NO SQL

A QUE LLAMAMOS NO SQL DATA BASES

- A un conjunto muy variado de base de datos no relacionales, que fueron diseñadas tomando en cuenta la existencia de datos no estructurados. Sus características principales son
  - No son relacionales
  - Distribuidas
  - Codigo Abierto
  - Escalan muy fácilmente en forma horizontal

A QUE LLAMAMOS NO SQL DATA BASES

- Las características mas generalmente aceptadas son :
  - No tienen esquema (schema free) ,
  - Tienen mecanismos sencillos de replicacion,
  - Tienen interfases sencillas
  - Pueden almacenar grandes volúmenes de datos
  - Cumplen con las propiedades BASE, no ACID

BASE

Basic Availability: el Sistema garantiza la disponibilidad en términos del teorema CAP

Soft-state: El estado de la base de datos puede cambiar a lo largo del tiempo, aun sin intervención externa. Esto se debe a la consistencia eventual

Eventual consistency: el sistema va a ser consistente a lo largo del tiempo, siempre que durante ese tiempo no reciba inputs

BASE VS. ACID

- BASE contiene 3 principios básicos:
- Basic Availability. La aproximación de las bases de datos NoSQL se focaliza en la disponibilidad de los datos, aun cuando sucedan múltiples fallas. Consigue este objetivo mediante el uso de una administración de base de datos altamente distribuida. En lugar de mantener una única copia de la base de datos y concentrarse en la tolerancia a fallas , las bases de datos NoSQL distribuyen los datos en muchos discos diferentes con un alto grado de replicación. Si sucede una falla que inhabilita el acceso a un grupo de datos, esto no necesariamente afecta a toda la base de datos.

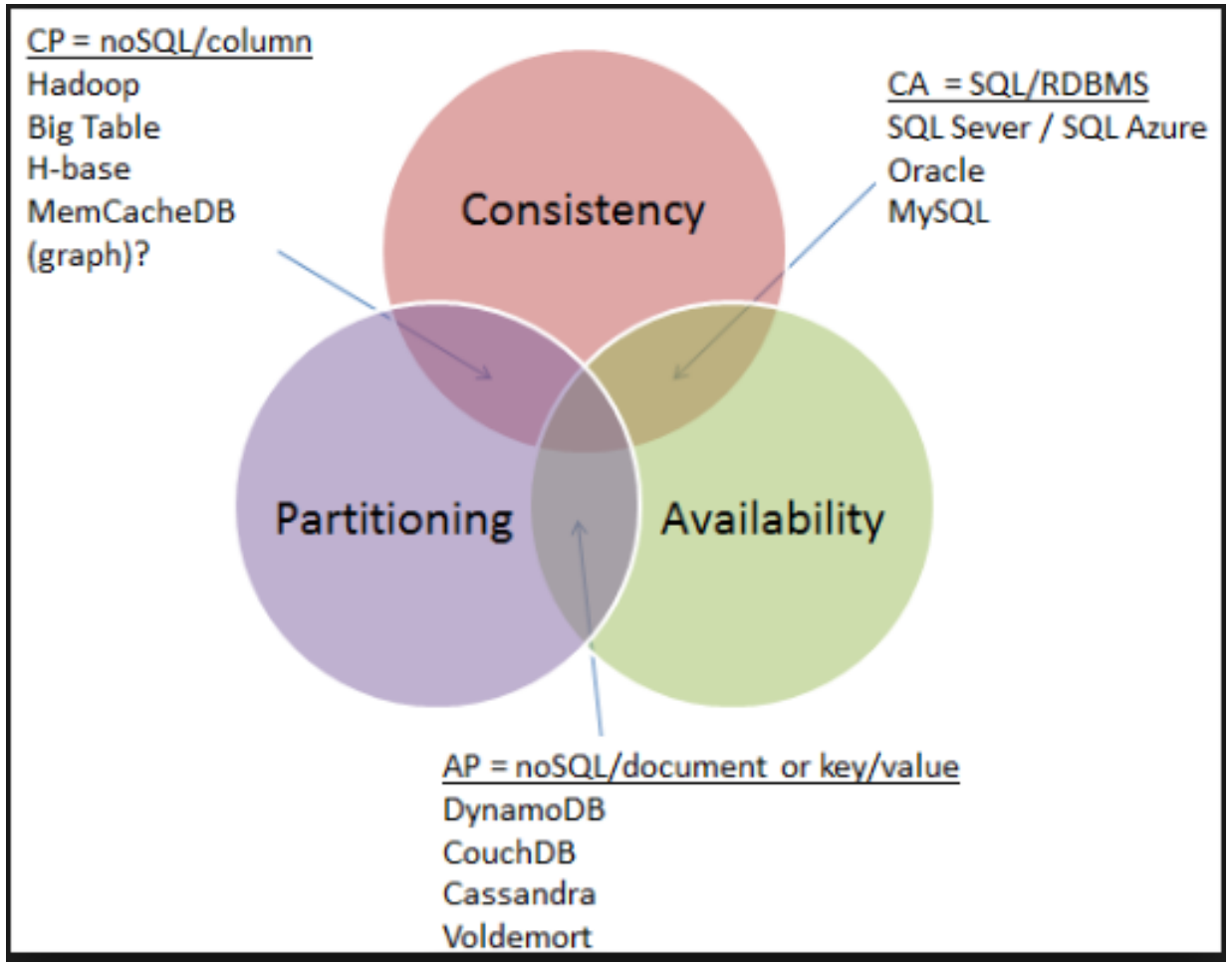
BASE VS. ACID

- Soft State. BASE se aleja del requerimiento de consistencia establecido en ACID que implicaba una única "versión" de cada dato. En BASE la consistencia es un problema de los desarrolladores y por lo tanto no debe ser administrada por las bases de datos.
- Eventual Consistency. El único requisito que tienen las bases de datos NoSQL en términos de consistencia es que los datos deben converger a un único estado en algún momento del futuro. Sin embargo, no hay garantías de en que momento va a ocurrir esto.

TEOREMA CAP

- Fue publicado por primera vez en el año 1998 y presentado por Eric Brewen en el año 2000 en el ACM Symp.Principles of Distributed Computing (PODC 00)
- El teorema CAP establece que cualquier Sistema que comparta datos a través de una red puede tener a lo sumo dos de las siguientes propiedades:
  - consistencia (C) equivalente a tener una única copia de los datos;
  - Alta disponibilidad (A) de los datos para modificaciones;
  - Tolerancia a las particiones(P).

CAP THEOREM


CP = noSQL/column
Hadoop
Big Table
H-base
MemCacheDB
(graph)?
Consistency
CA = SQL/RDBMS
SQL Sever / SQL Azure
Oracle
MySQL
Partitioning
Availability
AP = noSQL/document or key/value
DynamoDB
CouchDB
Cassandra
Voldemort

CAP HOY

- Según Brewen El objetivo de CAP hoy debería ser encontrar el balance entre consistencia y disponibilidad mas conveniente para cada aplicación, al mismo tiempo que establecer la forma en que se van a manejar las particiones y también la manera en que se van a recuperar los errores que se produzcan durante las mismas

ESTRATEGIA PARA LAS PARTICIONES

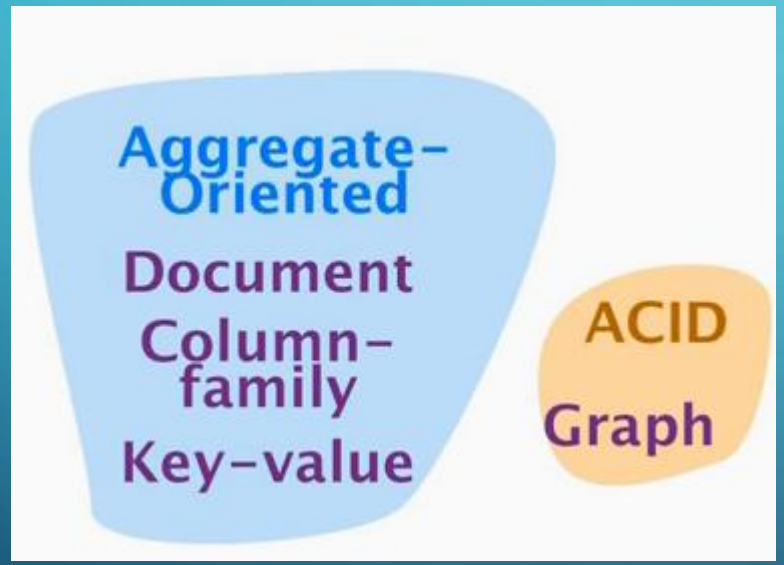
- Cualquier estrategia de este tipo tiene que tener al menos 3 pasos
  - Detectar la partición
  - Entrar en modo “particionado”
  - Recuperar del modo anterior

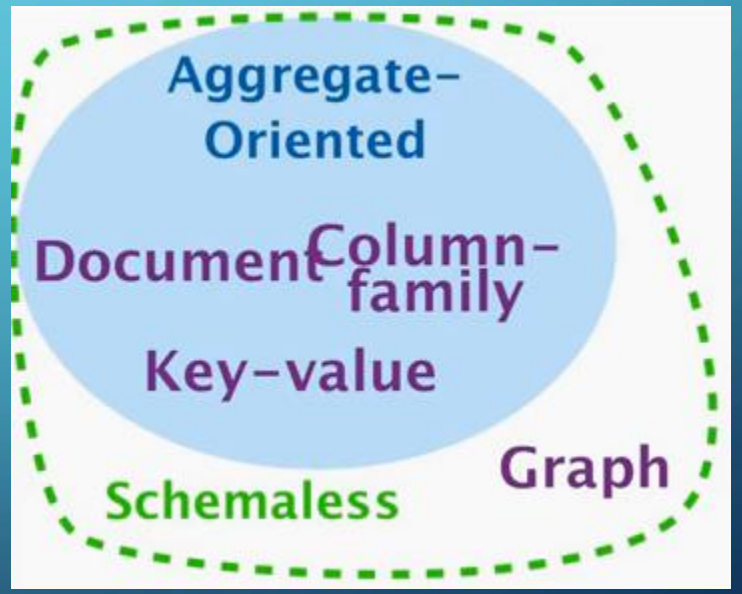
ESTRUCTURA

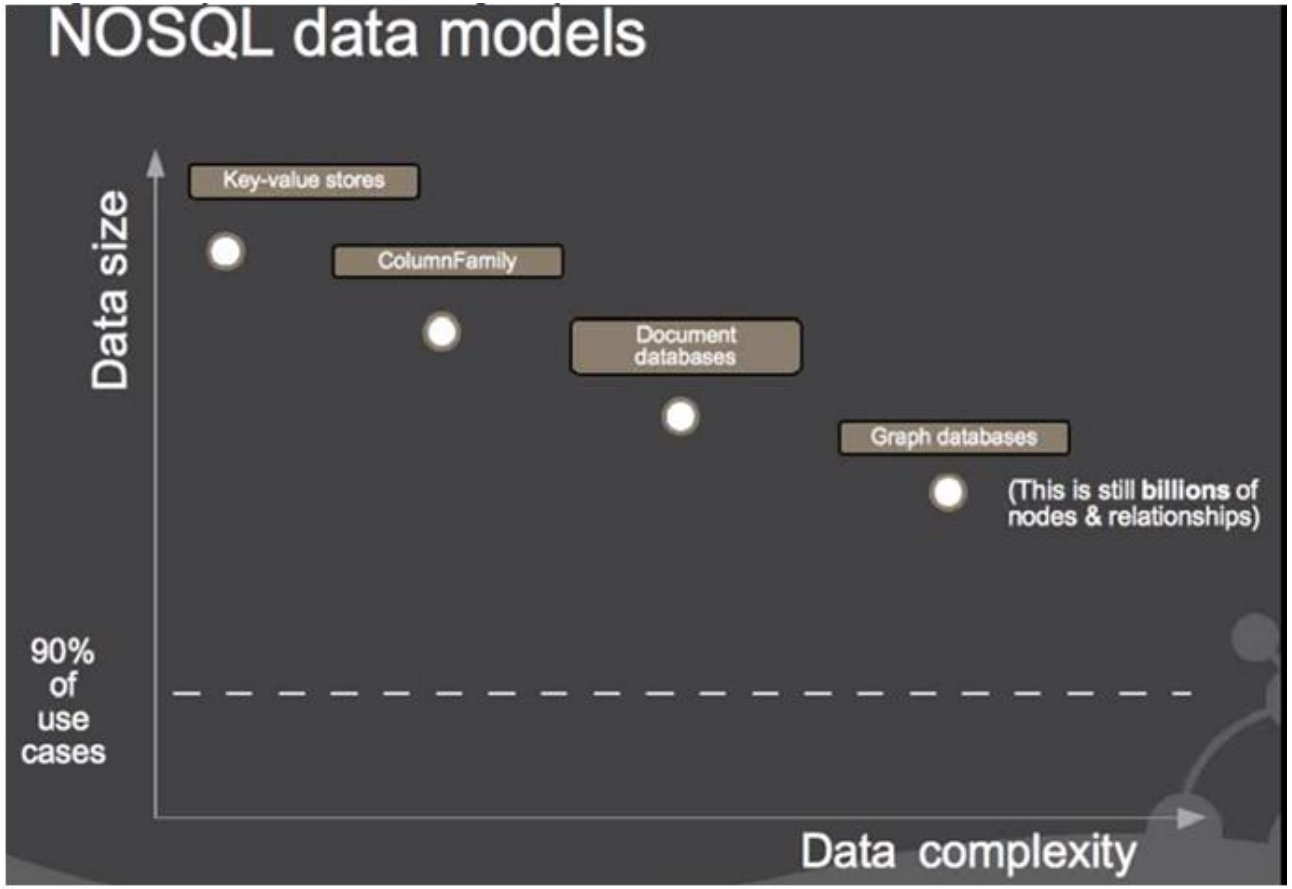
- En lugar de usar tablas para almacenar los datos , las bases de datos NoSQL usan el concepto de almacenamientos del tipo clave/ valor. Es un concepto bastante sencillo donde no hay un esquema (schema) para la base de datos . Simplemente almacena los valores provistos para cada clave y los distribuye a lo largo de la base de datos, permitiendo una recuperacion muy eficiente. Al no tener esquema no pueden realizarse consultas complejas

DIFERENTES TIPOS DE BASES NOSQL

- **Key-Value-stores:** la menor unidad de modelado es un par clave -valor
- **Column Family Databases:** la menor unidad de modelado es una familia de columnas
- **Document-stores:** la menor unidad de modelado es un documento
- **Graph Databases:** modela toda la estructura como un grafo



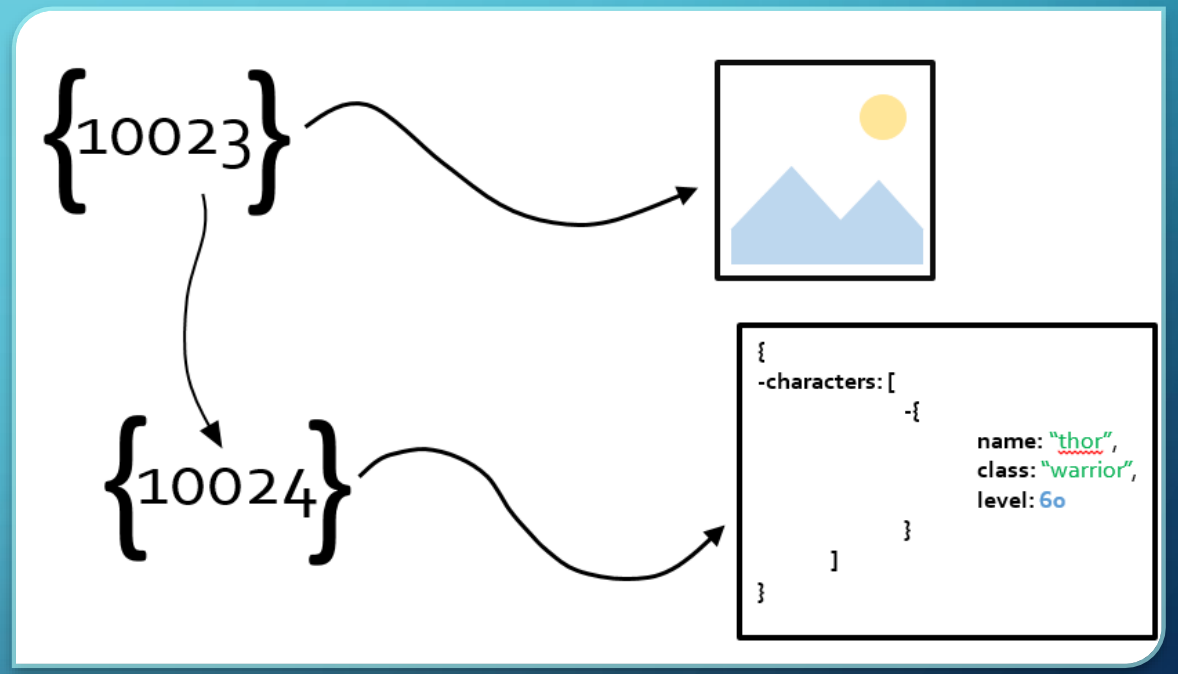




DIFERENTES TIPOS DE BASES NOSQL

CLAVE VALOR

- Es la mas sencilla de las base de datos No SQL. Cada ítem en la base de datos es almacenado como una clave y su valor



CARACTERÍSTICAS DE LAS BASES DEL TIPO CLAVE VALOR

- Los datos guardados en este tipo de base de datos pueden ser de diferentes tipos: texto plano, XML , JSON, imágenes, etc.
- Son mas adecuadas para la web o para aplicaciones mobiles, donde puede ser necesario guardar diferentes objetos.
- Están orientadas a los ítems, y todas la información correspondiente a un ítem se almacena con su clave. Los ítems pueden almacenarse en “dominios” que son similares a las tablas, con la única diferencia que no tienen un schema.
- Los ítems no tienen esquema y por lo tanto pueden guardar objetos muy diferentes.
- Este tipo de estructura puede producir redundancia entre los ítems. Por ejemplo la descripción de un articulo puede estar repetida entre diferentes ítems que de una forma u otra se vinculen con el.
- No existen ningún concepto equivalente al de integridad referencial.

VENTAJAS DE LAS BASES CLAVE - VALOR

- **Adecuadas para la Nube**
  - Al ser simples escalan mucho mejor que las bases relacionales

- **Mas adecuadas para el código orientada a objetos**
  - Las bases de datos relacionales tienen ciertas incompatibilidades con el código orientado a objetos que complican la forma de almacenamiento. Las bases clave valor permiten almacenarlos directamente asociados a un valor

DESVENTAJAS DE LAS BASES CLAVE VALOR

- **Integridad de datos y restricciones**
  - Al no existir mecanismos similiares a los de las bases de datos relacionales para la definicion de restricciones, toda esa administracion queda en manos de los programadores
- **Limitaciones para el analisis**
  - Como los datos carecen de estructura lo mas probale es que para poder hacer análisis sobre los mismos sea necesario pasarlos a una base de datos donde se les pueda dar estructura
- **Escalabilidad vs. Funcionalidad y Complejidad**
  - No hay que olvidar las prioridades de negocio y no tiene sentido poner la escalabilidad por delante de la funcionalidad requerida.

CARACTERISTICAS DE LAS BASES DE DATOS ORIENTADAS A DOCUMENTOS

- Si bien son similares a las bases de tipo calve – valor, tienen diferencias que les permiten una mayor claridad en los datos que tienen almacenados, en detriment de la “libertad” que tienen las bases clave-valor.
- Si bien los “valores” ahora tienen que ser documentos (por ejemplo XML o JSON) , los mismos pueden diferir entre si, tanto en término de datos como de estructura
- Se tiene metadata sobre los datos, lo que permite efecutar indexaciones sobre algunos campos

MONGODB

- **Load balancing:** Mongo DB escala horizontalmente usando sharding. EL usuario establece la clave que determina de que forma los datos van a ser distribuidos. Los datos se cortan en rangos (basados en la clave) y se distribuyen en múltiples shards. Uno de esos shards se convierte en el master , con uno o mas esclavos. Son muy sencillas de configurar

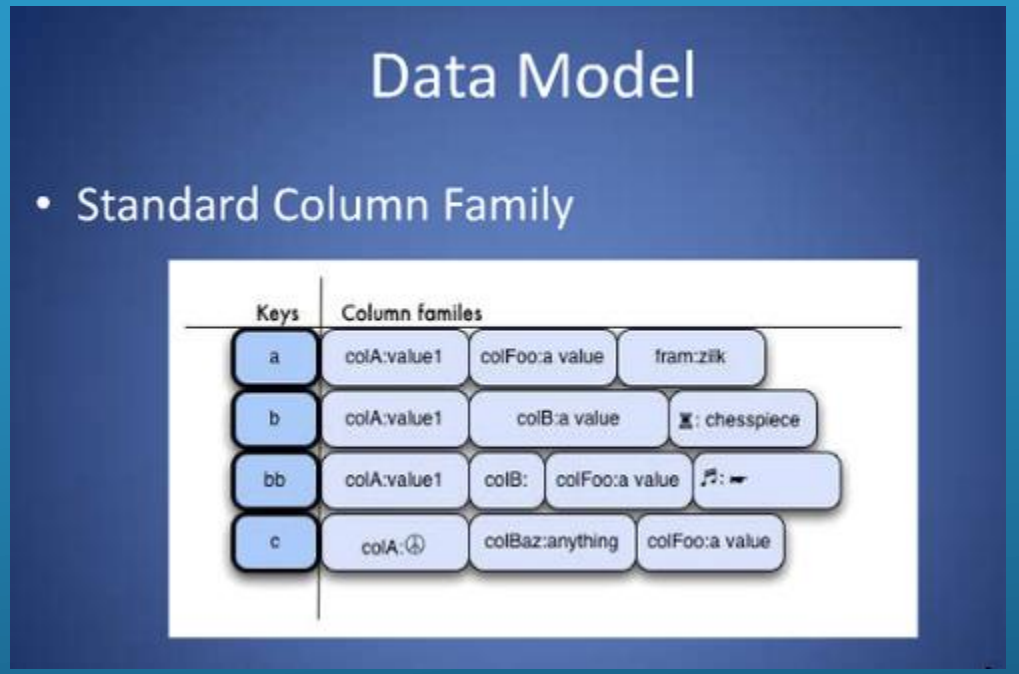
- **Integración :** Mongo DB tiene incorporadas funciones de MapReduce en su API para procesamiento batch. Tiene un framework para manejar agregaciones que le permite a los usuarios obtener resultados similares a los del GROUP BY.

- **Ad hoc queries:** Mongo DB permite búsquedas por campo, por rangos, busquedas de expresiones regulares,.

COLUMN FAMILY DATABASES

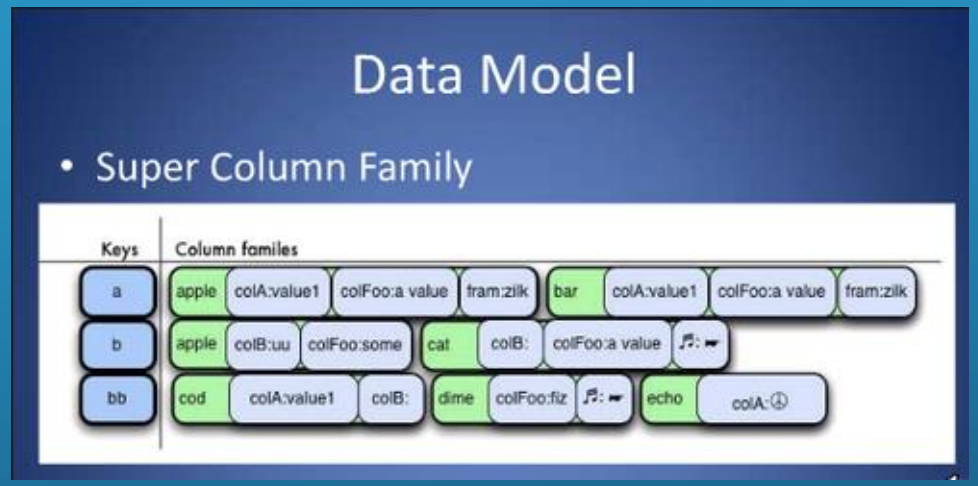
- Son bases de datos consistentes en columnas de datos relacionados. Un grupo de columnas tiene una función similar a las de las tablas en las bases relacionales
- Las “column families” son grupos de datos que, con frecuencia, son accedidos juntos
- Proveen mecanismos naturales de partición vertical
- Tienen un almacenamiento multi dimensional (como mapas o vectores asociativos)

STANDARD COLUMN FAMILY



Son pares clave valor, donde el “valor” esta representado por un conjunto de columnas.

SUPER COLUMN FAMILY



Son pares clave valor, donde el “valor” esta representado por un conjunto de conjuntos de columnas. Haciendo una analogía con las bases relacionales , se podría decir que los “valores” serian similares a las vistas

COLUMN FAMILY DATABASES EXAMPLES

Static column family

row key	columns …			
jbellis	name	email	address	state
	jonathan	jb@ds.com	123 main	TX
dhutch	name	email	address	state
	daria	dh@ds.com	45 2nd St.	CA
egilmore	name	email		
	eric	eg@ds.com		

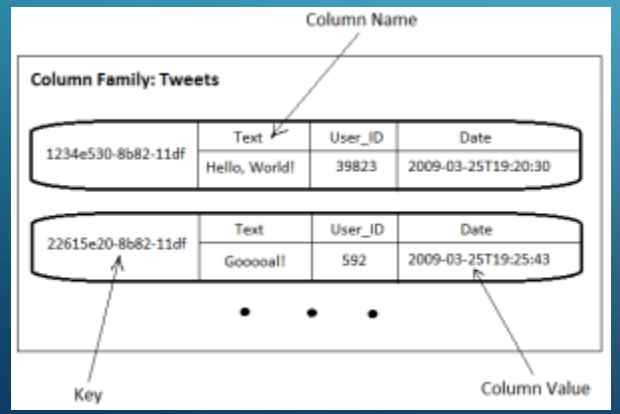
Dynamic column family

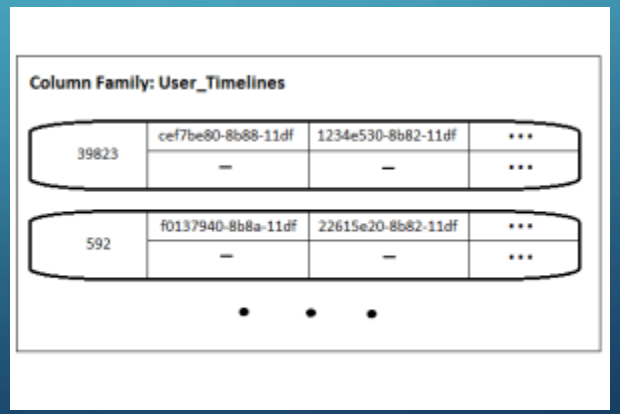
row key	columns …			
jbellis	dhutch	egilmore	datastax	mzcassie
dhutch	egilmore			
egilmore	datastax	mzcassie		

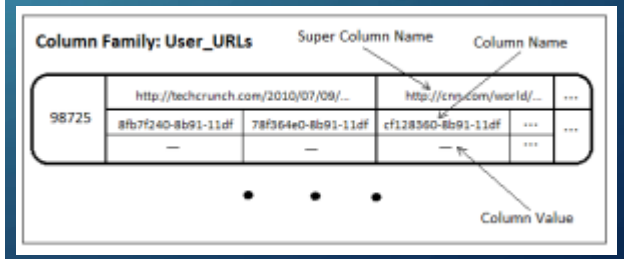
Column

column_name
value
timestamp

Example: Twitter







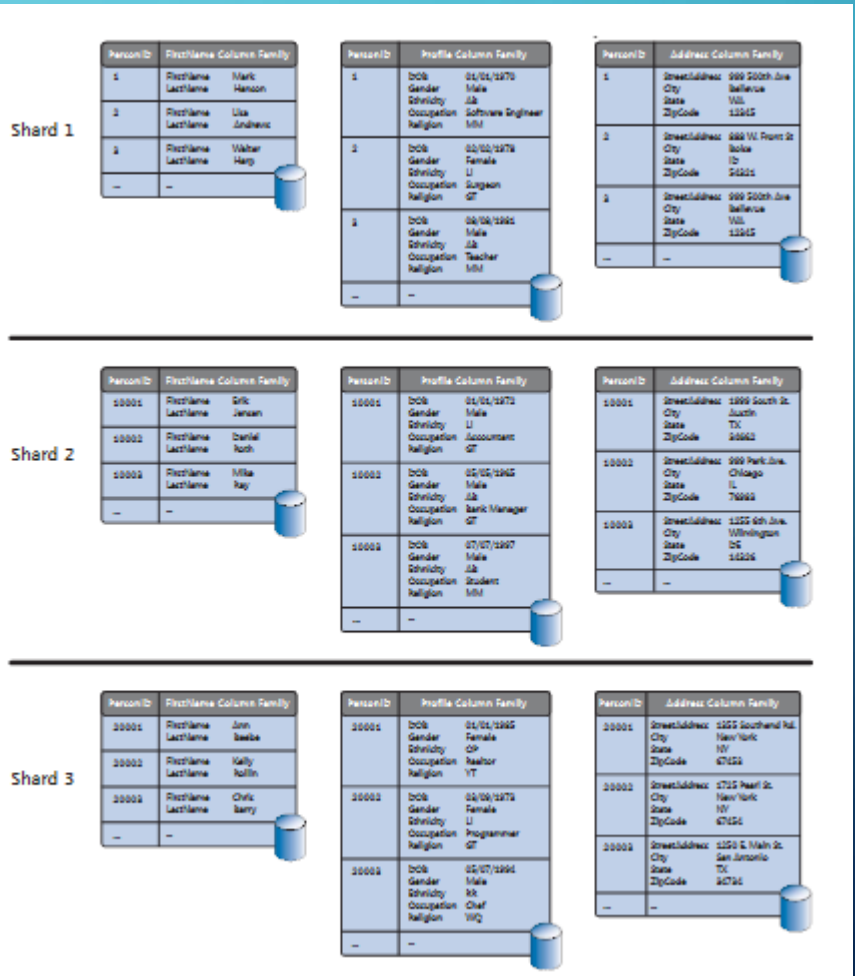
COLUMN FAMILY DATABASES PARTITIONING

Example: Census database

PersonID	FirstName Column Family	
1	FirstName LastName	Mark Hanson
2	FirstName LastName	Lisa Andrews
3	FirstName LastName	Walter Harp

PersonID	Address Column Family	
1	StreetAddress City State ZipCode	999 500th Ave Bellevue WA 12345
2	StreetAddress City State ZipCode	888 W. Front St Boise ID 54321
3	StreetAddress City State ZipCode	999 500th Ave Bellevue WA 12345

PersonID	Profile Column Family	
1	DOB Gender Ethnicity Occupation Religion	01/01/1970 Male AB Software Engineer MM
2	DOB Gender Ethnicity Occupation Religion	02/02/1978 Female LI Surgeon GT
3	DOB Gender Ethnicity Occupation Religion	08/08/1981 Male AB Teacher MM



VENTAJAS Y DESVENTAJAS

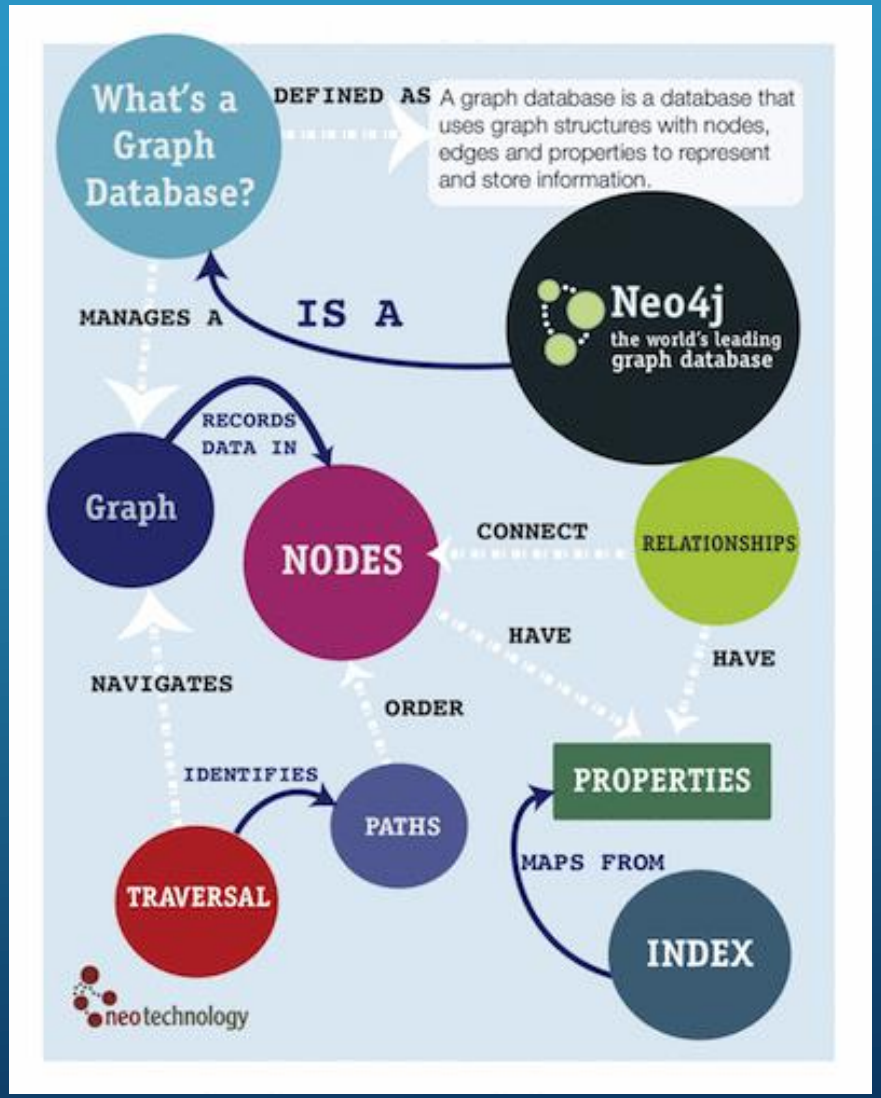
Ventajas:

- Diseñadas para almacenar grandes volúmenes de datos
- Soporta datos semi estructurados
- Permite indexación
- Es escalable y provee alta disponibilidad

Desventanas:

- No son adecuadas para datos relacionales
- No proveen consistencia inmediata

BASE DE DATOS DE GRAFOS NEO4J



BASE DE DATOS DE GRAFOS NEO4J

Neo4j almacena los datos como nodos conectados por medio de relaciones dirigidas y tipadas. Permite almacenar propiedades tanto en los nodos como en las relaciones,

Principales Características

- *Intuitivo*
- *Cumple con las propiedades ACID*
- *Escalable*
- *Alta disponibilidad*
- *Soporta lenguajes de consulta de grafos*
- *Puede ser accedido mediante servicios REST o mediante una interfase API orientada a objetos*
- *Hay problemas que se representan mucho mas naturalmente como grafos*

BASE DE DATOS RELACIONALES Y GRAPH DATABASES

- Tanto SQL Server como Oracle tienen en la actualidad funcionalidad de graph databases
- https://www.youtube.com/watch?v=dgGB8uttlxw
- https://www.red-gate.com/simple-talk/sql/sql-development/sql-server-graph-databases-part-1-introduction/

NOSQL PROS AND CONS

NoSQL PROs	NoSQL CONs
Escalable	Muchas opciones, lo que dificulta la eleccion de la base mas adecuada
Schema flexible	Capacidad de query limitada
Características distribuidas (confiable, escalable, recursos compartidos, velocidad)	La consistencia eventual no es intuitiva, para entornos de negocio como el bancario
No tiene interrelaciones complicadas	No tiene Joins, ni Group by, ni Order by
Menor costo por la facilidad de escalar en maquinas comunes	Carece de ACID
Open Source	Soporte Limitado

RESUMIENDO RDBMS VS NOSQL

Caracteristicas	NoSQL	RDBMS
Volumen	Maneja muchísimo volumen	Limitaciones en el volumen
Validez de los datos	Poco garantizada	Muy garantizada
Escalabilidad	Horizontal	Horizontal y Vertical
Query Language	No tiene lenguaje	Structured Query Language (SQL)
Schema	Sin esquema o con esquemas muy flexibles	Esquemas predefinidos
Tipos de Datos	Soporta tipos de datos no estructurados, no tiene limites	Han incorporado extensiones para permitir el almacenamiento de datos no estructurados
ACID/BASE	En general, BASE	ACID
Administración de transacciones	Garantía débil	Garantía muy fuerte

RESUMIENDO RDBMS VS NOSQL

Opt NoSQL	Opt RDBMS
Si hay muchos datos , no estructurados y muy variables	Si es necesario hacer analisis sobre los datos
Esquema flexible	Esquema rígido
Escalabilidad horizontal, por sobre escalabilidad “vertical”	Permite “scale up” y la escalabilidad horizontal es limitada
Usa hardware sin grandes requerimientos. Fácil de incrementar la capacidad	Servers propietarios , de alto costo. Si no se usan mecanismos de tipo Cloud costosa para “crecer”

REFERENCIAS

- http://nosql-database.org/
- http://www.couchbase.com/why-nosql/nosql-database
- http://www.linuxjournal.com/article/3294
- http://www.christof-strauch.de/nosqldbs.pdf
- http://databases.about.com
- http://www.analyticsvidhya.com/blog/2014/05/hadoop-simplified/
- https://www.usenix.org/legacy/events/lisa11/tech/slides/stonebraker.pdf 32