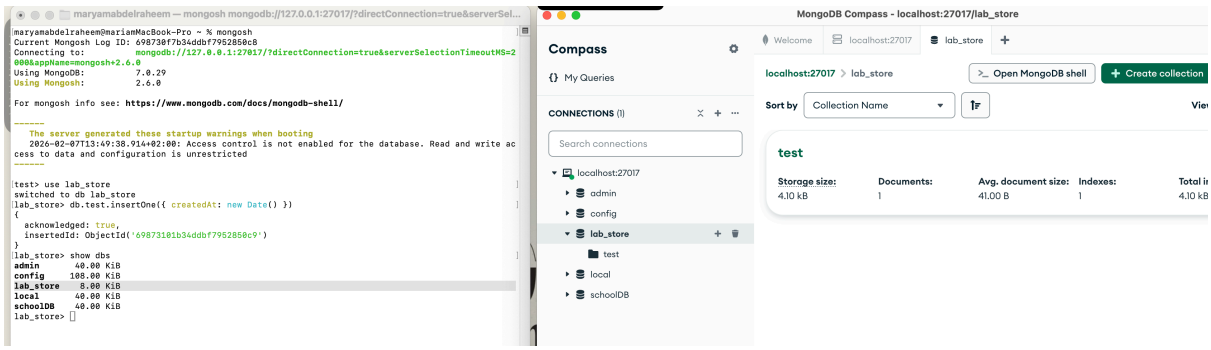
Saturday, 7 February 2026

Mongo lab1

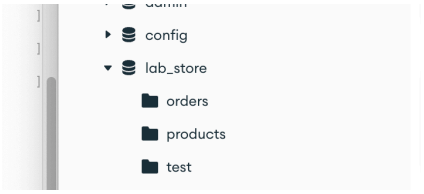
---

1. Create a database named lab_store and verify it appears in the database list.



2. Create a collections products and orders with no options and confirm it exists.

```
lab_store> db.createCollection("products")
{ ok: 1 }
lab_store> db.createCollection("orders")
{ ok: 1 }
lab_store> show collections
orders
products
test
lab_store>
```



3. Drop the orders collection

```
lab_store> db.orders.drop()
true
lab_store> show collections
products
test
lab_store>
```

4. Insert one product with fields:

{ name: "USB-C Cable", category: "Accessories", price: 149, inStock: true, tags: ["cable", "usb-c"], createdAt: <now> }. Verify that _id is auto-generated and of type ObjectId.

```
test> db.products.insertOne({
|   name: "USB-C Cable",
|   category: "Accessories",
|   price: 149,
|   inStock: true,
|   tags: ["cable", "usb-c"],
|   createdAt: new Date()
| })
{
  acknowledged: true,
  insertedId: ObjectId('698732eb30ec4eca29d3f68c')
}
test> db.products.findOne()
{
  _id: ObjectId('698732eb30ec4eca29d3f68c'),
  name: 'USB-C Cable',
  category: 'Accessories',
  price: 149,
  inStock: true,
  tags: [ 'cable', 'usb-c' ],
  createdAt: ISODate('2026-02-07T12:41:15.042Z')
}
test>
```

6. Insert a customer with your own email and isMember: false. Verify the type of createdAt (should be Date).

```
test> db.customers.insertOne({
|    email: "maryam@example.com",
|    isMember: false,
|    createdAt: new Date()
| })
{
  acknowledged: true,
  insertedId: ObjectId('698733ec30ec4eca29d3f68d')
}
test> db.customers.findOne()
{
  _id: ObjectId('698733ec30ec4eca29d3f68d'),
  email: 'maryam@example.com',
  isMember: false,
  createdAt: ISODate('2026-02-07T12:45:32.091Z')
}
test>
```

7. Insert one order that references the product and customer you created (use their _id values).
items should be an array of one item; include priceAtPurchase.

```
test> db.createCollection("orders")
{ ok: 1 }
test> const product = db.products.findOne({ name: "USB-C Cable" })
|

test> const customer = db.customers.findOne({ email: "maryam@example.com" })

test> print("Product ID:", product._id)
| print("Customer ID:", customer._id)
Product ID: ObjectId('698732eb30ec4eca29d3f68c')
Customer ID: ObjectId('698733ec30ec4eca29d3f68d')

test> db.orders.insertOne({
|   customerId: customer._id,
|   items: [
|     {
|       productId: product._id,
|       quantity: 1,
|       priceAtPurchase: product.price
|     }
|   ],
|   createdAt: new Date()
| })
{
  acknowledged: true,
  insertedId: ObjectId('698735c630ec4eca29d3f68e')
}
test> db.orders.findOne()
{
  _id: ObjectId('698735c630ec4eca29d3f68e'),
  customerId: ObjectId('698733ec30ec4eca29d3f68d'),
  items: [
    {
      productId: ObjectId('698732eb30ec4eca29d3f68c'),
      quantity: 1,
      priceAtPurchase: 149
    }
  ],
  createdAt: ISODate('2026-02-07T12:53:26.091Z')
}
test>
```

8. Insert multiple products (5–10 docs) spanning categories: Accessories, Laptops, Phones. Include varying price, inStock, tags.

```
test> db.products.insertMany([
|   {
|     name: "Wireless Mouse",
|     category: "Accessories",
|     price: 299,
|     inStock: true,
|     tags: ["mouse", "wireless"],
|     createdAt: new Date()
|   },
|   {
|     name: "Laptop Pro 14",
|     category: "Laptops",
|     price: 24000,
|     inStock: true,
|     tags: ["laptop", "pro"],
|     createdAt: new Date()
|   },
|   {
|     name: "Budget Laptop",
|     category: "Laptops",
|     price: 8500,
|     inStock: false,
|     tags: ["laptop", "budget"],
|     createdAt: new Date()
|   },
|   {
|     name: "Smartphone X",
|     category: "Phones",
|     price: 12000,
|     inStock: true,
|     tags: ["phone", "android"],
|     createdAt: new Date()
|   },
|   {
|     name: "Phone Charger",
|     category: "Accessories",
|     price: 199,
|     inStock: true,
|     tags: ["charger", "usb-c"],
|     createdAt: new Date()
|   },
|   {
|     name: "USB Hub",
|     category: "Accessories",
|     price: 399,
|     inStock: false,
|     tags: ["usb", "hub"],
|     createdAt: new Date()
|   },
|   {
|     name: "Gaming Laptop 16",
|     category: "Laptops",
|     price: 32000,
|     inStock: true,
|     tags: ["laptop", "gaming"],
|     createdAt: new Date()
|   }
| ])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('6987365bfa6ceff7e5ea9da6'),
    '1': ObjectId('6987365bfa6ceff7e5ea9da7'),
    '2': ObjectId('6987365bfa6ceff7e5ea9da8'),
    '3': ObjectId('6987365bfa6ceff7e5ea9da9'),
    '4': ObjectId('6987365bfa6ceff7e5ea9daa'),
    '5': ObjectId('6987365bfa6ceff7e5ea9dab'),
    '6': ObjectId('6987365bfa6ceff7e5ea9dac')
  }
}
test>
```

```
test> db.products.find().pretty()
[
  {
    _id: ObjectId('698732eb30ec4eca29d3f68c'),
    name: 'USB-C Cable',
    category: 'Accessories',
    price: 149,
    inStock: true,
    tags: [ 'cable', 'usb-c' ],
    createdAt: ISODate('2026-02-07T12:41:15.042Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9da6'),
    name: 'Wireless Mouse',
    category: 'Accessories',
    price: 299,
    inStock: true,
    tags: [ 'mouse', 'wireless' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9da7'),
    name: 'Laptop Pro 14',
    category: 'Laptops',
    price: 24000,
    inStock: true,
    tags: [ 'laptop', 'pro' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9da8'),
    name: 'Budget Laptop',
    category: 'Laptops',
    price: 8500,
    inStock: false,
    tags: [ 'laptop', 'budget' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9da9'),
    name: 'Smartphone X',
    category: 'Phones',
    price: 12000,
    inStock: true,
    tags: [ 'phone', 'android' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9daa'),
    name: 'Phone Charger',
    category: 'Accessories',
    price: 199,
    inStock: true,
    tags: [ 'charger', 'usb-c' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9dab'),
    name: 'USB Hub',
    category: 'Accessories',
    price: 399,
    inStock: false,
    tags: [ 'usb', 'hub' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9dac'),
    name: 'Gaming Laptop 16',
    category: 'Laptops',
```

9. Retrieve one product in the Accessories category. Print only name and price (exclude _id).

```
test> db.products.findOne(
|    { category: "Accessories" },
|    { _id: 0, name: 1, price: 1 }
| )
{ name: 'USB-C Cable', price: 149 }
test>
```

10. Find all products that are inStock: true. Return only name, price, inStock.

```
test> db.products.find
 | ({inStock:true},
| {_id:0, name:1 , price: 1, inStock: 1})
[
  { name: 'USB-C Cable', price: 149, inStock: true },
  { name: 'Wireless Mouse', price: 299, inStock: true },
  { name: 'Laptop Pro 14', price: 24000, inStock: true },
  { name: 'Smartphone X', price: 12000, inStock: true },
  { name: 'Phone Charger', price: 199, inStock: true },
  { name: 'Gaming Laptop 16', price: 32000, inStock: true }
]
test>
```

11. Products price between 100 and 1000 inclusive

```
test> db.products.find(
| {price: {$gte:100, $lte: 1000}})
[
  {
    _id: ObjectId('698732eb30ec4eca29d3f68c'),
    name: 'USB-C Cable',
    category: 'Accessories',
    price: 149,
    inStock: true,
    tags: [ 'cable', 'usb-c' ],
    createdAt: ISODate('2026-02-07T12:41:15.042Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9da6'),
    name: 'Wireless Mouse',
    category: 'Accessories',
    price: 299,
    inStock: true,
    tags: [ 'mouse', 'wireless' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9daa'),
    name: 'Phone Charger',
    category: 'Accessories',
    price: 199,
    inStock: true,
    tags: [ 'charger', 'usb-c' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  },
  {
    _id: ObjectId('6987365bfa6ceff7e5ea9dab'),
    name: 'USB Hub',
    category: 'Accessories',
    price: 399,
    inStock: false,
    tags: [ 'usb', 'hub' ],
    createdAt: ISODate('2026-02-07T12:55:55.580Z')
  }
]
test>
```

12. What is Cursor

12.1 Definition:
A cursor is an object returned by mongoDB the you perform a query using find().

12.2 why it exists:
To fetch documents in batches to save memory , and efficient for large datasets.

12.3 how it works:
You can iterate over it using .foreach(), .next(), or convert it to an array with .toArray()
+it supports chaining -> .sort(), .limit(), .skip()

12.4 Example

```
test> const cursor = db.products.find({ inStock: true })
|

test> cursor.forEach(doc => print(doc.name, "-", doc.price))
USB-C Cable - 149
Wireless Mouse - 299
Laptop Pro 14 - 24000
Smartphone X - 12000
Phone Charger - 199
Gaming Laptop 16 - 32000

test>
```