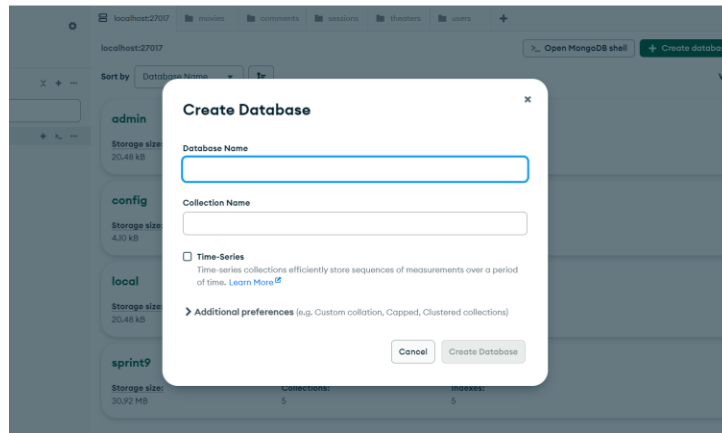


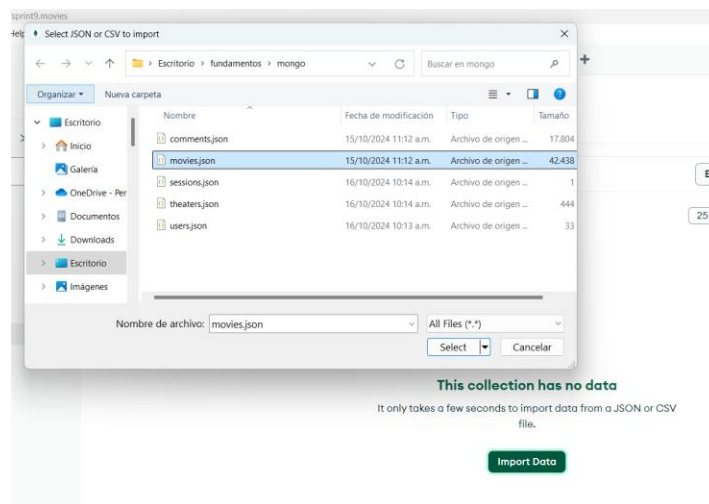
SPRINT 9

NIVEL 1

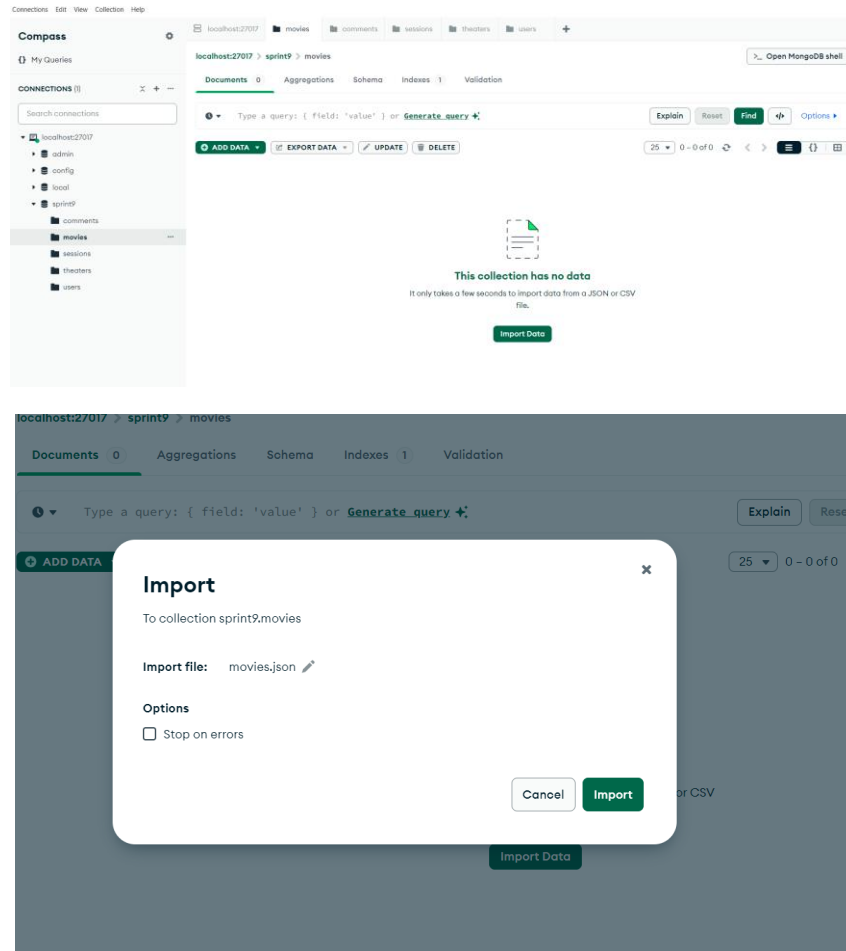
Crear una base de datos MongoDB con los archivos adjuntos



Luego de crear mi base de datos Sprint9 le di los nombre de cada archivo adjunto en **Colecction Name**



Al tener los archivos en nuestra base de datos, importamos la data para cada uno de ellos, en el import data

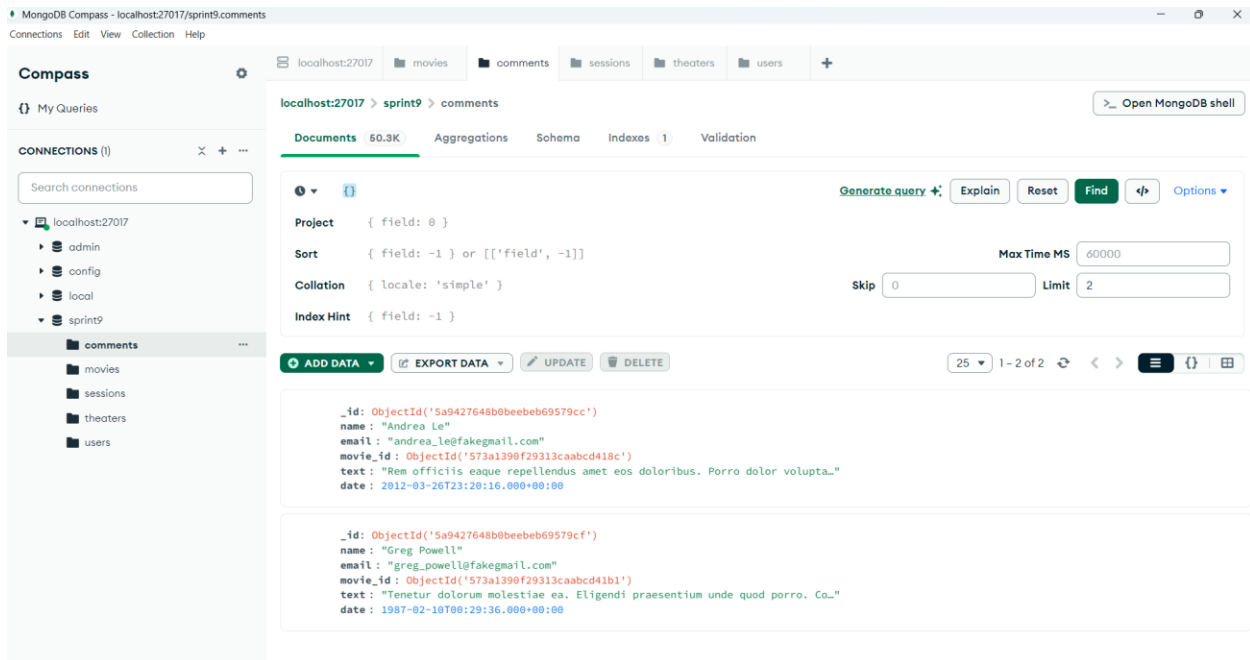


Ejercicio 1

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

Para este ejercicio, seleccioné la colección **comments**, luego a la pestaña **Documents**, obtengo todos los comentarios sin ningún filtro, dejo el campo de búsqueda vacío {}.

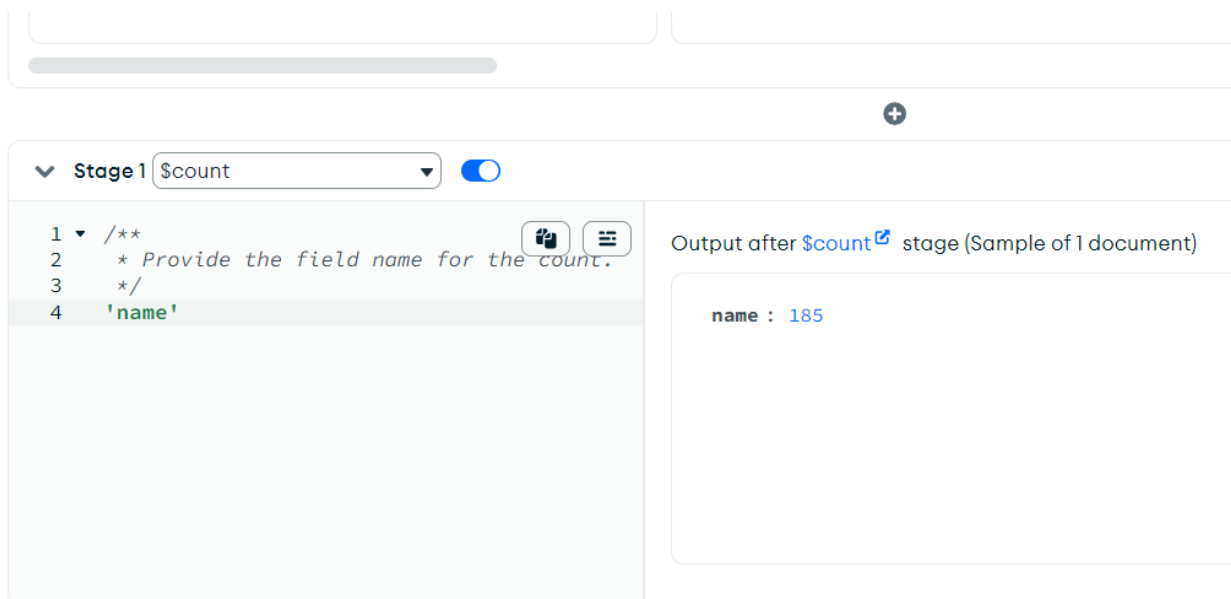
Se busca el campo que dice **"Limit"** en la parte inferior de la ventana de documentos, ingreso el numero 2 para los dos comentarios, ejecuto la consulta en el boton **"Find"**



¿Cuántos usuarios tenemos registrados?

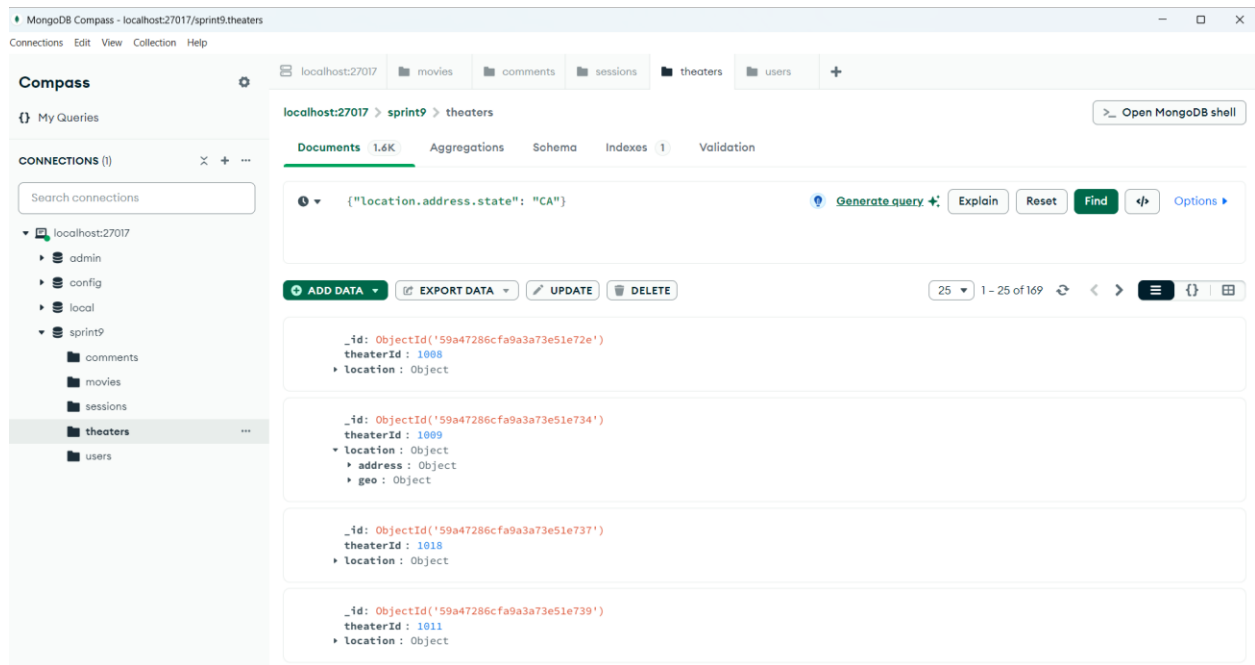
Para ello se ha utilizado la ventana **Aggregation** y se ha utilizado el stage **\$count** y el field 'name' de la colección users para realizar el recuento

185 usuarios



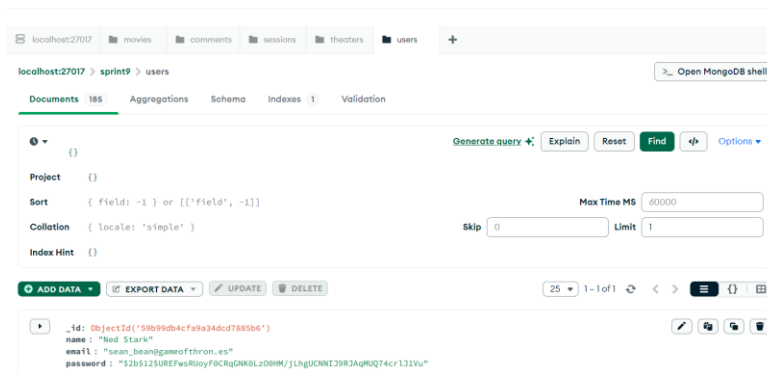
Cuántos cines hay en el estado de California?

Se han encontrado 169 cines en california



Quien fue el primer usuario/aria al registrarse?

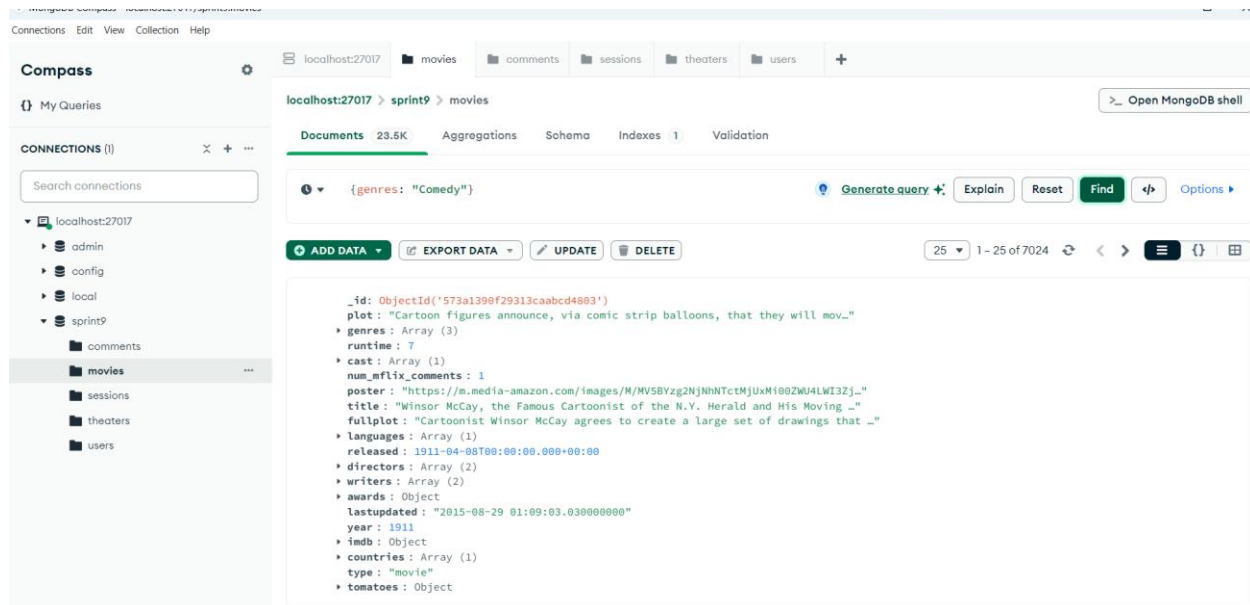
Seleccione **users** , me coloco en **Documents**, luego **options** y coloco **Limit 1**



¿Cuántas películas de comedia hay en nuestra base de datos?

Para este ejercicio selecciono **movies**, y utilice solo la query

{genres:"Comedy"} son 7024

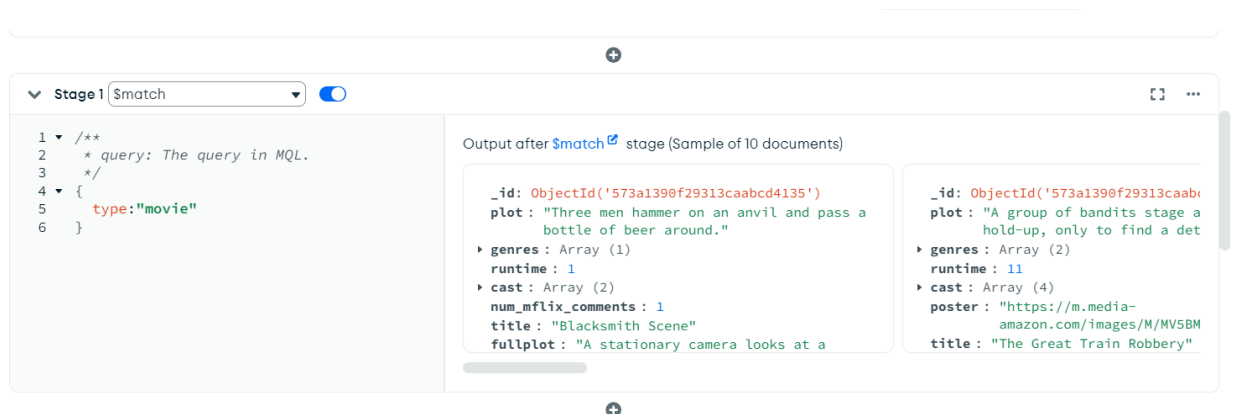


Ejercicio 2

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


Para este ejercicio realice una **Aggregation** para el primer stage utilice el **metodo \$match** colocando en el query

type:"movie"



En el segundo stage el año

year:1932



Stage 2 \$match

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

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

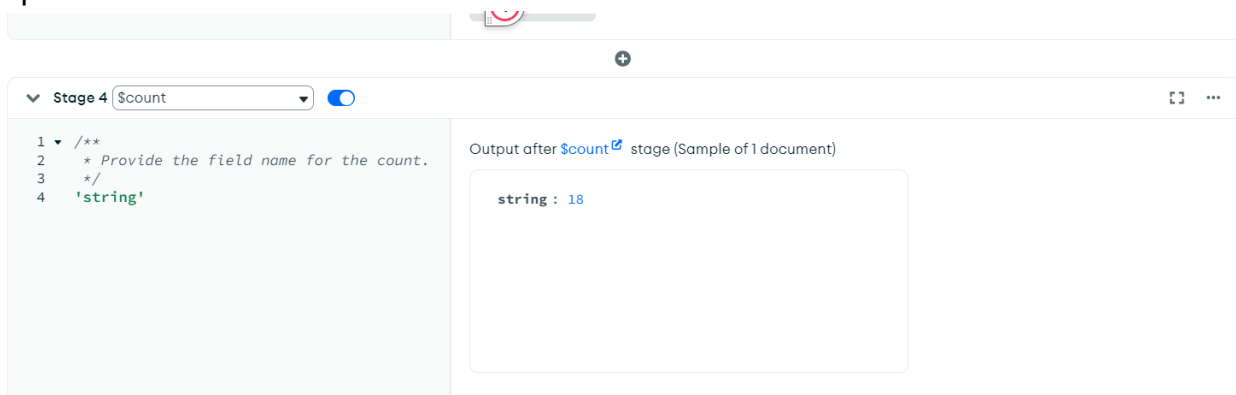
```
released: 2010-05-20T00:00:00.000+00:00
directors: Array (1)
writers: Array (1)
awards: Object
year: 1932
imdb: Object
countries: Array (1)
type: "movie"
tomatoes: Object

_id: ObjectId('573a1392f29313caab')
plot: "Junta is hated by the peop
village where she lives, es
genres: Array (3)
runtime: 85
cast: Array (4)
poster: "https://m.media-
amazon.com/images/M/MV5BN
title: "The Blue Light"
```

Un tercer stage con el ultimo **metodo \$match con lo siguiente**

```
{
  $or:[{genres:"drama"},{languages:"French"}]
}
```

Y por ultimo un contador



Stage 4 \$count

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

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

```
string: 18
```

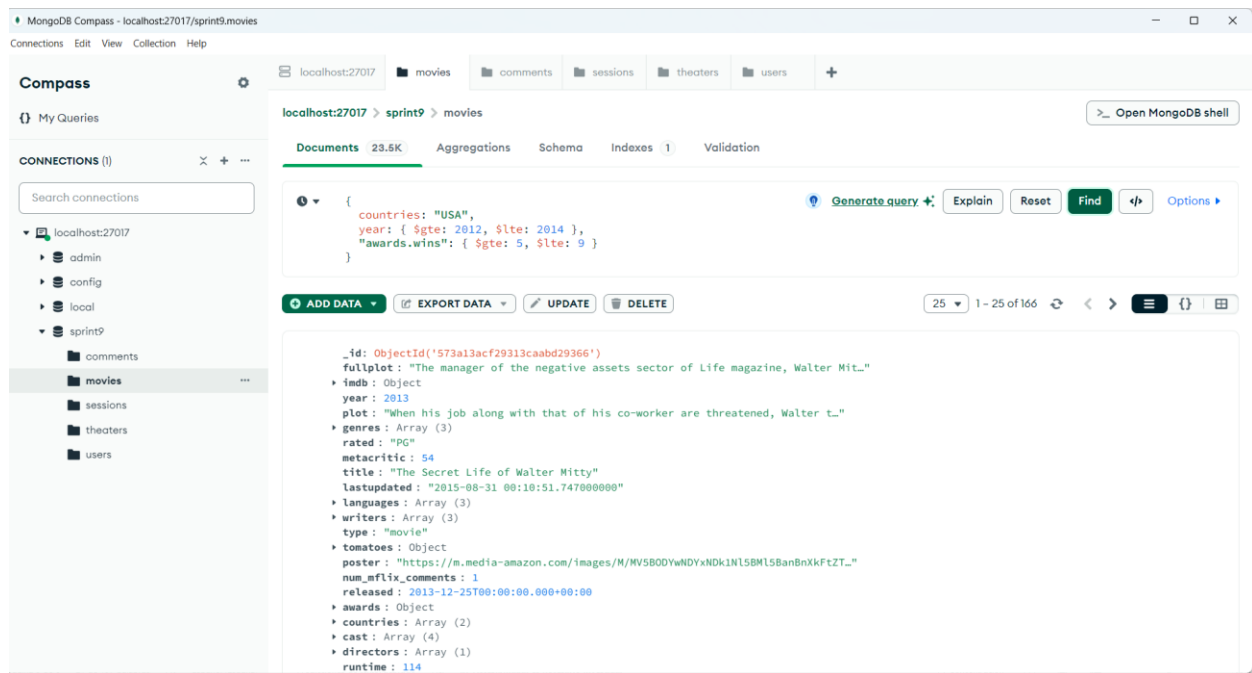
Me da un total de 18

Ejercicio 3

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

La query es la siguiente:

```
{ countries: "USA",
  year: { $gte: 2012, $lte: 2014 },
  "awards.wins": { $gte: 5, $lte: 9 }
}
```



Total de 166

Nivel 2

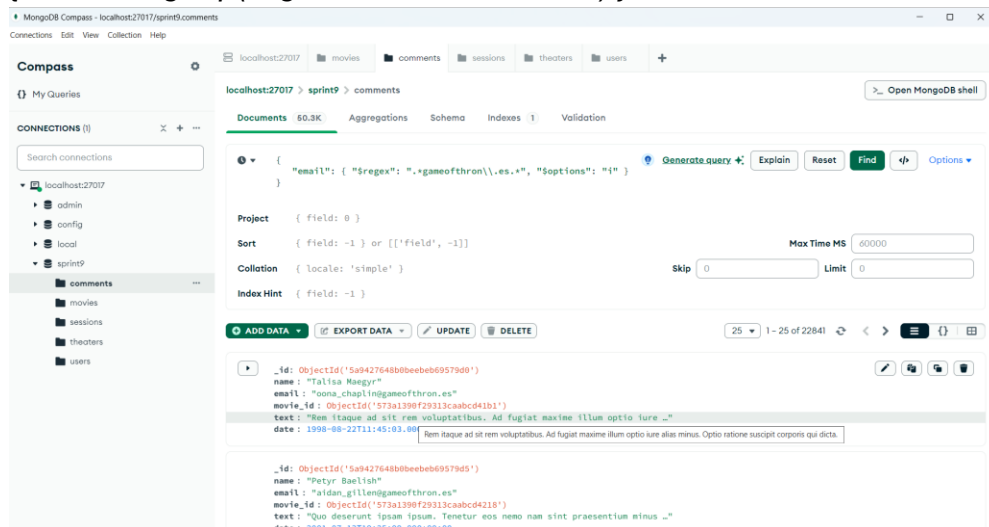
Ejercicio 1

Cuenta cuántos comentarios escribe un usuario/aria que utiliza "GAMEOFTHRON.ES"

Son 22841

La query fue la siguiente

```
{ email: RegExp(".*gameofthron\\.es.*", "i") }
```



Ejercicio 2

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

Para este ejercicio realice una **Aggregation** para el primer stage utilice el **metodo \$match** para filtrar aquellos cines en **DC**

```
"location.address.state": "DC"
```

En el segundo stage se ha utilizado el **metodo \$group**

```
{ "_id": "$location.address.zipcode", // Agrupamos por el código postal
```

```
"theater_count": { "$sum": 1 } // Contamos los cines por cada código postal
```

```
}
```


Stage 1

\$match

1

2

3

4

5

6

/**

* query: The query in MQL.

*/

{

"location.address.state": "DC"

}

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

_id: ObjectId('59a47286cfa9a3a73e51e785')

theaterId: 1092

location: Object

_id: ObjectId('59a47287cfa9a3a73e51e786')

theaterId: 801

location: Object

Stage 2

\$group

1

2

3

4

5

6

7

8

9

10

11

/**

* _id: The id of the group.

* fieldN: The first field name.

*/

{

"_id": "\$location.address.zipcode", //

"theater_count": { "\$sum": 1 } // Conta

}

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

_id: "20016"

theater_count : 1

_id: "20002"

theater_count : 1

ALL RESULTS OUTPUT OPTIONS

_id: "20016"

theater_count : 1

_id: "20010"

theater_count : 1

_id: "20002"

theater_count : 1

Un total de 3

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.

Para este ejercicio realice una **Aggregation** para el primer stage utilice el **metodo \$match** para filtrar el director Jonh Landis

Stage 1 \$match

```
1 /**
2  * query: The query in MQL.
3  */
4 {
5   directors: "John Landis"
6 }
```

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

```
{
  "_id": ObjectId('573a1397f29313caabce6d94'),
  "fullplot": "Faber College has one frat house so disreputable it will take anyone. ...",
  "imdb": {
    "year": 1978,
    "plot": "At a 1962 college, Dean Vernon Wormer is determined to expel the entire...",
    "genres": [ "Drama" ]
  }
}
```

```
{
  "_id": ObjectId('573a1397f29313caabce7...'),
  "plot": "Jake Blues, just out from prison together his old band to save...",
  "genres": [ "Drama", "Music", "Romance" ],
  "runtime": 133,
  "rated": "R",
  "cast": [ "John Landis", "John Cazale", "Faye Dunaway", "John Cazale" ],
  "num_mflix_comments": 1,
  "poster": "https://m.media-..."
}
```

Para el segundo stage utilice otro **metodo \$match** para filtrar la puntuacion de las peliculas entre los valores mostrados.

Stage 2 \$match

```
1 /**
2  * query: The query in MQL.
3  */
4 {
5   "imdb.rating": { $gte: 7.5, $lte: 8 }
6 }
```

Output after \$match stage (Sample of 4 documents)

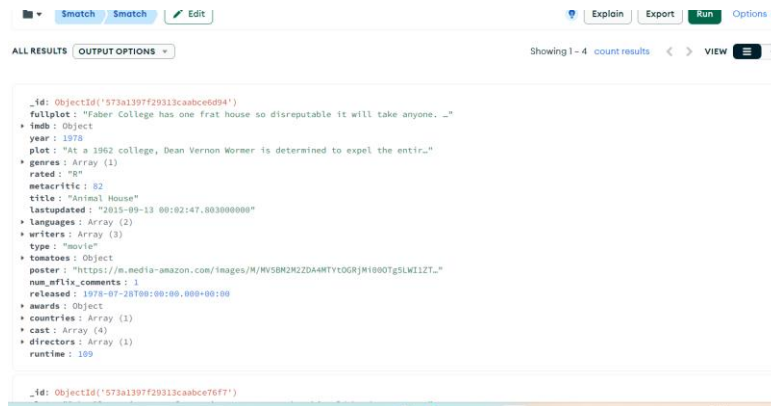
```
{
  "_id": ObjectId('573a1397f29313caabce6d94'),
  "fullplot": "Faber College has one frat house so disreputable it will take anyone. ...",
  "imdb": {
    "year": 1978,
    "1": "At a 1962 college, Dean Vernon Wormer is determined to expel the entire...",
    "genres": [ "Drama" ]
  }
}
```

```
{
  "_id": ObjectId('573a1397f29313caabce7...'),
  "plot": "Jake Blues, just out from prison together his old band to save...",
  "genres": [ "Drama", "Music", "Romance" ],
  "runtime": 133,
  "rated": "R",
  "cast": [ "John Landis", "John Cazale", "Faye Dunaway", "John Cazale" ],
  "num_mflix_comments": 1,
  "poster": "https://m.media-..."
}
```

+ Add Stage

[Learn more about aggregation pipeline stages](#)

Mostrando 4 peliculas



The screenshot shows a Jupyter Notebook interface with a code cell containing a JSON object. The interface includes a top bar with 'Smatch' buttons and an 'Edit' button, and a right bar with 'Explain', 'Export', 'Run', and 'Options' buttons. Below the code cell, there is a 'VIEW' button and a 'count results' indicator. The JSON object represents a movie entry with various fields including _id, fullPlot, imdb, year, plot, genres, rated, metascore, title, lastupdated, languages, writers, tomatoes, poster, num_offline_comments, released, awards, countries, cast, directors, and runtime.

```
{
  "_id": "573a1397f29313caabce6d94",
  "fullPlot": "Faber College has one frat house so disreputable it will take anyone. ...",
  "imdb": {
    "year": 1979,
    "plot": "At a 1962 college, Dean Vernon Wormer is determined to expel the entire...",
    "genres": [
      "Comedy"
    ],
    "rated": "R",
    "metascore": 82,
    "title": "Animal House",
    "lastupdated": "2015-09-13 00:02:47.003000000",
    "languages": [
      "English"
    ],
    "writers": [
      "Harold Heins"
    ],
    "tomatoes": {
      "poster": "https://m.media-amazon.com/images/M/MV5BM2M2ZDAAHTY0RjM1ODU0TgSLW11ZT..."
    },
    "num_offline_comments": 1,
    "released": "1978-07-28T00:00:00.000+00:00",
    "awards": {
      "countries": [
        "USA"
      ],
      "cast": [
        "John Belushi",
        "Sally Field",
        "John Cazale",
        "Charles Hallahan"
      ],
      "directors": [
        "John Wood"
      ],
      "runtime": 109
    }
  }
}
```

Ejercicio 2

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

Para realizar el mapa colección **theathres** entre a **schema**, luego **Analyze** y abrir **location**, luego **geo** y abre directamente el mapa

This report is based on a sample of 1000 documents. [Learn more](#)

