BD Life Cycle

- ✓ It's important to know the steps, and what is the input and output of each steps or stage in DB Life Cycle.

1- **Analysis:**
   - **Input:** of this step is group of meetings that did by the analyst with the clint.
   - **Output:** Requirement Document → scope of the project.
2- **DB design:**
   - **Input:** Requirement Document → scope of the project.
   - **Output:** ERD → Logical Representation (This document can generate more than one ERD because we think different from each other).
3- **DB Mapping:**
   - **Input:** taking the ERD and apply the rule
   - **Output:** Actual table →logical database schema

- ✓ The rules of this step are fixed or preserved 2 = 1+1 مثل 😊
- ✓ If we found one to many will do this . . .
- ✓ There is no more negotiation!

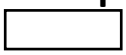
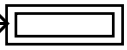
4- **DB implementation:**
   - **Input:** take the Actual table →logical database schema
   - **Output:** physical schema → online Database (centralized DB) → allow us to work with it 😊

   - ➢ **As I mention before that DB without Interface no use of it!**

5- **Application GUI** → Interface
6- **Clint** → Test

   - ➢ **All our work in this track will be on step 4 DB implementation.**

Let's quickly review the main components of ERD 😊

- ✓ **Entities** → Strong Entity (PK)→ 
  → Weak Entity (Partial Key) → 

- ✓ **Attribute** → Simple (cannot divided, not repeated and we can not fund at run time)
  → Composite (combine from 2 things or more)
  → Multi-Value (contain 2 value or more)
  → Driven (birthdate → age)
  →Complex (Multi-Valued + Composite)

- ✓ **Relationship** →Degree (unary, binary and attribute)
  →Cardinality (1:1, 1:M, M:M)
  →Participation (total, partial)

- **We said that all the entities that we draw will convert → to table, but what about relationship?!**

- ✓ **FK constrain is PK in another table**
- ✓ **PK → unique and not null**
- ✓ **FK → it allowed to be repeated and can take null value (any value in FK must match a value in PK)**
- ✓ **Its optional to be the FK, PK with different name but must be same datatype.**
- ✓ **Value in FK must match value in PK but the visa vires no, its optional that the value in PK should be in FK**

- **Relationship (Tables) in database (parent and child)→ cannot delete parent that have child**

- **For example**: I want to delete Department 1 → cannot delete or edit because Mark is in Dep 1
- If we have Department with no employee → yes you can delete or edit 😊
- ❖ Department → **parent**
- ❖ Employee → **child**

Employees

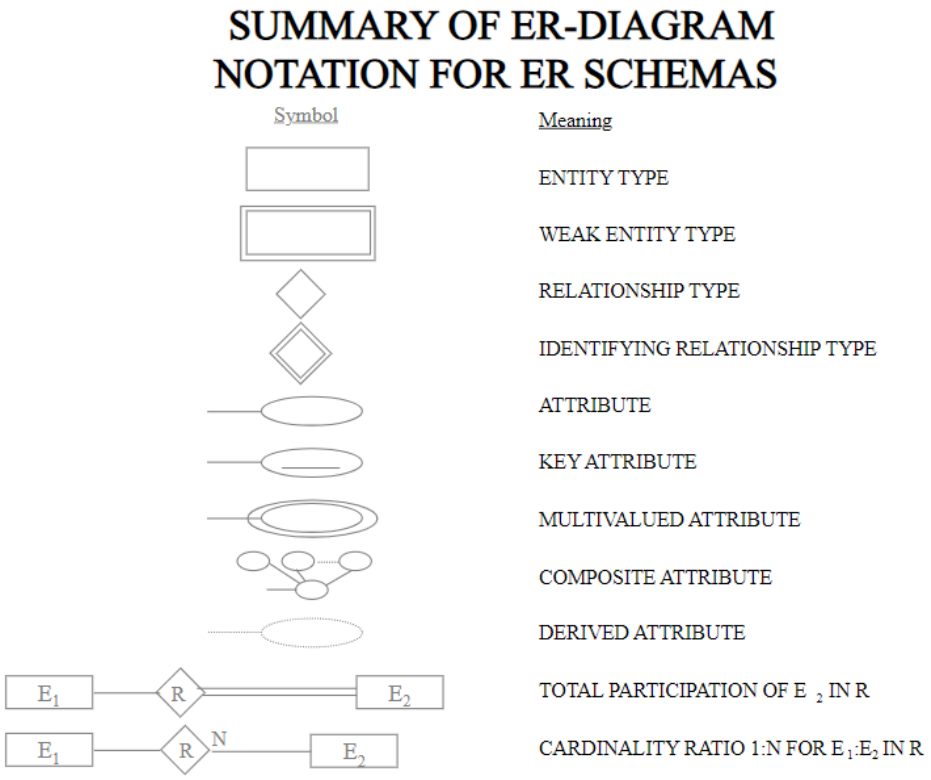
EmployeeID	EmployeeName	DepartmentID
1	Mark	1
2	John	1
3	Mike	1
4	Mary	2
5	Stacy	3

Departments

DepartmentID	DepartmentName
1	IT
2	HR
3	Payroll

Let's together move to Mapping 😊

- ✓ First you should know the symbol



- ❖ **Here we remove the participation because it will convert like: FK allow null or not?**

- ✓ **If the participation is partial** → that is mean FK can be null
- ✓ **If the participation is total** →that is mean the FK must be not null
  - **For example:** employee must have department→ total participation with employee → that is mean the FK that we take it from Department and put it on Employee must be not null → to represent the total relationship.

- ❖ **This means we do mapping without considering the participation.**

Relational Database Definitions

Table or Entity: a collection of record

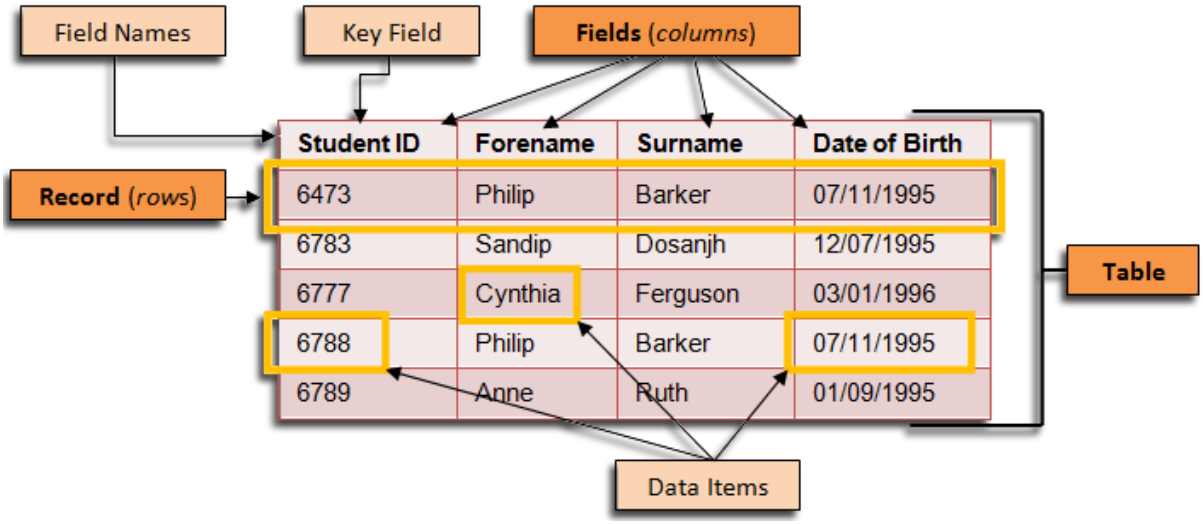
Attribute or Column or Field: characteristic of one entity

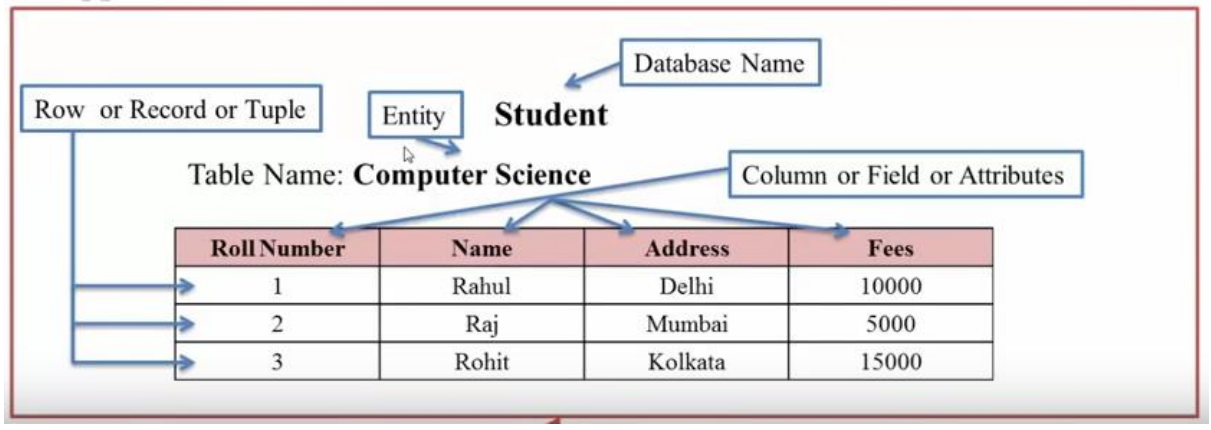
Row or Record or Tuble: the specific characteristic of one entity

Database: a collection of tables

- **We will observe this when we open the tool → we will see the hierarchy**

Server → database → tables → rows and column →

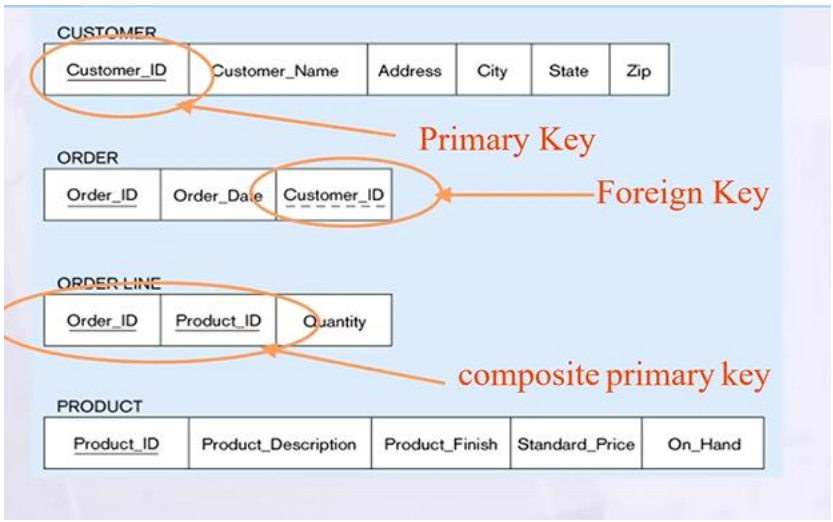




- ✓ Every column in database have domain.
- ✓ That is mean the data type in the column → its not only the type, the rang of the value also.
- ✓ Domain (rang of values, data type →tinyint→1 Byte (-128 : +127)
- ✓ Second things that I want age column(20 -30)→ No age with minus or we want specific range of age → **here we should apply constraints.**
- ✓ From this constraints you choose the PK,FK→ which is connect the tables
- ✓ every column ***must*** have datatype.
- ✓ Add **constraint** to ***control the value*** (e.g. salary>1000).

Mapping → DB schema

- ✓ This is the output of the mapping



- The most important thing is the connecting arrow between FK and PK.
- The arrowhead is pointing at PK.

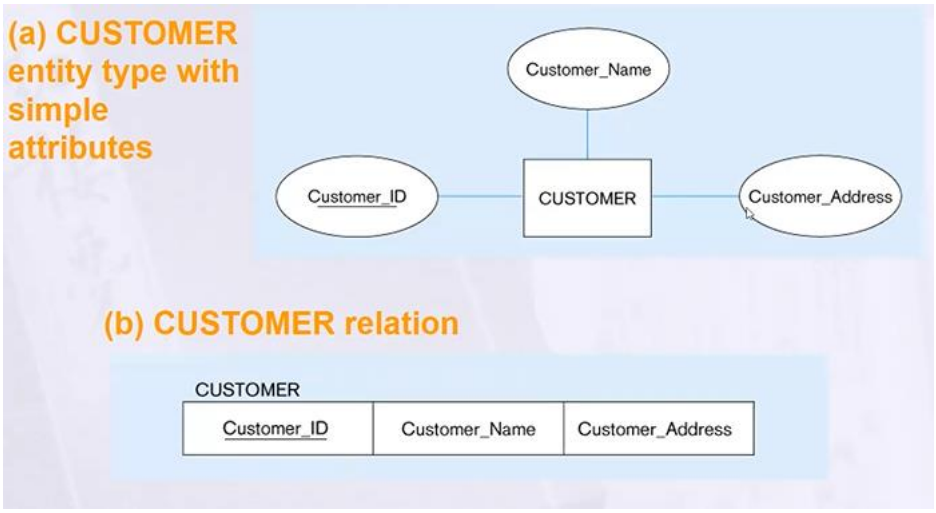
So, what is the steps? 😊 This is the best practice

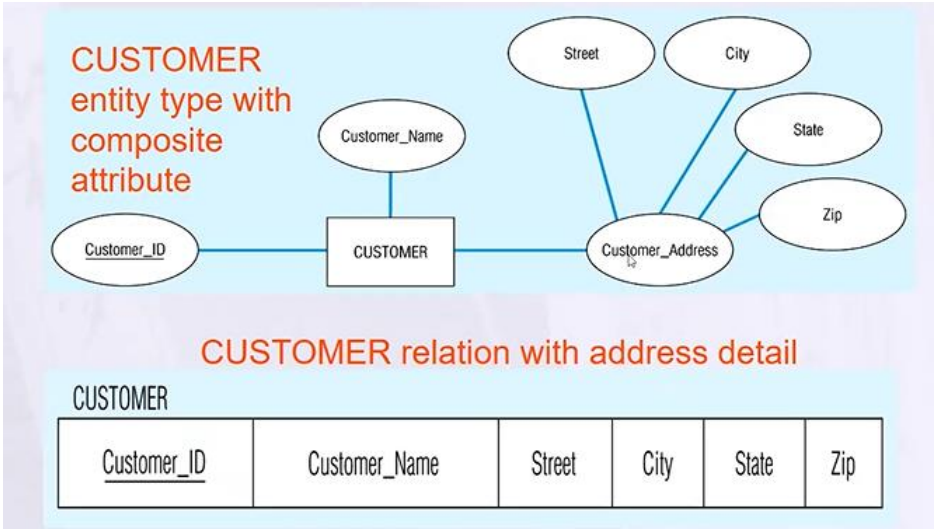
✓ ***ER - to-Relational Mapping***

➔ **Step 1:** Mapping of Regular Entity types (Strong)
➔ **Step 2:** Mapping of Weak Entity types
➔ **Step 3:** Mapping of 1 : 1 Entity types
➔ **Step 4:** Mapping of 1 : N Entity types
➔ **Step 5:** Mapping of M : N Entity types
➔ **Step 6:** Mapping of N-ary Entity types
➔ **Step 7:** Mapping of Unary Entity types

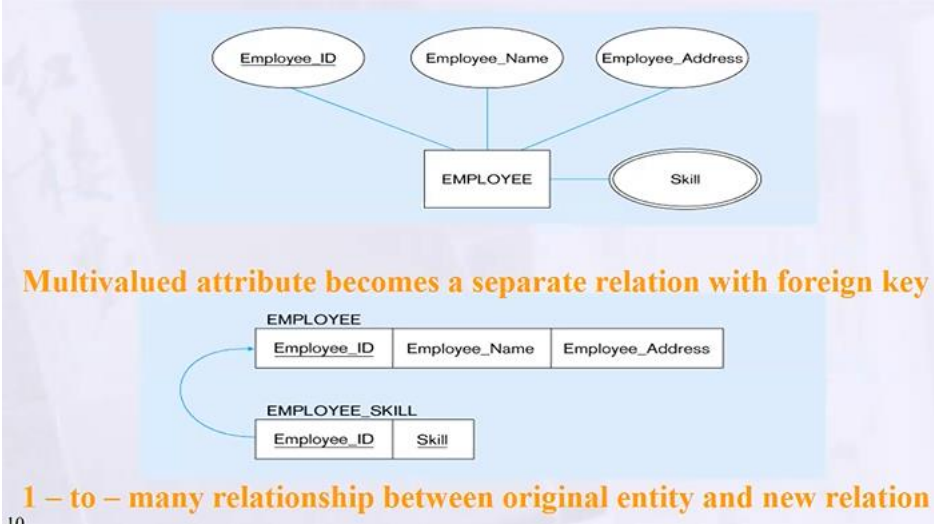
✓ ***Step 1: Mapping of Regular Entity types (Strong) → Direct mapping***

- Create table for each entity type.
- Choose one of key attributes to be the PK.





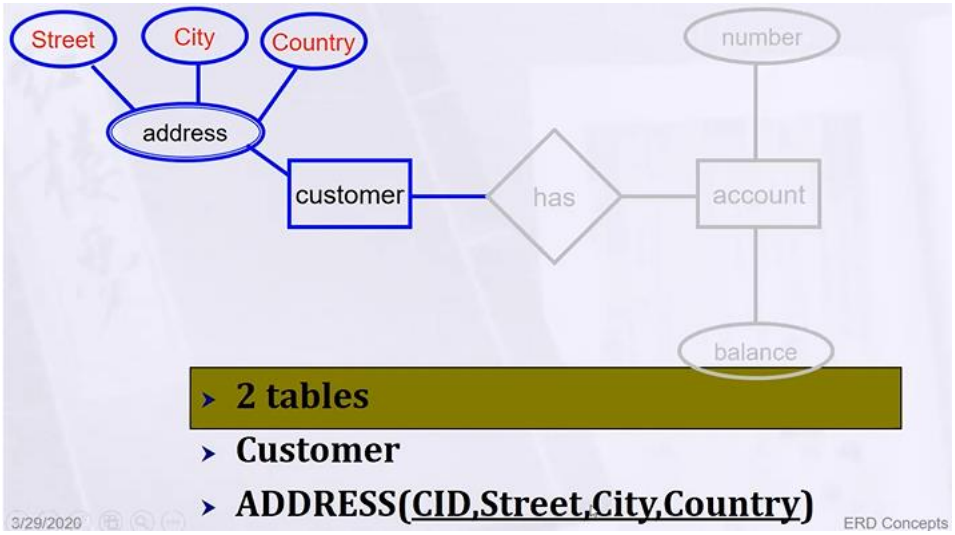
- ✓ For the Multi-valued attribute cannot add this to the table →we will have repeated PK
- ✓ So we create new table employee skill
- ✓ Employee_ID as FK → we take both of them composite PK because it can be repeated.
- ✓ So as I told you before its wrong to say entity is a table → because one entity have number of table



Mapping Derived & Complex

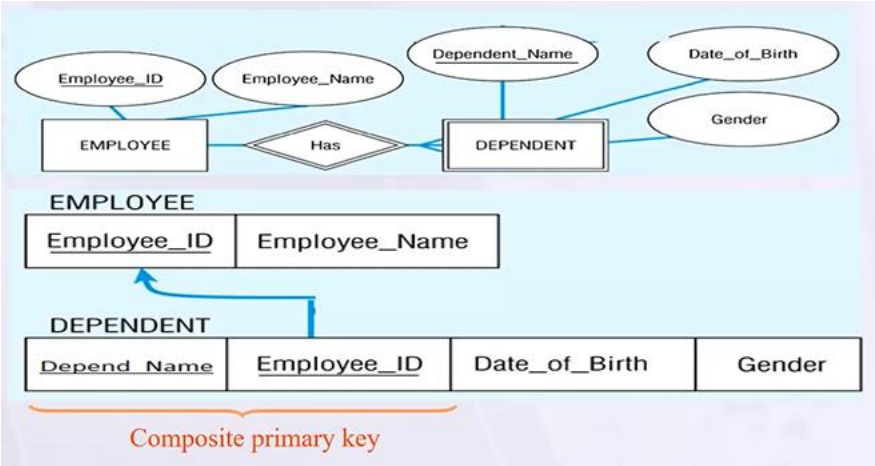
- ✓ In the most cases Derived attribute not be stored in DB
- ✓ Mapping Complex Like mapping Multivalued then including part of the multivalued attribute as columns in DB.

Other example:



Step 2: Mapping of Weak Entity types

- ✓ Create table for each weak entity.
- ✓ Add FK that correspond to the owner entity type
- ✓ PK composed of: هذا طبعا منطقي لأنه هي لوحدها ما لها لازمه في النظام 😊
  - Partial identifier of weak entity (composite)
  - Pk of strong entity (simple attribute)



Step 3: Mapping of 1 : 1 Entity types

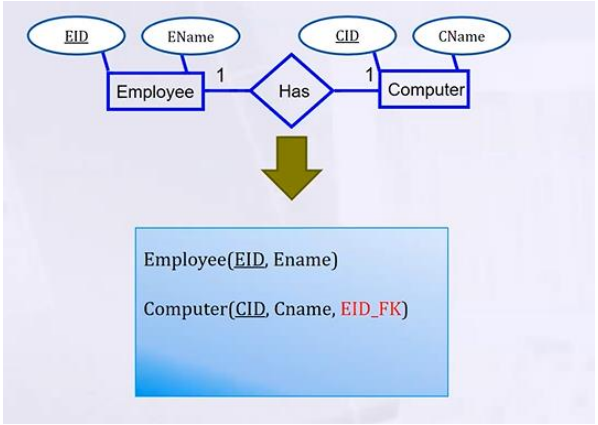
- Start with binary relationship: →**binary 1 to 1**

  →Binary 1 to N

  →Binary M to N

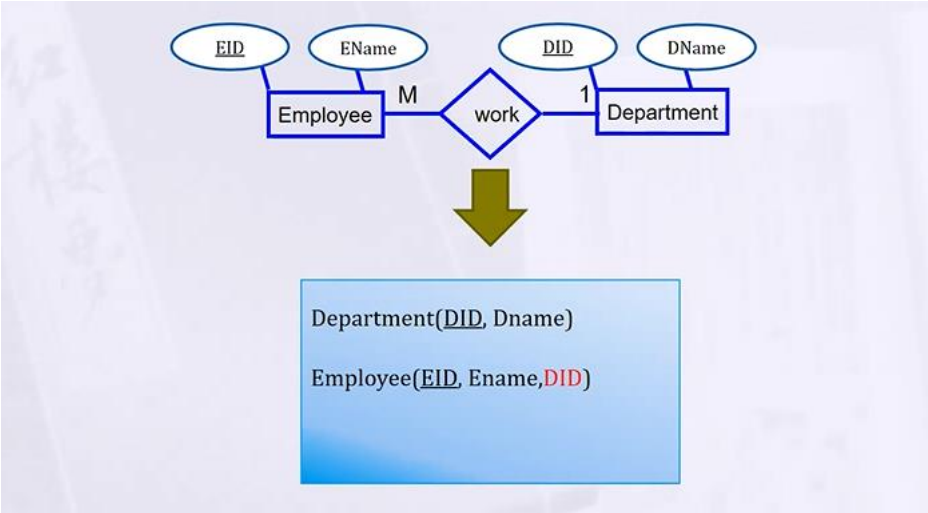
- So, as I told you Relationship → convert to FK constrain
- **Question here** 😊 : مفتاح أي جدول اخذه واحطه ف الثاني ؟!
- While the relationship is 1:1 I will take the key of any table and put it in the other one.

لكن عشان تكون متأكد اكثر شوف الجدول الي ممكن ينحط فيه صفوف كثير وخذ البرايمري كي تبعه وحطه في الجدول المتوقع يكون فيه صفوف اقل



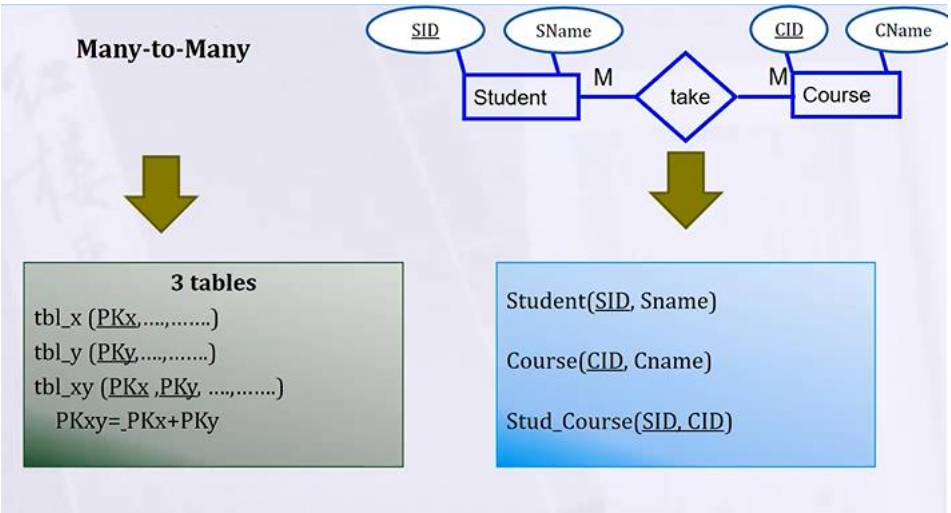
→Binary 1 to N

- (take the PK in 1 put it in N)

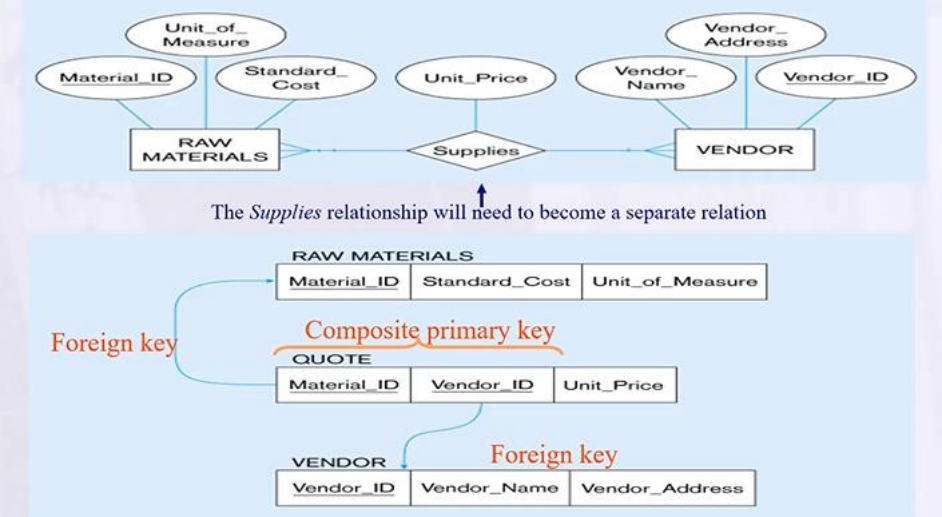


Step 5: Mapping of M : N Entity types

- Create a new third table
- Add FKs to the new table for both parent tables (will be composite PK)
- Add simple attributes of relationship to the new table if any.



✓ **Question** 😊 if there is attributes on the relationship

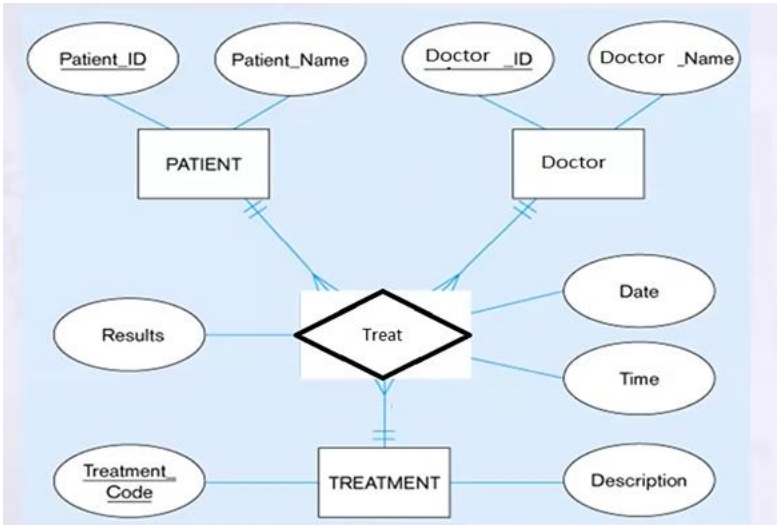


Step 6: Mapping of N-ary Entity types

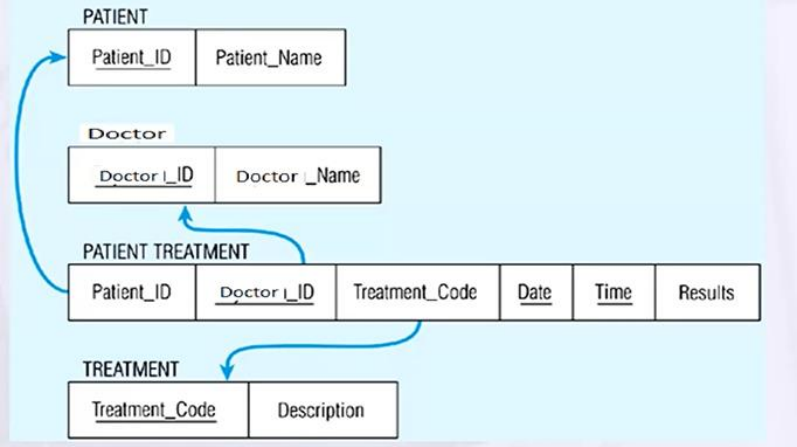
(we don't see to the cardinality and participation → convert to new table)

If n > 2 then:

- Create a new third table: اجاوب على ثلاث أسئلة: كم كولوم في الجدول الجديد وكم فورن كي ومن البرايمري كي تبعه
- Add FKs to the new table for all parent tables

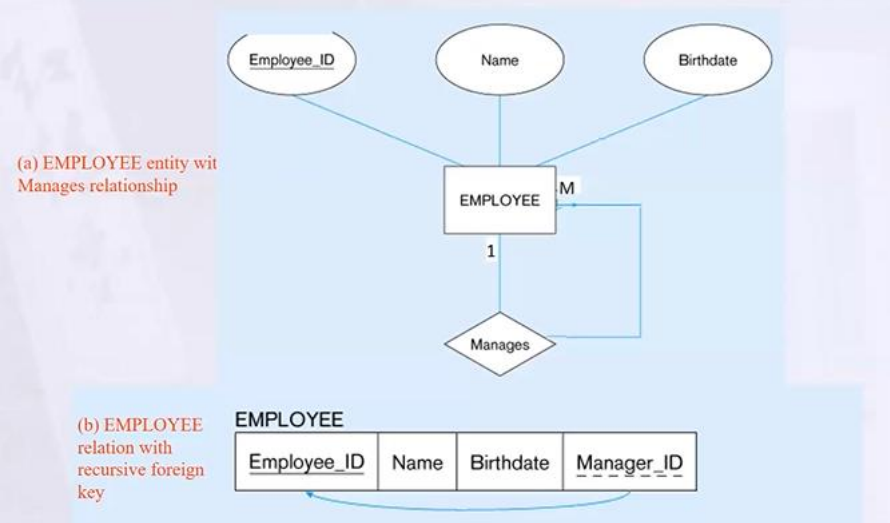


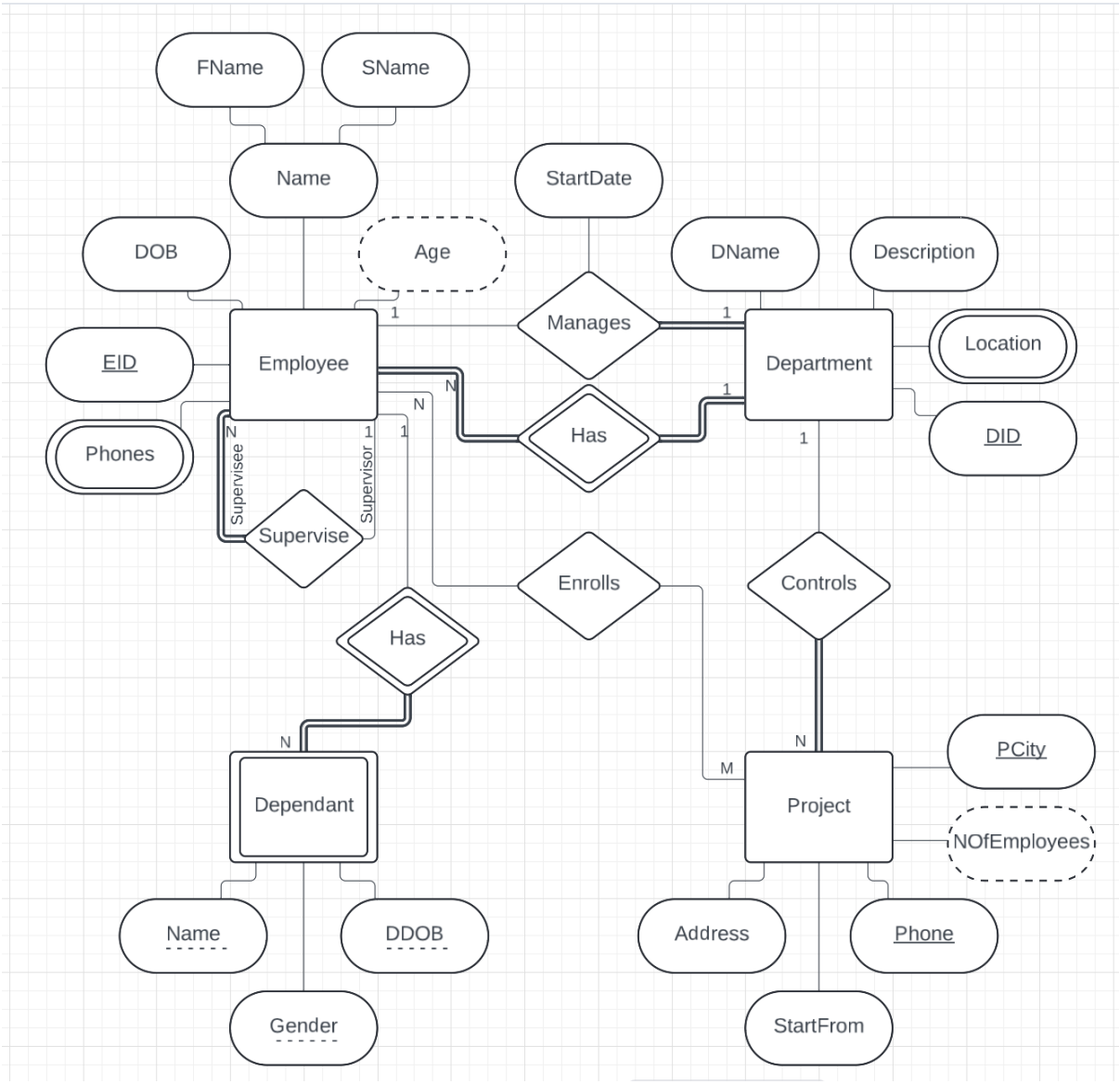
- ✓ FK: PID,DID,TreatmentCode
- ✓ Columns: 3 FKs with (date ,Time, Result)
- ✓ PK: دوروا عليه 😊تباديل وتوافيق
- ✓ (PID, Date, time) IF I don't have date and time بتفلسف وبحط واحد من معي 😊

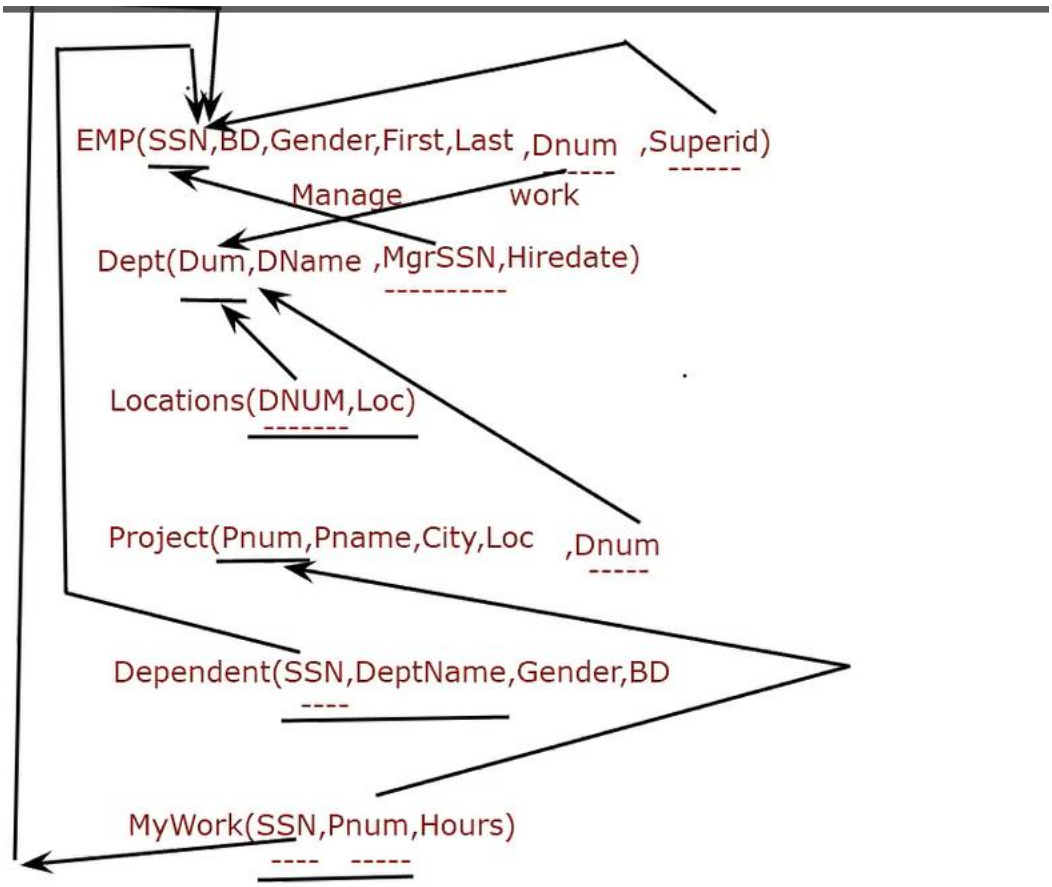


Step 7: Mapping of Unary Entity types

نرجع لأصل الموضوع




FName
SName
Name
StartDate
DOB
Age
DName
Description
Manages
EID
Employee
Department
Location
Has
Phones
DID
Supervisee
Supervisor
Supervise
Enrolls
Controls
Has
PCity
Dependant
Project
NOfEmployees
Name
DDOB
Address
Phone
Gender
StartFrom


EMP(SSN,BD,Gender,First,Last ,Dnum ,Superid)
Manage
work
Dept(Dum,DName ,MgrSSN,Hiredate)
Locations(DNUM,Loc)
Project(Pnum,Pname,City,Loc ,Dnum
Dependent(SSN,DeptName,Gender,BD
MyWork(SSN,Pnum,Hours)

ANSI SQL (American National standard institute)

- **Microsoft:** Transact SQL
  - RDBMS (tools)→Access, SQLServe
    - DDL → Structure
      - ✓ Create table
      - ✓ Create function,
      - ✓ Create view
      - ✓ Alter
      - ✓ Drop
      - ✓ Select into
    - DML→ Data
      - ✓ Insert
      - ✓ Update
      - ✓ Delete
      - ✓ merge
    - DCL→permission
      - ✓ Grant
      - ✓ Deny
      - ✓ Revoke
    - DQL→display
      - ✓ Select
      - ✓ Joins
      - ✓ Union
      - ✓ Subquery
      - ✓ Grouping
      - ✓ Aggregation function
    - TCL→DB consistency
      - ✓ Begin Transaction
      - ✓ Commit
      - ✓ Rollback
- **Oracle**: PL SQL
- **IBM**: IBM PI-SQL
- **MySQL**: Open source