

SPRINT S9 – MONGO DB - BASE DE DATOS NO RELACIONAL

NIVEL 1

Se trabajará con una base de datos que contiene colecciones relacionadas con una aplicación de entretenimiento cinematográfico.

users: Almacena información de usuarios/as, incluyendo nombres, emails y contraseñas cifradas.

theatres: Contiene datos de cines, como ID, ubicación (dirección y coordenadas geográficas).

sesiones: Guarda sesiones de usuario, incluyendo ID de usuario y tokens JWT para la autenticación.

movies: Incluye detalles de películas, como trama, géneros, duración, elenco, comentarios, año de lanzamiento, directores, clasificación y premios.

comments: Almacena comentarios de usuarios/as sobre películas, con información del autor/a del comentario, ID de la película, texto del comentario y la fecha.

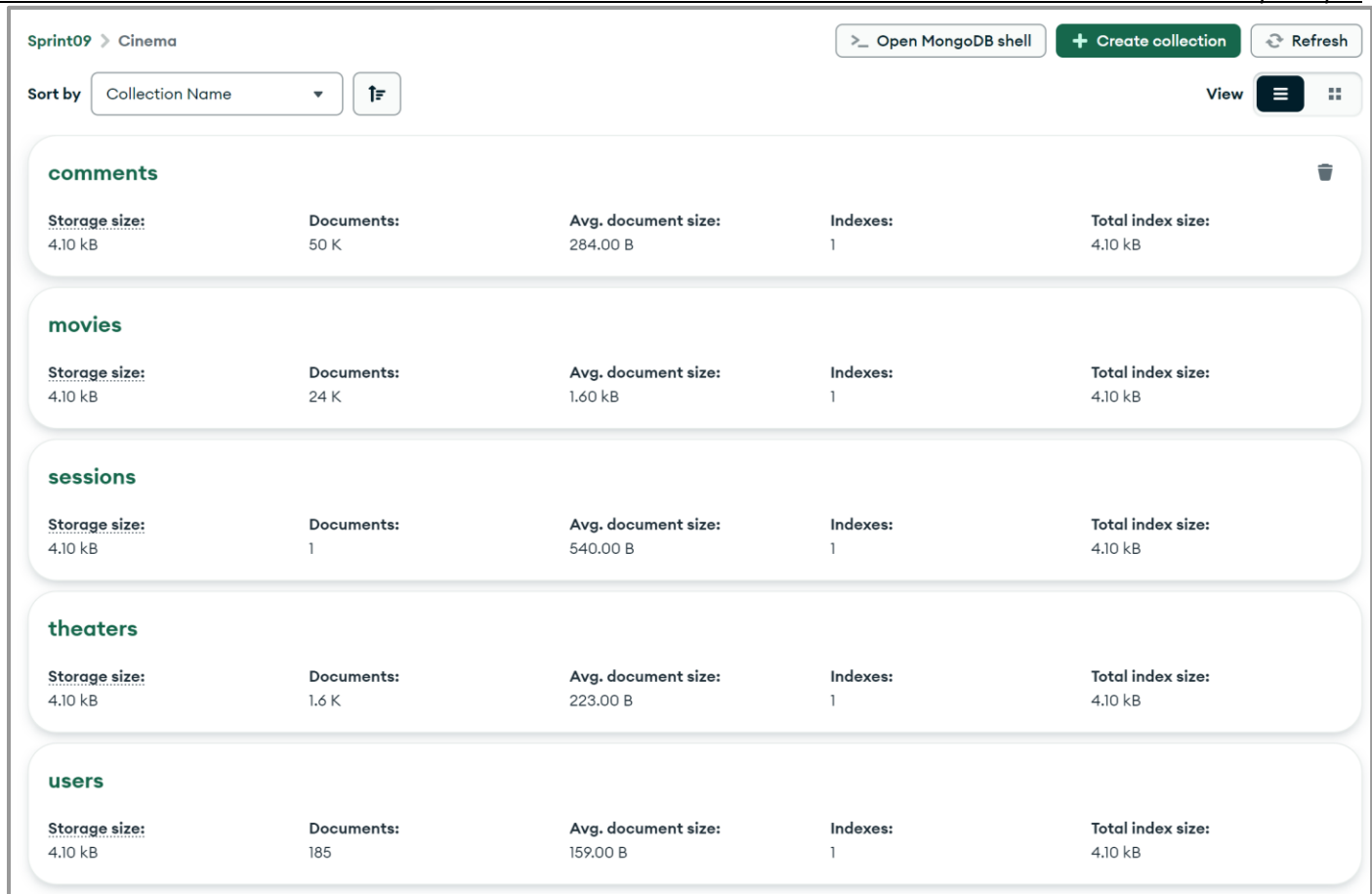
Se realizarán algunas consultas que pida el cliente/a, quien está midiendo si se tiene la capacidad o no de hacernos cargo de la parte analítica del proyecto vinculado con su base de datos.

Ejercicio 1.1

En primer lugar se crea la base de datos con Mongo DB Compass, la cual denominamos **Cinema**.

En segundo lugar se crearán colecciones para cada archivo de datos proporcionado. Estas colecciones serán parte de la base de datos **Cinema**.

Luego de cargar todos los archivos como colecciones, la base de datos se visualizará como sigue:



Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
4.10 kB	50 K	284.00 B	1	4.10 kB
4.10 kB	24 K	1.60 kB	1	4.10 kB
4.10 kB	1	540.00 B	1	4.10 kB
4.10 kB	1.6 K	223.00 B	1	4.10 kB
4.10 kB	185	159.00 B	1	4.10 kB

A través de **show dbs** se verifican todas las bases de datos que existen en la conexión actual:

```
> show dbs
< Cinema    28.27 MiB
  admin      40.00 KiB
  comments   8.00 KiB
  config     28.03 MiB
  local      72.00 KiB
  test01     8.00 KiB
```

- Se muestran los 2 primeros comentarios que hay en la base de datos.

Código utilizado: **db.comments.find().limit(2)**

Donde:

- **find()** es el equivalente a where en MySQL y realizará un filtrado según el argumento ingresado.

- **limit()** limita la muestra del resultado según el argumento ingresado, en este caso los 2 primeros comentarios.

```
> db.comments.find().limit(2)
< {
  _id: ObjectId('5a9427648b0beebe69579cc'),
  name: 'Andrea Le',
  email: 'andrea_le@fakegmail.com',
  movie_id: ObjectId('573a1390f29313caabcd418c'),
  text: 'Rem officiis eaque repellendus amet eos doloribus. Porro dolor voluptatum voluptates neque culpa molestias. Voluptate
  date: 2012-03-26T23:20:16.000Z
}
{
  _id: ObjectId('5a9427648b0beebe69579cf'),
  name: 'Greg Powell',
  email: 'greg_powell@fakegmail.com',
  movie_id: ObjectId('573a1390f29313caabcd41b1'),
  text: 'Tenetur dolorum molestiae ea. Eligendi praesentium unde quod porro. Commodi nisi sit placeat rerum vero cupiditate neq
  date: 1987-02-10T00:29:36.000Z
}
Cinema >
```

Resultado: los 2 primeros comentarios son de Andrea Le y Greg Powell.

- ¿Cuántos usuarios hay registrados?

Código utilizado: **db.users.find().count()**

Donde: **find()** es el equivalente a **where** en MySQL y realizará un filtrado según el argumento ingresado. Y **count()** realiza el conteo.

```
>_ mongosh: Sprint09 | Sprint09
>_MONGOSH
> db.users.find().count()
< 185
```

Resultado: 185 usuarios registrados.

- ¿Cuántos cines existen en el estado de California?

Código utilizado: **db.theaters.find({"location.address.state":"CA"}).count()**

Dentro de `find()` se especifica que únicamente queremos aquellos valores que cumplan estar en el estado CA, es decir California.

```
> db.theaters.find({"location.address.state":"CA"}).count()
< 169
Cinema > |
```

Resultado: 169 cines en el estado de California.

- ¿Cuál fue el primer usuario en registrarse?

Código utilizado: `db.users.find().sort({_id:1}).limit(1)`

Donde: `sort()` realizará una ordenación a partir del argumento introducido.

```
> db.users.find().sort({_id:1}).limit(1)
< {
  _id: ObjectId('59b99db4cfa9a34dcd7885b6'),
  name: 'Ned Stark',
  email: 'sean_bean@gameofthron.es',
  password: '$2b$12$UREFwsRUoyF0CRqGNK0Lz00HM/jLhgUCNNIJ9RJAqMUQ74crLJ1Vu'
}
Cinema > |
```

Resultado: Ned Stark, como primer usuario registrado.

- ¿Cuántas películas de comedia existen en la base de datos?

Código utilizado: `db.movies.find({$and:[{type:"movie"},{genres:"Comedy"}]}).count()`

En donde `$and` es la conjunción que vendría a establecer que se realice un filtro considerando todas las condiciones que le siguen

```
> db.movies.find({$and:[{type:"movie"}, {genres:"Comedy"}]}).count()
< 7002
Cinema > |
```

Resultado: escogiendo como genero comedia y tipo movie obtenemos el valor de 7002.

Ejercicio 1.2

Mostrar todos los documentos de las películas producidas en 1932, pero que el género sea drama o estén en francés.

Código utilizado:

```
db.movies.find({$and:[{type:"movie"},{year:1932},{$or:[{genres:"Drama"},{languages:"French"}]}})
```

```
> db.movies.find({$and:[{type:"movie"},{year:1932}, {$or:[{genres:"Drama"},{languages:"French"}]}})
< {
  _id: ObjectId('573a1391f29313caabcd9458'),
  plot: 'A young artist draws a face at a canvas on his easel. Suddenly the mouth on the drawing comes into life and starts',
  runtime: 55,
  rated: 'UNRATED',
  cast: [
    'Enrique Rivero',
    'Elizabeth Lee Miller',
    'Pauline Carton',
    'Odette Talazac'
  ],
  num_mflix_comments: 1,
  poster: 'https://m.media-amazon.com/images/M/MV5BYWY3ODE5ZWEtYjlmVi00NjA4LTk4ZWYtMzBhZDE5MjY0YTYxXkEyXkFqcGdeQXVyNzI4MDM',
  title: 'The Blood of a Poet',
  lastupdated: '2015-09-16 13:13:05.537000000',
  languages: [
    'French'
  ],
  released: 2010-05-20T00:00:00.000Z,
  directors: [
    'Jean Cocteau'
  ],
  writers: [
    'Jean Cocteau'
  ],
  awards: {
    wins: 1,
```

```
> db.movies.find({$and:[{type:"movie"},{year:1932},{$or:[{genres:"Drama"},{languages:"French"}]}]}).count()  
< 18  
Cinema >
```

En la primera imagen se muestra el listado y en la segunda el conteo agregándole la función `count()` al código anterior:

```
db.movies.find({$and:[{type:"movie"},{year:1932},{$or:[{genres:"Drama"},{languages:"French"}]}]}).count()
```

Resultado: 18.

Ejercicio 1.3

Mostrar todos los documentos de películas estadounidenses que tengan entre 5 y 9 premios que fueron producidas entre 2012 y 2014.

Código utilizado:

```
db.movies.find({countries:"USA", "awards.wins": {$gte:5,$lte:9},year:{$gte:2012, $lte:2014}})
```

en donde:

\$gte: *greater than or equal to* (mayor o igual que)

\$lte: *lesser than or equal to* (menor o igual que)

```
>_MONGOSH
```

```
> db.movies.find({countries:"USA", "awards.wins": {$gte:5,$lte:9},year:{$gte:2012, $lte:2014}})
< {
  _id: ObjectId('573a13acf29313caabd29366'),
  fullplot: "The manager of the negative assets sector of Life magazine, Walter Mitty, has been working for sixteen years for
  imdb: {
    rating: 7.4,
    votes: 211230,
    id: 359950
  },
  year: 2013,
  plot: 'When his job along with that of his co-worker are threatened, Walter takes action in the real world embarking on a g
  genres: [
    'Adventure',
    'Comedy',
    'Drama'
  ],
  rated: 'PG',
  metacritic: 54,
  title: 'The Secret Life of Walter Mitty',
  lastupdated: '2015-08-31 00:10:51.747000000',
  languages: [
    'English',
    'Spanish',
    'Icelandic'
  ],
  writers: [
    'Steve Conrad (screenplay)',
    'Steve Conrad (screen story by)',
    'James Thurber (based on the short story by)'
  ],
  type: 'movie',
```

En la primera imagen se muestra el listado y en la segunda el conteo agregándole la función `count()` al código anterior:

```
db.movies.find({countries:"USA", "awards.wins": {$gte:5,$lte:9},year:{$gte:2012, $lte:2014}}).count()
```

```
> db.movies.find({countries:"USA", "awards.wins": {$gte:5,$lte:9},year:{$gte:2012, $lte:2014}}).count()
< 166
Cinema> |
```

Resultado: 166 serían las películas que cumplen con las características solicitadas.

NIVEL 2**Ejercicio 2.1**

Observar cuántos comentarios escribe un usuario que utiliza "GAMEOFTHRON.ES" como dominio de correo electrónico.

Se utiliza el **\$regex** para indicar que el dominio del email debe ser "gameofthron.es". Se utiliza el símbolo "i" para que sea indistinto el uso de minúsculas o mayúsculas.

```
<
> db.comments.find({
  email: {$regex: /@gameofthron.es$/i}
}).count()
< 22841
```

Resultado: 22.841 comentarios realizados por usuarios cuyo dominio de email finalice en "gameofthron.es".

Ejercicio 2.2

Cuántos cines hay en cada código postal ubicado dentro del estado de Washington D. C. (DC)?

En la consulta se utiliza el **\$match** para filtrar por estado, que en este caso sería: DC. Además, se utiliza **\$group** para agrupar por código postal. Finalmente a través de **\$count** se realizará el recuento de cines por cada código postal.

```
> db.theaters.aggregate([
  {$match: {'location.address.state': 'DC'}},
  {$group: {_id: '$location.address.zipcode',
    numero_cinemas: {$count: {}}}}
])
< {
  _id: '20010',
  numero_cinemas: 1
}
{
  _id: '20016',
  numero_cinemas: 1
}
{
  _id: '20002',
  numero_cinemas: 1
}
Cinema > |
```


Resultado: serían 3 códigos postales distintos, indicando la existencia de un solo cine en cada uno de ellos.

NIVEL 3

Ejercicio 3.1

Mostrar todos las películas que fueron dirigidas por John Landis con una puntuación IMDb (internet Movie Database) de entre 7,5 y 8.

Realizando la consulta:

```
> db.movies.find({
  'type':'movie',
  'directors':'John Landis',
  'imdb.rating':{'$gte':7.5,'$lte':8}
}) .count()
< 4
Cinema >
```

Resultado: 4 películas dirigidas por John Landis, las cuales se detallan a continuación:

```
<
> db.movies.find({
  'type':'movie',
  'directors':'John Landis',
  'imdb.rating':{'$gte':7.5,'$lte':8}
})
< [
  {
    _id: ObjectId('573a1397f29313caabce6d94'),
    fullplot: "Faber College has one frat house so disreputable it will take anyone. It has a second one full of white, anglo-saxon, rich
    imdb: {
      rating: 7.6,
      votes: 84834,
      id: 77975
    },
    year: 1978,
    plot: 'At a 1962 college, Dean Vernon Wormer is determined to expel the entire Delta Tau Chi Fraternity, but those trouble-makers hav
    genres: [
      'Comedy'
    ],
    rated: 'R',
    metacritic: 82,
    title: 'Animal House',
    lastupdated: '2015-09-13 00:02:47.803000000',
    languages: [
      'English',
      'Italian'
    ],
    writers: [
      'Harold Ramis',
      'Douglas Kenney',
      'Chris Miller'
```

Ejercicio 3.2

Mostrar en un mapa la ubicación de todos los teatros de la base de datos.

Se tiene información de la geolocalización a través de las coordenadas, entonces sencillamente se ha seleccionado '**coordinates**' para visualizar a través de puntos la localización, como se muestra a continuación:

