

**Memoria Trabajo Práctico 2 (TP2)**

Bases de Datos Documentales: MongoDB

***Alumno****: Francisco Javier Piqueras Martínez*

***Asignatura****: Gestión y Almacenamiento de Información no Estructurada*

***Fecha******de******entrega****: 20 de abril de 2020*

Índice

[1. Descripción del documento 4](#_Toc38105356)

[2. Ejercicios 4](#_Toc38105357)

[2.1. Ejercicio 1: Caso práctico Blog de Noticias. 4](#_Toc38105358)

[2.1.1. Diseño de la base de datos documental. 5](#_Toc38105359)

[2.1.2. Definición de índices 6](#_Toc38105360)

[2.1.3. Inserciones 6](#_Toc38105361)

[2.2. Ejercicio 2 7](#_Toc38105362)

Ilustraciones

[Ilustración 2‑1. Diagrama diseño Base de Datos. Blog de noticias. 5](#_Toc37801163)

# Descripción del documento

Este documento consiste en la memoria del Trabajo Práctico 2, de la asignatura de “Gestión y Almacenamiento de Información no Estructurada” del “Máster en Ingeniería y Ciencia de Datos” de la UNED.

El trabajo consiste en la realización de tres ejercicios con el objetivo de familiarizarse con todos los aspectos de la utilización de bases de datos documentales, y en concreto de la tecnología MongoDB.

En el primer ejercicio, se explorará el diseño de una base de datos utilizando dicha tecnología. En el segundo ejercicio se proporcionarán los datos necesarios para poblar una base de datos MongoDB ya definida y se solicitará al estudiante la implementación de diversas consultas que permitan explorar las diferentes funcionalidades que proporciona la tecnología. Finalmente, en el tercer ejercicio se explorarán las características de MongoDB en lo relativo al clustering y la distribución de los datos, para lo cual el estudiante deberá́ seguir un tutorial propuesto para la creación de una estructura de clúster para la base de datos y la exploración de sus características.

# Ejercicios

## Ejercicio 1: Caso práctico Blog de Noticias.

En el primer ejercicio se propone un caso práctico concreto para el cual se solicita el diseño más adecuado para una base de datos documental:

* Se precisa diseñar un blog de noticias donde los usuarios registrados puedan publicar sus comentarios.
  + Cada autor tiene un nombre, un nombre de usuario, una cuenta de Twitter y una descripción. Además, de forma opcional, los usuarios pueden proporcionar como datos su dirección postal (pueden tener una como máximo, con calle, número, puerta, C.P y ciudad) o sus teléfonos de contacto (pueden tener varios).
  + Las noticias tienen un título, un cuerpo y una fecha de publicación. Son publicadas por un solo autor y pueden contener o no una lista de tags.
  + Las noticias reciben comentarios (de 0 a n), quedando registrado la persona que lo escribió, el comentario escrito y el momento en el que lo hizo.
* Las consultas más frecuentes en la base de datos son las siguientes:
  + Consultas por nombre de usuario.
  + Consultas por cuenta de Twitter.
  + Agregaciones por código postal (número de usuarios que tienen el mismo C.P.)
  + Consultas de noticias de un usuario, ordenadas por fecha (las "n" últimas noticias publicadas, de la más reciente a la más antigua).
  + Número de comentarios por noticia, por día o por usuario.

### Diseño de la base de datos documental.

Para explicar el diseño de la base de datos, se ha implementado el siguiente diagrama:

Captura de pantalla de un celular

Descripción generada automáticamente

Ilustración ‑. Diagrama diseño Base de Datos. Blog de noticias.

En el que se pueden observar las siguientes colecciones: Contact (que puede ser tanto Address como Phone), User, Comment y New.

* **User**: Cada usuario registrado en el blog. Ejemplo de la colección:

|  |
| --- |
| {  "name": "Javier Piqueras",  "username": "javipiquerasm",  "twitter\_username": "javipiquerasm",  "description": "Hi, my name is Javi, and I'm here to read all the interesting news related Covid-19 topic",  "contact": {  "phone\_numbers": ["655601565", "65894523", "914568759"]  }  } |

* **Contact**: Cada contacto asociado a un usuario (puede ser de tipo **Address** o **Phone**), dependiendo el tipo de contacto que el usuario haya preferido. Ejemplos:

|  |
| --- |
| {  "street": "C/ Donoso Cortés",  "number": "9",  "door": "B",  "postal\_code": "28015",  "city": "Madrid"  }  {  "phone\_numbers": ["655601565", "65894523", "914568759"]  } |

* **New**: Cada noticia publicada en el blog. Ejemplo:

|  |
| --- |
| {  "\_id": "46258",  "title": "Bajan el número de casos nuevos del coronavirus",  "body": "En las últimas 24 horas, el número de nuevos confirmados por coronavirus en España ha descendido un 4.4%",  "publication\_date": 2020-04-15T12:01:000,  "tags": ["virus", "coronavirus", "covid19"],  "author\_username": "marcorecuenco\_",  "comments": [  {  "content": "Buenísimo artículo, ¡muchas gracias pro compartirlo!",  "publication\_date": 2020-04-15T12:15:000,  "author\_username": "javipiqueras"  }  ]  } |

* **Comment**: Cada comentario publicado por un usuario registrado en una noticia. Ejemplo:

|  |
| --- |
| {  "content": "Buenísimo artículo, ¡muchas gracias pro compartirlo!",  "publication\_date": 2020-04-15T12:15:000,  "author\_username": "javipiqueras"  } |

Nótese que las relaciones son tanto de tipo “*One-to-one*”, “*One-to-many*” embebido, como “*One-to-many*” con referencias.

### Definición de índices

Para optimizar las operaciones que el usuario debe ser capaz de hacer, resultaría interesante añadir los siguientes índices:

* Sobre la colección Users, un índice compuesto sobre los siguientes campos:
  + username
  + twitter\_username
  + contact.postal\_code
* Sobre la colección News, un índice compuesto sobre los siguientes campos:
  + autor\_username
  + publication\_date

Esto optimizará la búsqueda del usuario por cualquiera de ambos campos.

El código que inserta los índices se encuentra en el notebook correspondiente al ejercicio\_1 (ejercicio\_1.ipynb), en el apartado “Creación de Índices”.

### Inserciones

El código necesario para las inserciones se encuentra en el notebook correspondiente al ejercicio 1 (ejercicio\_1.ipynb), en el apartado “Inserciones”.

## Ejercicio 2

Para la realización del ejercicio 2 se proporciona el fichero “webs.json”, una colección que contiene información sobre las paginas web que han visitado distintos usuarios, y sobre la que vamos a realizar consultas y agregaciones. Cada documento se puede entender como una “sesión” realizada por el usuario y contiene el nombre de usuario, la fecha de inicio de la sesión, el navegador utilizado y un array que contiene las páginas visitadas. Para cada página visitada se almacena la URL y el número de segundos en los que la página ha estado activa.

El primer paso para la realización de este ejercicio será importar la colección almacenada en “webs.json” a nuestra base de datos MongoDB. Para ello, teniendo levantado nuestro contenedor Docker con nombre “<nombreContenedor>”, que corre el proceso mongod, primero copiaremos el fichero “webs.json” (situado en la carpeta desde la que ejecutamos el comando) a una carpeta temporal del contenedor:

**$> docker cp webs.json <nombreContenedor>:/tmp/webs.json**

A continuación, haremos uso de la utilidad “mongoimport” para efectivamente importar la colección:

**$> docker exec -it <nombreContenedor> mongoimport --db ejercicio2 --collection webs --file /tmp/webs.json**

Una vez que se ha importado la colección, nos conectaremos a la base de datos tal y como se ha indicado en el apartado 2 a través del notebook Jupyter y pymongo.

Se pide ejecutar las siguientes consultas y agregaciones:

* Consulta 1: Número de sesiones (documentos) en los que se ha visitado la web “https://www.mikokoloko.com”. Nota: si una sesión incluye dos visitas solo se contará una vez.
* Consulta 2: Mostrar los distintos navegadores que se han utilizado en toda la colección.
* Consulta 3: Mostrar únicamente el campo “navegador” de las sesiones realizadas por el usuario “Bertoldo” entre las fechas “20/02/2016” y “25/02/2016” (a las 00:00:00.000 horas en ambos casos).
* Agregación 1: Obtener el número de sesiones por cada navegador, ordenadas en orden ascendente. La salida mostrará, para cada navegador, un “\_id” que contiene el nombre del navegador, y un campo "total" con el número de sesiones de dicho navegador.
* Agregación 2: Obtener la persona con mayor número de sesiones totales en la colección. La salida mostrará para cada persona (sólo 1) un “\_id” conteniendo el nombre de la persona y un campo “total” con el número de sesiones totales.
* Agregación 3: Para cada persona indicar el numero de URLs visitadas durante más de 100 segundos (se cuentan las repeticiones). La salida mostrará para cada persona un “\_id” conteniendo el nombre de la persona y un campo “num” con el número de URLs. Pista: utilizar la opción $unwind de aggregate.
* Agregación 4: Nombre de la persona que visita más páginas de media por sesión. La salida mostrará un “\_id” conteniendo el nombre de la persona y un campo “media” con el valor de la media de páginas visitadas. Pistas: calcular el número de URLs para cada sesión y luego calcular la media de ese dato para cada persona / No hace falta utilizar $unwind.
* Agregación 5: Calcular el total de sesiones con el navegador “Explorer” en 2016. La salida mostrará el “\_id” conteniendo el año (2016) y un campo “total” con el número total de sesiones. Pista: para obtener el año de una fecha se puede utilizar {$year:”$fecha”}.
* Agregación 6: Para cada persona calcular el tiempo acumulado de todas las URLs visitadas, en minutos. La salida mostrará un campo “nombre” con el nombre de la persona, y un campo “tiempomin” con el tiempo acumulado. La salida estará́ ordenada alfabéticamente por los nombres de los usuarios. Pista: utilizar la opción $unwind de aggregate.
* Agregación 7: Obtener el nombre de las personas que han visitado alguna página que comience por “https”. La salida mostrará para cada persona únicamente el campo “nombre” que contendrá su nombre. Pista: utilizar el operador de proyección $substr.

El código necesario para las consultas y agregaciones se encuentra en el notebook correspondiente al ejercicio 2 (ejercicio\_2.ipynb).