Introducción a Bases de Datos NoSQL

Paulina Bermudez
Fernanda Espinoza
Paola Franco
Jocelyn Jaimes

Diseño de bases de datos relacionales

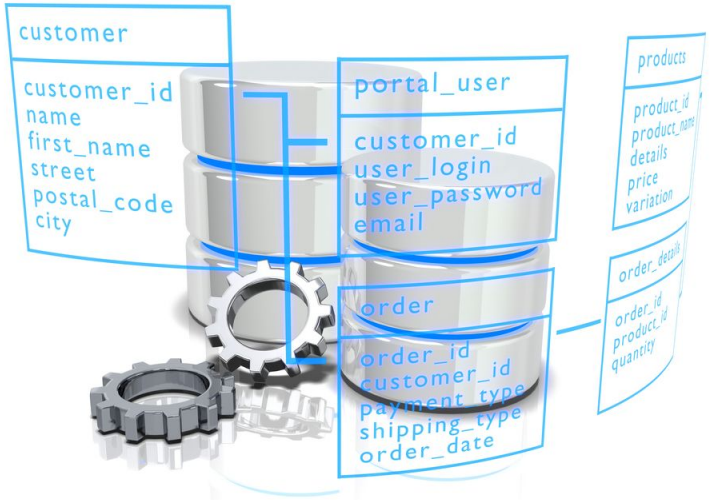
Las **bases de datos relacionales** se diseñan para soportar cientos de usuarios simultáneamente.

Una de sus características más importantes es asegurar que todos los usuarios que visualicen los datos tengan una visualización consistente de éstos, así como tener tiempo, almacenamiento y recursos computacionales.

Diferentes aplicaciones necesitan diferentes tipos de bases de datos.

La comunidad de manejo de datos actual generó nuevos sistemas de manejo de datos, conocidos como **bases de datos NoSQL**. El “No” puede significar que no se utiliza SQL en la base de datos, o que se usa de una manera muy mínima.

Algunas de las características que se encuentran en los primeros sistemas de manejo de bases de datos aparecieron de nuevo en las bases de datos NoSQL.



Primeros sistemas de manejo de bases de datos:

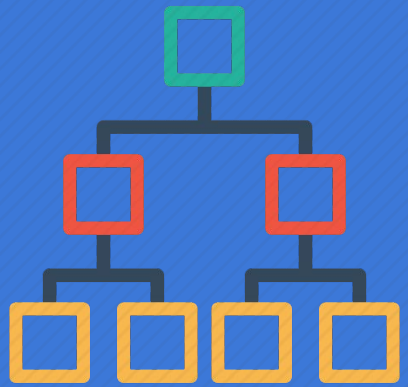
Sistemas de gestión de datos de archivos planos

Un **archivo** es una colección de datos organizados guardados en un medio a largo plazo, que de los 50s a los 70s fueron en **cintas magnéticas**.

Algunos de los problemas asociados con este tipo de sistema es que llevan a datos duplicados, inconsistencia de datos, problemas de seguridad e ineficiencia para acceder a los datos.

Primeros sistemas de manejo de bases de datos:

Sistemas de gestión de datos jerárquicos



Este tipo de modelos organizan los datos mediante una jerarquía de **relaciones padres-hijos**.

Mejoraron con respecto a los archivos planos al hacer búsquedas más eficientes.

Sus limitaciones incluyen uso ineficiente del espacio de almacenamiento al duplicar datos, inconsistencias y potencial de errores al añadir datos.

Primeros sistemas de manejo de bases de datos:

Sistemas de gestión de datos de red

Tiene dos componentes esenciales: un **esquema** y una **base de datos**.

Los registros de datos se conocen como **nodos** y las uniones entre ellos son **orillas**. La colección de nodos y orillas se conoce como **grafos**.

La mayor limitante de este tipo de bases de datos es que son difíciles de diseñar y mantener.

La revolución de las bases de datos relacionales

En 1970 E. F. Codd publicó las características para un nuevo modelo: la **base de datos relacional**.

Utiliza un modelo matemático para describir los datos y sus relaciones, el cual elimina anomalías como la inconsistencia de datos.

Un **sistema de administración base de datos relacionales** le permite al usuario añadir, actualizar, leer y borrar datos a través de un lenguaje llamado **SQL**.

Organización de los sistemas de administración de bases de datos relacionales

Programas de administración del almacenamiento

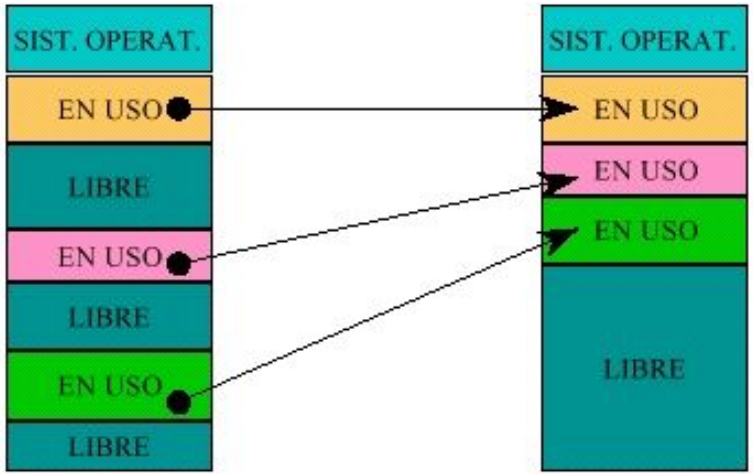
En su nivel más básico las RDBMSs deben de ubicar dónde se guardan los datos. Para ello, son capaces de leer y escribir bloques de datos accesibles a través de **índices**.

Asimismo, optimizan la posición de los datos en discos al comprimir datos y realizar respaldos.

Organización de los sistemas de administración de bases de datos relacionales

Programas de administración de la memoria

Estos componentes son los responsables de traer y mantener datos en la memoria cuando se necesite, y borrarlo cuando ya no se utilicen para crear espacio para datos adicionales.

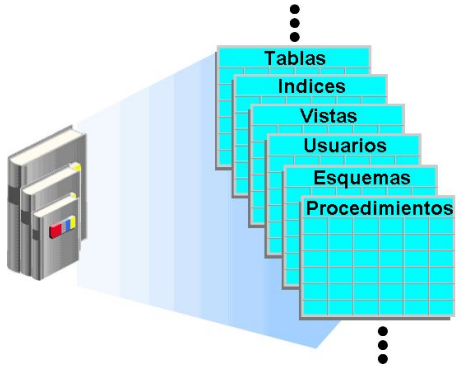


Organización de los sistemas de administración de bases de datos relacionales

Diccionario de datos

Mantiene la información acerca de las estructuras de datos almacenados. Se incluye la información de diversos niveles de la base de datos, incluyendo:

- Esquemas
- Tablas
- Columnas
- Índices
- Restricciones
- Vistas

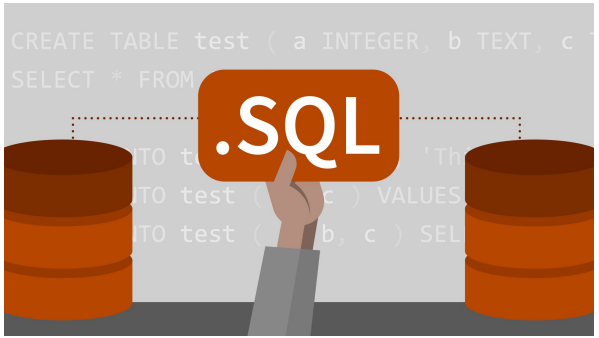


Organización de los sistemas de administración de bases de datos relacionales

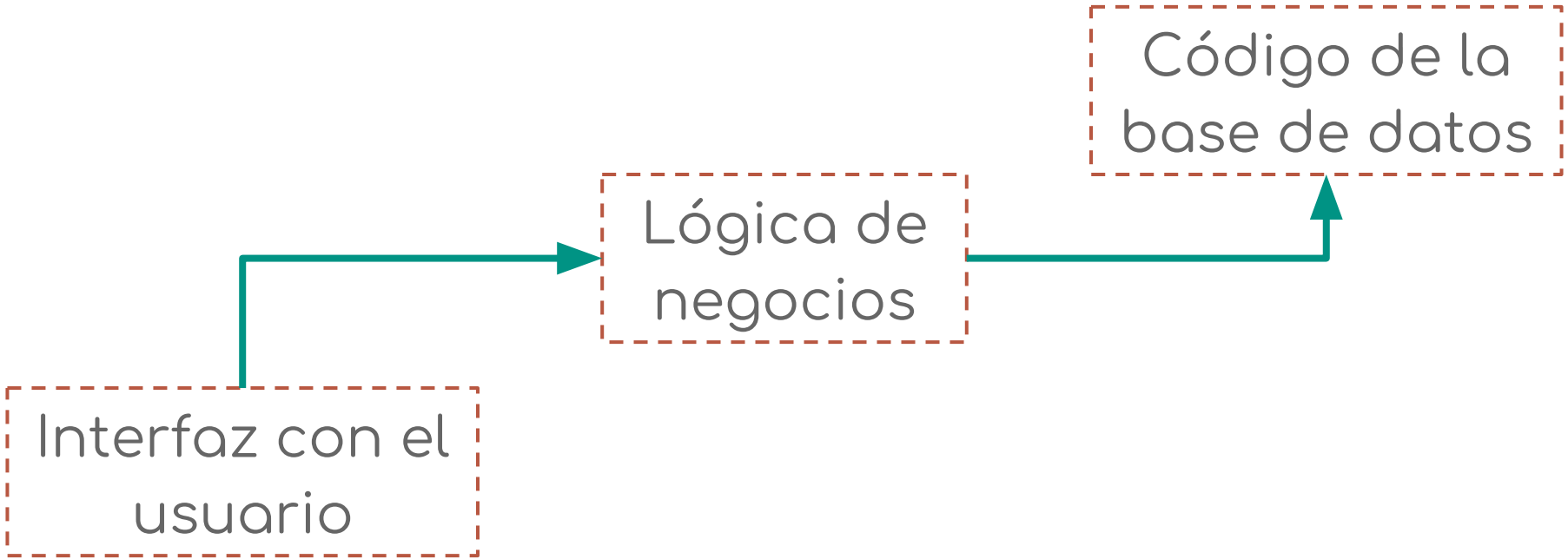
Lenguaje de consultas

Los operadores de SQL se pueden dividir en dos grandes ramas:

- Definición de estructuras de datos **(DDL)** que permite crear y borrar estructuras de datos.
- Manipulación de estructuras de datos **(DML)** que permite añadir datos y modificarlos.

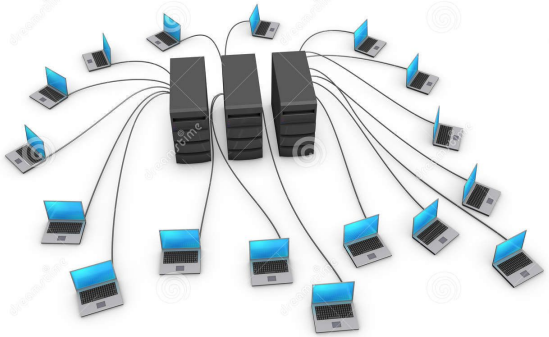


Organización de las aplicaciones que utilizan RDBMSs



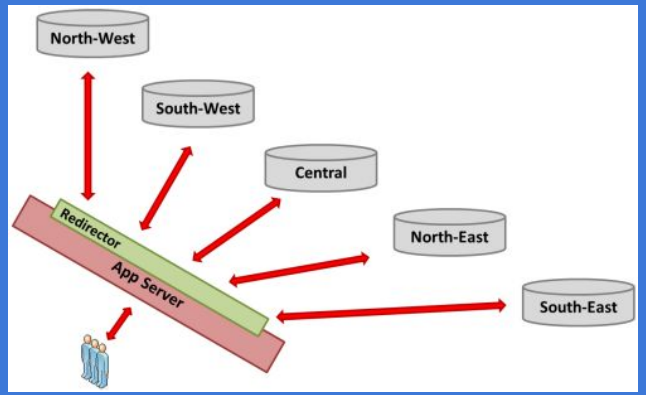
Con el auge del internet es más complicado soportar grandes números de usuarios. Asimismo, se busca conseguir grandes volúmenes de operaciones para leer y escribir, con tiempo de respuestas rápidos y disponibilidad alta.

Estas operaciones son **costosas** y tienen límites con respecto a su c**apacidad de memoria** y el **número de servidores conectados**.



Motivaciones para utilizar bases de datos Not Just / NoSQL

Nivel de escala



Habilidad para satisfacer las necesidades eficientemente para diversas cargas de trabajo.

Los NoSQL están diseñados para utilizar servidores disponibles en un grupo de computadoras con la intervención mínima de los diseñadores.

Motivaciones para utilizar bases de datos Not Just / NoSQL

Costo

El costo de las licencias de la base de datos es complicado de considerar para las empresas.

Estos costos se solventan al utilizar software de código libre y otras compañías que proveen servicios de soporte.

Motivaciones para utilizar bases de datos Not Just / NoSQL

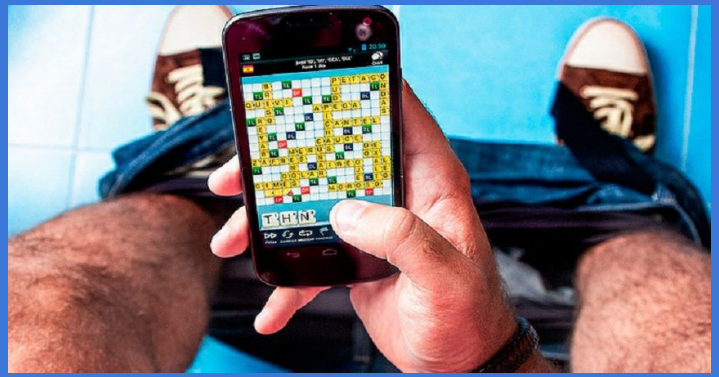
Flexibilidad

Los diseñadores de bases de datos relacionales deben de tomar en cuenta todas las columnas requeridas en las tablas en el planteamiento de las tablas

Contrario a las bases de datos relacionales, las NoSQL no requieren una estructura fija de tablas: se consideran **dinámicas**.

Motivaciones para utilizar bases de datos Not Just / NoSQL

Disponibilidad



Se espera que las páginas web y las aplicaciones estén disponibles en cualquier momento que se deseen ocupar.

Los NoQSL están diseñados para aprovechar **múltiples servidores** a bajo costo, a eso se le conoce como **sistemas distribuidos**.

Variedad de bases de datos NoSQL

- Bases de datos de **valores de llave**: manejan un modelo en el que los datos se asocian con sus identificadores o llaves.
- Bases de datos de **documentos**: también se utilizan identificadores, pero los documentos son colecciones de datos en estructuras flexibles.
- Bases de datos de **familias de columnas**: son similares a las relacionales porque se organizan los datos en colecciones de columnas, pero no se pueden unir tablas entre sí.
- Bases de datos de **grafos**: simplifican el modelado de objetos y las relaciones entre los mismos.

Manejo de datos en sistemas distribuidos

Almacenamiento de datos continuos

Los datos deben almacenarse para que no se pierdan si el servidor se apaga.

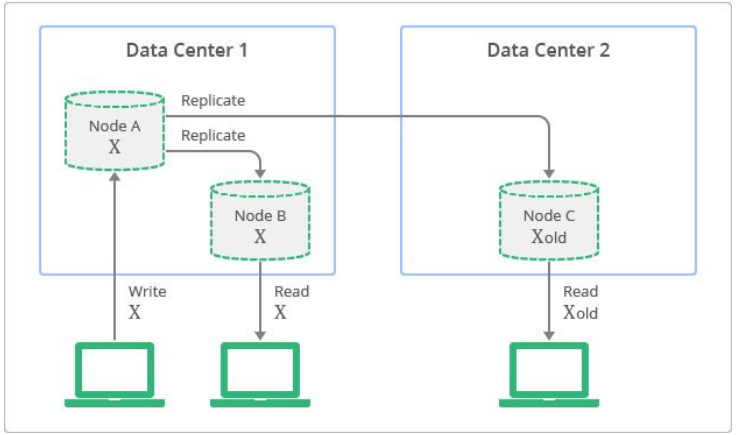
Asimismo, los datos deben de poder recuperarse desde el inicio del archivo hasta su término mediante **índices**.

Manejo de datos en sistemas distribuidos

Mantenimiento de consistencia de datos

Si la operación de leer o escribir datos en la base de datos no es precisa, la base de datos no sirve.

Deben de poder soportar procedimientos múltiples tratados como una operación única.

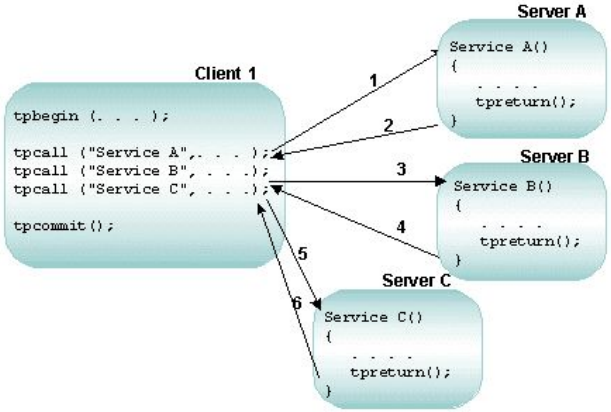


Manejo de datos en sistemas distribuidos

Asegurar disponibilidad de datos

Al trabajar con más de un servidor se asegura que en caso de que alguno falle, la base de datos no se pierda.

Commit de dos fases sucede cuando cualquier cambio en la base de datos primaria se refleja en las demás.



Disponibilidad y consistencia en bases de datos distribuidas

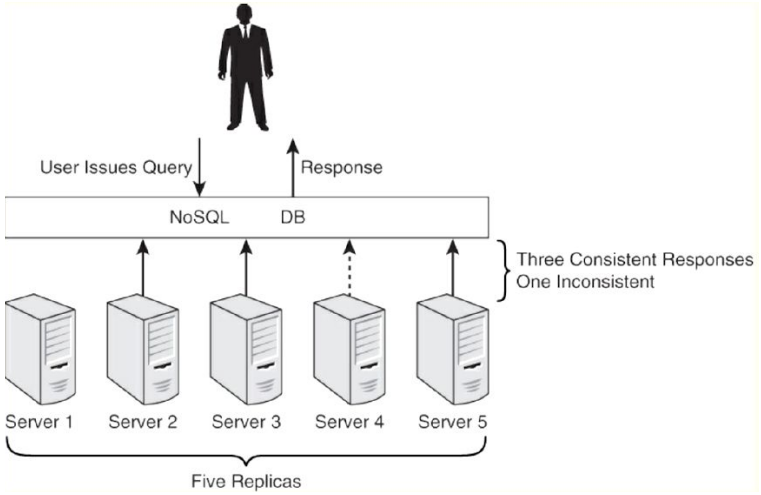
La **consistencia** se refiere al mantenimiento de una única y lógica vista de datos, así como el estado de la copia de datos en sistemas distribuidos.

Se tolera la inconsistencia de datos en un periodo muy corto para sistemas que no requieran tanta seguridad.

Las bases de datos NoSQL implementan la consistencia **eventual**: hay un periodo de tiempo en el que las bases de datos tienen diferentes valores, pero se emparejan.

Un ***quórum*** es el número de servidores que deben responder a una operación de leer/escribir para que ésta se considere completa. Éste afecta tanto el tiempo de respuesta como la consistencia.

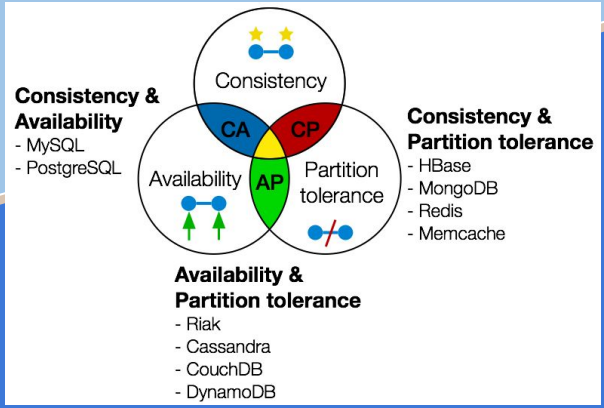
La **durabilidad** es el mantenimiento de una copia de los datos por largos periodos de tiempo



Balance de tiempo de respuesta, consistencia y durabilidad

Teorema de Brewer (CAP) :

Consistency Availability Partitioning



Consistencia: copias consistentes en múltiples servidores.

Disponibilidad: provee respuestas a una consulta.

Protección contra la división: si algún servidor falla, los demás seguirán disponibles.

Una base de datos distributiva no puede tener los tres al mismo tiempo.

Bibliografía

Sullivan, D. (2015). ***NoSQL for Mere Mortals.*** Boston: Addison-Wesley.