**Mongo I**

**¿Qué hacemos con los datos?**

\*“Los caminos nos ayudan a llegar a un lugar. Los datos nos ayudan a tomar una decisión.” Victor Rico — Hispanic/Latino Builder @Facebook with the @WhatsApp team\*



[Fuente](http://www.freepik.es/)

Las formas de almacenar información se van modificando y sofisticando constantemente debido al incremento y la demanda de datos. Como ya sabes, en programación contamos con distintos sistemas de almacenamiento. Hasta ahora, guardamos información en variables, arrays o archivos. Con las dos primeras sucede que si reiniciamos el server, la información se pierde. Con la última, al guardar los datos en archivos, sucede que debemos acceder al disco todo el tiempo lo cual es un problema, porque es lento. Esto nos trae complicaciones cuando queremos guardar o editar dos datos al mismo tiempo. Entonces desde el backend surge la necesidad de guardar y leer la información que ya ha sido modificada o eliminada, conocida como [persistencia de datos](https://es.wikipedia.org/wiki/Persistencia_de_datos). Por este motivo existen las bases de datos (BD, por sus siglas en inglés). Según el libro de Github, [*Sistemas de Información Geográfica*](https://volaya.github.io/libro-sig/index.html):

\*".Las bases de datos son un elemento fundamental en el entorno informático hoy en día y tienen aplicación en la práctica totalidad de campos. Concebidas con un propósito general, son de utilidad para toda disciplina o área de aplicación en la que exista una necesidad de gestionar datos, tanto más cuanto más voluminosos sean estos." \*

Lasbases de datos son un conjunto de información ordenada de modo sistemático para su posterior recuperación, análisis y/o transmisión.Proveen una forma rápida y ordenada de almacenar los datos y al mismo tiempo garantizan su integridad y seguridad. También se pueden realizar operaciones sobre la informaciónde forma más sencilla, flexible y estructurada.

Entre sus ventajas destacamos:

* **Datos globales.**Permiten compartir información simultáneamente con otras personas usuarias u otras bases de datos.
* **Control de datos.**Chequea la duplicidad o triplicidad de almacenamiento de espacio en disco.
* **Aumento de la productividad.**Permite acceder con mayor velocidad a los datos de forma inmediata.
* **Centralización de la información.**Permite tener los datos en un mismo lugar para que los diversos sistemas puedan trabajar sobre esa BD.
* **Independencia.**Permite autonomía de programas y/o aplicaciones.

Utilizar bases de datos tiene tanto ventajas como desventajas. Hay que tener en cuenta que requieren mucho espacio en disco ya que pueden ir creciendo a medida que se agreguen datos. Si eso pasa, puede llegar a ponerse lenta, afectando las búsquedas y la recuperación de información. Además, un fallo en la BD afectará a todo el entorno. Esto conlleva a un mayor  mantenimiento, y a su vez hace que los costos suban. También puede requerir de capacitación, asesoría y acompañamiento para enseñar a otros/as su manejo. Pero ¡que no cunda el pánico! Profundizaremos sobre cómo hacer para que  las aplicaciones guarden información sobre una base de datos.

**Distintas formas para guardar información**

Imagina que cuentas con una base de datos con una lista de personas invitadas a una fiesta con su “nombre”, “edad” y “preferencia alimenticia”. Podrías saber cuántas personas mayores a 18 años son vegetarianas o cuántas entre 2 a 12 años son menores para identificar los menús infantiles. Esto lo vamos a hacer a partir de un motor de base de datos, una herramienta que permite guardar, manipular y recuperar la información. Existen dos grandes grupos de motores de bases de datos: los relacionales y los no relacionales.

* **Motor BD Relacional (SQL).**Se basan en la organización de la información en trozos pequeños, que se enlazan mediante la relación de identificadores. Guardan cada una de sus entidades (usuarios, vehículos, aviones, reservas, etc) en **tablas** y cada una de las propiedades (para la entidad usuarios, nombre, apellido, email) en **campos.**Los más populares son: [MySQL](https://www.mysql.com/), [Oracle](https://www.oracle.com/index.html) y [Microsoft SQL Server](https://www.microsoft.com/es-es/sql-server/sql-server-downloads).
* **Motor BD No Relacional (No SQL).**A diferencia de las relacionales, guardan la información en colecciones o documentos JSON y no es necesario definir un formato (tablas y campos en las relacionales). Habrá un objeto JSON por cada uno de los valores que desee almacenar.Los más populares son: [MongoDB](https://www.mongodb.com/), [Cassandra](http://cassandra.apache.org/) y [Redis](https://redis.io/).

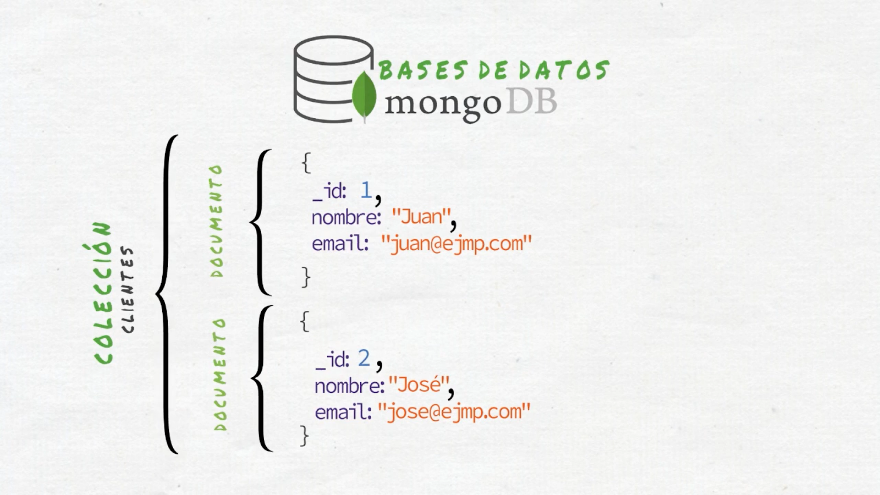
Saber cuál elegir siempre dependerá de las necesidades de un proyecto.

Si tienes un proyecto donde la información que debes almacenar puede sufrir muchas variaciones, la recomendación es utilizar una DB no relacional debido a que no deberás definir una estructura para almacenar la información.

Si tu proyecto tiene un alcance definido y estimas que las variaciones de la aplicación serán menores, una DB relacional te brindará un orden establecido a través de las tablas y campos.

**MongoDB base de datos sin tablas**

Si vas a manipular una gran cantidad de datos y buscas velocidad y flexibilidad en los datos ¡MongoDB es ideal! Esun sistema para la gestión de bases de datos no relacional que utiliza un modelo de documentos basado en JSON. Además utiliza colecciones para agrupar información donde almacena documentos.  Su composición se ve así:



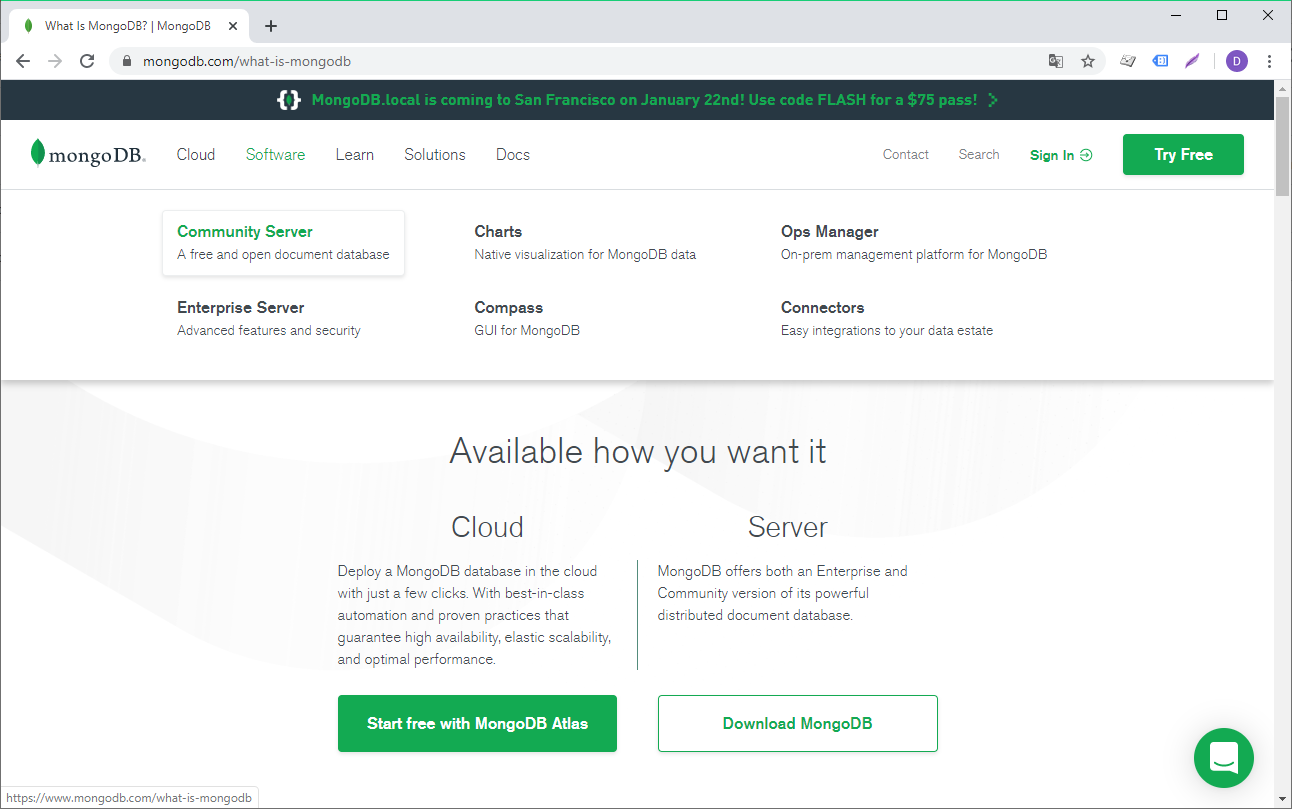
En MongoDB, como menciona [Agustín Navarro Galdon](https://medium.com/@xagustin93) en su [artículo](https://medium.com/@xagustin93/comandos-b%C3%A1sicos-de-mongodb-59978f2a81b7), se habla de [desnormalización](https://es.wikipedia.org/wiki/Denormalizaci%C3%B3n_(base_de_datos)) de datos:

*“[...]significa cambiar totalmente la forma de pensar en una base de datos para nosotros que tenemos años de trabajo usando bases de datos relaciones será difícil, pero una vez que te adaptas todo irá mucho mejor y será mucho más comprensible cada vez que comiences un nuevo proyecto.”*

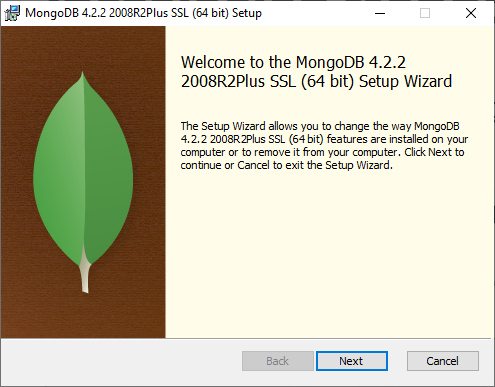
Iremos paso a paso para que llegues a utilizar esta herramientas y luego realizar operaciones [CRUD](https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/crud-las-principales-operaciones-de-bases-de-datos/)(**C**reate, **R**ead, **U**pdate and **D**elete) sobre la base de datos.

**Conexión con MongoDB**

Para instalarlo ingresa [aquí](https://www.mongodb.com/) y descarga la versión Community Server:



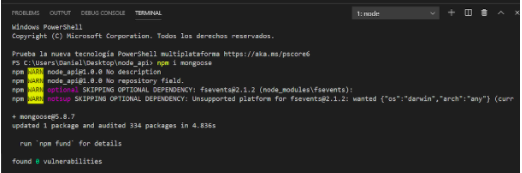
Luego sigue las instrucciones del instalador:



Crearemos nuestro primer script para que se conecte a MongoDB, y poder así comenzar a operar.

**1. Script de conexión**

Primero hay que instalar la librería mongoose que permite conectar nuestra aplicación con Mongodb:



Luego vamos a crear el script que permitirá conectarnos a la base de datos:



**2. Modelos**

Vamos a almacenar nuestro primer dato en Mongo. Para eso nos apoyaremos en Modelos y Esquemas.

* **Primero definimos un esquema**que representa las propiedades y el tipo de valor que guardará.

schema = {nombre: string, apellido: string, edad: Number}

Un esquema tiene diferentes tipos de datos, de acuerdo a lo que quieras almacenar elige el adecuado:

* String
* Number
* Date
* Boolean
* ObjectId
* Array

Algunos te resultarán conocidos, pero hay muchos más. Puedes encontrar la lista completa [aquí](https://mongoosejs.com/docs/schematypes.html).

* **Creamos un modelo con el esquema.**El modelo nos permitirá comunicarnos con la base de datos y nos brindará un montón de métodos para enviar y recibir información.El primer parámetro es el nombre del modelo. El segundo, el esquema (que definimos hace un instante).

const Usuarios = mongoose.model(“Usuarios”, schema);

* **Definimos un objeto**con el dato que queremos almacenar y lo guardamos a través del modelo.

const yo = {nombre: “Juan”, apellido: “Perez”, edad: 24};

let nuevo\_usuario = new Usuarios(yo)

nuevo\_usuario.save();

Observa que en la primera línea del ejemplo creamos un objeto con los datos que deseamos almacenar. En la segunda, instanciamos el modelo Usuarios y enviamos como parámetro el objeto recién creado. En la tercera, ejecutamos el método save para almacenar el objeto en la base de datos.

La instancia del modelo también nos permite buscar información a través del método find.

Usuarios.find().then(function (resultados){

console.log(resultados);

});

El método find es capaz de recibir un parámetro y filtrar de la colección las coincidencias que encuentre.

Usuarios.find({nombre: “juan”}).then(function (resultados){

console.log(resultados);

});

**Operaciones CRUD**

Ahora que ya tienes tu primer script conectado a MongoDB, veamos un poco más acerca de las operaciones CRUD que te mencionamos más arriba. Estas te permitirán realizar una serie de acciones básicas en tu base de datos.

**Create:**Crear. Esta operación te servirá para crear o insertar documentos a una nueva colección. Puedes insertar un solo documento, o varios documentos en una sola operación.

**Read:**Leer. Esta operación te permitirá obtener documentos de una colección. Puedes incluso especificar criterios para identificar los documentos que buscas.

**Update:**Actualizar. Puedes modificar documentos existentes en una colección. Puedes modificar un solo documento, o varios a la vez.

**Delete:**Eliminar. Esta operación te servirá para eliminar documentos existentes de una colección. Al igual que las demás operaciones, podrás eliminar un solo documento como varios a la vez.

**Cierre**

Hasta aquí ya sabes qué son las bases de datos y qué ventajas tienes al utilizarlas, tanto para el manejo de datos como para la seguridad de la información. También ya logras diferenciar los métodos de bases de datos y aprendiste un método no relacional ¡Confía! Iremos despacio. En la próxima meeting profundizaremos en las operaciones con Mong RD.