1. Create a Mongodb Database named “Inventory”.

>>use Inventory



2. Create a collection named ‘Products’ and Insert the following documents.

>> db.createCollection("Productss")



db.Productss.insertMany([

{

"\_id": 1,

"name": "xPhone",

"price": 799,

"releaseDate": ISODate("2011-05-14"),

"spec": { "ram": 4, "screen": 6.5, "cpu": 2.66 },

"color": ["white", "black"],

"storage": [64, 128, 256]

},

{

"\_id": 2,

"name": "xTablet",

"price": 899,

"releaseDate": ISODate("2011-09-01"),

"spec": { "ram": 16, "screen": 9.5, "cpu": 3.66 },

"color": ["white", "black", "purple"],

"storage": [128, 256, 512]

},

{

"\_id": 3,

"name": "SmartTablet",

"price": 899,

"releaseDate": ISODate("2015-01-14"),

"spec": { "ram": 12, "screen": 9.7, "cpu": 3.66 },

"color": ["blue"],

"storage": [16, 64, 128]

},

{

"\_id": 4,

"name": "SmartPad",

"price": 699,

"releaseDate": ISODate("2020-05-14"),

"spec": { "ram": 8, "screen": 9.7, "cpu": 1.66 },

"color": ["white", "orange", "gold", "gray"],

"storage": [128, 256, 1024]

},

{

"\_id": 5,

"name": "SmartPhone",

"price": 599,

"releaseDate": ISODate("2022-09-14"),

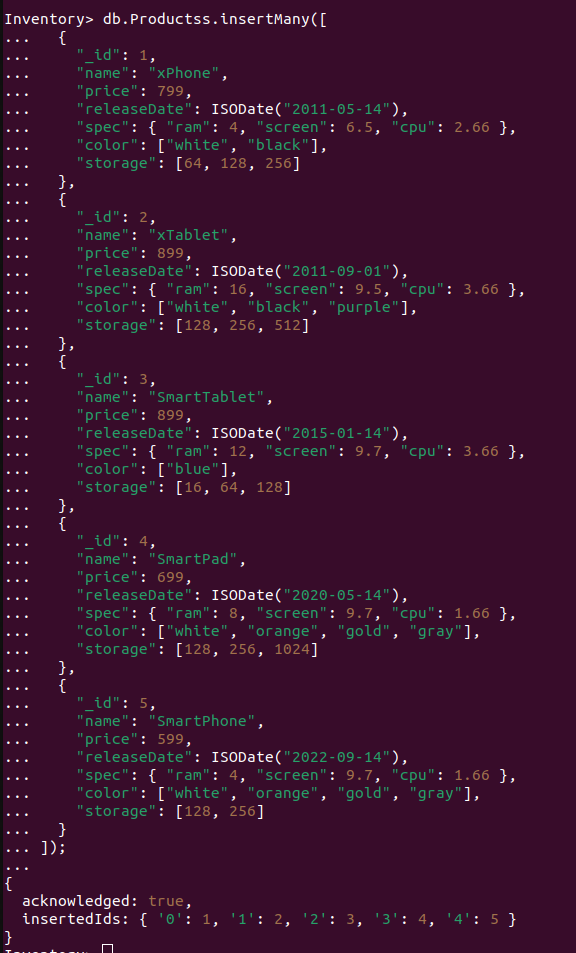
"spec": { "ram": 4, "screen": 9.7, "cpu": 1.66 },

"color": ["white", "orange", "gold", "gray"],

"storage": [128, 256]

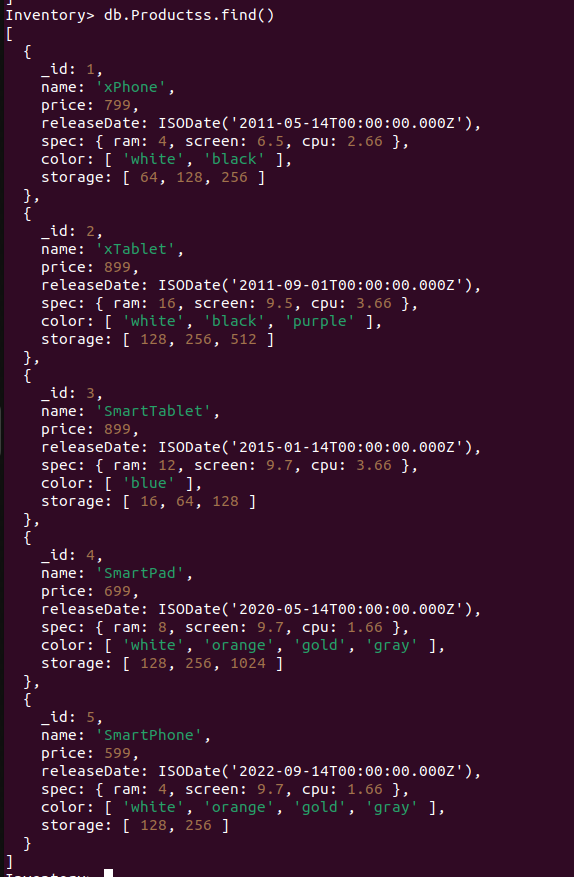
}

]);



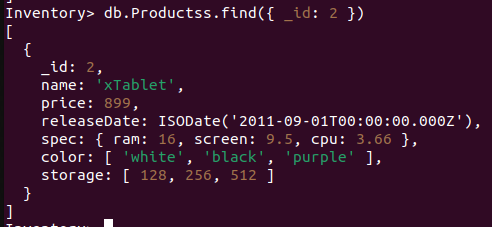
3. Display all documents in the collection product.

>>db.Productss.find()



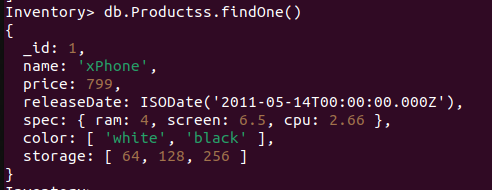
4. Display all the details of product with \_ id is 2.

>>db.Productss.find({ \_id: 2 })



5. Display the first document in the collection product.

>>db.Productss.findOne()



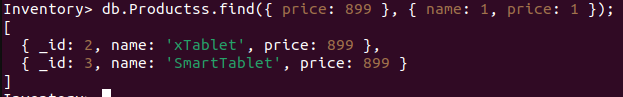
6. Display name and price of product with \_ id is 5.

>>db.Productss.find({ \_id: 5 }, { name: 1, price: 1 })



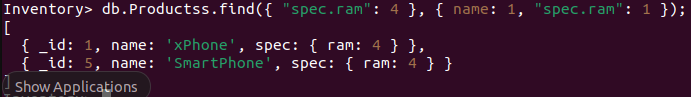
7. Select all documents where the price equals 899

>>db.Productss.find({ price: 899 }, { name: 1, price: 1 })



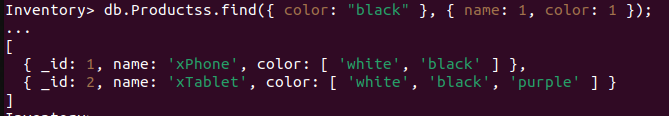
8. Find documents where the spec.ram equals 4

>>db.Productss.find({ "spec.ram": 4 }, { name: 1, "spec.ram": 1 })



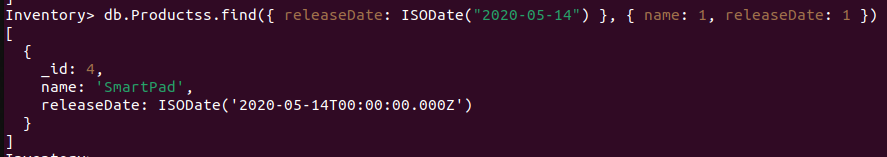
9. Find documents where the color array contains "black"

>>db.Productss.find({ color: "black" }, { name: 1, color: 1 })



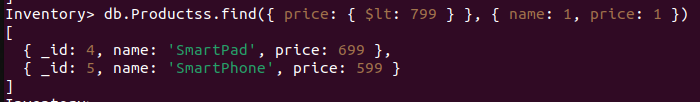
10. Find products with release date "2020-05-14"

>>db.Productss.find({ releaseDate: ISODate("2020-05-14") }, { name: 1, releaseDate: 1 })



11. Find products where price is less than 799

>>db.Productss.find({ price: { $lt: 799 } }, { name: 1, price: 1 })



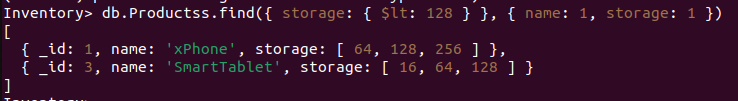
12. Find products where spec.screen is less than 7

>>db.Productss.find({ "spec.screen": { $lt: 7 } }, { name: 1, "spec.screen": 1 })



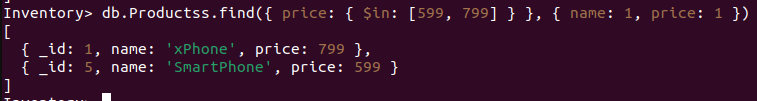
13. Find products where the storage array has at least one value less than 128

>>db.Productss.find({ storage: { $lt: 128 } }, { name: 1, storage: 1 })

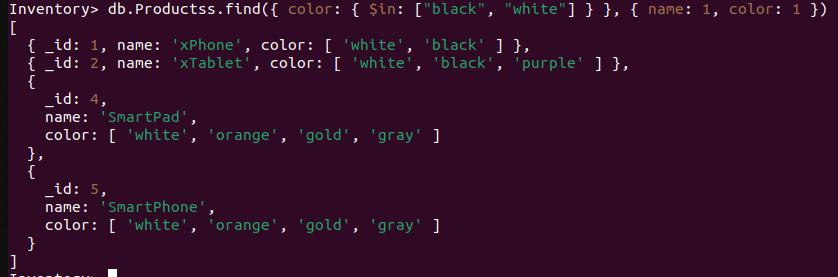


14. Display documents where price is either 599 or 799

>>db.Productss.find({ price: { $in: [599, 799] } }, { name: 1, price: 1 })

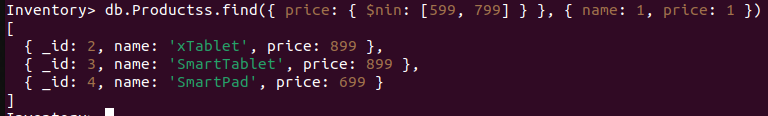


15. Display documents where color contains "black" or "white"

>>db.Productss.find({ color: { $in: ["black", "white"] } }, { name: 1, color: 1 })

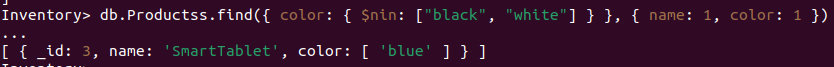
16. Display documents where price is neither 599 nor 799

>>db.Productss.find({ price: { $nin: [599, 799] } }, { name: 1, price: 1 })



17. Display documents where color does not contain "black" or "white"

>>db.Productss.find({ color: { $nin: ["black", "white"] } }, { name: 1, color: 1 })



18. Find products where price is 899 and color is either "black" or "white"

>>db.Productss.find({

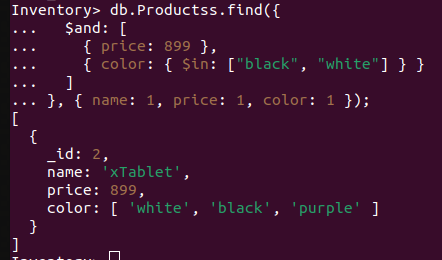
$and: [

{ price: 899 },

{ color: { $in: ["black", "white"] } }

]

}, { name: 1, price: 1, color: 1 })



19. Find products where price is less than 699 or greater than 799

>>db.Productss.find({

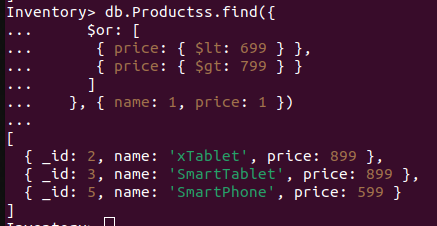
$or: [

{ price: { $lt: 699 } },

{ price: { $gt: 799 } }

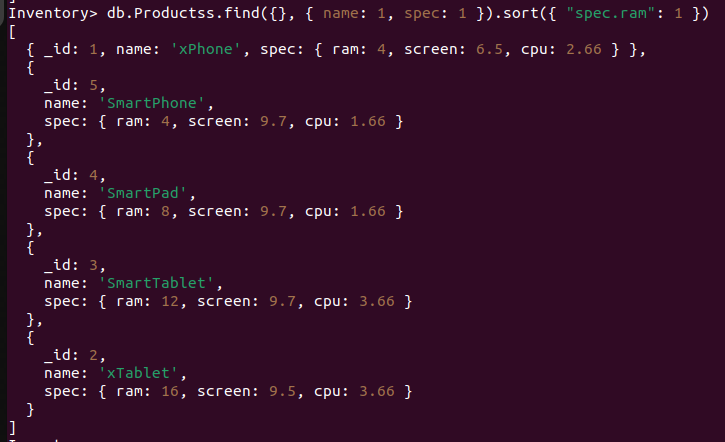
]

}, { name: 1, price: 1 })



20. Sort products by spec.ram in ascending order

>>db.Productss.find({}, { name: 1, spec: 1 }).sort({ "spec.ram": 1 })



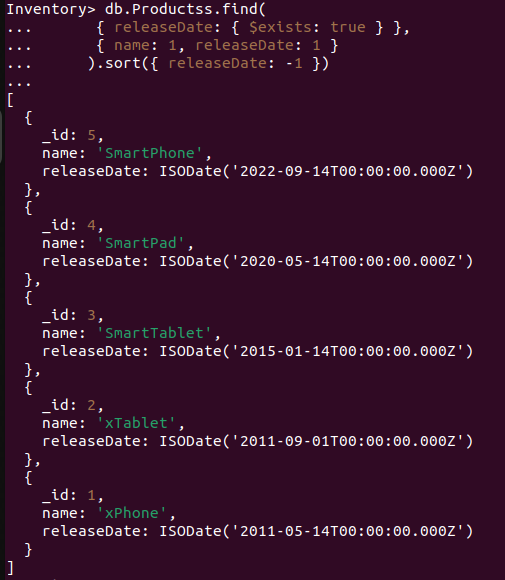
21. Sort products by releaseDate in descending order

>>db.Productss.find(

{ releaseDate: { $exists: true } },

{ name: 1, releaseDate: 1 }

).sort({ releaseDate: -1 })



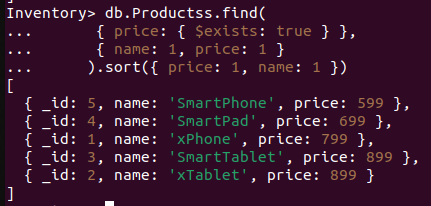
22. Sort by price and then name in ascending order

>>db.Productss.find(

{ price: { $exists: true } },

{ name: 1, price: 1 }

).sort({ price: 1, name: 1 })



23. Get the most expensive product

>>db.Productss.find({}, { name: 1, price: 1 }).sort({ price: -1, name: 1 }).limit(1)

