Lecture 5

Conceptual Design with the Entity Relationship Data Model (continued)

Week 3

Overview

- Relationships
- Participation and cardinality constraints
- Higher order relationships
- Roles

Relationship type

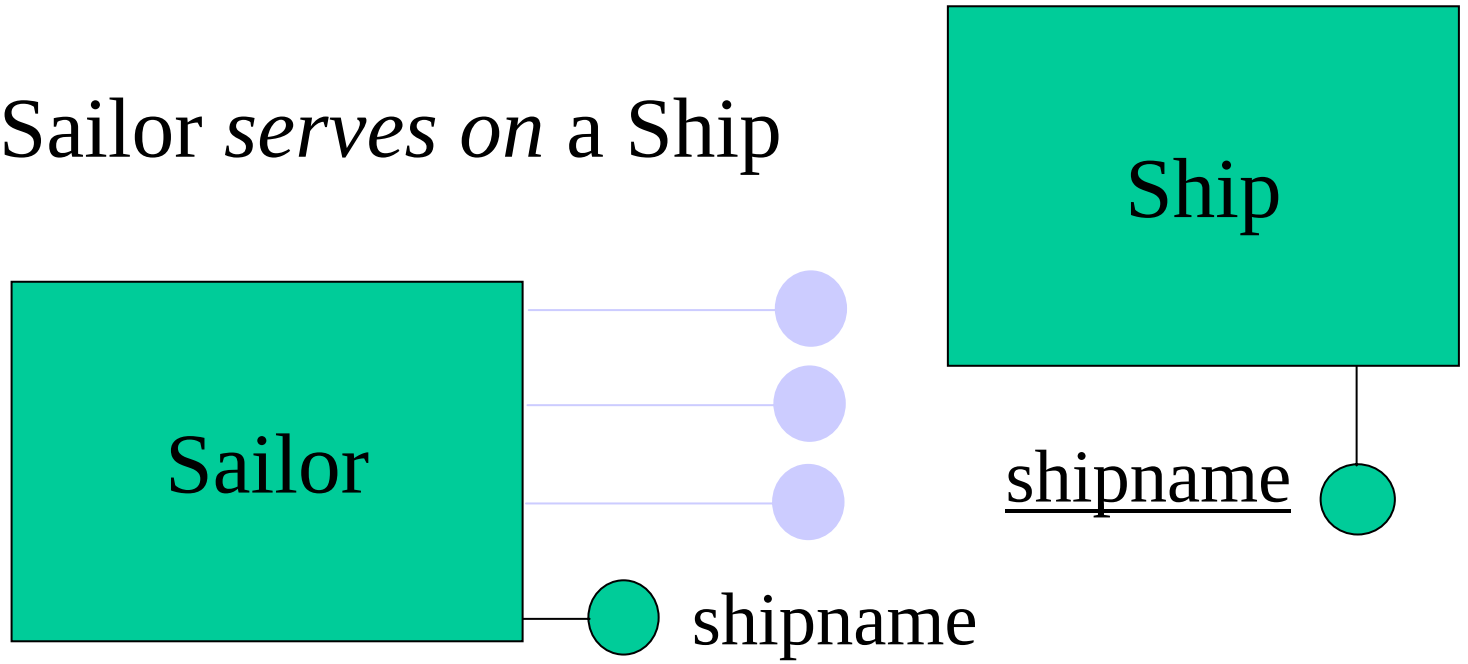
- A relationship type defines a property of entities of an entity type

  E.g. Sailor *serves on* a Ship

Relationship type

- A relationship type defines a property of entities of an entity type

  E.g. Sailor *serves on* a Ship

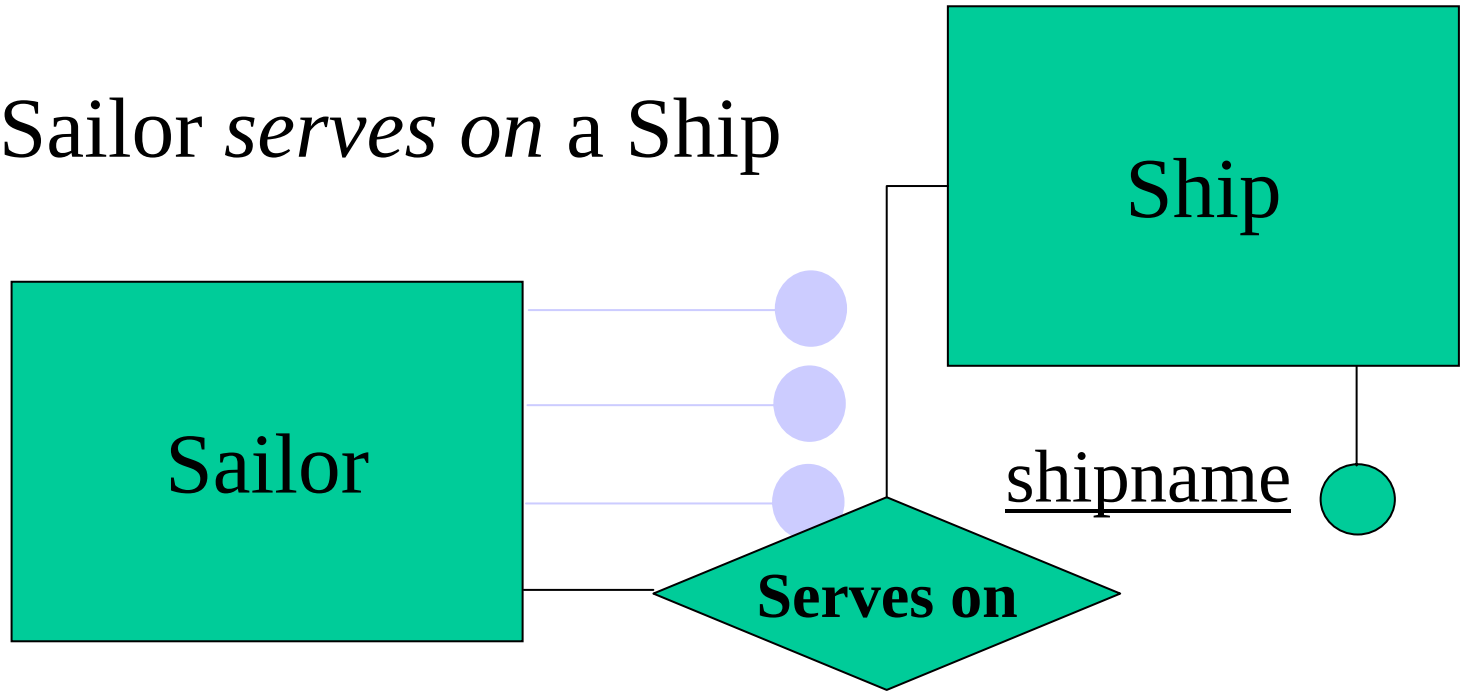


shipname is the name of a ‘Ship’. Hence, we do not represent this property of ‘Sailor’ as an attribute, but instead...

Relationship type

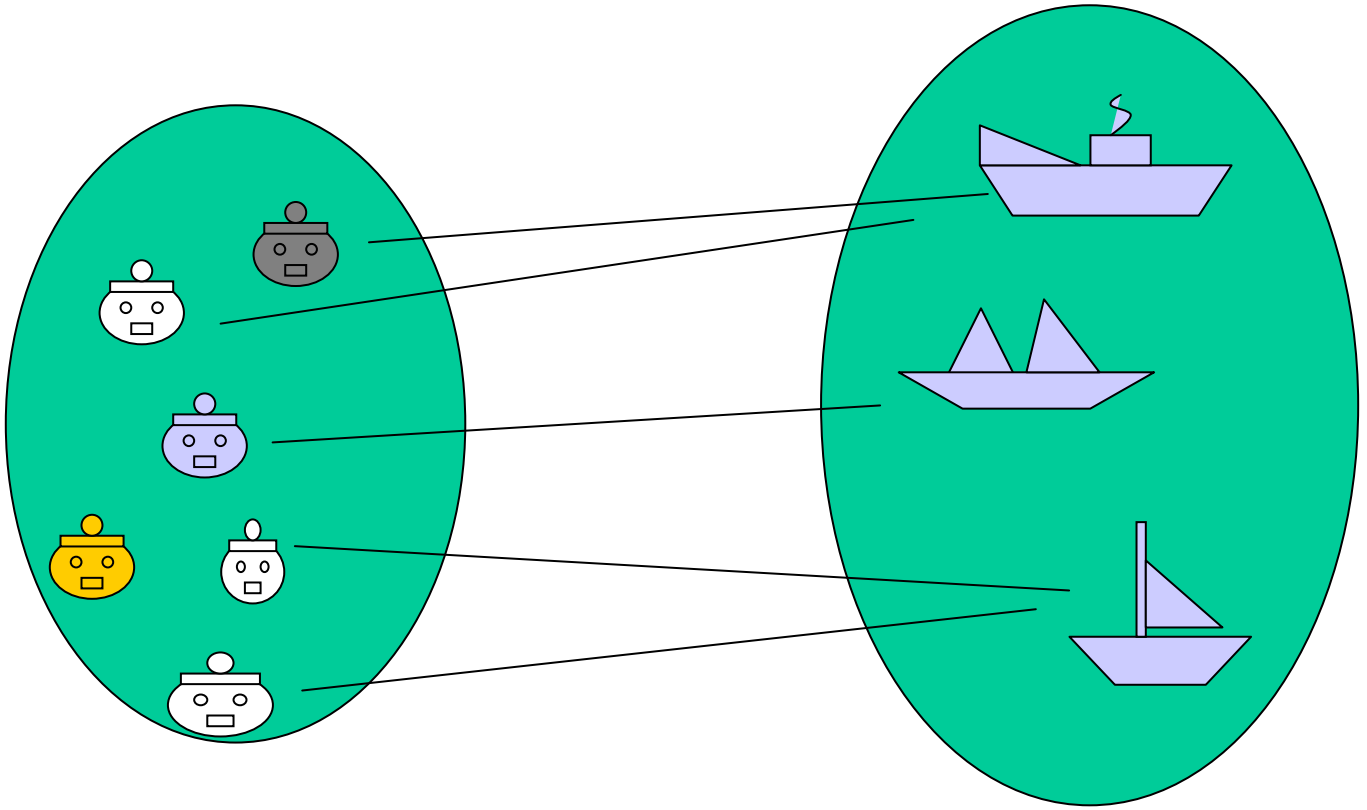
- A relationship type defines a property of entities of an entity type

  E.g. Sailor *serves on* a Ship

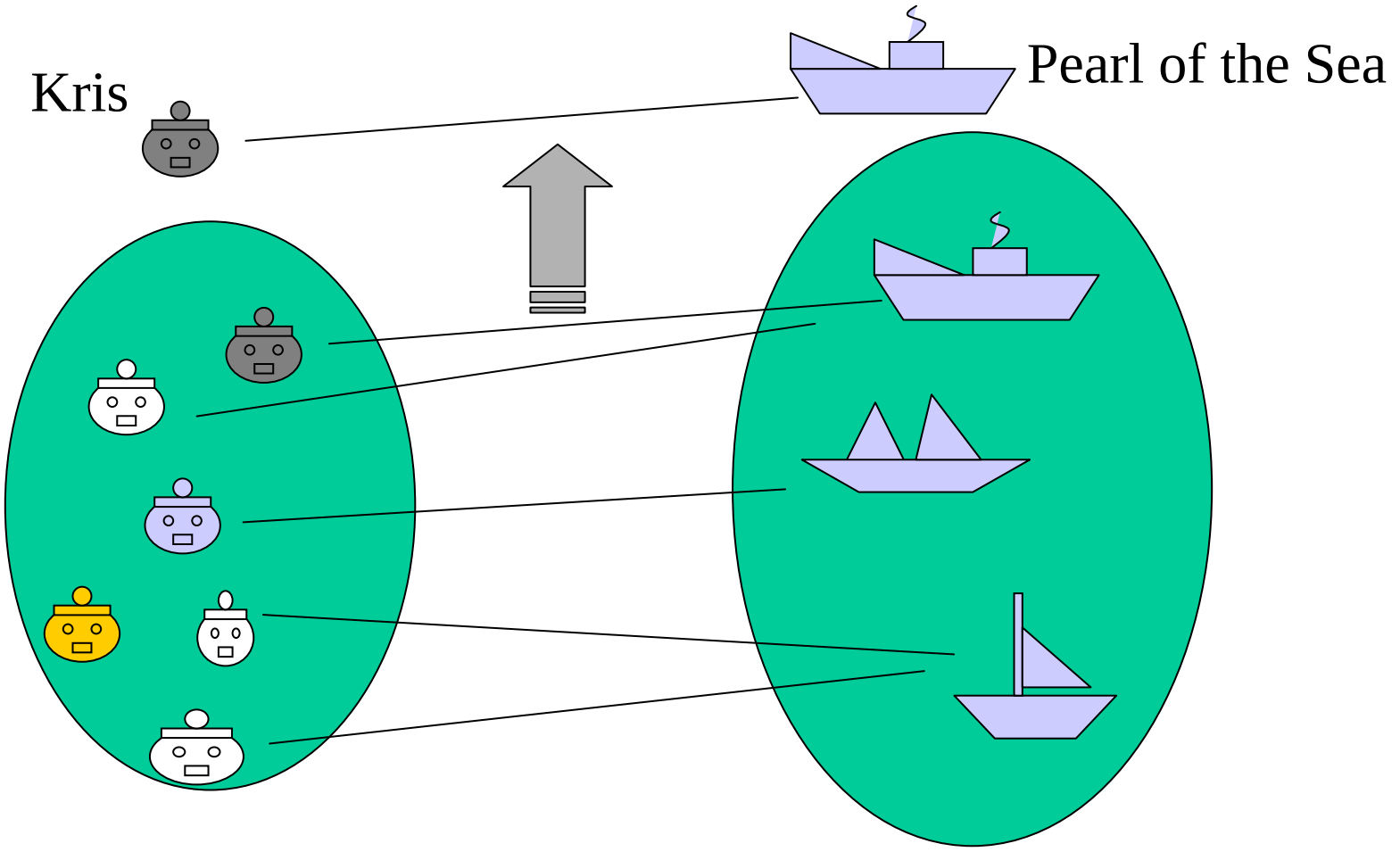


We represent this fact as a *relational property*

Relation, Relationship Instances



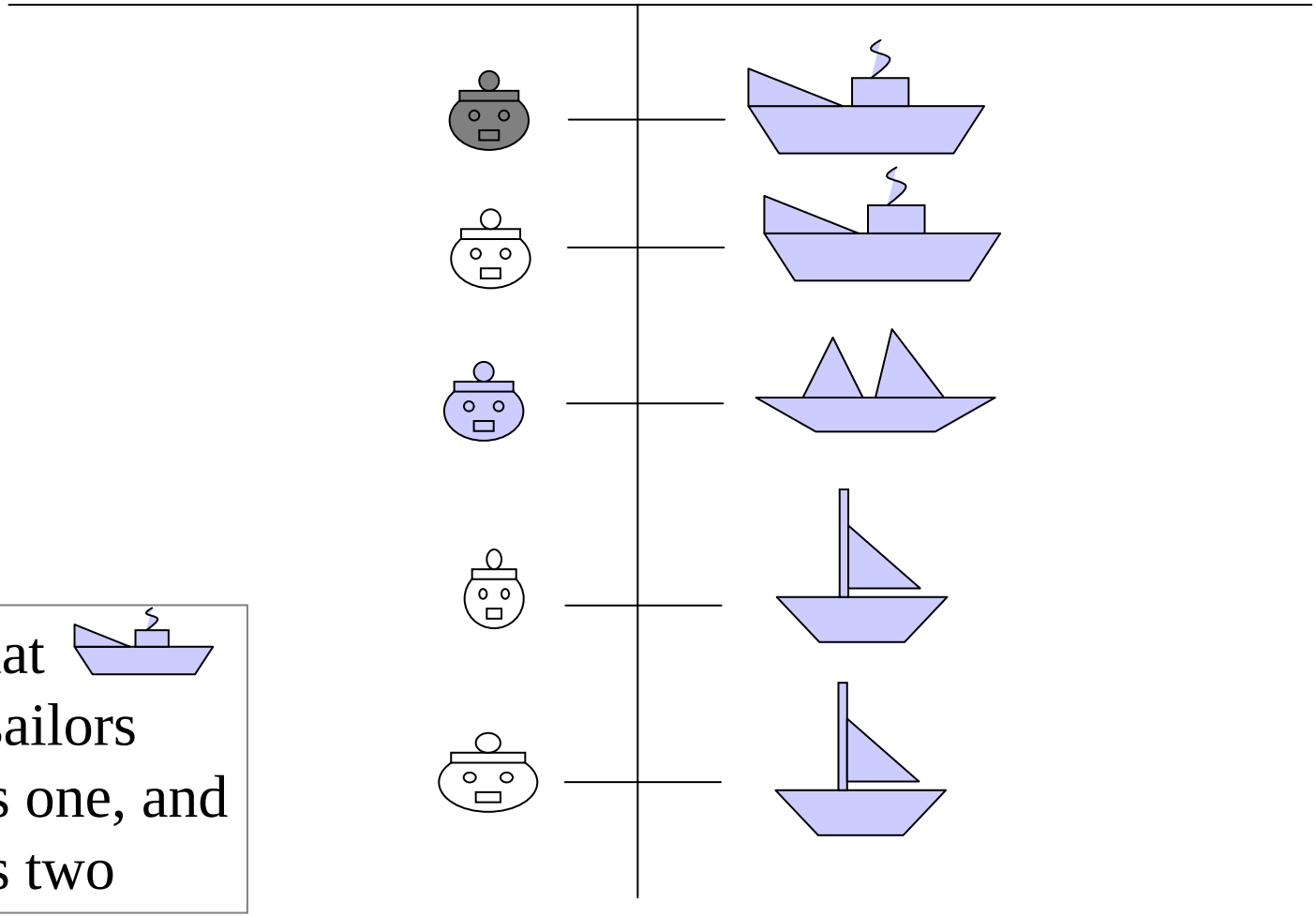
Relationship Instance



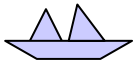
Example relationship instance or relationship) :

Kris ***serves on*** the Pearl of the Sea

Relation, Relationship Instances



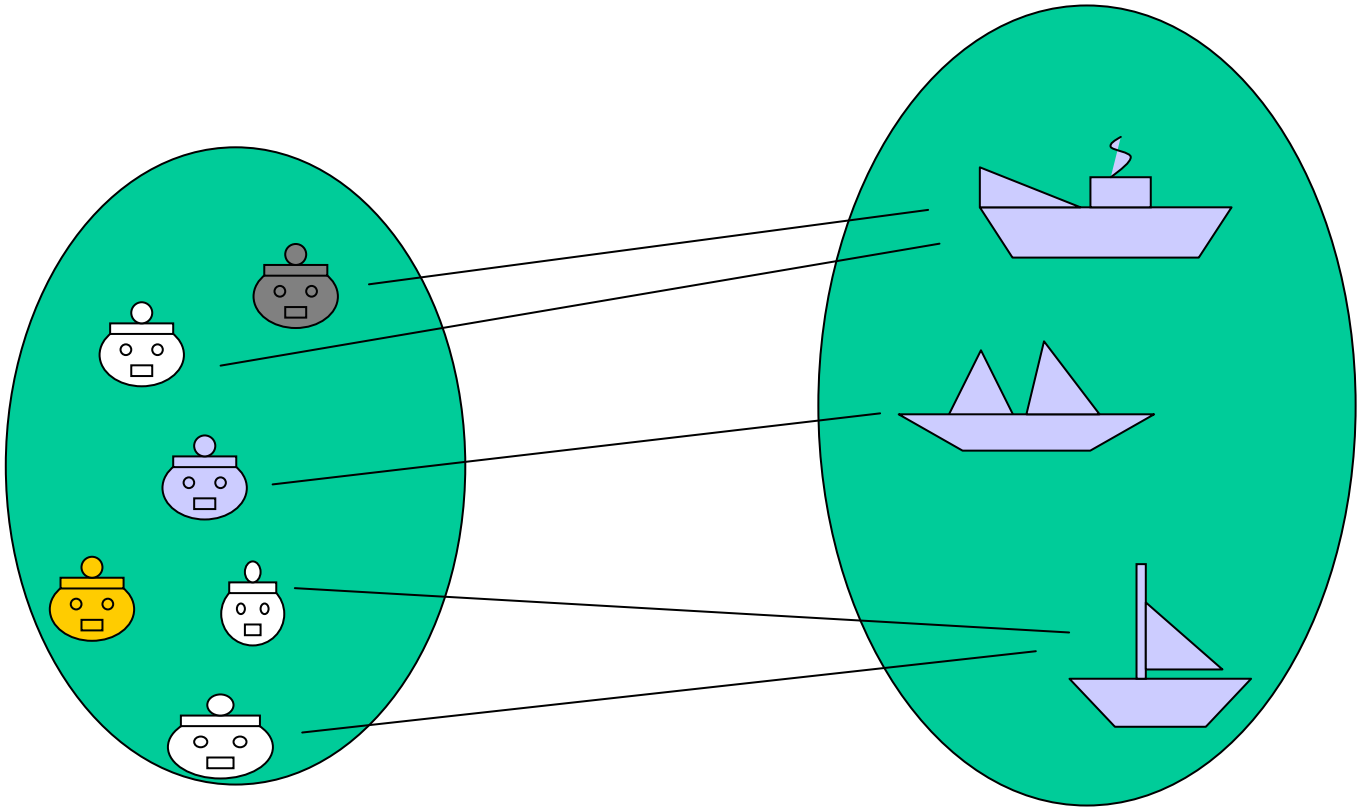
Notice that
has two sailors

 has one, and

 has two

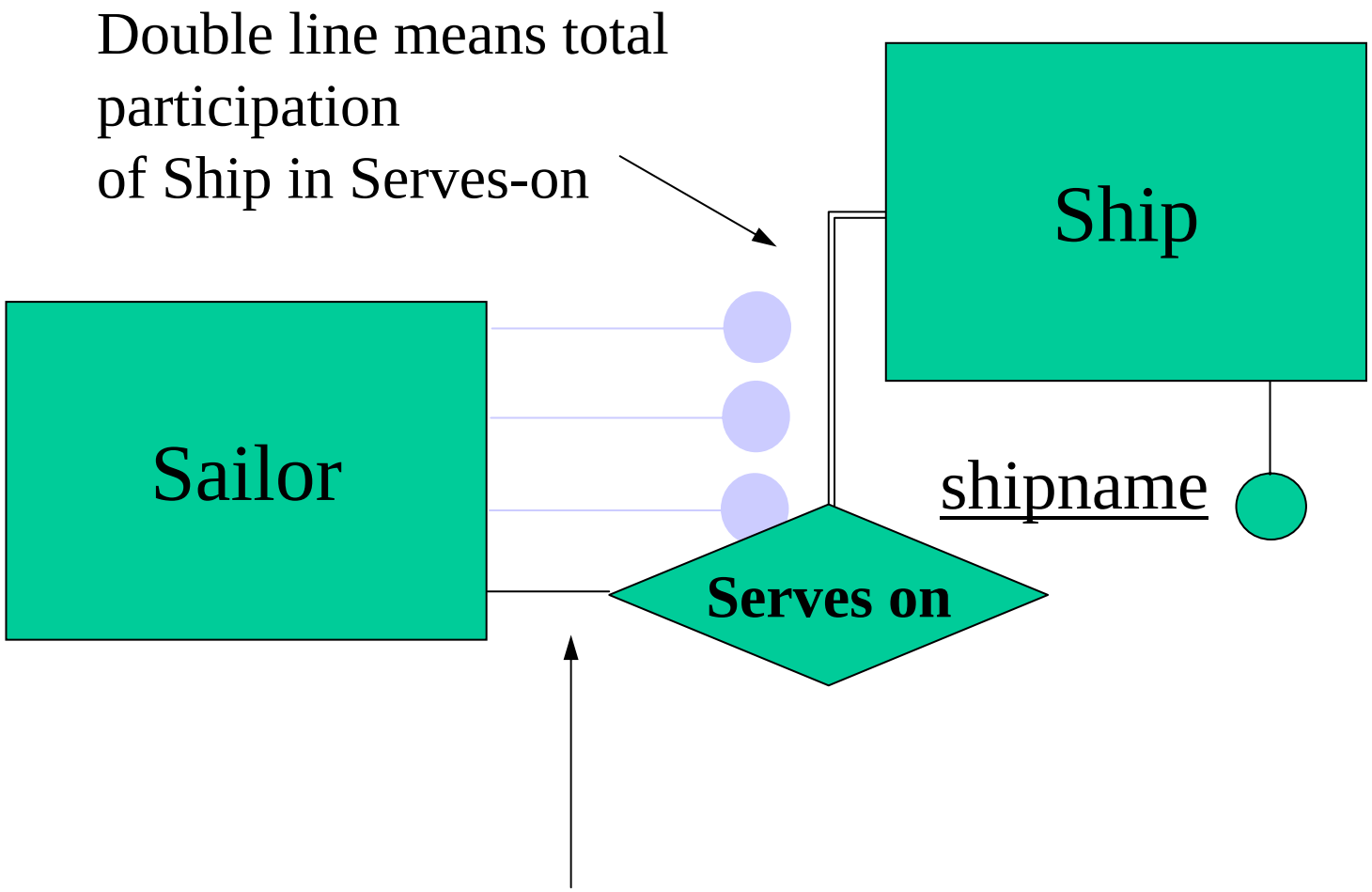
A relation is a set of relationship instances

Participation and cardinality constraints



Some sailors do not serve on any ship (***partial*** participation)

Every ship has at least one sailor (***total*** participation)

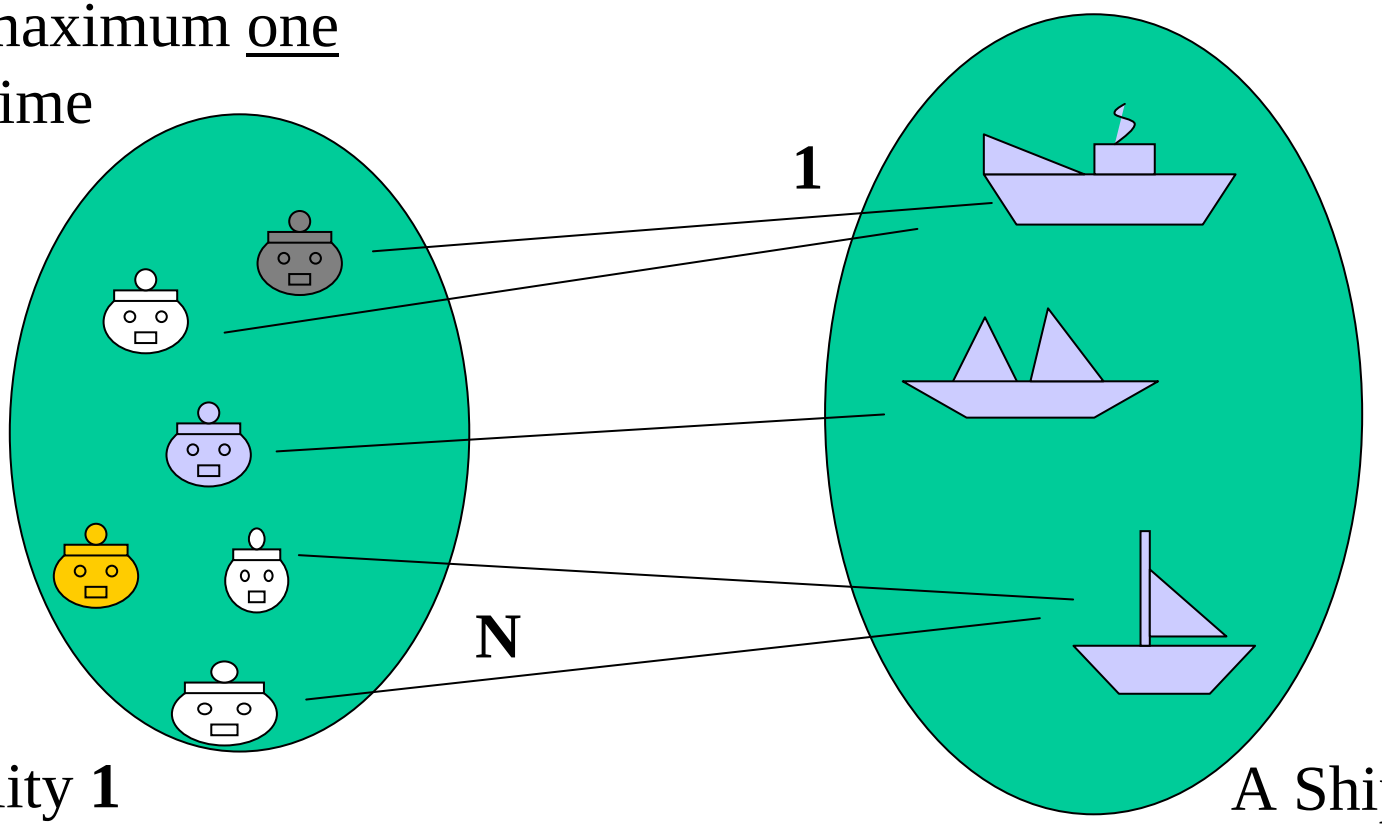


(Note: There are alternative representations)

Participation and cardinality constraints

A Sailor may only serve on maximum one Ship at a time

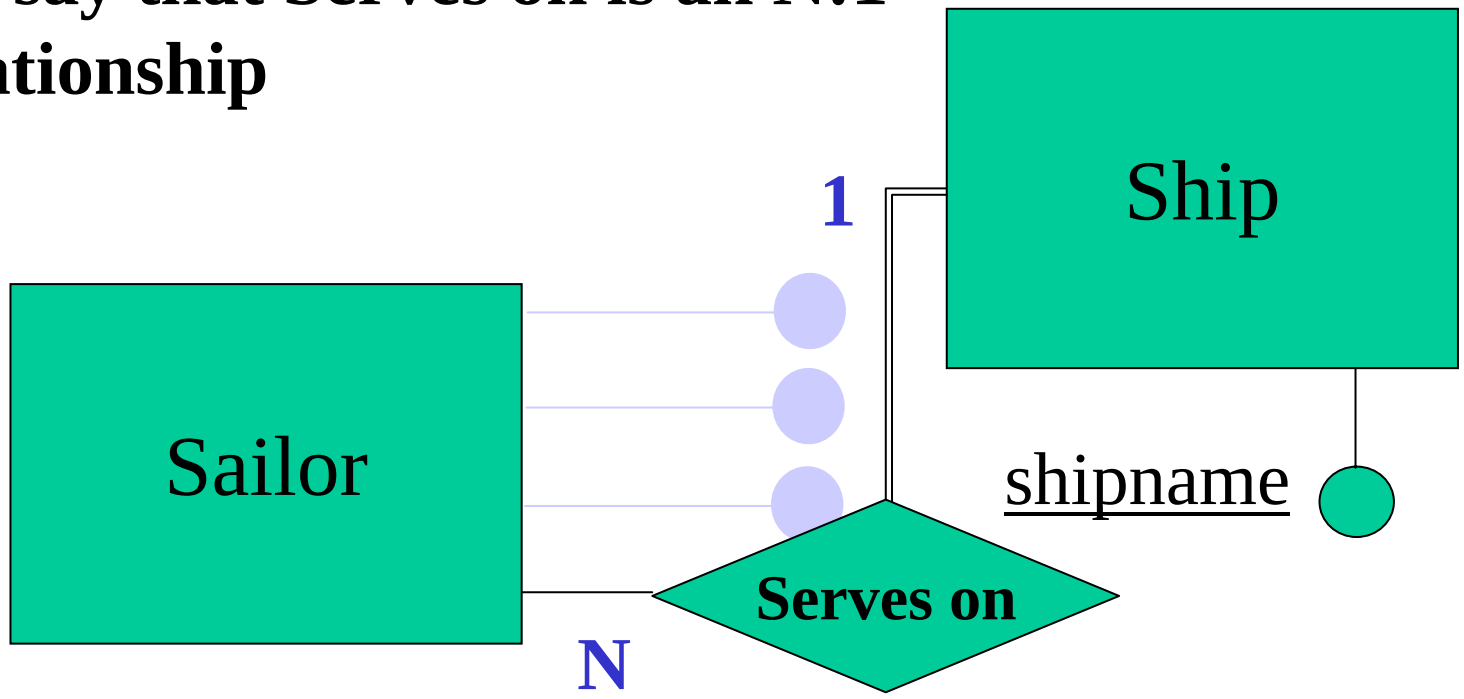
Cardinality **N**



Cardinality **1**

A Ship may have more than one Sailor serving on it

We say that Serves on is an N:1 relationship



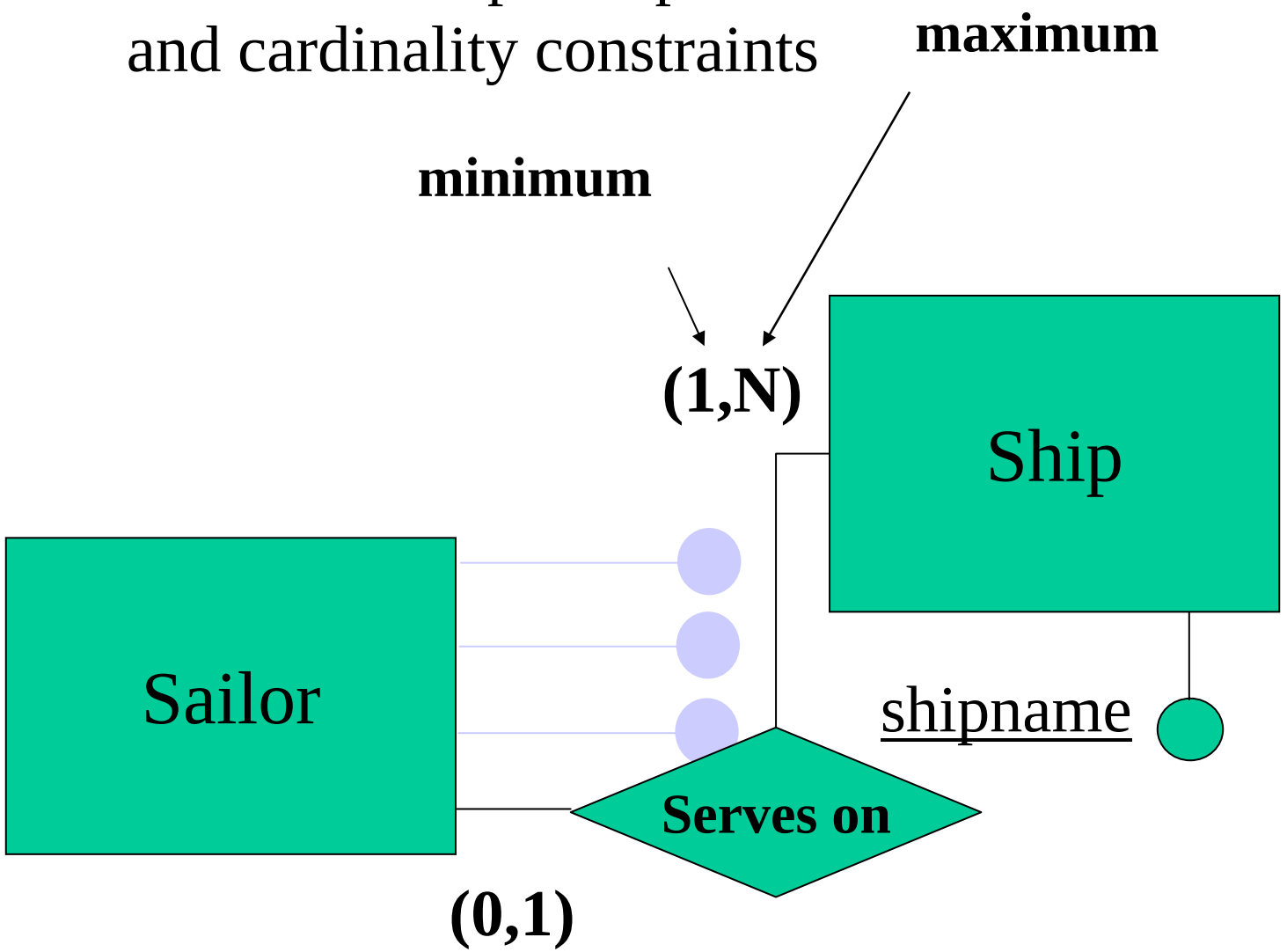
A Ship must have minimum one Sailor, but may have *several* (**N**) Sailors serving on it

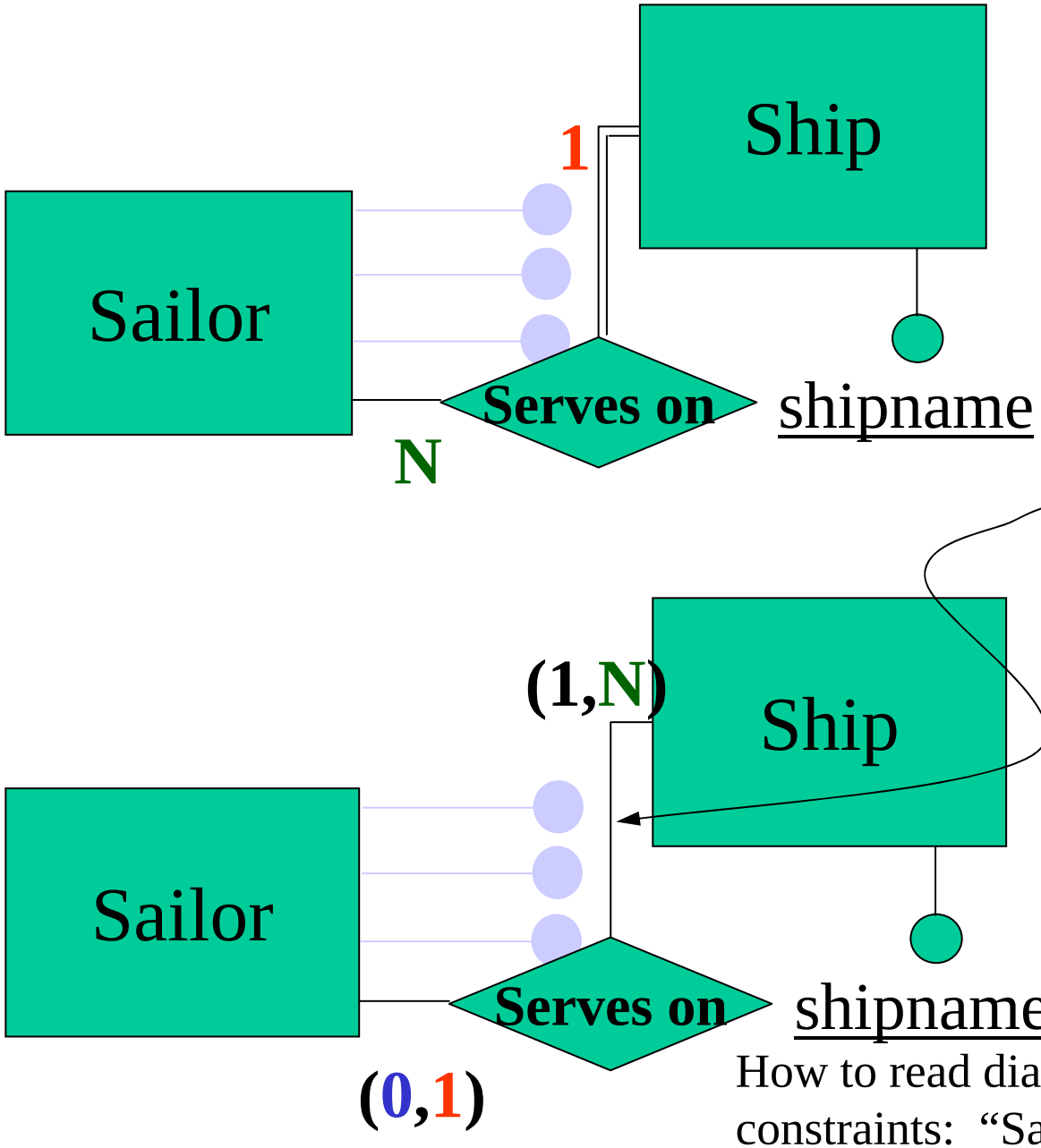
A Sailor may serve on a ship. A Sailor may not serve on more then *one* (**1**) ship at a time

How to read the diagram: Sailor - Serves on - 1 - Ship

Ship has - serving on it - N - Sailors

Alternative (but equivalent) representations exist for participation and cardinality constraints



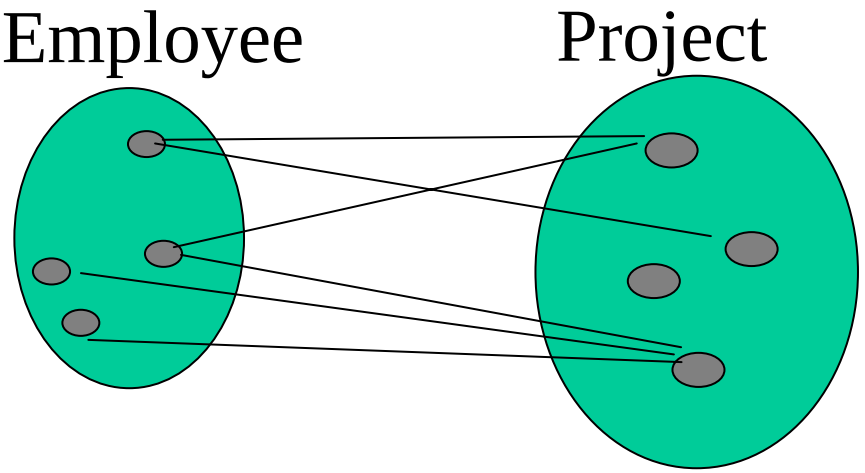


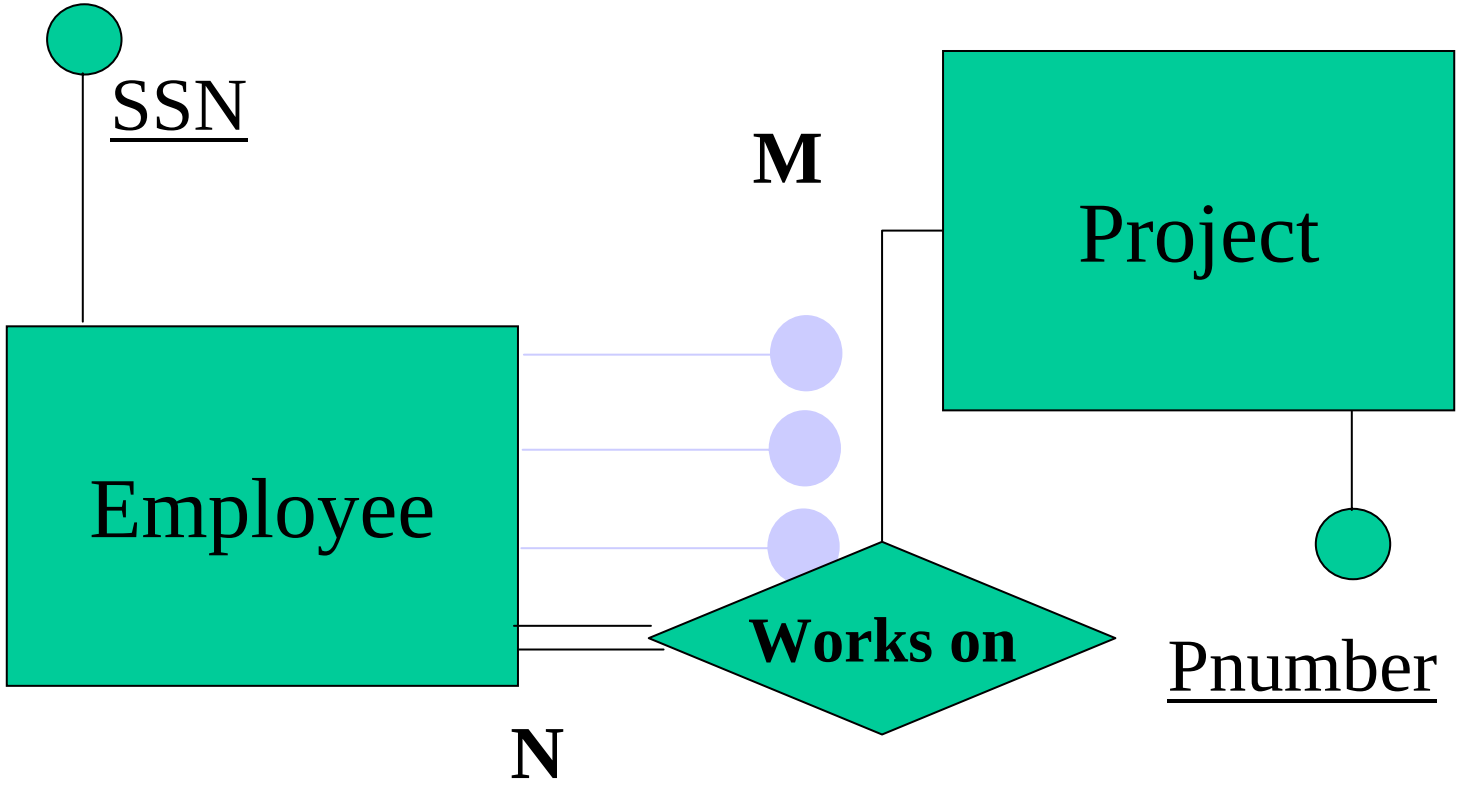
Equivalence of the two notations

Note: the double-line notation was **not** used here for total participation, because ‘minimum 1’ already means total participation.

How to read diagram with min/max constraints: “Sailor appears min 0 and max 1 times in the ‘Serves-on’ relation”

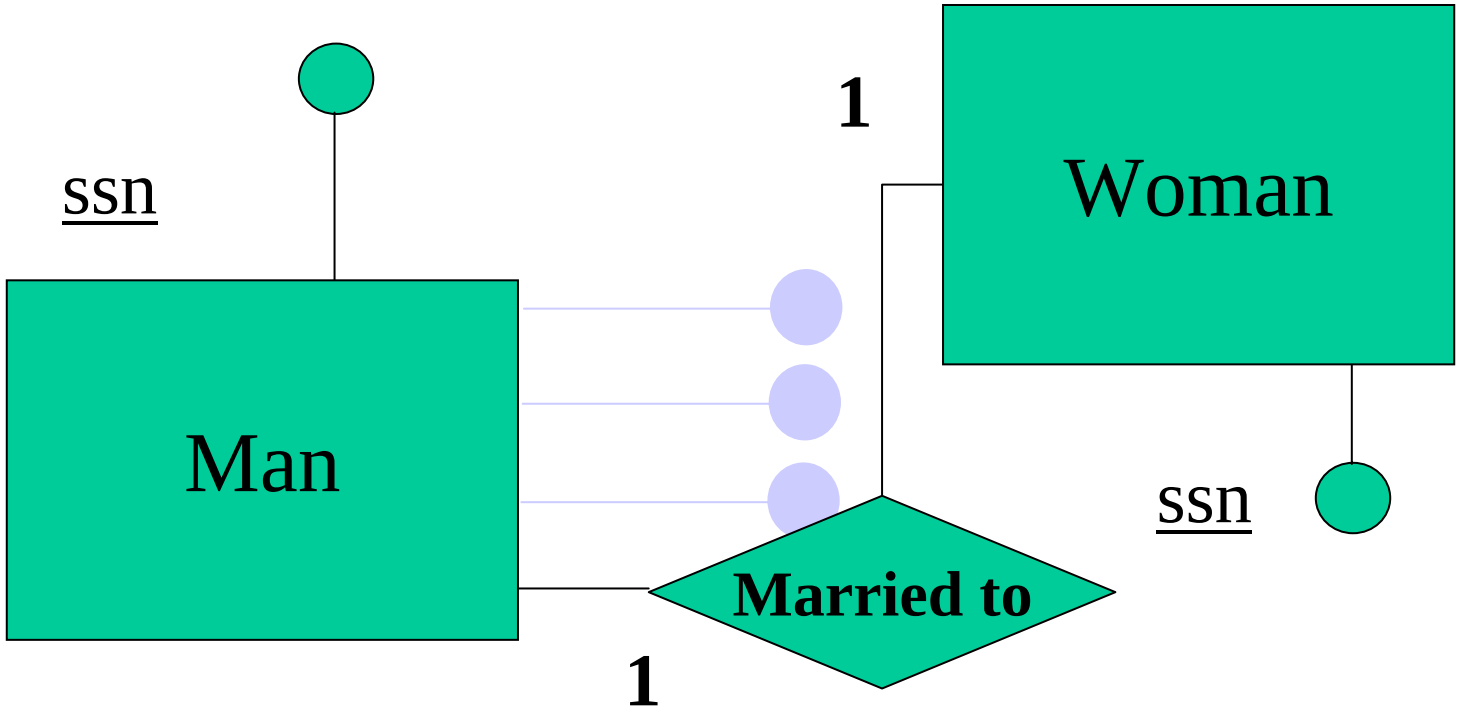
N:M Relations





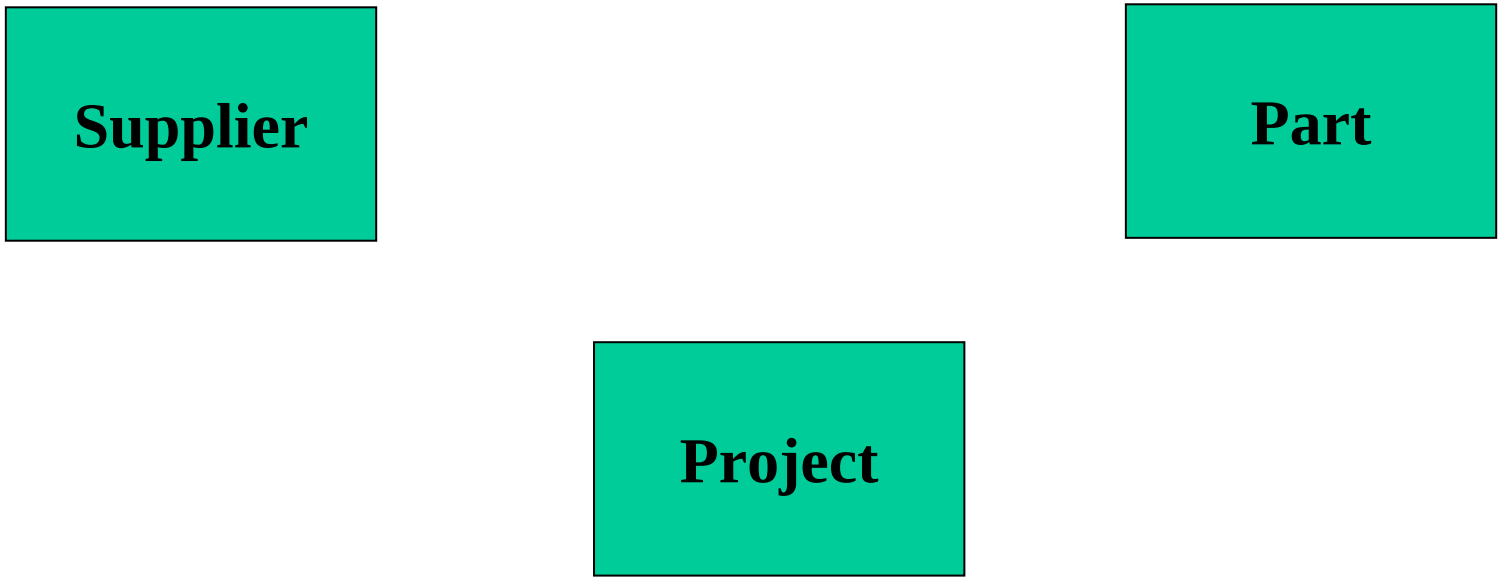
1:1 Relations

Example of 1:1 relation (relationship type) - typical in *some* universes of discourse)



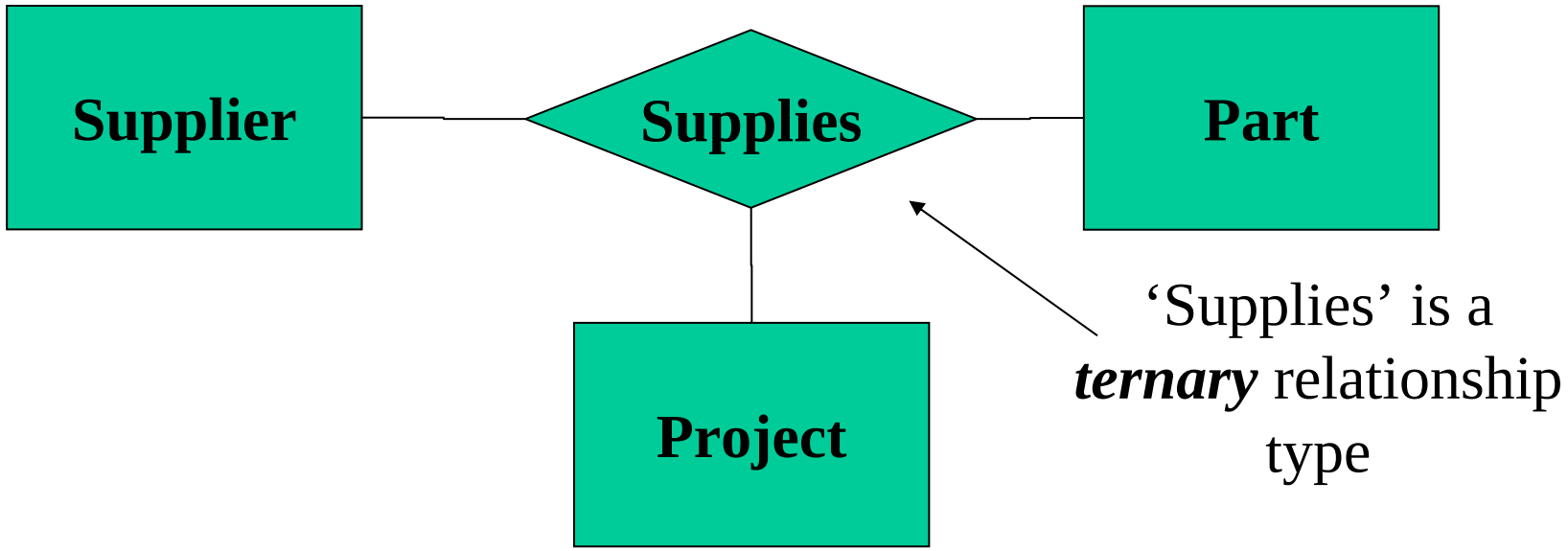
Higher order relationships

- Up till now we represented relations which existed between two entity sets. Every relationship instance was a ‘binary fact’ establishing a relationship between *two* entity instances (one ship and one sailor, for example)
- However, consider ‘Supplier supplies Part to Project’



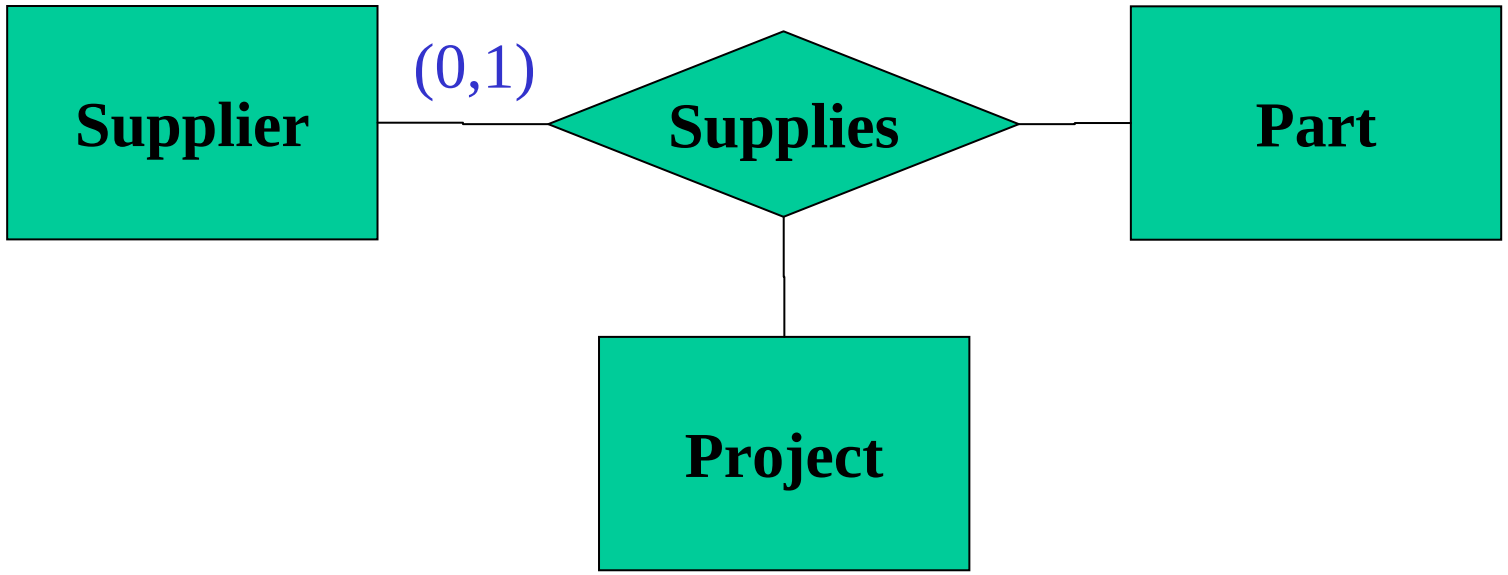
Higher order relations

- "Supplies" can not be decomposed into two elementary fact types: e.g. 'S_1 supplies $Part_1$' and '$Part_1$ is used by $Proj_1$' loses information about which supplier supplied $Part_1$ to $Project_1$! (e.g. it could be that S_2 supplied $Part_1$ to $Project_1$)



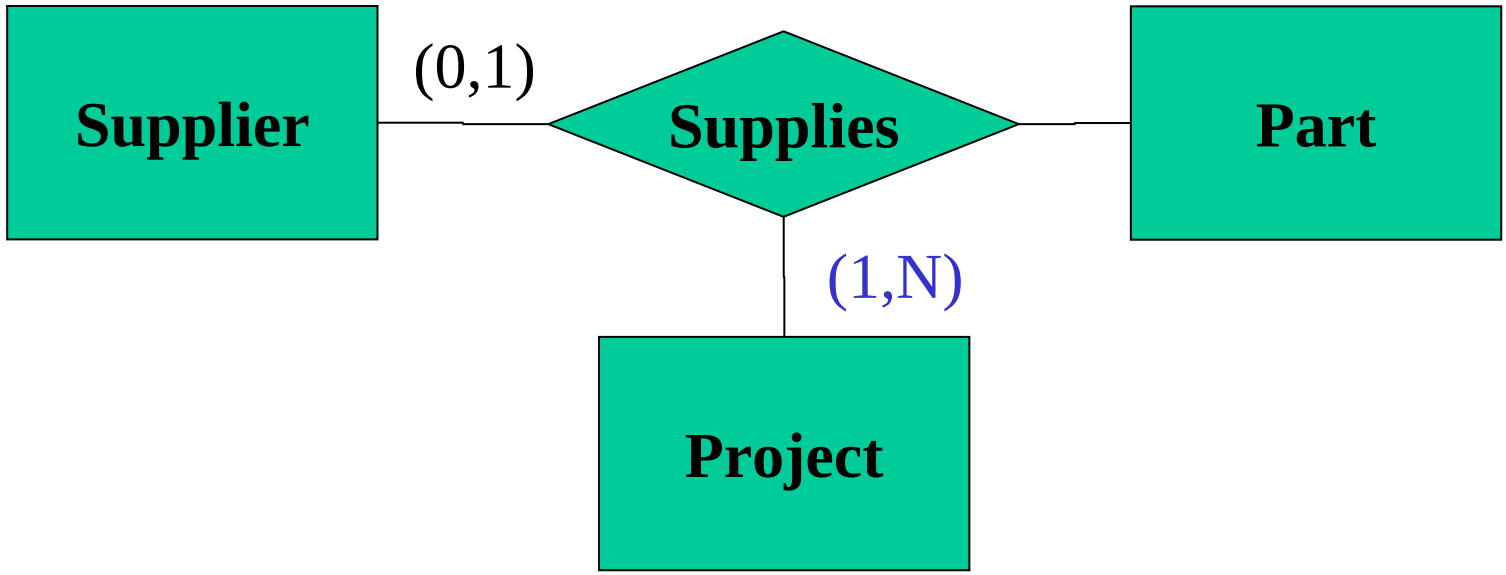
Representing constraints on higher order relations...

- Assume that if a supplier supplies a part to a project then it is not allowed to supply anything to any other project.
  Assume also that every project must have at least one part supplied to it. Parts may be represented in the database even if no project uses them at the moment.
- Represent the participation and cardinality constrains:



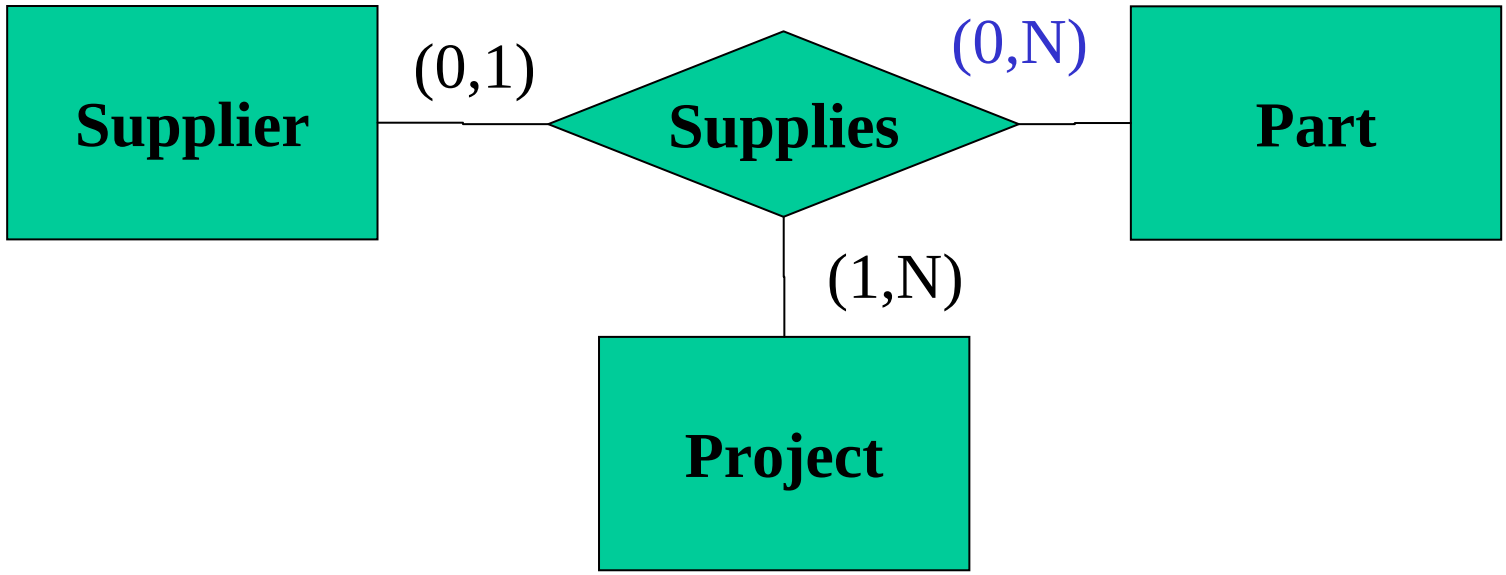
Representing constraints on higher order relations...

- Assume that if a supplier supplies a part to a project then it is not allowed to supply anything to any other project. Assume also that every project must have at least one part supplied to it, and Parts may be represented in the database even if no project uses them at the moment.
- Represent the participation and cardinality constrains:

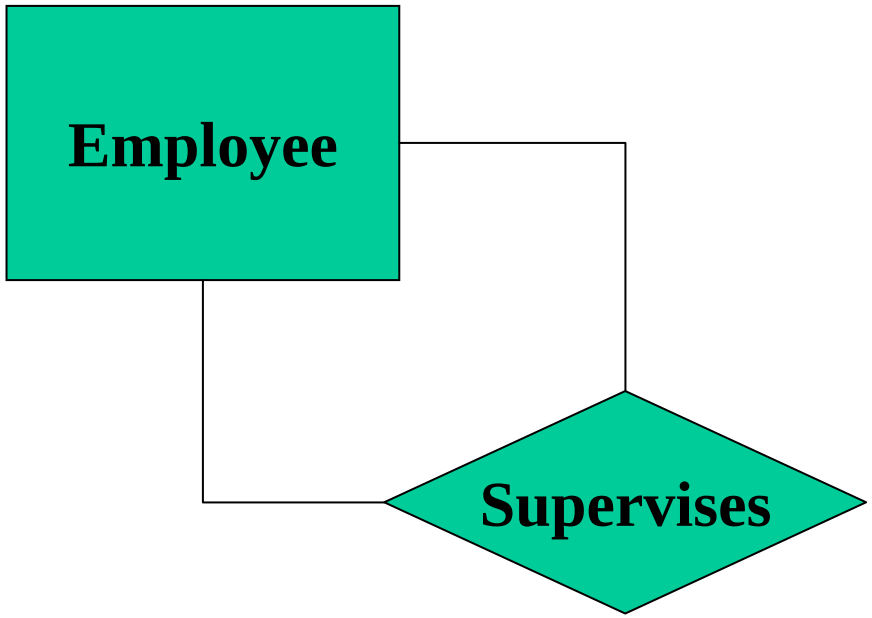


Representing constraints on higher order relations...

- Assume that if a supplier supplies a part to a project then it is not allowed to supply anything to any other project. Assume also that every project must have at least one part supplied to it, and Parts may be represented in the database even if no project uses them at the moment.
- Represent the participation and cardinality constrains:

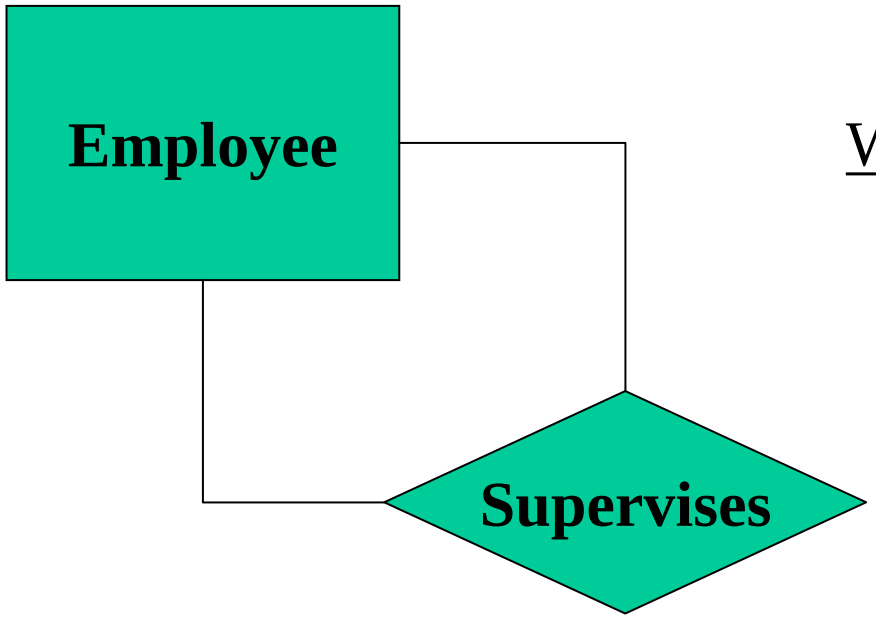


Roles



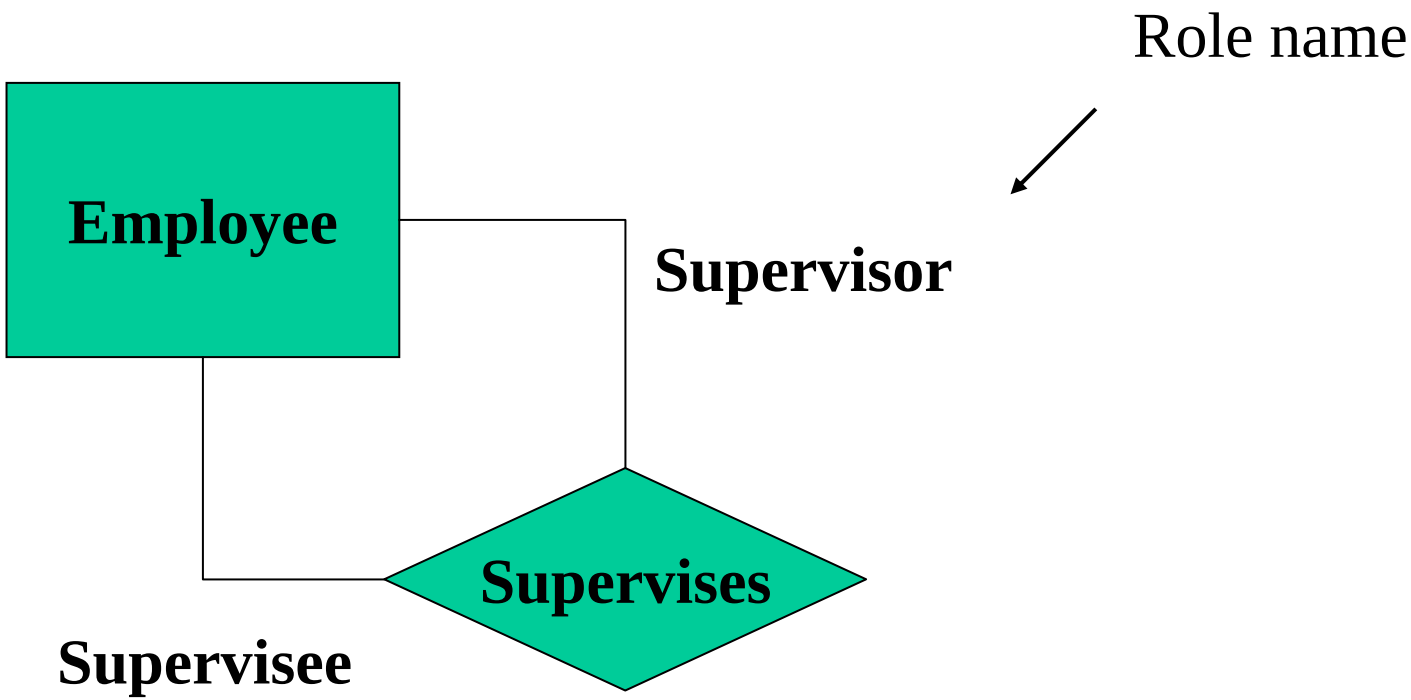
Roles

E.g. Supervises(John,Paul)



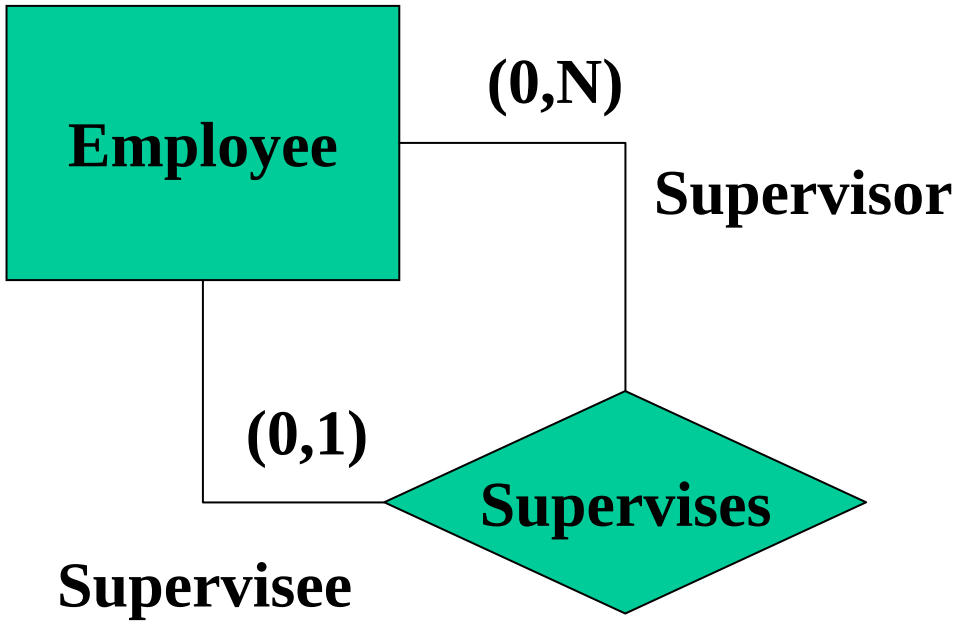
Who supervises whom?

Roles



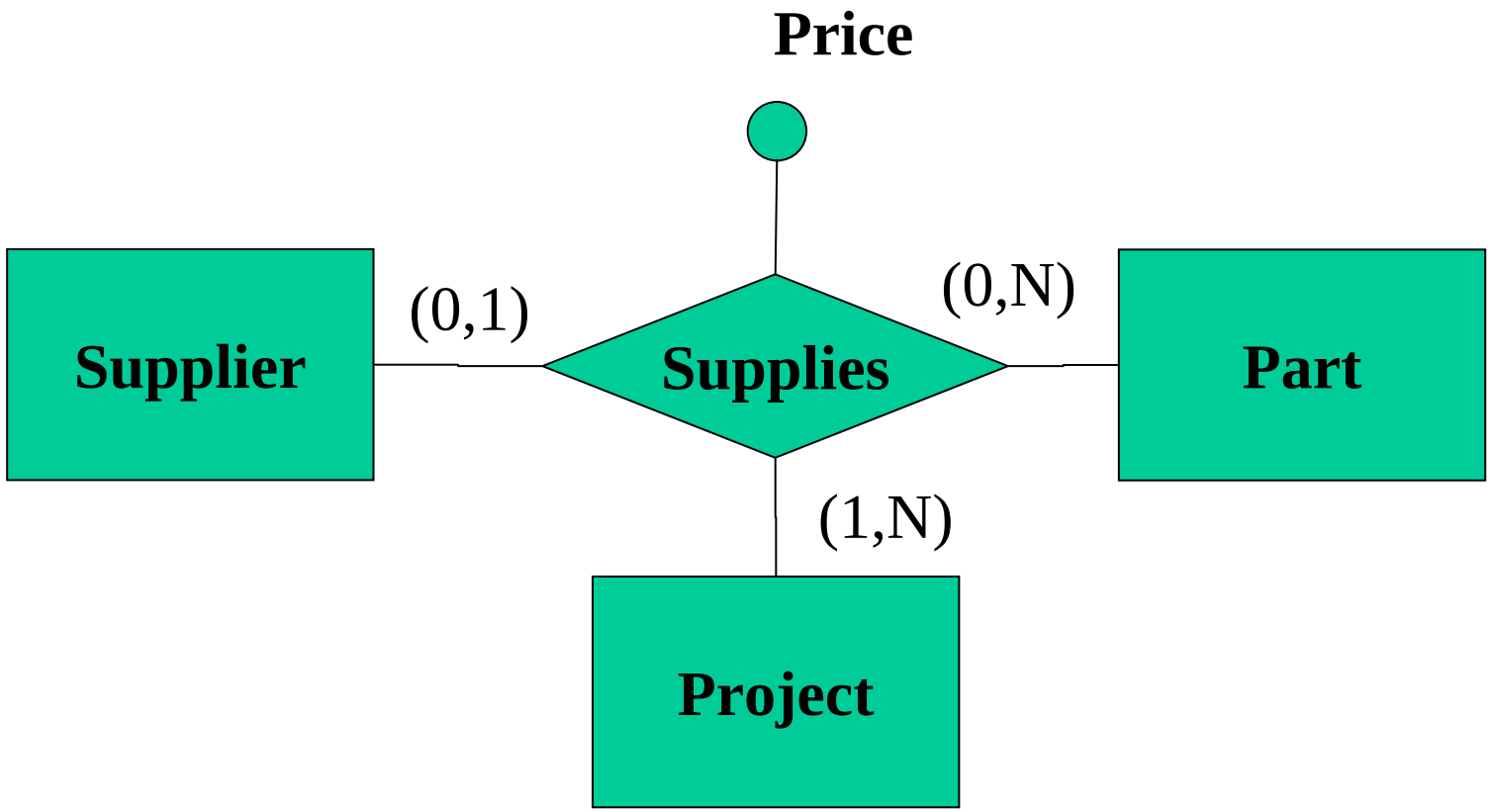
Supervises(supervisor:John, supervisee:Paul)

We represent constraints as with any other relation



Attributes of relationship types

Price is not an attribute of any of the entities, it is an attribute of the relationship type “Supplies”. (The price is determined for the ‘deal’)



Summary

- We have introduced the following concepts:
- Entity
- Attribute (simple, set valued, complex)
- Attribute domain
- Key (candidate, primary)
- Relationship
- Participation and cardinality constraints
- Binary, ternary, higher order relationship types
- Roles
- Attributes of relationship types

The end