

**1970** -> Se crea el primer modelo de Bdatos. Una serie de lineamientos sobre como deberíamos guardar la información, como organizarla y luego como consultarla cuando la necesitemos. Hasta ese entonces no existía un modelo no había un estándar (SQL), tan eficiente que domino la tecnología por décadas.

Sin embargo, la tecnología avanza tanto que hacia los años 2000 **SQL** ya no podía responder al tipo de necesidades de las nuevas empresas de la web como **Google, Amazon, Facebook** entre otros.

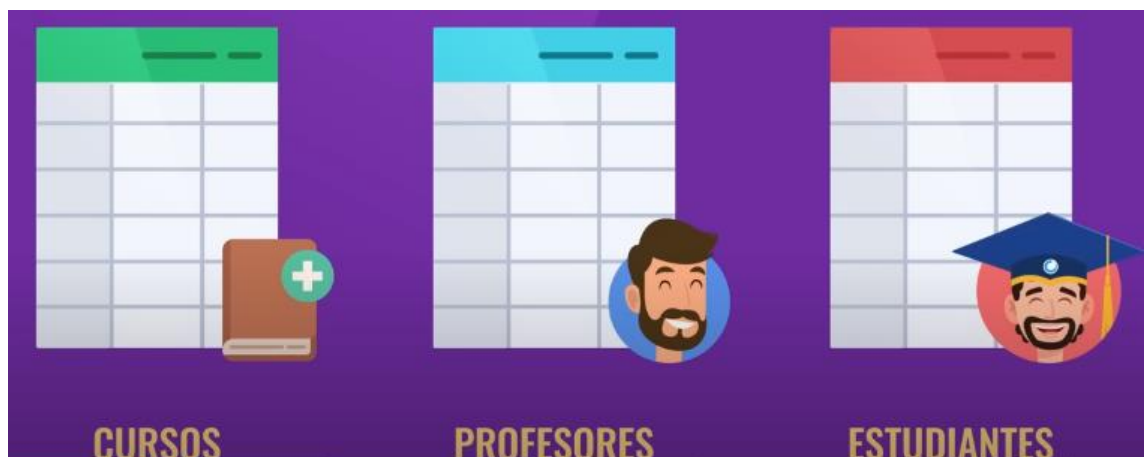
Es entonces cuando aparecen las bases de datos **NoSQL**, que traen un paradigma diferente de almacenar los datos.

**¿Entonces que tipo de bdatos es mejor las SQL o las NoSQL?**

¿Deberíamos dejar las SQL porque son antiguas y utilizar las NoSQL porque son modernas? **¡DEPENDENDE!**

**Hablemos de SQL primeramente.....**

- Organizan la información en tablas. Y cada tabla se utiliza para guardar un tipo especifico de información como en Excel.




Así que en SQL crearíamos una tabla para cada una de esas entidades. Ahora si quiero mostrar los cursos que dicta un profesor, tomo los datos del profesor, los datos de los cursos los junto y los muestro. Gracias a esta forma de mostrar los datos evitamos las redundancias (**Normalización de bases de datos**) que es la duplicación de los datos.

Estas tablas tienen registros y campos. Cada registro es un nuevo elemento de la entidad. Ejemplo en la tabla de profesor:

REGISTROS

CAMPOS

NOMBRE	APELLIDO	EDAD	PAIS	EMAIL
Carla	Sanchez	36	Colombia	carla@com



Los campos (**NOMBRE, APELLIDO, EDAD, PAIS, EMAIL**), obviamente de esta forma vamos llenando el campo para cada uno de esos profesores que vamos añadiendo a la base de datos.

Las tablas tienen además **FOREIGN KEYS** (claves foraneas), que no es otra cosa que un campo que permite crear las relaciones entre las tablas y además restringir las operaciones que se pueden hacer con esos datos.

Por ejemplo, si tienes una tabla de **profesores** y una de **cursos**, podría poner la restricción de que un profesor no se

puede eliminar si tiene cursos, porque no podemos tener un curso sin profesor.

Como pudimos observar la prioridad de las bases de datos relacionales es **la prioridad de los datos**.

Existen 2 cosas importantes que debemos de tener claro en bases de datos relacionales es:

DIFERENCIAS	
BASES DE DATOS	MOTOR DE BASES DE DATOS
Son los datos nada más.	Software con el cual se administran estos datos, se organizan, se estructuran en tablas, se consultan se llama SISTEMA GESTOR DE BASES DE DATOS.
¡BASES DE DATOS != MOTOR DE BASES DE DATOS	



## EXISTEN DIFERENTE MOTORES DE BASES DE DATOS

<b>(Oracle 1977)</b>	Primer BD SQL comercial.
<b>(SQL Server 1989)</b>	Solo para Windows.
<b>(SQL Server 2006)</b>	Multiplataforma
<b>(MySQL 1995)</b>	Mas utilizado en la web
<b>(PostgreSQL 1996)</b>	Mas completo que MySQL
<b>SQLite</b>	Se encarga en guardar la información de las aplicaciones de los teléfonos en la misma aplicación en lugar de hacerlo en lugar de estar en un servidor diferente.
<b>MariaDB</b>	Completamente compatible con MySQL

**IBM** -> Creo el lenguaje SQL.

**Oracle** -> Motor de bases de datos.

## BASES DE DATOS NoSQL (Bd No Relacionales)

Supongamos que vamos a realizar un post tipo Instagram utilizando SQL necesitaríamos:

- Una tabla para -> usuarios, Imágenes, Post, Likes.
- Luego extraer la información de cada una de esas tablas juntarlas y mostrar el post. Realmente es muy fácil.



El problema real es que para los años 2000, las aplicaciones tenían usuarios de todo el mundo, **¡Eso nunca antes visto!**, con respecto a los años anteriores. Pero entrando al año 2000 eran tan pero tan masiva que todo el planeta estaba entrando a la web. Y era la primera vez que se veían aplicaciones que tenían clientes y usuarios de todo el mundo a la vez.

En ese escenario las bases de datos se volvían lentas por eso de estar consultando de una tabla de la otra etc. Uniendo la información entonces se crea un siguiente modelo denominado: **NoSQL** que niega los principios de **SQL**.

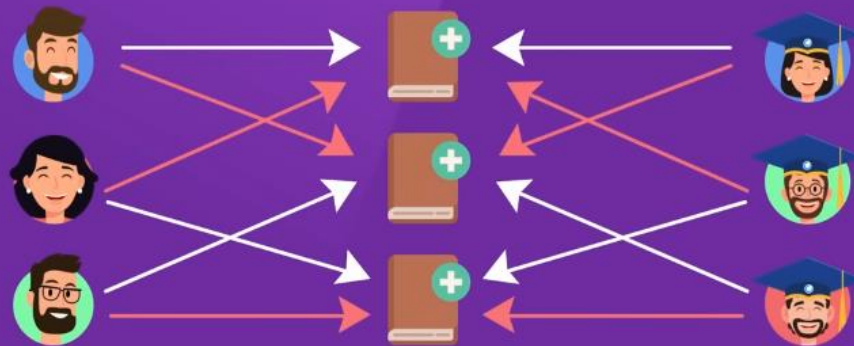
Las bases de datos **NoSQL** no organizan la información en tablas **porque justamente dividir la información en tablas es lo que las hace lentas cuando hay muchísimos usuarios concurrentes.**



En otras palabras, en **NoSQL**, hay redundancia, no hay normalización, no hay claves foraneas ósea sálvese quien pueda porque la integridad de los datos esta en riesgo.

Se acuerdan el ejemplo anterior de que era mala idea de que el profesor se repita de curso. ..

## BASES DE DATOS SQL (RELACIONALES)



PROFESORES

CURSOS

ESTUDIANTES

En **NoSQL**, no importa que se repita lo importante es traerme esa información rapidito. Quiero tener todo empaquetado junto.

### Resumen.

- **Redundancia de datos.**
- **Velocidad sobre integridad.** Lo mas importante es acceder a los datos rapidito.
- **No existe un lenguaje de consultas.** (En las relacionales existe el SQL. Acá no.)

Entonces puesto que en **NoSQL** pueden guardar la información de diferentes maneras, sin ajustarse a las famosas tablas, EXISTEN 3 TIPOS DE BASES DE DATOS NO RELACIONALES (NoSQL):

1. **CLAVE VALOR.** Información de parejas clave valor. Como diccionario donde esta la palabra y luego la definición. la **clave es un identificador único**, y en el lado del valor se guarda la información, entonces con ese identificador único podemos acceder a la información que se encuentra allí.



Donde se usan las bases de datos de clave y valor principalmente en sesiones de usuario donde se va guardando toda su información. O por ejemplo en carritos de compra. Justamente **Amazon Dynamo DB**, fue quien creo el motor de bases de datos **clave valor** más grande, porque perdían mucho dinero si los usuarios demoraban en agregar productos al carrito.

**2. DOCUMENTALES. (MongoDB)** Este tipo guarda la información en documentos. Son archivos de tipo **json**, y puesto que json es un estándar.

**3. DE GRAFOS. (mis favoritos. Facebook)**

Motores más conocidos tenemos:





## SEGURIDAD DE BASES DE DATOS NoSQL

La seguridad en las bases de datos NoSQL varía según la implementación y la configuración específica de cada sistema. Aquí hay algunos aspectos a considerar en cuanto a la seguridad de las bases de datos NoSQL:

**Autenticación y Autorización:** Las bases de datos NoSQL suelen ofrecer mecanismos de autenticación y autorización para controlar quién puede acceder a los datos y qué operaciones pueden realizar. Esto puede incluir la autenticación basada en usuarios y contraseñas, así como roles y permisos para controlar el acceso a bases de datos, colecciones o documentos específicos.

**Encriptación de Datos:** Muchas bases de datos NoSQL ofrecen opciones para cifrar los datos almacenados en reposo y en tránsito. Esto ayuda a proteger los datos sensibles de accesos no autorizados o de la interceptación durante la transmisión.

**Seguridad a Nivel de Red:** Además de las características específicas de la base de datos, la seguridad de una implementación NoSQL también depende de las medidas de seguridad aplicadas a nivel de red, como firewalls, VPNs y configuraciones de red seguras.

**Prácticas de Seguridad del Desarrollo de Aplicaciones:** A menudo, la seguridad de una base de datos NoSQL también está influenciada por las prácticas de seguridad del desarrollo de la aplicación que interactúa con la base de datos. Esto incluye la validación de datos, la prevención de inyecciones de código y otras vulnerabilidades comunes.

En resumen, si se implementan adecuadamente con las medidas de seguridad apropiadas, las bases de datos NoSQL pueden ser tan seguras como las bases de datos relacionales tradicionales. Sin embargo, es importante comprender y aplicar las mejores prácticas de seguridad tanto a nivel de base de datos como en el desarrollo de aplicaciones que utilizan estas bases de datos.

