

# What is a database? (video)

# How is data related? (video)

## Relational data example charts

Data gets collected and stored in databases from various sources for various reasons. For example, customer orders, student course enrollments, and user interaction and feedback to personalize content and improve services.

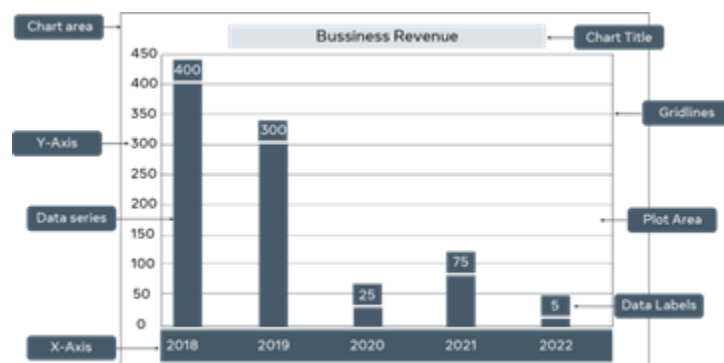
It's important to organize data, process it, and present it efficiently to make it more useful and meaningful to people. The way data is related and presented enables people to form a better understanding of existing data. This understanding can be aided by relevant charts that present data visually using combinations of text, symbols, and graphic elements to illustrate the relationship between data in a meaningful way.

Charts can convey a great deal of information and can capture people's attention in a way that helps them to make better decisions and take suitable actions. Here, you will learn about basic charts commonly used to relate data together and present it in a simple visual way.

## Bar chart

A bar chart is a graph that presents categorical data with rectangular bars, where the heights of the bars are proportional to the values that they represent.

For example, the owner of a bookshop in London has had many challenges during the COVID-19 lockdown and wants to know more about their business performance and progress each year from 2019. A bar chart could be very useful to show how sales revenue has changed over the past few years and how the pandemic impacted the business during lockdown.

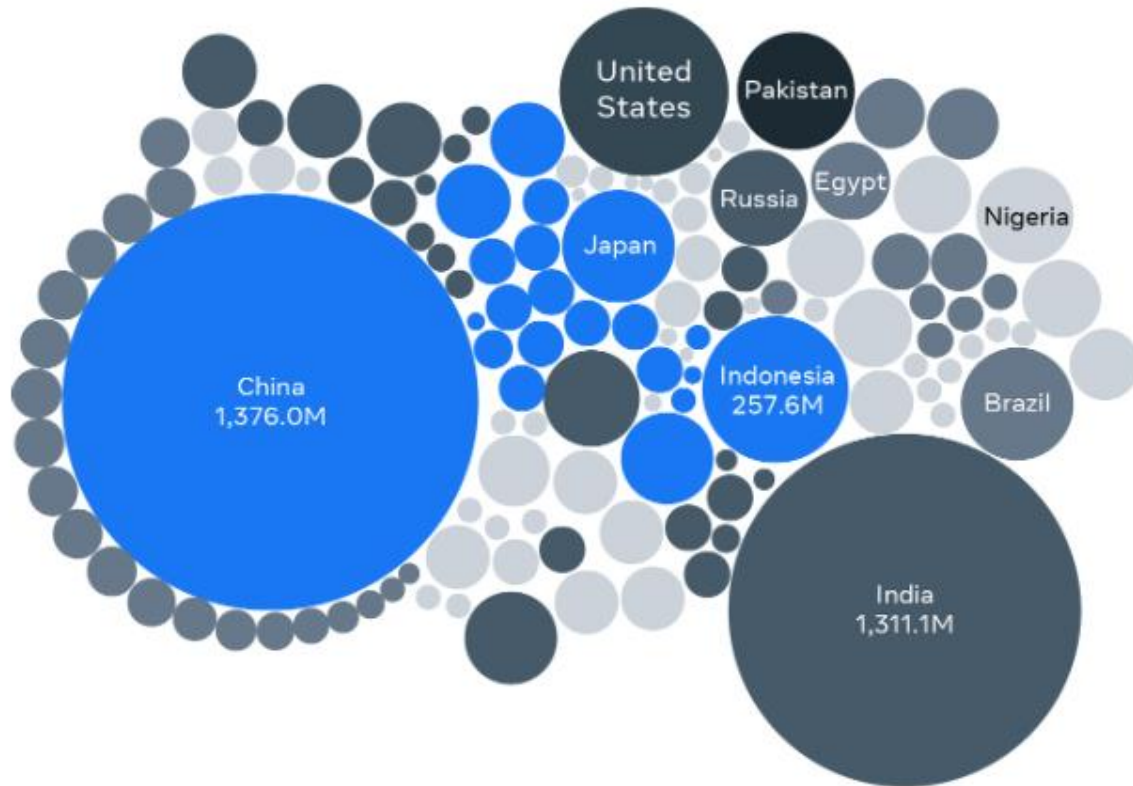


This chart uses bars to present the bookshop's sales data between 2019 and 2022. The x-axis presents the individual years, while the y-axis presents the sales value. The bars illustrate the sales achieved each year. The taller the bar, the greater the value of sales. In this case, the tallest bar is in 2018, which indicates that this was the most successful year for the business. The smallest bar is in 2022, which indicates that this was the worst year for sales.

## Bubble chart

A Bubble chart is another popular type of data chart. It shows how different values compare to each other in terms of bubble size. The smaller bubbles represent smaller values, and the larger bubbles represent larger values.

Let's examine the bubble chart below, which presents information about the 10 largest countries in the world in terms of population in 2015.



In this example, a country's population value determines the size of each country's bubble.

There are large bubbles for China (around 1.4 billion people) and India (about 1.3 billion people), as these countries have the largest populations. Then there are medium-sized bubbles for the USA (about 330 million people) and Indonesia (about 270 million people). Russia (about 145 million people) and Egypt (around 100 million people) have smaller-sized bubbles, as they have smaller comparable populations.

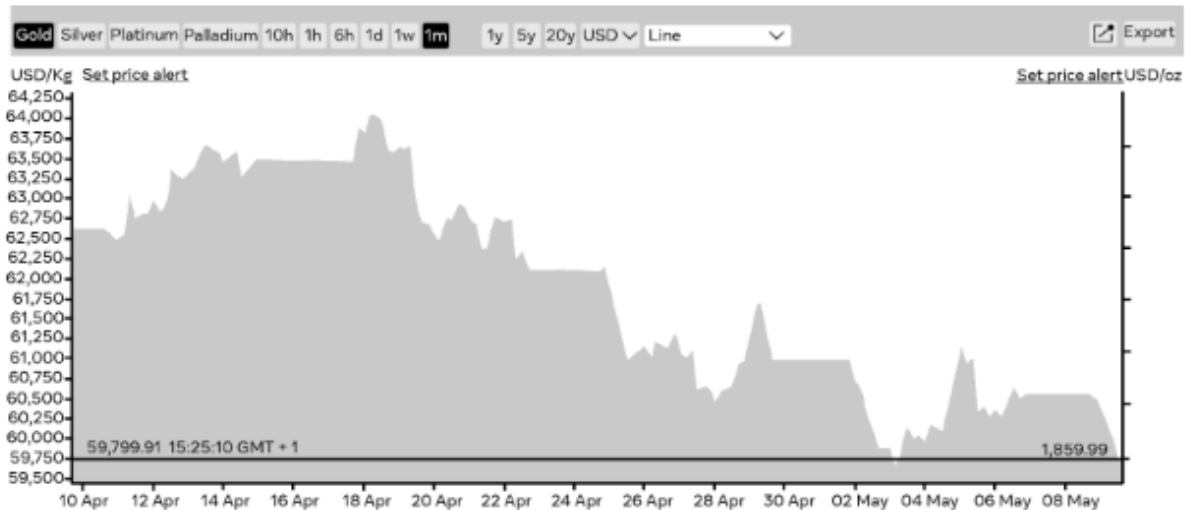
These bubbles give you a good idea of the difference between the countries regarding population sizes. The bubbles also help people to remember this kind of information, as human memory prefers graphical representation of data. After all, "a picture is worth a thousand words".

## Line chart

A line chart presents information as a series of data points called “markers” connected by straight line segments. Line charts are extremely popular and are widely used in most data analytics fields.

The chart below depicts a company's gold price over the past month. There is a line that starts with the 10th of April when gold stood at \$62,650 for 1kg. This line connects the dots that visualize the change in the gold price over time. The up-and-down movement of the line helps highlight positive and negative changes.

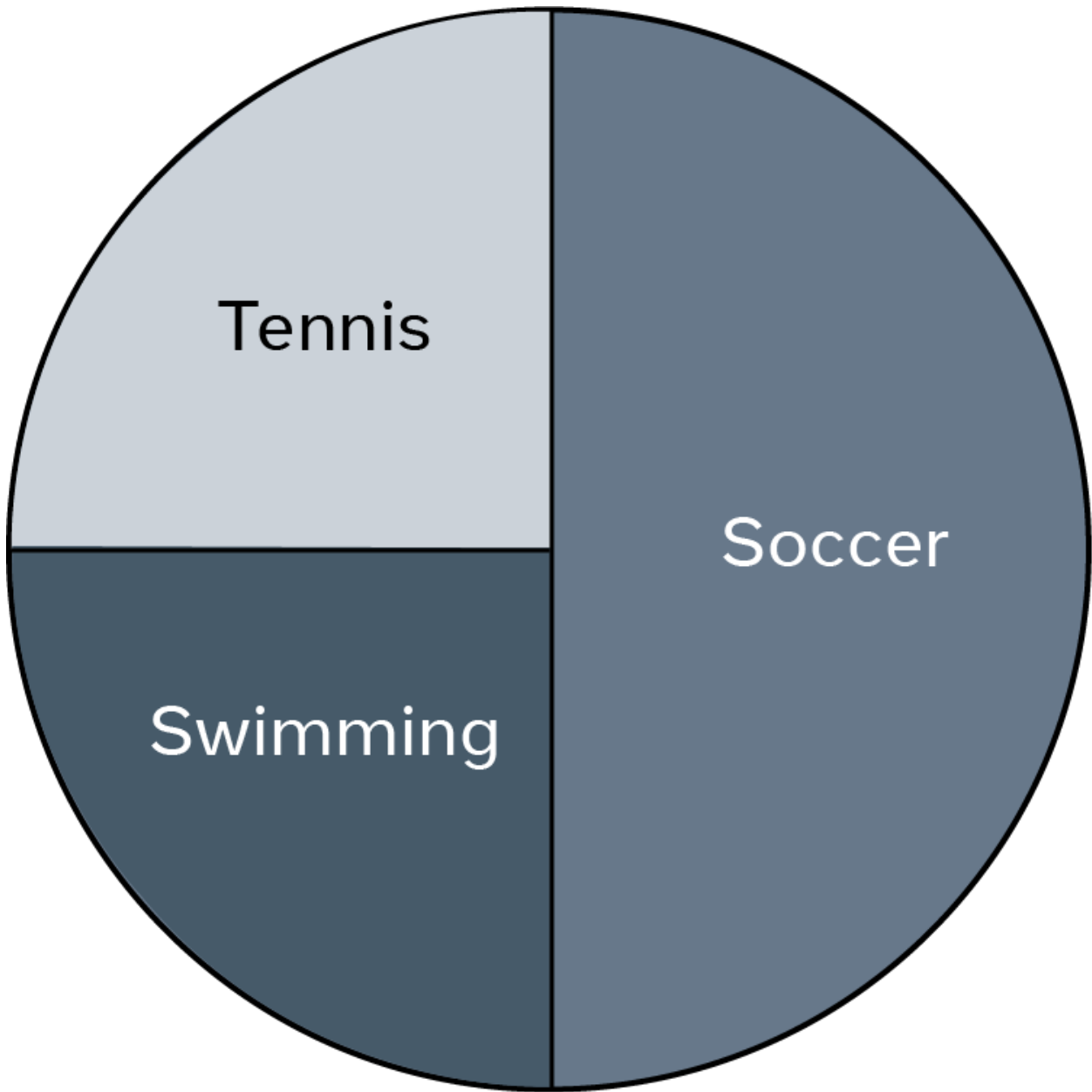
Data analysts commonly use this chart to predict the market's future based on overall trends.



## Pie Chart

A pie chart is another type of data chart that displays how various data make up a whole of 100 percent. In this type of chart, each data point is allocated a "slice" of the pie according to its value.

The following “Sports pie chart” depicts the type of sport students prefer in a class.



If you ask someone looking at this chart about the percentage of students who like soccer, their answer will be 50%, as it is the slice's size that helps them to identify the percentage. In this case, "Soccer" occupies half of the pie and, therefore, it is 50% of the whole. "Tennis" and "Swimming" represent the other half of the pie. Since they are equal, each is a quarter of the pie, which is 25% of the whole.

In addition to the charts introduced earlier in this reading, other charts could be used for different purposes. An example is the area chart, which combines the line chart and the bar chart to compare two or more quantities of data. Other commonly used charts include:

- dual axis charts,
- Gantt charts,
- heat maps
- and scatter plot charts.

# What chart do I choose to present my data?

Some charts can serve multiple purposes, whereas others are much better at conveying specific types of information to the audience. Line charts, for example, are best used to identify trends that help predict the future. Pie charts are a simple way to show how various parts create a whole. They are also quite easy to build. However, it's difficult to add a percentage to each slice if there are many slices or if the slices are not exactly a half, a quarter, or a third of the whole.

The answer to the question depends on several factors, including:

- the target audience who will use the information,
- the idea you intend to present,
- and the goal you want to achieve.

Your choice of chart will be determined by the message you want to deliver to your audience, the type and amount of data you want to load to the graph, and so on.

Once you have identified the audience and assessed the data, you can experiment with different charts to find the best option. If multiple charts are suitable to present your data, choose the one that engages your audience and boosts their interest in the information.

By considering all these factors, you should be able to identify the most appropriate chart that serves your purpose.

# Alternative types of databases (video)

## Database Evolution

**NOTE: This is an optional reading. Don't worry if you don't understand all this material. Many of these concepts will be covered at a later stage in the course.**

You've learned what a database is and how data is related in a database. You've also learned about common database projects and alternative types of databases. In this reading, you'll find out more about the history of database technology and how it evolved.

## The history

The history of databases begins in the 1960s with the computerization of databases. Computers emerged as a more cost-effective option for organizations. It also became easier to shift data storage and databases to computers.

The chronological order of the development of databases is as follows:


- (1970s-1990s) - Flat files, hierarchical and network
- (1980s-present) - Relational
- (1990s- present) - Object-oriented, object-relational, web-enabled

## Flat files

Flat files databases were used during the 1970s-1990s. This is a type of database system that stores data in a single file or table. They are basically text files, where every line contains one record and fields either have fixed lengths or are separated by commas, whitespaces and tabs. Such a file cannot contain multiple tables.

Below is an example of what a flat file database looks like. This text file stores lines of data, where each line represents a record. The fields OrderID, CustomerID and OrderDate are separated by commas.

---

 \*Untitled - Notepad

---

File Edit Format View Help

```
"OrderID", "CustomerID", "OrderDate"
"01", "001", "06/06/2021"
"02", "369", "06/06/2021"
"03", "151", "06/06/2021"
"04", "014", "06/06/2021"
"05", "061", "06/06/2021"
"06", "220", "06/06/2021"
```

## Hierarchical database systems

Hierarchical database systems that were in use during the same era store data in a hierarchically arranged manner.

Think about it this way: parents can have many children, but one child can only have one parent. In other words, the database represents a one-to-many relationship: all attributes of a specific record are listed under an entity type.

Below is an example of how data is stored in a hierarchical database. In this case, it is data on college students who are taking different courses. A course can be assigned to only a single student, but a student can take as many courses as they want. Thus, there is a one-to-many relationship.

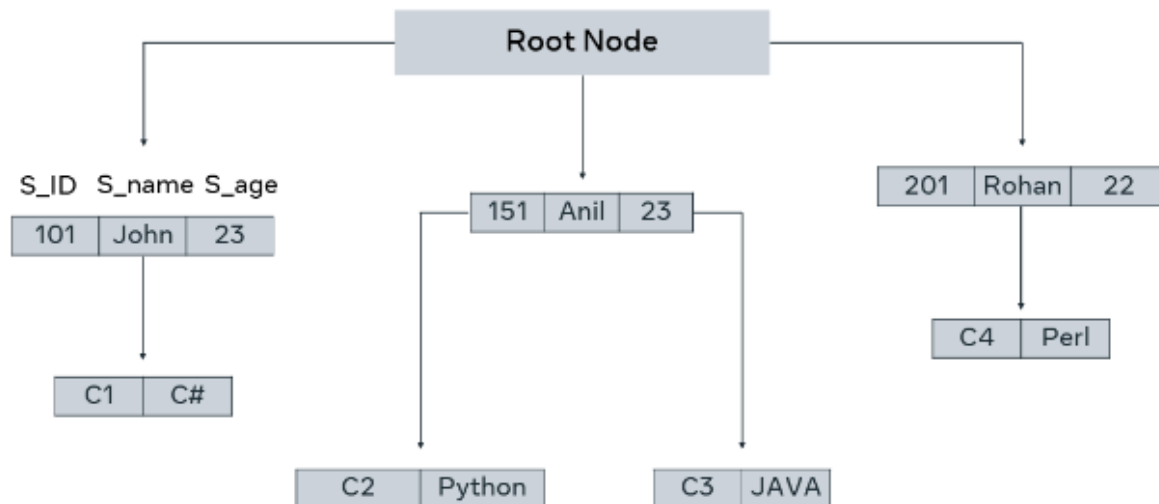
There are three students:

- John,
- Anil
- and Rohan

And there are four courses:

- C#,
- Perl,
- Python
- and Java.

Student and Course are the entity types. John takes C# and Anil takes both Python and Java. Rohan takes Perl.

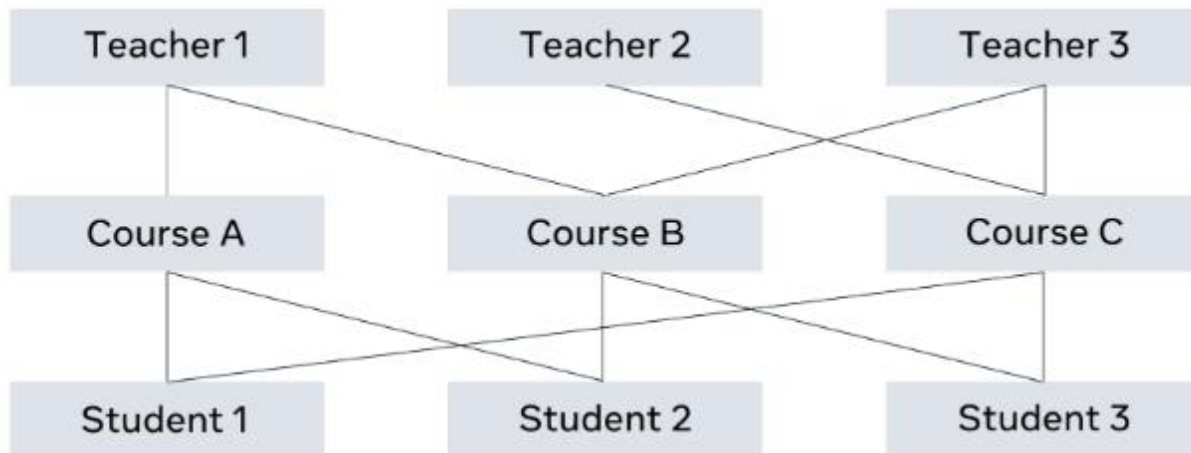


## Network databases

Network databases were introduced by Charles Bachmann. Unlike the hierarchical database model, a network database allows multiple parent and child relationships. In other words, many-to many relationships. In network database terminology, a child record is known as a member. A member or child can be reached through more than one parent, which is called an owner.

A network database has a graph-like structure, and it allows you to represent more complex relationships among data.

Here's an example of a network database. A teacher can teach multiple courses and a course can have multiple teachers teaching it.



In this era, a language known as the SEQUEL query language was used to work with databases. Later on, with relational databases, this developed into SQL (Structured Query Language) which was made a standard query language to work with databases by the American National Standards Institute once relational database systems were introduced.

## Relational database system

The relational database system that was introduced in the 1980s is still the most used database system. It was invented by E. F. Codd and it's the successor of hierarchical and network database systems. It was viewed as a major paradigm shift in database technology.

In a relational database system, data is stored in tables. The columns of the table hold attributes of the data. Each record usually has a value for each attribute, making it easy to establish the relationships between data points. In a relational database, each row in the table is a record with a unique ID attribute called the primary key. A relational database stores and provides access to data that are related to one another using an attribute known as a foreign key.

Here's an example of what a Relational Database would look like. Here, there are tables with attributes/ columns that store rows/records of data in them. The relationships between data in tables are established using key columns known as foreign keys that are themselves primary key(s) of a given table. For example, the primary key of the PROFESSOR table is PROF\_ID and in the CLASS table, it's there as a foreign key. It creates the relationship between the PROFESSOR table and the CLASS table. Another example, the COURSE\_ID is the primary key of the COURSE table, and it is there in the CLASS table as a foreign key. It establishes the relationship between the COURSE table and the CLASS table.

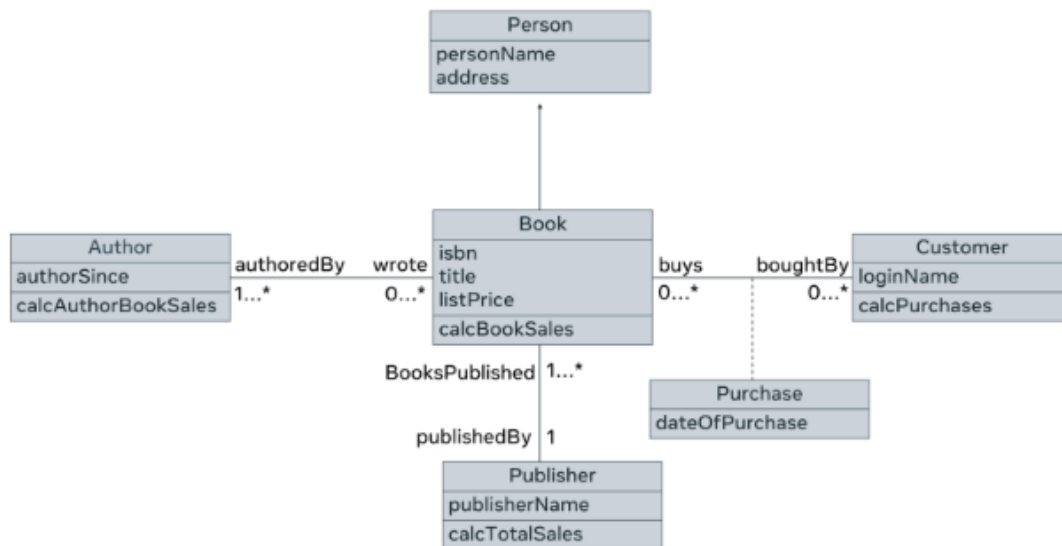
## Object-oriented databases

In the 1990s, object-oriented databases were introduced. This was when the object-oriented (OO) programming paradigm became popular and there was a need to represent data in a system as objects as well. Unlike relational databases, object-oriented databases work in the framework of real programming languages like Java and C++, for example.

Below is what an object-oriented database looks like. Instead of tables, there are entities or classes like Author, Book and Customer with their attributes and behaviors.



It's possible to represent data according to OO concepts like inheritance and parent-child relationships among data. For example, an Author and Customer are both descendants of Person. Thus, a person is a generic entity that can represent both an Author and a Customer.



## NoSQL databases

Relational databases that are widely used even at present only allows to store structured data. Later on, there was a need to work more and more with unstructured data. This was when NoSQL databases came about as a response to the Internet and the need for faster speed and the processing of unstructured data. NoSQL databases are preferred over relational databases because of their speed and flexibility in storing data. It does not store data in relations or tables that belong to a strict structure. Data can be stored in an ad-hoc manner and they allow to store and process high volumes of different kinds of data. NoSQL databases are capable of processing unstructured big data that's generated by social media, IoT and others. Therefore, social platforms like Twitter, LinkedIn, Facebook, and Google for example makes use of NoSQL databases.

These are some of the advantages of NoSQL databases:

- Higher scalability
- Distributed
- Lower costs
- A flexible schema
- Can process unstructured and semi-structured data
- Has no complex relationships

Over time there were different types of NoSQL databases that were introduced:

- Document databases store data in documents similar to JSON (JavaScript Object Notation) objects. Each document contains pairs of fields and values. The values can typically be a variety of types including things like strings, numbers, booleans, arrays, or objects.
- Key-value databases are a simpler type of database where each item contains keys and values.
- Wide-column databases store data in tables, rows, and dynamic columns.

- Graph databases store data in nodes and edges. Nodes typically store information about people, places, and things, while edges store information about the relationships between the nodes.

## Additional resources

The following resources are some additional reading material that introduces you to the concept of a database, different types of databases, about relational databases in specific and also about the history of databases. These will add to the knowledge that you've got on these areas throughout this lesson.

- [Oracle](https://www.oracle.com/uk/database/what-is-database/)
- <https://www.oracle.com/uk/database/what-is-database/>
- 
- [Javapoint](https://www.javatpoint.com/types-of-databases)
- <https://www.javatpoint.com/types-of-databases>
- 
- [IBM](https://www.ibm.com/topics/relational-databases)
- <https://www.ibm.com/topics/relational-databases>
- 
- [Tutorialspoint](https://www.tutorialspoint.com/sql/sql-databases.htm)
- <https://www.tutorialspoint.com/sql/sql-databases.htm>
- 
- [Graphdatamodeling](http://graphdatamodeling.com/GraphDataModeling/History.html)
- <http://graphdatamodeling.com/GraphDataModeling/History.html>