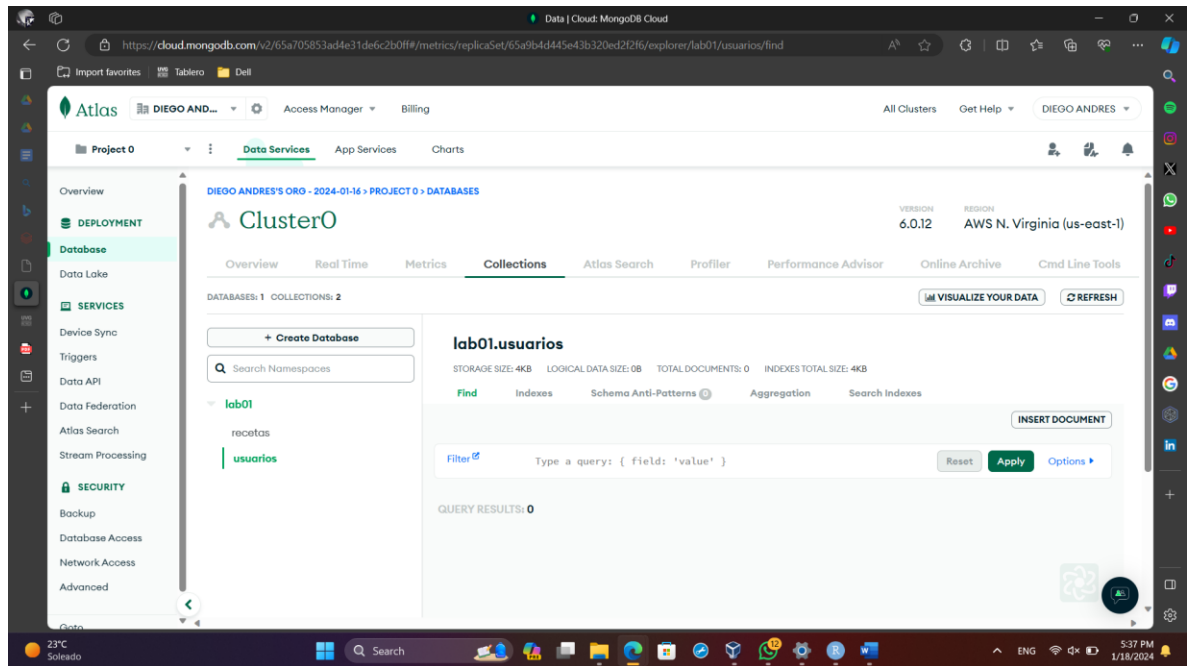


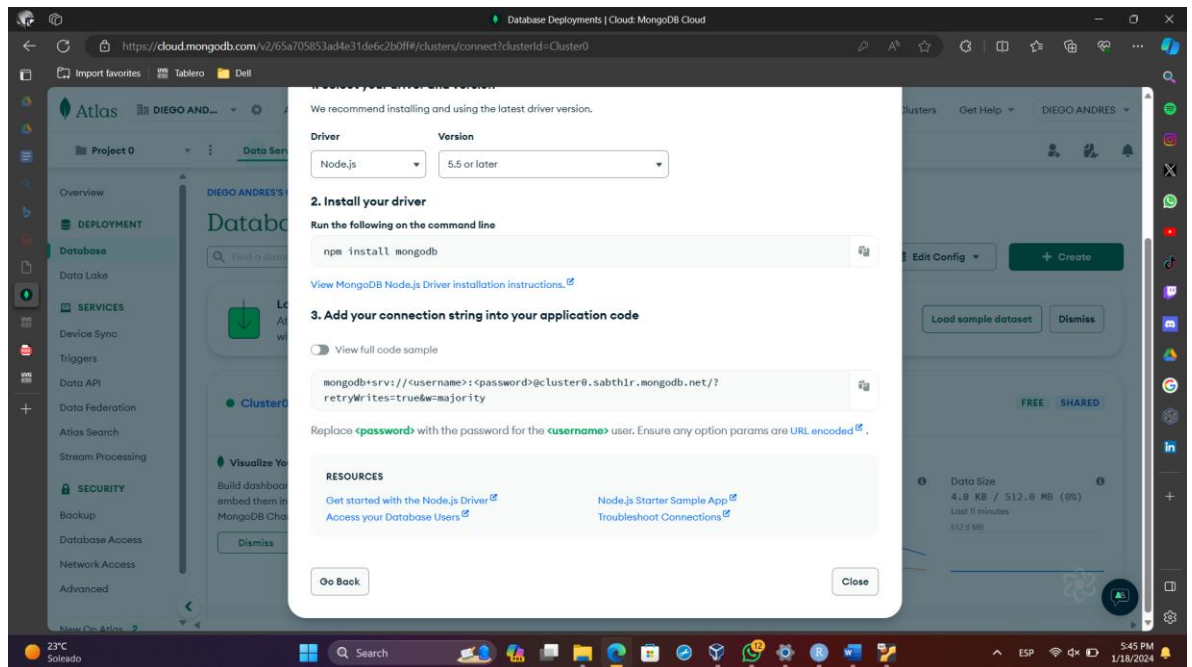
Lab 01 – DB2

1.

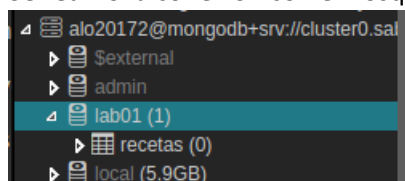
1.1. Crear la base de datos



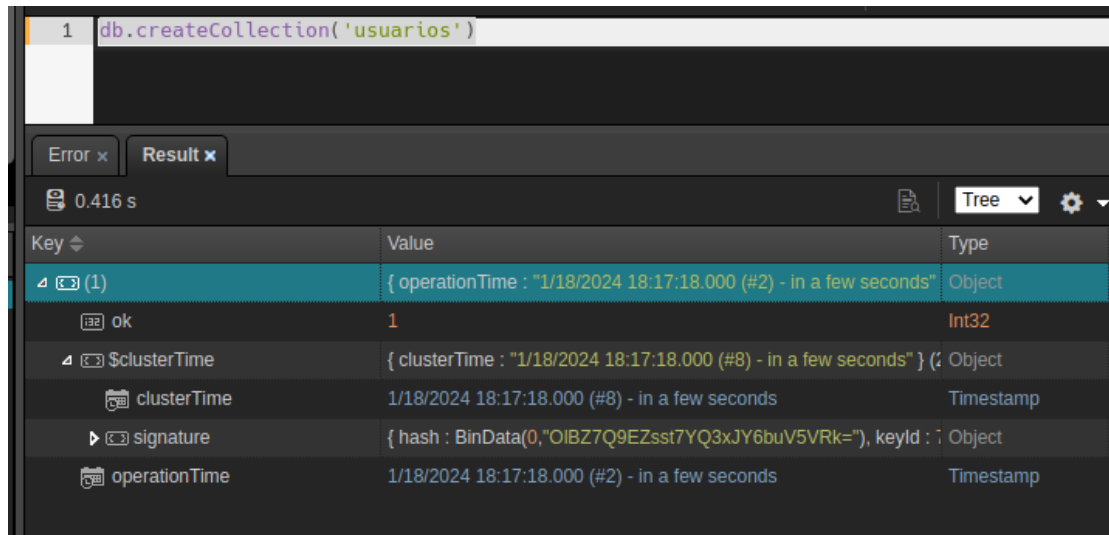
1.2. Obtener la URI



Se realizó la conexión con el nosql



1.3. Se creo la colección de usuarios



1.4. Para importar en línea de comando recetas

```
const contents = [
  {
    content:
"/media/sf_D_DRIVE/UVG/CODING/Semestre7/DB2/Labs/Lab1/data/data/recipes.json",
    collection: "recetas",
    idPolicy: "overwrite_with_same_id",
//overwrite_with_same_id|always_insert_with_new_id|insert_with_new_id_if_id_exists|skip_documents_with_existing_id|abort_if_id_already_exists|drop_collection_first|log_errors

//Use the transformer to customize the import result
//transformer: (doc)=>{ //async (doc)=>{
// doc["importDate"]= new Date();

// doc["oid"] = mb.convert({input:doc["oid"], to:"objectId", onError:"remain_unchanged",
onNull:null}); //to: double|string|objectId|bool|date|int|long|decimal

// return doc; //return null skips this doc

//}
}
];
```

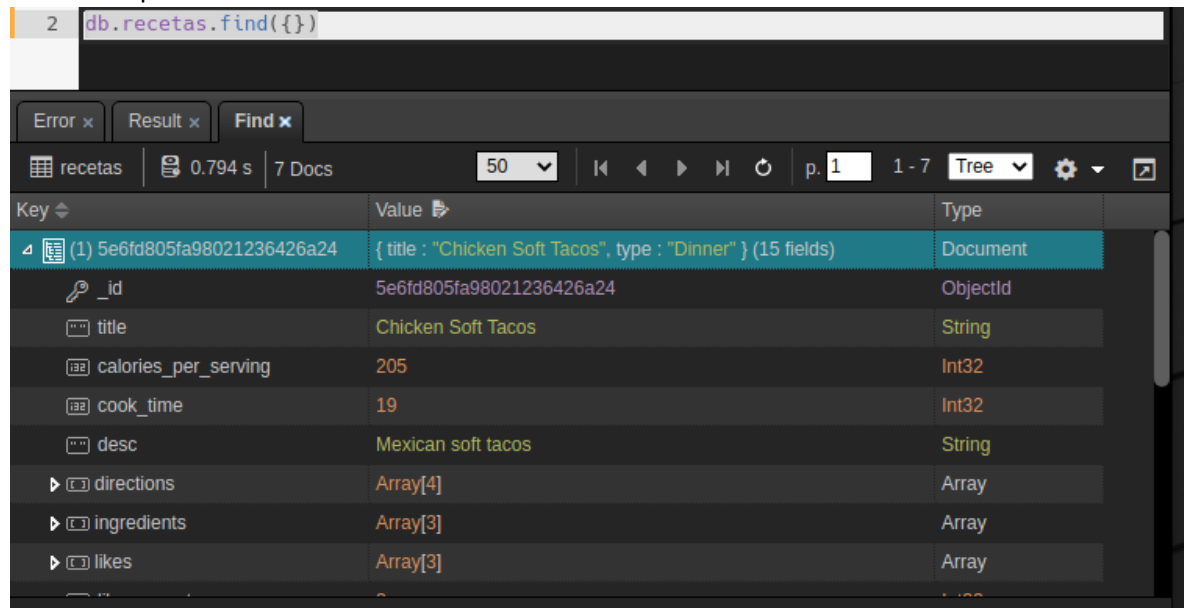
```
mb.importContent({
  connection: "mongodb+srv://cluster0.sabth1r.mongodb.net",
```

```

    database: "lab01",
    fromType: "file",
    batchSize: 2000,
    contents
  })

```

Recetas importado:



Usuarios:

```
const contents = [
```

```

  {
    content:
"/media/sf_D_DRIVE/UVG/CODING/Semestre7/DB2/Labs/Lab1/data/data/users.json",
    collection: "usuarios",
    idPolicy: "overwrite_with_same_id",
//overwrite_with_same_id|always_insert_with_new_id|insert_with_new_id_if_id_exists|skip_documents_with_existing_id|abort_if_id_already_exists|drop_collection_first|log_errors

    //Use the transformer to customize the import result

    //transformer: (doc)=>{ //async (doc)=>{

    // doc["importDate"]= new Date();

    // doc["oid"] = mb.convert({input:doc["oid"], to:"objectId", onError:"remain_unchanged",
onNull:null}); //to: double|string|objectId|bool|date|int|long|decimal

    // return doc; //return null skips this doc

```

```
//}  
}  
];  
  
mb.importContent({  
  connection: "mongodb+srv://cluster0.sabth1r.mongodb.net",  
  database: "lab01",  
  fromType: "file",  
  batchSize: 2000,  
  contents  
})
```

3 db.usuarios.find({})

Error x Result x Find x Find (1) x

usuarios 0.573 s 2 Docs 50 p. 1 1 - 2 Tree

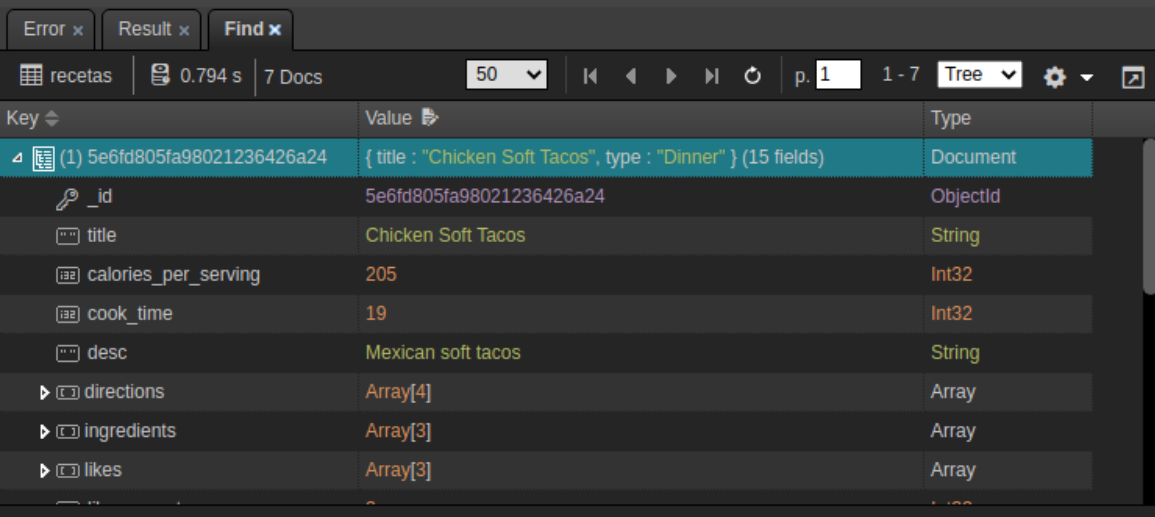
Key	Value	Type
(1) 5ebc44f23a289bc862491d92	{ firstName : "Grace", title : "Rear Admiral", email : "grace@navy.mil"	Document
_id	5ebc44f23a289bc862491d92	ObjectId
user_id	2	Int32
firstName	Grace	String
lastName	Hopper	String
title	Rear Admiral	String
email	grace@navy.mil	String
password	C8w4&CWC^egwecwoWei79chwf	String
(2) 5ee695203260aab97ea0d58c	{ firstName : "Caderyn", lastName : "Jenkins", email : "none@exam"	Document

Free Edition Line: 3, Column: 1 (20 selected) Show Log Feedback 6:23:44 PM

2.

2.1. Cree una consulta para obtener todas las recetas

```
2 db.recetas.find({})
```

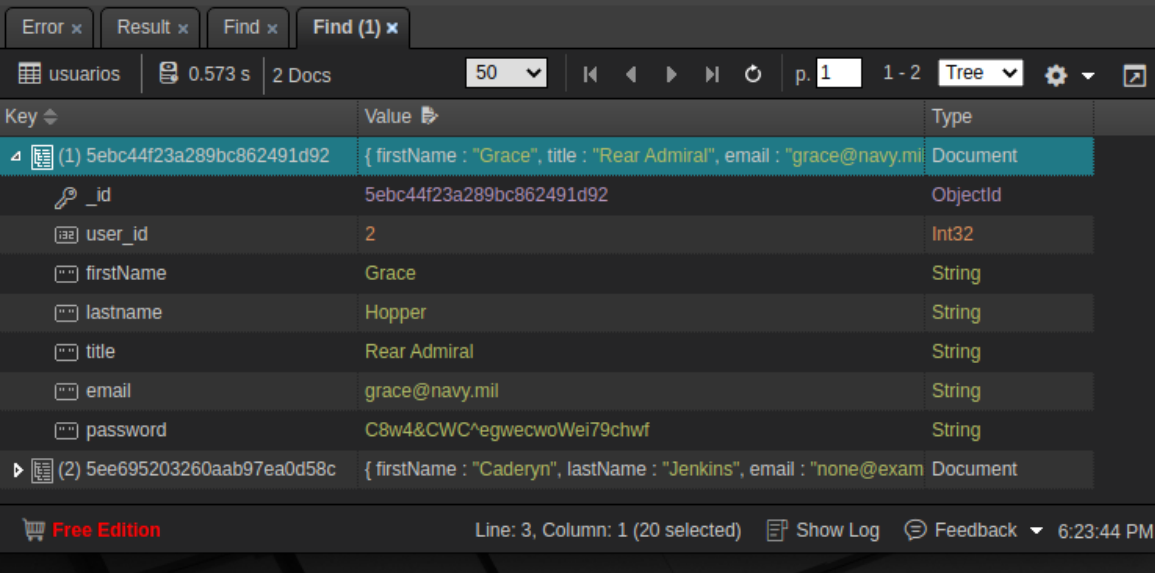


recetas | 0.794 s | 7 Docs | 50 | p. 1 | 1 - 7 | Tree

Key	Value	Type
(1) 5e6fd805fa98021236426a24	{ title : "Chicken Soft Tacos", type : "Dinner" } (15 fields)	Document
_id	5e6fd805fa98021236426a24	ObjectId
title	Chicken Soft Tacos	String
calories_per_serving	205	Int32
cook_time	19	Int32
desc	Mexican soft tacos	String
directions	Array[4]	Array
ingredients	Array[3]	Array
likes	Array[3]	Array

2.2. Cree una consulta para obtener todos los usuarios.

```
3 db.usuarios.find({})
```



usuarios | 0.573 s | 2 Docs | 50 | p. 1 | 1 - 2 | Tree

Key	Value	Type
(1) 5ebc44f23a289bc862491d92	{ firstName : "Grace", title : "Rear Admiral", email : "grace@navy.mil" }	Document
_id	5ebc44f23a289bc862491d92	ObjectId
user_id	2	Int32
firstName	Grace	String
lastname	Hopper	String
title	Rear Admiral	String
email	grace@navy.mil	String
password	C8w4&CWC^egwecwoWei79chw	String
(2) 5ee695203260aab97ea0d58c	{ firstName : "Caderyn", lastName : "Jenkins", email : "none@exam" }	Document

Free Edition | Line: 3, Column: 1 (20 selected) | Show Log | Feedback | 6:23:44 PM

2.3. Crear nuevo documento

```
4 db.recetas.insertOne({
5   "title": "Pepian",
6   "cook_time": 500,
7   "desc": "Comida típica guatemalteca"
8 })
```

0.417 s

Key	Value	Type
(1)	{ acknowledged : true, insertedId : ObjectId("65a9c457fe57ab2ee20fd51f") }	Object
acknowledged	true	Bool
insertedId	65a9c457fe57ab2ee20fd51f	ObjectId

Free Edition Line: 4, Column: 1 (103 selected) Show Log Feedback 6:38:03 PM

2.4. Buscar la receta nueva creada.

```
10 db.recetas.find({}, {'title': 'pepian'})
```

recetas 0.395 s 8 Docs 50 p. 1 1 - 8

Key	Value	Type
(1) 5e6fd805fa98021236426a24	{ title : "pepian" }	Document
_id	5e6fd805fa98021236426a24	ObjectId
title	pepian	String

Free Edition Line: 10, Column: 1 (38 selected) Show Log Feedback 6:41:05 PM

2.5. Cree una consulta en la que liste las recetas, mostrando únicamente el título y su tiempo de cocción

```
29 db.recetas.find({}, {'_id':0, 'title':1, 'cook_time':1})
```

recetas 0.336 s 8 Docs 50 p. 1 1 - 8

Key	Value	Type
(1)	{ title : "Chicken Soft Tacos", cook_time : 19 }	Object
title	Chicken Soft Tacos	String
cook_time	19	Int32

Free Edition Line: 29, Column: 1 (55 selected) Show Log Feedback 6:51:18 PM

- 2.6. Cree una consulta en la que se listen las recetas ordenadas por mayor tiempo de cocción

```
30 db.recetas.find({}).sort({'cook_time':-1})
```

Key	Value	Type
comments	Array[1]	Array
cook_time	10	Int32
desc	Everyone's favorite pancakes	String
directions	Array[7]	Array

- 2.7. Investigue la instrucción update() para poder agregar un rating más a una receta y actualizar el rating promedio

```
1 db.recetas.updateOne({'title':'Pepian'},{
2   $set: {'rating':100}
3 })
```

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 1, modifiedCount : 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Aquí agregamos al rating los valores que utilizaríamos para sacar el rating promedio

```
1 db.recetas.updateOne(
2   { 'title': 'Pepian' },
3   { $set: { 'rating': [100, 99, 98, 90, 91] } }
4 )
```

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 1, modifiedCount : 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

De manera que queda así:

```
21 db.recetas.find({title:'Pepian'});
22 db.recetas.updateOne(
23   { 'title': 'Pepian' },
24   { $push: { 'rating': 97 } }
25 );
```

Find x	Find (1) x	Result x	Find (2) x	Find (3) x	Find (4) x	Result (1) x	Find (5) x
recetas 0.435 s 1 Doc							
Key		Value		Type			
(1) 65add4d9a63216f0fd2a185		{ title : "Pepian", cook_time : 500, desc : "Comida tipica guatemalteca", rating : [100, 99, 98, 90, 91] }	(5 fields)	Document			
_id		65add4d9a63216f0fd2a185		ObjectId			
title		Pepian		String			
cook_time		500		Int32			
desc		Comida tipica guatemalteca		String			
rating		Array[5]		Array			
0		100		Int32			
1		99		Int32			
2		98		Int32			
3		90		Int32			
4		91		Int32			

Ahora para agregar un dato extra seria así:

```
22 db.recetas.updateOne(
23   { 'title': 'Pepian' },
24   { $push: { 'rating': 97 } }
25 );
```

Find x	Find (1) x	Result x	Find (2) x	Find (3) x	Find (4) x	Result (1) x	Find (5) x	Result (2) x
0.416 s								
Key		Value		Type				
(1)		{ acknowledged : true, matchedCount : 1, modifiedCount : 1 }		Object				
acknowledged		true		Bool				
matchedCount		1		Int32				
modifiedCount		1		Int32				

Lo que deja el array de rating como:

```
21 db.recetas.find({title:'Pepian'});
22 db.recetas.updateOne(
23   { 'title': 'Pepian' },
24   { $push: { 'rating': 97 } }
25 );
```

Find x	Find (1) x	Result x	Find (2) x	Find (3) x	Find (4) x	Result (1) x	Find (5) x	Result (2) x	Find (6) x
recetas 0.416 s 1 Doc									
Key		Value		Type					
(1) 65add4d9a63216f0fd2a185		{ title : "Pepian", cook_time : 500, desc : "Comida tipica guatemalteca", rating : [100, 99, 98, 90, 91, 97] }	(5 fields)	Document					
_id		65add4d9a63216f0fd2a185		ObjectId					
title		Pepian		String					
cook_time		500		Int32					
desc		Comida tipica guatemalteca		String					
rating		Array[6]		Array					
0		100		Int32					
1		99		Int32					
2		98		Int32					
3		90		Int32					
4		91		Int32					
5		97		Int32					

Aquí actualicé al rating promedio:

```

36 }
37 };
38
39 db.recetas.updateMany(
40   { title: "Pepian" },
41   [
42     {
43       $set: {
44         rating_avg: {
45           $avg: "$rating"
46         }
47       }
48     }
49   ]
50 );
  
```

0.279 s

Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Y así quedó:

```

27 db.recetas.find({title:'Pepian'});
28
  
```

0.623 s | 1 Doc

Key	Value	Type
(1) 65add4d9a63216f0fd2a185	{ title: "Pepian" } (6 fields)	Document
_id	65add4d9a63216f0fd2a185	ObjectId
title	Pepian	String
cook_time	500	Int32
desc	Comida tipica guatemalteca	String
rating	Array[6]	Array
0	100	Int32
1	99	Int32
2	98	Int32
3	90	Int32
4	91	Int32
5	97	Int32
rating_avg	95.8333	Double

2.8. Se añadieron nuevos ingredientes a esta receta de pepián:

```

53
54 db.recetas.find({})
55
56 db.recetas.updateOne(
57   {title:"Pepian"},
58   {
59     $set: {
60       ingredients: [
61         { name: 'Meat', quantity: { amount: 1, unit: "lbs" } },
62         { name: 'Tomato', quantity: { amount: 5, unit: "lbs" } },
63         { name: 'Water', quantity: { amount: 30, unit: "oz" } },
64         { name: 'Pepper Waque', quantity: { amount: 1, unit: "unit" } },
65       ]
66     }
67   }
68 );
  
```

0.392 s

Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Así quedó la receta:

```
52 db.recetas.find({title:'Pepian'})
53
54 db.recetas.find({})
55
56 db.recetas.updateOne(
57   {title:"Pepian"},
58   {
59     $set: {
60       ingredients:[
61         { name:'Meat', quantity:{amount:1, unit:"lbs"} },
62         { name:'Tomato', quantity:{amount:5, unit:"lbs"} },
63         { name:'Water', quantity:{amount:30, unit:"oz"} },
64         { name:'Pepper Waque', quantity:{amount:1, unit:"unit"} }
65       ]
66     }
67   })
```

recetas 0.310 s 1 Doc

Key	Value	Type
title	Pepian	String
cook_time	500	Int32
desc	Comida típica guatemalteca	String
rating	Array[5]	Array
0	100	Int32
1	99	Int32
2	98	Int32
3	90	Int32
4	91	Int32
rating_avg	95.6	Double
ingredients	Array[4]	Array
0	{ name: "Meat", quantity: { amount: 1, unit: "lbs" } }	Object
1	{ name: "Tomato", quantity: { amount: 5, unit: "lbs" } }	Object
2	{ name: "Water", quantity: { amount: 30, unit: "oz" } }	Object
3	{ name: "Pepper Waque", quantity: { amount: 1, unit: "unit" } }	Object

Eliminar uno de los ingredientes en la receta de pepián:

```
68
69 db.recetas.updateOne(
70   { title: "Pepian" },
71   { $pull: { ingredients: { name: "Tomato" } } }
72 )
```

0.379 s

Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Así quedó luego de eliminar el tomate de la receta:

```
52 db.recetas.find({title:'Pepian'})
53
54 db.recetas.find({})
55
56 db.recetas.updateOne(
57   {title:"Pepian"},
58   {
59     $set: {
60       ingredients:[
61         { name:'Meat', quantity:{amount:1, unit:"lbs"} },
62         { name:'Tomato', quantity:{amount:5, unit:"lbs"} },
63         { name:'Water', quantity:{amount:30, unit:"oz"} },
64         { name:'Pepper Waque', quantity:{amount:1, unit:"unit"} }
65       ]
66     }
67   })
```

recetas 0.458 s 1 Doc

Key	Value	Type
_id	bb93c43a7fb5ab2ee2f0db1f	ObjectId
title	Pepian	String
cook_time	500	Int32
desc	Comida típica guatemalteca	String
rating	Array[5]	Array
0	100	Int32
1	99	Int32
2	98	Int32
3	90	Int32
4	91	Int32
rating_avg	95.6	Double
ingredients	Array[3]	Array
0	{ name: "Meat", quantity: { amount: 1, unit: "lbs" } }	Object
1	{ name: "Water", quantity: { amount: 30, unit: "oz" } }	Object
2	{ name: "Pepper Waque", quantity: { amount: 1, unit: "unit" } }	Object

2.9. Este es el orden descendente del rating promedio de recetas:

```
74 db.recetas.aggregate(
75   {$sort: { rating_avg:-1 } }
76 )
77 db.recetas.aggregate(
```

Result (2) xResult (3) xResult (4) xResult (5) xFind (10) xResult (6) xFind (11) xAggregate (2) xAggregate (3) xError (4) xAggregate (4) xAggregate (5) xAggregate (6) x

recetas0.434 s8 Docs

Key ↓Value ↓Type

▶ (1) 65a9c457fe57ab2ee20fd51f{ title : "Peplan" } (7 fields)Document

▶ (2) 5edf1cd43260aab97ea0d588{ title : "Apple Pie", type : "Dessert" } (16 fields)Document

▶ (3) 5e877cba20a4f574c0aa56da{ title : "Pancakes", type : "Breakfast" } (15 fields)Document

▶ (4) 5e6fd805fa98021236426a24{ title : "Chicken Soft Tacos", type : "Dinner" } (15 fields)Document

▶ (5) 5e5e9c470d33e9e8e3891b35{ title : "Tacos", type : "Dinner" } (15 fields)Document

▶ (6) 5e87856d07beb474c074c5ca{ title : "Brown Sugar Meatloaf", type : "Dinner" } (15 fields)Document

▶ (7) 5edf1d313260aab97ea0d589{ title : "Zucchini Brownies", type : "Dessert" } (14 fields)Document

▶ (8) 5e878f5220a4f574c0aa56db{ title : "Maple Smoked Salmon", type : "Dinner" } (13 fields)Document

Este es el tercer registro con el promedio rating más alto

77 db.recetas.aggregate(
78 {\$sort: { rating_avg:-1 } },
79 { \$skip : 3 }
80).limit(1)

« 3 x

Result (4) x

Result (5) x

Find (10) x

Result (6) x

Find (11) x

Aggregate (2) x

Aggregate (3) x

Error (4) x

Aggregate (4) x

Aggregate (5) x

Aggregate (6) x

Aggregate (7) x

Aggr

recetas | 0.319 s | 1 Doc

Key ↕

Value ↗

Type

▶ (1) 5e6fd805fa98021236426a24

{ title : "Chicken Soft Tacos", type : "Dinner" } (15 fields)

Document

↗ _id

5e6fd805fa98021236426a24

ObjectId

↗ title

Chicken Soft Tacos

String

↗ calories_per_serving

205

Int32

↗ cook_time

19

Int32

↗ desc

Mexican soft tacos

String

▶ ↗ directions

Array[4]

Array

▶ ↗ ingredients

Array[3]

Array

▶ ↗ likes

Array[3]

Array

↗ likes_count

3

Int32

↗ prep_time

10

Int32

▶ ↗ rating

Array[7]

Array

↗ rating_avg

3.71

Double

↗ servings

5

Int32

▶ ↗ tags

Array[4]

Arrav

Free Edition

Lin

77 db.recetas.aggregate(
78 {\$sort: { rating_avg:-1 } },
79 { \$skip : 3 }
80).limit(1)

« 3 x

Result (4) x

Result (5) x

Find (10) x

Result (6) x

Find (11) x

Aggregate (2) x

Agg

recetas | 0.319 s | 1 Doc

Key ↕

Value ↗

▶ (1) 5e6fd805fa98021236426a24

{ title : "Chicken Soft Tacos", type : "Dinner" } (15 fields)

2.10. Recetas que tienen comentarios

82 // Recetas que tienen comentarios
83 db.recetas.find({ comments:{\$exists: true } }).projection({_id:0, title:1, comments:1})

recetas 0.269 s 2 Docs

Key	Value	Type
(1)	{ title: "Pancakes" } (2 fields)	Object
title	Pancakes	String
comments	Array[1]	Array
0	{ body: "I love these! They are so fluffy!", name: "Grace Hopper", user_id: 2 } (4 fields)	Object
(2)	{ title: "Apple Pie" } (2 fields)	Object
title	Apple Pie	String
comments	Array[2]	Array
0	{ name: "Caderyn Jenkins" } (4 fields)	Object
1	{ body: "Mine is better, but this pretty good.", name: "Grace Hopper", user_id: 2 } (4 fields)	Object

2.11. Recetas que son de postres

85 db.recetas.find({ type: "Dessert" }).projection({_id:0, title:1, type:1})

recetas 0.258 s 2 Docs

Key	Value	Type
(1)	{ title: "Apple Pie", type: "Dessert" }	Object
title	Apple Pie	String
type	Dessert	String
(2)	{ title: "Zucchini Brownies", type: "Dessert" }	Object
title	Zucchini Brownies	String
type	Dessert	String

2.12. Recetas que son fáciles:

3 db.recetas.find({ tags:'easy' }).projection({_id:0, title:1, tags:1})

recetas 0.323 s 4 Docs

Key	Value	Type
(1)	{ title: "Chicken Soft Tacos", tags: ["mexican", "quick", "easy", "chicken"] }	Object
title	Chicken Soft Tacos	String
tags	Array[4]	Array
0	mexican	String
1	quick	String
2	easy	String
3	chicken	String
(2)	{ title: "Brown Sugar Meatloaf", tags: ["ground beef", "family meal", "easy"] }	Object
title	Brown Sugar Meatloaf	String
tags	Array[3]	Array
0	ground beef	String
1	family meal	String
2	easy	String
(3)	{ title: "Zucchini Brownies", tags: ["sweets", "easy"] }	Object
title	Zucchini Brownies	String
tags	Array[2]	Array
0	sweets	String
1	easy	String
(4)	{ title: "Tacos", tags: ["mexican", "quick", "easy", "ground beef"] }	Object
title	Tacos	String
tags	Array[4]	Array
0	mexican	String
1	quick	String

Eliminados:

```
6 db.recetas.deleteMany({ tags:'easy' })
```

Find x	Find (1) x	Find (2) x	Result x
0.319 s			
Key	Value	Type	
(1)	{ acknowledged : true, deletedCount : 4 }	Object	
acknowledged	true	Bool	
deletedCount	4	Int32	

Recetas que son fáciles luego del delete:

```
3 db.recetas.find({ tags:'easy' }).projection({ _id:0, title:1, tags:1 })
4
5 // Eliminar recetas faciles
```

Find x	Find (1) x	Find (2) x	Result x	Find (3) x
recetas 0.313 s 0 Doc				
Key	Value	No records found		

Total de recetas luego de hacer el delete

```
12 db.recetas.find({})
13
```

Result x	Result (1) x	Result (2) x	Find x	Result (3) x	Find (1) x
recetas 0.414 s 4 Docs					
Key	Value	Type			
(1) 5e877cba20a4f574c0aa56da	{ title : "Pancakes", type : "Breakfast" } (15 fields)	Document			
_id	5e877cba20a4f574c0aa56da	ObjectId			
title	Pancakes	String			
calories_per_serving	232	Int32			
comments	Array[1]	Array			
cook_time	10	Int32			
desc	Everyone's favorite pancakes	String			
directions	Array[7]	Array			
ingredients	Array[9]	Array			
likes	Array[5]	Array			
likes_count	4	Int32			
prep_time	10	Int32			
rating	Array[9]	Array			
rating_avg	3.88	Double			
servings	4	Int32			
type	Breakfast	String			
(2) 5e878f5220a4f574c0aa56db	{ title : "Maple Smoked Salmon", type : "Dinner" } (13 fields)	Document			
(3) 5edf1cd43260aab97ea0d588	{ title : "Apple Pie", type : "Dessert" } (16 fields)	Document			
(4) 65ad72de1efb78c7027c2ac1	{ title : "Pepian" } (6 fields)	Document			

- 2.13. Con base a la estructura observada en la colección de usuarios, cree 3 nuevos documentos de usuarios en una sola instrucción, que contenga los siguientes campos: nombre, apellido, correo electrónico y contraseña.

```

14 // Agregar 3 documentos de usuarios
15 db.usuarios.find();
16 db.usuarios.insertMany([
17   {firstName:'Diggs', lastName:'Papu', email:'diggs.papu@gmail.com', password: 'Manager123'},
18   {firstName:'Lionel', lastName:'Messi', email:'messi.d10s@gmail.com', password: 'copa2022#'},
19   {firstName:'Diego Armando', lastName:'Maradona', email:'elpelusa@gmail.com', password: 'ElDiegote'},
20 ]
21 )

```

Result x Result (1) x Result (2) x Find x Result (3) x Find (1) x Find (2) x Result (4) x

0.344 s

Key	Value	Type
(1)	{ } (2 fields)	Object
acknowledged	true	Bool
insertedIds	Array[3]	Array
0	65ad79b81efb78c7027c2ac2	ObjectId
1	65ad79b81efb78c7027c2ac3	ObjectId
2	65ad79b81efb78c7027c2ac4	ObjectId

Así quedó la colección de personas:

```

5 db.usuarios.insertMany([
6   {firstName:'Diggs', lastName:'Papu', email:'diggs.papu@gmail.com', password: 'Manager123'},
7   {firstName:'Lionel', lastName:'Messi', email:'messi.d10s@gmail.com', password: 'copa2022#'},
8   {firstName:'Diego Armando', lastName:'Maradona', email:'elpelusa@gmail.com', password: 'ElDiegote'},
9 ])
10
11
12 db.usuarios.find()

```

Result x Find x Result (1) x Find (1) x

usuarios 0.272 s 5 Docs

Key	Value	Type
(1) 5ebc44f23a289bc862491d92	{ firstName: "Grace", title: "Rear Admiral", email: "grace@navy.mil" } (7 fields)	Document
(2) 5ee695203260aab97ea0d58c	{ firstName: "Caderyn", lastName: "Jenkins", email: "none@example.com" } (6 fields)	Document
(3) 65ad837e1efb78c7027c2ac5	{ firstName: "Diggs", lastName: "Papu", email: "diggs.papu@gmail.com", password: "Manager123" } (5 fields)	Document
(4) 65ad837e1efb78c7027c2ac6	{ firstName: "Lionel", lastName: "Messi", email: "messi.d10s@gmail.com", password: "copa2022#" } (5 fields)	Document
(5) 65ad837e1efb78c7027c2ac7	{ firstName: "Diego Armando", lastName: "Maradona", email: "elpelusa@gmail.com", password: "ElDiegote" } (5 fields)	Document

2.14. Así está la colección de personas antes de que se haga el update de las recetas favoritas:

```

64 // 2.14 Cree las consultas para agregarle la receta favorita a cada uno de los usuarios creados anteriormente
65 db.usuarios.find({});
66

```

Result (13) x Result (7) x Find (14) x Result (8) x Find (15) x Find (16) x

usuarios 0.623 s 5 Docs

Key	Value	Type
(1) 5ebc44f23a289bc862491d92	{ firstName: "Grace", title: "Rear Admiral", email: "grace@navy.mil" } (7 fields)	Document
(2) 5ee695203260aab97ea0d58c	{ firstName: "Caderyn", lastName: "Jenkins", email: "none@example.com" } (6 fields)	Document
(3) 65ade03d9a63216f0fd2a186	{ firstName: "Diggs", lastName: "Papu", email: "diggs.papu@gmail.com", password: "Manager123" } (5 fields)	Document
(4) 65ade03d9a63216f0fd2a187	{ firstName: "Lionel", lastName: "Messi", email: "messi.d10s@gmail.com", password: "copa2022#" } (5 fields)	Document
(5) 65ade03d9a63216f0fd2a188	{ firstName: "Diego Armando", lastName: "Maradona", email: "elpelusa@gmail.com", password: "ElDiegote" } (5 fields)	Document

```

14 // Receta favorita
15 db.usuarios.updateOne({firstName:'Lionel', lastName:'Messi'}, {$set:{'favorite_recipe':'Milanesas a la napolitana'}});
16 db.usuarios.updateOne({firstName:'Diego Armando', lastName:'Maradona'}, {$set:{'favorite_recipe':'Ranas Ranas'}});
17 db.usuarios.updateOne({$and:[{firstName:'Diggs'}, {lastName:'Papu'}]}, {$set:{'favorite_recipe':'Pepian'}});
18

```

Result x Find x Result (1) x Find (1) x Result (2) x

0.614 s

Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Así quedó luego del update de la receta favorita:

```

69 db.usuarios.find().projection({_id:0, firstName:1, lastName:1, lastname:1, favorite_recipe:1});
70 // 2.15 Cree una consulta para consultar los distintos nombres de usuarios
71 db.usuarios.distinct("firstName", {"lastName": {"$exists": true}});
72

```

Find (12) x Find (13) x Result (7) x Find (14) x Result (8) x Find (15) x Find (16) x Result (9) x Find (17) x Find (18) x Find (19) x

usuarios 0.542 s 5 Docs

Key	Value	Type
(1)	{ firstName: "Grace", lastname: "Hopper" }	Object
firstName	Grace	String
lastname	Hopper	String
(2)	{ firstName: "Caderyn", lastName: "Jenkins" }	Object
(3)	{ firstName: "Diggs", lastName: "Papu", favorite_recipe: "Pepian" }	Object
(4)	{ firstName: "Lionel", lastName: "Messi", favorite_recipe: "Milanesas a la napolitana" }	Object
(5)	{ firstName: "Diego Armando", lastName: "Maradona", favorite_recipe: "Ranas Ranas" }	Object

2.15. Consulta para obtener los nombres de los usuarios:

```

70 // 2.15 Cree una consulta para consultar los distintos nombres de usuarios
71 db.usuarios.distinct("firstName", {"lastName": {"$exists": true}});
72 // 2.16 Investigue el uso de expresiones regulares en la instrucción find() y cree una consulta para buscar todos
73 db.usuarios.find({email:{$regex:'gmail'}});
74 // 2.17 Agregar un campo de actividad a los usuarios, para indicar si están activos o inactivos con un valor booleano
75 db.usuarios.updateMany({}, {$set: {"activity": false}});
76 db.usuarios.find();
77

```

Find (14) x Result (8) x Find (15) x Find (16) x Result (9) x Find (17) x Find (18) x Find (19) x Find (20) x Find (21) x Result (10) x

0.745 s 4 Docs

Key	Value	Type
(1)	Caderyn	String
(2)	Diego Armando	String
(3)	Diggs	String
(4)	Lionel	String

2.16. Consulta para obtener los usuarios que utilizan Gmail:

```

24 db.usuarios.find({email:{$regex:'gmail'}});

```

Find (5) x Find x Find (1) x Find (2) x

usuarios 0.518 s 3 Docs

Key	Value	Type
(1) 65ad837e1efb78c7027c2ac5	{ firstName: "Diggs", lastName: "Papu", email: "diggs.papu@gmail.com" } (6 fields)	Document
(2) 65ad837e1efb78c7027c2ac6	{ firstName: "Lionel", lastName: "Messi", email: "messi.d10s@gmail.com" } (6 fields)	Document
(3) 65ad837e1efb78c7027c2ac7	{ firstName: "Diego Armando", lastName: "Maradona", email: "elpelusa@gmail.com" } (6 fields)	Document

2.17. Agregar un campo de actividad a los usuarios, para indicar si están activos o inactivos con un valor booleano.

```

27 db.usuarios.updateMany({}, {$set: {"activity": false}});

```

Find (5) x Find x Find (1) x Find (2) x Find (3) x Result x Result (1) x Result (2) x Result (3) x Result (4) x Find (4) x Result (5) x Result (6) x

0.558 s

Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 5, modifiedCount: 5 }	Object
acknowledged	true	Bool
matchedCount	5	Int32
modifiedCount	5	Int32

Y así quedó:

31 db.usuarios.find();

Find (5) xFind xFind (1) xFind (2) xFind (3) xResult xResult (1) xResult (2) xResult (3) xResult (4) xFind (4) xResult (5) xResult (6) xFind (6) x

usuarios0.541 s5 Docs

Key ⇅Value ⇅Type

1

{ firstName: "Grace", title: "Rear Admiral", email: "grace@navy.mil" } (8 fields)

Document

🔗

_id

5ebc44f23a289bc862491d92

ObjectId

🔑

user_id

2

Int32

📄

firstName

Grace

String

📄

lastName

Hopper

String

📄

title

Rear Admiral

String

📄

email

grace@navy.mil

String

📄

password

C8w4&CWC*egwecwoWei79chwif

String

📄

activity

false

Bool

2

{ firstName: "Caderyn", lastName: "Jenkins", email: "none@example.com" } (7 fields)

Document

3

{ firstName: "Diggs", lastName: "Papu", email: "diggs.papu@gmail.com" } (7 fields)

Document

4

{ firstName: "Lionel", lastName: "Messi", email: "messi.d10s@gmail.com" } (7 fields)

Document

5

{ firstName: "Diego Armando", lastName: "Maradona", email: "telpelusa@gmail.com" } (7 fields)

Document

2.18. Cree una consulta en la que inactive a 2 usuarios.

Primero voy a hacer el inverso, una consulta que active a 2 usuarios:

33 db.usuarios.updateMany({\$or:[{firstName:"Diggs"}, {firstName:"Lionel"}]},{\$set:{activity:true}})

Find (5) xFind xFind (1) xFind (2) xFind (3) xResult xResult (1) xResult (2) xResult (3) xResult (4) xFind (4) xResult (5) xResult (6) xFind (6) xResult (7) x

0.451 s

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 2, modifiedCount : 2 }	Object
acknowledged	true	Bool
matchedCount	2	Int32
modifiedCount	2	Int32

Queda así:

34db.usuarios.find().projection({_id:0, firstName:1, activity:1});

35// Cree una consulta que inactive a 2 usuarios.

Find (5) xFind xFind (1) xFind (2) xFind (3) xResult xResult (1) xResult (2) xResult (3) xResult (4) xFind (4) xResult (5) xResult (6) xFind (6) xResult (7) x

usuarios0.559 s5 Docs

Key ↕	Value ↗	Type
▶ (1)	{ firstName: "Grace", activity: false }	Object
▶ (2)	{ firstName: "Caderyn", activity: false }	Object
▶ (3)	{ firstName: "Diggs", activity: true }	Object
▶ (4)	{ firstName: "Lionel", activity: true }	Object
▶ (5)	{ firstName: "Diego Armando", activity: false }	Object

Ahora inactivando:

```
36 db.usuarios.updateMany({$or:[{firstName:"Diggs"}, {firstName:"Lionel"}]},{$set:{activity:false}});
```

Find (5) x Find x Find (1) x Find (2) x Find (3) x Result x Result (1) x Result (2) x Result (3) x Result (4) x Find (4) x Result (5) x Result (6) x Find (6) x Result (7) x

0.409 s

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 2, modifiedCount : 2 }	Object
acknowledged	true	Bool
matchedCount	2	Int32
modifiedCount	2	Int32

Queda así:

34

db.usuarios.find().projection({_id:0, firstName:1, activity:1});

35

// Cree una consulta que inactive a 2 usuarios.

36

db.usuarios.updateMany({\$or:[{firstName:"Diggs"}, {firstName:"Lionel"}]},{\$set:{activity:false}});

Find (5) x

Find x

Find (1) x

Find (2) x

Find (3) x

Result x

Result (1) x

Result (2) x

Result (3) x

Result (4) x

Find (4) x

Result (5) x

Result (6) x

Find (6) x

Result (7) x

usuarios

0.435 s

5 Docs

Key ↕

Value ↕

Type

▶ (1)

{ firstName : "Grace", activity : false }

Object

▶ (2)

{ firstName : "Caderyn", activity : false }

Object

▶ (3)

{ firstName : "Diggs", activity : false }

Object

▶ (4)

{ firstName : "Lionel", activity : false }

Object

▶ (5)

{ firstName : "Diego Armando", activity : false }

Object

2.19. Cree una consulta en la que cambie la unidad de medida de todas las recetas que tienen lb a kg

Estos serían los documentos que se tendrían que actualizar:

```
24 db.recetas.find({"ingredients.amount.unit":"lbs"}).projection({_id:0, title:1, ingredients:1})
25 db.recetas.updateMany(
26   {"ingredients.amount.unit":"lbs"},
27   {"ingredients.amount.unit":"kg"},
28 )
```

Find x Result x Find (1) x Result (1) x Result (2) x Result (3) x Find (2) x Result (4) x Find (3) x Find (4) x Find (5) x

recetas 0.742 s 2 Docs

Key	Value	Type
(1)	{ title : "Maple Smoked Salmon" } (2 fields)	Object
title	Maple Smoked Salmon	String
ingredients	Array[2]	Array
0	{ amount : { quantity : 3, unit : "lbs", name : "salmon" }	Object
1	{ amount : { quantity : 0.5, unit : "cup", name : "maple syrup" }	Object
(2)	{ title : "Peplan" } (2 fields)	Object
title	Peplan	String
ingredients	Array[3]	Array
0	{ name : "Meat", amount : { quantity : 1, unit : "lbs" } }	Object
1	{ name : "Water", amount : { quantity : 30, unit : "oz" } }	Object
2	{ name : "Pepper Waque", amount : { quantity : 1, unit : "unit" } }	Object

De manera que al ejecutar:

```
39 db.recetas.updateMany(
40   {"ingredients.amount.unit":"lbs"},
41   {$set:{"ingredients.$[elem].amount.unit":"kg"}},
42   {arrayFilters:[{"elem.amount.unit":"lbs"}]}
43 );
44
```

Find x Result x Find (1) x Result (1) x Result (2) x Result (3) x Find (2) x Result (4) x Find (3) x Find (4) x Find (5) x Error x Result (5) x Result (6) x Find (6) x

0.355 s

Key	Value	Type
(1)	{ acknowledged : true, matchedCount : 2, modifiedCount : 2 }	Object
acknowledged	true	Bool
matchedCount	2	Int32
modifiedCount	2	Int32

Al buscar por lbs no nos retorna nada:

```
24 db.recetas.find({"ingredients.amount.unit":"lbs"}).projection({_id:0, title:1, ingredients:1});
25 db.recetas.updateMany(
26   {"ingredients.amount.unit":"lbs"},
27   {$set:
```

Find x Result x Find (1) x Result (1) x Result (2) x Result (3) x Find (2) x Result (4) x Find (3) x Find (4) x Find (5) x Error x Result (5) x Result (6) x Find (6) x

recetas 0.380 s 0 Doc

Key	Value	Type
	No records found	

Y al buscar por kg queda así:

```
36 db.recetas.find({"ingredients.amount.unit":"kg"},projection({_id:0, title:1, ingredients:1});
37
38
39 db.recetas.updateMany(
40   {"ingredients.amount.unit":"lbs"},
41   {$set:{"ingredients.$[elem].amount.unit":"kg"}},
42   {arrayFilters:[{"elem.amount.unit":"lbs"}]}
43 );
44
```

recetas 0.422 s 2 Docs

Key	Value	Type
(1)	{ title: "Maple Smoked Salmon" } (2 fields)	Object
title	Maple Smoked Salmon	String
ingredients	Array[2]	Array
0	{ amount: { quantity: 3, unit: "kg" }, name: "salmon" }	Object
1	{ amount: { quantity: 0.5, unit: "cup" }, name: "maple syrup" }	Object
(2)	{ title: "Pepian" } (2 fields)	Object
title	Pepian	String
ingredients	Array[3]	Array
0	{ name: "Meat", amount: { quantity: 1, unit: "kg" } }	Object
1	{ name: "Water", amount: { quantity: 30, unit: "oz" } }	Object
2	{ name: "Pepper Waque", amount: { quantity: 1, unit: "unit" } }	Object

2.20. Cree una consulta en la que elimine a los usuarios inactivos.

Bueno, dado que en la fase anterior puse a todos en inactivo, lo que haré será reutilizar el código para pasar los 2 a activos y después borraré todos los que estén inactivos.

Updates a esos dos para convertirlos a activos.

```
44 db.usuarios.updateMany({$or:[{firstName:'Diggs'},{firstName:'Lionel'}]},{$set:{'activity':'true'}})
```

0.316 s

Key	Value	Type
(1)	{ acknowledged: true, matchedCount: 2, modifiedCount: 2 }	Object
acknowledged	true	Bool
matchedCount	2	Int32
modifiedCount	2	Int32

Quedando así:

```
46 db.usuarios.find({activity:true})
```

usuarios 0.337 s 2 Docs

Key	Value	Type
(1) 65ad837e1efb78c7027c2ac5	{ firstName: "Diggs", lastName: "Papu", email: "diggs.papu@gmail.com" } (7 fields)	Document
_id	65ad837e1efb78c7027c2ac5	ObjectId
firstName	Diggs	String
lastName	Papu	String
email	diggs.papu@gmail.com	String
password	Manager123	String
favorite_recipe	Pepian	String
activity	true	Bool
(2) 65ad837e1efb78c7027c2ac6	{ firstName: "Lionel", lastName: "Messi", email: "messi.d10s@gmail.com" } (7 fields)	Document

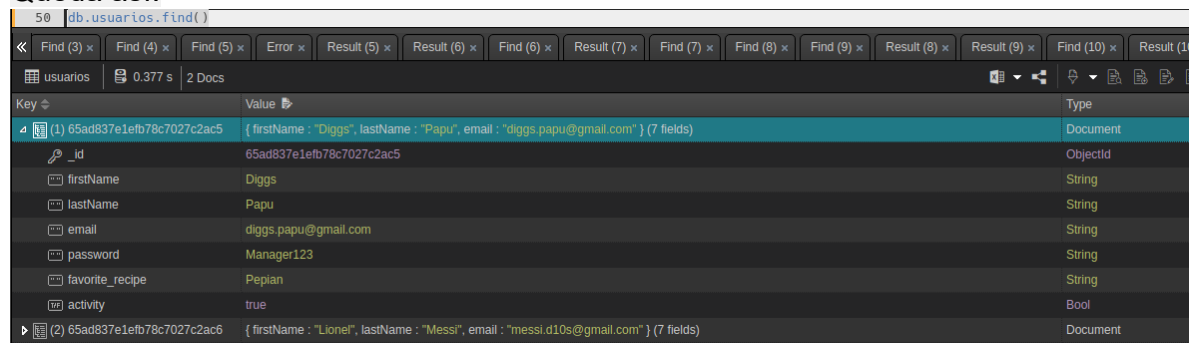
Borrando:

```
48 db.usuarios.deleteMany({"activity":false});
```

0.387 s

Key	Value	Type
(1)	{ acknowledged: true, deletedCount: 3 }	Object
acknowledged	true	Bool
deletedCount	3	Int32

Queda así:



The screenshot shows the MongoDB Compass interface. At the top, the query bar contains `db.usuarios.find()`. Below the query bar, the interface displays the results of the query. The first document is expanded, showing its fields and values. The second document is partially visible below it.

Key	Value	Type
(1) 65ad837e1efb78c7027c2ac5	{ firstName : "Diggs", lastName : "Papu", email : "diggs.papu@gmail.com" } (7 fields)	Document
_id	65ad837e1efb78c7027c2ac5	ObjectId
firstName	Diggs	String
lastName	Papu	String
email	diggs.papu@gmail.com	String
password	Manager123	String
favorite_recipe	Peplan	String
activity	true	Bool
(2) 65ad837e1efb78c7027c2ac6	{ firstName : "Lionel", lastName : "Messi", email : "messi.d10s@gmail.com" } (7 fields)	Document

De manera que solo quedan los usuarios con activo.