Day 8 Manga,Cassandra,Influx,Neo4j

Hard disk

Seek time = Seek + Rotational latency + Transfer time

Exp: 2GHz → 2 cycles
Exp: 2GHz → 1 cycle
Exp: 2GHz → 512 KB, 1ms

For analytics, we take entire block to run a query. Even if run one sec, take entire block. Better the processing. This is the logic behind vertical partitioning.

Select ... where designation = 'Manager'

→ Row based
→ Row to row
→ Row to row

So to prevent this, there is COLUMNAR DBS

CASSANDRA →

Column is retrieved → faster in terms of processing

Per column can contain same data type

Cols	Stage	Gender
Ehm	Mdm	M
Ehm	Mr	M
Ehm	Ms	M
Bix	Sg	M
Bix	Mr	M

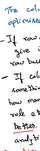
The columns can be further optimized called compression

- If row wise queries are more, it gives info of the empid, then row wise DB is better
- If column wise queries are more, something more analytical like how many she in a city, particular role etc. then columnar DB is better. Cassandra is used in analytical intensive DB

Searching for data:

Indexing → This index table is stored in the disk/disk
So we just need to seek a second index table → access block via pointers

Index table



Indexing → Reads ↑
writes ↓ (extra work offloaded)
This is trade off

For read intensive data, frequent updates then Indexing is good.

Hitting another index table called Secondary Indexing

For more index you add. Read ↑
writes ↓

From another "view point"

This is B-tree. Thats why B-tree is used for indexing

Mongo DB

→ Uses B-tree for indexing.
→ Read heavy operations (also true for B-tree)

Data model → Non-normal → normally stored as binary. Binary

Schema flexibility → Flexible (as it is stored as binary)

(Mongo → Document oriented → Read schema flexible schema)

C/A

Availability → Horizontal scaling
Sharding

Collections → tables
documents → rows

Cloud managed's can

LSM Tree

Log Structured Merge

Sequential write operations

① Mem table (write data in memtable, as writes is sequential)

② SSTable (the memtable SSTable is stored in SSTable in HDD)

So this is a two step work but it is optimized for writes as first writes happen in memory to memtable

→ Cassandra, Influx DB uses LSM
↓
Columnar DB
↓
Time series DB

→ Both are optimized for write heavy operations

Influx DB

Time series Data Base

Data Model → Time → Influx Key (measurement)
Long series (Time series) DB
Influx → Time series Database. Just data is on the time

Schema Flexibility → Flexible

C/A → Availability

R/W performance → LSM tree
Rich write throughput
influx data can be done as data sets

Other Time series DB:
Influx DB
Timescale
Prometheus
Graphite
Open TSDB
Riak DB

Why → people in what all situations which DB we can use?

Neo4j → - Performance intensive
- Data intensive

Amazon → - Transaction based and Non Transaction data based

Youtube :

Swiggy :

Graph DB

Neo4j
Amazon Neptune
Dgraph
OrientDB

Stored as graph

Nodes store data
Edges store relationship
Properties

Consists neo4j :

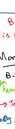
Usecase: Shortest distance, Personalization, relationship

Neo4j is ACID Transaction
Consistency
Read optimized

Indexing is of two types
- Node based index
- Relationship based index

Cassandra

- Most common DB in industry
- Great for High Write Throughput
- Columnar DB
- DB is Masterless
  - Scalable
- Flexible as it is easy to add column
- Can handle unstructured Data
- Cassandra is peer to peer, not master slave
- Cassandra packages is called Replication factor
- Uses Consistent hashing heavily
- great for analytical
- you can configure read consistency with consistency Quorum



This is called tuning the consistency

datastax.com