Chapter 1: Introduction

Data:

By **Data** we means known facts that can be recorded and that have some implicit meaning for a specific group of people.

Information:

When data have some explicit meaning, it is called **information**.

Example: Let's consider about a world library database system, where the central library of every country store data. Mean India, China, England every one store their library information in this database. But Indian cannot understand the meaning of those data which are stored by other country cause of language problem. So all the data except Indian data is only data not information in respect of India. But the data stored by India is data as well as information in front of Indian.

Knowledge:

The term **knowledge** indicates the information which is not only utilized in current decision making and planning but also store for later use. We go through this information to acquire knowledge and then applied the knowledge in drawing, calculation or in making some decision. All information is not requiring later and hence they should not be considered as knowledge.

Database:

The **Database** is an organized collection of related information, presented to serve a specific purpose.

Example: Student Database, Library Database, Hospital Database etc.

DBMS (Data Base Management System)

DBMS is the collection of program that enable user to create and maintain database conveniently and efficiently.

Example: Oracle, DB2, Microsoft Access.

Different between Data Processing and Data Management

The procedure used for converting data into information is called data processing, i.e. **data processing** is an activity of the refinement.

Data Management, on the other hand, is the function of controlling the acquisition to acquire something specially analysis storage, retrieval and distribution of data as well as security, validation, garbage collection etc are the basic task of the data management.

Disadvantage of Traditional File Processing System or Advantage of the Database Management System

1. Data Redundancy and Inconsistency:

Since different programmer create the files and application program over a long period various files likely to have different format and program may be written in several programming languages. Mo0re over some information may be duplicated in several files which leads to higher storages cost as well as inconsistency.

2. Difficulty of accessing data:

Conventional file processing environment does not allow require data to be retrieved inconvenient and efficient manner.

3. Data Isolation:

Because data are scattered in various files and files may be in different format, writing new application program to retrieve the appropriate data is difficult.

4. Integrity Problem:

Data values stores in database certain time of constraints for consistency when more constraints are require to add is more difficult to change, the program are to enforce them, The problem is compounded, constraint involved, several data items from different files.

5. Atomicity Problem:

A computer system like mechanical or electrical devices is subject to failure in many application, it is crucial that, if a failure occur, that data be restored to the consistent state that existed prior to the failure but this kind of atomicity cannot be assure for a traditional file processing system.

6. Concurrent access anomalies (abnormality):

For the seek of overall performances of the system and faster responses many system allows multiple user to update data simultaneously such an environment, interaction of concurrent updates many result in inconsistent data. The conventional file processing system is unable to avoid such anomalies.

7. Security Problem

Security Constraints as many application programs are acting upon then it is harder maintain for a traditional file processing.

Database Instance and Database Schema

The collection of information stored in the database at a particular point of time is termed as a **Database Instance**.

On the other hand the overall design of database is termed as **Database Schema**

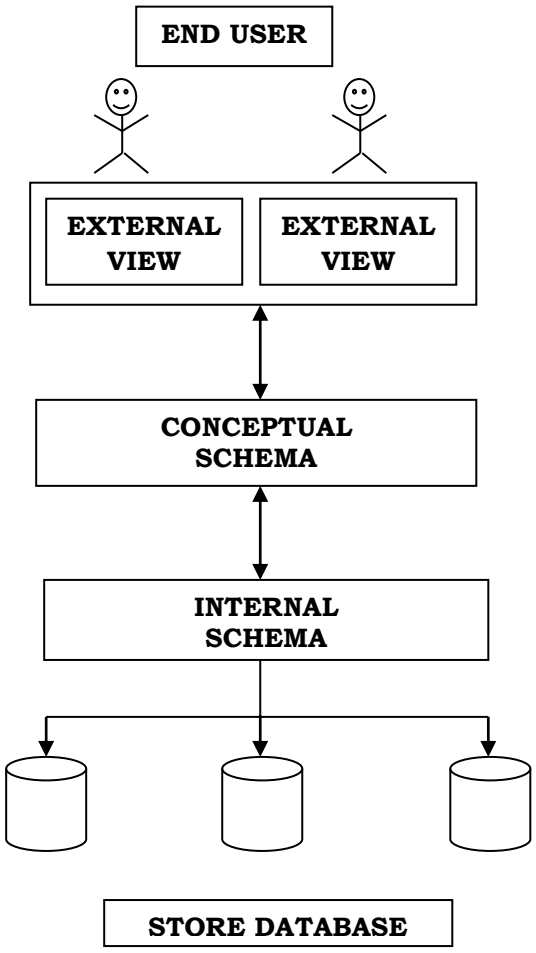
ANSI / SPARC: 3 Schema Architecture:

The goal of 3 schema architecture is to do separate the user application and the physical database, in this architecture; schema can be dividing at the following 3 level

1) The ***Internal Level*** has an internal schema which described the physical storages structure of the database. The internal schema uses a physical data model and describes the complete details of data storages and the access path of the database.

2) The ***Conceptual Level*** has a conceptual schema, which described the hold database structure for a community of users. The conceptual schema hides the details of physical storages structure concentrate on describing entities, data types, relationship, user operations and constraints. A high level data model or an implementation data model can be used at this level.

3) The ***External or View Level*** includes a numbers of external schema or user views. Each external schema describes the parts of a database that particular user group is interested in and hides the rest of the database from the groups. A high level data model or an implementation data model can be used at this level.



The above mentioned 3 schema are only description of data, which exist at the physical level in actual senses. In a DBMS based on this architecture, each user groups referred only to it's owns external schema. Hence DBMS must transform a request specified on an external schema / view in to a request against a conceptual schema and then into a request against an internal schema, for processing over the stored data. If the request is retrieved from the database, the data extracted from the stored database must be reformatted to match the user external view. This process of transformation of request and request between two levels called ***Mapping.***

Data Independency

Data independency is the capacity to change the schema at one level of database system without having to change the schema at the next higher level; we can define two types of data independencies.

1) ***Logical data Independency*** is the capacity to change the conceptual schema without having changed the external schema of application program only the view definition and the mapping need to be changed in a DBMS that supports logical data independence. Application programs, that referrer to external schema constructs, much work as before, after the conceptual schema undergoes a logical reorganizations changes to constraints can be applied also to the conceptual schema without effecting the external schema or application program.

2) ***Physical Data Independency*** is the capacity to changes internal schema without having to change the conceptual or external schema. Changes to the internal schema may be needed because some physical files have to be reorganized to improve the performances of retrieval or update.

The 3 schema architecture can make it easier to achieve through data independency, both physical and logical. However the two level of mapping create and overhead during compilation or execution of a quires or program, leading to inefficiencies in DBMS. Because of this few DBMS have implement the full 3 schema architecture.

Data Administrator Roles

In order to achieve the objective of a database it is obvious that in an enterprise with a database system there should be some identifiable person who will bear the central responsibility for the operational data. The personal assigned with this administrative person is called a ***Database Administrator (DBA)***

The function of DBA called ***Role of Database Administrator***. DBA is not the person who process the database but the protector of data.

Data base Administrator roles are follows.

1) ***Describing the information of the database:*** It is the DBS's job to decide actually what information to be held in the database that is to identify the entities of interest to the enterprise and the identified the information to be recoded about those entities.

2) ***Deciding storage structure and access strategy:*** The DBA must decides have data is to be represented in the database and must specify the representation by which the storage structure has been defined.

3) ***Defining authority checks and validation:*** Authority checks and validation procedures may be consider as logical extension of the conceptual schema. The conceptual DDL (Data Definition Language) will include such factors to satisfy such checks and procedures.

4) ***Defining a strategy for backup and recovery:*** In the event of the damage of any portion of the database by any cause, it should be recoverable with a maximum time delay. DBA must define as well as implement such a appropriate recovery strategy.

5) ***Monitoring performance and responding to changes in requirement:*** The DBA is responsible for organizing the system has to get enterprise and making the appropriate adjustment as requirement changes.

Data Dictionary

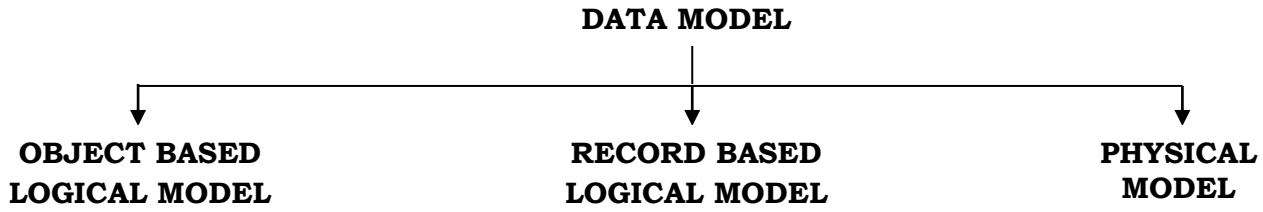
A ***data dictionary*** is a database which certain ***dada about data (Meta Data)*** i.e. description of other objects within the system rather than sampling the raw data. A comprehensive dictionary will also exclude cross reference information showing which pieces of program are using which piece of data.

It should be possible to query the dictionary just like any other database, so that the DBA can discover easily which programs are likely to be effected by same proposed change in the system.

The benefit of using a data dictionary is related to the effective collection, specification and the management of total database resources of an enterprise. A data dictionary should help a database user in

1) Communicating with other user.
2) Controlling the data element in a simple and effective member.
3) Determine the impact of changes of data element on a database.
4) Centralizing the control of a data element as an aid in the database design and in extending the design.

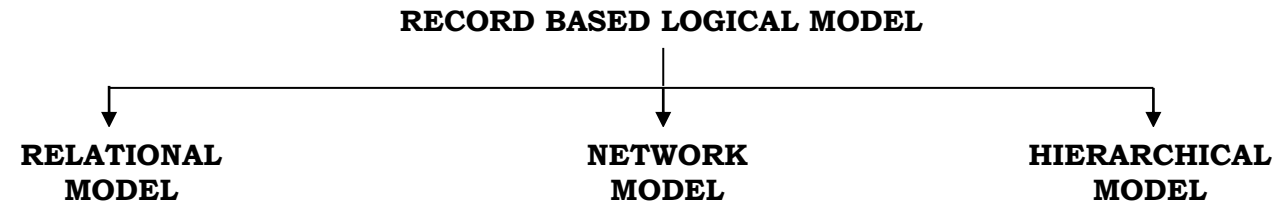
Data Models



Underlying structure of a database is the ***Data Model***.

Record Base Logical Model

Record base logical model are used in describing data in logical and view level.



Relational Model

Relational model uses a collection of tables to represent both data and the relationship among them. Each table has multiple columns and each column has a name.

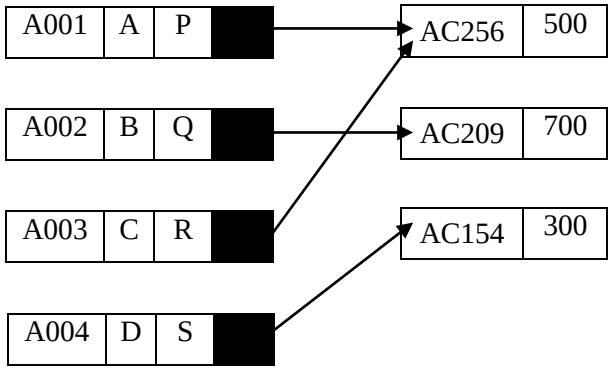
CUSTOMER		
ID	**NAME**	**CITY**
A001	A	P
A002	B	Q
A003	C	R
A004	D	S

ACCOUNT	
ACC NO	**BALANCE**
AC256	500
AC209	700
AC154	300

DIPOSITER	
ID	**ACC NO**
A001	AC256
A002	AC209
A003	AC256
A004	AC154

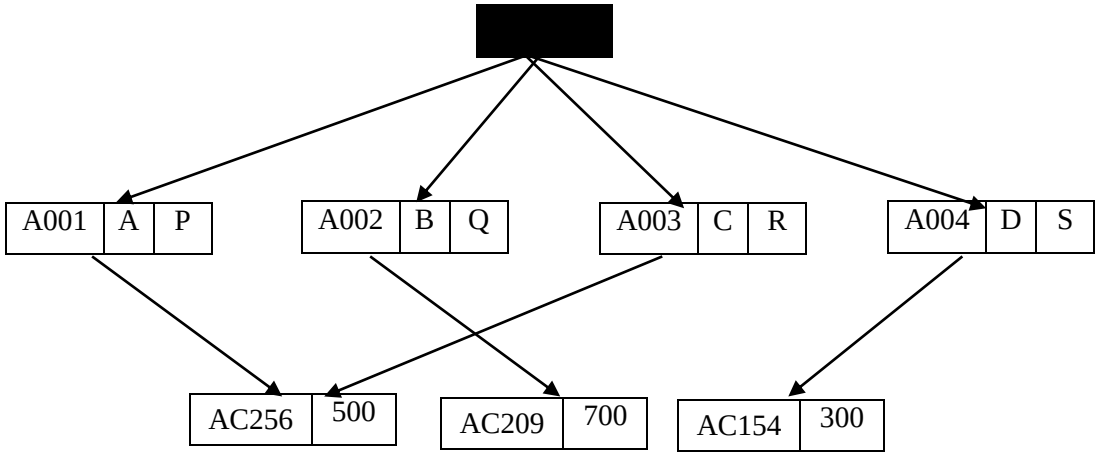
Network Model

Data in ***Network Model*** are representing by collection of records and relationship among data is represented by links which can be viewed as pointed. The record in the database is organized as collections of arbitrary graphs.



Hierarchical Model

Like Network Model, in data is represented by collection of records and relationship among data are represented by links but records are organized as tree rather than arbitrary graph.



Compares of 3 Models

Sl. No	Relational Model	Hierarchical Model	Network Model
1	Database consist of set of sample tables	It consist of several types of records in a tree like structure along with link connecting them	Similar as hierarchical but less complex.
2	Can have 1 to 1 or 1 to many relationships.	1 to many relationship	Supports all kind of relationship as in relational model
3	Queries are symmetric in nature	Asymmetric as it maintains superiors and dependence relationship with tree structure.	More symmetric with than hierarchical model but less symmetric than relational model
4	No anomalies arises during updating, insertion or deletion	Anomalies Occurs	Does not Occurs
5	Data Structure and associated DML are very simple	Most Complex	Less Complex
6	Flexible structure to changing data and access requirement	Rigid structure	Limited flexible
7	Access to record types is achieved by using relational algebra or relational calculus statement	It is achieved by navigating from the root record types to record type lower down in hierarchical in a preorder traversal.	It is achieved by navigating through the structure and links through pointers.