

# SBD02-02

## Practica Consultas en MongoDB

DNI: 39465322J

Alumno : Bieito Sousa Barreiro

Url Entrega : [Consultas en MongoDB](#)

- SBD02-02

- Practica Consultas en MongoDB

- Objetivo

- Entorno

- install docker

- docker imagen mongodb

- Instrucciones de entrega

- Enunciado

- 1.- Importa en una colección llamada “bieitolibros” el contenido del archivo books.json

- 2.- Importa en una colección llamada “bieitoventas” el contenido del archivo sales.json

- 3.- Usando la colección libros, escribe la consulta que necesitas para obtener el documento con \_id igual a 287.

- 4.- Usando la colección libros, escribe una consulta que devuelva los libros con exactamente 500 páginas

- 5.- Usando la colección libros, escribe la consulta que devuelva los libros escritos por “Robi Sen” o por “Chris King”

- 6.- Explica con tus palabras qué hace la siguiente consulta:

- ```
db.javilibros.find ( { “_id” : { $gt : 10 }, “_id” : { $lt : 15 } } )
```

- 7.- Usando la colección libros, escribe la consulta que devuelva los libros publicados en enero del 2.011

- 8.- Usando la colección libros, escribe la consulta que devuelve todos los libros en cuyo título podemos encontrar la palabra “Microsoft”

- 9.- Usando la colección libros, escribe una consulta que indique cuantos libros en la categoría de “Java” se han publicado antes del año 2.000.

- 10.- Usando la colección libros, escribe una consulta que indique solamente el título y la fecha de publicación del libro más recientemente publicado.

- 11.- En la colección ventas, explica dónde está el problema en la segunda consulta de agregados de esta captura

- indica el error de la siguiente consulta

- Explicación

- Corrección

- 12.- En la colección de ventas, indica la consulta que usarías para obtener un listado en el que aparezca cuantas unidades de cada elemento se han vendido en cada tienda y que aparezca ordenado alfabéticamente por tienda y

descendentemente por número de elementos vendidos. Debería quedar como en esta captura.

- Q12 Resolución
  - Q12 vemos la estructura del primer documento
  - Q12 output stage 1
  - Q12 output stage 2
  - Q12 output stage 3
  - Q12 output final stage 4
- 13.- Indica la consulta que muestre las 3 tiendas que más han facturado donde se vea su nombre y la cantidad facturada.
- Anotaciones ejemplo de uso

# Objetivo

- Destacar la importancia de tener un usuario para cada responsabilidad, en este caso uno para la gestión de mongoDB a nivel de sistema operativo.
- Personalizar y configurar las rutas de datos y logs (y en general el archivo de configuración) en un servidor mongoDB
- Conocer las rutas y, especialmente el puerto predeterminado.
- Interactuar con las aplicaciones a través de sus servicios para autoarrancarlos, pararlos, reiniciarlos,...
- Comprender el papel de servidor y el papel de cliente incluso en la misma máquina.
- Configurar un cliente MongoDB con IP y puerto

# Entorno

## install docker

```
cd ~
# instalar llave publica
wget https://download.docker.com/linux/ubuntu/gpg
# desarmarla pasar de texto plano a
sudo gpg -o /usr/share/keyrings/docker.gpg --dearmor gpg
# ver version de linux
lsb_release -a
# No LSB modules are available.
# Distributor ID: Ubuntu
# Description:      Ubuntu 20.04.6 LTS
# Release:          20.04
# Codename:         focal
sudo nano /etc/apt/sources.list.d
##### añadimos a /etc/apt/sources.list.d
deb [ arch=amd64 signed-by=/usr/share/keyrings/docker.gpg] https://download.docker.co
#####
sudo apt update # forzar a apt que lea este nuevo repo
sudo apt update && sudo apt upgrade && sudo apt install docker-ce
sudo systemctl start docker # activar servicio
sudo systemctl status docker.service #verificar servicio
sudo usermod -aG docker bieitostudies # añadimos usuario bieitostudies a grupo docke
sudo usermod -aG docker bieito # añadimos usuario bieito a grupo docker
```

## docker imagen mongodb

```
docker ps
docker run --name mongo -d mongodb/mongodb-community-server
docker exec -it mongo bash
# copiamos el archivo de configuración a local
docker cp mongo:/etc/mongod.conf.orig .
# editamos el archivo de configuración para que se puedan conectar todas las ips 0.0.
nano mongod.conf.orig
# # network interfaces
# net:
#   port: 27017
#   bindIp: 0.0.0.0

# paramos contenedor
docker stop mongo
# eliminamos el contenedor
docker rm mongo
# vuelvo a crear el contenedor redirigiendo los puertos
docker run -p 27017:27017 --name mongo -d mongodb/mongodb-community-server
# sobre escribo la configuración de mongo
docker cp ./mongod.conf.orig mongo:/etc/mongod.conf.orig
# actualizo el contenedor
docker stop mongo
docker start mongo
# me conecto
mongo "mongodb://localhost:27017"
```

## Instrucciones de entrega

Elabora y envía un documento PDF con tu nombre completo y DNI en la portada. Usa una nueva hoja por cada apartado, copia el enunciado y contesta. Las capturas de pantalla deben ser de “pantalla completa”, no solo del detalle.

# Enunciado

## 1.- Importa en una colección llamada “bieitolibros” el contenido del archivo books.json

```
# copiamos los archivos a /tmp
docker cp ./sales.json mongo:/tmp/sales.json
# Successfully copied 4.87MB to mongo:/tmp/sales.json
docker cp ./books.json mongo:/tmp/books.json
# Successfully copied 539kB to mongo:/tmp/books.json
# nos conectamos a un terminal
docker exec -it mongo bash
# importamos books.json
mongoimport --db=SBD02-02-bieito --collection=bieitolibros --file=/tmp/books.json
```

```
mongodb@627d186231c4:/ $ mongoimport --db=SBD02-02-bieito --collection=bieitolibros --file=/tmp/books.json
2024-02-29T23:01:50.474+0000    connected to: mongodb://localhost/
2024-02-29T23:01:50.510+0000    431 document(s) imported successfully. 0 document(s) failed to import.
mongodb@627d186231c4:/ $
```

The screenshot shows the MongoDB Compass web interface. The title bar indicates the connection to 'localhost:27017/SBD02-02-bieito.bieitolibros'. The left sidebar shows the database structure with 'SBD02-02-bieito' expanded, revealing the 'bieitolibros' collection. The main panel displays the 'Documents' tab for the 'bieitolibros' collection, showing 431 documents and 1 index. A search filter is set to 'Type a query: { field: 'value' } or Generate query'. Below the filter, there are buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. The document list shows two sample documents with fields like '\_id', 'title', 'isbn', 'pageCount', 'publishedDate', 'thumbnailUrl', 'shortDescription', 'longDescription', 'status', 'authors', and 'categories'.

## 2.- Importa en una colección llamada “bieitoventas” el contenido del archivo sales.json

```
# importamos sales.json
mongoimport --db=SBD02-02-bieito --collection=bieitoventas --file=/tmp/sales.json
```

```
mongodb@627d186231c4:/$ mongoimport --db=SBD02-02-bieito --collection=bieitoventas --file=/tmp/sales.json
2024-02-29T23:01:37.492+0000    connected to: mongodb://localhost/
2024-02-29T23:01:37.934+0000    5000 document(s) imported successfully. 0 document(s) failed to import.
```

The screenshot shows the MongoDB Compass interface for the database 'SBD02-02-bieito' and collection 'bieitoventas'. The left sidebar shows the database structure with 'bieitoventas' selected. The main panel displays the 'Documents' tab, showing a list of documents. The top right corner indicates 5.0k documents and 1 index. The bottom of the panel shows three sample documents with their JSON structure.

**Documents** | Aggregations | Schema | Indexes | Validation

Filter  Type a query: { field: 'value' } or [Generate query](#) [Explain](#) [Reset](#) [Find](#) [Options](#)

[ADD DATA](#) [EXPORT DATA](#) [UPDATE](#) [DELETE](#) 1 - 20 of 5000

**Document 1:**

```
{
  "_id": ObjectId("5bd761dcae323e45a93ccfeb"),
  "saleDate": 2015-02-23T09:53:59.343+00:00,
  "items": Array (9),
  "storeLocation": "Seattle",
  "customer": Object,
  "couponUsed": false,
  "purchaseMethod": "In store"
}
```

**Document 2:**

```
{
  "_id": ObjectId("5bd761dcae323e45a93ccfec"),
  "saleDate": 2017-12-03T18:39:48.253+00:00,
  "items": Array (7),
  "storeLocation": "London",
  "customer": Object,
  "couponUsed": false,
  "purchaseMethod": "In store"
}
```

**Document 3:**

```
{
  "_id": ObjectId("5bd761dcae323e45a93ccfed"),
  "saleDate": 2015-09-02T16:11:59.565+00:00,
  "items": Array (2),
  "storeLocation": "London",
  "customer": Object,
  "couponUsed": false,
  "purchaseMethod": "In store"
}
```



### 3.- Usando la colección libros, escribe la consulta que necesitas para obtener el documento con `_id` igual a 287.

```
// nota verificar que los id sean un string o un objeto tipo `ObjectId()`  
use SBD02-02-bieito  
db.bieitolibros.findOne({  
  "_id": 287  
})
```

The screenshot displays the MongoDB Compass interface and a terminal window. The Compass window shows the 'SBD02-02-bieito.bieitolibros' collection with 431 documents and 1 index. A filter is applied: `{ "_id": 287 }`. The document details are visible:

```
{  
  "_id": 287,  
  "title": "JSP Tag Libraries",  
  "isbn": "193811809X",  
  "pageCount": 656,  
  "publishedDate": 2001-05-01T07:00:00.000+00:00,  
  "thumbnailUrl": "https://s3.amazonaws.com/AKIAJCSRLADLUHVRPFQD_book-thumb-images/shachor.jpg",  
  "longDescription": "JSP Tag Libraries is a bible for serious JSP developers. The reader will become acquainted with the world of custom JSP tags--new JSP technology that is beginning to have an enormous impact on the way people are de",  
  "status": "PUBLISH",  
  "authors": [  
    "Gal Shachor",  
    "Adam Chace",  
    "Magnus Rydin"  
  ],  
  "categories": [  
    "Java",  
    "Internet"  
  ],  
}
```

The terminal window shows the execution of the query in the `SBD02-02-bieito` database:

```
> use SBD02-02-bieito  
< already on db SBD02-02-bieito  
> db.bieitolibros.findOne({ "_id": 287 })  
< {  
  "_id": 287,  
  "title": 'JSP Tag Libraries',  
  "isbn": '193811809X',  
  "pageCount": 656,  
  "publishedDate": 2001-05-01T07:00:00.000Z,  
  "thumbnailUrl": 'https://s3.amazonaws.com/AKIAJCSRLADLUHVRPFQD_book-thumb-images/shachor.jpg',  
  "longDescription": 'JSP Tag Libraries is a bible for serious JSP developers. The reader will become acquainted with the world of custom JSP tags--new JSP technology that is beginning to have an enormous impact on the way people are de',  
  "status": 'PUBLISH',  
  "authors": [  
    'Gal Shachor',  
    'Adam Chace',  
    'Magnus Rydin'  
  ],  
  "categories": [  
    'Java',  
    'Internet'  
  ],  
}
```

## 4.- Usando la colección libros, escribe una consulta que devuelva los libros con exactamente 500 páginas

```
db.bieitolibros.find({
  pageCount: 500,
});
```

The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017' and the selected database 'SBD02-02-bieito.bieitolibros'. The left sidebar shows the database structure with 'bieitolibros' selected. The main panel displays the 'Documents' tab for the 'bieitolibros' collection. A filter is applied: `{ "pageCount": 500 }`. The results show a single document with the following fields: `_id`: 73, `title`: "Spring Roo in Action". Below the Compass interface, a terminal window shows the execution of the query: `db.bieitolibros.find({ "pageCount": 500 })`. The output is a JSON array containing one document with the following details: `_id`: 54, `title`: 'Android in Practice', `isbn`: '1935182927', `pageCount`: 500, `publishedDate`: 2011-09-30T07:00:00.000Z, `thumbnailUrl`: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRPFQ.book-thumb-images/collins.jpg', `shortDescription`: 'Android in Practice is treasure trove of Android goodness, with over 100 tested, ready-to-use techniques including complete end-to-end example applications and practical tips for real world mobile application deve', `longDescription`: 'Android, Google's platform for mobile application development, provides powerful features, a robust SDK, and almost limitless possibilities. It's not hard to find the information you need to build your first Androi', `status`: 'PUBLISH', `authors`: ['Charlie Collins', 'Michael D. Galpin', '', 'Matthias Kaeppeler'], `categories`: ['Mobile Technology'].

## 5.- Usando la colección libros, escribe la consulta que devuelva los libros escritos por “Robi Sen” o por “Chris King”

```
// $in para búsquedas en u campo array
// es decir si se encuentra "dentro de la lista"
db.bieitolibros.find({
  authors: {
    $in: ["Robi Sen", "Chris King"],
  },
});
```

MongoDB Compass - localhost:27017/SBD02-02-bieito.bieitolibros

Connect Edit View Collection Help

My Queries bieitolibros

SBD02-02-bieito.bieitolibros 431 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter { "authors": { "\$in": ["Robi Sen", "Chris King"] } }

Generate query Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

1-3 of 3

| # | bieitolibros | _id | Int32 | title                       | String | isbn         | String | pageCount | Int32 | publishedDate            | Date | thumbnailUrl                | String | shortDescription            | String | longDescription             | String | status     |
|---|--------------|-----|-------|-----------------------------|--------|--------------|--------|-----------|-------|--------------------------|------|-----------------------------|--------|-----------------------------|--------|-----------------------------|--------|------------|
| 1 |              | 1   |       | "Unlocking Android"         |        | "1933988673" |        | 416       |       | 2009-04-01T07:00:00.000Z |      | "https://s3.amazonaws.co... |        | "Unlocking Android: A De... |        | "Android is an open sour... |        | "PUBLI..." |
| 2 |              | 2   |       | "Android in Action, Seco... |        | "1935182722" |        | 592       |       | 2011-01-14T08:00:00.000Z |      | "https://s3.amazonaws.co... |        | "Android in Action, Seco... |        | "When it comes to mobile... |        | "PUBLI..." |
| 3 |              | 514 |       | "Android in Action, Thir... |        | "1617298505" |        | 0         |       | 2011-11-15T08:00:00.000Z |      | "https://s3.amazonaws.co... |        | No field                    |        | No field                    |        | "PUBLI..." |

> MONGOSH

```
> // $in en un campo array si un/varios elementos pertenecen a el
// es decir si se encuentra "dentro de la lista"
db.bieitolibros.find({
  "authors": {
    $in: ["Robi Sen", "Chris King"]
  }
})
< {
  _id: 1,
  title: 'Unlocking Android',
  isbn: '1933988673',
  pageCount: 416,
  publishedDate: 2009-04-01T07:00:00.000Z,
  thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUKVRPFQJ.book-thumb-images/ableson.jpg',
  shortDescription: 'Unlocking Android: A Developer's Guide provides concise, hands-on instruction for the Android operating system and development tools. This book teaches important architectural concepts in a straightforward writing
longDescription: "Android is an open source mobile phone platform based on the Linux operating system and developed by the Open Handset Alliance, a consortium of over 30 hardware, software and telecom companies that focus on open st
status: 'PUBLISH',
  authors: [
```

## 6.- Explica con tus palabras qué hace la siguiente consulta:

```
db.javilibros.find ( { "_id" : { $gt : 10 }, "_id" : { $lt : 15 } } )
```

busca en la colección `javilibros` los documentos cuyo `id` es `> 10` y `< 15`

### Operadores relacionales

- `$eq` - equal - igual
- `$lt` - low than - menor que
- `$lte` - low than equal - menor o igual que
- `$gt` - greater than - mayor que
- `$gte` - greater than equal - mayor o igual que
- `$ne` - not equal - distinto
- `$in` - in - dentro de
- `$nin` - not in - no dentro de

```
db.bieitolibros.find({  
  //#   , concatena  
  "_id" : { $gt : 10 }, //# id mayor que 10  
  "_id" : { $lt : 15 }  //# id menor que 15  
})
```

## 7.- Usando la colección libros, escribe la consulta que devuelva los libros publicados en enero del 2.011

```
// para trabajar con fechas hay que ue pasarlas a objeto date
// desde el 2010-12-31 hasta 2011-02-01 no incluidos
db.bieitolibros.find({
  publishedDate: {
    $gte: new Date("2010-12-31"),
    $lt: new Date("2011-02-01"),
  },
});
```

```
db.bieitolibros.find({
  publishedDate: {
    $gte: ISODate("2010-12-31"),
    $lt: ISODate("2011-02-01"),
  },
});
```

MongoDB Compass - localhost:27017/SBD02-02-bieito.bieitolibros

Connect Edit View Collection Help

{ } My Queries bieitolibros x +

SBD02-02-bieito.bieitolibros 431 1 DOCUMENTS INDEXES

Documents Aggregations Schema Indexes Validation

Filter { "publishedDate": { "\$gte": new Date("2010-12-31"), "\$lt": new Date("2011-02-01") } } Generate query Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE 1-2 of 2

| # | bieitolibros | _id Int32 | title String               | isbn String  | pageCount Int32 | publishedDate Date       | thumbnailUrl String        | shortDescription String    | longDescription String     | status   |
|---|--------------|-----------|----------------------------|--------------|-----------------|--------------------------|----------------------------|----------------------------|----------------------------|----------|
| 1 | 2            |           | "Android in Action, Seco.. | "1935182722" | 592             | 2011-01-14T08:00:00.000Z | "https://s3.amazonaws.co.. | "Android in Action, Seco.. | "When it comes to mobile.. | "PUBLI.. |
| 2 | 155          |           | "Camel in Action"          | "1935182366" | 375             | 2011-01-04T08:00:00.000Z | "https://s3.amazonaws.co.. | "Camel in Action is for .. | "Apache Camel is a Java-.. | "PUBLI.. |

\_MONGODB

```
> db.bieitolibros.find({
  "publishedDate": {
    $gte: new Date("2010-12-31"),
    $lt: new Date("2011-02-01")
  }
})
< {
  _id: 2,
  title: 'Android in Action, Second Edition',
  isbn: '1935182722',
  pageCount: 592,
  publishedDate: 2011-01-14T08:00:00.000Z,
  thumbnailUrl: 'https://s3.amazonaws.com/AKIAJCSRLADLUMVRFFDQ.book-thumb-images/ableson2.jpg',
  shortDescription: 'Android in Action, Second Edition is a comprehensive tutorial for Android developers. Taking you far beyond "Hello Android," this fast-paced book puts you in the driver's seat as you learn important architectural
  longDescription: 'When it comes to mobile apps, Android can do almost anything and with this book, so can you! Android runs on mobile devices ranging from smart phones to tablets to countless special-purpose gadgets. It's the broad
  status: 'PUBLISH',
  authors: [
    'W. Frank Ableson',
    'Robi Sen'
```

## 8.- Usando la colección libros, escribe la consulta que devuelve todos los libros en cuyo título podemos encontrar la palabra “Microsoft”

```
/* # options
    # i toggles case insensitivity,
    # m toggles multiline regular expression
    # x toggles an "extended" capability.
*/
db.bieitolibros.find({
  title: {
    $regex: "Microsoft",
    $options: "i",
  },
});

// otra forma de escribirlo  /pattern/
db.bieitolibros.find({
  title: "/Microsoft/i",
});
```

nota si queremos que solo muestre x cabeceras las indicamos como 2 parámetro en el find con nombre y valor 1

```
db.bieitolibros.find(
  {
    title: /Microsoft/i,
  },
  {
    _id: 1,
    title: 1,
  }
);
```

MongoDB Compass - localhost:27017/SBD02-02-bieito.bieitolibros

Connect Edit View Collection Help

My Queries bieitolibros

SBD02-02-bieito.bieitolibros

431 DOCUMENTS1 INDEXES

Documents Aggregations Schema Indexes Validation

Filter

```
{  "title": {    $regex: "Microsoft", $options: 'i'  }}
```

Generate query Explain Reset Find Options

ADD DATAEXPORT DATAUPDATEDELETE

1-3 of 3

| bieitolibros |           |                            |              |                 |                             |                             |                             |               |           |
|--------------|-----------|----------------------------|--------------|-----------------|-----------------------------|-----------------------------|-----------------------------|---------------|-----------|
|              | _id Int32 | title String               | isbn String  | pageCount Int32 | publishedDate Date          | thumbnailUrl String         | longDescription String      | status String | author    |
| 1            | 130       | Microsoft.NET for Progr... | "1930110197" | 386             | 2002-12-01T08:00:00.000+... | "https://s3.amazonaws.co... | "Written for intermediat... | "PUBLISH"     | [ ] 1 e / |
| 2            | 183       | Microsoft Reporting Ser... | "1932394222" | 656             | 2004-08-01T07:00:00.000+... | "https://s3.amazonaws.co... | "Business reporting is a... | "PUBLISH"     | [ ] 1 e / |
| 3            | 270       | Microsoft Office Essent... | "132623129"  | 480             | 1996-07-01T07:00:00.000+... | "https://s3.amazonaws.co... | "Many books on Microsoft... | "PUBLISH"     | [ ] 1 e / |

>\_MONGODBH

> db.bieitolibros.find(  
 "title": /Microsoft/i  
) , {  
 "\_id": 1,  
 "title": 1  
}  
< {  
 \_id: 130,  
 title: 'Microsoft.NET for Programmers'  
}  
{  
 \_id: 183,  
 title: 'Microsoft Reporting Services in Action'  
}  
{  
 \_id: 270,  
 title: 'Microsoft Office Essentials'  
}  
}

SBD02-02-bieito>

## 9.- Usando la colección libros, escribe una consulta que indique cuantos libros en la categoría de “Java” se han publicado antes del año 2.000.

```
db.bieitolibros.count({
  "categories": { $in: ["Java"] }
  "publishedDate": { $lt: new Date("2000-01-01") }
})
```

MongoDB Compass - localhost:27017/SBD02-02-bieito.bieitolibros

Connect Edit View Collection Help

My Queries bieitolibros

SBD02-02-bieito.bieitolibros

431 1  
DOCUMENTS INDEXES

Documents Aggregations Schema Indexes Validation

Filter {  
 "categories": { \$in: ["Java"] },  
 "publishedDate": { \$lt: new Date("2000-01-01") }  
}

Generate query Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

1 - 9 of 9

| # | bieitolibros | _id                         | Int32 | title                       | String                      | pageCount | Int32 | publishedDate | Date | thumbnailUrl | String | status | String | authors | Array | categories | Array | isbn  | S |
|---|--------------|-----------------------------|-------|-----------------------------|-----------------------------|-----------|-------|---------------|------|--------------|--------|--------|--------|---------|-------|------------|-------|-------|---|
| 1 | 23           | "Hibernate in Action (Ch... | 480   | 1999-06-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 2 elements    | [ ]  | 1 elements   | No fie | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 2 | 132          | "Up to Speed with Swing,... | 560   | 1999-10-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 1 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 3 | 153          | "Java Network Programmin... | 860   | 1999-05-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 4 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 4 | 143          | "The Awesome Power of Po... | 378   | 1998-05-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 1 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 5 | 210          | "Distributed Programming... | 320   | 1999-09-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 1 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 6 | 271          | "Swing"                     | 0     | 1999-12-01T08:00:00.000+... | No field                    | "PUBLISH" | [ ]   | 2 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 7 | 273          | "The Awesome Power of Ja... | 500   | 1998-05-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 1 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |
| 8 | 293          | "Making Sense of Java"      | 180   | 1990-06-01T07:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 6 elements    | [ ]  | 2 elements   | "13263 | 13263  | 13263  | 13263   | 13263 | 13263      | 13263 | 13263 |   |
| 9 | 290          | "Java Applets and Channe... | 372   | 1999-12-01T08:00:00.000+... | "https://s3.amazonaws.co... | "PUBLISH" | [ ]   | 6 elements    | [ ]  | 1 elements   | "18847 | 18847  | 18847  | 18847   | 18847 | 18847      | 18847 | 18847 |   |

>\_MONGOOSH

```
> db.bieitolibros.countDocuments({
  "categories": { $in: ["Java"] },
  "publishedDate": { $lt: new Date("2000-01-01") }
})
< 9
SBD02-02-bieito
```



## 10.- Usando la colección libros, escribe una consulta que indique solamente el título y la fecha de publicación del libro más recientemente publicado.

```
// find({$1},{ $2})
// - parm $1 filtros
// - param $2 headers mostrados (1) ocultos (0)
// soft({name:value})
// - orden descendente (-1)
// - orden ascendente (+1)
db.bieitolibros
  .find(
    // no filtramos ningún dato
    {},
    // mostramos title y publishedDate (1) el resto lo ocultamos (0)
    { title: 1, publishedDate: 1, _id: 0 }
    // orden descendente (-1) por fecha de publicación
  )
  .sort(
    { publishedDate: -1 }
    // nos quedamos solo con el primer resultado
  )
  .limit(1);
```

The screenshot shows the MongoDB Compass interface. At the top, the title bar reads "MongoDB Compass - localhost:27017/SBD02-02-bieito.bieitolibros". The main window displays the "Documents" tab for the "SBD02-02-bieito.bieitolibros" collection. The document count is 431, and the index count is 1. The query editor shows the following query:

```
{
  "title": 1, "publishedDate": 1, "_id": 0
}, {
  "publishedDate": -1
}
```

The query options are set to "Skip: 0", "Limit: 1", and "MaxTimeMS: 60000". Below the query editor, the "EXPORT DATA" button is visible. The results table shows one document:

| # | title String                                | publishedDate Date       |
|---|---------------------------------------------|--------------------------|
| 1 | "The Well-Grounded Rubyist, Second Edition" | 2014-06-24T07:00:00.000Z |

At the bottom, the MongoDB shell output shows the same query and its result:

```
> db.bieitolibros.find(
  // no filtramos ningún dato
  {},
  // mostramos title y publishedDate (1) el resto lo ocultamos (0)
  { "title": 1, "publishedDate": 1, "_id": 0 },
  // orden descendente (-1) por fecha de publicación
  { "publishedDate": -1 },
  // nos quedamos solo con el primer resultado
  { limit: 1 }
)
{
  title: 'The Well-Grounded Rubyist, Second Edition',
  publishedDate: 2014-06-24T07:00:00.000Z
}
```

## 11.- En la colección ventas, explica dónde está el problema en la segunda consulta de agregados de esta captura

indica el error de la siguiente consulta

```
use SBD02-02-bieito

db.bieitoventas.aggregate([
  { // Stage 1
    $group: { _id: "$storeLocation" },
  },
  { // Stage 2 : ordenar
    // ⚠ tenemos de entrada la salida de Stage1
    // existe `_id` , `storeLocation` ya no existe
    $sort: { storeLocation: 1 },
  },
]);
```

### Explicación

- Se ejecuta Stage 1

```
db.bieitoventas.aggregate([
  { // Stage 1
    $group: { _id: "$storeLocation" },
  },
  [...]
```

- Salida Stage 1

```
[
  { _id: "Denver" },
  { _id: "New York" },
  { _id: "San Diego" },
  { _id: "Seattle" },
  { _id: "Austin" },
  { _id: "London" },
];
```

- Se ejecuta Stage 2

- recordemos que tiene como entrada el output de Stage1

```
[...]
{ // Stage 2 : ordenar
  // ⚠️ tenemos de entrada la salida de Stage1
  // existe `_id` , `storeLocation` ya no existe
  $sort: { storeLocation: 1 },
},
[...]
```

- cuando el motor intenta asignar la operación de \$sort , no puede interpretar storeLocation puesto que no forma parte de la salida de Stage 1

## Corrección

```
db.bieitoventas.aggregate([
  {
    $group: { _id: "$storeLocation" },
  },
  {
    // para que funcione debemos ordenar por `_id`
    $sort: { _id: 1 },
  },
]);
```

>\_MONGOSH

```
> db.bieitoventas.aggregate([
  {
    $group: {"_id" : "$storeLocation"}
  },
  {
    $sort:{"_id" : 1}
  }
]
```

```
])
```

```
< {
  _id: 'Austin'
}
{
  _id: 'Denver'
}
{
  _id: 'London'
}
{
  _id: 'New York'
}
{
  _id: 'San Diego'
}
{
  _id: 'Seattle'
}
```

SBD02-02-bieito> |

>\_MONGOSH

```
> db.bieitoventas.aggregate([
  {
    $group: {"_id" : "$storeLocation"}
  },
  {
    $sort:{"_id" : -1}
  }
]
```

```
])
```

```
< {
  _id: 'Seattle'
}
{
  _id: 'San Diego'
}
{
  _id: 'New York'
}
{
  _id: 'London'
}
{
  _id: 'Denver'
}
{
  _id: 'Austin'
}
```

SBD02-02-bieito>

**12.- En la colección de ventas, indica la consulta que usarías para obtener un listado en el que aparezca cuantas unidades de cada elemento se han vendido en cada tienda y que aparezca ordenado alfabéticamente por tienda**

**y descendientemente por número de elementos vendidos. Debería quedar como en esta captura.**

## **Q12 Resolución**

```
db.bieitoventas.aggregate([
  {
    // Stage 1
    // desglosamos el documento por el array $items
    $unwind: "$items",
  },
  {
    // Stage 2: Agrupamos y campo calculado
    $group: {
      _id: {
        // agrupo por tienda y elemento
        tienda: "$storeLocation",
        elemento: "$items.name",
      },
      // campo calculado suma de unidades vendidas
      totalUnidadesVendidas: { $sum: "$items.quantity" },
    },
  },
  {
    // Stage 3: Ordenamos
    $sort: {
      // ordenación 1 : alfabéticamente (1) por tienda (_id.tienda)
      "_id.tienda": 1,
      // ordenación 2: descendientemente (-1) por número de elementos vendidos (totalU
      totalUnidadesVendidas: -1,
    },
  },
  {
    // Stage 4: Mostramos
    $project: {
      _id: 0, // no mostramos el _id
      tienda: "$_id.tienda", // renombramos $_id.tienda a tienda
      elemento: "$_id.elemento", // renombramos $_id.elemento a elemento
      totalUnidadesVendidas: 1, // mostramos unidades vendidas
    },
  },
]);
```

## Q12 vemos la estructura del primer documento

```
db.bieitoventas.aggregate([{$limit:1}])
```



```
[
  {
    _id: ObjectId("5bd761dcae323e45a93ccfeb"),
    saleDate: ISODate("2015-02-23T09:53:59.343Z"),
    items: [
      {
        name: "binder",
        tags: ["school", "general", "organization"],
        price: Decimal128("20.08"),
        quantity: 1,
      },
      {
        name: "pens",
        tags: ["writing", "office", "school", "stationary"],
        price: Decimal128("23.08"),
        quantity: 4,
      },
      {
        name: "backpack",
        tags: ["school", "travel", "kids"],
        price: Decimal128("82.73"),
        quantity: 2,
      },
      {
        name: "printer paper",
        tags: ["office", "stationary"],
        price: Decimal128("15.98"),
        quantity: 3,
      },
      {
        name: "notepad",
        tags: ["office", "writing", "school"],
        price: Decimal128("27.24"),
        quantity: 4,
      },
      {
        name: "notepad",
        tags: ["office", "writing", "school"],
        price: Decimal128("27.7"),
        quantity: 5,
      },
      {
        name: "pens",
```

```
    tags: ["writing", "office", "school", "stationary"],
    price: Decimal128("59.86"),
    quantity: 5,
  },
  {
    name: "binder",
    tags: ["school", "general", "organization"],
    price: Decimal128("27.33"),
    quantity: 9,
  },
  {
    name: "notepad",
    tags: ["office", "writing", "school"],
    price: Decimal128("13.59"),
    quantity: 1,
  },
],
storeLocation: "Seattle",
customer: { gender: "F", age: 45, email: "vatiress@ta.pe", satisfaction: 3 },
couponUsed: false,
purchaseMethod: "In store",
},
];
```

## Q12 output stage 1

desglosamos el array items

```
db.bieitoventas.aggregate([{$unwind: "$items"},{$limit:2}])
```

```
[
  {
    _id: ObjectId("5bd761dcae323e45a93ccfeb"),
    saleDate: ISODate("2015-02-23T09:53:59.343Z"),
    items: {
      name: "binder",
      tags: ["school", "general", "organization"],
      price: Decimal128("20.08"),
      quantity: 1,
    },
    storeLocation: "Seattle",
    customer: { gender: "F", age: 45, email: "vatiress@ta.pe", satisfaction: 3 },
    couponUsed: false,
    purchaseMethod: "In store",
  },
  {
    _id: ObjectId("5bd761dcae323e45a93ccfeb"),
    saleDate: ISODate("2015-02-23T09:53:59.343Z"),
    items: {
      name: "pens",
      tags: ["writing", "office", "school", "stationary"],
      price: Decimal128("23.08"),
      quantity: 4,
    },
    storeLocation: "Seattle",
    customer: { gender: "F", age: 45, email: "vatiress@ta.pe", satisfaction: 3 },
    couponUsed: false,
    purchaseMethod: "In store",
  },
];
```

## Q12 output stage 2

agrupamos y creamos el campo calculado

```
db.bieitoventas.aggregate([
  {
    $unwind: "$items",
  },
  {
    $group: {
      _id: {
        tienda: "$storeLocation",
        elemento: "$items.name",
      },
      totalUnidadesVendidas: { $sum: "$items.quantity" },
    },
  },
  {
    $limit: 1,
  },
]);
```

output

```
[
  {
    _id: { tienda: "Denver", elemento: "laptop" },
    totalUnidadesVendidas: 2025,
  },
];
```

## Q12 output stage 3

ordenamos

```
db.bieitoventas.aggregate([
  {
    $unwind: "$items",
  },
  {
    $group: {
      _id: {
        tienda: "$storeLocation",
        elemento: "$items.name",
      },
      totalUnidadesVendidas: { $sum: "$items.quantity" },
    },
  },
  {
    $sort: {
      "_id.tienda": 1,
      totalUnidadesVendidas: -1,
    },
  },
  {
    $limit: 1,
  },
]);
```

output

```
[
  {
    _id: { tienda: "Austin", elemento: "envelopes" },
    totalUnidadesVendidas: 3490,
  },
];
```

## Q12 output final stage 4

seleccionamos los campos que queremos mostrar y los nombramos

```
db.bieitoventas.aggregate([
  {
    $unwind: "$items",
  },
  {
    $group: {
      _id: {
        tienda: "$storeLocation",
        elemento: "$items.name",
      },
      totalUnidadesVendidas: { $sum: "$items.quantity" },
    },
  },
  {
    $sort: {
      "_id.tienda": 1,
      totalUnidadesVendidas: -1,
    },
  },
  {
    $project: {
      _id: 0,
      tienda: "$_id.tienda",
      elemento: "$_id.elemento",
      totalUnidadesVendidas: 1,
    },
  },
]);
```

output

```
[
  {
    totalUnidadesVendidas: 3490,
    tienda: "Austin",
    elemento: "envelopes",
  },
  { totalUnidadesVendidas: 3331, tienda: "Austin", elemento: "binder" },
  {
    totalUnidadesVendidas: 3128,
    tienda: "Austin",
    elemento: "notepad",
  },
  { totalUnidadesVendidas: 1821, tienda: "Austin", elemento: "pens" },
  {
    totalUnidadesVendidas: 1597,
    tienda: "Austin",
    elemento: "printer paper",
  },
  { totalUnidadesVendidas: 1035, tienda: "Austin", elemento: "laptop" },
  {
    totalUnidadesVendidas: 949,
    tienda: "Austin",
    elemento: "backpack",
  },
  { totalUnidadesVendidas: 7986, tienda: "Denver", elemento: "binder" },
  {
    totalUnidadesVendidas: 7832,
    tienda: "Denver",
    elemento: "envelopes",
  },
  {
    totalUnidadesVendidas: 6276,
    tienda: "Denver",
    elemento: "notepad",
  },
  { totalUnidadesVendidas: 4188, tienda: "Denver", elemento: "pens" },
  {
    totalUnidadesVendidas: 3727,
    tienda: "Denver",
    elemento: "printer paper",
  },
  {
    totalUnidadesVendidas: 2121,
```

```
    tienda: "Denver",
    elemento: "backpack",
},
{ totalUnidadesVendidas: 2025, tienda: "Denver", elemento: "laptop" },
{
    totalUnidadesVendidas: 4037,
    tienda: "London",
    elemento: "envelopes",
},
{ totalUnidadesVendidas: 3871, tienda: "London", elemento: "binder" },
{
    totalUnidadesVendidas: 3278,
    tienda: "London",
    elemento: "notepad",
},
{ totalUnidadesVendidas: 2232, tienda: "London", elemento: "pens" },
{
    totalUnidadesVendidas: 2042,
    tienda: "London",
    elemento: "printer paper",
},
{
    totalUnidadesVendidas: 1177,
    tienda: "London",
    elemento: "backpack",
},
];
```



**13.- Indica la consulta que muestre las 3 tiendas que más han facturado donde se vea su nombre y la cantidad facturada.**

Será necesaria la siguiente expresión para calcular el precio por cantidad.

```
{ $multiply: [ "$campo1", "$campo2" ] }
```

```

db.bieitoventas.aggregate([
  {
    // Stage 1
    // desglosamos el documento por el array $items
    $unwind: "$items",
  },
  {
    // Stage 2
    // como desglosamos es obligatorio agrupar
    // para obligar a que los id sean únicos
    $group: {
      // agrupamos por tienda
      _id: "$storeLocation",
      totalFacturado: {
        // creamos un campo calculado
        $sum: {
          // creamos un contador que acumule
          // en cada iteración acumulamos:
          // los datos de precio por cantidad
          $multiply: ["$items.price", "$items.quantity"],
        },
      },
    },
  },
  {
    // Stage 3 : Mostramos datos
    $project: {
      tienda: "$_id", // renombramos _id a tienda
      totalFacturado: 1, // mostramos total calculado
      _id: 0, // no mostramos _id
    },
  },
  {
    // Stage 4 : Ordenamos Datos
    // como necesitamos los que más facturamos
    // ordenamos desc y mostramos los 3 últimos resultados
    $sort: {
      // ordenamos desc por totalFacturado
      totalFacturado: -1,
    },
  },
  {
    // Stage 4 : Limitamos Datos a mostrar

```

```
    $limit: 3, // mostramos los 3 primeros resultados
  },
]);
```

# Anotaciones ejemplo de uso

```
// ejemplo de secuencia de uso
db.bieitoventas.aggregate([
  // nota se puede acceder a los arrays y objetos :
  // array.obj1.atrib1.
  // nota los valores se pasan por referencia `$`

  { // Stage 1
    // aplana el documento
    // `desagregamos` un array creando las copias necesarias del documento
    // siendo el elemento del array un elemento individual
    // ⚠ hay que tener cuidado dado que la salida genera un copia de documentos
    // por tanto varios documentos van a tener el mismo id
    $unwind: {"key":"arraykey"} // desagregamos por el array
  },
  { // Stage 2
    $match: {"key":"value"} // hace lo mismo que un .find()
  },
  { // Stage 3 , como entrada ya tengo los datos filtrados
    // $group: {_id:{key:value}, ...} estructura obligatoria
    $group: {
      _id: { //nota el valor es una referencia de ahi`$`
        "key":"$value"
      },
      // si necesitamos `acumular` valores hacemos un `count` de 1`
      // es decir cada vez que se itere el valor por el cual agrupamos
      // incrementa en 1 el contador
      countkey: {$sum:1}
      // campo calculados
      // atribuimos a un campo un valor calculado
      // la secuencia depende del operador y los argumentos que necesite
      // si necesitamos varios argumentos se declaran dentro de un array
      // no olvidar que si hay etapas anteriores los datos son por referencia `$`
      key1: { { <operator>: [ <argument1>, <argument2> ... ] } },
      key2: { { <operator>: <argument1> } },
    }
  },
  { // Stage 4 ,
    $project{
      // campos que quiero ver
      // 1 visible
    }
  }
])
```

```
        // 0 oculto
        "key": 1 | 0
    }
},

{ // Stage 5
    // limita el numero de resultados
    $limit: 3
}

1)
```