

<b>Carrera: INGENIERIA EN INFORMATICA</b>
<b>Asignatura:</b> 3641 – Bases de Datos Aplicada.
<b>Tema:</b> NoSQL
<b>Unidad:</b> Unidad 4
<b>Objetivo:</b> Comprender los conceptos básicos del modelo BASE. Que el alumno sea capaz de implementar en la práctica las instrucciones básicas del lenguaje NoSql a través de la herramienta MongoDB
<b>Competencia/s a desarrollar:</b> <ul style="list-style-type: none"> <li>• Identificación, formulación y resolución de problemas de ingeniería en sistemas de información/informática.</li> <li>• Concepción, diseño y desarrollo de proyectos de ingeniería en sistemas de información / informática.</li> <li>• Gestión, planificación, ejecución y control de proyectos de ingeniería en sistemas de información / informática.</li> <li>• Utilización de técnicas y herramientas de aplicación en la ingeniería en sistemas de información / informática.</li> <li>• Generación de desarrollos tecnológicos y/o innovaciones tecnológicas.</li> <li>• Desempeño en equipos de trabajo.</li> <li>• Comunicación efectiva.</li> </ul>
<b>Descripción de la actividad:</b> <ol style="list-style-type: none"> <li>1. Tiempo estimado de resolución: 1 semana</li> <li>2. Metodología: En computadora.</li> <li>3. Forma de entrega: No obligatoria</li> <li>4. Metodología de corrección y feedback al alumno: Presencial y por Miel.</li> </ol>

## Contenido

Guía práctica .....	3
Creación de Base de datos .....	3
Creación de Colecciones.....	3
Inserción de Documentos .....	5
Tipos de datos.....	9
Actualización de datos.....	12
Borrado de documentos .....	17
Importar Documentos (CVS) .....	19
Indices.....	20
Vistas .....	23
Operaciones de agregación.....	28
Practica adicional .....	34
Documentación complementaria .....	34

# Guía práctica

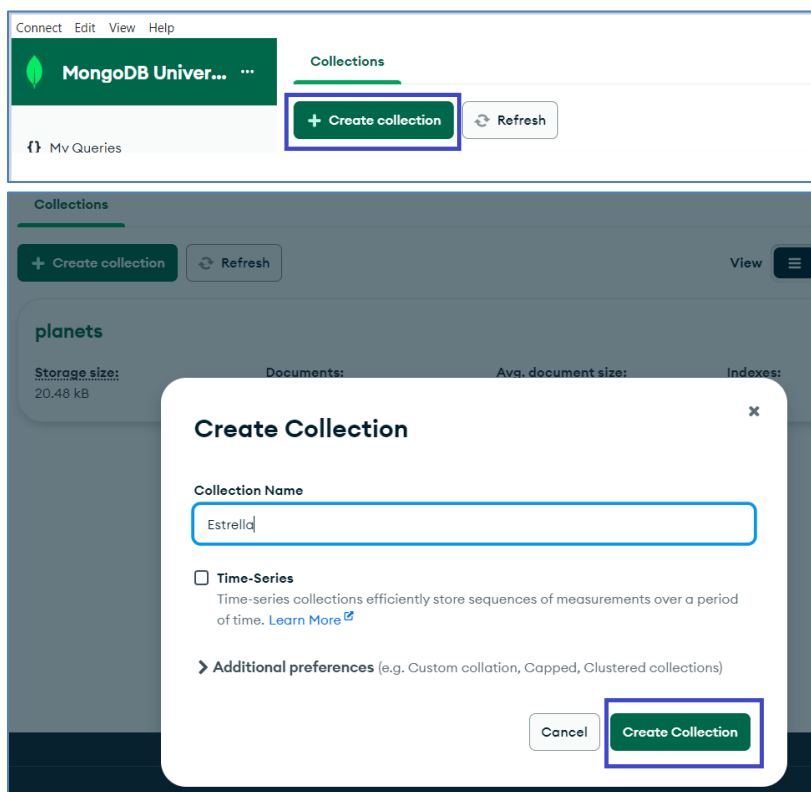
## Ejercicios

### Creación de Base de datos

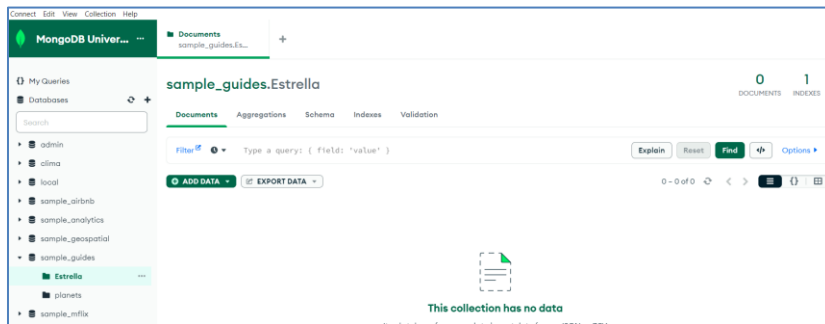
- 1) Instalar la Herramienta Mongo DB (ver ppt). En caso de tener inconvenientes o dudas comunicarse por los canales indicados.
  - a) Conectarse a la base de datos On Cloud indicada en la ppt
  - b) Crear una Base de datos local

### Creación de Colecciones

- 2) Colección: Creación de la colección
  - a) A través de la GUI



## Presionar Refresh



b) Por línea de comando:

Por defecto, mongosh, se conecta a la base de datos test. Para usar una base de datos diferente, ejecute el siguiente comando en mongosh:  
use <database name>

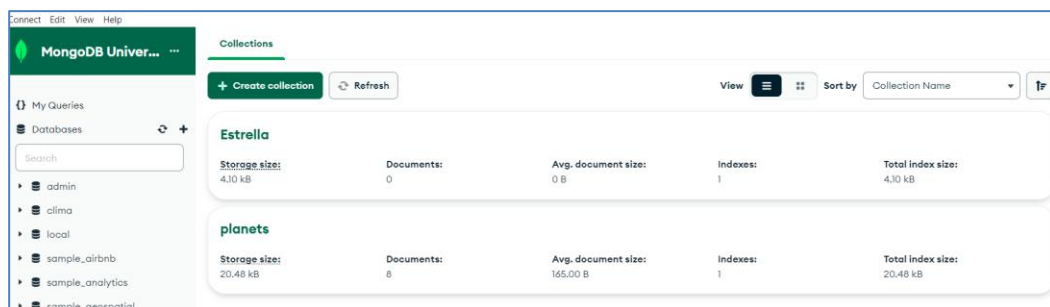
use sample\_guides  
db.createCollection('Estrella');

```
>_MONGOSH

> use sample_guides
< switched to db sample_guides

> db.createCollection('Estrella');
< { ok: 1 }

Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```



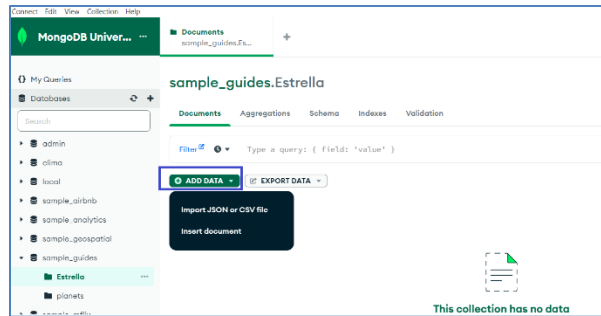
Para ver las colecciones por línea de comando:

*db.getCollectionNames()*

# Inserción de Documentos

## 3) Insertar documentos

### a) Por la GUI



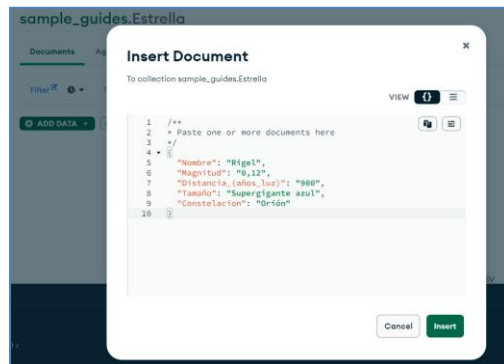
Elegir la opción Insert Document

```
/**
```

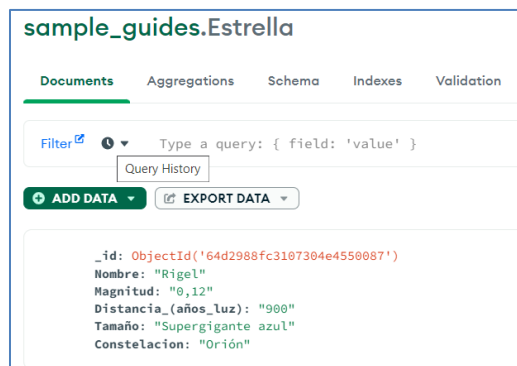
\* Paste one or more documents here

```
*/
```

```
{
  "Nombre": "Rigel",
  "Magnitud": "0,12",
  "Distancia_(años_luz)": "900",
  "Tamaño": "Supergigante azul",
  "Constelacion": "Orión"
}
```



Presionar el botón Insert

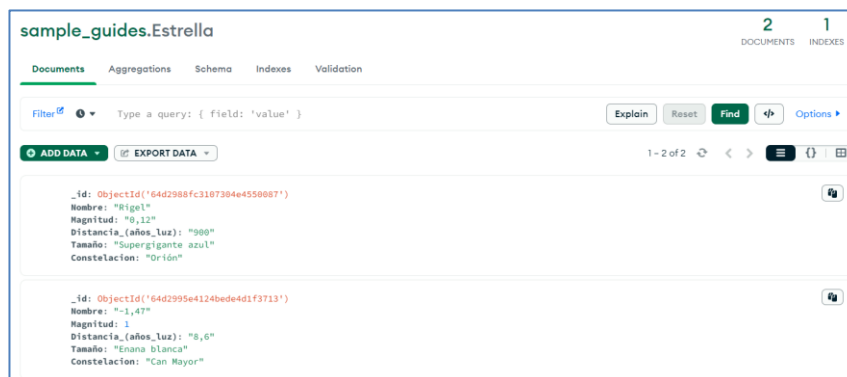


b) Por línea de comando

```
db.Estrella.insertOne({
  "Nombre": "-1,47",
  "Magnitud": 1,
  "Distancia_(años_luz)": "8,6",
  "Tamaño": "Enana blanca",
  "Constelacion": "Can Mayor"
})
```

```
>_MONGOSH

> db.Estrella.insertOne({
  "Nombre": "-1,47",
  "Magnitud": 1,
  "Distancia_(años_luz)": "8,6",
  "Tamaño": "Enana blanca",
  "Constelacion": "Can Mayor"
})
< {
  acknowledged: true,
  insertedId: ObjectId("64d2995e4124bede4d1f3713")
}
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```



- 4) Insertar/Agregar varios documentos por línea de comando:

```
db.Estrella.insertMany([
  {
    "Nombre": "Canopus",
    "Magnitud": "0,72",
    "Distancia_(años_luz)": "309,8",
    "Tamaño": "Supergigante blanco-amarilla",
    "Constelacion": "Carina"
  },
  {
    "Nombre": "Arturo",
    "Magnitud": "0,04",
    "Distancia_(años_luz)": "36,7",
    "Tamaño": "Gigante naranja",
    "Constelacion": "Boyero"
  },
  {
    "Nombre": "Alfa Centauri A",
    "Magnitud": "-0,01",
    "Distancia_(años_luz)": "4,37",
    "Tamaño": "Enana amarilla",
    "Constelacion": "Centauro"
  }
])
```

```
>_MONGOSH
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>> db.Estrella.insertMany([
  {
    "Nombre": "Canopus",
    "Magnitud": "0,72",
    "Distancia_(años_luz)": "309,8",
    "Tamaño": "Supergigante blanco-amarilla",
    "Constelacion": "Carina"
  },
  {
    "Nombre": "Arturo",
    "Magnitud": "0,04",
    "Distancia_(años_luz)": "36,7",
    "Tamaño": "Gigante naranja",
    "Constelacion": "Boyero"
  },
  {
    "Nombre": "Alfa Centauri A",
    "Magnitud": "-0,01",
    "Distancia_(años_luz)": "4,37",
    "Tamaño": "Enana amarilla",
    "Constelacion": "Centauro"
  }
])
```

```
>_MONGOSH

    "Constelacion": "Carina"
  },
  {
    "Nombre": "Arturo",
    "Magnitud": "0,04",
    "Distancia_(años_luz)": "36,7",
    "Tamaño":"Gigante naranja",
    "Constelacion": "Bojero"
  },
  {
    "Nombre": "Alfa Centauri A",
    "Magnitud": "-0,01",
    "Distancia_(años_luz)": "4,37",
    "Tamaño":"Enana amarilla",
    "Constelacion": "Centauro"
  }
])
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("64d29ade4124bede4d1f3714"),
    '1': ObjectId("64d29ade4124bede4d1f3715"),
    '2': ObjectId("64d29ade4124bede4d1f3716")
  }
}

Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```

sample\_guides.Estrella

21DOCUMENTSINDEXES

DocumentsAggregationsSchemaIndexesValidation

FilterType a query: { field: 'value' }

ExplainResetFindOptions

ADD DATAEXPORT DATA

1 - 5 of 5

	Estrella					
	_id ObjectId	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	
1	ObjectId('64d2988fc3107304e45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"	
2	ObjectId('64d2995e4124bede4d1...')	"-1,47"	1	"8,6"	"Enana blanca"	
3	ObjectId('64d29ade4124bede4d1...')	"Canopus"	"0,72"	"309,8"	"Supergigante blanco-amarilla"	
4	ObjectId('64d29ade4124bede4d1...')	"Arturo"	"0,04"	"36,7"	"Gigante naranja"	
5	ObjectId('64d29ade4124bede4d1...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"	



## Tipos de datos

### 5) Tipos de datos

Al crear un documento con datos el tipo de dato se asigna según como se setea el contenido al momento del insert

★ Estrella					
	_id ObjectId	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String
1	ObjectId('64d2988fc3107304e45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"
2	ObjectId('64d2995e4124bede4d1...')	"Sirio"	1	"8,6"	"Enana blanca"
3	ObjectId('64d29ade4124bede4d1...')	"Canopus"	"0,72"	"309,8"	"Supergigante blanco-amarilla"
4	ObjectId('64d29ade4124bede4d1...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"

Desde la GUI podemos modificar el tipo de dato que queremos establecer para ese campo. Al corregir el tipo de dato, se redefine el tipo de dato de la colección.

★ Estrella					
	_id ObjectId	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String
1	ObjectId('64d2988fc3107304e45...')	"Rigel"	0,12 String +		"Supergigante azul"
CANCEL UPDATE					
2	ObjectId('64d2995e4124bede4d1...')	"Sirio"	1	"8,6"	"Enana blanca"
3	ObjectId('64d29ade4124bede4d1...')	"Canopus"	"0,72"	"309,8"	"Supergigante blanco-amarilla"
4	ObjectId('64d29ade4124bede4d1...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"

★ Estrella					
	_id ObjectId	Nombre String	Magnitud Double	Distancia_(años_luz) String	Tamaño String
1	ObjectId('64d2988fc3107304e45...')	"Rigel"	0.12	"900"	"Supergigante azul"
2	ObjectId('64d2995e4124bede4d1...')	"Sirio"	1	"8,6"	"Enana blanca"
3	ObjectId('64d29ade4124bede4d1...')	"Canopus"	0.72	"309,8"	"Supergigante blanco-amarilla"
4	ObjectId('64d29ade4124bede4d1...')	"Alfa Centauri A"	-0.01	"4,37"	"Enana amarilla"

Recuerden que las Base de datos NoSQL nos ofrecen flexibilidad, es decir, en cada campo se pueden de cada documento se pueden colocar diferentes tipos de datos.

Crear la colección Asteroide

Insertar los documentos según la tabla que se encuentra debajo

Modificar el tipo de dato en caso de que sea necesario.

Nombre	Dimensiones	Año Descubrimiento
Winchester 747	171.71 km	1913
Hilda 153	170.63 km	1875
Pretoria 790	170 km	1912
Aegle 96	169.9 km	1868

## **String**

Este tipo de datos es uno de los más utilizados en MongoDB para almacenar datos, los strings BSON son de tipo UTF-8. Por ende, los controladores para cada lenguaje de programación se convierten del formato de string del lenguaje a UTF-8 mientras serializan y desrealizan BSON. El string debe ser un UTF-8 válido.

## **Integer**

Los datos de tipo integer (entero) en MongoDB se usan para almacenar valores numéricos. Podemos almacenar el tipo de datos integer (entero) de 2 formas: entero de 32 bits y entero de 64 bits con signo.

## **Double**

Este tipo de datos se utiliza para almacenar los valores de punto flotante, tener en cuenta que el separador decimal es el punto '.' veamos el ejemplo a continuación:

```
"Distancia_(años_luz)": 8.6  
"Magnitud": 0.71
```

## **Booleano**

Los booleans o booleanos se utilizan para almacenar valores true o false, veamos un ejemplo a continuación:

```
"hasRings": false  
"hasRings": true
```

## **Null**

Este tipo de dato se utiliza para definir el valor de un campo como nulo o inexistente.  
"Constelacion": null

## **Array**

Este tipo de datos esta compuesto por un conjunto de valores en su interior, en un array podemos almacenar valores del mismo tipo en una sola clave, veamos un ejemplo a continuación:

```
"mainAtmosphere": ["CO2", "N"]
```

### **Object (Objeto)**

Este tipo de datos almacena documentos incrustados, a los cuales también se les conoce como documentos anidados. Los documentos incrustados o anidados son aquellos tipos de documentos que contienen un documento dentro de otro documento. Veamos un ejemplo a continuación:

```
"surfaceTemperatureC": {  
  "min": null,  
  "max": null,  
  "mean": 464  
}
```

### **Date (Fecha)**

Este tipo de datos es también uno de los más usados, pues se considera que la mayoría de datos debe tener una fecha para poder gestionarlos mejor. El tipo de dato fecha es un entero (integer) de 64 bits que representa el número de milisegundos. El tipo de datos BSON generalmente admite la fecha y hora UTC y está firmado. Si el valor de este tipo de datos es negativo, entonces representa fechas anteriores al año 1970.

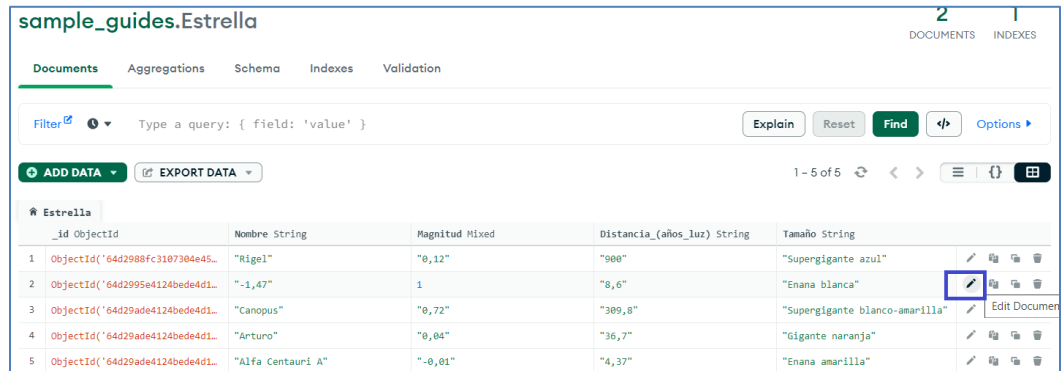
Existen diferentes métodos para devolver fechas, se puede devolver como un string o como un objeto de fecha usando `Date()`, `new Date()` y `new ISODate()`.

```
"Fecha Descubrimiento": "Wed Oct 16 2012 09:16:11 GMT-0500 "  
"Fecha Descubrimiento": "2012-10-16T00:00:02.012+00:00"  
"Fecha Descubrimiento": "2012-10-16"
```

## Actualización de datos

### 6) Update/Actualizar

#### a) Por la GUI




sample\_guides.Estrella

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA 1 - 5 of 5

	_id ObjectId	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	
1	ObjectId('64d2988fc3107304e45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"	
2	ObjectId('64d2995e4124bede4d1...')	"-1,47"	1	"8,6"	"Enana blanca"	
3	ObjectId('64d29ade4124bede4d1...')	"Canopus"	"0,72"	"309,8"	"Supergigante blanco-amarilla"	
4	ObjectId('64d29ade4124bede4d1...')	"Arturo"	"0,04"	"36,7"	"Gigante naranja"	
5	ObjectId('64d29ade4124bede4d1...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"	

#### Valores Actuales

2	ObjectId('64d2995e4124bede4d1...')	<input type="text" value="-1,47"/>	<input type="text" value="String"/>	<input type="text" value="8,6"/>	<input type="text" value="Enana blanca"/>	<input type="button" value="CANCEL"/> <input type="button" value="UPDATE"/>
---	------------------------------------	------------------------------------	-------------------------------------	----------------------------------	---	---

#### Escribo el nuevo valor y presiono el valor Update

2	ObjectId('64d2995e4124bede4d1...')	<input type="text" value="Sirio"/>	<input type="text" value="String"/>	<input type="text" value="8,6"/>	<input type="text" value="Enana blanca"/>	<input type="button" value="CANCEL"/> <input type="button" value="UPDATE"/>
---	------------------------------------	------------------------------------	-------------------------------------	----------------------------------	---	---

Document modified.

#### Valor actualizado

2	ObjectId('64d2995e4124bede4d1...')	"Sirio"	1	"8,6"	"Enana blanca"	
---	------------------------------------	---------	---	-------	----------------	--

#### b) Por línea de comando:

```
db.collection. update (  
    {query}, --> que documento queremos modificar  
    {$set: {campo:Valor} --> campo y valor  
})
```

```
db.Estrella.update({Nombre:"Rigel"},{$set:{"Fecha Descubrimiento":null}})
```

```
> MONGOSH  
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides> db.Estrella.update({Nombre:"Rigel"},{$set:{"Fecha Descubrimiento":null}})
```

Se verifica que se realizó el cambio observando los valores que se encuentran en los parámetros:

acknowledged: true,  
insertedId: null, --> valor insertado  
matchedCount: 1, --> cantidad de documentos encontrados  
modifiedCount: 0, --> cantidad de documentos modificados  
upsertedCount: 0 --> cantidad de documentos creados

```
>_MONGOSH
> db.Estrella.update({Nombre:"Rigel"},{$set:{"Fecha Descubrimiento":null}})
< DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```

Otro ejemplo

```
db.Estrella.update({_id:"64d2995e4124bede4d1f3713"},{$set:{"Fecha Descubrimiento":null}})
```

```
>_MONGOSH
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides> db.Estrella.update({_id:"64d2995e4124bede4d1f3713"},{$set:{"Fecha Descubrimiento":null}})
```

En este caso vemos que no encontró el objeto indicado, por lo tanto, no realiza ninguna modificación.

```
>_MONGOSH
> db.Estrella.update({_id:"64d2995e4124bede4d1f3713"},{$set:{"Fecha Descubrimiento":null}})
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```

El error se debe a la forma en que se indica el campo query, para poder realizar la búsqueda por ese campo se debe escribir de la siguiente manera:

```
db.Estrella.update(
  {_id: ObjectId("64d2995e4124bede4d1f3713")},
  {
    $set: {
      "Fecha Descubrimiento":null
    }
  })
```

```
>_MONGOSH
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides> db.Estrella.update(
  {_id: ObjectId("64d2995e4124bede4d1f3713")},
  {
    $set: {
      "Fecha Descubrimiento":null
    }
  })
```

```
>_MONGOSH
> db.Estrella.update(
  {_id: ObjectId("64d2995e4124bede4d1f3713")},
  {
    $set: {
      "Fecha Descubrimiento":null
    }
  })
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>|
```

Y vemos en la gui el valor actualizado

sample\_guides.Estrella

2 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA

1 - 5 of 5

# Estrella	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	Constelacion String	Fecha Descubrimiento	
1	"0,12"	"900"	"Supergigante azul"	"Orión"	null	
2	1	"8,6"	"Enana blanca"	"Can Mayor"	null	
3	"0,72"	"309,8"	"Supergigante blanco-amarilla"	"Carina"	No field	
4	"0,04"	"36,7"	"Gigante naranja"	"Boyero"	No field	
5	"-0,01"	"4,37"	"Enana amarilla"	"Centauro"	2012-10-16T00:00:02.01	

Update y UpdateOne funcionan de la misma manera, he aquí un ejemplo:

```
db.Estrella.updateOne(
  {Nombre: "Canopus"},
  {
    $set: {
      "Fecha Descubrimiento":null
    }
  })
```

```
>_MONGOSH
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides> db.Estrella.updateOne(
  {Nombre: "Canopus"},
  {
    $set: {
      "Fecha Descubrimiento":null
    }
  })

>_MONGOSH
> db.Estrella.updateOne(
  {Nombre: "Canopus"},
  {
    $set: {
      "Fecha Descubrimiento":null
    }
  })
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```

7) Armar el script para realizar un insert, utilizando los datos que se encuentran debajo.

Nombre	Magnitud	Distancia (años luz)	Tamaño	Constelación
Sirio	-1,47	8,6	Enana blanca	Can Mayor
Canopus	0,72	309,8	Supergigante blanco-amarilla	Carina
Arturo	0,04	36,7	Gigante naranja	Boyero
Alfa Centauri A	-0,01	4,37	Enana amarilla	Centaurio
Vega	0,03	25,3	Gigante blanca	Lira
Rigel	0,12	900	Supergigante azul	Orión
Procyon	0,34	11,4	Supergigante azul	Canis Minor
Achernar	0,5	144	Gigante blanca	Eridano
Betelgeuse	0,3-1,2	640	Supergigante roja	Orión
Hadar	0,6	530	Gigante blanca	Centaurio
Capella	0,71	42	Gigante amarilla	Auriga
Altair	0,8	16,8	Enana blanca	Águila
Aldebarán	0,9	65	Gigante naranja	Tauro
Spica	1,4	260	Gigante azul	Virgo
Antares	1,09	550	Supergigante roja	Scorpio
Pollux	1,15	33,7	Gigante roja	Géminis
Fomalhaut	1,16	25	Enana-subgigante blanca	Piscis Austrinus
Deneb	1,34	1.425	Supergigante blanca	Cygnus
Becrux	1,3	280	Subgigante azul	Crux Australis
Alfa Centauri B	1,2	4,5	Enana naranja	Centaurus
Regulus	1,35	77	Subgigante azul	Leo
Acrux A	1,33	325	Subgigante azul	Crux Australis
Shaula	1,62	570	Subgigante azul	Scorpio
Gacrux	1,63	88	Gigante roja	Crux Australis
Bellatrix	1,64	240	Gigante azul	Orión
Elnath	1,68	131	Gigante azul	Tauro
Beta Carinae	1,7	110	Subgigante blanca	Carina
Alnilam	1,7	1.340	Supergigante azul	Orión
Alnitak A	1,7	700	Supergigante azul	Orión

8) Al dato insertado en el punto anterior realizar el update del campo Fecha Descubrimiento por el valor null



# Borrado de documentos

## 9) Comando delete

### a) Por la GUI

sample\_guides.Estrella

DOCUMENTSINDEXES
















DocumentsAggregationsSchemaIndexesValidation

FilterType a query: { field: 'value' }

ExplainResetFindOptions

ADD DATAEXPORT DATA

1 - 5 of 5

#	Estrella	jectid	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	Const
1	Id('64d298fc3187384a45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"	"Ordr	  
2	Id('64d2995e4124bede4d...')	"Sirio"	1	"8,6"	"Enana blanca"	"Can"	  
3	Id('64d29ade4124bede4d...')	"Canopus"	"0,72"	"389,8"	"Supergigante blanco-amarillo"	"Carir"	  
4	Id('64d29ade4124bede4d...')	"Antares"	"0,84"	"36,7"	"Gigante naranja"	"Boyer"	  
5	Id('64d29ade4124bede4d...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"	"Centi"	  

Se selecciona el documento que se desea eliminar y se presiona el botón delete

sample\_guides.Estrella

21

DOCUMENTSINDEXES

DocumentsAggregationsSchemaIndexesValidation
















FilterType a query: { field: 'value' }

ExplainResetFindOptions

ADD DATAEXPORT DATA

1-5 of 5

# Estrella

	jectid	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	Const
1	Id('64d298fc3187384a45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"	"Ordr"   
2	Id('64d2995e4124bede4d...')	"Sirio"	1	"8,6"	"Enana blanca"	"Can"   
3	Id('64d29ade4124bede4d...')	"Canopus"	"0,72"	"389,8"	"Supergigante blanco-amarillo"	"Carir"   
4	Id('64d29ade4124bede4d...')	"Antares"	"0,84"	"36,7"	"Gigante naranja"	"Boyer"   
5	Id('64d29ade4124bede4d...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"	"Centi"   

Delete Document

Se presiona el botón delete

sample\_guides.Estrella

2

DOCUMENTS

1

INDEXES

Documents

Aggregations

Schema

Indexes

Validation

Filter

Type a query: { field: 'value' }

Explain

Reset

Find

Options

ADD DATA

EXPORT DATA

1 - 5 of 5

Estrella

jectid	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	Const	
1	Id('64d2988fc3187384a45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"	"Ordr" / /
2	Id('64d2995e4124bede4d...')	"Sirio"	1	"8,6"	"Enana blanca"	"Can" / /
3	Id('64d29ade4124bede4d...')	"Canopus"	"0,72"	"389,8"	"Supergigante blanco-amarilla"	"Carir" / /
4	Id('64d29ade4124bede4d...')	"Antares"	"0,84"	"36,7"	"Gigante naranja"	"Boyer" / /

Document flagged for deletion.

CANCEL

DELETE

5 Id('64d29ade4124bede4d...') | "Alfa Centauri A" | "-0,01" | "4,37" | "Enana amarilla" | "Centi" / / |

Se verifica el documento eliminado

sample\_guides.Estrella

41

DOCUMENTSINDEXES

DocumentsAggregationsSchemaIndexesValidation

FilterType a query: { field: 'value' }

ExplainResetFindOptions

ADD DATAEXPORT DATA

1 - 4 of 4

#	Estrella	jectId	Nombre String	Magnitud Mixed	Distancia_(años_luz) String	Tamaño String	Const
1	Id('64d298fc3187384a45...')	"Rigel"	"0,12"	"900"	"Supergigante azul"	"Ordr	
2	Id('64d2995e4124bede4d...')	"Sirio"	1	"8,6"	"Enana blanca"	"Can"	
3	Id('64d29ade4124bede4d...')	"Canopus"	"0,72"	"389,8"	"Supergigante blanco-amarilla"	"Carir"	
4	Id('64d29ade4124bede4d...')	"Alfa Centauri A"	"-0,01"	"4,37"	"Enana amarilla"	"Centi"	

- b) Por línea de comando
- Seleccionamos el documento a eliminar
- ```
db.movies.delete( { Nombre: "Arturo" })
```

sample\_guides.Estrella

5 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA 1 - 5 of 5

|   | _id ObjectId                       | Nombre String     | Magnitud Mixed | Distancia_(años_luz) Mixed | Tamaño String                  |  |
|---|------------------------------------|-------------------|----------------|----------------------------|--------------------------------|--|
| 1 | ObjectId('64d2988fc3107304e45...') | "Rigel"           | "0,12"         | "900"                      | "Supergigante azul"            |  |
| 2 | ObjectId('64d2995e4124bede4d1...') | "Sirio"           | 1              | "8,6"                      | "Enana blanca"                 |  |
| 3 | ObjectId('64d29ade4124bede4d1...') | "Canopus"         | "0,72"         | "309,8"                    | "Supergigante blanco-amarilla" |  |
| 4 | ObjectId('64d29ade4124bede4d1...') | "Alfa Centauri A" | "-0,01"        | "4,37"                     | "Enana amarilla"               |  |
| 5 | ObjectId('64d2df14c3107304e45...') | "Arturo"          | 0.04           | 36.7                       | "Gigante naranja"              |  |

```
> MONGOSH
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides> db.Estrella.deleteOne({ Nombre: "Arturo" })
{
  acknowledged: true,
  deletedCount: 1
}
Atlas atlas-pdqv0y-shard-0 [primary] sample_guides>
```

Se verifica que el documento no se encuentra en la base

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options

ADD DATA EXPORT DATA 1 - 4 of 4

|   | _id ObjectId                       | Nombre String     | Magnitud Mixed | Distancia_(años_luz) String | Tamaño String                  |  |
|---|------------------------------------|-------------------|----------------|-----------------------------|--------------------------------|--|
| 1 | ObjectId('64d2988fc3107304e45...') | "Rigel"           | "0,12"         | "900"                       | "Supergigante azul"            |  |
| 2 | ObjectId('64d2995e4124bede4d1...') | "Sirio"           | 1              | "8,6"                       | "Enana blanca"                 |  |
| 3 | ObjectId('64d29ade4124bede4d1...') | "Canopus"         | "0,72"         | "309,8"                     | "Supergigante blanco-amarilla" |  |
| 4 | ObjectId('64d29ade4124bede4d1...') | "Alfa Centauri A" | "-0,01"        | "4,37"                      | "Enana amarilla"               |  |

Borrar uno de los documentos insertados

# Importar Documentos (CVS)

10) Crear la base de datos Clima de forma local importando un archivo cvs

Dirigirse a la pagina <https://open-meteo.com/en/docs/historical-weather-api>

The Open-Meteo Historical Weather API made available additional weather information dating back to 1940! Read the [blog article](#).

**Select Coordinates or City**

Latitude:  Longitude:

**Specify Time Interval**

Start date:  End date:

**Hourly Weather Variables**

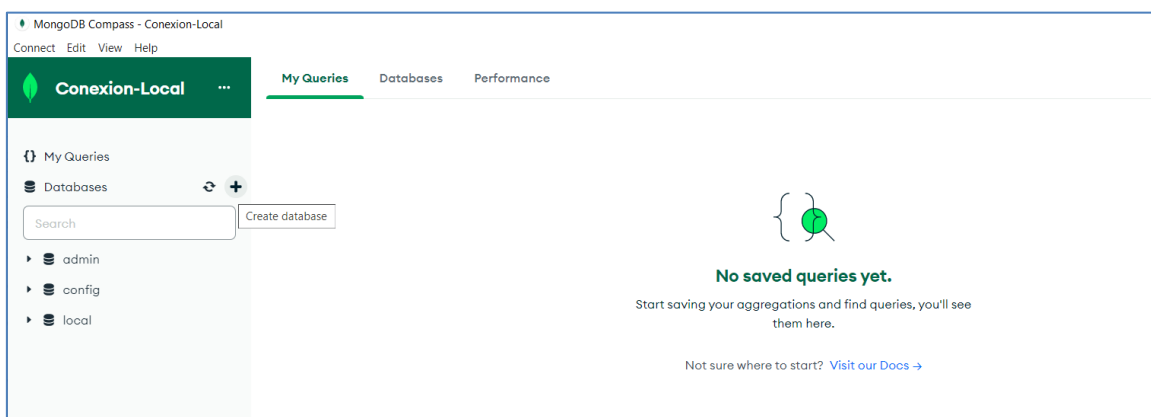
|                                                       |                                                                          |                                                        |                                                        |
|-------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------|--------------------------------------------------------|
| <input checked="" type="checkbox"/> Temperature (2 m) | <input type="checkbox"/> Weathercode                                     | <input type="checkbox"/> Wind Speed (10 m)             | <input type="checkbox"/> Soil Temperature (0-7 cm)     |
| <input type="checkbox"/> Relative Humidity (2 m)      | <input type="checkbox"/> Sealevel Pressure                               | <input checked="" type="checkbox"/> Wind Speed (100 m) | <input type="checkbox"/> Soil Temperature (7-28 cm)    |
| <input type="checkbox"/> Dewpoint (2 m)               | <input type="checkbox"/> Surface Pressure                                | <input type="checkbox"/> Wind Direction (10 m)         | <input type="checkbox"/> Soil Temperature (28-100 cm)  |
| <input type="checkbox"/> Apparent Temperature         | <input type="checkbox"/> Cloudcover Total                                | <input type="checkbox"/> Wind Direction (100 m)        | <input type="checkbox"/> Soil Temperature (100-255 cm) |
| <input type="checkbox"/> Precipitation (rain + snow)  | <input type="checkbox"/> Cloudcover Low                                  | <input type="checkbox"/> Wind Gusts (10 m)             | <input type="checkbox"/> Soil Moisture (0-7 cm)        |
| <input checked="" type="checkbox"/> Rain              | <input type="checkbox"/> Cloudcover Mid                                  |                                                        | <input type="checkbox"/> Soil Moisture (7-28 cm)       |
| <input type="checkbox"/> Snowfall                     | <input type="checkbox"/> Cloudcover High                                 |                                                        | <input type="checkbox"/> Soil Moisture (28-100 cm)     |
|                                                       | <input type="checkbox"/> Reference Evapotranspiration (ET <sub>a</sub> ) |                                                        | <input type="checkbox"/> Soil Moisture (100-255 cm)    |
|                                                       | <input type="checkbox"/> Vapor Pressure Deficit                          |                                                        |                                                        |

Elegir como locacion Buenos Aires

Seleccionar un rango de fechas de 6 meses

Descargar el archivo en formato CVS.

- Crear la base de datos llamada Clima (desde línea de comando o desde la GUI)
- Crear la colección llamada: clima\_diario (desde línea de comando o desde la GUI)
- Desde la colección importar el archivo descargado.
- Observar los tipos de datos.
- Que sucede si se importa el archivo tal cual se descargó?
- Como son los tipos de datos importados?



# Indices

## 11) Crear un índice a la colección clima diario.

Un índice posibilita el acceso directo y rápido haciendo más eficiente las búsquedas. Sin índice, MongoDB debe recorrer secuencialmente todos los documentos de una colección para encontrar un documento en particular.

En MongoDB el primer índice que se crea en forma automática corresponde al campo `_id`, recordemos que dicho campo podemos asignarle un valor nosotros o hacer que se cree automáticamente

Los índices son estructuras asociadas a las colecciones, una colección almacena los campos indexados y se crean para acelerar las consultas.

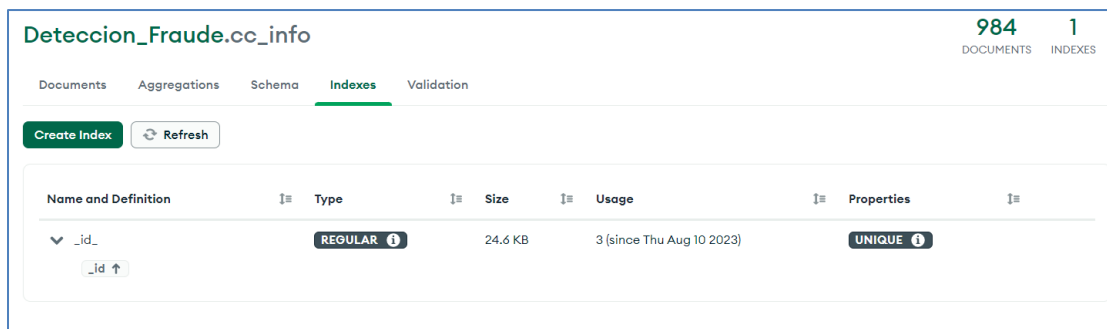
No se recomienda crear índices sobre campos que no se usan con frecuencia en consultas o en colecciones muy pequeñas.

Los cambios sobre las colecciones, como inserción, actualización o eliminación de documentos son incorporados automáticamente en los archivos índices.

La desventaja es que consume espacio en el disco y genera costo de mantenimiento (tiempo y recursos) cuando se efectúan inserciones y modificaciones.

- Crear la base de datos Deteccion\_Fraude de forma local
- Descomprimir el archivo archive.zip
- Crear una colección por cada archivo csv
- Importar el archivo

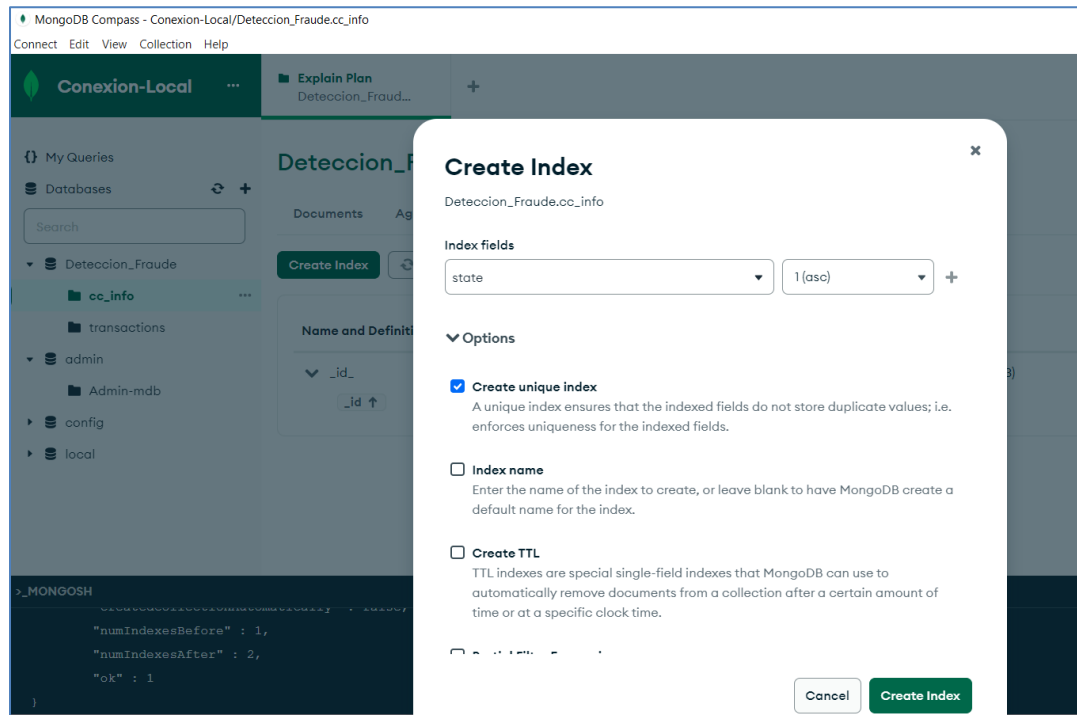
Vemos que por defecto se crea de forma automática el índice sobre el campo `id`



| Deteccion_Fraude.cc_info                                |           |         |                           |            |  | 984       | 1       |
|---------------------------------------------------------|-----------|---------|---------------------------|------------|--|-----------|---------|
|                                                         |           |         |                           |            |  | DOCUMENTS | INDEXES |
| Documents Aggregations Schema <b>Indexes</b> Validation |           |         |                           |            |  |           |         |
| Create Index Refresh                                    |           |         |                           |            |  |           |         |
| Name and Definition                                     | Type      | Size    | Usage                     | Properties |  |           |         |
| ▼ _id_                                                  | REGULAR ⓘ | 24.6 KB | 3 (since Thu Aug 10 2023) | UNIQUE ⓘ   |  |           |         |
| _id ↑                                                   |           |         |                           |            |  |           |         |

Para crear un índice

a) Por la GUI



b) Por línea de comando

Visualización de los índices existentes

`db.cc_info.getIndexes()`

```
>_MONGOSH
> db.cc_info.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { state: 1 }, name: 'state_1' }
]
Deteccion_Fraude>
```

`db.cc_info.createIndex({"city_idx": 1})`

`db.cc_info.getIndexes()`

```
>_MONGOSH
> db.cc_info.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { city: 1 }, name: 'city_idx' },
  { v: 2, key: { state: 1 }, name: 'state_1' }
]
Deteccion_Fraude>
```

## Uso del índice en búsquedas

db.cc\_info.find( { city: "Dallas" } )

Deteccion\_Fraude.cc\_info

984

1

DOCUMENTSINDEXES

Documents

Aggregations

Schema

Indexes

Validation

Create Index

Refresh

| Name and Definition | Type      | Size    | Usage                      | Properties |
|---------------------|-----------|---------|----------------------------|------------|
| > _id_              | REGULAR ⓘ | 24.6 KB | 10 (since Thu Aug 10 2023) | UNIQUE ⓘ   |
| > city_idx          | REGULAR ⓘ | 24.6 KB | 3 (since Thu Aug 10 2023)  |            |
| > state_1           | REGULAR ⓘ | 20.5 KB | 10 (since Thu Aug 10 2023) |            |

```
>_MONGOOSH
> db.cc_info.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { city: 1 }, name: 'city_idx' },
  { v: 2, key: { state: 1 }, name: 'state_1' }
]
> db.cc_info.find( { city: "Dallas" } )
< [
  {
    _id: ObjectId("64d50d6a911dd692b6a1be50"),
    credit_card: 1280981422329509,
    city: 'Dallas',
    state: 'PA',
    zipcode: 18612,
    credit_card_limit: 6000
  },
  {
    _id: ObjectId("64d50d6a911dd692b6a1be7e"),
    credit_card: 9722420012449776,
    city: 'Dallas',
    state: 'PA',
    zipcode: 18612,
    credit_card_limit: 6000
  }
]
```

Para borrar los índices: dropIndex() El método se usa para eliminar el índice único y el método dropIndexes () se usa para eliminar múltiples índices.

db.cc\_info.dropIndex("city\_idx")

```
>_MONGOOSH
> db.cc_info.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { city: 1 }, name: 'city_idx' },
  { v: 2, key: { state: 1 }, name: 'state_1' }
]
> db.cc_info.dropIndex("city_idx")
< { nIndexesWas: 3, ok: 1 }
> db.cc_info.getIndexes()
< [
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { state: 1 }, name: 'state_1' }
]
Deteccion_Fraude>
```

## Vistas

### 12) Creación de vistas

#### a) Por línea de comando

```
db.createView( "Transacciones_Estado", "transactions", [
  {
    $lookup:
    {
      from: "cc_info",
      localField: "credit_card",
      foreignField: "credit_card",
      as: "Transacciones_Estado"
    }
  },
  {
    $project:
    {
      _id: 0,
      credit_card:1,
      date:1,
      transaction_dollar_amount:1,
      city:"$Transacciones_Estado.city"
    }
  },
  { $unwind: "$city"}
] )
```

#### \$lookup

Se utiliza para "unir" documentos cuando los campos localField: y foreignField coinciden.

Los documentos coincidentes se agregan como una matriz (as:)

Es el equivalente al Join/where de sql

#### \$project

Se selecciona un subconjunto de los campos disponibles.

Es el Equivalente al Select

\_id: 0 → con cero indicamos que no vamos a utilizar ese campo (false)

credit\_card:1 → indicamos que vamos a utilizar ese campo (true)

city:"\$Transacciones\_Estado.city" → indicamos que es el campo de la tabla "foranea" por ende lo invocamos utilizando las comillas dobles y el signo pesos.

#### \$unwind

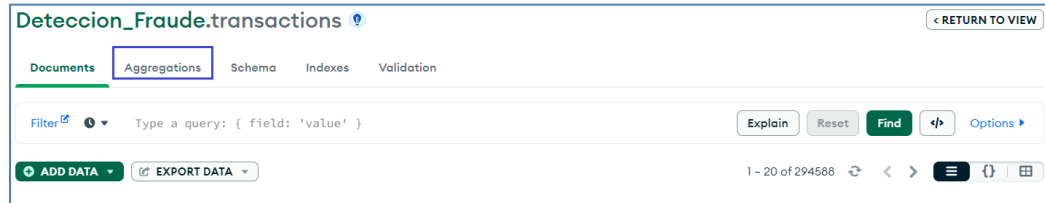
Convierte el campo de una matriz a un valor escalar.

Al traer los campos de otro documento, si no se utiliza este parametro va a crear la vista con

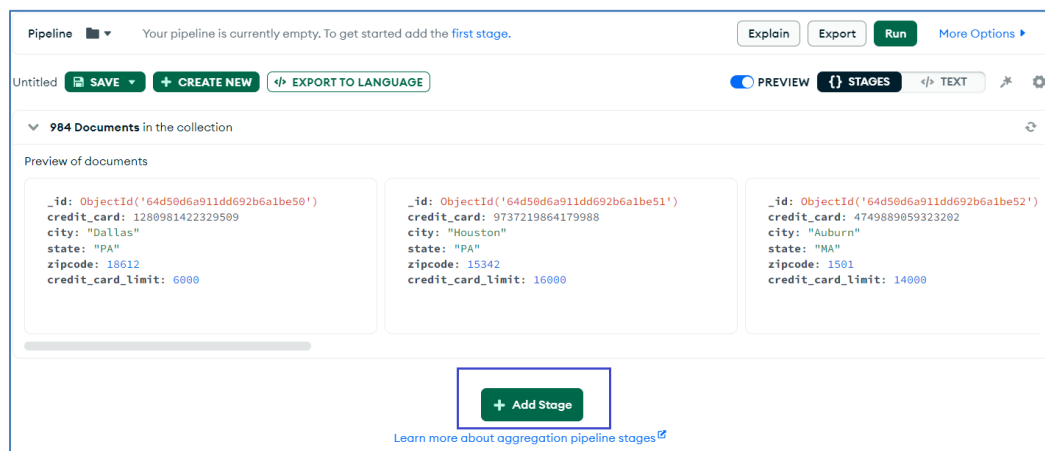
teniendo los campos de la tabla en una matriz.

b) Por la GUI

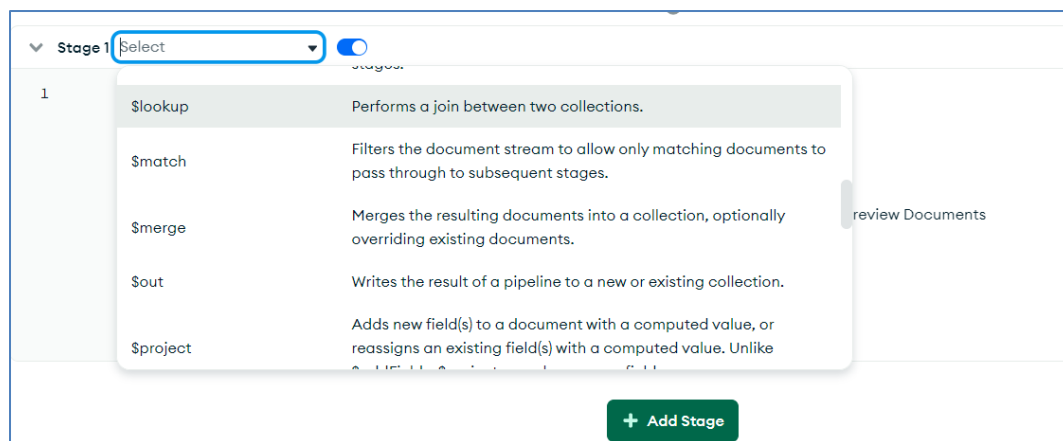
Se debe ingresar a la colección y desde allí ir a “Aggregations”



Click en Add Storage

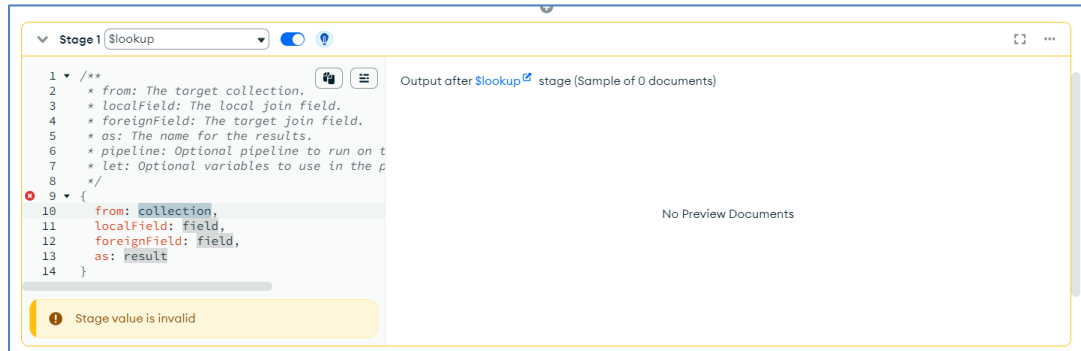


Dentro del Stage 1 seleccionar \$lookup

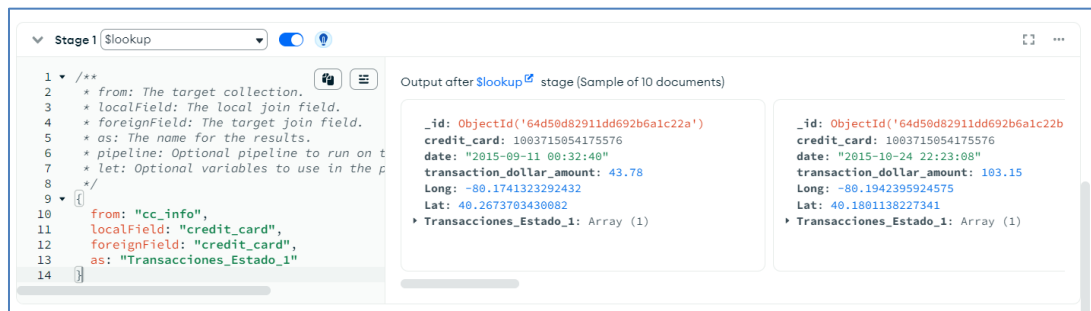




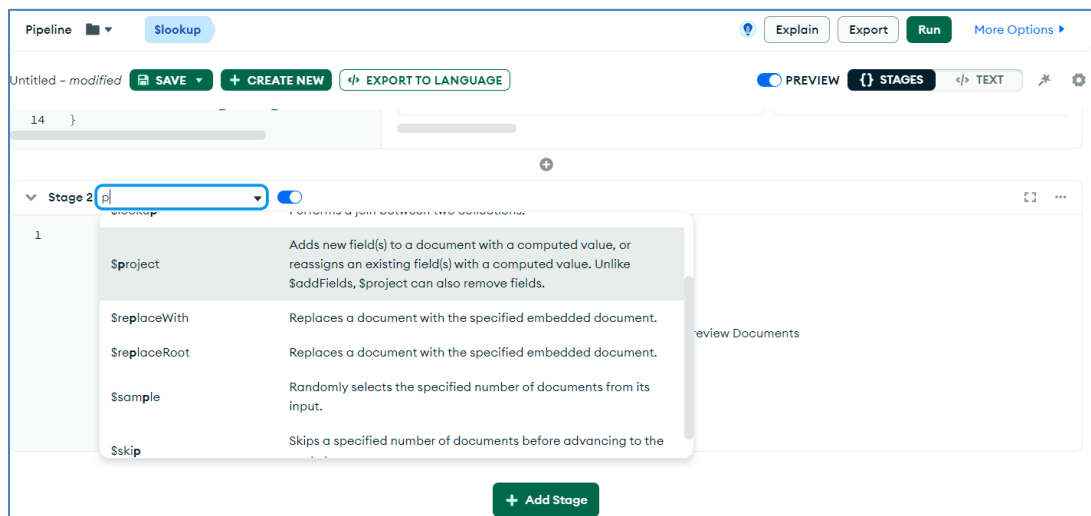
Completar los campos según corresponda con la vista a crear

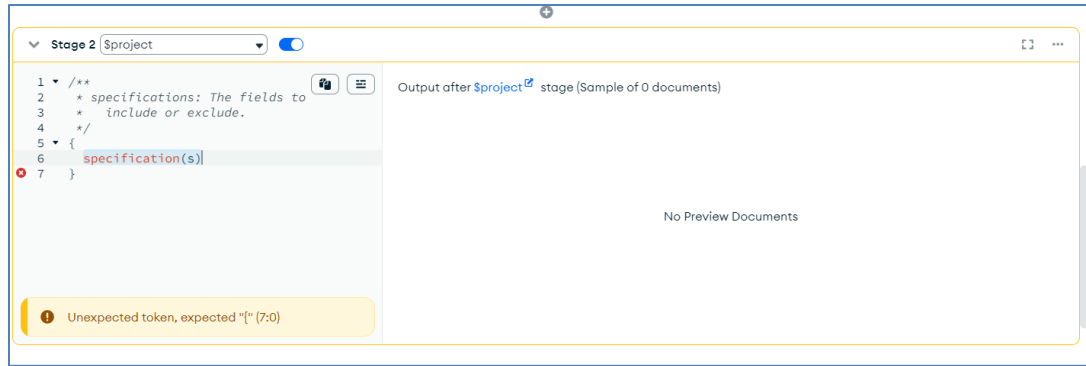


En este caso completo con los mismo campos de la vista creada anteriormente

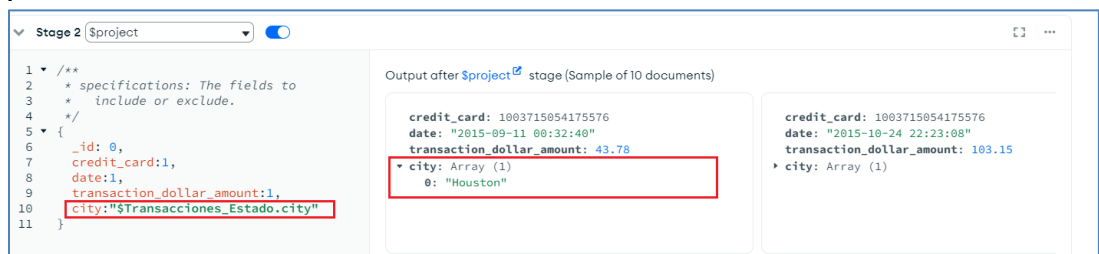


Procedemos a agregar un nuevo Stage, en este caso elegimos project

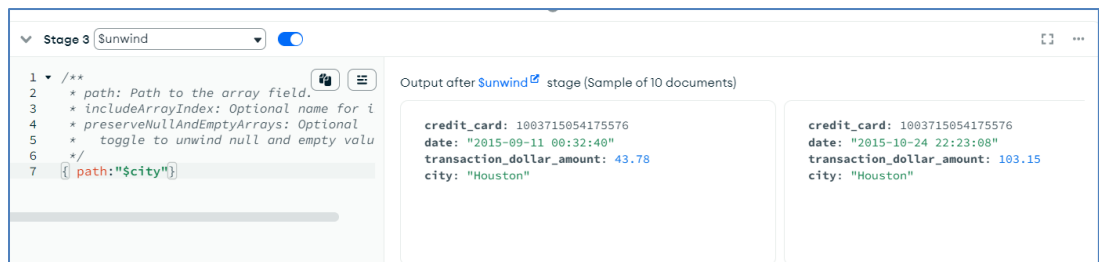




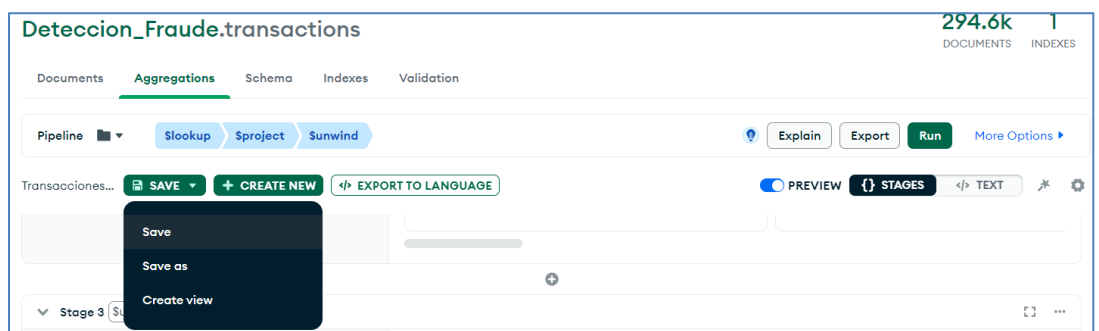
Y completamos con los campos que deseamos incluir en la vista  
 Nota: Cuando el campo no pertenece a la colección desde donde la estamos creando, el mismo se crea en forma de matriz; por ello se debe utilizar luego el parametro \$unwind



Agregamos un nuevo Stage y colocamos el campo city, tal como se muestra en la imagen  
 Ya se visualiza el campo city como un String y no como un array.



Por último guardamos la vista seleccionando Create View



# Create a View

**Name**

Transacciones\_Estado\_2

Cancel

Create

Visualizamos desde la base de datos la vista creada

My Queries

Databases

Search

Deteccion\_Fraude

Transacciones\_Estado\_1

Transacciones\_Estado\_2

cc\_info

transactions

## Deteccion\_Fraude.cc\_info

984 DOCUMENTS 2 INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' }

EXPLAIN RESET FIND Options

ADD DATA EXPORT DATA

1 - 20 of 984

```
{ "_id": ObjectId("64d58d6a911d6692b6a1be50"), "credit_card": 1288981422329589 }
```

Para ver el plan de ejecucion y las metricas del mismo, se presiona sobre el boton Explain y allí nos muestra las estadísticas.

## Explain Plan

Explain provides key execution metrics that help diagnose slow queries and optimize index usage. [Learn more](#)

Visual Tree

Raw Output

EQ\_LOOKUP

Returned 294588 Execution Time 4.3 s

PROJECTION\_DEFAULT

Returned 294588 Execution Time 1.1 s

COLLSCAN

Returned 294588 Execution Time 1.3 s

Documents Examined: 294588

### Query Performance Summary

- 294588 documents returned
- 295672 documents examined
- 10447 ms execution time
- Is not sorted in memory
- 0 index keys examined
- No index available for this query.

# Operaciones de agregación

## 13) Operaciones de agregación

### Aggregation Pipeline – Canalización

Una canalización/tubería de agregación consta de una o más etapas que procesan documentos, es la forma en que en mongodb se le da tratamiento a los documentos/colecciones.

Cada etapa (stage) realiza una operación sobre los documentos de entrada. Por ejemplo, una etapa puede filtrar documentos, agrupar documentos y calcular valores.

Los documentos que salen de una etapa pasan a la etapa siguiente.

Una canalización de agregación puede devolver resultados para grupos de documentos. Por ejemplo, devuelva los valores total, promedio, máximo y mínimo.

Las operaciones de agregación se utilizan mayoritariamente para:

- Agrupar valores de varios documentos.
- Procesamiento y operaciones matemáticas.
- Analizar cambios de datos a lo largo del tiempo.

Las operaciones aggregation pueden guardarse como consultas (queries) o guardarse como vistas.

Nota: utilizar la base Deteccion\_Fraude para las siguientes actividades

### Agrupar valores:

\$match: funciona de forma similar al método find() o el filter. Nos permite filtrar documentos de la colección según la consulta que pasemos como parámetro.

```
db.cc_info.aggregate([{$match : {city: "Dallas" }}])
```

```
>_MONGOOSH
> db.cc_info.aggregate([{$match : {city: "Dallas" }}])
< {
  _id: ObjectId("64d58d6a91dd692b6a1be50"),
  credit_card: 1280981422329509,
  city: 'Dallas',
  state: 'PA',
  {
    _id: ObjectId("64d58d6a91dd692b6a1be7e"),
    credit_card: 9722428012449776,
    city: 'Dallas',
    state: 'PA',
    zipcode: 18612,
    credit_card_limit: 6000
  }
  {
    _id: ObjectId("64d58d6a91dd692b6a1beb3"),
    credit_card: 4290726060535834,
    city: 'Dallas',
    state: 'PA',
    zipcode: 18612,
    credit_card_limit: 10000
  }
}
```

### Procesamiento y operaciones matemáticas:

\$sum: Permite acumular los valores pasamos como parámetro

Sumamos el total de gastos realizados por una tarjeta de credito, agrupados por estado origen de la tarjeta

```
b.Transacciones_Estado_1.aggregate([{$group: {_id: "$city", total: { $sum: "$transaction_dollar_amount" } }}])
```

```
>_MONGOOSH
> db.Transacciones_Estado_1.aggregate([
  {
    $group: {
      _id: "$city",
      total: { $sum: "$transaction_dollar_amount" }
    }
  }
])
< {
  _id: 'Quitman',
  total: 31010.07
}
{
  _id: 'Roanoke',
  total: 2137.7200000000003
}
{
  _id: 'Akron',
  total: 101845.37
}
{
  _id: 'Somerset',
  total: 8771.03
}
```

O tambien se puede utilizar como contador: Contar la cantidad de transacciones por estado

```
db.Transacciones_Estado_1.aggregate([
  {
    $group: {
      _id: "$city",
      totalTransacciones: { $sum: 1 }
    }
  }
])
```

```
>_MONGOSH
< {
  _id: 'Somerset',
  totalTransacciones: 84
}
{
  _id: 'Topeka',
  totalTransacciones: 672
}
{
  _id: 'Roanoke',
  totalTransacciones: 12
}
{
  _id: 'Quitman',
  totalTransacciones: 584
}
{
  _id: 'Akron',
  totalTransacciones: 1447
}
```

Puede realizarse por la GUI, se muestra la siguiente imagen a modo de ejemplo:

The screenshot shows the MongoDB Compass interface for the 'Deteccion\_Fraude.Transacciones\_Estado\_1' collection. The 'Aggregations' tab is active, displaying a single stage named '\$group'. The pipeline editor shows the following JSON:

```
1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5 {
6   _id: "$city",
7   totalTransacciones: {
8     $sum: 1
9   }
10 }
```

Below the editor, the 'Output after \$group stage (Sample of 10 documents)' is displayed in a table:

| Document                                         |
|--------------------------------------------------|
| { "_id": "Buffalo", "totalTransacciones": 1012 } |
| { "_id": "Auburn", "totalTransacciones": 43 }    |

The interface includes various controls such as 'Pipeline', 'Stage 1', 'Preview', 'Export', and 'Run' buttons.

\$max, \$min

Permite obtener el valor máximo y el valor mínimo de la expresión que se evalúa para cada documento generado por el agrupamiento

```
db.Transacciones_Estado_1.aggregate([
  {
    $group : { _id: 'idAux',
               maximo: {$max: '$transaction_dollar_amount'},
               minimo: {$min: '$transaction_dollar_amount'}
            }
  }
])
```

```
>_MONGOSH
> db.Transacciones_Estado_1.aggregate([{$group : { _id: 'idAux', maximo: {$max: '$transaction_dollar_amount'}, minimo: {$min: '$transaction_dollar_amount'}}}]);
< {
  _id: 'idAux',
  maximo: 999.97,
  minimo: 0.01
}
Deteccion_Fraude>
```

Se pueden combinar los parametros vistos, según la necesidad de obtener datos:

Ejemplo: obtener la suma total de transacciones dado un estado

```
db.Transacciones_Estado_1.aggregate( [
  {
    $match: { city: "Houston" }
  },
  {
    $group: { _id: "$city", total: { $sum: "$transaction_dollar_amount" } }
  }
])
```

```
>_MONGOSH
> db.Transacciones_Estado_1.aggregate( [
  {
    $match: { city: "Houston" }
  },
  {
    $group: { _id: "$city", total: { $sum: "$transaction_dollar_amount" } }
  }
] )
< {
  _id: 'Houston',
  total: 1809789.05
}
Deteccion_Fraude>
```

Operadores comparativos:

\$gt, \$gte, \$lt, \$lte:

Estos operadores nos permiten comparar desigualdades, \$gt y \$gte para “mayor que” y “mayor o igual que” respectivamente, y \$lt y \$lte para “menor que” y “menor o igual que” respectivamente.

Retornar true cuando se cumple el criterio y false en caso contrario

```
db.Transacciones_Estado_1.find({
  transaction_dollar_amount: { $gte: 500, $lte: 700 }
})
```

```
>_MONGOSH
> db.Transacciones_Estado_1.find({
  transaction_dollar_amount: { $gte: 500, $lte: 700 }
})
< {
  credit_card: 4832198644199161,
  date: '2015-09-13 22:05:27',
  transaction_dollar_amount: 543.63,
  city: 'Washington'
}
{
  credit_card: 5644143320838397,
  date: '2015-09-21 19:59:59',
  transaction_dollar_amount: 541.47,
  city: 'Denver'
}
```

\$eq, \$ne:

Estos operadores nos permiten comparar la igualdad o diferencia de dos expresiones que le pasamos como parámetro.

\$eq retorna true cuando los valores son equivalentes y false cuando no lo son, \$ne hace lo opuesto.

```
db.Transacciones_Estado_1.find({ transaction_dollar_amount: { $eq : 0.01 } })
```

```
>_MONGOSH
> db.Transacciones_Estado_1.find({ transaction_dollar_amount: { $eq : 0.01 } })
< {
  credit_card: 1013870087888817,
  date: '2015-10-23 18:06:25',
  transaction_dollar_amount: 0.01,
  city: 'Washington'
}
{
  credit_card: 1144894232607400,
  date: '2015-10-29 17:48:04',
  transaction_dollar_amount: 0.01,
  city: 'Louisville'
}
```



\$cmp:

Toma dos valores en una matriz y devuelve un número entero. El valor devuelto es:

Un número negativo si el primer valor es menor que el segundo.

Un número positivo si el primer valor es mayor que el segundo.

0 si los dos valores son iguales.

```
db.Transacciones_Estado_1.aggregate([
  {$match: {
    credit_card: 9383760568579492,
    date: "2015-08-16 19:02:57"
  }},
  {$project:
    {result: {$cmp:["transaction_dollar_amount", 43.78]}}
}]
```

```
>_MONGOSH

> db.Transacciones_Estado_1.aggregate([
  {$match: {
    credit_card: 9383760568579492,
    date: "2015-08-16 19:02:57"
  }},
  {$project:
    {result: {$cmp:["transaction_dollar_amount", 43.78]}}
  }
])
< {
  result: 1
}

Deteccion_Fraude>
```

Pipeline ▾

\$match

\$project

?

 Explain

Export

Run

More Options ▾

itled - modified 

SAVE

+ CREATE NEW

EXPORT TO LANGUAGE

PREVIEW

STAGES

TEXT

⚙

1 ▾ [

2 ▾ {

3     \$match:

4         /\*\*

5         \* query: The query in MQL.

6         \*/

7         {

8             credit\_card: 9383760568579492,

9             date: "2015-08-16 19:02:57",

10         },

11     },

12     {

13         \$project:

14             /\*\*

15             \* specifications: The fields to

16             \* include or exclude.

17             \*/

18             {

19                 result: {

20                     \$cmp: [

21                         "transaction\_dollar\_amount",

22                         43.78,

23                     ],

24                 },

PIPELINE OUTPUT

Sample of 1 document

result: 1

OUTPUT OPTIONS ▾

## Practica adicional

Usando la base clima

- Ingresar el campo Mes, en el cual se indique el nombre del mes según la fecha indicada en el documento
- Ingresar el campo ciudad, colocarle el dato Buenos Aires para todos los documentos
- Descargar un nuevo archivo con los mismos campos, y el mismo rango de fechas para la localidad Montevideo
- Crear una vista con el promedio de los campos temperature\_2m (°C), rain (mm), windspeed\_100m (km/h) por día
- Obtener el máximo y el mínimo de temperatura diaria promedio
- Obtener la/s fechas en los que se produjo temperatura diaria promedio
- Obtener para un día determinado (fecha a elección) las condiciones meteorológicas de Buenos Aires y Montevideo

## Documentación complementaria

14) Documentación de los comandos disponibles en Mongo DB

<https://www.mongodb.com/docs/manual/reference/limits/#std-label-restrictions-on-db-names>

<https://www.mongodb.com/docs/mongodb-shell/crud/#std-label-mdb-shell-crud>

<https://www.mongodb.com/docs/manual/reference/method/>

<https://www.mongodb.com/docs/manual/aggregation/>

<https://www.youtube.com/watch?v=6weTUQVtiD8>