

# Big Data: Conceptos y Tecnologías

Big Data es un concepto que se refiere al manejo y análisis de conjuntos de datos tan grandes y complejos que no pueden ser gestionados eficientemente con herramientas tradicionales. Su relevancia ha crecido exponencialmente debido al aumento de la generación de datos desde diversas fuentes: redes sociales, dispositivos IoT, sensores, transacciones digitales, entre otros.

No hay una definición formal para el volumen de datos necesario para ser considerado dentro de Big Data. Lo importante es entender que este concepto va más allá de un software o una tecnología específica, representando un nuevo paradigma en el manejo de información a gran escala.

 **por Kibernetum Capacitación S.A.**



# Preguntas sobre Big Data y Sistemas Distribuidos



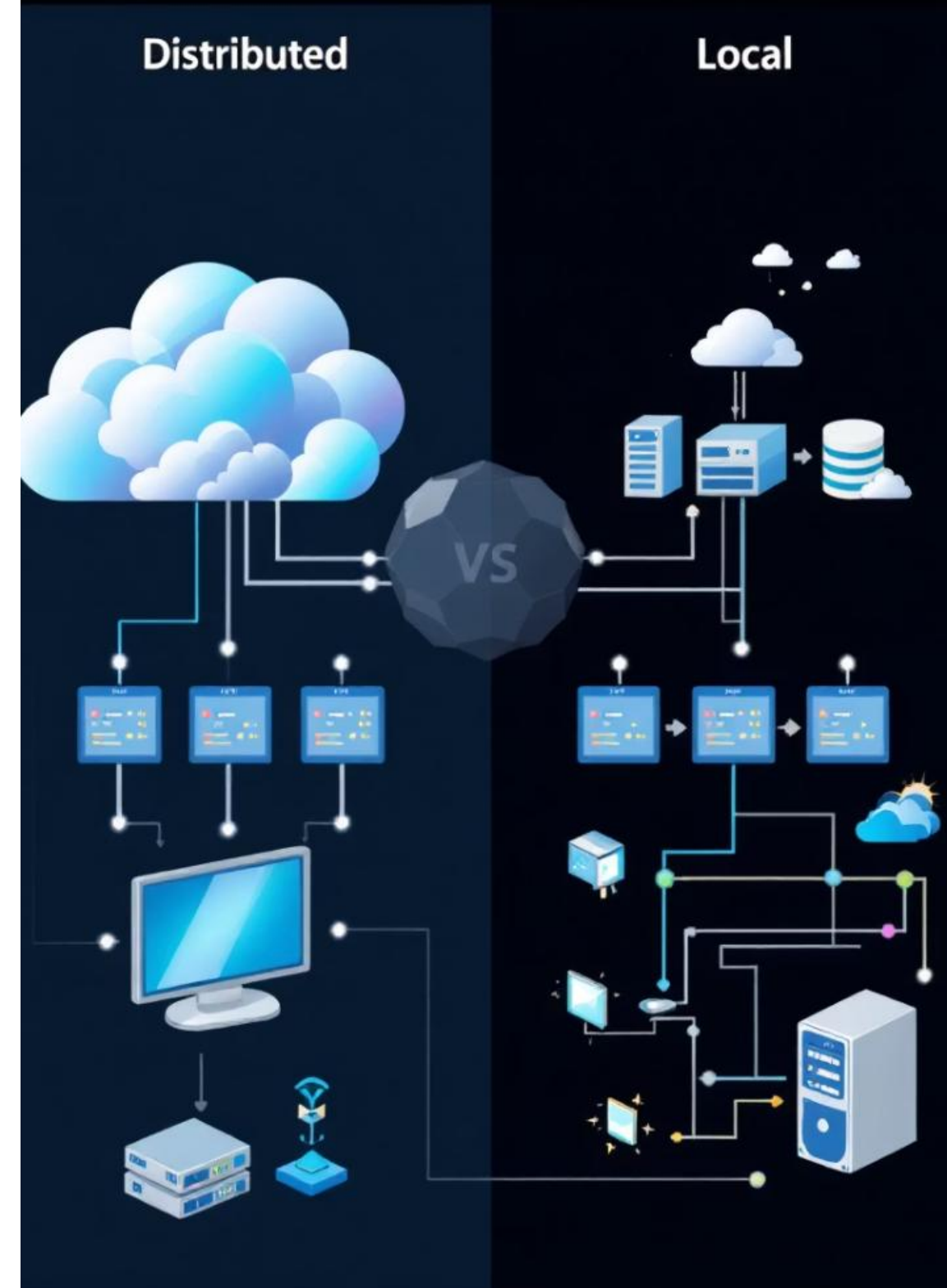
¿Has oído hablar del término "Big Data"? ¿Qué crees que significa o a qué tipo de datos se refiere?



¿Qué ejemplos conoces de empresas o servicios que podrían estar usando Big Data en su funcionamiento diario?



¿Sabes qué diferencias hay entre un sistema local (como el de tu computador) y un sistema distribuido? ¿Por qué crees que algunas empresas prefieren usar sistemas distribuidos?



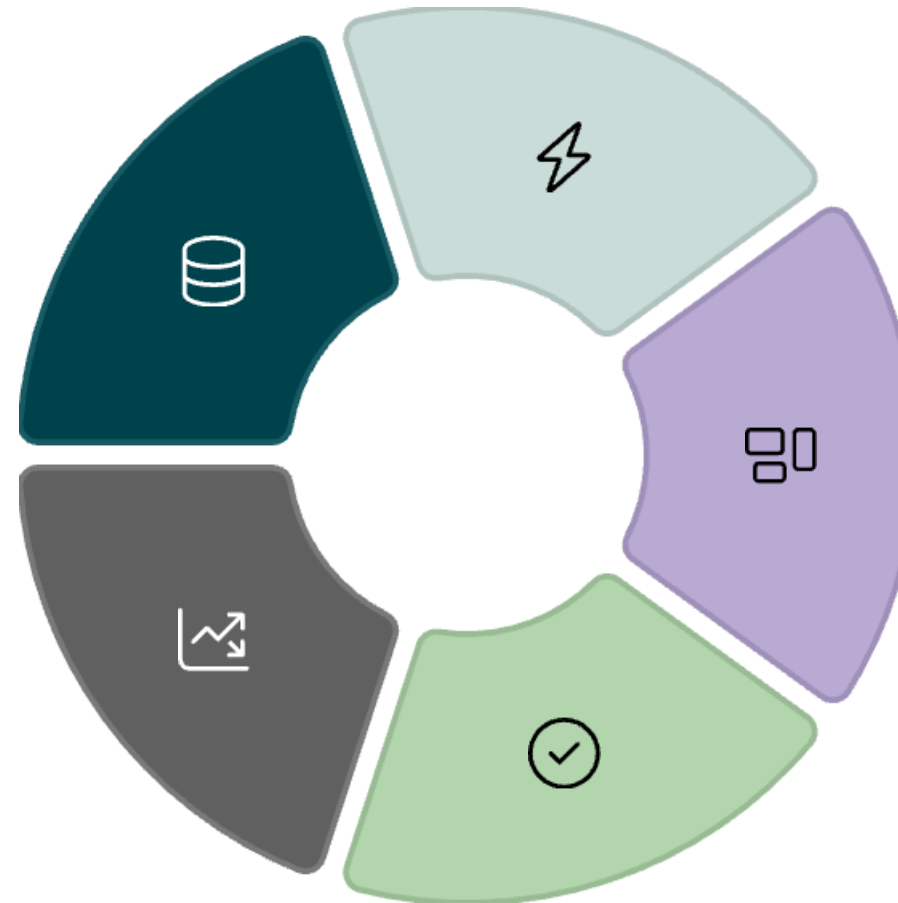
# Las 5V's de Big Data

## Volumen

Cantidad masiva de datos generados continuamente desde múltiples fuentes.

## Valor

Utilidad y beneficios que pueden extraerse de los datos mediante análisis adecuados.



## Velocidad

Rapidez con la que los datos se generan, procesan y analizan en tiempo real.

## Variedad

Diferentes formatos de datos (texto, imagen, video, sensores) tanto estructurados como no estructurados.

## Veracidad

Calidad y confiabilidad de los datos para asegurar análisis precisos.





# Problemas que resuelve Big Data

## **Análisis de sentimientos**

Permite analizar millones de comentarios en redes sociales para determinar la percepción pública sobre marcas, productos o eventos.

## **Mantenimiento predictivo**

Anticipa fallos en maquinaria industrial mediante el análisis de datos de sensores, como temperatura, presión y velocidad.

## **Recomendaciones personalizadas**

Plataformas como Netflix o Amazon utilizan Big Data para ofrecer sugerencias basadas en comportamientos previos de usuarios.

## **Prevención de fraude**

Detecta patrones sospechosos en transacciones financieras en tiempo real para evitar actividades fraudulentas.





# Evolución de las Tecnologías Big Data



## **Etapas 1: Sistemas Relacionales**

Basados en el modelo relacional y lenguaje SQL, con escalabilidad vertical (mejoras de hardware) y alta dependencia de estructuras fijas de datos.



## **Etapas 2: NoSQL y Hadoop**

Aparecen sistemas como MongoDB, Cassandra, y Hadoop con HDFS y MapReduce, con soporte para datos no estructurados y distribución del procesamiento.



## **Etapas 3: Procesamiento en memoria**

Emergen frameworks como Apache Spark, que permiten procesamiento distribuido en memoria, mucho más rápido que MapReduce, junto con soluciones de streaming como Apache Kafka.

# Sistemas Locales vs. Distribuidos

## Sistemas Locales

Un sistema local se basa en un solo servidor o computadora que ejecuta todo el procesamiento y almacenamiento de datos. Son adecuados para volúmenes de datos pequeños o moderados.

- Todo el cómputo ocurre en una única máquina
- Limitados por la capacidad física del hardware
- Riesgo de punto único de falla
- Menor complejidad de configuración

## Sistemas Distribuidos

Los sistemas distribuidos reparten la carga de procesamiento y almacenamiento entre varios nodos interconectados. Son ideales para manejar grandes volúmenes de datos.

- Procesamiento paralelo en múltiples nodos
- Mayor tolerancia a fallos
- Alta escalabilidad y rendimiento
- Mayor complejidad en configuración

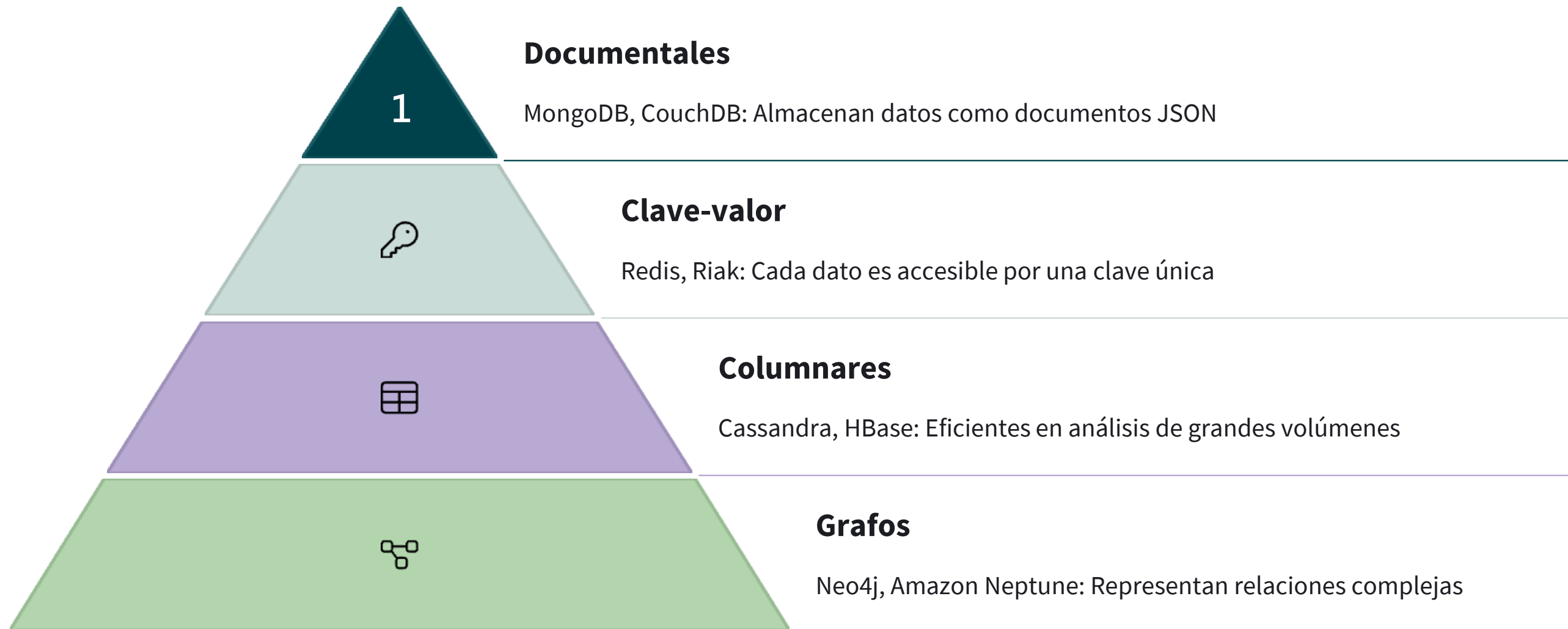
# Tecnologías de Big Data

En un mundo donde los datos crecen a un ritmo vertiginoso, las tecnologías tradicionales de almacenamiento y procesamiento se han vuelto insuficientes. La solución a esta problemática ha venido de la mano del ecosistema de tecnologías Big Data, diseñadas para manejar altos volúmenes, velocidades y variedades de datos en distintos entornos: corporativos, industriales, financieros, científicos, entre otros.

Estas tecnologías no solo permiten almacenar grandes volúmenes de datos, sino también analizarlos en tiempo real, extraer conocimiento valioso y tomar decisiones de forma rápida y escalable.



# Tecnologías NoSQL



NoSQL ("Not Only SQL") es un conjunto de bases de datos no relacionales que surgen como alternativa a las limitaciones de las bases de datos SQL tradicionales. A diferencia de las bases de datos relacionales, que requieren esquemas fijos y estructuras tabulares, las NoSQL permiten almacenar y consultar datos en estructuras más flexibles.



# Tecnologías NoSQL

## Escenarios de uso:

- Aplicaciones web de alta concurrencia.
- Almacenamiento de logs y registros de sensores.
- Análisis de redes sociales y patrones de conexión.
- Recomendadores personalizados (productos, contenidos).

## Estructura general de un documento NoSQL (MongoDB):

```
{  
  "usuario": "jhondoe",  
  "correo": "jhondoe@email.com",  
  "compras": [  
    {"producto": "Laptop", "precio": 950},  
    {"producto": "Mouse", "precio": 25}  
  ]  
}
```

## Ventajas principales:

- Escalabilidad horizontal.
- Flexibilidad para manejar datos estructurados, semiestructurados o no estructurados.
- Mayor rendimiento en consultas específicas.

## Escenarios de uso:

- Aplicaciones web con millones de usuarios.
- Sistemas de recomendación en tiempo real.
- Almacenamiento de registros de sensores IoT.

# Procesamiento en Tiempo Real

El procesamiento en tiempo real se refiere a la capacidad de analizar, transformar y actuar sobre los datos **al mismo tiempo que son generados**. A diferencia del procesamiento batch, que trabaja con lotes de datos almacenados previamente, el enfoque en tiempo real permite obtener insights inmediatos y ejecutar acciones automáticas.

## Problemas que resuelve:



### Detección de fraudes financieros

Identificar transacciones sospechosas en milisegundos para prevenir actividades fraudulentas.



### Sistemas de recomendación dinámica

Actualizar sugerencias de productos o contenidos en tiempo real basados en comportamiento del usuario.



### Monitoreo industrial e IoT

Responder a alertas o anomalías en sensores para prevenir fallos en equipos críticos.



### Análisis de redes sociales

Detectar tendencias, hashtags o eventos virales al momento para responder oportunamente.

# Procesamiento en Tiempo Real

## Tecnologías destacadas:

- ✓ Apache Kafka: sistema de mensajería distribuida para ingesta masiva de datos.
- ✓ Apache Flink: procesamiento de streams con muy baja latencia y tolerancia a fallos.
- ✓ Apache Storm: arquitectura ligera para procesamiento de flujos en tiempo real.
- ✓ Apache Spark Structured Streaming: integración poderosa con el ecosistema Spark para análisis en memoria.
- ✓ Amazon Kinesis / Google Cloud Dataflow: opciones de streaming en la nube.
- ✓ Ejemplos de uso:
- ✓ Detección de fraudes bancarios.
- ✓ Monitoreo de redes sociales.
- ✓ Sistemas de alerta en salud y transporte.





# Almacenamiento Distribuido



## División de datos

Los datos se dividen en fragmentos distribuidos entre múltiples nodos



## Replicación

Copias redundantes aseguran disponibilidad y tolerancia a fallos



## Acceso paralelo

Múltiples usuarios pueden leer y escribir datos simultáneamente



## Escalabilidad

Capacidad de añadir más nodos para incrementar el almacenamiento

El almacenamiento distribuido es una arquitectura que permite guardar datos en múltiples nodos o servidores conectados en red, en lugar de centralizar toda la información en un único lugar. Esta estrategia es esencial en entornos Big Data, donde los volúmenes de información superan las capacidades de los sistemas tradicionales.

# Almacenamiento Distribuido

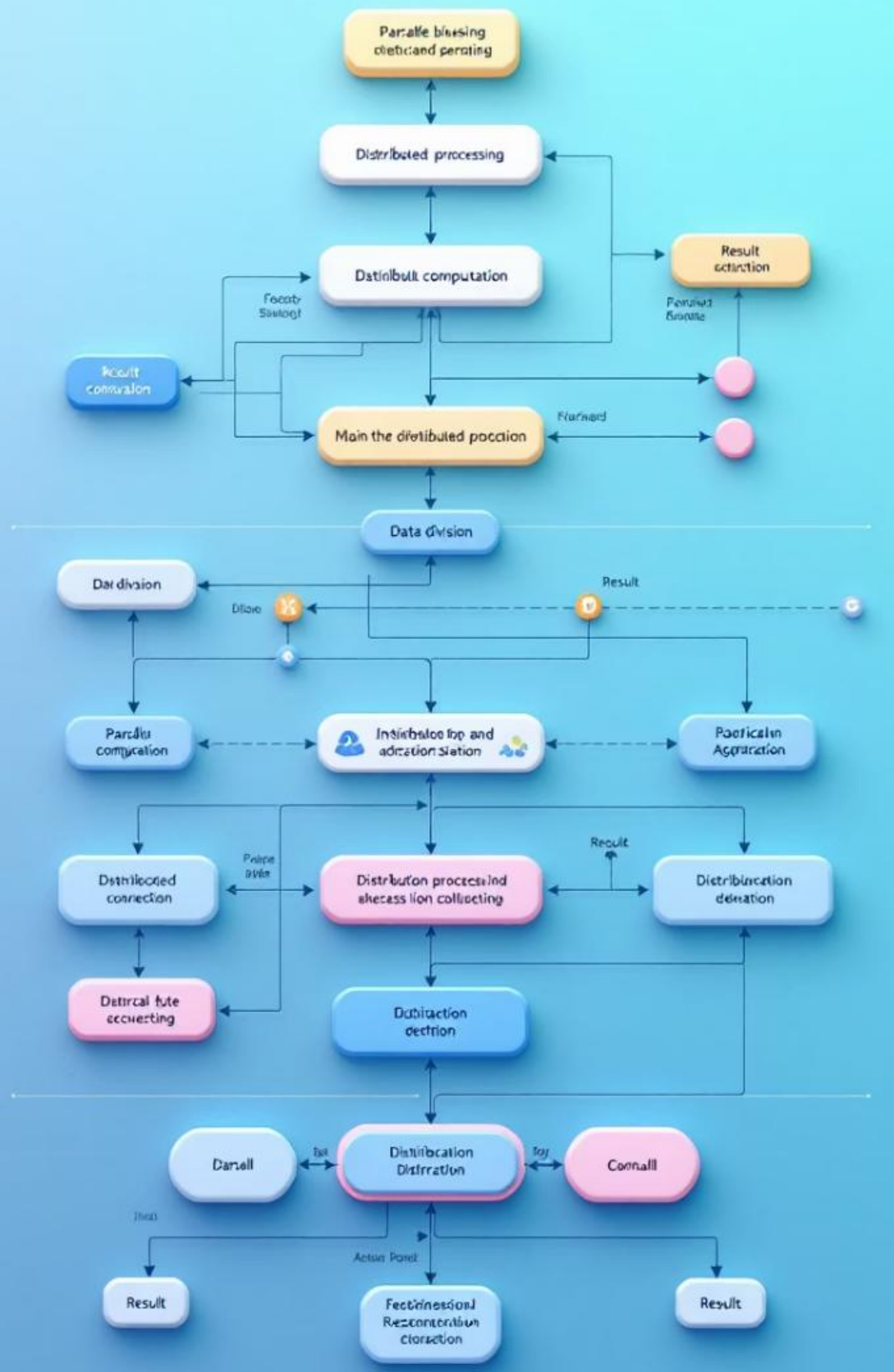
## Casos de uso:

- Plataformas de streaming de video (Netflix, YouTube).
- Sistemas de respaldo y almacenamiento empresarial.
- Almacenamiento de datos históricos de sensores IoT.
- Análisis de logs de navegación o transacciones bancarias.

## Tecnologías disponibles:

- HDFS (Hadoop Distributed File System): Arquitectura de almacenamiento tolerante a fallos, núcleo del ecosistema Hadoop.
- Amazon S3: Almacenamiento escalable basado en la nube, con alta disponibilidad y durabilidad.
- Google Cloud Storage / Azure Blob Storage: Soluciones similares en la nube pública.
- Ceph / GlusterFS: Alternativas de código abierto para almacenamiento distribuido.





# Procesamiento Distribuido

## División de tareas



# Procesamiento Distribuido

## Problemas que resuelve:

- Cuellos de botella en procesamiento secuencial.
- Limitaciones de hardware en servidores únicos.
- Falta de disponibilidad ante caídas de servicio.
- Procesos lentos en contextos de grandes volúmenes de datos.

## Ejemplo práctico:

Una empresa desea analizar los logs de acceso web de millones de usuarios para detectar patrones de navegación. Usando Apache Spark, divide los archivos entre **múltiples nodos**, que procesan los datos en paralelo, reduciendo el tiempo de ejecución de horas a minutos.

El procesamiento distribuido permite dividir tareas entre varios nodos de un clúster, acelerando tiempos de ejecución y aumentando la capacidad de análisis.

## Principales herramientas:

- Apache Hadoop (MapReduce): Modelo batch, tolerante a fallos.
- Apache Spark: Procesamiento en memoria, rápido y versátil.

# Frameworks de Big Data

Los frameworks de Big Data son conjuntos integrados de herramientas, bibliotecas y servicios que permiten diseñar, implementar y ejecutar procesos de almacenamiento, transformación, análisis y visualización de grandes volúmenes de datos. Estos entornos proporcionan una base estandarizada y escalable para trabajar con arquitecturas distribuidas de forma eficiente.

## ¿Qué son y para qué sirven?

- Son plataformas que agrupan múltiples componentes tecnológicos para facilitar el desarrollo de soluciones de análisis de datos a gran escala.

## Problemas que resuelven:

- Falta de integración entre herramientas aisladas.
- Dificultades para escalar soluciones personalizadas.
- Alta complejidad en configuraciones técnicas de bajo nivel.
- Falta de rendimiento y eficiencia en entornos de datos masivos.



### Apache Hadoop

Framework pionero en procesamiento batch y almacenamiento distribuido con HDFS, ideal para grandes volúmenes de datos.



### Apache Beam

Modelo unificado para flujos batch y streaming, compatible con motores como Flink y Dataflow.



### Apache Spark

Plataforma para procesamiento distribuido en memoria, ideal para tareas batch, streaming, ML y SQL con mayor velocidad.



### Databricks

Plataforma comercial basada en Spark, con herramientas colaborativas para Ciencia de datos.



### Apache Flink

Framework para procesamiento de datos en flujo (streaming) con baja latencia, optimizado para análisis en tiempo real.

# Arquitecturas Típicas de Big Data

**Identificación de fuentes**  
Determinar los orígenes de los datos a procesar

**Utilización**  
Aplicar los resultados en la toma de decisiones



## Obtención de datos

Recopilar los datos desde las fuentes identificadas

## Almacenamiento

Guardar los datos de manera estructurada en sistemas adecuados

## Procesamiento

Transformar y analizar los datos para extraer valor



# Características Principales de la Arquitectura Big Data

- ❖ Tolerancia a fallos: El sistema debe estar diseñado para continuar operando incluso ante errores o fallos en algunos de sus componentes.
- ❖ Escalabilidad: Capacidad de aumentar las capacidades de procesamiento y almacenamiento a medida que crece el volumen de datos.
- ❖ Procesamiento distribuido: Los datos se procesan en múltiples máquinas, mejorando los tiempos de ejecución y la eficiencia.
- ❖ Datos distribuidos: Los datos se almacenan en diferentes ubicaciones para optimizar el acceso y la redundancia.
- ❖ Localidad del dato: Los procesos de análisis se ejecutan cerca de donde se almacenan los datos para reducir latencias y mejorar el rendimiento.

## Principales tipologías de arquitectura Big Data

- Big Data On-Premise
- Big Data en la Nube

La arquitectura Big Data es esencial para la toma de decisiones y la innovación en diversos sectores. Su capacidad para gestionar grandes volúmenes de datos, procesarlos y extraer información valiosa impulsa avances en inteligencia artificial, análisis predictivo y personalización de servicios, otorgando a las empresas ventajas competitivas significativas.



# Apache Hadoop

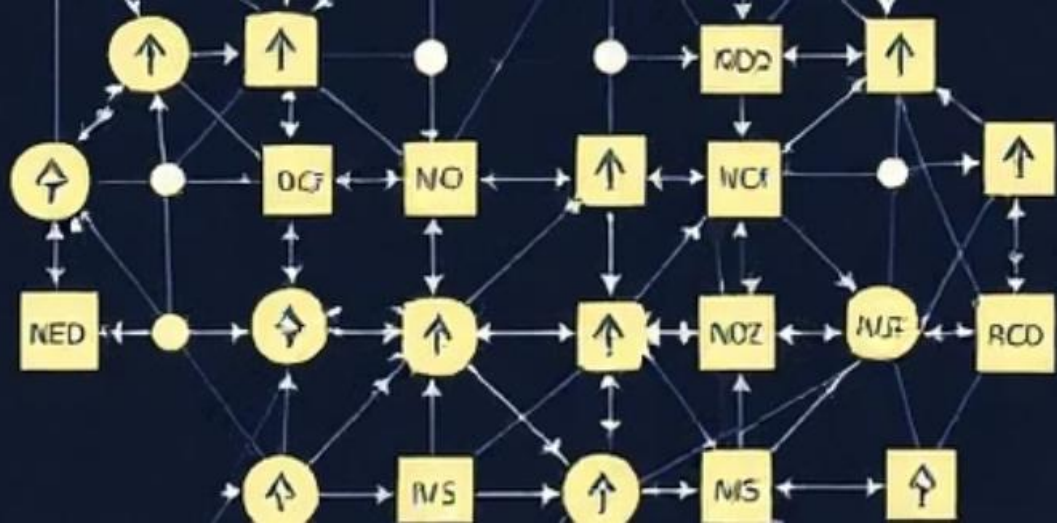
## Ventajas de Hadoop:

- ✓ Escalabilidad horizontal: puedes agregar más nodos fácilmente.
- ✓ Tolerancia a fallos: gracias a la replicación de datos en diferentes nodos.
- ✓ Eficiencia en el procesamiento: el modelo MapReduce permite procesamiento paralelo.
- ✓ Costo-efectivo: se puede usar hardware económico.
- ✓ Código abierto y comunidad activa.
- ✓ Flexibilidad: puede trabajar con datos estructurados y no estructurados.

## Desventajas de Hadoop:

- ✗ Curva de aprendizaje pronunciada: especialmente para MapReduce.
- ✗ No es en tiempo real: Hadoop está pensado para procesamiento **por lotes**, no para respuestas inmediatas.
- ✗ Problemas con datos pequeños: no es eficiente con volúmenes de datos pequeños.
- ✗ Requiere una buena configuración y administración del clúster.
- ✗ Competencia con nuevas tecnologías: como Apache Spark, que es más rápido para ciertas tareas.





# Apache Spark

## Procesamiento en memoria

Spark procesa datos en memoria (in-memory), lo que lo hace hasta 100 veces más rápido que Hadoop MapReduce para ciertas operaciones.

## Ecosistema unificado

Integra procesamiento batch, streaming, machine learning (MLlib) y consultas SQL en una sola plataforma con APIs coherentes.

## Facilidad de uso

Proporciona APIs de alto nivel en Python, Scala, Java y R, con una interfaz interactiva para exploración de datos y desarrollo ágil.

## Integración flexible

Se conecta fácilmente con HDFS, Hive, Cassandra, Kafka y otras tecnologías del ecosistema Big Data.

Material Complementario Documentación Apache Spark: <https://spark.apache.org/>

# Apache Spark

## **Spark se usa principalmente para:**

- Procesar grandes volúmenes de datos rápidamente.
- Realizar análisis en tiempo real.
- Ejecutar modelos de machine learning sobre big data.
- Consultar y transformar datos con SQL.
- Procesar flujos de datos en vivo (streaming).

## **Es una herramienta clave en áreas como:**

- Inteligencia de negocios (BI)
- Ciencia de datos
- Inteligencia artificial
- Analítica predictiva
- IoT y análisis en tiempo real

## **Ventajas de Apache Spark:**

- Velocidad: procesamiento en memoria reduce el tiempo de ejecución drásticamente.
- Versatilidad: soporta múltiples tipos de procesamiento (batch, streaming, ML, SQL, grafos).
- Escalabilidad: diseñado para escalar desde laptops hasta clústeres de miles de nodos.
- Interfaz amigable: APIs de alto nivel para programadores en distintos lenguajes.
- Integración fácil: se puede conectar con HDFS, Hive, Cassandra, Kafka, entre otras tecnologías.
- Amplio ecosistema: Spark MLlib (machine learning), Spark SQL, GraphX, Spark Streaming.

## **Desventajas de Apache Spark:**

- Alto consumo de memoria: el procesamiento en memoria requiere gran cantidad de RAM.
- Complejidad en configuración y despliegue en producción.
- No reemplaza completamente a bases de datos: no está optimizado para operaciones OLTP o consultas transaccionales.
  - Curva de aprendizaje inicial en clústeres grandes.
  - La versión de Spark Streaming original no es completamente "real-time" (usa micro-batching).

# Otras Tecnologías del Ecosistema Big Data



## Ingesta y Transmisión

Apache Kafka: Plataforma distribuida para manejar flujos de datos en tiempo real. Apache NiFi: Herramienta de integración de datos orientada a flujos, con interfaz visual. Flume: Diseñada para recolectar y mover grandes cantidades de datos de logs.



## Visualización

Kibana: Visualización de datos almacenados en Elasticsearch. Grafana: Dashboard potente para métricas en tiempo real. Tableau/Power BI: Plataformas comerciales para análisis empresarial y visualización interactiva.





## Machine Learning

TensorFlow: Librería de código abierto para aprendizaje profundo. Scikit-learn: Librería de Python para machine learning clásico. MLflow: Herramienta para gestionar el ciclo de vida de los modelos de machine learning.



# Performanse Hadoop vs Apache Spark

### Apache Hadoop



### Apache Hadoop

Disipution anzor anal  
distribluite tion or sturing  
batch: processing

Pesaupelnd  
storage, and for  
distributed storage


**\$4.5 anilly**

**MOrin: 114.25**


rf timp in full Dlartfumenafes

Performance and  
capallties

### Apache Spark



In-memory processing



**In Real-Time  
Processing**

**\$,50 kmin**

Real-time  
Analytics to have processing  
analytics andr analytics  
and untr machine learning

**\$5.6 anilly**

**MOrin: 114.05**

Your Machine Learning

Morde of the offern apcins  
pproactions and 'leende and  
tentls ancha in undate of love

## Comparativa: Hadoop vs. Spark

Característica	Hadoop MapReduce	Apache Spark
Velocidad	Lento (procesamiento en disco)	Rápido (procesamiento en memoria)
Streaming	No	Sí
Machine Learning	Limitado	Incluye MLlib
Complejidad	Más bajo nivel	APIs de alto nivel
Escenarios ideales	Procesos batch grandes	Batch + Streaming + ML

Hadoop y Spark son tecnologías complementarias en muchos casos. Hadoop proporciona un sistema de archivos distribuido (HDFS) robusto y un gestor de recursos (YARN) que Spark puede aprovechar. Sin embargo, para procesamiento rápido, análisis iterativo y aplicaciones en tiempo real, Spark ofrece ventajas significativas gracias a su procesamiento en memoria y APIs más intuitivas.

# Actividad Práctica Guiada: Detección de Transacciones Fraudulentas ¿Cómo diseñar una arquitectura Big Data?

**OBJETIVO:** Proponer una arquitectura adecuada basada en tecnologías Big Data para resolver un problema real de análisis de transacciones en tiempo real, aplicando los conceptos aprendidos en la clase (las 5V's, sistemas locales vs distribuidos, tecnologías emergentes).

**Escenario:** Una entidad financiera desea detectar **transacciones potencialmente Fraudulentas** en tiempo real para evitar afectar a sus clientes. Cada segundo, se generan miles de operaciones a través de tarjetas de crédito, transferencias electrónicas, apps móviles y cajeros automáticos en todo el país. El equipo técnico necesita diseñar una solución tecnológica que permita procesar y analizar esta gran cantidad de datos para identificar patrones sospechosos a medida que ocurren.

## **Instrucciones:**

- 1.Responde: ¿Cuáles de las 5V's están presentes en este caso? Justifica cada una.
- 2.Responde: ¿Por qué este problema no podría resolverse eficientemente con una arquitectura local?



# Actividad Práctica Guiada: Detección de Transacciones Fraudulentas ¿Cómo diseñar una arquitectura Big Data?

**Desarrollo:**

## **1. Aplicación de la 5V's**

Aplicación 5V's	Descripción en este caso
<b>Volumen</b>	Miles de transacciones por segundo, provenientes de todo el país.
<b>Velocidad</b>	Se requiere análisis en tiempo real, tan pronto ocurre la transacción.
<b>Variedad</b>	Datos estructurados (monto, hora) y semiestructurados (dispositivo, ubicación geográfica).
<b>Veracidad</b>	Necesidad de datos precisos y confiables para evitar falsos positivos.
<b>Valor</b>	Detectar fraudes rápidamente protege a clientes y reduce pérdidas económicas.

## **2. Este problema no podría resolverse eficientemente con una arquitectura local:**

- Las bases de datos relacionales no pueden escalar para soportar el volumen y la velocidad requerida.
- Un único servidor tendría cuellos de botella y puntos de falla.
- No ofrecen análisis en tiempo real (solo procesamiento batch).
- No permiten integración eficiente con múltiples fuentes de datos heterogéneos.

# Preguntas de Reflexión sobre Big Data

1

¿Cómo describirías el valor que aporta el Big Data a una organización en el contexto actual?

2

En tus palabras, ¿cuáles son las 5V del Big Data y por qué son importantes? ¿Podrías dar un ejemplo de cada una?

3

Después de lo visto hoy, ¿cómo abordarías el problema de los correos fraudulentos usando tecnologías Big Data? ¿Qué tipo de arquitectura o herramientas considerarías más adecuadas?