Introduction to ER Diagram (Entity-Relationship Diagram)

An **Entity-Relationship (ER) Diagram** is a visual representation of the entities within a database and the relationships between them. It is a conceptual design tool used to model the structure of a database before it is implemented physically in a Database Management System (DBMS). The ER diagram helps database designers to visually map out the system's data, making it easier to understand the system's architecture and structure.

Key Components of an ER Diagram:

1. **Entities**:
    - An **entity** is any object or concept that has a distinct existence in the database and can be represented as a set of data.
    - Examples of entities: `Customer`, `Order`, `Product`, `Employee`, etc.
    - Entities are represented by **rectangles** in an ER diagram.
2. **Attributes**:
    - An **attribute** is a property or characteristic of an entity.
    - Examples of attributes: `Customer Name`, `Order Date`, `Employee Address`, etc.
    - Attributes are represented by **ellipses** connected to their corresponding entity.
3. **Primary Key**:
    - A **primary key** is an attribute (or set of attributes) that uniquely identifies an entity in the database.
    - In the ER diagram, the primary key is often underlined.
4. **Relationships**:
    - A **relationship** represents an association between two or more entities.
    - Examples of relationships: `Places`, `Purchases`, `EmployedBy`.
    - Relationships are represented by **diamonds** in an ER diagram, with lines connecting them to the entities involved.
5. **Cardinality**:
    - **Cardinality** defines the number of instances of one entity that can or must be associated with each instance of another entity.
    - Types of cardinality:
        - **One-to-One (1:1)**: An instance of one entity is associated with only one instance of another entity.
        - **One-to-Many (1:M)**: An instance of one entity can be associated with many instances of another entity.

        - **Many-to-Many (M:N)**: Instances of one entity can be associated with many instances of another entity.
6. **Weak Entities**:
    - A **weak entity** is an entity that cannot be uniquely identified by its own attributes alone and relies on another entity (called the **owner entity**) to provide a unique identification.
    - Weak entities are represented by **double rectangles** and the relationship with the owner entity is represented by a **double diamond**.
7. **Multi-valued Attributes**:
    - A **multi-valued attribute** is an attribute that can have multiple values for a single entity.
    - For example, an employee may have multiple phone numbers.
    - Multi-valued attributes are represented by **double ellipses**.
8. **Derived Attributes**:
    - A **derived attribute** is an attribute whose value can be derived from other attributes in the database.
    - For example, the `Age` of a person could be derived from the `Date of Birth`.
    - Derived attributes are represented by **dashed ellipses**.

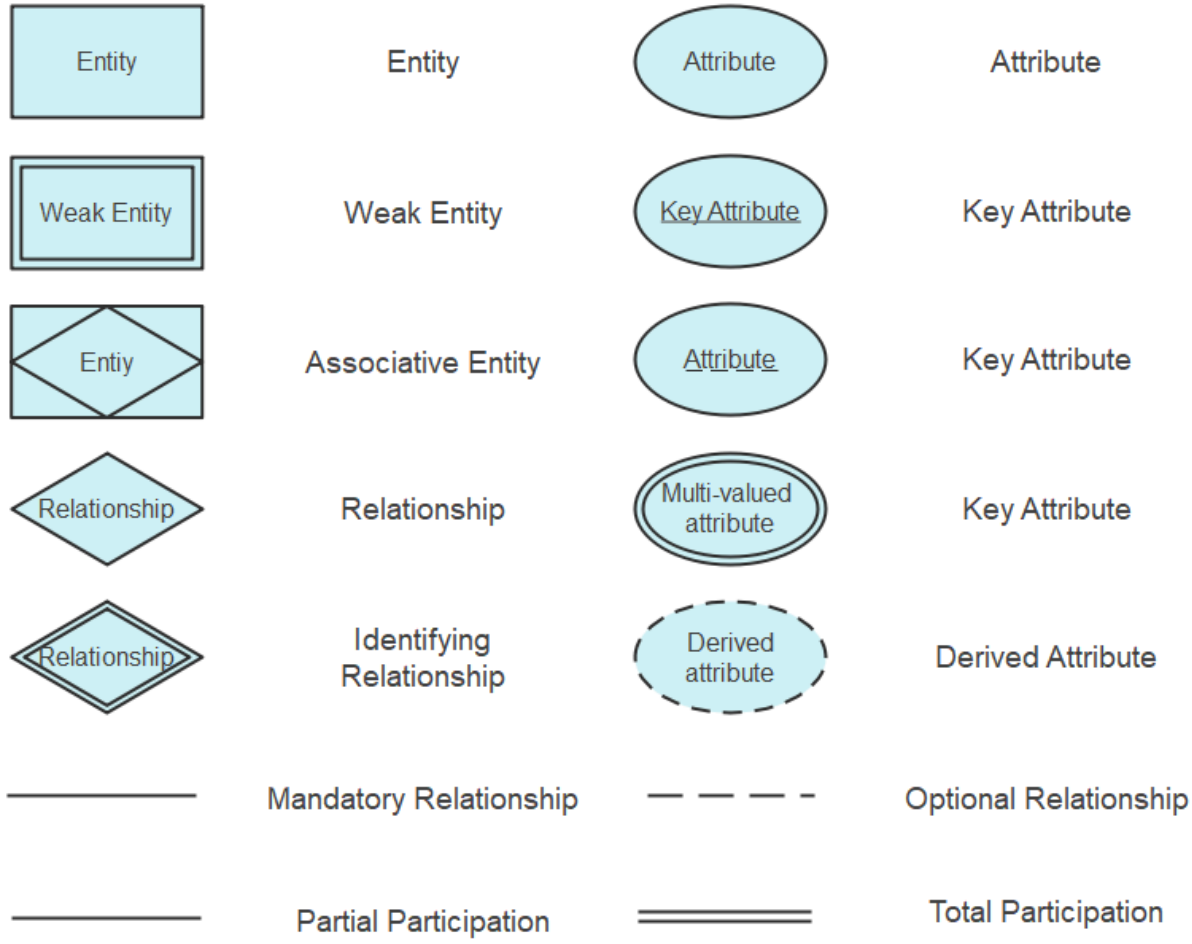
Advantages of ER Diagrams:

1. **Clear Visualization**: Provides a clear and concise visual representation of the database structure.
2. **Easy Communication**: Helps communicate the design between stakeholders, developers, and database administrators.
3. **Simplifies Design Process**: Simplifies the process of translating business requirements into database structures.
4. **Foundation for Implementation**: Acts as a blueprint for creating the physical database schema.

Conclusion:

ER Diagrams are essential in the initial stages of database design, helping to conceptualize the database's structure and identify potential design issues before implementation. They provide a clear and understandable way of organizing and presenting data, which is crucial for building efficient and maintainable databases.

Symbolic Representation:



Mapping Cardinality in ER Diagram

Mapping Cardinality (also known as cardinality constraint) refers to the number of occurrences of one entity that are related to the number of occurrences of another entity in a relationship in an Entity-Relationship (ER) diagram. It specifies how many instances of one entity can be associated with instances of another entity.

Types of Mapping Cardinality

In an ER diagram, mapping cardinality is typically represented as one of the following four types:

1. One-to-One (1:1)
2. One-to-Many (1:N)
3. Many-to-One (N:1)
4. Many-to-Many (M:N)

Let's discuss each one in detail with examples:

---

1. One-to-One (1:1)

A one-to-one relationship means that an entity from the first set can be associated with at most one entity from the second set, and vice versa.

Example: Consider the relationship between Employee and Office. If each employee is assigned to exactly one office, and each office is assigned to exactly one employee, then this is a one-to-one relationship.

ER Diagram Representation:

Employee (1) <------> (1) Office

- Each Employee is associated with exactly one Office.
- Each Office is assigned to exactly one Employee.

2. One-to-Many (1:N)

A one-to-many relationship means that an entity from the first set can be associated with many entities from the second set, but an entity from the second set is associated with only one entity from the first set.

Example: Consider the relationship between Department and Employee. A department can have many employees, but each employee belongs to exactly one department.

ER Diagram Representation:

Department (1) <------> (N) Employee

- One Department can have many Employees.
- Each Employee belongs to exactly one Department.

3. Many-to-One (N:1)

A many-to-one relationship is essentially the reverse of a one-to-many relationship. Here, multiple instances of the first entity are associated with exactly one instance of the second entity.

Example: Consider the relationship between Order and Customer. Multiple orders can be placed by a single customer, but each order is placed by only one customer.

ER Diagram Representation:

Order (N) <------> (1) Customer

- Many Orders can belong to one Customer.
- Each Order is placed by exactly one Customer.

4. Many-to-Many (M:N)

A many-to-many relationship means that entities from both sets can be associated with many entities from the other set.

Example: Consider the relationship between Student and Course. A student can enroll in multiple courses, and each course can have many students enrolled.

ER Diagram Representation:

Student (M) <------> (N) Course

- Many Students can enroll in many Courses.
- Each Course can have many Students.

Visual Representation in ER Diagram:

- 1:1 is typically represented by a line with a 1 on both ends.
- 1:N is represented by a line with 1 on one side and N (or ∞) on the other side.
- N:1 is just the reverse of 1:N, with N on one side and 1 on the other.
- M:N is represented with M on one side and N (or ∞) on the other side, indicating a many-to-many relationship.

Examples of Mapping Cardinality:

1. One-to-One (1:1):
    - Example: Person and Passport. A person can have only one passport, and a passport can be issued to only one person.

ER Diagram:
Person (1) <------> (1) Passport

    -
2. One-to-Many (1:N):
    - Example: Teacher and Student. One teacher can teach many students, but each student has only one teacher.

ER Diagram:
Teacher (1) <------> (N) Student

    -
3. Many-to-One (N:1):
    - Example: Book and Author. Many books can be written by a single author, but each book has only one author.

ER Diagram:
Book (N) <------> (1) Author

    -
4. Many-to-Many (M:N):
    - Example: Student and Course. A student can enroll in many courses, and a course can have many students.

ER Diagram:
Student (M) <------> (N) Course

-

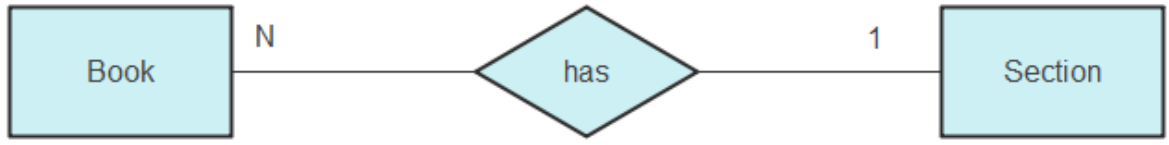
Summary:

- Mapping Cardinality defines the relationship between entities in terms of how many instances of one entity are associated with how many instances of another entity.
- The types of mapping cardinality are One-to-One (1:1), One-to-Many (1:N), Many-to-One (N:1), and Many-to-Many (M:N).
- Cardinality helps to represent real-world relationships accurately in an ER diagram and ensures that data integrity and consistency are maintained in the database design.

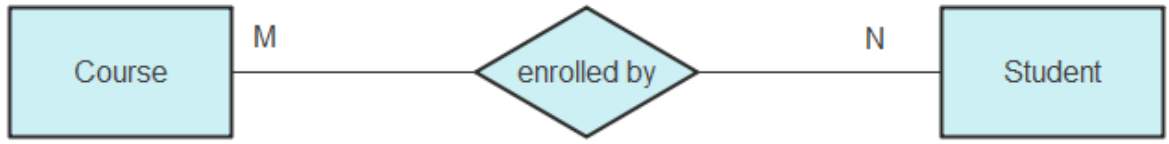
one-to-one (1:1)

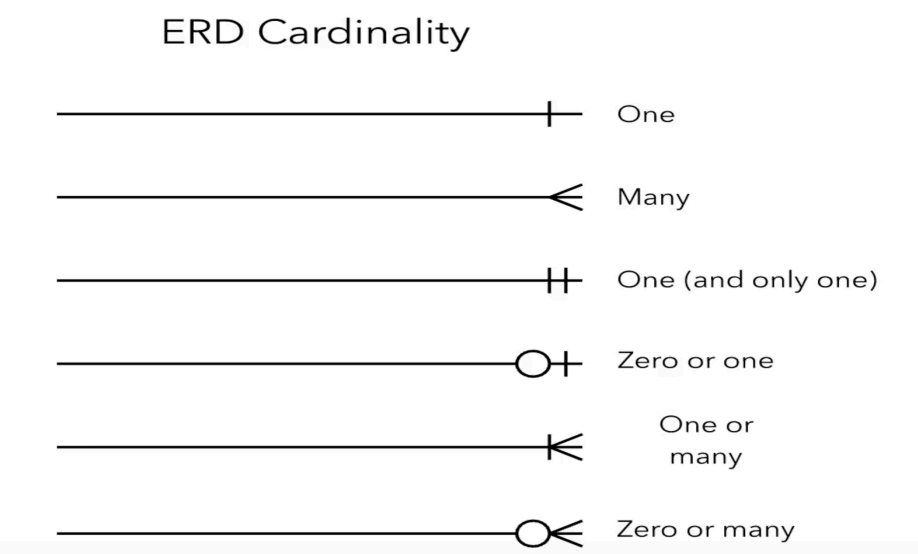
one-to-many (1:N)

many-to-one (N:1)

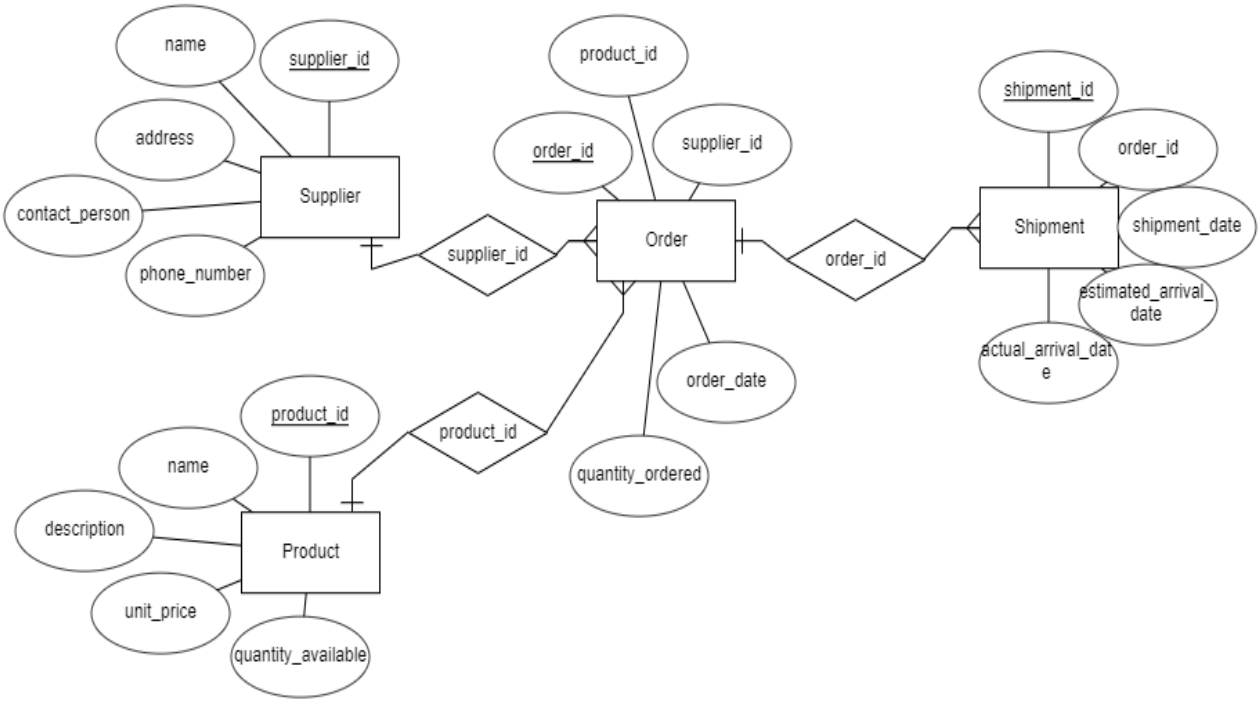


many-to-many (M:N)





Example:



ER Diagram for Supply Chain Management

Examples:

Entities and Attributes of the Supply Chain Management

1. Supplier: It Represent Providing goods or services.

- supplier_id (Primary Key): Unique identifier for each supplier.
- name: Name of the supplier.
- address: Address of the supplier.
- contact_person: Name of the contact person at the supplier.
- phone_number: Phone number of the supplier.

2. Product: It Represent Goods or services offered.

- product_id (Primary Key): Unique identifier for each product.
- name: Name of the product.
- description: Description of the product.
- unit_price: Price per unit of the product.
- quantity_available: Quantity of the product available in inventory.

3. Order: It Represent Requests for products or services.

- order_id (Primary Key): Unique identifier for each order.
- product_id (Foreign Key referencing Product): Identifier of the product ordered.
- supplier_id (Foreign Key referencing Supplier): Identifier of the supplier from whom the product is ordered.
- order_date: Date when the order was placed.
- quantity_ordered: Quantity of the product ordered.

4. Shipment: It Represent Movement of products.

- shipment_id (Primary Key): Unique identifier for each shipment.
- order_id (Foreign Key referencing Order): Identifier of the order associated with the shipment.
- shipment_date: Date when the shipment was sent.
- estimated_arrival_date: Estimated arrival date of the shipment.
- actual_arrival_date: Actual arrival date of the shipment.

Relationships Between These Entities

1. Supplier - Product Relationship

- **One-to-many relationship: Each supplier can supply multiple products.**
- **Foreign key: supplier_id in Product table referencing supplier_id in Supplier table.**

2. Product - Order Relationship

- **One-to-many relationship: Each product can be ordered multiple times.**
- **Foreign key: product_id in Order table referencing product_id in Product table.**

3. Order - Shipment Relationship

- **One-to-one relationship: Each order can have one shipment.**
- **Foreign key: order_id in Shipment table referencing order_id in Order table.**