UNIT 1: INTRODUCTION

Q.1 Data, Information and Knowledge

Data are plain facts. The word "data" is plural for "datum." When data are processed, organized, structured or presented in a given context so as to make them useful, they are called **Information**. It is not enough to have data (such as statistics on the economy). Data themselves are fairly useless, but when these data are interpreted and processed to determine its true meaning, they becomes useful and can be named as Information.

DATA	INFORMATION
• Raw Facts and figures • It is in unorganized form • Data doesn't help in decision making process • Example: 460	• Processed Data • It is in organised form • Information helps in decision making process • Example Total Marks 460

Knowledge: Knowledge refers to the practical use of information. Knowledge necessarily involves a personal experience.

Q. 2 FILE ORIENTED APPROACH:

The conventional approach for data processing is to store locally needed data and develop a program(s) for each type of application. In the past, different users had programs that handled their own independent stored data. This resulted in more than one data files with same data (Redundancy) for different applications. Maintenance for information in separate files for each business application has the following disadvantages.

- Each file requires separate processing program
- The practice of preparing two or more copies of each file leads to wastage of resources, efforts and time.
- Each department needs to keep its files updated separately.

Disadvantages of File Oriented Approach

- **Data Redundancy:** Unnecessary repetition/duplication of same data in different files.
- **Data Inconsistency:** Data Inconsistency occurs when similar data is kept in different formats in more than one file (data redundancy) and then it is important to match the data between files. Sometimes, during maintenance/updating, the changes are not reflected in every files leading to different copies of same data.
- **Difficulty in Accessing Data:** The conventional file processing environments do not allow needed data to be retrieved in a convenient and efficient manner.
- **Data Isolation:** Since data is scattered in various files, and files may be in different formats, it is difficult to write new application programs to retrieve the appropriate data.
- **Concurrent Access Anomalies:** In order to improve the overall performance of the system and obtain a faster response time, many systems allow multiple users to update

the data simultaneously. In such an environment, interaction of concurrent updates may result in inconsistent data.

- **Security Problems:** Not every user of the database system should be able to access all the data. For example, in banking system, payroll personnel need only that part of the database that has information about various bank employees. They do not need access to information about customer accounts. It is difficult to enforce such security constraints.
- **Integrity Problems:** The data values stored in the database must satisfy certain types of consistency constraints. For example, the balance of a bank account may never fall below a prescribed amount. These constraints are enforced in the system by adding appropriate code in the various application programs. When new constraints are added, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items for different files.
- **Does not** support **multiple views** of the same data

Q. 3 DATABASE: A database is a collection of information that is organized so that it can easily be accessed, managed, and updated. In other words we can say that it is a collection of information arranged and presented to serve an assigned purpose.

Few Examples / Applications of Database Systems: Few applications/organizations that use databases for their business processing activities are:

- **Banking:** For customer information, accounts, and loans, and banking transactions.
- **Airlines:** For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner—terminals situated around the world accessed the central database system through phone lines and other data networks.
- **Universities:** For student information, course registrations, and grades.
- **Credit Card Transactions:** For purchases on credit cards and generation of monthly statements.
- **Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
- **Finance:** For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds.
- **Sales:** For customer, product, and purchase information.
- **Railway Reservation Systems:** For reservations and schedule information.
- **Web:** For access the Back accounts and to get the balance amount.
- **E –Commerce:** For Buying a book or music CD and browse for things like watches, mobiles from the Internet.

Q. 4 CHARACTERISTICS OF DATABASE APPROACH: Some of the features of database approach are discussed below

- **Structured and Described Data:** Fundamental feature of the database approach is that the database system does not only contain the data but also the complete definition and description of these data. These descriptions are basically details about the extent, the structure, the type and the format of all data and, additionally, the relationship between the data. This kind of stored data is called **metadata** ("data about data").
- **Separation of Data and Applications:** Application software does not need any knowledge about the physical data storage like encoding, format, storage place, etc. It only communicates with the management system of a database (DBMS) via a standardized interface with the help of a standardized language like SQL. The access to the data and the metadata is entirely done by the DBMS. In this way all the applications can be totally separated from the data.
- **Data Integrity:** Data integrity is a byword for the quality and the reliability of the data of a database system. In a broader sense data integrity includes also the protection of the database from unauthorized access (confidentiality) and unauthorized changes.
- **Transactions:** A transaction is a bundle of actions which are done within a database to bring it from one consistent state to a new consistent state. In between the data are inevitable inconsistent. *A* transaction is atomic what means that it cannot be divided up any further. Within a transaction all or none of the actions need to be carried out. Doing only a part of the actions would lead to an inconsistent database state.
  Example: One example of a transaction is the transfer of an amount of money from one bank account to another.
- **Data Persistence:** Data persistence means that in a DBMS all data is maintained as long as it is not deleted explicitly. The life span of data needs to be determined directly or indirectly be the user and must not be dependent on system features. Additionally data once stored in a database must not be lost. Changes of a database which are done by a transaction are persistent. When a transaction is finished even a system crash cannot put the data in danger

Q. 5 DATABASE-MANAGEMENT SYSTEM: A DBMS is a collection of interrelated data and a set of programs to access those data. It is a general purpose software system that facilitates the process of defining**,** constructing and manipulating databases for various applications.

Goals of DBMS:

The primary goal of a DBMS is to provide a way to store and retrieve database information that is both *convenient* and *efficient*

- Manage large bodies of information
- Provide convenient and efficient ways to store and access information
- Secure information against system failure or tampering

- Permit data to be shared among multiple users

Properties of DBMS:

- A Database represents some aspect of the real world. Changes to the real world reflected in the database
- A Database is a logically coherent collection of data with some inherent meaning.
- A Database is designed and populated with data for a specific purpose.

Need of DBMS:

- Before the advent of DBMS, organizations typically stored information using a "File Processing Systems". Example of such systems is File Handling in High Level Languages like C, Basic and COBOL etc., these systems have Major disadvantages to perform the Data Manipulation. So to overcome those drawbacks now we are using the DBMS.
- Database systems are designed to manage large bodies of information.
- In addition to that the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access. If data are to be shared among several users, the system must avoid possible anomalous results.

Q. 6 DATABASE SYSTEM: The database and DBMS software together is called as Database system.

Q. 7 FILE PROCESSING SYSTEM V/S DATABASE SYSTEM

FILE PROCESSING SYSTEM	DATABASE SYSTEM
In a file processing system, the system stores permanent records in various files. Application programs are needed to extract records from, and add records to the appropriate files. As time passes new files and corresponding application programs are added to the system.	In the database systems, there exists a collection of interrelated files and a set of application programs to access and modify these files. Details of data storage and maintenance are hidden from the user.
Supported by convenient operating systems such as MS DOS	A database may be generated automatically or it may be computerized. The computerized database may be created and maintained by DBMS
Since the files and application programs are created by different programmers over a long period of time, data in the files is likely to get repeated (Redundancy) which can lead to inconsistency.	A database system so designed does not involve the problem of data redundancy or inconsistency.
No Automatic backup and recovery	Automatic Backup and recovery is provided.
In File Processing System the application	A Database approach is so designed

programs are added in an Adhoc manner, it is difficult to enforce such security constraints.	provides the security mechanism to prevent unauthorized access to certain parts or whole of the database.
Multiple user access to the information is difficult to provide through this approach.	Almost all the database systems allow multiple users to access/update the database at the same time.

Q.8 CHARACTERISTICS OF DATA IN A DATABASE

FIELD: The smallest piece of meaningful information in a file is called as data item or field. For Example, **Registration Number, Name, Parentage, Address, Contact Number, Marks** are all fields.

RECORD: The collection of related fields is called as record. For Example, **Registration Number, Name, Parentage, Address, Contact Number and Marks** of a particular student is called as record.

FILE: File is a collection of all related records. For Example Details of all the Students of Batch 2017 Computer Applications of Degree College Bandipora.

DATABASE: A database is a collection of related files arranged and stored in an efficient manner. For Example, Details of all the Students of Degree College Bandipora.

Figures below: Concept of Filed, Record and File

Fields/Data Items

Registration Number	Name	Parentage	Address	Contact No.	Marks

Record

101-BPB-2017	**Adil Hamid**	**Ab Hamid**	**Patushai**	**1234**	**465**

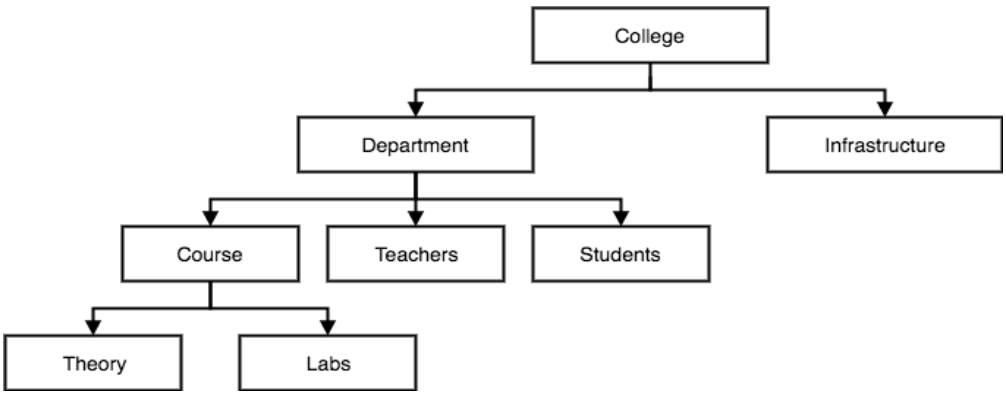
File

Registration Number	Name	Parentage	Address	Contact No.	Marks
101-BPB-2017	**Adil Hamid**	**Ab Hamid**	**Patushai**	**1234**	**465**
102-BPB-2017	**Zubair Ahmad**	**Ab Majeed**	**Bandipora**	**4569**	**438**
…..	**….**	**….**	**….**	**….**	**….**
171-BPB-2017	**Iqbal Bhat**	**Zubair Bhat**	**Onagam**	**1657**	**385**

Q.9 DATABASE MODEL: A database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in a Database management System. The various database models are discussed below and among them the Relational Model is most widely used database model.

Hierarchical Model: This database model organizes data into a tree-like-structure, with a single root, to which all the other data is linked. The hierarchy starts from the **Root** data, and expands like a tree, adding child nodes to the parent nodes. In this model, a child node will only have a single parent node. This model efficiently describes many real-world relationships like index of a book, recipes etc.

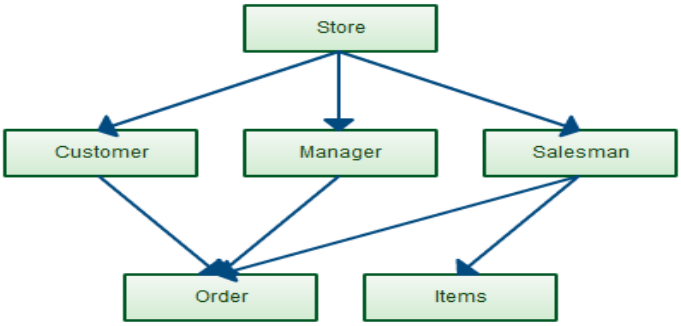
In hierarchical model, data is organised into tree-like structure with one one-to-many relationship between two different types of data, for example, one department can have many courses, many professors and of-course many students.



Network Model: This is an extension of the Hierarchical model. In this model data is organised more like a graph, and are allowed to have more than one parent node.

In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and fast. This database model was used to map many-to-many data relationships.

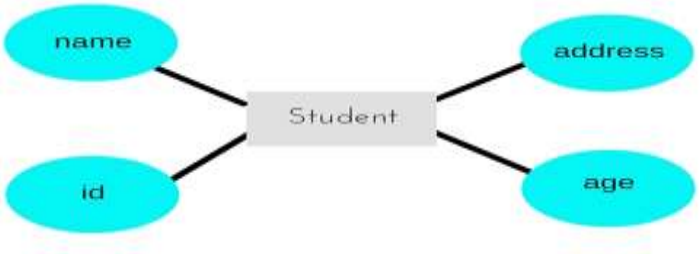
This was the most widely used database model, before Relational Model was introduced.



Entity-relationship Model: In this database model, relationships are created by dividing object of interest into entity and its characteristics into attributes. Different entities are related using relationships.

E-R Models are defined to represent the relationships into pictorial form to make it easier for different stakeholders to understand. This model is good to design a database, which can then be turned into tables in relational model.

Let's take an example, If we have to design a School Database, then **Student** will be an **entity** with **attributes** name, age, address etc. As **Address** is generally complex, it can be another **entity** with **attributes** street name, pincode, city etc, and there will be a relationship between them.



(NOTE: It will explained in detail in the upcoming days)

Relational Model: In this model, data is organised in two-dimensional **tables** and the relationship is maintained by storing a common field. This model was introduced by E.F Codd in 1970, and since then it has been the most widely used database model, infact, we can say the only database model used around the world. The basic structure of data in the relational model is tables. All the information related to a particular type is stored in rows of that table. Hence, tables are also known as **relations** in relational model.

Registration Number	Name	Parentage
101-BPB-2017	Adil Hamid	Ab Hamid
102-BPB-2017	Zubair Ahmad	Ab Majeed
171-BPB-2017	Iqbal Bhat	Zubair Bhat

Registration Number	Contact No.	Marks
101-BPB-2017	1234	465
102-BPB-2017	4569	438
171-BPB-2017	1657	385

Registration Number	Name	Marks
101-BPB-2017	Adil Hamid	465
102-BPB-2017	Zubair Ahmad	438
171-BPB-2017	Iqbal Bhat	385

The other database models are

- Object Oriented Database Model
- Distributed Database Model

Q.10 DBMS ARCHITECTURE

Database architecture focuses on the design, development, implementation and maintenance of computer programs that store and organize information for businesses, agencies and institutions. A database architect develops and implements software to meet the needs of users.

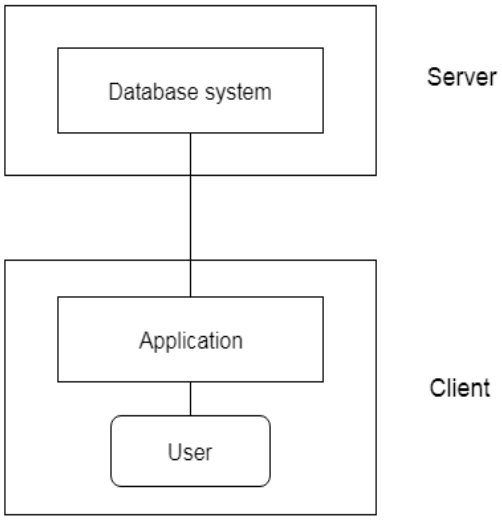
The design of a DBMS depends on its architecture. It can be **centralized** (all the data stored at one location), **decentralized** (multiple copies of database at different locations) or **hierarchical.** The architecture of a DBMS can be seen as either single tier or 2-tier or multi-tier.

1. **1-Tier Architecture:** In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it. Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.

The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

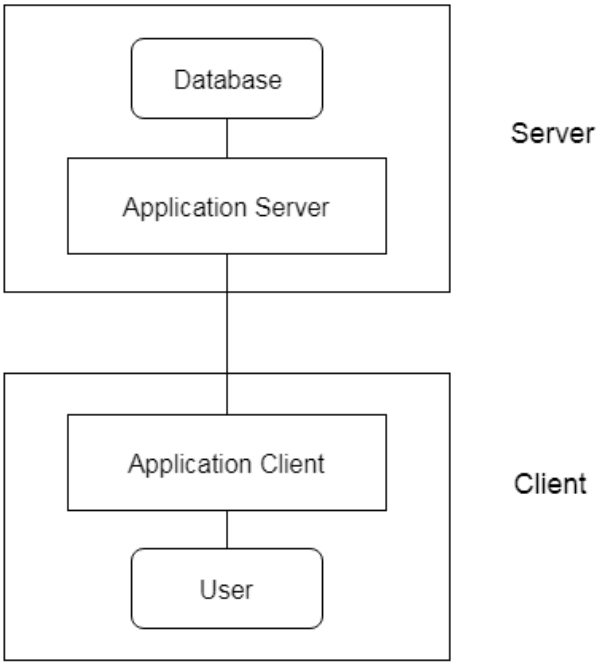
2. **2-Tier Architecture:** The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: **ODBC**, **JDBC** are used.

   The user interfaces and application programs are run on the client-side and the server side is responsible to provide the functionalities like: query processing and transaction management. To communicate with the DBMS, client-side application establishes a connection with the server side.



3. **3-Tier Architecture:** The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server. The application on the client-end interacts with an application server which further communicates with the database system. End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.

   The 3-Tier architecture is used in case of large web application.



If you have used **MySQL**, then you must have seen **PHPMyAdmin**, it is the best example of 3-tier DBMS architecture

Q.11 DEFINE SCHEMA AND SUBSCHEMA?

The term schema refers to an overall structure (organization) of all data items including their record types stored in a database.

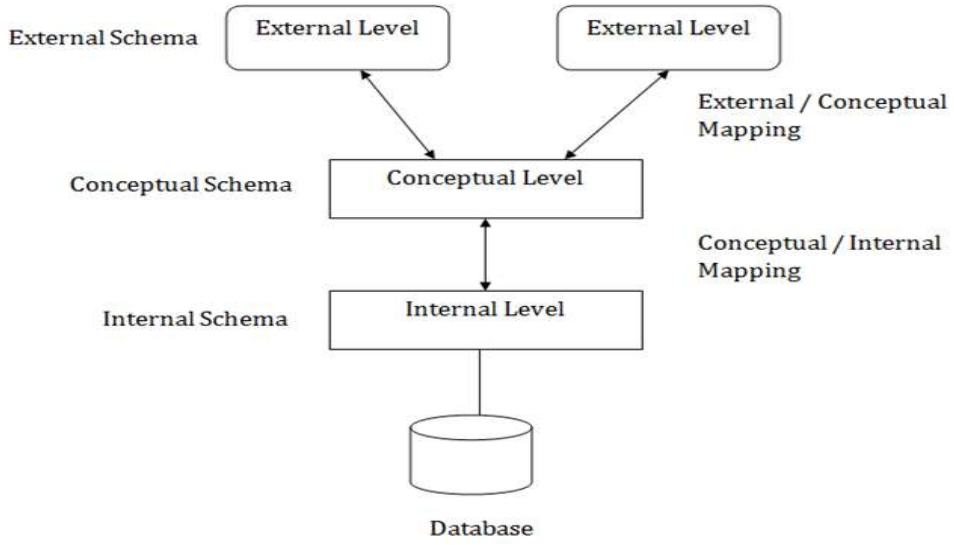
The application programmer's view of the data items and record types which the user would use is called as Subschema. Many subschema's can be derived from a single schema. The sub schema is also called as **logical view.**

Q.12 THREE SCHEMA/LEVEL ANSI/SPARC ARCHITECTURE OF DBMS

The generalization architecture of a database system is called as ANSI/SPARC model. It is a framework which is used to describe the structure of a specific database system. The three schema architecture is also used to separate the user applications and physical database. The three schema architecture contains three-levels. It breaks the database down into three different categories.

1. Internal level or Physical level
2. Conceptual level or Logical level
3. External level or View level

The three-schema architecture is as follows:



1. Internal Level: The internal schema is also known as a physical schema. The lowest level of abstraction is the internal level. It is the one closest to the physical storage device. It is used to define that how the data is actually stored in the storage medium such as hard disk., magnetic disk, etc. At this level, complex level data structures are described in detail (Detailed Figure below)

The level of data abstraction that describes how the data is actually stored on the storage device is called as **physical view.**

2. Conceptual Level: Conceptual level is also known as logical level. The conceptual schema describes the design of a database at the conceptual level. The conceptual schema describes the structure of the whole database. This level describes what data are to be stored in the database and also describes what relationship exists among those data. In this level, internal details such as an implementation of the data structure are hidden. Programmers and database administrators work at this level.

The level of data abstraction that describes what data is stored in a database and not how these logical data structures will be implemented is called as **Conceptual view.**

3. External Level: External level or View level of the architecture of a Database System is the level closest to the users and is also known as view schema. At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different view of the database. Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group.

The view schema describes the end user interaction with database systems.

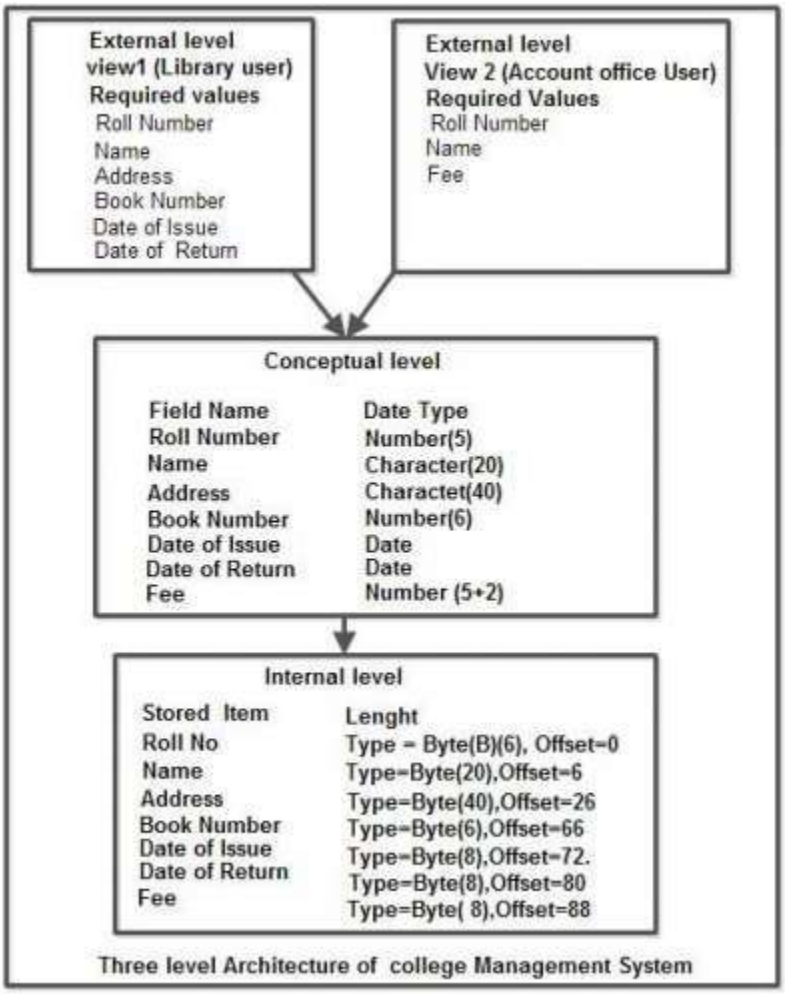


Fig: Inter-relationship between data and three levels of ANSI/SPARC model of DBMS Architecture.

Q.13 DATA INDEPENDENCE

In three-schema architecture of DBMS a database is viewed through any three levels of abstraction. Any change at any level may affect other level's scheme (structure of database). As the database keeps growing, there may be changes made at some level. However, this should never lead to redesigning and reimplementation of a database. The concept of data independence proves beneficial in such a context.

The ability to modify a schema definition in one level without affecting a schema definition in the next higher level is called as data independence

There are two types of data independence:

1. Logical Data Independence: Logical data independence refers characteristic of being able to change the conceptual schema without having to change the external schema. Logical data

independence is used to separate the external level from the conceptual view. If we do any changes in the conceptual view of the data, then the user view of the data would not be affected.

Logical data independence occurs at the user interface level.

2. Physical Data Independence: Physical data independence can be defined as the capacity to change the internal schema without having to change the conceptual schema. If we do any changes in the storage size of the database system server, then the Conceptual structure of the database will not be affected. Physical data independence is used to separate conceptual levels from the internal levels.

Physical data independence occurs at the logical interface level.

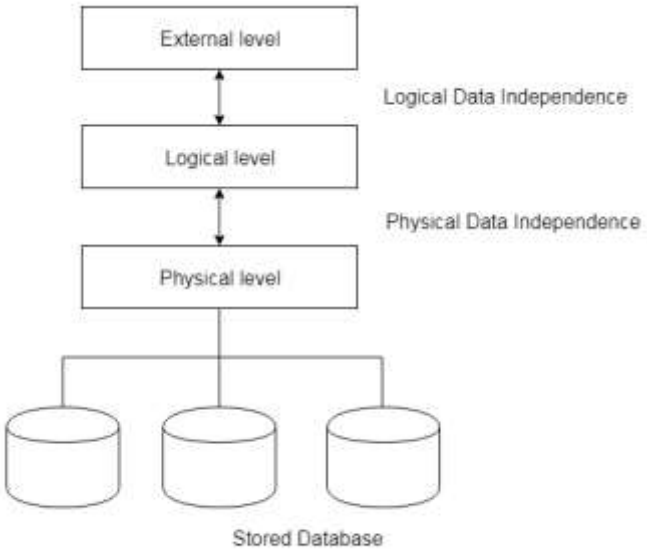


Fig: Data Independence

UNIT II: ER MODEL

Q.1 ENTITY-RELATIONSHIP MODEL/DIAGRAM

Entity-Relationship model is a model used for design and representation of relationships between data. The overall structure of a database can be expressed graphically by an E-R Model. The E-R model is based on perception of a real world which consists of a set of basic objects. The E-R model and its corresponding diagramming techniques were introduced by *P. P. Chen.* The main components of E-R Model are:

1. Entity and Entity Set
2. Attributes
3. Relationships
4. Key Attribute

1) Entity and Entity Set: An entity is a class of persons, places, objects, events or concepts about which we need to collect and store data. Each entity is distinguishable from the other entities. Categories of different entities include:

- **Persons**: Employees, Students, Customers, etc
- **Places**: College, Branch Office, Building, Room, etc
- **Objects**: Book, Computer, Vehicle, etc
- **Events**: Admission, Registration, Sale, Order, etc
- **Concepts**: Qualification, Account, Course, Stocks, etc

The ***Instance*** of an entity is a single occurrence of that entity. For example, the entity STUDENT may have multiple instances or occurrences such as Sameer, Reyaz, Mohan, etc.

An **Entity Set** is the collection of entities of the same type i.e. the entities which share some common properties or attributes. For example, Students of BG 1st Semester, Employees of Degree College Bandipora, etc.

In E-R Model entities/entity sets are represented by means of a labeled **rectangle**.

2) Attributes: An Attribute is a descriptive property or characteristic of an entity and each entity can have a number of Attributes. For example, an entity, say student, can have characteristics like Name, Parentage, Address, Contact Number, Marks, Subjects, etc.

In E-R Model Attributes are represented by means of a labeled **Oval**.

An attribute can be of many types:

1. **Simple attribute:** The attributes with values that are atomic and cannot be broken down further are simple attributes. For example, student's **age**.
2. **Composite attribute:** A composite attribute is made up of more than one simple attribute. For example, student's **address** will contain, **house no.**, **street name**, **pincode** etc.
3. **Derived attribute:** These are the attributes which are not present in the whole database, but are derived using other attributes. For example, *average age of students in a class*.
4. **Single-valued attribute:** As the name suggests, they have a single value. For Example Students Registration Number, Aadhar Card Number, etc
5. **Multi-valued attribute:** An attribute that can have multiple values at a time. For example, phone number, email address, etc

3) Relationships: The association among entities is called a relationship. For example, an employee **works_at** a department, a student **enrolls** in a course. Here, works_at and Enrolls are called relationships.

In E-R Model relationship between two entities is represented by a labeled diamond box.

Three types of relationships exist among entities. These as:

a. **One-to-One Relationship (1:1):** The one-to-one relationship is an association only between two entities. For Example, in a college each department can have only one Head. Moreover, one faculty member cannot have more than more than one department.

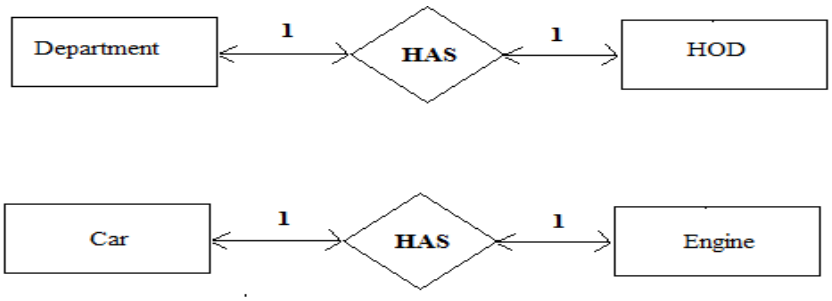


Figure: 1:1 Relationship Examples

b. **One-to-Many Relationship:** One-to-Many (1:M) Relationship exists when one entity is related to more than one entity. For example, a father may have many children but a child has one father. Similarly a Scientist can invent many inventions, but the invention is done by the only specific scientist.

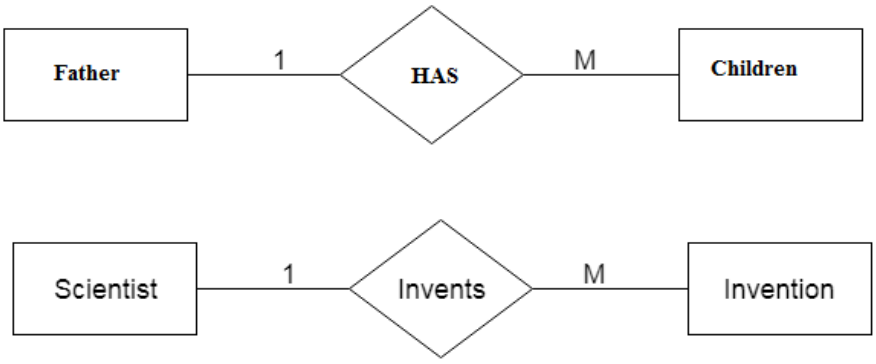


Figure: 1:M Relationship Examples

c. **Many-to-Many Relationship:** When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship (M:N).
For example, Employee can assign by many projects and project can have many employees.

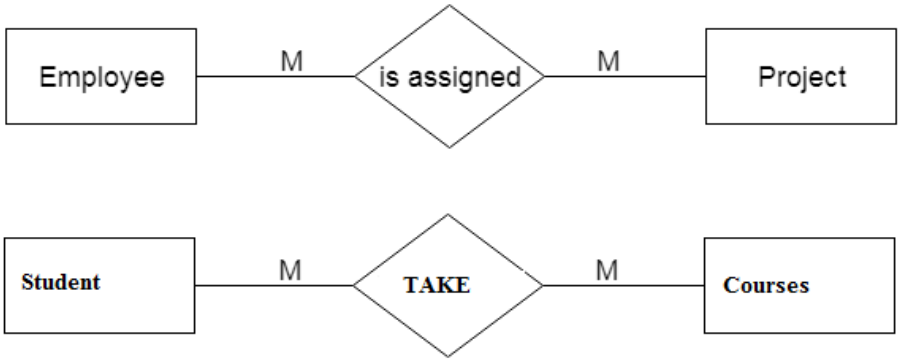


Figure: M:N Relationship Examples

4) Key Attribute: The Key attribute is an attribute that uniquely identifies an entity in the entity set. For Example, *Registration_Number* can be the key attribute for the entity set *Student.* Similarly *Employee_Code* can be the key attribute for the entity set *Employee.*

In E-R Model a key attribute is represented by an underlined label of the Oval Shape.

Q.2 STEPS TO CONSRUCT AN ERD

When documenting a system or process, looking at the system in multiple ways increases the understanding of that system. ERD diagrams are commonly used in conjunction with a data flow diagram to display the contents of a data store. They help us to visualize how data is connected in a general way, and are particularly useful for constructing a relational database. The ERD is constructed by the following steps.

- **Identify the entities.** The first step in making an ERD is to identify all of the entities that we have to use. Represent them by means of a rectangle with a description of something that the system stores information about.
- **Identify relationships.** Look at two entities, are they related? If so draw a solid line connecting the two entities.
- **Describe the relationship.** How are the entities related? Draw an action diamond between the two entities on the line you just added. In the diamond write a brief description of how they are related. Also specify the type of relationship here.
- **Add attributes.** Identify the required attributes of each entity and add those using oval-shaped symbols. Any key attributes of entities should be added here also and represent them by an underlined label.
- **Complete the diagram.** Continue to connect the entities with lines, and adding diamonds to describe each relationship until all relationships have been described. Each of your entities may not have any relationships, some may have multiple relationships.

Q.3 STRONG AND WEAK ENTITIES

There are two types of entities namely, dependent entities (also called as weak entities) and independent entity (also called as regular or strong entities). A weak entity is the one whose existence depends on the existence of another entity while as a strong entity is the one whose existence does not depend on the existence of other entity.

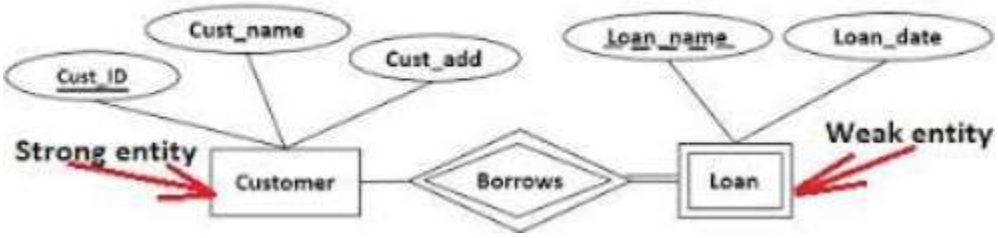


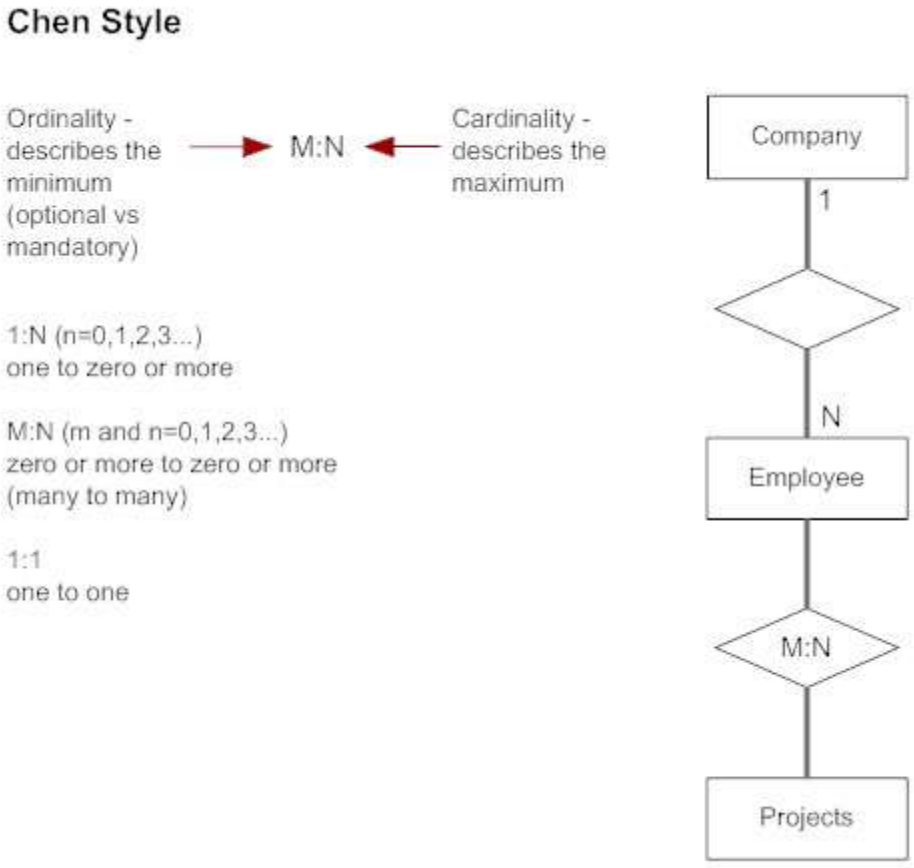
Figure: Example of Strong and Weak Entity

S.NO	STRONG ENTITY	WEAK ENTITY
1.	Strong entity always has primary key.	While weak entity has partial discriminator key.

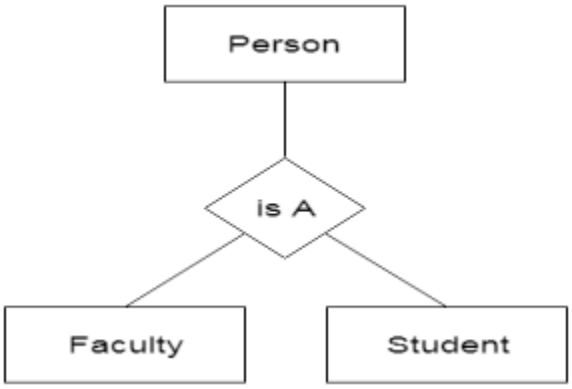
2.	Strong entity is not dependent of any other entity.	Weak entity is dependent on strong entity.
3.	Strong entity is represented by single rectangle.	Weak entity is represented by double rectangle.
4.	Two strong entity's relationship is represented by single diamond.	While the relation between one strong and one weak entity is represented by double diamond.

Q.4 FEW TERMS

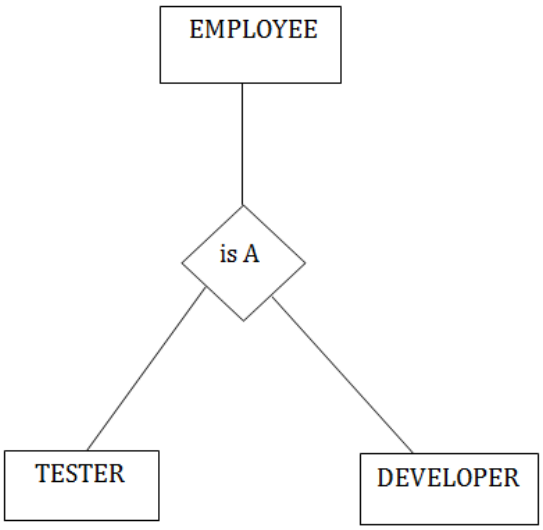
1. **Cardinality & Ordinality:** Cardinality defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.
   Ordinality is also closely linked to cardinality. While cardinality specifies the occurrences of a relationship, ordinarily describes the relationship as either mandatory or optional. In other words, cardinality specifies the maximum number of relationships and ordinarily specifies the absolute minimum number of relationships.



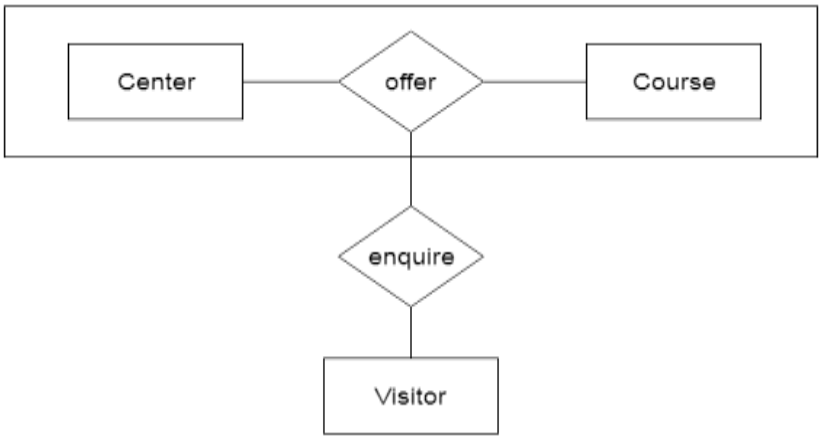
2. **Generalization:** Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity.
   Generalization is more like subclass and super class system, but the only difference is the approach. Generalization uses the bottom-up approach. In generalization, entities are combined to form a more generalized entity, i.e., subclasses are combined to make a super class. For example Faculty and Student entities can be generalized and create a higher level entity Person.



3. **Specialization:** Specialization is a top-down approach in which one higher level entity is broken down into lower level entities.
   Specialization is used to identify the subset of an entity set that shares some distinguishing characteristics. Normally, the super class is defined first, the subclass and its related attributes are defined next, and relationship set are then added. **For example:** In an Employee management system, EMPLOYEE entity can be specialized as TESTER or DEVELOPER based on what role they play in the company.



4. **Aggregation:** As the E-R Model does not support relationship within relationships, this process has been introduced to overcome the limitation. It is a process when relation between two entities is treated as a single entity. In aggregation, relationship with its corresponding entities is aggregated into a higher level entity.
   For example: Center entity offers the Course entity act as a single entity in the relationship which is in a relationship with another entity visitor. In the real world, if a visitor visits a coaching center then he will never enquiry about the Course only or just about the Center instead he will ask the enquiry about both.



5. **Recursive Relationship:** When an entity is related to itself, it is called as Unary Relationship or Recursive Relationship.
   For Example, an employee may act as a *Manager* for one or more employees, while an employee may be *Managed_by* an employee (it's one-to-many relationship).

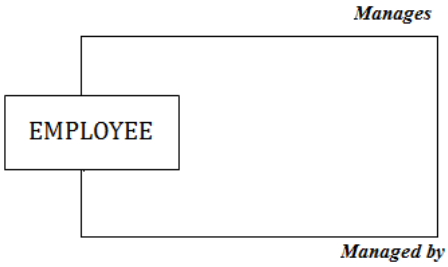


Figure: Example of Recursive Relationship