Weak Entities Expressed Without Weakness

Maarten Fokkinga

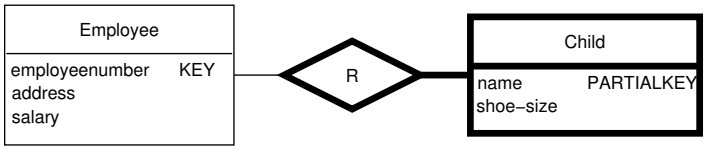
Version of March 29, 2006, 10:33

Weak entities can be described by Entity-Relationship diagrams without using the notion of weak entities.

Example. Consider the following standard example. A company wants to know for each of its employees the children. We assume that no two children in the same family have equal first names. So, to record a child of an employee, the child's first name alone suffices.

We shall show how these aspect of the real world can be expressed in terms of Entity-Relationship diagrams, with weak entities and without. The demonstration makes clear how to eliminate weak entities from any ER-diagram (while preserving the semantics).

With weakness. Consider the following ER-diagram featuring *weak* entities:

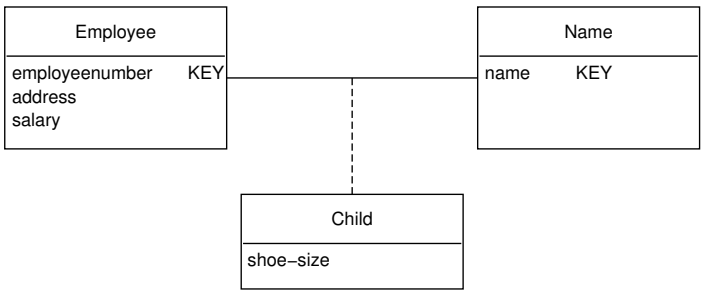


For this diagram one says that *Child* is a *weak* entity type (denoted by the fat lines), with *name* as *partial key*, and relation R is the *identifying* relationship of *Child* (denoted by the fat lines). The semantics of the fat lines is this:

- Each *Child* entity is related through R to exactly one *Employee* entity.
- Attribute *name* of *Child* together with *employeenumber* of *Employee* identify children.

One also says that *Child* is *existence dependent* on *Employee*: a child cannot exist without an employee, the employee to which it is related through the identifying relation.

Without weakness. Consider the following diagram (in the UML notation, with $0..*$ as default multiplicity):



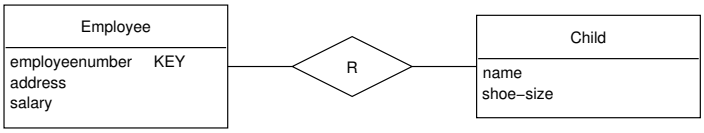
Since *Child* is an associative entity type (a relationship considered as an entity type in itself), a key for *Child* is the tuple consisting of the keys of the participating entities: (*employeenumber*, *name*). So, *name* is a partial key of *Child*. It also follows that for each *Child* there is exactly one *Employee* with the same *employeenumber* as that of the child's key. (Similarly, for each *Child* there is exactly one *Name* with the same *name* as that of the child's key.)

Entity type *Name* is an auxiliary entity type; only used to describe the existence of different names. Notice that different children can have the same name.

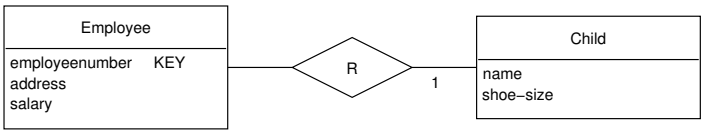
I consider the first ER-diagram as a notational abbreviation of the second ER-diagram. Every ER-diagram featuring *weakness* can similarly be translated to an equivalent ER-diagram without *weakness*.

Appendix

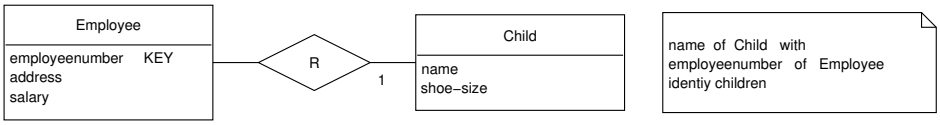
The above ER-diagrams satisfactorily describe the aspects of the real worlds of the given case. Other attempts to do so may easily lead to failures. For example, consider this diagram:



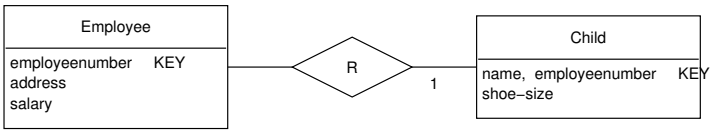
The problem is that, according to this diagram, an employee can have several equally named children, and a child can be related to several employees. Adding the cardinality constraint that is implicit in the very first diagram doesn't help:



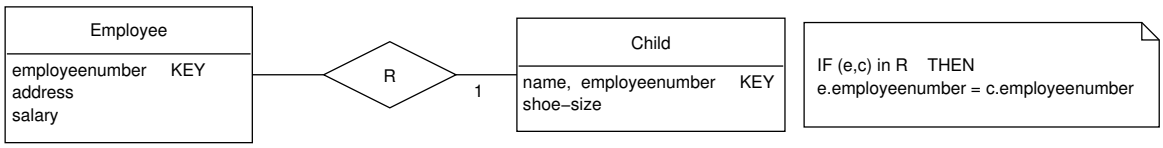
Although each child belongs to exactly one employee, it can be the case, according to the diagram, that equally named children belong to the same employee. (Note that the diagram does not indicate the key of *Child*). To solve this defect, one may add a note:



The accompanying note says that *name* of *Child* together with *employeenumber* of *Employee* identify children, so that *name* of *Child* is a partial key. However, a solution with notes is not considered successful. Here is another attempt:



The problem now is that the *employeenumber* of a child may differ from that of the employee it belongs to. Again, this can be repaired with a note:



Although this diagram precisely characterizes the real worlds of interest, the added note makes it not being a proper ER-diagram. Moreover, the double occurrence in the diagram proper of attribute *employeenumber* is not at all elegant.