***MongoDB***

* *It is a NoSQL database (Document database)*
* *It stores data in JSON like document*

***Comparison between MySQL and MongoDB***

| ***Relational Database (MySQL)*** | ***MongoDB*** |
| --- | --- |
| Database | Database |
| Table | Collection |
| Row | Document |
| Column | Field |
| Joins | Embedded Documents |

***Sample document -***

*{*

*“\_id”: ObjectId("5ca4bbcea2dd94ee58162a69"),*

*“first\_name”: “Roushan”,*

*“last\_name”: “Kumar”,*

*“organization”: “Sigmoid”,*

*“email”: “roushan.kumar@sigmoid.io”*

*}*

***Setup MongoDB locally -***

***Installing and running MongoDB on MacOS*** - [Install MongoDB Community Edition on macOS — MongoDB Manual](https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/)

***MongoDB UI*** - [Robo 3T | Free, open-source MongoDB GUI (formerly Robomongo)](https://robomongo.org/)

***CRUD operations -***

* ***CREATE***

*insert*

*insertOne*

*insertMany*

* ***READ***

*find*

* ***UPDATE***

*update*

*updateMany*

* ***DELETE***

*delete*

*deleteMany*

***// insert single document***

*db.getCollection('users').insert([*

*{'name': "Roushan", "dept": "DE", "enail": "roushan.kr@sigmoid.com"}*

*])*

***// insert multiple documents***

*db.getCollection('users').insertMany([*

*{'name': "Akash", "dept": "DE", "enail": "akash.kr@sigmoid.com"},*

*{'name': "Sachin", "dept": "DS", "enail": "sachin.kr@sigmoid.com"}*

*])*

***// update users set dept = Data Engg where name = Akash***

*db.getCollection("users").update({"name": "Akash"}, {$set: {dept: "Data Engg"}})*

***// delete from users where name = Akash***

*db.getCollection("users").remove({"name": "Akash"})*

***// rename a column***

*db.getCollection('users').updateMany({}, {$rename: {enail: "email"}})*

***// create unique index on single column***

*db.users.createIndex({name: 1}, {unique: true})*

***// create unique index on multiple columns***

*db.users.createIndex({name: 1, dept: 1, email: 1}, {unique: true})*

***Loading Sample datasets into MongoDB*** - [Sample Analytics Dataset](https://drive.google.com/file/d/1tx9mfXIAd9TWPHieNqHpkHlvhA05uM0O/view?usp=sharing)

*mongoimport customers.json -d json\_db -c customers*

*mongoimport accounts.json -d json\_db -c accounts*

***Queries on accounts Collection -***

***// count total records***

*db.getCollection('accounts').count()*

***// rename limit to traxLimit***

*db.students.updateMany( {}, { $rename: { "limit": "traxLimit" } } )*

***// distinct traxLimit values***

*db.accounts.distinct("traxLimit")*

***// Find max traxLimit***

*db.accounts.find().sort({traxLimit: -1}).limit(1)*

***// Find min traxLimit***

*db.accounts.find().sort({traxLimit: 1}).limit(1)*

***// Find min and max traxLimit***

*db.accounts.aggregate([*

*{*

*$group: {*

*\_id: null,*

*maxLimit: { $max: "$traxLimit" },*

*minLimit: { $min: "$traxLimit" }*

*}*

*},*

*{*

*$project: {*

*\_id: 0*

*}*

*}*

*])*

***// Total number of accounts by traxLimit***

*db.accounts.aggregate([*

*{*

*$group: {*

*\_id: "$traxLimit", totalAccounts: {$sum: 1}*

*}*

*},*

*{*

*$project: {*

*\_id: 0, traxLimit: "$\_id", totalAccounts: 1*

*}*

*},*

*{$sort: {traxLimit: -1}}*

*])*

*db.accounts.find({products: {$all: ["InvestmentStock", "Brokerage"]}})*

*db.accounts.find({*

*$and: [*

*{ products: {$size: 2} },*

*{ products: {$all: ["InvestmentStock", "Brokerage"]} }*

*]*

*})*

***// Max traxLimit***

*db.accounts.find({}, {\_id: 0, traxLimit: 1}).sort({traxLimit: 1}).limit(1)*

*db.accounts.distinct("traxLimit")*

***// Find the records that contain both "InvestmentFund", "InvestmentStock" as products***

*db.accounts.find({*

*$and: [*

*{ products: {$all: ["InvestmentFund", "InvestmentStock"]} },*

*{ products: {$size: 2} }*

*]*

*})*

***Queries on customers collection -***

***// top 5 customers with maximum number of accounts***

*db.getCollection('customers').aggregate([*

*{$project: {*

*name: "$name", totAccounts: {$size: "$accounts"}*

*}},*

*{$sort: {totAccounts: -1}},*

*{$limit: 5},*

*])*

***// Total customers born by year***

*db.getCollection('customers').aggregate([*

*{$project: {*

*bYear: {$year: "$birthdate"}*

*}},*

*{$group: {\_id: "$bYear", count: {$sum: 1}}},*

*{$sort: {\_id: 1}}*

*])*

***// email domains and their counts***

*db.getCollection('customers').aggregate([*

*{$project: {*

*emailDomain: { $arrayElemAt: [ {$split: ["$email", "@"]}, 1 ] }*

*}},*

*{$group: {\_id: "$emailDomain", count: {$sum: 1}}}*

*])*

***// Find duplicate usernames***

*db.customers.aggregate([*

*{$group: {\_id: "$username", count: {$sum: 1}}},*

*{$match: {count: {$gt: 1}}}*

*])*

***// Customers who born in a specific Month***

*// 1st Way*

*db.getCollection('customers').find({birthdate: {$gte: new Date("1994-02-01"), $lt: new Date("1994-03-01")}})*

*// 2nd Way*

*db.getCollection('customers').aggregate([*

*{$project: {*

*\_id: 0, username: "$username", name: "$name", birthdate: "$birthdate", year: {$year: "$birthdate"}, month: {$month: "$birthdate"}*

*}},*

*{$match: {*

*year: 1994, month: 2*

*}}*

*])*

***// Find duplicate usernames***

*db.customers.aggregate([*

*{$group: {\_id: "$username", count: {$sum: 1}}},*

*{$match: {count: {$gt: 1}}}*

*])*

***// count of people born in each year sorted by tear in ascending order***

*db.customers.aggregate([*

*{$project: {year: {$year: "$birthdate"}}},*

*{$group: { \_id: "$year", totPeople: {$sum: 1} }},*

*{$project: { year: "$\_id", totPeople: 1, \_id: 0 }},*

*{$sort: {totPeople: -1}}*

*])*

***// Find 5 customers with maximum number of accounts***

*db.getCollection('customers').aggregate([*

*{$project: {name: "$name", totAccounts: {$size: "$accounts"}}},*

*{$sort: {totAccounts: -1}},*

*{$limit: 5}*

*])*