**Sistemas de almacenamiento**

* Datos en reposo: Aquellos que se encuentran fuera del acceso habitual. Son inmutables y se almacenan en redes de almacenamiento local (SAN) o en la nube.
  + IOPS: Input Output Per Second
* Estos datos se emplean para guardar datos históricos.
* Datos en uso: Son aquellos que residiendo en un tipo de almacenamiento están siendo usados por sistemas del negocio. Dependiendo del tipo de sistema soportará unos tipos de operaciones más que otros.
* Datos en tránsito:

**Sistema de Gestión de Base de Datos Relacional (RDBMS)** : La mejor opción para cargar OLTP, y soluciones en ERP, CRM y MDM.

**Modelo Relacional**: Definido en 1970 por E.F.Codd.

* Se organizan los datos en filas y columnas.
* Un conjunto de ocurrencias representa una entidad, constituyendo una tabla.
* Relación entre entidades es relación entre tablas.
* **El lenguaje SQL** sirve para gestionar los datos.

**Gestión de cargas analíticas**

Tecnología para sistemas de data warehouse. Un RMBDS es:

* Poliglotía: Son híbridos. Incluye soporte para XML, JSON o RDF (**datos semiestructurados**), puede cubrir funciones de un data lake.
* Extensibilidad: Soporta despliegues de lógica de negocio mediante:
  + Lenguaje procedimental: A través de funciones y procedimientos almacenados
  + Lenguaje externo: Java, C, Python.. etc.

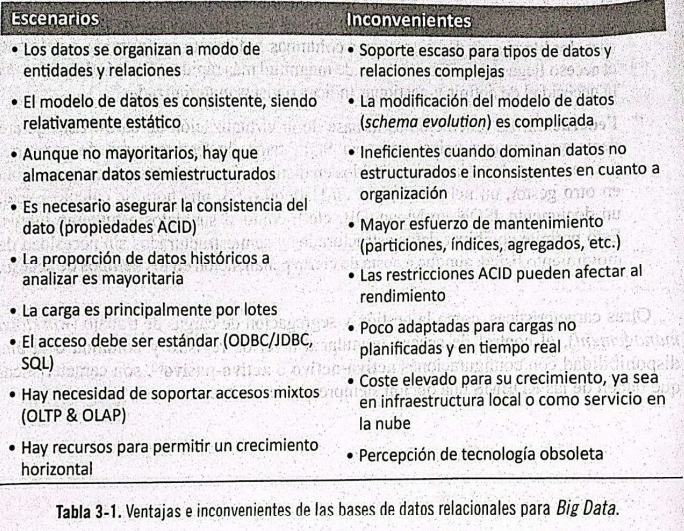
Implementa la **localidad del dato**, manipulando datos no estructurados

* Rendimiento: Soporta arquitecturas distribuidas en forma de cluster compuesto por varios nodos. **Fragmentación de datos** (data sharding)
* Organización: En sistemas de OLTP (acceso aleatorio), soportan organización en columnas.
* Federación: Base de la virtualización de datos. Con una única sentencia SQL, podemos acceder a los datos distribuidos en distintas fuentes heterogéneas (Hadoop, S3, Excel..).

Otras características:

* Gestión y segregación de cargas de trabajo
* Control de acceso granular
* Alta disponibilidad con configuraciones **activo-activo o activo-pasivo**

**Escenarios e inconvenientes**



**Software y soluciones para data warehouse**

Algunos ejemplos de motores tradiciones de base de datos:

* Licencias comerciales: Oracle Database, Microsoft SQL Server
* Código abierto: PostgreSQL, MariaDB, MySQL

Sistemas de data warehouse nativos en la nube:

* PaaS: Servicio de Plataforma
* SaaS: Servicio de Software

**Modelo serverless**: Asignación de recursos al cluster es realizada por el proveedor del servicio. Elástico, y costo ajustado. Soporta tablas externas.

**Sistemas de archivos distribuidos**

**Sistema de archivos (FS)**: Encargado de almacenar y recuperar los datos. Flexibilidad en estructura de datos, sencillez a la hora de programar

**Sistema de archivos distribuido (DFS):** Cuando se realiza en múltiples servidores conectados por red. Escalado horizontal prácticamente ilimitado.

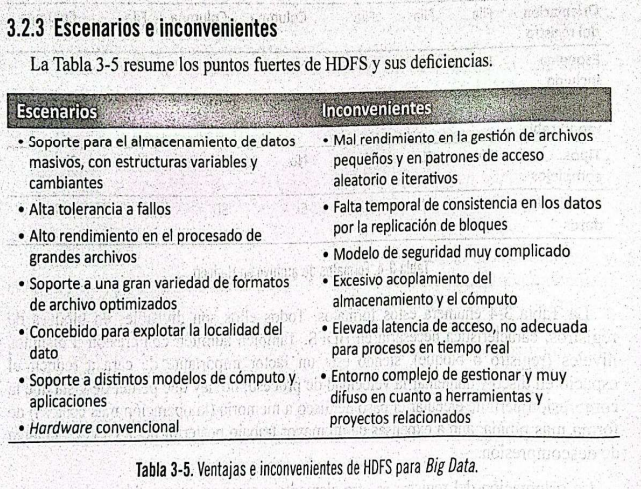
**Apache Hadoop – HDFS**: Proyecto de código abierto. Consistente en una librería de software para el procesamiento distribuido de grandes conjuntos de datos. Capa de almacenamiento.

Características/Premisas:

* Resiliencia: Tolerable a fallos
* Portabilidad: HDFS es fácilmente portable
* Escalable: Puede escalar a varios cientos de nodos por clúster
* Acceso por lotes: La mayoría de operaciones son de lectura
* Localidad del dato: Ejecución de aplicaciones en los nodos

Desde el punto de vista estructural:

* **Servidor de nombres (NameNode):** Elemento de software. Habitual disponer de uno secundario para evitar un único punto de fallo (Mantiene una copia de los archivos). Su función consiste en mantener el espacio de nombre del sistema de archivos (estructura jerárquica y nomenclatura).
  + **FSImage:** Foto actual con metadatos
  + **EditLog:** Archivo de transacciones
* **Servidor de datos (DataNode):** Elemento de software. Considera la distribución de nodos entre bastidores para la ubicación de los bloques, es decir, la ubicación y las réplicas es decisión suya.



Mantenimiento y desarrollo de Hadoop: Cloudera y HortonWorks

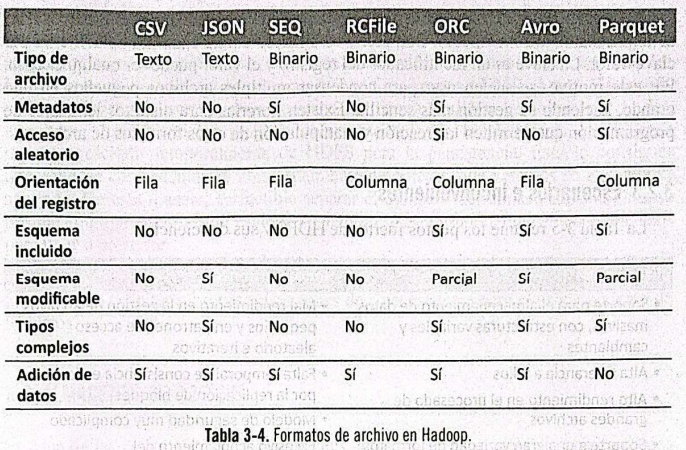
Proveedores de servicios Hadoop y Spark: Amazon EMR, Azure y Google Dataproc

**YARN**: Entorno de programación de aplicaciones distribuidas. Es el único elemento imprescindible de Hadoop. Planificación de trabajos y gestión de recursos del cluster

**WORM**: Write Once Read Many. Modelo de una escritura y múltiples lecturas, el acceso se efectúa sobre todo el contenido.

**MapReduce**: Procesamiento en paralelo de grandes conjuntos de datos

**Formatos de archivos en Hadoop**



**ORC** (Optimized Row Columnar) y **Parquet** son 2 **formatos de archivo** orientados a **almacenamiento en columnas** más empleados.

Si existe la necesidad de que el esquema pueda variar con el tiempo, usar formato **Avro**

El formato de secuencia (**SEQ**) es muy empleado en Hadoop como mecanismo de almacenamiento intermedio.

**Almacenes de objetos**

Cada **datanode** del cluster tiene sus propios **discos conectados directamente (DAS)**

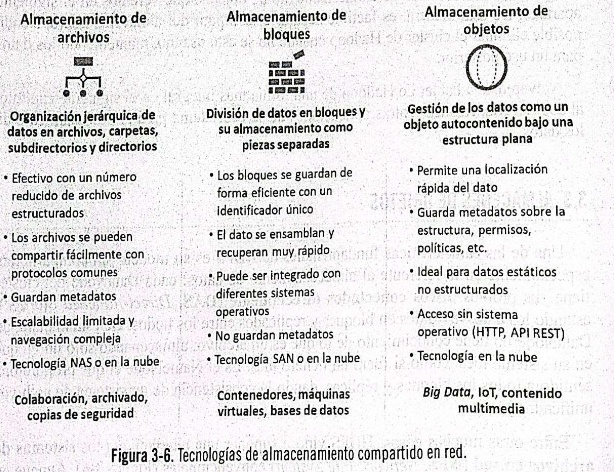
**HDFS** vino a suponer una alternativa a los **sistemas de archivos en red (NFS)**

En **NFS** existe un almacenamiento compartido en forma de **servidor conectado en red (NAS)**

Los **NAS** se utilizan como **servidores de ficheros**

**Compartición de datos** a través de una **red de** **dispositivos de almacenamiento (SAN)**

**Tecnologías de almacenamiento compartido en red**



**Almacenamiento de objetos**: El elemento de gestión es un objeto, formado por datos con un identificador único y un conjunto de metadatos.

Características:

* Permite gestión de grandes volúmenes de datos
* Ofrece accesos programáticos (Se manipula desde distintas interfaces)

**Su popularidad y uso está ligada al desarrollo de la computación en la nube**

Los objetos se agrupan en contenedores, que permite un control de acceso sobre los objetos. Implementa clases de almacenamiento.

**Falta de características ACID (Transacciones)**

Atomicity, Consistency, Isolation and Durability: Atomicidad, Consistencia, Aislamiento y Durabilidad, en español.

**Asume el procesamiento por lotes y en tiempo real**, enfocado a solventar problemas dentro del data lake

|  |  |
| --- | --- |
| **Escenario** | **Inconveniente** |
| Necesidad de interoperabilidad | Datos inmutables |
| Libertad de etiquetado, localización rápida | No soporte transaccional (ACID) |
| Alta disponibilidad y resiliencia | Variabilidad en los tiempos de acceso |
| Gran elasticidad | Organización de objetos plana |
| Optimización de almacenamiento | Mayor latencia |
| Soporte a gran variedad de formatos | Dependencia de los servicios del proveedor |
| Diversas opciones de encriptado | Mal rendimiento en cargas de trabajo |

**Algunos servicios**:

* Amazon S3
* Google Cloud
* Azure

**Bases de datos NOSQL**

Rasgos comunes entre ellos:

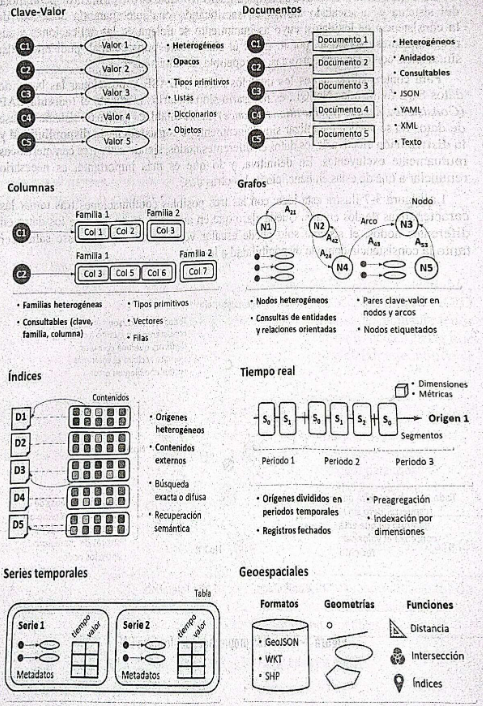
* Almacenan datos en un formato sin tabular, elimina los joins
* Son muy flexibles
* Crecimiento de la bd horizontal, aumentando o reduciendo nodos según necesidad
* No implementan ACID
* Desarrollo de soluciones es más simple

**Modelo Base**: Es a las Bases NoSQL lo que el modelo ACID a las transacciones. Escalado horizontal. Algunas propiedades de este modelo (Son las siglas de BASE):

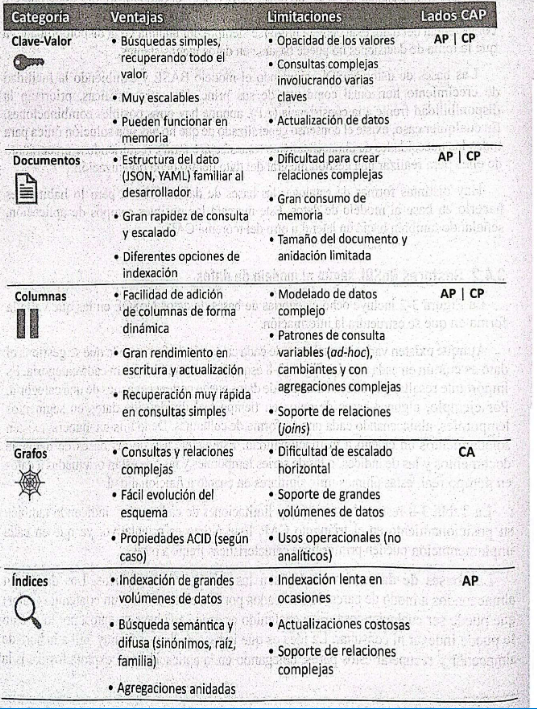
* Consistencia eventual: No asegura consistencia inmediata una vez finaliza una operación
* Estado transitorio: Puede variar entre accesos
* Disponibilidad: Tolera situaciones de fallo

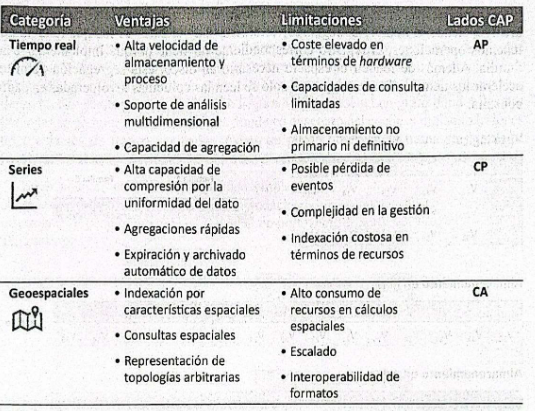
**Teorema CAP**: No se puede garantizar la **consistencia, disponibilidad y distribución (siglas CAP)** de datos entre diferentes nodos. Necesario renunciar a 1 para beneficio de las otras 2

**Categorías de las BD NoSQL**



**Características de las BD NoSQL**





* **En SQL** un **sistema OLTP** debe ubicarse necesariamente en la intersección CA
* **Las NoSQL** siguen el **modelo BASE**

**Bases de datos Clave-Valor**

Los datos se almacenan a modo de pares, identificados por una clave única y su valor. No son buena opción para transacciones complejas

Software y servicio: Redis

**Base de datos de documentos**

Parten de la misma idea que las clave-valor, pero el valor es con una estructura tipo XML, JSON o YAML. Suele estar desnormalizado. No son buena opción para transacciones complejas

Software y servicio: MongoDB

**Bases de datos en columnas**

Agrupan atributos en familia y almacenan de forma contigua.

* Ej: Una familia de columnas puede contener información de un cliente, y otra familia de columnas sobre operaciones

Presentan un escalado horizontan, y una flexibilidad alta.

Software y servicio: ApacheCassandra

**Almacenes de grafos**

Representar multiples relaciones complejas a través de una estructura en forma de red.

Un grafo se compone de un conjunto de nodos conectados por arcos, que estos a su vez presentan multiples etiquetas y propiedades (estas ultimas en formato de clave-valor)

Ejemplo: Gremlin

Software y servicio: Neo4J

**Bases de datos de índices**

Son capaces de hacer consultas en otros almacenes, indexando su contenido. Son utilizados para búsquedas textuales utilizando palabras clave.

Es habitual que incorporen capacidades de **procesamiento del lenguaje natural (NLP).**

**Bases de datos en tiempo real y series temporales**

Comparten una serie de características comunes:

* Trabajan con información fechada
* Permiten realizar análisis multidimensionales

**Bases de datos geoespaciales**

Suelen ser extensiones de las bases de datos relaciones o documentos, funciones para la manipulación de estos e índices para soportar consultas espaciales.

Ejemplo: Aplicaciones como el calculo de rutas

**Aplicaciones de las bases de datos NoSQL**

