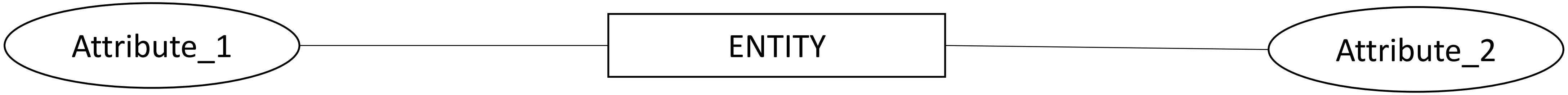
EE4202 Database Systems

HIGH-LEVEL (ENTITY RELATIONSHIP) DATABASE MODEL

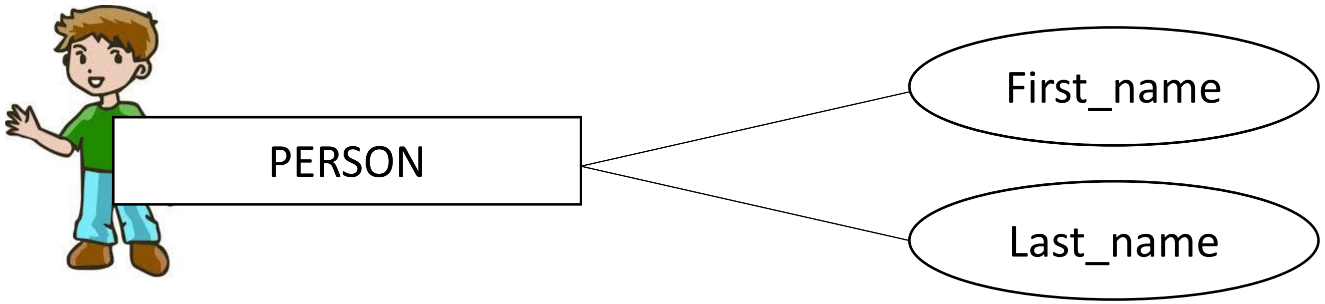
ENTITIES AND ATTRIBUTES

- Entity Relationship (ER) model is a high-level database model used in conceptual schema of database design.
- **Entities** are specific objects or things in the real-world that are represented in the database. E.g. EMPLOYEE John Silva, Research DEPARTMENT, PROJECT
- **Attributes** are properties used to describe an entity. E.g. an EMPLOYEE entity may have a Name, NIC, Address, Gender, Birth date
- A specific entity will have a value for each of its attributes. E.g. a specific employee entity may have Name='John Silva', NIC=‘65456789', Address ='731, First lane, Hapugala, Galle’, Gender='M', Birth date='09-JAN-65‘
- Each attribute has a value set(or data type) associated with it. E.g. integer, string, …
- An entity is represented by a rectangle and an attribute is represented by an oval in ER diagrams. Convention is to write entity name with singular word with all capitals.



ATTRIBUTES

- An attribute that cannot be further divided is an **atomic attribute** represented by a single line oval in ER diagrams**.** E.g. attribute *First_name* of person entity type. Convention is to write first letter capital

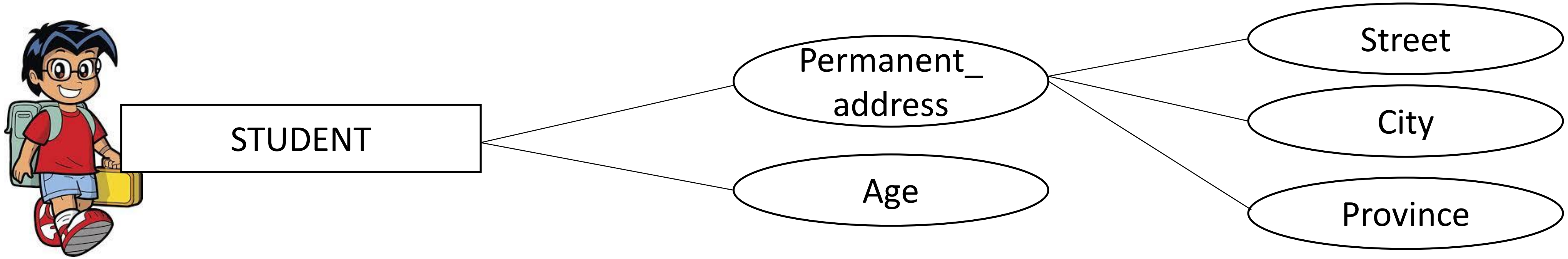


PERSON

First_name

Last_name

- An attribute which can be further divided into several attribute is known as a **composite_attribute.** E.g. attribute *Permanent_address* of student entity type is a composite attribute which can be subdivided into street, city, province. A composite attribute can be subdivided in an ER diagram.



STUDENT

Permanent_ address

Age

Street

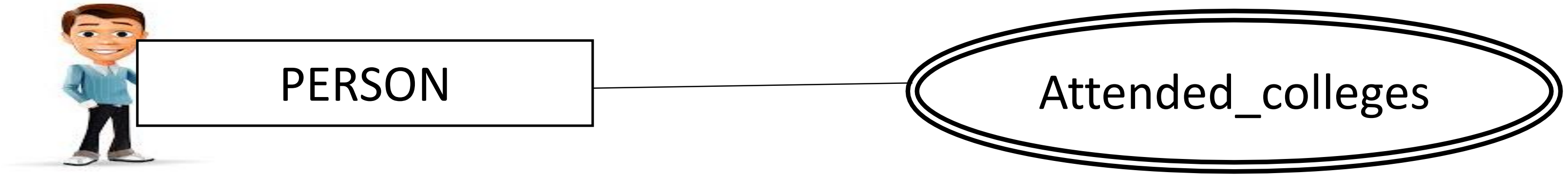
City

Province

- Components of a composite attribute are written within ().
- E.g. (43 main street, Hapugala, Galle) is an instance for address attribute

ATTRIBUTES

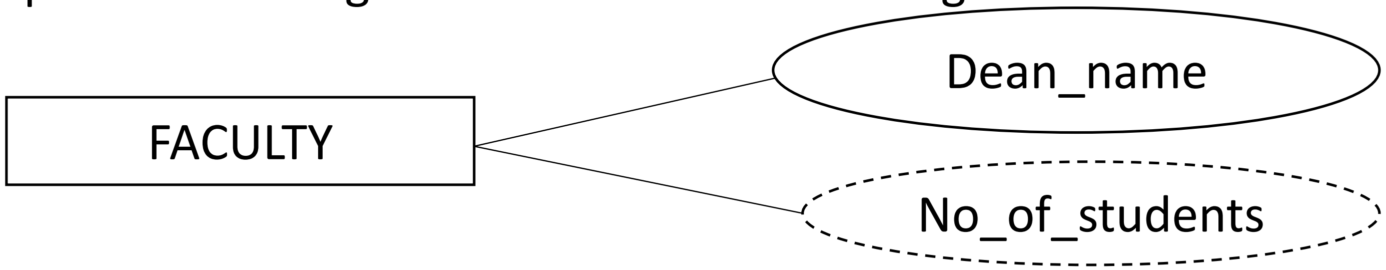
- An attribute having multiple values is an **multivalued attribute** represented by a double line oval in ER diagrams**.** E.g. attribute *Attended_colleges* of person entity type. Components of a multivalued attribute are written within { }.
- E.g. {Dhammissara, Maliyadewa, Joseph Vaaz} are an instance for attended_colleges



PERSON

Attended_colleges

- An attribute which can be derived from an entity or from a stored attribute is known as a **derived_attribute.** Eg: attribute *no_of_students* of Faculty entity type is derived attribute. A derived attribute can be represented using a dashed oval in an ER diagram.



FACULTY

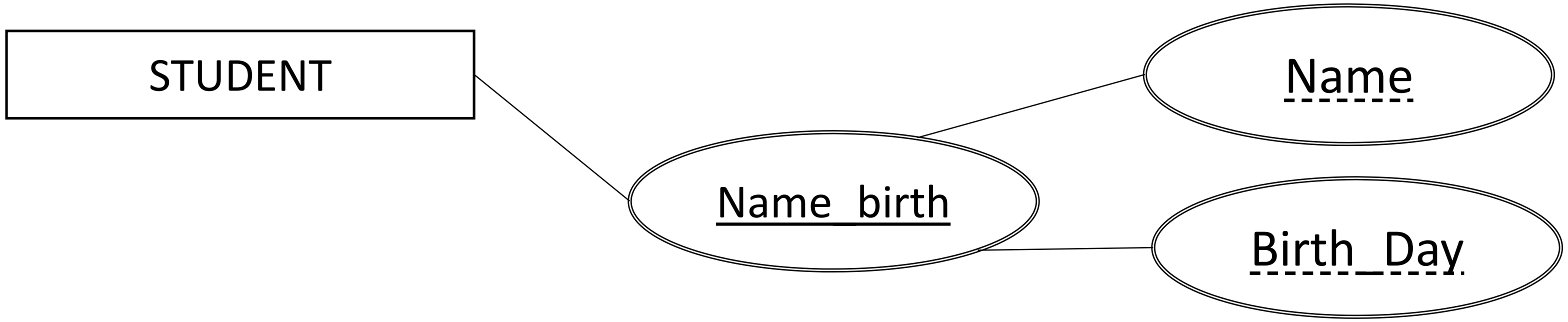
Dean_name

No_of_students

- *A* **null value** can be assigned to an attribute if the attribute is not applicable (E.g. apartment number), or value exists and missing (E.g. Height) or unknown about existence(E.g. Degrees).

ATTRIBUTES

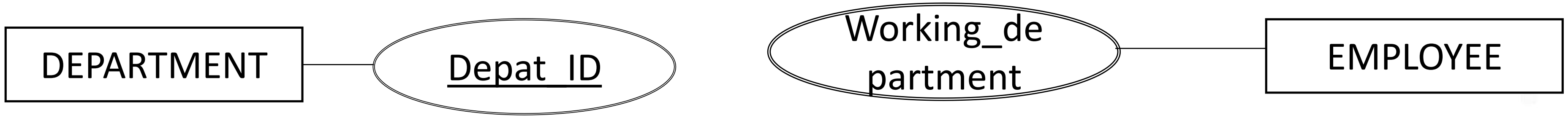
- **Primary key** - unique identifier for each record, such as a driver license number, telephone number (including area code), NIC number
- When one or a *minimal* combination of attributes form ***distinct entities*** for a particular entity type, such attribute or combination of attributes is known as the **key attribute**. Key attributes are underlined in ER diagrams.



- When a combination of attributes form a key, such attributes can be represented in ER diagram as a composite attribute. Each attribute forming a **composite key attribute** becomes a partial key shown as dashed underline in ER diagrams.
- An entity can have *multiple keys* or a *single key* or *no key* attribute.

RELATIONSHIPS

- If one attribute of an entity associates another attribute of another entity, there exists a relationship in ER data model. Consider the two entities department and employee.



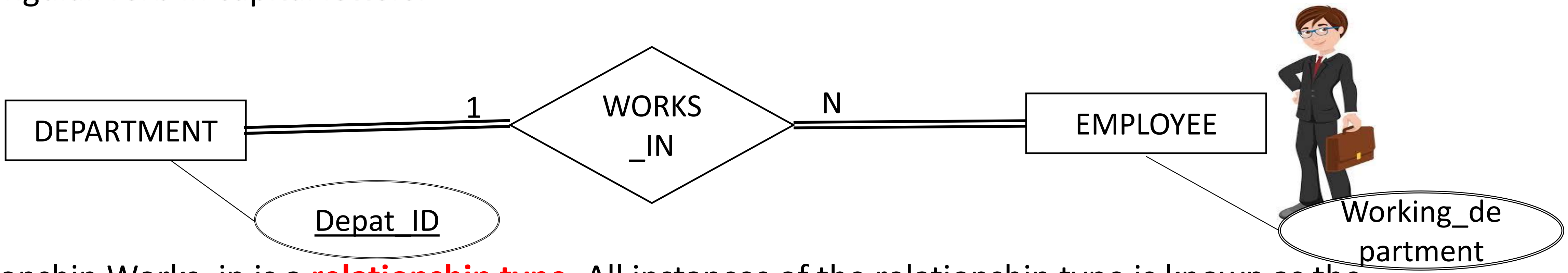
DEPARTMENT

Depat_ID

Working_de partment

EMPLOYEE

- In ER diagrams the relationship is drawn between the entities (Even though the relationship really exists between attributes). A relationship is notated using a diamond shape connected by lines. The name is written as a singular verb in capital letters.



DEPARTMENT

1

WORKS _IN

N

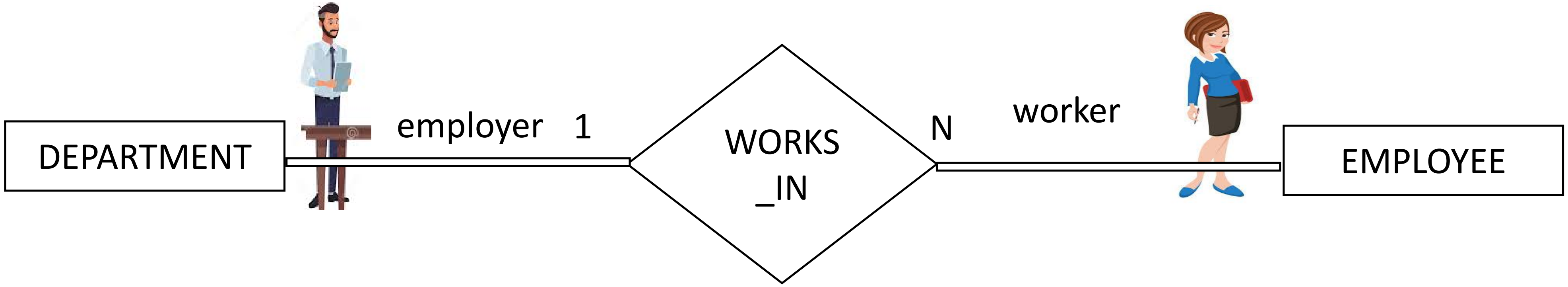
EMPLOYEE

Depat_ID

Working_de partment

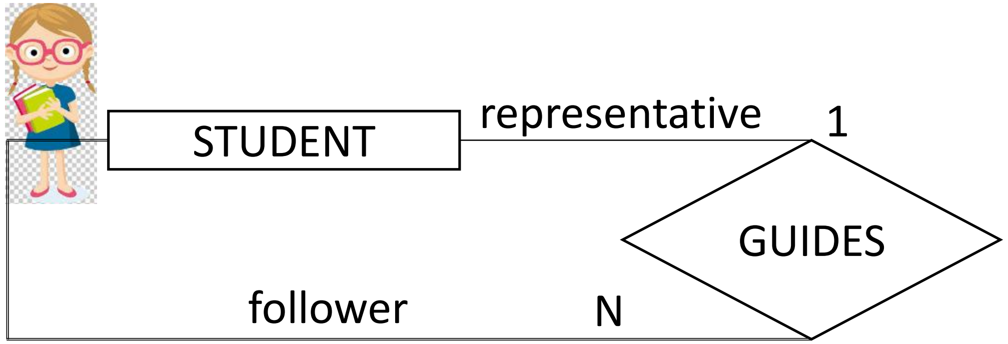
- Relationship Works_in is a **relationship type.** All instances of the relationship type is known as the **relationship set.** E.g. Erans WORKS_IN Electrical department, Raveen works in Mechanical department is a relationship set.
- The **degree** of a relationship is the number of entities participating in the relationship. WORKS_IN is a binary relationship as only 2 entities participate.

RELATIONSHIPS



- **Role name** is the act that each participating entity in the relationship plays. E.g. DEPARTMENT entity plays the *employer* role and EMPLOYEE entity plays the *worker* role. Role is written as a singular noun with all simple letters.

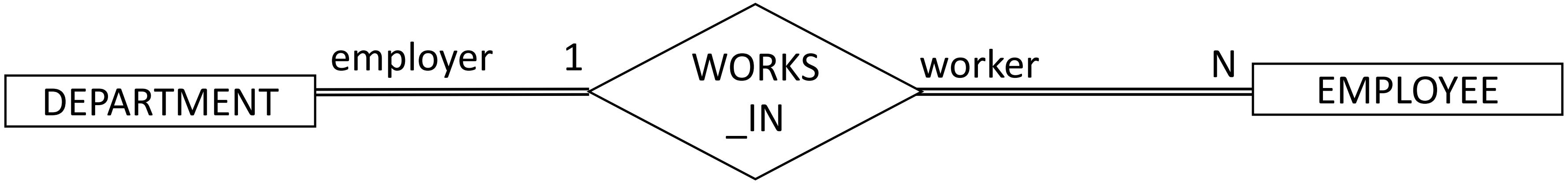
- **Cardinality ratio** is the maximum number of relationship instances an entity can participate in a relationship. Ex: for WORKS_IN relationship the cardinality ratio of department: employee is 1:N. That is an EMPLOYEE can work in ***at most*** 1 department. A department can get work from ***at most*** N employees.

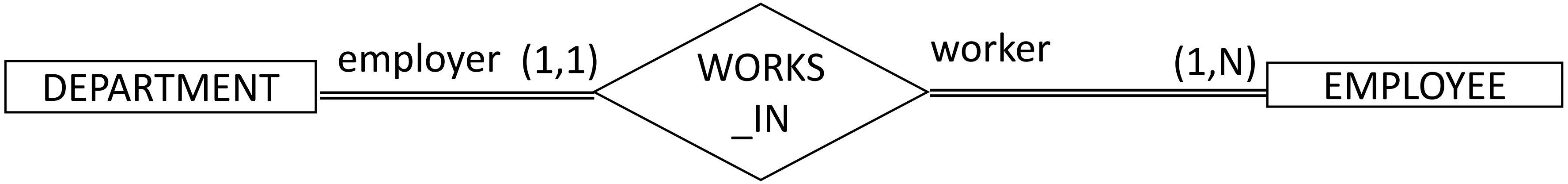


- When the same entity type participates in a relationship, then relationship is **recursive**.

RELATIONSHIPS

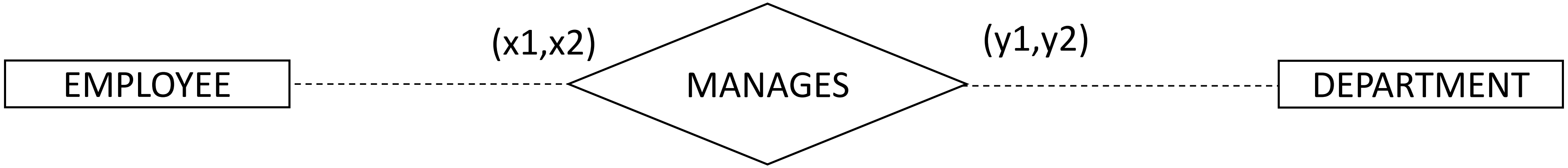
- **Participation constraint:** Specifies the minimum number of relationship instances that an entity can participate in. If the minimum is one, then it is known as **total participation/existence dependency.** If minimum is zero, the participation is **partial.**
- Total participation is denoted by using double lines and partial participation is represented using a single line.

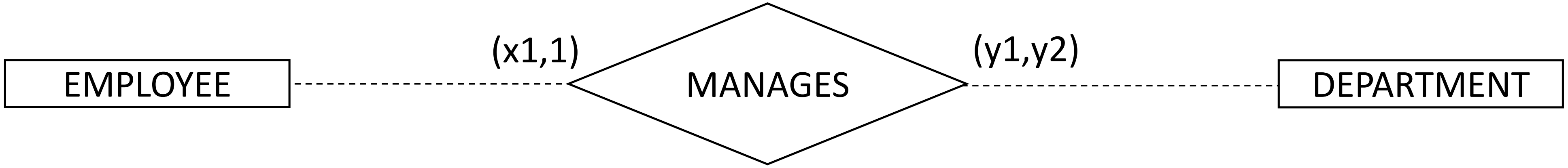


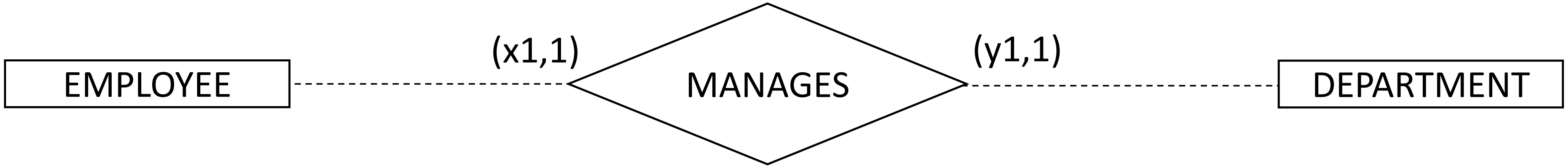


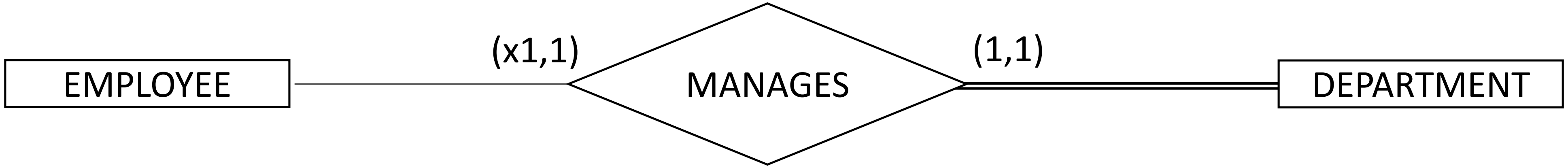
- Alternatively, structural constraint (participation constraint, maximum cardinality ratio) can be represented using (min, max) notation with a single line.
- In the WORKS_IN relationship type, all employees must have ***at least*** one worker role. All departments must have ***at least*** one employer role. So, worker role and employer role both have total participation.

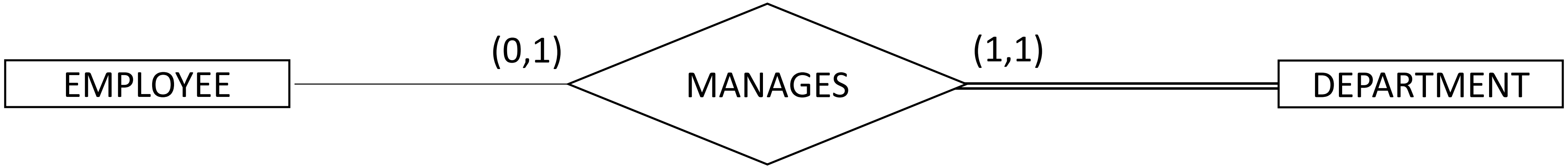
EXAMPLE


EMPLOYEE
(x1,x2)
MANAGES
(y1,y2)
DEPARTMENT


EMPLOYEE
(x1,1)
MANAGES
(y1,y2)
DEPARTMENT

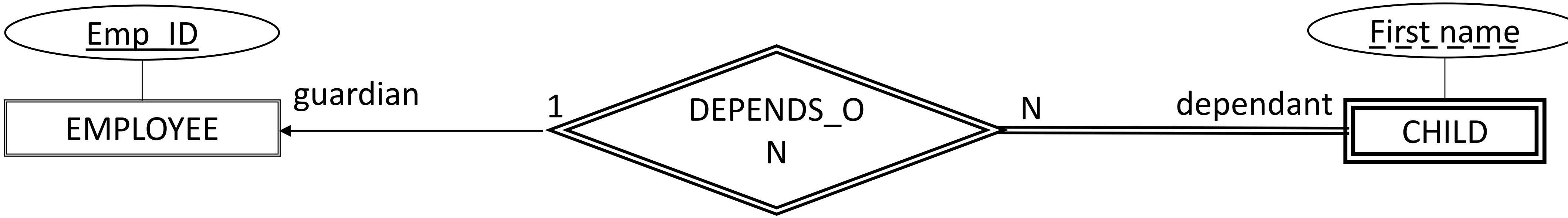

EMPLOYEE
(x1,1)
MANAGES
(y1,1)
DEPARTMENT


EMPLOYEE
(x1,1)
MANAGES
(1,1)
DEPARTMENT


EMPLOYEE
(0,1)
MANAGES
(1,1)
DEPARTMENT

WEAK ENTITIES

- An entity with no key attribute is known as a **weak entity type** represented by a rectangle with double lines. Entity with a single or multiple keys is a **strong entity.**
- A weak entity is always in a relationship with an **identifying or owner entity type.** Since a weak entity type doesn't have a key, it is identified using a combination of an attribute of identifying entity and an attribute of weak entity.
- The relationship between identifying entity and weak entity is **identifying relationship** notated by double line diamond shape which is a weak relationship type. Identifying entity is marked with an arrow (optional).
- Weak entity always has total participation in the relationship. But all total participations do not necessarily be weak entities.
- Attribute in weak entity which uniquely identifies weak entities belonging to same owner entity type becomes a **partial key**.
- The key attribute of identifying entity and partial key will combine to form the key of weak entity type.


Emp_ID
EMPLOYEE
guardian
1
DEPENDS_ON
N
dependant
First name
CHILD