José A. Clemente

01

MongoDB

mongoDB

Base de datos NoSQL (Not only SQL) orientada a documentos.

Entre sus principales caracteristicas esta:

- SchemaLess (sin esquema). Datos sin estructura predefinida. Podemos agregar o quitar propiedades de ellos.
- Utiliza JSON como formato de datos.
- Escalabilidad horizontal.

https://www.mongodb.com/es

mongoDB

- Colecciones
- Documentos
- Campos
- Documentos embebidos

Base de datos relacionales:

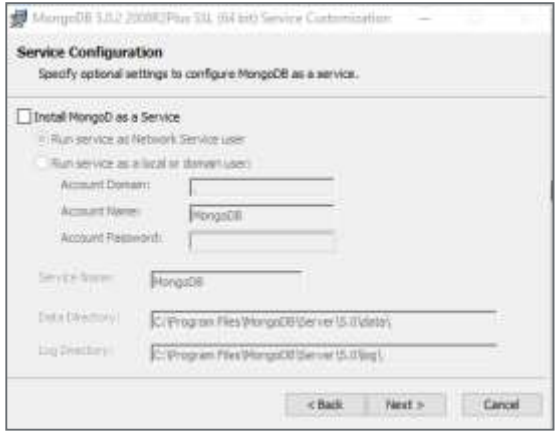
- Tablas.
- Filas
- Columnas
- Joins

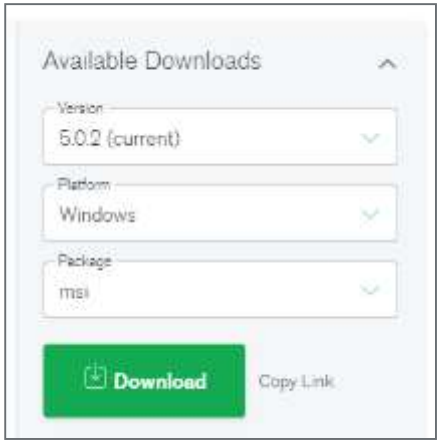
02

INSTALLATION

INSTALLATION (local installation)

1. https://www.mongodb.com/es
2. Software / Community Server
3. On-premises (MongoDB locally) → MongoDB Community Server.
4. Complete Installation or Custom da igual
5. Hay dos opciones instalarlo como servicio o como programa.





INSTALLATION (local installation)

6. Posibilidad de instalar tambien MongoDB Compass. Compass es la interfaz grafica. Antes Compass y Server se instalaban por separado. Compass se instalaba como Tool.
7. Al no estar como servicio hace falta arrancar el servidor (mongod.exe)
8. C:\Program Files\MongoDB\Server\5.0\bin
9. Si abrimos y se cierra es porque falta algo de configuración.
10. En C:\ falta carpeta data y dentro carpeta db (lo necesita Windows para trabajar con MongDB)
11. Ahora al ejecutar mongod.exe se mantiene la consola abierta. Debe estar siempre abierta para que el servidor este en ejecución.

INSTALLATION (Test)

1. Mongod se mantiene abierto desde el directorio de instalación.
2. Tambien podemos ejecutar mongod --version desde cualquier ubicación. Esto devolvera resultado si hemos introducido en el path esta ruta.
3. Lanzar mongo.exe, ver que se mantiene abierto y ejecutar algun comando para ver el resultado.
4. Comando help devuelve una serie de comandos de ayuda.
5. Comando show dbs; Muestra las 3 bases de datos que vienen por defecto en la configuración (admin, config y local)
6. Show logs; muestra los logs
7. Use local; indico la base de datos a usar. En este caso local.
8. Show collections; muestra las colecciones de elementos dentro de una base de datos.

Directorio de instalación

C:\Program Files\MongoDB\Server\[version]\bin

Tenemos varias cosas en este directorio:

mongod.exe: Servicio de MongoDB. Necesario que este en marcha.

mongo.exe: Consola (shell) de MongoDB. Nos permite trabajar con las bases de datos. Se conecta con el server.

mongod

Es necesario arrancarlo previamente a trabajar con MongoDB. Esto deja el servicio de MongoDB en segundo plano y activo. Equivale a realizar la instalación de MongoDB como servicio.

Por tanto, es necesario que este arrancado para que las bases de datos funcionen.
¿Como agregar al path?
Este equipo, botón derecho Propiedades. Vamos a Configuración Avanzada del Sistema. Dentro de la pestaña Opciones avanzadas, vamos a variables de entorno. En la tabla de variables de usuario para el usuario la agregamos.

Ahora por ejemplo mongod --version estara devolvera resultado al estar accesible desde cualquier directorio.

PATH

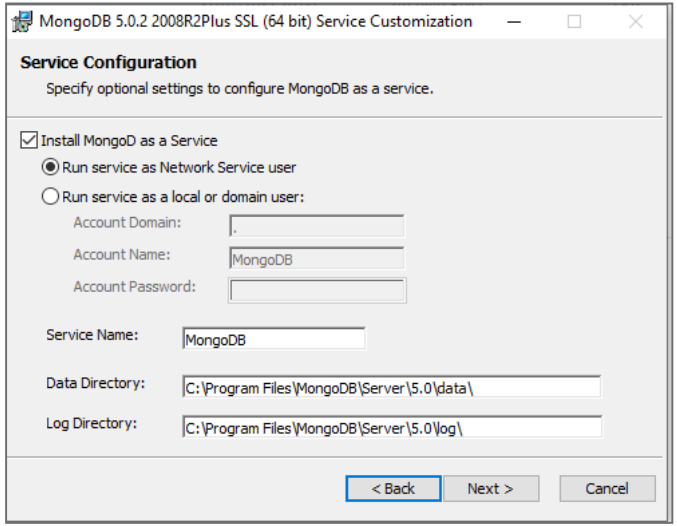
El path son algunas rutas a las cuales el S.O. tendra o tiene necesidad de acceder desde cualquier parte. Sin necesidad de estar en esas rutas.

¿Como agregar al path?
Este equipo, botón derecho Propiedades. Vamos a Configuración Avanzada del Sistema. Dentro de la pestaña Opciones avanzadas, vamos a variables de entorno. En la tabla de variables de usuario para el usuario la agregamos.

Ahora por ejemplo mongod --version estara devolvera resultado al estar accesible desde cualquier directorio.

INSTALLATION (as a service)

1. Seleccionar Install MongoD as a Service.
2. No hara falta arrancar mongod cada vez que se quiera trabajar con MongoDB.
3. Si vamos a servicios veremos el servicio de MongoDB.
4. Podemos gestionarlo desde Servicios.
5. Cuando instalamos como servicio no necesitamos las carpetas /data/db en C:/. Se genera una carpeta data que cuelga de: C:\Program Files\MongoDB\Server\5.0



INSTALLATION (Test)

1. Abrir shell de MongoDB (mongo.exe) y ejecutar alguno de sus comandos (show dbs;show logs;)

Ventaja de instalar como servicio

Cuando arranquemos el equipo ya estara disponible el servicio.
Util si vamos a trabajar todos los días con MongoDB. Así estaria siempre listo para escuchar.

En caso contrario, mejor no tenerlo como servicio. O por lo menos que no se arranque al inicio porque consume recursos.

03

BASIC CONCEPTS

Conceptos básicos (I)

Base de datos.Programa que guarda datos relacionados con un mismo contexto. Pero no vale guardar por guardar, los datos deben estar ordenados. Ahí surge el concepto de **colecciones**.

Las divisiones de los datos son **colecciones**. Dentro de un contexto de tienda online tendriamos: coleccion de usuarios, productos, vendedores, etc. Equivalente a las tablas en una base de datos SQL.

¿Que se guarda en las colecciones? Los datos se guardan en formato **JSON**. Por tanto, se guardan **documentos**.

Conceptos básicos (II)

Shell de MongoDB. Ademas de permitirnos interactuar con el servidor, es un interprete de JavaScript. Podemos crear variables y asignar valores. Se pueden usar librerias JS.

```
> Math.sin(Math.PI / 2)
1
> new Date()
ISODate("2021-08-08T05:44:36.959Z")
```

Tambien se pueden crear funciones. Por ejemplo, la función factorial:

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> function factorial (n) { if (n <= 1) return 1; return n * factorial(n-1) }
> factorial(3)
6
> factorial(4)
24
>
```

Conceptos básicos (III)

Crear colecciones. Sobre una de datos vacia creo una colección.

```
> use webstore
switched to db webstore
> show collections
> db.createCollection("users")
{ "ok" : 1 }
> show collections
users
>
```

```
> use webstore
switched to db webstore
> show collections
users
> db.products.insert({
...     "nombre": "Lenovo ThinkPad",
...     "precio": 1100,
...     "active": false,
...     "created_At": new Date(),
...     "features": {
...         "name": "pc01",
...         "S.O.": "Windows 10",
...         "location": {
...             "city": "Gandia",
...             "community": "Valencia"
...         },
...         "other_data": [1, "hola", true]
...     }
... })
WriteResult({ "nInserted" : 1 })
> show collections
products
users
>
```

Conceptos básicos (III)

Visualizar colecciones. Sobre una de datos vacia creo una colección.

```
> show collections
products
users
> db.products.find()
{ "_id" : ObjectId("610f748aca799cf7685a35b1"), "nombre" : "Lenovo ThinkPad", "precio" : 1100, "active" : false, "created_At"
 : ISODate("2021-08-08T06:07:06.630Z"), "features" : { "name" : "pc01", "S.O." : "Windows 10", "location" : { "city" : "Gandi
a", "community" : "Valencia" }, "other_data" : [ 1, "hola", true ] } }
> db.products.find().pretty()
{
        "_id" : ObjectId("610f748aca799cf7685a35b1"),
        "nombre" : "Lenovo ThinkPad",
        "precio" : 1100,
        "active" : false,
        "created_At" : ISODate("2021-08-08T06:07:06.630Z"),
        "features" : {
                "name" : "pc01",
                "S.O." : "Windows 10",
                "location" : {
                        "city" : "Gandia",
                        "community" : "Valencia"
                },
                "other_data" : [
                        1,
                        "hola",
                        true
                ]
        }
}
```

07

MapReduce

MapReduce

Framework que nos permite procesar de manera paralela una gran cantidad de datos.

Framework más usado para proceso de datos en Big Data. Soportado por:

- Hadoop
- Cassandra
- CouchDB
- MongoDB

MapReduce

Podemos implementarlo con JavaScript. No hace falta compilar nada. Los inicios de MapReduce fueron en Google (calcular page range de una página).
Se divide en 2 fases:

- Map. Aplicado en paralelo para cada uno de los registros a procesar. Agrupa por clave-valor.

- Reduce. Esta función pasando un parametro como clave, nos devuelve un listado de los valores por clave.

MapReduce

1. Map. Este proceso hace un split por cada una de las claves indicadas y devuelve un conjunto clave-valor.
2. Sort. El framework aplica de forma innata una ordenación.
3. Reduce. Procesa por cada una de las claves todos los valores pasados.

Nos permite procesar una gran cantidad de registros muy rapido.

MapReduce

```
{
    ‘cliente’: ‘c001’,
    ‘lineas’: [
        { ‘id’: ‘pantalon’, ‘unidades’: 4, ‘importe’: 10 },
        { ‘id’: ‘calcetin’, ‘unidades’: 2, ‘importe’: 6 }
    ]
}
```

Podemos ejecutar mapReduce o aplicarlo desde la shell o bien en un script JS (mapReduce.js,p. ej.).

MapReduce

```
var mapFunction = function() {
    for (var i = 0; i<= this.lineas.length; i++) {
        var key = this.lineas[i].id;
        var value = { subtotal: this.lineas[i].unidades };

        //Función emit para agregar un valor a la clave
        emit(key, value);
    }
}
```

Con el emit no se van guardando. Las va cogiendo al vuelo y ya las tiene. Me saca la clave-valor que me interesa (id y unidades). Podrian haber sido otras. Podemos aplicar MapReduce sobre nuestros conjuntos de datos tantas veces como queramos. Creando salidas nuevas (o colecciones cada vez).

MapReduce

```
var reduceFunction= function(id, countObjVals) {
	reduceVal = { total: 0 }

	for (var i = 0; i<= countObjVals.length; i++) {
		reduceVal.total += countObjVals[i].subtotal;
	}
	return reduceVal ;
}
```

Esta función incrementa el numero de unidades para el id especificado.

MapReduce

Falta ejecutarlo. Se puede hacer desde la shell o con el script creado. Por ejemplo, con esta función:

db.facturas.mapReduce(mapFunction, reduceFunction, { out: ‘map_reduce_result’ })

Se ejecuta sobre una colección (faccturas). Tercer parametro indica donde quiero el resultado. En este caso será una nueva colección llamada ‘map_reduce_result’

MapReduce

¿Cómo ejecuto el script?

mongo 127.0.0.1/facturas mapreduce.js

Si nos conectamos a la base de datos y hacemos un show collection, veremos la nueva colección.

Si quiero ver en detalle mi colección: db.map_reduce_result.find()

THANKS!

Do you have any questions?