**Diferencias:**

* SQL permite combinar de forma eficiente diferentes tablas para extraer información relacionada, mientras que NoSQL no lo permite o muy limitadamente.
* NoSQL permite distribuir grandes cantidades de información; mientras que SQL facilita distribuir bases de datos relacionales.
* SQL permite gestionar los datos junto con las relaciones existentes entre ellos; en NoSQL no existe este tipo de utilidades.
* NoSQL permite un escalado horizontal sin problemas – por su capacidad de distribución-; mientras que escalar SQL resulta más complicado.

Caracteristicas

* Aporta escasas garantías de consistencia.
* Algunos sistemas noql aportan todas las garantias de los sistemas ACID (Atomicity, Consistency,Isolation(aislamiento), durability(persistencia)), añadiendo una capa intermedia.
* Escalabilidad:

Capacidad de un sistema para adaptarse a las necesidades de rendimiento a medida que el volumen de datos y de transacciones aumentan.

* Escalabilidad Horizontal:

Se tienen varios servidores (nodos) trabajando como un conjunto, a este conjunto o red se le llama Cluster, si el performance del cluster se ve afectado, se añaden nuevos nodos al cluster.

Debe existir un servidor primario que administra el cluster. Esto permite un crecimiento infinito, si un nodo falla los demás siguen trabajando, soporta el balanceo de cargas.

* al ser open source se depende mucho de la comunidad.
* Falta de madurez en el mercado.
* Limitaciones en BI
* No hay un Estándar de tipos de datos, cada bd nos+++, c#, **Javascript**. (en MongoDB)

Aplicaciones

**¿CUÁNDO USAR NOSQL?**

Básicamente se utilizan en:

* Redes sociales: casi obligatorio.
* Desarrollo Web: debido a la poca uniformidad de la información que se encuentra en Internet; aun cuando también puede emplearse SQL.
* Desarrollo Móvil: debido a la tendencia – en crecimiento- de Bring Your Own Device.
* BigData: debido a la administración de grandísimas cantidades de información y su evidente heterogeneida.
* Cloud (XaaS): “Everything as a service”; NoSQL puede adaptarse casi a cualquier necesidad del cliente, y sus particularidades.

**SQL**

* Educación: para estructurar información, y aportar conocimiento lógico al estudiante.
* Desarrollos web: para mantener jerarquía de datos, siempre y cuando la capacidad de concurrencia, almacenamiento y mantenimiento no sean de considerable dificultad y la información sea consistente.
* Negocios: inteligencia y análisis de negocios, son temas que requieren el uso de SQL para facilitar el consumo de la información y la identificación de patrones en los datos.
* Empresarial: porque tanto el software a la medida y el software empresarial, poseen la característica de mantener información con estructura consistente.

### **Visión unificada**

La primera de estas categorías incluye a la empresa **MetLife** . El gigante de los seguros trabaja con MongoDB para conseguir según afirman “una visión de 360 grados” de sus más de cien millones de clientes. La aplicación consiste en la creación de un repositorio central que ofrece una visión a partir de muchas fuentes de datos que provienen de otros repositorios o sistemas.

### **Internet de las cosas**

En el mundo del Internet de las cosas, todos los dispositivos están conectados entre sí, generando y compartiendo información. Así, la empresa **Bosch** está poniendo a prueba una aplicación que es capaz de capturar datos del vehículo como el sistema de frenado, la dirección asistida, los limpiaparabrisas, etc. Con todos esos datos capturados, se pueden hacer diagnósticos de necesidad de mantenimiento preventivo.

### **Tecnologías móviles**

**The Weather Channel** utiliza MongoDB para sus aplicaciones móviles que disfrutan cerca de cuarenta millones de personas en todo el mundo. La idea de que MongoDB se convierta en la parte backend del mundo de los smartphones y tablets toma cada vez más fuerza.

### **Analítica en tiempo real**

Cuando hablamos de analítica en tiempo real nos referimos a la necesidad de conseguir resultados de manera inmediata. **La ciudad de Chicago** ha desarrollado una aplicación llamada WindyGrid que está basada en MongoDB, y recoge datos de policía, transporte e incendios. La aplicación además notifica alertas por obras en carretera, retrasos en recolección de basura, quejas por ruido o tweets públicos, entre otros.

### **Personalización**

La aplicaciones con base en la personalización buscan crear experiencias hechas a la medida de los usuarios y en tiempo real. Para ello es necesario hacer un análisis rápido y certero del perfil de usuario, comportamiento, datos demográficos, gustos, etc. La agencia de viajes norteamericana **Expedia** ha creado Scratchpad, una aplicación basada en MongoDB que permite al usuario disponer de un bloc de notas digital para almacenar y personalizar sus búsquedas. A partir del seguimiento de las búsquedas y comparaciones, Expedia ofrece a sus clientes viajes y ofertas especiales en tiempo real.

### **Administración de contenido**

Los permiten gestionar archivos junto a sus metadatos. **Forbes** construyó todo su sistema de gestión de contenidos en MongoDB. Además, utiliza MongoDB para analítica en tiempo real, al igual que el Ayuntamiento de Chicago. Cuando algún artículo se hace viral, Forbes detecta la forma en que se está compartiendo entre los usuarios y de este modo sabe qué tipo de contenido le debe ofrecer a sus lectores.

1. Administración de perfiles de cuentas de usuarios

2. Aplicaciones Real Time

3. Big Data

4. Gestión de contenidos

5. Administración de clientes 360°

6. Aplicaciones móviles

7. Internet de las cosas (IoT)

8. Comunicación Digital

9. Inteligencia Artificial (AI)

10. Sistemas de Anti-fraudes

11. Sistemas Centralizados de Eventos y Transacciones

12. Analítica Web

<https://platzi.com/blog/aplicaciones-bases-datos-no-relacionales-nosql/>

<https://www.silicon.es/bases-datos-no-relacionales-nosql-cuando-usarlas-2324948?inf_by=5afc432b671db874338b4a36>

<https://smarterworkspaces.kyocera.es/blog/por-que-las-bases-de-datos-relacionales-no-manejan-bien-el-big-data/>

<https://sg.com.mx/content/view/1097>

<https://dl.acm.org/citation.cfm?id=2800963>

db documental.

* JSON no es el lenguaje de consulta, en un formato que permite estructurar un doc, para que un lenguaje de programación pueda comprenderlo.
* Se constituye por un conjunto de programas que almacenan, recuperan, gestionan datos estructurados (ej:docs)
* En general todas comparten que los docs encapsulan y codifican datos siguiendo algún formato estándar. (ej de codificacion: XML,YAML,JSON,BSON).
* Los documentos no tienen que ajustarse a algún esquema a diferencia de las bd relacionales. PUEDE QUE DOS O VARIOS DOCS TENGAN informacion similar y otra diferente.
* Los documentos se direccionan por una llave primaria, la bd tiene un indice de pk lo que permite una rapida recuperacion.
* Hay distintas maneras de implementar la organizacion de los docs( ej: colecciones, etiquetas, Metadatos oculos, jerarquias de directorios.

NO SQL

DEJE DE HACER COPY PASTE :V

El término NoSQL aparece con la llegada de la web 2.0 ya que hasta ese momento sólo subían contenido a la red aquellas empresas que tenían un portal, pero con la llegada de aplicaciones como Facebook, Twitter o Youtube, cualquier usuario podía subir contenido, provocando así un crecimiento exponencial de los datos. Es en este momento cuando empiezan a aparecer los primeros problemas de la gestión de toda esa información almacenada en bases de datos relacionales. En un principio, para solucionar estos problemas de accesibilidad, las empresas optaron por utilizar un mayor número de máquinas pero pronto se dieron cuenta de que esto no solucionaba el problema, además de ser una solución muy cara. La otra solución era la creación de sistemas pensados para un uso específico que con el paso del tiempo han dado lugar a soluciones robustas, apareciendo así el movimiento NoSQL. Por lo tanto hablar de bases de datos NoSQL es hablar de estructuras que nos permiten almacenar información en aquellas situaciones en las que las bases de datos relacionales generan ciertos problemas debido principalmente a problemas de escalabilidad y rendimiento de las bases de datos relacionales donde se dan cita miles de usuarios concurrentes y con millones de consultas diarias. Además de lo comentado anteriormente, las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad–relación. Tampoco utilizan una estructura de datos en forma de tabla donde se van almacenando los datos sino que para el almacenamiento hacen uso de otros formatos como clave–valor, mapeo de columnas o grafos (ver epígrafe ‘Tipos de bases de datos NoSQL’).

NoSQL puede servir gran cantidad de carga de lecturas y escrituras.

Las arquitecturas NoSQL frecuentemente aportan escasas garantías de consistencia, tales como consistencia de eventos o transaccional restringida a ítems únicos de datos. Algunos sistemas, sin embargo, aportan todas las garantías de los sistemas ACID en algunas instancias añadiendo una capa intermedia (como por ejemplo, AppScale o CloudTPS). Hay dos sistemas que han sido desplegados y que aportan aislamiento snapshot para almacenamientos de columna: El sistema Percolator de Google (basado en el sistema BigTable) y el sistema transaccional de Hbase desarrollado por la universidad de Waterloo.

Bastantes sistemas NoSQL emplean una arquitectura distribuida, manteniendo los datos de forma redundante en varios servidores, usando frecuentemente una tabla hash distribuida. De esta forma, el sistema puede realmente escalar añadiendo más servidores, y el fallo en un servidor puede ser tolerado.

bibliografía

Lith, Adam; Mattson, Jakob (2010). Investigating storage solutions for large data

Göteborg: Department of Computer Science and Engineering, Chalmers University of Technology