DBMS - 5

16 December 2021 13:07

NoSQL with mongoDB -> document based storage

DB

SQL

-> schema / db
-> table (constraints PK, FK)
-> insert

NoSQL

-> No table, schema

-> rows / columns

-> documents, collection

-> ACID -> banking system

Performance ↓ scale ↑

secure

DB SQL

secure

security

NoSQL

millions of records

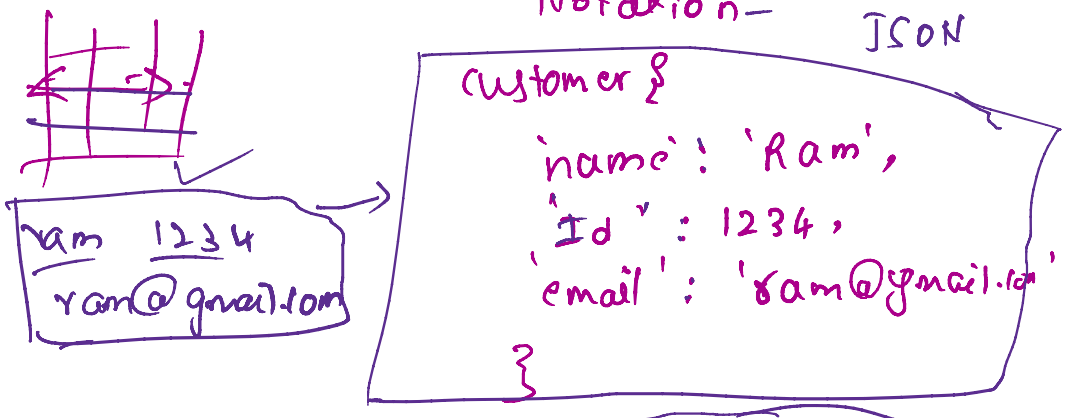
SQL

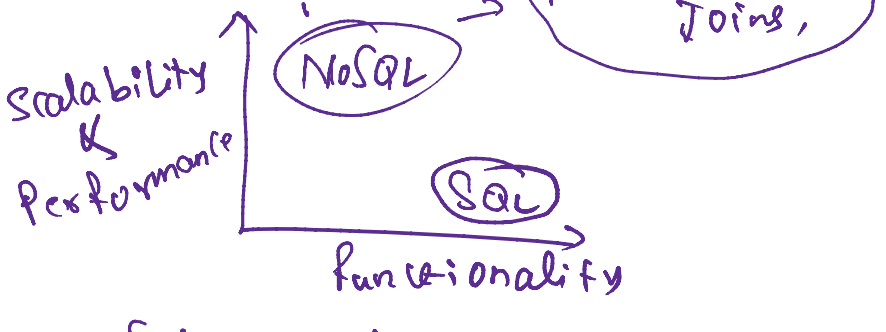
-> Rows / column
-> extract / interpret

NoSQL

-> JSON, object

Java Script Object Notation - JSON





NoSQL when to use ?

NoSQL when to use ?

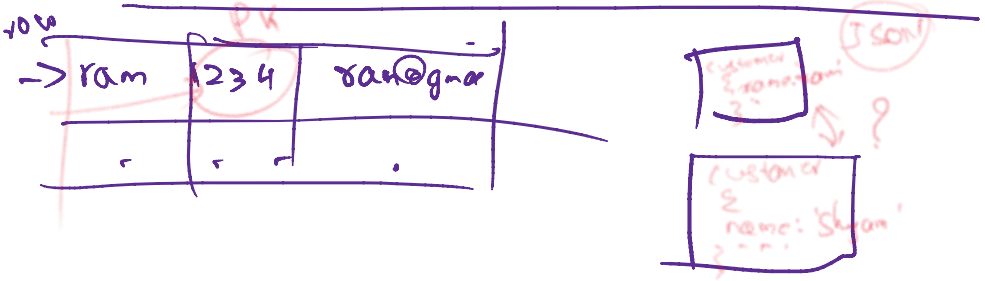
-> Huge data
-> relationship not imp
-> not structured
-> scalability.

MongoDB

Customer {

field => 'name': 'Ram', attribute/column
'Id' : 1234,
'email' : 'ram@gmail.com
}

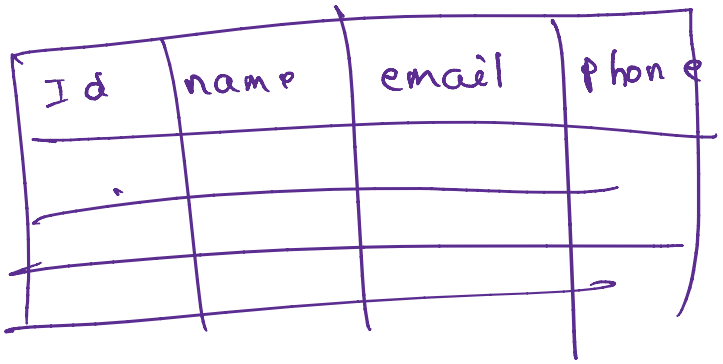
SQL	NoSQL
column	field
row/tuple	document
Table	collection
Database	Database



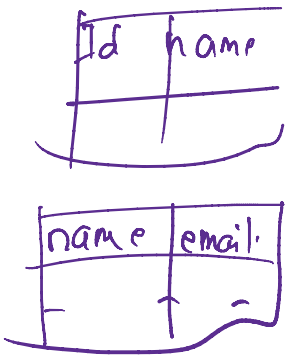
MongoDB -> Unique Id for every document

Customer{
'_Id' : <- 12 bytes Id -->
name: 'Ram',
'Id' : 1234,
'email' : 'ram@gmail.com'
}

CRUD operations



NoSQL



mongoDB commands

1) show dbs → show databases

2) use <db.name> → switch to db
(creates a new if doesn't exist)

3) db → shows db that is currently used.

4) Show collections → . . .

5) db.createCollection(name) → create new collection
↑ upper case

6) db.<collection>.insert(<JSON>)

```
db.employee.insert({
        "name": "Ram",
        "emp_id": 1001,
        "designation": "manager",
        "salary": 50000 })
```

7) Drop DB

db.dropDatabase()

8) drop a collection:

db.product.drop()
↳ collection name

Show data from collection:

1) db.collection.name.find()

db.employee.find()

} displays all data

db.employee.find().forEach(printjson)

db.employee.find().pretty()

2) db.employee.find({"name": "Ram"})

criterias { $ne $lte $gte
$lt $gt

db.employee.find({"salary":
{$gte: 40000}})

update queries:

db.collectionname.update(criteria,

db.collectionname.update(criteria, updatedata)

ex:

```
db.employee.update({"emp-id":1003},
        {$set:{"designation":"HR mgr"}},
        {multi:true})
```

optional ↑ updates all matching documents.