**1**

El ***Big Data*** es un concepto que hace referencia a la **acumulación masiva de datos** y a los procedimientos usados para identificar patrones recurrentes dentro de esos datos. Esta cantidad de datos supera la capacidad del software habitual de capturarlos, gestionarlos y procesarlos en un tiempo razonable. Big Data se puede dividir en tres tipos: Estructurado, No estructurado y Semiestructurado

**2**

Los datos que utiliza Big Data pueden ser:

**Generados por las personas**

**Transacciones de datos**

***Marketing* online y contenido web**

***Máquina a máquina* (M2M)**

**Biométricos**

**3**

Dentro de Big Data encontramos NoSQL, nombre que hace referencia a Not Only SQL o No Solo SQL, el cual es un sistema que almacenamiento de datos que no cumple con el esquema entidad-relación que se encuentra en las bases de datos relacionales.

**4**

Los objetivos de una base de datos NoSQL son:

* Presentar simpleza en el diseño de la Base de datos.
* Permitir escalabilidad horizontal. Escalabilidad horizontal se refiere a la posibilidad de aumentar el rendimiento del sistema simplemente añadiendo más nodos, sin necesidad en muchos casos de realizar ninguna otra operación más que indicar al sistema cuáles son los nodos disponibles
* Permitir controlar la disponibilidad de datos.

**5**

A diferencia de las bases de datos relacionales, NoSQL provee un sistema de almacenamiento mucho más flexible y concurrente, y permite manipular grandes cantidades de información de manera mucho más rápida. Son de esquema libre, fáciles de replicar, y cuentan con APIs para acceder de manera simple a los datos. Sin embargo, también presenta una desventaja en relación a las bases de datos relacionales, tales como, no soportan operaciones JOIN.

**6 – 7 – 8 – 9**

Según la manera de almacenar los datos, podemos clasificar a las bases de datos NoSQL en cuatro tipos:

* **Almacenamiento Clave-Valor (Key-Value):** Los datos se almacenan de forma similar a los [diccionarios de datos](https://es.wikipedia.org/wiki/Diccionario_de_datos), donde se accede al dato a partir de una clave única. Los valores son aislados e independientes entre ellos. Este sistema de almacenamiento carece de una estructura de datos clara y establecida. Dentro de este tipo de almacenamiento se encuentra Cassandra como la tecnología más reconocida.
* **Almacenamiento Orientado a Columnas:** Su modelo de datos es definido como “un mapa de datos [multidimensional](https://es.wikipedia.org/w/index.php?title=Multidimensional&action=edit&redlink=1) poco denso, distribuido y persistente. Se orienta a almacenar datos con tendencia a escalar horizontalmente, por lo que permite guardar diferentes [atributos](https://es.wikipedia.org/wiki/Atributo_(inform%C3%A1tica)) y [objetos](https://es.wikipedia.org/wiki/Objeto_(programaci%C3%B3n)) bajo una misma Clave. Permite agrupar columnas en familias y guardar la información cronológicamente, mejorando el rendimiento. Su precursor es [BigTable](https://es.wikipedia.org/wiki/BigTable" \o "BigTable) de Google.
* **Almacenamiento en Grafo:** Se basan en la [teoría de grafos](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_grafos), donde se establece que la información son los nodos y las relaciones entre la información son las aristas. Su mayor uso se contempla en casos de relacionar grandes cantidades de datos que pueden ser muy variables. Encontramos un ejemplo en las redes sociales: en [Facebook](https://es.wikipedia.org/wiki/Facebook" \o "Facebook) cada nodo se considera un [usuario](https://es.wikipedia.org/wiki/Usuario), que puede tener [aristas de](https://es.wikipedia.org/wiki/Arista_(teor%C3%ADa_de_grafos)) amistad.
* **Almacenamiento Documental:** Las [bases de datos documentales](https://es.wikipedia.org/wiki/Base_de_datos_documental) son parecidas a las Claves-Valor, diferenciándose en el dato que guardan, estos datos son semiestructurados. Estos datos pasan a llamarse documentos, y pueden estar formateados en [XML](https://es.wikipedia.org/wiki/XML) o [JSON](https://es.wikipedia.org/wiki/JSON). Todos los documentos tienen una clave única con la que puede ser accedido e identificado explícitamente. Las más conocidos son [CouchDB](https://es.wikipedia.org/wiki/CouchDB) o [MongoDB](https://es.wikipedia.org/wiki/MongoDB" \o "MongoDB).

**10**

Coach Vs MongoDB

Ambas opciones son muy similiares, pero presentan algunas diferencias como, la replicación, Mongo permite replicación maestro-esclavo y Coach solo maestro-maestro. Y también en la fragmentación de datos, donde mongo cuenta con fragmentación nativa de datos, mientras que coach, no presenta un método nativo pero se puede lograr a mediante aplicaciones intermedias.

**11**

CoachDB

Es un gestor de bases de datos de código abierto. La misma emplea JSON para el intercambio y almacenamiento de datos, Javascript como lenguaje de consulta y el protocolo HTTP como API para acceder a los datos. Esta implementado en Erlang, el cual ha sido diseñado para funcionar en dispositivos mucho más pequeños y menos potentes que un teléfono de hoy en día.

**12 – 13 – 14**

Caracteristicas:

* Almacenamiento en documentos: almacena los datos como documentos, que no tienen una estructura física. En la base de datos, tienen un identificador único y no requieren un esquema determinado.
* ACID: Respeta atomicidad, consistencia, aislamiento y durabilidad, implementando una forma de control multiversión, permitiendo manejar gran cantidad de lectores y escritores, sin que surjan conflictos.
* Vistas e Indices: Los datos almacenados se estructuran por medio de vistas. Estas se indexan y así se pueden mantener actualizados esos índices a medida que se agregan, eliminan o actualizan documentos. en CouchDB las vistas se crean usando funciones escritas en el lenguaje JavaScript, aunque se puede usar otros lenguajes como Ruby, Python, etc. mediante plugins.
* Arquitectura distribuida con replicación: se diseñó con teniendo en mente la replicación bidireccional (o sincronización) y la operación off-line. Eso significa que múltiples réplicas pueden tener cada una sus propias copias de los mismos datos, modificarlas y luego sincronizar esos cambios en un momento posterior.
* Interfaz REST: Todos los ítems tienen una URI única que queda expuesta vía HTTP. REST usa los métodos HTTP POST, GET, PUT y DELETE para las cuatro operaciones básicas CRUD (Create, Read, Update, Delete).
* Hecha para operar offline: Está creada para funcionar de manera offline sin ningún tipo de problemas, pudiendo replicar datos a dispositivos (como smartphones) que pueden quedar offline y manejar automáticamente la sincronización de los datos cuando el dispositivo vuelve a estar en línea.

**15 – 16**

Interactuando con CouchDB

El protocolo HTTP, nos permite interactuar de manera intuitiva con Couch, a través de los métodos propios que proporciona. Gracias a esto, podemos llevar a cabo las operaciones básicas CRUD.

* PUT: Permite crear base de datos y tablas, solo con indicar su nombre.
* POST: Permite crear documentos y asignárselos a las bases de datos creadas, solo indicando el nombre del documento.
* GET: El método GET, nos permite obtener tanto datos como estructuras de bases de datos, ya sea de un documento o la misma base de datos. Solo debemos especificar el ID del elemento que queremos consultar.
* DELETE: Este método nos permite eliminar tanto datos, documentos y bases de datos. Al igual que el GET, solo debemos especificar el ID del elemento a eliminar.
* COPY: Por último, tenemos este método, que lo que nos permite es copiar el contenido completo de un documento y crear un documento nuevo con este contenido. En este caso, debemos especificar el nombre del documento a copiar y el nombre del documento a crear.

**17**

Escalando en CouchDB

En CouchDB se pueden escalar tres elementos:

* La lectura de datos: Estas consultas se pueden almacenar en Caché evitando el retraso en estas consultas
* La escritura de datos: La escritura se debe dar en todos los servidores, para eso debemos pensar en clusters.
* Los datos en sí: Para escalar los datos, los mismos deben ser divididos en partes manejables y luego almacenadas en distintos servidores.

**18**

*¿Por qué elegir CouchDB?*

* La arquitectura interna tolera fallos, y los errores se dan en un entorno controlado y se resuelven de manera apropiada. Problemas puntuales no afectan al servidor entero si no que permanecen aislados en una única consulta.
* Está diseñado para manejar tráfico variable. CouchDB normalmente absorbe muchas de las consultas concurrentes sin caerse. Probablemente lleve un poco más de tiempo procesar cada consulta, pero todas se responderán.

**19**

¿Quienes confiaron en CouchDB?

* El Horóscopo y las tarjetas de felicitaciones de cumpleaños en Facebook.
* Ubuntu: Lo utilizaba para el servicio de sincronización “Ubuntu One”.
* BBC: Para su plataforma de contenidos dinámicos
* Meebo: Era una plataforma web, que permitia usar todos los tipos de mensajería (Google Talk, Facebook Messenger, Windows Messenger, Yahoo Messenger), hasta que fue comprada por Google y fue descontinuada.

**20**

Ejemplo Big Data

Al hablar de Big Data, pensamos que esta fuera de nuestro alcance, pero la realidad es que lo vemos cotidianamente, como por ejemplo, cuando utilizamos las redes sociales, toda nuestra información, búsquedas, movimientos y localizaciones son almacenadas para predecir hecho o tomar decisiones a través de los datos recopilados mediante ciertos comportamientos en Internet, ofreciéndonos publicidades y recomendaciones acordes a nuestra persona.