

## Tasca S5.01. Consultes amb MongoDB

# Introducció

N1: Treballarem amb una base de dades no relacional que conté col·leccions relacionades amb una aplicació d'entreteniment cinematogràfic:

Crea una base de dades amb MongoDB utilitzant com a col·leccions els arxius adjunts

Descripció de la base de dades, les taules que tindre (col·leccions)

- users: Emmagatzema informació d'usuaris/es, incloent-hi noms, emails i contrasenyes xifrades.
- theatres: Conté dades de cinemes, com ID, ubicació (direcció i coordenades geogràfiques).
- sessions: Guarda sessions d'usuari, incloent-hi ID d'usuari i tokens JWT per a l'autenticació.
- movies: Inclou detalls de pel·lícules, com a trama, gèneres, durada, elenc, comentaris, any de llançament, directors, classificació i premis.
- comments: Emmagatzema comentaris d'usuaris/es sobre pel·lícules, amb informació de l'autor/a del comentari, ID de la pel·lícula, text del comentari i la data.

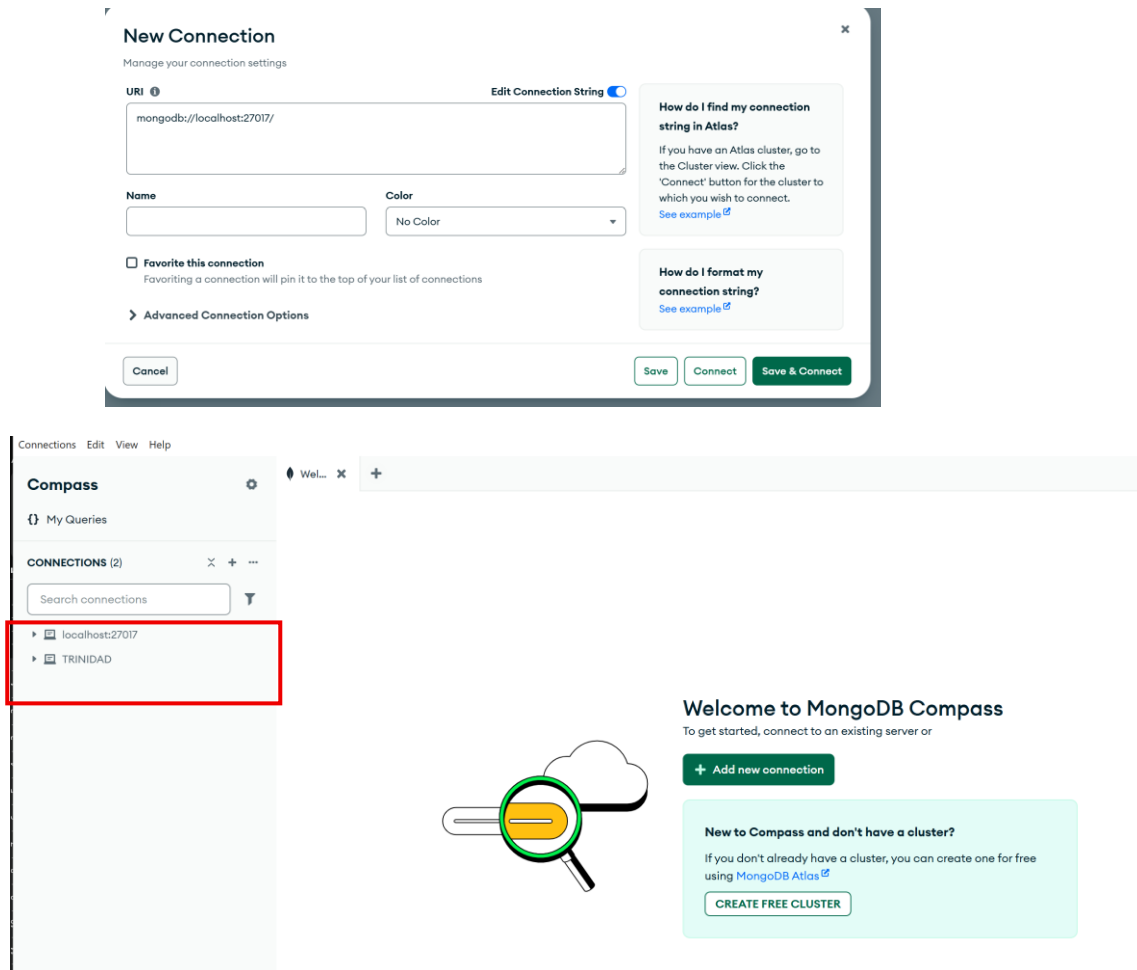
Duràs a terme algunes consultes que et demanda el client/a, el qual està mesurant si seràs capaç o no de fer-te càrrec de la part analítica del projecte vinculat amb la seva base de dades.

Instal·lació MongoDB per Windows:

1.- Anar a la pàgina oficial de MongoDB Community Server:

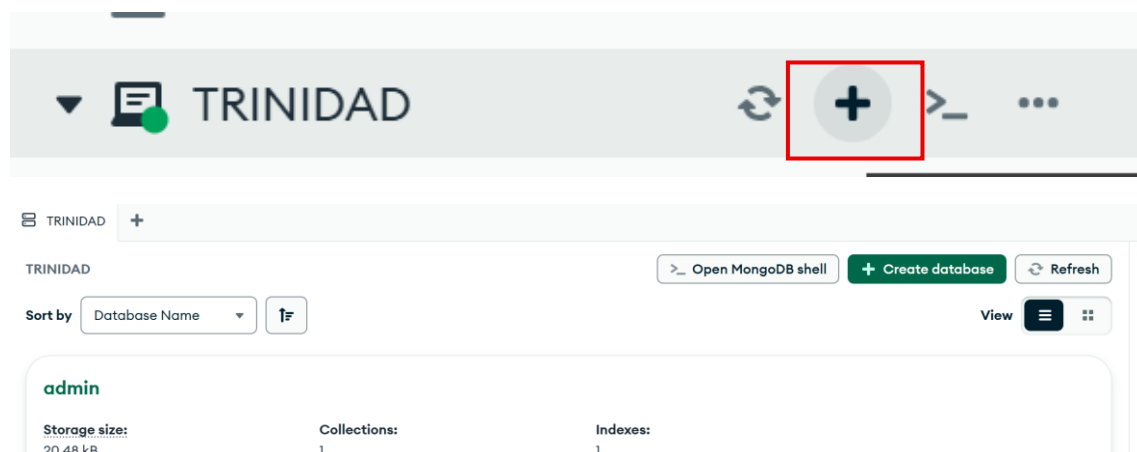
<https://www.mongodb.com/try/download/community>

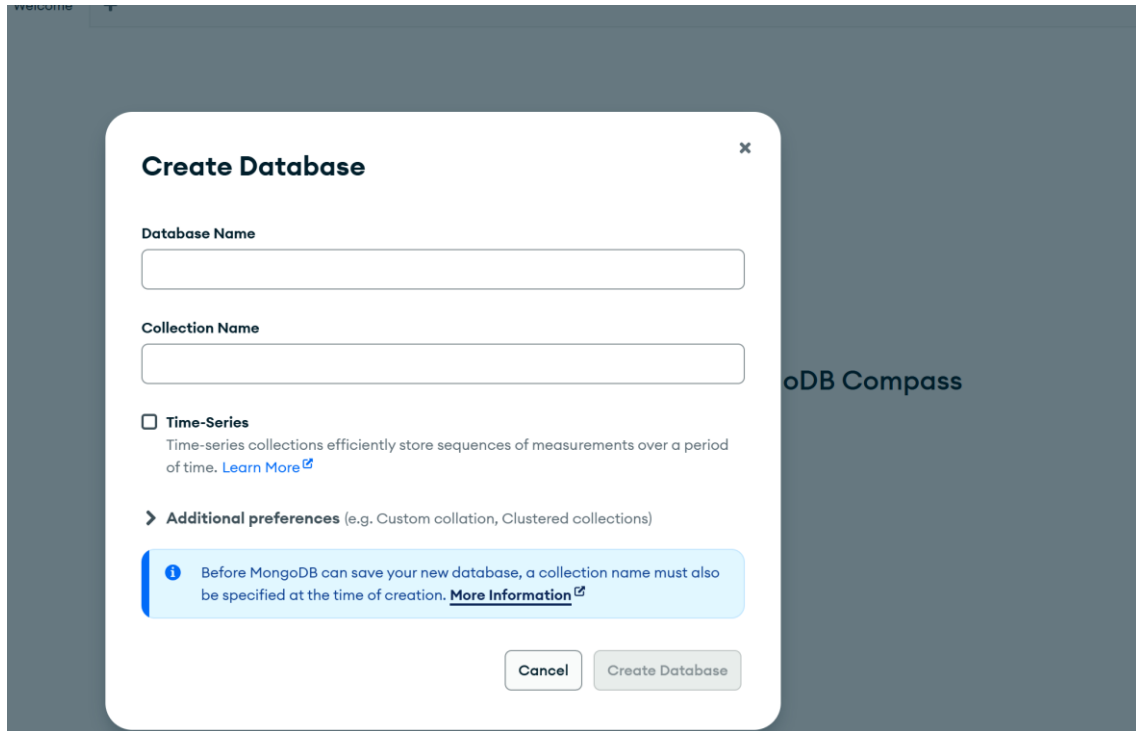
2.- Un cop instal·lat he creat una nova connexió anomenada Trinidad per fer els exercicis proposats en el sprint\_5.



El següent pas ha sigut crear una BD per l'entrenament cinematogràfic, les bases de dades es pot crear o bé, cliquem amb el símbol + , s'obrirà el quadre de diàleg.

La segona opció es utilitzar el botó CREATE DATABASE

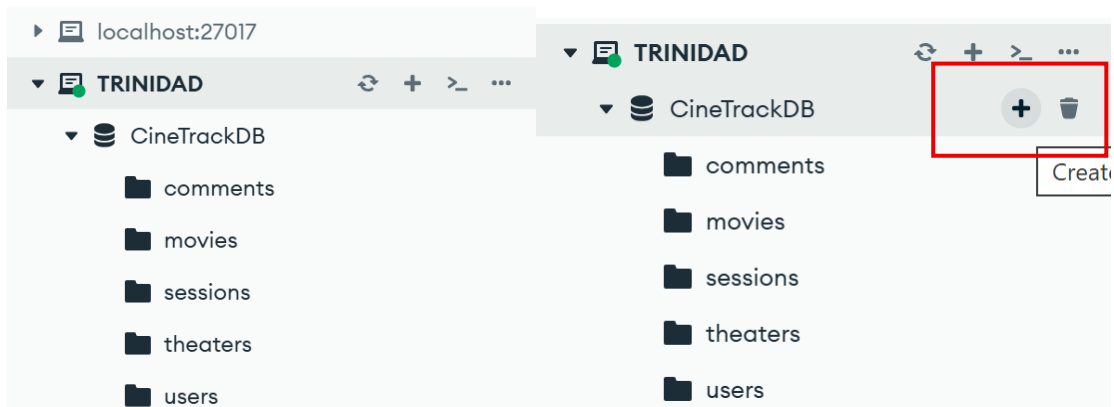




En el quadre de diàleg s'ha de posar el nom de la DB, que en aquest cas és CineTrackDB.

També hi ha l'opció d'importar la primera collecció, en mongoDB aquestes colleccions son les taules que es fan servir en les basedades relacionals.

La resta de les colleccions es carreguen des de el símbol +. Sortirà el quadre de diàleg per introduir el nom de la collecció, i poder carregar les dades des de la ubicació determina.



**Create Collection**

Collection Name:

☐ Time-Series  
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> Additional preferences (e.g. Custom collation, Clustered collections)

Cancel Create Collection

Per importar els registres en la taula creada s'ha de fer servir la pestanya de ADD DATE

Type a query: { field: 'valu

**ADD DATA** EXPORT DATA

Import JSON or CSV file

Insert document

La importació més comú de les dades es fer servir import JSON or CSV file. Es fa l'elecció de l'arxiu a importar, i s'importaran les dades.

Select JSON or CSV to import

Organizar Nueva carpeta

Nombre Estado

- comments
- movies
- sessions
- theaters
- users

TRINIDAD CineTrackDB sessions

Documents Aggregations Schema Indexes Validation

Type a query: { field: 'value' } or [Generate query](#)

EXPLAIN RESET FIND OPTIONS

ADD DATA EXPORT DATA UPDATE DELETE

25 1-1 of 1

```
{
  "_id": ObjectId("5a97f9c91c887bb9c6eb5fb4"),
  "user_id": "t3qulfeem@kwiv5.6ur",
  "jwt": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOiJlMTk5MDkzMjEsIm5iZiI6Im..."
}
```

# NIVELL 1

## N1EX\_1

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



```
>_ mongosh: TRINIDAD CineTrackDB TRINIDAD users movies comments >_ mongosh: TRINIDAD sessions theaters +
>_MONGOSH
> db.comments.find({}, { text: 1, _id: 0 }).sort({ createdAt: -1 }).limit(2).pretty()
< {
  text: 'Rem officiis eaque repellendus amet eos doloribus. Porro dolor voluptatum voluptates neque culpa molestias. Voluptate unde
}
{
  text: 'Tenetur dolorum molestiae ea. Eligendi praesentium unde quod porro. Commodi nisi sit placeat rerum vero cupiditate neque. D
}
CineTrackDB>
```

1.- find({},..) el primer paràmetre {} significa que no apliques cap filtre, selecciona tots els documents de la col·lecció comments.

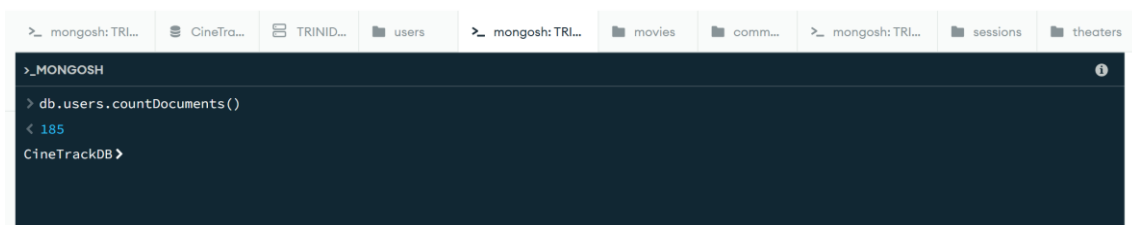
2.- Projecció {text:1, \_id:0}; text:1 mostra el camp text ; \_id:0 oculta el camp \_id, que per defecte MongoDB sempre mostra si no fas el procés d'ocultar.

3.- .sort({createdAt: -1}) ordena els resultats per el camp createdAt. -1 significa descendent (més recent primer), 1 significa ascendent (mes antic primer).

4.- .limit(2) limita el numero de documents només tornarà 2, en el nostre codi es veuen 2 comentaris mes recents.

5.- .pretty(), format de sortida al JSON més llegible en la consola.

### 1.2. Quants usuaris tenim registrats?



```
>_ mongosh: TRI... CineTra... TRINID... users >_ mongosh: TRI... movies comm... >_ mongosh: TRI... sessions theaters
>_MONGOSH
> db.users.countDocuments()
< 185
CineTrackDB>
```

1.- db.users: fa referencia a la col·lecció users.

2.- .countDocuments() fa el recomta de documents que hi ha en la col·lecció.

- 1.3. Quants cinemes hi ha en l'estat de Califòrnia? El resultat son 169 cines a l'estat de Califòrnia.

Consulta en format JSON

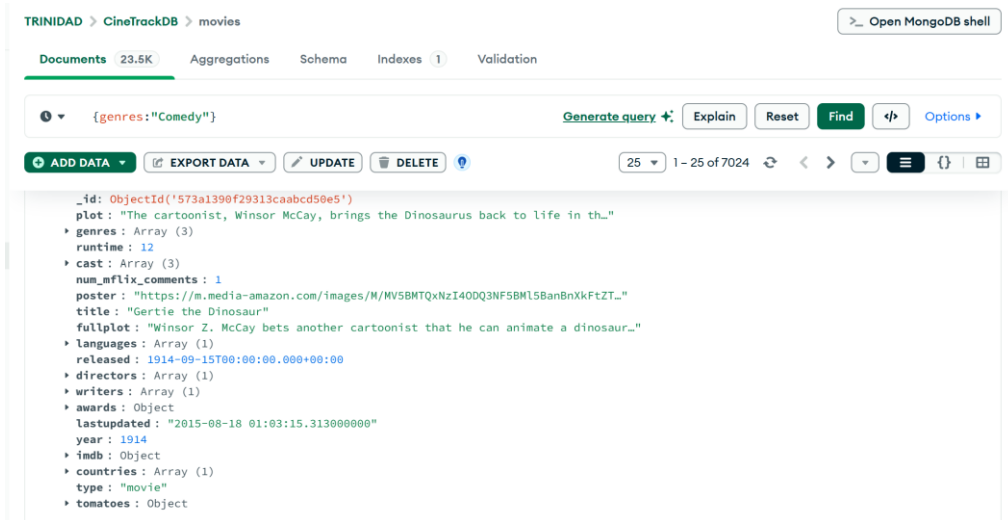
The screenshot shows the MongoDB Compass interface. At the top, the breadcrumb navigation is 'TRINIDAD > CineTrackDB > theaters'. Below this, there are tabs for 'Documents' (1.6K), 'Aggregations', 'Schema', 'Indexes' (1), and 'Validation'. A red box highlights the query editor area, which contains the query: `{ "location.address.state": "CA" }`. To the right of the query are buttons for 'Generate query', 'Explain', 'Reset', 'Find', and 'Options'. Below the query editor, there are buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. The main area displays the results of the query, showing three documents. Each document has a unique '\_id' (ObjectId), a 'theaterId' (number), and a 'location' object containing 'address' (street, city, state, zipcode) and 'geo' (type, coordinates).

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

```
> db.users.find().sort({ createdAt: 1 }).limit(1)
< {
  _id: ObjectId('59b99db4cfa9a34dcd7885b6'),
  name: 'Ned Stark',
  email: 'sean_bean@gameofthron.es',
  password: '$2b$12$UREFwsRUoyF0CRqGNK0Lz00HM/jLhgUCNNIJ9RJAqMUQ74cr1J1Vu'
}
```

- 1.- db.users fa referència a la col·lecció users dins de la base de dades CineTrackDB.
- 2.- find() obté tots els documents, és a dir, usuaris de la col·lecció.
- 3.- .sort({createdAt: 1}), per ordenar els resultats per camp createdAt en l'ordre ascendent, és a dir, del més antic al més recent, com ja s'ha comentat el valor 1 indica ordre ascendent, -1 indica ordre descendent.
- 4.- .limit(1), per limitar el resultat a només 1 document, en aquest cas, el primer usuari creat.

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



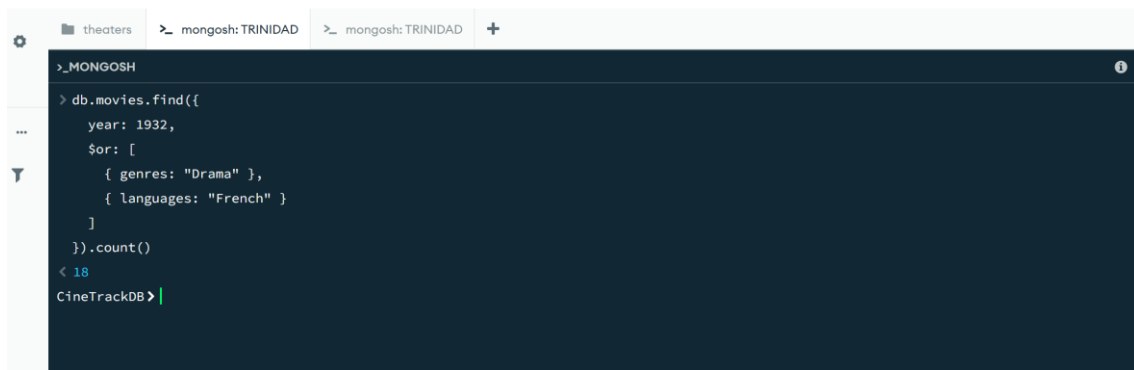
## Comprovació bash

```
>_MONGOSH
> use CineTrackDB
< switched to db CineTrackDB
> db["movies"].countDocuments({ genres: "Comedy" })
< 7024
```

N1EX\_2 Mostra'm tots els documents de les pel·lícules produïdes en 1932, però que el gènere sigui drama o estiguin en francès.

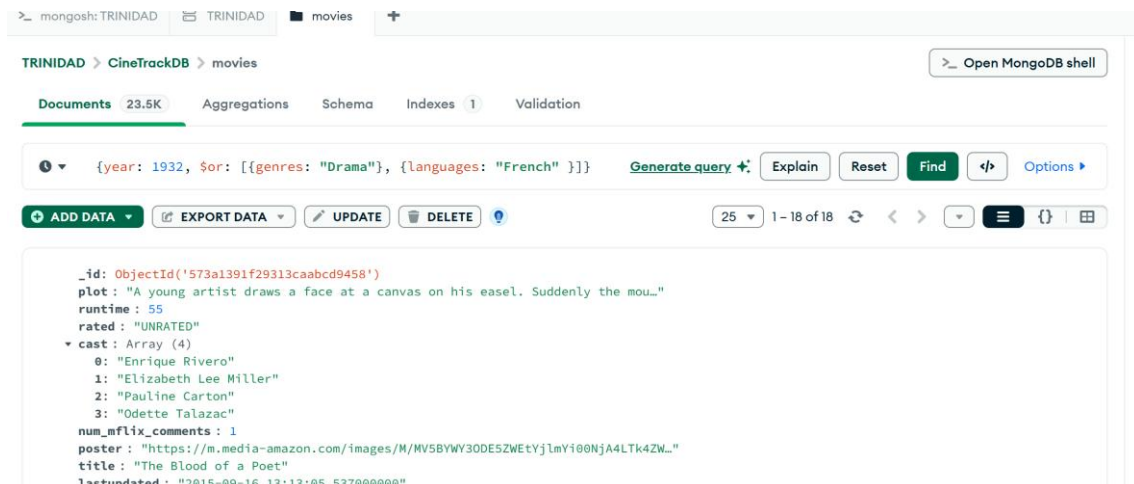
```
> db.movies.find({
  year: 1932,
  $or: [
    { genres: "Drama" },
    { languages: "French" }
  ]
})
< {
  _id: ObjectId('573a1391f29313caabcd9458'),
  plot: 'A young artist draws a face at a canvas on his easel. Suddenly the mouth on the drawing comes into life and starts talking.',
  runtime: 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/MV5BYWY3ODk5ZWVjYjlmYi00NjA4LTk4ZWYtMzBhZDE5MjY0YTYxXkEyXkFqcGdeQXVyNzI4MDMyMTU@._V1_...',
  title: 'The Blood of a Poet',
  lastupdated: '2015-09-16 13:13:05.537000000',
  languages: [
    'French'
  ],
}
```

Afegim un `.count()`, per comptar donar el numero de pel·lícules, produïdes en 1932, però que el gènere sigui drama o estiguin en francès.



```
>_MONGOSH
> db.movies.find({
  year: 1932,
  $or: [
    { genres: "Drama" },
    { languages: "French" }
  ]
}).count()
< 18
CineTrackDB>
```

Comprovació versió JSON:



TRINIDAD > CineTrackDB > movies

Documents 23.5K Aggregations Schema Indexes 1 Validation

{year: 1932, \$or: [{genres: "Drama"}, {languages: "French"}]}

Generate query Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1 - 18 of 18

```
{
  "_id": "573a1391f29313caabcd9458",
  "plot": "A young artist draws a face at a canvas on his easel. Suddenly the mou...",
  "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/MV5BYWY3ODE5ZWetYjlmYi00NjA4LTk4ZWw...",
  "title": "The Blood of a Poet",
  "lastupdated": "2015-09-16 13:13:05.537000000"
}
```

N1EX\_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.

Versió com a objecte JSON vàlid



The screenshot shows the MongoDB Compass interface. At the top, the breadcrumb navigation is 'TRINIDAD > CineTrackDB > movies'. There are tabs for 'Documents' (23.5K), 'Aggregations', 'Schema', 'Indexes' (1), and 'Validation'. A search bar contains a query: `{ $gte: 2012, $lte: 2014 }, "awards.wins": { $gte: 5, $lte: 9 }}`. Below the query bar are buttons for 'Generate query', 'Explain', 'Reset', 'Find', and 'Options'. A toolbar shows 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE' buttons, along with a pagination indicator '1 - 25 of 166'. The main area displays a JSON document for a movie:

```
{
  "_id": ObjectId('573a13acf29313caabd29366'),
  "fullplot": "The manager of the negative assets sector of Life magazine, Walter Mit...",
  "imdb": {
    "rating": 7.4,
    "votes": 211230,
    "id": 359950,
    "year": 2013,
    "plot": "When his job along with that of his co-worker are threatened, Walter t..."
  },
  "genres": Array (3)
    0: "Adventure"
    1: "Comedy"
    2: "Drama"
  "rated": "PG"
  "metacritic": 54
  "title": "The Secret Life of Walter Mitty"
  "lastupdated": "2015-08-31 00:10:51.747000000"
  "languages": Array (3)
    0: "English"
    1: "Spanish"
    2: "Icelandic"
  "writers": Array (3)
    0: "Steve Conrad (screenplay)"
    1: "Steve Conrad (screen story by)"
    2: "James Thurber (based on the short story by)"
  "type": "movie"
  "tomatoes": {
    "website": "http://WalterMitty.com"
  }
  "viewer": {
    "rating": 3.7,
    "numReviews": 77663
  }
}
```

Codi:

1.- Countries: "USA" busca pel·lícules fetes als Estats Units, countries es una array (una llista), i aquí es prova si "USA" esta o no en la llista.

2.- Year: {\$gte: 2012, \$lte:2014}

\$gte vols dir "mes gran o igual que" 2012

\$lte vols dir "menys o igual que" 2014.

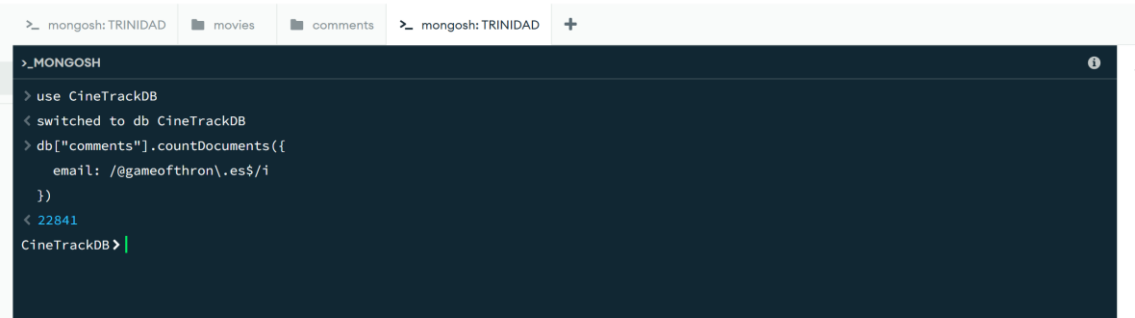
Per tant, inclou 2012, 2013, 2014.

3.- "awards.wins"; {\$gte: 5, \$lte:9}, pel·lícules que han guanyat entre 5 i 9 premis.

Estem accedint a un camp que està dins d'un altre camp: awards és un objecte, wins és un número dins de awards.

## NIVELL 2

N2\_EX1: Compte quants comentaris escriu un usuari/ària que utilitza "GAMEOFTHRON.ES" com a domini de correu electrònic.

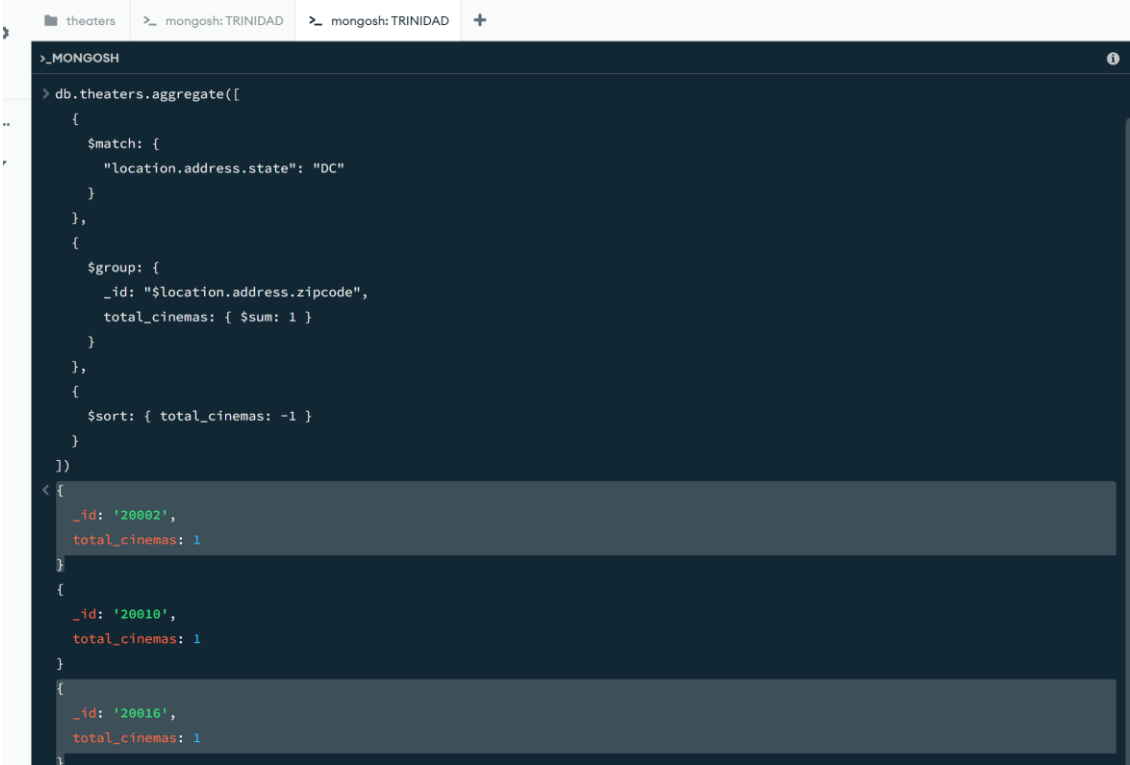


- 1.-db.comments.find() per cerca a la collecció comments
  - 2.-email: /@gameofthron\.es\$/i es l'expressió per filtrar correus que acaben amb el nom de l'usuari, @gameofthron.es.
- I fa que sigui insensible a majúscules/minúscules.



Entorn	Síntaxi correcta
Mongo shell	db.comments.countDocuments({ email: /@gameofthron\.es\$/i })
JSON pur (API/Postman)	{ "email": { "\$regex": "@gameofthron\\.es\$", "\$options": "i" } }

N2\_EX2: Quants cinemes hi ha en cada codi postal situats dins de l'estat Washington D. C. (DC)?



The screenshot shows a MongoDB terminal window with the following content:

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

El resulta és l'agregació de cinemes agrupats per codi postal dins de Washington D.C.

\_id es el valor del camp zipcode que en aquest cas 20002", "20010", "20016", que son els codis postals de Washington DC.

Total\_cinemas es el recompte de quants cines hi ha registrats dins del codis postals.

En el nostre cas, només indica 1 per cadascun del codis postals registrats en Washington D.C.

## NIVELL 3

N3\_EX1: Troba totes les pel·lícules dirigides per John Landis amb una puntuació IMDb (Internet Movie Database) d'entre 7,5 i 8.

```
> use CineTrackDB
< already on db CineTrackDB
> db.movies.find(
  {
    "directors": "John Landis",
    "imdb.rating": { $gte: 7.5, $lte: 8 }
  },
  {
    title: 1,
    year: 1,
    "imdb.rating": 1,
    _id: 0
  }
)
< {
  imdb: {
    rating: 7.6
  },
  year: 1978,
  title: 'Animal House'
}
{
  title: 'The Blues Brothers',
  year: 1980,
  imdb: {
    rating: 7.9
  }
}
{
```

```
> db.movies.find(
  {
    "directors": "John Landis",
    "imdb.rating": { $gte: 7.5, $lte: 8 }
  },
  {
    title: 1,
    year: 1,
    "imdb.rating": 1,
    _id: 0
  }
).count()
< 4
```

“directors”: “John Landis”, busca

"directors": "John Landis" → busca només les pel·lícules dirigides per John Landis.

"imdb.rating": { \$gte: 7.5, \$lte: 8 } → filtra només aquelles amb una puntuació IMDb entre 7.5 i 8.0 (inclosos els extrems).

El segon objecte és la projecció → només mostrarem title, year i imdb.rating.

N3\_EX2 Mostra en un mapa la ubicació de tots els teatres de la base de dades.

```

>_ mongosh: TRINIDAD  users  movies  comments  theaters  >_ mongosh: TRINIDAD  >_ mongosh: TRINIDAD  +
>_MONGOSH
> db.theaters.find(
  {},
  {
    _id: 0,
    theaterId: 1,
    "location.address.street1": 1,
    "location.address.city": 1,
    "location.address.state": 1,
    "location.geo.coordinates": 1
  }
).toArray()
< [
  { theaterId: 1000, location: { address: [Object], geo: [Object] } },
  { theaterId: 1003, location: { address: [Object], geo: [Object] } },
  { theaterId: 1008, location: { address: [Object], geo: [Object] } },
  { theaterId: 1004, location: { address: [Object], geo: [Object] } },
  { theaterId: 1002, location: { address: [Object], geo: [Object] } },
  { theaterId: 1010, location: { address: [Object], geo: [Object] } },
  { theaterId: 1014, location: { address: [Object], geo: [Object] } },
  { theaterId: 1012, location: { address: [Object], geo: [Object] } },
  { theaterId: 1009, location: { address: [Object], geo: [Object] } },
  { theaterId: 1013, location: { address: [Object], geo: [Object] } },
  { theaterId: 1017, location: { address: [Object], geo: [Object] } },
  { theaterId: 1018, location: { address: [Object], geo: [Object] } },
  { theaterId: 1015, location: { address: [Object], geo: [Object] } },
  { theaterId: 1011, location: { address: [Object], geo: [Object] } },
  { theaterId: 1019, location: { address: [Object], geo: [Object] } },
  { theaterId: 101, location: { address: [Object], geo: [Object] } },
  { theaterId: 1021, location: { address: [Object], geo: [Object] } },
  { theaterId: 1093, location: { address: [Object], geo: [Object] } },
  { theaterId: 1104, location: { address: [Object], geo: [Object] } },
  { theaterId: 1092, location: { address: [Object], geo: [Object] } },
  { theaterId: 10, location: { address: [Object], geo: [Object] } },
  { theaterId: 1106, location: { address: [Object], geo: [Object] } },
  { theaterId: 1107, location: { address: [Object], geo: [Object] } },
  { theaterId: 1109, location: { address: [Object], geo: [Object] } },
  { theaterId: 1108, location: { address: [Object], geo: [Object] } },
  { theaterId: 1102, location: { address: [Object], geo: [Object] } },
  { theaterId: 1113, location: { address: [Object], geo: [Object] } },
  { theaterId: 1115, location: { address: [Object], geo: [Object] } },
  { theaterId: 1087, location: { address: [Object], geo: [Object] } },
  { theaterId: 1110, location: { address: [Object], geo: [Object] } },
  ... 1464 more items
]

```