

## 1. Analizar con find la colección.

-Comando:

```
4 //1. Analizar con find la colección.
5 db.Movies.find()
6
```

-Resultado:

Movies	0.035 s	28.795 Docs
1	/* 1 createdAt:30/4/2024 22:28:50*/	
2	{	
3	"_id" : ObjectId("663154828ca38dba8183d74e"),	
4	"title" : "Caught",	
5	"year" : 1900,	
6	"cast" : [ ],	
7	"genres" : [ ]	
8	},	
9		
10	/* 2 createdAt:30/4/2024 22:28:50*/	
11	{	
12	"_id" : ObjectId("663154828ca38dba8183d74f"),	
13	"title" : "After Dark in Central Park",	
14	"year" : 1900,	
15	"cast" : [ ],	
16	"genres" : [ ]	
17	},	
18		

## 1. Contar cuántos documentos (películas) tiene cargado.

-Comando:

```
//2. Contar cuántos documentos (películas) tiene cargado.
db.Movies.countDocuments()
```

-Resultado:

Find x	Console x	Find (1) x	Result x
0.016 s			
1	28795		

## 2. Insertar una película.

### -Comando:

```
//3. Insertar una película.  
var nueva_peli={ "title": "Avengers Endgame", "year": 2019, "cast": [], "genres": []}  
db.Movies.insertOne(nueva_peli)  
  
db.Movies.find()  
  
var query= { "title": "Avengers Endgame"}  
db.Movies.find(query)
```

### -Resultado:

Find x	Console x	Find (1) x	Result x	Result (1) x
0.014 s				
Key	Value	Type		
acknowledged	true	Boolean		
insertedId	663154c340227b00ce71764	ObjectId		

Movies	0.013 s	28.796 Docs
--------	---------	-------------

Key	Value	Type
_id	663154c340227b00ce71764	ObjectId
title	Avengers Endgame	String
year	2019	Int32
cast	Array[]	Array
genres	Array[]	Array

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

### -Comando:

```
//4. Borrar la película insertada en el punto anterior  
db.Movies.deleteOne(query)  
db.Movies.find()  
  
var query= { "title": "Avengers Endgame"}  
db.Movies.find(query)
```

### -Resultado:

Key	Value	Type
acknowledged	true	Boolean
deletedCount	1	Int32

4. Contar cuantas películas tienen actores (cast) que se llaman "and". Estos nombres de actores están por ERROR.

-Comando:

```
//5. Contar cuantas películas tienen actores (cast) que se llaman "and".  
//Visualizar cuantos documentos tienen "and"  
var r = {"cast":"and"}  
db.Movies.find(r)
```

-Resultado:

0.023 s
1 93

5. 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 actores.

-Comando:

```
//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.  
  
db.Movies.updateMany(  
  { cast: { $elemMatch: { $eq: "and" } } },  
  { $pull: { cast: "and" } } )
```

-Resultado:

```
1 {  
2   "acknowledged" : true,  
3   "matchedCount" : 93,  
4   "modifiedCount" : 93  
5 }
```

6. Contar cuantos documentos (películas) tienen el array 'cast' vacío.

-Comando:

```
//7. Contar documentos con el array 'cast' vacío|
db.Movies.countDocuments({ cast: { $exists: true, $eq: [] } })
```

-Resultado:

0.021 s
1 986

7. Actualizar TODOS los documentos (películas) que tengan el array cast vacío, 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" ].

-Comando:

```
//8. Actualizar todos los documentos con el array 'cast' vacío
db.Movies.updateMany(
  { cast: { $exists: true, $eq: [] } },
  { $set: { cast: [ "Undefined" ] } })
```

-Resultado:

```
1 {
2   "acknowledged" : true,
3   "matchedCount" : 986,
4   "modifiedCount" : 986
5 }
```

8. Contar cuantos documentos (películas) tienen el array genres vacío.

-Comando:

```
//9 Contar cuantos documentos (películas) tienen el array genres vacío.

db.Movies.countDocuments({ genres: { $exists: true, $eq: [] } })
```

**-Resultado:**

```
1 901
```

9. Actualizar TODOS los documentos (películas) que tengan el array genres vacío, 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" ].

**-Comando:**

```
//10 Actualizar TODOS los documentos (películas) que tengan el array genres vacío, 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" ]

// Actualizar todos los documentos con el array 'genres' vacío
db.Movies.updateMany(
  { genres: { $exists: true, $eq: [] } },
  { $set: { genres: ["Undefined"] } })
```

**-Resultado:**

```
1 {
2   "acknowledged" : true,
3   "matchedCount" : 901,
4   "modifiedCount" : 901
5 }
```

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

**-Comando:**

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

// Obtener una lista de todos los años únicos de las películas
var añosUnicos = db.Movies.distinct("year")

// Encontrar el máximo año de la lista
var añoMasReciente = Math.max(...añosUnicos)

print("El año más reciente / actual de todas las películas es: " + añoMasReciente)
```

**-Resultado:**



```
1 El año más reciente / actual de todas las películas es: 2018
```

11. 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.

-Comando:

```
12 //12 Contar cuántas películas han salido en los últimos 20 años. Debe hacerse desde el último año que se tienen
13 //registradas películas en la colección, mostrando el resultado total de esos años. Se debe hacer con el Framework de
14 //Agregación.
15 //Pelis en los ultimos 20 años
16
17
18 var fase1={"$match":{"year":{"$gt:1998 , $lte:2018}}}
19 var fase2={"$group":{"_id":"null", "total_movies":{"$sum:1}}}
20 var fase3={"$project":{"_id":0}}
21 var etapas=[fase1, fase2, fase3]
22 db.Movies.aggregate(etapas)
```

-Resultado:

80 //de Agregacion.	
 Movies	 0.063 s 1 Doc
Key	Value
1 (1)	{ total_movies : 4787 }
total_movies	4787 (4.8K)

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

-Comando:

```
13 //13Contar cuántas películas han salido en la década de los 60 (del 60 al 69 incluidos). Se debe hacer con el Framework
14 //de Agregación.
15
16 //Películas del 60 al 69
17 var fase1={"$match":{"year":{"$gte:1960 , $lte:1969}}}
18 var fase2={"$group":{"_id":"null", "movies_60_69":{"$sum:1}}}
19 var fase3={"$project":{"_id":0}}
20 var etapas=[fase1, fase2, fase3]
21 db.Movies.aggregate(etapas)
```

-Resultado:

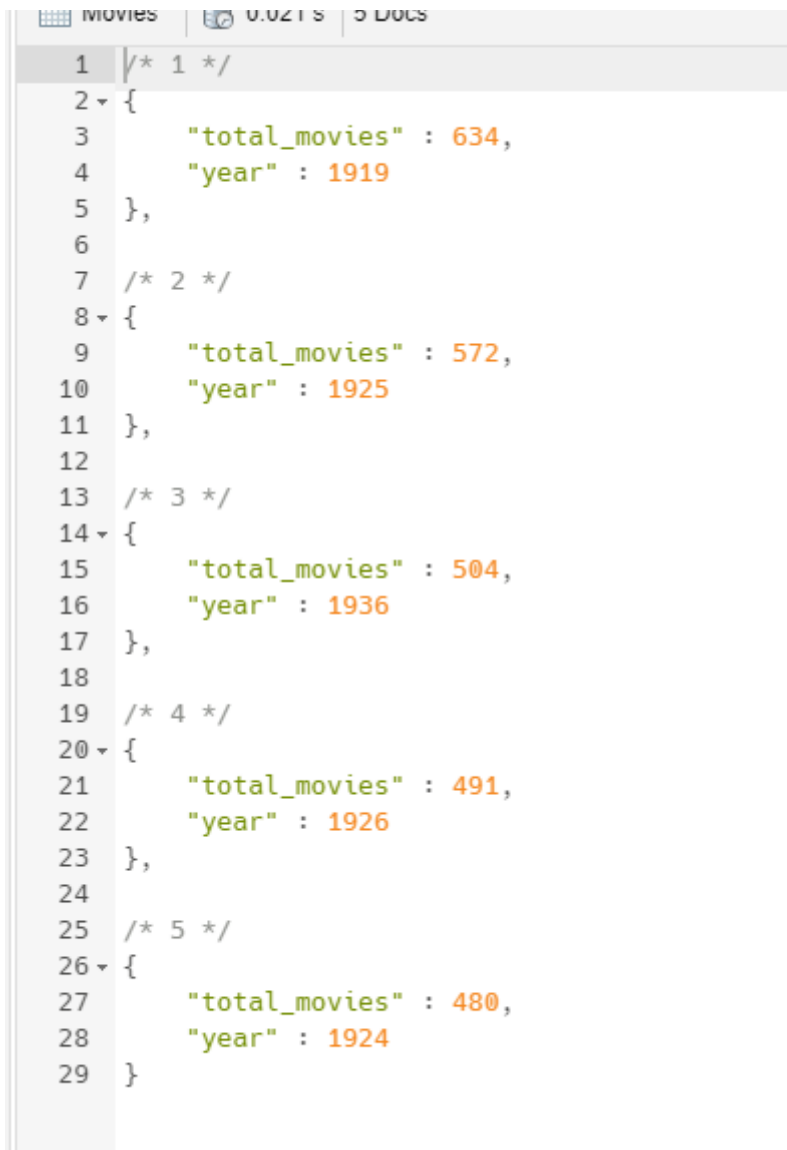
```
1 {
2   "movies_60_69" : 1414
3 }
```

13. 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.

-Comando:

```
//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.  
  
var fase1={"$group":{"_id":"$year", "total_movies":{"$sum":1}}}  
var fase2={"$project":{"_id":0,"year":"$_id","total_movies":1}}  
var fase3={"$sort":{"total_movies":-1}}  
var fase4={"$limit":5}  
var etapas=[ fase1,fase2, fase3,fase4]  
db.Movies.aggregate(etapas)
```

-Resultado:



The screenshot shows a MongoDB aggregation result in a web interface. The interface has a header with 'MOVIES', '0.021 S', and '5 Docs'. The result is displayed as a JSON array with 5 documents. Each document contains 'total\_movies' and 'year' fields. The years are 1919, 1925, 1936, 1926, and 1924, with total movie counts of 634, 572, 504, 491, and 480 respectively. The interface includes line numbers from 1 to 29 on the left side of the code editor.

```
1 /* 1 */  
2 {  
3   "total_movies" : 634,  
4   "year" : 1919  
5 },  
6  
7 /* 2 */  
8 {  
9   "total_movies" : 572,  
10  "year" : 1925  
11 },  
12  
13 /* 3 */  
14 {  
15   "total_movies" : 504,  
16   "year" : 1936  
17 },  
18  
19 /* 4 */  
20 {  
21   "total_movies" : 491,  
22   "year" : 1926  
23 },  
24  
25 /* 5 */  
26 {  
27   "total_movies" : 480,  
28   "year" : 1924  
29 }
```

**14. 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.**

**-Comando:**

```
15 //15 Mostrar el año u años con menos peliculas mostrando el número de películas de ese año. Revisar si varios años
16 //pueden compartir tener el menor número de películas.
17
18 var fase1={"$group":{"_id":"$year", "total_movies":{"$sum":1}}}
19 var fase2={"$project":{"_id":0,"year":"$_id","total_movies":1}}
20 var fase3={"$sort":{"total_movies":1}}
21 var fase4={"$limit":5}
22 var etapas=[ fase1,fase2, fase3,fase4]
23 db.Movies.aggregate(etapas)
```

**-Resultado:**

```
1  /* 1 */
2  {
3    "total_movies" : 7,
4    "year" : 1907
5  },
6
7  /* 2 */
8  {
9    "total_movies" : 7,
10   "year" : 1906
11 },
12
13 /* 3 */
14 {
15   "total_movies" : 7,
16   "year" : 1902
17 },
18
19 /* 4 */
20 {
21   "total_movies" : 17,
22   "year" : 1900
23 },
24
25 /* 5 */
26 {
27   "total_movies" : 18,
28   "year" : 1908
29 }
```



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

-Comando:

```
//16 Guardar en nueva colección llamada "actors" realizando la fase $unwind por actor. Después, contar cuantos
//documentos existen en la nueva colección.

// Fase para expandir los elementos de la matriz cast en documentos separados
var etapaUnwind = { "$unwind": "$cast" };
var etapaProyeccion = { "$project": { "_id": 0, "title": "$title", "year": "$year", "cast": "$cast", "genres": "$genres" } }
var etapaOut = { "$out": "actors" }
var pipeline = [etapaUnwind, etapaProyeccion, etapaOut]
db.Movies.aggregate(pipeline)
var count = db.actors.countDocuments()
print("Número de documentos en la colección 'actors': " + count)

db.actors.countDocuments()
```

-Resultado:



16. Sobre actores (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.

-Comando:

```
//17 . Sobre actores (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.

var filtro = {"cast":{"$ne": "Undefined"}}
var fase2= {"$match":filtro}
var fase3={"$group":{"_id":"$cast", "total_movies":{"$sum":1}}}
var fase4= {"$sort":{"total_movies":-1}}
var fase5= {"$limit": 5}

var fases=[ fase2, fase3, fase4, fase5]
db.actors.aggregate(fases)
```

**-Resultado:**

```
1  /* 1 */
2  {
3      "_id" : "Harold Lloyd",
4      "total_movies" : 190
5  },
6
7  /* 2 */
8  {
9      "_id" : "Hoot Gibson",
10     "total_movies" : 142
11 },
12
13 /* 3 */
14 {
15     "_id" : "John Wayne",
16     "total_movies" : 136
17 },
18
19 /* 4 */
20 {
21     "_id" : "Charles Starrett",
22     "total_movies" : 116
23 },
24
25 /* 5 */
26 {
27     "_id" : "Bebe Daniels",
28     "total_movies" : 103
29 }
```

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

**-Comando:**

```
3 // 18. Sobre actores (nueva colección), agrupar por película y año mostrando las 5 en las que más actores hayan
4 // participado, mostrando el número total de actores.
5
6 //var filtro = {"cast":{"$ne": "Undefined"}}
7 //var fase2= {"$match":filtro}
8 var fase3={"$group":{"_id":{"title":"$title","year":"$year" }, "cuenta":{"$sum":1}}}
9 var fase4= {"$sort":{"cuenta":-1}}
10 var fase5= {"$limit": 5}
11
12 var fases=([ fase3, fase4, fase5])
13 db.actors.aggregate(fases)
14
```

**-Resultado:**

```
/* 1 */
{
  "_id" : {
    "title" : "The Twilight Saga: Breaking Dawn - Part 2",
    "year" : 2012
  },
  "cuenta" : 35
},

/* 2 */
{
  "_id" : {
    "title" : "Anchorman 2: The Legend Continues",
    "year" : 2013
  },
  "cuenta" : 33
},

/* 3 */
{
  "_id" : {
    "title" : "Cars 2",
    "year" : 2011
  },
  "cuenta" : 32
},

/* 4 */
{
  "_id" : {
    "title" : "Avengers: Infinity War",
    "year" : 2018
  },
  "cuenta" : 29
},

/* 5 */
{
  "_id" : {
    "title" : "Grown Ups 2",
    "year" : 2013
  },
  "cuenta" : 28
}
```

18. Sobre actores (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 ha trabajado. 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.

### -Comando:

```
5 // 19.Sobre actores (nueva colección), mostrar los 5 actores cuya carrera haya sido la más larga. Para ello, se debe
6 //mostrar cuándo comenzó su carrera, cuándo finalizó y cuántos años ha trabajado. Importante! Se necesita previamente
7 //filtrar para descartar aquellos actores llamados "Undefined". Aclarar que no se eliminan de la colección, sólo que
8 //filtramos para que no aparezcan.
9
10 var filtro = { "cast": { "$ne": "Undefined" } }
11 var fase2 = { "$match": filtro }
12 var fase3 = { "$group": { "_id": "$cast", "primer_año": { "$min": "$year" }, "ultimo_año": { "$max": "$year" }}}
13 var fase4 = { "$project": { "_id": 1, "primer_año": 1, "ultimo_año": 1, "años_totales": { "$subtract": [ "$ultimo_año", "$primer_año" ] }}}
14 var fase5 = { "$sort": { "años_totales": -1 } }
15 var fase6 = { "$limit": 5 }
16
17 var fases = [ fase2, fase3, fase4, fase5, fase6 ]
18 db.actors.aggregate(fases)
19
```

### -Resultado: (Hay actores con el mismo nombre, de ahí los años totales tan elevados).

```
1 /* 1 */
2 {
3   "_id" : "Harrison Ford",
4   "primer_año" : 1919,
5   "ultimo_año" : 2017,
6   "años_totales" : 98
7 },
8
9 /* 2 */
10 {
11   "_id" : "Gloria Stuart",
12   "primer_año" : 1932,
13   "ultimo_año" : 2012,
14   "años_totales" : 80
15 },
16
17 /* 3 */
18 {
19   "_id" : "Lillian Gish",
20   "primer_año" : 1912,
21   "ultimo_año" : 1987,
22   "años_totales" : 75
23 },
24
25 /* 4 */
26 {
27   "_id" : "Kenny Baker",
28   "primer_año" : 1937,
29   "ultimo_año" : 2012,
30   "años_totales" : 75
31 },
32
33 /* 5 */
34 {
35   "_id" : "Angela Lansbury",
36   "primer_año" : 1944,
37   "ultimo_año" : 2018,
38   "años_totales" : 74
39 }
40
```

19. 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.

-Comando:

```
70 // 20. Sobre actors (nueva colección), Guardar en nueva colección llamada “genres” realizando la fase $unwind por
71 //genres. Después, contar cuantos documentos existen en la nueva colección.
72
73 var fase1 = { "$unwind": "$genres" }
74 var fase2 = { "$project": { "_id": 0, "title": "$title", "year": "$year", "cast": "$cast", "genres": "$genres" } }
75 var fase3 = { "$out": "genres" }
76 var pipeline = [fase1, fase2, fase3]
77 db.actors.aggregate(pipeline)
78 var count = db.genres.countDocuments()
79 print("Número de documentos en la colección 'genres': " + count)
80
81 db.genres.find()
82
```

-Resultado:

```
1 Número de documentos en la colección 'genres': 104950
```

20. 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.

-Comando:

```
184 // 21. Sobre genres (nueva colección), mostrar los 5 documentos agrupados por “Año y Género” que más número de
185 //películas diferentes tienen mostrando el número total de películas.
186
187 var filtro = { "genres": { "$ne": "Undefined" } }
188 var fase2 = { "$match": filtro }
189 var fase3={ "$group": { "_id": { "genero": "$genres", "year": "$year" }, "pelis": { "$addToSet": "$title" } } }
190 var fase4= { "$project": { "_id": 1, "pelis": { "$size": "$pelis" } } }
191 var fase5= { "$sort": { "pelis": -1 } }
192 var fase6= { "$limit": 5 }
193
194 var fases=[ fase2, fase3, fase4, fase5, fase6 ]
195 db.genres.aggregate(fases)
196
197
```

**-Resultado:**

```
1 /* 1 */
2 {
3   "_id" : {
4     "genero" : "Drama",
5     "year" : 1919
6   },
7   "pelis" : 291
8 },
9
10 /* 2 */
11 {
12   "_id" : {
13     "genero" : "Drama",
14     "year" : 1925
15   },
16   "pelis" : 247
17 },
18
19 /* 3 */
20 {
21   "_id" : {
22     "genero" : "Drama",
23     "year" : 1924
24   },
25   "pelis" : 233
26 },
27
28 /* 4 */
29 {
30   "_id" : {
31     "genero" : "Comedy",
32     "year" : 1919
33   },
34   "pelis" : 226
35 },
36
37 /* 5 */
38 {
39   "_id" : {
40     "genero" : "Drama",
41     "year" : 1922
42   },
43   "pelis" : 209
44 }
```

21. 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.

**-Comando:**

```

9 // 22. Sobre genres (nueva colección), mostrar los 5 actores y los géneros en los que han participado con más número de
10 //géneros diferentes, se debe mostrar el número de géneros diferentes que ha interpretado. Importante! Se necesita
1 //previamente filtrar para descartar aquellos actores llamados "Undefined". Aclarar que no se eliminan de la colección,
2 //sólo que filtramos para que no aparezcan.
3
4 var filtro = { "cast": { "$ne": "Undefined" } }
5 var fase3 = { "$match": filtro }
6 var fase4 = { "$group": { "_id": "$cast", "numgeneros": { "$addToSet": "$genres" }, "generos": { "$addToSet": "$genres" } } }
7 var fase5 = { "$project": { "_id": 1, "numgeneros": { "$size": "$numgeneros" }, "generos": "$generos" } }
8 var fase6 = { "$sort": { "numgeneros": -1 } }
9 var fase7 = { "$limit": 5 }
10
11 var fases = [ fase3, fase4, fase5, fase6, fase7 ]
12 db.genres.aggregate(fases)
13

```

## -Resultado:

```

1 /* 1 */
2 {
3   "_id" : "Dennis Quaid",
4   "numgeneros" : 20,
5   "generos" : [
6     "Fantasy",
7     "Biography",
8     "Action",
9     "Drama",
10    "Adventure",
11    "Comedy",
12    "Sports",
13    "Crime",
14    "Thriller",
15    "Family",
16    "Musical",
17    "Disaster",
18    "Suspense",
19    "Horror",
20    "Western",
21    "Satire",
22    "Animated",
23    "Dance",
24    "Romance",
25    "Science Fiction"
26  ],
27 },
28
29 /* 2 */
30 {
31   "_id" : "Michael Caine",
32   "numgeneros" : 19,
33   "generos" : [
34     "Spy",
35     "Biography",
36     "Superhero",
37     "Action",
38     "Science Fiction",
39     "Mystery",
40     "Crime",
41     "Comedy",
42     "Family",
43     "Thriller",
44     "Undefined",
45     "Adventure",
46     "War",
47   ]
48 }

```

_id	numgeneros	generos
1 Dennis Quaid	20	Array[20]
2 Michael Caine	19	Array[19]
3 James Mason	19	Array[19]
4 Gene Hackman	18	Array[18]
5 Danny Glover	18	Array[18]

**22. 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.**

## -Comando:

```

5 // 23. Sobre genres (nueva colección), mostrar las 5 películas y su año correspondiente en los que más géneros diferentes
7 //han sido catalogados, mostrando esos géneros y el número de géneros que contiene.
8
9 var filtro = { "genres": { "$ne": "Undefined" }}
10 var fase1 = { "$match": filtro }
11 var fase2 = { "$group": { "_id": { "title": "$title", "year": "$year" }, "genres": { "$addToSet": "$genres" }}}
12 var fase3 = { "$addFields": { "numgeneros": { "$size": "$genres" }}}
13 var fase4 = { "$sort": { "numgeneros": -1 }}
14 var fase5 = { "$limit": 5 }
15
16 var fases = [fase1, fase2, fase3, fase4, fase5]
17 db.genres.aggregate(fases)
18

```

### -Resultado:

```

1 /* 1 */
2 {
3   "_id" : {
4     "title" : "American Made",
5     "year" : 2017
6   },
7   "genres" : [
8     "Thriller",
9     "Historical",
10    "Crime",
11    "Biography",
12    "Action",
13    "Comedy",
14    "Drama"
15  ],
16   "numgeneros" : 7
17 },
18
19 /* 2 */
20 {
21   "_id" : {
22     "title" : "Thor: Ragnarok",
23     "year" : 2017
24   },
25   "genres" : [
26     "Action",
27     "Adventure",
28     "Comedy",
29     "Superhero",
30     "Science Fiction",
31     "Fantasy"
32   ],
33   "numgeneros" : 6
34 },
35
36 /* 3 */
37 {
38   "_id" : {
39     "title" : "Dunkirk",
40     "year" : 2017
41   },
42   "genres" : [
43     "Drama",
44     "Thriller",
45     "War",
46     "Action",
47     "Historical",
48     "Adventure"
49   ],
50   "numgeneros" : 6
51 },
52

```

```

5 /* 3 */
6 {
7   "_id" : {
8     "title" : "Dunkirk",
9     "year" : 2017
10  },
11  "genres" : [
12    "Drama",
13    "Thriller",
14    "War",
15    "Action",
16    "Historical",
17    "Adventure"
18  ],
19  "numgeneros" : 6
20 },
21
22 /* 4 */
23 {
24   "_id" : {
25     "title" : "My Little Pony: The Mov
26     "year" : 2017
27   },
28   "genres" : [
29     "Family",
30     "Musical",
31     "Adventure",
32     "Comedy",
33     "Animated",
34     "Fantasy"
35   ],
36   "numgeneros" : 6
37 },
38
39 /* 5 */
40 {
41   "_id" : {
42     "title" : "The Dark Tower",
43     "year" : 2017
44   },
45   "genres" : [
46     "Adventure",
47     "Horror",
48     "Western",
49     "Fantasy",
50     "Science Fiction",
51     "Action"
52   ],
53   "numgeneros" : 6
54 },
55

```

### 23. Mostrar los 5 géneros con más películas diferentes por año, mostrando el número total de películas por género:

#### -Comando:



```

30 //24.Mostrar los 5 géneros con más películas diferentes por año, mostrando el número total de películas por género:
31 var filtro = { "genres": { "$ne": "Undefined" }}
32 var pipeline = [
33     // Filtrar para descartar documentos con genres Undefined
34     { "$match": filtro },
35     // Agrupar por año y género, contando el número total de películas por género
36     { "$group": { "_id": { "year": "$year", "genre": "$genres" }, "total_movies": { "$sum": 1 } }},
37     // Ordenar los resultados por número total de películas en orden descendente
38     { "$sort": { "total_movies": -1 } },
39     // Limitar los resultados a los primeros 5 géneros por año
40     { "$group": { "_id": "$_id.year", "top_genres": { "$push": { "genre": "$_id.genre", "total_movies": "$total_movies" } } }},
41     // Proyectar los resultados para mostrar solo los 5 géneros por año
42     { "$project": { "_id": 0, "year": "$_id", "top_genres": { "$slice": [ "$top_genres", 5 ] } } }
43 ]
44 db.genres.aggregate(pipeline)
45

```

**-Resultado:**

```

1  /* 1 */
2  {
3      "year" : 2003,
4      "top_genres" : [
5          {
6              "genre" : "Comedy",
7              "total_movies" : 287
8          },
9          {
10             "genre" : "Drama",
11             "total_movies" : 208
12          },
13          {
14             "genre" : "Crime",
15             "total_movies" : 116
16          },
17          {
18             "genre" : "Action",
19             "total_movies" : 116
20          },
21          {
22             "genre" : "Romance",
23             "total_movies" : 85
24          }
25      ]
26  },
27
28  /* 2 */
29  {
30      "year" : 1975,
31      "top_genres" : [
32          {
33             "genre" : "Drama",
34             "total_movies" : 124
35          },
36          {
37             "genre" : "Comedy",
38             "total_movies" : 81
39          },
40          {
41             "genre" : "Crime",
42             "total_movies" : 43
43          },
44          {
45             "genre" : "Action",
46             "total_movies" : 42
47          },
48          {
49             "genre" : "Western",
50             "total_movies" : 32
51          }
52      ]
53  }
54

```

**24. Agrupar por película y año mostrando las 5 en las que menos actores haya participado, mostrando el número total de actores.**

**-Comando:**

```

9 //25. Agrupar por película y año mostrando las 5 en las que menos actores hayan
9 //participado, mostrando el número total de actores.
1
2 var filtro = {"cast":{"$ne": "Undefined"}}
3 var fase2= {"$match":filtro}
4 var fase3={"$group":{"_id":{"title":"$title","year":"$year" },"cuenta":{"$sum":1}}}
5 var fase4= {"$sort":{"cuenta":1}}
6 var fase5= {"$limit": 5}
7
8
9 var fases=[ fase2, fase3, fase4, fase5])
9 db.actors.aggregate(fases)

```

#### -Resultado:

```

/* 1 */
{
  "_id" : {
    "title" : "Paratroop Command",
    "year" : 1959
  },
  "cuenta" : 1
},
/* 2 */
{
  "_id" : {
    "title" : "Favela Rising",
    "year" : 2005
  },
  "cuenta" : 1
},
/* 3 */
{
  "_id" : {
    "title" : "Gran Torino",
    "year" : 2008
  },
  "cuenta" : 1
},
/* 4 */
{
  "_id" : {
    "title" : "The Witness",
    "year" : 1992
  },
  "cuenta" : 1
},
/* 5 */
{
  "_id" : {
    "title" : "Penn & Teller Get Killed",
    "year" : 1989
  },
  "cuenta" : 1
}

```

25. Lista de los 100 actores con carrera más larga, año de inicio, final, y géneros realizados.

#### -Comando:

```

61 //26. Lista de los 100 actores con carrera mas larga, año de inicio, final, y generos realizados.
62
63
64 var filtro = { "cast": { "$ne": "Undefined" }, "genres": { "$ne": "Undefined" }}
65 var fase2 = { "$match": filtro }
66 var fase3 = { "$group": { "_id": "$cast", "primer_año": { "$min": "$year" }, "ultimo_año": { "$max": "$year" }, "genres": { "$addToSet": "$genres" }}}
67 var fase4 = { "$project": { "_id": 1, "primer_año": 1, "ultimo_año": 1, "genres": 1, "años_totales": { "$subtract": [ "$ultimo_año", "$primer_año" ] }}}
68 var fase5 = { "$sort": { "años_totales": -1 } }
69 var fase6 = { "$limit": 100 }
70
71 var fases = [ fase2, fase3, fase4, fase5, fase6 ]
72 db.genres.aggregate(fases)
73
74

```

## -Resultado:

```

/* 1 */
{
  "_id" : "Harrison Ford",
  "primer_año" : 1919,
  "ultimo_año" : 2017,
  "genres" : [
    "Thriller",
    "Noir",
    "Western",
    "Crime",
    "Adventure",
    "Action",
    "Comedy",
    "Sports",
    "Drama",
    "Science Fiction",
    "Romance",
    "Mystery",
    "Historical"
  ],
  "años_totales" : 98
},

/* 2 */
{
  "_id" : "Gloria Stuart",
  "primer_año" : 1932,
  "ultimo_año" : 2012,
  "genres" : [
    "Historical",
    "Horror",
    "Crime",
    "Musical",
    "Mystery",
    "Drama",
    "Disaster",
    "Adventure",
    "Comedy",
    "Romance",
    "Science Fiction",
    "Biography"
  ],
  "años_totales" : 80
},

/* 3 */
{
  "_id" : "Lillian Gish",
  "primer_año" : 1912,
  "ultimo_año" : 1987,
  "genres" : [

```