MongoDB

Create database and insert documents in collection.

use productDatabase

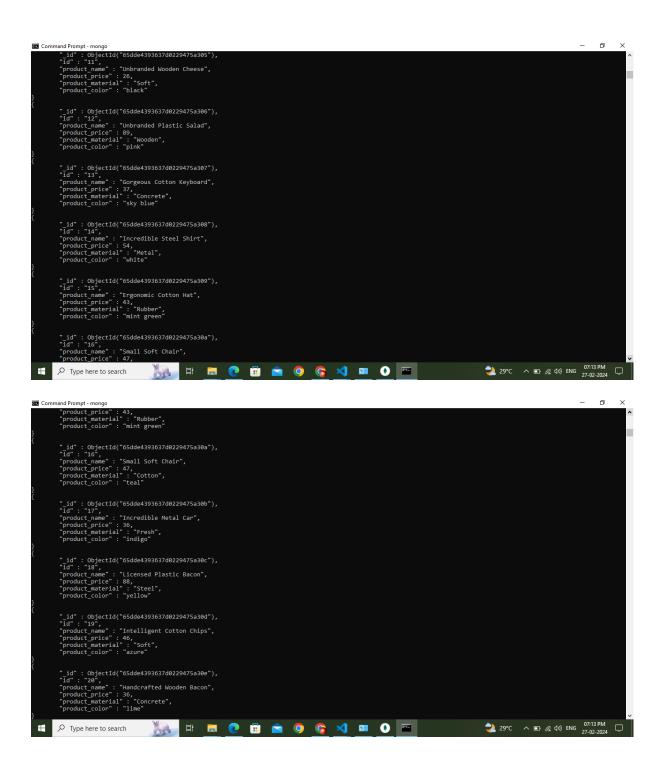
db.products.insertMany()

```
"id": "1",
"product_name": "Intelligent Fresh Chips",
"product_price": 655.00,
"product_material": "Concrete",
"product_color": "mint green"
              :: '4",
oduct_name": "Gorgeous Plastic Pants",
oduct_price": 492.00,
oduct_material": "Soft",
oduct_color": "plum"
     P Type here to search
                                                                                                                              29°C ^ ■ // (4)) ENG 07:06 PM □ (7:02-2024
          "id": "25",
"product_name": "Licensed Steel Car",
"product_price":20.00,
"product material": "Cotton",
"product_color": "indigo"
```

1. Find all the information about each products

db.products.find().pretty();

```
Command Prompt - mong
                                          "id": ObjectId("65dde4393637d0229475a2fb"),
"id": "1",
"product_name": "Intelligent Fresh Chips",
"product_price": 655,
"product_material": "Concrete",
"product_color": "mint green"
                                          "_id" : ObjectId("65dde4393637d0229475a2fc"),
"id" : "2",
"product_name" : "Practical Fresh Sausages",
"product_price" : 911,
"product_material" : "Cotton",
"product_color" : "indigo"
                                          "_id" : ObjectId("65dde4393637d0229475a2fd"),
"id" : "3",
"product_name" : "Refined Steel Car",
"product_price" : 690,
"product_material" : "Rubber",
"product_color" : "gold"
                                          "_id" : ObjectId("65dde4393637d0229475a2fe"),
"id" : "4",
"product_name" : "Gorgeous Plastic Pants",
"product_price" : 492,
"product_material" : "Soft",
"product_color" : "plum"
                                          "_id" : ObjectId("65dde4393637d0229475a2ff"),
"id" : "5",
"product_name" : "Sleek Cotton Chair",
"product_price" : 33,
"product_material" : "Fresh",
"product_color" : "black"
                                          "_id" : ObjectId("65dde4393637d0229475a300"),
"id" : "6",
       ## $\to \tag{Type here to search} \frac{1}{27-02-2024} \quad \quad \tag{T-02-2024} \quad \qq \quad \qu
                                          "_id": ObjectId("65dde4393637d0229475a300"),
"id": "6",
"product_name": "Awasome Wooden Towels",
"product_price": 474,
"product_material": "Plastic",
"product_color": "orange"
                                          "_id" : ObjectId("65dde4393637d0229475a301"),
"id" : "7",
"product_name" : "Practical Soft Shoes",
"product_price" : 500,
"product_material" : "Rubber",
"product_color" : "pink"
                                          "_id" : ObjectId("65dde4393637d0229475a302"),
"id" : "8",
"product_name" : "Incredible Steel Hat",
"product_price" : 78,
"product_material" : "Rubber",
"product_color" : "violet"
                                            "_id" : ObjectId("65dde4393637d0229475a303"),
"Id" : "9"
"product_name" : "Awesome Wooden Ball",
"product_price" : 28,
"product_praterial" : "Soft",
"product_color" : "azure"
                                          "id": ObjectId("65dde4393637d0229475a304"),
"id": "l0",
"product_name": "Generic Wooden Pizza",
"product_price": 84,
"product_material": "Frozen",
"product_color": "indigo"
                                          "_id" : ObjectId("65dde4393637d0229475a305"),
"id" : "11",
"product_name" : "Unbranded Wooden Cheese",
```



```
Command Prompt- mengo

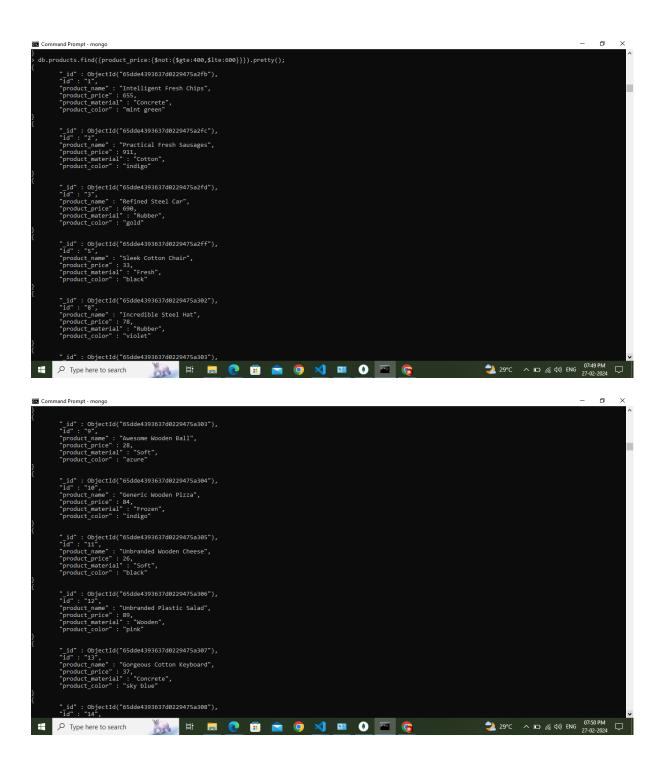
( ".id": Object10("65dde4393637d0229475a30f"),
    "ja": 21",
    "product_pice": "90,
    "product_pice": "90,
    "product_color": "gold"

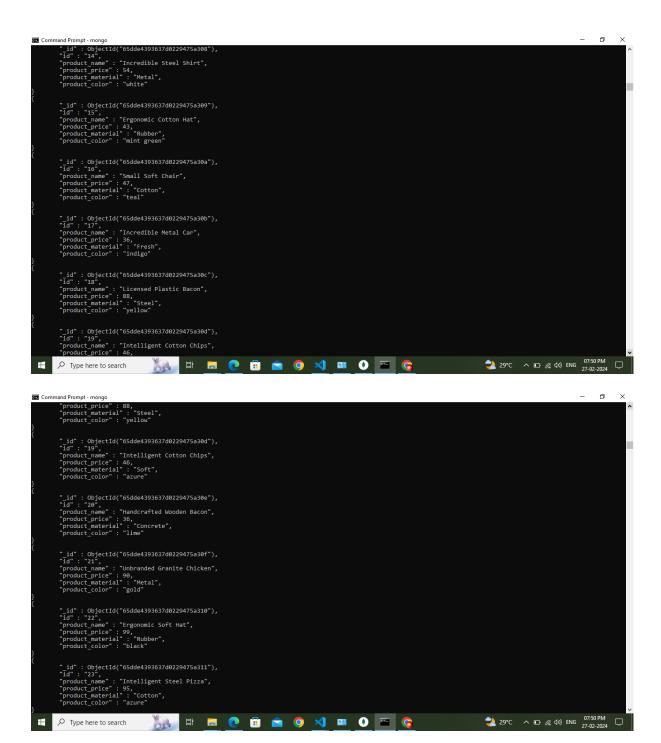
( ".id": Object10("65dde4393637d0229475a310"),
    "ja": Object10("65dde4393637d0229475a310"),
    "ja": Object10("65dde4393637d0229475a310"),
    "ja": 23",
    "product_mane": "Rubnem"; "Rubnem",
    "product_mane": "Intelligent Steel Pizza",
    "product_mane": "Intelligent Steel Pizza",
    "product_mane": "Tasty Rubnem Cheese",
    "ja": Object10("65dde4393637d0229475a312"),
    "ja": 24",
    "ja": Object10("65dde4393637d0229475a312"),
    "ja": 24",
    "ja": Object10("65dde4393637d0229475a312"),
    "ja": 24",
    "ja": Object10("65dde4393637d0229475a312"),
    "ja": 24",
    "ja": Object10("65dde4393637d0229475a312"),
    "ja": 25",
    "ja": Object10("65dde4393637d0229475a313"),
    "ja": 25",
    "ja": Object10("65dde4393637d0229475a313"),
    "ja": 25",
    "ja": 25",
    "ja": 25",
    "ja": 25",
    "ja": 25",
    "ja": 25",
    "ja": Object10("65dde4393637d0229475a313"),
    "ja": 25",
    "ja"
```

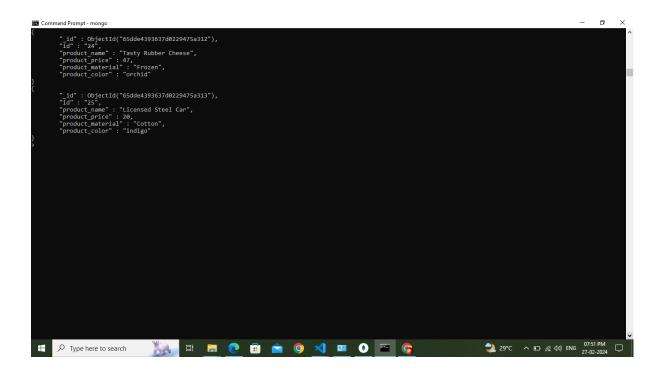
2. Find the product price which are between 400 to 800

db.products.find({product_price:{\$gte:400,\$lte:800}}).pretty();

3. Find the product price which are not between 400 to 600 db.products.find({product_price:{\$not:{\$gte:400,\$lte:600}}}).pretty(







4. List the four product which are greater than 500 in price db.products.find({product_price:{\$gt:500}}).limit(4).pretty();

Only three products are available with price greater than 500.

If we give greater than or equal to 500 then we can get four products. db.products.find({product_price:{\$gte:500}}).limit(4).pretty();

5. Find the product name and product material of each products

db.products.find({},{product_name:1,product_material:1}).pretty();

```
| ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Series | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Series | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Fresh" | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Fresh" | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Series | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Sories | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Mooslen | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Mooslen | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Mooslen | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Metal": ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Metal": ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Rubber" | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Rubber" | ".d": ObjectId("65dde439363708229475a308"),
| "product_material": "Fresh" | ".d": ObjectId("65dde439363708229475a308"),
| "product_material":
```

```
Command Prompt - mongo

"_id": ObjectId("65dde4393637d0229475a30c"),
    "product_material": "Steel"

("_id": ObjectId("65dde4393637d0229475a30d"),
    "product_material": "Soft"

("_id": ObjectId("65dde4393637d0229475a30d"),
    "product_material": "Soft"

("_id": ObjectId("65dde4393637d0229475a30e"),
    "product_material": "Concrete"

("_id": ObjectId("65dde4393637d0229475a30e"),
    "product_material": "Concrete"

("_id": ObjectId("65dde4393637d0229475a30e"),
    "product_material": "Retal"

("_id": ObjectId("65dde4393637d0229475a31e"),
    "product_material": "Retal"

("_id": ObjectId("65dde4393637d0229475a31e"),
    "product_material": "Rubber"

("_id": ObjectId("65dde4393637d0229475a31e"),
    "product_material": "Rubber"

("_id": ObjectId("65dde4393637d0229475a31e"),
    "product_material": "Fauber"

("_id": ObjectId("65dde439
```

6. Find the product with a row id of 10

db.products.findOne({id:"10"});

7. Find only the product name and product material

db.products.find({},{_id:0,product_name:1,product_material:1}).pre
tty();

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

db.products.find({product_material:{\$eq:''Soft''}}).pretty();

9. Find products which contain product color indigo and product price 492.00

None. No products have product color "indigo" and product price "492.00".

But if we can retrieve products which have either product color "indigo" or product price "492.00".

db.products.find({\$or:[{product_color:{\$eq:''indigo''}},{product_price:{\$eq:492.00}}]}).pretty();

```
## Command Prompt - morpo

| db.products.find((Sor:[product_color:(Seq:'indigo']), (product_price:(Seq:492.00)))).pretty();
| di : objectId("65dde439363706229475a2fc'),
| "id : objectId("65dde439363706229475a2fc'),
| "id : objectId("65dde439363706229475a2fc'),
| "id : objectId("65dde439363706229475a2fc'),
| "product_name : "Gorgoous Plastic Pants",
| "product_name : "Gorgoous Plastic Pants",
| "product_name : "Soft",
| "product_name : "Soft",
| "product_name : "Soft",
| "product_name : "Soft",
| "product_name : "Gorgoous Plastic Pants",
| "product_name : "Soft",
| "product_name : "Soft",
| "product_name : "Soft",
| "product_name : "Gorgoous Plastic Pants",
| "product_name : "Soft",
| "product_name : "Indigo"
| "id : objectId("65dde439363706229475a304"),
| "id : "10',
| "product_name : "Indigo"
| "id : objectId("65dde439363706229475a306"),
| "product_name : "Incredible Netal Car",
| "product_name : "Indigo"
| "id : objectId("65dde439363706229475a313"),
| "id : "25',
| "product_name : "Licensed Steel Car",
| "product_name : "L
```

10. Delete the products which product price value are same

```
var
pricesToDelete=db.products.aggregate([{$group:{_id:''$product_price''}
,count:{$sum:1},docs:{$push:''$_id''}}},{$match:{count:{$gt:1}}}]).toAr
ray();
pricesToDelete.forEach(function(priceGroup){db.products.deleteMany
({_id:{$in:priceGroup.docs}});});
```

Initially we have 25 products, in that product_prices 47 and 36 occur twice.

After deleting those now we have 21 products.

```
## Some products.count()

25

30h.products.count()

25

30h.products.count()

25

30h.products.count()

26

30h.products.count()

31

30h.products.aggregate([($group:(_idi:$product_price*,count:($sum:1),docs:($push:"$_id"))),($match:(count:($gt:1))]),1cAnray();

30h.products.aggregate([($group:(_idi:$product_price*,count:($sum:1),docs:($push:"$_id"))),($match:(count:($gt:1))]),1cAnray();

30h.products.count();

30h.products.fonel.etee.db.products.aggregate([($group:(_id:$product_price*,count:($sum:1),docs:($push:"$_id"))),($match:(count:($gt:1))]),toAnray();

30h.products.count();

30h.products.fonel.etee.fonel.etee.db.products.aggregate([($group:(_id:$product_price*,count:($sum:1),docs:($push:"$_id"))),($match:(count:($gt:1))]),toAnray();

30h.products.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.etee.fonel.
```