CEU
Universidad
San Pablo

Tema 4: Introducción a las BBDD *NoSQL*

Bases de Datos I

Guillermo de la Calle Velasco
Universidad San Pablo CEU
Madrid

TEMA 4: Introducción a las Bases de Datos NoSQL

NoSQL → No sólo SQL / *Not Only SQL*

- Surgen por la necesidad de almacenar datos masivamente
  - Web 2.0 (Facebook, Twitter, Youtube…)
- Las BBDD NoSQL permiten almacenar información en determinados escenarios donde las BBDD relacionales clásicas tienen problemas:
  - Escalabilidad
  - Rendimiento (accesos concurrentes elevados)
- Las BBDD NoSQL no cumplen con el esquema Entidad Relación, ni están estructuradas en "tablas" sino que utilizan otros formatos "clave-valor", "por columnas" o grafos, no permiten *JOINS,* no intentan garantizar ACID (transacciones)

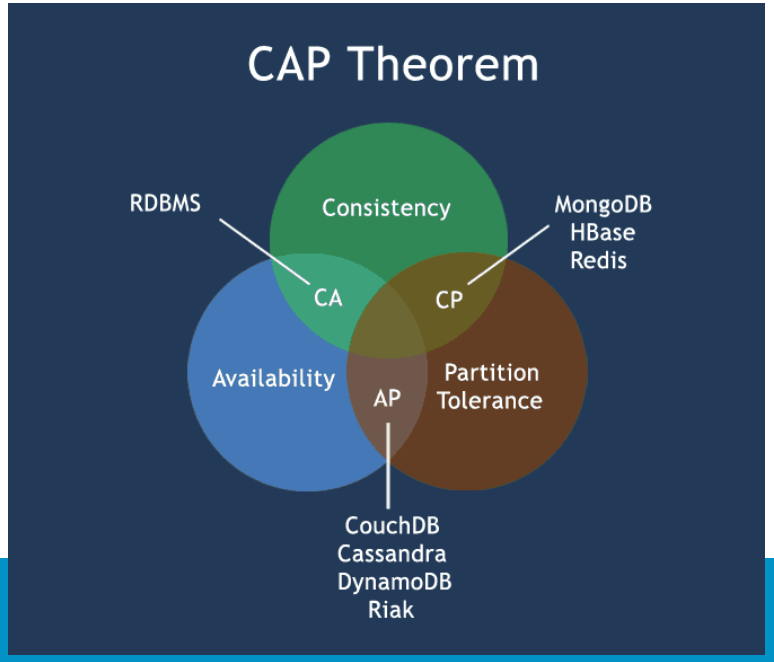
Ventajas de los Sistemas **NoSQL**

- Pueden ejecutar en máquinas con pocos recursos
  - No requieren gran capacidad de cómputo, a diferencia de los sistemas relacionales
  - Coste de implantación más reducido
- Escalabilidad horizontal
  - Añadir más nodos (clústeres) para obtener mayor rendimiento
- Manejo de gran cantidad de datos
  - Estructuras distribuidas
  - Tablas *Hash*
- No generan "cuellos de botella"
  - Los sistemas SQL necesitan hacer muchas comprobaciones
  - Punto de entrada común → muchas peticiones pueden ralentizar el sistema

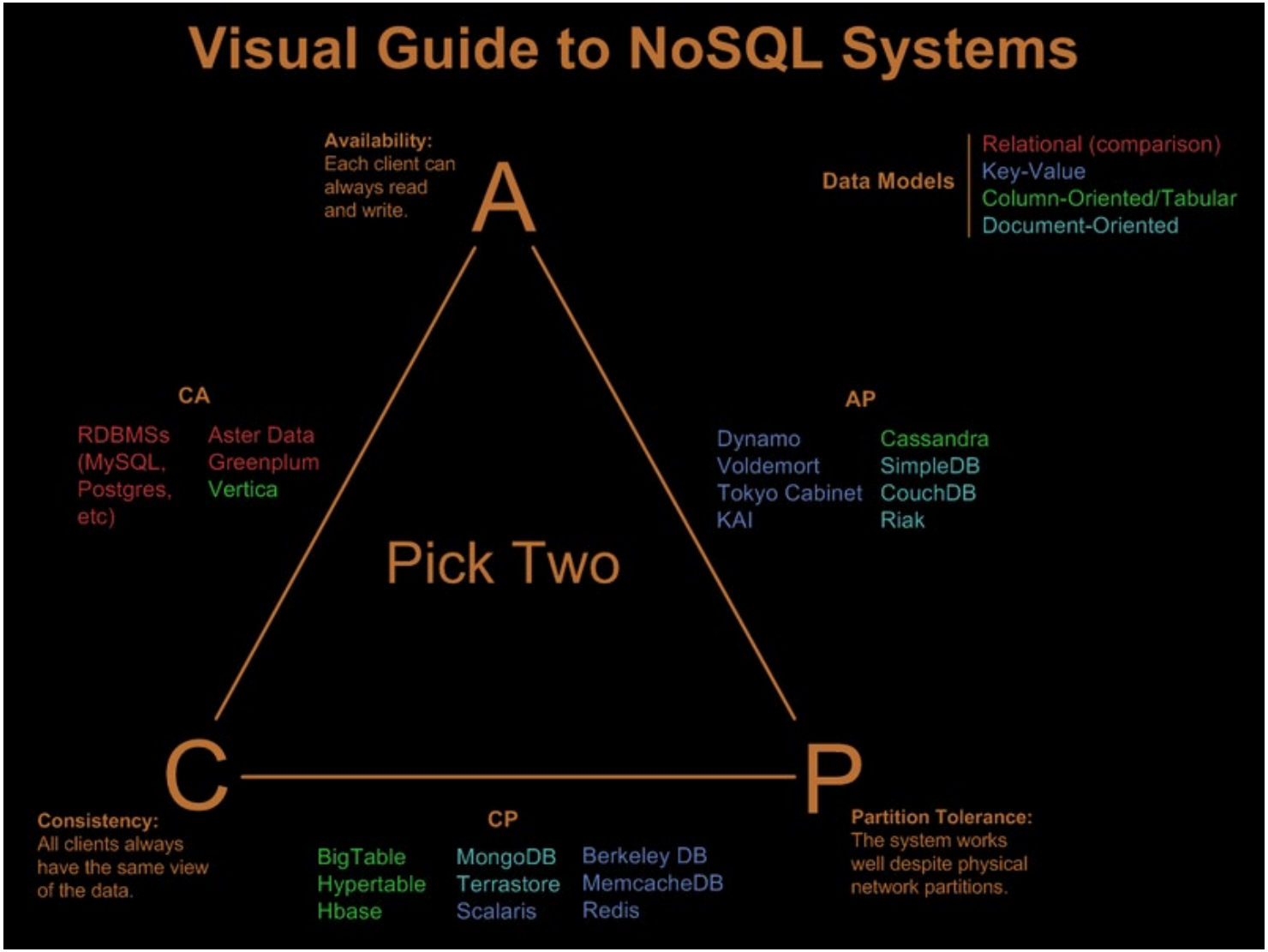
Teorema CAP (Brewer)

"Es imposible para un sistema computacional distribuido ofrecer simultáneamente las siguientes tres garantías"

- *Consistencia*. Todos los nodos ven los mismos datos al mismo tiempo.
- *Disponibilidad (Availability):* garantiza que cada petición recibe una respuesta
- *Tolerancia a la partición (Partition):* sigue funcionando a pesar de pérdida de mensajes



"You can have it good, you can have it fast, you can have it cheap: pick two"


Visual Guide to NoSQL Systems
Availability:
Each client can always read and write.
A
Data Models
Relational (comparison)
Key-Value
Column-Oriented/Tabular
Document-Oriented
CA
RDBMSs (MySQL, Postgres, etc)
Aster Data
Greenplum
Vertica
AP
Dynamo
Voldemort
Tokyo Cabinet
KAI
Cassandra
SimpleDB
CouchDB
Riak
Pick Two
C
P
Consistency:
All clients always have the same view of the data.
CP
BigTable
Hypertable
Hbase
MongoDB
Terrastore
Scalaris
Berkeley DB
MemcacheDB
Redis
Partition Tolerance:
The system works well despite physical network partitions.

Principales diferencias con las BBDD SQL

- No utilizan *SQL* como lenguaje de consulta
  - Muchos sistemas definen su propio lenguaje de consulta o utilizan SQL como lenguaje de apoyo
- Estructuras flexibles
  - La información no se estructura en tablas, sino en pares clave-valor, objetos o grafos
- No suelen permitir operaciones ***JOIN***
  - Resultan inviables por la cantidad de datos manejados
  - Desnormalización
- Arquitectura distribuida
  - Los sistemas tradicionales suelen estar centralizados

Tipos de bases de datos NoSQL

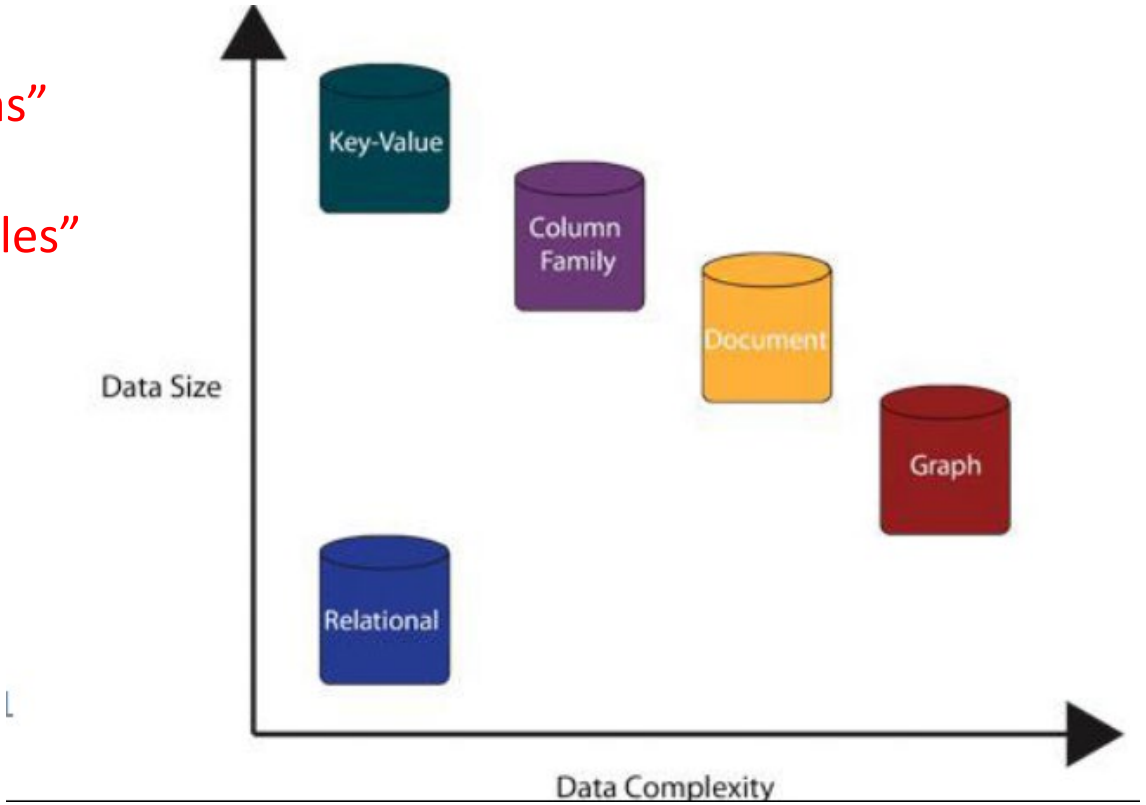
- Habitualmente, se definen 4 tipos de bases de datos *NoSQL*, en función de la forma como almacenan la información

  a) Bases de datos *"clave-valor"*

  b) Bases de datos "de columnas"

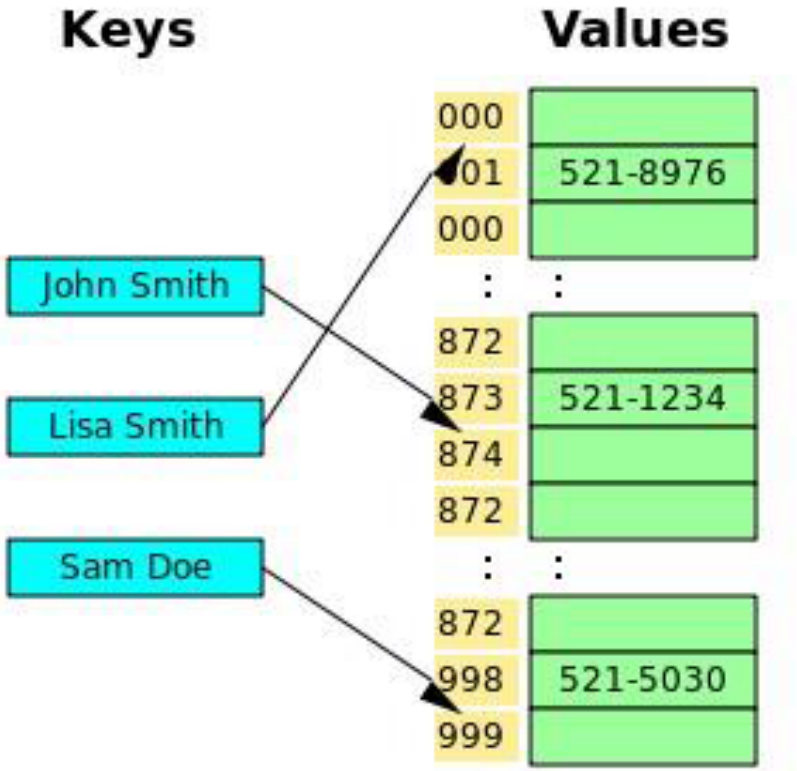
  c) Bases de datos "documentales"

  d) Base de datos "de grafos"



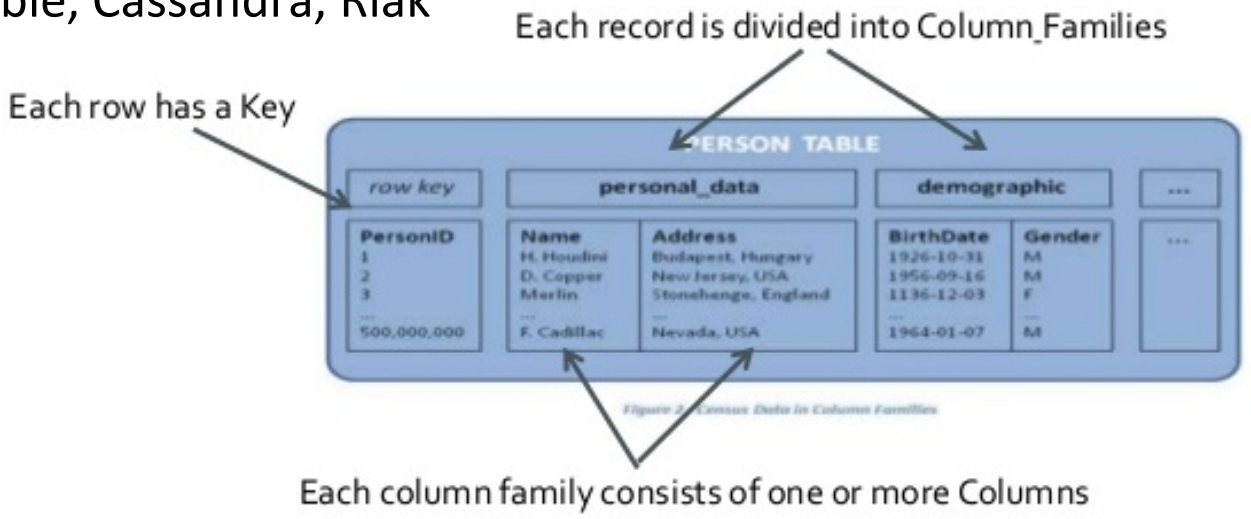
Bases de datos “clave-valor”

- Modelo más sencillo y extendido (popular)
- Precursor: Amazon Dynamo (*DHT - Distributed Hash Tables)*
- Modelo de datos: colección de pares “clave/valor”. Cada elemento se identifica por una clave única
- Acceso y recuperación de información muy rápidas
- Muy eficientes tanto en lecturas como escrituras
- Ejemplos: Dynomite, Voldemort, Tokyo, Cassandra, BigTable, HBase…



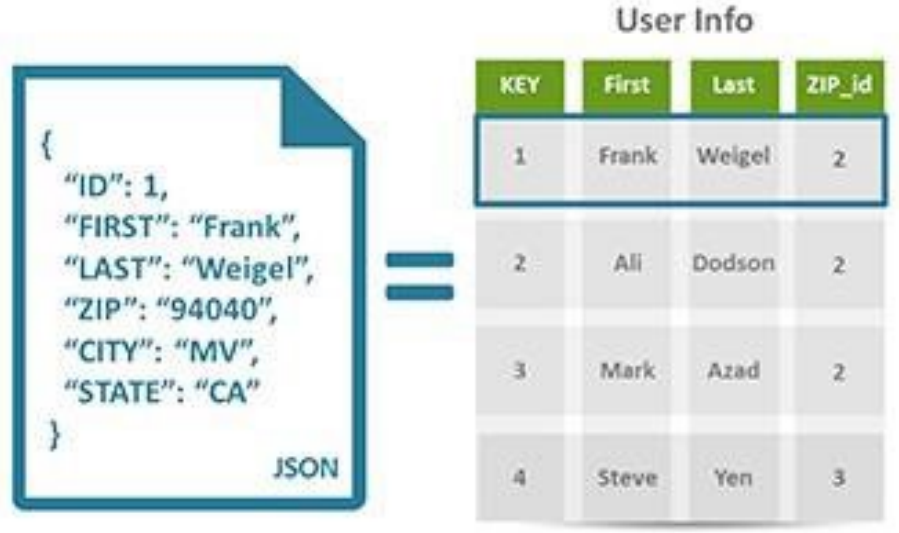
Bases de datos “de columnas”

- Precursor: *Google BigTable*
- Modelo de datos: modelo tabular donde cada fila puede tener una configuración diferente de columnas. Se guardan los datos por columnas en lugar de por filas (RDBMS)
- Ejemplos: HBase, Hypertable, Cassandra, Riak
- Buenas para:
  - Gestión de tamaño
  - Alta disponibilidad
  - MapReduce



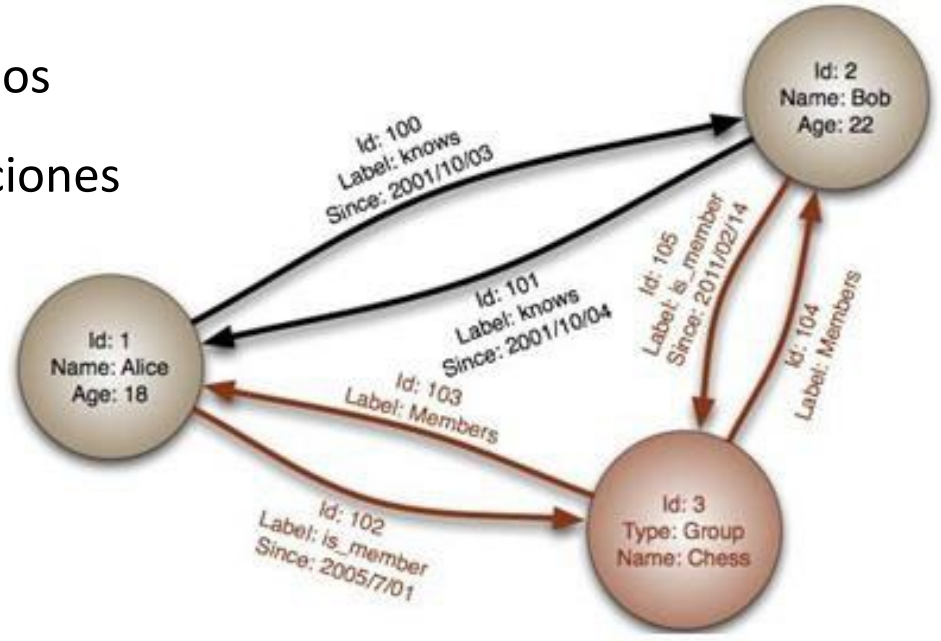
Bases de datos “documentales”

- Inspiradas en *Lotus* Notes. La información se almacena como un documento
- Suelen utilizar una estructura simple basada en *JSON* o *XML*
- Además de permitir consultas por “clave-valor”, permiten realizar consultas más avanzadas sobre el contenido de los documentos
- Quizás las más versátiles: modelado de datos natural, amigables para el programador, desarrollo rápido, CRUD
- Ejemplos: MongoDB, CouchDB



Bases de datos “de grafos”

- Inspiradas por *Euler* y la teoría de grafos
- La información se representa como nodos de un grafo y las aristas representan las relaciones
- Uso de teoría de grafos para recorridos
- Navegación más eficiente entre relaciones
- Ejemplos: Neo4j, InfoGrid, Virtuoso, AllegroGraph, VertexDB



SQL vs NoSQL

- En general, *NoSQL* permite recuperar datos más rápidamente que los sistemas *RDBMS*, pero el tipo de consultas es limitado (la complejidad se traslada a la aplicación)
- *NoSQL* a menudo no soporta ACID (Transacciones) ni bloqueos
- Los sistemas *NoSQL* suelen ser más complejos de instalar y gestionar
- En sistemas grandes, debería considerarse la combinación de *SQL* y *NoSQL*
  - *LinkedIn* comenzó utilizando sólo *RDBMS*, pero terminó desarrollando su propia base de datos *NoSQL (Voldemort)*
  - *Facebook* tiene una arquitectura híbrida donde se combinan diferentes tecnologías de persistencia de datos: *Memcached, MySQL, Cassandra, Hbase…*

- Algunas razones para seleccionar una base de datos NoSQL frente a los clásicos SGBD relacionales basados en SQL pueden ser:
  - Cuando el volumen de los datos crece muy rápidamente en momentos puntuales, pudiendo llegar a superar el Terabyte de información
  - Cuando la escalabilidad de la solución relacional no es viable tanto a nivel de costes como a nivel técnico
  - Cuando tenemos elevados picos de uso del sistema por parte de los usuarios en múltiples ocasiones
  - Cuando el esquema de la base de datos no es homogéneo, es decir, cuando en cada inserción de datos la información que se almacena puede tener campos distintos

Ejemplos de bases de datos NoSQL

Ejemplos de bases de datos NoSQL

Fuente: http://db-engines.com/en/ranking

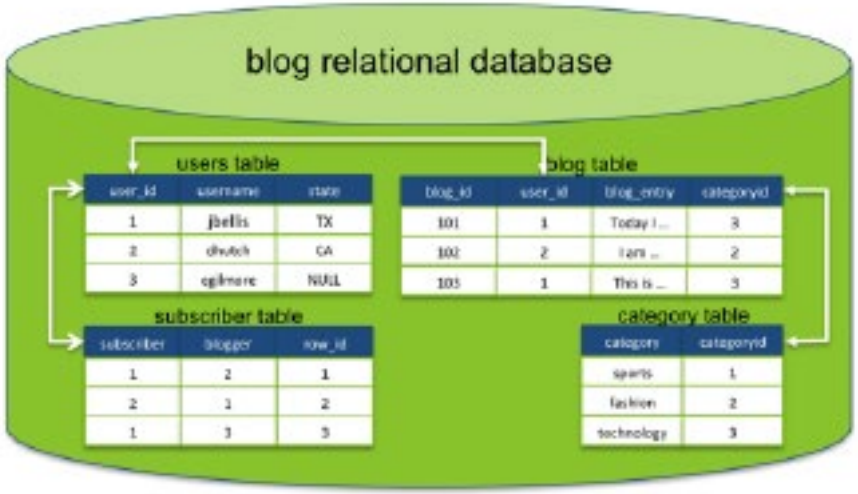
381 systems in ranking, December 2021

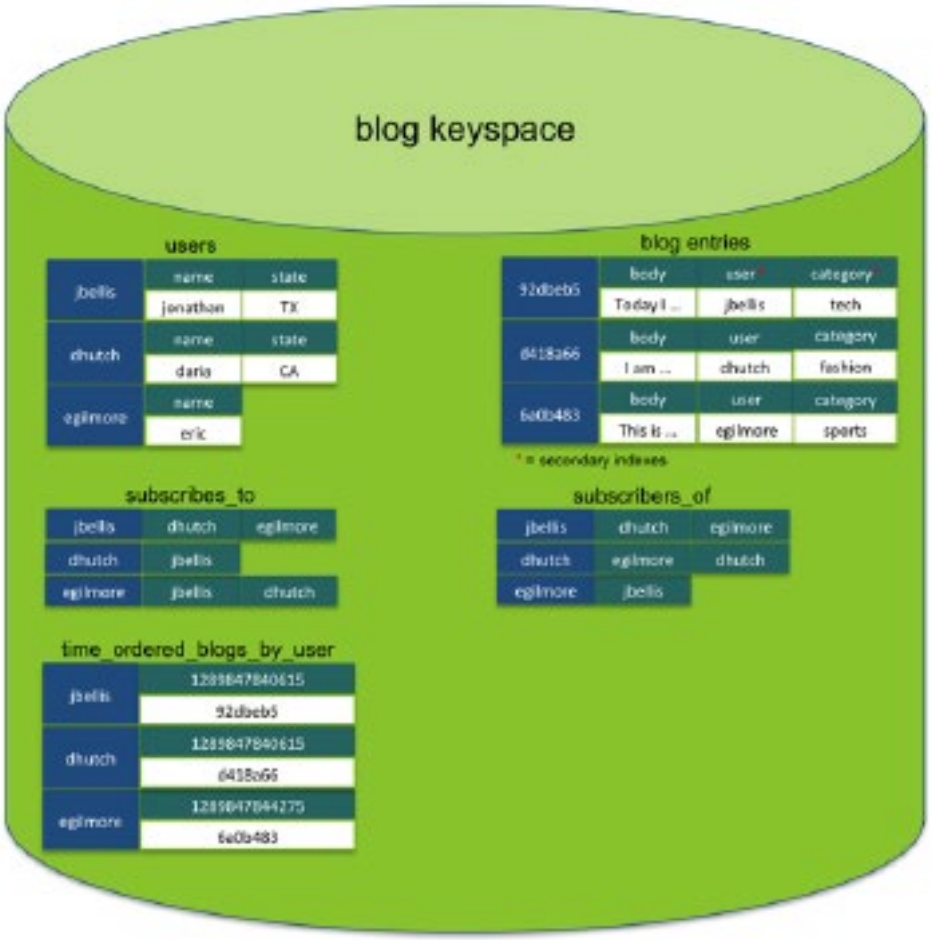
Rank			DBMS	Database Model	Score		
Dec 2021	Nov 2021	Dec 2020			Dec 2021	Nov 2021	Dec 2020
1.	1.	1.	Oracle	Relational, Multi-model	1281.74	+9.01	-43.86
2.	2.	2.	MySQL	Relational, Multi-model	1206.04	-5.48	-49.41
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	954.02	-0.27	-84.07
4.	4.	4.	PostgreSQL	Relational, Multi-model	608.21	+10.94	+60.64
5.	5.	5.	MongoDB	Document, Multi-model	484.67	-2.67	+26.95
6.	6.	↑ 7.	Redis	Key-value, Multi-model	173.54	+2.04	+19.91
7.	7.	↓ 6.	IBM Db2	Relational, Multi-model	167.18	-0.34	+6.74
8.	8.	8.	Elasticsearch	Search engine, Multi-model	157.72	-1.36	+5.23
9.	9.	9.	SQLite	Relational	128.68	-1.12	+7.00
10.	↑ 11.	↑ 11.	Microsoft Access	Relational	125.99	+6.75	+9.25
11.	↓ 10.	↓ 10.	Cassandra	Wide column	119.20	-1.68	+0.36
12.	12.	12.	MariaDB	Relational, Multi-model	104.36	+2.17	+10.75
13.	13.	13.	Splunk	Search engine	94.32	+2.02	+7.32
14.	↑ 15.	↑ 16.	Microsoft Azure SQL Database	Relational, Multi-model	83.25	+1.93	+13.76
15.	↓ 14.	15.	Hive	Relational	81.93	-1.38	+11.66
16.	16.	↑ 17.	Amazon DynamoDB	Multi-model	77.63	+0.64	+8.51
17.	↑ 18.	↑ 41.	Snowflake	Relational	71.03	+6.84	+58.12
18.	↓ 17.	↓ 14.	Teradata	Relational, Multi-model	70.29	+0.71	-3.54
19.	19.	19.	Neo4j	Graph	58.03	+0.05	+3.40
20.	↑ 22.	↑ 21.	Solr	Search engine, Multi-model	57.72	+3.87	+6.48
21.	↓ 20.	↓ 20.	SAP HANA	Relational, Multi-model	54.58	-0.95	+2.08
22.	↓ 21.	22.	FileMaker	Relational	53.86	-0.36	+6.16
23.	23.	↓ 18.	SAP Adaptive Server	Relational, Multi-model	51.39	+0.45	-3.50
24.	24.	24.	Google BigQuery	Relational	45.81	+0.80	+10.05
25.	25.	↓ 23.	HBase	Wide column	45.54	+0.53	-1.38
26.	26.	↓ 25.	Microsoft Azure Cosmos DB	Multi-model	39.71	-1.11	+6.17
27.	27.		PostGIS	Spatial DBMS, Multi-model	32.44	+0.52	
28.	28.	↓ 26.	Couchbase	Document, Multi-model	28.45	-1.42	-3.37
29.	29.	↓ 27.	InfluxDB	Time Series, Multi-model	28.38	-0.16	+2.24

Apache Cassandra (http://cassandra.apache.org/)

- Es una base de datos tipo "clave-valor" creada por *Apache*
- Desarrollada en Java (código abierto) → multiplataforma con la *JVM*
  - Documentación: http://docs.datastax.com/en/cassandra/1.2/
- Dispone de un lenguaje propio de consultas *CQL (Cassandra Query Language)*
- "Desventajas":
  - No hay *JOINS* → más rápido
  - No hay integridad referencial
  - No permite ordenar resultados en tiempo de consulta
- Compañías usándolo: Facebook, Twitter, Netflix, Digg, Rackspace…

Ejemplos de bases de datos NoSQL





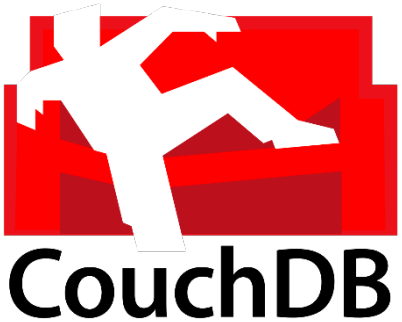
Redis

- Base de datos tipo "clave-valor"
- Como un array gigante **en memoria** para almacenar datos de tipo cadena, *hashes*, conjuntos de datos o listas
- Operaciones atómicas y persistentes.
- Replicación de datos Maestro-Esclavo
- Desarrollada en ANSI C, funciona en plataformas *UNIX, Linux, Solaris, OS/X*. No soporte para Windows
- Compañías usándolo: Flickr, Instagram, Github…

mongoDB

- Base de datos orientada a documentos ("hu**mongo**us" → enorme)
- De esquema libre; cada elemento puede tener un esquema de datos diferente al resto
- Sistema rápido (escrito en C++) que utiliza un sistema propio de documento (BSON – *Binary JSON*) que es una evolución del *JSON* pero admitiendo datos binarios
- No implementa ACID (transacciones), ni *JOINS*
- Problemas de escalabilidad a partir de 100Gb
- Compañías usándolo: FourSquare, SourgeForge, CERN…

CouchDB



- Creado por *Apache* para sistemas tipo *POSIX* (NO *Windows*)
- Base de datos orientada a documentos
- Utiliza servicios REST (Restfull HTTP API) como interfaz y *Javascript* como principal lenguaje de interacción
- La información se almacena en *JSON*
- Muy escalable, alta disponibilidad y robustez, incluso cuando se ejecuta en máquinas convencionales
- Compañías usándolo: BBC, Credit Suisse

Neo4j

- Base de datos orientada a **grafos**
- Apropiada cuando se quieren modelar datos que están conectados (recomendaciones, redes sociales, datos geoespaciales…)
- Alta disponibilidad, soporte para transacciones (ACID), escalable
- Servidor con una API REST o Java API
- Compañías usándolo: Infojobs…