

Mongo-DB

- ① {
- ① document-oriented database.
 - ② high Performance, high availability, easy Scalability.
 - ③ Collection and document

② SQL

① data model.

Row and column.

② Data structure

Pre-define schema

③ Scalability

Vertical

④ Query language

SQL

⑤ Degree Property

ACID

Transaction.

No-SQL.

- ① key-value
- ② document
- ③ column family.
- ④ Graph based db.

Dynamic schema.

horizontal

According to No-SQL DB

BASE / CAP

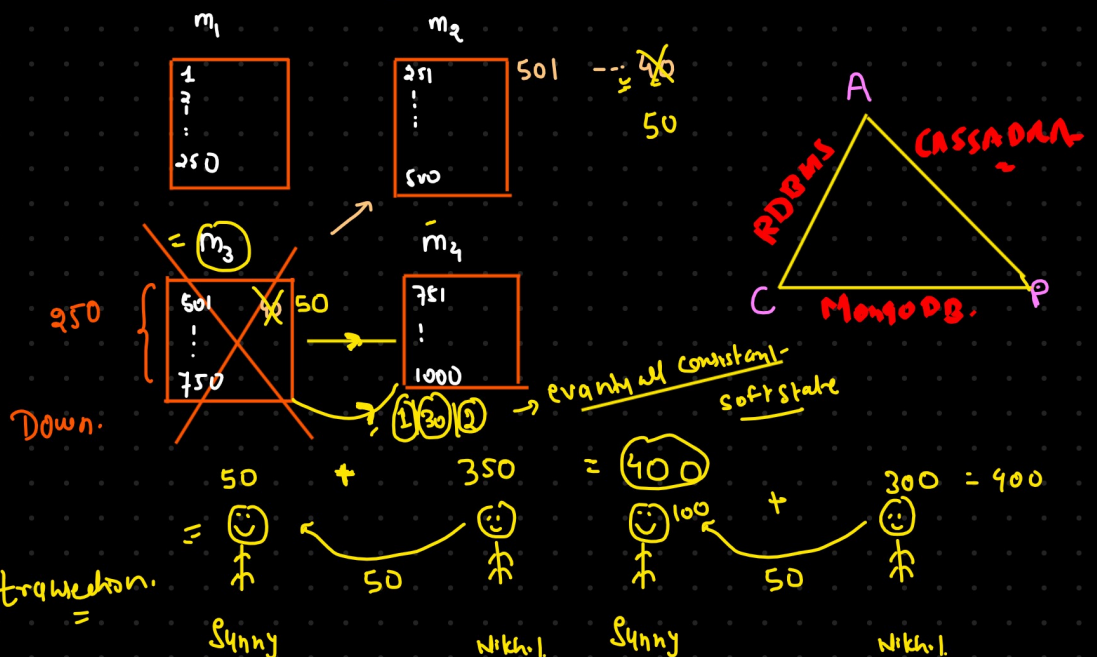
③ CAP → Consistency
Availability
Partition

CP OR AP OR CA

CAP

Replication and duplication

E.ID	E.Name	Age
1	Deepak	30
2	Rajbir	25
3	Sunny	25
4	Venkush	35
5	Shashank	32
6	-	-
!	-	-
1000	-	-



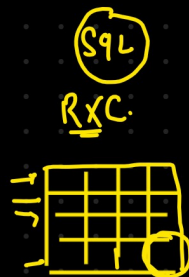
BASE ⇒ { Basic availability.
Soft State.
Eventually Consistent. }

nosql

SQL → ACID

④ HBASE | CASSANDRA | MongoDB.

- ① NO-SQL
- ② It does not follow ACID | BASE
- ③ Open-Source.
- ④ Cross-platform



- ① Key: value.
- ② Column family
- ③ document-based (S3)
- ④ Graph based DB

① Key → Numeric.
value → can be anything.
(hashing)

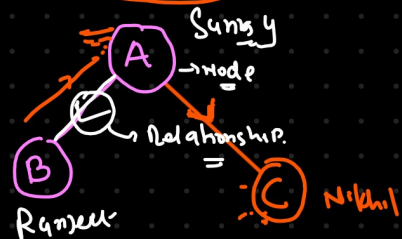
② Column family.
① Hbase
② Cassandra

③ document-based DB.

⇒ JSON BSON. Can be anything =
⇒ { key: value =

④ Graph based.

Intagram



⑤ Which DB is best/better according to req.

- ① Data model
- ② availability.
- ③ Write | Read Speed.
- ④ Work load
- ⑤ API | language | query support.
- ⑥ Index query. (mongoDB)
- ⑦ Data aggregation.

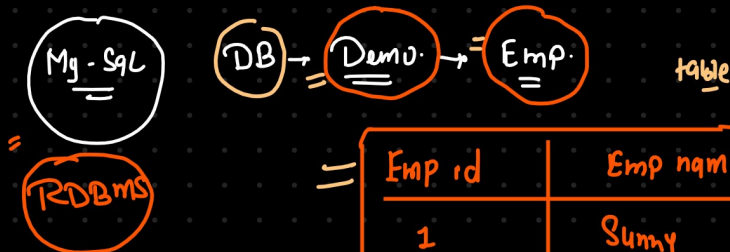
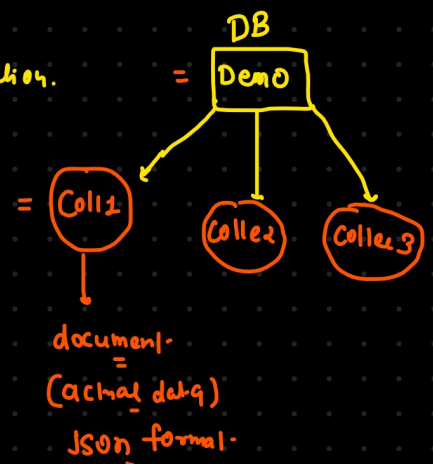
key: value, json,
graph.

(mongoDB) (Cassandra)

⑥ = HBASE | CASSANDRA | MongoDB

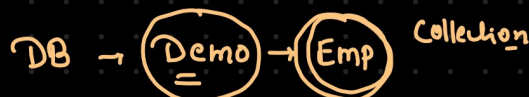
- = ① Description.
- = ② Data model.
- = ③ Query language.
- = ④ Performance.
- = ⑤ Security
- = ⑥ Replication methods. (Availability)
- = ⑦ Application/market = matrix.

- ① Datbase → Physical Container for Collection.
- ② Collection → group of documents
- ③ document → JSON view of data.
OR
key: value.



Emp id	Emp name	Age
1	Sunny	25
2	Ravi	30
3	Suvaru	28

Mongo-DB



= { { emp id : 1
emp name : Sunny
age : 25
address : ... }
= { emp id : 2
emp name : Ravi
age : 28 }
= { emp id : 3
emp name : Suvaru
age : 30 } }

(Dynamic)