Additional Requirements

So far we have only considered a few of the requirements for the university database — here are some more.
- Departments have a department number, a department name and a contact phone number
- Departments have a lecturer who is the head of department
- Lecturers may work in more than one department, in which case they have a percentage of their time associated with each department

Another entity set
These requirements introduce another straightforward entity set.

By now the procedure for identifying entity sets and their attributes and key should seem straightforward — at least for these very well specified small examples!

Headship – participation constraints
The Head relationship is between lecturers and departments. First we cover the participation constraints.
Every department has a head of department, and so there is total participation of the entity set Department in this relation, to be indicated by a thick black line.
However not every lecturer is a head, and so there is only partial participation of the entity set Lecturer, hence just a normal line.
Headship – key constraints

Next we cover the key constraints. Each department has exactly one lecturer as head and so here we have a key constraint — each department can only appear in one tuple of the relationship set. This is indicated by an arrow which goes from the entity set that is constrained to appear only once.

The Works-In relationship set

At first sight the Works-In relationship set is straightforward:

Note the thick lines indicating that every lecturer works in some department, and every department has some lecturers in it.

What’s the problem?

Suppose Associate Professor Smith works 60% of the time for CSSE and 40% of the time for Maths. The ER diagram gives us no mechanism for storing the percentages at the moment. We cannot extend the entity set Lecturer to contain a field percent because that field makes no sense when considering a lecturer in isolation.

Similarly we can not add this field to the entity set Department.
**Relationship Attributes**

The percentage time is not an attribute of a lecturer alone, nor of a department alone, but in fact is an attribute of a relationship between a lecturer and a department.

So in our example, the 60% amount is associated with the pair of entities (Smith, CSSE) and the 40% amount is associated with the pair of entities (Smith, Maths).

So percent is actually an attribute of the relationship set, not of any of its constituent entities.

(GF Royle 2006-8, N Spadaccini 2008)

**Diagramming relationship attributes**

An oval-shaped attribute is added to the relationship symbol to indicate that this attribute “belongs to” the relationship itself rather than to its components.

(GF Royle 2006-8, N Spadaccini 2008)

**Formally**

R & G sidestep this issue, but their formal definition of a relationship set needs to be modified.

If $E_1, E_2, \ldots, E_n$ are entity sets and $A_1, A_2, \ldots, A_m$ are sets (i.e. attribute sets) then a relationship set is a subset

$$ R \subseteq E_1 \times E_2 \times \cdots \times E_n \times A_1 \times \cdots A_m $$

We think of an element of the relationship set as consisting of $n$ entities (which of course possess their own attributes), together with $m$ values that describe aspects of the relationship.

(GF Royle 2006-8, N Spadaccini 2008)

**Example**

Thus for our example, if $L$ is the lecturer entity set, and $D$ the department entity set and $P = \{ p \mid 0 \leq p \leq 100 \}$ then

$$ \text{Works-In} \subseteq L \times D \times P $$

So now we can say sensible things like

$$ (\text{Smith, CSSE, 60}) \in \text{Works-In} $$

to capture the fact that Smith is 60% associated with CSSE.

Notice however that the ER model is NOT strong enough to specify various other types of “business logic” such as ensuring that the percentages for each lecturer add up to 100.
**Roles and Aggregation**

**Relationships between a set and itself**

In our definition of a relationship set, there was no requirement for the entity sets $E_1$, $E_2$, etc to be *different*.

Suppose that the university commences a formal system of *peer review of teaching* and specifies that:

- Each lecturer has another lecturer, called a *peer reviewer* who attends a few of their lectures and provides feedback.

Thus we have a binary relationship set in which *both* components come from the Lecturer entity set.

---

**Roles in an ER diagram**

![ER diagram](image)

---

**Aggregation**

So far we have covered relationship sets that involve a number of *entity sets* and that may (or may not) have *relationship attributes*.

However in a number of circumstances we want to model a relationship between an *entity set* and a *relationship set*.

This can be achieved by using a feature of the ER model called *aggregation* which provides a way of indicating that an entire relationship set is participating in another relationship set.

Essentially it allows us to view a relationship set as a “special sort” of entity set.
When is it used?

Suppose that we have additional requirements for research projects as follows.

- Graduate students have a student number, a name and a degree course.
- Each research project has one or more graduate students working on it.
- When graduate students work on a project they have one supervisor for their work on that project; they can work on more than one project in which case they may have different supervisors for each one.

But how to add the supervision requirement?

First approach

This introduces a ternary relationship called Supervises that relates a graduate student, a project and a lecturer.

The elements of this relationship set would be triples containing one graduate student, one project and a lecturer. So a triple of the form 

\((g, p, l)\)

could be interpreted as meaning

Graduate \(g\) is supervised by lecturer \(l\) while working on project \(p\).

This does capture some of the requirements — but what about participation and key constraints?
Key constraints

The problem is that we cannot correctly model the key constraints — that each "graduate student working on a project" should have a unique supervisor.

The problem is that we really want a binary relationship that has \((g, p)\) as one of its components and \(l\) as the other.

This is exactly the concept of aggregation in that we treat the relationship set \(\text{Works-On}\) as an entity set in its own right.

Boxes

This now models the exact concept that we wanted:

- Every graduate working on a project has exactly one supervisor for that piece of work.
- If a graduate works on more than one project, then they can have a potentially different supervisor for each project.

The dashed box indicates that for the purposes of the \(\text{Supervises}\) relationship set, the contents of the dashed box are to be taken as a single entity set.

A balancing act

Even with these additions, there are many aspects of a normal business that an ER diagram cannot capture.

The reason that relational databases have been so very successful is because:

- The ER model is strong enough to model a useful proportion of the activity of a business, yet.
- The ER model is simple enough to be implemented and mathematically analysed.

More expressive models are hard to implement and optimize, while less expressive models omit important functionality.