Logic

The Meaning of Entity-Relationship Diagrams, part II

Enrico Franconi

`franconi@inf.unibz.it`

`http://www.inf.unibz.it/~franconi`

Faculty of Computer Science, Free University of Bozen-Bolzano

Motivation

Show how a Conceptual Data Model – like temporal Entity-Relationship – can be extended and mapped to an underlying 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 enriched 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.
- In order to formally define the interpretation, an ER diagram is mapped into a set of closed *First Order Logic* (FOL) formulas.
- 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:

- 1-ary 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 $\exists^{\geq n}$.

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

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

ER: The Interpretation function

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

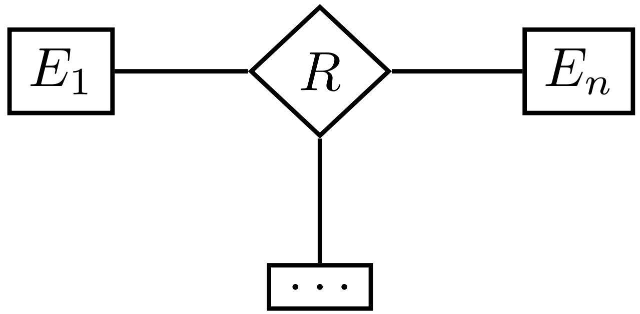
- $\mathbf{D} = \Omega \cup \mathcal{B}$, where:
  - $\mathcal{B} = \cup_{i=1}^{m} \mathcal{B}_{Di}$. $\mathcal{B}_{Di}$ is the set of values associated with each basic domain (i.e., integer, string, etc.); and $\mathcal{B}_{Di} \cap \mathcal{B}_{Dj} = \emptyset, \forall i, j . \, i \neq j$
  - Ω is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

ER: The Formal Semantics for the Atoms

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

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

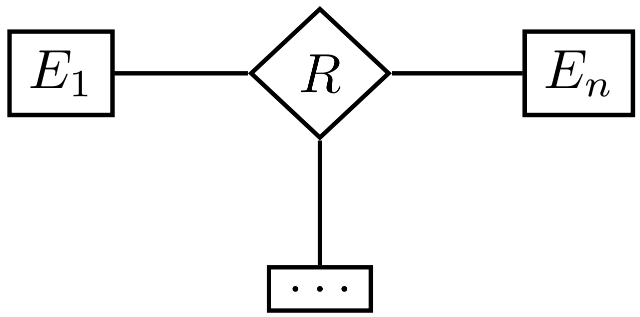
The Relationship Construct



- The meaning of this constraint is:

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

The Relationship Construct



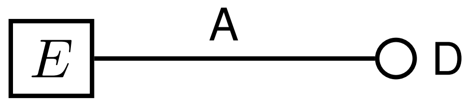
- The meaning of this constraint is:

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

- The FOL translation is the formula:

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

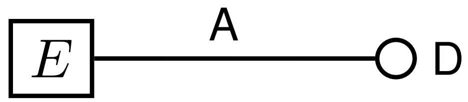
The Attribute Construct



- The meaning of this constraint is:

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

The Attribute Construct



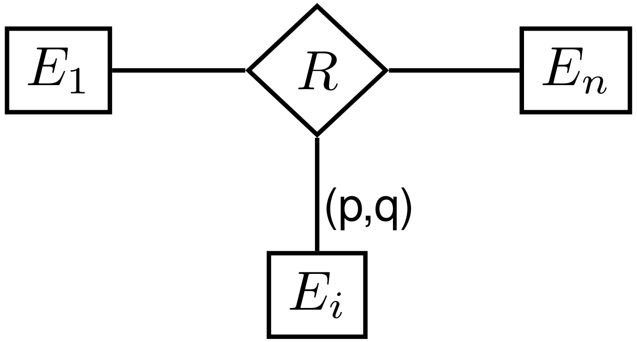
- The meaning of this constraint is:

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

- The FOL translation is the formula:

$$\forall x.\, E(x) \rightarrow \exists y. A(x,y) \wedge D(y)$$

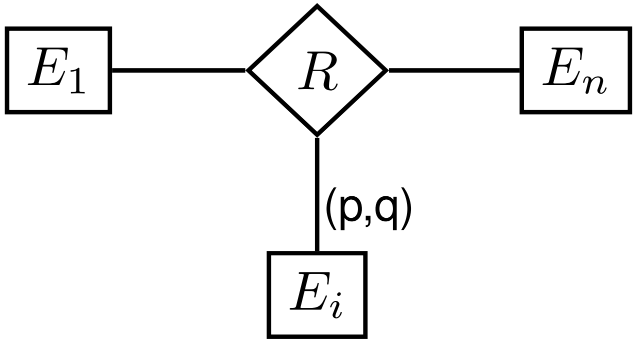
The Cardinality Construct



- The meaning of this constraint is:

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

The Cardinality Construct



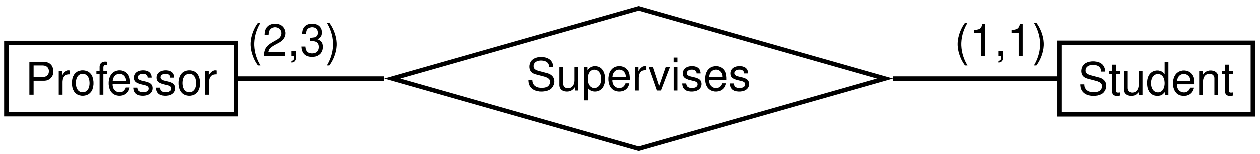
- The meaning of this constraint is:

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

- The FOL translation is the formula:

$$\begin{aligned} \forall x_i.\, E(x_i) \rightarrow\ & \exists^{\geq p} x_1, \ldots, x_{i-1}, x_{i+1}, \ldots x_n.\, R(x_1, \ldots, x_n) \wedge \\ & \exists^{\leq q} x_1, \ldots, x_{i-1}, x_{i+1}, \ldots x_n.\, R(x_1, \ldots, x_n) \end{aligned}$$

The Cardinality Construct: An Example



A valid Database is:

Professor

professorId
Alex
Bob

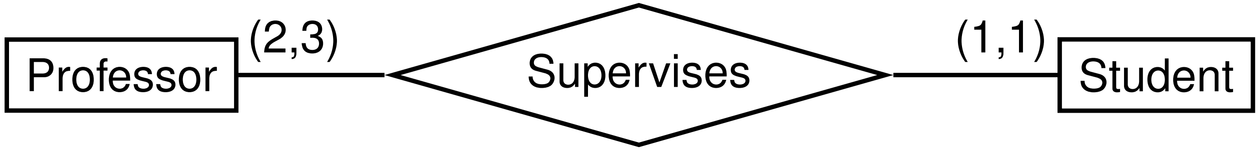
Student

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

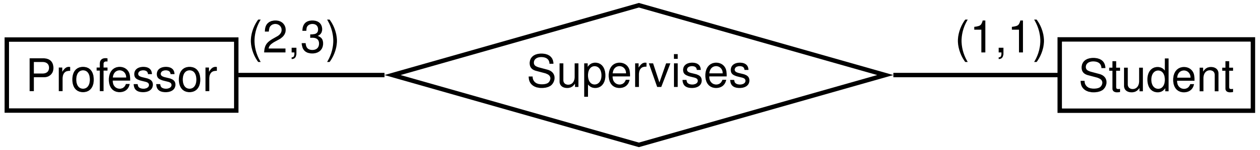
Student

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:

$$\forall x, y. \texttt{Supervises}(x, y) \rightarrow \texttt{Professor}(x) \wedge \texttt{Student}(y)$$

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

$$\forall y. \texttt{Student}(y) \rightarrow \exists^{=1} x. \texttt{Supervises}(x, y)$$

ISA Relations

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

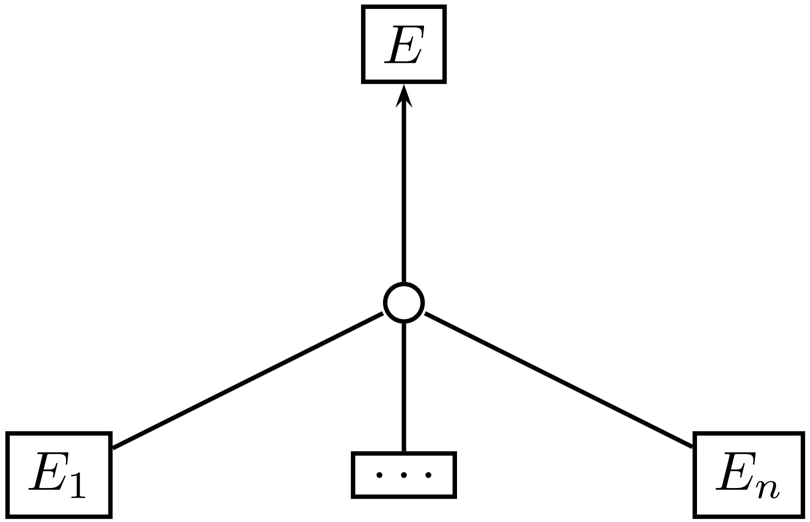
Subentity-set = contains entities with more properties – both more attributes and different participation in relationships – not pertinent to the Superentity-set.

A Subentity-set *inherits* all the properties of its Subentity-sets.

We distinguish between the following different ISA relations:

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

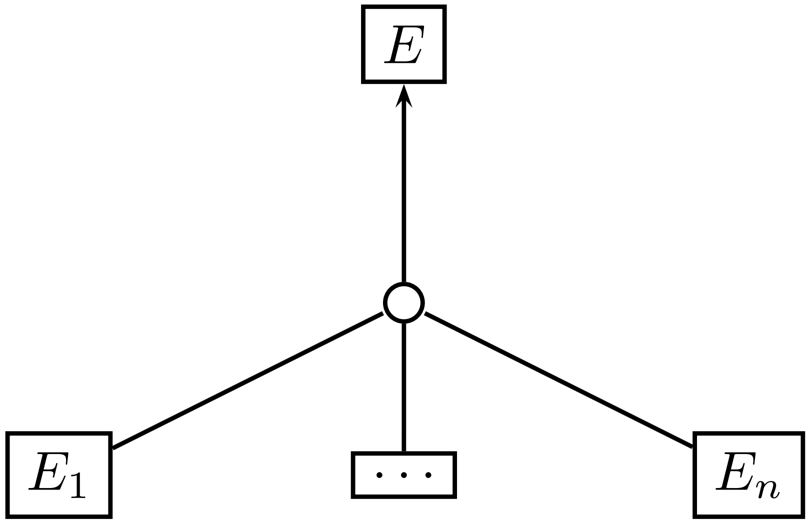
The Overlapping Partial Construct



- 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



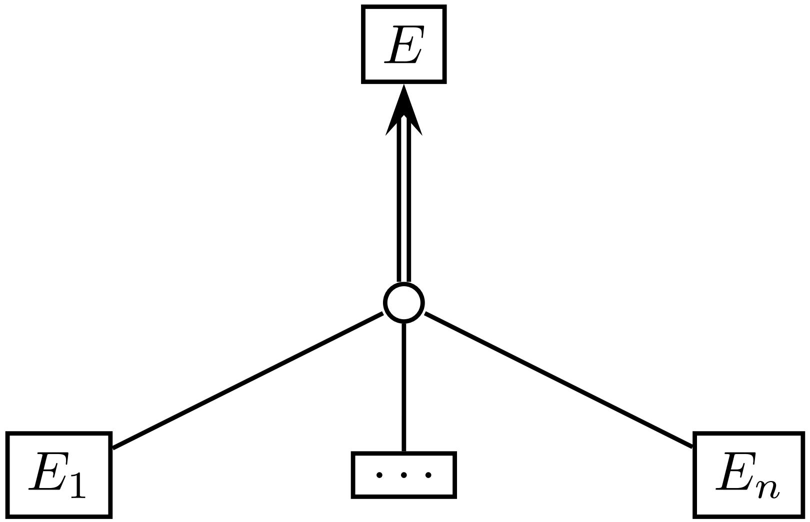
- The meaning of this constraint is:

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

- The FOL translation is the formula:

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

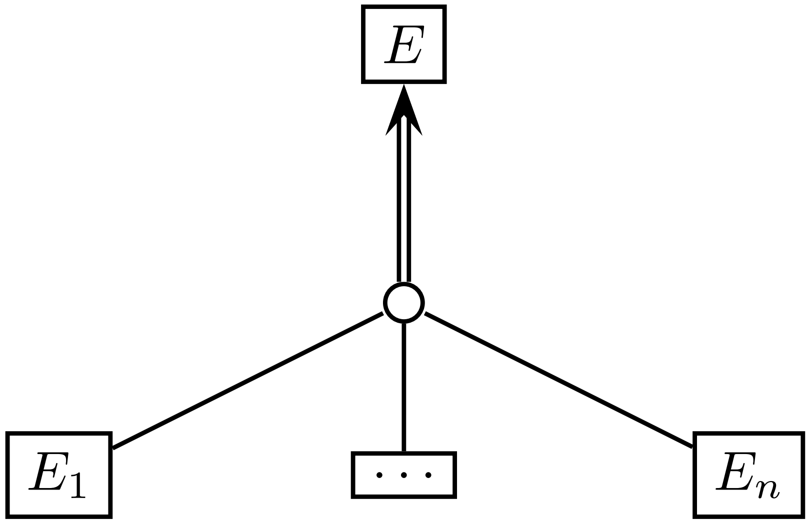
The Overlapping Total Construct



- The meaning of this constraint is:

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

The Overlapping Total Construct



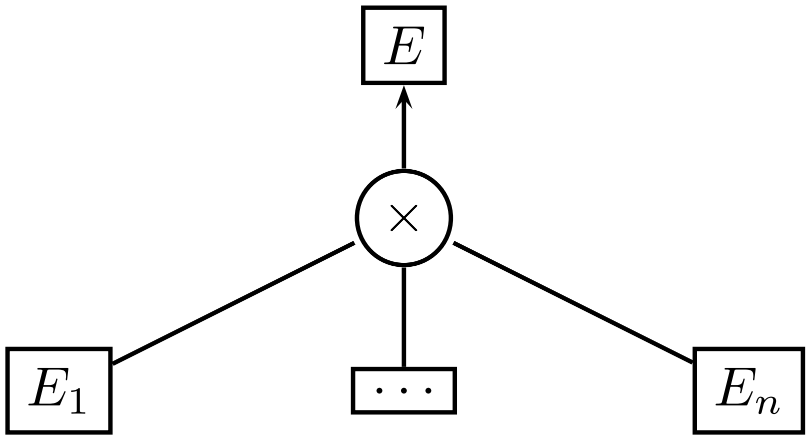
- The meaning of this constraint is:

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

- The FOL translation is the set of formulas:

$$
\begin{array}{rcl}
\forall x.\, E_i(x) & \rightarrow & E(x), \text{ for all } i = 1, \ldots, n \\
\forall x.\, E(x) & \rightarrow & E_1(x) \vee \ldots \vee E_n
\end{array}
$$

The Disjoint Partial Construct

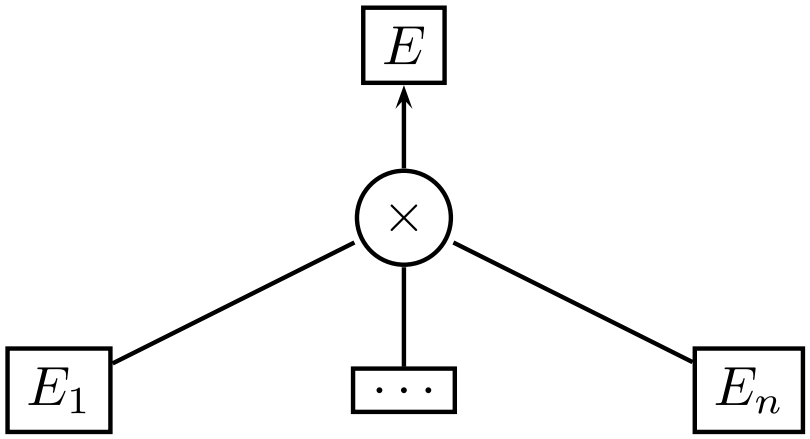


- The meaning of this constraint is:

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

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

The 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:

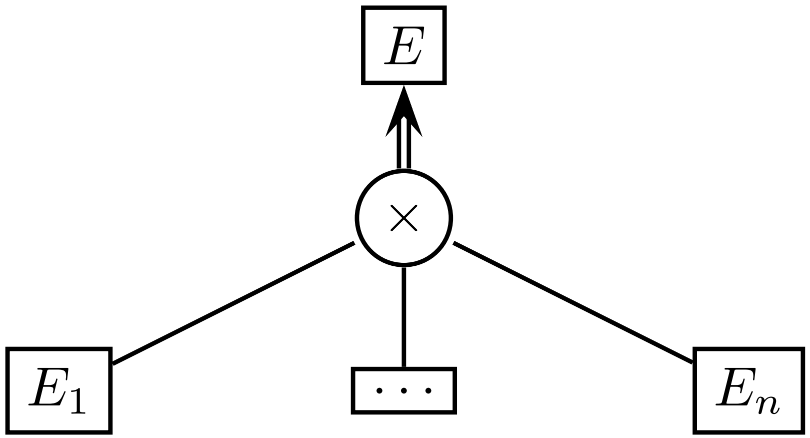
$$\forall x.\, E_1(x) \quad \rightarrow \quad E(x) \wedge \neg E_2(x) \wedge \ldots \wedge \neg E_n(x)$$

$$\forall x.\, E_2(x) \quad \rightarrow \quad E(x) \wedge \neg E_3(x) \wedge \ldots \wedge \neg E_n(x)$$

$$\forall x.\, E_{n-1}(x) \quad \rightarrow \quad E(x) \wedge \neg E_n(x)$$

$$\forall x.\, E_n(x) \quad \rightarrow \quad E(x)$$

The Disjoint Total Construct



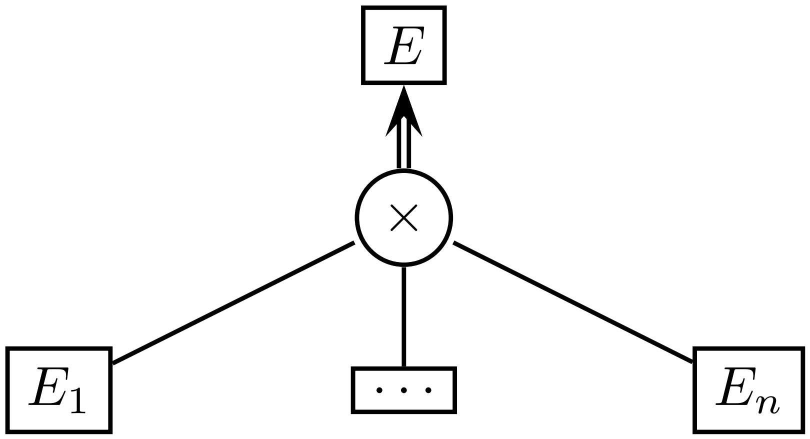
- The meaning of this constraint is:

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

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

  $E^{\mathcal{I}} \subseteq E_1{}^{\mathcal{I}} \cup \ldots \cup E_n{}^{\mathcal{I}}$

The Disjoint Total Construct



- The meaning of this constraint is:

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

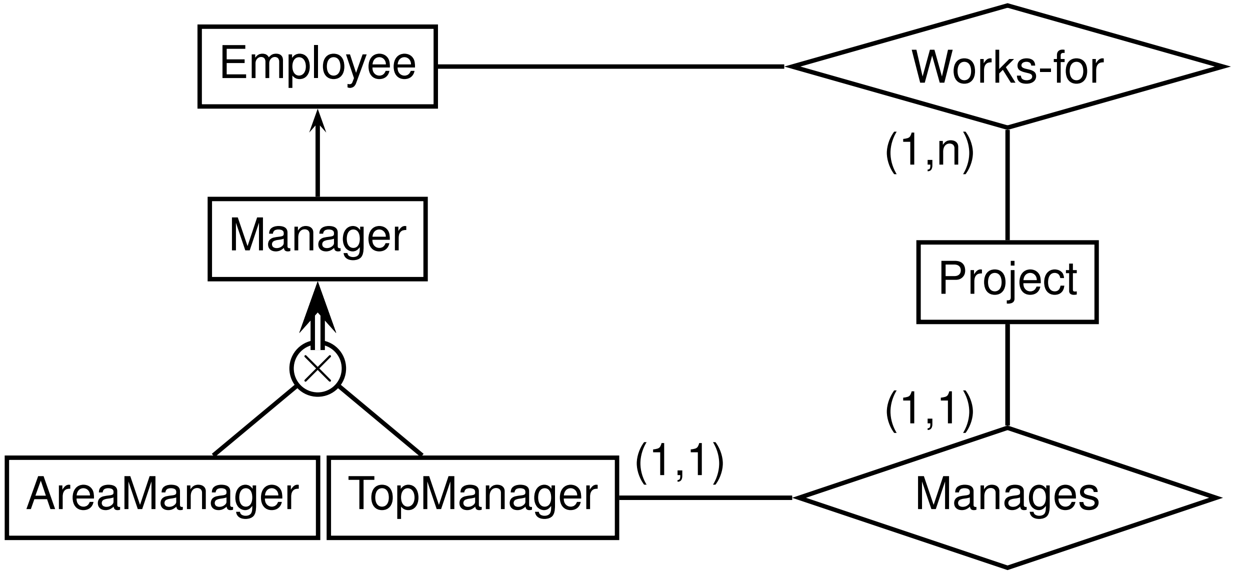
  $E_i{}^{\mathcal{I}} \cap E_j{}^{\mathcal{I}} = \emptyset$ for all $i \neq j$

  $E^{\mathcal{I}} \subseteq E_1{}^{\mathcal{I}} \cup \ldots \cup E_n{}^{\mathcal{I}}$

- The FOL translation is the set of formulas:

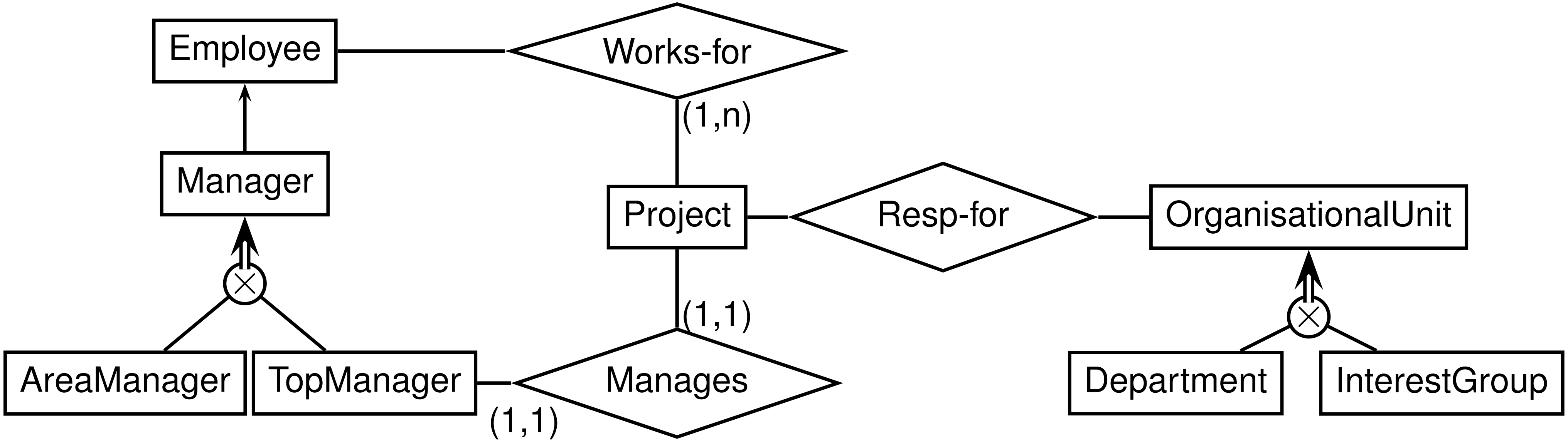
$$
\begin{array}{lcl}
\forall x. E(x) & \rightarrow & E_1(x) \vee \ldots \vee E_n \\
\forall x. E_1(x) & \rightarrow & E(x) \wedge \neg E_2(x) \wedge \ldots \wedge \neg E_n(x) \\
\forall x. E_2(x) & \rightarrow & E(x) \wedge \neg E_3(x) \wedge \ldots \wedge \neg E_n(x) \\
\forall x. E_{n-1}(x) & \rightarrow & E(x) \wedge \neg E_n(x) \\
\forall x. E_n(x) & \rightarrow & E(x)
\end{array}
$$

FOL Translation: An Example



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

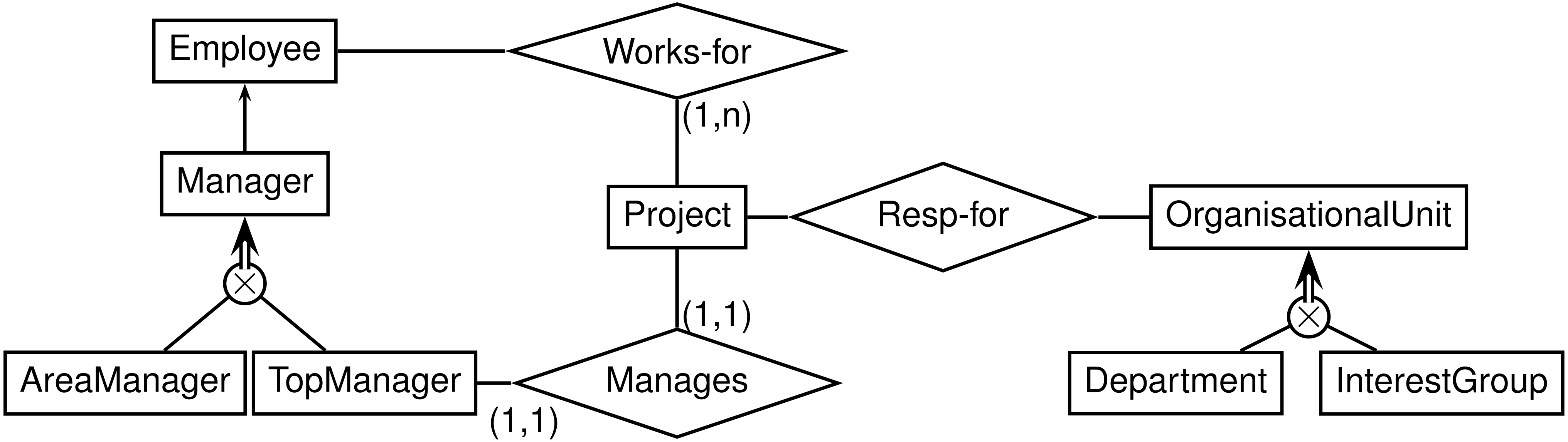
Additional (integrity) constraints



- 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

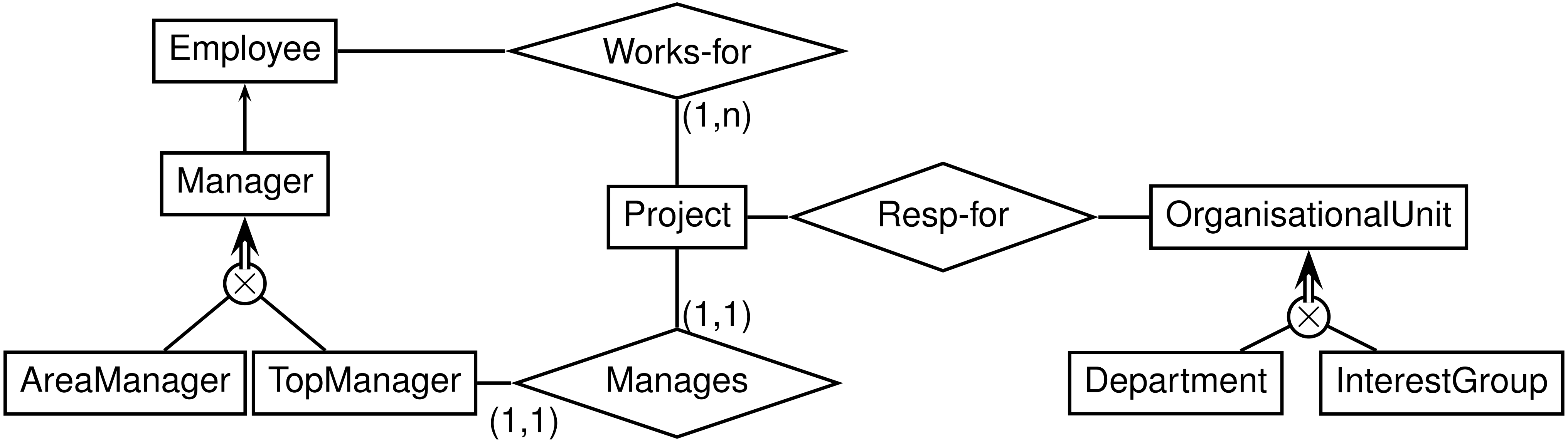


- Managers do not work for a project (she/he just manages it).

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

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

Additional (integrity) constraints



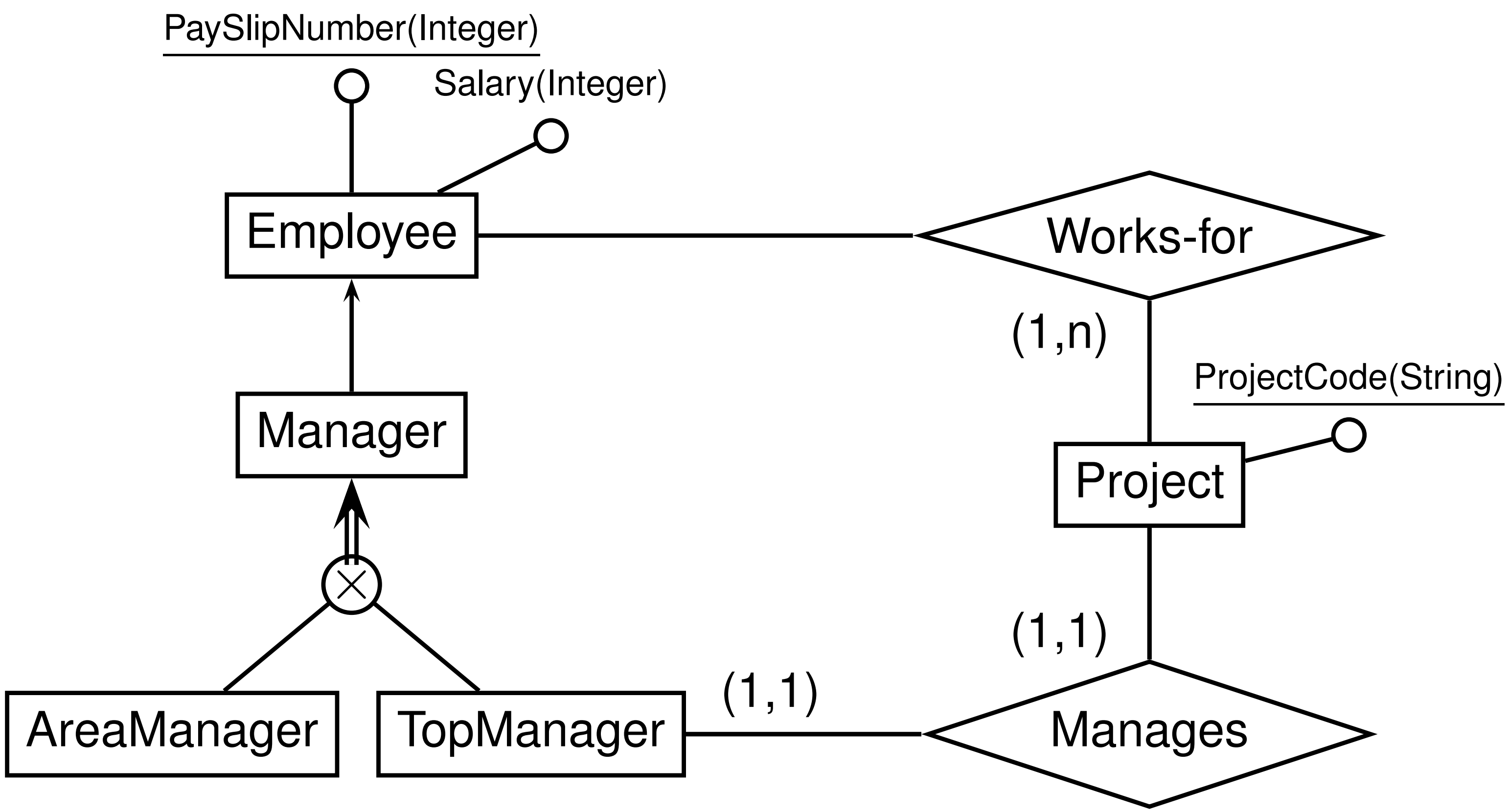
- Managers do not work for a project (she/he just manages it).

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

- 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.



$$\forall x.\ \mathtt{Project}(x) \rightarrow \exists^{=1} y.\ \mathtt{ProjectCode}(x, y) \land \mathtt{String}(y)$$

$$\forall y.\ \exists x.\ \mathtt{ProjectCode}(x, y) \rightarrow \exists^{=1} x.\ \mathtt{ProjectCode}(x, y) \land \mathtt{Project}(x)$$

Key constraints and relational schema

- A key is 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.

Standard mappings to the relational schema

Explain the following:

- No abstract instance is found ever in a database.
- Entities are mapped through their attributes.
- If there is a mandatory one-to-many binary relationship, the key of the dependant entity is added to the attributes of the main entity.