

DS108-03-07 - NoSQL - Lesson 3

Practice Hands-On

For your Lesson 3 Practice Hands-On, you will be working with your new knowledge on NoSQL. **This Hands-On will not be graded**, but we encourage you to complete it. The best way to become great at working with databases is to practice! Once you have submitted your project, you will be able to access the solution on the next page.

[!caution] To Submit Be sure to zip and submit your NoSQL-HandsOn3 text document when finished! You will not be able to re-submit, so be sure the screenshots to each part are located within this document.

Requirements

This Hands-On is structured into *two* parts, and each part may ask you to run multiple queries. After each query, please take a screenshot and add it to a text document (or an equivalent) and name this file NoSQL-HandsOn3. This way, you will be able to submit your answers to each part all at once. You will continue working with the `inventory` collection in Atlas that you created during the last lessons. Good luck!

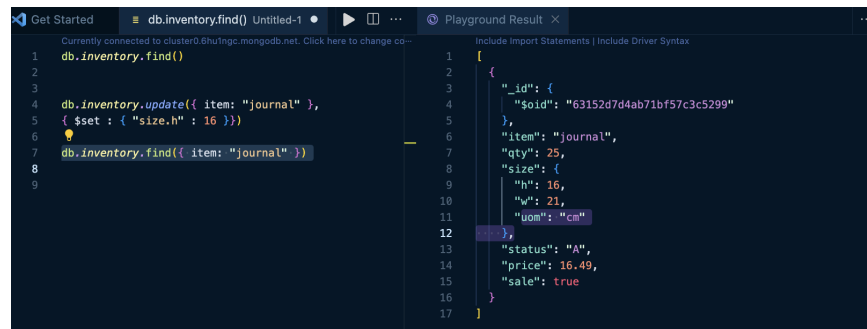
Part 1

Within your inventory collection, run queries to accomplish the following:

1.1 Update the item of journal to have a height of 16.

```
db.inventory.update({ item: "journal" },  
{ $set : { "size.h" : 16 } })
```

```
db.inventory.find({ item: "journal" })
```



The screenshot shows the MongoDB Playground interface. On the left, the code editor contains the following commands:

```
1 db.inventory.find()  
2  
3  
4 db.inventory.update({ item: "journal" },  
5 { $set : { "size.h" : 16 } })  
6  
7 db.inventory.find({ item: "journal" })  
8  
9
```

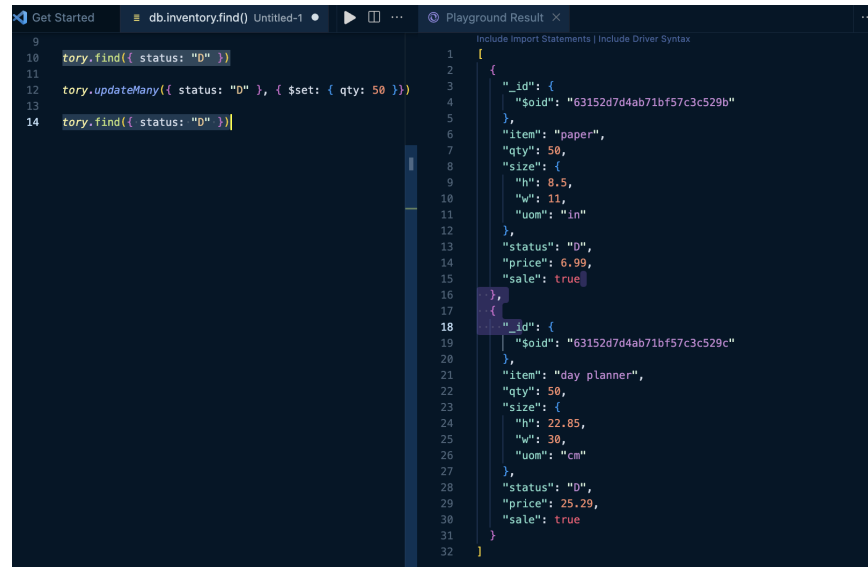
On the right, the 'Playground Result' pane shows the output of the find command, which is a JSON document representing an inventory item:

```
1 {  
2   "_id": {  
3     "$oid": "63152d7d4ab71bf57c3c5299"  
4   },  
5   "item": "journal",  
6   "qty": 25,  
7   "size": {  
8     "h": 16,  
9     "w": 21,  
10    "uom": "cm"  
11  },  
12  "status": "A",  
13  "price": 16.49,  
14  "sale": true  
15 }  
16  
17
```

1.2 Update all items with a status of D to have a quantity of 50.

```
db.inventory.updateMany({ status: "D" }, { $set: { qty: 50 } })
```

```
db.inventory.find({ status: "D" })
```



The screenshot shows a MongoDB Playground interface. The left pane contains a JavaScript snippet with three lines of code: `db.inventory.find({ status: "D" })`, `db.inventory.updateMany({ status: "D" }, { $set: { qty: 50 } })`, and `db.inventory.find({ status: "D" })`. The right pane, titled "Playground Result", displays the JSON output of the final find operation. It shows two documents: one for "paper" with a quantity of 50 and one for "day planner" with a quantity of 50. Both documents have a status of "D", a price, and a sale flag set to true.

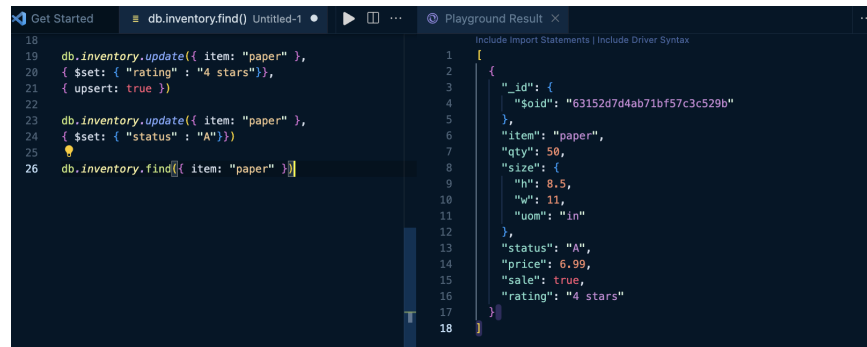
```
1  {
2    "_id": {
3      "$oid": "63152d7d4ab71bf57c3c529b"
4    },
5    "item": "paper",
6    "qty": 50,
7    "size": {
8      "h": 8.5,
9      "w": 11,
10     "uom": "in"
11   },
12   "status": "D",
13   "price": 6.99,
14   "sale": true
15 },
16 {
17   "_id": {
18     "$oid": "63152d7d4ab71bf57c3c529c"
19   },
20   "item": "day planner",
21   "qty": 50,
22   "size": {
23     "h": 22.85,
24     "w": 30,
25     "uom": "cm"
26   },
27   "status": "D",
28   "price": 25.29,
29   "sale": true
30 }
31 }
```

1.3 Update the item paper to include a field rating that has the value of 4 stars and change its status to A.

```
db.inventory.update({ item: "paper" },  
{ $set: { "rating" : "4 stars"}},  
{ upsert: true })
```

```
db.inventory.update({ item: "paper" },  
{ $set: { "status" : "A" }})
```

```
db.inventory.find({ item: "paper" })
```



The screenshot shows a MongoDB Playground interface with two panels. The left panel, titled 'db.inventory.find() Untitled-1', contains the following code:

```
18 db.inventory.update({ item: "paper" },  
19 { $set: { "rating" : "4 stars"}},  
20 { upsert: true })  
21  
22 db.inventory.update({ item: "paper" },  
23 { $set: { "status" : "A"}})  
24  
25  
26 db.inventory.find({ item: "paper" })
```

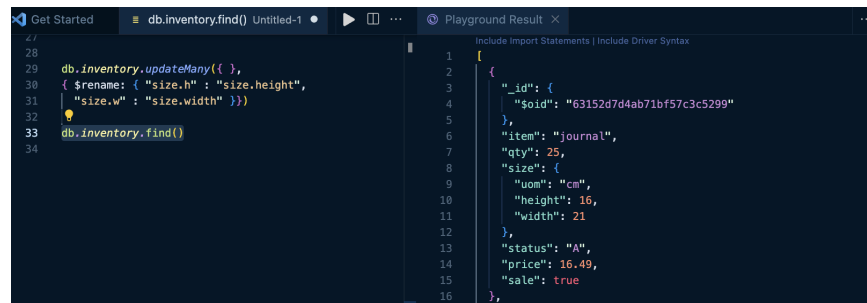
The right panel, titled 'Playground Result X', shows the result of the find command as a JSON document:

```
1 {  
2   "_id": {  
3     "$oid": "63152d7d4ab71bf57c3c529b"  
4   },  
5   "item": "paper",  
6   "qty": 50,  
7   "size": {  
8     "h": 8.5,  
9     "w": 11,  
10    "uom": "in"  
11  },  
12  "status": "A",  
13  "price": 6.99,  
14  "sale": true,  
15  "rating": "4 stars"  
16  }  
17  
18
```

1.4 Update all items to change the h and w fields to be height and width.

```
db.inventory.updateMany({ },  
{ $rename: { "size.h" : "size.height",  
            "size.w" : "size.width" } })
```

```
db.inventory.find()
```



The screenshot shows a MongoDB Playground interface. The left pane contains the following code:

```
28  
29 db.inventory.updateMany({ },  
30 { $rename: { "size.h" : "size.height",  
31 "size.w" : "size.width" } })  
32  
33 db.inventory.find()  
34
```

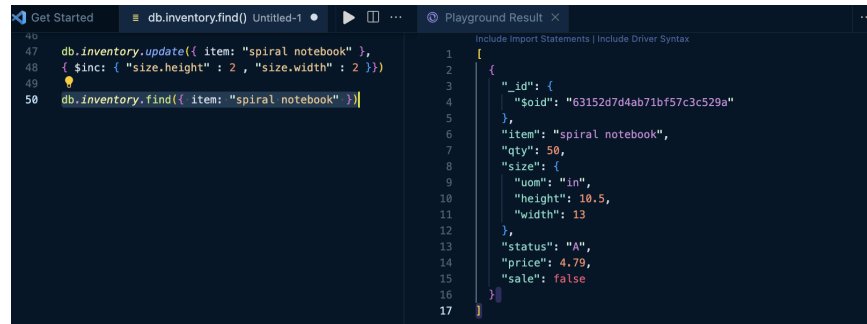
The right pane, titled "Playground Result", shows the output of the find operation as a JSON array with one document:

```
1  
2 {  
3   "_id": {  
4     "$oid": "63152d7d4ab71bf57c3c5299"  
5   },  
6   "item": "journal",  
7   "qty": 25,  
8   "size": {  
9     "uom": "cm",  
10    "height": 16,  
11    "width": 21  
12  },  
13  "status": "A",  
14  "price": 16.49,  
15  "sale": true  
16 },
```

1.5 Update the item spiral notebook so that the height and width is incremented by 2.

```
db.inventory.update({ item: "spiral notebook" },  
{ $inc: { "size.height" : 2 , "size.width" : 2 } })
```

```
db.inventory.find({ item: "spiral notebook" })
```



The screenshot shows a MongoDB Playground interface with two panels. The left panel, titled 'db.inventory.find() Untitled-1', contains the following code:

```
46  
47 db.inventory.update({ item: "spiral notebook" },  
48 { $inc: { "size.height" : 2 , "size.width" : 2 } })  
49  
50 db.inventory.find({ item: "spiral notebook" })
```

The right panel, titled 'Playground Result', shows the result of the find operation as a JSON document:

```
1 {  
2   "_id": {  
3     "$oid": "63152d7d4ab71bf57c3c529a"  
4   },  
5   "item": "spiral notebook",  
6   "qty": 50,  
7   "size": {  
8     "uom": "in",  
9     "height": 10.5,  
10    "width": 13  
11  },  
12  "status": "A",  
13  "price": 4.79,  
14  "sale": false  
15 }  
16  
17
```

1.6a Update the items paper to multiply the height by 3.

```
db.inventory.find({ item: "paper" })
```

```
db.inventory.update({ item: "paper" },  
{ $mul: { "size.height" : 3 } })
```

```
db.inventory.find({ item: "paper" })
```

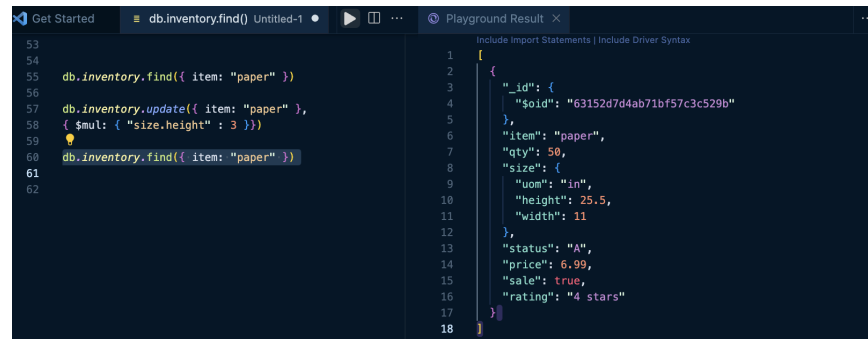


The screenshot shows a MongoDB playground interface. On the left, the code editor contains the following commands:

```
53  
54  
55 db.inventory.find({ item: "paper" })  
56  
57 db.inventory.update({ item: "paper" },  
58 { $mul: { "size.height" : 3 } })  
59  
60
```

On the right, the 'Playground Result' tab displays the JSON output of the first command, which is the document for the 'paper' item:

```
1  
2 {  
3   "_id": {  
4     "$oid": "63152d7d4ab71bf57c3c529b"  
5   },  
6   "item": "paper",  
7   "qty": 50,  
8   "size": {  
9     "uom": "in",  
10    "height": 8.5,  
11    "width": 11  
12  },  
13  "status": "A",  
14  "price": 6.99,  
15  "sale": true,  
16  "rating": "4 stars"  
17 }  
18
```



The screenshot shows the same MongoDB playground interface after the update command has been executed. The code editor now includes an additional command:

```
53  
54  
55 db.inventory.find({ item: "paper" })  
56  
57 db.inventory.update({ item: "paper" },  
58 { $mul: { "size.height" : 3 } })  
59  
60 db.inventory.find({ item: "paper" })  
61  
62
```

The 'Playground Result' tab shows the JSON output of the third command, which is the document for the 'paper' item after the update:

```
1  
2 {  
3   "_id": {  
4     "$oid": "63152d7d4ab71bf57c3c529b"  
5   },  
6   "item": "paper",  
7   "qty": 50,  
8   "size": {  
9     "uom": "in",  
10    "height": 25.5,  
11    "width": 11  
12  },  
13  "status": "A",  
14  "price": 6.99,  
15  "sale": true,  
16  "rating": "4 stars"  
17 }  
18
```

1.6b Update the item day planner to multiply the height by 3.

```
db.inventory.find({ item: "day planner" })
```

```
db.inventory.update({ item: "day planner" },  
{ $mul: { "size.height" : 3 }})
```

```
db.inventory.find({ item: "day planner" })
```

The screenshot shows a MongoDB playground interface. On the left, the code editor contains the following commands:

```
53  
54  
55 db.inventory.find({ item: "paper" })  
56  
57 db.inventory.update({ item: "paper" },  
58 { $mul: { "size.height" : 3 }})  
59  
60 db.inventory.find({ item: "paper" })  
61  
62  
63  
64  
65 db.inventory.find({ item: "day planner" })  
66  
67 |
```

On the right, the 'Playground Result' tab shows the JSON output of the last command:

```
1  
2 {  
3   "_id": {  
4     "$oid": "63152d7d4ab71bf57c3c529c"  
5   },  
6   "item": "day planner",  
7   "qty": 50,  
8   "size": {  
9     "uom": "cm",  
10    "height": 22.85,  
11    "width": 30  
12  },  
13  "status": "D",  
14  "price": 25.29,  
15  "sale": true  
16 }  
17
```

The screenshot shows the same MongoDB playground interface after the update command. The code editor now has:

```
64  
65 db.inventory.find({ item: "day planner" })  
66  
67 db.inventory.update({ item: "day planner" },  
68 { $mul: { "size.height" : 3 }})  
69  
70 db.inventory.find({ item: "day planner" })
```

The 'Playground Result' tab shows the updated JSON output:

```
1  
2 {  
3   "_id": {  
4     "$oid": "63152d7d4ab71bf57c3c529c"  
5   },  
6   "item": "day planner",  
7   "qty": 50,  
8   "size": {  
9     "uom": "cm",  
10    "height": 68.55000000000001,  
11    "width": 30  
12  },  
13  "status": "D",  
14  "price": 25.29,  
15  "sale": true  
16 }  
17
```


Part 2

For the second part, you will be working through a word problem and you will need a bit more information. Please run the following query to add documents into your inventory collection then complete the below requirements:

```
db.inventory.insertMany([
  {
    item: 'sticky note pads',
    size: { height: 8.9, width: 9, uom: 'cm' },
    status: 'B',
    quantity: 5
  },
  {
    item: 'pens',
    size: { height: 12, width: 1.3, uom: 'cm' },
    status: 'A',
    quantity: 4
  },
  {
    item: 'pencils',
    size: { height: 13, width: 1.4, uom: 'cm' },
    status: 'D',
    quantity: 10
  },
  {
    item: 'clipboard',
    size: { height: 13, width: 7, uom: 'in' },
    status: 'B',
    quantity: 2
  },
  {
    item: 'printer ink',
    size: { height: 2, width: 3, uom: 'in' },
    status: 'C',
    quantity: 2
  }
]);
```

The company you are working for wants to update their inventory database. Currently, there are ten pieces of inventory in the database.

2.1 Update Inventory Rating

They want to give each product a rating of 4 stars to start, as well as a field that shows if it has been rated yet using a boolean. They would like this to be reflected in the database as rating with two fields: `numberOfStars`, and `hasBeenRated`.

```
db.inventory.find()
```

```
db.inventory.update({ item: "paper" },  
{ $unset: { "rating" : "" } })
```

```
db.inventory.find({ item: "paper" })
```

```
db.inventory.updateMany({ },  
{ $set: { "rating.numberOfStars" : "4 stars" ,  
"rating.hasBeenRated" : true }},  
{ upsert: true })
```

```
db.inventory.find()
```

The screenshot shows a MongoDB Playground interface with a dark theme. On the left, a list of commands is executed: `db.inventory.find()`, `db.inventory.update({ item: "paper" }, { $unset: { "rating" : "" } })`, `db.inventory.find({ item: "paper" })`, `db.inventory.updateMany({ }, { $set: { "rating.numberOfStars" : "4 stars" , "rating.hasBeenRated" : true }}, { upsert: true })`, and finally `db.inventory.find()` with the cursor index set to 0. On the right, the resulting JSON document is displayed, showing a product with `item: "journal"`, `rating.numberOfStars: "4 stars"`, and `rating.hasBeenRated: true`.

```
115 db.inventory.find()  
116  
117 db.inventory.update({ item: "paper" },  
118 { $unset: { "rating" : "" } })  
119  
120 db.inventory.find({ item: "paper" })  
121  
122  
123 db.inventory.updateMany({ },  
124 { $set: { "rating.numberOfStars" : "4 stars" ,  
125 "rating.hasBeenRated" : true }},  
126 { upsert: true })  
127  
128  
129 db.inventory.find()
```

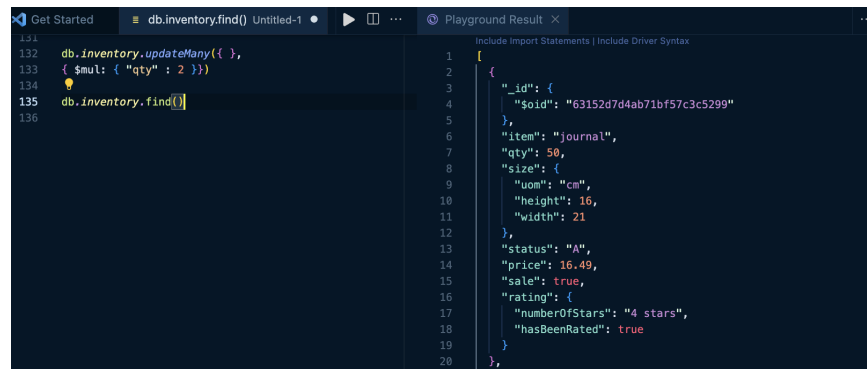
```
{  
  "_id": {  
    "$oid": "63152d7d4ab71bf57c3c5299"  
  },  
  "item": "journal",  
  "qty": 25,  
  "size": {  
    "uom": "cm",  
    "height": 16,  
    "width": 21  
  },  
  "status": "A",  
  "price": 16.49,  
  "sale": true,  
  "rating": {  
    "numberOfStars": "4 stars",  
    "hasBeenRated": true  
  }  
}
```

2.2 Double Inventory Quantity

Also, they want to double the quantity of every product since the company is rapidly growing.

```
db.inventory.updateMany({ },  
{ $mul: { "qty" : 2 } })
```

```
db.inventory.find()
```



The screenshot shows a MongoDB Playground interface. The left pane contains the following code:

```
131  
132 db.inventory.updateMany({ },  
133 { $mul: { "qty" : 2 } })  
134  
135 db.inventory.find()  
136
```

The right pane, titled "Playground Result", shows the output of the find query as a JSON document:

```
1 {  
2   "_id": {  
3     "$oid": "63152d7d4ab71bf57c3c5299"  
4   },  
5   "item": "journal",  
6   "qty": 50,  
7   "size": {  
8     "uom": "cm",  
9     "height": 16,  
10    "width": 21  
11  },  
12  "status": "A",  
13  "price": 16.49,  
14  "sale": true,  
15  "rating": {  
16    "numberOfStars": "4 stars",  
17    "hasBeenRated": true  
18  }  
19 }  
20
```

2.3 Update Inventory Status

Lastly, they would like you to find all products that are sized using centimeters and give them a status of “B”.

```
db.inventory.find({ "size.uom" : "cm" })
```

```
db.inventory.updateMany({ "size.uom" : "cm" },  
{ $set: { "status" : "B" } })
```

```
db.inventory.find()
```

The screenshot shows the MongoDB Playground interface. On the left, the code editor contains the following code:

```
137  
138 db.inventory.find({ "size.uom" : "cm" })  
139  
140
```

On the right, the 'Playground Result' tab is active, displaying a single document from the 'inventory' collection. The document is a JSON object with the following fields:

```
{  
  "_id": {  
    "$oid": "63152d7d4ab71bf57c3c5299"  
  },  
  "item": "journal",  
  "qty": 50,  
  "size": {  
    "uom": "cm",  
    "height": 16,  
    "width": 21  
  },  
  "status": "A",  
  "price": 16.49,  
  "sale": true,  
  "rating": {  
    "numberOfStars": "4 stars",  
    "hasBeenRated": true  
  }  
}
```

The screenshot shows the MongoDB Playground interface after running an update operation. The code editor on the left now contains:

```
137  
138 db.inventory.find({ "size.uom" : "cm" })  
139  
140 db.inventory.updateMany({ "size.uom" : "cm" },  
141 { $set: { "status" : "B" } })  
142  
143 db.inventory.find()
```

The 'Playground Result' tab on the right shows the same document as before, but with the 'status' field updated to 'B':

```
{  
  "_id": {  
    "$oid": "63152d7d4ab71bf57c3c5299"  
  },  
  "item": "journal",  
  "qty": 50,  
  "size": {  
    "uom": "cm",  
    "height": 16,  
    "width": 21  
  },  
  "status": "B",  
  "price": 16.49,  
  "sale": true,  
  "rating": {  
    "numberOfStars": "4 stars",  
    "hasBeenRated": true  
  }  
}
```