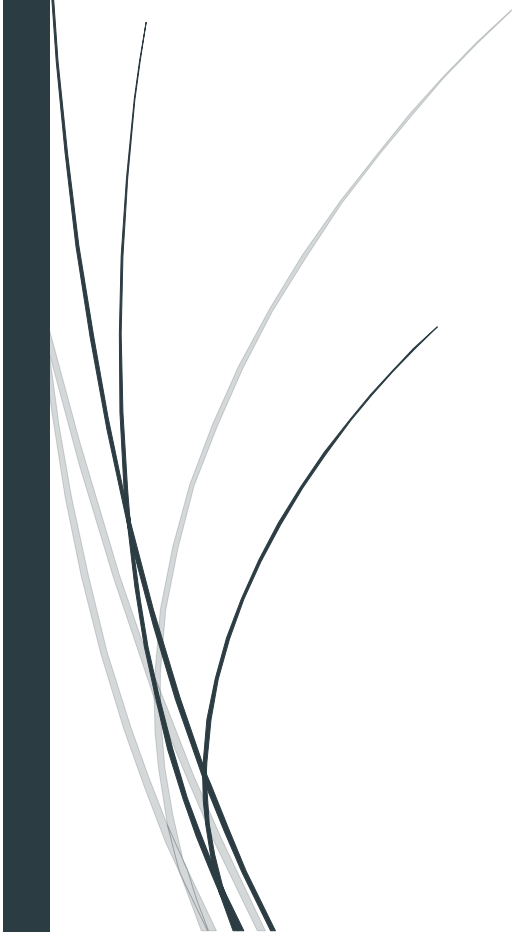
27-10-2024

Bases de datos no relacionales

Pec1



Juan Luis Acebal Rico
GRADO DE CIENCIA DE DATOS APLICADA

Indice

Ejercicio 1

1. Explica qué ventajas y desventajas puede tener el hecho de que una base de datos NoSQL no tenga un esquema predefinido.

Una de las principales ventajas de las bases de datos NoSQL (o más bien, NO ¨solamente¨ SQL, ya que no podemos prescindir de SQL hoy en día), es que no requiere un esquema predefinido, y eso da flexibilidad en la gestión y en el almacenamiento de datos. Esto ayuda a manejar datos de todo tipo, especialmente semiestructurados y estructurados, y adaptándose rápidamente a los cambios sin problemas de esquemas rígidos, como es el caso de las BD relacionales, y esto ayuda en el despliegue y desarrollo de aplicaciones, y junto a las bases de datos relacionales, se convierten en 2 herramientas muy poderosas en las organizaciones.

Cita: (Pagina 5 de B2_T3_1_ModelosAgregacionMotivacion.pdf: ¨…restricciones de integridad principalmente en los programas…¨)

2. Explica por qué la necesidad de almacenamiento de datos de forma distribuida supone una limitación a la estrategia *one size fits all* asociada a las bases de datos relacionales.

La estrategia one size fits all, de las bases de datos relacionales se basan en la idea de que un solo tipo de bases de datos puede satisfacer todas las necesidades de almacenamiento. Pero hoy en día no es así, la creciente necesidad de almacenar grandes cantidades de datos de forma distribuida limita esta estrategia, siendo el límite físico de las bases de datos no relacionales, ya que no están diseñadas para escalar horizontalmente de manera eficiente, y ningún sistema puede escalar verticalmente en gran cantidad.

Claro, si no podemos escalar verticalmente en ningún sistema tanto como quisiéramos (recordemos que escalar verticalmente significa un mejor equipo informático, hacer esto se vuelve caro y tiene límites), y escalar verticalmente no es eficiente en SQL, ya que eso significaría ¨dividir¨ las tablas, existiendo varios métodos para hacerlo.

Y ahora viene explicar qué es la persistencia políglota, y es simplemente usar en una organización, diferentes sistemas informáticos, y siendo el que más convenga para cada situación dentro del mismo “proyecto”, así podemos mejorar el rendimiento, ya que, si no tenemos un único sistema, al tener “varios” estamos escalando verticalmente, es decir, dividimos las cargas de trabajo en diferentes sistemas, y además no solamente es eso, al usar diferentes sistemas, mejoramos la escalabilidad, y la eficiencia del conjunto del sistema.

Cita: (¨Guía de creación de particiones de datos¨, https://learn.microsoft.com/es-es/azure/architecture/best-practices/data-partitioning)

(Pagina 11 de B2_T3_1_ModelosAgregacionMotivacion.pdf ¨…importante es que el diseño de agregados no da soporte a cualquier necesidad de información…¨)

3. Explica la afirmación “en el modelo en grafo, la representación de las relaciones entre datos es más natural, clara y eficiente”

Un modelo de grafos es una representación de los datos como nodos y relaciones, que éstas son aristas, y en ellas se crean conexiones que representan la información del mundo real. Esto facilita la representación y comprensión de relaciones complejas sin hacer decenas de operaciones de unión de estilo join, lo cual, al menos en mi entorno laboral, lo he visto y son consultas SQL muy largas, con muchos joins, que no es fácil comprender qué información estás extrayendo.

Realmente esto puede ser interesante en redes sociales, en representación de relaciones laborales o familiares, por ejemplo, en sistemas de recomendación o de detección de fraudes, entre otros.

Cita: (Pagina 11 de B2_T4_1_Modelosengrafo.pdf: ¨…grafos puede representar de forma más natural datos altamente…¨)

4. Explica qué implicaciones tiene la redundancia de información en el modelo de agregados.

El modelo de agregados agrupa datos relacionados en una sola unidad de almacenamiento, que puede ocasionar redundancia de información. Esto permite acceder a los datos necesarios en una única consulta sin tener que hacer diferentes operaciones de unión

Sin embargo, la redundancia, también implica actualizaciones y escrituras pesadas y complejas. Los cambios redundantes deben propagarse en todos los lugares donde aparecen, y esto aumenta la complejidad del sistema, y el riesgo de crear inconsistencias si no se gestiona adecuadamente.

Por tanto, es importante, crear mecanismos de sincronización y coherencia, además de una buena supervisión, formación y experiencia para poder mantener la integridad de los datos en todo el sistema en todo momento.

Cita: (Página 13 de B2_T3_1_ModelosAgregacionMotivacion.pdf: ¨la aplicación deberá estar atenta a que no se produzca ningún error de operación¨ y ¨ Fijémonos que tenemos un problema derivado de la redundancia de los datos …¨)

Ejercicio 2

Afirmación 1. Se necesita crear un sistema para almacenar la información biométrica de relojes deportivos. Cada modelo de reloj puede almacenar información con diferentes datos y estructuras. En este caso, ¿la estrategia *one size fits all* con una base de datos relacional funcionaría adecuadamente?

No. Las BBDD relacionales requieren un esquema claro y predefinidido que no puede cambiar en tiempo de ejecución o en producción sin más. La Persistencia Políglota funcionaría aquí muy bien, ya que permitiría adaptar cada necesidad y estructura con un sistema de almacenamiento diferente.

Cita: (Pagina 4 de B2_T3_3_ModelosAgregacionTipos.pdf: ¨…en NoSQL, las bases de datos suelen ser aproximaciones a los modelos de datos. Esto significa que tanto pueden no implementarlos en su totalidad, como pueden añadir funcionalidades extra Este hecho puede provocar que algunas bases de datos NoSQL se puedan clasificar de diferentes maneras, según la fuente de información ….¨)

Afirmación 2. Una aplicación de detección de terremotos donde se procesan los datos de forma continuada y en tiempo real encaja mejor con una arquitectura *pull*.

No, en este caso sería una arquitectura push, es decir, donde los datos se envían automáticamente conforme se producen, y no una arquitectura pull, que almacena datos hasta que se "piden". Aunque los materiales no abordan claramente los modelos de agregación, si se indica que hay ciertos modelos más adecuados para alto rendimiento en escritura y procesamiento en tiempo real.

Cita: (Pagina 14 de B2_T3_3_ModelosAgregacionTipos.pdf: ¨…recuperar datos eficientemente, sólo con los atributos relevantes en cada momento. Otra característica de este modelo es que, en comparación con el modelo relacional, no todas la filas de una misma tabla deben tener el mismo conjunto de columnas. El modelo por columnas puede ser un poco confuso al principio, pero es un modelo muy versátil y con muchas posibilidades…¨)

Afirmación 3.En el modelo en grafo, hay una diferenciación semántica entre los conceptos y los objetos del mundo real, de manera que se representan de distinta forma.

No, no existe diferencia de significado entre conceptos y objetos del mundo real, de modo que ambos se representan como nodos conectados por aristas.

Cita: Pagina 15 de B2_T4_1_Modelosengrafo.pdf:
¨No obstante, en el modelo en grafo, los conceptos y los objetos del mundo real se representan de la misma forma: mediante nodos.¨

Afirmación 4. En una aplicación de reservas de habitaciones de hotel, donde los elementos principales (habitaciones, reservas y clientes) pueden organizarse de forma jerárquica, utilizar un modelo de agregación no es una buena idea.

No, es una buena idea en este caso. Ya que los datos pueden agruparse y accederse conjuntamente, y lo más interesante sería utilizar sus relaciones que representen una reserva completa.

Cita: Pagina 8 de B2_T4_1_Modelosengrafo.pdf:
¨ A partir de lo que hemos visto, es fácil intuir que el modelo en grafo es útil cuando los datos a almacenar tienen multitud de relaciones y cuando …¨

Afirmación 5. A diferencia del modelo clave-valor, en el modelo documental los agregados se pueden recuperar tanto por su clave como por su contenido.

Si. Aquí podemos buscar por su clave, o por incluso, por el contenido de los documentos, y a diferencia del modelo clave-valor, que únicamente se puede por su clave.

Cita: (Pagina 6 de B2_T3_3_ModelosAgregacionTipos.pdf: ¨Los documentos se acceden mediante una clave única, o mediante atributos de los mismos. Esto permite, por ejemplo: – La recuperación de una parte del documento…¨)

Ejercicio 3

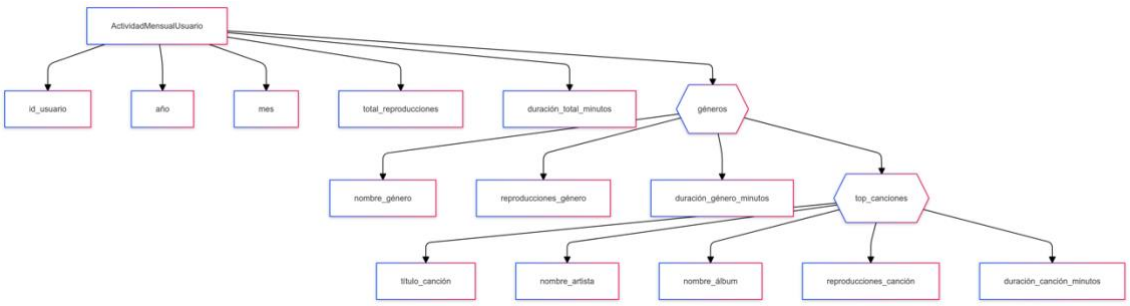
CONSULTA 1

Para cada usuario, mes y año se quiere obtener:

El número total de reproducciones.

La duración total en minutos de esas reproducciones.

Una lista de esas reproducciones por género, guardando el nombre del género, el número, su duración total, y una lista de las canciones más reproducidas, esas canciones a su vez tienen que guardar su nombre, el artista, álbum, número de reproducciones y duración de las reproducciones realizadas. Por tanto, la estructura vista en un diagrama y su archivo JSON son:




Id_usuario:
Año:
Mes:
Total_reproducciones:
Duracion_total_minutos:
Generos:
Nombre_genero:
Reproducciones_genero:
Duración_genero_minutos:
Top_canciones:
Titulo_cancion:
Nombre_artista:
Nombre_album:
Reproducciones_cancion:
Duración_cancion_minutos:
Titulo_cancion:
Nombre_artista:
Nombre_album:
Reproducciones_cancion:
Duración_cancion_minutos:
Nombre_genero:
Reproducciones_genero:
Top_canciones:
Titulo_cancion:
Nombre_artista:
Nombre_album:
Reproducciones_cancion:
Duración_cancion_minutos:

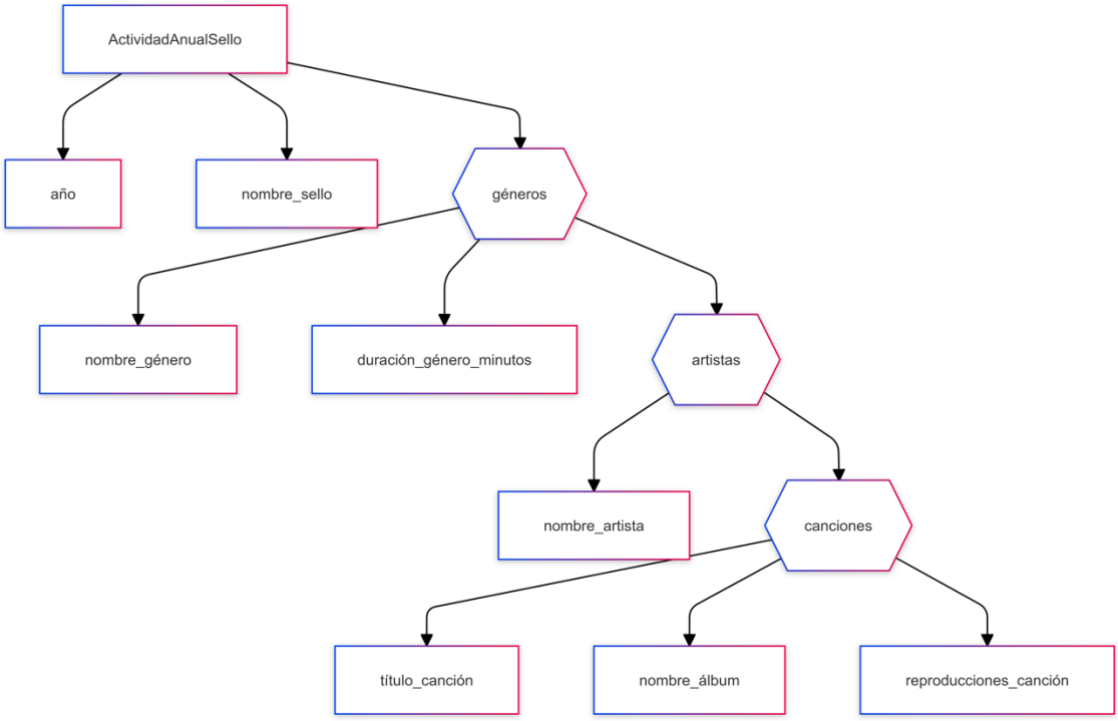
Y su JSON:

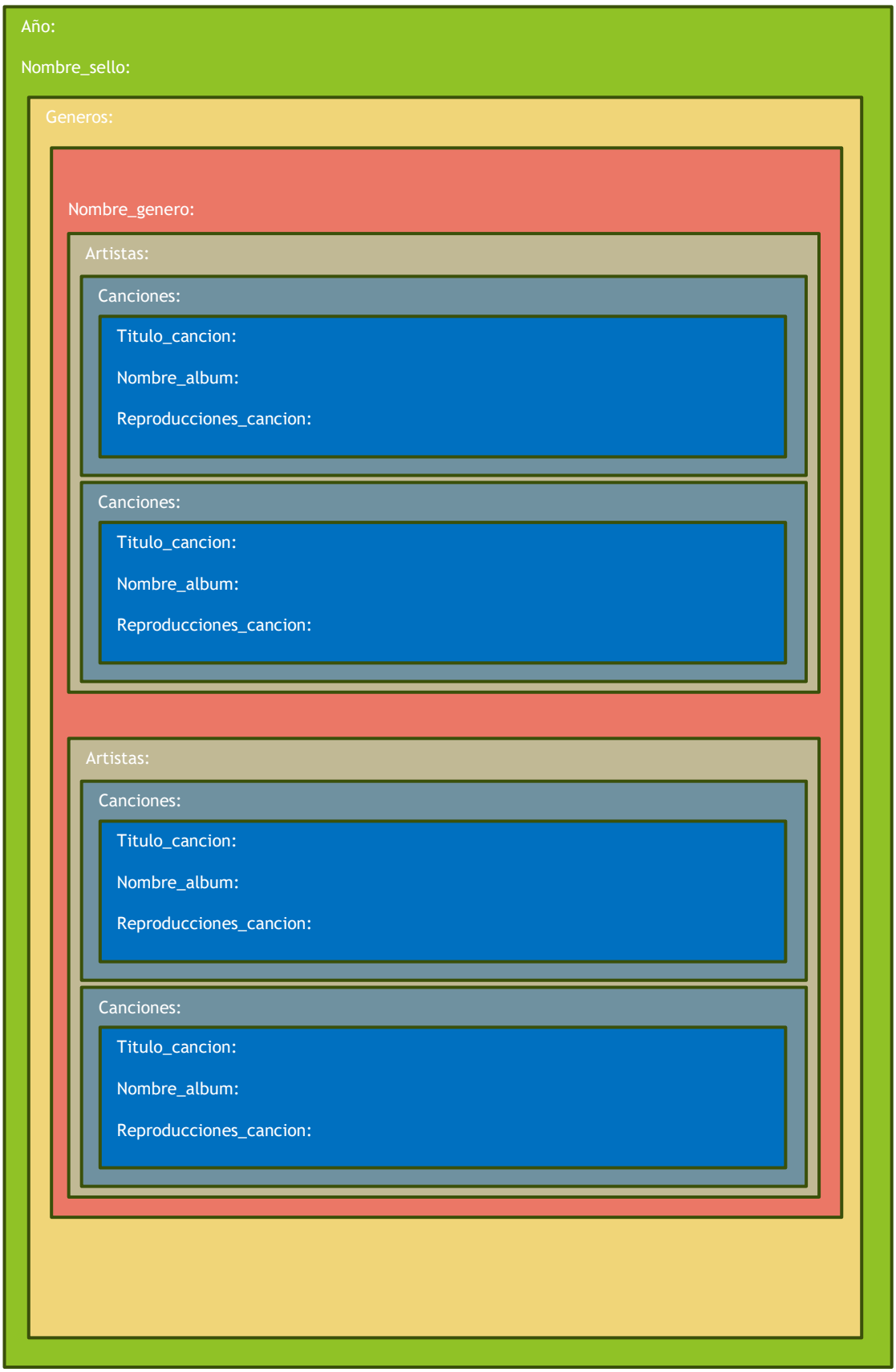
```json
{
  "id_usuario": "juanluisacebal",
  "año": 2024,
  "mes": 10,
  "total_reproducciones": 150,
  "duración_total_minutos": 564,
  "géneros": [
    {
      "nombre_género": "Rock",
      "reproducciones_género": 80,
      "duración_género_minutos": 270,
      "top_canciones": [
        {
          "título_canción": "Thunderstruck",
          "nombre_artista": "AC/DC",
          "nombre_álbum": "The Razors Edge",
          "reproducciones_canción": 30,
          "duración_canción_minutos": 113
        },{
          "título_canción": "Back in Black",
          "nombre_artista": "AC/DC",
          "nombre_álbum": "Back in Black",
          "reproducciones_canción": 20,
          "duración_canción_minutos": 89
        }
        // ... top canciones
      ]
    },{
      "nombre_género": "Pop",
      "reproducciones_género": 70,
      "duración_género_minutos": 280,
      "top_canciones": [
        {
          "título_canción": "Cliris Runs",
          "nombre_artista": "Elisabeth",
          "nombre_álbum": "When you go?",
          "reproducciones_canción": 25,
          "duración_canción_minutos": 75
        }
        // ... top canciones
      ]
    } // ... otros géneros
  ]}
```

CONSULTA 2

Ahora tenemos, una división anual para cada sello discográfico, al que le pertenecen una serie de reproducciones de una plataforma, y se quiere obtener como:

Para cada género, las reproducciones de ese sello discográfico, y guardaríamos los datos del nombre del género, el tiempo de reproducción, y por cada artista, su nombre, y el top de canciones incluyendo el título de la canción, el álbum, y el número de reproducciones de esa canción:




Año:
Nombre_sello:
Generos:
Nombre_genero:
Artistas:
Canciones:
Titulo_cancion:
Nombre_album:
Reproducciones_cancion:
Canciones:
Titulo_cancion:
Nombre_album:
Reproducciones_cancion:
Artistas:
Canciones:
Titulo_cancion:
Nombre_album:
Reproducciones_cancion:
Canciones:
Titulo_cancion:
Nombre_album:
Reproducciones_cancion:

Y el JSON que representa esta información:

```json
{
  "año": 2023,
  "nombre_sello": "Universal Music Group",
  "géneros": [
    {
      "nombre_género": "Rock",
      "duración_género_minutos": 5000,
      "artistas": [
        {
          "nombre_artista": "Queen",
          "canciones": [
            {
              "título_canción": "Bohemian Rhapsody",
              "nombre_álbum": "A Night at the Opera",
              "reproducciones_canción": 1000
            },
            {
              "título_canción": "Don't Stop Me Now",
              "nombre_álbum": "Jazz",
              "reproducciones_canción": 800
            }
          ]
        },{
          "nombre_artista": "The Rolling Stones",
          "canciones": [
            {
              "título_canción": "(I Can't Get No) Satisfaction",
              "nombre_álbum": "Out of Our Heads",
              "reproducciones_canción": 600
            }
          ]
        }
        // ... otros artistas
      ]
    },{
      "nombre_género": "Pop",
      "duración_género_minutos": 4000,
      "artistas": [
        {
          "nombre_artista": "Taylor Swift",
          "canciones": [
            {
              "título_canción": "Shake It Off",
```

```
44                "nombre_álbum": "1989",
45                "reproducciones_canción": 1200
46              }
47            ]
48          }
49          // ... otros artistas
50        ]
51      }
52      // ... otros géneros
53    ]
54  }
55
56
```

Ejercicio 4

1. Una base de datos relacional es la mejor solución para las necesidades de almacenamiento y procesamiento descritas en el supuesto.

No, Aquí las bases de datos relacionales tienen el problema de manejar grandes cantidades de datos heterog'eneos, y que se mueven a alta velocidad. Según el artículo, las BD NoSQL y NewSQL se presentan como alternativas capaces de manejar grandes cantidades de datos, con escalabilidad y que estos datos no tengan una estructura tan rigida como en SQL.

2. La utilización de un esquema explícito para los datos a almacenar sería mejor que usar un esquema implícito. Además, en caso de no usar un modelo relacional, la mejor solución sería utilizar una base de datos NewSQL.

No, dado que los datos vienen de fuentes heterogéneas, y tienen estructuras variables, utilizar un sistema rígido no sería la solución mas inteligente. Tal y como dice el artículo, las BD NoSQL, ofrecen esquemas flexibles, e incluso a veces no tienen esquemas, y por eso están preparadas para trabajar con muchas estructuras distintas de datos.

3. De acuerdo con los requisitos planteados, la solución debe ser fácilmente escalable en un entorno distribuido.

Si, al ser una gran cantidad de datos los que implica los dispositivos IoT, que además tienen que ser procesados en tiempo real, es importante que sea un entorno escalable y distribuido para manejar tan cantidad de datos. El articulo habla sobre cloud vs onPremise.

4. Respecto a la replicación, los datos de interés no permiten una fragmentación ni una replicación eficiente. Por lo tanto, sería mejor que no haya réplicas y exista una única copia en un entorno centralizado.

No, aunque los datos son diferentes, pueden estar fragmentados, y no organizados, y esto puede llevar a que sea más complicado replicarlo, tener una copia tampoco es una solución, ya que no va a ser práctico para tratar tan cantidad de datos y que la disponibilidad también sea la requerida. El articulo habla de eso, de la replicabilidad, que aumenta la confiabilidad y tolerancia a fallos, además de que es esencial que sea así si se quiere que ese sistema dure.

5. Si se considera que la mejor solución es una base de datos NoSQL, entonces el modelo óptimo sería orientado a grafos.

No necesariamente, aunque las BD orientadas a grafos son útiles para manejar datos altamente relacionados, quizás el caso de uso aquí no sea ese, o pudiera valorarse otro ya que lo importante en este caso de uso no son las relaciones, y en un caso de uso de BD de grafos normalmente lo importante son las relaciones y datos que entre si están interconectados. Quizas aquí un modelo documental como un dato semiestructurado en JSON, o columnas tal y como es una BD relacional, o mismo un archivo CSV, serían a mi juicio, mejor idea que un modelo orientado a grafos, ya que no veo mucho qué tendría que ver una relación con valores de aparatos IoT.