

Find only the product name and product material

The screenshot shows the Humongous.io MongoDB playground interface. The query entered is `db.collection.find({}, {product_name:1 , product_material:1 , _id:0 })`. The results are displayed in a JSON array format, showing product names and materials for various items like chips, sausages, a car, pants, a chair, towels, shoes, a hat, and a ball. The status bar at the bottom indicates "25 results in 3 ms".

```
db.collection.find({}, {product_name:1 , product_material:1 , _id:0 })
```

```
[
  {
    product_name: "Intelligent Fresh Chips",
    product_material: "Concrete",
  },
  {
    product_name: "Practical Fresh Sausages",
    product_material: "Cotton",
  },
  {
    product_name: "Refined Steel Car",
    product_material: "Rubber",
  },
  {
    product_name: "Gorgeous Plastic Pants",
    product_material: "Soft",
  },
  {
    product_name: "Sleek Cotton Chair",
    product_material: "Fresh",
  },
  {
    product_name: "Awesome Wooden Towels",
    product_material: "Plastic",
  },
  {
    product_name: "Practical Soft Shoes",
    product_material: "Rubber",
  },
  {
    product_name: "Incredible Steel Hat",
    product_material: "Rubber",
  },
  {
    product_name: "Awesome Wooden Ball",
    product_material: "Soft",
  }
]
```

25 results in 3 ms

List the four product which are greater than 500 in price

The screenshot shows the Humongous.io MongoDB playground interface. The query entered is `db.collection.find({product_price: {$gt:500}});`. The results are displayed in a JSON array format, showing four products with prices greater than 500: Intelligent Fresh Chips (price 655), Practical Fresh Sausages (price 911), Refined Steel Car (price 690), and Practical Soft Shoes (price 911). The status bar at the bottom indicates "4 results in 3 ms".

```
db.collection.find({product_price: {$gt:500}});
```

```
[
  {
    _id: "6504485d2606656f207e43b1",
    id: "1",
    product_name: "Intelligent Fresh Chips",
    product_price: 655,
    product_material: "Concrete",
    product_color: "Mint green",
  },
  {
    _id: "6504485d2606656f207e43b2",
    id: "2",
    product_name: "Practical Fresh Sausages",
    product_price: 911,
    product_material: "Cotton",
    product_color: "Indigo",
  },
  {
    _id: "6504485d2606656f207e43b3",
    id: "3",
    product_name: "Refined Steel Car",
    product_price: 690,
    product_material: "Rubber",
    product_color: "Gold",
  },
  {
    _id: "6504485d2606656f207e43b4",
    id: "4",
    product_name: "Practical Soft Shoes",
    product_price: 911,
    product_material: "Rubber",
    product_color: "Indigo",
  }
]
```

4 results in 3 ms

Find the product price which are between 400 to 800

The screenshot shows the Humongous.io MongoDB playground interface. The query entered is `db.collection.find({product_price:{$gte: 400 , $lte : 600}});`. The results show three documents:

- `{_id: '65044a51260656f207e4b03', id: '4', product_name: 'Gorgeous Plastic Pants', product_price: 492, product_material: 'Soft', product_color: 'plum', ...}`
- `{_id: '65044a51260656f207e4b05', id: '6', product_name: 'Awesome Wooden Towels', product_price: 474, product_material: 'Plastic', product_color: 'orange', ...}`
- `{_id: '65044a51260656f207e4b04', id: '7', product_name: 'Practical Soft Shoes', product_price: 500, product_material: 'Rubber', product_color: 'pink', ...}`

At the bottom, it indicates "3 results in 3 ms".

Find the product price which are not between 400 to 600

The screenshot shows the Humongous.io MongoDB playground interface. The query entered is `db.collection.find({ $or: [ { product_price : { $lte : 400 } }, { product_price : { $gte : 600 } } ] });`. The results show five documents:

- `{_id: '65044ae260656f207e5170', id: '1', product_name: 'Intelligent Fresh Chips', product_price: 450, product_material: 'Concrete', product_color: 'mint green', ...}`
- `{_id: '65044ae260656f207e5171', id: '2', product_name: 'Practical Fresh Sausages', product_price: 511, product_material: 'Cotton', product_color: 'indigo', ...}`
- `{_id: '65044ae260656f207e5172', id: '3', product_name: 'Refined Steel Car', product_price: 609, product_material: 'Rubber', product_color: 'gold', ...}`
- `{_id: '65044ae260656f207e5174', id: '5', product_name: 'Sleek Cotton Chair', product_price: 33, product_material: 'Fresh', product_color: 'black', ...}`
- `{_id: '65044ae260656f207e5177', id: '8', product_name: 'Incredible Steel Hat', product_price: 71, product_material: 'Rubber', ...}`

At the bottom, it indicates "5 results in 3 ms".

## Find all the information about each products

The screenshot shows the Humongous.io MongoDB playground interface. The query entered is `db.collection.find()`. The results are displayed in a JSON array format, showing five product documents. Each document contains fields for `_id`, `product_name`, `product_price`, `product_material`, and `product_color`.

```
1 db.collection.find()
```

```
[
  {
    "_id": "6504436269656220a2d6a0",
    "id": "1",
    "product_name": "Intelligent Fresh Cheese",
    "product_price": 875,
    "product_material": "Concrete",
    "product_color": "Black green",
  },
  {
    "_id": "6504436269656220a2d6a1",
    "id": "2",
    "product_name": "Practical Fresh Sausages",
    "product_price": 711,
    "product_material": "Cotton",
    "product_color": "Indigo",
  },
  {
    "_id": "6504436269656220a2d6a2",
    "id": "3",
    "product_name": "Refined Steel Cpu",
    "product_price": 691,
    "product_material": "Rubber",
    "product_color": "Gold",
  },
  {
    "_id": "6504436269656220a2d6a3",
    "id": "4",
    "product_name": "Organic Plastic Pants",
    "product_price": 492,
    "product_material": "Soft",
    "product_color": "Plum",
  },
  {
    "_id": "6504436269656220a2d6a4",
    "id": "5",
    "product_name": "Sleek Cotton Chair",
    "product_price": 37,
    "product_material": "Rubber",
  }
]
```

12 results in 2 ms

## Delete the products which product price value are same

The screenshot shows the Humongous.io MongoDB playground interface. The query entered is an aggregation pipeline to find duplicate product prices and then delete them. The pipeline uses `$group` to count occurrences of each price, `$match` to filter for prices that appear more than once, and `$deleteMany` to remove the documents with those duplicate prices.

```
1 db.collection.aggregate([
2   {
3     $group: {
4       _id: "$product_price",
5       counts: { $sum: 1 },
6       duplicates: { $push: "$_id" }
7     }
8   },
9   {
10    $match: {
11      counts: { $gt: 1 }
12    }
13  }
14 ])
15
16 // Once you have the duplicatesToDelete array, you can use it in the deleteMany operation.
17 db.collection.deleteMany({ _id: { $in: duplicatesToDelete } })
18
```

```
[
  {
    "_id": 47,
    "counts": 2,
    "duplicates": [
      "6504436269656220a2d6a3",
      "6504436269656220a2d6a2"
    ]
  },
  {
    "_id": 34,
    "counts": 2,
    "duplicates": [
      "6504436269656220a2d6a4",
      "6504436269656220a2d6a0"
    ]
  }
]
```

12 results in 2 ms

Find the product with a row id of 10

The screenshot shows the Humongous.io MongoDB playground interface. The query editor on the left contains the following query:

```
1 db.collection.find( {id:"10"})
```

The results pane on the right displays a single document:

```
{
  "_id": "650445287e47696c78118334",
  "id": "10",
  "product_name": "Awesome Wooden Pizza",
  "product_price": 91,
  "product_material": "Tronox",
  "product_color": "Indigo",
}
```

The status bar at the bottom indicates "1 result in 2 ms".

Find all products which contain the value of soft in product material

The screenshot shows the Humongous.io MongoDB playground interface. The query editor on the left contains the following query:

```
1 db.collection.find( {product_material:"Soft"} )
```

The results pane on the right displays four documents:

```
{
  "_id": "650445287e47696c78118334",
  "id": "4",
  "product_name": "Gorgeous Plastic Pants",
  "product_price": 692,
  "product_material": "Soft",
  "product_color": "plum",
},
{
  "_id": "650445287e47696c78118334",
  "id": "9",
  "product_name": "Awesome Wooden Ball",
  "product_price": 29,
  "product_material": "Soft",
  "product_color": "azure",
},
{
  "_id": "650445287e47696c78118335",
  "id": "11",
  "product_name": "Unbranded Wooden Cheese",
  "product_price": 19,
  "product_material": "Soft",
  "product_color": "black",
},
{
  "_id": "650445287e47696c78118335",
  "id": "19",
  "product_name": "Intelligent Cotton Chips",
  "product_price": 69,
  "product_material": "Soft",
  "product_color": "azure",
},
}
```

The status bar at the bottom indicates "4 results in 3 ms".

Find products which contain product color indigo and product price 492.00

The screenshot shows the Humongous.io MongoDB playground interface. The browser's address bar displays the URL `humongous.io/app/playground/mongod/new`. The page header includes the Humongous.io logo, a 'Sign in' button, and a 'Get started' button. Below the header, there are 'Run' and 'Share' buttons on the left, and a 'Format Query' button on the right. The main area is divided into two panels. The left panel, titled 'Query', contains a MongoDB query: 

```
1 db.collection.find({
2   $or: [
3     { product_color: 'indigo' },
4     { product_price: 492.00 }
5   ]
6 });
7
```

 The right panel, titled 'Results', displays the query results in a JSON array format. The results include four documents: 1. A document with `product_name: 'Practical Fresh Sausages'`, `product_price: 911`, `product_material: 'Cotton'`, and `product_color: 'indigo'`. 2. A document with `product_name: 'Gorgeous Plastic Pants'`, `product_price: 492`, `product_material: 'Soft'`, and `product_color: 'plum'`. 3. A document with `product_name: 'Generic Wooden Pizza'`, `product_price: 89`, `product_material: 'Frozen'`, and `product_color: 'indigo'`. 4. A document with `product_name: 'Incredible Metal Car'`, `product_price: 36`, `product_material: 'Fresh'`, and `product_color: 'indigo'`. At the bottom left, a status bar indicates '14 results in 3 ms'. At the bottom right, there are links for 'MongoDB GUI' and 'Blog'.

Humongous.io

Sign in Get started

Run Share Format Query

Query

```
1 db.collection.find({
2   $or: [
3     { product_color: 'indigo' },
4     { product_price: 492.00 }
5   ]
6 });
7
```

Database content Results Examples

```
{
  "_id": "650447e57e47696c78119265",
  "id": "2",
  "product_name":
    "Practical Fresh Sausages",
  "product_price": 911,
  "product_material": "Cotton",
  "product_color": "indigo",
},
{
  "_id": "650447e57e47696c78119267",
  "id": "4",
  "product_name":
    "Gorgeous Plastic Pants",
  "product_price": 492,
  "product_material": "Soft",
  "product_color": "plum",
},
{
  "_id": "650447e57e47696c7811926d",
  "id": "10",
  "product_name":
    "Generic Wooden Pizza",
  "product_price": 89,
  "product_material": "Frozen",
  "product_color": "indigo",
},
{
  "_id": "650447e57e47696c78119274",
  "id": "17",
  "product_name":
    "Incredible Metal Car",
  "product_price": 36,
  "product_material": "Fresh",
  "product_color": "indigo",
},
{
  "_id": "650447e57e47696c7811927c",
  "id": "25",
  "product_name": "Licensed Steel Car",
  "product_price": 30,
}
```

14 results in 3 ms

MongoDB GUI Blog