Unidad 7. DESARROLLO AVANZADO DE BACKEND

Unidad 7: CRUD y Manejo de Bases de Datos con MongoDB

Objetivos de la clase

- Comprender el concepto de una base de datos
- Comprender las diferencias entre bases de datos relacionales y no relaciones
- Comprender el concepto de MongoDB y su respectiva instalación
- Comprender el concepto de CRUD
- Aplicar el CRUD a la base de datos de MongoDB

CLASE N°11

Glosario

Websocket: Protocolo de comunicación que permite una sesión activa entre cliente y servidor

Sweetalert2: Librería que permite utilizar alertas con presentación profesional. Sirve para autenticaciones.

Autenticación : proceso por el cual el usuario tiene que identificarse para poder hacer uso de algún servicio.

Deploy: Desplegar una aplicación para pasar de un entorno local a la nube, con el fin de que los usuarios puedan acceder a dicha aplicación.

Github: Es una forja, o una plataforma de desarrollo colaborativo, donde podemos alojar nuestro repositorio con nuestra aplicación

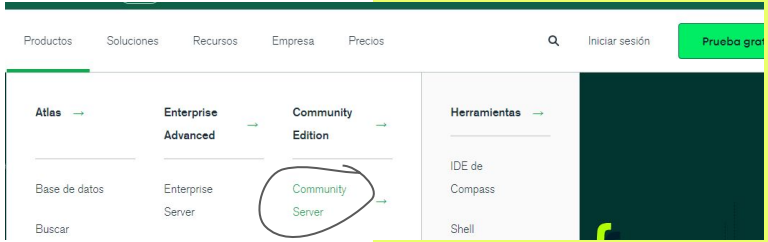
Glitch.com : Página utilizada para poder hacer deploy de nuestra aplicación, a partir de su conexión con Github

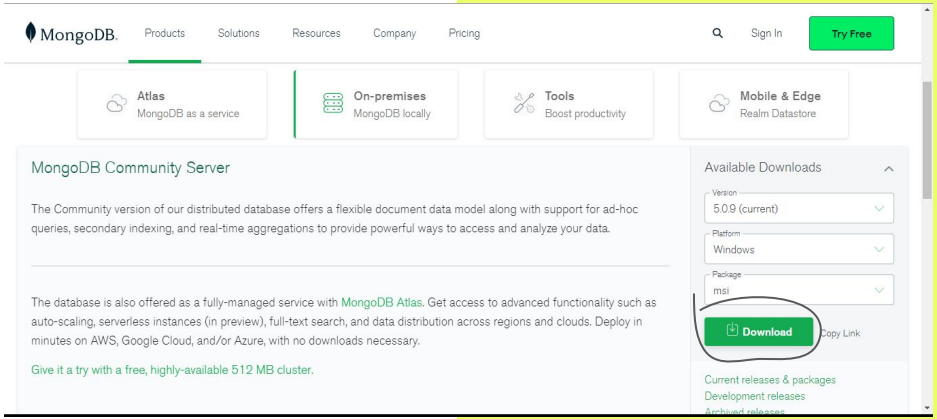
Instalar MongoDB

APROXIMACIÓN AL PROCESO

1. Visitar la página oficial

- ✓ Comenzamos instalando el **community Server** que se encuentra en el apartado de **Productos**, en la pestaña de **Community Edition.**
- ✓ Seleccionamos nuestro sistema operativo y presionamos el botón de descarga.





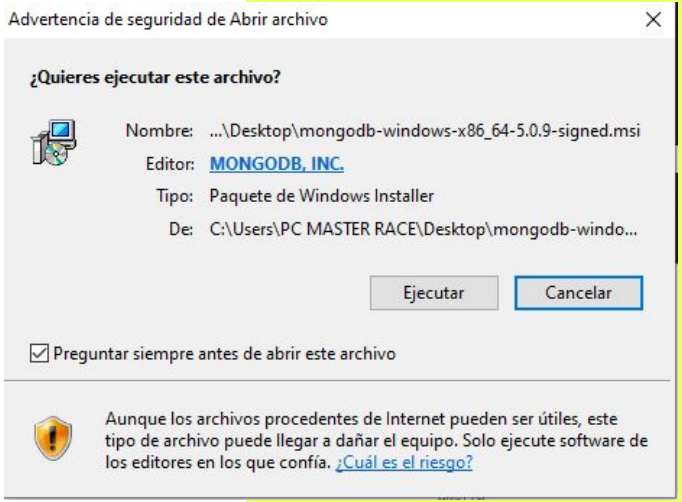
CODERHOUSE

APROXIMACIÓN AL PROCESO

2. Ejecutar instalador con permisos de administrador

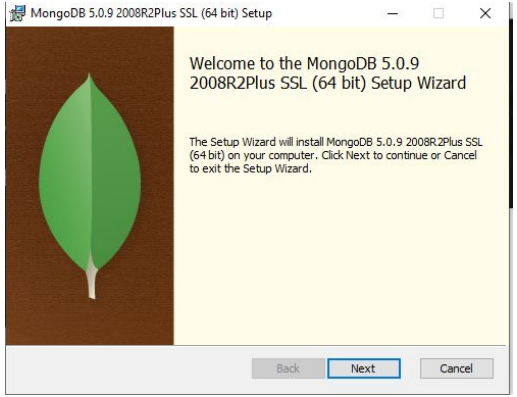
Recuerda que la razón por la cual se requiere ejecutar con permisos de administrador, es para poder evitar cualquier posible conflicto de permisos que pueda ocurrir durante el proceso de instalación.

El resto del proceso será sólo seguir el wizard de instalación

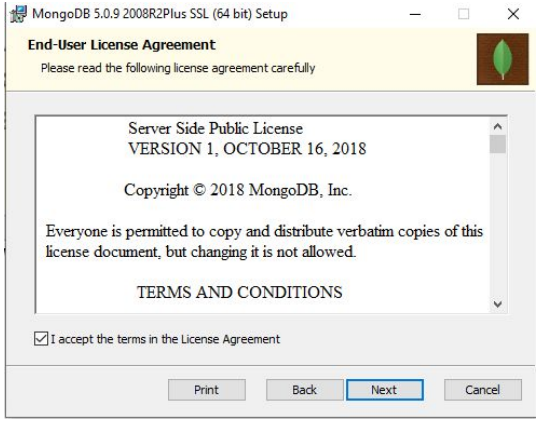


CODERHOUSE

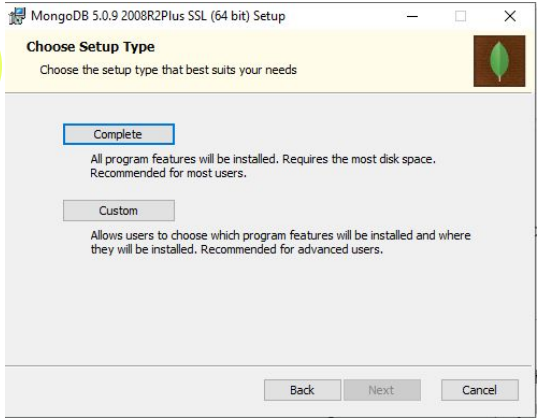
1



2

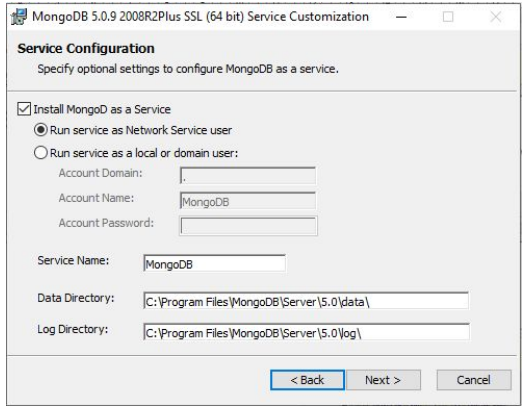


3



Recuerda seleccionar la instalación completa

4



Se recomienda instalar como servicio

Ahora que ya instalamos MongoDB...

probemos comandos que te pueden resultar útiles

CODERHOUSE

Comandos iniciales: creación

```
> show dbs
admin           0.000GB
config          0.000GB
```

```
> use miPrimeraBase
switched to db miPrimeraBase
>
```

```
> db.createCollection('usuarios')
{ "ok" : 1 }
>
```

- ***show dbs*** mostrará la lista de bases de datos que tengamos activas.
- ***use <nombre>*** crea una nueva base de datos (en caso de no existir) y se posiciona en ella para utilizarla.
- ***db.createCollection(nombre)*** creará nuestra primera colección para comenzar a ingresar documentos.

Insertando un dato en la colección “usuarios”

Una vez que sabemos qué colección vamos a utilizar, utilizaremos el comando “insertOne”, pasando el objeto que queremos guardar en dicha colección:

```
> db.usuarios.insertOne({name:"Miguel",last_name:"Espinosa",age:30,email:"correoMiguel@hotmail.com"});
{
        "acknowledged" : true,
        "insertedId" : ObjectId("62b8c5e7f41150bf5c4b9ae4")
}
```

¡Nota lo sencillo que es! la sintaxis base.colección.método es bastante fácil de recordar, porque nos recuerdan a los métodos que solemos utilizar al trabajar con objetos y clases en javascript. Además, el documento insertado es un objeto con propiedades en clave-valor, lo cual hace que podamos visualizar y entender más fácil qué es lo que estamos insertando.
Ahora falta visualizar la información.

Leyendo información de nuestra base de datos

Utilizando la misma sintaxis que un objeto o una clase de javascript, leemos la colección con el método ***find()***

```
> db.usuarios.find()
{ "_id" : ObjectId("62b8c5e7f41150bf5c4b9ae4"), "name" : "Miguel", "last_name" : "Espinosa", "age" : 30, "email"
eoMiguel@hotmail.com" }
```

Notamos cómo el resultado es el documento que recién ingresamos a la colección. ¡Sin archivos, sin complicaciones!

Además, MongoDB nos autogestiona un id, mismo que nos asegurará contar con una clave irrepetible, evitando que tengamos que programar nuestra propia gestión de identificadores.

Primeros pasos con Mongo

Duración: **10-15min**

CODERHOUSE

ACTIVIDAD EN CLASE

Primeros pasos con Mongo

- ✓ Una vez que corrobores que mongo está instalado en computador, a partir del cliente CLI, crear una base de datos de nombre “estudiantes”
- ✓ Agregar 5 estudiantes diferentes con los campos “nombre”, “apellido”, “curso”, “correo”. Puedes utilizar db.collection.insertMany()
- ✓ Una vez agregados, listar a los estudiantes de dicha colección y corroborar su persistencia.

Ejemplo en vivo

Se creará una base de datos llamada “baseCRUD”.

- ✓ Se agregará una colección llamada “mascotas”
- ✓ Se agregarán 3 mascotas con las propiedades: nombre, especie, edad
- ✓ Se buscarán mascotas por su especie
- ✓ Contar el número de mascotas totales agregadas.

Tiempo estimado: **10 minutos**

Para pensar

¿Inconsistencia de datos?
En diapositivas anteriores se utilizó el término “conteo estimado”, lo cual hace referencia a una cifra que puede no ser igual al valor real.

¿Por qué obtendríamos un valor estimado? ¿Podemos asegurar que el valor siempre será verdadero?
Contesta en el chat de zoom

CRUD – CR

Practicarás los primeros dos elementos del CRUD

Duración: **10-15 min**

CODERHOUSE

ACTIVIDAD EN CLASE

CRUD – CR

Sobre una base de datos llamada “colegio”, crear una colección “estudiantes” donde se agregarán documentos con los siguientes datos:

- ✓ nombre
- ✓ apellido
- ✓ curso
- ✓ edad
- ✓ correo
- ✓ sexo

Crear 5 estudiantes (Insert Many) con los campos mencionados arriba. Además, crear un estudiante sólo con nombre, apellido y curso. ¿Es posible?

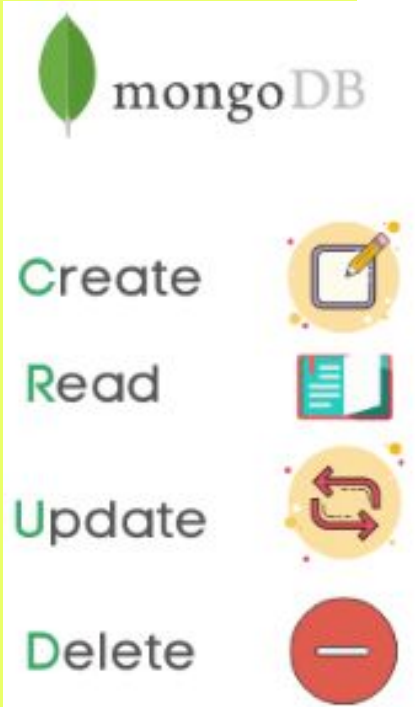
- ✓ Realizar una búsqueda para obtener a todos los estudiantes.
- ✓ Realizar una búsqueda para obtener a todos los estudiantes de sexo H (hombre)
- ✓ Realizar un conteo para obtener el número de documentos totales.
- ✓ Realizar un conteo para obtener el número de documentos totales que cumplan con el criterio: “Es mujer”

Ejemplo en vivo

Basado en nuestra base de datos “colegio”

- ✓ Se agregarán 5 estudiantes más, con diferentes campos y con la misma estructura. además, crear 1 alumno sólo con nombre.
- ✓ Realizar una búsqueda aplicando ordenamientos, proyecciones, saltos y límites.
- ✓ Se analizarán los resultados de las proyecciones, saltos, ordenamientos y límites. ¿Cómo se comportan los documentos que tienen campos incompletos?

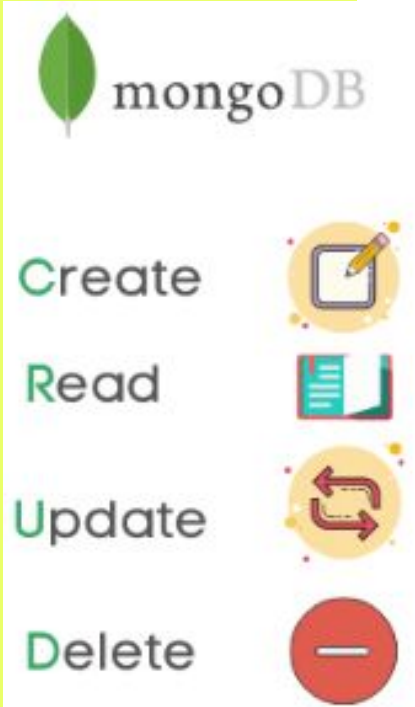
CRUD – UD
Update y Delete



CRUD : U (Update)

Las operaciones Update se pueden realizar de dos maneras: Actualizar un documento, o actualizar múltiples documentos.

- **db.collection.updateOne(query,update,option)**
  - query: sirve para filtrar qué elementos actualizar (usa los filtros iguales al find)
  - update: Apartado para indicar qué actualizar de los documentos que cumplen con el filtro. Update tiene sus propios operadores como $set, $unset, $inc, $rename, $mul, $min, $max
  - option: Opciones a tomar en cuenta para la actualización (como upsert, que inserta el valor en caso de que el documento a actualizar ni siquiera exista).



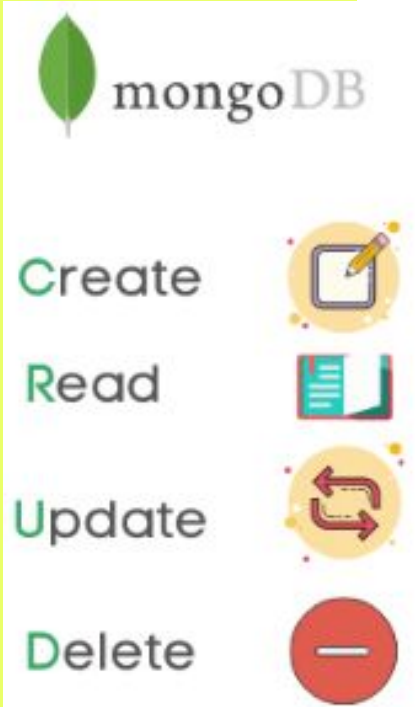
CRUD : U (Update)

- ✓ **db.collection.updateMany(query,update,options)**
  Actualiza múltiples documentos que cumplan con el criterio.

Lista de operadores para Update

Notamos cómo, antes de definir qué operación realizar, colocamos un filtro para saber qué documentos son de mi interés para actualización.

```
db.coll.update({"_id": 1}, {$set: {"year": 2016, name: "Max"}})
db.coll.update({"_id": 1}, {$unset: {"year": 1}})
db.coll.update({"_id": 1}, {$rename: {"year": "date"} })
db.coll.update({"_id": 1}, {$inc: {"year": 5}})
db.coll.update({"_id": 1}, {$mul: {price: NumberDecimal("1.25"), qty: 2}})
db.coll.update({"_id": 1}, {$min: {"imdb": 5}})
db.coll.update({"_id": 1}, {$max: {"imdb": 8}})
db.coll.update({"_id": 1}, {$currentDate: {"lastModified": true}})
db.coll.update({"_id": 1}, {$currentDate: {"lastModified": {$type: "timestamp"}}})
```



CRUD : D (Delete)

Nuestra última operación es para eliminar datos, si bien hay muchas variantes de una eliminación, sólo veremos las dos principales.

- ✓ **db.collection.deleteOne({key:val})** : Elimina sólo el primer elemento que cumpla con el criterio, se usa principalmente para encontrar identificadores específicos. Se recomienda no utilizar si somos conscientes de que el valor a buscar no es repetido.
- ✓ **db.collection.deleteMany({key:val}) :** Elimina todos los documentos que cumplan con el criterio, se usa cuando sabemos que más de un valor va a contar con ese valor y necesitamos hacer una limpieza general.

Operaciones con Filtros

Duración: **15-20min**

CODERHOUSE

ACTIVIDAD EN CLASE

Operaciones con Filtros

Sobre la base y los datos cargados anteriormente

1. Insertar cinco documentos más en la colección clientes con los siguientes datos:
   { "nombre" : "Pablo", "edad" : 25 }
   { "nombre" : "Juan", "edad" : 22 }
   { "nombre" : "Lucia", "edad" : 25 }
   { "nombre" : "Juan", "edad" : 29 }
   { "nombre" : "Fede", "edad" : 35 }
2. Listar todos los documentos de la colección clientes ordenados por edad descendente.
3. Listar el cliente más joven.
4. Listar el segundo cliente más joven.
5. Listar los clientes llamados 'Juan'
6. Listar los clientes llamados 'Juan' que tengan 29 años.
7. Listar los clientes llamados 'Juan' ó 'Lucia'.

ACTIVIDAD EN CLASE

Operaciones con Filtros

8. Listar los clientes que tengan más de 25 años.
9. Listar los clientes que tengan 25 años ó menos.
10. Listar los clientes que NO tengan 25 años.
11. Listar los clientes que estén entre los 26 y 35 años.
12. Actualizar la edad de Fede a 36 años, listando y verificando que no aparezca en el último listado.
13. Actualizar todas las edades de 25 años a 26 años, listando y verificando que aparezcan en el último listado.
14. Borrar los clientes que se llamen 'Juan' y listar verificando el resultado.
15. Eliminar además todos los documentos de estudiantes que hayan quedado con algún valor.

#FindTheBug

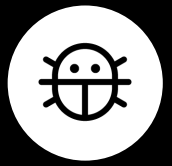
Encuentra los errores

A continuación se te presentará una lista de consultas y una descripción del resultado que se espera.

¿La consulta es correcta?
En caso contrario ¿Cuál es la sintaxis correcta?

Duración: **15 minutos**

CODERHOUSE



#FindTheBug

- ✓ Insertar múltiples mascotas:
  - ✓ db.pets.insertOne([{name:”aletas”,specie:”fish”},{name:”Doby”,{specie:”dog”}])

- ✓ Obtener sólo las últimas 5 mascotas que sean peces
  - ✓ db.pets.find({specie:”fish}).limit(5)

- ✓ Obtener sólo el nombre de las últimas 5 mascotas cuya edad sea menor de 10 años:
  - ✓ db.pets.find(age:{ $gte:{10}}},{name:1}).sort(age:1).limit(5)

¿Preguntas?

Muchas gracias.

#DemocratizandoLaEducación