Name : Sarthak Pagar

Roll No. : 34

Class : TE (IT)

Practical : 2

sarthak1594@UbuntuInWin:~$ mongosh

Current Mongosh Log ID: 670eb176c4b92e3beafe6910

Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.2

Using MongoDB: 8.0.0

Using Mongosh: 2.3.2

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

------

The server generated these startup warnings when booting

2024-10-15T23:10:40.921+05:30: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem

2024-10-15T23:10:45.375+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted

2024-10-15T23:10:45.375+05:30: For customers running the tcmalloc-google memory allocator, we suggest setting the contents of sysfsFile to 'always'

2024-10-15T23:10:45.375+05:30: For customers running the updated tcmalloc-google memory allocator, we suggest setting the contents of sysfsFile to 'defer+madvise'

2024-10-15T23:10:45.376+05:30: We suggest setting the contents of sysfsFile to 0.

2024-10-15T23:10:45.376+05:30: vm.max\_map\_count is too low

2024-10-15T23:10:45.376+05:30: We suggest setting swappiness to 0 or 1, as swapping can cause performance problems.

------

test> use Library

switched to db Library

Library> db.createCollection("Inventory")

{ ok: 1 }

Library> db.Inventory.insert([{Book\_id:10,Book\_name:"Harry Potter",Qty:30,Price:7000,Publication:"Bloomsbury"},{Book\_id:11,Book\_name:"War and Peace",Qty:20,Price:5000,Publication:"The Russian Messenger"},{Book\_id:12,Book\_name:"Epics Collection",Qty:10,Price:5000,Publication:"Bloomsbury"},{Book\_id:13,Book\_name:"It Wasn't Me",Qty:40,Price:2000,Publication:"Bloomsbury"},{Book\_id:14,Book\_name:1984,Qty:10,Price:6000,Publication:"Secker & Warburg"},{Book\_id:15,Book\_name:"The Lord of The Rings",Qty:40,Price:8000,Publication:"JRR"},{Book\_id:16,Book\_name:"Train to Pakistan",Qty:50,Price:4000,Publication:"India Penguin"},{Book\_id:17,Book\_name:"Mahabharat",Qty:100,Price:10000,Publication:"HarperCollins India"},{Book\_id:18,Book\_name:"The Hobbit",Qty:20,Price:2000,Publication:"JRR"},{Book\_id:19,Book\_name:"The Silmarillion",Qty:5,Price:1000,Publication:"JRR"}])

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

{

acknowledged: true,

insertedIds: {

'0': ObjectId('670eb27353bbf33226fe6911'),

'1': ObjectId('670eb27353bbf33226fe6912'),

'2': ObjectId('670eb27353bbf33226fe6913'),

'3': ObjectId('670eb27353bbf33226fe6914'),

'4': ObjectId('670eb27353bbf33226fe6915'),

'5': ObjectId('670eb27353bbf33226fe6916'),

'6': ObjectId('670eb27353bbf33226fe6917'),

'7': ObjectId('670eb27353bbf33226fe6918'),

'8': ObjectId('670eb27353bbf33226fe6919'),

'9': ObjectId('670eb27353bbf33226fe691a')

}

}

Library> db.Inventory.find()

[

{

\_id: ObjectId('670eb27353bbf33226fe6911'),

Book\_id: 10,

Book\_name: 'Harry Potter',

Qty: 30,

Price: 7000,

Publication: 'Bloomsbury'

},

{

\_id: ObjectId('670eb27353bbf33226fe6912'),

Book\_id: 11,

Book\_name: 'War and Peace',

Qty: 20,

Price: 5000,

Publication: 'The Russian Messenger'

},

{

\_id: ObjectId('670eb27353bbf33226fe6913'),

Book\_id: 12,

Book\_name: 'Epics Collection',

Qty: 10,

Price: 5000,

Publication: 'Bloomsbury'

},

{

\_id: ObjectId('670eb27353bbf33226fe6914'),

Book\_id: 13,

Book\_name: "It Wasn't Me",

Qty: 40,

Price: 2000,

Publication: 'Bloomsbury'

},

{

\_id: ObjectId('670eb27353bbf33226fe6915'),

Book\_id: 14,

Book\_name: 1984,

Qty: 10,

Price: 6000,

Publication: 'Secker & Warburg'

},

{

\_id: ObjectId('670eb27353bbf33226fe6916'),

Book\_id: 15,

Book\_name: 'The Lord of The Rings',

Qty: 40,

Price: 8000,

Publication: 'JRR'

},

{

\_id: ObjectId('670eb27353bbf33226fe6917'),

Book\_id: 16,

Book\_name: 'Train to Pakistan',

Qty: 50,

Price: 4000,

Publication: 'India Penguin'

},

{

\_id: ObjectId('670eb27353bbf33226fe6918'),

Book\_id: 17,

Book\_name: 'Mahabharat',

Qty: 100,

Price: 10000,

Publication: 'HarperCollins India'

},

{

\_id: ObjectId('670eb27353bbf33226fe6919'),

Book\_id: 18,

Book\_name: 'The Hobbit',

Qty: 20,

Price: 2000,

Publication: 'JRR'

},

{

\_id: ObjectId('670eb27353bbf33226fe691a'),

Book\_id: 19,

Book\_name: 'The Silmarillion',

Qty: 5,

Price: 1000,

Publication: 'JRR'

}

]

Library> var map=function(){emit(this.Publication,this.Qty)};

Library> var reduce=function(keyPublication,valueQty){return Array.sum(valueQty)};

Library> db.Inventory.mapReduce(map,reduce,{out:"Result"})

DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation instead.

See https://docs.mongodb.com/manual/core/map-reduce for details.

{ result: 'Result', ok: 1 }

Library> db.Result.find()

[

{ \_id: 'JRR', value: 65 },

{ \_id: 'India Penguin', value: 50 },

{ \_id: 'Secker & Warburg', value: 10 },

{ \_id: 'Bloomsbury', value: 80 },

{ \_id: 'The Russian Messenger', value: 20 },

{ \_id: 'HarperCollins India', value: 100 }

]

Library> db.Inventory.insert([{Book\_id:20,Book\_name:"Time Travel",Qty:10,Price:8000,Publication:"Bloomsbury"},{Book\_id:21,Book\_name:"Animal Farm",Qty:10,Price:7000,Publication:"Secker & Warburg"},{Book\_id:22,Book\_name:"CPD Library",Qty:10,Price:3000,Publication:"Bloomsbury"}])

{

acknowledged: true,

insertedIds: {

'0': ObjectId('670ebb8253bbf33226fe691b'),

'1': ObjectId('670ebb8253bbf33226fe691c'),

'2': ObjectId('670ebb8253bbf33226fe691d')

}

}

Library> db.Inventory.aggregate([{$match:{Qty:10}},{$group:{\_id:"$Publication",total:{$sum:"$Price"}}}])

[

{ \_id: 'Bloomsbury', total: 16000 },

{ \_id: 'Secker & Warburg', total: 13000 }

]

Library> db.Inventory.createIndex({"Book\_name":1})

Book\_name\_1

Library> db.Inventory.getIndexes()

[

{ v: 2, key: { \_id: 1 }, name: '\_id\_' },

{ v: 2, key: { Book\_name: 1 }, name: 'Book\_name\_1' }

]

Library> db.Inventory.dropIndex({"Book\_name":1})

{ nIndexesWas: 2, ok: 1 }

Library> db.Inventory.getIndexes()

[ { v: 2, key: { \_id: 1 }, name: '\_id\_' } ]

Library> db.Inventory.createIndex({Book\_name:1})

Book\_name\_1

Library> db.Inventory.getIndexes()

[

{ v: 2, key: { \_id: 1 }, name: '\_id\_' },

{ v: 2, key: { Book\_name: 1 }, name: 'Book\_name\_1' }

]

Library> db.Inventory.dropIndex({Book\_name:1})

{ nIndexesWas: 2, ok: 1 }

Library> db.Inventory.getIndexes()

[ { v: 2, key: { \_id: 1 }, name: '\_id\_' } ]

Library> db.Inventory.createIndex({Book\_id:-1,Qty:1})

Book\_id\_-1\_Qty\_1

Library> db.Inventory.getIndexes()

[

{ v: 2, key: { \_id: 1 }, name: '\_id\_' },

{ v: 2, key: { Book\_id: -1, Qty: 1 }, name: 'Book\_id\_-1\_Qty\_1' }

]

Library> db.Inventory.dropIndex({Book\_id:-1,Qty:1})

{ nIndexesWas: 2, ok: 1 }

Library> db.Inventory.getIndexes()

[ { v: 2, key: { \_id: 1 }, name: '\_id\_' } ]

Library> db.Inventory.createIndex({Book\_name:1})

Book\_name\_1

Library> db.Inventory.getIndexes()

[

{ v: 2, key: { \_id: 1 }, name: '\_id\_' },

{ v: 2, key: { Book\_name: 1 }, name: 'Book\_name\_1' }

]

Library> db.Inventory.ensureIndex({Book\_name:1})

[ 'Book\_name\_1' ]

Library> db.Inventory.explain()

Explainable(Library.Inventory)

Library> db.Inventory.explain("Book\_name")

Explainable(Library.Inventory)