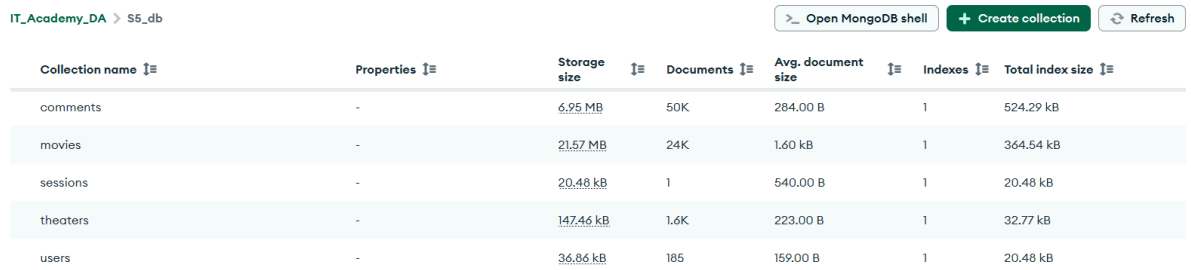
Tasca S5.01. Consultes amb MongoDB

Nivell 1

Exercici 1

Creamos la base de datos con los archivos JSON correspondientes, para poder trabajar con ellos posteriormente (use S5_db).



IT_Academy_DA > S5_db

Open MongoDB shell | + Create collection | Refresh

Collection name	Properties	Storage size	Documents	Avg. document size	Indexes	Total index size
comments	-	6.95 MB	50K	284.00 B	1	524.29 kB
movies	-	21.57 MB	24K	1.60 kB	1	364.54 kB
sessions	-	20.48 kB	1	540.00 B	1	20.48 kB
theaters	-	147.46 kB	1.6K	223.00 B	1	32.77 kB
users	-	36.86 kB	185	159.00 B	1	20.48 kB

Mostramos los 2 primeros comentarios de la base de datos, considerando que se refiere a más antiguos. (db.comments.find().sort({date: 1}).limit(2))

```
>_MONGOSH
> db.comments.find().sort({date: 1}).limit(2)
< {
    _id: ObjectId('5a9427648b0beebeb695e67e'),
    name: 'Mercedes Tyler',
    email: 'mercedes_tyler@fakegmail.com',
    movie_id: ObjectId('573a1399f29313caabcec3ce'),
    text: 'Optio totam dolores magni. Enim ratione fuga tempora voluptatum est cumque animi.
    date: 1970-01-01T01:07:09.000Z
  }
  {
    _id: ObjectId('5a9427658b0beebeb6969efe'),
    name: 'Jon Snow',
    email: 'kit_harington@gameofthron.es',
    movie_id: ObjectId('573a13a4f29313caabd117df'),
    text: 'Dolorem animi tempora ullam quas. Iusto nobis reprehenderit aspernatur cupiditate
    date: 1970-01-01T09:37:49.000Z
  }
S5_db>
```

Mostramos el número de usuarios registrados. (db.users.countDocuments())

```
>_MONGOSH
> db.users.countDocuments()
< 185
S5_db>
```

Mostramos cuantos cines hay en California. (db.theaters.countDocuments({ "location.address.state": "CA" }))

```
>_MONGOSH
> db.theaters.countDocuments({ "location.address.state": "CA" })
< 169
S5_db>
```

Como la colección de usuarios no tiene fecha de registro, consideramos que para hacer comentarios requiere registrarse, por lo que el usuario del comentario más antiguo lo consideramos el primer usuario en registrarse. (db.comments.find({},{name: 1, email: 1, date: 1}).sort({date: 1}).limit(1))

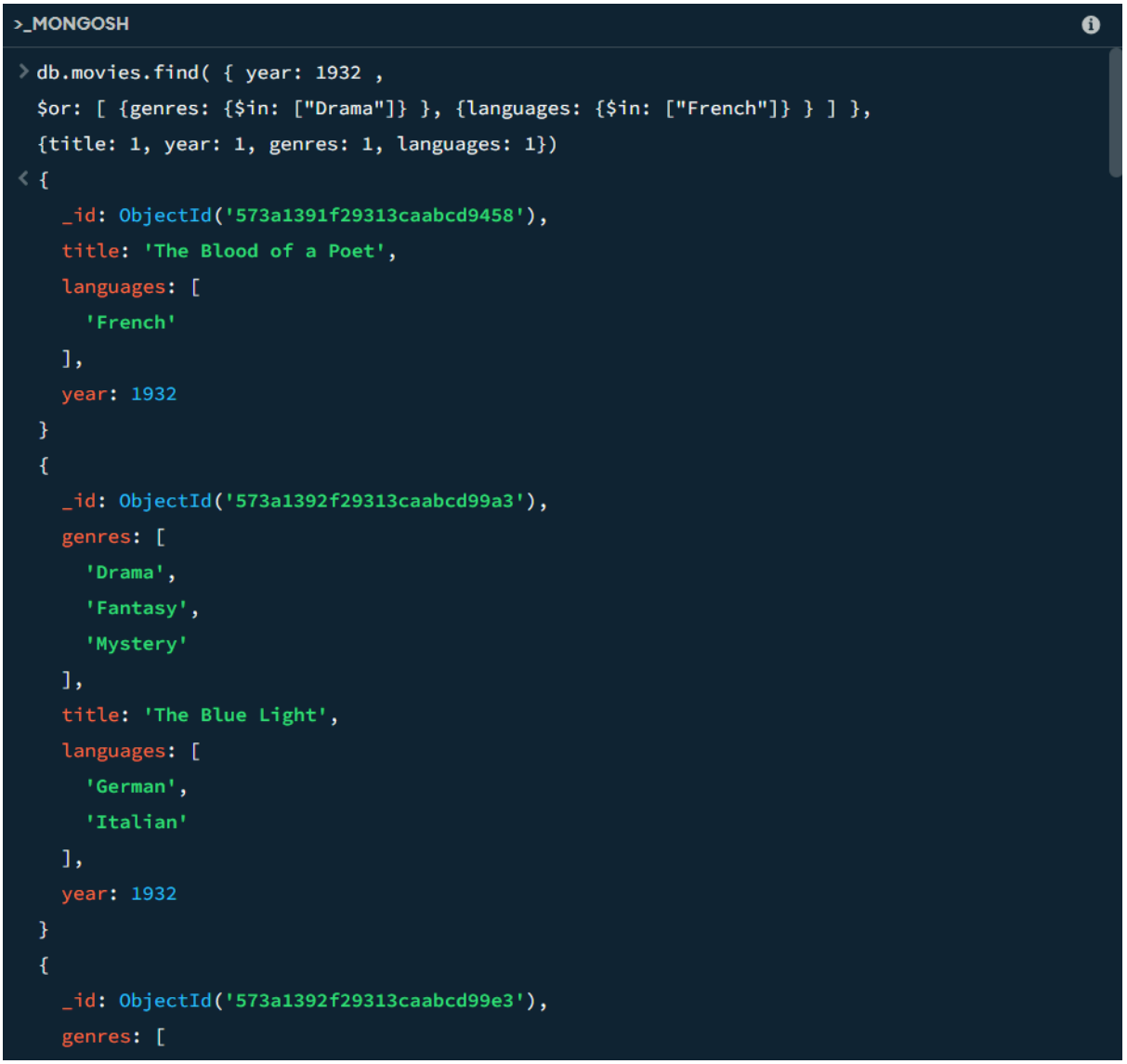
```
>_MONGOSH
> db.comments.find({},{name: 1, email: 1, date: 1}).sort({date: 1}).limit(1)
< {
    _id: ObjectId('5a9427648b0beebeb695e67e'),
    name: 'Mercedes Tyler',
    email: 'mercedes_tyler@fakegmail.com',
    date: 1970-01-01T01:07:09.000Z
  }
S5_db>
```

Mostramos el número de películas de comedia. (db.movies.countDocuments({ genres: { $in: ["Comedy"] } }))

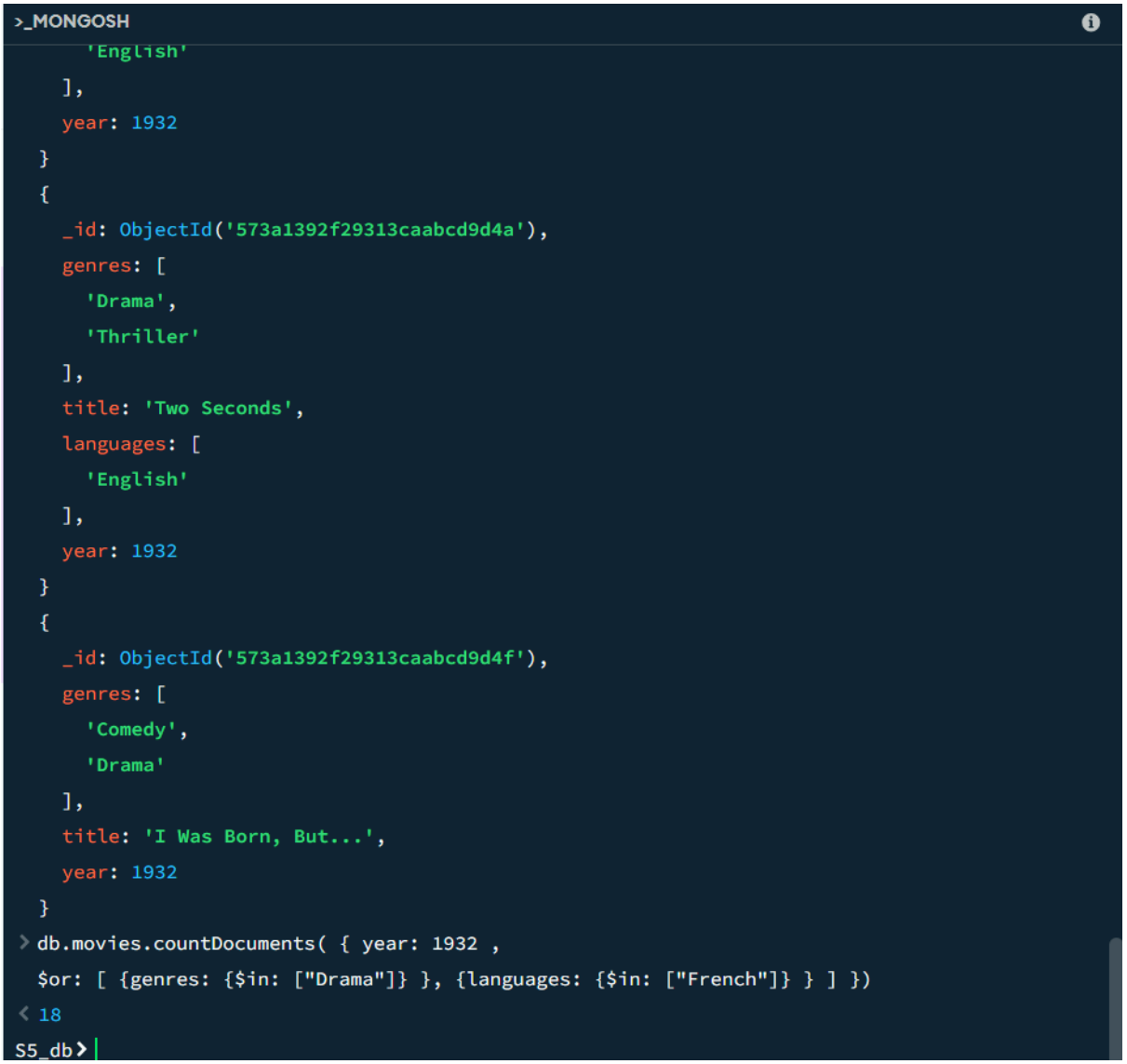
```
>_MONGOSH
> db.movies.countDocuments({ genres: { $in: ["Comedy"] } })
< 7024
S5_db>
```

Exercici 2

Utilizamos:
db.movies.find({ year: 1932 ,
$or: [{genres: {$in: ["Drama"]} }, {languages: {$in: ["French"]} }] },
{title: 1, year: 1, genres: 1, languages: 1})



Calculamos cuantos documentos son: db.movies.countDocuments({ year: 1932 , $or: [{genres: {$in: ["Drama"]} }, {languages: {$in: ["French"]} }] })



Exercici 3

Utilizamos:
db.movies.find({ countries: {$in: ["USA"]} ,
"awards.wins": { $gte: 5, $lte: 9 },
year: { $gte: 2012, $lte: 2014 } },
{title: 1, countries: 1, "awards.wins": 1, year: 1})

```
>_MONGOSH
> db.movies.find({ countries: {$in: ["USA"]} ,
  "awards.wins": { $gte: 5, $lte: 9 },
  year: { $gte: 2012, $lte: 2014 } },
  {title: 1, countries: 1, "awards.wins": 1, year: 1})
< {
    _id: ObjectId('573a13acf29313caabd29366'),
    year: 2013,
    title: 'The Secret Life of Walter Mitty',
    awards: {
      wins: 6
    },
    countries: [
      'USA',
      'Canada'
    ]
  }
  {
    _id: ObjectId('573a13b5f29313caabd45772'),
    title: 'The Croods',
    awards: {
      wins: 8
    },
    year: 2013,
    countries: [
      'USA'
    ]
  }
  {
```

Contamos cuantos documentos son: db.movies.countDocuments({ countries: {$in: ["USA"]} ,
"awards.wins": { $gte: 5, $lte: 9 },
year: { $gte: 2012, $lte: 2014 } })

```
> db.movies.countDocuments({ countries: {$in: ["USA"]} ,
  "awards.wins": { $gte: 5, $lte: 9 },
  year: { $gte: 2012, $lte: 2014 } })
< 166
S5_db>
```

Nivell 2
Exercici 1

Calculamos el número de comentarios que cumple lo requerido. Comparamos el resultado con un problema similar.
db.comments.countDocuments({ email: { $regex: /GAMEOFTHRON.ES$/ } })
db.comments.countDocuments({ email: { $regex: /gameofthron.es$/ } })

```
>_MONGOSH
> db.comments.countDocuments({ email: { $regex: /GAMEOFTHRON.ES$/ } })
< 0
> db.comments.countDocuments({ email: { $regex: /gameofthron.es$/ } })
< 22841
S5_db>
```

Exercici 2

Para saber este resultado, utilizamos un aggregate(), el cual primero solo elige los documentos dentro del estado de DC, agrupa los resultados por zipcode y suma cuantas veces aparece cada zipcode y finalmente los ordena de forma decreciente.
db.theaters.aggregate([
{ $match: { "location.address.state": "DC" } },
{ $group: { _id: "$location.address.zipcode", total: { $sum: 1 } } },
{ $sort: { total: -1 } }])

```
>_MONGOSH
> db.theaters.aggregate( [
    { $match: { "location.address.state": "DC" } },
    { $group: { _id: "$location.address.zipcode", total: { $sum: 1 } } },
    { $sort: { total: -1 } } ] )
< {
    _id: '20002',
    total: 1
  }
  {
    _id: '20010',
    total: 1
  }
  {
    _id: '20016',
    total: 1
  }
S5_db>
```

Nivell 3
Exercici 1

Mostramos los resultados de la búsqueda indicada:
db.movies.find({ directors: {$in: ["John Landis"]},
"imdb.rating": { $gte: 7.5, $lte: 8 } } ,
{title: 1, directors: 1, "imdb.rating": 1})

```
>_MONGOSH
> db.movies.find( { directors:  {$in: ["John Landis"]},
  "imdb.rating": { $gte: 7.5, $lte: 8 } } ,
  {title: 1, directors: 1,  "imdb.rating": 1})
< {
    _id: ObjectId('573a1397f29313caabce6d94'),
    imdb: {
      rating: 7.6
    },
    title: 'Animal House',
    directors: [
      'John Landis'
    ]
  }
  {
    _id: ObjectId('573a1397f29313caabce76f7'),
    title: 'The Blues Brothers',
    directors: [
      'John Landis'
    ],
    imdb: {
      rating: 7.9
    }
  }
  {
    _id: ObjectId('573a1397f29313caabce7d06'),
    imdb: {
      rating: 7.6
    },
```

Calculamos cuantos documentos son:
db.movies.countDocuments({ directors: {$in: ["John Landis"]},
"imdb.rating": { $gte: 7.5, $lte: 8 } })

Exercici 2

Para mostrar el mapa, no pudimos utilizar el Schema de MongoDB Compass porque solo considera 1000 documentos, y hay 1564 en total. Debido a eso, decidimos extraer las coordenadas de cada teatro en un documento JSON (db.theaters.find({},{ "location.geo.coordinates": 1 })), que posteriormente usamos en PowerBI para la elaboración de un mapa.

Se intentó hacer la conexión de PowerBI con MongoDB, pero no se logró elaborar todos los pasos. Creemos que la única manera de hacerlo ahora es pagando (MongoDB Enterprise Advanced subscription, ver https://www.mongodb.com/try/download/bi-connector).

IT_Academy_DA > S5_db > theaters

Open MongoDB shell

Documents 1.6K | Aggregations | Schema | Indexes 1 | Validation

Type a query: { field: 'value' } or Generate query

Explain | Reset | Find | </> | Options

Project { "location.geo.coordinates": 1 }

Sort { field: -1 } or [['field', -1]]

Max Time MS 60000

Collation { locale: 'simple' }

Skip 0 Limit 0

Index Hint { field: -1 }

EXPORT DATA

25 1 - 25 of 1564

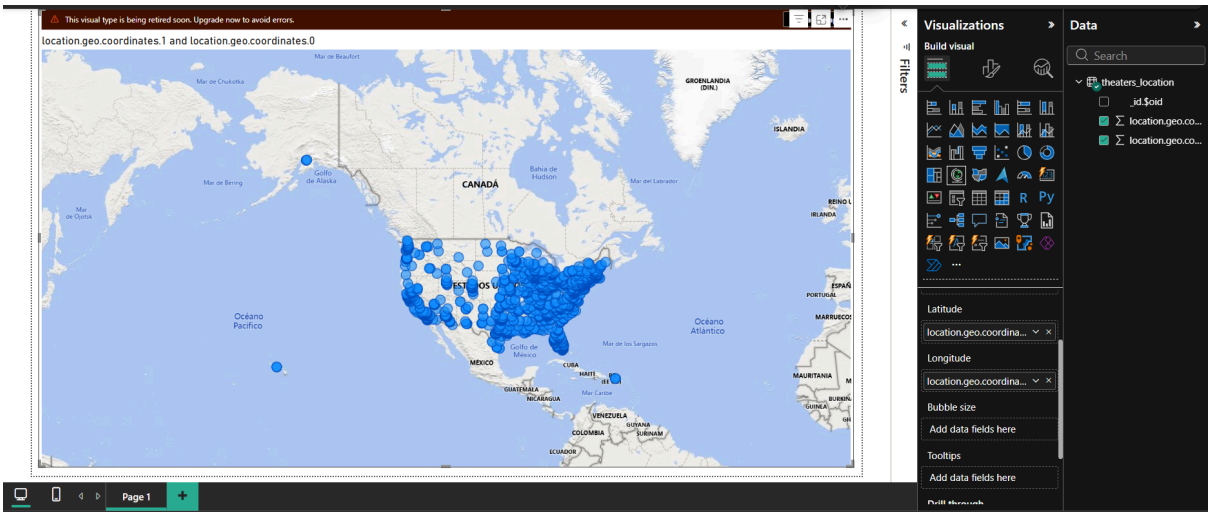
```
_id: ObjectId('59a47286cfa9a3a73e51e72c')
location : Object

_id: ObjectId('59a47286cfa9a3a73e51e72d')
location : Object

_id: ObjectId('59a47286cfa9a3a73e51e72e')
location : Object

_id: ObjectId('59a47286cfa9a3a73e51e72f')
location : Object

_id: ObjectId('59a47286cfa9a3a73e51e730')
location : Object
```



Revisión peer-to-peer
Revisado por Minu Campoy

- Si la colección de users tuviera la información, se podría usar db.users.sort({ createdAt: 1 }).limit(1).pretty().
- Se podria usar pretty().