Università degli Studi di Palermo

dipartimento di ingegneria unipa

Database NoSQL

CORSO DI BIG DATA

a.a. 2021/2022

Prof. Roberto Pirrone

LABORATORIO DI INTERAZIONE UOMO-MACCHINA
CHILAB

Sommario

- RDBMS per la gestione di grandi moli di dati
  - Legge di Amdhal
  - Protocollo 2PC
- CAP Theorem
- Proprietà BASE
- Caratteristiche NoSQL
- Tipologie principali dei database NoSQL
- Vantaggi e svantaggi dei database NoSQL

Immagini e diagrammi da: data-flair.training e www.guru99.com

RDBMS per grandi moli di dati

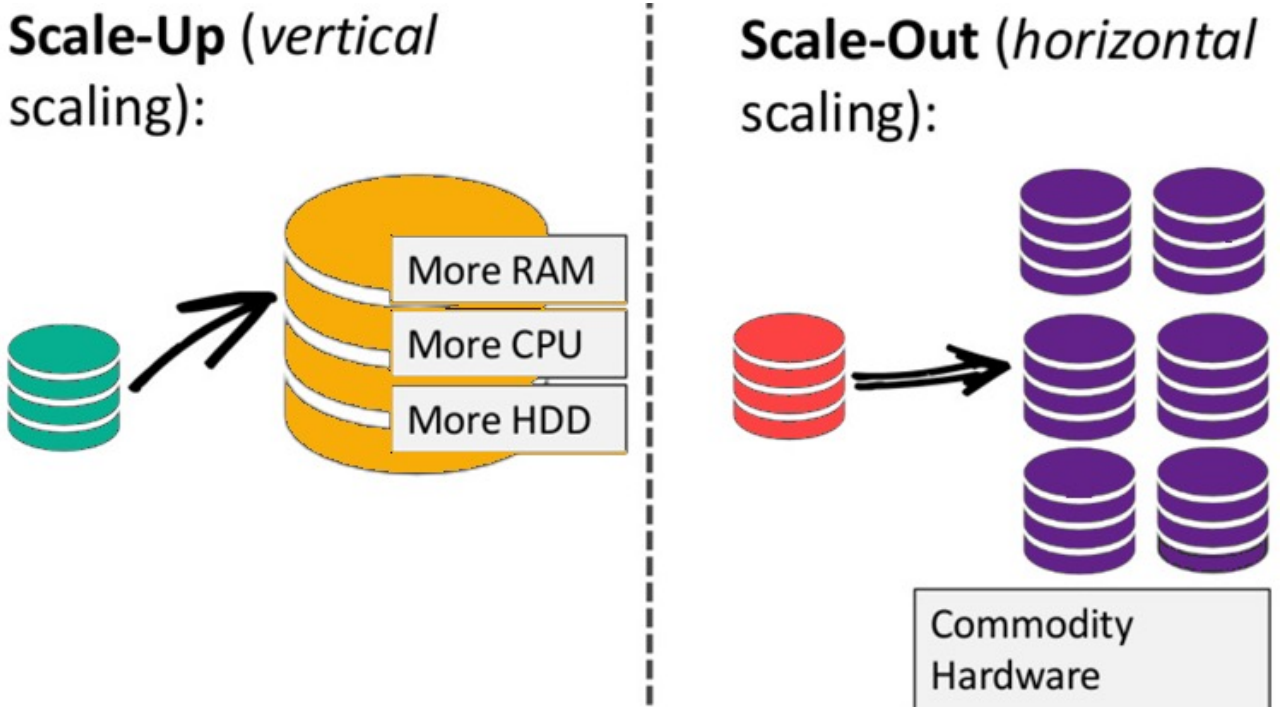
- Lo scenario Big Data si propone immediatamente all'interno delle stesse compagnie che sono *generatrici di big data*

- Google, Amazon, Facebook e gli altri social media *conservano* i dati generati sulle proprie piattaforme per analisi ulteriori e, in genere, a fini di business

- Questo scenario implica la necessità di costruire architetture di RDBMS scalabili per grandi moli di dati

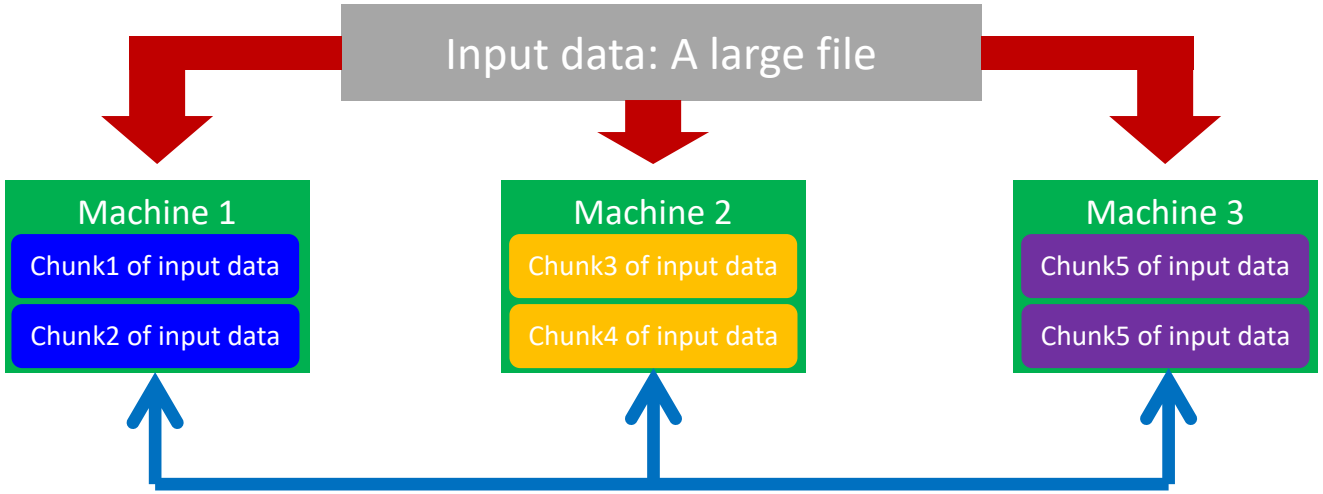
RDBMS per grandi moli di dati

- Un RDBMS può essere scalato
  - Verticalmente
    - Necessita di ingenti investimenti in upgrade hardware
    - Limitato dalla scalabilità della singola macchina/cluster
  - Orizzontalmente
    - Aggiungengdo più macchine
    - Database *sharding* e/o *replication*
    - Overhead di comunicazioni
    - Rapporto read/write limitato



RDBMS per grandi moli di dati

- Database sharding



E.g., Chunks 1, 3 and 5 can be accessed in parallel

- *Si innescano i problemi connessi all'elaborazione parallela*

RDBMS per grandi moli di dati

- Legge di Amdahl
  - Il tempo T_p di esecuzione di un task su p processori in parallelo, è legato al tempo T_1 di esecuzione dello stesso task su singolo processore secondo la formula seguente
    - s è la frazione di programma obbligatoriamente sequenziale

$$\frac{T_1}{T_p} = \frac{T_1}{\left(T_1 \cdot s + T_1 \cdot \frac{1-s}{p}\right)} = \frac{1}{s + \frac{1-s}{p}}$$

80% parallelizzabile
4 processori

Solo 2.5 volte!!!

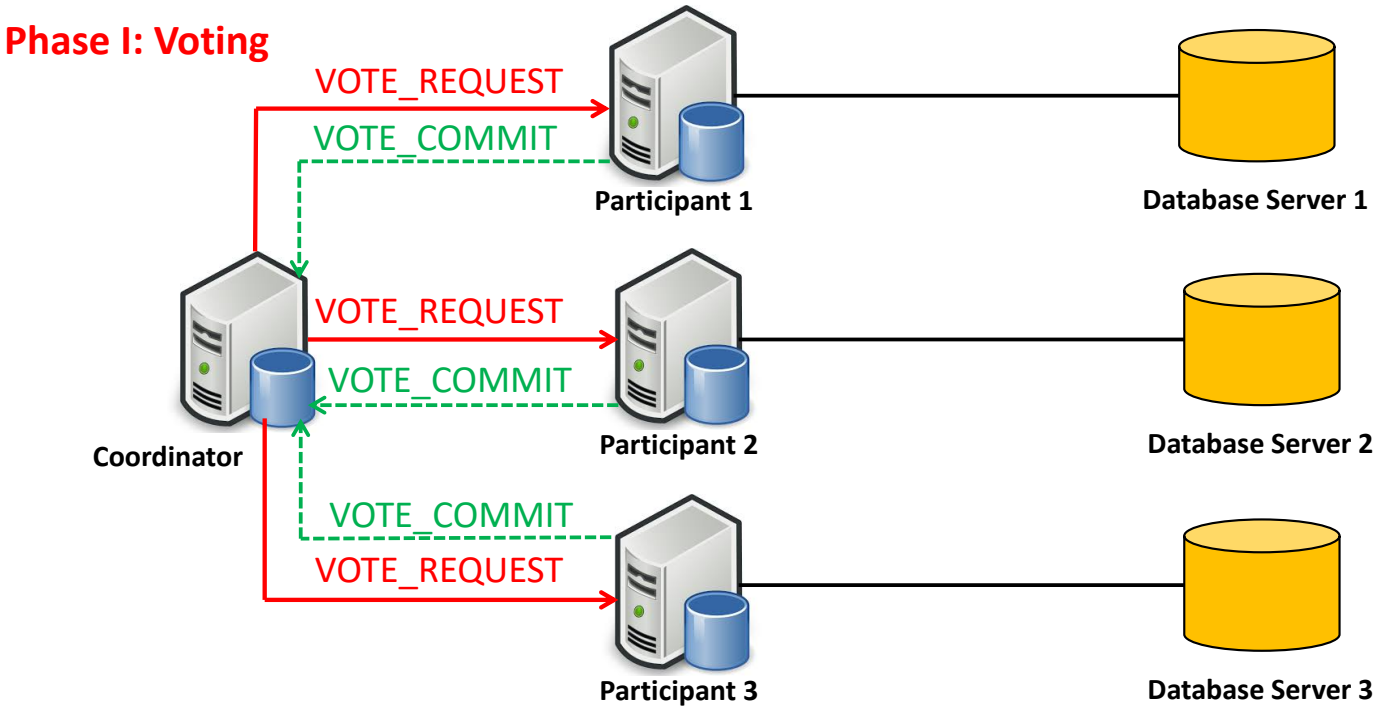
  - Ulteriore riduzione per effetto del overhead di comunicazione e del carico sbilanciato

RDBMS per grandi moli di dati

- Una soluzione migliore dello sharding è la *data replication*
  - Maggiore performance perché si evitano i colli di bottiglia delle architetture parallele
  - Non ci sono più Single Point of Failure (SPOF)
  - Scalabilità
- Il problema in questo caso è il *mantenimento della consistenza dei dati*

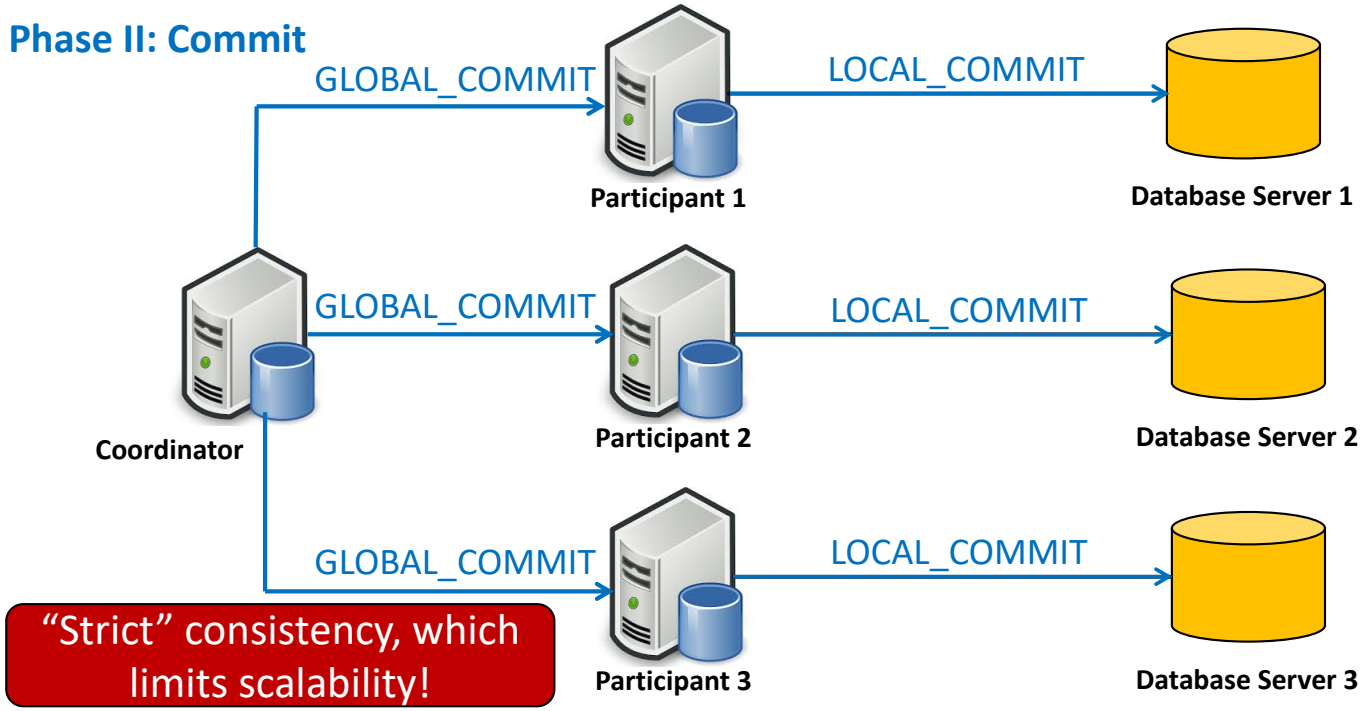
RDBMS per grandi moli di dati

- Protocollo 2 Phase Commit (2PC)



RDBMS per grandi moli di dati

- Protocollo 2 Phase Commit (2PC)



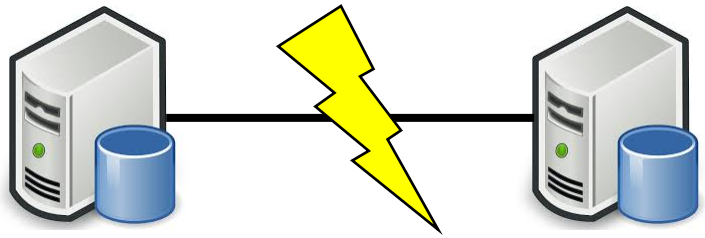
CAP Theorem

- Le tre principali caratteristiche che si richiedono a un database distribuito sono
  - *C*onsistenza (stretta) – ogni nodo vede esattamente gli stessi dati in ogni istanza del database
  - *A*vailability (disponibilità) – il sistema resta operativo anche se alcuni nodi hanno un crash o vengono spenti per manutenzione
  - *P*artition tolerance (tolleranza alle partizioni) – il sistema resta operativo anche in presenza di interruzioni nella rete
- *Un qualunque database distribuito con dati condivisi può garantire al più due delle proprietà CAP*

CAP Theorem

- Si pensi a due nodi sui due lati di una interruzione della rete

- *AP non consente C* proprio per effetto immediato dell'interruzione
- *CP non consente A* poiché implica necessariamente che uno dei due nodi debba andare offline per garantire *C* e *P*
- *CA non consente P* perché implica esattamente che nessun nodo sia irraggiungibile



Proprietà BASE

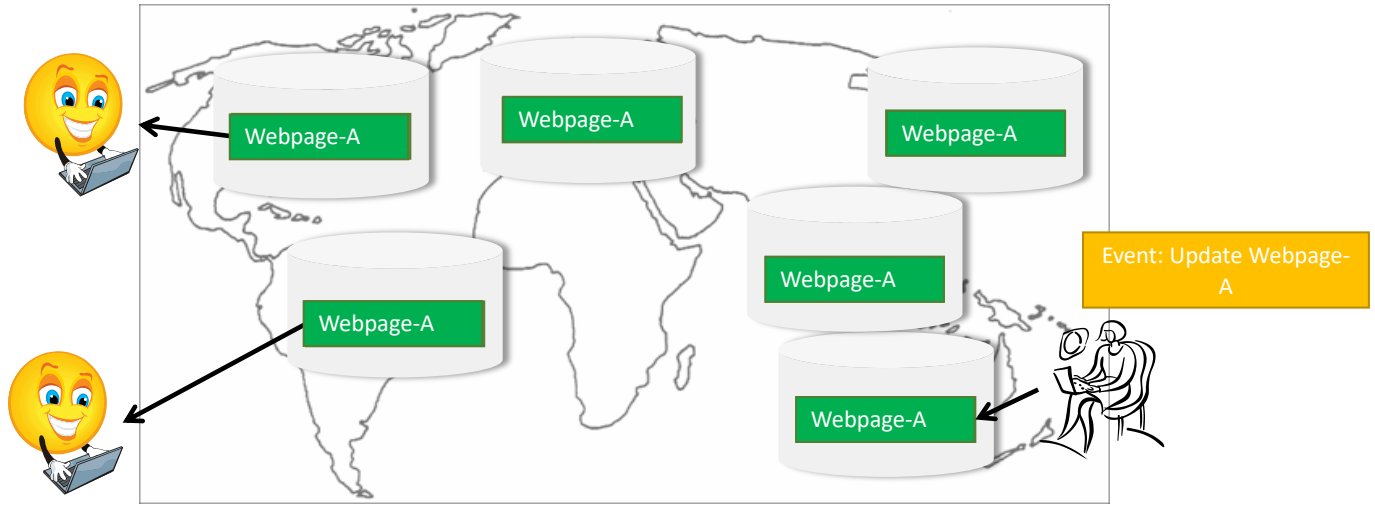
- La soluzione genericamente adottata per un database distribuito è la *consistenza lasca*
  - Le grandi aziende che utilizzano questi sistemi necessitano di garantire *AP* perché si tratta di database che devono servire utenti/consumatori in tutto il giorno con disponibilità 24/7
  - Un sistema a consistenza lasca è in genere molto efficiente e semplice da implementare

Proprietà BASE

- In particolare si cerca di garantire le seguenti proprietà
  - ***B****asically* ***A****vailable*: il sistema è sempre disponibile
  - ***S****oft-sate*: lo stato del sistema *può* cambiare nel tempo
  - ***E****ventually consistent*: il sistema *tende* a divenire consistente
    - Tutte le repliche si allineano gradualmente, in assenza di modifiche

Proprietà BASE

- Consistenza «Read-Your-Own-Writes»
  - Quando deve eseguire una transazione, il nodo master genera un *commit token* che invia alle repliche
  - Ogni replica può stabilire se è allineata o meno
  - Il client può sapere se una replica è allineata o meno su un certo update attraverso i token

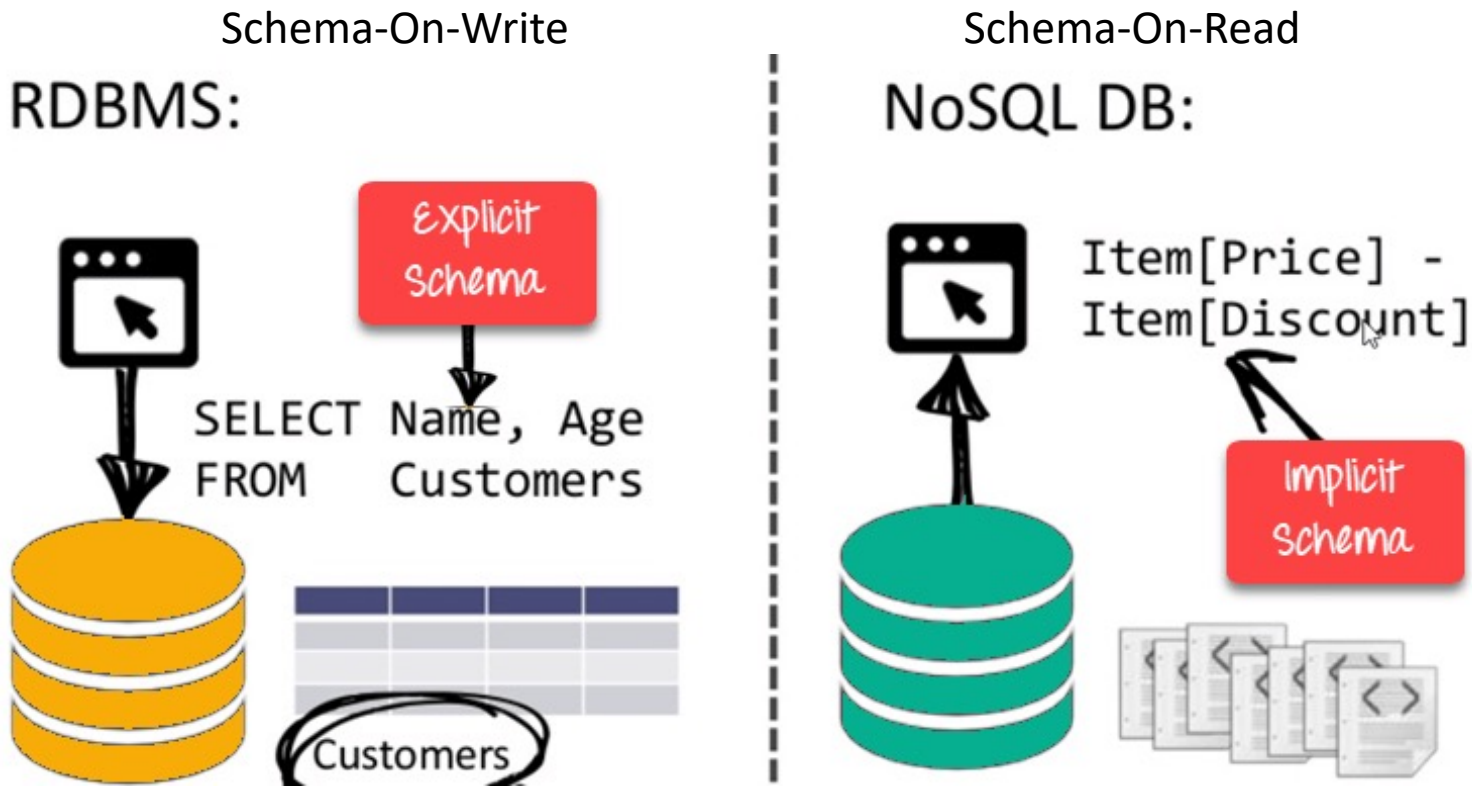


Caratteristiche dei database NoSQL

- NoSQL: Not only SQL
  - Un database che non usa lo schema relazionale
  - Non usa tabelle con record di formato fissato
  - Utilizza strutture dati auto-consistenti o addirittura blob
  - Non necessita di ORM o di normalizzazione dei dati
  - Non utilizza linguaggi di query
  - Non utilizza vincoli di integrità referenziale e non ci sono transazioni ACID

Caratteristiche dei database NoSQL

- NoSQL: Not only SQL
  - Non ha uno schema ovvero ha schemi rilassati
  - Non richiede la definizione a priori dello schema
  - Offre una serie di strutture dati per gestire i dati

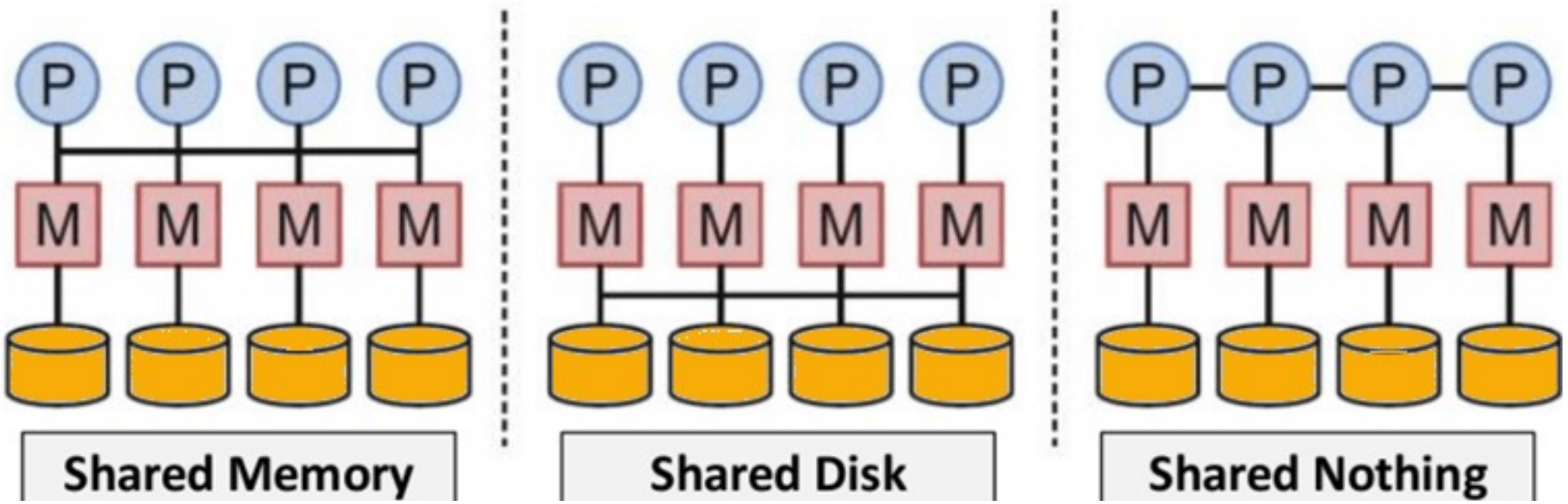


Caratteristiche dei database NoSQL

- NoSQL: Not only SQL
  - Offre semplici API per la manipolazione anche a basso livello
  - In genere offrono interfaccia HTTP REST e protocolli di testo per la comunicazione, ad es. JSON
  - Interfaccia Web
  - Ambiente distribuito che utilizza eventual consistency
  - Architettura «Shared Nothing»

Caratteristiche dei database NoSQL

- NoSQL: Not only SQL

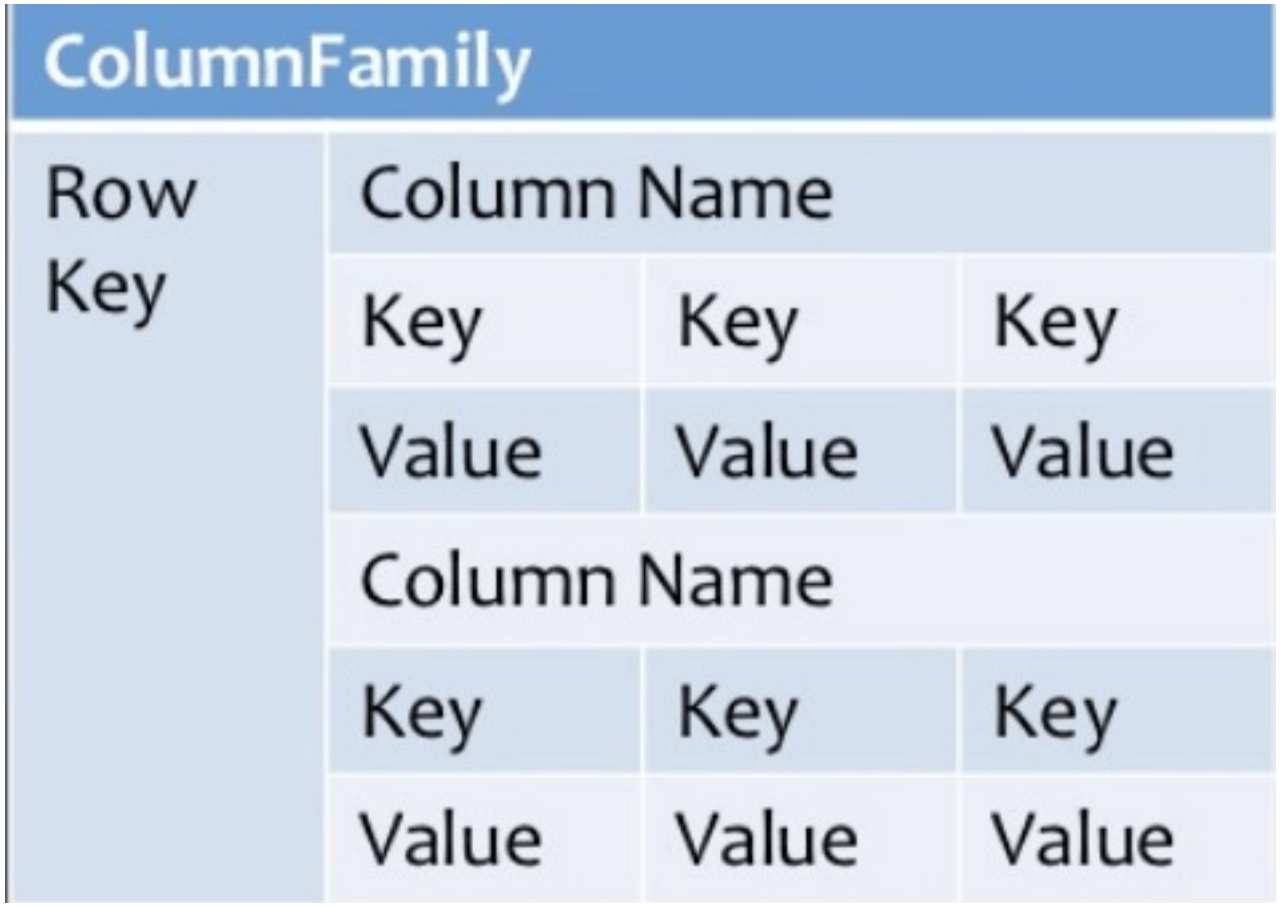


Tipologie principali dei database NoSQL

- Coppie chiave-valore
  - I dati sono conservati come coppie chiave-valore
    - Le chiavi sono univoche
    - I valori possono essere eterogenei: stringhe, JSON, BLOB (Binary Large OBjects)
  - Le chiavi sono implicitamente lo schema: non è richiesto schema a priori
    - Es. gestione dei carrelli elettronici
  - Amazon Dynamo

Tipologie principali dei database NoSQL

- Database a colonne
  - Molto efficienti per query cumulative (conteggio, somma, media …) lungo le singole colonne
  - Basati su Google BigTable
  - Cassandra, HBase



Tipologie principali dei database NoSQL

- Database a colonne
  - I dati sono indicizzati direttamente sulle colonne
  - Possono essere raggruppati per «righe» o più in generale in «gruppi di colonne» che non debbono coincidere necessariamente con un intero record

ColumnFamily			
Row Key	Column Name		
	Key	Key	Key
	Value	Value	Value
	Column Name		
	Key	Key	Key
	Value	Value	Value

Tipologie principali dei database NoSQL

- Database documentali
  - Ottimo per CMS, piattaforme di blogging, analytics, e-commerce …
  - MongoDB

Col1	Col2	Col3	Col4
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data

Document 1

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Document 2

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Document 3

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Tipologie principali dei database NoSQL

- Database documentali
  - Ogni record è un «documento» rappresentato in un formato testo strutturato come JSON e/o XML

Col1	Col2	Col3	Col4
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data	Data	Data

Document 1

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Document 2

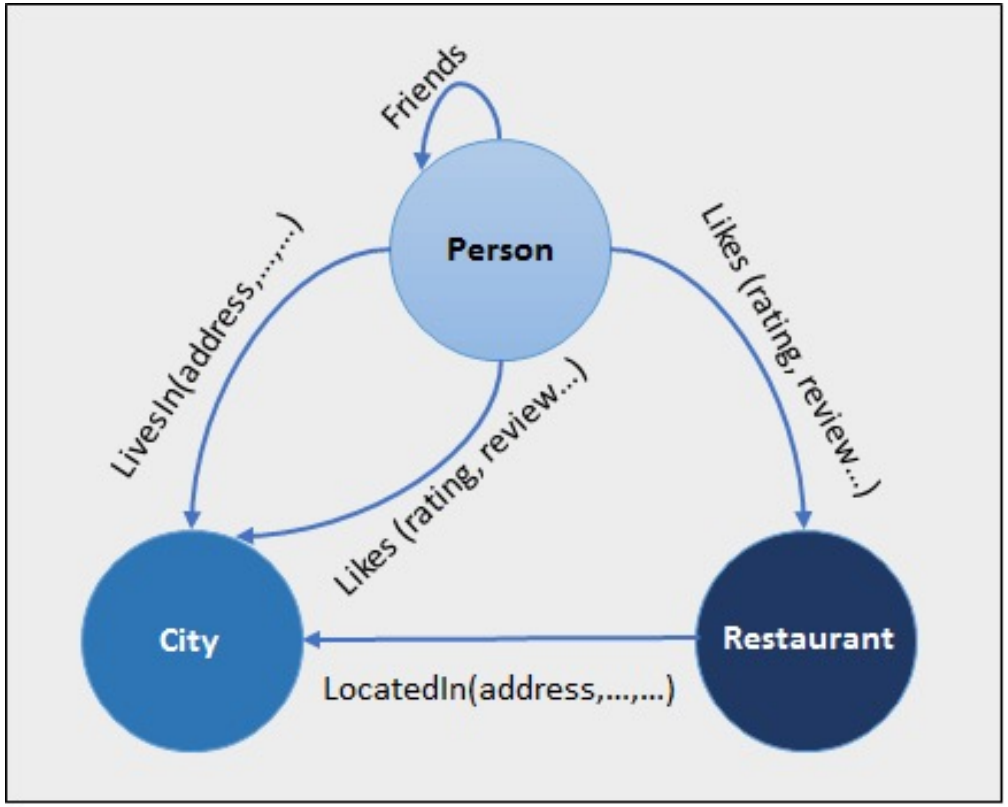
```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Document 3

```
{
  "prop1": data,
  "prop2": data,
  "prop3": data,
  "prop4": data
}
```

Tipologie principali dei database NoSQL

- Database a grafo
  - I dati sono rappresentati come nodi e archi etichettati di un grafo
  - Struttura «multirelazionale»
  - Le query si mappano in operazioni tra grafi (ad es. shortest path)
  - Neo4j



Vantaggi e Svantaggi dei database NoSQL

- Vantaggi
  - Possono essere usati sia per scopi di analytics sia come data lake
  - Orientati ai Big Data
  - Non c’è più un SPOF
  - Performance e scalabilità orizzontale
  - Gestione di dati strutturati, semi-strutturati o non strutturati

Vantaggi e Svantaggi dei database NoSQL

- Vantaggi
  - Programmazione semplice in OOP
  - Non necessitano di server dedicati ad alte performance
  - Pensati per i database distribuiti
  - Schemi sui dati non predefiniti che possono essere alterati senza interruzione del servizio

Vantaggi e Svantaggi dei database NoSQL

- Svantaggi
  - Non c’è standardizzazione
  - Limitate capacità di query
  - Le tecnologie RDBMS sono molto più mature
  - Non garantiscono consistenza
  - I sistemi open source non sono molto apprezzati/popolari nelle organizzazioni pubbliche o private