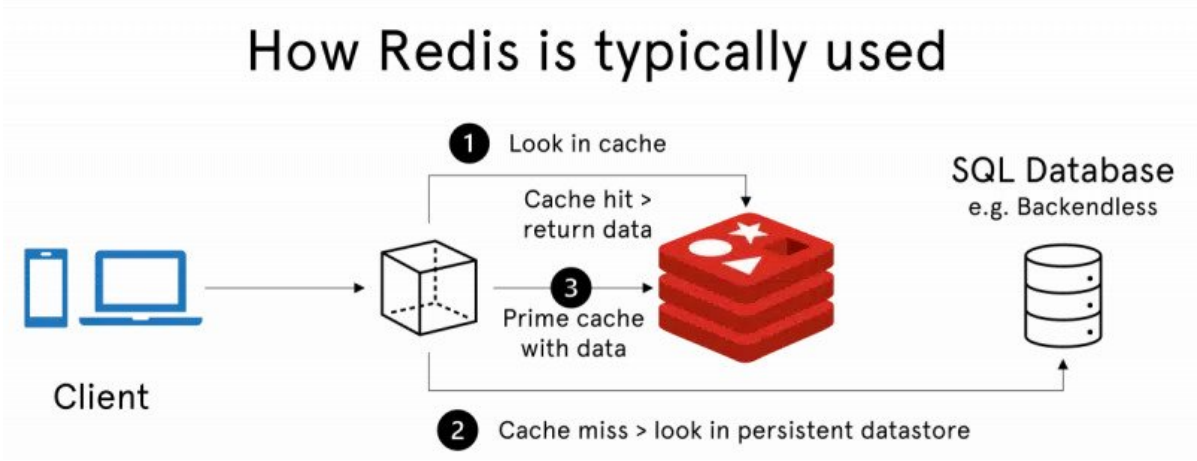
Clase del martes 22 de agosto del 2023

Redis

Redis es una estructura de almacenamiento de datos en memoria utilizada como base de datos, caché motor de streaming y message broker.

El uso común de redis es el siguiente:



Apache Kafka

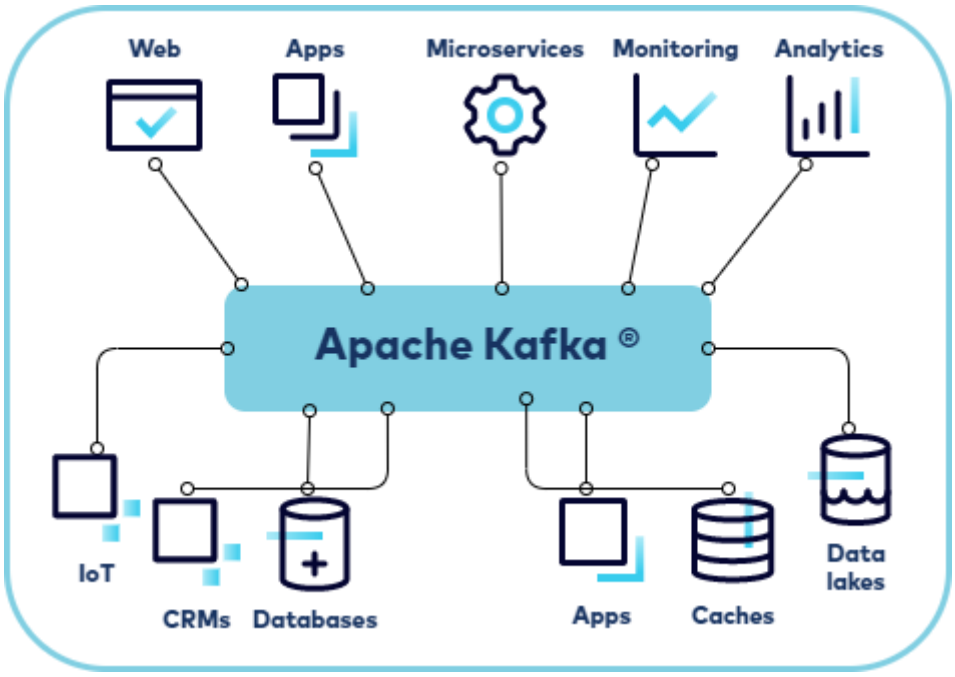
Kafka es una base de datos en memoria principal, aunque a muchas personas no les gusta considerarlo como tal, debido a que su objetivo principal
es almacenar información de forma temporal.

Kafka almacena temporalmente los registros antes de almacenarlos, en buffers en una ventana de Kafka, se procesan y se almacenan.

Trabaja con raw data, que son datos que no han sido procesados ni transformados de ninguna manera.

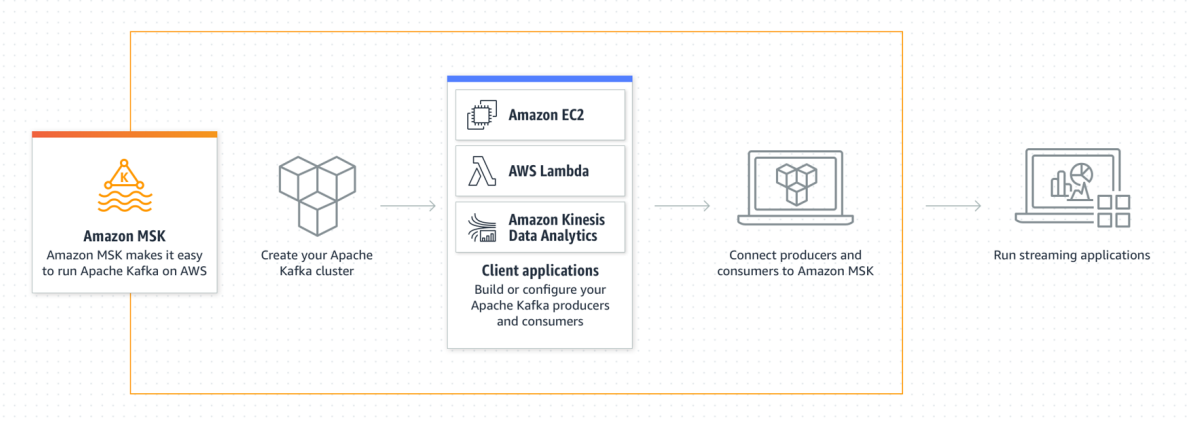
La tarea principal de una base de datos es almacenar datos.

La transformación de datos se refiere a cambiar su formato.



Amazon MSK es un producto de Amazon para Kafka, este mete lo que sea que mantiene la información

Si se desea contar con alta disponibilidad de datos se debe gastar mucho dinero, ya que se debe invertir en equipo y tecnologías.

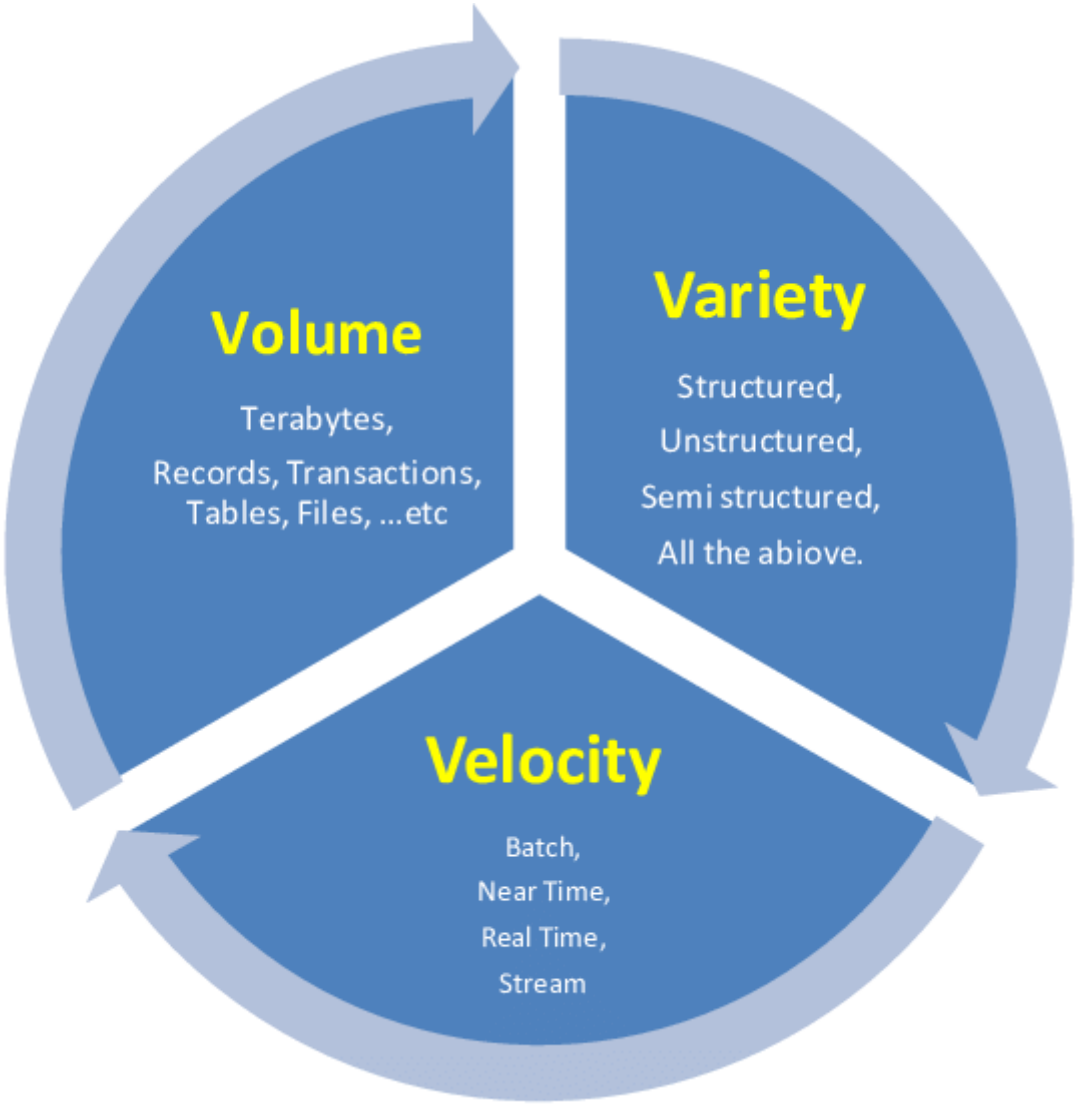


Big data

Paul Vixie define Big data como: "aquella cantidad de datos que no puedo tener en memoria principal al mismo tiempo" lo que implica que es necesario subir y bajar información a disco.

Big data tiene entre sus valores principales :

- velocidad
- volumen
- variedad



En Big data se tienen datos con mucha velocidad, mucho volumen, mucha variedad donde se necesita un buen tiempo de respuesta para procesarlo cuando se tiene poca memoria.

En Big data se trabaja en un sistema distribuido con varias computadoras trabajando a la vez, cada una con un trabajo.
Para pasar datos entre computadoras se usa la red y se desperdician recursos ya que se despierta el sistema operativo muchas veces.

MapReduce:

Es un Framework que toma un dataset batch y lo particiona generando splits generalmente de 256 megas.
Una vez particionada se pasa por un proceso de mapeo.

MapReduce cuenta con las siguientes fases:

map: es una función que se aplica a la información, es una transformación

shuffing: los datos generados por map, se agrupan según una clave de modo que los datos relacionados estén disponibles para el reduce.

Cuando se ejerce mucha presión a la red se producen context switches lo que baja el rendimiento de todo el sistema.

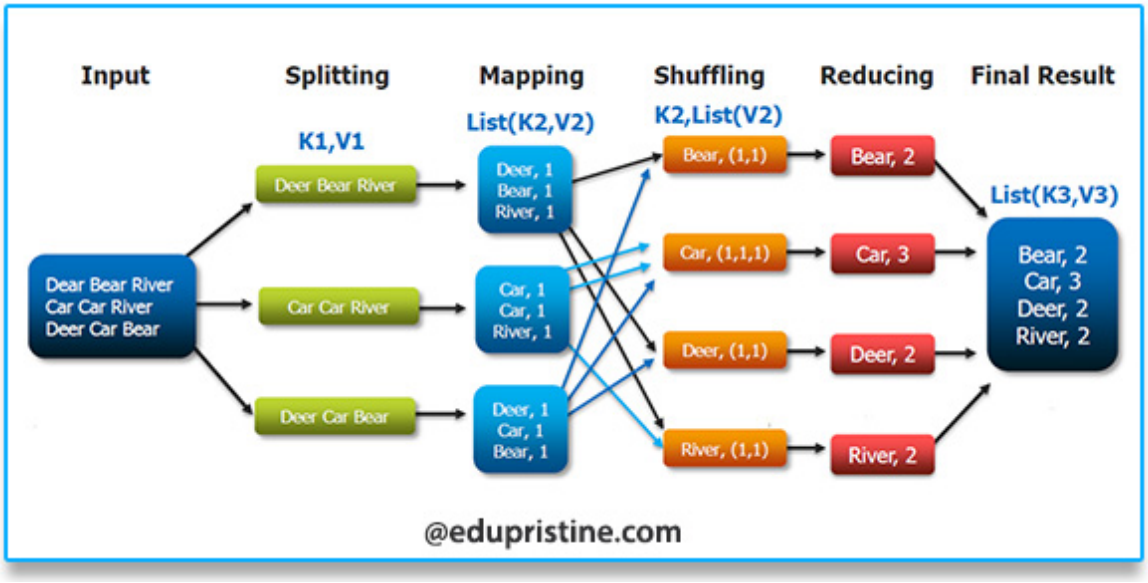
Reduce: Toma la información y la sumariza.

NIC: posee un buffer, desde la red se recibe más información de la que puede recibir, en estos casos el buffer bota información lo que provoca que se obtengan timeouts y el sistema indica que está caído, no sirve.

En un sistema distribuido el mayor problema se encuentra en la red.

¿como me doy cuenta de que el sistema está fallando en MapReduce?

Por medio de observabilidad.



Hadoop

Framework por excelencia para procesar grandes cantidades de datos, este pone a correr en las computadoras designadas y comienza a procesar en paralelo.

Tiene cliente que por medio de una aplicación de consola se le envía algo a procesa. Se envía una tarea por medio de un script donde se le indica cuantas particiones necesita.

Hadoop cuenta con los siguintes conceptos:

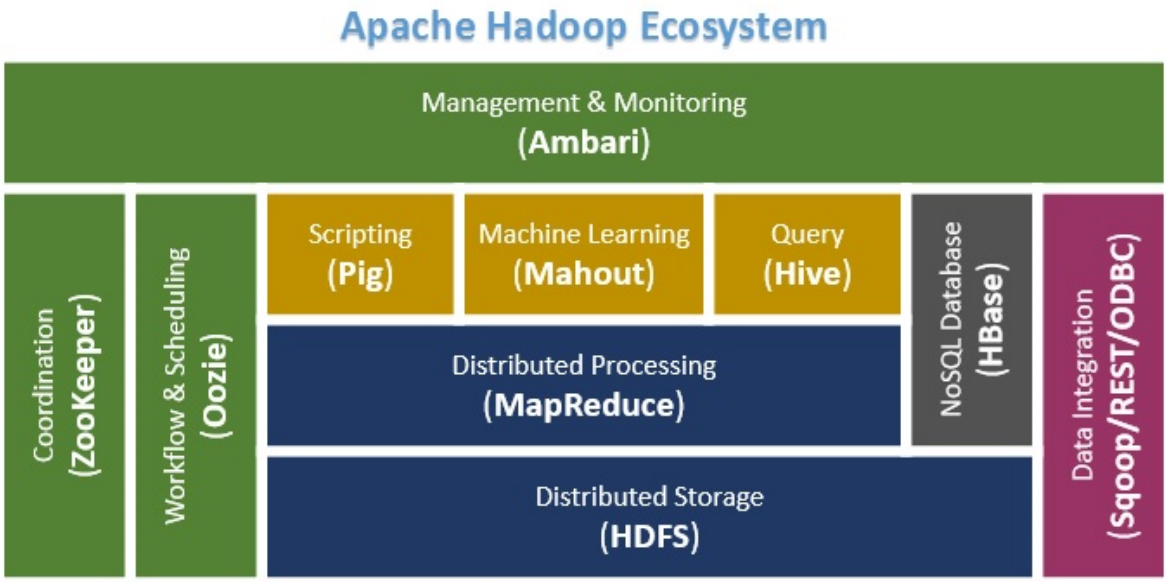
Coalesción: el proceso en el cual se combinan particiones de datos en una sola partición más grande.

Partición: es la división de datos en bloques más pequeños

Yarn: recibe el task realiza la mejor estrategia para ejecutar el task, manda particiones a diferentes máquinas y el task manager empieza la ejecución en memoria principal y cuando termina empieza con una nueva tarea.

Hadoop implementa un algoritmo de divide y vencerás.

RDD: es un conjunto de información, este le dice al framework que lea un archivo, luego realiza un scan de las primera 100 líneas del archivo (valor por defecto que puede cambiarse) luego empiezan las particiones, de 256 megas (configurable), posteriormente las particiones se asignan a diferentes máquinas dentro de sistemas distribuidos en Hadooop, estas tienen un disco. Las particiones nunca se distribuyen una a una y se intenta tener redundancia por si falla una de las máquinas.



Spark

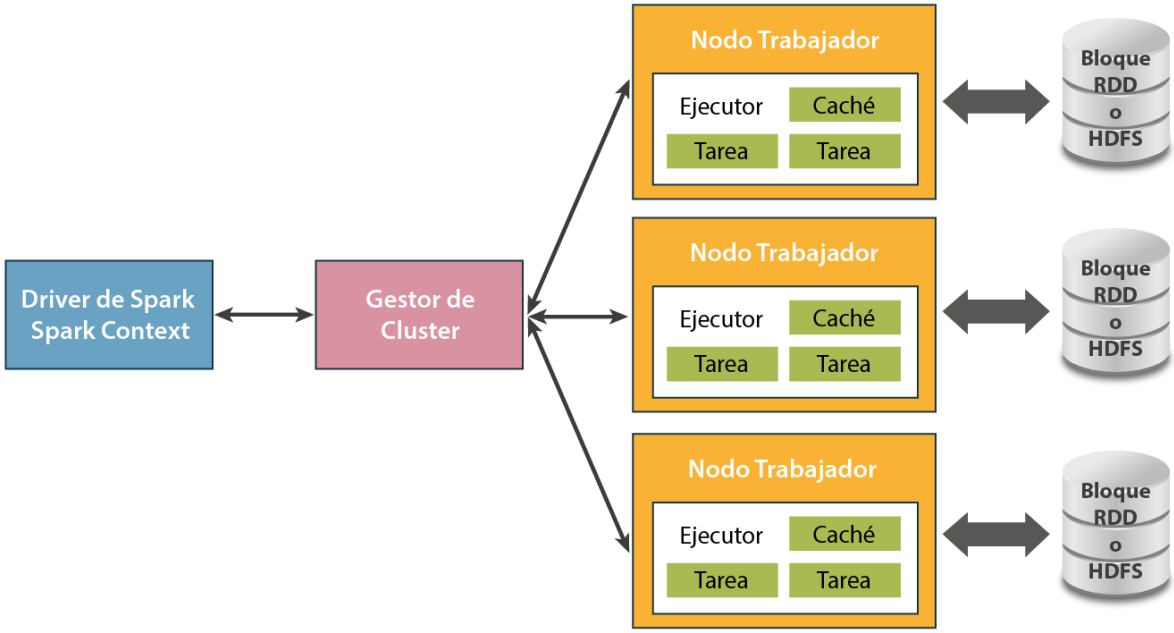
Posee **lazy evaluation**, lo que significa que se le instruye al sistema la cantidad de transformaciones que se quiere ejecutar sobre los datos, pero el sistema no los hace sino que se toman transformaciones y se empiezan a pegar una detrás de la otra, a esto se le llama transformaciones encadenadas.
Al ejecutar una acción se desencadena todo el procesamiento.

Las tareas especulativas utilizan divide y vencerás para sacar la tarea en el menor tiempo y de la forma mas óptima posible.
Los RDD no tienen tipos, trabajan con datos estructurados, no estructurados y semiestructurados.

Spark posee inmutabilidad.

spark no va a modificar el dataset original sino que va a generar otro datasaet, y así sigue con cada transformación.

Spark me permite en un dataset, serializar la información y al conocer el tipo de dato que se maneja y las restricciones de tipado permite transformarlo y enviarlo por la red y devolverlo a lo normal del otro lado, se salta lo orientado a objetos y se obtiene la misma información de memoria en ambos lados.



Blue Apache toma fuentes de datos de bases relacionales y no relacionales, encuentra matches y genera una base de datos con ello.