

Laboratorio Nro 09 - Instalación y Gestión de una base de datos MongoDB

Universidad Privada de Tacna

Escuela Profesional de Ingeniería de Sistemas

Base de DATos II

Robles Flores , Anthony Richard (2016056192)

Tacna, Perú

1. INFORMACIÓN GENERAL

Objetivos:

- Realizar la instalacion de Base de Datos NoSQL MongoDB
- Importar un archivo JSON con colecciones de datos a MongoDB
- Realizar consultas de registros con MongoDB

Requerimientos

Hardware

- Virtualization activada en el BIOS.
- CPU SLAT capable feature.
- Al menos 4GB de RAM.

Software

- Windows 10 64 bit
- Docker Desktop

2. DESARROLLO DE LABORATORIO

Parte I : Instalación de MongoDB (seguir los pasos)

- a) Ingresar a la URL (<https://www.mongodb.com/download-center/community>) para descargar el instalador que nos ofrece en la pagina.

Select the server you would like to run:

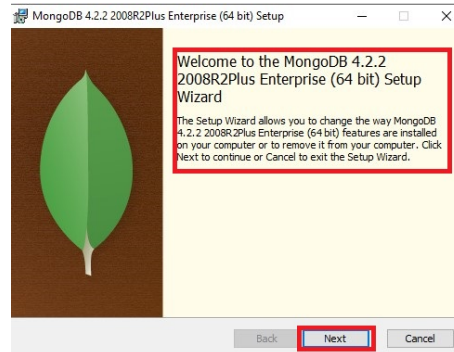
MongoDB Community Server
FEATURE RICH. DEVELOPER READY.

Version: 4.2.2 (current release) OS: Windows x64 x64

Package: MSI

Download

- b) Iniciar el instalador como administrador de windows.



- c) Crear las carpetas de almacenamiento y configuración de MongoDB. Dar permisos escritura y lectura a estas carpetas. Tal como se ve en la imagen C:/data - C:/data/db

Archivos de programa	10/12/2019 16:42	Carpeta de archivos
Archivos de programa (x86)	30/10/2019 16:55	Carpeta de archivos
data	10/12/2019 16:48	Carpeta de archivos

- d) Iniciar el servidor del servicio de MongoDB: mongod. Primero dirigir a la siguiente ruta (C:/Program Files/MongoDB/Server/4.2/bin). Y ejecutar el comando mongod:

```
C:\Users\AGONZALES>cd "C:\Program Files\MongoDB\Server\4.2\bin"
C:\Program Files\MongoDB\Server\4.2\bin>mongod
2019-12-12T11:43:20.490-0500 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslD
2019-12-12T11:43:20.500-0500 I CONTROL [initandlisten] MongoDB starting : pid=9972 port=27017 dbpath=C:\data\db\ 64-b
2019-12-12T11:43:20.501-0500 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2019-12-12T11:43:20.502-0500 I CONTROL [initandlisten] db version v4.2.2
2019-12-12T11:43:20.502-0500 I CONTROL [initandlisten] git version: a0bbbff6ada159e19298d37946ac8dc4b497eadf
2019-12-12T11:43:20.502-0500 I CONTROL [initandlisten] allocator: tcmalloc
2019-12-12T11:43:20.502-0500 I CONTROL [initandlisten] modules: enterprise
2019-12-12T11:43:20.502-0500 I CONTROL [initandlisten] build environment:
2019-12-12T11:43:20.503-0500 I CONTROL [initandlisten] distmod: windows-64
2019-12-12T11:43:20.503-0500 I CONTROL [initandlisten] distarch: x86_64
2019-12-12T11:43:20.503-0500 I CONTROL [initandlisten] target_arch: x86_64
2019-12-12T11:43:20.503-0500 I CONTROL [initandlisten] options: {}
2019-12-12T11:43:20.518-0500 I STORAGE [initandlisten] Detected data files in C:\data\db\ created by the 'wiredTiger'
```

- e) Iniciar el shell de mongo : mongo.exe

```
C:\Program Files\MongoDB\Server\4.2\bin>mongo.exe
MongoDB shell version v4.2.2
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("fb7568b7-fc8d-45fd-a157-8c2e46b26122") }
MongoDB server version: 4.2.2
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
  http://docs.mongodb.org/
Questions? Try the support group
  http://groups.google.com/group/mongodb-user
Server has startup warnings:
2019-12-11T22:49:03.127-0500 I CONTROL [initandlisten]
2019-12-11T22:49:03.128-0500 I CONTROL [initandlisten] ** WARNING: Access control is not enabled.
2019-12-11T22:49:03.128-0500 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-11T22:49:03.128-0500 I CONTROL [initandlisten]
MongoDB Enterprise > _
```

Parte II: Importar Datos

- f) Para importar datos utilizaremos el comando import. Este comando es capaz de importar datos en distintos formatos a nuestra base de datos.
(mongoimport.exe -host localhost -port 27017 -db test -collection people -type json -file C:/Users/AGONZALES/Documents/mongodb-consultas.json)

```
C:\Program Files\MongoDB\Server\4.2\bin>mongoimport.exe --host localhost --port 27017 --db test --collection people
--type json --file C:\Users\AGONZALES\Documents\mongodb-consultas.json
2019-12-12T14:04:54.112-0500 connected to: mongodb://localhost:27017/
2019-12-12T14:04:54.196-0500 23 document(s) imported successfully. 0 document(s) failed to import.
```

- g) Con este comando, lo que haremos será incorporar los datos a nuestro servidor, que está en el puerto 27017, concretamente a una base de datos llamada test y a una colección que hemos llamado people.

Parte III : Consultas

- h) Iniciar el shell de mongo mongo.exe.

```
C:\Program Files\MongoDB\Server\4.2\bin>mongo.exe
MongoDB shell version v4.2.2
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("fb7568b7-fc8d-45fd-a157-8c2e46b26122") }
MongoDB server version: 4.2.2
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
  http://docs.mongodb.org/
Questions? Try the support group
  http://groups.google.com/group/mongodb-user
Server has startup warnings:
2019-12-11T22:49:03.127-0500 I  CONTROL  [initandlisten]
2019-12-11T22:49:03.128-0500 I  CONTROL  [initandlisten] ** WARNING: Access control is not
2019-12-11T22:49:03.128-0500 I  CONTROL  [initandlisten] **           Read and write access
unrestricted.
2019-12-11T22:49:03.128-0500 I  CONTROL  [initandlisten]
MongoDB Enterprise >
```

- i) Para realizar consultas a la base de datos, deberemos usar el comando `db.nombredecoleccion.find()`. Este comando puede recibir dos parámetros: una consulta y una proyección. Ambos comandos son opcionales por lo que si ejecutamos el comando: `'db.people.find()'`.

```
MongoDB Enterprise > db.people.find()
{ "_id" : ObjectId("5d7283280e46ca2940f6ca22"), "isActive" : true, "balance" : "$3,841.00",
  "company" : "Steganoconiche", "phone" : "876-438-29861", "email" : "melanie@steganoconiche.com",
  "commodo aliqua tempor. Quis ipsum exercitation irure esse aute eiusmod anim consequat dolor
  nt laborum cupidatat amet sunt dolore elit sit amet officia excepteur. Labore commodo elit
  Dolore ad id nisi amet.\r\n", "registered" : "1999-12-29T09:36:57 -01:00", "latitude" : 44
  ds" : [ { "id" : 1, "name" : "Madelyn Watson" }, { "id" : 2, "name" : "Valeria Oswald" }, {
  m" : "student" }
```

- j) Obtendremos una larga lista con los 20 elementos de la colección. Al ejecutar el comando podemos ver que el resultado no está demasiado formateado, por lo que es muy difícil leerlo. Para solucionar este problema podemos usar el modificador `pretty` que nos devolverá un resultado mucho más legible `'db.people.find().pretty()'`.

```

MongoDB Enterprise > db.people.find().pretty()
{
  "_id" : ObjectId("5df28328be46ca2940f6ca22"),
  "isActive" : true,
  "balance" : "$3,841.00",
  "picture" : "http://placeholder.it/32x32",
  "age" : 23,
  "name" : "Melanie Bush",
  "company" : "Steganoconiche",
  "phone" : "876-438-29861",
  "email" : "melanie@steganoconiche.com",
  "address" : "16899, Columbia, Stanton Streets",
  "about" : "Sunt elit commodo aliqua tempor. Quis  

  isicing adipisicing nisi. Sunt laborum cupidatat amet su  

  n incididunt nisi amet irure. Dolore ad id nisi amet.\r\n",
  "registered" : "1999-12-29T09:36:57 -01:00",
  "latitude" : 44,
  "tags" : [
    "sit",
    "esse",
    "dolore",
    "ad",
    "magna",
    "velit",
    "qui"
  ],
  "friends" : [
    {
      "id" : 1,
      "name" : "Madelyn Watson"
    }
  ]
}

```

- k) Ahora vamos a añadir la consulta al comando find, para que filtre los elementos según nuestras necesidades. Para ello especificaremos un objeto JSON como primer parámetro del comando, con los campos por los que queremos filtrar:

```
db.people.find( age:34 ).pretty()
```

```

MongoDB Enterprise > db.people.find({age:34}).pretty()
{
  "_id" : ObjectId("5df28e94a248515d14b836e7"),
  "isActive" : false,
  "balance" : "$3,131.00",
  "picture" : "http://placeholder.it/32x32",
  "age" : 34,
  "name" : "Morgan Cook",
  "company" : "Robocomm",
  "phone" : "816-455-37785",
  "email" : "morgan@robocomm.com",
  "address" : "25266, Cincinnati, Grand Street",
  "about" : "Mollit dolore enim Lorem et irure q  

  eserunt dolor incididunt tempor cillum consequat ulla

```

- l) Con ese comando obtendremos las personas cuya edad es de 34 años. Podemos añadir tantos filtros como queramos.

```
db.people.find( age:34,isActive:true ).pretty()
```

```
MongoDB Enterprise > db.people.find({age:34,isActive:true}).pretty()
{
  "_id" : ObjectId("5df297add08dd97f3e32aa8c"),
  "isActive" : true,
  "balance" : "$1,066.00",
  "picture" : "http://placeholder.it/32x32",
  "age" : 34,
  "name" : "Julia Young",
  "company" : "Genland",
  "phone" : "831-524-30334",
  "email" : "julia@genland.com",
  "address" : "18282, ChulaVista, Bleecker Street",
  "about" : "Nisi excepteur Lorem mollit amet amet aliqua non c
update aliquip sit voluptate nisi cillum pariatur. Incidunt amet co
  "registered" : "2005-10-27T05:30:21 -02:00",
  "latitude" : -18,
```

- m) En este caso filtramos por age y por isActive. Como se ve los resultados nos muestran todos los campos de cada elemento. Es como si hubiésemos utilizado el asterisco en una consulta SELECT. Si queremos solo algunos de los campos, deberemos utilizar el segundo parámetro de la consulta find para definir una proyección.

```
db.people.find( age:34,isActive:true, name:1,age:1,isActive:1 ).pretty()
```

```
MongoDB Enterprise > db.people.find({age:34,isActive:true},{name:1, age:1, isActive:1}).pretty()
{
  "_id" : ObjectId("5df297add08dd97f3e32aa8c"),
  "isActive" : true,
  "age" : 34,
  "name" : "Julia Young"
}
```

- n) Que no devuelva solo los campos que se requieren, además del id. El id por defecto se muestra siempre, así que si queremos ocultarlo hay que especificarlo en la proyección.

```
db.people.find( age:34,isActive:true, name:1,age:1,isActive:1,id:0
```

```
MongoDB Enterprise > db.people.find({age:34,isActive:true},{name:1, age:1, isActive:1, id:0}).pretty()
{ "isActive" : true, "age" : 34, "name" : "Julia Young" }
```

- ñ) Si queremos mostrar todos los campos, pero quitando solo algunos, lo que haremos será desactivar los no deseados en la proyección:

```
db.people.find(
age:34,isActive:true,
name:0,age:0,isActive:0,id:0).pretty()
```



```
MongoDB Enterprise > db.people.find({age:34,isActive:true},{name:0, age:0, isActive:0,_id:0}).pretty()
{
  "balance" : "$1,066.00",
  "picture" : "http://placeholder.it/32x32",
  "company" : "Genland",
  "phone" : "831-524-30334",
  "email" : "julia@genland.com",
  "address" : "18282, ChulaVista, Bleecker Street",
  "about" : "Nisi excepteur Lorem mollit amet amet aliqua non consectetur adipisicing ex eu ex con",
  "update" : "aliquip sit voluptate nisi cillum pariatur. Incididunt amet commodo laboris sit.\r\n",
  "registered" : "2005-10-27T05:30:21 -02:00",
  "latitude" : -18,
  "tags" : [
```

- o) El comando `findOne` tiene el mismo funcionamiento que el comando `find`, con la diferencia de que, si el comando encuentra más de un resultado que cumpla las condiciones de la consulta, tan solo nos devolverá el primero.
- `db.people.findOne(age:34,isActive:true,name:0,age:0,isActive:0,id:0)`

```
MongoDB Enterprise > db.people.findOne({age:34,isActive:true},{name:0, age:0, isActive:0,_id:0}).pretty()
2019-12-12T14:55:38.924-0500 E QUERY [js] uncaught exception: TypeError: db.people.findOne(...).pretty()
@ (shell) 1:1
MongoDB Enterprise > db.people.findOne({age:34,isActive:true},{name:0, age:0, isActive:0,_id:0})
{
  "balance" : "$1,066.00",
  "picture" : "http://placeholder.it/32x32",
  "company" : "Genland",
  "phone" : "831-524-30334",
  "email" : "julia@genland.com",
  "address" : "18282, ChulaVista, Bleecker Street",
  "about" : "Nisi excepteur Lorem mollit amet amet aliqua non consectetur adipisicing ex eu ex con",
  "update" : "aliquip sit voluptate nisi cillum pariatur. Incididunt amet commodo laboris sit.\r\n",
  "registered" : "2005-10-27T05:30:21 -02:00",
  "latitude" : -18,
  "tags" : [
    "in",
    "id",
    "sit",
    "labore",
    "ipsum"
```

- p) Además `findOne` no acepta `pretty`, pero ya devuelve el resultado formateado.

```
MongoDB Enterprise > db.people.findOne({age:34,isActive:true},{name:0, age:0, isActive:0,_id:0}).pretty()
2019-12-12T14:55:38.924-0500 E QUERY [js] uncaught exception: TypeError: db.people.findOne(...).pretty is not a function :
```

- q) En las consultas, en cambio, no se ha especificado dichas comillas. Esto es porque el motor JavaScript de MongoDB se encarga de añadirlas. Esto nos facilita la escritura de consultas, ya que no son obligatorias. De hecho, la siguiente consulta, funcionará perfectamente:
- `db.people.findOne(`
`"age":34,"isActive":true,`

“name”:0,”age”:0,”isActive”:0,”id”:0)

```
MongoDB Enterprise > db.people.findOne({"age":34,"isActive":true},{"name":0, "age":0, "isActive":0,"_id":0})
{
  "balance" : "$1,066.00",
  "picture" : "http://placeholder.it/32x32",
  "company" : "Genland",
  "phone" : "831-524-30334",
  "email" : "julia@genland.com",
  "address" : "18282, ChulaVista, Bleecker Street",
  "about" : "Nisi excepteur Lorem mollit amet amet aliqua non consectetur adipisicing ex eu ex consequ
update aliquip sit voluptate nisi cillum pariatur. Incididunt amet commodo laboris sit.\r\n",
  "registered" : "2005-10-27T05:30:21 -02:00",
  "latitude" : -18,
  "tags" : [
    "in",
    "id",
    "sit",
    "labore",
    "irure",
    "in",
    "aute"
  ],
  "friends" : [
    {
      "id" : 1,
      "name" : "Evelyn Davidson"
    },
  ],
}
```