DIGITALHOUSE

Revisión de la semana 6

Certified Tech Developer

The Ultimate Degree

Índice

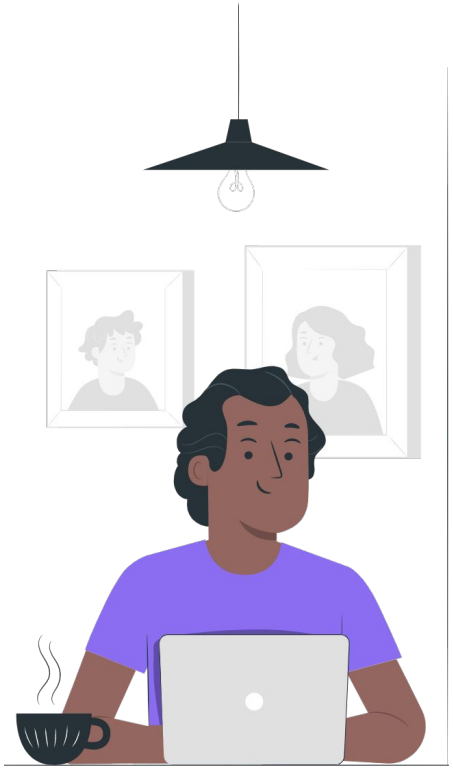
01

Recapitulando clase 16: Relacionales vs. no relacionales

Recapitulando: Relacionales vs. no relacionales

En la clase 16, vimos que las bases de datos no relacionales o NoSQL están diseñadas para modelos de datos específicos. También aprendimos que tienen esquemas flexibles para satisfacer las necesidades de las aplicaciones modernas que utilizan y generan tipos de datos complejos y en evolución.

Es una lectura totalmente diferente del concepto de bases de datos relacionales. Se creó para trabajar con grandes volúmenes de datos. Por lo tanto, utiliza JSON para la codificación de datos.



SQL vs. NoSQL

Las principales diferencias entre SQL y NoSQL:

Características	SQL	NoSQL
Relacional	Sí	No
Esquema de datos	Rígida	Flexíble
Escalabilidad	Vertical	Horizontal
Lenguaje de consulta	SQL	No hay norma universal
ACID	Sí	No

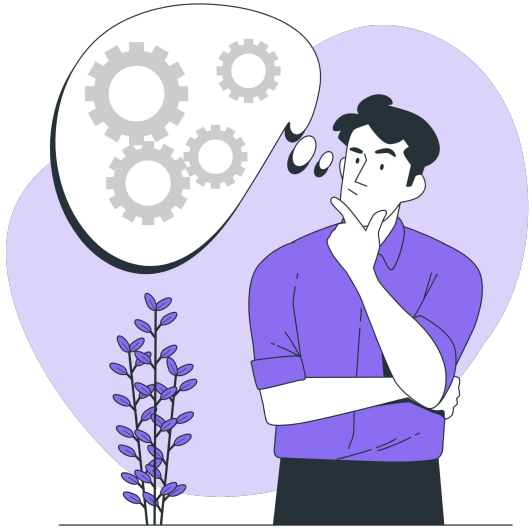
02

Recapitulando clase 17: MongoDB

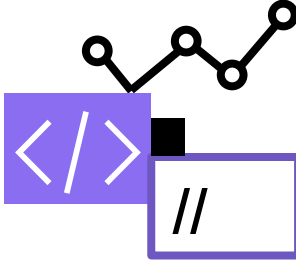
Recapitulando: MongoDB

Repasemos a continuación lo que vimos en la clase 17:

- Instalación de MongoDB
- Técnicas de modelado
- Creación de colecciones y documentos
- Borrar, limitar y ordenar las consultas
- Presentación de la herramienta Compass, que es un editor visual para MongoDB



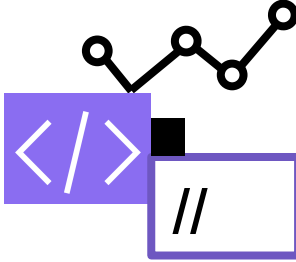
03 Revisión de los comandos de MongoDB



Comando: db

El comando db muestra el nombre de la base de datos que se está utilizando actualmente.

```
> db
mibanco
```

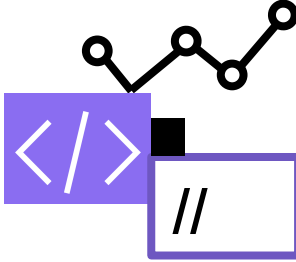


Comando: use

Para crear una base de datos o modificar una existente, utilizamos el comando **use**.

```
> use clase
switched to db clase
```

El retorno "switched to db clase" nos indica que ahora estamos utilizando la base de datos aula.

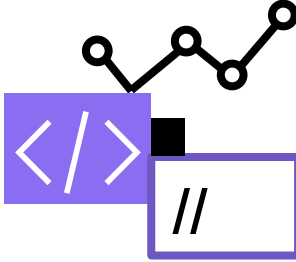


Comando: db.dropDatabase( )

El comando **db.dropDatabase( )** excluye una base de dados.

Si no fue seleccionado ningún banco, la base de datos **test** será excluida.

```
clase> db.dropDatabase()
{ ok: 1, dropped: 'clase' }
clase>
```



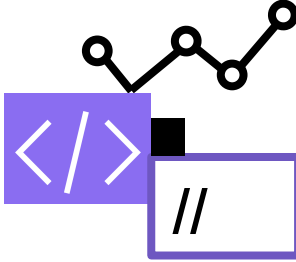
Comando: db.createCollections

El comando **db.createCollections** crea una colección.

MongoDB es bastante flexible, por lo que si no utilizamos el comando, mongoDB crea la colección por nosotros cuando insertamos un documento.

```
{ }
clase> db.createCollection("curso")
{ ok: 1 }
```

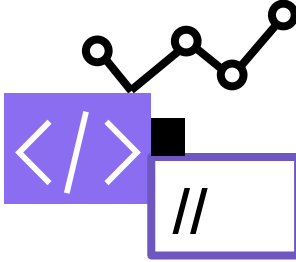
```
{ }
clase> db.curso.insertOne({nombre: "MongoDB"})
{
  acknowledged: true,
  insertedId: ObjectId("62253b5dfd59f32cd09dd8ad")
}
```



Comando: insert

El comando **insertOne( )** inserta un documento en una colección.

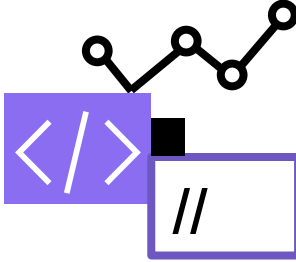
```
clase> db.curso.insertOne({_id: 002, titulo: "Curso MongoDB",
descripción: "El curso de MongoDB tiene como prerrequisito el MySQL",
tags: ['mongodb', 'database', 'noSQL']})
{ acknowledged: true, insertedId: 2 } // retorno do insert
```



Comando: find( )

El comando **find()** consulta los datos en las colecciones de MongoDB. Para mostrar los resultados de forma formateada, podemos utilizar la función **pretty( )**.

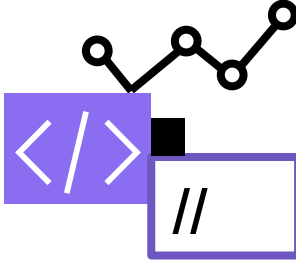
```
clase> db.curso.find().pretty()
[
  { _id: ObjectId("62253b5dfd59f32cd09dd8ad"), nombre: 'MongoDB' },
  {
    _id: 2,
    título: 'Curso MongoDB',
    descripción: 'El curso de MongoDB tiene como prerrequisito el MySQL',
    tags: [ 'mongodb', 'database', 'noSQL' ]
  }
]
```



Comando: find( )

El find() también nos permite consultar un registro específico. Por ejemplo, si queremos mostrar un registro cuyo título sea “Curso MongoDB’, debemos hacer lo siguiente:

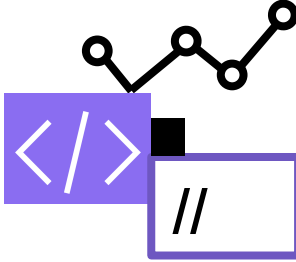
```
clase> db.curso.find({título: “Curso MongoDB”})
[
  {
    _id: 2,
    título: 'Curso MongoDB',
    descripción: 'El curso de MongoDB tiene como prerrequisito el MySQL',
    tags: [ 'mongodb', 'database', 'noSQL' ]
  }
]
```



Comando: limit( )

El comando **limit()** limita los registros en MongoDB. Este método establece el número de documentos que desea que se muestren.

```
clase> db.curso.find().limit(1)
[ { _id: ObjectId("62253b5dfd59f32cd09dd8ad"), nombre: 'MongoDB' } ]
```



Comando: sort( )

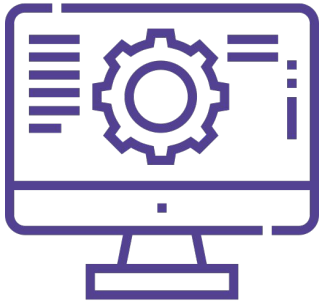
El comando **sort ( )** ordena documentos.
Este método establece un orden ascendente cuando el parámetro es 1, mientras que -1 se utiliza para establecer un orden descendente.

```
clase> db.curso.find({}).sort({título: -1})
[
  {
    _id: 2,
    título: 'Curso MongoDB',
    descripción: 'El curso de MongoDB tiene como prerrequisito el
MySQL',
    tags: [ 'mongodb', 'database', 'noSQL' ]
  },
  { _id: ObjectId("62253b5dfd59f32cd09dd8ad"), nombre: 'MongoDB' }
```

Base de datos test

Por defecto, MongoDB crea la base de datos "test".

Supongamos que introducimos un documento sin especificar la base de datos. Este documento se almacenará automáticamente en la base de datos "de prueba".

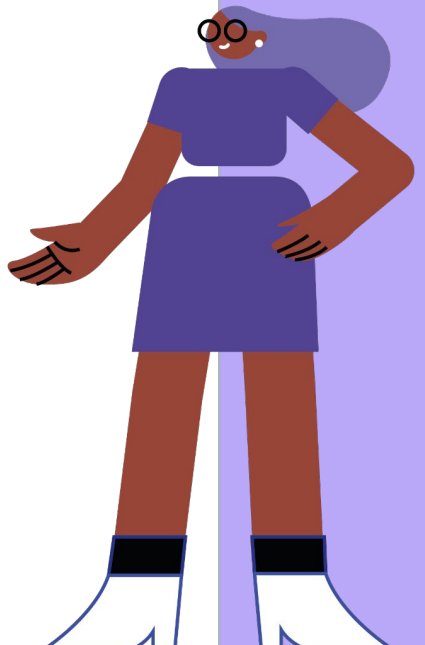


Conclusiones

Es muy importante que practiquemos estos comandos. Cuanto más ejercitemos, más fácil será crear y manipular nuevos documentos.

Ahora depende de ustedes. Practiquen, busquen nuevos comandos e intenten ser creativos.

¿Vamos?



DIGITALHOUSE

¡Muchas gracias!