MongoDB Assignment Solutions – E-Commerce Store

## PART A – Insert & Model Data

### A1. Insert 3 Products

db.products.insertMany([

{

name: "Laptop Pro 15",

brand: "DevByte",

price: 74999,

category: "Laptops",

available: true,

specs: {

cpu: "Intel i7",

ram: "16GB",

storage: "512GB SSD"

},

tags: ["high-performance", "work", "gaming"],

ratings: [

{ userId: "u001", rating: 5, review: "Super fast!" },

{ userId: "u002", rating: 4, review: "Great for devs" }

]

},

{

name: "Smartphone Ultra X",

brand: "MobiTech",

price: 39999,

category: "Mobiles",

available: true,

specs: {

storage: "256GB",

ram: "12GB",

camera: "108MP"

},

tags: ["camera-phone", "android"],

ratings: []

},

{

name: "Gaming Mouse Z1",

brand: "ClickZone",

price: 2499,

category: "Accessories",

available: false,

tags: ["gaming", "usb", "ergonomic"]

}

])

### A2. Insert 2 Users

db.users.insertMany([

{

userId: "u001",

name: "John Doe",

email: "john@example.com",

address: { city: "Chennai", pincode: "600001" },

wishlist: ["Laptop Pro 15", "Gaming Mouse Z1"],

cart: [ { productId: "Laptop Pro 15", quantity: 2 } ]

},

{

userId: "u002",

name: "Jane Smith",

email: "jane@example.com",

address: { city: "Delhi", pincode: "110001" },

wishlist: [],

cart: []

}

])

### A3. Insert Minimal Product

db.products.insertOne({

name: "Wireless Charger V2",

brand: "ChargeMate",

price: 1599,

category: "Accessories",

available: true

})

## PART B – Find Operations

### B1. specs.ram = "16GB"

db.products.find({ "specs.ram": "16GB" })

### B2. Tag = "gaming"

db.products.find({ tags: "gaming" })

### B3. Rating by userId = "u001"

db.products.find({ "ratings.userId": "u001" })

### B4. Products without tag "work"

db.products.find({ tags: { $ne: "work" } })

### B5. Users from Chennai with "Laptop Pro 15" in wishlist

db.users.find({

"address.city": "Chennai",

wishlist: "Laptop Pro 15"

})

### B6. Users with Laptop Pro 15 in cart

db.users.find({ "cart.productId": "Laptop Pro 15" })

### B7. Products where specs.camera is missing

db.products.find({ "specs.camera": { $exists: false } })

### B8. Products with more than 1 rating

db.products.find({ "ratings.1": { $exists: true } })

## PART C – Update Operations

### C1. Add new rating

db.products.updateOne(

{ name: "Laptop Pro 15" },

{

$push: {

ratings: { userId: "u003", rating: 4, review: "Solid performance." }

}

}

)

### C2. Add tag using $addToSet

db.products.updateOne(

{ name: "Laptop Pro 15" },

{ $addToSet: { tags: "editor-pick" } }

)

### C3. Remove "work" tag

db.products.updateOne(

{ name: "Laptop Pro 15" },

{ $pull: { tags: "work" } }

)

### C4. Update brand

db.products.updateMany(

{ brand: "DevByte" },

{ $set: { brand: "ByteForge" } }

)

### C5. Remove camera field

db.products.updateMany({}, { $unset: { "specs.camera": "" } })

### C6. Update pincode

db.users.updateOne(

{ userId: "u001" },

{ $set: { "address.pincode": "600002" } }

)

### C7. Add processor to mobile

db.products.updateOne(

{ category: "Mobiles" },

{ $set: { "specs.processor": "Snapdragon 8 Gen 2" } }

)

## PART D – Delete Operations

### D1. Delete one unavailable product

db.products.deleteOne({ available: false })

### D2. Remove "review" field from all ratings

db.products.updateMany(

{},

{ $unset: { "ratings.$[].review": "" } }

)

### D3. Remove a rating by userId

db.products.updateOne(

{ name: "Laptop Pro 15" },

{ $pull: { ratings: { userId: "u001" } } }

)

### D4. Remove from all wishlists

db.users.updateMany(

{},

{ $pull: { wishlist: "Laptop Pro 15" } }

)

### D5. Delete users with empty cart from Chennai

db.users.deleteMany({

"address.city": "Chennai",

cart: { $size: 0 }

})