BASE DE DATOS

1º AÑO

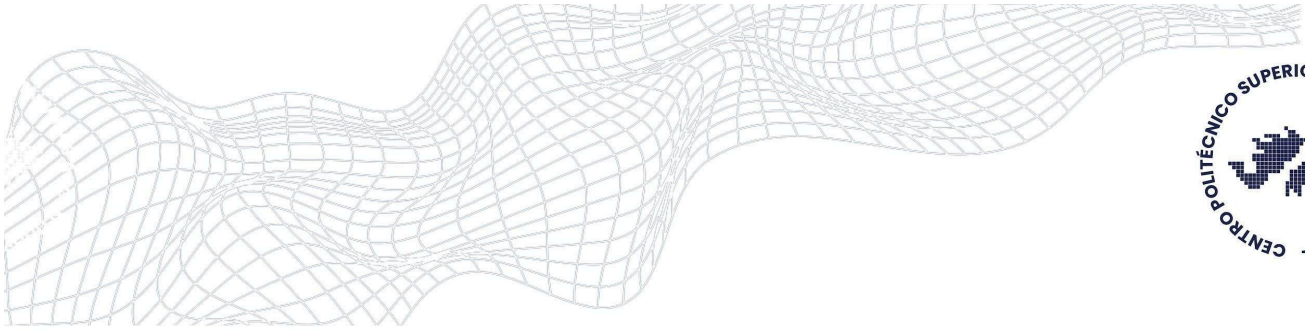
CLASE Nº 9

INTRODUCCIÓN A NoSQL: EXPLORANDO LAS BASE DE DATOS NO RELACIONALES

CONTENIDO

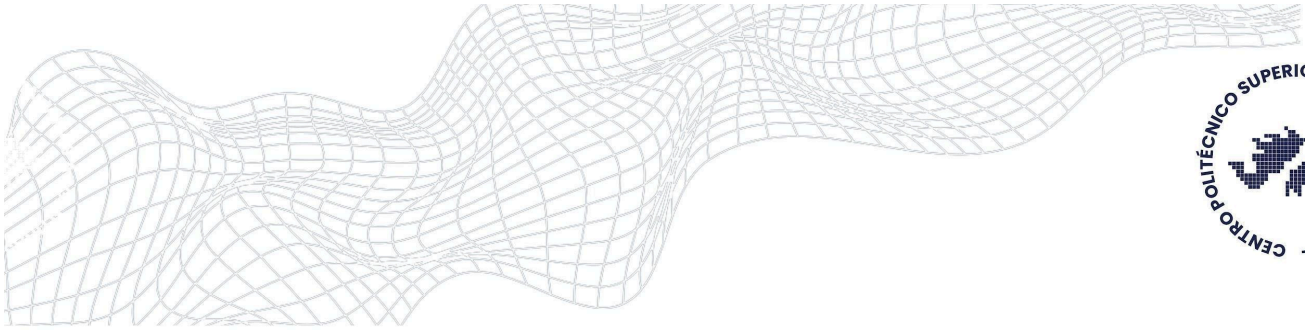
En la clase de hoy trabajaremos los siguientes temas:

- **¿Qué son las bases de datos NoSQL?**
    - Definición y concepto fundamental.
    - Diferencias clave con las bases de datos relacionales (SQL).
- **Características principales de las bases de datos NoSQL**
    - Escalabilidad horizontal.
    - Flexibilidad de esquema.
    - Disponibilidad y rendimiento.
    - Modelos de datos diversos.
- **Tipos de bases de datos NoSQL**
    - Bases de datos Clave-Valor.
    - Bases de datos Documentales (Document-oriented).



- Bases de datos Orientadas a Columnas (Column-family).
- Bases de datos de Grafos (Graph databases).

- **Casos de uso comunes para bases de datos NoSQL**
  - Aplicaciones en tiempo real (ej. redes sociales, juegos).
  - Big Data y análisis de datos.
  - Internet de las Cosas (IoT).
  - Sistemas de gestión de contenido.
- **Ventajas y desventajas de las bases de datos NoSQL**
  - **Ventajas:** Escalabilidad, flexibilidad, rendimiento para ciertos tipos de datos.
  - **Desventajas:** Consistencia eventual, madurez de las herramientas, curva de aprendizaje.



1. PRESENTACIÓN

Presentación de la Clase N° 9: Introducción a Bases de Datos NoSQL

¡Bienvenidos a la clase N° 9 de la materia **BASE DE DATOS**!

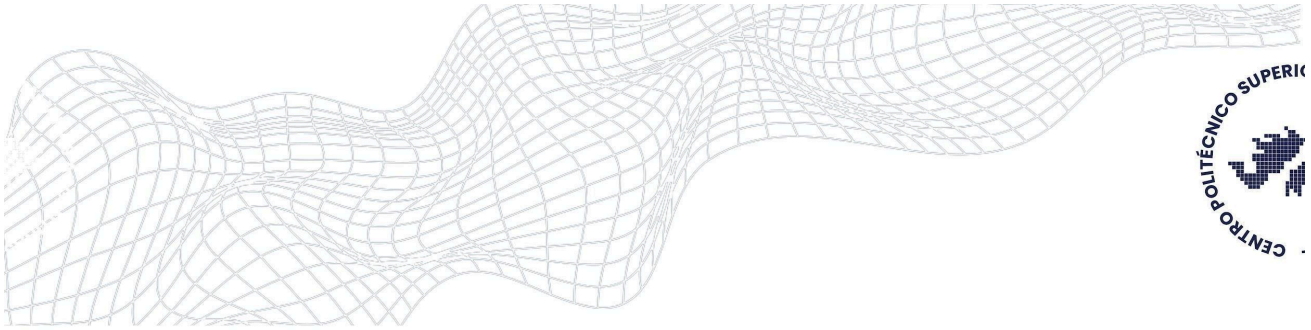
Hoy nos embarcamos en un fascinante viaje para explorar el universo de las **bases de datos NoSQL**. En esta sesión introductoria, desvelaremos los conceptos fundamentales de este moderno paradigma de almacenamiento de datos. Analizaremos sus **características distintivas** y nos adentraremos en los **diversos tipos de bases de datos NoSQL** que se utilizan en la industria actual. Prepárense para comprender cómo estas bases de datos están transformando la forma en que gestionamos la información en el mundo digital.

2. DESARROLLO

¿Qué son las Bases de Datos NoSQL?

Las bases de datos **NoSQL**, también conocidas como **bases de datos no relacionales**, representan una evolución en los sistemas de gestión de bases de datos (SGBD) que se alejan del modelo relacional tradicional. A diferencia de las bases de datos SQL, que estructuran los datos en tablas rígidas con filas y columnas, las bases de datos NoSQL ofrecen una **mayor flexibilidad y escalabilidad** para almacenar y consultar datos diversos y en constante cambio.

Actualmente, las bases de datos NoSQL gozan de gran auge y generan debate. Por un lado, sus defensores destacan su **alto rendimiento, velocidad y la libertad** que ofrecen



para organizar los datos, así como su capacidad de **escalabilidad horizontal**. Por otro lado, los seguidores de las bases de datos relacionales tradicionales argumentan la falta de formalidad y reglas para definir modelos, y la necesidad de predefinir las consultas en algunas implementaciones NoSQL.

Lo cierto es que cada paradigma tiene sus **ventajas y desventajas** frente a problemas específicos, como ocurre con todas las tecnologías. La decisión sobre cuál utilizar recae en los diseñadores, arquitectos y programadores de sistemas, quienes deben evaluar cuidadosamente los requisitos de cada proyecto.

Además, es crucial entender que estas tecnologías son **compatibles**. Es decir, una no reemplaza a la otra. De hecho, es posible obtener lo mejor de ambos mundos si logran que ambas convivan en sintonía dentro de una aplicación, utilizando cada una para lo que mejor fue concebida.

Si han decidido explorar el potencial de una base de datos NoSQL para un proyecto particular, a continuación, presentamos una reseña de las bases de datos NoSQL más populares, brindando un panorama general de las opciones destacadas en el panorama actual.

Características Principales de las Bases de Datos NoSQL

Las bases de datos NoSQL se distinguen por varias características clave que las hacen atractivas para ciertos escenarios:

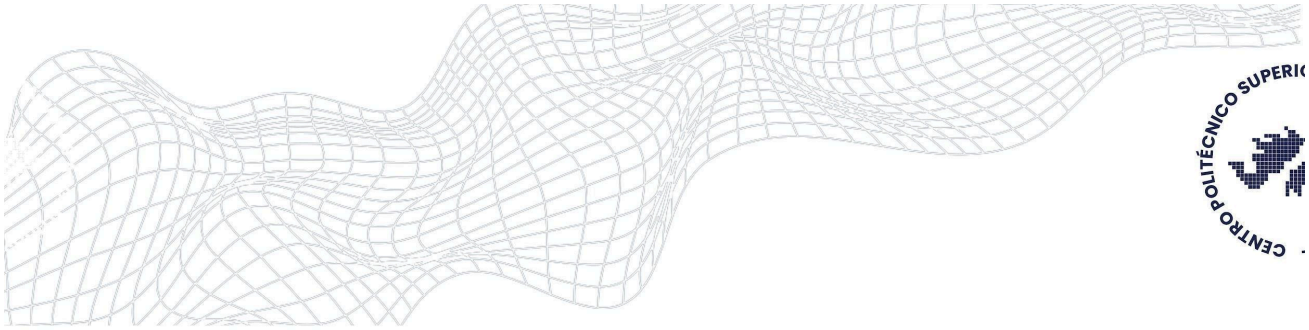
- **Flexibilidad de modelado de datos:** Permiten almacenar datos en estructuras no tabulares, como documentos JSON, grafos o conjuntos de claves-valores, adaptándose a esquemas dinámicos y cambiantes.
- **Escalabilidad horizontal:** Su diseño facilita la distribución de datos y la carga de trabajo en múltiples servidores, permitiendo manejar volúmenes masivos de información de manera eficiente.
- **Alto rendimiento:** Ofrecen velocidades de lectura y escritura rápidas, lo que las hace ideales para aplicaciones que requieren baja latencia y alta concurrencia.
- **Simplicidad:** En general, su diseño es más simple que el de las bases de datos relacionales, lo que puede facilitar su implementación y mantenimiento en ciertos contextos.

Comparativa de Bases de Datos NoSQL por Modelo de Datos

Para comprender mejor los diferentes tipos de bases de datos NoSQL, es útil compararlos según su modelo de datos, características principales, aplicaciones típicas y ejemplos concretos:

Modelo de datos	**Características**	**Tipo de aplicaciones**	**Ejemplos**
Clave-Valor **Columnas** Variante de clave-valor que permite más de un valor	• Muy alto rendimiento. • Muy escalable. • Útil para representar datos no estructurados.	Aplicaciones que busca alto rendimiento en las consultas, que precisen de alta escalabilidad y no necesiten	• Cassandra • Redis • HBase • Mencached • Riak • MariaDB

(columna) por clave.	• No existe el concepto de relaciones	implementar relaciones entre sus datos.	
<u>Documentos</u> 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
<u>Grafos</u> **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

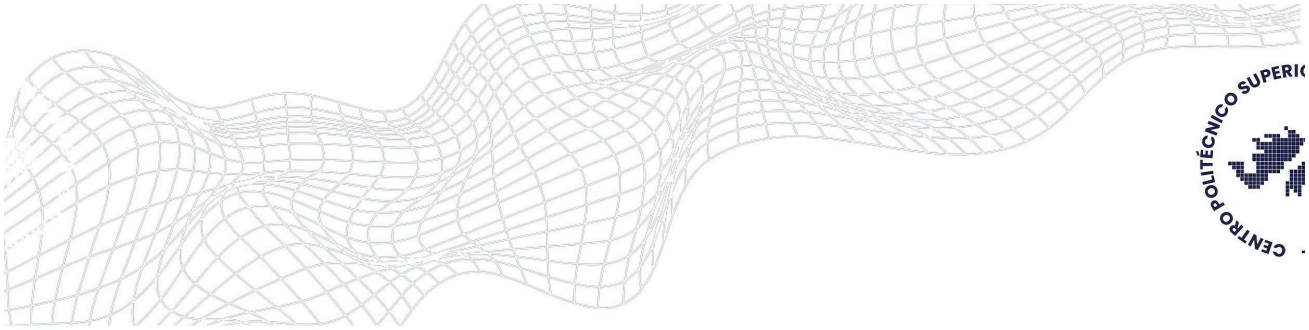
Complementando la tabla anterior, podemos clasificar los tipos principales de bases de datos NoSQL de la siguiente manera:

- **Bases de datos de documentos:** Almacenan datos en forma de documentos **JSON, XML o BSON**. Ejemplos populares incluyen **MongoDB** y **CouchDB**.
- **Bases de datos clave-valor:** Asocian **claves únicas con valores arbitrarios**. Son ideales para almacenar datos simples y de acceso rápido. Ejemplos incluyen **Redis** y **Amazon DynamoDB**.
- **Bases de datos de grafos:** Representan **relaciones entre entidades** mediante nodos y aristas. Destacan **Neo4j** y **OrientDB**.
- **Bases de datos NoSQL especializadas:** Diseñadas para casos de uso muy específicos, como bases de datos de series de tiempo (**InfluxDB**) o bases de datos triples (**Apache Jena**).

Casos de Uso Comunes para Bases de Datos NoSQL

Las bases de datos NoSQL son particularmente adecuadas para los siguientes escenarios:

- **Aplicaciones web a gran escala:** Permiten manejar enormes volúmenes de datos de usuarios, sesiones y transacciones con alta concurrencia.
- **Aplicaciones móviles:** Facilitan el almacenamiento eficiente de datos en dispositivos con recursos limitados y la sincronización con servidores.



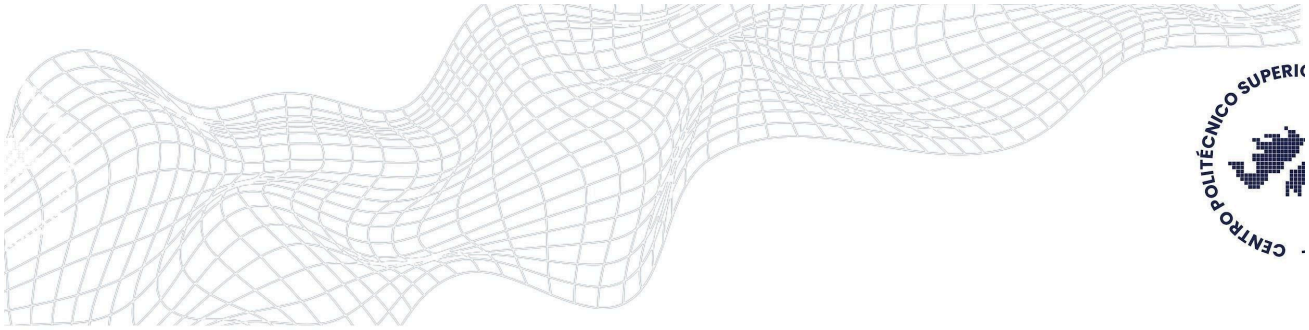
- **Redes sociales:** Ideales para gestionar grandes conjuntos de datos de usuarios, publicaciones, conexiones y la complejidad de las interacciones.
- **IoT (Internet de las Cosas):** Procesan y analizan datos en tiempo real provenientes de sensores y dispositivos conectados.
- **Big Data y Análisis de Datos:** Facilitan el almacenamiento y procesamiento de volúmenes masivos de datos no estructurados o semiestructurados para fines analíticos.

Ventajas y Desventajas de las Bases de Datos NoSQL

Las bases de datos NoSQL ofrecen una serie de ventajas y desventajas en comparación con las bases de datos relacionales tradicionales. Es crucial entenderlas para tomar decisiones informadas.

Ventajas de las bases de datos NoSQL:

- **Escalabilidad Horizontal:** Diseñadas para distribuir la carga de trabajo en múltiples servidores, permitiendo manejar grandes volúmenes de datos y cargas crecientes de manera eficiente.
- **Flexibilidad de Esquema:** Permiten almacenar datos con estructuras flexibles y variables. No están limitadas por un esquema fijo, lo que facilita la adaptación a cambios en los requisitos y la manipulación de datos no estructurados o semiestructurados.
- **Rendimiento y Velocidad:** Ofrecen un rendimiento y velocidad excepcionales en operaciones de lectura y escritura, especialmente en escenarios de alta concurrencia, al optimizar el rendimiento para casos de uso específicos.



- **Alta Disponibilidad y Tolerancia a Fallos:** Generalmente diseñadas para garantizar la disponibilidad continua en entornos distribuidos, utilizando técnicas como la replicación y el particionamiento para la redundancia de datos y la recuperación ante fallos.

Desventajas de las bases de datos NoSQL:

- **Menor Consistencia Estricta:** Algunas bases de datos NoSQL sacrifican la consistencia estricta (el principio ACID: Atomicidad, Consistencia, Aislamiento, Durabilidad) a cambio de una mayor escalabilidad y rendimiento. Esto puede significar que, en ciertos casos, los datos pueden no estar inmediatamente disponibles en todos los nodos de la base de datos, lo que podría generar resultados inconsistentes en situaciones particulares (se basan en el modelo BASE: Basically Available, Soft state, Eventually consistent).
- **Menor Soporte de Consultas Complejas:** A diferencia de las bases de datos relacionales con su potente SQL, las bases de datos NoSQL pueden tener limitaciones en cuanto a la complejidad de las consultas que se pueden realizar. Algunas están optimizadas para consultas simples y rápidas, pero pueden no ser adecuadas para escenarios que requieren operaciones complejas de agregación, uniones o combinación de datos entre diferentes colecciones o tablas.
- **Menor Madurez y Ecosistema de Herramientas Limitado:** Si bien están creciendo rápidamente, algunas bases de datos NoSQL son relativamente nuevas en comparación con las bases de datos relacionales. Esto puede traducirse en una

menor madurez del software, un ecosistema de herramientas más limitado y menos recursos de aprendizaje o soporte comunitario en algunos casos.

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

Actividades Prácticas: Explorando Bases de Datos NoSQL

Actividad 1: Consultas Fundamentales en una Base de Datos NoSQL de Documentos

Objetivo: Desarrollar la habilidad de realizar consultas básicas en una base de datos NoSQL de documentos y validar los resultados obtenidos.

Para este ejercicio, utilizaremos **MongoDB**, uno de los sistemas de gestión de bases de datos de documentos más populares.

Pasos a seguir:

1. **Configuración del entorno:** Asegúrense de tener una instancia de MongoDB en funcionamiento (local o en la nube). Podrán interactuar con ella a través de la **MongoDB Shell (mongosh)** o una interfaz gráfica como **MongoDB Compass**.

2. **Creación de la colección y los documentos:** Vamos a crear una colección llamada `usuarios` e insertar los siguientes documentos. Estos documentos representan información básica de usuarios y serán la base de nuestras consultas.

JSON

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

JavaScript

```
db.usuarios.insertMany([
  { "_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" }
]);
```

JSON

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

Comando de inserción (ejemplo en mongosh):

Realización de consultas y verificación de resultados: Ejecuten las siguientes consultas y comparen los resultados con los esperados.

- **Consulta 1: Obtener todos los usuarios.**
    - **Comando (ejemplo en mongosh):**

```
JavaScript
db.usuarios.find({});
```

    - Resultado esperado:

JSON

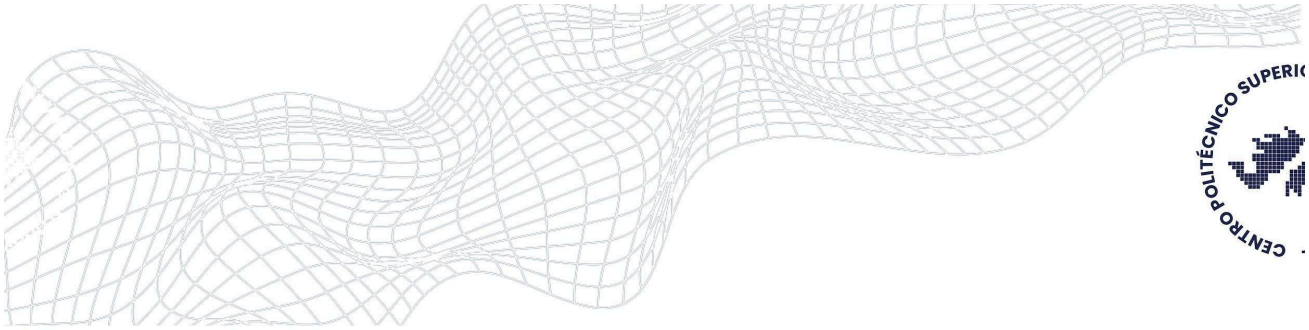
```
[
  {
    "_id": 1,
    "nombre": "Juan",
    "edad": 25,
    "ciudad": "Buenos Aires"
  },
  {
    "_id": 2,
    "nombre": "María",
    "edad": 30,
    "ciudad": "Madrid"
  },
  {
    "_id": 3,
    "nombre": "Carlos",
    "edad": 28,
    "ciudad": "Santiago"
  }
]
```

- **Consulta 2: Obtener el usuario con el nombre "María".**
    - **Comando (ejemplo en mongosh):**

JavaScript

```
db.usuarios.findOne({ "nombre": "María" });
```

- Resultado esperado:



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

Consulta 3: Obtener los usuarios mayores de 26 años.

- **Comando (ejemplo en mongosh):**

```JavaScript
db.usuarios.find({ "edad": { "$gt": 26 } });
```

Resultado esperado:

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

Actividad 2: Modelado y Consultas en una Base de Datos NoSQL de Grafos

Objetivo: Comprender el modelado de datos en un paradigma de grafos y ejecutar consultas para explorar relaciones entre entidades.

Para este ejercicio, utilizaremos **Neo4j**, una base de datos de grafos líder en el mercado.

Pasos a seguir:

1. **Configuración del entorno:** Asegúrense de tener una instancia de Neo4j en funcionamiento (local o en la nube, por ejemplo, a través de Neo4j Desktop o AuraDB). Podrán interactuar con ella utilizando el **Neo4j Browser**.

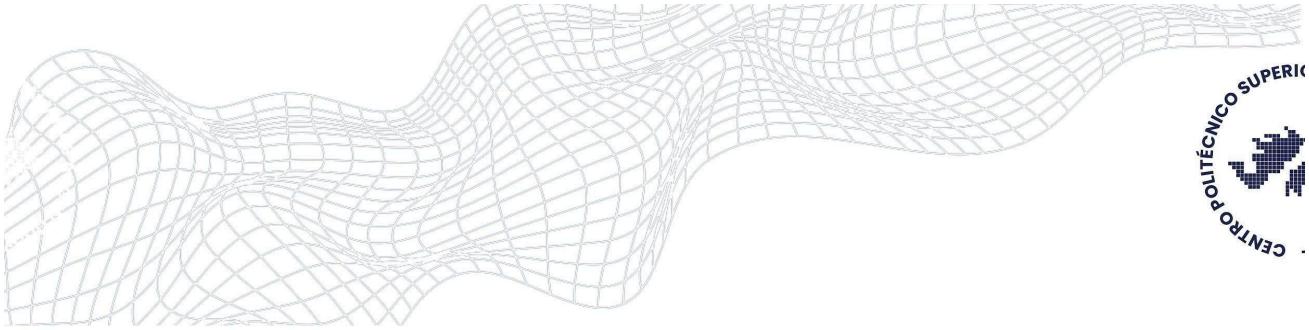
2. **Creación de nodos y relaciones:** Crearemos nodos para representar personas y definiremos relaciones de amistad entre ellas. El lenguaje de consulta para grafos es **Cypher**.

    - **Crear los siguientes nodos de personas:**

```
Cypher

CREATE (:Persona {nombre: 'Juan'})
CREATE (:Persona {nombre: 'María'})
CREATE (:Persona {nombre: 'Carlos'})
```

- Crear las siguientes relaciones de amistad:



```
Cypher

MATCH (juan:Persona {nombre: 'Juan'}), (maria:Persona {nombre: 'María'})
CREATE (juan)-[:AMIGO]->(maria)
```

```
Cypher

MATCH (maria:Persona {nombre: 'María'}), (carlos:Persona {nombre: 'Carlos'}
CREATE (maria)-[:AMIGO]->(carlos)
```

(Nota: En Cypher, las relaciones pueden ser direccionales, como [:AMIGO]->, o bidireccionales si se crean ambas direcciones o se considera una relación no direccional para la consulta.)

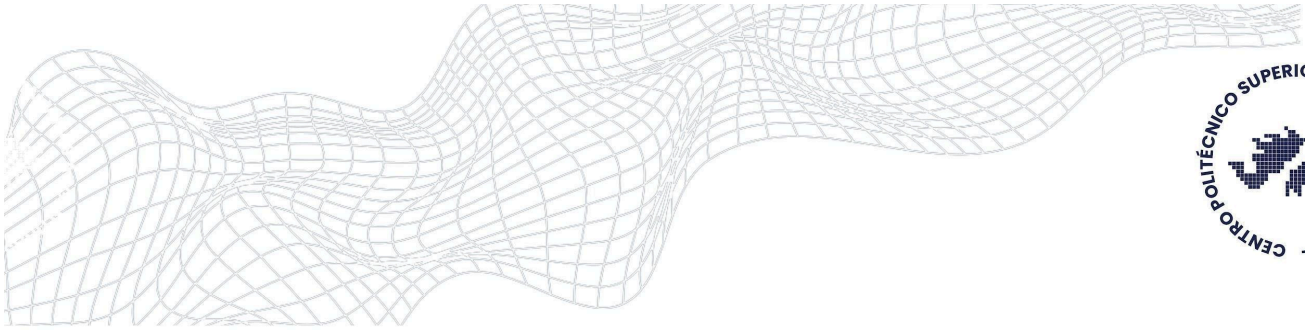
3. Realización de consultas y verificación de resultados: Ejecuten las siguientes consultas y comparen los resultados con los esperados.

- **Consulta 1: Obtener todos los amigos de Juan.**
    - **Comando (ejemplo en Neo4j Browser):**

```
Cypher

MATCH (:Persona {nombre: 'Juan'})-[:AMIGO]->(amigo:Persona)
RETURN amigo.nombre AS nombre
```

- Resultado esperado:



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

Consulta 2: Obtener todos los amigos de María.

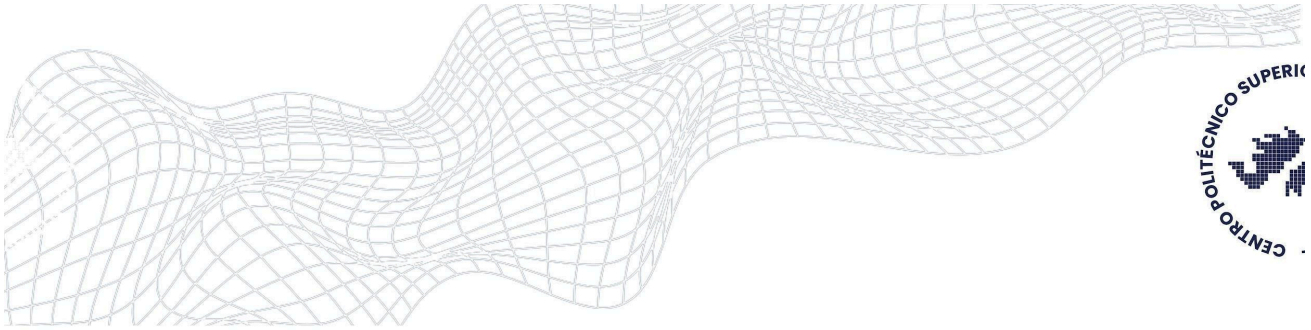
- **Comando (ejemplo en Neo4j Browser):**

```cypher
MATCH (maria:Persona {nombre: 'María'})-[rel:AMIGO]-(amigo:Persona)
RETURN amigo.nombre AS nombre
```

(Nota: Hemos usado `-[rel:AMIGO]-(amigo:Persona)` para buscar relaciones en cualquier dirección, ya que la amistad suele ser recíproca. Si solo hubiéramos querido los que María 'apunta', sería `->`)

- Resultado esperado:

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



3. Actividad Integradora de Cierre: Diseño y Desarrollo de una Aplicación con Bases de Datos NoSQL

Objetivo: Consolidar los conocimientos adquiridos sobre bases de datos NoSQL mediante el diseño y desarrollo de una aplicación práctica que demuestre su aplicación real.

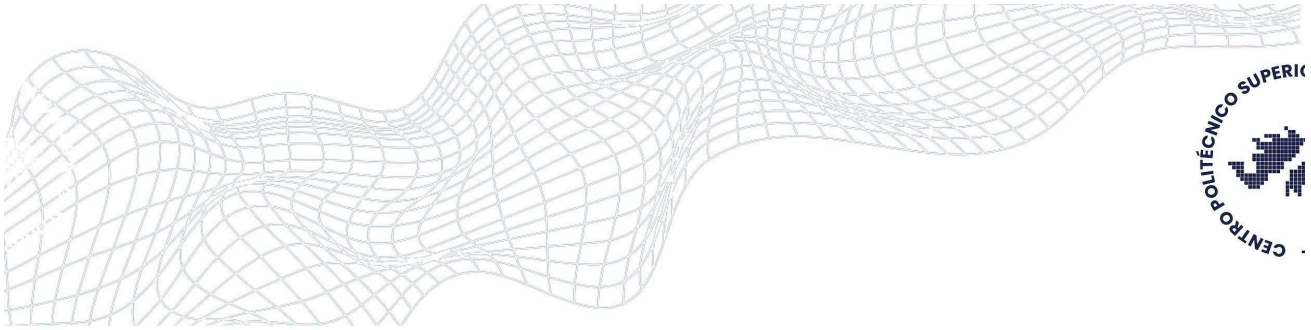
Esta actividad los desafiará a aplicar lo aprendido de forma colaborativa, simulando un entorno de desarrollo profesional.

Pasos a seguir:

1. **Formación de Equipos:** Organicen equipos de desarrollo, fomentando la diversidad de roles y habilidades entre los participantes.

2. **Definición del Alcance de la Aplicación:** Cada equipo deberá definir claramente el objetivo y la funcionalidad principal de una aplicación. Piensen en escenarios donde las bases de datos NoSQL demuestren su valor, como una **aplicación de gestión de tareas con flexibilidad de esquemas**, un **sistema de recomendación de películas basado en relaciones entre usuarios y contenido**, o una **plataforma de mensajería en tiempo real**.

3. **Selección de la Base de Datos NoSQL:** Basándose en los requisitos y el tipo de datos de su aplicación, elijan la base de datos NoSQL más adecuada. Consideren opciones como **MongoDB** para datos documentales, **Cassandra** para escalabilidad masiva y alta disponibilidad, **Neo4j** para modelado de relaciones



complejas, o **Redis** para caching y datos en memoria. Justifiquen su elección.

4. **Diseño del Esquema de Datos:** Realicen un diseño detallado del esquema de datos para su aplicación dentro del modelo NoSQL seleccionado. Esto implica identificar las entidades clave, sus atributos y, lo más importante, cómo se almacenarán y relacionarán (o desnormalizar) eficientemente en su base de datos NoSQL elegida.

5. **Implementación de la Aplicación:** Desarrollen la aplicación utilizando un lenguaje de programación y un *framework* de desarrollo web de su elección (por ejemplo, Node.js con Express, Python con Flask/Django, Java con Spring Boot, etc.).

6. **Integración y Desarrollo de Funcionalidades:** Conecten su aplicación con la base de datos NoSQL seleccionada. Implementen las funcionalidades esenciales que permitan la interacción con la base de datos, incluyendo la **inserción (creación), consulta (lectura), actualización y eliminación (CRUD)** de datos según las necesidades de su aplicación.

7. **Pruebas Rigurosas:** Realicen pruebas exhaustivas para asegurar el correcto funcionamiento de la aplicación y una integración fluida con la base de datos NoSQL. Identifiquen y corrijan cualquier error o inconsistencia.

8. **Documentación del Proceso:** Documenten de forma clara y concisa todo el proceso de diseño y desarrollo. Incluyan:

    - El esquema de datos final y su justificación.
    - Las decisiones clave tomadas en cada etapa (elección de la DB, modelado, etc.).
    - Los desafíos técnicos encontrados y cómo fueron superados.
    - Cualquier solución innovadora o aprendizaje significativo.
9. **Presentación Final:** Preparen una presentación impactante donde cada equipo mostrará su aplicación en funcionamiento, explicará el diseño de su esquema de datos NoSQL y compartirá las lecciones aprendidas durante el proceso de desarrollo.

10. **Sesión de Intercambio:** Fomentaremos una sesión de discusión y retroalimentación constructiva entre los equipos. Aprovechen esta oportunidad para hacer preguntas, compartir ideas y ofrecer comentarios valiosos sobre cada proyecto.

---

4. Cierre de la Clase: Reflexiones Finales sobre Bases de Datos NoSQL

Las **bases de datos NoSQL** han emergido como un pilar fundamental en el panorama moderno del almacenamiento de datos, ofreciendo soluciones excepcionalmente

flexibles y escalables para las demandas de las aplicaciones actuales. Es crucial entender que no pretenden ser un reemplazo directo de las bases de datos relacionales tradicionales. Más bien, se presentan como un **complemento poderoso**, aportando un conjunto único de características que las posiciona como una opción altamente atractiva para una vasta diversidad de casos de uso.

Hemos explorado sus diferentes modelos, ventajas y desventajas, y cómo se aplican en escenarios reales. La clave del éxito en el diseño de sistemas de bases de datos reside en la **elección informada** y la capacidad de **integrar diversas tecnologías** cuando sea necesario.

erramos nuestra **Clase 9: Introducción a NoSQL, explorando las bases de datos no relacionales**. Ha sido un viaje fascinante por un paradigma diferente de gestión de datos, y para consolidar lo aprendido, les presento su **primera autoevaluación**.

Sobre la autoevaluación

Esta actividad es una herramienta diseñada para que **reflexionen sobre su propio proceso de aprendizaje y crecimiento**. No la vean como un examen, sino como una **oportunidad invaluable para conocerse mejor como estudiantes**, identificar sus **fortalezas** y reconocer las **áreas en las que pueden seguir mejorando**.

Para que esta autoevaluación sea realmente útil, les pido que:

- **Presten mucha atención**: Lean cada pregunta con detenimiento, asegurándose de comprender lo que se les solicita.
- **Sean sinceros**: Respondan con total honestidad. La sinceridad es clave para que obtengan una visión clara y precisa de su progreso.
- **Tómense su tiempo**: No se apresuren. Reflexionen sobre cada pregunta y respondan de manera reflexiva y consciente.

Confío en que realizarán un excelente trabajo y que esta experiencia les será de gran utilidad.

Instancia de autoevaluación

La autoevaluación estará disponible en el **campus virtual** y se realizará a través de un **cuestionario en Moodle.**

- **Intentos**: Cuentan con dos intentos.
- **Aprobación**: Se aprueba con una calificación de 6 (seis) o más.
- **Habilitación**: Estará disponible desde el **viernes 23 de mayo de 2025 a las 18:00 hs**.
- **Cierre**: La fecha límite para completarla es el **Domingo 1 de junio de 2025 a las 23:00 hs**.

¡Muchos éxitos a todos en esta autoevaluación!

Bibliografía Obligatoria Recomendada:

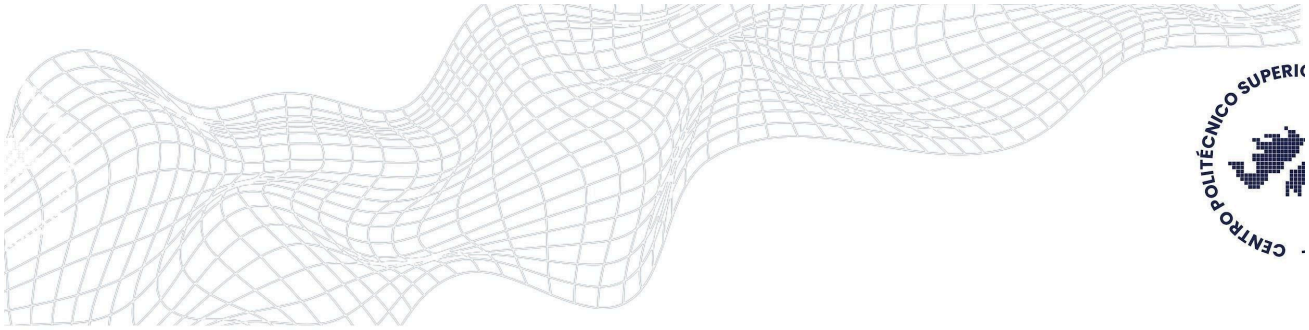
A continuación, se presenta una selección de recursos bibliográficos esenciales para profundizar en el estudio de las bases de datos NoSQL, organizados desde un nivel introductorio para principiantes hasta temas más avanzados y específicos.

Nivel Introductorio y General:

- **"NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence"** por Pramod J. Sadalage y Martin Fowler.
    - Este libro ofrece una **introducción concisa y altamente accesible** al universo de las bases de datos NoSQL. Explica de manera clara los diferentes tipos de bases de datos no relacionales, sus modelos de datos fundamentales y los escenarios en los que brillan. Es una lectura indispensable para cualquier estudiante que desee comprender los **conceptos clave y las implicaciones de diseño** al trabajar con NoSQL.

Bases de Datos NoSQL Específicas (Documentales y Clave-Valor):

- **"MongoDB: The Definitive Guide"** por Kristina Chodorow y Shannon Bradshaw.
    - Enfocado en **MongoDB**, una de las bases de datos NoSQL de documentos más populares, esta guía exhaustiva cubre desde la instalación y configuración inicial hasta el **modelado de datos avanzado** y las



operaciones cotidianas. Es un recurso esencial para quienes busquen dominar esta tecnología.

- **"Redis in Action"** por Josiah L. Carlson.
    - Este libro explora **Redis**, una base de datos NoSQL de clave-valor excepcionalmente rápida y versátil. Detalla una amplia gama de casos de uso prácticos, incluyendo la implementación de **cachés de alto rendimiento, colas de mensajes y análisis en tiempo real**. Proporciona una visión profunda sobre cómo aprovechar eficazmente Redis en diversas aplicaciones.

Bases de Datos NoSQL Específicas (Columnar y de Grafos):

- **"Cassandra: The Definitive Guide"** por Jeff Carpenter y Eben Hewitt.
    - Una guía completa sobre **Apache Cassandra**, una robusta base de datos de columnas amplias, ideal para entornos de Big Data y alta disponibilidad. Este libro ofrece una visión detallada de su arquitectura, estrategias de **modelado de datos, operaciones de cluster y principios de escalabilidad**. Es una referencia imprescindible para aquellos interesados en implementar soluciones de bases de datos distribuidas a gran escala con Cassandra.
- **"Graph Databases"** por Ian Robinson, Jim Webber y Emil Eifrem.
    - Este texto explora los **fundamentos de las bases de datos de grafos**, incluyendo su poderoso modelo de datos y las técnicas de consulta específicas para relaciones complejas. Se centra en **Neo4j**, una base de datos de grafos ampliamente utilizada, y ofrece una guía práctica para

diseñar y utilizar este tipo de bases de datos de manera efectiva en escenarios como redes sociales o sistemas de recomendación.