A Relational Database Management System (RDBMS) is software that stores, manages, and retrieves data structured in tables with rows and columns. Based on the relational model, it uses Structured Query Language (SQL) to manipulate data and enforce integrity, enabling relationships between tables. Key examples include MySQL, Oracle, and SQL Server.

Key Features and Concepts of RDBMS:

- **Tables (Relations):** Data is organized into rows (records) and columns (attributes).
- **Relationships:** Tables can be linked together using primary and foreign keys to relate data across tables.
- **SQL (Structured Query Language):**

  The standard language used to create, read, update, and delete (CRUD) data

  .
- **Data Integrity:** RDBMS enforces strict rules (constraints) to ensure data accuracy, consistency, and reliability, such as `PRIMARY KEY` constraints.
- **ACID Compliance:** Ensures transactions are processed reliably (Atomicity, Consistency, Isolation, Durability).

Common RDBMS Examples:

- MySQL
- **Oracle Database**
- **Microsoft SQL Server**
- **PostgreSQL**
- **SQLite**

RDBMS vs. DBMS:
While a DBMS stores data, an RDBMS specifically uses the relational model, supports multiple users, ensures ACID compliance, and provides better security and normalization, making it superior for complex, structured data.

A relational database is a collection of information that organizes data in predefined relationships where data is stored in one or more tables (or "relations") of columns and rows, making it easy to see and understand how different data structures relate to each other. Relationships are a logical connection between different tables, established on the basis of interaction among these tables.

The relational database model

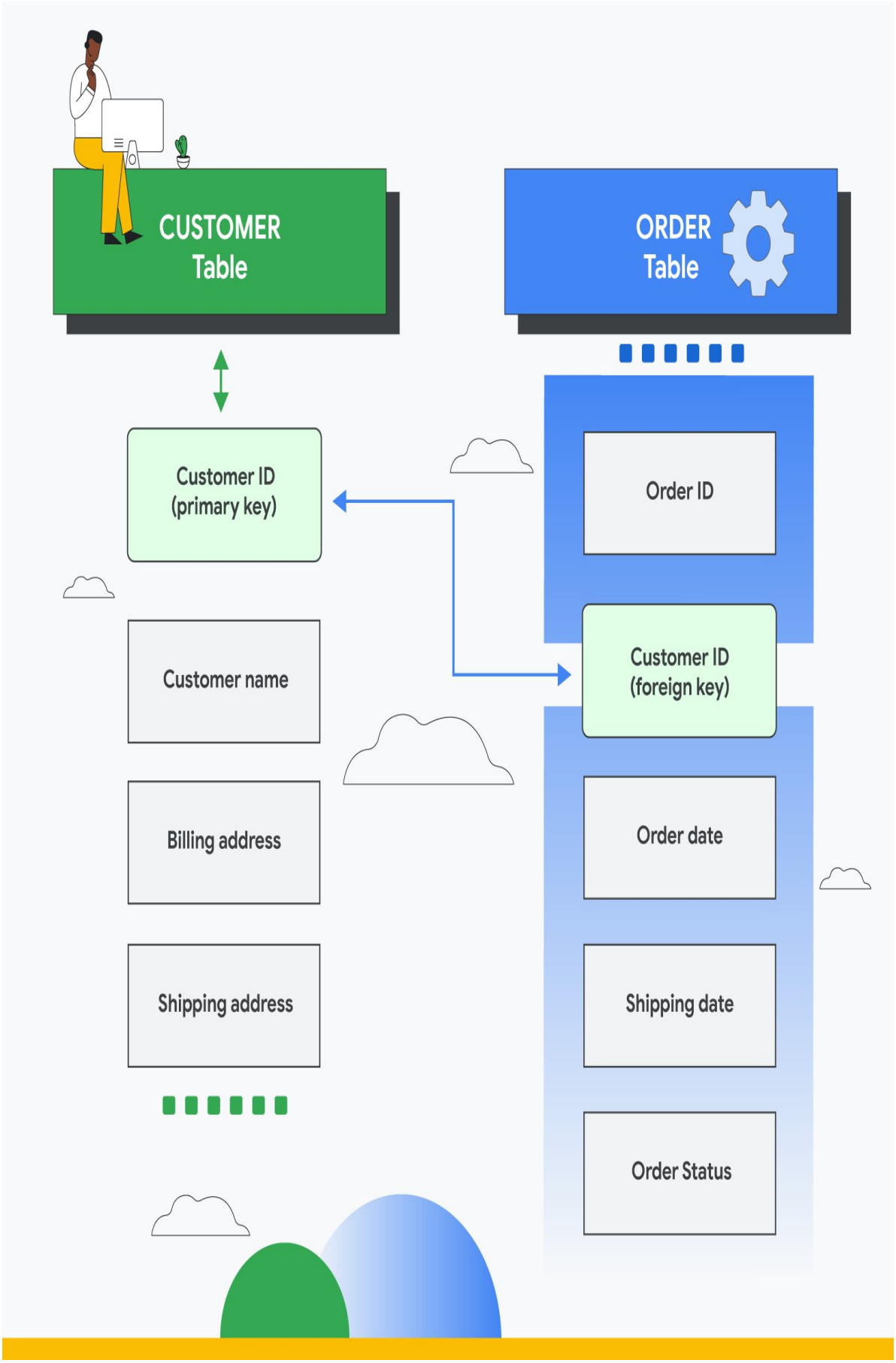
Developed by E.F. Codd from IBM in the 1970s, the relational database model allows any table to be related to another table using a common attribute. Instead of using hierarchical structures to organize data, Codd proposed a shift to using a data model where data is stored, accessed, and related in tables without reorganizing the tables that contain them.

Think of the relational database as a collection of spreadsheet files that help businesses organize, manage, and relate data. In the relational database model, each “spreadsheet” is a table that stores information, represented as columns (attributes) and rows (records or *tuples*).

Attributes (columns) specify a data type, and each record (or row) contains the value of that specific data type. All tables in a relational database have an attribute known as the **primary key**, which is a unique identifier of a row, and each row can be used to create a relationship between different tables using a **foreign key**—a reference to a primary key of another existing table.

Lets take a look at how the relational database model works in practice:

Say you have a **Customer** table and an **Order** table.


CUSTOMER
Table
ORDER
Table
Customer ID
(primary key)
Customer name
Billing address
Shipping address
Order ID
Customer ID
(foreign key)
Order date
Shipping date
Order Status

The **Customer** table contains data about the customer:

- Customer ID (primary key)
- Customer name
- Billing address
- Shipping address

In the **Customer** table, the customer ID is a primary key that uniquely identifies who the customer is in the relational database. No other customer would have the same Customer ID.

The **Order** table contains transactional information about an order:

- Order ID (primary key)
- Customer ID (foreign key)
- Order date
- Shipping date
- Order status

Here, the primary key to identify a specific order is the Order ID. You can connect a customer with an order by using a foreign key to link the customer ID from the **Customer** table.

The two tables are now related based on the shared customer ID, which means you can query both tables to create formal reports or use the data for other applications. For instance, a retail branch manager could generate a report about all customers who made a purchase on a specific date or figure out which customers had orders that had a delayed delivery date in the last month.

The above explanation is meant to be simple. But relational databases also excel at showing very complex relationships between data, allowing you to reference data in more tables as long as the data conforms to the predefined relational schema of your database.

As the data is organized as pre-defined relationships, you can query the data declaratively. A declarative query is a way to define what you want to extract from

the system without expressing how the system should compute the result. This is at the heart of a relational system as opposed to other systems.

Examples of relational databases

Now that you understand how relational databases work, you can begin to learn about the many relational database management systems that use the relational database model. A relational database management system (RDBMS) is a program used to create, update, and manage relational databases. Some of the most well-known RDBMSs include MySQL, PostgreSQL, MariaDB, Microsoft SQL Server, and Oracle Database.

Cloud-based relational databases like Cloud SQL, Cloud Spanner and AlloyDB have become increasingly popular as they offer managed services for database maintenance, patching, capacity management, provisioning and infrastructure support.

Ready to get started? Create a 90-day Cloud Spanner free trial instance with 10 GB of storage at no cost.

Benefits of relational databases

The main benefit of the relational database model is that it provides an intuitive way to represent data and allows easy access to related data points. As a result, relational databases are most commonly used by organizations that need to manage large amounts of structured data, from tracking inventory to processing transactional data to application logging.

There are many other advantages to using relational databases to manage and store your data, including:

Flexibility

It's easy to add, update, or delete tables, relationships, and make other changes to data whenever you need without changing the overall database structure or impacting existing applications.

ACID compliance

Relational databases support ACID (Atomicity, Consistency, Isolation, Durability) performance to ensure data validity regardless of errors, failures, or other potential mishaps.

Ease of use

It's easy to run complex queries using SQL, which enables even non-technical users to learn how to interact with the database.

Collaboration

Multiple people can operate and access data simultaneously. Built-in locking prevents simultaneous access to data when it’s being updated.

Built-in security

Role-based security ensures data access is limited to specific users.

Database normalization

Relational databases employ a design technique known as normalization that reduces data redundancy and improves data integrity.

Relational vs. non-relational databases

The main difference between relational and non-relational databases (NoSQL databases) is how data is stored and organized. Non-relational databases do not store data in a rule-based, tabular way. Instead, they store data as individual, unconnected files and can be used for complex, unstructured data types, such as documents or rich media files.

Unlike relational databases, NoSQL databases follow a flexible data model, making them ideal for storing data that changes frequently or for applications that handle diverse types of data.