**UNIDAD 1: MongoDB**

1. [UNIDAD 1: Introducción](https://learn.nextu.com/mod/page/view.php?id=10280&pid=P_WEB_DATABASE)

**Introducción**

**MongoDB**

MongoDB es un sistema manejador de base de datos NoSQL orientado a documentos, donde cada base de datos consiste de colecciones y las colecciones tienen documentos. En esta unidad comenzamos introduciendo las funciones para la creación de bases de datos MongoDB y las operaciones para el respaldo y recuperación de las bases de datos, así como también los mecanismos de importación y exportación de datos. Luego de ello, aprenderás a realizar consultas de mediana complejidad, con la finalidad de que explores el contenido de los documentos en las bases de datos. Uno de los conceptos más completos en MongoDB es el de agregación, que permite agrupar datos usando diversas opciones. De esta forma puedes generar desde vistas sencillas de tus datos hasta vistas más complejas resultantes del procesamiento de los datos usando las diversas funciones proporcionadas por el manejador. Finalmente, podrás definir usuarios y sus roles en tu base de datos a través de los mecanismos de autenticación soportados por MongoDB.

**Objetivos de aprendizaje**

1. Crear bases de datos NoSQL orientadas a documentos usando MongoDB que puedan ser respaldadas y recuperadas posteriormente.
2. Aplicar las consultas de agregación usando diversos operadores.
3. Implementar consultas de mediana complejidad sobre las bases de datos MongoDB con la finalidad de visualizar los datos de diversas formas.
4. Aplicar las funciones de autenticación de MongoDB para crear usuarios y sus correspondientes roles en el sistema, con la finalidad de validar los usuarios que pueden acceder las diferentes bases de datos.
5. Lección 1: Retomando MongoDB
   1. [Creando y consultando datos](https://learn.nextu.com/mod/lesson/view.php?id=10281&pid=P_WEB_DATABASE)

Comandos:

use database

show collections

db.collection.find()

db.collection.insert( 🡪 Se puede presionar tab para ver opciones

Ejemplo de insert:

db.tareas.insert ( [ { nombre: “Tarea2”},{ nombre: “Tarea3”}, detalle “dt” ])

Find con condición:

db.tareas.find({ detalle : “dt” })

db.tareas.find({ detalle : “dt”, nombre: “Tarea3” }) 🡪 es condición AND

Para hacer condición OR

db.tareas.find({ $or: [{ nombre: “Tarea2” },{ “Tarea3” }] })

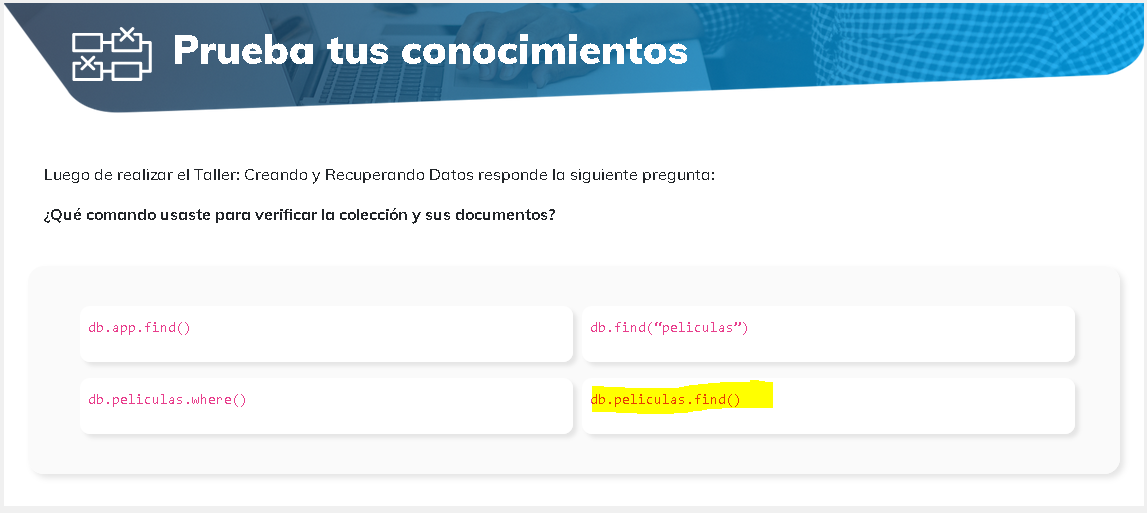
Otro ejemplo buscando documentos con valor fecha\_limite nulo:

db.tareas.find({ fecha\_limite: null })

Para eliminar documentos:

db.tareas.deleteMany ( { fecha\_limite: null } )

* 1. Taller 1
  2. Actividad Interactiva 1



* 1. Respaldando y recuperando datos

**Respaldo:**

mongodump (Respaldo todas las DBs)

mongodump -d productividad

Genera una carpeta que se llama dump 🡪 y se crea metadata y bson (binary son) 🡪 Datos en binario

**Recuperacion:**

mongorestore

**Importacion:**

Json archive

Con MongoDB Compass 🡪 click en collection 🡪 Import Data

Formato JSON o CSV

Y se selecciona el archivo

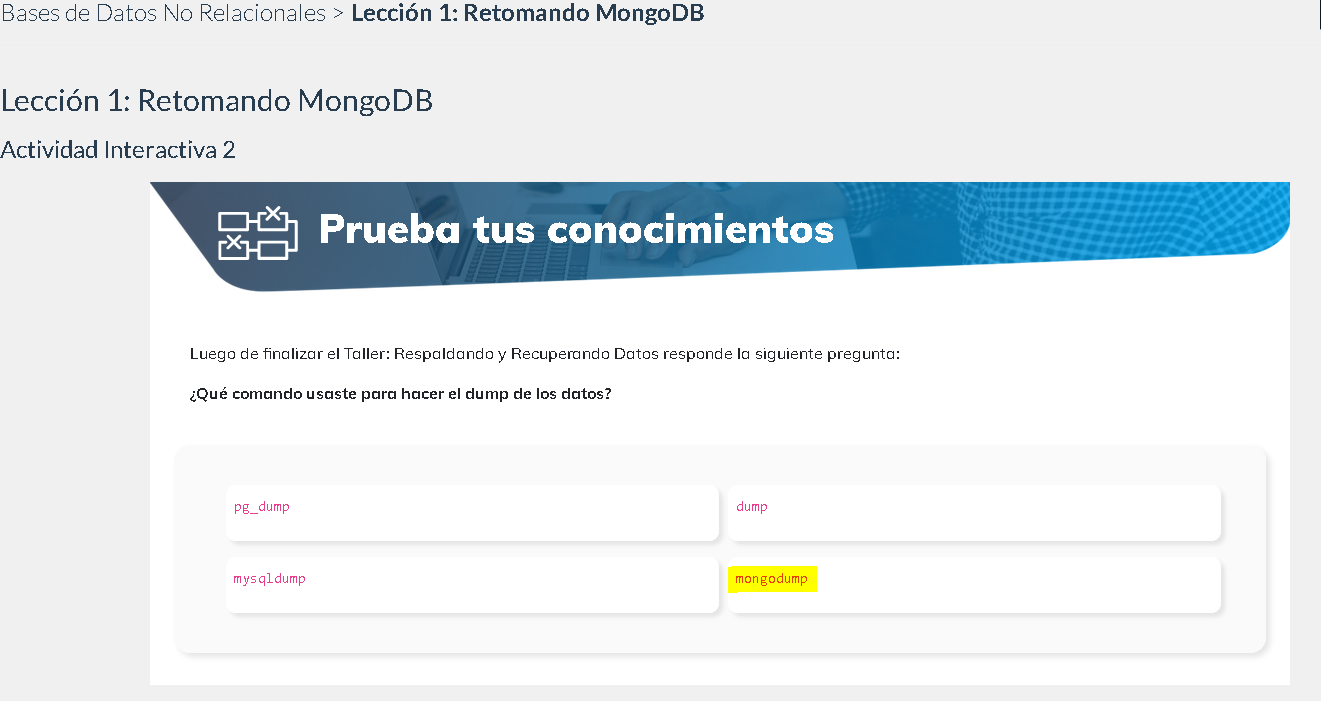
**Exportación:**

Con MongoDB Compass 🡪 click en collection 🡪 Export Data

Formato JSON o CSV

Seleccionar el nombre de archivo destino

* 1. Taller 2
  2. Actividad Interactiva 2



* 1. Ejercicio Práctico
  2. ¿Sabías qué?

**Cápsula de conocimiento**

**Resumen de comandos en MongoDB: Consultas Respaldo y Recuperación**

Este recurso te ofrece una guía rápida de los comandos en Mongo que hemos trabajado en esta lección.

* **Insertar un documento:**

> db.<coleccion>.insert({})

* **Consultas OR de dos condiciones:**

> db.<coleccion>.find({$or: [ , , ...]})

* **Consultar documentos sin un atributo:**

> db.<coleccion>.find({ : null })

* **Eliminar muchos:** Donde se eliminarán todos los documentos con el patrón indicado por {}.

> db.<coleccion>.deleteMany({})

* **Respaldo:** La operación de respaldo crea una carpeta llamada `dump` desde donde se ejecuta.

terminal> mongodump -d <base\_de\_datos>

* **Recuperación:** La operación de recuperación se realiza usando los archivos de la carpeta llamada `dump`.

terminal> mongorestore

Esperamos que este resumen sea de utilidad para ti en el futuro cercano. Recuerda que estas no son las únicas formas de utilizar estos comandos, existen otras, sin embargo, son las más comunes.

1. Lección 2: Consultas
   1. [Consultando y ordenando atributos](https://learn.nextu.com/mod/lesson/view.php?id=10282&pid=P_WEB_DATABASE)

Db.collection.find().pretty() 🡪 Una forma más ordenada

Db.collection.find().sort({ nombre: 1 }).pretty() 🡪 Ordernar, 1 ordernar de forma ascendente, -1 ordenar de forma descendente, 0 no ordenar

Db.collection.find().sort({ nombre: 1 }).limit(2).pretty() 🡪 Solo 2 documentos

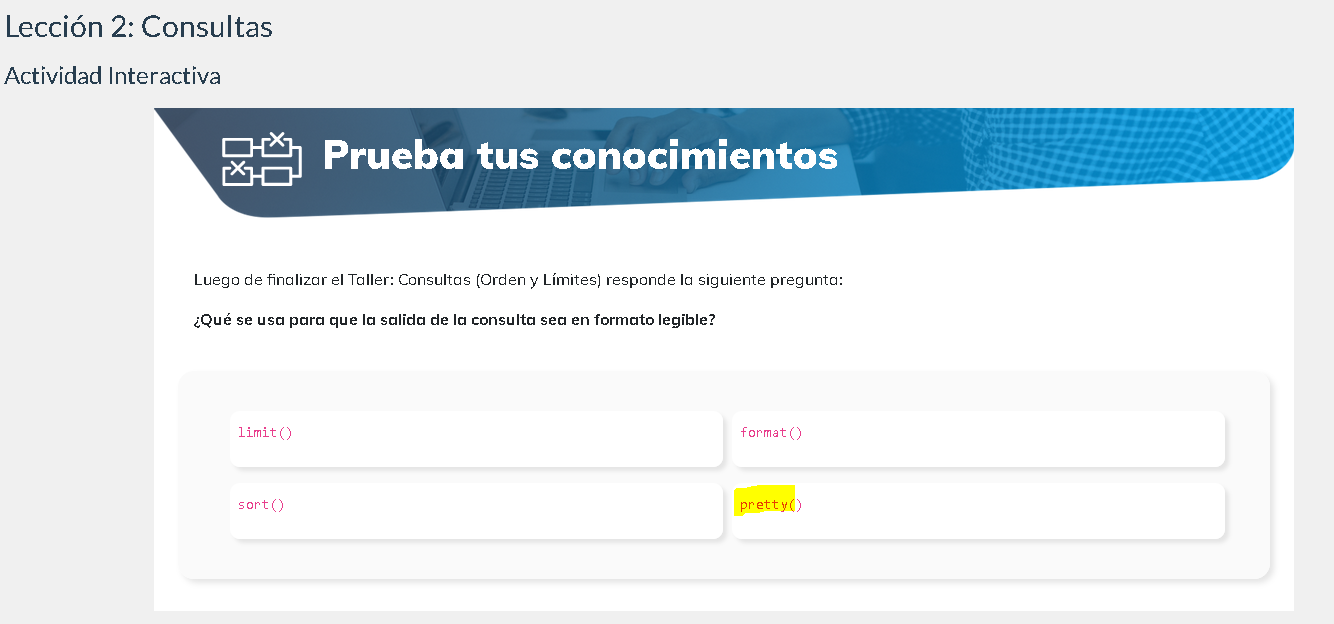
Db.collection.find( {}, { nombre: 1 } ) 🡪 1 significa que se muestre el campo, y 0 que no.

Db.collection.find( {}, { nombre: 1 , \_id: 0 } ) 🡪 Para desactivar el \_id

Db.collection.find({ nombre: “AlgunNombre” }) 🡪 Busca los documentos con nombre AlgunNombre

Db.collection.find({ nombre: { $ne: “AlgunNombre” } }) 🡪 Busca los documentos que no tenga nombre AlgunNombre

* 1. Taller 1
  2. Actividad Interactiva



* 1. Consultando textos

Db.collection.find({ nombre: { $regex: /ruiz/ }) 🡪 Para usar expresiones regulares

Es sensitivo a mayúsculas

Db.collection.find({ nombre: { $regex: /Ruiz/i }) 🡪 Para usar expresiones regulares no sensitivo a mayúsculas

Creando índices.

Db.collection.createIndex({ nombre : “text” })

Búsqueda de texto

Db.usuarios.find({ $text: {$search: “ruiz”} })

Db.usuarios.createIndex({ nombre: 1 }, { unique: true }) 🡪 se crea un índice para el nombre, y que se únique

* 1. Taller 2
  2. Ejercicio Práctico
  3. ¿Sabías qué?

**Cápsula de conocimiento**

**Resumen de Comandos en MongoDB: Consultas y Búsqueda**

A continuación, encontrarás un resumen de los comandos en MongoDB que hemos utilizado en esta lección.

* **Find con formato de salida mejorado:**

> db.<coleccion>.find().pretty()

* **Ordenando documentos:**

> db.<coleccion>.find().sort({atributo1: (1 ó 0 ó -1), ...})

* **Limit: Número máximo de resultados**

> db.<coleccion>.find().limit(<número>)

* **Solo mostrar algunos atributos:**

> db.<coleccion>.find({...},{atributo1: (1 ó 0), ...})

* **$ne (No igual a):**

> db.<coleccion>.find({$ne: “palabra”})

* **Expresión regular:**

> db.<coleccion>.find({atributo: {$regex: “expresion\_regular”}})

* **Crear índices:**

> db.<coleccion>.createIndex({atributo1: “text”, ...})

* **Búsqueda de texto con índices:**

> db.<coleccion>.find({$text: {$search: “palabra”}})

Con este material te hemos ofrecido una guía muy rápida de los comandos de MongoDB, que puede servirte para recordar la sintaxis de los mismos cuando estés usando tu base de datos Mongo.

1. Lección 3: Agrupando los datos
   1. [Agrupando datos usando Project](https://learn.nextu.com/mod/lesson/view.php?id=10283&pid=P_WEB_DATABASE)

Db.collection.count()

Db.collection.distinct(“atributo”)

Db.collection.aggregate ([ { $project: { descripcion: 1, \_id : 0, suma : { $add : [“atributo1” , “atributo2”, …]} } } ])

* 1. Taller 1
  2. Agrupando datos usando Group y Match

Db.collection.aggregate( [ { $group: { \_id : “atributo” } }] )

Db.collection.aggregate( [ { $group: { \_id : “atributo”, total : { $sum : “$atributo” } } }] )

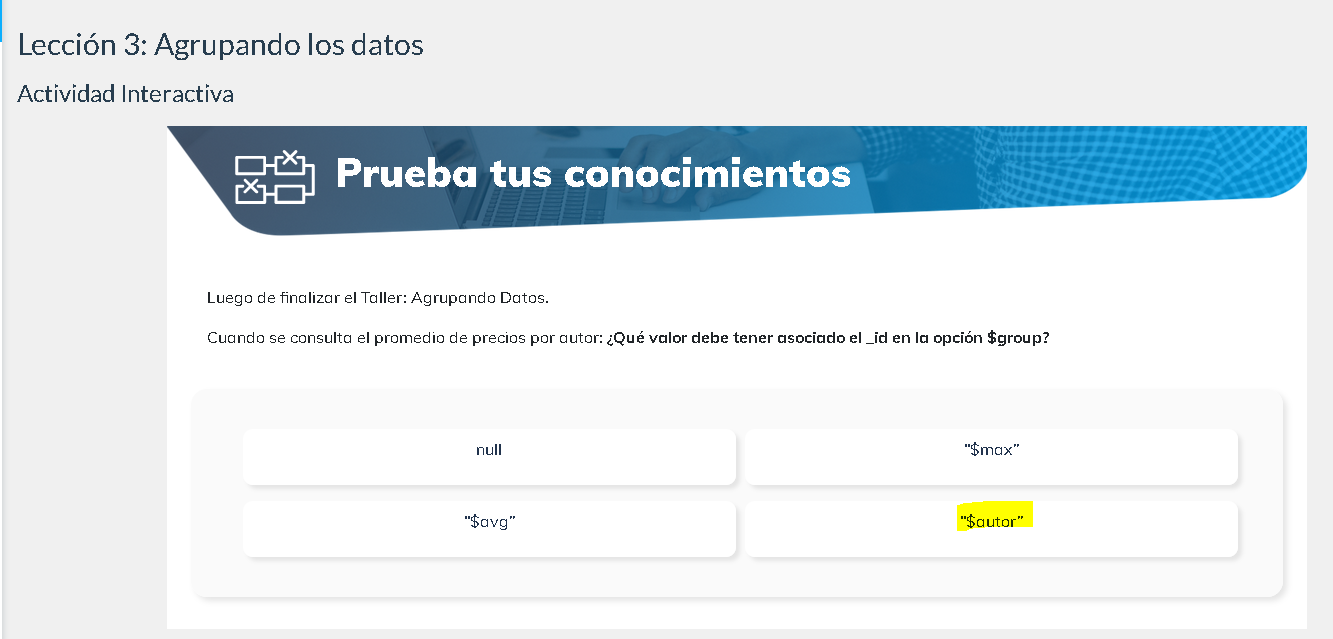
Db.collection.aggregate( [ { $group: { \_id : “atributo”, total : { $avg : “$atributo” } } }] )

Db.collection.aggregate( [ { $group: { \_id : “atributo”, total : { $sum : 1 } } }] )

Db.collection.aggregate( [ { $match : { atributo : valor} }, { $group: { \_id : “atributo”, total : { $sum : “$atributo” } } }] )

Db.collection.aggregate( [ { $match : { atributo : { $gt: 0 }} }, { $group: { \_id : “atributo”, total : { $sum : “$atributo” } } }] )

* 1. Taller 2
  2. Actividad Interactiva



* 1. Ejercicio práctico
  2. ¿Sabías qué?

**Agregación con Lookup**

¡Felicitaciones! Nos encanta el ritmo de trabajo e interés que has puesto en descubrir las bases de datos no relacionales. A continuación, vamos a explorar la agregación usando Lookup.

En las bases de datos relacionales es común hacer JOINS para unir datos que están en distintas tablas. En particular, en las bases de datos orientadas a documentos, concretamente en MongoDB, existe un mecanismo similar, el cual permite insertar un documento dentro de otro especificado en una consulta de agregación. Este mecanismo se llama `lookup` y a continuación se especifica su sintaxis.

{ $lookup: { from: <coleccion a insertar>, localField: <campo local que referencia, foreignField: <campo remoto>, as: <nombre de la salida> }}

Veamos un ejemplo. Tengamos las colecciones **productos** y **ventas** con los siguientes datos:

* **Productos**

{"\_id" : 1, "nombre" : "Camisa", "precio" : 5}   
{"\_id" : 2, "nombre" : "Corbata", "precio" : 2}

* **Ventas**

{"\_id" : 10, "articulo" : 1, "cantidad" : 10}   
{"\_id" : 11, "articulo" : 2, "cantidad" : 20}

Si se hace la siguiente consulta:

db.ventas.aggregate({  
$lookup:  
 {  
 from: "productos",  
 localField: "articulo",  
 foreignField : "\_id",  
 as: "producto"  
 }  
})

Se obtiene el siguiente resultado:

{  
 "\_id" : 10, "articulo" : 1, "cantidad" : 10,  
 "producto" : [ {"\_id" : 1, "nombre" : "Camisa" : 5}]}  
{  
 "\_id" : 11,"articulo" : 2, "cantidad" : 20,  
 "producto" : [ {"\_id" : 2, "nombre" : "Corbata","precio" : 2}]  
}

Aunque estas no son las únicas formas de utilizar estos comandos, son las más comunes. Esperamos que te sea de utilidad cuando estés trabajando con tu base de datos MongoDB.

* 1. ¿Sabías qué?

**Cápsula de conocimiento**

**Resumen de comandos en MongoDB: Consultas Agregadas**

Este recurso te ofrece una guía rápida de los comandos en MongoDB que hemos trabajado en esta lección.

* **Count - Contar documentos:**

> db.<coleccion>.count()

* **Distinct:**

> db.<coleccion>.distinct(“atributo”)

* **Project - Seleccionando atributos:**

> db.<coleccion>.aggregate([{$project: { atributo, nuevo\_atributo: {operaciones}}}])

* **Group - Agrupar campos:**

> db.<coleccion>.aggregate([{$group:{\_id: “$atributo”, atributo\_agregado: {funcion\_agregada}}}])

* **Match - Filtrado en agrupación:**

> db.<coleccion>.aggregate([{$match: {condicion}},{$group:{\_id: “$atributo”, atributo\_agregado: {funcion\_agregada}}}])

Aunque estas no son las únicas formas de utilizar estos comandos, son las más comunes. Esperamos que este resumen sea de utilidad cuando estés trabajando tu base de datos MongoDB.

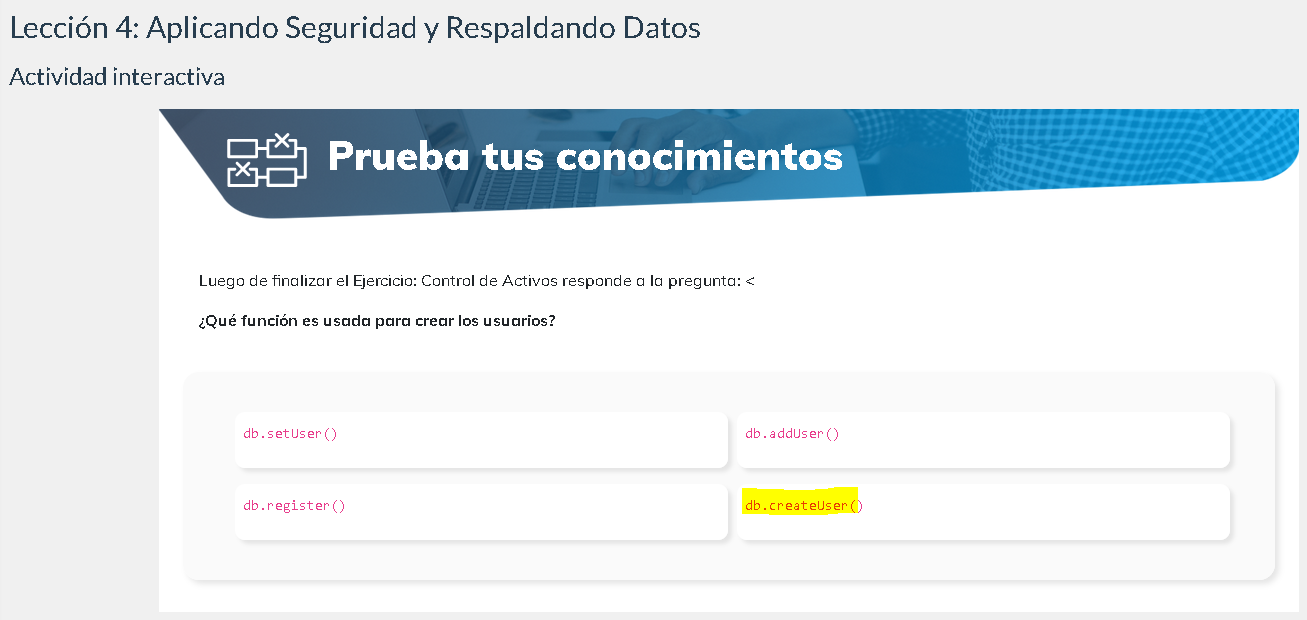
1. Lección 4: Aplicando Seguridad y Respaldando Datos
   1. [Autenticación en MongoDB](https://learn.nextu.com/mod/lesson/view.php?id=10284&pid=P_WEB_DATABASE)

db.getUsers();

db.createUser({user: "test1", pwd: "123123", roles: ["read"]});

db.createUser({user: "test1", pwd: "123123", roles: [{role: "read", db: "carros"}]})

* 1. Ejercicio práctico
  2. Actividad interactiva



* 1. ¿Sabías qué?

**Cápsula de conocimiento**

**Resumen de comandos en MongoDB: Autenticación y Permisos**

A continuación, te presentamos una lista con los comandos de MongoDB relacionados a la autenticación.

* **Entrar con un usuario:**

> mongo -u <usuario> -p <contraseña> <base\_de\_datos>

* **Listar usuarios:**

> db.getUsers();

SQL

* **Crear usuario:**

> db.createUser({user: “<usuario>”, pwd: “contraseña”, roles: [ <rol\_1>, <rol\_2>]});

Algunos roles predefinidos:

* + root
  + read
  + readWrite
  + dbAdmin
  + dbOwner
  + backup
  + restore
  + readAnyDatabase
  + readWriteAnyDatabase
  + dbAdminAnyDatabase
* **Ver información de un usuario:**

> db.getUser(<usuario>)

Esperamos que esta ayuda sea de utilidad cuando estés autenticando tus datos en MongoDB.

* 1. ¿Sabías qué?

**Cápsula de conocimiento**

**Equivalencias entre MongoDB y SQL**

El objetivo de este recurso es proporcionarte una comparación rápida entre los conceptos de base de datos en MongoDB y SQL.

 **Conceptos:**

| **SQL** | **MongoDB** |
| --- | --- |
| base de datos | base de datos |
| tabla | colección |
| fila | documento |
| índice | índice |

 **Consultas:**

| **SQL** | **MongoDB** |
| --- | --- |
| SELECT | db.coleccion.find() |
| SELECT … WHERE condiciones | db.coleccion.find({condiciones}) |
| SELECT COUNT() | db.coleccion.count() |
| JOINS | documentos embebidos |
| GROUP BY | agregación |

 **Actualización:**

| **SQL** | **MongoDB** |
| --- | --- |
| INSERT INTO tabla... | db.coleccion.insert({}) |
| UPDATE | db.coleccion.updateMany({}) |
| DELETE FROM | db.coleccion.deleteMany({}) |
| CREATE INDEX | db.coleccion.createIndex |
| DROP TABLE | db.coleccion.drop() |

Como pudiste observar, aunque MongoDB sigue un paradigma distinto al SQL entre estos existen sus equivalencias.

1. UNIDAD 1: Prueba

|  |  |
| --- | --- |
| **Comenzado en** | Monday, 24 de May de 2021, 17:11 |
| **Estado** | Terminados |
| **Finalizado en** | Monday, 24 de May de 2021, 17:19 |
| **Tiempo empleado** | 8 mins 49 segundos |
| **Puntos** | 10/10 |
| **Calificación** | **100** de un total de 100 |

Top of Form

### Pregunta 1

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

El usuario **test1** está tratando de insertar un documento en la colección **tabletas** de las base de datos **productos** de un proveedor de dispositivos móviles. Y obtiene un mensaje que dice “no autorizado en productos para ejecutar el comando insert en tabletas”. ¿Qué puede estar sucediendo?

Seleccione una:

a. El usuario test1 tiene los mismos roles que el usuario root

b. El usuario test1 olvido colocar un parámetro cuando realiza el insert

c. El usuario test1 no tiene permiso para insertar en tabletas en la base de datos productos

d. La autorización no está habilitada en el sistema

#### Retroalimentación

**La única opción posible es que el usuario test 1 no tenga permiso para insertar en la base de datos productos de lo contrario no hubiese obtenido el error que se desplegó.**

La respuesta correcta es: El usuario test1 no tiene permiso para insertar en tabletas en la base de datos productos

### Pregunta 2

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Supongamos que tenemos la base de datos Mongo **deportes** que incluye la colección **artes\_marciales** y los documentos {“nombre”: “judo”, “origen” : japones};  {“nombre”: “karate”, “origen” : japones};  {“nombre”: “KungFu”, “origen” : chino}; .  ¿Cuál sería el comando en mongo para insertar el arte marcial Taekwondo de origen koreano y el arte marcial jutjitsu de origen japonés usando una sola instrucción? Asuma que ya estamos usando la base de datos **deportes**.

Seleccione una:

a. db.artes\_marciales.insert ( [ {nombre: "jujitsu", origen: “japones” }, {nombre: "taekwondo", origen: “koreano” } ] )

b. db.artes\_marciales.insert (  [ {“nombre”: "kungfu", “chino” }, {“nombre”: "taekwondo", “koreano” } ] )

c. db.lote.insert (  [ {nombre: "kungfu", origen: “chino” }], [{nombre: "taekwondo", origen: “koreano” } ] )

d. db.artes\_marciales.insert ([ {nombre: "kungfu", origen: “chino” }]);

#### Retroalimentación

El comando en Mongo para insertar múltiples documentos es: db.coleccion.insert([{},{}])

La respuesta correcta es: db.artes\_marciales.insert ( [ {nombre: "jujitsu", origen: “japones” }, {nombre: "taekwondo", origen: “koreano” } ] )

### Pregunta 3

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Relacione los términos de MongoDB en la parte derecha con los conceptos de SQL en la parte izquierda.

|  |  |
| --- | --- |
| tabla | Respuesta 1 |

- Bottom of Form

|  |
| --- |
|  |
| SELECT | Respuesta 2 |

|  |
| --- |
|  |
| CREATE INDEX | Respuesta 3 |

|  |
| --- |
|  |
| GROUP BY | Respuesta 4 |

|  |
| --- |
|  |

#### Retroalimentación

La respuesta correcta es: tabla → coleccion, SELECT → db.coleccion.find(), CREATE INDEX → db.coleccion.createIndex, GROUP BY → aggregate

### Pregunta 4

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Haga coincidir las definiciones en la columna de la derecha con los conceptos en la parte izquierda.

|  |  |  |  |
| --- | --- | --- | --- |
| mongorestore | | Respuesta 1 | |
|  |
| csv | Respuesta 2 | |

|  |
| --- |
|  |
| mongodump | Respuesta 3 |

|  |
| --- |
|  |

#### Retroalimentación

La respuesta correcta es: mongorestore → se usa para recuperar las bases de datos del sistema, csv → es un formato que se puede usar para importar datos usando Compass, mongodump → se usa para respaldar todas las bases de datos del sistema

### Pregunta 5

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

¿Cuál de las siguientes funciones en MongoDB desplegará  los nombres de los deportes extremos en los documentos de la colección extremos y sus correspondientes ids en la base de datos deportes?

Seleccione una:

a. db.extremos.find ({}, {nombre: 1})

b. db.extremos.find ({}, {nombre: 0})

c. db.extremos.find ({}, {nombre: -1})

d. db.extremos.find ({}, {nombre:  “Paracaidismo”})

#### Retroalimentación

El comando **> db.<coleccion>.find({...},{atributo1: (1 ó 0), ...})**  
permite mostrar algunos atributos de los documentos en una colección. El atributo 1 indica que el atributo estará presente y el 0 que no.

La respuesta correcta es: db.extremos.find ({}, {nombre: 1})

### Pregunta 6

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

¿Cuál es la función de Mongo que debe usarse para indicar que solo deseamos ver el atributo **origen** de los documentos de la colección **artes\_marciales** de la base de datos **deportes** en MongoDB en una consulta agregada?

Seleccione una:

a. db.artes\_marciales.aggregate ( [ { $project:  { origen: 1 } } ] )

b. db.artes\_marciales.aggregate ( [ { $project:  { origen: 0 } } ] )

c. db.artes\_marciales.aggregate ( [ { $match:  { origen: 1 } } ] )

d. db.artes\_marciales.aggregate ( [ { $match:  { origen: 0 } } ] )

#### Retroalimentación

**La fase $project del método aggregate permite incluir o excluir atributos de los documentos en una colección. La opción 1 indica que el atributo será incluido y la opción 0 que no.**

La respuesta correcta es: db.artes\_marciales.aggregate ( [ { $project:  { origen: 1 } } ] )

### Pregunta 7

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ es el método usado para agregar (agrupar) valores de los datos en una vista particular.

Seleccione una:

a. db.coleccion.aggregate()

b. db.coleccion.find()

c. db.coleccion.Insert()

d. db.coleccion.createIndex()

#### Retroalimentación

**El aggregate permite unir o agregar datos de una colección en un vista simple o incluso en una colección de acuerdo a los que se denominan etapas tales como $match y $project**

La respuesta correcta es: db.coleccion.aggregate()

### Pregunta 8

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ se usa para filtrar documentos en MongoDB en agregaciones.

Seleccione una:

a. $match

b. $sort

c. $project

d. $group

#### Retroalimentación

El $match permite filtrar documentos de acuerdo a una determinada condición

La respuesta correcta es: $match

### Pregunta 9

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

¿Cuáles de las siguientes son  características de las búsquedas basada en expresiones regulares en MongoDB?.  Señala todas las que apliquen.

Seleccione una o más de una:

a. Busca por cualquier expresión regular que se indique en la función find.

b. Requiere que se defina un índice para el atributo que se utiliza en la búsqueda

c. Se debe utilizar el atributo $regex cuando se define la condición de la búsqueda

d. Son sensitivas a las mayúsculas al menos que se especifique lo contrario

#### Retroalimentación

**Cuando se realizan búsquedas por expresiones regulares se usa la función find acompañada por los atributos sobre las cuales se desea realizar la búsqueda; cada atributo debe ir acompañado por una condición de la búsqueda que se indica a través del atributo $regex. Las búsquedas son sensitivas a las mayúscula al menos que se use la opción i.**

Las respuestas correctas son: Busca por cualquier expresión regular que se indique en la función find., Se debe utilizar el atributo $regex cuando se define la condición de la búsqueda, Son sensitivas a las mayúsculas al menos que se especifique lo contrario

### Pregunta 10

Correcta

Puntúa 1 sobre 1

Señalar con bandera la pregunta

#### Texto de la pregunta

Supongamos que tenemos la base de datos Mongo **deportes** que incluye la colección **artes\_marciales** y los documentos {“nombre”: “judo”, “origen” : japones};  {“nombre”: “karate”, “origen” : “japones”};  {“nombre”: “KungFu”, “origen” : “chino”}; .  ¿Cuál sería el comando en mongo para realizar una búsqueda que retorne las artes marciales de origen japonés? Asuma que ya estamos usando la base de datos **deportes**.

Seleccione una:

a. db.artes\_marciales.find ({origen: “japones”)

b. db.deportes.artes\_marciales.find ({origen: “japones”)

c. db.artes\_marciales.find ({nombre: “japones”)

d. db.japones. artes\_marciales.find ({origen: “japones”)

#### Retroalimentación

El comando en Mongo para realizar búsquedas simples por un atributo es: db.<coleccion>.find({<patron1>})

La respuesta correcta es: db.artes\_marciales.find ({origen: “japones”)