

ENTREGABLE: sprint 05

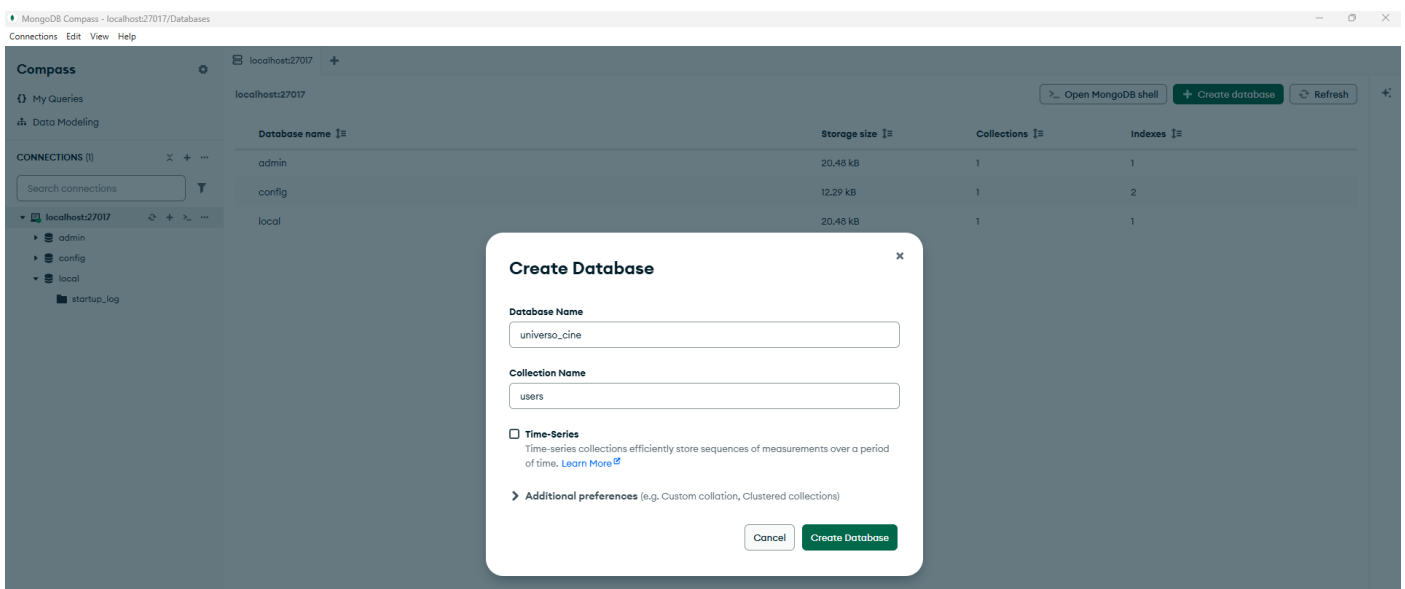
ALUMNO: Ignacio Soldevilla Royo

FECHA: 10-11-2025

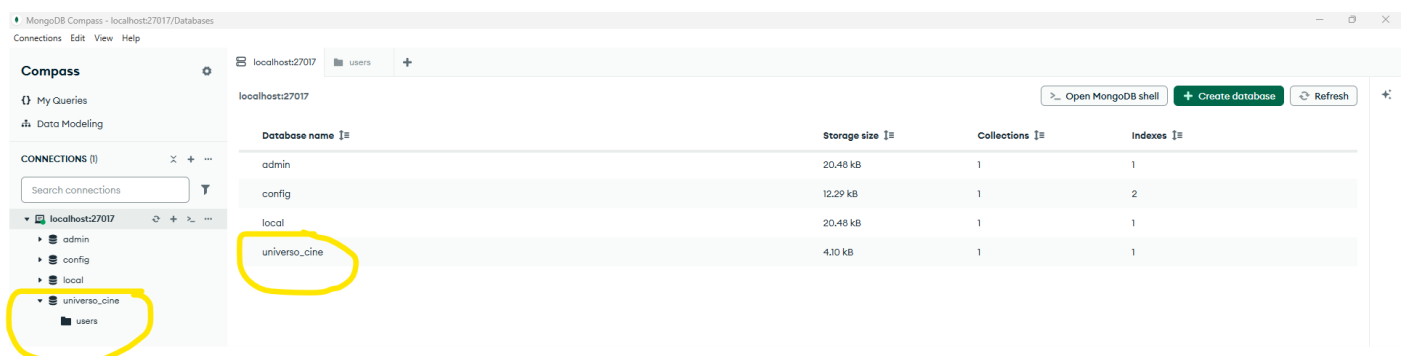
NIVEL 1

Crea una base de datos con MongoDB utilizando como colecciones los archivos adjuntos.

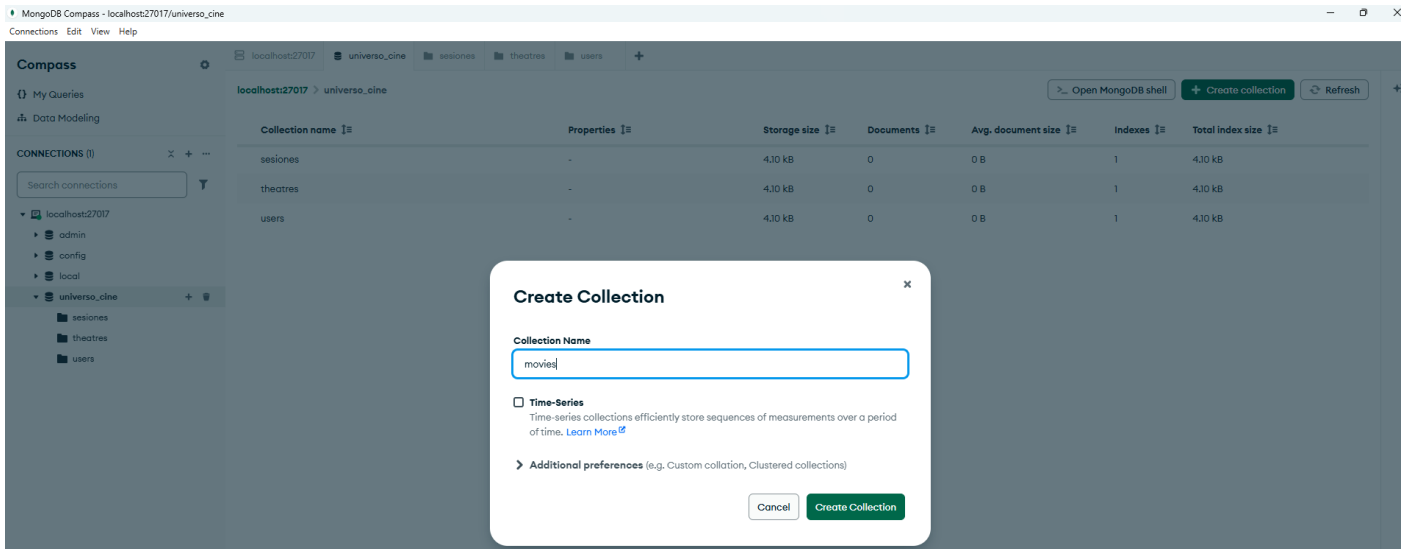
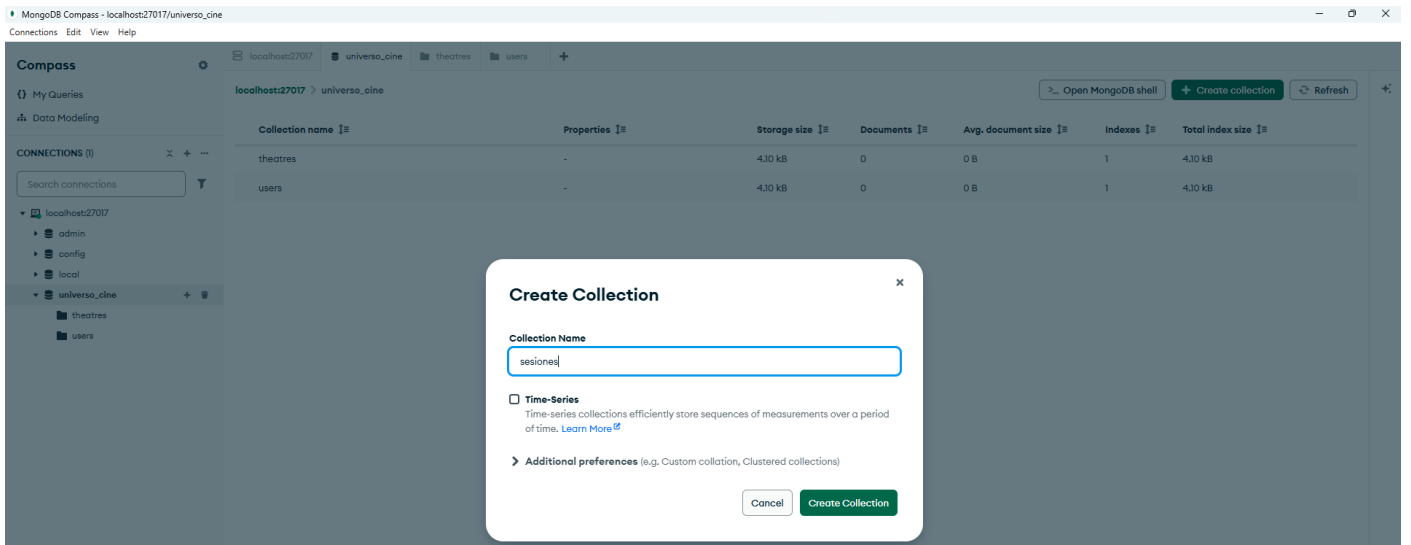
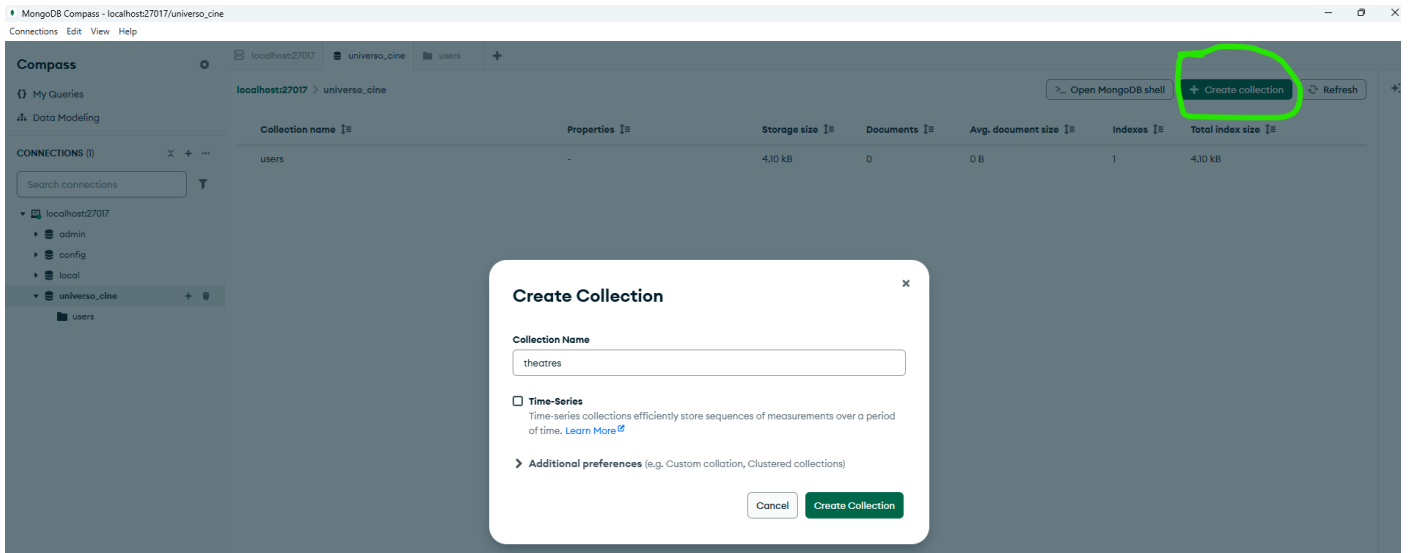
1. La base de datos la voy a llamar “universo_cine” y la voy a crear desde compass junto con la primera colección “users”, ya que siempre al crear una BD hay que crearla con 1 colección

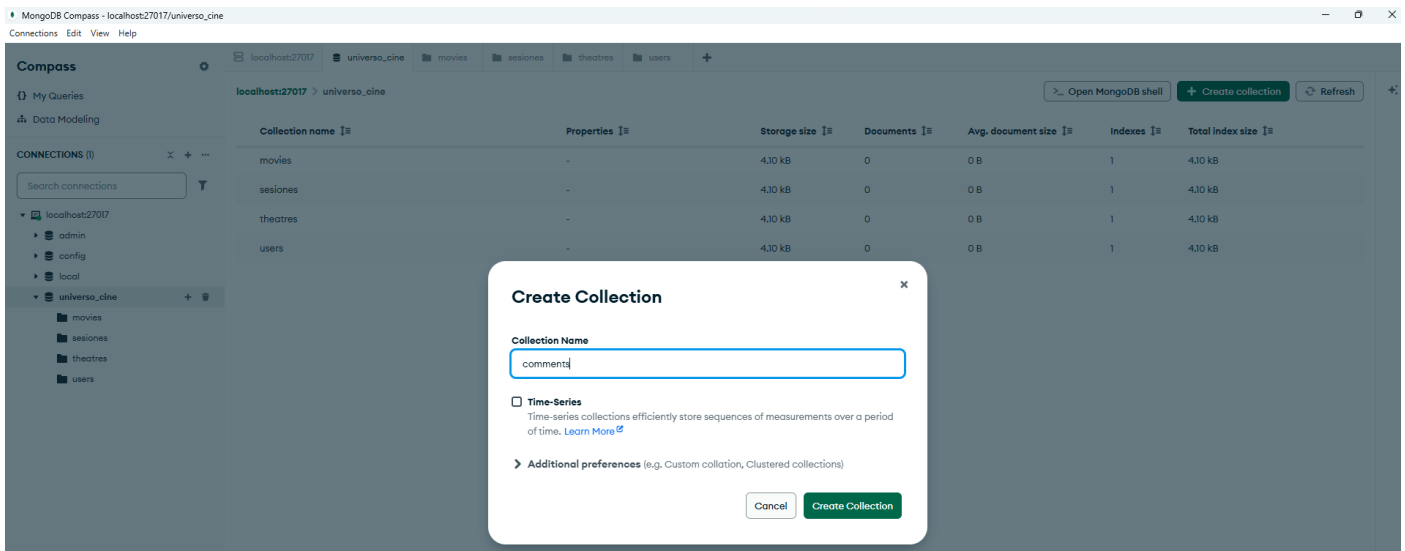


2. Compruebo que se ha creado tanto la base de datos como la colección:

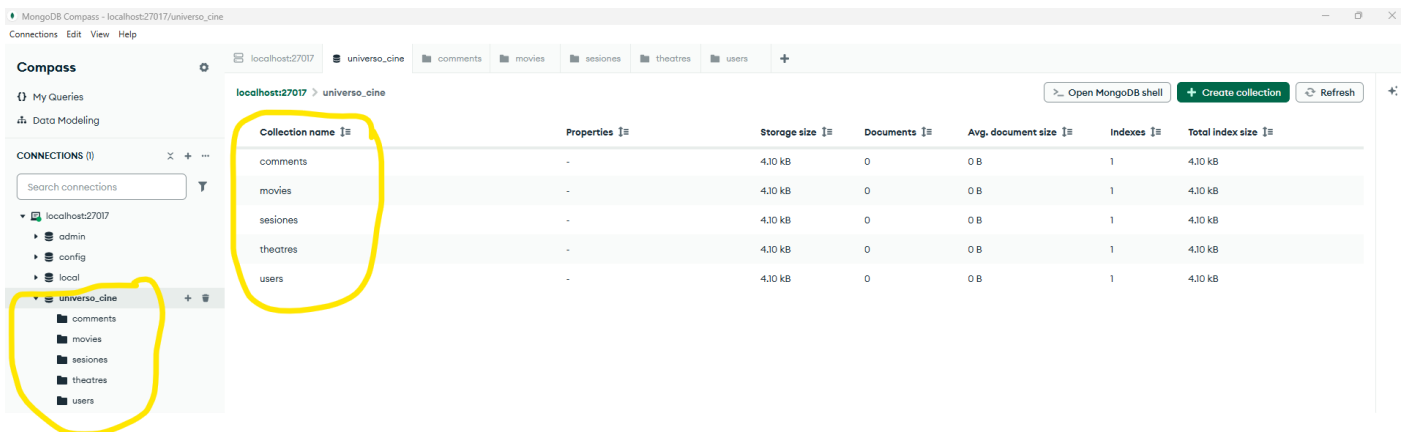


3. Creo el resto de colecciones: “theatres”, “sesiones”, “movies” y “comments”, entrando en la BD y con el botón de “+ Create collection”. Hay que repetir esta acción para cada nueva colección:

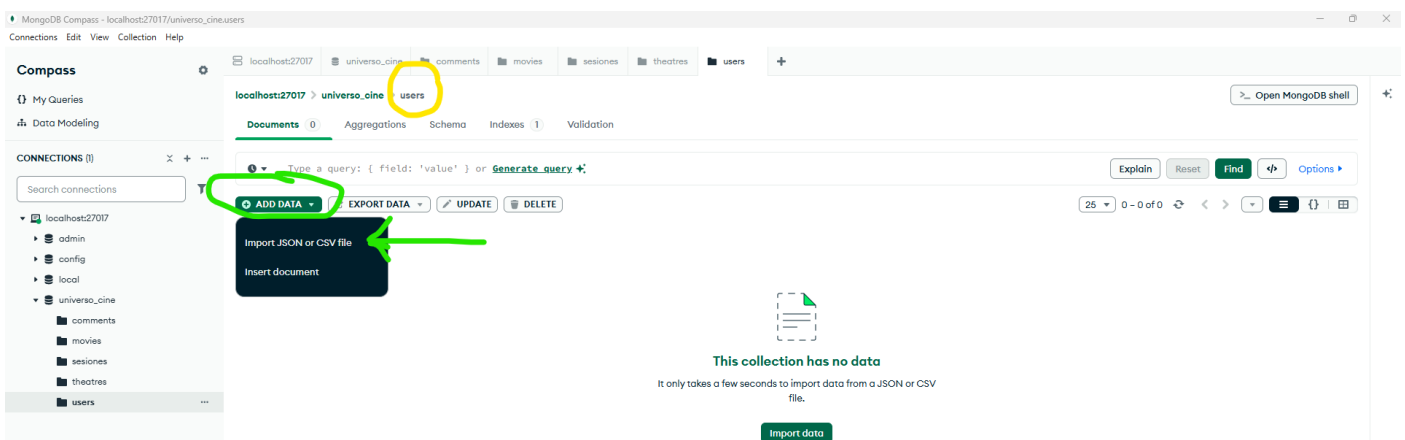




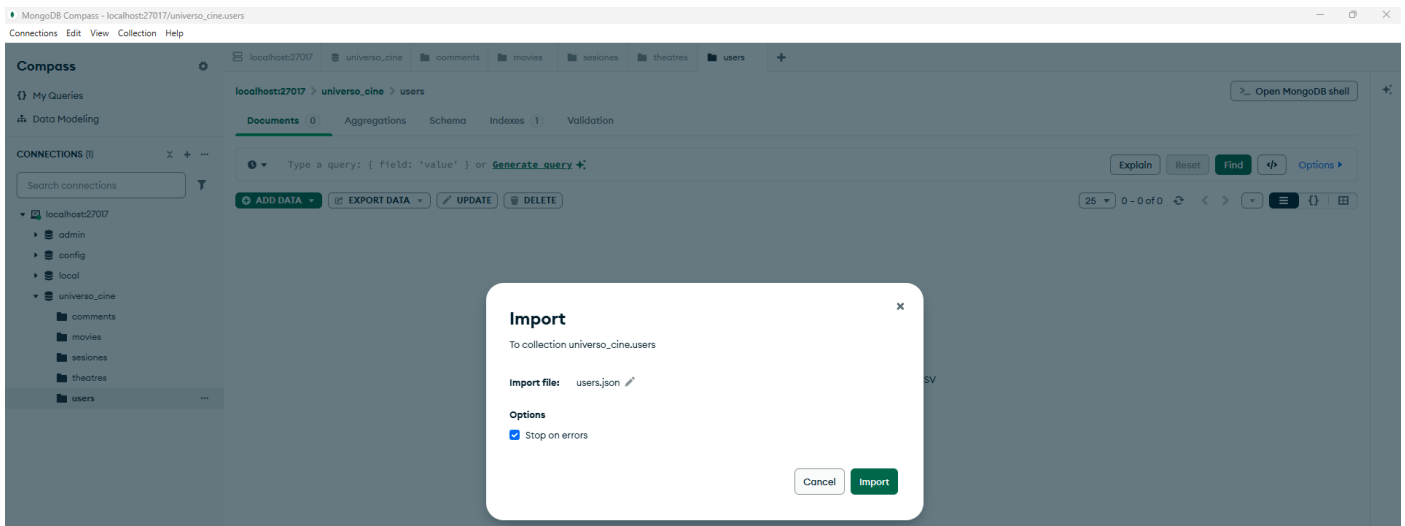
4. Compruebo que ya tengo las 5 colecciones creadas en la base de datos o contenedor llamado “universo_cine”:



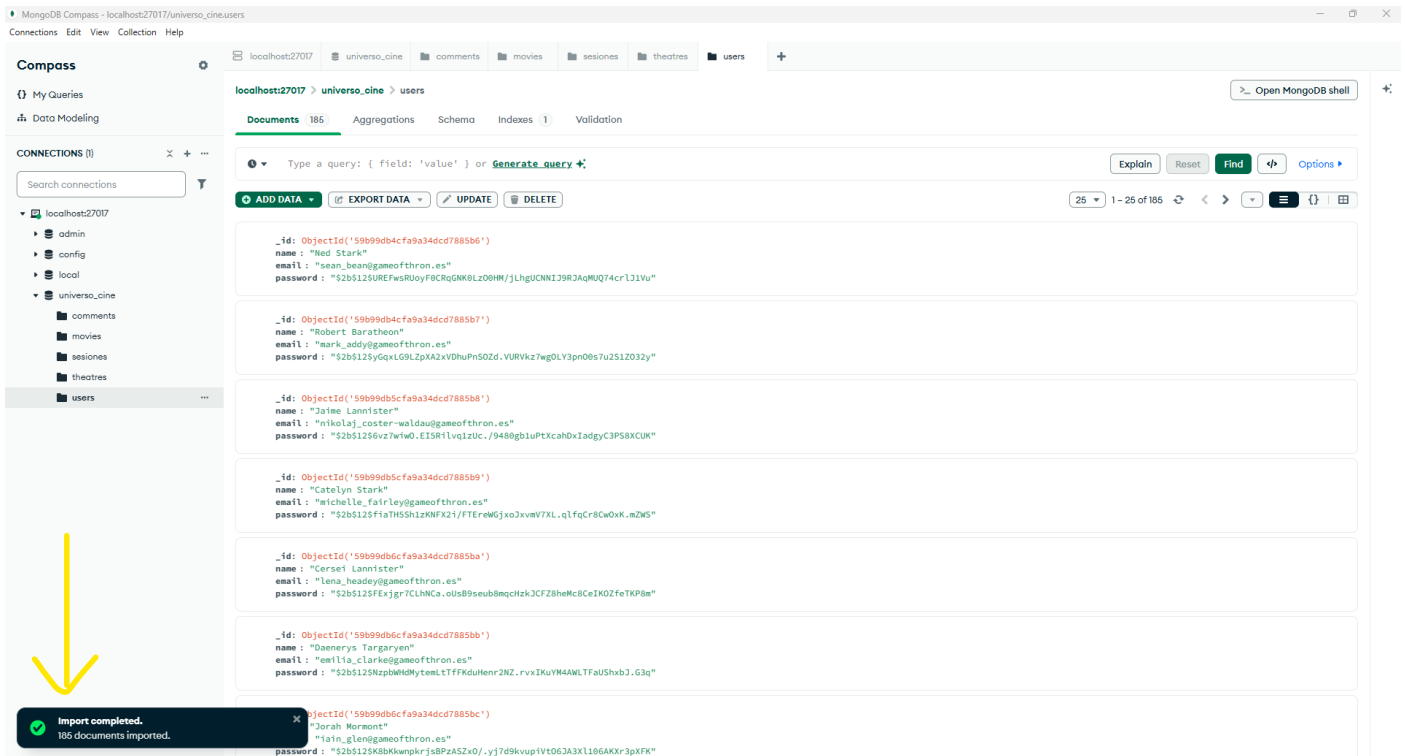
5. Una vez creadas las colecciones, hay que importar los documentos de los ficheros json que me da el ejercicio. Para ello empiezo por la colección “users”. Entro en la colección y en el botón “+ ADD DATA” cojo la opción “Import JSON or CSV file”:



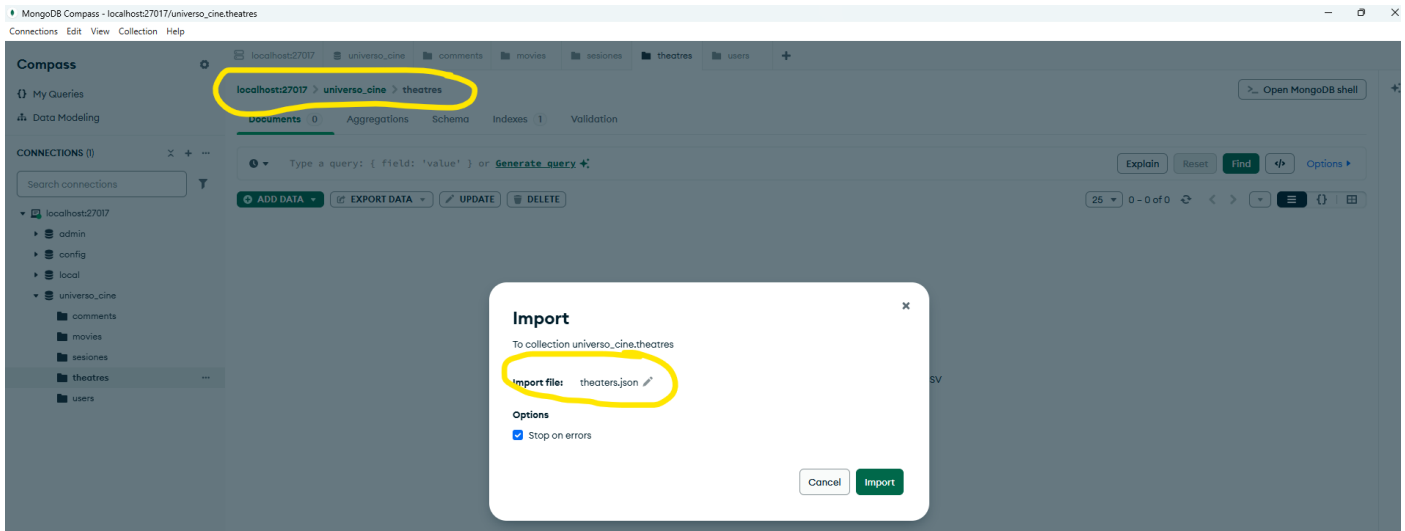
Voy a marcar la opción “Stop on errors” para tener 100% claro si hay errores o no al importar los documentos:



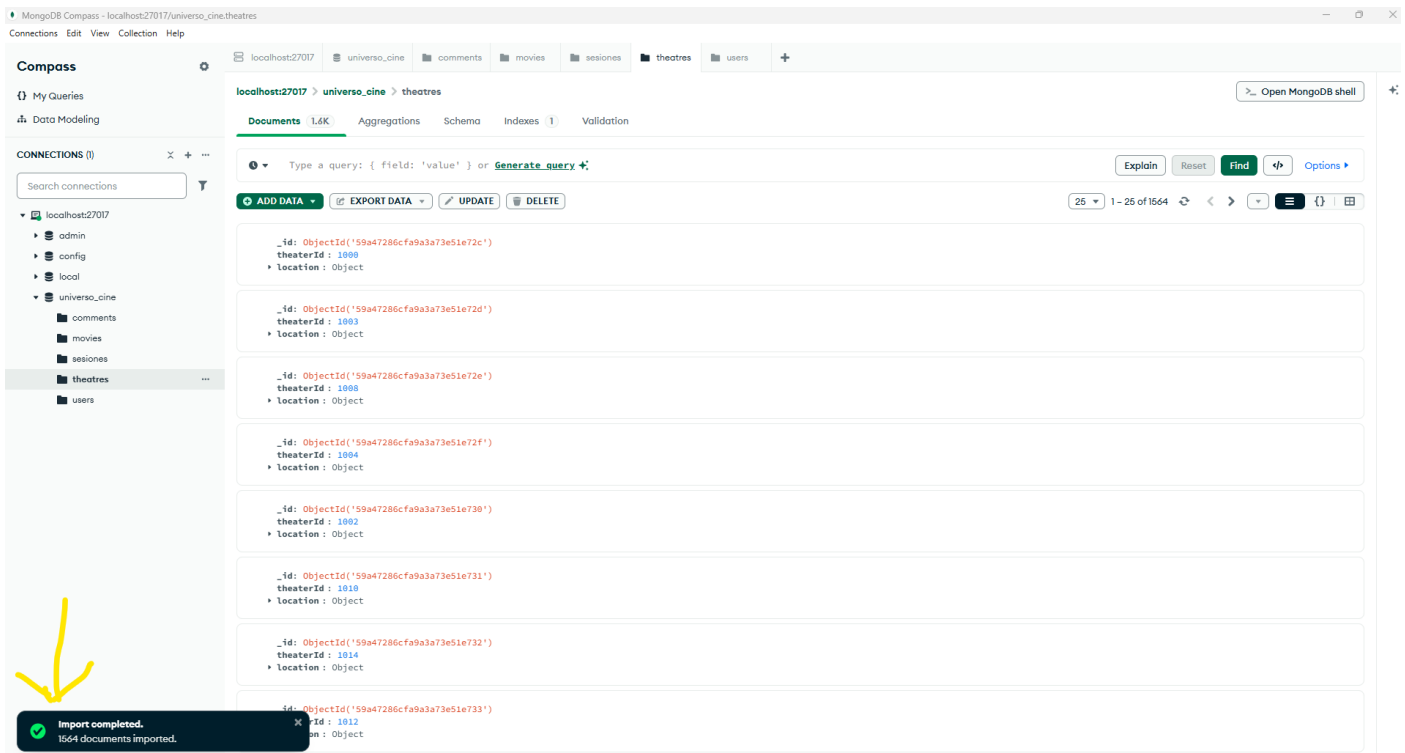
Y una vez pulsado el botón “Import” veo que se han importado todos los datos correctamente porque no hay ningún mensaje de error y por el mensaje en pantalla y también porque son 185 documentos que coincide con el nº de líneas del fichero users.json que se ve si se edita con un text.edit o con Visual Studio Code:



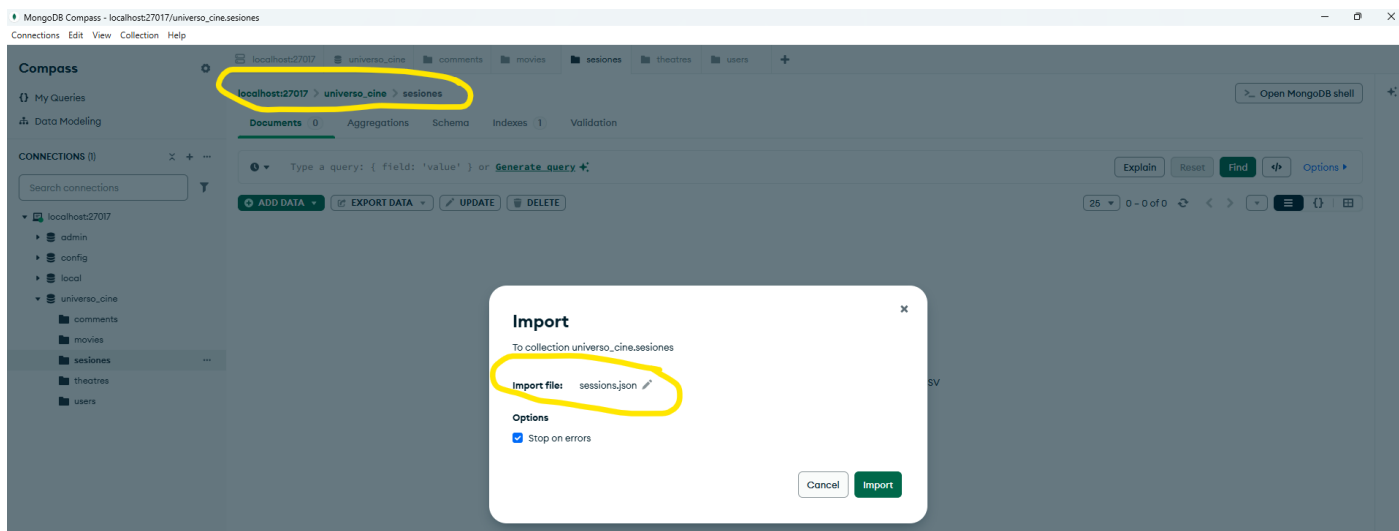
6. Repito los pasos del punto 5 anterior ahora para la colección “theatres”:



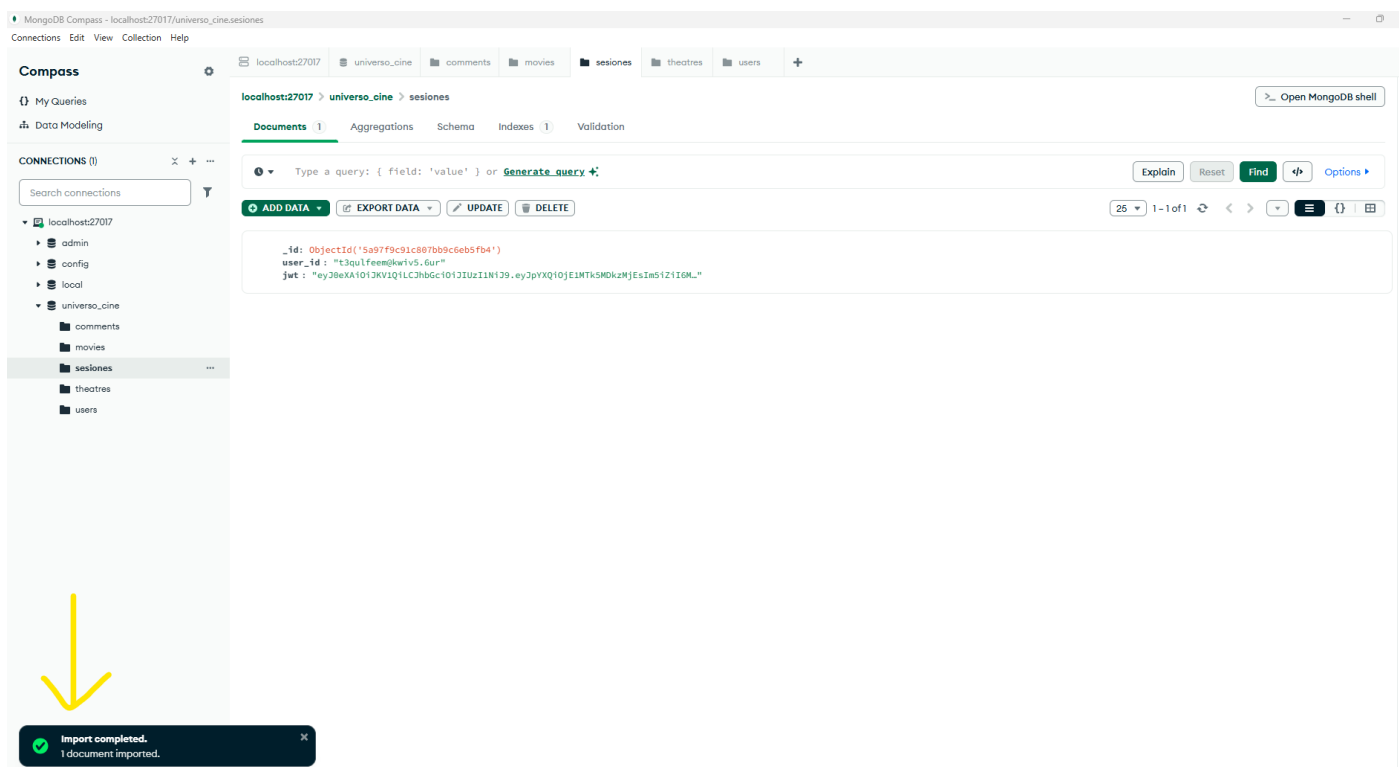
Y una vez pulsado el botón “Import” veo que se han importado todos los datos correctamente porque no hay ningún mensaje de error y por el mensaje en pantalla y también porque son 1564 documentos que coincide con el nº de líneas del fichero theatres.json que se ve si se edita con un text.edit o con Visual Studio Code:



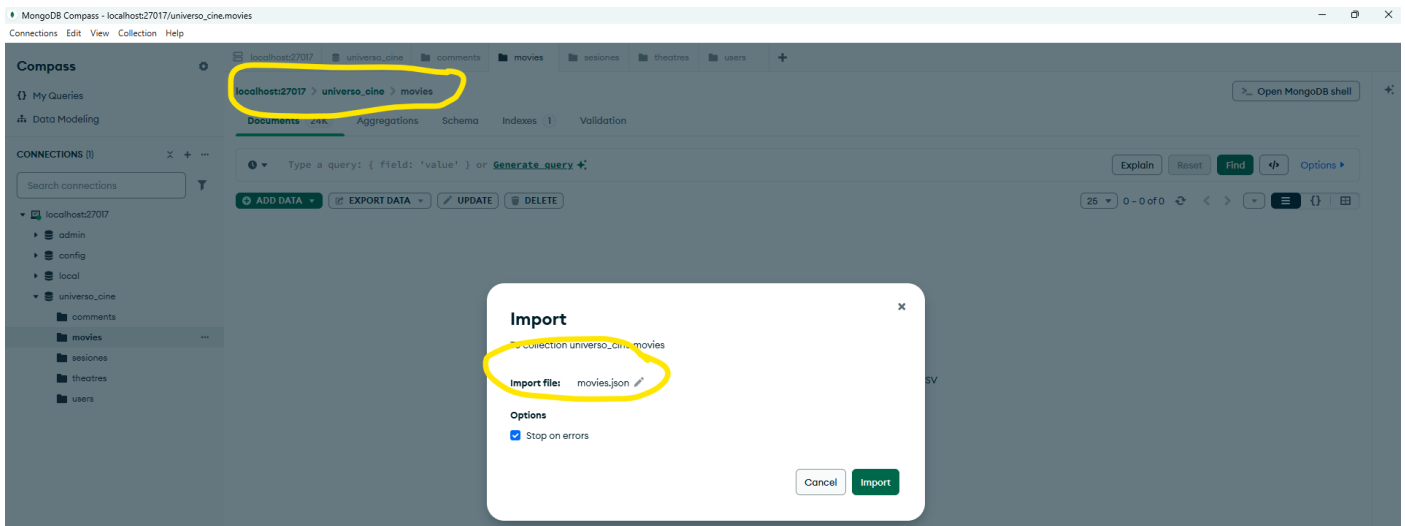
7. Repito los pasos del punto 5 anterior ahora para la colección “sesiones”:



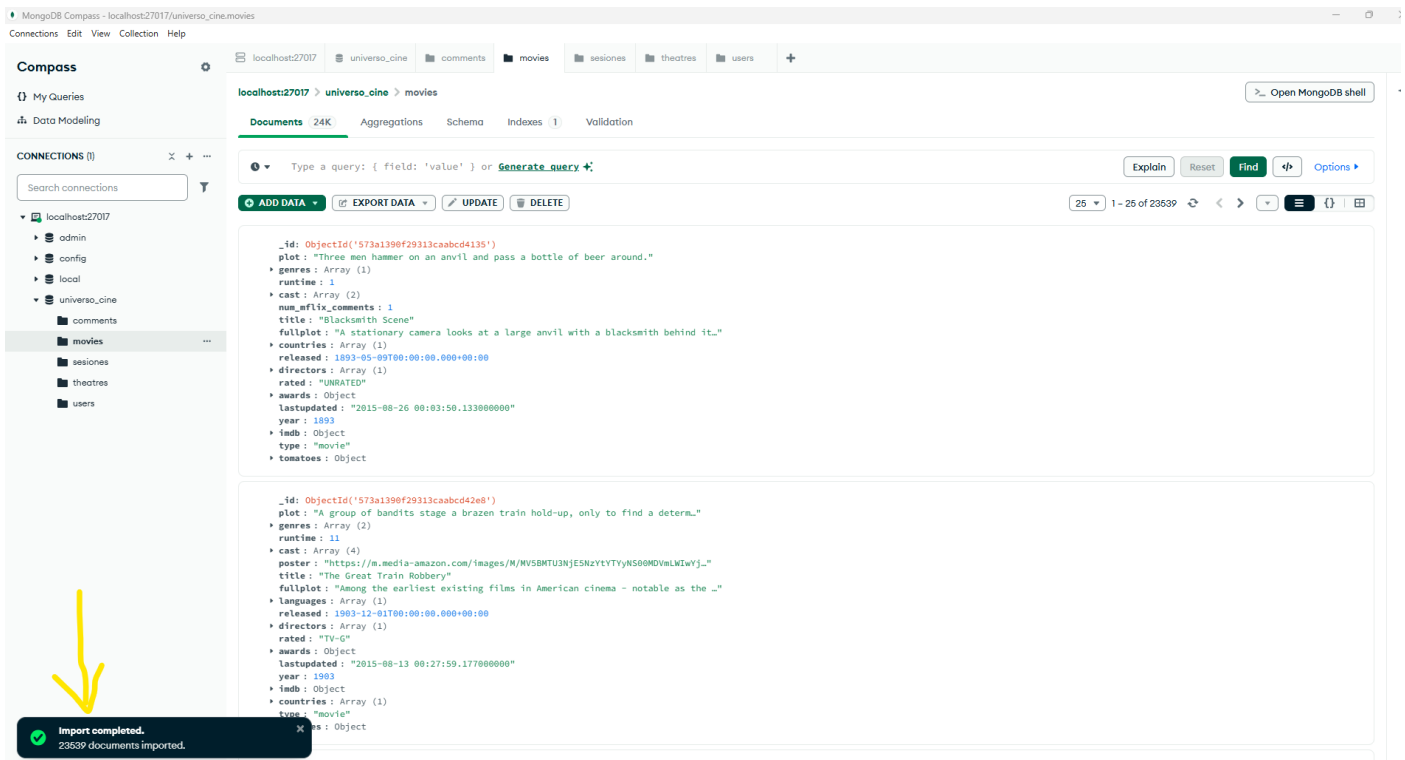
Y una vez pulsado el botón “Import” veo que se han importado todos los datos correctamente porque no hay ningún mensaje de error y por el mensaje en pantalla y también porque son 1 documentos que coincide con el nº de líneas del fichero sesiones.json que se ve si se edita con un text.edit o con Visual Studio Code:



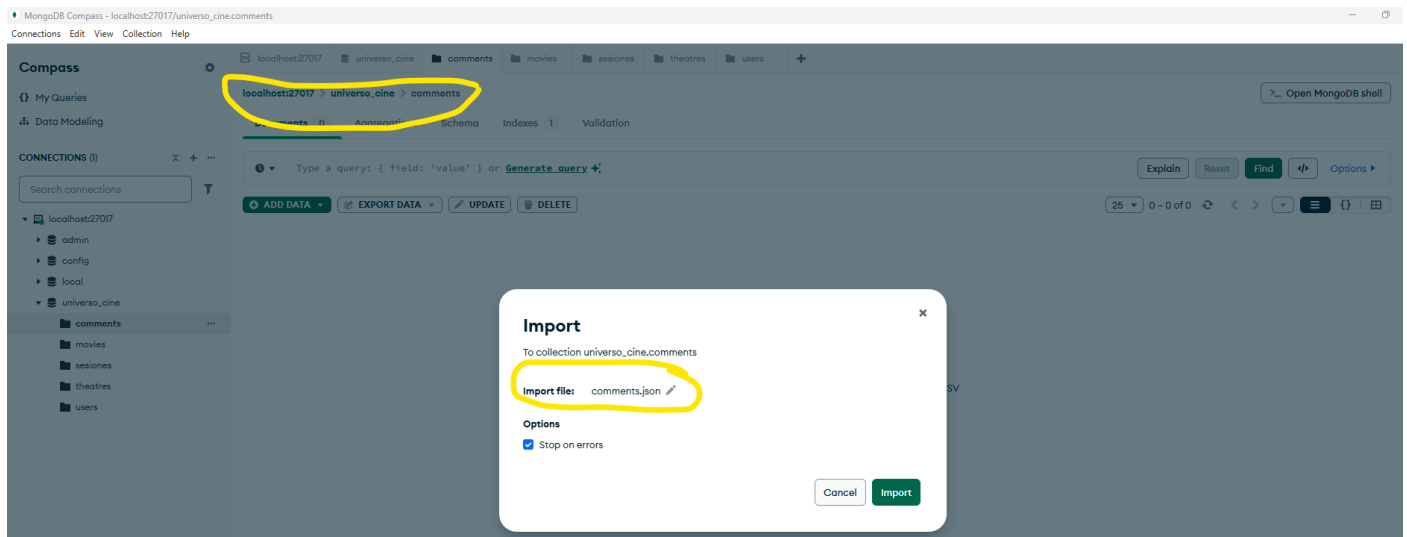
8. Repito los pasos del punto 5 anterior ahora para la colección “movies”:



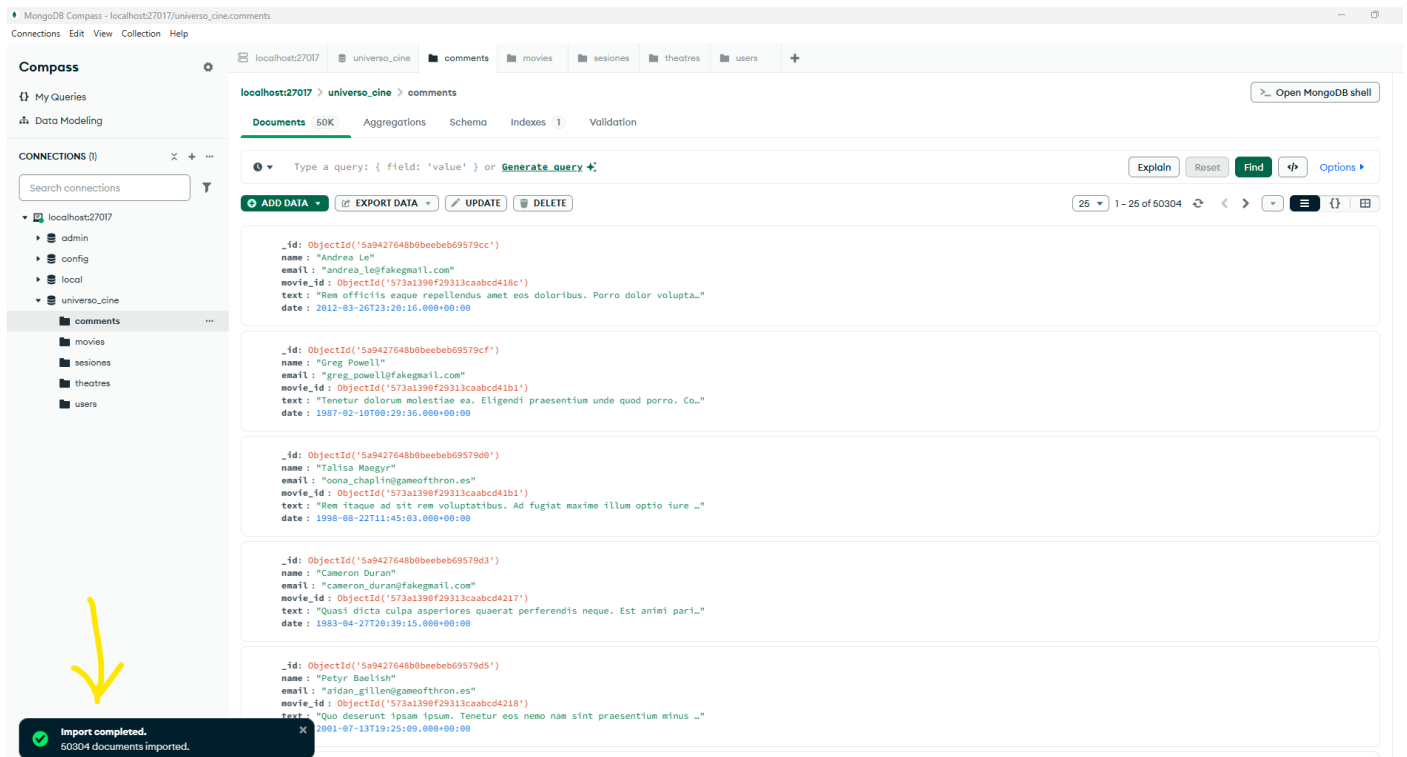
Y una vez pulsado el botón “Import” veo que se han importado todos los datos correctamente porque no hay ningún mensaje de error y por el mensaje en pantalla y también porque son 23.539 documentos que coincide con el nº de líneas del fichero movies.json que se ve si se edita con un text.edit o con Visual Studio Code:



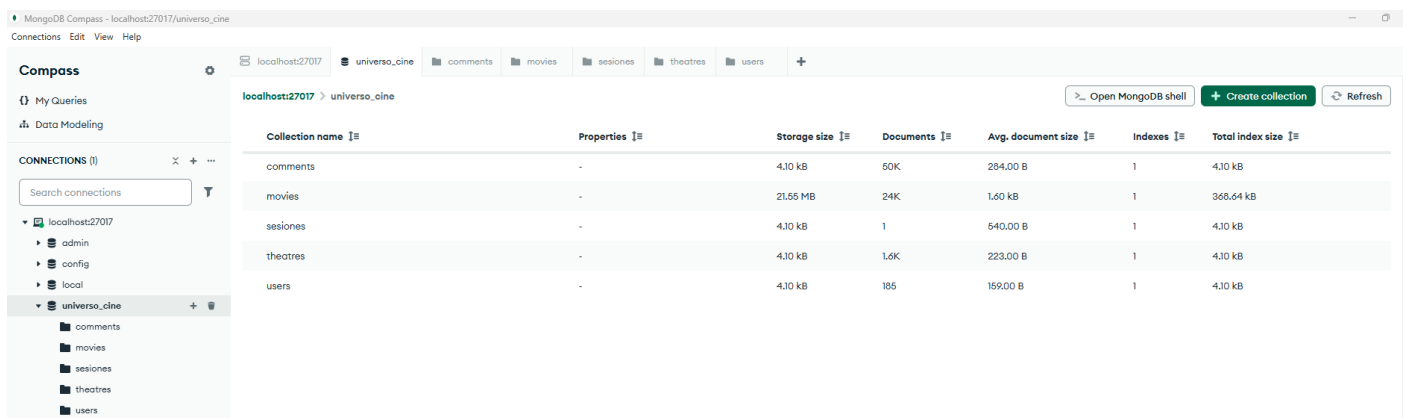
9. Repito los pasos del punto 5 anterior ahora para la colección “comments”:



Y una vez pulsado el botón “Import” veo que se han importado todos los datos correctamente porque no hay ningún mensaje de error y por el mensaje en pantalla y también porque son 50.304 documentos que coincide con el nº de líneas del fichero comments.json que se ve si se edita con un text.edit o con Visual Studio Code:



10. Por último, visualizo el resumen de la base de datos con sus colecciones y sus informaciones:

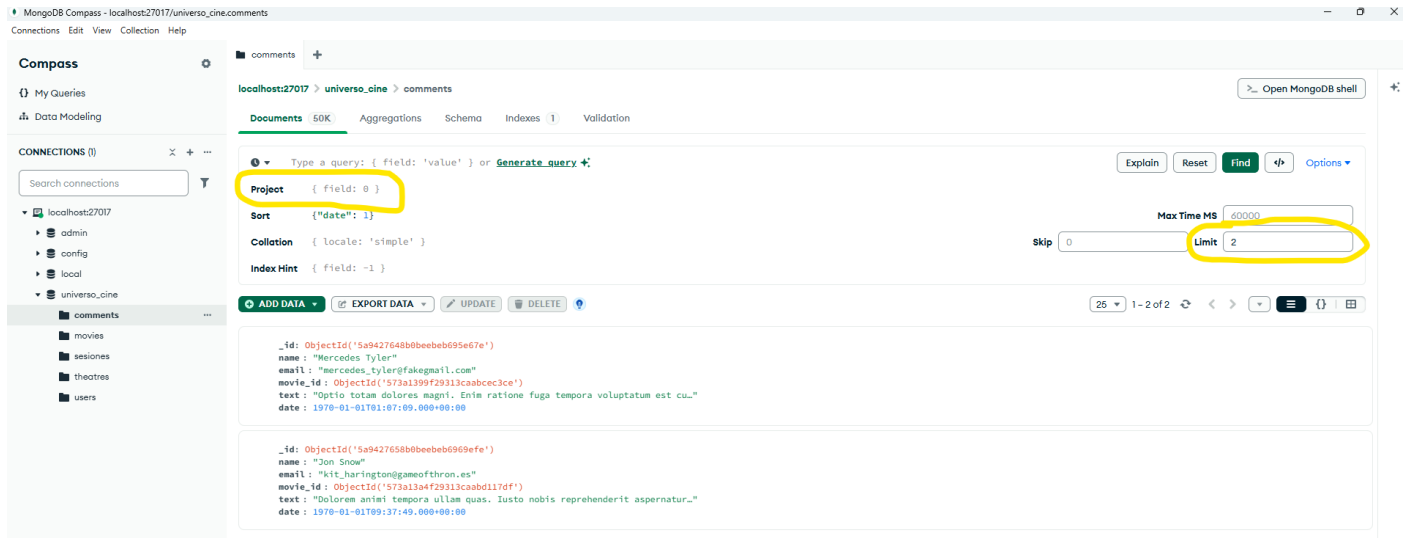


Ejercicio 1

1.1 Muestra los 2 primeros comentarios que aparecen en la base de datos.

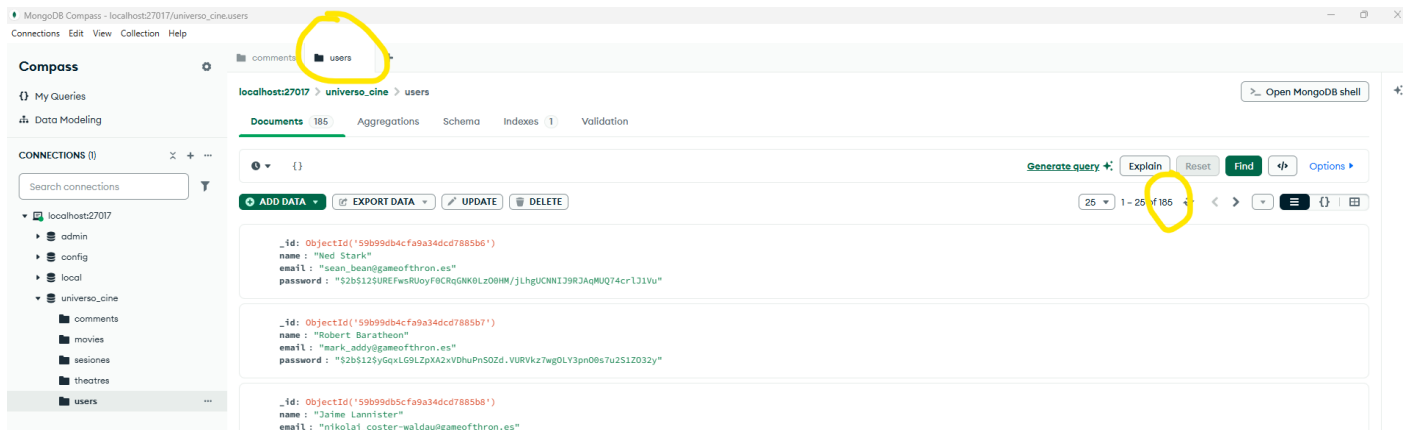
Entro en la colección “comments”, amplio los campos para hacer el filtro pulsando “Options” y pongo:

- Sort {“date”: 1} para que me ordene los documentos por fecha ascendente
- Limite a 2 para que me visualice solo los 2 primeros comentarios y le doy a Find



1.2 ¿Cuántos usuarios tenemos registrados?

Entro en la colección “users” y filtro por la query {} que me visualiza todos los documentos (hay unos por cada usuario) y arriba a la derecha, al lado de la visualización de documentos por página, puedo ver el número total de documentos que son 185



También lo puedo hacer en la pestaña “Aggregations”, con un nuevo stage usando \$count, así:

The screenshot shows the MongoDB Compass interface for the 'users' collection. The aggregation pipeline is defined as follows:

```

1 // **
2 * Provide the field name for the count.
3 */
4 { '$count': '$total_usuarios' }

```

The output preview shows a single document: `{ "total_usuarios": 185 }`. The value 185 is highlighted with a yellow circle.

1.3 ¿Cuántos cines existen en el estado de California?

Entro en la colección “theatres” y construyo el filtro usando la notación de puntos para ir entrando en cada documento hasta llegar al campo city, que buscare los documentos en que la city coincida con “California”. Arriba a la derecha, al lado de la visualización de documentos por página, puedo ver el número total de documentos que son 1

The screenshot shows the MongoDB Compass interface for the 'theatres' collection. The query filter is defined as:

```

{ "location.address.city": "California" }

```

The output preview shows a single document with the following structure:

```

{
  "_id": ObjectId("59a47286cfa9a3a73e51e72d"),
  "theaterId": 1003,
  "location": {
    "address": {
      "street1": "45235 Worth Ave.",
      "city": "California",
      "state": "MD",
      "zipcode": "20619"
    },
    "geo": {
      "type": "Point",
      "coordinates": [
        -76.512016,
        38.29697
      ]
    }
  }
}

```

The pagination controls show 1 of 1 document, with the number 1 highlighted by a yellow circle.

1.4 ¿Cuál fue el primer usuario en registrarse?

Entro en la colección “users” y veo que en los documentos, no hay un campo de fecha de creación o registro, así que voy a asumir que el campo “_id” se creó automáticamente y ascendente, con lo que, en el filtro, ordenaré ascendente todos los documentos por este campo y limitare la visualización a 1 para ver el primero

MongoDB Compass - localhost:27017/universo_cine.users

Connections Edit View Collection Help

Compass

My Queries

Data Modeling

CONNECTIONS (1)

localhost:27017

- admin
- config
- local
- universo_cine
 - comments
 - movies
 - sesiones
 - theatres
 - users

localhost:27017 > universo_cine > users

Documents 185 Aggregations Schema Indexes 1 Validation

Project { field: 0 }

Sort {"_id": 1}

Collation { locale: 'simple' }

Index Hint { field: -1 }

Generate query Explain Reset Find Options

Max Time MS 60000

Skip 0 Limit 1

ADD DATA EXPORT DATA UPDATE DELETE

25 1 - 1 of 1

`{ "_id": ObjectId("59b99db4cfa9a34cd7885b6"), "name": "Ned Stark", "email": "sean_bean@gameofthron.es", "password": "$2b$12$UREFwsRuoyF0CRqGNK0Lz0BHM/jLhgUCNNI39RJAqMUQ74cr1J1Vu" }`

1.5 ¿Cuántas películas de comedia existen en nuestra base de datos?

Entro en la colección “movies” y veo que el campo donde esta el genero es el campo “genres”, que es del tipo vector o array y que una misma película puede tener varios géneros, con lo cual “Comedy” puede estar en posiciones del vector distintas en documentos diferentes. Como nos dice que este, independientemente de la posición, el filtro será simple, así { "genres": "Comedy" } y veo, arriba a la derecha, al lado de la visualización de documentos por página, que el número total de documentos o películas que tienen género “Comedy” es de 7.024

MongoDB Compass - localhost:27017/universo_cine.movies

Connections Edit View Collection Help

Compass

My Queries

Data Modeling

CONNECTIONS (1)

localhost:27017

- admin
- config
- local
- universo_cine
 - comments
 - movies
 - sesiones
 - theatres
 - users

localhost:27017 > universo_cine > movies

Documents 24K Aggregations Schema Indexes 1 Validation

{ "genres": "Comedy" }

Generate query Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1 - 25 of 7024

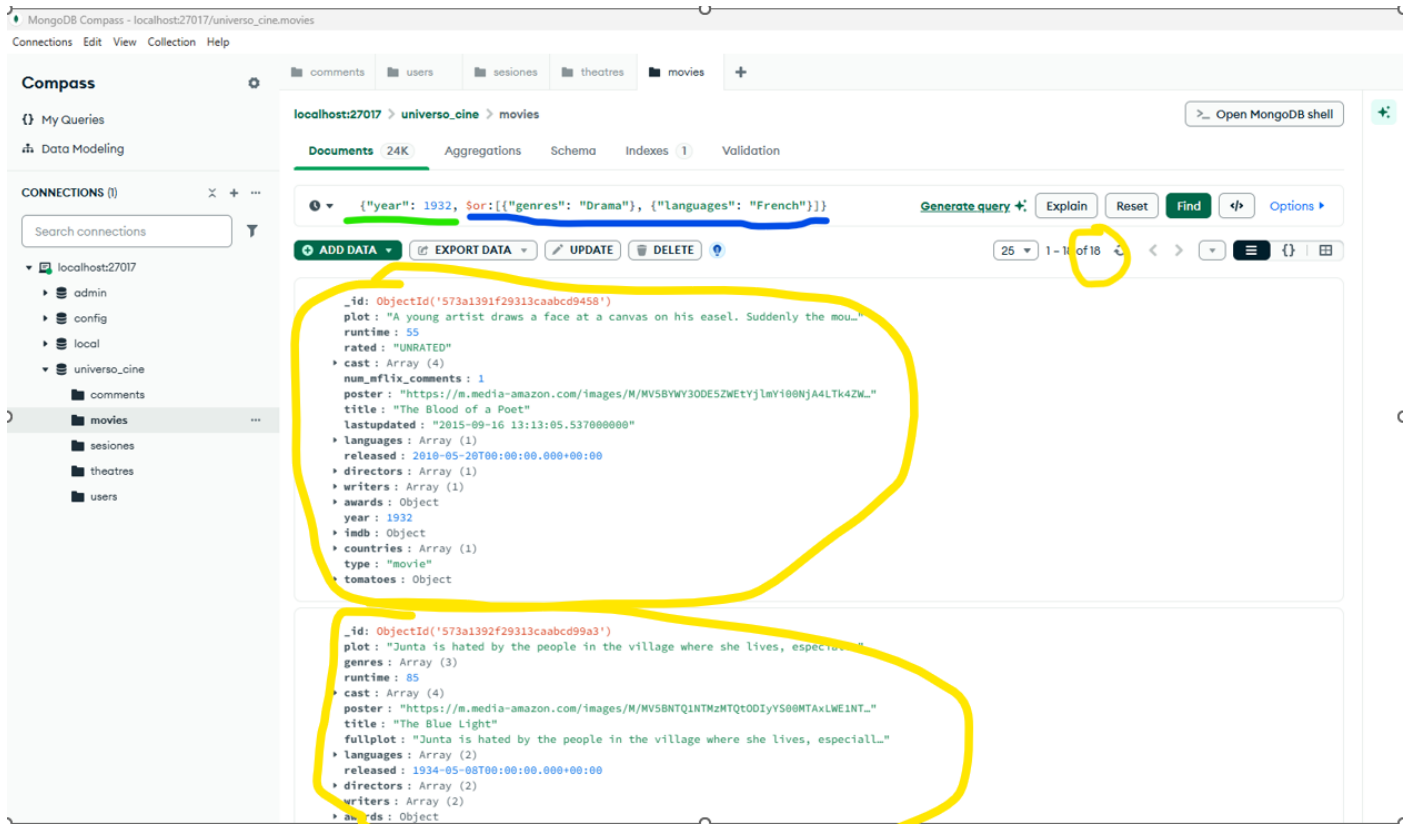
`{ "_id": ObjectId("573a1390f29313caabed50e5"), "plot": "The cartoonist, Winsor McCay, brings the Dinosaur back to life in th...", "genres": Array (3), "runtime": 12, "cast": Array (3), "num_mflix_comments": 1, "poster": "https://m.media-amazon.com/images/M/MV5BMTQxNzI4ODQ3NF5BM15BanBnXkFtZT...", "title": "Gertie the Dinosaur", "fullplot": "Winsor Z. McCay bets another cartoonist that he can animate a dinosaur...", "languages": Array (1), "released": "1914-09-15T00:00:00+00:00", "directors": Array (1), "writers": Array (1), "awards": Object, "lastupdated": "2015-08-18 01:03:15.313000000", "year": 1914, "imdb": Object, "countries": Array (1), "type": "movie", "tomatoes": Object }`

`{ "_id": ObjectId("573a1390f29313caabed50e4"), "plot": "Charlie is an immigrant who endures a challenging voyage and gets into...", "genres": Array (3), "runtime": 30, "cast": Array (4), "num_mflix_comments": 3, "poster": "https://m.media-amazon.com/images/M/MV5BMTNkYMU5YjMtMjY2My00MDI4LTl...", "title": "The Immigrant", "fullplot": "Charlie is on his way to the USA. He wins in a card game, puts the mon...", "languages": Array (1), "released": "1917-06-17T00:00:00+00:00", "directors": Array (1), "writers": Array (1), "awards": Object, "lastupdated": "2015-08-18 01:03:15.313000000", "year": 1917, "imdb": Object, "countries": Array (1), "type": "movie", "tomatoes": Object }`

Ejercicio 2

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

Entro en la colección “movies” en documents. Si pongo varias condiciones en el filtro son por defecto AND, pero en este caso tengo que usar también una OR al pedirme que sea Drama o en Frances, así que voy a tener que usar el operador \$or para esta segunda condición y añadirlo a la primera condición. Esto lo logro con el filtro: {"year": 1932, \$or:[{"genres": "Drama"}, {"languages": "French"}]} para obtener que son 18 películas las que cumplen las condiciones del ejercicio. En el pantallazo se muestran 2 de los 18 que se verían con scroll



También lo podría hacer creando un pipeline en Aggregations paso a paso de la siguiente manera (separo en 2 etapas distintas el and y el or para claridad, aunque se podría poner en una misma etapa):

localhost:27017 > universo_cine > movies

Open MongoDB shell

[Documents](#)
[24K](#)
[Aggregations](#)
[Schema](#)
[Indexes](#)
[1](#)
[Validation](#)

[\\$match](#)
[\\$match](#)
[\\$count](#)

Generate aggregation

Explain

Export

Run

Options

Untitled - modified

SAVE

+ CREATE NEW

EXPORT TO LANGUAGE

PREVIEW

STAGES

TEXT

WIZARD

23539 Documents in the collection

Preview of documents

```

_id: ObjectId('573a1390f29313caabcd4135')
plot: "Three men hammer on an anvil and
pass a bottle of beer around."
genres: Array (1)
runtime: 1
cast: Array (2)
num_mflix_comments: 1
title: "Blacksmith Scene"

```

```

_id: ObjectId('573a1390f29313caabcd42e8')
plot: "A group of bandits stage a brazen
train hold-up, only to find a
determ..."
genres: Array (2)
runtime: 11
cast: Array (4)
poster: "https://m.media-
amazon.com/images/M/MV5BNTU3NTE-

```

```

_id: ObjectId('573a1390f29313caabcd4323')
plot: "A young boy, oppressed by his mo
goes on an outing in the country
genres: Array (3)
runtime: 14
rated: "UNRATED"
cast: Array (4)
num_mflix_comments: 2

```

Stage 1 \$match

```

1 /**
2  * query: The query in MQL.
3  */
4  {
5    "year": 1932
6  }

```

Output preview after \$match stage (Sample of 10 documents)

```

_id: ObjectId('573a1391f29313caabcd9458')
plot: "A young artist draws a face at a
canvas on his easel. Suddenly the
mou..."
runtime: 55
rated: "UNRATED"
cast: Array (4)
num_mflix_comments: 1
poster: "https://m.media-

```

```

_id: ObjectId('573a1392f29313caabcd99a')
plot: "Junta is hated by the people in
village where she lives, especia
genres: Array (3)
runtime: 85
cast: Array (4)
poster: "https://m.media-
amazon.com/images/M/MV5BNTQ1NT
title: "The Blue Light"

```

Stage 2 \$match

```

1 /**
2  * query: The query in MQL.
3  */
4  {
5    $or:[{"genres": "Drama"}, {"languages": "French"}]
6  }

```

Output preview after \$match stage (Sample of 10 documents)

```

_id: ObjectId('573a1391f29313caabcd9458')
plot: "A young artist draws a face at a
canvas on his easel. Suddenly the
mou..."
runtime: 55
rated: "UNRATED"
cast: Array (4)
num_mflix_comments: 1
poster: "https://m.media-

```

```

_id: ObjectId('573a1392f29313caabcd99a')
plot: "Junta is hated by the people in
village where she lives, especia
genres: Array (3)
runtime: 85
cast: Array (4)
poster: "https://m.media-
amazon.com/images/M/MV5BNTQ1NT
title: "The Blue Light"

```

Stage 3 \$count

```

1 /**
2  * Provide the field name for the count.
3  */
4  'Numero peliculas resultante'

```

Output preview after \$count stage (Sample of 1 document)

Numero peliculas resultante: 18

+ Add stage

[Learn more about aggregation pipeline stages](#)

Ejercicio 3

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

Entro en la colección "movies" en documents. Como hay 2 condiciones que ponen límite mínimo y máximo, voy a utilizar los operadores mayor o igual (\$gte) y menor o igual (\$lte) y voy encadenado las 3 condiciones:

1. {"countries": "USA",
2. "awards.wins": {\$gte: 5, \$lte: 9},
3. "year": {\$gte: 2012, \$lte: 2014}}

Quedando el filtro final: {"countries": "USA", "awards.wins": {\$gte: 5, \$lte: 9}, "year": {\$gte: 2012, \$lte: 2014}}

Para obtener un total de 166 documentos. En el pantallazo se ven los 2 primeros documentos, para el resto hay que hacer scroll

The screenshot shows the MongoDB Compass interface. At the top, there's a navigation bar with tabs for 'comments', 'users', 'sesiones', 'theatres', and 'movies'. Below this, the breadcrumb path is 'localhost:27017 > universo_cine > movies'. The 'Documents' tab is selected, showing 24K documents. A query filter is applied: `es: "USA", "awards.wins": {$gte: 5, $lte: 9}, "year": {$gte: 2012, $lte: 2014}`. The interface shows two document results, each circled in yellow. The first document is for 'World War Z' (2013) and the second is for 'Godzilla' (2014). The document structure includes fields like 'genres', 'runtime', 'metacritic', 'rated', 'cast', 'poster', 'title', 'fullplot', 'languages', 'released', 'directors', 'writers', 'awards', 'lastupdated', 'year', 'imdb', 'countries', 'type', and 'tomatoes'.

```
{
  "genres": Array (3)
  "runtime": 116
  "metacritic": 63
  "rated": "PG-13"
  "cast": Array (4)
  "poster": "https://m.media-amazon.com/images/M/MV5BNQ4YzFmNzktMmM5ZC00MDZjLTk1OT..."
  "title": "World War Z"
  "fullplot": "Life for former United Nations investigator Gerry Lane and his family ..."
  "languages": Array (4)
  "released": 2013-06-21T00:00:00.000+00:00
  "directors": Array (1)
  "writers": Array (6)
  "awards": Object
  "lastupdated": "2015-09-10 17:37:02.647000000"
  "year": 2013
  "imdb": Object
  "countries": Array (2)
  "type": "movie"
  "tomatoes": Object
}
```

```
{
  "_id": ObjectId('573a13b9f29313caabd4f21f')
  "fullplot": "In 1999, the Janjira nuclear plant was mysteriously destroyed with mos..."
  "imdb": Object
  "year": 2014
  "plot": "The world is beset by the appearance of monstrous creatures, but one o..."
  "genres": Array (3)
  "rated": "PG-13"
  "metacritic": 62
  "title": "Godzilla"
}
```

También lo podría hacer creando un pipeline en Aggregations paso a paso de la siguiente manera

Documents 24K Aggregations Schema Indexes 1 Validation

[Generate aggregation](#) [Explain](#) [Export](#) [Run](#) [Options](#)

☒ PREVIEW
 ☒ STAGES
 ☐ TEXT
 ☐ WIZARD

Preview of documents

```
_id: ObjectId('573a1390f29313caabacd4323')
plot: "A young boy, oppressed by his mother, goes on an outing in the country"
genres: Array (3)
runtime: 14
rated: "UNRATED"
cast: Array (4)
num_mflix_comments: 2
```

```
1 ▾ /**
2   * query: The query in MQL.
3   */
4 ▾ {
5   "countries": "USA"
6 }
```

```
_id: ObjectId('573a1390f29313caabcd4135')
plot: "Three men hammer on an anvil and
      pass a bottle of beer around."
└─ genres: Array (1)
  runtime: 1
└─ cast: Array (2)
  num_mflix_comments: 1
  title: "Blacksmith Scene"
  fullplot: "A stationary camera looks at a
```

```
_id: ObjectId('573a1390f29313caabcd42ef')  
plot : "A group of bandits stage a brazilian hold-up, only to find a determ..."  
└─ genres: Array (2)  
    runtime: 11  
└─ cast : Array (4)  
    poster: "https://m.media-  
amazon.com/images/M/WV5BMTUxNj
```

```
1 ▾ /**
2   * query: The query in MQL.
3   */
4 ▾ {
5   "awards.wins": {$gte: 5, $lte:9}
6 }
```

```
_id: ObjectId('573a1391f29313caabcf727')
plot: "A prospector goes to the Klondike i
      search of gold and finds it and m..."
genres: Array (3)
runtime: 95
rated: "NOT RATED"
title: "The Gold Rush"
num_mflix_comments: 2
poster: "https://m.media-
```

```
_id: ObjectId('573a1391f29313caabcd88d6')
plot: "A street cleaner saves a young woman's life, and the pair slowly fall ..."
genres: Array (2)
runtime: 110
rated: "NOT RATED"
title: "7th Heaven"
num_mflix_comments: 1
```

```
1 ▾ /**
2   * query: The query in MQL.
3   */
4 ▾ {
5   "year": {$gte: 2012, $lte: 2014}
6 }
```

```
_id: ObjectId('573a13acf29313caabd29366')
fullplot: "The manager of the negative  
assets sector of Life magazine,  
Walter Mit..."
imdb: Object
year: 2013
plot: "When his job along with that of his  
co-worker are threatened, Walter t..."
genres: Array (3)
```

```

_id: ObjectId('573a13b5f29313caabd5777')
plot: "After their cave is destroyed,
caveman family must trek through
u..."
> genres: Array (3)
runtime: 98
metacritic: 55
rated: "PG"
> cast: Array (4)

```

Stage 4 \$count

```
1 /**
2  * Provide the field name for the count.
3  */
4  'Total_documentos'
```

Output preview after \$count stage (Sample of 1 document)

Total_documentos : 166

+ Add stage

[Learn more about aggregation pipeline stages](#)

NIVEL 2

Ejercicio 1

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

Para este ejercicio voy a utilizar la coincidencia por subcadena utilizando el operador \$regex para buscar todos los documentos en los que el valor de incluya el término "gameofthron.es" y como me pide los comentarios de unos de ellos, luego los voy a agrupar por el campo "email" que es único para cada usuario y sumando 1 con el operador \$sum:1 por cada vez que aparecen. Las agrupaciones y conteos por grupo solo se pueden hacer en aggregations.

Primero con el Stage 1 filtro todos los que tienen el mismo dominio "@gameofthron.es" :

```
{
  "_id": ObjectId('5a9427648b0beebe69579cc'),
  "name": "Andrea Le",
  "email": "andrea_le@fakegmail.com",
  "movie_id": ObjectId('573a1390f29313caabcd41...'),
  "text": "Rem officiiis eaque repellendus amet eos doloribus. Porro dolor volupta...",
  "date": 2012-03-26T23:20:16.000+00:00
}
```

```
{
  "_id": ObjectId('5a9427648b0beebe69579cf'),
  "name": "Greg Powell",
  "email": "greg_powell@fakegmail.com",
  "movie_id": ObjectId('573a1390f29313caabcd41...'),
  "text": "Tenetur dolorum molestiae ea. Eligendi praesentium unde quod porro. Co...",
  "date": 1987-02-10T00:29:36.000+00:00
}
```

```
{
  "_id": ObjectId('5a9427648b0beebe69579d0'),
  "name": "Talisa Maegyr",
  "email": "oona_chaplin@gameofthron.es",
  "movie_id": ObjectId('573a1390f29313caabcd41...'),
  "text": "Rem itaque ad sit rem voluptatib. Ad fugiat maxime illum optio iure...",
  "date": 1998-08-22T11:45:03.000+00:00
}
```

```
1 /**
2  * query: The query in MQL.
3  */
4  {
5    "email": { $regex: "@gameofthron.es" }
6  }
```

```
{
  "_id": ObjectId('5a9427648b0beebe69579d0'),
  "name": "Talisa Maegyr",
  "email": "oona_chaplin@gameofthron.es",
  "movie_id": ObjectId('573a1390f29313caabcd41b...'),
  "text": "Rem itaque ad sit rem voluptatibus. Ad fugiat maxime illum optio iure...",
  "date": 1998-08-22T11:45:03.000+00:00
}
```

```
{
  "_id": ObjectId('5a9427648b0beebe69579d1'),
  "name": "Petyr Baelish",
  "email": "aidan_gillen@gameofthron.es",
  "movie_id": ObjectId('573a1390f29313caabcd41b...'),
  "text": "Quo deserunt ipsam ipsum. Tenet nemo nam sint praesentium minus",
  "date": 2001-07-13T19:25:09.000+00:00
}
```

Y con el stage 2 los agrupo por el campo "email" sumando 1 cada vez que aparecen. Aquí ya veo por ejemplo que el usuario con email [conleth_hill@gameofthron.es](#) ha hecho un total de 271 comentarios:

```
1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5  {
6    "_id": "$email",
7    "Total_Comentarios": {
8      $sum: 1
9    }
10 }
```

```
{
  "_id": "conleth_hill@gameofthron.es",
  "Total_Comentarios": 271
}
```

```
{
  "_id": "hafthór_júlíus_björnsson@gameofthron.es",
  "Total_Comentarios": 285
}
```

Ejercicio 2

¿Cuántos cines existen en cada código postal situados dentro del estado Washington DC (DC)?

Como hay que contar por la segmentación de código postal, lo tengo que hacer por aggregations.

Con el Stage 1 filtro los documentos que tienen "state" igual a "DC" que son 3

theatres

comments users sesiones theatres movies +

localhost:27017 > universo_cine > theatres Open MongoDB shell

Documents 1.6K Aggregations Schema Indexes 1 Validation

\$match \$group Generate aggregation Explain Export Run Options

Untitled - modified SAVE CREATE NEW EXPORT TO LANGUAGE PREVIEW STAGES TEXT WIZARD

1564 Documents in the collection

Preview of documents

```
{ "_id": ObjectId("59a47286cfa9a3a73e51e72c"), "theaterId": 1000, "location": {} }
```

```
{ "_id": ObjectId("59a47286cfa9a3a73e51e72d"), "theaterId": 1003, "location": {} }
```

```
{ "_id": ObjectId("59a47286cfa9a3a73e51e72e"), "theaterId": 1008, "location": {} }
```

Stage 1 \$match ON

```
1 /**
2  * query: The query in MQL.
3  */
4
5 {
6   "location.address.state": "DC"
7 }
```

Output preview after \$match stage (Sample of 3 documents)

```
{ "_id": ObjectId("59a47286cfa9a3a73e51e785"), "theaterId": 1092, "location": {} }
```

```
{ "_id": ObjectId("59a47287cfa9a3a73e51ec3"), "theaterId": 801, "location": {} }
```

Y con el stage 2 los grupo por su "zipcode" y sumo 1 por cada vez que aparecen:

Stage 2 \$group ON

```
1 /**
2  * _id: The id of the group.
3  * fieldN: The first field name.
4  */
5
6 {
7   _id: "$location.address.zipcode",
8   numero_cines: {
9     $sum: 1
10  }
11 }
```

Output preview after \$group stage (Sample of 3 documents)

```
{ "_id": "20002", "numero_cines": 1 }
```

```
{ "_id": "20010", "numero_cines": 1 }
```

+ Add stage

[Learn more about aggregation pipeline stages](#)

Y si expando el resultado se visualiza mejor que hay 1 cine en cada uno de los 3 códigos postales:

Stage 2: \$group **ENABLED** **ADD STAGE**

STAGE INPUT
Sample of 3 documents

```

1 {
2   _id: ObjectId('59a47286cfa9a3a73e51e785')
3   theaterId: 1092
4   location: Object
5     address: Object
6       street1: "3100 14th St Nw"
7       city: "Washington"
8       state: "DC"
9       zipcode: "20019"
10    geo: Object
11      type: "Point"
12      coordinates: Array (2)
13        0: -77.034
14        1: 38.929
15    }
16  }

```

STAGE OUTPUT
Sample of 3 documents

```

1 {
2   _id: "20002"
3   numero_cines: 1
4 }
5 {
6   _id: "20010"
7   numero_cines: 1
8 }
9 {
10  _id: "20016"
11  numero_cines: 1
12 }

```

NIVEL 3

Ejercicio 1

Encuentra todas las películas dirigidas por John Landis con una puntuación IMDb (Internet Movie Database) de entre 7,5 y 8.

Entro en la colección “movies” en documents. Como hay 2 condiciones que ponen en el rating, límite mínimo (7.5) y máximo (8), voy a utilizar los operadores mayor o igual (\$gte) y menor o igual (\$lte), ya que entiendo que si una película tiene el valor de los limites también hay que contarla (el 7.5 y el 8)

El filtro sería: {"directors": "John Landis", "imdb.rating": {\$gte: 7.5, \$lte:8}}

Y voy a visualizar solo el id, el titulo y el rating, para eso en el campo de filtrado Project pongo: {"_id": 1, "title": 1, "imdb.rating": 1 }

movies +

localhost:27017 > universo_cine > movies Open MongoDB shell

Documents 24K Aggregations Schema Indexes 1 Validation

{ "directors": "John Landis", "imdb.rating": { \$gte: 7.5, \$lte: 8 } }
 Generate query + Explain Reset Find </> Options ▾

Project { "_id": 1, "title": 1, "imdb.rating": 1, "directors": 1 }
 Sort { field: -1 } or [['field', -1]]
 Collation { locale: 'simple' }
 Index Hint { field: -1 }

Max Time MS 60000
 Skip 0 Limit 0

EXPORT DATA 25 1-4 of 4

- _id: ObjectId('573a1397f29313caabce6d94')
 imdb: Object
 rating: 7.6
 title: "Animal House"
 directors: Array (1)
 0: "John Landis"
- _id: ObjectId('573a1397f29313caabce76f7')
 title: "The Blues Brothers"
 directors: Array (1)
 0: "John Landis"
 imdb: Object
 rating: 7.9
- _id: ObjectId('573a1397f29313caabce7d96')
 imdb: Object
 rating: 7.6
 title: "An American Werewolf in London"
 directors: Array (1)
 0: "John Landis"
- _id: ObjectId('573a1398f29313caabce8deb')
 title: "Trading Places"
 directors: Array (1)
 0: "John Landis"
 imdb: Object
 rating: 7.5

Ejercicio 2

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

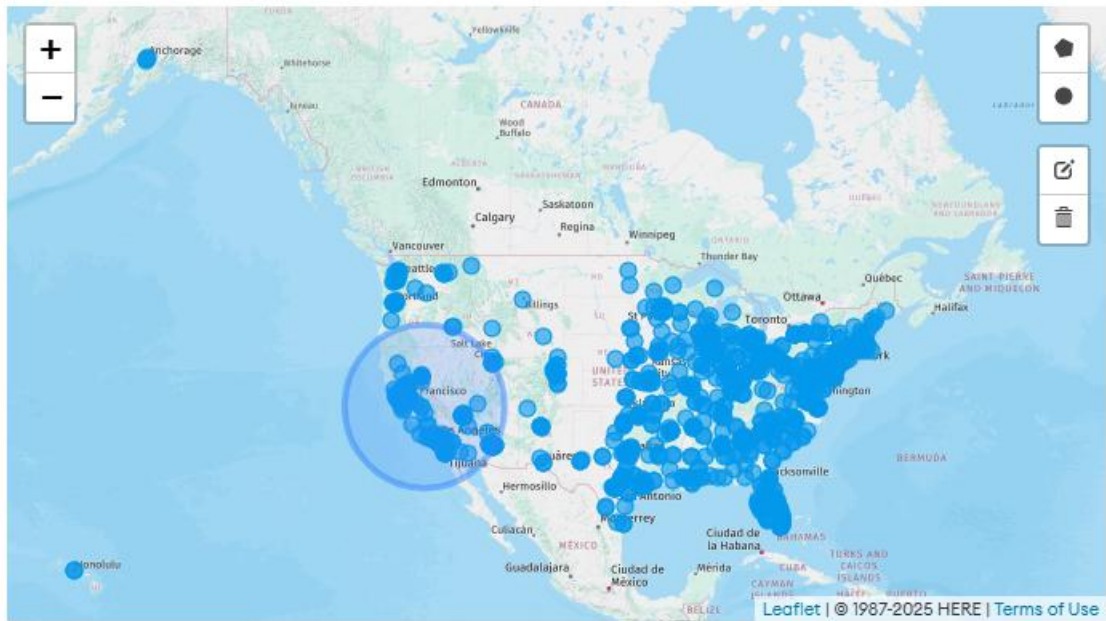
Entro en la colección “movies” en la pestaña “Schema” y le doy al botón de analizar. Una vez termina, en los resultados expando el apartado geo y ahí ya tengo el mapa:

geo

coordinates

Leaflet | © 1987-2025 HERE | Terms of Use

Y si quisiera buscar una zona mas concreta dentro del mapa, hay que usar la opción del punto del mapa para determinar la zona que me gustaría expandir haciendo un círculo y volviendo a dar a el botón de “Analyze”, por ejemplo:



Y después de dar al botón Analyze obtengo:

