

# An overview of relational databases

By Ronny Shani

---

Databases make the world go around.

From governments to businesses, everything around us can be traced back to a database maintained by some entity or another. Our identities and assets, the digital items we create and consume, the physical objects that surround us, sometimes even the natural ones.

Databases store large amounts of data in a structured way. This structure makes organizing, accessing, and analyzing data points easy. The database model determines the structure and the way data is organized and handled. The most popular model is a table-based relational database.

Like spreadsheets, each table in a relational database consists of rows (also called records) and columns (also called fields). Columns represent data categories, while rows represent individual items.

Here's a simple example of a **Customers** table:

Customer ID	Name	Address	Phone number
1	Lovely Brown	Some St. 1	1234567890
2	Will Hope	Other St. 1	0987654321

The table is categorized into the **Customer ID**, **Name**, **Address**, and **Phone number** columns, and the content of each row is the person's details.

## How do relational databases work?

Relational databases have a superpower: they allow you to link different pieces of data from various tables. When a table references the records stored in another table, it creates connections. These form the relationships that give this model its name.

Let's say you have another table, **Orders**:

Order ID	Status
1	Pending
2	Shipped

A relational database lets you link the customer and the order, like so:

Order ID	Customer ID	Status
1	1	Pending
2	2	Shipped

Connecting discrete bits is what makes data valuable. It allows you to transform the pieces into meaningful, coherent information, which then becomes knowledge that informs your decision. This is the literal meaning of the term “data-driven”.


Whether you manage an inventory, a workforce, an event, or a project, a relational database helps you become more informed, efficient, and ultimately productive.

## The relationship types

The relationship types control how the data (and the tables) relate to one another. Let’s examine the three most common.


### One-to-One

Each row in Table 1 corresponds only to one row in Table 2, and vice versa.

 A temporal analogy for a scenario where One-to-One can be relevant is a booking system for a hospitality business: a restaurant owner can reserve a table for one party at a time. The same applies to hotel rooms, bike rentals, etc.

### When to use it?

A One-to-One model is the least common type of relationship, as it is very restrictive. From a practical perspective, it’s rare to have two entities that are tightly and exclusively coupled. Most business use cases that seem applicable at first sight quickly become irrelevant: Users can have more than one email, employees can have more than one company-issued computer, customers can have more than one shipping address, products can have more than one vendor, etc.

 You may think that storing everything in the same table would solve the problem, but you’ll quickly end up at square one: wrangling heaps of data points in an inefficient format.


That's why One-to-One is usually created when an explicit restriction is what you need.

For example, when your business must comply with data protection regulations. Say you have an **Employees** table listing contact information, position, department, direct manager, and pay grade. Multiple teams—IT, HR, office management, and finance—need to interact with it, but not all should have access to each employee's entire set of data points.

That's where the One-to-One model shines, allowing you to split certain records into separate tables and restrict access based on relevant criteria.

## One-to-Many

Each row in Table 1 can be associated with multiple rows in Table 2, but each row in Table 2 can only be associated with a single row in Table 1.

 A good way to think about the One-to-Many relationship as a family tree: a parent can have children (plural), but a child can only have one birth mother.

## When to use it?


The One-to-Many model is the most common relationship type, as it covers numerous use cases (including those listed as irrelevant to One-To-One above). Let's explore a few typical scenarios:

- Consider the relationships between your **Customers**, **Orders**, and **Products** tables: one customer can place many orders, each consisting of multiple products.
- The **Departments** table in your HR database can list multiple employees in each team, while the **Employees** table specifies that each employee belongs to a single department.
- If you're an event manager, your **Sponsors** and **Events** tables could reference each other in a One-to-Many relationship: each event has one exclusive sponsor, which, in turn, can provide backing to many events.

What happens if you want multiple sponsors to finance the same event? Keep reading to learn about the relationship type suitable for your purpose.

## Many-to-Many

Each row in Table 1 can be associated with multiple rows in Table 2 and vice versa.

 The Many-to-Many is similar to a conference, where any number of speakers can participate in multiple panels held in multiple venues.

## When to use it?

The Many-to-Many model is the most flexible relationship type and lends itself well to diverse use cases.

- If you're running a consulting business, you can link your **Clients** and **Services** tables: each client can select multiple services, and each service is offered to various clients.
- If you have a store, you can link your **Inventory** and **Customers** tables: many customers can buy multiple products available in stock.
- Your **Employees** and **Projects** tables reflect the fact that most employees work on many projects, and each project involves several coworkers.

This model fits numerous business workflows—from running a small agency to managing an international construction project, from HR to ERP, and more.

## Turn knowledge into action

Now that you have a better understanding of relational databases and relationship types, you can apply these principles to your needs. Pick the option that works best with your data and requirements and unlock the full potential of our platform.