Entity: An **entity** is a thing or object in the real world with an independent existence. An entity may be

- **Concrete entity:** An object with a **physical existence**. **For example:** a particular person, car, house, or employee.
- **Abstract entity:** An object with a **conceptual existence**. **For example:** a company, a job, or a university course.

An entity has a set of properties, and the values for some set of properties must uniquely identify an entity.

Entity Set: An entity set is a set of entities of the same type that share the same properties, or attributes. **For example:** The set of all people who are instructors at a given university, can be defined as the entity set instructor.

Extension of the entity set: In the process of modeling, the term **entity set** is used in abstract, without referring to a particular set of individual entities. **Extension of the entity set** refer to the actual collection of entities belonging to the entity set. **For example:** The set of actual instructors in the university forms the extension of the entity set instructor. This distinction is similar to the difference between a **relation** and a **relation instance**.

Entity sets do not need to be disjoint. For example, it is possible to define the entity set **person** consisting of all people in a university. A **person** entity may be an **instructor** entity, a student **entity**, both, or neither.

Attribute: An attribute is a characteristic or property of an entity. Each entity has attributes – the particular properties that describe it.

1. **Composite versus Simple (Atomic) Attributes:**
   - Attributes that are not divisible are called **simple** or **atomic** attributes.
   - Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings.
   - Composite attributes help us to group together related attributes, making the modeling cleaner.
   - Composite attributes can form a hierarchy.

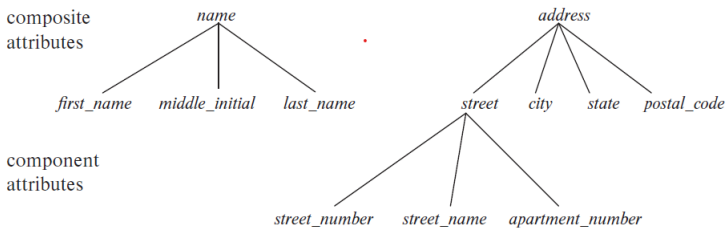

Figure 1: Composite attributes *name* and *address*.

2. **Single-Valued versus Multivalued Attributes:**
   - A single-valued attribute is an attribute that can have only one value for a given entity.
   - Multi-valued attribute is an attribute that can have multiple values for a given entity.
   - A multivalued attribute may have lower and upper bounds to constrain the number of values allowed for each individual entity. **For example:** the Colors attribute of a car may be restricted to have between one and five values.
3. **Stored versus Derived Attributes:**
   - A stored attribute is an attribute that is explicitly stored in a database.
   - A derived attribute is an attribute that is not explicitly stored in a database but is derived from other stored attributes. The value of a derived attribute is computed based on the value of one or more stored attributes.
   - Some attribute values can be derived from related entities; for example, an attribute `Number_of_employees` of a DEPARTMENT entity can be derived by counting the number of employees related to (working for) that department.

An **entity set** is represented in an E-R diagram by a rectangle, which is divided into two parts.

- First part contains the name of the entity set
- Second part contains the names of all the attributes of the entity set

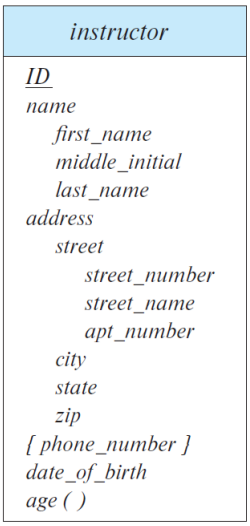

Figure 2: E-R diagram with composite, multivalued, and derived attributes. `name` and `address` are composite attribute. `phone_number` is a multivalued attribute and `age` is a derived attribute.

Domain, or Value set of Attribute: For each attribute (simple), there is a set of permitted values, called the domain, or value set, of that attribute. **For example:** The domain of attribute `semester` might be strings from the set {`Fall`, `Winter`, `Spring`, `Summer`}.

Key Attribute: An entity set usually has one or more attributes whose values are distinct for each individual entity in the entity set. Such an attribute is called a **key attribute**, and its values can be used to identify each entity uniquely.

Relationship: A relationship is an association among several entities. It means multiple entities can participate in a relationship.

Relationship Set: A relationship set is a set of relationships of the same type.

A **relationship set** is represented in an E-R diagram by a diamond, which is linked via lines to a number of different entity sets (rectangles).

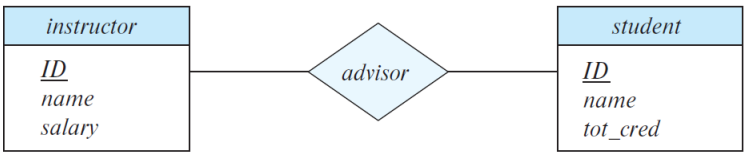

Figure 3: E-R diagram showing relationship set **advisor** between entity sets **instructor** and **student**.

Relationship Instance: A relationship instance in an E-R schema represents an association between the named entities in the real-world enterprise that is being modeled.

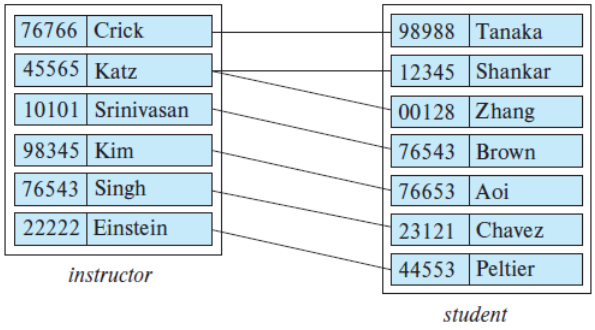

Figure 4: Relationship set **advisor** (only some attributes of **instructor** and **student** are shown).

Formally, a **relationship set** is a mathematical relation on $n \geq 2$ (possibly nondistinct) entity sets. If $E_1, E_2, \ldots, E_n$ are entity sets, then a relationship set R is a subset of

$$\{(e_1, e_2, \ldots, e_n) \,|e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$$

where $(e_1, e_2, \ldots, e_n)$ is a relationship instance.

Participation: The association between entity sets is referred to as participation; *i.e.*, the entity sets $E_1, E_2, \ldots, E_n$ participate in relationship set R.

Degree of Relationship Set: The number of entity sets that participate in a relationship set is the degree of the relationship set. A binary relationship set is of degree 2; a ternary relationship set is of degree 3.

Role of Entity: The function that an entity plays in a relationship is called that entity's role.

- **Entity sets** participating in a **relationship set** are generally distinct, roles are **implicit** and are not usually specified.
- Roles are useful when the meaning of a relationship needs clarification.
  - When the entity sets of a relationship set are not distinct; that is, the same entity set participates in a relationship set more than once, in different roles.

Recursive Relationship: A recursive relationship refers to a relationship between an entity and itself. In this type of relationship, explicit role names are necessary to specify how an entity participates in the relationship.

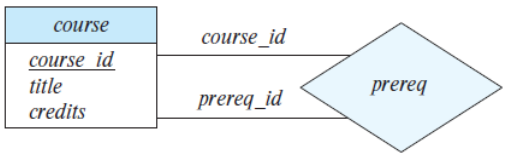

Figure 5: E-R diagram with role indicators.

Descriptive Attributes: A relationship may also have attributes called descriptive attributes.

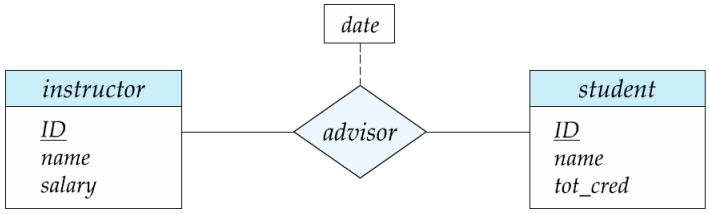

Figure 6: E-R diagram showing descriptive attributes.

It is possible to have more than one relationship set involving the same entity sets.

- An relationship set **advisor** between entity set **instructor** and **student**.
- An relationship set **teaches** between entity set **instructor** and **student**.

Mapping cardinalities (Cardinality Ratio): It express the number of entities to which another entity can be associated via a relationship set.

For a binary relationship set R between entity sets A and B, the mapping cardinality must be one of the following:

1. **One-to-one:** An entity in A is associated with **at most one** entity in B, and an entity in B is associated with **at most one** entity in A.
2. **One-to-many:** An entity in A is associated with any number **(zero or more)** of entities in B. An entity in B, however, can be associated with **at most one** entity in A.
3. **Many-to-one:** An entity in A is associated with **at most one** entity in B An entity in B however, can be associated with any number **(zero or more)** of entities in A.
4. **Many-to-many:** An entity in A is associated with any number **(zero or more)** of entities in B, and an entity in B is associated with any number **(zero or more)** of entities in A.

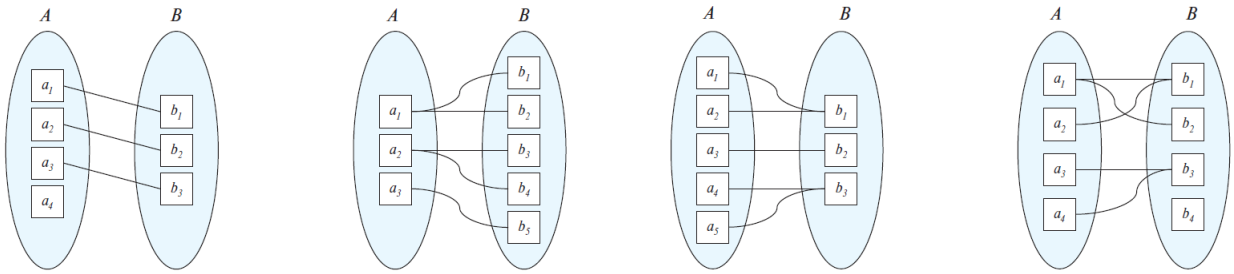

Figure 7: Mapping cardinalities. (a) One-to-one. (b) One-to-many. (c) Many-to-one. (d) Many-to-many.

Participation Constraints: This constraint specifies the minimum number of relationship instances that each entity can participate in and is sometimes called the **minimum cardinality constraint**. There are two types of participation constraints – total and partial.

1. **Total Participation:** The participation of an entity set E in a relationship set R is said to be **total** if every entity in E must participate in at least one relationship in R. Total participation is also called **existence dependency**. Total participation of an **entity** in a **relationship set** is indicated using double lines.
2. **Partial Participation:** A participation that is not total is said to be **partial**.

- **Scenario-1:** Every student to have at least one advisor.
  - The participation of **student** in the relationship set **advisor** is **total**.
- **Scenario-2:** An instructor need not advise any student.

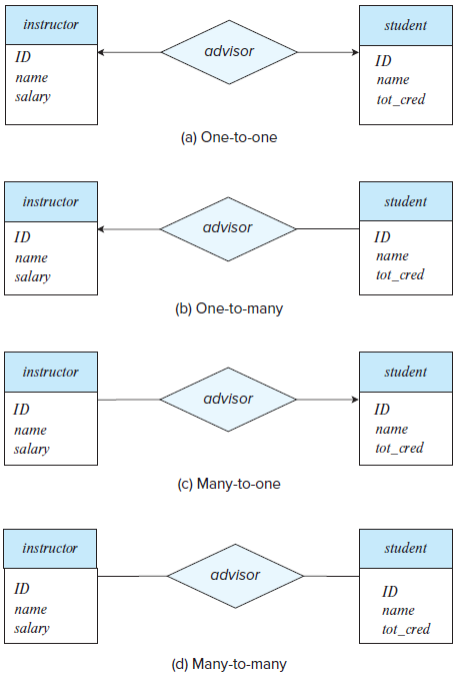

Figure 8: Relationship cardinalities.

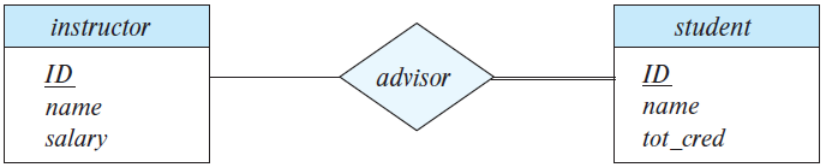

Figure 9: E-R diagram showing total participation.

– The participation of **instructor** in the relationship set **advisor partial**.

E-R diagrams also provide a way to indicate more **complex constraints** on the **number of times** each entity **participates** in relationships in a relationship set.

- A line may have an associated minimum and maximum cardinality shown in the form $l..h$, where l is the minimum and h the maximum cardinality.
- A minimum value of 1 indicates total participation of the entity set in the relationship set.
- A maximum value of 1 indicates that the entity participates in at most one relationship, while a maximum value $*$ indicates no limit.

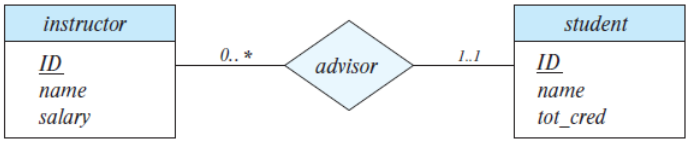

Figure 10: E-R diagram showing Cardinality limits on relationship sets.

Insight from Figure 10: The line between advisor and student has a cardinality constraint of 1..1, meaning the minimum and the maximum cardinality are both 1. It means each student must have exactly one advisor. The line between advisor and instructor has a cardinality constraint of 0..$*$, meaning an instructor can have zero or more

students. Thus, the relationship **advisor** is **one-to-many** from **instructor** to **student**, and the participation of **student** in **advisor** is **total**.

Let R be a relationship set involving entity sets $E_1, E_2, \ldots, E_n$. Let *primary-key*(E_i) denote the set of attributes that forms the primary key for entity set E_i. Assume that the attribute names of all primary keys are unique. The composition of the primary key for a relationship set depends on the set of attributes associated with the relationship set R.

- If the relationship set R has no attributes associated with it, then the following set of attributes describes an individual relationship in set R.

$$\textit{primary-key}(E_1) \cup \textit{primary-key}(E_2) \cup \cdots \cup \textit{primary-key}(E_n)$$

- If the relationship set R has attributes $a_1, a_2, \ldots, a_m$ associated with it, then the following set of attributes describes an individual relationship in set R.

$$\textit{primary-key}(E_1) \cup \textit{primary-key}(E_2) \cup \cdots \cup \textit{primary-key}(E_n) \cup \{a_1, a_2, \ldots, a_m\}$$

A **relationship set** is a set of relationship instances, and each instance is uniquely identified by the entities that participate in it. Thus, in both of the preceding cases, the following set of attributes forms a superkey for the relationship set.

$$\textit{primary-key}(E_1) \cup \textit{primary-key}(E_2) \cup \cdots \cup \textit{primary-key}(E_n)$$

The choice of the primary key for a binary relationship set depends on the mapping cardinality of the relationship set.

- **For one-to-one relationships**, the primary key of **either one of the participating entity sets** forms a minimal superkey, and either one can be chosen as the primary key of the relationship set.
  - An **instructor** can advise only one **student**. Each **student** can be advised by only one **instructor**.
  - The primary key of either **student** or **instructor** can be chosen as the primary key for **advisor**.
- **For one-to-many and many-to-one relationships**, the primary key of the **"many" side** is a minimal superkey and is used as the primary key of the relationship set.
  - **One-to-many:**
    - An **instructor** can advise many **student**s. Each **student** can be advised by only one **instructor**.
    - The primary key of **student** can be chosen as the primary key for **advisor**.
  - **Many-to-one:**
    - An **instructor** can advise only one **student**s. A **student** can be advised by many **instructor**s.
    - The primary key of **instructor** can be chosen as the primary key for **advisor**.
- **For many-to-many relationships**, the primary key of **participating entity sets** forms a minimal superkey and is chosen as the primary key of the relationship set.
  - An **instructor** can advise many **student**s. A **student** can be advised by many **instructor**s.
  - The primary key of **instructor** and **student** together can be chosen as the primary key for **advisor**.

Weak Entity Set: A weak entity set is one whose existence is dependent on another entity set, called its **identifying entity set**. Instead of associating a primary key with a weak entity, we use the primary key of the identifying entity, along with extra attributes, called **discriminator** attributes to uniquely identify a weak entity.

Strong Entity Set: An entity set that is not a weak entity set is termed a strong entity set.

Every **weak entity** must be associated with an **identifying entity**; that is, the weak entity set is said to be **existence dependent** on the **identifying entity set**. The **identifying entity set** is said to own the **weak entity set** that it identifies.

Identifying Relationship: The relationship associating the **weak entity set** with the **identifying entity set** is called the **identifying relationship**.

- The identifying relationship is many-to-one from the **weak entity set** to the **identifying entity set**, and the **participation** of the **weak entity set** in the relationship is **total**.

A **weak entity** type always has a **total participation** constraint (existence dependency) with respect to its identifying relationship because a weak entity cannot be identified without an owner entity. However, not every existence dependency results in a weak entity type. For example, a **DRIVER_LICENSE** entity cannot exist unless it is related to a **PERSON** entity, even though it has its own key (`License_number`) and hence is not a weak entity.

Weak Entity: child entity / subordinate entity
Identifying Entity: owner entity / parent entity / dominant entity

Partial Key (Discriminator): A weak entity normally has a partial key (or discriminator), which is the attribute that can uniquely identify weak entities that are related to the same owner entity.

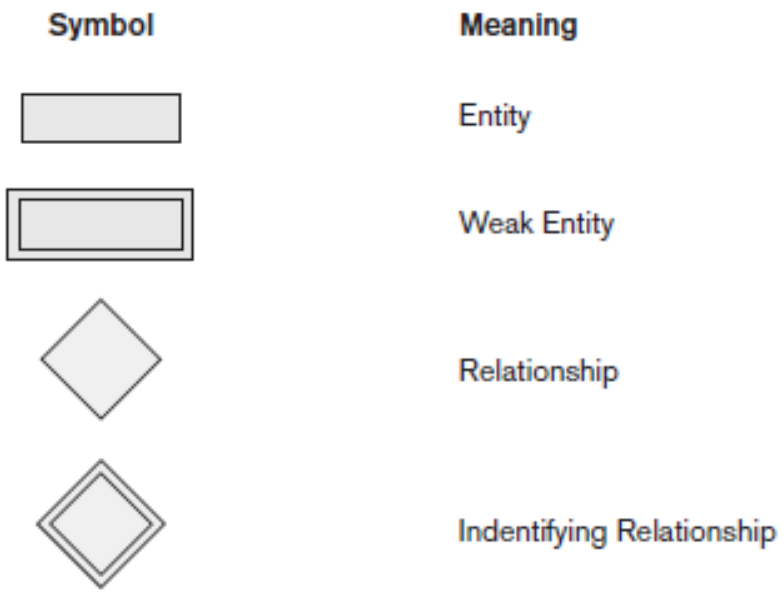

Figure 11: Notion of E-R diagram.

(a) Simple Attribute (b) Composite Attribute (c) Multivalued Attribute (d) Derived Attribute

Figure 12: Attribute Representation in E-R Diagram.