

Práctica 3

Se pide desarrollar un API REST utilizando Node.js, Express y MongoDB. La temática será libre e implementará al menos los siguientes endpoints:

- A. Obtener todos los documentos de la colección.
- B. Obtener un documento dado su id.
- C. Obtener todos los documentos que cumplan dos condiciones.
- D. Insertar uno o varios documentos.
- E. Actualizar un documento dada una condición.
- F. Eliminar un documento dado su id

Comentario del alumno

Este documento muestra los pasos que he seguido para crear y gestionar una base de datos MongoDB.

Me hubiera gustado, en un documento adjunto, explicar cómo se crean y consumen los endpoints de una API creado con el entorno 'EXPRESS', pero no es el caso.

Ya me parece bastante útil aprender a manejar las bases de datos 'MongoDB', la aplicación de escritorio 'MongoDBCompass' y el framework de 'Mongoose'.

Es una pena que no se dedique más tiempo a estudiar MongoDB en la escuela, pero entiendo que también forma parte de las tareas del alumno, realizar investigaciones por cuenta propia y formarse una idea de las nuevas tecnologías que surgen cada año y que vienen a mejorar las ya existentes.

Primeros pasos

La práctica consiste en gestionar 3 entornos diferentes, que finalmente realizan la tarea conjunta de administrar una base de datos en MongoDB. Ya ha quedado explicado qué es MongoDB. Una base de datos no relacional, formada a base de colecciones de documentos con formato JSON y con un esquema definido por el administrador, que finalmente contiene las llaves para manejar toda la información.

Personalmente, pienso que las bases de datos no relacionales presentan facilidades evidentes frente a las relacionales. Además de esto, el analista inteligente que conoce perfectamente su base de datos, es capaz de extraer ventajas superiores al manejo de tablas con identificadores únicos.

En primer lugar MongoDB requiere de un usuario registrado, que suscriba su cuenta gratuita o de pago. Esto se hace a través de su página WEB.

En segundo lugar, aunque no es obligatorio, me ha resultado muy útil contar con la aplicación de escritorio MongoDB Compass, para visualizar las colecciones de manera interactiva gracias a su interfaz gráfica.

Estas aplicaciones son fundamentales en la transición desde un entorno gráfico, hacia la línea de comandos o el propio código de `mongoose`.

Creación de la base de datos

‘Mongo ATLAS’, creamos un cluster que sirve de almacenamiento de nuestras bases de datos. El motivo del cluster es doble, por un lado define la capacidad máxima de la que disponemos, y por otro lado supone nuestro plan de suscripción a la plataforma online. Mejores planes implican mejores condiciones de cluster.

Ahora ya podemos conectar con MongoATLAS desde nuestro programa node.js. Simplemente necesitamos una línea de código.

Si no se especifica una base de datos, seremos automáticamente conectados a una base de datos ‘test’.

Para conectarnos a nuestra base de datos dentro del cluster, añadimos la ruta en la línea de conexión.

Añadiendo documentos a nuestra base de datos

El funcionamiento para añadir documentos a nuestra base de datos es increíblemente sencillo.

Primero creamos el esquema que da formato a un documento. Es importante darse cuenta que podemos tener tantos documentos como deseemos y así, tantos esquemas como necesitemos. Nuestra base de datos se libera del esquema rígido del modelo relacional

A continuación creamos el documento propiamente dicho. Debemos darle parametros de nombre de base de datos y esquema del modelo.

Se crea un nuevo registro, atendiendo al esquema.

Como buscar documentos dentro de nuestra base de datos

Es muy sencillo. Primero definimos el modelo del documento que queremos encontrar. Entonces guardamos en una variable el modelo de respuesta. Para terminar ejecutamos el comando 'find' con el patrón de búsqueda '{}' que implica buscar cualquier documento.

Realizando tareas de mantenimiento de nuestra coleccion

A continuación vamos a realizar tareas sencillas con nuestra coleccion de documentos.

Podemos empezar buscando documentos según diferentes parametros. En este caso, estamos buscando un documento por el `_id` que usa de manera interna MongoDB

Tambien podemos encontrar un documento y actualizarlo con nuevos parametros

Y por último, también podemos encontrar por el id y luego borrar el documento.