DATA BASE-Course15

What is Database? (20:55)

What is Database?

A database is an organized collection of data so that it can be easily accessed. To manage these databases, Database Management Systems (DBMS) are used.

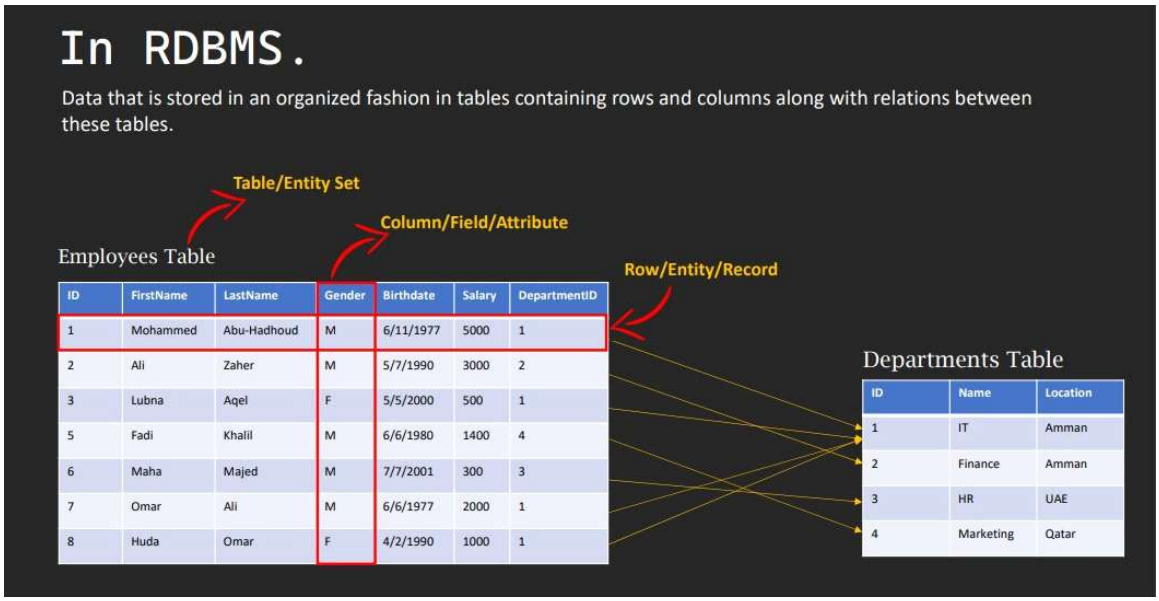
Types of DBMS:

In general, there are two common types of databases:

- Non-Relational (DBMS): File System, XML..etc.
- Relational (RDBMS): enhanced version of DBMS but with relations, examples SQLServer, Oracle, MySQL ..etc.

Employees File

ID	FirstName	LastName	Gender	Birthdate	Salary	DepID	DeptName	DeptLocation
1	Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
2	Ali	Zaher	M	5/7/1990	3000	2	Finance	Amman
3	Lubna	Aqel	F	5/5/2000	500	1	IT	Amman
5	Fadi	Khalil	M	6/6/1980	1400	4	Marketing	Qatar
6	Maha	Majed	M	7/7/2001	300	3	HR	UAE
7	Omar	Ali	M	6/6/1977	2000	1	IT	Amman
8	Huda	Omar	F	4/2/1990	1000	1	IT	Amman



DBMS and RDBMS are NOT the same.

DBMS stands for Database Management Systems.

DBMS has NOT relations between Data.

RDBMS has relations between data.

RDBMS stands for Relational Database Management Systems .

DBMS has two types: None Relational Database and Relational Database.

File System, XML are samples of DBMS.

SQL Server , Oracle , MySQL are examples of RDBMS.

Dealing with RDBMS is much easier than
dealing with DBMS.

In RDBMS: Data that is stored in an organized fashion in tables containing rows and columns along with relations between these tables.

Column/Field/Attribute are the same.
Row/Entity/Record are the same.
Table/Entity Set are the same.

Relationships are represented using references to data from other tables.

What is NULL? (8:33)

What is Null?

Employees Table

ID	FirstName	LastName	Gender	Birthdate	Salary	DepartmentID
1	Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1
2	Ali	Zaher	M	Null	Null	2
3	Lubna	Aqel	F	5/5/2000	500	1
5	Fadi	Khalil	M	6/6/1980	1400	4
6	Maha	Majed	M	7/7/2001	300	3
7	Omar	Ali	M	6/6/1977	2000	1
8	Huda	Omar	F	4/2/1990	1000	1

Departments Table

ID	Name	Location
1	IT	Amman
2	Finance	Amman
3	HR	UAE
4	Marketing	Qatar

What is Null

- In a database, "NULL" is a special marker used to indicate that a data value does not exist in the database. It represents a missing or unknown value in a table column.
- When a field or column in a table has a NULL value, it means that the field has no value at all, and it's different from having an empty or zero value. NULL is not the same as a blank space or a zero, and it's a distinct value in the database.
- NULL can be used in several ways, such as indicating missing data, representing optional fields, or as a placeholder for values that are not yet known. However, it's essential to use NULL values carefully because they can affect the results of queries and calculations in unexpected ways.

Summary

- In summary, NULL is a special value in a database that represents the absence of a value in a table column.
- It's used to indicate missing or unknown data and should be used carefully to avoid unexpected results in queries and calculations.

NULL is a special value in a database that represents the absence of a value in a table column.

NULL is used to indicate missing or unknown data and should be used carefully to avoid unexpected results in queries and calculations.

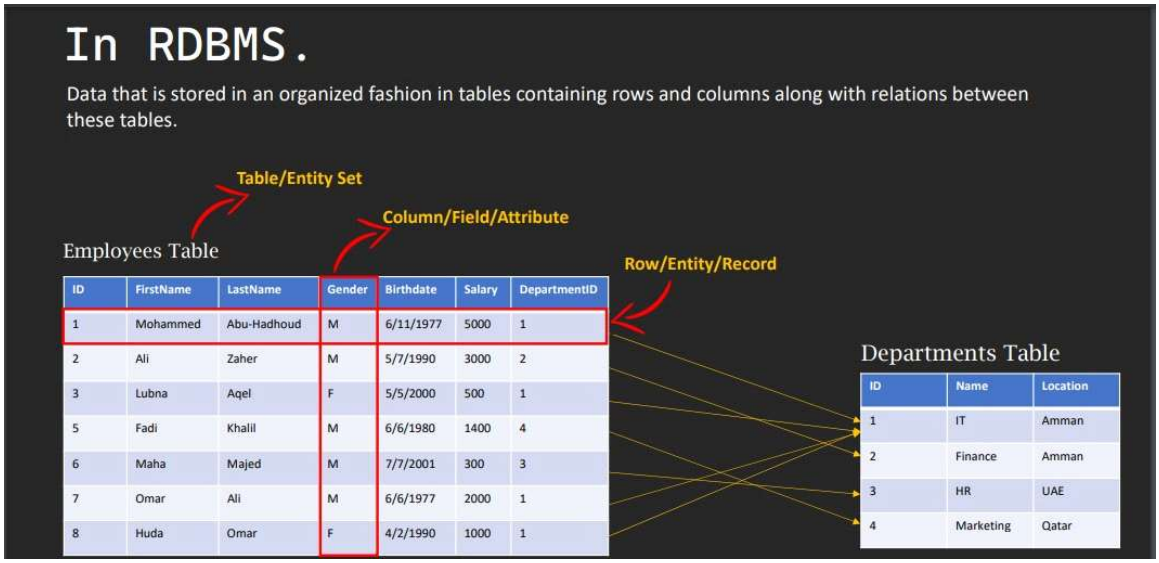
When a salary field in a table has a NULL value it means its value is missing or unknown.

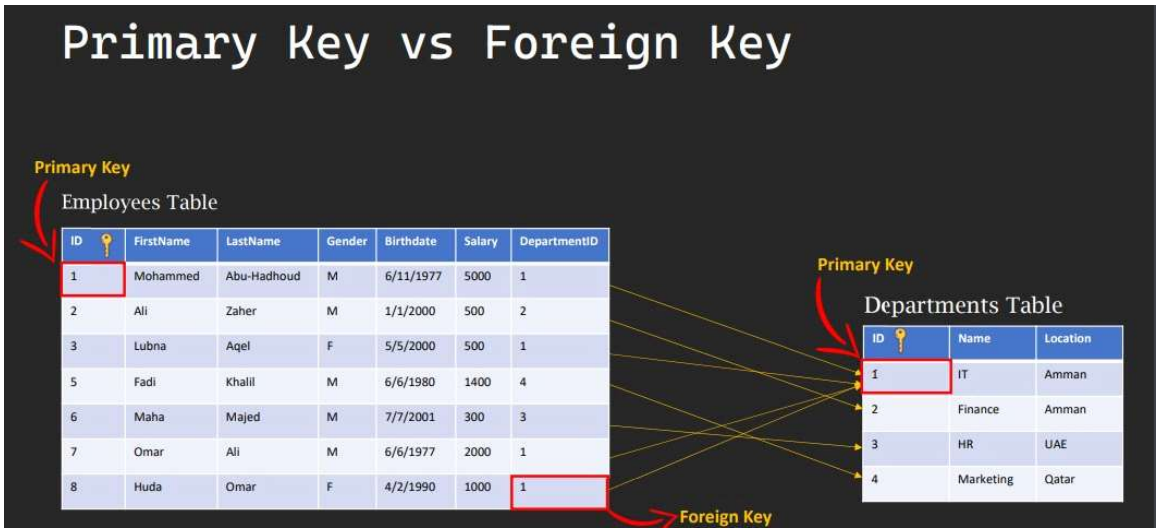
When a field in a table has a NULL value it means
its optional.

NULL is not the same as a blank space or a zero.

When a NumberOfChildren field in Employees table has a NULL value this means that we don't know yet that value, it's missing or unknown to us.

Primary Key vs Foreign Key / Referential Integrity (24:36)





Primary Key

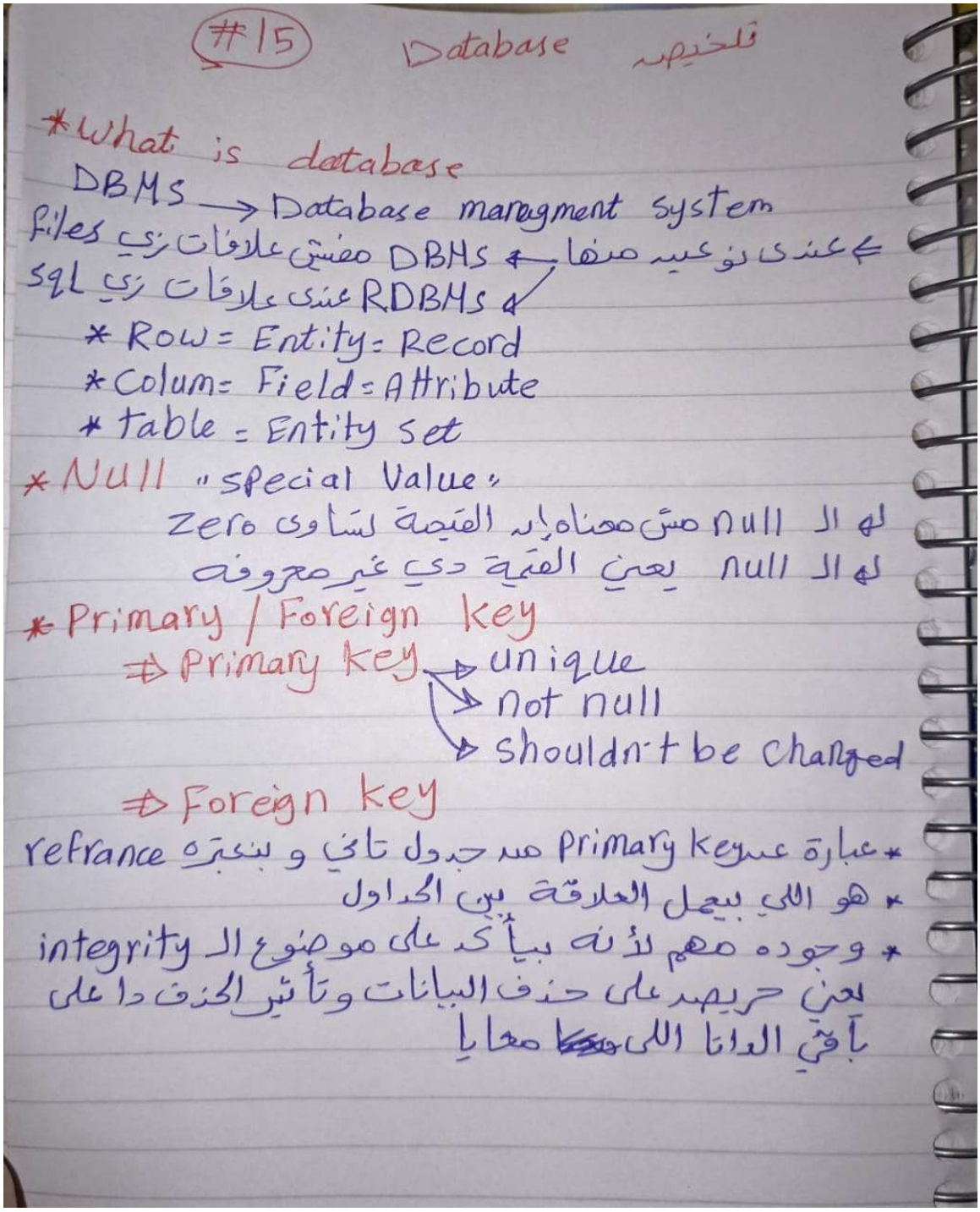
- A primary key is a column or set of columns in a relational database table that uniquely identifies each row or record in the table.
- It is a unique identifier for each record and serves as a reference point for other tables that have a relationship with the table in question.
- Each table in a relational database must have a primary key, and it should be a non-null value that is unique and stable over time.
- Primary Key should not be changed.

Foreign Key

- A foreign key, on the other hand, is a column or set of columns in a table that refers to the primary key of another table.
- It establishes a relationship between two tables, allowing data to be shared and linked between them.
- The foreign key ensures that referential integrity is maintained by ensuring that any value in the foreign key column must exist in the primary key column of the related table.

Summary

- In summary, a primary key uniquely identifies a record in a table, while a foreign key establishes a relationship between two tables by referencing the primary key of another table.



In summary, a primary key uniquely identifies a record in a table, while a foreign key establishes a relationship

between two tables by referencing the primary key of another table.

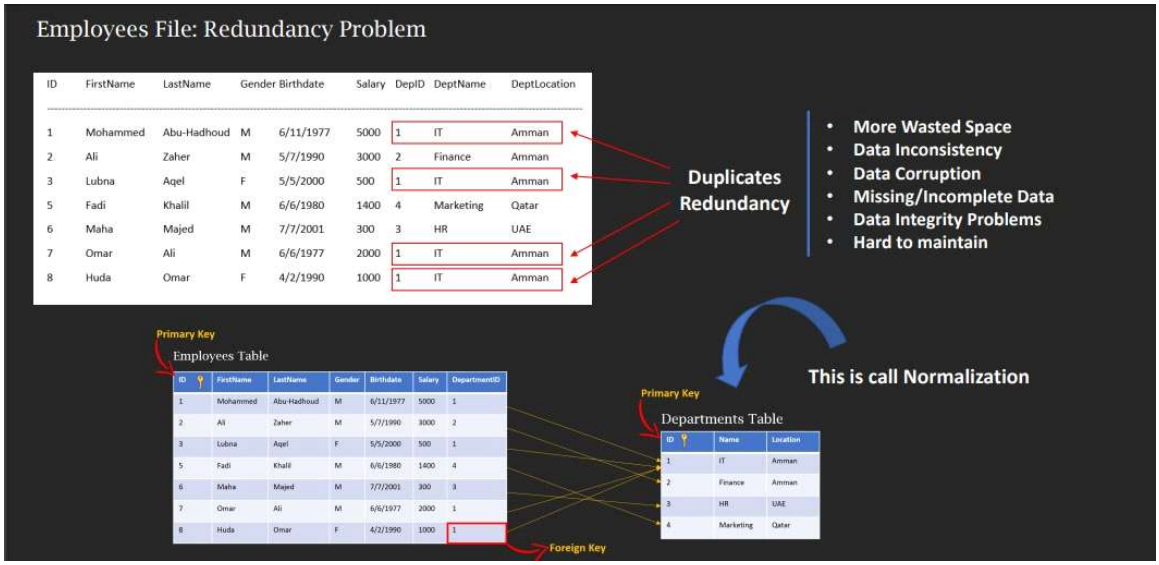
Primary Key can be more than one filed.

It's better to use numbers as primary keys because they are fast in search.

Primary Key should be Unique, Not NULL , and should not be changed.

Value in the foreign key column must exist in the primary key column of the related table.

What is Redundancy? and why it's a problem? (19:22)

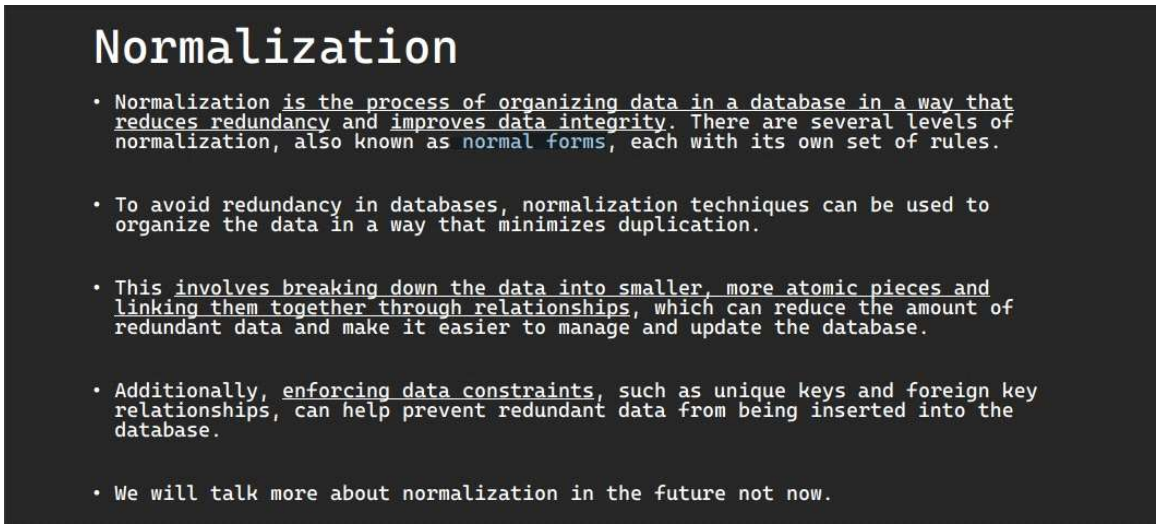


Redundancy

- Redundancy refers to the presence of duplicated data within the database. Redundancy can occur in different ways, such as storing the same information multiple times, or storing information that can be derived from other data in the database.

- While redundancy can sometimes be useful, it can also cause problems. For example, redundant data takes up additional storage space and can make it more difficult to maintain consistency within the database. If one copy of the data is updated, the other copies may become outdated, leading to inconsistencies and errors.

- To avoid redundancy in databases, normalization techniques can be used to organize the data in a way that minimizes duplication.



Redundancy refers to the presence of duplicated data within the database.

Redundancy can occur in different ways, such as storing the same information multiple times, or storing information that can be derived from other data in the database.

Redundancy can sometimes be useful.

Redundant data takes up additional storage space.

Redundancy can make it more difficult to maintain consistency within the database. If one copy of the data is

updated, the other copies may become outdated, leading to inconsistencies and errors.

To avoid redundancy in databases, normalization techniques can be used to organize the data in a way that minimizes duplication.

Normalization is the process of organizing data in a database in a way that reduces redundancy.

Normalization improves data integrity.

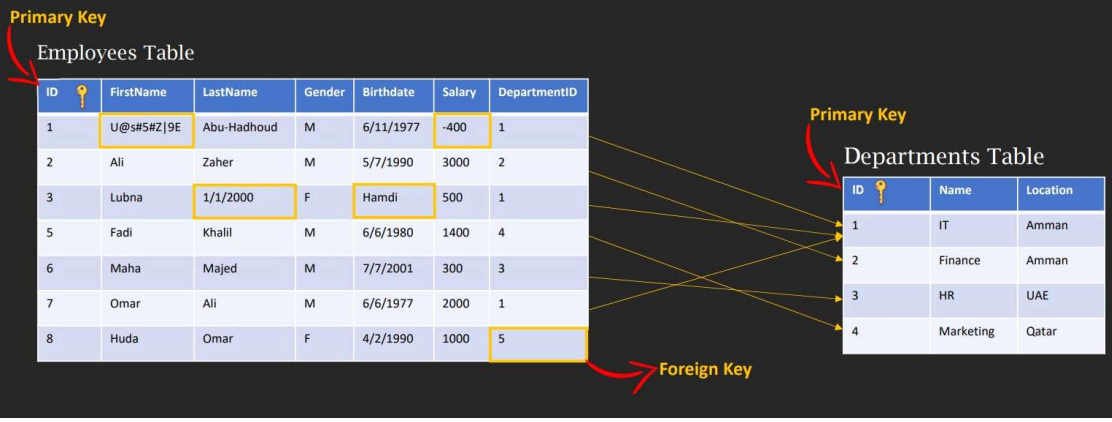
Normalization involves breaking down the data into smaller, more atomic pieces and linking them together through relationships, which can reduce the amount of redundant data and make it easier to manage and update the database.

Enforcing data constraints, such as unique keys and foreign key relationships, can help prevent redundant data from being inserted into the database.

Constraints are Restrictions/Rules on Data.

What is Data Integrity? and Why it's Important and Critical? (21:38)

Data Integrity



Data Integrity

- Data integrity refers to the accuracy, consistency, and reliability of data over its entire life cycle, from creation to deletion. In other words, it refers to the assurance that data is complete, accurate, and trustworthy.
- There are several factors that can impact data integrity, including human error, hardware or software failure, security breaches, and data transfer errors.
- To maintain data integrity, it is important to establish appropriate policies and procedures, and to implement appropriate technologies, such as encryption, backups, and access controls.

There are different types of data integrity that organizations need to consider:

1. Entity integrity: This ensures that each row or record in a table is unique and can be uniquely identified. This is typically achieved through the use of primary keys.

2. Referential integrity: This ensures that relationships between tables are maintained and that there are no orphaned records. This is typically achieved through the use of foreign keys.

3. Domain integrity: This ensures that data is within acceptable ranges or values. For example, a date field should only contain valid dates, and a numeric field should only contain valid numbers.

4. Business integrity: This ensures that data meets business rules and requirements. For example, a bank might have rules around minimum and maximum account balances, or a hospital might have rules around patient data confidentiality.

- Maintaining data integrity is critical for organizations that rely on accurate and trustworthy data to make informed decisions. Without data integrity, organizations risk making decisions based on incomplete, inaccurate, or unreliable data, which can lead to poor outcomes, financial losses, and damage to reputation.

- To maintain data integrity we use Constraints, we will explain them in the next lesson ☺.

Data integrity refers to the accuracy, consistency, and reliability of data over its entire life cycle, from creation to deletion.

Data Integrity refers to the assurance that data is complete, accurate, and trustworthy.

There are several factors that can impact data integrity, including human error, hardware or software failure,

security breaches, and data transfer errors.

There are different types of data integrity that organizations need to consider: Entity integrity, Referential integrity, Domain integrity, and Business integrity.

Entity integrity: This ensures that each row or record in a table is unique and can be uniquely identified. This is typically achieved through the use of primary keys.

Referential integrity: This ensures that relationships between tables are maintained and that there are no orphaned records. This is typically achieved through the use of foreign keys.

Domain integrity: This ensures that data is within acceptable ranges or values. For example, a date field should only contain valid dates, and a numeric field should only contain valid numbers.

Business integrity: This ensures that data meets business rules and requirements. For example, a bank might have rules around minimum and maximum account balances, or a hospital might have rules around patient data confidentiality.

What is Constraint? and Why it's Important? (13:47)

Constraints

• In the context of databases, constraints are rules or conditions that are

applied to the data to ensure its integrity and consistency. Constraints

can be applied to individual (فردي) columns or to entire tables, and they are used

to enforce various(متنوع) rules and restrictions (قيود) on the data.

• By using constraints, you can help ensure that your data is accurate (دقيقة) ,

consistent (ثابتة) , and easy to manage.

- In the context of databases, constraints are rules or conditions that are applied to the data to ensure its integrity and consistency. Constraints can be applied to individual columns or to entire tables, and they are used to enforce various rules and restrictions on the data.

- By using constraints, you can help ensure that your data is accurate, consistent, and easy to manage.

Here are some common types of constraints used in databases:

1. Primary Key Constraint: This constraint ensures that a column or a set of columns uniquely identifies each row in a table. This constraint helps to enforce data integrity and ensure that there are no duplicate rows in the table.

2. Foreign Key Constraint: This constraint establishes a relationship between two tables based on a key field. The foreign key constraint ensures that data in one table matches data in another table, and it helps to maintain referential integrity in the database.

3. Unique Constraint: This constraint ensures that the data in a column or set of columns is unique across all rows in the table. This constraint helps to enforce data integrity and prevent duplicate values from being inserted into the table.

Here are some common types of constraints used in databases:

4. Not Null Constraint: This constraint ensures that a column or set of columns cannot contain null (empty) values. This constraint helps to ensure that the data is complete and accurate, and it can help prevent errors in queries and calculations.

5. Check Constraint: This constraint ensures that the data in a column or set of columns meets a specified condition. This constraint helps to enforce data integrity and prevent invalid data from being inserted into the table.

Interview Question?

What is the difference between Primary Key Constraint and Unique Constraint?

Primary Key is Unique but it does not allow NULL ☺ while Unique allows NULL.

In the context of databases, constraints are rules or conditions that are applied to the data to ensure its integrity and consistency. Constraints can be applied to individual columns or to entire tables, and they are used to enforce various rules and restrictions on the data.

By using constraints, you can help ensure that your data is accurate, consistent, and easy to manage.

The constraint that ensures that a column or a set of columns uniquely identifies each row in a table. This constraint helps to enforce data integrity and ensure that there are no duplicate rows in the table is?

Primary Key Constraint

Which constraint ensures that the data in a column or set of columns meets a specified condition. This constraint helps to enforce data integrity and prevent invalid data from being inserted into the table?

Check Constraint

Which constraint establishes a relationship between two tables based on a key field. The foreign key constraint

ensures that data in one table matches data in another table, and it helps to maintain referential integrity in the database?

Foreign Key Constraint

Which constraint ensures that a column or set of columns cannot contain null (empty) values. This constraint helps to ensure that the data is complete and accurate, and it can help prevent errors in queries and calculations?

Not Null Constraint

Which constraint ensures that the data in a column or set of columns is unique across all rows in the table. and allows Null as well, This constraint helps to enforce data integrity and prevent duplicate values from being inserted into the table?

Unique Constraint

Which constraint(s) ensures that the data in a column or set of columns is unique across all rows in the table. This constraint helps to enforce data integrity and prevent duplicate values from being inserted into the table.

Primary Key Constraint

Unique Constraint

Primary Key is Unique Constraint but does not allow NULL while Unique Constraint allows NULL.

Both Primary Key Constraint and Unique Constraint prevent duplicates

To achieve Entity integrity we use:

Primary Key Constraints

To achieve Referential integrity we use:

Foreign Key Constraints

To achieve Domain integrity we use:

Check Constraints

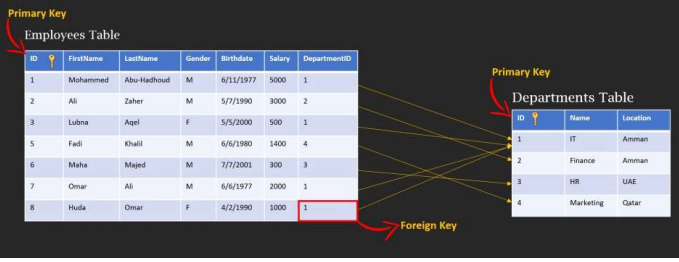
What is SQL? (17:51)

Employees File:

ID	FirstName	LastName	Gender	Birthdate	Salary	DepID	DeptName	DeptLocation
1	Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
2	Ali	Zaher	M	5/7/1990	3000	2	Finance	Amman
3	Lubna	Aqel	F	5/5/2000	500	1	IT	Amman
5	Fadi	Khalil	M	6/6/1980	1400	4	Marketing	Qatar
6	Maha	Majed	M	7/7/2001	300	3	HR	UAE
7	Omar	Ali	M	6/6/1977	2000	1	IT	Amman
8	Huda	Omar	F	4/2/1990	1000	1	IT	Amman

To Get All Female Employees

- You have to write code and loops.
- Slow process
- Slow performance



- You Use SQL Query ☺
- Simple and fast.

```
Select * from Employees
Where Gender = 'F'
```

What is SQL

- SQL stands for Structured Query Language
- Pronounced as "S-Q-L" or sometimes as "See-Quel".
- SQL is used to communicate with a database.
- SQL lets you access and manipulate databases
- Database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc.

What Can You Do With SQL?

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views

Examples Of SQL Statements:

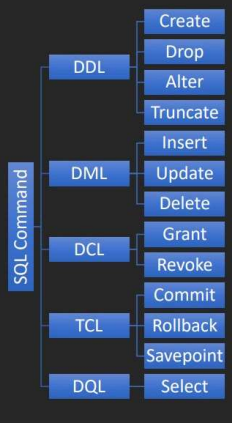
- SELECT * FROM Employees WHERE Salary < 1000;
- SELECT FirstName , LastName FROM Employees WHERE Salary < 1000 and Gender='M';
- SELECT * FROM Employees WHERE Salary between 500 and 1000;
- Select Count(*) from Employees;
- Select Sum(Salary) from Employees;
- Select Avg(Salary) from Employees;
- Delete from Employees where ID=10;
- Update Employees set FirstName = 'Amjad' where ID = 10;

Examples:

- CREATE DATABASE MyDatabase;
- DROP DATABASE MyDatabase;
- CREATE TABLE Employees (
  ID int,
  FirstName varchar(255),
  LastName varchar(255),
  Address varchar(255),
  City varchar(255)
  );

Types of SQL Statements:

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Transaction Control Language (TCL)
- Data Query Language (DQL)



SQL stands for **Structured Query Language**.

SQL is Pronounced as “S-Q-L” or sometimes as “See-Quel”.

SQL is used to communicate with a database.

Database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc.

SQL can execute queries against a database.

SQL can retrieve data from a database.

SQL can insert and update records in a database.

SQL can delete records from a database.

SQL can create new databases, Tables, Views ..etc
in the database

SQL can set permissions on tables, procedures, and views.

5 Types of SQL Statements :

Data Definition Language (DDL)

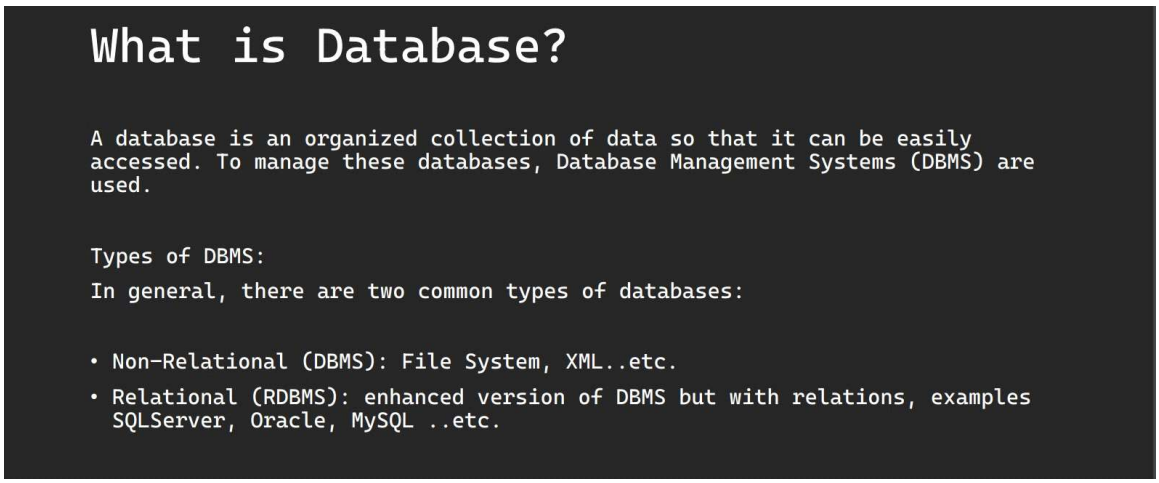
Data Manipulation Language (DML)

Data Control Language (DCL)

Transaction Control Language (TCL)

Data Query Language (DQL)

DBMs vs RDBMS Summary (8:50)

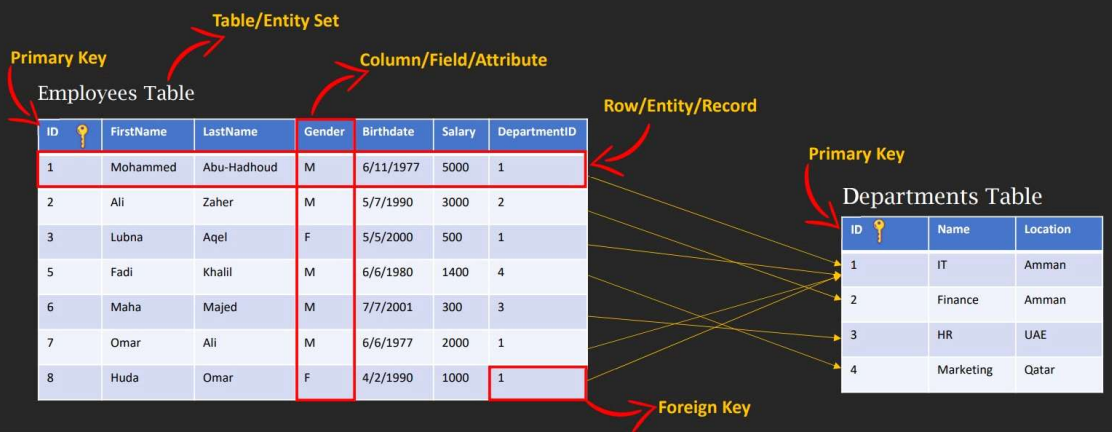


Employees File

ID	FirstName	LastName	Gender	Birthdate	Salary	DepID	DeptName	DeptLocation
1	Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1	IT	Amman
2	Ali	Zaher	M	5/7/1990	3000	2	Finance	Amman
3	Lubna	Aqel	F	5/5/2000	500	1	IT	Amman
5	Fadi	Khalil	M	6/6/1980	1400	4	Marketing	Qatar
6	Maha	Majed	M	7/7/2001	300	3	HR	UAE
7	Omar	Ali	M	6/6/1977	2000	1	IT	Amman
8	Huda	Omar	F	4/2/1990	1000	1	IT	Amman

In RDBMS.

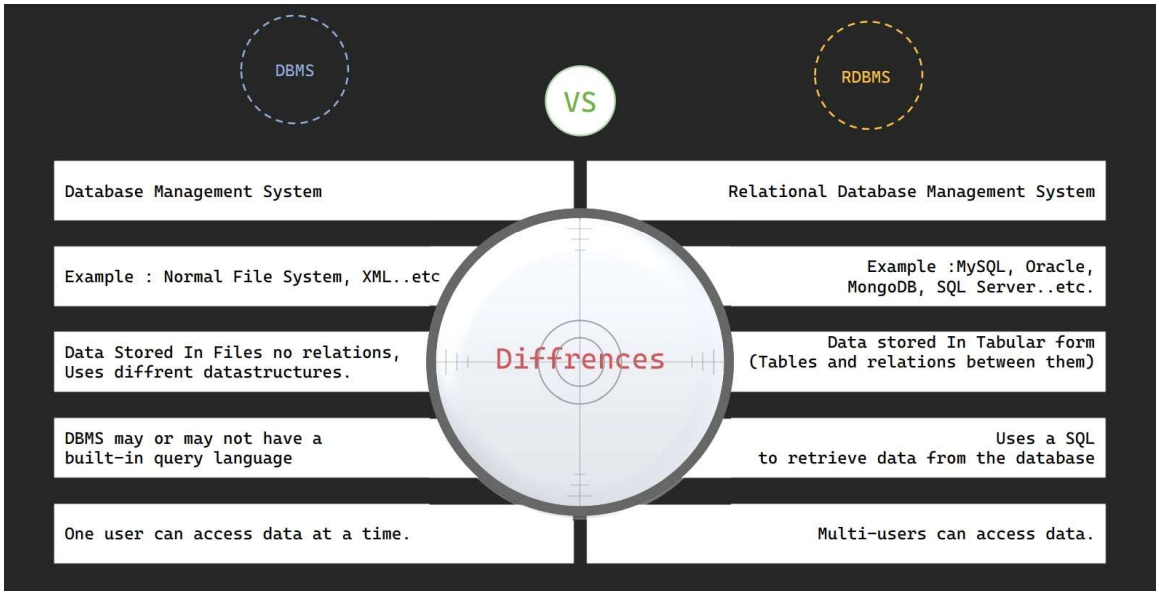
Data that is stored in an organized fashion in tables containing rows and columns along with relations between these tables.



ID	FirstName	LastName	Gender	Birthdate	Salary	DepartmentID
1	Mohammed	Abu-Hadhoud	M	6/11/1977	5000	1
2	Ali	Zaher	M	5/7/1990	3000	2
3	Lubna	Aqel	F	5/5/2000	500	1
5	Fadi	Khalil	M	6/6/1980	1400	4
6	Maha	Majed	M	7/7/2001	300	3
7	Omar	Ali	M	6/6/1977	2000	1
8	Huda	Omar	F	4/2/1990	1000	1

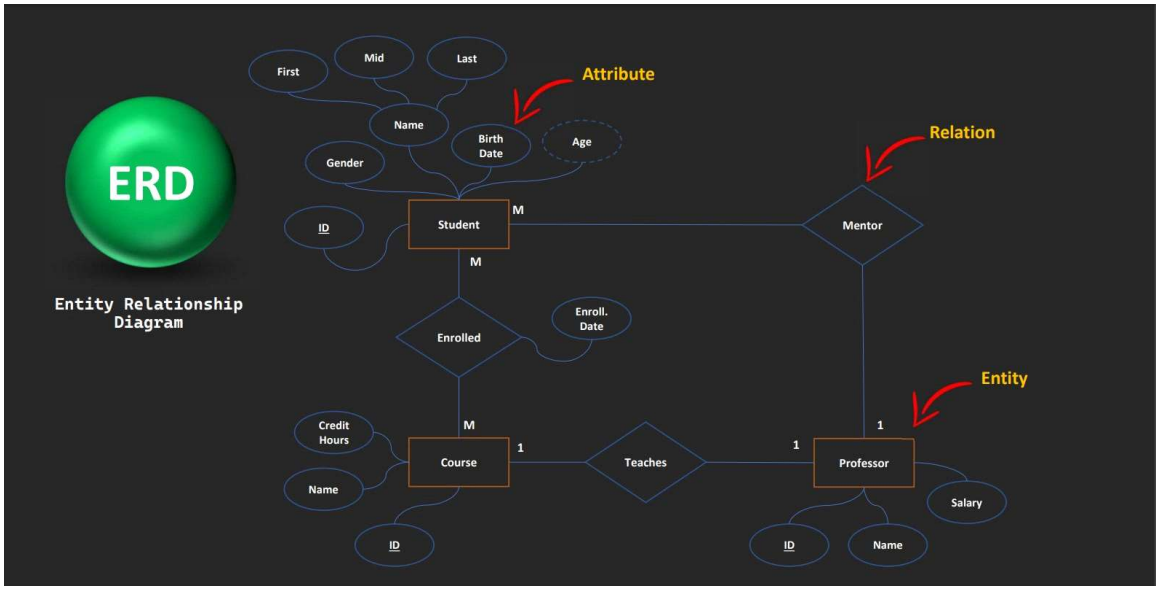
ID	Name	Location
1	IT	Amman
2	Finance	Amman
3	HR	UAE
4	Marketing	Qatar

Differences between DBMS vs RDBMS



Database Design: Conceptual Design

What is ERD? and Why? (20:22)

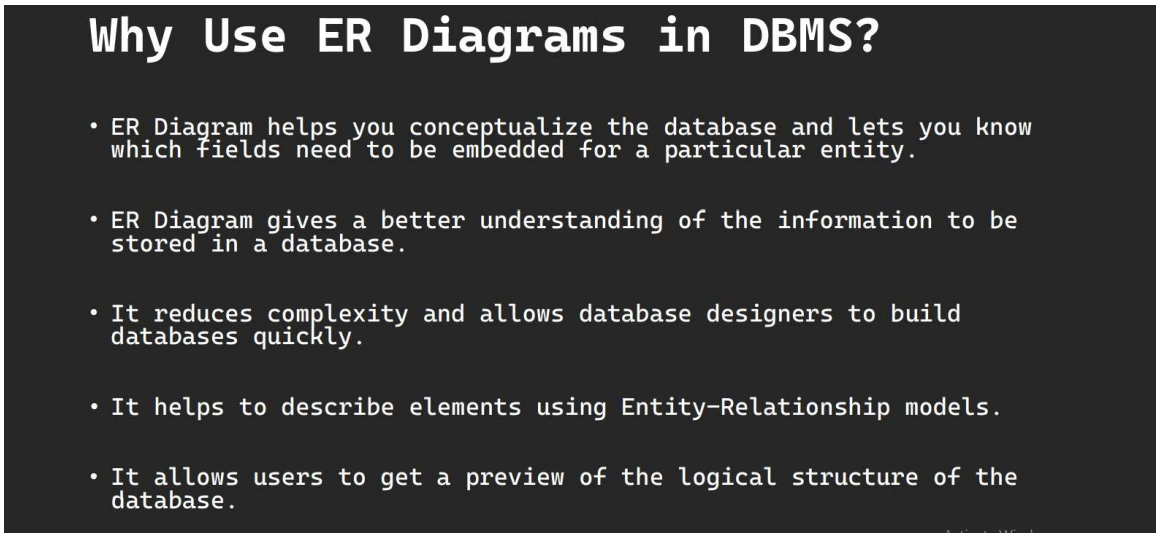


What is ERD?

- An Entity Relationship Diagram (ER Diagram) pictorially explains the relationship between entities to be stored in a database.
- Fundamentally, the ER Diagram is a structural design of the database.
- It acts as a framework created with specialized symbols for the purpose of defining the relationship between the database entities.
- ER diagram is created based on three principal components: entities, attributes, and relationships.

What is an ER Model?

- An Entity-Relationship Model represents the structure of the database with the help of a diagram.
- ER Modelling is a systematic process to design a database as it would require you to analyze all data requirements before implementing your database.



An Entity Relationship Diagram (ER Diagram) pictorially explains the relationship between entities to be stored in a database.

ER Diagram is a **structural design** of the database.

ER Diagram is a **conceptual design** of the database.

ER diagram is created based on three principal

components: entities, attributes, and relationships.

An Entity-Relationship Model represents the structure of the database with the help of a diagram.

ER Modelling is a systematic process to design a database as it would require you to analyze all data requirements before implementing your database.

The cost of updating ERD is much cheaper than the cost of updating Database after you build it.

ER Diagram helps you conceptualize the database and lets you know which fields need to be embedded for a particular entity.

ER Diagram gives a better understanding of the information to be stored in a database.

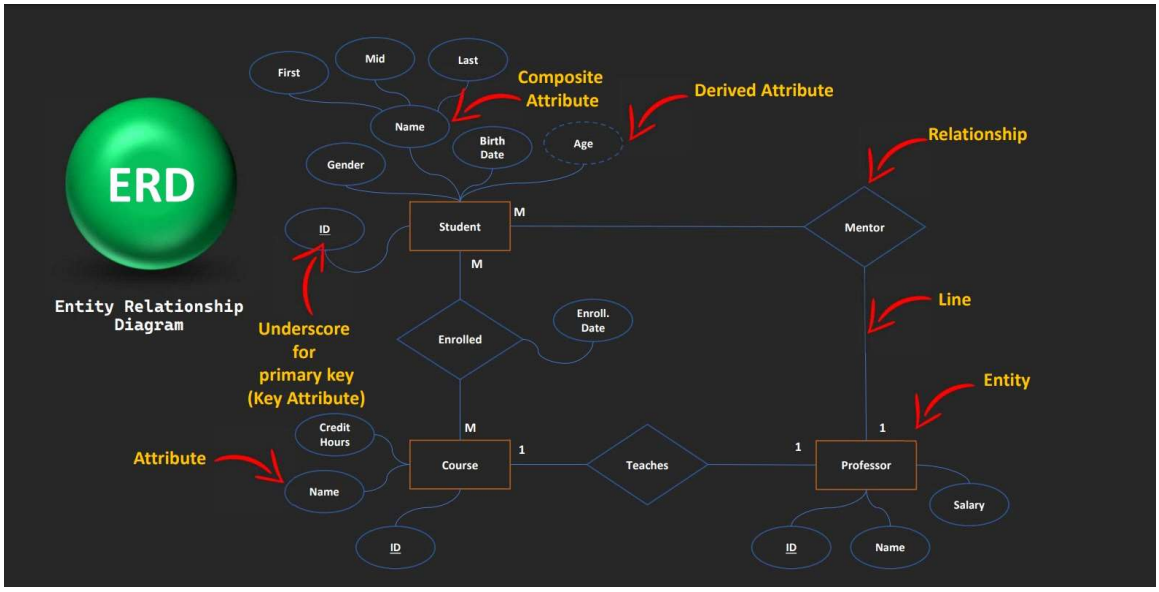
ERD **reduces complexity** and allows database designers to **build databases quickly**.

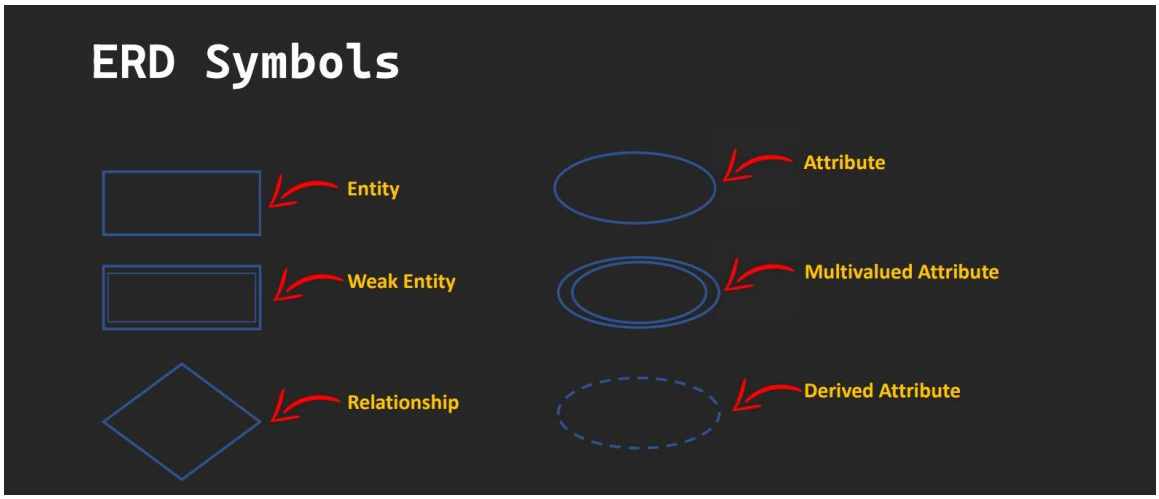
ER Diagram in RDBMS is widely **used** to **describe** the **conceptual design of databases**.

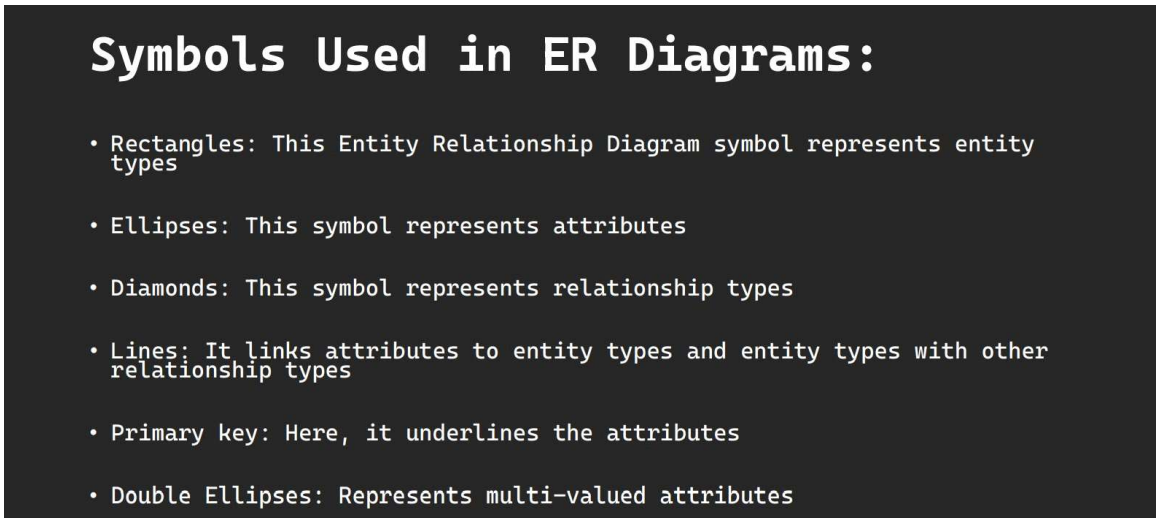
ERD helps both users and database developers to preview

the structure of the database before implementing the database.

ERD Symbols (12:33)







Which ERD symbol **represents entity types**?

Ellipses

Rectangles

Diamonds

Double Ellipses

Which ERD symbol **represents attributes**?

Rectangles

Diamonds

Ellipses

Lines

Which ERD symbol represents **relationship types**?

Diamonds

Ellipses

Rectangles

Double Ellipses

Which ERD symbol **links attributes to entity types** and **entity types with other relationship types**?

Rectangles

Ellipses

Diamonds

Lines

How to represent primary key in ERD?

Underline the attributes name.

Strong Entity is the entity that has primary key(s).

Weak Entity is represented by double rectangles because it has no primary key for it and it depends on other entities to be identified.

Components of ERD (2:27)

Components of ER Diagram:

We base an ER Diagram on three basic concepts:

- Entities
  - Entity (Strong Entity)
  - Weak Entity
- Attributes
  - Attribute
  - Key Attribute
  - Composite Attribute
  - Multivalued Attribute
  - Derived Attribute
- Relationships
  - One-to-One Relationships
  - One-to-Many Relationships
  - Many-to-One Relationships
  - Many-to-Many Relationships

Entity (Strong) and Weak Entity (9:25)

Entities

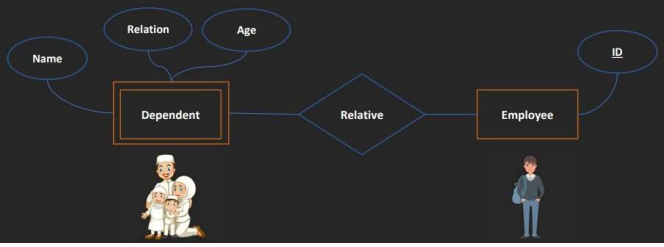
- Entity (Strong Entity).
- Weak Entity.

Entities(Strong Entities)

- An entity can be either a living or non-living component.
- It showcases an entity as a rectangle in an ER diagram.
- Each Entity has a primary key
- For example, in a student study course, both the student and the course are entities.

Weak Entities

- An entity that makes reliance over another entity is called a weak entity
- You showcase the weak entity as a double rectangle in ER Diagram.
- Weak Entity has no primary key.
- In the example below, Employee is a strong entity because it has a primary key attribute – ID. Unlike Dependent, the dependent is a weak entity because it does not have any primary key and the name here acts only as a discriminator.



There are **two types** of entities, **strong entities** and **weak entities**.

Strong Entity has to have primary key.

Weak Entity has no primary key(s)

An entity can be either a living or non-living component.

Which of the following assures the Entity Integrity?

Strong Entities

Weak Entities

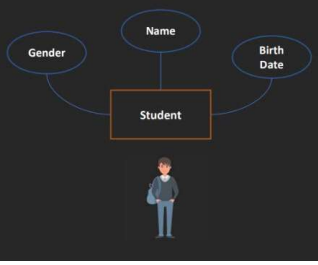
Strong Entity is represented by **single rectangle** in ERD.

Weak Entity is represented by **double rectangles** in ERD.

Attributes (7:08)

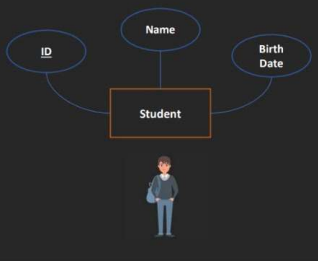
Attribute

- An attribute exhibits the properties of an entity.
- You can illustrate an attribute with an oval shape in an ER diagram.



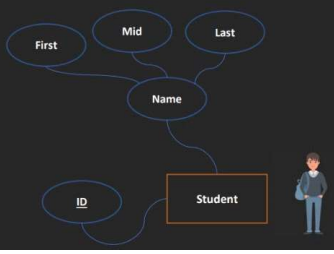
Key Attribute

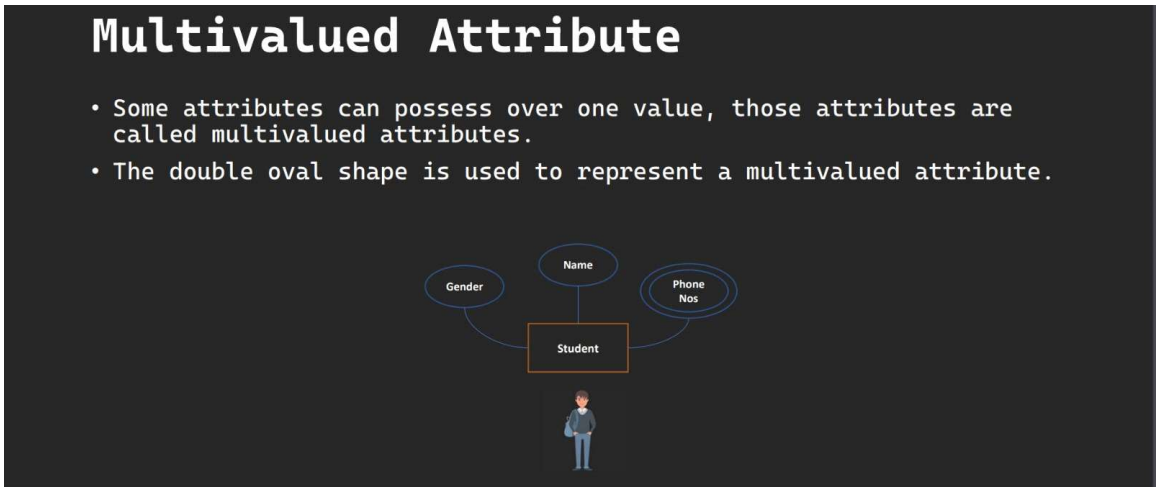
- Key attribute uniquely identifies an entity from an entity set.
- It underlines the text of a key attribute.
- For example: For a student entity, the ID can uniquely identify a student from a set of students.

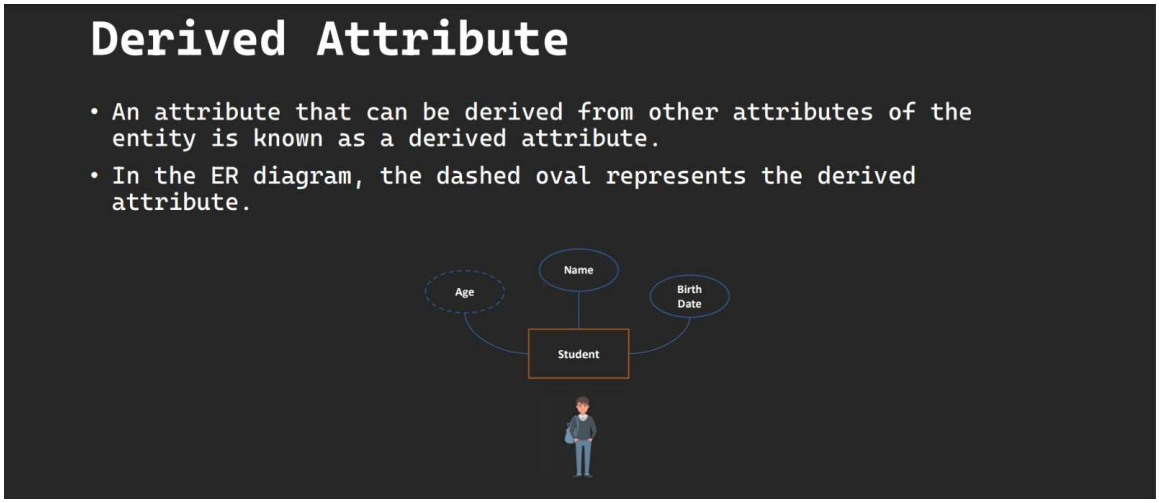


Composite Attribute

- An attribute that is composed of several other attributes is known as a composite attribute.
- An oval showcases the composite attribute, and the composite attribute oval is further connected with other ovals.







An attribute exhibits the properties of an entity.

You can illustrate an attribute with an oval shape in an ER diagram.

Key attribute uniquely identifies an entity from an entity set.

Key Attribute is represented by underlining the text of a

key attribute.

An attribute that is composed of several other attributes is known as a composite attribute.

An oval showcases the composite attribute, and the composite attribute oval is further connected with other ovals.

Some attributes can possess over one value, those attributes are called multivalued attributes.

The double oval shape is used to represent a multivalued attribute.

It's not recommended to have multivalued attributes.

An attribute that can be derived from other attributes of the entity is known as a derived attribute.

In the ER diagram, the dashed oval represents the derived attribute.

At the end the attribute is a field or column in the database.

Relationships (16:39)

Relationship

- The diamond shape showcases a relationship in the ER diagram.
- It depicts the relationship between two entities.
- In the example below, both the student and the course are entities, and Enrolled is the relationship between them.

- In the example below, both the student and the Identification-Card are entities, and “has” is the relationship between them.

- In the example below, both the customer and the order are entities, and “Place” is the relationship between them.

- In the example below, both the customer and the order are entities, and “Place” is the relationship between them.
- Order can have more than one product.

- In the example below, both the Member and the Book are entities, and “borrow” is the relationship between them.

- In the example below, both the Customer and the Car are entities, and “Rent” is the relationship between them.

Self Referencing Relationship

- In the example below, an employee has only one manager and manager is employee.
- It depicts the relationship between one entity.
- When an element of an entity is associated with a an element of same entity, it is called self relationship.

Employees

ID (PK)	Name	Salary	ManagerID (FK)
1	Ahmed	5000	Null
2	Amjad	1500	1
3	Maher	1300	2
4	Alia	500	2

Relationship Types

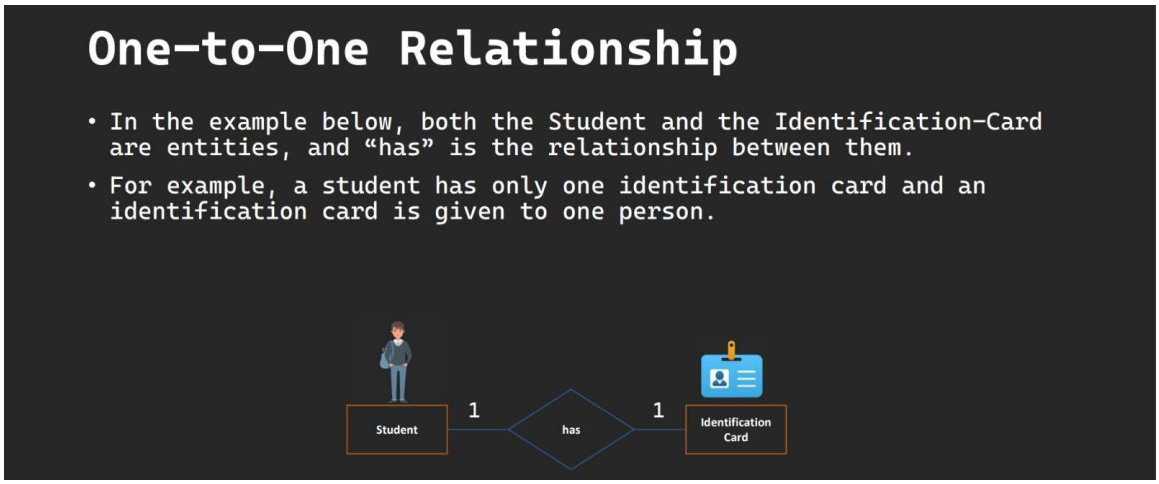
- One-to-One Relationship.
- One-to-Many Relationship.
- Many-to-One Relationship.
- Many-to-Many Relationship.

Relationship can be on one entity and we call it Self

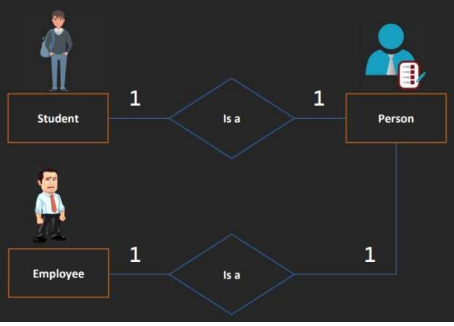
Reference Relationship.

We can have more than one relationship between two entities.

One-to-One Relationship (16:21)



- In the example below, both the Student and the Person are entities, and “is a” is the relationship between them.
- In the example below, both the Employee and the Person are entities, and “is a” is the relationship between them.



- In the example below, both the Member and the Book are entities, and “borrow” is the relationship between them.
- Member can borrow only one book, and book can be borrowed by one member.

- In the example below, both the Traveler and the Seat are entities, and “Set” is the relationship between them.
- Traveler can set on only one seat, and seat can be assigned to one traveler.

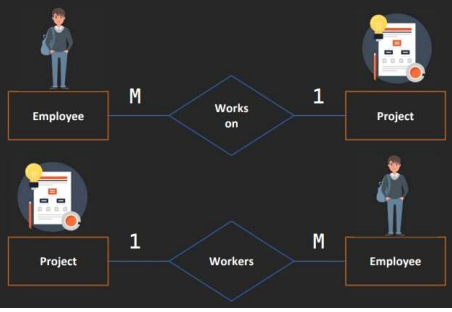
- In the example below, both the Employee and the Desk are entities, and “Set” is the relationship between them.
- Employee can set on only one Desk, and Desk can be assigned to one Employee.

- In the example below, both the Employee and the Task are entities, and “Assigned” is the relationship between them.
- For example, employee can work on a single Task, and a Task can be assigned to one employee.

When a single element of an entity is associated with a single element of another entity, it is called a one-to-one relationship.

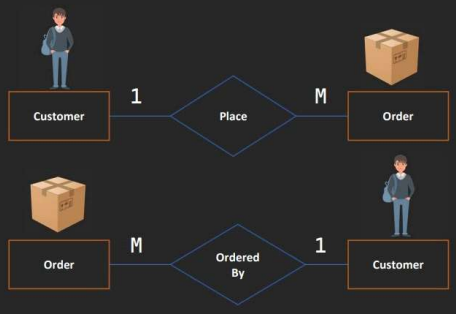
One-to-Many/Many-to-One Relationship (15:01)

- When more than one element of an entity is related to a single element of another entity, then it is called a many-to-one relationship.
- For example, employee can work on a single project, but a project can have many employees.



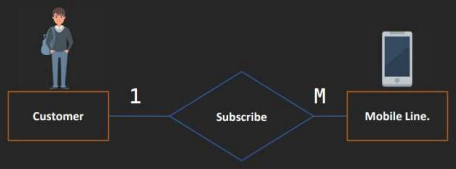
One-To-Many/Many-to-One Relationship

- When a single element of an entity is associated with more than one element of another entity, it is called a one-to-many relationship
- For example, a customer can place many orders, but an order cannot be placed by many customers.

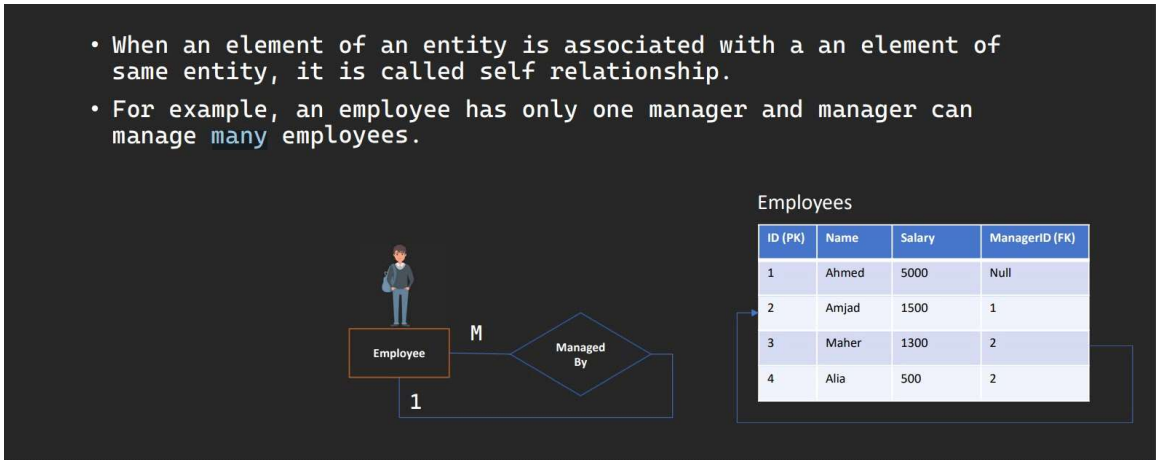


- In the example below, both the Member and the Book are entities, and “borrow” is the relationship between them.
- Member can borrow Many books, and book can be borrowed by one member.

- In the example below, both the Customer and the Mobile-Line are entities, and “Subscribe” is the relationship between them.
- Each Customer can subscribe to many lines, but each line can only be owned by one.



- In the example below, both the citizen and the Car are entities, and “Own” is the relationship between them.
- Citizen can own many cars, a car can be owned by one citizen.



When a single element of an entity is associated with more than one element of another entity, it is called a one-to-many relationship.

When more than one element of an entity is associated with a single element of another entity, it is called a many-to-one relationship.

Many-to-Many Relationship (6:26)

Many-to-Many Relationship

- When more than one element of an entity is associated with more than one element of another entity, it is called a Many-to-Many Relationship.
- In the example below, both the student and the course are entities, and Enrolled is the relationship between them.
- Student can be enrolled in many courses.
- Course can be studied by Many students.

- In the example below, both the customer and the order are entities, and “Place” is the relationship between them.
- Order can have more than one product.
- Products can be placed in many orders.

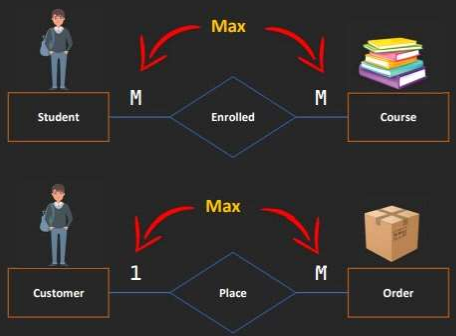
When more than one element of an entity is associated with more than one element of another entity, it is called a Many-to-Many Relationship.

what : لازم تسأل حالك entities لما بدك تحدد طبيعة العلاقة بين كل 2 is the maximum number I can take from bla bla وهكذا

Cardinality vs Ordinality (11:57)

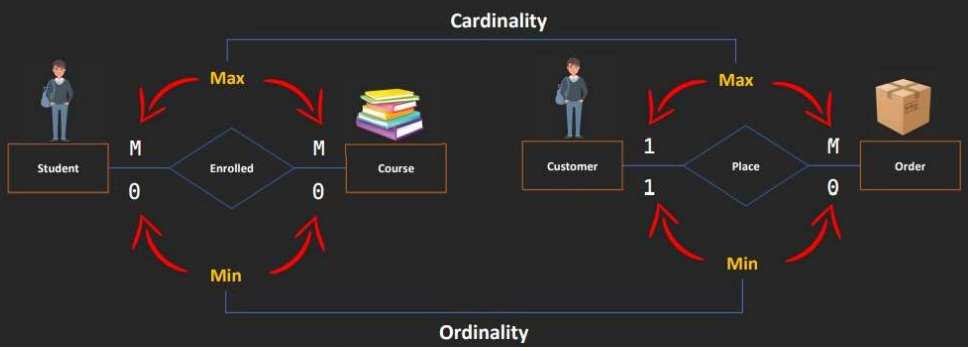
Cardinality

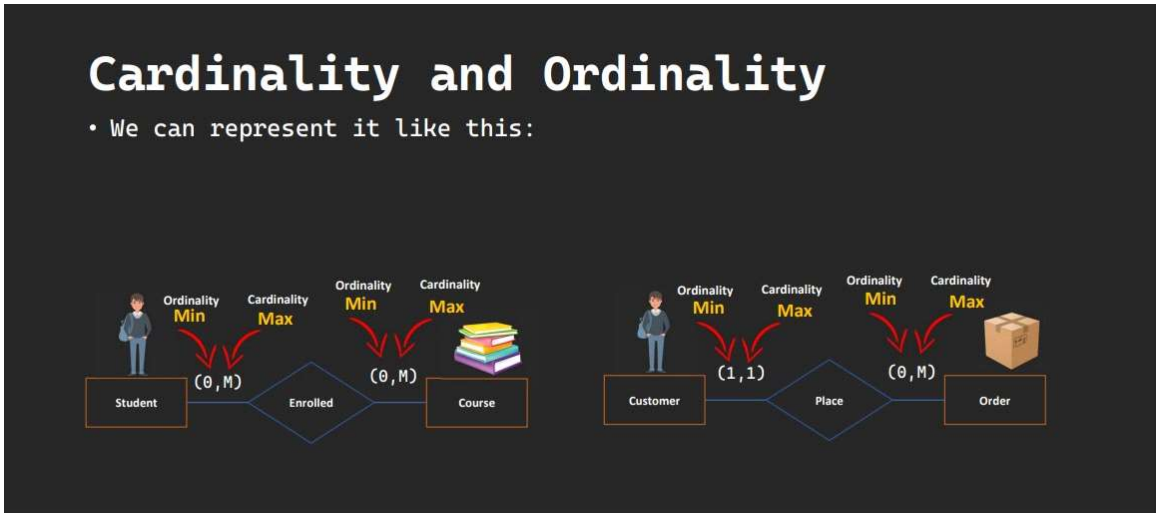
- Cardinality refers to the maximum number of times an instance in one entity can relate to instances of another entity



Ordinality

- Ordinality, on the other hand, is the minimum number of times an instance in one entity can be associated with an instance in the related entity. (in other words It specify if it's optional/mandatory/required or not).





Cardinality refers to the **maximum** number of times an instance **in one entity** can relate to instances of another entity.

Ordinality, on the other hand, is the **minimum number** of times an instance **in one entity** can be associated with an instance in the related entity. (in other words It specifies if it's optional/mandatory/required or not).

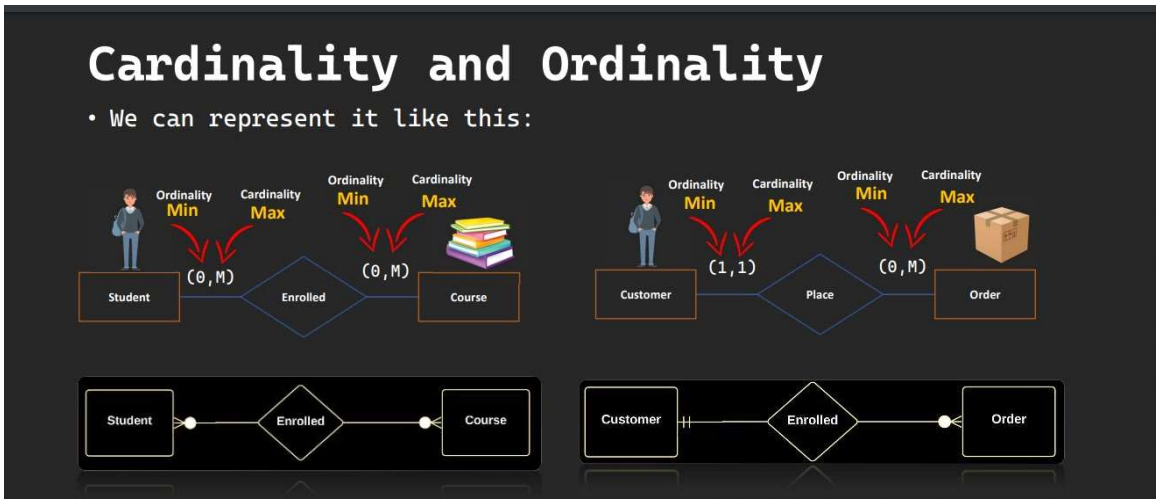
(0,M) : the first number represents the ordinality and the second one represents the cardinality.

When you ask **"What is the Max?"** you are identifying the cardinality.

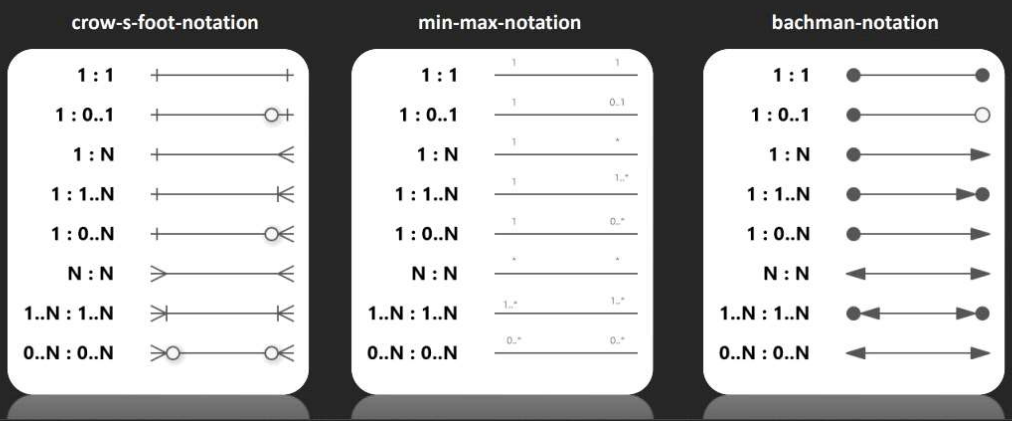
When you ask "What is the Min?" you are identifying

the ordinality.

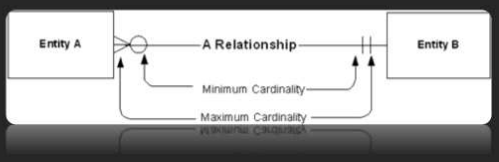
Cardinality Symbols and Practices (29:44)

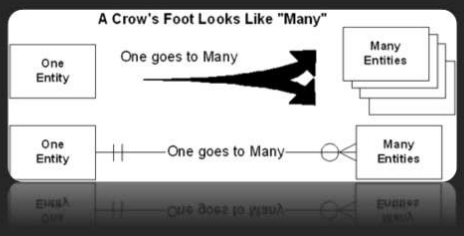


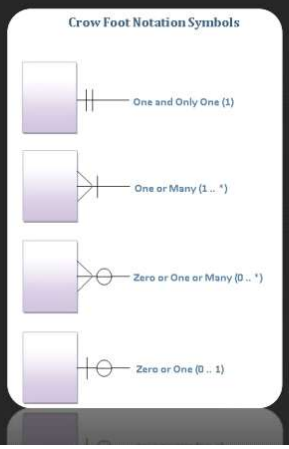
Cardinality Notation Symbols:

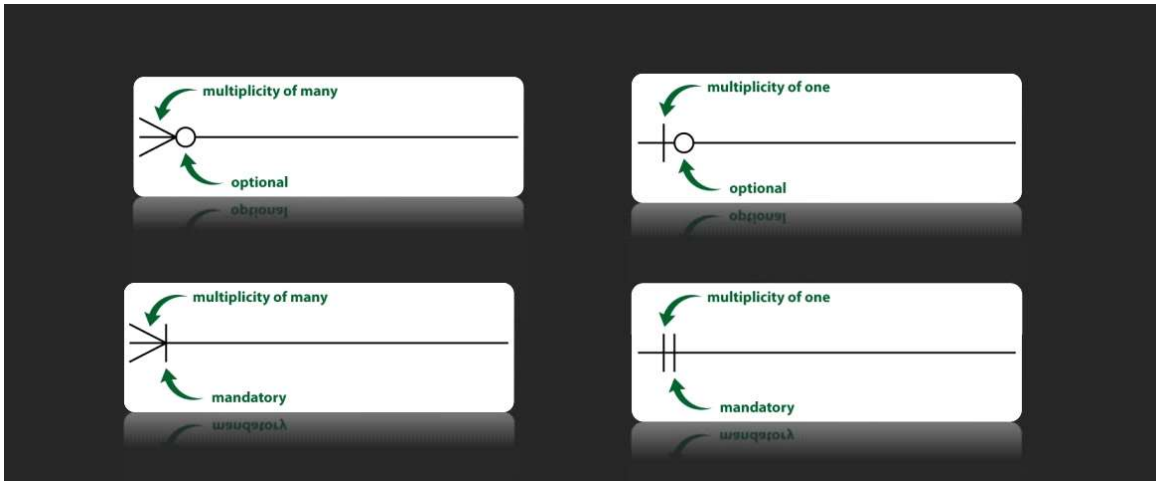


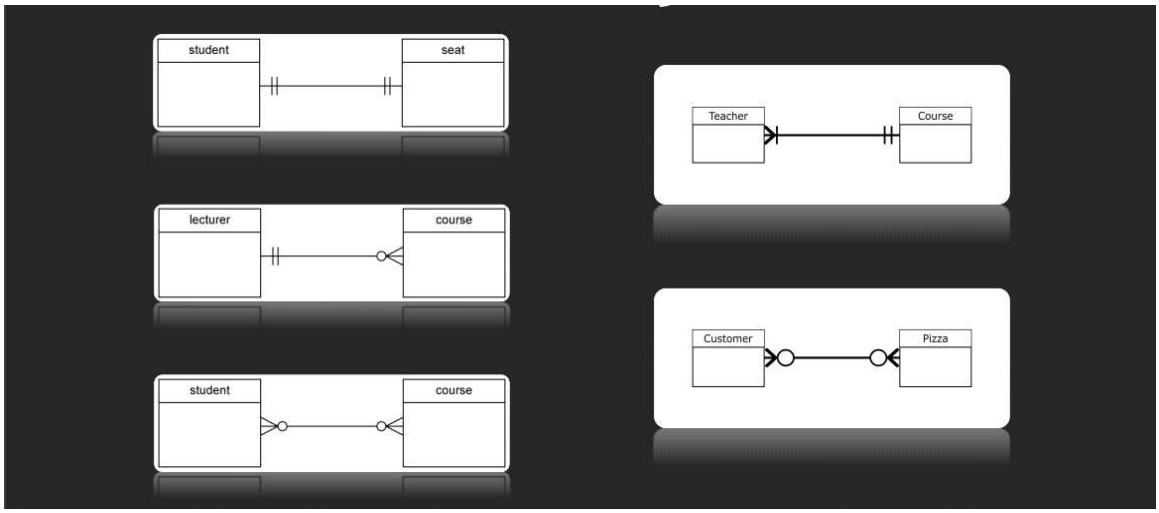
Crow's foot notation Symbols

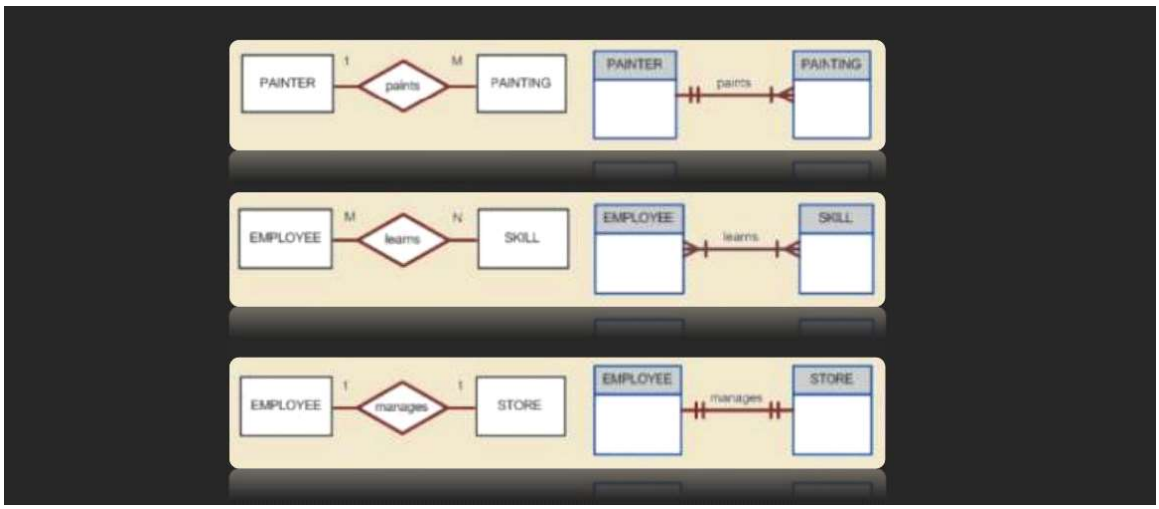




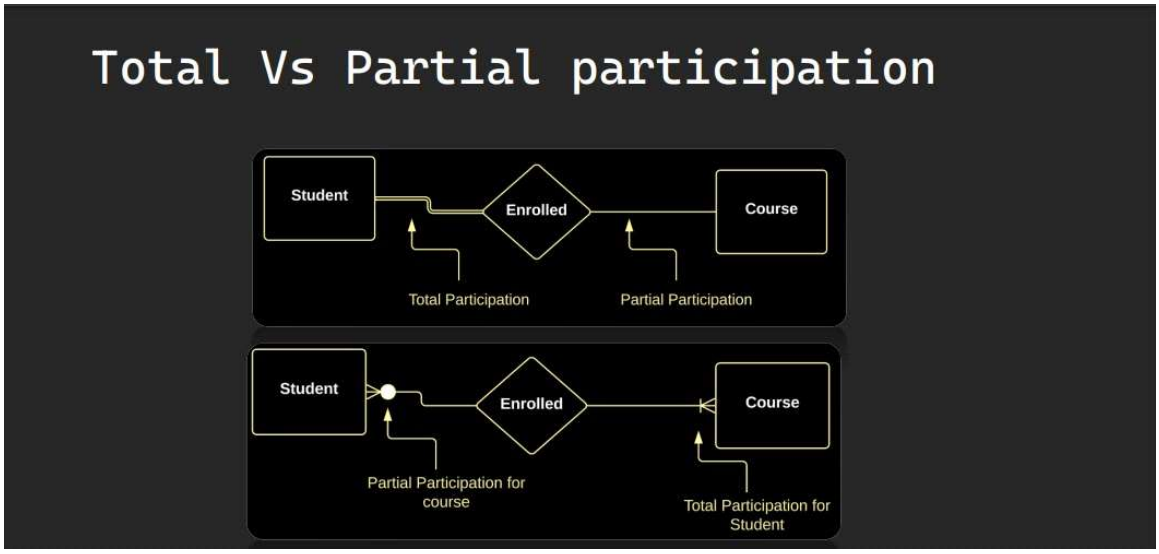


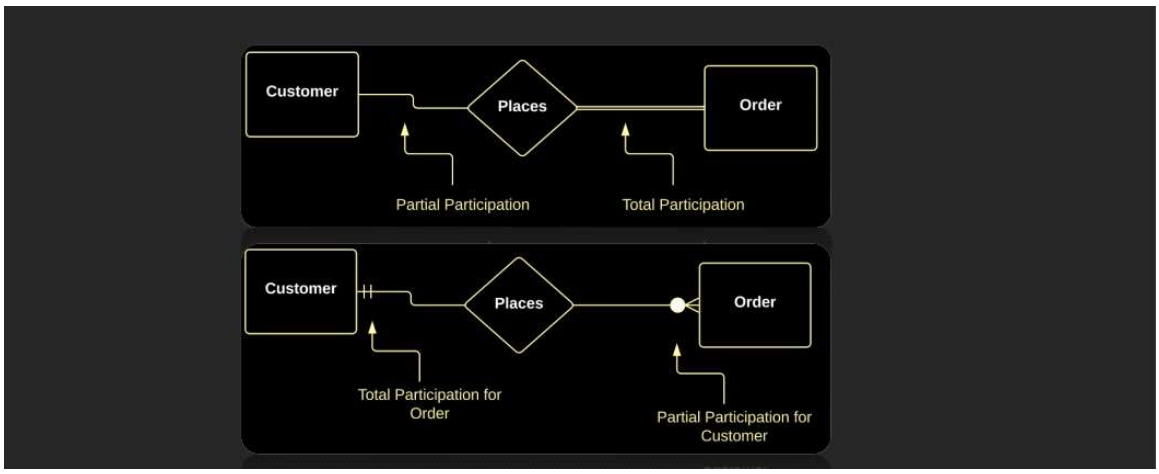

multiplicity of many
optional
multiplicity of one
optional
multiplicity of many
mandatory
multiplicity of one
mandatory


student
seat
lecturer
course
student
course
Teacher
Course
Customer
Pizza

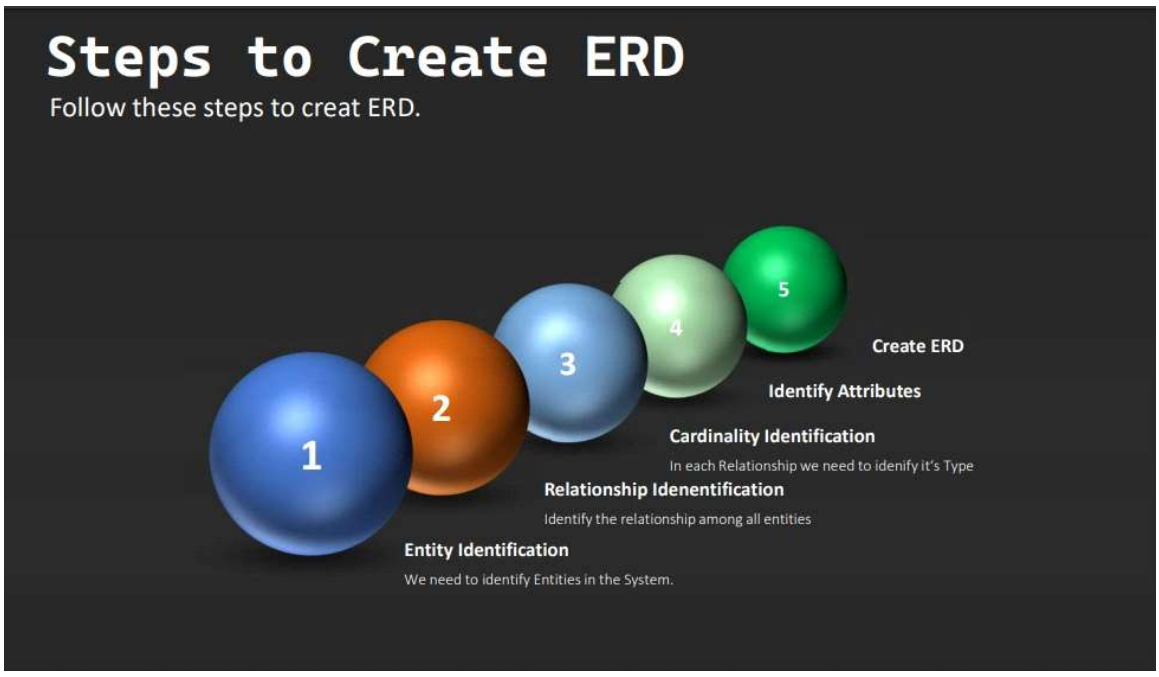

PAINTER
1
paints
M
PAINTING
PAINTER
paints
PAINTING
EMPLOYEE
M
learns
N
SKILL
EMPLOYEE
learns
SKILL
EMPLOYEE
1
manages
1
STORE
EMPLOYEE
manages
STORE

Total Vs Partial Participation (5:54)





Process of Creating ERD Step by Step - Small Project (26:57)

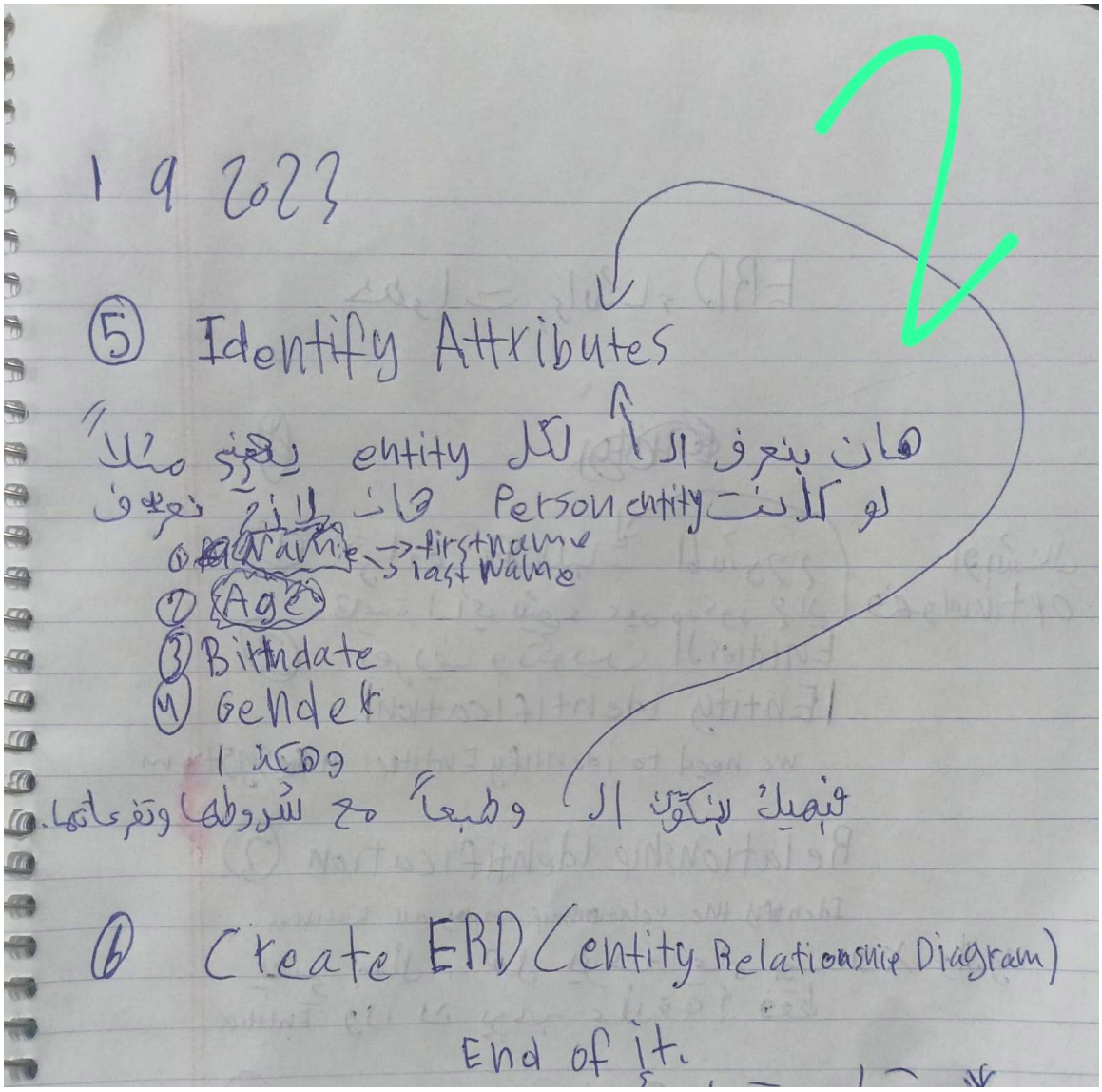


خطوات إنشاء ERD

1. قراءة المتطلبات المشروع
قاعدة : أي شيء غير مذكور فهو اوبشنال optional
2. تعريف وتكوين الـ Entities
Entity identification
we need to identify Entities in the System

3. Relationship Identification
Identify the relationship among all Entities
يعني بنسأل : هل يوجد علاقة بين الـ X والـ Y Entities ولا لا يوجد علاقة ؟ فقط

4. Cardinallity identification
In each Relationship, We need to identify it's type.
يعني بعد ما نحدد هل يوجد علاقة أولا بين كل entity (الخطوة رقم 3 أعلاه)
* يجب أن نحدد نوع العلاقة مثل: one to one, one to Many, many to many


1 9 2023
⑤ Identify Attributes
هان بنعرف ال لكل entity يعني مثلا
لو كانت Person entity هان لازم نعرف
① Name → firstname
lastname
② Age
③ Birthdate
④ Gender
وهكذا
فبهيك بتكون ال وطبعاً مع شروطها وتفرعاتها.
⑥ Create ERD (entity Relationship Diagram)
End of it.

Create ERD for University

Requirements:

- In a university, a Student enrolls in Courses. A student can enroll in more than one course, Each course can have more than one student.
- Each student has only one mentor(prof), Professor can mentor more than one student.
- Each Professor can teach only one course.
- Each Course can have only one professor.
- Course can have un-assigned professor.
- Professor can teach no courses.

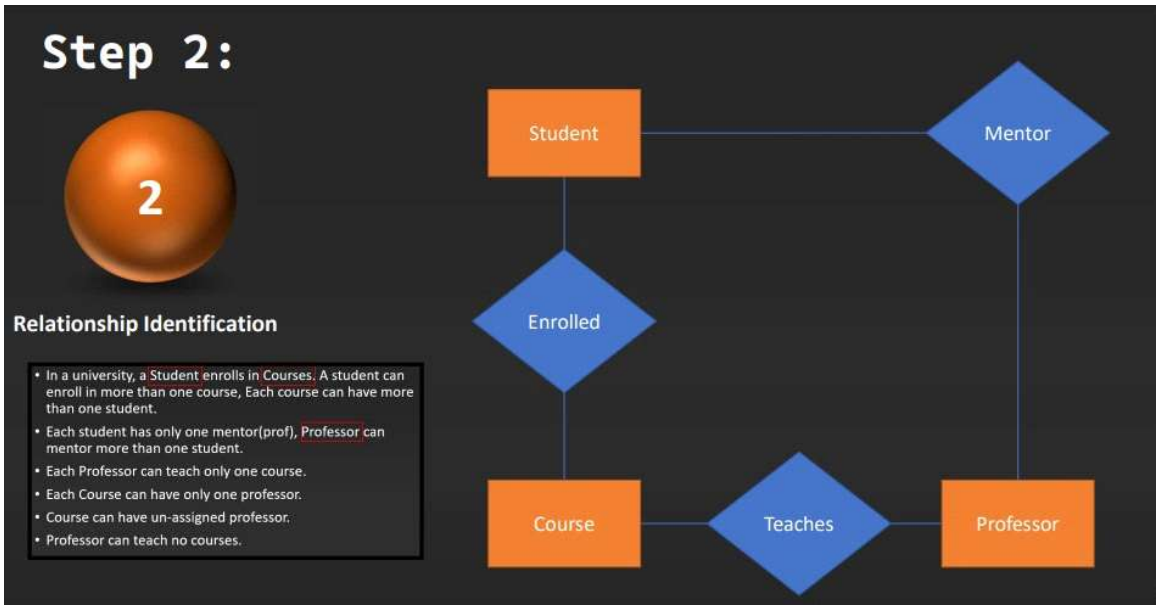
Step 1:

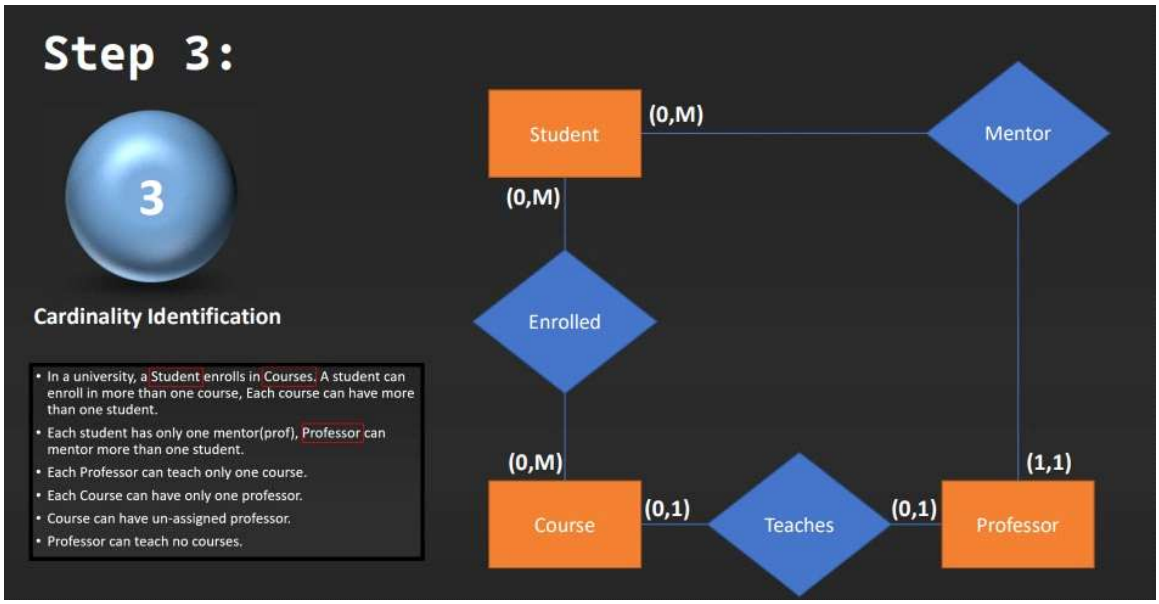
Entity Identification

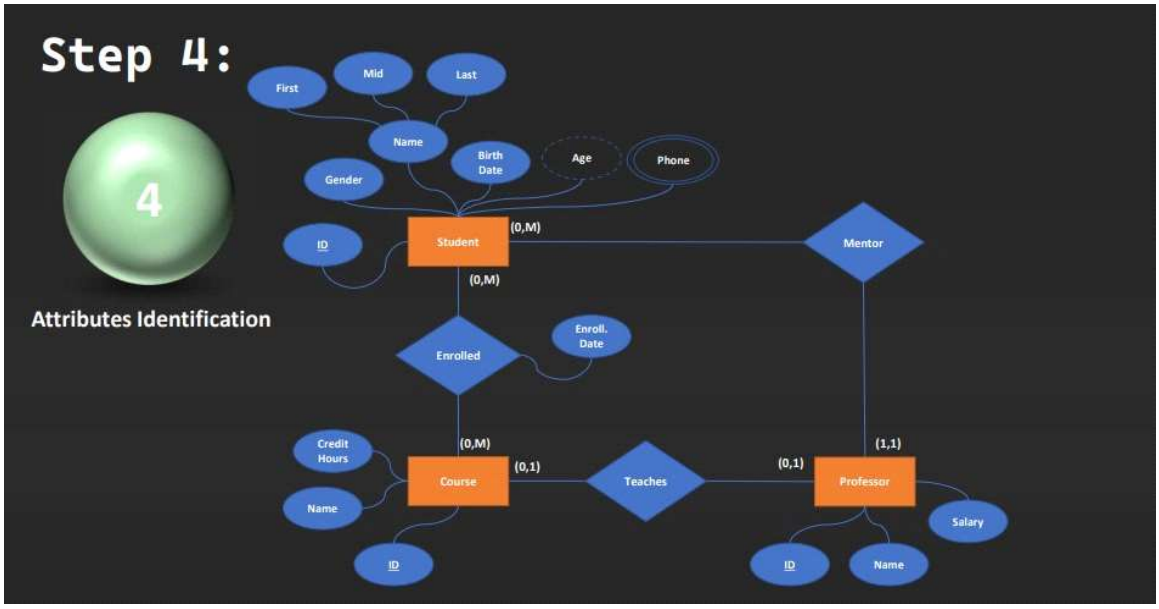
Requirements:

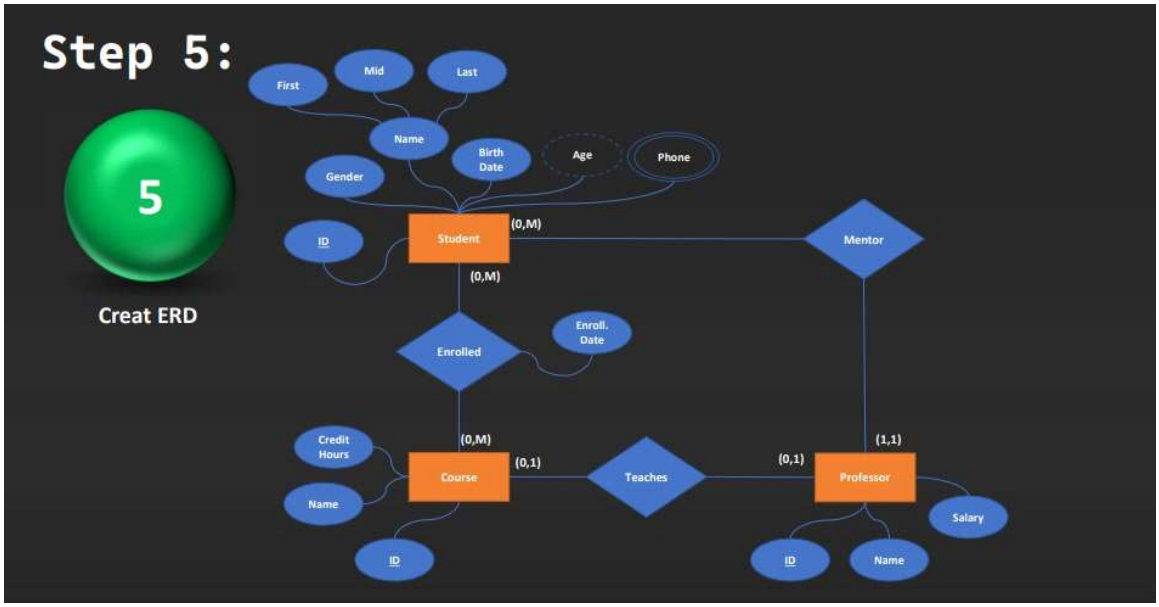
- In a university, a Student enrolls in Courses. A student can enroll in more than one course, Each course can have more than one student.
- Each student has only one mentor(prof), Professor can mentor more than one student.
- Each Professor can teach only one course.
- Each Course can have only one professor.
- Course can have un-assigned professor.
- Professor can teach no courses.

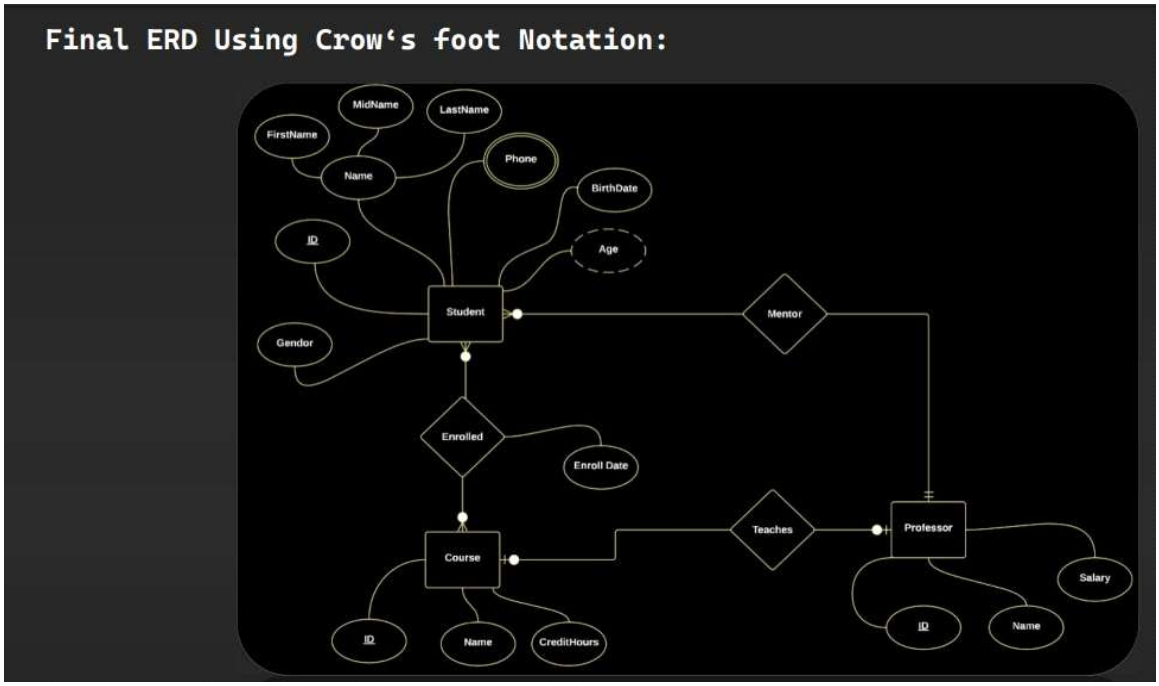
Student | Course | Professor


Step 2:
2
Relationship Identification
• In a university, a Student enrolls in Courses. A student can enroll in more than one course, Each course can have more than one student.
• Each student has only one mentor(prof), Professor can mentor more than one student.
• Each Professor can teach only one course.
• Each Course can have only one professor.
• Course can have un-assigned professor.
• Professor can teach no courses.
Student
Mentor
Enrolled
Course
Teaches
Professor


Step 3:
3
Cardinality Identification
• In a university, a Student enrolls in Courses. A student can enroll in more than one course, Each course can have more than one student.
• Each student has only one mentor(prof), Professor can mentor more than one student.
• Each Professor can teach only one course.
• Each Course can have only one professor.
• Course can have un-assigned professor.
• Professor can teach no courses.
Student
(0,M)
Mentor
(0,M)
Enrolled
(0,M)
(1,1)
Course
(0,1)
Teaches
(0,1)
Professor

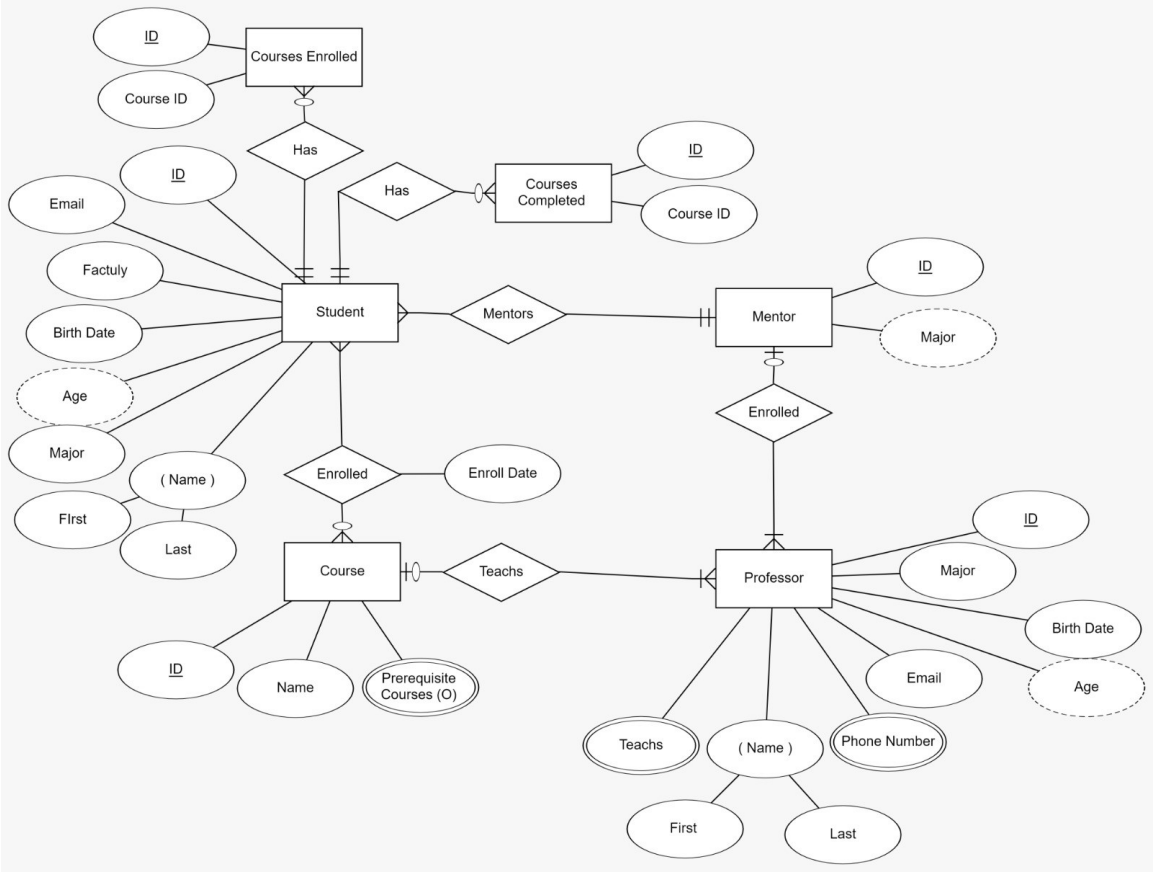

Step 4:
4
Attributes Identification
First
Mid
Last
Name
Gender
Birth Date
Age
Phone
ID
Student
(0,M)
Mentor
(0,M)
Enrolled
Enroll. Date
Credit Hours
(0,M)
Course
(0,1)
Teaches
(0,1)
(1,1)
Professor
Salary
Name
ID
ID
Name


Step 5:
5
Creat ERD
First
Mid
Last
Name
Gender
Birth Date
Age
Phone
ID
Student
(0,M)
Mentor
(0,M)
Enrolled
Enroll. Date
Credit Hours
(0,M)
Course
(0,1)
Teaches
(0,1)
(1,1)
Professor
Salary
Name
ID
ID
Name

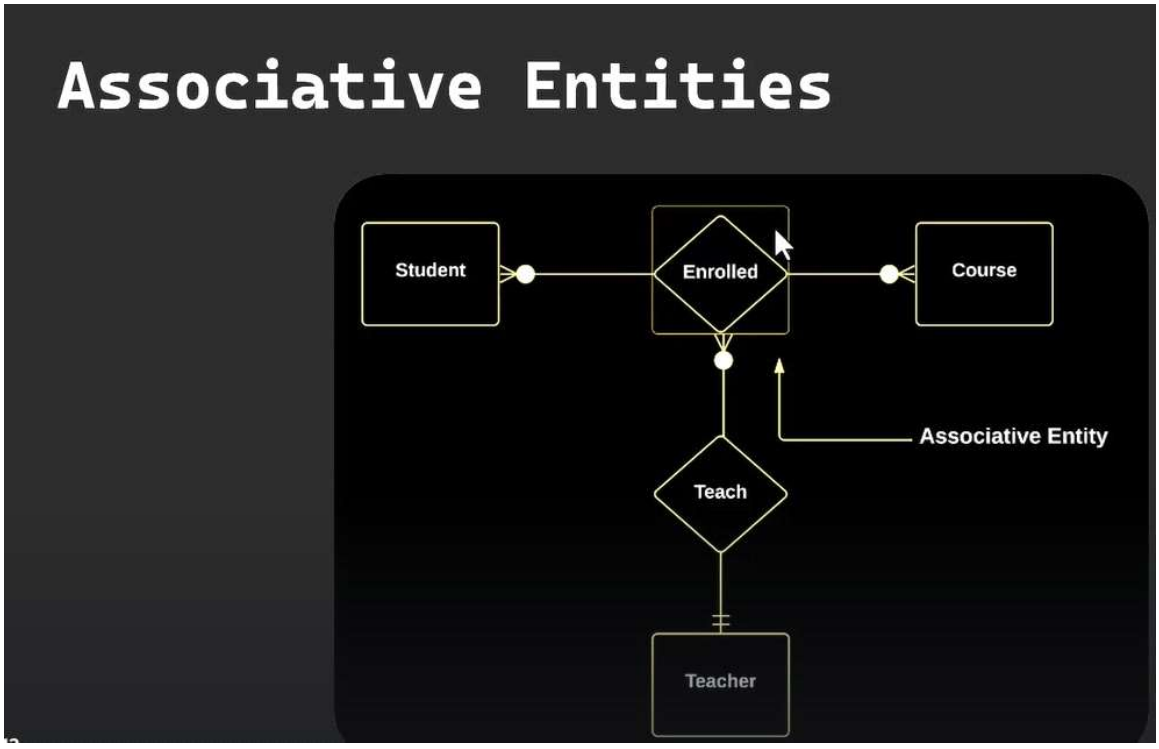


University ERD Project

My University ERD Project



Aggregation / Associative Entities (5:45)



An **associative entity** is a type of entity in a database that is used to **model a many-to-many** relationship between two other entities. It is also sometimes called a **junction table**, a **linking table**, or a **cross-reference table**.

By using an associative entity, we can represent complex relationships between entities in a structured and **efficient way, without** having to **duplicate data** or create confusing relationships between tables. It allows us to model many-to-many relationships and **avoid data redundancy**, making it a useful concept in database design.

Representing relationship between an entity and a relationship which may be required in some scenarios. In those cases, a relationship with its corresponding entities is aggregated into a higher level entity. Aggregation is an abstraction through which we can represent relationships as higher level entity sets.

ولا فهمت اشي

Aggregation is a specialized form of association between two or more Entities in which each Entity has its own Existence.

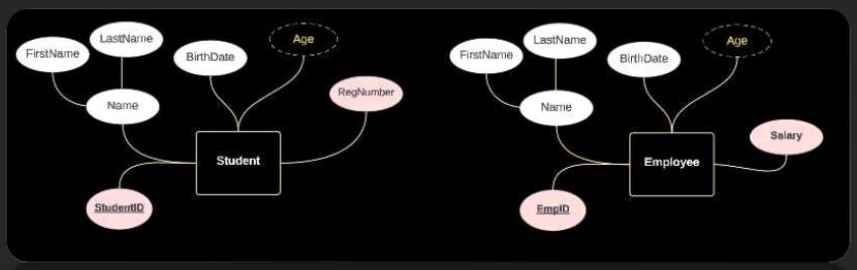
In aggregation, the relation between two entities is treated as a single entity. In aggregation, relationship with its corresponding entities is aggregated into a higher level entity.

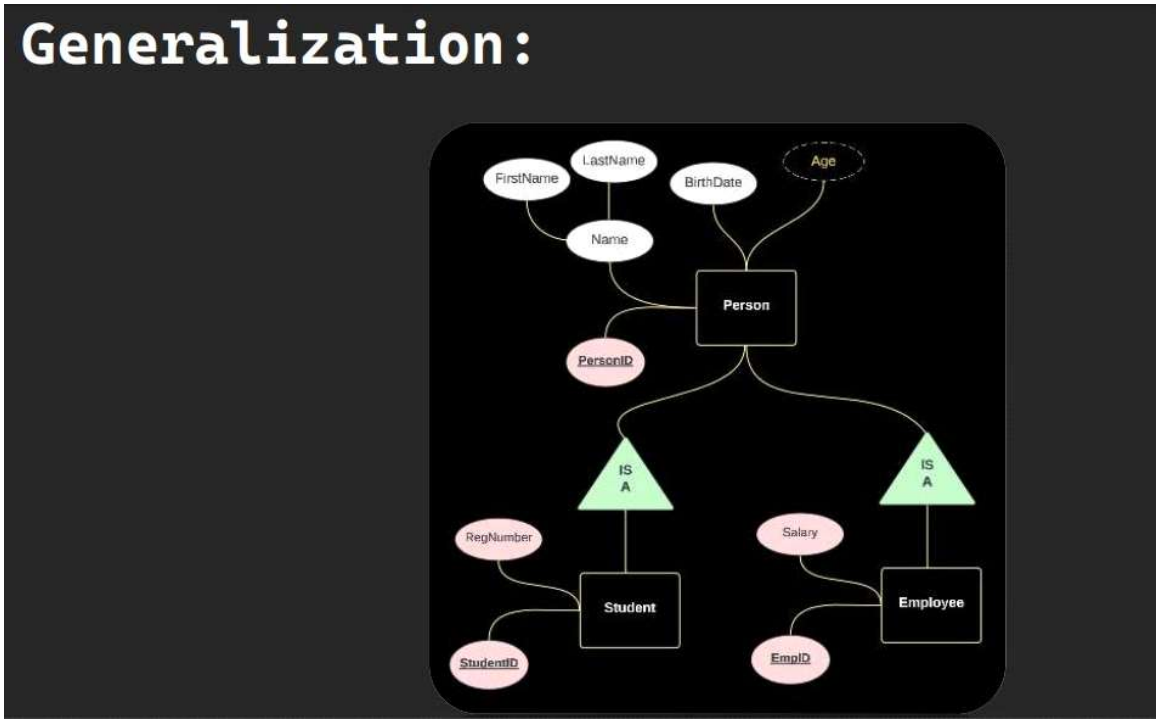
Generalization (13:26)

Generalization

- Generalization is the process of extracting common properties from a set of entities and create a generalized entity from it.
- It is a bottom-up approach in which two or more entities can be generalized to a higher level entity if they have some attributes in common.

Problem:





Generalization is the process of extracting common properties from a set of entities and create a generalized entity from it.

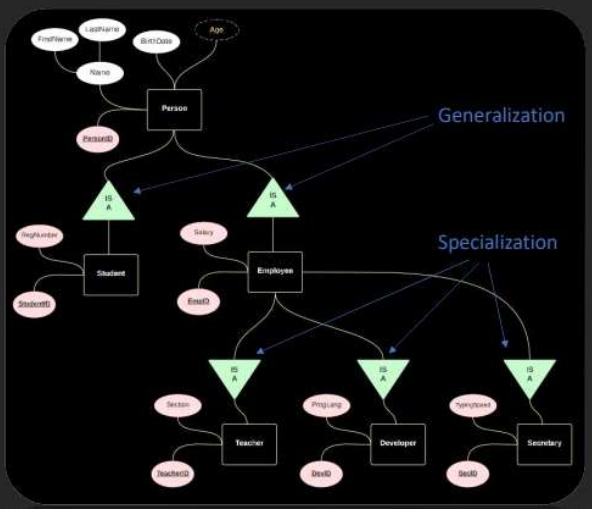
Generalization is **Bottom-Up** approach.

Specialization (15:38)

Specialization

- In specialization, an entity is divided into sub-entities based on their characteristics.
- Specialization is a top-down approach in which a higher-level entity is divided into multiple specialized lower-level entities.

Specialization:



In specialization, an entity is divided into sub-entities based on their characteristics.

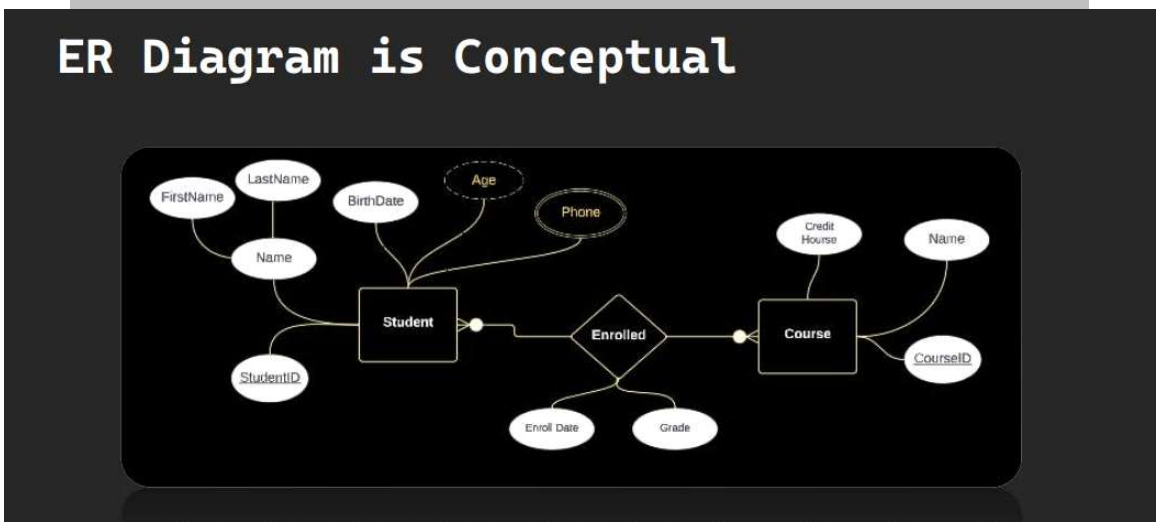
Specialization is a top-down approach.

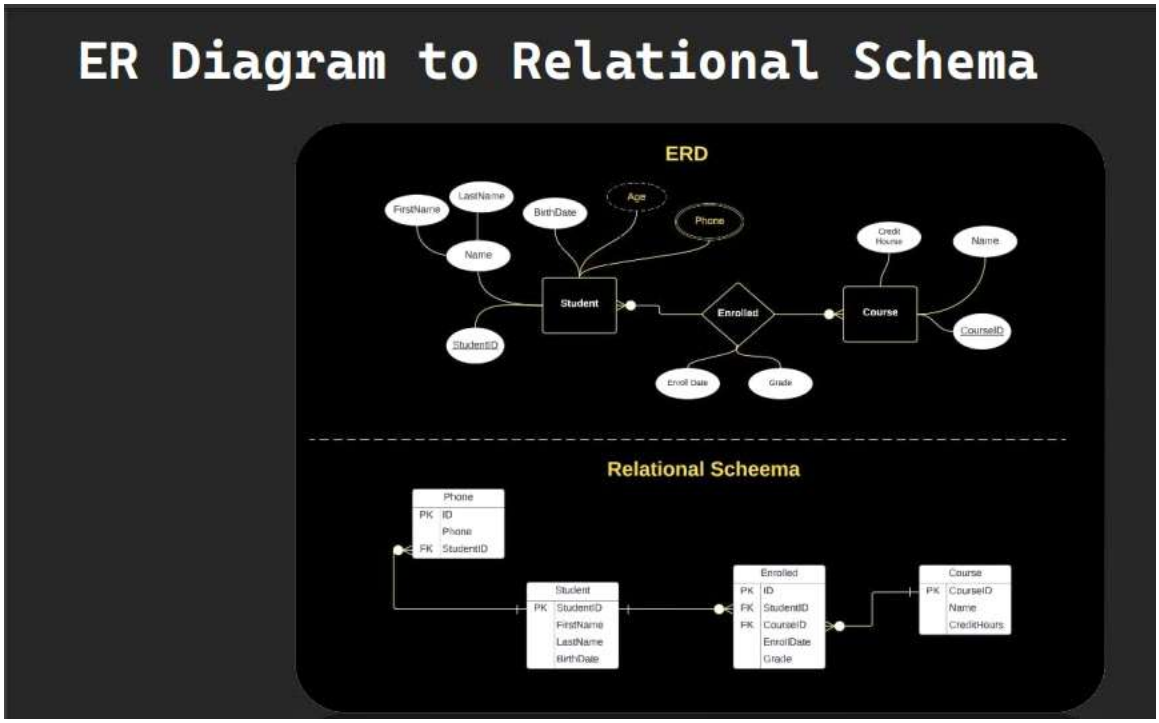
Specialization is a top-down approach in which a higher-

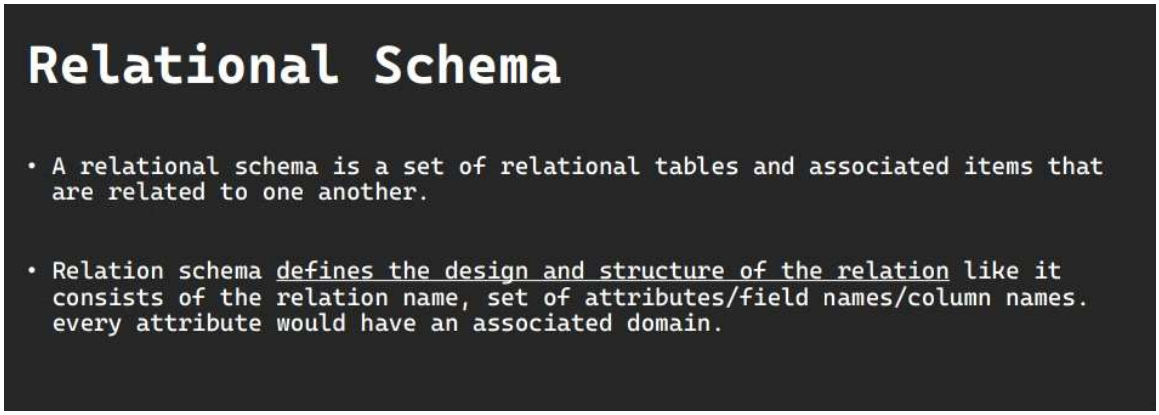
level entity is divided into multiple specialized lower-level entities.

Relational Schema

What is Relational Schema? (13:58)





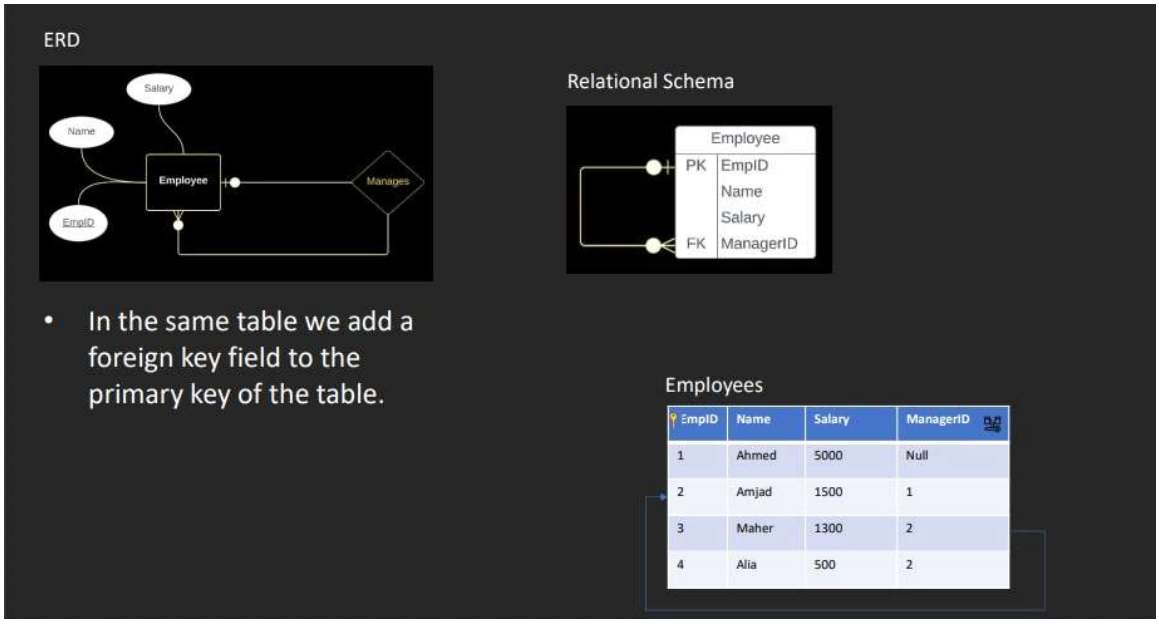


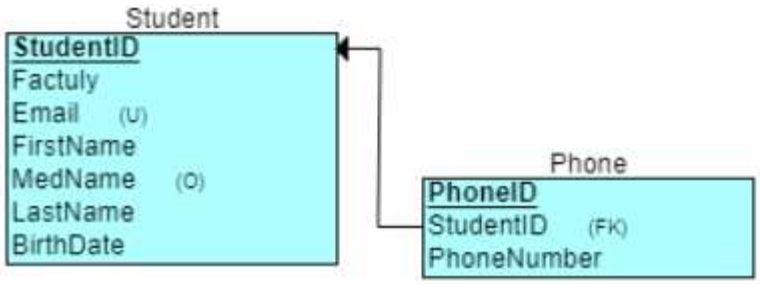
A relational schema is a set of relational tables and associated items that are related to one another.

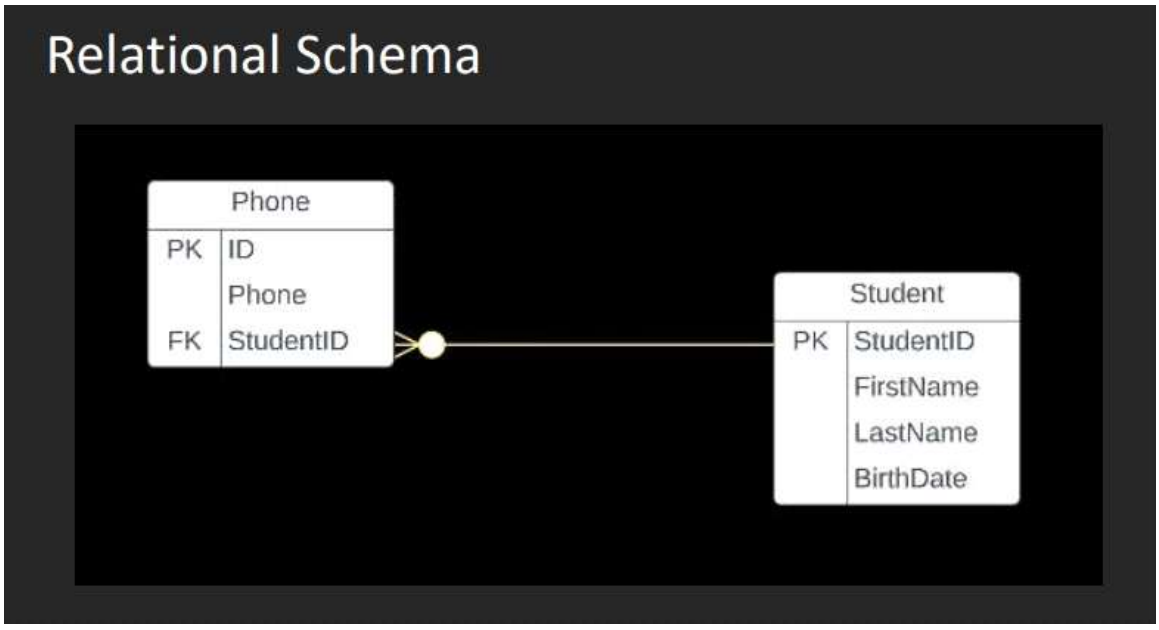
Relation schema defines the design and structure of the relation like it consists of the relation name, set of attributes/field names/column names. every attribute

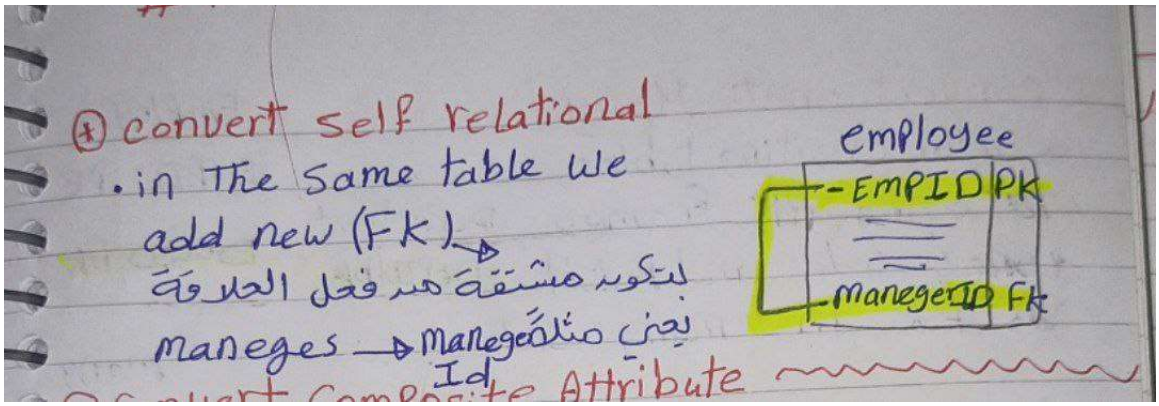
would have an associated domain.

Convert Self Referential ==> Relational Schema (7:07)



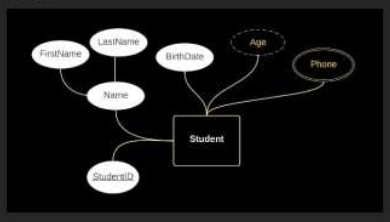

Student
StudentID
Factuly
Email (U)
FirstName
MedName (O)
LastName
BirthDate
Phone
PhoneID
StudentID (FK)
PhoneNumber


Relational Schema
Phone
PK ID
Phone
FK StudentID
Student
PK StudentID
FirstName
LastName
BirthDate


convert self relational
in The Same table we
add new (FK)
بتكون مشتقة من فعل العلاقة
maneges → Manegeمثلاً يعني
Id
Employee
EMPID PK
ManegerID FK

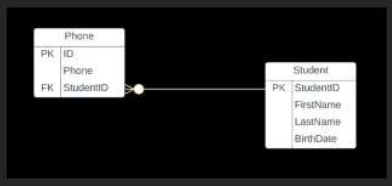
Convert Composite-Multivalued-Derived Attributes ==> Relational Schema (15:19)

ERD



1. Create Table for the entity.
2. Move attributes to that table.
3. Composite attribute: only take roots.
4. Derived Attribute will be ignored.
5. We create another table for the multivalued attribute.
6. We take the primary key from the main entity and put it on the new one as FK

Relational Schema



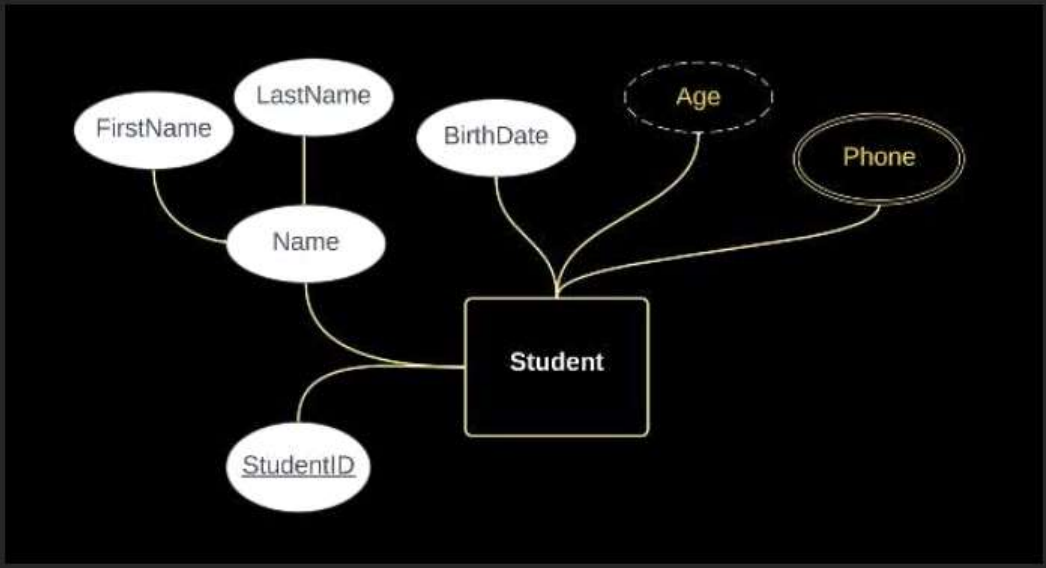
Students

StudentID	FirstName	LastName	BirthDate
1	Mohammed	Abu-Hadhoud	6-11-1977
2	Ali	Amjad	12-3-2000
3	Maha	Omaran	11-6-2003
4	Fidaa	Safwan	6-6-1991

Phones

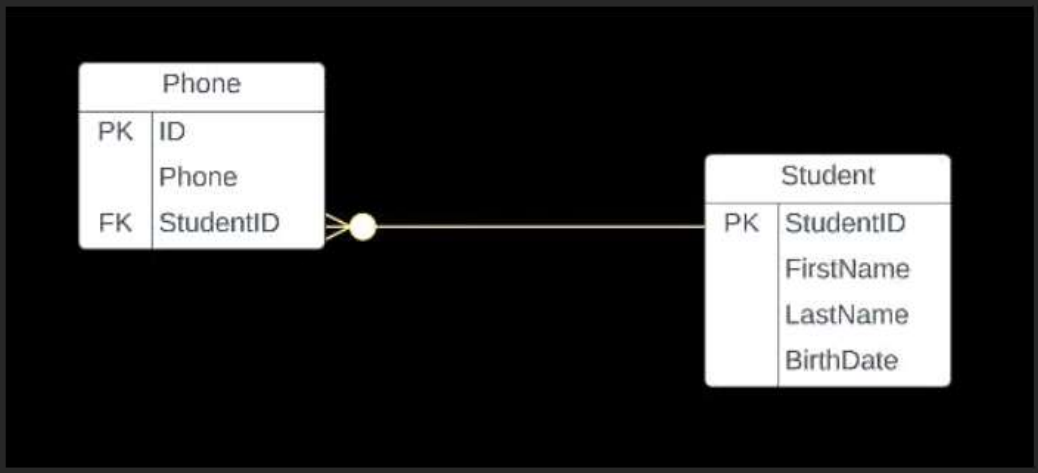
PhoneID	StudentID	Phone
1	1	07729929
2	1	079882882
3	2	059939921
4	3	065500501

ERD



1. Create Table for the entity.
2. Move attributes to that table.
3. Composite attribute: only take roots.
4. Derived Attribute will be ignored.
5. We create another table for the multivalued attribute.
6. We take the primary key from the main entity and put it on the new one as FK

Relational Schema

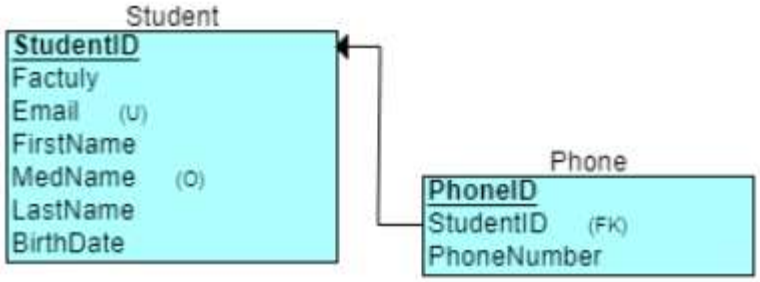


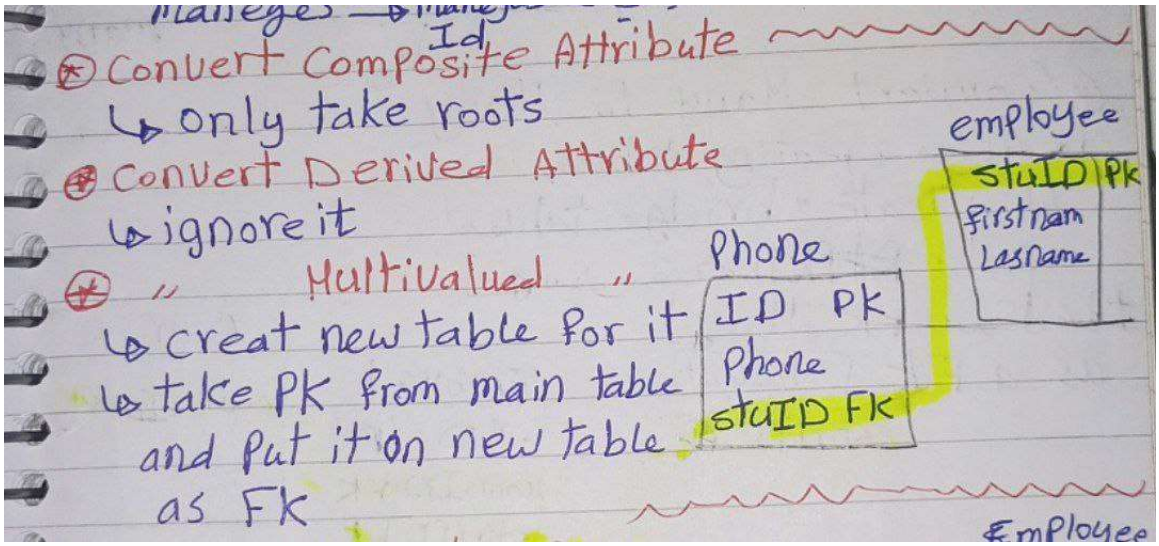
Students

StudentID	FirstName	LastName	BirthDate
1	Mohammed	Abu-Hadhoud	6-11-1977
2	Ali	Amjad	12-3-2000
3	Maha	Omaran	11-6-2003
4	Fidaa	Safwan	6-6-1991

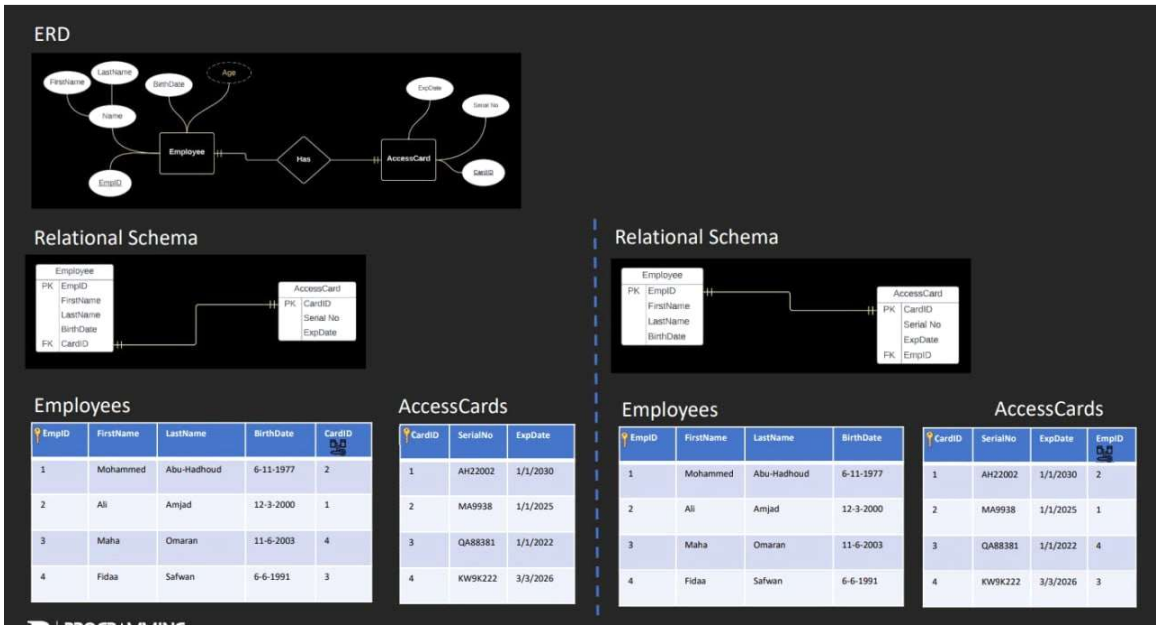
Phones

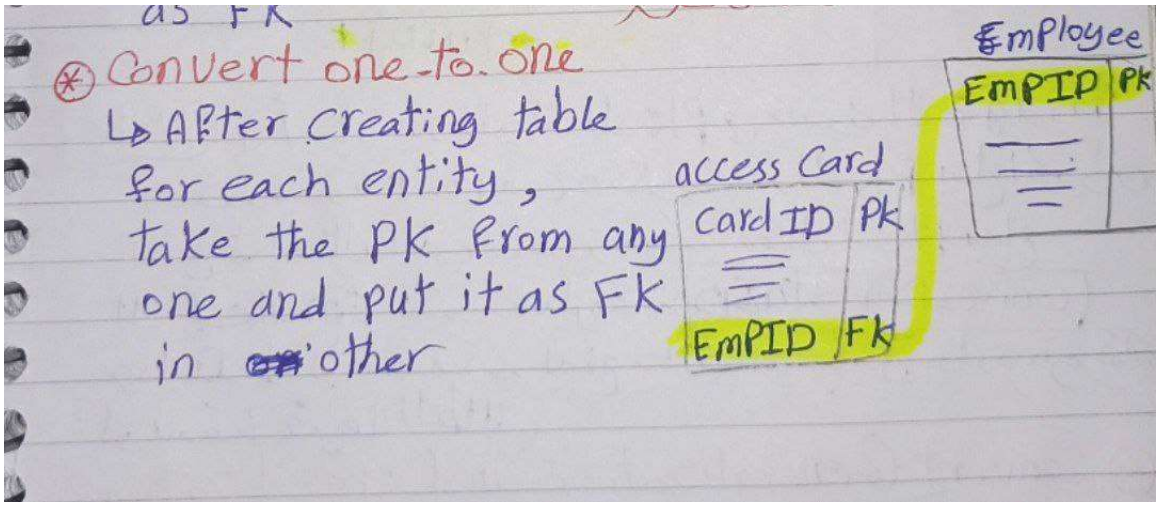
PhoneID	StudentID	Phone
1	1	07729929
2	1	079882882
3	2	059939921
4	3	065500501



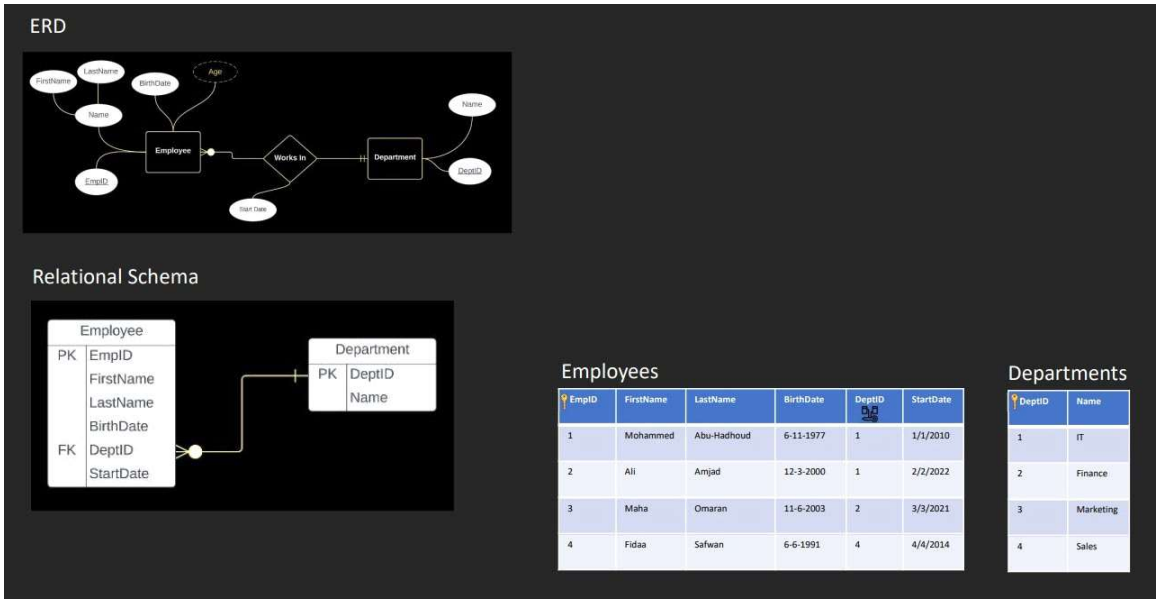


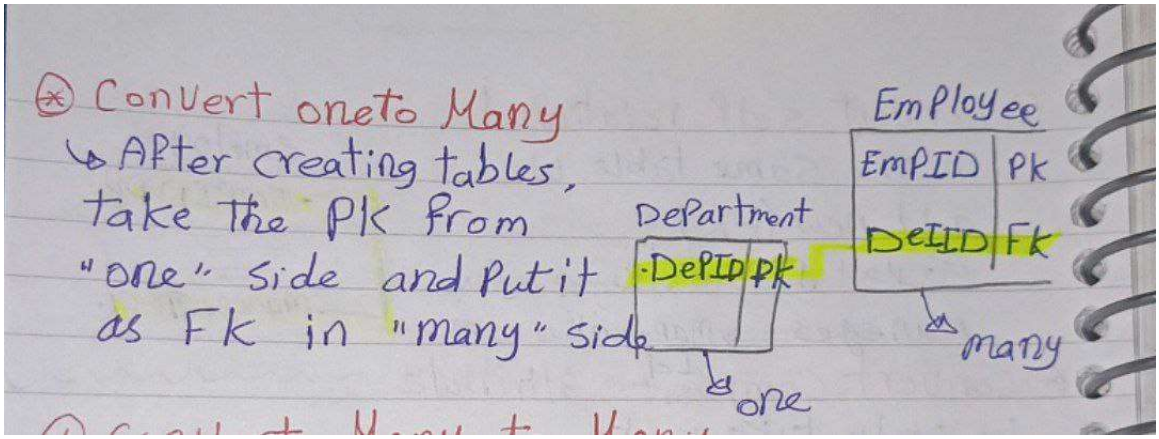
Convert One-to-One ==> Relational Schema





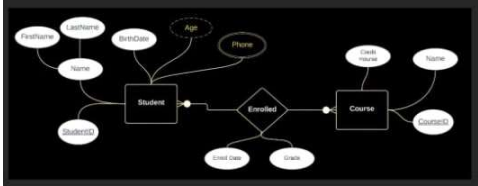
Convert One-to-Many/Many-to-One ==> Relational Schema



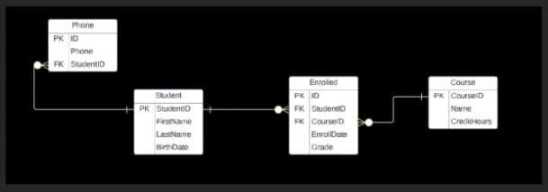


Convert Many-to-Many ==> Relational Schema

ERD



Relational Schema



Phones

PhoneID	StudentID	Phone
1	1	07729929
1	1	079882882
3	2	059939921
4	3	065500501

Students

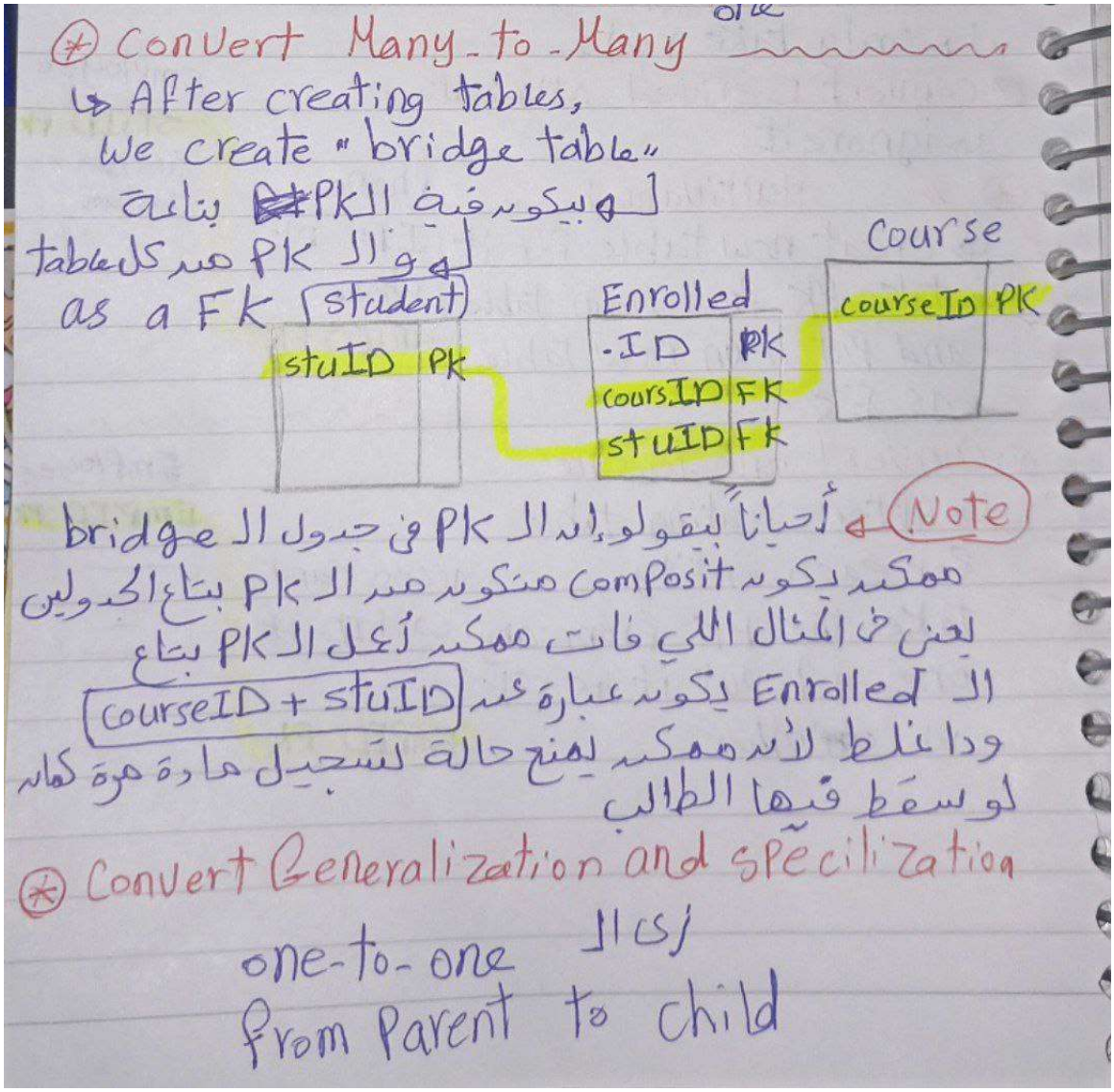
StudentID	FirstName	LastName	BirthDate
1	Mohammed	Abu-Hadhoud	6-11-1977
2	Ali	Amjad	12-3-2000
3	Maha	Omaran	11-6-2003
4	Fidaa	Safwan	6-6-1991

Enrollments

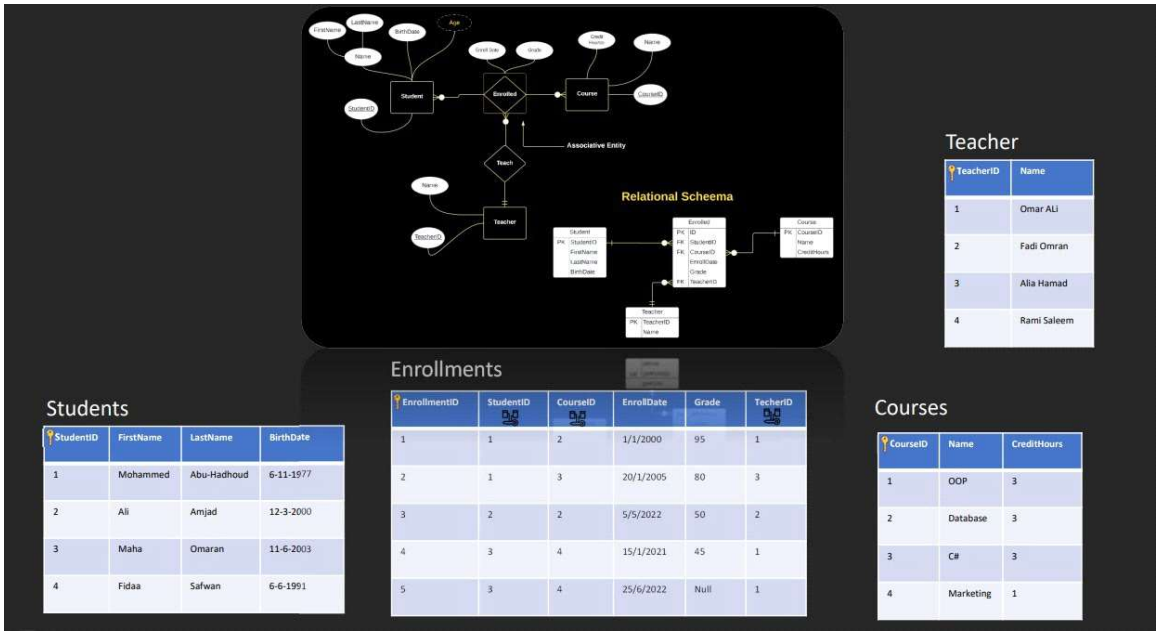
EnrollmentID	StudentID	CourseID	EnrollDate	Grade
1	1	2	1/1/2000	95
2	1	3	20/1/2005	80
3	2	2	5/5/2022	50
4	3	4	15/1/2021	45
5	3	4	25/6/2022	Null

Courses

CourseID	Name	CreditHours
1	OOP	3
2	Database	3
3	C#	3
4	Marketing	1



Convert Associative Entity to Relational Schema



Teacher

TeacherID	Name
1	Omar ALi
2	Fadi Omran
3	Alia Hamad
4	Rami Saleem

Students

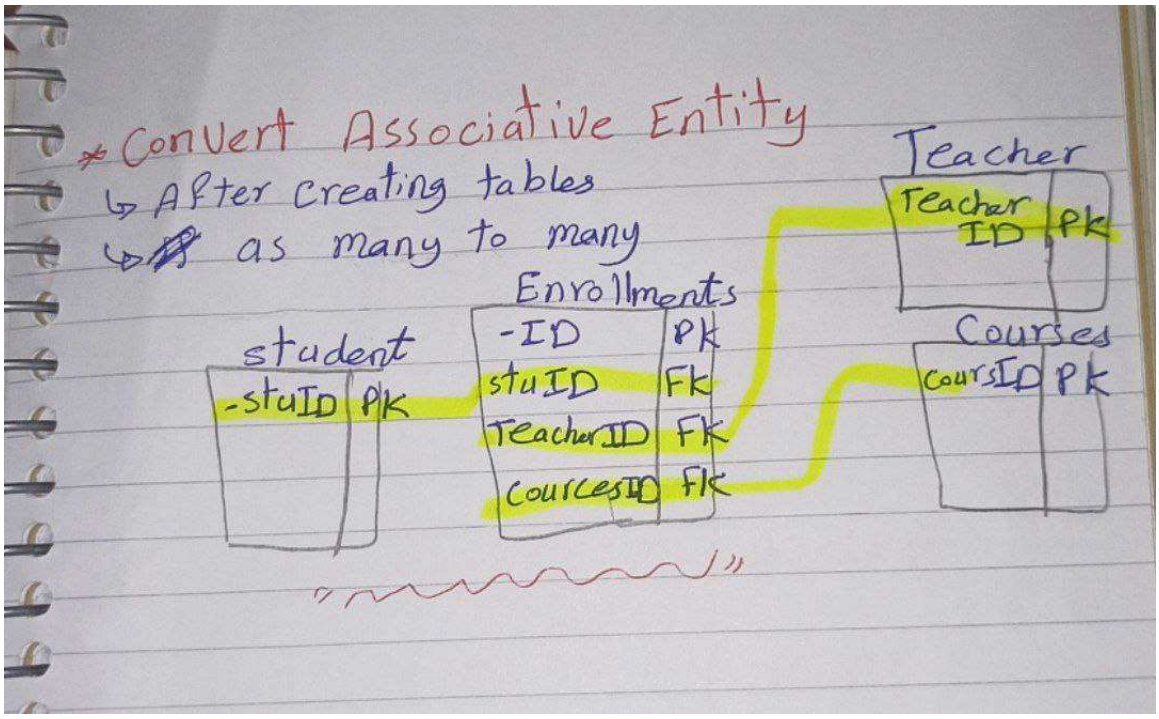
StudentID	FirstName	LastName	BirthDate
1	Mohammed	Abu-Hadhoud	6-11-1977
2	Ali	Amjad	12-3-2000
3	Maha	Omaran	11-6-2003
4	Fidaa	Safwan	6-6-1991

Enrollments

EnrollmentID	StudentID	CourseID	EnrollDate	Grade	TecherID
1	1	2	1/1/2000	95	1
2	1	3	20/1/2005	80	3
3	2	2	5/5/2022	50	2
4	3	4	15/1/2021	45	1
5	3	4	25/6/2022	Null	1

Courses

CourseID	Name	CreditHours
1	OOP	3
2	Database	3
3	C#	3
4	Marketing	1



مصطلحات مترجمة

#15 Dictionary

- maintain يحافظ على
- establishes يضع / يرسخ
- consistancy متناسقة
- appropriate مناسب
- Procedure إجراء
- Pictorialy تصوري
- systematic process تعبير على إنها خطوات منظمة
- Conceptualize تصور
- Rectangle ▭ مستطيل
- Diamonde ◇ معين
- ellipse شكل بيضاوي
- composite مركب
- Drived مشتقة
- reliance الإعتماد
- exhibils معارضة
- illustrate يوضح
- oval = ellipse ⬭
- depicts يصور
- associated مرتبط
- instances الحالات
- mandatory إلزامي

- associative ترابط
- aggregation تجميع
- extracting إستخراج
- approach نهج