Sprint 9 – Mongo DB (DB no relacional)

- Establecer el escritorio de trabajo

Instalación y ejecución del programa Mongo DB

Instalé el mongodb desde la página oficial

En Productos → instalar Community Edition

Durante la instalación: destildar que se inicie automáticamente, y ver que esté tildado que se instale también el mongo compas.

Luego para que el programa funcione:

Desde el explorador de Windows, creé en C:\ una carpeta "**data**" y una subcarpeta "**bd**" para que pueda correr el programa, y se alojen allí los documentos que se generen.

Finalmente ejecuté el MongoDB: ir a inicio de Windows → ejecutar → cmd → se abre la consola y escribo "mongod" para ejecutar

Creamos una conexión y aparecen 3 bd preestablecidas

Creamos una nueva base de datos, y luego colecciones en las cuales alojamos cada uno de los archivos de los recursos

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

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

• theatres: Almacena 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: Tiene detalles de películas, como trama, géneros, duración, elenco, comentarios, año de lanzamiento, directores, clasificación y premios.

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

Nivel 1

Con « cls » limpio la terminal.

- Ejercicio 1

Mostra els 2 primers comentaris que hi ha en la base de dades.

Para visualizar los primeros dos documentos de la colección comentarios usamos el comando:

db.coleccion.find().limit(2)

db["comments"].find() es una forma alternativa útil si el nombre de la colección contiene caracteres especiales o espacios.

```
> db.comments.find().limit(2)
< {
    _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
    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
    date: 1987-02-10T00:29:36.000Z
  }
mi_bd>
```

Quants usuaris tenim registrats?

db.users.countDocuments()

```
> db.users.countDocuments()
< 185
```

Quants cinemes hi ha en l'estat de Califòrnia?

Contamos los documentos que hay en la colección de cines y filtramos por aquellos ubicados en el estado CA

db.theaters.countDocuments({‘location.address.state’:’CA’})

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

Quin va ser el primer usuari/ària en registrar-se?

Para encontrar el primer usuario registrado en la colección, usualmente se buscaría un campo que indique la fecha de creación. Sin embargo, en este caso no hay un campo de fecha explícito, por lo que usamos el campo _id, ya que los ObjectId generados por MongoDB contienen una marca de tiempo en sus primeros cuatro bytes, la cual representa los segundos transcurridos desde el 1 de enero de 1970 (época UNIX).

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

```
> // Para encontrar el primer usuario registrado en la colección,
  // usualmente se buscaría un campo que indique la fecha de alta de usuario.
  // Sin embargo, en este caso no hay un campo de fecha explícito,
  // por lo que usamos el campo _id, ya que los ObjectId generados
  // por MongoDB contienen una marca de tiempo en sus primeros cuatro bytes,
  // la cual representa los segundos transcurridos desde el 1 de enero de 1970 (época UNIX).
  db.users.find().sort({_id:1}).limit(1)
< {
    _id: ObjectId('59b99db4cfa9a34dcd7885b6'),
    name: 'Ned Stark',
    email: 'sean_bean@gameofthron.es',
    password: '$2b$12$UREFwsRUoyF0CRqGNK0LzO0HM/jLhgUCNNIJ9RJAqMUQ74crlJ1Vu'
  }
```

Quantes pel·lícules de comèdia hi ha en la nostra base de dades?

Primero consulto cúales son los distintos géneros que existen, y encontrar cómo aparece escrito ‘comedia’ para luego filtrarlo.

db.movies.distinct("genres")

```
> db.movies.distinct("genres")
< [
    'Action',      'Adventure', 'Animation',
    'Biography',   'Comedy',    'Crime',
    'Documentary', 'Drama',     'Family',
    'Fantasy',     'Film-Noir', 'History',
    'Horror',      'Music',     'Musical',
    'Mystery',     'News',      'Romance',
    'Sci-Fi',      'Short',     'Sport',
    'Talk-Show',   'Thriller',  'War',
    'Western'
  ]
```

Luego consultamos un par de registros para ver qué datos contienen

db.movies.find().limit(2)

y podemos apreciar un campo type dónde dice movie, lo exploramos:

db.movies.distinct('type')

y descubrimos que hay tanto películas como series

```
> db.movies.distinct('type')
< [ 'movie', 'series' ]
```

Finalmente hacemos la consulta con estos dos parámetros

db.movies.count({

genres: "Comedy",

type: "movie"

})

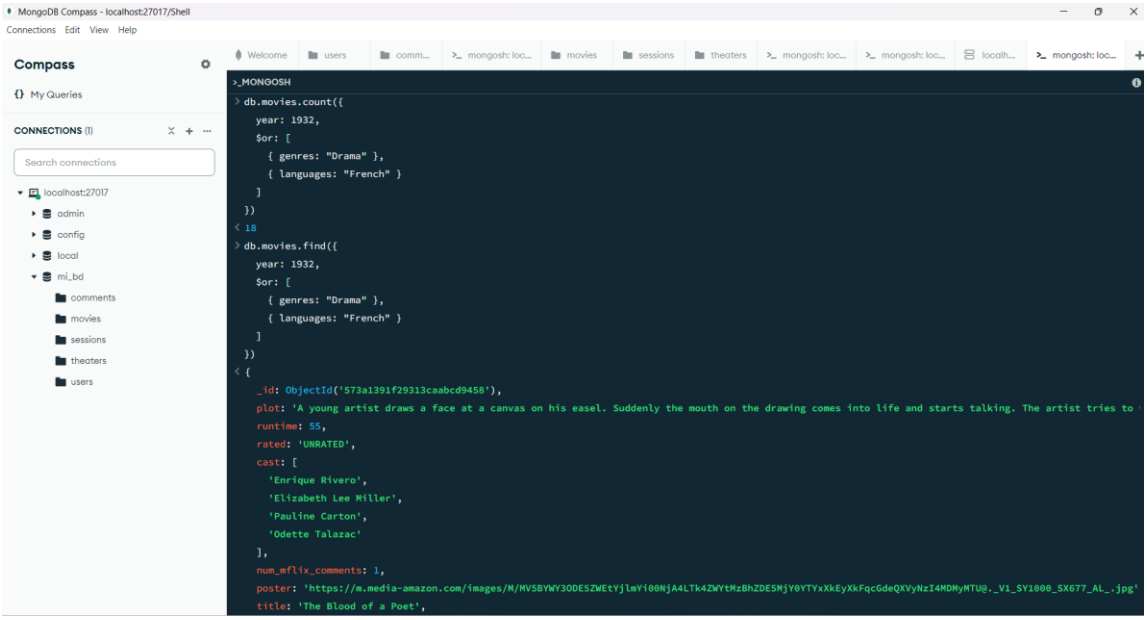
```
> db.movies.count({
    genres: "Comedy",
    type: "movie"
  })
< 7002
```

- Ejercicio 2

Mostra'm tots els documents de les pel·lícules produïdes en 1932, però que el gènere sigui drama o estiguin en francès.

Primero vamos a trabajar con el count para ver que vamos haciendo los filtros correctos.

Filtramos el año y luego que sea drama o en francés. Son 18 registros



En la captura no se aprecia, pero controlamos los registros que cumplan con los filtros de año y género o idioma.

- Ejercicio 3

Mostra'm tots els documents de pel·lícules estatunidenques que tinguin entre 5 i 9 premis que van ser produïdes entre 2012 i 2014.

Mostramos las consultas y un fragemento del primer registro, de los 162 que cumplen con los parámetros solicitados.

Se utiliza {$gte: valor, $lte: valor} para comparaciones de menor o igual que y mayor o igual que.

```
> db.movies.find({
  type:'movie',
  countries: 'USA',
  'awards.wins': {$gte:5,$lte:9},
  year:{$gte:2012,$lte:2014}
  }).count()
< 162
> db.movies.find({
  type:'movie',
  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
    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
    genres: [
      'Adventure',
      'Comedy',
      'Drama'
```

Verificamos los parámetros de algunos registros

```
    poster: 'https://m.media-amazon.com/images/M/MV5BODYwNDY
    num_mflix_comments: 1,
    released: 2013-12-25T00:00:00.000Z,
    awards: {
      wins: 6,
      nominations: 13,
      text: '6 wins & 13 nominations.'
    },
    countries: [
      'USA',
      'Canada'
```

Nivel 2

- Ejercicio 1

Compte quants comentaris escriu un usuari/ària que utilitza "GAMEOFTHRON.ES" com a domini de correu electrònic.

En la consulta utilizamos $regex, que es equivalente al LIKE en lenguaje SQL.

$regex: /@gameofthron.es$/i

- $: para indicar que email tiene que terminar en '@gameofthron.es'
- i: para indicar que no haga distinción entre mayúsculas y minúsculas

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

- Ejercicio 2

Quants cinemes hi ha en cada codi postal situats dins de l'estat Washington D. C. (DC)?

Aggregate

Para este ejercicio ya vamos a necesitar realizar operaciones mas complejas por tanto utilizaremos la el método aggregate, que permite filtros, agrupaciones y cálculos en varias etapas. Cada etapa realiza una tarea específica y pasa los resultados a la siguiente etapa.

$match para filtrar dentro de aggregate como lo hacíamos con find

$group agrupo por zipcode, por convención debo asignarle _id al agrupador, asigno un nombre al resultado del agrupamiento en el que contamos la cantidad de documentos de cada grupo (zipcode)

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

Nivel 3

- Ejercicio 1

Troba totes les pel·lícules dirigides per John Landis amb una puntuació IMDb (Internet Movie Database) d'entre 7,5 i 8.

Para encontrar el resultado filtramos el tipo movie, el director John Landis y la puntuación entre 7.5 y 8. Luego verificamos que hay 4 películas y mostramos un fragmento de la primera.

```
>_MONGOSH
> db.movies.find({
  'type':'movie',
  'directors':'John Landis',
  'imdb.rating':{'$gte':7.5,'$lte':8}
  }).count()
< 4
> 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 wh
    imdb: {
      rating: 7.6,
```

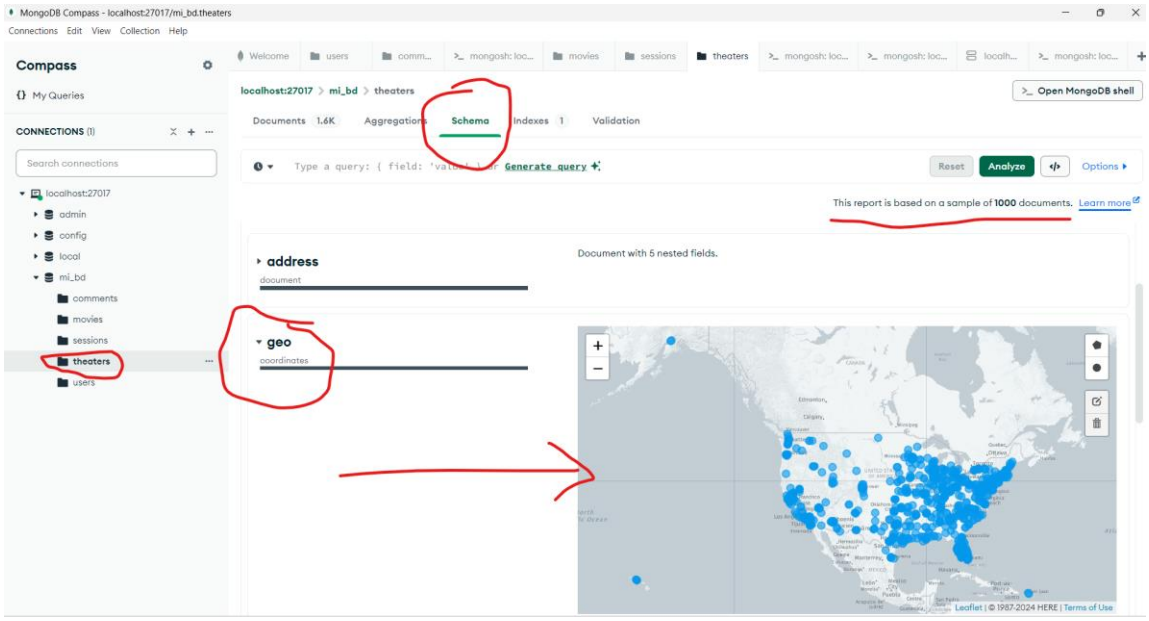
Finalmente chequeamos que los registros estén filtrando adecuadamente según los parámetros. Dado que cada documento es muy largo y resulta difícil la visualización de todos los parámetros de un documento, decidimos visualizar de cada Movie sólo los parámetros que estamos filtrando y el id.

```
> db.movies.find({
  'type':'movie',
  'directors':'John Landis',
  'imdb.rating':{'$gte':7.5,'$lte':8}
  },
  {'type':1,
  'directors':1,
  'imdb.rating':1
  }
  )
< {
    _id: ObjectId('573a1397f29313caabce6d94'),
    imdb: {
      rating: 7.6
    },
    type: 'movie',
    directors: [
      'John Landis'
    ]
  }
  {
    _id: ObjectId('573a1397f29313caabce76f7'),
    directors: [
      'John Landis'
    ],
    imdb: {
      rating: 7.9
    },
    type: 'movie'
```

- Ejercicio 2

Mostra en un mapa la ubicació de tots els teatres de la base de dades.

Para resolver este ejercicio no voy a utilizar el mongo db Shell (la consola o la terminal) sino que voy a utilizar las opciones de menú de MongoDB Compass



Seleccionamos el Schema y la colección de theaters, allí la geoloalización dónde están los datos de latitud y longitud.

Se muestra un mapa con la ubicación de los cines. Nos indica que se toma una muestra de 1000 documentos para hacer mas eficiente su procesamiento.