Entity-Relationship Diagrams and FOL

Alessandro Artale

Free University of Bozen-Bolzano
Faculty of Computer Science
http://www.inf.unibz.it/˜artale

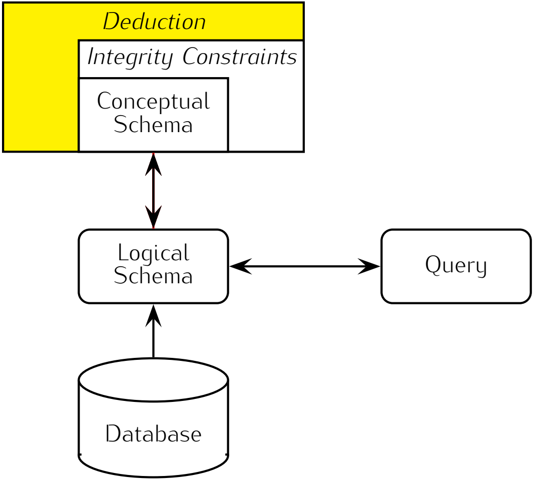
Descrete Mathematics and Logic — BSc course

Thanks to Prof. Enrico Franconi for provoding the slides

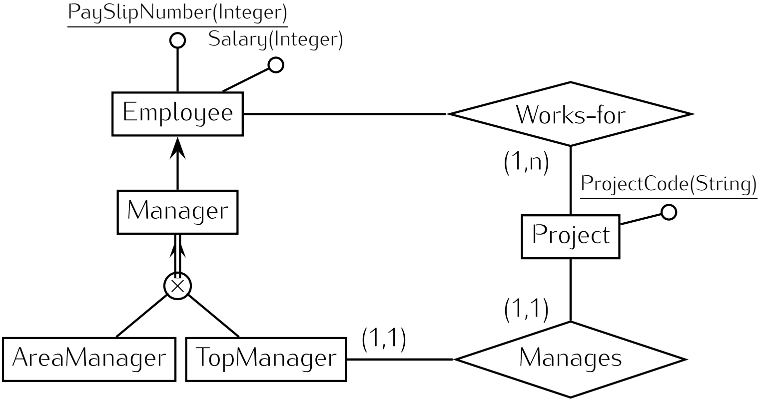
What is a Conceptual Schema

- A conceptual schema is a formal conceptualisation of the world.
- A conceptual schema specifies a set of *constraints*, which declare what should necessarily hold in any possible database.
- Given a conceptual schema, a *legal database* is a database satisfying the constraints.

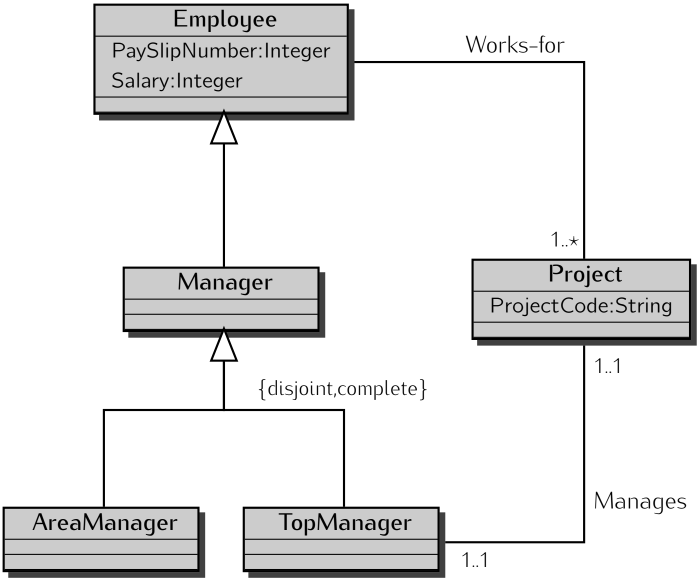
The Architecture of a Database



Entity-Relationship Diagram



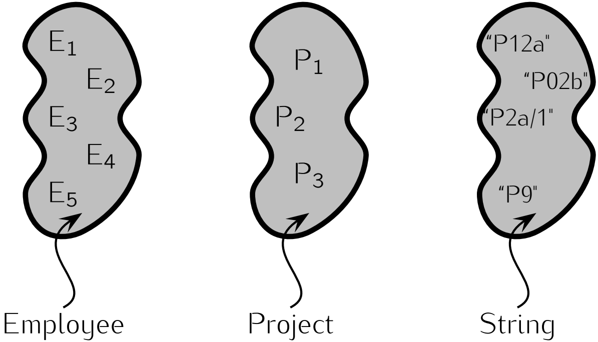
UML Class Diagram



Meaning of Basic Constructs

In a specific legal database:

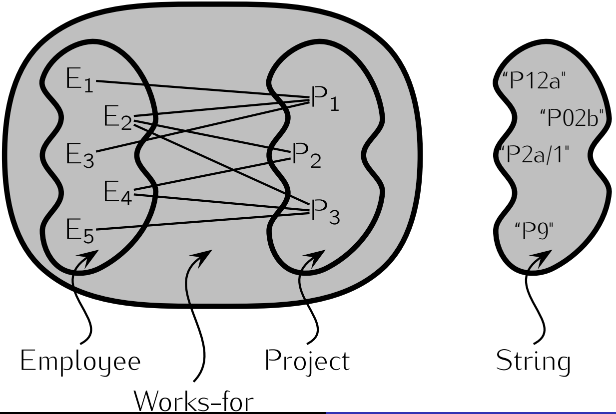
- An entity is a ***set of abstract instances***;
- a n-ary relationship is a ***set of n-tuple of abstract instances***;
- an attribute is a ***set of pairs of an abstract instance and a concrete domain element***.



Meaning of Basic Constructs

In a specific legal database:

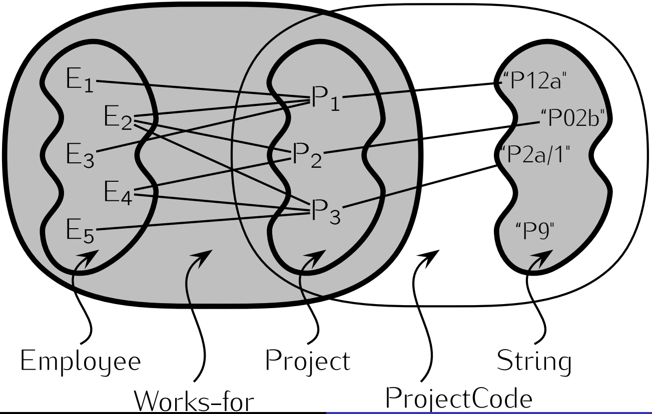
- An entity is a ***set of abstract instances***;
- a n-ary relationship is a ***set of n-tuple of abstract instances***;
- an attribute is a ***set of pairs of an abstract instance and a concrete domain element***.



Meaning of Basic Constructs

In a specific legal database:

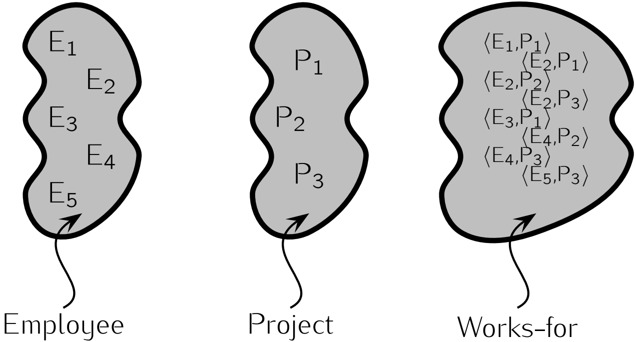
- An entity is a ***set of abstract instances***;
- a n-ary relationship is a ***set of n-tuple of abstract instances***;
- an attribute is a ***set of pairs of an abstract instance and a concrete domain element***.



To distinguish between concrete and abstract values we partition the interpretation domain:

- $\Delta = \Omega \cup \Delta_{\mathcal{D}}$, where
- Ω is the set of abstract instances, and
- $\mathcal{D}$ is the set of concrete values, i.e., $\mathcal{D} = \mathit{Int} \cup \mathit{String} \cup \ldots$

Relations as Sets of Tuples



The Relational Representation

Employee

employeeId
E_1
E_2
E_3
E_4
E_5

Project

projectId
P_1
P_2
P_3

String

anystring
"P12a"
"P02b"
"P2a/1"
"P9"
...

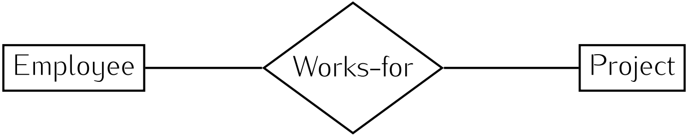
Works-for

employeeId	*projectId*
E_1	P_1
E_2	P_1
E_2	P_2
E_2	P_3
E_3	P_1
E_4	P_2
E_4	P_3
E_5	P_3

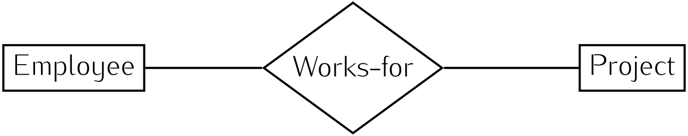
ProjectCode

projectId	*pcode*
P_1	"P12a"
P_2	"P02b"
P_3	"P2a/1"

Meaning of Relationships

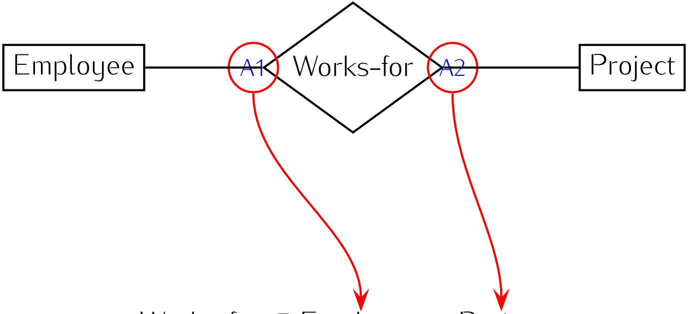


Meaning of Relationships



$$\text{Works-for} \subseteq \text{Employee} \times \text{Project}$$

Meaning of Relationships



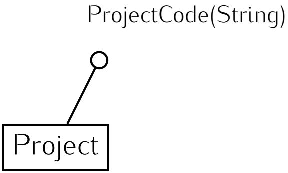
$$\text{Works-for} \subseteq \text{Employee} \times \text{Project}$$

Meaning of Attributes

An Attribute models a local concrete property of a Class.
It is characterized by:

- a name (which is unique only in the class it belongs to)
- a type (a set of possible concrete values, e.g., integer, string, etc.)
- and possibly a multiplicity (usually it is mandatory).

Meaning of Attributes (Cont.)

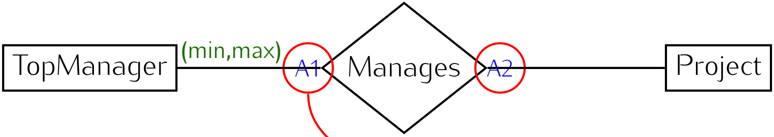


$$\text{Project} \subseteq \{e \in \Omega \mid \sharp\{\text{ProjectCode} \cap (\{e\} \times \texttt{String})\} \geq 1\}$$
$$\text{ProjectCode} \cap (\text{Project} \times \mathcal{D}) \subseteq \text{Project} \times \mathcal{D}_D$$

Note 1. The notation $\sharp\{\ldots\}$ means the cardinality of the set.
Note 2. The same attribute can be used in many entities possibly with a different range.

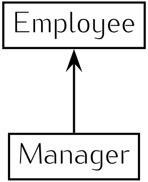
Meaning of Cardinality Constraints

Meaning of Cardinality Constraints



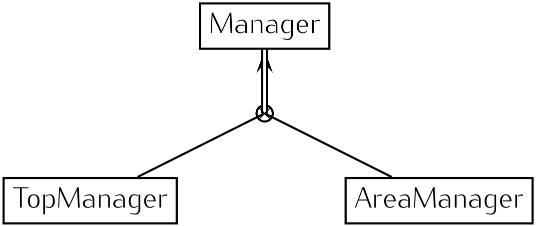
$$TopManager \subseteq \{e \in \Omega \mid max \geq \sharp\{Manages \cap (\{e\} \times \Omega)\} \geq min\}$$

Meaning of ISA

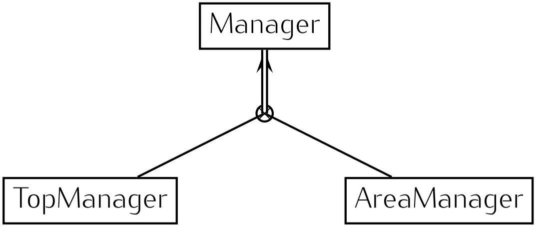


$$\text{Manager} \subseteq \text{Employee}$$

Meaning of *disjoint* and *total* Constraints



Meaning of *disjoint* and *total* Constraints



- *ISA:* AreaManager $\subseteq$ Manager
- *ISA:* TopManager $\subseteq$ Manager
- *disjoint:* AreaManager $\cap$ TopManager $= \emptyset$
- *total:* Manager $\subseteq$ AreaManager $\cup$ TopManager

Meaning of the initial diagram

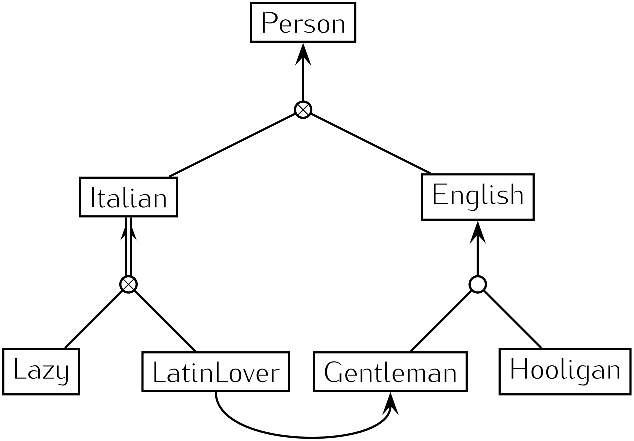
$$
\begin{aligned}
&\text{Works-for} \subseteq \text{Employee} \times \text{Project}\\
&\text{Manages} \subseteq \text{TopManager} \times \text{Project}\\
&\text{Employee} \subseteq \{e \in \Omega \mid \sharp\{\text{PaySlipNumber} \cap (\{e\} \times \texttt{Integer})\} \geq 1\}\\
&\text{Employee} \subseteq \{e \in \Omega \mid \sharp\{\text{Salary} \cap (\{e\} \times \texttt{Integer})\} \geq 1\}\\
&\text{Project} \subseteq \{e \in \Omega \mid \sharp\{\text{ProjectCode} \cap (\{e\} \times \texttt{String})\} \geq 1\}\\
&\text{TopManager} \subseteq \{e \in \Omega \mid 1 \geq \sharp\{\text{Manages} \cap (\{e\} \times \Omega)\} \geq 1\}\\
&\text{Project} \subseteq \{e \in \Omega \mid 1 \geq \sharp\{\text{Manages} \cap (\Omega \times \{e\})\} \geq 1\}\\
&\text{Project} \subseteq \{e \in \Omega \mid \sharp\{\text{Works-for} \cap (\Omega \times \{e\})\} \geq 1\}\\
&\text{Manager} \subseteq \text{Employee}\\
&\text{AreaManager} \subseteq \text{Manager}\\
&\text{TopManager} \subseteq \text{Manager}\\
&\text{AreaManager} \cap \text{TopManager} = \emptyset\\
&\text{Manager} \subseteq \text{AreaManager} \cup \text{TopManager}
\end{aligned}
$$

Inferences

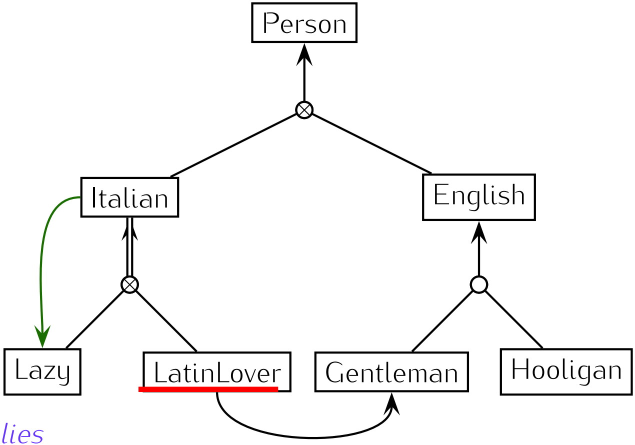
Given a collection of constraints, such as an Entity-Relationship diagram, it is possible that additional constraints can be inferred.

- An entity is inconsistent/unsatisfiable if it denotes the empty set in any legal database.
- An entity is a sub-entity of another entity if the former denotes a subset of the set denoted by the latter in any legal database.
- Two entities are equivalent if they denote the same set in any legal database.
- A stricter contraint is inferred – e.g., a **cardinality** contraint – if it holds in in any legal database.
- ...

Inferences (cont.)



Inferences (cont.)



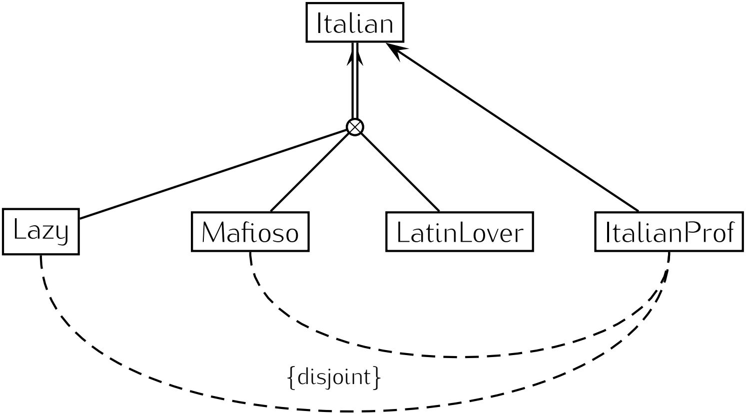
Implies

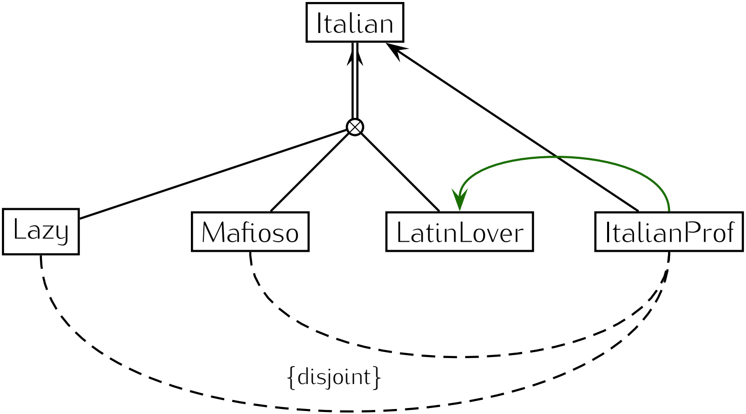
LatinLover $= \emptyset$ Then, LatinLover is an inconsistent entity.

Italian $\subseteq$ Lazy Then, Italian is a sub-entity of Lazy.

Italian $\equiv$ Lazy Then, Italian and Lazy are equivalent entities.

Inferences: Reasoning by cases

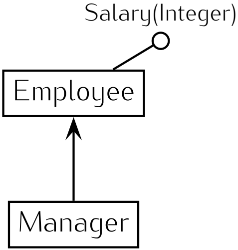


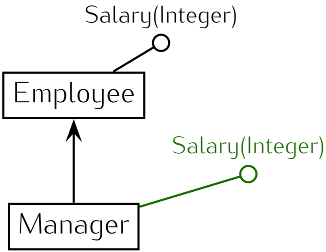


Implies

ItalianProf $\subseteq$ LatinLover Then, ItalianProf is a sub-entity of LatinLover.

Inferences: ISA and Inheritance

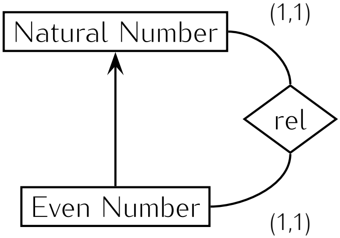




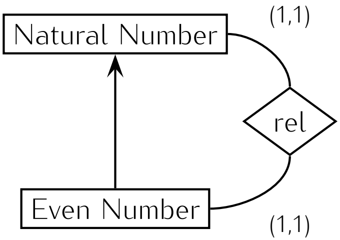
Implies

$$\text{Manager} \subseteq \{e \in \Omega \mid \sharp\{\text{Salary} \cap (\{e\} \times \texttt{Integer})\} \geq 1\}$$

Bijection bewteen Entities



Bijection bewteen Entities

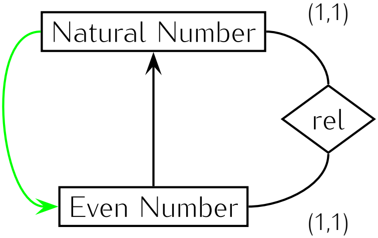


Implies

Since `rel` is a one-to-one correspondence, then:

"the entities *'Natural Number'* and *'Even Number'* contain the same number of instances".

Bijection bewteen Entities



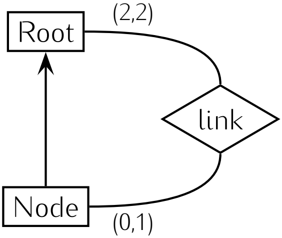
Implies

Since `rel` is a one-to-one correspondence, then:

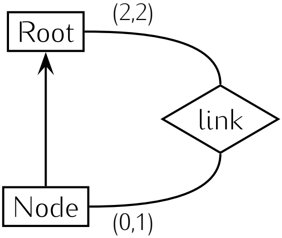
"the entities *'Natural Number'* and *'Even Number'* contain the same number of instances".

If the domain is finite: Natural Number $\equiv$ Even Number

Infinite Databases



Infinite Databases



Implies
"the classes Root and Node contain an infinite number of instances".
Note. If we admit just finite databases the above ER schema is unsatisfiable.

ER to FOL: Motivations

Show how a Conceptual Data Model can be mapped to a logical formalism.

Advantages:

- A clear semantics for the various ER constructs
- Ability to express complex integrity constraints
- Availability of decision procedures for consistency and logical implication in the data model.

Entity-Relationship and First Order Logic

- Entity-Relationship is a visual language to specify a set of constraints that should be satisfied by the relational database realising the ER diagram.
- The *interpretation* of an ER diagram is defined as the collection of all the *legal databases* – i.e., all the (finite) relational structures which conform to the constraints imposed by the conceptual schema.
- An ER diagram is mapped into a set of closed *First Order Logic* (FOL) formulas in such a way that the mapping preserves the semantics of the ER diagram:
  - The legal databases of an ER diagram are all the finite relational structures in which the translated set of FOL formulas evaluate to true.

ER Vs. FOL: The Alphabet

The Alphabet of the FOL language will have the following set of *Predicate* symbols:

- unary predicate symbols: $E_1, E_2, \ldots, E_n$ for each Entity-set; $D_1, D_2, \ldots, D_m$ for each Basic Domain.
- binary predicate symbols: $A_1, A_2, \ldots, A_k$ for each Attribute.
- n-ary predicate symbols: $R_1, R_2, \ldots, R_p$ for each Relationship-set.

FOL Notation

- *Vector variables* indicated as $\overline{x}$ stand for an n-tuple of variables: $\overline{x} = x_1, \ldots, x_n$
- *Counting existential quantifier* indicated as $\exists^{\leq n}$ or $\exists^{\geq n}$.

$$\begin{aligned}
&\exists^{\leq n} x.\, \varphi(x) \equiv \\
&\qquad \forall x_1, \ldots, x_n, x_{n+1}.\, \varphi(x_1) \land \ldots \land \varphi(x_n) \land \varphi(x_{n+1}) \rightarrow \\
&\qquad\qquad (x_1 = x_2) \lor \ldots \lor (x_1 = x_n) \lor (x_1 = x_{n+1}) \lor \\
&\qquad\qquad (x_2 = x_3) \lor \ldots \lor (x_2 = x_n) \lor (x_2 = x_{n+1}) \lor \\
&\qquad\qquad \ldots\ldots \lor (x_n = x_{n+1})
\end{aligned}$$

$$\begin{aligned}
&\exists^{\geq n} x.\, \varphi(x) \equiv \\
&\qquad \exists x_1, \ldots, x_n.\, \varphi(x_1) \land \ldots \land \varphi(x_n) \land \\
&\qquad\qquad \neg(x_1 = x_2) \land \ldots \land \neg(x_1 = x_n) \land \\
&\qquad\qquad \neg(x_2 = x_3) \land \ldots \land \neg(x_2 = x_n) \land \\
&\qquad\qquad \ldots\ldots \land \neg(x_{n-1} = x_n)
\end{aligned}$$

Interpretation: $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$, where Δ is an arbitrary non-empty set such that:

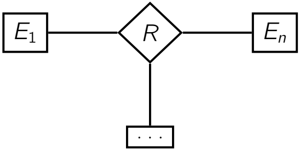
- $\Delta = \mathcal{D} \cup \Omega$, where:
  - $\mathcal{D} = \cup_{i=1}^{m} \mathcal{D}_{Di}$. $\mathcal{D}_{Di}$ is the set of values associated with each basic domain (i.e., integer, string, etc.); and $\mathcal{D}_{Di} \cap \mathcal{D}_{Dj} = \emptyset$, $\forall i, j.\, i \neq j$
  - Ω is the abstract entity domain such that $\mathcal{D} \cap \Omega = \emptyset$.

ER: The Formal Semantics for the Atoms

$\cdot^{\mathcal{I}}$ is the interpretation function that maps:

- *Basic Domain Predicates* to elements of the relative basic domain:
  $D_i^{\mathcal{I}} = \mathcal{D}_{Di}$ (e.g., $String^{\mathcal{I}} = \mathcal{D}_{String}$).
- *Entity-set Predicates* to elements of the entity domain:
  $E_i^{\mathcal{I}} \subseteq \Omega$.
- *Attribute Predicates* to binary relations such that:
  $A_i^{\mathcal{I}} \subseteq \Omega \times \mathcal{D}$.
- *Relationship-set Predicates* to n-ary relations over the entity domain:
  $R_i^{\mathcal{I}} \subseteq \Omega \times \Omega \ldots \times \Omega = \Omega^n$.

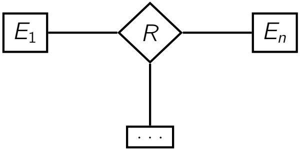
The Relationship Construct



- The meaning of this constraint is:

$$R^{\mathcal{I}} \subseteq {E_1}^{\mathcal{I}} \times \ldots \times {E_n}^{\mathcal{I}}$$

The Relationship Construct



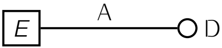
- The meaning of this constraint is:

$$R^{\mathcal{I}} \subseteq {E_1}^{\mathcal{I}} \times \ldots \times {E_n}^{\mathcal{I}}$$

- The FOL translation is the formula:

$$\forall x_1, \ldots, x_n.\, R(x_1, \ldots, x_n) \rightarrow E_1(x_1) \wedge \ldots \wedge E_n(x_n)$$

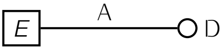
The Attribute Construct



- The meaning of this constraint is:

$$E^{\mathcal{I}} \subseteq \{e \in \Omega \mid \sharp\{A^{\mathcal{I}} \cap (\{e\} \times \mathcal{D}_{\mathtt{D}})\} \geq 1\}$$
$$A^{\mathcal{I}} \cap (E^{\mathcal{I}} \times \mathcal{D}) \subseteq A^{\mathcal{I}} \times \mathcal{D}_D$$

The Attribute Construct



- The meaning of this constraint is:

$$E^{\mathcal{I}} \subseteq \{e \in \Omega \mid \sharp\{A^{\mathcal{I}} \cap (\{e\} \times \mathcal{D}_{\mathtt{D}})\} \geq 1\}$$
$$A^{\mathcal{I}} \cap (E^{\mathcal{I}} \times \mathcal{D}) \subseteq A^{\mathcal{I}} \times \mathcal{D}_{D}$$

- The FOL translation is the formula:

$$\forall x. E(x) \to \exists y. A(x, y) \land D(y)$$

The Cardinality Construct

- The meaning of this constraint is:

$$E_1{}^{\mathcal{I}} \subseteq \{e \in \Omega \mid p \leq \sharp\{R^{\mathcal{I}} \cap (\{e\} \times \Omega)\} \leq q\}$$

The Cardinality Construct

- The meaning of this constraint is:

$$E_1{}^{\mathcal{I}} \subseteq \{e \in \Omega \mid p \leq \sharp\{R^{\mathcal{I}} \cap (\{e\} \times \Omega)\} \leq q\}$$

- The FOL translation is the formula:

$$\forall x. E(x) \rightarrow \exists^{\geq p} y. R(x,y) \wedge \exists^{\leq q} y. R(x,y)$$

The Cardinality Construct: An Example

A valid Database is:

Professor

professorId
Alex
Bob

Student

studentId
John
Mary
Nick
Paul
Laura

Supervises

professorId	*studentId*
Alex	John
Bob	Laura
Alex	Mary
Bob	Nick
Alex	Paul

The Cardinality Construct: An Example

An invalid Database is:

Professor

professorId
Alex
Bob

Student

studentId
John
Mary
Nick
Paul
Laura

Supervises

professorId	*studentId*
Alex	John
Bob	Laura
Alex	Mary
Bob	Nick
Alex	Paul
Alex	Laura

The Cardinality Construct: An Example

- The FOL translation is:

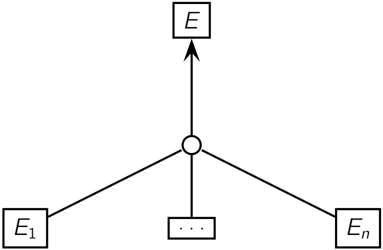
$$\forall x, y.\, \texttt{Supervises}(x, y) \rightarrow \texttt{Professor}(x) \land \texttt{Student}(y)$$
$$\forall x.\, \texttt{Professor}(x) \rightarrow \exists^{\geq 2} y.\, \texttt{Supervises}(x, y) \land \exists^{\leq 3} y.\, \texttt{Supervises}(x, y)$$
$$\forall y.\, \texttt{Student}(y) \rightarrow \exists^{=1} x.\, \texttt{Supervises}(x, y)$$

The **ISA** relation is a constraint that specifies *subentity sets*.

We distinguish between the following different ISA relations:

- Overlapping Partial;
- Overlapping Total;
- Disjoint Partial;
- Disjoint Total.

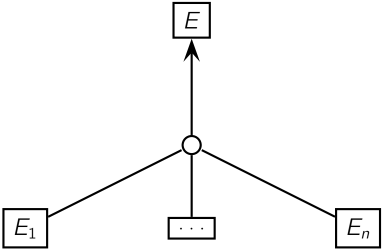
The Overlapping Partial Construct



- The meaning of this constraint is:

$$E_i^{\mathcal{I}} \subseteq E^{\mathcal{I}}, \text{ for all } i = 1, \ldots, n.$$

The Overlapping Partial Construct



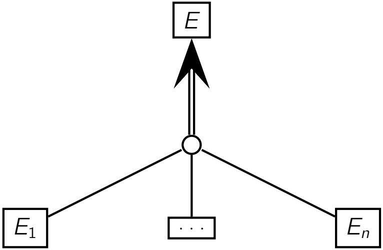
- The meaning of this constraint is:

$$E_i^{\mathcal{I}} \subseteq E^{\mathcal{I}}, \text{ for all } i = 1, \ldots, n.$$

- The FOL translation is the formula:

$$\forall x. E_i(x) \rightarrow E(x), \text{ for all } i = 1, \ldots, n.$$

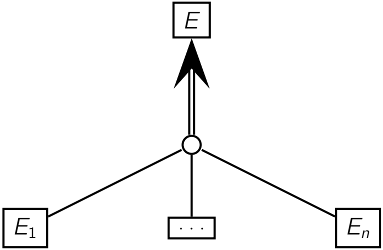
The Overlapping Total Construct



- The meaning of this constraint is:

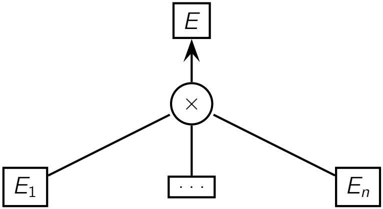
$$
\begin{array}{lll}
E_i^{\mathcal{I}} & \subseteq & E^{\mathcal{I}}, \quad \text{for all } i = 1, \ldots, n \\
E^{\mathcal{I}} & \subseteq & E_1^{\mathcal{I}} \cup \ldots \cup E_n^{\mathcal{I}}
\end{array}
$$

The Overlapping Total Construct



- The meaning of this constraint is:
  $E_i^{\mathcal{I}} \quad \subseteq \quad E^{\mathcal{I}}, \quad \text{for all } i = 1, \ldots, n$
  $E^{\mathcal{I}} \quad \subseteq \quad E_1^{\mathcal{I}} \cup \ldots \cup E_n^{\mathcal{I}}$
- The FOL translation is the set of formulas:
  $\forall x. E_i(x) \quad \rightarrow \quad E(x), \quad \text{for all } i = 1, \ldots, n$
  $\forall x. E(x) \quad \rightarrow \quad E_1(x) \vee \ldots \vee E_n(x)$

The Disjoint Partial Construct

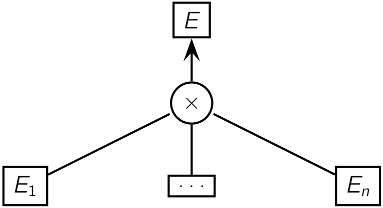


- The meaning of this constraint is:

  $E_i^{\mathcal{I}} \subseteq E^{\mathcal{I}}$ for all $i = 1, \ldots, n$

  $E_i^{\mathcal{I}} \cap E_j^{\mathcal{I}} = \emptyset$ for all $i \neq j$

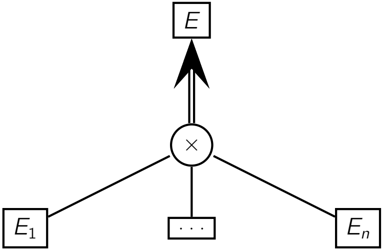
The Disjoint Partial Construct



- The meaning of this constraint is:
$$E_i^{\mathcal{I}} \subseteq E^{\mathcal{I}} \qquad \text{for all } i = 1, \ldots, n$$
$$E_i^{\mathcal{I}} \cap E_j^{\mathcal{I}} = \emptyset \qquad \text{for all } i \neq j$$

- The FOL translation is the set of formulas:
$$\begin{array}{lcl}
\forall x.\, E_1(x) & \rightarrow & E(x) \land \neg E_2(x) \land \ldots \land \neg E_n(x) \\
\forall x.\, E_2(x) & \rightarrow & E(x) \land \neg E_3(x) \land \ldots \land \neg E_n(x) \\
\forall x.\, E_{n-1}(x) & \rightarrow & E(x) \land \neg E_n(x) \\
\forall x.\, E_n(x) & \rightarrow & E(x)
\end{array}$$

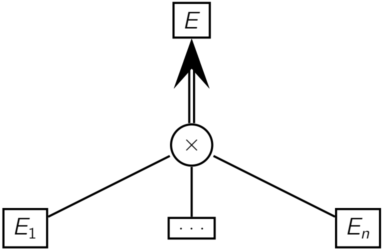
The Disjoint Total Construct



- The meaning of this constraint is:

$$E_i^{\mathcal{I}} \subseteq E^{\mathcal{I}} \quad \text{for all } i = 1, \ldots, n$$
$$E_i^{\mathcal{I}} \cap E_j^{\mathcal{I}} = \emptyset \quad \text{for all } i \neq j$$
$$E^{\mathcal{I}} \subseteq E_1^{\mathcal{I}} \cup \ldots \cup E_n^{\mathcal{I}}$$

The Disjoint Total Construct



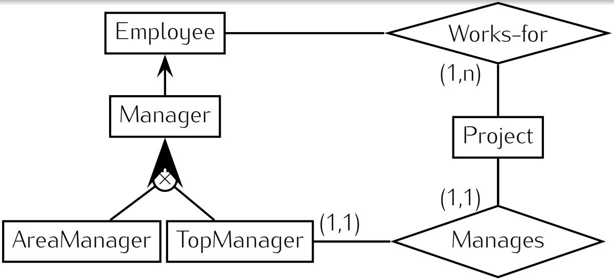
- The meaning of this constraint is:

$$
\begin{array}{ll}
E_i^{\mathcal{I}} \subseteq E^{\mathcal{I}} & \text{for all } i = 1, \dots, n \\
E_i^{\mathcal{I}} \cap E_j^{\mathcal{I}} = \emptyset & \text{for all } i \neq j \\
E^{\mathcal{I}} \subseteq E_1^{\mathcal{I}} \cup \dots \cup E_n^{\mathcal{I}} &
\end{array}
$$

- The FOL translation is the set of formulas:

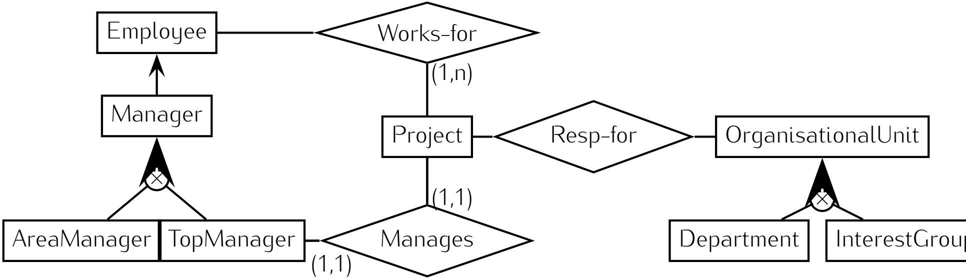
$$
\begin{array}{lcl}
\forall x.\, E(x) & \rightarrow & E_1(x) \lor \dots \lor E_n \\
\forall x.\, E_1(x) & \rightarrow & E(x) \land \neg E_2(x) \land \dots \land \neg E_n(x) \\
\forall x.\, E_2(x) & \rightarrow & E(x) \land \neg E_3(x) \land \dots \land \neg E_n(x) \\
\dots & \rightarrow & \dots \\
\forall x.\, E_n(x) & \rightarrow & E(x)
\end{array}
$$

FOL Translation: An Example



$$
\begin{array}{lcl}
\forall x, y.\, \texttt{Works-for}(x, y) & \rightarrow & \texttt{Employee}(x) \land \texttt{Project}(y) \\
\forall x, y.\, \texttt{Manages}(x, y) & \rightarrow & \texttt{Top-Manager}(x) \land \texttt{Project}(y) \\
\forall y.\, \texttt{Project}(y) & \rightarrow & \exists x.\, \texttt{Works-for}(x, y) \\
\forall y.\, \texttt{Project}(y) & \rightarrow & \exists^{=1} x.\, \texttt{Manages}(x, y) \\
\forall x.\, \texttt{Top-Manager}(x) & \rightarrow & \exists^{=1} y.\, \texttt{Manages}(x, y) \\
\forall x.\, \texttt{Manager}(x) & \rightarrow & \texttt{Employee}(x) \\
\forall x.\, \texttt{Manager}(x) & \rightarrow & \texttt{Area-Manager}(x) \lor \texttt{Top-Manager}(x) \\
\forall x.\, \texttt{Area-Manager}(x) & \rightarrow & \texttt{Manager}(x) \land \neg \texttt{Top-Manager}(x) \\
\forall x.\, \texttt{Top-Manager}(x) & \rightarrow & \texttt{Manager}(x)
\end{array}
$$

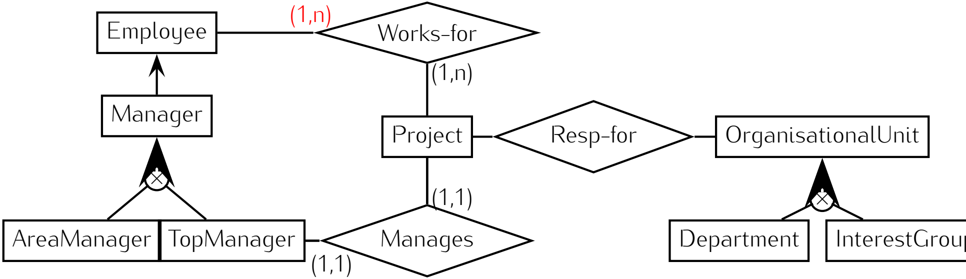
Additional (integrity) constraints



- Managers do not work for a project (she/he just manages it).

$$\forall x. \texttt{Manager}(x) \rightarrow \forall y. \neg \texttt{WORKS-FOR}(x, y)$$

Additional (integrity) constraints

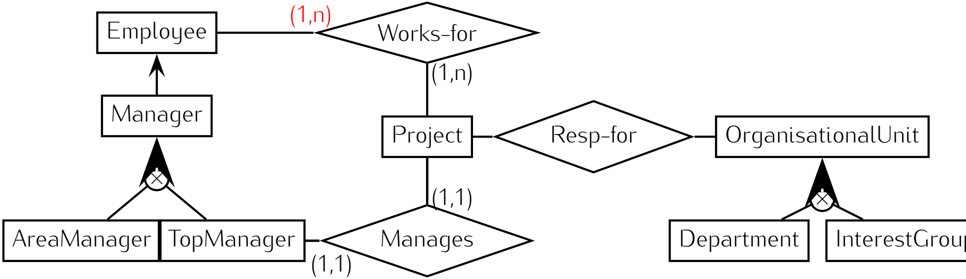


- Managers do not work for a project (she/he just manages it).

$$\forall x.\, \texttt{Manager}(x) \rightarrow \forall y.\, \neg \texttt{WORKS-FOR}(x, y)$$

- If the minimum cardinality for the participation of employees to the *works-for* relationship is increased, then . . .

Additional (integrity) constraints



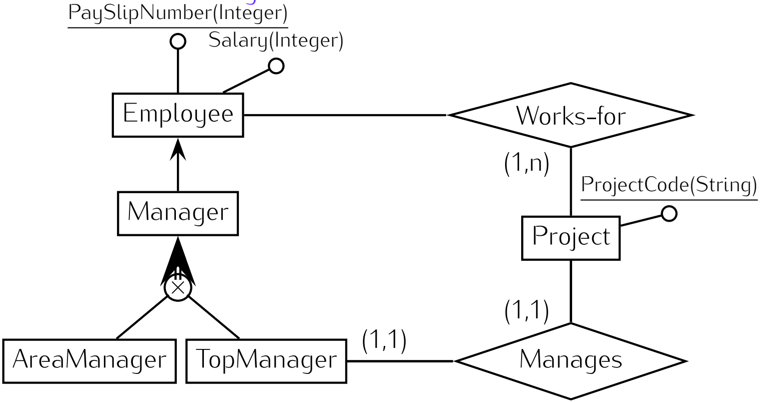
- Managers do not work for a project (she/he just manages it).

$$\forall x.\, \texttt{Manager}(x) \rightarrow \forall y.\, \neg\texttt{WORKS-FOR}(x, y)$$

- If the minimum cardinality for the participation of employees to the *works-for* relationship is increased, then ...
- If an ISA link is added stating that Interest Groups are Departments, then ...

Key constraints

A key is a set of attributes of an entity whose value uniquely identify elements of the entity itself.



$$\forall x.\ \big(\texttt{Project}(x) \rightarrow \exists^{=1} y.\ \texttt{ProjectCode}(x,y) \wedge \texttt{String}(y)\big)$$
$$\forall y.\ \big(\exists x.\ \texttt{ProjectCode}(x,y) \rightarrow \exists^{=1} x.\ \texttt{ProjectCode}(x,y) \wedge \texttt{Project}(x)\big)$$

Key constraints and relational schema

- According to ER modelling, a key must be specified for each entity.
- There is a one-to-one correspondence between (tuple) values of key attribute(s) and instances of an entity.
- This is why entities are mapped into the relational schema directly with the keys (which have concrete values) rather than with the abstract entity instances.
- Key values *are* the concrete representative for the instance of the entity.