

---

**M6.UF3.A5.P2**  
**BASE DE DATOS NOSQL**  
**MONGODB**

**Eduard Lara**

# INDICE

---

1. Introducción
2. Instalación Mongodb
3. Operaciones básicas
4. Practica Mongo Queries
5. Acceso Mongodb desde Java
6. Practica CRUD con Mongo

# 1. INTRODUCCION

---

- MongoDB es un sistema de base de datos multiplataforma orientado a documentos, se podrá almacenar cualquier tipo de contenido sin obedecer a un modelo o esquema.
- Se trata de software de licencia libre
- Características: velocidad y flexibilidad. No soporta Joins ni transacciones.
- Almacena documentos JSON (JavaScript Object Notation): es un formato ligero de intercambio de datos.
- JSON está constituido por dos estructuras:
  - **Una colección de pares de nombre/valor**
  - **Una lista ordenada de valores**
- Estas estructuras - comunes a prácticamente todos los lenguajes- en JSON se presentan de estas formas: objeto, array, valor, cadena de caracteres, número.

# 1. ESTRUCTURA JSON

## De XML a JSON

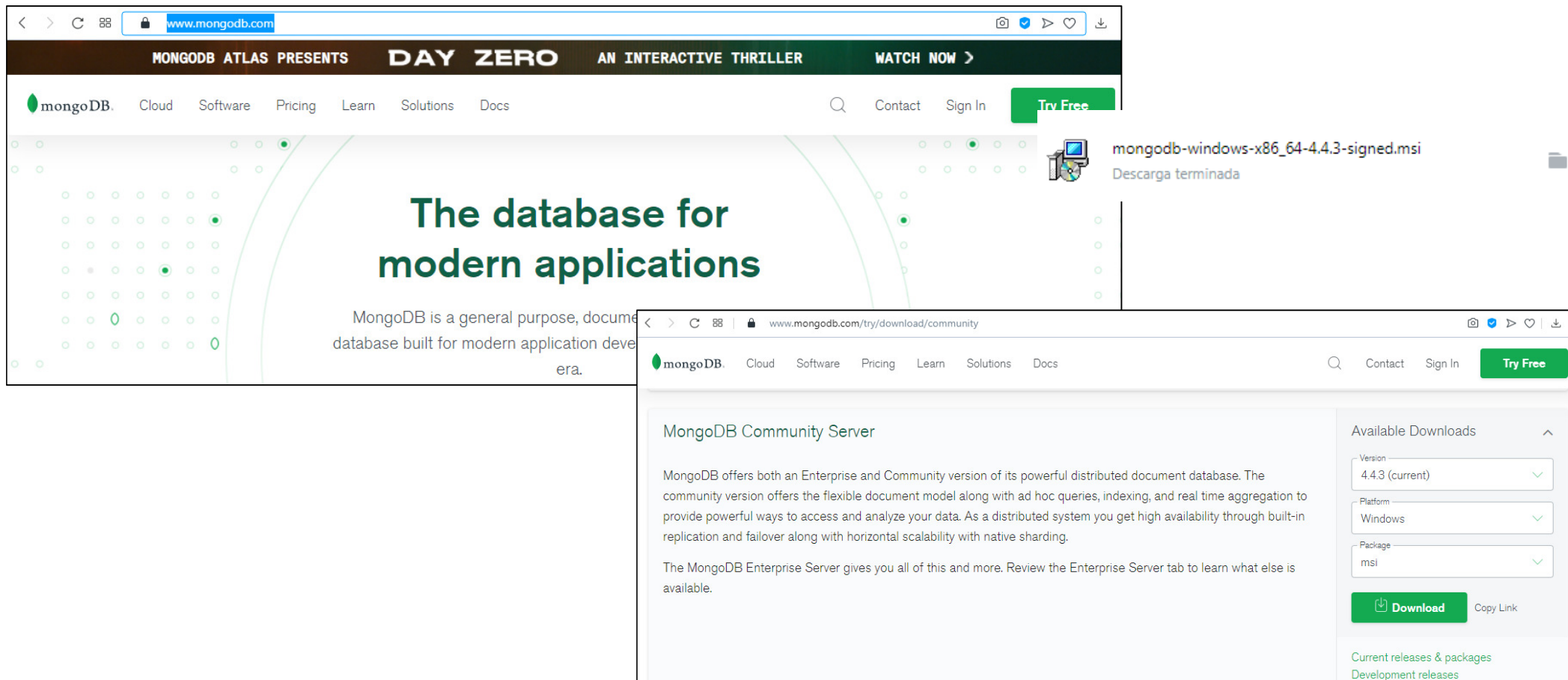
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<departamentos>
<TITULO>DATOS DE LA TABLA DEPART</TITULO>
<DEP_ROW>
  <DEPT_NO>10</DEPT_NO>
  <DNOMBRE>CONTABILIDAD</DNOMBRE>
  <LOC>SEVILLA</LOC>
</DEP_ROW>
<DEP_ROW>
  <DEPT_NO>20</DEPT_NO>
  <DNOMBRE>INVESTIGACION</DNOMBRE>
  <LOC>MADRID</LOC>
</DEP_ROW>
<DEP_ROW>
  <DEPT_NO>30</DEPT_NO>
  <DNOMBRE>VENTAS</DNOMBRE>
  <LOC>BARCELONA</LOC>
</DEP_ROW>
<DEP_ROW>
  <DEPT_NO>40</DEPT_NO>
  <DNOMBRE>PRODUCCION</DNOMBRE>
  <LOC>BILBAO</LOC>
</DEP_ROW>
</departamentos>
```

XML a JSON

```
{
  "departamentos": {
    "TITULO": "DATOS DE LA TABLA DEPART",
    "DEP_ROW": [
      {
        "DEPT_NO": "10",
        "DNOMBRE": "CONTABILIDAD",
        "LOC": "SEVILLA"
      },
      {
        "DEPT_NO": "20",
        "DNOMBRE": "INVESTIGACION", "LOC":
        "MADRID"
      },
      {
        "DEPT_NO": "30",
        "DNOMBRE": "VENTAS",
        "LOC": "BARCELONA"
      },
      {
        "DEPT_NO": "40",
        "DNOMBRE": "PRODUCCION",
        "LOC": "BILBAO"
      }
    ]
  }
}
```

## 2. INSTALACION MONGODB

**Paso 1.** Ir a la página <https://www.mongodb.com>. Descargar la versión Community Server



The image shows two overlapping screenshots of the MongoDB website. The top screenshot displays the main homepage with the heading "The database for modern applications" and a navigation bar. The bottom screenshot shows the "MongoDB Community Server" download page, which includes a description of the community version and a sidebar for selecting download options.

**MongoDB Community Server**

MongoDB offers both an Enterprise and Community version of its powerful distributed document database. The community version offers the flexible document model along with ad hoc queries, indexing, and real time aggregation to provide powerful ways to access and analyze your data. As a distributed system you get high availability through built-in replication and failover along with horizontal scalability with native sharding.

The MongoDB Enterprise Server gives you all of this and more. Review the Enterprise Server tab to learn what else is available.

**Available Downloads**

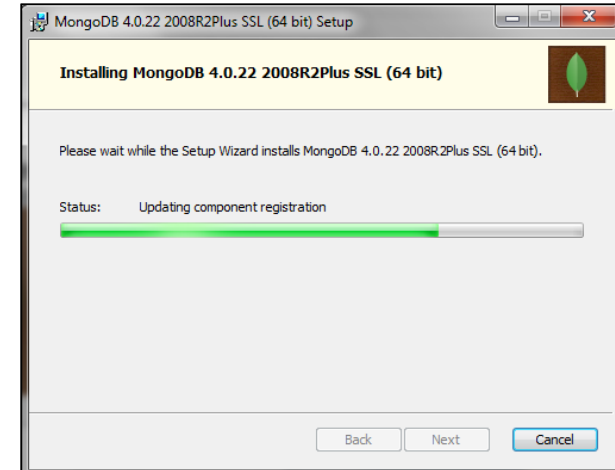
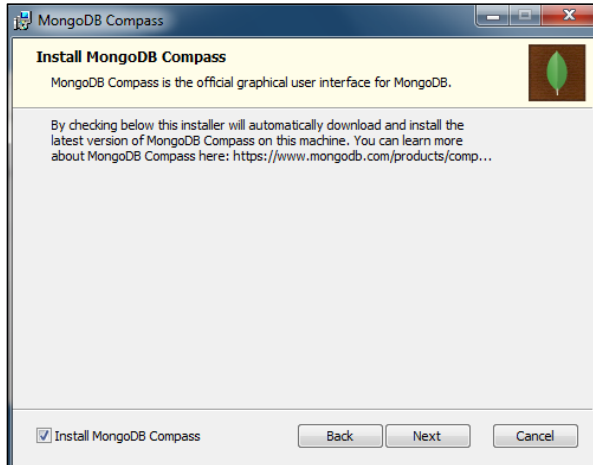
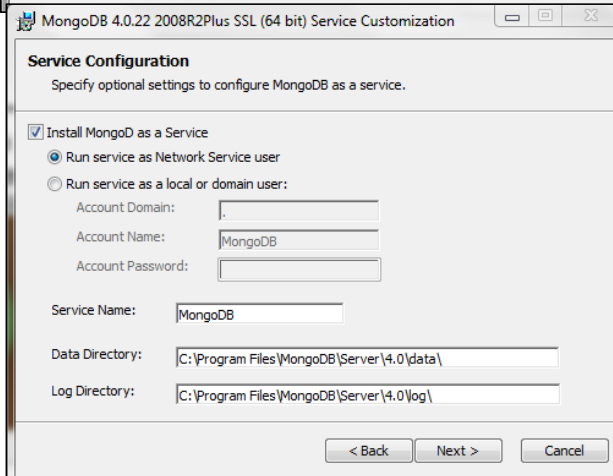
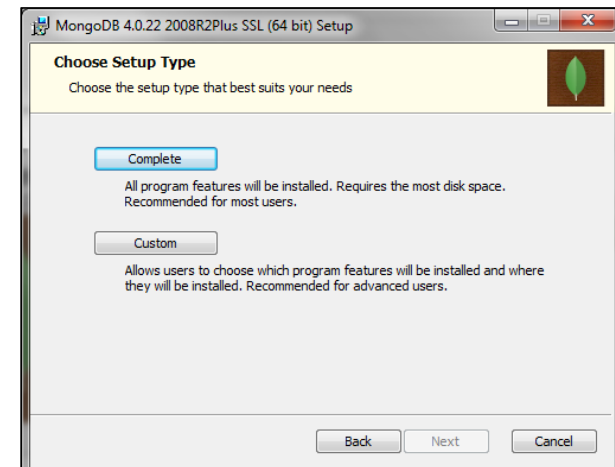
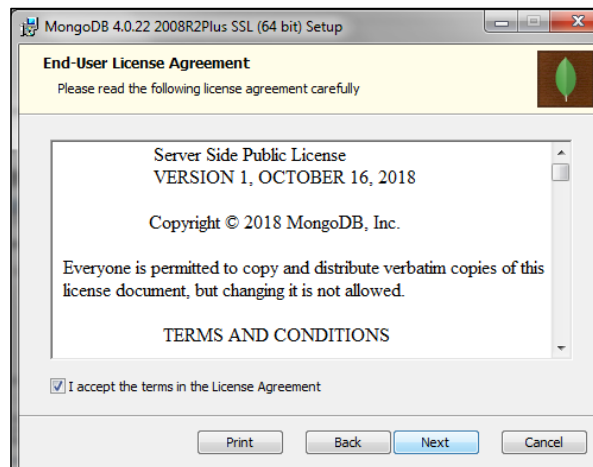
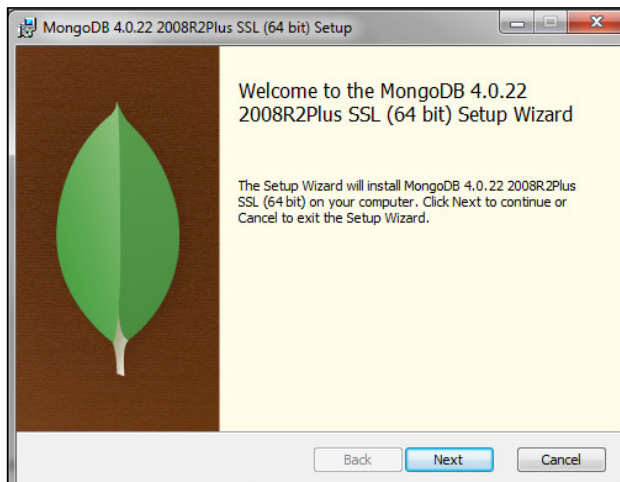
- Version: 4.4.3 (current)
- Platform: Windows
- Package: msi

[Download](#) [Copy Link](#)

[Current releases & packages](#)  
[Development releases](#)

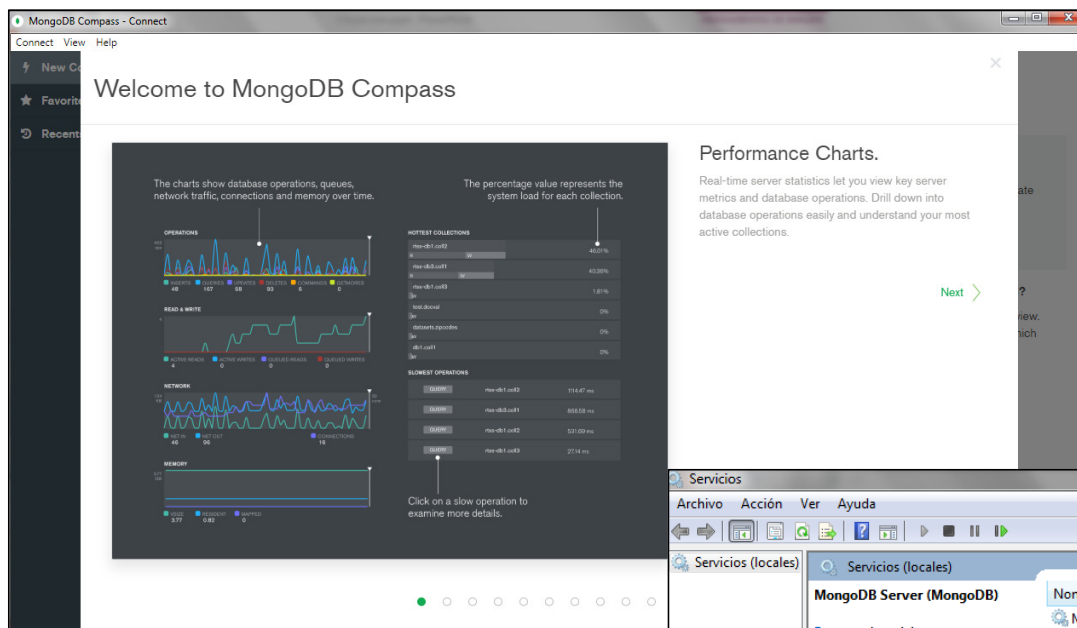
## 2. INSTALACION MONGODB

**Paso 2.** Iniciar la instalación y seguir los siguientes pasos:



## 2. INSTALACION MONGODB

**Paso 3.** Al finalizar la instalación, se queda abierta la aplicación mongodb Compass, es como un workbench para mongodb, que permite conectarse a un servicio mongodb y desde allí gestionar sus bases de datos



En services.msc de Windows podemos ver como queda iniciado el servicio mongodb.

The screenshot shows the Windows Services console. The "Services (locales)" list on the left includes "MongoDB Server (MongoDB)". The main pane displays a table of services, with "MongoDB Server (MongoDB)" selected and highlighted in blue.

Nombre	Descripción	Estado	Tipo de inicio	Iniciar sesión como
Microsoft .NET Framework NGEN v4.0.30319_X86	Microsoft.N...		Automático (i...	Sistema local
Microsoft Edge Elevation Service (MicrosoftEdgeElevationServi...	Keeps Micro...		Manual	Sistema local
Módulos de creación de claves de IPsec para IKE y AuthIP	El servicio IK...	Iniciado	Automático	Sistema local
<b>MongoDB Server (MongoDB)</b>	<b>MongoDB D...</b>	<b>Iniciado</b>	<b>Automático</b>	<b>Servicio de red</b>
Motor de filtrado de base	El Motor de ...	Iniciado	Automático	Servicio local
Mozilla Maintenance Service	El servi de ...		Manual	Sistema local
Net Logon	Mantiene u...		Manual	Sistema local
Office Source Engine	Guarda los a...		Manual	Sistema local
Office Software Protection Platform	Enables the ...		Manual	Servicio de red
Resnet Controls	Extensión de ...		Manual	Servicio local

## 2. INSTALACION MONGODB

**Paso 4.** La url de conexión a mongodb es mongodb://localhost:27017, tanto si nos conectamos desde Compass o desde la API de Java:

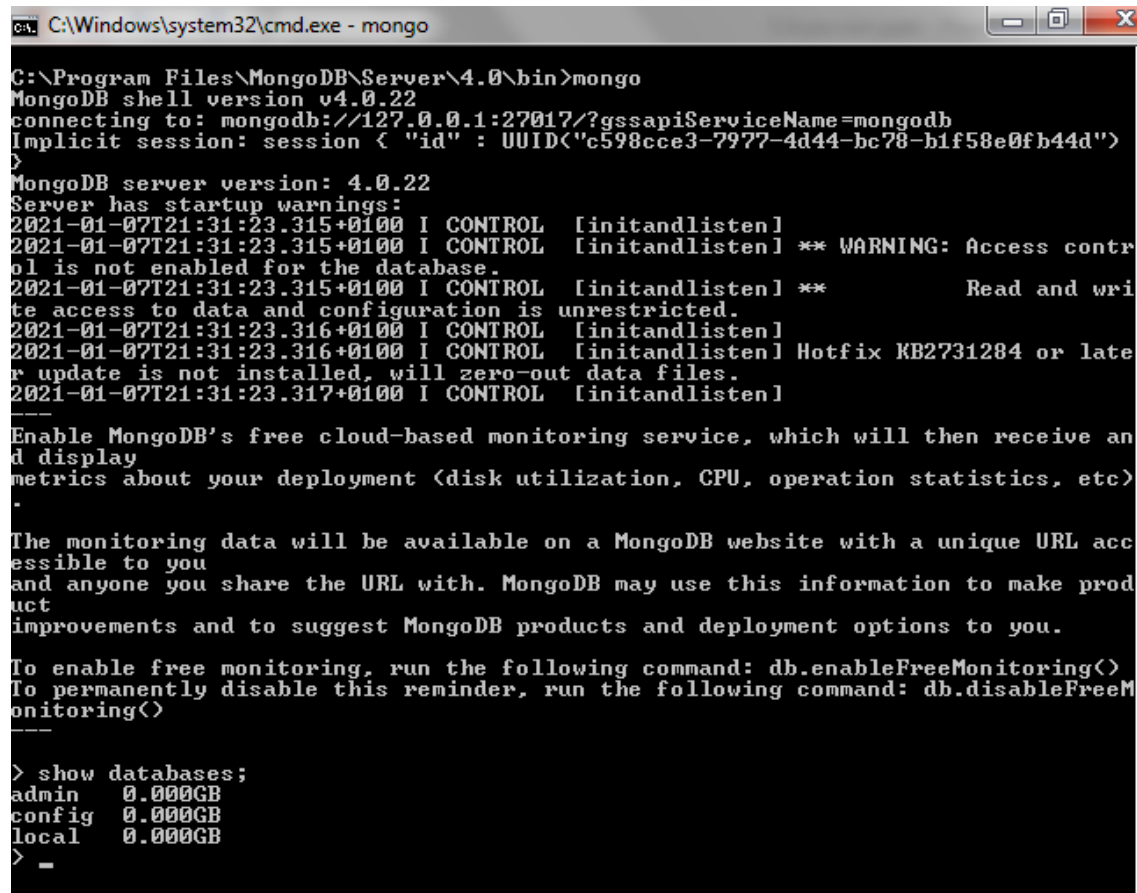
The image shows two screenshots of the MongoDB Compass application. The left screenshot shows the 'New Connection' dialog with the connection string 'mongodb://localhost:27017' entered. The right screenshot shows the 'Databases' tab of the MongoDB Compass interface, displaying a table of databases.

Database Name	Storage Size	Collections	Indexes
admin	16.0KB	0	1
config	16.0KB	0	2
local	16.0KB	1	1



## 2. INSTALACION MONGODB

**Paso 5.** Para conectarnos a la base de datos Mongodb desde terminal, se debe ir al directorio de instalación de mongo/Server/4.0/bin y ejecutar **mongo**:



```
C:\Windows\system32\cmd.exe - mongo

C:\Program Files\MongoDB\Server\4.0\bin>mongo
MongoDB shell version v4.0.22
connecting to: mongodb://127.0.0.1:27017/?gssapiServiceName=mongodb
Implicit session: session < "id" : UUID("c598cce3-7977-4d44-bc78-b1f58e0fb44d") >
MongoDB server version: 4.0.22
Server has startup warnings:
2021-01-07T21:31:23.315+0100 I CONTROL [initandlisten]
2021-01-07T21:31:23.315+0100 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-01-07T21:31:23.315+0100 I CONTROL [initandlisten] **
2021-01-07T21:31:23.315+0100 I CONTROL [initandlisten] ** WARNING: Read and write access to data and configuration is unrestricted.
2021-01-07T21:31:23.316+0100 I CONTROL [initandlisten]
2021-01-07T21:31:23.316+0100 I CONTROL [initandlisten] Hotfix KB2731284 or later update is not installed, will zero-out data files.
2021-01-07T21:31:23.317+0100 I CONTROL [initandlisten]
-----
Enable MongoDB's free cloud-based monitoring service, which will then receive and display metrics about your deployment (disk utilization, CPU, operation statistics, etc)
.

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----

> show databases;
admin    0.000GB
config   0.000GB
local    0.000GB
> -
```

### 3. OPERACIONES BASICAS

---

- MongoDB almacena los documentos en colecciones. Es decir, una colección contiene varios documentos JSON.
- Cada base de datos gestionada por mongo podrá contener un conjunto de colecciones.
- Correspondencia entre términos con base de datos relacionales

NoSQL	SQL
Database	database
coleccion	tabla
documento	Registro

# 3. OPERACIONES BASICAS

## Operaciones básicas en terminal:

- **show databases:** lista bases de datos albergadas en el sistema
- **db:** muestra la base de datos actual.
- **use <otraBBDD> :** para cambiar de base de datos. Si <otraBBDD> no existe, la creará.
- **show collections:** mostrar colecciones de la base de datos actual
- **help:** acceso a la ayuda
- **db.help():** acceso a la ayuda sobre las funciones accesibles

```
> show databases;
admin      0.000GB
config     0.000GB
local      0.000GB
> show collections;
> db
test
> help
db.help()          help on db methods
db.mycoll.help()   help on collection methods
sh.help()          sharding helpers
rs.help()          replica set helpers
help admin         administrative help
help connect       connecting to a db help
help keys          key shortcuts
help misc          misc things to know
help mr            mapreduce

show dbs           show database names
show collections   show collections in current database
show users         show users in current database
show profile       show most recent system.profile entries with time >= 1ms
show logs          show the accessible logger names
show log [name]    prints out the last segment of log in memory
y, 'global' is default
use <db_name>      set current database
db.foo.find()      list objects in collection foo
db.foo.find( { a : 1 } ) list objects in foo where a == 1
it                result of the last line evaluated; use to further iterate
DBQuery.shellBatchSize = x set default number of items to display on shell
exit              quit the mongo shell
>
```

### 3. OPERACIONES BASICAS

#### Creación de una base de datos e Inserción de documentos en terminal

```
> use nueva
switched to db nueva
> db.profes.insert({
...  _id:1, nombre:"pablo", edad:40, modulos:["bbdd","so"]})
WriteResult({ "nInserted" : 1 })
> db.profes.find()
{ "_id" : 1, "nombre" : "pablo", "edad" : 40, "modulos" : [ "bbdd", "so" ] }
>
```

**use nueva** → Creamos la bbdd “nueva” y nos “colocamos en ella

**db.profes.insert( {} )** → Añadimos a la colección “profes” (si no existe creará la colección) un primer documento JSON. Se pueden añadir tantos documentos JSON como se quieran

# 3. OPERACIONES BASICAS

## Consultas en terminal

- Para realizar consultas utilizamos la función find que tiene dos argumentos:
  - Primero: es un documento. Indica el filtro. (Where)
  - Segundo: es tb. un documento. Indica la proyección (Select)
- Haz:
  - db.profes.find()

```
> db.profes.find();  
< { "_id" : 1, "nombre" : "pablo", "edad" : 40, "modulos" : [ "bbdd", "so" ] }  
>
```

- db.profes.find().pretty() (-->Más bonito)

```
> db.profes.find().pretty();  
< {  
  "_id" : 1,  
  "nombre" : "pablo",  
  "edad" : 40,  
  "modulos" : [  
    "bbdd",  
    "so"  
  ]  
}  
>
```

# 3. OPERACIONES BASICAS

## Consultas en terminal

- `db.profes.find({nombre:"pablo"})` → Muestra los documentos con el campo nombre igual a "pablo"

```
> db.profes.find({nombre:"pablo"});  
{ "_id" : 1, "nombre" : "pablo", "edad" : 40, "modulos" : [ "bbdd", "so" ] }  
>
```

- `db.profes.find({}, {edad:1})` → Devolverá las edades de todos los documentos (sistema añade el `_id` por defecto, para quitarlo mirar siguiente ejemplo)

```
> db.profes.find({}, {edad:1});  
{ "_id" : 1, "edad" : 40 }  
>
```

- `db.profes.find({}, {edad:1, _id:0})` → No mostrará el id

```
> db.profes.find({}, {edad:1, _id:0});  
{ "edad" : 40 }  
>
```

# 3. OPERACIONES BASICAS

---

## Consultas en terminal

- Si un documento carece del campo que solicito aparecerá NULL
- En realidad, la función find devuelve un cursor que podríamos recoger con una variable...

# 3. OPERACIONES BASICAS

---

## Operadores y comparadores en consultas: \$gt, \$lt, \$in, \$not, \$or, ...

- Ejemplo:
  - `db.profes.find({nombre:{$exists:true}},{nombre:1})` → Devolverá el campo “nombre” de aquellos documentos en los que existe un campo nombre.
  - `db.profes.find({edad:{$gt:30}})` → Mostrará profesores mayores de 30
  - `db.profes.find({modulos:{$in:["bbdd"]}})` → Mostrará profesores que pueden dar bbdd
- Modificadores del resultado: limit, count, sort...
  - Para verlos podemos hacer: `db.profes.find().help()`
  - Para limitar la salida: `db.profes.find().limit(5)`
  - Para ordenar la salida `db.profes.find().sort({name:1})`
  - Número de filas: `db.profes.find().count()`.



## 3. OPERACIONES BASICAS

### Actualización en terminal

- Puede ser parcial, es decir, solo modificamos el valor de algún campo, o total, modificamos el documento entero.
- Ejemplos de modificaciones totales (reemplazos):
  - `db.profes.update({ _id: 1 }, { nombre : "Francisco" })` → El documento con `_id=1` quedará con un solo campo.
  - NOTA: no podemos modificar el `_id`

```
> db.profes.find();
{ "_id" : 1, "nombre" : "pablo", "edad" : 40, "modulos" : [ "bbdd", "so" ] }
> db.profes.update(<<_id:1>>,<nombre:"Francisco">>);
WriteResult(<< "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 >>)
> db.profes.find();
{ "_id" : 1, "nombre" : "Francisco" }
>
```

# 3. OPERACIONES BASICAS

## Actualización en terminal

- Ejemplos de modificaciones parciales:
  - `db.profes.update({ _id:1}, {$set: {nombre:"Francisco"}})` → cambiamos el campo nombre del documento con `_id=1`.
  - `db.profes.update({ _id:1}, {$push:{modulos:"html"}})` → añadimos un elemento al array "modulos".

```
> db.profes.find();
{ "_id" : 1, "nombre" : "pablo", "edad" : 40, "modulos" : [ "bbdd", "so" ] }
> db.profes.update(<_id:1>,<$set:<nombre:"Francisco">>);
WriteResult(< "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 >)
> db.profes.find();
{ "_id" : 1, "nombre" : "Francisco", "edad" : 40, "modulos" : [ "bbdd", "so" ] }
```

- Por defecto, el update solo afecta a un documento (el primero que pase el filtro)

# 3. OPERACIONES BASICAS

---

## Borrado en terminal

- Simplemente indicamos el filtro que deben pasar los documentos para ser eliminados.
- Ejemplo:
  - `db.profes.remove({ _id:1})` → Elimina todos los documentos que cumplan la condición.

```
> db.profes.remove(<{_id:1}>);  
WriteResult(< "nRemoved" : 1 >)  
> db.profes.find(<);  
>  
>
```

# 3. OPERACIONES BASICAS

## BSON

- BSON es un formato de serialización binaria
- Se utiliza para almacenar documentos y hacer llamadas a procedimientos en MongoDB.
- La especificación BSON se encuentra en **[bsonspec.org](https://bsonspec.org)**. BSON
- Cada tipo de dato tiene un número y un alias que se pueden utilizar con el operador **\$type** para consultar los documentos de tipo BSON.
- Algunos de los tipos BSON (Tipo, Numero, Alias) soportados como valores en los documentos:

Double, 1, “double”;	String, 2, “string”;	Object, 3, “object”;	Array, 4, “array”;
Binary data, 5, “binData”;	ObjectId, 7, “objectId”;	Boolean, 8, “bool”;	Date, 9, “date”;
Null, 10, “null”;	Symbol, 14, “symbol”;	Timestamp, 17, “timestamp”	

## 4. PRACTICA MONGO QUERIES

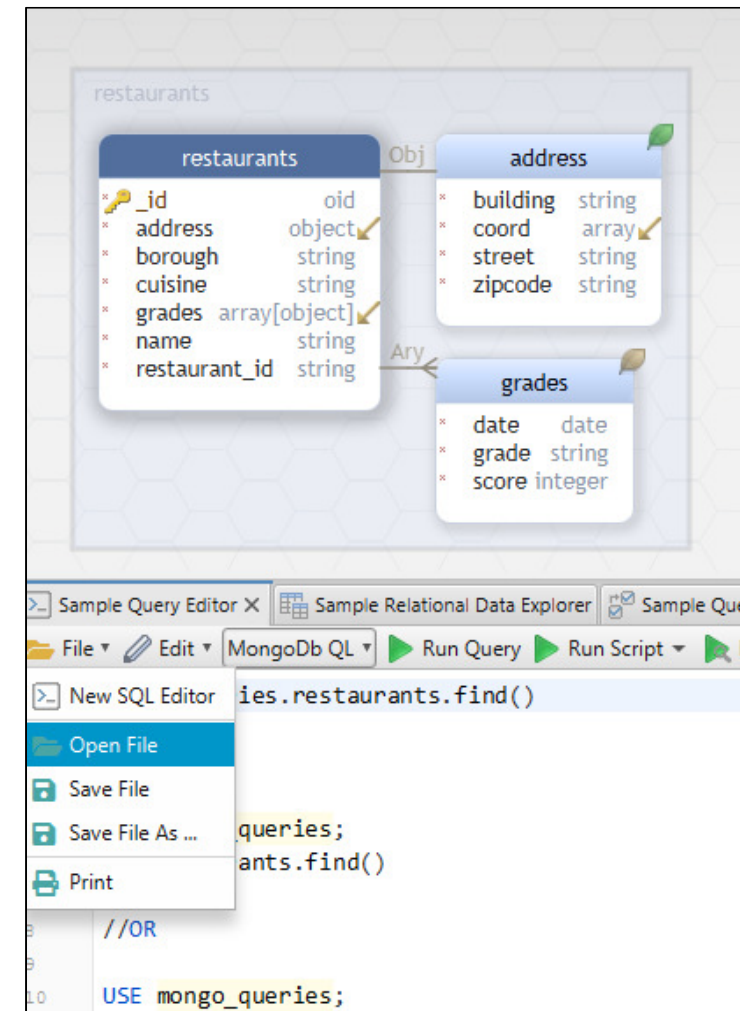
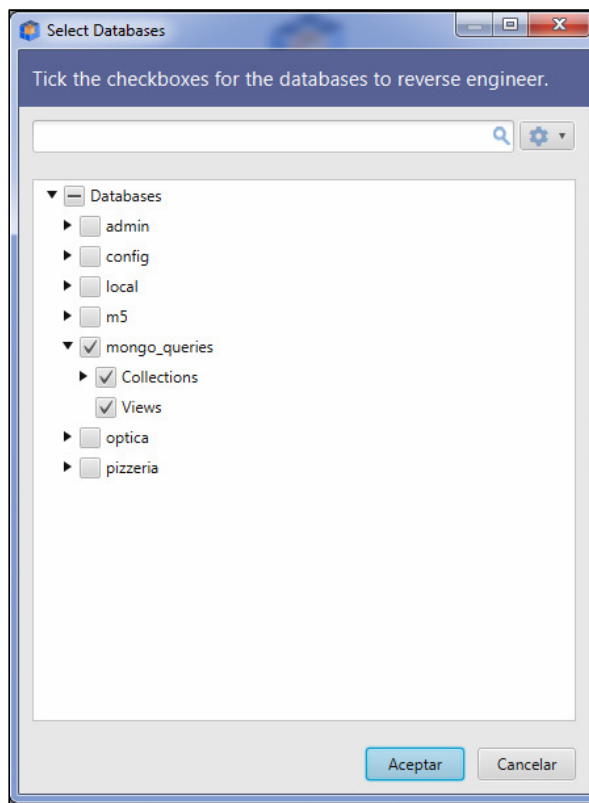
**Pas 0.** Crea con MongoDB Compass una base de dades mongo\_queries i la col.lecció restaurants. Afegeix les dades important el fitxer restaurants.json:

The image shows three overlapping windows from the MongoDB Compass application:

- Create Database:** The 'Database Name' field contains 'mongo\_queries'. The 'Collection Name' field contains 'restaurants'. There are checkboxes for 'Capped Collection' and 'Use Custom Collation', both of which are unchecked. A blue informational box states: 'Before MongoDB can save your new database, a collection name must also be specified at the time of creation. [More Information](#)'. At the bottom are 'CANCEL' and 'CREATE DATABASE' buttons.
- mongo\_queries.restaurants:** This window shows the 'Documents' tab. It has a filter bar with '{ field: 'value' }'. Below it is an 'ADD DATA' button with a dropdown menu that is open, showing 'Import File' and 'Insert Document' options. There are also 'VIEW' and 'GRID' icons.
- Import To Collection mongo\_queries.restaurants:** This window shows the 'Select File' field with the path 'C:\Users\ADMIN\Desktop\ITAcademy\Modulo 5\restaurants.json' and a 'BROWSE' button. Under 'Select Input File Type', the 'JSON' option is selected. There is an 'Options' section with a 'Stop on errors' checkbox. A green progress bar is at the bottom, showing 'Import completed' and '3772 (100%)'. A 'DONE' button is in the bottom right corner.

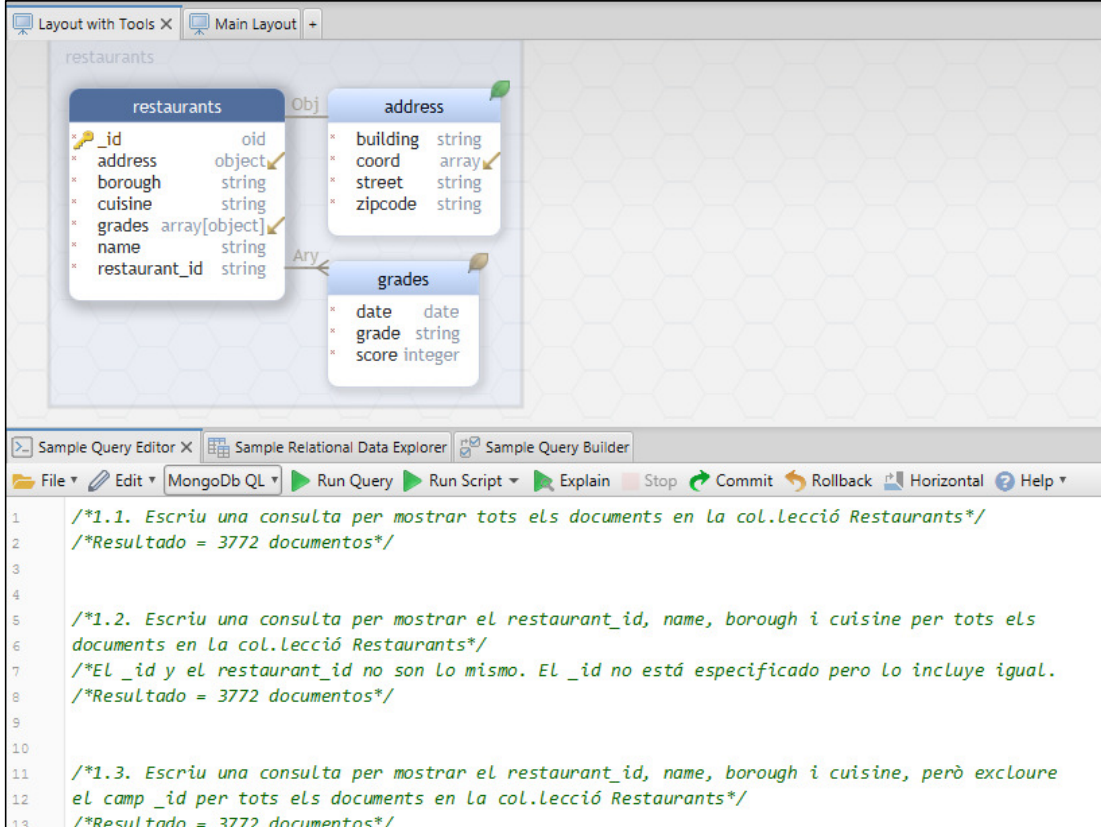
## 4. PRACTICA MONGO QUERIES

**Pas 1.** Amb dbschema obre la base de dades mongo\_queries i el fitxer mongo\_quèries\_dbchema:



## 4. PRACTICA MONGO QUERIES

**Pas 2.** Un cop obert el fitxer script, n'hi ha prou amb seleccionar una query i donar-li al botó de Run Query per executar-la de forma exclusiva:



The screenshot shows the MongoDB Compass interface. On the left, a schema diagram for the 'restaurants' collection is displayed, showing fields like \_id, address, borough, cuisine, grades, name, and restaurant\_id. The main area shows a script with three queries:

```
1 /*1.1. Escriu una consulta per mostrar tots els documents en la col.Llecció Restaurants*/  
2 /*Resultado = 3772 documentos*/  
3  
4  
5 /*1.2. Escriu una consulta per mostrar el restaurant_id, name, borough i cuisine per tots els  
6 documents en la col.Llecció Restaurants*/  
7 /*El _id y el restaurant_id no son lo mismo. El _id no está especificado pero lo incluye igual.  
8 /*Resultado = 3772 documentos*/  
9  
10  
11 /*1.3. Escriu una consulta per mostrar el restaurant_id, name, borough i cuisine, però excloure  
12 el camp _id per tots els documents en la col.Llecció Restaurants*/  
13 /*Resultado = 3772 documentos*/
```

On the right, the 'Sample Query Editor' is open, showing the same script. A tooltip indicates 'Run query (Ctrl-Enter)'. Below the editor, the results of the first query are displayed as a table:

Na...	V...	Type	_id	address	borough	cuisine	
...	606...	Obj...	6065ddae45157a19883c3eaa	Document({building=1007, coord=[-73.856077, 40.848447], street=Mor...	Bronx	Bakery	[Document]
▶	...	Obj...	6065ddae45157a19883c3eab	Document({building=469, coord=[-73.961704, 40.662942], street=Flatb...	Brooklyn	Hambu...	[Document]
...	Bronx	Stri...	6065ddae45157a19883c3eac	Document({building=351, coord=[-73.98513559999999, 40.7676919], st...	Manhattan	Irish	[Document]
...	Bak...	Stri...	6065ddae45157a19883c3ead	Document({building=2780, coord=[-73.98241999999999, 40.579505], st...	Brooklyn	American	[Document]
▶	...	List...	6065ddae45157a19883c3eae	Document({building=97-22, coord=[-73.8601152, 40.7311739], street=6...	Queens	Jewish/...	[Document]
...	Mo...	Stri...	6065ddae45157a19883c3eaf	Document({building=8825, coord=[-73.8803827, 40.7643124], street=A...	Queens	American	[Document]
...	300...	Stri...	6065ddae45157a19883c3eb0	Document({building=2206, coord=[-74.1377286, 40.6119572], street=Vi...	Staten Isl...	Jewish/...	[Document]
			6065ddae45157a19883c3eb1	Document({building=7114, coord=[-73.9068506, 40.6199034], street=A...	Brooklyn	Delicat...	[Document]
			6065ddae45157a19883c3eb2	Document({building=6409, coord=[-74.00528899999999, 40.628886], st...	Brooklyn	American	[Document]
			6065ddae45157a19883c3eb3	Document({building=1839, coord=[-73.9482609, 40.6408271], street=N...	Brooklyn	Ice Cre...	[Document]
			6065ddae45157a19883c3eb4	Document({building=2300, coord=[-73.8786113, 40.8502883], street=S...	Bronx	American	[Document]
			6065ddae45157a19883c3eb5	Document({building=7715, coord=[-73.9973325, 40.61174889999999], ...	Brooklyn	American	[Document]

## 4. PRACTICA MONGO QUERIES

---

/\*1.1. Escriu una consulta per mostrar tots els documents en la col.lecció Restaurants\*/  
/\*Resultado = 3772 documentos\*/  
/\*1.2. Escriu una consulta per mostrar el restaurant\_id, name, borough i cuisine per tots els documents en la col.lecció Restaurants\*/  
/\*El \_id y el restaurant\_id no son lo mismo. El \_id no está especificado pero lo incluye igual.\*/Resultado = 3772 documentos\*/  
/\*1.3. Escriu una consulta per mostrar el restaurant\_id, name, borough i cuisine, però excloure el camp \_id per tots els documents en la col.lecció Restaurants. Resultado = 3772 documentos\*/  
/\*1.4. Escriu una consulta per mostrar restaurant\_id, name, borough i zip code, però excloure el camp \_id per tots els documents en la col.lecció Restaurants\*/  
/\*El campo zipcode está anidado dentro del campo address y cambia la sintaxis.\*/  
/\*Resultado 3772 documentos)\*/  
/\*1.5. Escriu una consulta per mostrar tot els restaurants que estan en el Bronx\*/  
/\*Resultado 309 documentos)\*/  
/\*1.6. Escriu una consulta per mostrar els primers 5 restaurants que estan en el Bronx\*/  
/\*Resultado los primeros 5 documentos)\*/  
/\*1.7. Escriu una consulta per mostrar el pròxim 5 restaurants després de saltar els primers 5 del Bronx\*/  
/\*Resultado los siguientes 5 documentos)\*/



## 4. PRACTICA MONGO QUERIES

---

/\*1.8. Escriu una consulta per trobar els restaurants que tenen un score de m s de 90\*/

/\*Resultado 3 documentos)\*/

/\*1.9. Escriu una consulta per trobar els restaurants que tenen un score de més que 80 però menys que 100. Resultado 4 documentos)\*/

/\*1.10. Escriu una consulta per trobar els restaurants quins localitzen en valor de latitud menys que -95.754168. Resultado 3 documentos)\*/

/\*1.11. Escriu una consulta de MongoDB per a trobar els restaurants que no preparen cap cuisine de 'American' i el seu puntaje de qualificació superior a 70 i latitud inferior a -65.754168\*/

/\*La palabra "American" tiene un espacio detrás y la búsqueda hay que hacerla por "American "

/\*Resultado 5 documentos)\*/

/\*1.12. Escriu una consulta per trobar els restaurants quins no preparen cap cuisine de 'American' i va aconseguir un marcador més que 70 i localitzat en la longitud menys que -65.754168. Nota : Fes aquesta consulta sense utilitzar \$and operador. Resultado 5 documentos)\*/

/\*2.1. Escriu una consulta per trobar els restaurants quins no preparen cap cuisine de 'American ' i va aconseguir un punt de grau 'A' no pertany a Brooklyn. S'ha de mostrar el document segons la cuisine en ordre descendent. Resultado 2017 documentos)\*/

/\*2.2. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants quin contenir 'Wil' com les tres primeres lletres en el seu nom. Resultado 3 documentos)\*/

## 4. PRACTICA MONGO QUERIES

/\*2.3. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants quin contenir 'ces' com les últimes tres lletres en el seu nom. Resultado 6 documentos\*/  
db.restaurants.find( { name: {"\$regex": "ces\$"}}, {restaurant\_id:1, name:1, borough:1, cuisine:1});

/\*2.4. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants quin contenir 'Reg' com tres lletres en algun lloc en el seu nom. Resultado 7 documentos\*/

/\*2.5. Escriu una consulta per trobar els restaurants quins pertanyen al Bronx i va preparar qualsevol plat American o xinés. "American " va con un espacio al final. Resultado 91 documentos\*/

/\*2.6. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que pertanyen a Staten Island o Queens o Bronx or Brooklyn. Resultado 1889 documentos\*/

/\*2.7. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que no pertanyen a Staten Island o Queens o Bronx or Brooklyn. Resultado 1883 documentos\*/

/\*2.8. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que aconseguixin un marcador quin no és més que 10. Resultado 3529 documentos \*/

/\*2.9. Escriu una consulta per trobar el restaurant\_id, name, borough i cuisine per a aquells restaurants que preparen peix excepte 'American ' i 'Chinese' o el name del restaurant comença amb lletres 'Wil'\*/  
/\*Resultado 2402 documentos \*/

/\*2.10. Escriu una consulta per trobar el restaurant\_id, name, i grades per a aquells restaurants que aconseguixin un grau "A" i un score 11 en dades d'estudi ISODate "2014-08-11T00:00:00Z"\*/  
/\*Resultado 0 documentos \*/

## 4. PRACTICA MONGO QUERIES

/\*3.1. Escriu una consulta per trobar el restaurant\_id, name i grades per a aquells restaurants on el 2n element de varietat de graus conté un grau de "A" i marcador 9 sobre un ISODate "2014-08-11T00:00:00Z". Resultado 0 documentos\*/

/\*3.2. Escriu una consulta per trobar el restaurant\_id, name, adre a i ubicaci geogr fica per a aquells restaurants on el segon element del array coord cont un valor quin s m s que 42 i fins a 52\*/

/\*Resultado 7 documentos\*/

/\*3.3. Escriu una consulta per organitzar el nom dels restaurants en ordre ascendent juntament amb totes les columnas. Resultado 3772 documentos\*/

/\*3.4. Escriu una consulta per organitzar el nom dels restaurants en descendir juntament amb totes les columnas. Resultado 3772 docuemntos. El primero (DESC) es Zum Stammtisch\*/

/\*3.5. Escriu una consulta a organitzar el nom de la cuisine en ordre ascendent i per el mateix barri de cuisine. Ordre descendint. Resultado 3772 documentos\*/

/\*3.6. Escriu una consulta per saber tant si totes les direccions contenen el carrer o no\*/

/\*Resultado 0 documentos\*/

/\*3.7. Escriu una consulta quin seleccionerà tots el documents en la col.lecció de restaurants on el valor del camp coord és Double. Resultado 3772 documentos\*/

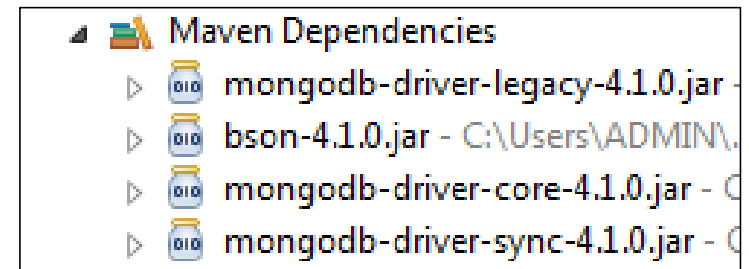
/\*3.8. Escriu una consulta quin seleccionerà el restaurant\_id, name i grade per a aquells restaurants quins retorns 0 com a resta después de dividir el marcador per 7. Resultado 1585 documentos\*/

/\*3.9. Escriu una consulta per trobar el nome de restaurant, borough, longitud i altitud i cuisine per a

## 5. ACCESO MONGODB DESDE JAVA

- Para trabajar en Java con MongoDB podemos descargar el driver desde la URL de MongoDB <https://mongodb.github.io/mongo-java-driver/>
- Si se utiliza Maven, debemos agregar las siguientes dependencias en pom.xml:

```
<dependencies>
  <dependency>
    <groupId>org.mongodb</groupId>
    <artifactId>mongodb-driver-legacy</artifactId>
    <version>4.1.0</version>
  </dependency>
</dependencies>
```



# 5. ACCESO MONGODB DESDE JAVA

---

## Conexión base de datos

- Para conectarnos a la bbdd creamos una instancia **MongoClient**. Es el equivalente a la típica variable Connection de las bases de datos relacionales
- Por defecto, se puede instanciar un objeto MongoClient sin ningún parámetro para conectarse a una instancia MongoDB ejecutándose en localhost:27017:

**MongoClient mongoClient = MongoClient.create();**

- O se puede especificar el ConnectionString:

**String connectString = "mongodb://localhost:27017";**

**MongoClient mongoClient = MongoClient.create(connectString);**

# 5. ACCESO MONGODB DESDE JAVA

## CRUD

- Todos los métodos con operaciones CRUD (Create, Read, Update, Delete) en Java se acceden a través de la interfaz **MongoCollection**
- Las instancias de **MongoCollection** se pueden obtener a partir de una instancia **MongoClient** por medio de una **MongoDatabase**.
- **MongoCollection** es una interfaz genérica: el parámetro de tipo Document es la clase que los clientes utilizan para insertar o modificar los documentos de una colección y es el tipo predeterminado para devolver búsquedas (find).
- El método de un solo argumento **getCollection** devuelve una instancia de **MongoCollection <Document>**, y así es como podemos trabajar con instancias de la clase de documento.

```
MongoClient mongoClient = MongoClient.create();  
MongoDatabase db = mongoClient.getDatabase("nueva");  
MongoCollection <Document> coleccion = db.getCollection("profes");
```

# 5. ACCESO MONGODB DESDE JAVA

## Listado de las bases de datos

```
List<Document> databases = mongoClient.listDatabases().into(new ArrayList<>());  
databases.forEach(db -> System.out.println(db.toJson()));  
  
MongoCursor<String> dbsCursor = mongoClient.listDatabaseNames().iterator();  
while(dbsCursor.hasNext()) {  
    System.out.println(dbsCursor.next());  
}
```

## Listado de las colecciones de una base de datos

```
MongoDatabase db = mongoClient.getDatabase ("nueva");  
for (String name : db.listCollectionNames()) {  
    System.out.println(name);  
}
```

# 5. ACCESO MONGODB DESDE JAVA

## Listado de los documentos de una colección (READ)

- Los datos de una colección se pueden cargar en una lista utilizando el método **find().into()** de la siguiente manera:

```
MongoCollection<Document> coleccion = db.getCollection("profes");
List<Document> consulta = coleccion.find().into(new ArrayList<Document> ());
for (int i =0; i < consulta.size(); i++) {
    System.out.println("-" + consulta.get(i).toString());
}
```

- El método **find()** devuelve un cursor, en concreto devuelve una instancia **FindIterable**. Podemos utilizar el método **iterator()** para recorrer el cursor. Los documentos de la colección y se visualizan en formato Json:

```
MongoCursor<Document> cursor = coleccion.find().iterator();
while (cursor.hasNext()) {
    Document doc = cursor.next();
    System.out.println (doc.toJson());
}
cursor.close();
```



## 5. ACCESO MONGODB DESDE JAVA

### Inserción documentos (CREATE)

- Para insertar documentos, creamos un objeto Document, con el método put asignamos los pares *clave-valor*, donde el primer parámetro es el nombre del campo o la clave, y el segundo el valor.
- Mediante el método insertOne se inserta un documento en la colección:

```
//Insercion con clave _id  
Document amigo = new Document();  
amigo.put("_id", 1);  
amigo.put("nombre", "Pedro");  
amigo.put("edad", 30);  
coleccion.insertOne(amigo);
```

```
{"_id": 1, "nombre": "Pedro", "edad": 30}
```

# 5. ACCESO MONGODB DESDE JAVA

## Inserción documentos (CREATE)

- Inserción de un documento sin id.

```
//Insercion sin clave _id
Document agenda = new Document();
agenda.put("nombre", "Jose");
agenda.put("telefono", 925677);
agenda.put("curso", "DAM2");
coleccion.insertOne(agenda);
```

```
{"_id": {"$oid": "600c886a35f9321b63de0ea2"}, "nombre": "Jose", "telefono": 925677, "curso": "DAM2"}
```

- Inserción de un documento con fecha.

```
//Insercion con _id y Date
Document agenda = new Document();
agenda.put("_id", 2);
agenda.put("telefono", 655577);
agenda.put("curso", "DAM2");
agenda.put("fecha", new Date());
coleccion.insertOne(agenda);
```

```
{"_id": 2, "telefono": 655577, "curso": "DAM2", "fecha": {"$date": "2021-01-24T10:18:29.483Z"}}
```

## 5. ACCESO MONGODB DESDE JAVA

### Inserción documentos (CREATE)

- Inserción de un objeto dentro de un documento:

```
//Insercion de un objeto dentro de otro objeto
Document main = new Document();
main.put("_id", 3);
main.put("nombre", "Maria");
main.put("telefono", 444444);
BasicDBObject clase = new BasicDBObject();
clase.put("nombre", "Informatica");
clase.put("nivel", "DAM2");
main.put("clase", clase);
coleccion.insertOne(main);
```

```
{"_id": 3, "nombre": "Maria", "telefono": 444444, "clase": {"nombre": "Informatica", "nivel": "DAM2"}}
```

## 5. ACCESO MONGODB DESDE JAVA

### Inserción documentos (CREATE)

- Inserción de un array dentro de un documento:

```
//Insercion de un array de string
Document main = new Document();
main.put("_id", 4);
main.put("nombre", "Pilar");
main.put("telefono", 678944);
List <String> lista = new ArrayList <String>();
lista.add("dao");
lista.add("java");
main.put("modulos", lista);
coleccion.insertOne(main);
```

```
{"_id": 4, "nombre": "Pilar", "telefono": 678944, "modulos": ["dao", "java"]}
```

## 5. ACCESO MONGODB DESDE JAVA

### Inserción documentos (CREATE)

- Inserción de un array de objetos dentro de un documento:

```
//Insercion de un array de objetos
Document main = new Document();
main.put("_id", 5);
main.put("nombre", "Juan");
main.put("telefono", 678944);
BasicDBList dbl = new BasicDBList();
dbl.add(new BasicDBObject("informatica", "DAM2"));
dbl.add(new BasicDBObject("fol", "ASIX1"));
main.put("modulos", dbl);
coleccion.insertOne(main);
```

```
{"_id": 5, "nombre": "Juan", "telefono": 678944, "modulos": [{"informatica": "DAM2"}, {"fol": "ASIX1"}]}
```

## 5. ACCESO MONGODB DESDE JAVA

### Inserción documentos (CREATE)

- Inserción mezcla de array y un dato normal:

```
//Insercion mezclada atributos y array
Document main = new Document();
main.put("_id", 6);
main.put("nombre", "Lucas");
main.put("telefono", 4478944);
BasicDBList dbl = new BasicDBList();
dbl.add(new BasicDBObject("informatica", "DAM2"));
dbl.add(new BasicDBObject("fo1", "ASIX1"));
BasicDBObject outer=new BasicDBObject("institut", "ies").append("items", dbl);
main.put("modulos", outer);
coleccion.insertOne(main);
```

```
{"_id": 6, "nombre": "Lucas", "telefono": 4478944,
```

```
"modulos": {"institut": "ies", "items": [{"informatica": "DAM2"}, {"fo1": "ASIX1"}]}}
```

## 5. ACCESO MONGODB DESDE JAVA

### Modificación documentos (UPDATE)

- Modificación de un documento:

```
{"_id": 3, "nombre": "pedro", "edad": 30}
```

```
BasicDBObject query = new BasicDBObject();  
query.put("_id", 3);  
  
BasicDBObject newDocument = new BasicDBObject();  
newDocument.put("nombre", "Mateo");  
  
BasicDBObject updateObject = new BasicDBObject();  
updateObject.put("$set", newDocument);  
  
coleccion.updateOne(query, updateObject);
```

```
{"_id": 3, "nombre": "Mateo", "edad": 30}
```

## 5. ACCESO MONGODB DESDE JAVA

### Modificación documentos (UPDATE)

- Modificación de muchos documentos:

```
{ "_id": 2, "nombre": "erer", "edad": 30 }  
{ "_id": 3, "nombre": "Mateo", "edad": 30 }
```

```
BasicDBObject searchQuery = new BasicDBObject();  
//searchQuery.append("", "");  
  
BasicDBObject updateQuery = new BasicDBObject();  
updateQuery.append("$set", new BasicDBObject().append("edad", "100"));  
  
coleccion.updateMany(searchQuery, updateQuery);
```

```
{ "_id": 2, "nombre": "erer", "edad": "100" }  
{ "_id": 3, "nombre": "Mateo", "edad": "100" }
```



## 5. ACCESO MONGODB DESDE JAVA

---

### Borrado de documentos (DELETE)

- Borrado de todos los documentos de una colección usando un objeto BasicDBObject en blanco:

```
BasicDBObject document = new BasicDBObject();  
coleccion.deleteMany(document);
```

- Borrado de un documento concreto:

```
coleccion.deleteOne(new Document("_id", id));
```

## 6. PRACTICA CRUD CON MONGO

- Realiza un CRUD en Java que acceda a la base de datos mongoDB.
- Debe de realizar las siguientes funcionalidades:

```
//CRUD--Create, Read, Update, Delete
boolean fin=false;
while (!fin) {
    System.out.println("1. Lista las bases de datos de Mongodb");
    System.out.println("2. Lista las colecciones de una base datos");
    System.out.println("3. Visualiza todos los documentos de profesores");
    System.out.println("4. Inserta un nuevo documento profesor");
    System.out.println("5. Inserta un documento profesor con un objeto aula");
    System.out.println("6. Modifica un documento profesor");
    System.out.println("7. Borra un documento profesor por id");
    System.out.println("8. Borra todos los documentos profesor");
    System.out.println("9. Consulta con alguna funcion de agregación");
    System.out.println("10. Vuelca todos los documentos en un fichero de texto");
    System.out.println("11. Salir");
    System.out.print("Introduce que opcion quieres?");
```