NoSQL

Agradecimientos

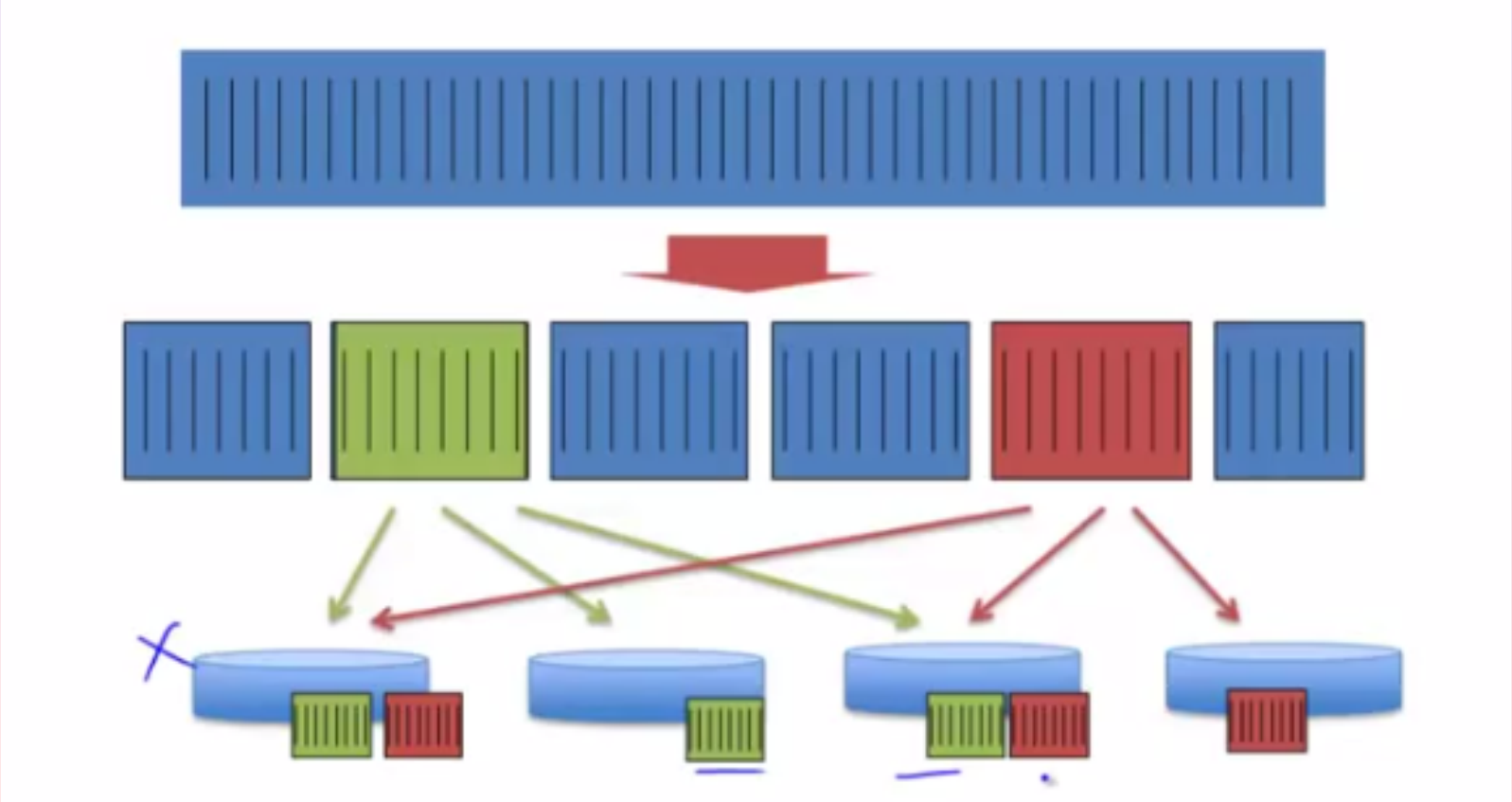
- Este curso es una versión libre del curso on-line Introducción a la Ciencia de Datos (Bill Howe, Univ. de Wasshington) https://www.coursera.org/course/datasci

NoSQL

- Sistemas de gestión de datos que surgieron para dar soporte a aplicaciones web de gran tamaño
- Las bases de datos relacionales no pueden escalar a 1000s de máquinas
  - Limitaciones relacionadas con la consistencia y la transaccionalidad

Contexto operativo

- Escalabilidad (1000s de nodos)
- Alta disponibilidad
- Tolerancia a fallos
- Soportar updates de las réplicas (copias redundantes)



Teorema CAP

CAP Theorem [Brewer 2000, Lynch 2002]

- Consistency
  - Do all applications see all the same data?
- Availability
  - Can I interact with the system in the presence of failures?
- Partitioning
  - If two sections of your system cannot talk to each other, can they make forward progress on their own?
    - If not, you sacrifice Availability
    - If so, you might have to sacrific Consistency – can't have everything

- Conventional databases assume no partitioning – clusters were assumed to be small and local
- NoSQL systems may sacrifice consistency

Consistencia

UNIVERSITY of WASHINGTON

Example

User: Joe
Friends: Sue, …
Status: "I'm sleepy"
Wall: "…", "…"

User: Sue
Friends: Joe, Kai, …
Status: "Headed to new Bond flick"
Wall: "…", "…"

User: Kai
Friends: Sue, …
Status: "Done for tonight"
Wall: "…", "…"

Write: Update Sue's status. Who sees the new status, and who sees the old one?

Databases: *"Everyone MUST see the same thing, either old or new, no matter how long it takes."*

NoSQL: *"For large applications, we can't afford to wait that long, and maybe it doesn't matter anyway"*

Consistencia eventual

El modelo de consistencia que predomina en las bases de datos NoSQL se denomina *consistencia eventual*:

- En caso de un update, las réplicas convergen a la misma información a lo largo de un cierto tiempo (o sea: no es inmediato como en el modelo transaccional puro del modelo relacional)
- Lo que las aplicaciones ven mientras tanto es difícil de predecir debido a esta naturaleza progresiva de propagación de los cambios

Características generales de Bases NoSQL

- Capacidad de escalar horizontalmente (muchos nodos) operaciones simples:
  - Búsquedas por clave, leer o escribir un conjunto reducido de registros
- Capacidad de replicar y particionar la información entre muchos nodos
- API simple (no SQL)
- Modelo de concurrencia simple (no transaccional)
- Uso eficiente de índices distribuidos y de la memoria para el almacenamiento de los datos
- Capacidad de agregar dinámicamente nuevos atributos a los registros (sin esquema predefinido)

Taxonomía de Bases NoSQL

Es posible clasificar las herramientas NoSQL de acuerdo a la caracterización de la información que procesan:

- **Documento**: estructuras anidadas (XML, JSON)
- **Registros extensibles**: conjuntos de atributos (esquema); se pueden agregar atributos dinámicamente
- **Objetos Key-Value**: conjuntos de pares clave-valor (sin anidamiento)

Joins

- Algo importante que las bases de datos NoSQL relegan (en general) es la posibilidad de hacer joins
- Hay variantes que colocalizan los datos de modo de que ciertas entidades queden anidadas (ej.: en un blog las entidades post y comentario pueden quedar intercaladas)

Algunas críticas a NoSQL

- No soportan ACID (como las bases de datos transaccionales)
  - Atomicity: las transacciones se ejecutan de modo todo (commit) o nada (rollback)
  - Consistency: las transacciones pasan la base de un estado válido a otro (en relación a la reglas del modelo de datos)
  - Isolation: las transacciones no se interfieren; el resultado de la ejecución es como si las transacciones se hubieran ejecutado en serie (en algún orden)
  - Durability: los cambios son persistentes (tolerancia a fallos); log
- No están estandarizadas