



UNIVERSITÀ  
DEGLI STUDI  
DI PALERMO



# Cassandra

CORSO DI BIG DATA  
a.a. 2019/2020

Prof. Roberto Pirrone

# Sommario

- Caratteristiche principali
- Cassandra vs MongoDB
- Architettura
- Modello dei dati

# Caratteristiche principali

- Column-oriented
  - Le colonne contengono direttamente i dati e non i metadati (i nomi dei campi)
- Architettura distribuita altamente scalabile
  - Basata sui concetti di nodo, data center e cluster, può essere facilmente scalata in entrambe le direzioni
  - Comunicazione master-slave o peer-to-peer tra i database
- Fault tolerant
  - Connessione ad anello tra i nodi, data center multipli, gestione del carico per singolo nodo/cluster

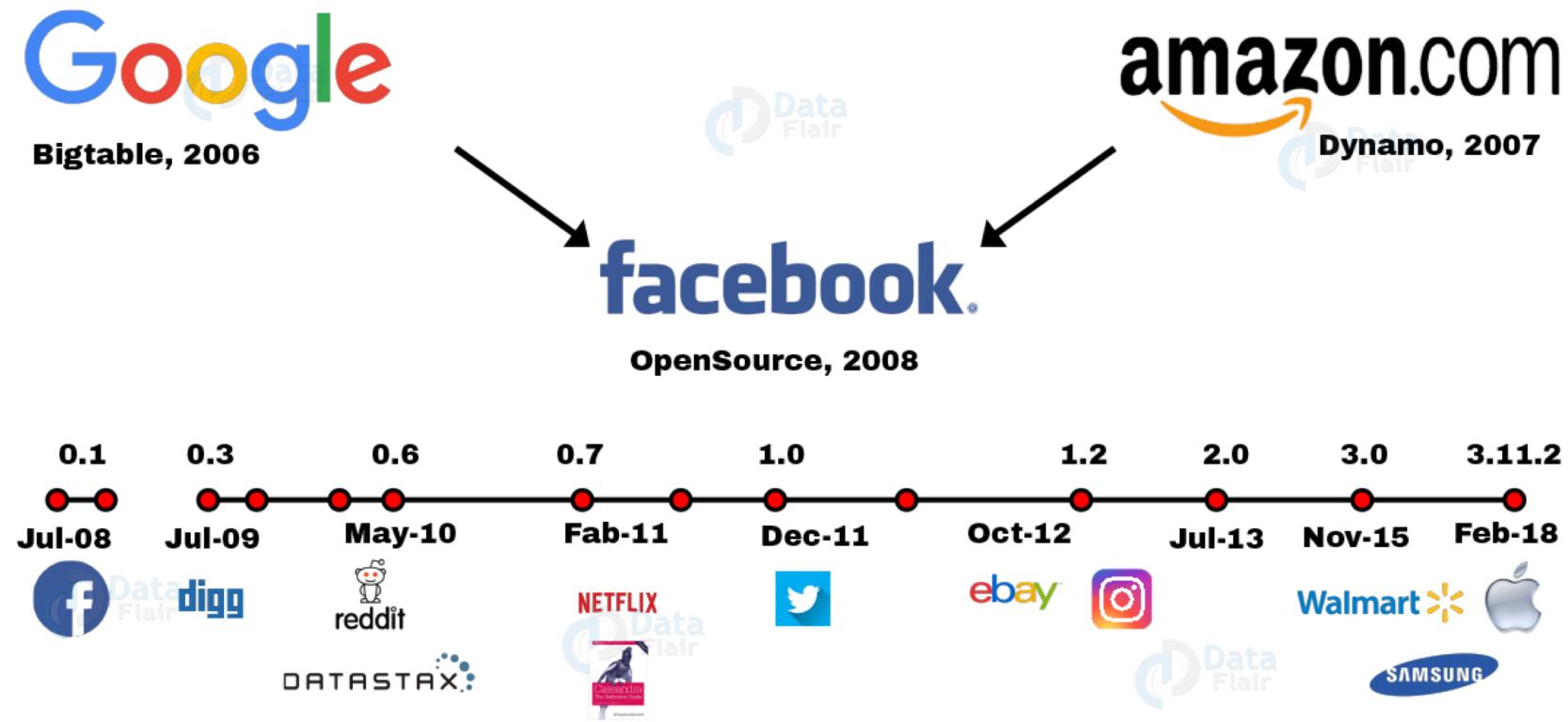
# Caratteristiche principali

- Elevate prestazioni
  - È disegnato per la gestione rutinaria di elevatissime moli di dati
- Consistenza configurabile
  - Gestisce la consistenza stretta, quella «eventual», ma anche un mix delle due
- Schema lasco
  - Anche se sembra di interagire con un RDBMS, l'aggregato principale delle colonne è la «column family» cioè righe ciascuna delle quali può contenere gruppi di colonne differenti

# Caratteristiche principali

- Applicazioni tipiche
  - Storage di dati molto eterogenei e in grande quantità (Digg)
  - Sviluppo del back-end delle applicazioni (Netflix, Google App Engine)
  - Analisi di serie temporali (WebEx di Cisco per lo storage dei feed utente)
  - Strumenti di monitoring (CERN)
  - Strumenti di Analytics (BlackRock)

# Caratteristiche principali

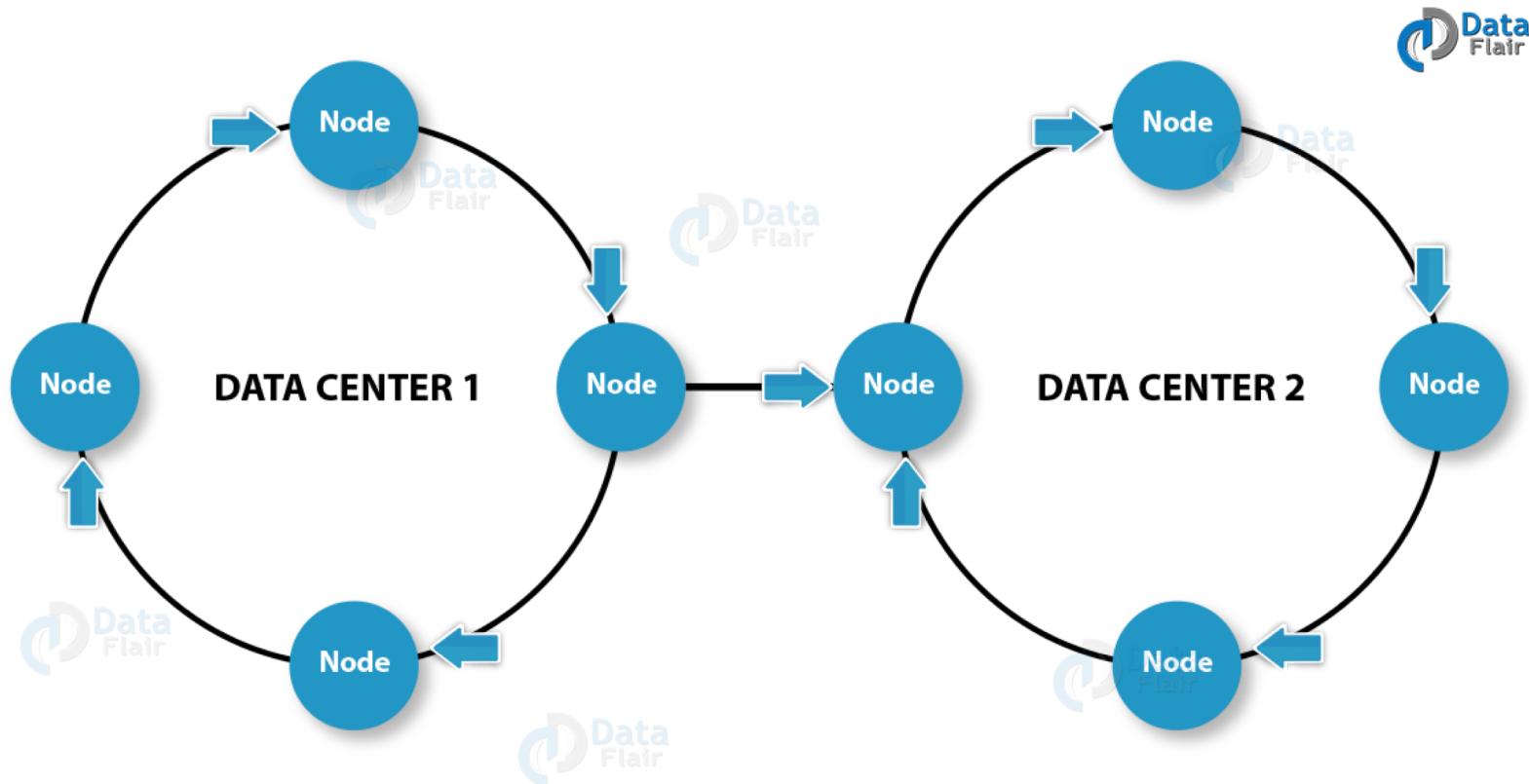


# Cassandra vs MongoDB

	Cassandra	MongoDB
Modello dei dati	Basato su tabelle con righe e colonne	Modello ricco ed espressivo object-oriented
Nodo master	Multipli: se uno va down, un altro prende il suo posto	Uno solo: processo di elezione necessario in caso di failure
Indici secondari	Supporto per indici su singole colonne che gestiscono solo l'uguaglianza dei valori (sono semplici cursori)	Ogni proprietà o sub-documento è indicizzabile
Scalabilità	Elevata	Limitata
Linguaggio di query	CQL (simile a SQL)	Nessuno: la query è un documento JSON
Aggregazione	Non presente	Aggregation framework che include anche MapReduce
Schema	Tipizzazione statica: l'utente deve definire il tipo di una colonna	Assolutamente schema free

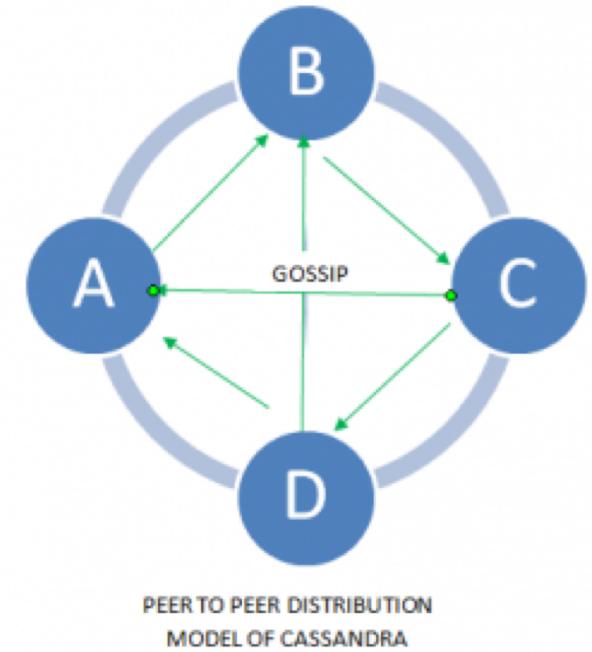
# Architettura

- Nodi organizzati circolarmente in *cluster*
- I cluster possono essere suddivisi fisicamente su più *data center* dislocati in luoghi diversi
- Un nodo è un singolo server in un *rack*

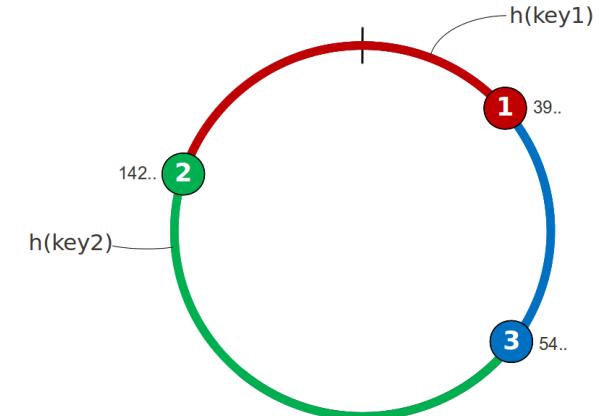


# Architettura

- I nodi comunicano tra loro in modalità peer-to-peer usando il Gossip protocol per stabilire l'appartenenza al cluster
  - Comunicano randomicamente a coppie per *diffondere* le informazioni
- L'organizzazione circolare al suo interno suddivide gli intervalli di valori delle chiavi

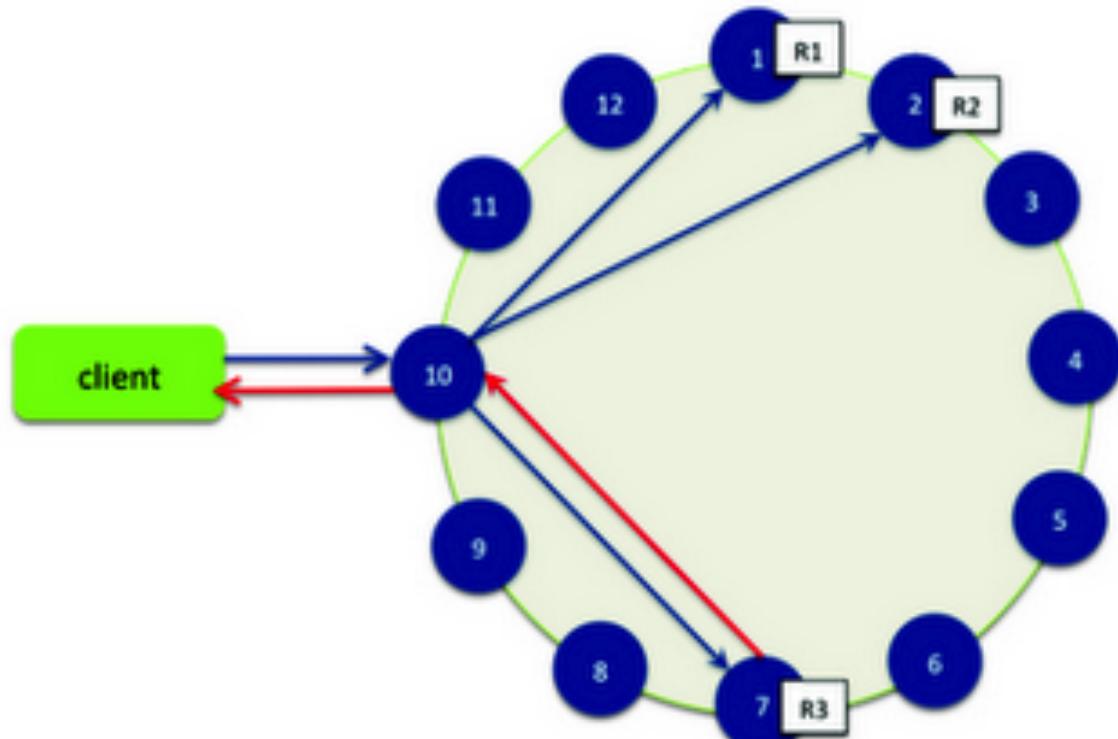


Partitioning of Keys in the Cluster



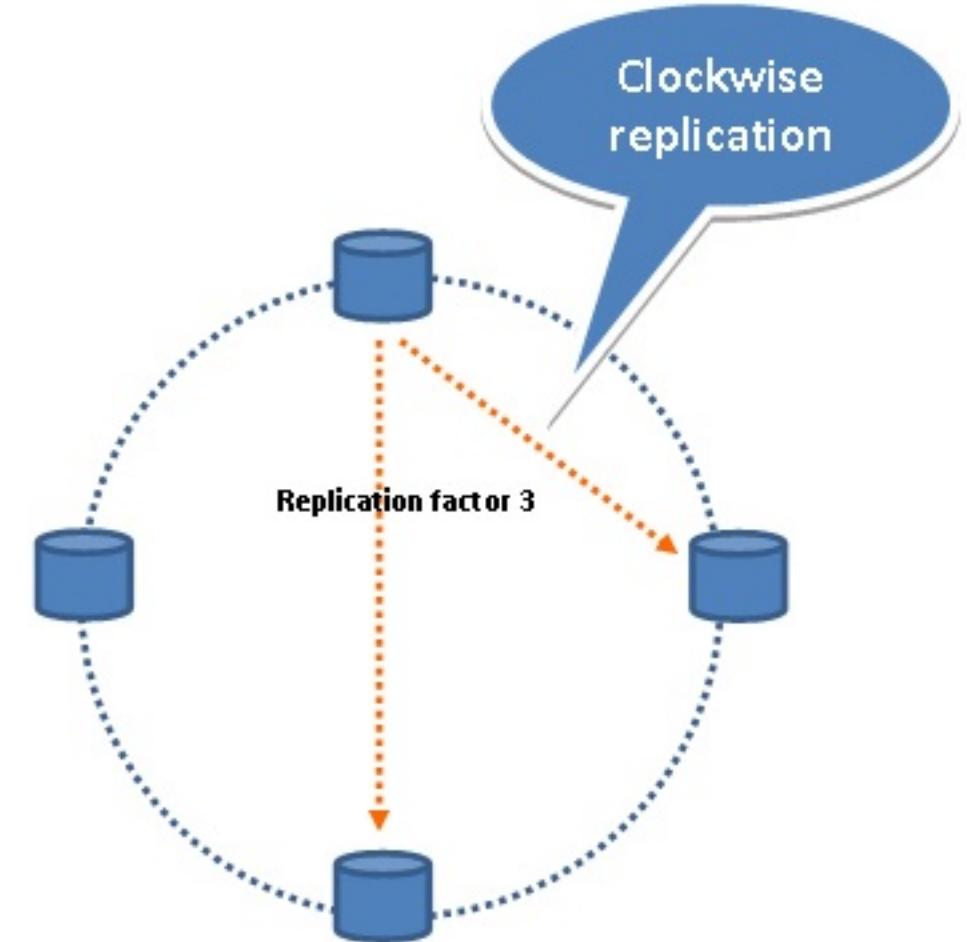
# Architettura

- Una read/write dal client avviene verso un nodo qualunque del cluster che diviene il *coordinatore*
  - Individua i nodi effettivamente responsabili dell'intervallo di valori di chiavi coinvolto
  - Coordina la replicazione



# Architettura

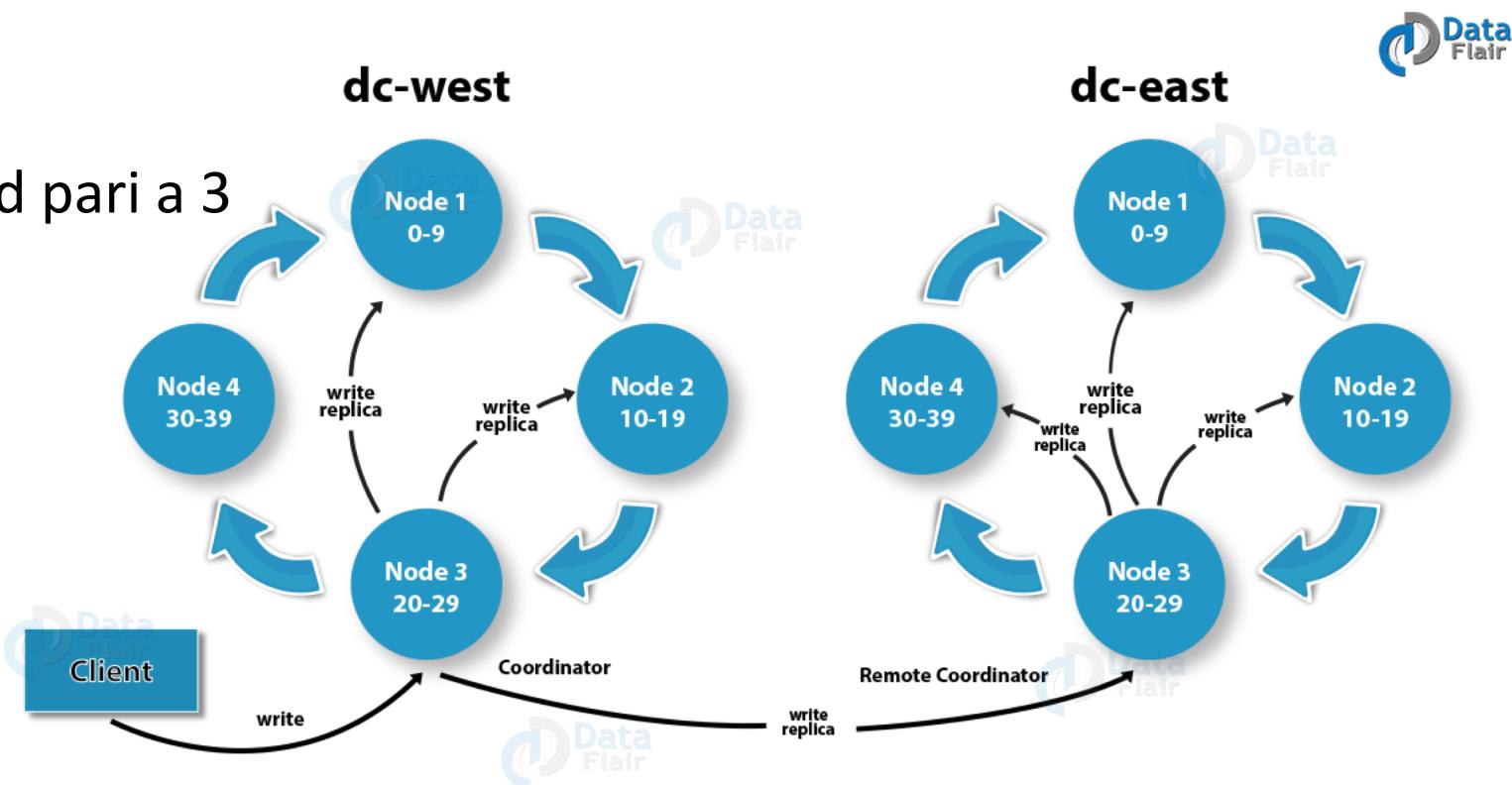
- Strategie di replicazione
  - Replication factor standard pari a 3
  - Simple Strategy
  - Network Topology



# Architettura

- Strategie di replicazione

- Replication factor standard pari a 3
- Simple Strategy
- Network Topology



# Architettura



**Commit log**

**Mem Table**

**Row Cache**

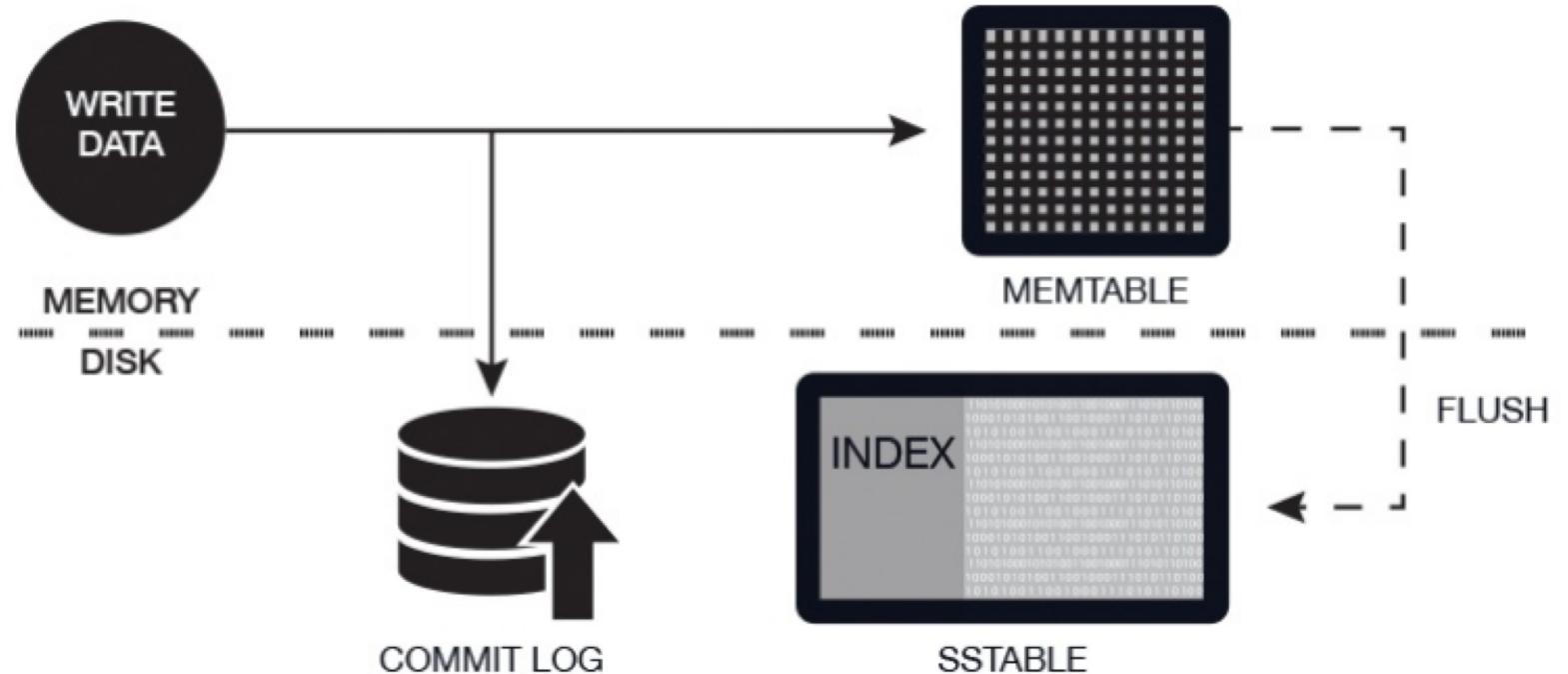
**Bloom Filter**

**SST Table**

**Key Cache**

# Architettura

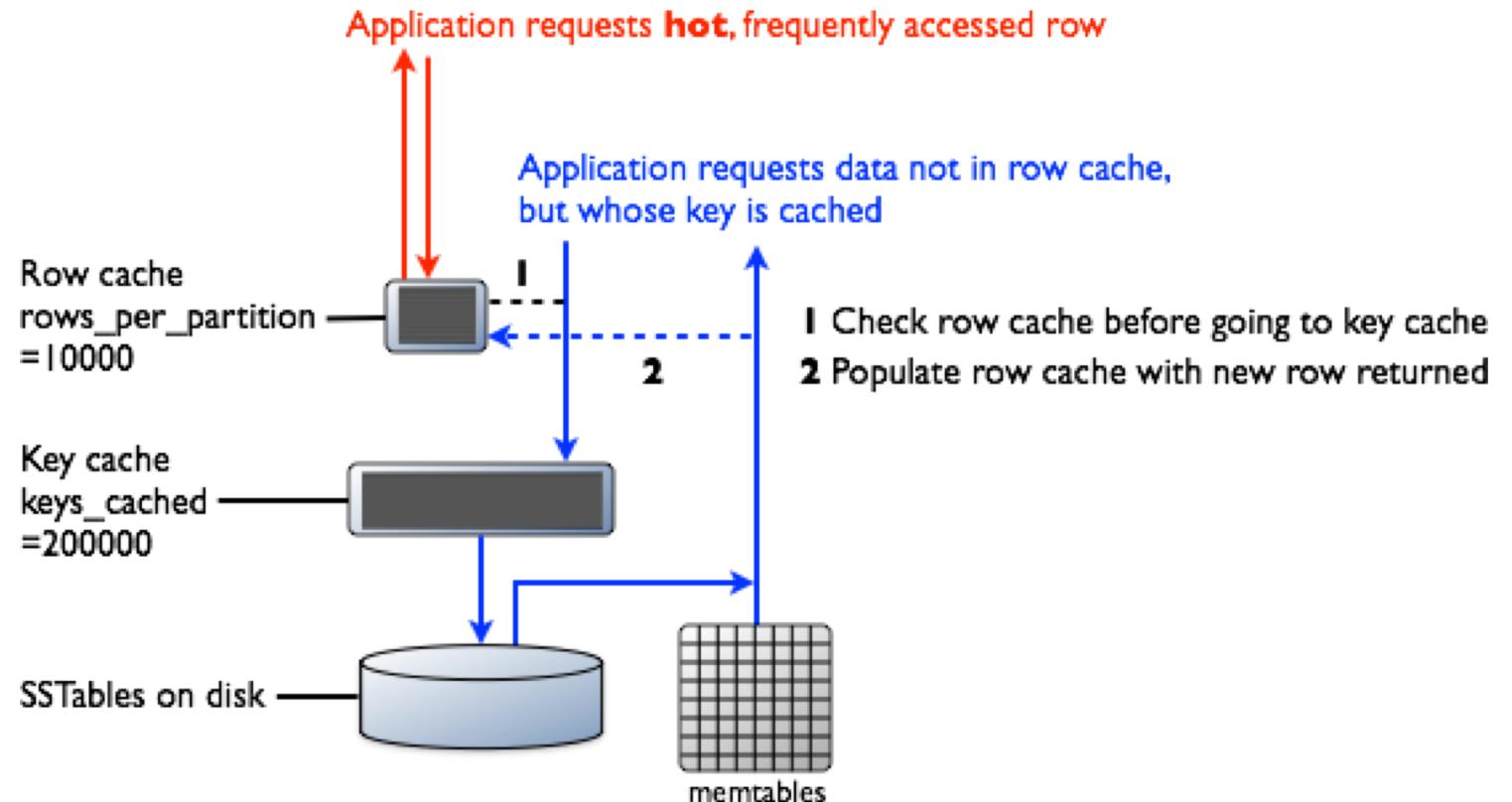
- Flusso di scrittura
  - Il write viene annotato nel Commit Log
  - La scrittura avviene fisicamente nella Mem Table
  - La memtable effettua poi il flushing nella SSTable



# Architettura

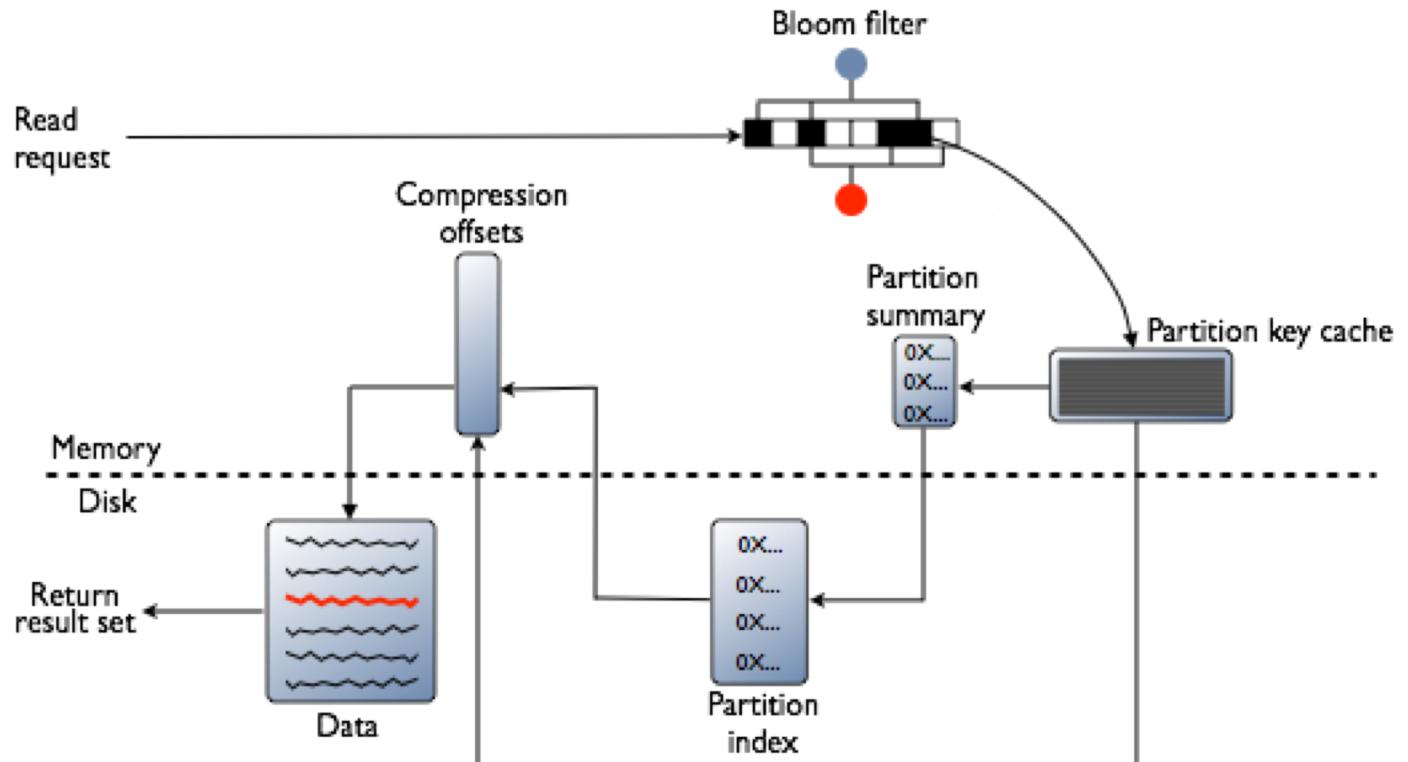
- Flusso di lettura

- Se abilitata, viene consultata la row cache
- Altrimenti la partition cache
- I dati recuperati dalle SSTables vengono spostati sulle relative MemTables



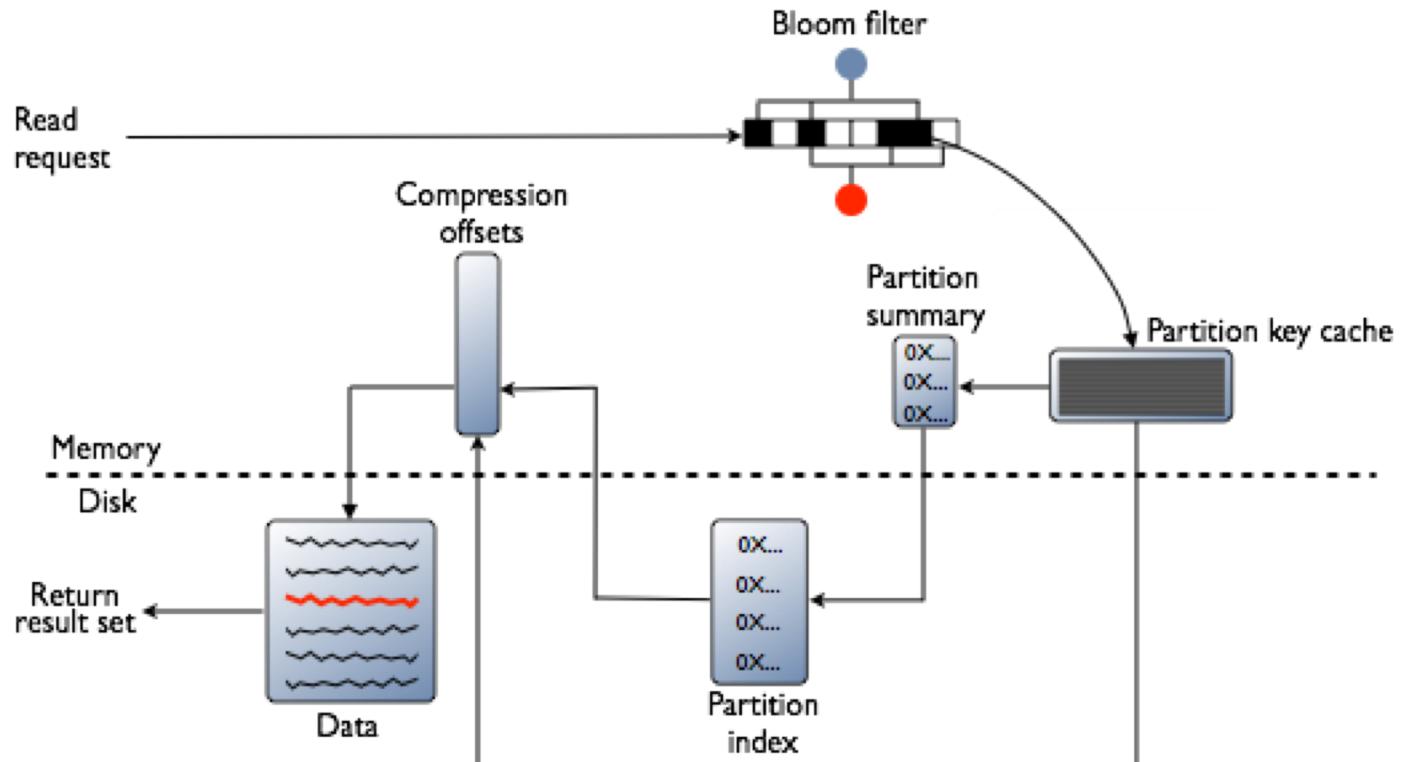
# Architettura

- Flusso di lettura (dettagliato) in caso di mancanza/fallimento della row cache
  - Il Bloom filter verifica se i dati sono direttamente nella MemTable
  - La Partition key cache, in caso di successo, va a indirizzare la compression offset map per individuare i dati su disco



# Architettura

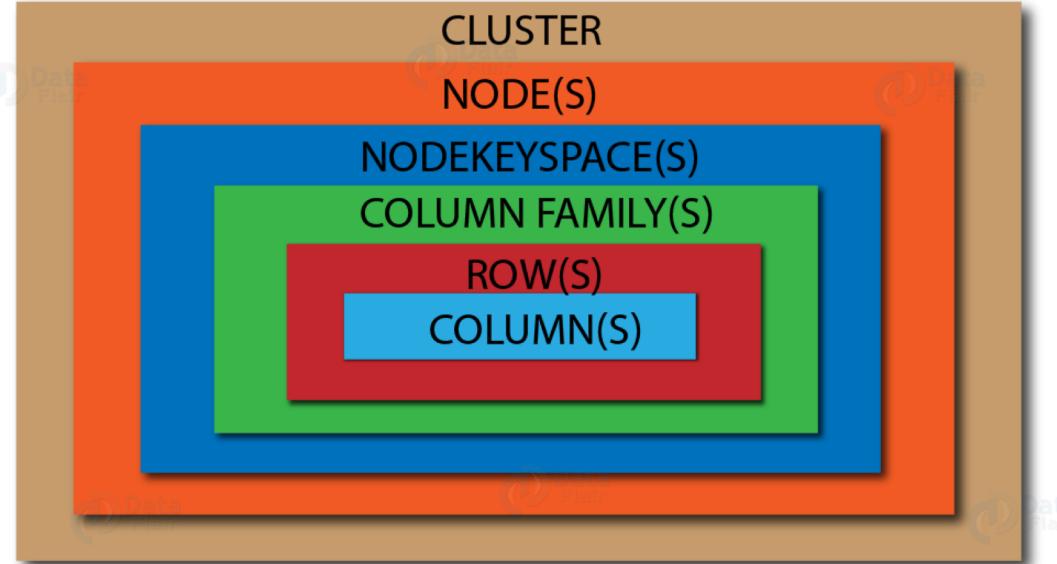
- Flusso di lettura (dettagliato) in caso di mancanza/fallimento della row cache
  - Se la Partition key cache fallisce indica il Partition summary che accede al Partition index
  - Si ottiene il riferimento per la Compression offset map e l'effettiva lettura



# Modello dei dati

- Key space

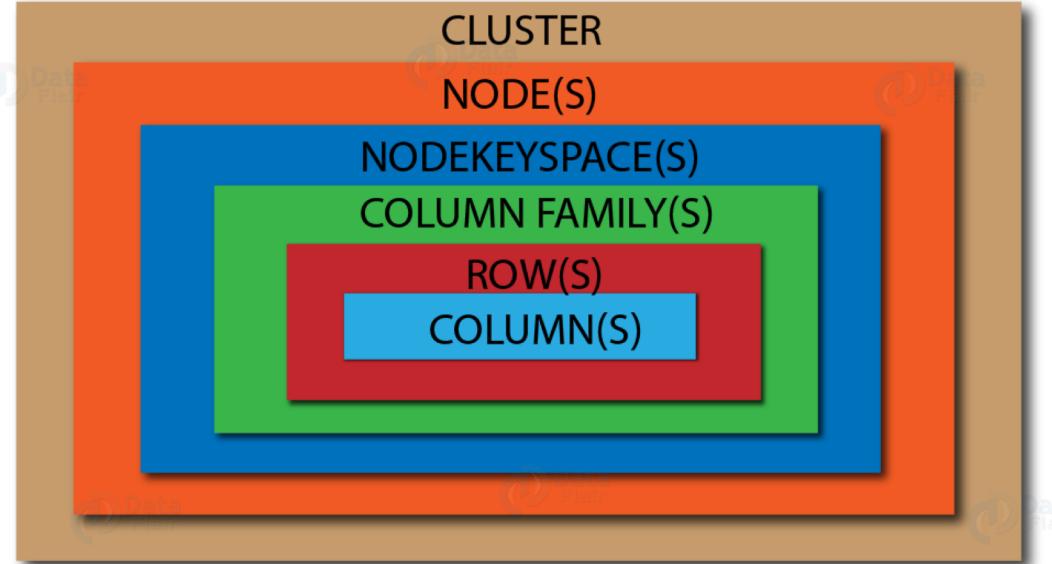
- Può essercene più di uno per nodo
- E' equivalente al concetto di database
- E' la struttura di storage più esterna



# Modello dei dati

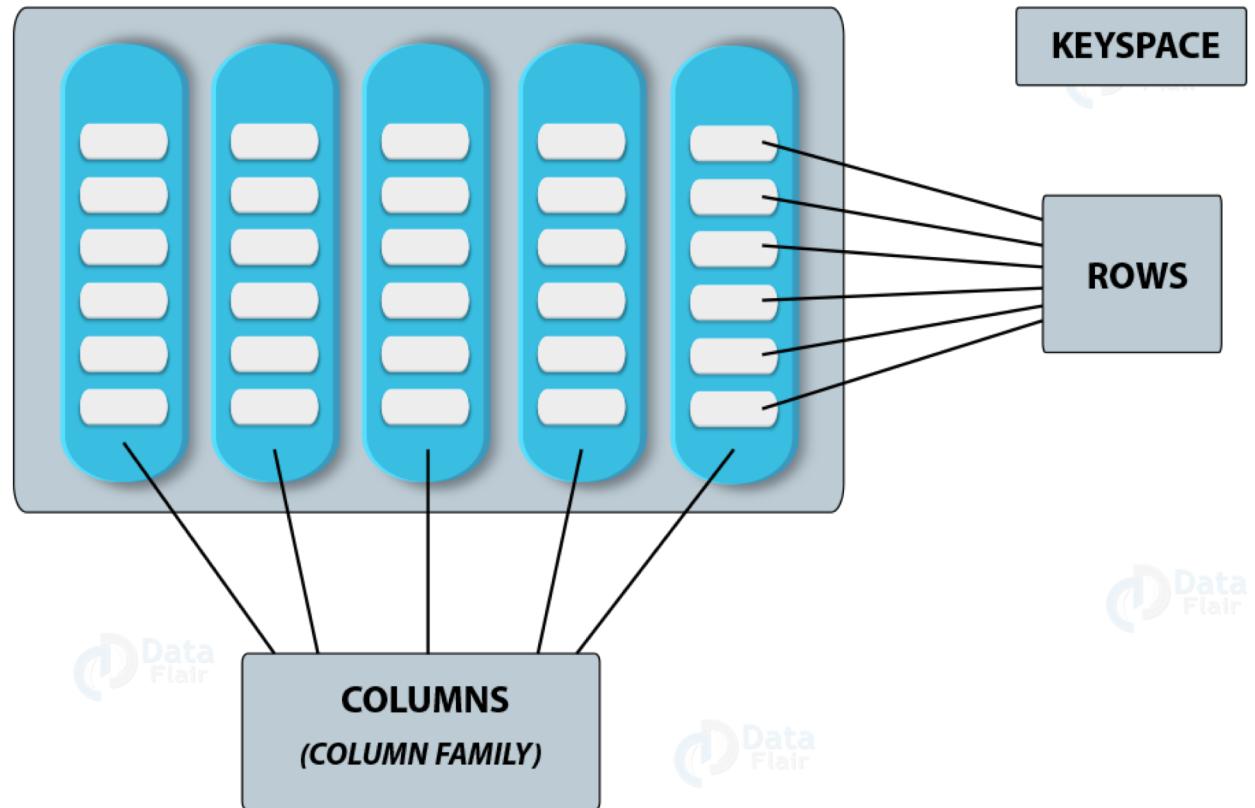
- Column family

- La suddivisione del key space
- Possono essere assimilate a tavelle
  - Si accedono come tavelle in CQL



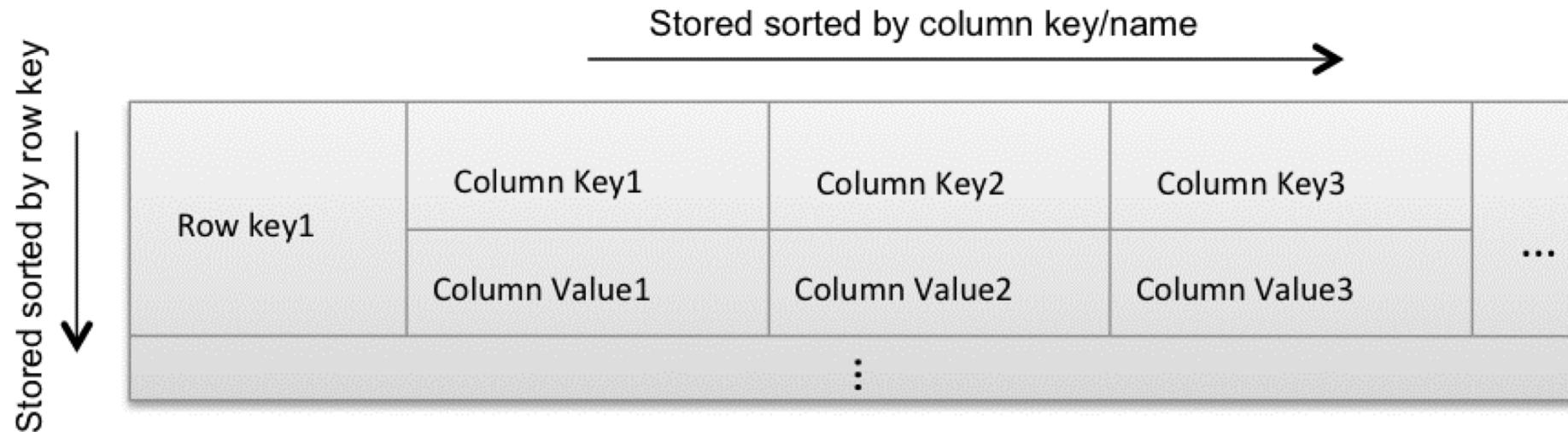
# Modello dei dati

- Rapporto tra Key space, righe e colonne



# Modello dei dati

- Righe e colonne



# Modello dei dati

Relational Model	Cassandra Model
Database	Keyspace
Table	Column Family (CF)
Primary key	Row key
Column name	Column name/key
Column value	Column value