

Conceptos básicos y modelos NoSQL

Qué es NoSQL, cuáles son sus modelos de datos y cuándo es conveniente usar cada uno de ellos.

**Prueba de Evaluación Continua (PEC1)**

Asignatura: Arquitecturas de bases de datos no tradicionales

Autor: Lukaz Martin Doehne

marzo de 2023

# Ejercicio 1 (20%)

1. Explica qué implicaciones tiene el concepto de agregado a nivel operativo para la base de datos.[[1]](#footnote-0)

El concepto de agregado en la base de datos implica **una colección de objetos relacionados que se quieren tratar como unidad independiente** (atómica). Se caracteriza por:

**· La manipulación y el acceso de datos** en un sistema gestor de la base de datos se realiza a través de la unidad mínima conocida como agregado.

**· La consistencia y el control de concurrencia** son importantes para garantizar la integridad en la base de datos y garantizar la definitividad de los cambios sobre el agregado. La responsabilidad de mantener restricciones de integridad recae en los programas y no en la base de datos.

**·** En caso de bases de datos distribuidas, se usa el agregado como **unidad de distribución de datos** en diferentes nodos del sistema.

1. Explica la razón por la que una base de datos orientada a agregados puede favorecer la escalabilidad horizontal.[[2]](#footnote-1)

La escalabilidad horizontal en bases de datos NoSQL orientadas a agregados puede ser favorecida debido a **la capacidad de distribuir los agregados en múltiples nodos** utilizando generalmente técnicas de *hash*, lo que permite tener varias copias del mismo agregado y repartir la carga de trabajo.

Esta estrategia resulta más eficiente ya que aumenta la disponibilidad y el almacenamiento al agregar más nodos al clúster, lo que permite la escalabilidad horizontal de la base de datos.

1. Explica algunas características de la estructura interna de los agregados en el modelo documental. [[3]](#footnote-2)

El modelo documental, a diferencia del modelo clave-valor, se caracteriza por tener una estructura interna de documento.

Una de sus características es que es posible distribuir los documentos entre diferentes nodos mediante la **definición de índices** en el objeto.

Hay que tener en cuenta que se **limitan los tipos de datos que pueden ser almacenados**, lo cual permite un acceso más flexible. Por ejemplo, es posible realizar consultas basada en atributos del documento para **recuperar sólo parte del agregado** en lugar de todo el documento. La mayoría de bases de datos documentales almacenan documentos en formato JSON.

1. Explica si es adecuado o no el uso de bases de datos relacionales para el procesamiento de flujos de datos.[[4]](#footnote-3)

Las bases de datos relacionales **no son ideales para el procesamiento de flujos de datos** debido a su arquitectura de almacenamiento ***pull***. En este tipo de arquitectura, los datos deben almacenarse en la base de datos antes de poder ser consultados.

Esto provoca retrasos innecesarios en **procesos que requieren información en tiempo real**, como en el mercado bursátil.

Una manera más eficiente es la estructura ***push***, como es común en estructuras NoSQL, donde los datos pueden ser almacenados en memoria y, para una escritura más rápida, son escritos en la base de datos en intervalos de tiempo o tramos de datos. Los datos son transmitidos desde el origen hasta el destino sin necesidad de ser almacenados en la base de datos. Además, se pueden sacrificar las propiedades ACID a cambio de una mayor velocidad de respuesta.

# Ejercicio 2 (20%)

### Afirmación 1

La normalización es un proceso del que se benefician las lecturas cuando se realiza un diseño adecuado de los agregados de una base de datos.

**Falso**. La normalización tiene como objetivo evitar redundancias y anomalías en los datos. En bases de datos NoSQL se utiliza la técnica de desnormalización para un mejor rendimiento en la lectura, lo que implica la duplicación de datos o incumplimiento de propiedades ACID.[[5]](#footnote-4)

### Afirmación 2

Una de las diferencias entre los modelos de datos relacionales y el modelo en grafo está en la representación de las relaciones. En el modelo en grafo, las relaciones se representan implícitamente mientras que en el modelo relacional se representan explícitamente.

**Falso**. Justo lo contrario, en el modelo relacional las relaciones entre tablas se representan implícitamente (claves foráneas) y en el modelo de grafos explícitamente (aristas), lo que permite una recuperación más eficiente de los datos relacionados, sin necesidad de operaciones *join*.[[6]](#footnote-5)

### Afirmación 3

Una de las razones por las que la estrategia *one size fits all* no siempre funciona es por la necesidad en las aplicaciones modernas de realizar operaciones simples sobre modelos de datos complejos.

**Verdadero**. Una de las limitaciones de *one size fits alls* es que las consultas a objetos complejos o sin estructura suelen requerir varias operaciones *join*. Para solucionar esto, se puede optar por utilizar bases de datos NoSQL.[[7]](#footnote-6)

### Afirmación 4

Dos características presentes en el modelo documental son: a) Un documento puede ser recuperado mediante su clave o mediante el valor que toman sus atributos, y b) Es necesario definir explícitamente el esquema que seguirán los datos de los documentos.

**Falso.** Aunque bien es cierto que se puede recuperar un documento mediante su clavo o el valor que toman sus atributos, no es necesario definir explícitamente el esquema. La estructura interna es implícita con una definición de estructuras y tipos permitidos. [[8]](#footnote-7)

### Afirmación 5

El responsable de mantener las restricciones de integridad en un conjunto de agregados es el sistema de gestión de bases de datos, el cual se ocupará de comprobar que cualquier operación mantiene la consistencia de la información.

**Verdadero.** La responsabilidad del mantenimiento de las restricciones de integridad no es de la base de datos, sino del sistema gestor de la base de datos.[[9]](#footnote-8)

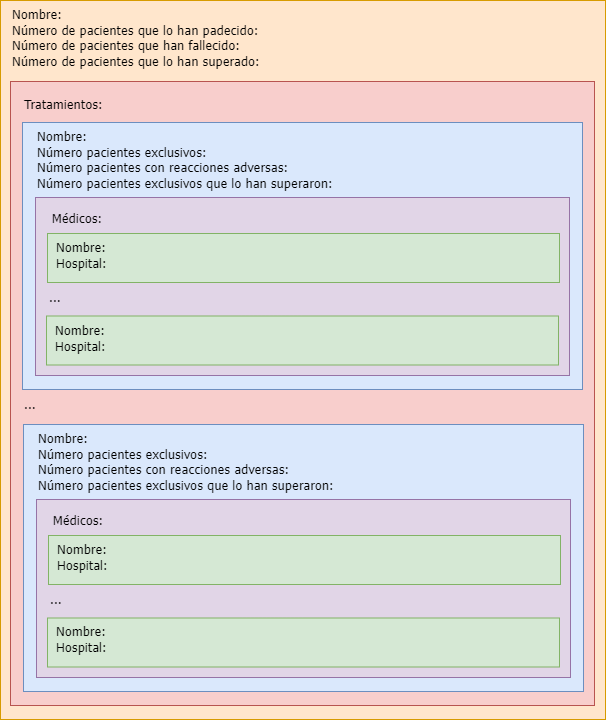
# Ejercicio 3 (30%)

Considerar una base de datos documental del ámbito médico donde se almacena información acerca datos particulares de los pacientes, las dolencias que padecen y los tratamientos que reciben para las dolencias. Se pide diseñar los tipos de documentos más eficientes para las siguientes tres consultas que se desean realizar:

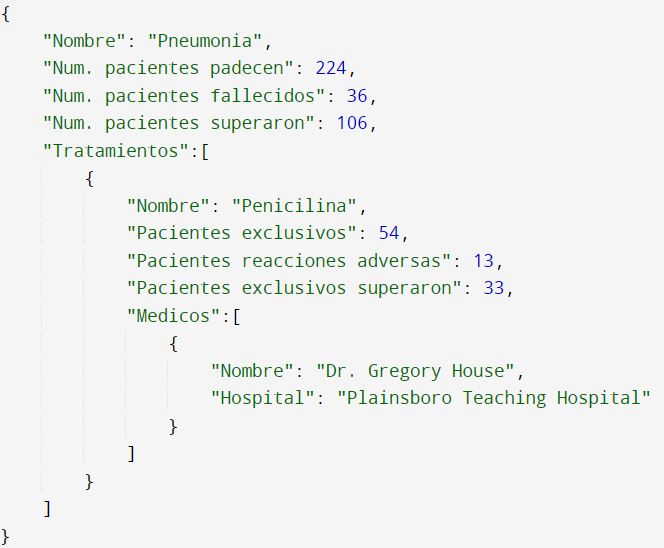
Consulta 1: Para cada tipo de dolencia se quiere conocer su nombre, número de pacientes que la han padecido, número de pacientes que fallecieron debido a esta dolencia, número de pacientes que superaron la enfermedad y una lista de los tratamientos utilizados. Para cada tipo de tratamiento utilizado contra esta dolencia se quiere conocer el nombre del tratamiento, el número de pacientes que siguieron exclusivamente este tratamiento, el número de pacientes que sufrieron reacciones adversas, el número de pacientes que superaron la enfermedad usando exclusivamente el tratamiento, y los médicos que usaron estos tratamientos. Para cada médico se desea conocer su nombre y el hospital en el que trabaja.

Se propone una estructura de documento que contemple un agregado para representar la información de una dolencia específica. Este agregado estará compuesto por campos externos como Nombre, Número de pacientes que padecen, Número de pacientes fallecidos y Número de pacientes recuperados. Asimismo, se incluirá un campo denominado Tratamientos que contendrá una colección de subdocumentos correspondientes a cada tratamiento, y a su vez, cada tratamiento estará compuesto por campos como Nombre, Número de pacientes exclusivos, Número de pacientes con reacciones adversas y Número de pacientes exclusivos recuperados. Finalmente, se incorporará un campo llamado Médicos que contendrá una colección de agregados por cada médico, y cada uno de ellos estará conformado por los campos Nombre y Hospital. Esta estructura permitirá el acceso a toda la información en una única lectura.

El diseño planteado se puede visualizar en el siguiente diagrama:



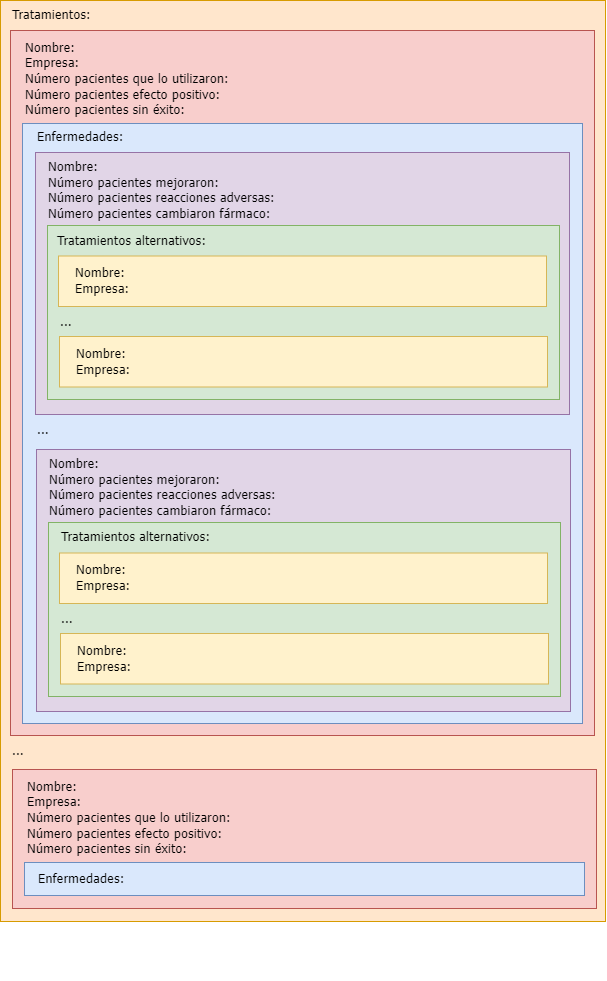
Un ejemplo de la consulta en formato JSON podría ser:



Consulta 2 : Se desea obtener un listado de los tratamientos, donde se quiere conocer su nombre, la empresa que lo fabrica, el número de pacientes que utilizaron el tratamiento, el número de pacientes en los que tuvo efectos positivos sobre la dolencia padecida, el número de pacientes en los que no tuvo éxito y una lista de las enfermedades sobre las que se utilizó. Para cada enfermedad se quiere conocer el nombre de la enfermedad, el número de pacientes que mejoraron usando el medicamento, el número de pacientes que sufrieron reacciones adversas al usar el tratamiento, el número de pacientes que tuvieron que cambiar de fármaco para tratar la enfermedad, y un listado de otros tratamientos alternativos considerados. Para cada tratamiento alternativo se desea conocer su nombre y la empresa que lo fabrica.

Se propone un modelo de documento con una estructura de agregado (Tratamientos) que incluye una colección de subdocumentos para cada tratamiento. Cada subdocumento contendrá información detallada sobre el tratamiento, como Nombre, Empresa, Número de pacientes que lo utilizaron, Número de pacientes con efecto positivo y Número de pacientes sin éxito. Además, se incluirá un campo llamado Enfermedades que contendrá una colección de agregados por cada enfermedad. Cada agregado de enfermedad incluirá información sobre Nombre, Número de pacientes que mejoraron, Número de pacientes con reacciones adversas y Número de pacientes que cambiaron de fármaco. Por último, se incluirá un campo llamado Tratamientos alternativos que contendrá una colección de agregados por cada tratamiento alternativo, cada uno de los cuales contendrá los campos Nombre y Empresa. Con esta estructura, se puede acceder a toda la información relevante en una sola lectura, mejorando la eficiencia y la rapidez del proceso de consulta.

El diseño planteado se puede visualizar en el siguiente diagrama:



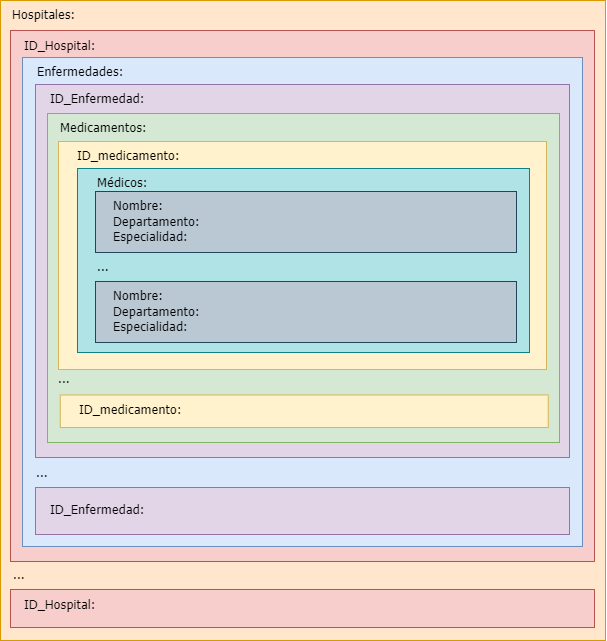
Un ejemplo de la consulta en formato JSON podría ser:



Consulta 3: Para cada hospital se quiere obtener una lista de enfermedades que ha tratado y, para cada una de ellas, la lista de los médicos que trataron esa enfermedad (nombre, departamento y especialidad) agrupados por el medicamento que utilizan para tratar la enfermedad.

Se propone un tipo de documento con un agregado (Hospitales) que contiene una colección de subdocumentos por cada hospital. Cada hospital contendrá un identificador único (ID\_Hospital) y un campo llamado Enfermedades que contendrá una colección de agregados para cada enfermedad que tratan. Cada enfermedad contendrá un identificador único (ID\_Enfermedad) y un campo llamado Medicamentos que contendrá una colección de agregados para cada medicamento usado para tratar la dolencia. Cada medicamento contendrá un identificador único (ID\_medicamento) y un campo llamado Médicos que contendrá una colección de agregados para cada médico involucrado en el tratamiento. Cada médico contendrá los campos: Nombre, Departamento y Especialidad. Con esta estructura, se puede acceder a toda la información en una lectura eficiente y es importante añadir los identificadores únicos para saber a qué hospital, enfermedad o medicamento pertenece la información.

El diseño planteado se puede visualizar en el siguiente diagrama:



Un ejemplo de la consulta en formato JSON podría ser:



# Ejercicio 4 (30%)

Considerad el desarrollo de un sistema de generación de alarmas ante terremotos a nivel estatal que deberá procesar la información sísmica que se recibe de forma continuada de la red de estaciones sísmicas repartidas por todo el territorio peninsular. La aplicación debe ser capaz de procesar, en tiempo real, toda la información y buscar patrones que pudieran indicar el inicio de un terremoto y, en caso de confirmarse, deberá calcular el epicentro del movimiento sísmico mediante una triangulación de la información recibida, así como calcular otras magnitudes como la velocidad de propagación de las ondas y la intensidad entre otras. Así mismo, se sabe que la información recibida en muchos casos contiene datos erróneos, datos ausentes y que la codificación de los datos puede ser diferente dependiendo del fabricante de cada estación sísmica. También se conoce que periódicamente se cambian las estaciones sísmicas por otras modernas en las que probablemente se utilicen otras codificaciones de la información. Con ese supuesto en mente, os animamos a realizar la lectura del artículo: “Data management in cloud environments: NoSQL and NewSQL data stores” que se puede encontrar en la siguiente dirección web: <https://journalofcloudcomputing.springeropen.com/articles/10.1186/2192-113X-2-22>

Tomando como base la lectura del artículo, así como los apuntes de los temas vistos, se pide responder de forma breve y concisa, pero justificada, a las siguientes preguntas acerca del supuesto planteado:

1. De acuerdo a las características del sistema que se plantea construir, ¿Qué tipo de base de datos sería mejor (NoSQL/Relacional/ NewSQL/Otros tipos de bases de datos)?

Debido a la necesidad de procesamiento en tiempo real y datos inconsistentes (nuevas codificaciones, datos ausentes o erróneos) la base de datos recomendable es NoSQL.

La base de datos NewSQL y las bases de datos relacionales no presentan un esquema flexible[[10]](#footnote-9). Dado que se espera que la estructura de los datos cambie con frecuencia, estas dos opciones presentan demasiada rigidez. Además, las bases de datos relacionales podrán tener problemas de escalabilidad (volumen alto de datos) y rendimiento (procesamiento en tiempo real).[[11]](#footnote-10)

A la base de datos NoSQL se le atribuye las características de escalabilidad horizontal, proporcionar alta disponibilidad y tener esquemas flexibles[[12]](#footnote-11), que son necesarios para el sistema de alarmas que se comentarán en el siguiente punto.

La aplicación ha de ser capaz de calcular el epicentro del movimiento sísmico mediante la triangulación, así como otras magnitudes. Para ello usamos el procesamiento distribuido, otra característica de las bases de datos NoSQL, donde cada nodo podría trabajar en paralelo recolectando la información de su área geográfica, y la información de todos los nodos podría combinarse para calcular la ubicación del inicio del terremoto.

Dado el contexto del sistema de alarmas, una base de datos NoSQL de tipo documental en formato JSON podría ser una buena opción. Este tipo de modelo tiene una estructura interna al que se han de adaptar las estaciones sísmicas y utiliza un sistema distribuido de nodos. Un ejemplo podría ser MongoDB.[[13]](#footnote-12)

1. Describe breve y conceptualmente cómo serían las características del sistema en cuanto a disponibilidad, escalabilidad, consistencia de los datos, y distribución geográfica.

Cada una de estas características es necesaria para la creación del sistema de generación de alarmas:[[14]](#footnote-13)

· **Alta disponibilidad**: ha de ser capaz de procesar información en tiempo real, ya que se trata de un sistema crítico de alarmas en el que pequeños retrasos podrían poner en riesgo vidas humanas.

**· Escalabilidad horizontal:** se espera que el sistema procese una gran cantidad de información de las estaciones sísmicas repartidas por todo el territorio peninsular. La escalabilidad horizontal nos permite aumentar el almacenamiento y procesamiento añadiendo nuevos nodos al clúster.

**· Consistencia de datos:** es necesario debido a la naturaleza de los datos sísmicos (datos erróneos, ausentes o con diferentes codificaciones) que se tenga un esquema flexible, así una estación sísmica más moderna puede añadir datos sin necesidad de cambios significativos en la estructura de la base de datos y asegura un almacenaje fiable. Paralelamente, se han de procesar y asegurar de manera automática que los datos sean válidos y con la codificación correcta.

**· Distribución geográfica:** idealmente, el sistema de generación de alarmas usa un procesamiento distribuido. En el cual, cada nodo representando estaciones sísmicas del territorio peninsular recolectan información de su área geográfica. Por lo que si una estación sísmica falla (e.g. por desastres naturales) las estaciones cercanas hayan podido recolectar información y el sistema pueda emitir la alarma igualmente.

1. [B2\_T3\_1\_ModelosAgregacionMotivacion.pdf](https://campus.uoc.edu/webapps/classroom/download.do?nav=activitats&sub-nav=descarregar-adjunt&id=959290&serial=false&s=9df92b53ff0f967297b0f4298b58b4300c846a0c9c89c2ddc971864bb58e7c9213d10d79c919742832f9127a718f607939257d246c1f6a6139fa6870039200a1&domainId=928860&proposedFilename=B2_T3_1_ModelosAgregacionMotivacion.pdf&idLang=&classroomId=933928&subjectId=928860&activityId=1260136&javascriptDisabled=false): pg. 5,7-13, NoSQL Distilled-Aggregate Data Models: chapter 2.1, B2\_T3\_2\_ModelosAgregacionCaracteristicas.pdf: pg. 4,6,7 [↑](#footnote-ref-0)
2. PID\_00236790.pdf: pg. 25, B2\_T3\_2\_ModelosAgregacionCaracteristicas.pdf: pg.6 [↑](#footnote-ref-1)
3. B2\_T3\_3\_ModelosAgregacionTipos%20.pdf: pg. 6-9 ,NoSQL Distilled-Aggregate Data Models: chapter 2.2 [↑](#footnote-ref-2)
4. B1\_T2\_PersistenciaPoliglota.pdf: pg. 13,15 [↑](#footnote-ref-3)
5. B2\_T3\_1\_ModelosAgregacionMotivacion.pdf: pg. 8,9, B2\_T3\_2\_ModelosAgregacionCaracteristicas.pdf: pg. 4, B1\_T2\_PersistenciaPoliglota.pdf: pg. 15 [↑](#footnote-ref-4)
6. B2\_T4\_1\_ModelosEnGrafo.pdf: pg. 15, 17 [↑](#footnote-ref-5)
7. B1\_T2\_PersistenciaPoliglota.pdf: pg. 10, 11 [↑](#footnote-ref-6)
8. B2\_T3\_1\_ModelosAgregacionMotivacion.pdf: pg. 6 ,NoSQL Distilled-Aggregate Data Models: chapter 2.2 [↑](#footnote-ref-7)
9. B2\_T3\_1\_ModelosAgregacionMotivacion.pdf: pg. 4,5 [↑](#footnote-ref-8)
10. [NoSQL and NewSQL](https://journalofcloudcomputing.springeropen.com/articles/10.1186/2192-113X-2-22#Sec2:~:text=The%20category%20of%20NewSQL%20data,the%20well%2Dknown%20SQL%20standard) Background and related work. - NoSQL and NewSQL - último párrafo [↑](#footnote-ref-9)
11. B1\_T1\_IntroduccionNoSQL.pdf: pg 9 [↑](#footnote-ref-10)
12. [NoSQL and NewSQL](https://journalofcloudcomputing.springeropen.com/articles/10.1186/2192-113X-2-22#Sec2:~:text=NoSQL%20and%20NewSQL.-,NoSQL%20and%20NewSQL,enterprises%20are%20other%20barriers%20to%20the%20adoption%20of%20NoSQL%20data%20stores.,-The%20category%20of) Background and related work. - NoSQL and NewSQL [↑](#footnote-ref-11)
13. B2\_T3\_3\_ModelosAgregacionTipos.pdf: pg. 6-9 [↑](#footnote-ref-12)
14. B1\_T2\_PersistenciaPoliglota.pdf pg. 10-13 [↑](#footnote-ref-13)