|  |
| --- |
| MONGODB COMMANDS |
| MOHAMMAD ADNAN SHARIFF  ID:2200090035  Branch: CSIT  [Website] |

PART-1

Microsoft Windows [Version 10.0.22631.3085]

(c) Microsoft Corporation. All rights reserved.

C:\Users\adnan shariff>mongosh

Current Mongosh Log ID: 65c3572ab7d9b113960911a6

Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.1.1

Using MongoDB: 6.0.8

Using Mongosh: 2.1.1

mongosh 2.1.3 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------

The server generated these startup warnings when booting

2024-02-06T12:31:59.548+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted

------

test> show dbs // shows all the databases available

admin 40.00 KiB

config 84.00 KiB

local 120.00 KiB

test> use studentdb //creates a new database and creates a new database

switched to db studentdb

studentdb> db.createCollection("student") //used to create a collection

{ ok: 1 }

studentdb> show dbs

admin 40.00 KiB

config 84.00 KiB

local 120.00 KiB

studentdb 8.00 KiB

studentdb> db

studentdb

studentdb> show collections //shows all the collections available in database

student

studentdb> db.student.drop() //used to delete a collection in a database

true

studentdb> show collections

studentdb> db.dropDatabase() //used to delete the database

{ ok: 1, dropped: 'studentdb' }

studentdb> show dbs

admin 40.00 KiB

config 108.00 KiB

local 120.00 KiB

studentdb> use admin

switched to db admin

admin> use studentdb

switched to db studentdb

studentdb> //create collection

studentdb> db.createCollection("student")

{ ok: 1 }

studentdb> show collections

student

studentdb> db.student.insert({sid:30001,sname:"KLU"}) //used to insert documents and it’s fields in collection

DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.

{

acknowledged: true,

insertedIds: { '0': ObjectId('65c35f7db7d9b113960911a7') }

}

studentdb> db.student.find() //used to show all the documents available in the collection

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

}

]

studentdb> db.student.insertOne({sid:30002,sname:"klef"}) //used to insert only document

{

acknowledged: true,

insertedId: ObjectId('65c36070b7d9b113960911a8')

}

studentdb> db.student.insertOne({sid:30003,sname:"klef",dept:"CSIT"})

{

acknowledged: true,

insertedId: ObjectId('65c360cbb7d9b113960911a9')

}

studentdb> db.student.find()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

}

]

studentdb> db.student.find().pretty() //.pretty() used to display all the documents in clean format

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

}

]

studentdb> db.student.find()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

}

]

studentdb> db.student.insertMany([{sid:30004},{sid:30005,sname:"30865"}]) //used to insert multiple documents in a collection

{

acknowledged: true,

insertedIds: {

'0': ObjectId('65c362b2b7d9b113960911aa'),

'1': ObjectId('65c362b2b7d9b113960911ab')

}

}

studentdb> db.student.find().size() //used to find the number of records

5

studentdb> db.student.countDocuments() // used to count the number of documents in a collection

5

studentdb> db.student.find().limit(1) //used get the required number of documents

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

}

]

studentdb> db.student.find().limit(2)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

}

]

studentdb> db.student.find().limit(14)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 },

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: '30865'

}

]

studentdb> db.student.find().sort({sid:1}) //used to sort the documents either ascending (1) or descending(1)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 },

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: '30865'

}

]

studentdb> db.student.find().sort({sid:-1})//descending order

[

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: '30865'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 },

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

}

]

studentdb> db.student.find().sort()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 },

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: '30865'

}

]

studentdb> db.student.updateOne({sid:30005},{$set:{sname:"RAMA"}}) //used to update any record in a particular document

{

acknowledged: true,

insertedId: null,77

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.student.find()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 },

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: 'RAMA'

}

]

studentdb> db.studentdb.find().limit(1)

studentdb> db.studentdb.find().limit(4)

studentdb> db.student.find().limit(1)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

}

]

studentdb> db.student.find().limit(2)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

}

]

studentdb> db.student.find().limit(3)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

}

]

studentdb> db.student.find().limit(4)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 }

]

studentdb> db.student.find().limit(5)

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef'

},

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSIT'

},

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 },

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: 'RAMA'

}

]

studentdb> db.student.find({sid:30005})

[

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: 'RAMA'

}

]

studentdb> db.student.findOne({sid:30001}) //used to find only one document

{ \_id: ObjectId('65c35f7db7d9b113960911a7'), sid: 30001, sname: 'KLU' }

studentdb> db.student.findOne({sid:30004})

{ \_id: ObjectId('65c362b2b7d9b113960911aa'), sid: 30004 }

studentdb> db.student.updateOne({sid:30004},{$set:{sname:"manish",dept:"CSE"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.student.findOne({dept:"CSE"})

{

\_id: ObjectId('65c362b2b7d9b113960911aa'),

sid: 30004,

dept: 'CSE',

sname: 'manish'

}

studentdb> db.student.updateOne({dept:"CSIT"},{$set:{dept:"CSE"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.student.findOne({dept:"CSE"})

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSE'

}

studentdb> db.student.find({sid:30001,sname:"KLU"})

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU'

}

]

studentdb> db.student.find({sid:30001,sname:"ECE"})

studentdb> db.student.find({sid:30005})

[

{

\_id: ObjectId('65c362b2b7d9b113960911ab'),

sid: 30005,

sname: 'RAMA'

}

]

studentdb> db.student.deleteOne({sid:30005}) //used to delete one particular document based on condition

{ acknowledged: true, deletedCount: 1 }

studentdb> db.student.c

studentdb.student.c

studentdb> db.student.countDocuments()

4

studentdb> db.student.find({dept:"CSE"})

[

{

\_id: ObjectId('65c360cbb7d9b113960911a9'),

sid: 30003,

sname: 'klef',

dept: 'CSE'

},

{

\_id: ObjectId('65c362b2b7d9b113960911aa'),

sid: 30004,

dept: 'CSE',

sname: 'manish'

}

]

studentdb> db.student.deleteMany({dept:"CSE"}) //used to delete multiple documents

{ acknowledged: true, deletedCount: 2 }

studentdb> db.student.countDocuments()

2

studentdb>

PART-2

MongoDBCommands Part -2

test> show dbs

admin 40.00 KiB

config 72.00 KiB

local 120.00 KiB

studentdb 72.00 KiB

test> use studentdb

switched to db studentdb

studentdb> db

studentdb

studentdb> show collections

student

studentdb> db.student.find()

studentdb> db.student.updateMany({},{$set:{gender:"FEMALE"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 2,

modifiedCount: 2,

upsertedCount: 0

}

studentdb> db.student.find()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU',

gender: 'FEMALE'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'klef',

gender: 'FEMALE'

}

]

studentdb> db.student.updateOne({sid:30002},{$set:{sname:"RAMA"}}

... )

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.student.find()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU',

gender: 'FEMALE'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'RAMA',

gender: 'FEMALE'

}

]

studentdb> db.student.updateMany({sid:30003},{$set:{sname:"SRINU",gender:"MALE"}},{upsert:true})

{

acknowledged: true,

insertedId: ObjectId('65ca5c5a553c341947e4ab94'),

matchedCount: 0,

modifiedCount: 0,

upsertedCount: 1

}

studentdb> db.student.find()

[

{

\_id: ObjectId('65c35f7db7d9b113960911a7'),

sid: 30001,

sname: 'KLU',

gender: 'FEMALE'

},

{

\_id: ObjectId('65c36070b7d9b113960911a8'),

sid: 30002,

sname: 'RAMA',

gender: 'FEMALE'

},

{

\_id: ObjectId('65ca5c5a553c341947e4ab94'),

sid: 30003,

gender: 'MALE',

sname: 'SRINU'

}

]

studentdb> db.student.find({},{sname:1})

[

{ \_id: ObjectId('65c35f7db7d9b113960911a7'), sname: 'KLU' },

{ \_id: ObjectId('65c36070b7d9b113960911a8'), sname: 'RAMA' },

{ \_id: ObjectId('65ca5c5a553c341947e4ab94'), sname: 'SRINU' }

]

studentdb> db.student.find({},{sname:1,\_id:0})

[ { sname: 'KLU' }, { sname: 'RAMA' }, { sname: 'SRINU' } ]

studentdb> db.student.updateOne({sid:30001},{$set:{sname:"KLU"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 0,

upsertedCount: 0

}

studentdb> db.student.distinct("MALE") //

[]

studentdb> db.student.distinct("gender")

[ 'FEMALE', 'MALE' ]

studentdb> db.student.find({}).size()

3

studentdb> db.student.find({gender:"MALE"}).size()

1

studentdb> db.student.find({gender:"FEMAKE"}).size()

0

studentdb> db.student.find({gender:"FEMALE"}).size()

2

studentdb> db.student.countDocument({gender:"FEMALE"})

TypeError: db.student.countDocument is not a function

studentdb> db.student.countDocument({gender:"FEMALE"})

TypeError: db.student.countDocument is not a function

studentdb> db.student.countDocuments({gender:"FEMALE"})

2

studentdb> db.student.countDocuments({})

3

studentdb> db.createCollection("emp")

{ ok: 1 }

studentdb> show collections

emp

student

studentdb> db.emp.insertMany([{\_id:1,name:"A"},{\_id:2,name:"B"}])

{ acknowledged: true, insertedIds: { '0': 1, '1': 2 } }

studentdb> db.emp.insertMany([{\_id:3,name:"C"},{\_id:4,name:"C"}])

{ acknowledged: true, insertedIds: { '0': 3, '1': 4 } }

studentdb> db.emp.find({})

[

{ \_id: 1, name: 'A' },

{ \_id: 2, name: 'B' },

{ \_id: 3, name: 'C' },

{ \_id: 4, name: 'C' }

]

studentdb> db.emp.updateOne({\_id:1},{$set:{sal:10000.5}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5 },

{ \_id: 2, name: 'B' },

{ \_id: 3, name: 'C' },

{ \_id: 4, name: 'C' }

]

studentdb> db.emp.updateOne({\_id:2},{$set:{sal:20000.5}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.updateOne({\_id:3},{$set:{sal:30000.5}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.updateOne({\_id:4},{$set:{sal:40000.5}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5 },

{ \_id: 2, name: 'B', sal: 20000.5 },

{ \_id: 3, name: 'C', sal: 30000.5 },

{ \_id: 4, name: 'C', sal: 40000.5 }

]

studentdb> db.emp.updateMany({},{$set:{gender:"FEMALE"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 4,

modifiedCount: 4,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find({sal:{$gt:20000.5}})

[

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find({sal:{$lt:20000.5}})

[ { \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' } ]

studentdb> db.emp.find({sal:{$lte:20000.5}})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find({sal:{$lte:30000.5}})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find({sal:{$gte:30000.5}})

[

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find({sal:{$ne:30000.5}})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find().sort({sal:1})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find().sort({sal:~1})

MongoInvalidArgumentError: Invalid sort direction: -2

studentdb> db.emp.find().sort({sal:1})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find().sort({sal:-1})

[

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' }

]

studentdb> db.emp.find().sort({sal:-1}).limit(1)

[ { \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' } ]

studentdb> db.emp.find({},{name:1}).sort({sal:-1}).limit(1)

[ { \_id: 4, name: 'C' } ]

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'FEMALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'FEMALE' }

]

studentdb> db.emp.updateMany({sal:{$gt:20000.5}},{$set:{gender:"MALE"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 2,

modifiedCount: 2,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'FEMALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'MALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'MALE' }

]

studentdb> db.emp.updateMany({sal:{$gt:20000}},{$set:{gender:"MALE"}}) //used to update the genA

{

acknowledged: true,

insertedId: null,

matchedCount: 3,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'MALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'MALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'MALE' }

]

studentdb> db.emp.find({gender:{$in:["MALE","FEMALE"]}})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'MALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'MALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'MALE' }

]

studentdb> db.emp.find({gender:{$nin:["MALE"]}})

[ { \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' } ]

studentdb> db.emp.updateOne({\_id:1},{$set:{gender:"FEMALE"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 0,

upsertedCount: 0

}

studentdb> db.emp.renameCollection("employee")

{ ok: 1 }

studentdb> show collections

employee

student

studentdb> db.emp.find({$and:[{name:"C"},{sal:{$lt:40000.5}}]})

studentdb> db.emp.find({$and:[{name:"C"},{sal:{$gt:30000.5}}]})

studentdb> db.emp.renameCollection("emp")

MongoServerError: Can't rename a collection to itself

studentdb> db.employee.renameCollection("emp")

{ ok: 1 }

studentdb> show collections

emp

student

studentdb> db.emp.find({$and:[{name:"C"},{sal:{$gt:30000.5}}]})

[ { \_id: 4, name: 'C', sal: 40000.5, gender: 'MALE' } ]

studentdb> db.emp.find({$or:[{name:"C"},{sal:{$lt:30000.5}}]})

[

{ \_id: 1, name: 'A', sal: 10000.5, gender: 'FEMALE' },

{ \_id: 2, name: 'B', sal: 20000.5, gender: 'MALE' },

{ \_id: 3, name: 'C', sal: 30000.5, gender: 'MALE' },

{ \_id: 4, name: 'C', sal: 40000.5, gender: 'MALE' }

]

studentdb> db.emp.updateMany({},{$rename:{sal:"salary"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 4,

modifiedCount: 4,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb>

PART -3

Microsoft Windows [Version 10.0.22631.3085]

(c) Microsoft Corporation. All rights reserved.

C:\Users\adnan shariff>mongosh

Current Mongosh Log ID: 65cb00f86c88aad6265a1956

Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.1.1

Using MongoDB: 6.0.8

Using Mongosh: 2.1.1

mongosh 2.1.4 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------

The server generated these startup warnings when booting

2024-02-09T21:51:43.013+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted

------

test> show dbs

admin 40.00 KiB

config 84.00 KiB

local 120.00 KiB

studentdb 128.00 KiB

test> use studentdb

switched to db studentdb

studentdb> show collections

emp

student

studentdb> db.emp.find({})

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb> db.emp.find({}).sort({salary:-1}).limit(1)

[ { \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 } ]

studentdb> db.emp.find({name:1}).sort({salary:-1}).limit(1)

studentdb> db.emp.find({},{name:1,\_id:0}).sort({salary:-1}).limit(1)

[ { name: 'C' } ]

studentdb> //aggregate functions or multi-document functions

studentdb> db.emp.aggregate({$count:"totalcount"})

[ { totalcount: 4 } ]

studentdb> db.emp.find({})

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:null,totalsal:{$sum:"$salary"}}})

[ { \_id: null, totalsal: 100002 } ]

studentdb> db.emp.aggregate({$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}})

[

{ \_id: 'FEMALE', totalsal: 10000.5 },

{ \_id: 'MALE', totalsal: 90001.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:"$name",totalsal:{$sum:"$salary"}}})

[

{ \_id: 'A', totalsal: 10000.5 },

{ \_id: 'C', totalsal: 70001 },

{ \_id: 'B', totalsal: 20000.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:"$name",totalsal:{$sum:"$salary"}}}).sort()

MongoServerError: the $sort key specification must be an object

studentdb> db.emp.aggregate({$group:{\_id:"$gender",totalsal:{$avg:"$salary"}}})

[

{ \_id: 'MALE', totalsal: 30000.5 },

{ \_id: 'FEMALE', totalsal: 10000.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:null,totalsal:{$avg:"$salary"}}})

[ { \_id: null, totalsal: 25000.5 } ]

studentdb> db.emp.aggregate({$group:{\_id:null,avgsal:{$avg:"$salary"}}})

[ { \_id: null, avgsal: 25000.5 } ]

studentdb> db.emp.aggregate({$group:{\_id:null,avgsal:{$min:"$salary"}}})

[ { \_id: null, avgsal: 10000.5 } ]

studentdb> db.emp.aggregate({$group:{\_id:null,avgsal:{$max:"$salary"}}})

[ { \_id: null, avgsal: 40000.5 } ]

studentdb> db.emp.aggregate({$group:{\_id:null,maxsal:{$max:"$salary"}}})

[ { \_id: null, maxsal: 40000.5 } ]

studentdb> db.emp.aggregate({$group:{\_id:null,minsal:{$min:"$salary"}}})

[ { \_id: null, minsal: 10000.5 } ]

studentdb> db.emp.aggregate({$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}})

[

{ \_id: 'FEMALE', totalsal: 10000.5 },

{ \_id: 'MALE', totalsal: 90001.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:"$gender",avgsal:{$avg:"$salary"}}})

[

{ \_id: 'MALE', avgsal: 30000.5 },

{ \_id: 'FEMALE', avgsal: 10000.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:"$gender",minsal:{$min:"$salary"}}})

[

{ \_id: 'FEMALE', minsal: 10000.5 },

{ \_id: 'MALE', minsal: 20000.5 }

]

studentdb> db.emp.aggregate({$group:{\_id:"$gender",maxsal:{$max:"$salary"}}})

[

{ \_id: 'FEMALE', maxsal: 10000.5 },

{ \_id: 'MALE', maxsal: 40000.5 }

]

studentdb> db.emp.aggregate.({$match:{gender:"MALE"}})

Uncaught:

SyntaxError: Unexpected token (1:17)

> 1 | db.emp.aggregate.({$match:{gender:"MALE"}})

| ^

2 |

studentdb> db.emp.aggregate({$match:{gender:"MALE"}})

[

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb> db.emp.aggregate([{$match:{gender:"MALE"}},{$group:{\_id:null,totalsal:{$sum:"$salary"}}}])

[ { \_id: null, totalsal: 90001.5 } ]

studentdb> db.emp.aggregate([{$match:{gender:"MALE"}},{$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}}])

[ { \_id: 'MALE', totalsal: 90001.5 } ]

studentdb> db.emp.aggregate([{$match:{}},{$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}}])

[

{ \_id: 'FEMALE', totalsal: 10000.5 },

{ \_id: 'MALE', totalsal: 90001.5 }

]

studentdb> db.emp.aggregate([{$match:{}},{$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}},{$project:{\_id:0}}])

[ { totalsal: 90001.5 }, { totalsal: 10000.5 } ]

studentdb> db.emp.aggregate([{$match:{}},{$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}},{$project:{\_id:0}},{$sort:{totalsal:1}}])

[ { totalsal: 10000.5 }, { totalsal: 90001.5 } ]

studentdb> db.emp.aggregate([{$match:{}},{$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}},{$project:{\_id:0}},{$sort:{totalsal:-1}}])

[ { totalsal: 90001.5 }, { totalsal: 10000.5 } ]

studentdb> db.emp.aggregate([{$match:{}},{$group:{\_id:"$gender",totalsal:{$sum:"$salary"}}},{$project:{\_id:0}},{$sort:{totalsal:-1}},{$limit:1}])

[ { totalsal: 90001.5 } ]

studentdb> db.emp.find({})

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb> db.emp.updateMany({},{$set:{location:"VIZAG"}}) //used to add a key and it’s value

{

acknowledged: true,

insertedId: null,

matchedCount: 4,

modifiedCount: 4,

upsertedCount: 0

}

studentdb> db.emp.updateMany({},{$unset:{location:""}}) //used to remove keys in multiple collections

{

acknowledged: true,

insertedId: null,

matchedCount: 4,

modifiedCount: 4,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb> db.emp.find({salary:{$not:{$gt:20000}}})

[ { \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 } ]

studentdb> db.emp.aggregate([{$match:{}},{$out:"faculty"}]) //used to copy the content of one collection into another

studentdb> show collections

emp

faculty

student

studentdb> db.faculty.find()

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb>

PART-4

Microsoft Windows [Version 10.0.22631.3085]

(c) Microsoft Corporation. All rights reserved.

C:\Users\adnan shariff>mongosh

Current Mongosh Log ID: 65cc92399148b9bd8983b73c

Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.1.1

Using MongoDB: 6.0.8

Using Mongosh: 2.1.1

mongosh 2.1.4 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------

The server generated these startup warnings when booting

2024-02-13T22:06:36.233+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted

------

test> use studentdb

switched to db studentdb

studentdb> show collections

emp

faculty

student

studentdb> db.emp.find({})

[

{ \_id: 1, name: 'A', gender: 'FEMALE', salary: 10000.5 },

{ \_id: 2, name: 'B', gender: 'MALE', salary: 20000.5 },

{ \_id: 3, name: 'C', gender: 'MALE', salary: 30000.5 },

{ \_id: 4, name: 'C', gender: 'MALE', salary: 40000.5 }

]

studentdb> db.emp.updateMany({\_id:1,{$set:{location:"VIZAG"}}})

Uncaught:

SyntaxError: Unexpected token (1:25)

> 1 | db.emp.updateMany({\_id:1,{$set:{location:"VIZAG"}}})

| ^

2 |

studentdb> db.emp.updateMany({\_id:1},{$set:{location:"VIZAG"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.updateMany({\_id:2},{$set:{location:"GUNTUR"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.updateMany({\_id:3},{$set:{location:"ELURU"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.updateMany({\_id:4},{$set:{location:"SHIRIDI"}})

{

acknowledged: true,

insertedId: null,

matchedCount: 1,

modifiedCount: 1,

upsertedCount: 0

}

studentdb> db.emp.find()

[

{

\_id: 1,

name: 'A',

gender: 'FEMALE',

salary: 10000.5,

location: 'VIZAG'

},

{

\_id: 2,

name: 'B',

gender: 'MALE',

salary: 20000.5,

location: 'GUNTUR'

},

{

\_id: 3,

name: 'C',

gender: 'MALE',

salary: 30000.5,

location: 'ELURU'

},

{

\_id: 4,

name: 'C',

gender: 'MALE',

salary: 40000.5,

location: 'SHIRIDI'

}

]

studentdb> db.emp.find({},{location:1})

[

{ \_id: 1, location: 'VIZAG' },

{ \_id: 2, location: 'GUNTUR' },

{ \_id: 3, location: 'ELURU' },

{ \_id: 4, location: 'SHIRIDI' }

]

studentdb> db.emp.find({},{name:1,location:1})

[

{ \_id: 1, name: 'A', location: 'VIZAG' },

{ \_id: 2, name: 'B', location: 'GUNTUR' },

{ \_id: 3, name: 'C', location: 'ELURU' },

{ \_id: 4, name: 'C', location: 'SHIRIDI' }

]

studentdb> db.emp.find()

[

{

\_id: 1,

name: 'A',

gender: 'FEMALE',

salary: 10000.5,

location: 'VIZAG'

},

{

\_id: 2,

name: 'B',

gender: 'MALE',

salary: 20000.5,

location: 'GUNTUR'

},

{

\_id: 3,

name: 'C',

gender: 'MALE',

salary: 30000.5,

location: 'ELURU'

},

{

\_id: 4,

name: 'C',

gender: 'MALE',

salary: 40000.5,

location: 'SHIRIDI'

}

]

Patterns

studentdb> db.emp.find({location:/^V/})//starts with

[

{

\_id: 1,

name: 'A',

gender: 'FEMALE',

salary: 10000.5,

location: 'VIZAG'

}

]

studentdb> db.emp.find({},{name:1,location:1})

[

{ \_id: 1, name: 'A', location: 'VIZAG' },

{ \_id: 2, name: 'B', location: 'GUNTUR' },

{ \_id: 3, name: 'C', location: 'ELURU' },

{ \_id: 4, name: 'C', location: 'SHIRIDI' }

]

studentdb> db.emp.find({location:/^R/})//ends with r

studentdb> db.emp.find({location:/^R$/})//ends with r

studentdb> db.emp.find({location:/R$/})//ends with r

[

{

\_id: 2,

name: 'B',

gender: 'MALE',

salary: 20000.5,

location: 'GUNTUR'

}

]

studentdb> db.emp.find({location:/DI$/})//ends with r

[

{

\_id: 4,

name: 'C',

gender: 'MALE',

salary: 40000.5,

location: 'SHIRIDI'

}

]

studentdb> db.emp.find({},{name:1,location:1})

[

{ \_id: 1, name: 'A', location: 'VIZAG' },

{ \_id: 2, name: 'B', location: 'GUNTUR' },

{ \_id: 3, name: 'C', location: 'ELURU' },

{ \_id: 4, name: 'C', location: 'SHIRIDI' }

]

studentdb> db.emp.find({location:/UR/})

[

{

\_id: 2,

name: 'B',

gender: 'MALE',

salary: 20000.5,

location: 'GUNTUR'

},

{

\_id: 3,

name: 'C',

gender: 'MALE',

salary: 30000.5,

location: 'ELURU'

}

]

studentdb> db.emp.find({location:/V.\*G/})//star means zero or more charecters

[

{

\_id: 1,

name: 'A',

gender: 'FEMALE',

salary: 10000.5,

location: 'VIZAG'

}

]

studentdb> db.emp.find({location:/^G.\*R$/})

[

{

\_id: 2,

name: 'B',

gender: 'MALE',

salary: 20000.5,

location: 'GUNTUR'

}

]

studentdb> db.emp.find({}).forEach((e)=>{

... print(e)

... }

... ) // here we have used forEach loop to iterate through the documents.

{

\_id: 1,

name: 'A',

gender: 'FEMALE',

salary: 10000.5,

location: 'VIZAG'

}

{

\_id: 2,

name: 'B',

gender: 'MALE',

salary: 20000.5,

location: 'GUNTUR'

}

{

\_id: 3,

name: 'C',

gender: 'MALE',

salary: 30000.5,

location: 'ELURU'

}

{

\_id: 4,

name: 'C',

gender: 'MALE',

salary: 40000.5,

location: 'SHIRIDI'

}

studentdb> db.emp.find({}).forEach((e) => { print(e.\_id); } )

1

2

3

4

studentdb> db.emp.find({}).forEach((e) => { print(e.\_id+" "+e.name); } )

1 A

2 B

3 C

4 C

studentdb> db.emp.find({salary:{$gt:30000}}).forEach((e) => { print(e.\_id+" "+e.name); } )

3 C

4 C

studentdb> db.createCollection("shcolar")

{ ok: 1 }

studentdb> show collections

emp

faculty

shcolar

student

studentdb> db.scholar.insertOne({\_id:101,name:"Aravind"},skills:["AWS","MSWD"])

Uncaught:

SyntaxError: Unexpected token, expected "," (1:52)

> 1 | db.scholar.insertOne({\_id:101,name:"Aravind"},skills:["AWS","MSWD"])

| ^

2 |

studentdb> db.scholar.insertOne({\_id:101,name:"Aravind",skills:["AWS","MSWD"]})

{ acknowledged: true, insertedId: 101 }

studentdb> db.scholar.insertOne({\_id:102,name:"Kavya",skills:["MSWD"]})

{ acknowledged: true, insertedId: 102 }

studentdb> db.scholar.insertOne({\_id:103,name:"Minnie",skills:["JFSD"]})

{ acknowledged: true, insertedId: 103 }

studentdb> db.scholar.find()

[

{ \_id: 101, name: 'Aravind', skills: [ 'AWS', 'MSWD' ] },

{ \_id: 102, name: 'Kavya', skills: [ 'MSWD' ] },

{ \_id: 103, name: 'Minnie', skills: [ 'JFSD' ] }

]