Relationships and Cardinality

by Sophia

WHAT'S COVERED

This tutorial explores the classifications of cardinality between tables including 1:1 (one-to-one), 1:M (one-to-many), and M:N (many-to-many), in three parts:

1. Introduction
2. Relationship Types
3. Movie Ratings Example

1. Introduction

You may have noticed the term cardinality. **Cardinality**, in database design, represents the relationship between tables. You have already seen the different types of relationships, but now it's time to define them. The three types of relationships (or cardinality) that occur are:

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

TERM TO KNOW

Cardinality

In database design, cardinality represents the relationship between 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 An example of this is if a company has an employee associated with one spouse, and one spouse is associated with one employee. This may be needed for employee benefits, perhaps, but these situations are quite rare.

The **one-to-many (1:M)** 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.

EXAMPLE An example of this type of relationship is if a company has a department that consists of multiple employees, but at any given time, an employee can only belong to one department.

The **many-to-many (M:N)** relational 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 such a relationship between two entities, it should generally be changed into two one-to-many relationships with a bridge/associative/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 An example of this type of relationship is if a company has an employee who works on multiple projects, and a project has multiple employees involved. Although you have these two tables, you would have to create a linking table between entities, perhaps called "assignment," to contain the primary key of the employee table and the primary key of the project table. If this bridge entity 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)

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.

One-to-many (1:M)

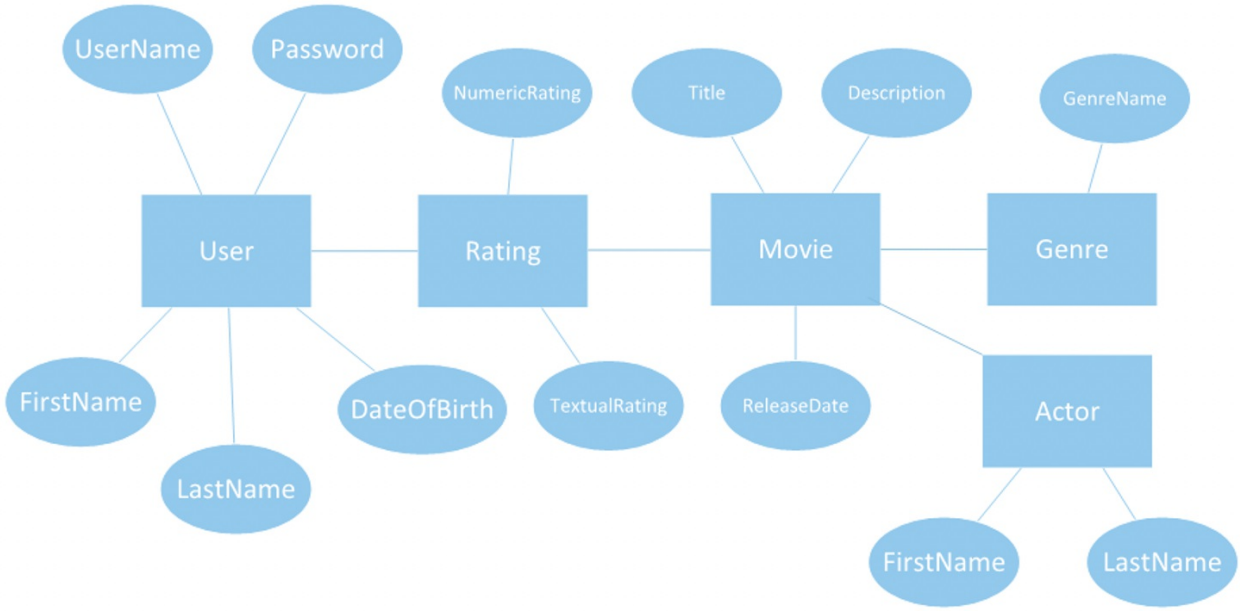
The one-to-many (1:M) relationship type means that one row in a database table relates to many rows in a second table.

Many-to-many (M:N)

The many-to-many (M:N) relational 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.

3. Movie Ratings Example

Using Chen's 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, for example, which side is the one and which side is the many. Let's go back to your movie ratings ERD from the prior tutorial:



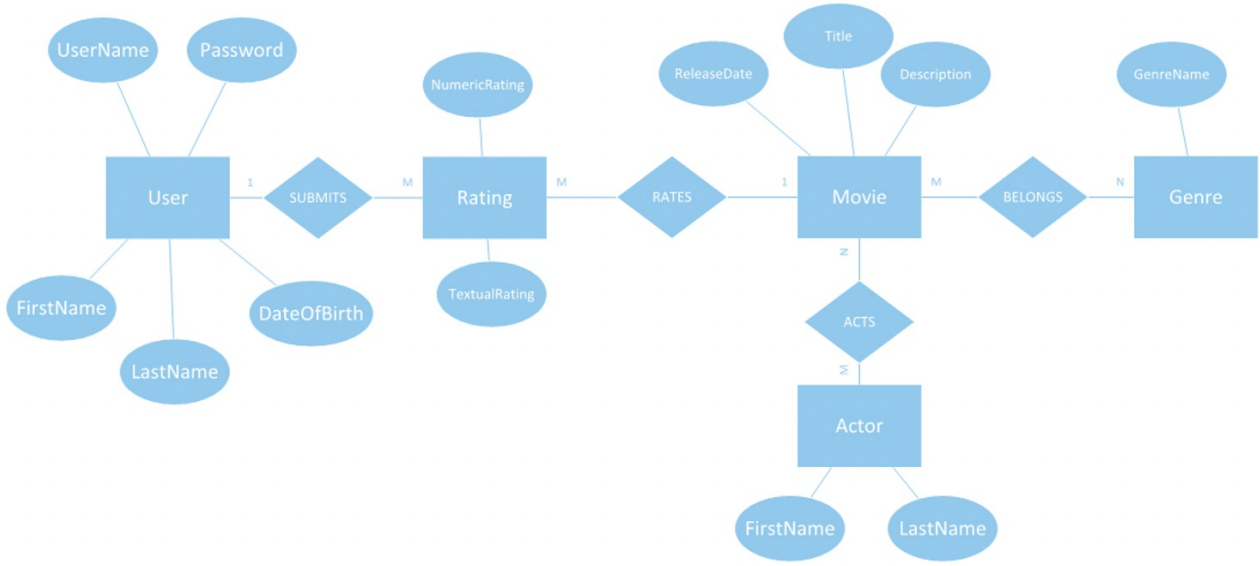
You have identified specific relationships between entities. For example, a movie can belong to multiple genres, and a genre can have multiple movies (M:N).

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).

Now you have the following ERD, which consists of the entities, attributes, and relationships.



SUMMARY

In this tutorial in the **introduction**, you learned that the term cardinality is the relationship that exists between tables. You then reviewed the **three relationship types**, namely 1:1 (one-to-one), 1:M (one-to-many), and M:N (many-to-many). Finally, you learned that relationships are drawn between entities with a diamond shape, and the cardinality is listed beside each entity. You saw this relationship type creation using the **movie ratings example**.

Source: Authored by Vincent Tran

TERMS TO KNOW

Cardinality

In database design, cardinality represents the relationship between tables.

Many-to-many (M:N)

The many-to-many (M:N) relational 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.

One-to-many (1:M)

The one-to-many (1:M) relationship type means that 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.