



INTRODUCTION OF DATABASE:-

DATA:

Data is a raw material that can be processed for any computing machine. For example: student name, marks of student, any number, image.

INFORMATION:

It is the data that has been converted into more useful form. For example: Report card of student.

Why we need Information?

- To gain information about surroundings.
- To keep the system up to date.

File System:

- File system is the method of storing and organizing the computer files and the data that contain to make it easy to find and access them.

DRAWBACK OF CONVENTIONAL FILE PROCESSING SYSTEM:

✓ Data redundancy and inconsistency:

In file processing system, the same information may be duplicated in several files. For example, the address and phone number of a particular customer may appear in a file that consists of saving account and in a file checking account records. It may lead to data inconsistency that is the various copy of same data may no long agree. For example, changed customer address may be reflected to saving account records but not in checking account details.

✓ Difficulty in accessing data:

The data may be stored in a file but it need not be used at the time program development. For example, one of the bank officers need to find out name of all customer who live within a particular postal code. The bank officer can do either obtain the list of all customers and extract the information manually or write the necessary application program, both alternatives are obviously unsatisfactory.

✓ Data isolation:

Because data are scattered in various file and file may be in different formats, writing new application programs to retrieve the appropriate data is difficult.

✓ Atomicity problem:

A computer system like any other electronic devices it leads to power failure. For example, If I want to transfer 500 RS from account A to Account B, if a system failure occur during the execution of the program. It is possible that 500 RS was removed from Account A but not credited to account B. It leads to result in inconsistent database state.

✓ **Integrity Problems:**

Data value store in database must satisfy certain types of consistency constraint. For example, the starting balance of any account must be greater than 1000. Program developer enforces this constraint in a program by adding appropriate integrity.

✓ **Concurrent access anomalies:**

Concurrent access to data sometimes make it inconsistency, many system allows multiple users to update data simultaneously. For example, Account A consist 5000 RS. If two customers withdraw funds 500 RS and 1000 Rs respectively from Account A at the same time. If two program run concurrently they may read the value 5000 RS and write back 4500 and 4000 respectively. Depending on which one writes the value last, the account may contain 4500 or 4000 rather than 3500.

✓ **Security Problem:**

Not every user of the database should be able to access all the data. For example, in a banking system, user cannot be able to see the personal detail of other users.

DATABASE

“Database is a collection of data organized in a meaningful way where information is stored in table.”

“A Database Management System (DBMS) is a collection of program that enables user to create, maintain and manipulate a database.”

- The DBMS is hence a general purpose software system that facilitates the process of defining, constructing and manipulating database for various applications.
- The primary goal of database management system is to provide a way to store and retrieve database information that is convenient and efficient.

- A database management system is a combination of hardware and software that can be used to setup and monitor a database.
- Access2003 databases have a **.MDB extension** and OpenOffice databases have a **.ODB extension**
- There are 2 types of database

1. **Flat Database:** Database can be stored in a single file with all the information stored together in tables and we cannot create relation between them. Ex. Ms Excel.

2. **Relational Database:** Database can be stored in multiple table with some common fields and create relation between number of tables based on same field. Ex. MS Access.

TABLE:

“Table is a collection of relation information about specific topic. You can keep many tables in a database”.

Table is a group of records and fields stored in rows and column in a table, a row is referred to as a record , and a column is referred as a field.

Roll No	Name	Marks1	Marks2
1	Om	43	45
2	Sai	45	40
3	Ram	35	47

Here the information about student result is stored in form of row and column. Each row contains information related to different student. In Excel, First row is used for heading which is known as “field” and other row contain student’s information is known as “Record”

TUPLE:

A single row of a table, which contains a single record for that relation is called a tuple.

FIELD:

- It is a data structure for a single piece of data. It is a container that is reserved for holding specific value.
- In above example, Roll No., Name, Marks1, Marks2 are known as field.

RECORD

- Records in a table is an array of information.
- It is a set of data about a person or thing.
- To view the information of any person, record is used.
- In above example, different student's rollno wise information are displayed. All these rows are known as records.

IMPORTANCE/MERIT/ADVANTAGE OF DATABASE

- **Controlled data redundancy:** During database design, various files are integrated and each logical data item is stored at central location. This eliminates same data item in different files, and ensures consistency and saves the storage space.
- **Enforcing data integrity:** In database approach, enforcing data integrity is much easier. Various integrity constraints are identified by database designer during database design.
- **Data sharing:** The data stored in the database can be shared among multiple users or application programs.
- **Ease of application development:** The application programmer needs to develop the application programs according to the users' needs.
- **Data security:** Since the data is stored centrally, enforcing security constraints is much easier. The DBMS ensures that the only means of access to the database is through an authorized channel.
- **Backup and recovery:** The DBMS provides backup and recovery subsystem that is responsible for recovery from hardware and software failures.

- **Databases Save Time :** Instead of rummaging through endless piles of paperwork, a database pulls up information with simple query. A user can enter in specific keywords in order to recall information. The database becomes a more efficient solution than paper files held in a file folder.

DEMERIT/DISADVANTAGE OF DATABASE

- **Cost of Hardware and Software:** A processor with high speed of data processing and memory of large size is required to run the DBMS software. It means that you have to up grade the hardware used for file-based system. Similarly, DBMS software is also very costly.
- **Cost of Data Conversion:** When a computer file-based system is replaced with database system, the data stored into data file must be converted to database file. It is very difficult and costly method to convert data of data file into database. You have to hire database system designers along with application programmers.
- **Cost of Staff Training:** Most database management system are often complex systems so the training for users to use the DBMS is required.
- **Appointing Technical Staff:** The trained technical persons such as database administrator, application programmers, data entry operations etc. are required to handle the DBMS. You have to pay handsome salaries to these persons. Therefore, the system cost increases.
- **Database Damage:** In most of the organization, all data is integrated into a single database. If database is damaged due to electric failure or database is corrupted on the storage media, the your valuable data may be lost forever.

COMPONENT OF OPEN OFFICE BASE:

DATABASE WINDOW

- All The objects of a database are stored on a single file having on extension **.odb**. The objects are managed through database window.

TABLES

- Any information you enter in the database are stored in table.
- Table consists of row and column of values.

QUERIES

- A query is an object that provides a personal view of the data stored in existing tables.
- Various types of queries exist to select, update, erase data, but for now we use them to extract from the tables the data which complies with certain conditions.
- For example, we will be able to create a query to obtain all the information about clients who have the zip code 46625.

FORMS

- A form is an Open Office object designed to introduce, view and edit the data in the tables.
- There are various types of forms, but the most commonly used is the data record form for the introduction of the different clients in the CLIENTS table (for example).

REPORTS

- A report is an Open Office object designed to format, calculate, print and summarize the data selected in a particular table.
- A report is generally used to print data.

DATA TYPE USED IN DATABASE

<u>DATA TYPE</u>	<u>DESCRIPTION</u>
VARCHAR	A VARIABLE length string (can contain letters, numbers, and special characters). The <i>size</i> parameter specifies the maximum column length in characters - can be from 0 to 65535
CHAR	A FIXED length string (can contain letters, numbers, and special characters). The <i>size</i> parameter specifies the column length in characters - can be from 0 to 255. Default is 1

INTEGER	It is used for calculation the long integer . Store whole number form -2147483648 to 2147483647. Each field use 8 Bytes.
Date/Time	It is used to store data and time. Data and time values from the year 100 to 9999. Each field use 8 Byte in memory.
DOUBLE(size,d)	A normal-size floating point number. The total number of digits is specified in <i>size</i> . The number of digits after the decimal point is specified in the <i>d</i> parameter
FLOAT(size,d)	A floating point number. The total number of digits is specified in <i>size</i> . The number of digits after the decimal point is specified in the <i>d</i> parameter.

DIFFERENCE BETWEEN EXCEL AND ACCESS

<u>EXCEL</u>	<u>ACCESS</u>
Excel is for spreadsheets and financial calculations.	Access is a relational database program.
Extension of Excel file is .xls	Extension of access file is .mdb
The primary use of Excel is to create financial spreadsheets. Excel has some limited capabilities to sort data but its primary function is to create financial spreadsheets.	You would use Access to collect, manipulate and sort data.
Excel has limited functionality and cannot make the connections that Access can.	If you need to create relationships between your data , Access is the only choice.
Excel can only handle up to 15,000 records.	Access is meant to handle thousands of records , all related through multiple tables.

Excel does not have the ability to create queries , although it can create charts, graphs and pivot tables.	Access is meant to create calculated queries , making connections based through a "wizard" or through user programming.
It use flat database	It use flat and relational database .

CREATING A FORM IN OPEN OFFICE DATABASE

1. Open database by Select File → Open and select database .
2. Click on the Form tab under objects.
3. Select the creation method you wish to use. There are 2 way to create form
 1. Create form in Design view.
 2. Create form by using wizard.
4. Select the data source and click Ok.
5. Select the Form Field to be used and click **Next**.

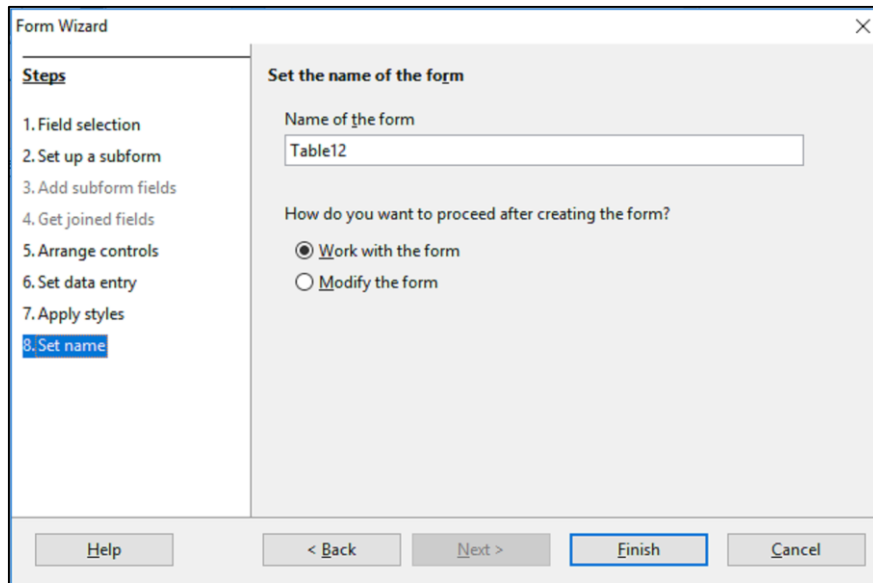
6. Select the Form Layout from Columnar, Tabular, Datasheet, Justified, Pivottable, Pivotchart

7. Select Data entry mode if you want, Click Next.

8. Apply style of your form if you want, Click Next.

9. Provide the title for Form

10. Click on **Finish**.



CREATING A REPORT IN OPEN OFFICE

To generate report in MS Access following steps are followed.

1. Open database by Select File → Open and select database .
2. Click on the Report tab under objects and select **Create report by using wizard**
3. Report wizard dialog box appear as per figure. Select the table and select the fields which you want to insert into report using button: >, >>, <, <<. Then press Next button.

4. Provide Grouping information if you want to Group data and Press Next Button.

Grouping simply “groups” records by an item in the report you are designing. We’ll group by state. This means that “records” from a state will be in a “group” (e.g. people from Virginia will be in one group, the folks from Washington in another, and so on).

VIDYABHARTI TRUST COLLEGE OF BBA & BCA. UMRAKH

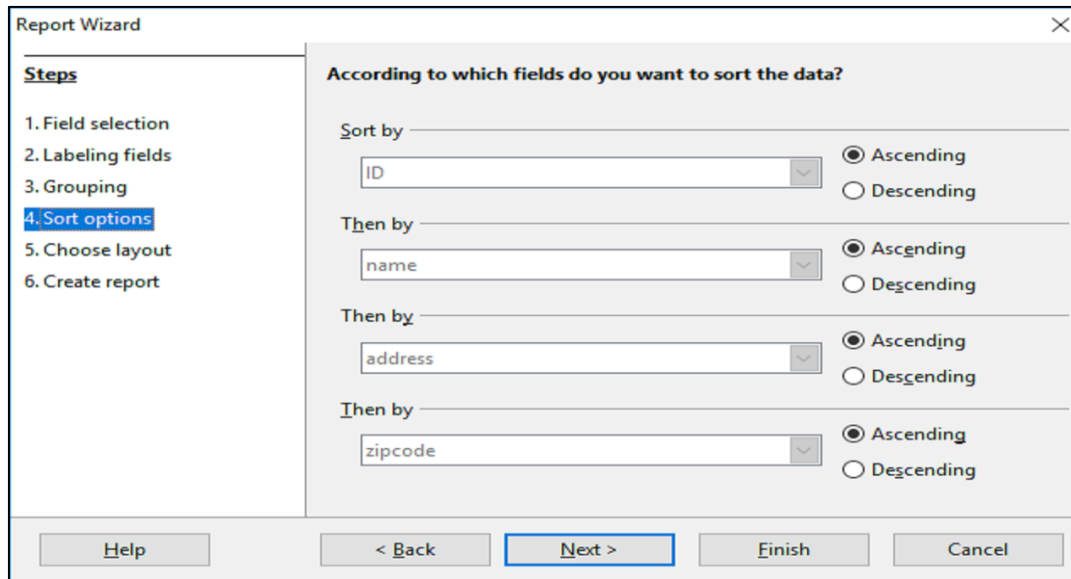
SUB: 105-Data Manipulation and Analysis

Unit : 3 – Concept of Database

5. Provide Sorting information if you want to sort data and Press Next Button.

The screenshot shows the 'Report Wizard' dialog box, specifically the 'Do you want to add grouping levels?' step. On the left, a 'Steps' list shows '3. Grouping' as the current step. The main area has two panes: 'Fields' containing 'ID', 'name', 'address', and 'zipcode', and an empty 'Groupings' pane. Navigation buttons at the bottom include '< Back', 'Next >', 'Finish', and 'Cancel'. A note at the bottom states: 'Note: The dummy text will be replaced by data from the database when the report is created.'

The screenshot shows the 'Tables or queries' dialog box. A dropdown menu at the top shows 'Table: Vacations'. Below, there are two lists: 'Available fields' on the left and 'Fields in report' on the right. In the 'Available fields' list, 'MPayment' is selected. In the 'Fields in report' list, 'Miscellaneous' is selected. Between the lists are four buttons: '>', '>>', '<', and '<<'. Other fields in the 'Available fields' list include Odometer, BPayment, LPayment, SPayment, SnackNo, SnPayment, MiscNotes, and MiscPayment. Other fields in the 'Fields in report' list include Date, Motel, Tolls, Breakfast, Lunch, Supper, and SnackCost.



6. Select the layout of report and Press Next Button.

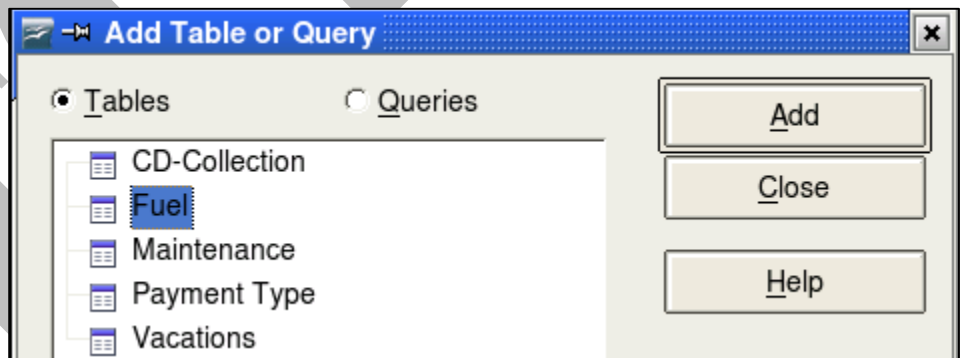
7. Give the report title and Press Finish Button.

CREATING A QUERY USING DESIGN VIEW IN OPEN OFFICE

To generate query in Open Office following steps are followed.

1. Open database by Select File → Open and select database .

2. Click **Create Query in Design View**



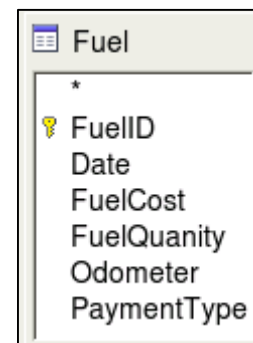
3. **Add Table.**

4. Add field to the table at bottom.

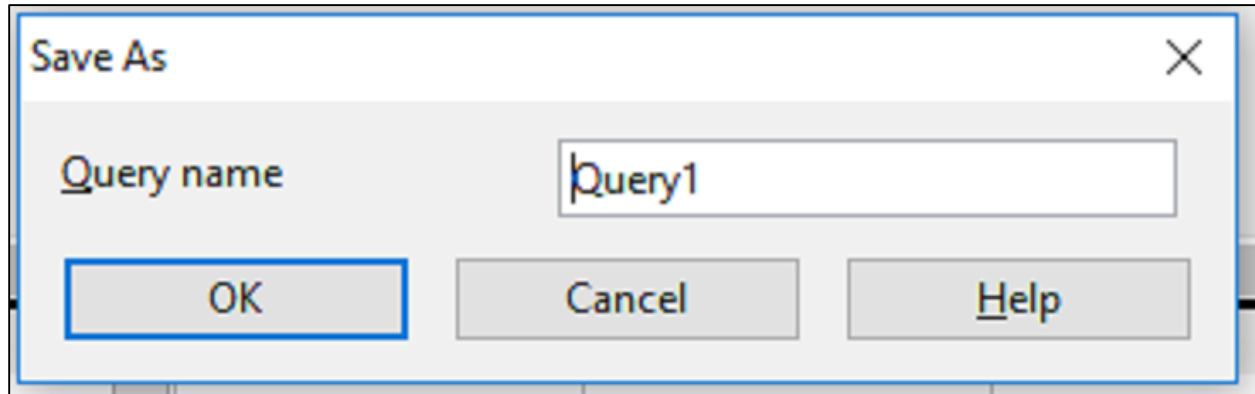
a. Double-click the *FuelID* field in the Fuel table.

b. Double-click the Odometer field.

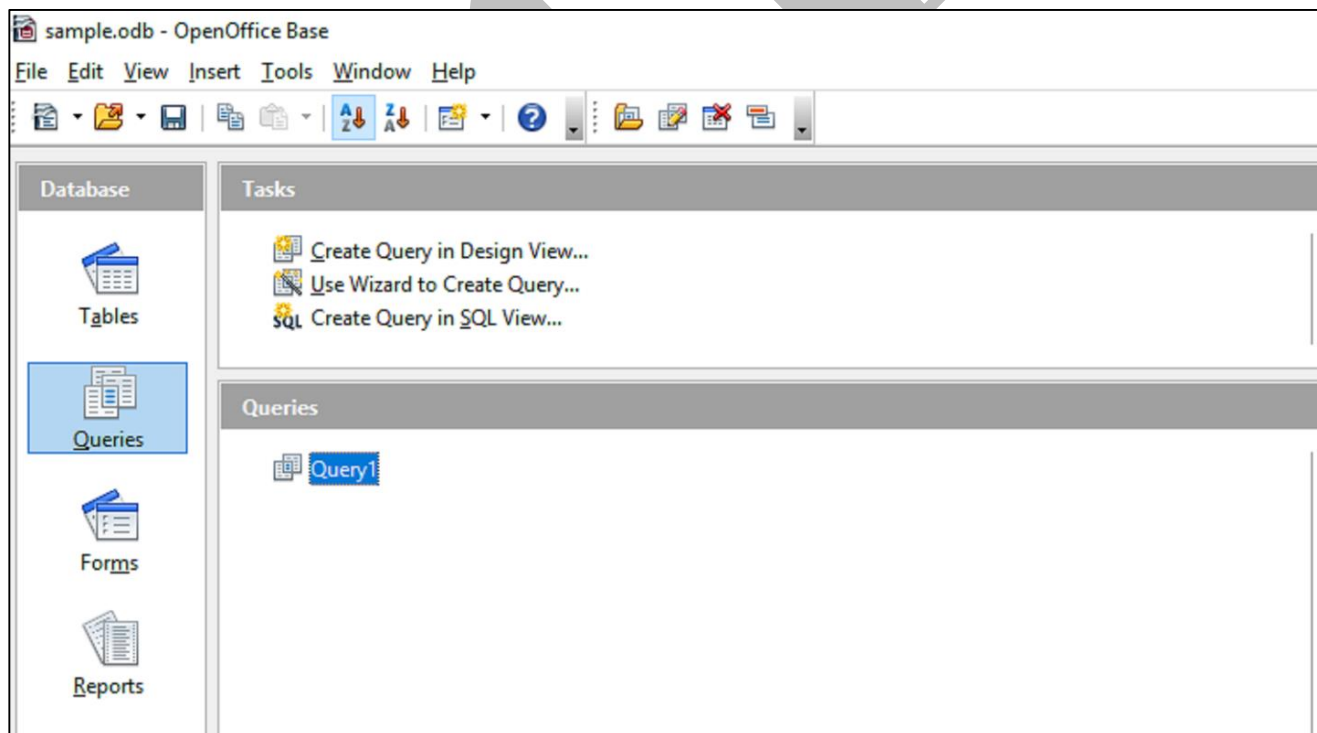
c. Double-click the FuelQuantity field



5. The Save As window should appear. Enter the name that you'd like to assign to the query and click on the OK button. In this example, we've saved the query as Query1.



6. You should now see the query appear in the Database Window



❖ **History of DBMS:-**

- 1960 - Charles Bachman designed first DBMS system
- 1970 - Codd introduced IBM'S Information Management System (IMS)
- 1976- Peter Chen coined and defined the Entity-relationship model also know as the ER model
- 1980 - Relational Model becomes a widely accepted database component
- 1985- Object-oriented DBMS develops.
- 1990s- Incorporation of object-orientation in relational DBMS.
- 1991- Microsoft ships MS access, a personal DBMS and that displaces all other personal DBMS products.
- 1995: First Internet database applications
- 1997: XML applied to database processing. Many vendors begin to integrate XML into DBMS products.

3.1 DATABASE CHARACTERISTICS:

Traditionally, data was organized in file formats. DBMS was a new concept then, and all the research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following characteristics –

- **Real-world entity** – A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses the behavior and attributes too. For example, a school database may use students as an entity and their age as an attribute.
- **Relation-based tables** – DBMS allows entities and relations among them to form tables. A user can understand the architecture of a database just by looking at the table names.
- **Isolation of data and application** – A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.
- **Less redundancy** – DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.
- **Consistency** – Consistency is a state where every relation in a database remains consistent. There exist methods and techniques, which can detect

attempt of leaving database in inconsistent state. A DBMS can provide greater consistency as compared to earlier forms of data storing applications like file-processing systems.

- **Query Language** – DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.
- **ACID Properties** – DBMS follows the concepts of **Atomicity**, **Consistency**, **Isolation**, and **Durability** (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database. ACID properties help the database stay healthy in multi-transactional environments and in case of failure.
- **Multiuser and Concurrent Access** – DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when users attempt to handle the same data item, but users are always unaware of them.
- **Multiple views** – DBMS offers multiple views for different users. A user who is in the Sales department will have a different view of database than a person working in the Production department. This feature enables the users to have a concentrate view of the database according to their requirements.
- **Security** – Features like multiple views offer security to some extent where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. For example, a user in the Sales department cannot see the data that belongs to the Purchase department. Additionally, it can also be managed how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.

DATABASE SYSTEMS: DATA ABSTRACTION

The major purpose of a database system is to provide users with an abstract view of the system. The system hides certain details of how data is stored and maintained. Complexity should be hidden from database users.

There are several levels of abstraction:

(a) Physical Level:

- How the data are stored.
- E.g. index, B-tree, hashing.
- Lowest level of abstraction.
- Complex low-level structures described in detail.

(b) Conceptual Level:

- Next highest level of abstraction.
- Describes what data are stored.
- Describes the relationships among data.
- Database administrator level.

(c) View Level:

- Highest level.
- Describes part of the database for a particular group of users.
- Can be many different views of a database.
- E.g. tellers in a bank get a view of customer accounts, but not of payroll data.

Data Abstraction Model

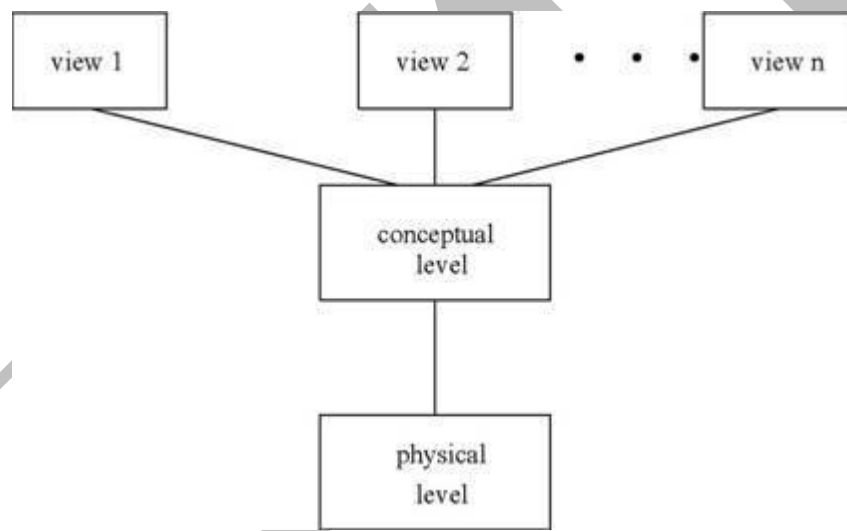


Figure The three levels of data abstraction

- ❖ **Instance And Schema:** **Instance:** Database change over time. The information in a database at a particular time is called instance of the database. **Schema:** The overall design of the database is called the

database Schema. There are several schemas, corresponding to levels of abstraction:

1) Physical Schema: It describes the database design at physical level of abstraction.

2) Logical Schema: It describes the database design at logical level of data abstraction.

3) Sub Schema: Database may have many schemas at the view level called sub schema.

3.1.1 Data Independence (Logical and Physical) :

Data Independence: The ability to modify schema definition in one level without affecting a schema definition in the next higher level is called data independence. There are two levels of data independence.

Physical Data Independence: Physical data independence is the ability to modify the physical schema without affecting logical Schema. Schema modification at the physical level is occasionally necessary to improve performance.

Logical Data Independence: Logical data independence is the ability to modify the logical schema without affecting Physical Schema. Modifications at the logical levels are necessary whenever the logical structure of the database is altered. Logical data independence is more difficult to achieve than the physical data independence, since application programs are heavily dependent on the logical structure of the data then very access.

3.1.2 Components of Database (User, Application , DBMS, Database)

Component Name	Task
Application Programmers	The Application programmers write programs in various programming languages to interact with databases.
Database Administrators	Database Admin is responsible for managing the entire DBMS system. He/She is called Database admin or DBA.

End-Users

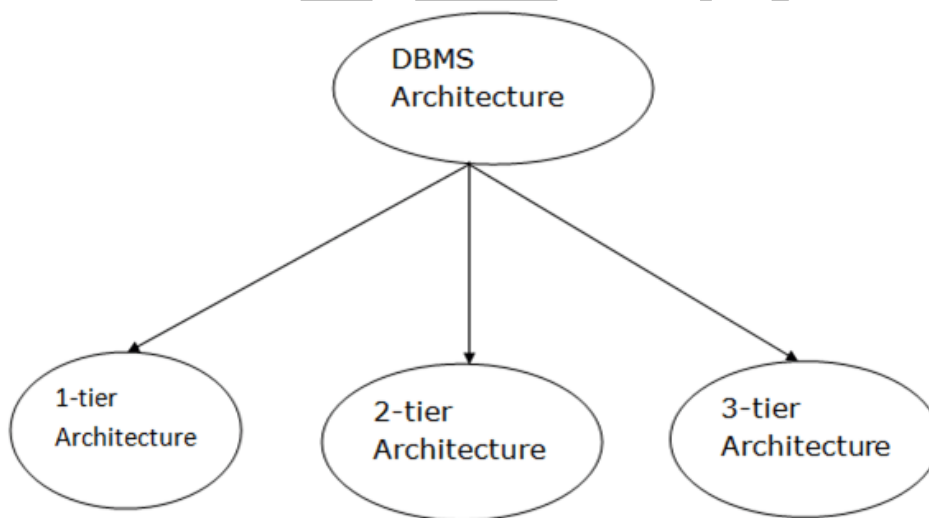
The end users are the people who interact with the database management system. They conduct various operations on database like retrieving, updating, deleting, etc.

3.1.3 Database Architecture (1-tier, 2-tier, 3-tier)

3.1.3.1 Comparison, advantages and disadvantages.

- The DBMS design depends upon its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.
- The client/server architecture consists of many PCs and a workstation which are connected via the network.
- DBMS architecture depends upon how users are connected to the database to get their request done.

Types of DBMS Architecture



Database architecture can be seen as a single tier or multi-tier. But logically, database architecture is of two types like: **2-tier architecture** and **3-tier architecture**.

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.

Advantages

1. Easy to implement and optimize performance.
2. Do not have compatibility or Context switching issues.
3. The cost of deployment is less eg - development and management cost.

Disadvantages

1. Do not support remote/ distributed access for data resources.
2. Monolithic manner of the code causes higher maintenance.
3. The cost of the central mainframe is high

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.
- 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.

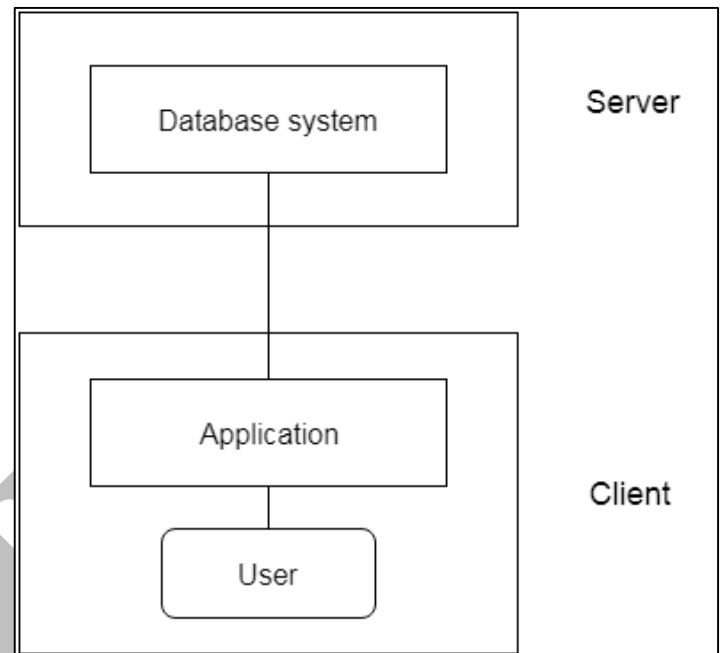


Fig: 2-tier Architecture

Advantages:

1. Easy to maintain and modification is bit easy
2. Communication is faster

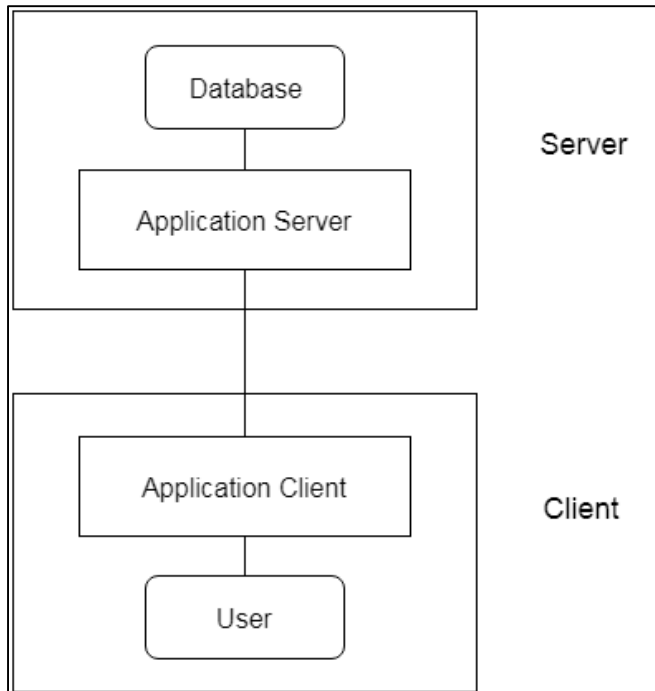
Disadvantages:

1. In two tier architecture application performance will be degrade upon increasing the users.
2. Cost-ineffective

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.



Advantages

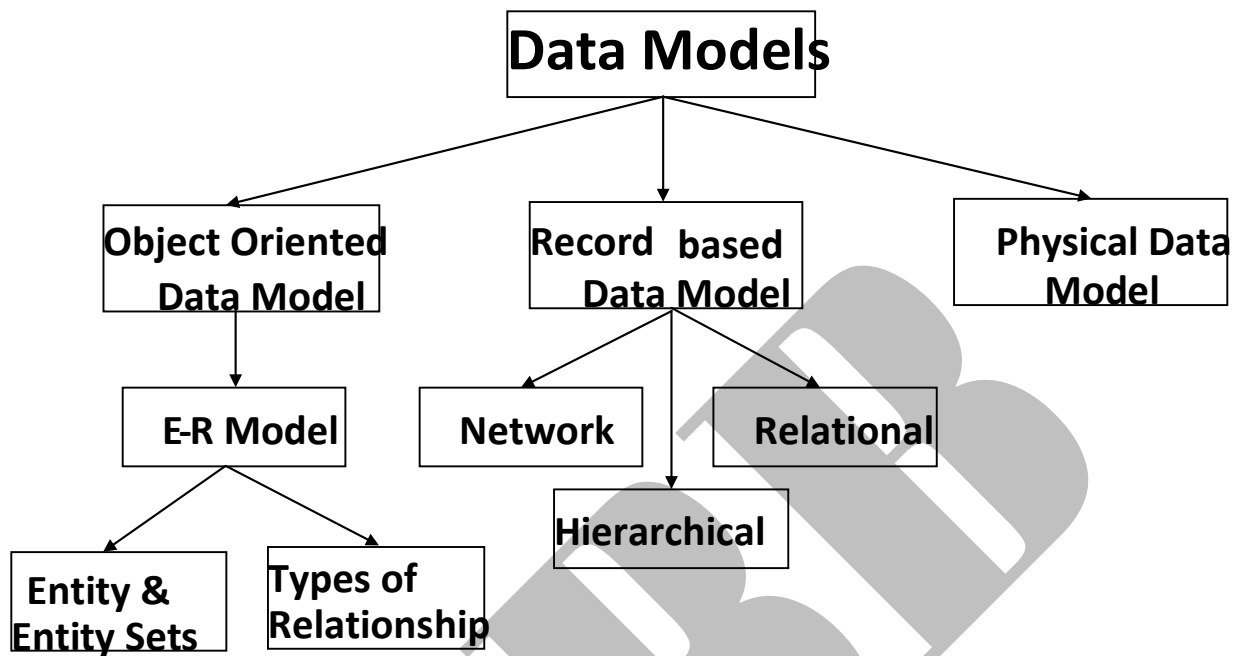
1. High performance, lightweight persistent objects
2. Scalability – Each tier can scale horizontally
3. Performance – Because the Presentation tier can cache requests, network utilization is minimized, and the load is reduced on the Application and Data tiers.
4. High degree of flexibility in deployment platform and configuration
5. Better Re-use
6. Improve Data Integrity
7. Improved Security – Client is not direct access to database.
8. Easy to maintain and modification is bit easy, won't affect other modules
9. In three tier architecture application performance is good.

Disadvantages

1. Increase Complexity/ Effort

3.2 DATABASE MODELS (HIERARCHICAL, NETWORK, E/R, RELATIONAL)

Data models are collection of conceptual tools for describing data, data relationship and data constraints. Data models can facilitate interaction among the designer, application programmers and end users. Data models define how data is connected to each other and how they are processed and stored inside the system.

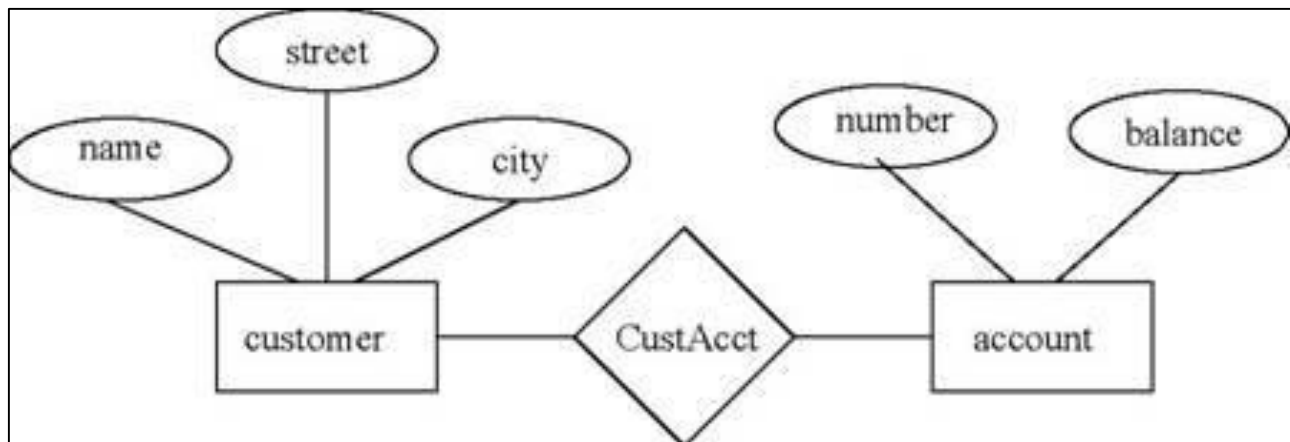


Data models are a collection of conceptual tools for describing data, data relationships, data semantics and data constraints. There are three different groups:

1. Object-based Logical Models.
2. Record-based Logical Models.
3. Physical Data Models.

1. Object-base Logical Models:

- Describe data at the conceptual and view levels.
- Provide fairly flexible structuring capabilities.
- Allow one to specify data constraints explicitly.



Over 30 such models, including

- Entity-relationship model.
- Object-oriented model.
- Binary model.
- Semantic data model.
- Info logical model.
- Functional data model.

At this point, we'll take a closer look at the entity-relationship (E-R) and object-oriented models.

❖ **The Object-Oriented Model**

1. The object-oriented model is based on a collection of objects, like the E-R model.

- An object contains values stored in instance variables within the object.
- Unlike the record-oriented models, these values are themselves objects.
- Thus objects contain objects to an arbitrarily deep level of nesting.
- An object also contains bodies of code that operate on the object.
- These bodies of code are called methods.
- Objects that contain the same types of values and the same methods are grouped into classes.
- A class may be viewed as a type definition for objects.
- Analogy: the programming language concept of an abstract data type.
- The only way in which one object can access the data of another object is by invoking the method of that other object.

- This is called sending a message to the object.
- Internal parts of the object, the instance variables and method code, are not visible externally.
- Result is two levels of data abstraction.
- For example, consider an object representing a bank account.
- The object contains instance variables number and balance.
- The object contains a method pay-interest which adds interest to the balance.
- Under most data models, changing the interest rate entails changing code in application programs.
- In the object-oriented model, this only entails a change within the pay-interest method.

2. Unlike entities in the E-R model, each object has its own unique identity, independent of the values it contains:

- Two objects containing the same values are distinct.
- Distinction is maintained in physical level by assigning distinct object identifiers.

2. Record based Data model:

Record base logical data model are used in describing data at view levels. Record based data models do not include a mechanism for direct representation of code in the database. These data models are used on user level of data.

Record base data models are so popular because the database is structured in fixed format that is record. Each record type defines a fixed number of fields or attributes. Each attributes of a fix length. The use of fixed length record simplifies the physical level implementation of database.

There are 3 record base data models:

- **Relational Data Model**
- **Network data Model**
- **Hierarchical data Model**

1) Relational Data Model:

Data and relationships are represented by a collection of tables. Each table has a number of columns with unique name.

Example: Customer, Account.

Main reason to introducing this mode was to increase the productivity of the application program when a change is made to the database users need not know the exact physical structure to use the database.

Those models define how they are structured in database physically and how they are inter-related.

Example: Consider tables **Department** and **Employee**.

Employee:-

Emp_id	Emp_name	Address	Dept_no
100	Joseph	Citylight	10
101	Ajay	Althan	20
102	Mukesh	Adajan	30
103	Rajesh	Bhatar	10

Department:-

Dept_no	Dept_name
10	Account
20	Marketing
30	Sales

- Now in Employee table, we have column with unique identifies each employee- that is Emp_id. This column has unique value and able to differentiate each employee from each other. Such column is called **primary key**.
- Similarly Department table has Dept_id as Primary key.
- Instead of storing whole information about the Department, we have Dept_id from Department table stored.
- By using a data from Department table, we have established the relation between Employee and Department table.
- This type of data model is called **Relational Data model**.

Advantage:

Structure Independence: Any changes to the database structure. Example: Age is added to Employee table but it does not change the relation between these two tables.

Ease of Design: To implement relational model program need not be aware how table are stored internally.

Simplicity in access: Because data are stored in table format.

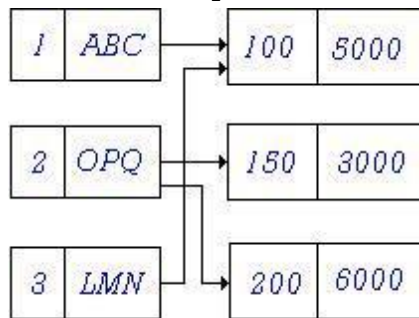
Disadvantages:

High Hardware and software overhead: Relational model frees database designer, programmer and end users from their tedious efforts required to stored, retrieve and design relationship between data.

2) Network data Model:

Data are represented by collection of records. Relationships among data are represented by links. This links can be viewed as a pointer. The network data model represents data for an entity set by logical record type.

Network data model conceived as flexible way of representing objects and their relationship.



Here above diagram represent that customer ABC having account number 100 and balance is 5000.

Advantages:

Conceptually simple: This model is conceptually very easy to design.

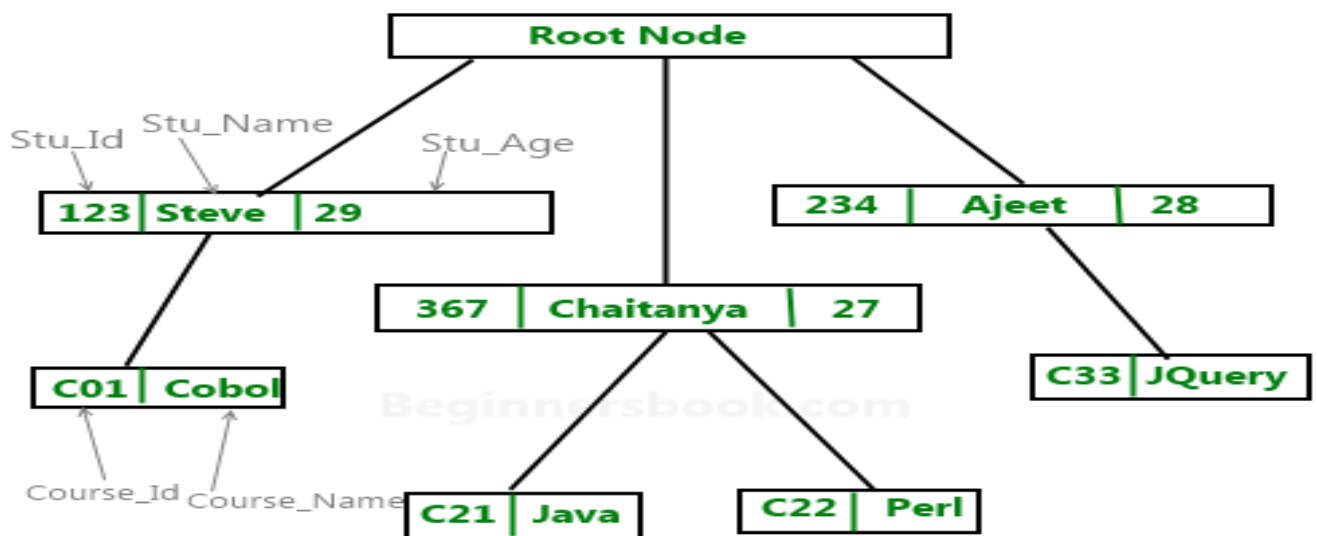
Handle Complex Relationship: Many too many relationship can be implemented without any repetition.

Disadvantages:

Complexity: In network data model data are stored using pointer. In this model to design a graph many pointer are required. Thus the entire database becomes very complex.

3) Hierarchical data Model:

The hierarchical data model organizes data in a tree structure. There is a hierarchy of parent and child data segments. This structure implies that a record can have repeating information, generally in the child data segments. Hierarchical Model is similar to network model. Hierarchical model represented by collection of record and relationships are represented by link. A link is an association between precisely to record. In this model records are organize in the form of rooted tree where the root of a tree is a dummy



3. Physical Data Models:

The Physical data model specifies implementation details which may be features of a particular product or version as well as configuration choice for the database instance. Physical data model is a representation of a data design which takes into account the facilities and constraints of a given database management system.

These data model are used to describe data at the lowest level. A complete physical data model will include all the database artifacts

required to create relationship between tables or achieve goals, such as indexes, constraints, linking tables, and portioned table.

The physical data model can used to calculate storage estimates and may include specific storage allocation details for a given database system. This involves the actual design of a database according to the requirement that were established during logical modeling.

Difference between Object base data model and Record base data model:

	Object Oriented data model	Record Base data model
Definition	Object oriented data model is based on a collection of objects. An object is an instance of the class. A is one of the structure available in the object oriented which shows the collection of objects.	Relational data model falls under record based logical models. These are used in describing data at conceptual and view levels.
Structure	Object contains data as well as the methods are called functions to operate upon the data contained in the object itself.	The database is structured in fixed format records of several types. Each record type defines a fixed number of fields and each fields is usually of fix length
Represent	It generally represents or defines types.	It represents value or objects.
Implements	It is conceptual model, so various techniques can be applied to implements objects and relationships of this data model.	I gives idea of implementation. For example relational model shows relationship and objects both as tables. While network and hierarchy model, relationship shows using links.

Example:	Object model, E-R model	Relational model Hierarchical Model Network Model
-----------------	-------------------------	---

3.2.1 E/R model : Entity, Relationship, Attribute

❖ The E-R Model

The entity-relationship model is based on a perception of the world as consisting of a collection of basic objects (entities) and relationships among these objects.

- An entity is a distinguishable object that exists.
- Each entity has associated with it a set of attributes describing it.
- E.g. number and balance for an account entity.
- A relationship is an association among several entities.
- E.g. A cust acct relationship associates a customer with each account he or she has.
- The set of all entities or relationships of the same type is called the entity set or relationship set.
- Another essential element of the E-R diagram is the mapping cardinalities, which express the number of entities to which another entity can be associated via a relationship set.

We'll see later how well this model works to describe real world situations.

E-R data model have three basic notations:

- **Entity & Entity Sets**
- **Relationship Set**
- **Attributes**

➤ **Entity & Entity Sets:**

- **Entity:** An entity is a “thing” or “object” in the real world that is distinguishable from all other object.

Entities are described in a database by a set of attributes. Ex Roll No, Name and Address are attributes of student entity. **Example:** Customers, Accounts, Person.

- **Entity Sets:** It is a set of entities of same type that shares the same properties or attributes.
Individual entities that are said to be entity set.

Example:

- All individual customers are the extension of the entity set Customer.
- In bank individual employee is an entity but a combination of employees is said to an entity set.

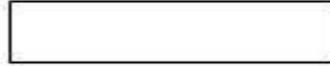
CUSTOMER TABLE

Customer_id	Customer_Name	Customer city
10100590123	Amit Patel	Surat
10100578999	Chirag Mehta	Valsad
10100554319	Kajal Bhanushali	Surat
10100561234	Yatin Patel	Vapi
10100578643	Krishna Patel	Valsad
10100590201	Mohini Patel	Surat

➤ **Entity Relationship Diagram Elements :-**

1. Rectangle:-

- It represents entity set.



2. Ellipse:-

- It represents attributes.



3. Diamond:-

- It represents relationship among entity set



4. Line:-

- It link attribute to entity set and entity set to relationship.

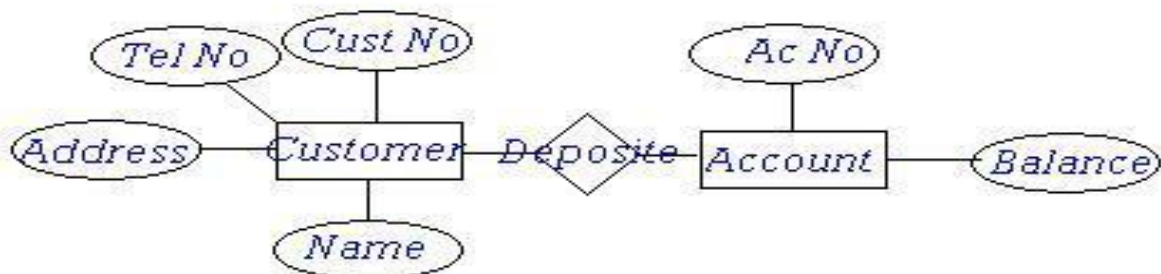
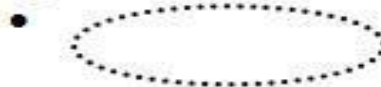
5. Double Elipse:-

- It represent multi valued attributes.



6. Dashed Elips

- It Represent derived attributes



This diagram represents banking system consist of customers and the accounts of customers. The E-R diagram indicates that there are two entities set customer A/C will attribute as out line earlier. The diagram also shows a relationship

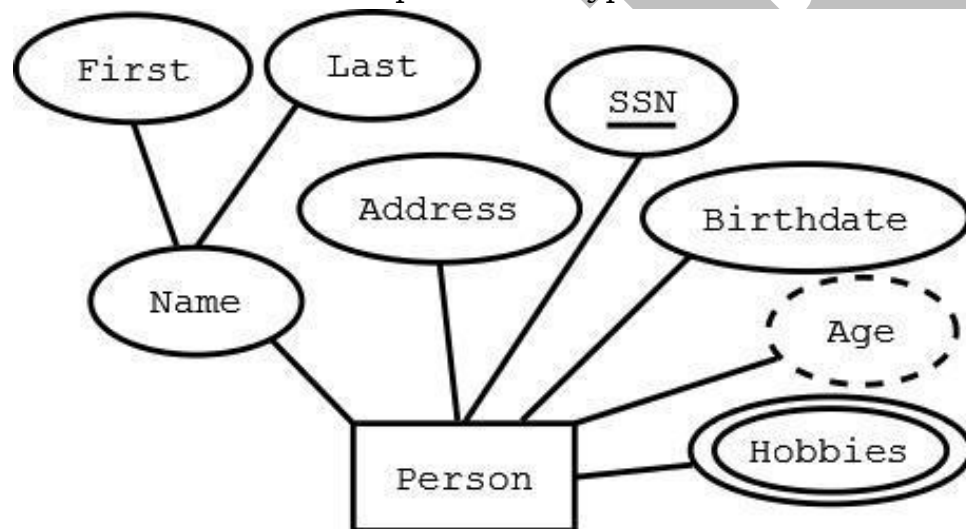
depositor between customers and an account. This diagram also represents the relationship between customers and A/C as many to many.

3.2.2 E/R Diagram : One to one, one to many, many to one, many to many

Relationship: A relationship is an association among the several entities.

Example: A depositor relationship associates a customer's entity with account entity.

The set of all relationships of same type is called **relationship set**.

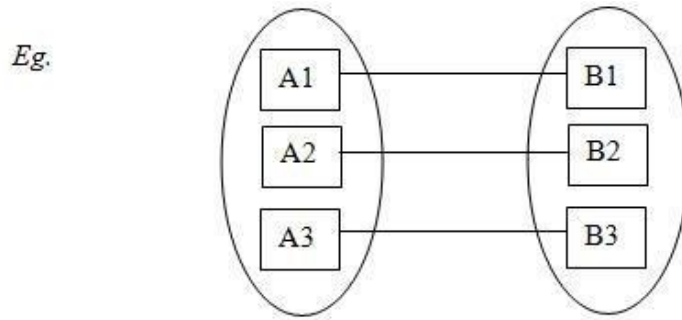


✓ **Types of Relationship (Mapping Cardinalities):** Mapping cardinalities or cardinality ratios, express the number of entities to which another entity can be associated via a relationship set.

- **One to One**
- **One to Many**
- **Many to One**
- **Many to Many**

1. **One to One:** An Entity in "A" is associated with at most one entity in "B" and an Entity in "B" is associated with at most one Entity with "A".

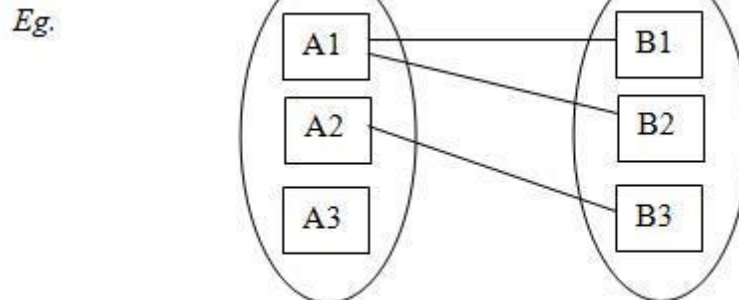
Example:-



2. One to Many:

in „A“ is Associated with any number of entities in „B“ and an Entity in „B“ is Associated with at most one Entity with „A“.

Example:

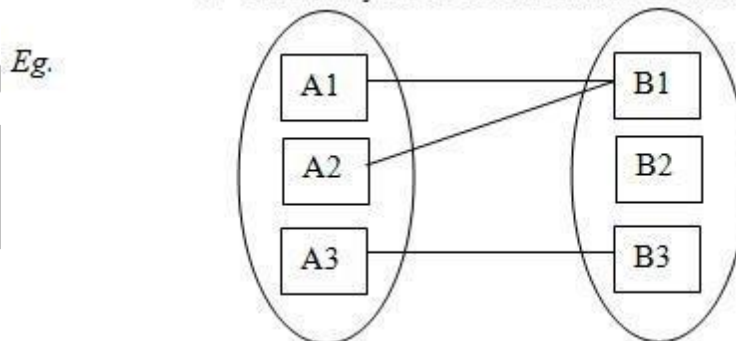


Eg. Parent & Children

3. Many to One:

An Entity in „A“ is Associated with at most one entity in „B“ and an Entity in „B“ However can be associated with any number of entities in it.

Example:

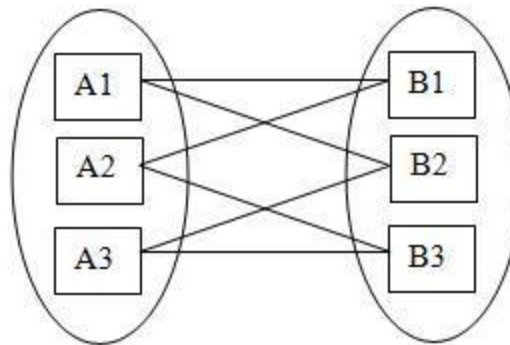


4. Many to Many:

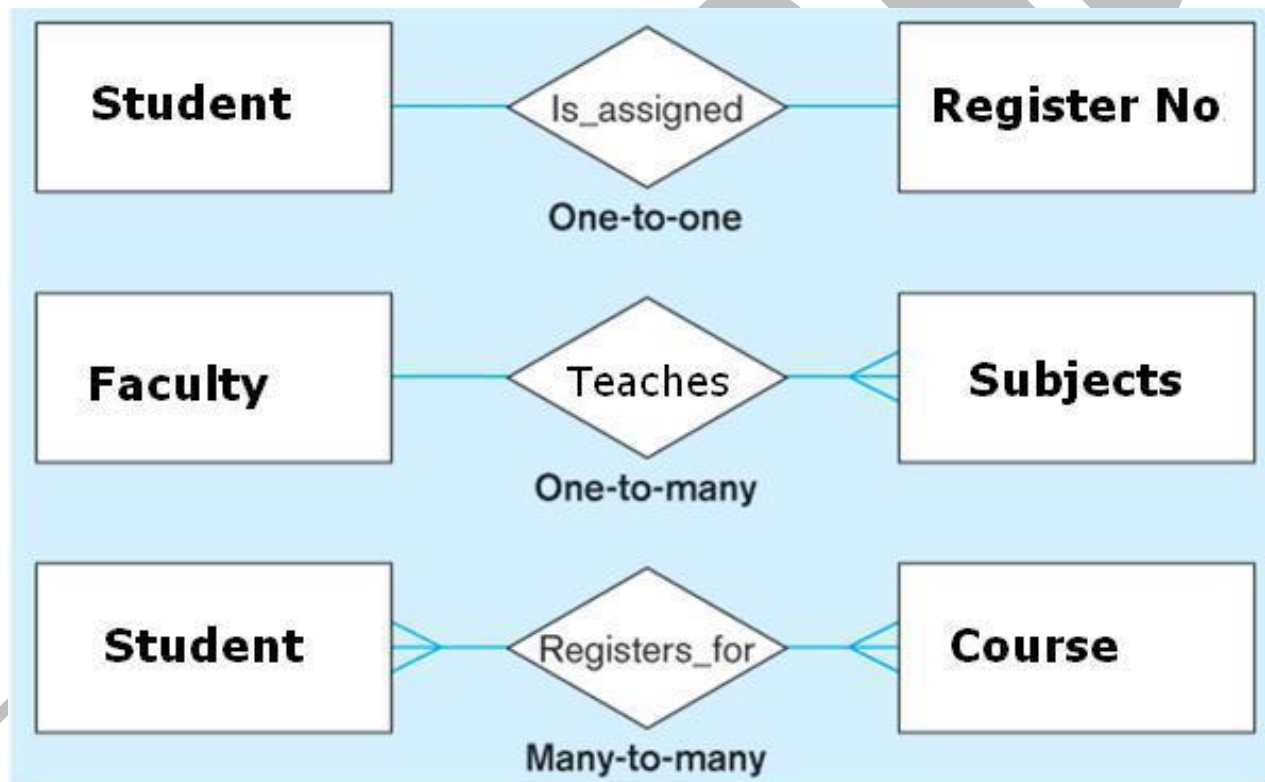
An Entity in „A“ is Associated with any number of entities in „B“ and an Entity in „B“ is associated with any number of entities in „A“.

Example:

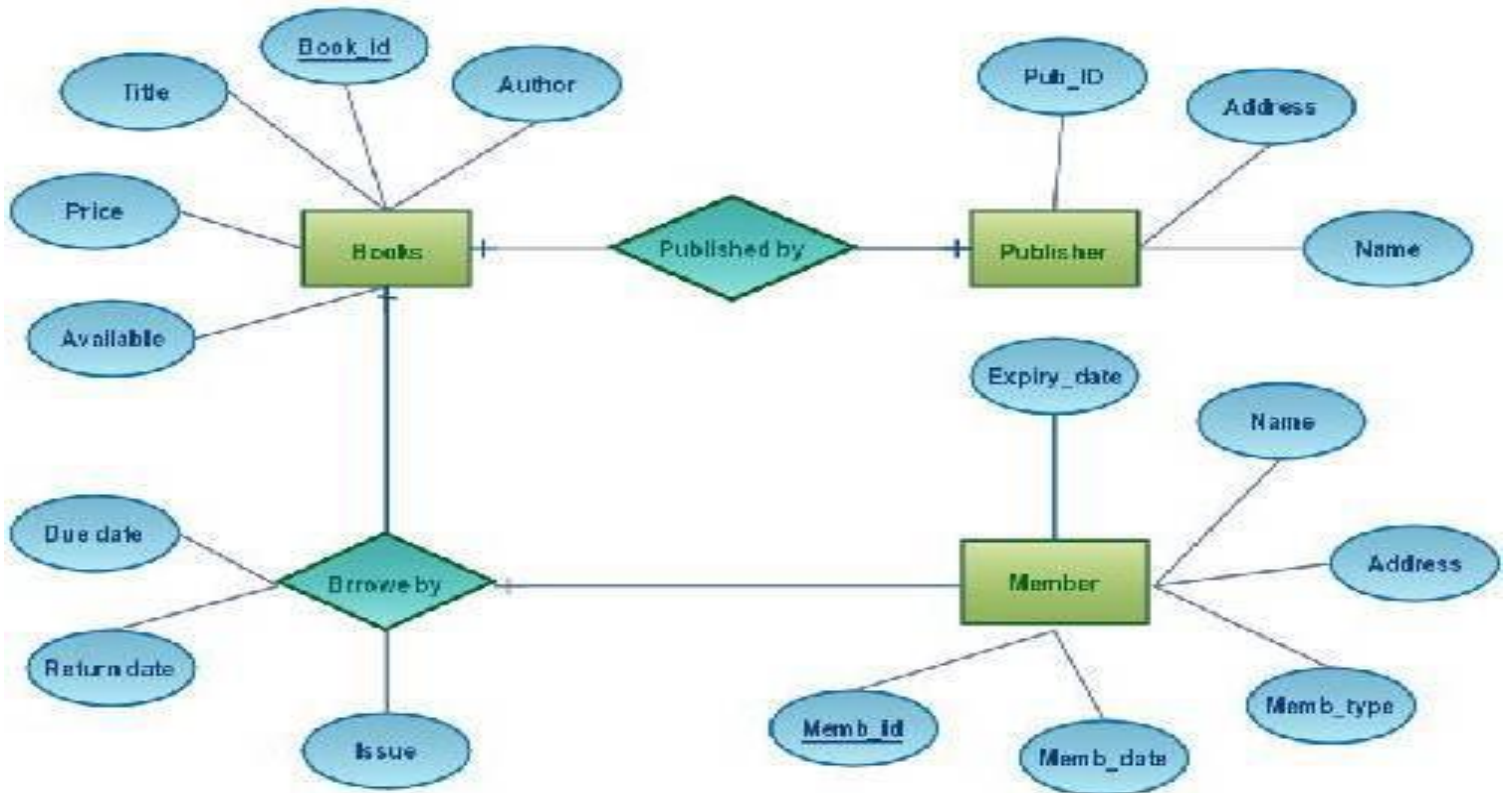
Eg.



Eg. Supplier to Purchaser



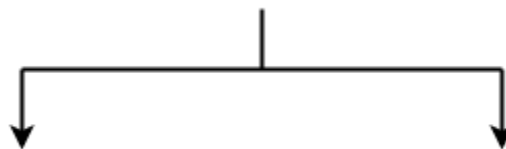
E-R Diagram for Library Management System



3.2.3 Strong entity, weak entity

Types of Entity Set:-

Entity Set



Strong Entity Set

Weak Entity Set

An entity set may be of the following two types-

1. Strong entity set
2. Weak entity set

1. Strong Entity Set-

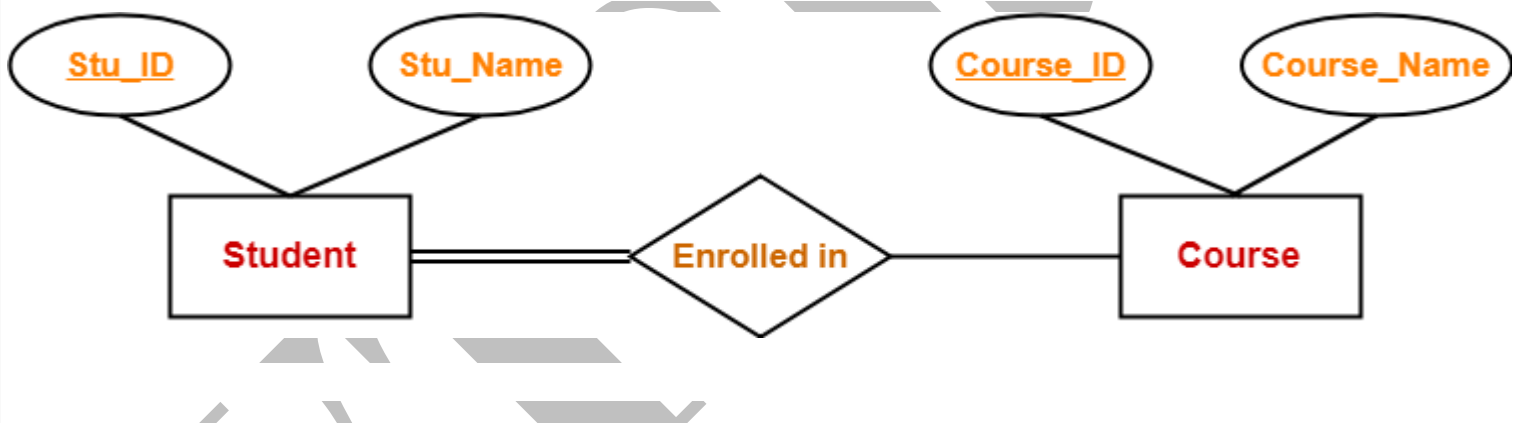
- A strong entity set is an entity set that contains sufficient attributes to uniquely identify all its entities.
- In other words, a primary key exists for a strong entity set.
- Primary key of a strong entity set is represented by underlining it.

Symbols Used-

- A single rectangle is used for representing a strong entity set.
- A diamond symbol is used for representing the relationship that exists between two strong entity sets.
- A single line is used for representing the connection of the strong entity set with the relationship set.
- A double line is used for representing the total participation of an entity set with the relationship set.
- Total participation may or may not exist in the relationship.

Example-

Consider the following ER diagram-



In this ER diagram,

- Two strong entity sets “**Student**” and “**Course**” are related to each other.
- Student ID and Student name are the attributes of entity set “Student”.
- Student ID is the primary key using which any student can be identified uniquely.
- Course ID and Course name are the attributes of entity set “Course”.
- Course ID is the primary key using which any course can be identified uniquely.

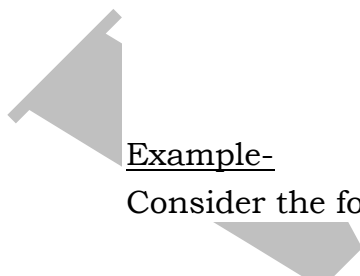
- Double line between Student and relationship set signifies total participation.
- It suggests that each student must be enrolled in at least one course.
- Single line between Course and relationship set signifies partial participation.
- It suggests that there might exist some courses for which no enrollments are made.

2. Weak Entity Set-

- A weak entity set is an entity set that does not contain sufficient attributes to uniquely identify its entities.
- In other words, a primary key does not exist for a weak entity set.
- However, it contains a partial key called as a **discriminator**.
- Discriminator can identify a group of entities from the entity set.
- Discriminator is represented by underlining with a dashed line.

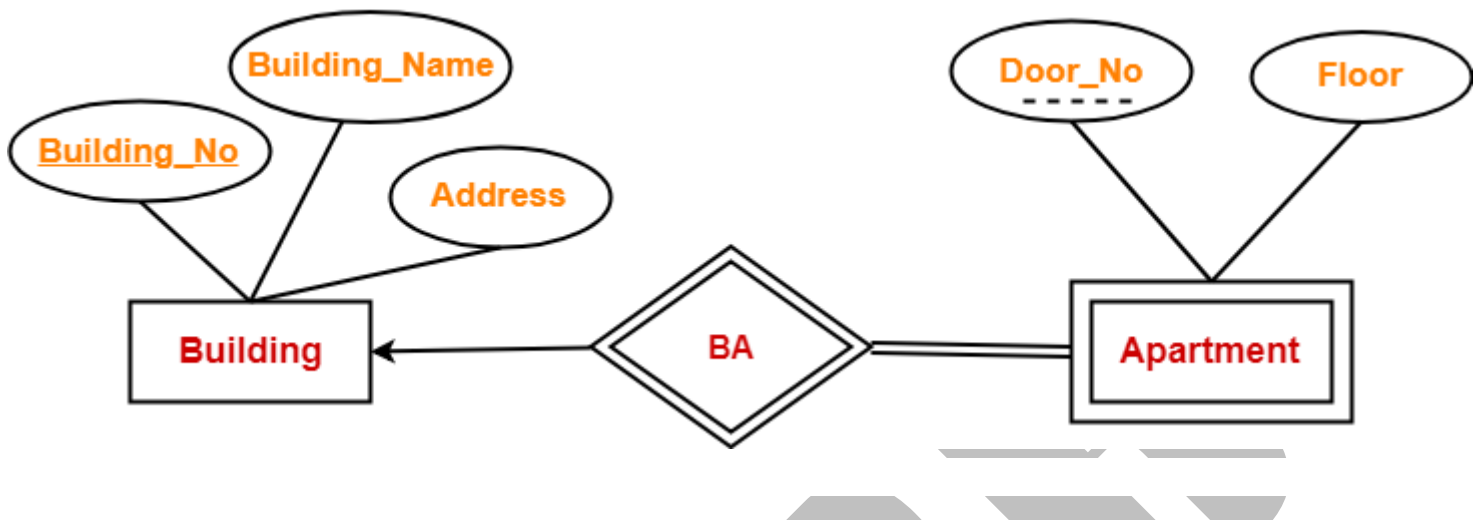
Symbols Used-

- A double rectangle is used for representing a weak entity set.
- A double diamond symbol is used for representing the relationship that exists between the strong and weak entity sets and this relationship is known as **identifying relationship**.
- A double line is used for representing the connection of the weak entity set with the relationship set.
- Total participation always exists in the identifying relationship.



Example-

Consider the following ER diagram-



In this ER diagram,

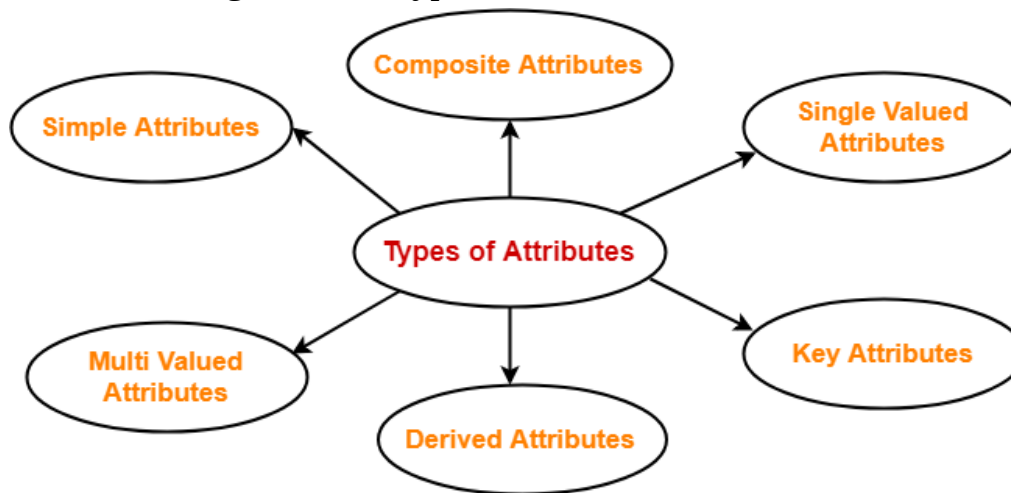
- One strong entity set “**Building**” and one weak entity set “**Apartment**” are related to each other.
- Strong entity set “Building” has building number as its primary key.
- Door number is the discriminator of the weak entity set “Apartment”.
- This is because door number alone can not identify an apartment uniquely as there may be several other buildings having the same door number.
- Double line between Apartment and relationship set signifies total participation.
- It suggests that each apartment must be present in at least one building.
- Single line between Building and relationship set signifies partial participation.
- It suggests that there might exist some buildings which has no apartment.

3.2.4 key attribute, derived attribute, Multi-valued attribute

- **Attributes:** - An Entity is represented by a set of attributes. Attributes are a Descriptive Properties of an entity. Each entity may have its own attributes.

Example: Attributes of Customer entity set are Customer_id, Customer_Name, Customer city, Customer_contactno., Customer_age etc.....

An attributes as used in the E-R diagram can be characterized by the following attribute types.



1. Simple and Composite attributes:

An attributes which are not divided into sub parts known as **simple attribute**. Example: Customer_id, rollno etc.

Composite attributes on other hand can be divided into sub parts.

Example: Address □□

Address can be sub divided in to Street , City , State , Pincode , etc ...

2. Single valued and multi- valued attributes :

Single value attributes have a single value for a particular entity.

EXAMPLE: Loan Number.

Where as multi value attributes have multiple value for a particular entity.

EXAMPLE: Phone Number.

3. Derived Attribute:

Value of this type of attribute is derived from the value of other related attributes or Entities. Suppose that is the customer Entity set has attribute Age that indicates customer's Age. If the customer entity Set also has an attribute date of birth, we can calculate Age from the DOB. Thus Age is a derived attribute.

4. Key Attribute:

Key Attributes are those attributes which can identify an entity uniquely in an entity set.

Example:-the attribute "Roll_no"(PK) is a key attribute as it can identify any student uniquely.

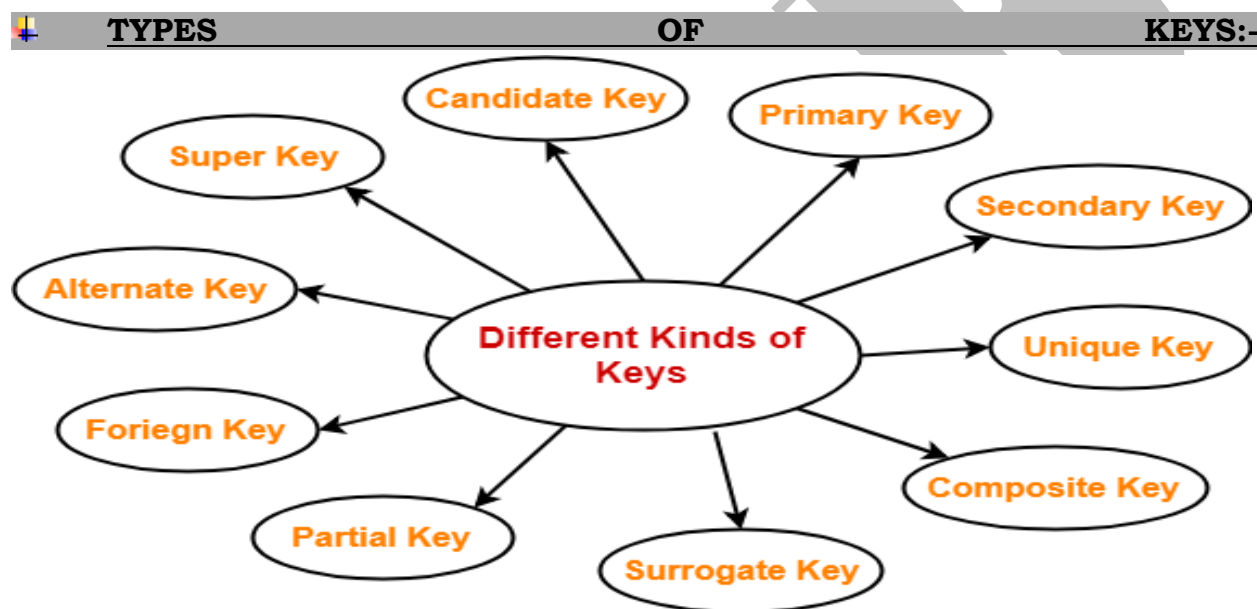
5. Null Attribute :

An attribute is used when Entity Does not have the value for an attribute that can accept null value. Here null means entire information is missing or unknown.

EXAMPLE: If particular value of attribute is unknown say contact_no.

3.3 TYPES OF KEYS :

3.3.1 Super key, candidate key, Primary key, Composite key, Foreign key, Unique key



A key is an attribute (also known as column or field) or a combination of attribute that is used to identify records. Sometimes we might have to retrieve data from more than one table, in those cases we require to join tables with the help of keys. The purpose of the key is to bind data together across tables without repeating all of the data in every table.

The various types of key with e.g. in SQL are mentioned below, (For examples let suppose we have an Employee Table with attributes 'ID' , 'Name' , 'Address' , 'Department_ID' , 'Salary')

❖ **Super key:-**

An attribute or a combination of attribute that is used to identify the records uniquely is known as Super Key. A table can have many Super Keys.

E.g. of Super Key

- 1 ID
- 2 ID, Name
- 3 ID, Address
- 4 ID, Department_ID
- 5 ID, Salary
- 6 Name, Address
- 7 Name, Address, Department_ID So on as any combination which can identify the records uniquely will be a Super Key.

❖ **Candidate key:-**

It can be defined as minimal Super Key or irreducible Super Key. In other words an attribute or a combination of attribute that identifies the record uniquely but none of its proper subsets can identify the records uniquely.

E.g. of Candidate Key

- 1 Code
- 2 Code,Name

For above table we have only two Candidate Keys (i.e. Irreducible Super Key) used to identify the records from the table uniquely. Code Key can identify the record uniquely and similarly combination of Code and Name can identify the record uniquely, but neither Code nor Name can be used to identify the records uniquely as it might be possible that we have two employees with similar name or two employees with the same name.

❖ **Primary key:-**

The PRIMARY KEY constraint uniquely identifies each record in a database table. Primary keys must contain UNIQUE values, and cannot contain NULL values. A table can have only one primary key, which may consist of single or multiple fields. When multiple fields are used as a primary key, they are called a composite key. If a table has a primary key defined on any field, then you cannot have two records having the same value of that field.

Syntax:

Primary Key at Column level

```
CREATE TABLE table_name  
(column1 datatype(size) Primary Key,  
column2 datatype(size),  
column3 datatype(size) );
```


Primary Key at Table level

```
CREATE TABLE table_name  
( column1 datatype(size),  
column2 datatype(size), column3 datatype(size),  
Primary key (Column_name) );
```

Example:

Primary Key at Column level

Create table Customer

```
(  
    ID Numeric Primary Key,  
    NAME varchar(10) NOT NULL,  
    AGE Numeric NOT NULL,  
    ADDRESS Varchar(30),  
    SALARY Numeric  
);
```

OR

Primary Key at Table level

Create table Customer

```
(  
    ID Numeric,  
    NAME varchar(10) NOT NULL,  
    AGE Numeric NOT NULL,  
    ADDRESS Varchar(30),  
    SALARY Numeric,  
    Primary key(ID)  
);
```

❖ **Composite key:-**

If we use multiple attributes to create a Primary Key then that Primary Key is called Composite Key (also called a Compound Key or Concatenated Key).

E.g. of Composite Key, if we have used “Name, Address” as a Primary Key then it will be our Composite Key.

❖ **Foreign key:-**

A FOREIGN KEY is a key used to link two tables together. A FOREIGN KEY is a field (or collection of fields) in one table that refers to the PRIMARY KEY in another table. The table containing the foreign key is called the child table, and the table containing the candidate key is called the referenced or parent table.

The relationship between 2 tables matches the Primary Key in one of the tables with a Foreign Key in the second table.

Person Table:

PERSONID	FIRSTNAME	LASTNAME	AGE
1	HETA	DESAI	22
2	MEHUL	NAIK	25
3	NIKITA	PATEL	27

Order Table

ORDERID	ORDERNUMBER	PERSONID
1	7752	3
2	8292	3
3	7884	1
4	4477	2

NOTE:

The "PersonID" column in the "Orders" table points to the "PersonID" column in the "Persons" table.

The "PersonID" column in the "Persons" table is the PRIMARY KEY in the "Persons" table.

The "PersonID" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.

The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.

The FOREIGN KEY constraint also prevents invalid data from being inserted into the foreign key column, because it has to be one of the values contained in the table it points to.

Syntax

```
CREATE TABLE Child_table_name  
(  
    column1 datatype(size),  
    column2 datatype(size),    column3  
    datatype(size),  
    Foreign key (Column_name) references Parent_table(Column_name)  
);
```

Example:

Parent table: (Person table) create table person

```
(  
    PersonId numeric primary key,  
    Firstname varchar(20),  
    Lastname varchar(20),  
    Age Numeric,  
    Address Varchar(25)  
);
```

Child table: (Order1 table) create table

```
Order1  
(  
    OrderId numeric primary key,  
    OrderNumber numeric, PersonId numeric, foreign  
    Key(PersonId) references Person(PersonId) );
```

Behavior of Foreign Key Constraint:

There are two ways to maintain the integrity of data in Child table, when a particular record is deleted in main table. When two tables are connected with Foreign key, and certain data in the main table is deleted, for which record exist in child table too, then we must have some mechanism to save the integrity of data in child table.

❖ **Unique key:-**

The UNIQUE Constraint prevents two records from having identical values in a column. The UNIQUE constraint ensures that all values in a column are different. Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. A PRIMARY KEY constraint automatically has a UNIQUE constraint. However, you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.

Syntax:

```
CREATE TABLE table_name  
( column1 datatype(size) UNIQUE,  
  column2 datatype(size),  
  column3 datatype(size) );
```

Example:

```
CREATE TABLE Persons  
( ID numeric NOT NULL UNIQUE,  
  LastName varchar(255) NOT NULL,  
  FirstName varchar(255),  
  Age numeric );
```