UCS310: DATABASE MANAGEMENT SYSTEM

INTRODUCTION TO DATABASE

Syllabus
(L-T-P-Cr:3-0-2-4)

- **Course Objectives:** Emphasis is on the need of database systems. Main focus is on E-R diagrams, relational database, concepts of normalization and denormalization and SQL commands.

- **Introduction**: Data, data processing requirement, desirable characteristics of an ideal data processing system, traditional file based system, its drawback, concept of data dependency, Definition of database, database management system, 3-schema architecture, database terminology, benefits of DBMS.

- **Relational Database:** Relational data model: Introduction to relational database theory: definition of relation, keys, relational model integrity rules.

- **Database Analysis:** Conceptual data modeling using E-R data model - entities, attributes, relationships, generalization, specialization, specifying constraints, Conversion of ER Models to Tables, Practical problems based on E-R data model.

Syllabus
(Continued)

- **Relational Database Design:** Normalization- 1NF, 2NF, 3NF, BCNF, 4NF and 5NF. Concept of De-normalization and practical problems based on these forms.

- **Transaction Management and Concurrency control:** Concept of Transaction, States of Transaction and its properties, Need of Concurrency control, concept of Lock, Two phase locking protocol.

- **Recovery Management:** Need of Recovery Management, Concept of Stable Storage, Log Based Recovery Mechanism, Checkpoint.

- **Database Implementation:** Introduction to SQL, DDL aspect of SQL, DML aspect of SQL – update, insert, delete & various form of SELECT- simple, using special operators, aggregate functions, group by clause, sub query, joins, co-related sub query, union clause, exist operator. PL/SQL - cursor, stored function, stored procedure, triggers, error handling, and package.

Syllabus
(Continued)

- **Laboratory work:** Students will perform SQL commands to demonstrate the usage of
  - DDL and DML,
  - joining of tables,
  - grouping of data and will implement
  - PL/SQL constructs.
  - They will also implement one project.

Project: It will contain database designing & implementation, should be given to group of 2-4 students. While doing projects emphasis should be more on back-end programming like use of SQL, concept of stored procedure, function, triggers, cursors, package etc. Project should have continuous evaluation and should be spread over different components.

Syllabus
(Continued)

- **Course Learning Outcomes (CLOs) / Course Objectives (COs): On completion of this course, the students will be able to:**

1. Analyze the Information Systems as socio-technical systems, its need and advantages as compared to traditional file-based systems.
2. Analyze and design database using E-R data model by identifying entities, attributes and relationships.
3. Apply and create Relational Database Design process with Normalization and Denormalization of data.
4. Comprehend the concepts of transaction management, concurrence control and recovery management.
5. Demonstrate use of SQL and PL/SQL to implementation database applications.

Syllabus
(Continued)

Text Books:

1. Silverschatz A., Korth F. H. and Sudarshan S., **Database System Concepts**, Tata McGraw Hill (2010) 6th ed.
2. Elmasri R. and Navathe B. S., **Fundamentals of Database Systems**, Pearson (2016) 7th ed.

Reference Books:

1. Bayross I., SQL, PL/SQL the Programming Language of Oracle, BPB Publications (2009) 4th ed.
2. Hoffer J., Venkataraman, R. and Topi, H., Modern Database Management, Pearson (2016) 12th ed.
3. Parteek Bhatia and Gurvinder Singh, Simplified Approach to DBMS.

Introduction to Database

Data

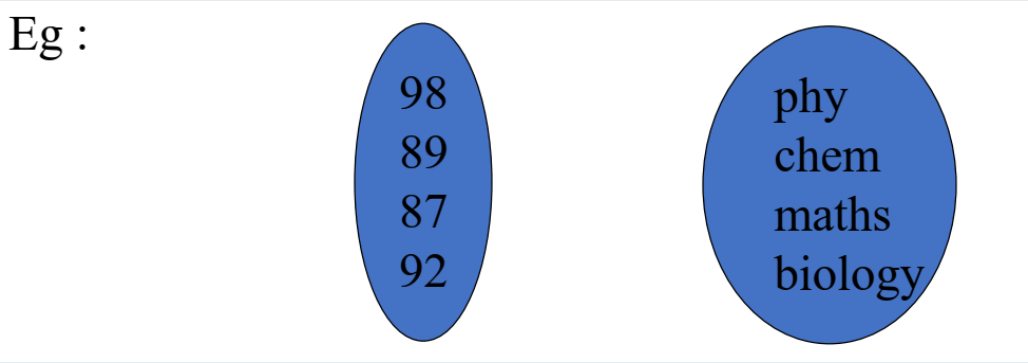
- Data comes from the Latin word **datum**, meaning "something given." Over the time, the English language has evolved to use data as plural.

"Data is the raw fact and figures that can be recorded, stored and processed by any computing machine."

- When we talk about data, we think of megabytes of binary code.
- Data can be any character, text, words, number, pictures, sound, video etc.

Data

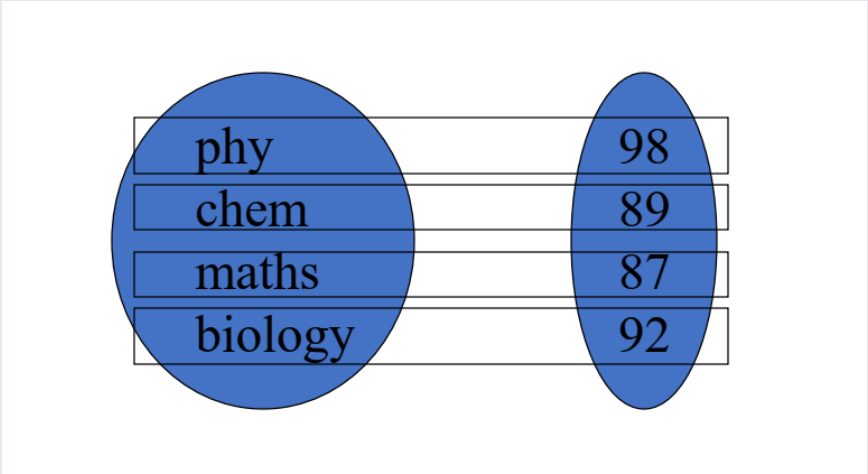
- Data may be numerical data which may be integers or floating point numbers, and non-numerical data such as characters, date and etc.,



The above numbers may be anything: It may be distance in kms or amount in rupees or no of days or marks in each subject etc.,

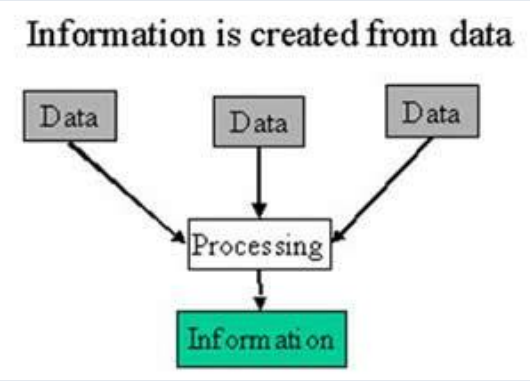
Information

- **Information is RELATED DATA**



Information

- The word 'Information' is derived from latin word '*Informare*', which means '**to instruct**'.
- The word ***information*** has existed in the English language for far longer than the word *data*.
- When data is processed, organized, structured or presented in a given context so as to make it useful, it is called **information**.
- Ex1:
  - Data: Shyam, 85, Physics, Marks, Scored, in.
  - Information: Shyam scored 85 marks in Physics.
- Ex2:
  - Each student's test score is **one piece of data**.
  - The average score of a class or of the entire school is **information** that can be derived from the given data.



Difference between Data and Information

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.

- **Data:** I have one item. *The data displays a 1, not a zero.*
- **Information:** It's a tomato. *Now, we understand the item and its characteristics.*
- **Knowledge:** A tomato is a vegetable.
- **Wisdom (Decision making):** Tomato is never added to a fruit salad.

How we Store data?

- A *data store* is a repository for storing collections of data, such as a database, a file system or a directory.
  - File System
  - Database

File System

- **File:** A collection of records or documents dealing with one organization, person, area or subject
  - Manual (paper) files
  - Computer files

Various Types of files

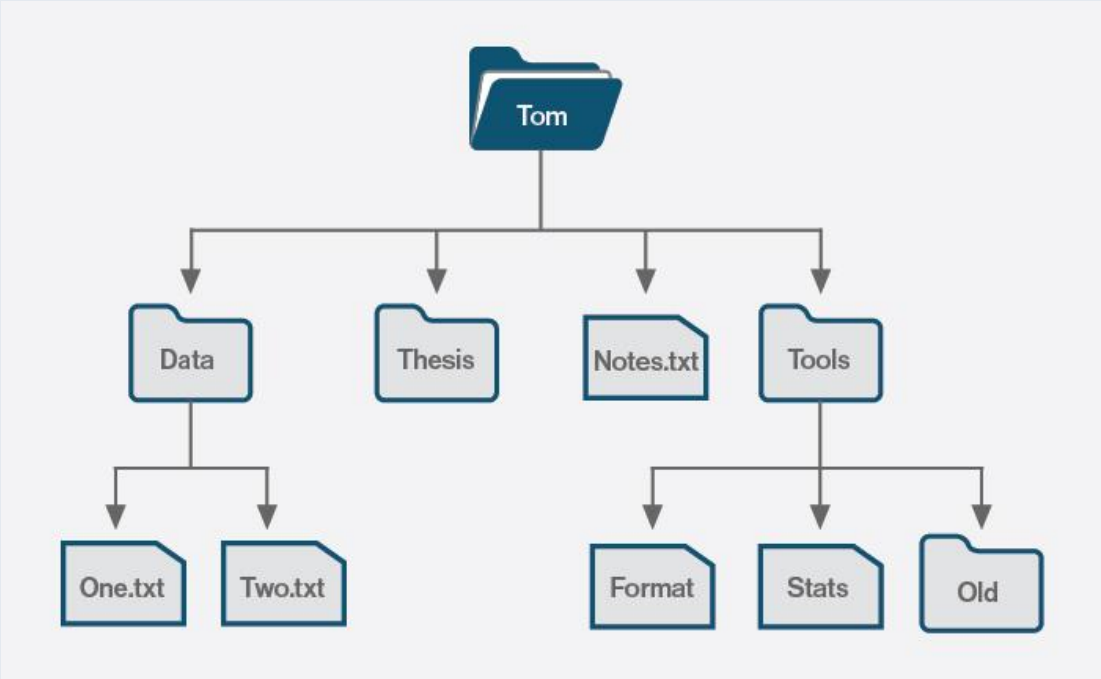
Depending upon the type of content, a file can be categorized as:

1. Data file
2. Program file
3. Object code file
4. Executable file
5. Text file

File System

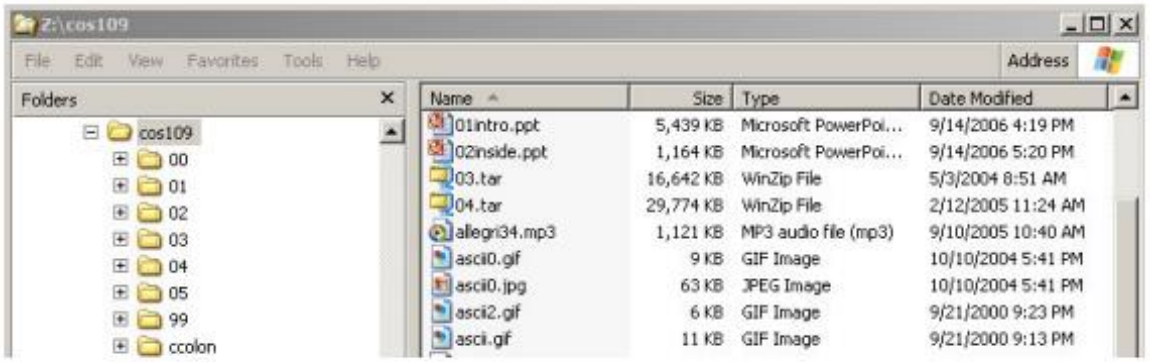
- A **file** is just a sequence of bytes of information.
- The contents of a file is determined and interpreted by programs, not the operating system.
  - A **file system** is the method and data structure that an operating system uses to keep track of files on a disk or partition.

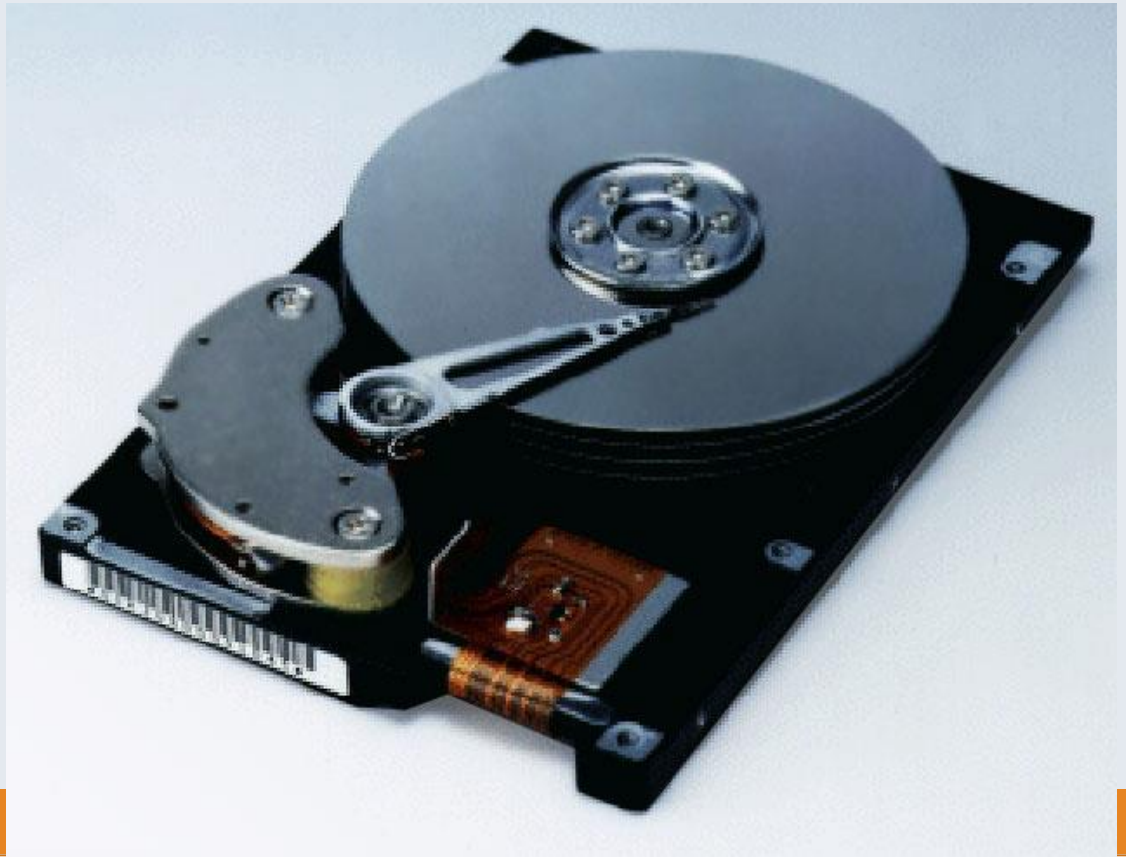
    or
  - A **file System** is the software that provides hierarchical storage and organization of files, usually on a single computer – part of the operating system


Tom
Data
Thesis
Notes.txt
Tools
One.txt
Two.txt
Format
Stats
Old

Managing Files

- **Logical Structure:** users and programs see a hierarchy of folders (or directories) and files
  - **Folder:**
    - a folder contains references to folder and files
    - "root" folder ultimately leads to all others
    - a folder is a special file that contains names of other folders & files plus other information like size, time of change, etc.
    - contents are completely controlled by the operating system
- **Physical Structure:** disk drives operate in tracks, sectors, etc. (other storage devices have other physical properties)
  - **Disks:**
    - A place to store information when the power is turned off
    - Usually based on magnetic surfaces, rotating machinery

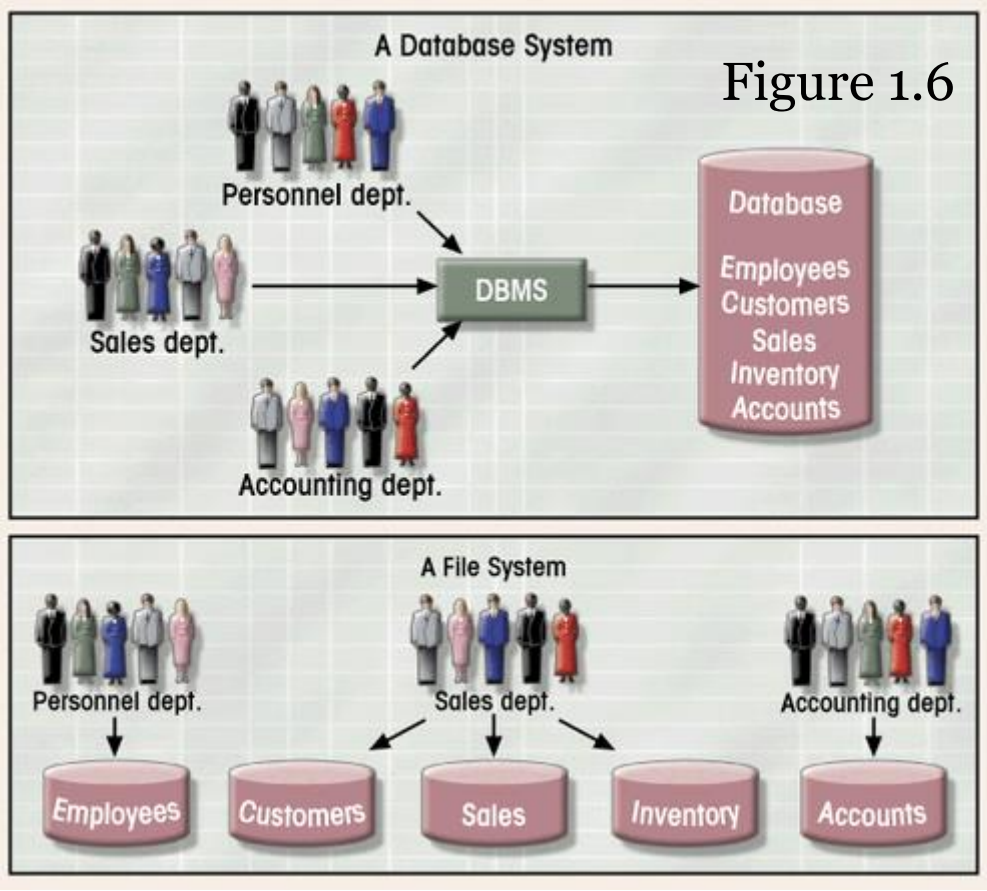

Z:\cos109
File Edit View Favorites Tools Help
Address
Folders
cos109
00
01
02
03
04
05
99
ccolon
Name Size Type Date Modified
01intro.ppt 5,439 KB Microsoft PowerPoi... 9/14/2006 4:19 PM
02inside.ppt 1,164 KB Microsoft PowerPoi... 9/14/2006 5:20 PM
03.tar 16,642 KB WinZip File 5/3/2004 8:51 AM
04.tar 29,774 KB WinZip File 2/12/2005 11:24 AM
allegri34.mp3 1,121 KB MP3 audio file (mp3) 9/10/2005 10:40 AM
ascii0.gif 9 KB GIF Image 10/10/2004 5:41 PM
ascii0.jpg 63 KB JPEG Image 10/10/2004 5:41 PM
ascii2.gif 6 KB GIF Image 9/21/2000 9:23 PM
ascii.gif 11 KB GIF Image 9/21/2000 9:13 PM



Managing Files(cntd..)

- The operating system converts between these two views
  - does whatever is necessary to maintain the file/folder illusion
  - hides physical details so that programs don't depend on them
  - presents a uniform interface to disparate physical media
- "file system" is the part of the operating system that does this conversion

Database System vs. File System



Drawbacks of File Systems

- Consider a bank want to keep information about all the saving accounts
  - The bank is using file system
  - To allow users to manipulate information, the system has a number of application programs, that manipulates the files,
  - Example:
    - A program to debit or credit an account
    - A program to add a new account
    - A program to find the balance of an account
    - A program to generate monthly statement

  Suppose, A bank decides to offer current account. For this bank will create new permanent files to store information of checking account and also new application program for checking accounts. Such as

  A program to check overdraft(Withdrawing more money than balance)

  This will lead to More Files and More Programs with time

Disadvantages of File System

1. Data Redundancy
   - The files are created in the file system as and when required by an enterprise over its growth path. So in that case the repetition of information about an entity cannot be avoided.

   Eg. The addresses of customers will be present in the file maintaining information about customers holding savings account and also the address of the customers will be present in file maintaining the current account. Even when same customers have a saving account and current account his address will be present at two places.

2. Inconsistency

Data redundancy leads to greater problem than just wasting the storage i.e. it may lead to **inconsistent data**. Same data which has been repeated at several places may not match after it has been updated at some places.

For example: Suppose the customer requests to change the address for his account in the Bank and the Program is executed to update the saving bank account file only but his current bank account file is not updated. Afterwards the addresses of the same customer present in saving bank account file and current bank account file will not match. Moreover there will be no way to find out which address is latest out of these two.

3. Difficulty in Accessing Data

- Suppose that one of the bank officers needs to find out the names of all customers who live within a particular postal-code area. The officer asks the data-processing department to generate such a list. Because the designers of the original system did not anticipate this request, there is no application program on hand to meet it. There is, however, an application program to generate the list of *all customers*. The bank officer has now **two choices**:
  - either obtain the list of all customers and extract the needed information manually **or**
  - ask a system programmer to write the necessary application program.
- Both alternatives are obviously unsatisfactory.
- Suppose that such a program is written, and that, several days later, the same officer needs to trim that list to include only those customers who have an account balance of $10,000 or more. As expected, a program to generate such a list does not exist. Again, the officer has the preceding two options, neither of which is satisfactory.

4. Data Isolation

- Since the data files are created at different times and by different people, the structures of different files generally will not match. The data will be scattered in different files for a particular entity.

For example: Suppose the Address in **Saving Account file have fields**: Add line1, Add line2, City, State, Pin while the fields in address of **Current account** are: House No., Street No., Locality, City, State, Pin. Administrator is asked to provide the list of customers living in a particular locality. Providing a list of all the customers will require looking in both files. But they both have different way of storing the address. Writing a program to generate such a list will be difficult.

5. Integrity Problems

- **Data integrity is the overall accuracy, completeness, and consistency of data.**
- All the consistency constraints have to be applied to database through appropriate checks in the coded programs. This is very difficult when number of such constraint is very large.

For example:

- ✓ An account should not have balance less than Rs. 500.
- ✓ To enforce this constraint appropriate check should be added in the program which add a record and the program which withdraw from an account.
- ✓ Suppose later on this amount limit is increased then all those check should be updated to avoid inconsistency.
- ✓ These time to time changes in the programs will be great headache for the administrator.

6. Atomicity

- It is difficult to ensure atomicity in file processing system.
  - Atomicity means: Execute all the operations of a transaction or none of them. (All OR None)

- **For example:** transferring $100 from Account A to account B. If a failure occurs during execution there could be situation like $100 is deducted from Account A and not credited in Account B.

7. Concurrent Access anomalies

- If multiple users are updating the same data simultaneously it will result in inconsistent data state.
- In file processing system it is very difficult to handle this using program code.
- This results in concurrent access anomalies.

8. Security Problems

- Enforcing Security Constraints in file processing system is very difficult as the application programs are added to the system in an ad-hoc manner.

- For example:
  - The Payroll Personnel in a bank should not be allowed to access accounts information of the customers.
  - Student should not allowed to access information about administration data.

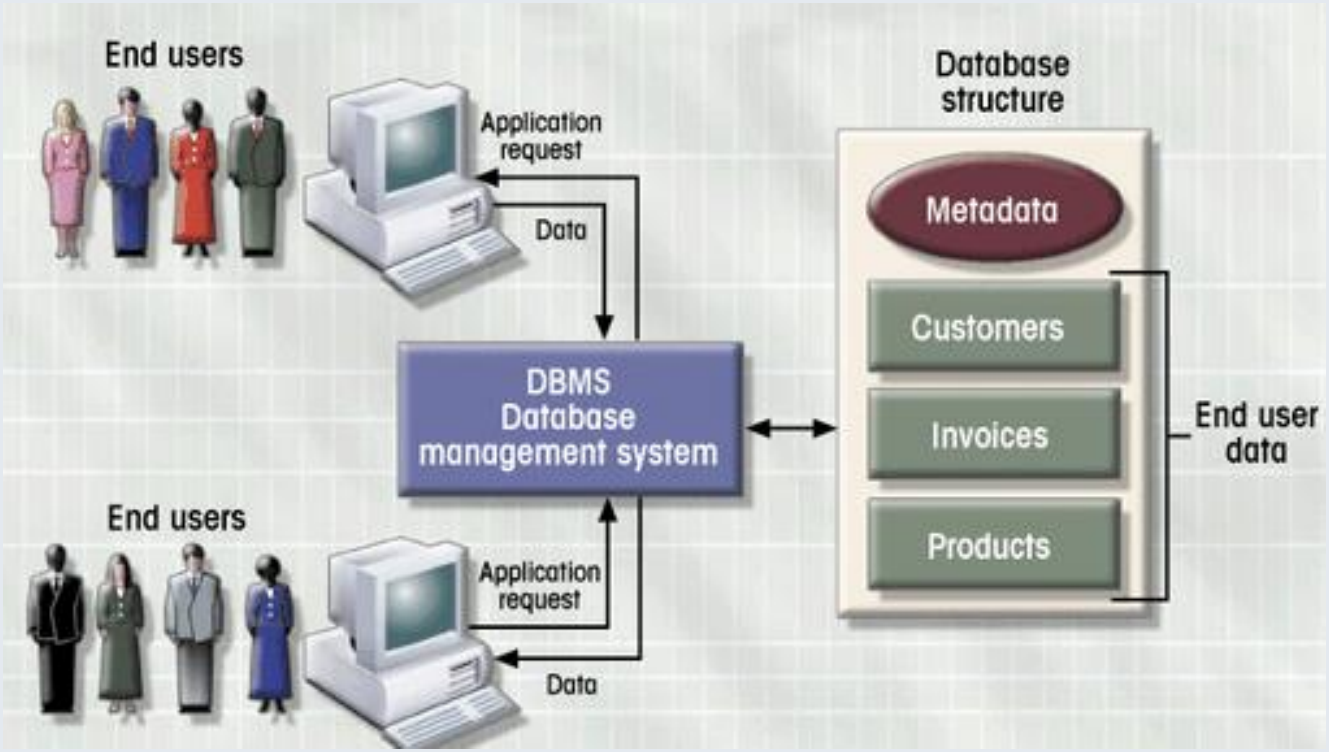
Database Management System

- **Database** is a collection of ***related data*** and **data** is a collection of facts and figures that can be processed to produce information.
  - If we have data about marks obtained by all students
    - We can then conclude about toppers and average marks
    - We can calculate percentage marks obtained by a student
    - We can print marks sheets
- A **database management system(DBMS)** is a set of programs that is used to access, modify and extract information from a database
- **The DBMS** stores data in such a way that it becomes easier to retrieve, manipulate, and produce information.
- The general purpose of a DBMS is to provide for the ***definition, storage, and management of data*** that can be shared by many users.

Database Management System

- The related information when placed is an organized form makes a database.
- Database is a collection of information organized in such a way that a computer program can quickly select desired pieces of data.
- The organization of data/information is necessary because unorganized information has no meaning

DBMS Manages Interaction



Operations on Databases

- To **add new information**
- To **view or retrieve** the stored information
- To **modify or edit** the existing information
- To **remove or delete** the unwanted information
- **Arranging the information** in a desired order
- etc.

Components of the DBMS Environment

- Hardware
- Software
- Data
- Users
- Procedures

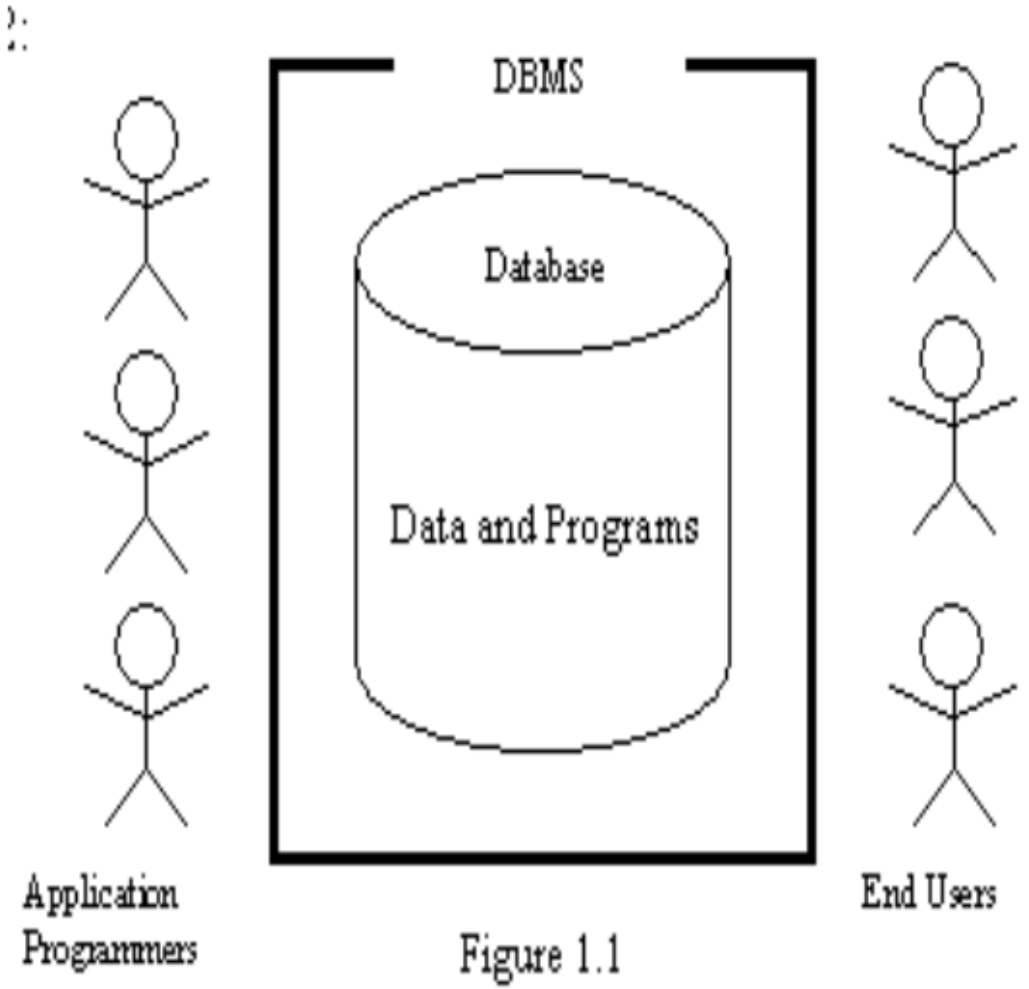
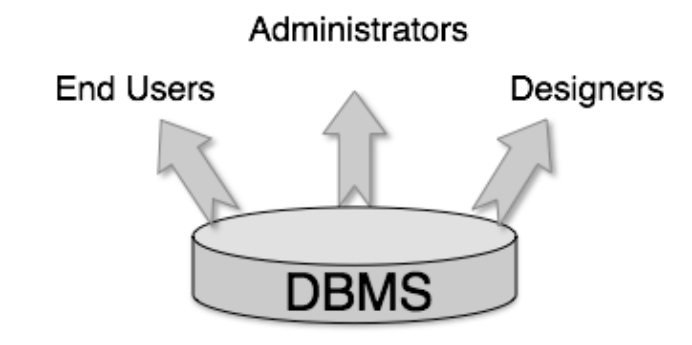


Figure 1.1

Users of DBMS



- **Administrators:**
  - Administrators maintain the DBMS and are responsible for administrating the database.
  - They are responsible to look after its usage and by whom it should be used.
  - They create access profiles for users and apply limitations to maintain isolation and force security.

- **Database Designers:**
  - Designers are the group of people who actually work on the designing part of the database.
  - They keep a close watch on what data should be kept and in what format.
  - They identify and design the whole set of entities, relations, constraints, and views.

- **End Users:**
  - End users are those who actually takes the benefits of having a DBMS.
  - The End user can be of following types:
    - **On-line user**: The user can directly access the database through online-terminal or user interface. They know the presence of the database and can manipulate database using SQL language.
    - **Naïve User**: The users who does not have deep knowledge about the database. These users uses the database through menu oriented application programs.
    - **Programmer**: The user of database who develop the application programs for the On-line users and Naïve users.

Use of DBMS

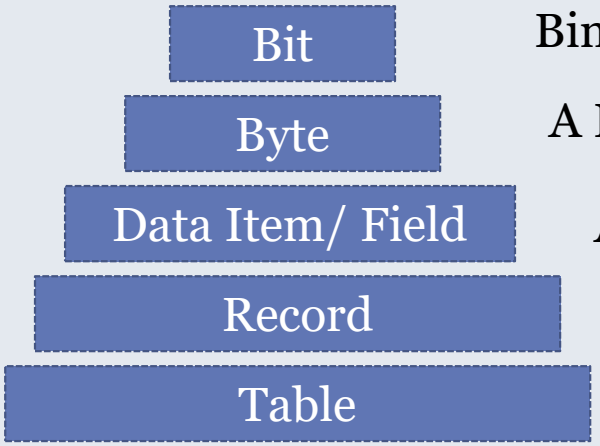
Databases are widely used. Here are some applications of database:

- **University database**: To store information details about Students, Faculties, Courses etc.
- **Banking**: To store customer information, accounts, loans, transactions etc.
- **Credit card transaction:** For purchases on credit card and generation of monthly bill statements
- **Airline**: For reservation and schedule information.
  - Airlines were first to use databases.
  - The terminals situated around the world accessed the central database through phone lines and data networks.

Use of DBMS

- **Library:** To store information about books, journals etc.
- **Telecommunication**: For keeping records of calls/sms made, generating monthly bills, maintaining balance for prepaid calling, storing information about communication networks
- **Human Resources**: For information about employees, salaries, benefits etc.
- **Hospital:** To store information about patients, doctors, staffs, department, equipments, etc
- **Pharmacy:** To store information about medicines.
- **E-commerce:** For information about products, customers, Offers etc.

Arrangement of Data in Database



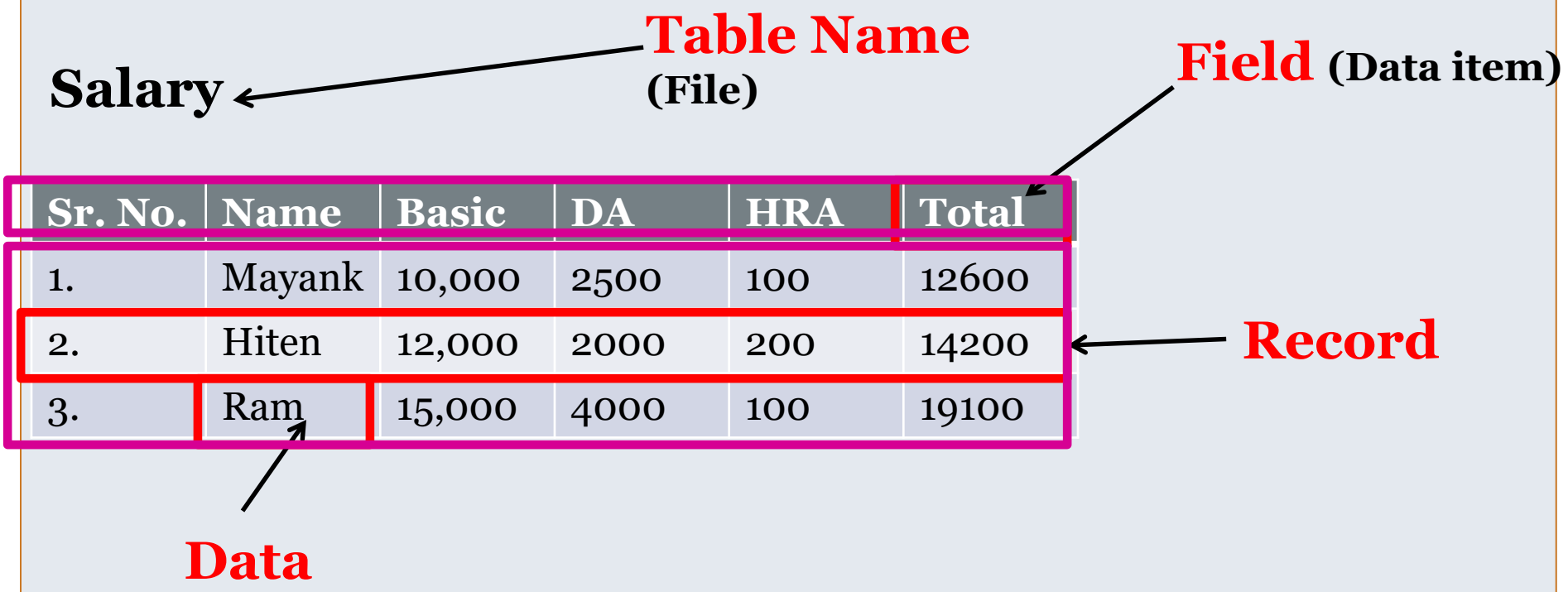
Binary digit 0 or 1

A Logical collection of bits

A Logical collection of bytes

Collection of relevant fields

Collection of Similar Records



Sr. No.	Name	Basic	DA	HRA	Total
1.	Mayank	10,000	2500	100	12600
2.	Hiten	12,000	2000	200	14200
3.	Ram	15,000	4000	100	19100

Advantages of DBMS

- **Controlling Redundancy**

- **Integrity can be enforced**

Integrity of data means that data in database is always accurate, such that incorrect information cannot be stored in database.

- **Inconsistency can be avoided**

When the same data is duplicated and changes are made at one site, which is not propagated to the other site, it gives rise to inconsistency and the two entries regarding the same data will not agree.

Advantages of DBMS

- Data can be shared
- Providing Backup and Recovery
- Standards can be enforced
- Restricting unauthorized access
- Solving enterprise requirement than individual requirement

Disadvantages of DBMS

- Complexity
- Size
- Performance
- Higher impact of a failure
- Cost of DBMS
- Additional Hardware costs
- Cost of Conversion

DBMS Architecture

- The design of a DBMS depends on its architecture.
- It can be centralized or decentralized or hierarchical.
- The architecture of a DBMS can be seen as either single tier or multi-tier.
- An **n-tier** architecture divides the whole system into related but independent **n modules**, which can be independently modified, altered, changed, or replaced.

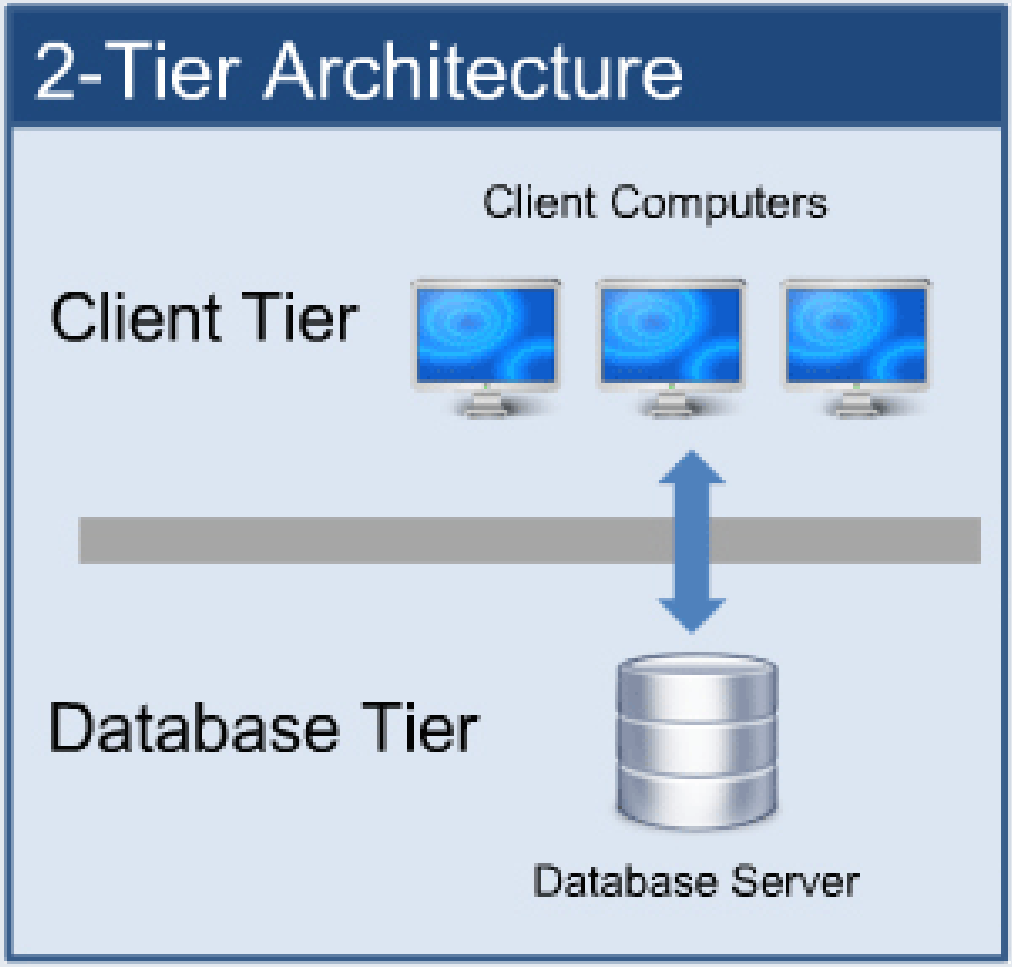
1-tier architecture

- In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it.
- Any changes done here will directly be done on the DBMS itself.
- Database designers of the database normally prefer to use single-tier architecture.
- 1-tier architecture is simple and cheap, but usually unsecured and data can easily be lost if you are not careful.

2-tier architecture

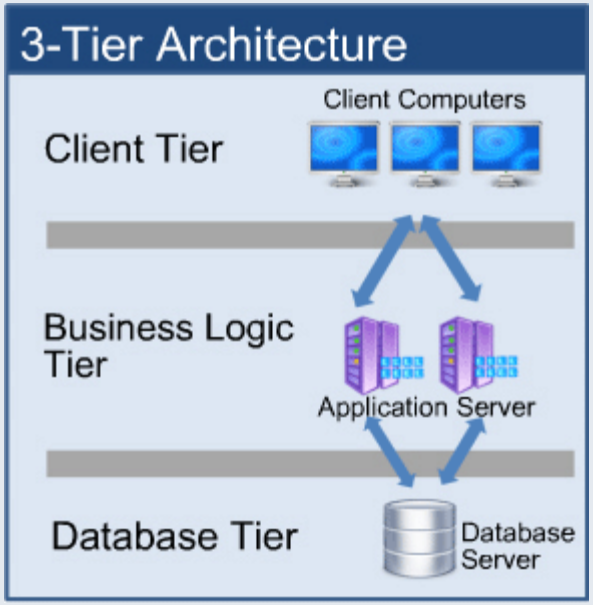
- This architecture is also called **Client-Server architecture** because of the two components: The client that runs the application and the server that handles the database back-end.
- If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed.
- Programmers use 2-tier architecture where they access the DBMS by means of an application.
- Here the application tier is entirely independent of the database in terms of operation, design, and programming.

2-Tier Architecture



3-tier architecture

- A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database.
- It is the most widely used architecture to design a DBMS.



- **Database (Data) Tier:**
  - At this tier, the database resides along with its query processing languages.
  - We also have the relations that define the data and their constraints at this level.
  - The database tier is not aware of any other user beyond the application tier.

- **Application (Middle) Tier/Business Logic Tier:**
  - At this tier reside the application server and the programs that access the database.
  - For a user, this application tier presents an abstracted view of the database.
  - The application layer sits in the middle and acts as a mediator between the end-user and the database.

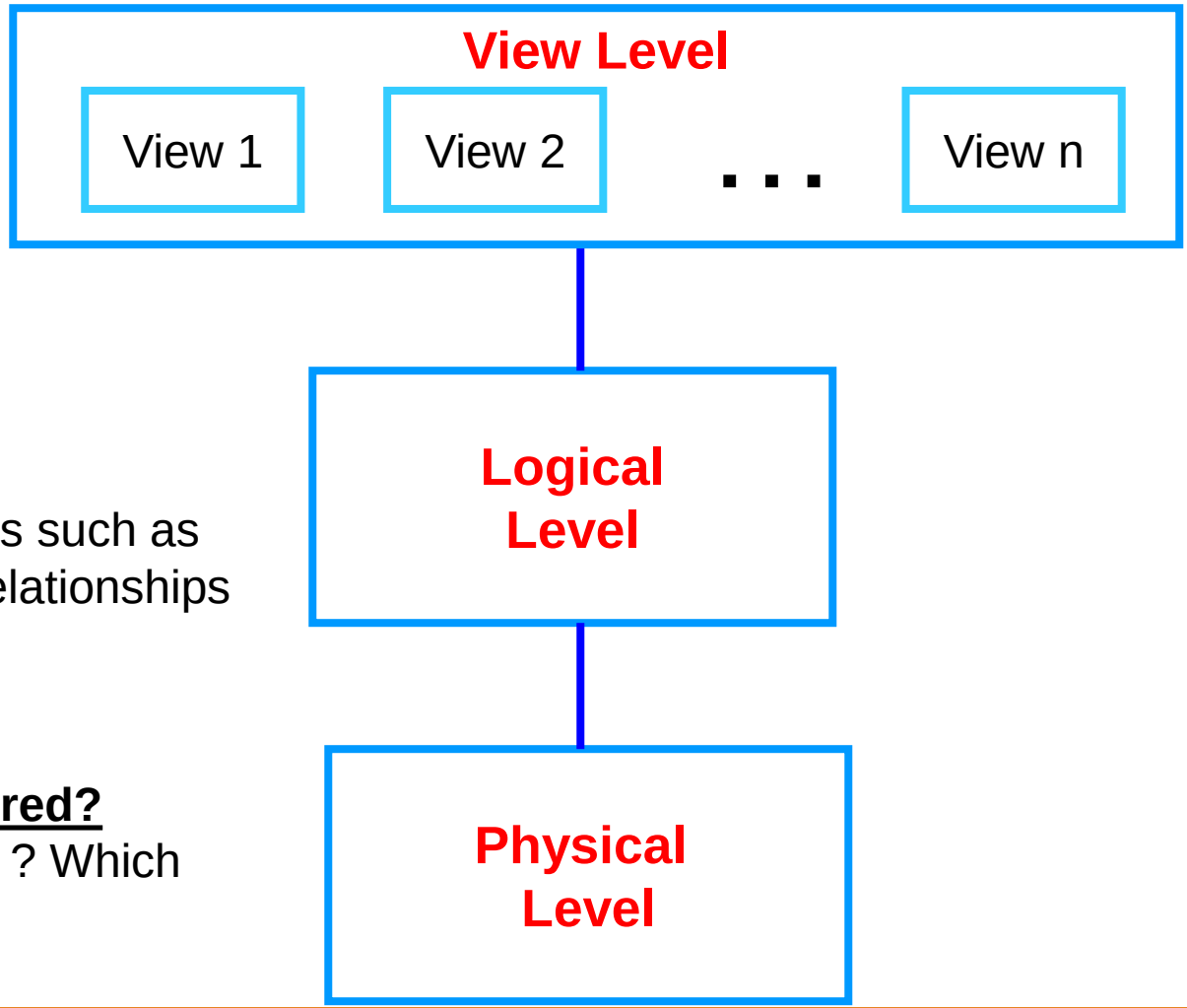
- **User (Presentation) Tier/ Client Tire:**
  - End-users operate on this tier and they know nothing about any existence of the database beyond this layer.
  - At this layer, multiple views of the database can be provided by the application.
  - All views are generated by applications that reside in the application tier.

Data Abstraction

- A database system is the collection of interrelated data and set of programs that allows users to access and modify these data.
- The database system can be meaningful only if it provide data retrieval efficiently.
- To make database efficient designer has to used complex data structure to maintain the records/files in the database.
- Since not all the database users are trained, developers hides the complexity from the users.
- That is , the system hides certain details of how the data are stored and maintained.

Three Levels of Abstraction

What data users and application programs see ?

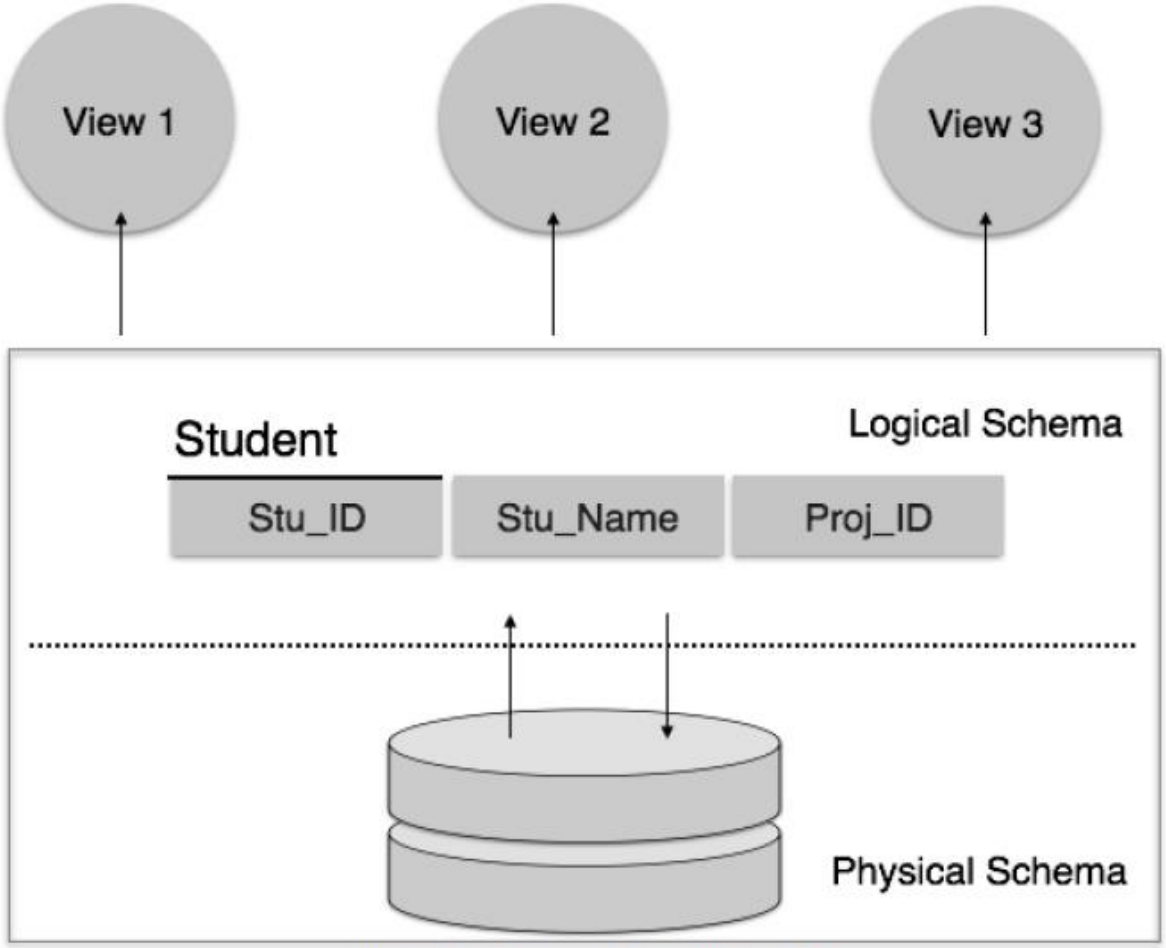


What **data is stored?**

describe data properties such as data semantics, data relationships

How **data is actually stored?**

e.g. are we using disks ? Which file system ?



[*Image: Database Schemas*]

Physical Level

- It is the lowest level of Abstraction.
- Describes how the data are actually stored.
- Deals with the Data Structure Details.

Logical Level

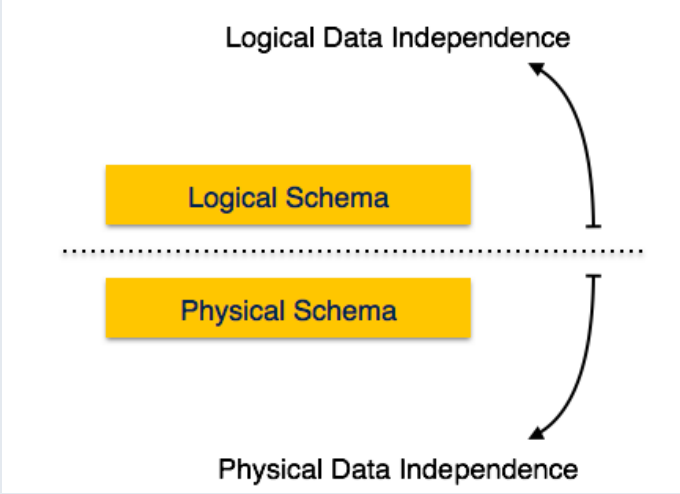
- This level describes **what** data stored in database and what relationship exist among those data.
  - ER-Diagram(Schema of database)
- The user of the logical level does not need to be aware of the complexity of the physical level.

View Level

- The highest level of abstraction.
- Describes what data users and application programs should see.
- Many user (Naïve users, online users, programmers) does not need all the information about details of the logical level. They only need their parts of the database.
- The view level simplify the users interactions with the database system.

Data Independence

- If a database system is not multi-layered, then it becomes difficult to make any changes in the database system.
- Database systems are designed in multi-layers(View, Logical, Physical).



Logical Data Independence

- Logical data is data about database, that is, it stores information about how data is managed inside.
- For example, **a table** (relation) stored in the database and all its constraints, applied on that relation.

- Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk.
- If we do some changes on table format, it should not change the data residing on the disk.

Physical Data Independence

- All the schemas are logical, and the actual data is stored in bit format on the disk.
- Physical data independence is the power to change the physical data without impacting the schema or logical data.
  - i.e the ability to modify the physical schema without changing the logical schema

- For example, in case we want to change or upgrade the storage system itself – suppose we want to replace hard-disks with SSD – it should not have any impact on the logical data or schemas.
  - (Solid State Drive (SSD), Hard Disk Drive (HDD))

  - 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.

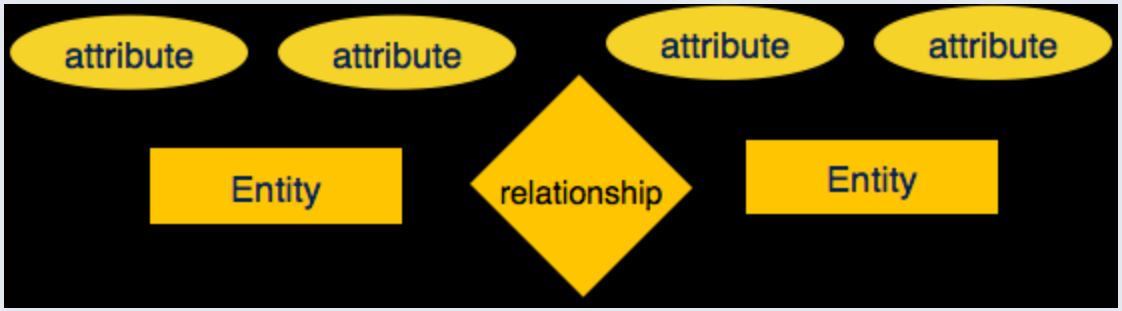
Data Model

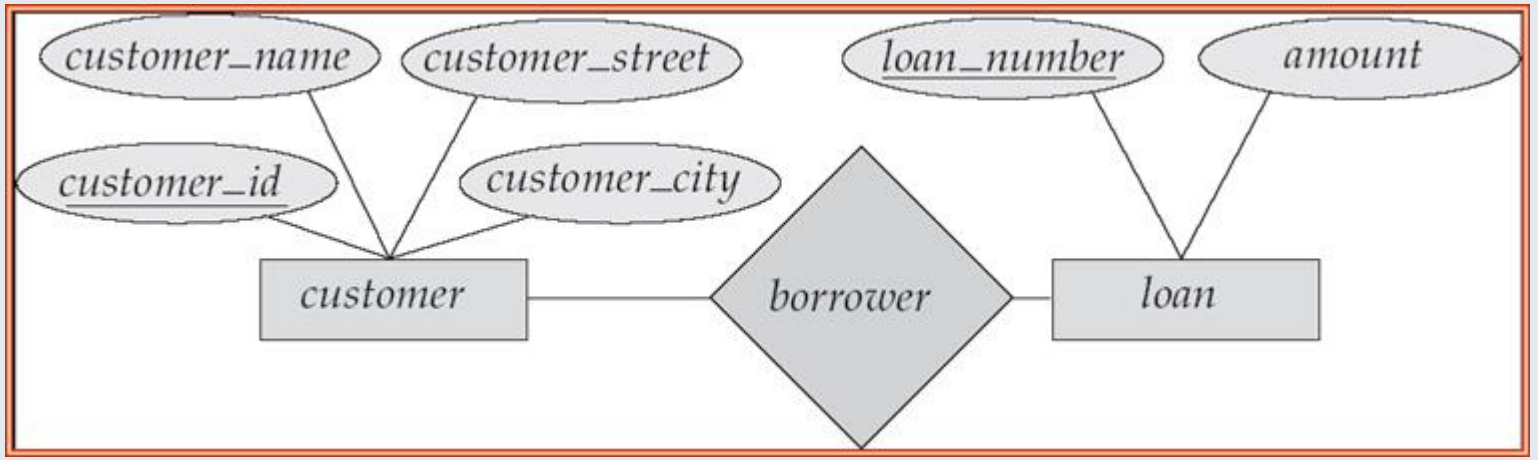
- Data models define how the logical structure of a database is modeled.
- Data models define how data is connected to each other and how they are processed and stored inside the system.
- There are two types of data models:
  - **Entity –Relationship Model**
  - **Relational Model**

  Both provides a way to describe the design of a database at the logical level

Entity-Relationship(ER) Model

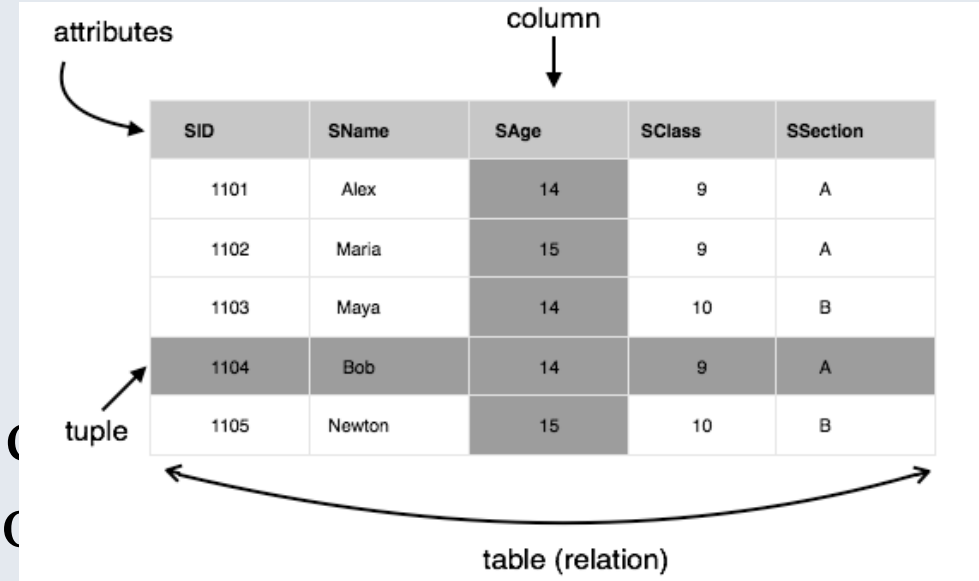
- Entity-Relationship (ER) Model is based on perceptions of real-world entities and relationships among them.
- While formulating real-world scenario into the database model, the ER Model creates
  - entity set,
  - relationship set,
  - general attributes and
  - constraints.
- ER Model is best used for the conceptual design of a database.
- ER Model is based on –
  - Entities and their attributes.
  - Relationships among entities.


attribute
attribute
attribute
attribute
Entity
relationship
Entity

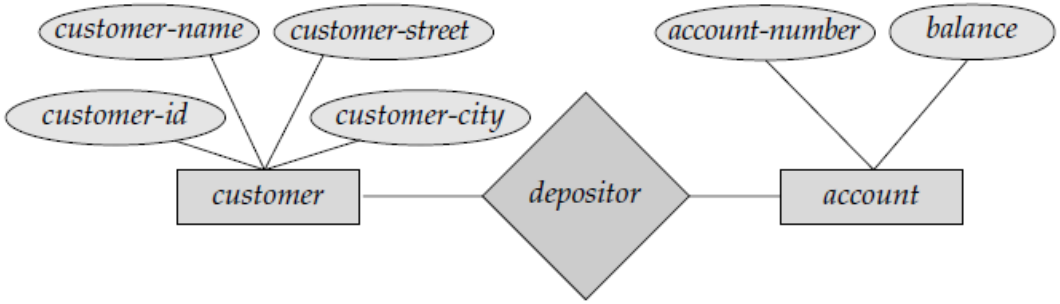

customer_name
customer_street
loan_number
amount
customer_id
customer_city
customer
borrower
loan

Relationship Model

- Relational model uses a collection of tables to represent both data and relationships among those data.
- Each table has multiple columns and each column has unique name.



- The database design is c
translated to the Relatic



E-R diagram

customer-id	customer-name	customer-street	customer-city
192-83-7465	Johnson	12 Alma St.	Palo Alto
019-28-3746	Smith	4 North St.	Rye
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The *customer* table

account-number	balance
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

(b) The *account* table

customer-id	account-number
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

Relational database