DBMS

UNIT – 1

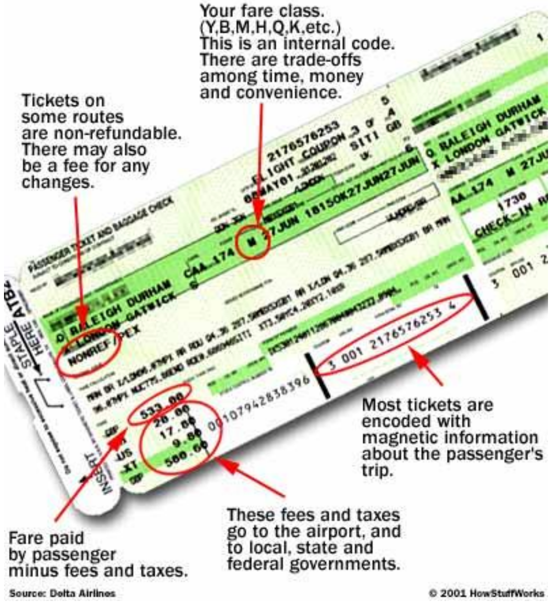
- **Introduction:**
  - Introduction to database,
  - Characteristics of Database approach,
  - Advantages of using DBMS approach,
  - Three-schema architecture and data independence.
  - DBMS Architecture.

What is Database???

- A **Database** is a collection of related data organized in a way that data can be easily accessed, managed and updated.

- A database can be of any size and complexity.
  - Phone numbers stored in SIM memory
  - Phone numbers stored on Memory card.
- A database may be generated and maintained manually or it may be computerized.
  - Library cards in your colleges (PUC)
  - Library cards in GIT

Other databases you may use



Your fare class. (Y,B,M,H,Q,K,etc.) This is an internal code. There are trade-offs among time, money and convenience.

Tickets on some routes are non-refundable. There may also be a fee for any changes.

Most tickets are encoded with magnetic information about the passenger's trip.

These fees and taxes go to the airport, and to local, state and federal governments.

Fare paid by passenger minus fees and taxes.

Source: Delta Airlines

© 2001 HowStuffWorks

Implicit properties of Database

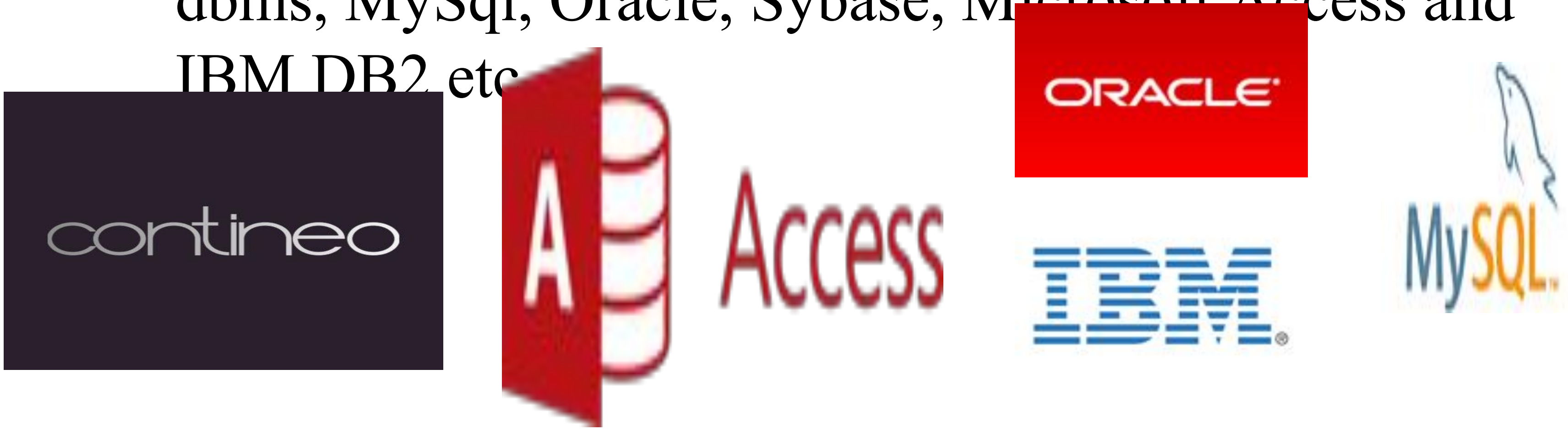
- It represents some aspects of real world.
- It is logically collection of data with some inherent meaning.
- It is designed, built and populated with data for a specific purpose.

Implicit properties of Database

- It represents some aspects of real world.
- It is logically collection of data with some inherent meaning.
- It is designed, built and populated with data for a specific purpose.

What is DBMS???

- A **DBMS** is a software that allows creation, definition and manipulation of database. Dbms is actualy a tool used to perform any kind of operation on data in database. Dbms also provides protection and security to database. It maintains data consistency in case of multiple users. Here are some examples of popular dbms, MySql, Oracle, Sybase, Microsoft Access and IBM DB2 etc.



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

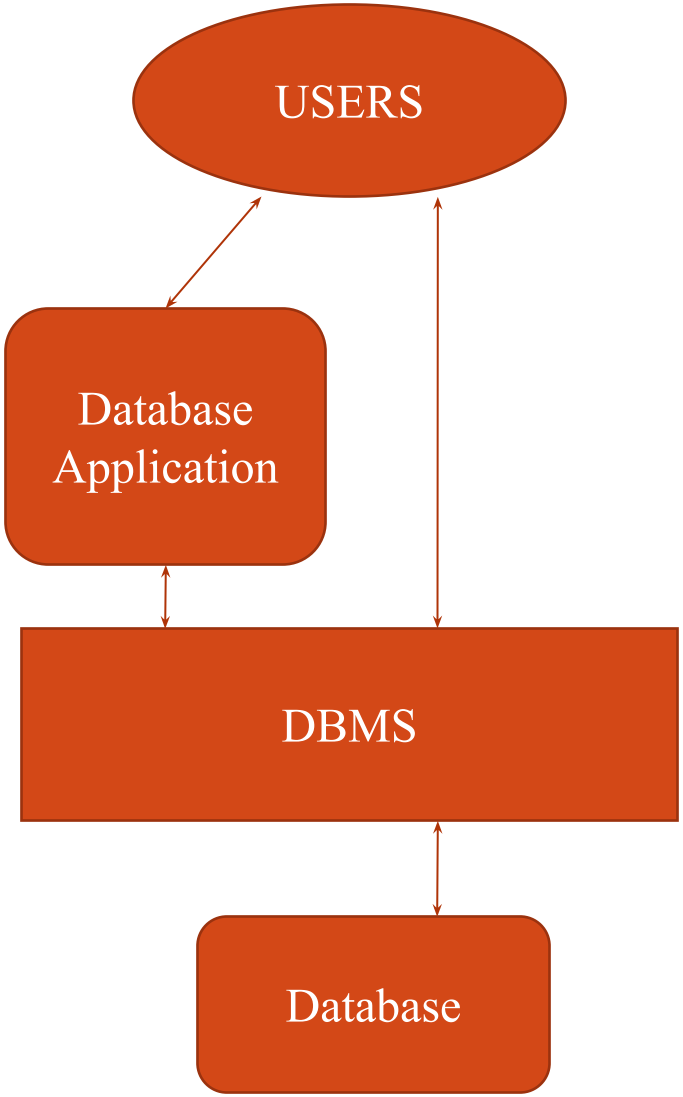
Other features:

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

Examples of Database Applications

- Purchases using your credit card
- Booking a holiday at the travel agents
- Using the local library
- Taking out insurance
- Using the Internet
- Studying at university
- Many more …….

Components of Database System



- **Users :** Users may be of various type such as DB administrator, System developer and End users.
- **Database application :** Database application may be Personal, Departmental, Enterprise and Internal
- **DBMS :** Software that allow users to define, create and manages database access, Ex: MySql, Oracle etc.
- **Database :** Collection of logical data.

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.
- 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.

- 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. OLTP (Online Transaction Processing) is a major part of database applications

Actors on the scene

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

  a)Casual end users (Managers)

  b)Parametric end users(Accountant)

  c) Sophisticated end users (Engineers)

  d) Stand-alone end users (Users)

Actors on the scene

- **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.
- **End users:** they use the data for queries, reports and
  he te t

Workers behind the scene

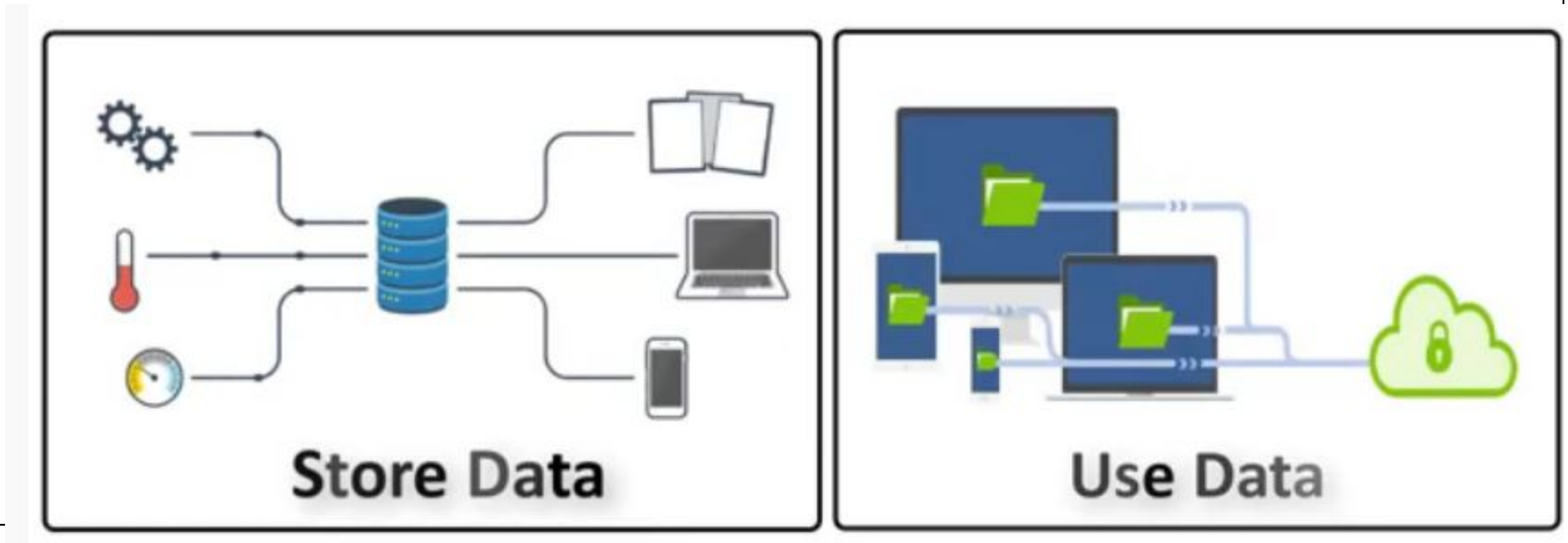
- **DBMS System designer and implementers:** design and implement the DBMS modules and interfaces as a software package. It includes: catlogs, query processing, interfaces, accessing data, data recovery and security.
- **Tool developers:** designs and implements tools. They include packages for database design, performance monitoring, simulations, test data generation.
- **Operators and maintenance personnel:** are responsible for actually running and maintenance of the hardware and software environment of the database system.

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 persistent storage for program Objects
- Providing Storage Structures 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.
- Drawing Inferences and Actions using rules

DBMS ARCHITECTURE

- The design of DBMS depends on its architecture.
- Architecture controls how to store and how to use data.



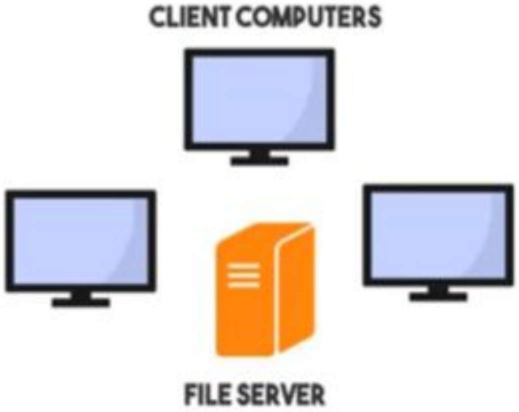
DBMS ARCHITECTURE

- Its divided logically into 3 types:
  - One Tier
  - Two Tier
  - Three Tier

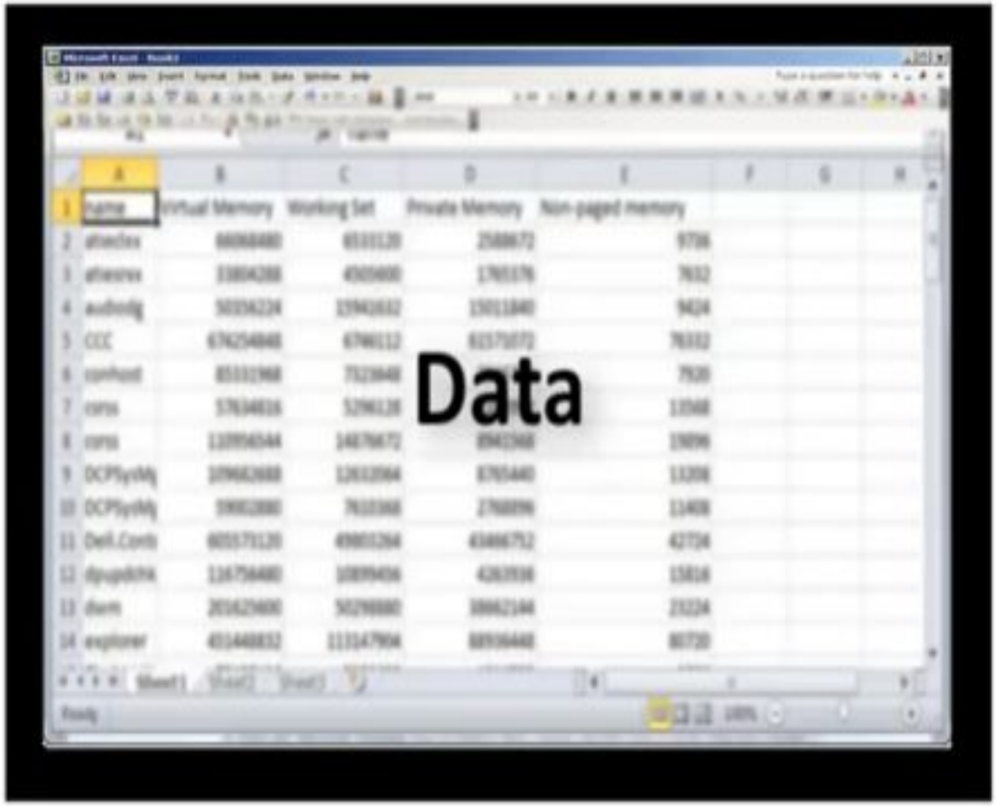
DBMS Architecture Types
One Tier
Two Tier
Three Tier

One Tier Architecture

- All in one computer
- Application
- Data
- Presentation
  - Eg: MS office

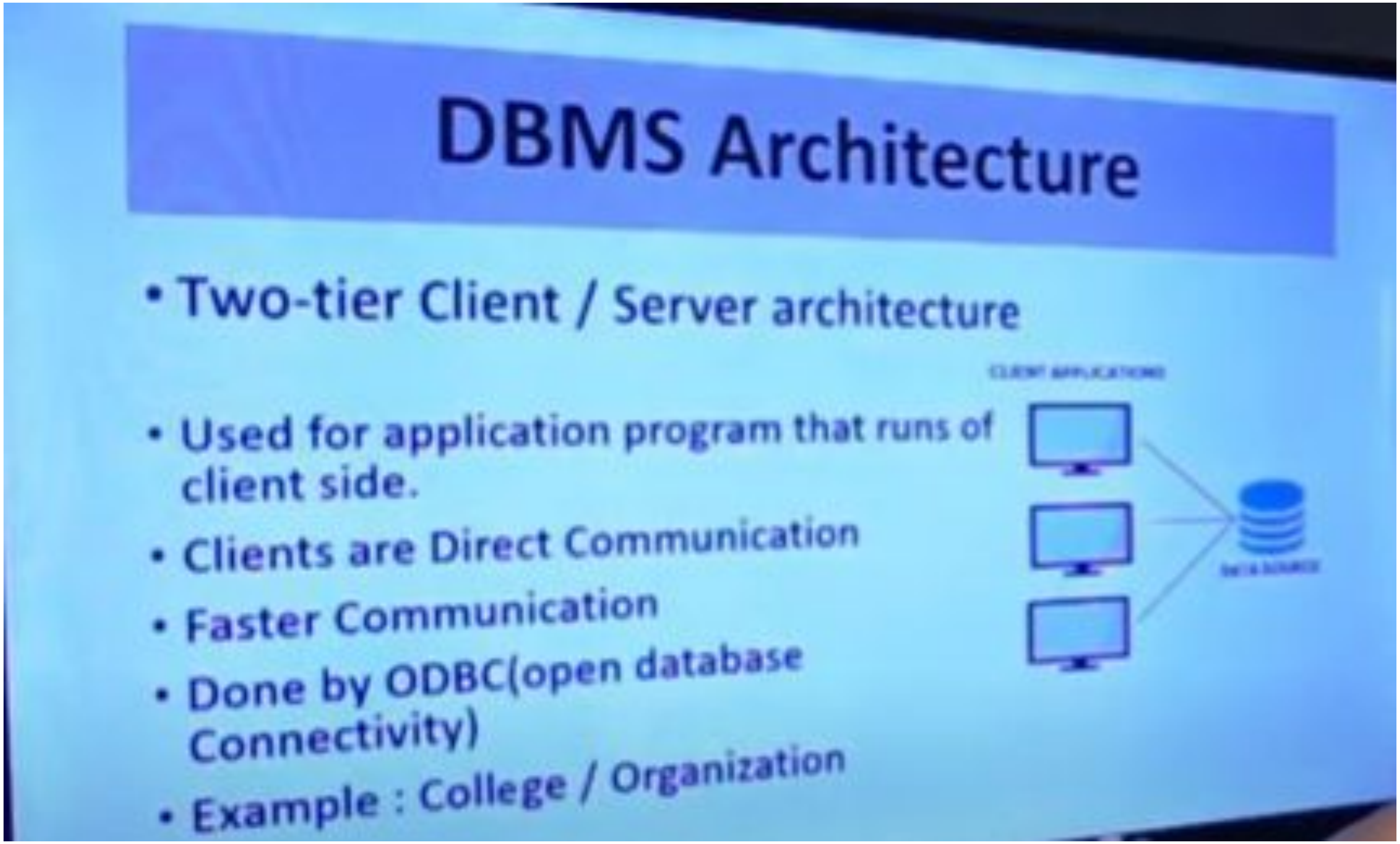


Application


Data

	A	B	C	D	E
1	name	Virtual Memory	Working Set	Private Memory	Non-paged memory
2	atiedxx	66068480	6533120	2588672	9736
3	atiesrxx	33804288	4505600	1765376	7632
4	audiodg	50356224	15941632	15011840	9424
5	CCC	674254848	6746112	61571072	76332
6	conhost	85331968	7323648	[illegible]	7920
7	csrss	57634816	5296128	[illegible]	13568
8	csrss	110956544	14876672	8941568	19896
9	DCPSysMg	109662688	12632064	8765440	13208
10	DCPSysMg	59002880	7610368	2768896	13408
11	Dell.Contr	60571120	49803264	43466752	42734
12	dpupdchk	116756480	10899456	4263936	15816
13	dwm	201625600	50298880	38662144	23224
14	explorer	451448832	111147904	88936448	80720

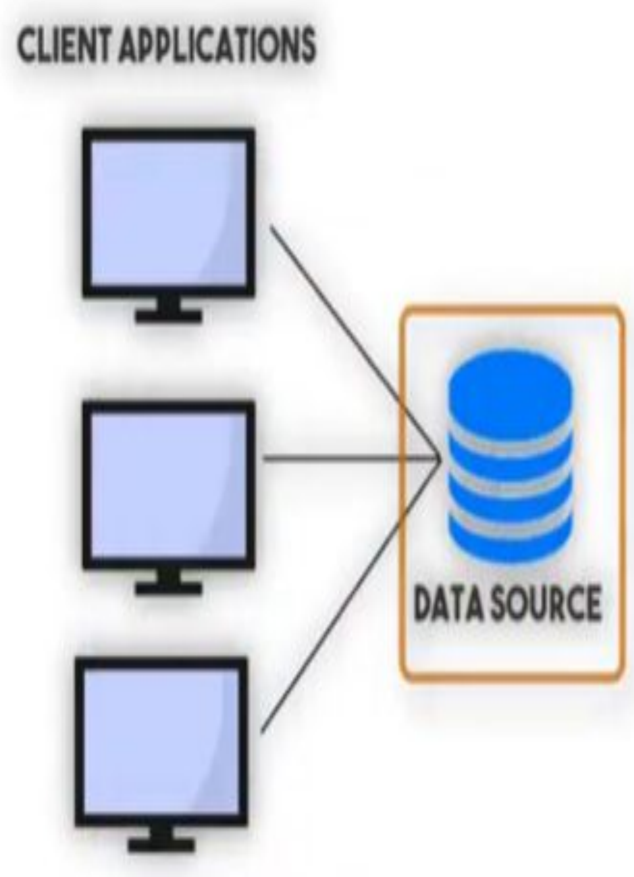
Two Tier Architecture



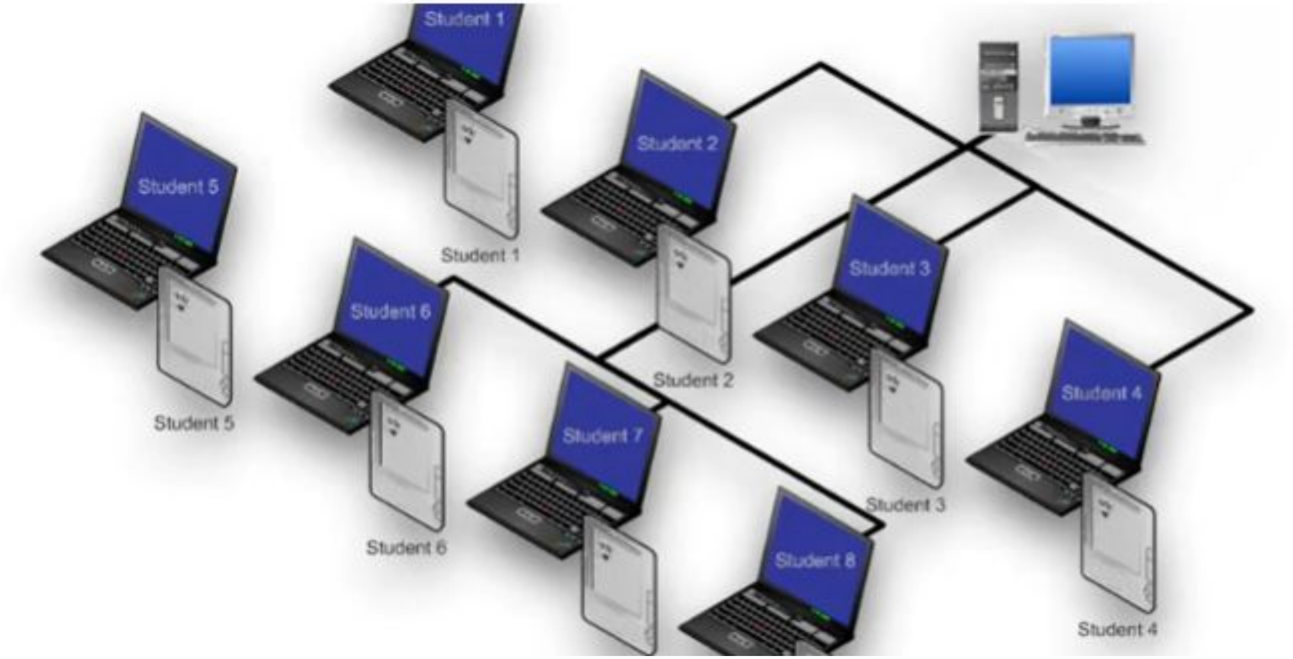
DBMS Architecture

- Two-tier Client / Server architecture
- Used for application program that runs of client side.
- Clients are Direct Communication
- Faster Communication
- Done by ODBC(open database Connectivity)
- Example : College / Organization

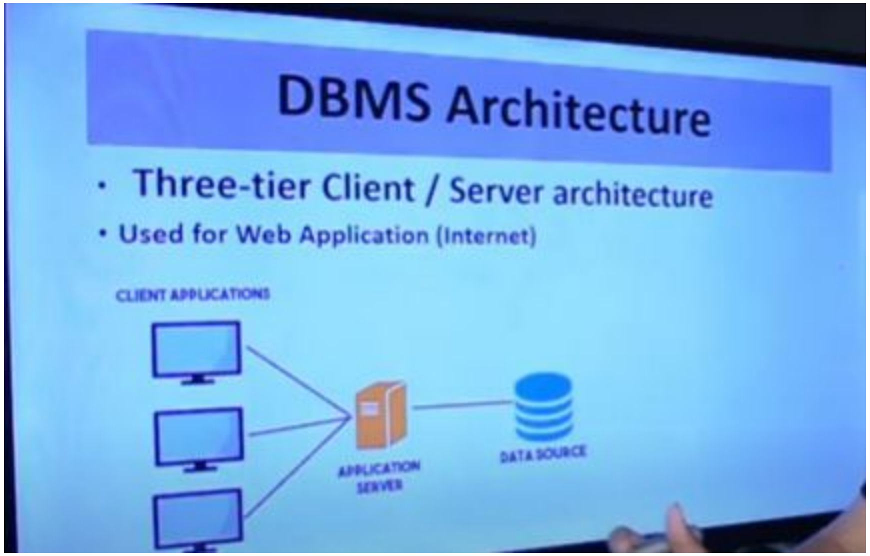
- Two computers
  - One client
  - One server



Example


Student 1
Student 1
Student 2
Student 2
Student 5
Student 5
Student 3
Student 3
Student 6
Student 6
Student 4
Student 4
Student 7
Student 8

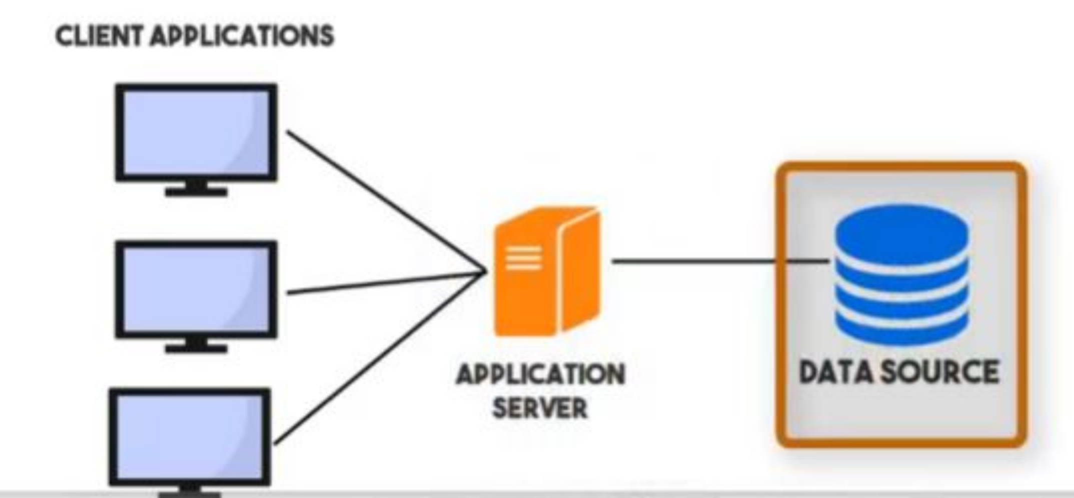
Three Tier Architecture



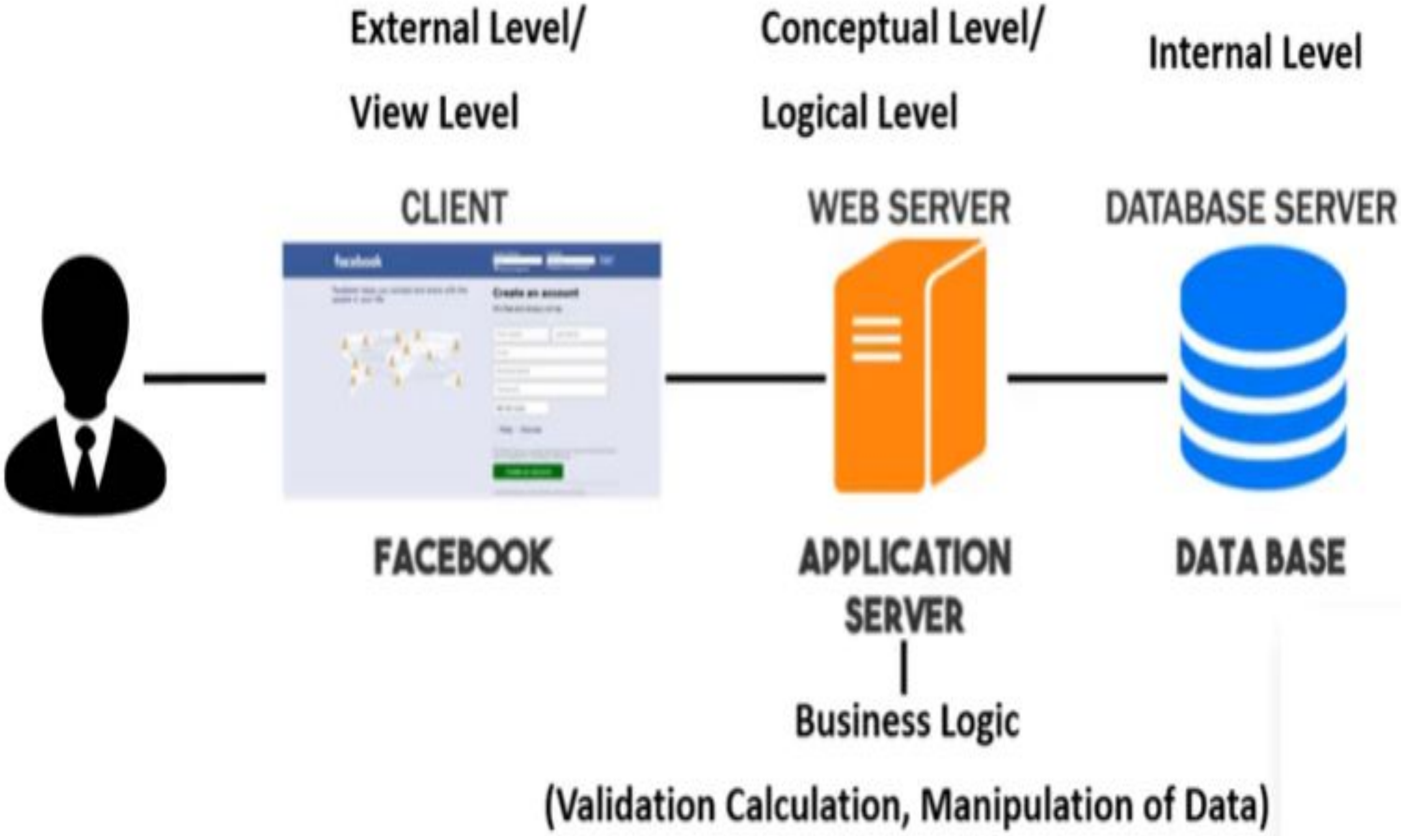
DBMS Architecture

- Three-tier Client / Server architecture
- Used for Web Application (Internet)

CLIENT APPLICATIONS
APPLICATION SERVER
DATA SOURCE



example



External Level/

View Level

Conceptual Level/

Logical Level

Internal Level

CLIENT

WEB SERVER

DATABASE SERVER

FACEBOOK

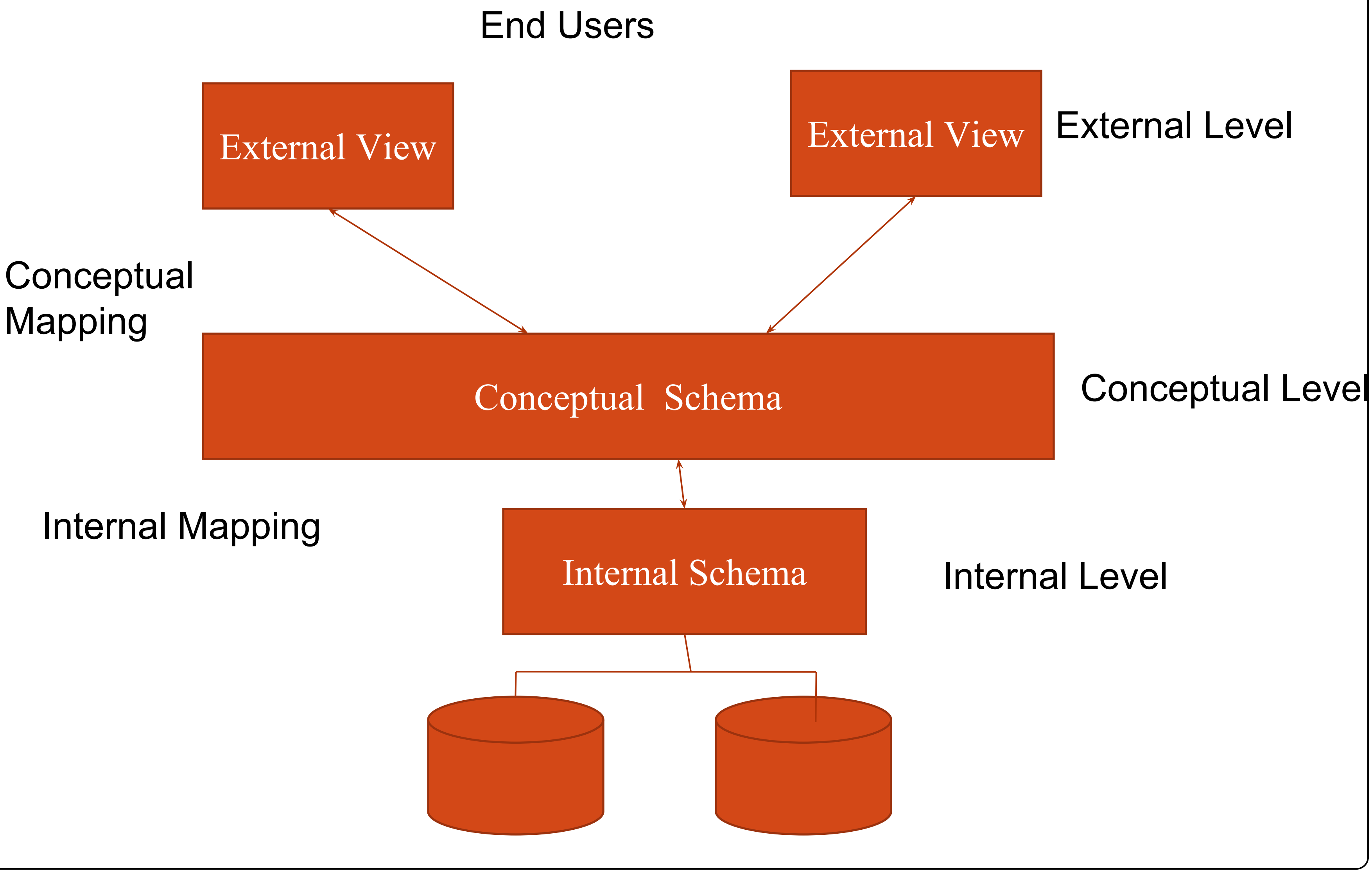
APPLICATION SERVER

DATA BASE

Business Logic

(Validation Calculation, Manipulation of Data)

Three-Schema Architecture



End Users

External View

External View

External Level

Conceptual Mapping

Conceptual Schema

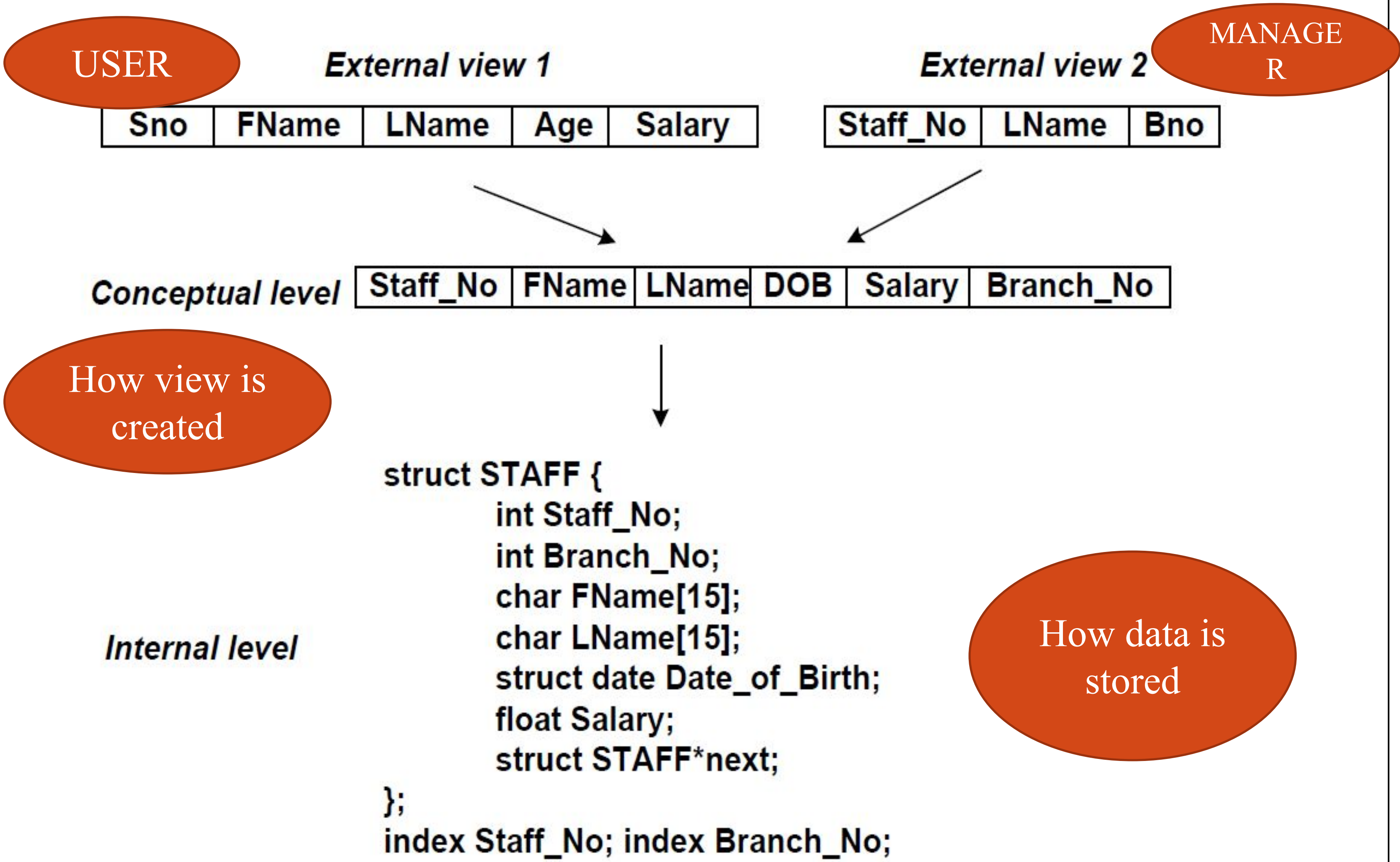
Conceptual Level

Internal Mapping

Internal Schema

Internal Level

Example of 3 Schema Architecture



USER

External view 1

Sno	FName	LName	Age	Salary

MANAGE
R

External view 2

Staff_No	LName	Bno

Conceptual level

Staff_No	FName	LName	DOB	Salary	Branch_No

How view is created

Internal level

```
struct STAFF {
        int Staff_No;
        int Branch_No;
        char FName[15];
        char LName[15];
        struct date Date_of_Birth;
        float Salary;
        struct STAFF*next;
};
index Staff_No; index Branch_No;
```

How data is stored

- **EXTERNAL LEVEL (highest level)**
  - The user’s view of the database.
  - Consists of a number of different external views of the DB.
  - Describes part of the DB for particular group of users.
  - Provides a powerful and flexible security mechanism by hiding parts of the DB from certain users.
  - It permits users to access data in a way that is customized to their needs, so that the same data can be seen by different users in different ways, at the same time.

- **CONCEPTUAL LEVEL**
  - The logical structure of the entire database as seen by DBA
  - What data is stored in the database
  - The relationships among the data
  - Represents:
    - entities, attributes, relations
    - constraints on data
    - semantic information on data
    - security, integrity information

- **INTERNAL LEVEL**
  - Physical representation of the DB on the computer
  - How the data is stored in the database
  - Storage space allocation for data and indexes
  - Data compression, encryption
  - Record description for storage

Three-Schema Architecture

- Defines DBMS schemas at *three levels*:
  - **Internal schema** at the internal level to describe physical storage structures and access paths. Typically uses a *physical* data model.
  - **Conceptual schema** at the conceptual level to describe the structure and constraints for the *whole* database for a community of users. Uses a *conceptual* or an *implementation* data model.
  - **External schemas** at the external level to describe the various user views. Usually uses the same data model as the conceptual level.

Data Independence

- **Logical Data Independence:** The capacity to change the conceptual schema without having to change the external schemas and their application programs.
- **Physical Data Independence:** The capacity to change the internal schema without having to change the conceptual schema.

When not to use a DBMS

- **Main inhibitors (costs) of using a DBMS**:
  - High initial investment and possible need for additional hardware.
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions.
- **When a DBMS may be unnecessary:**
  - If the database and applications are simple, well defined, and not expected to change.
  - If access to data by multiple users is not required.
- **When no DBMS may suffice:**
  - If the database system is not able to handle the complexity of data because of modeling limitations