



<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security



main ▾

ud2-practica-2-mongo-db-Dansarasix-DML / Readme.md



Dansarasix-DML update img

a3e9b4c · 8 months ago



242 lines (177 loc) · 7.84 KB

Preview

Code

Blame



Raw



Big Data Aplicado

UD2 - Procesado y Presentación de Datos Almacenados

🔗 Práctica 2 MongoDB

Recuerda que para hacer la prácticas puedes optar por cualquiera de la las 3 opciones.

- Instalarla en tu máquina local
 - Usar Atlas MongoDB: <https://www.mongodb.com/docs/atlas/getting-started/>
 - Crear un contenedor docker.
-
- **Ejercicio 1:** Crea una base de datos con MongoDB llamada "db_pract2", así como una colección con nombre "Organizaciones" en la que importes el contenido del archivo dataset_organizaciones.json

Creamos la base de datos como en el ejercicio anterior e importamos la data.

Create Database

Database Name
db_pract2

Collection Name
Organizaciones

☐ **Time-Series**
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> Additional preferences (e.g. Custom collation, Capped, Clustered collections)

Cancel

Create Database

Type a query: { field: 'value' } or [Generate query](#)

Explain

Reset

Find

</>

Options

+ ADD DATA

EXPORT DATA

UPDATE

DELETE

25 1 - 25 of 163

⋮

{ }

⌂

```
_id: ObjectId('6359a54adafc61784453764c')
datasetid: "directorio-organizaciones-profesionales-comercio"
recordid: "dc382c46532d6ede99115048311fa8b8f06e2d12"
fields: Object
geometry: Object
record_timestamp: "2021-09-07T11:07:00.513+02:00"
```

```
_id: ObjectId('6359a54adafc61784453764d')
datasetid: "directorio-organizaciones-profesionales-comercio"
recordid: "cdbafd42ebf5c714e17756ff190fd8fc03194ffa"
fields: Object
geometry: Object
record_timestamp: "2021-09-07T11:07:00.513+02:00"
```

- **Ejercicio 2:** Sobre la base de datos "db_pract2" realizar las siguientes consultas sobre la colección "Organizaciones":
 - Obtén el número total de documentos que se han importado a la colección Organizaciones.

Como antes, hacemos `countDocuments()` para obtener el número de documentos que se han importado.

```
> db["Organizaciones"].countDocuments()
< 163
```

```
db["Organizaciones"].countDocuments()
```



- o Muestra los datos de todas las organizaciones que se encuentran en la provincia de Burgos. ¿Cuántas organizaciones podemos encontrar ubicadas en la capital de la provincia?

Primero hacemos `$match` para filtrar todas las organizaciones que estén en la provincia de Burgos.

Stage 1 `$match`

```
1 /**
2  * query: The query in MQL.
3  */
4  {
5    "fields.provincia" : { $eq: "BURGOS" }
6  }
```

Output after `$match` stage (Sample of 10 documents)

```
{
  "_id": ObjectId('6359a54adafc61784453764c'),
  "datasetid": "directorio-organizaciones-
    profesionales-comercio",
  "recordid": "dc382c46532d6ede99115048311fa8b8f0...",
  "fields": {
    "fax": "947509212",
    "provincia": "BURGOS",
    "codigo_postal": "09400",
    "prov": "BU"
  }
}
```

Y luego hacemos un `$count` para sacar la cantidad del `$match` anterior.

Stage 2 `$count`

```
1 /**
2  * Provide the field name for the count.
3  */
4  'organizaciones'
```

Output after `$count` stage (Sample of 1 document)

```
{
  "organizaciones": 46
}
```

```
db.getCollection('Organizaciones').aggregate(
[
  {
    $match: {
      'fields.provincia': { $eq: 'BURGOS' }
    }
  },
  { $count: 'organizaciones' }
],
{ maxTimeMS: 60000, allowDiskUse: true }
);
```



También se puede hacer con `find()` :

```
db["Organizaciones"].find({"fields.provincia":"BURGOS"}).count()
```



```
> db["Organizaciones"].find({"fields.provincia":"BURGOS"}).count()
< 46
```

- Busca todas las organizaciones que se encuentren en la ciudad de Ponferrada cuyo email de contacto pertenezca al dominio Gmail.

Volvemos a hacer un `$match` sobre la localidad para filtrar por Ponferrada.

Stage 1 `$match`

```
1 /**
2  * query: The query in MQL.
3  */
4  {
5    "fields.localidad" : { $eq: "PONFERRADA"
6  }
```

Output after `$match` stage (Sample of 3 documents)

```
provincia : "LEÓN"
codigo_postal : "24401"
prov : "LE"
coordenadas : Array (2)
direccion : "Paseo de San Antonio, 3"
no_de_registro : "LE-FLE001-167"
localidad : "PONFERRADA"
nombre : "ASOCIACIÓN DE COMERCIANTES DE
          PONFERRADA"
telefono : "987417551"
```

```
_id: ObjectId(
datasetid : "d
pr
recordid : "e2
fields : Objec
geometry : Objec
record_timestamp
```

Volvemos a hacer otro `$match` para filtrar por el correo usando una expresión regular que busque solo los que sean gmail.

Stage 2 `$match`

```
1 /**
2  * query: The query in MQL.
3  */
4  {
5    "fields.email" : {
6      $regex: "@gmail\\.com$",
7      $options: "i"
8    }
9  }
```

Output after `$match` stage (Sample of 1 document)

```
_id: ObjectId('6359a54adafc6178445376a6')
datasetid : "directorio-organizaciones-profesionales-comercio"
recordid : "d2188a3e064f7611ca68edbf8cb6134f592f..."
fields : Object
geometry : Object
record_timestamp : "2021-09-07T11:07:00.513+02:00"
```

Y el resultado es el siguiente:

ALL RESULTS OUTPUT OPTIONS

Showing 1 - 1 count results < > VIEW

```
_id: ObjectId('6359a54adafc6178445376a6')
datasetid : "directorio-organizaciones-profesionales-comercio"
recordid : "d2188a3e064f7611ca68edbf8cb6134f592fc71a"
fields : Object
fax : "987423575"
provincia : "LEÓN"
codigo_postal : "24402"
prov : "LE"
coordenadas : Array (2)
direccion : "Avda. España, 38 - 1ª dcha"
no_de_registro : "LE-000-138"
localidad : "PONFERRADA"
nombre : "ASOCIACION LOCAL DE EMPRESAS CENTRO URBANO " TEMPLARIUM""
telefono : "987423551"
email : "templariumponferrada@gmail.com"
geometry : Object
record_timestamp : "2021-09-07T11:07:00.513+02:00"
```

```
db.getCollection('Organizaciones').aggregate(
[
```



```

    {
      $match: {
        'fields.localidad': { $eq: 'PONFERRADA' }
      }
    },
    {
      $match: {
        'fields.email': {
          $regex: '@gmail\\.com$',
          $options: 'i'
        }
      }
    }
  ],
  { maxTimeMS: 60000, allowDiskUse: true }
);

```

También se puede simplificar en un solo `$match` :

```

db.getCollection('Organizaciones').aggregate(
[
  {
    $match: {
      'fields.localidad': { $eq: 'PONFERRADA' },
      'fields.email': {
        $regex: '@gmail\\.com$',
        $options: 'i'
      }
    }
  }
],
{ maxTimeMS: 60000, allowDiskUse: true }
);

```

- Extrae todas las organizaciones que se encuentren en la provincia de Valladolid o Soria.

Hacemos un `find()` y ponemos un `$or` sobre los valores de la provincia para que saque los de Valladolid y Soria.

```

db["Organizaciones"].find({ $or: [
  {"fields.provincia":"VALLALODID"},
  {"fields.provincia":"SORIA"}
]})

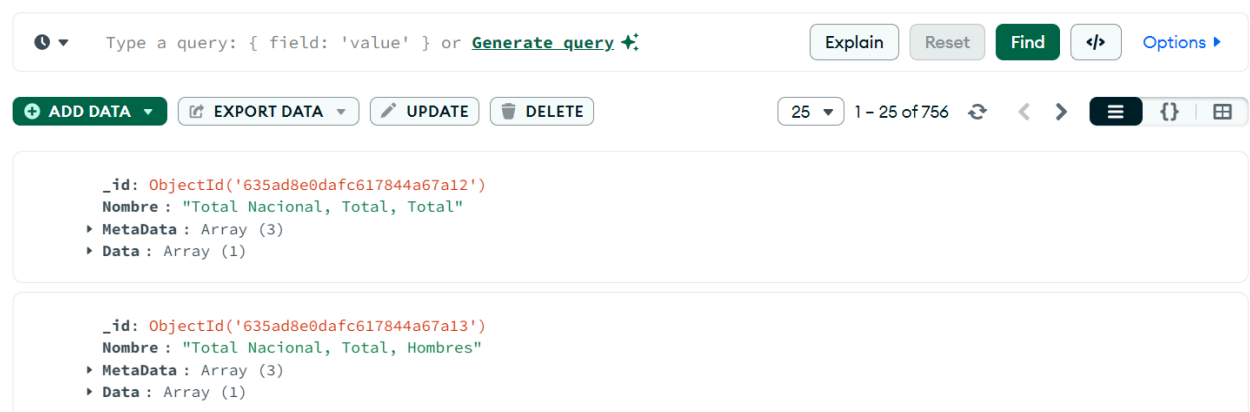
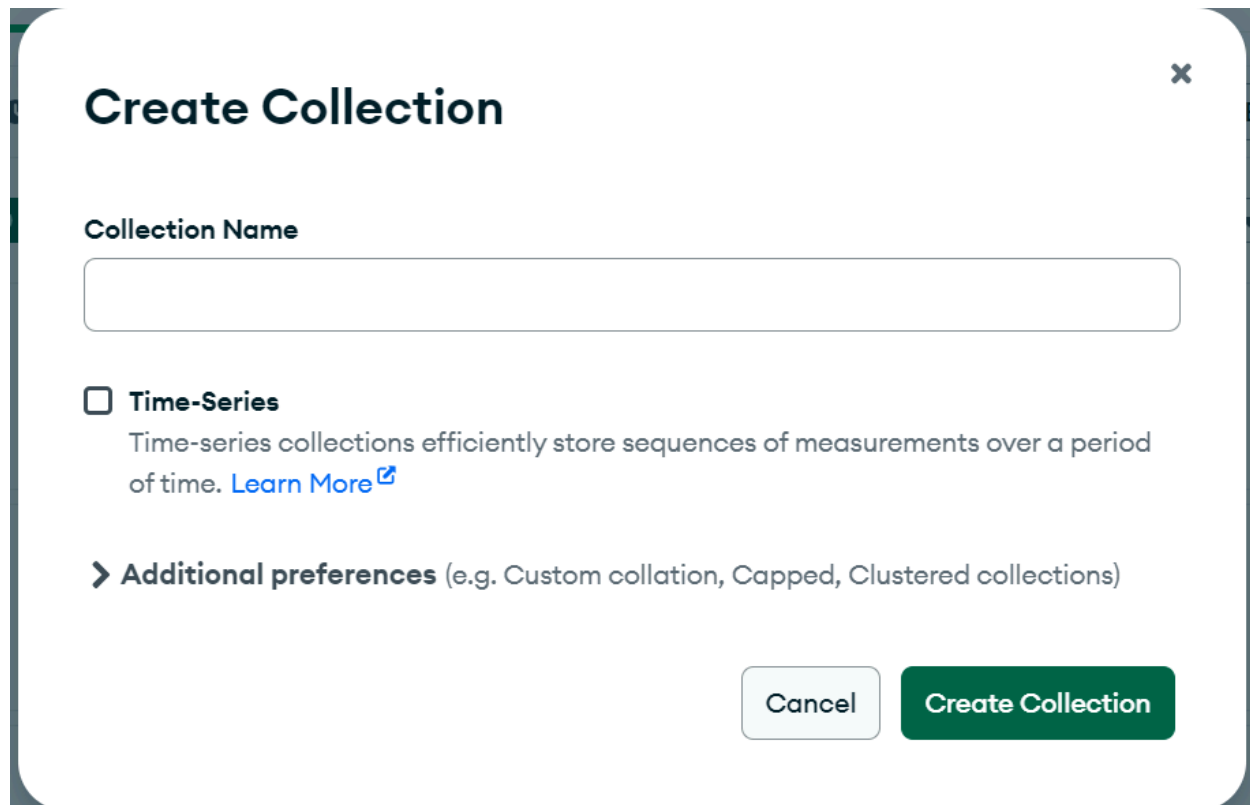
```

Si hacemos `count()` saldrá que la consulta ha devuelto 50 organizaciones que están en Valladolid o en Soria.

```
> db["Organizaciones"].find({ $or: [
  {"fields.provincia":"VALLADOLID"}, {"fields.provincia":"SORIA"}
]).count()
< 50
```

- **Ejercicio 3:** Añade a la base de datos "db_pract2" una colección con nombre "Veterinarios" en la que importes el contenido del archivo dataset_veterinarios.json

Como cuando creamos una base de datos, también se puede agregar otra colección. Creamos una nueva colección y le pasamos la data deseada.



```
_id: ObjectId('635ad8e0dafc617844a67a12')
Nombre: "Total Nacional, Total, Total"
Metadata: Array (3)
Data: Array (1)

_id: ObjectId('635ad8e0dafc617844a67a13')
Nombre: "Total Nacional, Total, Hombres"
Metadata: Array (3)
Data: Array (1)
```

- **Ejercicio 4.** Sobre la colección "Veterinarios" realizar las siguientes consultas:
 - Obtén el número total de veterinarios no jubilados en Andalucía.

Hacemos una agregación donde primero un `$match`, para filtrar por nombre de Andalucía y por Colegiados no jubilados.

Stage 1 \$match

```

1 /**
2  * query: The query in MQL.
3  */
4 {
5   "Metadata.0.Nombre": "Andalucía",
6   "Metadata.1.Nombre": "Colegiados no jub
7 }

```

Output after \$match stage (Sample of 3 documents)

```

_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
      Total"
Metadata: Array (3)
Data: Array (1)

```

```

_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
      Total"
Metadata: Array (3)
Data: Array (1)

```

Luego hacemos un \$unwind sobre el array Data y en un \$group sin agrupar hacemos una suma de los valores en el array Data.

Stage 2 \$unwind

```

1 /**
2  * path: Path to the array field.
3  * includeArrayIndex: Optional name for i
4  * preserveNullAndEmptyArrays: Optional
5  * toggle to unwind null and empty valu
6  */
7 {
8   path: "$Data"
9 }

```

Output after \$unwind stage (Sample of 3 documents)

```

_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
      Total"
Metadata: Array (3)
Data: Object

```

```

_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
      Total"
Metadata: Array (3)
Data: Object

```

Stage 3 \$group

```

1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5 {
6   _id: null,
7   totalVeterinarios: { $sum: "$Data.Valo
8 }

```

Output after \$group stage (Sample of 1 document)

```

_id: null
totalVeterinarios: 8782

```

```

db.getCollection('Veterinarios').aggregate(
[
  {
    $match: {
      'Metadata.0.Nombre': 'Andalucía',
      'Metadata.1.Nombre':
        'Colegiados no jubilados'
    }
  },
  { $unwind: { path: '$Data' } },
  {
    $group: {
      _id: null,
      totalVeterinarios: { $sum: '$Data.Valor' }
    }
  }
],

```



```
{ maxTimeMS: 60000, allowDiskUse: true }
);
```

- Extrae un listado de comunidades autónomas, ordenando los resultados de forma descendente y usando como campo de ordenación el número de colegiados no jubilados (campo Data).

Debemos usar el campo Nombre de la Variable en el campo 0 del MetaData que debe ser "Comunidades y Ciudades Autónomas" y el Nombre del campo 1 del MetaData que será "Colegiados no jubilados". Lo haremos con `$match`

▼ Stage 1 `$match` ☒

```
1 1 /**
2 2  * query: The query in MQL.
3 3  */
4 4  {
5 5  "MetaData.0.Variable.Nombre": "Comunida
6 6  'MetaData.1.Nombre': "Colegiados no jub
7 7  }
```

Output after `$match` stage (Sample of 10 documents)

```
_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
Total"
▶ MetaData: Array (3)
▶ Data: Array (1)
```

```
_id: ObjectId(
Nombre: "Anda
Hombre
▶ MetaData: Arra
▶ Data: Array (:
```

Luego hacemos un `$unwind` sobre el array Data.

▼ Stage 2 `$unwind` ☒

```
1 1 /**
2 2  * path: Path to the array field.
3 3  * includeArrayIndex: Optional name for i
4 4  * preserveNullAndEmptyArrays: Optional
5 5  * toggle to unwind null and empty valu
6 6  */
7 7  {
8 8  path: "$Data"
9 9  }
```

Output after `$unwind` stage (Sample of 10 documents)

```
_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
Total"
▶ MetaData: Array (3)
▶ Data: Object
```

```
_id: ObjectId(
Nombre: "Anda
Hombre
▶ MetaData: Arra
▶ Data: Object
```

Y por último, hacemos `$sort` para ordenar usando Data.Valor.

▼ Stage 3 `$sort` ☒

```
1 1 /**
2 2  * Provide any number of field/order pair
3 3  */
4 4  {
5 5  "Data.Valor": -1
6 6  }
```

Output after `$sort` stage (Sample of 10 documents)

```
_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados,
Total"
▶ MetaData: Array (3)
▶ Data: Object
```

```
_id: ObjectId(
Nombre: "Cata
Total'
▶ MetaData: Arra
▶ Data: Object
```


ALL RESULTS OUTPUT OPTIONS

Showing 1 - 20 count results < > VIEW {}

```

_id: ObjectId('635ad8e0dafc617844a67a21')
Nombre: "Andalucía, Colegiados no jubilados, Total"
▶ Metadata: Array (3)
▼ Data: Object
  Valor: 4391

```

```

_id: ObjectId('635ad8e0dafc617844a67bc5')
Nombre: "Cataluña, Colegiados no jubilados, Total"
▶ Metadata: Array (3)
▼ Data: Object
  Valor: 3960

```

```

_id: ObjectId('635ad8e0dafc617844a67c91')
Nombre: "Madrid, Comunidad de, Colegiados no jubilados, Total"
▶ Metadata: Array (3)
▼ Data: Object
  Valor: 3592

```

```

db.getCollection('Veterinarios').aggregate(
[
  {
    $match: {
      'Metadata.0.Variable.Nombre':
        'Comunidades y Ciudades Autónomas',
      'Metadata.1.Nombre':
        'Colegiados no jubilados'
    }
  },
  { $unwind: { path: '$Data' } },
  { $sort: { 'Data.Valor': -1 } }
],
{ maxTimeMS: 60000, allowDiskUse: true }
);

```



- Muestra un listado de provincias con más de 50 profesionales en activo. Deberá aparecer ordenado de forma ascendente

Primero hacemos el `$match` como antes pero para buscar provincias y luego un `$unwind` sobre el array `Data`.

Stage 1 \$match

```

1  ▼
2  -y: The query in MQL.
3
4  ▼
5  >Data.0.Variable.Nombre": "Provincias",
6  >Data.1.Nombre": "Colegiados no jubilados"
7

```

Output after \$match stage (Sample of 10 documents)

```

_id: ObjectId('635ad8e0dafc617844a67a2d')
Nombre: "Almería, Colegiados no jubilados, Total"
▶ Metadata: Array (3)
▶ Data: Array (1)

```

```

_id: ObjectId('635ad8e0dafc617844a67a2e')
Nombre: "Almería, Colegiados no jubilados, Total"
▶ Metadata: Array (3)
▶ Data: Array (1)

```

Stage 2 \$unwind

```

1 /**
2  * path: Path to the array field.
3  * includeArrayIndex: Optional name for i
4  * preserveNullAndEmptyArrays: Optional
5  * toggle to unwind null and empty valu
6  */
7 {
8   path: "$Data"
9 }

```

Output after \$unwind stage (Sample of 10 documents)

```

_id: ObjectId('635ad8e0dafc617844a67a2d')
Nombre: "Almería, Colegiados no jubilados, Total"
  ▶ Metadata: Array (3)
  ▶ Data: Object

_id: ObjectId('635ad8e0dafc617844a67a2e')
Nombre: "Almería, Colegiados no jubilados, Total"
  ▶ Metadata: Array (3)
  ▶ Data: Object

```

Luego sobre Data.Valor hacemos un \$match con un \$gt para sacar aquellos valores más grandes que 50 y ordenamos con \$sort .

Stage 3 \$match

```

1 /**
2  * query: The query in MQL.
3  */
4 {
5   "Data.Valor": { $gt: 50 }
6 }

```

Output after \$match stage (Sample of 10 documents)

```

_id: ObjectId('635ad8e0dafc617844a67a2d')
Nombre: "Almería, Colegiados no jubilados, Total"
  ▶ Metadata: Array (3)
  ▶ Data: Object

_id: ObjectId('635ad8e0dafc617844a67a2e')
Nombre: "Almería, Colegiados no jubilados, Total"
  ▶ Metadata: Array (3)
  ▶ Data: Object

```

Stage 4 \$sort

```

1 /**
2  * Provide any number of field/order pair
3  */
4 {
5   "Data.Valor": 1
6 }

```

Output after \$sort stage (Sample of 10 documents)

```

_id: ObjectId('635ad8e0dafc617844a67cc2')
Nombre: "Araba/Álava, Colegiados no jubilados, Hombres"
  ▶ Metadata: Array (3)
  ▶ Data: Object
    Valor: 52

_id: ObjectId('635ad8e0dafc617844a67cc3')
Nombre: "Cuenca, Colegiados no jubilados, Mujeres"
  ▶ Metadata: Array (3)
  ▶ Data: Object
    Valor: 64

```

```

_id: ObjectId('635ad8e0dafc617844a67cc2')
Nombre: "Araba/Álava, Colegiados no jubilados, Hombres"
  ▶ Metadata: Array (3)
  ▶ Data: Object
    Valor: 52

```

```

_id: ObjectId('635ad8e0dafc617844a67ba2')
Nombre: "Cuenca, Colegiados no jubilados, Hombres"
  ▶ Metadata: Array (3)
  ▶ Data: Object
    Valor: 60

```

```

_id: ObjectId('635ad8e0dafc617844a67ba3')
Nombre: "Cuenca, Colegiados no jubilados, Mujeres"
  ▶ Metadata: Array (3)
  ▶ Data: Object
    Valor: 64

```

```

db.getCollection('Veterinarios').aggregate(
[
{
  $match: {
    'Metadata.0.Variable.Nombre':
      'Provincias',
    'Metadata.1.Nombre':
      'Colegiados no jubilados'
  }
}
]
)

```



```
    }  
  },  
  { $unwind: { path: '$Data' } },  
  { $match: { 'Data.Valor': { $gt: 50 } } },  
  { $sort: { 'Data.Valor': 1 } }  
],  
{ maxTimeMS: 60000, allowDiskUse: true }  
);
```