MEDAC
Instituto Oficial de Formación Profesional

ACCESO A DATOS

TÉCNICO EN DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Introducción a las bases de datos No-SQL

11

ÍNDICE

/ 1. Introducción y contextualización práctica 3

/ 2. Características de las bases de datos No-SQL 4

/ 3. Fundamentos de las bases de datos No-SQL 5

/ 4. Beneficios de las bases de datos No-SQL I 5

/ 5. Caso práctico 1: “REDIS NoSQL Database” 6

/ 6. Beneficios de las bases de datos No-SQL II 7

/ 7. Beneficios de las bases de datos No-SQL III 8

/ 8. Tipos de bases de datos NoSQL 8

/ 9. Introducción a Big Data 9

/ 10. Tipos de Big Data 10

/ 11. Caso práctico 2: “Tipos estructuras Big Data” 11

/ 12. Resumen y resolución del caso práctico de la unidad 11

/ 13. Webgrafía 12

OBJETIVOS

Aprender los fundamentos de las bases de datos No-SQL

Introducirnos en el concepto genérico de Big Data

/ 1. Introducción y contextualización práctica

En esta unidad nos centraremos en el mundo de las bases de datos No-SQL, veremos cuáles son sus diferencias respecto a los principales sistemas gestores de bases de datos relacionales, y por supuesto estudiaremos cuáles son sus características principales.

Dedicaremos un buen apartado para estudiar y evaluar los beneficios reales que presentan estas bases de datos en el mundo empresarial

Por último, realizaremos una breve introducción al Big Data, definiremos ciertos conceptos básicos que rodean este mundo y veremos para qué son usadas este tipo de almacenes de grandes datos.

Planteamiento del caso práctico inicial

A continuación, vamos a plantear un caso práctico a través del cual podremos aproximarnos de forma práctica a la teoría de este tema.

Escucha el siguiente audio donde planteamos la contextualización práctica de este tema, encontrarás su resolución en el apartado Resumen y Resolución del caso práctico.

Fig. 1. Bases de datos No-SQL.

Audio intro. "Top 5 NoSQL"
https://bit.ly/3lMnkju

/ 2. Características de las bases de datos No-SQL

Tradicionalmente, la industria del *software* como ya bien sabemos, usa las bases de datos relacionales para almacenar y manejar datos de forma persistente. No solo bases de datos SQL y No-SQL son las que han emergido en este pasado reciente.

No-SQL se refiere a todas las bases de datos y almacenes de datos que no están basadas en el sistema tradicional de bases de datos relacionales en sí, o no están basadas en sus principios.

Suele relacionarse con **grandes conjuntos de datos a los que se accede y se manipulan a escala web.** El concepto No-SQL no representa un producto único o una única tecnología, representa un grupo de productos y un conjunto relacionado de conceptos para el almacenamiento y su administración. Fue también un *hashtag* escogido para una gran reunión técnica para discutir las nuevas bases de datos.

Las **bases de datos No-SQL** tienen una serie de **características comunes** como:

1. Modelo de datos no relacional.
2. Funcionan bien en *clusters*.
3. Suelen ser en su mayoría de código abierto.
4. Su construcción está orientada para las aplicaciones web de nueva generación.
5. Son bases de datos con ausencia de esquema (*schemaless*)

Dentro de las diferentes características mencionadas aclararemos algunas de ellas que puedan dar lugar a dudas:

- **Funciona bien en modo *cluster*:** Podemos definir básicamente este concepto como un conjunto de máquinas implementadas como servidores de procesamiento paralelo. Funcionan entre sí como un único recurso. Y como bien sabemos a cada elemento o máquina lo llamamos nodo.

Audio 1. "Cluster"
https://bit.ly/3kBrYiU

- **Bases de datos con ausencia de esquema o *schemaless*:** Es una característica muy flexible, se puede almacenar información no uniforme y se facilita así la evolución.

Fig 2. Bases de datos NoSQL.

/ 3. Fundamentos de las bases de datos No-SQL

Con la explosión del social-media y el contenido manejado por el usuario, han incrementado el volumen y el tipo de datos que se producen, se manejan, analizan y se archivan, por lo que han provocado la **rápida eclosión de las bases de datos No-SQL.** Además, a esta cantidad ingente de datos, hay que sumarle las aportaciones de las nuevas fuentes de información como sensores, sistemas globales de posicionamiento o GPS, rastreadores, y otros tipos de sistemas que monitorean grandes cantidades de información de forma regular.

Estos grandes paquetes de información, han introducido nuevos retos y oportunidades para el almacenamiento de información. Además, los datos son cada vez más semiestructurados y escasos. Esto quiere decir que **las bases de datos relacionales son examinadas y requieren una definición de esquema inicial y referencias relacionales.**

Para resolver el problema relacionado con los grandes volúmenes de información y los datos semi-estructurados, surge nuevas clases de bases de datos dentro de la familia No-SQL. Este tipo de base de datos consisten en el almacenamiento **basado en columnas, clave/valor y de documentos**, que estudiaremos más adelante.

A continuación, veremos las diferencias más notables entre las bases de datos relacionales y las No-SQL:

BASES DE DATOS RELACIONALES	BASES DE DATOS NOSQL
La información está almacenada en un modelo relacional con filas y columnas. Una fila contiene información sobre un elemento mientras que las columnas contienen información específica.	La información está almacenada en un cliente o en una base de datos diferente con modelos de datos distintos.
Sigue un esquema fijo. Las columnas son definidas y establecidas antes de la entrada de datos. Además, cada fila contiene información de cada columna.	Sigue un esquema dinámico, puedes añadir columnas en cualquier momento
A favor del escalado vertical.	Facilita el escalado horizontal. Se puede escalar a través de servidores múltiples. Los servidores múltiples tienen la ventaja del precio en relación al Hardware que se va añadiendo comparado con el escalado vertical.
Atomicidad, consistencia, aislamiento y durabilidad. (ACID)	No está a favor de los principios ACID.

Tabla 1. Diferencias BBDD relacionales y NoSQL.

/ 4. Beneficios de las bases de datos No-SQL I

A continuación, en las próximas diapositivas, veremos en profundidad los diferentes beneficios técnicos de una solución No-SQL a nivel empresarial:

a) Capacidad de fuente de datos primaria y de análisis

El primer criterio de una clase empresarial NoSQL es que debe servir como **fuente principal de datos que recibe información de distintas aplicaciones de negocio.** También debe actuar como segunda fuente de información o análisis las aplicaciones de *"business intelligence"*. Desde el punto de vista de negocio, este tipo de bases de datos deben de ser capaces de integrarse de una forma rápida a todos los tipos de datos estructurados, semiestructurados o sin estructura. Además, deben ser capaces de ejecutar consultas de alto rendimiento.

b) Capacidad Big Data

Las bases de datos NoSQL no se limitan a trabajar con Big Data. Una base de datos de este tipo de clase empresarial, puede escalar para **administrar grandes volúmenes de datos desde Terabytes hasta Petabytes.** Además de almacenar grandes volúmenes de datos, ofrece un alto rendimiento para la velocidad, variedad y complejidad de los datos.

c) Disponibilidad continua

Para que una base de datos sea considerada de clase empresarial, debe ofrecer disponibilidad continua, sin un solo punto de fallo.

Además, en lugar de proporcionar la función de disponibilidad continua fuera del *software*, la solución NoSQL ofrece una disponibilidad continúa integrada, que debe incluir las siguientes **características clave:**

1. Todos los nodos de un clúster deben poder atender solicitudes de lectura incluso si algunas máquinas no funcionan.
2. Debe ser capaz de replicar y segregar datos fácilmente entre diferentes partes físicas en un centro de datos. Esto evitará cortes de *hardware*.
3. Debe poder admitir diseños de distribución de datos que sean centros de datos múltiples ya sea en instalaciones físicas o en la nube.

Fig 3. Big Data.

Video 1. "Desventajas NoSQL"
https://bit.ly/38WFTOi

/ 5. Caso práctico 1: "REDIS NoSQL Database"

Planteamiento: Se requiere realizar una conexión directa con una base de datos No-SQL Redis. Para ello imaginemos que nuestra aplicación está desarrollada en lenguaje Java y tenemos que implementar la parte de la conexión a base de datos.

Nudo: En este caso nos disponemos a realizar la construcción de la capa de acceso a datos de la aplicación

Desenlace: Los pasos a seguir serán los siguientes:

1. Agregación de dependencias a nuestro fichero pom.xml del proyecto, para cargar la librería necesaria y así acceder a la base de datos Redis.

```
<dependency>
    <groupId>redis.clients</groupId>
    <artifactId>jedis</artifactId>
    <version>2.9.0</version>
</dependency>
```

Código 1. Dependencia Redis.

Como vemos en la imagen anterior de esta forma agregaremos la versión específica que nos realizará la conexión con la base de datos.

2. Implementación de la nueva capa usando las librerías previamente importadas en Maven.

```
import redis.clients.jedis.Jedis;

public class RedisConnector {
   public static void main(String[] args) {
      Jedis cliente = new Jedis("localhost");
      client.set("clave", "valor");
      String value = client.get("clave");
      System.out.println(value);
      client.close();
   }
}
```

Código 2. Integración Redis.

De esta forma podremos instanciar tanto el cliente, como establecer valores en la base de datos al igual que obtenerlos con los diferentes métodos que observamos en el código anterior.

/ 6. Beneficios de las bases de datos No-SQL II

Continuaremos con algunos beneficios de las bases de datos NoSQL:

a) Capacidad de tener múltiples centros de datos

Por lo general, en un entorno profesional, las empresas poseen bases de datos altamente distribuidas que se encuentran en varios **centros de datos y ubicaciones geográficas distintas.**

La replicación de datos es una característica que ofrecen todas las bases de datos relacionales. Sin embargo, ninguna puede ofrecer un modelo simple de distribución de datos entre varios centros de datos sin causar problemas de rendimiento.

Una buena solución empresarial NoSQL debe admitir la implementación de varios centros de datos y debe proporcionar una opción configurable para mantener un equilibrio entre el rendimiento y la coherencia.

b) Fácil replicación independientemente de la ubicación

Para evitar que la pérdida de datos afecte a una aplicación, una buena solución NoSQL proporciona una gran capacidad de replicación. Estos incluyen una capacidad de lectura y escritura en cualquier lugar con compatibilidad total y dependencia de ubicación.

Fig. 4. Clúster.

Esto significa que se pueden **escribir datos en cualquier nodo de un clúster, hacer que se repliquen en otros nodos,** y ponerlos a disposición de todos los usuarios independientemente de su ubicación.

Además, la capacidad de escritura en cualquier nodo debe garantizar que los datos estén seguros en caso de un corte de suministro eléctrico o cualquier otro tipo de incidente.

/ 7. Beneficios de las bases de datos No-SQL III

a) Sin capa de almacenamiento en caché separada

Una buena solución NoSQL es capaz de utilizar múltiples nodos y distribuir datos entre todos los nodos adjuntos.

Una vez aclarado esto, podemos decir que **no requiere una capa de almacenamiento en caché específica para almacenar datos.** Las memorias caché de todos los nodos pueden almacenar datos para una entrada y salida rápida o para un acceso entrada/salida. La base de datos NoSQL elimina el problema de sincronizar los datos de la caché con la base de datos persistente. Por lo tanto, admite una escalabilidad simple con menos problemas de administración

b) Lista para la nube

Dado que la adaptación de la infraestructura de la nube aumenta día a día, una solución NoSQL de nivel empresarial debe estar preparada para la nube.

Un clúster de base de datos NoSQL debe poder funcionar en una configuración de la nube, como Amazon EC2, y también debe poder extender y reducir un clúster cuando sea necesario. También debe admitir una solución híbrida en la que parte de la base de datos se aloje dentro de las instalaciones de la empresa y otra parte se aloje en la nube.

c) Alto rendimiento con escalabilidad lineal

Una base de datos de este tipo **puede mejorar el rendimiento agregando nodos a un clúster.** Normalmente, el rendimiento de otros sistemas de bases de datos puede disminuir cuando se agregan los nodos adicionales, sin embargo, una buena solución NoSQL aumenta el rendimiento de las operaciones de lectura y escritura cuando se agregan nodos adicionales. Estas ganancias de rendimiento son de naturaleza lineal.

Finalmente, algunos beneficios más de las bases de datos No-SQL serían:

- Soporte de esquema flexible.
- Admite lenguajes y plataformas clave para desarrolladores.
- Fácil de implementar, mantener y extender.
- Comunidad de código abierto.

Fig. 5. Beneficios No-SQL.

/ 8. Tipos de bases de datos NoSQL

Existen cuatro tipos de bases de datos NoSql en el mercado:

1. **Bases de datos clave/valor:** La base de datos clave valor, es realmente una tabla *hash* de claves y valores. Algunos de los ejemplos son: Riak, Tokyo Cabinet, servidor Redis, Memcached y Scalaris.

Fig. 6. NOSQL clave valor.

2. **Bases de datos basadas en documentos:** Este tipo de bases de datos basadas en documentos almacena documentos compuestos por elementos etiquetados. Algunos ejemplos son: MongoDB, CouchDB, OrientDB y RavenDB.

Fig. 7. NOSQL documentos.

3. **Bases de datos basadas en columnas:** Cada bloque de almacenamiento contiene datos de una sola columna. Ejemplos: BibTable, Cassandra, Hbase e Hypertable.

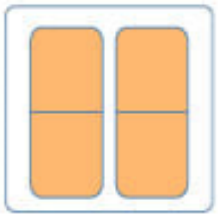

Fig. 8. NOSQL columnas.

4. **Bases de datos basadas en gráficos:** Una base de datos basada en gráficos es una base de datos de red que utiliza nodos para representar y almacenar datos. Algunos ejemplos son: Neo4J, InfoGrid, Infinite Graph y FlockDB. La disponibilidad de opciones en las bases de datos NoSQL tiene sus propias ventajas (permite elegir un diseño de acuerdo a los requisitos de su sistema) e inconvenientes (aun ajustando el producto a los recursos del sistema no siempre funcionará correctamente).

Fig. 9. Gráficos.

/ 9. Introducción a Big Data

En primer lugar, intentaremos definir qué son los datos (DATA). Realmente son las **cantidades, los caracteres o símbolos con los que se realizan operaciones mediante una máquina.** Pueden almacenarse y **transmitirse en forma de señales eléctricas y registrarse en soportes de almacenamiento magnéticos, ópticos o mecánicos.**

Una vez conocido el concepto de datos veremos *Big Data*.

Big Data son realmente datos, pero con la diferencia que su tamaño es muy grande. Con *Big Data* definimos en realidad una colección de datos que es gigante y, sin embargo, crece cada vez más con el tiempo. Los datos que manejamos son tan grandes y tan tediosos de administrar que no se podía manejarlos con las herramientas que había hasta la fecha.

¿Por qué las bases de datos *Big Data* son sistemas diferentes?

Para poder trabajar con *Big Data* necesitaríamos en principio el mismo conocimiento como si fuésemos a trabajar con datos de menor volumen. Pero algunas características como la escalabilidad a gran volumen, la rapidez de mover información y de procesarla, y las propiedades de algunos de los datos que se tratarán en el proceso, presentan nuevos desafíos importantes a la hora de diseñar soluciones. El objetivo de la mayoría de los sistemas de Big Data es sacar a la luz conocimientos y conexiones de grandes volúmenes de datos heterogéneos que no serían posibles con métodos convencionales.

Fig. 10. Flujo Big data.

Vídeo 2. "Aplicaciones Big Data"
https://bit.ly/3kGx7q1

/ 10. Tipos de Big Data

Podemos encontrar *Big Data* de tres formas diferentes:

a) Estructurado

Definimos datos estructurados como toda aquella **información que pueda ser almacenada, se pueda acceder y se accede, de forma fija.**

Poco a poco conforme ha ido pasando el tiempo, los sistemas de almacenamiento han sido mejorados con distintas técnicas innovadoras que mejoran el trabajo con este tipo de datos. Pero en la actualidad existe un problema real, los datos crecen en tal medida que alcanzan tamaños como el ***zettabyte.***

10 elevado a 21 bytes forman 1 *zettabyte* o lo que es lo mismo un billón de terabytes forman 1 *zettabyte*.

Viendo estas cifras podemos entender fácilmente por qué se le da el nombre de *Big Data*.

b) Datos no estructurados

Cualquier **dato cuya forma o estructura sea desconocida** se clasificará como datos no estructurados. Además de que el tamaño es enorme, los datos no estructurados plantean múltiples desafíos si nos referimos al procesamiento de los mismos. Hoy en día muchas compañías disponen de sistemas no estructurados donde mezclan colecciones que contienen imágenes de texto, números, direcciones, ficheros media, etc. Esto puede suponer un problema real si estos datos no son procesados y si no se sabe sacar partido de ellos.

Fig. 11. Datos semiestructurados.

c) Datos semi-estructurados

Aparentemente los datos semi-estructurados tienen la similitud de ser en parte, **datos estructurados en cuyo interior poseen datos no estructurados.** La realidad es que no están planteados para ser plenamente estructurados como una tabla. Un ejemplo sería los datos representados como en el formato XML.

/ 11. Caso práctico 2: "Tipos estructuras Big Data"

Planteamiento: Se requiere clasificar entre los distintos tipos de estructuras existentes de *Big Data*, los siguientes tipos:

Employee_ID	Employee_Name	Gender	Department	Salary_In_lacs
2365	Rajesh Kulkarni	Male	Finance	650000
3398	Pratibha Joshi	Female	Admin	650000
7465	Shushil Roy	Male	Admin	500000
7500	Shubhojit Das	Male	Finance	500000
7699	Priya Sane	Female	Finance	550000

Fig. 12. Tipo de estructura Big Data.

```
<rec><name>Prashant Rao</name><sex>Male</sex><age>35</age></rec>
<rec><name>Seema R.</name><sex>Female</sex><age>41</age></rec>
<rec><name>Satish Mane</name><sex>Male</sex><age>29</age></rec>
<rec><name>Subrato Roy</name><sex>Male</sex><age>26</age></rec>
<rec><name>Jeremiah J.</name><sex>Male</sex><age>35</age></rec>
```

Fig. 13. Tipo de estructura Big Data.

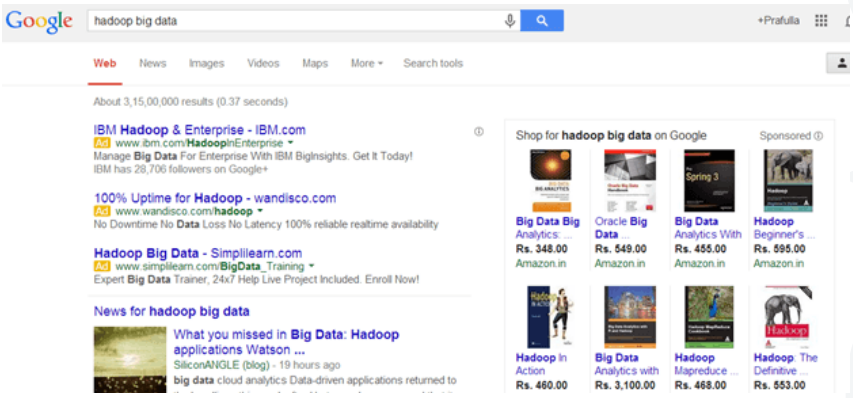

Fig. 14. Tipo de estructura Big Data.

Nudo: Se requiere identificar la tipología de Big Data en función a los parámetros estudiados previamente.

Desenlace: Si nos fijamos en la primera de las imágenes podemos ver que es un tipo de **datos estructurados**, es el ejemplo de una tabla de empleado con sus columnas bien definidas.

En la imagen número 2 podemos ver que tenemos un ejemplo claro de **datos semi estructurados**, un buen ejemplo de datos semi-estructurados son los ficheros XML y su estructura.

En la imagen número 3 podemos observar un ejemplo de datos **no estructurados** en este caso es la respuesta de una búsqueda en famoso navegador google Chrome. Podemos ver que tenemos una respuesta de texto, etiquetas, información multimedia, etc.

/ 12. Resumen y resolución del caso práctico de la unidad

A lo largo de este tema nos hemos centrado en dos temas principales y relacionados. Por un lado, hemos realizado una introducción sobre las **bases de datos No-SQL.** Estudiamos las diferencias que existen con las bases de datos relacionales y aprendimos a instalar un conector desde Java en nuestra primera práctica.

Más adelante vimos el concepto de dato y su orientación a las modernas bases de datos ***Big Data,*** grandes almacenes de datos con características específicas. Vimos también los distintos tipos de estructuras de datos que se pueden dar en *Big Data.*

Resolución del caso práctico inicial

Una vez realizada la investigación de cuáles son las bases de datos No-SQL más conocidas en el mercado, realizaremos la clasificación en las 5 principales:

- **MongoDB:** Estamos ante la base de datos NoSQL más conocida a nivel de solución empresarial. Es un importante gestor de datos que almacena documentos en un formato muy parecido a JSON.

- **Apache Casandra:** Base de datos basada en columnas, diseñada para almacenar cantidades muy grandes de datos y realizar balanceo a distintos nodos. A modo curiosidad es una de las herramientas que se usan en Facebook.

- **CouchDB:** Nació con la idea de ser la principal base de datos NoSQL de la red, pero en 2008 el proyecto pasó a formar parte del proyecto Apache Incoubator. Su intención principal es la de adaptar y compatibilizar la web con distintos tipos de dispositivos. Almacena los datos en formato Json.

- **Redis:** Es otro importante sistema de base de datos NoSQL en este caso clave-valor. Es de código abierto y patrocinada por RedisLabs. Está basado en el almacenamiento de tablas tipo hash, aunque no se cierra solo a esta tipología, no obstante, es su fuerte.

- **Neo4J:** Fue desarrollada en software libre, es totalmente distinta a las 4 anteriores que hemos presentado ya que es una base de datos orientada a grafos, desarrollada en lenguaje Java.

/ 13. Webgrafía

https://www.digitalocean.com/community/tutorials/an-introduction-to-big-data-concepts-and-terminology