



Entity-Relationship (ER) Model

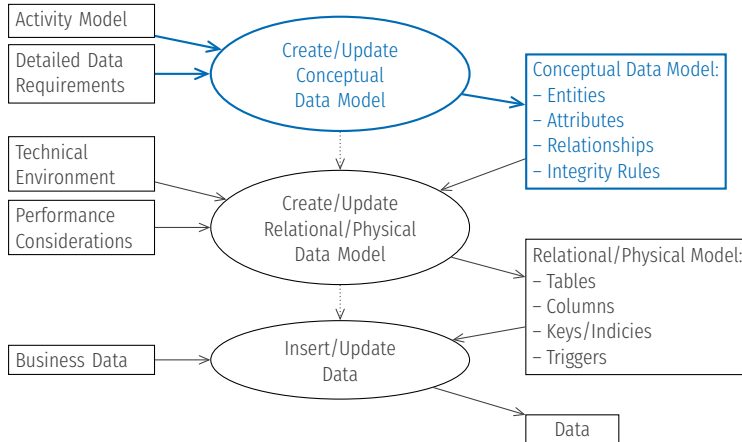
GRA 4142 Data Management and Python Programming

Jan Kudlicka (jan.kudlicka@bi.no)

Department of Data Science and Analytics

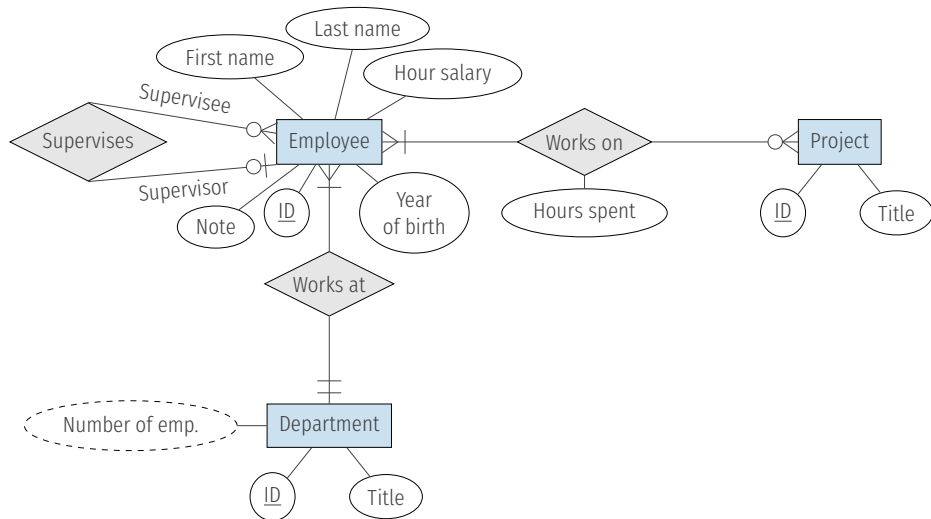
November 11, 2021

Data modeling: creating a data model for an information system



Source: EPISTLE, modified

- ▶ Comparing to the natural language:
 - ▶ More formal
 - ▶ Not ambiguous
- ▶ Comparing to the relational model:
 - ▶ Neither implementation nor (R)DBMS specific details
- ▶ High-level description of the mini world (a part of the real world which we wish to model).
- ▶ Easy to understand by people without technical background.
- ▶ Can be used as the documentation.
- ▶ Can be easily transformed to a relational model (and then implemented in a relational database).



Three main components:

- ▶ Entities
- ▶ Attributes
- ▶ Relationships

(Strong) entity: a “thing” or “object” in the mini world (for which we want to store some data) that is capable of an independent existence.

Examples of entities in an university IT system: each student, teacher, course, department, building, classroom, etc.

Attribute: a (relevant) property or feature of entities.

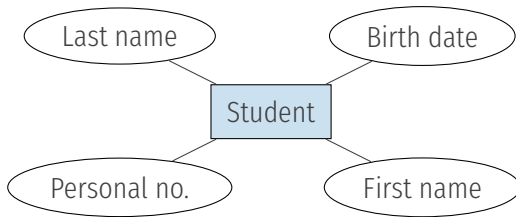
Examples of attributes for a teacher: the personal no., the first name, the last name, the birthdate, etc.

Entity type: a set of entities having the same attributes.

Entity type Student is a set of all students.

- ▶ Entities do not have to exist physically.
E.g. a department, a course or an order in an online store.
- ▶ It must be possible to distinguish entities from each other, ie. each entity must have some identity.
E.g. the personal no. for teachers or students, a combination of the building name and the room no. for classrooms.
- ▶ If you are familiar with object-oriented programming:
Entity type corresponds to a class and entities to instances of this class.

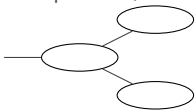
In the ER diagram we use rectangles for entity types and ovals for attributes. The attributes are connected to the entity type by lines.



- ▶ Simple: each entity has a single atomic (i.e. indivisible) value. — 

E.g. "ID", "Age", "First name".

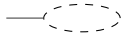
- ▶ Composite (or compound): the attribute can be divided into several components.



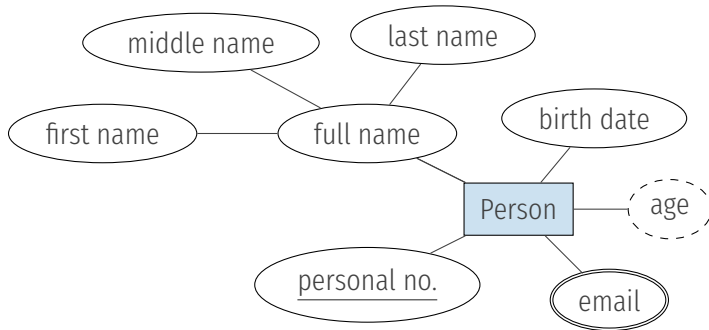
E.g. "Name" consists of the first, middle and last name, "Address" consists of the street name, the house no., the city and the postal no.

- ▶ Multi-valued: the attribute might have multiple values. — 

E.g. "Email address" for a person, "Keyword" for a blog post.

- ▶ Derived – the value of the attribute can be determined from the other attributes. — 

E.g. "Age" if there is an attribute for the birth date.



Relationship: a relation between entities of given entity types.

Note: Distinguish between a relation in a relational database (i.e. a table) and a relation(ship) between entities.

Jan Kudlicka (entity of the type Teacher) teaches (relationship) the GRA 4142 course (entity of the type Course).

Ola Nordmann (entity of the type Student) takes (relationship) the GRA 4142 course (entity of the type Course).

Relationship type: a set of all relations between entities of the same entity types.

Note: We will only work with binary relationships, i.e. relationships involving two entity types, as these are the most common. The relationships can however involve more than 2 entity types.

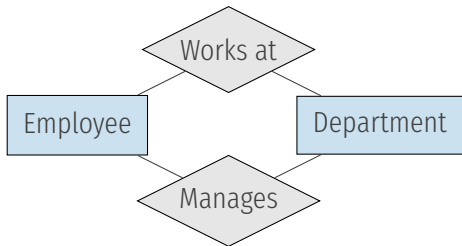
A diamond with lines to the involved entity types.



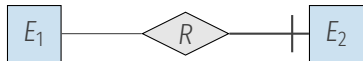
Relationships might also exist between entities of the same type.

Homer Simpson (entity of the type Person) is a father (relationship) of Lisa Simpson (entity of the same type).

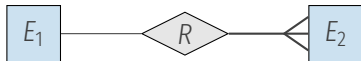
There might be several relationships between the same entity types.



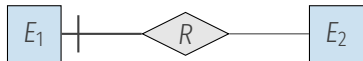
An E_1 entity can be related to **one** E_2 entity:



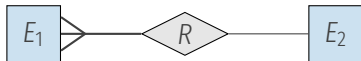
An E_1 entity can be related to **several** E_2 entities:



An E_2 entity can be related to **one** E_1 entity:



An E_2 entity can be related to **several** E_1 entities:



Three different types of relationships based on cardinalities:

- ▶ One-to-one (1:1)
- ▶ One-to-many (1:N)
- ▶ Many-to-many (N:N)

One-to-one (1:1)



No employee can be the head of several departments. No department can have several heads.

One-to-many (1:N)



A publisher can publish many books. No book can have several publishers.

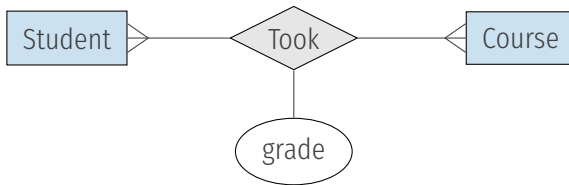
Many-to-many (N:N)



An author can have written several books and a book can have several authors.

Relationships might have attributes too.

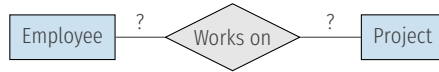
Example: Ole Nordmann (entity of the type Student) took (relationship) the GRA 4142 course (entity of the type Course) and got a grade A. The grade is an attribute of the relationship, not an attribute of a student, nor of a course.



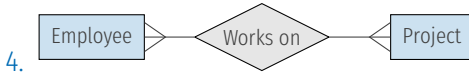
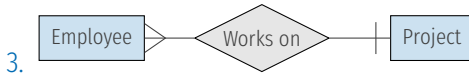
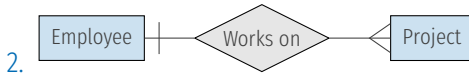
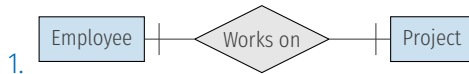
Example: Oslo Nye Teater (entity of the type Theater) plays (relationship) Hamlet (entity of the type Play) this Saturday at 7pm. The date and time is an attribute of the relationship.

Chen proposed the following “rules of thumb” for mapping natural language descriptions into ER diagrams:

English grammar structure	ER structure
Common noun	Entity type
Proper noun	Entity
Transitive verb	Relationship type
Intransitive verb	Attribute
Adjective	Attribute for entity
Adverb	Attribute for relationship



An employee can work on several projects, and a project can have several employees working on it.





No employee manages more than one department and each department can only have one manager.

1.

Diagram 1 shows a one-to-one relationship. The 'Employee' entity is connected to the 'Manages' relationship with a vertical bar and a crow's foot symbol (mandatory one). The 'Department' entity is connected to the 'Manages' relationship with a vertical bar and a crow's foot symbol (mandatory one).
2.

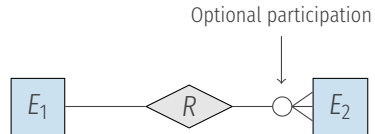
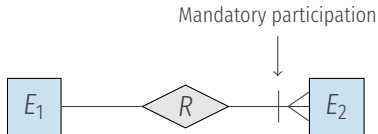
Diagram 2 shows a one-to-many relationship. The 'Employee' entity is connected to the 'Manages' relationship with a vertical bar and a crow's foot symbol (mandatory one). The 'Department' entity is connected to the 'Manages' relationship with a vertical bar and a crow's foot symbol with a branching line (mandatory many).
3.

Diagram 3 shows a many-to-one relationship. The 'Employee' entity is connected to the 'Manages' relationship with a crow's foot symbol with a branching line (mandatory many). The 'Department' entity is connected to the 'Manages' relationship with a vertical bar and a crow's foot symbol (mandatory one).
4.

Diagram 4 shows a many-to-many relationship. Both the 'Employee' and 'Department' entities are connected to the 'Manages' relationship with a crow's foot symbol with a branching line (mandatory many).

- ▶ **Mandatory participation:** each entity must be related to at least one entity of the other type.
A book has to have at least one author.
- ▶ **Optional participation:** an entity does not have to be related to any entity of the other type.
An employee does not have to be a manager of any project.

Notation:





Each employee must work at some department and each department must have some employees.

- 1.
- 2.
- 3.
- 4.

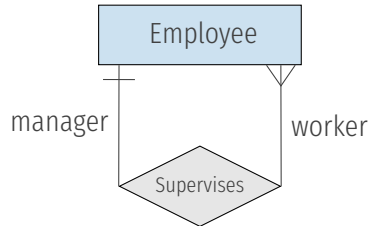


Not every employee is a manager, but every department has a manager.



A **role** describes what function an entity has in a relationship.

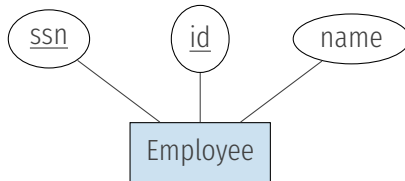
Usually not specified, except for relationships between entities of the same type:



Key attribute: an attribute that uniquely identifies entities of a given type (i.e., each entity has a unique value of this attribute).

Several attributes together might be used as a key (with the help of a composite attribute).

Each (strong) entity type has to have at least one key. If none of the attributes can be used as a (natural) key, you can add a “surrogate” key attribute (e.g. an integer ID).



- ▶ **Weak entities:** entities that cannot be uniquely identified by its own attributes and must rely on the identity of an “owner” entity.

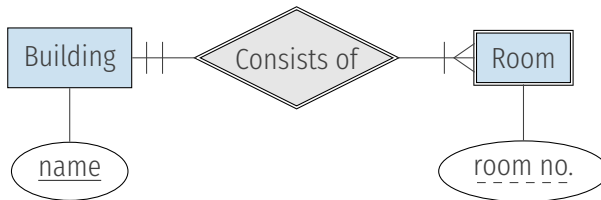
E.g. invoice items on an invoice, monthly payments of a loan, rooms in a building, departments in an organization.

- ▶ A relationship to the owner entity is called an **identifying relationship**.
 - ▶ An identifying relationship can only be 1:1 or 1:N.
 - ▶ In the 1:N case, the weak entity type needs a **partial key**—an attribute which together with the owner’s key can uniquely identify the weak entities.
- ▶ Weak entities can be owners of other weak entities.

Weak entity type is represented by a double-lined rectangle.

Identifying relationship is represented by a double-lined diamond.

Partial key is underlined using a dashed line.



Create an ER model for a simple online database for movies based on the following specification:

- ▶ Each movie has a title, release year, runtime in minutes, MPA rating (see below), one or more directors, one or more writers, one or more actors, and a storyline (a description of the plot of the movie).
- ▶ Each movie can be assigned one or more genres, such as *comedy*, *drama*, *action*, *horror*, etc.
- ▶ Each actor, writer or director has a name (consisting of the first, middle and last name), date and place of birth, and a short biography.
- ▶ For each movie we wish to store names of all characters and who played them (e.g. *John McClane* in *Die Hard*, played by *Bruce Willis*). Note that the same actor might play several characters in the same movie.