Entity-Relationship

André Restivo

Loading [MathJax]/extensions/AssistiveMML.js

Index

Introduction
Attributes
Entities
Relationships
Constraints
Example
Visual Notation

Introduction

Entity-Relationship Model

- Abstract and conceptual model for data representation.
- Database perspective as seen by the **developer**.
- Based on three key concepts: **entities**, **attributes** and **relationships**.

Entities

Object or concept from the real world with an independent existence:

- **Physical** entity: car, employee, student, ...
- **Conceptual** entity: class, profession, type, ...

Relationships and Attributes

Relationships: represent interactions between 2 or more entities.

? car belongs to person.

Attributes: properties that describe an entity.

? person has a name and an address.

Attributes

Attribute Domain

- In an entity-relationship schema, attributes represent the **values** that will be stored in the database.
- An attribute should **never** be a reference to another entity.
- Attribute **domain**: The set of possible values for that attribute.

? The email address attribute can only contain valid email addresses.

Attributes: Simple and Composite

- **Simple**: Only one atomic component.
- **Composite**: Composed of several attributes.

? Address is composed by a street name, a door number, an appartment number and a zip code.

Attributes: Single or Multi-Valued

- Single Valued: Only one value per entity.
- Multi-Valued: Possibly more than one value per entity.

? A person can have several phone numbers.

Attributes: Derived

- Attributes that can be calculated from other attributes or entities.
- They don’t need to be a part of the database.
- Nevertheless, they are commonly represented in the entity relationship model for completness sake.

? A person’s age can be calculated from that person’s birth date.

Attributes: Null

- Some attributes might allow null values.
- A null value can represent an **unknown** value or that the value is not **appliable**.

Attributes: Keys

- Attribute, or set of attributes, that **uniquely** identifies an entity.
- No two entities of the same type, can have the same set of key values.

? Two different cars cannot have the same plate number. If we know the plate number we can identify the car it belongs to.

Entities

Entity-Type

- The schema of a set of entities that share the same structure.
- Carachterized by the entity’s name and it’s attributes.

? Employee (name, address, city)

Notation

- Key Attribute: **underlined**.
- Composite Attribute: attribute name and **list** of composing attributes.
- Multi-valued Attribute: between **curly brackets**.

? Employee (<u>number</u>, name, address(street, number, apartment), {phone})

Relationships

Relationships

- A relationship is an association between one or more entity-types.
- Should be used instead of having attributes that reference other entities.

? A car belongs to an employee.

- Never use the key attribute of another entity to mimic a relationship.

? Car (plate, make, model, employee_num)

? Students are enrolled in a class.

Notation

- Relationships are described by a name, the list of participant entities and its attributes.
- The name is normally an expression that makes sense when read in the following order:

? Entity1 $\rightarrow$ relationship $\rightarrow$ Entity2

Examples:

? belongsTo(Car, Person)

? enrolled(Students, Class)

Degree

Unary

? supervises(Employee, Employee)

Binary (the most common)

? belongsTo(Car, Person)

Ternary (can be transformed into binaries)

? supplied(Supplier, Product, Project)

Constraints

Constraints

Allows us to limit the possible combinations between entities participating in a relationship.

Constraint types:

- Relationship **cardinality**.
- Entity **participation** in a relationship.

Cardinality

N:1 ou 1:N (many-to-one) ou (one-to-many)

? belongsTo(Car, Person) N:1

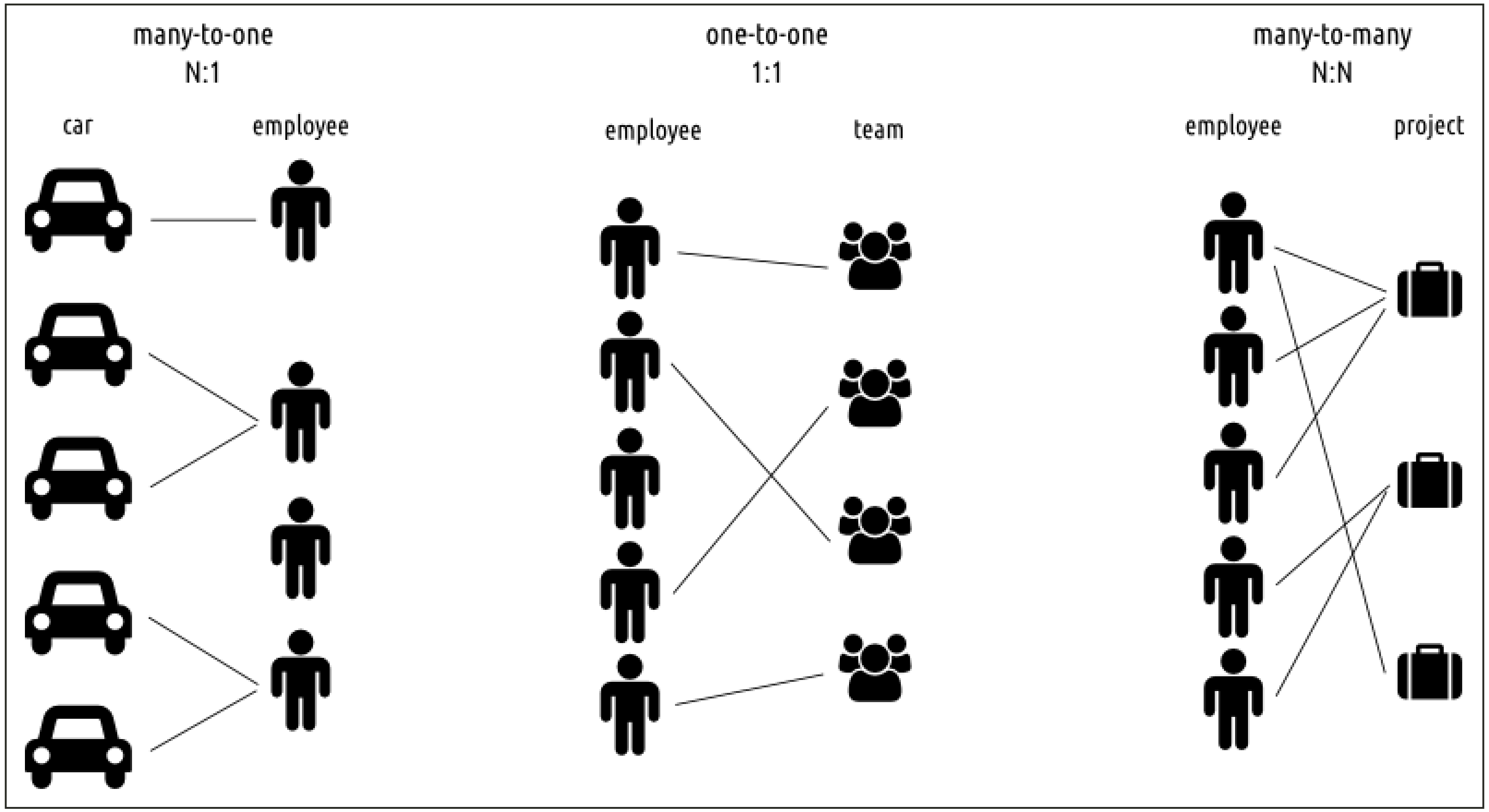
1:1 (one-to-one)

? leaderOf(Employee, Team) 1:1

N:N (many-to-many)

? worksOn(Employee, Project) N:N

Cardinality



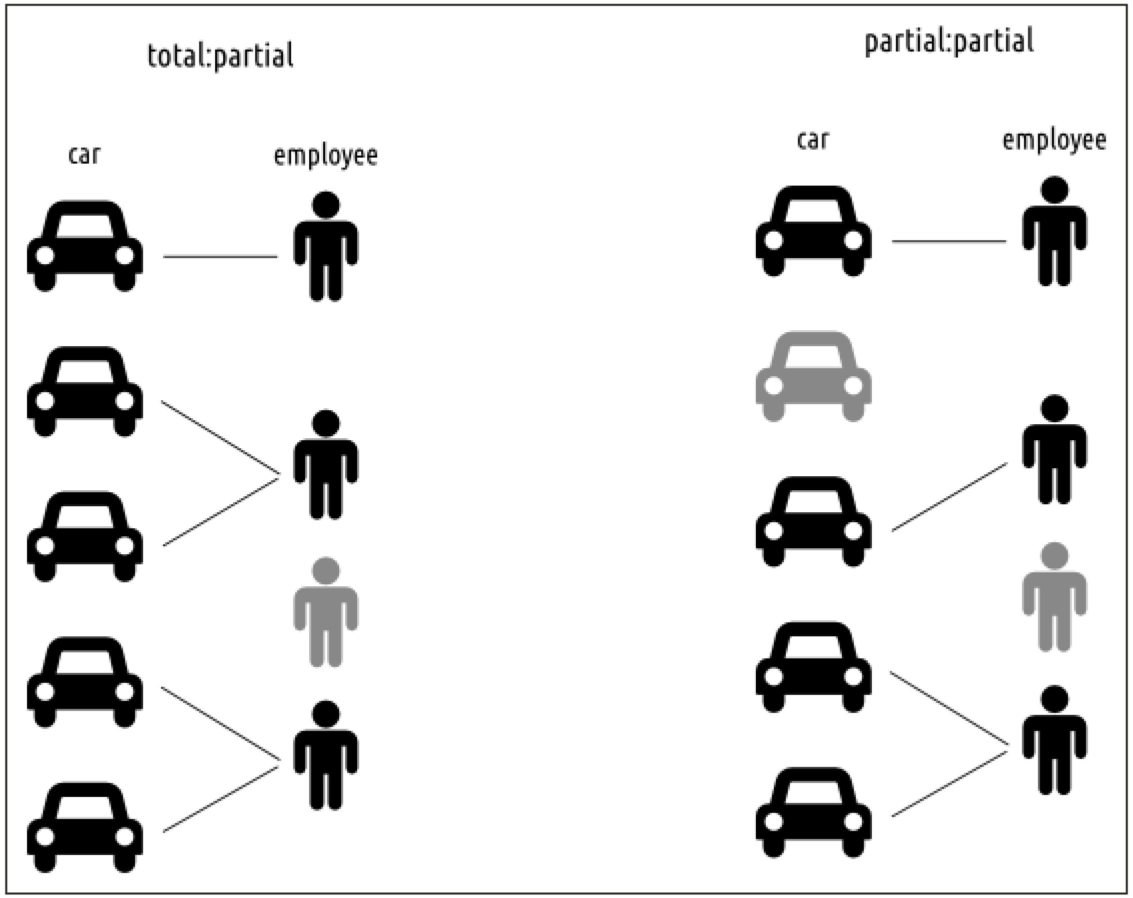
Participation

- Total (existencial dependency): All entities of this type must participate in the relationship.
- Partial: Entities may not participate in the relationship.

? belongsTo(Car, Person) N:1 t/p

? All cars in the database must have an owner. Not all persons have to own cars.

Participation



Total Participation

- We should **only** label a participation as total when it is necessary to ensure that all entities participate in a relationship.
- A total participation is a **constraint**. It restricts the possible values for an attribute.
- It is most common in **N:1** relationships in the **many** side.

Example

Example

- A company has several departments.
- Each department has a name, a number and a manager. The date when the manager was assigned to the department should also be stored.
- A department controls a number of projects. Each project has a name and a number.
- Each employee has a name, a number, an address and a salary.
- An employee belongs to a department but can work in many projects (some of them from different departments).
- The number of hours each employee is supposed to work in each project should also be saved.
- Each employee can have a supervisor assigned to him.

Solution

Entities

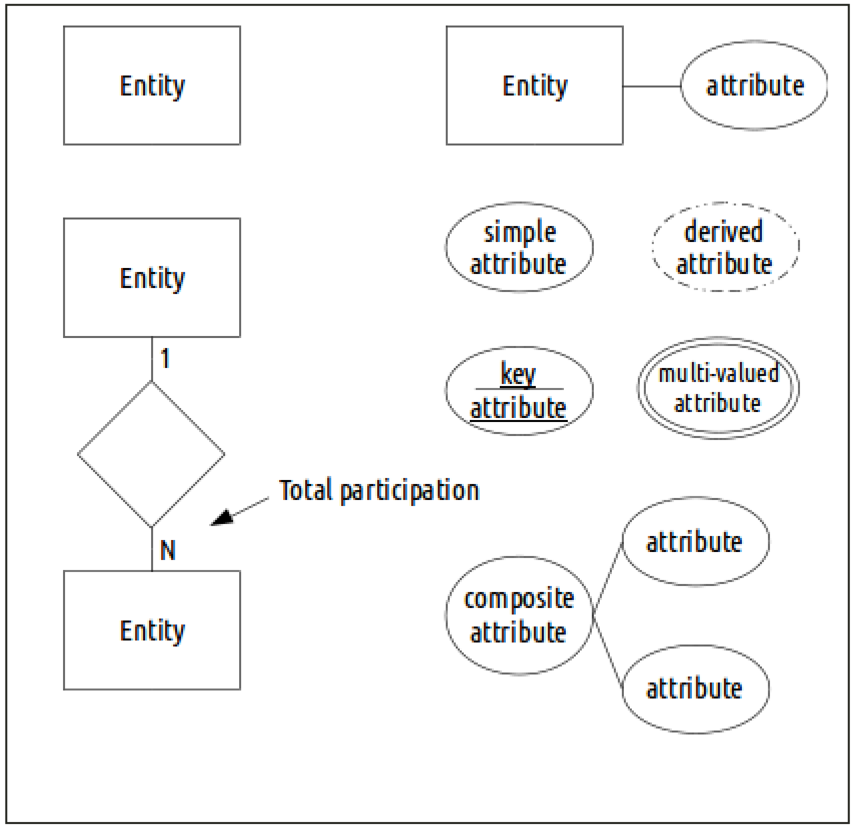
- Employee (number, name, address, salary)
- Project (number, name)
- Department (number, name)

Relationships

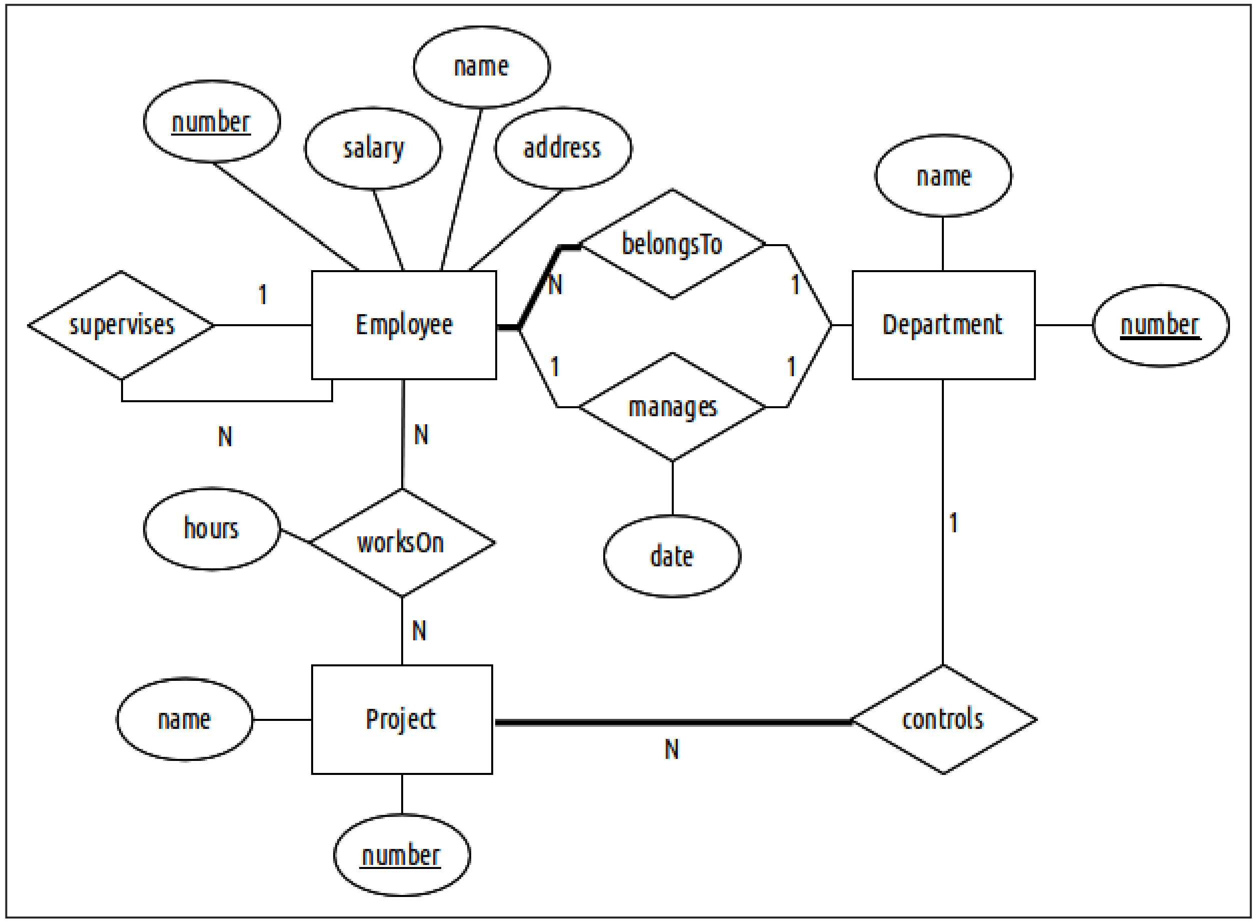
- manages (Employee, Department) 1:1 p/p
- controls (Department, Project) 1:N p/t
- belongsTo (Employee, Department) N:1 t/p
- worksOn (Employee, Project) N:N p/p
- supervises (Employee, Employee) 1:N p/p

Visual Notation

Visual Notation



Example



number
salary
name
address
name
supervises
1
Employee
belongsTo
N
1
Department
number
1
1
manages
N
N
hours
worksOn
date
1
N
name
Project
controls
N
number