

SPRINT 5 – Data Analytics

A continuación, se presentan los resultados del Sprint #5 de la especialización Data Analytics de IT Academy, realizados por Rossemary Castellanos entregado el día 17/10/2025. Se realizaron los 3 niveles del Sprint. Considerando que los resultados por su extensión en ocasiones es complicado mostrarlos por completo, se cuantificaron para así comparar las respuestas.

Trabajaremos 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.

Realizarás algunas consultas que te pide el cliente/a, quien está midiendo si serás capaz o no de hacerte cargo de la parte analítica del proyecto vinculado con su base de datos.

Las colecciones de datos se encuentran dentro de los siguientes archivos. JSON listados:

- "comments.json"
- "movies.json "
- "sesions.json "
- "theaters.json"
- "users.json"

NIVEL 1

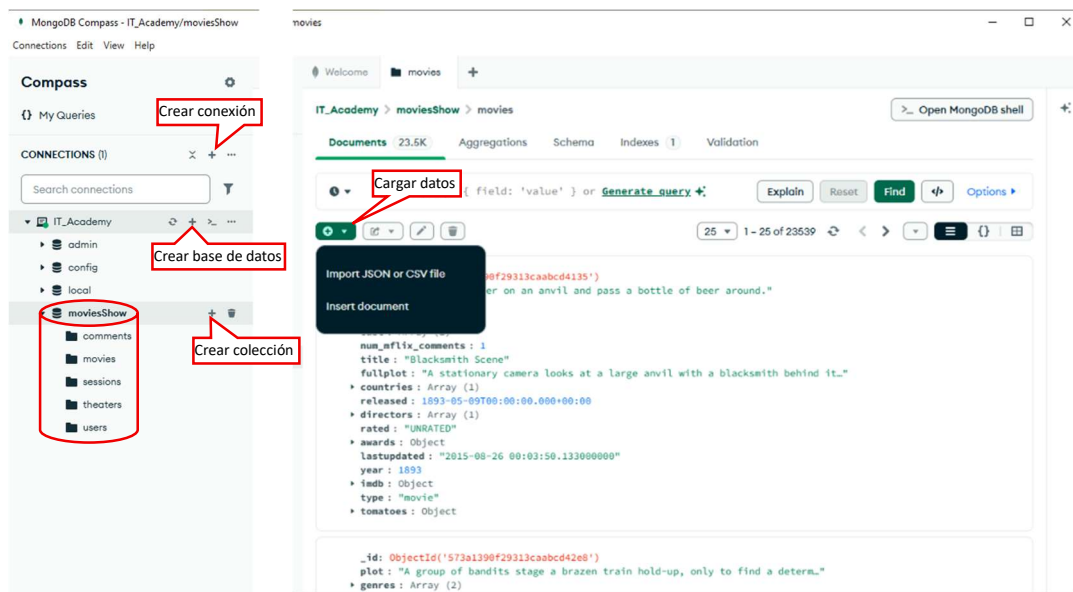
Crea una base de datos con MongoDB utilizando como colecciones los archivos adjuntos.

Primero se instala MongoDB, descargando del sitio web oficial el archivo de instalación según el sistema operativo, luego ejecutar el instalador siguiendo los pasos. Es recomendable también instalar MongoDB Compass para tener una interfaz gráfica para gestionar la base de datos.

Al abrir la aplicación MongoDB Compass se debe crear una conexión que hace de puente con el servidor para enviar y recibir datos de la BD. Después se crea la Base de Datos, en este sprint se llamó **moviesShow**.

Lo siguiente es crear y cargar las colecciones (homologo a las tablas en MySQL). Se crean cada una de las colecciones en función de los archivos json entregados: *comments.json*, *movies.json*, *sesions.json*, *theaters.json*, *users.json*. MongoDB permite cargar archivos json, csv o realizar carga manual.

En la imagen anexa se describe los puntos de creación y carga de la base de datos y las colecciones, así como la estructura de la base de datos **moviesShow**.

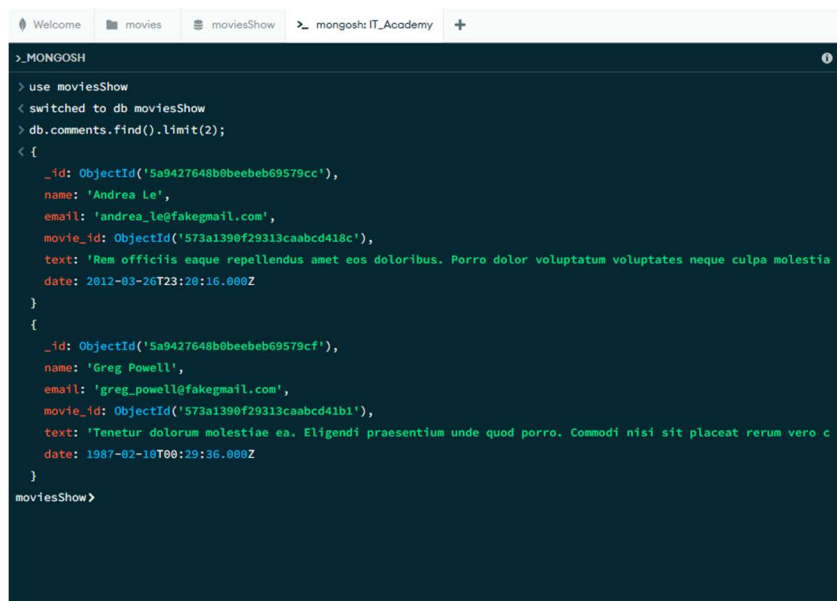


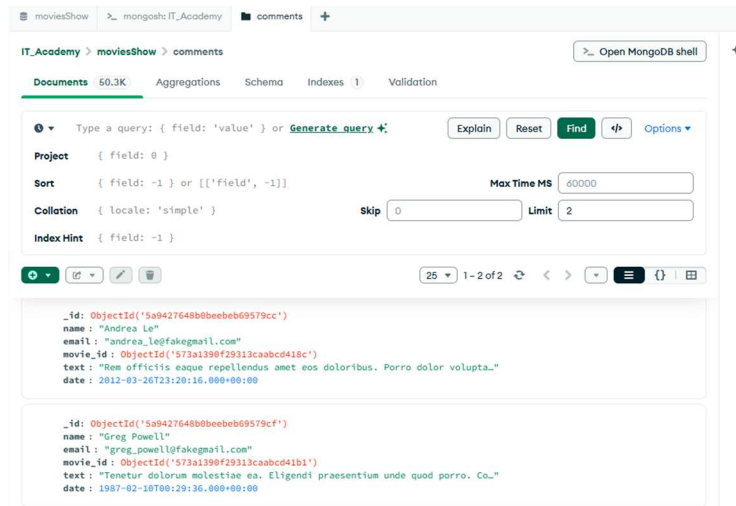
Es importante resaltar que MongoDB Compass cuenta con opción de desplegar una ventana terminal (Open MongoDB Shell), la cual fue utilizada para ejecutar las consultas.

Ejercicio 1

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

La forma de realizar consultas en MongoDB es situarse en la base de datos, con el comando *“use nombre_BD”* como se observa en la imagen. Ya dentro de **moviesShow** se llama a la colección donde se realizará la consulta *“db.nombre_colección”*. La instrucción para realizar búsquedas es *.find()* en caso de tener filtros para la búsqueda se introducen dentro de los paréntesis. Para delimitar se usa *limit()* indicando entre los paréntesis cuantos documentos se desean ver, esta instrucción presenta los documentos en el orden que fueron registrados. En este ejercicio se consultó la colección **comments** aplicando *limit(2)* y ver 2 documentos.





Las siguientes preguntas están plasmadas en la imagen terminal anexa al final.

¿Cuántos usuarios tenemos registrados?

Para esta consulta se aplicó `.find()` sobre la colección **users**, junto a la instrucción `.count()` que permite cuantificar el resultado. Resultado 185 usuarios.

¿Cuántos cines existen en el estado de California?

Aquí el `.find()` se ejecutó sobre la colección **theaters**, con el filtro de la ubicación en el estado de California (CA) como `{ "location.address.state": "CA" }`, junto al `.count()` para contar los cines de California. Resultado 169 cines.

¿Cuál fue el primer usuario en registrarse?

Se tiene un comando que permite consultar solo el primer documento de la colección `.findOne()`, esta instrucción se ejecuta sobre la colección **users**.

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

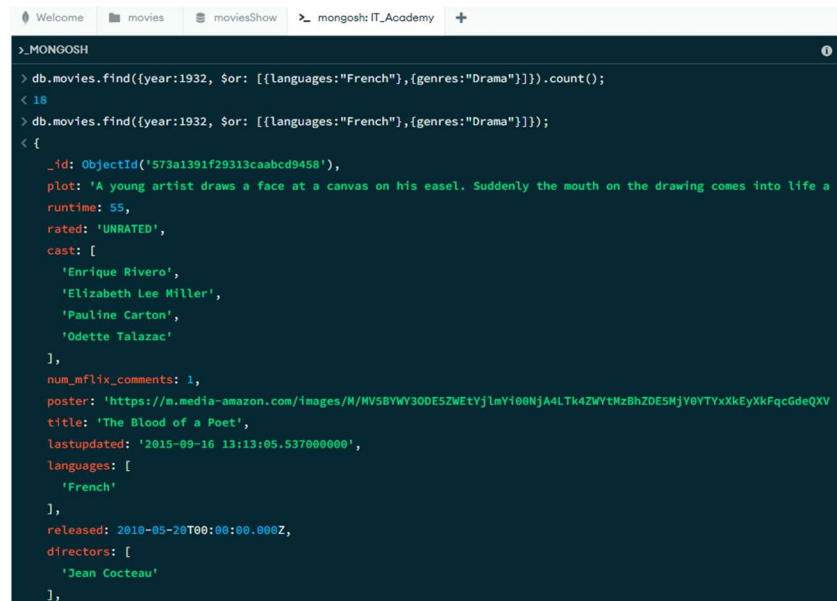
Esta consulta se realiza sobre la colección **movies**, dentro del comando `.find()` se define el filtro del género `{ "genres": "Comedy" }`, para finalmente cuantificar el resultado con un `.count()`. Resultado 7024 películas con genero comedia.

```
> mongosh: IT_Academy | moviesShow | movies | > mongosh: IT_Academy | users | > mongosh: IT_Academy | +
> MONGOSH
> db.users.find().count();
< 185
> db.theaters.find({ "location.address.state": "CA" }).count();
< 169
> db.users.findOne();
< {
  _id: ObjectId('59b99db4cfa9a34dcd7885b6'),
  name: 'Ned Stark',
  email: 'sean_bean@gameofthron.es',
  password: '$2b$12$UREFwsRUoyF0CRqGNK0Lz00HM/jLhgUCNNIJ9RJAqMUQ74cr1J1Vu'
}
> db.movies.find({ "genres": "Comedy" }).count();
< 7024
moviesShow >
```

Ejercicio 2

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

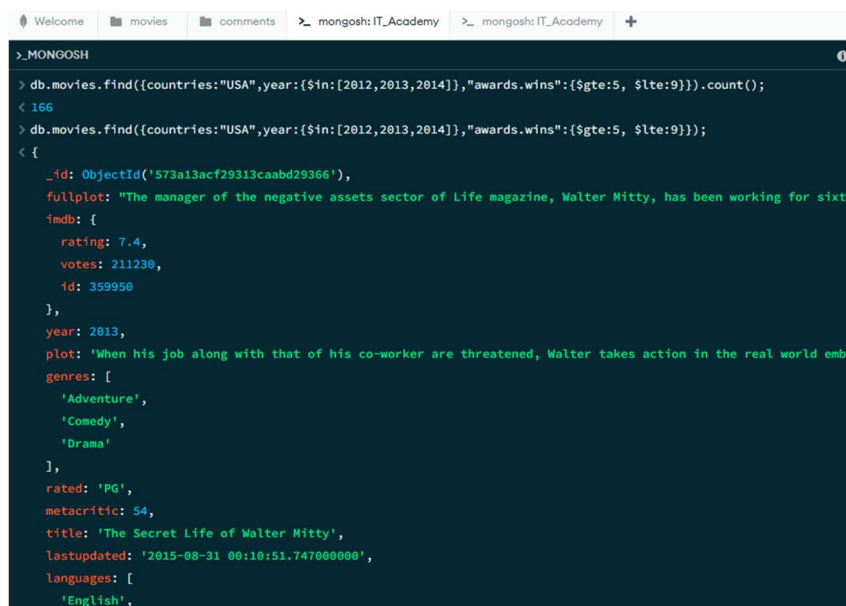
En esta consulta se usa `.find()` en la colección **movies**, los filtros son el año 1932 y lenguaje francés o genero drama; para esto se requiere el comando `$and` y el comando `$or`. El `$and` esta predeterminado al tener dentro del find más de 2 filtros, se asume que las condiciones se adicionan sin necesidad de escribir `$and`. En caso contrario de querer una condición u otra si se debe declarar el comando `$or`. En la imagen anexa se muestra la consulta identificando 18 películas.



```
> MONGOSH
> db.movies.find({year:1932, $or: [{languages:"French"},{genres:"Drama"}]}).count();
< 18
> db.movies.find({year:1932, $or: [{languages:"French"},{genres:"Drama"}]});
< {
  _id: ObjectId('573a1391f29313caabdc9458'),
  plot: 'A young artist draws a face at a canvas on his easel. Suddenly the mouth on the drawing comes into life a
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/MV5B8YVY30DE5ZWetYjlmYi00NjA4LTk4ZWYtMzZhZDE5MjY0YTYxXkEyXkFqc6deQXV
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'
],
}
```

Ejercicio 3

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



```
> MONGOSH
> db.movies.find({countries:"USA",year:{ $in:[2012,2013,2014]},"awards.wins":{$gte:5, $lte:9}}).count();
< 166
> db.movies.find({countries:"USA",year:{ $in:[2012,2013,2014]},"awards.wins":{$gte:5, $lte:9}});
< {
  _id: ObjectId('573a13acf29313caabd29366'),
  fullplot: "The manager of the negative assets sector of Life magazine, Walter Mitty, has been working for sixt
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 emb
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',
]
```

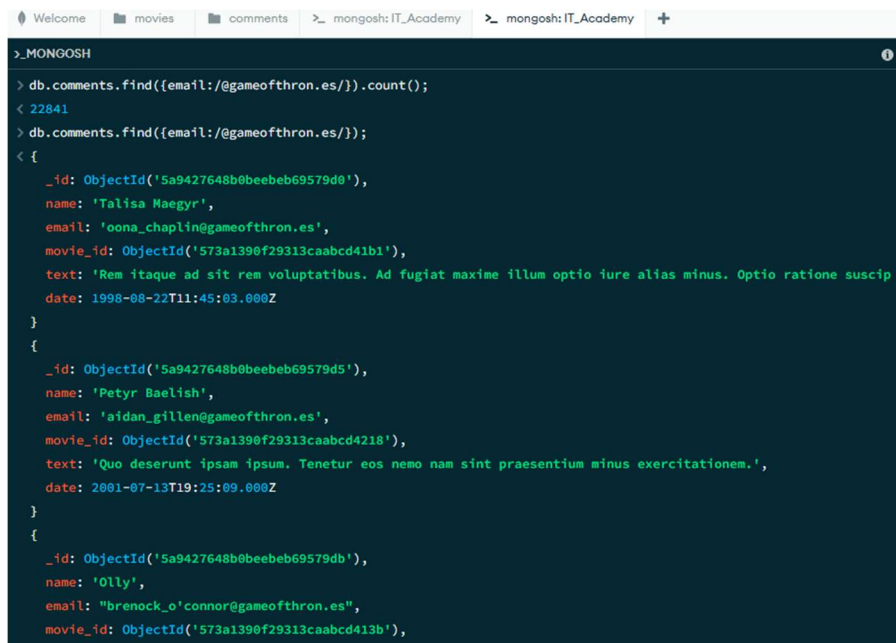
Esta búsqueda va sobre la colección **movies** con el comando `.find()`, los filtros serán país Estados Unidos (USA), los años son 2012,2013 y 2014, y los premios recibidos. Para esta última condición se usó el comando mayor o igual que (\geq) `$gte` y menor o igual que (\leq) `$lte`, con la finalidad de que no se excluyan los valores decimales. En la imagen anterior se presenta la consulta y los resultados siendo 166 películas.

NIVEL 2

Ejercicio 1

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

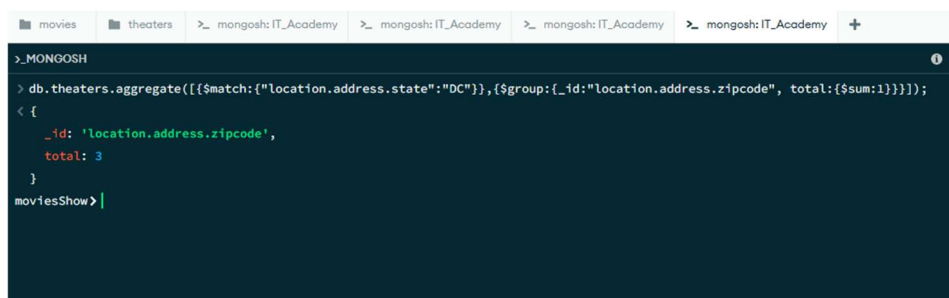
La consulta se aplica sobre la colección **comments**, con el comando `.find()` se filtran el email indicando que debe tener dentro el dominio `"@gameofthron.es"`, para esto se utilizan las expresiones regulares. En este caso se envuelve el texto que buscamos entre 2 barras inclinadas (`/` `/`). En la imagen se muestra la sentencia y el resultado siendo 22841 comentarios con este dominio.



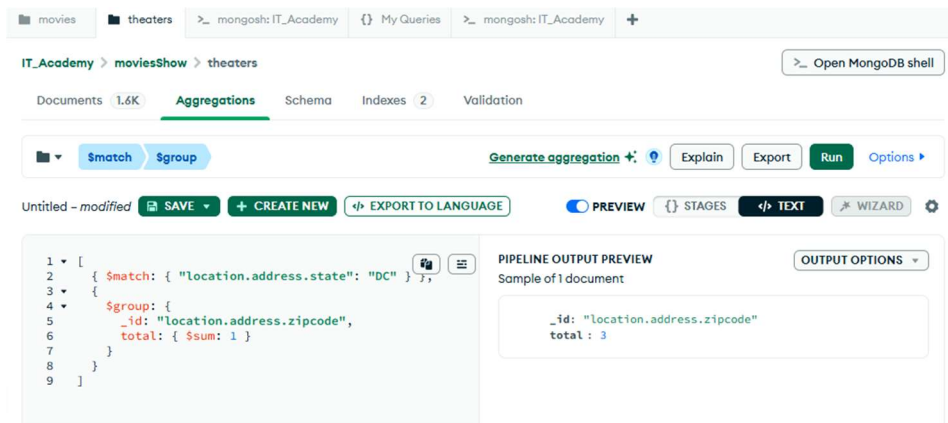
```
> MONGOSH
> db.comments.find({email:/@gameofthron.es/}).count();
< 22841
> db.comments.find({email:/@gameofthron.es/});
< [
  {
    _id: ObjectId('5a9427648b0beebe69579d0'),
    name: 'Talisa Maegyr',
    email: 'oona_chaplin@gameofthron.es',
    movie_id: ObjectId('573a1390f29313caabcd41b1'),
    text: 'Rem itaque ad sit rem voluptatibus. Ad fugiat maxime illum optio iure alias minus. Optio ratione suscip',
    date: 1998-08-22T11:45:03.000Z
  },
  {
    _id: ObjectId('5a9427648b0beebe69579d5'),
    name: 'Petyr Baelish',
    email: 'aidan_gillen@gameofthron.es',
    movie_id: ObjectId('573a1390f29313caabcd4218'),
    text: 'Quo deserunt ipsum ipsum. Tenetur eos nemo nam sint praesentium minus exercitationem.',
    date: 2001-07-13T19:25:09.000Z
  },
  {
    _id: ObjectId('5a9427648b0beebe69579db'),
    name: 'Olly',
    email: 'brenock_o'connor@gameofthron.es',
    movie_id: ObjectId('573a1390f29313caabcd413b'),
    date: 2001-07-13T19:25:09.000Z
  }
]
```

Ejercicio 2

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



```
> MONGOSH
> db.theaters.aggregate([{$match:{'location.address.state':'DC'}},{$group:{_id:'location.address.zipcode', total:{$sum:1}}]);
< [
  {
    _id: 'location.address.zipcode',
    total: 3
  }
]
moviesShow>
```



Esta consulta se ejecuta sobre la colección **theaters**, pero en este caso se necesita agrupar por código postal por lo que se debe usar el comando `.aggregate()`. Dentro se definen los filtros con la instrucción `$match` y se agrupa con `$group`.

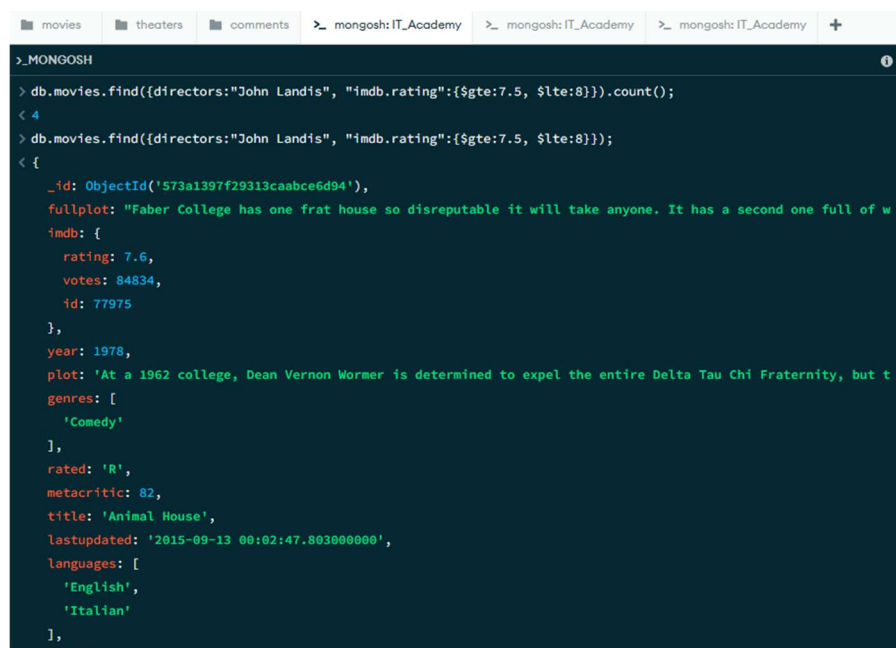
El filtro es el estado de Washington `"location.address.state": "DC"`, y para agrupar se debe identificar el campo `_id: "location.address.zipcode"` con la función de agregación a realizar, aquí totalizar el número de cines total

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.

En este ejercicio se utiliza el comando `.find()` sobre la colección **movies**, dentro se filtra por el director John Landis y la valoración del IMDb (Internet Movie Database, base de datos del mundo en línea sobre cine y televisión), para la valoración se considera usar las instrucciones `$gte` y `$lte`, ya que se pide el *rating* entre 7.5 y 8. El resultado es 4 películas.



Ejercicio 2

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

En este caso la consulta va sobre la colección **theaters**, al revisar los documentos se observa que se cuenta con las coordenadas geográficas de cada cine.

Según la documentación de MongoDB se aceptan para datos geoespaciales formato GeoJSON de 3 tipos (point, linestring y poligon), Los archivos GeoJSON se estructura de 2 campos:

- *type* donde se especifica el tipo de objeto GeoJSON y
- *coordinates* con las coordenadas del objeto.

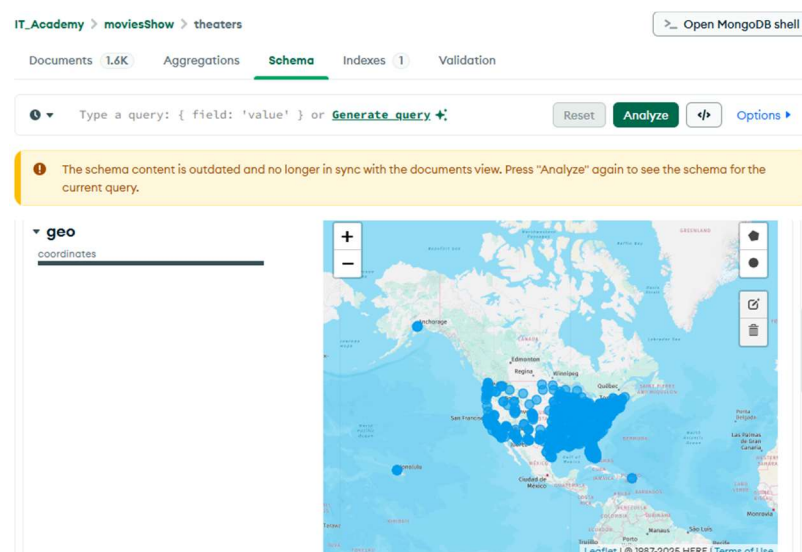
Al revisar la parte del documento donde están las coordenadas se aprecia este tipo de formato.

Para realizar la consulta de coordenadas, se necesita generar un índice con el comando *createIndex*. Los índices geoespaciales permiten realizar consultas sobre datos almacenados, la aplicación tiene 2 tipos *2d* y *2dsphere*.

En la imagen anexa se presenta la configuración de los datos de coordenadas en formato GeoJSON, y la creación de índice geoespacial.

```
> MONGODB
> db.theaters.findOne();
< {
  _id: ObjectId('59a47286cfa9a3a73e51e72c'),
  theaterId: 1888,
  location: {
    address: {
      street1: '348 W Market',
      city: 'Bloomington',
      state: 'MN',
      zipcode: '55425'
    },
    geo: {
      type: 'Point',
      coordinates: [
        -93.24565,
        44.85466
      ]
    }
  }
}
> db.theaters.createIndex({"location.geo": "2dsphere"});
< location_geo_2dsphere
moviesShow>
```

A continuación, se presenta el mapa donde se aprecia la ubicación de los cines de la colección **theaters**. Este se encuentra en la pestaña de esquema.



Como resumen mongodb permite guardar las consultas realizadas en {} My Queries. Esto anexándola como favorita y dando un nombre descriptivo de que realiza.

