



MÁSTER BIG DATA Y DATA SCIENCE

Evaluación bases de datos NoSQL

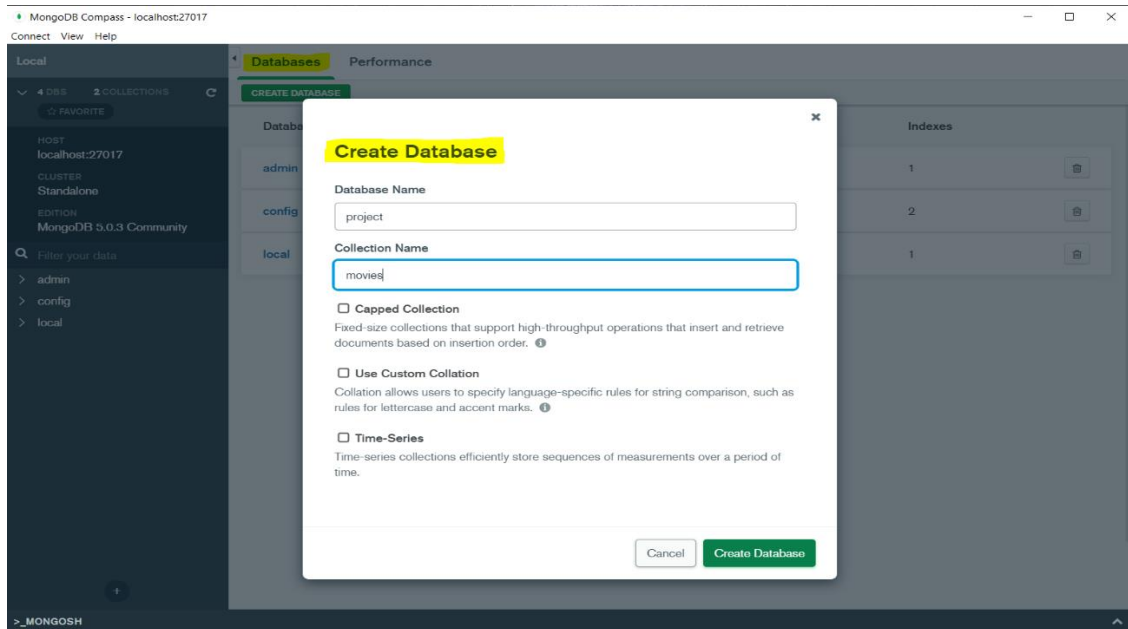
MÓDULO: Not Only Structured Query Language (NoSQL)

PROFESOR: Álvaro Bravo

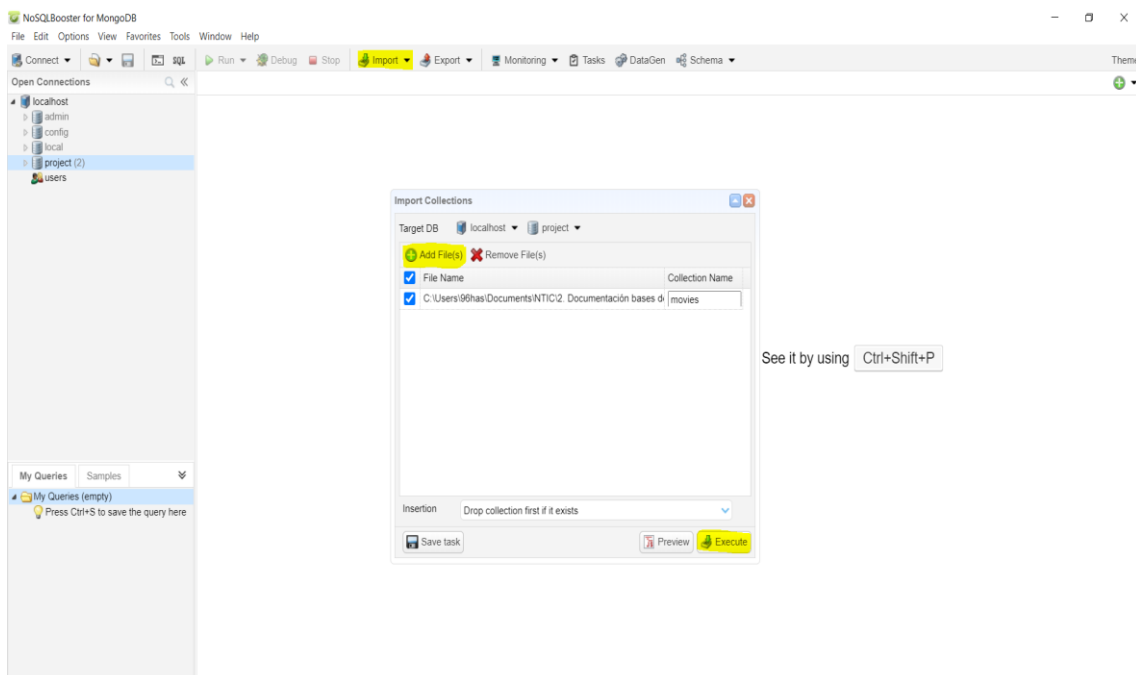
AUTOR: Hassan Chafi Xavier

0. Realizar la importación del json en una colección llamada “movies”.

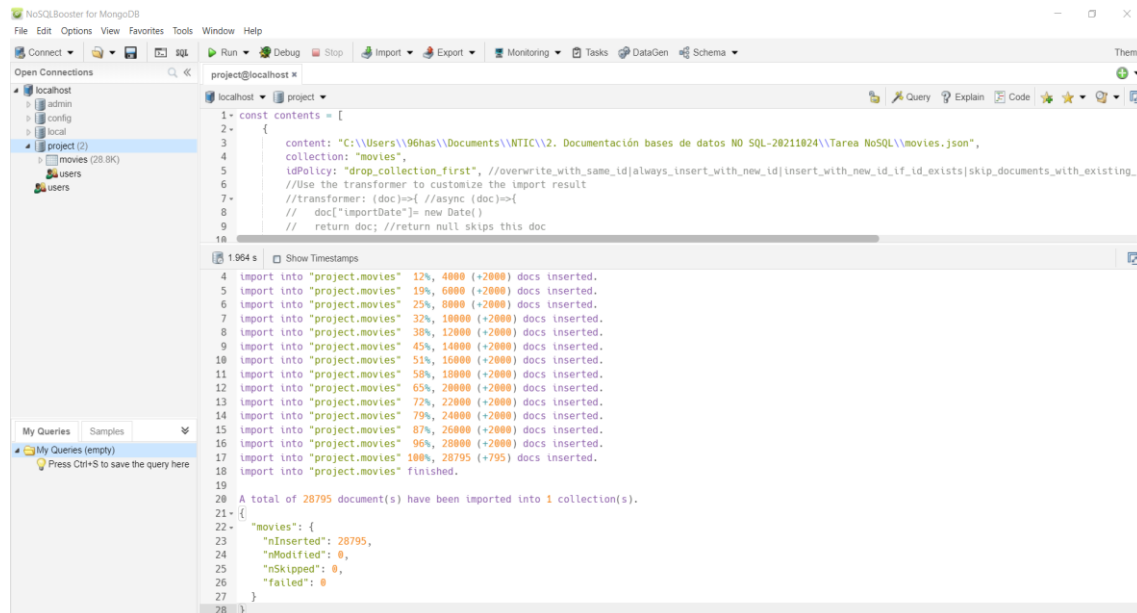
Se crea la base de datos con la herramienta MongoDB Compass que, en este caso, llevará el nombre de “project” y la colección en la que se importarán los datos que tendrá el nombre “movies”.



A continuación, se importan los datos contenidos en el archivo del dataset dado que tiene el nombre de movies.json en NoSQLBooster for MongoDB:



Una vez terminada la ejecución, se indica la exitosa importación de los 28.795 documentos en la colección.



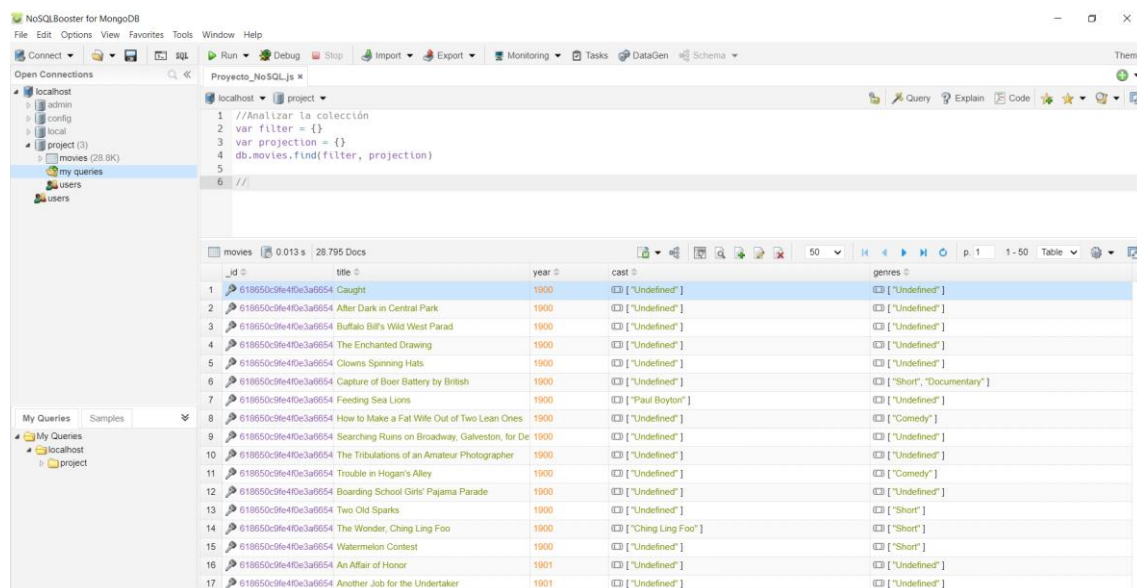
1. Analizar con find la colección.

Al analizar la colección devuelve como resultado 28.795 películas que coincide con el número de documentos cargados en la colección previamente.

```
var filter = {}
```

```
var projection = {}
```

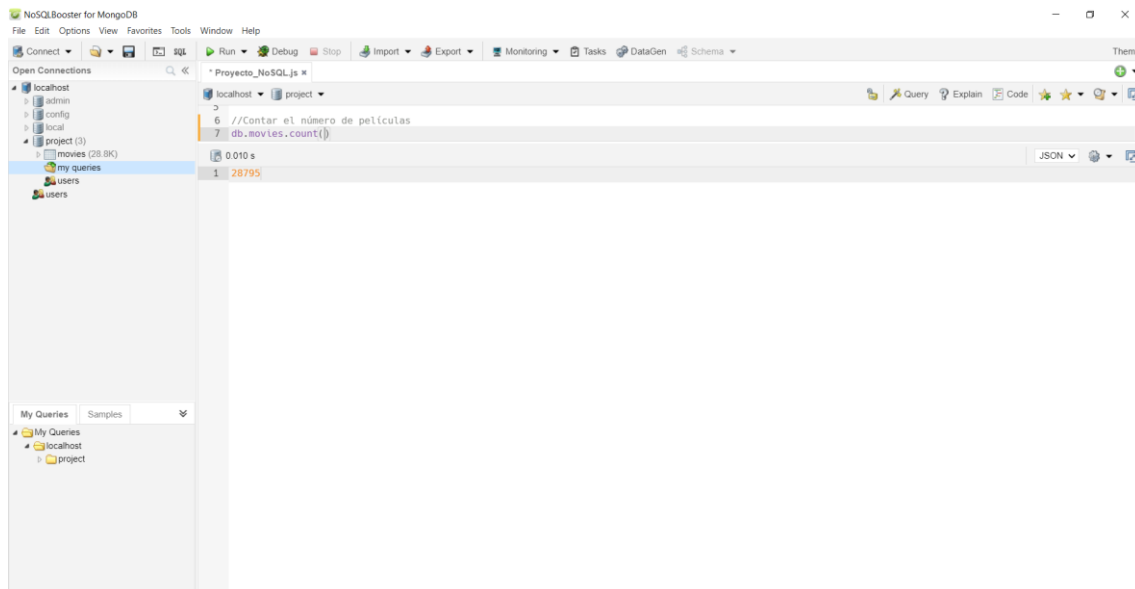
```
db.movies.find(filter, projection)
```



2. Contar cuántos documentos (películas) tiene cargado.

El conteo del número total de películas coincide con los 28.795 documentos importados en la colección.

db.movies.count()



3. Insertar una película.

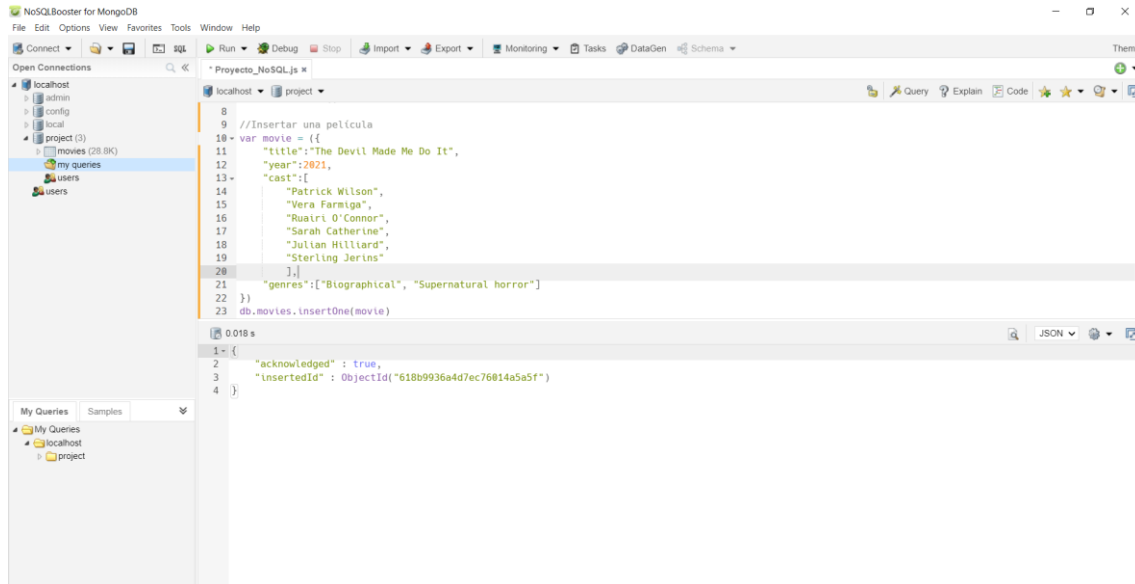
Se inserta una película dentro de la colección de la que se dispone.

```
var movie = ({
  "title": "The Devil Made Me Do It",
  "year": 2021,
  "cast": [
    "Patrick Wilson",
    "Vera Farmiga",
    "Ruairi O'Connor",
    "Sarah Catherine",
    "Julian Hilliard",
    "Sterling Jerins"
  ],
```

"genres":["Biographical", "Supernatural horror"]

}}

db.movies.insertOne(movie)



The screenshot shows the NoSQLBooster for MongoDB interface. On the left, the 'Open Connections' panel shows a connection to 'localhost' with a 'project' database. The 'My Queries' panel shows a query for 'project'. The main editor displays a JavaScript script for inserting a movie document. The script defines a movie object with title 'The Devil Made Me Do It', year 2021, a cast of five actors, and genres 'Biographical' and 'Supernatural horror'. It then calls db.movies.insertOne(movie). The output panel shows the result of the insert operation, which was successful, returning an acknowledged status of true and an inserted ID.

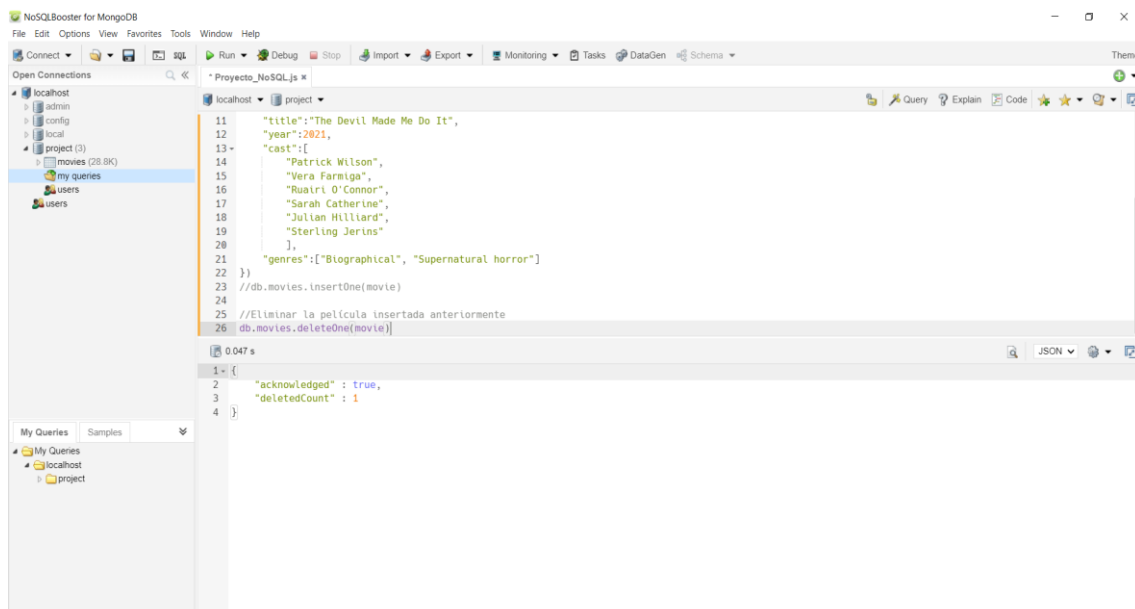
```
8
9 //Insertar una película
10 var movie = {
11   "title": "The Devil Made Me Do It",
12   "year": 2021,
13   "cast": [
14     "Patrick Wilson",
15     "Vera Farmiga",
16     "Ruairi O'Connor",
17     "Sarah Catherine",
18     "Julian Hilliard",
19     "Sterling Jerins"
20   ],
21   "genres": ["Biographical", "Supernatural horror"]
22 }
23 db.movies.insertOne(movie)
```

```
1 {
2   "acknowledged": true,
3   "insertedId": ObjectId("618b9936a4d7ec76814a5a5f")
4 }
```

4. Borrar la película insertada en el punto anterior (en el 3).

Se elimina la película insertada en el punto anterior.

db.movies.deleteOne(movie)



The screenshot shows the NoSQLBooster for MongoDB interface. The main editor displays a JavaScript script that includes the insertOne operation from the previous step, followed by a deleteOne operation. The deleteOne operation calls db.movies.deleteOne(movie). The output panel shows the result of the delete operation, which was successful, returning an acknowledged status of true and a deleted count of 1.

```
11   "title": "The Devil Made Me Do It",
12   "year": 2021,
13   "cast": [
14     "Patrick Wilson",
15     "Vera Farmiga",
16     "Ruairi O'Connor",
17     "Sarah Catherine",
18     "Julian Hilliard",
19     "Sterling Jerins"
20   ],
21   "genres": ["Biographical", "Supernatural horror"]
22 }
23 //db.movies.insertOne(movie)
24
25 //Eliminar la película insertada anteriormente
26 db.movies.deleteOne(movie)
```

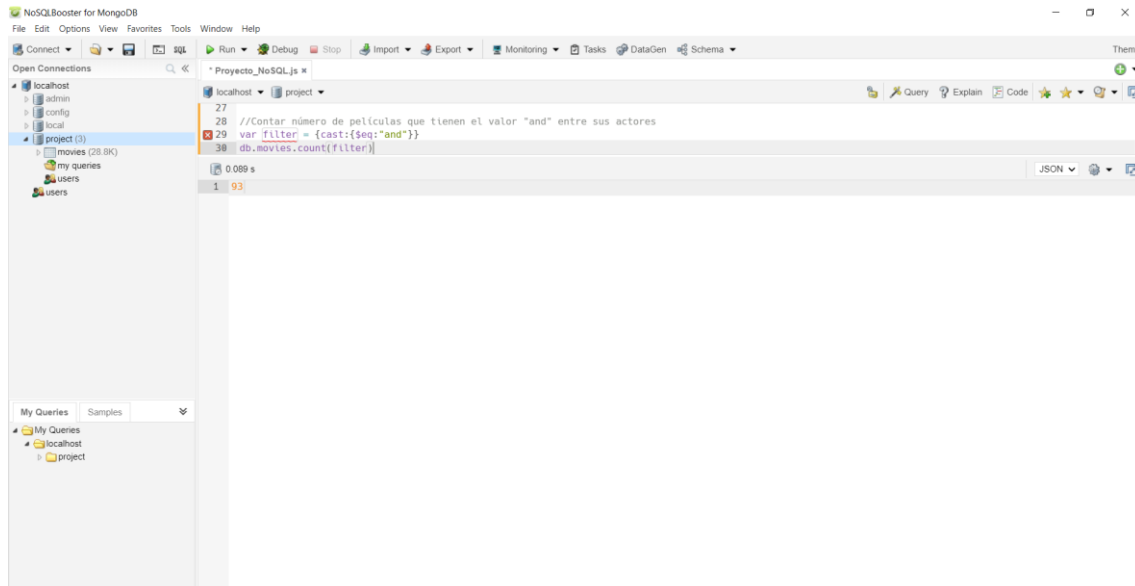
```
1 {
2   "acknowledged": true,
3   "deletedCount": 1
4 }
```

5. Contar cuantas películas tienen actores (cast) que se llaman “and”. Estos nombres de actores están por ERROR.

Se cuenta el número de películas que figura el valor “and” entre sus actores.

```
var filter = {cast:{$eq:"and"}}
```

```
db.movies.count(filter)
```



6. Actualizar los documentos cuyo actor (cast) tenga por error el valor “and” como si realmente fuera un actor. Para ello, se debe sacar únicamente ese valor del array cast. Por lo tanto, no se debe eliminar ni el documento (película) ni su array cast con el resto de los actores.

```
var filter = {cast:{$eq:"and"}}
```

```
var update = {$pull:{cast:{$eq:"and"}}}
```

```
var options = {multi:true}
```

```
db.movies.update(filter, update, options)
```

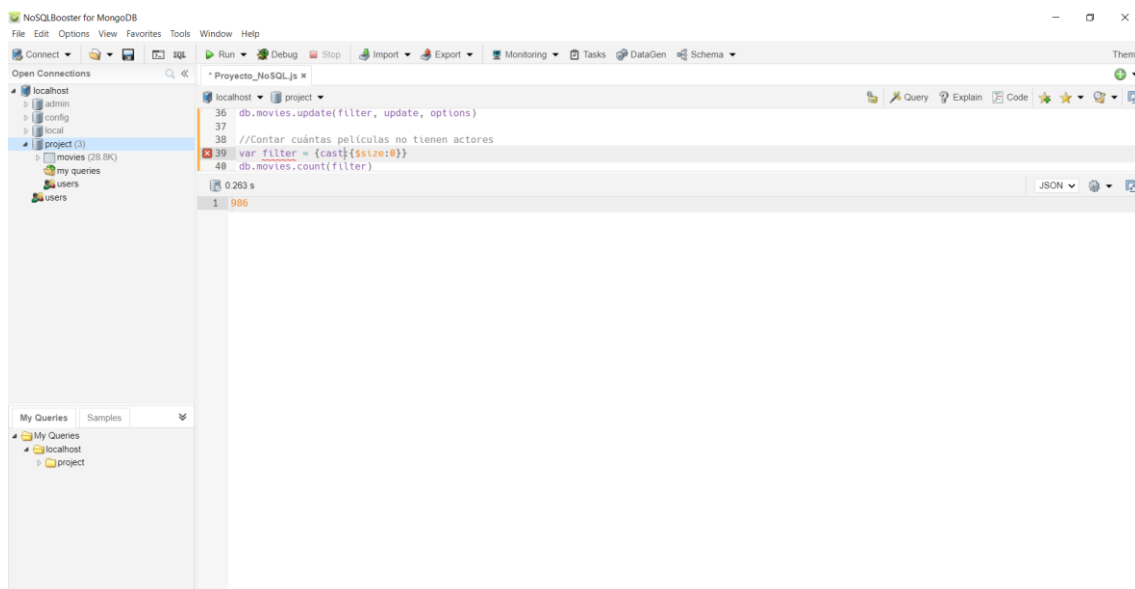


7. Contar cuantos documentos (películas) no tienen actores (array cast).

Se cuentan las películas que no tienen actores (array cast vacío) contando los arrays cast de tamaño cero.

```
var filter = {"cast":{$size:0}}
```

```
db.movies.count(filter)
```



8. Actualizar TODOS los documentos (películas) que no tengan actores (array cast), añadiendo un nuevo elemento dentro del array con valor Undefined. ¡Cuidado! El tipo de cast debe seguir siendo un array. El array debe ser así -> ["Undefined"].

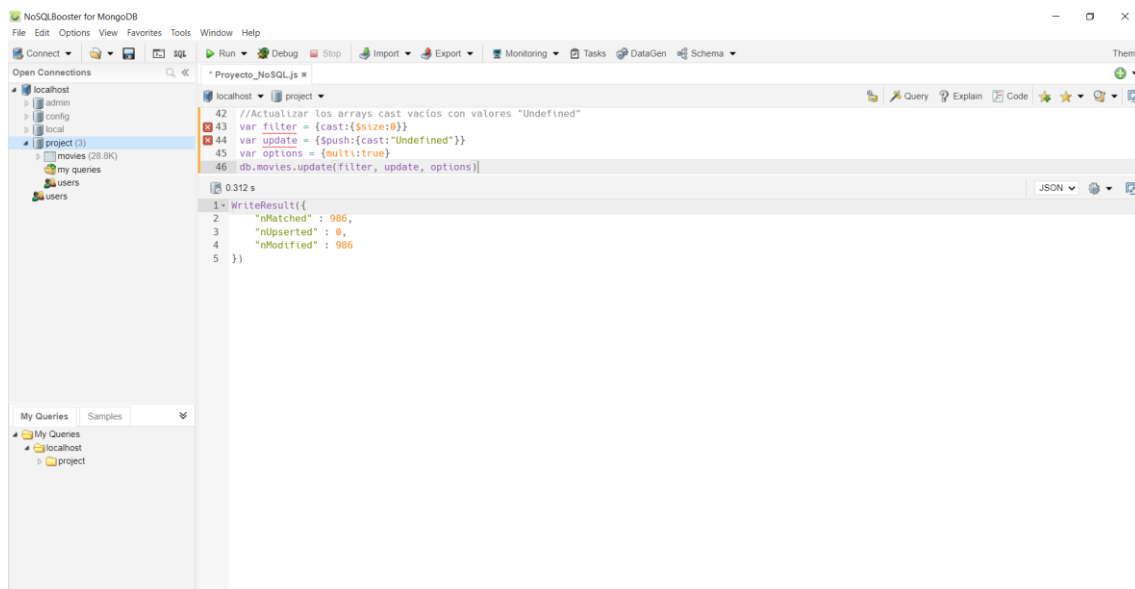
Se actualizan las películas sin actores (array cast vacío) añadiendo el valor "Undefined" en los arrays vacíos.

```
var filter = {cast:{$size:0}}
```

```
var update = {$push:{cast:"Undefined"}}
```

```
var options = {multi:true}
```

```
db.movies.update(filter, update, options)
```

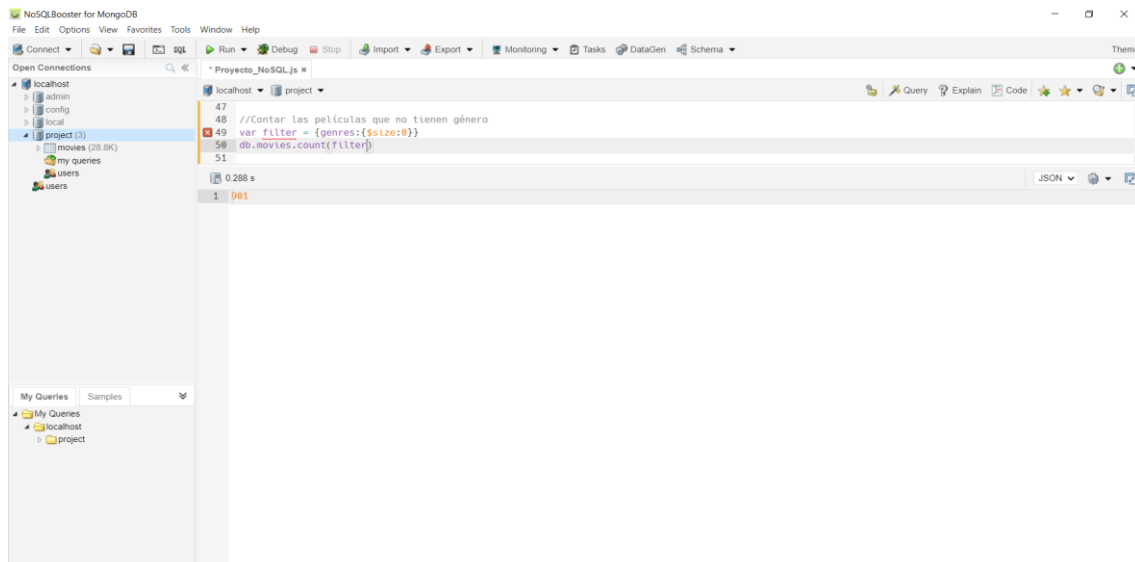


9. Contar cuantos documentos (películas) no tienen Género (array genres).

Se cuentan cuántas películas no tienen género (array genres vacío) contando los arrays que tienen tamaño cero.

```
var filter = {genres:{$size:0}}
```

```
db.movies.count(filter)
```

10. Actualizar TODOS los documentos (películas) que no tengan géneros (array genres), añadiendo un nuevo elemento dentro del array con valor Undefined. ¡Cuidado! El tipo de genres debe seguir siendo un array. El array debe ser así -> ["Undefined"] .

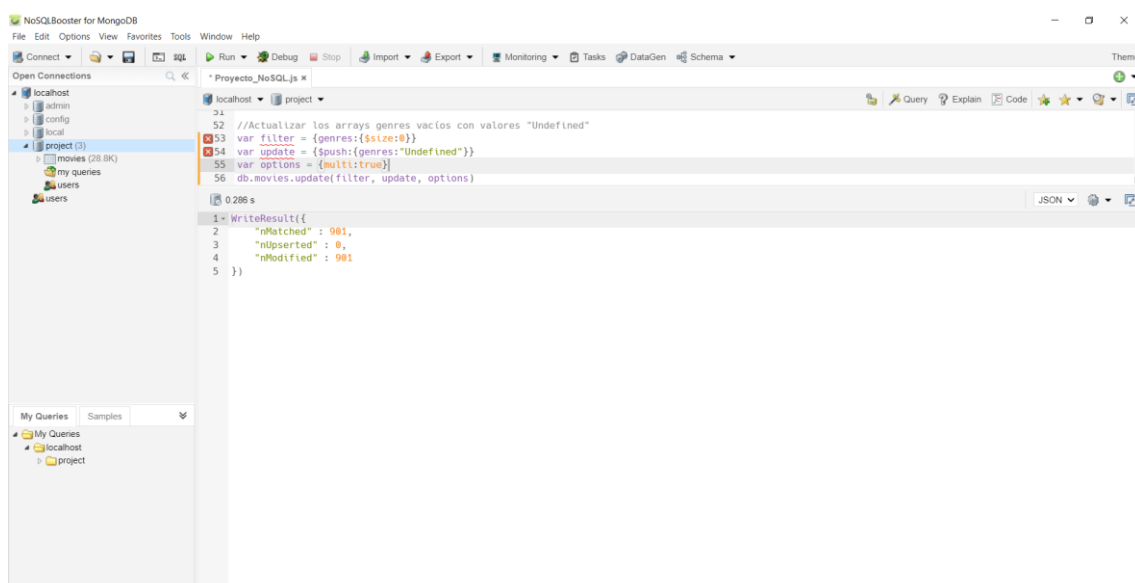
Se actualizan las películas sin géneros (array genres vacío) añadiendo el valor “Undefined” en los arrays vacíos.

```
var filter = {genres:{$size:0}}
```

```
var update = {$push:{genres:"Undefined"}}
```

```
var options = {multi:true}
```

```
db.movies.update(filter, update, options)
```



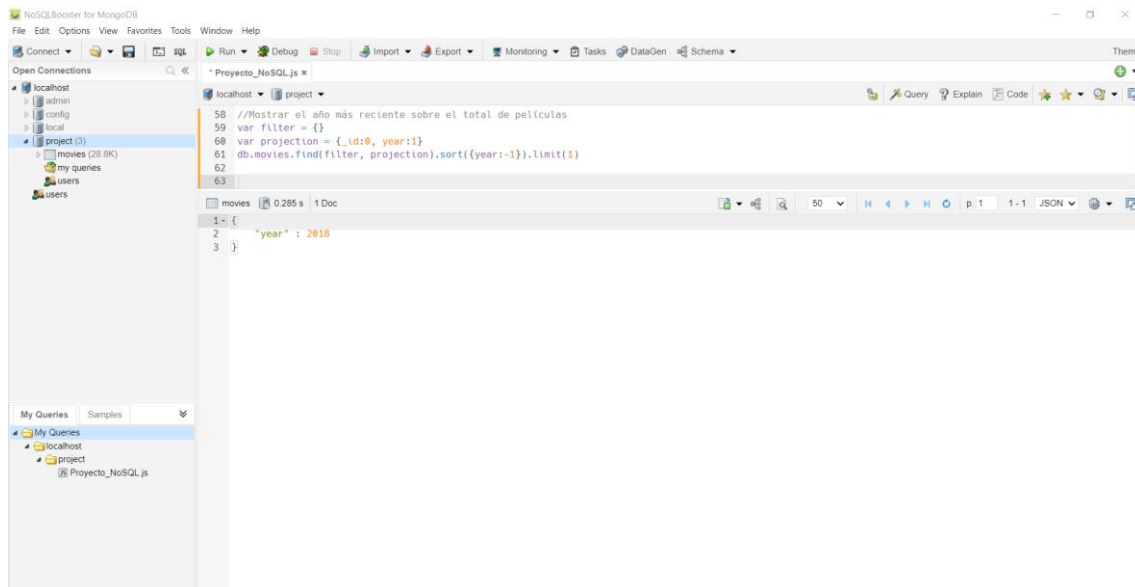
11. Mostrar el año más reciente / actual que tenemos sobre todas las películas.

Se muestra el último año para el que constan la realización de películas.

```
var filter = {}
```

```
var projection = {_id:0, year:1}
```

```
db.movies.find(filter, projection).sort({year:-1}).limit(1)
```



12. Contar cuántas películas han salido en los últimos 20 años. Debe hacerse desde el último año que se tienen registradas películas en la colección, mostrando el resultado total de esos años. Se debe hacer con el Framework de Agregación.

Se cuentan el número de películas realizadas en los últimos veinte años, 1998 (exclusive) y 2018 (inclusive). Como resultado se tienen 4.787 películas.

```
var variable_1 = {$max:"$year"}
```

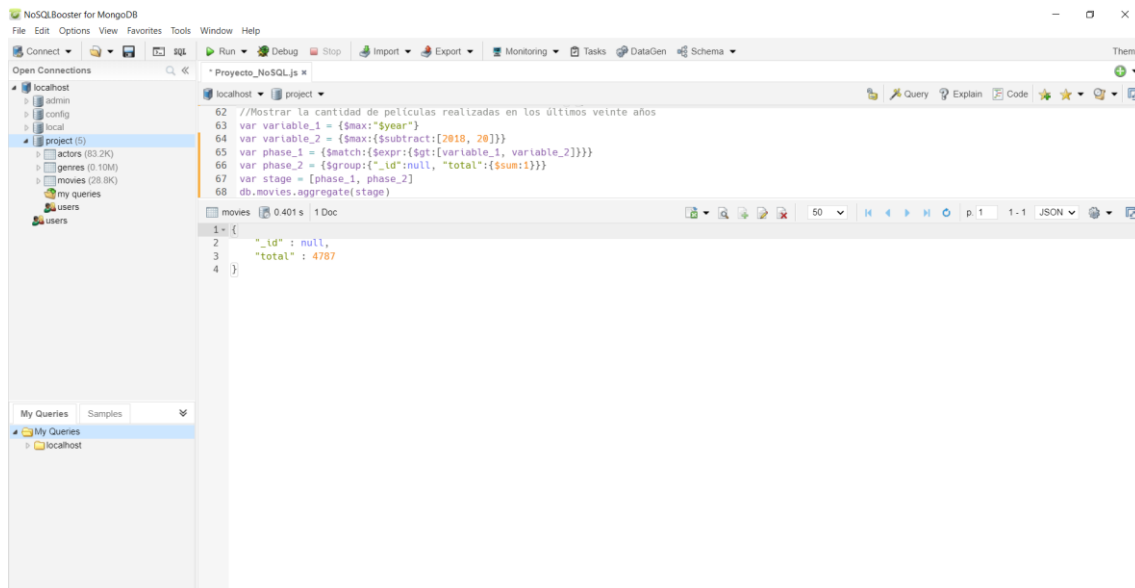
```
var variable_2 = {$max:{$subtract:[2018, 20]}}
```

```
var phase_1 = {$match:{$expr:{$gt:[variable_1, variable_2]}}}
```

```
var phase_2 = {$group:{"_id":null, "total":{$sum:1}}}
```

```
var stage = [phase_1, phase_2]
```

```
db.movies.aggregate(stage)
```



13. Contar cuántas películas han salido en la década de los 60 (del 60 al 69 incluidos). Se debe hacer con el Framework de Agregación.

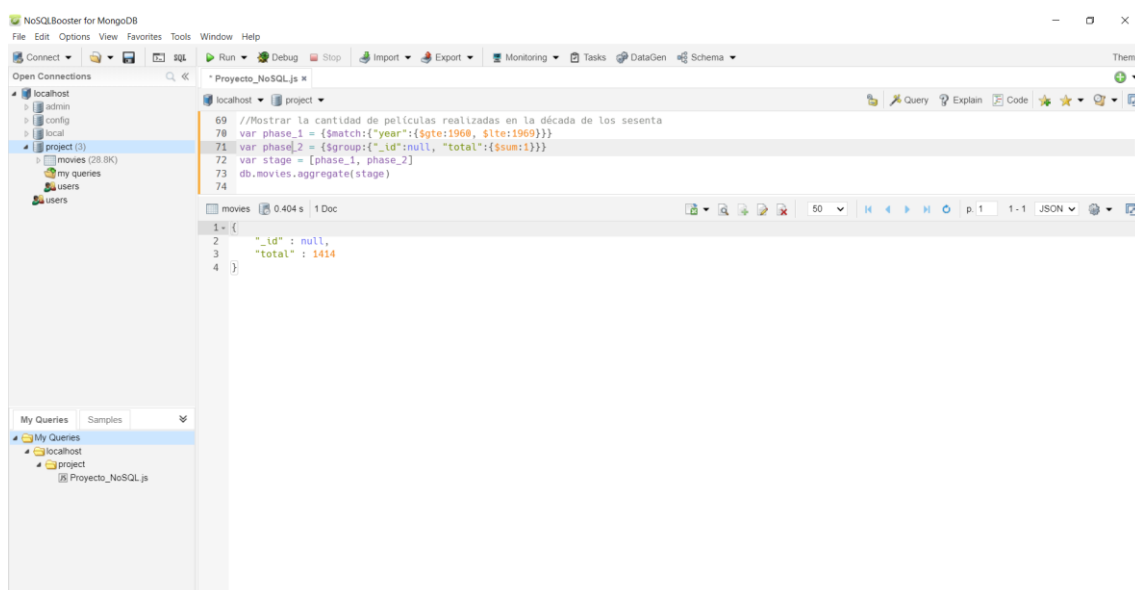
Se cuentan el número de películas realizadas en la década de los sesenta, 1960 (inclusive) y 1969 (inclusive). Como resultado se tienen 1.414 películas.

```
var phase_1 = {$match:{"year":{$gte:1960, $lte:1969}}}
```

```
var phase_2 = {$group:{"_id":null, "total":{$sum:1}}}
```

```
var stage = [phase_1, phase_2]
```

```
db.movies.aggregate(stage)
```



14. Mostrar el año u años con más películas mostrando el número de películas de ese año. Revisar si varios años pueden compartir tener el mayor número de películas.

Se obtiene el año en el que más películas se han realizado. Siendo 1919 el año buscado con 634 películas y el único año que más películas tiene.

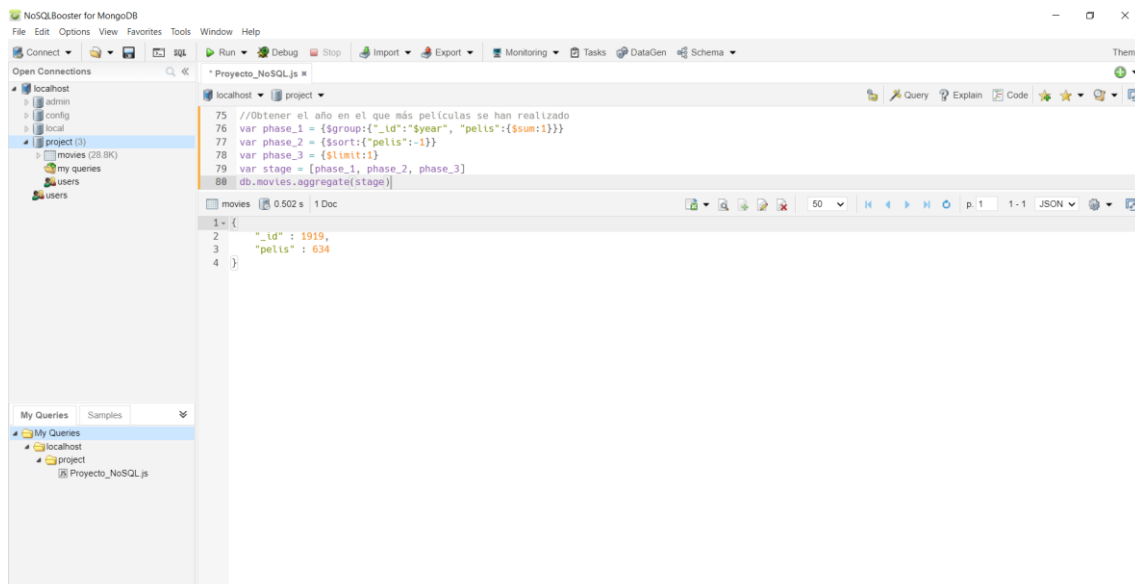
```
var phase_1 = {$group:{"_id":"$year", "pelis":{$sum:1}}}
```

```
var phase_2 = {$sort:{"pelis":-1}}
```

```
var phase_3 = {$limit:1}
```

```
var stage = [phase_1, phase_2, phase_3]
```

```
db.movies.aggregate(stage)
```



15. Mostrar el año u años con menos películas mostrando el número de películas de ese año. Revisar si varios años pueden compartir tener el menor número de películas.

Se obtiene el año en el que más películas se han realizado. Siendo 1902, 1906 y 1907 los años buscados con 7 películas cada uno de estos años.

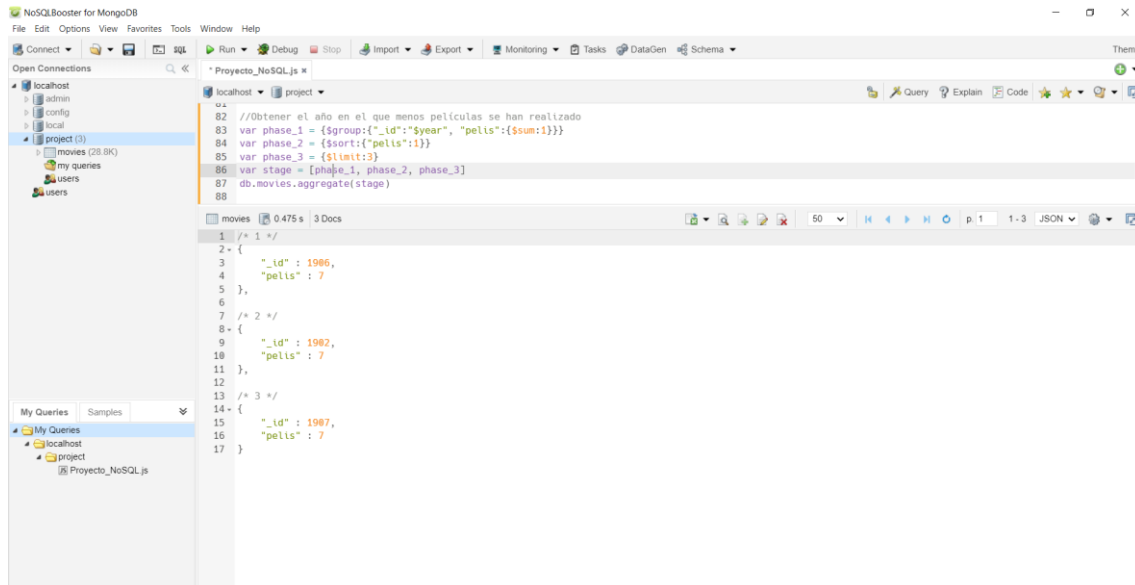
```
var phase_1 = {$group:{"_id":"$year", "pelis":{$sum:1}}}
```

```
var phase_2 = {$sort:{"pelis":1}}
```

```
var phase_3 = {$limit:3}
```

```
var stage = [phase_1, phase_2, phase_3]
```

```
db.movies.aggregate(stage)
```



16. Guardar en nueva colección llamada “actors” realizando la fase \$unwind por actor. Después, contar cuantos documentos existen en la nueva colección.

Se guarda una nueva colección llamada “actors” en la base de datos, esta nueva colección estará compuesta de 83.224 películas.

```
var phase_1 = {$unwind:"$cast"}
```

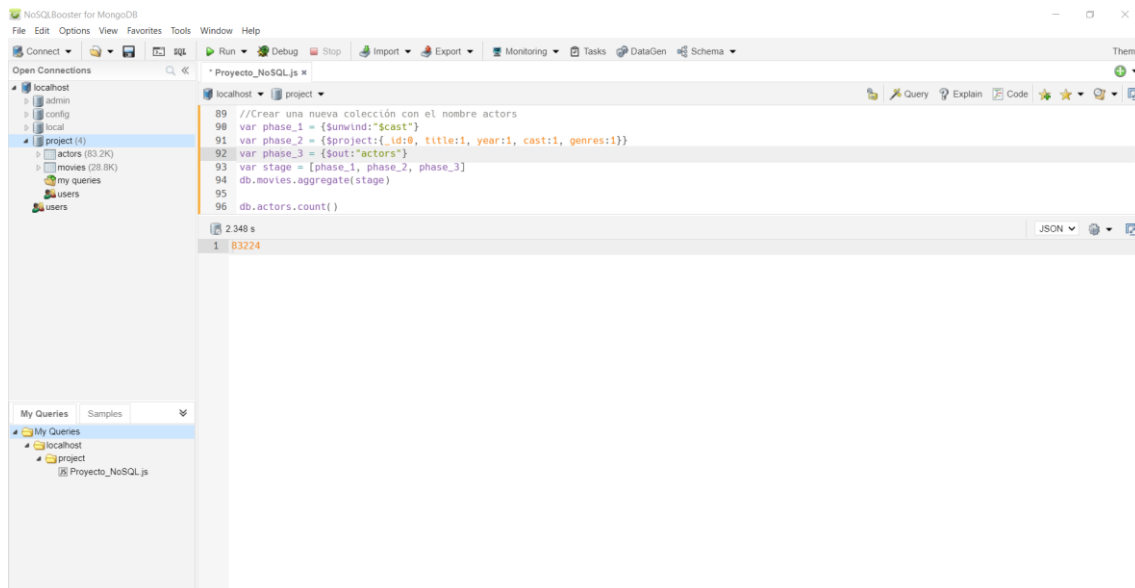
```
var phase_2 = {$project:{_id:0, title:1, year:1, cast:1, genres:1}}
```

```
var phase_3 = {$out:"actors"}
```

```
var stage = [phase_1, phase_2, phase_3]
```

```
db.movies.aggregate(stage)
```

```
db.actors.count()
```



17. Sobre actors (nueva colección), mostrar la lista con los 5 actores que han participado en más películas mostrando el número de películas en las que ha participado. ¡Importante! Se necesita previamente filtrar para descartar aquellos actores llamados "Undefined". Aclarar que no se eliminan de la colección, sólo que filtramos para que no aparezcan.

Los primeros cinco actores que han participado en mayor cantidad de películas y el número total de películas son:

1. Harold Lloyd → 190 participaciones
2. Hoot Gibson → 142 participaciones
3. John Wayne → 136 participaciones
4. Charles Starrett -> 116 participaciones
5. Bebe Daniels → 103 participaciones

var phase_1 = {\$match:{cast:{\$ne:"Undefined"}}}

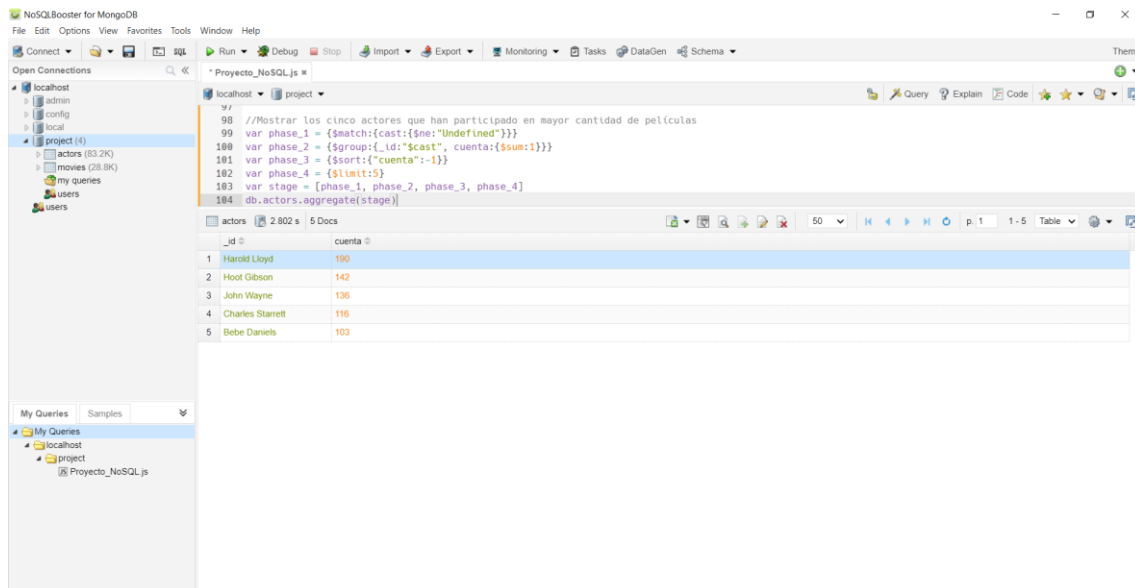
var phase_2 = {\$group:{_id:"\$cast", cuenta:{\$sum:1}}}

var phase_3 = {\$sort:{"cuenta":-1}}

var phase_4 = {\$limit:5}

var stage = [phase_1, phase_2, phase_3, phase_4]

db.actors.aggregate(stage)



18. Sobre actors (nueva colección), agrupar por película y año mostrando las 5 en las que más actores hayan participado, mostrando el número total de actores.

Las cinco películas en las que mayor cantidad de actores han participado y el número total de actores son:

1. The Twilight Saga: Breaking Dawn - Part 2 → 35 actores
2. Anchorman 2: The Legend Continues → 33 actores
3. Cars 2 → 32 actores
4. Avengers: Infinity War → 29 actores
5. Grown Ups 2 → 28 actores

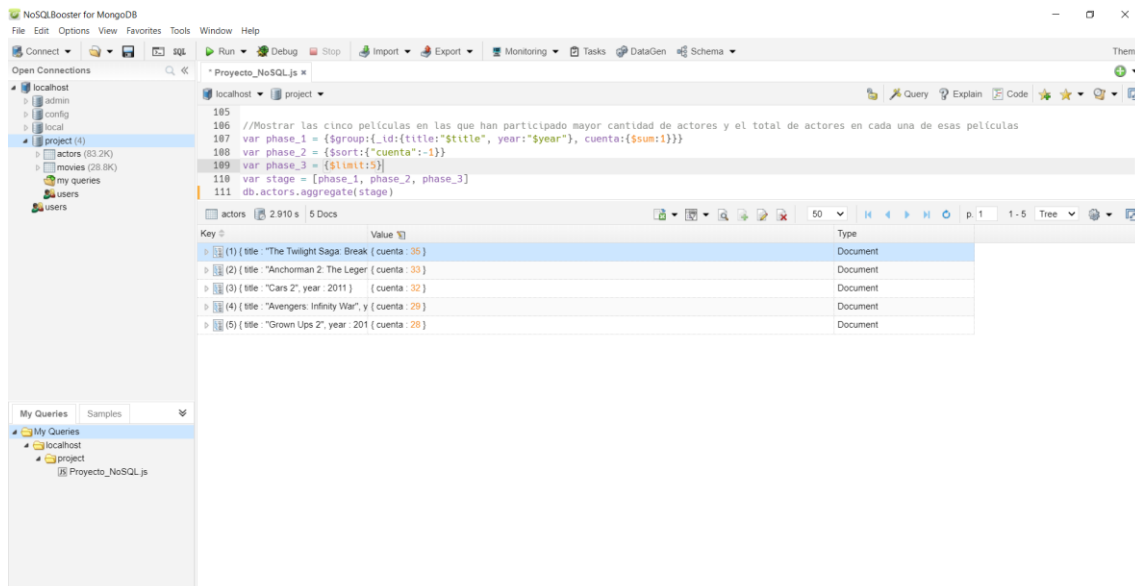
```
var phase_1 = {$group:{_id:{title:"$title", year:"$year"}, cuenta:{$sum:1}}}
```

```
var phase_2 = {$sort:{"cuenta":-1}}
```

```
var phase_3 = {$limit:5}
```

```
var stage = [phase_1, phase_2, phase_3]
```

```
db.actors.aggregate(stage)
```



19. Sobre actors (nueva colección), mostrar los 5 actores cuya carrera haya sido la más larga. Para ello, se debe mostrar cuándo comenzó su carrera, cuándo finalizó y cuántos años han pasado. ¡Importante! Se necesita previamente filtrar para descartar aquellos actores llamados "Undefined". Aclarar que no se eliminan de la colección, sólo que los filtramos para que no aparezcan.

Los primeros cinco actores que han tenido una carrera más larga y la duración en años de sus carreras son:

1. Harrison Ford → 98 años
2. Gloria Stuart → 80 años
3. Lillian Gish → 75 años
4. Kenny Baker → 75 años
5. Mickey Rooney / Angela Lansbury → 74 años

Nótese que la carrera de Harrison Ford tiene una duración de 98 años. Esto se debe a que verdaderamente existen dos actores con el mismo nombre, pero al no haber una clave identificativa que diferencia a ambos actores, la duración de estos dos actores se muestra como si fuera la de un único actor.

```
var phase_1 = {$match:{cast:{$ne:"Undefined"}}}
```

```
var phase_2 = {$group:{_id:"$cast", comienza:{$min:"$year"},
termina:{$max:"$year"}}
```

```
var phase_3 = {$project:{_id:1, comienza:1, termina:1,
años:{$subtract:["$termina", "$comienza"]}}}
```

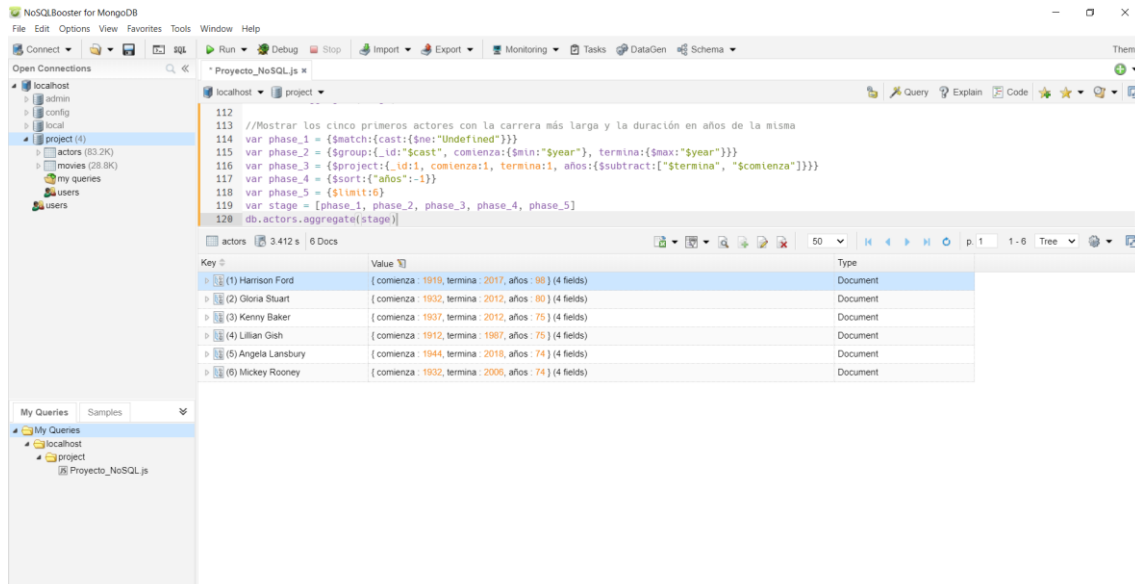


```
var phase_4 = {$sort:{"años":-1}}
```

```
var phase_5 = {$limit:6}
```

```
var stage = [phase_1, phase_2, phase_3, phase_4, phase_5]
```

```
db.actors.aggregate(stage)
```



20. Sobre actors (nueva colección), guardar en nueva colección llamada “genres” realizando la fase \$unwind por genres. Después, contar cuantos documentos existen en la nueva colección.

Se guarda una nueva colección llamada “genres” en la base de datos, esta nueva colección estará compuesta de 104.950 películas.

```
var phase_1 = {$unwind:"$genres"}
```

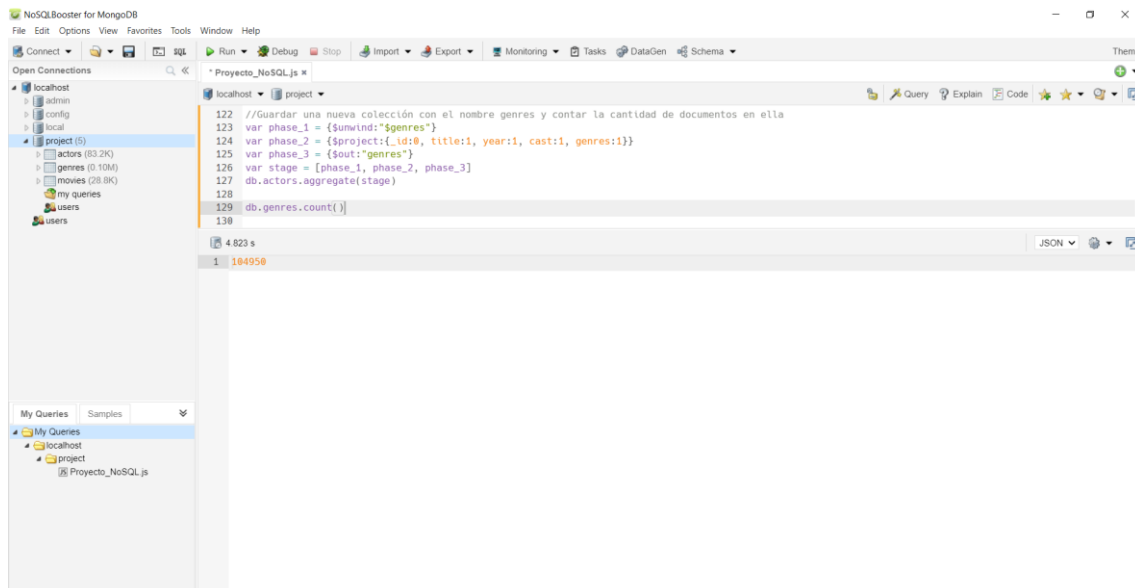
```
var phase_2 = {$project:{_id:0, title:1, year:1, cast:1, genres:1}}
```

```
var phase_3 = {$out:"genres"}
```

```
var stage = [phase_1, phase_2, phase_3]
```

```
db.actors.aggregate(stage)
```

```
db.genres.count()
```



21. Sobre genres (nueva colección), mostrar los 5 documentos agrupados por “Año y Género” que más número de películas **diferentes tienen mostrando el número total de películas.**

Se agrupan los documentos por año y género y se muestran los cinco primeros documentos que mayor cantidad de películas diferentes tienen:

1. Dramas en 1919 con 291 películas.
2. Dramas en 1925 con 247 películas.
3. Dramas en 1924 con 233 películas.
4. Comedias en 1919 con 226 películas.
5. Dramas en 1922 con 209 películas.

```
var phase_1 = {$group:{_id:{year:"$year", genre:"$genres"},
                      total:{$addToSet:"$title"}}}
```

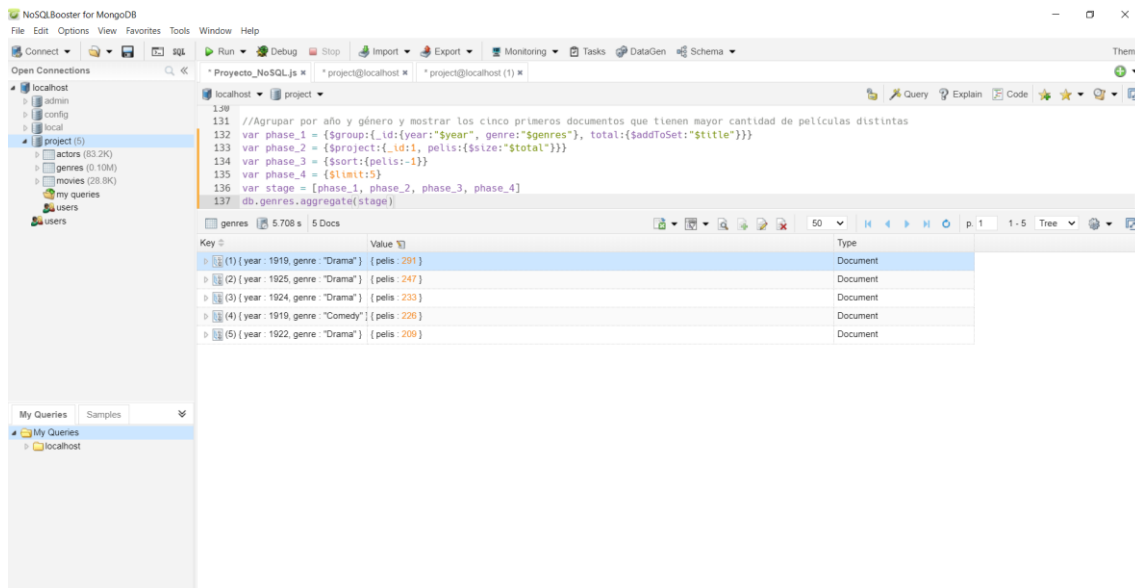
```
var phase_2 = {$project:{_id:1, pelis:{$size:"$total"}}}
```

```
var phase_3 = {$sort:{pelis:-1}}
```

```
var phase_4 = {$limit:5}
```

```
var stage = [phase_1, phase_2, phase_3, phase_4]
```

```
db.genres.aggregate(stage)
```



22. Sobre genres (nueva colección), mostrar los 5 actores y los géneros en los que han participado con más número de géneros **diferentes, se debe mostrar el número de géneros diferentes que ha interpretado. ¡Importante! Se necesita previamente filtrar para descartar aquellos actores llamados "Undefined". Aclarar que no se eliminan de la colección, sólo que filtramos para que no aparezcan.**

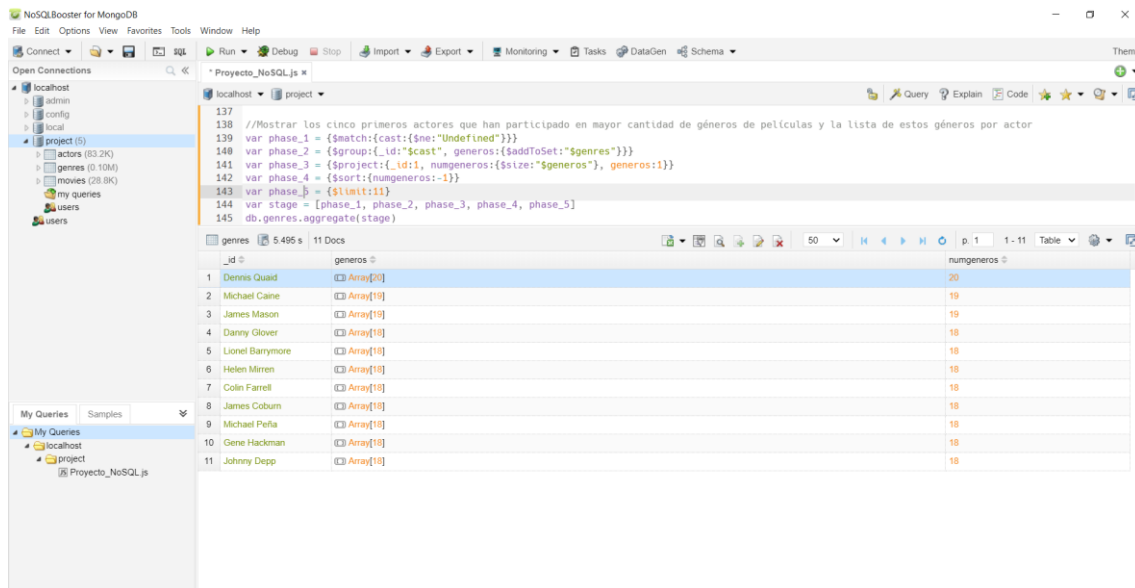
Se muestran los cinco primeros actores que en mayor cantidad de géneros de películas han participado y cuáles son estos géneros en los que han participado a lo largo de sus carreras:

1. Dennis Quaid → 20 géneros
2. James Mason y Michael Caine → 19 géneros
3. Colin Farrell | Danny Glover | Helen Mirren | James Coburn | Johnny Deep | Gene Hackman | Michael Peña | Lionel Barrymore → 18 géneros

```

var phase_1 = {$match:{cast:{$ne:"Undefined"}}}
var phase_2 = {$group:{_id:"$cast", generos:{$addToSet:"$genres"}}}
var phase_3 = {$project:{_id:1, numgeneros:{$size:"$generos"},
                        generos:1}}
var phase_4 = {$sort:{numgeneros:-1}}
var phase_5 = {$limit:11}
var stage = [phase_1, phase_2, phase_3, phase_4, phase_5]
db.genres.aggregate(stage)

```



23. Sobre genres (nueva colección), mostrar las 5 películas y su año correspondiente en los que más géneros **diferentes han sido catalogados, mostrando esos géneros y el número de géneros que contiene.**

Se muestran las cinco primeras películas y su año que han sido catalogadas en mayor cantidad de géneros diferentes mostrando esos géneros y la cantidad total que es:

1. American Made → 7 géneros
2. Wonder Woman | Thor: Ragnarok | The Dark Tower | Dunkirk | My Little Pony: The Movie → 6 géneros

```
var phase_1 = {$group:{_id:{title:"$title", year:"$year"},
                      generos:{$addToSet:"$genres"}}}
```

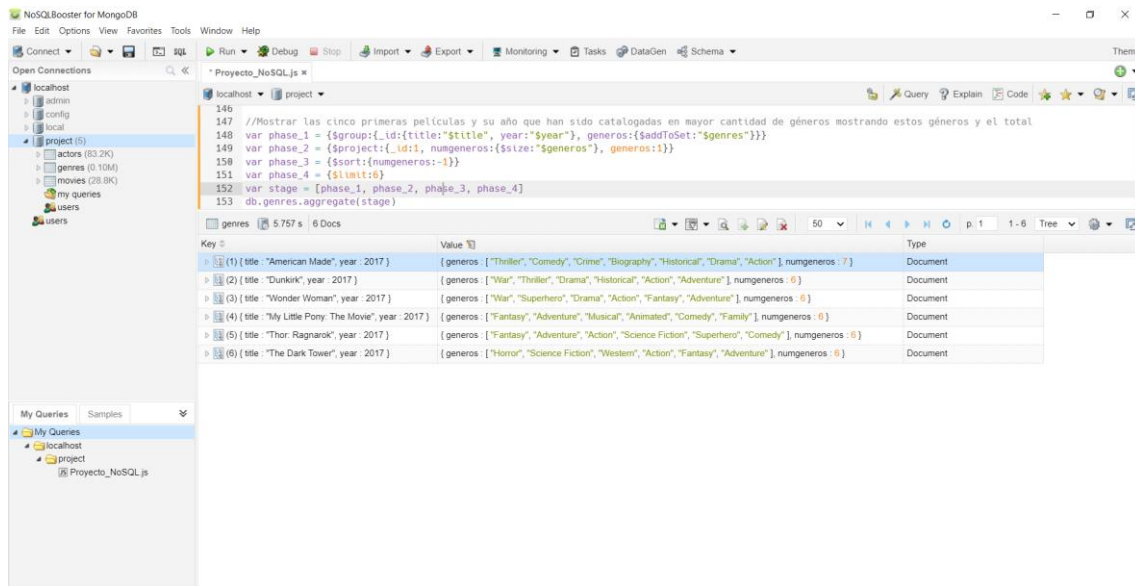
```
var phase_2 = {$project:{_id:1, numgeneros:{$size:"$generos"},
                        generos:1}}
```

```
var phase_3 = {$sort:{numgeneros:-1}}
```

```
var phase_4 = {$limit:6}
```

```
var stage = [phase_1, phase_2, phase_3, phase_4]
```

```
db.genres.aggregate(stage)
```



24. Query libre sobre el pipeline de agregación.

Se realiza una query con el pipeline de agregación sobre la colección llamada actors en la que se quiere conocer qué actor ha participado en más cantidad de películas diferentes en un mismo año. Para esta consulta, se obtiene un único actor que cumple estas condiciones tratándose de Harold Lloyd con 38 participaciones en películas diferentes en el año 1919, se consulta también de qué películas se tratan estas 38 participaciones.

```
var phase_1 = {$match:{cast:{$ne:"Undefined"}}}
```

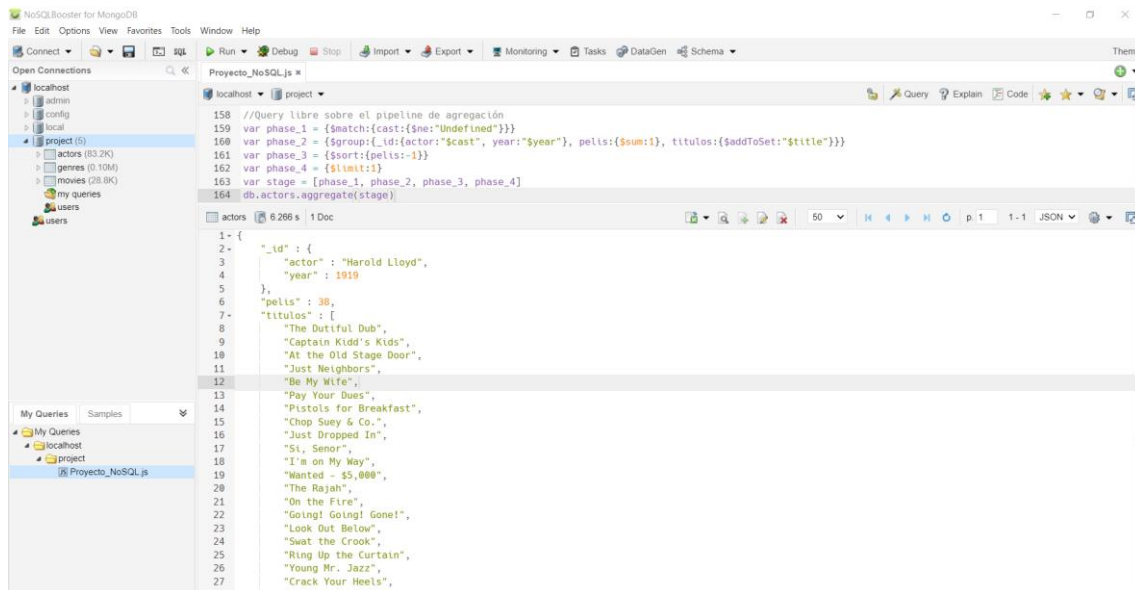
```
var phase_2 = {$group:{_id:{actor:"$cast", year:"$year"}, pelis:{$sum:1},  
titulos:{$addToSet:"$title"}}
```

```
var phase_3 = {$sort:{pelis:-1}}
```

```
var phase_4 = {$limit:1}
```

```
var stage = [phase_1, phase_2, phase_3, phase_4]
```

```
db.actors.aggregate(stage)
```



25. Query libre sobre el pipeline de agregación.

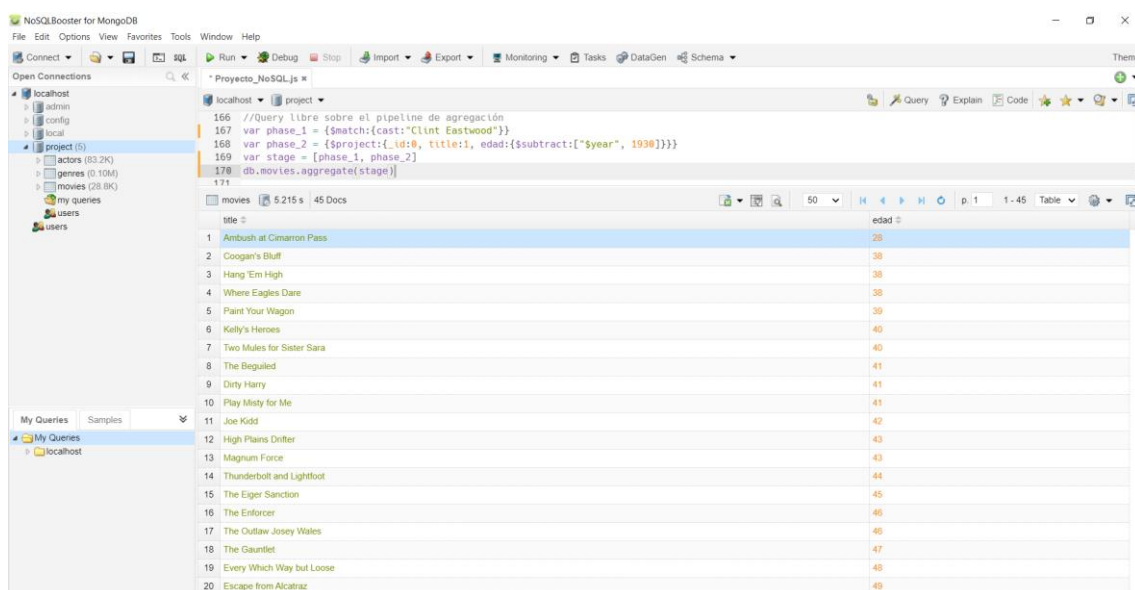
Se realiza una query con el pipeline de agregación sobre la colección llamada movies en la que se quiere consultar la edad del actor Clint Eastwood en el momento de participar en cada una de las películas de su carrera.

```
var phase_1 = {$match:{cast:"Clint Eastwood"}}
```

```
var phase_2 = {$project:{_id:0, title:1, edad:{$subtract:[$year, 1930]}}}
```

```
var stage = [phase_1, phase_2]
```

```
db.movies.aggregate(stage)
```



26. Query libre sobre el pipeline de agregación.

Se realiza una query con el pipeline de agregación sobre la colección llamada actors para conocer si la actriz Angelina Jolie ha participado en alguna película realizada en el año 1998, realizando la misma consulta para el año 2018. Como resultado se obtiene que en el año 1998 si ha participado en alguna película y en el año 2018 no ha participado en ninguna película.

```
var phase_1 = {$match:{cast:"Angelina Jolie"}}
```

```
var phase_2 = {$group:{_id:"$cast", año:{$addToSet:"$year"}}}
```

```
var phase_3 = {$project:{_id:1, pelicula_en_1998:{$in:[1998, "$año"]},  
pelicula_en_2018:{$in:[2018, "$año"]}}}
```

```
var stage = [phase_1, phase_2, phase_3]
```

```
db.actors.aggregate(stage)
```

