Database Management System (DBMS) (BCA-314)

Unit-1

Data: Data is a collection of information. So the facts that can be recorded and which have implicit meaning known as 'data'.

Ex: Customer -

1. c_name.
2. c_no.
3. c_city.

Database: Collection of interrelated data. These can be stored in the form of tables.

Ex: Customer database consists the fields as c_name, c_no, and c_city

c_name	**c_no**	**c_city**
Ram	**45662**	**XYZ**
Shyam	**78598**	**ABC**

DBMS stands for **Database Management System**. We can break it like this

DBMS = Database + Management System.

As we discuss that by data we mean useful information. In other ways Data as a general concept refers to the fact that have some accessible information or knowledge is represented or coded in some form suitable for better usage or processing.

Database is a collection of that data and Management System is a set of programs to store and retrieve those data. Based on this we can define DBMS like this:

So we can define database management system (DBMS) as a collection of inter-related data and set of programs to store & access those data in an easy and effective manner.

Need of DBMS

Database systems are basically developed for large amount of data. When dealing with huge amount of data, there are two things that require optimization: **Storage of data** and **retrieval of data**.

Storage: According to the principles of database systems, the data is stored in such a way that it acquires lot less space as the redundant data (duplicate data) has been removed before storage.

Let's take an example to understand this:
In a banking system, suppose a customer is having two accounts, one is saving account and another is salary account. Let's say bank stores saving account data at one place and salary account data at another place, in that case if the customer information such as customer name, address etc. are stored at both places then this is just a wastage of storage (redundancy/ duplication of data), to organize the data in a better way the information should be stored at one place and both the accounts should be linked to that information somehow. The same thing we achieve in DBMS.

Fast Retrieval of data: Along with storing the data in an optimized and systematic manner, it is also important that we retrieve the data quickly when needed. Database systems ensure that the data is retrieved as quickly as possible.

Purpose of Database Systems

The main purpose of database systems is to **manage the data**. Consider a university that keeps the data of students, teachers, courses, books etc. To manage this data we need to store this data somewhere where we can add new data, delete unused data, update outdated data, retrieve data, to perform these operations on data we need a Database management system that allows us to store the data in such a way so that all these operations can be performed on the data efficiently.

Database Applications – DBMS

Applications where we use Database Management Systems are:

- **Telecom**: There is a database to keeps track of the information regarding calls made, network usage, customer details etc. Without the database systems it is hard to maintain that huge amount of data that keeps updating every millisecond.
- **Industry**: Where it is a manufacturing unit, warehouse or distribution centre, each one needs a database to keep the records of ins and outs. For example distribution centre should keep a track of the product units that supplied into the centre as well as the products that got delivered out from the distribution centre on each day; this is where DBMS comes into picture.
- **Banking System**: For storing customer info, tracking day to day credit and debit transactions, generating bank statements etc. All this work has been done with the help of Database management systems.
- **Sales**: To store customer information, production information and invoice details.
- **Airlines**: To travel though airlines, we make early reservations, this reservation information along with flight schedule is stored in database.

- **Education sector**: Database systems are frequently used in schools and colleges to store and retrieve the data regarding student details, staff details, course details, exam details, payroll data, attendance details, fees details etc. There is a hell lot amount of inter-related data that needs to be stored and retrieved in an efficient manner.
- **Online shopping**: You must be aware of the online shopping websites such as Amazon, Flipkart etc. These sites store the product information, your addresses and preferences, credit details and provide you the relevant list of products based on your query. All this involves a Database management system.

Database System Vs File System

In traditional approach, before to computer, all information were stored in papers. When we need information, we used to search through the papers. After the invention of computer all data and information were stored in files.

File System

File processing systems was an early attempt to computerize the manual filing system. A file system is a method for storing and organizing computer files and the data they contain to make it easy to find and access them. File systems may use a storage device such as a hard disk.

So a file system is a process that manages how and where data on a storage disk, typically a hard disk drive (HDD), is stored, accessed and managed. It is a logical disk component that manages a disk's internal operations as it relates to a computer and is abstract to a human user.

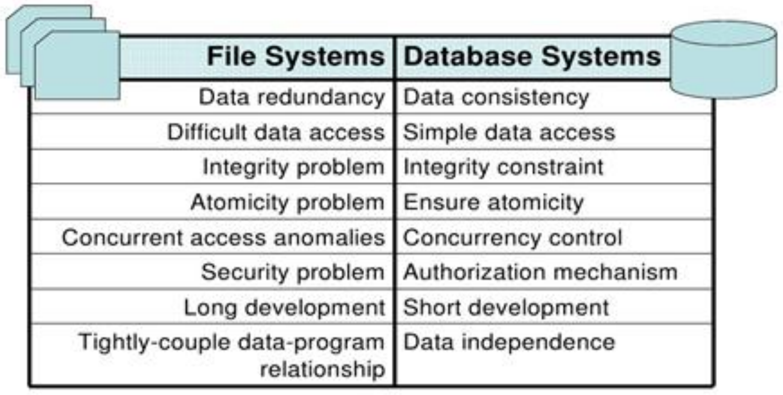
Regardless of type and usage, a disk contains a file system and information about where disk data is stored and how it may be accessed by a user or application. A file system typically manages operations, such as storage management, file naming, directories/folders, metadata, access rules and privileges.

Commonly used file systems include File Allocation Table 32 (FAT 32), New Technology File System (NTFS) and Hierarchical File System (HFS).

There are many different kinds of file systems. Each one has different structure and logic, properties of speed, flexibility, security, size and more. Some file systems have been designed to be used for specific applications. For example, the ISO 9660 file system is designed specifically for optical discs.

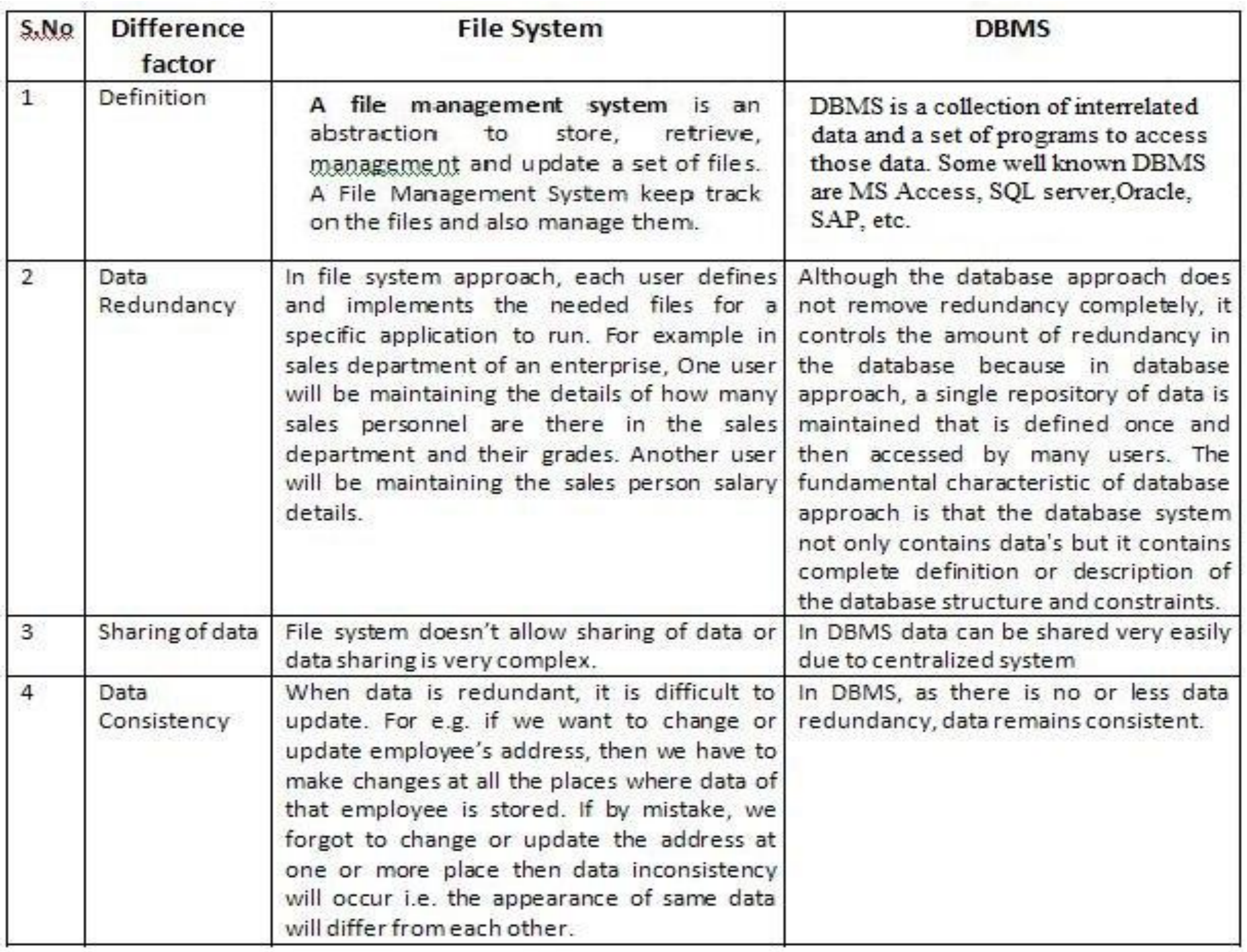
Advantage of DBMS over file system

There are several advantages of Database management system over file system. Few of them are as follows:



File Systems	Database Systems
Data redundancy	Data consistency
Difficult data access	Simple data access
Integrity problem	Integrity constraint
Atomicity problem	Ensure atomicity
Concurrent access anomalies	Concurrency control
Security problem	Authorization mechanism
Long development	Short development
Tightly-couple data-program relationship	Data independence

Now we will describe in detail how DBMS is different from traditional file system and what ware the limitations of file system which has covered by DBMS system.



S.No	Difference factor	File System	DBMS
1	Definition	**A file management system** is an abstraction to store, retrieve, management and update a set of files. A File Management System keep track on the files and also manage them.	DBMS is a collection of interrelated data and a set of programs to access those data. Some well known DBMS are MS Access, SQL server,Oracle, SAP, etc.
2	Data Redundancy	In file system approach, each user defines and implements the needed files for a specific application to run. For example in sales department of an enterprise, One user will be maintaining the details of how many sales personnel are there in the sales department and their grades. Another user will be maintaining the sales person salary details.	Although the database approach does not remove redundancy completely, it controls the amount of redundancy in the database because in database approach, a single repository of data is maintained that is defined once and then accessed by many users. The fundamental characteristic of database approach is that the database system not only contains data's but it contains complete definition or description of the database structure and constraints.
3	Sharing of data	File system doesn't allow sharing of data or data sharing is very complex.	In DBMS data can be shared very easily due to centralized system
4	Data Consistency	When data is redundant, it is difficult to update. For e.g. if we want to change or update employee's address, then we have to make changes at all the places where data of that employee is stored. If by mistake, we forgot to change or update the address at one or more place then data inconsistency will occur i.e. the appearance of same data will differ from each other.	In DBMS, as there is no or less data redundancy, data remains consistent.

5	Difficult to search/access data	In conventional file system, if we want to search/retrieve/access some data item, it becomes very difficult because in file system for every operation we have to we have to make different programs.	In DBMS searching/retrieval/accessing of data item is very easy and user-friendly because searching and querying operations are already available in the system.
6	Data isolation	In file system there is no standard format of data or we can say data is scattered in various formats or files which also make data retrieval difficult.	In DBMS, due to centralized system the format of similar type of data remains same.
7	Data Integrity	The value of data in database must follow or satisfy some rules or consistency constraints. For e.g. A company have a policy that the age of an employee must be >=18. The value which is not satisfying this constraint must not be stored in the respective column. 8In file system, there is no procedure to check these constraints automatically.	DBMS maintains the data integrity by enforcing the constraints by adding appropriate code.
8	Security problems	In file system there is no or very less security. General security provided by file system are locks, guards etc.	DBMS have high level security like encryption, passwords, biometric security (fingerprint matching, face and voice detection etc) etc.
9	Atomicity	Atomicity means a transaction must be all-or-nothing i.e. the transaction must either fully happen, or not happen at all. It must not complete partially. E.g. if A want to transfer 5000rs to B's a/c. In this case A's a/c should be debited and B's a/c should be credited with the same amount. Let suppose A's a/c is debited with 5000rs and then transaction fails. Now the transaction is incomplete because B's a/c is not credited. These type of problems occur in file system because there is no procedure to stop such type of anomalies.	Transaction atomicity is a special feature of DBMS. In DBMS either a transaction completed fully or none of the action is performed. For this, DBMS maintains the transaction log in which intermediate values are stored.
10	Concurrent Access Anomalies	Any multi-user database application has to have some method for dealing with concurrent access to data -- when more than one user is accessing the same data at the same time. A problem occurs when user X reads a row for editing, user Y reads the same row for editing, user Y saves changes, then user X saves changes. The changes made by user Y are lost unless something prevents user X from blindly overwriting the row. File system does not provide any procedure to stop such type of anomalies.	DBMS along with an appropriate application provides safety towards concurrent access. For this locks are available in DBMS. If 2 or more transactions want to change/update or write a data item, an exclusive lock is issued to one of these transactions. Until and unless the transaction release that lock no other transaction can acquire the lock and hence cannot update/write the data item.

DBMS Architecture

Database management systems architecture will help us understand the components of database system and the relation among them.
The architecture of DBMS depends on the computer system on which it runs. For example, in a client-server DBMS architecture, the database systems at server machine can run several requests made by client machine.

Types of DBMS Architecture

There are three types of DBMS architecture:

1. Single tier architecture
2. Two tier architecture
3. Three tier architecture

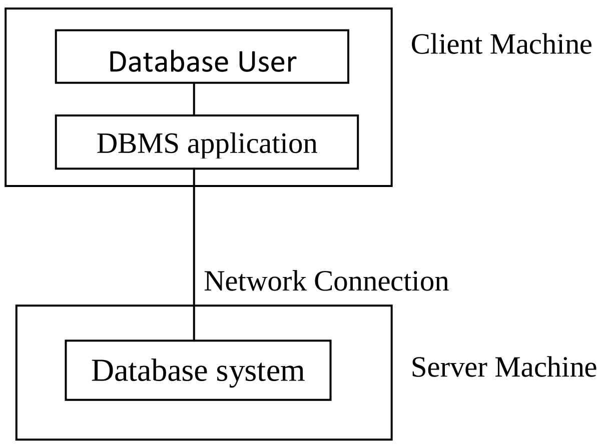
1. Single tier architecture

In this type of architecture, the database is readily available on the client machine, any request made by client doesn't require a network connection to perform the action on the database.

For example, lets say you want to fetch the records of employee from the database and the database is available on your computer system, so the request to fetch employee details will be done by your computer and the records will be fetched from the database by your computer as well. This type of system is generally referred as local database system.

2. Two tier architecture

In two-tier architecture, the Database system is present at the server machine and the DBMS application is present at the client machine, these two machines are connected with each other through a reliable network as shown in the above diagram.

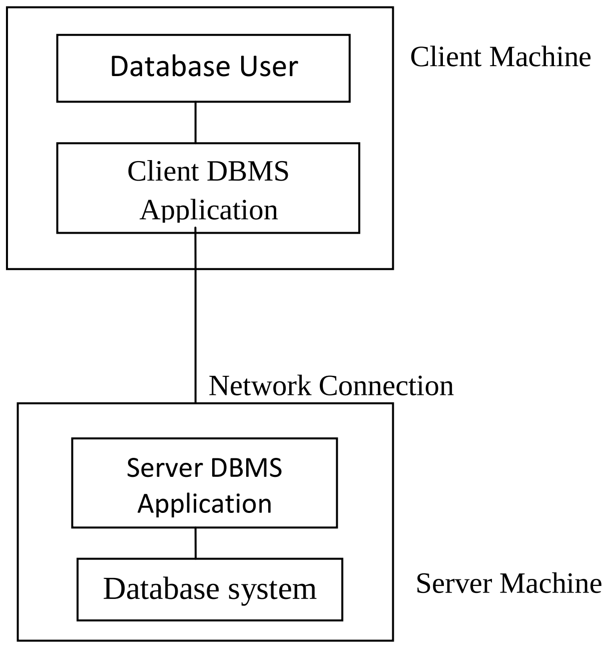


Two-Tier Architecture

Whenever client machine makes a request to access the database present at server using a query language like sql, the server perform the request on the database and returns the result back to the client. The application connection interface such as JDBC, ODBC are used for the interaction between server and client.

3. Three tier architecture

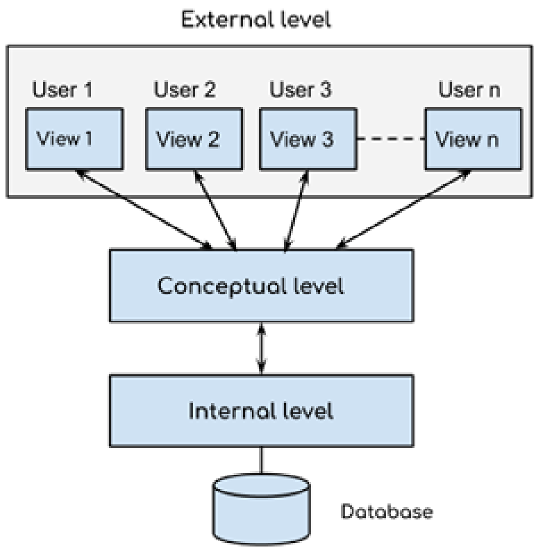
In three-tier architecture, another layer is present between the client machine and server machine. In this architecture, the client application doesn't communicate directly with the database systems present at the server machine, rather the client application communicates with server application and the server application internally communicates with the database system present at the server



Three-Tier Architecture

DBMS – Three Level Architecture

DBMS Three Level Architecture Diagram
This architecture has three levels:
1.External level
2.Conceptual level
3.Internal level



1. External level

It is also called **view level**. The reason this level is called "view" is because several users can view their desired data from this level which is internally fetched from database with the help of conceptual and internal level mapping.
External level is the "**top level**" of the Three Level DBMS Architecture.

2. Conceptual level

It is also called **logical level**. The whole design of the database such as relationship among data, schema of data etc. are described in this level. Database constraints and security are also implemented in this level of architecture.

3. Internal level

This level is also known as physical level. This level describes how the data is actually stored in the storage devices. This level is also responsible for allocating space to the data. This is the lowest level of the architecture.

Database system concepts

<u>DATA Model</u>

Data Model is a logical structure of Database. **Data models define how data is connected** to each other and how they are processed and stored inside the system.

Categories of Data Models

High-level or conceptual data models- It provide concepts that are close to the way many users perceive data.

Low-level or physical data models- It provide concepts that describe the details of how data is stored on the computer storage media.

Concepts provided by low-level data models are generally meant for computer specialists, not for end users. Between these two extremes is a class of **representational (or implementation) data models**, which provide concepts that may be easily understood by end users.

Conceptual data models use concepts such as entities, attributes, and relationships.

<u>Entity</u>- An **entity** represents a real-world object or concept, such as an employee or a project from the miniworld that is described in the database.

<u>Attribute</u>- An **attribute** represents some property of an entity, such as the employee's name or salary.

<u>Relationship-</u> A relationship among two or more entities represents an association among the entities, for example, a works-on relationship between an employee and a project.

Types of Data Models

There are several types of data models in DBMS.

Object based logical Models-Describe data at the conceptual and view levels.

1. E-R Model
2. Object oriented Model

Record based logical Models

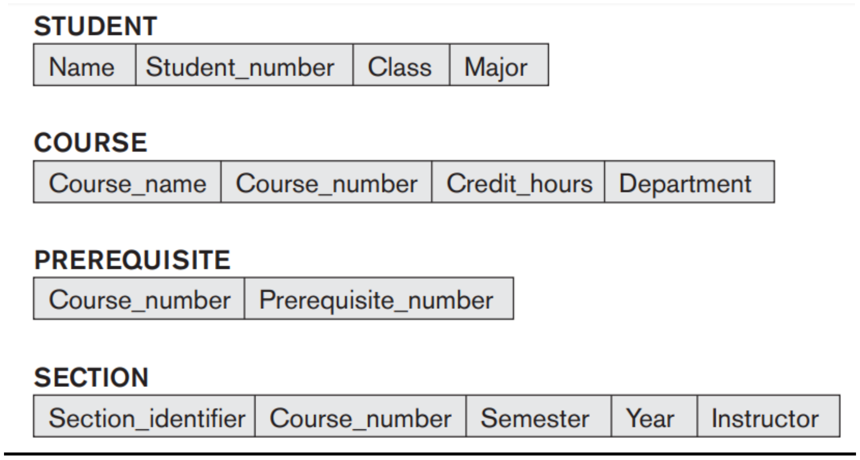
1. Relational Model
2. Hierarchical Model
3. Network Model

Physical Data Models

Data Abstraction - Abstraction is one of the main features of database systems. Hiding irrelevant details from user and providing abstract view of data to users, helps in easy and efficient **user-database** interaction. This process of hiding irrelevant details from user is called data abstraction.

Example: Let's say we are storing customer information in a customer table. At **physical level** these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are often hidden from the programmers.

Schemas : The description of a database is called the database schema, which is specified during database design and is not expected to change frequently. Most data models have certain conventions for displaying schemas as diagrams.A displayed schema is called a schema diagram.



STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

fig-schema

DBMS Instance

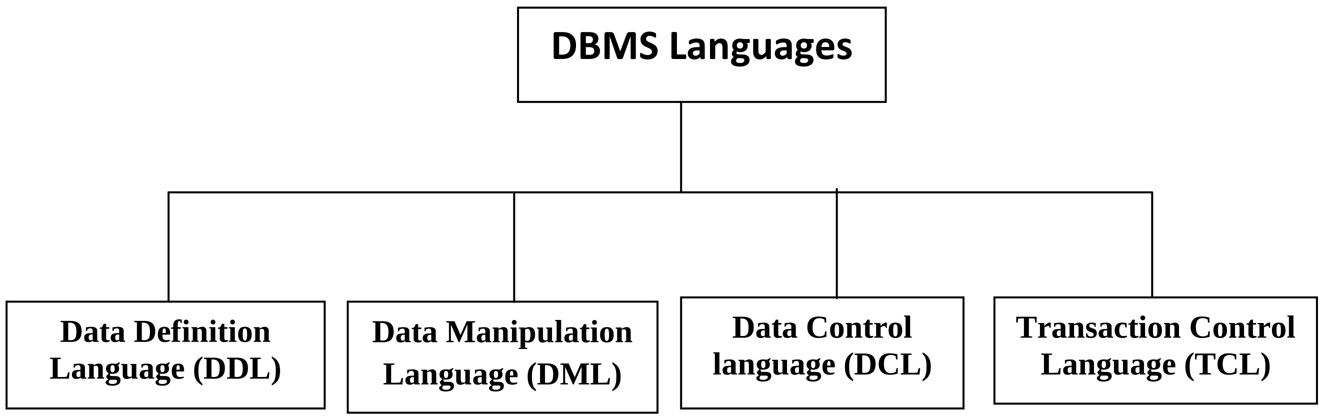
The data stored in database at a particular moment of time is called instance of database. The actual data in a database may change quite frequently.

For example, lets say we have a single table student in the database, today the table has 100 records, so today the instance of the database has 100 records. Lets say we are going to add another 100 records in this table by tomorrow so the instance of database tomorrow will have 200 records in table. In short, at a particular moment the data stored in database is called the instance, that changes over time when we add or delete data from the database.

DBMS languages

Database languages are used to read, update and store data in a database. There are several such languages that can be used for this purpose; one of them is SQL (Structured Query Language).

Types of DBMS languages:


DBMS Languages
Data Definition Language (DDL)
Data Manipulation Language (DML)
Data Control language (DCL)
Transaction Control Language (TCL)

Data Definition Language (DDL)

DDL is used for specifying the database schema. It is used for creating tables, schema, indexes, constraints etc. in database. Lets see the operations that we can perform on database using DDL:

- To create the database instance – CREATE
- To alter the structure of database – **ALTER**
- To drop database instances – DROP
- To delete tables in a database instance – **TRUNCATE**
- To rename database instances – **RENAME**
- To drop objects from database such as tables – **DROP**
- To Comment – **Comment**

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

CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),

 City varchar(255)
);
```

All of these commands either defines or update the database schema that’s why they come under Data Definition language.

Data Manipulation Language (DML)

DML is used for accessing and manipulating data in a database. The following operations on database comes under DML:

- To read records from table(s) – SELECT
- To insert record(s) into the table(s) – **INSERT**
- Update the data in table(s) – UPDATE
- Delete all the records from the table – DELETE

```
INSERT INTO table_name (column1, column2, column3, ...)
VALUES (value1, value2, value3, ...);
```

```
INSERT INTO student (stu_name,    stu_rollno, stu_course ...)
VALUES (ram, 19005221, BCA, ...);
```

```
SELECT column1, column2, ...
FROM table_name;
```

```
SELECT CustomerName, City FROM Customers;
```

Data Control language (DCL)

DCL is used for granting and revoking user access on a database –

- To grant access to user – GRANT
- To revoke access from user – REVOKE

Transaction Control Language (TCL)

The changes in the database that we made using DML commands are either performed or rollbacked using TCL.

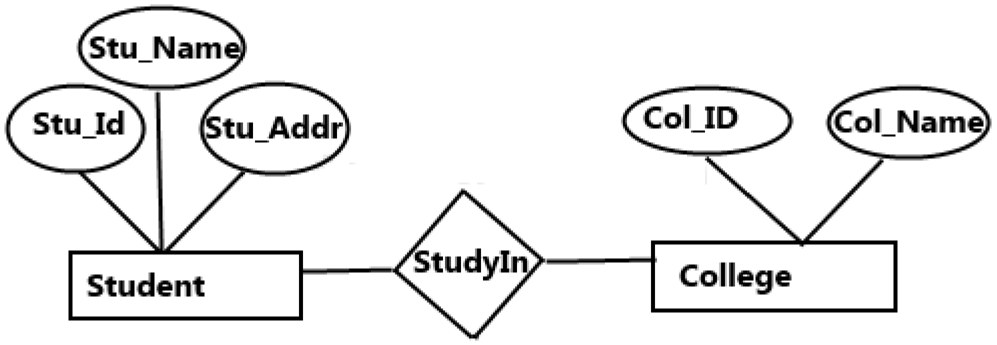
- To persist the changes made by DML commands in database – COMMIT
- To rollback the changes made to the database – ROLLBACK

Entity Relationship Diagram – ER Diagram in DBMS

An **Entity–relationship model (ER model)** describes the structure of a database with the help of a diagram, which is known as **Entity Relationship Diagram (ER Diagram)**. An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

A simple ER Diagram:

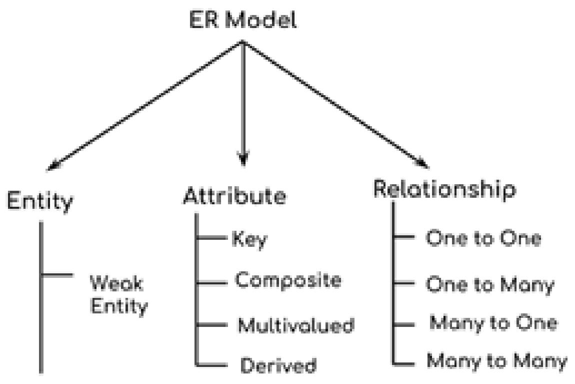
In the following diagram we have two entities Student and College and their relationship. The relationship between Student and College is many to one as a college can have many students however a student cannot study in multiple colleges at the same time. Student entity has attributes such as Stu_Id, Stu_Name & Stu_Addr and College entity has attributes such as Col_ID & Col_Name.



Here are the geometric shapes and their meaning in an E-R Diagram.

Rectangle: Represents Entity sets.
Ellipses: Attributes
Diamonds: Relationship Set
Lines: They link attributes to Entity Sets and Entity sets to Relationship Set
Double Ellipses: Multivalued Attributes
Dashed Ellipses: Derived Attributes
Double Rectangles: Weak Entity Sets
Double Lines: Total participation of an entity in a relationship set

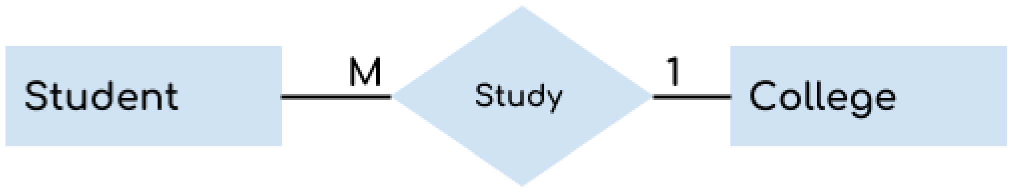
Components of a ER Diagram



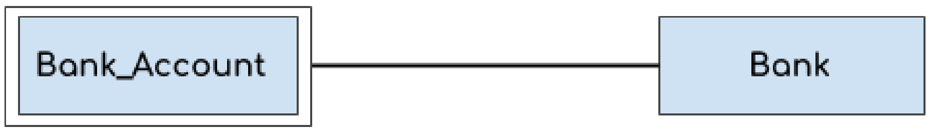
Components of ER Diagram

1. Entity

An entity is an object or component of data. An entity is represented as rectangle in an ER diagram. For example: In the following ER diagram we have two entities Student and College and these two entities have many to one relationship as many students study in a single college.



Weak Entity:
An entity that cannot be uniquely identified by its own attributes and relies on the relationship with other entity is called weak entity. The weak entity is represented by a double rectangle. For example – a bank account cannot be uniquely identified without knowing the bank to which the account belongs, so bank account is a weak entity.



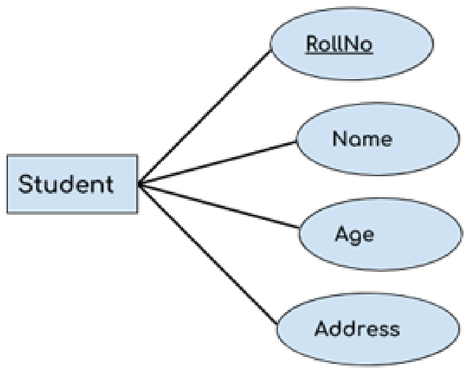
2. Attribute

An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. There are four types of attributes:

1. Key attribute
2. Composite attribute
3. Multivalued attribute
4. Derived attribute

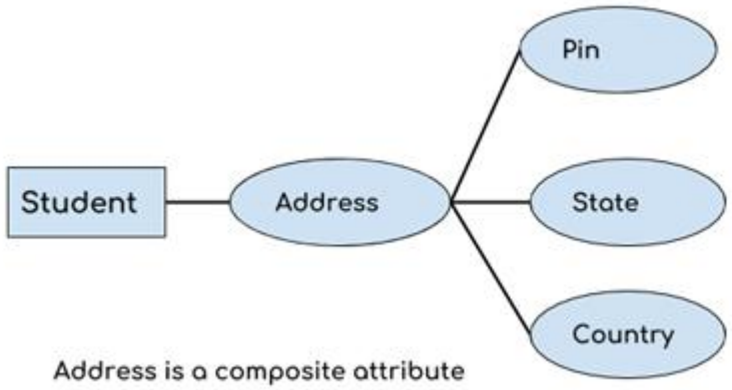
1. Key attribute:

A key attribute can uniquely identify an entity from an entity set. For example, student roll number can uniquely identify a student from a set of students. Key attribute is represented by oval same as other attributes however the **text of key attribute is underlined**.



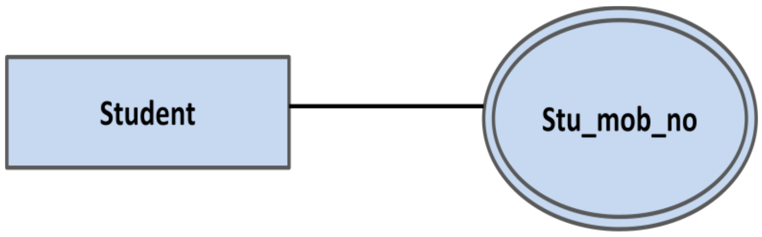
2. Composite attribute:

An attribute that is a combination of other attributes is known as composite attribute. For example, In student entity, the student address is a composite attribute as an address is composed of other attributes such as pin code, state, country.



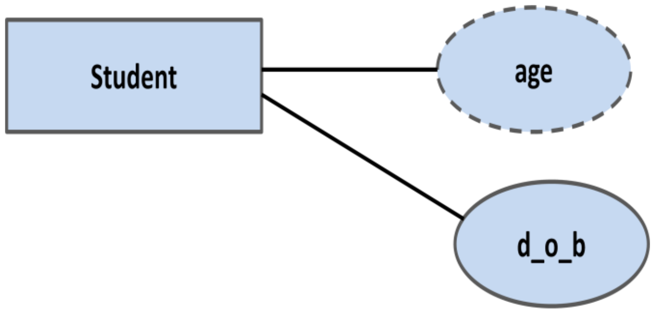
3. Multivalued attribute:

An attribute that can hold multiple values is known as multivalued attribute. It is represented with **double ovals** in an ER Diagram. For example – A person can have more than one phone numbers so the phone number attribute is multivalued.

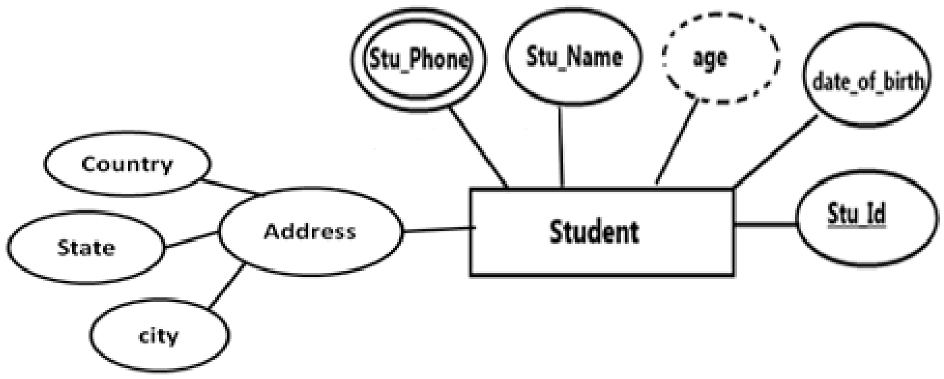


4. Derived attribute:

A derived attribute is one whose value is dynamic and derived from another attribute. It is represented by **dashed oval** in an ER Diagram. For example – Person age is a derived attribute as it changes over time and can be derived from another attribute (Date of birth).



E-R diagram with key, composite, multivalued and derived attributes:

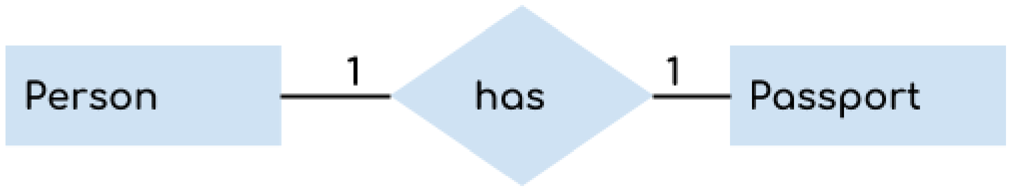


3. Relationship

A relationship is represented by diamond shape in ER diagram, it shows the relationship among entities. There are four types of relationships:
1. One to One
2. One to Many
3. Many to One
4. Many to Many

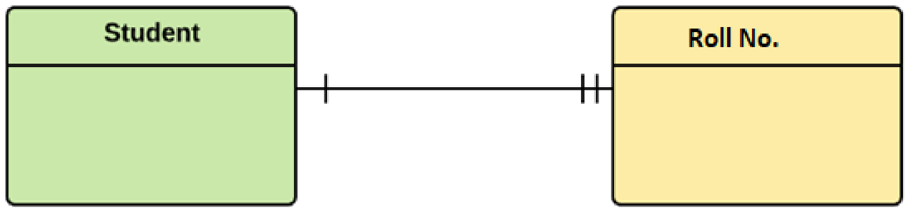
1. One to One Relationship

When a single instance of an entity is associated with a single instance of another entity then it is called one to one relationship. For example, a person has only one passport and a passport is given to one person.



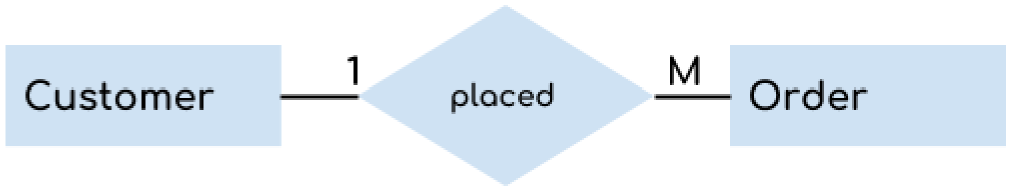
One entity from entity set X can be associated with at most one entity of entity set Y and vice versa.

Example: One student can have one roll no. And one roll no can be assigned to one student.



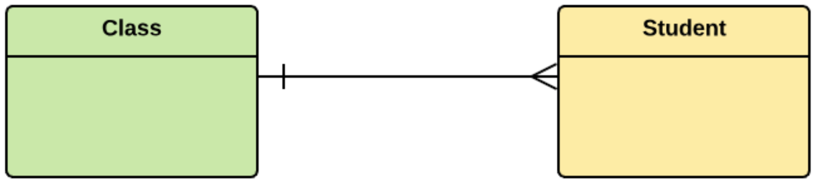
2. One to Many Relationship

When a single instance of an entity is associated with more than one instances of another entity then it is called one to many relationship. For example – a customer can place many orders but a order cannot be placed by many customers.



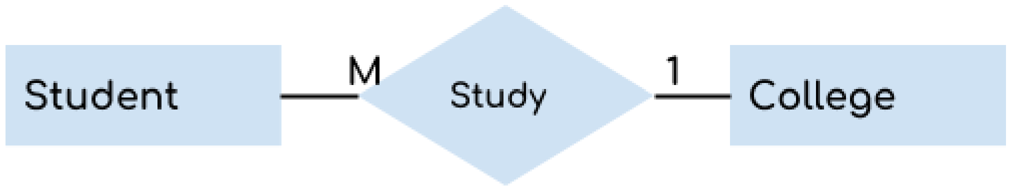
One entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity.

For example, one class is consisting of multiple students and multiple students may have in a class.



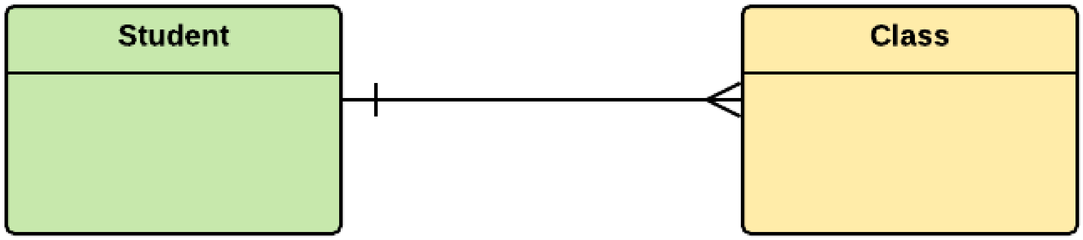
3. Many to One Relationship

When more than one instances of an entity is associated with a single instance of another entity then it is called many to one relationship. For example – many students can study in a single college but a student cannot study in many colleges at the same time.



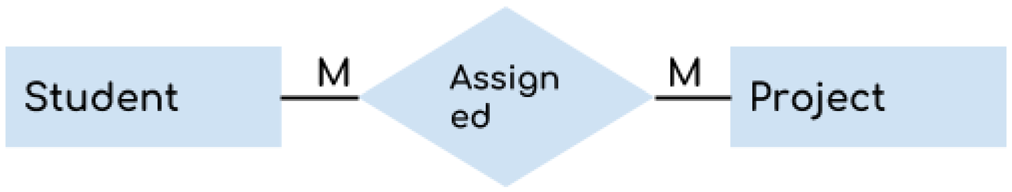
More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set Y may or may not be associated with more than one entity from entity set X.

For example, many students belong to the same class.



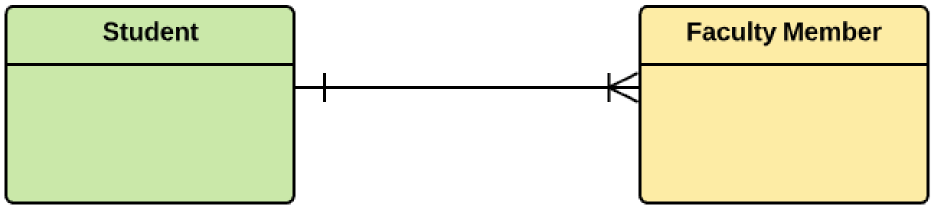
4. Many to Many Relationship

When more than one instances of an entity is associated with more than one instances of another entity then it is called many to many relationship. For example, a student can be assigned to many projects and a project can be assigned to many students.

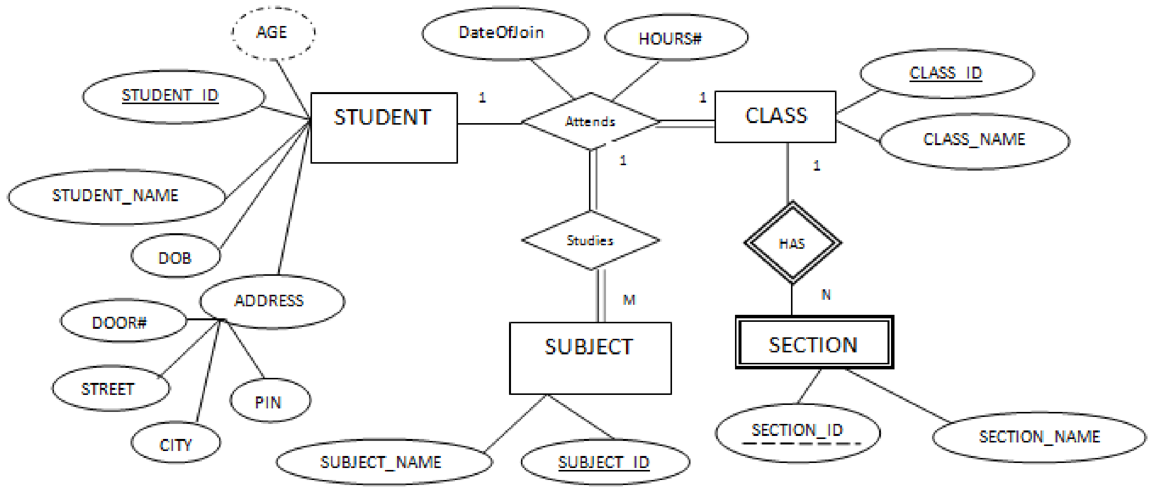


More than one entity from X can be associated with more than one entity from Y and vice versa.

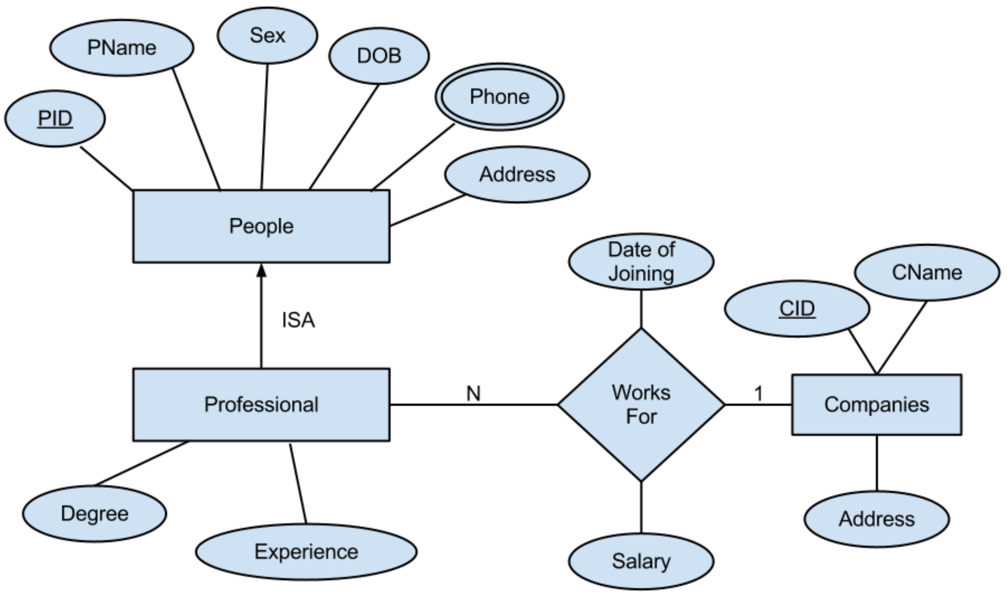
For example, Students as a group are associated with multiple faculty members, and faculty members can be associated with multiple students.



ER diagram for College Database

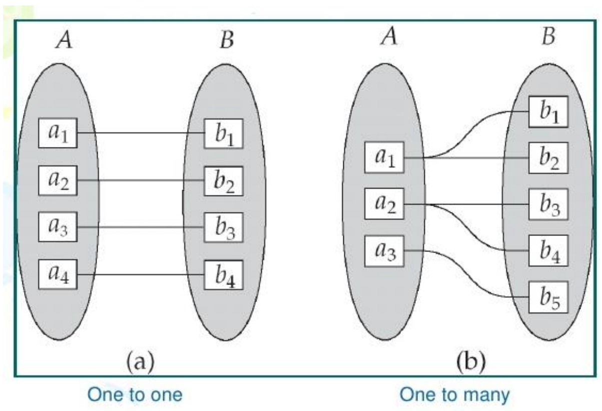


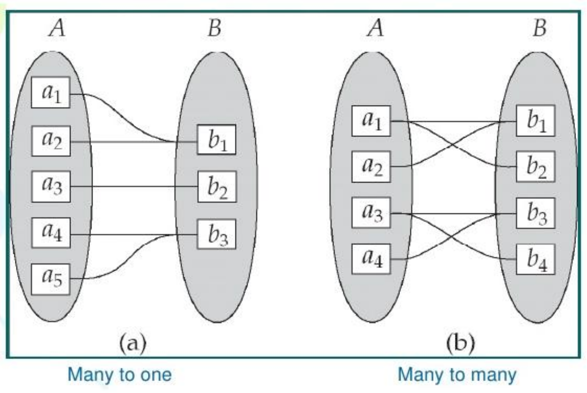
ER diagram for Company Database



Mapping Constraints

- A mapping constraint is a data constraint that expresses the number of entities to which another entity can be related via a relationship set.
- It is most useful in describing the relationship sets that involve more than two entity sets.
- For binary relationship set R on an entity set A and B, there are four possible mapping cardinalities. These are as follows:
    - One to one (1:1)
    - One to many (1:M)
    - Many to one (M:1)
    - Many to many (M:M)





- <u>One to One:</u> An entity of entity-set A can be associated with at most one entity of entity-set B and an entity in entity-set B can be associated with at most one entity of entity-set A.
- <u>One to Many</u>: An entity of entity-set A can be associated with any number of entities of entity-set B and an entity in entity-set B can be associated with at most one entity of entity-set A.
- <u>Many to One:</u> An entity of entity-set A can be associated with at most one entity of entity-set B and an entity in entity-set B can be associated with any number of entities of entity-set A.
- <u>Many to Many</u>: An entity of entity-set A can be associated with any number of entities of entity-set B and an entity in entity-set B can be associated with any number of entities of entity-set A.