2021/2022

BASI DI DATI E CONOSCENZA

Introduzione alle tecnologie NoSQL

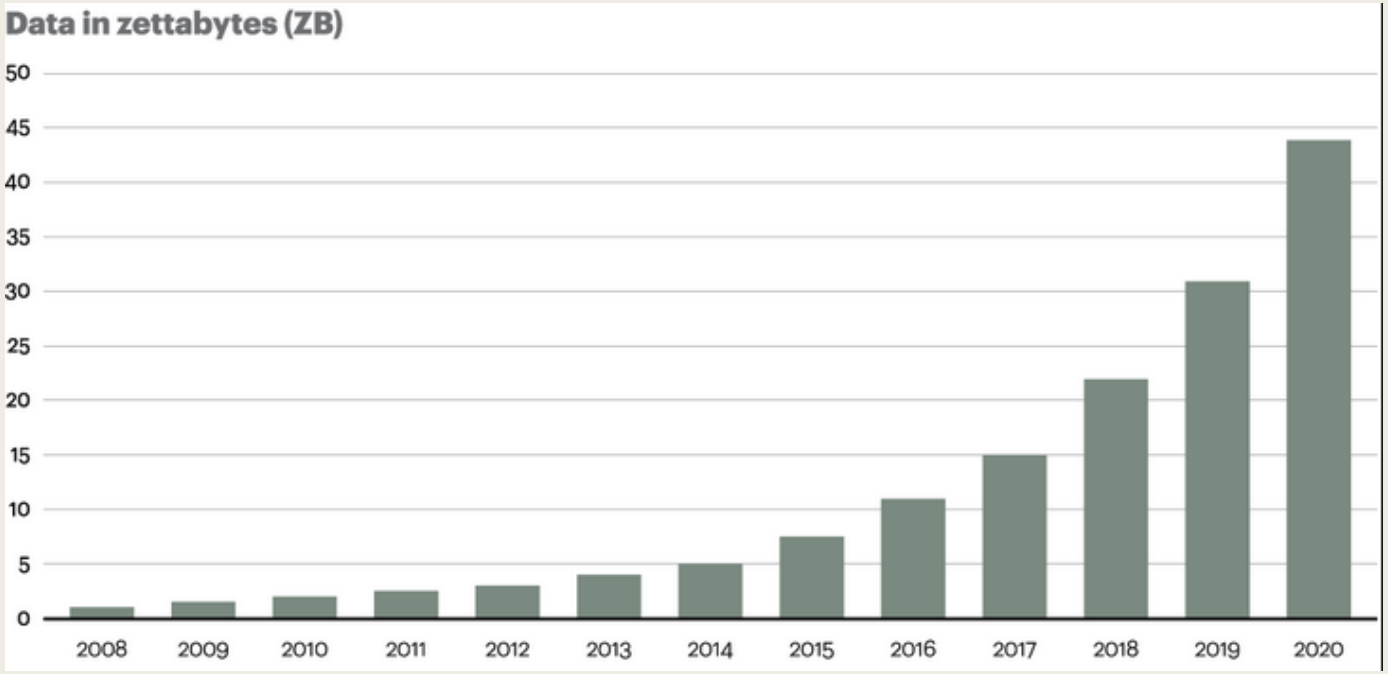
(D. Margiotta)

RDBMS

- Schema predefinito per lo storage di dati strutturati
- Maturi e accuratamente testati
- Facile adozione/integrazione
- Basati sulle proprietà ACID
- Strong consistency
- Data Retrieval: Standard Query Language (SQL)
- Scalabilità verticale

The Internet of Things (1)

Nel corso degli anni l'ammontare dei dati disponibili sul web è in rapida crescita e la natura di questi dati sta cambiando in conseguenza all'utilizzo che gli sviluppatori fanno di nuovi tipi di dati.



Zettabyte	ZB	10^{21}

The Internet of Things (2)

- Dal 2021 ci sono più di 45zb di dati sul web, Si tratta di dati prodotti da fabbriche, ospedali, startup. Ci si connette da casa o dall'auto. Ci si connette dallo smarthphone e dal tablet. Si ricevono continuamente dati su ambiente, posizioni geografiche, spostamenti, temperature, meteo da oltre 50 bilioni di sensori.
- La chiave è l'accesso globale Real-Time.
- I dati relativi alle telemetrie sono di piccole dimensioni, semi-strutturati e continui.
- Si tratta di una grossa sfida per i database relazionali i quali richiedono dati strutturati e con uno schema rigido
- I NoSQL raccolgono questa sfida.

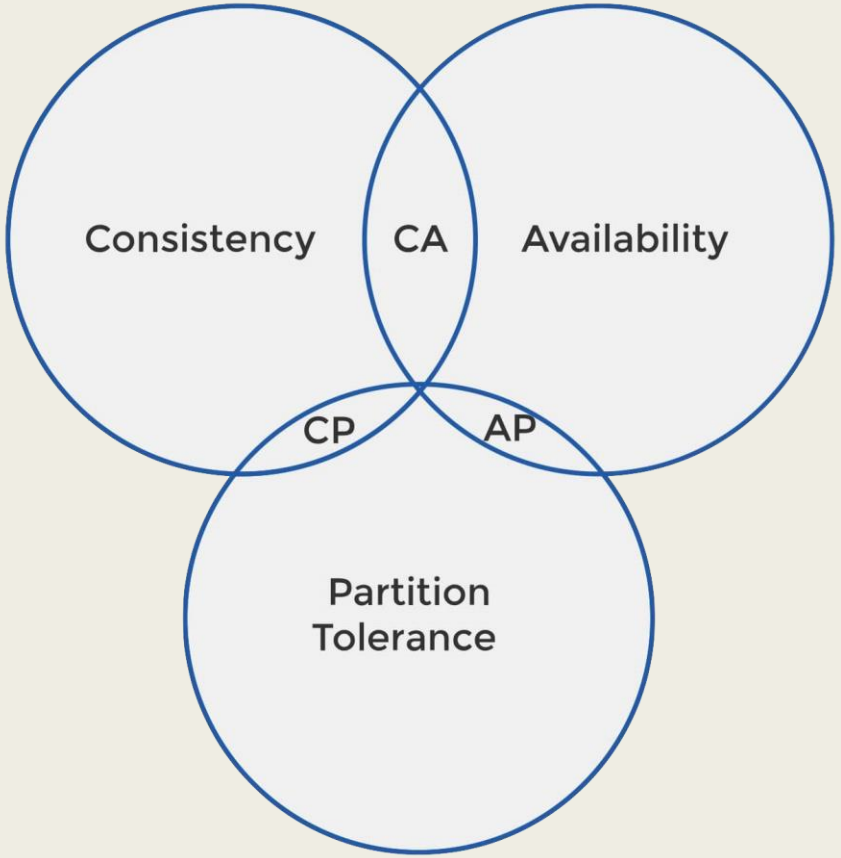
Caratteristiche dei NoSQL

- **Non relazionali**: l'approccio rigido dei DB relazionali non permette di memorizzare dati fortemente dinamici. I DB NoSQL sono "schemaless" e consentono di memorizzare attributi senza averli definiti a priori.
- **Distribuiti**: la flessibilità nella clusterizzazione e nella replicazione dei dati permette di distribuire su più nodi lo storage, in modo da realizzare potenti sistemi "fault tolerant".
- **Scalabili orizzontalmente**: in contrapposizione alla scalabilità verticale, abbiamo architetture enormemente scalabili, che consentono di memorizzare e gestire una grande quantità di informazioni.
- **Open-source**: filosofia alla base del movimento NoSQL.
- **Big-Data**: capacità di gestire grosse quantità di dati.
- **No DBAs**: un sistema RDBMS di alto livello può essere mantenuto solamente con l'assistenza di amministratori altamente esperti (e costosi). I Database NoSQL sono richiedono una minore manutenzione.
- **Great speed**: molto veloce nelle risposte delle query.
- **BASE**: le proprità ACID non sono richieste.
- **CAP Theorem.**

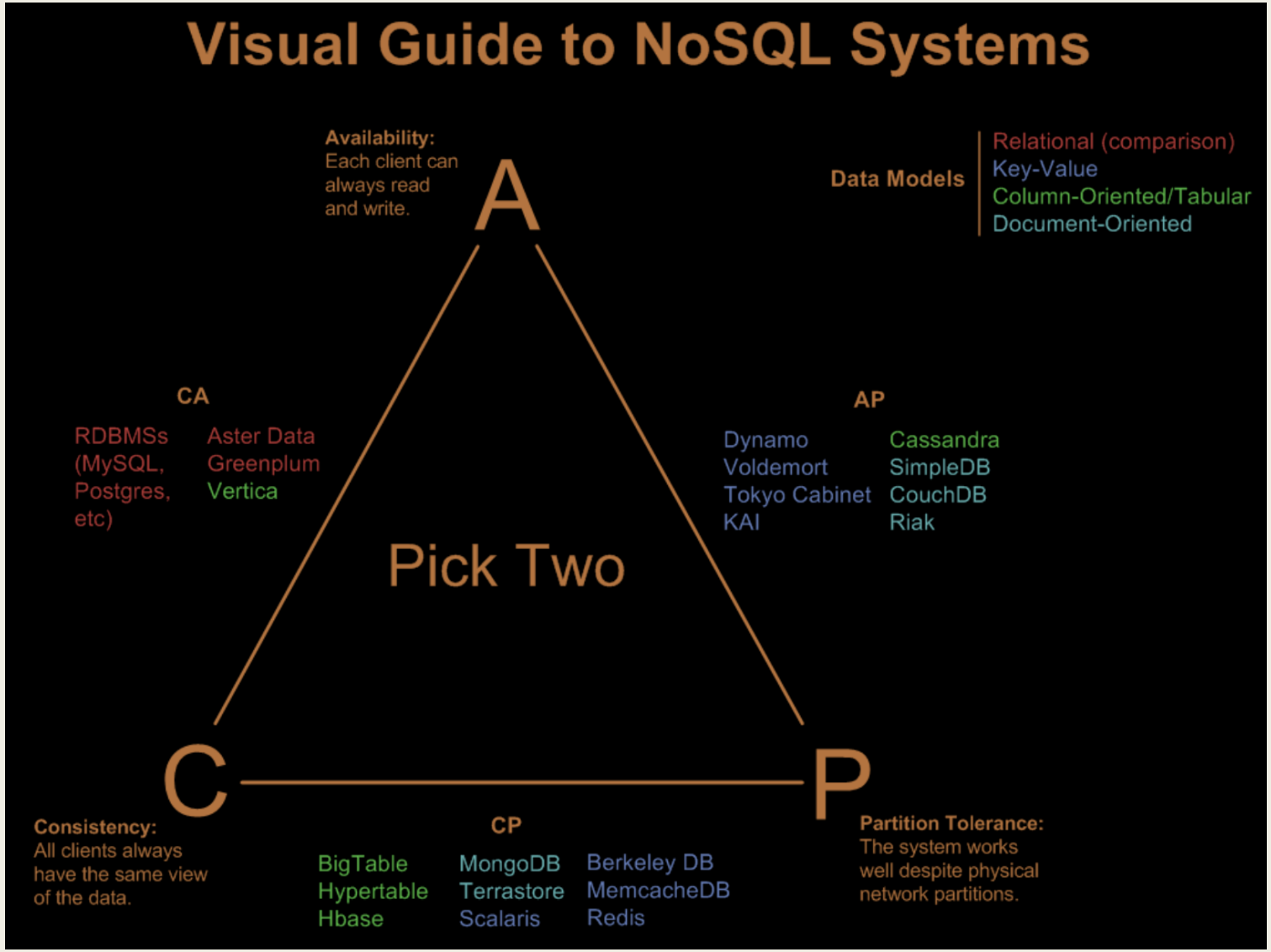
Brewer’s CAP Theorem (1)

Un sistema distribuito è in grado di supportare solamente due tra le seguenti caratteristiche:

- **Consistency**: tutti i nodi vedono lo stesso dato allo stesso tempo.
- **Availability**: ogni operazione deve sempre ricevere una risposta.
- **Partition Tolerance**: capacità di un sistema di essere tollerante ad una aggiunta, una rimozione di un nodo nel sistema distribuito o alla non disponibilità di un componente singolo.



Brewer's CAP Theorem (2)



ACID vs BASE

- La scalabilità e le migliori prestazioni del NoSQL si ottengono sacrificando la compatibilità con le proprietà ACID.

Atomic, Consistent, Isolated, Durable

- Il NoSQL ha invece la compatibilità con le proprietà BASE.

Basically Available, Soft state, Eventual consistency

Recap ACID - Requirement for SQL DBs

- **A**tomic : Tutte le operazioni nella transazione saranno completate, oppure nessuna lo sarà.
- **C**onsistent: Le transazioni non portano mai a risultati con dati incoerenti.
- **I**solated: La transazione si comporta come se fosse l'unica operazione eseguita sul database (cioè le transazioni non impegnate sono isolate).
- **D**urable: Al completamento della transazione, l'operazione non sarà invertita (le transazioni impegnate sono permanenti).

BASE - Requirement for NoSQL DBs

I NoSQL si basano su un modello più morbido e flessibile. Il modello BASE si sposa con la flessibilità offerta dai NoSQL e da approcci similari per il management dei dati non strutturati.

- **Basically Available**: garantire la disponibilità dei dati anche in presenza di fallimenti multipli. Questo obiettivo è raggiunto attraverso un approccio fortemente distribuito.
- **Soft state**: abbandona il requisito della consistenza dei modelli ACID quasi del tutto. La consistenza dei dati diventa un problema dello sviluppatore e non deve essere gestita dal database.
- **Eventually Consistent**: l'unico requisito riguardante la consistenza è garantire che in un certo punto nel futuro i dati possano convergere ad uno stato consistente.

Ostacoli del NoSQL

- **Maturità**: i sistemi RDBMS sono in esercizio da più tempo, per molti la maturità di un RDBMS è rassicurate.
- **Supporto**: tutte le aziende che sviluppato e vendono RDBMS forniscono un alto livello di supporto alle aziende. I sistemi NoSQL invece sono spesso progetti open source.
- **Amministrazione**: un obiettivo di design per i database NoSQL è quello di non necessitare di gran supporto amministrativo, ma la realtà è che al giorno d'oggi sono necessarie grosse competenze per la loro installazione e manutenzione.
- **Expertise**: è più semplice trovare un esperto amministrazione RDBMS che non un esperto in NoSQL.

Tipi di DB NoSQL

- La parola **NoSQL**, non indica una ben precisa tipologia di database, bensì è generalmente usata per **raggruppare tutti quei DBMS che non possono essere definiti relazionali**.
- I principali database NoSQL utilizzano paradigmi anche abbastanza differenti l'uno dall'altro, che ovviamente ne condizionano la progettazione, la manutenzione e l'interfacciamento.

- I tipi di DB che analizzeremo sono quattro:

Document data store

Key/Value data store

Graph-based data store

Column-oriented data store

Document data store (1)

- Utilizzano dati non strutturati;
- Supporto a diversi tipi di documento;
- Un documento è identificato da una chiave primaria;
- Schema-less;
- Scalabilità orizzontale.

Document data store (2)

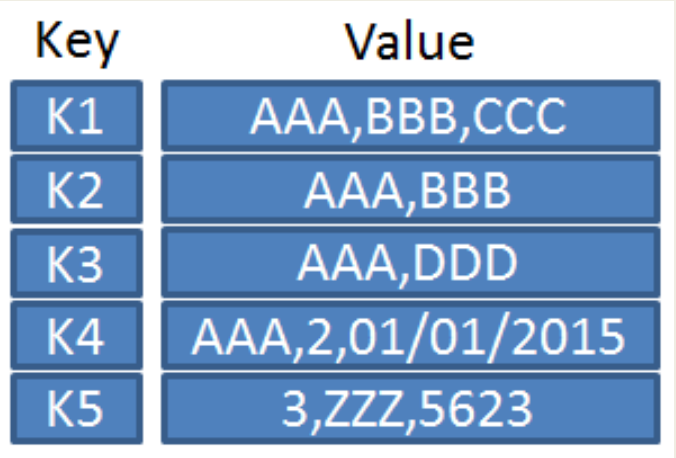
- La rappresentazione dei dati è affidata a strutture simili ad oggetti, dette ***documenti***, ognuno dei quali possiede un certo numero di proprietà che rappresentano le informazioni.
- Per i **documenti** non è obbligatorio avere una struttura rigida fissa.
  Esempio: se i documenti devono rappresentare delle persone, probabilmente avremo in tutti proprietà quali (nome, cognome, data di nascita), ma potremmo decidere di dotare un documento di un numero di telefono, mentre un altro di una email, in base alle informazioni che possediamo.
- Il **documento** può essere visto come il record delle tabelle.
- I **documenti** possono essere messi in relazione tra loro con dei “link”, pertanto i database NoSQL sono particolarmente adatti a rappresentare delle gerarchie.

Key/Value data store (1)

- Utilizza un array associativi (coppie chiave-valore) come modello per lo storage.
- Storage, update e ricerca basato sulle chiavi.
- Tipi di dati primitivi.
- Veloce nel recupero dei dati.
- Capacità di gestire grandi moli di dati.

Key/Value data store (2)

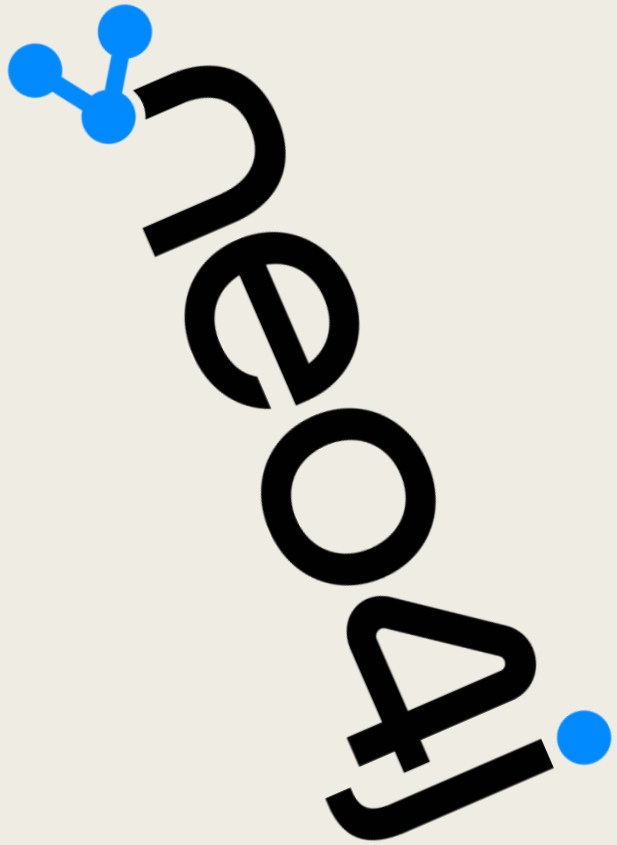
- Costruiti come *dizionari* o *mappe*, in cui vengono inseriti due valori alla volta:
  - *una chiave (identificatore che permette l'estrazione rapida del valore)*
  - *il valore ad essa associato (l'informazione vera e propria)*
- Il loro utilizzo principale è quello di offrire la possibilità di effettuare ricerche rapide di singoli blocchi di informazione.
  Esempio: un sistema di messaggistica, la **chiave** rappresenta un "account" cui verranno indirizzate delle comunicazioni, il **valore** corrispondente è la lista dei messaggi ricevuti.
- Tali database puntano tutto sulla ricerca della velocità: sono spesso ottimizzati per conservare i dati in memoria dinamica e salvarli periodicamente su disco in modo da garantire, sia un certo livello di efficienza della risposta, sia la persistenza dei dati.



Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

Graph-based data store (1)

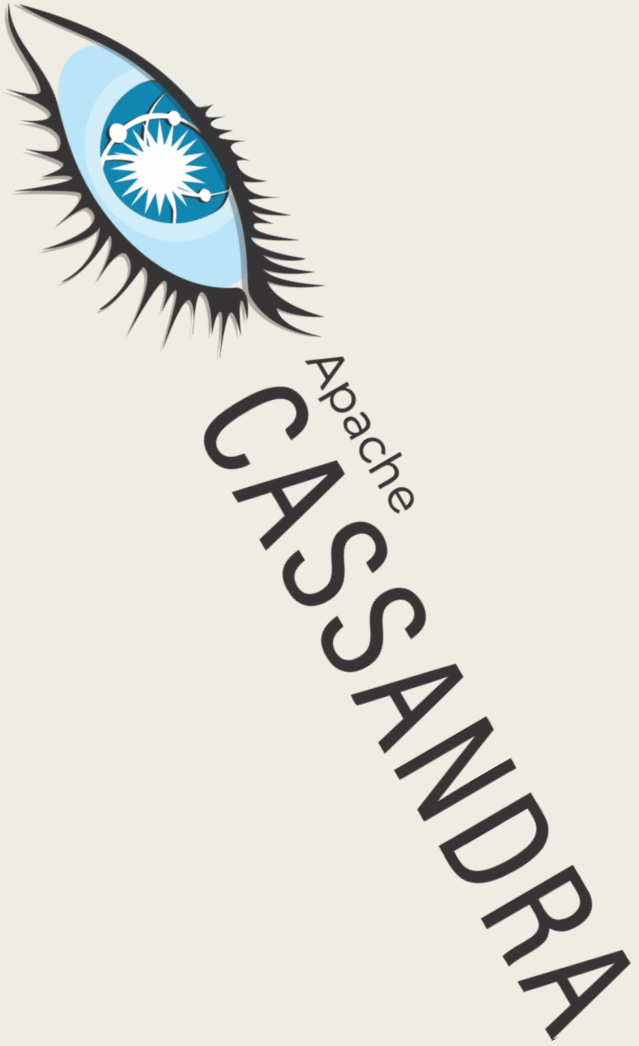
- Utilizza nodi (entità), proprietà (attributi) e archi (relazioni).
- Modello logico semplice e intuitivo.
- Ogni elemento contiene un puntatore agli elementi adiacenti.
- Efficiente per la rappresentazione di reti sociali o dati sparsi.



Graph-based data store (2)

- Le informazioni possono essere custodite sia nei nodi sia negli archi.
- La forza di questa tipologia è tutto l'insieme del valore informativo che si può estrapolare ricostruendo percorsi attraverso il grafo.
- Con un database a grafo si può trovare un percorso tra due città lontane – quindi non collegate direttamente – e, sommando la distanza di ogni tratto, si potrebbe calcolare la distanza totale dalla partenza alla destinazione, più ogni altra grandezza derivata (come tempi, costi, eccetera)
  Esempio: nei Social Network la navigazione del grafo consente di proporre ad un utente nuove potenziali conoscenze recuperando gli "amici di amici".

Column-oriented data store



- I dati sono nelle colonne anziché nelle righe;
- Un gruppo di colonne è chiamato **famiglia** e vi è un'analogia con le tabelle di un database relazionale
- Le colonne possono essere facilmente distribuite
- Scalabile, Performante e Fault-tolerant

Classificazione dei DB

- All’aumentare della complessità nella struttura dati diminuisce la capacità di memorizzazione dei dati stessi.