**Database Workshop**

**Day1:**

* **DB Concept:**

In general, what is the idea of DB

* **DB Design:**

What is the benefit of good DB design 🡪 like building architecture

* **ERD:**
* What is ERD, why people think to draw ERD for DB
* Is it the actual DB?
* This ERD must go through mapping (DB mapping)🡪 set of rules apply it for ERD
* What is the aim of ERD?
* How can we deal with this table?
* SQL: to deal with DB
* DB engine & the language that we use
* **File based system :**
* What is File based system; question here should all software have DB in background?
* Delimited file and fixed width
* Disadvantage
* **DB system:**

**Type of users:**

* System analyst
* DB designer
* DB developer
* DB administrator
* App developer
* BI developer
* **DB Life cycle: same as SW engineering**
* Analysis
* DB design
* DB mapping
* DB implementation
* GUI application
* Client
* **Basic Definitions:**
* **Database:** collection of related data or tables
* **Database Management system (DBMS):** tool 🡪 SQL server to create DB 🡪 mdf (doc)
* **Database System:** SW + DB (interface)

**DB:** collection of tables, built table 🡪 very structural

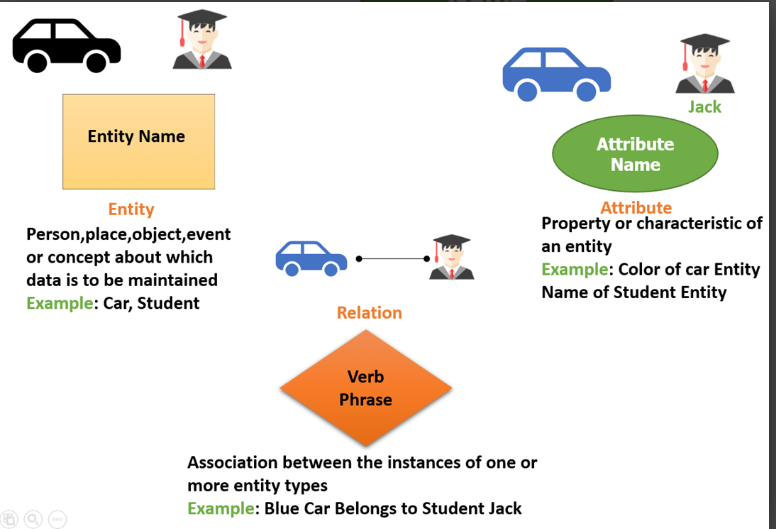
**Employee Department**

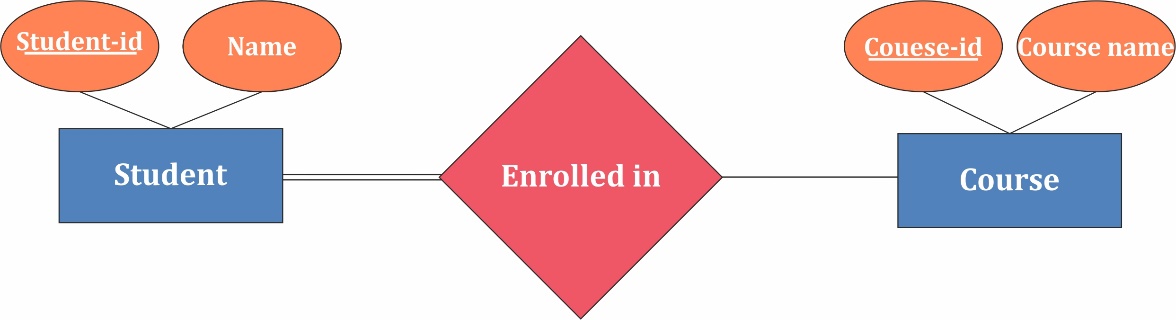
|  |  |  |
| --- | --- | --- |
| **Id** | **name** | **Dept-id** |
| **10** | **FATMA** | **1** |
| **20** | **Ali** | **2** |

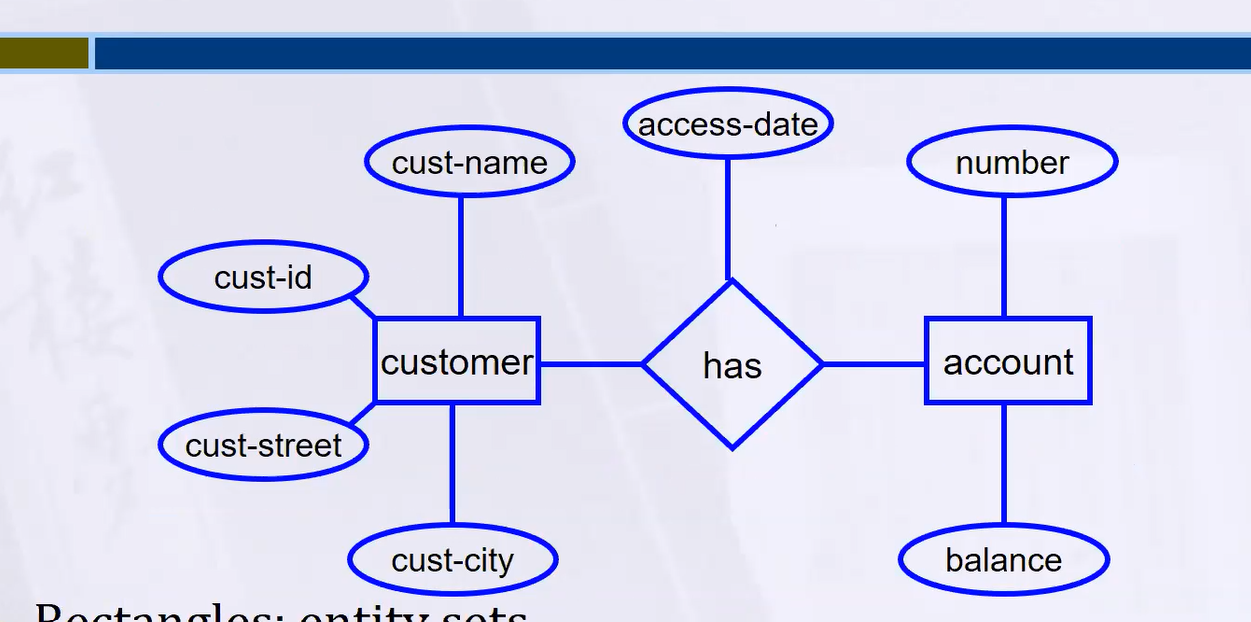
|  |  |  |
| --- | --- | --- |
| **Dept-Id** | **Name** |  |
| **1** |  |  |
| **2** |  |  |

* Its collection of rows and columns
* Every table should have PK ( UNIQE, NOT NULL)
* Every table related to each table so any data that we enter to the first table will affect the other table
* Every column should have data type
* **The ER Model**
* **What is an Entity Relationship Diagram (ERD)?**
* After gathering the requirements, we start converting these lines into a diagram.
* While you read the requirement lines, try to identify the main components.

* **Entities:** Any important objects inside the system, it could be:
* **person/role**: (e.g. Student, employee, Doctor, Teacher)
* **object:** (e.g. Invoice, Product, Vehicle, Book)
* **concept:** abstract ideas or categories important to a system (e.g. Profile or account)-> (username, password). (e.g. Permission)-> Represents access rights or privileges granted to users in a system (Attributes could include permission type (read, write, delete), description, and associated roles).
* **Event:** (e.g. Transaction)-> transaction ID, amount, date. (e.g. Appointment, Login Attempt, Payment, booking, Registration, sale )
* **Attributes:** column, an attribute is a property or characteristic of the entity that holds it
* **Relationship:** Links between the entities

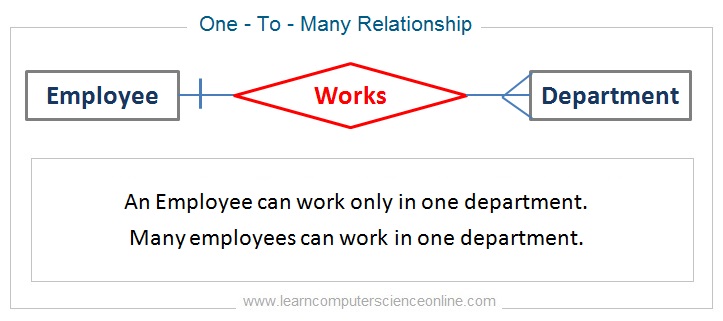




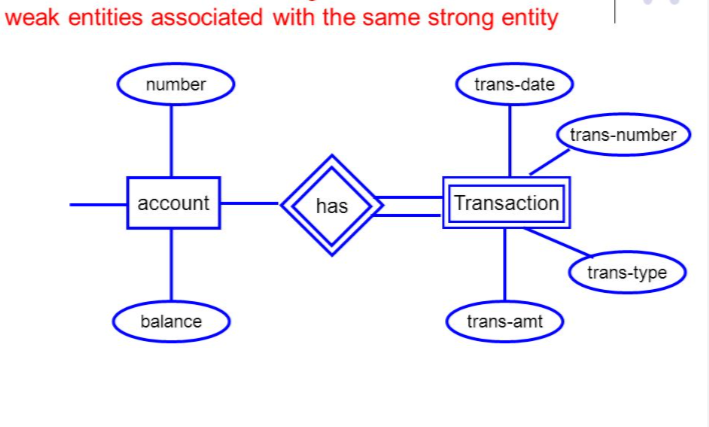


**We say that attribute is a property or characteristic of the entity, I can see in this diagram that we have attribute on the relationship!!**

* This attribute is shared among them.
* Same as (Student and Course)-> where can I put the grade of the student? its shared between them so we will write it on the relationship.
* Number of hours the employee worked on a specific project ( Employee and Project)-> number of hours is attribute on the relationship.
* **Let’s learn more about a Strong entity VS Weak Entity 😊**

****

* we have this example here, "If an employee is dismissed from a specific department, will that department be closed?" No !
* "Or vice versa, if I close a specific department, does that mean I am dismissing the employees?" No !
* **Yes, there is a relationship, but it hasn't reached the level where deleting a row from one table would affect the other table. So this is Strong entity (Employee, Department).**

****

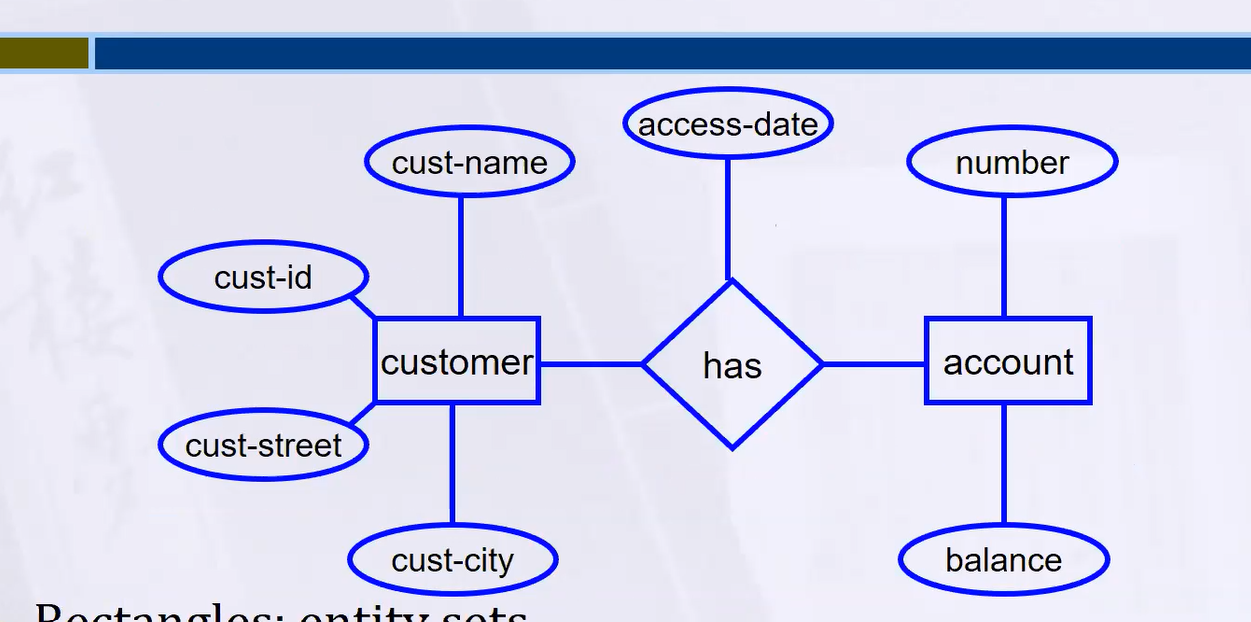
* **Weak Entities:** It is the Entity which existence in the system can disappear with the disappearance of another entity.
* **For example:** A person with their credit card uses it to withdraw and deposit money. If they go to the bank to cancel the credit card, is there a need to store the transactions that were made with the credit card? No! So, Transaction is weak entity
* **If the account disappears -> Transaction will disappear also.**
* **Another example:** Courses and the labs, if the courses disappeared the labs will automatically disappear.
* **Another example:** The employee's insurance is cancelled if the employee resigns

|  |  |
| --- | --- |
| Strong entity | Weak Entity |
| Strong entity set always has a primary key. | It does not have enough attributes to build a primary key. |
| It is represented by a rectangle symbol. | It is represented by a double rectangle symbol. |
| It contains a Primary key represented by the underline symbol. | It contains a Partial Key which is represented by a dashed underline symbol. |
| In a weak entity set, it is a combination of primary key (account number) and partial key (transaction number). |

* **Types of Attributes 😊**

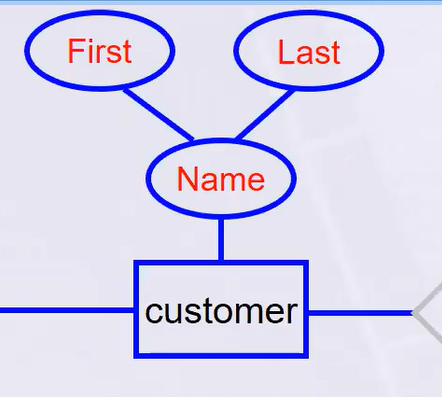
There are different types of attributes as discussed below:

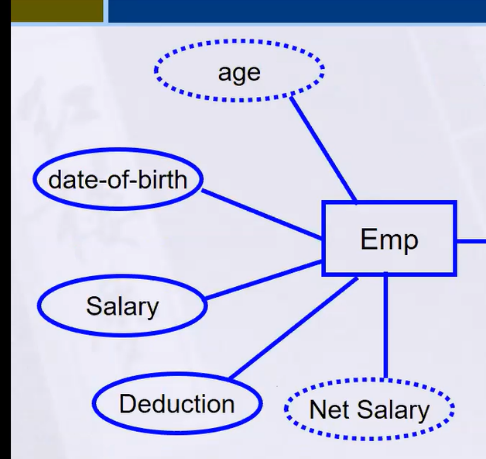
* **Composite Attribute**
* **Multi-Valued Attribute**
* **Derived Attribute**
* **Complex Attribute**
* **Simple Attribute**



* **Simple Attribute**

If this attribute is not divisible, not calculated at runtime, not repeated for the same person.

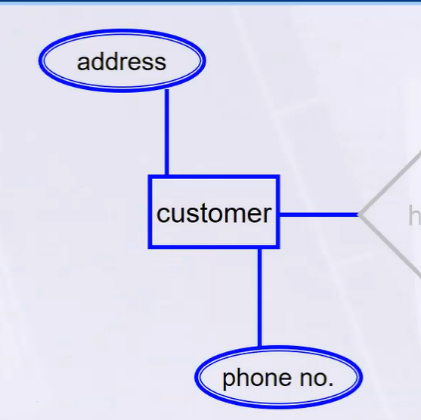
* **Example:** City,  roll number of a student, the ID number of an employee, gender, name )Is it possible to create an equation to calculate the name?(, but if mention (First, Last name)-> **Composite Attribute**



* **Derived Attribute**

Something calculated at runtime, something I can derive from another thing. And it is represented by dotted oval shape.

**Example:** We have birthday-> I can calculate age, Total and average marks of a student, age of an employee that is derived from date of birth.

* **Multi-Valued Attribute**

Something that is repeated for the same person.

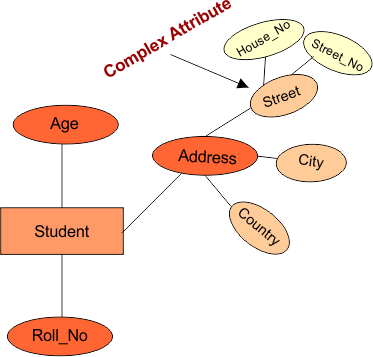
And it is represented by double oval shape.

**Example:**Phone number, more than one

* **Complex Attribute**

Multi-Valued + Composite

**Example:**Address because address contain composite value like street, city, state, PIN code and also multivalued because one people has more that one house address.

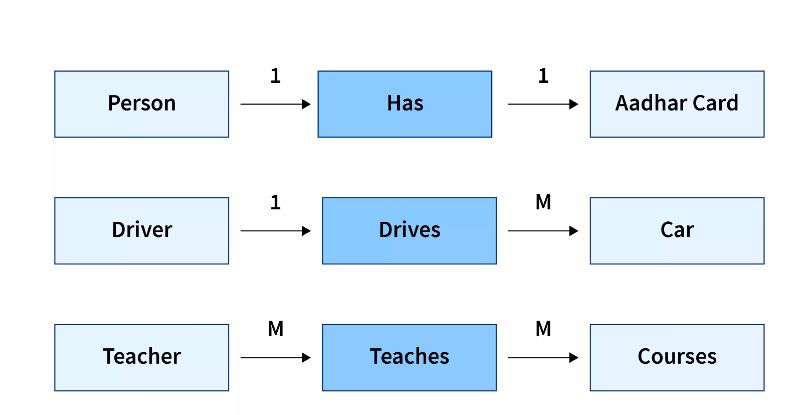


* **What Is Relationship?**

When we read the requirements document, we find nouns and verbs

**Nouns:** could be entity or attribute.

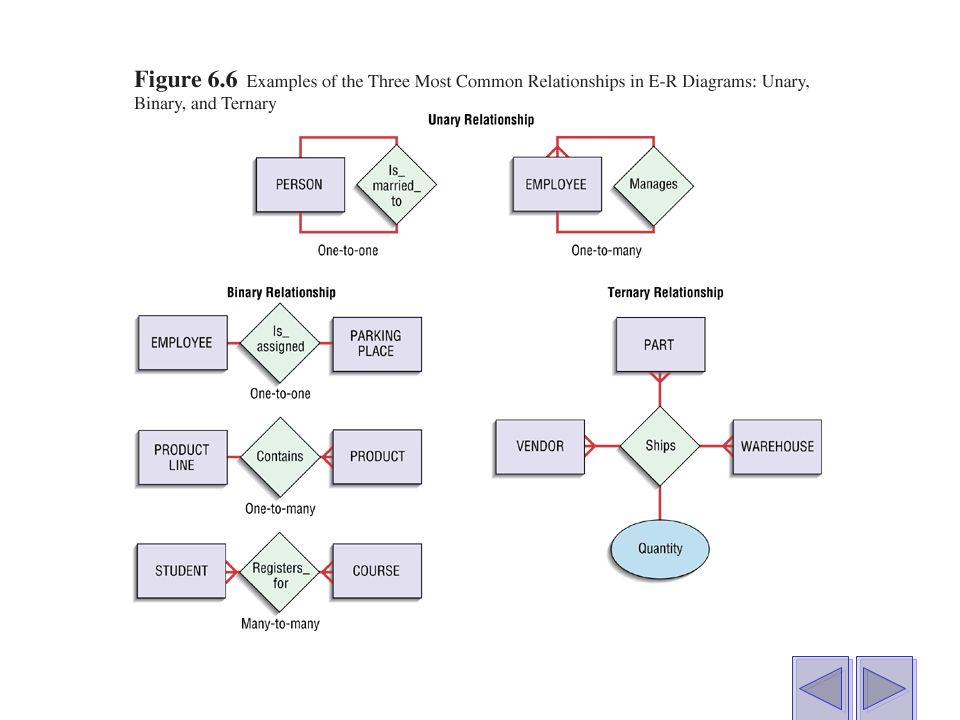
**Verbs:** Relationship.



* **What is the types Of Relationships? 😊**
* Degree of relationship
* Cardinality constraint
* Participation constraint

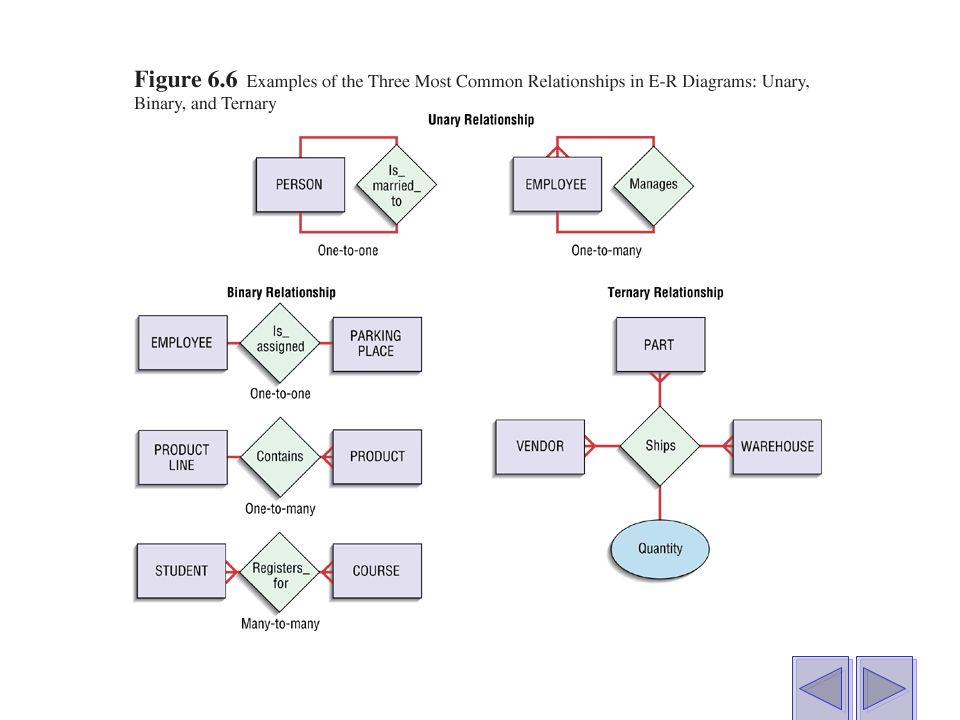
**Degree of relationship:**

**Degree:** number of entity types that participate in relationship.

**Three cases:**

* **Unary:** between 2 instance of one entity type (e.g. shared attribute Employee(id,name) mange Manger (id,name))=> employee could be manger also! So later on we can have one entity. Called ( Employee).
* **Binary:** between the instance of 2 entity type( The most common in databases(, no attribute is shared.(e.g. Employee and department).
* **Ternary:** among the instance of 3 entity type.

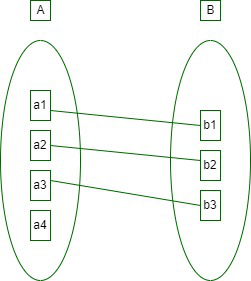
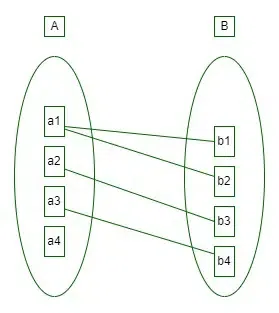
There is a specific attribute that combine three entities**. (e.g. Course teaches by number of teacher (Fatma and Karim 😊), Teacher, Student)=>**here Grade will appear but if we said that the course is teach by one teacher => this will be binary.

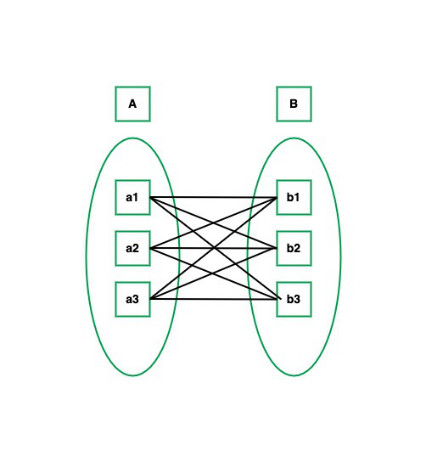


**Cardinality constraint:**

Each row in the first entity is linked to how many rows in the second entity.

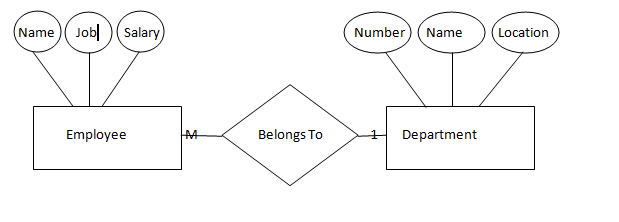
**One-to-One One-to-Many**

****

**Many-to-Many**

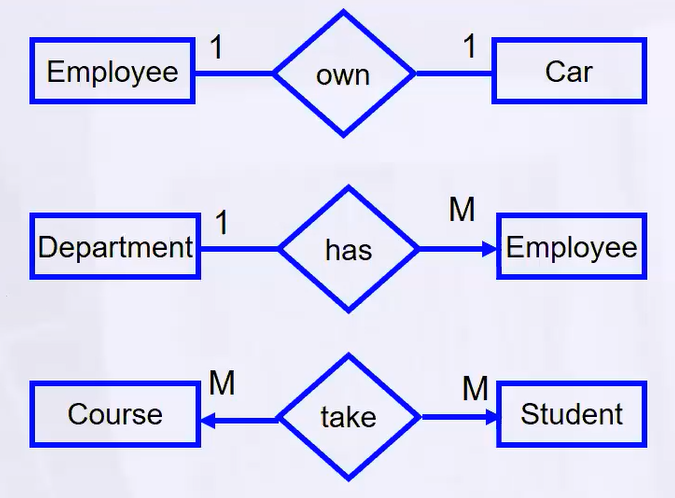
A diagram of a work flow

Description automatically generated

* **One-to-One**
* **One-to-Many**
* **Many-to-Many**

1 1

mange



**Note:** All the cardinality depends on the business requirement. If its not appear in the requirement, ask the system analyst => ***we can ask our specialist Kaim*** 😊 Do not assume the logic by yourself!

**Participation constraint:**

* Are all the rows involved in the relationship or not?
* **For example:** Do all employees must to have a department?
* Optional (may, some)= one line
* Mandatory (must) = 2 lines

**A diagram of a diagram of a number of objects

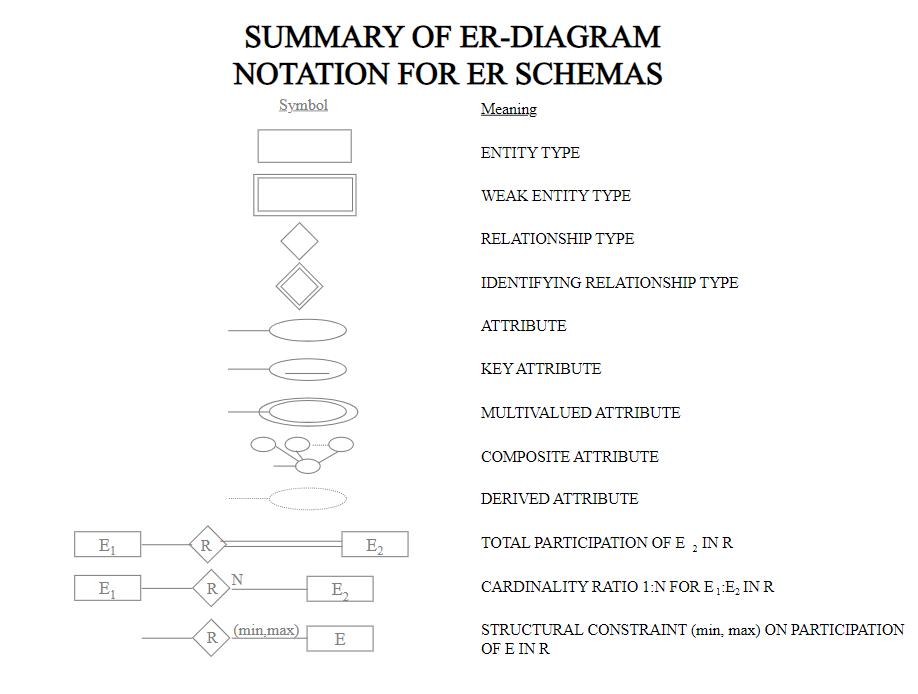
Description automatically generated with medium confidenceTotal Participation Partial Participation**

**A diagram of a number of objects

Description automatically generated**



* **Keys**:
* **Primary Key**
* **Candidate Key**
* **Foreign Key (Tomorrow, we will explain it 😊)**
* **Composite Key**
* **Partial Key => for the weak entity**
* **Super Key**
* **Alternate Key**
* **Artificial Key … your job is to search about it there are more than this ! 😊**
* There are many attributes, and you want to choose the PK, before you choose the PK you must first determine candidate Key.
* **What is candidate key? the** one that could be PK you choose the suitable one.
* **ID => PK** with underline.
* Assume that I don’t have ID , so we will go with composite PK (name, address together ) اعمل تبادل وتوافيق بينهم للاختيار😊
* If all the composite is not suitable then we set one PK.

****

* **Case study**

Day2:

**Recap of what we discussed in day 1**

**Relational DB definition (Table or entity, attribute or column or filed, Row or Record or tuble, DB)**

**Mapping -> DB schema**

**Mapping rule**

**Case study**

**ANSI SQL**

**Create DB (SQL server)**