Relationship Types

A Relationship Type defines a relationship set among entities of certain entity types.

A relationship type is illustrated in an ERD using a diamond symbol. Consider the following diagram that shows two entity types (Department and Instructor) and the concept (a relationship) that instructors work in departments. This relationship involves two entity types and is referred to as a binary relationship.

Below is a ternary relationship (a relationship type involving three entity types).

Relationship types are useful for capturing/expressing certain business rules.

The next diagram shows two entity sets and one relationship set, and following diagram shows an ERD. Each set comprises a number of instances. In the ERD we define a relationship type for "register" and two entity types for "student" and "course".

April is taking two courses
Peter is taking zero courses
Jim is taking one course

Student Entity Set
with 3 student entities
Register
Relationship
Set with 3 relationships
Course Entity Set
with 2 course entities
Exercise:

Consider a typical company that you are familiar with. What entity types are there? How does the company organize itself? Do employees work in particular departments? Is there one employee designated the manager? If so then you would have two relationships between the same entity types.

**Relationship Sets**

An Relationship Set is a collection of relationships all belonging to one relationship type.

In the set are instances of relationships. For instance, if a relationship type is registration then each enrollment of a student in a course is an instance of registration and appears in the relationship set.

\[
\begin{align*}
\text{April is taking two courses} \\
\text{Peter is taking zero courses} \\
\text{Jim is taking one course}
\end{align*}
\]

\[
\begin{tikzpicture}
  \node[shape=circle,draw] (A) at (0,0) {April};
  \node[shape=circle,draw] (P) at (2,0) {Peter};
  \node[shape=circle,draw] (J) at (4,0) {Jim};
  \node[shape=circle,draw] (C1) at (2,-2) {Intro to CS};
  \node[shape=circle,draw] (C2) at (2,-4) {Calculus};
  \node[shape=circle,draw] (S1) at (-2,0) {Student Entity Set with 3 student entities};
  \node[shape=circle,draw] (R) at (0,-2) {Register Relationship Set with 3 relationships};
  \node[shape=circle,draw] (C) at (0,-4) {Course Entity Set with 2 course entities};

  \path[->] (A) edge (C1);
  \path[->] (A) edge (C2);
  \path[->] (P) edge (C1);
  \path[->] (P) edge (C2);
  \path[->] (J) edge (C1);
  \path[->] (J) edge (C2);
\end{tikzpicture}
\]

**Relationships**

A Relationship is one instance in a Relationship Set.

Consider a relationship type, registration, where each instance is the enrollment of a student in a course.

A relationship between the Student Type and Course Type occurs when a student is registered for a course. In the following diagram we have three relationships (three relationship instances) since April is taking two courses and Jim is taking one course.
Cardinality

Cardinality is a constraint on a relationship specifying the number of entity instances that a specific entity may be related to via the relationship. Consider the relationship "works in".

When we ask How many employees can work in a single department? or How many departments can an employee work in? we are asking questions regarding the cardinality of the relationship.

The three classifications are: one-to-one, one-to-many, and many-to-many.

Below, the ERD shows a relationship between invoice lines and products.

The "n" represents an "arbitrary number of instances", and the "1" represents "at most one instance". We interpret the cardinality specifications with the following business rule statements:

- The "n" indicates that the same Product entity can be specified on "any number of" Invoice Lines.
- The "1" indicates that an Invoice Line entity specifies "at most one" Product entity.
Exercise:

Consider the University and entity types: Class, Course, Department, Student.

What relationship types exist and what cardinalities apply to the various relationships and entities?

**One-to-One Relationships**

One-to-one relationships have 1 specified for both cardinalities, and do not seem to arise very often. To illustrate a one-to-one, we require very specific business rules.

Suppose we have People and Vehicles. Assume that we are only concerned with the current driver of a vehicle, and that we are only concerned with the current vehicle that a driver is operating. Then, we have a one-to-one relationship between Vehicle and Person (note the role shown for Person in this relationship):

![One-to-One Relationship Diagram]

**One-to-Many Relationships**

This type of relationship has 1 and n specified for cardinalities, and is very common in database designs. Suppose we have customers and orders and the business rules:

- an order is related to one customer, and
- a customer can have any number (zero or more) of orders.

We say there is a one-to-many relationship between customer and order, and we draw this as:

![One-to-Many Relationship Diagram]

**Many-to-Many Relationships**

Many-to-many relationships have "many" specified for both cardinalities, and are also very common. However, should you examine a data model in some business, there is a good chance you will not see any many-to-many relationships on the diagram. In those cases, the data modeler has resolved the many-to-many relationships into two one-to-many relationships. Suppose we are interested in courses and students and the fact that students register for courses: Any student may take several courses, A course may be
taken by several students. This situation is represented with a many-to-many relationship between Course and Student:

![Diagram of a many-to-many relationship between Student and Course]

Exercise:

When a student registers for courses, How many courses does he/she register for? How many students register for a single course? Is there a many-to-many relationship?

How many courses does an instructor teach? Can more than one instructor be assigned to the same course? Is there a many-to-many relationship?

Can you replace a many-to-many to become two one-to-many's?

**Degree**

The degree of a relationship is the number of entities participating in the relationship. When we speak of a student registered for a course, we are discussing a relationship, register, where two entity sets (Student and Course) are involved; the relationship is of degree 2 because each instance of register will always involve one student entity and one course entity.

![Diagram of a binary relationship between Student and Course]

Relationships of degree 2 are called binary relationships; relationships of degree 3 are called ternary relationships. In general we speak of n-ary relationships where n entities participate in a relationship. Most OLTP systems are designed using binary relationships, and these are mostly what this set of notes discuss.

Below is a ternary relationship involving stores, products and days. This model is useful if we need to represent the quantity sold and price that quantity was sold at, for a given day, a given store, and a given product. Each instance of the relationship involves 3 entities and so the relationship is a ternary relationship.
Identifying Relationships

An Identifying Relationship is a relationship between a strong and a weak entity type, where the key of the strong entity type is required to uniquely identify instances of the weak entity type. The weak entity type will have a partial key (discriminator) attribute that, in conjunction with the key of the strong entity type, uniquely identifies weak entity instances.

Consider a university environment where many different courses are offered. Many universities assign numbers to their courses and so, for example, we could have courses such as 2914, 3902, 3913, etc. It is typical that a Course may be offered zero, one, or more times, in a single term. Hence, there may be several Sections of the same Course each term. Each Section needs to be identified distinctly from any other. The typical solution is use a section number such as 001, 002, 003, etc. The section number is appended to the course identifier to yield identifiers for the course offerings, such as 2914-001, 2914-002, 3902-050, etc. In this example, Section is modeled as a Weak Entity Type and its relationship to Course is an identifying relationship.

Note in the following diagram that the identifying relationship is drawn with a double line. Since Section is a weak entity it is shown with a double lined box, and since a Section cannot exist on its own but in relationship to a Course we shown a mandatory relationship too. Note that Section has a partial key sectionNo, Course has a key courseNo.
Exercise:

Suppose Course is a weak entity type too. Assume that departments are entities that offer courses. Suppose course numbers are only unique within departments and that departments have a unique attribute department number. That is, department number is a key for department. The combination department number and course number uniquely identify a course. Modify the ERD above to account for these new business rules.

**Non-Identifying Relationships**

A Non-Identifying Relationship is a relationship between strong entity types. Each entity type has a key specified; each entity has an attribute that uniquely identifies it and distinguishes it from any other instance in the corresponding entity set or extension.

Suppose each University department has one office, but that the office could be the location for more than one department.

**Recursive Relationships**

Recursion in a data model is an especially difficult topic. We say we have a recursive relationship if the same entity type appears more than once in a relationship. Recursive relationships seem to always occur somewhere in a data model of decent size. A typical business example arises when we have a business rule such as "an employee supervises other employees":

In the above model, why is it necessary that the entity participation is optional?

Consider other situations such as:

- a team plays a game against another team
- a person is a child of a person
- an organizational unit (e.g. department, division, branch, ...) comprises other units
- a course is a prerequisite for another course

### Mandatory Participation

Consider an educational system where students enroll in courses and consider questions like:

- Must a course have students enrolled in it?
- Must a student enroll in a course?

These questions focus on the participation requirement of an entity in a relationship. If all entities in an entity set must participate in a relationship, we say its participation is mandatory and we use a double line for the relationship line adjacent to the entity set for that relationship.

Suppose we have the following business rules: A book may have zero or more authors (book's participation is optional), and an author must be related to at least one book (an author must participate):

We say in this case that Author has an *existence dependency* on Book... an author cannot exist in this model unless we can identify a book related to the author.
**Optional Participation**

Consider an educational system where students enroll in courses and consider questions like:

- Must a course have students enrolled in it?
- Must a student enroll in a course?

These questions focus on the participation requirement of an entity in a relationship. If a business rule does not require an entity (in the entity set) to participate in a relationship, then we say the participation of the entity type in the relationship is optional. We use a simple single line (as opposed to a double line) for the relationship line adjacent to the entity type.

Suppose we have the following business rules: A book may have zero or more authors (book's participation is optional), and an author must be related to at least one book (an author must participate):

![Diagram](image-url)