DB Administration and Management

=> Reference Book : Database systems by Connolly and Begg

=> Database:

Shared collection of logically related data (and description of this data), designed to meet the information needs of an organization.

-> DB was started in 1970s, called relational data model (also called ER (Entity relation) modelling).

-> Concerned with entities and relation b/w them.

* Entity relationship modelling: (: ER Modelling)
  - Used to view conceptual view of data.
  - OR Logical view of data, that we make before implementing data (such as entities).

- Entity: Any object that has physical existance. such as student, course etc.

- Attributes: Properties of entities.

- Relationship: Relationship / Association b/w two or more than two entities.

★ ER Diagram

★ Types of attributes :

- Simple attr :
  Having only a single (atomic) value which can't be broken further.

  eg:
  Favorite color : red

- Composite attr:
  Can have multiple parts.

  eg:
  Name : fname + lname.

- Derived attr:
  The value which can be derived from other (given) attributes.

  eg:
  Age from DoB.

- Stored attr:
  That is not derived from other attr.

- Single-valued attr:
  Only one possible value.

  eg: Roll no

- Multi-valued attr:
  Can have multiple values.

eg
  Mobile number : Can be more than one.

→ Using these attributes we can make further four different attr:

eg

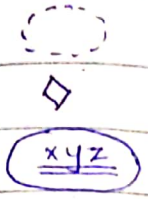
- simple and single
- Simple and multivalued — Favorite color
- Composite and single-valued
- Composite and multivalued

★ ER Diagram:
- Logical view of database
- The overall logical structure of database can be expressed graphically using ER diagram.

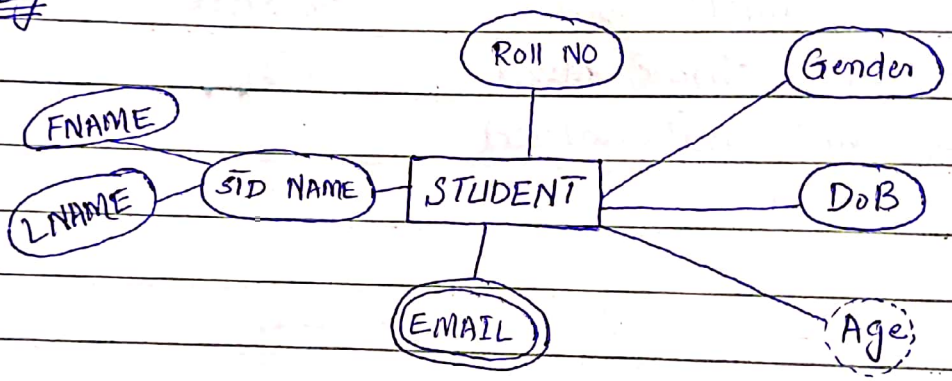
→ Representation of different objects

- Entity — Box 
- Attr — circle 
- Multivalued attr — double-circle 

- Derived attr — Dotted-eclipse
- Relationship — Diamond
- Key attr (unique key) — Underlined



Eg



⇒ Relationship:

A relation is an association b/w two entities.

Eg:

Students	enrolls	cources
Teacher	teaches	courses
Depts	offers	courses

Mathematically :

$R \subseteq S \times C$:

(subset of cartesian product)

⇒ Degree of relationship :

- Binary — Two entities
- Ternary — Three entities
- N-ary — N entities

→ Recursive relationship :

This is the relationship from entity to itself.

eg:

Course is prerequisite for another course.

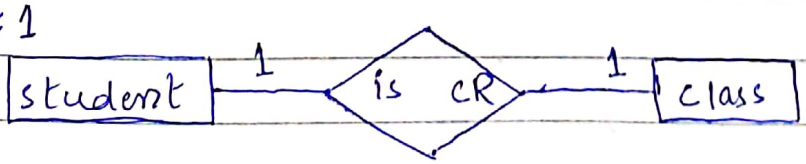
→ Participating entities :

⇒ Every entity relationship has a cardianality ratio

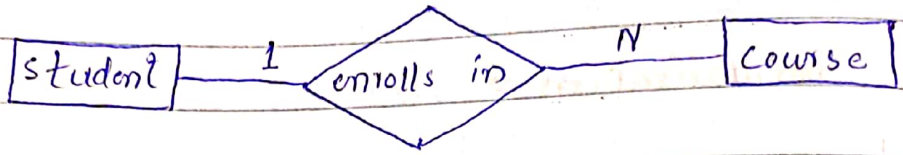
- one-to-one (1 : 1)
- one-to-many (1 : N)
- many-to-one (N : 1)
- many-to-many (M : N)

* Examples :

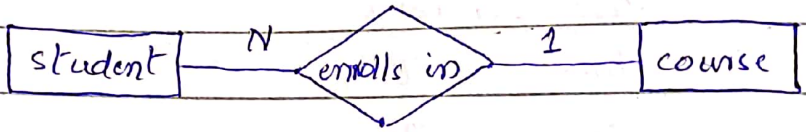
1 : 1



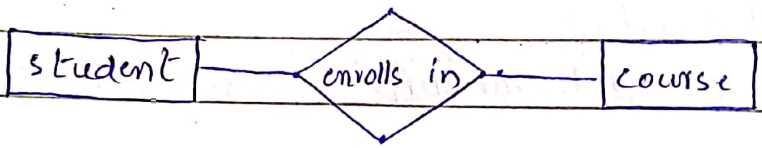
1 : N



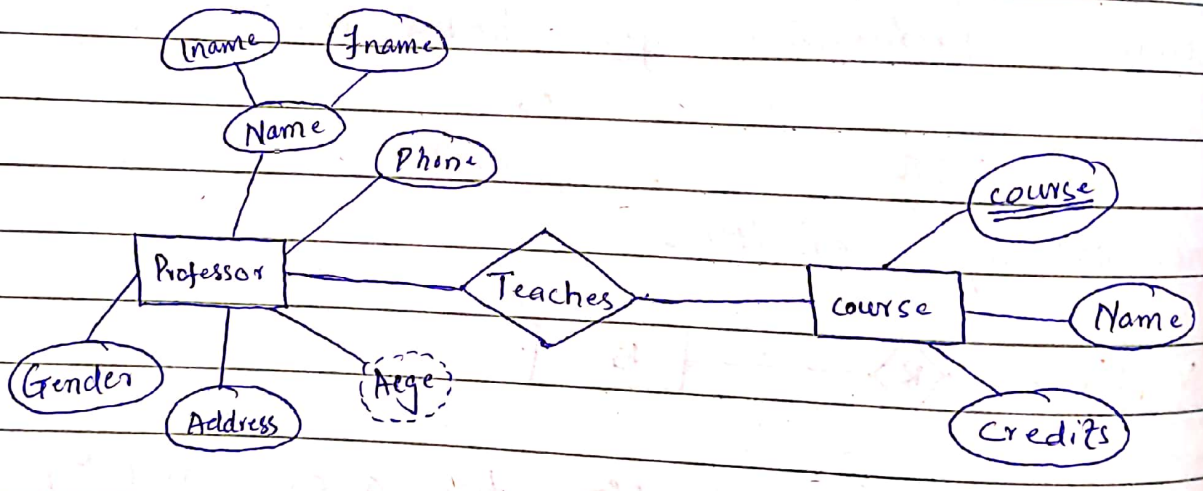
N : 1



M:N



Eg



Normalization

⇒ What is normalization ?

Process of organizing the data in the database. Normalization is used to minimize the redundancy from a relation or set of relations. and reducing data anomalies.

⇒ What is redundancy ?

- Multiple copies of the same data. OR
- Duplicate copies of same data stored at many places.

→ Disadvantage of this is that when we delete or update the data, we have to do the same at many places.

⇒ Normal Forms :

- First normal form (1NF)
- Second normal form (2NF)
- Third normal form (3NF)

(PPT)

⇒ What is inconsistancy ?

A situation where there are multiple tables within a database that deals with the same data but may receive it from different inputs. Inconsistency is generally compounded by data redundancy.

⇒ Why Normalization ?

- Normalization is needed because if you do not normalize your database, 3 anomalies (faults) will/may occur in your db.
  - Update
  - Insert
  - Delete

Running Example:

We have a unnormalized table with 4 attributes (emp_id, name, address, emp_dept)

- Now if address of an emp who is working in multiple depts is changed, then his address needs to be updated individually in multiple depts, (update anomaly)
- If emp_dept doesn't accept (Null) then an employ under-training with no dept can't be inserted. (Insert anomaly)
- If one dept is deleted from the table then info about all emps will also be deleted. (Delete anamaly)

⇒ 1 NF

→ A table must not contain repeating groups of attributes.

→ A table must not contain composite or multi-valued attributes.

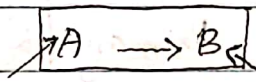
⇒ In simple words:

- Each field of a table may contain only one item.
- All of the data items in a column must mean the same thing.
- Each row of the table must be unique.
- A table must have no repeating column.
- All attributes must depend on a primary key.

⇒ Dependencies:

→ Functional dependency:

- A relationship b/w attributes.
- If value of one attribute is known, it is possible to obtain the value of another attribute.



Determinant — Determined / Dependent.

Eg:

student

Roll_No	St_Name	DoB	Email

Determinant	Roll_No	→	St_Name
	Roll_No	→	DoB
	Roll_No	→	Email

Determined

OR

Roll_No → St_Name, DoB, Email

⇒ An attribute can depend on two or more attributes.

course

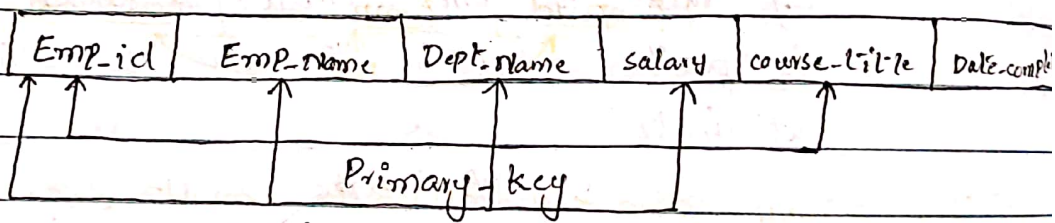
Roll_No	course-id	date_completed

Roll_No , course-id → date-completed

⇒ Partial dependency :

One or more non-key attributes are functionally dependent on a part of a primary key.

eg:



Partial dependency

(Emp_id → Emp_Name , dept_Name , salary)

Partial Primary-key

Emp_id , course_title → Date_completed

complete P.K.

⇒ Transitive Dependency :

Function dependency b/w two or more non-key attributes.

There are some conditions for this dependency that must be true in Transitive dependency.

$X \to Y$

$Y \not\to X$

$Y \to Z$

=> 2 NF

Rules :

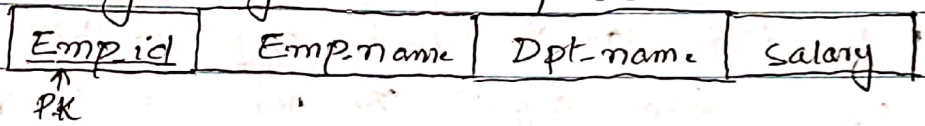
1- Relation must be in 1NF.

2- All non-key attributes must depend on complete set of primary key.

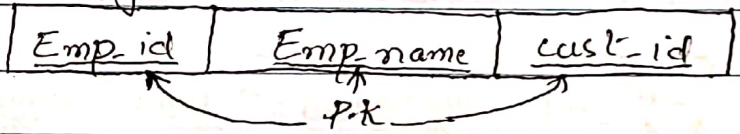
3- No Partial dependency exist.

=> Partial key won't exist if : one of these is true.

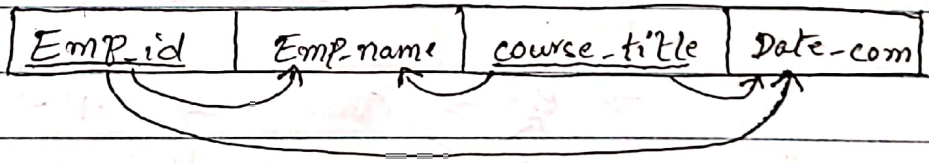
1- Primary key consist of one attribute.



2- No non-key attribute exist.



3- Every non-key attribute must depend on every attribute (Full set) of primary key.



Eg

Normalizing to second normal form.

-Here we take a table which is 2NF.

Emp_id	Emp_Name	Dept_name	salary	course_title	Date_comp

Emp_id → Emp_Name , Dept_Name , salary

Emp_id , course_title → Date_comp

- Now we have identified partial depende and we will make 2 seperate table so that no partial dependecies exist in them.

EmpId	Emp_Name	Dept-Name	Salary

Emp_id	course-title	date-comp

Both table don't have any partial dependency and both fulfill the rules of 1NF. Though both are in 2 NF.

⇒ 3 NF

Rules :

1- Relation (table) should be in 2NF
2- No transitive dependency exists.

A → B → C ⇒ A → C

A: key attribute
B, C: Non-key attribute

Example :-

Sales (Cust-id , cust-Name , sales-Man , Region)

Transitive dependency:
cust-id → cust-Name , sales-Man
sales-Man → Region
cust-id → Region indirectly

Eg:

We have a table in 2NF.

Sales

cust-id	cust-name	sales-Man	Region

→ 2NF.

Dependencies:

. cust-id → cust-name, sales-man

sales-Man → Region.

(1-transitive dependency)

→ We identified the dependencies.

→ Now to remove that transitive dependency we will make this table 'sales' in two seperate tables and make sales-Man the P.K.

sales

cust-id	cust-Name	sales-Man

sales-Man

Sales-Man	Region

Now it has no transitive dependecies and was already in 2NF, though now it is in 3NF.