Ingesta de Datos en Streaming

La ingesta de datos en streaming es el proceso de recibir, procesar y almacenar datos en tiempo real a medida que se generan o se transmiten desde una fuente. A diferencia de la ingesta por lotes, que procesa grandes volúmenes de datos en intervalos programados, la ingesta en streaming trabaja de forma continua y en tiempo real.

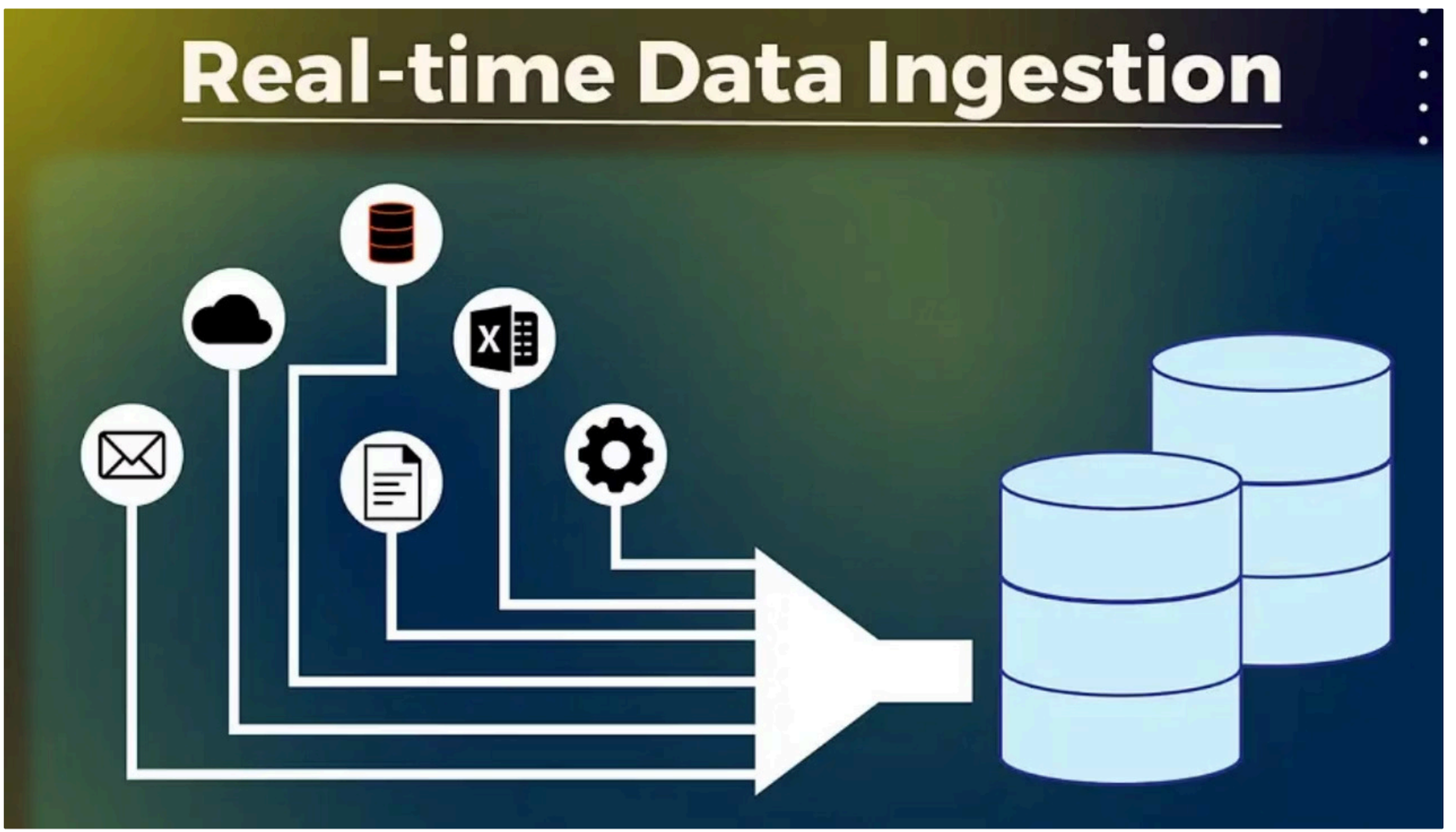
por Kibernum Capacitación S.A.

¿Qué es la Ingesta de Datos en Streaming?

Este proceso es crucial para sistemas que necesitan reaccionar de inmediato ante los datos entrantes, como en aplicaciones de monitoreo en tiempo real, análisis de comportamiento en línea, sistemas de alerta, procesamiento de transacciones financieras o redes sociales.

Características clave de la ingesta de datos en streaming:

- Tiempo real: Los datos se procesan tan pronto como son generados o transmitidos, sin necesidad de esperar por lotes.
- Escalabilidad: El sistema debe ser capaz de manejar flujos de datos con alta frecuencia y volumen sin que se detenga.
- Baja latencia: La latencia (el tiempo de retraso entre la llegada de los datos y su procesamiento) debe ser mínima.
- Resiliencia: El sistema debe ser capaz de recuperarse rápidamente de fallos y seguir procesando los datos sin pérdida.
- Procesamiento continuo: Los datos se procesan de forma continua, en lugar de almacenar grandes volúmenes y luego analizarlos.



Diferencia con la Ingesta de Datos Batch

La ingesta de datos en batch y la ingesta de datos en streaming son dos enfoques diferentes para procesar y manejar datos, y se diferencian principalmente en cómo y cuándo se procesan esos datos.

Frecuencia de procesamiento:

- Ingesta en Streaming: Los datos se procesan en tiempo real o en intervalos muy pequeños. El sistema recibe y procesa los datos de manera continua conforme llegan. No se espera a que se acumule un volumen significativo de datos para comenzar el procesamiento.
- Ingesta en Batch: Los datos se recopilan y almacenan en "lotes" durante un período determinado (por ejemplo, cada hora, día o semana). El procesamiento solo ocurre cuando se tiene un lote completo de datos, lo que significa que se procesa un conjunto de datos en un solo proceso en lugar de en tiempo real.

Latencia:

- Ingesta en Streaming: Tiene una latencia baja, ya que los datos se procesan casi inmediatamente después de llegar. La reacción ante los eventos es casi instantánea.
- Ingesta en Batch: Tiene una latencia alta, ya que los datos deben esperar hasta que se forme un lote completo. Los datos no se procesan en tiempo real, sino que esperan al siguiente ciclo de procesamiento.

Característica	Ingesta de Datos en Streaming	Ingesta de Datos en Batch
Frecuencia de procesamiento	Tiempo real, continuo	Procesado en intervalos (lotes)
Latencia	Baja, procesamiento inmediato	Alta, con retrasos entre lotes
Volumen de datos	Flujos constantes de datos	Lotes de datos grandes
Complejidad de procesamiento	Alta, necesita procesamiento en tiempo real	Baja, procesamiento en lotes
Almacenamiento	Temporal, en memoria o bases de datos distribuidas	Almacenamiento en bases de datos
Casos de uso	Monitoreo en tiempo real, IoT, alertas	Informes, análisis de datos históricos

Ventajas y Desventajas de la Ingesta en Streaming

Ventajas

- Procesamiento en tiempo real: La principal ventaja de la ingesta de datos en streaming es la capacidad de procesar datos a medida que llegan, lo que permite respuestas inmediatas ante eventos o condiciones específicas.
- Toma de decisiones en tiempo real: Facilita la toma de decisiones instantáneas basadas en los datos que se están generando en el momento.
- Mejora de la experiencia del usuario: En plataformas de servicios interactivos, permite proporcionar contenido dinámico y recomendaciones instantáneas.
- Monitoreo y alerta continua: Permite el monitoreo constante de eventos y condiciones en sistemas como IoT, redes de sensores, salud digital, o sistemas de seguridad.
- Escalabilidad: Muchos sistemas de ingesta en streaming están diseñados para ser altamente escalables.
- Mejor uso de los recursos: Al procesar los datos de manera continua, se pueden aprovechar mejor los recursos de procesamiento y almacenamiento.

Desventajas

- Complejidad en el procesamiento: El procesamiento de datos en streaming es más complejo que en batch.
- Mayor coste de infraestructura: La infraestructura necesaria para procesar y almacenar datos en tiempo real puede ser más costosa.
- Tolerancia a fallos y recuperación: La recuperación ante fallos puede ser más compleja que en el caso de la ingesta por lotes.
- Alto uso de recursos: Los sistemas de ingesta en streaming requieren un alto uso de recursos computacionales de manera continua.
- Desafíos en el manejo de datos inconsistentes: Puede ser difícil garantizar que los datos recibidos sean consistentes o completos.
- Requiere un diseño robusto: Las arquitecturas de ingesta en streaming deben estar muy bien diseñadas desde el principio.

Ventajas	Desventajas
Procesamiento en tiempo real.	Complejidad en el procesamiento.
Toma de decisiones instantáneas.	Mayor coste de infraestructura.
Mejora de la experiencia del usuario.	Tolerancia a fallos y recuperación compleja.
Monitoreo y alerta continua.	Alto uso de recursos.
Escalabilidad.	Desafíos en el manejo de datos inconsistentes.
Mejor uso de los recursos de procesamiento y almacenamiento.	Requiere un diseño robusto.

Principios Básicos y Herramientas para la Ingesta en Streaming

Ingesta Continua de Datos

Los datos se recogen de manera continua desde diversas fuentes sin necesidad de esperar a que se complete un "lote" de información.

Procesamiento en Tiempo Real

Los datos no solo se reciben de manera continua, sino que también se procesan en el mismo instante en que llegan, sin almacenamiento intermedio ni esperas.

Baja Latencia

La latencia (el tiempo entre la recepción de los datos y su procesamiento) debe ser mínima.

Escalabilidad y Flexibilidad

Los sistemas de streaming deben poder escalar fácilmente para manejar grandes volúmenes de datos y adaptarse a cambios en la cantidad de datos generados.

Tolerancia a Fallos y Resiliencia

El sistema debe ser capaz de recuperarse rápidamente de cualquier tipo de fallo, garantizando la continuidad de la ingesta de datos sin pérdidas.

Procesamiento de Datos en Eventos

Los datos se procesan en función de los eventos que ocurren, y cada evento puede desencadenar un flujo de trabajo específico. Esto permite que el sistema se enfoque en la lógica de negocio relacionada con el evento en lugar de manejar todos los datos de manera uniforme.

Eficiencia en el Uso de Recursos

Dado que el procesamiento de datos es continuo, el sistema debe optimizar el uso de recursos como la CPU, memoria y almacenamiento para evitar la sobrecarga del sistema y garantizar un rendimiento constante.

Tecnologías Populares

Apache Kafka

Es una plataforma distribuida de mensajería y procesamiento de flujo de datos en tiempo real. Originalmente creado por LinkedIn, Kafka es ampliamente utilizado para la ingesta de datos en streaming debido a su alta capacidad para manejar grandes volúmenes de datos de manera eficiente y en tiempo real.

Apache Flink

Es un framework de procesamiento de flujos de datos en tiempo real, diseñado para ejecutar aplicaciones de procesamiento de datos complejos y en streaming. Flink permite procesamiento tanto de streams (datos en tiempo real) como de batch (lotes de datos), pero se especializa en el procesamiento de datos de eventos en tiempo real.

Apache Spark Streaming

Es una extensión de Apache Spark que permite procesar datos en tiempo real. A diferencia de otras soluciones como Flink, Spark Streaming utiliza el modelo de micro-batch, donde los datos se agrupan en pequeños lotes (micro-batches) para ser procesados.

COMPARACIÓN Y CARACTERÍSTICAS PARA UNA BUENA ELECCIÓN

Tecnología	Tipo de Procesamiento	Escalabilidad	Modelo de Fallos	Casos de Uso
Apache Kafka	Mensajería y transmisión de flujos de datos	Alta (horizontal)	Alta, con replicación y particionamiento	Comunicación entre microservicios, eventos de redes sociales, datos de IoT
Apache Flink	Procesamiento de flujos y eventos en tiempo real	Alta (horizontal)	Alta, con checkpoints	Análisis de datos en tiempo real, monitoreo IoT, procesamiento de eventos complejos
Apache Spark Streaming	Micro-batch (procesamiento en pequeños lotes)	Alta (horizontal)	Alta, con recuperación de estado	Análisis de logs, procesamiento de eventos de redes sociales, análisis de datos en tiempo real

Conclusión:

Apache Kafka

Es ideal cuando se necesita una plataforma robusta y distribuida para la transmisión de mensajes de alta disponibilidad en tiempo real.

Apache Flink

Es adecuado cuando se requiere procesamiento complejo de eventos en tiempo real, análisis de patrones o eventos correlacionados.

Apache Spark Streaming

es útil cuando se prefieren los micro-batches y cuando ya se está utilizando el ecosistema Spark para otras tareas como machine learning o análisis SQL.

Cada herramienta tiene sus ventajas dependiendo del caso de uso, por lo que la elección entre ellas dependerá de los requisitos específicos de la aplicación que se esté desarrollando. Si necesitas más detalles o ejemplos específicos de implementación, no dudes en preguntar.

Casos de Uso y Actividad Práctica Guiada

Casos de Uso Populares

- **Detección de Fraude en Tiempo Real (Apache Kafka y Apache Flink):** Las instituciones financieras, como bancos y plataformas de pagos, deben monitorear las transacciones de los usuarios en tiempo real para detectar actividades sospechosas o fraudulentas.
- **Monitoreo en Tiempo Real de Sensores IoT (Apache Kafka y Apache Spark Streaming):** En aplicaciones de Internet de las Cosas (IoT), se recopilan enormes cantidades de datos de sensores distribuidos en diversos dispositivos, como vehículos, fábricas o dispositivos de salud.

Actividad Práctica Guiada: Monitoreo en Tiempo Real de Sensores en una Planta Industrial

En este caso práctico, vamos a simular un escenario en una planta industrial donde se monitorean sensores IoT de temperatura y vibración de máquinas en tiempo real para detectar posibles fallos o anomalías.

1 Ingesta de datos en streaming

De los sensores de temperatura y vibración en tiempo real desde la planta industrial.

2 Análisis en tiempo real

Para detectar condiciones anómalas (por ejemplo, temperaturas altas o vibración fuera de rango) y generar alertas inmediatamente.

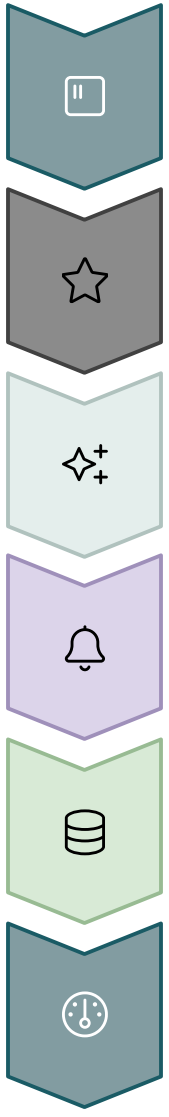
3 Visualización de datos

En un panel para monitorear el estado de las máquinas y el comportamiento de los sensores.

Elementos a Utilizar para la Ingesta de Datos en Streaming

1. **Sensores IoT**
2. **Apache Kafka (Ingesta de Datos)**
3. **Apache Spark Streaming (Procesamiento en Tiempo Real)**
4. **Base de Datos NoSQL o Sistema de Almacenamiento**
5. **Sistema de Alerta (Notificaciones en Tiempo Real)**
6. **Interfaz de Usuario (Dashboard para Monitoreo)**
7. **API o Servicio Web para Acceso Externo**

Flujo de Trabajo de la Ingesta de Datos en Streaming:



Recopilación de datos de sensores

Sensores de temperatura y vibración recopilan datos continuamente de las máquinas industriales.

Ingesta con Kafka

Los datos son ingestados en tiempo real mediante Apache Kafka para su distribución.

Procesamiento con Spark Streaming

Apache Spark Streaming analiza los datos para detectar anomalías y condiciones fuera de rango.

Generación de alertas

Se generan notificaciones inmediatas cuando se detectan condiciones anómalas.

Almacenamiento de datos históricos

Los datos se almacenan en MongoDB/Cassandra/HDFS para análisis posteriores.

Visualización en tiempo real

Grafana/Kibana muestra dashboards con el estado actual e histórico de las máquinas.

Tecnologías Utilizadas:

Sensores IoT

Sensores de temperatura y vibración.

Apache Kafka

Para la ingesta de datos en tiempo real desde los sensores.

Apache Spark Streaming

Para el procesamiento y análisis en tiempo real de los datos de los sensores.

MongoDB/Cassandra/HDFS

Para almacenamiento de los datos históricos.

Grafana/Kibana

Para la visualización en tiempo real de los datos y alertas.

Twilio/SendGrid/Slack

Para el sistema de notificación en tiempo real.

API RESTful

Para acceso a los datos y alertas desde sistemas externos.