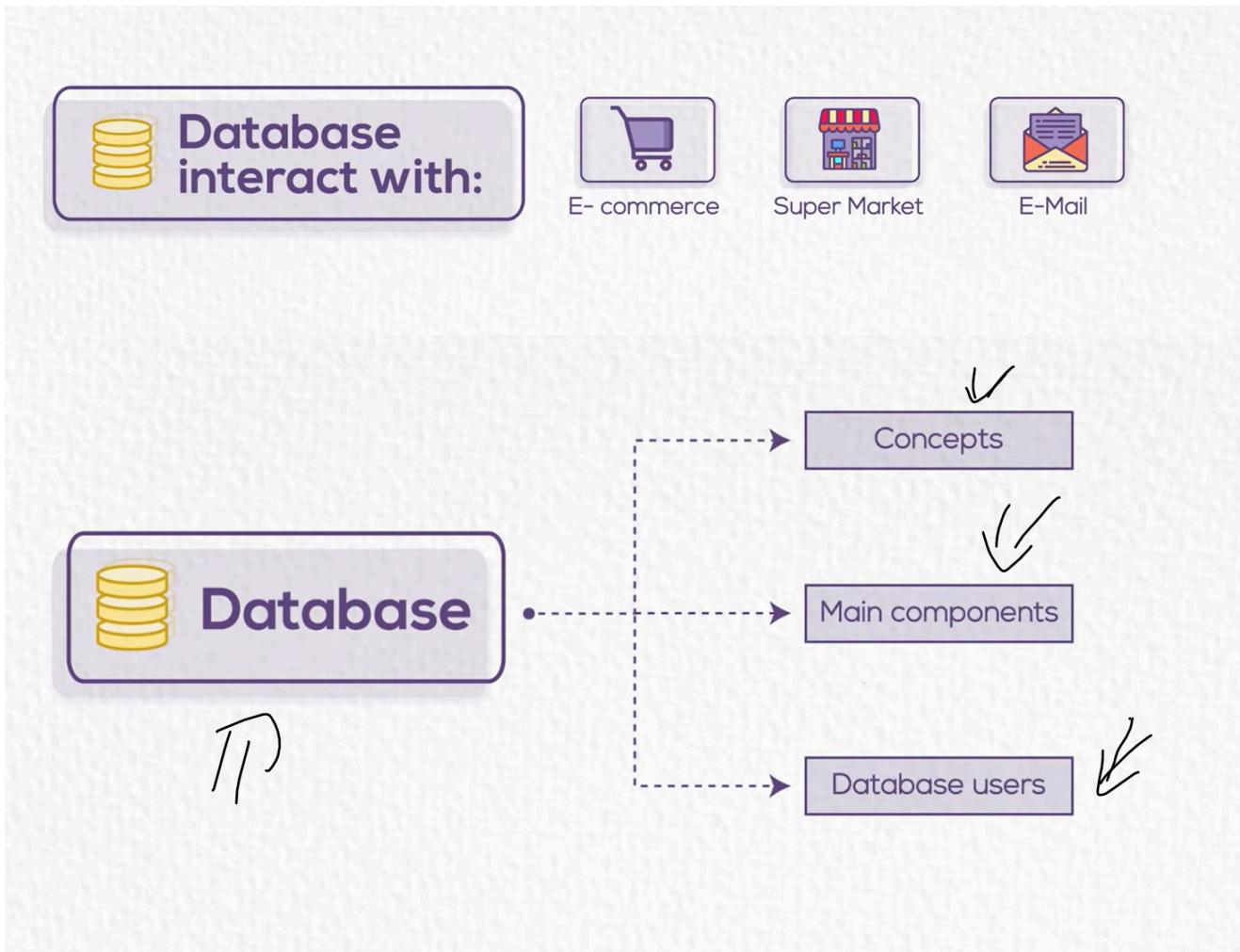


# Database Mahara Tech

## Chapter 1 - Introduction

Any App want a database to store data and retrieve it



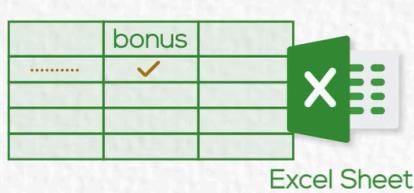
## File Based System



Finance dept.



HR dept.



Isolation Of data  
Incompatible file formats  
Duplicate data



Word file

Acquired a degree



## Limitations Of File Based System

- ✓ Separation & Isolation Of data.
- ✓ Duplication of data
- ✓ Program Data Dependence
- ✓ Incompatible file formats

## Database

التعريفات

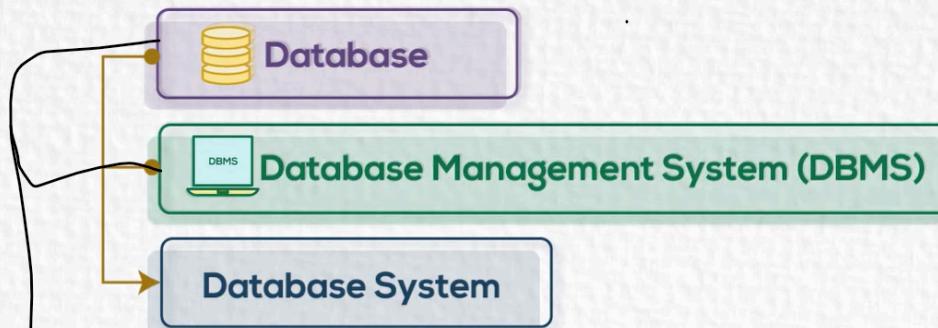


## Database

- A collection of related data.

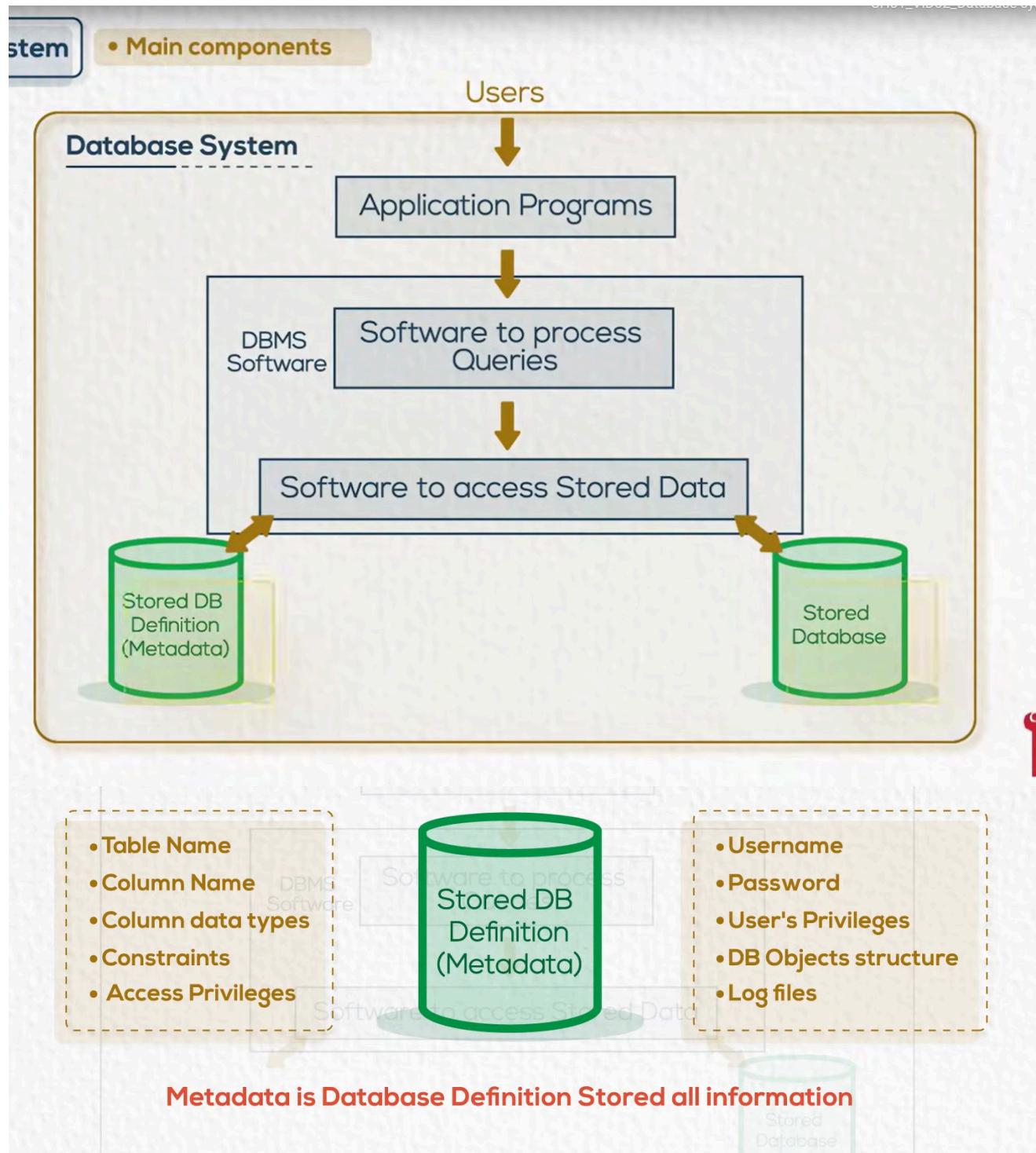
## Database Management System (DBMS)

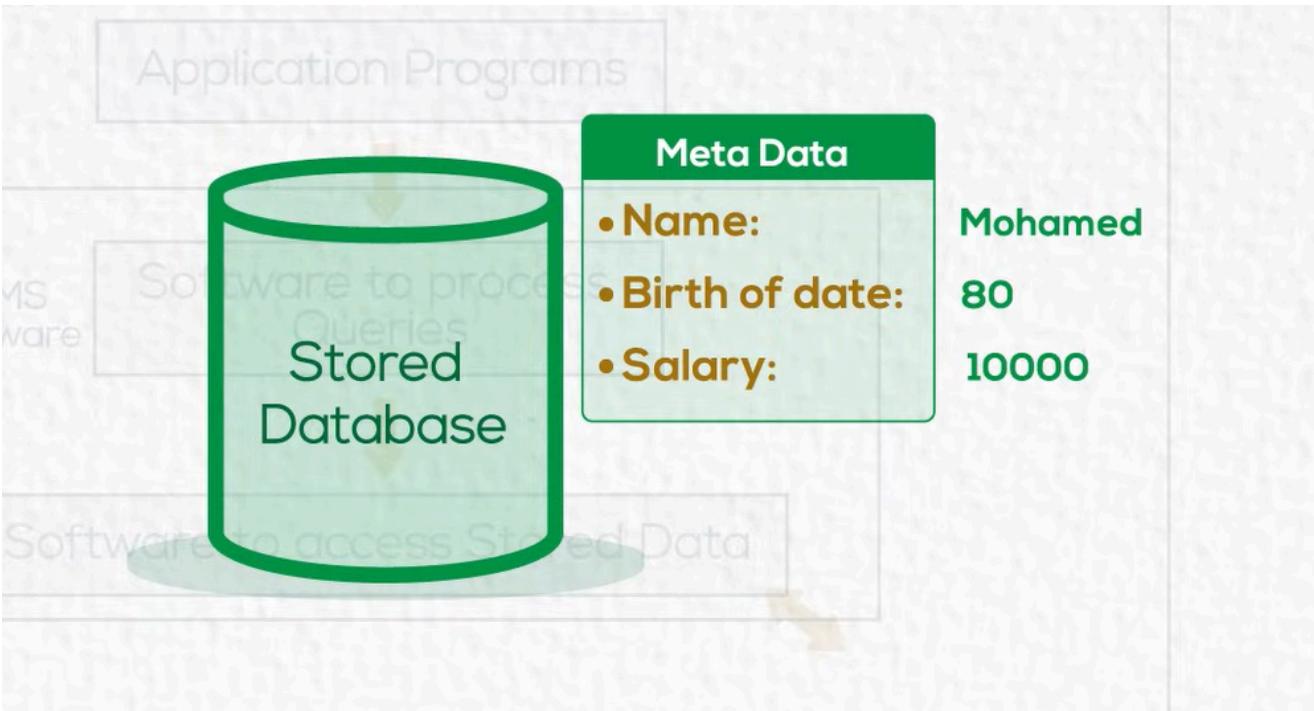
- A software package/ system to facilitate the creation and maintenance of a computerized database.



- The **DBMS** software **together** with the data itself. Sometimes, the applications are also included. **(Software + Database)**

## Main Components



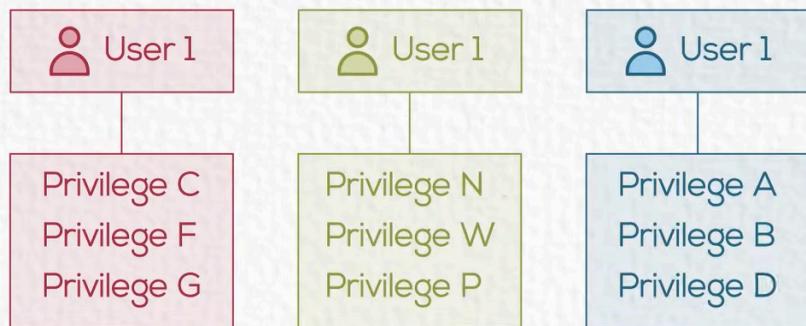


## Advantages

- Controlling Redundancy.



- Restricting Unauthorized Access.



- Sharing data.

- Enforcing Integrity Constraints.

Integrity of data							
Phone number	ID						
<table border="1"> <tr> <td>123456</td><td>123</td></tr> <tr> <td>Text</td><td>456</td></tr> <tr> <td>789101</td><td>456</td></tr> </table>	123456	123	Text	456	789101	456	
123456	123						
Text	456						
789101	456						

- Enforcing Integrity Constraints.

**Inconsistency can be avoided.**



- Inconsistency can be avoided.

- Providing Backup and Recovery.

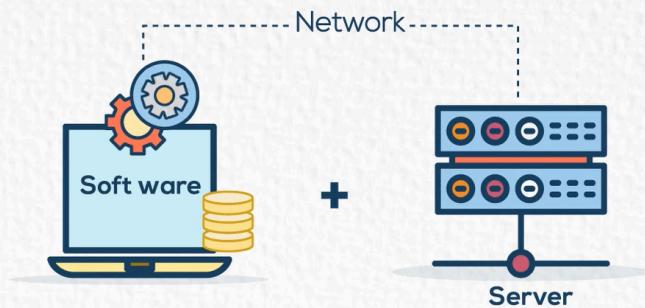


## Disadvantages

■ Needs expertise to use.

- DBMS is expensive.

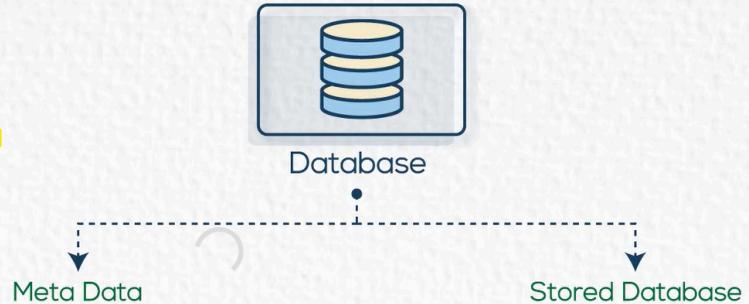
	Support DBMS
	Database
	User



- Needs expertise to use.

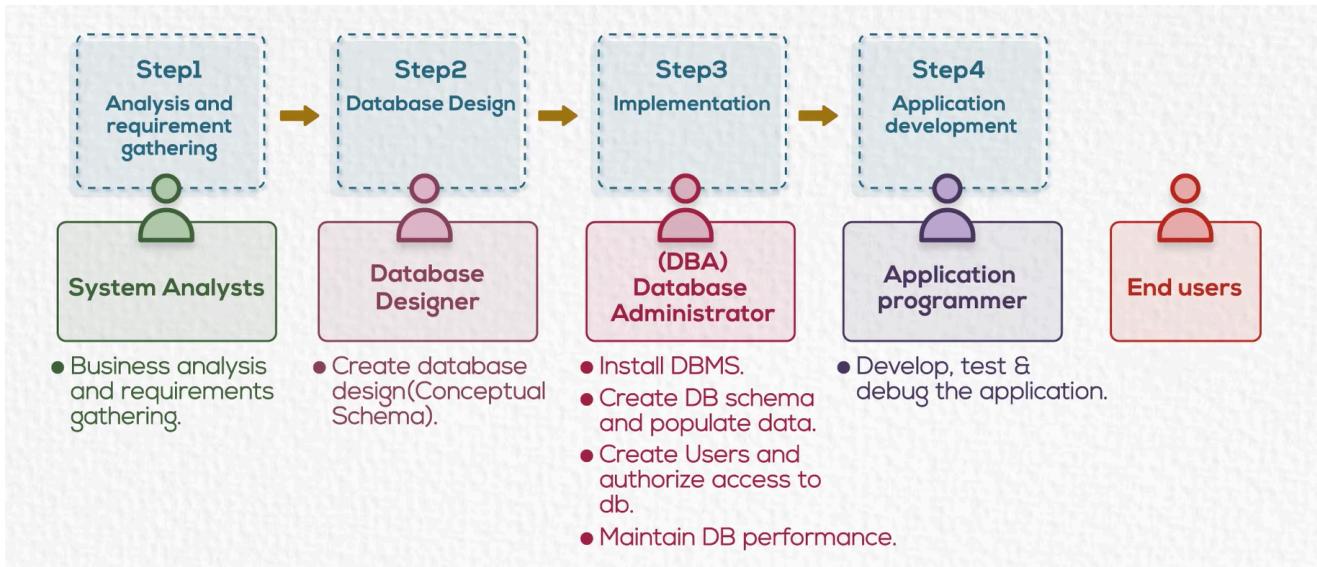
- DBMS is expensive.

**■ May be incompatible with any other available DBMS.**



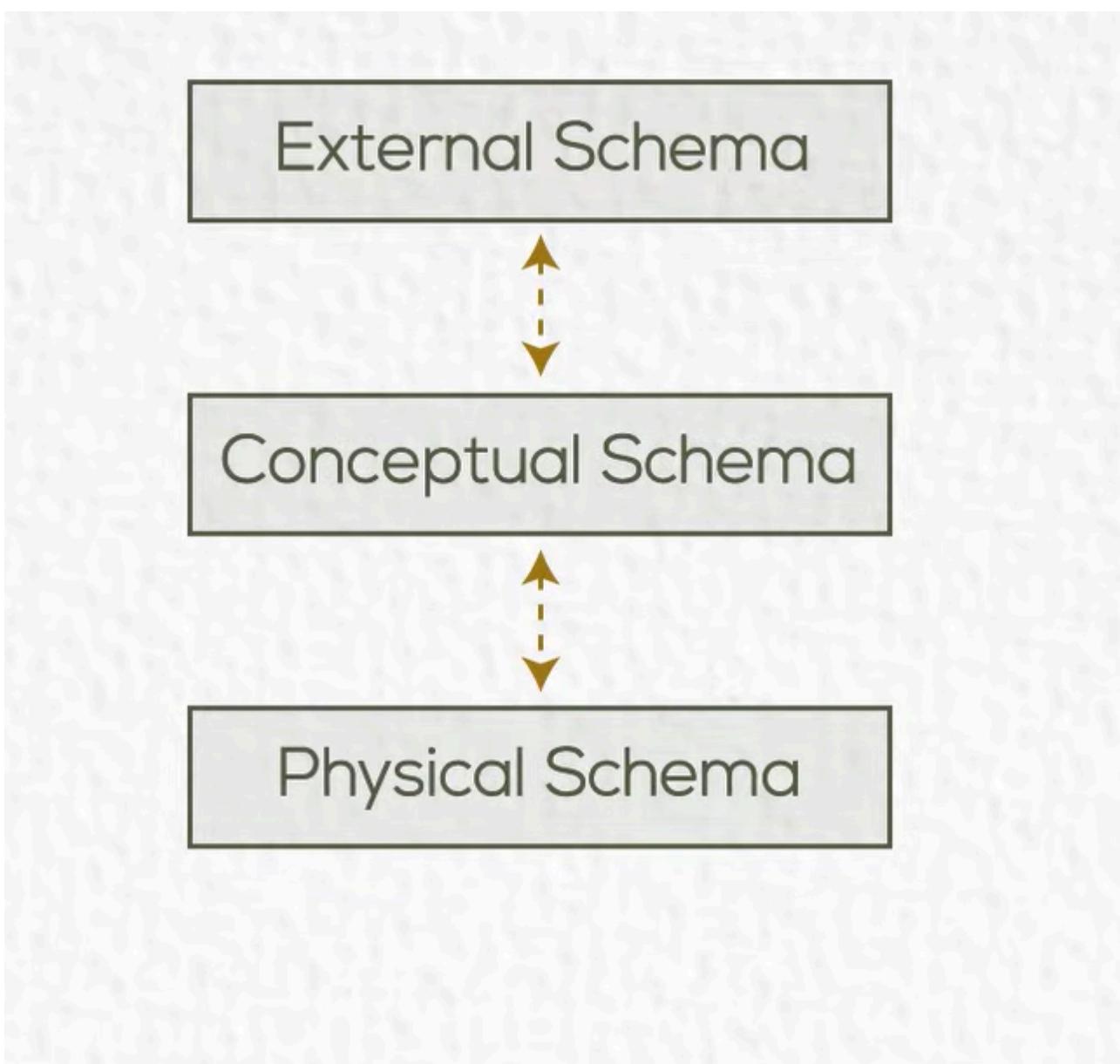
## Database Users

### Process

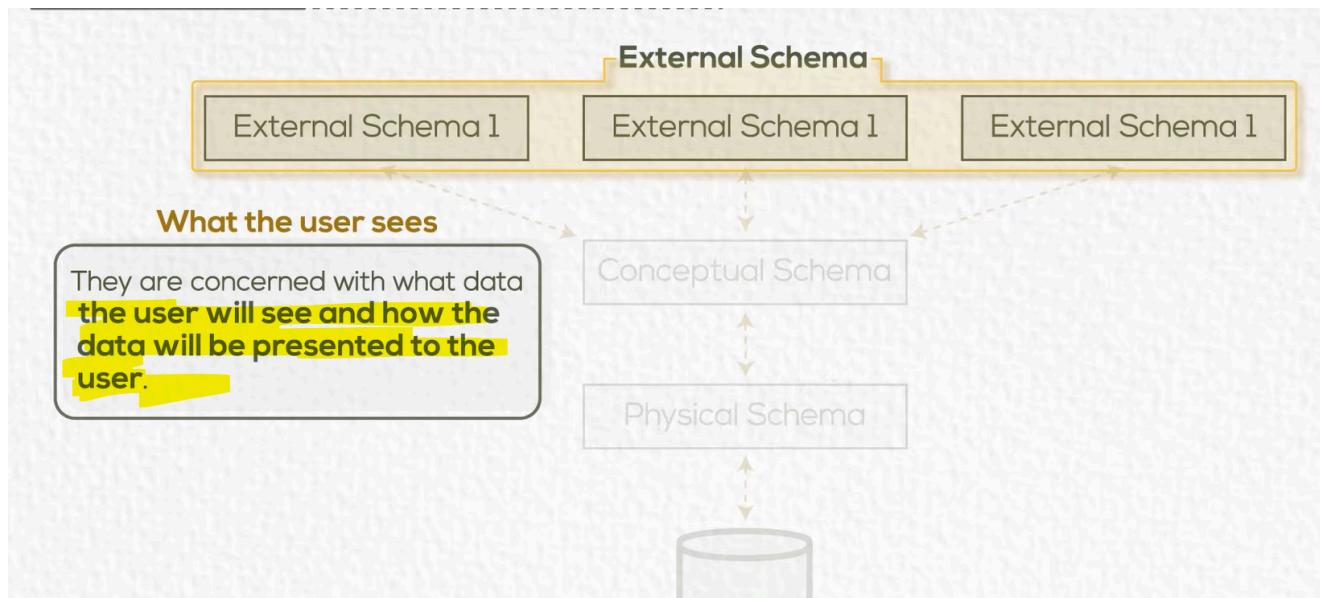


## DBMS Architecture

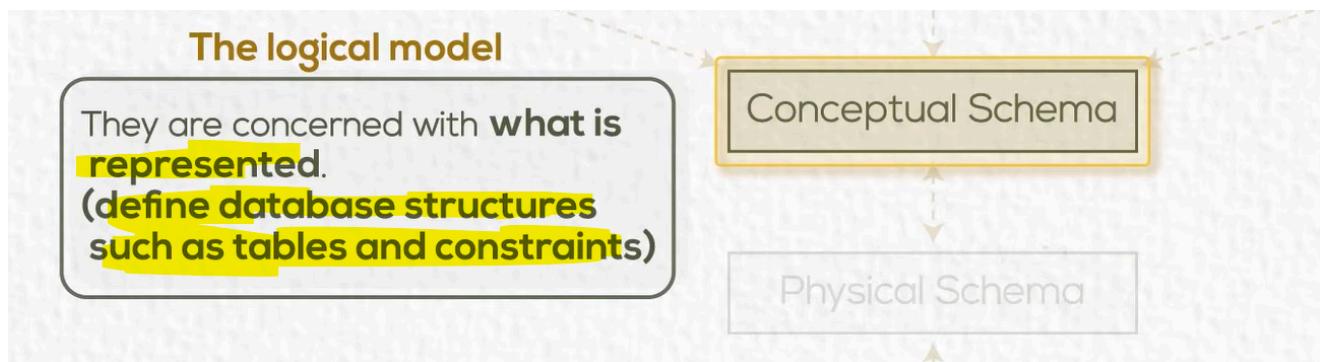
### Three Types of Schema



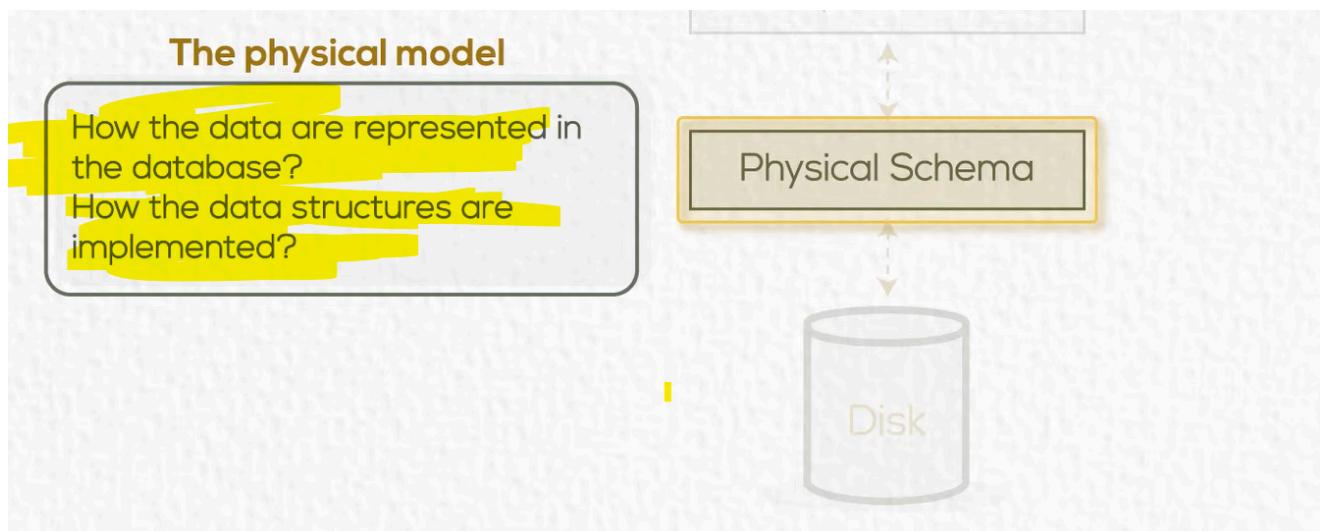
## External Schema



## Conceptual Schema



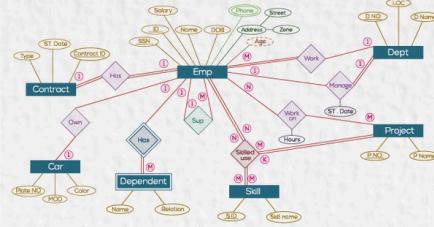
## Physical Schema



## Data Models

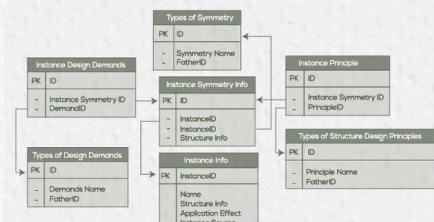
## Data Models

### The logical model /conceptual model



provide concepts that are close to the way many users perceive data, entities, attributes and relationships. (Ex. ERD)

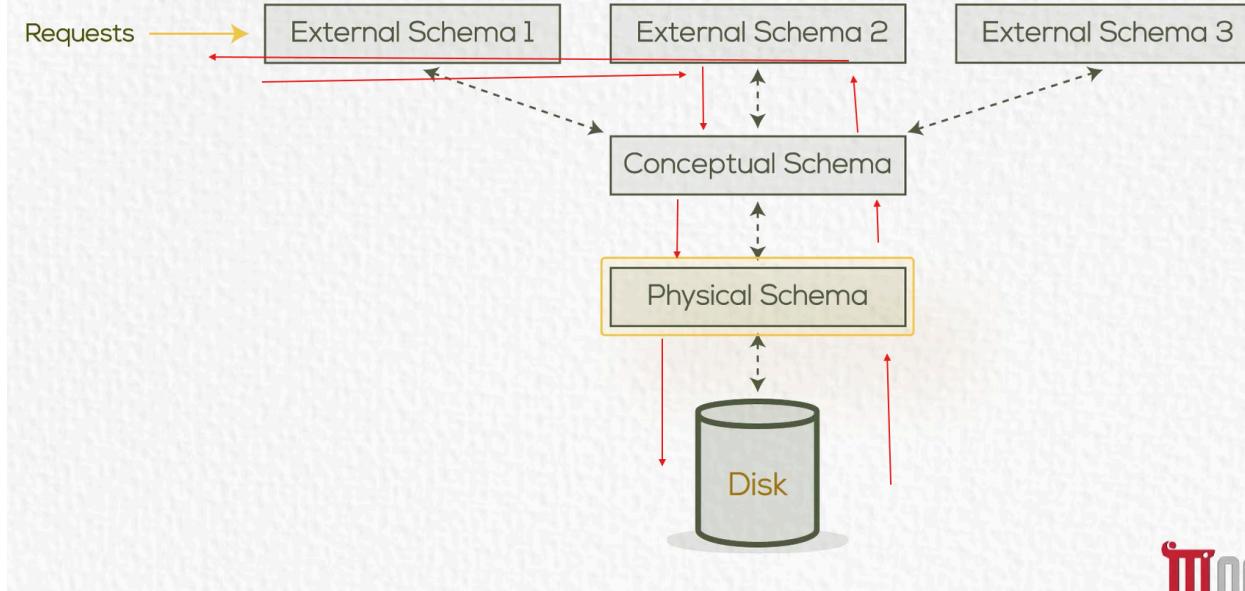
### The physical model



describes how data is stored in the computer and the access path needed to access and search for data.

## Mappings

- **Definition:** It is the processes of transforming requests and results between levels.



MOOCs

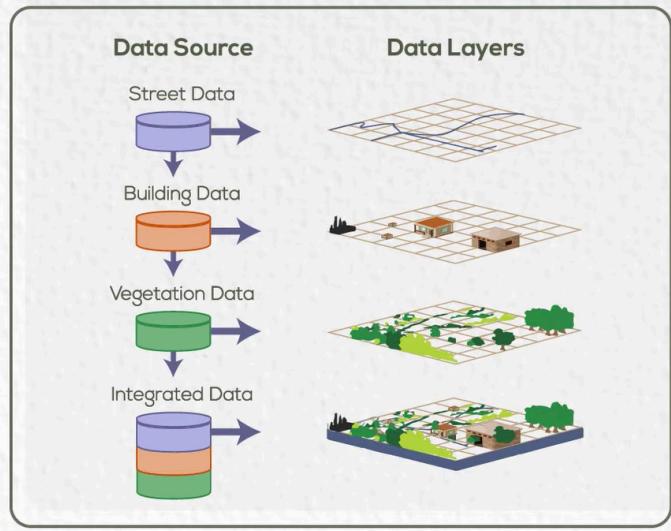
## DBMS Other function



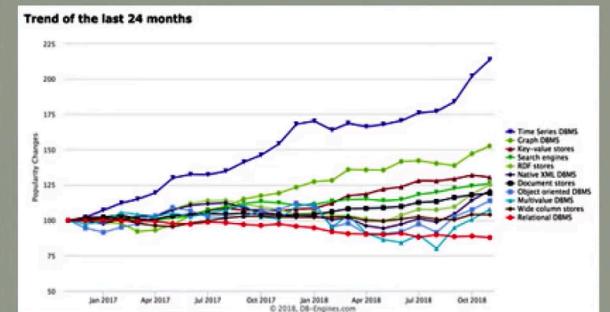
## Text/Number/Image/Audio/ Video

### DBMS Other Function

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



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



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

Clustering  
Classification  
Association Rules

**Example**



## DBMS Environment

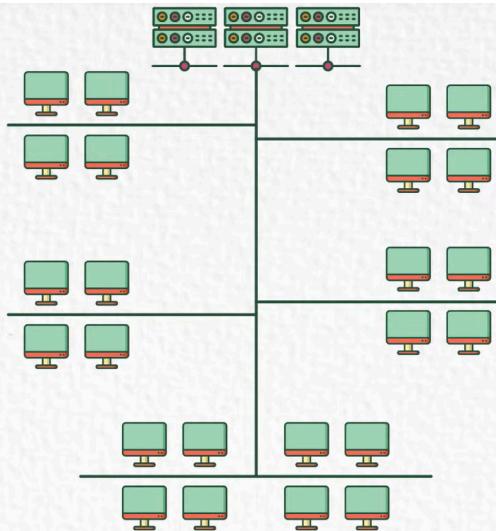
### Centralized Database Environment

## 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.

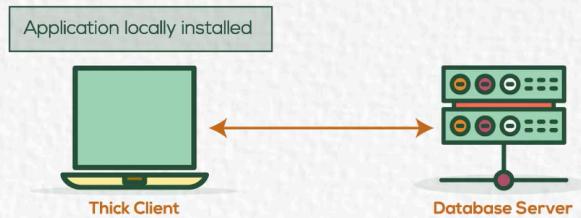


### Problems with this environment

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

### Advantages

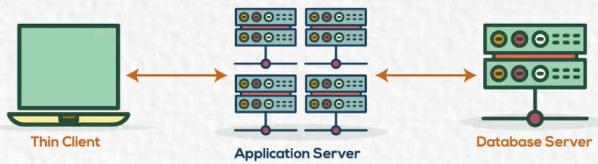
- Application layer **isn't** a single Point of failure.



### ③ Internet Computing environment (Three-tier architecture).



Problems with this environment



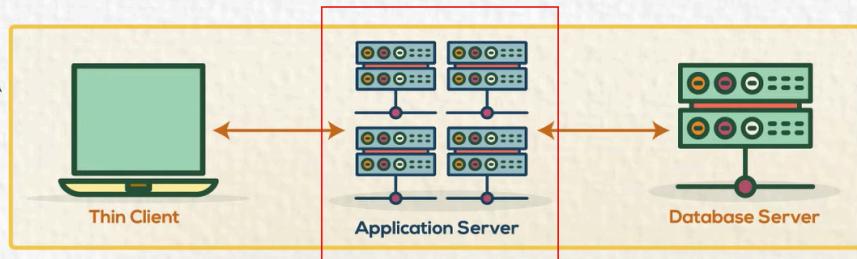
- Application server is a single point of failure.
- Database is a single point of failure.

Advantages

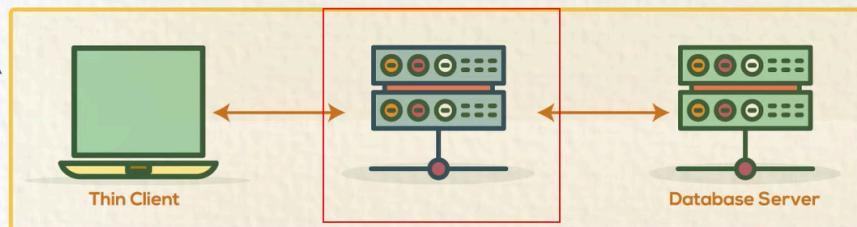
- Lower cost for support and maintenance.

### ③ Internet Computing environment (Three-tier architecture).

N-tier architecture



Three tier architecture



## Distributed Database Environment

Distributed Database

Support high availability of Database

Replication



Fragmentation



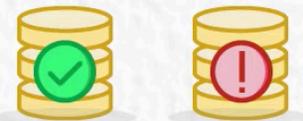
se

## Replication

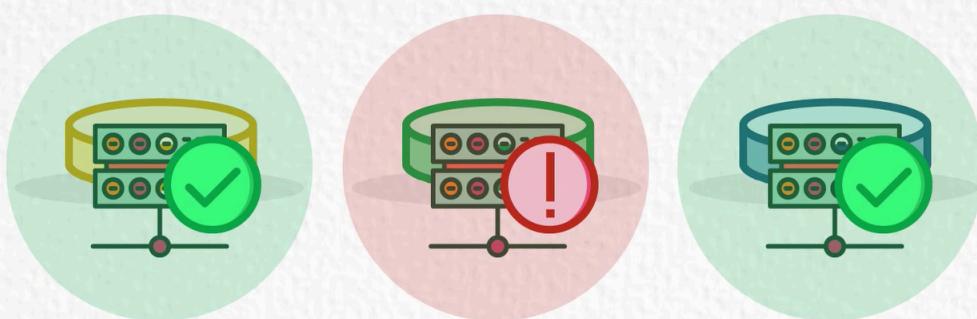
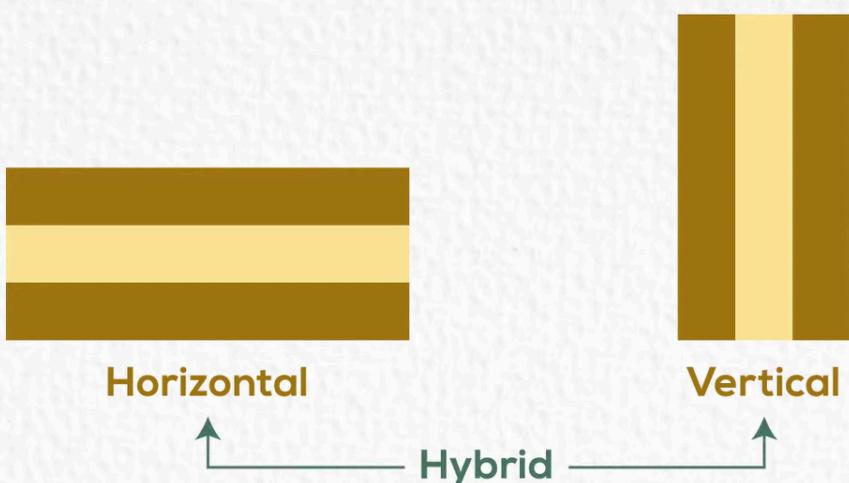
Partial Replication



Full Replication

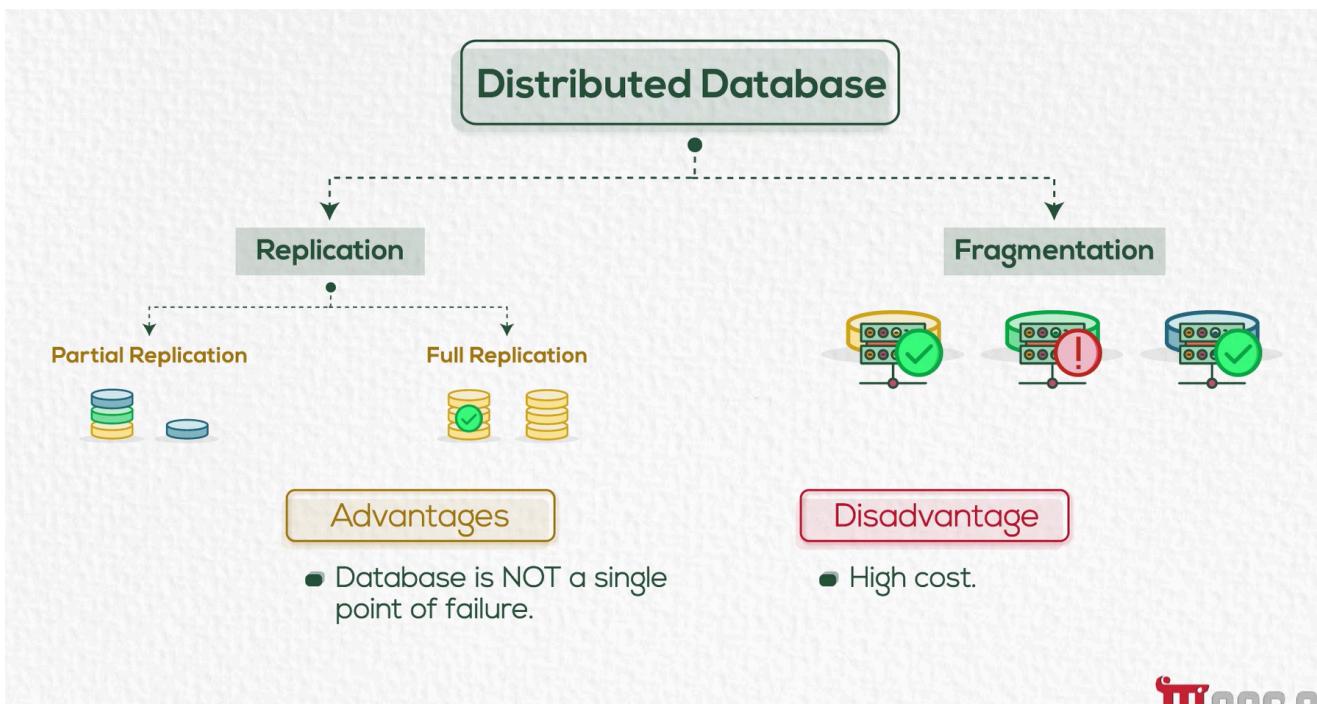
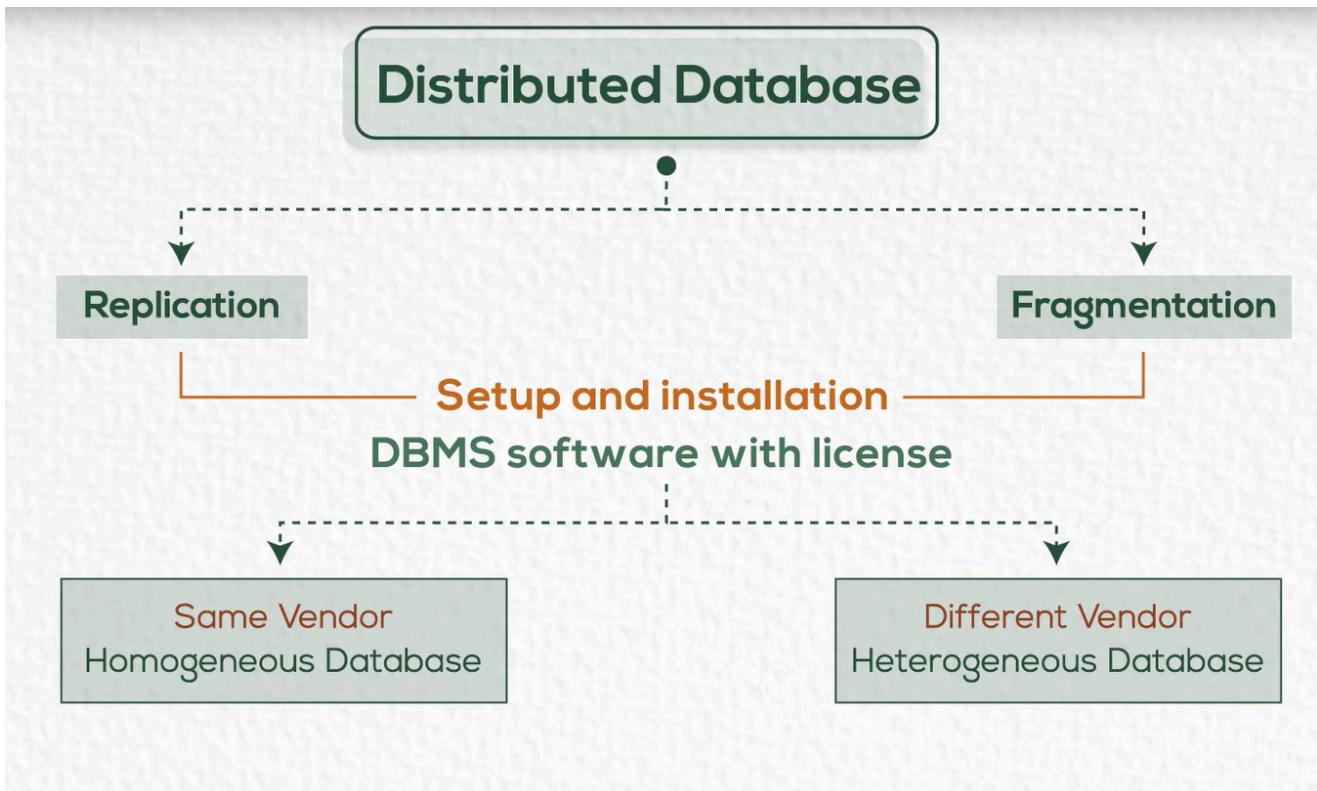


# Fragmentation



## Advantages

- Database is NOT a single point of failure.



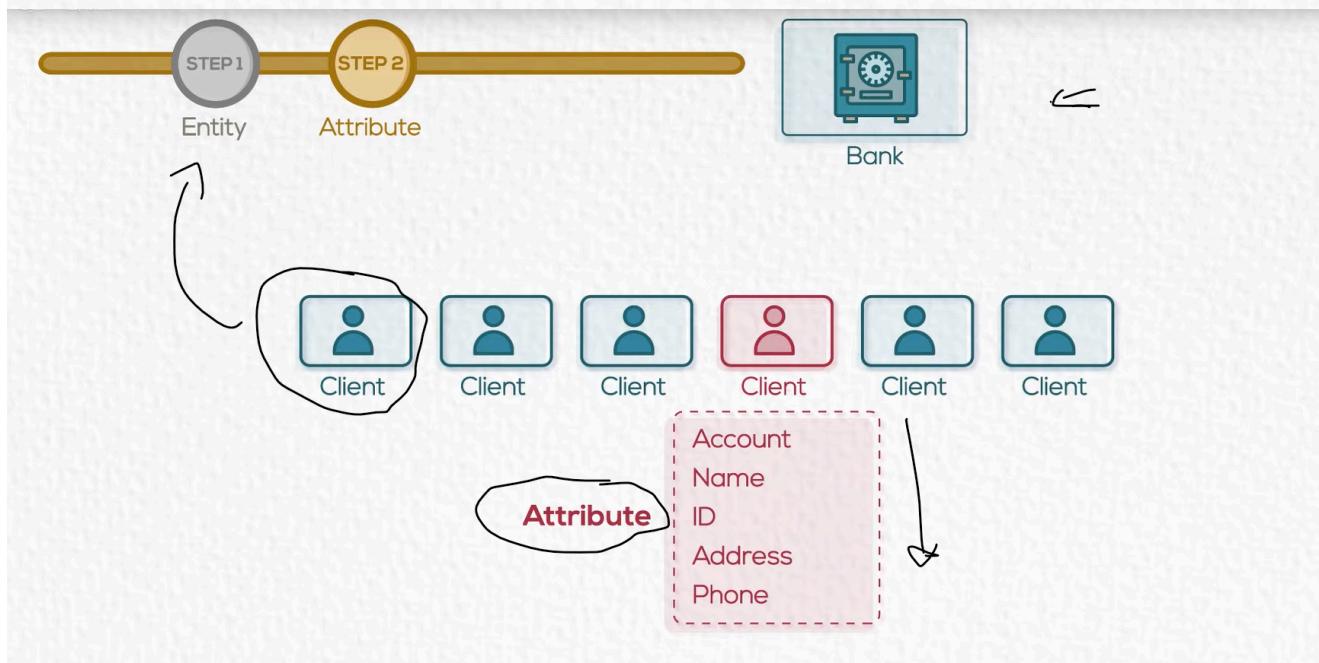
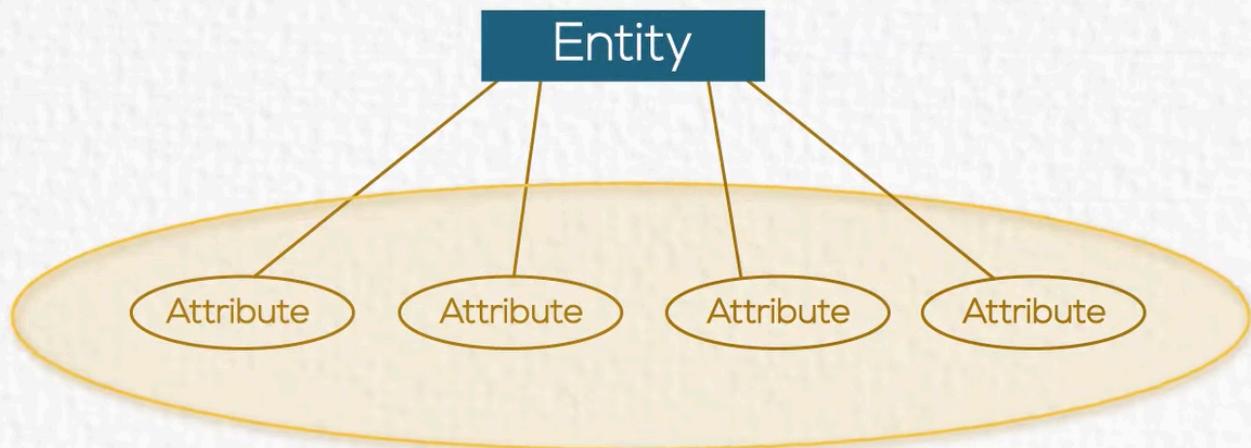
## Chapter 2 - Entity Relation Diagram

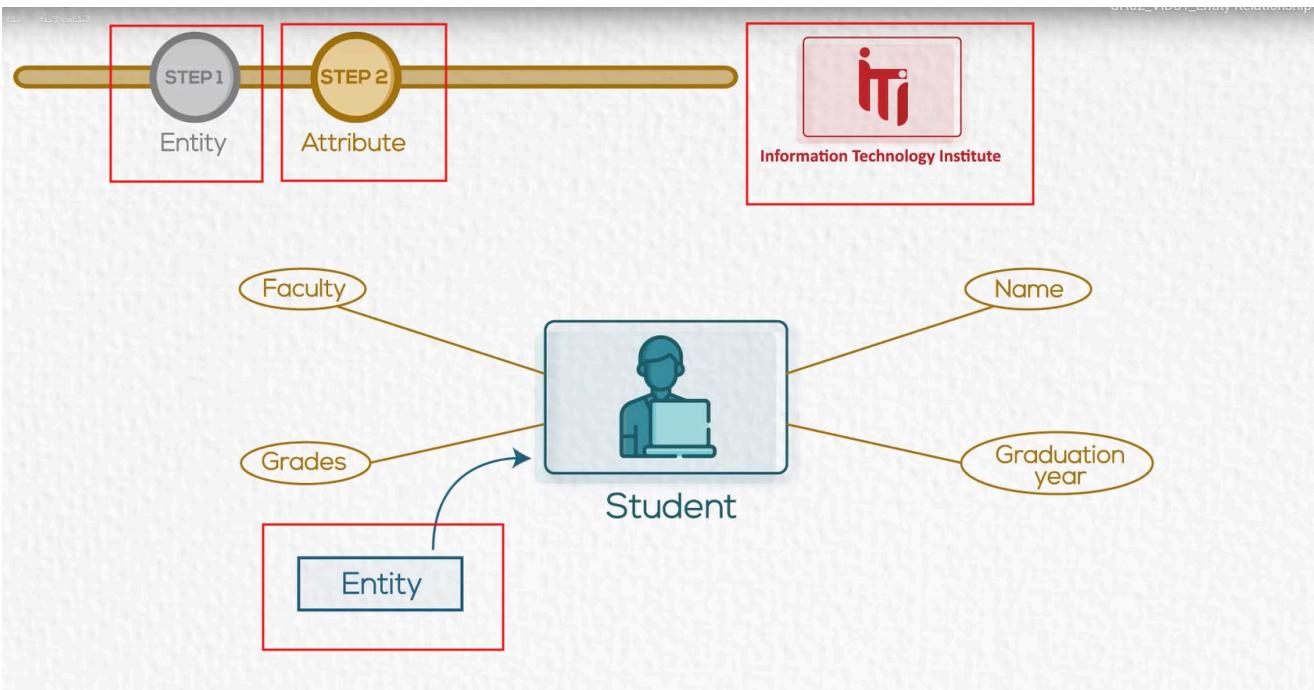
### Entity Relation Diagram (ERD)

- 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 بتعايي 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 تحديد العلاقات بين كل الكيانات وبعضها



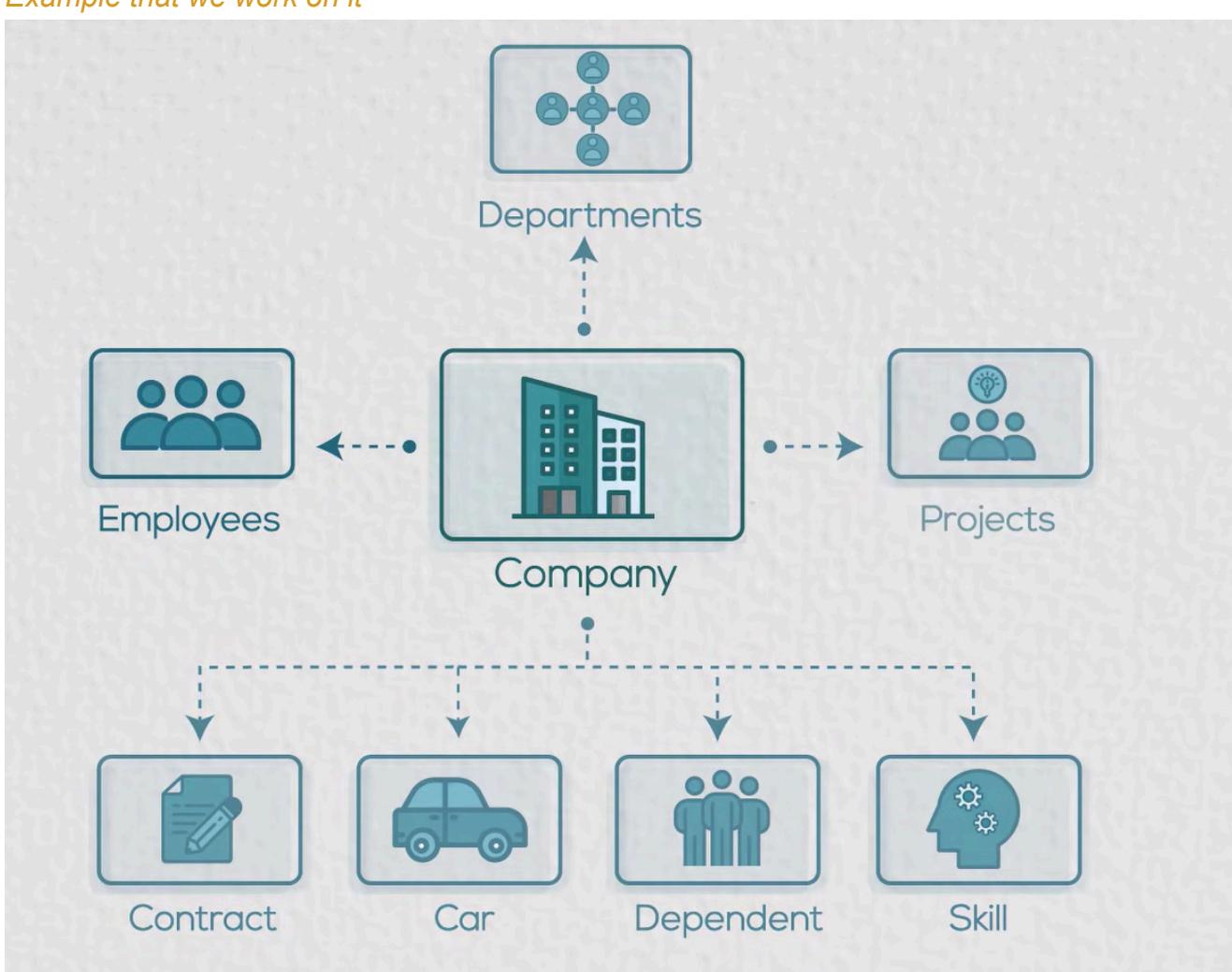
4- What associations or relationships exist between entities?

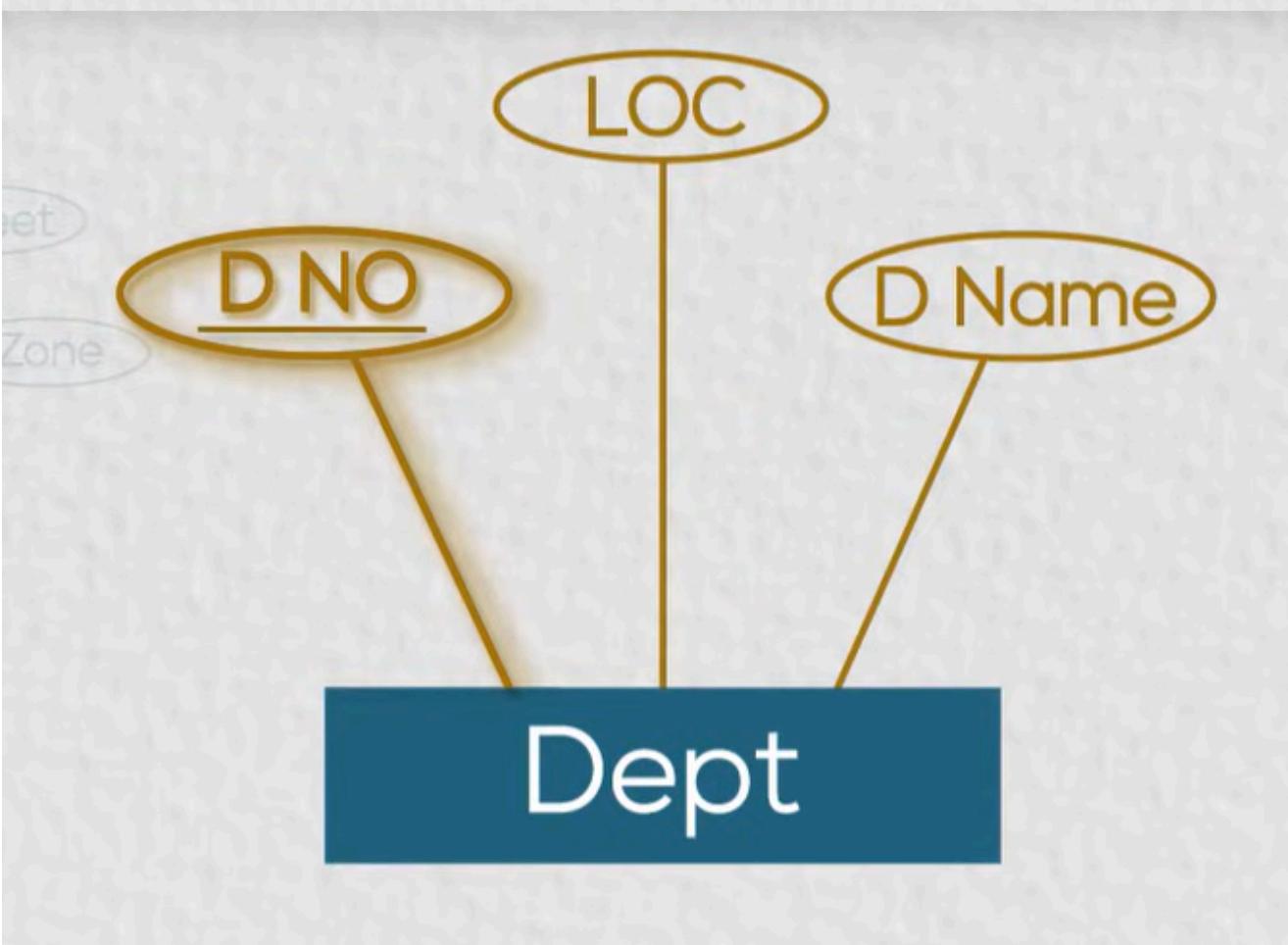
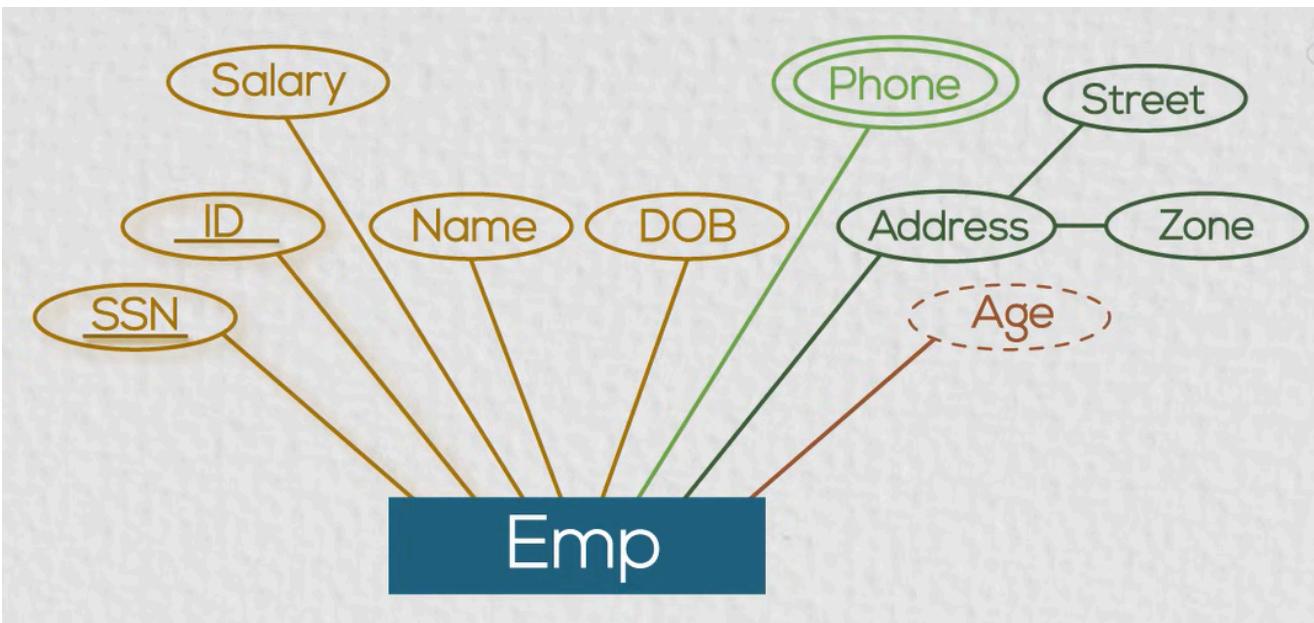
## Conceptual Design

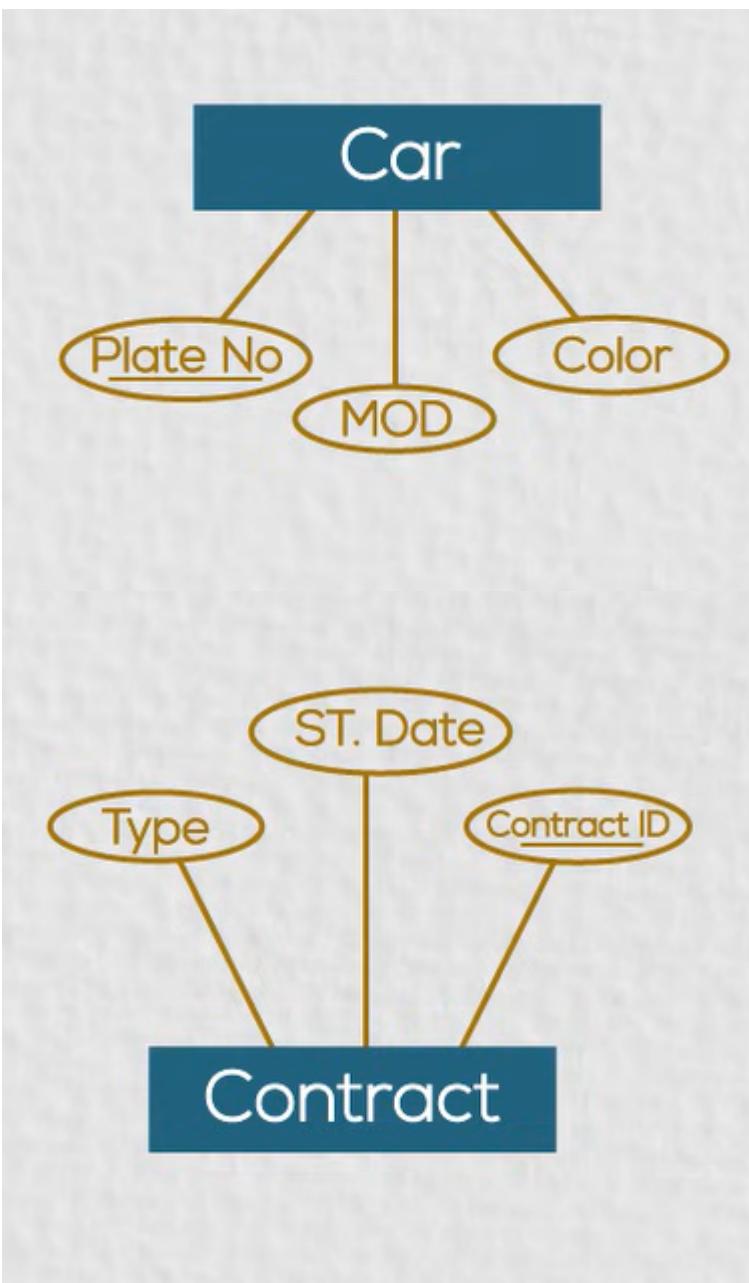
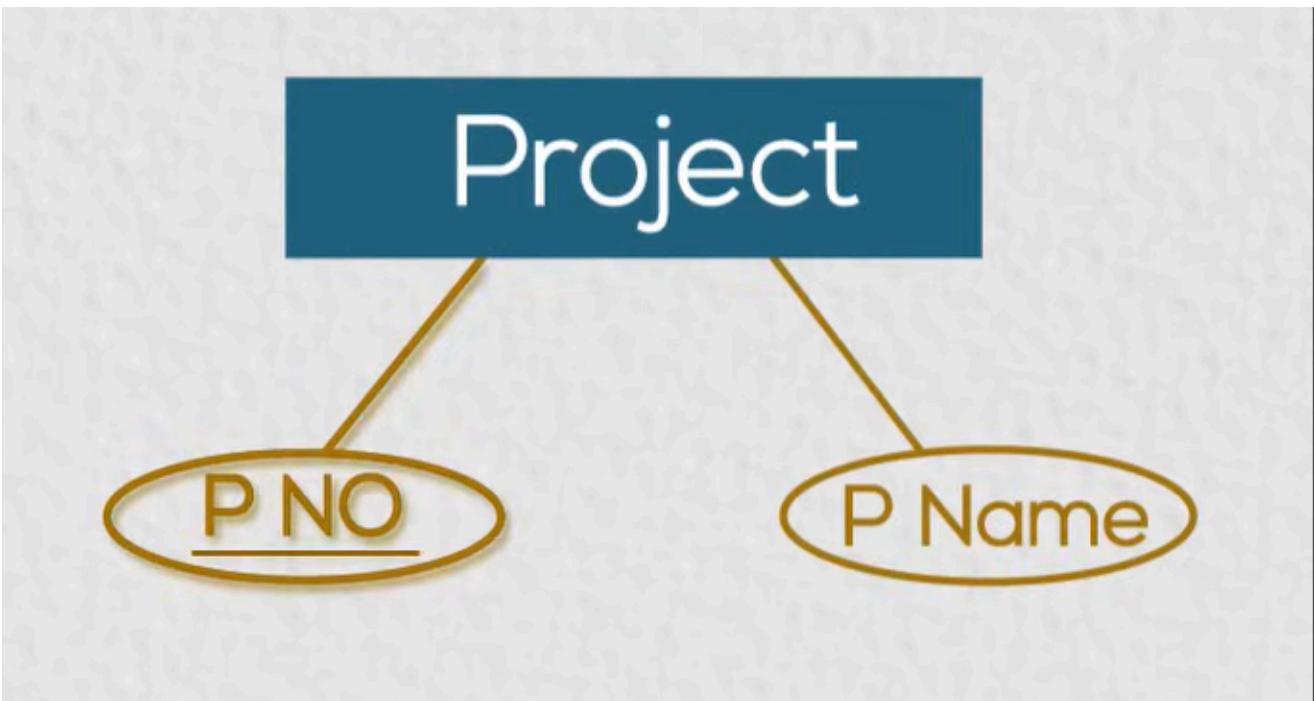


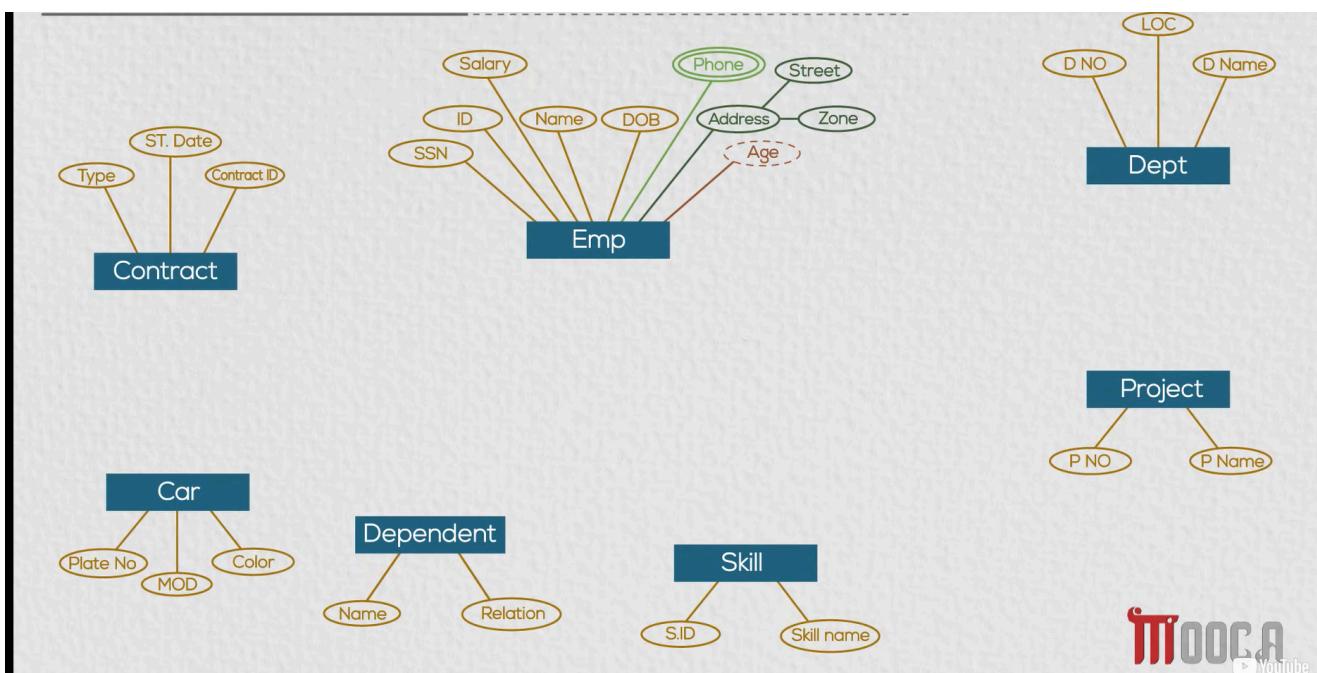
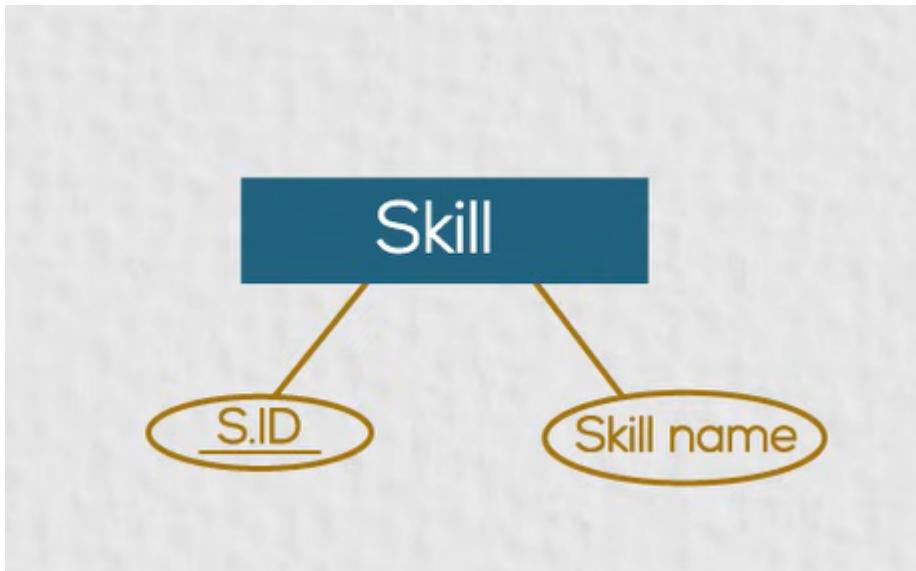
# Conceptual Design

*Example that we work on it*

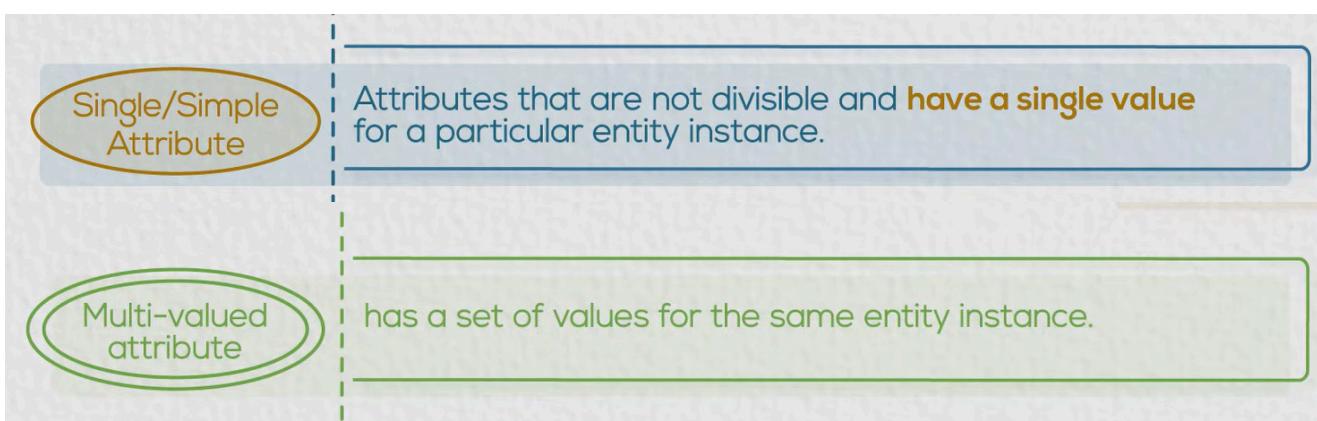


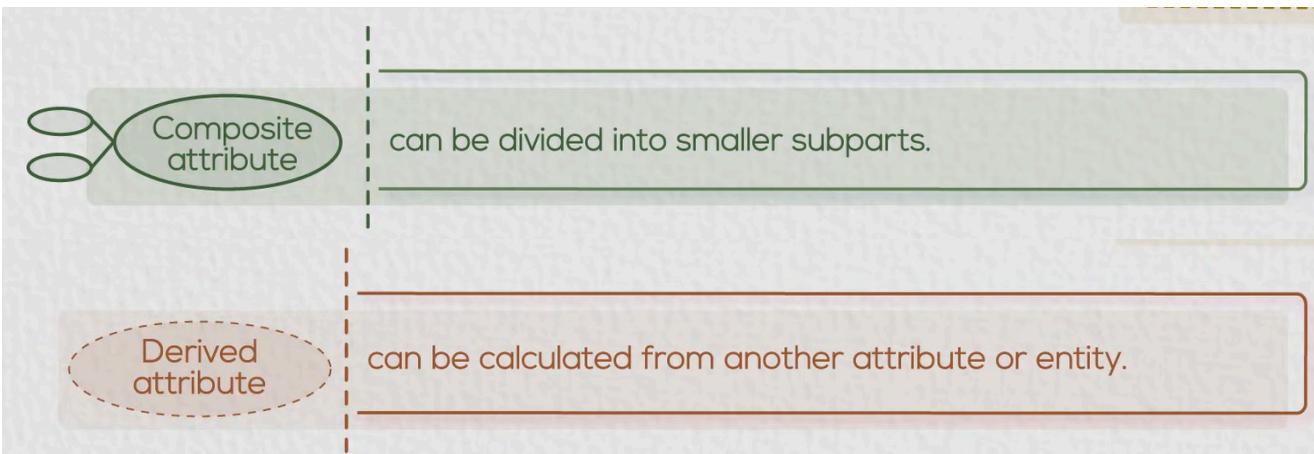






## Types of Attributes





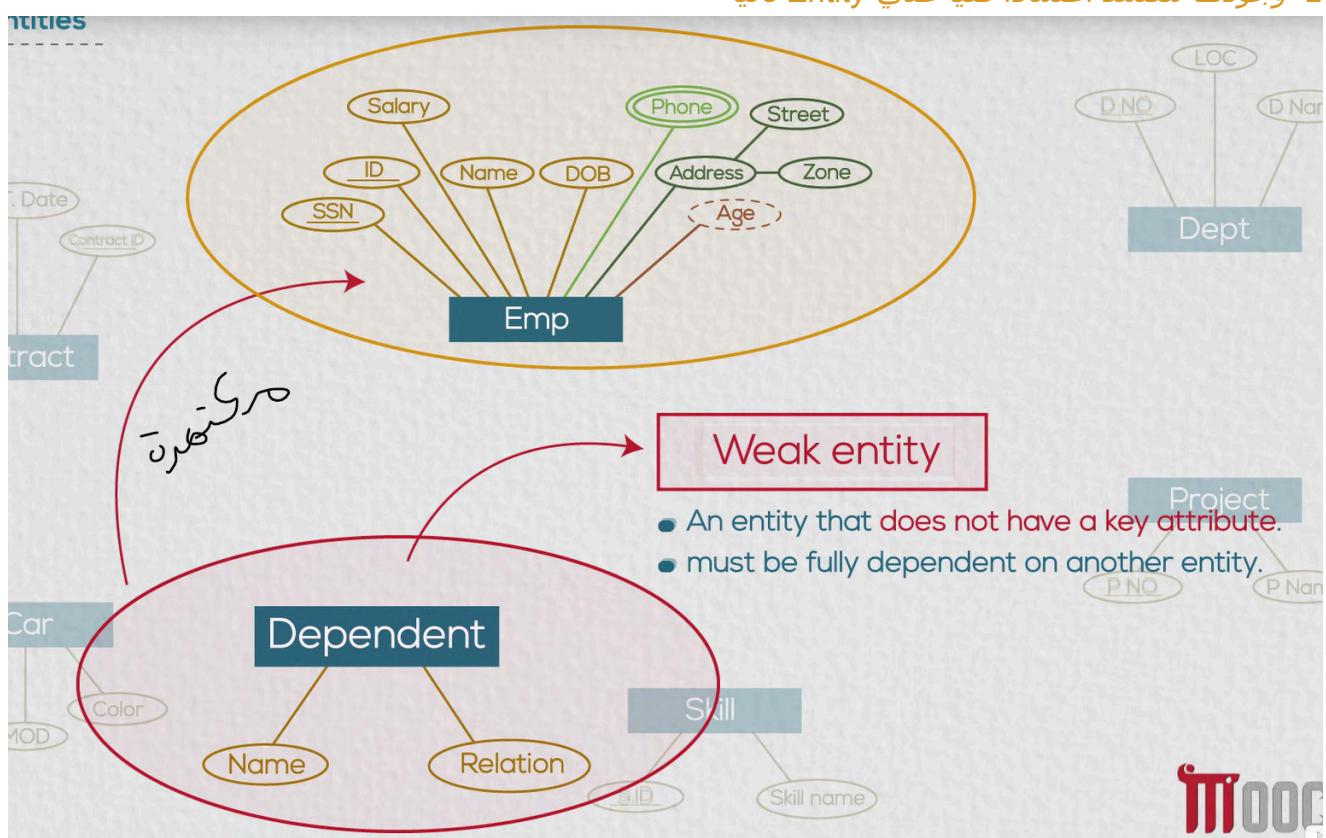
## Types of Entity

ممكن تحدد لها unique key في ميزه من مميزاتها بخليلها مختلفه عن الباقي او اقدر احددها بيها

## Strong entity

بتحقق عندي شرطين

- 1- وهي مفيش ولا Attribute من بتوعها اقدر احددها بيها
- 2- وجودها معتمد اعتمادا كلها على Entity تانية

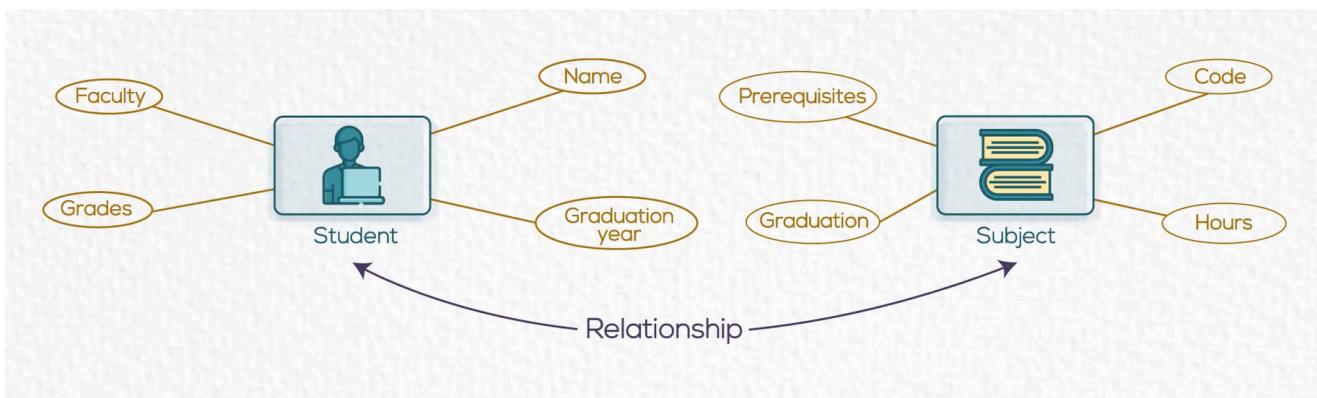


## Types of Keys

## Candidate Key

when an entity type has more than one key, those are candidate keys.

## Relationships



## Relationships

A relationship is a connection between entity classes.

Degree of a relationship

Cardinality Ratio

Participation

## Degree of Relationship

العدد الذي يشارك في العلاقة دي

is the number of participating entity.

## Cardinality Ratio

هتروح تقف عند كل Entity اللي موجودين في العلاقة وليكن مثلا عندنا موظف وقسم هتروح تقف عن الموظف وتسأل هل الموظف بيشتغل في اكثر من قسم ولا قسم واحد طبعا قسم واحد تروح

تحط واحد عن القسم و تقف عند القسم و تسأله هل القسم يشتغل فيه اكتر من موظف ولا موظف واحد ولا اكيد اكتر من موظف تروح تحط  $n$  عند الموظف وهكذا بقي

- Specifies the maximum number of relationship

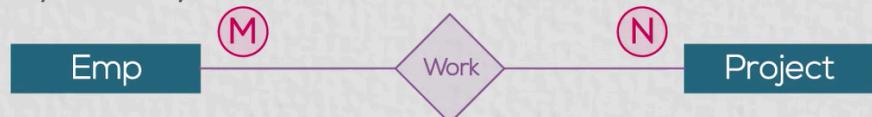
1- One to many



2- One to one

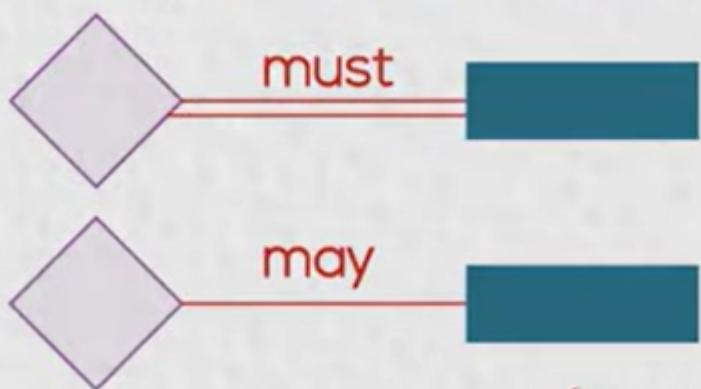


3- Many to many



## Participation

- specifies the minimum number of relationship instances that each entity can participate with.



## Types of Relationships



## Binary relationship

بین ال Entity ونفسها



## Unary / Recursive relationship



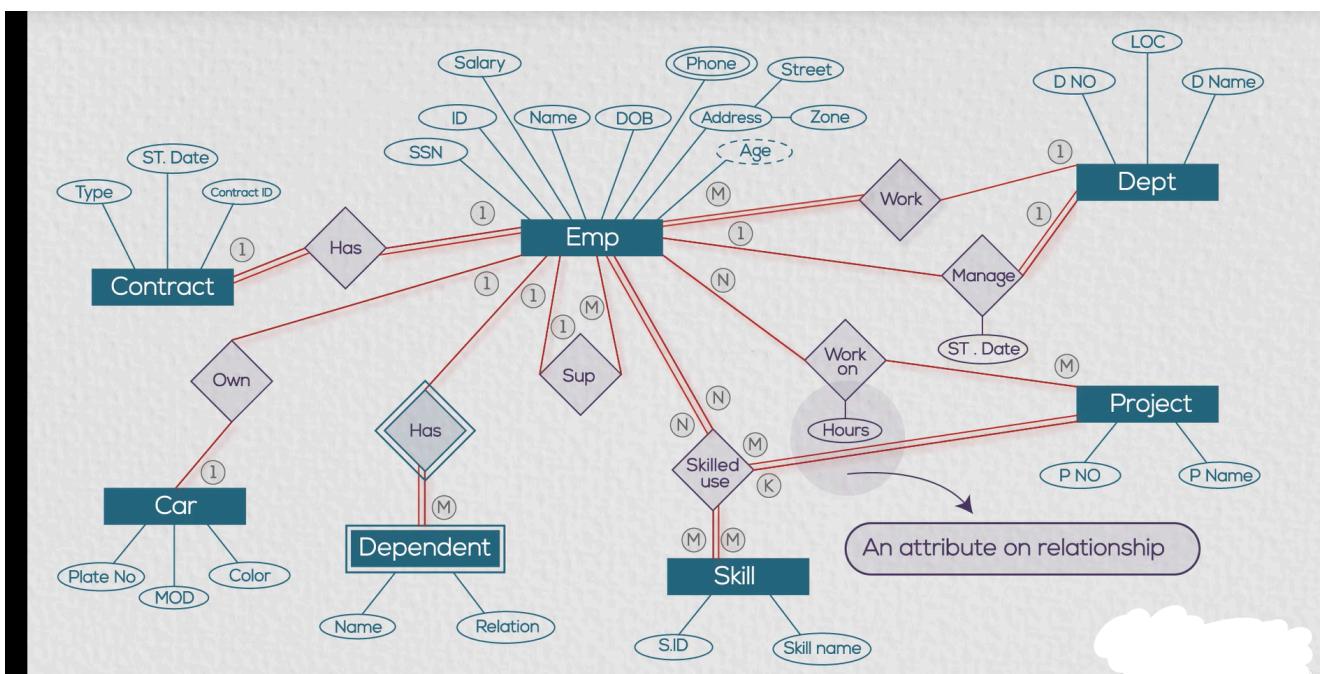
## Ternary relationship

بتكون بين ال Strong Entity and Weak Entity ، ويكون ليها خطلين فوق بعض كدا



## Identifying relationship

## Final ERD

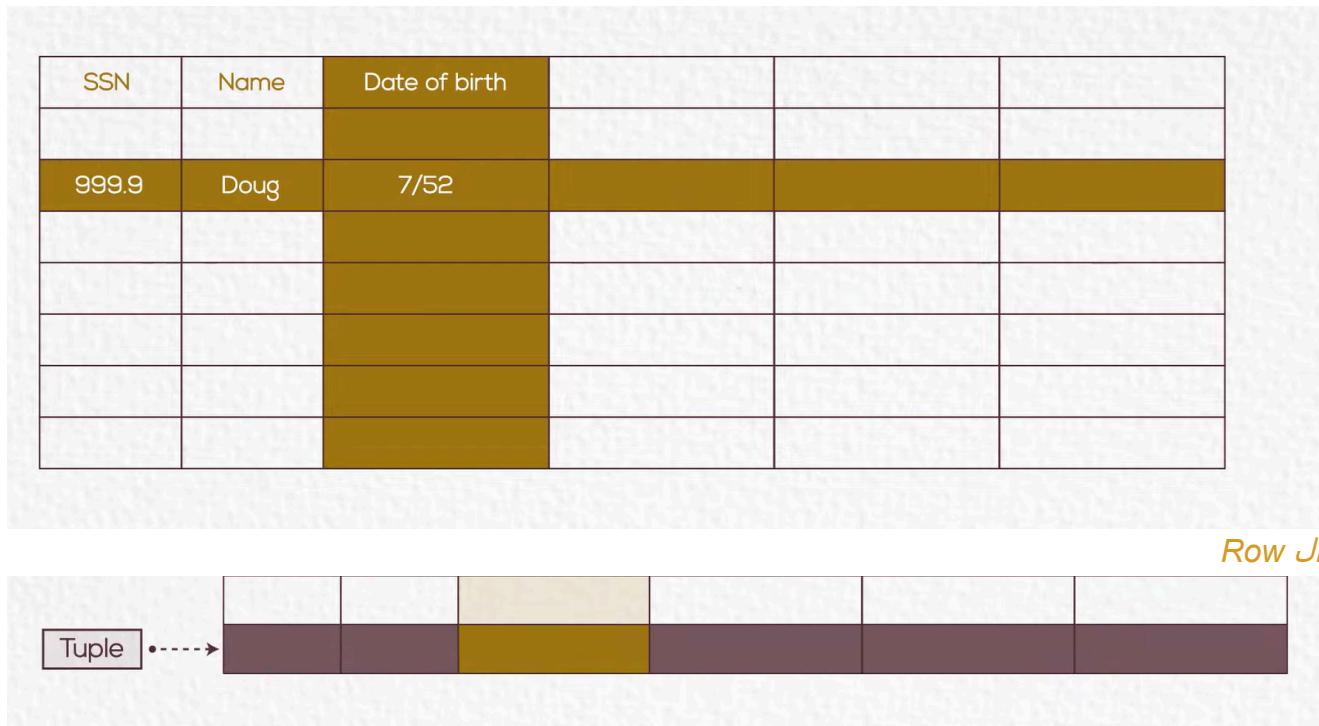


# Chapter 3 - ERD Mapping to Tables

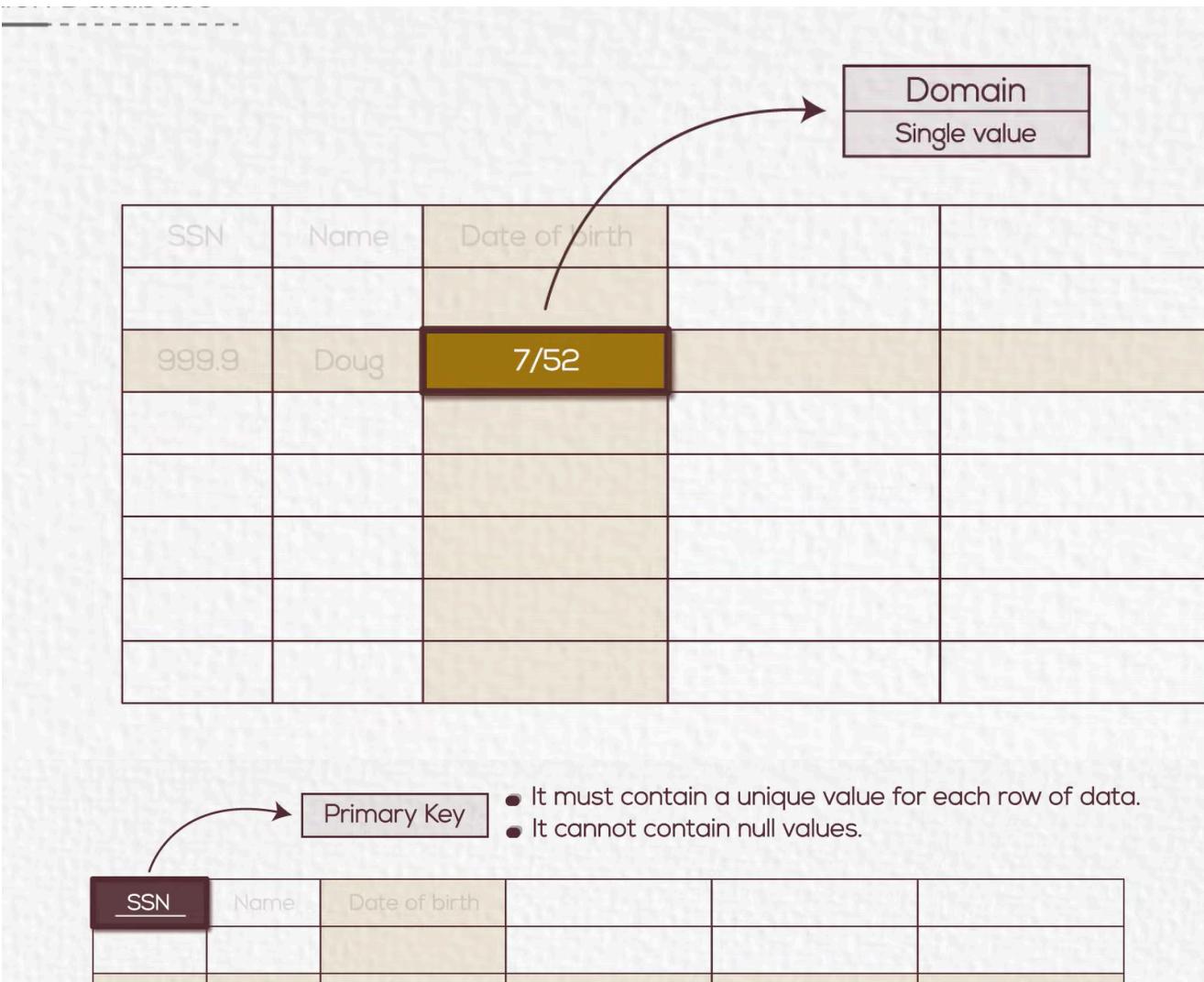
## Mapping Strong and Weak Entities



### Relation Database







## How Can Convert Conceptual Design to Logical Design

- 1- Create Table for Each Entity
- 2- Choose one of key attributes to be the primary key
- 3- Separate Multivalue attribute in new table with Primary key of Entity as Foreign 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

مقدمة

المتحدة لجامعة

### Step 1: Mapping of regular entity types

### Step 2: Mapping of weak entity types

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

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)

## Mapping Relation Types

في حالة العلاقة لما تكون احادية او ثنائية سواء 1:1 او N:N هنأخذ ال primary key بناء على مثلك مثل N  
as foreign key ونحطه

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

Emp - Phone (SSN, Phone)

Dept (DNO, D Name, LOC)

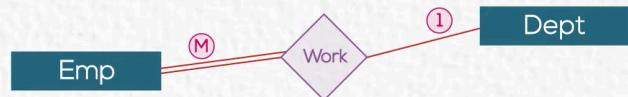
Project (PNO, P Name)

Dependent (SSN, Name, Relation)

Car (Plate\_NO, Model, Color)

Contract (Contract\_ID, Type, Start\_date)

Skill (Skill\_id, Skill\_name)



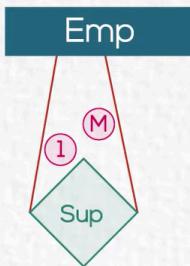
Emp (ID, SSN, Salary, Name, DOB, Street, Zone, DNO, Sup-SSN)

Emp - Phone (SSN, Phone)

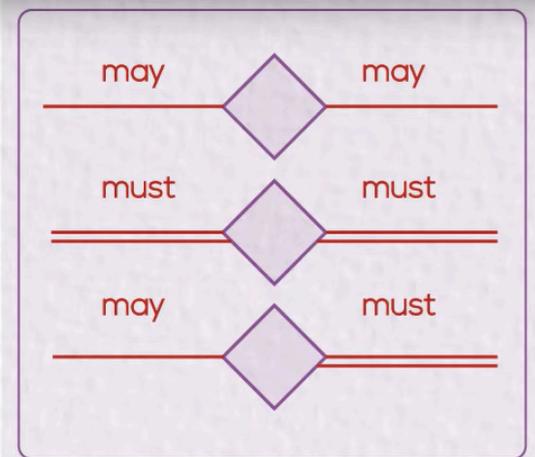
Dept (DNO, D Name, LOC)

Project (PNO, P Name)

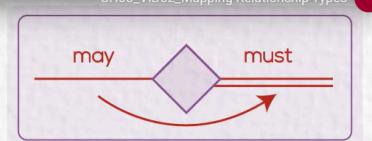
Dependent (SSN, Name, Relation)

**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)

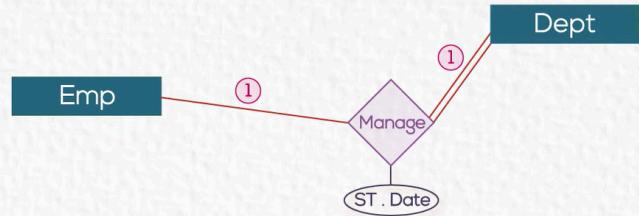
Dept (DNO, D Name, LOC, SSN)

Project (PNO, P Name)

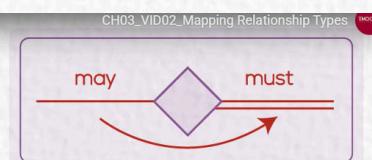
Dependent (SSN, Name, Relation)

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)

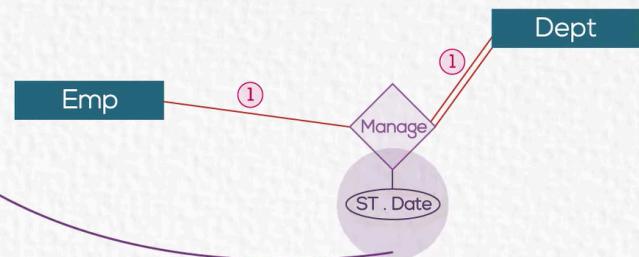
Dept (DNO, D Name, LOC, MGR\_SSN, ST. Date)

Project (PNO, P Name)

Dependent (SSN, Name, Relation)

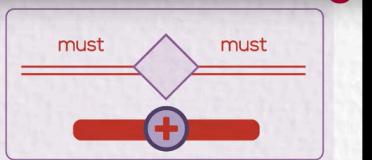
Car (Plate\_NO, Model, Color)

Contract (Contract\_ID, Type, Start\_date)



يتدرج الجداول 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)

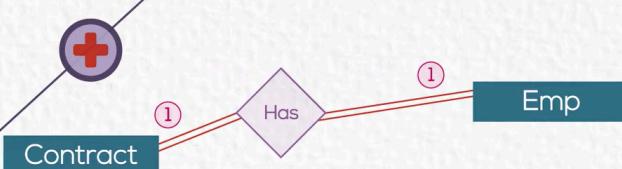
Dependent (SSN, Name, Relation)

Car (Plate\_NO, Model, Color)

Contract (Contract\_ID, Type, Start\_date)

Skill (Skill\_id, Skill\_name)

Work\_On (SSN, PNO, Hours)



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, SI)

```

    erDiagram
        class Emp;
        class Skill;
        class Project;

        Emp ||--o Skill : "Skilled use"
        Emp ||--o Project : "Skilled use"
        Skill ||--o Project : "Skilled use"
    }
  
```

## 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)

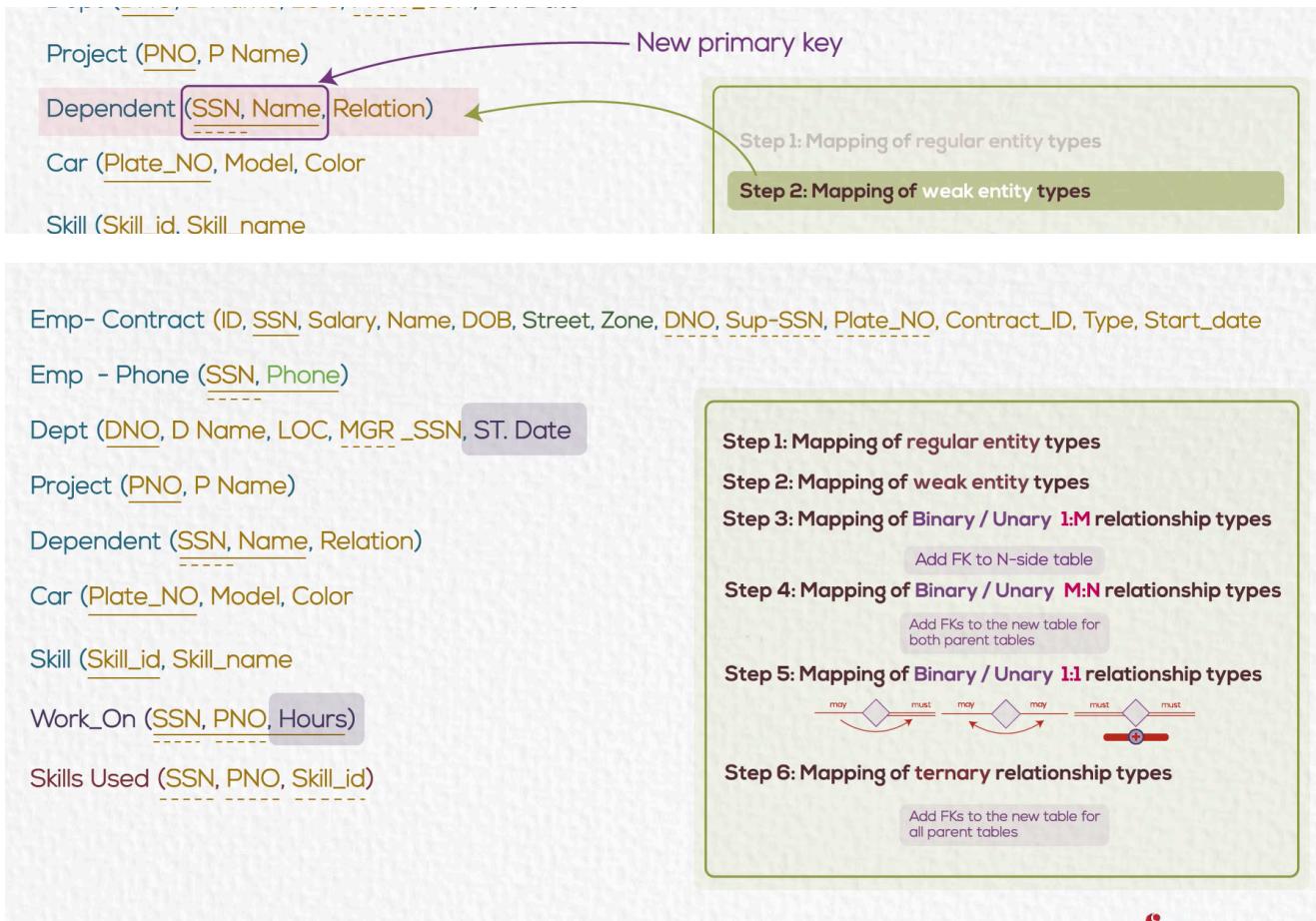
Dept (DNO, D Name, LOC, MGR \_SSN, ST. Date)

Project (PNO, P Name)

Dependent (SSN, Name, Relation)

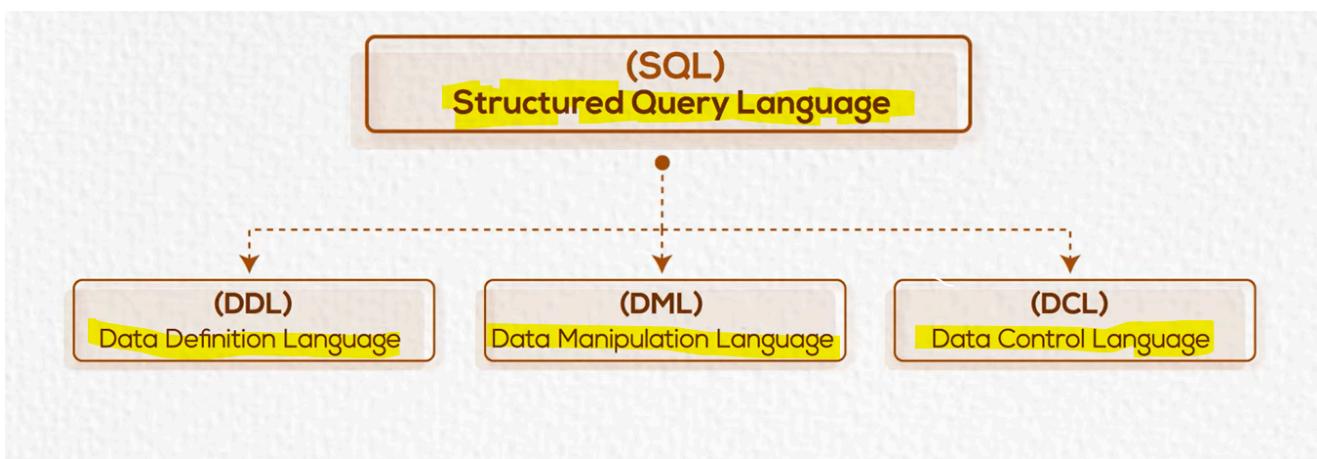
Car (Plate\_NO, Model, Color)

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.

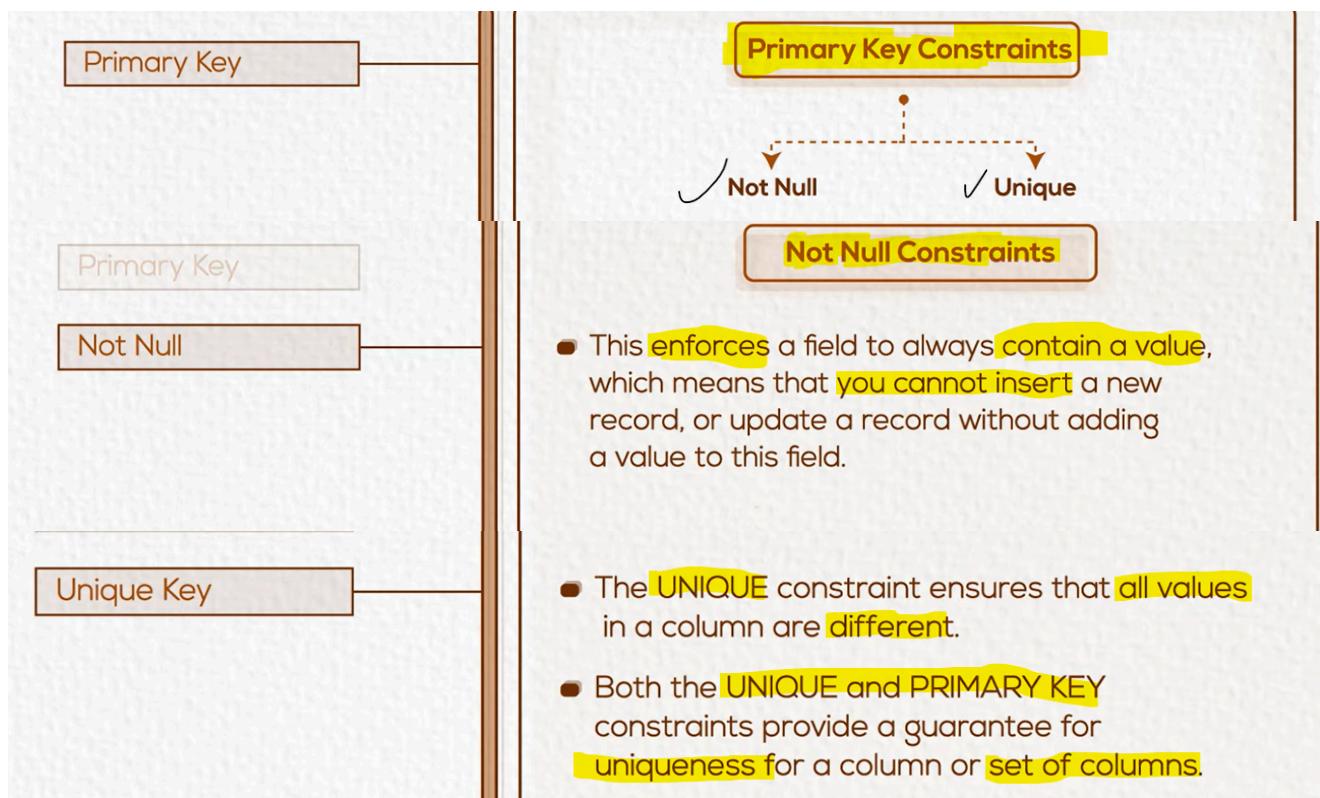
The most commonly used data types are:

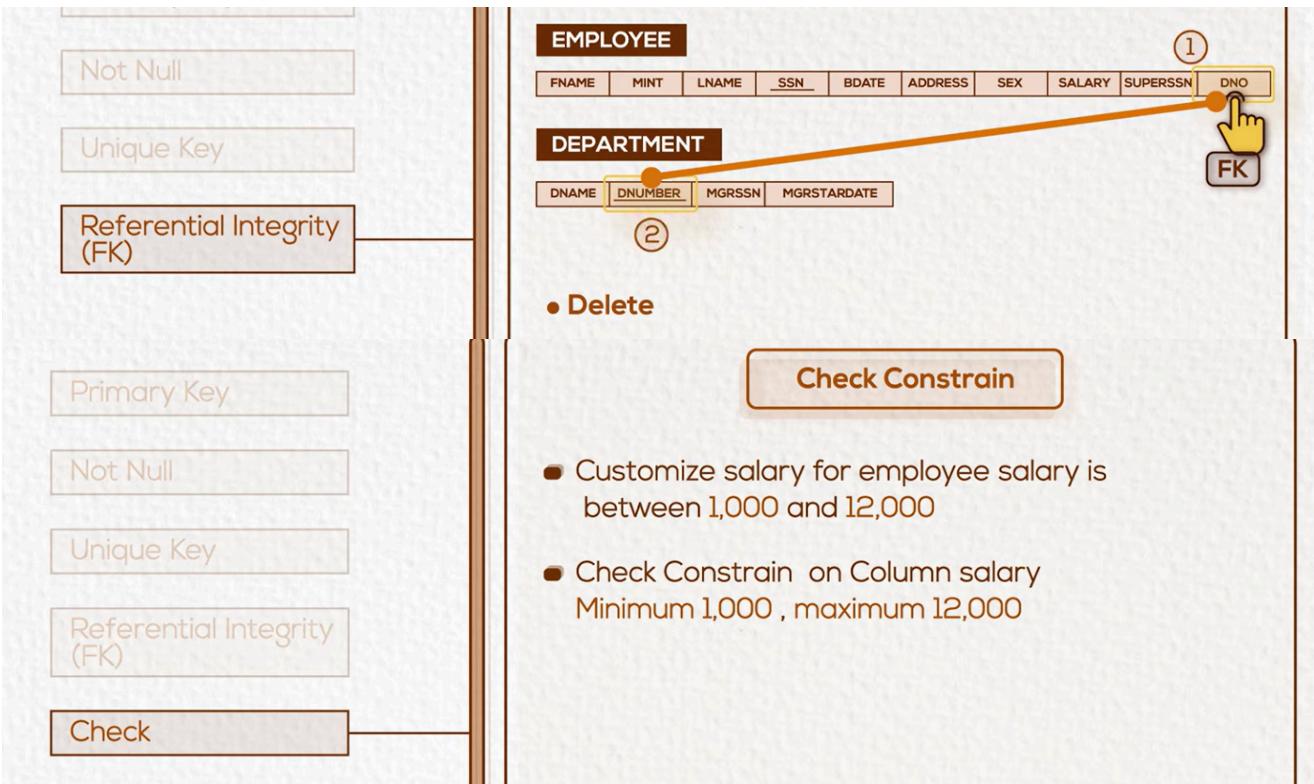
- |                 |                       |
|-----------------|-----------------------|
| ✓ Alphanumeric  | ✓ Variable Characters |
| ✓ Numeric       | ✓ Integer             |
| ✓ Date and Time | ✓ Float data type     |
| ✓ Characters    |                       |

## Database Constraints

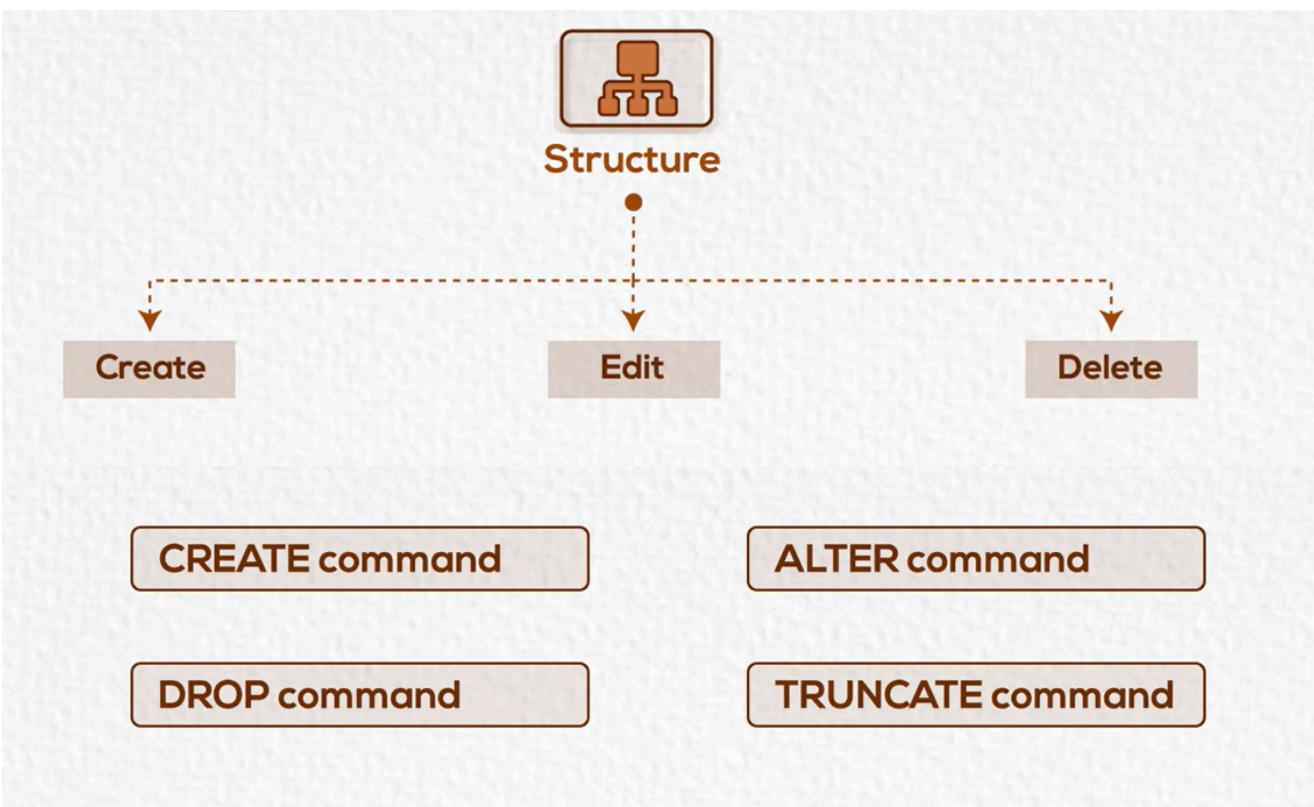
- Restrictions on Database table or object to help **Maintain integrity of data**

### Constraints Types





## SQL - Data Definition Language(DDL)



### Create

**CREATE TABLE Students** *name*

```
(ID NUMBER PRIMARY KEY, First_Name
CHAR(50) NOT NULL, Last_Name CHAR(50),
Address CHAR(50), City CHAR(50), Country
CHAR(25), Birth_Date DATE)
```

- Create students table with ID pk, first name, last name, address, city, country, birth date
- Add column postal code
- Remove column country
- Remove table students

## Alter

**ADD**

**alter table students add postal\_code number** *name*

**REMOVE**

**alter table students drop column country** *name*

## Drop

**drop table students**

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

**GRANT command**

Use a command that gives privilege access to data.

**REVOKE command**

**GRANT command**

Use a command that gives **privilege** access to data.

**REVOKE command**

System privilege

Object privilege

- access to User

## Object privilege

Create privilege

**GRANT**

Remove privilege

**REVOKE**

### Grant

Create privilege

**GRANT**

Command **GRANT SELECT** ON TABLE employees TO **Ahmed;**

**GRANT ALL** ON TABLE department TO **Mary, Ahmed;**

**GRANT SELECT** ON TABLE employees **TO Ahmed WITH**

**GRANT OPTION:**

### Revoke

Remove privilege

**REVOKE**

**REVOKE UPDATE** ON TABLE **department** FROM **Mary;**

**REVOKE ALL** ON TABLE **department** FROM **Mary, Ahmed;**

## Chapter 5 - Data Manipulation Language (DML)

**INSERT Command**

**UPDATE Command**

**DELETE Command**

**SELECT Command**

## Insert

1 Insert into employee (Fname, Lname, SSN, Bdate, Address, Sex, Salary, Superssn, Dno)  
Values ('Ahmed', 'Hassan', 102672, '8/8/1988', '20 el-haram st', 'M', 2000, 112233, 30)

- **Insert Full data of employee (Moheb, Rafaat, 102674, 5/6/1984, 6 Makram ebid st, M, 1600, 112233, 30)**

2 Insert into employee Values ('Moheb', 'Rafaat', 102674, '5/6/1984', '6 Makram Ebid St', 'M', 1600, 112233, 30)

- **Insert the data of employee (Asmaa, Ali, 102661, 18/10/1985, F, 10)**

3 Insert into employee (fname, lname, SSN, bdate, sex, dno)  
Values ('Asmaa', 'Ali', 102661, '18/10/1985', 'F', 10)

- **Update Salary for the employee whose SSN is**

### First Way

```
Insert into employee (Fname, Lname, SSN,  
Address, Sex, Salary, Superssn, Dno)  
Values ('Ahmed', 'Hassan', 102672,  
'8/8/1988', '20 el-haram st', 'M', 2000,  
112233, 30)
```

فُرْتِي

### Second Way

```
Insert into employee Values ('Moheb',  
'Rafaat', 102674, '5/6/1984', '6 Makram Ebid  
St', 'M', 1600, 112233, 30)
```

لُوكْسِفِي

### Update

for one Column

```
update employee  
set salary = 1200  
where SSN = 102661
```

لُوكْسِفِي

*more than one Column*

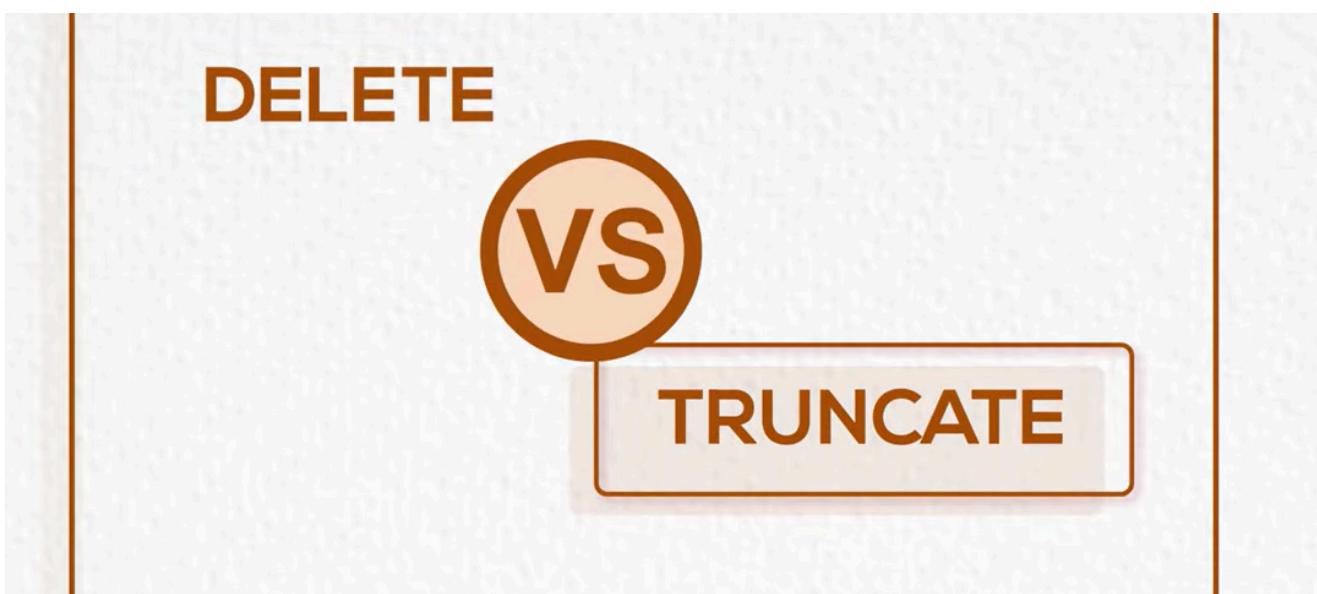
```
update employee  
set salary = 1700 , dno = 10  
where ssn = 102674
```

## Delete

*Record Level*

```
delete from employee  
where ssn = 102672
```

## Difference between Delete & Truncate



## TRUNCATE

- Deletes all data but keeps the structure of the table
- Deletes data unconditionally (doesn't have WHERE clause)
- Can't be rolled back because it's a DDL statement
- De-allocates the physical memory assigned to data

## DELETE

- Deletes all data but keeps the structure of the table (if it doesn't have WHERE clause)
- Can include a WHERE clause to delete data conditionally
- Can be rolled back since it is a DML statement
- Keeps the physical memory assigned to data until a commit or rollback is issued

## Select

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

Table

select \*  
as  
from departments  
where mgrssn = 223344

## Comparison & logical Operators

Query1 Employee Relationships Departments

```
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
```

Query1 Employee Relationships Departments

Select SSN, Fname  
from employee  
where superssn = 321654  
or superssn = 223344

Multi-Row Operator

Query1 Employee Relationships Departments

Select SSN, Fname  
from employee  
where superssn IN (321654, 223344)



Like Operator

Query1 Employee Relationships Departments

```
select *  
from employee  
where fname like '?o*' 
```

## Alias

Query1 Employee Relationships Departments

```
Select fname, salary*0.1 as Bonus  
from employee
```

```
select fname+' '+lname as [Full Name]  
from employee  
where salary*12> 10000
```

## Order by

```
select Fname, SSN
from employee
order by fname desc
```

هيرتب الأول تصاعدي وبعد كدا تنازلي علي حسب كل عمود

```
select *
from employee
order by dno asc, salary desc
```

2(ج)

## Distinct

```
Select distinct Dno
from employee
```

كلينش - كرلا

## Inner Join

لو عاوز تعرض البيانات من اكتر من جدول عندك معلومة من هنا ومعلومة هناك ودابما يكون علاقة بين الـ Primary key , foreign key  
واحنا هنا لازم يكون لكل قيمة على الشمال قيمة موجودة على اليمين

### equal join

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

Join Condition

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

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 Table

2 Join

### Outer , Full Join

لو عاوز في قيمة موجوده على اليمين او الشمال وفي الناحية الثانية مفيش

## Types of Outer

### Left Outer Join

هنا هيرجع كل القيم اللي في الجدول الشمالي

### Right Outer Join

هنا هيرجع كل القيم اللي في الجدول اليميني

### Full Outer Join

بيرجع كل اللي هنا واللي هنار حتى لو مش ليه قيمة

```
Select fname, dname  
from employee e left outer join  
departments d  
on e.dno = d.dno
```

fname	dname
Amr	
Mohamed	
Ahmed	DP1
Kamel	DP1
Hanaa	DP1
Moheb	DP1
Asmaa	DP1
Ahmad	DP1
Noha	DP2
Mariam	DP2
Edward	DP3
Maged	DP3

\* رجع كل اسماء الموظفين  
حتى اللي مش ليه إدارة

## Self Join

```
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

Select \*

from employee

where salary > all (select salary from  
employee where dno ≠ 10)



لما تكون جملة السيلكت بترجع اكتر من قيمة

## Max , Min , Count Functions

بتتجاهل الـ null

```
select max(salary), min(salary)  
from employee
```

## Group by & Having

```
select avg(salary)  
from employee  
group by dno  
having max(salary)>1800
```

if your condition is on Aggregate function  
use having not where

## 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 [ (column1 [, column2] ...) ]

**AS**

**SELECT**

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

بتعامل معاه بعد كدا أكنه جدول عااادي

- **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;
```

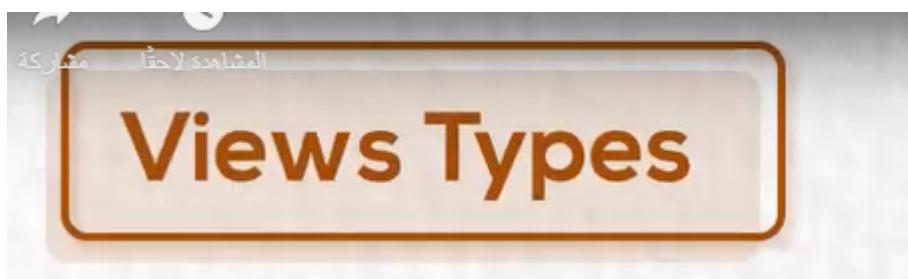
## Removing a View

- 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

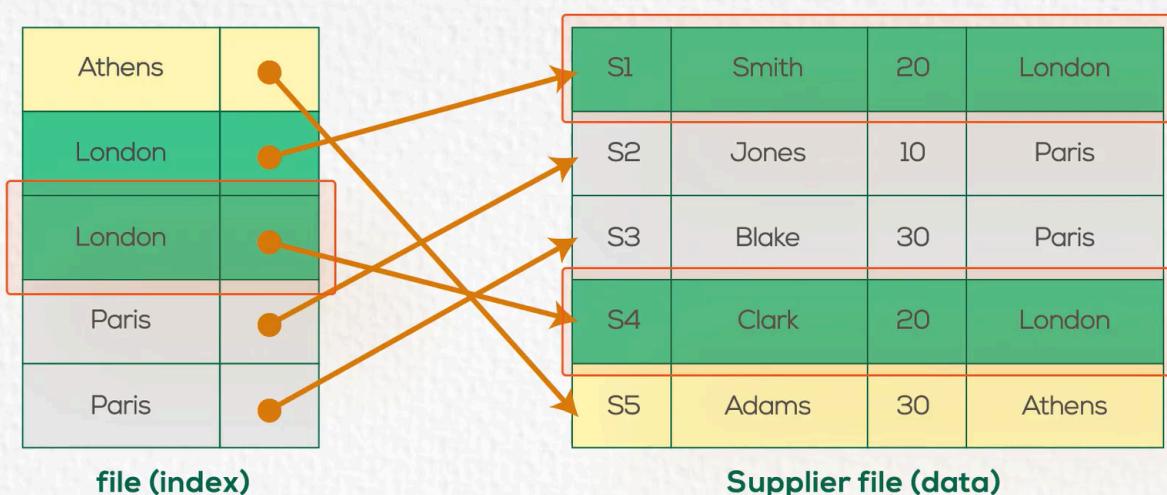
Not sorted

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**.



## Indexes cause overhead on DML (insert, update, delete)

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

### Guidelines

- **Retrieving data** heavily from table.
  - Columns are **used** in **search conditions** and joins.
  - Column contain **large number** of **nulls**.
- Do not create an index when:
- Table is updated **frequently**.

### Create index

#### Creation Indexes

- **CREATE INDEX** index \_name **ON** Table\_name (column\_name);  
**CREATE INDEX** emp\_inx **ON** Employee (Salary);

#### Removing Indexes

- **DROP INDEX** index \_name

## 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 بنسخدمها عشان نقلل التكرار اللي في الداتا وكمان ممكن نعيid تصميم السيستم بتاعنا



## Normalization

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

**EMP-DEPT**

Ename	SSN	Bdate	Address	Dnumber	Dname	Dmgr_SSN
Smith, John B.	123456789	1965-01-09	731 Fondern, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyca A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

- Minimize redundancy
- Insert anomalies
- Update anomalies
- Delete anomalies

**EMP- ProJ**

SSN	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyca A.	ProductX	Bellaire
453453453	2	20.0	English, Joyca A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

Redundancy Problem



## To avoid:

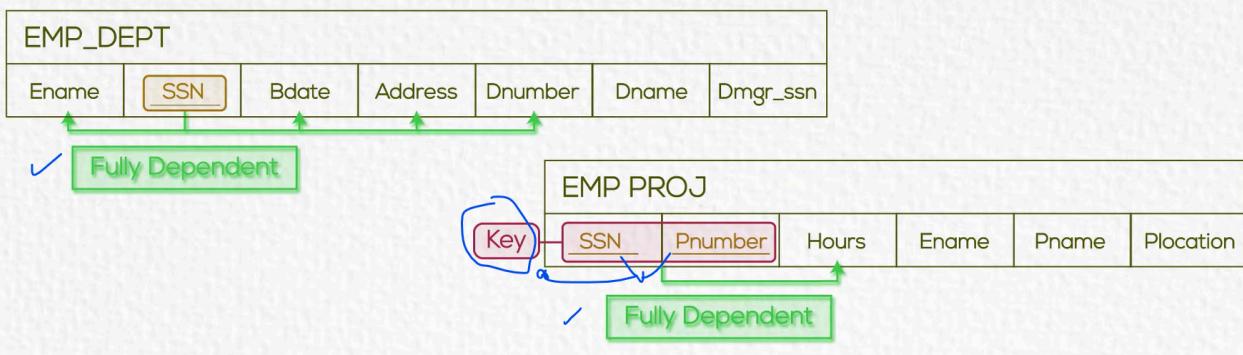
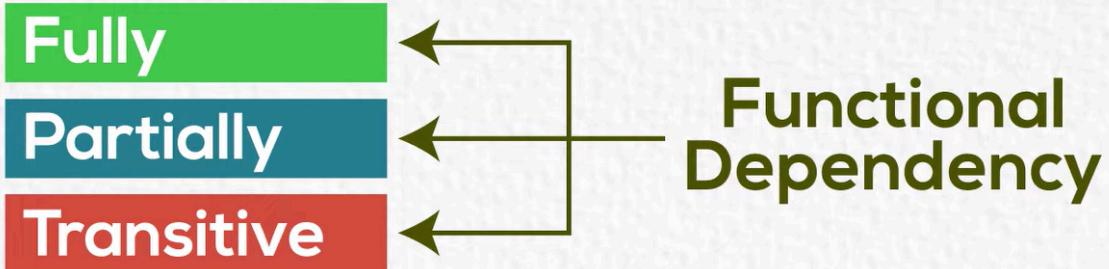
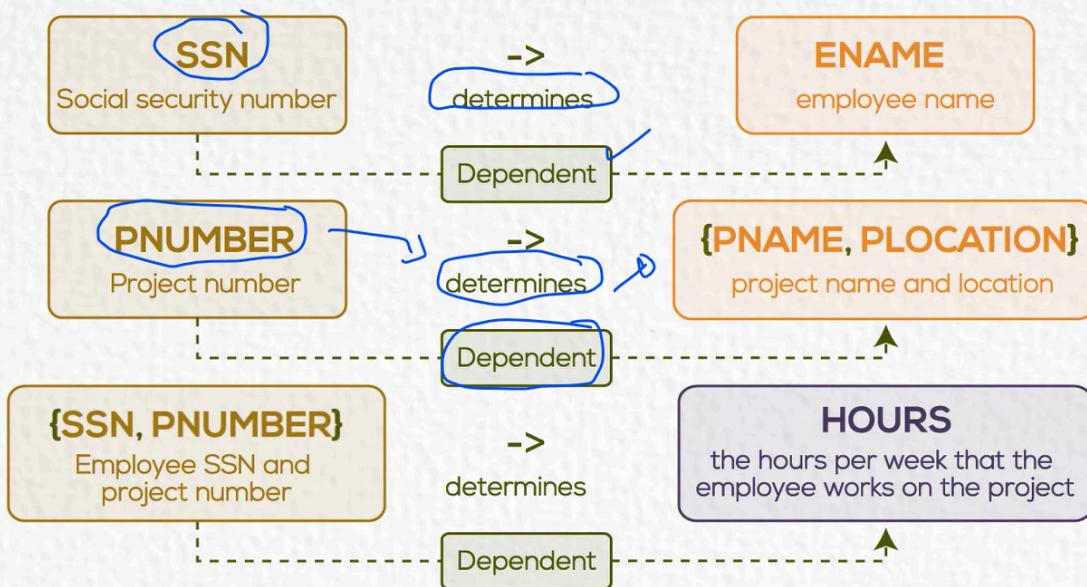
- Redundancy (Duplication of Data)
- Insert anomalies
- Update anomalies
- Delete anomalies
- Frequent Null Values

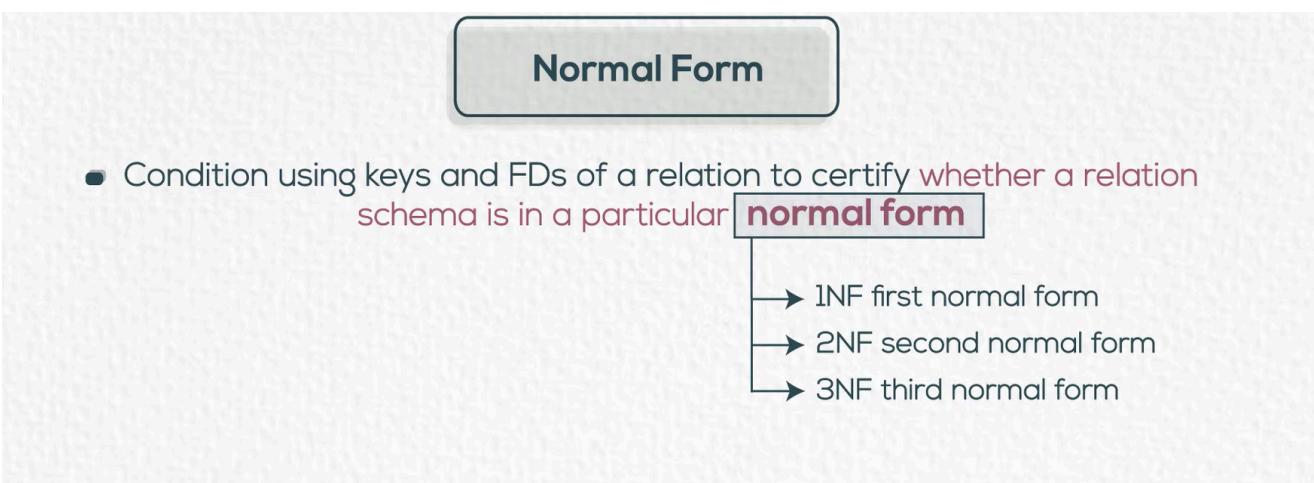
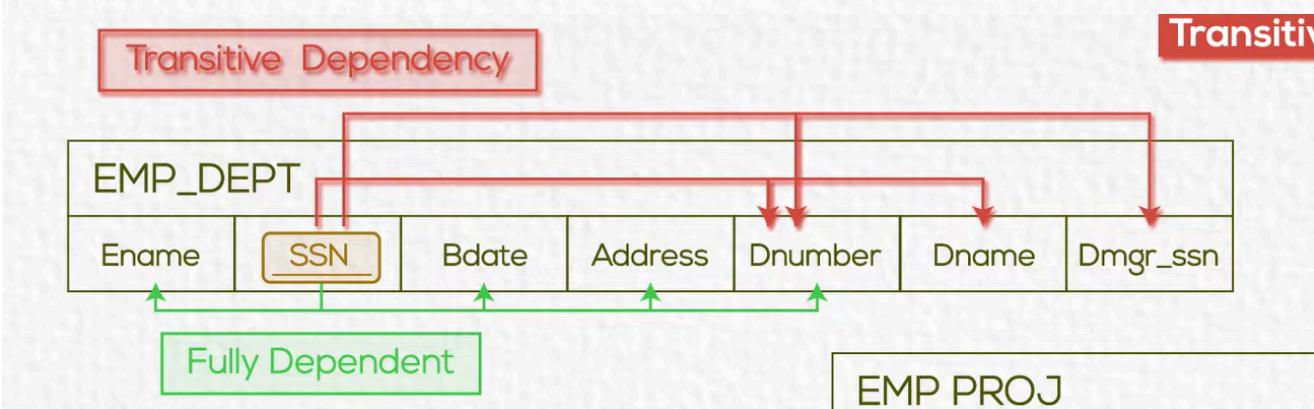
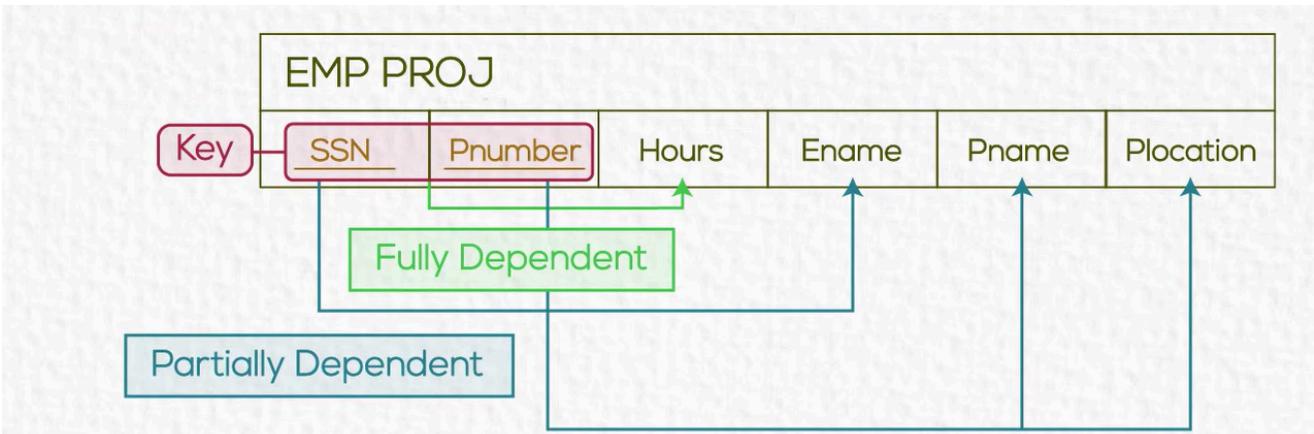


## Functional Dependency

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

value of A uniquely determines the value of B





## First Normal Form

## Multivalued Attribute

## Repeating group

## Composite attribute

Key

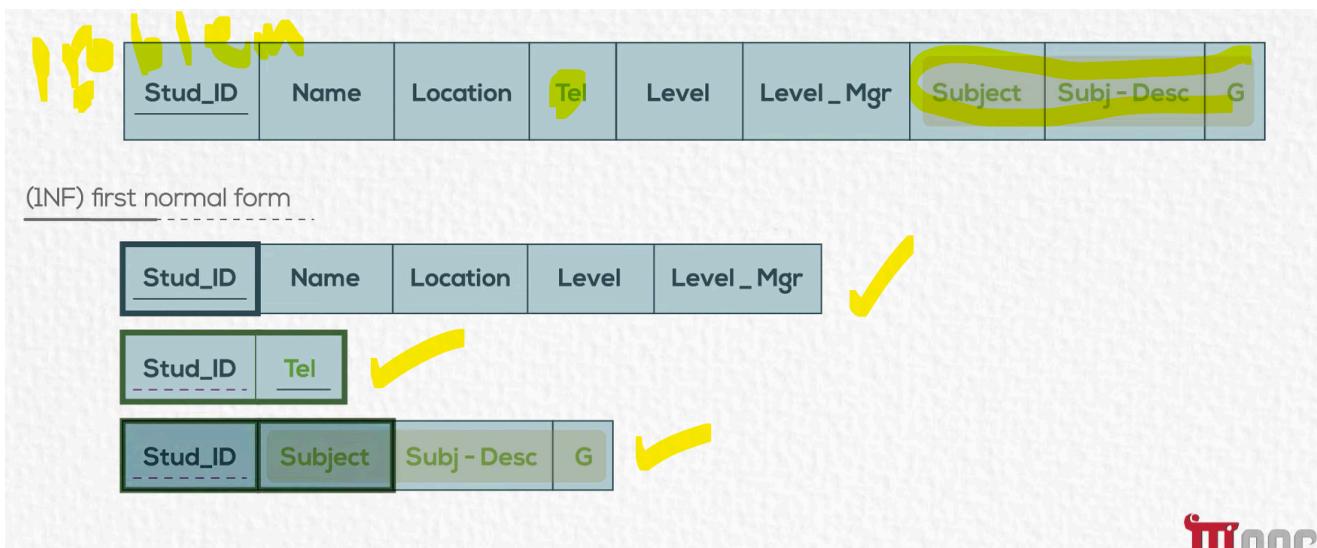
Stud_ID	Name	Location	Tel	Level	Level_Mgr	Subject	Subj - Desc	G
11	Ali	Cairo	010	Primary	Noha M.	DB, CN	Database, Networks	A, B
22	Mai	Giza	011 010	Primary	Noha M.	CN, DB	Networks, Database	B, C
33	Marwa	Giza	010	Secon.	Moh.A.	SW, DB	Software, Database	A, A

Multivalued Attribute

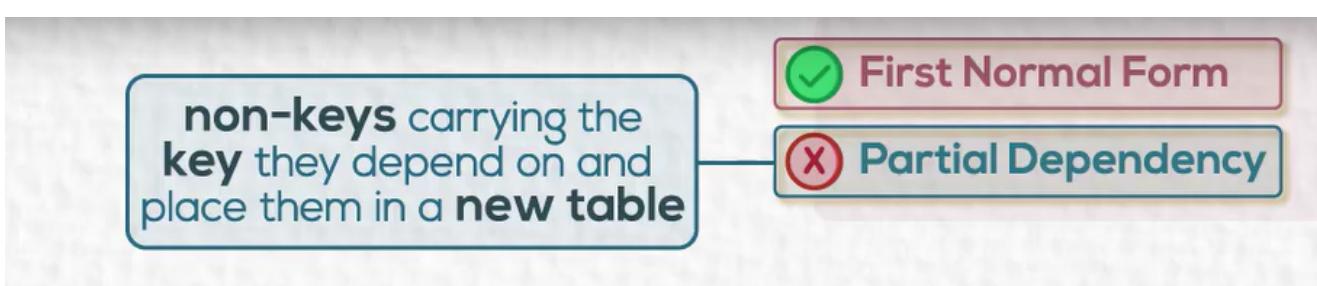
Multivalued Attribute

Repeating group

Related



## Second Normal Form



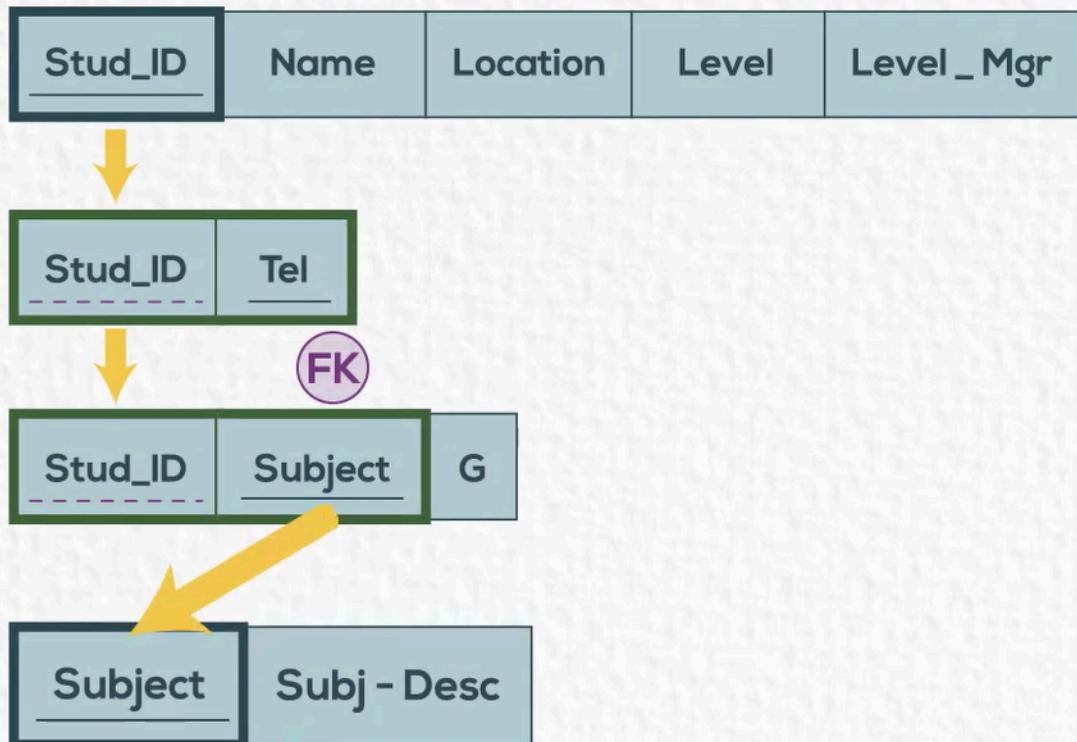
Stud_ID	Subject	Subj - Desc	G
---------	---------	-------------	---



Stud_ID	Subject	Subj - Desc	G
-----	-----	-----	-----

Subject	Subj - Desc
-----	-----

↙ ↘ ↙ ↘



## Third Normal Form

CH07\_VID05\_Third Normal Form

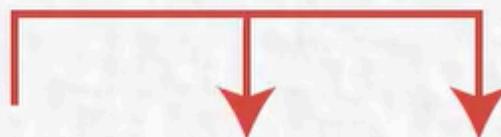
TMOC



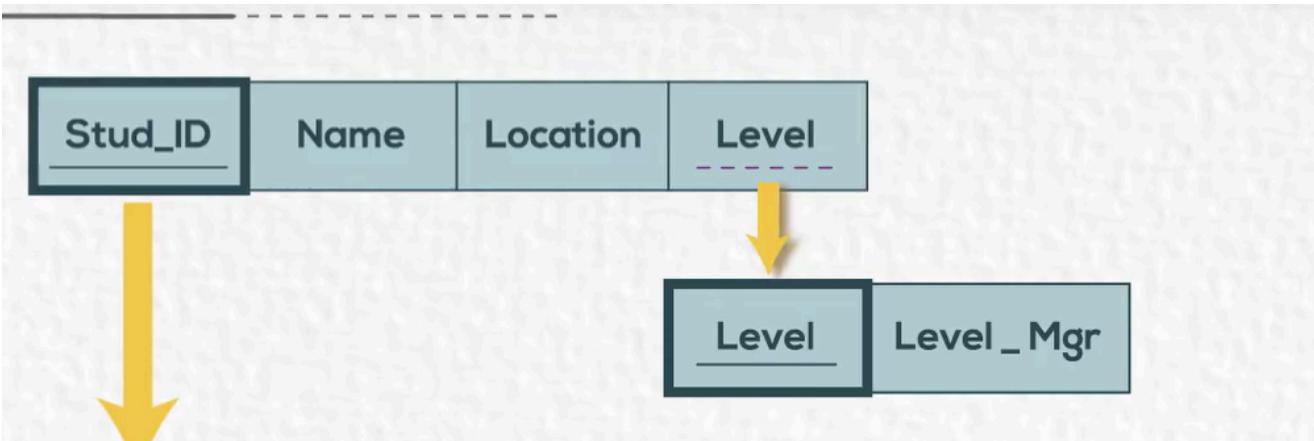
**Second Normal Form**



**Transitive Dependency**



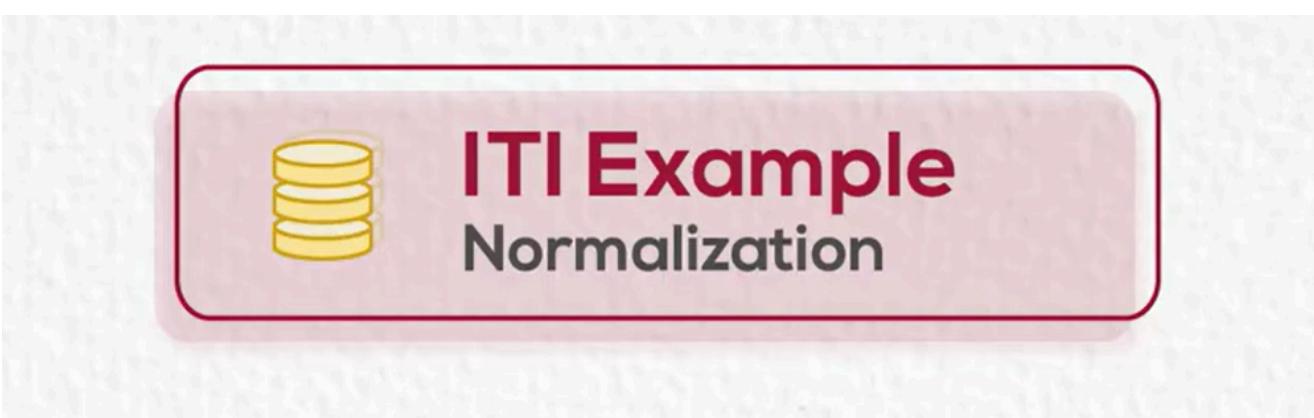
	Dependent	Dependent							
Stud_ID	Name	Location	Tel	Level	Level_Mgr	Subject	Subj_Desc	G	



## Summary

■ 1NF first normal form	<ul style="list-style-type: none"> <li>✗ Multivalued Attribute → place it in a new table carrying the <u>PK</u> as a <u>FK</u></li> <li>✗ Repeating group → subparts each in a column when necessary</li> <li>✗ Composite attribute → subparts each in a column when necessary</li> </ul>
■ 2NF second normal form	<ul style="list-style-type: none"> <li>✓ First Normal Form</li> <li>✗ Partial Dependency → non-keys carrying the key they depend on and place them in a new table</li> </ul>
■ 3NF third normal form	<ul style="list-style-type: none"> <li>✓ Second Normal Form</li> <li>✗ Transitive Dependency → non-key attributes carrying the non-key attribute they depend on and place them in a new table</li> </ul>

## 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

O NF

(Stud \_No, Stud \_Name, Address (Street, City),

Tel \_No, F-Code, Faculty, Major,

Dept \_Name, Dept \_Desc, AD \_Grade, Comment)

CH07\_VID06\_Summary - example

**ITI Students Sheet**

<b>Student Number:</b> ITI205-40	<b>F-code:</b> ENG		
<b>Student Name:</b> Hassan Ali Ahmed	<b>Faculty:</b> Engineering		
<b>Address(Street, City):</b> 12 Haram St, Giza	<b>Major:</b> Computer		
<b>Tel no/Mobile:</b> 33868420 / 01111111253			
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

O NF

(Stud \_No, Stud \_Name, Address (Street, City),

① Composite attribute

Tel \_No, F-Code, Faculty, Major,

② Multivalued attribute

Dept \_Name, Dept \_Desc, AD \_Grade, Comment)

③ Repeating group

**ITI Students Sheet**

<b>Student Number:</b> ITI205-40	<b>F-code:</b> ENG		
<b>Student Name:</b> Hassan Ali Ahmed	<b>Faculty:</b> Engineering		
<b>Address(Street, City):</b> 12 Haram St, Giza	<b>Major:</b> Computer		
<b>Tel no/Mobile:</b> 33868420 / 01111111253			
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

• First Normal Form rule

- ④ Multivalued attribute
- ⑤ Repeating group
- ⑥ Composite attribute

## 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