Upgrade hub

NoSQL
(Not Only SQL)

Introducción a las BBDD noSQL

La forma en que las aplicaciones web tratan los datos, ha cambiado de forma significativa durante la última década.

Cada vez se recopilan más datos y cada vez son más los usuarios que acceden a estos datos al mismo tiempo.

Esto significa que la escalabilidad y el rendimiento se han convertido en auténticos retos para las bases de datos relacionales basadas en esquemas.

Introducción a las BBDD noSQL

Una fuente ingente de datos es internet. Incluso las estimaciones más conservadoras asumen que la web contiene actualmente más de 1 Yotta-byte en total: 1.000.000.000.000.000.000.000.000 Bytes!

Por eso cuando las grandes compañías de internet como Google se encontraron con el problema de indexar una cantidad ingente de páginas que además eran modificadas continuamente, los "grandes números" de las bases de datos relacionales se les quedaron muy cortos. Incluso compañias como Twitter se encuentran con alrededor de **500 millones de tweets nuevos al día**. Mensajes que además no son homogéneos ya que pueden incluir contenido multimedia. Este es el mundo Big Data.

Introducción a las BBDD noSQL

Las limitaciones del modelo relacional ha venido desde dos fuentes:

1.- Como hemos visto al hablar de Big Data, las aplicaciones modernas necesitan procesar gran cantidad de datos a gran velocidad. Para colmo puede tratarse de datos heterogéneos, es decir con estructura diferente. El modelo relacional no fue pensado para este contexto: trata de adaptarse, pero no es lo suyo.

Introducción a las BBDD noSQL

2.- Cambios en el hardware

- Aumento del nivel de paralelismo: ordenadores con más núcleos
- Aparición de la llamada "commodity computing era"
- La nube (Cloud)

El primer punto es el "problema", el segundo abre nuevas posibilidades que en particular proporcionan nuevos modos de almacenar y gestionar grandes cantidades de datos. Es decir estas nuevas posibilidades para tener bases de datos escalables sin gran coste. Se trata del **escalado horizontal** o "scale-out" que propone añadir más y más nodos (equipos) según aumentan la cantidad de datos va creciendo

Introducción a las BBDD noSQL

La ventaja de el escalado horizontal es sobre todo el precio: salen más baratos 10 servidores de bajo coste que uno 10 veces más potentes (escalado vertical). Pero hay que dejar claro que el escalado horizontal también tiene sus problemas:

- Cuantos más equipos, más posibilidades de que uno falle en cualquier momento. Esto obliga a replicar la información.
- Además en el clúster todos los equipos deben estar coordinados, esto complica el software y a menudo significa pérdida de tiempo en transferencia de datos entre nodos.

Introducción a las BBDD noSQL

En primer lugar, había tipos de bases de datos NoSQL (de origen cerrado), desarrolladas por grandes empresas para satisfacer sus necesidades específicas, como BigTable de Google, que se cree es el primer sistema NoSQL y DynamoDB de Amazon.

El éxito de estos sistemas patentados, inició el desarrollo de varios sistemas de bases de datos de código abierto y de propietarios similares siendo los más populares Hypertable, Cassandra, MongoDB, DynamoDB, HBase y Redis

Introducción a las BBDD noSQL

Los sistemas de bases de datos NoSQL crecieron con las principales redes sociales, como Google, Amazon, Twitter y Facebook.

Con el crecimiento de la web en tiempo real existía una necesidad de proporcionar información procesada a partir de grandes volúmenes de datos que tenían unas estructuras horizontales más o menos similares. Estas compañías se dieron cuenta de que el rendimiento y sus propiedades de tiempo real eran más importantes que la coherencia, en la que las bases de datos relacionales tradicionales dedicaban una gran cantidad de tiempo de proceso

Introducción a las BBDD noSQL

SQL → lenguaje estructurado de consultas…. (MySQL, ORACLE, PostGreSQL)

clave: un Id, un DNI, … Es un dato que nos identifica quién es, y nos facilita su relaciones.

BBDD no Relacionales: colecciones de datos que se parecen entre sí, pero que no son necesariamente iguales.

clave: podemos tener, pero no la necesitamos.

Introducción a las BBDD noSQL

Bajo la denominación de NoSQL se engloba un conjunto de bases de datos que han sido construidas para soportar entornos de aplicación en donde las bases de datos relacionales no constituyen la mejor la solución.

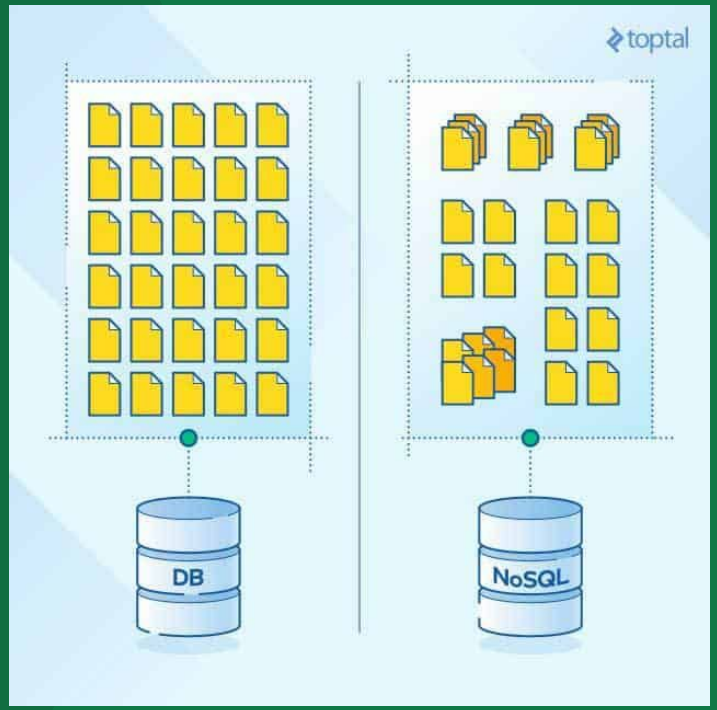
Estos entornos de aplicación tienen dos características principales:

Introducción a las BBDD noSQL

- En primer lugar, se trata de entornos que necesitan disponer de esquemas de datos más flexibles, es decir, se trata de entornos que no se ajustan a la estructuración rígida en forma tabla que ofrecen las bases de datos relacionales.

- Y en segundo lugar, estas BD se utilizan en entornos de aplicación altamente distribuidos que necesitan estar siempre operativos (en línea) y que necesitan gestionar un volumen importante de datos, posiblemente heterogéneos.

Introducción a las BBDD noSQL



Una diferencia clave entre las bases de datos de NoSQL y las bases de datos relacionales tradicionales, es el hecho de que NoSQL es una forma de almacenamiento estructurado pero no fijo.

NoSQL no tiene una estructura de tabla fija como las que se encuentran en las bases de datos relacionales.

Introducción a las BBDD noSQL

Las bases de datos NoSQL están altamente optimizadas para las operaciones recuperar y agregar, y normalmente no ofrecen mucho más que la funcionalidad de almacenar los registros (p.ej. almacenamiento clave-valor).

La pérdida de flexibilidad en tiempo de ejecución, comparado con los sistemas SQL clásicos, se ve compensada por ganancias significativas en escalabilidad y rendimiento cuando se trata con ciertos modelos de datos.

Introducción a las BBDD noSQL

Las implementaciones típicas de SGBDR se han afinado o bien para una cantidad pequeña pero frecuente de lecturas y escrituras o para un gran conjunto de transacciones que tiene pocos accesos de escritura.
Por otro lado NoSQL puede servir gran cantidad de carga de lecturas y escrituras.

50 TB de la búsqueda de la bandeja de entrada de Facebook

Introducción a las BBDD noSQL

Bastantes sistemas NoSQL emplean una arquitectura distribuida, manteniendo los datos de forma redundante en varios servidores.

De esta forma, el sistema puede realmente escalar añadiendo más servidores, y el fallo en un servidor puede ser tolerado.

Ventajas NoSQL

- A diferencia de las bases de datos relacionales, las bases de datos NoSQL están basadas en key-value pairs

- Algunos tipos de almacén de bases de datos NoSQL incluyen por ejemplo el almacenamiento de columnas, de documentos, de key value store, de gráficos, de objetos, de XML , ...

- Las bases de datos NoSQL de código abierto tienen una implementación rentable. Ya que no requieren las tarifas de licencia y pueden ejecutarse en hardware de precio bajo.

Ventajas NoSQL

- La expansión es más fácil y más barata que cuando se trabaja con bases de datos relacionales.

Se realiza un **escalado horizontal** (aumento de máquinas o nodos) y se distribuye la carga por todos los nodos. En lugar de realizarse una escala vertical, más típica en los sistemas de bases de datos relacionales (aumento de capacidad de disco o cambio por un procesador más rápido).

Ventajas NoSQL

- Pueden manejar enormes cantidades de datos.

- No generan cuellos de botella.

-Diferentes DBs NoSQL para diferentes proyectos

Desventajas NoSQL

La mayoría de las bases de datos NoSQL no admiten funciones de fiabilidad, que son soportadas por sistemas de bases de datos relacionales. Estas características de fiabilidad pueden resumirse en: “atomicidad, consistencia, aislamiento y durabilidad.” (ACID)

Las bases de datos NoSQL, que no soportan esas características, ofrecen consistencia para el rendimiento y la escalabilidad.

Desventajas NoSQL

Falta de madurez de la mayoría de los sistemas NoSQL y los posibles problemas de inestabilidad → desconfianza

La falta de experiencia y falta de profesionales

Problemas de compatibilidad.- Las bases de datos relacionales, comparten ciertos estándares, **PERO** las bases de datos NoSQL tienen pocas normas en común. Cada base de datos NoSQL tiene su propia API, las interfaces de consultas son únicas y tienen peculiaridades. Esta falta de normas significa que es imposible cambiar simplemente de un proveedor a otro, por si no quedara satisfecho con el servicio.

Desventajas NoSQL

Con el fin de apoyar las características de fiabilidad y coherencia, los desarrolladores deben implementar su propio código, lo que agrega más complejidad al sistema.

Esto podría limitar el número de aplicaciones en las que podemos confiar para realizar transacciones seguras y confiables, ¿cuáles por ejemplo?

Sistemas de reservas

Sistemas bancarios

Seguridad Social y Admon en general

Desventajas NoSQL

No usan SQL como lenguaje principal de consultas → se necesita un lenguaje de consulta manual, haciendo los procesos mucho más lentos y complejos.

Los datos almacenados no requieren estructuras fijas como tablas, normalmente no soportan operaciones JOIN

Funcionalidades NoSQL vs. BBDDRR

Feature	NoSQL Databases	Relational Databases
Performance	High	Low
Reliability	Poor	Good
Availability	Good	Good
Consistency	Poor	Good
Data Storage	Optimized for huge data	Medium sized to large
Scalability	High	High (but more expensive)

Esta tabla muestra una comparación a nivel de la base de datos, no sobre los diversos sistemas de gestión de bases de datos que implementan ambos modelos. Estos sistemas proporcionan *sus propias técnicas patentadas* para superar los problemas y deficiencias encontradas en el sistema, además de intentar mejorar significativamente el rendimiento y la fiabilidad.

Tipos de almacenamiento de datos NoSQL

Key Value Store: se utiliza una tabla hash en la que una clave única apunta a un elemento.

Todo lo que se necesita para hacer frente a los elementos almacenados en la base de datos: es la clave. Los datos se almacenan en forma de una cadena, JSON.

USO: Aplicaciones que sólo utilizan consulta de datos por un solo valor de la clave

DynamoDB de Amazon

Tipos de almacenamiento de datos NoSQL

Almacenes de documentos: son similares a los almacenes de valores clave, porque no tienen un esquema y se basan en un modelo de nombreCampo:Dato

– “Intuitivo” → representa la información tal cual.

– Manera natural de modelar datos cercana a la programación orientada a objetos

– Flexibles, con esquemas dinámicos

– Reducen la complejidad de acceso a los datos.

Tipos de almacenamiento de datos NoSQL

```
{
        título: “Doctor”,
        Nombre: “Pedro”,
        Edad: 37,
        City: "Madrid",
        PerfilFB:”https://fb.com”,
         Amigos:[ {nombre: “Maria”, edad:28}, {nombre:”Luis”, alias: “ito”}],
        Aficiones:[“leer”,”tennis”, “viajar”]
 }
```

Tipos de almacenamiento de datos NoSQL

En el almacén de documentos, los valores (documentos) proporcionan codificación XML, JSON o BSON (JSON codificado binario) para los datos almacenados.

USO: Se pueden utilizar en diferentes tipos de aplicaciones debido a la flexibilidad que ofrecen

MongoDB y Apache CouchDB

Tipos de almacenamiento de datos NoSQL

Almacenamiento en columnas: en lugar de almacenarse en filas, (BBDDRR).

Un almacén de columnas está compuesto por una o más familias de datos que se agrupan de forma lógica en determinadas columnas en la base de datos. Una clave se utiliza para identificar y señalar a un número de columnas en la base de datos. Cada columna contiene filas de nombres o tuplas, y valores, ordenados y separados por comas.

Tipos de almacenamiento de datos NoSQL

ejemplo simple de una base datos de 4 columnas y 3 filas que contiene los siguientes datos:

ID	apellido	nombre	bono
1	González	Manuel	6000
2	Martínez	Antonio	3000
3	Gutiérrez	Alberto	2000

En un sistema de gestión de base de datos orientado a filas los datos se almacenarán de la siguiente manera: 1, González, Manuel, 6000; 2, Martínez, Antonio, 3000; 3, Gutiérrez, Alberto, 2000;

En un sistema de gestión de base de datos columnar los datos se almacenan de la siguiente manera: 1, 2, 3; González, Martínez, Gutiérrez; Manuel, Antonio, Alberto; 6000, 3000, 2000;

Tipos de almacenamiento de datos NoSQL

Tienen acceso rápido de lectura y escritura a los datos almacenados. En un almacén de columnas, las filas que corresponden a una sola columna se almacenan como una sola entrada de disco, lo cual facilita el acceso durante las operaciones de lectura y escritura.

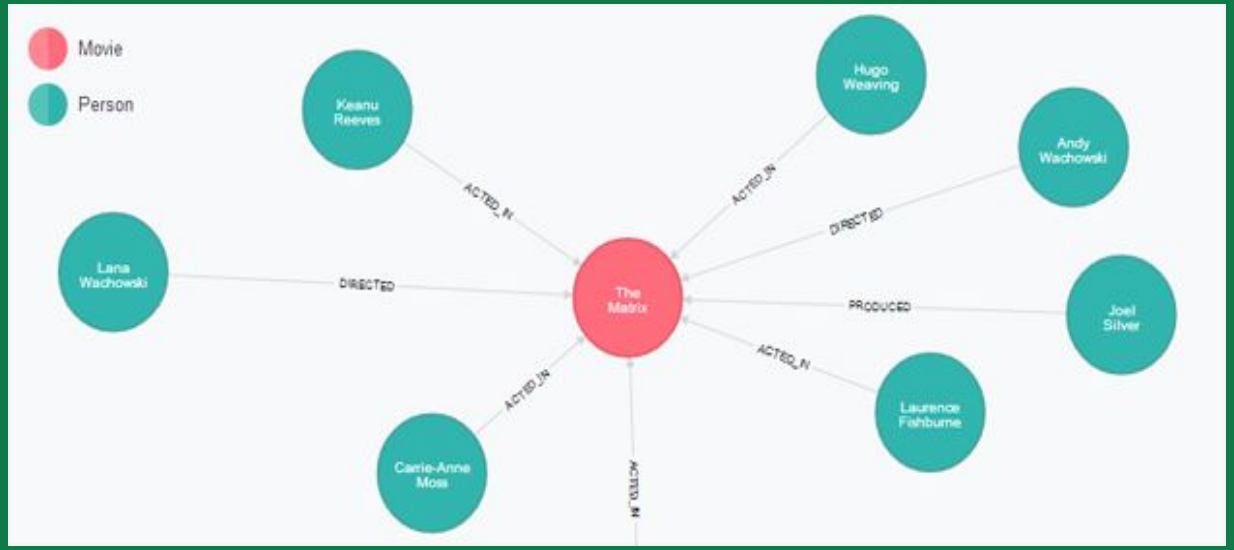
Los datos pueden ser altamente comprimidos. La compresión permite que las operaciones columnares como MIN, MAX, SUM, COUNT y AVG se realicen muy rápidamente.

Como es auto indexable, utiliza menos espacio en disco que un sistema de gestión de base de datos relacional que contenga los mismos datos.

Google BigTable, HBase y Cassandra (desarrollada por FaceBook).

Tipos de almacenamiento de datos NoSQL

Orientada a grafos: se utiliza una estructura de grafos para representar los datos. El gráfico está compuesto por aristas (relaciones) y nodos (información).



Tipos de almacenamiento de datos NoSQL

– Los datos se modelan como un conjunto de relaciones entre elementos específicos.

– Flexibles, atributos y longitud de registros variables

– Permite consultas más amplias y jerárquicas.

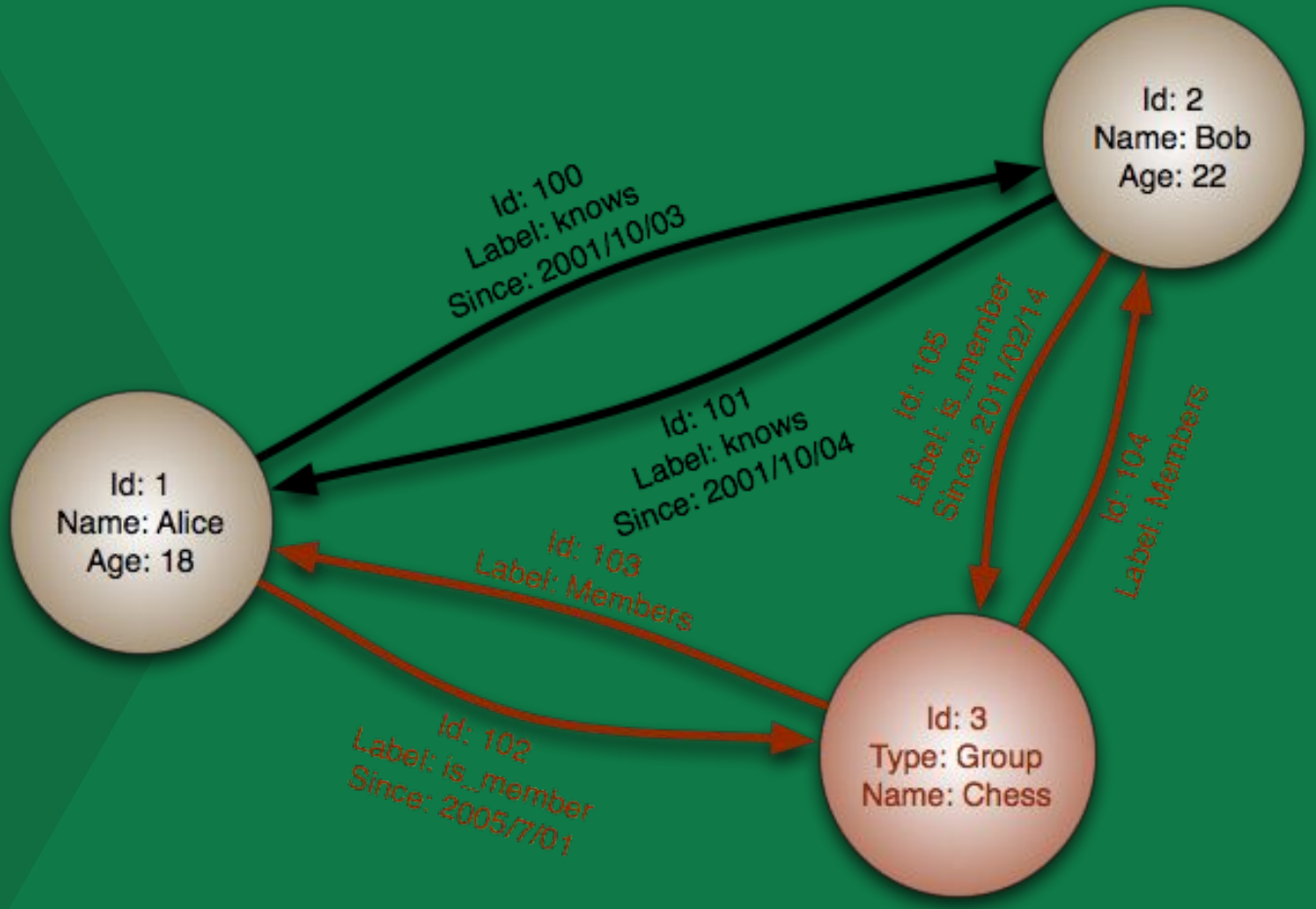
USO: Redes sociales, software de recomendación, geolocalización, topologías de red, ...

InfoGrid, InfiniteGraph y Neo4j

Tipos de almacenamiento de datos NoSQL

Ejemplos de APLICACIÓN de grafos.- permite conectar de forma eficaz a las personas con nuestros productos y servicios, en función de la información personal, sus perfiles en redes sociales y su actividad online reciente. En este sentido, las bases de datos orientadas a grafos son interesantes porque son capaces de conectar personas e intereses. Con esa información, una empresa puede ajustar sus productos y servicios a su público objetivo y personalizar las recomendación en función de los perfiles. Eso es lo que permite que se aumente la precisión comercial y el compromiso del cliente.

Cómo se relacionan las personas que trabajan en una empresa y el conocimiento que poseen, la relación que existe entre lectores de blogs y el contenido de los artículos que leen o incluso cómo interactúan los clientes con los productos de una compañía. Al igual que en ámbitos más científicos, entendiendo las relaciones entre los genes del ADN o entre las partículas que forman la materia.


Id: 2
Name: Bob
Age: 22
Id: 100
Label: knows
Since: 2001/10/03
Id: 101
Label: knows
Since: 2001/10/04
Id: 105
Label: is_member
Since: 2011/02/14
Id: 104
Label: Members
Id: 1
Name: Alice
Age: 18
Id: 103
Label: Members
Id: 102
Label: is_member
Since: 2005/7/01
Id: 3
Type: Group
Name: Chess

Modelo de consultas de datos NoSQL

Cada aplicación tiene unos requisitos distintos. En algunos casos, es suficiente tener un modelo de consultas básico en el que la aplicación acceda a los registros basándose en una clave primaria.

Sin embargo, en la mayoría de aplicaciones, es necesario poder ejecutar consultas basándose en varios valores distintos para cada uno de los registros.

Modelo de consultas de datos NoSQL

- **Bases de datos orientadas a documento**: Este tipo de bases de datos proporcionan la posibilidad de ejecutar consultas en base a cualquier tipo de campo dentro del documento.
  a. También permiten hacer consultas basadas en índices secundarios. Esto permite actualizar registros incluyendo uno o más campos dentro del documento.

Modelo de consultas de datos NoSQL

- **Bases de datos orientadas a grafo:** El almacenamiento en este tipo de bases de datos está optimizado para ejecutar la navegación entre nodos (traversals). Por este motivo, las bases de datos orientadas a grafo son eficientes para realizar consultas en las que existan relaciones de proximidad entre datos, y no para ejecutar consultas globales.

Modelo de consultas de datos NoSQL

- **Bases de datos clave-valor y orientadas a columna:** Este tipo de sistemas permiten obtener y actualizar datos en base a una clave primaria. Las bases de datos clave-valor y orientadas a columna ofrecen un modelo de consultas limitado que puede imponer costes de desarrollo y requisitos a nivel de aplicación para ofrecer un modelo de consultas avanzado. Un ejemplo de esto son los índices, que deben ser gestionados por el propio usuario.

Modelo de consistencia de datos NoSQL

Los sistemas NoSQL típicamente mantienen varias copias de los datos para proporcionar escalabilidad y disponibilidad. Estos pueden utilizar dos tipos de consistencia distinta entre los datos de sus múltiples copias: consistencia y consistencia eventual.

En los **sistemas consistentes** se garantiza que las escrituras sean inmediatamente visibles para las consultas posteriores. Este tipos de sistemas son especialmente útiles para aplicaciones en las que se hace indispensable que los datos sean siempre coherentes. Los sistemas consistentes proporcionan ventajas en las escrituras, aunque, por contra, las lecturas y las actualizaciones son más complejas.

Modelo de consistencia de datos NoSQL

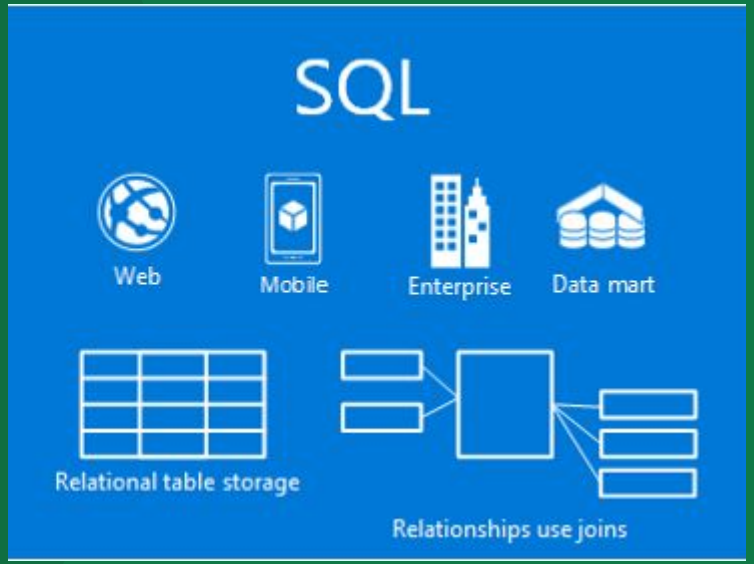
En los **sistemas eventualmente consistentes**, existe un periodo durante el que no todas las copias de los datos están sincronizadas. El hecho de no tener que comprobar la consistencia de los datos en cada una de las operaciones supone una mejora importante en el rendimiento y disponibilidad del sistema, aunque para ello se sacrifique la coherencia de los datos.

Estos tipos de sistemas son especialmente útiles para datos que NO cambian a menudo, como archivos históricos o logs.

Modelo de consistencia de datos NoSQL

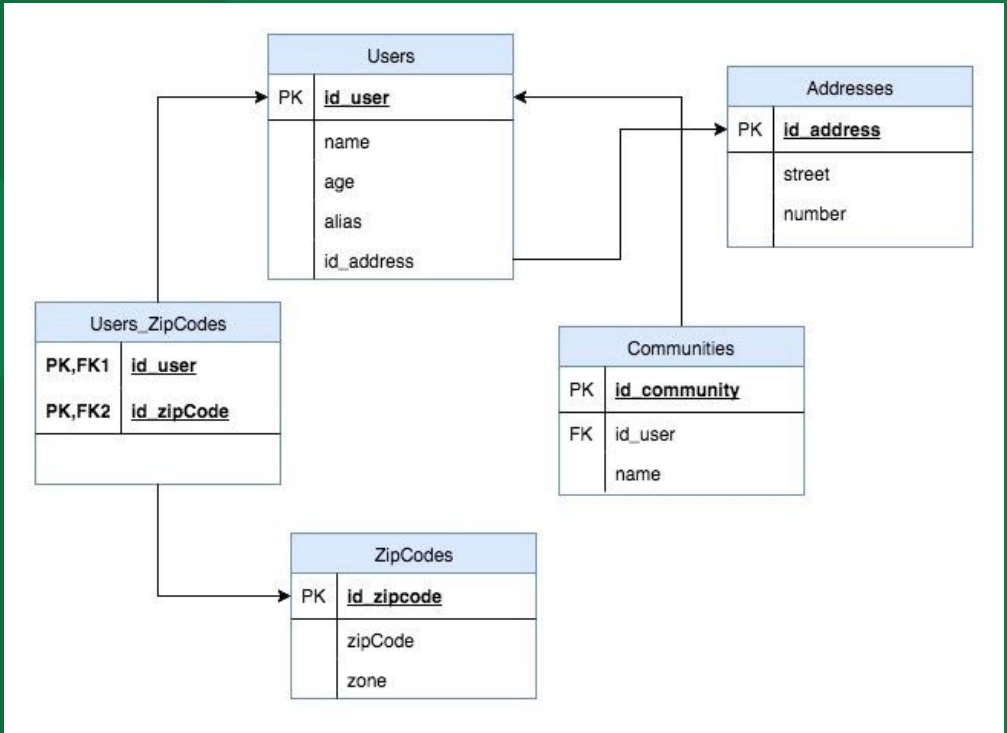
Las bases de datos orientadas a documento o grafo existentes pueden ser consistentes o eventualmente consistentes, mientras que las bases de datos clave-valor y orientadas a columna son típicamente eventualmente consistentes.

SQL → NoSQL


SQL
Web
Mobile
Enterprise
Data mart
Relational table storage
Relationships use joins


NoSQL
Gaming
Social
IoT
Web
Mobile
Enterprise
$
Key/value store
{ }
Document database
Column family store

SQL → NoSQL



Users <collection>

```
{
 name  : "Marcela Sena",
 age   : 29,
 alias : "MarceStarlet",
 memberOf : ["TechWo", "JavaGDL"],
 address  : { street : "Main Boulevard", number : 234 },
 zipCodes : [
   { zipCode : 1234, zone: "Guadalajara" },
   { zipCode : 4321, zone: "arajaladauG" }
 ]
}
```

SQL → NoSQL

La diferencia sustancial sería la abstracción completa de todos los datos definidos en un solo documento y no en distintas tablas relacionadas.

Las restricciones y la existencia de “ids” de otras tablas como parte de nuestros registros ya no son relevantes pues lo que nos interesa es la información como tal.