

1 Dec 2022 DBMS

CRUD - create Read update Delete Execute

Defⁿ - Database

* Intro -

- The Database management system is ^{Defining} required as software system that allows users to define, create, manage and control access to the database.

- DBMS makes it possible for end users to create, read, update, delete in database.

Advantages - DBMS

① Reducing data Redundancy (Repetition)

meaning - The File base DBMS contained multiple files that were stored in many different locations in a system.

→ because of this there must work sometimes multiple copies of same file which lead to data Redundancy.

→ This is prevented in Database as there is a single database and any change in it is reflected immediately.

→ Because of this there is no change or encounter duplicate data.

② Sharing of data

→ In a database, the users of the database can share the data among themselves.

→ There are various levels of authorization to access the data and consequently the data can only be shared based on the correct authorization, protocols to be followed.

no cc

motherboard

magnetic disk

handel - change, as

access

③ Data integrity -

→ It means that the data is accurate and consistent in the database.

→ It is very important as there are multiple databases in a DBMS and it is visible to multiple users.

→ So, it is necessary to ensure that data is correct in all the databases and for all the users.

2 Dec 2022

④ Data Security -

- It is a vital (important) concept in a database.

- Only authorized users should be allowed to access the database and their identity should be authenticated using the username and password.

⑤ Privacy -

⑥ Backup and recovery

- DBMS automatically takes care of the backup and recovery.

- The users don't need to backup data periodically but because this is taken care of by DBMS and it also restores the database after a crash or a system failure.

hiding of data

* Abstraction in DBMS -

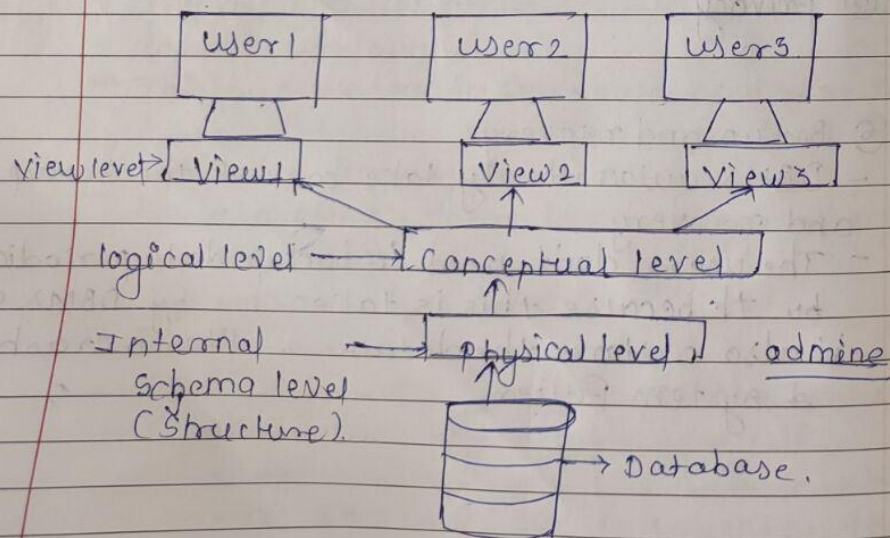
- It refers to the process of hiding irrelevant details from the user.

For ex:-

- If we want to access any mail from our gmail there we don't know where the data is physically stored, that is the data is present in India or USA.
- We are not concerned about this things and hence location of data and data models are irrelevant to us, and in data abstraction we do this only.

There are 3 Levels of data Abstraction,

- 1) View level
- 2) Conceptual
- 3) physical level



1) View level :-

- This level tells the application how the data should be shown to the user.

eg:-

- If we have a login id and password in a University system, then as a student, we can view our marks, fee structure etc.
- but the Faculty of University have different view. We will have options like edit marks or student enter attendance etc.
- So both the student and Faculty have a different view, and by view the security and privacy of the system is maintained.

2) Conceptual level (logical)

create
update

- It tells how the data is actually stored and structured

eg:-

- If we use the relational model for storing the data we have to store the data of the student using columns in the student table naming it as stu name, age, roll no, etc.
- We have to define all this at level by creating the database hence the blueprint is created at conceptual level.

3 Dec 2022

3) physical level -

Data
Base
admin

- The physical level tells us that where the data is actually stored
- The DBA decide that which data should be kept at which particular disk drive, how the data has to be fragmented, where it has to be stored etc.

- They also decide if data has to be centralised or distributed hence it depends on a DBA how he or she manages the database at the physical level.

* DBMS Architecture

1 tier
2 tier
3 tier

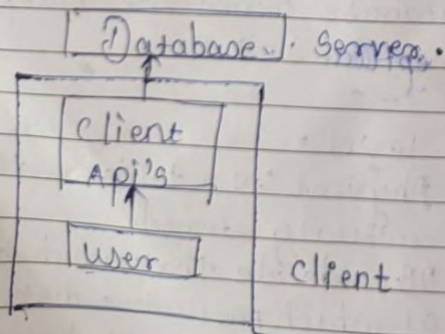
- It depends on how users are connected to the database to get their request done.
- There are 3 types of DBMS Architecture

① 1 tier.

- The database is directly available through the user.
- It means that user can be directly handle the DBMS and used it.
- It is use for development of the local application, where programmers can directly communicate with the database for quick response.

② 2 tier

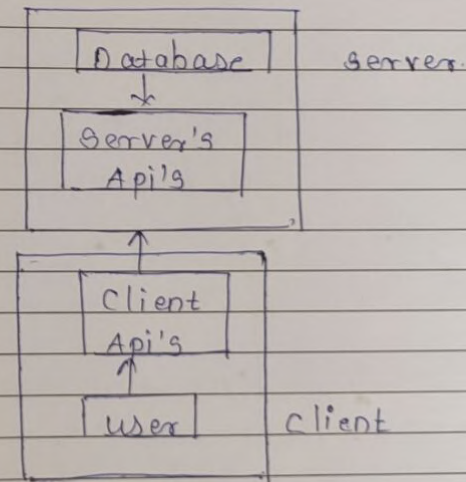
Diagram -



- It is same as the basic client server Architecture.
- Here the ^{application} client & directly communicate with the data at the server side. For this interaction Api's like ^{SQL} ~~SQL~~, Oracle, ODBC, JDBC.
- The server side is responsible to provide the functionalities like ^{query} ~~query~~ processing & transaction management.

③ 3 tier -

Diagram -



- It contains another layer between the client and server.
- In this architecture, client can not be directly communicate with server.
- end user has not idea about the existence of the database beyond the application server.
- The 3 tier architecture is used in case of large web application.

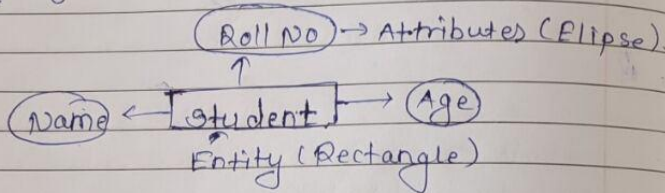
chapter-2

3 Dec 2022

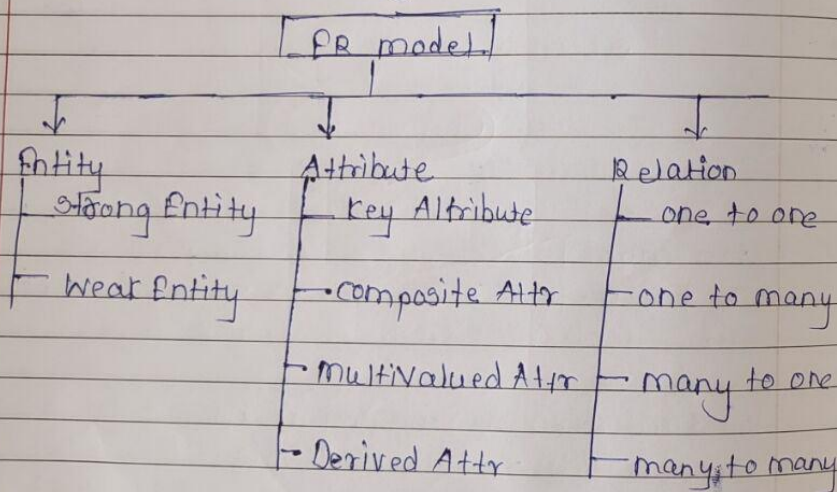
Entity-Relationship Model (ER Model)

- It is a high level data model
- This model is use to define data elements & relationship for a specified system.
- In ER modeling, the database structure is modeled as a diagram called an ER diagram

Eg:-



* Components of ER diagram.



* Entity -

- It is an object, class, person or place
- It can be represented in rectangles.

There are 2 types of Entity

① Weak Entity

depends on another entity

eg - tax money and trust

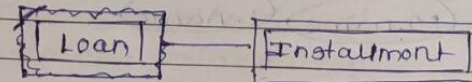
fan, cor- entity and object

BARFI Page No. Date

BARFI Page No. Date

- An entity is depends on the another entity is called as weak entity. It doesn't contain any key Attributes on it own.
- It is represented using double rectangle.

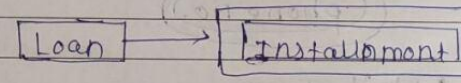
eg:-



② Strong entity.

- An entity is doesn't depends on the another entity is called as strong entity.
- It contain key Attributes on it own.
- It is represented using single rectangle.

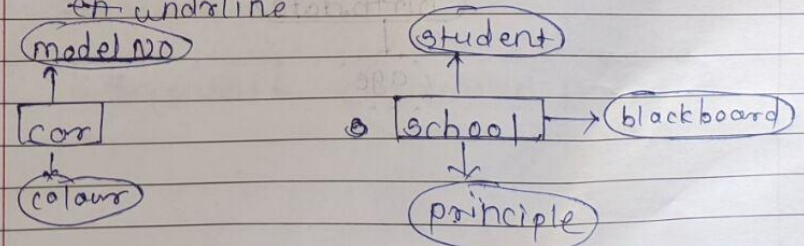
eg:-



3 Dec 2022

* Key Attributes -

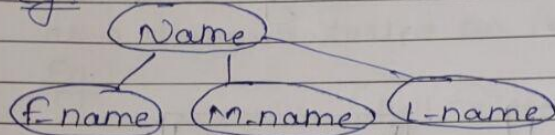
- The Attribute is use to di-describe the property of an entity
- ellipse is used to resp-represent the Attribute
- It is use to represent the main characteristic of an entity. it represents the primary key. It is represented by an ellipse with the text on underline.



*2) Composite Attribute -

- The Attribute that composite of many other attributes is known as a composite Attribute.

eg:-

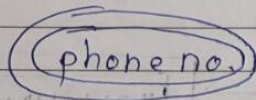


3) Multivalued Attribute -

- An attribute ^{can} have more than 1 value are known as multivalued Attribute. it is represented using double oval.

eg:-

• A student can have more than 1 phone no.



4) Derived Attribute -

- An attribute can be derived from other attribute it known as the derived attribute it is represented as the dash ellipse

eg:-

A person's age changes over time and can be derived from other attribute such as birthdate

Birth date

age

name → n

* Relationship -

- It is used to describe the relation between entities, it is represented using \diamond diamond.

* Types *

1)

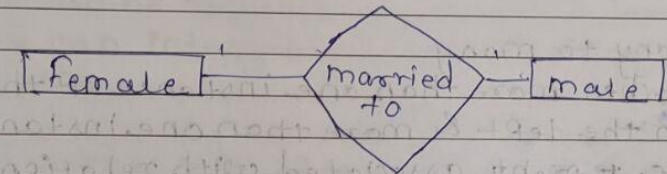
one to one - ^{number}

1 country
1 capital

- When one (instance) of an entity is associated with the relationship then it is known as one to one relationship.

eg:-

A Female can marry to one male, and male can marry to one female.



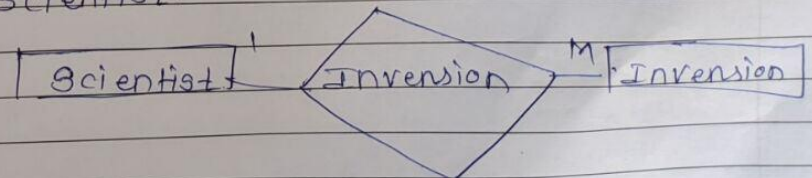
a)

One to many

- When only one instance of the entity on the left ~~is~~, & more than 1 instance of an entity on the right associated with the relationship then this is known as one to many

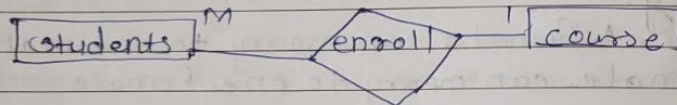
eg:-

Scientist is one and inventions are many but the invention is done by only specific Scientist.



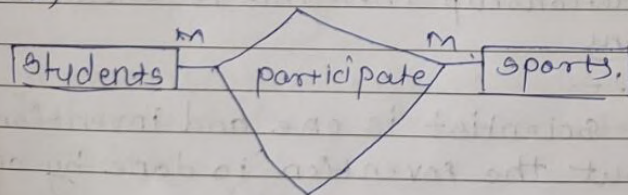
- 3) many to one - ~~more than~~ only one instance of the entity on the right & more than instance of entity on the left associated with relation. is called as many to one relationship.

eg:- student and enrol for only one course but course can have many student.



- 4) many to many
- when more than one instance of the entity on the left & more than one instance on the right associated with relation is called as many to many relationship.

eg:-
1) Students ^{way} are participate in more than one sports.
2)



Diff entity vs Attribute

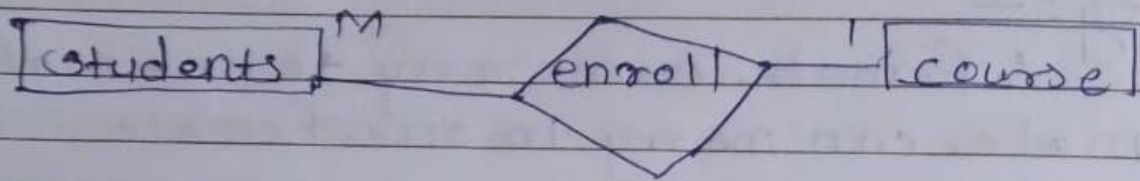
Entity

Attribute

- 1) An entity is a distinguishable real world object that exist. - It describes the elementary feature of an entity.
- 2) A row in a database table structure is an entity. - It, A column header of a database table is an attribute.
- 3) It can be tangible or can intangible entity. - It is involved in forming a key.
- 4) Eg:- the info. of a single student in a student table represents an entity. eg - name, roll No, etc, that forms the complete info. of the a student represent attribute.

- 3) many to one - ~~more than~~ only
- When the ~~only one~~ instance of the entity on the right & more than instance of entity on the left ^{associated with relation.} is called as many to one relationship.
eg:-

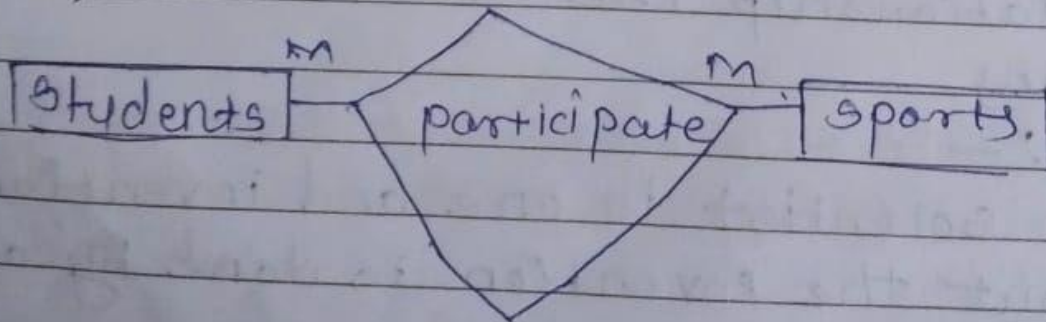
Student and Roll no for only one course but course can have many student.



- 4) many to many
- When more than one instance of the entity on the left & more than one instance on the right associated with relation is called as many to many relationship.
eg:-

1) Students ^{was} ~~are~~ participate in more than one sports.

2)



Diff entity vs Attribute

Entity

Attribute

- 1) An entity is a distinguishable real world object that exist.
- It describes the elementary feature of an entity.
- 2) A row in a database ^{table} structure is an entity.
- It, A column header of a database table is an attribute.
- 3) It can be tangible or can intangible entity.
- It is involved in forming a key.
- 4) Eg:- The info. of a single student in a student table represents an entity.
eg - name, roll No, etc; that forms the complete info. of the a student represent attribute.