HERRAMIENTAS PARA CARGA MASIVA DE DATOS EN MOONGODB

**Mongoose:**

Herramienta creada con JavaScript para realizar operaciones con una base de datos MongoDB. En una analogía muy burda, es una conexión a una base de datos, muy al estilo de JDBC.

**Su funcionamiento:**

Mongoose realiza una conexión a través del protocolo MongoDb Wire protocol usando un controlador (driver) proporcionado por Node.js. Una vez realizada la conexión, se crea un Schema que es la representación de una colección de MongoDB en nuestro código, tipo ORM. A estas representaciones de colecciones se le pueden agregar métodos para realizar tratado de la información antes de ejecutar operaciones con la base de datos. Todo lo anterior se compila en un modelo que representa los métodos y el esquema que utiliza Mongoose.

**Ejemplo:**

1. Se Instala Mongoose con el comando npm install moongoose.
2. **Conexión a base de datos**

Para realizar la conexión a la base de datos necesitaremos dos cosas: primero es importar la biblioteca de Mongoose y después de eso crear la URI de conexión con MongoD.

**Line de comandos en VISUAL STUDIO CODE.**

const mongoose=require('mongoose');

mongoose.connect('mongodb://localhost:27017/Usuarios',

{

useNewUrlParser:true,

useUnifiedTopology: true,

useFindAndModify: false,

useCreateIndex: true

});

El primer parámetro es la Uri de conexión hacia MongoDb, el segundo es una serie de opciones para crear la conexión, de todas las opciones la única indispensable es useNewUrlParser:true, ya que si no se especifica esta opción Mongoose utilizará un método obsoleto para intentar parsear la URI de conexión.

1. **Definición del Schema**

Como menciono anteriormente, el Schema es una representación de una colección que existirá en MongoDB. La definición en código es parecida a esto:

const Schema=mongoose.Schema;

const ObjectId=Schema.ObjectId;

const UsuarioSchema=new Schema({

id: ObjectId,

nombreUsuario: String,

apellidoPaterno:String,

balance:Number

});

Las primeras dos líneas son constantes que representan partes de la conexión, una es el esquema Schema y la otra es una representación del \_id autogenerado de MongoDB cuyo nombre es ObjectID. El código restante es simplemente un objeto que representa la colección de datos que crearemos/manipularemos.

Schema y la otra es una representación del \_id autogenerado de MongoDB cuyo nombre es ObjectID. El código restante es simplemente un objeto que representa la colección de datos que crearemos/manipularemos.

1. **Definición de Modelo**

Un modelo es una representación del Schema que puede ser instanciado, es decir ser un objeto. Este se utiliza para hacer operaciones como save(). Se define en el código de la siguiente manera:

var Usuario= mongoose.model('usuarios',UsuarioSchema);

El primer parámetro es la colección que representa y el segundo es el esquema por utilizar.

1. Definición de métodos.

A los esquemas que creamos le podemos agregar dos tipos de métodos, method y static, los primeros son para que se creen métodos de ejecución en instancias del modelo, mientras los segundos son métodos que se crean para el Schema. Lo único que varia es la palabra clave al final:

//estatico

UsuarioSchema.statics.buscarPorId=function(id){

return Usuario.findById(id);

}

//en instancia

UsuarioSchema.methods.buscarPorId=function(id){

return Usuario.findById(id);

}

Operaciones básicas

Mongoose cuenta con unas operaciones básicas predefinidas llamadas Queries, que precisamente son para hacer consultas en la base de datos, puedes encontrar las operaciones predefinidas aquí. Mongoose ejecuta los Queries de dos maneras: al hacer llamado a una función callback, es decir una función que se ejecute al hacer el llamado del Query o consulta y por medio de la palabra clave then. Aunque la documentación oficial lo especifica, yo también lo hare, las consultas o Queries de Mongoose no son promesas, son métodos asíncronos. El ejemplo de un Query es el siguiente:

//con then

Usuario.findById(id).then(()=>{//operaciones});

//con callback

Usuario.findById(id,function(result,error){

//operaciones de la funcion

});

Cabe destacar que Mongoose hace un soporte completo a todas las operaciones disponibles para MongoDB, es decir, puedes hacer un a consulta con un where, por ejemplo de la siguiente manera:

Usuario.update({nombreUsuario:nombreUsuario,apellidoPaterno:apellidoPaterno,balance:balance}).where({\_id:id});

Esta es una consulta que obtiene un registro por un id.