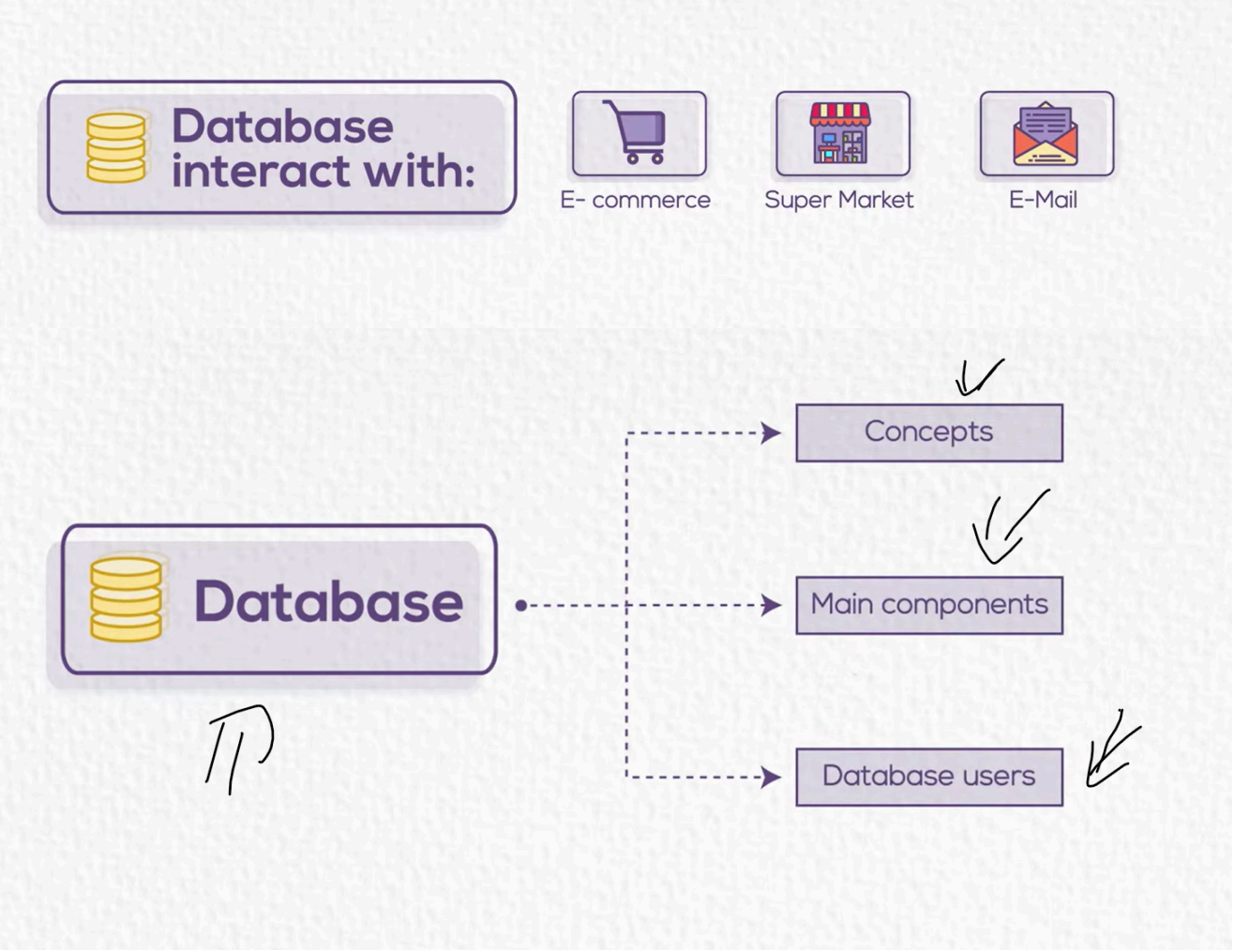
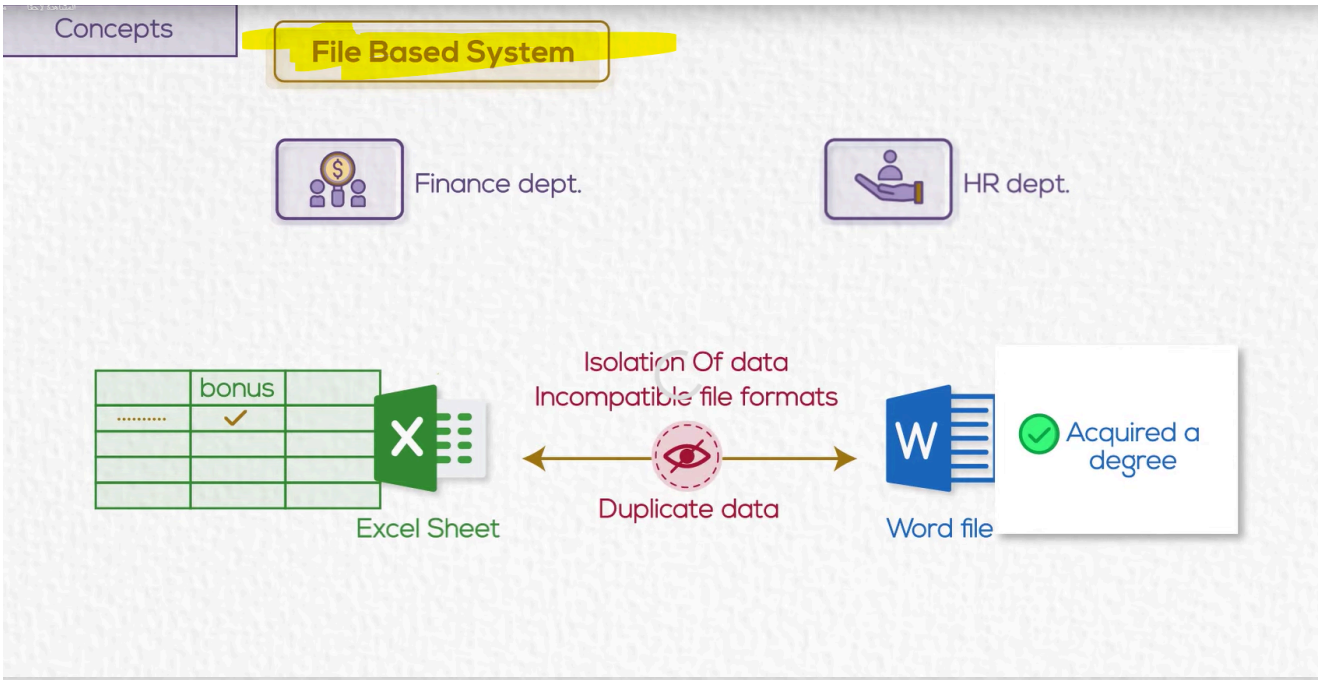
# Database Mahara Tech

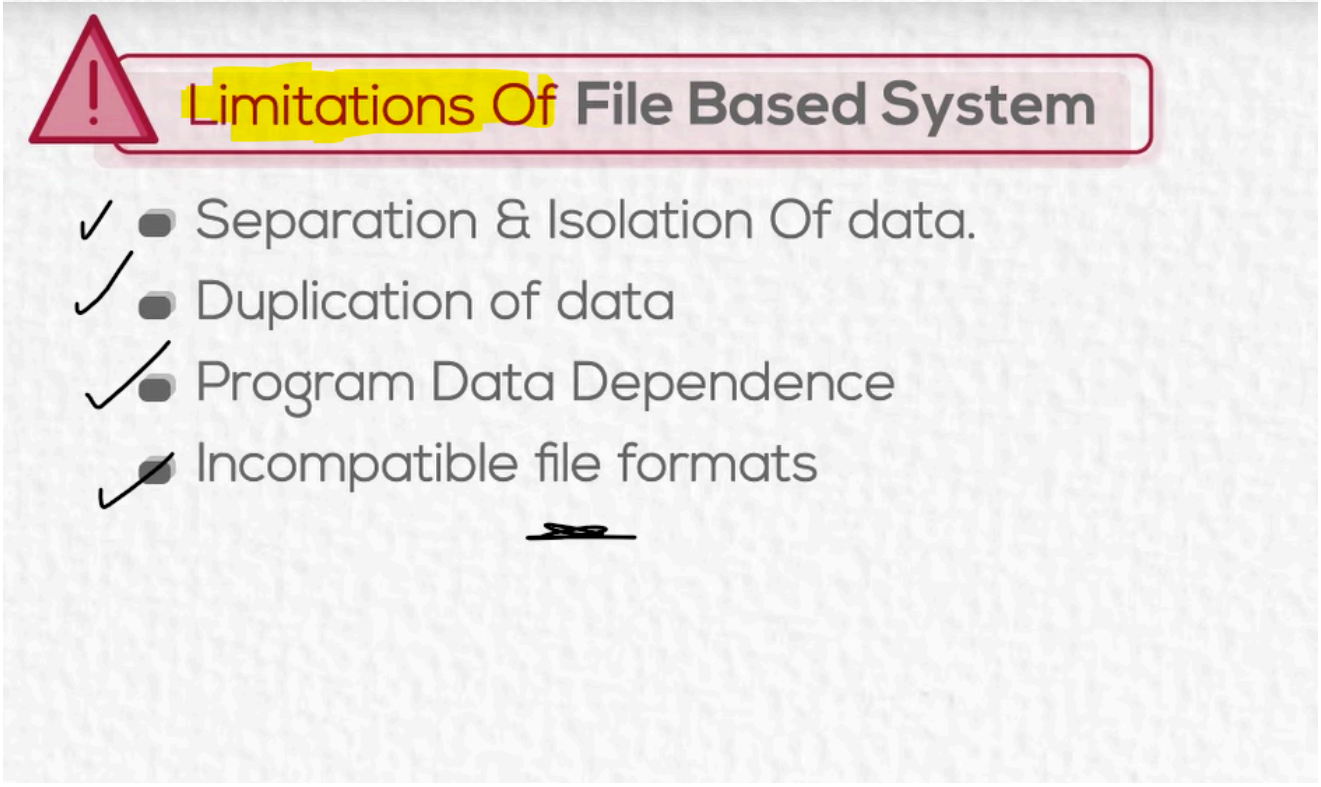
# Chapter 1 - Introduction

*Any App want a database to store data and retrieve it*



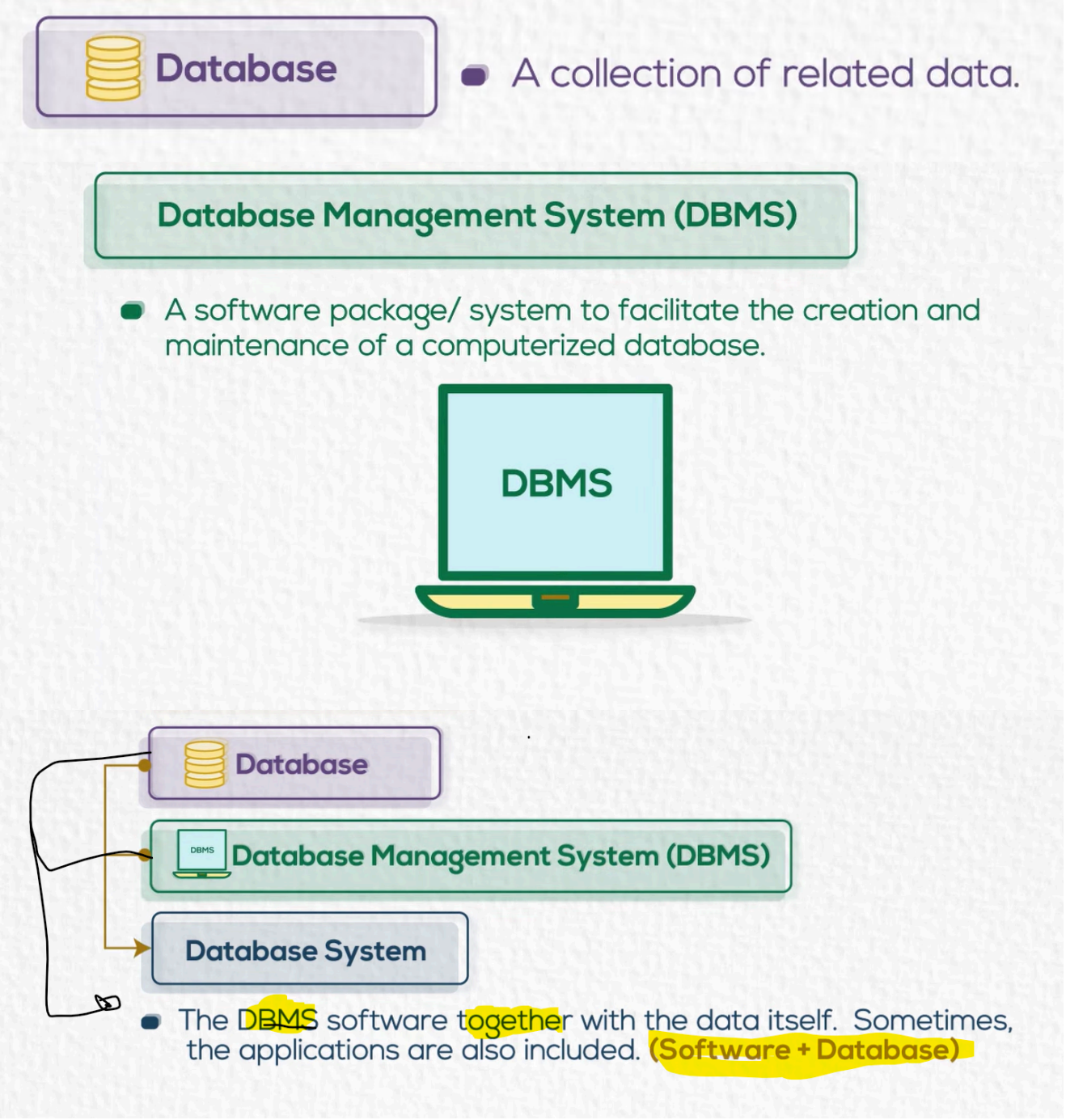

## File Based System



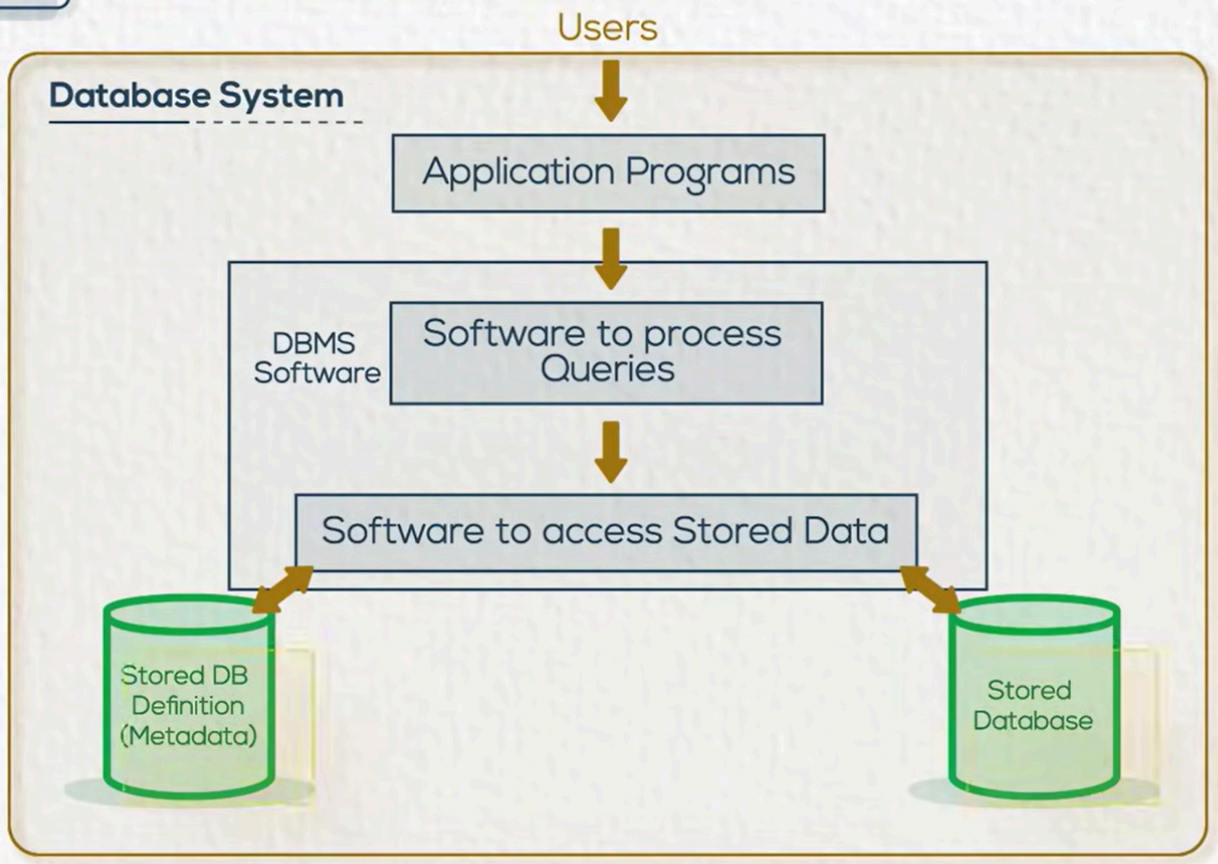

# Database

التعريفات




## Main Components

stem • Main components

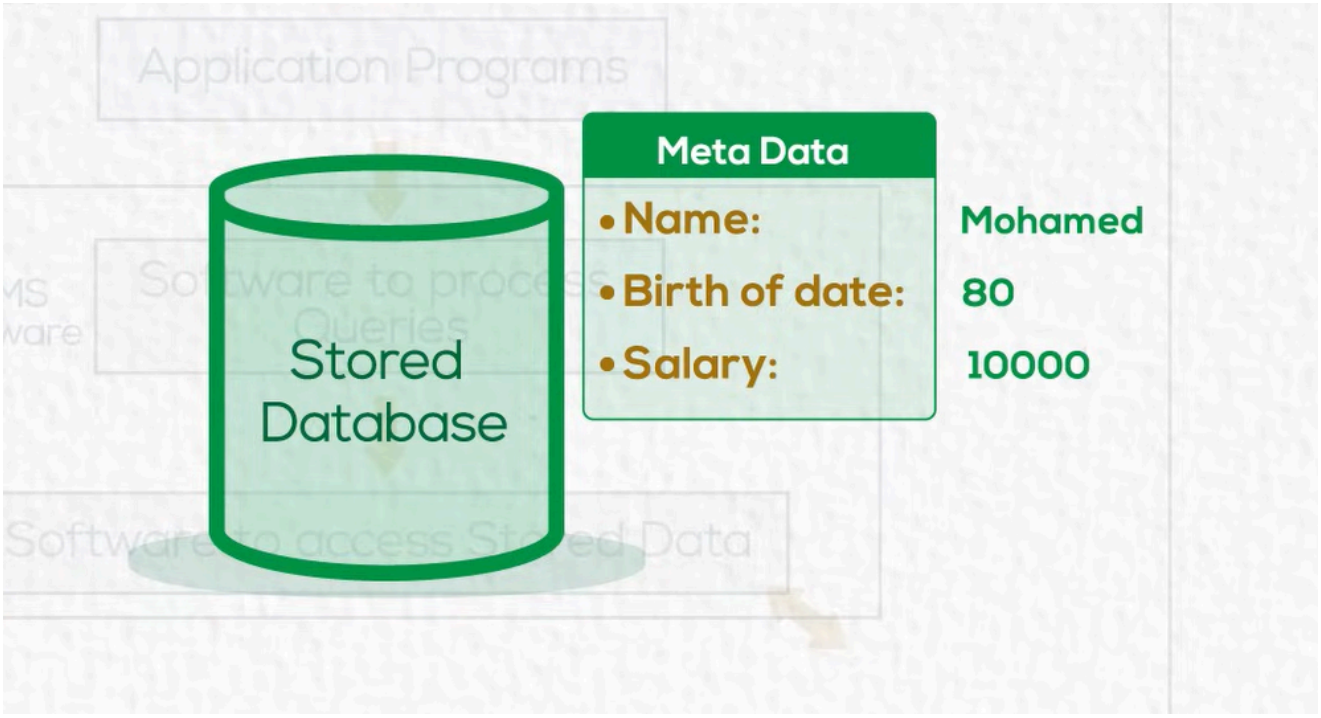



- Table Name
- Column Name
- Column data types
- Constraints
- Access Privileges

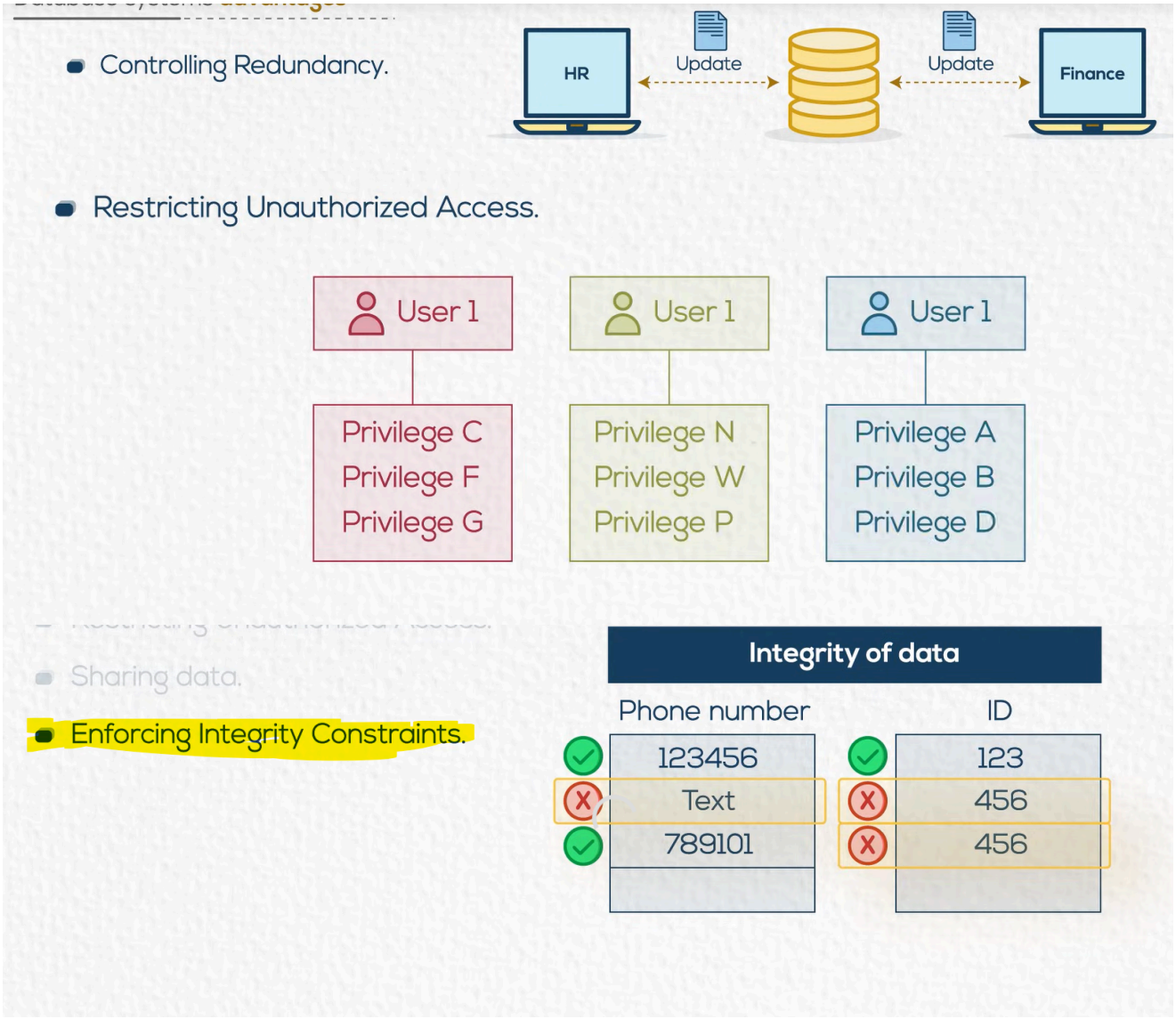
Stored DB Definition (Metadata)

- Username
- Password
- User's Privileges
- DB Objects structure
- Log files

Metadata is Database Definition Stored all information

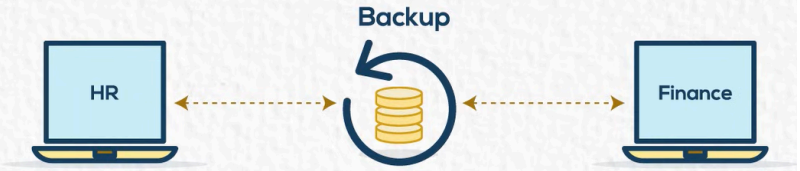

Application Programs
Stored Database
Meta Data
• Name:
• Birth of date:
• Salary:
Mohamed
80
10000

## Advantages


Controlling Redundancy.
HR
Update
Update
Finance
Restricting Unauthorized Access.
User 1
Privilege C
Privilege F
Privilege G
User 1
Privilege N
Privilege W
Privilege P
User 1
Privilege A
Privilege B
Privilege D
Sharing data.
Enforcing Integrity Constraints.
Integrity of data
Phone number
123456
Text
789101
ID
123
456
456

- Enforcing Integrity Constraints.
- Inconsistency can be avoided.

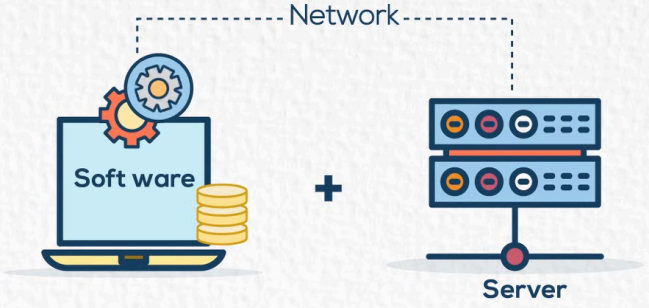

- Inconsistency can be avoided.
- Providing Backup and Recovery.



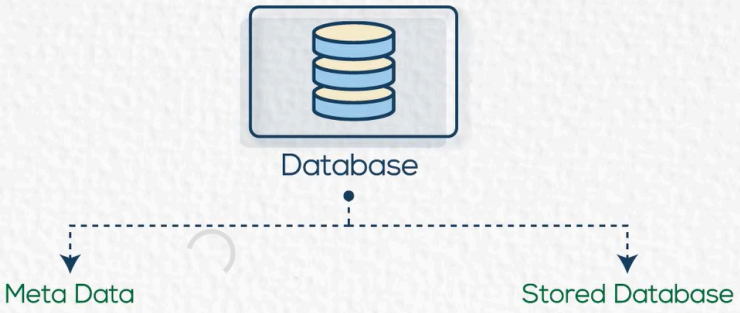

## Disadvantages

- Needs expertise to use.
- DBMS is expensive.

| | |
|---|---|
| | Support DBMS |
| | Database |
| | User |

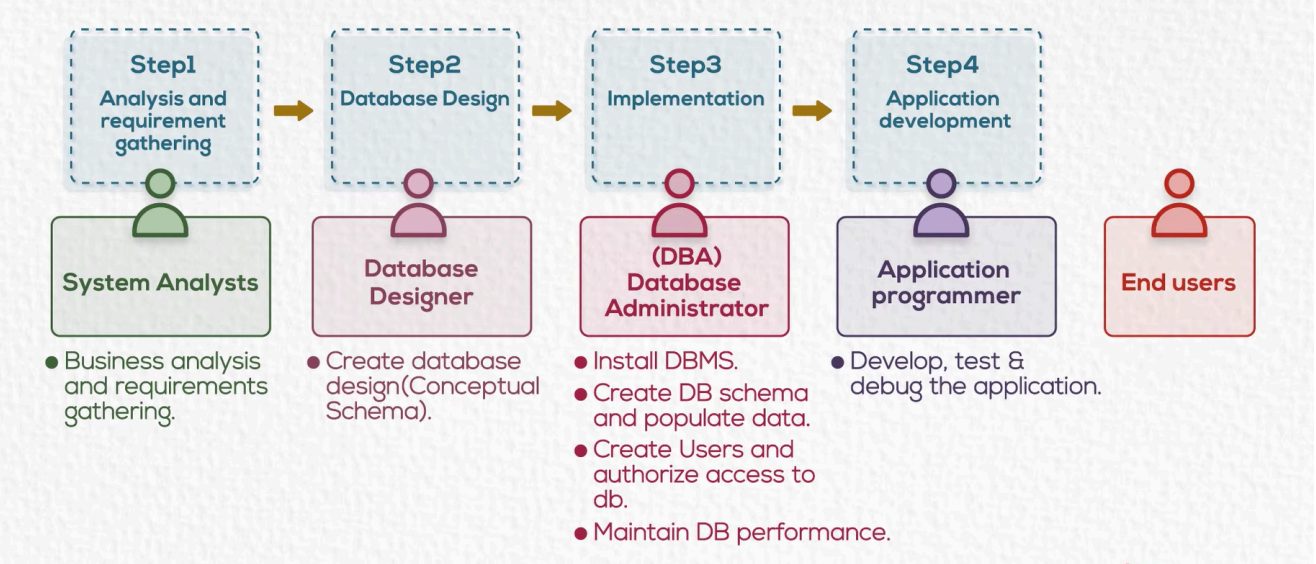



- Needs expertise to use.
- DBMS is expensive.
- May be incompatible with any other available DBMS.




## Database Users

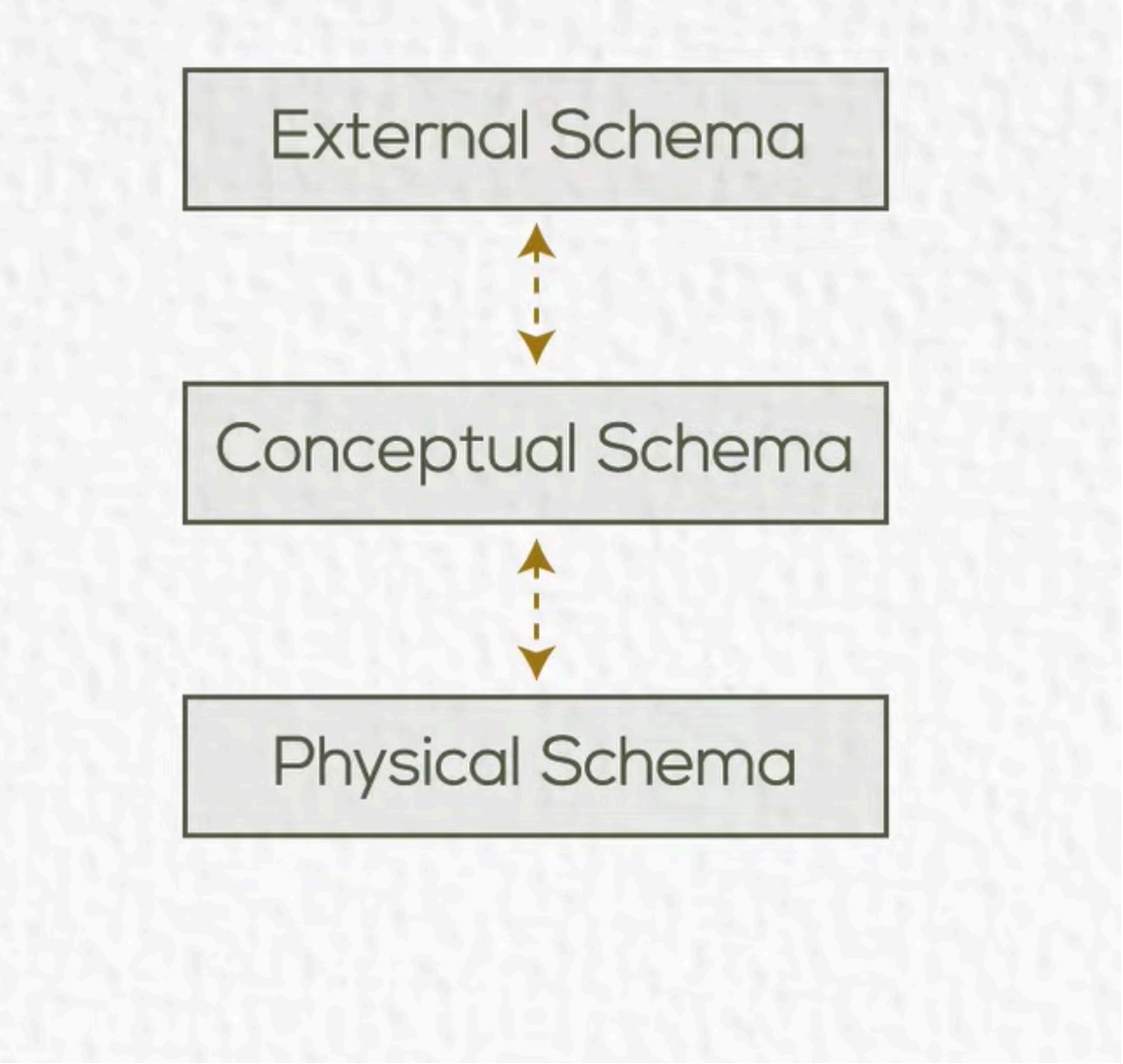
## Process



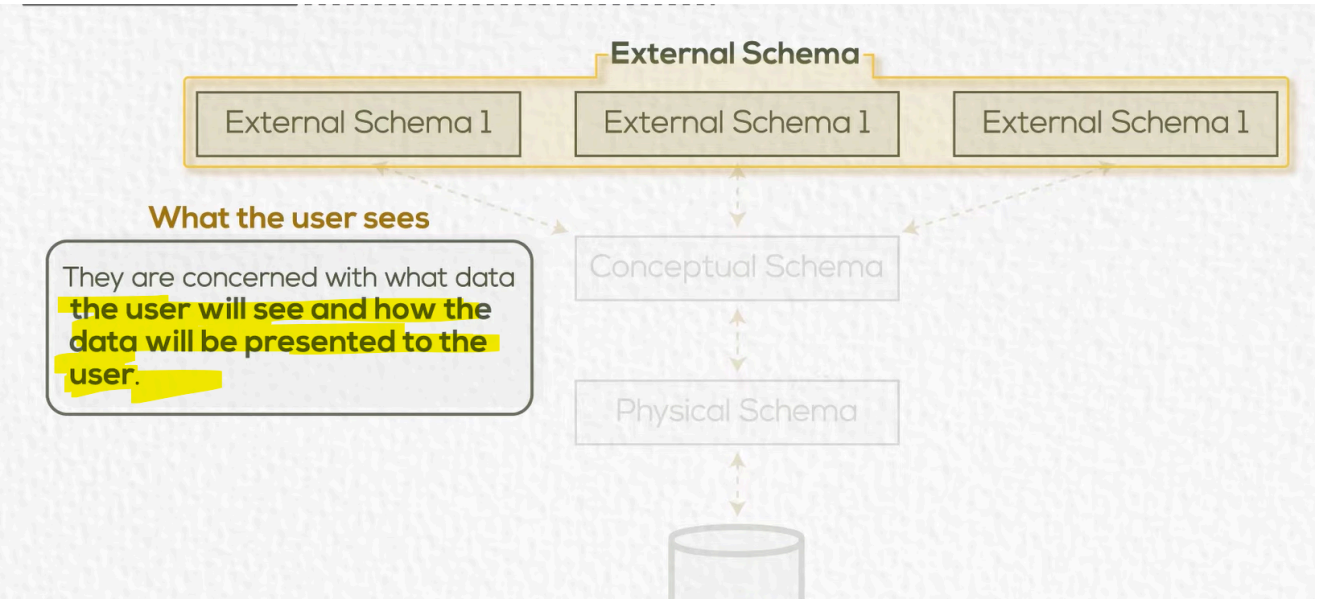

## DBMS Architecture

.

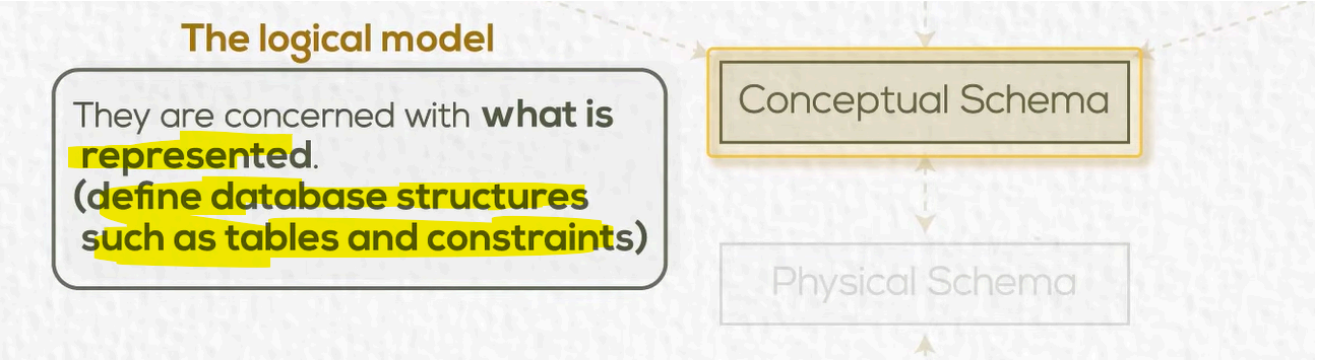
## Three Types of Schema



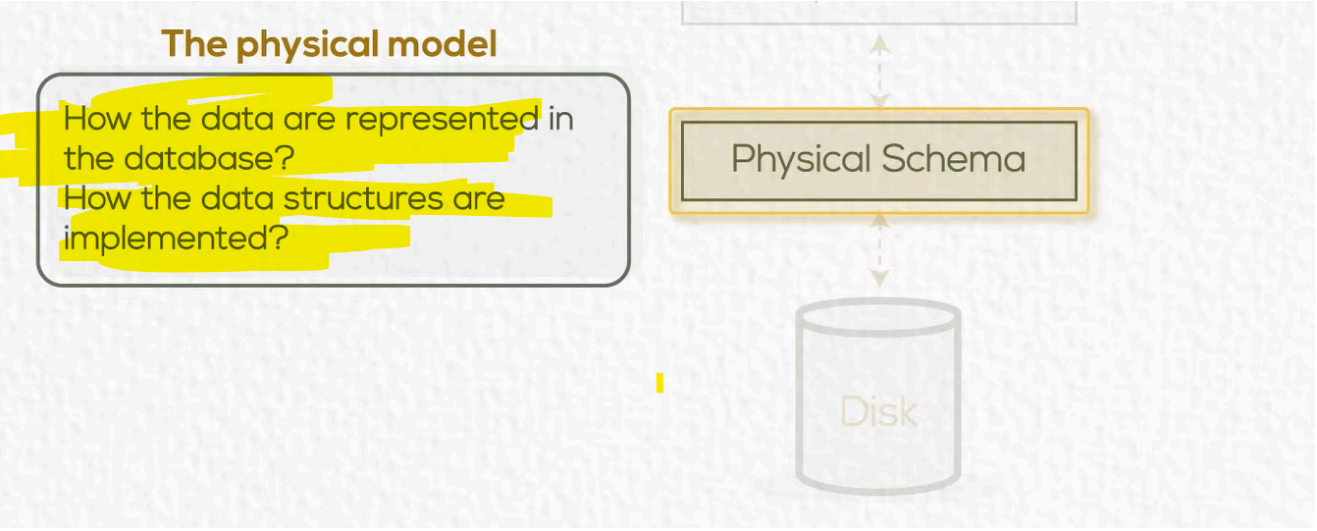
## External Schema



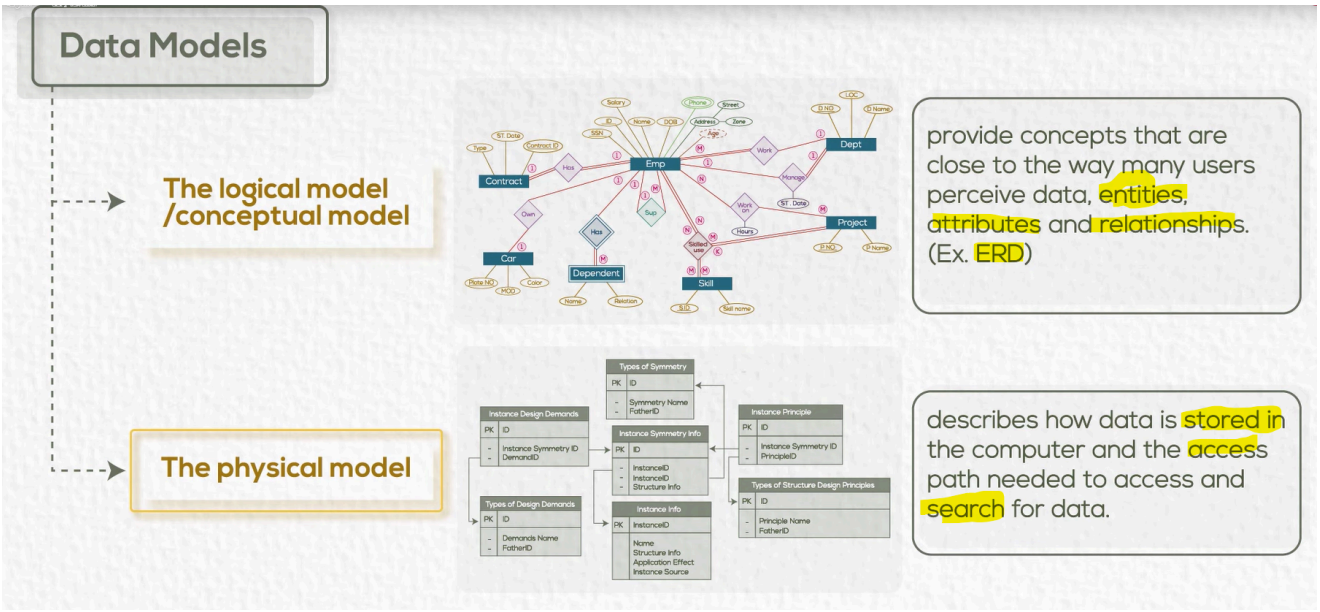

## Conceptual Schema



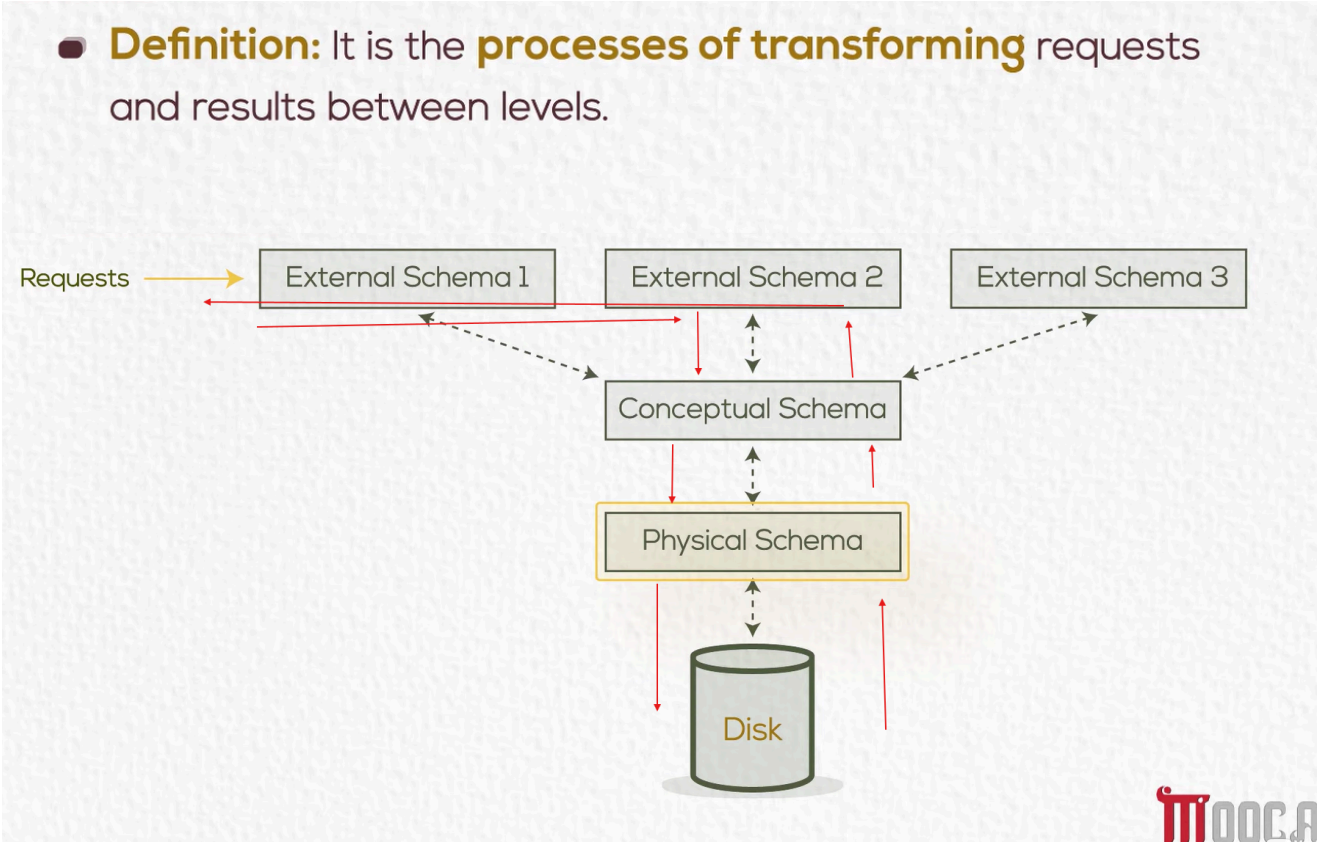

## Physical Schema




## Data Models




## Mappings

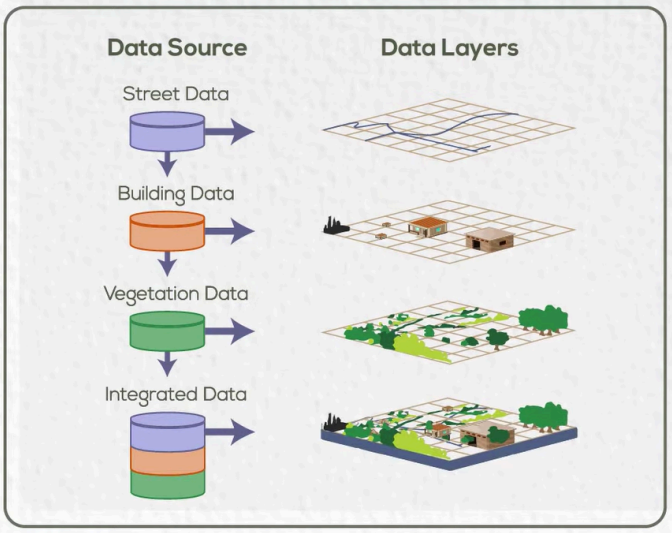



## DBMS Other function

- Text/Number/Image/Audio/ Video

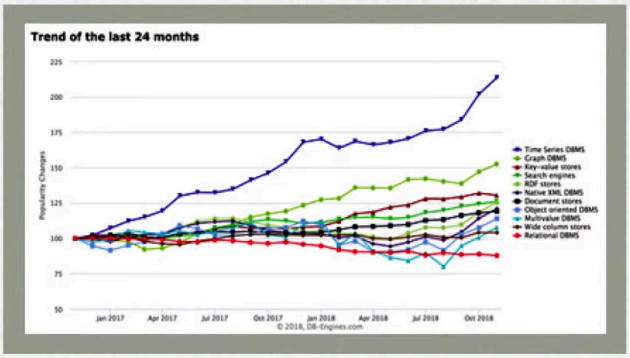
DBMS other function

- Multimedia Function
- Spatial Data
- Time Series
- Data mining



DBMS other function

- Multimedia Function
- Spatial Data
- Time Series
- Data mining

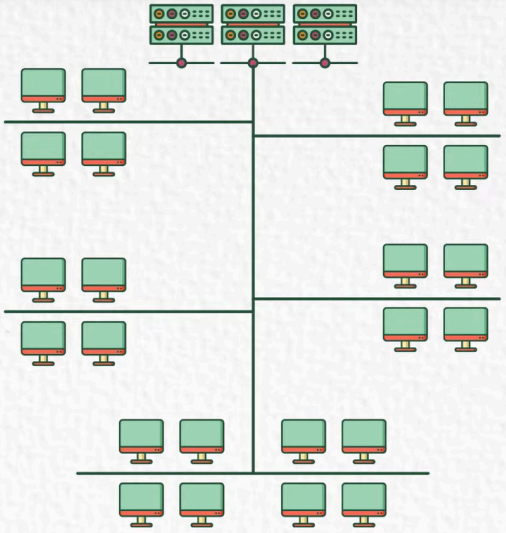



- Multimedia Function
- Spatial Data
- Time Series
- Data mining


# DBMS Environment

## Centralized Database Environment

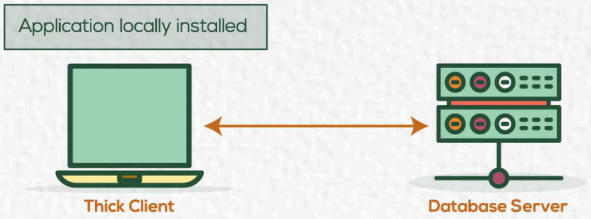
## ① Mainframe environment.



### Problems with this environment

- The processing depends on one server.
- The performance is very slow.
- Database and application layer has Single Point of failure.

## ② Client/Server environment.



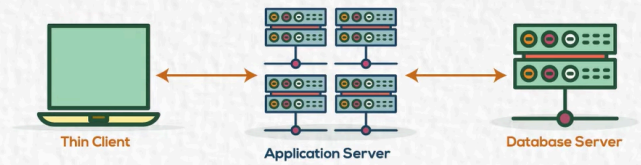

### Problems with this environment

- Database is a single point of failure.
- High cost For support.

### Advantages

- Application layer **isn't** a single Point of failure.

③ Internet Computing environment (Three-tier architecture).



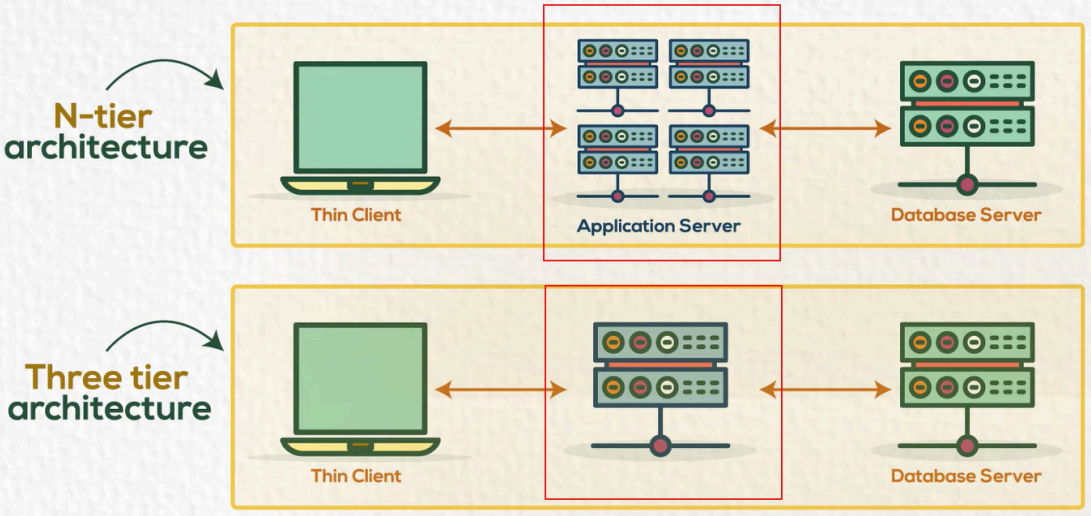

**Problems with this environment**

- Application server is a single point of failure.
- Database is a single point of failure.

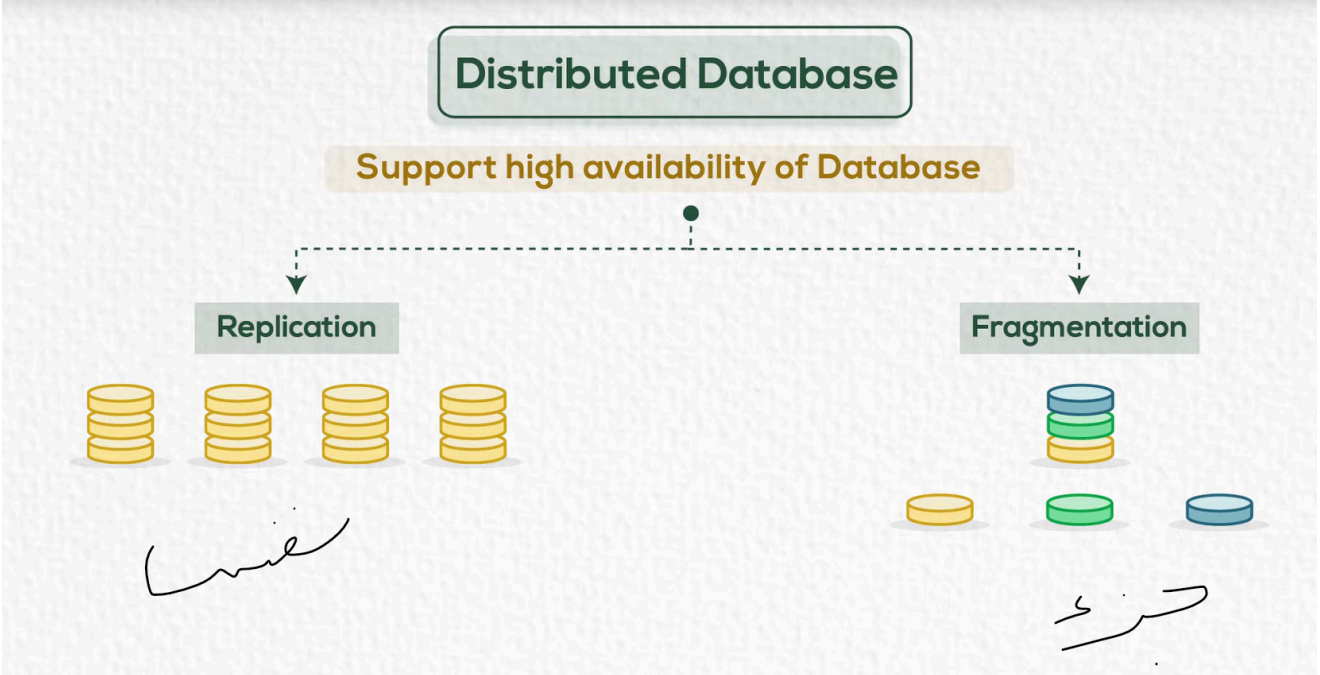
**Advantages**

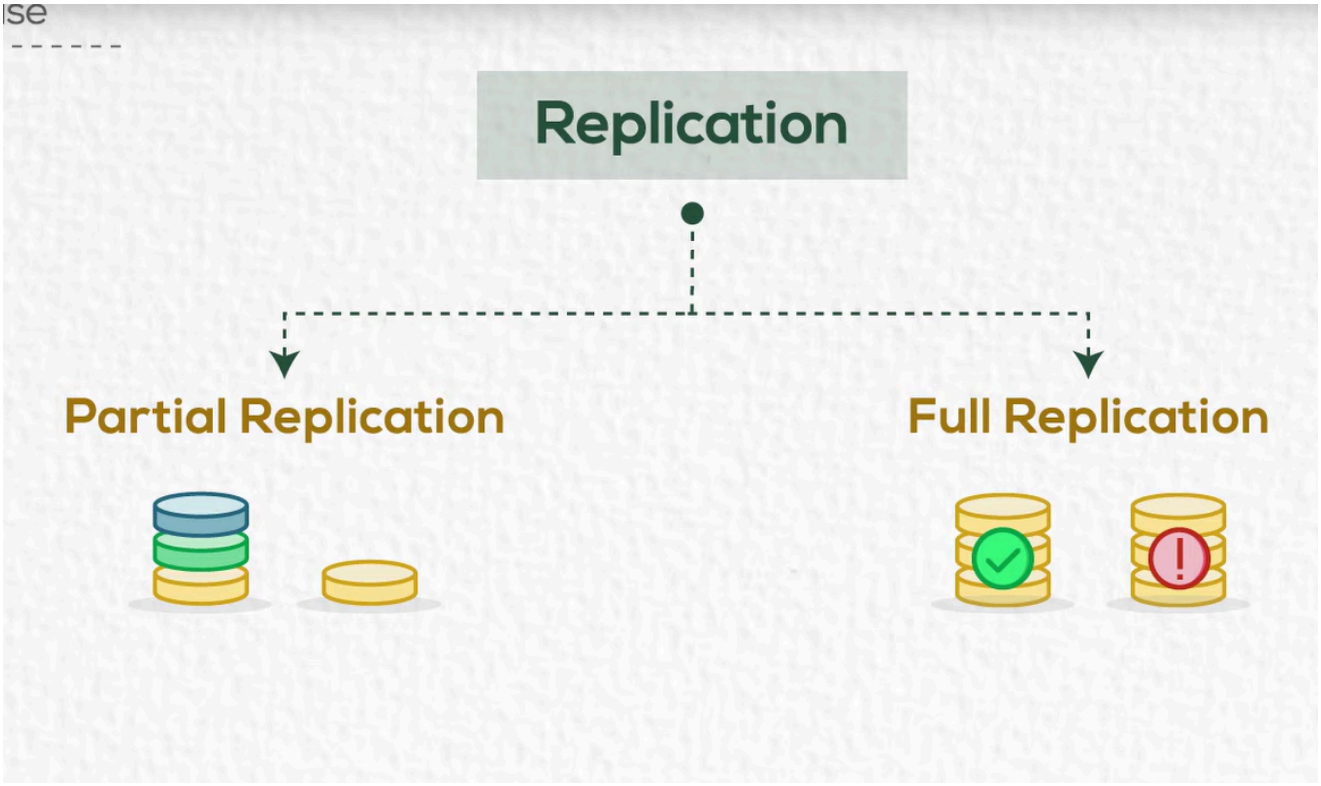
- Lower cost for support and maintenance.

③ Internet Computing environment (Three-tier architecture).




## Distributed Database Environment




Replication
Partial Replication
Full Replication

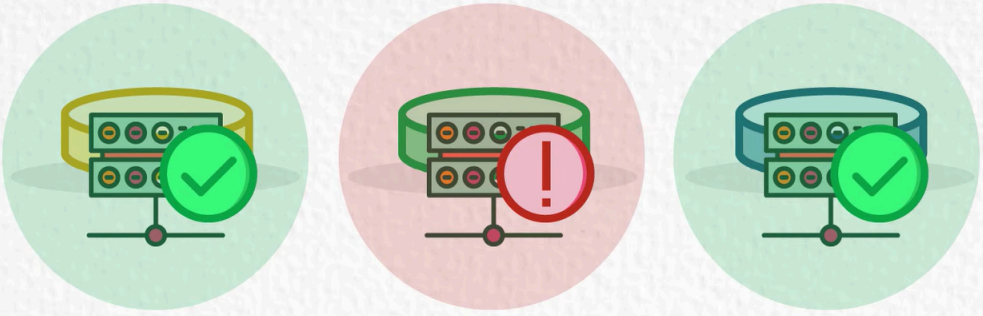
# Fragmentation

Horizontal



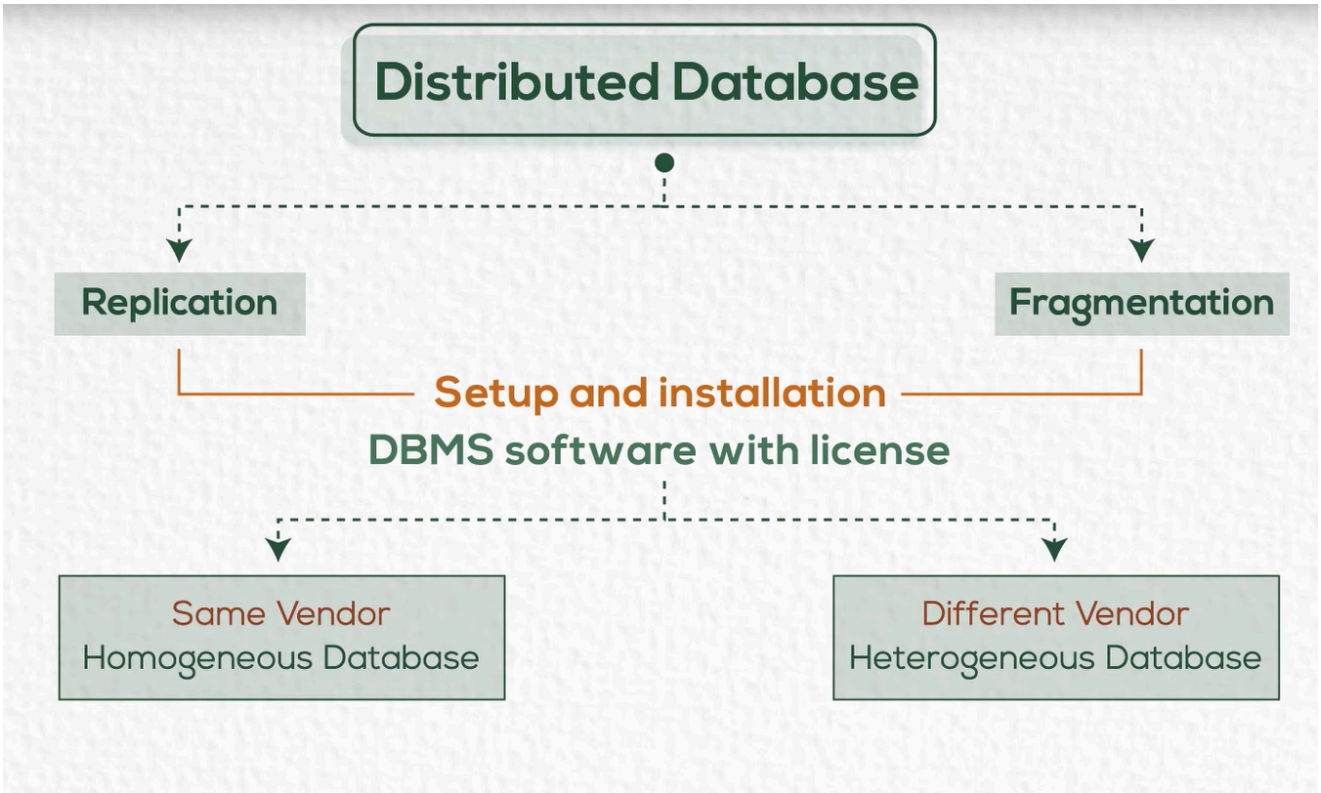
Vertical

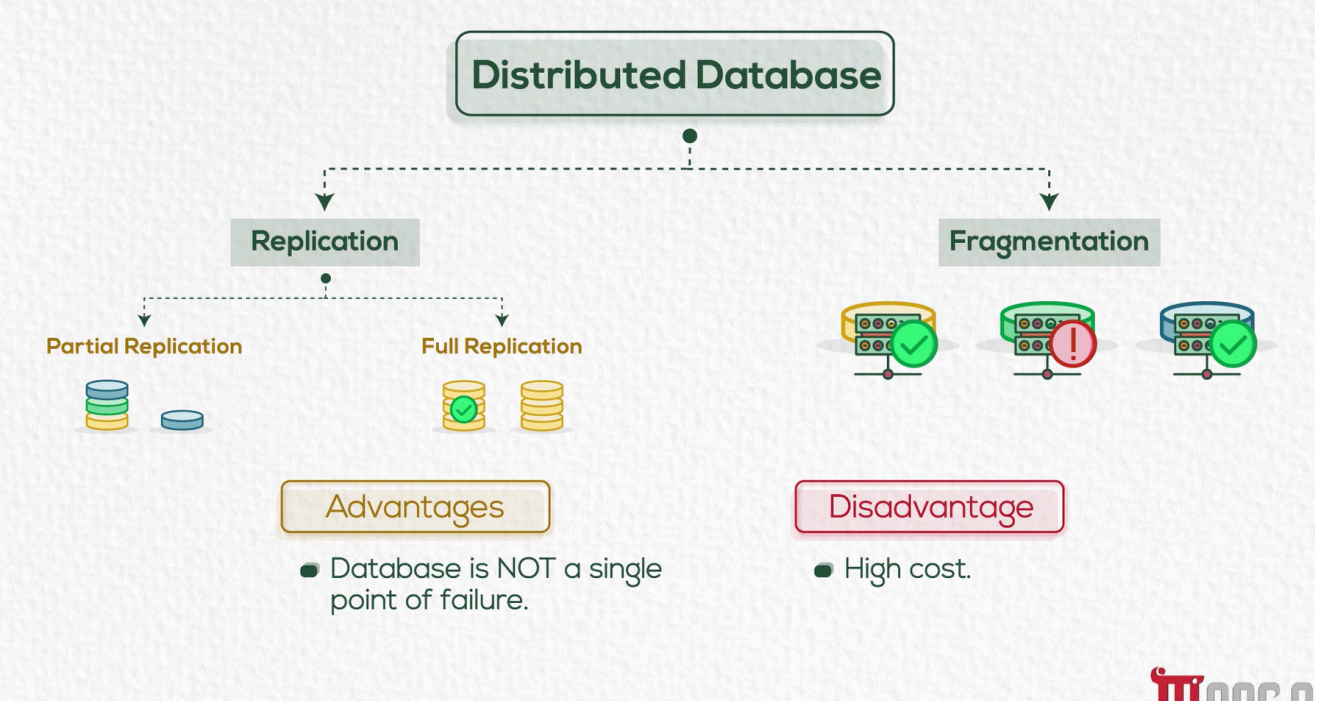
Hybrid



Advantages

- Database is NOT a single point of failure.



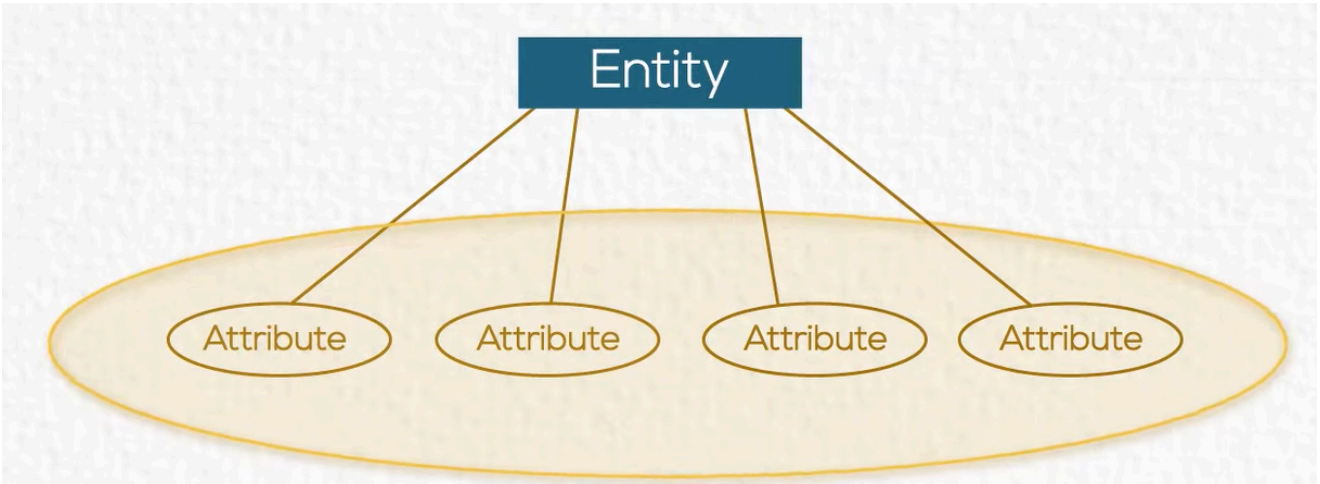



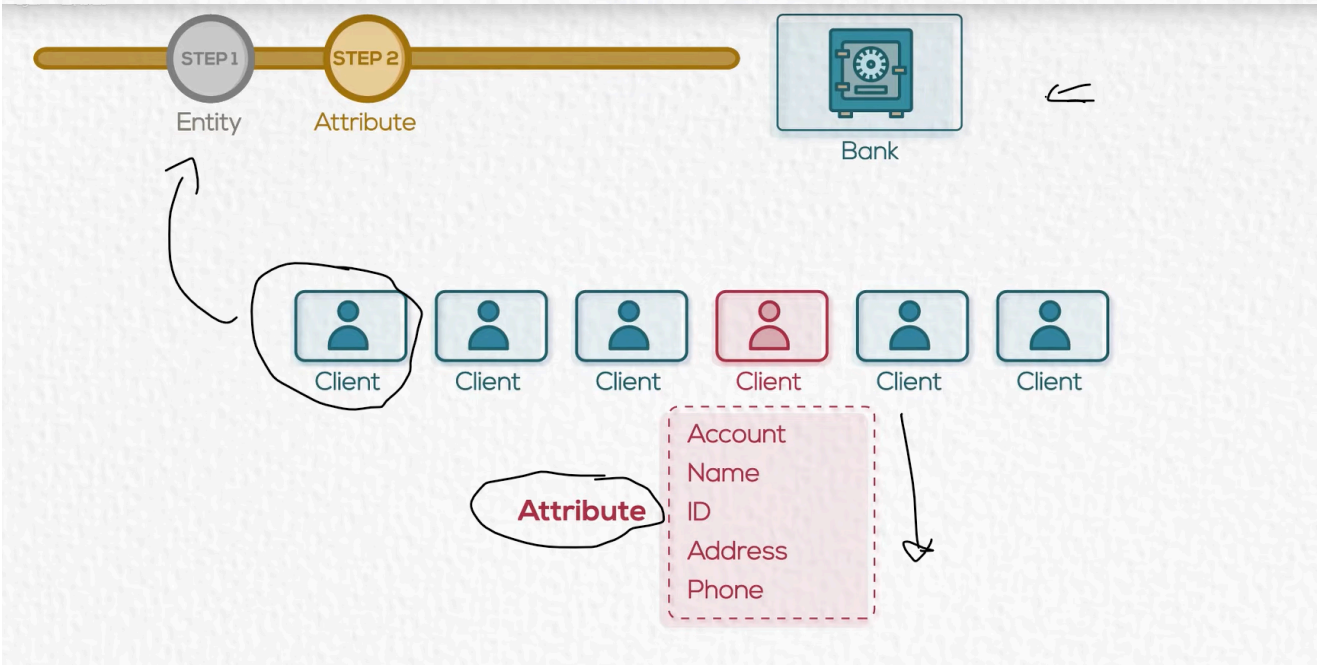

# Chapter 2 - Entity Relation Diagram

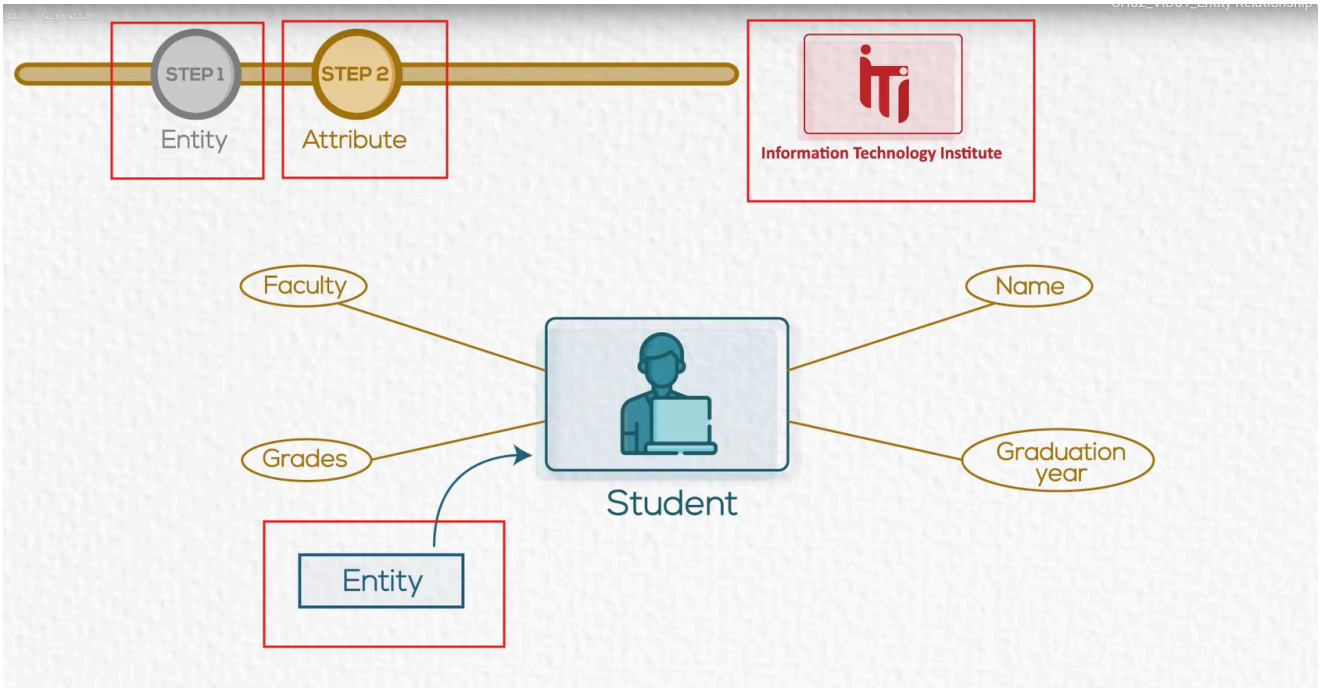
## Entity Relation Diagram (ERD)

- Identifies information required by the business by displaying the relevant **entities** and **relationships** between them.

**Entity** is a thing in the real world with an independent existence. Physical existence or conceptual existence.






## Entity Relationship Modeling

**عشان تقدر ترسمه الدايجرام عندك لازم تسأل نفسك شوية اسئلة**

*ايه هي الكيانات المستقلة اللي عندي*

1- What entities need to be described in the model?

*ايه هي الخصائص اللي بتوصف الكيان المستقل دا*

2- What characteristics or attributes of those entities need to be recorded?

*هل فيه ميزه او مجموعة من المميزات تقدر تخليلي الكيان المستقل بتاعي دا مميز عن الباقي*

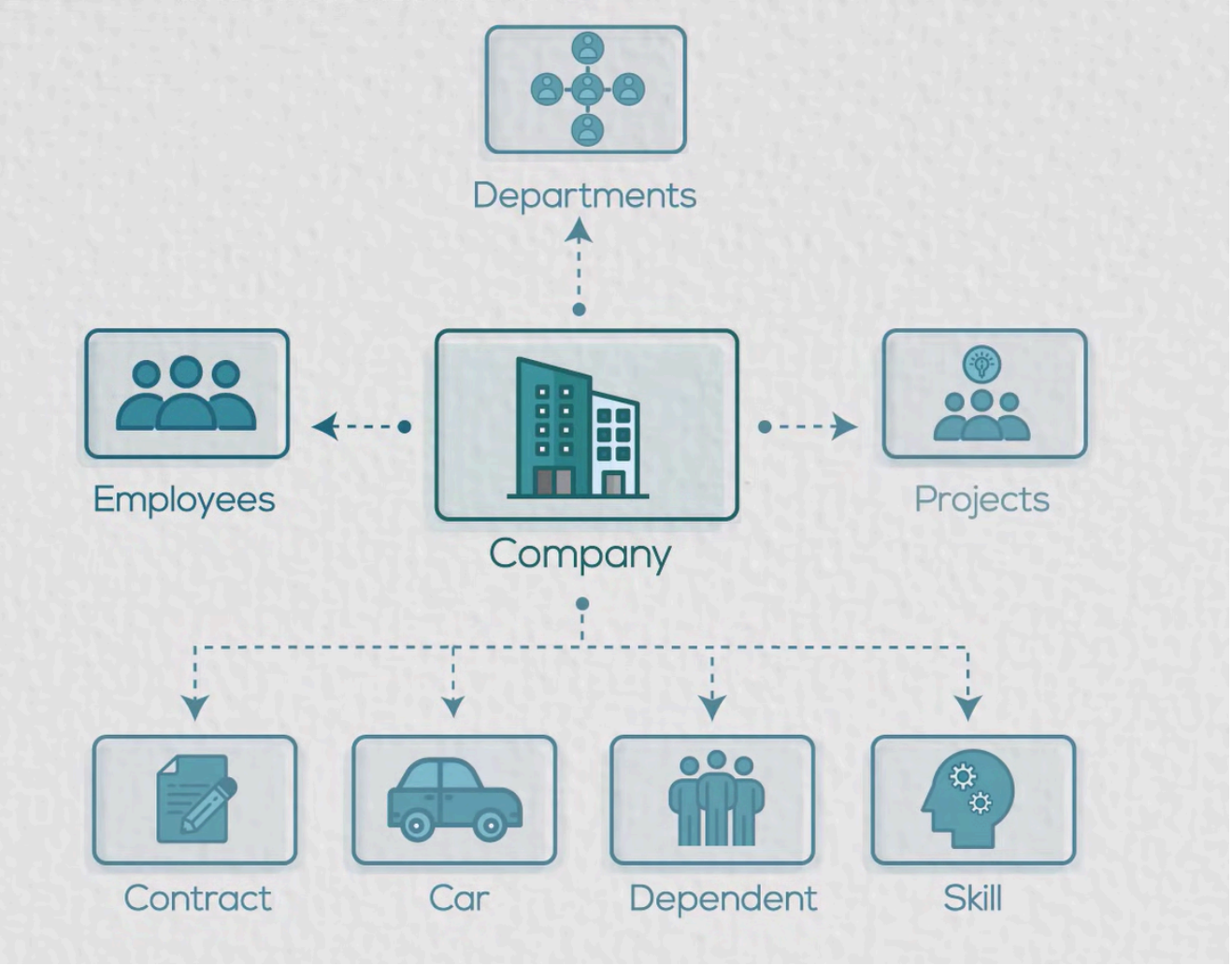
3- Can an attribute or a set of attributes be identified that will uniquely identify one specific occurrence of an entity?

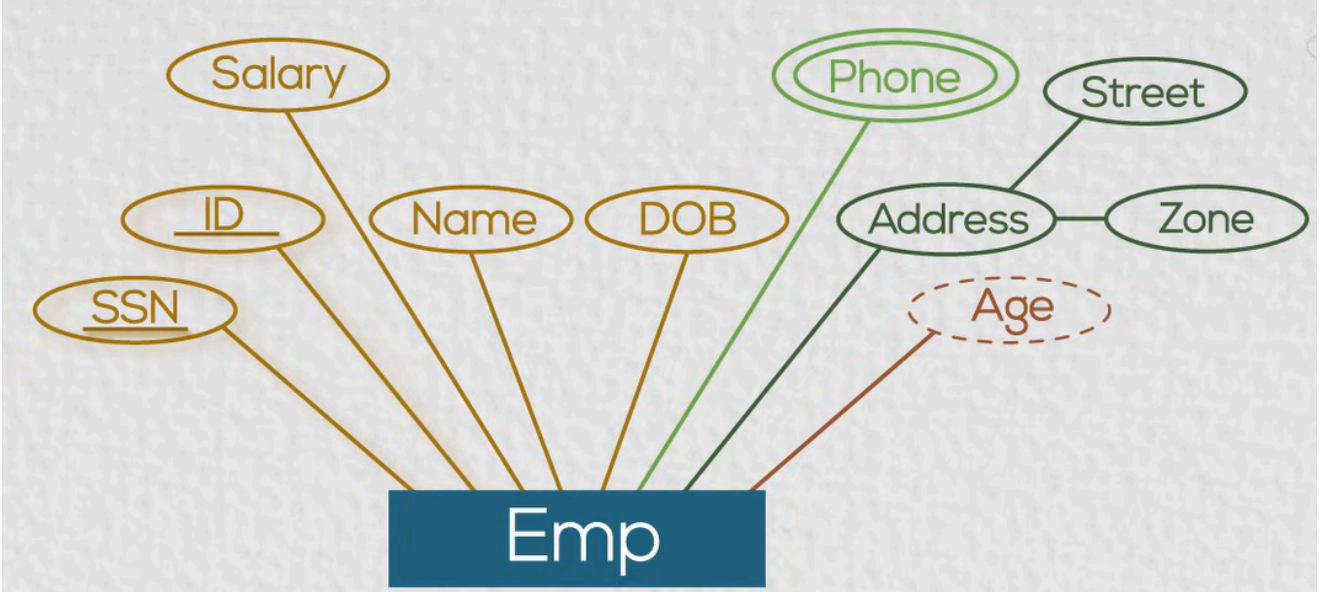
*تبدأ تحدد العلاقات بين كل الكيانات و بعضها*

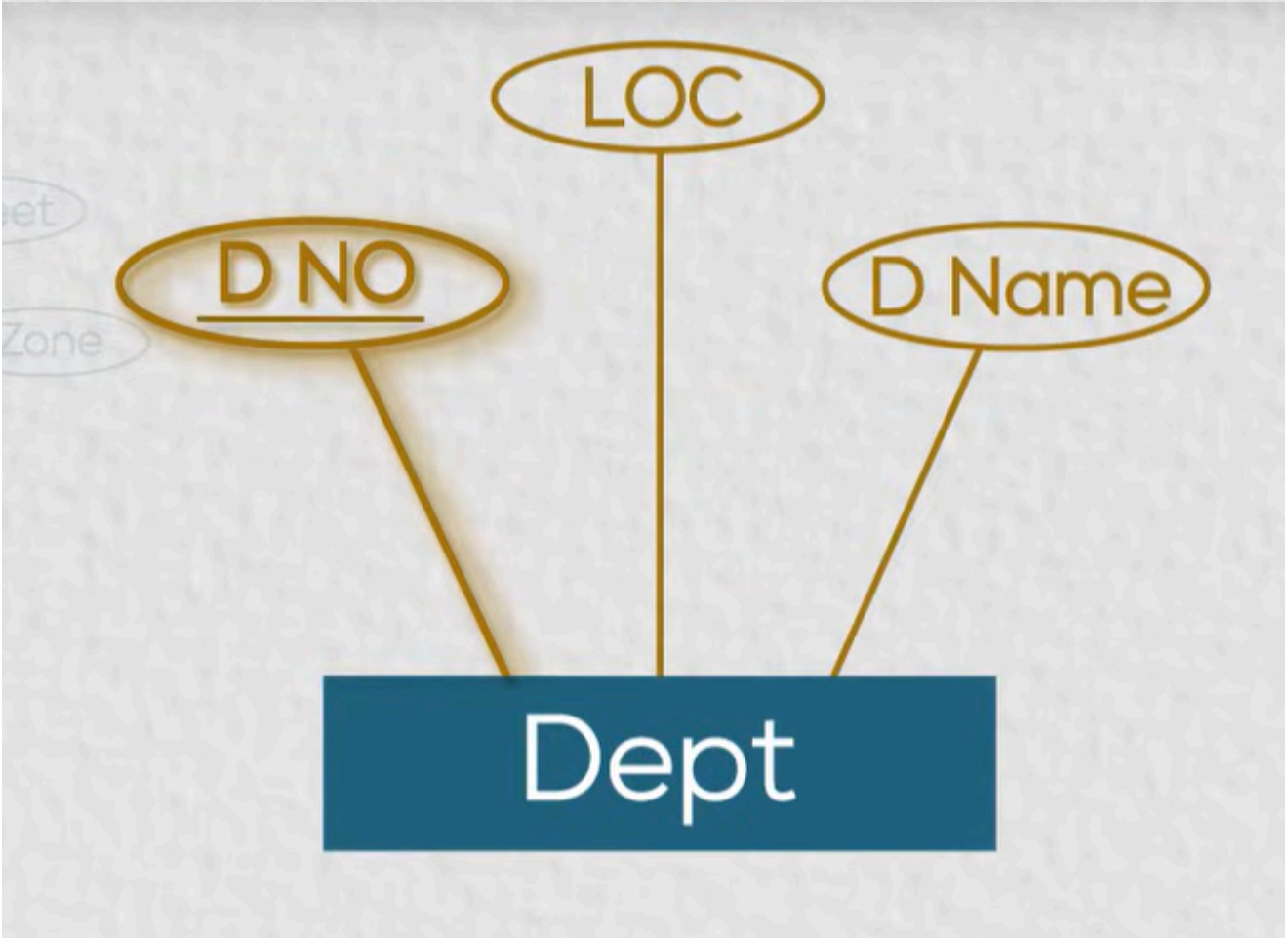
4- What associations or relationships exist between entities?

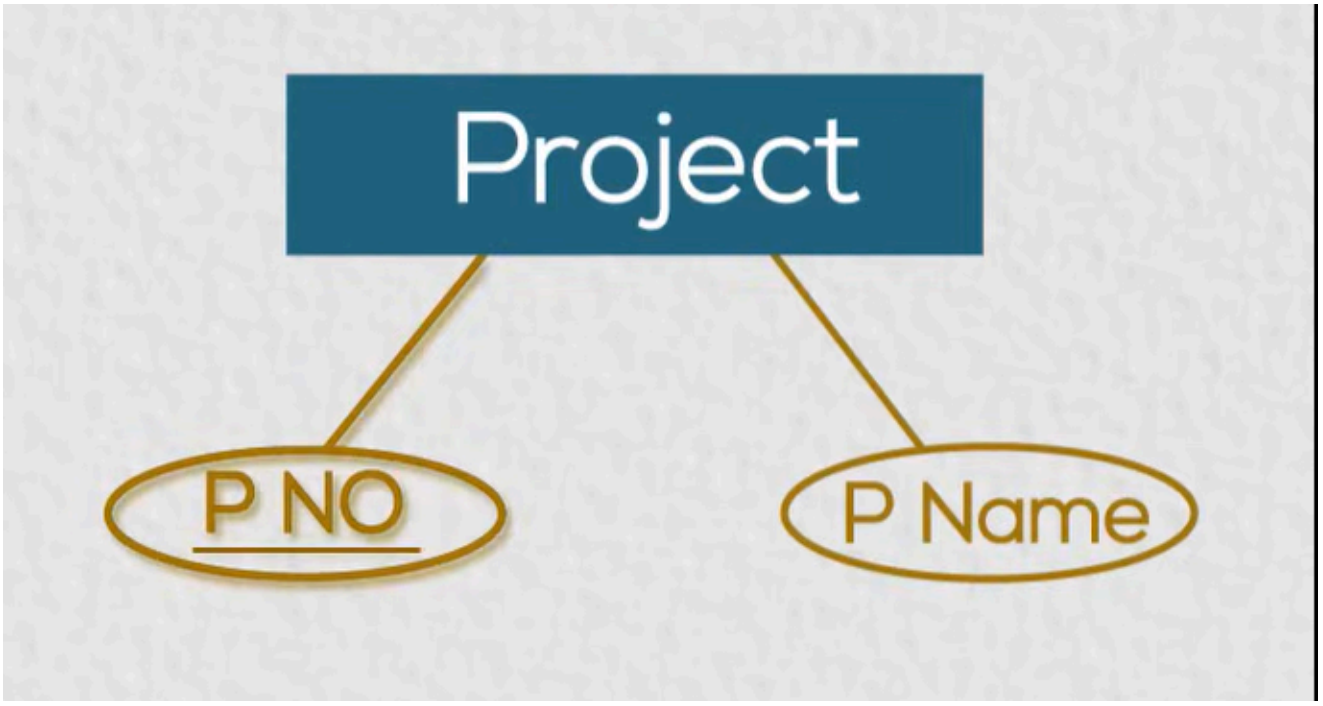
## Conceptual Design

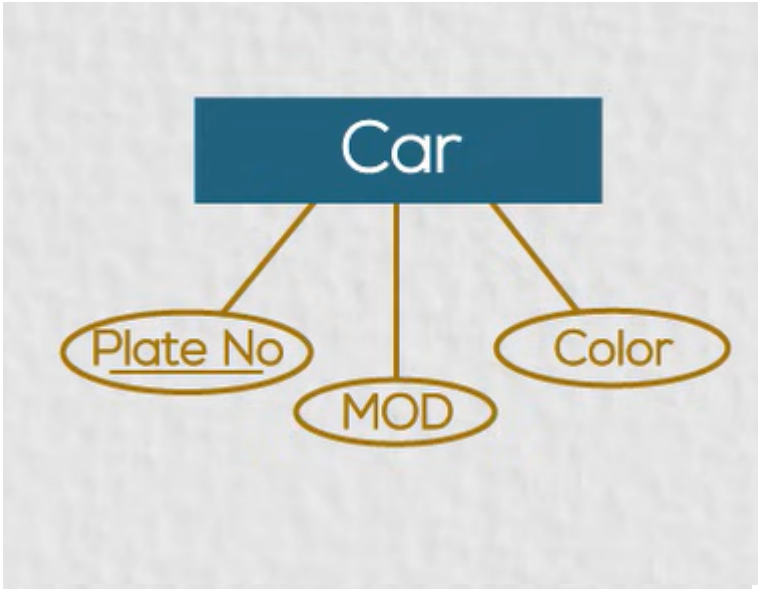

*Example that we work on it*

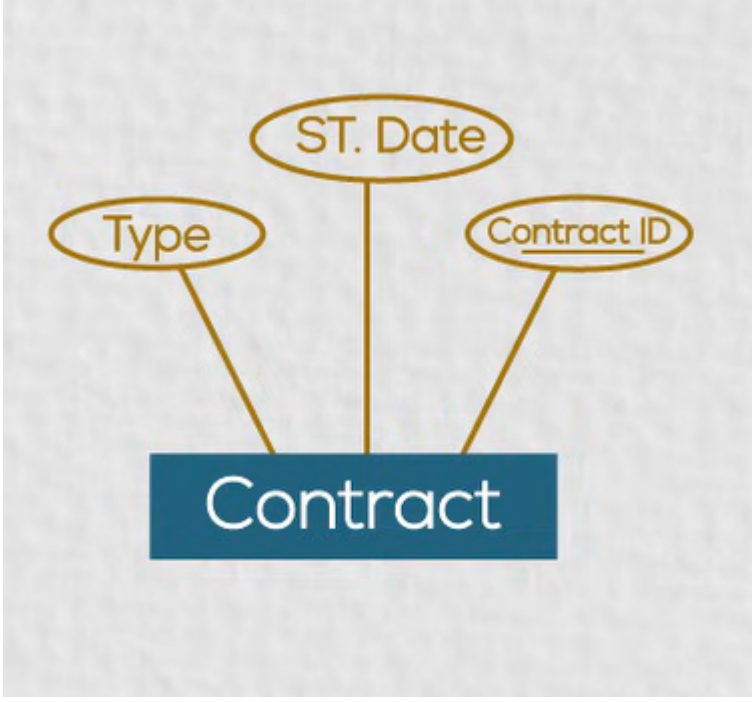


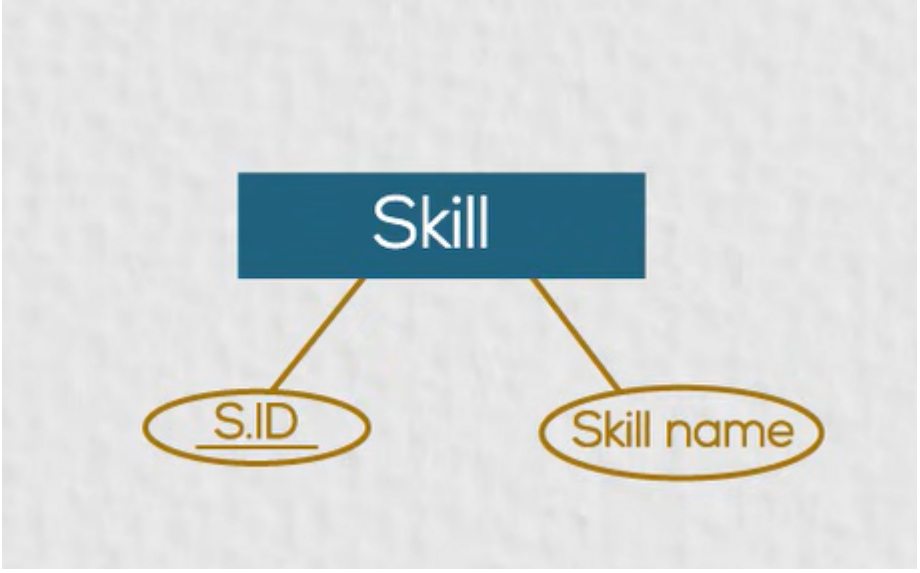

Salary
Phone
Street
ID
Name
DOB
Address
Zone
SSN
Age
Emp

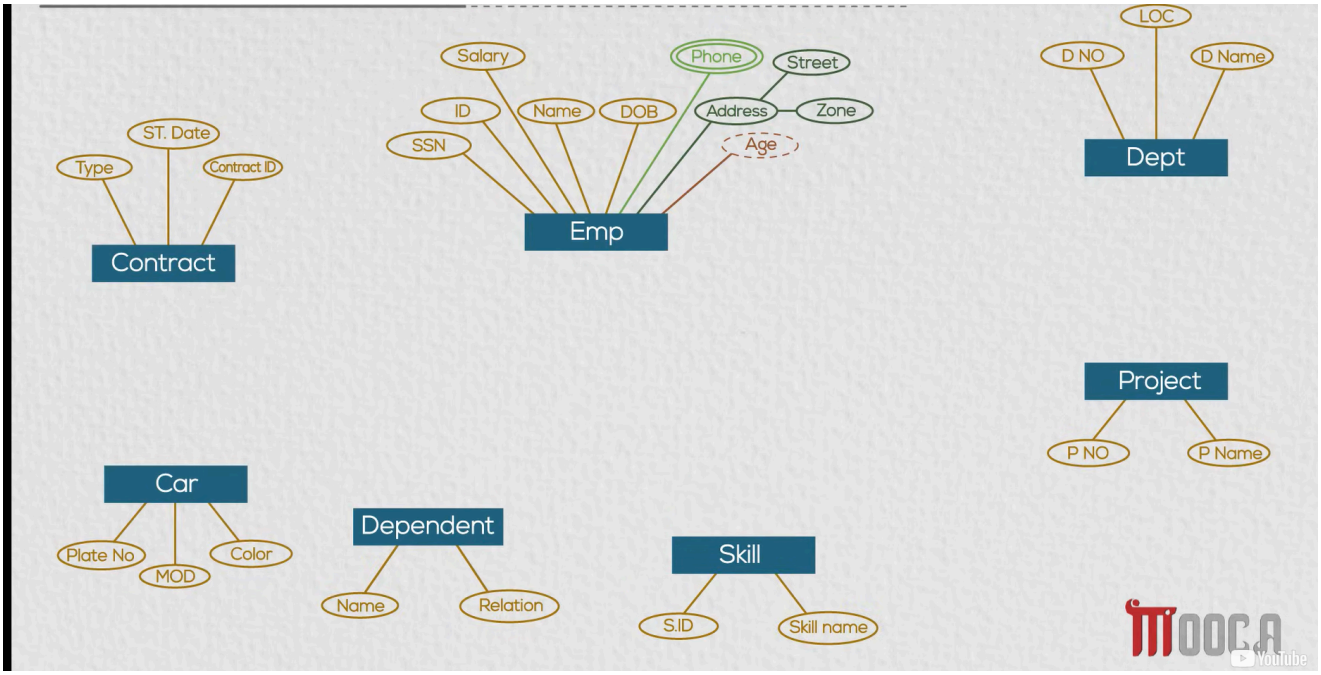

LOC
D NO
D Name
Dept


Project
P NO
P Name

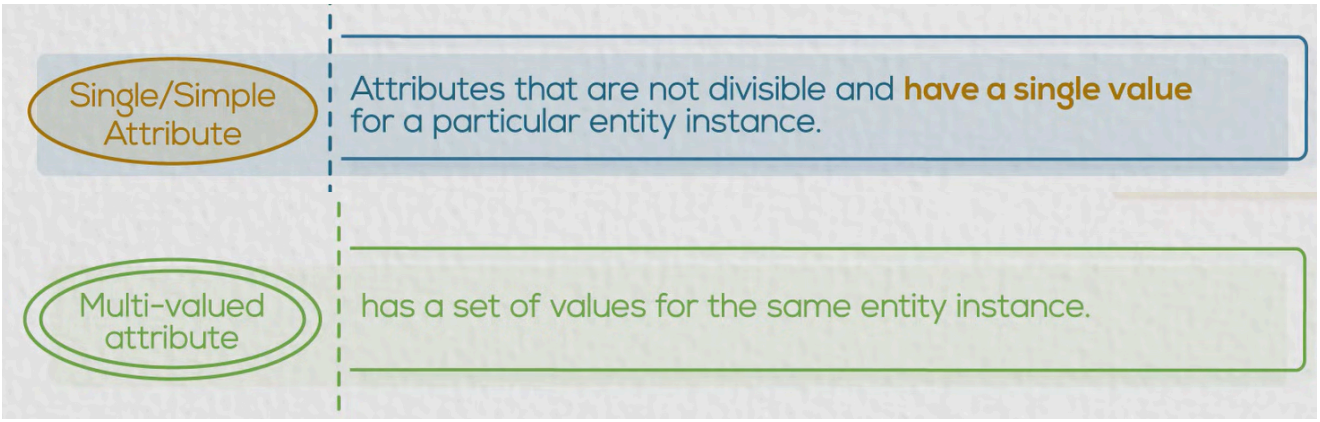

Car
Plate No
MOD
Color

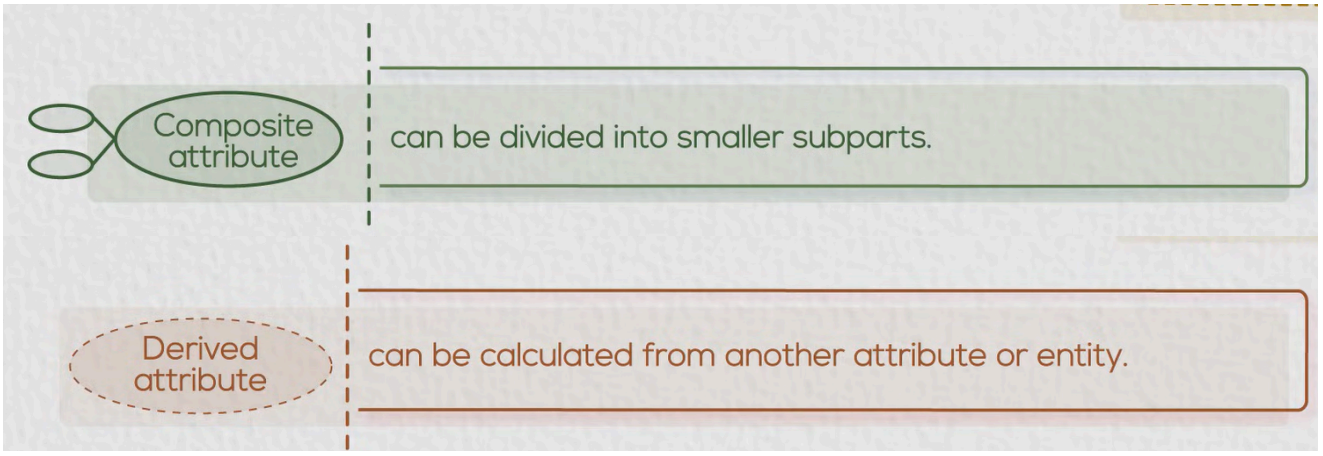

ST. Date
Type
Contract ID
Contract




## Types of Attributes





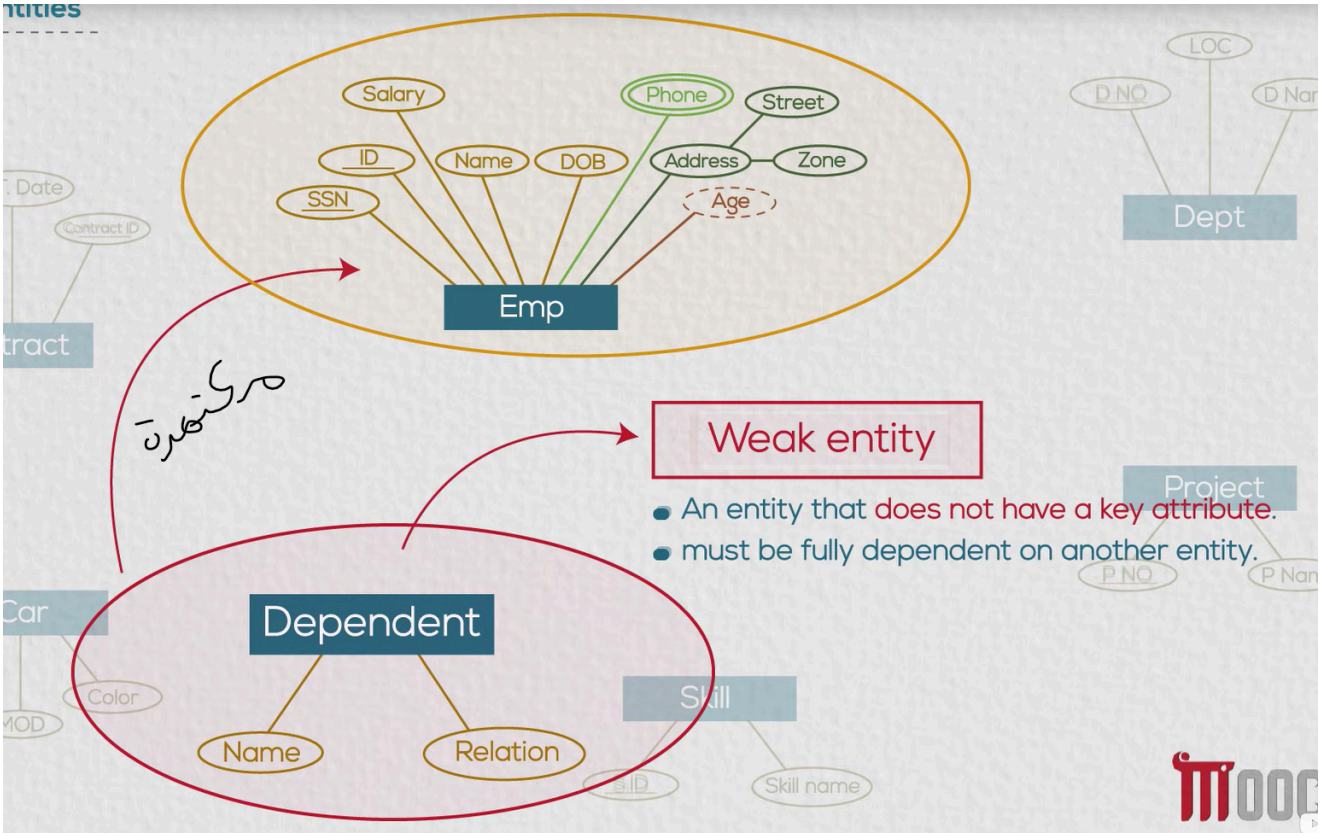

## Types of Entity

*ممكن تحدد ليها unique key في ميزه من مييزاتها بتخليها مختلفه عن الباقي او اقدر احددها بيها*

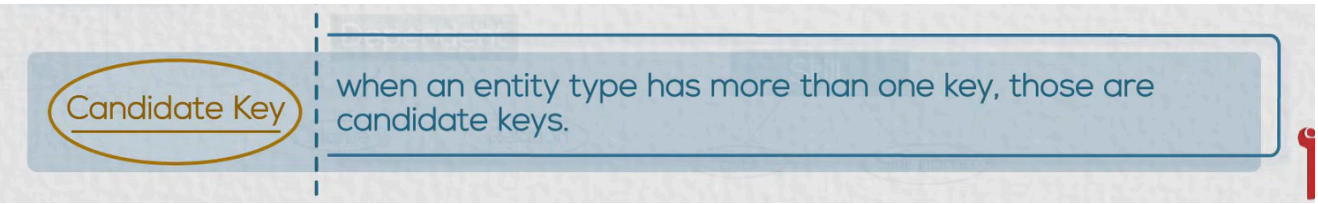

*بتحقق عندي شرطين*

*1- وهي مفيش ولا Attribute من بتوعها اقدر احددها بيها*

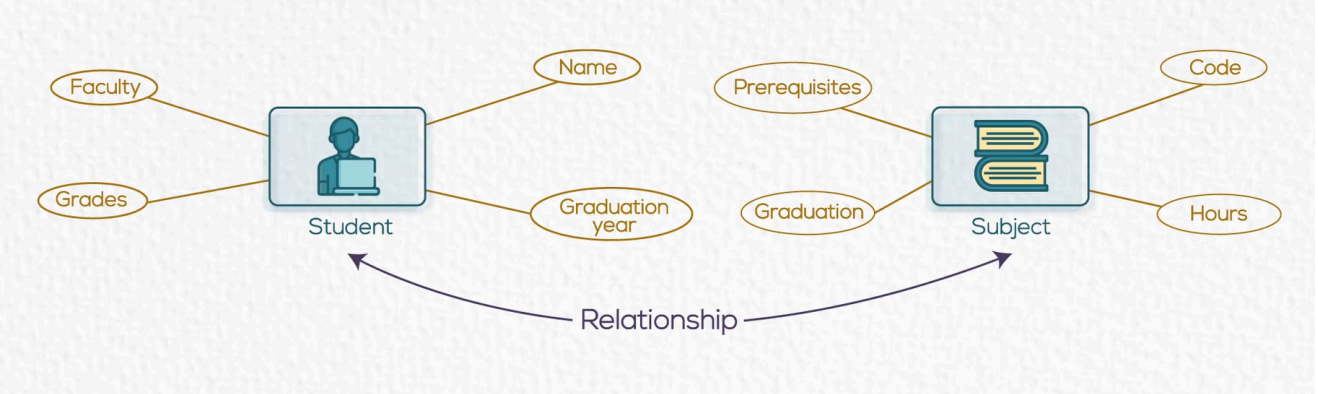
*2- وجودها معتمد اعتمادا كليا علي Entity تانية*

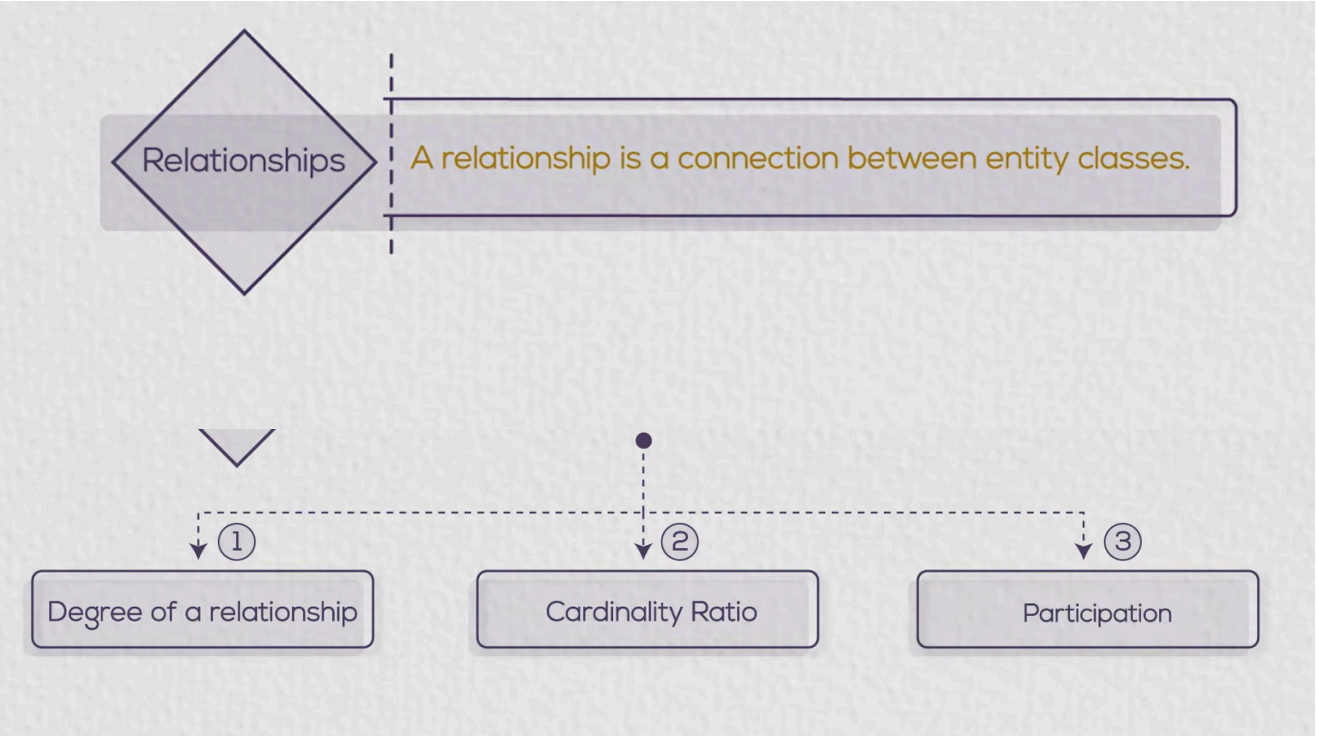



## Types of Keys




## Relationships



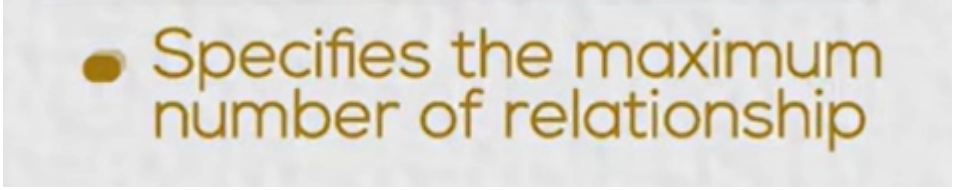

### Degree of Relationship

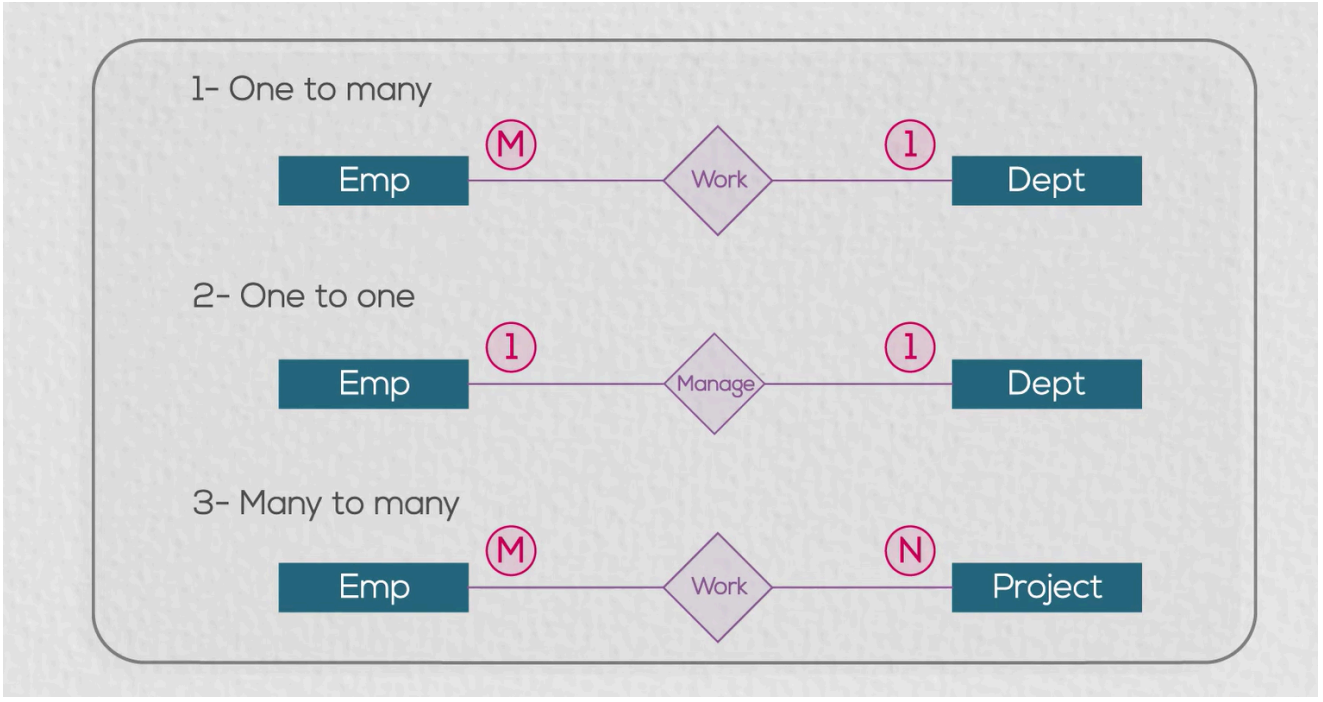
*العدد اللي بيشارك في العلاقة دي*


### Cardinality Ratio

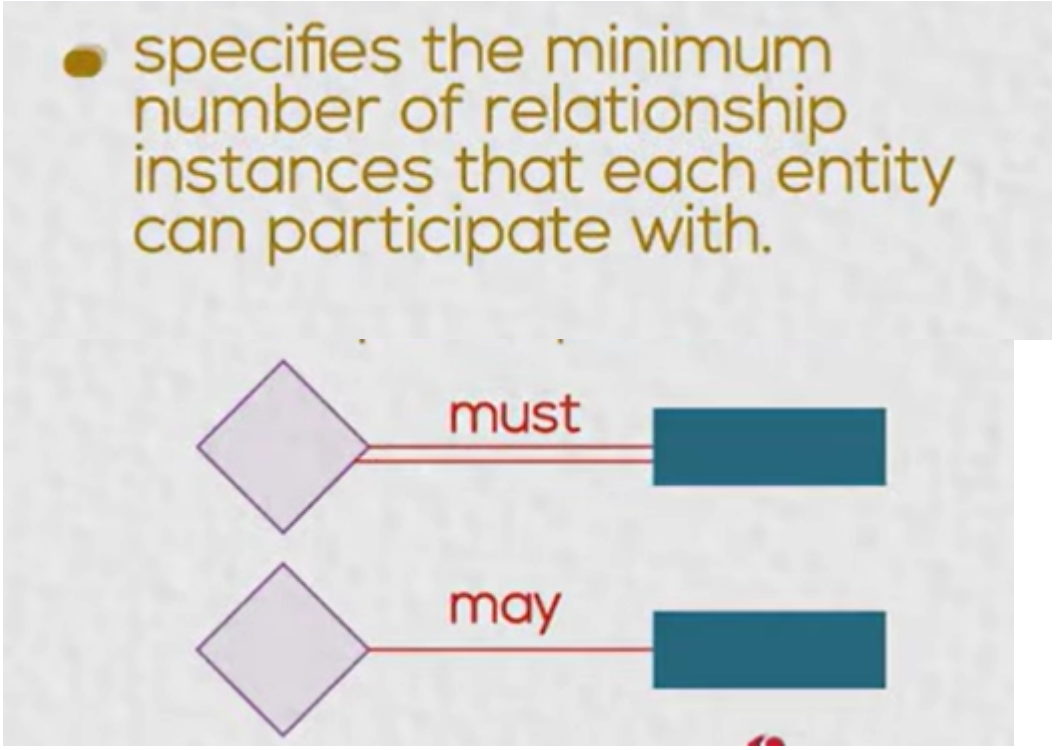
*هتروح تقف عند كل Entity اللي موجودين في العلاقة وليكن مثلا عندنا موظف و قسم هتروح تقف عن الموظف وتسال سؤال هل الموظف بيشتغل في اكتر من قسم و لا قسم وحد طبعا قسم واحد تروح*

*تحط واحد عن القسم و تقف عند القسم و تسال هل القسم بيشتغل فيه اكتر من موظف ولا موظف واحد ولا اكيد اكتر من موظف تروح تحط n عند الموظف وهكذا بقي*

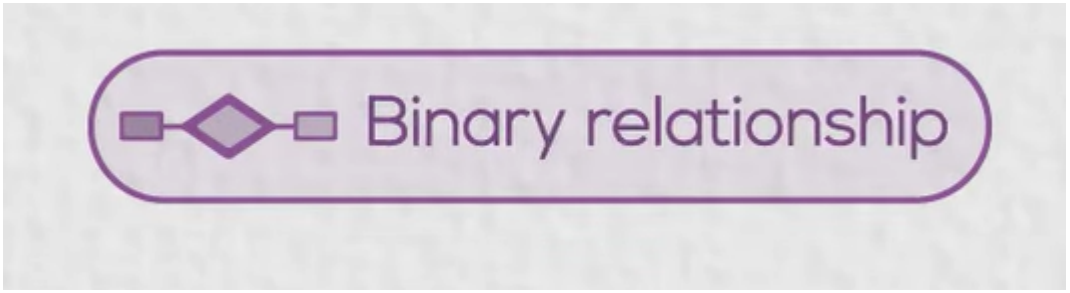


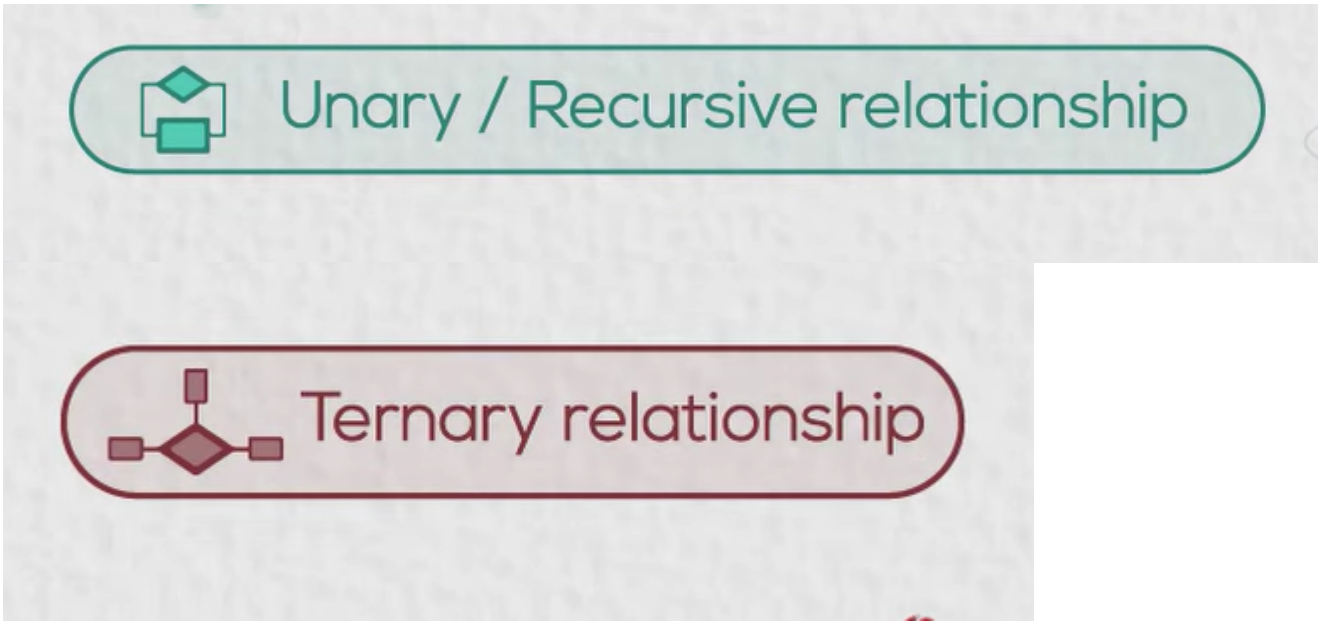

## Participation




## Types of Relationships

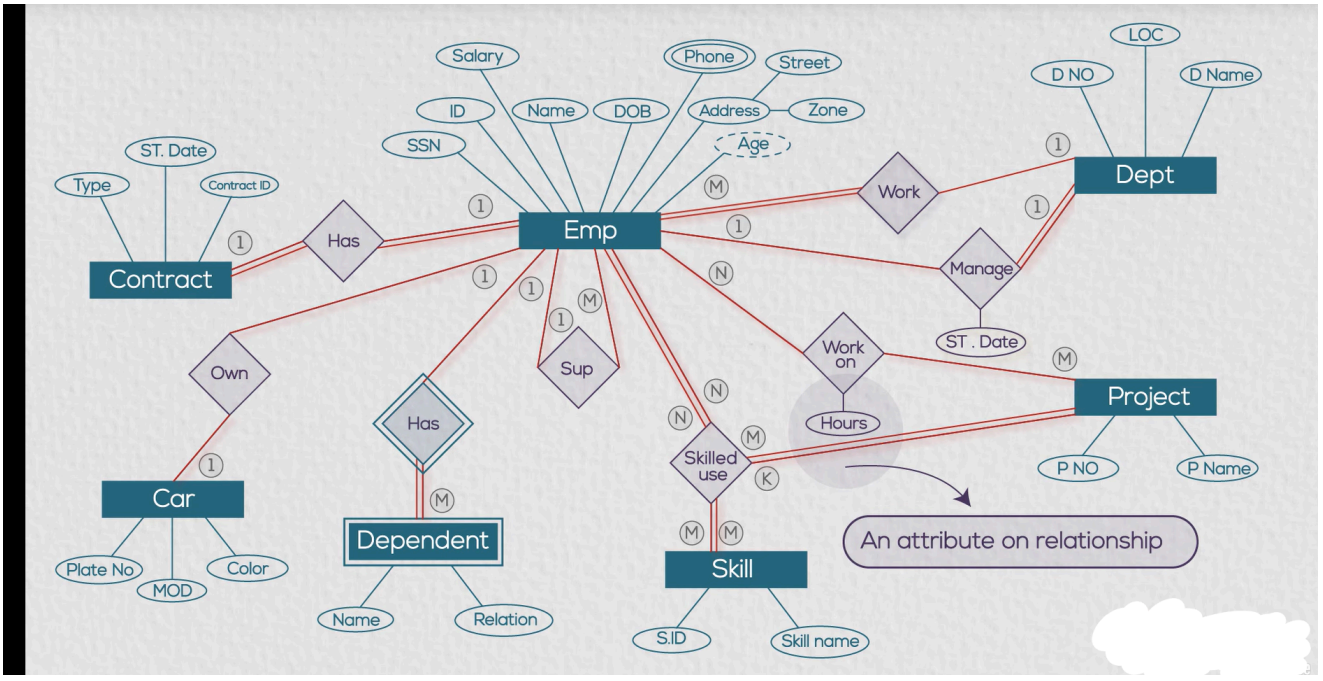



بين ال Entity ونفسها




بتكون بين ال Strong Entity and Weak Entity , وبيكون ليها خطلين فوق بعض كدا


## Final ERD



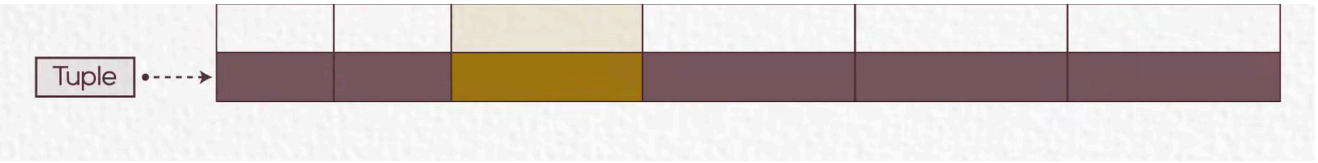
# Chapter 3 - ERD Mapping to Tables

## Mapping Strong and Weak Entities

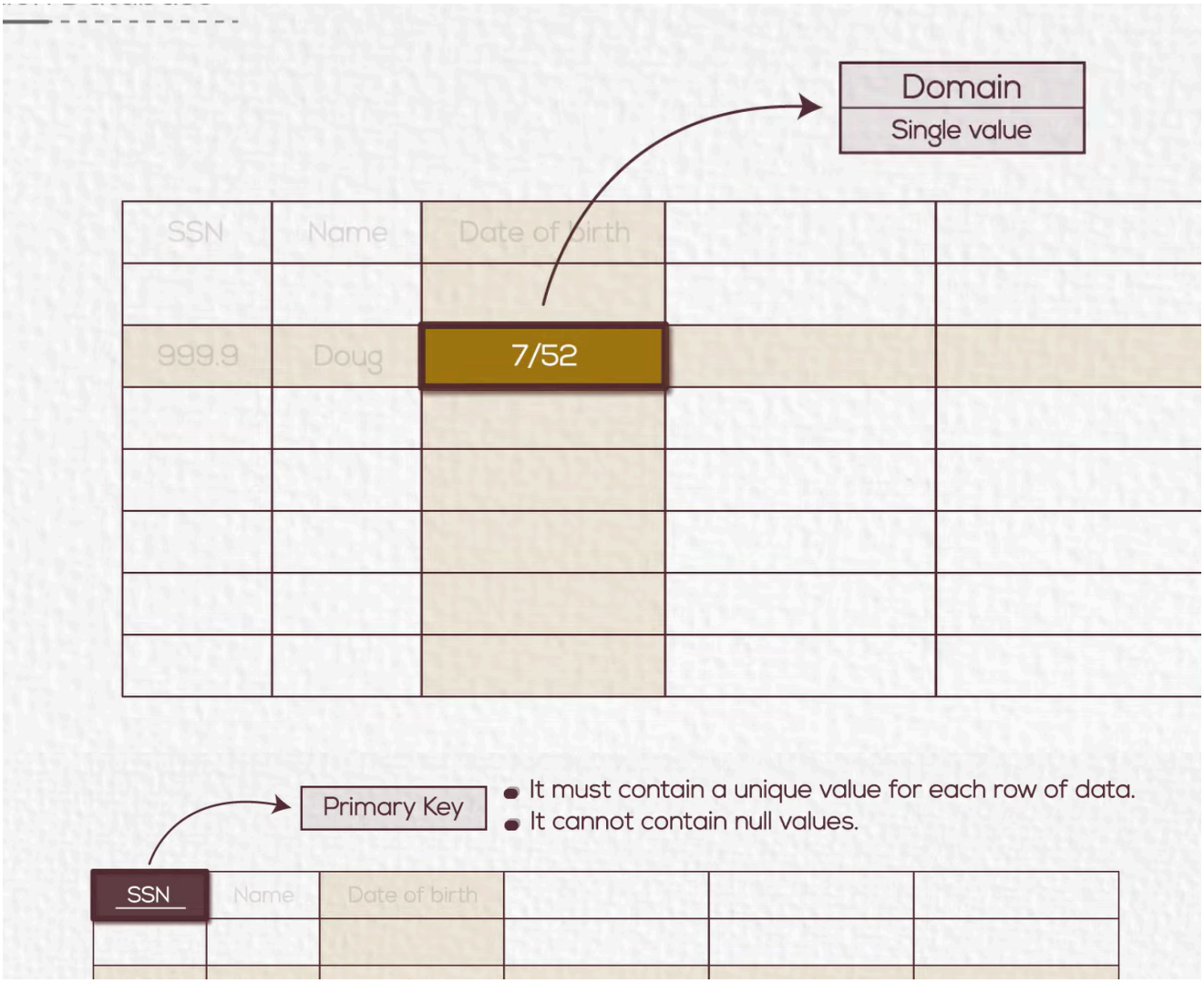

## Relation Database

| SSN | Name | Date of birth | | | |
|---|---|---|---|---|---|
| | | | | | |
| 999.9 | Doug | 7/52 | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

*Row Jl*



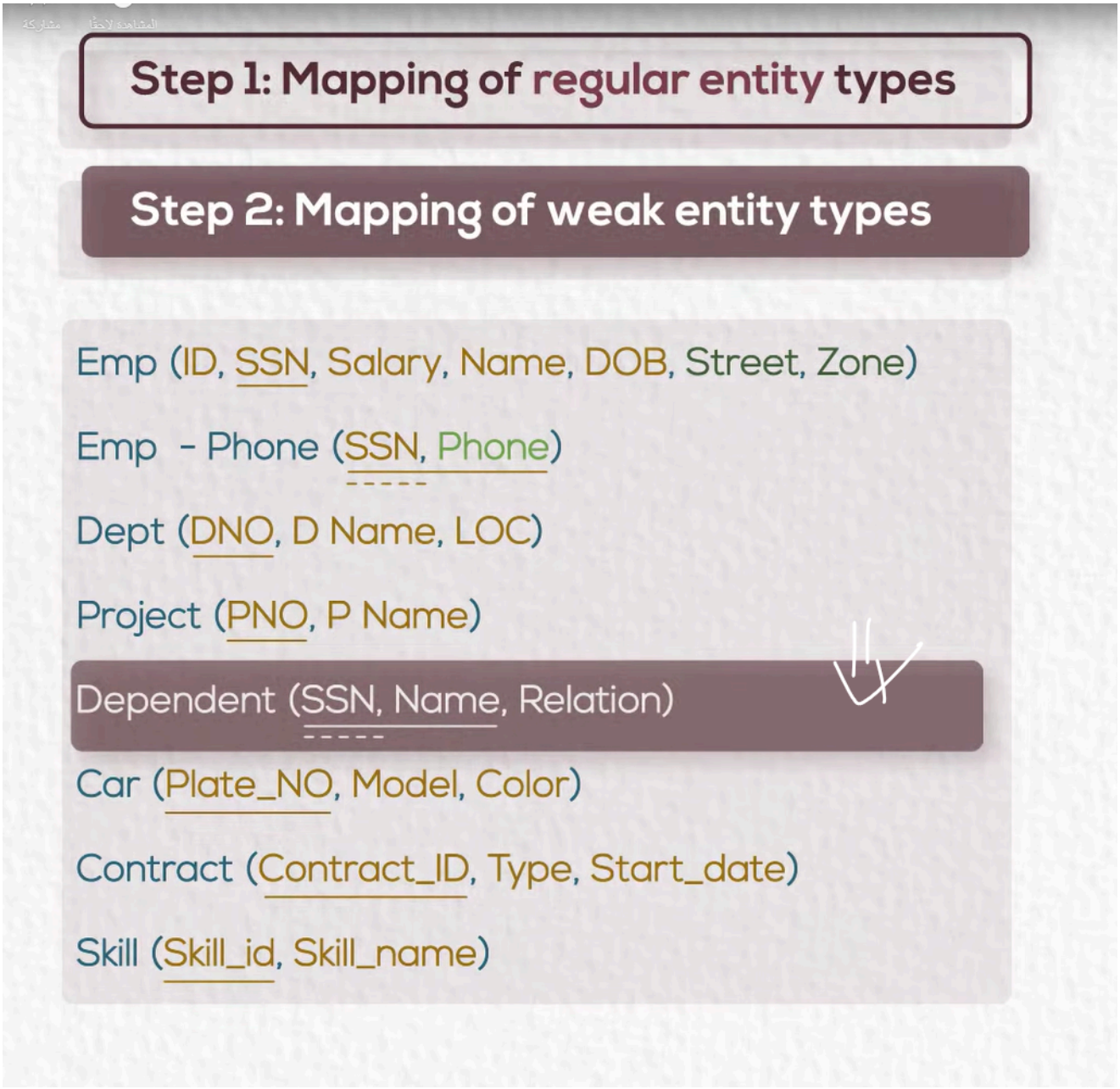

Column
SSN
999.9




## How Can Convert Conceptual Design to Logical Design

1- Create Table for Eeach Entity
2- Choose one of key attributes to be the primary key
3- Seperate Multivalue attribute in new table with Primary key of Entity as Forign key and two together is primary key
4- In table of weak entity must put the primary key of strong entity

## Mapping Strong and Weak Entities




## Mapping Relation Types

*في حالة العلاقة لما تكون احادية او ثنائية سواء 1:1 او N:1 هتاخد ال primary key بتاع اللي مش N وتحطه as foreign key*

## Step 3: Mapping of relationship types **Binary / Unary 1:N**

Add FK to N-side table

Emp (ID, SSN, Salary, Name, DOB, Street, Zone DNO

Emp - Phone (SSN, Phone)

Dept (DNO, D Name, LOC)

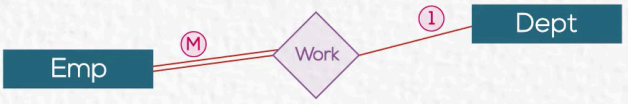
Project (PNO, P Name)

Dependent (SSN, Name, Relation)

Car (Plate_NO, Model, Color)

Contract (Contract_ID, Type, Start_date)

Skill (Skill_id, Skill_name



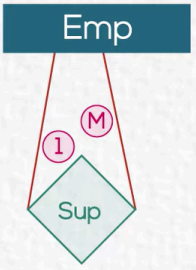

Emp (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN

Emp - Phone (SSN, Phone)

Dept (DNO, D Name, LOC)

Project (PNO, P Name)

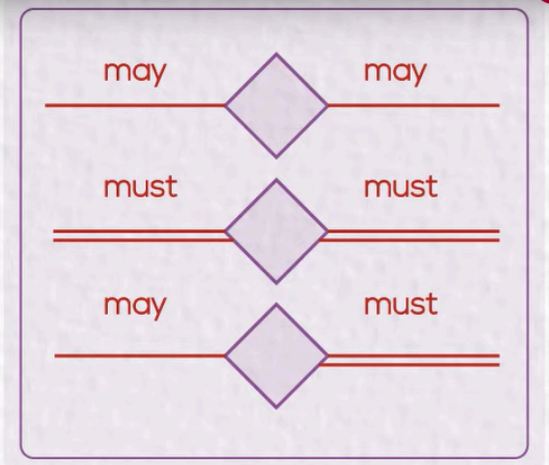
Dependent (SSN, Name, Relation)



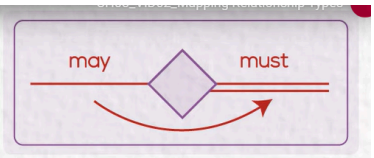

## Binary / Unary M:N

Add FKs to the new table for both parent tables

## Binary / Unary 1:1



## Step 5: Mapping of relationship types **Binary / Unary 1:1**




Emp (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN

Emp - Phone (SSN, Phone)

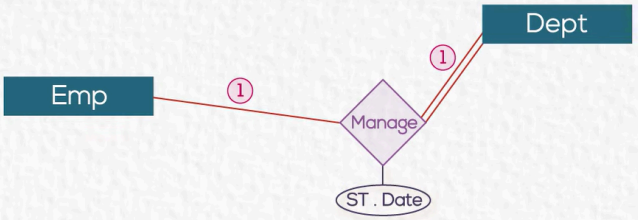
Dept (DNO, D Name, LOC, SSN

Project (PNO, P Name)

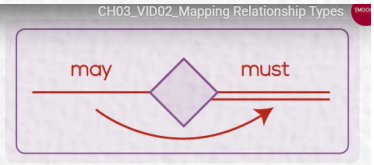
Dependent (SSN, Name, Relation)

Car (Plate_NO, Model, Color)

Contract (Contract_ID, Type, Start_date)




## Step 5: Mapping of relationship types **Binary / Unary 1:1**




Emp (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN

Emp - Phone (SSN, Phone)

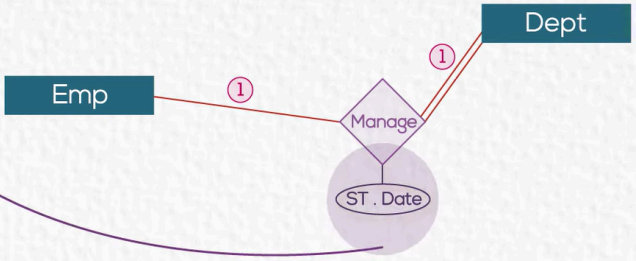
Dept (DNO, D Name, LOC, MGR _SSN ST. Date

Project (PNO, P Name)

Dependent (SSN, Name, Relation)

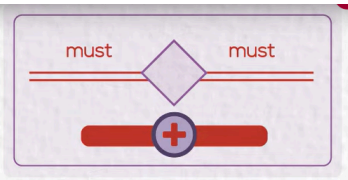
Car (Plate_NO, Model, Color)

Contract (Contract_ID, Type, Start_date)




بتندمج الجدولين في جدول واحد

## Step 5: Mapping of relationship types **Binary / Unary 1:1**




Emp (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN, Plate_NO

Emp - Phone (SSN, Phone)

Dept (DNO, D Name, LOC, MGR _SSN, ST. Date

Project (PNO, P Name)

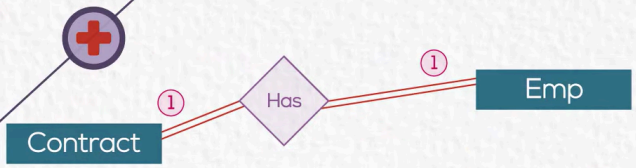
Dependent (SSN, Name, Relation)

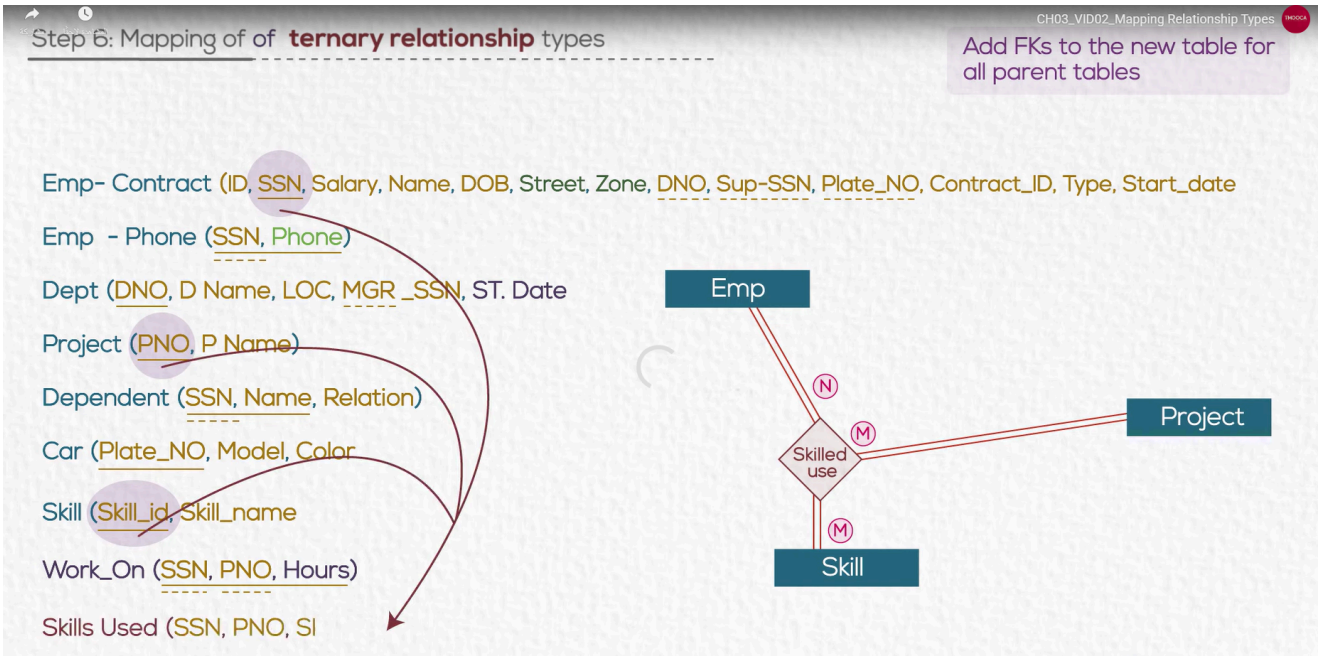
Car (Plate_NO, Model, Color

Contract (Contract_ID, Type, Start_date)

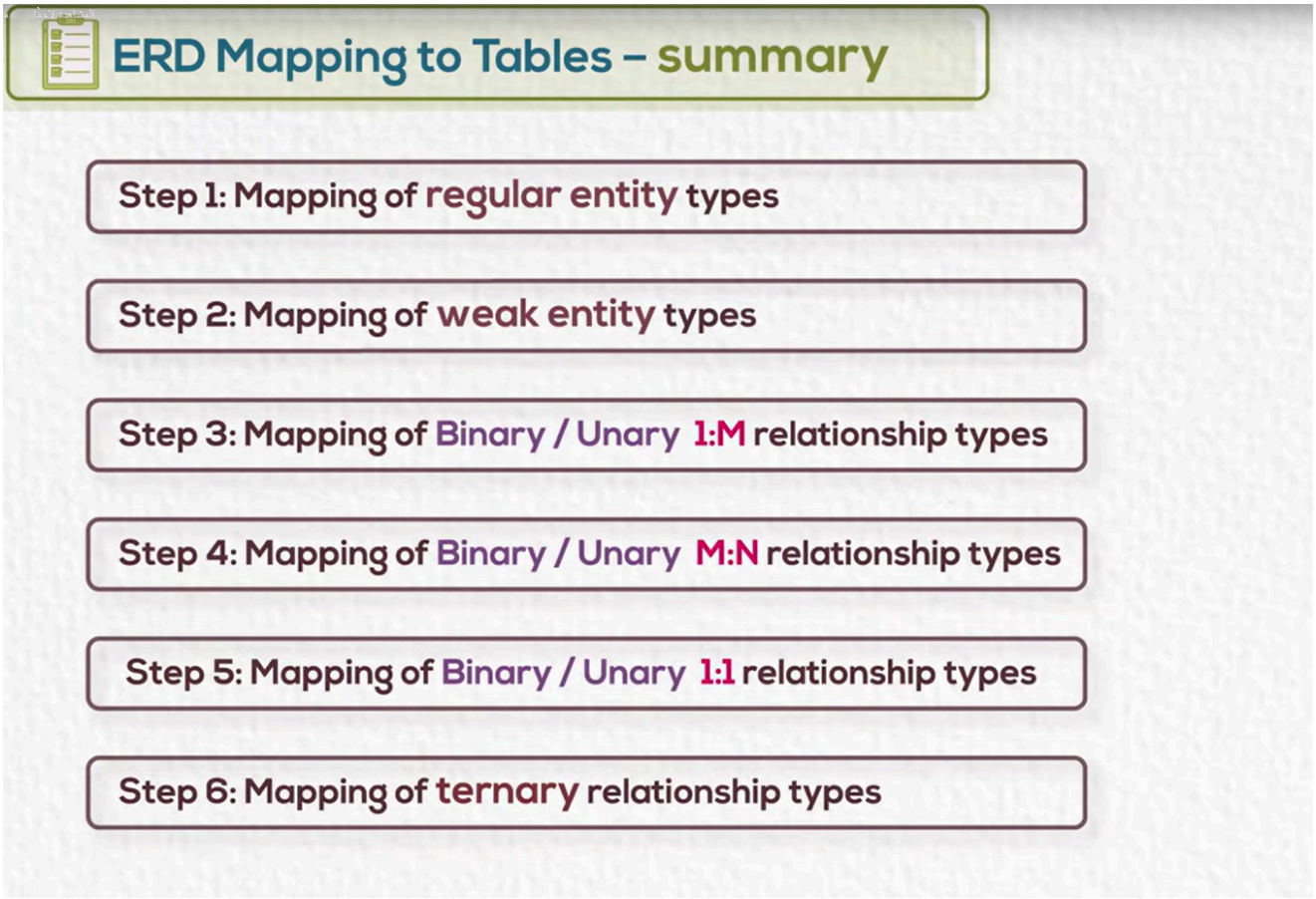
Skill (Skill_id, Skill_name

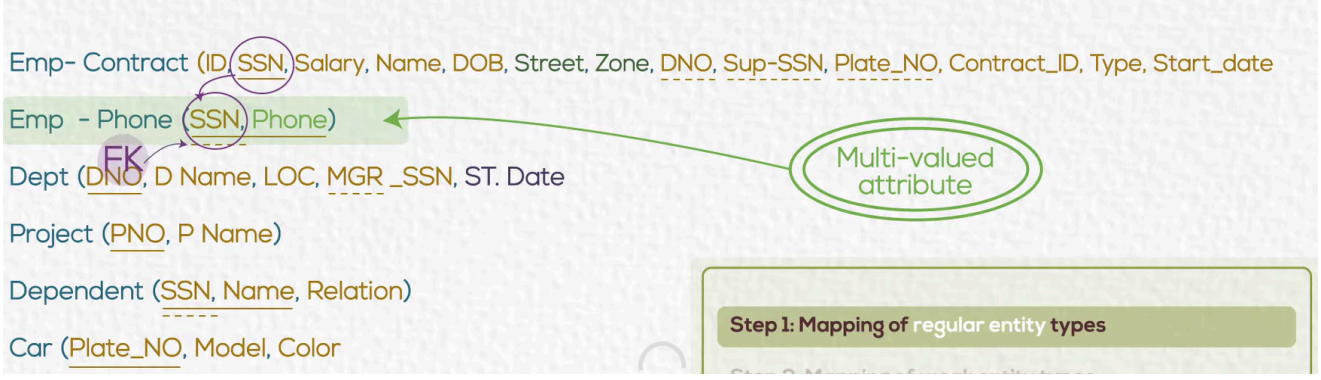
Work_On (SSN, PNO, Hours)

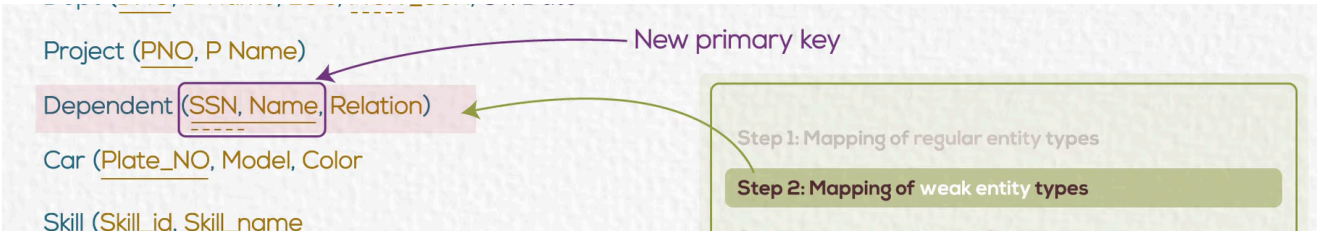



Step 6: Mapping of ternary relationship types
CH03_VID02_Mapping Relationship Types
Add FKs to the new table for all parent tables
Emp- Contract (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN, Plate_NO, Contract_ID, Type, Start_date
Emp - Phone (SSN, Phone)
Dept (DNO, D Name, LOC, MGR _SSN, ST. Date
Project (PNO, P Name)
Dependent (SSN, Name, Relation)
Car (Plate_NO, Model, Color
Skill (Skill_id, Skill_name
Work_On (SSN, PNO, Hours)
Skills Used (SSN, PNO, Sl
Emp
N
M
Skilled use
Project
M
Skill

## Conclusion


ERD Mapping to Tables – summary
Step 1: Mapping of regular entity types
Step 2: Mapping of weak entity types
Step 3: Mapping of Binary / Unary 1:M relationship types
Step 4: Mapping of Binary / Unary M:N relationship types
Step 5: Mapping of Binary / Unary 1:1 relationship types
Step 6: Mapping of ternary relationship types

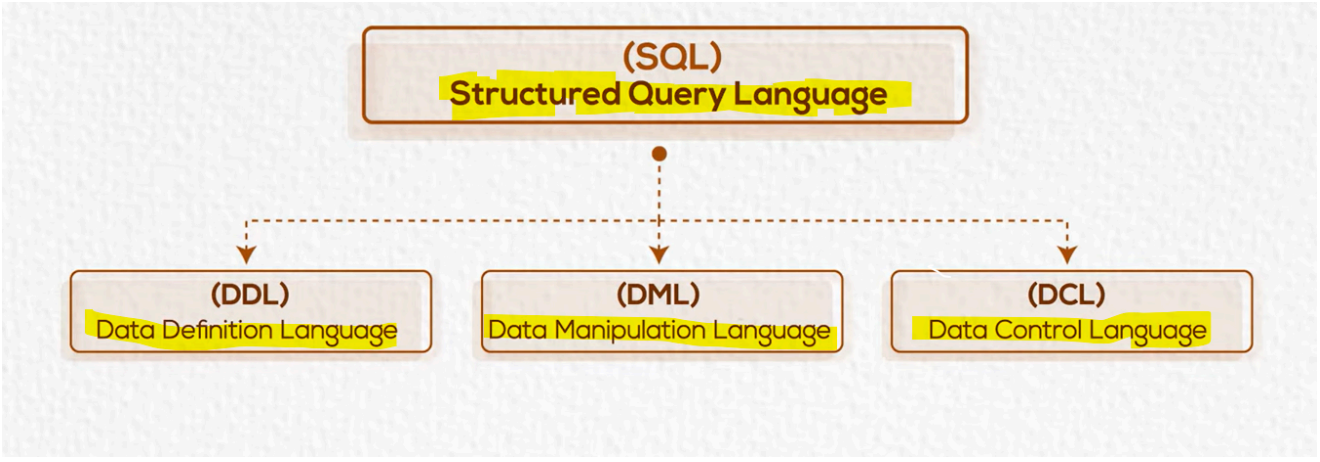

Emp- Contract (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN, Plate_NO, Contract_ID, Type, Start_date
Emp - Phone (SSN, Phone)
FK
Dept (DNO, D Name, LOC, MGR _SSN, ST. Date
Project (PNO, P Name)
Dependent (SSN, Name, Relation)
Car (Plate_NO, Model, Color
Multi-valued attribute
Step 1: Mapping of regular entity types





# Chapter 4 - Structured Query Language (SQL)

## Database Schema & Constraints




### Database Schema

- A schema is a **group of related objects in a database.**
- There is **one owner** of a schema who has **access to manipulate the structure** of any object in the schema.

### Data types

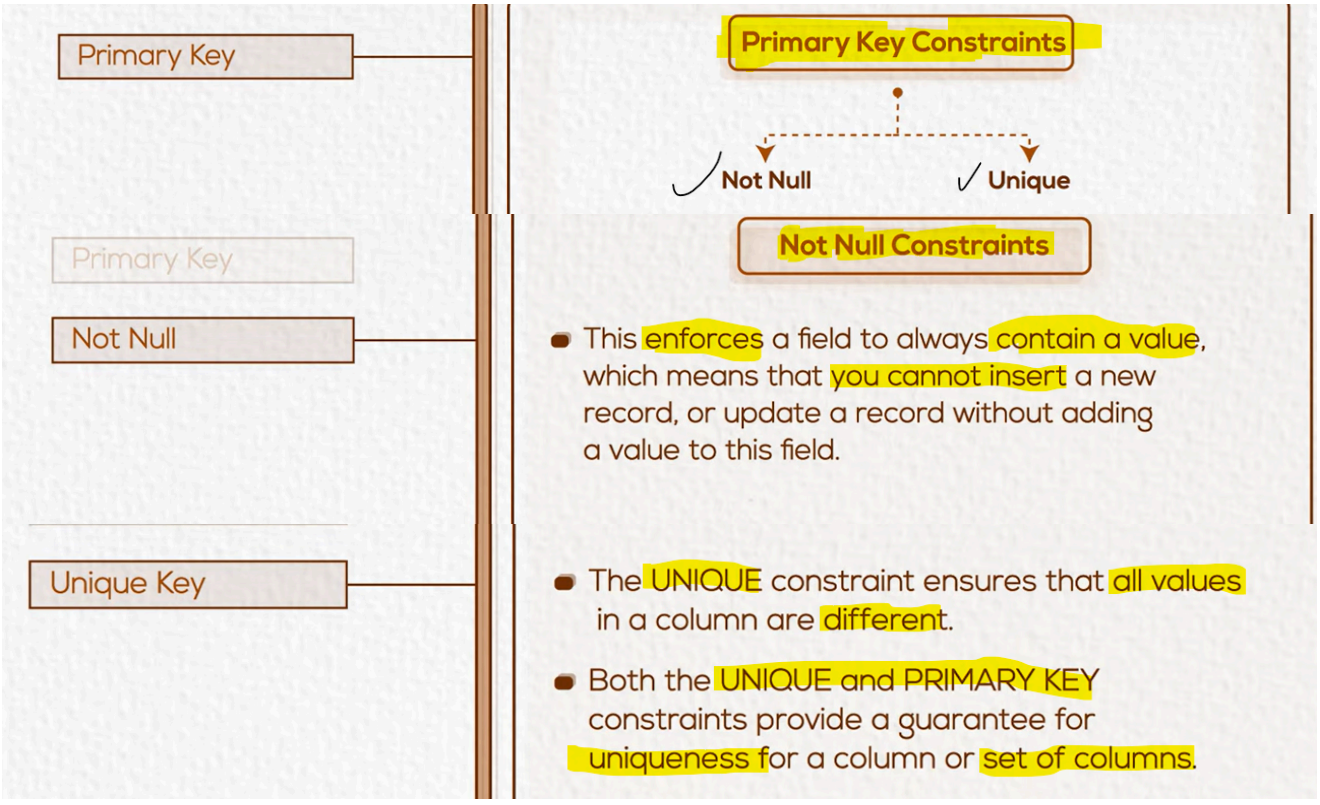
- A data type determines the type of data that can be stored in a database column.
  The most commonly used data types are:

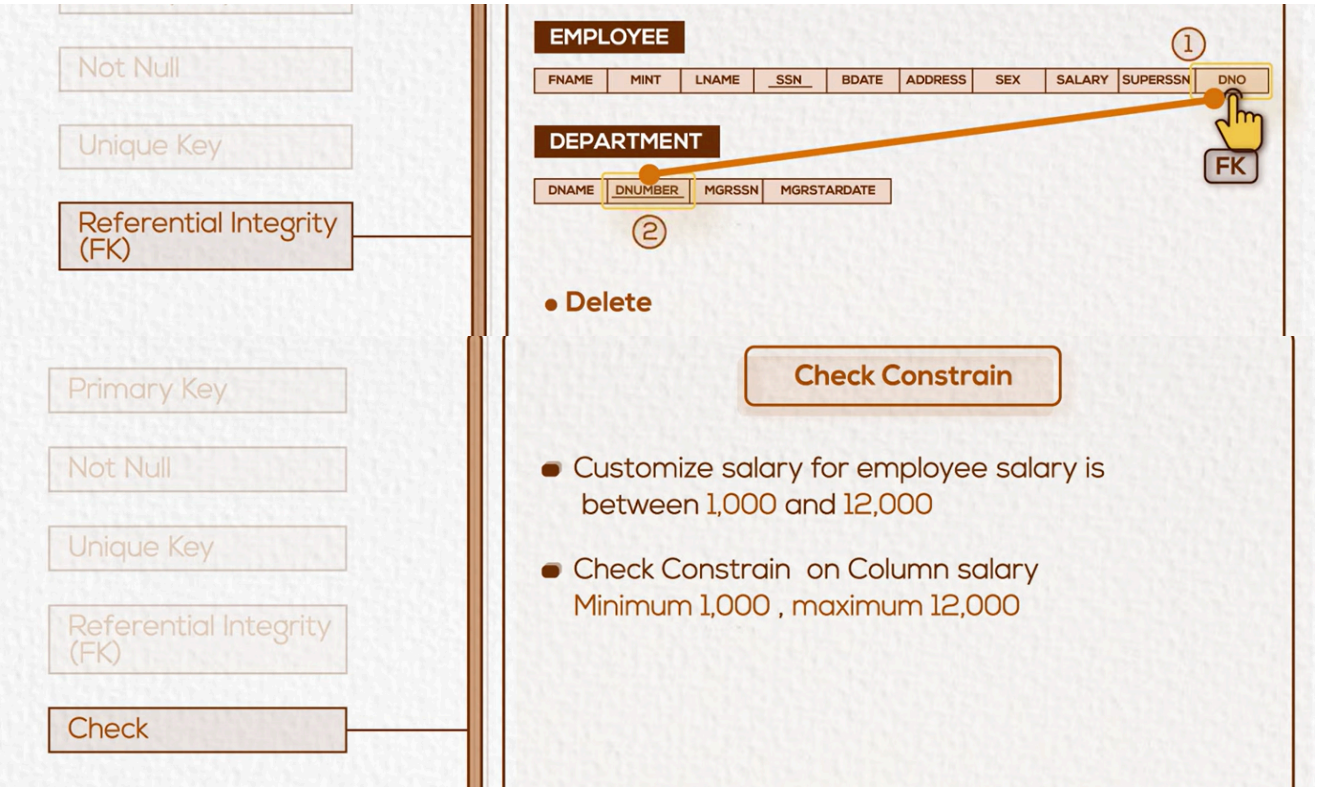
- Alphanumeric
- Numeric
- Date and Time
- Characters
- Variable Characters
- Integer
- Float data type

## Database Constraints

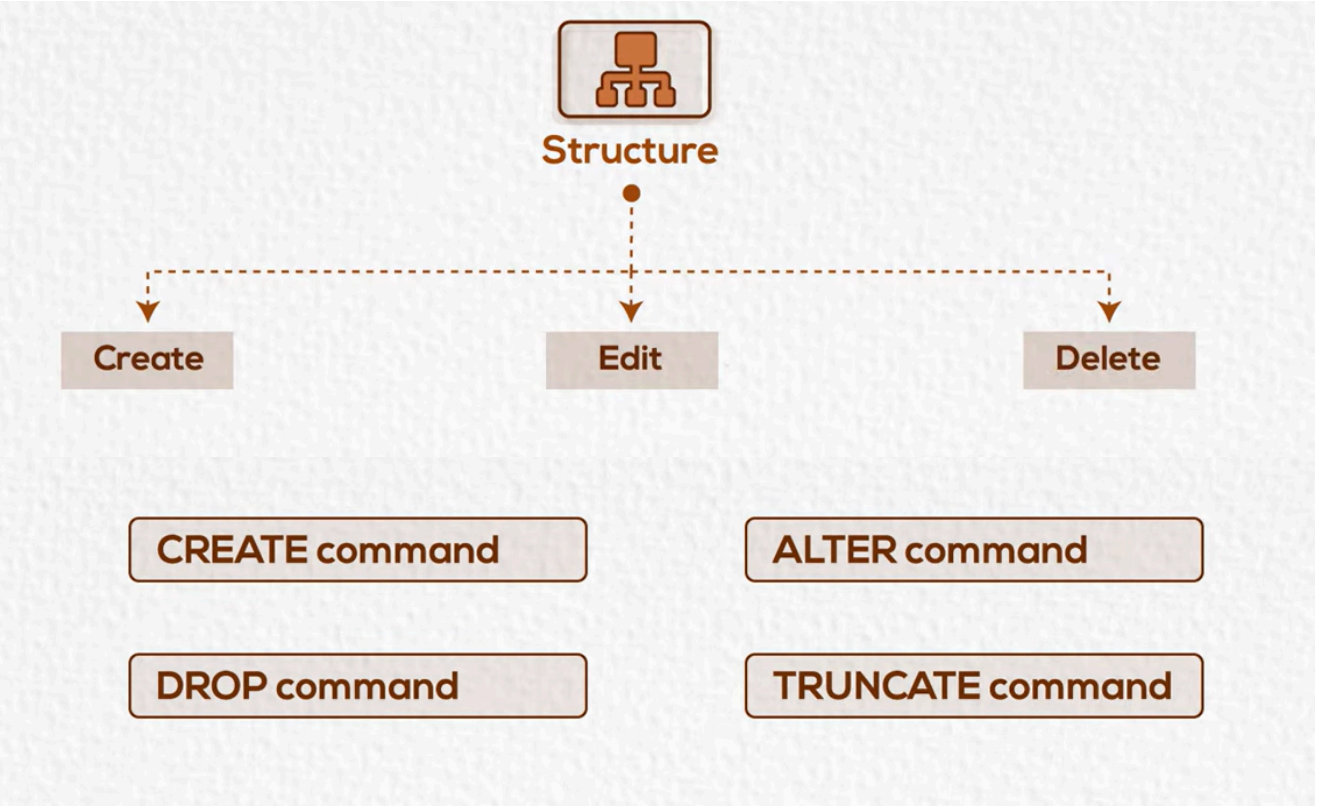
- Restrictions on Database table or object to help **Maintain integrity of data**

### Constraints Types

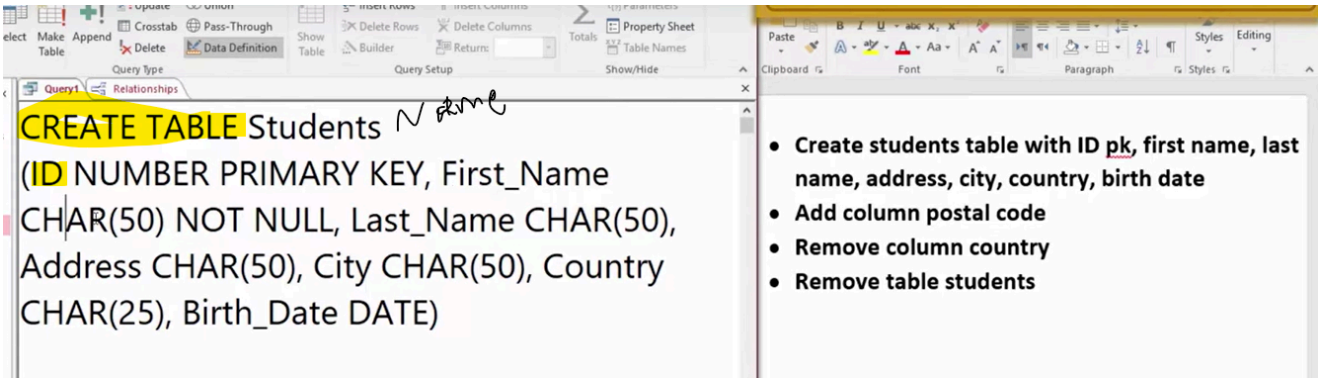





## SQL - Data Definition Language(DDL)

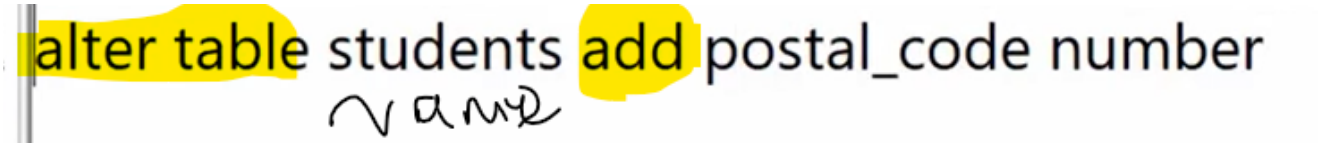



### Create

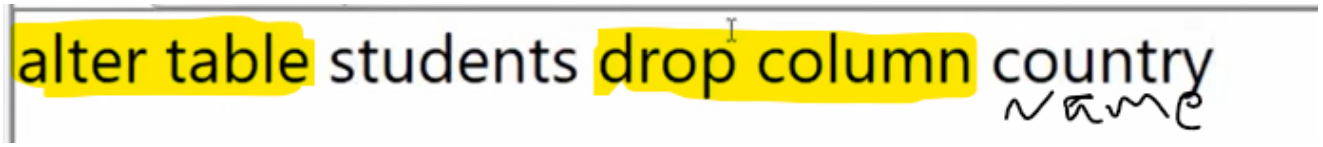



## Alter

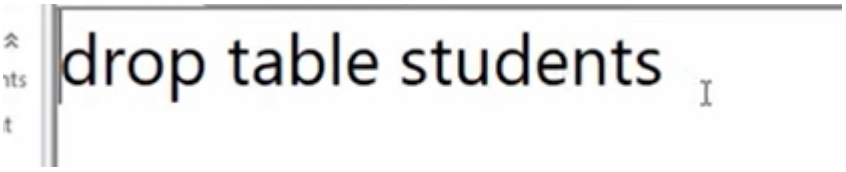
*ADD*



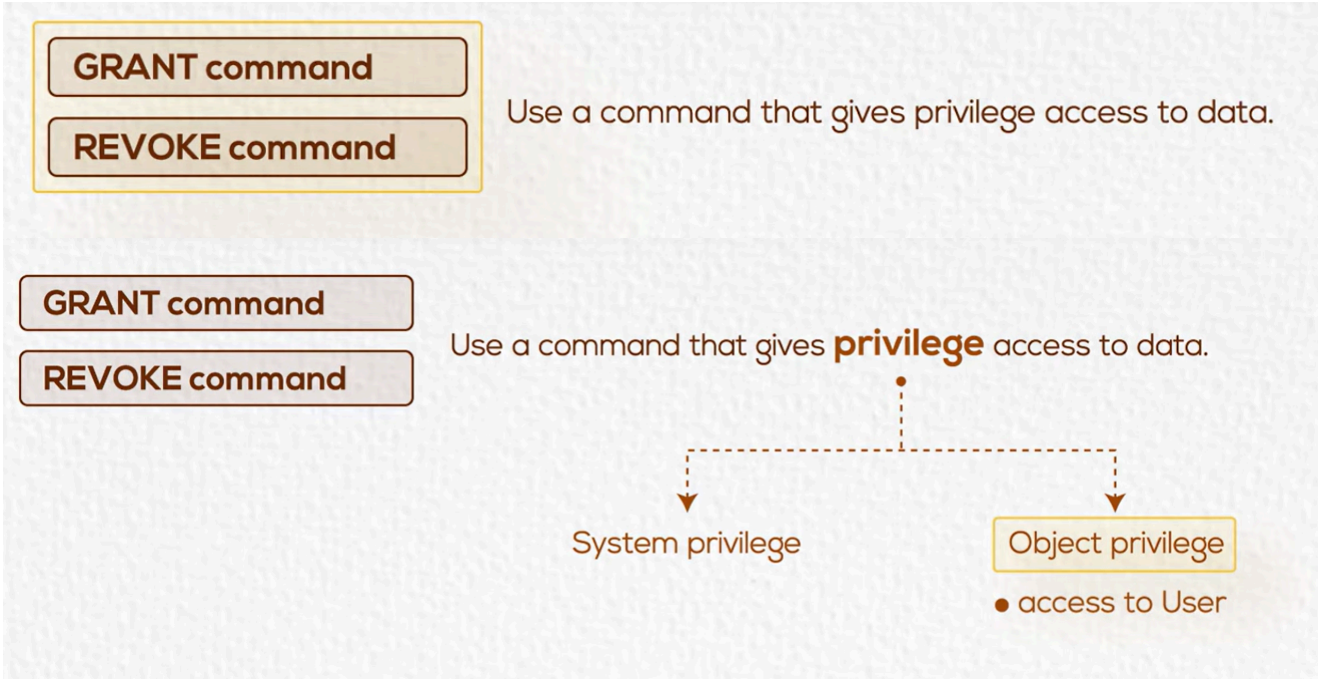

*REMOVE*

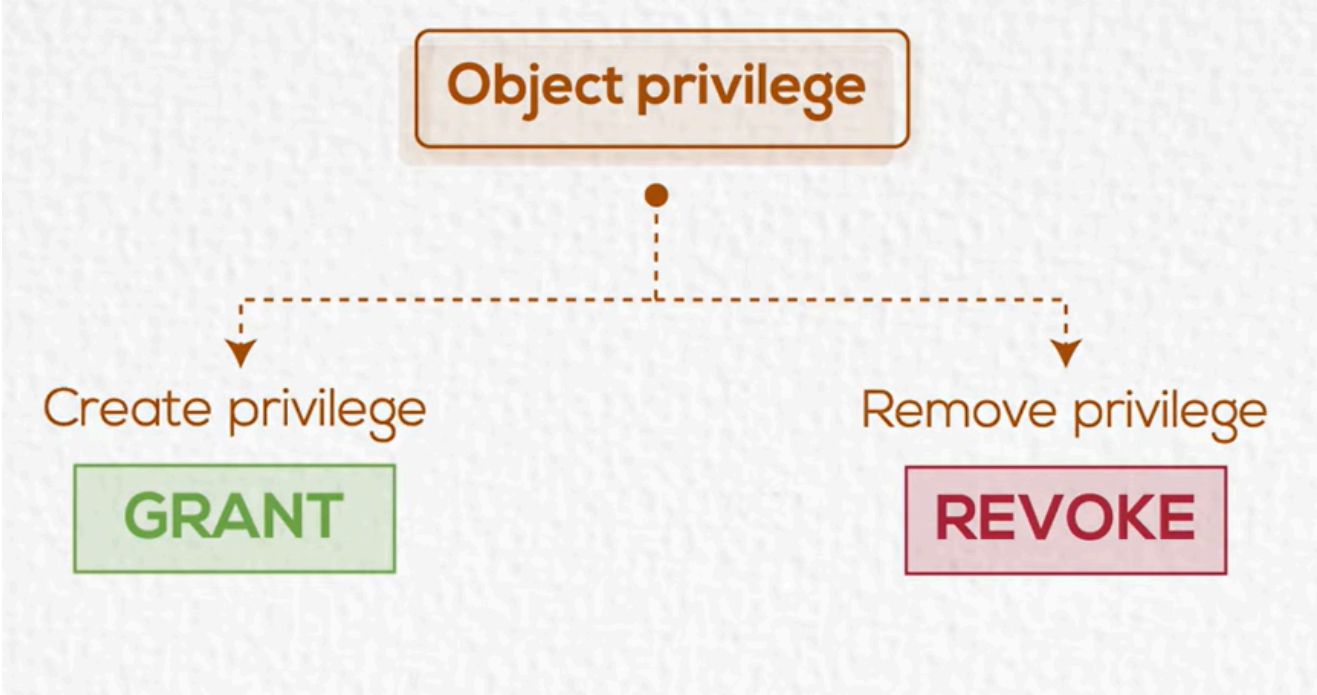



## Drop

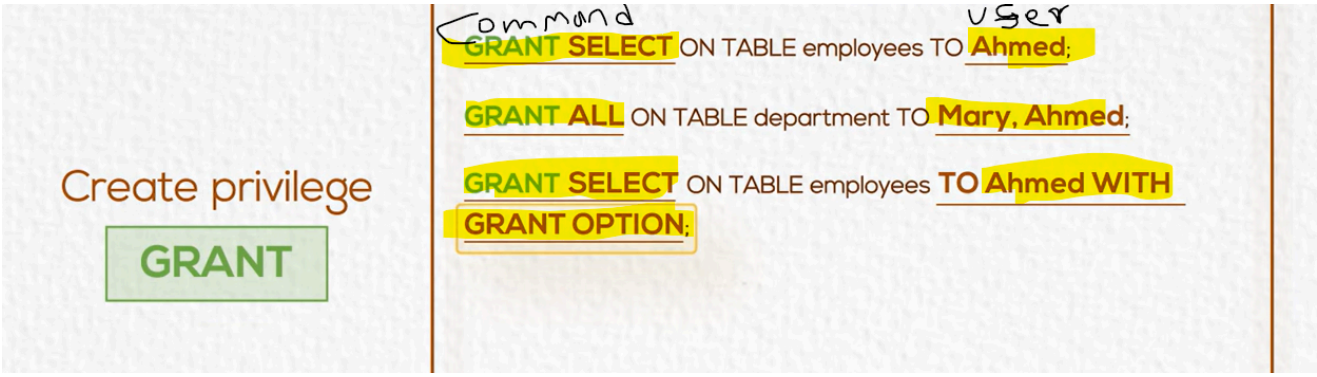



## SQL - Data Control Language (GRANT, REVOKE)






### Grand




### Revoke


## Chapter 5 - Data Manipulation Language (DML)

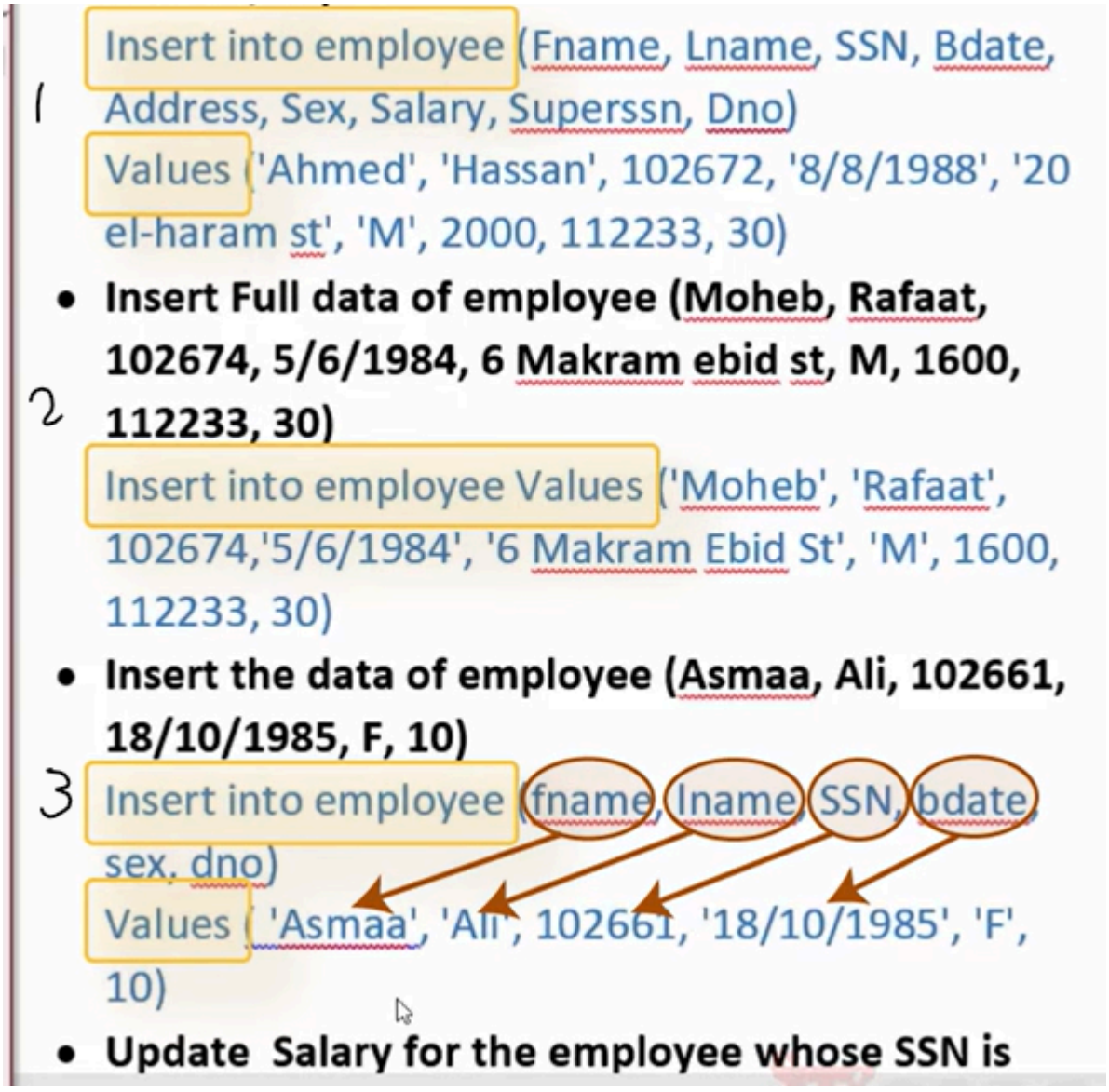
INSERT Command

UPDATE Command

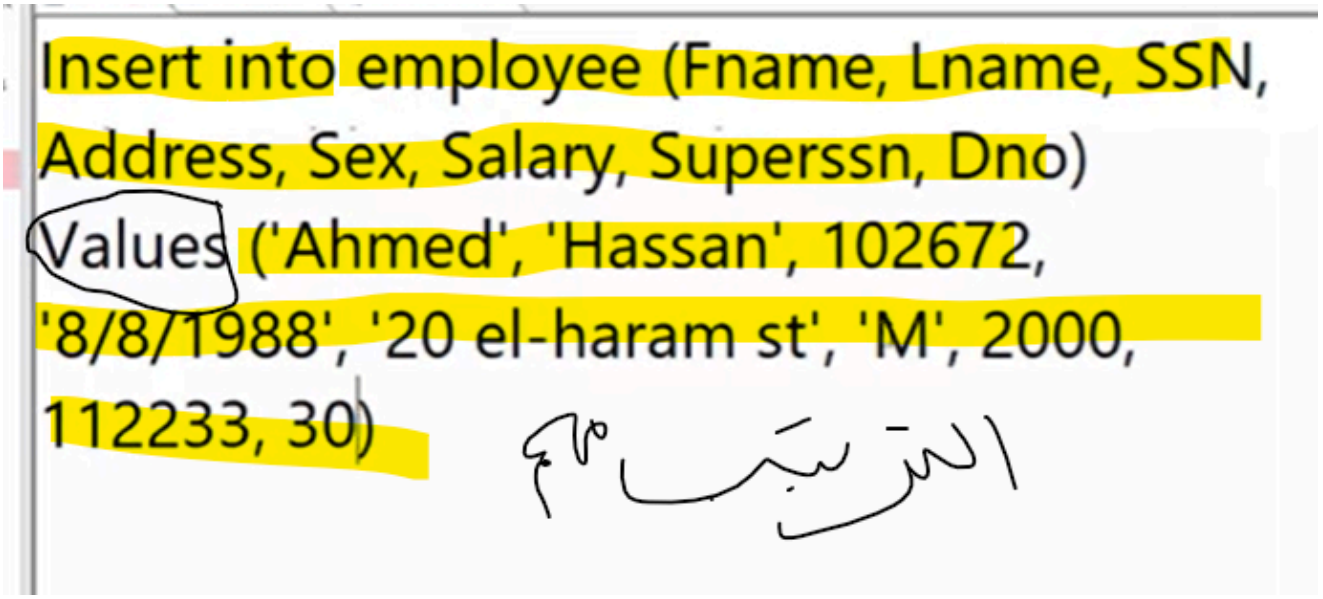
DELETE Command

SELECT Command

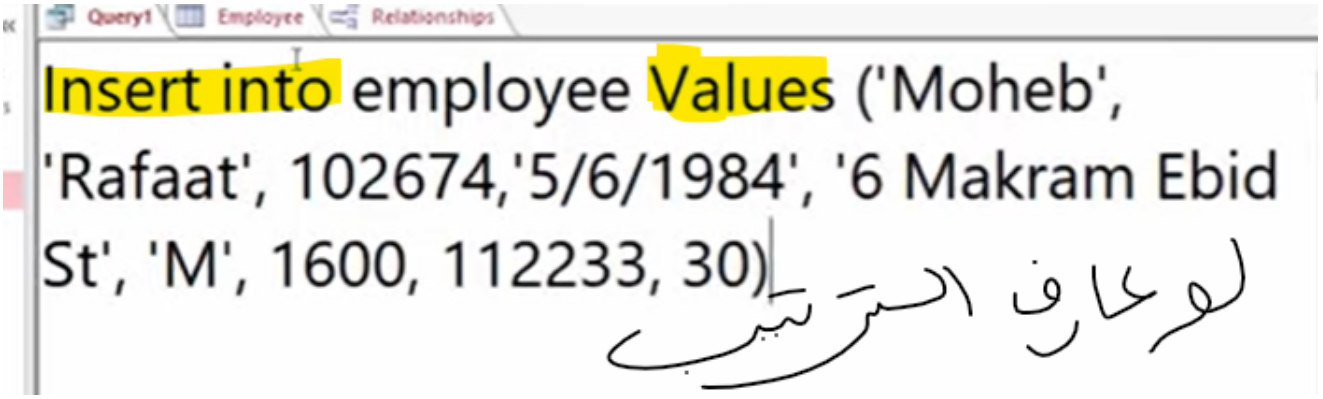
## Insert



*First Way*

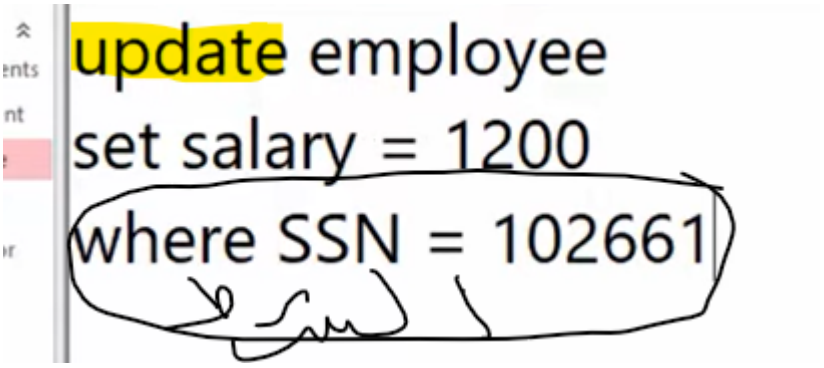



*Second Way*

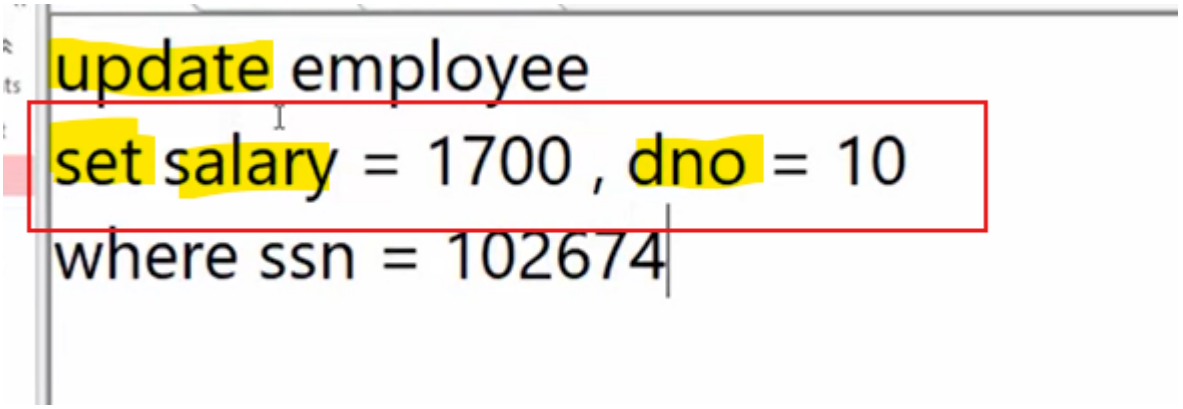



## Update

*for one Column*

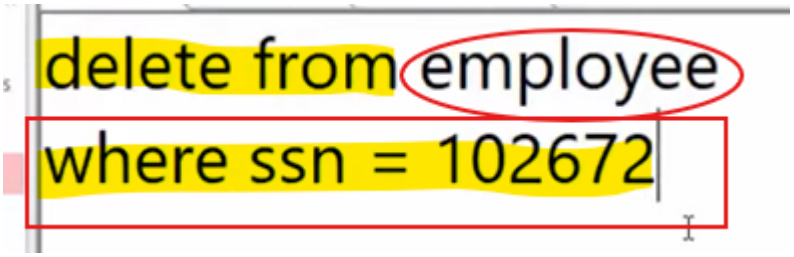


*more than one Column*

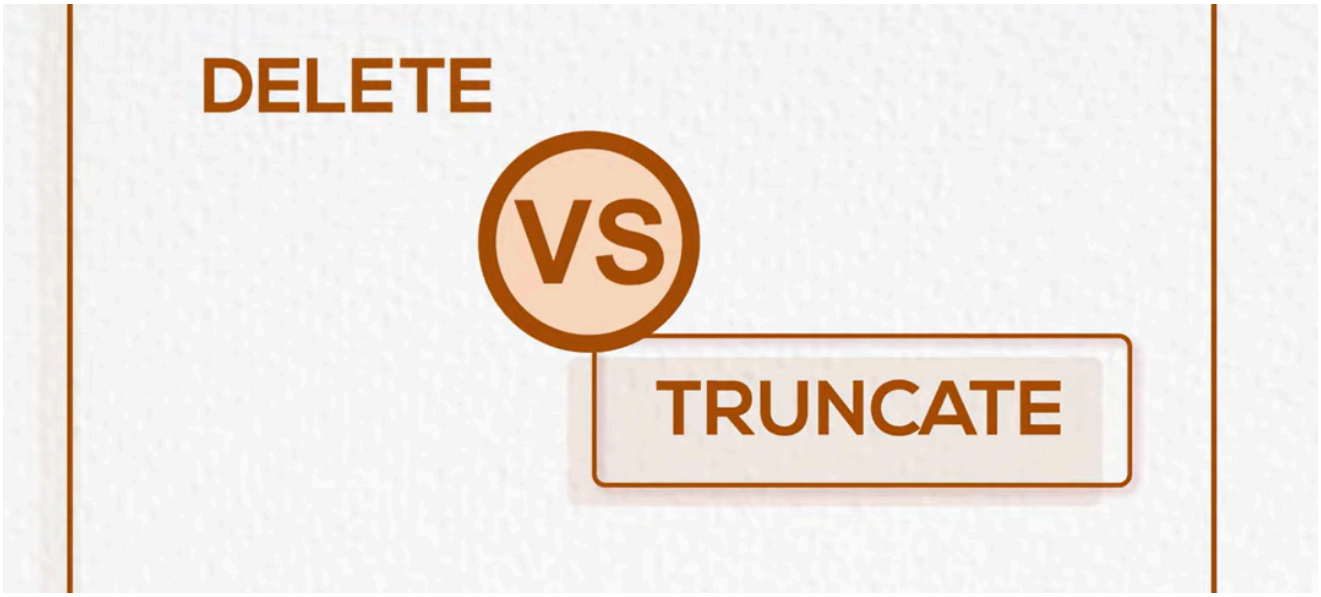


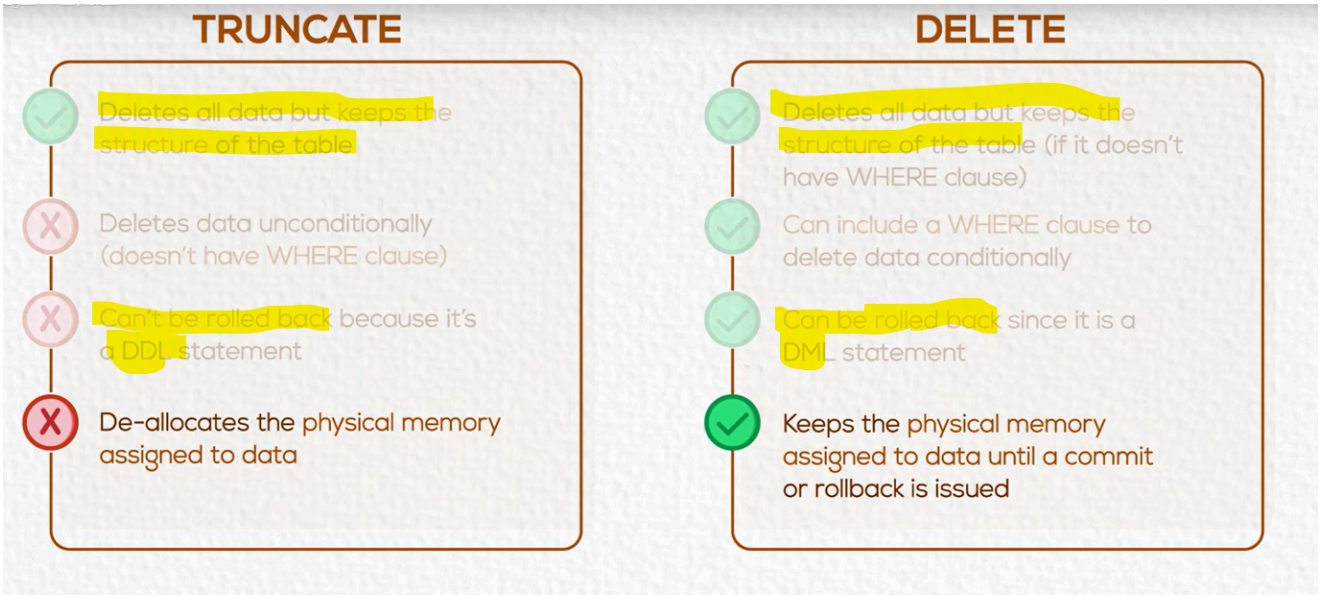

## Delete

*Record Level*




## Difference between Delete & Truncate






## Select

```
Select Dname, Dnum, MGRSSN, [MGRStart
date]
from departments
```

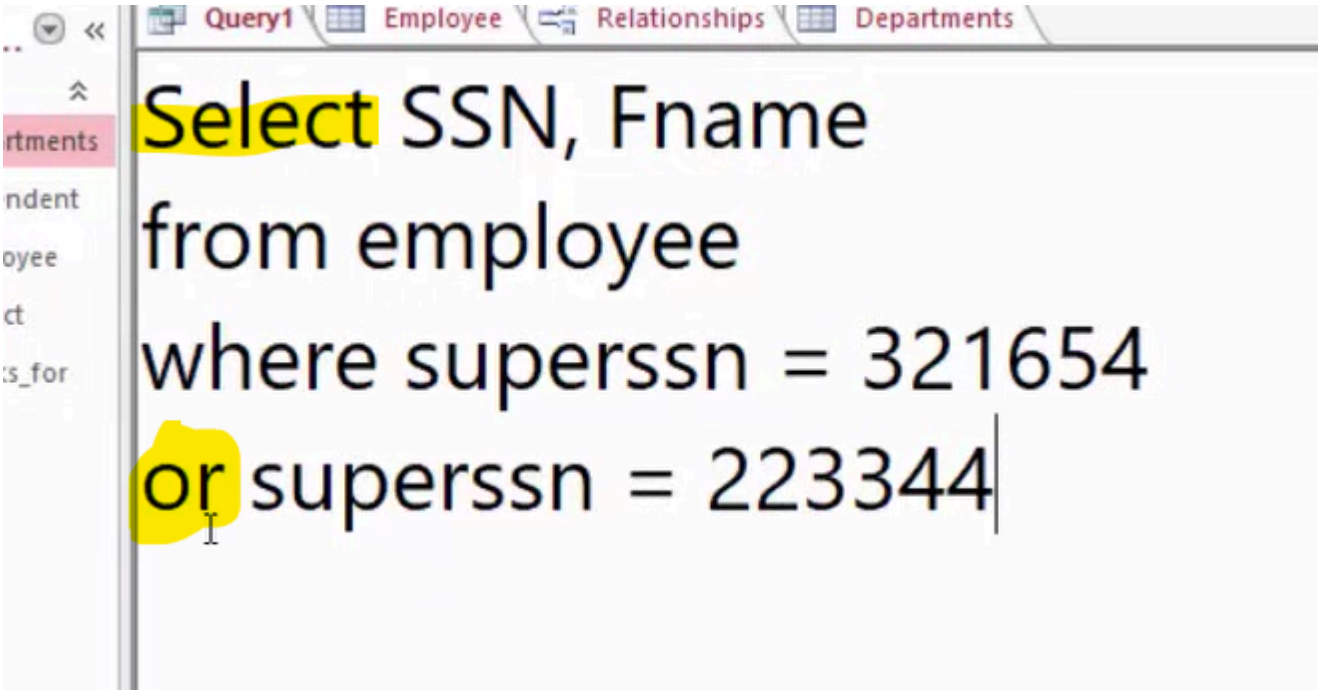
```
select *
from departments
where mgrssn = 223344
```

## Comparison & logical Operators

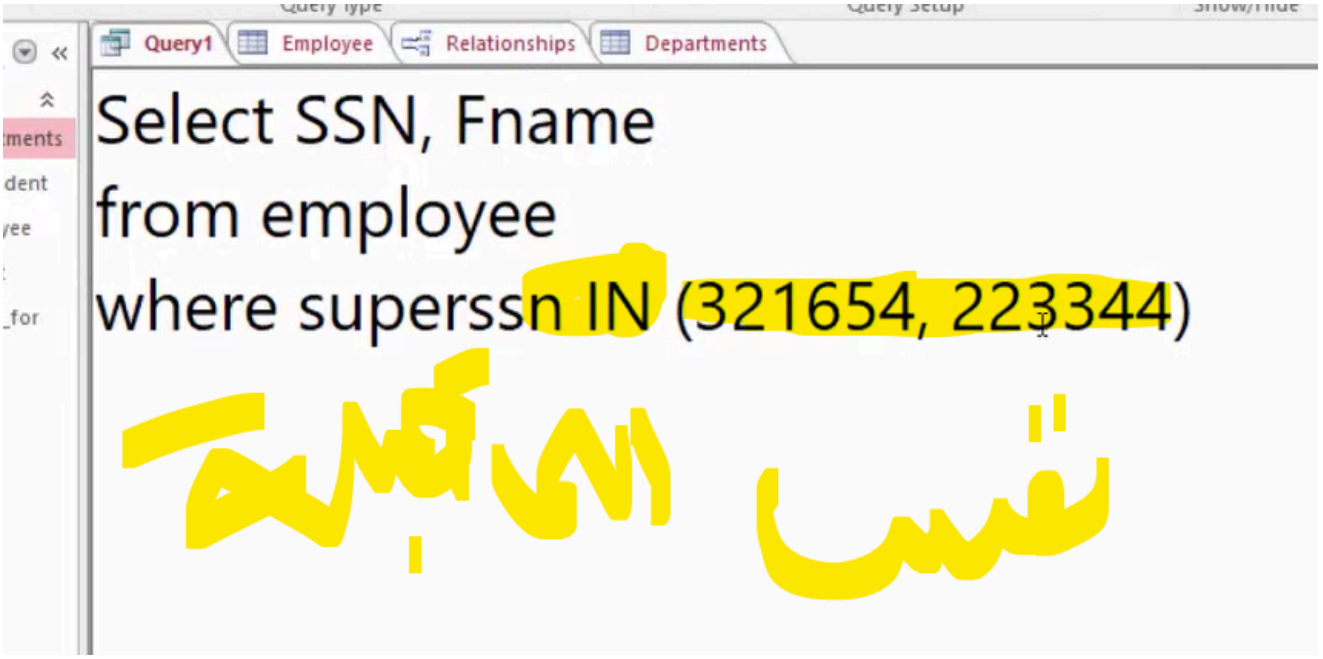
```
select *
from employee
where salary > 1500
```

```
select fname
from employee
where salary >= 1500
and salary <=2500
```

```
select fname
from employee
where salary between 1500 and 2500
```

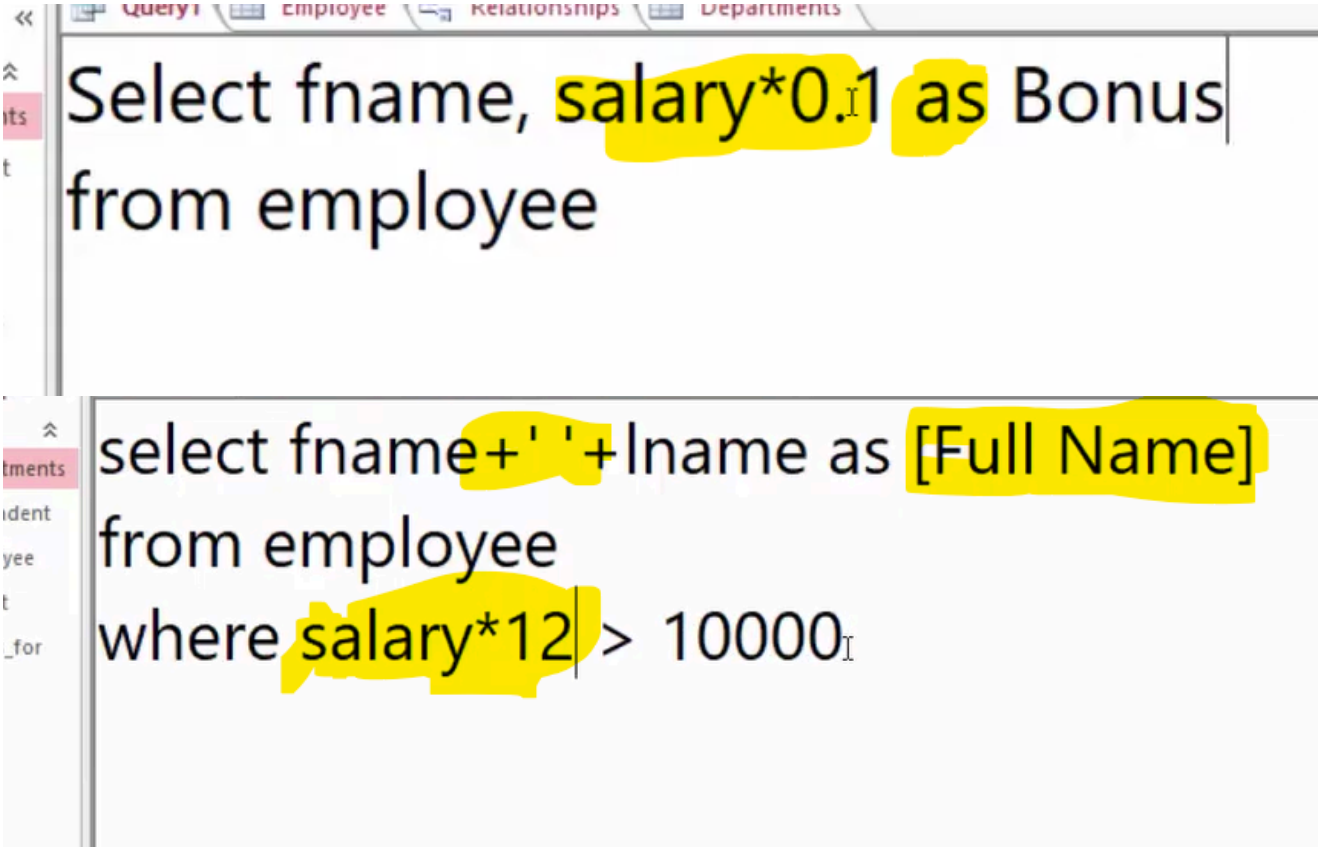



*Multi-Row Operator*




## Like Operator

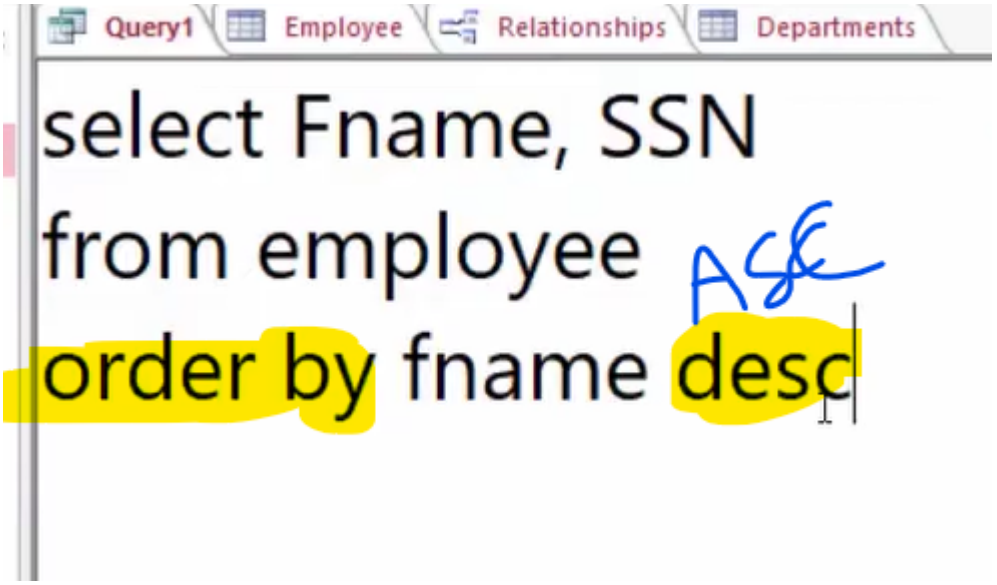

## Alias



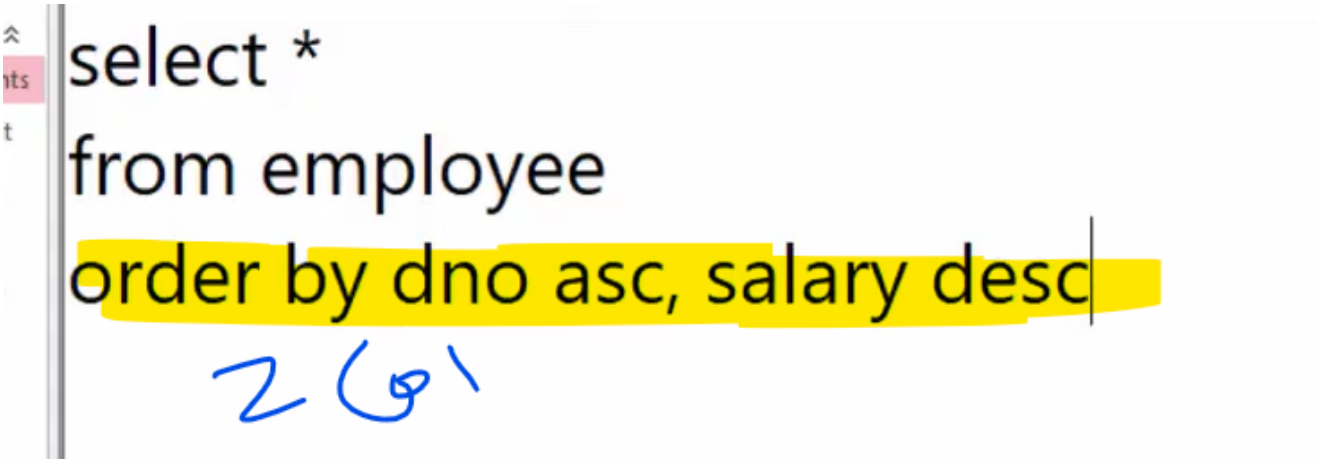




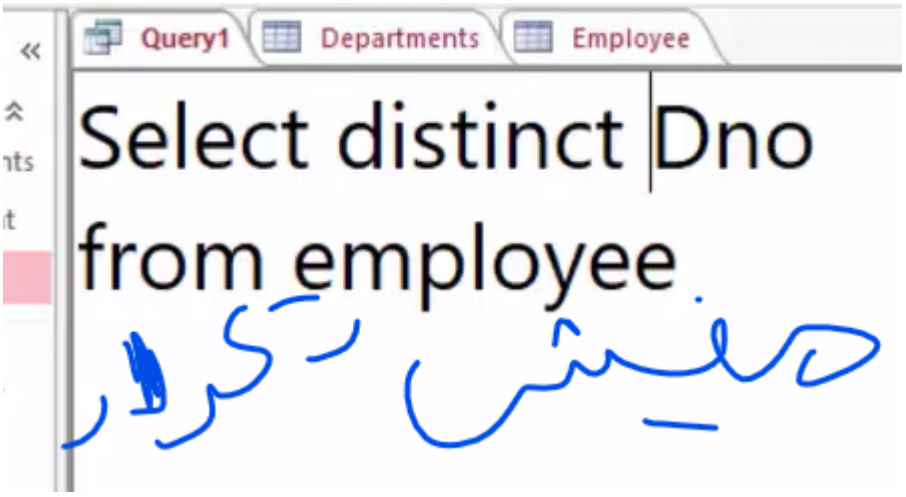
## Order by




*هيرتب الأول تصاعدي وبعد كدا تنازلي علي حسب كل عمود*




## Distinct




## Inner Join

*لو عاوز تعرض الداتا من اكتر من جدول عندك عندك معلومة من هنا ومعلومة هناك ودايما بكون علاقة بين ال Primary key , foreign key*
*واحنا هنا لازم يكون لكل قمية علي الشمال قمية موجوده علي اليمين*

equal join

```
select fname, dname
from employee, departments
where MGRSSN = SSN
```

اسماء الجداول اللي عندك

Join Condition

```
select fname, dname
from employee e, departments as d
where e.dno = d.dno
```

Alias

استخدمته
تحت عشان اعرف افرق كل واحد
جاي من انهي جدول

Inner join

```
select fname, dname
from employee e inner join departments as
d
on e.dno = d.dno
```

```
select fname, pname, hours
from employee, project, works_for
where ssn = essn
and pno = pnumber
```

3 Tables

2 Join

## Outer , Full Join

لو عاوز في قيمة موجوده علي اليمين او الشمال و في الناحية التانية مفيش

## Types of Outer

### Left Outer Join

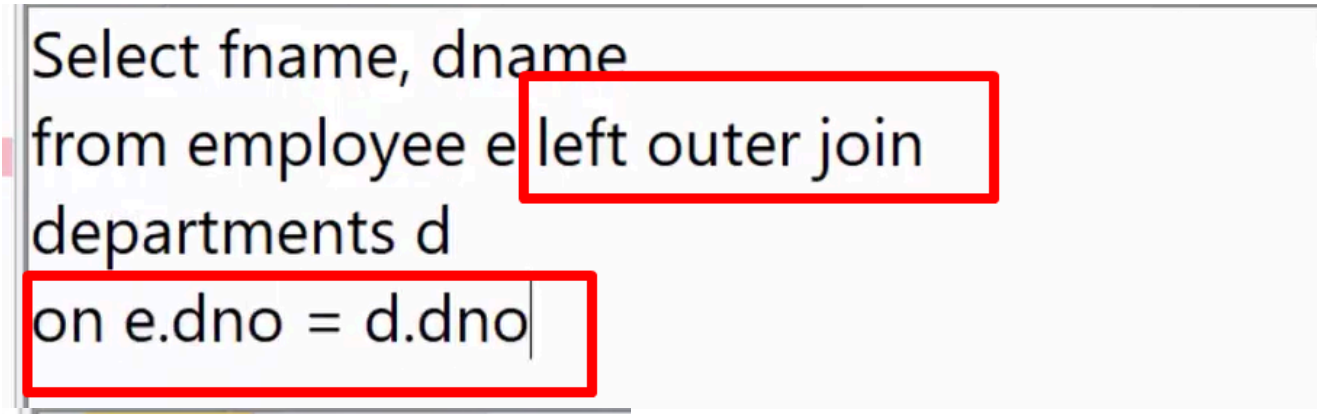
هنا هيرجع كل القيم اللي في الجدول الشمال

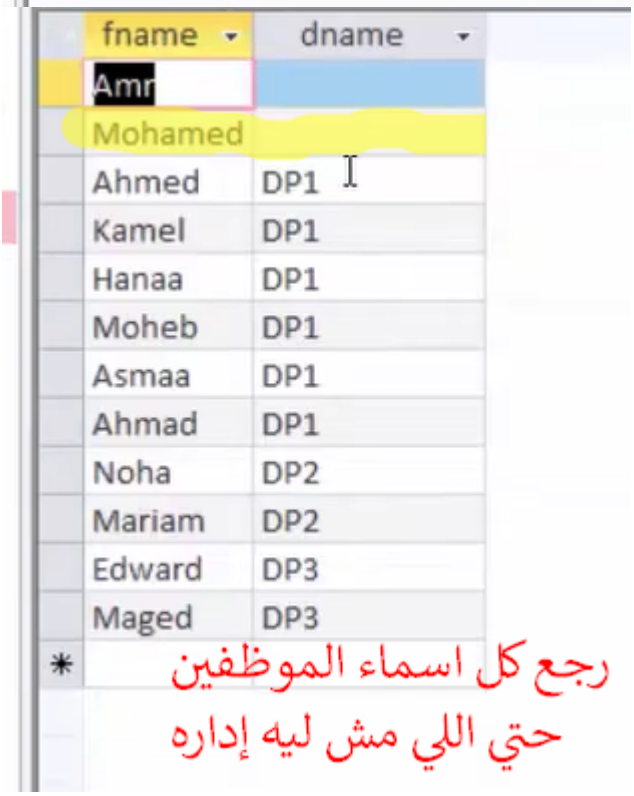
### Right Outer Join

هنا هيرجع كل القيم اللي في الجدول اليمين

### Full Outer Join

بيرجع كله اللي هنا واللي هنان حتي لو مش ليه قمية




## Self Join

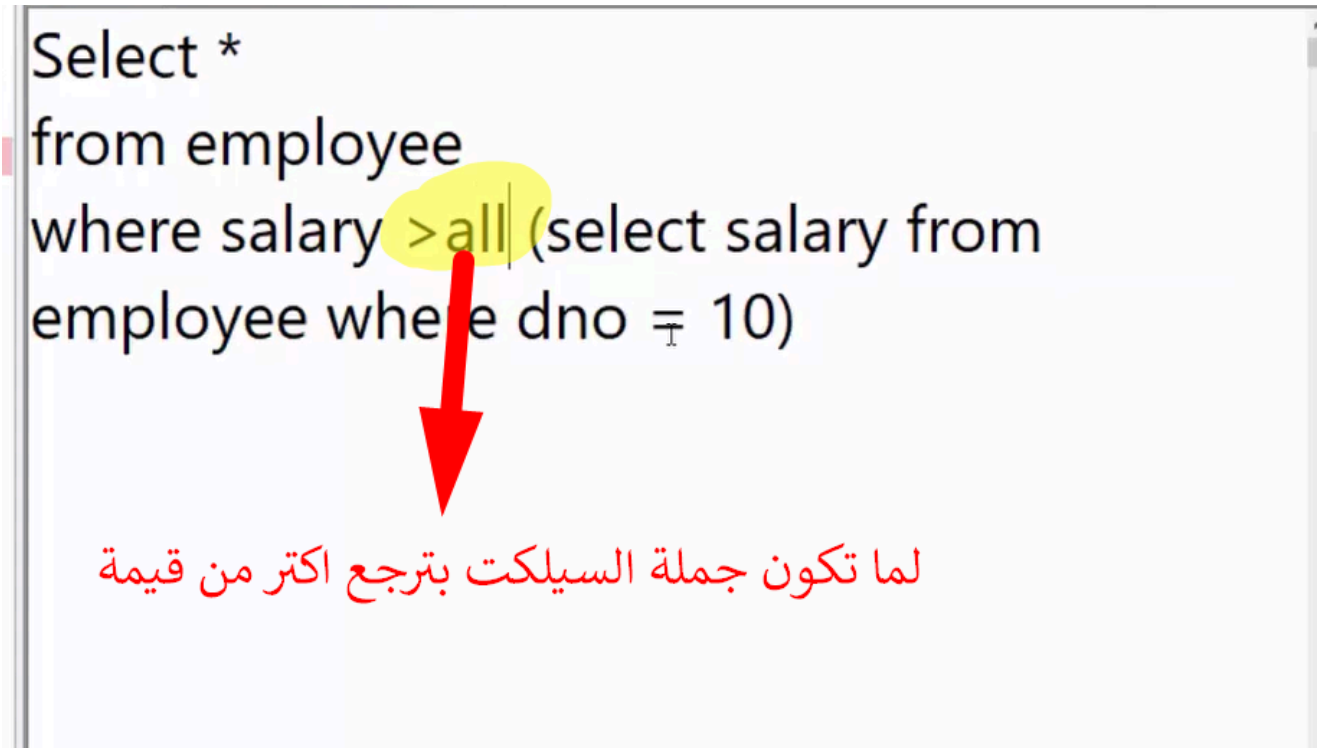
بنعمل جوين للجدول بنفسه ودي بتكون في ال Recursive relationship

```
select e.fname, s.fname
from employee e, employee s
where e.superssn = s.ssn
```

## Sub-Queries

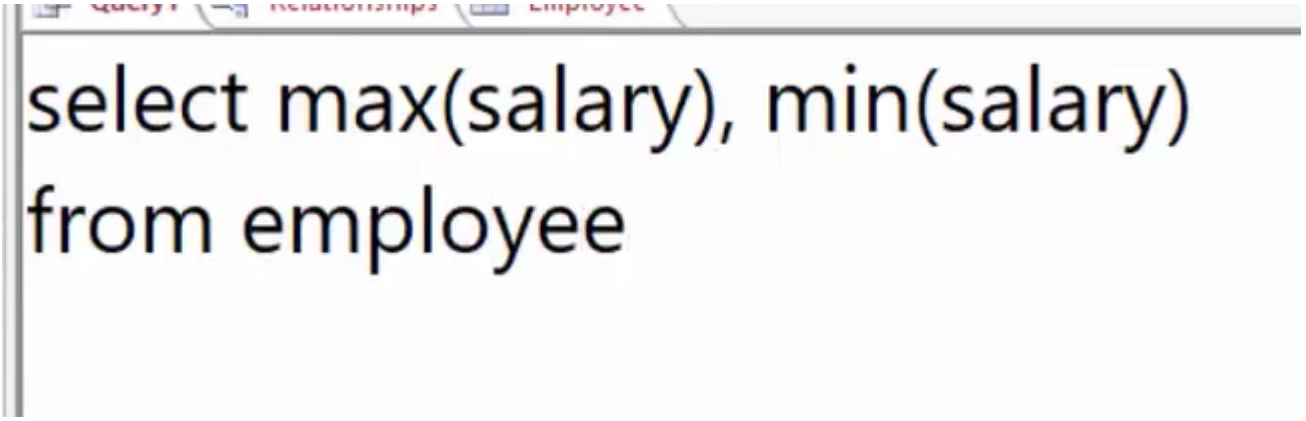
```
select *
from employee
where salary > (select salary from employee
where fname ='Ahmed' and lname = 'Ali')
```

لو انا عندي معلومة مش موجوده
عندي مباشرة فهروح اعمل ليها سيلكت

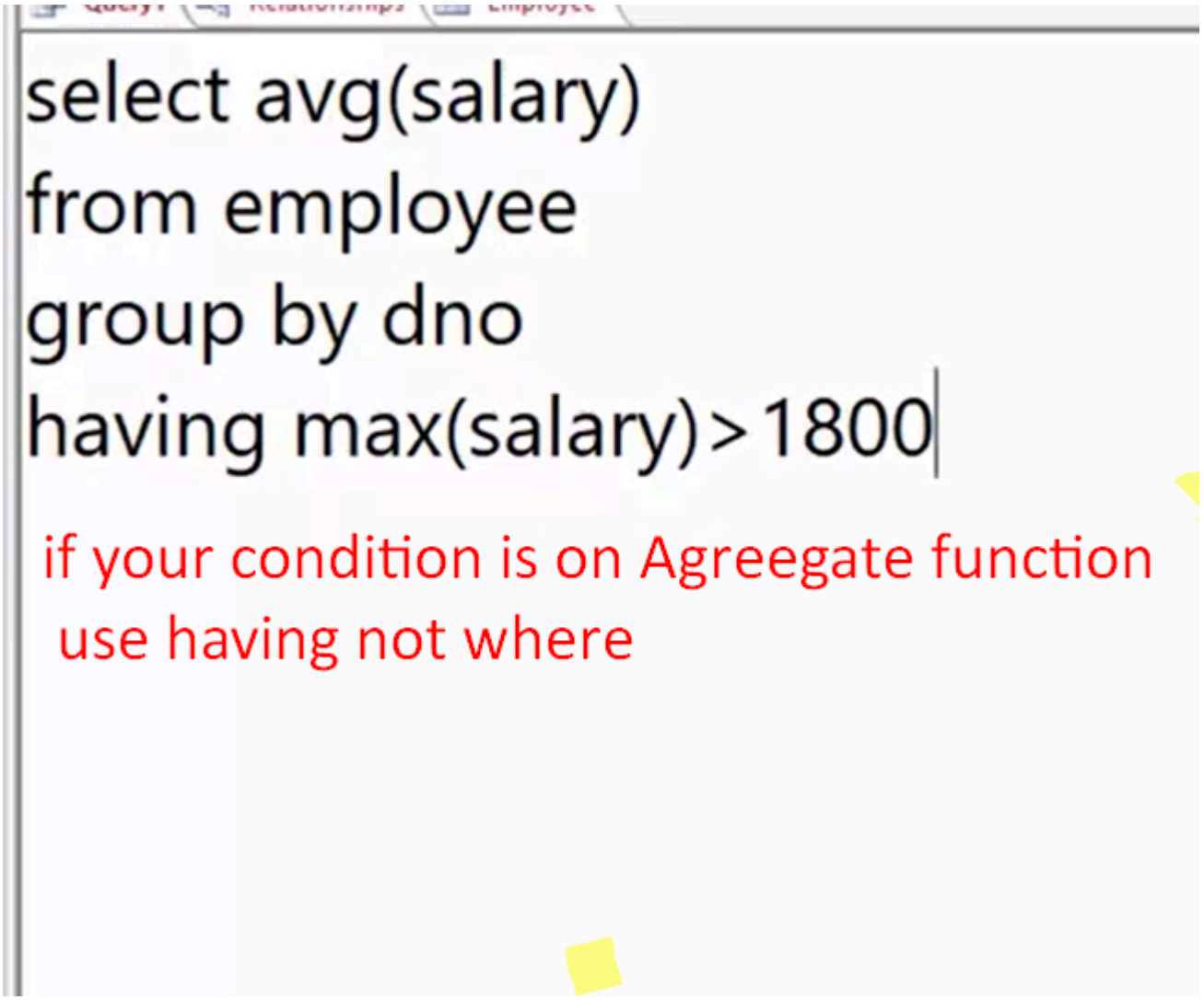



## Max , Min , Count Functions

*بتتجاهل ال null*




## Group by & Having




### Select Conclusion

## Chapter 6 - SQL Other DB Objects

### Views (Create , Modify , Remove, Types)

- A view is a **logical table** based on a **table** or **another view**.
- A view contains no data of its own, but is like a **window through** which **data from tables** can be viewed or changed.
- The tables on which a view is based are **called base tables**.

- The **view** is stored as a **SELECT statement** in the data dictionary.

- **CREATE VIEW** view [ (column 1 [ , column2 ] ... ) ]

  **AS**

  **SELECT**

- Create a view to display **employee names** and total **hours employee worked** on a project

بنتعامل معاه بعد كدا أكنه جدول عاادي

- **CREATE VIEW** vw_work_hrs

  **AS**

  **SELECT** Fname, Lname, Pname, Hours

  **FROM** Employee, Project, Works_on

  **WHERE** SSN=ESSN **AND** PNO=PNUMBER

## Create Views with Check option

```
CREATE VIEW Suppliers
AS
SELECT *
FROM suppliers
WHERE status > 15
WITH CHECK OPTION;
```

## Modifying a View

- Syntax

  **CREATE OR REPLACE VIEW** view_name

- Example

  **CREATE OR REPLACE VIEW** vw_work_hrs
  AS
  SELECT Fname , Lname , Pname , Hours
  FROM Employee, Project , Works_on
  WHERE SSN=ESSN AND PNO=PNUMBER AND **Dno = 5**;

## Removing a View

- Syntax

  **DROP VIEW** view_name;

## Advantages of Views

1- Restrict data access

2- Make complex queries easy

3- Provide data independence.

4- Present different views of the same data


| Feature | Simple Views | Complex Views |
|---|---|---|
| Number of tables | One | One or more |
| Contain functions | No | Yes |
| Contain groups of data | No | Yes |
| DML operations through a view | Yes | Not always |

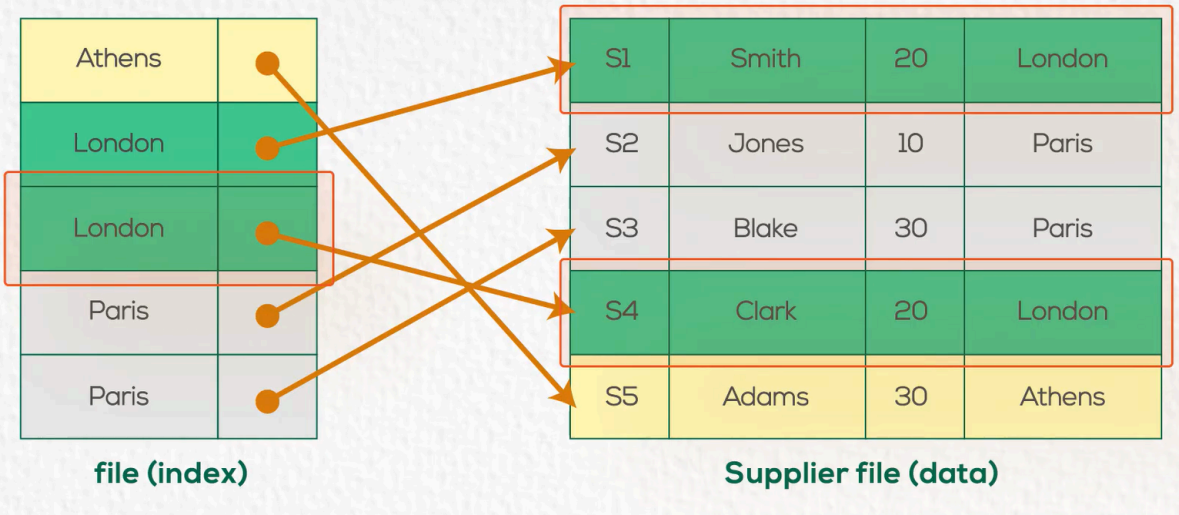
## Indexes

- Use indexes to solve the following problem

1. Not sorted
2. Scattered

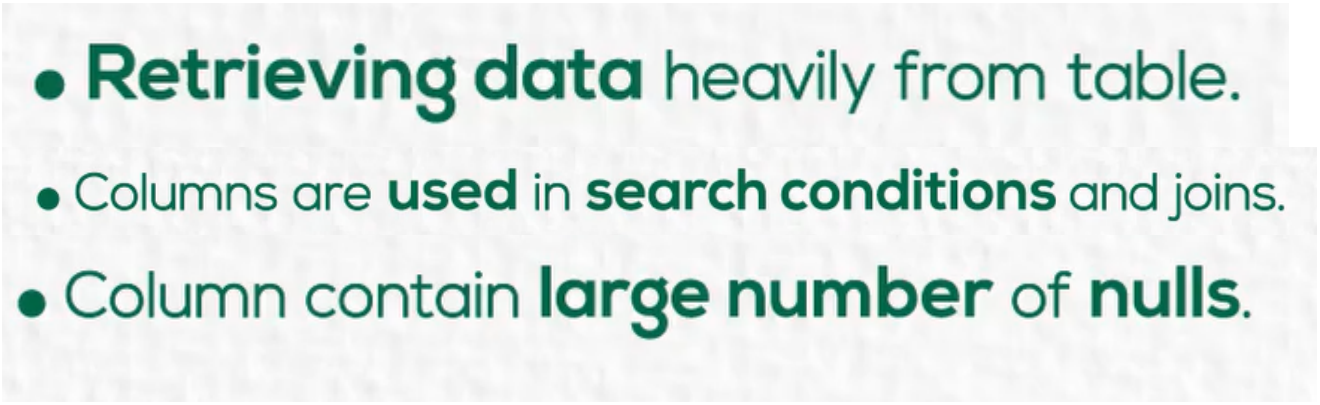
## Why indexes?

- They are used to **speed up the retrieval** of records in response to certain search conditions.
- May be **defined** on **multiple columns**.
- Can be created by the **user** or by the **DBMS**.
- Are used and **maintained by the DBMS**.



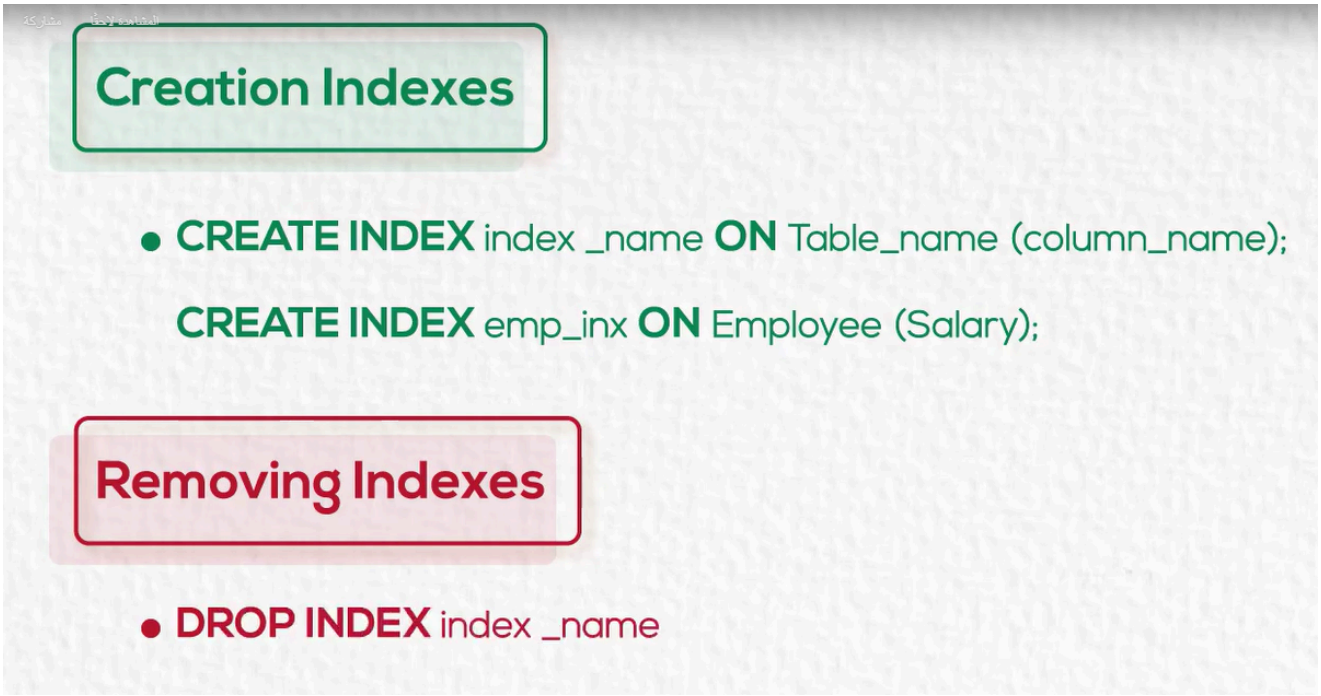

*ليه بيبطئ العمليات دي عشان هو لما بيجي وليكن عاوز يضيف فا بيضيف في اكتر من مكان مش في مكان واحد فا بياخد وقت وهكذا بقي*

### Guidelines





### Create index



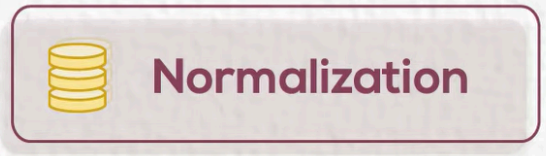

## Chapter 7 - Normalization

### What is Normalization

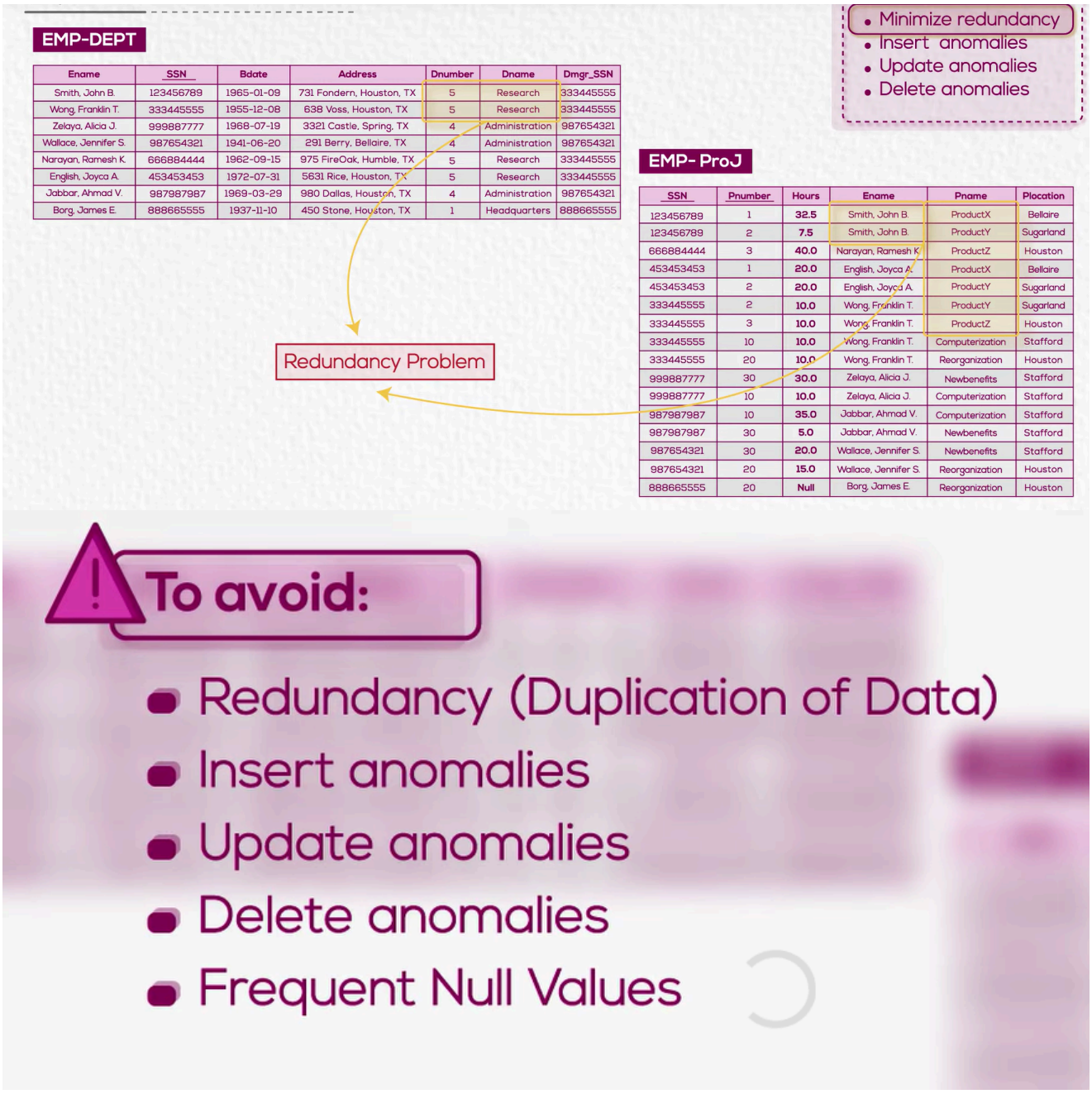
a process that takes a table through a series of tests (Normal Forms).

- Certify the goodness of a design and thus to minimize redundancy
  and insert anomalies
  update anomalies
  delete anomalies
- Use Normalization to another method for create a database design.

هي طريقة بنستخدمها عشان نقلل التكرار اللي في الداتا وكمان ممكن نعيد تصميم السيستم بتاعنا




- The process of decomposing unsatisfactory "bad" relations **by breaking up their attributes into smaller relations**.





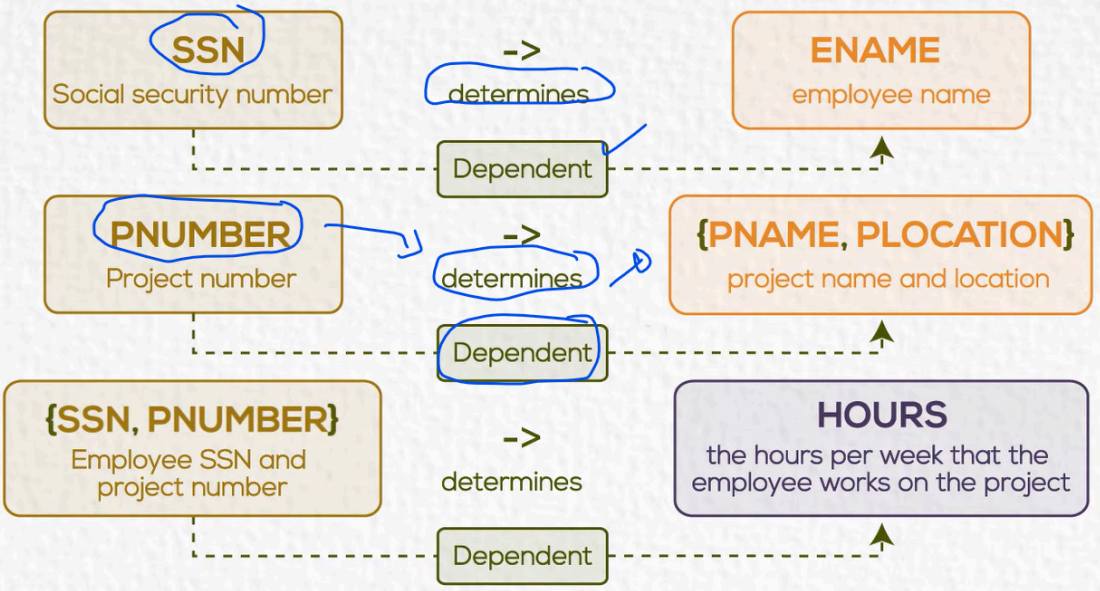


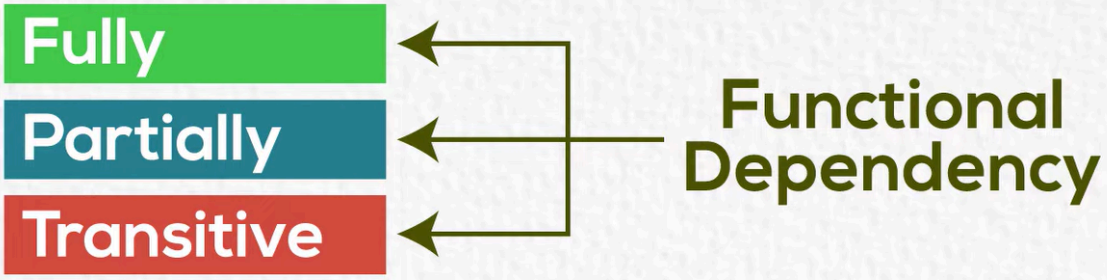
## Functional Dependency

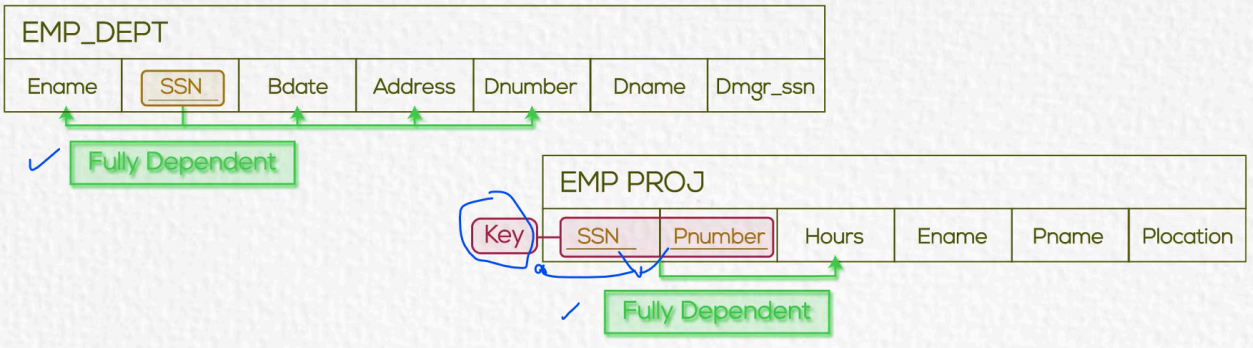
- A constraint between two attributes (columns) or two sets of columns.

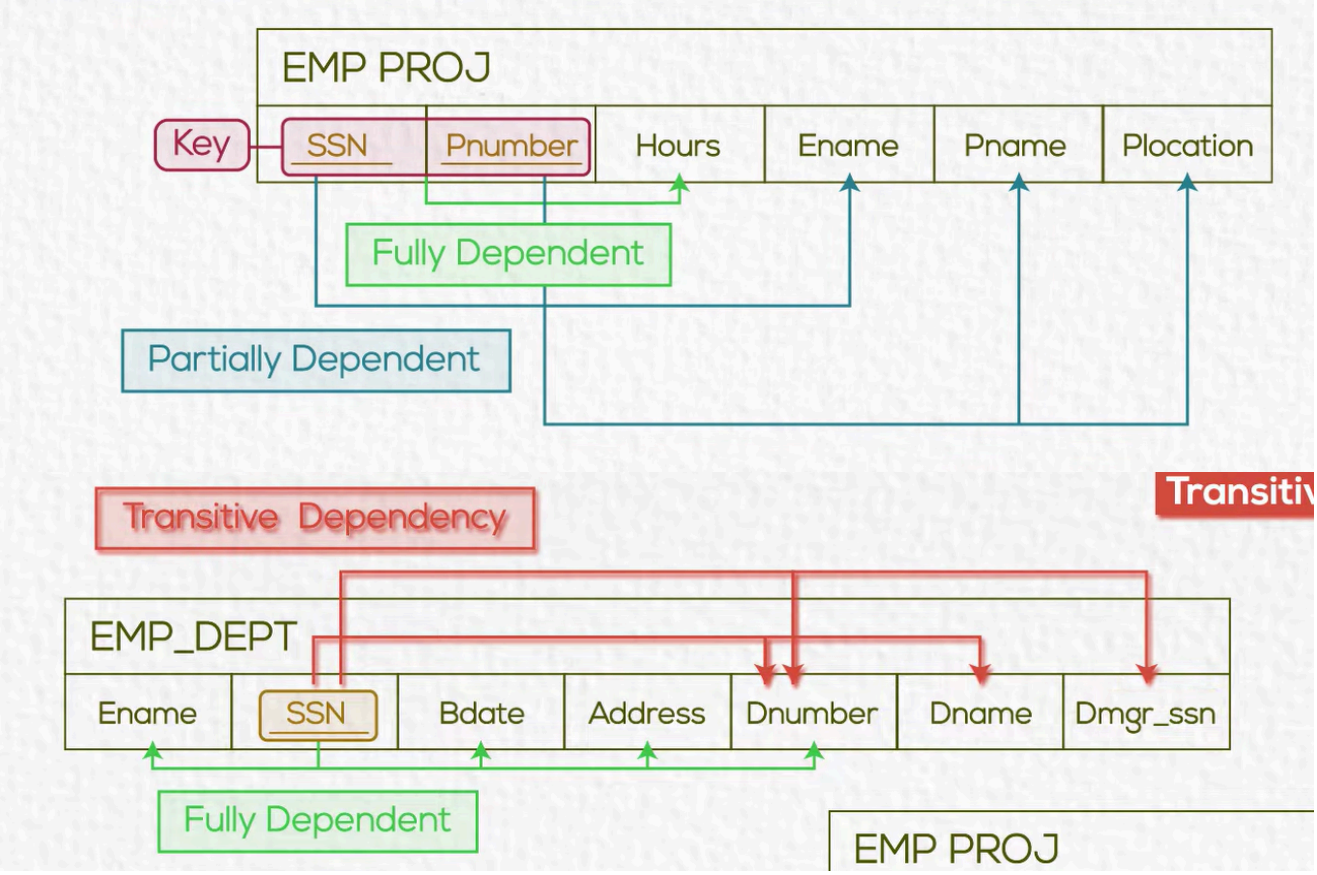
## Dependency

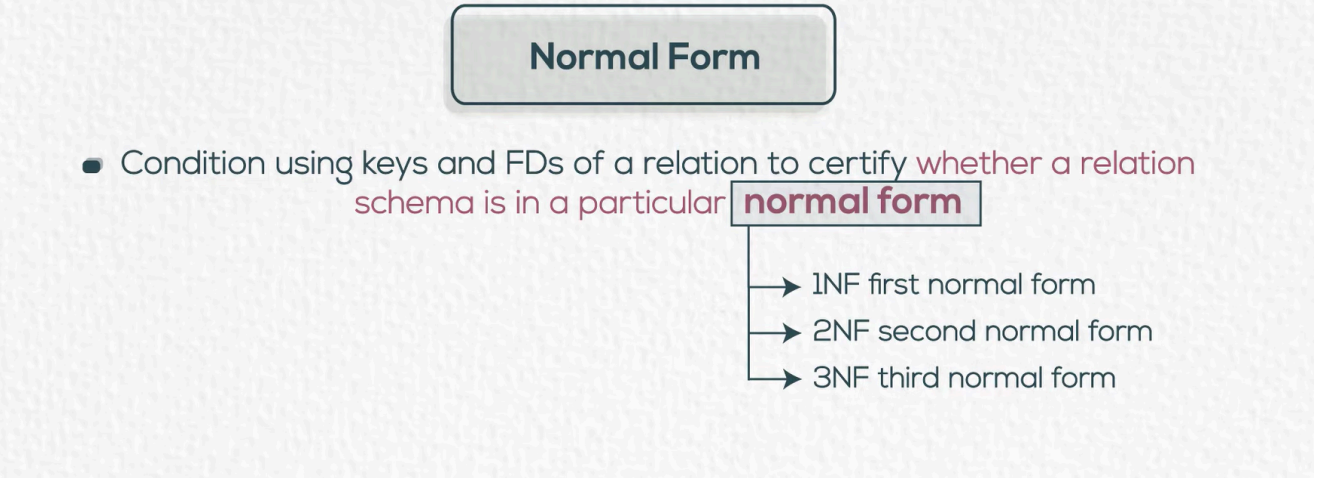
value of **A** uniquely **determines** the value of **B**



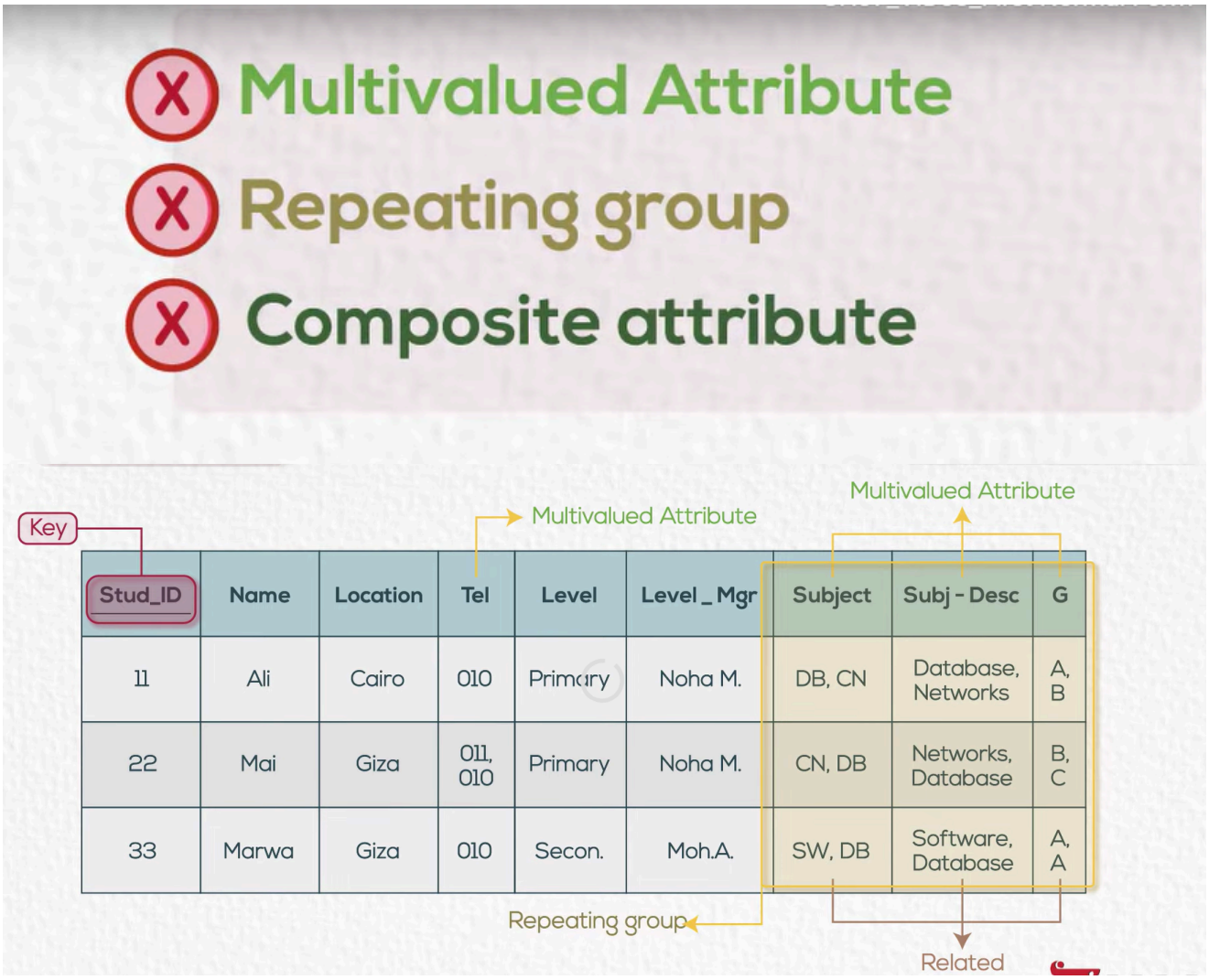



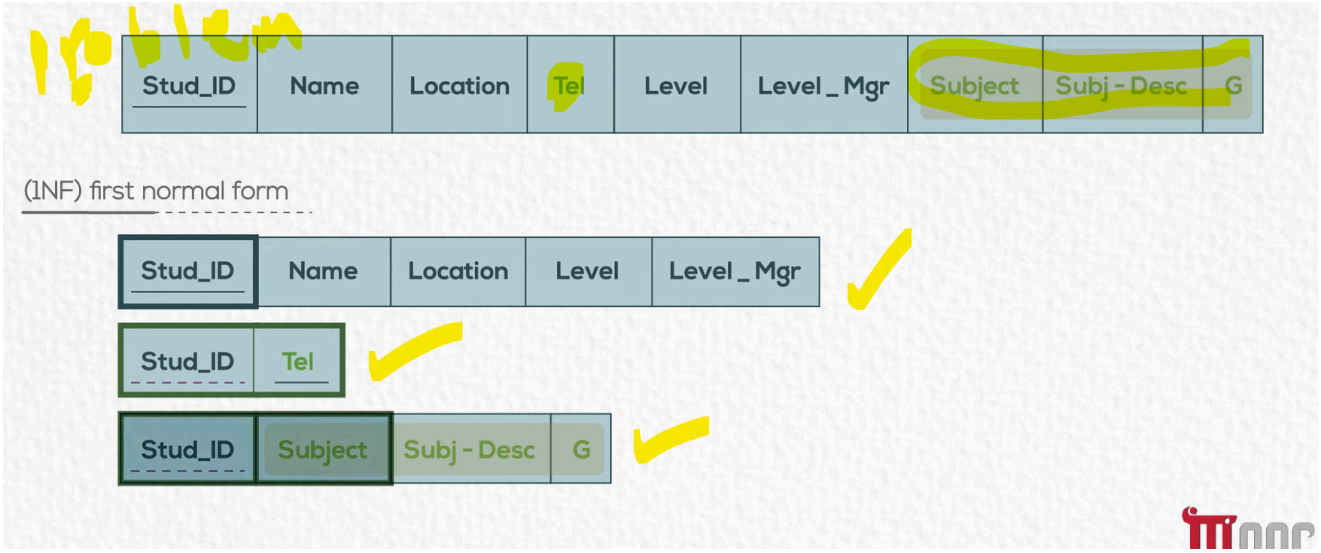





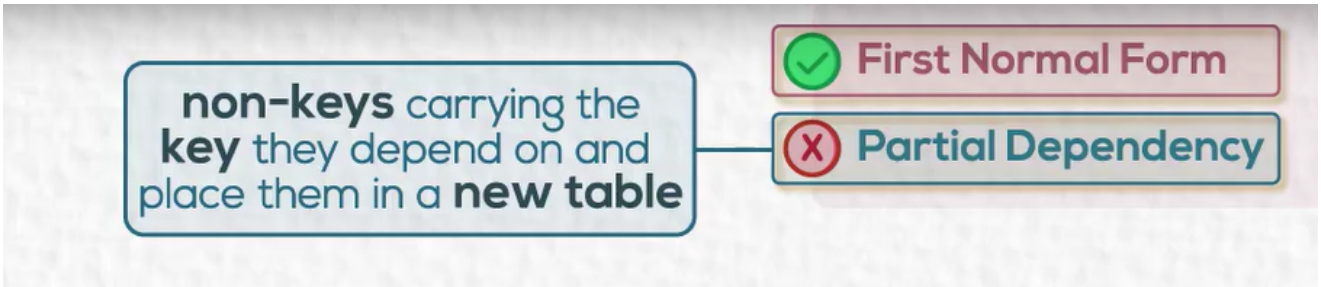



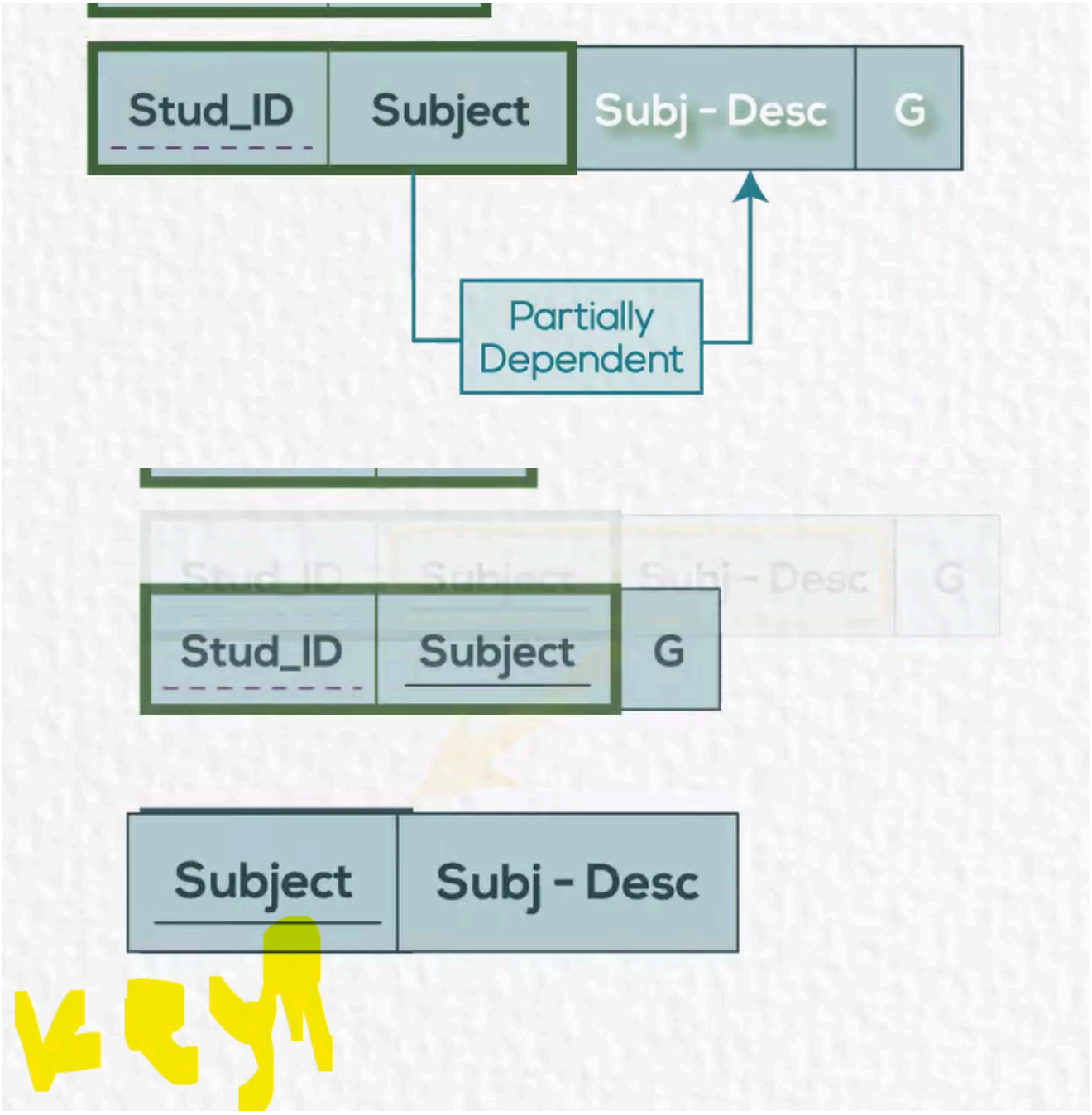

## First Normal Form

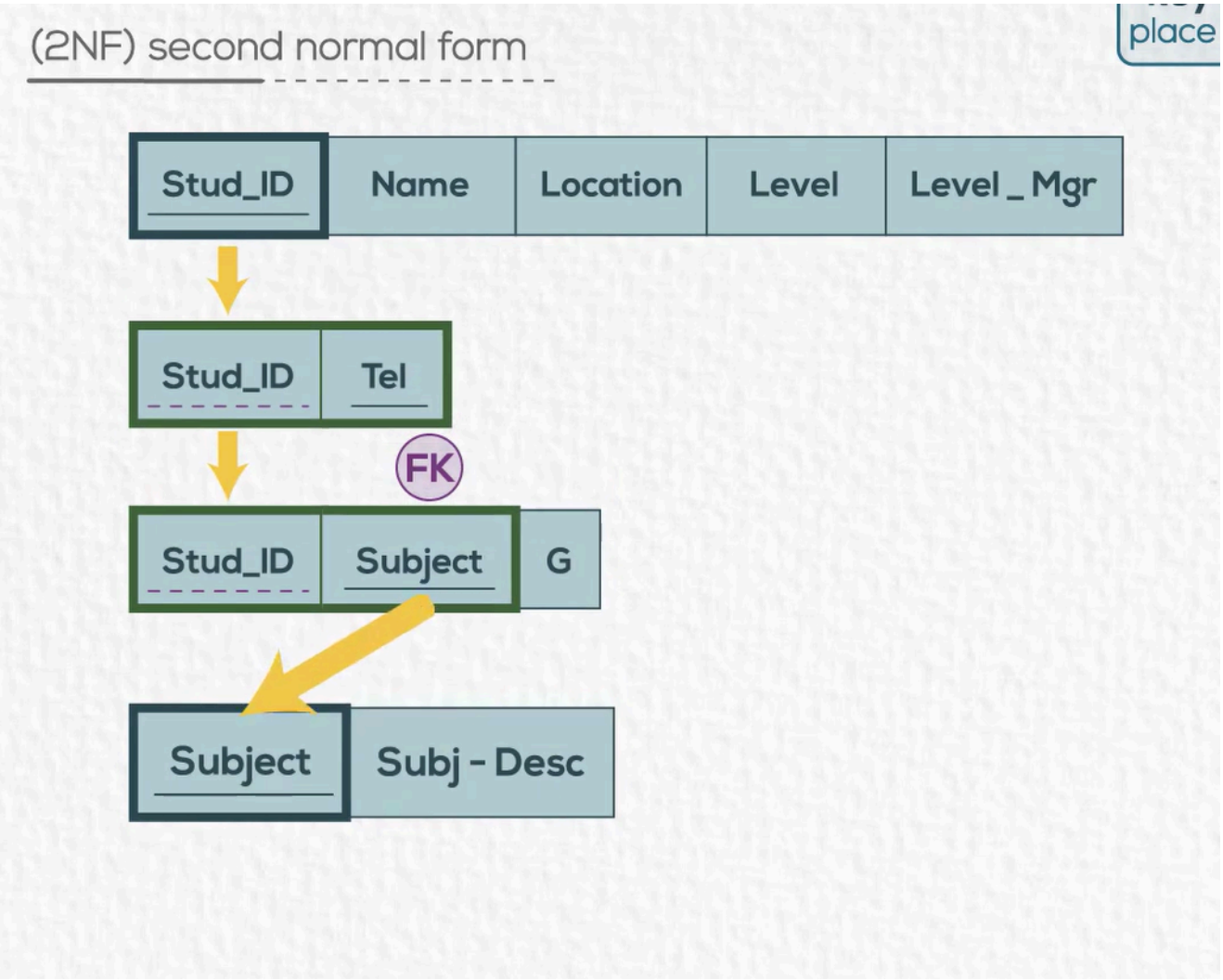


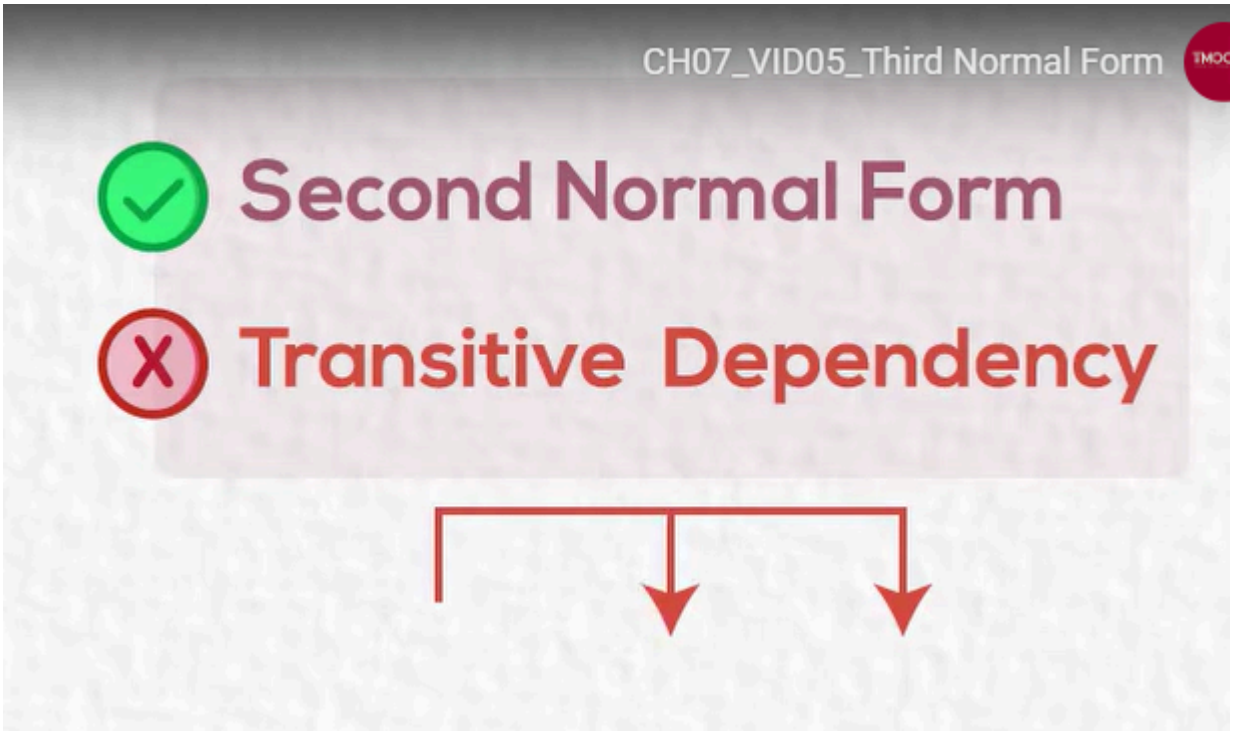

## Second Normal Form

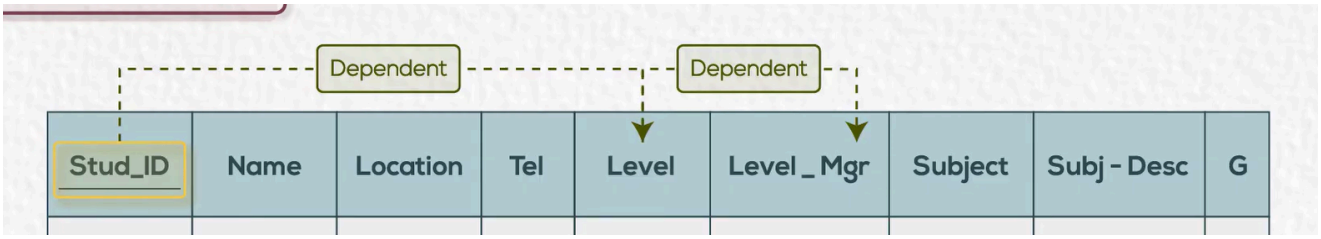


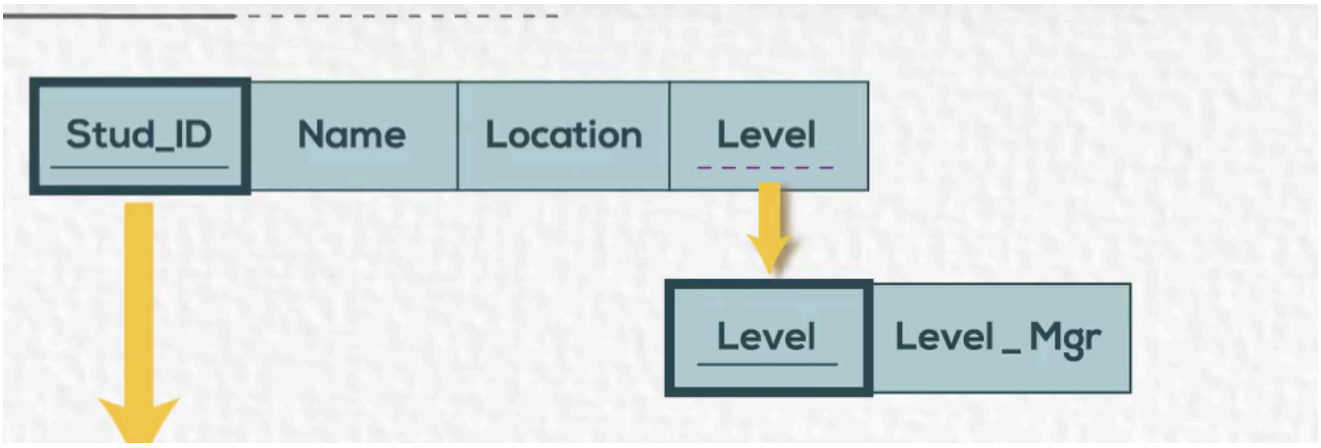

Stud_ID
Subject
Subj - Desc
G
Partially Dependent
Stud_ID
Subject
G
Subject
Subj - Desc



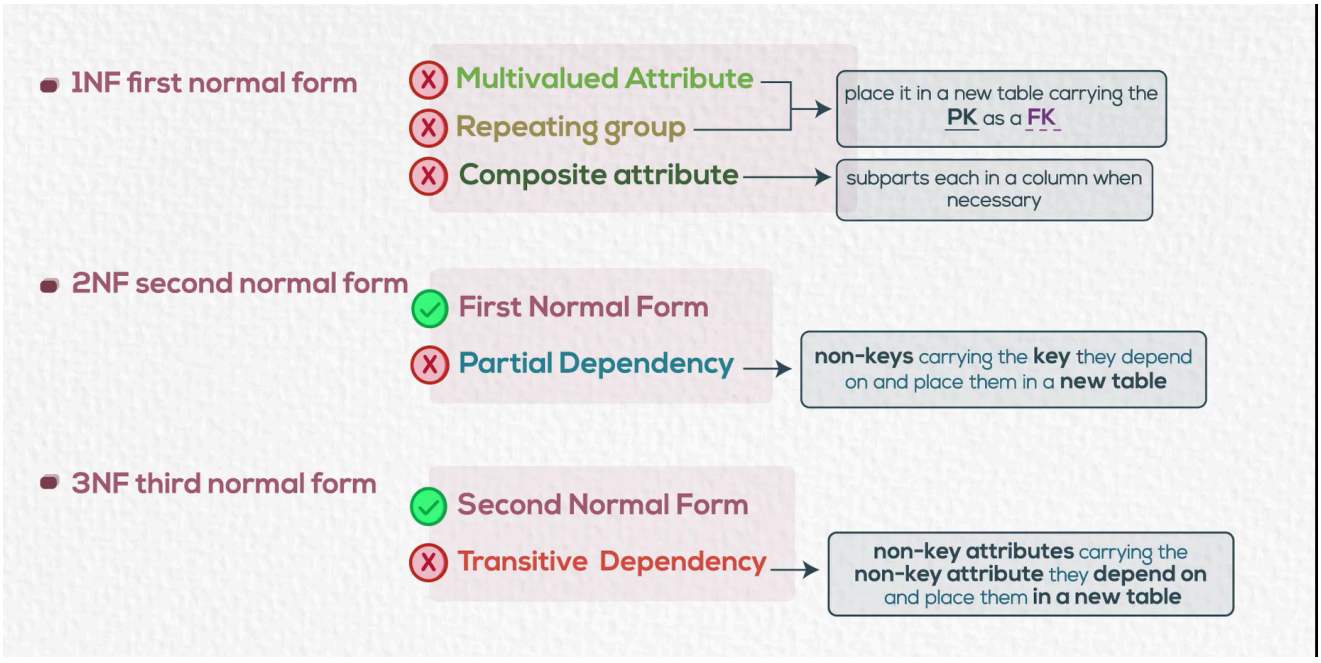

## Third Normal Form



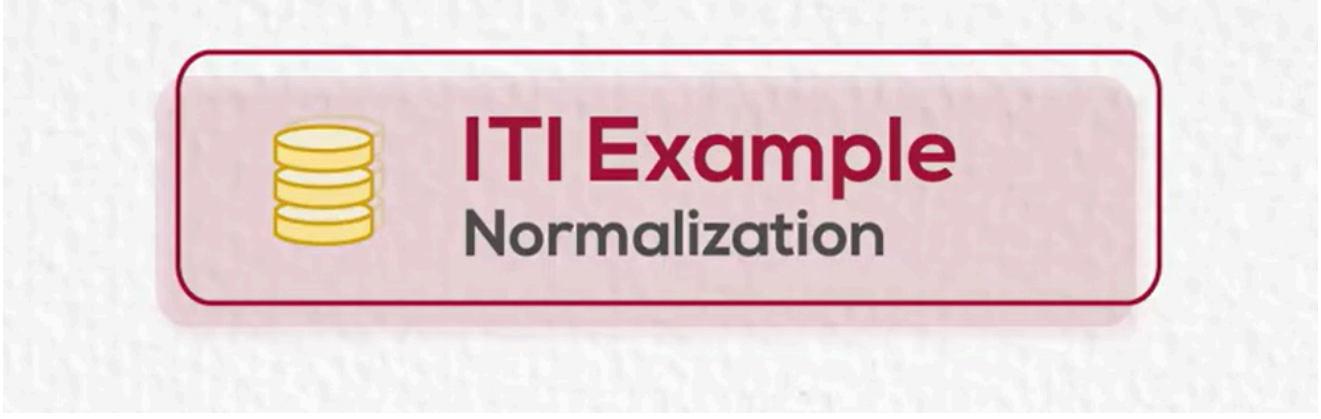



## Summary




## Example

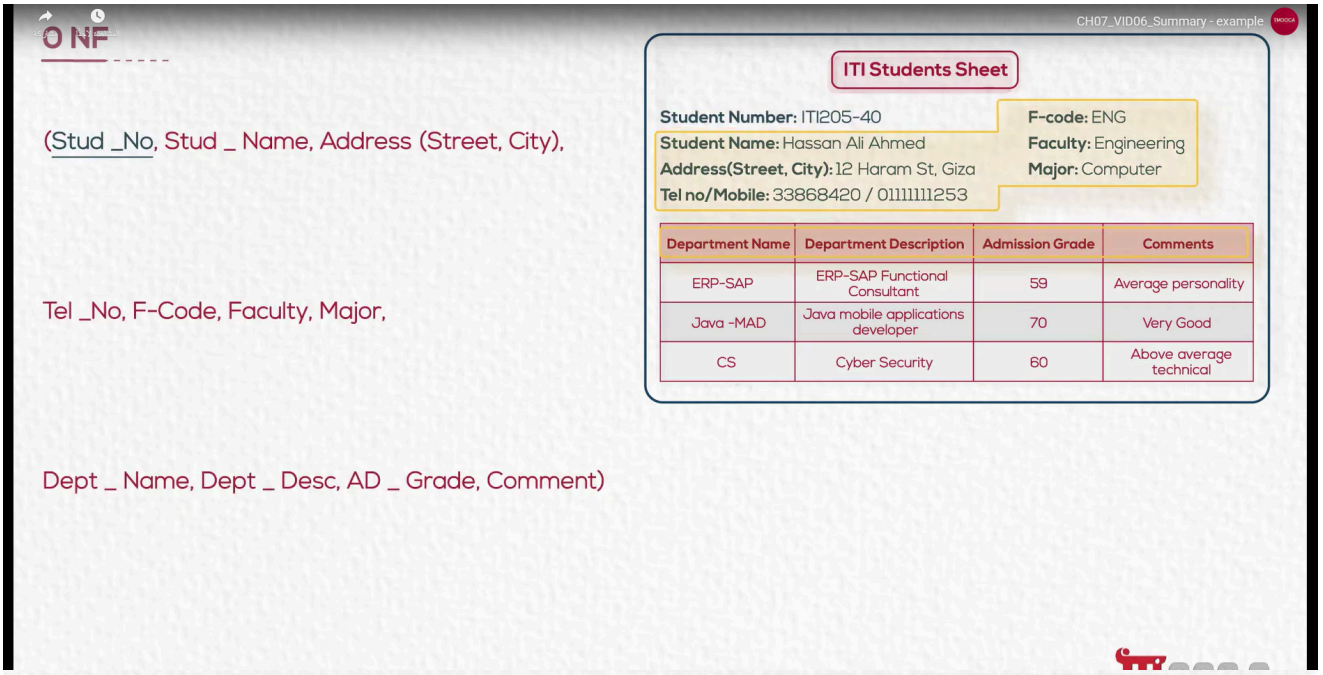


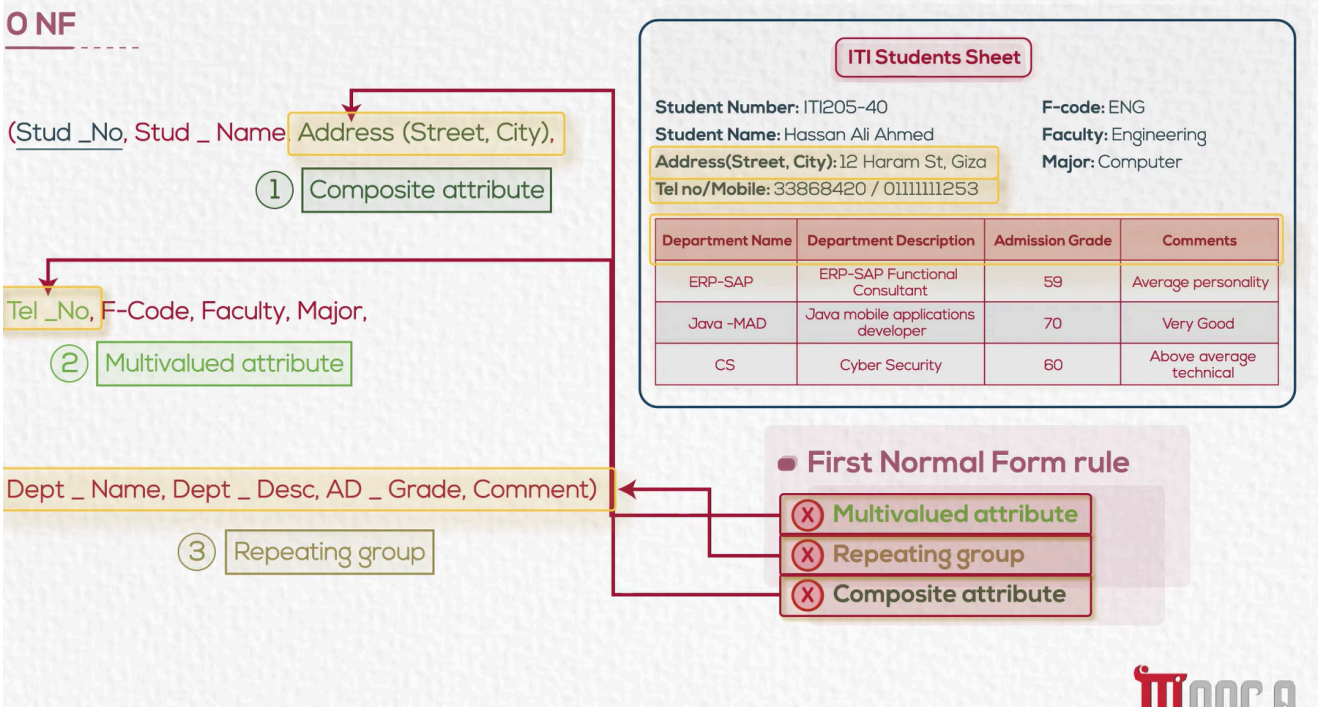
## ITI Students Sheet

**Student Number:** ITI205-40
**Student Name:** Hassan Ali Ahmed
**Address(Street, City):** 12 Haram St, Giza
**Tel no/Mobile:** 33868420 / 01111111253

**F-code:** ENG
**Faculty:** Engineering
**Major:** Computer

| Department Name | Department Description | Admission Grade | Comments |
|---|---|---|---|
| ERP-SAP | ERP-SAP Functional Consultant | 59 | Average personality |
| Java -MAD | Java mobile applications developer | 70 | Very Good |
| CS | Cyber Security | 60 | Above average technical |



# 2 NF

- Student (Stud_No, Stud_Name, F-code, Faculty, Major, Street, City)
- Student_Tel (Stud_No, Tel_No)
- Department_Student (Dept_Name, Stud_No, Dept_desc, Ad_Grade, Comments)

Dependent — Partially Dependent

**ITI Students Sheet**

**Student Number:** ITI205-40
**Student Name:** Hassan Ali Ahmed
**Address(Street, City):** 12 Haram St, Giza
**Tel no/Mobile:** 33868420 / 01111111253

**F-code:** ENG
**Faculty:** Engineering
**Major:** Computer

| Department Name | Department Description | Admission Grade | Comments |
|---|---|---|---|
| ERP-SAP | ERP-SAP Functional Consultant | 59 | Average personality |
| Java -MAD | Java mobile applications developer | 70 | Very Good |
| CS | Cyber Security | 60 | Above average technical |

- Second Normal Form rule
  - ✓ First Normal Form
  - ✗ Partial Dependency