Comandos de MongoDB

Comandos primarios

mongosh mongodb+srv://sergio:root123456@cluster0.4i0l5oa.mongodb.net/	Acceder al CLI de Mongo Shell
show dbs	Listar las bases de datos
use <nombre_base_datos>	Seleccionar una base de datos
show collections	Listar las colecciones (entidades)
db.createCollection("personas")	Crear una nueva collection
db.coll.drop()	Eliminar una collection
exit	Salir del CLI de Mongo Shell

Comandos de CRUD

Create	
db.coll.insertOne({ nombre: "Juan" })	Inserta un nuevo documento (registro)
db.coll.insertMany([{ nombre: "Pablo" }, { nombre: "Angel" }])	Inserta dos nuevos documentos ordenándolos por el primer atributo de dichos documentos
db.coll.insertMany ([{ nombre: "Pablo" }, { nombre:"Angel" }], { ordered: false })	Inserta dos nuevos documentos sin ordenar

*Durante la inserción, mongoDB creará el atributo **_id** y le asignará un valor único de tipo ObjectId(). De ser necesario, se puede declarar un atributo **_id** específico pero; el valor de este no se podrá repetir dentro de la colección.*

Read – Funciones primarias	
db.coll.findOne({ nombre: "Pablo" })	Devuelve un documento (registro) que coincidan con el nombre Pablo
db.coll.find([{ nombre: "Pablo" }])	Devuelve los documentos que coincidan con el nombre Pablo (por defecto, retorna 20 documentos)
db.coll.find([{ nombre: "Pablo" }, { edad: 21 }])	Devuelve los documentos que coincidan con el nombre Pablo y con la edad 21 (esto es un AND implicito)
db.coll.find().batchSize(30)	Devuelve 30 documentos que coincidan con la consulta
db.coll.find().sort({ idx_nombre: 1 })	Devuelve 20 documentos ordenados ascendentemente por el atributo “nombre”
db.coll.find().limit(10)	Devuelve 10 documentos que coincidan con la consulta
db.coll.find().skip(5).limit(10)	Omite los primeros 5 documentos que coincidan con la consulta retornando los siguientes 10 documentos
db.coll.distinct("ciudad")	Devuelve el primer documento de

	cada conjunto de documentos que tengan una misma ciudad

READ – Funciones de comparación	
db.coll.find({ edad: { $eq: 21 } })	Devuelve los documentos en donde el valor de edad es equivalente a 21 (equal)
db.coll.find({ edad: { $gt: 21 } })	Devuelve los documentos en donde el valor de edad es mayor a 21 (greater than)
db.coll.find({ edad: { $gte: 21 } })	Devuelve los documentos en donde el valor de edad es mayor o igual a 21
db.coll.find({ edad: { $in: [18, 20, 25] } })	Devuelve los documentos en donde el valor de edad es igual a 18, 20 o 25 (in)
db.coll.find({ edad: { $lt: 21 } })	Devuelve los documentos en donde el valor de edad es menor a 21 (less than)
db.coll.find({ edad: { $lte: 21 } })	Devuelve los documentos en donde el valor de edad es menor o igual a 21
db.coll.find({ edad: { $ne: 21 } })	Devuelve los documentos en donde el valor de edad es diferente de 21 (not equal)
db.coll.find({ edad: { $nin: [18, 20, 25] } })	Devuelve los documentos en donde el valor de edad es diferente de 18, 20 o 25 (not in)

READ – Funciones lógicas	
db.coll.find({ $and: [{ nombre: { $eq: "Juan" } }, { edad: { $gte: 21 } }] })	Devuelve los documentos en donde el valor de nombre es Juan y el valor de edad es mayor o igual a 21
db.coll.find({ $or: [{ nombre: { $eq: "Juan" } }, { edad: { $lt: 21 } }] })	Devuelve los documentos en donde el valor de nombre es Juan o el valor de edad es menor a 21
db.coll.find({ nombre: { $not: "Juan" } })	Devuelve los documentos en donde el valor de nombre es diferente a Juan

De ser necesario, se puede combinar estas funciones de forma anidada.

READ – Funciones elementales	
db.coll.find({ dni: { $exists: true } })	Devuelve los documentos que poseen el atributo dni. Es decir, determina los atributos nulos o faltantes
db.coll.find({ dni: { $type: "string" } })	Devuelve los documentos en donde el valor de dni es de tipo string

Tipo de dato $type	Código	Alias	Nota de actualidad
Double	1	"double"	
String	2	"string"	
Object	3	"object"	
Array	4	"array"	
Binary data	5	"binData"	
Undefined	6	"undefined"	Deprecated.
ObjectId	7	"objectId"	
Boolean	8	"bool"	
Date	9	"date"	
Null	10	"null"	
Regular Expression	11	"regex"	
DBPointer	12	"dbPointer"	Deprecated.
JavaScript	13	"javascript"	
Symbol	14	"symbol"	Deprecated.
JavaScript code with scope	15	"javascriptWithScope"	Deprecated
32-bit integer	16	"int"	
Timestamp	17	"timestamp"	
64-bit integer	18	"long"	
Decimal128	19	"decimal"	
Min key	-1	"minKey"	
Max key	127	"maxKey"	

READ – Funciones de evaluación	
db.coll.find({ nombre: { $regex: /^Ju/ } })	Devuelve los documentos en donde el valor de nombre comienza con Ju
db.coll.find({ $text: { $search: "Juan", $caseSensitive: true } })	Devuelve los documentos en donde el valor de nombre es Juan pero distingue entre minúsculas y mayúsculas (por defecto, false). Es necesario un índice de tipo text

En este link https://regex101.com/ puedes encontrar referencias y hacer practicas de patrones regex..

UPDATE	
db.coll.updateOne({ nombre: "Juan"}, { $set: { nombre: "Pedro" } })	Reemplaza el nombre (Pedro) del primer documento que tenga como nombre Juan
db.coll.updateOne({ id: 10 }, { $set: { nombre: "Lorena", edad: 30 } })	Reemplaza el nombre y edad del documento que tenga como id el valor 10
db.coll.updateMany({ nombre: "Juan"}, { $set: { nombre: "Pedro" } })	Reemplaza el nombre (Pedro) de todos los documentos que tengan como nombre Juan

DELETE	
db.coll.deleteOne({ nombre: "Juan"})	Elimina el primer documento que tenga como nombre Juan
db.coll.deleteMany ({ nombre: "Lorena" })	Elimina todos los documento que tengan como nombre Lorena
db.coll.deleteMany({ })	Elimina todos los documentos de la colección