GAI – TP2 – Mongo DB

Adrián José Zapater Reig

Ejercicio 1 – Diseño de BD Documental

Diagrama UML:

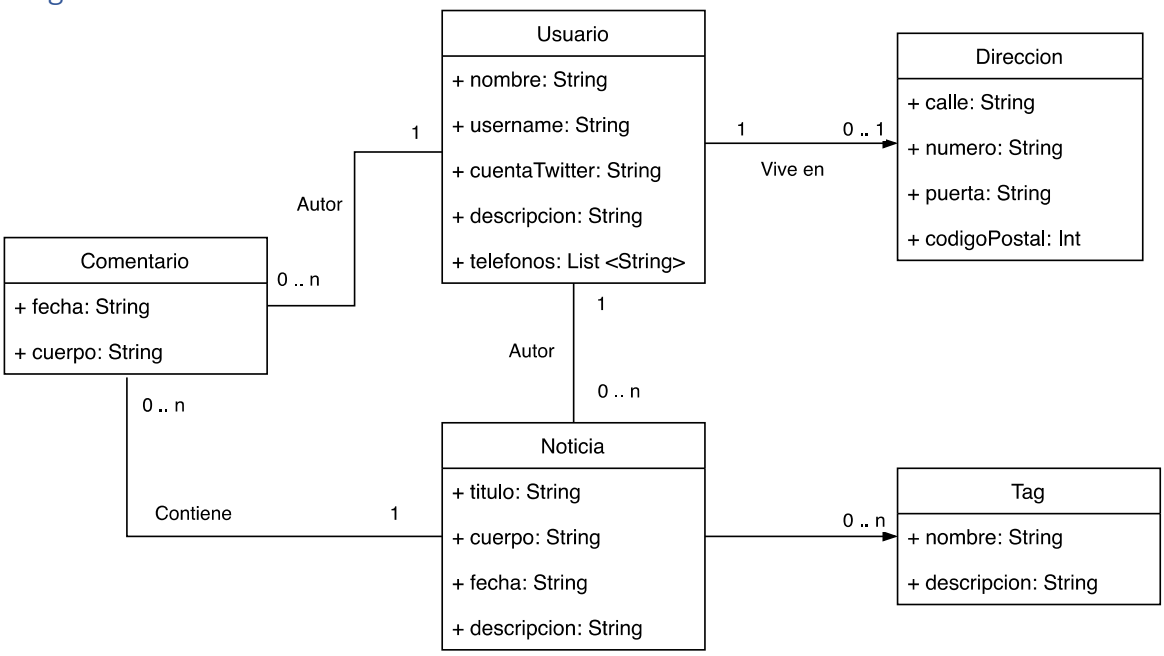

Ilustración 1 - Diagrama UML de blog de noticias

Decisiones de Diseño:

- Se ha decidido mantener Tag como una entidad independiente de Noticia ya que actuará como un ENUM. Un Tag no tiene porque saber de la existencia de Noticias. Por ese motivo la relación de dependencia va desde Noticia, que si que necesita saber qué Tags tiene asociados, a Tag. Cada Tag será un documento independiente dentro de una colección que se referenciará desde Noticia.
- Dirección se ha separado de Autor en una entidad a parte con una relación de dependencia de Autor a Dirección. Se ha decidido tomar esta decisión para representar Dirección como un Documento Embebido dentro de Autor.
- La entidad Comentario y Autor tienen una relación de asociación. La relación entre un documento de cada entidad será por referencia porque no queremos cargar todos los comentarios de un autor al buscar por autor, ni cargar toda la información del autor al buscar por comentario.
- La entidad Autor y Noticia tienen una relación de asociación por referencia por el mismo motivo que la relación Comentario-Autor.

- La relación Noticia-Comentario es de asociación. La relación se realizará por referencia ya que el contenido de un Comentario / Noticia puede ser muy extenso y pesado.
- La relación Autor – Noticia es de asociación. Como en el caso anterior la relación será por referencia para no tener que cargar todas las Noticias de un Autor al buscar por Autor.

Diseño de la Base de Datos:

Basándonos en el diagrama UML de la ilustración 1, se pueden extraer las siguientes colecciones:

Usuario:

Tabla 1 – Colección Usuario

```
{
  "nombre": "Adrian",
  "username": "adri95",
  "cuentaTwitter": "adrian95Twitter",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. ",
  "telefonos": [
    "61609077563",
    "61609077562"
  ],
  "direccion": {
    "calle": "Calle de la Bolsa",
    "numero": "110",
    "puerta": "3C",
    "codigoPostal": 28012
  }
}
```

La colección **Usuario** contiene la entidad *Usuario* y *Direccion* con una relación One-To-One embebida. Se ha optado por este tipo de relación ya que la dirección no ocupa mucho espacio y se hacen búsquedas agregadas por *codigoPostal*.

Comentario

Tabla 2 – Colección Comentario

```
{
  "fecha": "2020-04-06T10:00:000",
  "cuerpo": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "autorUsername": "adri95",
  "noticiaId": 123
}
```

La colección **Comentario** recogerá la entidad Comentario. Esta entidad tiene una relación Many-To-One referencial con *Noticia* y con *Usuario* usando _id y username respectivamente. *Comentario* referencia a ambas entidades pero *Usuario* y *Noticia* no ya que si se busca un comentario de un usuario/noticia en concreto, se puede usar el campo *autorUsername* / noticiaId*.*

Tag

Tabla 3 – Colección Tag

```
{
  "nombre": "etiqueta1",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit."
}
```

La colección **Tag** recoge la información del Tag que se puede asignar a una noticia. Tiene una relación Many-To-Many referencial con Noticia, pero sólo se referencia desde Noticia usando el campo *nombre*. De esta manera se pueden crear los tags de forma aislada a las noticias y se puede consultar una lista de tags disponibles que asignar a una noticia. Si se quiere obtener todas las noticias de un tag en concreto, siempre existe la posibilidad de filtrar por la referencia de *Noticia* a *Tag.*

Noticia

Tabla 4 – Colección Noticia

```
{
  "_id": 123,
  "titulo": "Titulo1",
  "cuerpo": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "fecha": "2020-04-06T10:00:000",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "autorUsername": "adri95",
  "tags": [
    "etiqueta1",
    "etiqueta2",
    "etiqueta3"
  ]
}
```

La colección **Noticia** contiene las noticias del Blog. Tiene una relación Many-To-Many referencial con *Tag*, aunque sólo se referencia desde *Noticia*. *Noticia* tiene una relación Many-To-One referencial con *Usuario* aunque sólo se referencia desde Noticia, esto nos permite filtrar las noticias por usuario sin necesitar tener los *_id*s de *Noticia* en la entidad *Usuario.*

Diseño de índices:

Las consultas mas frecuentes son:

- 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.

Usuario:

Para esta colección se espera tener muy pocas escrituras pero muchas lecturas (Un usuario solo se crea una vez y se actualiza en pocas ocasiones). Las lecturas se realizarán principalmente por *username*, *cuentaTwitter* y *dirección.codigoPostal*, por lo que se recomienda crear un índice para cada uno. En este caso, el orden (1 o -1) del índice no es relevante porque no hay ordenación. Usaremos ordenación ascendente 1.

Tabla 5

```
use gai_db


db.Usuario.createIndex( {"username": 1})
db.Usuario.createIndex( {"cuentaTwitter": 1})
db.Usuario.createIndex( {"direccion.codigoPostal": 1})
```

Noticia:

Esta colección espera tener muchas escrituras y muchas mas lecturas. Además, la mayor parte de las lecturas serán ordenadas. También hay que tener en cuenta que la colección es referenciada por Comentario y que necesita un índice único que identifique cada noticia.
Se ha decidido usar 2 índices, uno simple proporcionado por defecto por mongo (_id) y otro un índice compuesto usando *autorUsername* y *fecha*. Como se va a consultar con frecuencia “las últimas N noticias de un usuario”, *autorUsername* será un índice ordenado ascendentemente y *fecha* descendentemente.

Tabla 6

```
db.Noticia.createIndex({autorUsername:1, fecha: -1})
```

Comentario:

Esta colección, como Noticia, se escribirá mucho y se leerá mucho. Las consultas mas frecuentes incluyen las consultas por usuario, noticia y fecha. Para las primeras dos se propone usar un índice simple para autorUsername y noticiaId respectivamente. Para fecha tenemos 2 alternativas:

1. Tener un campo mas llamado *fechaDia* en Comentario que sea la fecha con precisión hasta día (2020-04-06) además de la fecha exacta del comentario.
2. Tener un índice simple sobre el campo *fecha* y buscar usando $regex. Ver ejemplo de tabla 7.

Tabla 7 – Ejemplo alternativa 2.

```
db.Comentario.find({fecha: {$regex: "2020-04-07"}})
```

Tabla 8

```
db.Comentario.createIndex({ autorUsername : 1 })
db.Comentario.createIndex({ noticiaId : 1 })

# Alternativa 1
db.Comentario.createIndex({ fechaDia : 1 })

# Alternativa 2
db.Comentario.createIndex({ fecha : 1 })
```

Tag:

Recibirá muchas búsquedas y pocas escrituras. El caso de uso mas frecuente que me viene a la mente es el de elegir un Tag desde una lista de posibilidades o cargar una lista de Tags que mas se aproximen a una cadena de texto en su nombre o descripción. Para poder realizar estas búsquedas se puede utilizar los índices de texto proporcionados por Mongo. Estos índices nos permiten dar un peso a la "importancia" de encontrar una palabra que machee en uno de los campos, para este caso de uso tiene mas valor acertar el nombre del Tag que una palabra en la descripción por lo que he optado por poner 20 y 5 de peso a *nombre* y *descripción* respectivamente.

Tabla 9

```
> db.Tag.createIndex(
```

```
... {
...  nombre: "text",
...  descripcion: "text"
... },
... {
...  weights: {
...    nombre: 20,
...    descripcion: 5
...  }
... }
... )
{
```

Por ejemplo:
Si busco el texto “amet” (ver tabla 10), el primer resultado es el que tiene amet como nombre aunque haya otros documentos con “amet” en la descripción (ver tabla 11).

Tabla 10

```
db.Tag.find( { $text: { $search: "amet" } } )
```

Tabla 11

```
{ "nombre" : "amet", "descripcion" : "Lorem ipsum dolor sit amet, consectetur adipiscing elit." }
{  "nombre" : "etiqueta6", "descripcion" : "Lorem ipsum dolor sit amet, consectetur adipiscing elit." }
{ "nombre" : "etiqueta1", "descripcion" : "Lorem ipsum dolor sit amet, consectetur adipiscing elit." }
{ "nombre" : "etiqueta7", "descripcion" : "Lorem ipsum dolor sit amet, consectetur adipiscing elit. etiqueta6" }
```

Si busco “etiqueta6” (ver tabla 12), el primer resultado es el comentario con nombre “etiqueta6” a pesar de que no lo tiene en la descripción. El segundo resultado es un comentario que lo tiene como palabra en la descripción.

Tabla 12

```
{ "nombre" : "etiqueta6", "descripcion" : "Lorem ipsum dolor sit amet, consectetur adipiscing elit." }
{ "nombre" : "etiqueta7", "descripcion" : "Lorem ipsum dolor sit amet, consectetur adipiscing elit. etiqueta6" }
```

Insertar datos en la BD:

Se proporciona el código para insertar 2 elementos en cada colección:

Usuario:

Tabla 13

```
db.Usuario.insert(
[
 {
  "nombre": "Adrian",
  username": "Adrian95",
  "cuentaTwitter": "Adrian95Twitter",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. ",
  "telefonos": [    "61609077563",    "61609077562"   ],
  "direccion": {
   "calle": "Calle de la Bolsa",
   "numero": "110",
   "puerta": "3C",
   "codigoPostal": 28012
  }
 },
 {
  "nombre": "Miguel",
  username": "miguel95",
  "cuentaTwitter": "miguel95Twitter",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. ",
  "telefonos": [    "61609077563",    "61609077562"   ],
  "direccion": {
   "calle": "Calle de la Bolsa",
   "numero": "110",
   "puerta": "3C",
   "codigoPostal": 28011
  }
 }
]
)
```

Noticia:

Tabla 14

```
db.Noticia.insert(
[
 {
  "_id": 3,
  "titulo": "Titulo1",
  "cuerpo": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "fecha": "2020-04-07T12:00:000",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "autorUsername": "miguel95",
  "tags": [    "etiqueta1",    "etiqueta2",    "etiqueta3"   ]
 },
 {
  "_id": 4,
  "titulo": "Titulo2",
  "cuerpo": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "fecha": "2020-04-07T13:00:000",
  "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "autorUsername": "miguel95",
  "tags": [    "etiqueta1",    "etiqueta2",    "etiqueta3"   ]
 }
]
)
```

Comentario:

Tabla 15

```
db.Comentario.insert(
[
 {
  "fecha": "2020-04-06T10:00:000",
  "cuerpo": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "autorUsername": "adri95",
  "noticiaId": 123
 },
 {
  "fecha": "2020-04-06T11:00:000",
  "cuerpo": "Lorem ipsum dolor sit amet, consectetur adipiscing elit.",
  "autorUsername": "adri95",
  "noticiaId": 124
 }
]
)
```

Tag:

Tabla 16

```
db.Tag.insert(
[
  {
    "nombre": "etiqueta1",
    "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit."
  },
  {
    "nombre": "etiqueta2",
    "descripcion": "Lorem ipsum dolor sit amet, consectetur adipiscing elit. etiqueta1"
  }
]
)
```

Ejercicio 2

Ejercicio resuelto y comentado en el jupyter notebook ejercicio_2.ipynb.

Ejercicio 3

Ejercicio resuelto y comentado en el jupyter notebook ejercicio_3.ipynb.