Day 4

Mongodb Comments :

Install mongodb

Choose file path

Run mongosh

Check command : mongod

Start :

Use Sarath (giving file name )

db.product.insertOne({}) ( update products details)

db.product.findOne() ( to check the one product )

db.product.find() ( to check all products)

ctrl + L ( clear the wrong commands)

ctrl + Enter ( for next line)

CRED operation – Create Read Edit Delete operation

Show dbs ( files check )

test> use office

switched to db office

office> db.employee.insertOne({name:"Logesh", details: { position: "PHP Developer" salary:40000 }}}

,

{

acknowledged: true,

insertedId: ObjectId("636a6b8db5980aee606b28a3")

}

office> db.employee.insertMany([

{name:"Rahul", details: { position: "Java Developer", salary:60000 } },

{name:"Nizith", details: { position: "SEO Analyst", salary: 30000 } },

{ name: "Daniel", details: { position: "PHP Developer", salary:50000} }

])

{

acknowledged: true,

insertedIds: {

'0': ObjectId("636a6c9cb5980aee606b28a4"),

'1': ObjectId("636a6c9cb5980aee606b28a5"),

'2': ObjectId("636a6c9cb5980aee606b28a6")

}

}

office> db.employee.findOne({name:"Nizith"}) ( to check one person details )

{

\_id: ObjectId("636a6c9cb5980aee606b28a5"),

name: 'Nizith',

details: {position: 'SEO Analyst', salary: 30000 }

}

Db.employee.find()

office> db.employee.find({"details.position":"PHP Developer"})

[

{

\_id: ObjectId("636a6b8db5980aee606b28a3"),

name: 'Logesh',

details: {position: 'PHP Developer', salary: 40000 }

},

{

\_id: ObjectId("636a6c9cb5980aee606b28a6"),

name: Daniel',

details: {position: 'PHP Developer', salary: 50000 }

}

]

office> db.employee.updateOne({ name: "Nizith" }, { $set:{ "details.position":"Java Developer" }})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount:|

upsertedCount: 0

}

office> db.employee.updateMany( { "details.position":"PHP Developer" }, { $set:{ "details.salary":60000 } } }

{

acknowledged: true,

insertedId: null,

matchedCount: 2,

modifiedCount: 2,

upsertedCount: 0

}

office> db.employee.deleteOne( {name:"Nizith" } }

{ acknowledged: true, deletedCount: 1 }

office> db.employee.deleteMany( { "details.position":"PHP Developer" } }

{ acknowledged: true, deletedCount: 2 }

Creating Collection :

library> db.createCollection('users', { })

{ ok: 1 }

library> db.users.find()

library> db.users.find({})

library> db.users.insertOne({name:"Logesh"})

{

acknowledged: true,

insertedId: ObjectId("6374a48431248b79fb30b5ba")

}

library> db.users.find({})

[ { \_id: ObjectId("6374a48431248b79fb30b5ba"), name: 'Logesh' } ]

library>

office> db.employee.insertMany ( [

{name:"Tameem", details: { position: "Python Developer", salary:80000 } },

{ name: "Babu", details: { position: "UI Designer", salary: 30000}},

]).

office> db.employee.find({ name:"Tameem" })

[

{

\_id: ObjectId("636d14060b0faa0e204a8608"),

name: 'Tameem',

details: {position: 'Python Developer', salary: 80000 }

}

]

office> db.employee.find( { "details.salary":{ $gt: 50000 } } )

[

{

\_id: ObjectId("636cf8f88b0faa0e204a8606"),

name: 'Rahul',

details: {position: 'Java Developer', salary: 60000 }

},

{

\_id: ObjectId("636d14060b0faa0e204a8608"),

name: 'Tameem',

details: {position: 'Python Developer', salary: 80000 }

}

]

office> db.employee.find( { "details.salary":{ $lt: 50000 } } >

[

{

id: ObjectId("636d14060b0faa0e204a8609"),

name: 'Babu',

details: {position: 'UI Designer', salary: 30000 }

}

]

db.customers.insertMany()

db.customers.find().toArray() ( many insert)

We can use javascripts also

Eg: console.log(“”)

db.customers.find().forEach(data)=>{ print(data))}) (this command will loop every data)

Projection :

Db.employee.find({},{name:1}) (filter only name in the database)

Db.employee.find({},{name:1, \_id:0}) if we want to remove id in the name section use this command )

Db.employee.find().limit(2) (it will limit the count )

db.employee.find().skip(2) (it will skip first 2 data )

db.employee.find().sort({name:1}) (ascending order)

db.employee.find().sort({name:-1}) (descending order)

Query Operators :

Comparison

Logical

Evaluation

db.employee.find({ email: { $eq :” mail id “}}) ( extracting related value)

db.employee.find({ email: { $ne :” mail id “}}) ( extracting non-related value)

db.employee.find( { "details.salary":{ $gt: 50000 } } ) greter than values

db.employee.find( { "details.salary":{ $lt: 50000 } } ) lower than values

db.employee.find( { "details.salary":{ $gte: 50000 } } ) greater than or equal value

db.employee.find( { "details.salary":{ $lte: 50000 } } ) lower than or equal values

$regres – allows the use of regular expression

db.employee.find({ name : { $regex:”J” } } ) ( only if the specific letter is capital it will show )

db.employee.find({ name : { $regex:”J”, $options : “I” } } ) ( if the letter is available it will show capital or small )