Relationships and Cardinality

by Sophia

WHAT'S COVERED

In this lesson, you will explore the classifications of cardinality between tables, including 1:1 (one-to-one), 1:N or 1:M (one-to-many), and M:N (many-to-many), in three parts. Specifically, this lesson will cover:

1. Introduction

In database design, **cardinality** is the description of the numerical relationships between two tables. It describes the number of instances of one entity (or table) that can be associated with a single instance of another entity (or table) through a relationship. The three types of relationships (or cardinality) that occur are:

- One-to-one (1:1)
- One-to-many (1:N)
- Many-to-many (M:N)

TERM TO KNOW

Cardinality
The description of the numerical relationships between two tables.

2. Relationship Types

The **one-to-one (1:1)** relationship type is quite rare in database design, as this defines that a single row in one table is related to just one row in another table and vice versa.

EXAMPLE For example, you might have an Employee table and a Spouse table. Each person in the Spouse table is associated with only one record in the Employee table and vice versa. This type of relationship is rarely used in database design.

The **one-to-many (1:N)** relationship type means that one row in a database table relates to many rows in a second table. This type is the most common relational model relationship. This relationship type is the norm for most entity relationships. The "one" side of the relationship typically involves the primary key in that table, and the "many" side is the foreign key.

EXAMPLE For example, a department can consist of multiple employees, but an employee can belong to only one department. The "one" side is the Department table and the "many" side is the Employee table.

The **many-to-many (M:N)** relationship type means that many rows in one table are related to many rows in a second table. This relationship type isn't one that can be implemented in a relational model.

BIG IDEA

If you do have a many-to-many relationship between two entities, it should generally be changed into two one-to-many relationships with a linking table (also called a bridge, associative, or composite table) in between them. The linking table between the two entities will have multiple occurrences of the foreign key values. The same linking table should at least contain the primary keys of the original tables, but it may have its own primary key as well.

EXAMPLE For example, a company might have an employee who works on multiple projects, and each project might have multiple employees working on it. You would need to create a linking table between the Employee table and the Project table, perhaps one called Assignment, to contain the primary key of the Employee table and the primary key of the project table. If this linking table is not referenced anywhere else, you may simply choose to use the combination of the foreign keys as the primary key. However, if the table is referenced from other tables, creating a new primary key is probably the best choice so that there is only one value to track in those related tables, rather than a combination of the other two (or more) keys.

TERMS TO KNOW

One-to-One (1:1)
A relationship in which a single row in one table is related to just one row in another table and vice versa.

One-to-Many (1:N)
A relationship in which one row in a database table relates to many rows in a second table.

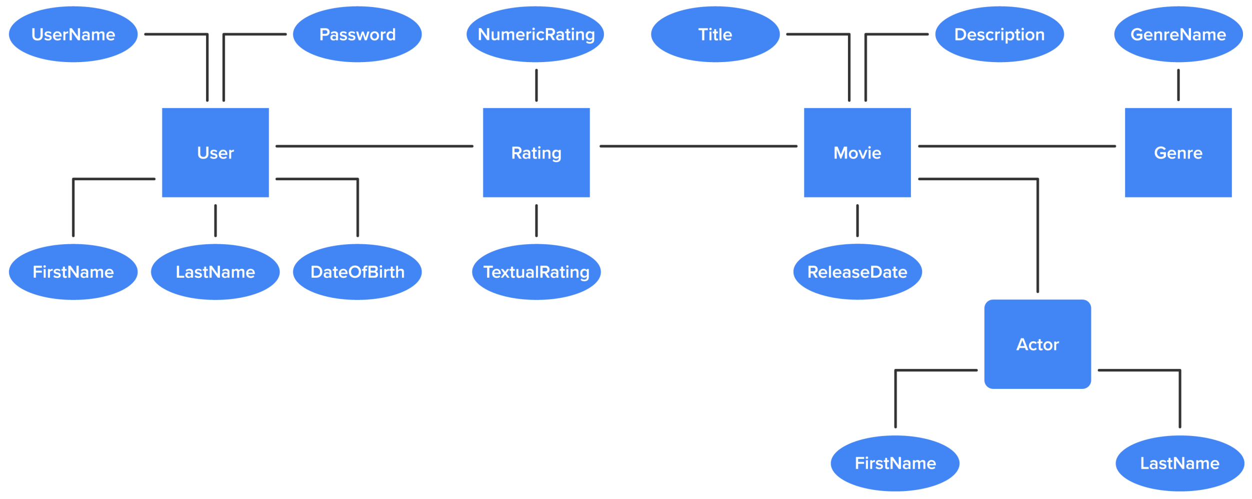
Many-to-Many (M:N)
A relationship in which many rows in one table are related to many rows in a second table. This relationship type isn't one that can be implemented in a relational model.

3. Movie Ratings Example

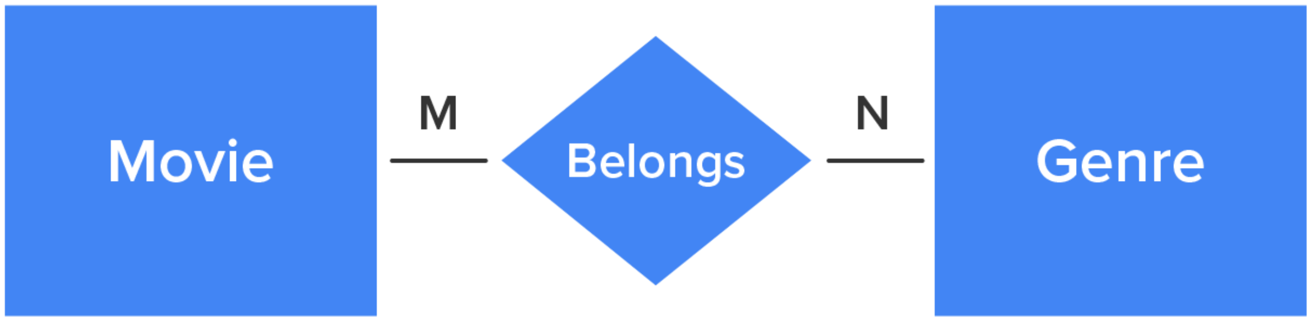
Recall that using Chen notation, you signify a relationship between two entities by drawing a diamond, with an action verb describing the relationship written inside. Alongside each entity on the relationship line, you define the cardinality to show which side is the one and which side is the many. Let's go back to the movie ratings ERD from the prior lesson.

TRY IT

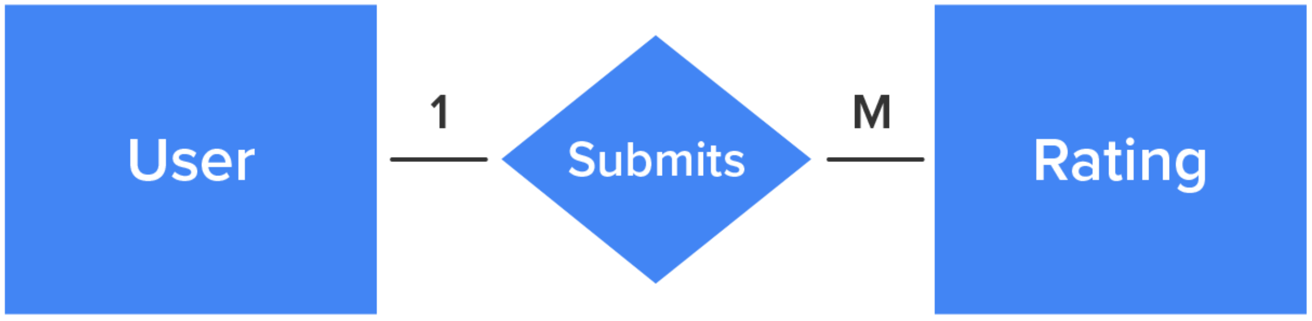
See if you can figure out the correct cardinality for each of the relationships defined in the movie ratings ERD shown below.



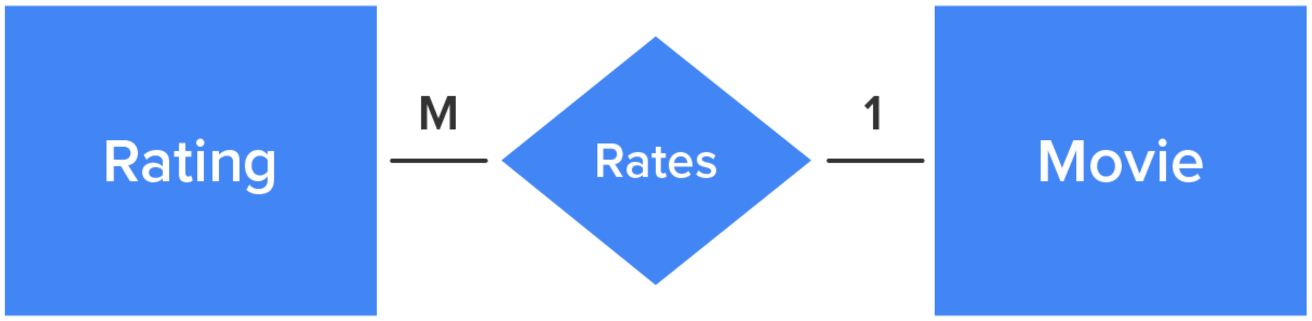
For example, a movie can belong to multiple genres, and a genre can have multiple movies (M:N).



A user can submit multiple ratings, and a rating can only belong to a single user (1:M).



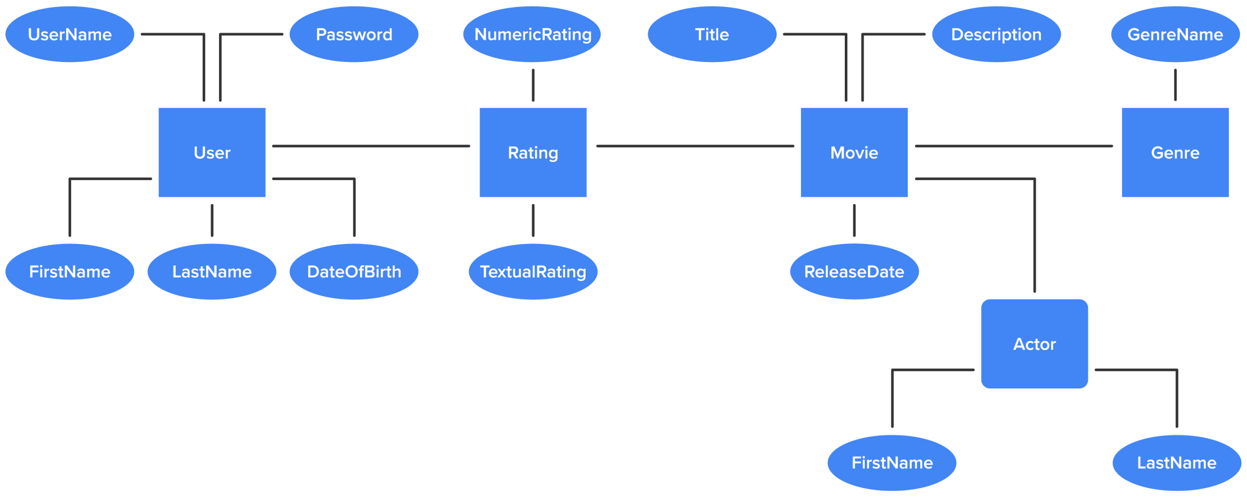
A rating is only for a single movie, and a movie can be rated multiple times (1:M).



Likewise, a movie can have many actors, and an actor can act in many movies (M:N).

TRY IT

Check your work from the earlier Try It activity against the relationship cardinalities defined in the following diagram, and make any changes needed to your own diagram.



SUMMARY

In this lesson **introduction**, you learned that cardinality is a crucial aspect of an ERD. You reviewed the three **types of relationship** cardinality in a relational database and read about an example of each one: one-to-one, one-to-many, and many-to-many. One-to-one is uncommon; one-to-many is the norm for most relationships. Many-to-many is not allowed in a well-constructed relational database; it must be converted to two one-to-many relationships using a linking table. You then looked at the **movie ratings database example** and identified the cardinality of each relationship between the tables.

Source: THIS TUTORIAL WAS AUTHORED BY DR. VINCENT TRAN, PHD (2020) AND Faithe Wempen (2024) FOR SOPHIA LEARNING. PLEASE SEE OUR TERMS OF USE.

TERMS TO KNOW

Cardinality

The description of the numerical relationships between two tables.

Many-to-Many (M:N)

A relationship in which many rows in one table are related to many rows in a second table. This relationship type isn't one that can be implemented in a relational model.

One-to-Many (1:N)

A relationship in which one row in a database table relates to many rows in a second table.

One-to-One (1:1)

The one-to-one (1:1) relationship type is quite rare in database design, as this defines that a single row in one table is related to just one row in another table and vice versa.