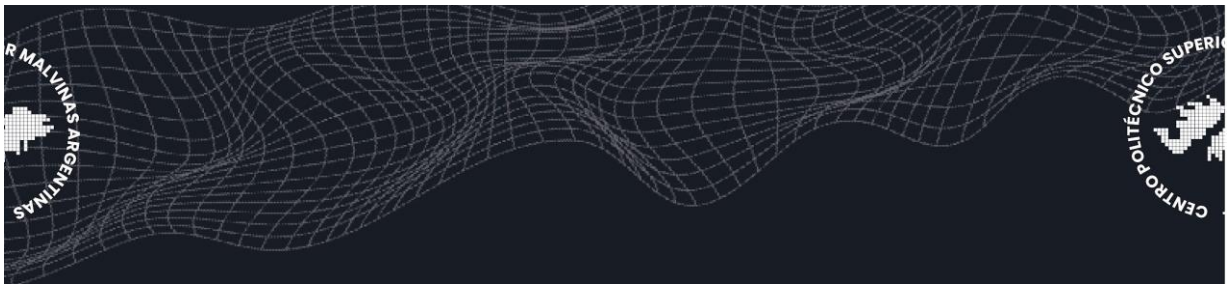


BASE DE DATOS

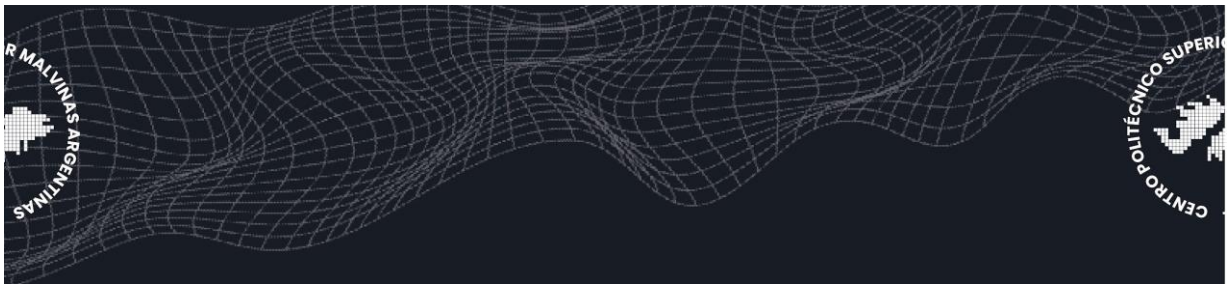
1° AÑO

Clase Nº 9: Introducción a NoSQL: Explorando las Bases de Datos No Relacionales.

Contenido: que es una base de datos no relacionales, características,tipos de base de datos NoSQL, casos de usos , ventajas y desventajas.

En la clase de hoy trabajaremos los siguientes temas:

- ¿Qué son las bases de datos NoSQL?
- Características principales de las bases de datos NoSQL
- Tipos de bases de datos NoSQL
- Casos de uso comunes para bases de datos NoSQL
- Ventajas y desventajas de las bases de datos NoSQL



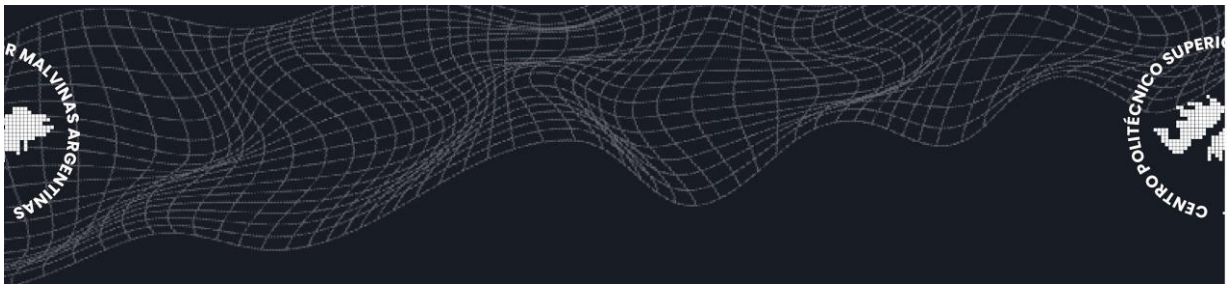
1. Presentación:

Bienvenidos a la clase N° 9 del espacio BASE DE DATOS, l día de hoy nos embarcaremos en un viaje para descubrir el mundo de las bases de datos NoSQL. En esta clase introductoria, exploraremos los conceptos fundamentales de este paradigma de almacenamiento de datos, sus características distintivas y los diferentes tipos de bases de datos NoSQL que existen.

2.-Desarrollo y Actividades:

¿Qué son las bases de datos NoSQL?

Las bases de datos NoSQL, también conocidas como bases de datos no relacionales, son un tipo de sistema de gestión de bases de datos (SGBD) que se aleja del modelo relacional tradicional. En lugar de estructurar los datos en tablas con filas y columnas rígidas, las bases de datos NoSQL ofrecen mayor flexibilidad y escalabilidad para almacenar y consultar datos diversos.



Las base de datos no relacionales o NoSql como mejor se las conoce, están en un momento de auge y controversia; por un lado los que apoyan este movimiento enfatizan en su gran performance, velocidad y en la libertad que proveen para organizar los datos, la escalabilidad, etc. Mientras que los seguidores de las tradicionales bases de datos relacionales se basan en argumentos como la falta de formalidad y reglas para definir modelos y la necesidad de definir de antemano los querys necesarios para tu aplicación, entre otros.

Lo cierto es que cada paradigma tiene sus ventajas y desventajas frente a ciertos problemas como todas las tecnologías y la decisión de cual usar reside en los diseñadores, arquitectos y/o programadores de sistemas; cada uno evaluará para su realidad qué tecnología usar teniendo en cuenta sus requerimientos.

Además, son tecnologías compatibles, es decir, una no es reemplazo de la otra y por lo tanto puedes obtener lo mejor de los dos mundos si logras que en tu aplicación convivan ambas en sintonía, cada una para lo que mejor fue concebida.

Ahora bien, si te has decidido por una base de datos NoSql, hemos confeccionado esta pequeña reseña de las bases más populares, lo que queremos mostrar es un panorama general de las base de datos NoSql que actualmente se destacan.

Características principales de las bases de datos NoSQL:

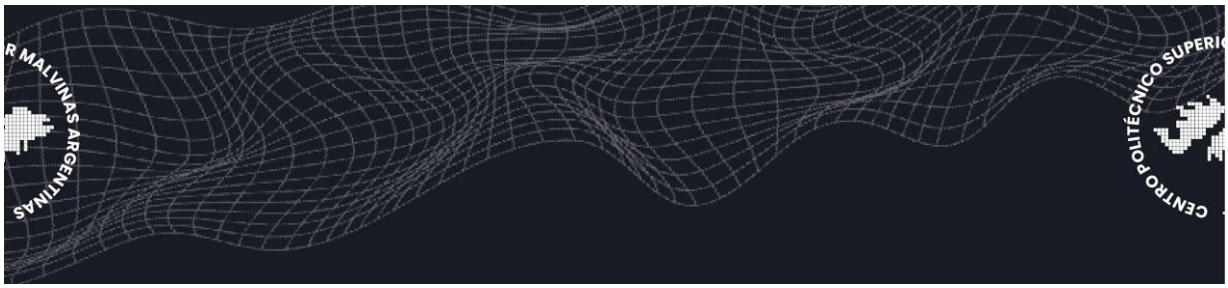
Flexibilidad de modelado de datos: Permiten almacenar datos en estructuras no tabulares, como documentos JSON, grafos o conjuntos de claves-valores.

Escalabilidad horizontal: Pueden distribuirse fácilmente en múltiples servidores para manejar grandes volúmenes de datos.

Alto rendimiento: Ofrecen velocidades de lectura y escritura rápidas, ideales para aplicaciones en tiempo real.

Simplicidad: Su diseño suele ser más simple que las bases de datos relacionales, lo que facilita su implementación y mantenimiento.

Comparativa de bases de datos NoSQL



Modelo de datos	Características	Tipo de aplicaciones	Ejemplos
Clave-Valor **Columnas** Variante de clave-valor que permite más de un valor (columna) por clave.	• Muy alto rendimiento. • Muy escalable. • Útil para representar datos no estructurados. • No existe el concepto de relaciones	Aplicaciones que busca alto rendimiento en las consultas, que precisen de alta escalabilidad y no necesiten implementar relaciones entre sus datos.	• Cassandra • Redis • HBase • Mencached • Riak • MariaDB
Documentos XML, JSON o BSON.	• Almacenan datos de tipo documento (los documentos representan estructuras clave valor anidadas) • Se representan en formato XML, JSON o BSON. • Flexible en esquemas de datos dinámicos. • Reducción de la complejidad en la consultas para datos asociados.	Aplicaciones que preceden de esquemas cambiantes y necesiten flexibilidad.	• MongoDB • Couchbase • Amazon_Dynamo • CouchDB • RethinkDB • RavenDB • Cloudant • GemFire
Grafos **Atributos:** Nodos con propiedades. **Aristas**: relaciones.	• Los datos se modelan como un conjunto de relaciones entre elementos. • Alto rendimiento en consultas de relaciones de proximidad entre datos, y no para ejecutar consultas globales. • Flexibilidad en la definición de atributos y longitud de registros.	Redes sociales, software de recomendación, aplicaciones de geolocalización, aplicaciones de optimización de rutas, topologías de red …	• Neo4j • Titan • DEX/Sparksee • AllegroGraph • OrientDB • InfiniteGraph • Sones GraphDB • InfoGrid • HyperGraphDB

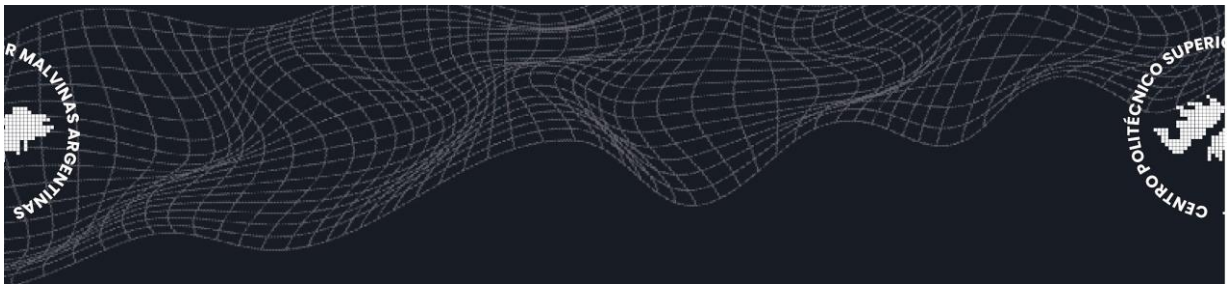
Tipos de bases de datos NoSQL:

- Bases de datos de documentos: Almacenan datos en forma de documentos JSON, como MongoDB y CouchDB.

- Bases de datos clave-valor: Asocian claves únicas con valores arbitrarios, como Redis y DynamoDB.

- Bases de datos de grafos: Representan relaciones entre entidades mediante nodos y aristas, como Neo4j y OrientDB.

- Bases de datos NoSQL especializadas: Diseñadas para casos de uso específicos, como bases de datos de tiempo series (InfluxDB) o bases de



datos triples (Apache Jena).

Casos de uso comunes para bases de datos NoSQL:

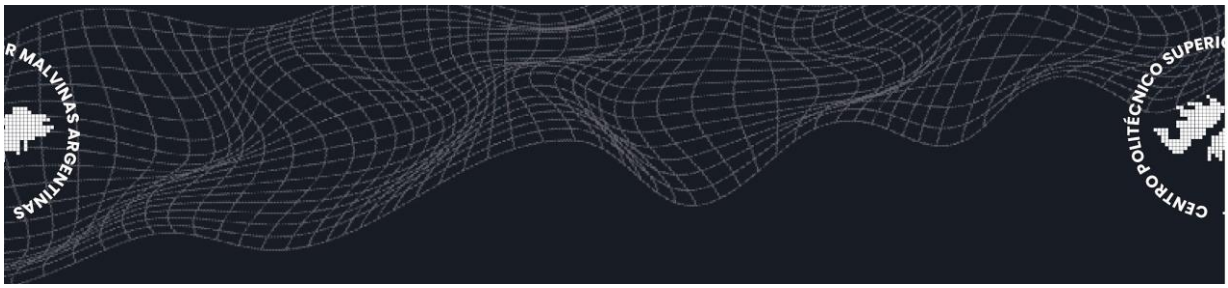
- Aplicaciones web a gran escala: Manejan grandes volúmenes de datos de usuarios y transacciones.
- Aplicaciones móviles: Almacenan datos de manera eficiente en dispositivos con recursos limitados.
- Redes sociales: Gestionan grandes conjuntos de datos de usuarios, publicaciones y conexiones.
- IoT (Internet de las cosas): Procesan y analizan datos en tiempo real de sensores y dispositivos.

Ventajas y desventajas de las bases de datos NoSQL:

Las bases de datos NoSQL (Not Only SQL) ofrecen una serie de ventajas y desventajas en comparación con las bases de datos relacionales tradicionales. A continuación, se presentan algunas de las ventajas y desventajas más comunes:

Ventajas de las bases de datos NoSQL:

1. **Escalabilidad horizontal:** Las bases de datos NoSQL están diseñadas para escalar horizontalmente, lo que significa que pueden manejar grandes volúmenes de datos distribuyendo la carga de trabajo en múltiples servidores. Esto permite un mejor rendimiento y la capacidad de manejar cargas de trabajo crecientes de manera eficiente.
2. **Flexibilidad de esquema**: Las bases de datos NoSQL permiten almacenar datos con estructuras flexibles y variables. No están limitadas por un esquema fijo como las bases de datos relacionales, lo que facilita la adaptación a cambios en los requisitos y la manipulación de datos no estructurados.
3. **Rendimiento y velocidad:** Las bases de datos NoSQL pueden ofrecer un rendimiento y velocidad excepcionales en operaciones de lectura y escritura, especialmente en escenarios de alta concurrencia. Al eliminar algunas restricciones impuestas por las bases de datos relacionales, pueden optimizar el rendimiento para casos de uso específicos.
4. **Alta disponibilidad y tolerancia a fallos**: Las bases de datos NoSQL están



diseñadas para garantizar la disponibilidad continua en entornos distribuidos. Utilizan técnicas como la replicación y el particionamiento para garantizar la redundancia de datos y la recuperación ante fallos.

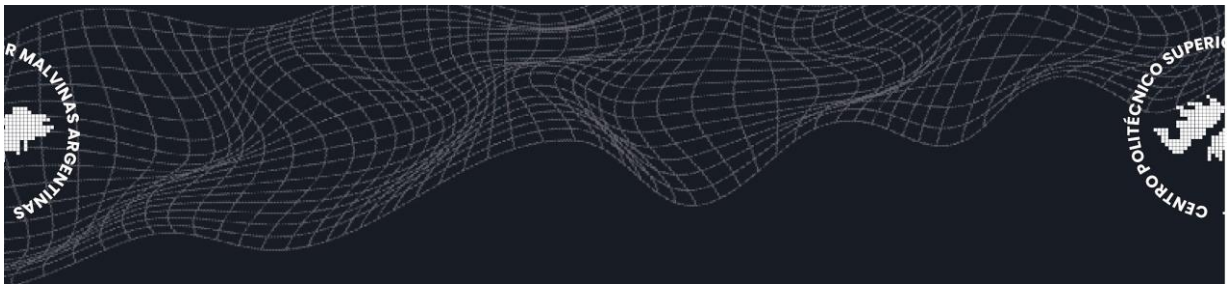
Desventajas de las bases de datos NoSQL:

1. **Menor consistencia:** Algunas bases de datos NoSQL sacrifican la consistencia estricta a cambio de una mayor escalabilidad y rendimiento. Esto significa que, en ciertos casos, los datos pueden no estar inmediatamente disponibles en todos los nodos de la base de datos, lo que puede generar resultados inconsistentes en ciertas situaciones.
2. **Menor soporte de consultas complejas:** A diferencia de las bases de datos relacionales, las bases de datos NoSQL pueden tener limitaciones en cuanto a las consultas complejas que se pueden realizar. Algunas bases de datos NoSQL están optimizadas para consultas simples y rápidas, pero pueden no ser adecuadas para escenarios que requieren operaciones complejas de agregación o combinación de datos.
3. **Menor madurez y herramientas limitadas**: Algunas bases de datos NoSQL son relativamente nuevas en comparación con las bases de datos relacionales tradicionales, lo que significa que pueden tener una menor madurez y un conjunto de herramientas más limitado. Esto puede dificultar el desarrollo, la administración y el mantenimiento de bases de datos NoSQL en comparación con las bases de datos relacionales bien establecidas.

Es importante tener en cuenta que las ventajas y desventajas de las bases de datos NoSQL pueden variar según el tipo específico de base de datos NoSQL y los requisitos del caso de uso. Es recomendable evaluar cuidadosamente las características y limitaciones de cada sistema antes de decidir utilizar una base de datos NoSQL en un proyecto determinado.

Ventajas:

- Flexibilidad y escalabilidad.
- Alto rendimiento y velocidad.
- Simplicidad y facilidad de uso.
- Adecuadas para datos no estructurados.



Desventajas:

- Menor consistencia de datos en comparación con las bases de datos relacionales.
- Complejidad en consultas complejas.
- Menor madurez y ecosistema de herramientas.

Actividad 1

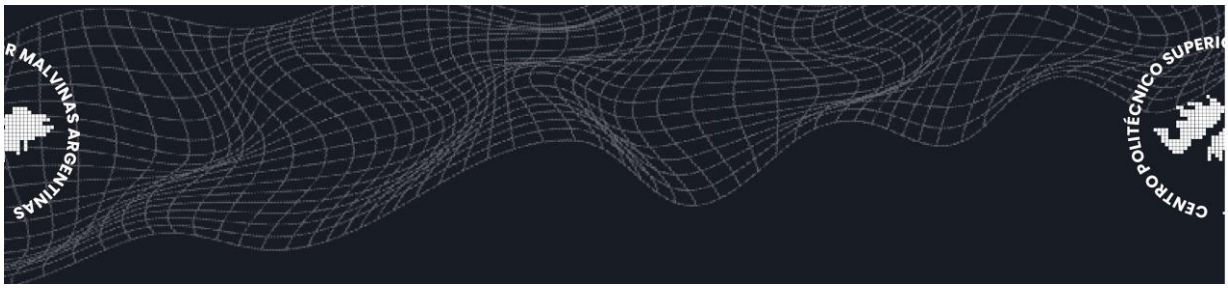
Ejercicio práctico 1: Consultas en una base de datos NoSQL de documentos

Objetivo: Realizar consultas básicas en una base de datos NoSQL de documentos y obtener los resultados esperados.

Pasos:

Utilizar una base de datos NoSQL de documentos como MongoDB.
Crear una colección llamada "usuarios" con los siguientes documentos:
Documento 1:

```
json
{
  "_id": 1,
  "nombre": "Juan",
  "edad": 25,
  "ciudad": "Buenos Aires"
}
```



Documento 2:

```
json
{
  "_id": 2,
  "nombre": "María",
  "edad": 30,
  "ciudad": "Madrid"
}
```

Documento 3:

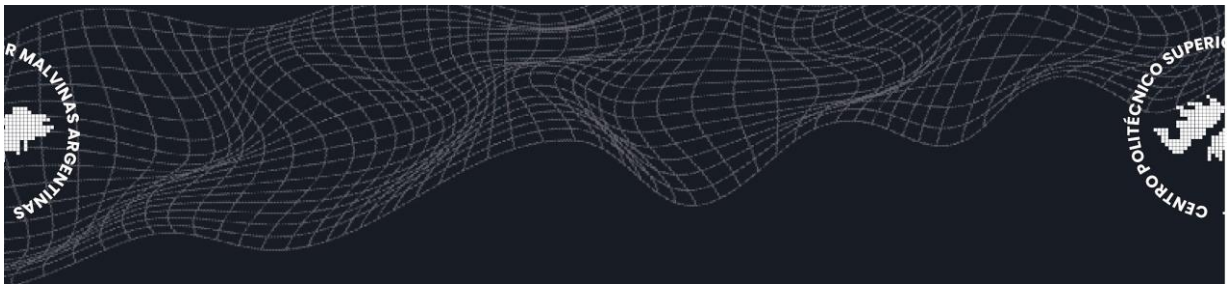
```
json
{
  "_id": 3,
  "nombre": "Carlos",
  "edad": 28,
  "ciudad": "Santiago"
}
```

Realizar las siguientes consultas y obtener los resultados esperados:

Consulta 1: Obtener todos los usuarios.

Resultado esperado:

```
json
[
  {
    "_id": 1,
    "nombre": "Juan",
    "edad": 25,
    "ciudad": "Buenos Aires"
  },
  {
    "_id": 2,
    "nombre": "María",
    "edad": 30,
    "ciudad": "Madrid"
  },
  {
    "_id": 3,
    "nombre": "Carlos",
    "edad": 28,
    "ciudad": "Santiago"
  }
]
```



Consulta 2: Obtener el usuario con el nombre "María".

Resultado esperado:

```
json
{
  "_id": 2,
  "nombre": "María",
  "edad": 30,
  "ciudad": "Madrid"
}
```

Consulta 3: Obtener los usuarios mayores de 26 años.
Resultado esperado:

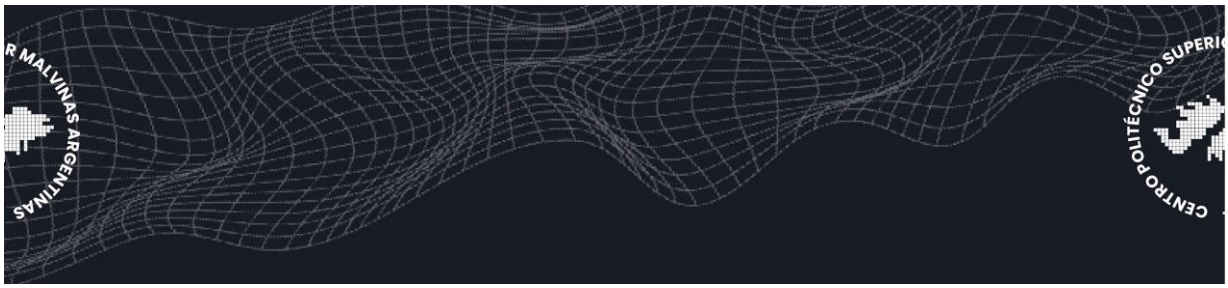
```
json
[
  {
    "_id": 2,
    "nombre": "María",
    "edad": 30,
    "ciudad": "Madrid"
  },
  {
    "_id": 3,
    "nombre": "Carlos",
    "edad": 28,
    "ciudad": "Santiago"
  }
  ]
```

Ejercicio práctico 2: Modelado de datos en una base de datos NoSQL de grafos

Objetivo: Diseñar y ejecutar consultas utilizando una base de datos NoSQL de grafos para modelar relaciones entre entidades.
Pasos:
Utilizar una base de datos NoSQL de grafos como Neo4j.
Crear nodos para representar personas y relaciones para representar amistades entre ellas.
Crear los siguientes nodos:
Persona 1:

```
(:Persona {nombre: 'Juan'})
```

Persona 2:
Copiar



(:Persona {nombre: 'María'})

Persona 3:

(:Persona {nombre: 'Carlos'})

Crear las siguientes relaciones de amistad:

Juan y María son amigos:

(juan:Persona)-[:AMIGO]->(maria:Persona)

María y Carlos son amigos:

(maria:Persona)-[:AMIGO]->(carlos:Persona)

Realizar las siguientes consultas y obtener los resultados esperados:

Consulta 1: Obtener todos los amigos de Juan.

Resultado esperado:

```
[
  {
    "nombre": "María"
  }
]
```

Consulta 2: Obtener todos los amigos de María.

Resultado esperado:

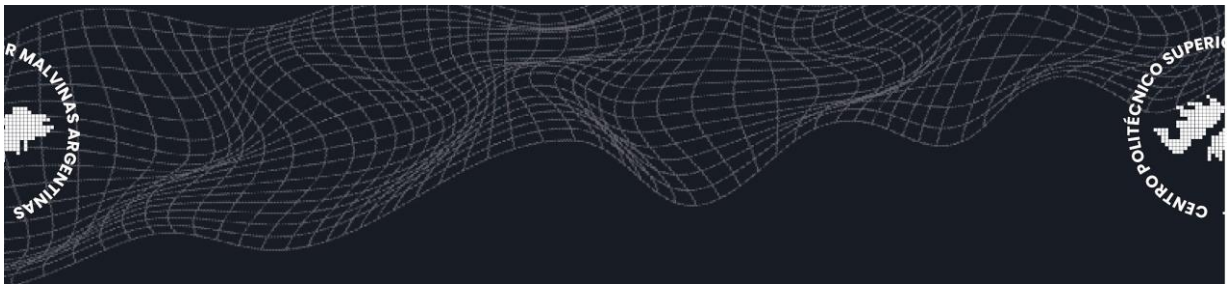
json

```
[
  {
    "nombre": "Juan"
  },
  {
    "nombre": "Carlos"
  }
```

3. Actividad Integradora de Cierre:

Actividad Integradora: Diseño y desarrollo de una aplicación utilizando bases de datos NoSQL

Objetivo: Aplicar los conocimientos adquiridos sobre bases de datos NoSQL en el diseño y desarrollo de una aplicación práctica.



Pasos:

1. Formar equipos de desarrollo, cada uno compuesto por varios participantes.

2. Definir el objetivo de la aplicación que se va a desarrollar. Por ejemplo, una aplicación de gestión de tareas o un sistema de recomendación de películas.

3. Elegir una base de datos NoSQL adecuada para satisfacer los requisitos de la aplicación. Pueden considerar opciones como MongoDB, Cassandra, Neo4j o Redis, según el tipo de datos y las consultas necesarias.

4. Realizar el diseño del esquema de datos para la aplicación. Esto implica identificar las entidades y relaciones relevantes y decidir cómo se almacenarán en la base de datos NoSQL seleccionada.

5. Implementar la aplicación utilizando un lenguaje de programación y un framework de desarrollo web de su elección.

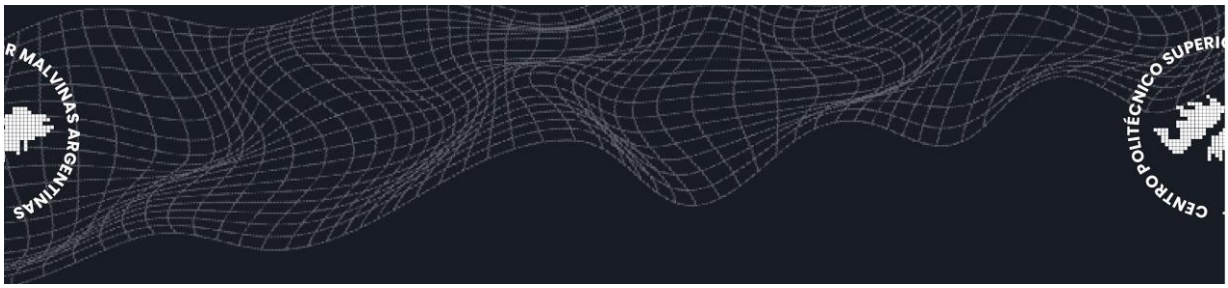
6. Integrar la base de datos NoSQL en la aplicación y desarrollar las funcionalidades necesarias para interactuar con ella, como la inserción, consulta, actualización y eliminación de datos.

7. Realizar pruebas exhaustivas para asegurar el correcto funcionamiento de la aplicación y la integración con la base de datos NoSQL.

8. Documentar el proceso de diseño y desarrollo, incluyendo el esquema de datos, las decisiones tomadas y cualquier desafío o solución destacada encontrada durante el proceso.

9. Preparar una presentación final en la que cada equipo muestre y explique su aplicación, el diseño de su esquema de datos y las lecciones aprendidas durante el desarrollo.

10. Fomentar la discusión y el intercambio de ideas entre los equipos, así como realizar preguntas y comentarios constructivos sobre cada proyecto.



4. <u>Cierre:</u>

Las bases de datos NoSQL han revolucionado el panorama del almacenamiento de datos, ofreciendo alternativas flexibles y escalables para aplicaciones modernas. Si bien no son un reemplazo directo de las bases de datos relacionales, ofrecen un conjunto único de características que las convierten en una opción atractiva para una amplia gama de casos de uso.

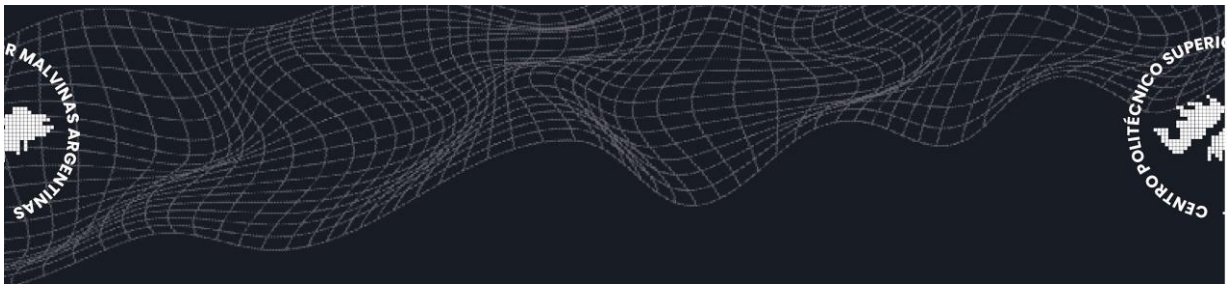
<u>5.-Bibliografía Obligatoria:</u>

1. "NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence" by Pramod J. Sadalage and Martin Fowler.

    - Este libro proporciona una introducción clara y concisa a las bases de datos NoSQL, explicando los diferentes tipos de bases de datos NoSQL y sus casos de uso. Es una lectura recomendada para comprender los conceptos fundamentales y las decisiones de diseño asociadas con las bases de datos NoSQL.

2. "MongoDB: The Definitive Guide" by Kristina Chodorow and Shannon Bradshaw.

    - Enfocado específicamente en MongoDB, uno de los sistemas de bases de datos NoSQL más populares, este libro abarca aspectos desde la instalación hasta el modelado de datos y las operaciones avanzadas. Es una lectura esencial para aquellos que deseen adquirir un conocimiento profundo de MongoDB.



3. "Cassandra: The Definitive Guide" by Jeff Carpenter and Eben Hewitt.

    - Este libro es una guía completa sobre Apache Cassandra, una base de datos de columnas amplias. Proporciona una visión general detallada de la arquitectura, el modelado de datos, las operaciones y la escalabilidad de Cassandra. Es una referencia imprescindible para aquellos interesados en utilizar Cassandra como su base de datos NoSQL.

4. "Graph Databases" by Ian Robinson, Jim Webber, and Emil Eifrem.

    - Este libro explora los fundamentos de las bases de datos de grafos, incluido el modelo de datos de grafos y las consultas de grafos. Se centra en Neo4j, una base de datos de grafos ampliamente utilizada, y ofrece una guía práctica para diseñar y utilizar bases de datos de grafos efectivamente.

5. "Redis in Action" by Josiah L. Carlson.

    - Redis es una base de datos NoSQL de clave-valor extremadamente rápida y versátil. Este libro cubre una amplia gama de casos de uso de Redis, incluidos cachés, colas de mensajes y análisis en tiempo real. Proporciona información detallada sobre cómo utilizar Redis de manera efectiva en aplicaciones prácticas.