1. db.createCollection("employees", {

validator: {

$jsonSchema: {

bsonType: "object",

required: ["employeeId", "firstName", "lastName"],

properties: {

employeeId: {

bsonType: "int",

description: "must be an integer and is required"

},

firstName: {

bsonType: "string",

description: "must be a string and is required"

},

lastName: {

bsonType: "string",

description: "must be a string and is required"

},

email: {

bsonType: "string",

pattern: "^[^@\\s]+@[^@\\s]+\\.[^@\\s]+$",

description: "must be a valid email address"

},

salary: {

bsonType: "double",

minimum: 0,

description: "must be a double and greater than or equal to 0"

}

}

}

}

});

2. db.runCommand({

collMod: "employees",

validator : {

$jsonSchema: {

bsonType: 'object',

required: [

'employeeId',

'firstName',

'lastName'

],

properties: {

employeeId: {

bsonType: 'int',

description: 'must be an integer and is required'

},

firstName: {

bsonType: 'string',

description: 'must be a string and is required'

},

lastName: {

bsonType: 'string',

description: 'must be a string and is required'

},

email: {

bsonType: 'string',

pattern: '^[^@\\s]+@[^@\\s]+\\.[^@\\s]+$',

description: 'must be a valid email address'

},

salary: {

bsonType: 'double',

minimum: 7000,

description: 'must be a double and greater than or equal to 0'

}

}

}

}

})

>mongodump -d ECommerceDB -c Products –out Backup

>mongorestore

>mongoexport -d ECommerceDB -c Products --out ExportedData.json

>mongoimport -d ECommerceDB -c Product --file ExportedData.json

>db.Products.getIndexes()

>db.Products.createIndex({Name : 1})

>db.Products.find({ Price: { $gt: 100 } }).explain("executionStats")

>db.Products.dropIndex("Name\_1")

>db.Products.find({ Price: { $gt: 100 } })

>db.Products.find({ Price: { $lt: 50 } })

>db.Products.find({ Price: { $gte: 150 } })

>db.Products.find({ Price: { $lte: 75 } })

>db.Products.find({ Category: { $eq: "Electronics" } })

>db.Products.find({ Category: { $ne: "Clothing" } })

>db.Products.find({

$and: [

{ Price: { $gte: 100 } },

{ Category: "Electronics" }

]

})

>db.Products.find({

$or: [

{ Category: "Clothing" },

{ Price: { $gte: 300 } }

]

})

>db.Products.find({ Price: { $not: { $gte: 100 } } })

>db.Products.find({

$nor: [

{ Category: "Electronics" },

{ Price: { $gte: 500 } }

]

})  
  
>db.Products.find({ Discount: { $exists: true } })

>db.Products.find({ Discount: { $exists: false } })

>db.Products.find({ Price: { $type: "string" } })

>db.Products.find({ Tags: { $type: "array" } })

>db.Products.aggregate([

{ $match: { Price: { $gt: 100 } } }

])

>db.Products.aggregate([

{ $group: { \_id: "$Category", totalRevenue: { $sum: "$Price" } } }

])

>db.Products.aggregate([

{ $project: { \_id: 0, ProductName: 1, DiscountedPrice: { $multiply: ["$Price", 0.9] } } }

])

>db.Products.aggregate([

{ $sort: { Price: -1 } }

])

>db.Products.aggregate([

{ $limit: 3 }

])

>db.Products.aggregate([

{ $skip: 5 },

{ $limit: 3 }

])

>