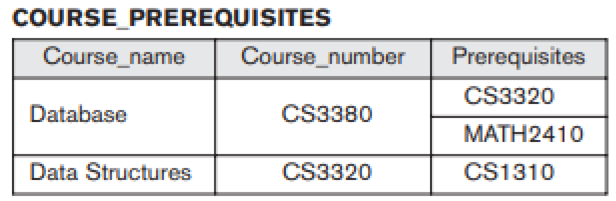
A second user, who is interested only in checking that students have taken all the prerequisites of each course for which they register, may require the view shown in Figure Course_Prerequisites.



TRANSCRIPT

Student_name	Student_transcript				
	Course_number	Grade	Semester	Year	Section_id
Smith	CS1310	C	Fall	08	119
	MATH2410	B	Fall	08	112
Brown	MATH2410	A	Fall	07	85
	CS1310	A	Fall	07	92
	CS3320	B	Spring	08	102
	CS3380	A	Fall	08	135

(a)



COURSE_PREREQUISITES

Course_name	Course_number	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data Structures	CS3320	CS1310

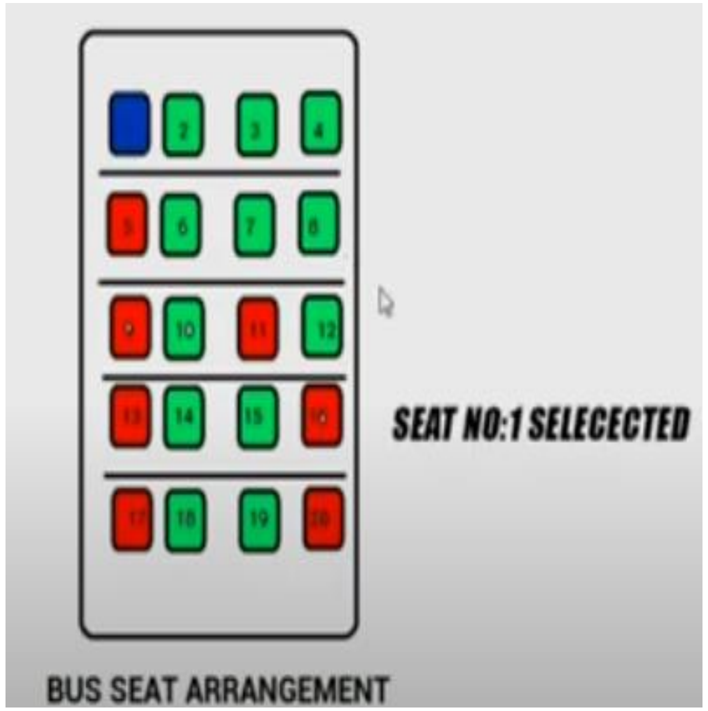
(b)

4. Sharing of Data and Multiuser Transaction Processing

The DBMS must include concurrency control to ensure that several users trying to update the same data do so in a controlled manner .

These types of applications are generally called online transaction processing (OLTP) applications

Eg. Online Airline/Railway ticket reservation system



DBMS must enforce several transaction properties

(program under execution)

i)Isolation   ii)Atomicity

Database-System Applications

Enterprise Information

- Sales: For customer, product, and purchase information.
- Accounting: For payments, receipts, account balances, assets, and

other accounting information.

- Human resources: For information about employees, salaries,

payroll taxes, and benefits, and for generation of paychecks

- Manufacturing For management of the supply chain and for tracking production of items in factories, inventories of items in warehouses and stores,

and orders for items.

Database-System Applications

- **Banking and Finance**
- **Banking**: For customer information, accounts, loans, and banking

transactions.

- **Credit card transactions**: For purchases on credit cards and generation of monthly statements.
- **Finance**: For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable online trading by customers and automated trading by the firm.
- **Universities**
- For student information, course registrations, and grades
- **Airlines**
- For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.

Database-System Applications

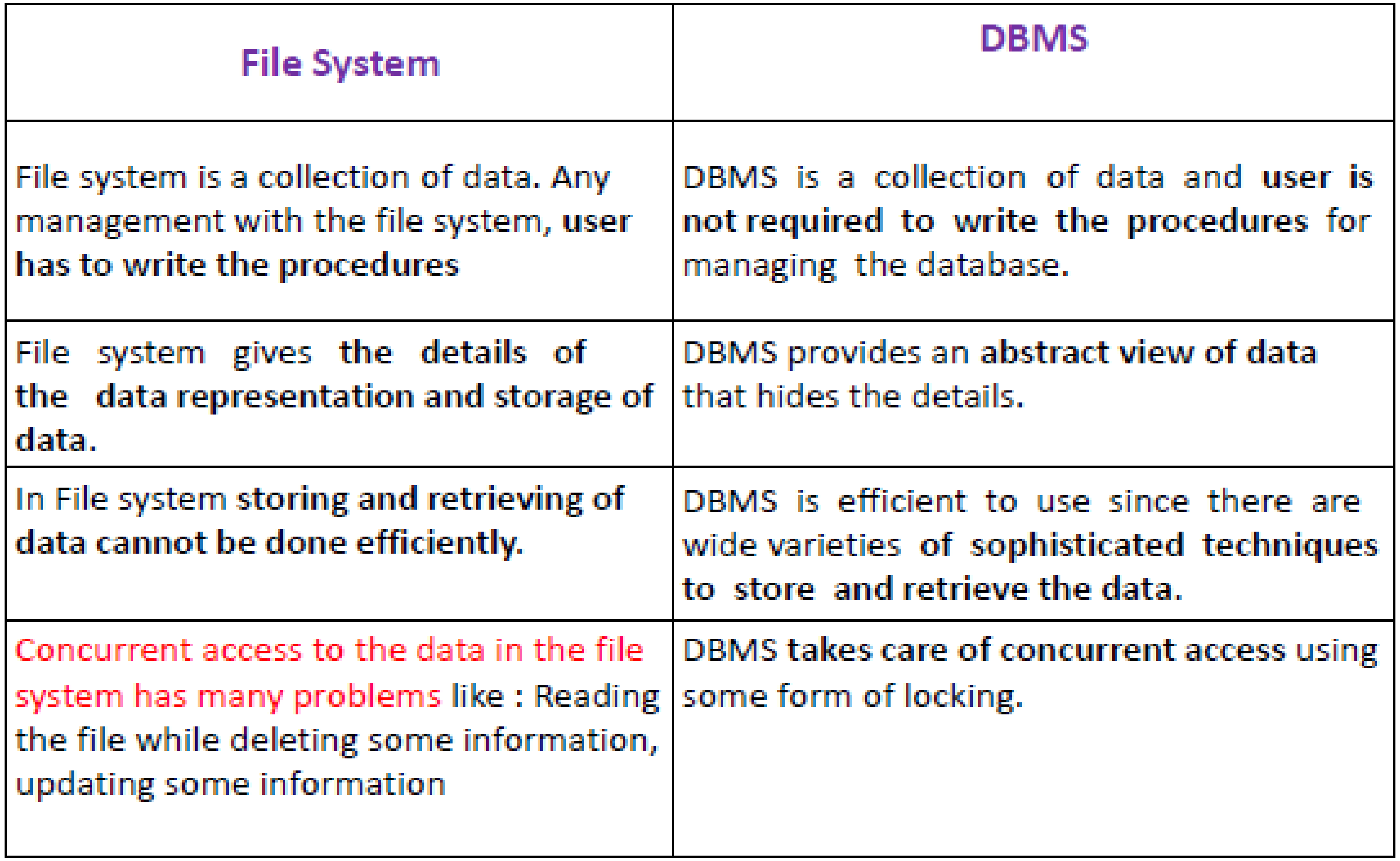
.**Telecommunication**

- For keeping records of calls, texts, and data usage, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
- **Web-based services**
- **Social-media**: For keeping records of users, connections between users(such as friend/follows information), posts made by users, rating/like information about posts, etc.
- **Online retailers:** For keeping records of sales data and orders as for any retailer, but also for tracking a user's product views, search terms, etc., for the purpose of identifying the best items to recommend to that user.
- **Document databases**
- For maintaining collections of new articles, patents, published research papers, etc.
- Navigation systems
- For maintaining the locations of varies places of interest along
with the exact routes of roads, train systems, buses, etc.

Advantages of DBMS over File System

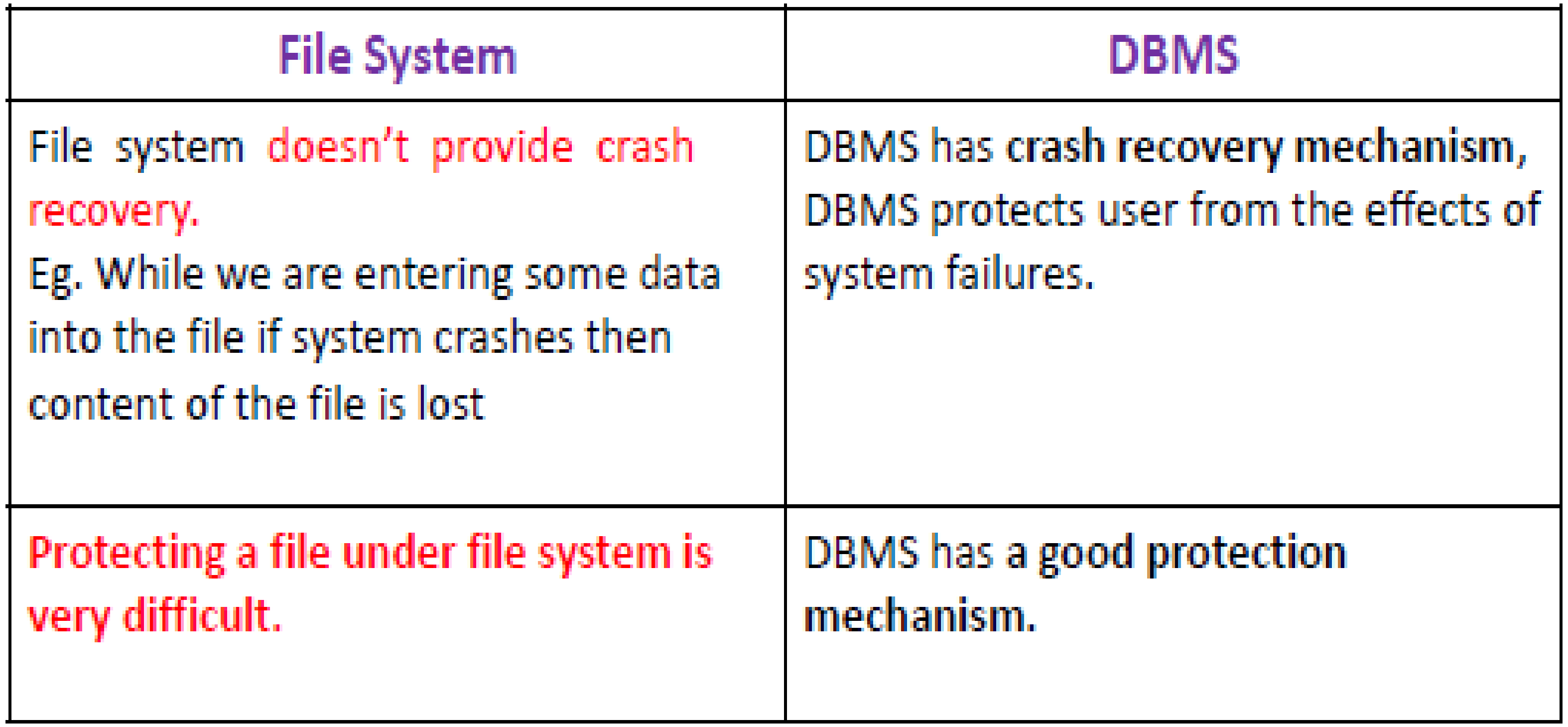
- Controlling redundancy in data storage and in development and maintenance efforts
- Sharing of data among multiple users
- Restricting unauthorized access to data
- Providing persistent storage for program objects
- Providing Storage Structures (e.g. indexes) for efficient Query Processing
- Providing backup and recovery services
- Providing multiple interfaces to different classes of users
- Representing complex relationships among data
- Enforcing integrity constraints on the database

Comparison of File System with Database Approach



File System	DBMS
File system is a collection of data. Any management with the file system, **user has to write the procedures**	DBMS is a collection of data and **user is not required to write the procedures** for managing the database.
File system gives the details of the **data representation and storage of data.**	DBMS provides an **abstract view of data** that hides the details.
In File system **storing and retrieving of data cannot be done efficiently.**	DBMS is **efficient to use since there are wide varieties of sophisticated techniques to store and retrieve the data.**
Concurrent access to the data in the file system has many problems like : Reading the file while deleting some information, updating some information	DBMS **takes care of concurrent access** using some form of locking.

Comparison of File System with Database Approach



File System	DBMS
File system doesn't provide crash recovery. Eg. While we are entering some data into the file if system crashes then content of the file is lost	DBMS has crash recovery mechanism, DBMS protects user from the effects of system failures.
Protecting a file under file system is very difficult.	DBMS has a good protection mechanism.

Data Abstraction

A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data.

A major purpose of a database system is to provide users with an abstract view of the data.

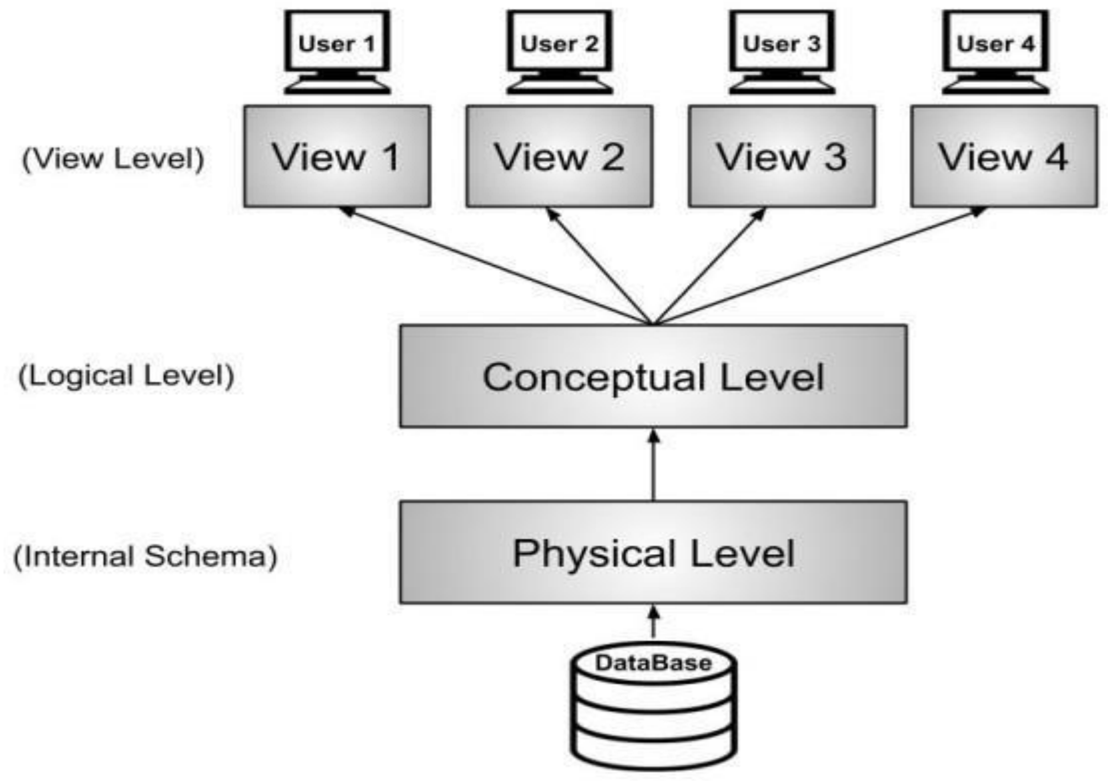
That is, the system hides certain details of how the data are stored and maintained.

Since many database-system users are not computer trained, developers hide the complexity from users through several levels of abstraction

Levels of Abstraction

These three levels of data abstraction / DBMS Three Level Architecture Diagram:

1. View Level/ External level
2. Conceptual Level/ Logical level
3. Physical Level/ internal level

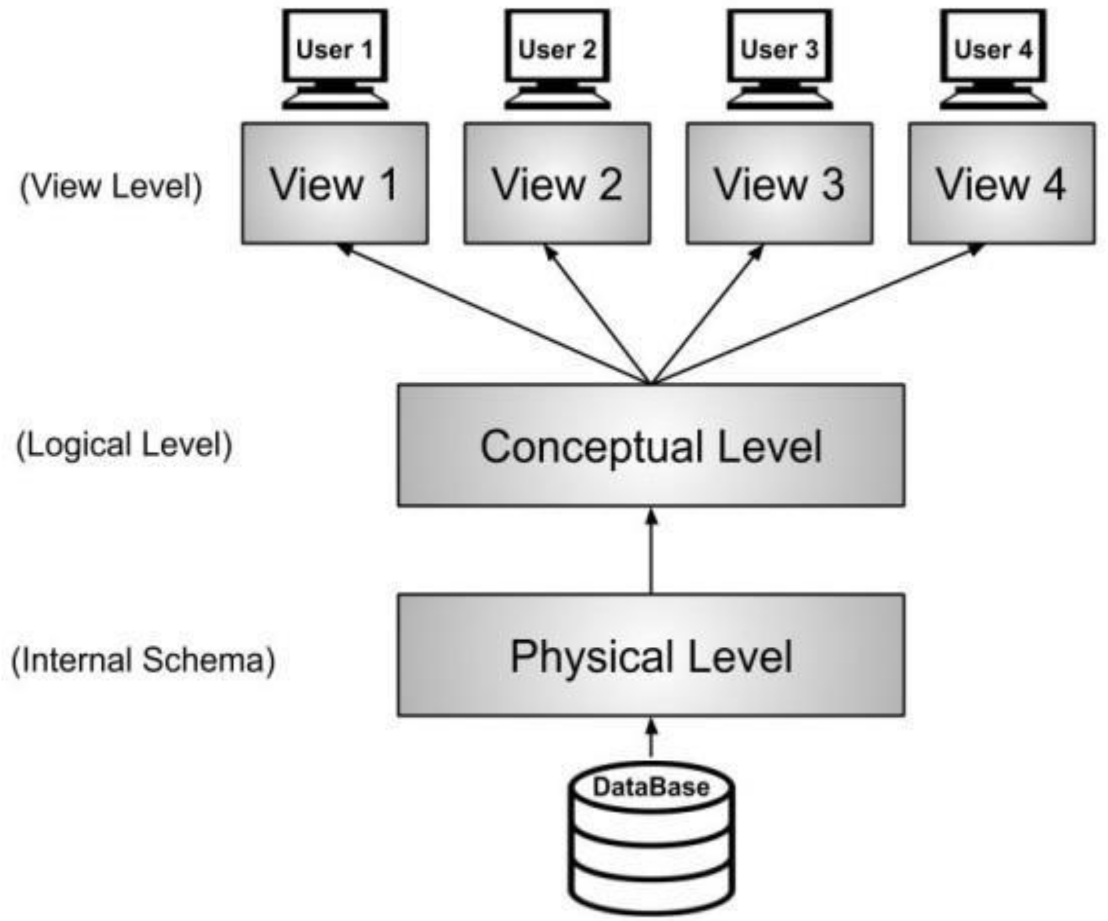


Levels of Data Abstraction

1. Physical Level or Internal Level

This is the lowest level of data abstraction.

- It tells us how the data is actually stored in physical memory (hard disks, magnetic tapes etc).
- The access methods like sequential or random access and file organization methods like B+ trees, hashing used for the organization purpose
- Developers would know usability, size of memory, and accessing frequency which makes easy to design this level.
- Blocks of storage and the amount of memory used for these purposes is kept hidden from the user.



Levels of Data Abstraction

2. Conceptual or Logical Level

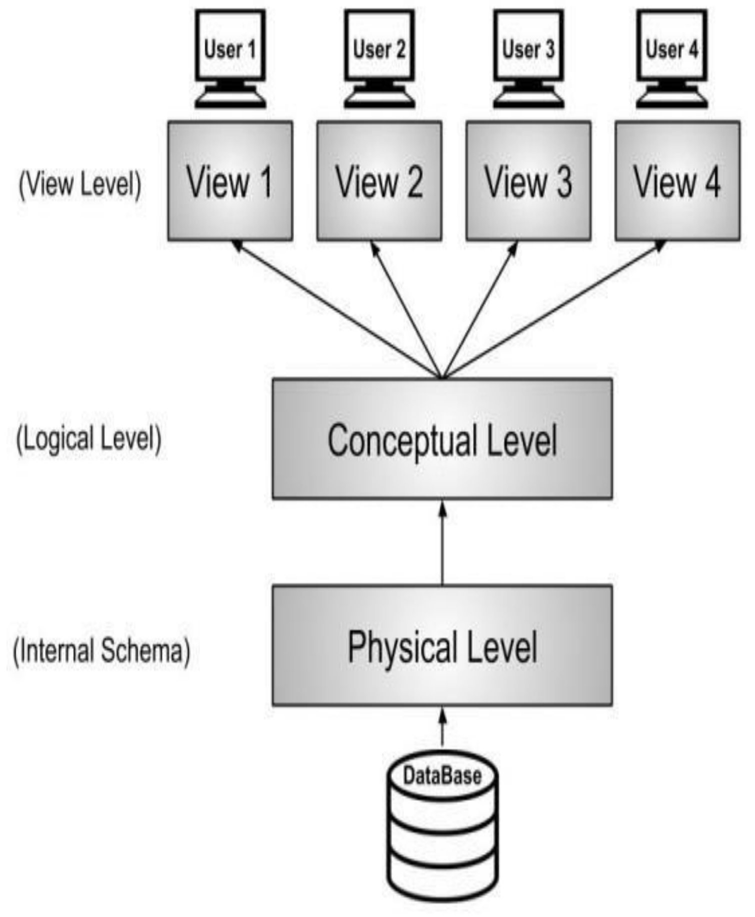
The conceptual level describes the structure of the whole database.

• This level acts as a middle layer between the physical storage and user view.

• It explains what data to be stored in the database, what relationship exists among those data, and what the data types are.

• There is only one conceptual schema per database.

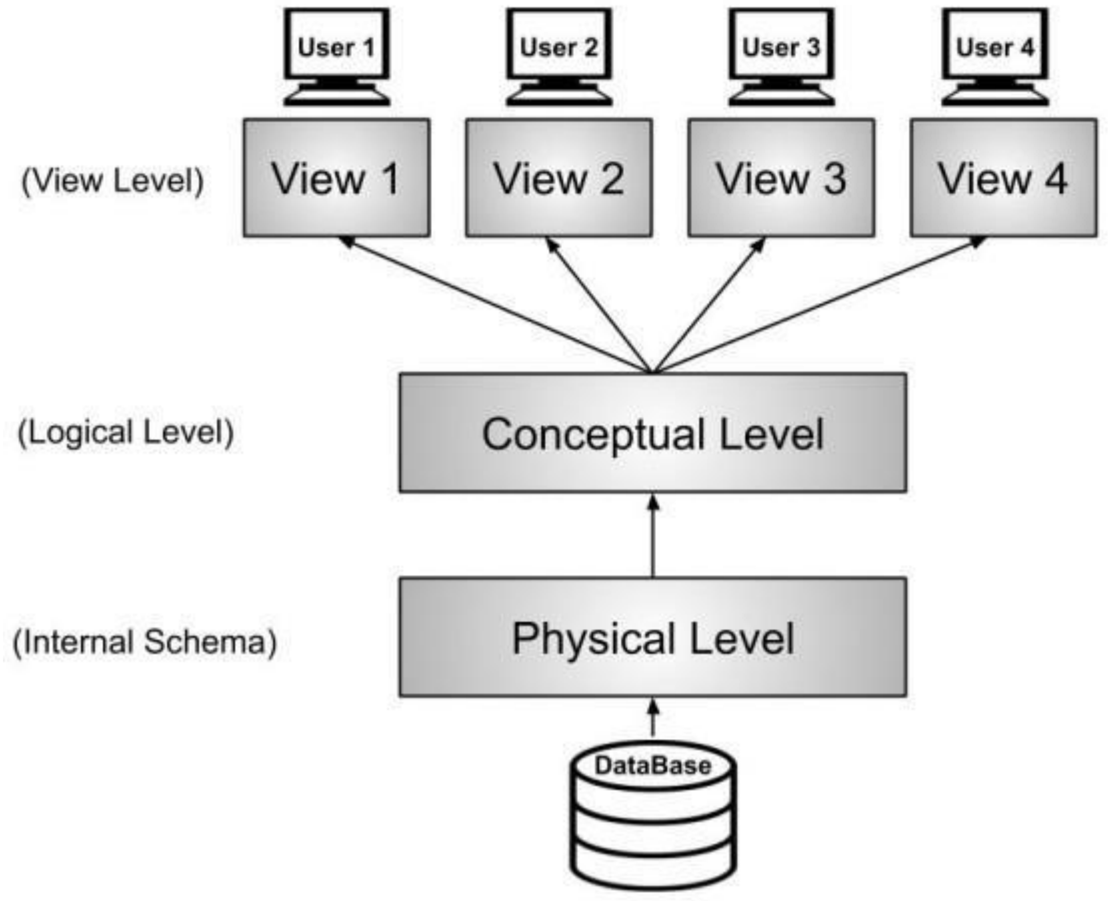
• Database administrator and the programmers use the logical level of abstraction to decide what information to keep in a database



Levels of Data Abstraction

3. External or View Level

• This is the highest level of database abstraction
• This level describes user interaction with database system
• Many users are not aware of technical details of the system, and also they need not access whole information from database
Hence it is necessary to provide a simple and short interface for such users as per their requirements
• Multiple views can be created for same database for multiple users



Levels of Data Abstraction

Levels of Abstraction

Example: Storing all employees information in employee table

- Physical level: Records can be described as blocks of storage(bytes, gigabytes, terabytes etc) in memory. All these details are usually hidden from developer
- Logical Level: Records can be described as fields/ attributes along with their datatypes
- Relationship between these fields can be implemented logically
- Usually developer works at this level because they have all such knowledge required
- View Level: Each user interacts with help of GUI and enters details at screen
- User is not aware of how the data is stored and what data is stored, such details are hidden from them

Data Independence

Definition: Capacity to change the schema at one level of a database system without having change in schema at next higher level

- Data independence is not possible in file processing systems, as file structure and application programs are tightly coupled together
- Data independence is possible in DBMS only because application programs do not deal with data in database directly
- Three level DBMS illustrates two types of data independence :
- Physical Data Independence
- Logical Data independence

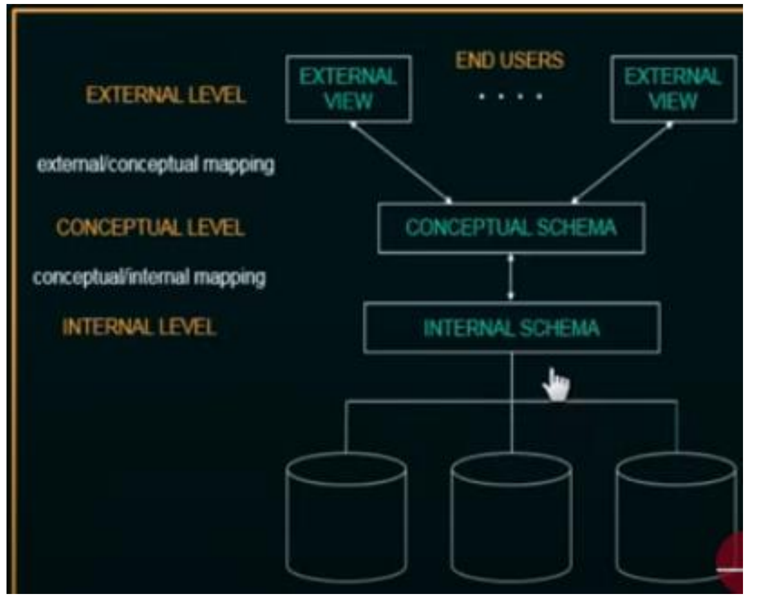
Data Independence(contd..)

Physical Data Independence

- It works in between Conceptual schema and Internal schema
- Ability to modify physical schema without changing the conceptual schema
- Modifications at the physical level are occasionally necessary to

improve performance

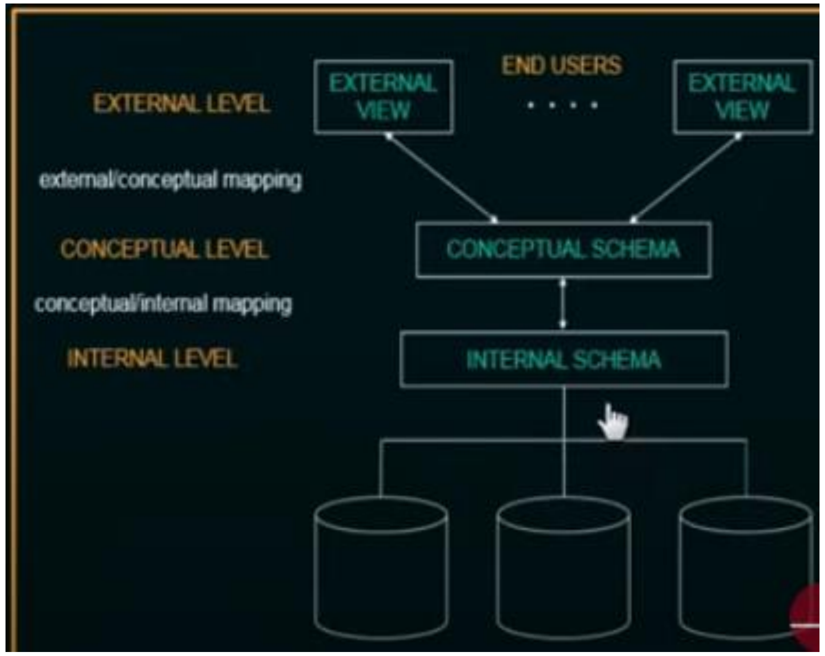
- New changes are absorbed by mapping techniques



Data Independence(contd..)

Logical Data Independence:

- It works in between **Conceptual schema and external schema**
- Ability to modify logical schema(conceptual schema) without causing external schema or application program to be rewritten
- Modifications at logical level are necessary whenever the logical structure of database is altered
- Logical data independence means if we add some new columns or remove columns from table then user view or program should not change
- Application program need not know:
    - Ordering of data fields in a record
    - Ordering of records in a file
    - Size of each record
    - Size of each field
    - Format and type of each data item
    - Whether file is sorted or not



Data Independence(contd..)

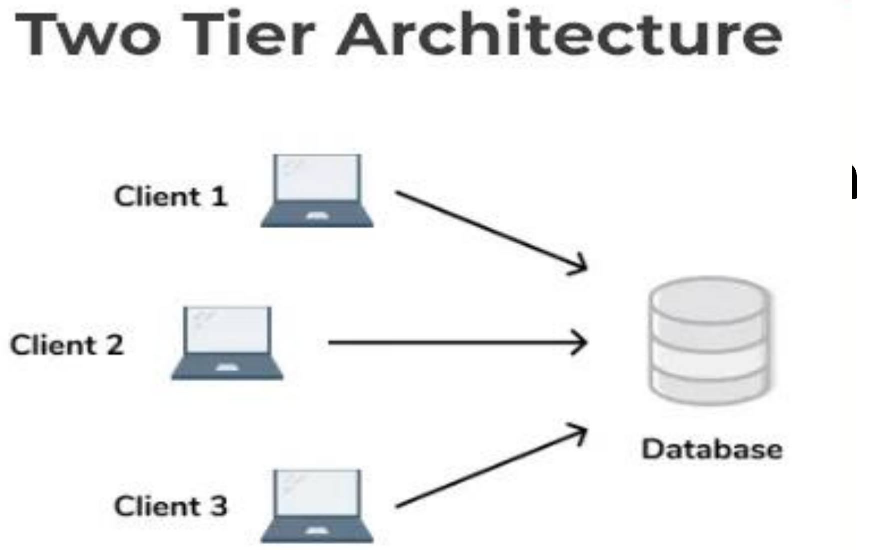
For example :

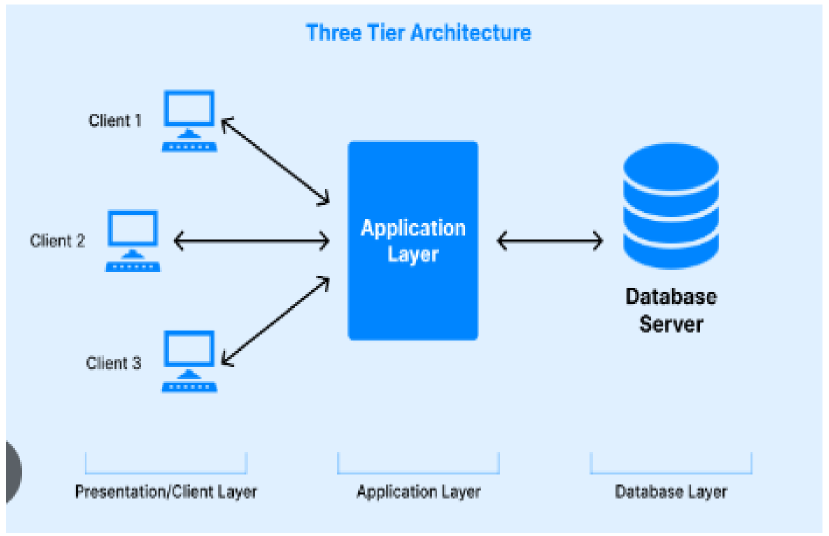
• Consider two users A & B. Both are selecting the fields "EmployeeNumber" and "EmployeeName".

• If user B adds a new column (e.g. salary) to his table, it will not affect the external view for user A, though the internal schema of the database has been changed for both users A & B.

• Logical data independence is more difficult to achieve than physical data independence, since application programs are heavily dependent on the logical structure of the data that they access.

DBMS System Architecture





- Client connects with database server with the help of JDBS-ODBC
- Client send query and database server process that query.
- Application program written in higher level language like Java, this prog. Is processed in Database server
- It is also called as client server architecture.

Eg. Traditional way of Railway ticket booking at booking counter
Traditional banking
Adv: less maintenance
Problem: Scalability
Chances of bottleneck
Security

Here Client /user interact with the system via interface, automatically query Is generated while dealing with interface.

Eg. Online booking
The query processing is done in the Application/business layer

Eg. Gmail

Adv. Scalable
Secured

- Below example is the online shopping diagram for 3 Tier architecture.
- Here frontend used is Dot Net environment while for backend database MS SQLServer 2008 is used.

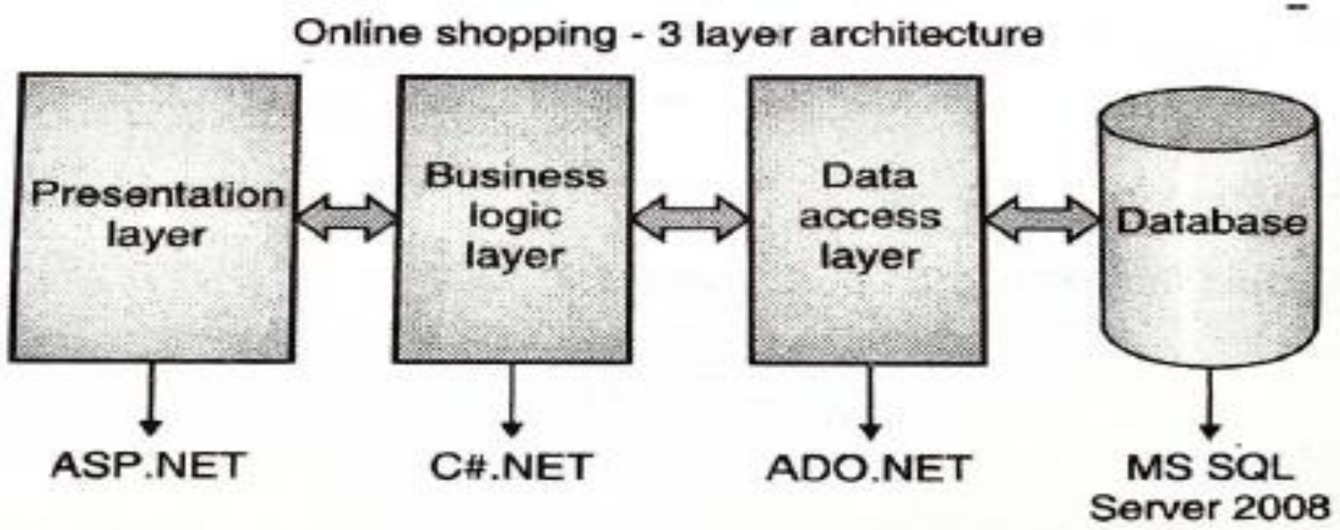


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

1. **Presentation layer:**

- This layer consist of **user interface designed** for the interaction with end user. This layer is created in ASP.Net.
- It includes the screens which will be used by the **end user for shopping.**
- Theses screen show the **products with details as per their categories.**
- User can **select the product to purchase and add them into cart.**
- This design is created with advanced controls available in ASP.Net

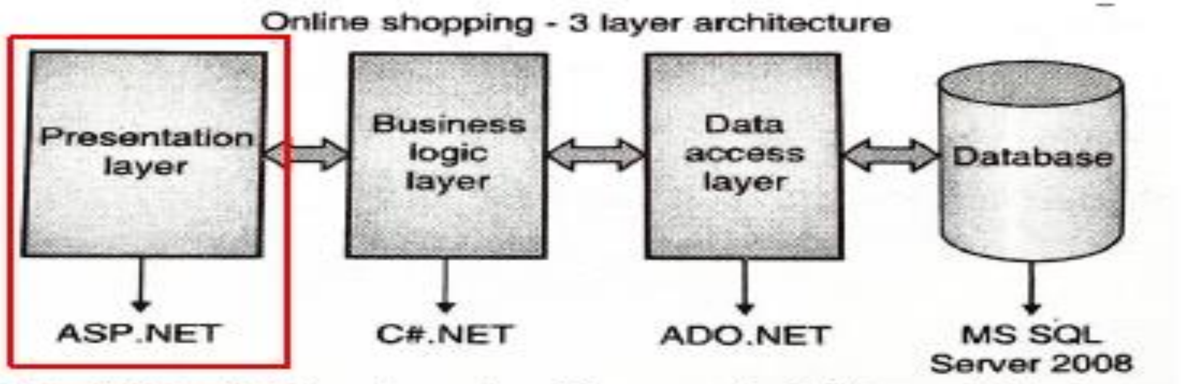


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

2. Business Layer:

- This layer consist of validation checking code related to product selection of user.
- Accidently user may select wrong number of products to purchase.
- For example if any user is giving order to purchase 10000 TV sets, then the order should be validated.
- This logical code is implemented using the C# Net.

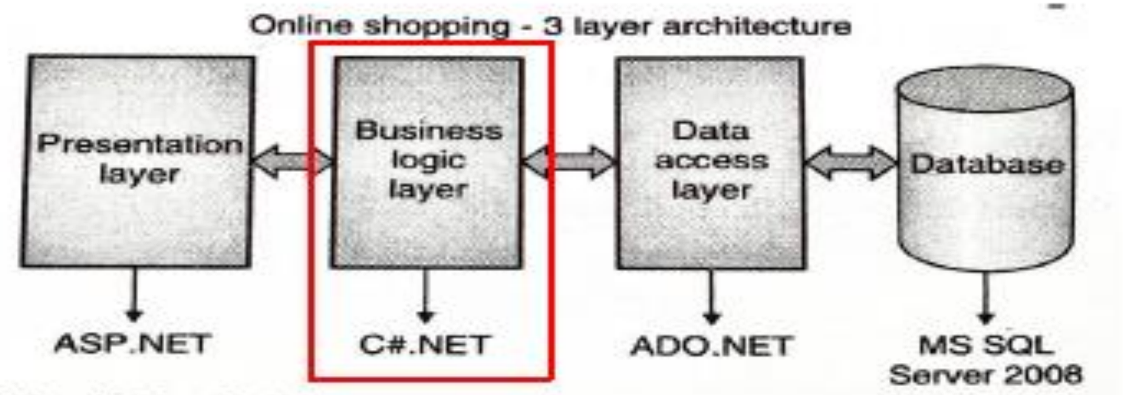


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

3. Data Layer:

- This layer contains the code interacting with database on the server.
- For example accessing product details from database, inserting transaction details of user order in database etc.
- This database handling is implemented using the ADO.Net

Here all the three layers work independently and efficiently.

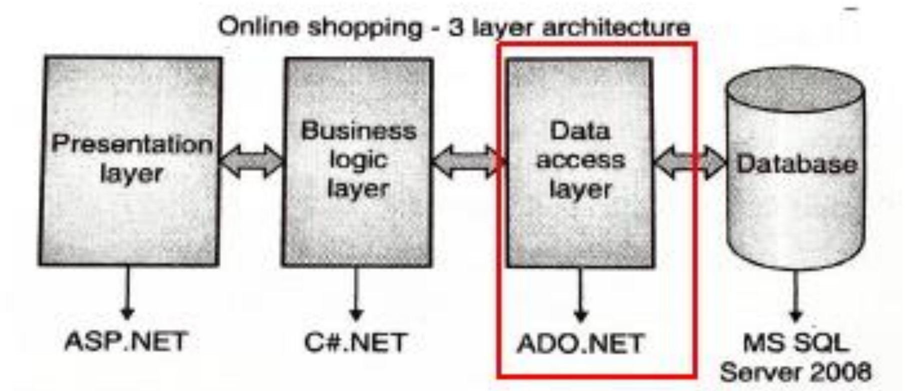


Fig. 1.7.2 : Online shopping diagram for 3-tier architecture

Three tier Architecture: Advantages

In three tier architecture we can manage the data independently.

• Can make the changes in presentation layer without affecting other two tiers.

• As each tier is independent it is possible to use different groups of developers

• It is most secure since the client doesn't have direct access to the database layer.

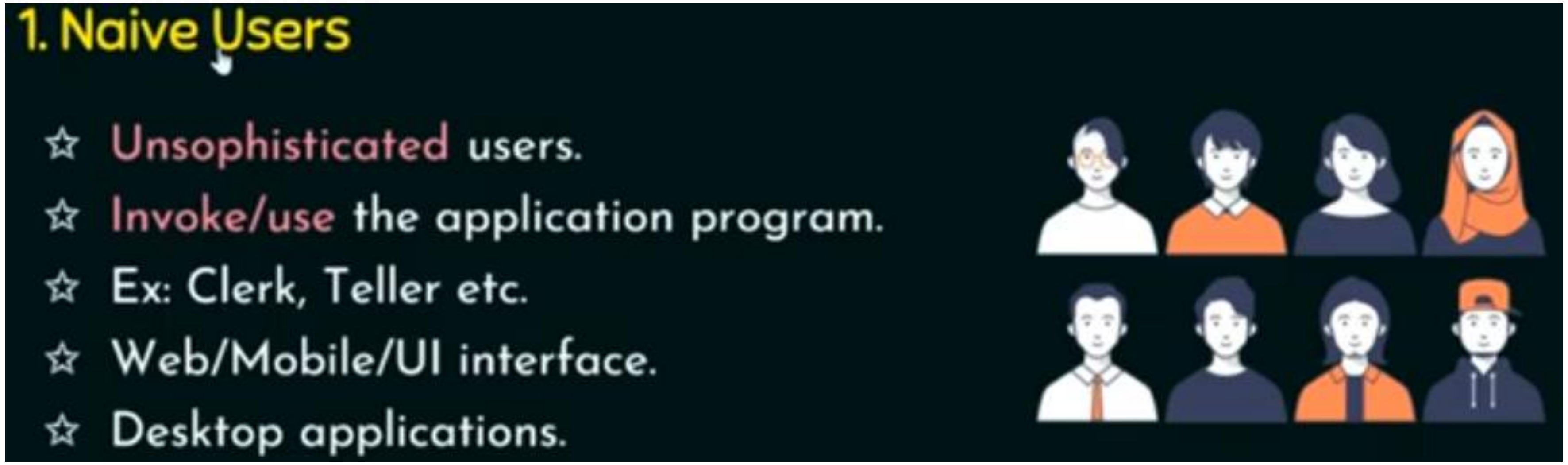
• When one tier fails there is no data loss, because you are always secure by accessing the other tier.

• Due to distributed deployment of application server, scalability is increased.

• It is reusable.

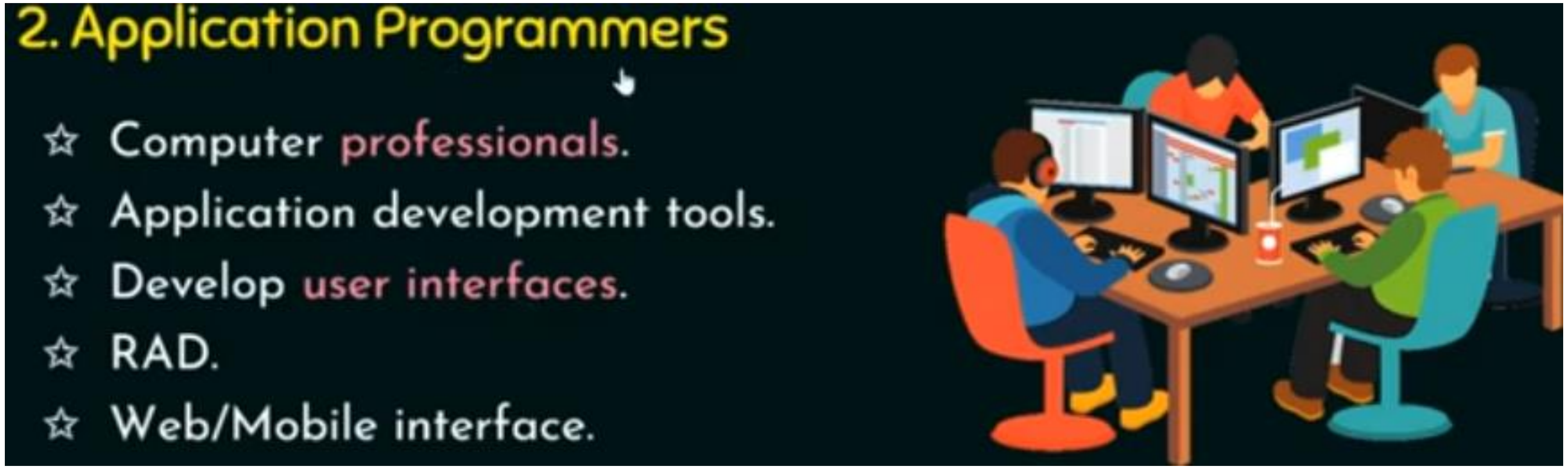
• It is robust and secure due to multiple layers.

Database Users



1. Naive Users

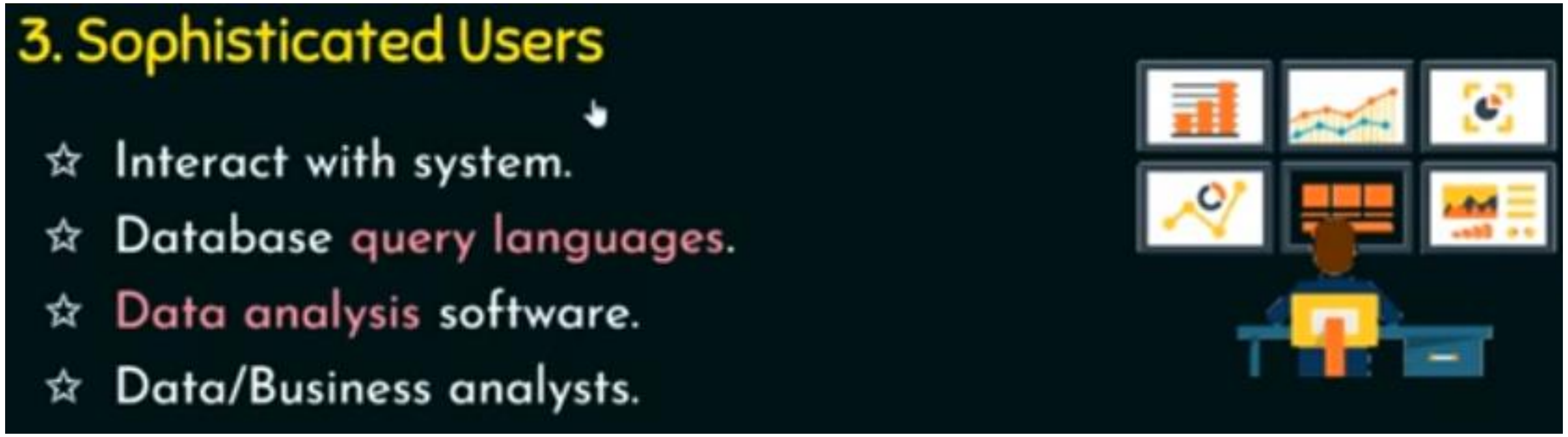
- ☆ Unsophisticated users.
- ☆ Invoke/use the application program.
- ☆ Ex: Clerk, Teller etc.
- ☆ Web/Mobile/UI interface.
- ☆ Desktop applications.



2. Application Programmers

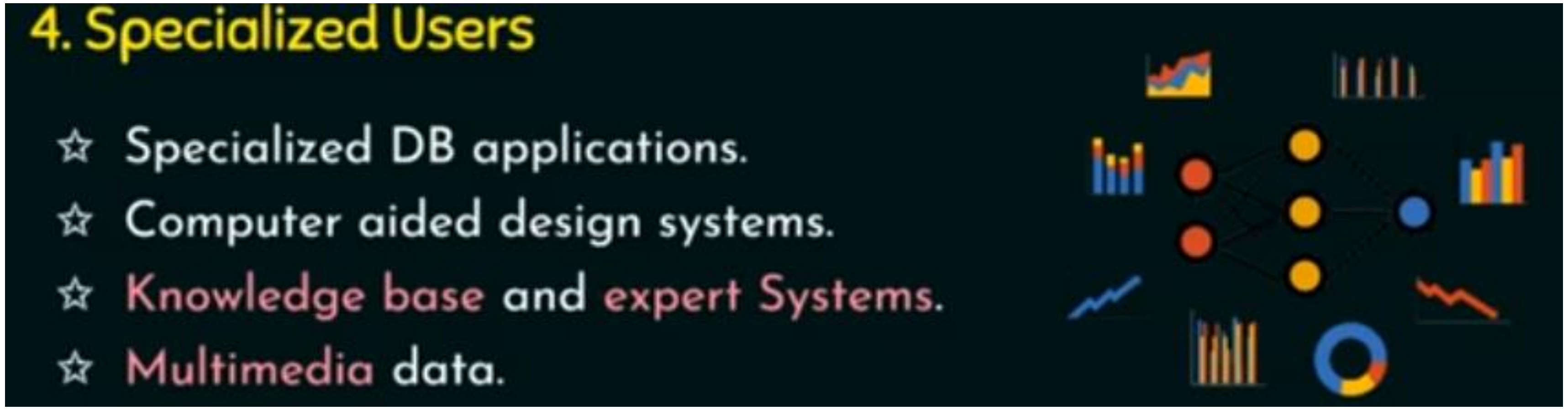
- ☆ Computer professionals.
- ☆ Application development tools.
- ☆ Develop user interfaces.
- ☆ RAD.
- ☆ Web/Mobile interface.

Database Users



3. Sophisticated Users

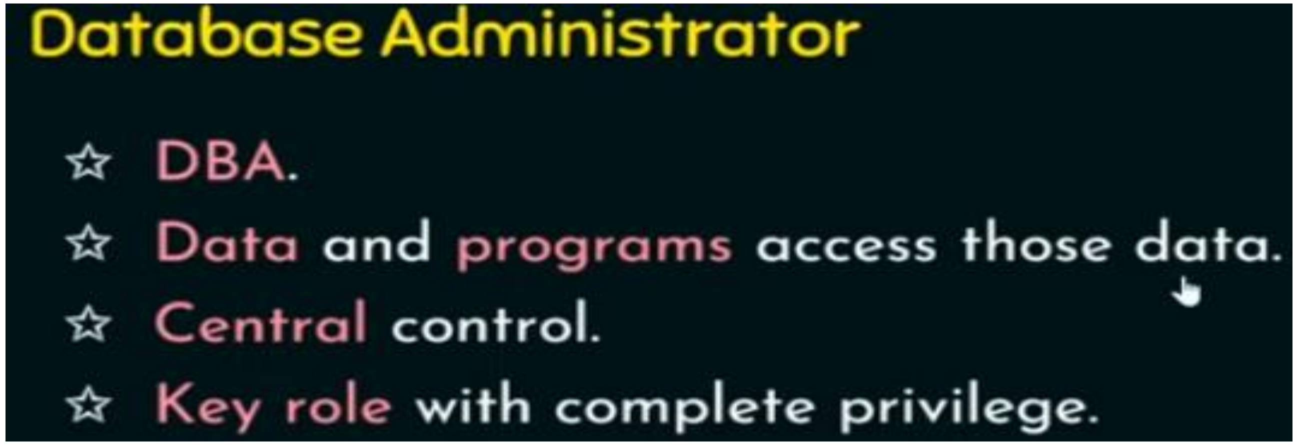
- ☆ Interact with system.
- ☆ Database query languages.
- ☆ Data analysis software.
- ☆ Data/Business analysts.



4. Specialized Users

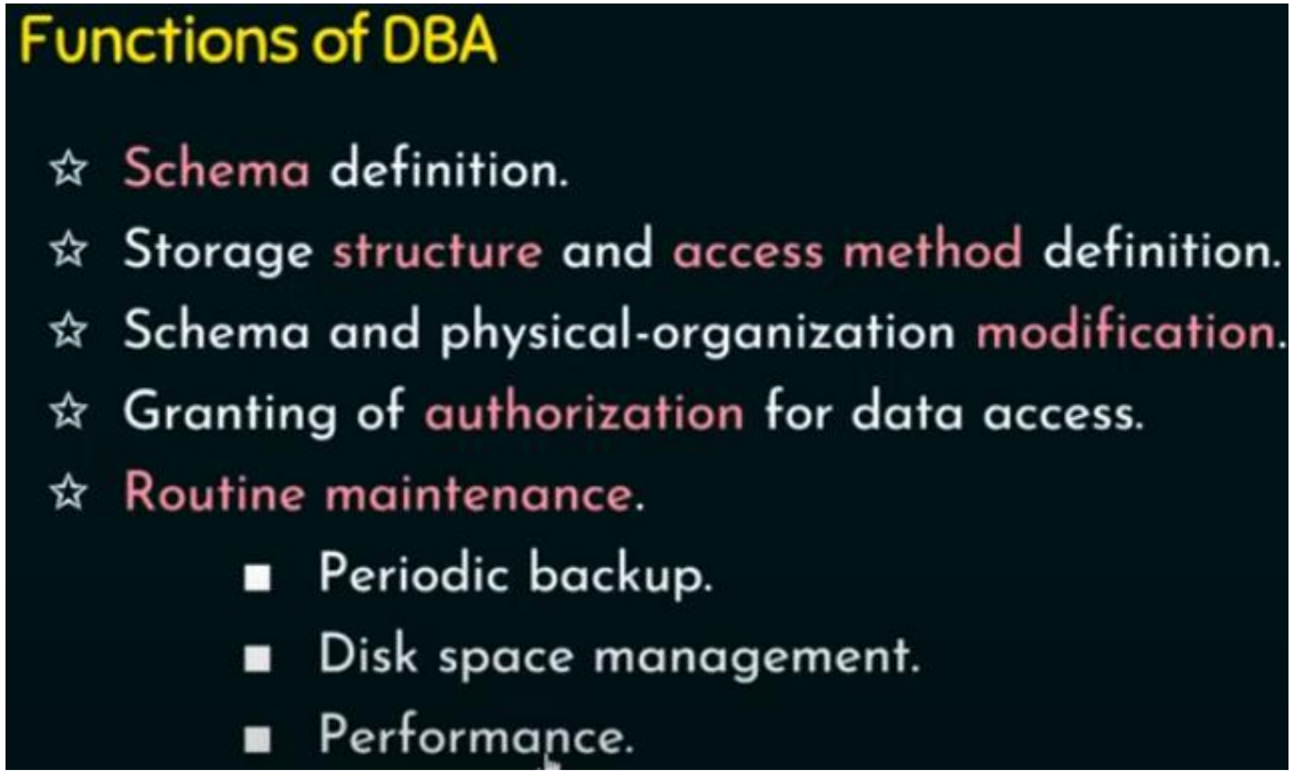
- ☆ Specialized DB applications.
- ☆ Computer aided design systems.
- ☆ Knowledge base and expert Systems.
- ☆ Multimedia data.

Database Administrator



Database Administrator

- ☆ DBA.
- ☆ Data and programs access those data.
- ☆ Central control.
- ☆ Key role with complete privilege.



Functions of DBA

- ☆ Schema definition.
- ☆ Storage structure and access method definition.
- ☆ Schema and physical-organization modification.
- ☆ Granting of authorization for data access.
- ☆ Routine maintenance.
  - ■ Periodic backup.
  - ■ Disk space management.
  - ■ Performance.

Thank You!!!