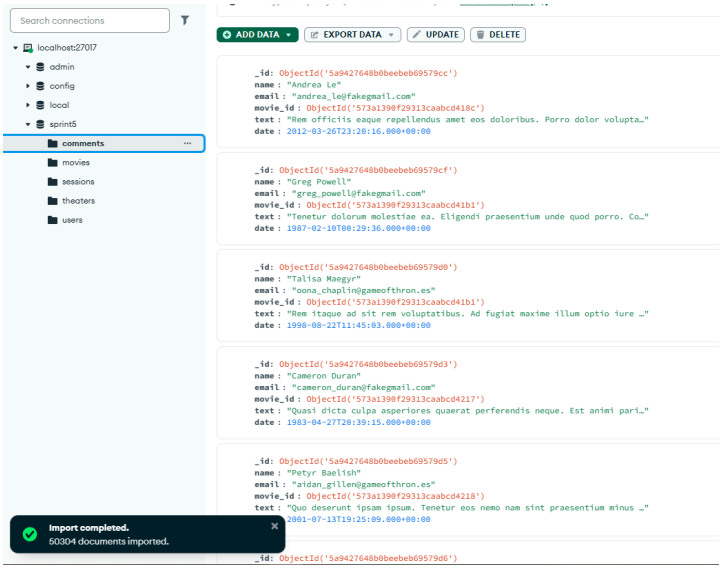
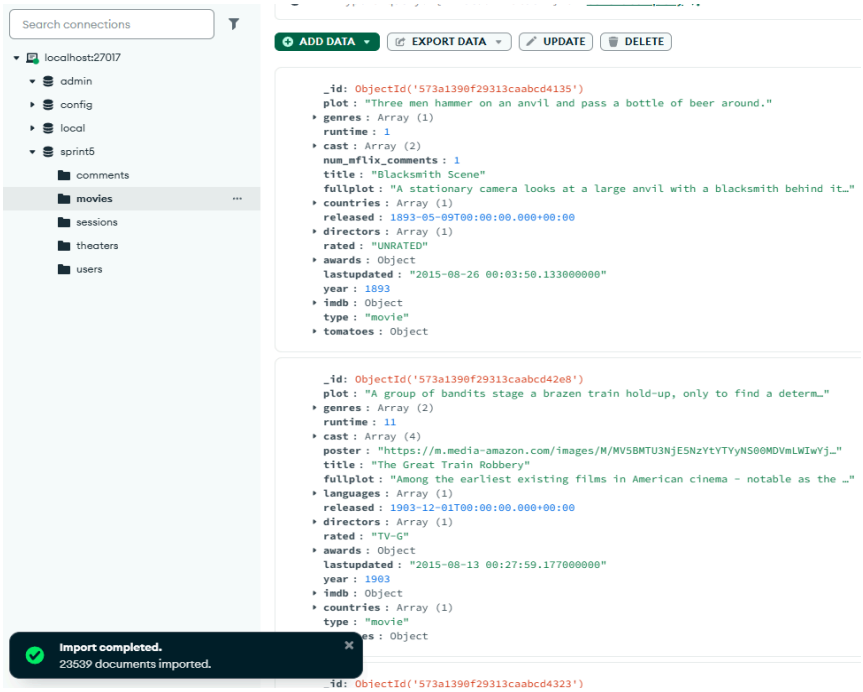
★☆☆ Nivel 1

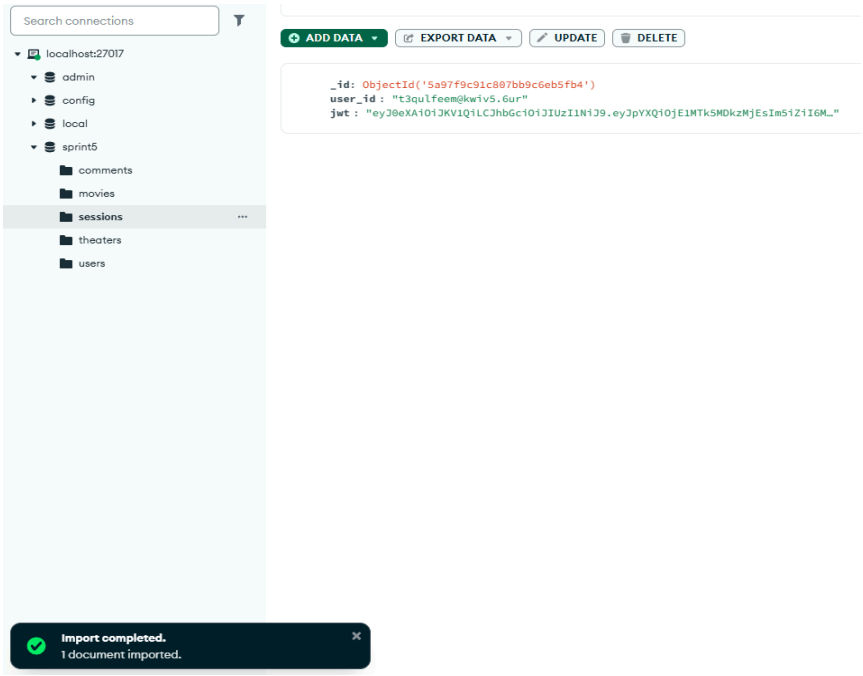
Crea una base de datos con MongoDB utilizando como colecciones los archivos adjuntos.

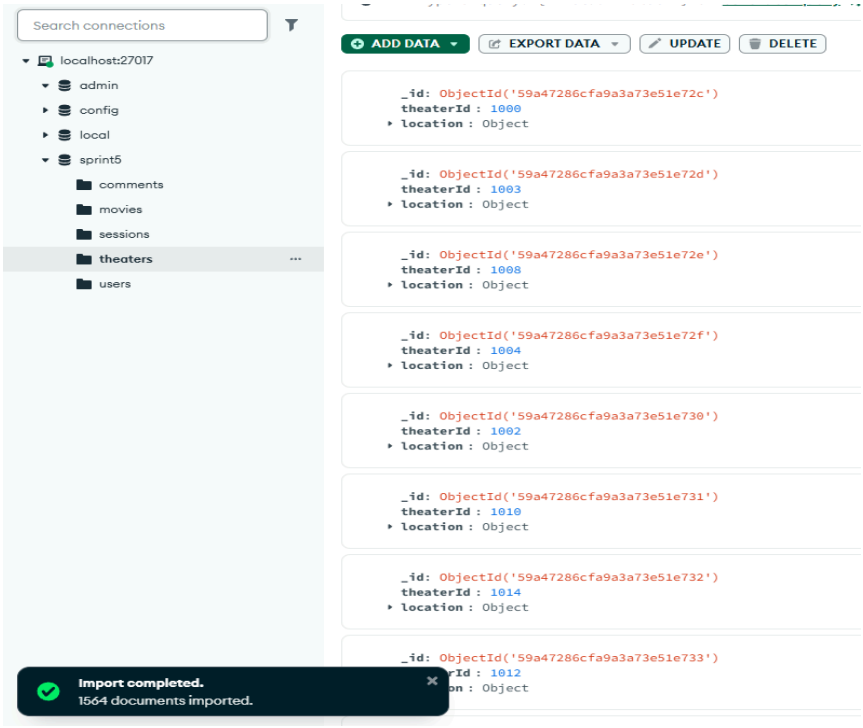
Explicación código: aquí se observa como el primer comando **use sprint5,** hace referencia a la base de datos, la cual aún no está creada. Al crear las collections con **db.createCollection** (comments, movies, sessions, theaters and users), ahí es cuando se crea.

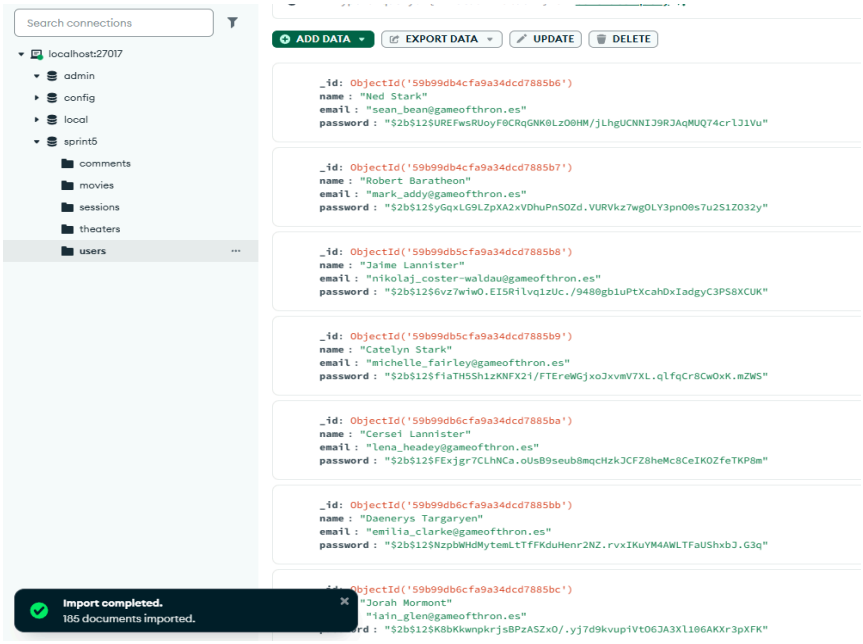
Para cargar los archivos **JSON** en cada una de las tablas que correspondía, se uso **ADD DATA**. Se muestran las imágenes por cada collection que se le cargó información.




Search connections
localhost:27017
admin
config
local
sprint5
comments
movies
sessions
theaters
users
ADD DATA
EXPORT DATA
UPDATE
DELETE
_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 large anvil with a blacksmith behind it…"
countries : Array (1)
released : 1893-05-09T00:00:00.000+00:00
directors : Array (1)
rated : "UNRATED"
awards : Object
lastupdated : "2015-08-26 00:03:50.133000000"
year : 1893
imdb : Object
type : "movie"
tomatoes : Object
_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/MV5BMTU3NjE5NzYtYTYyNS00MDVmLWIwYj…"
title : "The Great Train Robbery"
fullplot : "Among the earliest existing films in American cinema - notable as the …"
languages : Array (1)
released : 1903-12-01T00:00:00.000+00:00
directors : Array (1)
rated : "TV-G"
awards : Object
lastupdated : "2015-08-13 00:27:59.177000000"
year : 1903
imdb : Object
countries : Array (1)
type : "movie"
Import completed.
23539 documents imported.
_id: ObjectId('573a1390f29313caabcd4323')


Search connections
localhost:27017
admin
config
local
sprint5
comments
movies
sessions
theaters
users
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('5a97f9c91c807bb9c6eb5fb4')
user_id : "t3qulfeem@kwiv5.6ur"
jwt : "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1MTk5MDkzMjEsIm5iZiI6M…"
Import completed.
1 document imported.


Search connections
localhost:27017
admin
config
local
sprint5
comments
movies
sessions
theaters
users
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('59a47286cfa9a3a73e51e72c')
theaterId : 1000
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e72d')
theaterId : 1003
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e72e')
theaterId : 1008
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e72f')
theaterId : 1004
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e730')
theaterId : 1002
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e731')
theaterId : 1010
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e732')
theaterId : 1014
location : Object
_id: ObjectId('59a47286cfa9a3a73e51e733')
theaterId : 1012
location : Object
Import completed.
1564 documents imported.


Search connections
localhost:27017
admin
config
local
sprint5
comments
movies
sessions
theaters
users
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('59b99db4cfa9a34dcd7885b6')
name : "Ned Stark"
email : "sean_bean@gameofthron.es"
password : "$2b$12$UREFwsRUoyF0CRqGNK0LzO0HM/jLhgUCNNIJ9RJAqMUQ74crlJ1Vu"
_id: ObjectId('59b99db4cfa9a34dcd7885b7')
name : "Robert Baratheon"
email : "mark_addy@gameofthron.es"
password : "$2b$12$yGqxLG9LZpXA2xVDhuPnSOZd.VURVkz7wgOLY3pnO0s7u2S1ZO32y"
_id: ObjectId('59b99db5cfa9a34dcd7885b8')
name : "Jaime Lannister"
email : "nikolaj_coster-waldau@gameofthron.es"
password : "$2b$12$6vz7wiwO.EI5Rilvq1zUc./9480gb1uPtXcahDxIadgyC3PS8XCUK"
_id: ObjectId('59b99db5cfa9a34dcd7885b9')
name : "Catelyn Stark"
email : "michelle_fairley@gameofthron.es"
password : "$2b$12$fiaTH5Sh1zKNFX2i/FTEreWGjxoJxvmV7XL.qlfqCr8CwOxK.mZWS"
_id: ObjectId('59b99db6cfa9a34dcd7885ba')
name : "Cersei Lannister"
email : "lena_headey@gameofthron.es"
password : "$2b$12$FExjgr7CLhNCa.oUsB9seub8mqcHzkJCFZ8heMc8CeIKOZfeTKP8m"
_id: ObjectId('59b99db6cfa9a34dcd7885bb')
name : "Daenerys Targaryen"
email : "emilia_clarke@gameofthron.es"
password : "$2b$12$NzpDWHdMytemLtTfFKduHenr2NZ.rvxIKuYM4AWLTFaUShxbJ.G3q"
_id: ObjectId('59b99db6cfa9a34dcd7885bc')
"Jorah Mormont"
"iain_glen@gameofthron.es"
rd : "$2b$12$K8bKkwnpkrjsBPzASZxO/.yj7d9kvupiVtO6JA3Xl106AKXr3pXFK"
Import completed.
185 documents imported.

Ejercicio 1

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

```
> db.comments.find().limit(2).pretty()
< {
    _id: ObjectId('5a9427648b0beebeb69579cc'),
    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 unde nulla temporibus ullam.',
    date: 2012-03-26T23:20:16.000Z
  }
  {
    _id: ObjectId('5a9427648b0beebeb69579cf'),
    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 neque. Dolorum nihil vero animi.',
    date: 1987-02-10T00:29:36.000Z
  }
```

Explicación código: buscamos los comentarios con el comando **find** y lo limitamos, con **limit**, a que nos de 2 resultados. Como no hay un criterio de orden específico, me devolvió los primeros documentos que aparecen físicamente en la collection.

- ***¿Cuántos usuarios tenemos registrados?***

```
>_MONGOSH
> use sprint5
< switched to db sprint5
> db.users.countDocuments()
< 185
sprint5>
```

- ***¿Cuántos cines existen en el estado de California?***

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

Explicación código: en la colección **theaters**, se reflejaban los estados siguiendo una ruta de claves (**location.address.state**) y fue la que se utilizó para buscar el estado en la consulta.

- ***¿Cuál fue el primer usuario en registrarse?***

```
> db.users.find().sort({ _id: 1 }).limit(1)
< {
    _id: ObjectId('59b99db4cfa9a34dcd7885b6'),
    name: 'Ned Stark',
    email: 'sean_bean@gameofthron.es',
    password: '$2b$12$UREFwsRUoyF0CRqGNK0LzO0HM/jLhgUCNNIJ9RJAqMUQ74crlJ1Vu'
  }
```

Explicación código: en la colección de **users** no hay ninguna fecha que nos indique el orden de registro de los usuarios. Cuando MongoDB crea algo nuevo, le da un **_id**. Ese número ya guarda la fecha en que se creó. Por eso, se puede saber quién se registró primero solo mirando el orden de los id.

- ***¿Cuántas películas de comedia existen en nuestra base de datos?***

```
> db.movies.find({"genres":"Comedy"}).count()
< 7024
```

Ejercicio 2

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

```
> db.movies.find({$and: [ {$or:[{genres:"Drama"},{languages:"French"}]},{year:1932}]})
< {
    _id: ObjectId('573a1391f29313caabcd9458'),
    plot: 'A young artist draws a face at a canvas on his easel. Suddenly the mouth on the drawing come
    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/MV5BYWY3ODE5ZWEtYjlmYi00NjA4LTk4ZWYtMzBhZDE5MjY0YTYxXk
    title: 'The Blood of a Poet',
    lastupdated: '2015-09-16 13:13:05.537000000',
    languages: [
      'French'
    ],
    released: 2010-05-20T00:00:00.000Z,
```

Ejercicio 3

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

```
> 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 f
    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 wo
    genres: [
      'Adventure',
      'Comedy',
      'Drama'
    ],
    rated: 'PG',
    metacritic: 54,
    title: 'The Secret Life of Walter Mitty',
```

★★☆ Nivel 2

Ejercicio 1

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

```
> db.comments.find({
    email: /GAMEOFTHRON\.ES$/i
  }).count()
< 22841
```

Explicación código: la idea aquí es encontrar todos los correos que terminen en GAMEOFTHRON.ES, y lo hacemos de la siguiente manera

- ★ *Usamos los slashes (///) para definir un patrón de búsqueda que encuentre coincidencias con lo que contengan.*
- ★ *$, indica que la frase termina en todo lo que esté antes que él, en este caso GAMEOFTHRON.ES*
- ★ *\ , indica que lo que esté luego sea literal, en este caso el punto, ya que en mongodb el . solo indica cualquier caracter.*
- ★ *i, indica que lo que este antes sea insensible a mayúsculas/minúsculas*

Ejercicio 2

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

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

Explicación código: a continuación explico cada uno de los comandos usados y su función específica dentro de la consulta:

- ★ ***$match****: filtra los documentos que están en el estado DC .*
- ★ ***$group****: agrupa los documentos por código postal.*
- ★ ***id en $group****: Cada grupo es un código postal diferente.*
- ★ ***count:{ $sum: 1 }****: Cuenta cuántos cines hay en cada código postal sumando 1 por cada documento en el grupo.*

★★★ 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.

```
> db.movies.find({"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 w
    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 t
    genres: [
      'Comedy'
    ],
    rated: 'R',
    metacritic: 82,
```

Ejercicio 2

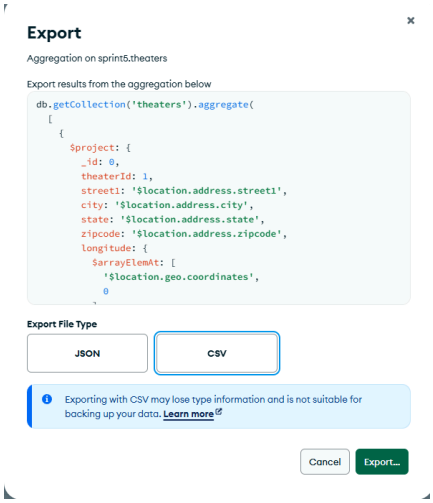
Muestra en un mapa la ubicación de todos los teatros de la base de datos.

1) *En **theaters** le damos formato correcto en Aggregations para exportarlo como csv agregando un nuevo stage llamado $project y corriendo el código de la imagen, convierte las rutas en columnas con nombres simples que compondrán el archivo csv.*

Stage 1 $project

```
{
  "_id": 0,
  "theaterId": 1,
  "street1": "$location.address.street1",
  "city": "$location.address.city",
  "state": "$location.address.state",
  "zipcode": "$location.address.zipcode",
  "longitude": { "$arrayElemAt": ["$location.geo.coordinates", 0] },
  "latitude": { "$arrayElemAt": ["$location.geo.coordinates", 1] }
}
```

2) *Luego lo exportamos como csv*



3) *Cargamos nuestro csv en google my maps (https://www.google.com/maps/d/)*