Caso Final:

Persistencia políglota y Bases de datos noSQL



Gema Caballero Muñoz

Alfonso Campos De Padua

Lourdes Díaz Mena

**Descripción del contexto:**

En aras de acabar con la piratería, un conjunto de productores independientes ha creado un portal web para visualizar películas y series *online*. El portal, llamado *Veeme*, funciona mediante suscripción mensual. En los últimos meses, el número de usuarios de *Veeme* ha aumentado exponencialmente gracias a su novedad, su publicidad y a su eficaz sistema de recomendación. Debido a este gran crecimiento, se debe renovar con la máxima celeridad su sistema de almacenamiento. Nos piden, como conocedores de las tecnologías NoSQL, que hagamos una propuesta de sistema de almacenamiento para el nuevo sistema.

**Gestión de suscripciones:**

Nos dice que el modelo se actualiza poco y que no es necesario que satisfaga una consistencia fuerte, sin embargo si es muy importante contar una gran disponibilidad ya que se soportará un elevado volumen de lectura.

Consecuencias:

Como no es necesario mantener una consistencia fuerte podemos descartar un modelo CP,

por otra parte se necesita contar de una gran disponibilidad por lo que también descartamos un

modelo AC.

Dado que lo prioritario es leer una clave y recuperar su valor (como si de una cache se tratara)

parece un caso de libro de uso de un sistema CLAVE-VALOR, por lo que hemos escogido la

implementación vista en el curso clave-valor (RIAK).

id\_usuario: {Estado}={Si/No}

**Catalogo de pelis y series:**

Se nos comunica que la integridad de los datos para el catálogo es un factor muy importante,

además se nos informa que el sistema no escala horizontalmente en demasía, por otra parte la finalidad es almacenar solo la metadata, ya que los datos en si serán accesibles desde otra infraestructura. El catalogo de películas y series debería ser accesible por un elevado número de usuarios.

Consecuencias:

La necesidad de que la integridad se mantenga nos lleva directamente a pensar en un sistema CP que granate la consistencia, por ello descartamos los sistemas AP (RIAK y CASSANDRA), así como el sistema CA (NEO4J).

Por ello nuestra decisión es escoger un sistema CP que garantice consistencia pero también soporte particiones, en este caso la solución escogida es MONGODB, el ejemplo documental visto durante el curso.

id\_peliserie: \_\_\_\_\_\_

disponible:{Si/No}

paises\_permitidos: {España, Italia,...}

nombre: \_\_\_\_

actores: \_\_\_\_

directores: \_\_\_\_

género: \_\_\_\_

año: \_\_\_\_

sinopsis: \_\_\_\_\_

**Análisis de usuarios:**

Esta base de datos va a recoger la información de navegación de los usuarios por el sistema, como factor resaltado se señala que debe soportar por ende una alta disponibilidad, además se nos informa de que el sistema debe ser capaz de proveer una elevada escalabilidad horizontal.

Consecuencias:

Dado el factor de elevada disponibilidad que requiere este sistema, podemos descartar la soluciones CP como MONGODB, así como soluciones CA (NEO4J).

Por tanto nuestra solución debe venir de la mano de un sistema AP. A la hora de escoger clave valor o columnar, nos hemos decantando por este último dado que el formato de almacenamiento de la información, tal y como se va almacenar, puede aprovechar las ventajas a nivel de búsqueda que proporciona las bases de datos columnares.

id\_usuario:

sesiontime

url\_página

timestamp

**Sistema de recomendación de usuarios:**

Debemos proporcionar una base de datos que almacene para cada usuario sus votaciones en películas, series y actores, así como la lista de recomendaciones actualizada por el sistema.

Las especificaciones no proveen un elevado detalle sobre los requisitos de escabilidad vertical del sistema, si bien podemos deducir de los patrones de uso que no van a ser tan frecuentes las escrituras de los usuarios como en los casos anteriores ni tampoco la actualización de la recomendaciones por el sistema. Además la estructura de los datos a almacenar debe de tener en cuenta el usuario en cuestión que hace la valoración, el objeto de esa valoración, así como incorporar las recomendaciones del sistema.

Consecuencias:

Todo ello nos lleva a estimar que en este caso no es necesario contar con un sistema donde prime el factor alta disponibilidad, por tanto descartamos el uso de un sistema AP (RIAK y CASSANDRA). Esto viene reforzado por el tipo de datos que vamos almacenar y como se debe de actualizar parcialmente los mismo, lo cual no encaja bien con los sistemas clave-valor o comulnares (el objeto valor es un objeto compuesto complejo que había que acceder con un patrón a medida y no es fácilmente estructurar la información en columnas).

Los sistemas CA los descartaríamos dado que no soporta distintas participaciones que den la carga de rendimiento que tendrá que proveer el sistema de datos, por lo que lo más adecuado es contar con un sistema documental (MONGODB), que provee mecanismo que permiten actualizar fácilmente las partes del versus al todo frente al conjunto proporcionando al mismo tiempo distintas particiones que den un mayor rendimiento.

id\_usuario: \_\_\_\_\_\_\_\_\_\_\_

valoraciones\_peliserie: {id\_peliserie: valor}

valoraciones\_actores: {id\_actor: valor}

recomendaciones: {id\_peliserie1, id\_peliserie2,..}