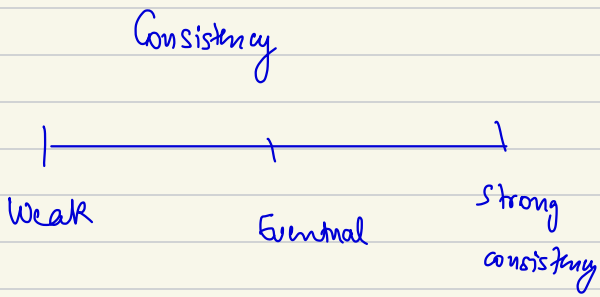
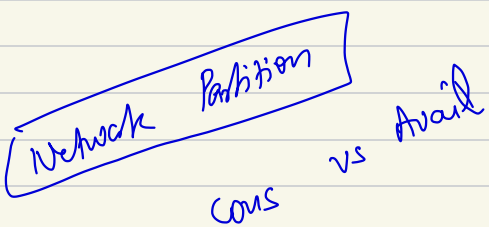
1/12/23

CAP Theorem



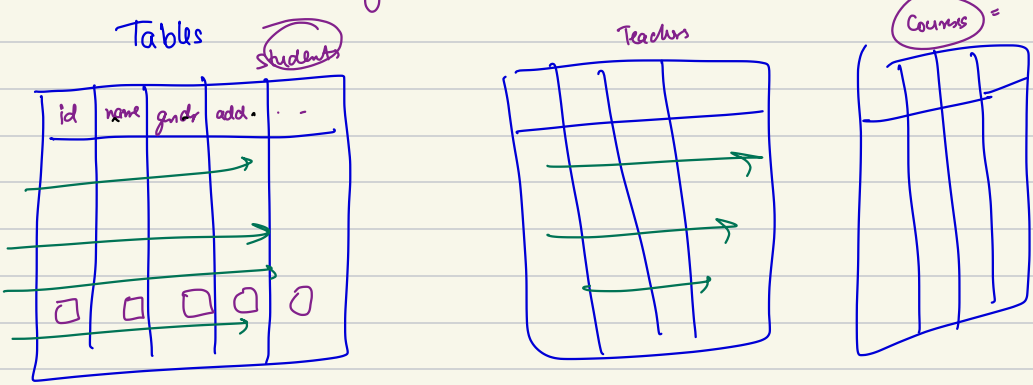
Availability



STORAGE LAYER

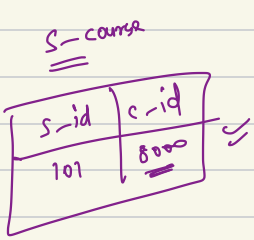
SQL vs NOSQL

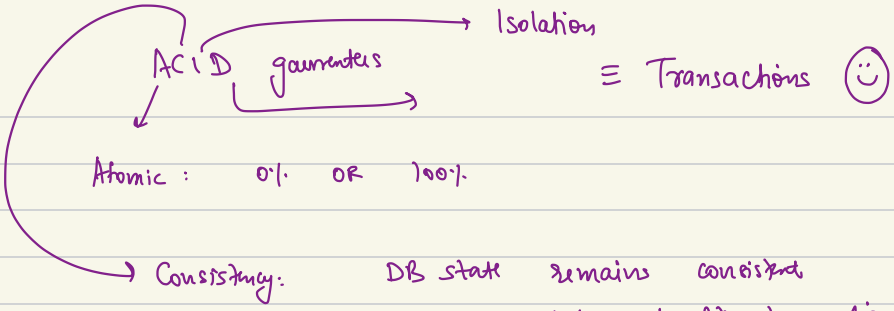
single machine database



① Data is going to follow a fixed schema

② Normalization → JOINS





PROs:

If your underlying data has fixed schema,

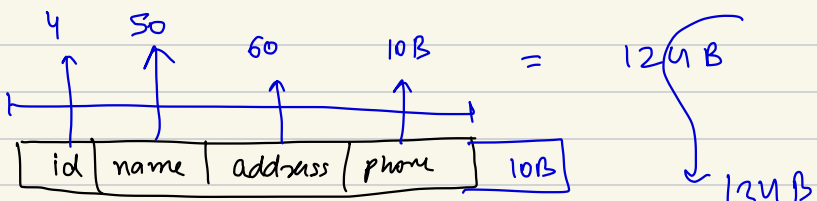
SQL allows to store data in a

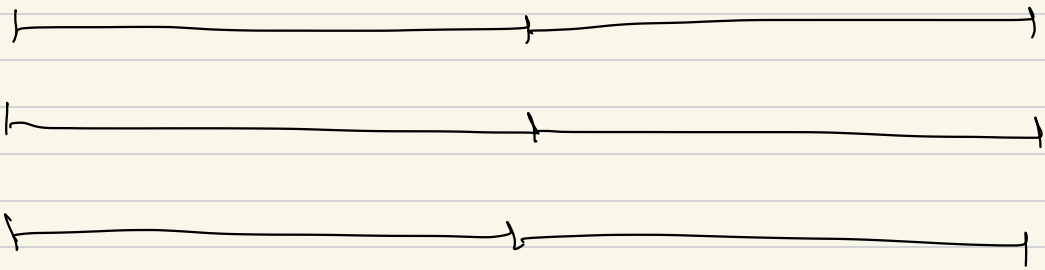
normalized manner.

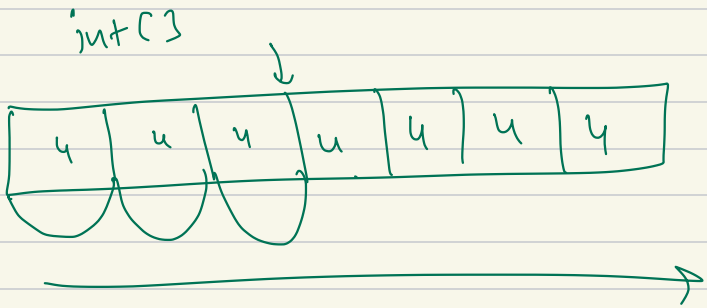
① Storage space is NOT wasted.

② Complicated JOINs relatively optimally

③ log N row access






int[]
4
4
4
4
4
4
4

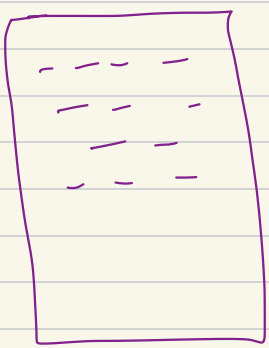
CONS

if your underlying data doesnot follow fixed schema

then a lot of CONS emerge

webpages

eg.



900 chars

↑

900B

1200 chars

↑

1200B

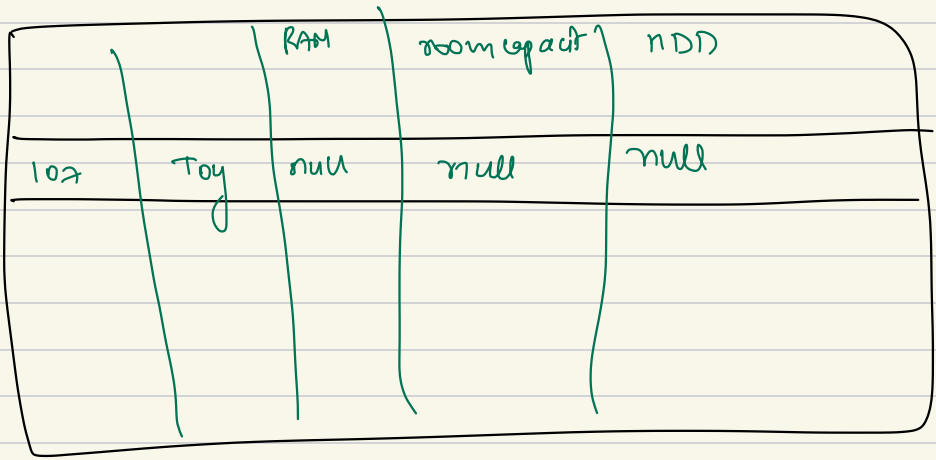
eg Amazon

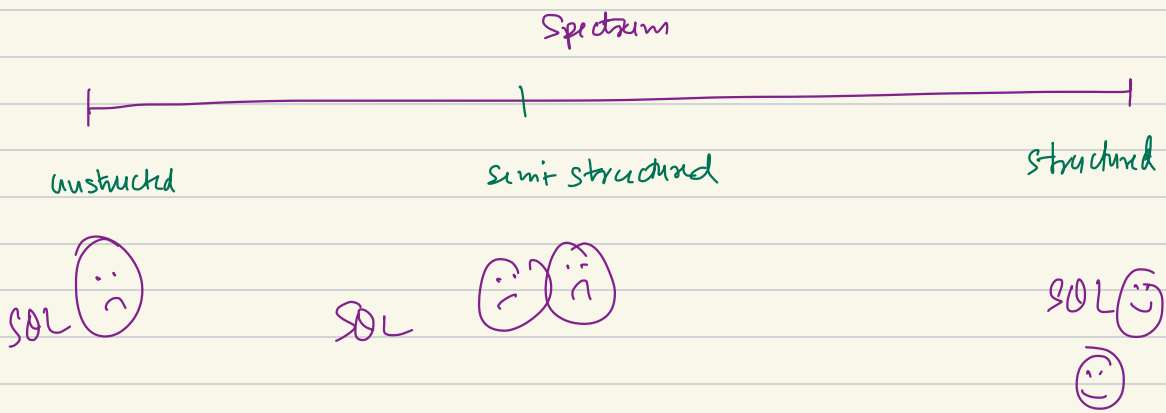
laptop
- RAM
- HDD
- Processor

air purifier
- room capacity
- power
- hepa
- Brand

toy
- Color
- age group
- material
-

1- Thousand of columns ⟶



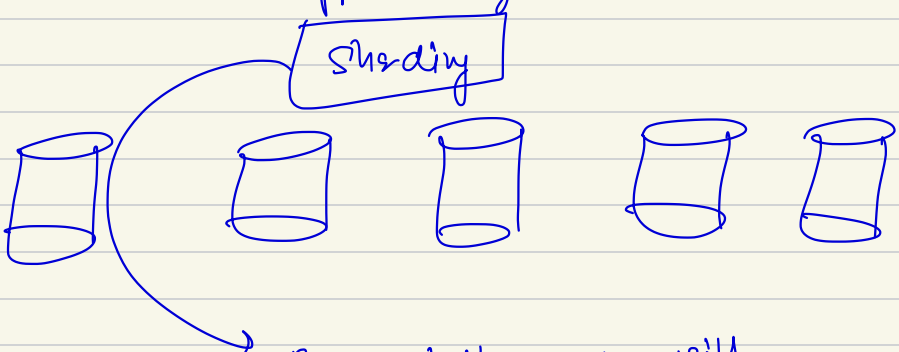


Assumption: SQL is single machine DB

→ normalization → JOINS

→ ACID guarantees

SQL to support huge data



But probably you will have to write the wrapper yourself.

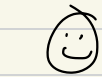
ACID guarantees → out of a single box

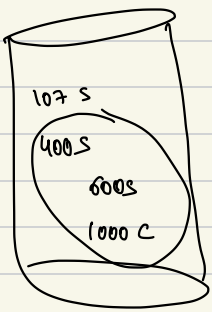
☆ Transaction on data which sits on a single machine

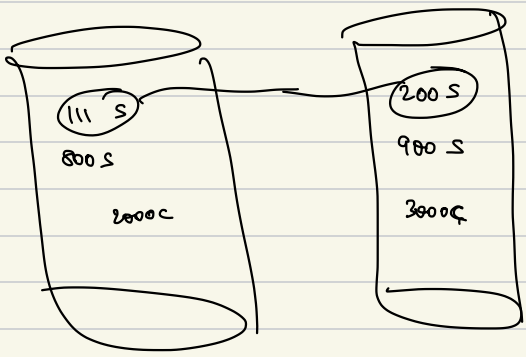
Only such transactions can give you ACID guarantees.

intra shard SQL transaction → ACID

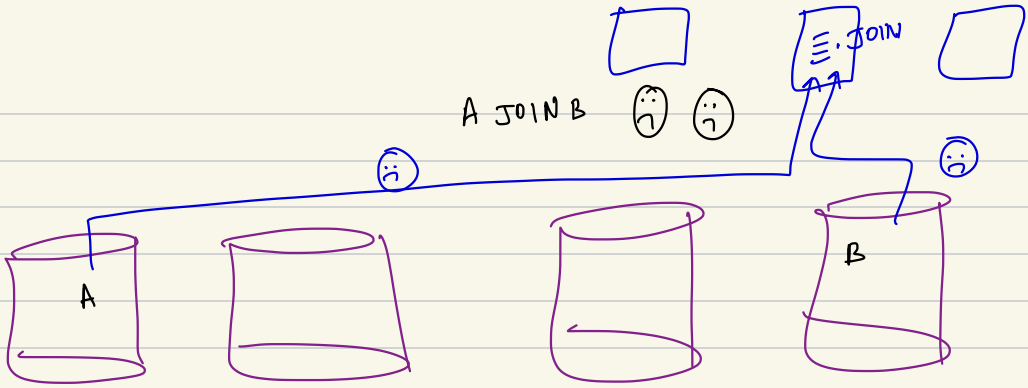
inter shard " " → NO ACID







In case of sharded SQL, the best way would be to assume that every shard behaves like an independent SQL database.



If JOINs have become super costly

[JOINs across shards are indeed super costly]

So you have to compromise normalization.

denormalization

SQL

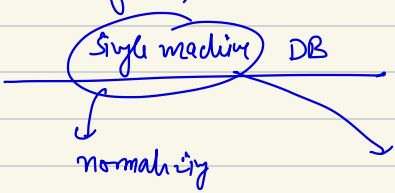
NoSQL

① structured data

semi-structured

no structured

② out of box, it is a



→ run as a cluster

⇒ denormalization

⇒ no overall ACID

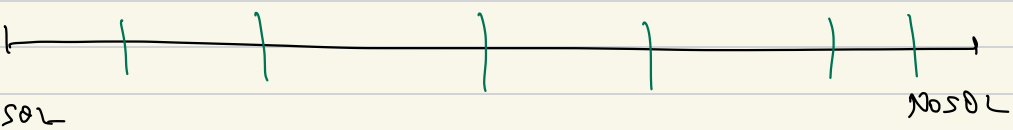
② out of the box, are cluster

DBs

⇒ denormalization

⇒ no overall ACID

Spectrum

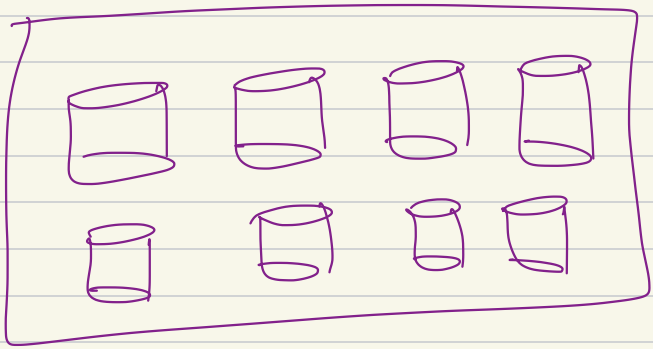


NoSQL

anything which is NOT a SQL is a NoSQL

① (semi structured, no structrd data)

② By design, horizontally sharded database.



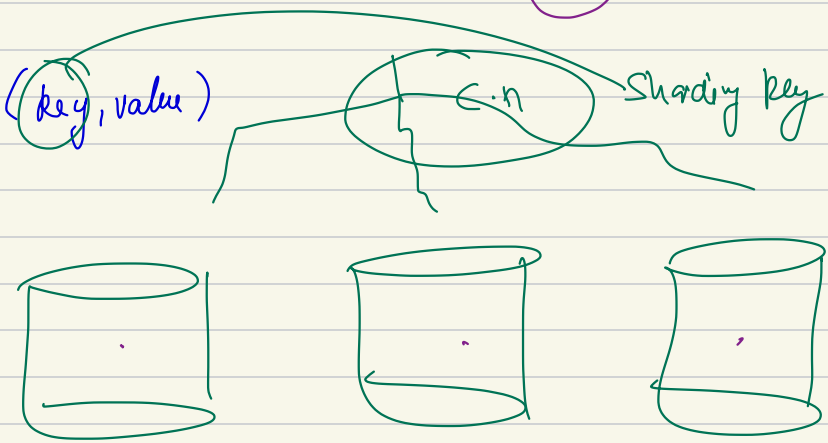
NoSQL

Sharding

Replication

① Key Value NoSQL database

Redis, Memcached.



eg

key = [aadhar number ( )

value = [" venu, 110052, male, - - - - - - "]

② Document DB — MongoDB, CouchDB, Elastic Stack

data as document

document-id, { JSON }

Sharding key → doc id

eg Amazon example (product)

107 : {
name : cat's toy
price : 400
color : Blue + orange
Category : Pets Toy ; Pets ; Toy
Seller : xyz
}

220 : {
name : Lenovo laptop
price : 80,000
Category : computers, laptops, Lenovo
hdd : 500GB
Ram : 8GB
Processor : - -
}

You can create

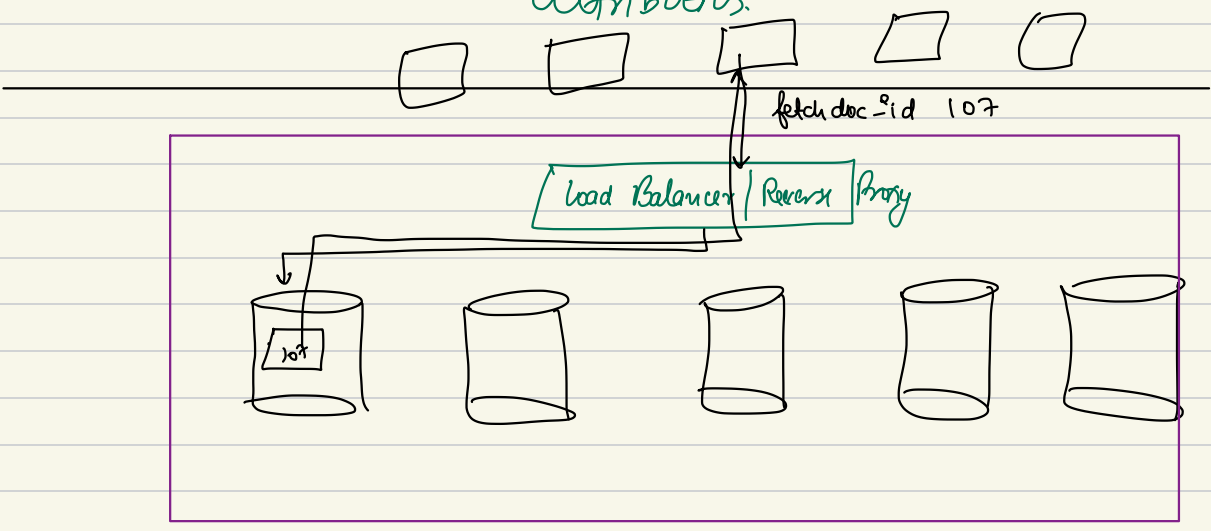
indexes

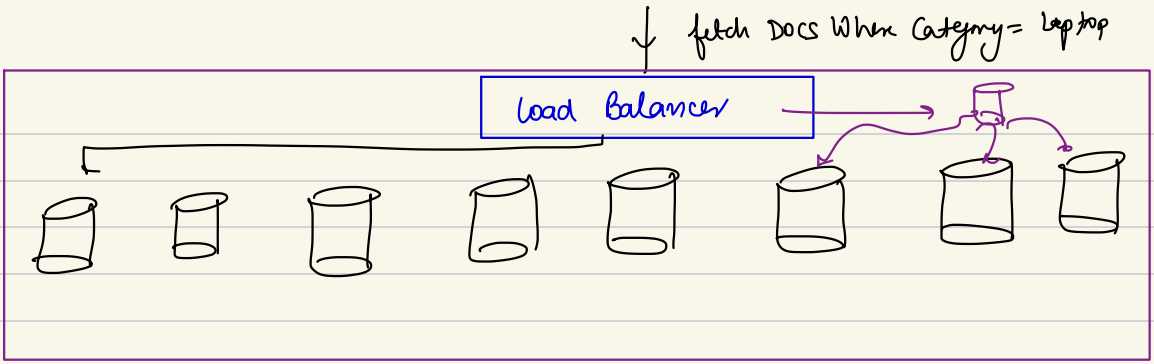
query documents

not only based on doc id

but also based on

attributes.





document dB also allows to create indexes

They allow queries (and also indexes to fasten these queries) on internal attributes of documents

documents are arranged in

collections
Tags
metadata
directories

Columnar Database.

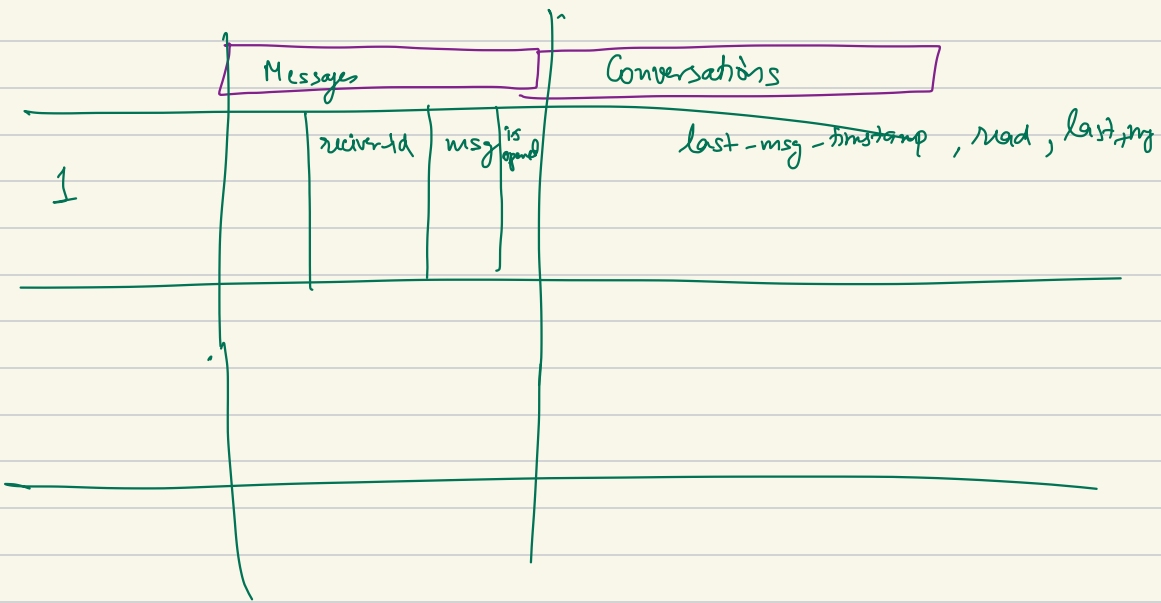
③ Wide Column Database

Column Family Database

eg Hbase, Cassandra ☺ Big Table

closest NoSQL database to a SQL

Facebook Messenger

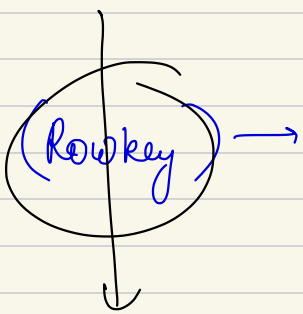


A wide column DB

has CFs (Column Families) defined

a collection of variably columns

not fixed



value of any column

sharding-key

Question

NoSQL has less checks and balances

TRUE

and to handle this, as a developer, you will have to ensure whatever checks are needed are handled in your application code at your app servers.